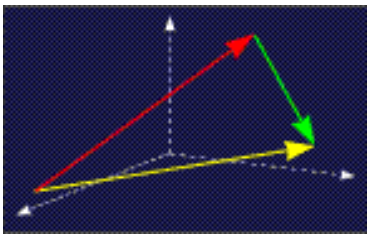
mathe online
Skripten
http://www.mathe-online.at/skripten/



Gradientenabstiegsverfahren

Franz Embacher

Fachhochschule Technikum Wien | Fakultät für Mathematik der Universität Wien
E-mail: embacher@technikum-wien.at
WWW: http://homepage.univie.ac.at/franz.embacher/

In diesem Skriptum werden die Grundideen der Gradientenabstiegsverfahren erster und zweiter Ordnung zur numerischen Suche nach lokalen Minimumstellen einer Funktion in mehreren Variablen skizziert. Als Voraussetzungen sind Grundkenntnisse der mehrdimensionalen Analysis (Gradient, Hesse-Matrix) und der linearen Algebra (Matrizenrechnung) sinnvoll.

1 Der Gradient

Die **Gradientenabstiegsverfahren**, kurz **Gradientenverfahren** (Mehrzahl, da es mehrere Varianten gibt, die auf einer gemeinsamen Grundidee beruhen) bilden eine Antwort auf die Frage, wie Extrema einer ein- oder zweimal stetig differenzierbaren Funktion in mehreren Variablen mit Hilfe numerischer Berechnungen gefunden werden können. Die Anwendungen reichen von der Suche nach Minima und Maxima von Funktionen in einigen wenigen Variablen bis zu Optimierungsproblemen von Funktionen mit sehr vielen (Millionen!) Variablen, wie sie im Bereich der künstlichen Intelligenz auftreten.

Wir wollen allgemein eine Funktion $f : \mathbb{R}^n \to \mathbb{R}$ betrachten. Die Minimalforderung besteht darin, dass die partiellen Ableitungen von f überall existieren und stetig sind. Wann immer zweite partielle Ableitungen von f eine Rolle spielen, wollen wir annehmen, dass auch diese überall existieren und stetig sind. (In realistischen Anwendungen können Funktionen auftreten, die nicht auf ganz $\mathbb{R}^n$ definiert oder nicht überall differenzierbar sind. Ob die hier beschriebenen Verfahren auch dann anwendbar sind, hängt vom Einzelfall ab – wir wollen uns die damit zusammenhängenden Komplikationen in diesem grundlegenden Text aber ersparen.)

Zu Notation: Wir werden die Variablen von f als $x_1, \ldots, x_n$ bezeichnen und die Symbole für Vektoren durch Fettdruck kennzeichnen, also allgemein $f(\mathbf{x})$ für einen Funktionswert schreiben.

Weiters schreiben wir Vektoren in Spaltenform an, also etwa

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \tag{1.1}$$

mit Ausnahme der Angabe eines Funktionswerts in der Form $f(x_1, \ldots, x_n)$, um uns zu ersparen, die Variablen in diesem Fall untereinander zu schreiben[1]. Um im Text einen Spaltenvektor anzugeben, kann man ihn als Zeilenvektor schreiben und das Zeichen ^{T} für das Transponieren hinzufügen, beispielsweise $\mathbf{x} = (2, 3)^T$, aber wenn klar ist, was gemeint ist, lässt man das ^{T} auch gerne weg.

Das zentrale mathematische Objekt bei den Gradientenabstiegsverfahren ist, wie schon der Name sagt, der **Gradient** von f. Er ist ein Vektorfeld, d.h. er besteht aus n Komponentenfunktionen, wobei die k-te Komponentenfunktion die partielle Ableitung von f nach der Variable x_k ist. Die zwei hauptsächlich verwendeten Bezeichnungen dafür sind $\mathbf{grad} f$ (diese Bezeichnung werden wir im Folgenden verwenden) und $\boldsymbol{\nabla} f$ (mit dem „Nabla"-Symbol $\boldsymbol{\nabla}$). Der Fettdruck drückt aus, dass der Gradient eine vektorielle Größe ist:

$$\mathbf{grad} f = \begin{pmatrix} \dfrac{\partial f}{\partial x_1} \\ \vdots \\ \dfrac{\partial f}{\partial x_n} \end{pmatrix}. \tag{1.2}$$

Um den Gradienten von f an einer bestimmten Stelle $\mathbf{x} \in \mathbb{R}^n$ anzugeben, schreiben wir

$$(\mathbf{grad} f)(\mathbf{x}) = \begin{pmatrix} \dfrac{\partial f}{\partial x_1}(\mathbf{x}) \\ \vdots \\ \dfrac{\partial f}{\partial x_n}(\mathbf{x}) \end{pmatrix}. \tag{1.3}$$

Die Klammer um das Symbol $\mathbf{grad} f$, die Sie hier sehen, machen wir vorsichtshalber, da es sich bei $\mathbf{x}$ meist um eine festgehaltene Stelle handelt. Ist beispielsweise $\mathbf{x} = (2, 3)$, und schreibt man $\mathbf{grad} f(2, 3)$, so könnte das Missverständnis entstehen, dass damit der Gradient von $f(2, 3)$ gemeint ist – aber da $f(2, 3)$ konstant ist (es handelt sich ja nur um eine Zahl), ist der Gradient von $f(2, 3)$ gleich $\mathbf{0}$. Die Schreibweise $(\mathbf{grad} f)(2, 3)$ bringt hingegen zum Ausdruck, dass der von den Variablen x_1, x_2 abhängige Gradient von f für $x_1 = 2$ und $x_2 = 3$ auszuwerten ist.

[1] Ein Element von $\mathbb{R}^n$ ist zwar weder eine „Zeile" noch eine „Spalte", sondern eine Liste von n reellen Zahlen, unabhängig davon, wie man sie aufschreibt, aber wenn man Matrizenrechnung betreibt – und das werden wir später in diesem Text, wenngleich in moderater Form –, muss man gewisse Zeilen-Spalten-Regeln einhalten.

Ohne genauere Begründung diskutieren wir nun kurz die wichtigsten Eigenschaften des Gradienten: Ist $\mathbf{n} \in \mathbb{R}^n$ ein Einheitsvektor, so gibt das Skalarprodukt von $\mathbf{n}$ mit dem Gradienten von f an einer Stelle $\mathbf{x}$,

$$\mathbf{n} \cdot (\mathbf{grad} f)(\mathbf{x}), \quad \text{manchmal auch geschrieben als} \quad \langle \mathbf{n}, (\mathbf{grad} f)(\mathbf{x}) \rangle \,, \tag{1.4}$$

die so genannte **Richtungsableitung**, an, wie stark sich die Funktionswerte von f ändern, wenn man ausgehend von $\mathbf{x}$ ein kleines Stück in Richtung $\mathbf{n}$ geht[2]. Zeigt $\mathbf{n}$ in Richtung einer der Koordinatenachsen, so reduziert sich die Richtungsableitung auf die entsprechende partielle Ableitung. Ist θ der Winkel, den $\mathbf{n}$ und $(\mathbf{grad} f)(\mathbf{x})$ einschließen, so ist

$$\mathbf{n} \cdot (\mathbf{grad} f)(\mathbf{x}) = \underbrace{\|\mathbf{n}\|}_{1} \, \|(\mathbf{grad} f)(\mathbf{x})\| \, \cos(\theta) = \|(\mathbf{grad} f)(\mathbf{x})\| \, \cos(\theta). \tag{1.5}$$

Wird θ variiert, so ergibt sich, sofern der Gradient $\neq \mathbf{0}$ ist,

(i) dass die Richtungsableitung am größten ist, wenn $\mathbf{n}$ in Richtung des Gradienten zeigt (dann ist $\theta = 0$, also $\cos(\theta) = 1$),

(ii) und am kleinsten, wenn $\mathbf{n}$ in die Gegenrichtung zeigt (dann ist $\theta = \pi$, also $\cos(\theta) = -1$).

