Überblick

Das Verfahren von Nelder-Mead

- Die bekannteste ableitungsfreie Methode ist das **Downhill-Simplex-Verfahren** von Nelder und Mead (auch Polyeder-Verfahren oder Simplexverfahren).
- Das Verfahren hat nichts mit dem Simplexverfahren für lineare Optimierungsprobleme zu tun.
- Grundprinzip des Verfahrens ist die Konstruktion von Simplizes.
- 1965: Erste Version des Verfahrens von Nelder und Mead
  1998: Modifikation des Verfahrens und Matlab-Implementierung von Lagarias u.a.

Simplex

Ein Simplex hat immer eine Ecke mehr als die Dimension, d.h. in 2D ist es ein Dreieck, in 3D ein Tetraeder, usw.

Definition (Simplex)

Zu $n+1$ Punkten $x_0, x_1, \dots, x_n \in \mathbb{R}^n$ $(n \in \mathbb{N})$,
wobei $\{x_i - x_0\}_{i=1,\dots,n}$ linear unabhängig,
heißt die transitive Hülle

$$S = \left\{ \sum_{i=0}^{n} \lambda_i x_i \;\middle|\; \lambda_i \geq 0,\ i = 0, 1, \dots, n, \quad \sum_{i=0}^{n} \lambda_i = 1 \right\}$$

(n-dimensionales) **Simplex** mit den Ecken $x_0, x_1, \dots, x_n$.

Grundidee des Verfahrens

- Ziel: Bestimmung eines Minimums von $f(x)$
- Ausgehend von einem Startsimplex wird eine Folge von Simplizes konstruiert.
- Es wird immer derjenige Eckpunkt des Simplex mit dem größten Funktionswert durch einen besseren Punkt ersetzt.
- Die Folge bewegt sich in Richtung eines lokalen Minimums.
- In der Umgebung des Minimums werden die Simplizes immer kleiner und konvergieren schließlich gegen das Minimum.

Algorithmus (Nelder-Mead-Verfahren)

1. Wähle einen Startpunkt $x^{[0,0]} \in \mathbb{R}^n$ und bestimme dazu mit Hilfe der Einheitsvektoren e_j für $j = 1, \ldots, n$ die Ecken des Startsimplex $x^{[0,j]} = x^{[0,0]} + e_j$. Setze $k = 0$.
2. Bestimme zum aktuellen Simplex den Eckpunkt $x^{[k,m]}$ mit dem größten Funktionswert
$$f(x^{[k,m]}) = \max\{f(x^{[k,0]}, \ldots, f(x^{[k,n]})\}$$
3. Verbessere den schlechtesten Funktionswert durch eine Reflexion am Schwerpunkt der übrigen Punkte $s_k = \frac{1}{n} \sum_{i=0, i\neq m}^{n} x^{[k,i]}$, d.h. ersetze die Ecke $x^{[k,m]}$ im Simplex durch einen anderen Punkt mit niedrigerem Funktionswert. Dabei verwendet man das Verfahren auf den folgenden Folien mit $x_k = x^{[k,m]}$, beginnend mit Schritt *a*
4. Falls Abbruchbedingung nicht erfüllt, setze $k := k + 1$ und gehe zu 2.

Durchführung von Schritt 3

a **Reflexion:** Berechnung einer Spiegelung von x_k an s_k:

$$\hat{x}_k = s_k + \gamma(s_k - x_k) \quad (0 < \gamma \leq 1)$$

Ist $\hat{x}_k$ besser als der zweitschlechteste Punkt, aber schlechter als der beste, so nimmt man $\hat{x}_k$ als neuen Wert für x_k; ansonsten:

b **Expansion:** Ist $\hat{x}_k$ besser als alle anderen Punkte, so versucht man, $\hat{x}_k$ auf der Geraden weiter zu schieben:

$$\hat{x}_k^* = s_k + \beta(\hat{x}_k - s_k) \quad (\beta > 1)$$

Man wählt den “Besseren” von $\hat{x}_k$ und $\hat{x}_k^*$ als neues x_k.

c **Innere partielle Kontraktion:** Ist $\hat{x}_k$ schlechter als x_k, so sucht man innerhalb des Simplex und geht auf der Geraden (durch x_k und s_k) nicht einmal bis s_k:

$$\hat{x}_k = s_k + \alpha(x_k - s_k) \quad (0 < \alpha < 1)$$

Ist $\hat{x}_k$ besser als x_k, so nimmt man es als nächstes x_k, ansonsten nimmt man die Variante *e*.

Durchführung von Schritt 3

d **Äußere partielle Kontraktion:** Ist $\hat{x}_k$ zwar besser als x_k, aber schlechter als alle anderen n Punkte, so sucht man näher am Simplex, d.h. man bestimmt:

$$\hat{x}_k{}^* = s_k + \alpha(\hat{x}_k - s_k) \quad (0 < \alpha < 1)$$

Ist $\hat{x}_k{}^*$ besser als x_k, so nimmt man es als neuen Wert für x_k, ansonsten nimmt man die Variante *e*.

e **Totale Kontraktion:** x_k bleibt unverändert und alle anderen Punkte bekommen den halben Abstand zu x_k:

$$\hat{x}_i = x_i + \frac{1}{2}(x_k - x_i) \quad (i = 0, 1, \dots, n \ \text{ mit } \ i \neq k)$$

Abbruchkriterien

Schritt 3 wird durchgeführt bis

- der Durchmesser des Simplex und der maximale Abstand der Funktionswerte kleiner als eine gegebene Toleranz ist (Implementierung von `fminsearch` in Matlab)

  oder

- die Standardabweichung der Funktionswerte an den Ecken des Simplex kleiner als eine vorgegebene Toleranz *tol* ist, d.h.

$$\sqrt{\frac{1}{n+1}\sum_{i=0}^{n}(f(x_i)-\bar{f})^2} < tol$$

  mit dem Mittelwert $\bar{f}$ der $n+1$ Funktionswerte (Vorschlag von Nelder und Mead).

Konvergenz des Verfahrens

- Folgende Werte für die Konstanten haben sich in der Praxis bewährt, müssen aber nicht immer optimale Ergebnisse liefern:
  - $\gamma = 1$ (Spiegelung)
  - $2 \leq \beta \leq 3$ (Expansion)
  - $0.4 \leq \alpha \leq 0.6$ (Kontraktion)
- Es gibt keinen allgemeinen Konvergenzbeweis (nur für einzelne niedrigdimensionale Funktionsklassen).
- Es gibt aber einzelne Beispiele für Konvergenz gegen nicht-stationäre Punkte (z.B. von McKinnon).
- Divergenz gegen Unendlich ist möglich.

Dennoch ist das Verfahren wegen seiner breiten Einsetzbarkeit und effizienten Implementierung sehr beliebt.

Weitere ableitungsfreie Verfahren

- Genetische Algorithmen,
  z.B. Mutations-Selektions-Verfahren, Simulated Annealing
- Interpolationsmethoden
- Verfahren mit Koordinaten- und Mustersuche
  in 1D z.B. Bisektionsverfahren, Verfahren des Goldenen Schnitts
- Verfahren der konjugierten Richtungen

Abstiegsverfahren und Schrittweitensteuerung

Definition (Abstiegsrichtung)

Seien $f : \mathbb{R}^n \to \mathbb{R}$ und $x \in \mathbb{R}^n$.

Eine Suchrichtung $d \in \mathbb{R}^n$ heißt **Abstiegsrichtung von** f **in** x, falls es ein $\bar{\alpha} > 0$ gibt mit

$$f(x + \alpha d) < f(x) \quad \text{für alle } 0 < \alpha \leq \bar{\alpha}.$$

Lemma (Kriterium für Abstiegsrichtung)

Sei $f : \mathbb{R} \to \mathbb{R}$ stetig differenzierbar in x.

Gilt $\nabla f(x)^\top d < 0$,
so ist d eine Abstiegsrichtung von f in x.

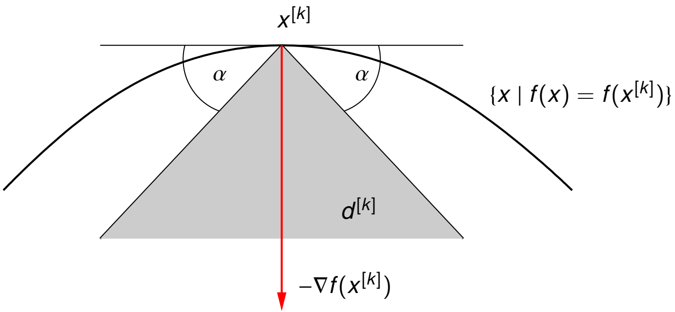
Winkelbedingung

Geometrische Interpretation des Lemma:

Winkel zwischen Gradient und Abstiegsrichtung zwischen 90° und 270°

Winkelbedingung (gradientenbezogene Abstiegsrichtung):

Für effiziente Algorithmen kommt sogar nur ein geeigneter grauer Kegel um den Gradienten für $d^{[k]}$ (mit $\alpha \in (0, 90°)$) in Frage, so dass man (unabhängig von k) von der Niveaulinie wegbleibt:



Abstiegsverfahren

Ziel: Verfahren zur Minimierung von $f \in C^1(\mathbb{R}^n)$
Idee: Kombiniere Abstiegsrichtung mit einer Schrittweitensteuerung

Algorithmus (*Allgemeines Abstiegsverfahren*)

(i) *Wähle Startpunkt* $x^{[0]} \in \mathbb{R}^n$.
Setze $i = 0$.

(ii) *Falls Abruchkriterium erfüllt, STOP.*

(iii) *Berechne Abstiegsrichtung* $d^{[i]}$ *und Schrittweite* $\alpha^{[i]} > 0$, *so dass*

$$f(x^{[i]} + \alpha^{[i]} d^{[i]}) < f(x^{[i]}).$$

Setze $x^{[i+1]} = x^{[i]} + \alpha^{[i]} d^{[i]}$.

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

Beispiele für Abstiegsverfahren

- Gradientenverfahren (Verfahren des steilsten Abstiegs):

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

  Abstiegsrichtung in nicht stationären Punkten.

  Naheliegend, aber nicht die effizienteste Wahl.

- Newton-Verfahren:

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

  Falls $H_f(x^{[i]})$ positiv definit, dann $d^{[i]}$ Abstiegsrichtung in nicht stationären Punkten.

Liniensuche/Vergleich von Abstiegsverfahren

Liniensuche zur Schrittweitenbestimmung entlang Linie

$$\varphi(\alpha) := f(x + \alpha d)$$

Beispiel

Minimiere $f(x_1, x_2) := x_1^2 + 10x_2^2$.

Exakte Liniensuche zur Bestimmung der Schrittweite:

$$\alpha := argmin\{\varphi(\alpha) \,|\, \alpha > 0\}$$

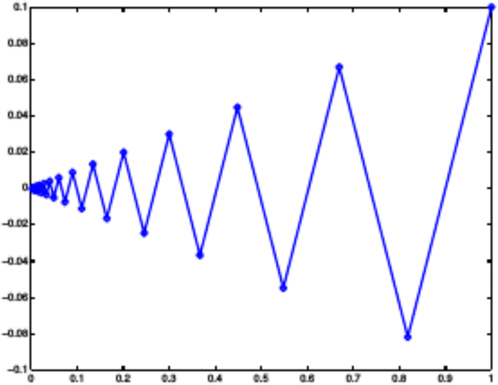
Abbruchkriterium $\|\nabla f(x)\|_2 \leq 10^{-5}$

Startvektor $x^{[0]} := (1, 0.1)^\top$

Gradientenverfahren benötigt (unskaliert) 63 Iterationen.

Newton-Verfahren benötigt 1 Iteration (und löst exakt).

Konvergenz des Gradientenverfahrens im Vergleich



Schrittweiten: Möglichkeiten zur Berechnung

Exakte Berechnung:

- bei quadratischen Problemen analytische Berechnung
- numerische Lösung des eindimensionalen Optimierungsproblems

Näherungsweise Berechnung:

- Armijo-Verfahren
- (Armijo-)Goldstein-Verfahren
- Wolfe-Powell-Verfahren

Armijo-Verfahren für Schrittweiten

Algorithmus (*Armijo-Regel*)

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

(ii) *Falls*

$$\varphi(\alpha) \leq \varphi(0) + \sigma\alpha\varphi'(0) \quad \textit{(Armijo-Bedingung)},$$

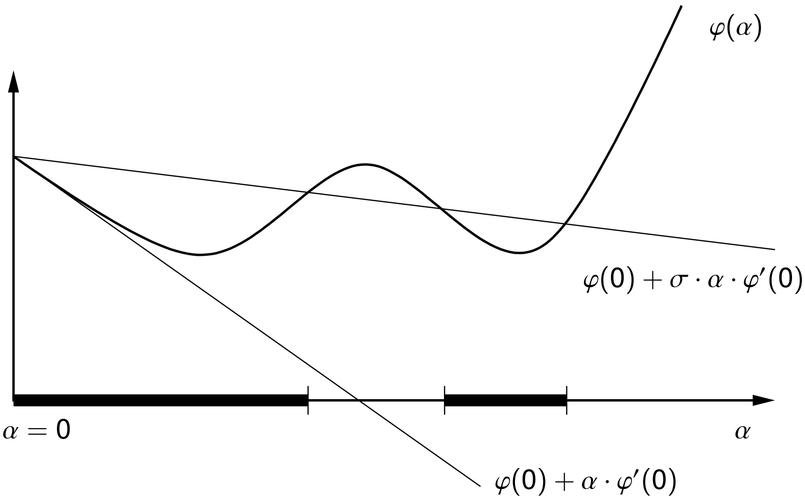
setze $\alpha^{[i]} := \alpha$ *und STOP.*

(iii) *Setze* $\alpha := \beta\alpha$.
Gehe zu (ii).

Die Armijo-Regel ist wohldefiniert, da $\varphi'(0) < 0$.

In der Praxis typischerweise: $\beta = 0.9$, $\sigma = 0.01$

Intervalle, die Armijo-Bedingung erfüllen



Eigenschaften des Armijo-Verfahrens

- Die Armijo-Regel führt häufig zu sehr kleinen Schrittweiten.
- Bessere Ergebnisse erhält man bei Verwendung der **Armijo-Regel mit Aufweitung**:
  Falls $\alpha^{[i]} = 1$, so multipliziert man so lange mit $1/\beta$, bis die Armijo-Bedingung nicht mehr erfüllt ist.

  Diese Regel wird in der Praxis eher selten verwendet.
- Stattdessen kombiniert man die Armijo-Regel häufig mit der **Wolfe-Powell-Regel** oder der **Goldstein-Bedingung**.

Wolfe-Powell (mit Armijo): Regel

Wolfe-Powell-Regel:

Zu vorgegebenem $\sigma \in (0, \frac{1}{2})$ und $\rho \in [\sigma, 1)$ bestimme die Schrittweite $\alpha^{[i]}$, so dass

(A) $\varphi(\alpha^{[i]}) \leq \varphi(0) + \sigma\,\alpha^{[i]}\,\varphi'(0)$

(W) $\varphi'(\alpha^{[i]}) \geq \rho\,\varphi'(0)$

Strenge Wolfe-Powell-Regel:

Zu vorgegebenem $\sigma \in (0, \frac{1}{2})$ und $\rho \in [\sigma, 1)$ bestimme die Schrittweite $\alpha^{[i]}$, so dass

(A) $\varphi(\alpha^{[i]}) \leq \varphi(0) + \sigma\,\alpha^{[i]}\,\varphi'(0)$

(W) $|\varphi'(\alpha^{[i]})| \leq -\rho\,\varphi'(0)$

(A) ist die Armijo-, (W) die Wolfe-Powell Bedingung.

Wolfe-Powell (mit Armijo): Algorithmus

Sei $\gamma > 1$ und $\alpha^{[i]}$ ein Startwert.

Expansionsphase: (Vergrößere $\alpha^{[i]}$, bis (A) nicht mehr gilt)

- Falls (A) und (W) für $\alpha^{[i]}$ erfüllt, dann STOP.
- Falls (A) für $\alpha^{[i]}$ erfüllt und (W) nicht, wiederhole die Expansionsphase mit $\alpha^{[i]} = \gamma\,\alpha^{[i]}$.
- Falls (A) für $\alpha^{[i]}$ nicht erfüllt, gehe zur Kontraktionsphase.

Setze $a = 0$ und $b = \alpha^{[i]}$

Kontraktionsphase: (Reduziere das Intervall, bis beide Bedingungen gelten) Bestimme $\alpha^{[i]} = \frac{a+b}{2}$:

- Falls (A) und (W) für $\alpha^{[i]}$ erfüllt, dann STOP.
- Falls (A) für $\alpha^{[i]}$ erfüllt und (W) nicht, wiederhole die Kontraktionsphase mit $a = \alpha^{[i]}$ und $b = b$.
- Falls (A) für $\alpha^{[i]}$ nicht erfüllt, wiederhole die Kontraktionsphase mit $a = a$ und $b = \alpha^{[i]}$.

Armijo-Goldstein

Zu vorgegebenem $\sigma \in (0, \frac{1}{2})$ und $\rho \in (\sigma, 1)$ bestimme die Schrittweite $\alpha^{[i]}$, so dass

$$\varphi(\alpha^{[i]}) \leq \varphi(0) + \sigma \, \alpha^{[i]} \, \varphi'(0)$$

$$\varphi(0) + (1 - \sigma)\alpha^{[i]} \, \varphi'(0) \leq \varphi(\alpha^{[i]})$$

Beachte: Die Armijo-Goldstein-Bedingung führt nicht immer zu einer Schrittweite in der Nähe des Minimums.

Effiziente Schrittweitenstrategie

- Bei einer gradientenbezogenen Suchrichtung $d^{[k]}$ gilt für hinreichend kleine $\alpha^{[k]}$ immer $f(x^{[k]} + \alpha^{[k]} d^{[k]}) < f(x^{[k]})$.
  Dies alleine reicht nicht für die Konvergenz von Abstiegsverfahren. Man benötigt zusätzlich eine effiziente Schrittweitenstrategie.

- Eine Schrittweitenstrategie heißt **effizient**, wenn es zu einem $x^{[0]}$ eine von x mit $f(x) \leq f(x^{[0]})$ und allen gradientenbezogenen Richtungen d unabhängige Konstante $c > 0$ gibt, so dass für die zu x und d berechnete Schrittweite α gilt:

$$f(x + \alpha d) \leq f(x) - c \left(\frac{\nabla f(x)^\top d}{\|d\|} \right)^2$$

Beispiele für effiziente Schrittweitenstrategien

- Die exakte Schrittweitenberechnung beim Gradientenverfahren und beim CG-Verfahren (siehe unten) ist effizient.
- Die Armijo-Regel ist im Allgemeinen nicht effizient.
- Sowohl die Schrittweitenstrategien nach Wolfe-Powell als auch nach Armijo-Goldstein sind effizient.

Winkelbedingung & Effizienz "=" Konvergenz

Theorem (Konvergenz von Abstiegsverfahren)

*Sei $f(x)$ eine stetig differenzierbare Funktion. Angenommen, man verwendet zur lokalen Minimierung von f ein Abstiegsverfahren, das in jedem Interationsschritt eine **gradientenbezogene** Abstiegsrichtung und eine **effiziente Regel** zur Schrittweitenbestimmung benutzt.*
*Dann **konvergiert** das Verfahren gegen ein lokales Minimum von f.*

Folgerungen:

- Das Gradientenverfahren und das CG-Verfahren konvergieren für beliebige stetig differenzierbare Funktionen f, falls man die Armijo-Goldstein oder die strenge Wolfe-Powell Schrittweitenverfahren benutzt.
- Die Armijo-Regel alleine reicht nicht aus (→Gegenbeispiele).

Herleitung von Abstiegsverfahren

Vorgehensweise:

- Entwickle Verfahren zunächst für **quadratische** Probleme mit positiv definiter Matrix (exakte Schrittweitenbestimmung möglich, leichtere Konvergenzaussagen, viele praktische Probleme sind quadratisch)
- Erweitere Verfahren auf **allgemeinere** Probleme (genäherte Schrittweitenbestimmung, schwierigere Konvergenzaussagen)

Verfahren:

- Gradientenverfahren (Verfahrens des steilsten Abstiegs)
- Konjugierte-Gradienten-Verfahren (CG)
- (Quasi-)Newton-Verfahren

Verfahren 1. Ordnung: Berechnung von Ableitungen

- **Symbolische Berechnung:**
  Explizite Berechnung der Ableitungen als Funktion:
  - von Hand oder
  - mit Mathematica bzw. Symbolic Math Toolbox von Matlab (falls möglich, beste Variante)

- **Numerische Berechnung:**
  Verwendung von Differenzenquotienten
  bzw. numerischen Ableitungsformeln (vgl. Numerik)

- **Automatisches Differenzieren:**
  Berechnung der Ableitung von Funktionen, die als Prozedur gegeben sind (mit Hilfe der Kettenregel)

Gradientenverfahren: Idee

- Für jeden Punkt gilt: Der negative Gradient ist die Richtung des steilsten Abstiegs.
- Lokal betrachtet ist das also die beste Richtung bei der Suche nach einem Minimum.
- Die Größe des Schritts bestimmt man bei quadratischen Optimierungsproblemen exakt, bei allen anderen genähert.
- Dabei wird auf der Linie des Gradienten der minimale Funktionswert gesucht.
- Dieses Verfahren wiederholt man mit dem dabei gefundenen Punkt usw. ...

Gradientenverfahren: Algorithmus

Algorithmus (*Gradientenverfahren mit Liniensuche*)

(0) *Wähle* $x^{[0]} \in \mathbb{R}^n$.
Wähle Schrittweiten(reduktions)faktor $\beta \in (0, 1)$ *und einen Steigungsfaktor* $\sigma \in (0, 1)$ *z.B. für Armijo-Regel.*
Setze $k := 0$.

