

Nichtlineare Optimierungsprobleme mit Komplexität

Definition eines Nichtlinearen Optimierungsproblems (NLP)

$$\min_{x \in S} f(x) \quad \text{bzw.} \quad \min f(x) \text{ s.d. } x \in S$$

wobei die zulässige Menge $S \subseteq \mathbb{R}^n$ typischerweise definiert ist durch

$$S \equiv \{x \in \mathbb{R}^n : h(x) = 0, c(x) \leq 0\}$$

für Gleichungs- und Ungleichungsrestriktionen definiert durch

$$h : \mathbb{R}^n \rightarrow \mathbb{R}^m \quad \text{und} \quad c : \mathbb{R}^n \rightarrow \mathbb{R}^p$$

Falls $m = 0 = p$ heisst das NLP Problem *unrestricted*. Müssen einige Komponenten x_j ganzzahlig sein so spricht man von einem **MINLP** in Analogie zum linearen **MILP**.

- 172 -

Entsprechendes Entscheidungsproblem:

Für welche Schranke φ hat das System algebraischer Gleichungen und Ungleichungen

$$f(x) \leq \varphi, \quad h(x) = 0, \quad c(x) \leq 0$$

überhaupt eine Lösung $x \in \mathbb{R}^n$?

Komplexitätsvergleich

Das jeweilige Entscheidungsproblem ist nur unwesentlich einfacher als das Optimierungsproblem, da letzteres durch eine Folge von Entscheidungsproblemen mit variierendem φ approximativ gelöst werden kann.

Bemerkung

Abgesehen vom unten besprochenen konvexen Fall ist schon das Entscheidungsproblem auch ohne Ganzzahligkeitsbedingung **NP** schwer. Unter Optimierern gehen die Meinungen über die praktische Bedeutung dieser theoretischen Aussage weit auseinander.

- 173 -

Wirkung von Nichtlinearität und Nichtkonvexität II

Im Falle reiner Gleichungssysteme wurde festgestellt, dass nichtlineare Probleme, für die alle Funktionen stetig differenzierbar sind, im *lokalen* Sinne (d.h. bei Vorgabe eines Anfangspunktes in der unmittelbaren Nähe einer Lösung) nur unwesentlich schwerer als lineare Probleme sind.

Das gilt auch in Kombination mit Ungleichungen. Als Verallgemeinerung von Newton's Methode nähert man dann das gegebenen NLP durch eine Folge von Systemen aus linearen Gleichungen und Ungleichungen an. Bei der direkten Lösung des Optimierungsproblems wird dabei die Zielfunktion quadratisch angenähert. Das führt zu den sogenannten *sukzessiven quadratischen Optimierungsverfahren (SQP)*.

Global, d.h. ohne Vorgabe guter Startwerte, sind nichtlineare Probleme viel schwerer, da schon die Suche nach einem auch nur annäherungsweise zulässigen Vektor einen in der Zahl seiner Komponenten exponentiellen Aufwand verursachen kann.

- 174 -

Wirkung von Nichtlinearität und Nichtkonvexität I

Die scheinbar harmlose polynomiale (Zusatz-)Gleichung

$$x_i(1 - x_i) = 0$$

erzwingt, dass die i -te Variable x_i binär ist, d.h. nur die Werte 0 oder 1 annehmen darf. (Man kann so leicht das klassische Entscheidungsproblem **SAT** als NLP schreiben.)

Ein guter Anfangswert bedeutet hier praktisch die Vorentscheidung, ob x_i nun 0 oder 1 sein soll.

Nur im Falle konvexer NLP (d.h. h muss linear sein, aber f und die p Komponenten von c können allgemeinere konvexe Funktionen sein) werden keine guten Startwerte benötigt. Denn dann sind sowohl die Menge aller zulässigen und insbesondere die Menge aller optimalen Lösungen selbst konvex und es gibt keine lokalen Minima.