(iii) Steht $\mathbf{n}$ normal auf den Gradienten, so ist die Richtungsableitung gleich 0 (dann ist $\theta = \frac{\pi}{2}$, also $\cos(\theta) = 0$).

Aus (i) folgt, dass der Gradient in jene Richtung zeigt, in der die lokale Änderungsrate von f maximal ist. Aus (ii) folgt, dass die Gegenrichtung des Gradienten jene Richtung ist, in der die lokale Änderungsrate von f minimal ist. Um eine nützliche Folgerung von (iii) zu erschließen, benötigen wir einen weiteren Begriff: Für $c \in \mathbb{R}$ ist die zu c gehörende **Niveaumenge** von f definiert durch

$$N_c = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) = c\}. \tag{1.6}$$

Hier einige Beispiele:

- Ist $n = 2$ und $f(x_1, x_2) = 3x_1 + 5x_2$, so ist N_c für jedes c eine Gerade. Alle diese Geraden sind zueinander parallel.
- Ist $n = 2$ und $f(x_1, x_2) = x_1^2 + x_2^2$, so ist N_c für negatives c die leere Menge, für positives c eine Kreislinie (die wir dann **Niveaulinie** nennen) und für $c = 0$ ein Punkt.
- Ist $n = 2$ und $f(x_1, x_2) = x_1 x_2$, so ist N_c für $c \neq 0$ eine Hyperbel (mit zwei Ästen) und für $c = 0$ die Vereinigungsmenge der beiden Koordinatenachsen, also zwei einander schneidende Geraden.
- Ist $n = 3$ und $f(x_1, x_2, x_3) = x_1^2 + x_2^2 + x_3^2$, so ist N_c für negatives c die leere Menge, für positives c eine Sphäre (Kugeloberfläche, wir nennen sie dann **Niveaufläche**) und für $c = 0$ ein Punkt.

Diese Beispiele illustrieren, dass die meisten nichtleeren Niveaumengen „glatte“ Gebilde (also Linien oder Flächen ohne Knicke oder Kanten) sind. In $\mathbb{R}^n$ ergibt sich dasselbe Bild: Von Ausnahmen abgesehen ist N_c ein glattes Gebilde der Dimension $n - 1$. Nun die Folgerung aus (iii): Liegt $\mathbf{x}$ auf einer solchen glatten Niveaumenge, so ist $(\mathbf{grad} f)(\mathbf{x})$ ein Normalvektor auf N_c.

[2] Genauer ausgedrückt: Die Richtungsableitung (1.4) gibt die lokale Änderungsrate der Funktion $t \mapsto f(\mathbf{x} + t\,\mathbf{n})$ an der Stelle $t = 0$ an.

Begründung: Geht man von $\mathbf{x}$ aus entlang einer Kurve, die in N_c liegt, dann ändert sich der Funktionswert nicht (da ja N_c so definiert ist – der Funktionswert ist stets gleich c). Die Richtungsableitung ist aber gerade dann gleich 0, wenn die Richtung $\mathbf{n}$ normal zu $(\mathbf{grad}f)(\mathbf{x})$ ist.

Die Ausnahmen (in den obigen Beispielen, wenn N_c nur ein Punkt ist oder wenn N_c aus zwei einander schneidenden Geraden besteht und man sich an deren Schnittpunkt befindet) treten nur an Stellen $\mathbf{x}$ auf, an denen der Gradient gleich $\mathbf{0}$ ist. Abgesehen von diesen Ausnahmen gilt: Der Gradient von f an der Stelle $\mathbf{x}$ steht stets normal auf die durch $\mathbf{x}$ verlaufende Niveaumenge.

2 Gradientenabstiegsverfahren erster Ordnung

Wir wollen nun ein lokales Minimum unserer Funktion $f : \mathbb{R}^n \to \mathbb{R}$ durch numerische Berechnungen (was in der Praxis bedeutet: durch Berechnungen am Computer) finden. (Die Minimumsuche tritt bei Optimierungsproblemen häufiger auf als die Maximumsuche. Sucht man ein Maximum, so betrachtet man einfach $-f$ anstelle von f.) Wie kann uns das Konzept des Gradienten dabei helfen? Ist $\boldsymbol{\xi} \in \mathbb{R}^n$ eine lokale Minumumstelle von f, so sind alle partiellen Ableitungen an dieser Stelle gleich 0.

Begründung für $n = 2$: Ist $\boldsymbol{\xi} = (\xi_1, \xi_2)^T$ eine lokale Minumumstelle von f, so hat die Funktion $x_1 \mapsto f(x_1, \xi_2)$ bei $x_1 = \xi_1$ ein lokales Minimum, und die Funktion $x_2 \mapsto f(\xi_1, x_2)$ hat bei $x_2 = \xi_2$ ein lokales Minumum. Die Ableitungen an diesen Stellen (die bekanntermaßen gleich 0 sein müssen), sind aber gerade die partiellen Ableitungen von f an der Stelle $\boldsymbol{\xi}$.

Daher muss

$$(\mathbf{grad}f)(\boldsymbol{\xi}) = \mathbf{0} \tag{2.1}$$

gelten. Das alleine ist noch keine Garantie, dass $\boldsymbol{\xi}$ eine lokale Minimumstelle ist. Es könnte sich auch um eine lokale Maximumstelle oder um eine Sattelstelle handeln. Allgemein nennen wir eine Stelle $\boldsymbol{\xi}$, an der (2.1) gilt, eine **kritische Stelle** (oder einen **kritischen Punkt**[3]). Wie soll man also eine kritische Stelle finden, an der f (zumindest lokal, d.h. verglichen mit Funktionswerten an Stellen in einer kleinen Umgebung von $\boldsymbol{\xi}$) minimal ist?

Nun nehmen wir an, f besitze (zumindest) eine lokale Minimumstelle und beginnen mit irgendeiner **Anfangsstelle** (einem **Startpunkt**[4]) $\mathbf{x}^{(0)} \in \mathbb{R}^n$. In der Regel wird der Gradient an dieser Stelle $\neq \mathbf{0}$ sein, denn soviel Glück wollen wir nicht unterstellen, dass zufällig eine kritische Stelle getroffen wird. In welche Richtung sollte man von $\mathbf{x}^{(0)}$ aus gehen, um eine realistische Chance zu haben, einer lokalen Minimumstelle näher zu kommen? Wir wissen, dass $(\mathbf{grad}f)(\mathbf{x}^{(0)})$ in jene Richtung zeigt, in der f am stärksten anwächst. Da wir keine lokale Maximumstelle von f suchen, sondern eine lokale Minimumstelle, drehen wir uns um: Gehen

[3] Im Zusammenhang mit Funktionen $\mathbb{R} \to \mathbb{R}$ macht man oft bewusst einen Unterschied zwischen „Stellen“ und „Punkten“, indem $x \in \mathbb{R}$ als Stelle und $(x, y) \in \mathbb{R}^2$, beispielsweise $(x, f(x))$, als Punkt bezeichnet wird. Daher werden die Elemente von $\mathbb{R}^n$ in ihrer Rolle als Argumente von f in diesem Skriptum ebenfalls „Stellen“ genannt. Ausnahme: In den später diskutierten Beispielen werden Stellen im $\mathbb{R}^2$ als kleine Kreise visualisiert, die wir als „Punkte“ bezeichnen.