(i) *Falls*

$$\|\nabla f(x^{[k]})\| \approx 0,$$

dann STOP.

(ii) *Berechne Suchrichtung* $d^{[k]} = -\nabla f(x^{[k]})$.

(iii) *Bestimme Schrittweite* $\alpha^{[k]}$, *z.B. mit der Armijo-Regel.*

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

Manchmal auch: $d^{[k]} = -\frac{\nabla f(x^{[k]})}{\|\nabla f(x^{[k]})\|}$

Abbruchkriterien

Die Abbruchbedingung $\nabla f(x^{[k]}) = 0$ kann man so nicht berechnen.

Je nach Problemstellung sind folgende Kriterien für eine gewünschte/mögliche Genauigkeit ϵ sinnvoll:

1. $f(x^{[k]}) - f(x^{[k+1]}) \leq \epsilon \cdot \max\{1, |f(x^{[k]})|\}$
   (oft auch relativer statt absolutem Fehler verwendet)
2. $\|x^{[k+1]} - x^{[k]}\| \leq \sqrt{\epsilon} \cdot \max\{1, \|x^{[k]}\|\}$
   (manchmal auch zusätzlich zu 1.)
3. $\|\nabla f(x^{[k]})\| \leq \sqrt[3]{\epsilon} \cdot \max\{1, \|x^{[k]}\|\}$
   (als Ersatz für das theoretische Kriterium $\nabla f(x^{[k]}) = 0$)

   Die theoretisch begründbare Wurzel $\sqrt{\epsilon}$ stellt sich in der Praxis als zu restriktiv heraus.

Konvergenz des Gradientenverfahrens

Theorem (Konvergenzsatz für Gradientenverfahren)

Wird durch das Gradientenverfahren mit Liniensuche für $f \in C^1(\mathbb{R}^n)$, $x^{[0]} \in \mathbb{R}^n$ eine nicht abbrechende Folge $\{x^{[k]}\}_{k\in\mathbb{N}}$ definiert, so gilt für jeden Häufungspunkt x^* dieser Folge $\nabla f(x^*) = 0$, d.h. x^* ist ein stationärer Punkt von f.

Vor-/Nachteile:

- Einfach zu implementieren
- Nur **lineare Konvergenz**:
  Unter Annahmen gilt

$$f(x^{[k+1]}) - f(x^*) \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^2 (f(x^{[k]} - f(x^*)),$$

  wobei $\kappa := \lambda_{max}/\lambda_{min}$, der Quotient des größten durch den kleinsten Eigenwert von $H_f(x^*)$.
- Die **Backpropagation** im maschinellen Lernen basiert auf diesem Verfahren.

Wh.: Konvergenzgeschwindigkeiten v. Folgen

Definition ((Super-)Lineare und quadratische Konvergenz)

Eine Folge $\{x^{[i]}\}_{i\in\mathbb{N}}$ **konvergiert** (mindestens)

(a) **linear** gegen $\hat{x}$, falls es ein $c \in (0,1)$ gibt, so dass

$$\|x^{[i+1]} - \hat{x}\| \leq c\,\|x^{[i]} - \hat{x}\|$$

für alle hinreichend großen i,

(b) **superlinear** gegen $\hat{x}$, falls es eine Nullfolge $\{c_i\}_{i\in\mathbb{N}}$, $c_i \downarrow 0$ gibt, so dass

$$\|x^{[i+1]} - \hat{x}\| \leq c_i\,\|x^{[i]} - \hat{x}\| \quad \text{für alle } i \in \mathbb{N},$$

(c) **in Ordnung** $p \geq 2$ gegen $\hat{x}$, falls es ein $c > 0$ gibt, so dass

$$\|x^{[i+1]} - \hat{x}\| \leq c\,\|x^{[i]} - \hat{x}\|^p \quad \text{für alle } i \in \mathbb{N},$$

d.h.

$$\|x^{[i+1]} - \hat{x}\| = O(\|x^{[i+1]} - \hat{x}\|^p).$$

Im Fall $p = 2$ heißt die Folge quadratisch konvegent.

Gradientenverfahren für QOP

$$min_{x \in \mathbb{R}^n} f(x) = \frac{1}{2} x^\top Q x + q^\top x$$

Algorithmus (*Gradientenverfahren für quadratische Optimierungsprobleme*)

(0) *Wähle* $x^{[0]} \in \mathbb{R}^n$.
Setze $k := 0$.

(i) *Falls*

$$\|\nabla f(x^{[k]})\| = \|Qx^{[k]} + q\| \approx 0,$$

dann STOP.

(ii) *Berechne Suchrichtung* $d^{[k]} = -\nabla f(x^{[k]})$.

(iii) *Bestimme exakte Schrittweite*

$$\alpha^{[k]} = \frac{(d^{[k]})^\top d^{[k]}}{(d^{[k]})^\top Q d^{[k]}}$$

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

Idee des konjugierten Gradientenverfahrens

Ideen zur Konstruktion des **CG-Verfahrens** *(conjugated gradients method)*:

- Optimale Schrittweite beim Verfahren des steilsten Abstiegs ist global nicht ideal.
- Bei einem n-dimensionalen Problem kann man theoretisch bei richtiger Wahl der Richtung und der Länge immer in n Schritten fertig sein.
- n aufeinander senkrecht stehende Richtungen funktionieren, sind aber sehr schlecht zu berechnen.

  Bessere Wahl: Q-konjugierte Richtungen $d^{[k]}$
- Vorgehen: Verändere den negativen Gradienten so, dass alle Abstiegsrichtungen $d^{[0]}, d^{[1]}, d^{[2]}, \ldots$ Q-konjugiert sind, und bestimme die optimale Schrittweite.

Q-konjugierte Richtungen

Definition (Konjugierte Vektoren)

Sei $Q \in \mathbb{R}^{n \times n}$ eine symmetrische, positiv definite Matrix.

Dann heißen die Vektoren $d^{[0]}, d^{[1]}, \ldots, d^{[k]} \in \mathbb{R}^n$ mit $k < n$ **zueinander konjugiert bezüglich** Q oder Q-konjugiert,

- wenn keiner von ihnen der Nullvektor ist und
- wenn folgendes gilt:

$$(d^{[i]})^\top Q d^{[j]} = \langle Q d^{[i]}, d^{[j]} \rangle = 0 \quad \text{für alle } i \neq j.$$

Man kann zeigen, dass Q-konjugierte Vektoren für symmetrisch, positiv definite Q linear unabhängig sind.

Umsetzung im CG-Verfahren

- Im CG-Verfahren verwendet man die Richtungen $d^{[0]} = -\nabla f(x^{[0]})$ und

$$d^{[k+1]} = -\nabla f(x^{[k+1]}) + \frac{\|\nabla f(x^{[k+1]})\|^2}{\|\nabla f(x^{[k]})\|^2} d^{[k]}$$

- Lösung des eindimensionalen Optimierungsproblems für die optimale Schrittweite liefert

$$\alpha^{[k]} = -\frac{\nabla f(x^{[k]})^\top d^{[k]})}{(d^{[k]})^\top Q d^{[k]}}$$

CG-Verfahren: Algorithmus

Algorithmus (*Verfahren der konjugierten Gradienten*)

(0) *Wähle* $x^{[0]} \in \mathbb{R}^n$*,*
berechne $\nabla f(x^{[0]}) = Qx^{[0]} + q$ *und* $d^{[0]} = -\nabla f(x^{[0]})$*.*
Setze $k := 0$*.*

(i) *Falls* $\|\nabla f(x^{[k]})\| \approx 0$*, dann STOP.*

(ii) *Bestimme zu* $d^{[k]}$ *die exakte Schrittweite*

$$\alpha^{[k]} = -\frac{\nabla f(x^{[k]})^\top d^{[k]}}{(d^{[k]})^\top Q d^{[k]}}.$$

(iii) *Setze* $x^{[k+1]} := x^{[k]} + \alpha^{[k]} d^{[k]}$*.*
Bestimme

$$\nabla f(x^{[k+1]}) = \nabla f(x^{[k]}) + \alpha^{[k]} Q d^{[k]}$$

$$\beta^{[k]} = \frac{\|\nabla f(x^{[k+1]})\|^2}{\|\nabla f(x^{[k]})\|^2}$$

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

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

Eigenschaften des Verfahrens

Theorem (Konvergenz des CG-Verfahrens)

Ist Q symmetrisch, positiv definit, so konvergiert das CG-Verfahren in höchstens n Iterationen gegen das Minimum des zugehörigen quadratischen Optimierungsproblems.

Beweisidee: (vgl. W. Alt, Satz 4.9.2)

- Das Residuum $r_k = \nabla f(x^{[k]})^\top$ steht senkrecht auf allen bislang ermittelten Suchrichtungen $d^{[i]}$, d.h.

$$\nabla f(x^{[k]})^\top d^{[i]} = 0 \quad \text{für} \;\; 0 \le i < k.$$

- Die Gradienten an allen Näherungswerten $x^{[k]}$ stehen senkrecht aufeinander, d.h.

$$\nabla f(x^{[k]})^\top \nabla f(x^{[i]}) \quad \text{für} \;\; 0 \le i < k.$$

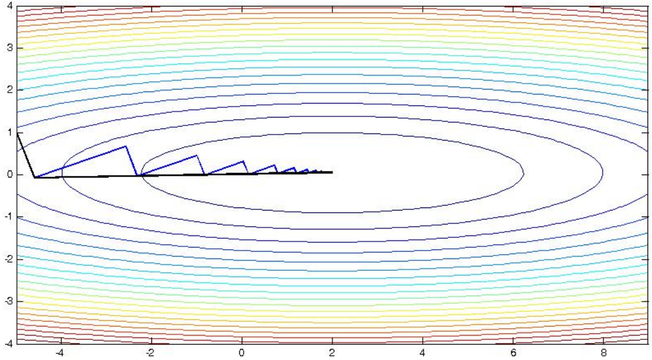
- Die Suchrichtungen sind Q-konjugiert.

Konvergenzaussagen

- Ist Q positiv definit, so konvergiert das CG-Verfahren in höchstens n Iterationen gegen das Minimum des zugehörigen quadratischen Optimierungsproblems.
- Theoretisch ist das CG-Verfahren also ein direktes Verfahren, das die exakte Lösung berechnet.
  In der Praxis führen Rundungsfehler dazu, dass die Berechnung näherungsweise (numerisch) erfolgt.
- Wie beim Gradientenverfahren konvergiert das CG-Verfahren umso schneller, je kleiner $\frac{\lambda_{max}}{\lambda_{min}}$ ist.

Gradienten- und CG-Verfahren beim Beispiel

$$f(x_1, x_2) = x_1^2 + 20x_2^2 - 4x_1 - 2x_2$$



Allgemeines CG-Verfahren: Algorithmus

Algorithmus (*Verfahren der konjugierten Gradienten nach Fletcher-Reeves*)

(0) *Wähle* $x^{[0]} \in \mathbb{R}^n$,
berechne $\nabla f(x^{[0]})$ *und* $d^{[0]} = -\nabla f(x^{[0]})$.
Setze $k := 0$.

(i) *Falls* $\|\nabla f(x^{[k]})\| \approx 0$, *dann STOP.*

(ii) *Bestimme zu* $d^{[k]}$ *die effiziente Schrittweite* $\alpha^{[k]}$, *z.B. mit dem Verfahren von Wolfe-Powell.*

(iii) *Setze* $x^{[k+1]} := x^{[k]} + \alpha^{[k]} d^{[k]}$.
Bestimme

$$\nabla f(x^{[k+1]})$$

$$\beta^{[k]} = \frac{\|\nabla f(x^{[k+1]})\|^2}{\|\nabla f(x^{[k]})\|^2}$$

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

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

Varianten des allgemeinen CG-Verfahrens

Folgende CG-Verfahren unterscheiden sich nur in der Berechnung von β:

- Fletcher-Reeves-Verfahren:

$$\beta^{[k]} = \frac{\|\nabla f(x^{[k+1]})\|^2}{\|\nabla f(x^{[k]})\|^2}$$

- Polak-Ribière-Verfahren:

$$\beta^{[k]} = \frac{(\nabla f(x^{[k+1]}) - \nabla f(x^{[k]}))^\top \nabla f(x^{[k+1]})}{\|\nabla f(x^{[k]})\|^2}$$

- Myers-Verfahren:

$$\beta^{[k]} = \frac{\|\nabla f(x^{[k+1]})\|^2}{\nabla f(x^{[k]})^\top d^{[k]}}$$

Vor-/Nachteile des CG-Verfahrens

- Wenige Rechenoperationen pro Iterationsschritt:
  - Keine Verwendung von Matrizen!
  - Funktions- und Gradientenauswertungen bei der Schrittweitenbestimmung
  - $\mathcal{O}(n)$ für Vektoroperationen
- Häufig langsamere Konvergenz, d.h. mehr Iterationen als bei Newton-artigen Verfahren
- $\Rightarrow$ Besonders geeignet für sehr große Optimierungsprobleme, d.h. viele Unbekannte
- Häufig Verbesserung durch “Restarts” möglich (d.h. Ersetzen der Suchrichtung durch negativen Gradienten)
- Bei (strengen) Wolfe-Powell-Schrittweiten kleines ρ wählen, also in der Nähe des Minimums suchen

Zusammenhang Optimierung - Lineare Gleichungssysteme

Die Lösung eines quadratischen Optimierungsproblems $\min f(x)$ mit

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

ist wegen

$$\nabla f(x) = (Qx + q)^\top$$

äquivalent zur Lösung des linearen Gleichungssystems

$$Qx = -q$$

Einige der Verfahren (insbesondere CG) werden häufig zur iterativen Lösung großer linearer Gleichungssysteme verwendet

$\Rightarrow$ Numerik 2

Newton-Verfahren: Motivation

3 verschiedene Arten der Herleitung:

- **Approximation**:
  Minimiere die lokale quadratische Approximation $\hat{f}$ von f um $x^{[i]}$ exakt:

$$\hat{f}(y) = f(x) + \nabla f(x)^\top (y-x) + \frac{1}{2}(y-x)^\top H_f(x)(y-x)$$

  Ist H_f pos. def., dann existiert genau ein Minimum $\hat{y}$ von $\hat{f}$:

$$\begin{aligned} \nabla \hat{f}(y) &= \nabla f(x) + H_f(x)(y-x) \stackrel{!}{=} 0 \\ \Leftrightarrow \quad d &= \hat{y} - x = -H_f(x)^{-1} \nabla f(x) \end{aligned}$$

- **Nullstellenverfahren**:
  Löse die notw. Bedingung $\nabla f(\hat{x}) = 0$ als Nullstellenproblem.
- **Abstiegsverfahren**:
  Man überprüft, dass d eine Abstiegsrichtung ist.

Die Voraussetzung $H_f(x)$ pos. def. ist durchaus eine Einschränkung! Diese ist nur in der Nähe des Minimums garantiert.

Lokales Newton-Verfahren

Algorithmus (*Lokales Newton-Verfahren*)

(i) *Wähle* $x^{[0]} \in \mathbb{R}^n$,
setze $i = 0$.

(ii) *Falls Abbruchkriterium erfüllt, dann STOP,*

(iii) *Berechne (falls möglich)* $d^{[i]}$ *als Lösung des LGS*

$$H_f(x^{[i]})d^{[i]} = -\nabla f(x^{[i]}).$$

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

Wenn H_f pos. def., lässt sich das lokale Newton-Verfahren als Abstiegsverfahren mit Schrittweite $\alpha^{[k]} = 1$ interpretieren.

Wh.: Lösung nichtlin. Gleichungssysteme

Für eine mehrdimensionale Funktion $g : \mathbb{R}^n \to \mathbb{R}^n$ bestimmt man die Linearisierung (Taylorreihe bis zum linearen Glied, "Tangentialhyperebene") an einer Stelle $x^{[i]} \in \mathbb{R}^n$:

$$g(x^{[i]}) + J_g(x^{[i]})\,(x - x^{[i]}) \approx \mathbf{0}\ .$$

Dabei bezeichnet $J_g(\xi) = J(g)(\xi)$ die Jacobi-Matrix von g an einer Stelle ξ.
Daraus ergibt sich für einen Startwert $x^{[0]} \in \mathbb{R}^n$ die Iterationsvorschrift des **Newton-Verfahrens**:

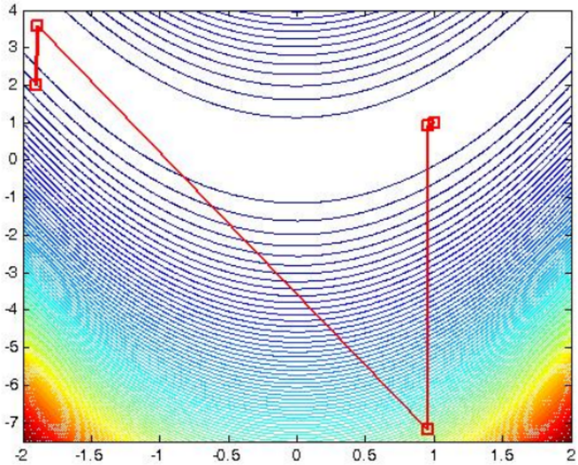
$$x^{[i+1]} = x^{[i]} - (J_g(x^{[i]}))^{-1} g(x^{[i]}) \qquad i = 0, 1, 2, \ldots$$

Ist die Jacobi-Matrix in der Umgebung von x^* regulär und Lipschitz-stetig, so konvergiert das Newton-Verfahren lokal quadratisch.

Newton-Verfahren - Beispiel

Angewendet auf das Beispiel der Rosenbrock-Funktion

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$



Konvergenz des Newton-Verfahrens

Theorem (Konvergenzsatz für Newton-Verfahren)

Besitze f ein (lokales) Minimum $\hat{x} \in \mathbb{R}^n$.

Es gibt ein $r > 0$, so dass, wenn durch das lokale Newton-Verfahren für $f \in C^2(\mathbb{R}^n)$, $x^{[0]} \in U_r(\hat{x})$ eine nicht abbrechende Folge $\{x^{[k]}\}_{k\in\mathbb{N}}$ definiert wird,

- *diese Folge superlinear konvergiert gegen $\hat{x}$.*
- *$\hat{x}$ ist das einzige lokale Minimum in $U_r(\hat{x})$.*
- *Erfüllt H_f zusätzlich die Lipschitz-Bedingung*

$$\|H_f(x) - H_f(y)\| \le L\,\|x - y\| \quad \text{für alle } x, y \in \mathbb{R}^n,$$

  so konvergiert diese Folge sogar quadratisch.

Globalisierung des Newton-Verfahrens

Globalisiertes (oder globales) Newton-Verfahren:

- Was macht man, wenn Newton-LGS nicht lösbar?
- Kombination mit dem Gradientenverfahren
- Verwendung einer kleineren Schrittweite (d.h. Dämpfung)
- *Weitere Option: Verwendung einer Bewertungsfunktion (für Schrittweite)*

Globalisiertes Newton-Verfahren

Algorithmus (*Global. Newton-Verfahren* [GK99, S. 92])

(i) *Wähle* $x^{[0]} \in \mathbb{R}^n$, $\sigma \in (0, 1/2)$, $\beta \in (0, 1)$, $\rho > 0$, $p > 2$, *setze* $i = 0$.

(ii) *Falls Abbruchkriterium erfüllt, dann STOP,*

(iii) *Berechne (falls möglich)* $d^{[i]}$ *als Lösung des LGS*

$$H_f(x^{[i]})d^{[i]} = -\nabla f(x^{[i]}).$$

Falls dieses LGS unlösbar oder $\nabla f(x^{[i]})^\top d^{[i]} > -\rho \|d^{[i]}\|^p$, *setze* $d^{[i]} = -\nabla f(x^{[i]})$.

(iv) *Bestimme Schrittweite* $\alpha^{[i]} > 0$ *(z.B. nach Armijo).*

(v) *Setze* $x^{[i+1]} := x^{[i]} + \alpha^{[i]} d^{[i]}$,
$i := i + 1$ *und*
gehe zu (ii).

Im Endlichdimensionalen kann man globale, superlineare Konvergenz gegen einen stationären Punkt beweisen.

Implementierungsdetails

- Lösen des linearen Gleichungssystems ist für kleine Gleichungssysteme mit dem Gauß-Algorithmus möglich, für große Systeme wird häufig das CG-Verfahren eingesetzt.
- Das Cholesky-Verfahren funktioniert nur für positiv definite Matrizen. Um mit evtl. nicht positiv definiten Matrizen rechnen zu können, kann eine modifizierte Version verwendet werden, die den Fall negativer Diagonalelemente berücksichtigt.
- Für Iterationen in der Nähe des Minimums ist die Schrittweite nach Armijo immer 1. Die lokal quadratische Konvergenz bleibt also erhalten.

Fazit Newton-Verfahren

Vorteile:

- Lokal quadratische Konvergenz, damit eines der schnellsten Verfahren
- Durch Globalisierung auch hinreichend robust

Nachteile:

- Die Hessematrix H_f wird in analytischer Form benötigt. Sie ist häufig entweder unbekannt oder schwer (unmöglich) zu bestimmen.
  Aufwand: $\mathcal{O}(n^2)$ Ableitungen analytisch oder numerisch ausrechnen, Gleichungssystem lösen in $\mathcal{O}(n^3)$ Operationen.
- Newton-Richtung ist i.A. keine Abstiegsrichtung, weil H_f nicht immer positiv definit ist.

Idee:

Ersetze H_f durch eine geeignete (z.B. positiv definite) und leichter zu bestimmende Matrix, in der Hoffnung, dass die Konvergenz immer noch superlinear ist.

