Vorlesungsnotizen

Optimierung

Wintersemester 24/25

Emma Bach

Vorlesung gehalten von
Prof. Rolf Backofen
und
Prof. Moritz Diehl

March 13, 2025

Inhalt

1 Einführung **2**
1.1 Anwendungsbeispiel: Support Vector Machines ... 2
1.2 Typische Problemklassen ... 2
1.3 Konvexität ... 3

2 Gradientenverfahren **4**
2.1 Wiederholung: Mehrdimensionale Differentiation ... 4
2.2 Iterative Optimierung ... 5
2.2.1 Welche Richtung ist optimal? ... 5
2.3 Bestimmung der Schrittweite ... 5
2.3.1 Die Armijo-Bedingung ... 6
2.3.2 Die Wolfe-Bedingungen ... 6
2.3.3 Line Search ... 7
2.3.4 Conjugate Gradient-Verfahren ... 8

3 Newton und Quasi-Newton-Verfahren **9**
3.1 Das Newton-Verfahren ... 9
3.2 Konvergenzordnungen ... 9
3.2.1 Konvergenzradius des Newton-Verfahren ... 10
3.3 Quasi-Newton Verfahren ... 10
3.3.1 Das Davidon-Fletcher-Powell Verfahren ... 11
3.3.2 Das Broyden-Fletcher-Goldfarb-Shanno Verfahren ... 11
3.4 Das Gauss-Newton Verfahren ... 11

Chapter 1

Einführung

Ein Optimierungsproblem besteht aus einer **zulässigen Menge** G und einer **Zielfunktion** $f : G \to H$, wobei wir im Allgemeinen von $H = \mathbb{R}$ ausgehen werden. Wir schreiben dann zum Beispiel

$$\min_{x \in \mathbb{R}} f(x),$$

um das Problem "finde den kleinsten Wert, den $f(x)$ bei reellem x annimt" zu notieren, und

$$\operatorname*{argmin}_{x \in \mathbb{R}} f(x) = 4$$

für das Problem "finde die kleinste reelle Zahl x, sodass $f(x) = 4$ ist.

Historisch ist das Feld der mathematischen Optimierung unter Nebenbedingungen stark verankert im Feld der **Operations Research**, welches sich mit der Optimierung von Produktionskosten unter gegebenen Bedingungen beschäftigt. Heutzutage hat die Optimierung zahlreiche Anwendungen, z.B. in der Pfadplanung, in Computer Vision, in der Bioinformatik, im maschinellen Lernen oder im Hardwaredesign.

1.1 Anwendungsbeispiel: Support Vector Machines

Ein klassisches Problem des maschinellen Lernes ist die Klassifizierung von Daten durch eine lineare Entscheidungsgrenze - alle Punkte (x_i, y_i) über einer Ebene werden einer Klasse zugeordnet, und alle Punkte unter der Ebene einer anderen Klasse. Das Problem besteht daraus, eine Optimale Trennungsebene zu finden. Diese wird beschrieben durch eine Gleichung der Form

$$\hat{y}(x) = w^T x + w_0.$$

Sei w^* der Vektor $(w_0, w_1, \ldots, w_{n-1})$. Es stellt sich heraus, dass das zu lösende Optimierungsproblem gegeben ist durch:

$$\begin{aligned} \operatorname*{argmin}_{w^* \in \mathbb{R}^n} \quad & \tfrac{1}{2}||w^*||^2 \\ \text{s.t.} \quad & \forall i \; y_i \hat{y}(x_i) \geq 1 \end{aligned}$$

Wir wollen die Länge $||w^*||$ des Vektors w^* minimieren, da dies zu einem größeren Abstand zwischen unserer Ebene und den Datenpunkten führt, wodurch unser Modell besser generalisiert. Die Nebenbedingung entspricht der Anforderung, dass alle Punkte korrekt klassifiziert werden sollen.

1.2 Typische Problemklassen

Optimierungsprobleme können gemäß diverser Kriterien klassifiziert werden:

- Probleme mit Nebenbedingungen vs Probleme ohne Nebenbedingungen

- Optimierung mit Variablen aus verschiedenen Mengen, insbesondere kontinuierliche Variablen vs diskrete Variablen
- Lineare vs nichtlineare Funktionen
- Eindimensionale vs mehrdimensionale Funktionen
- Konvexe Funktionen vs nicht konvexe Funktionen
- Konvexe Mengen vs nicht konvexe Mengen

Diese verschiedenen Problemklassen führen zu unterschiedlich schwierigen Problemen. Insbesondere sind konvexe Probleme einfacher zu lösen als nicht konvexe Probleme, kontinuierliche Probleme sind in der Regel einfacher zu lösen als diskrete Probleme, und lineare Probleme sind einfacher als nichtlineare Probleme. Relevante Fragen sind dann:

- Wie schnell kovergiert das Verfahren zu einer Lösung? Wie viele Iterationen sind nötig? Was ist die Komplexität (in O-Notation) einer einzelnen Iteration?
- Konvergiert das Verfahren immer gegen ein Globales Optimum? Falls nein, gibt es garantierte obere/untere Schranken für die maximale Abweichung vom globalen Optimum?