[4] Siehe Fußnote 3.

wir ein hinreichend kleines Stück in Richtung $-(\mathbf{grad}f)(\mathbf{x}^{(0)})$ zu einer Stelle $\mathbf{x}^{(1)}$, so wird der Funktionswert dort kleiner sein, d.h. es wird $f(\mathbf{x}^{(1)}) < f(\mathbf{x}^{(0)})$ gelten. Und was machen wir dort? Dasselbe! Also: $(\mathbf{grad}f)(\mathbf{x}^{(1)})$ berechnen und ein kleines Stück in die Gegenrichung zu einer Stelle $\mathbf{x}^{(2)}$ gehen. Und dort? Wieder dasselbe! Und so macht man weiter, gemäß dem Schema

$$\mathbf{x}^{(0)} \longrightarrow \mathbf{x}^{(1)} \longrightarrow \mathbf{x}^{(2)} \longrightarrow \mathbf{x}^{(3)} \rightarrow \dots \tag{2.2}$$

und verbunden mit der Hoffnung, auf diese Weise einer lokalen Mimimumstelle immer näher (im Idealfall beliebig nahe) zu kommen. Garantie dafür gibt es keine. Es könnte beispielsweise passieren, dass man einen zu großen Sprung macht und weit weg von der gesuchten Minimumstelle landet. Andererseits sollten die Schritte nicht zu klein sein, damit sich in einer akzeptablen Zeit eine annehmbare Approximation für eine lokale Minumumstelle ergibt. (Dieser Gesichtspunkt ist insbesondere dann relevant, wenn n sehr groß und jeder Schritt mit einem erheblichen Rechenaufwand verbunden ist). Die naheliegendste Möglichkeit, die bisherigen Überlegungen zu realisieren, ist ein Algorithmus der Form

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \varepsilon\,(\mathbf{grad}f)(\mathbf{x}^{(k)}) \qquad \text{für } k = 0, 1, 2, 3, \dots \tag{2.3}$$

mit einem frei wählbaren Parameter $\varepsilon > 0$, der sogenannten **Lernrate**, die die Schrittweite bestimmt. Das ist die **Methode des steilsten Abstiegs** (**mit konstanter Lernrate**). Sie ist die einfachste Version eines **Gradienten**(**abstiegs**)**verfahrens erster Ordnung**, wobei der Zusatz „erster Ordnung“ ausdrückt, dass man nur Informationen verwendet, die uns die ersten partiellen Ableitungen von f (die ja gemeinsam den Gradienten bilden) liefern. Die Bezeichung „Gradientenabstiegsverfahren“ erklärt sich aus der Idee dieses Algorithmus, zu immer kleineren Werten von f „abzusteigen“. Die **Abstiegsrichtung** ist immer die Gegenrichtung des Gradienten.

Um in der Praxis eine geeignete Lernrate ε zu wählen, muss man entweder mehrere Werte von ε ausprobieren oder ein bisschen mehr über die Funktion f wissen. In vielen Fällen kann der Gefahr, die gesuchte Minimumstelle zu überspringen, begegnet werden, indem man ε genügend klein wählt, denn zumindest intuitiv ist klar, dass der Betrag des Gradienten umso kleiner ist, je näher man einer kritischen Stelle gekommen ist, und umso kleiner ist dann auch die Schrittweite in (2.3).

Andererseits – was eine „genügend kleine Lernrate“ ist, muss nicht an jeder Stelle gleich sein. Insbesondere dann, wenn nicht ausgeschlossen werden kann, dass die Funktion f in verschiedenen Bereichen des $\mathbb{R}^n$ ganz unterschiedliches Verhalten zeigt, indem sie zum Beispiel entlang des Weges, den der Algorithmus einschlägt, zuerst sanft abfällt, aber ab einem gewissen Punkt schnell zu oszillieren beginnt, besteht eine verbesserte Methode darin, die Lernrate in jedem Schritt neu zu adaptieren. Das entspricht dem Schema

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \varepsilon^{(k)}\,(\mathbf{grad}f)(\mathbf{x}^{(k)}) \qquad \text{für } k = 0, 1, 2, 3, \dots, \tag{2.4}$$

wobei $\varepsilon^{(k)}$ beispielsweise, ausgehend von einem eher großen Startwert, so lange verringert (z.B. halbiert) wird, bis $f(\mathbf{x}^{(k+1)}) < f(\mathbf{x}^{(k)})$ gilt. Damit ist sichergestellt, dass man tatsächlich „absteigt“. Eine andere Variante (die sogenannte **Liniensuche**) besteht darin, $\varepsilon^{(k)}$ als die kleinste positive lokale Minimumstelle der Funktion

$$\varepsilon \;\mapsto\; f\big(\mathbf{x}^{(k)} - \varepsilon\,(\mathbf{grad}f)(\mathbf{x}^{(k)})\big) \tag{2.5}$$

zu wählen (was bedeutet, in jedem Schritt ein Optimierungsproblem für eine Funktion in einer Variable zu lösen). Techniken dieser Art werden unter dem Begriff **Schrittweitensteuerung** zusammengefasst und stellen ausgefeiltere (wenngleich berechnungsintensivere) Varianten der Methode des steilsten Abstiegs dar.

Es stellt sich nun noch die Frage, ob die bisher gewählte Abstiegsrichtung (die Gegenrichtung des Gradienten) immer die günstigste ist. Dabei ist zu beachten, dass der Gradient von f an einer gegebenen Stelle keine Information darüber enthält, wie sich die Funktion in der Nähe einer anderen Stelle verhält. Der Gradient muss *nicht* direkt in Richtung einer lokalen Maximumstelle zeigen, und seine Gegenrichtung muss *nicht* genau auf eine lokale Minimumstelle zielen.

