2. Projekt

Quasi-Newton-Verfahren & Gauß-Newton-Verfahren

im Fach

Numerische Optimierung

Juni 2020

Maximilian Gaul

Aufgabe 1

Siehe Programmcode in `Project2.m`.

Aufgabe 2

Für A^0 wählt man im BFGS-Verfahren üblicherweise die Einheitsmatrix, d.h. für den Induktionsanfang gilt:

$$A^0 = I$$

Wenn man das nun in die Formel für die zwei Rang-1-Modifikationen und in die BFGS-Update-Formel einsetzt erhält man den Ansatz (für $B = A^{-1} = I$):

$$\left(I + \frac{ss^T - ys^T + ss^T - sy^T}{y^Ts} - \frac{s^Tyss^T - y^Tyss^T}{(y^T)s^2}\right) \cdot \left(I - \frac{ss^T}{s^Ts} + \frac{yy^T}{y^Ts}\right)$$

Der Übersicht halber teile ich die einzelnen Produkte auf und füge sie nach dem Kürzen am Ende wieder zusammen. Es ist zu beachten, dass $y^Ts = s^Ty$. Skalare Werte werden gekürzt.

$$I \cdot I = I$$

$$I \cdot \left(-\frac{ss^T}{s^Ts}\right) = -\frac{ss^T}{s^Ts}$$

$$I \cdot \left(\frac{yy^T}{y^Ts}\right) = \frac{yy^T}{y^Ts}$$

$$\left(\frac{ss^T - ys^T + ss^T - sy^T}{y^Ts}\right) \cdot I = \frac{ss^T}{y^Ts} - \frac{ys^T}{y^Ts} + \frac{ss^T}{y^Ts} - \frac{sy^T}{y^Ts}$$

$$\left(\frac{ss^T - ys^T + ss^T - sy^T}{y^Ts}\right) \cdot \left(-\frac{ss^T}{s^Ts}\right) = -\frac{ss^Tss^T - ys^Tss^T + ss^Tss^T - sy^Tss^T}{y^Tss^Ts}$$

$$= -\frac{s\color{red}{s^Ts}s^T}{y^Ts\color{red}{s^Ts}} + \frac{y\color{red}{s^Ts}s^T}{y^Ts\color{red}{s^Ts}} - \frac{s\color{red}{s^Ts}s^T}{y^Ts\color{red}{s^Ts}} + \frac{s\color{red}{y^Ts}s^T}{\color{red}{y^Ts}s^Ts} = -\frac{ss^T}{y^Ts} + \frac{ys^T}{y^Ts} - \frac{ss^T}{y^Ts} + \frac{ss^T}{s^Ts}$$

$$\left(\frac{ss^T - ys^T + ss^T - sy^T}{y^Ts}\right) \cdot \left(\frac{yy^T}{y^Ts}\right) = \frac{ss^Ty - ys^Tyy^T + ss^Tyy^T - sy^Tyy^T}{(y^Ts)^2}$$

$$= \frac{s\color{red}{s^Ty}y^T}{(\color{red}{y^Ts})^2} - \frac{y\color{red}{s^Ty}y^T}{(\color{red}{y^Ts})^2} + \frac{s\color{red}{s^Ty}y^T}{(\color{red}{y^Ts})^2} - \frac{sy^Tyy^T}{(y^Ts)^2} = \frac{sy^T}{y^Ts} - \frac{yy^T}{y^Ts} + \frac{sy^T}{y^Ts} - \frac{sy^Tyy^T}{(y^Ts)^2}$$

$$\left(-\frac{s^Tyss^T - y^Tyss^T}{(y^Ts)^2}\right) \cdot I = -\frac{s^Tyss^T}{(y^Ts)^2} + \frac{y^Tyss^T}{(y^Ts)^2} = -\frac{ss^T}{y^Ts} + \frac{y^Tyss^T}{(y^Ts)^2}$$

$$\left(-\frac{s^Tyss^T - y^Tyss^T}{(y^Ts)^2}\right) \cdot \left(-\frac{ss^T}{s^Ts}\right) = \frac{s^Tyss^Tss^T - y^Tyss^Tss^T}{(y^Ts)^2s^Ts}$$

$$= \frac{s^Tyss^Tss^T}{(y^Ts)^2s^Ts} - \frac{y^Tyss^Tss^T}{(y^Ts)^2s^Ts} = \frac{ss^T}{y^Ts} - \frac{y^Tyss^T}{(y^Ts)^2}$$

$$\left(-\frac{s^Tyss^T - y^Tyss^T}{(y^Ts)^2}\right) \cdot \left(\frac{yy^T}{y^Ts}\right) = -\frac{s^Tyss^Tyy^T + y^Tyss^Tyy^T}{(y^Ts)^3}$$

$$= -\frac{s^Tyss^Tyy^T}{(y^Ts)^3} + \frac{y^Tyss^Tyy^T}{(y^Ts)^3} = -\frac{sy^T}{y^Ts} + \frac{y^Tysy^T}{(y^Ts)^2}$$

Zusammenfassen:

$$I - \frac{ss^T}{s^Ts} + \frac{yy^T}{y^Ts} + \frac{ss^T}{y^Ts} - \frac{ys^T}{y^Ts} + \frac{ss^T}{y^Ts} - \frac{sy^T}{y^Ts} - \frac{ss^T}{y^Ts} + \frac{ys^T}{y^Ts} - \frac{ss^T}{y^Ts} + \frac{ss^T}{s^Ts} + \frac{sy^T}{y^Ts} - \frac{yy^T}{y^Ts} + \frac{sy^T}{y^Ts}$$

$$-\frac{sy^Tyy^T}{(y^Ts)^2} - \frac{ss^T}{y^Ts} + \frac{y^Tyss^T}{(y^Ts)^2} + \frac{ss^T}{y^Ts} - \frac{y^Tyss^T}{(y^Ts)^2} - \frac{sy^T}{y^Ts} + \frac{y^Tysy^T}{(y^Ts)^2} = I$$

