Vorlesung Höhere Mathematik 2

Kapitel 6: Ausgleichsrechnung

9. Februar 2023

Zürcher Hochschule
für Angewandte Wissenschaften

Gliederung des Kapitels

Einführung

- In der Auswertung von Daten ist man häufig mit dem Problem konfrontiert, Datenpunkte mit einer gewissen Streuung durch eine relativ einfache Funktion anzunähern.
- Dies geschieht mithilfe der sogn. Ausgleichsrechnung.
- Es handelt sich dabei um eine mathematische Optimierungsmethode, mit deren Hilfe für eine Reihe von Messdaten die unbekannten Parameter einer vorgegebenen Funktion bestimmt oder geschätzt werden können.
- Im allgemeinen handelt es sich dabei um überbestimmte Probleme, d.h. es liegen mehr Datenpunkte vor, als Parameter zu bestimmen sind (und damit mehr Gleichungen als Unbekannte).

Einführung

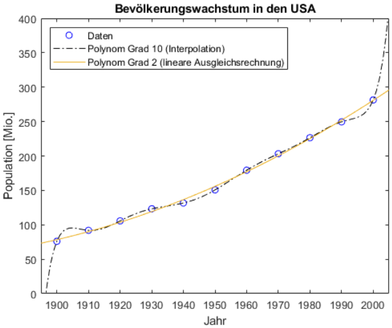
- Bei der linearen Ausgleichsrechnung führt dies zu einem überbestimmten linearen Gleichungssystem (das wir mit den Methoden aus Kap. 4 lösen können), in der nicht-linearen Ausgleichsrechnung zu einem überbestimmten nichtlinearen Gleichungssystem (wo wir die Methoden aus Kap. 5 hernanziehen).

Einführung

- Typischer Weise haben überbestimmte System keine oder keine eindeutige Lösung, es geht darum, eine optimale Funktion zu finden, die allfällige Fehler minimiert.
- In einem Spezialfall der linearen Ausgleichsrechnung jedoch, der Interpolation[1], erhalten wir ein eindeutig bestimmtes lineares Gleichungssystem mit einer eindeutigen Lösung (vgl. nächste Slide). In diesem Kapitel werden wir die Verfahren der linearen und nicht-linearen Ausgleichsrechnung kennen lernen.

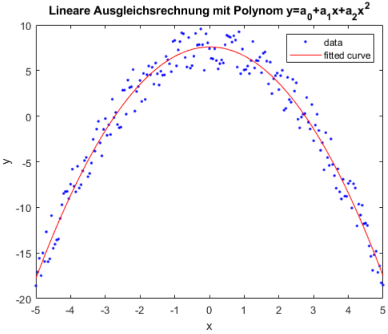
[1] Das Wort "Interpolation" stammt vom lateinischen "interpolare" (auffrischen, umgestalten, verfälschen) und bezeichnet den Versuch, fehlende Daten aufgrund bestehender Daten zu schätzen (bzw. zu interpolieren).

Einführung



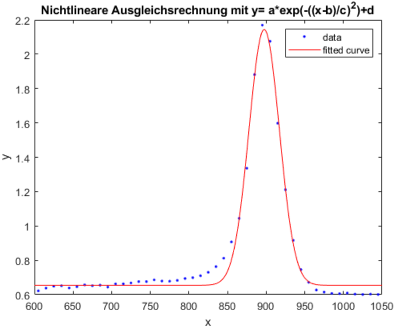
- Die Bevölkerungszahlen im Zeitraum 1900-2000 (mit 11 Datenpunkten) lassen sich durch ein Polynom vom Grad 10 exakt interpolieren. Das Polynom geht also exakt durch jeden einzelnen Datenpunkt, wie es bei einer Interpolation gefordert ist. An den Rändern des Intervalls beginnt das Polynom aber stark zu oszillieren, was nicht ideal ist. Die Datenreihe lässt sich auch mit linearere Ausgleichsrechnung durch ein Polynom niedereren Grades annähern. Das gezeigte Polynom 2. Grades geht nicht mehr exakt durch die Datenpunkte, liefert dafür aber ein stabiles Verhalten an den Rändern.

Einführung



- Ein weiteres Beispiel linearer Ausgleichsrechnung. Die 200 Datenpunkte werden durch ein Polynom 2. Grades angenähert.

Einführung



- Ein Beispiel für die nichtlineare Ausgleichsrechnung. Die Datenpunkte werden durch eine Gaussfunktion der Art $f(x) = a \cdot \mathrm{e}^{-[(x-b)/c]^2} + d$ angenähert. Die Parameter a, b, c, d werden mittels der nichtlinearen Ausgleichsrechnung bestimmt.

Lernziele

Lernziele:

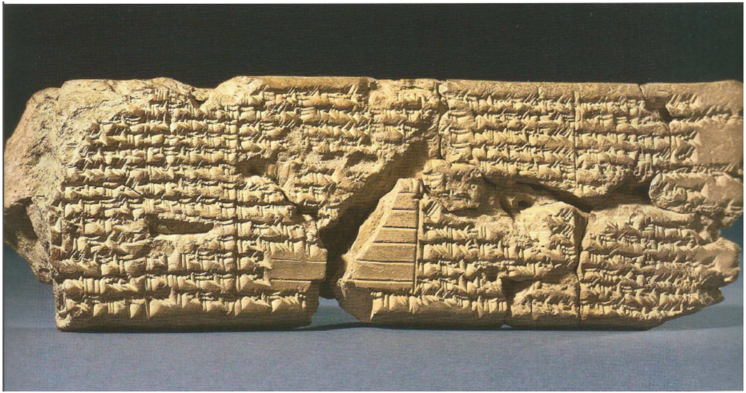
- Sie können mittels der Lagrange - Interpolationsformel eine Anzahl Messpunkte durch ein Polynom interpolieren.
- Sie können für vorgegebene Stützpunkte die natürliche kubische Splinefunktion berechnen.
- Sie können die Begriffe Ausgleichsproblem, Ansatzfunktion, Ausgleichsfunktion und Fehlerfunktioinal definieren.
- Sie kennen die Optimierung des Fehlerfunktionals im Sinne der kleinsten Fehlerquadrate (least squares).
- Sie können das lineare sowie das allgemeine Ausgleichproblem definieren und für spzeifische Beispiele lösen.
- Sie können das gedämpfte Gauss-Newton Verfahren anwenden und in Python implementieren.

6.1 Zur historischen Entwicklung

Zur historischen Entwicklung

- Die ersten Verfahren zur Interpolation gehen zurück in die frühesten Anfänge der Astronomie bzw. dem Versuch, anhand der jährlichen Veränderungen am Firmament Kalender zu entwickeln und Voraussagen zu ermöglichen.
- Die Positionen von astronomischen Objekten wie Sonne, Planeten, Kometen etc. wurden hierzu in sogn. Ephemeriden (Positionstabellen) festgehalten.
- In den Keilschriften der Babylonier bzw. eines Nachfolgereichs der Selekuiden (3.-1. Jhr. v.Chr.) wurde basierend auf solchen Ephemeriden Interpolations-Verfahren (lineare Interpolation, aber offenbar auch komplexere Verfahren) verwendet, um Lücken zu füllen.

Zur historischen Entwicklung



- Fragmentarische Ephemeriden (69 v.Chr.) aus der Selekuidenzeit. Aus »Astronomie und Astrologie« in Babylonien, M. Ossendrijver.

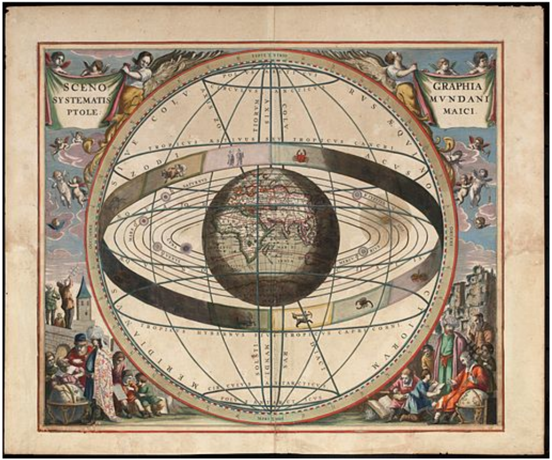
Zur historischen Entwicklung

- Die griechischen Astronomen Hipparchos (190-120 v.Chr.) und Ptolomäus (100-160 n.Chr.) verwendeten lineare Interpolation in ihren astronomischen Werken.
- Das Hauptwerk von Ptolomäus, der Almagest, zementierte das geozentrische Weltbild in Europa bis ins 16. Jahrhundert.
- Beispiele von nichtlinearer Interpolation (also z.B. die Bestimmung einer Parabel durch drei gegebene Punkte) finden sich in mehreren chinesischen (6. Jhr. n.Chr.), persichen und arabischen Werken (11. und 15. Jhr. n.Chr.).

Zur historischen Entwicklung

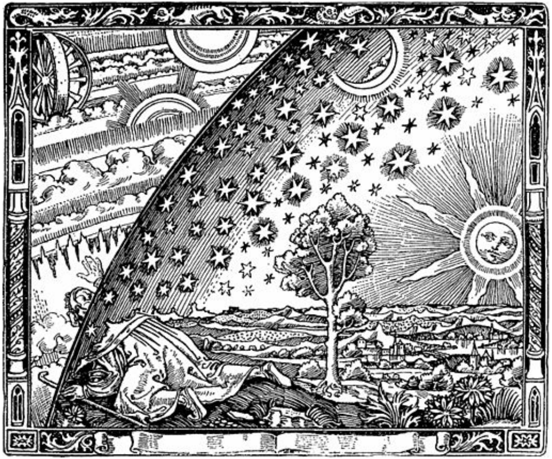
- Links: Hipparchus (Phantasiebild). Rechts: Claudius Ptolomäus (neuzeitliches Porträt)

Zur historischen Entwicklung



- Ptolomäisches (geozentrisches) Weltbild. By Loon, J. van (Johannes), ca. 1611–1686.

Zur historischen Entwicklung

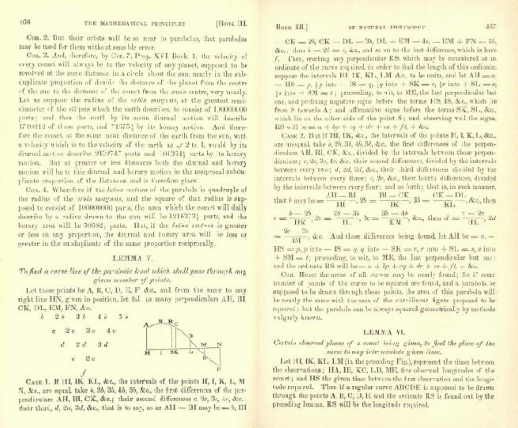


- Ablösung des geozentrischen (ptolomäischen) Weltbilds durch das kopernikanische (heliozentrische). Holzstich von Camille Flammarions

Zur historischen Entwicklung

- In Europa lieferten die Astronomen Nikolaus Kopernikus (1473-1543), Johannes Kepler (1572-1630) und Galileo Galilei (1564-1642) wichtige Beiträge zur Theorie der Interpolation, welche in den Verfahren von Issac Newton (1643-1727) weiterentwickelt wurden.
- So stellte Newton das Problem "To find a curved line of the parabolic kind which shall pass through any given number of points" (hier ist parabolisch gleichzusetzen mit polynominal) und wendete dessen Lösung im darauffolgenden Lemma an " Certain observed places of a comet being given, to find the place of the same at any intermediate given time" (vgl. nachstehende Abb.).

Zur historischen Entwicklung


466 THE MATHEMATICAL PRINCIPLES [Book III.

LEMMA V.

To find a curve line of the parabolic kind which shall pass through any given number of points.

Book III.] OF NATURAL PHILOSOPHY. 467

LEMMA VI.

Certain observed places of a comet being given, to find the place of the comet to any intermediate given time.

- Auszuüge aus Isaac Newtons Hauptwerk "Mathematical Principles of Natural Philosophy" (1687) zur Polynominterpolation.

Zur historischen Entwicklung

- Eine elegante alternative Formulierung von Newtons Interpolationsformeln gelang 1795 dem italienischen Astronom und Mathematiker Giuseppe Lodovico Lagrangia (1736-1813), besser bekannt als Joseph-Louis Lagrange (siehe Kap. 8.2 zur Lagrange-Interpolationsformel).
- Weiterentwicklungen bzw. Verallgemeinerungen wurden von den französichen Mathematikern Augustin-Louis Cauchy (1789-1857) und Charles Hermite (1822-1901) publiziert.

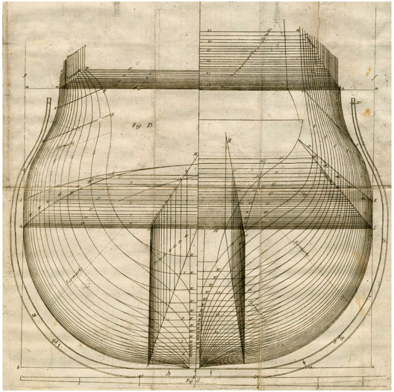
Zur historischen Entwicklung

- Ergänzend zur Theorie der Polynom-Interpolation wurde der Ansatz der stückweisen Intepolation durch sogn. Splines (siehe Kap. 6.2.3) entwickelt.
- Zu den Pionieren der Splineforschung gehörte der rumänsich-amerikanischen Mathematiker Isaac Jacob Schoenberg (1903-1990).
- Der Begriff Spline beschreibt hier eine glatte mathematische Kurve, stückweise zusammengesetzt aus Polynomen höherer (z.B. dritter) Ordnung.
- Im Schiffbau ist ein Spline eine lange dünne Latte, die an einzelnen Punkten fixiert wird und z.B. die Form des Schiffrumpfes definiert.
- Splines und weitere Interpolationstechniken kommen heute vor allem in der Signal- und Bildverarbeitung zum Einsatz.