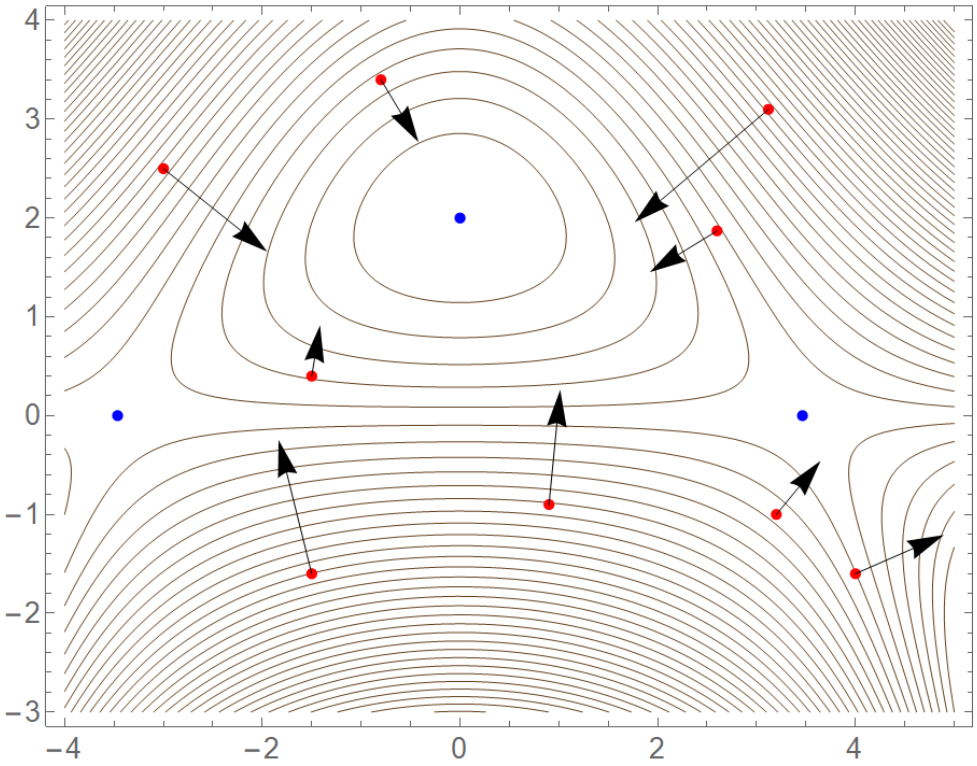


Abbildung 1: Sie sehen hier Niveaulinien der Funktion (2.6) und an einigen ausgewählten (roten) Stellen jeweils -0.07 mal dem Gradienten, als Pfeil dargestellt. Diese Grafik kann durchaus wie eine Wanderkarte mit Höhenschichtenlinien gelesen werden. Der Funktionswert an einer Stelle wäre dann etwa die Höhe über dem Meeresspiegel. Der obere blaue Punkt kennzeichnet die lokale Minimumstelle von f (auf einer Wanderkarte entspricht er dem tiefsten Punkt eines Tales), die beiden anderen blauen Punkte kennzeichnen die Sattelstellen von f. Die Pfeile (sie entsprechen auf einer Wanderkarte jeweils der steilsten Bergab-Richtung) stehen überall normal auf die Niveaulinien. Die meisten Pfeile weisen ungefähr in Richtung der Minimumstelle, aber nur einer (der oberste) einigermaßen genau, und manche weichen stark von dieser Richtung ab. Tritt einer der roten Punkte als ein $\mathbf{x}^{(k)}$ im Algorithmus (2.3) auf, so liegt $\mathbf{x}^{(k+1)}$ bei einer Lernrate von 0.07 genau bei der Spitze des entsprechenden Pfeils. Die Pfeile sind umso länger, je dichter die Niveaulinien liegen (d.h. je steiler es bergab geht). Ist man einmal mit dem Algorithmus (2.3) in die Nähe der Minimumstelle gelangt, so wird die Schrittweite automatisch kleiner, je näher man ihr kommt, weil die Pfeile immer kürzer werden.

Sehen wir uns das anhand eines Beispiels an: Wir wählen die Funktion in zwei Variablen

$$f(x_1, x_2) = x_1^2 x_2 + 3(x_2 - 2)^2. \tag{2.6}$$

Sie besitzt (bitte überprüfen Sie das selbst!) eine lokale Minimumstelle bei $(0, 2)$ und zwei Sattelstellen bei $(\pm 2\sqrt{3}, 0)$. Abbildung 1 zeigt diese drei kritischen Stellen, einige Niveaulinien von f und an einigen ausgewählten Stellen -0.07 mal dem Gradienten, als Pfeil dargestellt. Die Richtung jedes Pfeil wird durch den Verlauf der Niveaulinie, auf die er (wie in Abschnitt 1 besprochen) normal steht, festgelegt, und ist daher in der Regel *nicht* die Richtung direkt zur Minimumstelle.

Das Beispiel zeigt, dass die Gegenrichtung des Gradienten nicht unbedingt die günstigste Abstiegsrichtung ist. Das gilt auch in höheren Dimensionen: Wie in Abschnitt 1 besprochen, steht der Gradient, sofern er $\neq \mathbf{0}$ ist, normal auf die Niveaumenge durch den Punkt, an dem er gebildet wird. Seine Gegenrichtung weist nur in Ausnahmefällen direkt auf eine Minimumstelle von f hin. Daher ist es manchmal sinnvoll, eine andere Abstiegsrichtung zu wählen. Ist etwa von einer Funktion $f : \mathbb{R}^n \to \mathbb{R}$ bekannt, dass sie in den verschiedenen Koordinatenrichtungen unterschiedlich schnell variiert, so kann man an die Stelle von ε in (2.3) eine positiv definite $n \times n$-Diagonalmatrix D setzen, also den Algorithmus (2.3) in der Form

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - D\,(\mathbf{grad} f)(\mathbf{x}^{(k)}) \qquad \text{für } k = 0, 1, 2, 3, \ldots \tag{2.7}$$

verallgemeinern. Damit wird gewissermaßen für jede Koordinatenrichtung eine andere Lernrate (die jetzt dem jeweiligen Diagonalemenent von D entspricht) vorgegeben, und dadurch ändert sich auch die Abstiegsrichtung. Diese Methode wird **diagonal skalierter steilster Abstieg** genannt. Es sind auch andere Varianten denkbar, z.B. eine Anpassung von D in jedem Schritt (dann müsste in (2.7) D durch $D^{(k)}$ ersetzt werden), wobei die Minimalforderung an die Abstiegsrichtung in jedem Fall darin besteht, dass das Skalarprodukt des Verbindungsvektors von $\mathbf{x}^{(k)}$ zu $\mathbf{x}^{(k+1)}$ mit $(\mathbf{grad} f)(\mathbf{x}^{(k)})$ negativ ist. Damit ist zumindest sichergestellt, dass die Funktionswerte von f bei einem hinreichend kleinen Schritt kleiner werden.

Und wann hat der Algorithmus sein Ziel erreicht? Dass an einer der Stellen $\mathbf{x}^{(k)}$ der Gradient (im Rahmen der Rechengenauigkeit) gleich $\mathbf{0}$ ist, wäre schon eine sehr gute Nachricht. Anderenfalls besteht eine Möglichkeit darin, eine von der gewünschten Genauigkeit abhängige Schranke vorzugeben und abzubrechen, sobald die Norm des Gradienten diese unterschreitet. Dadurch hat man eine gewisse Kontrolle darüber, wie stark sich die Funktionswerte in einer hinreichend kleinen Umgebung der Abbruchstelle $\mathbf{x}^{(k)}$ noch von $f(\mathbf{x}^{(k)})$ unterscheiden können.

Ganz ist man aber damit noch nicht fertig, denn woher wissen wir, dass wir tatsächlich in (oder nahe) einer lokalen Minimumstelle gelandet sind? Statt dessen in einer lokalen Maximumstelle zu landen, ist eher unwahrscheinlich, insbesondere wenn eine geeignete Schrittweitensteuerung eingesetzt wird. Schließlich bestand ja die Idee darin, Schritt für Schritt zu immer kleineren Funktionswerten „abzusteigen“. Allerdings besteht die Möglichkeit, in der Nähe einer Sattelstelle zu landen. Für $n = 2$ kann man sich das gut vorstellen: Wenn Sie auf einem Gebirgspass zwischen zwei Bergen (einem „Sattel“) stehen und sich Berg A vor Ihnen und Berg B hinter Ihnen erhebt, so geht es links und rechts talwärts hinunter. Wer bei Nebel von Berg A zu Berg B spaziert, muss das nicht einmal bemerken, sondern glaubt, eine lokale Minimumstelle

der Landschaft durchschritten zu haben. So etwas kann einem Gradientenabstiegsverfahren durchaus auch passieren. In welcher Art von kritischer Stelle man gelandet ist, kann oft mit Hilfe der (im nächsten Abschnitt zu besprechenden) Hesse-Matrix von f entschieden werden.