Varianten des Newton-Verfahrens I

Hessematrix $H_f(x^{[i]})$ wird durch geeignete Matrix $A(x^{[i]})$ ersetzt, die in jeder Iteration durch "einfache" Rechenschritte bestimmt wird:

- Vereinfachtes Newton-Verfahren:

  $A = A(x^{[i]}) := H_f(x^{[0]})$ wird einmalig berechnet, wird einmal faktorisiert zur schnellen Lösung des Newton-LGS

- Finite-Differenzen-Approximation:

  Setze in Iteration i

  $$A(x^{[i]})_{jk} = \frac{\nabla f_j(x + h^{[i]} e_k) - \nabla f_j(x)}{h^{[i]}}, \quad j,k = 1, \ldots, n,$$

  wobei $h^{[i]} \to 0$ für $i \to \infty$

Varianten des Newton-Verfahrens II

- **Quasi-Newton-Verfahren**:

  Starte mit symm., pos. def. Matrix $A^{[0]}$
  Berechne $A^{[i+1]}$ rekursiv aus $A^{[i]}$ durch sogenannte Updateformeln

- Inverse Quasi-Newton-Verfahren:

  Wie Quasi-Newton-Verfahren, man approximiert aber $H_f^{-1}(x^{[i]})$

- Variable-Metrik-Verfahren ...

Quasi-Newton-Verfahren: Idee

Berechne die Richtung $d^{[k]}$ im allgemeinen Abstiegsverfahren mit Hilfe einer positiv definiten Matrix $A^{[k]}$ folgendermaßen:

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

bzw. löse das lineare Gleichungssystem

$$A^{[k]}d^{[k]} = -\nabla f(x^{[k]}) \ .$$

Wähle $A^{[k]}$ in jedem Schritt so, dass

- $A^{[k]}$ eine Approximation für $H_f(x^{[k]})$ ist.
- $A^{[k+1]}$ effizient aus $A^{[k]}$ berechnet werden kann.

Bestimme die Schrittweite $\alpha^{[k]}$

- durch exakte Rechnung bei quadratischen Optimierungsproblemen,
- mit dem Verfahren von Armijo-Goldstein oder Wolfe-Powell bei allgemeinen Problemen.

Quasi-Newton-Gleichung

Im Eindimensionalen geht man vom Newton-Verfahren über zum Sekantenverfahren durch Approximieren des Differentialquotienten $f''(x^{[k+1]})$ durch den Differenzenquotienten $\frac{f'(x^{[k+1]})-f'(x^{[k]})}{x^{[k+1]}-x^{[k]}}$.

Analog verwendet man im Mehrdimensionalen statt der 2. Ableitung, d.h. der Hesse-Matrix $H_f(x^{k+1})$ eine Matrix, die durch folgende Gleichung bestimmt ist:

$$A^{[k+1]}(x^{[k+1]} - x^{[k]}) = \nabla f(x^{[k+1]}) - \nabla f(x^{[k]})$$

Diese Gleichung heißt **Quasi-Newton-Gleichung** oder **Sekantenbedingung**. Sie bestimmt die Matrix $A^{[k+1]}$ aber nicht eindeutig, weil n Gleichungen $\frac{n(n+1)}{2}$ zu bestimmenden Matrix-Elementen gegenüberstehen.

Rang-1-Modifikationen

Idee zur Bestimmung von $A^{[k+1]}$:

- $A^{[k+1]}$ wird aus $A^{[k]}$ durch symmetrische Rang-1-Modifikationen berechnet, d.h.

$$A^{[k+1]} = A^{[k]} + c \cdot u \cdot u^\top \quad (c \in \mathbb{R}, u \in \mathbb{R}^n)$$

- In jeder Iteration ist das Gleichungssystem $A^{[k]} d^{[k]} = -\nabla f(x^{[k]})$ zu lösen oder die Inverse von $A^{[k]}$ zu berechnen.
- Bei Rang-1-Modifikationen kann die Inverse von $A^{[k+1]}$ effizient, d.h. in $O(n^2)$ berechnet werden, und zwar mit der Sherman-Morrison-Woodbury-Formel

$$(A + uv^\top)^{-1} = A^{-1} - \frac{A^{-1} u v^\top A^{-1}}{1 + v^\top A^{-1} u}$$

Bestimmung der Matrix $A^{[k+1]}$

Mit den Bezeichnungen

$$s^{[k]} = x^{[k+1]} - x^{[k]} \quad \text{und} \quad y^{[k]} = \nabla f(x^{[k+1]}) - \nabla f(x^{[k]})$$

erhält man $A^{[k+1]}$ durch zwei Rang-1-Modifikationen aus $A^{[k]}$:

$$A^{[k+1]} = A^{[k]} - \frac{A^{[k]} s^{[k]} (A^{[k]} s^{[k]})^\top}{(s^{[k]})^\top A^{[k]} s^{[k]}} + \frac{y^{[k]} (y^{[k]})^\top}{(y^{[k]})^\top s^{[k]}}$$

Die Matrix $A^{[k+1]}$

- ist symmetrisch und positiv definit,
- erfüllt die Quasi-Newton-Gleichung

$$A^{[k+1]} s^{[k]} = y^{[k]}.$$

BFGS-Verfahren

Verfahren von Broyden, Fletcher, Goldfarb, Shanno:

1. Wähle einen Startpunkt $x^{[0]} \in \mathbb{R}^n$ und eine symmetrische, positiv definite Matrix $A^{[0]}$. Setze $k = 0$.
2. Falls $\nabla f(x^{[k]}) \approx \mathbf{0}$, STOP.
3. Bestimme $(A^{[k]})^{-1}$ durch 2malige Anwendung der Sherman-Morrison-Woodbury-Formel
4. Berechne $d^{[k]} = -(A^{[k]})^{-1} \nabla f(x^{[k]})$
5. Bei Verwendung einer Schrittweitensteuerung: Berechne die Schrittweite $\alpha^{[k]}$ exakt bei quadratischen Optimierungsproblemen oder durch Wolfe-Powell bei allgemeinen Problemen.
6. Erhöhe k um 1 und gehe zu 2.

Eigenschaften der Matrix $A^{[k+1]}$

- Wenn $A^{[k]}$ symmetrisch positiv definit ist, ist auch $A^{[k+1]}$ symmetrisch positiv definit.
- Mit einem Rang-1-Update erhält man zwar eine symmetrische, aber keine positiv definite Matrix, da

$$\left(A^{[k]} - \frac{A^{[k]}s^{[k]}(A^{[k]}s^{[k]})^\top}{(s^{[k]})^\top A^{[k]}s^{[k]}}\right)s^{[k]} = 0\,.$$

  Sie erfüllt auch nicht die Quasi-Newton-Gleichung.
- Beim BFGS-Verfahren hat die Matrix $A^{[k+1]}$ die Eigenschaft

$$A^{[k+1]} = \min_A \|A - A^{[k]}\|$$

  über alle symmetrisch, positiv definiten Matrizen, die die Quasi-Newton-Gleichung erfüllen.
- Das Quasi-Newton-Verfahren konvergiert superlinear, aber nicht quadratisch.

Quasi-Newton-Verfahren: Implementierung

- Die Startmatrix $A^{[0]}$ muss positiv definit sein. Mögliche Matrizen sind
  - die Einheitsmatrix I_n (d.h. erster Schritt entlang negativem Gradienten)
  - die Matrix $\frac{(y^{[0]})^\top s^{[k]}}{(y^{[0]})^\top y^{[k]}} \cdot I_n$
- Typische Werte für die Schrittweitensuche nach Wolfe-Powell sind $\alpha^{[k]} = 1$, $\sigma = 10^{-4}$ und $\rho = 0.9$.
- Anstatt die Inverse mit der SMW-Formel auszurechnen, kann man das Gleichungssystem

$$A^{[k]} s^{[k]} = y^{[k]}$$

lösen, indem man aus der Cholesky-Zerlegung von $A^{[k-1]}$ direkt die Cholesky-Zerlegung von $A^{[k]}$ berechnet.

Fazit Quasi-Newton-Verfahren

- Erhebliche Reduktion des Rechenaufwands im Vergleich zum Newton-Verfahren:
  Keine Berechnung der Hessematrix und nur $O(n^2)$ Rechenoperationen pro Iterationsschritt
- Statt quadratischer Konvergenz immerhin noch superlineare Konvergenz
- Durch Rang-2 Updates bleibt Symmetrie und positive Definitheit der Matrizen $A^{[k]}$ erhalten. Damit erfolgt jeder Iterationsschritt entlang einer Abstiegsrichtung.

Weitere Quasi-Newton-Methoden

- **DFP-Methode:** (Davidon-Fletcher-Powell)
  ebenfalls Rang-2-Modifikationen, die die Sekantenbedingung erfüllen, allerdings wird hier die Inverse von $H^{[k]}$ approximiert

- **SR1-Methode:** (Symmetrische Rang-1-Modifikation)
  Matrizen erfüllen die Sekantenbedingung, sind aber nicht immer positiv definit;
  in vielen Fällen auch hiermit gute Ergebnisse

- **L-BFGS Methode:** (Limited memory BFGS)
  nicht die gesamte Matrix $A^{[k]}$ wird gespeichert, sondern nur einzelne Update-Vektoren;
  geeignet für sehr große Matrizen

Trust-Region-Verfahren: Idee

- Betrachte einen Vertrauensbereich (**trust region**) $B(x^{[k]}, \rho^{[k]})$ mit Radius $\rho^{[k]} > 0$, in dem die Zielfunktion f durch eine Modellfunktion $f^{[k]}(d)$ (typischerweise eine quadratische Entwicklung in d) approximiert wird.
- Trust-Region-Verfahren sind "dual" zur Liniensuche: Es wird zuerst eine Schrittweite (Größe der Trust Region) und dann eine Richtung $d^{[k]}$ aus

$$\min_{\|d\| \le \rho^{[k]}} f^{[k]}(d)$$

bestimmt, bei einer Liniensuche ist dies umgekehrt.

- Ist die Approximation adäquat (Abstieg passt zum Modell), dann wird die Trust Region erweitert, andernfalls wird die Trust Region kontrahiert.

Einordnung Trust-Region-Verfahren

- Newton-artiges Verfahren, da Taylor-Approximation 2. Ordnung
- Trust-Region-Verfahren werden auch als *restricted-step methods* bezeichnet.
- Interessant für Probleme mit vielen Variablen
- Matlab-Befehl `fminunc`
- Verwandtes Verfahren: Levenberg-Marquardt-Algorithmus

Rückblick: Lineare Ausgleichsrechnung

Gegeben:

- m Datenpunkte (t_i, y_i), z.B. aus einer Messung
- lineare Funktion mit n zu bestimmenden Parametern (typischerweise $m > n$)

$$g(t, x_1, \ldots, x_n) = g_1(t)x_1 + \cdots + g_n(t)x_n$$

Aufgabe:

Minimiere das Residuum

$$r(x_1, \ldots, x_n) = \begin{pmatrix} r_1(x_1,\ldots,x_n) \\ \vdots \\ r_m(x_1,\ldots,x_n) \end{pmatrix} = \begin{pmatrix} g_1(t_1)x_1 + \cdots + g_n(t_1)x_n - y_1 \\ \vdots \\ g_1(t_m)x_1 + \cdots + g_n(t_m)x_n - y_m \end{pmatrix},$$

bzw. bestimme $x_1, \ldots, x_n$, so dass

$$f(x) = \sum_{i=1}^{m} r_i(x_1, \ldots, x_n)^2 = \sum_{i=1}^{m} [g_1(t_i)x_1 + \cdots + g_n(t_i)x_n - y_i]^2$$

minimal wird.

Bedingung für ein Minimum: $\nabla f(x) = \mathbf{0}_n$

Mit

$$\nabla f(x) = 2 \cdot \begin{pmatrix} g_1(t_1)r_1 + g_1(t_2)r_2 + \cdots + g_1(t_m)r_m \\ \vdots \\ g_n(t_1)r_1 + g_n(t_2)r_2 + \cdots + g_n(t_m)r_m \end{pmatrix}$$

und

$$r'(x) = J_r(x) = \begin{pmatrix} g_1(t_1) & g_2(t_1) & \dots & g_n(t_1) \\ g_1(t_2) & g_2(t_2) & \dots & g_n(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ g_1(t_m) & g_2(t_m) & \dots & g_n(t_m) \end{pmatrix}$$

ergibt dies die **Normal(en)gleichung**

$$r'(x)^\top r'(x) x = r'(x)^\top y \,.$$

Die Normalengleichung kann also auch mit Hilfe des Gradienten hergeleitet werden...

(Nicht-)Lineare Ausgleichsrechnung: Ausfallzeiten u. Temperaturbelastung (Wh.)

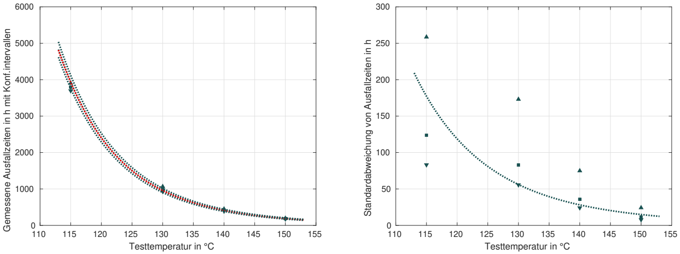


Abbildung: Ausgleichsrechnung über verschiedene Temperaturen T im Vergleich mit Messpunkten (Quadrate) samt Konfidenzintervallgrenzen zu $q = 90\%$ (Dreiecke). Links für $\mu_{krit}(T) \pm \sigma_{krit}(T)$, in Rot eine Referenzkurve. Rechts $\sigma_{krit}(T)$. [K., Dvorsky, Ließ 2018]

Modellbasierter Ansatz

$$f_{krit}(T) = t_{\ominus} + t_0 \exp\left(\left(\frac{T_a}{T - T_\infty}\right)^d\right)$$

Beispiel (Messung von Ausfallzeiten elektr. Bauteile)

T	[°C]	115	130	140	155
μ_{krit}	[h]	3791.62	987.74	439.66	189.94

Vermutung (modellbasierter Zugang): Arrhenius-Gesetz

$$\mu_{krit}(T) = t_0 \exp\left(\frac{T_a}{T - T_\infty}\right)$$

(μ_{krit} Ausfallzeit in h, t_0 reaktionskinetische Periodendauer in h, T Temperatur in K, T_a Aktivierungstemperatur in K, $T_\infty = 173.15$ Erstarrungstemperatur in K)

Lässt sich als affin-linearer Zusammenhang schreiben

$$ln(\mu_{krit}(T)) = ln(t_0) + \frac{T_a}{T - T_\infty} \qquad \Longleftrightarrow \qquad y(t) = x_1 + x_2 t$$

Durch Einsetzen der Messwerte ergibt sich ein lineares Gleichungssystem (LGS)

$$y_i = x_1 + x_2 t_i, \quad i = 1, \dots, 4.$$

Lin. Ausgleichsrechnung: Motivation & Ziele

Wir stellen fest:

- Im Allgemeinen mehr Messwerte (hier 4) als Parameter (hier 2)
- Messung eines Vorgangs fehlerbehaftet (auch ohne systematische Fehler)
- Überbestimmte LGS $\rightsquigarrow$ i.A. existiert keine Lösung

Ziele:

- Bestimme x_1, x_2 aus den Messdaten "bestmöglich"
- Allgemeine Methode hierzu

Wir sprechen von **linearer Ausgleichsrechnung** oder ***curve fitting*** im Englischen
bzw. **linearer Regression** (in der Statistik)

Spezialfall einer mathematischen Optimierungsmethode

Minimierungsproblem

Allgemein: $A \in \mathbb{R}^{m \times n}$, $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$

Idee: Bestimme $x = (x_1, \ldots, x_n)^\top$, so dass der Fehler im LGS

$$\|Ax - y\|_2 = \sqrt{\sum_{i=1}^{m} \left(\sum_{j=1}^{n} A_{ij} x_j - y_i \right)^2} \quad \text{bzw.} \quad \frac{1}{2} \|Ax - y\|_2^2$$

minimiert wird.

Das Minimum sei mit $\hat{x}$ bezeichnet.

Das Minimierungsproblem heißt **lineares Ausgleichsproblem** oder **Least-Squares-Problem.**

Methode der kleinsten Quadrate eigentlich **Methode der kleinsten Fehlerquadratsumme**
nach C. F. Gauß bzw. A.-M. Legendre (1805)

Allgemeine Problemstellung

Problem ($\star$) (Lin. Ausgleich als Minimierungsproblem)

Gegeben seien eine Matrix $A \in \mathbb{R}^{m \times n}$ und ein Vektor $y \in \mathbb{R}^m$ mit $m, n \in \mathbb{N}$.

Gesucht ist die Lösung $\hat{x} \in \mathbb{R}^n$ des Minimierungsproblems

$$\frac{1}{2}\|A\hat{x} - y\|_2^2 = \min_{x \in \mathbb{R}^n} \frac{1}{2}\|Ax - y\|_2^2 \qquad (\dagger).$$

- Falls $m = n$ und A invertierbar, dann hat $(\dagger)$ eine eindeutige Lösung.
- Der Fall $m > n$ ist in Anwendungen besonders relevant.
- Im Fall $m < n$ ist das LGS unterbestimmt. Es kann unlösbar sein, falls $Rang(A) \neq Rang(A \mid b)$

Auch andere Normen anstatt $\|\cdot\|_2$ können verwendet werden. Dann gestaltet sich die Berechnung von Lösungen u.U. schwieriger, da i.A. keine Differenzierbarkeit mehr vorliegt.

Gaußsche Normalgleichungen

Satz (Gaußsche Normalgleichungen)

$\hat{x}$ löst Problem ($\star$) genau dann, wenn die **Normalgleichungen**

$$A^\top A\,\hat{x} = A^\top y.$$

gelten.

$A^\top A \in \mathbb{R}^{n\times n}$ ist symmetrisch und positiv semi-definit.

Satz (Eindeutigkeit)

Sei $m \geq n$. Habe $A \in \mathbb{R}^{m\times n}$ maximalen Rang, d.h. $Rang(A) = n$.
Genau dann ist das Minimierungsproblem (†) bzw. sind die Normalgleichungen eindeutig lösbar.

Dann ist $A^\top A \in \mathbb{R}^{n\times n}$ invertierbar und positiv definit.

Nichtlineare Ausgleichsrechnung

Gegeben:

- m Datenpunkte (t_i, y_i), z.B. aus einer Messung
- nichtlineare Funktion mit n zu bestimmenden Parametern (typischerweise $m > n$)

$$g(t, x_1, \ldots, x_n)$$

Aufgabe:

Minimiere das Residuum

$$r(x_1, \ldots, x_n) = \begin{pmatrix} r_1(x_1,\ldots,x_n) \\ \vdots \\ r_m(x_1,\ldots,x_n) \end{pmatrix} = \begin{pmatrix} g(t_1,x_1,\ldots,x_n)-y_1 \\ \vdots \\ g(t_m,x_1,\ldots,x_n)-y_m \end{pmatrix},$$

bzw. bestimme $x_1, \ldots, x_n$ so, dass

$$f(x) = \sum_{i=1}^{m} r_i(x_1, \ldots, x_n)^2 = \sum_{i=1}^{m} [g(t_i, x_1, \ldots, x_n) - y_i]^2$$

minimal wird.

Nichtlin. Ausgleichsrechnung und Optimierung

Bedingung für ein Minimum:

$$\nabla f(x) = 2\cdot\begin{pmatrix} \frac{\partial g}{\partial x_1}(t_1)r_1 + \frac{\partial g}{\partial x_1}(t_2)r_2 + \cdots + \frac{\partial g}{\partial x_1}(t_m)r_m \\ \frac{\partial g}{\partial x_2}(t_1)r_1 + \frac{\partial g}{\partial x_2}(t_2)r_2 + \cdots + \frac{\partial g}{\partial x_2}(t_m)r_m \\ \vdots \\ \frac{\partial g}{\partial x_m}(t_1)r_1 + \frac{\partial g}{\partial x_m}(t_2)r_2 + \cdots + \frac{\partial g}{\partial x_m}(t_m)r_m \end{pmatrix} = 2r'(x)^\top r(x) \overset{!}{=} \mathbf{0}_n$$

Ausrechnen der Hesse-Matrix ergibt:

$$H_f(x) = 2\cdot r'(x)^\top r'(x) + 2\cdot\sum_{i=1}^{m} r_i(x) r_i''(x)$$

Lösung mit dem Newton-Verfahren: In jedem Schritt

- Lösen des Gleichungssystems $H_f(x^{[k]})d^{[k]} = \nabla f(x^{[k]})$
- Berechnen der nächsten Näherung $x^{[k+1]} = x^{[k]} - d^{[k]}$ (Beachte hier Vz. vor d!)

Gauß-Newton-Verfahren (Idee)

- Weil in der Nähe der Lösung das Residuum klein ist, vernachlässigt man bei der Hesse-Matrix den Summenanteil, d.h. man verwendet die Näherung

$$H_f(x) \approx 2 r'(x)^\top r'(x) \, .$$

- Im linearen Fall ist das die Normalengleichung.
- Im nichtlinearen Fall handelt es sich um die Normalengleichung eines linearen Ersatzproblems

$$\min_{d \in \mathbb{R}^n} \|r(x^{[k]}) + r'(x^{[k]}) d\|^2$$

anstelle von $\min_{x \in \mathbb{R}^n} \|r(x)\|^2$

- Man löst also in jedem Schritt ein lineares Ausgleichsproblem, das durch Taylorreihen-Entwicklung bis zur Ordnung 1 entsteht. Dies kann auch durch QR-Zerlegung erfolgen.