Im Induktionsschritt geht man von $k = 0$ auf allgemeines k, dann erhält man folgenden Ansatz (für $B = A^{-1}$, k wurde zur besseren Übersichtlichkeit weggelassen):

$$\left(A^{-1} + \frac{(s - A^{-1}y)s^T + s(s - A^{-1}y)^T}{y^Ts} - \frac{(s - A^{-1}y)^Tyss^T}{(y^Ts)^2}\right) \cdot \left(A - \frac{As(As)^T}{s^TAs} + \frac{yy^T}{y^Ts}\right)$$

Die einzeln Produkte werden wieder aufgeteilt:

$$\left(A^{-1}\right) \cdot (A) = I$$

$$\left(A^{-1}\right) \cdot \left(-\frac{As(As)^T}{s^TAs}\right) = -\frac{s(As)^T}{s^TAs} = -\frac{ss^TA}{s^TAs}$$

$$\left(A^{-1}\right) \cdot \left(\frac{yy^T}{y^Ts}\right) = \frac{A^{-1}yy^T}{y^Ts}$$

$$\left(\frac{(s - A^{-1}y)s^T + s(s - A^{-1}y)^T}{y^Ts}\right) \cdot (A) = \frac{ss^TA - A^{-1}ys^TA + ss^TA - sy^TA^{-1}A}{y^Ts}$$

$$= \frac{ss^TA}{y^Ts} - \frac{A^{-1}ys^TA}{y^Ts} + \frac{ss^TA}{y^Ts} - \frac{sy^T}{y^Ts}$$

$$\left(\frac{(s-A^{-1}y)s^T+s(s-A^{-1}y)^T}{y^Ts}\right)\cdot\left(-\frac{As(As)^T}{s^TAs}\right)=-\frac{ss^TAss^TA-A^{-1}ys^TAss^TA+s(s^T-y^TA^{-1})Ass^TA}{y^Tss^TAs}$$

$$=-\frac{ss^TAss^TA}{y^Tss^TAs}+\frac{A^{-1}ys^TAss^TA}{y^Tss^TAs}-\frac{ss^TAss^TA}{y^Tss^TAs}+\frac{sy^Tss^TA}{y^Tss^TAs}$$

$$=-\frac{ss^TA}{y^Ts}+\frac{A^{-1}ys^TA}{y^Ts}-\frac{ss^TA}{y^Ts}+\frac{ss^TA}{s^TAs}$$

$$\left(\frac{(s-A^{-1}y)s^T+s(s-A^{-1}y)^T}{y^Ts}\right)\cdot\left(\frac{yy^T}{y^Ts}\right)=\frac{ss^Tyy^T-A^{-1}ys^Tyy^T+ss^Tyy^T-sy^TA^{-1}yy^T}{(y^Ts)^2}$$

$$=\frac{ss^Tyy^T}{(y^Ts)^2}-\frac{A^{-1}ys^Tyy^T}{(y^Ts)^2}+\frac{ss^Tyy^T}{(y^Ts)^2}-\frac{sy^TA^{-1}yy^T}{(y^Ts)^2}$$

$$=\frac{sy^T}{y^Ts}-\frac{A^{-1}yy^T}{y^Ts}+\frac{sy^T}{y^Ts}-\frac{sy^TA^{-1}yy^T}{(y^Ts)^2}$$

$$\left(-\frac{(s-A^{-1}y)^Tyss^T}{(y^Ts)^2}\right)\cdot(A)=-\frac{s^Tyss^TA-y^TA^{-1}yss^TA}{(y^Ts)^2}$$

$$=-\frac{s^Tyss^TA}{(y^Ts)^2}+\frac{y^TA^{-1}yss^TA}{(y^Ts)^2}=-\frac{ss^TA}{y^Ts}+\frac{y^TA^{-1}yss^TA}{(y^Ts)^2}$$

$$\left(-\frac{(s-A^{-1}y)^Tyss^T}{(y^Ts)^2}\right)\cdot\left(-\frac{As(As)^T}{s^TAs}\right)=\frac{s^Tyss^TAss^TA-y^TA^{-1}yss^TAss^TA}{(y^Ts)^2s^TAs}$$

$$=\frac{s^Tyss^TAss^TA}{(y^Ts)^2s^TAs}-\frac{y^TA^{-1}yss^TAss^TA}{(y^Ts)^2s^TAs}=\frac{ss^TA}{y^Ts}-\frac{y^TA^{-1}yss^TA}{(y^Ts)^2}$$

$$\left(-\frac{(s-A^{-1}y)^Tyss^T}{(y^Ts)^2}\right)\cdot\left(\frac{yy^T}{y^Ts}\right)=-\frac{s^Tyss^Tyy^T-y^TA^{-1}yss^Tyy^T}{(y^Ts)^3}$$

$$=-\frac{s^Tyss^Tyy^T}{(y^Ts)^3}+\frac{y^TA^{-1}yss^Tyy^T}{(y^Ts)^3}=-\frac{sy^T}{y^Ts}+\frac{y^TA^{-1}ysy^T}{(y^Ts)^2}$$