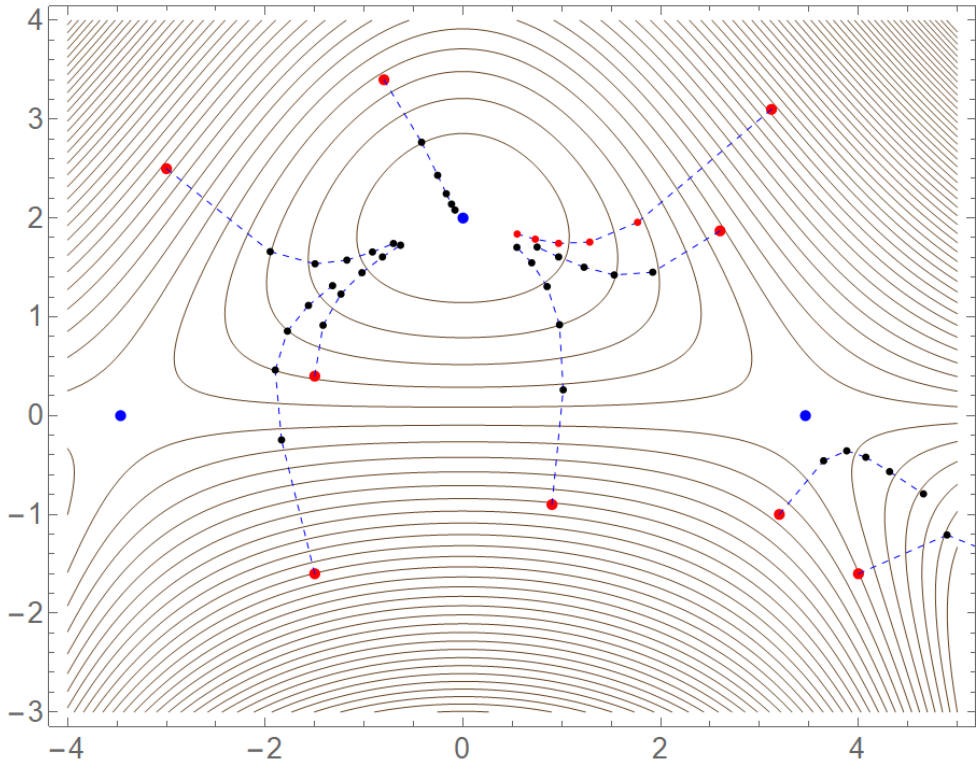


Abbildung 2: Die Grafik zeigt die Ergebnisse von neun Durchgängen der Methode des steilsten Abstiegs mit konstanter Lernrate $\varepsilon = 0.07$ und fünf Inerationsschritten, angewandt auf die Funktion (2.6). Als Anfangsstelle $\mathbf{x}^{(0)}$ wurde jeweils einer der neun roten Punkte von Abbildung 1 gewählt. In schwarz (in einem Fall rot) und etwas kleiner dargestellt sind jeweils die fünf Stellen $\mathbf{x}^{(1)}$ bis $\mathbf{x}^{(5)}$, die der Algorithmus berechnet, verbunden durch gerade strichlierte Linien, damit klar ersichtlich ist, welche dieser Stellen zu welchem Abstiegspfad gehören. Sieben der neun Durchläufe liefern ein befriedigendes Ergebnis – die berechneten Stellen streben der lokalen Minimumstelle zu, wenngeich nicht auf kürzestem Weg. In zwei Fällen führt der Algorithmus nach rechts unten weiter talwärts aus dem Bildausschnitt hinaus – hier liegen die Anfangsstellen offenbar zu nahe an einer Sattelstelle. Die beste Approximation nach fünf Schritten ergibt sich für jenen Abstiegspfad, der am obersten der roten Punkte beginnt. Die sechs anderen Pfade, die der Minimumstelle zustreben, kommen ungefähr gleich weit, obwohl ihre Anfangsstellen durchaus in unterschiedlicher Entfernung vom Ziel liegen. (Das liegt daran, dass der Gradient von f an diesen Anfangsstellen einen vergleichsweise großen Betrag und offenbar eine günstige Richtung hat – bereits der erste Schritt fällt groß aus und geht ungefähr in die richtige Richtung.)

Wir wollen am Ende dieses Abschnitts noch ein konkretes Beispiel einer Funktion $f : \mathbb{R}^2 \to \mathbb{R}$ betrachten. Da wir den $\mathbb{R}^2$ als Zeichenebene gut kennen und Vorgänge im $\mathbb{R}^2$ gut grafisch darstellen können, ist der Fall $n = 2$ generell günstig, um die entwickelten Konzepte und Verfahren gedanklich noch einmal zu durchdringen. Für Funktionen in zwei Variablen, die durch einen geschlossenen Term angegeben sind, sollte es auch keine Schwierigkeiten bereiten,

selbst damit ein bisschen zu experimentieren. Wir wählen die Funktion (2.6), deren Niveaulinien und kritische Stellen wir bereits kennen, und wenden die Methode des steilsten Abstiegs mit konstanter Lernrate an. Der Gradient von f ist gegeben durch

$$(\mathbf{grad} f)(\mathbf{x}) = \begin{pmatrix} 2x_1x_2 \\ x_1^2 + 6x_2 - 12 \end{pmatrix}. \tag{2.8}$$

Um uns einen Überblick über das Arbeiten des Algorithmus zu verschaffen, führen wir das Verfahren neun mal aus, wobei jedesmal einer der roten Punkte in Abbildung 1 als Anfangsstelle gewählt wird. Der Lernrate geben wir zunächst den Wert $\varepsilon = 0.07$. (Bei diesem Wert ist der Schritt von $\mathbf{x}^{(0)}$ zu $\mathbf{x}^{(1)}$ jeweils genau der Schritt vom Schaft bis zur Spitze des entsprechenden Pfeils in Abbildung 1.) In jedem Fall führen wir fünf Iterationsschritte von (2.3) durch. Die Ergebnisse dieser fünf Durchgänge sind in Abbildung 2 grafisch dargestellt. Von zwei Ausnahmen abgesehen (die sozusagen von der Nähe einer Sattelstelle „abgelenkt“ werden) streben die Abstiegspfade wie erwartet der lokalen Minimumstelle zu. Einer (und zwar jener, der am obersten roten Punkt beginnt) kommt ihr recht nahe, weil schon seine Anfangsstelle nicht weit von ihr entfernt war und auch die Richtung von Beginn an gut gepasst hat.

k	$x_1^{(k)}$	$x_2^{(k)}$	$\left\|\mathbf{x}^{(k)} - \boldsymbol{\xi}\right\|$	$\left\|(\mathbf{grad} f)(\mathbf{x}^{(k)})\right\|$
0	3.12	3.1	3.30823	25.318
1	1.76592	1.95659	1.76645	7.47807
2	1.28219	1.75653	1.30511	4.50815
3	0.966884	1.74371	1.00028	3.4254
4	0.73085	1.78591	0.761562	2.71618
5	0.548117	1.83844	0.571433	2.12348
6	0.407042	1.88526	0.422904	1.62134
7	0.299609	1.92185	0.309632	1.2124
8	0.218996	1.94839	0.224995	0.892602
9	0.159259	1.96671	0.162702	0.650251
10	0.115409	1.97892	0.117319	0.470583
⋮	⋮	⋮	⋮	⋮
20	0.00436679	1.99986	0.00436889	0.017484
⋮	⋮	⋮	⋮	⋮
50	$2.29174 \cdot 10^{-7}$	1.9999999999875	$2.29174 \cdot 10^{-7}$	$9.16694 \cdot 10^{-7}$

Tabelle 1: Ergebnisse der Methode des steilsten Abstiegs (2.3) mit Lernrate $\varepsilon = 0.07$ für die Funktion (2.6). Die Anfangsstelle ist der äußerste rote Punkt rechts oben in Abbildung 1. Die erste Spalte gibt die Nummer des Iterationsschritts an, die nächsten beiden die Koordinaten der Approximationsstelle, die vierte den Abstand der Approximationsstelle vom Ziel und die letzte den Betrag des Gradienten an der Approximationsstelle.

Um uns die Genauigkeit der Approximation und den Einfluss der Lernrate genauer anzusehen, wählen wir als Beispiel den äußersten roten Punkt rechts oben in Abbildung 1. Die ersten

fünf Approximationsstellen, die sich bei einer Lernrate von 0.07 ergeben, sind in der Abbildung ebenfalls rot dargestellt. Setzen wir den Algorithmus (2.3) über die ersten fünf Schritte hinaus weiter fort, so ergeben sich die in Tabelle 1 angegebenen Werte. Sie zeigen, dass sich die $\mathbf{x}^{(k)}$ der lokalen Minimumstelle $\boldsymbol{\xi} = (0, 2)$ eher gemächlich annähern.