Gauß-Newton-Verfahren (Algorithmus)

1. Wähle einen Startpunkt $x^{(0)}$.
   Setze $k = 0$.
2. Falls $\nabla f(x^{[k]}) = 2r'(x^{[k]})^\top r(x^{[k]}) = \mathbf{0}$, dann STOP.
3. Bestimme die Suchrichtung $d^{[k]}$ durch Lösen entweder von

   $$r'(x^{[k]})^\top r'(x^{[k]}) d^{[k]} = r'(x^{[k]})^\top r(x^{[k]})$$

   oder des überbestimmten Systems

   $$r'(x^{[k]}) d^{[k]} = r(x^{[k]})$$

   mit der QR-Zerlegung.
4. Falls gedämpftes Verfahren verwendet wird, bestimme eine effiziente Schrittweite $\alpha^{[k]}$ nach Armijo-Goldstein oder Wolfe-Powell (ansonsten $\alpha^{[k]} = 1$) und berechne damit

   $$x^{[k+1]} = x^{[k]} - \alpha^{[k]} d^{[k]}$$

5. Setze $k = k + 1$ und
   gehe zu 2.

Normalgleichungen

Algorithmus (Ein erster Algorithmus für Normalgleichungen)

(1) Berechne $r'(x)^\top r'(x)$ und $r'(x)^\top r(x)$

(2) Bestimme Cholesky-Zerlegung $r'(x)^\top r'(x) = L(x)L^\top(x)$

(3) Vorwärts-Rückwärts-Einsetzen

$$L(x)z = r'(x)^\top r(x)$$

$$L(x)^\top d = z$$

Problem: $r'(x)^\top r'(x)$ kann sehr schlecht konditioniert sein:

$$cond_2(r'(x)^\top r'(x)) = (cond_2(r'(x)))^2$$

Alternative: QR-Zerlegung

QR-Zerlegung auch nützlich für Lösung von LGS (numerisch stabiler als LR-Zerlegung, aber doppelter Aufwand) und für numerische Rangbestimmung einer Matrix

Gauß-Newton-Verfahren (Fazit)

- Große Ersparnis an Rechenzeit gegenüber dem Newton-Verfahren:
  Aufgrund der speziellen Struktur der Ausgleichsprobleme bekommt man den ersten Teil der Hesse-Matrix 'geschenkt'.
- Verfahren funktioniert meist gut, wenn das Residuum klein ist, weil dann die Näherung der Hesse-Matrix realistisch ist.
- Bei großen Residuen ist häufig das Levenberg-Marquardt-Verfahren besser geeignet.

Überblick

Restringierte Optimierung

In diesem Abschnitt betrachten wir die schon eingeführte Problemklasse:

Problem (Standard-Optimierungsproblem (SOP))

Minimiere $f(\mathbf{x})$ *u. d. N.*

$$g_i(\mathbf{x}) \le 0, \quad i = 1, \dots, m,$$
$$h_j(\mathbf{x}) = 0, \quad j = 1, \dots, p.$$

bzw. in Vektornotation

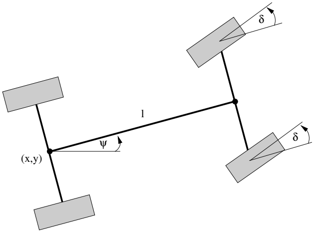
$$g(\mathbf{x}) \le \mathbf{0},$$
$$h(\mathbf{x}) = \mathbf{0}.$$

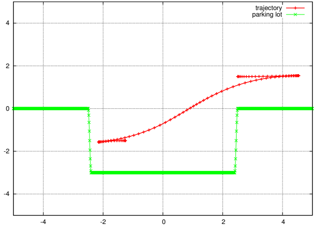
Dabei seien $\mathbf{x} \in \mathbb{R}^n$ *und die Funktionen*

$$f : \mathbb{R}^n \to \mathbb{R},$$
$$g = (g_1, \dots, g_m)^\top : \mathbb{R}^n \to \mathbb{R}^m \text{ und}$$
$$h = (h_1, \dots, h_p)^\top : \mathbb{R}^n \to \mathbb{R}^p.$$

Optimalsteuerung von gewöhnlichen DGL

Beispiel: Optimales Fahren - Einparkmanöver :





$f(\mathbf{y}, \mathbf{u}, t_f) = t_f$ (Gesamtfahrdauer)

Hier Zustand **y** und Steuerung **u** zeitabhängige Funktionen

Ungleichungs-NB.:
Steuerungs- und Zustandsbeschränkungen

Gleichungs-NB.:
DGL für **y**, mit Anfangs- und Endbedingungen

Anwendung: Produktionsplanung

- Herstellung von zwei Produkten P_1 und P_2, die einen Deckungsbeitrag von 10 bzw. 20 Euro pro Stück besitzen
- Zur Herstellung von einem Stück von P_1 oder P_2 ist jeweils eine Maschine nötig, von denen 100 vorhanden sind.
- Zur Herstellung von einem Stück P_1 werden 6 Einheiten des verwendeten Rohstoffs benötigt, bei P_2 sind es 9. Insgesamt sind 720 Einheiten vorhanden.
- In der Montageabteilung wird für P_1 niemand benötigt, für P_2 ein Arbeiter. Insgesamt arbeiten hier 60 Personen.
- Wie viele Stück von P_1 und P_2 muss man verkaufen, um den Deckungsbeitrag zu maximieren?

Formulierung als Optimierungsproblem

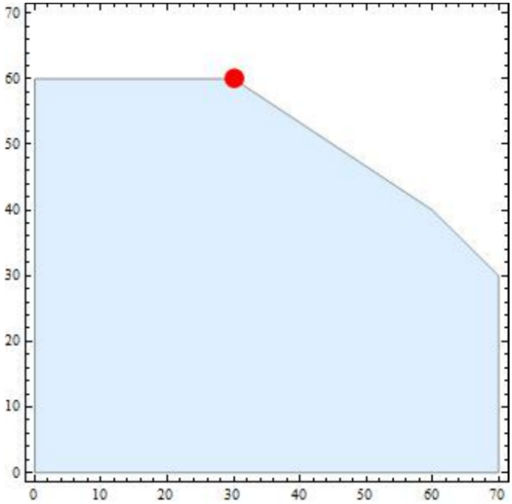
Mathematische Formulierung als lineares Optimierungsproblem:

$$\text{Maximiere} \quad f(x_1, x_2) = 10x_1 + 20x_2$$

unter den Nebenbedingungen

$$\begin{aligned} x_1 + x_2 &\leq 100 \quad \text{(Maschinen)} \\ 6x_1 + 9x_2 &\leq 720 \quad \text{(Rohstoffe)} \\ x_2 &\leq 60 \quad \text{(Montage)} \\ x_1, x_2 &\geq 0 \end{aligned}$$

Lösung



Erweiterung des Modells

Annahme: Der Verkaufspreis und damit der Deckungsbeitrag von zwei Gütern ist jeweils von der Absatzmenge abhängig, z.B.

- Absatzzahlen x_1 und x_2
- Herstellungskosten 10
- Verkaufspreis (Preis-Absatz-Funktion) $30 - x_1$ bzw. $30 - x_2$

$\Rightarrow$ Deckungsbeitrag beschrieben durch folgende Funktion:

$$\begin{aligned} f(x_1, x_2) &= (30 - x_1)x_1 - 10x_1 + (30 - x_2)x_2 - 10x_2 \\ &= 20x_1 - x_1^2 + 20x_2 - x_2^2 \end{aligned}$$

unter linearen Nebenbedingungen für Maschinen, Rohstoffe, etc.

$\Rightarrow$ Quadratisches Optimierungsproblem mit linearen Nebenbedingungen

Klassifizierung nichtlinearer Optimierungsprobleme

Lineare Zielfunktion:

- Lineare Ungleichungs-Nebenbedingungen
- Nichtlineare Nebenbedingungen

Quadratische Zielfunktion:

- Lineare Gleichungs-Nebenbedingungen
- Lineare Ungleichungs-Nebenbedingungen
- Nichtlineare Gleichungs-Nebenbedingungen
- Nichtlineare Ungleichungs-Nebenbedingungen

Beliebige nichtlineare Zielfunktion:

- Lineare Gleichungs-Nebenbedingungen
- Lineare Ungleichungs-Nebenbedingungen
- Nichtlineare Gleichungs-Nebenbedingungen
- Nichtlineare Ungleichungs-Nebenbedingungen

Grundprinzipien restringierte Optimierung

- Betrachte 2mal stetig differenzierbare $f : \mathbb{R}^n \to \mathbb{R}$.
- Maxima oder Minima der Zielfunktion $f(\mathbf{x})$ können im Inneren oder am Rand des zulässigen Bereichs auftreten.
- Bei **Extrema im Inneren** gelten die Optimalitätsbedingungen der unrestringierten Optimierung, also $\nabla f(\mathbf{x}) = 0$.
- Bei **Extrema am Rand** müssen Nebenbedingungen aktiv sein, d.h. mit Gleichheit gelten.
- Der **zulässige Bereich** X ist durch die Nebenbedingungen gegeben.
- Bei der Suche nach einem Minimum bewegt man sich von einem zulässigen Punkt in einer Suchrichtung weiter.
- Für jeden zulässigen Punkt $\mathbf{x} \in X$ ist $\mathbf{d}$ eine **zulässige Richtung**, wenn es ein $r > 0$ gibt, so dass für alle $\alpha \in [0, r]$ gilt: $\mathbf{x} + \alpha \cdot \mathbf{d} \in X$

Minimierung bei Gleichungs-NB.

Aufgabe: Bestimme das Minimum der Funktion

$$f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$$

unter den Nebenbedingungen

$$h(\mathbf{x}) = \mathbf{0} \quad \text{mit} \quad h(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}^p$$

Dabei ist n die Dimension des Problems und p die Anzahl der Gleichungs-Nebenbedingungen.

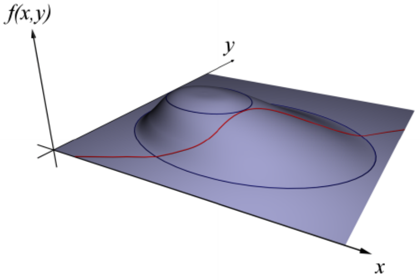
Bedingung für ein Minimum $\hat{\mathbf{x}}$: Der negative Gradient von $f(\hat{\mathbf{x}})$ liegt im Raum, der senkrecht auf den Nebenbedingungen steht,
d.h. bei nur einer Nebenbedingung $h(\hat{\mathbf{x}}) = 0$ gilt

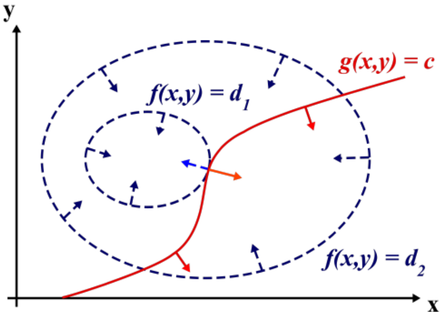
$$-\nabla f(\hat{\mathbf{x}}) = c \cdot \nabla h(\hat{\mathbf{x}}),$$

bei mehreren (mit $\boldsymbol{\mu} \in \mathbb{R}^p$):

$$-\nabla f(\hat{\mathbf{x}}) = J_h(\hat{\mathbf{x}})^\top \boldsymbol{\mu}$$

Funktion f und Gleichungs-NB. g





Notwendige Bedingungen 1. Ordnung

Man definiert für $\mathbf{x} = (x_1, \ldots, x_n)^\top$ und $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_p)^\top$ die **Lagrange-Funktion** $L : \mathbb{R}^{n+p} \to \mathbb{R}$ mit

$$L(\mathbf{x}, \boldsymbol{\mu}) = f(\mathbf{x}) + \boldsymbol{\mu}^\top h(\mathbf{x}) \, .$$

Mit Hilfe des Gradienten von L erhält man eine notwendige Bedingung für ein Minimum:

$$\nabla L(\mathbf{x}, \boldsymbol{\mu}) = \begin{pmatrix} \nabla_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\mu}) \\ \nabla_{\mu} L(\mathbf{x}, \boldsymbol{\mu}) \end{pmatrix} = \begin{pmatrix} \nabla f(\mathbf{x}) + J_h(\mathbf{x})^\top \boldsymbol{\mu} \\ h(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \mathbf{0}_n \\ \mathbf{0}_p \end{pmatrix}$$

mit der Jacobi-Matrix

$$J_h(\mathbf{x}) = \begin{pmatrix} \frac{\partial h_1(x_1,\ldots,x_n)}{\partial x_1} & \cdots & \frac{\partial h_1(x_1,\ldots,x_n)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_p(x_1,\ldots,x_n)}{\partial x_1} & \cdots & \frac{\partial h_p(x_1,\ldots,x_n)}{\partial x_n} \end{pmatrix}$$

Beispiel für $n = 2$ und $p = 1$

Aufgabe: Minimieren der Oberfläche eines Kreiszylinders bei gegebenem Volumen von einem Liter (d.h. 1000 cm^3)

Nichtlineare Minimierungsaufgabe: Minimiere

$$f(x_1, x_2) = 2\pi x_1(x_1 + x_2)$$

unter der Nebenbedingung

$$h(x_1, x_2) = \pi x_1^2 x_2 - V = 0\,.$$

Berechnung des Gradienten:

$$\nabla L(\mathbf{x}, \mu) = \begin{pmatrix} \nabla f(\mathbf{x}) + J_h(\mathbf{x})^\top \mu \\ h(\mathbf{x}) \end{pmatrix} = \pi \cdot \begin{pmatrix} 2(2x_1 + x_2 + x_1 x_2 \mu) \\ 2x_1 + x_1^2 \mu \\ x_1^2 x_2 - V/\pi \end{pmatrix}$$

$\Rightarrow$ Lösung: $x_1 = 5.42\,,\ x_2 = 10.84\,,\ \mu = -0.369$

Notwendige Bedingungen 2. Ordnung

Die Hesse-Matrix von $L(\mathbf{x}, \mu)$ ist

$$H_L(\mathbf{x}, \mu) = \begin{pmatrix} B(\mathbf{x}, \mu) & J_h^\top(\mathbf{x}) \\ J_h(\mathbf{x}) & 0 \end{pmatrix}$$

mit

$$B(\mathbf{x}, \mu) = \nabla^2_{\mathbf{xx}} L(\mathbf{x}, \mu) = H_f(\mathbf{x}) + \sum_{i=1}^{p} \mu_i H_{h_i}(\mathbf{x})\,.$$

An einem minimalen Punkt $\hat{\mathbf{x}}$ ist $H_L(\mathbf{x}, \mu)$ positiv definit für alle $\mathbf{d}$ aus dem Nullraum von $J_h^\top(\hat{\mathbf{x}})$, d.h. für alle Richtungen tangential zu den Nebenbedingungen, d.h.

$$\mathbf{d}^\top B(\mathbf{x}, \mu)\mathbf{d} \geq 0 \quad \text{für } \mathbf{d} \text{ senkrecht auf allen Zeilen von } J_h^\top(\hat{\mathbf{x}})\,.$$

Das ist erfüllt, wenn die Matrix $Z^\top B(\mathbf{x}, \mu) Z$ positiv definit für $(\hat{\mathbf{x}}, \hat{\mu})$ ist. Dabei sind die Spalten von Z eine Basis des Nullraums.

Beispiel für $n = 2$ und $m = 1$

Berechnung der Hesse-Matrix:

$$H_f(\mathbf{x}) = 2\pi \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{und} \quad H_h(\mathbf{x}) = 2\pi \begin{pmatrix} x_2 & x_1 \\ x_1 & 0 \end{pmatrix}$$

und damit

$$B(\hat{\mathbf{x}}, \hat{\mu}) = 2\pi \begin{pmatrix} 2 + x_2\mu & 1 + x_1\mu \\ 1 + x_1\mu & 0 \end{pmatrix} = \begin{pmatrix} -12.6 & -6.3 \\ -6.3 & 0 \end{pmatrix}$$

Berechnung der projizierten Hesse-Matrix:

Zu $J_h(\hat{\mathbf{x}}) = (369\,,\, 92.3)$ erhält man z.B. $Z = \begin{pmatrix} -0.243 \\ 0.970 \end{pmatrix}$ und damit $Z^\top BZ = 2.23$.

$\Rightarrow$ Minimum an der Stelle $\hat{\mathbf{x}}$ (mit $f(\hat{\mathbf{x}}) = 554\, cm^2$)

Lagrange-Newton-Verfahren für gleichungsrestringierte Probleme

(SOP) ohne Ungleichungsrestriktionen ergibt

Problem (Nichtlineares Optimierungsproblem, rein gleichungsrestringiert)

Minimiere $f(\mathbf{x})$ *u. d. N.*

$$h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, p.$$

Dabei seien $x \in \mathbb{R}^n$ *und die Funktionen*

$$f : \mathbb{R}^n \to \mathbb{R},$$
$$h = (h_1, \ldots, h_p)^\top : \mathbb{R}^n \to \mathbb{R}^p.$$

Hier kann man ggf. mithilfe des Satzes über implizite Funktionen auflösen, partitioniere dazu $\mathbf{x} = (\mathbf{y}, \mathbf{z})^\top \in \mathbb{R}^{n-p} \times \mathbb{R}^p$ und setze $h(\mathbf{y}, \mathbf{z}(\mathbf{y})) = \mathbf{0}$, somit

$$\boldsymbol{\mu}^\top := -\frac{\partial f}{\partial \mathbf{z}}(\hat{\mathbf{y}}, \hat{\mathbf{z}}) \left(\frac{\partial h}{\partial \mathbf{z}}(\hat{\mathbf{y}}, \hat{\mathbf{z}}) \right)^{-1}$$

und dann wie beim unrestringierten Optimierungsproblem vorgehen.

Lagrangesche Multiplikatorenregel

Theorem (Lagrangesche Multiplikatorenregel)

Seien $f : \mathbb{R}^n \to \mathbb{R}$ und $h : \mathbb{R}^n \to \mathbb{R}^p$, stetig differenzierbar und $\hat{\mathbf{x}}$ ein lokales Minimum von (SOP) für $m = 0$.

Des Weiteren gelte

$$\mathit{Rang}(h'(\hat{\mathbf{x}})) = p,$$

wobei $h'(\hat{\mathbf{x}})_{ij} = J_h(\hat{\mathbf{x}})_{ij} := \frac{\partial h_i}{\partial x_j}$ die Jacobi-Matrix von h in $\hat{\mathbf{x}}$.

Dann existiert ein $\boldsymbol{\mu} \in \mathbb{R}^p$, so dass gilt

$$\begin{aligned} \nabla_x L(\hat{\mathbf{x}}, \boldsymbol{\mu}) &= \mathbf{0}, && \textit{(Stationarität)} \\ h(\hat{\mathbf{x}}) &= \mathbf{0}. && \textit{(Zulässigkeit)} \end{aligned}$$

<u>Wh.:</u>

Der Rang wird durch Gauß-Elimination bestimmt.
Für reelle Zahlen gilt Zeilenrang = Spaltenrang.
In 1D gilt $h' = \nabla h^\top$.

Lagrange-Newton-Verfahren

Anwendung des Newton-Verfahrens auf $F(\mathbf{x},\boldsymbol{\mu})^\top := (\nabla_x L(\mathbf{x},\boldsymbol{\mu}), h(\mathbf{x}))^\top = \mathbf{0}$ ergibt

Algorithmus (*Lagrange-Newton-Verfahren*)

(i) *Wähle* $\mathbf{x}^{[0]} \in \mathbb{R}^n$ *und* $\boldsymbol{\mu}^{[0]} \in \mathbb{R}^p$.
Setze $k := 0$.

(ii) *Falls*

$$\|F(\mathbf{x}^{[k]},\boldsymbol{\mu}^{[k]})\| \approx 0,$$

dann STOP.

(iii) *Löse das LGS*

$$\begin{pmatrix} B(\mathbf{x}^{[k]},\boldsymbol{\mu}^{[k]}) & h'(\mathbf{x}^{[k]})^\top \\ h'(\mathbf{x}^{[k]}) & 0 \end{pmatrix} \begin{pmatrix} \mathbf{d} \\ \mathbf{v} \end{pmatrix} = - \begin{pmatrix} \nabla_x L(\mathbf{x}^{[k]},\boldsymbol{\mu}^{[k]}) \\ h(\mathbf{x}^{[k]}) \end{pmatrix}.$$

Setze $\mathbf{x}^{[k+1]} := \mathbf{x}^{[k]} + \mathbf{d}$ *und* $\boldsymbol{\mu}^{[k+1]} := \boldsymbol{\mu}^{[k]} + \mathbf{v}$.

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

Konvergenz des Lagrange-Newtonverfahrens

Aus der Konvergenz des lokalen Newtonverfahrens folgt die lokale Konvergenz des Lagrange-Newtonverfahrens superlinear bzw. quadratisch

Hinreichende Kriterien für die Invertierbarkeit von

$$\begin{pmatrix} B(\mathbf{x}^{[k]}, \mu^{[k]}) & h'(\mathbf{x}^{[k]})^\top \\ h'(\mathbf{x}^{[k]}) & \mathbf{0} \end{pmatrix}$$

sind:

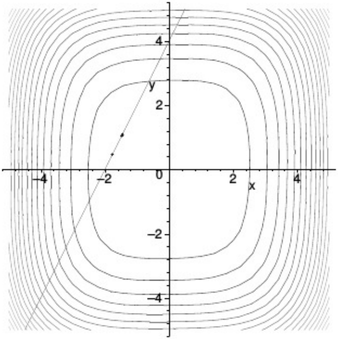
- Es gilt (LICQ) und
- die hinreichende Optimalitätsbedingung 2. Ordnung, d.h. $T(\hat{\mathbf{x}}) = \{\mathbf{d} \mid h'(\hat{\mathbf{x}})\mathbf{d} = 0\}$.