Zur historischen Entwicklung

- Isaac Jacob Schoenberg (1976)

Zur historischen Entwicklung



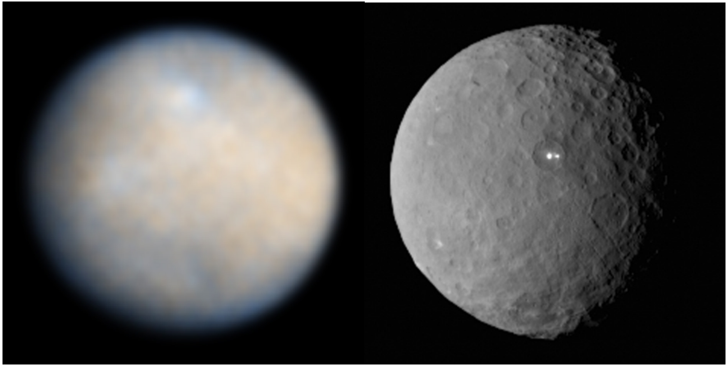
- Konstruktionsentwurf eines Schiffes. Quelle:Source: William Sutherland, The Shipbuilders Assistant: or, Some Essays Towards Completing the Art of Marine Architecture (London, 1711), 82.

Zur historischen Entwicklung

- Die Ausgleichrechnung geht zurück auf die Methode der kleinsten Quadrate von Gauss (1777-1855).
- Am Neujahrstag 1801 entdeckte der italienische Astronom Giuseppe Piazzi (1746-1826) den Zwergplaneten Ceres. 40 Tage lang konnte er die Bahn verfolgen, dann verschwand Ceres hinter der Sonne.
- Im Laufe des Jahres versuchten viele Wissenschaftler erfolglos, anhand von Piazzis Beobachtungen die Bahn zu berechnen – unter der Annahme einer Kreisbahn, denn nur für solche konnten damals die Bahnelemente aus beobachteten Himmelspositionen mathematisch ermittelt werden.

Zur historischen Entwicklung

Der Zwergplanet Ceres: von der Erde aus (links) und von der Raumsonde Dawn



(NASA, ESA, J. Parker (Southwest Research Institute), P. Thomas (Cornell University), and L. McFadden (University of Maryland, College Park, siehe

Zur historischen Entwicklung

Grössenvergleich von Ceres:

Zur historischen Entwicklung

- Der 24-jährige Gauß hingegen konnte auch elliptische Bahnen aus drei Einzelbeobachtungen berechnen.
- Da aber deutlich mehr Bahnpunkte vorlagen, wandte er seine Methode der kleinsten Quadrate an, um so die Genauigkeit zu erhöhen.
- Als Astronomen im Dezember 1801 den Kleinplaneten genau an dem von Gauß vorhergesagten Ort wiederfanden, war das nicht nur ein großer Erfolg für Gauß' Methode: Piazzis Ruf, der aufgrund seiner nicht zu einer Kreisbahn passen wollenden Bahnpunkte stark gelitten hatte, war ebenfalls wiederhergestellt.

Zur historischen Entwicklung

Carl Friedrich Gauss (1777-1855). Ausschnitt aus einem Gemälde von Gottlieb Biermann (1887).

Zur historischen Entwicklung

- Die Grundlagen seines Verfahrens hatte Gauß schon 1795 im Alter von 18 Jahren entwickelt.
- Basis war eine Idee des französischen Mathematikers Pierre-Simon Laplace (1749-1827), die Beträge von Fehlern aufzusummieren, so dass sich die Fehler zu Null addieren.
- Gauß nahm stattdessen die Fehlerquadrate und konnte die Nullsummen-Anforderung an die Fehler weglassen.
- Unabhängig davon entwickelte der französische Mathematiker Adrien-Marie Legendre (1752-1833) dieselbe Methode erstmals im Jahre 1805 am Schluss eines kleinen Werkes über die Berechnung der Kometenbahnen und veröffentlichte eine zweite Abhandlung darüber im Jahr 1810.
- Von ihm stammt der Name Méthode des moindres carrés (Methode der kleinsten Quadrate).

Zur historischen Entwicklung

Links: Pierre-Simon Laplace (Kupferstich aus dem 19. Jhr.). Mitte: der Politiker Louis Legendre (1752-1797), dessen Bild offenbar während fast 200 Jahren fälschlicherweise für dasjenige des Mathematikers Adrien-Marie Legendre gehalten worden war, bis 2005 der Irrtum entdeckt wurde.Rechts: Karrikatur des Mathematikers Adrien-Marie Legendre (1752-1833).

Zur historischen Entwicklung

- 1809 publizierte Gauß dann im zweiten Band seines himmelsmechanischen Werkes Theoria motus corporum coelestium in sectionibus conicis solem ambientium (Theorie der Bewegung der Himmelskörper, welche in Kegelschnitten die Sonne umlaufen) sein Verfahren, inklusive der Normalgleichungen und des Gaußschen Eliminationsverfahrens.
- Dabei erwähnte er, dass er es schon vor Legendre entdeckt und benutzt habe, was zu einem Prioritätsstreit zwischen den beiden führte.
- Die Methode der kleinsten Quadrate wurde nun schnell das Standardverfahren zur Behandlung von astronomischen oder geodätischen Datensätzen.

Zur historischen Entwicklung

- Gauß benutzte dann das Verfahren intensiv bei seiner Vermessung des Königreichs Hannover durch Triangulation.
- 1821 und 1823 erschien die zweiteilige Arbeit sowie 1826 eine Ergänzung zur Theoria combinationis observationum erroribus minimis obnoxiae (Theorie der den kleinsten Fehlern unterworfenen Kombination der Beobachtungen), in denen Gauß eine Begründung liefern konnte, weshalb sein Verfahren im Vergleich zu den anderen so erfolgreich war: Die Methode der kleinsten Quadrate ist in einer breiten Hinsicht optimal, also besser als andere Methoden.

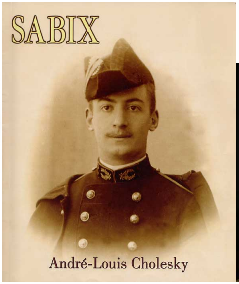
Zur historischen Entwicklung

- Die genaue Aussage ist als der Satz von Gauß-Markow bekannt, da die Arbeit von Gauß wenig Beachtung fand und schließlich im 20. Jahrhundert vom russischen Mathematiker Andrei Andrejewitsch Markow (1856-1922) wiederentdeckt und bekannt gemacht wurde.
- Theoria Combinationis enthält ferner wesentliche Fortschritte beim effizienten Lösen der auftretenden linearen Gleichungssysteme, wie das Gauß-Seidel-Verfahren und die LR-Zerlegung.

Zur historischen Entwicklung

- Der französische Vermessungsoffizier und Mathematiker André-Louis Cholesky (1875-1918) entwickelte während des Ersten Weltkrieges die Cholesky-Zerlegung, die gegenüber den Lösungsverfahren von Gauß nochmal einen erheblichen Effizienzgewinn darstellte.
- In den 1960er Jahren entwickelte der amerikanische Mathematiker Gene Golub (1932-2007) die Idee, die auftretenden linearen Gleichungssysteme mittels QR-Zerlegung zu lösen.

Zur historischen Entwicklung

Links: Andrei Andrejewitsch Markow (1856-1922). Mitte: André-Louis Cholesky (1875-1918). Rechts: Gene Golub (1932-2007).

6.2 Interpolation

Interpolation

- Die Interpolation ist ein Spezialfall der linearen Ausgleichsrechnung, bei dem wir zu einer Menge von vorgegebenen Punkten eine Funktion suchen, die exakt (und nicht nur näherungsweise) durch diese Punkte verläuft.

6.2.1 Problemstellung

Problemstellung

- Bei der Interpolation geht es z.B. darum, bei einer Wertetabelle, die Lücken aufweist, die fehlenden Funktionswerte anzunähern.
- Es sei also eine Wertetabelle einer Funktion f mit $y_i = f(x_i)$ der Art

x	x_0	x_1	x_2	$\dots$	x_{j-1}	x_j	x_{j+1}	$\dots$	x_n
y	y_0	y_1	y_2	$\dots$	y_{j-1}	?	y_{j+1}	$\dots$	y_n

  gegeben. Die $n+1$ Wertepaare (x_i, y_i) heissen Stützpunkte, die x_i Stützstellen und die y_i Stützwerte.

Problemstellung

- Gesucht ist nun eine möglichst gute Näherung des fehlenden Wertes y_j.
- Wir möchten also eine (stetige) Funktion finden, welche die eigentliche Funktion $f(x)$ an der Stelle $f(x_j)$ möglichst gut approximiert und exakt durch die bekannten Stützpunkte in der Umgebung von x_j geht.
- Eine solche Funktion nennt man Interpolierende.
- Interpolation kommt vor allem zur Anwendung, wenn die Funktion $f(x)$ nicht genügend bekannt oder nur schwer exakt zu berechnen ist. Auch in der digitalen Bildbearbeitung wird häufig interpoliert.

Problemstellung

Definition 6.1: Interpolationsproblem [1]

- Gegeben sind $n+1$ Wertepaare (x_i, y_i), $i = 0, \ldots, n$, mit $x_i \neq x_j$ für $i \neq j$. Gesucht ist eine stegige Funktion g mit der Eigenschaft $g(x_i) = y_i$ für alle $i = 0, \ldots, n$.

Beispiele 6.1

- Aus einer Temperaturmessung ergeben sich im Tagesverlauf die Werte. Gesucht ist zum Beispiel die Temperatur um 11 Uhr.

t	08.00	10.00	12.00	14.00	16.00	18.00	Uhr
T	11.2	13.4	15.3	19.5	18.9	16.1	$^\circ C$

- Wir betrachten die (nicht einfach zu berechnende) Funktion $f(x) = \int_0^x e^{\sin t} dt$, welche in tabellierter Form vorliegt. Gesucht ist der Wert $f(0.66) = ?$

x	0.4	0.5	0.6	0.7	0.8	0.9
y	0.4904	0.6449	0.8136	0.9967	1.1944	1.4063

Beispiele 6.1

- Bildverarbeitung: Ein Bild mit der Auflösung 800×800 Pixel soll vergrössert werden auf die neue Auflösung 1200×1200.
- Die zusätzlichen Pixel müssen 'gefüllt' bzw. interpoliert werden.
- Dabei werden nicht neue Informationen erzeugt, sonder nur die bestehenden Pixel sozusagen 'gestreckt'.
- Das vergrösserte Bild erscheint deshalb unscharf, aber dafür ohne Lücken.[2].

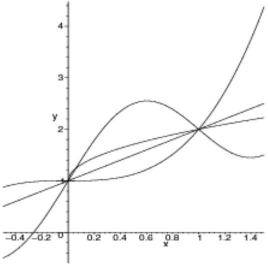
[2] Bild übernommen von http://de.wikipedia.org/w /index.php?title=Datei:ReconstructionsFromPixels.png&filetimestamp=20070524010356

Problemstellung

- Die Interpolation geschieht also ach dem folgenden Prinzip: man sucht eine Interpolierende $g(x)$, welche leicht berechenbar sein soll und für die gilt
$$g(x_i) = f(x_i)$$
für einige gegebenen Stützstellen in der Nähe von x_j.
- Es ist offensichtlich, dass bei einer solchen Fragestellung unendlich viele Lösungen möglich sind, wie im folgenden Beispiel gezeigt:

Beispiel 6.2

- Es soll eine Interpolierende für die beiden Stützpunkte (0, 1) und (1, 2) gefunden werden.
- Lösung: im einfachsten Fall könnte man eine Gerade durch die beiden Punkte legen, also $g(x) = x + 1$. Aber auch $g(x) = \sqrt{x} + 1$, $g(x) = x^3 + 1$ oder $g(x) = \sin(\pi x) + x + 1$ sind Lösungen, wie in der folgenden Abbildung[3] dargestellt.



[3] Übernommen aus Knorrenschild

6.2.2 Polynominterpolation

Polynominterpolation

- Im Folgenden betrachten wir den Fall, bei dem die Interpolierende ein Polynom ist.
- Gegeben sind $n+1$ Stützpunkte

x	x_0	x_1	x_2	$\dots$	x_n
y	y_0	y_1	y_2	$\dots$	y_n

- Gesucht ist ein Polynom $P_n(x)$, welches diese Punkte interpoliert. Wenn wir uns das Polynom in der üblichen Form nach Potenzen von x entwickelt denken, dann ist

$$P_n(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n.$$

Polynominterpolation

- Folglich gibt uns jeder Stützpunkt eine lineare Gleichung für die Bestimmung der Koeffizienten a_k.
- Wenn der Grad des Polynoms gleich n gewählt wird, erhalten wir ein lineares Gleichungssystem mit gleichvielen Gleichungen wie unbekannten Koeffzienten:

$$\begin{aligned} a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n &= y_0 \\ a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n &= y_1 \\ \vdots \quad &\vdots \quad \vdots \\ a_0 + a_1 x_n + a_2 x_n^2 + \ldots + a_n x_n^n &= y_n \end{aligned}$$

bzw.