Zum Vergleich ändern wir nun die Lernrate. Mit $\varepsilon = 0.2$ sind die Schritte wesentlich größer als zuvor, und auch die Annäherung an die lokale Minimumstelle erfolgt wesentlich schneller (Tabelle 2).

k	$x_1^{(k)}$	$x_2^{(k)}$	$\left\|\mathbf{x}^{(k)} - \boldsymbol{\xi}\right\|$	$\left\|(\mathbf{grad} f)(\mathbf{x}^{(k)})\right\|$
0	3.12	3.1	3.30823	25.318
1	-0.7488	-0.16688	2.29261	12.4431
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
10	$-9.47133 \cdot 10^{-7}$	2.00000242775	$2.60596 \cdot 10^{-6}$	0.0000150511
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
15	$-3.03081 \cdot 10^{-10}$	1.999999999223	$8.33908 \cdot 10^{-10}$	$4.81636 \cdot 10^{-9}$

Tabelle 2: Ergebnisse der Methode des steilsten Abstiegs (2.3) mit Lernrate $\varepsilon = 0.2$ für die Funktion (2.6). Die Anfangsstelle ist der äußerste rote Punkt rechts oben in Abbildung 1.

Bei einer weiteren Vergrößerung der Lernrate auf $\varepsilon = 0.3$ passiert jedoch ein Malheur, wie die Zahlen in Tabelle 3 zeigen.

k	$x_1^{(k)}$	$x_2^{(k)}$	$\left\|\mathbf{x}^{(k)} - \boldsymbol{\xi}\right\|$	$\left\|(\mathbf{grad} f)(\mathbf{x}^{(k)})\right\|$
0	3.12	3.1	3.30823	25.318
1	-2.6832	-1.80032	4.6521	18.3514
2	-5.58157	2.88039	5.65058	48.5952
3	4.06468	-8.05049	10.8413	78.7394
4	23.6983	5.0839	23.8981	628.165
5	-48.5896	-168.95	177.721	16472.6

Tabelle 3: Ergebnisse der Methode des steilsten Abstiegs (2.3) mit Lernrate $\varepsilon = 0.3$ für die Funktion (2.6). Die Anfangsstelle ist der äußerste rote Punkt rechts oben in Abbildung 1.

Offenbar springen wir aufgrund der großen Lernrate über das Ziel hinaus, werden nach zwei Schritten stets an eine Stelle katapultiert, an der der Betrag des Gradienten größer ist als an der vorigen und entfernen uns in Folge immer weiter weg vom Ziel. In einer grafischen Darstellung ergibt sich in solchen Fällen meist eine Zickzack-Bewegung mit immer größeren Sprüngen. Wir lernen daraus, dass die Lernrate zwar nicht zu klein sein soll (denn dann dauert es sehr lange, bis man eine vernünftige Approximation erzielt hat), aber sie darf auch nicht zu groß sein (denn dann wird man bei jedem Schritt zu weit hinaus geschossen und gelangt nie ans Ziel).

3 Die Hesse-Matrix

Als Vorbereitung für die Verfahren zweiter Ordnung besprechen wit kurz ein weiteres wichtiges Konzept der mehrdmensionalen Analysis: die **Hesse-Matrix** unserer Funktion f (die ab jetzt zweimal stetig differenzierbar sei). Sie ist die $n \times n$-Matrix der zweiten partiellen Ableitungen von f, definiert als

$$H_f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}. \tag{3.1}$$

Sie ist symmetrisch (da es aufgrund des Satzes von Schwarz nicht auf die Reihenfolge der partiellen Ableitungen ankommt) und besteht, ebenso wie der Gradient von f, aus Funktionen. Für die Hesse-Matrix von f an einer Stelle $\mathbf{x} \in \mathbb{R}^n$ schreiben wir $H_f(\mathbf{x})$.

Mit Hilfe des Gradienten und der Hesse-Matrix können wir die quadratische Approximation von f in der Nähe einer Stelle $\mathbf{a} \in \mathbb{R}^n$ in kompakter Form anschreiben:

$$f(\mathbf{x}) \approx f(\mathbf{a}) + (\mathbf{x}-\mathbf{a})^T (\mathbf{grad} f)(\mathbf{a}) + \frac{1}{2}(\mathbf{x}-\mathbf{a})^T H_f(\mathbf{a})\,(\mathbf{x}-\mathbf{a}). \tag{3.2}$$

Auf der rechten Seite steht das Taylorpolynom zweiter Ordnung von f mit Entwicklungsstelle $\mathbf{a}$. Wir werden auf diese Beziehung im nächsten Abschnitt zurückkommen.

Die Hesse-Matrix stellt in gewisser Weise eine „Krümmungsinformation“ dar[5]. Sie spielt eine wichtige Rolle bei der Klassifikation kritischer Stellen. Als reelle symmetrische Matrix ist sie diagonalisierbar, d.h. sie besitzt n Eigenwerte (die aber nicht alle verschieden sein müssen). Ist $\boldsymbol{\xi} \in \mathbb{R}^n$ eine kritische Stelle einer zweimal stetig differenzierbaren Funktion $f : \mathbb{R}^n \to \mathbb{R}$, so gilt[6]:

- Ist die Hesse-Matrix $H_f(\boldsymbol{\xi})$ **positiv definit** (d.h. sind alle ihre Eigenwerte positiv), so ist $\boldsymbol{\xi}$ eine lokale Minumumstelle von f.
- Ist die Hesse-Matrix $H_f(\boldsymbol{\xi})$ **negativ definit** (d.h. sind alle ihre Eigenwerte negativ), so ist $\boldsymbol{\xi}$ eine lokale Maximumstelle von f.
- Ist die Hesse-Matrix $H_f(\boldsymbol{\xi})$ **indefinit** (d.h. besitzt sie zumindest einen positiven und einen negativen Eigenwert), so ist $\boldsymbol{\xi}$ eine Sattelstelle von f.

[5] Im Fall $n = 2$ kann man sich das einigermaßen bildlich als Krümmung des Graphen von f, der ja als Fläche im $\mathbb{R}^3$ dargestellt werden kann, vorstellen.

[6] Die ersten beiden dieser Kriterien reduzieren sich für $n = 1$ auf die wohlbekannten Regeln „zweite Ableitung positiv $\Rightarrow$ Minimum“ und „zweite Ableitung negativ $\Rightarrow$ Maximum“.

- In allen anderen Fällen ist auf diese Weise keine eindeutige Entscheidung möglich. Kurz notiert:
  - Eigenwerte $(0, \text{positiv}) \Rightarrow$ lokale Minimumstelle oder Sattelstelle.
  - Eigenwerte $(0, \text{negativ}) \Rightarrow$ lokale Maximumstelle oder Sattelstelle.
  - Alle Eigenwerte sind $0 \Rightarrow$ lokale Minimumstelle oder lokale Maximumstelle oder Sattelstelle.

Wir erwähnen noch, dass eine reelle symmetrische $n \times n$-Matrix mit Koeffizienten a_{jk} genau dann positiv definit ist, wenn die Determinanten aller Matrizen, die man mit a_{jk} bilden kann, indem die Indizes auf $1 \leq j, k \leq m$ eingeschränkt werden (für $m = 1, 2, \ldots, n$) positiv sind. Für $n = 2$ reduziert sich das auf das folgende Kriterium: Eine reelle symmetrische 2×2-Matrix ist genau dann positiv definit, wenn eines ihrer Diagonalelemente und ihre Determinante positiv sind. (Das andere Diagonalelement ist dann automatisch positiv.)