1.3 Konvexität

Eine Menge G ist **konvex**, wenn für beliebige Punkte $x, y \in G$ auch beliebige lineare Interpolationen zwischen den Punkten in der Menge enthalten sind:

$$x, y \in G \implies \{(1-\alpha)x + \alpha y \mid \alpha \in [0,1]\} \subset G$$

Intuitiv entspricht das der Forderung, dass zwischen für alle Paare von Punkten eine Verbindungsstrecke zwischen den Punkten in der Menge enthalten sein muss.

Die **konvexe Hülle** einer Menge G ist die kleinste konvexe Menge H sodass $G \subset H$.

Analog zur Definition einer konvexen Menge ist eine **konvexe Funktion** definiert als eine Funktion, für die gilt:

$$x, y \in G \implies f((1-\alpha)x + \alpha y) \leq (1-\alpha)f(x) + \alpha f(y)\ \forall \alpha \in [0,1]$$

Dies entspricht der Forderung, dass jede Verbindungsstrecke zwischen zwei Punkten auf dem Graphen unterhalb des Graphen liegen muss.

Eine nicht konvexe Funktion, in der jedes lokale Minimum auch ein globales Minimum ist, wird als **quasikonvex** bezeichnet. Da dies dem Hauptvorteil von konvexen Funktionen in der Optimierung entspricht, ist die Optimierung von quasikonvexe Funktionen ebenfalls einfacher als die Optimierung allgemeiner nichtlinearer Funktionen.

Chapter 2

Gradientenverfahren

2.1 Wiederholung: Mehrdimensionale Differentiation

Zur Erinnerung: der **Gradient** ∇f ist ein Vektor, der alle partiellen Ableitungen von f enthält, also die Ableitung von f in alle Koordinatenrichtungen:

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right)^T$$

Im eindimensionalen Fall entspricht das genau der gewöhnlichen Ableitung, da partielle Ableitungen letztendlich genau wie ordinäre Ableitungen berechnet werden. Sei $f_1(x_1, x_2, x_3)$ eine beliebige Funktion $\mathbb{R}^3 \to \mathbb{R}$ und sei $f_2(x_1) = f_2(x_1, x_2, x_3)$ für beliebige Konstante x_2, x_3. Dann ist

$$\frac{\partial}{\partial x_1} f_1(x_1, x_2, x_3) = \frac{d}{dx_1} f_2(x_1)$$

Zum Beispiel:

$$\begin{aligned} f_1(x_1, x_2, x_3) &= x_1^2 x_2 + x_1 x_3 + 2x_2 \\ \frac{\partial}{\partial x_1} f_1(x_1, x_2, x_3) &= 2x_1 x_2 + x_3 \end{aligned}$$

entspricht der Ordinären Ableitung

$$\begin{aligned} f_2(x_1) &= x_1^2 x_2 + x_1 x_3 + 2x_2 \\ \frac{d}{dx_1} f_2(x_1) &= 2x_1 x_2 + x_3 \end{aligned}$$

Eine mehrdimensionale Funktion $f : \mathbb{R}^n \to \mathbb{R}$ kann nicht nur in die Koordinatenrichtungen abgeleitet werden, sondern in beliebige Richtungen $\boldsymbol{d} \in \mathbb{R}^n$ Um Formeln verständlicher zu notieren, werde ich so gut es geht Variablen $\boldsymbol{x} \in \mathbb{R}^n$ im Fall $n > 1$ immer durch fette Buchstaben notieren.. Diese **Richtungsableitung** wird geschrieben als $\nabla_d f(x)$. Analog zur ordinären Definition der Ableitung ist die Richtungsableitung definiert als:

$$\nabla_d f(x) = \lim_{h \to 0} \frac{f(x + h\boldsymbol{d}) - f(x)}{h}$$

f wird als differenzierbar bezeichnet, falls die Richtungsableitung in allen Richtungen existiert. Für ein differenzierbares f kann die Richtungsableitung mit Hilfe des Gradienten sehr einfach als Skalarprodukt berechnet werden:

$$\nabla_d f(x) = \nabla f(x)^T \boldsymbol{d}$$

Im Rahmen dieser Vorlesung gehen wir davon aus, dass $f \in C^1$, also dass die zu minimierende Funktion mindestens einmal differenzierbar ist. Im Allgemeinen ist es jedoch wichtig, anzumerken, dass f nicht umbedingt differenzierbar ist, nur weil partielle Ableitungen in alle Richtungen existieren.

