Martin-Pollich-Gymnasium
Mellrichstadt

Kollegstufenjahrgang
2001/2003

Regula falsi und Newton-Verfahren zum Lösen von nichtlinearen Gleichungen

Facharbeit im Leistungskurs
Mathematik
von
Dominik Gunreben

Martin-Pollich-Gymnasium
Mellrichstadt

Kollegstufenjahrgang
2001/2003

Facharbeit

des Schülers:

Dominik Gunreben

aus dem Leistungskurs:

Mathematik

zum Thema:

Regula falsi und Newton-Verfahren zum Lösen von nichtlinearen Gleichungen

Kursleiter: StR W. Fell

Abgabetermin: 03.02.2003

Erzielte Note: ..... in Worten: .....

Erzielte Punkte: ..... in Worten: .....
(einfache Wertung)

Unterschrift des Kursleiters: .....

Kenntnisnahme und Feststellung
des Ergebnisses durch das Direktorat: .....

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

Sowohl in der Mathematik als auch in anderen Naturwissenschaften stößt man immer wieder auf Gleichungen, die man nicht exakt lösen kann, da dieses mit sehr hohen Aufwand verbunden wäre. Um wenigstens eine gute Näherung der Lösung zu erhalten, hat man sogenannte Iterationsverfahren entwickelt. Man versucht sich hierbei schrittweise dem exakten Ergebnis beliebig genau anzunähern. Doch nicht nur einzelne Gleichungen, sondern auch Gleichungssysteme können auf diese Weise gelöst werden, auch wenn diese nichtlinearer Natur sind. Im Folgenden werden die beiden bekanntesten Möglichkeiten, Regula falsi und das Newton-Verfahren, vorgestellt.

2 Mathematische Grundlagen

Doch zuvor werden noch einige mathematische Grundlagen geschaffen, die gebraucht werden, um der Argumentation und den Erklärungen folgen zu können. Auf eine ausführliche Beschreibung wird an dieser Stelle verzichtet. Stattdessen wird das Wesentliche erläutert, das für das Verständnis der Facharbeit und der Verfahren notwendig ist.

2.1 Jacobi-Matrix

Ist f in $\vec{x}_0$ differenzierbar, dann heißt $f'(\vec{x}_0) := D_f(\vec{x}_0)$ die Ableitung von f an der Stelle $\vec{x}_0$.

Hat man also den Vektor $\vec{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}$ und will diesen nach $x_1, \ldots, x_n$ ableiten erhält man die Matrix: [1]

$$\vec{f}'(x) = D_f = \begin{pmatrix} \frac{\mathrm{d}f_1}{\mathrm{d}x_1} & \frac{\mathrm{d}f_1}{\mathrm{d}x_2} & \cdots & \frac{\mathrm{d}f_1}{\mathrm{d}x_n} \\ \frac{\mathrm{d}f_2}{\mathrm{d}x_1} & \frac{\mathrm{d}f_2}{\mathrm{d}x_2} & \cdots & \frac{\mathrm{d}f_2}{\mathrm{d}x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\mathrm{d}f_n}{\mathrm{d}x_1} & \frac{\mathrm{d}f_n}{\mathrm{d}x_2} & \cdots & \frac{\mathrm{d}f_n}{\mathrm{d}x_n} \end{pmatrix} \tag{2.1}$$

[1] [11]S.376

Will man die Ableitung von folgendem Gleichungssystem bestimmen:

$$\begin{aligned} f_1 &= x_1^2 + x_2 + x_3 - 4 \\ f_2 &= x_1 + x_2^2 + x_3 - 6 \\ f_3 &= x_1 + x_2 + x_3^2 - 4 \end{aligned} \tag{2.2}$$

Dann lautet die Ableitung:

$$\vec{f'} = D_f = \begin{pmatrix} 2x_1 & 1 & 1 \\ 1 & 2x_2 & 1 \\ 1 & 1 & 2x_3 \end{pmatrix} \tag{2.3}$$

2.2 Taylor-Polynom

Mithilfe des Taylor-Polynoms kann man jede Funktion an einem Funktionswert durch ein Polynom annähern. Der Grad des Polynoms ist hierbei ein Maß für die Genauigkeit. So kann man in Abbildung 1 auf der nächsten Seite erkennen, dass die Funktion $y = \sin x$ im Bereich von 0 bis 2π durch ein Polynom 17. Grades sehr gut angenähert wurde, wobei n die Gradzahl des Polynoms ist.
Diese Näherung ist besonders hilfreich, wenn man von einer Funktion nur eine lokale Betrachtung (z.B. Nullstellen) vornimmt, da die Funktion und das Polynom in diesem lokalen Bereich fast identisch sind. Man hat dadurch eine Vereinfachung erreicht, da man in der Regel mit diesem Polynom einfacher rechnen kann als mit der ursprünglichen Funktion.
Der Satz von Taylor[2] lautet:

$$\begin{aligned} f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \dots \\ + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n + R_n(x) \end{aligned} \tag{2.4}$$

wobei $R_n(x) = \frac{f^{(n+1)}(\epsilon)}{(n+1)!}(x - x_0)^{n+1}$ mit ϵ zwischen x und x_0.

Als Beispiel soll das Taylorpolynom 2-ter Ordnung der Funktion $\sin(x)$ am

[2] [11] S.354

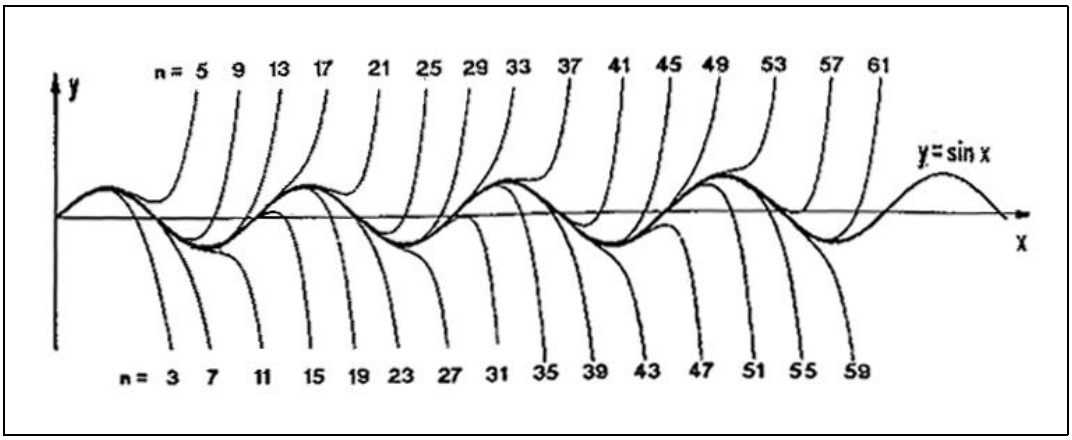

Abbildung 1: Taylorpolynome n-ter Ordnung für $y = \sin x$ ([10], S.238)

Punkt $x_0 = 0$ dienen:

$$f(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 \tag{2.5}$$

$$f(x) = 0 + \frac{f'(\sin 0)}{1!}(x - 0) + \frac{f''(\sin 0)}{2!}(x - 0)^2 \tag{2.6}$$

Da $f'(\sin 0) = 1$ und $f''(\sin 0) = 0$ folgt daraus:

$$f(x) = x \tag{2.7}$$

Ordnung	Funktionsterm
1	$f(x) = x$
3	$f(x) = x - \frac{x^3}{6}$
5	$f(x) = x - \frac{x^3}{6} + \frac{x^5}{120}$
7	$f(x) = x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040}$
9	$f(x) = x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362880}$

Tabelle 1: Polynome zur Näherung für $sin(x)$ am Punkt $x_0 = 0$

Die Polynome mit geradzahliger Ordnung n wurden in Tabelle 1 weggelassen, da sie mit den Polynomen $(n - 1)$-ter Ordnung übereinstimmen.

2.3 Direkte Fixpunktiteration

Funktionen lassen sich auf zwei Arten darstellen. Zum einen als Nullstellengleichung ($f(x) = 0$), zum anderen als Fixpunktgleichung ($x = \varphi(x)$). Lösungen dieser Gleichungen bezeichnet man als Nullstellen bzw. Fixpunkte. Beispiel:
Nullstellengleichung:

$$f(x) = 0 \tag{2.8}$$

$$x + y + 5 = 0 \tag{2.9}$$

Fixpunktgleichung:

$$\varphi(x) = x \tag{2.10}$$

$$-y - 5 = x \tag{2.11}$$

Anhand dieses Beispiels sieht man, dass jede Nullstelle von $f(x)$ auch ein Fixpunkt von $\varphi(x)$ ist und umgekehrt.
Schreibt man eine Iterationsfolge in Fixpunktform,

$$x_{n+1} := f(x_n) \qquad (n = 1, 2, 3, ...) \tag{2.12}$$

dann konvergiert diese Folge sicher für den Startwert $x_0 \in [a, b]$ gegen den Fixpunkt, wenn ...

... die Funktion $f(x)$ in $[a, b]$ stetig differenzierbar ist.

... gilt: $a \leq f(x) \leq b$ für alle $x \in [a, b]$.

... es eine Konstante K mit $|f'(x)| \leq K < 1$ für alle $x \in [a, b]$ gibt.[3]

[3] [10] S.132

2.4 Falksches Schema

Die Multipliktion zweier Matrizen errechnet man am besten unter zu Hilfenahme des Falkschen Schemas. Wird Matrix A mit dem Vektor $\vec{B}$ multipliziert erhält man den Vektor $\vec{AB}$. $A \cdot \vec{B} = \vec{AB}$

				3 -5 0	$\vec{B}$
A	1	3	7	-12	$\vec{AB}$
	2	-1	4	11	
	-1	0	1	-3	

Tabelle 2: Falksches Schema ([6] S.265)

Erklärungen zu Tabelle 2:

$$3 \cdot 1 - 5 \cdot 3 + 0 \cdot 7 = -12 \tag{2.13}$$

$$3 \cdot 2 - 5 \cdot (-1) + 0 \cdot 4 = \quad 11 \tag{2.14}$$

$$3 \cdot (-1) - 5 \cdot 0 + 0 \cdot 1 = - \ 3 \tag{2.15}$$

Man errechnet also das Skalarprodukt aus einem Zeilenvektor von A mit einem Spaltenvektor von B. Als Lösung erhält man:

$$\begin{pmatrix} 1 & 3 & 7 \\ 2 & -1 & 4 \\ -1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ -5 \\ 0 \end{pmatrix} = \begin{pmatrix} -12 \\ 11 \\ -3 \end{pmatrix} \tag{2.16}$$

Multipliziert man eine Determinante mit einem Vektor, erhält man wieder einen Vektor.[4]

[4] [6]S.265.