4 Gradientenabstiegsverfahren zweiter Ordnung

Kehren wir zur Suche nach einer lokalen Minimumstelle unserer Funktion f zurück. Ein **Gradienten(abstiegs)verfahren zweiter Ordnung** benutzt, neben dem Gradienten von f, auch Informationen, die uns die zweiten partiellen Ableitungen von f (die ja gemeinsam die Hesse-Matrix bilden) liefern. Den direktesten Weg zu einem solchen Algorithmus liefert die Formel (3.2) für die quadratische Approximation unserer Funktion. Ist $\boldsymbol{\xi}$ eine kritische Stelle von f, so reduziert sich (3.2) mit $\mathbf{a} = \boldsymbol{\xi}$ auf

$$f(\mathbf{x}) \approx f(\boldsymbol{\xi}) + \frac{1}{2}(\mathbf{x} - \boldsymbol{\xi})^T H_f(\boldsymbol{\xi})\,(\mathbf{x} - \boldsymbol{\xi}), \tag{4.1}$$

da ja $(\mathbf{grad} f)(\boldsymbol{\xi}) = 0$ gilt. Auf der rechten Seite steht der Term einer quadratischen Funktion. Beschränken wir uns also zunächst auf die Untersuchung quadratischer Funktionen: Sei $g : \mathbb{R}^n \to \mathbb{R}$ definiert durch

$$g(\mathbf{x}) = g(\boldsymbol{\xi}) + \frac{1}{2}(\mathbf{x} - \boldsymbol{\xi})^T A\,(\mathbf{x} - \boldsymbol{\xi}) \qquad \text{für alle } \mathbf{x} \in \mathbb{R}^n, \tag{4.2}$$

wobei A eine (konstante) symmetrische $n \times n$-Matrix ist. Jede quadratische Funktion, für die $\boldsymbol{\xi}$ eine kritische Stelle ist, kann in dieser Form geschrieben werden. Wir wollen nun zusätzlich annehmen, dass A positiv definit ist. Dann ist $\boldsymbol{\xi}$ eine lokale Minimumstelle. Die Hesse-Matrix von g ist gleich A und damit konstant. Geben wir eine beliebige Anfangstelle $\mathbf{x}^{(0)}$ vor, so ist $H_g(\mathbf{x}^{(0)}) = A$. Für ein Gradientenabstiegsverfahren benötigen wir noch den Gradienten von g an der Stelle $\mathbf{x}^{(0)}$. Dazu berechnen wir zuerst allgemein den Gradienten von g durch direkte Differentiation von (4.2):

$$(\mathbf{grad}\, g)(\mathbf{x}) = A\,(\mathbf{x} - \boldsymbol{\xi}). \tag{4.3}$$

Begründung: Bezeichnen wir die Koeffizienten von A mit a_{jk}, so ist

$$(\mathbf{x} - \boldsymbol{\xi})^T A\,(\mathbf{x} - \boldsymbol{\xi}) = \sum_{j,k=1}^{n} (x_j - \xi_j) a_{jk} (x_k - \xi_k). \tag{4.4}$$

Die l-te Komponente des Gradienten ist daher gegeben durch

$$\frac{1}{2}\frac{\partial}{\partial x_l}\sum_{j,k=1}^{n}(x_j-\xi_j)a_{jk}(x_k-\xi_k)=\frac{1}{2}\sum_{j,k=1}^{n}\frac{\partial}{\partial x_l}\Big((x_j-\xi_j)a_{jk}(x_k-\xi_k)\Big). \quad (4.5)$$

Jetzt denken Sie selbst kurz darüber nach, warum das gleich

$$\sum_{k=1}^{n}a_{lk}(x_k-\xi_k) \quad (4.6)$$

ist, also gleich der l-ten Komponente von $A\,(\mathbf{x}-\boldsymbol{\xi})$.

In (4.3) setzen wir jetzt $\mathbf{x}=\mathbf{x}^{(0)}$ ein, erhalten

$$(\mathbf{grad}\,g)(\mathbf{x}^{(0)})=A\,(\mathbf{x}^{(0)}-\boldsymbol{\xi}), \quad (4.7)$$

multiplizieren von links mit der zu A inversen Matrix A^{-1} (sie existiert, weil A positiv definit ist), kommen also zur Beziehung

$$A^{-1}(\mathbf{grad}\,g)(\mathbf{x}^{(0)})=\mathbf{x}^{(0)}-\boldsymbol{\xi} \quad (4.8)$$

und damit schließlich zu

$$\boldsymbol{\xi}=\mathbf{x}^{(0)}-A^{-1}(\mathbf{grad}\,g)(\mathbf{x}^{(0)}). \quad (4.9)$$

Das bedeutet, dass wir die Minimumstelle $\boldsymbol{\xi}$ in einem einzigen Schritt finden, wenn der Gradient und die Hesse-Matrix von g an einer beliebigen Stelle $\mathbf{x}^{(0)}$ bekannt sind. Das muss man sich einmal auf der Zunge zergehen lassen: Egal, wo man startet, nach einem einzigen Schritt ist man am Ziel, nämlich an der gesuchten Minimumstelle! Das ist natürlich eine spezielle Eigenschaft der quadratischen Funktionen mit positiv definiter Hesse-Matrix, aber sie stmmt uns zuversichtlich.

Mit diesem schönen Ergebnis drängt sich ein Verfahren auf, das wir für unsere (im Allgemeinen nicht-quadratische) Funktion f nutzen können: Die Matrix A ersetzen wir in (4.9) durch die Hesse-Matrix von f an der Stelle $\mathbf{x}^{(0)}$ (eine andere haben wir ja zunächst nicht). Wir können zwar nicht erwarten, mit einem einzigen Schritt eine lokale Miminumstelle von f zu finden, aber wir können hoffen, nahe an eine solche heranzukommen. An die Stelle von $\boldsymbol{\xi}$ muss daher in (4.9) ein Näherungswert $\mathbf{x}^{(1)}$ treten, von dem aus wir rekursiv weitergehen zu einem (hoffentlich noch besseren) Näherungswert $\mathbf{x}^{(2)}$, und so weiter. Insgesamt lassen wir also den Algorithmus

$$\mathbf{x}^{(k+1)}=\mathbf{x}^{(k)}-H_f(\mathbf{x}^{(k)})^{-1}\,(\mathbf{grad}f)(\mathbf{x}^{(k)}) \qquad \text{für } k=0,1,2,3,\ldots \quad (4.10)$$

laufen. Er ist eng mit dem **Newton-Verfahren** zur numerischen Berechnung von Nullstellen verwandt und wird auch hier mit diesem Namen bezeichnet[7]. Je besser sich das Verhalten von

[7] Das Newton-Verfahren zum Auffinden einer Nullstelle der Funktion $h:\mathbb{R}\to\mathbb{R}$ verwendet den Algorithmus

$$x^{(k+1)}=x^{(k)}-\frac{h(x^{(k)})}{h'(x^{(k)})}. \quad (4.11)$$

Ersetzt man h durch f', so ergibt sich (4.10) für $n=1$.

f in der Nähe einer lokalen Minimumstelle durch eine quadratische Funktion beschreiben lässt, und je näher $\mathbf{x}^{(0)}$ dieser Minimumstelle ist, umso besser funktioniert es, und es konvergiert dann in der Regel wesentlich schneller als ein Gradientenabstiegsverfahren erster Ordnung, benötigt aber – vor allem durch das Berechnen und Invertieren der Hesse-Matrix in jedem Schritt – mehr Rechenzeit.