Zusammenfassen

$$I-\frac{ss^TA}{s^TAs}+\frac{A^{-1}yy^T}{y^Ts}+\frac{ss^TA}{y^Ts}-\frac{A^{-1}ys^TA}{y^Ts}+\frac{ss^TA}{y^Ts}-\frac{sy^T}{y^Ts}-\frac{ss^TA}{y^Ts}+\frac{A^{-1}ys^TA}{y^Ts}-\frac{ss^TA}{y^Ts}$$

$$+\frac{ss^TA}{s^TAs}+\frac{sy^T}{y^Ts}-\frac{A^{-1}yy^T}{y^Ts}+\frac{sy^T}{y^Ts}-\frac{sy^TA^{-1}yy^T}{(y^Ts)^2}-\frac{ss^TA}{y^Ts}+\frac{y^TA^{-1}yss^TA}{(y^Ts)^2}$$

$$+\frac{ss^TA}{y^Ts} - \frac{y^TA^{-1}yss^TA}{(y^Ts)^2} - \frac{sy^T}{y^Ts} + \frac{y^TA^{-1}ysy^T}{(y^Ts)^2} = I$$

Da $y^TA^{-1}y$ ein Skalar ist, kann es beliebig verschoben werden, d.h. $sy^TA^{-1}yy^T = y^TA^{-1}ysy^T$.

Für $k+1$ ändert sich an der oben gezeigten Rechnung nichts. Jeder Vektor bzw. jede Matrix erhält dann einfach nur den Index $k+1$.

Aufgabe 3

Wenn die Suchrichtung des BFGS Verfahrens:

$$d = -B \cdot \nabla f(x)$$

keine Abstiegsrichtung ist, d.h. die Bedingung:

$$\nabla f(x^k)^T \cdot d^k \leq -\rho \cdot ||d^k||^p$$

nicht erfüllt ist, muss das Verfahren 'resettet' werden. Das Newton-Verfahren sucht nach einem stationären Punkt (d.h. Gradient = 0). Da der Gradient aber nicht nur in einem Minimum sondern auch in einem Maximum oder Sattelpunkt gleich Null sein kann, gibt das BFGS-Verfahren unter Umständen keine Abstiegsrichtungen aus. In diesem Fall bietet es sich an, die Suchrichtung auf den negativen Gradienten zu setzen:

$$d = -\nabla f(x)$$

Da nun die Abstiegsrichtung nicht mehr zur approximierten Inversen der Hesse-Matrix B passt, muss diese ebenfalls für den nächsten Schritt neu bestimmt werden (bzw. das nächste Update erfolgt dann mit dieser Matrix).

Hierzu bieten sich verschiedene Möglichkeiten an:

1. Wie beim Start des BFGS-Verfahrens $B = I$ setzen
   - Hierbei geht jeglicher berechnete Fortschritt verloren, es handelt sich um einen recht naiven Ansatz
2. Die Hesse-Matrix einmalig aus Differenzenquotienten des Gradienten bestimmen und anschließend invertieren
   - Hoher Rechenaufwand von $\mathcal{O}(n^2)$ für die Hesse-Matrix und nochmal $\mathcal{O}(n^3)$ für das Invertieren
   - Problem wenn die Hesse-Matrix nicht invertierbar ist bzw. aufgrund von Auslöschung oder anderen numerischen Fehlern die Inverse schlecht konditioniert ist
   - Bisher berechneter Fortschritt geht ebenfalls verloren aber die Approximation der Hesse-Matrix ist sehr genau

3. Man könnte, wie in den Vorlesungsfolien beschrieben, $\frac{y^T s}{y^T y} \cdot I_n$ als positiv-definitve Matrix verwenden

    - Der Fortschritt geht zwar ähnlich wie bei $B = I$ weitestgehend verloren, allerdings überträgt man noch Informationen von y und s auf den nächsten Schritt, erscheint also sinnvoller als die erste Variante

Da (2) gerade durch das BFGS-Verfahren vermieden werden soll da zu aufwändig und (1) keine Informationen über den vorherigen Verlauf des Algorithmus enthält, ist (3) die beste Wahl für die ich mich auch in meiner Implementierung entschieden habe.

Aufgabe 4

Auswertungen der Himmelblau-Funktion

$$f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$

Mit Gradient:

$$\nabla f(x_1, x_2) = \begin{bmatrix} 2 \cdot (x_1^2 + x_2 - 11) \cdot 2x_1 + 2 \cdot (x_1 + x_2^2 - 7) \\ 2 \cdot (x_1^2 + x_2 - 11) + 2 \cdot (x_1 + x_2^2 - 7) \cdot 2x_2 \end{bmatrix}$$

Schritt	**x**	$f(x)$
1	$[0.00, -1.00]^T$	180
2	$[4.50, -1.00]^T$	70.31
3	$[2.54, -1.86]^T$	41.79
4	$[2.68, -2.26]^T$	37.84
5	$[4.16, -1.94]^T$	20.24
⋮	⋮	⋮
13	$[3.58, -1.85]^T$	$6.37 \cdot 10^{-20}$

Abbildung 1: Verlauf von `InverseBFGS` für f bei einer Genauigkeit von 10^{-8}. Man erkennt gut die superlineare Konvergenz.

Schritt	x	$f(x)$
1	$[0.00, -1.00]^T$	180
2	keine Info	06.38
3	keine Info	05.91
4	keine Info	00.26
5	keine Info	0.00
$\vdots$	$\vdots$	$\vdots$
9	$[3.58, -1.85]^T$	$7.05 \cdot 10^{-13}$

Abbildung 2: Verlauf von `fminunc` für f bei einer Genauigkeit von 10^{-8} mit Einstellungen `Algorithm = 'quasi-newton'`, `HessUpdate = 'bfgs'` und `Display = 'iter-detailed'`.