- 175 -

Nichtlineare Ausgleichsprobleme

Eine wichtige Klasse (häufig unrestringierter) NLPs sind von der Form:

$$\min f(z) \equiv \frac{1}{2} \|F(z) - y\|^2 = \frac{1}{2} (F(z) - y)^\top (F(z) - y)$$

wobei $F: \mathbb{R}^n \rightarrow \mathbb{R}^m$ mit $m \geq n$ ein an verschiedenen Punkten ausgewertetes mathematisches Modell darstellt. Der Variablenvektor z soll so gewählt werden, dass der Euklidische Abstand $\|F(z) - y\|$ zu 'gemessenen' Daten $y \in \mathbb{R}^m$ möglichst klein ist.

Zum Beispiel könnte man die in Übung 3 Aufgabe 1 betrachteten 'synthetischen' (d.h. nicht wirklich gemessenen sondern künstlich erzeugten) Daten

$$y_i = \frac{1}{1 + 25x_i^2}, \quad x_i = \{-.6, -.3, -.1, 0, .1, .3, .6\}, \quad \text{für } i = 1, \dots, 7$$

auch nichtlinear annähern.

- 176 -

Fortsetzung des Beispiels

Statt die Daten durch eine Linearkombination von Monomen $u_j(x) = x^{j-1}$ oder sonstiger Basisfunktionen könnte man annehmen dass

$$y_i \approx F_i(z) \equiv \varphi(x_i, z) \quad \text{mit} \quad \varphi(x, z) \equiv z_1 + z_2 \cos(z_3 x + z_4)$$

Mit anderen Worten: Wir nutzen eine Kosinusfunktion mit den vier Parametern $z \equiv (z_i)_{i=1, \dots, 4}$ als Modell für unsere Daten.

Offensichtlich ist nun $F(z) - y = (F_i(z) - y_i)_{i=1, \dots, 4}$ nicht mehr linear und entsprechend $f(z) \equiv \|F(z) - y\|^2/2$ auch nicht quadratisch in z . Wegen der Oszillationen der Kosinusfunktion ist dieses Problem auch nicht konvex und hat mehrere lokale Minima.

Nichtlineare Ausgleichsprobleme können entweder mit allgemeinen Algorithmen zur nichtlinearen Optimierung oder mit verschiedenen Varianten des sogenannten *Gauss-Newton* - Verfahrens gelöst werden.

- 177 -

Gauss-Newton

Bei dieser Verallgemeinerung des Newton-Verfahrens wird am jeweiligen Annäherungswert z für einen zunächst beliebigen Schritt s approximiert

$$F(z + s) \approx F_z(s) \equiv F(z) + F'(z)s$$

wobei $F'(z) \in \mathbb{R}^{m \times n}$ wiederum die aus allen ersten partiellen Ableitungen $\partial F_i / \partial z_j$ von F nach z geformte Jacobimatrix darstellt.

Während im Newtonverfahren der Schritte s so gewählt wird, dass das Gleichungssystem $F'(z)s = -F(z)$ exakt erfüllt wird, geht dies im vorliegenden überbestimmten Falle $m \geq n$ im allgemeinen nicht. Hier wird wie beim linearen Ausgleichproblem s so gewählt, dass das Residuum $\|F'(z)s + F(z)\|$ minimiert. Im wohlbestimmten Falle $m = n$ ergibt dies den exakten Newton-Schritt $s = -F'(z)^{-1}F(z)$.

- 178 -

Gauss-Newton (Fortsetzung)

Wiederholung führt hier unter Nutzung der Normalgleichung zur Gauss-Newton - Iteration

$$z \leftarrow z - [F'(z)^\top F'(z)]^{-1} F'(z)^\top F(z)$$

Unter geeigneten Voraussetzungen ergibt sich von guten Anfangspunkten lineare Konvergenz gegen ein lokales Minimum von $f(z) = 1/2 \|F(z) - y\|^2$. Dabei muss gegebenenfalls eine Dämpfung der Schrittweite eingesetzt werden und selbst mit ihr ist Konvergenz von beliebigen Anfangspunkten nicht garantiert.