2.2 Iterative Optimierung

Eine Funktion $f(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ kann numerisch durch iteratives "Ausprobieren" verschiedener Werte von $\boldsymbol{x}$ optimiert werden. Wir starten mit einem beliebigen Startwert $\boldsymbol{x}^{(0)}$ [1] , und bewegen uns dann bei jedem Schritt k in eine Richtung $\boldsymbol{d}^{(k)} \in \mathbb{R}^n$, welche uns hoffentlich näher zum Optimum bringt. Wir nutzen zusätzlich einen Parameter $\tau^{(k)} \in \mathbb{R}$, welcher die Schrittweite beschreibt.

$$\begin{aligned} \boldsymbol{x}^{(0)} &\in \mathbb{R}^n \\ \boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)} \end{aligned}$$

In der Praxis wird oft einfach $\boldsymbol{x}^{(0)} = \mathbf{0}$ gewählt. Ist jedoch f nicht quasikonvex, so kann eine schlechte Wahl von $\boldsymbol{x}^{(0)}$ dazu führen, dass das Verfahren nur gegen ein lokales Minimum kovergiert, welches oft kein globales Minimum ist und somit in vielen Fällen ein schlechtes Ergebnis.

2.2.1 Welche Richtung ist optimal?

Das Optimale $\boldsymbol{x}^{(k+1)}$ ist das, das $f(\boldsymbol{x}^{(k+1)}$ minimiert. Wir können $f(\boldsymbol{x}^{(k+1)})$ für kleine Schrittweiten $\tau^{(k)}$ folgendermaßen approximieren (**Taylor-Approximation**):

$$f(\boldsymbol{x}^{(k+1)}) = f(\boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)}) \approx f(\boldsymbol{x}^{k}) + \tau^{(k)} \nabla f(\boldsymbol{x}^{(k)}) \boldsymbol{d}^{(k)}$$

Da wir den Wert von f möglichst stark verringern wollen, sollte $\tau^{(k)} \nabla f(\boldsymbol{x}^{(k)}) \boldsymbol{d}^{(k)}$ ein negativer Term mit möglichst großem Betrag sein. Dabei können wir $\tau^{(k)}$ aber nicht zu hoch wählen, da sonst die Approximation ungenau wird.

Nach der Definition des Skalarprodukts gilt:

$$\nabla f(\boldsymbol{x}^{(k)}) \boldsymbol{d}^{(k)} = ||\nabla f(\boldsymbol{x}^{(k)})||\ ||\boldsymbol{d}^{(k)}|| \cos(\theta) \tag{2.1}$$

Wobei θ der Winkel zwischen $\nabla f(\boldsymbol{x}^{(k)})$ und $\boldsymbol{d}^{(k)}$ ist. Diese Gleichung ist minimal, wenn $\cos(\theta) = -1$, also $\theta = 180$. Dementsprechend muss $\boldsymbol{d}^{(k)}$ in die umgekehrte Richtung von $\nabla f(\boldsymbol{x}^{(k)})$ zeigen, also $\boldsymbol{d}^{(k)} = -\nabla f(\boldsymbol{x}^{(k)})$. [2]

Mit dieser optimalen Wahl der Abstiegsrichtung $\boldsymbol{d}^{(k)}$ erhalten wir das **Gradientenverfahren** (auch bekannt als "Gradientenabstieg", auf Englisch "**Gradient Descent**"):

$$\begin{aligned} \boldsymbol{x}^{(0)} &\in \mathbb{R}^n \\ \boldsymbol{x}^{(k+1)} &= \boldsymbol{x}^{(k)} - \tau^{(k)} \nabla f(\boldsymbol{x}^{(k)}) \end{aligned}$$

2.3 Bestimmung der Schrittweite

Die Wahl der Schrittweite $\tau^{(k)}$ ist für das Gradientenverfahren von entscheidender Bedeutung. Ist $\tau^{(k)}$ zu klein, wird die Konvergenz des Algorithmus oft deutlich verlangsamt. Ist $\tau^{(k)}$ zu groß, wird die Approximation ungenau. Dies kann dazu führen, dass das Verfahren über das Ziel hinausschießt, und im schlimmsten Fall eventuell gar nicht konvergiert.