Beispiel für Lagrange-Newton-Verfahren

Minimiere $2x_1^4 + x_2^4 + 4x_1^2 - x_1 x_2 + 6x_2^2$
u.d.N. $2x_1 - x_2 = -4, \quad x_1, x_2 \in \mathbb{R}$.

Beispiel für Lagrange-Newton-Verfahren

Minimiere $2x_1^4 + x_2^4 + 4x_1^2 - x_1 x_2 + 6x_2^2$
u.d.N. $2x_1 - x_2 = -4, \quad x_1, x_2 \in \mathbb{R}$.




```
ITERATION  0
  ZIELFUNKTION  =         0.0000000000000000E+00
  BESCHRAENKUNG =         0.4000000000000000E+01
  NORM KKT      =         0.0000000000000000E+00
  X=              0.0000000000000000E+00          0.0000000000000000E+00
  LAMBDA=         0.0000000000000000E+00

ITERATION  1
  ZIELFUNKTION  =         0.3425678372606001E+02
  BESCHRAENKUNG =         0.0000000000000000E+00
  NORM KKT      =         0.4430579634007108E+02
  X=             -0.1769230769230769E+01          0.4615384615384616E+00
  LAMBDA=         0.7307692307692307E+01

ITERATION  2
  ZIELFUNKTION  =         0.2756373273045131E+02
  BESCHRAENKUNG =         0.0000000000000000E+00
  NORM KKT      =         0.5155005942054521E+01
  X=             -0.1452391998833495E+01          0.1095216002333011E+01
  LAMBDA=         0.1660806234685793E+02

ITERATION  3
  ZIELFUNKTION  =         0.2754452288430214E+02
  BESCHRAENKUNG =         0.0000000000000000E+00
  NORM KKT      =         0.1497798610217076E-01
  X=             -0.1467843643905216E+01          0.1064312712189567E+01
  LAMBDA=         0.1904961295898514E+02

ITERATION  4
  ZIELFUNKTION  =         0.2754452202163215E+02
  BESCHRAENKUNG =         0.0000000000000000E+00
  NORM KKT      =         0.6772437517980240E-06
  X=             -0.1467948113419141E+01          0.1064103773161718E+01
  LAMBDA=         0.1905680332151563E+02

ITERATION  5
  ZIELFUNKTION  =         0.2754452202163215E+02
  BESCHRAENKUNG =         0.0000000000000000E+00
  NORM KKT      =         0.3552713678800501E-14
  X=             -0.1467948118040920E+01          0.1064103763918160E+01
  LAMBDA=         0.1905680364713604E+02
```

Nachteile des Lagrange-Newton-Verfahrens

- Das Verfahren ist schlecht zu verallgemeinern auf nichtlineare Optimierungsprobleme mit Ungleichungs-Nebenbedingungen. Ansätze mit sog. NCP-Funktionen scheitern meist an der Differenzierbarkeit.
- Im Lagrange-Newton-Verfahren wird nur ein KKT-Punkt, nicht zwangsläufig ein Minimum berechnet.

Zur Erinnerung: Im Newton-Verfahren wird in jedem Schritt ein quadratisches Hilfsproblem (die Taylor-Approximation 2. Ordnung) exakt gelöst.

Idee: Übertrage dies auf den restringierten Fall
$\Rightarrow$ Sequential Quadratic Programming (SQP-Verfahren)

Verfahren der restringierten Optimierung I

- Eliminierung der Ungleichungsrestriktionen:
  - **Penalty- und Multiplikator-Verfahren**:
    Ankopplung der NB. als geeignet gewichteter **Penalty-Term** an die Zielfunktion
    dann wiederum Verfahren der unrestringierten Optimierung

    Bsp. Penalty-Verfahren, Multiplikator-Penalty-Verfahren
  - **Innere Punkte-Verfahren**:
    Mithilfe von **Barrierefunktionen** wird die Annäherung an den Rand ("die Barriere") des zulässigen Bereichs bestraft.
    Daher können Ungleichungsnebenbedingungen eliminiert werden.

Verfahren der restringierten Optimierung II

- **Sequentielle quadratische Programmierung (SQP)**:

  (SOP) wird lokal approximiert durch (QOP), das Suchrichtung liefert

  Erweiterung Lagrange-Newton-Verfahren auf Probleme mit Ungleichungs-NB.

- **Verfahren für Komplementaritätsprobleme**:

  Z.B. semiglatte Newtonverfahren, Variationsmethoden

  Man versucht die notwendigen Bedingungen umzuschreiben und direkt zu lösen

Globalisierungsstrategien in der restringierten Optimierung

- **Liniensuche**

  Z.B. Armijo-Verfahren

- **Trust-Region-Verfahren**

  Approximation auf einem Vertrauensbereich

- **Filterverfahren**

  Versuche nicht-dominierte Iterierte zu erzeugen, wobei Zielfunktionswert und Verletzung der Restriktionen betrachtet werden

Globalisierungsstrategien in der restringierten Optimierung

- **Liniensuche**

  Z.B. Armijo-Verfahren

- **Trust-Region-Verfahren**

  Approximation auf einem Vertrauensbereich

- **Filterverfahren**

  Versuche nicht-dominierte Iterierte zu erzeugen, wobei Zielfunktionswert und Verletzung der Restriktionen betrachtet werden

Wie in der unrestringierten Optimierung suche notwendige & hinreichende Optimaltätsbedingungen

KKT-Bedingungen

Definition (Lagrange-Funktion)

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := f(\mathbf{x}) + \sum_{i=1}^{m} \lambda_i \, g_i(\mathbf{x}) + \sum_{j=1}^{p} \mu_j \, h_j(\mathbf{x})$$

heißt **Lagrange-Funktion** für das Problem (SOP).

$\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_m)^\top \in \mathbb{R}^m$ und $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_p)^\top \in \mathbb{R}^p$ heißen **(Lagrangesche) Multiplikatoren.**

Definition ((In)aktive Ungleichungsrestriktionen)

Sei $\mathbf{x}$ zulässig für (SOP).

Die Beschränkung $g_i(\mathbf{x}) \leq \mathbf{0}$ heißt
aktiv in x, wenn $g_i(\mathbf{x}) = \mathbf{0}$,
inaktiv in x, wenn $g_i(\mathbf{x}) < \mathbf{0}$.

$\mathcal{A}(\mathbf{x}) := \{i \,|\, g_i(\mathbf{x}) = \mathbf{0},\ 1 \leq i \leq m\}$
heißt **Indexmenge der in x aktiven Ungleichungsrestriktionen.**

Notwendige Bedingungen 1. Ordnung

Theorem (Karush-Kuhn-Tucker-Bedingungen)

Seien f, g_i, $i = 1, \ldots, m$, und h_j, $j = 1, \ldots, p$, stetig differenzierbar und $\hat{x}$ ein lokales Minimum von (SOP).

Des Weiteren gilt die Linear Independence Constraint Qualification (LICQ) in $\hat{\mathbf{x}}$, d.h. $\nabla g_i(\hat{\mathbf{x}})$, $i \in \mathcal{A}(\hat{\mathbf{x}})$, und $\nabla h_j(\hat{\mathbf{x}})$, $j = 1, \ldots, p$, sind linear unabhängig.

Dann existieren eindeutige $\boldsymbol{\lambda} \in \mathbb{R}^m$ und $\boldsymbol{\mu} \in \mathbb{R}^p$, so dass gilt

$$\nabla_x L(\hat{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \qquad \textit{(Stationarität)} \tag{3.1}$$

$$\nabla f(\hat{\mathbf{x}}) + \sum_{i=1}^{m} \lambda_i \nabla g_i(\hat{\mathbf{x}}) + \sum_{j=1}^{p} \mu_j \nabla h_j(\hat{\mathbf{x}}) = \mathbf{0}, \tag{3.2}$$

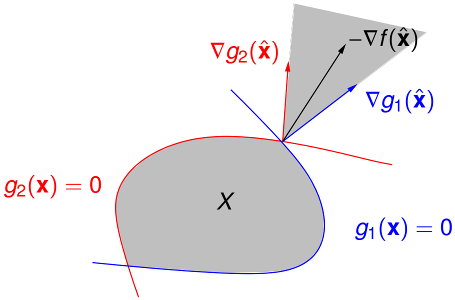
$$\boldsymbol{\lambda} \geq \mathbf{0}, \qquad \boldsymbol{\lambda}^\top g(\hat{\mathbf{x}}) = \mathbf{0}, \qquad \textit{(Komplementarität)} \tag{3.3}$$

$$g(\hat{\mathbf{x}}) \leq \mathbf{0}, \qquad h(\hat{\mathbf{x}}) = \mathbf{0}. \qquad \textit{(Zulässigkeit)} \tag{3.4}$$

KKT-Punkte, d.h. Punkte $(\hat{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu})$, die (3.1) – (3.4) erfüllen, sind Kandidaten für Optimallösungen eines (SOP).

KKT-Bedingungen: Geometrische Motivation

Bsp.: 2 Ungleichungsrestriktionen und keine Gleichungsrestriktionen



Falls in einem lokalen Minimum $\hat{x}$ die Gradienten der aktiven Ungleichungsrestriktionen linear unabhängig sind, kann der negative Gradient der Zielfunktion als nicht-negative Linearkombination der Gradienten der aktiven Beschränkungen dargestellt werden.

Regularitätsbedingungen

Falls **Regularitätsbedingungen** (constraint qualifications (CQ)) wie (LICQ) erfüllt, sind die KKT-Bedingungen notwendig.

Weitere Regularitätsbedingungen: Mangasarian-Fromowitz, Robinson, Abadie, Slater ("Existenz eines inneren Punkts"), Constant-Rank-CQ ...

Falls keine (CQ) erfüllt ist, dann gelten die KKT-Bedingungen i.A. so nicht:
Vor f tritt ein weiterer Multiplikator auf (Fritz-John-Bedingungen), der evtl. Null wird.

Die (CQ) schließen isolierte Punkte bzw. Punkte, in die man nicht in hineinlaufen kann, aus.

Typischerweise hängen (CQ) von $\hat{\mathbf{x}}$ ab $\rightsquigarrow$ schwierig *a priori* zu überprüfen

(LICQ) garantiert darüber hinaus die Eindeutigkeit der Multiplikatoren.

Kritischer Kegel

Sei $(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ ein KKT-Punkt von (SOP) und

$$\begin{aligned} I_0(\mathbf{x}) &:= \{i \in \{1, \ldots, m\} \mid g_i(\mathbf{x}) = 0, \lambda_i = 0\}, \\ I_>(\mathbf{x}) &:= \{i \in \{1, \ldots, m\} \mid g_i(\mathbf{x}) = 0, \lambda_i > 0\}, \end{aligned}$$

d.h. strikte Komplementarität verletzt bzw. erfüllt.

Dann definiere den sogenannten **kritischen Kegel**

$$T(\mathbf{x}) := \{\mathbf{d} \in \mathbb{R}^n \mid \begin{array}{ll} \nabla h_j(\mathbf{x})^\top d_j = 0, & j = 1, \ldots, p, \\ \nabla g_i(\mathbf{x})^\top d_i = 0, & i \in I_>(\mathbf{x}), \\ \nabla g_i(\mathbf{x})^\top d_i \leq 0, & i \in I_0(\mathbf{x}) \end{array} \}.$$

Interpretation: T ist Menge der mehrdeutigen (kritischen) Richtungen

Notwendige Bedingungen 2. Ordnung

Theorem (Notwendige Optimalitätsbedingung 2. Ordnung)

Sei $L : \mathbb{R}^{n \times p \times m} \to \mathbb{R}$ 2mal stetig differenzierbar bzgl. $\mathbf{x}$. Sei $\hat{\mathbf{x}}$ ein lokales Minimum von (SOP), so dass (LICQ) erfüllt ist.

Dann gilt

$$\mathbf{d}^\top \nabla^2_{xx} L(\hat{\mathbf{x}}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \mathbf{d} \geq 0 \quad \textit{für alle } \mathbf{d} \in T(\hat{\mathbf{x}}),$$

wobei $\boldsymbol{\lambda} \in \mathbb{R}^m$, $\boldsymbol{\mu} \in \mathbb{R}^p$ die eindeutig bestimmten Lagrangeschen Multiplikatoren aus den KKT-Bedingungen sind.

Hinreichende Bedingungen 2. Ordnung

Theorem (Hinreichende Optimalitätsbedingung 2. Ordnung)

Sei $L : \mathbb{R}^{n \times p \times m} \to \mathbb{R}$ *2mal stetig differenzierbar bzgl.* $\mathbf{x}$.
Sei $(\hat{\mathbf{x}}, \lambda, \mu)$ *ein KKT-Punkt von (SOP) mit*

$$\mathbf{d}^\top \nabla^2_{xx} L(\hat{\mathbf{x}}, \lambda, \mu)\mathbf{d} > 0 \quad \textit{für alle } \mathbf{d} \in T(\hat{\mathbf{x}}) \setminus \{\mathbf{0}\}.$$

Dann ist $\hat{x}$ *ein striktes lokales Minimum von (SOP).*

SQP-Verfahren für gleichungsrestringierte Probleme

Setzt man $B_k := \nabla^2_{xx} L(\mathbf{x}^{[k]}, \boldsymbol{\mu}^{[k]})$ im Gleichungssystem des Lagrange-Newton-Verfahrens,
so erhält man

$$\begin{aligned} B_k\, \mathbf{d}^{[k]} + h'(\mathbf{x}^{[k]})^\top (\mathbf{v}^{[k]} + \boldsymbol{\mu}^{[k]}) &= -\nabla f(\mathbf{x}^{[k]}), \\ h'(\mathbf{x}^{[k]})\mathbf{d} &= -h(\mathbf{x}^{[k]}). \end{aligned}$$

Dies sind genau die KKT-Bedingungen des quadratischen Problems

$$\begin{array}{lll} \min\limits_{\mathbf{d}} & \nabla f(\mathbf{x}^{[k]})\mathbf{d} + \frac{1}{2}\mathbf{d}^\top B_k \mathbf{d} & \\ \text{u.d.N.} & h_j(\mathbf{x}^{[k]}) + h_j'(\mathbf{x}^{[k]})\mathbf{d}_j = 0 & \text{für alle } j = 1, \dots, p. \end{array}$$

SQP-Verfahren allgemein

Einschließlich Ungleichungen und mit $B_k := \nabla^2_{xx} L(\mathbf{x}^{[k]}, \boldsymbol{\lambda}^{[k]}, \boldsymbol{\mu}^{[k]})$ erhält man

$$
\begin{array}{lll}
\min\limits_{\mathbf{d}} & \nabla f(\mathbf{x}^{[k]})\mathbf{d} + \frac{1}{2}\mathbf{d}^\top B_k \mathbf{d} & \\
\text{u.d.N.} & g_i(\mathbf{x}^{[k]}) + g_i'(\mathbf{x}^{[k]})\mathbf{d} \leq 0 & \text{für alle } i = 1, \ldots, m, \\
& h_j(\mathbf{x}^{[k]}) + h_j'(\mathbf{x}^{[k]})\mathbf{d} = 0 & \text{für alle } j = 1, \ldots, p.
\end{array}
$$

Man löst also in jedem Iterationsschritt ein quadratisches Problem.

Dies ist das wichtige **SQP-Verfahren (sequential quadratic programming).**

SQP-Algorithmus

1. Wähle $(\mathbf{x}^{[0]}, \lambda^{[0]}, \boldsymbol{\mu}^{[0]})^\top \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^p$ und setze $k = 0$
2. Ist $(\mathbf{x}^{[k]}, \lambda^{[k]}, \boldsymbol{\mu}^{[k]})^\top$ ein KKT-Punkt des Problems, dann STOP.
3. Berechne eine Lösung (oder zumindest einen KKT-Punkt) $\mathbf{d}^{[k]}$ des quadratischen Problems

$$\min_{\mathbf{d}} \nabla f(\mathbf{x}^{[k]})\mathbf{d} + \frac{1}{2}\mathbf{d}^\top B_k \mathbf{d}$$

unter den Nebenbedingungen

$$h(\mathbf{x}^{[k]}) + J_h(\mathbf{x}^{[k]})\mathbf{d} = \mathbf{0}_m$$
$$g(\mathbf{x}^{[k]}) + J_g(\mathbf{x}^{[k]})\mathbf{d} \leq \mathbf{0}_p$$

Dabei wird häufig die Active-Set-Methode (später) verwendet.
4. Setze

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

berechne B_{k+1}.
Setze $k := k + 1$ und gehe zu Schritt 2.

Konvergenz des SQP-Verfahrens

Satz: Es seien folgende Voraussetzungen erfüllt:

- $(\hat{\mathbf{x}}, \lambda, \mu)$ ist ein KKT-Punkt des Optimierungsproblems.
- Die Zeilen von $J_g(\hat{\mathbf{x}})$ und $J_h(\hat{\mathbf{x}})$ (d.h. die Gradienten $\nabla g_1(\hat{\mathbf{x}}), \ldots, \nabla g_m(\hat{\mathbf{x}}), \nabla h_1(\hat{\mathbf{x}}), \ldots, \nabla h_p(\hat{\mathbf{x}})$) sind linear unabhängig.
- Es gilt $\mathbf{d}^\top B(\hat{\mathbf{x}}, \lambda, \mu)\mathbf{d} > 0$ für alle $\mathbf{d} \neq \mathbf{0}_n$ mit $\nabla g_j(\hat{\mathbf{x}})\mathbf{d} = 0$ und $\nabla h_j(\hat{\mathbf{x}})\mathbf{d} = 0$.

  (Zur Erinnerung: $B(\hat{\mathbf{x}}, \lambda, \mu) = \nabla^2_{\mathbf{xx}} L(\hat{\mathbf{x}}, \lambda, \mu) = H_f(\hat{\mathbf{x}}) + \sum_{i=1}^{m} \lambda_i H_{g_i}(\hat{\mathbf{x}}) + \sum_{i=1}^{p} \mu_i H_{h_i}(\hat{\mathbf{x}})$)
- Es gilt strikte Komplementarität, d.h. $g_i(\hat{\mathbf{x}}) + \lambda_i \neq 0$ für $i = 1, \ldots, m$.

Dann ist das SQP-Verfahren wohldefiniert und die Folge $\{(\mathbf{x}^{[k]}, \lambda^{[k]}, \mu^{[k]})\}$ konvergiert superlinear gegen den KKT-Punkt $(\hat{\mathbf{x}}, \lambda, \mu)$.

Globalisiertes SQP-Verfahren

- Lokal hat man zwar sehr schnelle Konvergenz, allerdings ist dieser Bereich häufig sehr klein.
- Eine Globalisierung, z.B. durch Armijo-ähnliche Schrittweitenstrategien und Updates der Form

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

  ist im Allgemeinen notwendig. Allerdings ist es nicht möglich, t über einen Abstieg der Funktionswerte zu bestimmen.
- Stattdessen bestimmt man $t > 0$, so dass ein Abstieg einer Bewertungsfunktion (*merit function*) stattfindet. Als Bewertungsfunktion kann man z.B. eine Straffunktion betrachten:

$$P(\mathbf{x};\eta) := f(\mathbf{x}) + \eta \sum_{j=1}^{p} |h_j(\mathbf{x})| + \eta \sum_{i=1}^{m} \max\{0, g_i(\mathbf{x})\}$$

Matlab-Implementierung

- Im Befehl *fmincon* sind zwei Varianten der SQP-Methode implementiert.
- Wie im unrestringierten Fall kann die Berechnung der Hesse-Matrix durch Näherungen ersetzt werden. In Matlab wird eine BFGS-Formel verwendet.
- Die quadratischen Teilprobleme werden mit der Active-Set-Methode von Gill und Murray implementiert.
- Alternativ können auch für allgemeine nichtlineare Optimierungsprobleme Innere-Punkte-Verfahren verwendet werden.

Penalty- und Barriere-Methoden

Typen von Verfahren:

- Penalty-Verfahren:
  Zielfunktionen der Hilfsprobleme bestrafen das Verlassen des zulässigen Bereichs (mit verschiedenen Typen von Straffunktionen)
- Multiplikator-(Penalty-)Verfahren:
  Erweiterung der Penalty-Verfahren,
  durch erweiterte Lagrange-Funktion numerisch stabiler
- Barriere-Verfahren:
  Verhindern, dass man sich zu schnell dem Rand des zulässigen Bereichs annähert
  (auch Innere Penalty-Verfahren genannt)

Penalty-Verfahren: Übersicht

Idee: Approximiere iterativ durch Lösung von unrestringierten Hilfsproblemen

Hilfsprobleme minimieren **Penalty-Funktion**

$$P(\mathbf{x};\eta_k) := f(\mathbf{x}) + \frac{\eta_k}{2}\sum_{j=1}^{p}(h_j(\mathbf{x}))^2 + \frac{\eta_k}{2}\sum_{i=1}^{m}(\max\{0, g_i(\mathbf{x})\})^2$$

für geeignete $\eta_k > 0$

Nachteile:

- Gemäß Konvergenzresultat muss η_k gegen ∞ streben,
  $\rightsquigarrow$ Algorithmus schlecht konditioniert
- P nicht exakt (exakt heißt es existiert $0 < \bar{\eta} < \infty$ so dass $\hat{\mathbf{x}}$ lokales Minimum von $P(\cdot,\eta)$ für alle $\eta \geq \bar{\eta}$)
- P i.A. nicht differenzierbar

Mit quadratischer Penalty-Funktion, Optimierungsprobleme mit Gleichungs-NB.