- 179 -

Lösbarkeit allgemeiner NLPs

Man unterscheidet drei Möglichkeiten

- (i) zulässig $\iff \emptyset \neq S \equiv \{x \in \mathbb{R}^n : h(x) = 0, c(x) \leq 0\}$
- (ii) beschränkt $\iff -\infty < f_* \equiv \inf\{f(x) : x \in S\}$
- (iii) lösbar $\iff \emptyset \neq \operatorname{argmin}(f|S) \equiv \{x \in S : f(x) = f_*\}$

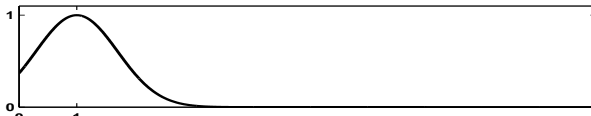
Bei der Linearen Programmierung, d.h. wenn f, c, h linear sind gilt

$$(i) \ \& \ (ii) \implies (iii) \quad \text{sowie} \quad \operatorname{argmin}(f|U \cap S) \subset \operatorname{argmin}(f|S)$$

wobei $f|M$ die Restriktion der Funktion f auf eine beliebige Teilmenge M seines Definitionsbereiches symbolisiert.

- 180 -

Nichtlineares Gegenbeispiel: $S = [0, \infty)$, $f(x) = e^{-(x-1)^2}$



- (iv) zulässig und beschränkt, aber nicht lösbar.
- (v) $x = 0$ ist lokales aber nicht globales Minimum.

Warnung:

Die Möglichkeiten (iv) und (v) können im Allgemeinen durch einen Optimierungsalgorithmus nicht festgestellt werden.

Praktisches Abbruchkriterium:

Gib auf, wenn die an benachbarten zulässigen Punkten erzielbaren Reduktionen des Funktionswertes kleiner als eine vorgegebene Toleranz ist (oder der Algorithmus anderen Hindernissen, wie zum Beispiel singulären Matrizen, begegnet ist.)

- 181 -

Grundlegende algorithmische Herangehensweisen

Lokale Abstiegsmethodik

Ausgehend von $x_0 \in S$ erzeuge eine Folge

$$x_{k+1} = x_k + s_k \quad \text{with} \quad f(x_{k+1}) < f(x_k)$$

so dass hoffentlich für ein offenes U

$$\lim_{k \rightarrow \infty} x_k = x_* \quad \text{with} \quad x_* \in \operatorname{argmin}(f|U \cap S)$$

Globale Optimierungsmethodik

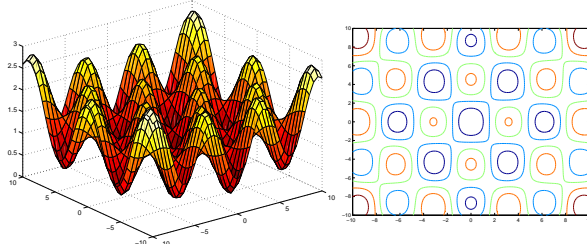
Erzeuge eine endliche Punktwolke $\mathcal{X} = \{x_k\}_{k=1}^K \subset \mathbb{R}^n$ möglicherweise unter Berücksichtigung des "Fitnesswertes" $f(x_k)$ und wähle

$$\tilde{x} \in \operatorname{argmin}(f|S \cap \mathcal{X})$$

See: Evolutionäre Algorithmen = Simulated Annealing
 + Genetic Algorithms (GA)
 + ...