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^n \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}.$$

Die hierbei auftretende Matrix wird Vandermonde-Matrix genannt.

Polynominterpolation

- Im Prinzip könnten wir dieses lineare Gleichungssystem mit den uns bekannten Verfahren nach den Koeffizienten nun auflösen.
- Allerdings ist dies unntötig aufwendig und die Matrix des Systems ist typischerweise schlecht konditioniert.
- Durch eine andere Darstellung des Polynoms P_n lässt sich die Berechnung vereinfachen.
- Dazu ist zu sagen, dass die Polynom-Interpolation früher beim Handrechnen einen wichtigen Stellenwert hatte. Allerdings ist die Berechnung für grosse $n \gtrsim 20$ numerisch instabil.
- Aus diesem Grund kommen, wenn viele Stützpunkte berücksichtigt werden müssen, neuere Techniken wie die Spline-Interpolation zum Zug. Zur Einführung konzentrieren wir uns aber vorderhand auf die Polynom-Interpolation.

Satz 6.1: Lagrange Interpolationsformel [2]

- Durch $n+1$ Stützpunkte mit verschiedenen Stützstellen (d.h. $x_i \neq x_j$ für $i \neq j$) gibt es genau ein Polynom $P_n(x)$ vom Grade $\leq n$, welches alle Stützpunkte interpoliert, d.h. wo gilt

$$P_n(x) = y_i, \; i = 0, 1, \ldots, n$$

- $P_n(x)$ lautet in der Lagrangeform

$$P_n(x) = \sum_{i=0}^{n} l_i(x) y_i \,,$$

dabei sind die $l_i(x)$ die Lagrangepolynome vom Grad n definiert durch

$$l_i(x) \quad = \quad \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \; i = 0, 1, \ldots, n$$

Beispiel 6.3

- Wir nehmen die Temperaturmessung aus Beispiel 6.1 und bestimmen die Temperatur um 11 Uhr. Um die Rechnung etwas zu vereinfachen, benutzen wir nur die ersten vier Messwerte

t	08.00	10.00	12.00	14.00	Uhr
T	11.2	13.4	15.3	19.5	$^\circ C$

Beispiel 6.3: Lösung

- Zur Demonstration bestimmen wir das Interpolationspolynom vollständig (d.h. für beliebige x-Werte).
- Wir haben $n+1=4$ Stützpunkte, also ist $n=3$ und das Interpolationspolynom hat die Form

$$P_n(x)=\sum_{i=0}^{3} l_i(x)y_i \;=11.2\cdot l_0(x)+13.4\cdot l_1(x)+15.3\cdot l_2(x)+19.5\cdot l_3(x).$$

Die Lagrangepolynome berechnen sich zu

$$\begin{aligned}
l_0(x) &= \frac{(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)} = \frac{(x-10)(x-12)(x-14)}{(-2)(-4)(-6)} = -\frac{1}{48}x^3+\frac{3}{4}x^2-\frac{107}{12}x+35\\
l_1(x) &= \frac{(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_0)(x_1-x_2)(x_1-x_3)} = \frac{(x-8)(x-12)(x-14)}{(2)(-2)(-4)} = +\frac{1}{16}x^3-\frac{17}{8}x^2+\frac{47}{2}x-84\\
l_2(x) &= \frac{(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_0)(x_2-x_1)(x_2-x_3)} = \frac{(x-8)(x-10)(x-14)}{(4)(2)(-2)} = -\frac{1}{16}x^3+2x^2-\frac{83}{4}x+70\\
l_3(x) &= \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_0)(x_3-x_1)(x_3-x_2)} = \frac{(x-8)(x-10)(x-12)}{(6)(4)(2)} = +\frac{1}{48}x^3-\frac{5}{8}x^2+\frac{37}{6}x-20
\end{aligned}$$

Beispiel 6.3: Lösung

- Damit ergibt sich

$$
\begin{aligned}
P_n(x) &= 11.2 \cdot l_0(x) + 13.4 \cdot l_1(x) + 15.3 \cdot l_2(x) + 19.5 \cdot l_3(x) \\
&= +\frac{13}{240}x^3 - \frac{133}{80}x^2 + \frac{2137}{120}x - \frac{263}{5}
\end{aligned}
$$

  bzw. $P_n(11) = 14.225° C$

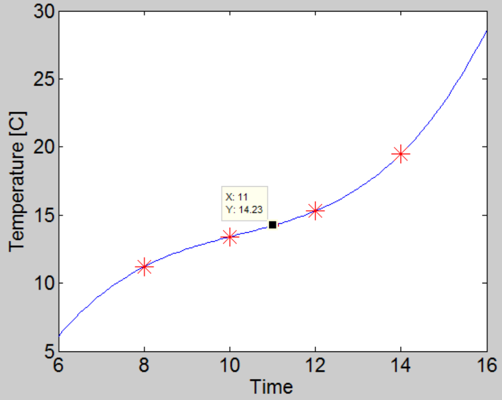
- Falls man wirklich nur den einen Wert bei $x = 11$ sucht, wäre die Rechnung einiges einfacher:

$$
\begin{aligned}
l_0(11) &= \frac{(11-10)(11-12)(11-14)}{(-2)(-4)(-6)} = \frac{(1)(-1)(-3)}{(-2)(-4)(-6)} = -\frac{1}{16} \\
l_1(11) &= \frac{(11-8)(11-12)(11-14)}{(2)(-2)(-4)} = \frac{(3)(-1)(-3)}{(2)(-2)(-4)} = \frac{9}{16} \\
l_2(11) &= \frac{(11-8)(11-10)(11-14)}{(4)(2)(-2)} = \frac{(3)(1)(-3)}{(4)(2)(-2)} = \frac{9}{16} \\
l_3(11) &= \frac{(11-8)(11-10)(11-12)}{(6)(4)(2)} = \frac{(3)(1)(-1)}{(6)(4)(2)} = -\frac{1}{16}
\end{aligned}
$$

  und damit
  $P_n(11) = 11.2 \cdot l_0(11) + 13.4 \cdot l_1(11) + 15.3 \cdot l_2(11) + 19.5 \cdot l_3(11) = 14.225,$

Beispiel 6.3: Lösung

- Die folgende Abbildung zeigt die vier Messpunkte (rot) und das Interpolationspolynom (blau) sowie den interpolierten Punkt (schwarz):



Aufgabe 6.1

- Wir betrachten die Funktion $f(x) = \int_0^x e^{\sin t} dt$ aus Beispiel 6.1. Berechnen Sie den Wert $f(0.66) = ?$ anhand der drei Punkte

x	0.6	0.7	0.8
y	0.8136	0.9967	1.1944

Aufgabe 6.1: Lösung

Bemerkung

- Wie man schnell sieht, ist es für viele Aufgabenstellungen einfacher, nicht ein Interpolationspolynom für alle gegebenen Stützpunkte zu berechnen, sondern nur die Punkte in unmittelbarer Umgebung des gesuchten Wertes zu berücksichtigen.
- Im einfachsten Fall nimmt man jeweils die zwei benachbarten Stützpunkte und verbindet sie mit einer Geraden.
- Das Interpolationspolynom ist dann also erster Ordnung und man spricht deshalb auch von *(stückweise) linearer Interpolation*.

Fehlerabschätzung

- Wie gross ist nun aber der Fehler, den wir bei einer Interpolation machen? Dazu gilt der folgende Satz:

Satz 6.2: Fehlerabschätzung [2]

- Sind die y_i Funktionswerte einer genügend oft stetig differenzierbaren Funktion f (also $y_i = f(x_i)$), dann ist der Interpolationsfehler an einer Stelle x gegeben durch

$$| f(x) - P_n(x) | \leq \frac{|(x-x_0)(x-x_1)\cdots(x-x_n)|}{(n+1)!} \max_{x_0 \leq \xi \leq x_n} | f^{(n+1)}(\xi) |$$

- Das heisst, wir müssen das Maximum der $(n+1)$-ten Ableitung der Funktion $f(x)$ auf dem Intervall $[x_0, x_n]$ kennen.
- Die Fehlerabschätzung ist also nur anwendbar, wenn wir die Funktion $f(x)$ bzw. ihre Ableitungen selbst kennen. Deshalb ist diese Fehlerabschätzung theoretischer Natur.

Aufgabe 6.2

- Von der Funktion $y = f(x) = 2^x$ seien die folgenden drei Stützpunkte gegeben:

x	-1	1	3
y	0.5	2	8

- Wie gross ist der Interpolationsfehler im Punkt $x = 2$, wenn diese Punkte durch ein Polynom interpoliert werden?
- Man schätze den Fehler mittels der obigen Fehlerabschätzung und berechne anschliessend das (allgemeine) Polynom und den exakten Fehler.

Aufgabe 6.2: Lösung

Aufgabe 6.2: Lösung

6.2.3 Spline-Interpolation

Spline-Interpolation

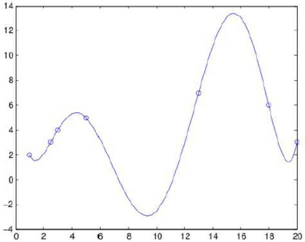
- Polynome mit einem hohen Grad oszillieren.
- Dieses Verhalten führt dazu, dass bei einer grossen Anzahl Stützpunkten das Interpolationspolynom meist keine gute Näherung mehr für die zu interpolierende Funktion $f(x)$ darstellt, wie im folgenden Beispiel dargestellt (aus [2]).

Beispiel 6.4

- Durch die Stützpunkte

x	1	2.5	3	5	13	18	20
y	2	3	4	5	7	6	3

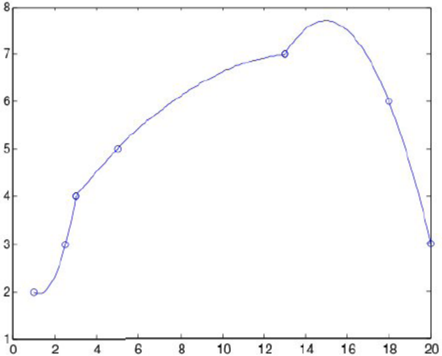
verläuft das folgenden Interpolationspolynom



- Wahrscheinlich ist dieses Polynom keine gute Näherung für die zugrundeliegende Funktion. Es könnte sich z.B. um physikalische Messwerte handeln, die immer positiv sein müssen. Dann ist eine Funktion, die zwischendurch negativ wird, nicht brauchbar.

Spline-Interpolation

- Als Alternative könnte man versuchen, die Stützpunkte stückweise zu interpolieren, z.B. jeweils drei aufeinanderfolgende Punkte durch eine Parabel anzunähern. Man erhält dann das folgende Bild (aus [2]):



Spline-Interpolation

- Hier stören jetzt vor allem die Knicke an den Übergängen.
- Die Idee der Spline-Interpolation besteht nun darin, durch Polynome niederen Grades zu interpoliern und damit die Schwingungen zu unterdrücken, wobei man gleichzeitig sicherstellt, dass keine Knicke entstehen.
- Um dies zu erreichen, müssen die Polynome an den Anschlusstellen nicht nur denselben Funktionswert, sondern auch *dieselbe Ableitung* haben.

Spline-Interpolation

- Man kann z.B. für jedes Intervall

$$[x_i, x_{i+1}],\ i = 0, 1, 2, ..., n-1$$

genau ein Polynom s_i ansetzen (der Index i bezeichnet hier nicht den Grad des Polynoms sondern die Nummer des Intervalls).