Und es hat noch einen weiteren Nachteil: Wir haben bei den Überlegungen zur quadratischen Funktion g angenommen, dass sie eine lokale Minimumstelle besitzt. Nun betrachten Sie einmal, wie der zweite Term auf der rechten Seite von (4.10) von f abhängt: Ersetzt man f durch $-f$ so ändert sich das Vorzeichen des Gradienten, aber auch das Vorzeichen der Hesse-Matrix – beide bestehen ja nur aus (ersten bzw. zweiten) partiellen Ableitungen von f. Das Produkt bleibt also gleich. Das bedeutet, dass der Algorithmus keinen Unterschied zwischen Minima und Maxima macht! Man kann ihn ebensogut bei der Suche nach einer lokalen Maximumstelle benutzen, was aber impliziert, dass er – auch wenn man f minimieren will – statt dessen in einer lokalen Maximumstelle landen kann! Bei den Gradientenabstiegsverfahren erster Ordnung war das nicht so: Wenn Sie in (2.3) f durch $-f$ ersetzen, ändert der zweite Term auf der rechten Seite sein Vorzeichen. So war ja der „Abstieg“ angelegt. Um mit (2.3) eine lokales Maximumstelle zu finden, müsste man ε negativ wählen. Beim Newton-Verfahren (4.10) hingegen spielt die Inverse der Hesse-Matrix die Rolle der Lernrate. Ist sie negativ-definit, so marschieren wir mitunter schnurstracks auf eine lokale Maximumstelle zu statt auf eine lokale Minimumstelle. Falls also f sowohl lokale Mimimumstellen als auch lokale Maximumstellen besitzt, ist es hier wichtiger als bei den Verfahren erster Ordnung, dass man an einer günstigen Anfangsstelle $\mathbf{x}^{(0)}$ beginnt, die sich nicht gerade in der Nähe einer solchen Maximumstelle befindet. Ebenso ist es gut möglich, dass der Algorithmus zu einer Sattelstelle führt, wenn f eine solche besitzt. Das zeigt sich natürlich auch für quadratische Funktionen, die ja alle vom Typ (4.2) sind: Mit Formel (4.9) bekommt man auch dann in einem einzigen Schritt die kritische Stelle, wenn A negativ definit oder indefinit ist.

Ein interessanter Aspekt des Newton-Verfahrens betrifft auch die Richtung, in die man sich von Schritt zu Schritt bewegt: Der Vektor $H_f(\mathbf{x}^{(k)})^{-1}\,(\mathbf{grad}f)(\mathbf{x}^{(k)})$ muss keineswegs parallel zu $(\mathbf{grad}f)(\mathbf{x}^{(k)})$ sein. Es wird also auch die „Abstiegsrichtung“, wenn es sich denn tatsächlich um einen Abstieg handelt, durch die Hesse-Matrix vorgegeben. Die Minimalforderung an die Abstiegsrichtung, die wir in Abschnitt 2 erwähnt haben, nämlich dass das Skalarprodukt des Verbindungsvektors von $\mathbf{x}^{(k)}$ zu $\mathbf{x}^{(k+1)}$ mit $(\mathbf{grad}f)(\mathbf{x}^{(k)})$ negativ sein soll, besagt nun

$$(\mathbf{grad}f)(\mathbf{x}^{(k)}) \cdot H_f(\mathbf{x}^{(k)})^{-1}\,(\mathbf{grad}f)(\mathbf{x}^{(k)}) \;>\; 0 \tag{4.12}$$

und ist zumindest dann, wenn $H_f(\mathbf{x}^{(k)})$ positiv definit (und der Gradient $\neq \mathbf{0}$) ist, erfüllt.

Wir betrachten als Beispiel wieder die Funktion (2.6), mit der wir in Abschnitt 2 ein Gradientenabstiegsverfahren erster Ordnung illustriert haben. Ihre lokale Minimumstelle ist, wie wir bereits wissen, $\boldsymbol{\xi} = (0, 2)$. Ihr Gradient wurde in (2.8) berechnet, ihre Hesse-Matrix ist gegeben durch

$$H_f(\mathbf{x}) = \begin{pmatrix} 2x_2 & 2x_1 \\ 2x_1 & 6 \end{pmatrix}, \tag{4.13}$$

ihre Inverse berechnet sich zu

$$H_f(\mathbf{x})^{-1} = \frac{1}{2(x_1^2 - 3x_2)} \begin{pmatrix} -3 & x_1 \\ x_1 & -x_2 \end{pmatrix}. \tag{4.14}$$

Um das Newton-Verfahren anzuwenden, brauchen wir nur noch eine Anfangsstelle. Wir wählen zunächst wieder den äußersten roten Punkt rechts oben in Abbildung 1, auf den sich die drei Tabellen in Abschnitt 2 bezogen haben. Die Ergebnisse sind in Tabelle 4 dargestellt.

k	$x_1^{(k)}$	$x_2^{(k)}$	$\left\|\mathbf{x}^{(k)} - \boldsymbol{\xi}\right\|$	$\left\|(\mathbf{grad} f)(\mathbf{x}^{(k)})\right\|$
0	3.12	3.1	3.30823	25.318
1	11.2561	−8.08398	15.1125	193.654
2	6.97967	−3.07129	8.6275	46.6107
3	4.76781	−0.97329	5.61894	10.4915
4	3.77045	−0.203599	4.36717	1.8294
5	3.48865	−0.0152167	4.02887	0.132584
6	3.46428	−0.000106319	4.00021	0.000946389
7	3.4641	$-5.45731 \cdot 10^{-9}$	4.0000000108	$4.92885 \cdot 10^{-8}$

Tabelle 4: Ergebnisse des Newton-Verfahrens (4.10) mit der Funktion (2.6), wenn als Anfangsstelle der äußerste rote Punkt rechts oben in Abbildung 1 gewählt wird.

Was ist jetzt passiert? Der Algorithmus konvergiert, aber er konvergiert nicht gegen die lokale Maximumstelle $(0, 2)$, sondern gegen die Sattelstelle $(2\sqrt{3}, 0) \approx (3.4641016, 0)$. Aber sagen Sie nicht, Sie wären nicht gewarnt worden – wir haben oben erwähnt, dass diese Möglichkeit beim Newton-Verfahren besteht! Tatsächlich landen wir in acht der neun Fälle, deren Anfangsstelle einer der roten Punkte von Abbildung 1 ist, in einer der beiden Sattelstellen. Lediglich für den obersten roten Punkt führt das Verfahren zur lokalen Minimumstelle, und zwar sagenhaft schnell, wie die Werte in Tabelle 5 zeigen. Eine grafische Darstellung des Abstiegspfads wie in Abbildung 2 hätte hier nicht viel Sinn, da bereits $\mathbf{x}^{(2)}$ kaum mehr von $\boldsymbol{\xi}$ zu unterscheiden wäre.

k	$x_1^{(k)}$	$x_2^{(k)}$	$\left\|\mathbf{x}^{(k)} - \boldsymbol{\xi}\right\|$	$\left\|(\mathbf{grad} f)(\mathbf{x}^{(k)})\right\|$
0	−0.8	3.4	1.61245	10.5506
1	−0.324686	2.02008	0.325307	1.3311
2	−0.000411144	2.01753	0.0175305	0.105167
3	$-3.57149 \cdot 10^{-6}$	2.00000002768	$3.57159 \cdot 10^{-6}$	0.0000142869
4	$-4.94322 \cdot 10^{-14}$	2.000000000002126	$2.12643 \cdot 10^{-12}$	$1.27567 \cdot 10^{-11}$

Tabelle 5: Ergebnisse des Newton-Verfahrens (4.10) mit der Funktion (2.6), wenn als Anfangsstelle der oberste rote Punkt in Abbildung 1 gewählt wird.

Dieses Beispiel illustriert, dass das Newton-Verfahren sehr schnell konvergiert, aber dass bei Vorhandensein von Sattelstellen (ebenso wie bei Vorhandensein von lokalen Maximumstellen) eine günstige Wahl der Anfangsstelle entscheidend dafür ist, dass es auch gegen den Punkt konvergiert, den man sucht.

Zuletzt wollen wir noch erwähnen, dass es auch andere Varianten des Gradientenabstiegsverfahrens zweiter Ordnung gibt, beispielsweise solche (die sogenannten **Quasi-Newton-Verfahren**), bei denen die Hesse-Matrix nicht in jedem Schritt neu berechnet und invertiert, sondern lediglich rekursiv angenähert wird, um den Rechenaufwand zu minimieren.

Dieses Skriptum wurde erstellt im November 2020 für den Master-Studiengang „Data Science“ an der Fachhochschule Technikum Wien (http://www.technikum-wien.at/).
Die Skripten-Seite von mathe online (http://www.mathe-online.at/) finden Sie unter http://www.mathe-online.at/skripten/.