Anstelle des Optimierungsproblems mit Gleichungsnebenbedingungen

$$\min\ f(\mathbf{x}) \quad \text{u. d. N.} \quad h(\mathbf{x}) = \mathbf{0}_p$$

löst man eine Folge von unrestringierten Problemen der Art

$$P(\mathbf{x};\eta) = f(\mathbf{x}) + \frac{\eta}{2}\,\|h(\mathbf{x})\|^2 \quad (\eta \geq 0)\,.$$

Der **Penalty-Term** $\frac{\eta}{2}\,\|h(\mathbf{x})\|^2$ bestraft Verletzungen der Nebenbedingungen.
η heißt **Penalty-Parameter** und $P(\mathbf{x};\eta)$ **Penalty-Funktion**.

Je größer der Parameter η, desto stärker wird das Verletzen von Nebenbedingungen bestraft. Für $\eta \to \infty$ erwartet man also eine Lösung des restringierten Problems.

Beispiel

Das eindimensionale Beispiel

$$\min\ x^3 \quad \text{u. d. N.} \quad x^2 - 1 = 0$$

hat die Lösung $\hat{x} = -1$.

Die Penalty-Funktion

$$P(x;\eta) = x^3 + \frac{1}{2}\eta(x^2 - 1)^2$$

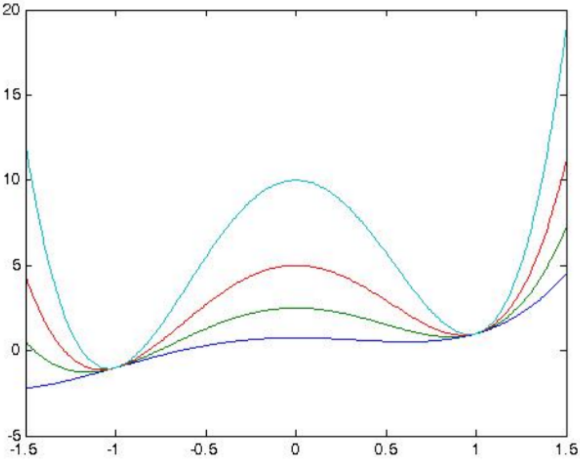
hat das globale Minimum

$$x(\eta) = -\frac{3}{4\eta} - \sqrt{1 + \frac{9}{16\eta^2}}$$

mit $\lim\limits_{\eta\to\infty} x(\eta) = -1$.

Penalty-Funktionen für wachsendes r

$P(x;\eta)$ mit $\eta = 1.5, 5, 10, 20$:



Penalty-Verfahren

Algorithmus (*Penalty-Verfahren (rein gleichungsrestringiert)*)

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

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

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

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

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

Sind f und g 2mal stetig differenzierbar, verwendet man in (ii) meist ein globalisiertes Newton-Verfahren oder ein Quasi-Newton-Verfahren.

Eigenschaften des Penalty-Verfahrens

Für stetige Funktionen f und h, einen nichtleeren zulässigen Bereich $X = \{\mathbf{x} \in \mathbb{R}^n \,|\, h(\mathbf{x}) = 0\}$ und eine streng monoton wachsende Folge $\{\eta^{[k]}\}$ erzeugt das Penalty-Verfahren eine Folge von Näherungen $(\mathbf{x}^{[k]})_k$. Dann gilt:

- $P(\mathbf{x}^{[k+1]}; \eta^{[k+1]}) \geq P(\mathbf{x}^{[k]}; \eta^{[k]})$
- $f(\mathbf{x}^{[k+1]}) \geq f(\mathbf{x}^{[k]})$
- $\|h(\mathbf{x}^{[k+1]})\| \leq \|h(\mathbf{x}^{[k]})\|$
- $\lim_{k\to\infty} \|h(\mathbf{x}^{[k]})\| = 0$
- Jeder Häufungspunkt der Folge $\{\mathbf{x}^{[k]}\}$ ist eine Lösung des Optimierungsproblems.

Konstruktion von Lagrange-Multiplikatoren

Ziel:

Konstruktion einer Folge von Lagrange-Multiplikatoren $\{\boldsymbol{\mu}^{[k]}\}$, so dass die Paare $(\mathbf{x}^{[k]}, \boldsymbol{\mu}^{[k]})$ gegen einen KKT-Punkt des Minimierungsproblems konvergieren.

In einem KKT-Punkt $(\hat{\mathbf{x}}, \boldsymbol{\mu})$ muss gelten

$$\nabla f(\hat{\mathbf{x}}) + \sum_{i=1}^{p} \mu_i \nabla h_i(\hat{\mathbf{x}}) = \mathbf{0}_n\ .$$

Außerdem gilt für den Gradienten der Straffunktion

$$\nabla P(\mathbf{x}^{[k]}; \eta^{[k]}) = \nabla f(\mathbf{x}^{[k]}) + \eta^{[k]} \sum_{i=1}^{p} h_i(\mathbf{x}^{[k]}) \nabla h_i(\mathbf{x}^{[k]}) \approx \mathbf{0}_n$$

für große k. Daher vermutet man, dass

$$\mu_i^{[k]} := \eta^{[k]} h_i(\mathbf{x}^{[k]}), \quad i = 1, \ldots, p,$$

eine gute Näherung für die Lagrange-Multiplikatoren sind.

Konvergenz der Lagrange-Multiplikatoren

Satz:

Angenommen, f und h sind stetig differenzierbar, $\{\mathbf{x}^{[k]}\}$ ist eine mit dem Penalty-Verfahren erzeugte Folge mit Grenzwert $\hat{\mathbf{x}}$ und die Gradienten $\nabla h_1(\hat{\mathbf{x}}), \ldots, \nabla h_p(\hat{\mathbf{x}})$ sind linear unabhängig.

Dann gilt:

- Die Folge $\{\boldsymbol{\mu}^{[k]}\}$ konvergiert gegen einen Vektor $\boldsymbol{\mu} \in \mathbb{R}^p$.
- Das Paar $(\hat{\mathbf{x}}, \boldsymbol{\mu})$ ist ein KKT-Punkt des Optimierungsproblems.

Optimierungsprobleme mit Ungleichungsnebenbedingungen

Beim Optimierungsproblem

$$\min\ f(\mathbf{x}) \quad \text{u. d. N.} \quad h(\mathbf{x}) = \mathbf{0}_p \ \text{ und } \ g(\mathbf{x}) \leq \mathbf{0}_m$$

schreibt man die Ungleichungs-Nebenbedingungen um:

$$(\max\{0, g_1(\mathbf{x})\}, \ldots, \max\{0, g_m(\mathbf{x})\})^\top = \mathbf{0}_m$$

Damit erhält man die Penalty-Funktion

$$P(\mathbf{x};\eta) = f(\mathbf{x}) + \frac{\eta}{2}\|h(\mathbf{x})\|^2 + \frac{\eta}{2}\sum_{i=1}^{m}(\max\{0, g_i(\mathbf{x})\})^2 \quad (\eta \geq 0)\,.$$

Man kann zeigen, dass $P(\mathbf{x};\eta)$ 1mal stetig differenzierbar ist, falls f, g und h 2mal stetig differenzierbar sind.

Kondition der Hessematrix

Mit der Jacobi-Matrix $J_h^{[k]}$ an der Stelle $\mathbf{x}^{[k]}$ erhält man die Hesse-Matrix für die Penalty-Funktion

$$H_P(\mathbf{x}^{[k]}, \eta^{[k]}) = H_f(\mathbf{x}^{[k]}) + \eta^{[k]} \left(\sum_{i=1}^{p} h_i(\mathbf{x}^{[k]}) H_{h_i}(\mathbf{x}^{[k]}) + (J_h^{[k]})^\top J_h^{[k]} \right) .$$

Mit Hilfe von $\mu_i^{[k]} = \eta^{[k]}\, h_i(\mathbf{x}^{[k]})$ ergibt dies

$$\begin{aligned} H_P(\mathbf{x}^{[k]}; \eta^{[k]}) &= H_f(\mathbf{x}^{[k]}) + \sum_{i=1}^{p} \mu_i^{[k]} H_{h_i}(\mathbf{x}^{[k]}) + \eta^{[k]} (J_h^{[k]})^\top J_h^{[k]} \\ &= H_L(\mathbf{x}^{[k]}, \lambda^{[k]}) + \eta^{[k]} (J_h^{[k]})^\top J_h^{[k]} \end{aligned}$$

Wegen $rang((J_h^{[k]})^\top J_h^{[k]}) = p$ steigen p der Eigenwerte der Hessematrix an, wenn $k \to \infty$, und die Kondition wird beliebig schlecht.

Exakte Penalty-Verfahren

Nachteile des quadratischen Penalty-Verfahrens:

- Der Penalty-Parameter η muss sehr hoch gewählt werden, damit man zu einer Lösung des ursprünglichen Optimierungsproblems gelangt und keine Nebenbedingung verletzt.
- Die Kondition der zu lösenden unrestringierten Hilfsprobleme ist dann sehr hoch und die Lösung ist schwierig.

Definition von exakten Penalty-Funktionen:

- Hier genügt ein fester, endlicher Penalty-Parameter für eine exakte Lösung des Minimierungsproblems, d.h. zu einem lokalen Minimum $\hat{\mathbf{x}}$ des restringierten Minimierungsproblems gibt es einen Parameter $\overline{\eta} > 0$, so dass $\hat{\mathbf{x}}$ für alle $\eta \geq \overline{\eta}$ auch ein lokales Minimum der Penalty-Funktion $P(\mathbf{x}; \eta)$ ist.

Eigenschaften exakter Penalty-Funktionen

Satz:

Zu einem restringierten Optimierungsproblem sei

$$P(\mathbf{x};\eta) = f(\mathbf{x}) + \eta\, p(\mathbf{x})$$

die Penalty-Funktion mit beliebigem Penalty-Term $\eta\, p(\mathbf{x})$. Angenommen, $\hat{\mathbf{x}}$ ist ein lokales Minimum des Optimierungsproblems mit $\nabla f(\hat{\mathbf{x}}) \neq \mathbf{0}$ und die Penalty-Funktion ist exakt in $\hat{\mathbf{x}}$.

Dann ist die Funktion $P(\mathbf{x})$ in $\hat{\mathbf{x}}$ nicht differenzierbar.

l_1-Penalty-Funktion

Die bekannteste exakte Penalty-Funktion erhält man aus der quadratischen (durch Weglassen der Quadrate und des Faktors $1/2$):
Die l_1**-Penalty-Funktion.** Sie ist definiert durch

$$P(\mathbf{x};\gamma) = f(\mathbf{x}) + \gamma \sum_{i=1}^{p} |h_i(\mathbf{x})| \,.$$

Häufig verwendet man statt der Konstanten γ auch einen Vektor $\boldsymbol{\gamma}$ mit Komponenten γ_i .

Zu einer beliebigen l_p-Norm definiert man die entsprechende l_p-Penalty-Funktion

$$P(\mathbf{x};\gamma) = f(\mathbf{x}) + \gamma \left(\sum_{i=1}^{m} |h_i(\mathbf{x})|^p \right)^{1/p} \,.$$

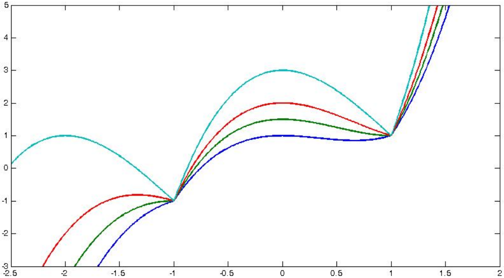
Beispiel

Wir betrachten noch einmal das Beispiel

$$\min\ x^3 \quad \text{u. d. N.} \quad x^2 - 1 = 0 \ ,$$

diesmal mit der l_1-Penalty-Funktion

$$P(x;\gamma) = x^3 + \gamma|x^2 - 1|$$



Verfahren mit l_1-Penalty-Funktion

- Das Verfahren funktioniert prinzipiell genauso wie bei der quadratischen Penalty-Funktion. Der Hauptunterschied ist, dass das unrestringierte Hilfsproblem nur mit Verfahren für nicht differenzierbare Funktionen gelöst werden kann, z.B. Nelder-Mead.

- Ist $\hat{\mathbf{x}}$ ein lokalen Minimum des gegebenen Optimierungsproblems und $\boldsymbol{\mu}$ der zugehörige Vektor der Lagrange-Multiplikatoren.
  Dann ist die Penalty-Funktion exakt, wenn $\gamma_i > |\mu_i|$ gewählt wird.

Multiplikator-Penalty-Verfahren

Betrachte hier nur Gleichungsrestriktionen

Betrachte anstatt Penalty-Funktion die **erweiterte Lagrange-Funktion** (Multiplikator-Penalty-Funktion)

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

für geeignete $\eta > 0$

Vorteile:

η muss nicht gegen ∞ streben.

L_a ist differenzierbar.

Nachteil:

Der optimale Lagrangeparameter $\boldsymbol{\mu}$ ist unbekannt.

Multiplikator-Penalty-Verfahren - Idee

Überlegung:

- Die quadratische Penalty-Funktion führt zu differenzierbaren Hilfsproblemen, aber nicht zu exakten Lösungen.
- Die exakte Penalty-Funktion führt zu nicht-differenzierbaren Hilfsproblemen, aber zur exakten Lösung für alle η, die größer als die Lagrange-Multiplikatoren sind.

Ansatz bei Multiplikator-Penalty-Verfahren:

- Kombiniere quadratischen Strafterm und Multiplikator-Term für Lagrange-Multiplikatoren, d.h. definiere

$$L_a(\mathbf{x}, \boldsymbol{\mu}; \eta) = f(\mathbf{x}) + \boldsymbol{\mu}^\top h(\mathbf{x}) + \frac{\eta}{2}\|h(\mathbf{x})\|^2 \quad (\eta \geq 0)\,.$$

Beispiel

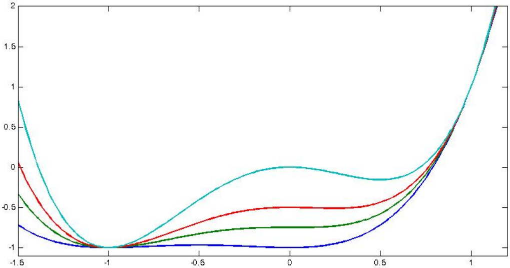
Und noch einmal das Beispiel

$$\min\ x^3 \quad \text{u. d. N.} \quad x^2 - 1 = 0\ ,$$

diesmal mit der Multiplikator-Penalty-Funktion

$$L_a(x, \mu; \eta) = x^3 + \mu(x^2 - 1) + \eta(x^2 - 1)^2$$

Für $\eta > \mu = 3/2$ ist die Multiplikator-Penalty-Funktion exakt:



Exaktheit der erweiterten Lagrange-Funktion

Hilfssatz:

Seien $Q \in \mathbb{R}^{n \times n}$ symmetrisch und $A \in \mathbb{R}^{m \times n}$ mit $m < n$ und vollem Zeilenrang m. Außerdem gilt für ein $\eta > 0$

$$\mathbf{d}^\top Q \mathbf{d} \geq \eta \|\mathbf{d}\|^2 \quad \forall \mathbf{d} \in kern(A) .$$

Dann ist für hinreichend großes η die Matrix $Q + \eta A^\top A$ positiv definit.

Satz:

Seien $(\hat{\mathbf{x}}, \lambda)$ ein KKT-Punkt des Optimierungsproblems, $f(\mathbf{x})$ 2mal stetig differenzierbar und die Zeilen von $J_h(\mathbf{x})$ linear unabhängig. Es ist außerdem die Optimalitätsbedingung 2. Ordnung erfüllt, d.h.

$$\mathbf{d}^\top L_{\mathbf{xx}}(\hat{\mathbf{x}}, \mu) \mathbf{d} \geq \overline{\eta} \|\mathbf{d}\|^2 \quad \forall \mathbf{d} \in kern(J_h(\hat{\mathbf{x}})) .$$

Dann ist $\hat{\mathbf{x}}$ für $\eta > \overline{\eta}$ ein lokaler Minimalpunkt von $L_a(\mathbf{x}, \mu; \eta)$.

Multiplikator-Penalty-Verfahren - Algorithmus

Algorithmus (*Multiplikator-Penalty-Verfahren*)

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

(ii) *Falls* $(\mathbf{x}^{[k]}, \boldsymbol{\mu}^{[k]})$ *KKT-Punkt, dann STOP.*

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

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

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

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

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

Das Multiplikator-Penalty-Verfahren lässt sich auch auf Ungleichungs-Nebenbedingungen erweitern.

Innere-Punkte-Verfahren

- Zur Lösung (nicht)linearer konvexer Optimierungsprobleme
- Im Gegensatz zum Simplex-Verfahren wird das Innere des zulässigen Bereichs durchquert.
- Man kann polynomiale Komplexität beweisen.

  Das Simplex-Verfahren (aus der linearen Optimierung) hat im worst case exponentielle Komplexität.
- Gängige IP-Verfahren können im average case mit dem Simplex-Verfahren konkurrieren.

Primal-dual interior-point method

Wir betrachten zu Demonstrationszwecken hier nur Ungleichungsnebenbedingungen:

$$\min f(\mathbf{x}) \quad u.d.N. \quad g(\mathbf{x}) \leq \mathbf{0}_m$$

zu $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^m$.

Die zugehörige logarithmische **Barrierefunktion** ist

$$B(\mathbf{x}, \eta) = f(\mathbf{x}) - \eta \sum_{i=1}^{m} \log(-g_i(\mathbf{x})),$$

wobei der **Barriereparameter** $\eta > 0$.

Idee: Minimum $B(\mathbf{x}, \eta)$ konvergiert gegen Optimallösung, wenn $\eta \to 0$.

Komplementaritätsbedingung mit Störung

Gradient der Barrierefunktion:

$$\nabla_{\mathbf{x}} B(\mathbf{x}, \eta) = \nabla f(\mathbf{x}) - \eta \sum_{i=1}^{m} \frac{\nabla g_i(\mathbf{x})}{g_i(\mathbf{x})}$$

Zur primalen Variable $\mathbf{x} \in \mathbb{R}^n$ führt man eine duale Variable $\boldsymbol{\lambda} \in \mathbb{R}^m$ durch eine **Komplementaritätsbedingung mit Störung** ein:

$$-g_i(\mathbf{x})\lambda_i = \eta \quad \text{für alle } i = 1, \ldots, m.$$

Man sucht $(\mathbf{x}, \lambda)$, so dass $\nabla_{\mathbf{x}} B(\mathbf{x}, \eta) = 0$:

$$\nabla f(\mathbf{x}) - J_g(\mathbf{x})\lambda = \mathbf{0}_n.$$

Interpretation:

- ∇f liegt im Unterraum, der durch die Gradienten der NB. aufgespannt wird.
- Die Komplementaritätsbedingung mit kleinem η heißt, dass das Minimum entweder am Rand liegt oder die Projektion von ∇f auf g_i nahe Null ist.

Innere-Punkte-Verfahren (Idee Algorithmus)

Anwendung des Newton-Verfahrens motiviert für das Update $(\mathbf{d}, \mathbf{u})^\top$ (zu $(\mathbf{x}, \lambda)$):

$$\begin{pmatrix} \nabla^2_{\mathbf{xx}} B(\mathbf{x}, \eta) & J_g^\top \\ -\lambda I_m J_g & -g I_m \end{pmatrix} \begin{pmatrix} \mathbf{d} \\ \mathbf{u} \end{pmatrix} = \begin{pmatrix} -\nabla f(\mathbf{x}) - J_g^\top \lambda \\ \eta \mathbf{e} + g I_m \lambda \end{pmatrix},$$

wobei $\mathbf{e} = (1, \ldots, 1)^\top \in \mathbb{R}^m$.

Die Bedingung $\lambda \geq \mathbf{0}_m$ wird durch die Wahl der Schrittweite α garantiert:

$$\begin{pmatrix} \mathbf{x} \\ \lambda \end{pmatrix} \to \begin{pmatrix} \mathbf{x} + \alpha \mathbf{d} \\ \lambda + \alpha \mathbf{u} \end{pmatrix}.$$

η muss auch noch über die Iterationen hin geeignet verkleinert werden.

Algorithmus für LP (primale Normalform)

Beachte duale Gleichungs-NB.: $A^\top \boldsymbol{\mu} + \boldsymbol{\lambda} = \mathbf{c}$

Algorithmus (*IP-Verfahren für Lineares Problem*)

(i) *Wähle* $\mathbf{x}^{[0]} \in \mathbb{R}^n$, $\boldsymbol{\mu}^{[0]} \in \mathbb{R}^m$, $0 < \boldsymbol{\lambda}^{[0]} \in \mathbb{R}^n$.

(ii) *Setze* $\eta = \frac{1}{n}\mathbf{x}^\top \boldsymbol{\lambda}$.

(iii) *Falls Abbruchbedingung (z.B.* $\|\nabla_{\mathbf{x}} B\|$ *und* $\eta \approx 0$*) erfüllt, dann STOP.*

(iv) *Reduziere* η*, so dass:* $0 < \eta^+ < \eta$.