2.5 Relativer Fehler

Berechnet man die Lösung einer Gleichung mithilfe eines Iterationsverfahrens, muss man ein sogenanntes Abbruchkriterium festlegen. D.h. man kann, falls sich die Werte nur noch minimal ändern, das Verfahren abbrechen. Das Abbruchkriterium wird sehr oft durch den relativen Fehler angegeben. Der relative Fehler wird meist in Prozent angegeben und gibt die Abweichung zwischen zwei Werten an[5]:

$$\delta x_i = \left| \frac{\text{Abweichung vom Bezugswert}}{\text{Bezugswert}} \right| \tag{2.17}$$

$$\delta x_i = \left| \frac{\Delta x_i}{x} \right| \tag{2.18}$$

Hat man als Bezugswert $2,9$ und als abweichenden Wert 3 gegeben, beträgt der relative Fehler zwischen ihnen:

$$\frac{3 - 2,9}{2,9} \approx \underline{3,45\%} \tag{2.19}$$

[5] [6] S.813

3 Die Regula falsi

3.1 Geometrische Beschreibung

Mit dem Regula falsi-Verfahren kann man die Nullstelle einer Funktion näherungsweise bestimmen. Das Verfahren beruht geometrisch auf der Näherung eines Graphen durch eine Sekante. Man sucht sich zwei Punkte des Graphen mit unterschiedlichen Vorzeichen in einem stetigen Intervall $[x_n, x_m]$.
Durch den Nullstellensatz[6] muss es eine Nullstelle zwischen diesen beiden Punkten geben. Die Regula falsi-Iteration wird in Abbildung 2 dargestellt. Die Nullstelle wird angenähert, indem man die Punkte $(x_n/f(x_n))$ und $(x_m/f(x_m))$ miteinander verbindet. Der Schnittpunkt dieser Strecke mit der x-Achse bei x_{n+1} stellt eine Näherungslösung für die Nullstelle bei x^* dar. Jetzt dienen die Punkte $(x_{n+1}/f(x_{n+1}))$ und $(x_m/f(x_m))$ als neue Startpunkte. Führt man dieses fort, kann eine beliebig genaue Näherung der tatsächlichen Nullstelle errechnet werden.[7]

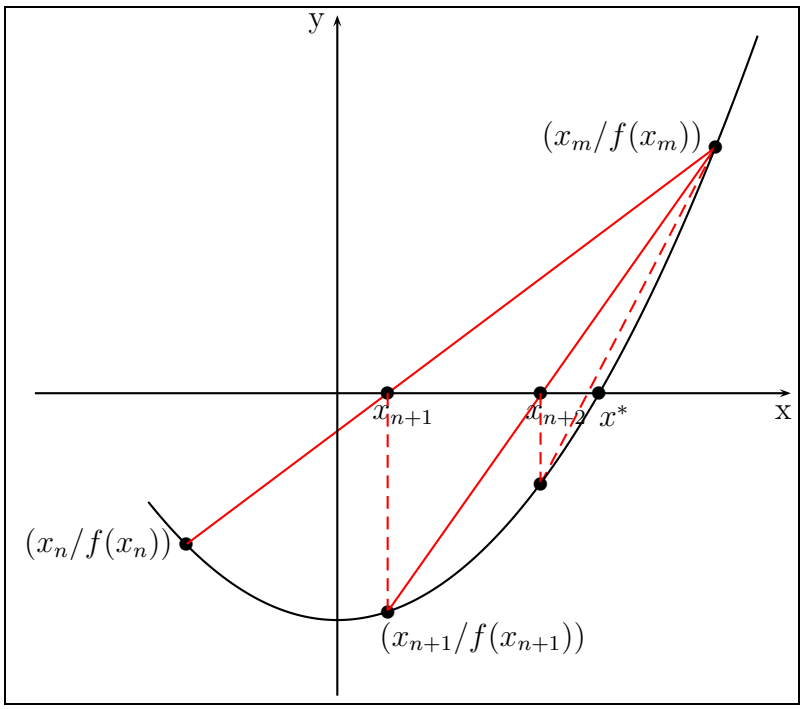

Abbildung 2: Regula falsi-Iteration ([6] S.909)

[6] [3] S.56
[7] [6] S.909

3.2 Mathematische Herleitung

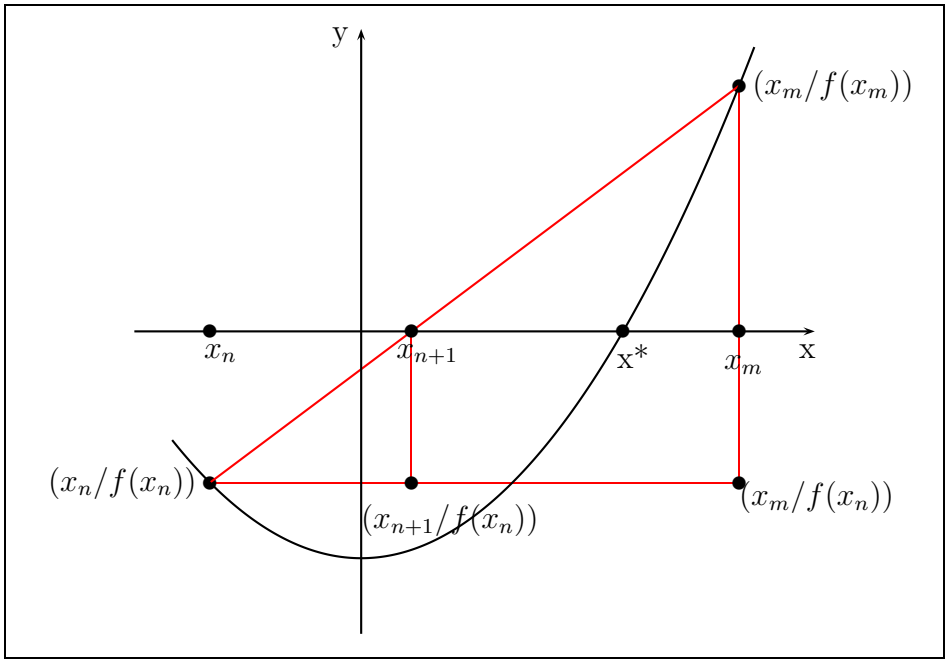

Abbildung 3: Herleitung von Regula falsi

Um dieses Verfahren mathematisch zu beschreiben, betrachtet man Abbildung 3. Die beiden Dreiecke $\triangle(x_n/(fx_n)), (x_{n+1}/f(x_n)), (x_{n+1}/0)$ und $\triangle(x_n/(fx_n)), (x_m/f(x_n)), (x_m/f(x_m))$ sind ähnlich, da sie in zwei gleichliegenden Innenwinkeln übereinstimmen. Es lässt sich mit dem 2.Strahlensatz folgende Beziehung aufstellen:[8]

$$\frac{f(x_m) - f(x_n)}{-f(x_n)} = \frac{x_m - x_n}{x_{n+1} - x_n} \tag{3.1}$$

Löst man diese Gleichung nach x_{n+1} auf, so erhält man:

$$x_{n+1} = x_n - f(x_n) \cdot \frac{(x_m - x_n)}{f(x_m) - f(x_n)} \tag{3.2}$$

Für die Gleichung $f(x) = \frac{1}{4}x^2 - 3$ erhält man mit dem Programm 11 auf Seite 37 die Tabelle 3 auf der nächsten Seite. Als Startwerte wurden $x_m = 5$ und $x_n = -2$ gewählt. Das Verfahren wird abgebrochen, falls der relative Fehler kleiner als 10^{-16} ist. Nach 24 Iterationen erhält man die Lösung

[8] [6] S.138

```
x 1=-2.000000000000000e+00 f(x 1)=-2.000000000000000e+00
x 2= 6.666666666666666e-01 f(x 2)=-2.888888888888889e+00
x 3= 2.705882352941177e+00 f(x 3)=-1.169550173010380e+00
x 4= 3.312977099236641e+00 f(x 4)=-2.560456849833926e-01
x 5= 3.436179981634527e+00 f(x 5)=-4.816678345353526e-02
x 6= 3.459018177860020e+00 f(x 6)=-8.798311308487579e-03
x 7= 3.463178618763914e+00 f(x 7)=-1.598463634118017e-03
x 8= 3.463934109676311e+00 f(x 8)=-2.901209552460598e-04
x 9= 3.464071218945588e+00 f(x 9)=-5.264751820779293e-05
x10= 3.464096099415917e+00 f(x10)=-9.553502856731767e-06
x11= 3.464100614252585e+00 f(x11)=-1.733583715866948e-06
x12= 3.464101433516814e+00 f(x12)=-3.145766887815375e-07
x13= 3.464101582180764e+00 f(x13)=-5.708318126561654e-08
x14= 3.464101609157369e+00 f(x14)=-1.035833113922392e-08
x15= 3.464101614052552e+00 f(x15)=-1.879625608153593e-09
x16= 3.464101614940833e+00 f(x16)=-3.410777775665619e-10
x17= 3.464101615102021e+00 f(x17)=-6.189205549976540e-11
x18= 3.464101615131270e+00 f(x18)=-1.123121886291434e-11
x19= 3.464101615136578e+00 f(x19)=-2.037918878788902e-12
x20= 3.464101615137541e+00 f(x20)=-3.695563820238501e-13
x21= 3.464101615137716e+00 f(x21)=-6.726672170664916e-14
x22= 3.464101615137748e+00 f(x22)=-1.188545789565509e-14
x23= 3.464101615137753e+00 f(x23)=-1.886078099255784e-15
x24= 3.464101615137754e+00 f(x24)=-3.475952164988527e-16
```

Tabelle 3: Ergebnisse einer Regula falsi-Iteration

$3,464101615137754$, was auf 16 Stellen der exakten Lösung $x^* = 2\sqrt{3}$ entspricht.