Die Bestimmung eines optimalen Werts für $\tau^{(k)} \in \mathbb{R}$ ist ein eigenes Optimierungsproblem. In einigen Fällen ist ein dauerhaft konstantes τ genug, um die Konvergenz zu garantieren. In diesen Fällen ist eine konstante Schrittweite oft effizienter.

[1] Ich schreibe hier $\boldsymbol{x}^{(k)}$ mit der k in Klammern, um klarzustellen, dass $\boldsymbol{x}^{(k)}$ nicht "$\boldsymbol{x}$ hoch k" ist, sondern "das k-te $\boldsymbol{x}$". Eigentlich würde ich das am liebsten als $\boldsymbol{x}_k$ notieren, aber da $\boldsymbol{x} \in \mathbb{R}^n$ ist, ist diese Notation viel zu leicht mit der Notation x_k für die verschiedenen Komponenten von $\boldsymbol{x}$ zu verwechseln ($\boldsymbol{x} = (x_1, \ldots, x_n)^T$). Vermutlich werde ich an einigen Stellen aus Versehen trotzdem $\boldsymbol{x}^k$ schreiben, ich entschuldige mich im Voraus. (´-w-‘)

[2] Technisch gesehen wird die Gleichung durch $\boldsymbol{d}^{(k)} = -\alpha \cdot \nabla f(\boldsymbol{x}^{(k)})$ mit möglichst großen $\alpha \in \mathbb{R}$ minimiert. Die "Rolle" dieses Parameters α wir jedoch bereits durch unseren Schrittweiten-Parameter $\tau^{(k)}$ abgedeckt, und es gilt auch für dieses theoretische α, dass die Approximation sehr ungenau wird, wenn wir ein hohes α wählen.

Um ein optimales $\tau^{(k)}$ für eine gegebene Iteration k zu finden, definieren wir eine neue Funktion $h : \mathbb{R} \to \mathbb{R}$:

$$h(\tau) = f(\boldsymbol{x}^{(k)} + \tau \boldsymbol{d}^{(k)})$$

$$\tau^{(k)} = \underset{\tau \in \mathbb{R}}{\operatorname{argmin}}\ h(\tau)$$

Um die Ableitung von h zu finden, beschreiben wir den Term $\boldsymbol{x}^{(k)} + \tau \boldsymbol{d}^{(k)}$ als eine eigene eindimensionale Funktion $g(\tau)$. Dann ist $h(\tau) = f(\boldsymbol{g(\tau)})$, gemäß der mehrdimensionalen Kettenregel gilt also:

$$\begin{aligned}
\frac{d}{d\tau} h(\tau) &= \frac{d}{d\tau} f(\boldsymbol{g(\tau)}) \\
&= \sum_{i=1}^{n} \left(\frac{d}{d\tau} \boldsymbol{g(\tau)} \right) \frac{\partial}{\partial g(\tau)_i} f(\boldsymbol{g(\tau)}) \\
&= \sum_{i=1}^{n} d_i^{(k)} \frac{\partial}{\partial g(\tau)_i} f(\boldsymbol{g(\tau)}) \\
&= \nabla f(\boldsymbol{g(\tau)}) \cdot \boldsymbol{d}^{(k)} \\
&= \nabla f(\boldsymbol{x}^{(k)} + \tau \boldsymbol{d}^{(k)}) \boldsymbol{d}^{(k)}
\end{aligned}$$

Eine exakte Bestimmung des Optimalwerts ist mit hohem Aufwand Verbunden und ist oft sowieso nur von geringem Vorteil, da in der Regel auch bei wiederholter optimaler Wahl von $\tau^{(k)}$ für den letztendlichen Gradientenabstieg viele Schritte nötig sind. Stattdessen werden in der Praxis approximative Lösungen gefunden, welche bestimmte **Qualitätsbedingungen** erfüllen.

Eine einfache Suche nach einem beliebigen Wert, welcher den Funktionswert reduziert, also

$$f(\boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)}) - f(\tau^{(k)}) \geq 0,$$

ist dabei nicht genug, da bei hinreichend schneller Konvergenz von $f(\boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)}) - f(\tau^{(k)})$ gegen 0 eventuell nie das Optimum erreicht wird.