(v) *Berechne Suchrichtung durch Lösung des LGS:*

$$\begin{pmatrix} 0_{n\times n} & -A^\top & -I_n \\ -A & 0_{m\times m} & 0_{m\times n} \\ \lambda I_n & 0_{n\times m} & \mathbf{x} I_n \end{pmatrix} \begin{pmatrix} \mathbf{d} \\ \mathbf{v} \\ \mathbf{u} \end{pmatrix} = \begin{pmatrix} -c + A^\top \boldsymbol{\mu} + \boldsymbol{\lambda} \\ Ax - b \\ \eta \mathbf{e} - \mathbf{x} I_n \boldsymbol{\lambda} \end{pmatrix}$$

(vi) *Wähle Schrittweite* $\alpha > 0$*, so dass* $\mathbf{x} + \alpha \mathbf{d} > \mathbf{0}_n$, $\boldsymbol{\lambda} + \alpha \mathbf{v} > \mathbf{0}_n$
(einige Varianten stellen weitere Bedingungen an α*)*

(vii) *Update:*

$$\mathbf{x}^+ = \mathbf{x} + \alpha \mathbf{d},$$
$$\boldsymbol{\mu}^+ = \boldsymbol{\mu} + \alpha \mathbf{v},$$
$$\boldsymbol{\lambda}^+ = \boldsymbol{\lambda} + \alpha \mathbf{u}.$$

(viii) *Gehe zu (ii).*

Eigenschaften von IP-Verfahren

- IP-Verfahren sind global konvergent.
- Gängige Varianten (insbesondere bzgl. der Wahl von η^+ und α) sind:
  Kurzschrittverfahren,
  Langschrittverfahren und
  Predictor-Corrector-Verfahren.
- Der worst case der Kurzschrittvariante benötigt $\mathcal{O}(\sqrt{n}/\varepsilon)$, um eine Lösung eines linearen Problems mit Genauigkeit ε zu finden.
- Praktisch beobachet man $\mathcal{O}(\log(n))$.

Lineare Optimierung: Wiederholung

Wir betrachten lineare Optimierungsprobleme, d.h. Zielfunktion und Nebenbedingung sind lineare Funktionen.

Definition (Lineare Funktion)

Eine Funktion $f : \mathbb{R}^n \to \mathbb{R}^m$ heißt **linear** genau dann, wenn für alle $x, y \in \mathbb{R}^n$ und alle $c \in \mathbb{R}$ gilt:

$$f(x + y) = f(x) + f(y),$$
$$f(cx) = cf(x).$$

Definition (Affin-lineare Funktion)

Eine Funktion $f : \mathbb{R}^n \to \mathbb{R}^m$ heißt **affin linear** genau dann, wenn es eine lineare Funktion $\tilde{f} : \mathbb{R}^n \to \mathbb{R}^m$ und einen konstanten Vektor $b \in \mathbb{R}^m$ gibt, so dass für alle $x \in \mathbb{R}^n$

$$f(x) = \tilde{f}(x) + b.$$

Jede lineare Funktion $f : \mathbb{R}^n \to \mathbb{R}^m$ hat die Darstellung $f(x) = Ax$ mit einer Matrix $A \in \mathbb{R}^{m \times n}$.
Jede affin-lineare Fn. $f : \mathbb{R}^n \to \mathbb{R}^m$ hat die Darstellung $f(x) = Ax + b$ mit einer Matrix $A \in \mathbb{R}^{m \times n}$ und einem Vektor $b \in \mathbb{R}^m$.

Definition (Lineares Programm (LP))

Minimiere $c^\top x = \sum_{j=1}^n c_j x_j$ u.d.N.

$$\sum_{j=1}^{n} a_{ij} x_j \begin{Bmatrix} \leq \\ \geq \\ = \end{Bmatrix} b_i, \qquad i = 1, \ldots, m,$$

$$x_i \begin{Bmatrix} \leq \\ \geq \end{Bmatrix} 0, \qquad i \in I,$$

wobei $I \subseteq \{1, \ldots, n\}$ eine Indexmenge und $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ und $A \in \mathbb{R}^{m \times n}$.

Variable x_i mit $i \notin I$ heißen **freie Variable.**

Primale Normalform

Das LP ist in allgemeiner Form gegeben. Für eine systematische Behandlung betrachten wir LPs in primaler Normalform:

Problem (Lineares Optimierungsproblem (LOP) in primaler Normalform)

Minimiere $\mathbf{c}^\top \mathbf{x}$ *u. d. N.*

$$\mathbf{x} \geq 0, \quad A\mathbf{x} = \mathbf{b},$$

wobei $\mathbf{x}, \mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$ *und* $A \in \mathbb{R}^{m \times n}$.

Alle LP sind äquivalent zu Problem in primaler Normalform:

- Eliminiere freie Variable x_i, $i \notin I$, durch Aufspaltung:

$$x_i = x_i^+ - x_i^-, \quad x_i^+ \geq 0,\ x_i^- \geq 0.$$

- Eliminiere Ungleichung $\sum_{j=1}^n a_{ij} x_j \begin{Bmatrix} \leq \\ \geq \end{Bmatrix} b_i$, $i = 1, \ldots, m$, durch Einführung einer **Schlupfvariable** $s_i \geq 0$:

$$\sum_{j=1}^n a_{ij} x_j \pm s_i = b_i \qquad (A \in \mathbb{R}^{m \times n},\ \mathbf{b} \in \mathbb{R}^m).$$

Lösbarkeit des LGS und Rang

Ein LGS $A\mathbf{x} = \mathbf{b}$ ist lösbar $\iff$ $Rang(A) = Rang(A \mid \mathbf{b})$.

Allgemein gilt für $A \in \mathbb{R}^{m \times n}$, dass $Rang(A) \leq \min\{m, n\}$.

Im Fall $m > n$ und $Rang(A) = n$ (nach Elimination linear abhängiger Spalten) gibt es höchstens eine Lösung[1].

Wenn im Fall $m = n$ die Matrix A invertierbar ist, dann gibt es eine eindeutige Lösung.

Bei einer Lösung gibt es nichts zu optimieren!

Der Fall $m < n$ ist von besonderem Interesse $\rightsquigarrow$ Annahme $m < n$.

Annahme $Rang(A) = m$ (durch Elimination linear abhängiger Zeilen)

[1] noch abhängig von $Rang(A \mid \mathbf{b})$

Konvexe Optimierungsprobleme

Ein (SOP) heißt **konvexes Optimierungsproblem** genau dann, wenn f und g konvexe, stetig differenzierbare Funktionen sind und h affin-linear ist.

LOP sind konvexe Optimierungsprobleme!

Konvexe Optimierungsprobleme

Ein (SOP) heißt **konvexes Optimierungsproblem** genau dann, wenn f und g konvexe, stetig differenzierbare Funktionen sind und h affin-linear ist.

LOP sind konvexe Optimierungsprobleme!

Theorem (Globale Minimierer konvexer Zielfunktionen (Wh.!))

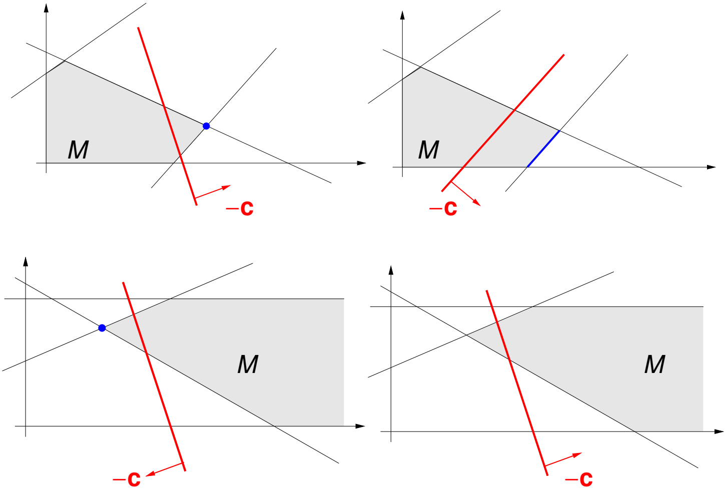
Seien $f : \mathbb{R}^n \to \mathbb{R}$ und $M \subseteq \mathbb{R}^n$ konvex.

Ist $\hat{\mathbf{x}}$ ein stationärer Punkt von f auf M,

dann ist $\hat{\mathbf{x}}$ ein globales Miminimum von f auf M.

Der zulässige Bereich eines LOP kann noch beschränkt oder unbeschränkt sein.

Mögliche Konstellationen von M in der linearen Optimierung



KKT-Bedingungen für lineare Optimierungsprobleme

Lagrange-Funktion:

$$L(\hat{\mathbf{x}}, \lambda, \mu) = \mathbf{c}^\top \mathbf{x} - \lambda^\top \mathbf{x} + \mu^\top (A\mathbf{x} - \mathbf{b})$$

Gleichungs- und Ungleichungsbedingungen:

- $\mathbf{c} - \lambda + A^\top \mu = \mathbf{0}_n$
- $A\hat{\mathbf{x}} - \mathbf{b} = \mathbf{0}_m$
- $\hat{\mathbf{x}} \geq \mathbf{0}_n$
- $\lambda \geq \mathbf{0}_n$
- $\hat{x}_i \hat{\lambda}_i = 0 \quad \text{für} \quad i = 1, \ldots, n$

Für eine Lösung des linearen Optimierungsproblems sind diese 5 Bedingungen notwendigerweise erfüllt.

Existenz und Eigenschaften der Optimallösung eines LOP

Theorem (Fundamentalsatz der linearen Programmierung)

Sei ein LOP in primaler Normalform,

$$\textit{Minimiere } \mathbf{c}^\top \mathbf{x} \quad \textit{u.d.N. } \mathbf{x} \geq \mathbf{0}_n,\ A\mathbf{x} = \mathbf{b},$$

gegeben mit $\mathbf{x}, \mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$ *und* $A \in \mathbb{R}^{m \times n}$.
Des Weiteren sei der zulässige Bereich $M = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}_n\}$ *nichtleer.*

Dann gelten:

(a) *Entweder ist die Zielfunktion auf M nach unten unbeschränkt oder es existiert mindestens eine Optimallösung, darunter ist mindestens ein Eckpunkt von M.*

(b) *Ist M beschränkt, dann existiert eine Optimallösung $\hat{\mathbf{x}} \in M$ und $\hat{\mathbf{x}}$ ist optimal genau dann, wenn*

$$\hat{\mathbf{x}} = \sum_{i=1}^{N} \lambda_i \mathbf{x}^i \quad \textit{mit } \sum_{i=1}^{N} \lambda_i = 1,\ \lambda_i \geq 0,\ i = 1, \ldots, N. \ \textit{(Konvexkombination)}$$

Aussage (b) motiviert das Simplex-Verfahren.

Subvektor und Submatrix

Notation:

Gegeben sei eine Indexmenge B.

Sei $\mathbf{x} \in \mathbb{R}^n$.
Dann ist $\mathbf{x}_B$ der (Sub-)Vektor mit den Komponenten x_j, $j \in B$.

Sei $A = (\mathbf{a}^1, \dots, \mathbf{a}^j, \dots, \mathbf{a}^n) \in \mathbb{R}^{m \times n}$.
Dann ist A_B die (Sub-)Matrix mit Spalten $\mathbf{a}^j$, $j \in B$.

Ecken

Definition (Ecke als zulässige Basislösung)

Seien $A \in \mathbb{R}^{m \times n}$ mit $Rang(A) = m$, $\mathbf{b} \in \mathbb{R}^m$ und $M = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}_n\}$ gegeben.

$\mathbf{x} \in M$ heißt **zulässige Basislösung** (Ecke, Basisvektor) **von M**, falls die Spalten von $A_{\tilde{B}}$ zu einer **Basis von** x, $\tilde{B} = \{j = \{1, \ldots, n\} \mid x_j > 0\}$, linear unabhängig sind.

Man kann zeigen, dass dies zur geometrischen Definition einer Ecke äquivalent ist.

Erweitere $A_{\tilde{B}}$ zu einer $m \times m$-Matrix A_B, setze $N := \{1, \ldots, n\} \setminus B$,

$$\mathbf{b} = A\mathbf{x} = A_B\mathbf{x}_B + A_N\mathbf{x}_N \quad \Longrightarrow \quad x_B := A_B^{-1}\mathbf{b} - A_B^{-1}A_N\mathbf{x}_N.$$

Setze $\mathbf{x}_N = \mathbf{0}_{n-m}$. Falls $\mathbf{x}_B \geq \mathbf{0}_m$, ist $\mathbf{x}$ zulässige Basislösung, ansonsten wähle andere linear unabhängige Spalten mit Indexmenge B.

Polyeder und Basisvektoren

- Eine Menge der Form

$$M = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} = \mathbf{b}\,,\ \mathbf{x} \geq \mathbf{0}_n\}$$

  mit $A \in \mathbb{R}^{m \times n}$, $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$ heißt **Polyeder in Normalform.**
- Ein zulässiger Punkt $\mathbf{x} \in M$ heißt **Basisvektor** oder **zulässige Basislösung** von M, wenn eine aus m Elementen bestehende Indexmenge existiert, so dass die Spaltenvektoren $\mathbf{a}^i$ mit $i \in B$ linear unabhängig sind und $x_j = 0$ für alle $j \notin B$.
- Hat A den Rang m, so gilt:
  - Die Basisvektoren sind die **Ecken** des Polyeders.
  - M besitzt mindestens einen und höchstens endlich viele Basisvektoren.
  - Die Lösung des linearen Optimierungsproblems $\min \mathbf{c}^\top \mathbf{x}$ in M ist einer der Basisvektoren.

Das primale Simplexverfahren - Phasen

Eckpunkte sind algebraisch charakterisiert über linear unabhängige Spalten von A

$\rightsquigarrow$ Simplexverfahren (George B. Dantzig 1947)

Idee:
Gehe von Eckpunkt des zulässigen Bereichs zu benachbartem Eckpunkt
Minimiere dabei Zielfunktionswert
Iteriere bis optimaler Eckpunkt erreicht

Phasen:

0) Transformiere das vorliegende LOP auf primale Normalform.

1) Bestimme zulässige Basislösung (Ecke) $\mathbf{x}$ mit Basisindexmenge B, Nichtbasismenge N, Basisvariable $\mathbf{x}_B \geq \mathbf{0}_m$ und Nichtbasisvariable $\mathbf{x}_N = \mathbf{0}_{n-m}$.

2) Bestimme solange Ecken $\mathbf{x}^+$ mit Indexmengen B^+ und N^+ bis Optimum erreicht oder entschieden werden kann, dass Problem keine Lösung hat.

Das Simplexverfahren als Active-Set-Methode

- Die Lösung des Optimierungsproblems ist theoretisch einfach: Man bestimmt alle Basisvektoren (jeweils als Lösung eines linearen Gleichungssystems der Größe $m \times m$), jeweils ihren Funktionswert und davon das Minimum. (Das LOP kann aber auch unbeschänkt sein.)
- Allerdings kann es exponentiell viele Basisvektoren geben, z.B. hat der Einheitswürfel in $\mathbb{R}^n$ bereits 2^n Ecken.
- Bei einem Basisvektor sind m der Ungleichungs-Nebenbedingungen inaktiv und die übrigen $n - m$ aktiv.
- Die Indexmengen N der aktiven und B der inaktiven Nebenbedingungen sollen in jedem Schritt so verändert werden, dass die Lösung nach wenigen Schritten gefunden wird. Dabei wird immer eine Bedingung aktiv und dafür eine andere inaktiv.
- Die Basismatrix A_B besteht aus den zu B gehörenden Spalten von A, die Nichtbasismatrix A_N aus den übrigen.

Herleitung eines Simplex-Schritts (1)

Sei $\mathbf{x}$ eine Basislösung und $\mathbf{z} = (\mathbf{z}_B, \mathbf{z}_N)^\top$ ein beliebiger zulässiger Punkt. Es sollen

$$\mathbf{c}^\top \mathbf{x} \quad \text{und} \quad \mathbf{c}^\top \mathbf{z}$$

verglichen werden. Dazu erhält man aus $A\mathbf{z} = \mathbf{b}$ zunächst

$$A_B \mathbf{z}_B + A_N \mathbf{z}_N = \mathbf{b}$$

und daraus

$$\mathbf{z}_B = A_B^{-1}\mathbf{b} - A_B^{-1} A_N \mathbf{z}_N \,.$$

Daraus folgt:

$$\begin{aligned}
\mathbf{c}^\top \mathbf{z} &= \mathbf{c}_B^\top \mathbf{z}_B + \mathbf{c}_N^\top \mathbf{z}_N \\
&= \mathbf{c}_B^\top (A_B^{-1}\mathbf{b} - A_B^{-1} A_N \mathbf{z}_N) + \mathbf{c}_N^\top \mathbf{z}_N \\
&= \mathbf{c}_B^\top \mathbf{x}_B + (\mathbf{c}_N^\top - \mathbf{c}_B^\top A_B^{-1} A_N)\mathbf{z}_N \\
&= \mathbf{c}^\top \mathbf{x} + \underbrace{(\mathbf{c}_N - A_N^\top (A_B^\top)^{-1} \mathbf{c}_B)}_{\mathbf{u}_N}{}^\top \mathbf{z}_N
\end{aligned}$$

Herleitung eines Simplex-Schritts (2)

Der Vektor $\mathbf{u}_N$ (Kostenänderung) berechnet sich durch Lösen von

$$A_B^\top \mathbf{y} = \mathbf{c}_B$$

und dann (mit $\mathbf{a}^j$, der j-ten Spalte von A) durch

$$u_j = c_j - (\mathbf{a}^j)^\top \mathbf{y} \quad \text{für } j \in N\,.$$

Sind alle $u_j \geq 0$, so ist $\mathbf{x}$ das gesuchte Minimum. Falls nicht, ersetzt man ein $q \in N$ mit $u_q < 0$ folgendermaßen:

$$x_q^+ = t \geq 0 \quad \text{und} \quad x_j^+ = 0 \quad \text{für alle } j \in N \setminus \{q\}\,.$$

Einsetzen in $A\mathbf{x}^+ = \mathbf{b}$ liefert $A_B \mathbf{x}_B^+ + t\,\mathbf{a}_q = \mathbf{b}$ und damit

$$\mathbf{x}_B^+ = \mathbf{x}_B - t\,A_B^{-1}\mathbf{a}^q$$

Dazu löst man zunächst $A_B \mathbf{d} = \mathbf{a}^q$.

Herleitung eines Simplex-Schritts (3)

Gilt $d_i \leq 0$ für alle $i \in I$, so ist das Problem nicht lösbar. Ansonsten erfordert die Bedingung $\mathbf{x}^+ \geq \mathbf{0}_n$

$$x_i - t \cdot d_i \geq 0 \quad \text{für} \quad i \in B \;\text{ mit }\; d_i > 0$$

und $t \geq 0$. Damit diese Bedingung erfüllt ist, wählt man

$$t = \min_{i \in B, d_i > 0} \frac{x_i}{d_i} = \frac{x_p}{d_p} \, .$$

Damit ist $x_p = 0$ und man entfernt q aus B und fügt es in N hinzu.

Wegen

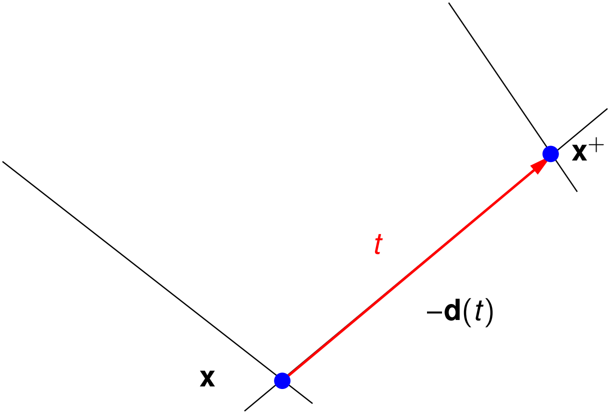
$$\mathbf{c}^\top \mathbf{x}^+ = \mathbf{c}^\top \mathbf{x} + t \cdot u_q$$

ist

$$\mathbf{c}^\top \mathbf{x}^+ \leq \mathbf{c}^\top \mathbf{x} \, .$$

Falls $\mathbf{x}$ nicht degeneriert ist, gilt sogar $<$.

Idee des Basiswechsels beim Simplex



Es wird entlang einer Kantenhalbgerade $\mathbf{z}(t) = \mathbf{x} - t\mathbf{d}$, mit Suchrichtung $-\mathbf{d}$ und Schrittweite $t \geq 0$ gesucht, so dass die Zielfunktion monoton fällt.