3.3 Eigenschaften

Der Vorteil von Regula falsi liegt in der sicheren Konvergenz, die dieses Verfahren garantiert, solange die Startbedingung erfüllt ist. Als Vorraussetzung werden zwei Punkte in einem stetigen Interval mit verschiedenen Vorzeichen gefordert[9]. Falls man nur an einer groben Näherung interessiert ist, ist das von Vorteil, da man keine großen Vorüberlegungen anstellen muss, um das Verfahren anzuwenden. Ist man an einer äußerst exakten Näherung interessiert, ist das Regula falsi-Verfahren weniger gut prädestiniert. Es konvergiert

[9] [6] S.909

im Vergleich zu anderen Verfahren relativ langsam, da, wie im nächsten Kapitel bewiesen wird, ein Verfahren 1-ter Ordnung vorliegt.
Es kann allerdings keine mehrfachen Nullstellen gerader Ordnung berechnen (z.B. $f(x) = x^2$), da diese keinen Vorzeichenwechsel haben und somit die Startbedingung nicht erfüllt ist.

3.4 Konvergenz des Regula falsi-Verfahrens

Für die Konvergenzordnung eines iterativen Verfahrens gilt: „Ist die erste von null verschiedene Ableitung der Funktion $F(x)$ an der Stelle x von k-ter Ordnung, dann ist auch das iterative Verfahren von k-ter Ordnung“. [10]
Daher untersuchen wir die Ableitung der Iterationsformel:[11]

$$F(x_n) = x_{n+1} = x_n - f(x_n) \cdot \frac{(x_m - x_n)}{f(x_m) - f(x_n)} \tag{3.3}$$

$$F'(x_n) = 1- \tag{3.4}$$
$$\frac{[f'(x_n)(x_m - x_n) - f(x_n)][f(x_m) - f(x_n)] - f(x_n)(x_m - x_n) \cdot (-f'(x_n))}{(f(x_n) - f(x_n))^2}$$

Geht man davon aus, dass man sich bereits stark an die Nullstelle angenähert hat, gilt $f(x_n) \approx 0$

$$F'(x_n) = 1 - \frac{f(x_m)(f'(x_n)(x_m - x_n))}{(f(x_m))^2} \tag{3.5}$$

$$F'(x_n) = 1 - \frac{f'(x_n)(x_m - x_n)}{f(x_m)} \tag{3.6}$$

Mit dieser Näherung darf auch $f(x_n)$ im Nenner subtrahiert werden ohne dass sich der Wert ändert.

$$F'(x_n) = 1 - f'(x_n)\frac{(x_m - x_n)}{f(x_m) - f(x_n)} \tag{3.7}$$

Betrachtet man den Quotienten, so erkennt man, dass dieser den Kehrwert der Steigung der Sekanten darstellt.
Da wir ein abgeschlossenes Intervall $[x_n, x_m]$ betrachten und die Funktion in diesem Intervall eine Ableitung besitzt, muss aufgrund des Mittelwertsatzes

[10] [7] S.38
[11] [2] S.18 f

der Differentialrechnung gelten:[12]

$$f'(\xi) = \frac{f(x_m) - f(x_n)}{x_m - x_n} \quad \text{mit } x_n < \xi < x_m \tag{3.8}$$

Die Abbildung 4 verdeutlicht diesen Satz.

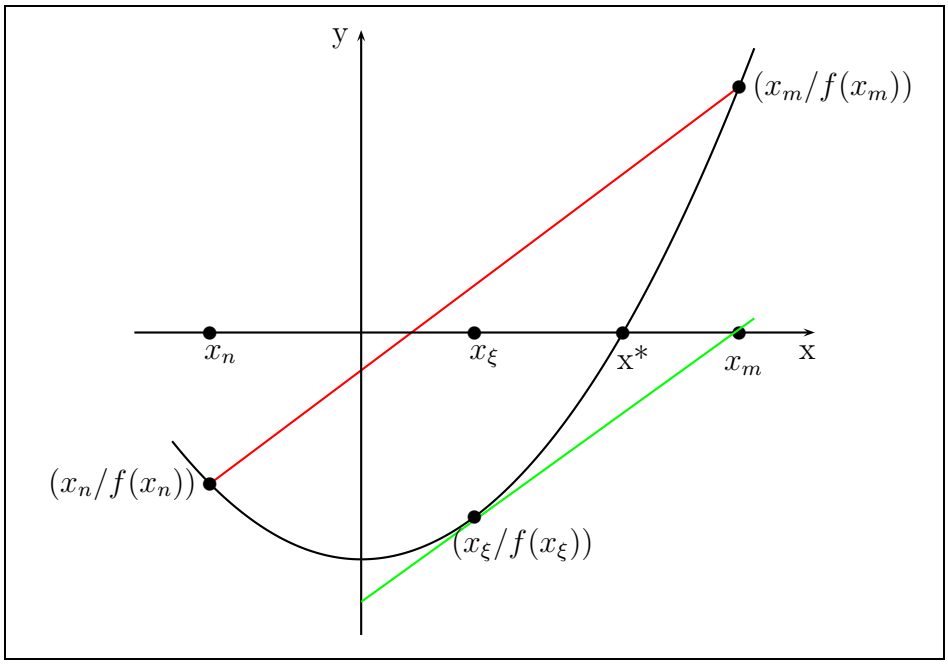

Abbildung 4: Sekante und Tangente sind parallel für ein $x_\xi \in]x_n; x_m[$

Die Gleichung 3.7 vereinfacht sich dann wie folgt:

$$F'(x) = 1 - \frac{f'(x_n)}{f'(\xi)} \tag{3.9}$$

Man erkennt, dass diese Gleichung nur für den seltenen Fall von $f'(x_n) = f'(\xi)$ null wird. Es müsste im Intervall $]x_n, x_m[$ ein x_ξ geben an dem die Steigung genauso groß wäre wie bei x_n. Dies wäre nur der Fall, wenn ein Wendepunkt zwischen x_n und x_m liegt. Dann würde das Verfahren unter Umständen schneller konvergieren. Da aber die wenigsten Funktionen ihren Wendepunkt in direkter Nähe der Nullstelle haben, wird dieser Fall nicht weiter besprochen. Somit gilt für die meisten Funktionen: $f'(x_n) \neq f'(\xi)$.
Damit ist die erste Ableitung in den allermeisten Fällen von null verschieden. Daraus folgt, dass Regula falsi ein Verfahren 1-ter Ordnung ist. Verfahren 1-ter Ordnung haben die Eigenschaft linear zu konvergieren, d.h. der Fehler

[12] [6] S.403

zwischen Näherungswert und exakter Lösung verringert sich pro Iterationsschritt um einen konstanten Faktor M[13]:

$$x^* - x_{n+1} = M(x^* - x_n) \tag{3.10}$$

$$\begin{aligned} x^* - x_{n+2} &= M(x^* - x_{n+1}) = M(M(x^* - x_n)) \\ &= M^2(x^* - x_n) \end{aligned} \tag{3.11}$$

Setzt man $\sigma = x^* - x_n$ und führt man diese Reihe fort, erhält man:

$$M\sigma, M^2\sigma, M^3\sigma, \ldots, M^n\sigma, \ldots \tag{3.12}$$

3.5 Sekanten-Verfahren

Die Beschleunigungsmöglichkeit für Regula falsi besteht darin, dass man, falls die Konvergenz einmal eingetreten ist, entgegen der Startbedingung $x_m = x_{n-1}$ setzt[14]. Damit erhält man folgende Iterationsvorschrift, die man als Sekanten-Verfahren bezeichnet:

$$x_{n+1} = x_n - f(x_n) \cdot \frac{(x_{n-1} - x_n)}{f(x_{n-1}) - f(x_n)} \tag{3.13}$$

Nach der Berechnung eines neuen x_{n+1}-Wertes wird davon der Funktionswert $f(x_{n+1})$ berechnet. Dieser Punkt wird mit dem Punkt $(x_n/f(x_n))$ verbunden. Der Schnittpunkt mit der x-Achse stellt nun eine bessere Näherung dar. Somit kann man eine deutliche Beschleunigung gegenüber dem normalen Regula falsi-Verfahren erreichen, weil man immer die beiden besten Näherungen für die Berechnung des nächsten Wertes verwendet. Graphisch kann man dieses Verfahren anhand der Abbildung 5 auf der nächsten Seite nachvollziehen.
Es soll wieder die Gleichung $y = \frac{1}{4}x^2 - 3$ mit dem Programm 12 auf Seite 38 mit den Startparametern $x_m = x_{n-1} = 5$ und $x_n = -2$ gelöst werden. Vergleicht man die Ergebnisse des Regula falsi-Verfahrens und des Sekanten-Verfahrens (Tabelle 3 auf Seite 13 und Tabelle 4 auf der nächsten Seite) miteinander, erkennt man anhand der Anzahl der benötigten Iterationsschritte den Geschwindigkeitsgewinn des Sekanten-Verfahrens.

[13] [7] S.39
[14] [6] S.909

Erklärung zu Abbildung 5:
Man verbindet die Punkte $f(x_5)$ und $f(x_4)$ miteinander und erhält den Schnittpunkt mit der x-Achse bei x_6. Der Funktionswert von $f(x_6)$ wird berechnet. Jetzt verbindet man die Punkte $f(x_6)$ und $f(x_5)$ und erhält x_7.

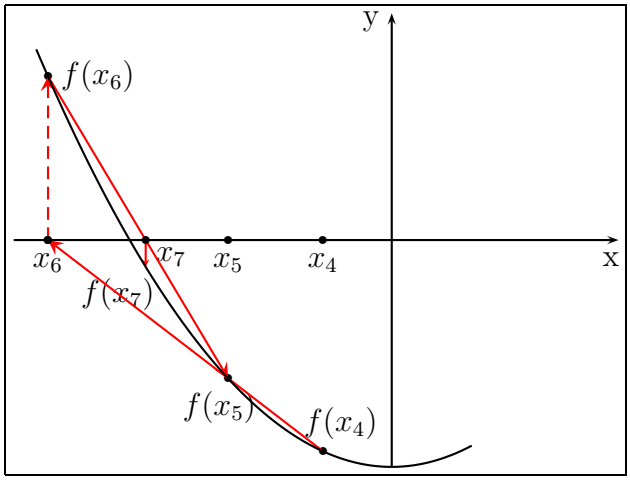

Abbildung 5: Sekanten-Verfahren

```
x 1=-2.000000000000000e+00    f(x 1)=-2.000000000000000e+00
x 2= 6.666666666666666e-01    f(x 2)=-2.888888888888889e+00
x 3=-8.000000000000000e+00    f(x 3)= 1.300000000000000e+01
x 4=-9.090909090909091e-01    f(x 4)=-2.793388429752066e+00
x 5=-2.163265306122449e+00    f(x 5)=-1.830070803831737e+00
x 6=-4.545893719806763e+00    f(x 6)= 2.166287427944642e+00
x 7=-3.254353352632585e+00    f(x 7)=-3.522960640522643e-01
x 8=-3.435012278320738e+00    f(x 8)=-5.017266194644325e-02
x 9=-3.465013725222837e+00    f(x 9)= 1.580028995660950e-03
x10=-3.464097769836669e+00    f(x10)=-6.660253153561611e-06
x11=-3.464101614631580e+00    f(x11)=-8.767198030924794e-10
x12=-3.464101615137755e+00    f(x12)= 4.215378046623641e-16
x13=-3.464101615137754e+00    f(x13)=-3.475952164988527e-16
```