2.3.1 Die Armijo-Bedingung

Die **Armijo-Bedingung** hat das Ziel, sicherzustellen, dass die Zielfunktion bei einem Schritt um die gegebenen Schrittweite $\tau^{(k)}$ hinreichend reduziert wird. Die Bedingung ist gegeben durch die folgende Ungleichung:

$$f(\boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)}) \leq f(\boldsymbol{x}^{(k)}) + \delta \tau^{(k)} \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)}$$

Die minimal notwendige Reduktion wird dabei durch einen Parameter $\delta \in (0, 1)$ gesteuert. Ein üblicher Wert ist dabei $\delta \approx 10^{-4}$. Die Multiplikation mit dem Gradienten bedeutet, dass bei einem steileren Gradienten auch nach einer höheren Reduktion gesucht wird.

2.3.2 Die Wolfe-Bedingungen

Die **schwache Wolfe-Bedingung** fordert, dass der Gradient an der Zielposition entweder weniger steil als an der Startposition ist oder das Vorzeichen ändert:

$$-\nabla f(\boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)})^T \boldsymbol{d}^{(k)} \leq -\eta \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)}$$

Das Minus auf beiden Seiten kommt davon, dass bei einer Richtung $\boldsymbol{d}^{(k)}$, welche zu einem Abstieg führt, notwendigerweise $\nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)}$ negativ ist.

Wir wählen hier $\eta \in (\delta, 1)$. In der Regel wird η deutlich größer als δ gewählt, Wikipedia zitiert einen Richtwert von $\eta \approx 0.9$.

Bei der **starken Wolfe-Bedingung** wird der Betrag der Terme genommen, also gefordert, dass auch bei Vorzeichenwechsel die Steigung nach dem Schritt weniger steil als davor sein soll:

$$|\nabla f(\boldsymbol{x}^{(k)} + \tau^{(k)} \boldsymbol{d}^{(k)})^T \boldsymbol{d}^{(k)}| \leq \eta |\nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)}|$$

Intuitiv verhindern die Wolfe-Bedingungen ineffizient kleine Schritte, bei denen man ohne Probleme in die gleiche Richtung weitergehen könnte.

Schrittweiten, welche sowohl die starke Wolfe-Bedingung als auch die Armijo-Bedingung [3] erfüllen, existieren unter den von uns gemachten Annahmen immer. Als Erinnerung: Wir haben folgende Annahmen gemacht:

- $f(x)$ ist kontinuierlich differenzierbar ($f(x) \in C^1$)
- $\boldsymbol{d}^{(k)}$ ist eine Abstiegsrichtung an der Stelle $\boldsymbol{x}^{(k)}$ (also $\nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)} < 0$)
- $0 \leq \delta \leq \eta \leq 1$

Für die schwache Wolfe-Bedingung muss zusätzlich gegeben sein, dass die Menge $\{f(\boldsymbol{x}^{(k)} + \tau \boldsymbol{d}^{(k)}) \mid \tau > 0\}$ von unten beschränkt ist.

Beweis, nach Nocedal und Wright [4] (In der Vorlesung nicht gegeben):

- $h(\tau) = f(\boldsymbol{x}^{(k)} + \tau \boldsymbol{d}^{(k)})$ ist nach Annahme für alle τ nach unten beschränkt.
- Da $\boldsymbol{d}^{(k)}$ eine Abstiegsrichtung und δ positiv ist, ist die Linie $l_{Arm}(\tau) := f(\boldsymbol{x}^{(k)}) + \delta \tau \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)}$ nach unten unbeschränkt.
- Es gilt außerdem $h(0) = l_{Arm}(0)$ und $l_{Arm}(0) = \delta h'(0) < h'(0)$, also muss für kleine τ in Abstiegsrichtung gelten, dass $h(\tau) < l_{Arm}(\tau)$.
- Somit muss $l_{Arm}(\tau)$ den Graphen von $h(\tau)$ für mindestens einen Wert $\tau > 0$ schneiden. Sei τ' der kleinste Wert, bei dem es das tut.
- Es folgt $\forall \tau < \tau' : h(\tau) < l_{Arm}(\tau)$, da sonst τ' nicht der minimale Schnittpunkt wäre. Also erfüllen alle $\tau \in (0, \tau')$ die Armijo-Bedingung.
- Nach dem Mittelwertsatz muss es nun ein $\tau'' \in (0, \tau')$ geben, sodass

$$f(\boldsymbol{x}^{(k)} + \tau' \boldsymbol{d}^{(k)}) - f(\boldsymbol{x}^{(k)}) = \tau' \nabla f(\boldsymbol{x}^{(k)} \tau'' \boldsymbol{d}^{(k)})^T \boldsymbol{d}^{(k)}$$