- 182 -

Griewank's function: The GA playground



$$f(x) = 1 + \sum_{i=1}^n \frac{x_i^2}{200} - \prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right)$$

google (Griewank function) $\implies \#1 \in 27$ Kilohits

- 183 -

Comparison between PSA/GAc and GA (Griewank function)

	dim	PSA	GA (20*20)	duGa
Optimum		1.0e-5order	0	0
Success rate	10	0.9	0.0	0.2
Evaluations		3008201	3200400	2676960
Success rate	30	1.0	0.7	0.9
Evaluations		3118041	2922120	1819760

Comparison between PSA/GAc and GA (Rosenbrock function)

Optimum	dim	1.0e-8order	0	0
Success rate	10	1.0	0.0	0.0
Evaluations		2750721	3200400	3200400
Success rate	30	1.0	0.0	0.0
Evaluations		2723441	3200400	3200400

- 184 -

Optimalitätsbedingungen unrestringierten Fall ($m = 0 = p$)

$$0 = \nabla f(x) \equiv \left(\frac{\partial f}{\partial x_i} \right)_{i=1, \dots, n} \equiv \text{Gradient verschwindet}$$

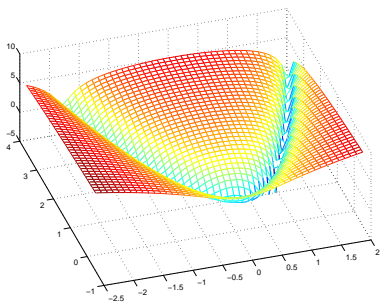
$$\Rightarrow \boxed{\text{Minimiere } f} \text{ ist lokal äquivalent zu } \boxed{\text{löse } g(x) \equiv \nabla f(x) = 0}$$

$$0 \preceq \nabla^2 f(x) \equiv \left(\frac{\partial^2 f}{\partial x_i \partial x_j} \right)_{j=1, \dots, n}^{i=1, \dots, n} \equiv \text{Hessematrix } H(x) \text{ ist positiv semi-definite}$$

$$g(x) = 0 \wedge H(x) \succeq 0 \wedge \det(H(x)) \neq 0 \Rightarrow x \text{ lokales Minimum}$$

- 185 -

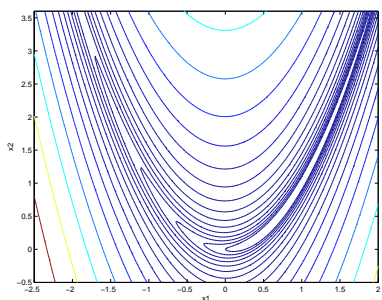
Rosenbrock – Funktion



Grafik: Alt, Walter, Nichtlineare Optimierung

- 186 -

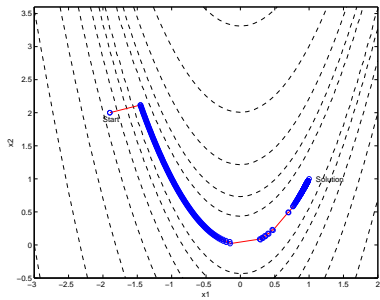
Höhenlinien der Rosenbrock – Funktion



Grafik: Alt, Walter, Nichtlineare Optimierung

- 187 -

Gradienten - Verfahren für Rosenbrock - Funktion

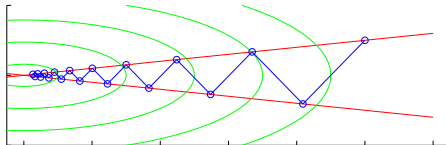


Grafik: Alt, Walter, Nichtlineare Optimierung

Was klemmt beim Steilsten Abstieg (Cauchy,1847) ???

$$x_{k+1} = x_k - \alpha_k g_k \quad \text{mit} \quad \alpha_k \approx \underset{\alpha > 0}{\operatorname{argmin}} (f(x_k - \alpha g_k))$$

Die Berechnung von α_k heisst **line-search** bzw **Strahlsuche**.
 Für $f(x_1, x_2) = \frac{1}{2}(x_1^2 + \kappa x_2^2)$ zeigt die Methode **zigzagging**:



Im allgemeinen Fall ist die Konvergenzrate
 $\|x_k - x_*\| \sim \|x_0 - x_*\| \left(1 - \frac{2}{\kappa + 1}\right)^k \approx \|x_0 - x_*\| (1 - 2\kappa / \kappa)$
 wobei $\kappa = \kappa(H_*) \equiv \|H_*\| \|H_*^{-1}\| = \lambda_{\max}(H_*) / \lambda_{\min}(H_*)$

Bedeutung von Skalierungsinvarianz

Steilster Abstieg funktioniert perfekt wenn

$$\kappa(H(x_*)) = 1 \iff H(x_*) = I$$

oder wenn angewandt auf das transformierte Problem

$$\tilde{f}(z) \equiv f(H_*^{-1/2} z)$$

\implies Newton's Methode \approx Dynamische Transformation

$$x_{k+1} = x_k + \alpha_k s_k \quad \text{mit} \quad H(x_k) s_k = -g_k$$

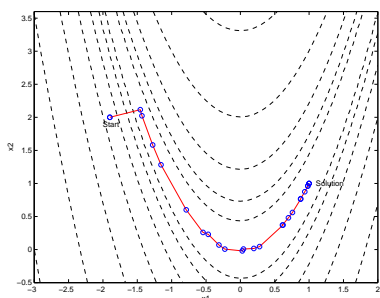
\implies Quasi-Newton Methode, z.B.

$$B_k s_k = -g_k \quad \text{mit} \quad H_k \approx B_k \equiv U(B_{k-1}, s_{k-1}, g_k - g_{k-1})$$

dies ist der einzige Weg zur superlinearen Konvergenz, d.h.

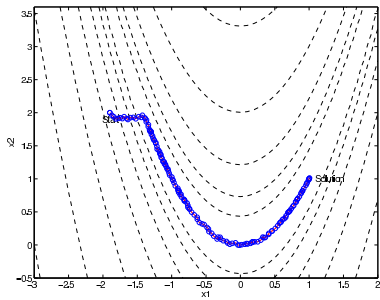
$$\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|} = 0$$

BFGS - Verfahren für Rosenbrock - Funktion



Grafik: Alt, Walter, Nichtlineare Optimierung

Mutations-Selektions - Verfahren für Rosenbrock-Funktion



Grafik: Alt, Walter, Nichtlineare Optimierung

- 192 -

Kosten der Linearen Algebra vs. Auswertungskomplexität

$$OPS(H_k^{-1}g_k) = \frac{1}{3}n^3 \gg OPS(B_k^{-1}g_k) \sim n^2$$

$$MEM(H_k) = n^2 \gg MEM(B_k) = kn \text{ for LM-BFGS}$$

LM \equiv Limited Memory Version

$$\frac{OPS(\nabla f(x))}{OPS(f(x))} \leq 4 \text{ via Algorithmischem Differenzieren}$$

$$\frac{OPS(\nabla^2 f(x))}{OPS(f(x))} \approx 4n \text{ im schlimmsten vollbesetzten Fall}$$

- 193 -

Zwischenfolgerungen (für den unrestringierten Fall)

- ▶ Gradienten kosten nur ein kleines Vielfaches der zu Grunde liegenden Funktionsauswertung vorausgesetzt diese ist durch einen Auswertungscode gegeben.
- ▶ Hesse- und Jacobimatrizen, e.g. $\nabla^2 f$ und im beschränkten Falle $\nabla h, \nabla c$, können sehr teuer zu faktorisieren sein, falls sie keine geeignete Dünnbesetztheitsstruktur besitzen.
- ▶ Gradientenbasierte quasi-Newton Methoden sind ein guter Kompromiss zwischen langsamen ableitungsfreien Verfahren und teuren Methoden zweiter Ordnung wie z.B. Newton.
- ▶ Bei unrestringierten Problemen kann durch Strahlsuche Konvergenz zu einem stationären Punkt erzwungen werden.
- ▶ Globale Optimierung ist extrem teuer und/oder sehr unzuverlässig.

- 194 -