Tabelle 4: Ausgabe des Sekanten-Verfahrens

4 Das Newton-Verfahren

4.1 Allgemeine Beschreibung des Verfahrens

Das Newton-Verfahren errechnet den nächsten Näherungswert auf eine andere Weise als die Regula falsi. Man wählt einen Startpunkt, der in der Nähe der Nullstelle liegt (d.h. man muss durch eine Skizze oder sonstige Näherungen z.B. Regula falsi schon ungefähr wissen, wo die Nullstelle sein wird). Von diesem Startwert berechnet man die Tangente an den Graphen der Funktion. Der Schnittpunkt dieser Tangente mit der x-Achse stellt eine erste Näherung an die Nullstelle dar. Der Differenzenquotient des Regula falsi-Verfahrens wurde durch die Ableitung ersetzt.[15]

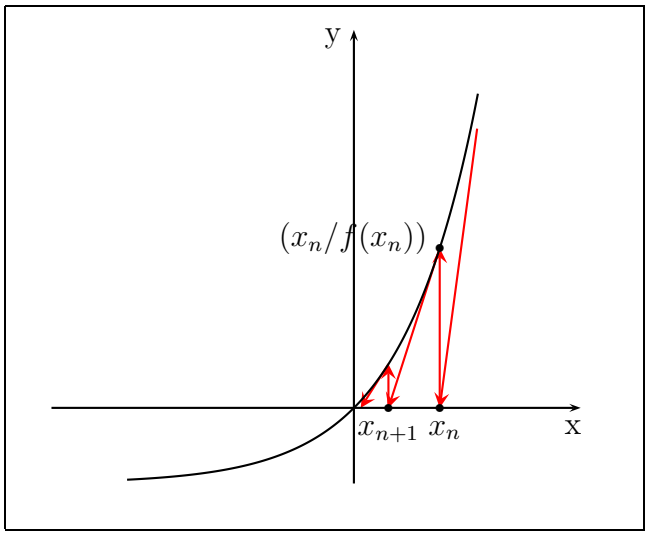

Abbildung 6: Newton-Verfahren ([6] S.909)

4.2 Mathematische Herleitung

Die mathematische Formel für die Errechnung der nächsten Näherung ist leicht nachzuvollziehen, wenn man das Dreieck $\triangle(x_{n+1}/0),(x_n/0),(x_n/f(x_n))$ betrachtet:

$$f'(x_n) = \frac{f(x_n)}{x_n - x_{n+1}} \tag{4.1}$$

[15] [10] S.133

Löst man diese Gleichung nach x_{n+1} auf, erhält man:

$$x_n - x_{n+1} = \frac{f(x_n)}{f'(x_n)} \tag{4.2}$$

$$F(x) = x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{4.3}$$

Da es sich bei diesem Verfahren um eine Fixpunktiteration handelt, müssen die Bedingungen der Direkten Fixpunktiteration aus Kapitel 2.3 erfüllt sein. Somit muss gelten:[16]

$$|F'(x)| \leq K < 1 \tag{4.4}$$

$$F'(x) = \left| \frac{f(x)f''(x)}{(f'(x))^2} \right| \leqq K < 1 \qquad (K = const) \tag{4.5}$$

Außerdem muss $f'(x_n) \neq 0$ sein, da sonst die Gleichung 4.3 nicht definiert ist.[17]

4.3 Eigenschaften des Newton-Verfahrens

Der größte Vorteil des Newton-Verfahrens besteht in der quadratischen Konvergenz, d.h. falls die Konvergenz einmal eingetreten ist, verdoppelt sich die Anzahl der richtigen Stellen pro Iterationsschritt. Somit ist diese Art der Näherung sehr schnell. Das Newton-Verfahren erreicht diese quadratische Konvergenz allerdings nur, wenn die Gleichungen bestimmte Eigenschaften erfüllen. Ist nämlich die gesuchte Nullstelle von gerader Ordnung oder liegen zwei Nullstellen sehr nahe beieinander, konvergiert das Verfahren sehr langsam. Auch wenn die Steigung im Bereich der Nullstelle sehr klein ist, wird die quadratische Konvergenz oft nicht erreicht. In diesem Fall muss man die Zwischenwerte für $f(x_n)$ und $f'(x_n)$ mit einer sehr hohen Genauigkeit berechnen, um einen akzeptablen Wert für x_{n+1} zu erhalten.
Ansonsten müssen nur die Werte für $f(x)$ sehr genau berechnet werden. Die Werte für $f'(x)$ kann man annähern, was aber in Kapitel 4.6 auf Seite 24 genauer besprochen wird[18].
Wählt man als Startwert eine komplexe Zahl, so erhält man auch die komple-

[16] [10] S.133
[17] [6] S.908
[18] [7] S.34 ff

xen Nullstellen.[19] Da komplexe Nullstellen nur selten benötigt werden, wird dieses Problem hier nicht weiter vertieft.
Als Beispiel soll wie schon bei Regula falsi die Funktion $f(x) = \frac{1}{4}x^2 - 3$ dienen. Mit dem Programm 13 auf Seite 39 ergibt sich mit dem Startwert $x_n = 5$ die Tabelle 5. Als Abbruchkriterium wurde ein relativer Fehler von weniger als 10^{-16} zwischen zwei Iterationswerten festgelegt. Das Ergebnis wird nach sechs Iterationen erreicht.

```
x 1= 5.000000000000000e+00    f(x 1)= 3.250000000000000e+00
x 2= 3.700000000000000e+00    f(x 2)= 4.225000000000003e-01
x 3= 3.471621621621622e+00    f(x 3)= 1.303917092768437e-02
x 4= 3.464109759818207e+00    f(x 4)= 1.410701693839249e-05
x 5= 3.464101615147329e+00    f(x 5)= 1.658405184012945e-11
x 6= 3.464101615137754e+00    f(x 6)=-3.475952164988527e-16
```

Tabelle 5: Iterationen des Newton-Verfahrens

4.4 Konvergenzordnung des Newton-Verfahrens

Beweis der quadratischen Konvergenz

Das Newton-Verfahren konvergiert quadratisch, wenn es sich ausreichend an die Lösung angenähert hat und die Lösung kein Extremwert ist.
Wir betrachten die Taylor-Entwicklung für diesen Fall. Da $(x - x_0)^n \approx 0$ mit $n > 2$, können wir Glieder höherer Ordnung außer Acht lassen[20].

$$f(x^*) = 0 = f(x_n) + \frac{f'(x_n)}{1!}(x^* - x_n) + \frac{f''(x_n)}{2!}(x^* - x_n)^2 \tag{4.6}$$

Da aus Gleichung 4.3 $f'(x) \neq 0$ folgt, darf man durch $f'(x)$ teilen:

$$0 = \frac{f(x_n)}{f'(x_n)} + x^* - x_n + \frac{f''(x)}{2!f'(x)}(x^* - x_n)^2 \tag{4.7}$$

[19] [6] S.909
[20] [7]S.35 und S.39 und [8] S.9

Mit Gleichung 4.2 erhält man:

$$0 = x_n - x_{n+1} + x^* - x_n + \frac{f''(x)}{2! f'(x)}(x^* - x_n)^2 \tag{4.8}$$

$$x^* - x_{n+1} = -\frac{f''(x)}{2 \cdot f'(x)}(x^* - x_n)^2 \tag{4.9}$$

Für $N = -\frac{f''}{2 \cdot f'(x)}$ gilt:

$$x^* - x_{n+1} = N(x^* - x_n)^2 \tag{4.10}$$

Wird diese Reihe fortgeführt, erhält man:

$$x^* - x_{n+2} = N(x^* - x_{n+1})^2 \tag{4.11}$$

Aus beiden Gleichungen ergibt sich mit $\sigma = x - x_n$:

$$x^* - x_{n+2} = N(N(x^* - x_n)^2)^2 = N^3(x^* - x_n)^4 \tag{4.12}$$

$$\hookrightarrow \; N\sigma^2, N^3\sigma^4, N^7\sigma^8, \ldots N^{-1}(N\sigma)^{2^r}, \ldots \tag{4.13}$$

Hier kann man ablesen, dass der Fehler von x_{n+1} proportional zum Quadrat des Fehlers von x_n ist. Es liegt also ein Verfahren 2-ter Ordnung vor. Da der Unterschied aufgrund der fortgeschrittenen Iteration kleiner als eins ist, verdoppelt sich die Anzahl der richtigen Stellen, was ein Kennzeichen für die quadratische Konvergenz ist.[21]

Beweis der Konvergenz von $f(x) = x^2 - a$

An dieser Stelle wird bewiesen, dass das Newton-Verfahren bei der Berechnung der Lösung von $f(x) = x^2 - a$ gegen den richtigen Wert konvergiert[22].

Es soll die Gleichung $x^2 = a$ gelöst werden. D.h. man sucht die Nullstellen der Funktion $f(x) = x^2 - a$. Daraus folgt:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{4.14}$$

[21] [7] S.35 und S.39
[22] [9]

Mit $f(x) = x^2 - a$:

$$x_{n+1} = x_n - \frac{x_n^2 - a}{2 \cdot x_n} \tag{4.15}$$

$$x_{n+1} = x_n - \frac{1}{2} \cdot x_n + \frac{1}{2} \cdot \frac{a}{x_n} \tag{4.16}$$

$$x_{n+1} = \frac{1}{2}(x_n + \frac{a}{x_n}) \tag{4.17}$$

Da man nur große n betrachtet, gilt: $\lim_{n\to\infty} x_n = \lim_{n\to\infty} x_{n+1}$

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} x_{n+1} = x^* = \lim_{n\to\infty} \frac{1}{2}\left(x_n + \frac{a}{x_n}\right) \tag{4.18}$$

$$x^* = \frac{1}{2}(x^* + \frac{a}{x^*}) \tag{4.19}$$

$$2x^* = x^* + \frac{a}{x^*} \tag{4.20}$$

$$x^* = \frac{a}{x^*} \Rightarrow x^{*2} = a \tag{4.21}$$

$$\hookrightarrow x^* = \sqrt{a} \qquad q.e.d \tag{4.22}$$

4.5 Divergenz

Aber das Newton-Verfahren muss nicht immer konvergieren. Es gibt drei Fälle, in denen das Newton-Verfahren nicht das gewünschte Ergebnis liefert. Diese Fälle können nicht nur am Anfang einer Iteration auftreten. Deshalb sollte man schon den Graphen der Funktion vor Augen haben, wenn man die Startwerte wählt, dass sicher eine Konvergenz eintritt.[23]

- Das Verfahren trifft auf einen Extremwert. In diesem Fall ist $f'(x) = 0$. Da man durch null nicht teilen darf, bricht das Verfahren ab (Abbildung 7 auf der nächsten Seite).
- Die Iterationsfolge pendelt zyklisch zwischen einer Reihe von Werten (Abbildung 8 auf der nächsten Seite).
- Das Verfahren nähert sich nicht an die Nullstelle an, sondern entfernt sich (Abbildung 9 auf der nächsten Seite).