- Das Polynom muss dann folgende Bedingung erfüllen:

$$\begin{aligned} s_i(x_i) &= y_i,\, s(x_{i+1}) = y_{i+1}, \ldots \quad \textbf{Interpolation} \\ s_i(x_{i+1}) &= s_{i+1}(x_{i+1}),\, s_{i+1}(x_{i+2}) = s_{i+2}(x_{i+2}) \ldots \quad \textbf{stetiger Übergang} \\ s_i'(x_{i+1}) &= s_{i+1}'(x_{i+1}), \quad s_{i+1}'(x_{i+2}) = s_{i+2}'(x_{i+2}), \ldots \quad \textbf{keine Knicke} \end{aligned}$$

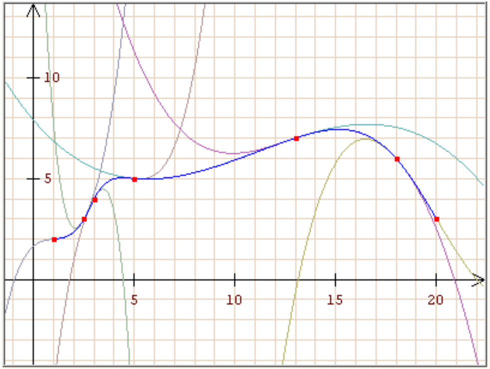
Spline-Interpolation

- Zusätzlich sollen die Polynome in den Übergangspunkten die gleiche Krümmung aufweisen, d.h. auch die zweite Ableitung soll übereinstimmen:

$$s_i''(x_{i+1}) = s_{i+1}''(x_{i+1}), \quad s_{i+1}''(x_{i+2}) = s_{i+2}''(x_{i+2}), \ldots \quad \textbf{gleiche Krümmung}$$

Spline-Interpolation

- Polynome, die all diese Bedingungen erfüllen, müssen mindestens den Grad 3 haben (siehe folgendes Beispiel). Man spricht dann auch von *kubischen Splines*.Folgende Abbildung zeigt die Approximation der obigen Stützpunkte durch kubische Splines



Spline-Interpolation

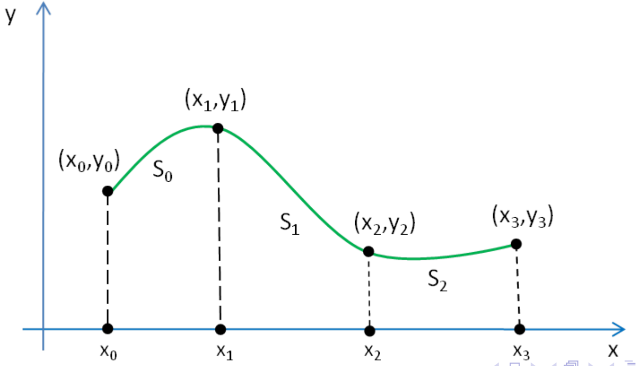
- Die Bezeichnung 'Spline' kommt aus dem Englischen und bezeichnet eine schlanke, biegsame Holzlatte, die im Schiffsbau eingesetzt wurde, um durch eine Anzahl vorgegebener Punkte eine möglichst glatte Kurve zu legen (beispielsweise für die Planken des Schiffsrumpfes).
- Kubische Splines werden unter anderem zur Berechnung des Bahnverlaufes bei Achterbahnen verwendet, um ruckartige Beschleunigungswechsel für die Fahrgäste zu vermeiden.
- Kubische Splines finden weitere Anwendung bei der exakten Verlegung der Schienen bei Hochgeschwindigkeitsstrecken der Eisenbahn.
- Auch beim Entwurf von Kurven und Oberflächen (sogen. „Freiformkurven und -flächen“), wie sie häufig im Schiff-, Flugzeug- und Automobilbau vorkommen, sind Splines von Bedeutung.

Spline-Interpolation

- Die Eignung von Splines für solche Anwendungen liegt daran, dass für jeden Polynomabschnitt Randbedingungen erstens in Form von Punkten aber auch in Form von Werten für erste und zweite Ableitung (und in Abhängigkeit davon Steigung und Krümmung/Kurvenradius) vorgeben lassen.
- Dadurch kann erreicht werden, dass die Krümmung entlang der Kurve immer stetig ist. Das hat den Vorteil, dass Querbeschleunigungen beim Abfahren der Kurve immer allmählich aufgebaut werden bzw. an den Knotenpunkten vorgegebene Werte einhalten

Beispiel 6.5

- Gegeben sind die vier Stützpunkte (x_i, y_i) für $i = 0, \ldots, 3$. Wir suchen nun die Splinefunktion S, die durch diese vier Punkte geht und sich zusammensetzt aus den drei kubischen Polynomen S_0, S_1 und S_2.



Beispiel 6.5

- Mit dem Ansatz

$$\begin{aligned}
S_0 &= a_0 + b_0(x-x_0) + c_0(x-x_0)^2 + d_0(x-x_0)^3, \; x \in [x_0, x_1] \\
S_1 &= a_1 + b_1(x-x_1) + c_1(x-x_1)^2 + d_1(x-x_1)^3, \; x \in [x_1, x_2] \\
S_2 &= a_2 + b_2(x-x_2) + c_2(x-x_2)^2 + d_2(x-x_2)^3, \; x \in [x_2, x_3]
\end{aligned}$$

  müssen wir $3 \cdot 4 = 12$ Koeffizienten berechnen, wofür wir 12 Bedingungen benötigen.

Beispiel 6.5

- Diese lauten:

  1. Interpolation der Stützpunkte:

$$\begin{aligned} S_0(x_0) &= y_0 \\ S_1(x_1) &= y_1 \\ S_2(x_2) &= y_2 \\ S_2(x_3) &= y_3 \end{aligned}$$

  2. Stetiger Übergang an den Stellen x_1 und x_2:

$$\begin{aligned} S_0(x_1) &= S_1(x_1) \\ S_1(x_2) &= S_2(x_2) \end{aligned}$$

Beispiel 6.5

3. Erste Ableitung an den Übergangsstellen muss übereinstimmen (keine Knicke):

$$\begin{aligned} S_0'(x_1) &= S_1'(x_1) \\ S_1'(x_2) &= S_2'(x_2) \end{aligned}$$

4. Zweite Ableitung an den Übergangsstellen muss übereinstimmen (gleiche Krümmung):

$$\begin{aligned} S_0''(x_1) &= S_1''(x_1) \\ S_1''(x_2) &= S_2''(x_2) \end{aligned}$$

Jetzt haben wir allerdings erst 10 Bedingungen für die 12 Koeffizienten. Das heisst, wir brauchen noch zwei zusätzliche. Diese können, in Abhängigkeit der Problemstellung, 'frei' gewählt werden und beziehen sich häufig auf die beiden Randstellen, hier x_0 und x_3.

Beispiel 6.5

Man unterscheidet zum Beispiel:

- die *natürliche kubische Splinefunktion*:

$$\begin{aligned} S_0''(x_0) &= 0 \\ S_2''(x_3) &= 0 \end{aligned}$$

  mit einem möglichen Wendepunkt im Anfangs- und Endpunkt.
- die *periodische kubische Splinefunktion*

$$\begin{aligned} S_0'(x_0) &= S_2'(x_3) \\ S_0''(x_0) &= S_2''(x_3) \end{aligned}$$

  wenn man die Periode $p = x_3 - x_0$ hat und damit $y_0 = y_3$ bzw. $S_0(x_0) = S_2(x_3)$ gilt.
- die kubische Splinefunktion mit *not-a-knot Bedingung*:

$$\begin{aligned} S_0'''(x_1) &= S_1'''(x_1) \\ S_1'''(x_2) &= S_2'''(x_2) \end{aligned}$$

Beispiel 6.5

- Übersetzt man alle 12 Gleichungen in ein 12×12 Gleichungssystem, erhält man im Falle der natürlichen kubischen Splinefunktion:

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & (x_1-x_0) & 0 & 0 & (x_1-x_0)^2 & 0 & 0 & (x_1-x_0)^3 & 0 & 0 \\
0 & 1 & 0 & 0 & (x_2-x_1) & 0 & 0 & (x_2-x_1)^2 & 0 & 0 & (x_2-x_1)^3 & 0 \\
0 & 0 & 1 & 0 & 0 & (x_3-x_2) & 0 & 0 & (x_3-x_2)^2 & 0 & 0 & (x_3-x_2)^3 \\
0 & 0 & 0 & 1 & -1 & 0 & 2(x_1-x_0) & 0 & 0 & 3(x_1-x_0)^2 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 2(x_2-x_1) & 0 & 0 & 3(x_2-x_1)^2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 6(x_1-x_0) & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 6(x_2-x_1) & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 6(x_3-x_2)
\end{pmatrix}
\begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ b_0 \\ b_1 \\ b_2 \\ c_0 \\ c_1 \\ c_2 \\ d_0 \\ d_1 \\ d_2 \end{pmatrix}
=
\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_1 \\ y_2 \\ y_3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

- Durch eine Verallgemeinerung des Gleichungssystems für beliebig viele Stützpunkte erhält man den folgenden Algorithmus zur Berechnung der Koeffizienten für die natürliche kubische Splinefunktion:

Algorithmus: natürliche kubische Splinefunktion

- Gegeben seien $n+1$ Stützpunkte (x_i, y_i) mit monoton aufsteigenden Stützstellen (Knoten) $x_0 < x_1 < \ldots < x_n$ $(n \geq 2)$. Gesucht ist die natürliche kubische Splinefunktion $S(x)$, welche in jedem Intervall $[x_i, x_{i+1}]$ mit $i = 0, 1, \ldots, n-1$ durch ein kubisches Polynom

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

dargestellt wird, also $S(x) = S_i(x)$ mit $x \in [x_i, x_{i+1}]$.

Algorithmus: natürliche kubische Splinefunktion

- Die Koeffizienten a_i, b_i, c_i und d_i der Polynome $S_i(x)$ für $i = 0, 1, \dots, n-1$ berechnen sich wie folgt:
  1. $a_i = y_i$
  2. $h_i = x_{i+1} - x_i$
  3. $c_0 = 0, \; c_n = 0$
  4. Berechnung der Koeffizienten $c_1, c_2, \dots, c_{n-1}$ aus dem Gleichungssystem
     1. $i = 1$:
        $2(h_0 + h_1)c_1 + h_1 c_2 = 3\frac{y_2 - y_1}{h_1} - 3\frac{y_1 - y_0}{h_0}$
     2. falls $n \geq 4$ gilt für $i = 2, \dots, n-2$:
        $h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = 3\frac{y_{i+1} - y_i}{h_i} - 3\frac{y_i - y_{i-1}}{h_{i-1}}$
     3. $i = n-1$:
        $h_{n-2}c_{n-2} + 2(h_{n-2} + h_{n-1})c_{n-1} = 3\frac{y_n - y_{n-1}}{h_{n-1}} - 3\frac{y_{n-1} - y_{n-2}}{h_{n-2}}$
  5. $b_i = \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}(c_{i+1} + 2c_i)$
  6. $d_i = \frac{1}{3h_i}(c_{i+1} - c_i)$

Algorithmus: natürliche kubische Splinefunktion

- Das Gleichungssystem unter Punkt 4. im obigen Algorithmus hat die Form

$$\boldsymbol{A}c = z$$

mit

$$\boldsymbol{A} = \begin{pmatrix} 2(h_0+h_1) & h_1 & & & \\ h_1 & 2(h_1+h_2) & h_2 & & \\ & h_2 & 2(h_2+h_3) & h_3 & \\ & \ddots & \ddots & \ddots & \\ & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix}$$

Algorithmus: natürliche kubische Splinefunktion

$$c = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix}, \quad \boldsymbol{z} = \begin{pmatrix} 3\frac{y_2-y_1}{h_1} - 3\frac{y_1-y_0}{h_0} \\ 3\frac{y_3-y_2}{h_2} - 3\frac{y_2-y_1}{h_1} \\ \vdots \\ 3\frac{y_n-y_{n-1}}{h_{n-1}} - 3\frac{y_{n-1}-y_{n-2}}{h_{n-2}} \end{pmatrix}$$

Aufgabe 6.3

- Zu den folgenden Stützpunkten soll die natürliche kubische Splinefunktion bestimmt werden, d.h. bestimmen Sie die Koeffizienten a_i, b_i, c_i, d_i der kubischen Polynome S_i für $i = 0, 1, 2$:

x_i	0	1	2	3
y_i	2	1	2	2

Aufgabe 6.3: Lösung

Aufgabe 6.3: Lösung

Aufgabe 6.3: Lösung

Aufgabe 6.3: Lösung

Aufgabe 6.3: Lösung

Aufgabe 6.4

- Implementieren Sie den obigen Algorithmus in Python für eine beliebige vorgegebene Reihe von Stützpunkten. Ihre Funktion soll die Stützpunkte sowie die natürliche kubische Splinefunktion plotten.
- Testen Sie Ihren Algorithmus an der Zeitreihe der Bevölkerungszahl (in Mio.) der USA:

t	1900	1910	1920	1930	1940	1950	1960	1970	1980	1990	2000
p(t)	75.995	91.972	105.711	123.203	131.669	150.697	179.323	203.212	226.505	249.633	281.422

- Benutzen Sie die Python-Funktionen
  - scipy.interpolate.spline(),

  um diese Messreihe durch eine Splinefunktion zu interpolieren und vergleichen Sie das Resultat.
- Lösung in den Übungen.

6.3 Ausgleichsrechnung

Ausgleichsrechnung

- Nach dem Spezialfall der Interpolation wenden wir uns jetzt dem allgemeinen Problem zu, wie wir eine Menge von n Datenpunkten durch eine vorgegebene Funktion $f : \mathbb{R} \longrightarrow \mathbb{R}$ annähern können.