Simplex-Algorithmus (1)

1. Starte mit einem Basisvektor $\mathbf{x}^{[0]} \in M$ mit Indexmenge $B^{[0]}$, $N^{[0]} = \{1, \dots, n\} \setminus B^{[0]}$.
   Setze $A_B^{[0]} = (\mathbf{a}^i)_{i \in B^{[0]}}$ und $k = 0$.
2. Löse das lineare Gleichungssystem $(A_B^{[k]})^\top \mathbf{y} = \mathbf{c}_{B^{[k]}}$.
3. Berechne $\zeta_j^{[k]} = -u_j^{[k]} = -c_j + (\mathbf{a}^j)^\top \mathbf{y}\ ,\ j \in N^{[k]}$.
4. Falls $\zeta_j^{[k]} \leq 0$ für alle $j \in N^{[k]}$: Ende (Minimum gefunden)
5. Wähle $q^{[k]} \in N^{[k]}$ mit $\zeta_{q^{[k]}}^{[k]} > 0$. ($q$ Index Pivotspalte)
6. Löse das lineare Gleichungssystem $A_B^{[k]} \mathbf{d}^{[k]} = \mathbf{a}^{q^{[k]}}$
7. Falls $\mathbf{d}_i^{[k]} \leq 0$ für alle $i \in B^{[k]}$: Ende (kein Minimum)
8. Bestimme $t^{[k]}$ und $p^{[k]} \in B^{[k]}$ mit $d_{p^{[k]}}^{[k]} > 0$ aus:

$$t^{[k]} = \min_{i \in B^{[k]}, d_i^{[k]} > 0} \frac{x_i^{[k]}}{d_i^{[k]}} = \frac{x_{p^{[k]}}^{[k]}}{d_{p^{[k]}}^{[k]}}$$

(p Index Pivotzeile)

Simplex-Algorithmus (2)

9. Berechne

$$x_i^{[k+1]} = \begin{cases} x_i^{[k]} - t^{[k]} d_i^{[k]} & \text{falls } i \in B^{[k]}\,,\; i \neq p^{[k]} \\ t^{[k]} & \text{falls } i = q^{[k]} \\ 0 & \text{sonst} \end{cases}$$

und

$$B^{[k+1]} = (I^{[k]} \cup \{q^{[k]}\}) \setminus \{p^{[k]}\}$$
$$N^{[k+1]} = \{1, \ldots, n\} \setminus B^{[k+1]}$$
$$A_B^{[k+1]} = (\mathbf{a}^i)_{i \in B^{[k+1]}}$$

10. Setze $k := k + 1$ und gehe zu 2.

Bemerkungen

- **Wahl des Start-Basisvektors:** Entweder durch Schlupfvariablen (bei vorheriger Transformation des Problems auf Standardform) oder durch Lösen des linearen Hilfsproblems $\min \sum_{i=1}^{m} z_i$ unter der Nebenbedingung

$$A\mathbf{x} + \mathbf{z} = \mathbf{b} \ , \ x_1, \ldots, x_n, z_1, \ldots, z_m \geq 0$$

  mit Startwert $(\mathbf{x}, \mathbf{z})^\top = (\mathbf{0}_n, \mathbf{b})^\top$ (oBdA: $\mathbf{b} \geq 0$).
- **Wahl von $\mathbf{q}^{[k]}$:** Entweder durch

$$\zeta_{q^{[k]}}^{[k]} = \max_{j \in N^{[k]}} \zeta_j^{[k]}$$

  oder (zur Vermeidung von Zyklen bei entarteten Basisvektoren) durch die Wahl des kleinstmöglichen $q^{[k]} \in N^{[k]}$ in Schritt 5 sowie des kleinstmöglichen $p^{[k]} \in B^{[k]}$ in Schritt 8.
- **Berechnen von $\mathbf{d}^{[k]}$ und $\mathbf{y}^{[k]}$:** Kein Lösen der Gleichungssysteme, sondern Rang-1-Updates mit der Sherman-Morrison-Woodbury-Formel.

Das primale Simplexverfahren - Algorithmus (für manuelle Rechnung)

Algorithmus (*Primales Simplexverfahren für Tableau*)

(0) *Phase 1:*
Bestimme zulässige Basislösung x mit Indexmengen B, N, Basismatrix A_B, Nichtbasismatrix A_N, Basisvariable $x_B \geq 0$ und Nichtbasisvariable $x_N = 0$ der primalen Normalform.

(1) *Zwischenwerte:*
Berechne $\Gamma_B^N := A_B^{-1} A_N$, $\beta_B := A_B^{-1} b$ und die negativen reduzierten Kosten $\zeta_N^\top := c_B^\top \Gamma_B^N - c_N^\top$.

(2) *Überprüfung der Optimalität:*
Falls $\zeta_j \leq 0$ für alle $j \in N$ gilt, so beende das Verfahren. Die aktuelle Basislösung $x_B = \beta_B$, $x_N = 0$ ist optimal. Der Zielfunktionswert ist $\delta = c_B^\top \beta_B$.

(3) *Überprüfung auf Unbeschränktheit:*
Gibt es einen Index q mit $\zeta_q > 0$ und $\gamma_{iq} \leq 0$ für alle $i \in B$, so besitzt das Problem keine Lösung, STOP.

(4) *Bestimmung des Pivotelements:*
Wähle einen Index q (für Pivotspalte) mit $\zeta_q > 0$. ($d_i = \gamma_{iq}$, $i \in B$)
Wähle einen Index p (für Pivotzeile) mit $\frac{\beta_p}{\gamma_{pq}} = \min\left\{ \frac{\beta_i}{\gamma_{iq}} \,\middle|\, \gamma_{iq} > 0, i \in B \right\}$.

(5) *Basiswechsel durchführen:*
Setze $B := (B \setminus \{p\}) \cup \{q\}$, $N := (N \setminus \{q\}) \cup \{p\}$.

(6) *Gehe zu (1).*

Simplextableau

Der Basiswechsel muss so erfolgen, dass zumindest:

$$
\begin{array}{ll}
Ax = b, x \geq 0 \quad \Rightarrow \quad Ax^+ = b, x^+ \geq 0 & \text{(Zulässigkeit)}, \\
\qquad\qquad c^\top x^+ \leq c^\top x & \text{(Abstieg im Zielfunktionswert).}
\end{array}
$$

$$
\begin{aligned}
x_B &= (A_B)^{-1} b - (A_B)^{-1} A_N x_N =: \beta_B - \Gamma_B^N x_N, \\
c^\top x &= c_B^\top x_B + c_N^\top x_N = c_B^\top \beta_B - \left(c_B^\top \Gamma_B^N - c_N^\top \right) x_N =: \delta - \zeta_N^\top x_N
\end{aligned}
$$

fasst man zusammen in einem Tableau:

	x_N		
x_B	$\Gamma_B^N = (A_B)^{-1} A_N$	$\beta_B = (A_B)^{-1} b$	$\frac{\beta_B}{\Gamma_B^q}$
	$\zeta_N^\top = c_B^\top \Gamma_B^N - c_N^\top$	$c_B^\top x_B$	

Updateformeln

Neuberechnung des Tableaus in jedem Schritt aufwendig

Der Tableauinhalt für die neue Basis zur Indexmenge B^+ ergibt sich aus dem alten Tableauinhalt (mit Daten γ_{ij}, β_i, ζ_j) nach folgendem Schema:

	$x_j, j \in N \setminus \{q\}$	x_p	
$x_i, i \in B \setminus \{p\}$	$\gamma_{ij} - \frac{\gamma_{iq}\gamma_{pj}}{\gamma_{pq}}$	$-\frac{\gamma_{iq}}{\gamma_{pq}}$	$\beta_i - \frac{\gamma_{iq}\beta_p}{\gamma_{pq}}$
x_q	$\frac{\gamma_{pj}}{\gamma_{pq}}$	$\frac{1}{\gamma_{pq}}$	$\frac{\beta_p}{\gamma_{pq}}$
	$\zeta_j - \frac{\zeta_q\gamma_{pj}}{\gamma_{pq}}$	$-\frac{\zeta_q}{\gamma_{pq}}$	$\delta - \frac{\zeta_q\beta_p}{\gamma_{pq}}$

Komplexität des Simplex-Verfahrens

- Pro Iterationsschritt beträgt der Rechenaufwand $O(n^2)$.
- Eine obere Schranke für die Anzahl der Ecken und damit der Iterationen ist $\binom{n}{m}$, die Anzahl der m-elementigen Teilmengen von n.
- Das Simplex-Verfahren konvergiert meist sehr schnell (in $O(n)$ Iterationen), aber es gibt Beispiele (z.B. von Klee und Minty), in denen fast alle Ecken durchlaufen werden.
- 1984: Veröffentlichung der Interior Point Method von N. Karmakar mit polynomialem Rechenaufwand als Alternative zum Simplex-Verfahren

Beispiel von Klee und Minty

3-dimensional:

$$\text{Maximiere} \qquad 100x_1 + 10x_2 + x_3$$

unter den Nebenbedingungen

$$\begin{array}{llll} x_1 & & & \leq & 1 \\ 20x_1 & +x_2 & & \leq & 100 \\ 200x_1 & +20x_2 & +x_3 & \leq & 10000 \end{array}$$

$$x_1, x_2, x_3 \geq 0$$

n-dimensional:

$$\text{Maximiere} \qquad \sum_{j=1}^{n} 10^{n-j} x_j$$

unter den Nebenbedingungen

$$2 \sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \leq 100^{i-1} \quad \text{für} \quad 1 \leq i \leq n$$

Active-Set-Methoden für quadratische Optimierungsprobleme
Quadratische nichtlineare Optimierung

Ein **quadratisches Optimierungsproblem** mit linearen Restriktionen hat die Form

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \mathbf{q}^\top\mathbf{x} + c$$

unter den Nebenbedingungen

$$G\mathbf{x} = \mathbf{b} \quad \text{und} \quad U\mathbf{x} \le \mathbf{r}\,.$$

Dabei ist $Q \in \mathbb{R}^{n\times n}$ symmetrisch und

$$\mathbf{x},\mathbf{q} \in \mathbb{R}^n\,,\ c \in \mathbb{R},\ G \in \mathbb{R}^{m\times n}\,,\ \mathbf{b} \in \mathbb{R}^m\,,\ U \in \mathbb{R}^{p\times n}\,,\ \mathbf{r} \in \mathbb{R}^p.$$

Sind Nebenbedingungen der Form $U\mathbf{x} \le \mathbf{r}$ vorhanden, hat das restringierte quadratische Problem Gleichungs- und Ungleichungsnebenbedingungen, falls nicht, nur Gleichungsnebenbedingungen.

KKT-Bedingungen für quadratische Optimierungsprobleme

Lagrange-Funktion:

$$L(\mathbf{x},\lambda,\mu) = \frac{1}{2}\mathbf{x}^\top Q\mathbf{x} + \mathbf{q}^\top\mathbf{x} + c + \lambda^\top(U\mathbf{x}-\mathbf{r}) + \mu^\top(G\mathbf{x}-\mathbf{b})$$

Gleichungsbedingungen:

- $Q\mathbf{x} + \mathbf{q} + G^\top\mu = \mathbf{0}_n$
- $G\mathbf{x} = \mathbf{b}$

Gleichungs- und Ungleichungsbedingungen:

- $Q\mathbf{x} + \mathbf{q} + G^\top\mu + U^\top\lambda = \mathbf{0}_n$
- $G\mathbf{x} = \mathbf{b}$
- $U\mathbf{x} \le \mathbf{r}$
- $\lambda_i \ge 0 \quad \text{für} \quad i = 1,\dots,p$
- $(\mathbf{u}_i\mathbf{x} - r_i)\lambda_i = 0 \quad \text{für} \quad i = 1,\dots,p$

  (Dabei bezeichnet $\mathbf{u}_i$ die i-te Zeile von U.)

Minimierung unter linearen Gleichungsrestriktionen

Die Optimalitätsbedingungen 2. Ordnung besagen, dass eine Lösung des KKT-Systems ein globales Minimum von f ist, wenn Q positiv definit auf dem Kern von G ist und der Rang von G gleich m ist.

Die Lösung des KKT-Systems erhält man durch Lösen des linearen Gleichungssystems

$$\begin{pmatrix} Q & G^\top \\ G & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mu \end{pmatrix} = \begin{pmatrix} -\mathbf{q} \\ \mathbf{b} \end{pmatrix}$$

Möglichkeiten der Lösung:

- Gauß-Algorithmus
- Nullraum-Methode
- Schur-Komplement-Methode

Beispiel (1)

Aufgabe: Minimiere die Funktion

$$f(\mathbf{x}) = 3x_1^2 + 2x_1x_2 + x_1x_3 + 2.5x_2^2 + 2x_2x_3 + 2x_3^2 - 8x_1 - 3x_2 - 3x_3$$

mit den Nebenbedingungen

$$\begin{aligned} x_1 + x_3 &= 3, \\ x_2 + x_3 &= 0. \end{aligned}$$

Man erhält:

$$Q = \begin{pmatrix} 6 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 4 \end{pmatrix}, \quad \mathbf{q} = \begin{pmatrix} -8 \\ -3 \\ -3 \end{pmatrix}, \quad G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$$

Beispiel (2)

Die KKT-Bedingungen ergeben das Gleichungssystem:

$$\begin{pmatrix} 6 & 2 & 1 & 1 & 0 \\ 2 & 5 & 2 & 0 & 1 \\ 1 & 2 & 4 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \mu_1 \\ \mu_2 \end{pmatrix} = \begin{pmatrix} 8 \\ 3 \\ 3 \\ 3 \\ 0 \end{pmatrix}$$

Daraus erhält man die Lösungen:

$$\hat{\mathbf{x}} = (2, -1, 1)^\top \quad \text{und} \quad \boldsymbol{\mu} = (-3, 2)^\top$$

Nachteil des Gauß-Algorithmus:

- Kein Ausnutzen der Symmetrie ($\Rightarrow$ symmetrische indefinite Zerlegung verwenden)
- Rechenaufwand: $O((n+m)^3)$

Minimierung unter linearen Ungleichungsrestriktionen

Hauptschwierigkeit:

- Herausfinden, welche Ungleichungsbedingungen aktiv sind

Idee:

- Schätze eine Menge aktiver Bedingungen durch Bestimmung eines zulässigen Punktes
- Löse das zugehörige (Hilfs-)Optimierungsproblem mit Gleichungsnebenbedingungen
- Füge (abhängig von der Lösung) entweder eine inaktive Nebenbedingung zu den aktiven hinzu oder entferne eine aktive Nebenbedingung
- Wiederhole die beiden letzten Schritte, bis ein Minumum gefunden ist

Die *Active-Set-Methode* gehört zu den *Verfahren zulässiger Richtungen*, weil in jedem Schritt ein zulässiger Punkt und eine zulässige Abstiegsrichtung bestimmt werden.

Beispiel

Minimiere die Funktion

$$f(\mathbf{x}) = \frac{1}{2}x_1^2 + \frac{1}{2}x_2^2 + x_1 + x_2$$

mit den Nebenbedingungen

$$\begin{aligned} -x_1 - x_2 &\leq 0, \\ x_2 &\leq 2, \\ x_1 + x_2 &\leq 5, \\ -x_1 + x_2 &\leq 2, \\ x_1 &\leq 5, \\ -x_2 &\leq 1. \end{aligned}$$

Bestimmung des Hilfsproblems

- Zu einem zulässigen Punkt $\mathbf{x}^{[k]}$ sucht man eine zulässige Abstiegsrichtung $\mathbf{d}^{[k]}$ durch Lösen eines quadratischen Hilfsproblems $\min f(\mathbf{x}^{[k]} + \mathbf{d})$ bzgl. $\mathbf{d} \in \mathbb{R}^n$ mit Gleichungs-Nebenbedingungen $G\mathbf{x} = \mathbf{b}$ und $U^{[k]}\mathbf{x} = \mathbf{r}^{[k]}$ (alle aktiven Ungleichungsnebenbedingungen).
- Dieses Problem löst man durch Umformen des KKT-Systems

$$\begin{pmatrix} Q & G^\top & (U^{[k]})^\top \\ G & 0 & 0 \\ U^{[k]} & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x}^{[k]} + \mathbf{d}^{[k]} \\ \boldsymbol{\mu} \\ \boldsymbol{\lambda}^{[k]} \end{pmatrix} = \begin{pmatrix} -\mathbf{q} \\ \mathbf{b} \\ \mathbf{r}^{[k]} \end{pmatrix}$$

zu

$$\begin{pmatrix} Q & G^\top & (U^{[k]})^\top \\ G & 0 & 0 \\ U^{[k]} & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{d}^{[k]} \\ \boldsymbol{\mu} \\ \boldsymbol{\lambda}^{[k]} \end{pmatrix} = \begin{pmatrix} -\nabla f(\mathbf{x}^{[k]})^\top \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}.$$

Vier mögliche Ergebnisse des Hilfsproblems

1. $\mathbf{d}^{[k]} = \mathbf{0}_n$ und $\lambda^{[k]} \geq \mathbf{0}$:
   KKT-Bedingungen des eigentlichen Problems erfüllt.
2. $\mathbf{d}^{[k]} = \mathbf{0}_n$ und $\lambda_m = \min_{i \in N} \lambda_i^{[k]} < 0$:
   Kein KKT-Punkt des eigentlichen Problems, keine weitere Minimierung mit diesen aktiven Ungleichungen möglich.
   $\Rightarrow$ Entferne eine aktive Nebenbedingung aus N.
3. $\mathbf{d}^{[k]} \neq \mathbf{0}_n$ und $\mathbf{x}^{[k]} + \mathbf{d}^{[k]}$ zulässig:
   Akzeptieren des neuen Punktes und Beibehalten der Indexmenge
4. $\mathbf{d}^{[k]} \neq \mathbf{0}_n$ und $\mathbf{x}^{[k]} + \mathbf{d}^{[k]}$ nicht zulässig:
   Eine inaktive Ungleichung wird jetzt verletzt.
   $\Rightarrow$ Reduziere die Schrittweite, so dass man einen zulässigen Punkt erhält.

Active-Set-Methode für quadratische Probleme (1)

1. Berechne einen zulässigen Startpunkt $\mathbf{x}^{[0]}$ und bestimme die Matrix $U^{[0]}$, deren Zeilen $\mathbf{u_i}$ die aktiven Ungleichungsnebenbedingungen sind.
   Setze $N = \{i \,|\, \mathbf{u_i}\mathbf{x}^{[0]} = r_i^{[0]}\}$ und $k = 0$.
2. Löse das quadratische Hilfsproblem:

$$\begin{pmatrix} Q & G^\top & (U^{[k]})^\top \\ G & 0 & 0 \\ U^{[k]} & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{d}^{[k]} \\ \boldsymbol{\mu} \\ \lambda^{[k]} \end{pmatrix} = \begin{pmatrix} -\nabla f(\mathbf{x}^{[k]})^\top \\ 0 \\ 0 \end{pmatrix}$$

3. Falls $\mathbf{d}^{[k]} = \mathbf{0}_n$ und $\lambda^{[k]} \geq 0$, Ende des Verfahrens.
4. Falls $\mathbf{d}^{[k]} = \mathbf{0}_n$ und $\lambda_q = \min_{i \in N} \lambda_i^{[k]} < 0$, setze
   $N = N \setminus \{q\}$ und bestimme $U^{[k+1]}$ aus $U^{[k]}$ durch Streichen der entsprechenden Zeile.
   Setze $\mathbf{x}^{[k+1]} = \mathbf{x}^{[k]}$, $k = k + 1$ und gehe zu 2.

Active-Set-Methode für quadratische Probleme (2)

5. Falls $\mathbf{d}^{[k]} \neq \mathbf{0}_n$ und $\mathbf{x}^{[k]} + \mathbf{d}^{[k]}$ zulässig, setze $\mathbf{x}^{[k+1]} = \mathbf{x}^{[k]} + \mathbf{d}^{[k]}$, $U^{[k+1]} = U^{[k]}$, $k = k + 1$ und gehe zu 2.
6. Falls $\mathbf{d}^{[k]} \neq \mathbf{0}_n$ und $\mathbf{x}^{[k]} + \mathbf{d}^{[k]}$ nicht zulässig, berechne

$$
\begin{aligned}
t^{[k]} &= \frac{r_p - \mathbf{u}_p \mathbf{x}^{[k]}}{\mathbf{u}_p \mathbf{d}^{[k]}} = \min_{i \notin N \text{ mit } \mathbf{u}_i \mathbf{d}^{[k]} > 0} \frac{r_i - \mathbf{u}_i \mathbf{x}^{[k]}}{\mathbf{u}_i \mathbf{d}^{[k]}}, \\
\mathbf{x}^{[k+1]} &= \mathbf{x}^{[k]} + t^{[k]} \mathbf{d}^{[k]}, \\
N &= N \cup \{p\}.
\end{aligned}
$$

Setze $k = k + 1$ und gehe zu 2.

Konvergenz(geschwindigkeit)

- Unter der Voraussetzung, dass Q positiv definit für alle $\mathbf{x} \in Kern(G)$ ist und dass alle Zeilen von $\begin{pmatrix} G \\ U \end{pmatrix}$ linear unabhängig sind, bestimmt die Active-Set-Methode in endlich vielen Schritten die eindeutig bestimmte Lösung des Optimierungsproblems.
- Wie beim Simplex-Verfahren gibt es exponentiell in n viele Möglichkeiten, die aktiven Nebenbedingungen zu bestimmen (Worst-Case-Komplexität). In der Praxis erhält man jedoch meist ein sehr viel besseres Verhalten.
- Als zusätzliches Abbruchkriterium sollte aber in jedem Fall eine maximale Anzahl an Schritten vorgegeben werden.
- Den Startwert $\mathbf{x}^{[0]}$ berechnet man beispielsweise mit Phase 1 des Simplex-Verfahrens.

Entwicklung der Verfahren

- Entwicklung der ersten Active-Set-Methode von Fletcher (1971)
- Implementierung der Methode von Gill und Murray (1978) in Matlab
- Innere-Punkt-Methoden besitzen polynomielle Worst-Case-Komplexität und sind daher für sehr große Probleme meist besser geeignet.
  In Matlab ist der Algorithmus von Gould und Toint (2004) implementiert.

Überblick

1 Einführung

2 Unrestringierte Optimierung

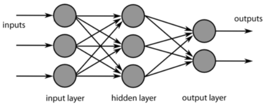
3 Wichtige Verfahren der numerischen Optimierung

4 Restringierte Optimierung

5 **Zusammenfassung - Ausblick und Wiederholung**
- Aus- und Einblick: Maschinelles Lernen, Data Science
- Aus- und Einblick: Optimalsteuerung
- Weiterer Ausblick
- Quellen

Maschinelles Lernen, Data Science: Überblick

- *Machine Learning*
  - Unüberwachtes Lernen
    erzeugt zu Daten statistisches Modell
    Z.B. Hidden Markov Model (HMM), Hauptkomponentenanalyse
  - Überwachtes Lernen
    Z.B. *Support Vector Machine*, *Reinforcement Learning*, *Active Learning* (benutze Abfragen)
- Data-mining (*Statistical Learning*):
  Auffinden neuer Gesetzmäßigkeiten
  - auch hier ML-Algorithmen
  - Regression
- Deep learning
  - Klasse "künstlicher neuronaler Netze"
  - *Hidden Layers* zwischen Ein- und Ausgabeschicht
  - zu unterscheiden von biologischen neuronalen Netzen und deren Simulation



Quelle: Wikipedia