[23] [8] S.11

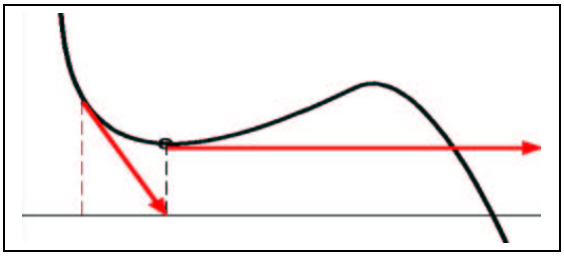

Abbildung 7: Divergenz bei einem Extremwert ([8] S.11)

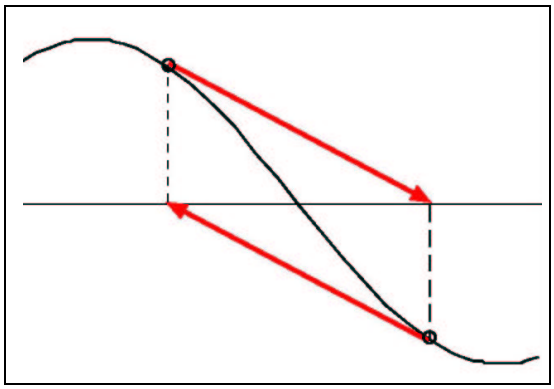

Abbildung 8: Endlosschleife ([8] S.11)

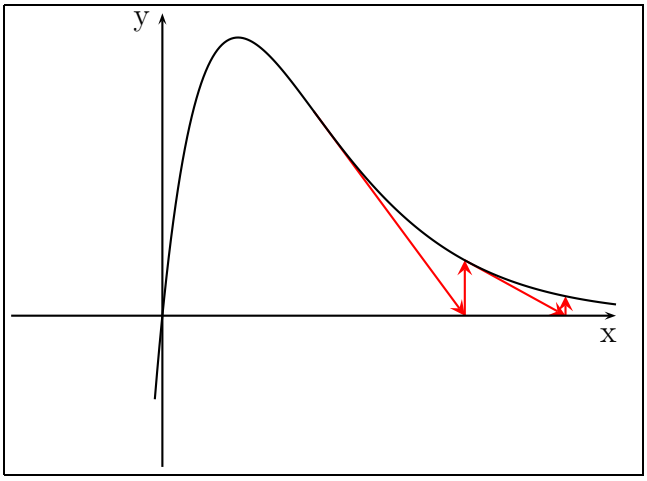

Abbildung 9: Newton-Verfahren divergiert

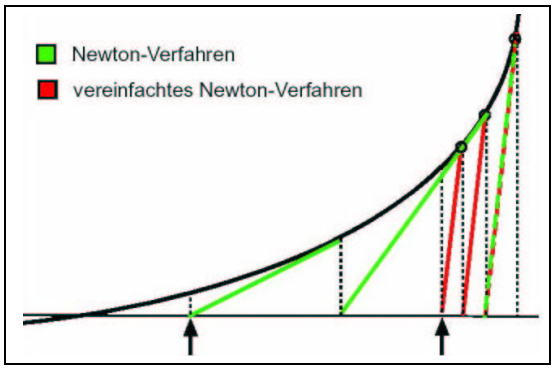

Abbildung 10: Vereinfachtes Newton-Verfahren ([8] S.10)

4.6 Vereinfachtes Newton-Verfahren

Man benötigt für das Newton-Verfahren sehr viel Rechenleistung, falls die Ableitung aufwendig zu berechnen ist. Es kann vorkommen, dass die Ableitung nicht explizit angeben werden kann, sondern nur partielle Ableitungen berechnet werden können. Dann muss man diese Ableitungen durch den Differenzenquotienten approximieren. Damit man diesen Rechenaufwand nicht bei jeder Näherung durchführen muss, kann die Rechnung vereinfacht werden.[24]

Ist die Iteration des Newton-Verfahrens schon fortgeschritten, hat die Konvergenz also bereits eingesetzt, kann eine berechnete Ableitung immer wieder verwendet werden. Dadurch spart man sich die Zeit, die Ableitung bei jedem Iterationsschritt neu zu berechnen, da immer wieder der gleiche Wert für $f'(x)$ eingesetzt wird. Dieses Verfahren konvergiert zwar nicht mehr quadratisch, hat aber dennoch eine hohe Geschwindigkeit[25]. Wie das vereinfachte Newton-Verfahren graphisch dargestellt wird, zeigt die Abbildung 10.

[24] [6] S.921
[25] [7] S.36

5 Das Lösen linearer Gleichungssysteme

An dieser Stelle werden zwei Verfahren vorgestellt, mit denen man lineare Gleichungssysteme lösen kann. Diese werden später beim Lösen von nichtlinaren Gleichungssystemen benötigt.
Es sei ein lineares Gleichungssystem mit $n \in \mathbb{R}$ gegeben[26].

$$\begin{aligned} a_{11}x_1 + a_{12}x_1 + \ldots + a_{1n}x_n =& b_1 \\ a_{21}x_1 + a_{22}x_1 + \ldots + a_{2n}x_n =& b_2 \\ & \vdots \\ a_{n1}x_1 + a_{n2}x_1 + \ldots + a_{nn}x_n =& b_n \end{aligned} \tag{5.1}$$

Es wird nun angenommen, dass a_{ii} von null verschieden ist. Somit kann man nach x_i auflösen und erhält:

$$\begin{aligned} x_1 =& \frac{(b_1 - a_{12}x_2 - a_{13}x_3 - \ldots - a_{1n}x_n)}{a_{11}} \\ x_2 =& \frac{(b_2 - a_{21}x_1 - a_{23}x_3 - \ldots - a_{2n}x_n)}{a_{22}} \\ & \vdots \\ x_n =& \frac{(b_n - a_{n1}x_1 - a_{n2}x_2 - a_{n3}x_3 - \ldots - a_{n,n-1}x_{n-1})}{a_{nn}} \end{aligned} \tag{5.2}$$

Jetzt wählt man Startwerte für $(x_1, \ldots, x_n)$ und errechnet so x_1. An dieser Stelle gibt es zwei Möglichkeiten fortzufahren, zum einen das Jacobi- und zum anderen das Gauss-Seidel-Verfahren.

5.1 Jacobi-Verfahren

Errechnet man nun auch mit den Startwerten einen Näherungswert für $x_i^{(k)}$ und benutzt die neuen besseren Näherungswerte für die Berechnung von $x_i^{(k+1)}$ erst, wenn man alle x_i berechnet hat, dann handelt es sich um das Jacobi- oder das Gesamtschritt-Verfahren.

[26] [4]

Das Jacobi-Verfahren hat folgende Startbedingungen[27]:

$$\max_k \sum_{\substack{i=1 \\ (i \neq k)}}^{n} \left| \frac{a_{ik}}{a_{ii}} \right| < 1 \qquad \textit{Spaltensummenkriterium} \tag{5.3}$$

oder

$$\max_i \sum_{\substack{k=1 \\ (k \neq i)}}^{n} \left| \frac{a_{ik}}{a_{ii}} \right| < 1 \qquad \textit{Zeilensummenkriterium} \tag{5.4}$$

Die Iterationsvorschrift für das Jacobi-Verfahren lautet [28]:

$$x_i^{(k+1)} = (b_i - a_{i1}x_1^{(k)} - \ldots - a_{i,i-1}x_{i-1}^{(k)} - a_{i,i+1}x_{i+1}^{(k)} - \ldots - a_{in}x_n^{(k)})/a_{ii} \tag{5.5}$$

In der Gleichung oben kennzeichnen die hochgestellten Zahlen den Iterationsschritt. So ist x^k die k-te Berechnung des Wertes x.

Als Beispiel soll folgendes Gleichungssystem gelöst werden:

$$\begin{aligned} 4x_1 + \ x_2 + 2x_3 &= 7 \\ 2x_1 + 4x_2 + \ x_3 &= 7 \\ x_2 + 2x_3 &= 3 \end{aligned} \tag{5.6}$$

In diesem Fall ist das Zeilensummenkriterium (5.4) erfüllt:

$$\begin{aligned} \left| \frac{1}{4} + \frac{2}{4} \right| &= \frac{3}{4} < 1 \\ \left| \frac{2}{4} + \frac{1}{4} \right| &= \frac{3}{4} < 1 \\ \left| \frac{1}{2} \right| &= \frac{1}{2} < 1 \end{aligned} \tag{5.7}$$

[27] [6] S.918
[28] [4]

Es ergibt sich folgende Iterationsvorschrift:

$$\begin{aligned} x_1^{(k+1)} &= \frac{7 - {x_2}^{(k)} - 2x_3^{(k)}}{4} \\ x_2^{(k+1)} &= \frac{7 - 2{x_1}^{(k)} - x_3^{(k)}}{4} \\ x_3^{(k+1)} &= \frac{3 - {x_2}^{(k)}}{2} \end{aligned} \tag{5.8}$$

Mit den Startwerten $x_1 = 100,\ x_2 = -30$ und $x_3 = -50$ ergibt sich:

$$\begin{aligned} x_1^{(2)} &= \frac{7 - (-30) - 2(-50)}{4} &&= \ 34{,}25 \\ x_2^{(2)} &= \frac{7 - 2(-30) - (-50)}{4} &&= -35{,}75 \\ x_3^{(2)} &= \frac{3 - (-30)}{2} &&= \ 16{,}50 \end{aligned} \tag{5.9}$$