6.3.1 Problemstellung

Problemstellung

- Im Unterschied zur Interpolation versuchen wir bei der Ausgleichsrechnung nicht, eine Funktion f zu finden, die exakt durch sämtliche n Datenpunkte geht, sondern diese nur möglichst gut approximiert.
- Dies ist insbesondere dann sinnvoll, wenn es eine grosse Anzahl von Datenpunkten gibt (die meist zusätzlich noch fehlerbehaftet sind) und diese durch eine Funktion mit nur wenigen Parametern beschrieben werden sollen.
- Es werden dabei zwei Fälle unterschieden:
  - lineare Ausgleichsrechnung
  - nicht-lineare Ausgleichsrechnung

Problemstellung

- Bei der linearen Ausgleichsrechnung ist die gesuchte Funktion $f(x)$ eine Linearkombination von m sogn. Basisfunktionen $f_i(x)$ (mit $i = 1, 2, \ldots, m$ und $m \leq n$), d.h. die gesuchten Parameter λ_i treten nur als multiplikative Faktoren bzw. als Koeffizienten der Linearkombination auf:

$$f(x) = \lambda_1 f_1(x) + \ldots + \lambda_m f_m(x)$$

Beispiele:

  - Das Polynom $f(x)$ vom Grad m setzt sich aus $m+1$ Basisfunktionen $f_i(x) = x^{i-1}$ $(i = 1, \ldots, m+1)$ zusammen :
  $f(x) = \lambda_1 \cdot \underbrace{1}_{f_1(x)} + \lambda_2 \cdot \underbrace{x}_{f_2(x)} + \lambda_3 \cdot \underbrace{x^2}_{f_3(x)} + \ldots + \lambda_{m+1} \cdot \underbrace{x^m}_{f_{m+1}(x)}$
  - Eine Linearkombination von beliebigen linearen und nicht-linearen Funktionen:
  $f(x) = \lambda_1 \cdot \underbrace{\sin(x)}_{f_1(x)} + \lambda_2 \cdot \underbrace{\cos(x)}_{f_2(x)} + \lambda_3 \cdot \underbrace{x}_{f_3(x)} + \lambda_4 \cdot \underbrace{\exp(x)}_{f_4(x)} + \lambda_5 \cdot \underbrace{1}_{f_5(x)} + \ldots$

Problemstellung

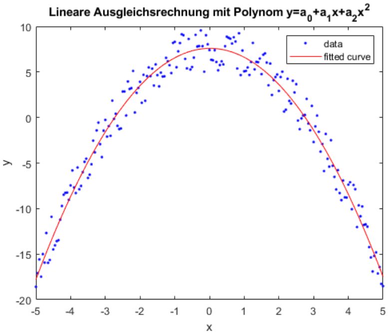
- Bei der nicht-linearen Ausgleichsrechnung lässt sich die gesuchte Funktion $f(x)$ nicht mehr als Linearkombination mit den gesuchten Parametern λ_i als Koeffizienten schreiben, diese treten "verwoben" mit der Funktionsgleichung auf:

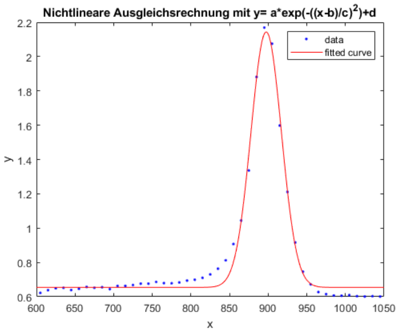
$$f = f(\lambda_1, \lambda_2, \ldots, \lambda_m, x)$$

Beispiele:

- Eine Gauss-Funktion: $f(x) = \lambda_1 e^{-\left(\frac{x-\lambda_2}{\lambda_3}\right)^2} + \lambda_4$
- Eine beliebige verkettete Funktion: $f(x) = \lambda_1 \cos(\lambda_2 x)$

Problemstellung





- Links: Ein Beispiel der linearen Ausgleichsrechnung. Eine Datenwolke aus 200 Datenpunkte wird angenähert durch ein Polynom $f(x) = \lambda_1 + \lambda_2 x + \lambda_3 x^2$.
- Rechts: Ein Beispiel der nicht-linearen Ausgleichsrechnung. Die rund 50 Datenpunkte werden angenähert durch $f(x) = f(x) = \lambda_1 e^{-\left(\frac{x-\lambda_2}{\lambda_3}\right)^2} + \lambda_4$. Natürlich könnnen die Parameter beliebig benannt werden (z.B. a statt λ_1 etc.).

Problemstellung: Definition Ausgleichproblem

Definition 6.2: Ausgleichproblem [1]

- Gegeben sind n Wertepaare (x_i, y_i), $i = 1, \ldots, n$ mit $x_i \neq x_j$ für $i \neq j$.
- Gesucht ist eine stetige Funktion $f : \mathbb{R} \longrightarrow \mathbb{R}$, die die Wertepaare in einem gewissen Sinn bestmöglich annähert, d.h. dass möglichst genau gilt

$$f(x_i) \approx y_i$$

für alle $i = 1, \ldots, n$.

Problemstellung: Definition Fehlerfunktional

Definition 6.3: Ansatzfunktionen / Ausgleichsfunktion / Fehlerfunktional / kleinste Fehlerquadrate [1]

- Gegeben sei eine Menge F von stetigen **Ansatzfuntkionen** f auf dem Intervall $[a,b]$ sowie n Wertepaare (x_i, y_i), $i = 1, ..., n$.
- Ein Element $f \in F$ heisst **Ausgleichsfunktion** von F zu den gegebenen Wertepaaren, falls das **Fehlerfunktional**

$$E(f) := \| \mathbf{y} - f(\mathbf{x}) \|_2^2 = \sum_{i=1}^{n} (y_i - f(x_i))^2$$

für f minimal wird, d.h. $E(f) = \min\{E(g) \mid g \in F\}$.

- Man nennt das so gefundene f dann optimal im Sinne der **kleinsten Fehlerquadrate** (least squares fit).

Problemstellung: Bemerkungen

- Diese Forderung der kleinsten Fehlerquardrate bedeutet nichts anderes, als dass das Quadrat der 2-Norm (vgl. Kap. 4.6.1) des Fehlervektors
$$\begin{pmatrix} y_1 - f(x_1) \\ \vdots \\ y_n - f(x_n) \end{pmatrix}$$
minimal sein soll. Andere Normen sind auch möglich, was zu anderen Ergebnissen führen kann. Die in der Praxis am häufigsten verwendete Norm ist allerdings die 2-Norm.

Problemstellung: Bemerkungen

- Es ist möglich, die Fehler noch mit Gewichten $w_i > 0$ zu versehen, d.h. man minimiert

$$\sum_{i=1}^{n} w_i \cdot (y_i - f(x_i))^2 .$$

- Wenn z.B. gewisse Datenpunkte (x_i, y_i) grosse Messungenauigkeiten aufweisen, können diese mit einem kleineren Gewicht versehen werden, als Datenpunkte mit höherer Messgenauigkeit. Die Summe der Gewichte sollte dabei normiert sein, also $\sum_{i=1}^{n} w_i = 1$.

Problemstellung: Bemerkungen

- Wählt man F als Menge aller Geraden, dann nennt man das so gefundene f *Ausgleichsgerade* oder in statistischen Zusammenhängen auch *Regressionsgerade*.

6.3.2 Lineare Ausgleichsprobleme

Lineare Ausgleichprobleme

- Beginnen wir mit dem einfachen Beispiel der linearen Regression.
- Ausgehend von der Wertetabelle

x_i	x_1	x_2	$\dots$	x_n
y_i	y_1	y_2	$\dots$	y_n

  suchen die Ausgleichsgerade der Form $f(x) = ax + b$, also $F := \{a_1 f_1 + a_2 f_2 \mid a_1, a_2 \in \mathbb{R}\}$ mit den Basisfunktionen $f_1(x) = x$ und $f_2(x) = 1$.
- Das Fehlerfunktional hat dann die Form

$$E(f)(a,b) := E(f) = \sum_{i=1}^{n} (y_i - f(x_i))^2 = \sum_{i=1}^{n} (y_i - (ax_i + b))^2 .$$

Lineare Ausgleichprobleme

- Dieses soll minimal werden, d.h. die partiellen Ableitung nach den Parametern a und b müssen verschwinden (in Analogie zum eindimensionalen Fall):

$$\begin{aligned}
0 &\overset{!}{=} \frac{\partial E}{\partial a} = \sum_{i=1}^{n} \frac{\partial}{\partial a}(y_i - (ax_i + b))^2 \\
&= \sum_{i=1}^{n} 2 \cdot (y_i - (ax_i + b)) \cdot (-x_i) \\
&= -2 \sum_{i=1}^{n} (y_i - (ax_i + b)) \cdot (x_i) \\
0 &\overset{!}{=} \frac{\partial E}{\partial b} = \sum_{i=1}^{n} \frac{\partial}{\partial b}(y_i - (ax_i + b))^2 \\
&= \sum_{i=1}^{n} 2 \cdot (y_i - (ax_i + b)) \cdot (-1) \\
&= -2 \sum_{i=1}^{n} (y_i - (ax_i + b))
\end{aligned}$$

Lineare Ausgleichprobleme

- Wir haben also zwei Gleichungen für die Unbekannten a, b. Durch Umformung erhalten wir

$$
\begin{aligned}
0 &\overset{!}{=} \sum_{i=1}^{n} (y_i - (ax_i + b)) \cdot (x_i) = \sum_{i=1}^{n} \left(x_i y_i - a x_i^2 - b x_i\right) \\
&= \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} a x_i^2 - \sum_{i=1}^{n} b x_i = \sum_{i=1}^{n} x_i y_i - a \sum_{i=1}^{n} x_i^2 - b \sum_{i=1}^{n} x_i \\
0 &\overset{!}{=} \sum_{i=1}^{n} (y_i - (ax_i + b)) = \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a x_i - \sum_{i=1}^{n} b \\
&= \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} a x_i - \sum_{i=1}^{n} b = \sum_{i=1}^{n} y_i - a \sum_{i=1}^{n} x_i - b \sum_{i=1}^{n} 1
\end{aligned}
$$

- Daraus folgt

$$\begin{aligned} a\sum_{i=1}^{n} x_i^2 + b\sum_{i=1}^{n} x_i &= \sum_{i=1}^{n} x_i y_i \\ a\sum_{i=1}^{n} x_i + b\underbrace{\sum_{i=1}^{n} 1}_{=n} &= \sum_{i=1}^{n} y_i \end{aligned}$$

oder in Matrixschreibweise erhalten wir das lineare Gleichungssystem für die Unbekannten (a, b)

$$\begin{pmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & n \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{pmatrix}$$

und können dieses nach a, b auflösen.

Beispiel 6.6

- Wir bestimmen die Ausgleichsgerade $f(x) = ax + b$ für die folgende Wertetabelle:

x_i	1	2	3	4
y_i	6	6.8	10	10.5

Beispiel 6.6: Lösung

- Mit $n = 4$ und

$$\sum_{i=1}^{4} x_i^2 = 30, \ \sum_{i=1}^{4} x_i = 10, \ \sum_{i=1}^{4} x_i y_i = 91.6, \ \sum_{i=1}^{4} y_i = 33.3$$

erhalten wir das lineare Gleichungssystem

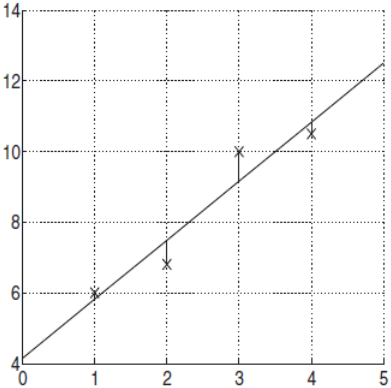
$$\begin{pmatrix} 30 & 10 \\ 10 & 4 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 91.6 \\ 33.3 \end{pmatrix}$$

mit der Lösung

$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 1.67 \\ 4.15 \end{pmatrix}.$$

Also ist die gesuchte Ausgleichsgerade $f(x) = 1.67x + 4.15$ (siehe folgende Abb.).

Beispiel 6.6: Lösung



- Ausgleichsgerade $f(x) = 1.67x + 4.15$ zu Bsp. 6.6 (aus [1]).

Lineare Ausgleichprobleme: Definition 6.4

Definition 6.4: Lineares Ausgleichproblem [1]

- Gegeben seien n Wertepaare $(x_i, y_i), i = 1, ..., n$, und m Basisfunktionen $f_1, ..., f_m$ auf einem Intervall $[a, b]$. Wir wählen F als die Menge der Ansatzfunktionen $f := \lambda_1 f_1 + ... + \lambda_m f_m$ mit $\lambda_j \in \mathbb{R}$, also $F = \{f = \lambda_1 f_1 + ... + \lambda_m f_m \mid \lambda_j \in \mathbb{R}, j = 1, ..., m\}$.
- Es liegt dann ein **lineares Ausgleichproblem** vor mit dem Fehlerfunktional

$$
\begin{aligned}
E(f) &= \| \boldsymbol{y} - f(\boldsymbol{x}) \|_2^2 = \sum_{i=1}^{n} (y_i - f(x_i))^2 \\
&= \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{m} \lambda_j f_j(x_i) \right)^2 = \| \boldsymbol{y} - \boldsymbol{A}\lambda \|_2^2
\end{aligned}
$$

$$
\boldsymbol{A} = \begin{pmatrix} f_1(x_1) & f_2(x_1) & \cdots & f_m(x_1) \\ f_1(x_2) & f_2(x_2) & \cdots & f_m(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ f_1(x_n) & f_2(x_n) & \cdots & f_m(x_n) \end{pmatrix}, \quad \boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \quad \lambda = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_m \end{pmatrix}
$$

- Das System $\boldsymbol{A}\lambda = \boldsymbol{y}$ heisst **Fehlergleichungssystem**.

Lineare Ausgleichprobleme: Bemerkungen

- Das Fehlergleichungssystem besitzt n Gleichungen und m Unbekannte.
- In der Regel ist $n > m$, wir haben also mehr Gleichungen als Unbekannte, d.h. das Gleichungssystem ist überbestimmt.
- Man kann daher nicht damit rechnen, dass es eine Lösung gibt (bzw. dass das Fehlerfunktional $E(f) = 0$ erreicht werden kann).

Lineare Ausgleichprobleme: Bemerkungen

- Für $m = n$ haben wir eine eindeutige Lösung und $E(f) = 0$.
- Dies ist gleichbedeutend damit, dass die Ausgleichfunktion f exakt durch sämtliche Wertepaare geht.
- Dies ist der Spezialfall der Interpolation.