- Da τ' ein Schnittpunkt ist, gilt nun

$$\begin{aligned}
& f(\boldsymbol{x}^{(k)} + \tau' \boldsymbol{d}^{(k)}) = f(\boldsymbol{x}^{(k)}) + \tau' \delta \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)} \\
\implies & f(\boldsymbol{x}^{(k)} + \tau' \boldsymbol{d}^{(k)}) - f(\boldsymbol{x}^{(k)}) = \tau' \delta \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)} \\
\implies & \tau' \nabla f(\boldsymbol{x}^{(k)} \tau'' \boldsymbol{d}^{(k)})^T \boldsymbol{d}^{(k)} = \tau' \delta \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)} \\
\implies & \nabla f(\boldsymbol{x}^{(k)} \tau'' \boldsymbol{d}^{(k)})^T \boldsymbol{d}^{(k)} = \delta \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)} \\
\overset{\eta > \delta}{\implies} & \nabla f(\boldsymbol{x}^{(k)} \tau'' \boldsymbol{d}^{(k)})^T \boldsymbol{d}^{(k)} > \eta \nabla f(\boldsymbol{x}^{(k)})^T \boldsymbol{d}^{(k)}
\end{aligned}$$

Also erfüllt τ'' sowohl die Armijo-Bedingung als auch die schwache Wolfe-Bedingung. Desweiteren ist der Term auf der linken Seite der finalen Ungleichung negativ, also gilt sogar die starke Wolfe-Bedingung. Da die Ungleichungen strikt sind existiert desweiteren ein Intervall um τ'', welches die beiden Bedingungen ebenfalls erfüllt.

2.3.3 Line Search

Ein einfaches Verfahren zur Bestimmung eines geeigneten τ ist die sogenannte **Backtracking Line Search**:

- Definiere $\tau^{(0)} = 1$

[3] In dieser Vorlesung wurden die starke Wolfe-Bedingung und die Armijo-Bedingung als seperate Bedingungen vorgestellt. In der Literatur wird die Armijo-Bedingung oft als Teil der starken Wolfe-Bedingung definiert, mit "Erfüllung der starken Wolfe-Bedingungen" ist dann die Erfüllung beider Bedingungen gemeint.

[4] J. Nocedal, S. J. Wright: Numerical Optimization, Springer, 2006, Seite 35ff

- Falls die Armijo-Bedingung nicht erfüllt ist, probiere als nächstes $\tau^{(k+1)} = \beta\tau^{(k)}$, wobei $\beta \in (0, 1)$.

Unter Nutzung des Gradienten lässt sich auch ein **Interpolationsverfahren** anwenden, welches schneller konvergiert:

- Definiere $\tau^{(0)} = 1$
- Betrachte die Quadratische Taylor-Approximation von $h(\tau)$:

$$h_q(\tau) = \left(\frac{h(\tau^{(0)}) - h(0) - \tau^{(0)}h'(0)}{{\tau^{(0)}}^2}\right)\tau^2 + h'(0)\tau + h(0)$$

  Diese ist minimiert durch

$$\tau^{(1)} = \frac{h'(0){\tau^{(0)}}^2}{2(h(\tau^{(0)}) - h(0) - h'(0)\tau^{(0)}}$$

- Solange $\tau^{(k)}$ die Armijo-Bedingung noch nicht erfüllt, bestimme $\tau^{(k+1)}$ durch "kubische Interpolation mit $h(0), h'(0), h(\tau^{(0)}, h(\tau^{(1)})$ [5]

Line Search Verfahren, welche die Wolfe-Bedingungen sicherstellen, sind komplizierter. Solche Verfahren bestehen aus zwei Komponenten:

- **Bracketing** erweitert das Suchintervall, bis darin geeignete Schrittweiten garantiert werden können.
- **Zooming** reduziert das Suchintervall, bis darin per Interpolationsverfahren eine geeignete Schrittweite gefunden wird.

2.3.4 Conjugate Gradient-Verfahren

Angenommen, unsere Zielfunktion ist eine quadratische Funktion in folgender Form:

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x} \tag{2.2}$$

Das sogenannte **Conjugate Gradient-Verfahren** funktioniert dann folgendermaßen:

- Wähle als erste Richtung den Gradienten an der Startposition:

$$\boldsymbol{d}^{(0)} = -\nabla f(\boldsymbol{x}^{(0)} = \boldsymbol{b} - A\boldsymbol{x}^0$$

- Bestimme die optimale Schrittweite:

$$\tau^{(k)} = \frac{|\nabla f(\boldsymbol{x}^{(k)})|}{{\boldsymbol{d}^{(k)}}^T A\boldsymbol{d}^{(k)}}$$

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \tau^{(k)}\boldsymbol{d}^{(k)}$$