Diese Werte werden nun als bessere Näherungen eingesetzt:

$$\begin{aligned} x_1^{(3)} &= \frac{7 - (-35,75) - 2(16,50)}{4} &&= \ 2{,}4375 \\ x_2^{(3)} &= \frac{7 - 2(34,25) - (16,50)}{4} &&= -19{,}5 \\ x_3^{(3)} &= \frac{3 - (-35,75)}{2} &&= \ 19{,}375 \end{aligned} \tag{5.10}$$

Das Pascalprogramm 14 auf Seite 40 führt diese Rechnungen automatisch durch und liefert bei einem relativen Fehler von 10^{-6} nach 38 Iterationen die Lösung $x_1 = 1,\ x_2 = 1$ und $x_3 = 1$. Die Tabelle 6 auf der nächsten Seite zeigt die gesamte Ausgabe des Programms.

```
 1 x1= 1.0000000e+02 x2=-3.0000000e+01 x3=-5.0000000e+01
 2 x1= 3.4250000e+01 x2=-3.5750000e+01 x3= 1.6500000e+01
 3 x1= 2.4375000e+00 x2=-1.9500000e+01 x3= 1.9375000e+01
 4 x1=-3.0625000e+00 x2=-4.3125000e+00 x3= 1.1250000e+01
 5 x1=-2.7968750e+00 x2= 4.6875000e-01 x3= 3.6562500e+00
 6 x1=-1.9531250e-01 x2= 2.2343750e+00 x3= 1.2656250e+00
 7 x1= 5.5859375e-01 x2= 1.5312500e+00 x3= 3.8281250e-01
 8 x1= 1.1757812e+00 x2= 1.3750000e+00 x3= 7.3437500e-01
 9 x1= 1.0390625e+00 x2= 9.7851562e-01 x3= 8.1250000e-01
10 x1= 1.0991211e+00 x2= 1.0273438e+00 x3= 1.0107422e+00
11 x1= 9.8779297e-01 x2= 9.4775391e-01 x3= 9.8632812e-01
12 x1= 1.0198975e+00 x2= 1.0095215e+00 x3= 1.0261230e+00
13 x1= 9.8455811e-01 x2= 9.8352051e-01 x3= 9.9523926e-01
14 x1= 1.0065002e+00 x2= 1.0089111e+00 x3= 1.0082397e+00
15 x1= 9.9365234e-01 x2= 9.9468994e-01 x3= 9.9554443e-01
16 x1= 1.0035553e+00 x2= 1.0042877e+00 x3= 1.0026550e+00
17 x1= 9.9760056e-01 x2= 9.9755859e-01 x3= 9.9785614e-01
18 x1= 1.0016823e+00 x2= 1.0017357e+00 x3= 1.0012207e+00
19 x1= 9.9895573e-01 x2= 9.9885368e-01 x3= 9.9913216e-01
20 x1= 1.0007205e+00 x2= 1.0007391e+00 x3= 1.0005732e+00
21 x1= 9.9952865e-01 x2= 9.9949646e-01 x3= 9.9963045e-01
22 x1= 1.0003107e+00 x2= 1.0003281e+00 x3= 1.0002518e+00
23 x1= 9.9979210e-01 x2= 9.9978173e-01 x3= 9.9983597e-01
24 x1= 1.0001366e+00 x2= 1.0001450e+00 x3= 1.0001091e+00
25 x1= 9.9990919e-01 x2= 9.9990442e-01 x3= 9.9992752e-01
26 x1= 1.0000601e+00 x2= 1.0000635e+00 x3= 1.0000478e+00
27 x1= 9.9996023e-01 x2= 9.9995799e-01 x3= 9.9996824e-01
28 x1= 1.0000264e+00 x2= 1.0000278e+00 x3= 1.0000210e+00
29 x1= 9.9998254e-01 x2= 9.9998156e-01 x3= 9.9998609e-01
30 x1= 1.0000116e+00 x2= 1.0000122e+00 x3= 1.0000092e+00
31 x1= 9.9999234e-01 x2= 9.9999191e-01 x3= 9.9999390e-01
32 x1= 1.0000051e+00 x2= 1.0000054e+00 x3= 1.0000040e+00
33 x1= 9.9999664e-01 x2= 9.9999645e-01 x3= 9.9999732e-01
34 x1= 1.0000022e+00 x2= 1.0000024e+00 x3= 1.0000018e+00
35 x1= 9.9999853e-01 x2= 9.9999844e-01 x3= 9.9999882e-01
36 x1= 1.0000010e+00 x2= 1.0000010e+00 x3= 1.0000008e+00
37 x1= 9.9999935e-01 x2= 9.9999932e-01 x3= 9.9999948e-01
38 x1= 1.0000004e+00 x2= 1.0000005e+00 x3= 1.0000003e+00
```

Tabelle 6: Ergebnisse des Jacobi-Verfahrens

5.2 Gauss-Seidel-Verfahren

Setzt man in die Gleichung für x_i^k schon die bessere Näherung für $x_{i-1}^{(k+1)}$ ein, dann handelt es sich um das Gauss-Seidel- oder Einzelschritt-Verfahren.
Die Iterationsvorschrift lautet:[29]

$$x_i^{k+1} = (b_i - a_{i1}x_1^{k+1} - \ldots - a_{i,i-1}x_{i-1}^{k+1} - a_{i,i+1}x_{i+1}^k - \ldots - a_{in}x_n^k)/a_{ii} \tag{5.11}$$

Es soll das Gleichungssystem 5.2 gelöst werden, das auch beim Jacobi-Verfahren als Beispiel dient:

$$\begin{aligned} x_1^{(k+1)} &= \frac{7 - {x_2}^{(k)} - 2x_3^{(k)}}{4} \\ x_2^{(k+1)} &= \frac{7 - 2{x_1}^{(k+1)} - x_3^{(k)}}{4} \\ x_3^{(k+1)} &= \frac{3 - {x_2}^{(k+1)}}{2} \end{aligned} \tag{5.12}$$

Mit den Startwerten $x_1 = 100,\ x_2 = -30$ und $x_3 = -50$ ergibt sich:

$$x_1^{(2)} = \frac{7 - (-30) - 2(-50)}{4} \quad = \quad 34{,}25$$

Die errechneten Näherungen werden sofort eingesetzt und man erhält:

$$\begin{aligned} x_2^{(2)} &= \frac{7 - 2(34,25) - (-50)}{4} &&= -\ 2{,}875 \\ x_3^{(2)} &= \frac{3 - (-2,875)}{2} &&= \ 16{,}50 \end{aligned} \tag{5.13}$$

Die gesamten Lösungen des Programms 15 auf Seite 41 können in Tabelle 7 auf der nächsten Seite nachvollzogen werden. Es wurden das gleiche Abbruchkriterium und die gleichen Startwerte wie schon beim Jacobi-Verfahren gewählt, so dass die Ergebnisse vergleichbar sind. Anhand der Zahl der Iterationsschritte erkennt man, dass das Gauss-Seidel-Verfahren sehr viel schneller konvergiert als das Jacobi-Verfahren. Dies ist auch nicht weiter verwunderlich, da die neuen verbesserten Werte sofort verwendet werden und diese nicht auf einen neuen Durchgang „warten“ müssen. Auch werden die nachfolgenden Werte mit einem geringeren Fehler berechnet, was zu einer schnelleren Konvergenz führt.

[29] [4]

Die Startwerte wurden im Beispiel bewusst sehr extrem gewählt, um einerseits den deutlichen Geschwindigkeitsunterschied der beiden Verfahren hervorzuheben und um andererseits zu zeigen, dass die Verfahren auch noch für Werte konvergieren, die sehr weit von der tatsächlichen Lösung entfernt sind.

```
 1 x1= 1.0000000e+02 x2=-3.0000000e+01 x3=-5.0000000e+01
 2 x1= 3.4250000e+01 x2=-2.8750000e+00 x3= 2.9375000e+00
 3 x1= 1.0000000e+00 x2= 5.1562500e-01 x3= 1.2421875e+00
 4 x1= 1.0000000e+00 x2= 9.3945312e-01 x3= 1.0302734e+00
 5 x1= 1.0000000e+00 x2= 9.9243164e-01 x3= 1.0037842e+00
 6 x1= 1.0000000e+00 x2= 9.9905396e-01 x3= 1.0004730e+00
 7 x1= 1.0000000e+00 x2= 9.9988174e-01 x3= 1.0000591e+00
 8 x1= 1.0000000e+00 x2= 9.9998522e-01 x3= 1.0000074e+00
 9 x1= 1.0000000e+00 x2= 9.9999815e-01 x3= 1.0000009e+00
10 x1= 1.0000000e+00 x2= 9.9999977e-01 x3= 1.0000001e+00
```

Tabelle 7: Iteration des Gauss-Seidel-Verfahrens

6 Das Lösen nichtlinearer Gleichungssysteme

6.1 Das Lösen von Polynomen

Gilt es ein Polynom n-ten Grades zu lösen, dann gibt es zwei Möglichkeiten dies mithilfe eines Iterationsverfahrens zu tun. Man kann versuchen durch geeignete Startwerte alle Nullstellen herauszubekommen (eine graphische Darstellung der Funktion ist dabei sehr hilfreich).
Etwas geschickter ist es jedoch, den Graphen zu vereinfachen, wenn eine Nullstelle gefunden wurde. Man zerlegt das Polynom n-ten Grades in ein Polynom (n-1)-ten Grades, indem man durch einen Linearfaktor teilt, der an der errechneten Nullstelle den Wert null annimmt. Diese Division kann z.B mit einer Polynomdivision durchgeführt werden. Jetzt wird das Iterationsverfahren auf das Polynom (n-1)-ten Grades angewendet. Führt man dieses immer weiter fort, erhält man alle Lösungen. Diese Vorgehensweise wird als Deflation bezeichnet.[30]
Die Vorteile dieses Vorgehens liegen in der einfacheren Form, die das Polynom (n-1)-ten Grades hat. Dadurch ist es einfacher, geeignete Startwerte zu finden. Die Berechnung der Funktionswerte und gegebenenfalls der Ableitung der Funktion vereinfacht sich erheblich.

Als Beispiel wird die Funktion $f(x) = x^4 - x^3 - 16x^2 + 4x + 48$ besprochen. Anhand ihres Graphen kann man erkennen, dass eine Nullstelle zwischen $x = 3$ und $x = 5$ liegt. Wendet man das Newton-Verfahren mit dem Startwert 5 auf diese Gleichung an, erhält man mit einem relativen Fehler von weniger als 10^{-6} nach 6 Iterationen den Wert $x = 4$. Natürlich könnte man hier auch problemlos das Regula falsi-Verfahren oder jedes andere besprochene Verfahren zum Lösen verwenden.
Jetzt führt man eine Polynomdivision für $x \neq 4$ durch:

$$x^4 - x^3 - 16x^2 + 4x + 48 : (x - 4) = x^3 + 3x^2 - 4x - 12 \tag{6.1}$$

Lässt man sich den Graphen $x^3 + 3x^2 - 4x - 12$ zeichnen, erkennt man eine Nullstelle in der Umgebung von $x = 2$. Das Newtonverfahren liefert auch tatsächlich mit dem Startwert $x = 2,5$ den Wert 2.