Die Rosenbrock-Funktion ist in `Projekt2.m` als `f_rosen_mult` definiert. Beispielhaft für $N = 3$

$$g(x_1, x_2, x_3) = \sum_{i=1}^{2} (1 - x_i)^2 + 100 \cdot (x_{i+1} - x_i^2)^2$$
$$=$$
$$(1 - x_1)^2 + 100 \cdot (x_2 - x_1^2)^2 + (1 - x_2)^2 + 100 \cdot (x_3 - x_2^2)^2$$

habe ich den Gradienten bestimmt:

$$\nabla g_3 = \begin{bmatrix} -2 \cdot (1 - x_1) + 200 \cdot (x_2 - x_1^2) \cdot (-2x_1) \\ 200 \cdot (x_2 - x_1^2) - 2 \cdot (1 - x_2) + 200 \cdot (x_3 - x_2^2) \cdot (-2x_2) \\ 200 \cdot (x_3 - x_2^2) \end{bmatrix}$$

Man erkennt eine Regel: Der Gradient besteht aus drei Teilen. Der erste Eintrag im Gradienten ist:

$$-2 \cdot (1 - x_1) + 200 \cdot (x_2 - x_1^2) \cdot (-2x_1)$$

Alle weiteren Einträge (bis auf den letzten an Position $N - 1$) sind:

$$200 \cdot (x_i - x_{i-1}^2) - 2 \cdot (1 - x_i) + 200 \cdot (x_{i+1} - x_i^2) \cdot (-2x_i)$$

Der letzte Eintrag ist:

$$200 \cdot (x_N - x_{N-1}^2)$$

Die Ableitung ist in `Projekt2.m` in der Funktion `f_rosen_mult_deriv_func` definiert.

Schritt	x	$f(x)$
1	$[-1.0, -1.0, \dots, -1.0]^T$	3636.00
2	$[0.57, 1.35, \dots, -0.22]^T$	682.94
3	$[0.94, 1.25, \dots, 0.20]^T$	508.69
4	$[1.33, 1.29, \dots, 0.52]^T$	419.10
5	$[1.45, 1.40, \dots, 1.43]^T$	379.44
$\vdots$	$\vdots$	$\vdots$
49	$[1.00, 1.00, \dots, 1.00]^T$	$2.2 \cdot 10^{-21}$

Abbildung 3: Verlauf von `InverseBFGS` für die 10-dim. Rosenbrock-Funktion bei einer Genauigkeit von 10^{-8}.

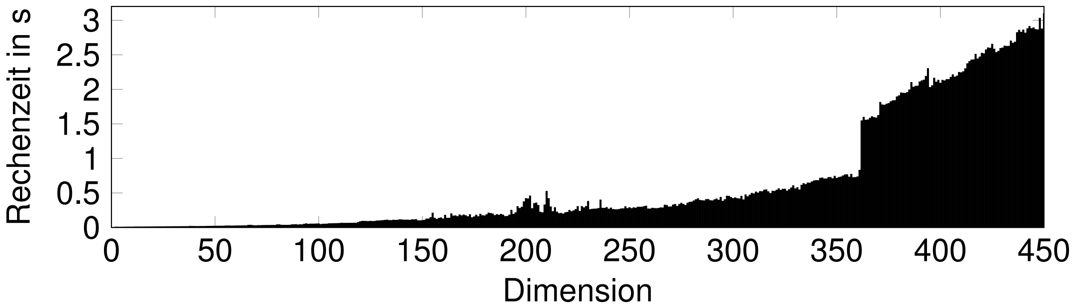

Abbildung 4: Rechenzeit der N-dimensionalen Rosenbrock-Funktion mit Startwert $[-1, \dots, -1]^T$ für eine Genauigkeit von 10^{-8} gemittelt über 100 Durchläufe von `InverseBFGS` (Intel Core i3-7100U, 8GB RAM, Windows 10 64-Bit)

Schritt	x	$f(x)$
1	$[-1.0, -1.0, \dots, -1.0]^T$	3636.00
2	keine Info	55.59
3	keine Info	41.02
4	keine Info	9.56
5	keine Info	9.49
$\vdots$	$\vdots$	$\vdots$
79	$[1.00, 1.00, \dots, 1.00]^T$	$6.07 \cdot 10^{-11}$

Abbildung 5: Verlauf von `fminunc` für die 10-dim. Rosenbrock-Funktion bei einer Genauigkeit von 10^{-8} mit Einstellungen `Algorithm = 'quasi-newton'`, `HessUpdate = 'bfgs'` und `Display = 'iter-detailed'`.