Lineare Ausgleichprobleme: Normalgleichungen

- Wir können nun versuchen, das Fehlerfunktional zu minimieren. Dafür müssen sämtliche partiellen Ableitungen von $E(f)$ nach den unbekannten Koeffizienten λ_j verschwinden, also

$$\frac{\partial E(f)}{\partial \lambda_j} = 0.$$

- Dies führt uns zu den Begriffen der Normalgleichungen und des Normalgleichungssystem.

Lineare Ausgleichprobleme: Normalgleichungen

Definition 6.5: Normalgleichungen / Normalgleichungssystem [1]

- Die Gleichungen

$$\frac{\partial E(f)(\lambda_1,\ldots,\lambda_m)}{\partial \lambda_j} = 0, \quad j = 1,\ldots,m$$

  heissen **Normalgleichungen** des linearen Ausgleichproblems.
- Das System sämtlicher Normalgleichungen heisst **Normalgleichungssystem** und lässt sich als lineares Gleichungssystem schreiben

$$\boldsymbol{A}^T\boldsymbol{A}\lambda = \boldsymbol{A}^T\boldsymbol{y}$$

- Die Lösung $\lambda = (\lambda_1,\ldots,\lambda_m)^T$ des Normalgleichungssystems beinhaltet die gesuchten Parameter des linearen Ausgleichproblems.

Lineare Ausgleichprobleme: Normalgleichungen

Bemerkungen:

- Das lineare Normalgleichungssystem lässt sich mit den Verfahren aus Kap. 4 lösen, im einfachsten Fall mit dem Gauss-Algorithmus bzw. der LR-Zerlegung.
- Die $m \times m$ Matrix $\boldsymbol{A}^T\boldsymbol{A}$ ist allerdings i.d.R. schlecht konditioniert, d.h. kleine Fehler in den Datenpunkten $\boldsymbol{y}$ können grosse Auswirkungen auf die Lösung λ haben.
- Das Normalgleichungssystem lässt sich stabiler mithilfe der QR-Zerlegung der Matrix $\boldsymbol{A}$ lösen (siehe Kap. 4.5.2). Zur Erinnerung: es gilt $\boldsymbol{A} = \boldsymbol{QR}$ mit einer Orthogonalmatrix $\boldsymbol{Q}$, wobei $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}_n$ ($\boldsymbol{I}_n$ ist die $n \times n$ Einheitsmatrix) resp. $\boldsymbol{Q}^{-1} = \boldsymbol{Q}^T$, $\boldsymbol{R}$ ist eine rechtsobere Dreiecksmatrix.

Lineare Ausgleichprobleme: Normalgleichungen

- Aus
$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$$
folgt für das Normalgleichungssystem
$$\boldsymbol{A}^T\boldsymbol{A}\lambda = \boldsymbol{A}^T\boldsymbol{y}$$
das äquivalente, häufig aber besser konditionierte Gleichungssystem
$$\boldsymbol{R}\lambda = \boldsymbol{Q}^T\boldsymbol{y}$$
da
$$\begin{aligned} \boldsymbol{A}^T\boldsymbol{A}\lambda &= \boldsymbol{A}^T\boldsymbol{y} \\ (\boldsymbol{Q}\boldsymbol{R})^T(\boldsymbol{Q}\boldsymbol{R})\lambda &= (\boldsymbol{Q}\boldsymbol{R})^T\boldsymbol{y} \\ \boldsymbol{R}^T\underbrace{\boldsymbol{Q}^T\boldsymbol{Q}}_{\boldsymbol{I}_n}\boldsymbol{R}\lambda &= \boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{y} \\ \boldsymbol{R}^T\boldsymbol{R}\lambda &= \boldsymbol{R}^T\boldsymbol{Q}^T\boldsymbol{y} \\ \Rightarrow \boldsymbol{R}\lambda &= \boldsymbol{Q}^T\boldsymbol{y} \end{aligned}$$

Beispiel 6.7

- Stellen Sie für das lineare Ausgleichproblem in Bsp. 6.6 das Normalgleichungssystem auf.

Beispiel 6.7: Lösung

- Mit $f_1(x) = x$ und $f_2(x) = 1$ erhalten wir

$$\boldsymbol{A} = \begin{pmatrix} f_1(x_1) & f_2(x_1) \\ f_1(x_2) & f_2(x_2) \\ f_1(x_3) & f_2(x_3) \\ f_1(x_4) & f_2(x_4) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 4 & 1 \end{pmatrix}$$

$$\Rightarrow \boldsymbol{A}^T\boldsymbol{A} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 2 & 1 \\ 3 & 1 \\ 4 & 1 \end{pmatrix} = \begin{pmatrix} 30 & 10 \\ 10 & 4 \end{pmatrix}$$

$$\boldsymbol{A}^T\boldsymbol{y} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} 6 \\ 6.8 \\ 10 \\ 10.5 \end{pmatrix} = \begin{pmatrix} 91.6 \\ 33.3 \end{pmatrix}$$

Damit entspricht $\boldsymbol{A}^T\boldsymbol{A}\lambda = \boldsymbol{A}^T\boldsymbol{y}$ aus Bsp. 9.1.

Aufgabe 6.5

- Schreiben Sie ein Python-Skript, welches Beispiel 6.7 mittels der $QR-$Zerlegung löst und die Daten sowie die Ausgleichsgerade plottet.

Aufgabe 6.5: Lösung

Aufgabe 6.5: Lösung

Beispiel 6.8

- Bestimmen Sie mit den bisher behandelten Techniken eine Funktion der Form $f(x) = ae^{bx}$, die die folgenden Daten möglichst gut approximiert:

x_i	0	1	2	3	4
y_i	3	1	0.5	0.2	0.05

Beispiel 6.8: Lösung

- Auf den ersten Blick liegt hier kein lineares Ausgleichproblem vor, da wir $f(x)$ nicht als Summe $af_1(x) + bf_2(x)$ schreiben können.
- Wir können aber durch beidseitiges Logarithmieren einen linearen Ansatz erreichen:

$$\ln f(x) = \ln\left(ae^{bx}\right) = \ln(a) + b \cdot \ln\left(e^x\right) = \ln(a) \cdot \underbrace{1}_{f_1(x)} + b \cdot \underbrace{x}_{f_2(x)}$$

- Wir werden durch die Lösung des Normalgleichungssystems also $\ln(a)$ und b berechnen können, wenn wir die y_i logarithmieren.

Beispiel 6.8: Lösung

$$\boldsymbol{A}^T\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{pmatrix} = \begin{pmatrix} 5 & 10 \\ 10 & 30 \end{pmatrix}$$

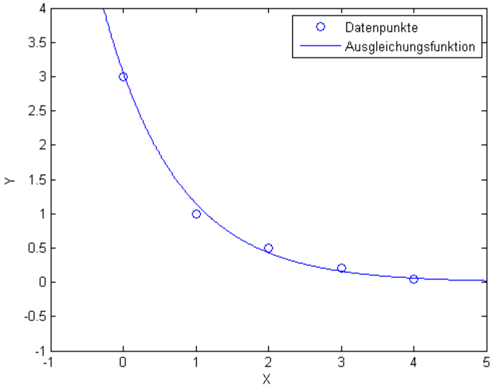
$$\boldsymbol{A}^T \cdot \widetilde{\boldsymbol{y}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 & 4 \end{pmatrix} \begin{pmatrix} \ln 3 \\ \ln 1 \\ \ln 0.5 \\ \ln 0.2 \\ \ln 0.05 \end{pmatrix} = \begin{pmatrix} -4.1997 \\ -18.1975 \end{pmatrix}$$

$$\boldsymbol{A}^T\boldsymbol{A}\lambda = \boldsymbol{A}^T\widetilde{\boldsymbol{y}} \Rightarrow \lambda = \begin{pmatrix} \ln a \\ b \end{pmatrix} = \begin{pmatrix} 1.11968... \\ -0.97981... \end{pmatrix}$$

$$\Rightarrow f(x) = \mathrm{e}^{\ln a} \cdot \mathbf{e}^{bx} = 3.06388... \cdot \mathbf{e}^{-0.97981...x}$$

Wir werden dieses Beispiel im nächsten Abschnitt bei nichtlinearen Ausgelichsproblemen gleich nochmals aufgreifen.

Beispiel 6.8: Lösung



- Ausgleichsfunktion $f(x) = 3.0638 \cdot e^{-0.9798x}$ zu Bsp. 6.8.

Aufgabe 6.6

- Bestimmen Sie für die folgenden Daten die Funktion der Form $f(x) = ae^x + b$, die diese Daten bestmöglich im Sinner der kleinsten Fehlerquadrate approximiert:

x_i	0	1	2	3	4
y_i	6	12	30	80	140

Aufgabe 6.6: Lösung

Aufgabe 6.6: Lösung

Aufgabe 6.6: Lösung

Aufgabe 6.6: Lösung

Aufgabe 6.6: Lösung

6.3.3 Nichtlineare Ausgleichprobleme

Nichtlineare Ausgleichprobleme

- Wir haben in Beispiel 6.8 gesehen, dass man allenfalls in Spezialfällen ein nichtlineares Ausgleichsproblem in ein lineares Ausgleichproblem umformen kann.
- Trotzdem wollen wir natürlich in der Lage sein, allgemeine nichtlineare Ausgleichprobleme zu lösen.

Nichtlineare Ausgleichprobleme

Definition 9.5: Allgemeines Ausgleichproblem [1]

- Gegeben seien n Wertepaare $(x_i, y_i), i = 1, ..., n$, und die Menge F der Ansatzfunktionen $f_p = f_p(\lambda_1, \lambda_2, ..., \lambda_m, x)$ mit m Parametern $\lambda_j \in \mathbb{R}$, $j = 1, ..., m$, also $F = \{f_p(\lambda_1, \lambda_2, ..., \lambda_m, x) \mid \lambda_j \in \mathbb{R}, j = 1, ..., m\}$.
- Das **allgemeine Ausgleichproblem** besteht darin, die m Parameter $\lambda_1, ..., \lambda_m$ zu bestimmen, so dass das Fehlerfunktional E

$$
\begin{aligned}
E(f) &= \sum_{i=1}^{n} \left(y_i - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_i)\right)^2 = \left\| \begin{pmatrix} y_1 - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_1) \\ y_2 - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_2) \\ \vdots \\ y_n - f_p(\lambda_1, \lambda_2, ..., \lambda_m, x_n) \end{pmatrix} \right\|_2^2 \\
&= \| \boldsymbol{y} - \boldsymbol{f}(\lambda) \|_2^2
\end{aligned}
$$

minimal wird unter allen zulässigen Parameterbelegungen.

Nichtlineare Ausgleichprobleme

Definition 9.5: Fortsetzung

- Dabei ist

$$
\begin{aligned}
\boldsymbol{f}(\lambda) &= \boldsymbol{f}(\lambda_1, \lambda_2, \dots, \lambda_m) := \begin{pmatrix} f_1(\lambda_1, \lambda_2, \dots, \lambda_m) \\ f_2(\lambda_1, \lambda_2, \dots, \lambda_m) \\ \vdots \\ f_n(\lambda_1, \lambda_2, \dots, \lambda_m) \end{pmatrix} \\
&= \begin{pmatrix} f_p(\lambda_1, \lambda_2, \dots, \lambda_m, x_1) \\ f_p(\lambda_1, \lambda_2, \dots, \lambda_m, x_2) \\ \vdots \\ f_p(\lambda_1, \lambda_2, \dots, \lambda_m, x_n) \end{pmatrix} \\
\boldsymbol{y} &= \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{\color{red}n} \end{pmatrix}, \ \lambda = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_{\color{red}m} \end{pmatrix}
\end{aligned}
$$

Nichtlineare Ausgleichprobleme: Bemerkungen

- Falls die Ansatzfunktionen f_p linear in den Parametern sind, haben wir den Spezialfall des linearen Ausgleichsproblems mit $\boldsymbol{f}(\lambda) = \boldsymbol{A}\lambda$ (vgl. Def. 6.4).

Nichtlineare Ausgleichprobleme: Bemerkungen

- Das allg. Ausgleichproblem ist also äquivalent zur Bestimmung des Minimums einer Funktion

$$E : \mathbb{R}^m \longrightarrow \mathbb{R}$$

  und wir könnten wieder die Normalgleichungen aufstellen , indem wir die partiellen Ableitungen von f nach den Parametern λ_i gleich Null setzen und das entstehende, diesmal nicht lineare Gleichungssystem lösen.
- Wir werden im folgenden Beispiel dies einmal ansatzweise machen und werden dann aber mit Gauss-Newton-Verfahren im nächsten Abschnitt ein anderes, effektiveres Verfahren kennen lernen.

Beispiel 6.9

- Wir nehmen wieder das Problem aus Beispiel 6.8, d.h. wir wollen eine Ansatzfunktion $f(x) = ae^{bx}$ bestmöglich im Sinn der kleinsten Fehlerquadrate an die gegebenen Daten anpassen:

x_i	0	1	2	3	4
y_i	3	1	0.5	0.2	0.05

Beispiel 6.9: Lösung

- Wir haben also zwei Parameter a und b in der Ansatzfunktion $f_p(a,b,x) = ae^{bx}$ so zu bestimmen, dass

$$E(f) = \sum_{i=1}^{5} (y_i - f_p(a,b,x_i))^2 = \sum_{i=1}^{5} (y_i - ae^{bx_i})^2$$

minimal wird.