[30] [1]

Nach einer Polynomdivision erhält man für $x \neq 2$:

$$x^3 + 3x^2 - 4x - 12 : (x - 2) = x^2 + 5x + 6 \tag{6.2}$$

Diesen Term kann man nun mit der Lösungsformel für quadratische Gleichungen lösen und erhält die beiden letzen Nullstellen bei $x = -3$ und $x = -2$.

6.2 Lösen nichtlinearer Gleichungssysteme mit n Gleichungen

Gegeben sei das nichtlineare Gleichungssystem mit n Gleichungen[31]:

$$\begin{aligned} f_1(x_1, x_2, \cdots, x_n) =& 0 \\ f_2(x_1, x_2, \cdots, x_n) =& 0 \\ & \vdots \\ f_n(x_1, x_2, \cdots, x_n) =& 0 \end{aligned} \tag{6.3}$$

Wendet man die Newton-Iterationsformel auf dieses Gleichungssystem an, muss man beachten, dass es sich dann um Vektoren handelt. Also ist $\vec{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}$ $\vec{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ $\Delta\vec{x} = \begin{pmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{pmatrix}$. Stellt man die Newton Iterationsvorschrift wie folgt um:

$$\vec{x}_{n+1} = \vec{x}_n - \frac{\vec{f}(x)}{\vec{f}'(x)} \tag{6.4}$$

$$\vec{f}'(x) \cdot (\vec{x}_{n+1} - \vec{x}_n) = -\vec{f}(x) \tag{6.5}$$

Und setzt man $\Delta\vec{x} = \vec{x}_{n+1} - \vec{x}_n$, ergibt sich:

$$\vec{f}'(x) \cdot \Delta\vec{x} = -\vec{f}(x) \tag{6.6}$$

Ein Problem stellt jetzt die Ableitung von $\vec{f}$ dar. Mit der Jacobi-Matrix lässt sich die Ableitung allerdings sehr leicht berechnen.
Mit $\vec{f}'(x) = D_f$ erhalten wir die Gleichung:

$$D_f \cdot \Delta\vec{x} = -\vec{f}(x) \tag{6.7}$$

[31] [5]

Wenn man jetzt Startwerte für $x_1, \ldots, x_n$ in die Funktionen einsetzt, erhält man $-\vec{f}(x)$. Setzt man diese Startwerte auch in die Jacobi-Matrix ein, dann ist nur noch $\Delta\vec{x}$ in der Gleichung 6.7 unbekannt.
Jetzt hat man zwei Möglichkeiten, diese Gleichung zu lösen. Man kann die inverse Matrix von D_f bestimmen und diese mit $-\vec{f}(x)$ multiplizieren. Der Term würde dann wie folgt aussehen:

$$\Delta\vec{x} = D_f^{-1} \cdot (-\vec{f}(x)) \tag{6.8}$$

Allerdings wird diese Möglichkeit hier nicht näher vorgestellt, da es schwierig ist, eine inverse Matrix zu bestimmen, da ein dreifacher Rechenaufwand bewältigt werden muss.[32] Falls man aber ohne Computer rechnet, kann es hilfreich sein die Gleichung 6.8 zu lösen. Die Rechnung bleibt dann übersichtlich, da sie direkt zum Ergebnis führt.
Die zweite Möglichkeit besteht darin, $D_f \cdot \Delta\vec{x}$ auszumultiplizieren und das erhaltene lineare Gleichungssystem mit einem Verfahren aus Kapitel 5 zu lösen. Das Ergebnis für $\Delta\vec{x}$ setzt man in die Gleichung

$$\vec{x}_{n+1} = \Delta\vec{x} + \vec{x}_n \tag{6.9}$$

ein und erhält neue Startwerte. Nun errechnet man damit wieder $-\vec{f}(x)$ und D_f und führt das Verfahren solange durch, bis sich die Lösungen nur noch wenig ändern.
Da in unserer Zeit die meisten Rechnungen von Computern berechnet werden, ist es geschickter, das lineare System aus Gleichung 6.7 zu lösen, da man nicht darauf angewiesen ist einen möglichst direkten Weg für die Rechnung zu finden. Ist der Rechenweg einmal programmiert führt der Computer diesen beliebig oft durch. Außerdem ist der Rechenaufwand bei dieser Methode erheblich kleiner.[33]

Als Beispiel soll folgendes Gleichungssystem dienen:

$$\begin{aligned} f_1 &= x^2 + y + z - 4 \\ f_2 &= x + y^2 + z - 6 \\ f_3 &= x + y + z^2 - 4 \end{aligned} \tag{6.10}$$

[32] [7] S.116
[33] [7] S.116 und S.119

Die Jacobi-Matrix von diesem System lautet:

$$D_f = \begin{pmatrix} 2x & 1 & 1 \\ 1 & 2y & 1 \\ 1 & 1 & 2z \end{pmatrix} \tag{6.11}$$

Diese Matrix wird mit den Startwerten des linearen Gleichungssystems (Δx_1, Δx_2 und Δx_3) multipliziert. Da das Programm 16 auf Seite 42 mit den Werten a,b und c rechnet setzen wir $a = \Delta x_1,\ b = \Delta x_2$ und $c = \Delta x_3$ ein.

$$D_f \cdot \Delta\vec{x} = -\vec{f} \tag{6.12}$$

$$\begin{pmatrix} 2x & 1 & 1 \\ 1 & 2y & 1 \\ 1 & 1 & 2z \end{pmatrix} \cdot \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} -f_1 \\ -f_2 \\ -f_3 \end{pmatrix} \tag{6.13}$$

Mit dem Falkschen Schema errechnet man:

$$\begin{aligned} 2x \cdot a + b + c &= -f_1 \\ a + 2y \cdot b + c &= -f_2 \\ a + b + 2z \cdot c &= -f_3 \end{aligned} \tag{6.14}$$

Löst man dieses System nach a,b und c auf, erhält man:

$$\begin{aligned} a &= \frac{-f_1 - b - c}{2x} \\ b &= \frac{-f_2 - a - c}{2y} \\ c &= \frac{-f_3 - a - b}{2z} \end{aligned} \tag{6.15}$$

Jetzt muss man nur noch die Werte einsetzen und das Gleichungssystem lösen. Für die Funktionswerte $\vec{f}$ ergibt sich mit den Startwerten für das nichtlineare Gleichungssystem $x = y = z = 4$:

$$\begin{aligned} f_1 &= 4^2 + 4 + 4 - 4 = 20 \\ f_2 &= 4 + 4^2 + 4 - 6 = 18 \\ f_3 &= 4 + 4 + 4^2 - 4 = 20 \end{aligned} \tag{6.16}$$

Außerdem wählen wir $a, b, c = 1$. Das Gleichungssystem 6.14 erfüllt für diese

Startwerte das Zeilensummenkriterium (5.4).

$$\begin{aligned} a &= \frac{-20-1-1}{8} = -2,75 \\ b &= \frac{-18+2,75-1}{8} = -2,03125 \\ c &= \frac{-20+2,75+2,03125}{8} = -1,90234 \end{aligned} \tag{6.17}$$

Nach einigen Iterationen erhält man:

$$\begin{aligned} a &= -\,2,0285714 \\ b &= -\,1,7428571 \\ c &= -\,2,0285714 \end{aligned} \tag{6.18}$$

Mit der Gleichung 6.9 erhalten wir verbesserte Werte für x_1, x_2 und x_3

$$\begin{aligned} x_1^{(2)} &= a + x_1^{(1)} = -\,2,0285714 + 4 = 1,9714286 \\ x_2^{(2)} &= b + x_2^{(1)} = -\,1,7428571 + 4 = 2,2571429 \\ x_3^{(2)} &= c + x_3^{(1)} = -\,2,0285714 + 4 = 1,9714286 \end{aligned} \tag{6.19}$$

Das Programm 16 auf Seite 42 führt diese Rechnungen solange durch, bis zwischen den neu errechneten Werten von x_1, x_2 und x_3 und den vorhergehenden Werten nur noch ein relativer Fehler von 10^{-16} besteht. Das Programm liefert die Lösungen $x_1 = 1, x_2 = 2$ und $x_3 = 1$.

```
1 x1= 4.0000000e+00 x2= 4.0000000e+00 x3= 4.0000000e+00
2 x1= 1.9714286e+00 x2= 2.2571429e+00 x3= 1.9714286e+00
3 x1= 1.2064585e+00 x2= 1.9231784e+00 x3= 1.2064585e+00
4 x1= 1.0142038e+00 x2= 1.9941487e+00 x3= 1.0142038e+00
5 x1= 1.0000764e+00 x2= 1.9999703e+00 x3= 1.0000764e+00
6 x1= 1.0000000e+00 x2= 2.0000000e+00 x3= 1.0000000e+00
7 x1= 1.0000000e+00 x2= 2.0000000e+00 x3= 1.0000000e+00
8 x1= 1.0000000e+00 x2= 2.0000000e+00 x3= 1.0000000e+00
```

Tabelle 8: Newton-Verfahren

Für dieses Gleichungssystem gibt es noch andere Lösungen. Diese erhält man, wenn z.B andere Startwerte gewählt werden. Allerdings wird hier nicht weiter darauf eingegangen.

7 Zusammenfassung

Als Abschluss lässt sich feststellen, dass es viele verschiedene Möglichkeiten gibt, mithilfe von Iterationsverfahren Gleichungen zu lösen. Die hier vorgestellten Verfahren stellen die bekanntesten Vertreter dieser Gruppen dar, sind aber bei weitem noch nicht alle. Es hängt von den jeweiligen Anforderungen und Zielen ab, welches Verfahren oder welche Kombination von Verfahren man anwendet.
Diese Arbeit sollte einen Einblick in die Iterationsverfahren Regula falsi und Newton gegeben, um damit nichtlineare Gleichungen zu lösen. Es würde den Rahmen der Arbeit sprengen, eine Fehlerbetrachtung und -analyse vorzunehmen. Auch wurde hier nicht genau auf den Rechenaufwand bei heutigen Computern eingegangen. Es wurde auch kein Bezug auf heutige Computer Algebra Systeme (CAS) wie z.B. Mathematika oder Maple genommen. Diese Programme sind zum Berechnen von Termen besser geeignet als Pascal. Sie können Ableitungen für eine ganze Funktion berechnen, während man bei Pascal die Ableitung angeben muss und die Programmiersprache somit nur die Ableitung an einem bestimmten Punkt berechnen kann. Auch Jacobi-Matrixen können diese Systeme selbstständig berechnen.