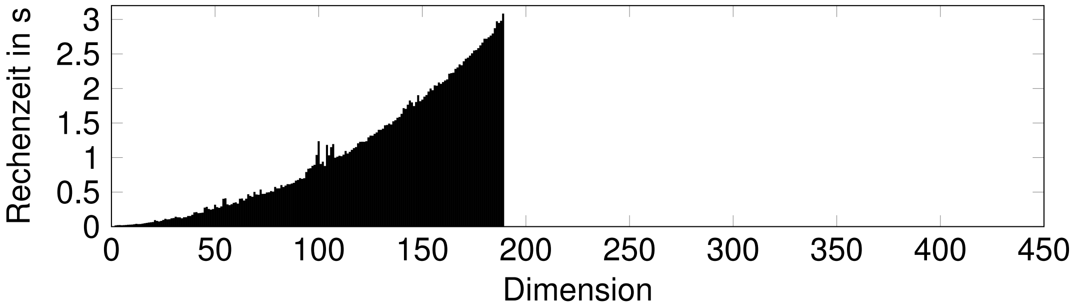

Abbildung 6: Rechenzeit der N-dimensionalen Rosenbrock-Funktion mit Startwert $[-1,\ldots,-1]^T$ für eine Genauigkeit von 10^{-8} gemittelt über 100 Durchläufe von `fminunc` mit `Algorithm = 'quasi-newton'`, `HessUpdate = 'bfgs'` (Intel Core i3-7100U, 8GB RAM, Windows 10 64-Bit)

Man erkennt, das mit höheren Dimensionen die Rechenzeit für beide Verfahren drastisch zunimmt und `fminunc` deutlich früher 3s pro Durchlauf benötigt.

Verfahren	**Dimensionen**	**Iterationen**	**Genauigkeit**	**Anmerkung**
`InverseBFGS`	750	941	10^{-8}	Zielfunktionswert: 3.98, die ersten 8 Werte bei 0.9967
`InverseBFGS`	850	1095	10^{-8}	Zielfunktionswert: $1.4 \cdot 10^{-20}$, keine Werte $\neq 1$
`InverseBFGS`	925	1079	10^{-8}	Zielfunktionswert: $5.8 \cdot 10^{-21}$, keine Werte $\neq 1$
`fminunc`	750	3936	10^{-8}	Zielfunktionswert: $1.8 \cdot 10^{-10}$, keine Werte $\neq 1$
`fminunc`	850	4455	10^{-8}	Zielfunktionswert: $1.7 \cdot 10^{-10}$, keine Werte $\neq 1$
`fminunc`	925	4852	10^{-8}	Zielfunktionswert: $1.9 \cdot 10^{-10}$, keine Werte $\neq 1$

Abbildung 7: Unterschiede zwischen beiden Verfahren bei der Berechnung der N-dimensionalen Rosenbrock-Funktion bei hohen Dimensionen

Möglichkeiten um ein höheres N zu erreichen:

- Nach den Regeln der Analysis lässt sich eine Summe aufteilen in zwei Summen:

$$\sum_{i=1}^{N-1}(1-x_i)^2 + 100 \cdot (x_{i+1}-x_i^2)^2$$
$$=$$
$$\sum_{i=1}^{a}(1-x_i)^2 + 100 \cdot (x_{i+1}-x_i^2)^2 + \sum_{i=a+1}^{N-1}(1-x_i)^2 + 100 \cdot (x_{i+1}-x_i^2)^2$$

Die Berechnung des Minimums dieser zwei Summen kann auf z.B. zwei Threads aufgeteilt und am Ende wieder zusammengefügt werden. Bei ungeradem N muss man sich entscheiden wie die Summe aufgeteilt wird, ein Thread bearbeitet dann eine Dimension mehr als der andere. Bei besonders großem N können auch diese beiden Summen wiederum aufgeteilt und somit auf noch mehr Threads verteilt werden.

- Verwenden von sparse-Matritzen und Vektoren durch die sich die Rechenzeit unter Umständen reduzieren kann. Sparse-Datenstrukturen verwenden eine spezielle Repräsentation der Werte in denen Einträge mit 0 effizienter gespeichert werden. Aufgrundessen beschleunigt sich die Berechnung von Matrix-Vektor-Produkten (die beim BFGS-Verfahren sehr oft verwendet werden)
- Gegebenfalls genauere Untersuchungen über Kondition und Stabilität der Operationen bzw. wie diese verbessert werden können. Durch Auslöschung in der Update-Formel könnte es unter Umständen zu fehlerhaften Richtungsvektoren kommen die sich bei besonders großen Problemen potenzieren und somit die Konvergenz verlangsamen.

Aufgabe 5

Die gegebene symmetrische Rang-1 Formel erhält man durch eine Rang-1 Modifikation

$$A^{k+1} = A^k + c \cdot u \cdot v^T$$

mit $c = \frac{1}{u^T s}$, $u = y - A^k s$ und $v = c \cdot u$.

Diese kann man nun in die Sherman-Morrison-Woodbury-Formel einsetzen und erhält somit den Ansatz:

$$A^{-1} - \frac{A^{-1}(y - As)\left(\frac{y-As}{(y-As)^T s}\right)^T A^{-1}}{1 + \left(\frac{y-As}{(y-As)^T s}\right)^T A^{-1}(y - As)}$$

Wenn man das T in den Bruch im Nenner und Zähler einfließen lässt, erhält man diesen Doppelbruch:

$$A^{-1} - \frac{\frac{A^{-1}(y-As)(y-As)^T A^{-1}}{((y-As)^T s)^T}}{1 + \frac{(y-As)^T A^{-1}(y-As)}{((y-As)^T s)^T}}$$

Nun den Zähler vom Zähler ausmultipliziert:

$$A^{-1} - \frac{\frac{(A^{-1}y - A^{-1}As)(y^T - (As)^T)A^{-1}}{((y-As)^T s)^T}}{1 + \frac{(y-As)^T A^{-1}(y-As)}{((y-As)^T s)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^T-s^TA)A^{-1}}{((y-As)^Ts)^T}}{1+\frac{(y-As)^TA^{-1}(y-As)}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^TAA^{-1})}{((y-As)^Ts)^T}}{1+\frac{(y-As)^TA^{-1}(y-As)}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{(y-As)^TA^{-1}(y-As)}{((y-As)^Ts)^T}}$$