Beispiel 6.9: Lösung

- Nullsetzen der partiellen Ableitungen liefert

$$\begin{aligned}
0 &= \frac{\partial E(f)}{\partial a} = \sum_{i=1}^{5} \frac{\partial}{\partial a}(y_i - a\mathrm{e}^{bx_i})^2 \\
&= \sum_{i=1}^{5} 2 \cdot (y_i - a\mathrm{e}^{bx_i}) \cdot (-\mathrm{e}^{bx_i}) \\
&= -2 \sum_{i=1}^{5} (y_i - a\mathrm{e}^{bx_i}) \cdot \mathrm{e}^{bx_i} \\
0 &= \frac{\partial E(f)}{\partial b} = \sum_{i=1}^{5} \frac{\partial}{\partial b}(y_i - a\mathrm{e}^{bx_i})^2 \\
&= \sum_{i=1}^{5} 2 \cdot (y_i - a\mathrm{e}^{bx_i}) \cdot (-a\mathrm{e}^{bx_i}) \cdot x_i \\
&= -2 \sum_{i=1}^{5} (y_i - a\mathrm{e}^{bx_i}) \cdot a\mathrm{e}^{bx_i} \cdot x_i
\end{aligned}$$

Beispiel 6.9: Lösung

- Dies sind zwei nichtlineare Gleichungen, die mit dem Newton-Verfahren für Systeme (vlg. Kap. 5) gelöst werden können.
- Wegen der Länge der Ausdrücke verzichten wir hier an dieser Stelle auf die Details , ein funktionierendes Python Skript ist unten angegeben.
- Für die Startwerte $a_0 = 3$ und $b_0 = -1$ und einer Fehlerschranke von 10^{-5} liefert es nach zweimaliger Iteration

$$a = 2.98165..., \; b = -1.00328...$$

- Zum Vergleich: in Bsp. 6.8 hatten wir durch Logarithmieren das nichtlineare Ausgleichsproblem auf ein lineares zurückgeführt und erhielten die Parameter

$$a = 3.06388..., \; b = -0.97981...$$

Beispiel 6.9: Lösung

- Der Unterschied wird nicht durch Rechenungenauigkeiten verursacht, tatsächlich handelt es sich um zwei verschiedene Modellanpasungen.

Beispiel 6.9: Lösung


```
#Skript zur Lösung des nichtlinearen Gleichungssysstems
import numpy as np
import sympy as sp
a, b = sp.symbols('a b')
x = np.array([0,1,2,3,4])
y = np.array([3, 1, 0.5, 0.2, 0.05])
f1 = 0;
for i in range(0,5):
    f1 = f1+(y[i]-a*sp.exp(b*x[i]))*sp.exp(b*x[i])
f1 = -2*f1
f2 = 0;
for i in range(0,5):
    f2 = f2+(y[i]-a*sp.exp(b*x[i]))*a*sp.exp(b*x[i])*x[i]
f2 = -2*f2
f = sp.Matrix([f1,f2])
lam = sp.Matrix([a,b]) # Achtung: die Unbekannten sind a und b (nicht x)
Df = f.jacobian(lam)
f = sp.lambdify([[[a],[b]]], f, "numpy")
Df = sp.lambdify([[[a],[b]]], Df, "numpy")
```

Beispiel 6.9: Lösung


```
# Newtons Methode für Systeme
def newton(f, Df, x0, tol):
   n=0
   x=np.copy(x0)
   err = np.linalg.norm(f(x),2)
   while err > tol:
      delta = np.linalg.solve(Df(x),-f(x))
      x = x+delta
      err = np.linalg.norm(f(x),2)
      n = n+1
   return(x,n)

# Aufruf
tol = 1e-5
x0 = np.array([[3,-1]]).T
[xn,n] = newton(f, Df, x0, tol)
print('x_' + str(n) + ' = ' + str(np.reshape(xn,(2))))
```

Beispiel 6.9: Lösung

- Das obige Verfahren führt zwar zu einem Ergebnis, hat jedoch mehrere Nachteile.
- Die partiellen Ableitungen des Fehlerfunktionals müsen explizit berechnet werden, und es konvergiert nur, wenn man genügend gute Anfangsnäherungen für die Parameter zur Verfügung hat, was häufig nicht der Fall ist.
- So würden wir mit einem Startwert $x_0 = [1, -1]$ im obigen Skript das Resultat $a = 3.00000\ldots$, $b = -13.60235\ldots$ erhalten.
- Das folgende iterative Verfahren hat diesbezüglich einige Vorteile.

6.3.4 Gauss- Newton-Verfahren

Gauss-Newton-Verfahren

- Wir defnieren:

Definition 6.7: Quadratmittelproblem [1]

- Gegeben ist eine Funktion $\boldsymbol{g} : \mathbb{R}^m \longrightarrow \mathbb{R}^n$ und das zugehörige Fehlerfunktional $E : \mathbb{R}^m \longrightarrow \mathbb{R}$, definiert durch $E(\boldsymbol{x}) := \| \boldsymbol{g}(\boldsymbol{x}) \|_2^2$. Das Problem, einen Vektor $\boldsymbol{x} \in \mathbb{R}^m$ zu finden, für den $E(\boldsymbol{x})$ minimal wird, nennt man Quadratmittelproblem.

Gauss-Newton-Verfahren

- Nichtlineare Ausgleichsprobleme sind also Quadratmittelprobleme, wobei wir $\boldsymbol{g}(\mathrm{x})$ ersetzen mit $\boldsymbol{g}(\lambda)$:

$$\boldsymbol{g}(\lambda) := \boldsymbol{y} - \boldsymbol{f}(\lambda), \ \mathrm{y} \in \mathbb{R}^n, \lambda \in \mathbb{R}^m$$

wie in Def. 6.6.

- Das Fehlerfunktional lautet entsprechend

$$E(\lambda) := \| \boldsymbol{g}(\lambda) \|_2^2 = \| \boldsymbol{y} - \boldsymbol{f}(\lambda) \|_2^2$$

und der Vektor $\lambda = (\lambda_1, ..., \lambda_m)^T$ beinhaltet die gesuchten Parameter.

Gauss-Newton-Verfahren

- Das folgende Gauss-Newton-Verfahren besteht aus einer Kombination von linearer Ausgleichrechnung und dem Newton-Verfahren.
- Wir ersetzen dabei in $\| \boldsymbol{g}(\lambda) \|_2^2$ den Vektor $\boldsymbol{g}(\lambda)$ durch die Linearkombination

$$\boldsymbol{g}(\lambda) \approx \boldsymbol{g}(\lambda^{(0)}) + \boldsymbol{Dg}(\lambda^{(0)}) \cdot (\lambda - \lambda^{(0)})$$

Dies entspricht der 'verallgemeinerten Tangentengleichung' aus Def. 5.3 mit der $n \times m$ Jacobi-Matrix $\boldsymbol{Dg}(\lambda^{(0)})$ in der Umgebung eines Startwertes $\lambda^{(0)}$:

$$\boldsymbol{Dg}(\lambda^{(0)}) := \begin{pmatrix} \frac{\partial g_1}{\partial \lambda_1}(\lambda^{(0)}) & \frac{\partial g_1}{\partial \lambda_2}(\lambda^{(0)}) & \cdots & \frac{\partial g_1}{\partial \lambda_m}(\lambda^{(0)}) \\ \frac{\partial g_2}{\partial \lambda_1}(\lambda^{(0)}) & \frac{\partial g_2}{\partial \lambda_2}(\lambda^{(0)}) & \cdots & \frac{\partial g_2}{\partial \lambda_m}(\lambda^{(0)}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial g_n}{\partial \lambda_1}(\lambda^{(0)}) & \frac{\partial g_n}{\partial \lambda_2}(\lambda^{(0)}) & \cdots & \frac{\partial g_n}{\partial \lambda_m}(\lambda^{(0)}) \end{pmatrix}$$

Gauss-Newton-Verfahren

- Das heisst, wir suchen statt des Minimums des ursprünglichen Fehlerfunktionals

$$E(\lambda) \quad = \quad \| \, \boldsymbol{g}(\lambda) \, \|_2^2$$

nun das Minimum des 'linearisierten Fehlerfunktionals'

$$\tilde{E}(\lambda) \quad = \quad \| \, \underbrace{\boldsymbol{g}(\lambda^{(0)})}_{\tilde{\boldsymbol{y}}} + \underbrace{\boldsymbol{Dg}(\lambda^{(0)})}_{-\tilde{\boldsymbol{A}}} \cdot \underbrace{(\lambda - \lambda^{(0)})}_{\delta} \, \|_2^2$$

und haben so das nichtlineare Ausgleichproblem zurückgeführt auf ein lineares Ausgleichproblem.

Gauss-Newton-Verfahren

- Mit den Subsitutionen

$$\begin{aligned} \widetilde{\boldsymbol{y}} &:= \boldsymbol{g}(\lambda^{(0)}) \\ \widetilde{\boldsymbol{A}} &:= -\boldsymbol{Dg}(\lambda^{(0)}) \\ \delta &:= \lambda - \lambda^{(0)} \end{aligned}$$

haben wir die gleiche Form wie in Def. 6.4

$$\tilde{E}(\lambda) = \| \widetilde{\boldsymbol{y}} - \widetilde{A}\delta \|_2^2$$

und die Lösung berechnet sich gemäss Def. 6.5

$$\widetilde{\boldsymbol{A}}^T \widetilde{\boldsymbol{A}} \delta = \widetilde{\boldsymbol{A}}^T \widetilde{\boldsymbol{y}}$$

bzw.

$$\boldsymbol{Dg}(\lambda^{(0)})^T \boldsymbol{Dg}(\lambda^{(0)}) \delta = -\boldsymbol{Dg}(\lambda^{(0)})^T \cdot \boldsymbol{g}(\lambda^{(0)}).$$

Wegen der besseren Konditionierung lösen wir das Normalgleichungssystem nicht mit dem Gauss-Algorithmus sondern benutzen wieder die $QR-$Zerlegung, diesmal von $\boldsymbol{Dg}(\lambda^{(0)}) = \boldsymbol{QR}$, und lösen das äquivalente Gleichungssystem

$$\boldsymbol{R}\delta = -\boldsymbol{Q}^T \boldsymbol{g}(\lambda^{(0)}).$$

Gauss-Newton-Verfahren

- Die Lösung von
$$\boldsymbol{R}\delta = -\boldsymbol{Q}^T\boldsymbol{g}(\lambda^{(0)})$$
liefert uns dann
$$\delta = \lambda - \lambda^{(0)}$$
und damit können wir nach der eigentlich gesuchten Grösse auflösen:
$$\lambda = \lambda^{(0)} + \delta$$
- Die Grösse δ liefert uns in diesem Sinne eine Korrektur bzw. "Verbesserung" zum Startwert $\lambda^{(0)}$
- Unter der Annahme, dass λ eine bessere Näherung ist als der Startwert $\lambda^{(0)}$, können wir $\lambda^{(1)} := \lambda$ setzen und diesen Schritt durch rekursives Einsetzen wiederholen. Damit erhalten wir das Gauss-Newton Verfahren:

Gauss-Newton-Verfahren

Definition 6.8: Gauss-Newton-Verfahren

- Sei $\boldsymbol{\lambda}^{(0)}$ ein Startvektor in der Nähe des Minimums des Fehlerfunktionals E der Ausgleichsfunktion $\boldsymbol{f}(\boldsymbol{\lambda})$. Das Gauss-Newton-Verfahren zur näherungsweisen Bestimmung des Minimums lautet:
- Berechne die Funktion
$$\boldsymbol{g}(\boldsymbol{\lambda}) := \boldsymbol{y} - \boldsymbol{f}(\boldsymbol{\lambda})$$
sowie deren Jacobi-Matrix
$$\boldsymbol{Dg}(\boldsymbol{\lambda})$$
- Für $k = 0, 1, ...$:
  1. Berechne $\boldsymbol{\delta}^{(k)}$ als Lösung des linearen Ausgleichproblems
  $$\min \| \boldsymbol{g}(\boldsymbol{\lambda}^{(k)}) + \boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)}) \cdot \boldsymbol{\delta}^{(k)} \|_2^2,$$
  d.h. löse konkret das Normalgleichungssystem
  $$\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)})^T \boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)}) \boldsymbol{\delta}^{(k)} = -\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)})^T \cdot \boldsymbol{g}(\boldsymbol{\lambda}^{(k)})$$
  nach $\boldsymbol{\delta}^{(k)}$ auf. Dies wird am stabilsten mit der QR–Zerlegung von $\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)})$ erreicht:
     (a) Berechne die QR–Zerlegung
     $$\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)}) = \boldsymbol{Q}^{(k)} \boldsymbol{R}^{(k)}$$
     (b) Löse nach $\boldsymbol{\delta}^{(k)}$ auf:
     $$\boldsymbol{R}^{(k)} \boldsymbol{\delta}^{(k)} = -\boldsymbol{Q}^{(k)T} \boldsymbol{g}(\boldsymbol{\lambda}^{(k)})$$
  2. Setze
  $$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \boldsymbol{\delta}^{(k)}.$$