A Pascal-Programme

An dieser Stelle werden die Pascal-Programme aufgeführt, die erstellt wurden, um die Iterationsverfahren anhand von Beispielen darzustellen. Die Sprache Pascal wurde gewählt, da sie auch für Laien leicht verständlich ist.

```
program regulafalsi (output);
{Loesen der Gleichung 1/4*x^2-3
mit den Startwerten xm=5 und xn=-2
und einem relativen Fehler von 10^(-16)}
    var
      a,xn,fxn,xalt : real;
                  i : integer;
    const
      xm =5;
      fxm=3.25;
begin
    {Startwerte werden zugewiesen}
    xn:=-2;
    fxn:=-2;
    xalt:=-10;
    for i:=1 to 30 do
     begin
     {Abbruchkriterium}
      if Abs((xn-xalt)/(xn))>= exp(-16*ln(10)) then
       begin
        xalt:=xn;
writeln('x',i:2,'=',xn:22,' f(x',i:2,')=',fxn);
    {Der Neue Funktionswert wird berechnet}
          xn:=xn-fxn*(xn-xm)/(fxn-fxm);
         fxn:=0.25*xn*xn -3;
       end;
     end
end.
```

Abbildung 11: Regula falsi-Verfahren für $f(x) = \frac{1}{4}x^2 - 3$

```
program sekanten (output);

{Loesen der Gleichung 1/4*x^2-3
mit den Startwerten xm=5 und xn=-2
und einem relativen Fehler von 10^(-16)}
    var
    xn,fxn : real;
    xalt,fxalt,a : real;
    i: integer;
begin
    {Startwerte werden zugewiesen}
     xalt := 5;
    fxalt := 3.25;
       xn :=-2;
      fxn :=-2;
 for i:=1 to 40 do
  begin
     {Abbruchkriterium}
   if Abs((xn-xalt)/xn)>=exp(-16*ln(10)) then
    begin
     a:=xn-fxn*(xn-xalt)/(fxn-fxalt);
    {Aktuelle Werte werden nun alt,
     da es bessere Naeherungen gibt}
     xalt:=xn;
    fxalt:=fxn;
 writeln('x',i:2,'=',xn:22,' f(x',i:2,')=',fxn);
    {Der Neue Funktionswert wird berechnet}
        xn:=a;
       fxn:=0.25*xn*xn -3;
    end;
  end
end.
```

Abbildung 12: Sekanten-Verfahren für $f(x) = \frac{1}{4}x^2 - 3$

```
program NetwonVerfahren (output);

{Loesen der Gleichung 1/4*x^2-3
mit dem Startwert xn=5
und einem relativen Fehler von 10^(-16)}
var
        xalt, xn,fxn : real;
                 i : integer;

begin
        {Startwert wird festgelegt}
    xn:=5;
    xalt:=0;
    i:=1;

repeat
    begin
       fxn:=0.25 *xn*xn -3;
  writeln('x',i:2,'=',xn:22,' f(x',i:2,')=',fxn);
        xalt:=xn;
          xn:=xn-(fxn )/(0.5*xn);
        i:=i+1
     end
until
     {Abbruchkriterium}
      Abs(xn-xalt)/xn<=exp(-16*ln(10))
end.
```

Abbildung 13: Newton-Verfahren für $f(x) = \frac{1}{4}x^2 - 3$

```
program Jacobi (output);
var
            x1,x2,x3  : real;
   x1alt,x2alt,x3alt  : real;
               a,b,c  : real;
                   i  : integer;

Procedure berechnen;
begin
        x1alt:=x1;
        x2alt:=x2;
        x3alt:=x3;
        a:=(7-1*x2-2*x3)/4;
        b:=(7-2*x1-1*x3)/4;
        c:=(3-1*x2)/2;
writeln(i:2,' x1=',x1:14,' x2=',x2:14,' x3=',x3:14);
        x1:=a;
        x2:=b;
        x3:=c;
       end;

begin
       x1:=100;
       x2:=-30;
       x3:=-50;
       x1alt:=0;
       x2alt:=0;
       x3alt:=0;
       i:=1;
 repeat
    begin
     berechnen;
     i:=i+1;
    end
    until (Abs((x1-x1alt)/x1alt)<=exp(-6*ln(10))) and
        (Abs((x2-x2alt)/x2alt)<=exp(-6*ln(10)))and
        (Abs((x3-x3alt)/x3alt)<=exp(-6*ln(10)))
end.
```

Abbildung 14: Jacobi-Algorithmus

```
program gauss (output);
var
            x1,x2,x3  :  real;
  x1alt,x2alt,x3alt  :  real;
              a,b,c  :  real;
                  i  :  integer;

Procedure berechnen;
    begin
writeln(i:2,' x1=',x1:14,' x2=',x2:14,' x3=',x3:14);
      x1alt:=x1;
      x2alt:=x2;
      x3alt:=x3;
      x1:=(7-1*x2-2*x3)/4;
      x2:=(7-2*x1-1*x3)/4;
      x3:=(3-1*x2)/2;
    end;

begin
    {Startwerte}
      x1:=100;
      x2:=-30;
      x3:=-50;
      x1alt:=0;
      x2alt:=0;
      x3alt:=0;
      i:=1;
repeat
    begin
      berechnen;
      i:=i+1;
    end
   until (Abs((x1-x1alt)/x1alt)<=exp(-6*ln(10))) and
      (Abs((x2-x2alt)/x2alt)<=exp(-6*ln(10))) and
      (Abs((x3-x3alt)/x3alt)<=exp(-6*ln(10))) ;
end.
```

Abbildung 15: Gauss-Seidel-Algorithmus

```
program newtonnlgs (output);
var
    x1,x2,x3, fx1,fx2,fx3: real;
    x1alt,x2alt,x3alt: real;
    d1fx1,d1fx2,d1fx3: real;
    d2fx1,d2fx2,d2fx3: real;
    d3fx1,d3fx2,d3fx3: real;
                a,b,c: real;
                y,z,i: integer;
procedure lgsberechnen;
begin
    for z:=1 to 20 do
        begin
          a:=(-fx1-d2fx1*b-d3fx1*c)/d1fx1;
          b:=(-fx2-d1fx2*a-d3fx2*c)/d2fx2;
          c:=(-fx3-d1fx3*a-d2fx3*b)/d3fx3;
        end;
          x1alt:=x1; x2alt:=x2; x3alt:=x3;
          x1:=x1+a;  x2:=x2+b;  x3:=x3+c;
end;
begin
        x1:=4;      x2:=4;      x3:=4;
        x1alt:=50; x2alt:=50; x3alt:=50;
        a:=1;       b:=1;       c:=1;
        i:=1;
repeat
    begin
      fx1:=x1*x1+x2+x3-4;
      fx2:=x1+x2*x2+x3-6;
      fx3:=x1+x2+x3*x3-4;
    d1fx1:= 2*x1; d2fx1:= 1;    d3fx1:= 1;
    d1fx2:= 1;    d2fx2:= 2*x2; d3fx2:= 1;
    d1fx3:= 1;    d2fx3:= 1;    d3fx3:= 2*x3;
writeln(i:2,' x1=' ,x1:14,' x2=',x2:14,' x3=',x3:14);
        lgsberechnen; i:=i+1;
    end
until (Abs((x1-x1alt)/x1alt)<=exp(-16*ln(10)))
    and (Abs((x2-x2alt)/x2alt)<=exp(-16*ln(10)))
    and (Abs((x2-x2alt)/x2alt)<=exp(-16*ln(10)));
end.
```

Abbildung 16: Newton-Verfahren für nichtlineare Gleichungssysteme

Literatur

[1] Dr. Graham Brindley and Dr. Christian Beardah. Numerical Methods 1, Nottingham Trent University, Burton Street, Nottingham, 1998, http://science.ntu.ac.uk/student/milanl/lectures/lect7.htm – Zugriff am 17.01.03.

[2] Greg Fasshauer. Soluton of Nonlinear Equations, http://amadeus.math.iit.edu/~fass/577_ch3.pdf.

[3] Dr. Friedrich Nikol und Karl Wörle Friedrich Barth, Paul Mühlbauer. Mathematische Formeln und Definitionen. Bayerischer Schulbuch-Verlag, Rosenheimer Str. 145, 81671 München, 1998.

[4] Wilfried Imrich. Skriptum zu Computereinsatz in den technischen Wissenschaften, 2001, http://www.unileoben.ac.at/~amat/lehrbetrieb/num/skriptum/node10.html – Zugriff am 17.01.03.

[5] Wilfried Imrich. Skriptum zu Computereinsatz in den technischen Wissenschaften, 2001, http://www.unileoben.ac.at/~amat/lehrbetrieb/num/skriptum/node9.html – Zugriff am 17.01.03.

[6] G. Musiol und H Mühlig I.N. Bronstein, K.A. Semendjaev. Taschenbuch der Mathematik. Verlag Harri Deutsch, Thun und Frankfurt am Main, 5. Auflage, 2001.

[7] Ben Noble. Numerisches Rechnen I. Bibliographisches Institut Mannheim /Wien /Zürich Hochschultaschenbücher Verlag, 1966.

[8] Felix Riedel. Näherungsverfahren zur Nullstellenermittlung, Abtei-Gymnasium Brauweiler, 2001.

[9] Gregor Snelting. Vorlesung über Elementare Programmierung, Informatik Sommercamp 2002, Universität Passau, 2002.

[10] Dr. Kurt Meyberg und Dr. Peter Vachenauer. Höhere Mathematik 1. Springer-Verlag, Berlin Heidelberg New York, 4. Auflage, 1997.

[11] Gerhard Merziger und Thomas Wirth. Repetitorium der Höheren Mathematik. Binomi Verlag, Am Bergfelde 28, 31832 Springe, 1991.

Erklärung zur Facharbeit des Schülers

Dominik Gunreben

Ich erkläre hiermit,

dass ich diese Facharbeit ohne fremde Hilfe angefertigt und nur die im Literaturverzechnis angeführten Quellen und Hilfsmittel benutzt habe.

Oberstreu, den 1. Februar 2003

. . . . . .

Dominik Gunreben