Ebenfalls den Zähler im Nenner ausmultiplizieren:

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{(y^T-(As)^T)A^{-1}(y-As)}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{(y^T-s^TA)A^{-1}(y-As)}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{(y^TA^{-1}-s^T)(y-As)}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{y^TA^{-1}y-y^TA^{-1}As-s^Ty+s^TAs}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{y^TA^{-1}y-y^Ts-s^Ty+s^TAs}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{1+\frac{y^T(A^{-1}y-s)-s^T(y-As)}{((y-As)^Ts)^T}}$$

Die 1 im Nenner erweitern:

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{\frac{((y-As)^Ts)^T}{((y-As)^Ts)^T}+\frac{y^T(A^{-1}y-s)-s^T(y-As)}{((y-As)^Ts)^T}}$$

mit dem zweiten Term verbinden:

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{\frac{((y-As)^Ts)^T+y^T(A^{-1}y-s)-s^T(y-As)}{((y-As)^Ts)^T}}$$

ausmultiplizieren:

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{\frac{s^T(y-As)+y^TA^{-1}y-y^Ts-s^Ty+s^TAs}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{\frac{s^Ty-s^TAs+y^TA^{-1}y-y^Ts-s^Ty+s^TAs}{((y-As)^Ts)^T}}$$

und kürzen:

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{\frac{y^TA^{-1}y-y^Ts}{((y-As)^Ts)^T}}$$

$$A^{-1} - \frac{\frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T}}{\frac{y^T(A^{-1}y-s)}{((y-As)^Ts)^T}}$$

Den Doppelbruch auflösen:

$$A^{-1} - \frac{(A^{-1}y-s)(y^TA^{-1}-s^T)}{((y-As)^Ts)^T} \cdot \frac{\left((y-As)^Ts\right)^T}{y^T(A^{-1}y-s)}$$

Wenn man sich die nun gegenüberstehenden Terme ansieht:

$$\left((y-As)^Ts\right)^T$$

erkennt man, dass es sich hierbei um eine Zahl handelt (beispielhaft für ein 2×2-System):

$$\left(\left(\begin{bmatrix}1\\1\end{bmatrix} - \begin{bmatrix}2 & 3\\3 & -1\end{bmatrix} \cdot \begin{bmatrix}3\\2\end{bmatrix}\right)^T \cdot \begin{bmatrix}3\\2\end{bmatrix}\right)^T$$

$$\left(\left(\begin{bmatrix}1\\1\end{bmatrix} - \begin{bmatrix}12\\7\end{bmatrix}\right)^T \cdot \begin{bmatrix}3\\2\end{bmatrix}\right)^T$$

$$\left(\left(\begin{bmatrix}-11\\-6\end{bmatrix}\right)^T \cdot \begin{bmatrix}3\\2\end{bmatrix}\right)^T$$

$$\left(\begin{bmatrix}-11 & -6\end{bmatrix} \cdot \begin{bmatrix}3\\2\end{bmatrix}\right)^T$$

$$(-45)^T = -45$$

D.h. man darf den Doppelbruch kürzen, dann erhält man:

$$A^{-1} - \frac{(A^{-1}y - s)(y^T A^{-1} - s^T)}{y^T(A^{-1}y - s)}$$

Um den Ausdruck noch schöner zu machen, kann man jeweils ein Minus aus den zwei Termen im Zähler herausziehen:

$$A^{-1} - \frac{(-A^{-1}y + s)(-y^T A^{-1} + s^T)}{y^T(A^{-1}y - s)}$$

und Summanden vertauschen:

$$A^{-1} - \frac{(s - A^{-1}y)(s^T - y^T A^{-1})}{y^T(A^{-1}y - s)}$$

Aus dem rechten Term kann man aufgrund der Symmetrie das Transponieren herausziehen:

$$A^{-1} - \frac{(s - A^{-1}y)(s - A^{-1}y)^T}{y^T(A^{-1}y - s)}$$

Weiterhin kann man aus dem Nenner ebenfalls ein Minus entwenden und die Summanden vertauschen:

$$A^{-1} + \frac{(s - A^{-1}y)(s - A^{-1}y)^T}{y^T(s - A^{-1}y)}$$

Da es sich beim Nenner ebenfalls um eine Zahl handelt (beispielhaft für ein 2×2-System):

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \left(\begin{bmatrix} 3 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 & -2 \\ -2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right)$$

kann man ihn auch so schreiben:

$$A^{-1} + \frac{(s - A^{-1}y)(s - A^{-1}y)^T}{(s - A^{-1}y)^T y}$$

Damit erhält man nun die Update-Formel für das Inverse-Verfahren:

$$B^{(k+1)} = B^k + \frac{(s^k - B^k y^k)(s^k - B^k y^k)^T}{(s^k - B^k y^k)^T y^k}$$

Aufgabe 6

Siehe `GaussNewton.m`.