- Wähle eine neue Richtung so, dass $\boldsymbol{d}^{(k+1)}$ und $\boldsymbol{d}^{(k)}$ gemäß der von A induzierten Metrik orthogonal sind, also ${\boldsymbol{d}^{(k+1)}}^T \cdot A \cdot \boldsymbol{d}^{(k)} = 0$. Dies wird garantiert durch:

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

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

Das Minimieren von f entspricht genau dem Lösen des Gleichungssystems $Ax = b$. Das CG-Verfahren garantiert dadurch für die Matrix $A = \mathbb{R}^{n\times n}$ eine bei exakter Zahlendarstellung exakte Lösung nach nur n Schritten. Wenn die **Konditionszahl** von A, definiert als der größte Eigenwert geteilt durch den kleinsten, klein ist, konvergiert es sogar wesentlich schneller.

Für nicht quadratische Funktionen kann das Conjugate-Gradient verfahren nahezu identisch angewandt werden, allerdings existiert im Allgemeinen für die Schrittweitenbedingung keine analytische Lösung , weshalb wieder Line Search nötig wird.

[5] Die Vorlesung war hier extrem vage - ich vermute, wir interpolieren iterativ immer $h(0), h'(0), h(\tau^{(k-1)}), h(\tau^{(k)})$? Wie genau man eine kubische Interpolation durchführt wurde aber nicht erklärt.

Chapter 3

Newton und Quasi-Newton-Verfahren

3.1 Das Newton-Verfahren

Das **Newton-Verfahren** ähnelt dem Gradientenverfahren, verwendet jedoch auch die Krümmung der Funktion, also die zweite Ableitung, gegeben durch die Hesse-Matrix H_f:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \tau^{(k)} H_f^{-1} \boldsymbol{x}^{(k)} \nabla f(\boldsymbol{x}^{(k)})$$

Dadurch ermöglicht das Newton-Verfahren größere Schritte in Richtungen mit schwacher Krümmung, bei denen die Approximation genauer und somit das "Risiko" geringer ist.

Zur Erinnerung: Die Hesse-Matrix H_f ist gegeben durch:

$$H_f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{pmatrix}$$

Die Hesse-Matrix ist symmetrisch, da $\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i}$.

Während das Gradientenverfahren also mit einer linearen Taylor-Approximation arbeitet, nutzt das Newton-Verfahren eine quadratische Taylor-Approximation:

$$\begin{aligned} \boldsymbol{m}^{(k)}(\boldsymbol{d}) &:= f(\boldsymbol{x}^{(k)} + \boldsymbol{d}) \approx f(\boldsymbol{x}^{(k)}) + \boldsymbol{d}^T \nabla f(\boldsymbol{x}^{(k)}) + \frac{1}{2} \boldsymbol{d}^T H_f \boldsymbol{x}^{(k)} \boldsymbol{d} \\ \nabla \boldsymbol{m}^{(k)}(\boldsymbol{d}) &= \nabla f(\boldsymbol{x}^{(k)}) + H_f(\boldsymbol{x}^{(k)}) \boldsymbol{d} \end{aligned}$$

Für das optimale d ist $\nabla \boldsymbol{m}^{(k)}(\boldsymbol{d}) = 0$ und wir erhalten:

$$\begin{aligned} H_f(\boldsymbol{x}^{(k)}) \boldsymbol{d} &= -\nabla f(\boldsymbol{x}^{(k)}) \\ \implies \boldsymbol{d} &= -H_f^{-1} \boldsymbol{x}^{(k)} \nabla f(\boldsymbol{x}^{(k)}) \end{aligned}$$

Die Optimale Schrittweite $\tau^{(k)}$ muss auch beim Newton-Verfahren durch Line Search bestimmt werden.

3.2 Konvergenzordnungen

Sei s eine beliebige Folge und $\overline{s} := \lim_{k \to \infty} s_k$. Wir betrachten nun die Konvergenzordnung der Folge s, also die Geschwindigkeit, mit der die Folge konvergiert. In dieser Vorlesung unterscheiden wir folgende Konvergenzordnungen:

- s konvergiert **linear**, wenn:

$$\lim_{k \to \infty} \frac{|s_{k+1} - \overline{s}|}{|s_k - \overline{s}|} < 1$$

ein Beispiel für eine linear konvergierende Folge ist $s_k = 0.9^k$. Das Gradientenverfahren konvergiert ebenfalls linear.

- s konvergiert **superlinear**, wenn:

$$\lim_{k\to\infty} \frac{|s_{k+1} - \overline{s}|}{|s_k - \overline{s}|} = 0$$

ein Beispiel für eine superlinear konvergierende Folge ist $s_k = \frac{1}{k!}$.