Gauss-Newton-Verfahren

- Die Grösse $\delta^{(k)}$ kann als 'Korrekturrichtung' der Näherung $\lambda^{(k)}$ aufgefasst werden.
- In der Praxis verwendet man besser das folgende gedämpfte Gauss-Newton-Verfahren.
- Analog zum gedämpften Newton-Verfahren aus Kapitel 5.4.3 akzeptieren wir die Korrekutr $\delta^{(k)}$ nur, wenn sie wirklich zu einer Abnahem des Fehlerfunktionals führt, also

$$E(\lambda^{(k+1)}) = \| \boldsymbol{g}(\lambda^{(k+1)}) \|_2^2 < \| \boldsymbol{g}(\lambda^{(k)}) \|_2^2 = E(\lambda^{(k)}).$$

erfüllt ist. Dies können wir durch die Dämpfung wie folgt anstreben:

Gedämpftes Gauss-Newton-Verfahren

Definition 6.9: Gedämpftes Gauss-Newton-Verfahren

- Sei $\boldsymbol{\lambda}^{(0)}$ ein Startvektor in der Nähe des Minimums von E der Ausgleichsfunktion $\boldsymbol{f}(\boldsymbol{\lambda})$. Das gedämpfte Gauss-Newton-Verfahren zur näherungsweisen Bestimmung des Minimums lautet:
- Berechne die Funktion
$$\boldsymbol{g}(\boldsymbol{\lambda}) := \boldsymbol{y} - \boldsymbol{f}(\boldsymbol{\lambda})$$
sowie deren Jacobi-Matrix
$$\boldsymbol{Dg}(\boldsymbol{\lambda})$$
- Für $k = 0, 1, ...$:
  1. Berechne $\boldsymbol{\delta}^{(k)}$ als Lösung des linearen Ausgleichproblems
  $$\min \| \boldsymbol{g}(\boldsymbol{\lambda}^{(k)}) + \boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)}) \cdot \boldsymbol{\delta}^{(k)} \|_2^2,$$
  d.h. löse konkret das Normalgleichungssystem
  $$\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)})^T \boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)}) \boldsymbol{\delta}^{(k)} = -\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)})^T \cdot \boldsymbol{g}(\boldsymbol{\lambda}^{(k)})$$
  nach $\boldsymbol{\delta}^{(k)}$ auf. Dies wird am stabilsten mit der QR–Zerlegung von $\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)})$ erreicht:
  (a) Berechne die QR–Zerlegung
  $$\boldsymbol{Dg}(\boldsymbol{\lambda}^{(k)}) = \boldsymbol{Q}^{(k)} \boldsymbol{R}^{(k)}$$
  (b) Löse nach $\boldsymbol{\delta}^{(k)}$ auf:
  $$\boldsymbol{R}^{(k)} \boldsymbol{\delta}^{(k)} = -\boldsymbol{Q}^{(k)T} \boldsymbol{g}(\boldsymbol{\lambda}^{(k)})$$
  2. Finde das minimale $p \in \{0, 1, ..., p_{max}\}$ mit
  $$\| \boldsymbol{g}\underbrace{\left(\boldsymbol{\lambda}^{(k)} + \frac{\boldsymbol{\delta}^{(k)}}{2^p}\right)}_{\boldsymbol{\lambda}^{(k+1)}} \|_2^2 < \| \boldsymbol{g}\left(\boldsymbol{\lambda}^{(k)}\right) \|_2^2$$
  3. Falls kein minimales p gefunden werden kann, rechne mit $p = 0$ weiter
  4. Setze
  $$\boldsymbol{\lambda}^{(k+1)} = \boldsymbol{\lambda}^{(k)} + \frac{\boldsymbol{\delta}^{(k)}}{2^p}.$$

Gedämpftes Gauss-Newton-Verfahren

Bemerkungen:

- Ein Vorteil des gedämpften Gauss-Newton-Verfahrens ist, dass es i.d.R. für einen grösseren Bereich von Startvektoren konvergiert als das ungedämpfte Gauss-Newton-Verfahren.
- Durch das fortlaufende Anpassen der 'Korrekturrichtung' kann es sozusagen noch mit Startvektoren umgehen, die weiter entfernt sind vom Optimum.
- Die Dämpfung ist aber keine Garantie für Konvergenz. Man benötigt noch weitere Bedingungen, auf die wir hier nicht eingehen werden.

Gedämpftes Gauss-Newton-Verfahren

Bemerkungen:

- Als Abbruchkriterium des Algorithmus kann z.B.

$$\| \frac{\delta^{(k)}}{2^p} \|_2 < TOL$$

verwendet werden. Wiederum ist das aber keine Garantie, dass die berechnete Näherung einen maximalen Abstand von TOL zum gesuchten Minimum besitzt.

Beispiel 6.10

- Wir nehmen nochmals das Problem aus Beispiel 6.8.
- Wir wollen also eine Ansatzfunktion $f(x) = ae^{bx}$ bestmöglich im Sinn der kleinsten Fehlerquadrate an die gegebenen Daten anpassen und vergleichen die Resultate des ungedämpften und des gedämpften Gauss-Newton-Verfahrens, einmal für den Startvektor $\lambda^{(0)} = (a^{(0)}, b^{(0)})^T = (1, -1.5)^T$ und dann nochmals für den Startvektor $\lambda^{(0)} = (a^{(0)}, b^{(0)})^T = (2, 2)^T$

x_i	0	1	2	3	4
y_i	3	1	0.5	0.2	0.05

Beispiel 6.10: Lösung

$$\boldsymbol{g}(a,b) := \boldsymbol{y} - \boldsymbol{f}(a,b) = \begin{pmatrix} g_1(a,b) \\ \vdots \\ g_5(a,b) \end{pmatrix} = \begin{pmatrix} y_1 - a\mathrm{e}^{bx_1} \\ \vdots \\ y_5 - a\mathrm{e}^{bx_5} \end{pmatrix} = \begin{pmatrix} 3 - a\mathrm{e}^{b\cdot 0} \\ 1 - a\mathrm{e}^{b\cdot 1} \\ 0.5 - a\mathrm{e}^{b\cdot 2} \\ 0.2 - a\mathrm{e}^{b\cdot 3} \\ 0.05 - a\mathrm{e}^{b\cdot 4} \end{pmatrix}$$

$$\boldsymbol{Dg}(a,b) := \begin{pmatrix} \frac{\partial g_1}{\partial a}(a,b) & \frac{\partial g_1}{\partial b}(a,b) \\ \frac{\partial g_2}{\partial a}(a,b) & \frac{\partial g_2}{\partial b}(a,b) \\ \vdots & \vdots \\ \frac{\partial g_5}{\partial a}(a,b) & \frac{\partial g_5}{\partial b}(a,b) \end{pmatrix} = \begin{pmatrix} -1 & 0 \\ -\mathrm{e}^{b\cdot 1} & -a\mathrm{e}^{b\cdot 1} \\ -\mathrm{e}^{b\cdot 2} & -2a\mathrm{e}^{b\cdot 2} \\ -\mathrm{e}^{b\cdot 3} & -3a\mathrm{e}^{b\cdot 3} \\ -\mathrm{e}^{b\cdot 4} & -4a\mathrm{e}^{b\cdot 4} \end{pmatrix}$$

Beispiel 6.10: Lösung

- Für den ersten Schritt des ungedämpften Verfahrens erhalten wir also

$$\boldsymbol{Dg}(1,-1.5) = \begin{pmatrix} -1 & 0 \\ -\mathbf{e}^{-1.5} & -\mathbf{e}^{-1.5} \\ -\mathbf{e}^{-3} & -2\mathbf{e}^{-3} \\ -\mathbf{e}^{-4.5} & -3\mathbf{e}^{-4.5} \\ -\mathbf{e}^{-6} & -4\mathbf{e}^{-6} \end{pmatrix}, \quad \boldsymbol{g}(1,-1.5) = \begin{pmatrix} 3-1 \\ 1-\mathbf{e}^{-1.5} \\ 0.5-\mathbf{e}^{-3} \\ 0.2-\mathbf{e}^{-4.5} \\ 0.05-\mathbf{e}^{-6} \end{pmatrix}$$

Beispiel 6.10: Lösung

- Für das lineare Gleichungssystem:

$$\begin{pmatrix} -1 & -e^{-1.5} & -e^{-3} & -e^{-4.5} & -e^{-6} \\ 0 & -e^{-1.5} & -2e^{-3} & -3e^{-4.5} & -4e^{-6} \end{pmatrix} \begin{pmatrix} -1 & 0 \\ -e^{-1.5} & -e^{-1.5} \\ -e^{-3} & -2e^{-3} \\ -e^{-4.5} & -3e^{-4.5} \\ -e^{-6} & -4e^{-6} \end{pmatrix} \begin{pmatrix} \delta_1^{(0)} \\ \delta_2^{(0)} \end{pmatrix}$$

$$= - \begin{pmatrix} -1 & -e^{-1.5} & -e^{-3} & -e^{-4.5} & -e^{-6} \\ 0 & -e^{-1.5} & -2e^{-3} & -3e^{-4.5} & -4e^{-6} \end{pmatrix} \cdot \begin{pmatrix} 3-1 \\ 1-e^{-1.5} \\ 0.5-e^{-3} \\ 0.2-e^{-4.5} \\ 0.05-e^{-6} \end{pmatrix}$$

Beispiel 6.10: Lösung

- Vereinfacht:

$$\begin{pmatrix} 1.0524 & 0.0551 \\ 0.0551 & 0.0609 \end{pmatrix} \begin{pmatrix} \delta_1^{(0)} \\ \delta_2^{(0)} \end{pmatrix} = \begin{pmatrix} 2.1980 \\ 0.2249 \end{pmatrix}$$

mit der Lösung

$$\begin{pmatrix} \delta_1^{(0)} \\ \delta_2^{(0)} \end{pmatrix} = \begin{pmatrix} 1.9894 \\ 1.8920 \end{pmatrix}$$

und damit

$$\begin{aligned} \lambda^{(1)} &= \lambda^{(0)} + \delta^{(0)}. \\ &= \begin{pmatrix} 1 \\ -1.5 \end{pmatrix} + \begin{pmatrix} 1.9894 \\ 1.8920 \end{pmatrix} = \begin{pmatrix} 2.9894 \\ 0.3920 \end{pmatrix} \end{aligned}$$

Beispiel 6.10: Lösung

- Die weiteren Lösungen des ungedämpften Verfahrens sind

i	0	1	2	5	10
$\lambda^{(i)}$	$\begin{pmatrix} 1 \\ -1.5 \end{pmatrix}$	$\begin{pmatrix} 2.99 \\ 0.392 \end{pmatrix}$	$\begin{pmatrix} 1.26 \\ 0.279 \end{pmatrix}$	$\begin{pmatrix} 2.91 \\ -0.856 \end{pmatrix}$	$\begin{pmatrix} 2.981658705 \\ -1.003280776 \end{pmatrix}$

- Ab $\lambda^{(13)} = (2.981658972, -1.003281352)^T$ tritt keine Veränderung der Ziffern mehr ein.
- Dies stimmt mit dem Ergebnis aus Bsp. 6.8 überein.
- Für den Startvektor $\lambda^{(0)} = (2,2)^T$ tritt hingegen keine Konvergenz ein, so erhalten wir z.B. $\lambda^{(13)} = (0.3698, 33.6868)^T$.

Beispiel 6.10: Lösung

- Dagegen sind die Ergebnisse des gedämpften Gauss-Newton-Verfahrens:

i	0	1	2	3	4
$\lambda^{(i)}$	$\begin{pmatrix} 1 \\ -1.5 \end{pmatrix}$	$\begin{pmatrix} 1.99 \\ -0.554 \end{pmatrix}$	$\begin{pmatrix} 2.919 \\ -0.951 \end{pmatrix}$	$\begin{pmatrix} 2.980 \\ -0.999 \end{pmatrix}$	$\begin{pmatrix} 2.981516868 \\ -1.002965939 \end{pmatrix}$

- Ab $\lambda^{(7)} = (2.981658324, -1.003279952)^T$ tritt keine Veränderung der Ziffern mehr ein.
- Man sieht dass das gedämpfte Gauss-Newton Verfahren bei gleichem Startvektor wesentlich schneller konvergiert.

Beispiel 6.10: Lösung

- Für den Startvektor $\lambda^{(0)} = (2,2)^T$ tritt beim gedämpften Verfahren ebenfalls Konvergenz ein:

i	0	1	2	5	10
$\lambda^{(i)}$	$\begin{pmatrix}2\\2\end{pmatrix}$	$\begin{pmatrix}0.00384\\2.00\end{pmatrix}$	$\begin{pmatrix}0.00384\\1.75\end{pmatrix}$	$\begin{pmatrix}0.207\\-0.752\end{pmatrix}$	$\begin{pmatrix}2.981652024\\-1.003266310\end{pmatrix}$

- Ab $\lambda^{(14)} = (2.981658971, -1.003281352)^T$ tritt keine Veränderung der Ziffern mehr auf.
- Man sieht, dass das gedämpfte Gauss-Newton-Verfahren auch für Startvektoren konvergiert, für die das ungedämpfte Gauss-Newton-Verfahren nicht konvergiert.

Aufgabe 6.7

- Bestimmen Sie mit dem Newton-Gauss Algorithmus für die folgenden Daten die Funktion der Form $f(x) = a\ln(x+b)$, die diese Daten bestmöglich im Sinne der kleinsten Fehlerquadrate approximiert bis auf eine Toleranz von 1e-5:

x_i	1	2	3	4
y_i	7.1	7.9	8.3	8.8

Aufgabe 6.7: Lösung