Aufgabe 7

Die erste Funktion

$$f(t, x_1, x_2) = x_1 \cdot e^{x_2 \cdot t}$$

mit partiellen Ableitungen:

$$\frac{df}{dx_1} = e^{x_2 \cdot t}, \frac{df}{dx_2} = t \cdot x_1 \cdot e^{x_2 \cdot t}$$

hat mit dem gegebenen Datensatz folgenden Residuen-Vektor:

$$r = \begin{bmatrix} f(0, x_1, x_2) - 2 \\ f(1, x_1, x_2) - 0.7 \\ f(2, x_1, x_2) - 0.3 \\ f(3, x_1, x_2) - 0.1 \end{bmatrix}$$

und diese Jacobi-Matrix (r'):

$$J = \begin{bmatrix} \frac{df}{dx_1}(0, x_1, x_2) & \frac{df}{dx_2}(0, x_1, x_2) \\ \frac{df}{dx_1}(1, x_1, x_2) & \frac{df}{dx_2}(1, x_1, x_2) \\ \frac{df}{dx_1}(2, x_1, x_2) & \frac{df}{dx_2}(2, x_1, x_2) \\ \frac{df}{dx_1}(3, x_1, x_2) & \frac{df}{dx_2}(3, x_1, x_2) \end{bmatrix}$$

Schritt	**x**	$\sum_{i=1}^{M}(r_i)^2$
1	$[0.50, 1.00]^T$	113.07
2	$[0.39, 0.74]^T$	16.64
3	$[0.69, 0.14]^T$	2.99
4	$[1.54, -0.66]^T$	0.24
5	$[1.98, -0.99]^T$	$2.5 \cdot 10^{-3}$
⋮	⋮	⋮
10	$[2.00, -1.0]^T$	$2.0 \cdot 10^{-3}$

Abbildung 8: Verlauf der aufsummierten quadratischen Residuen mit dem gegebenen Datensatz und `GaussNewton.m` bei Abbruchbedingung $||\nabla f|| \leq 10^{-8}$

Für f erhält man somit die Werte $x_1 = 1.9950$ und $x_2 = -1.0095$ nach 10 Iterationen.

Weiterhin gilt für die zweite Funktion

$$g(t, x_1, x_2, x_3) = x_1 \cdot e^{-(x_2^2 + x_3^2) \cdot t} \cdot \frac{\sinh(x_3^2 \cdot t)}{x_3^2}$$

mit den partiellen Ableitungen

$$\frac{dg}{dx_1} = e^{-(x_2^2+x_3^2)\cdot t} \cdot \frac{\sinh(x_3^2 \cdot t)}{x_3^2}$$

$$\frac{dg}{dx_2} = -2t \cdot x_2 x_1 \cdot e^{-(x_2^2+x_3^2)\cdot t} \cdot \frac{\sinh(x_3^2 \cdot t)}{x_3^2}$$

$$\frac{dg}{dx_3} = \frac{2 \cdot x_1 \cdot e^{-(x_2^2+x_3^2)\cdot t} \cdot (t \cdot x_3^2 \cdot \cosh(x_3^2 \cdot t) - (x_3^2 \cdot t + 1) \cdot \sinh(x_3^2 \cdot t))}{x_3^2}$$

und dem gegebenen Datensatz folgender Residuen-Vektor:

$$r = \begin{bmatrix} g(6, x_1, x_2, x_3) - 24.19 \\ g(12, x_1, x_2, x_3) - 35.34 \\ \vdots \\ g(180, x_1, x_2, x_3) - 55.21 \end{bmatrix}$$

und diese Jakobi-Matrix (r'):

$$J = \begin{bmatrix} \frac{dg}{dx_1}(6, x_1, x_2, x_3) & \frac{dg}{dx_2}(6, x_1, x_2, x_3) & \frac{dg}{dx_3}(6, x_1, x_2, x_3) \\ \frac{dg}{dx_1}(12, x_1, x_2, x_3) & \frac{dg}{dx_2}(12, x_1, x_2, x_3) & \frac{dg}{dx_3}(12, x_1, x_2, x_3) \\ \vdots & \vdots & \vdots \\ \frac{dg}{dx_1}(180, x_1, x_2, x_3) & \frac{dg}{dx_2}(180, x_1, x_2, x_3) & \frac{dg}{dx_3}(180, x_1, x_2, x_3) \end{bmatrix}$$

Schritt	**x**	$\sum_{i=1}^{M}(r_i)^2$
1	$[10.0, 0.05, 0.10]^T$	$1.81 \cdot 10^6$
2	$[9.12, 0.05, 0.12]^T$	$8.58 \cdot 10^5$
3	$[8.34, 0.05, 0.13]^T$	$4.82 \cdot 10^5$
4	$[7.74, 0.05, 0.13]^T$	$2.91 \cdot 10^5$
5	$[7.19, 0.05, 0.14]^T$	$1.84 \cdot 10^5$
⋮	⋮	⋮
25	$[3.78, 0.05, 0.15]^T$	$1.24 \cdot 10^3$
⋮	⋮	⋮
250	$[3.54, 0.05, 0.15]^T$	776.75

Abbildung 9: Verlauf der aufsummierten quadratischen Residuen mit dem gegebenen Datensatz und `GaussNewton` bei Abbruchbedingung $||\nabla f|| \leq 10^{-8}$ und konstanter Schrittweite $\alpha = 0.125$

Für g erhält man mit dem gegebenen Datensatz und den Startwerten:

- $x_0 = [10, 0.05, 0.1]^T$
- $x_0 = [7, 0.125, 0.25]^T$
- $x_0 = [3, 0.1, 0.05]^T$

somit den Wert

- $x_1 = 3.5355$, $x_2 = 0.0546$, $x_3 = 0.1539$

nach 250, 247 bzw. 609 Iterationen. Die Wolfe-Powell Schrittweitensteuerung hat bei dieser Funktion nicht zum Erfolg geführt da nach etwa 10 Durchläufen eine sehr kleine Schrittweite (im Bereich $\alpha = 10^{-8}$) angenommen wurde und das Verfahren daher nicht konvergiert ist. Wählt man dagegen händisch eine kleine Schrittweite (z.B. $\alpha = 0.1$ oder je nach Startwert $\alpha = 0.05$) konvergiert das Verfahren wieder.