- s konvergiert **quadratisch**, wenn:

$$\lim_{k\to\infty} \frac{|s_{k+1} - \overline{s}|}{|s_k - \overline{s}|^2} < 1$$

ein Beispiel für eine superlinear konvergierende Folge ist $s_k = 0.9^{(2^k)}$.

Das Newton-Verfahren konvergiert für glatte, konvexe Funktionen nahe der Lösung quadratisch. Nominal ist es also wesentlich effizienter als Gradientenabstieg. Dabei ist jedoch zu berücksichtigen, dass die Berechung und Invertierung der Hesse-Matrix wesentlich aufwendiger ist als die Berechnung des Gradienten. Demenssprechend ist ein einzelner Iterationsschritt des Newton-Verfahrens auch wesentlich aufwendiger als ein einzelner Schritt des Gradientenverfahrens.

3.2.1 Konvergenzradius des Newton-Verfahren

Damit $\boldsymbol{d}^{(k)} = -H_f^{-1}\boldsymbol{x}^{(k)}\nabla f(\boldsymbol{x}^{(k)})$ eine Abstiegsrichtung ist, muss H_f positiv definit sein, also die Krümmung positiv. Bei negativer Krümmung läuft das Newton-Verfahren in die falsche Richtung! Dies ist kein Problem für konvexe Funktionen, da diese überall positive Krümmung haben. Bei nicht konvexen Funktionen muss die Hesse-Matrix erst manipuliert werden, damit alle ihre Eigenwerte positiv werden.

3.3 Quasi-Newton Verfahren

Die Idee hinter Quasi-Newton-Verfahren ist es, die Hesse-Matrix mithilfe der Gradienten sukzessiv zu approxmieren. Man ersetzt also in der Iterationsformel des Newton-Verfahrens die Matrix H_f durch eine Approximation B:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \tau^{(k)}(B^k)^{-1}\nabla f(\boldsymbol{x}^{(k)})$$

Die einfachste Möglichkeit ist eine lineare Taylor-Approximation des Gradienten. Sei $s = \boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}$. Dann haben wir:

$$\begin{aligned} &\nabla f(\boldsymbol{x}^{(k)} + \boldsymbol{s} \approx \nabla f(\boldsymbol{x}^{(k)}) + H_f(\boldsymbol{x}^{(k)})\boldsymbol{s} \\ \implies\ &H_f(\boldsymbol{x}^{(k)})(\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}) \approx \nabla f(\boldsymbol{x}^{(k+1)}) - \nabla f(\boldsymbol{x}^{(k)}) \end{aligned}$$

Wir suchen also in jedem Schritt eine Matrix $B^{(k)}$, sodass:

$$B^{(k)}(\boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)} = \nabla f(\boldsymbol{x}^{(k)}) - \nabla f(\boldsymbol{x}^{(k-1)})) \tag{3.1}$$

Abgekürzt schreiben wir auch:

$$B^{(k)}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)}$$

Aus $B^{(k)}\boldsymbol{s}^{(k)} = \boldsymbol{y}^{(k)}$ folgt ${\boldsymbol{s}^{(k)}}^T B^{(k)}\boldsymbol{s}^{(k)} = {\boldsymbol{s}^{(k)}}^T\boldsymbol{y}^{(k)}$. Wir gehen von einer konvexen Funktion f aus, also ist $B^{(k)}$ positiv definit. Daraus folgt ${\boldsymbol{s}^{(k)}}^T B^{(k)}\boldsymbol{s}^{(k)} > 0$, damit gilt auch ${\boldsymbol{s}^{(k)}}^T\boldsymbol{y}^{(k)} > 0$. Diese Ungleichung können wir als Bedingung für die Schrittweitensuche nutzen.

Um die Anzahl an möglichen $B^{(k)}$ einzuschränken, können wir als weitere Bedingung fordern, dass $B^{(k)}$ in jedem Schritt möglichst ähnlich zur letzten Iteration sein soll:

$$B^{(k)} = \min_B ||B - B^{(k-1)}||$$

$||\cdot||$ kann eine beliebige Matrixnorm sein. An diesem Punkt führen verschiedene Normen zu verschiedenen Unterverfahren. Ein beliebtes Verfahren ist das **Davidon-Fletcher-Powell Verfahren**, welches eine sog. gewichtete Frobeniusnorm benutzt.

3.3.1 Das Davidon-Fletcher-Powell Verfahren

3.3.2 Das Broyden-Fletcher-Goldfarb-Shanno Verfahren

3.4 Das Gauss-Newton Verfahren

Yay Gauss :D