Schritt	**d**	α
1	$[0.1, 0.1, 0.1]^T$	0.1

Abbildung 10: Verlauf der Schrittweite α nach Wolfe&Powell bei `GaussNewton` für Funktion g

Allgemein reagiert die Funktion g sehr empfindlich auf kleine Änderungen in den Startwerten. Das liegt vermutlich an den stark unterschiedlichen Werten der einzelnen Faktoren ($x_1 \cdot e^-(x_2^2 + x_3^2) \cdot t$ ist sehr klein während $\frac{sinh(x_3^2 \cdot t)}{x_3^2}$ sehr groß wird) aufgrundessen es zu numerischer Instabilität kommt (z.B. zeigt Matlab dann `NaN`).

Aufgabe 8

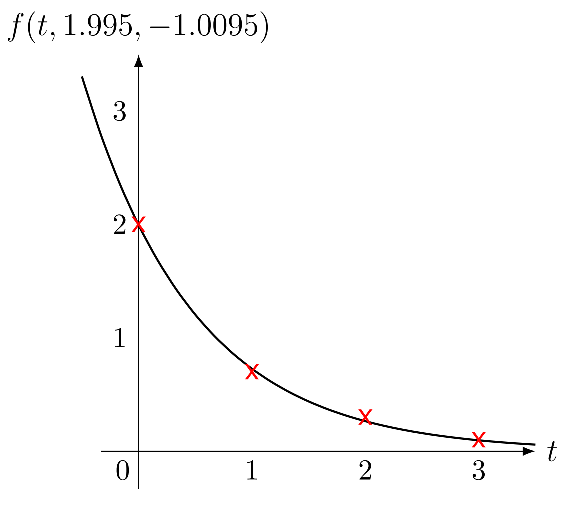

Abbildung 11: Modellfunktion und Datensatz für $f(t, x_1, x_2) = x_1 \cdot e^{x_2 \cdot t}$

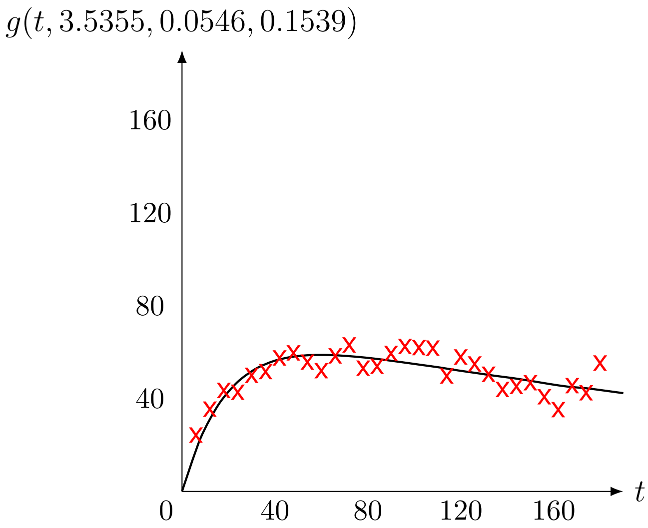

Abbildung 12: Modellfunktion und Datensatz für $g(t, x_1, x_2, x_3) = x_1 \cdot e^{-(x_2^2+x_3^2)\cdot t} \cdot \frac{sinh(x_3^2 \cdot t)}{x_3^2}$

Aufgabe 9

Bei einem Least-Squares-Problem wird der Abstand von einer Funktion f zu den gegebenen Datenpunkten (t, y) minimiert, die Zielfunktion (nach Foliensatz) ist daher:

$$f_{LeastSquares}(x) = \sum_{i=1}^{m} (f(t_i, x_1, x_2, ..., x_n) - y_i)^2$$

Es werden also alle Residuen:

$$r(x) = \begin{bmatrix} f(t_1, x_1, ..., x_n) - y_1 \\ f(t_2, x_1, ..., x_n) - y_2 \\ \vdots \end{bmatrix}$$

quadriert und aufsummiert. $f_{LeastSquares}(x)$ ist somit der erste Parameter, den `InverseBFGS` erhält.

Die BFGS-Funktion erhält ebenfalls den Gradienten als Parameter, der als: $\nabla f(x) = 2 \cdot J(x)^T \cdot r(x)$ definiert ist (mit J aus Aufgabe 7).

Schritt	x	$f(x)$
1	$[0.50, 1.00]^T$	113.07
2	$[-6.53, -9.17]^T$	73.42
3	$[4.82, -17.14]^T$	8.55
4	$[2.00, -15.16]^T$	0.59
5	$[2.00, -15.16]^T$	0.59
$\vdots$	$\vdots$	$\vdots$
19	$[1.9950, -1.0095]^T$	$2.0 \cdot 2.0^{-3}$

Abbildung 13: Verlauf von `InverseBFGS` für die Funktion f aus Aufgabe 7 bei einer Genauigkeit von 10^{-8}

Für die erste Funktion $f(t, x_1, x_2) = x_1 \cdot e^{x_2 \cdot t}$ liefert das BFGS-Verfahren den selben Wert wie GaussNewton, allerdings benötigt es etwas mehr Iterationen.

zweite Funktion liefert es jedoch z.B.
$[9.8962, 11.4152, 1.5202]^T$ da beim ersten Durchlauf der Wert des negativen Gradienten verwendet wird (da $B = I$) und dadurch die Norm des Gradienten auf 0 sinkt (Abbruchbedingung).