Numerische Methoden

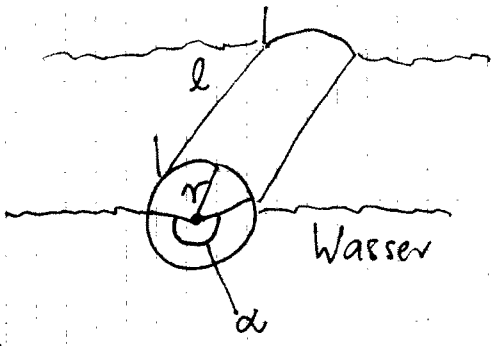
17.02.2008

I Gleichungen

Ziel: Lösen von Gld.

Bsp 1: Berechne Temperatur T aus der Energie E in Circuit Breaker

Bsp 2: Schwimmender Balken



Länge : l
Radius : r
spez. Gewicht Holz: ϱ_1
spez. " Wasser: ϱ_0
Winkel: α
Gewicht: G

▷ Archimedes:

$$G = \pi r^2 l \varrho_1 = \frac{r^2}{2}\alpha + r^2 \sin\left(\pi - \frac{\alpha}{2}\right)\cos\left(\pi - \frac{\alpha}{2}\right)\Big] l \varrho_0$$

$$| \cdot \frac{2}{r^2 \varrho_0}$$

$$\Rightarrow \quad \alpha - \sin(\alpha) = 2\pi\varrho \quad \text{mit} \quad \varrho = \frac{\varrho_1}{\varrho_0} \in (0,1)$$

▷ Fixpunktform

$\alpha = g(\alpha)$ — g ist bekannt; hier:

$\alpha = \sin(\alpha) + 2\pi\varrho$ — $g(\alpha) = \sin(\alpha) + 2\pi\varrho$

1. Iteration

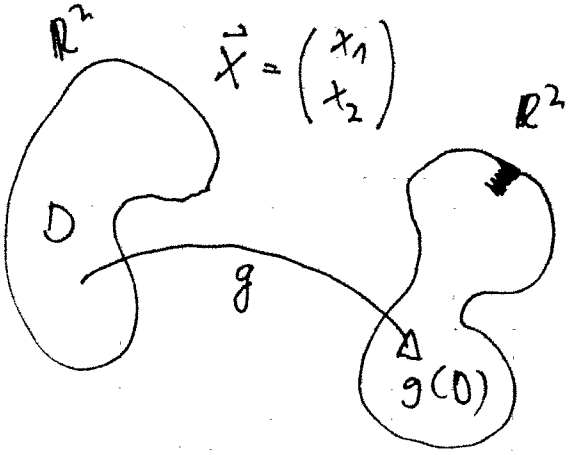
1.1 Fixpunkte

Def.bereich — Bildbereich

Gegeben: Fkt. $g : x \in D \longmapsto g(x) \in g(D)$

1 dim: $D = [a, b]$

2 dim:



▷ Def: $\xi \in D$ heisst Fixpkt.

Falls: $\xi = g(\xi)$

▷ Fixpktiteration:

Wähle $x_0 \in D$

$x_{k+1} = g(x_k)$ für $k = 0, 1, 2, \ldots$

Man benötigt: $g(D) \subset D$

Klar: Falls die Folge $\{x_k\}$ konvergiert gilt:

$\xi = \lim_{k \to \infty} x_k$ ist

Fixpkt von g.

▷ Zum Bsp. 2: Tannenholz: $\varrho = \frac{\delta_1}{\delta_0} = \frac{5}{6}$

$g(\alpha) = 2\pi \cdot \frac{5}{6} + \sin(\alpha)$

Startwert: z. Bsp. $\alpha = 0$

$\Rightarrow \quad \alpha_1 = 5.236\ldots$

$\alpha_2 = 4,370\ldots$

$\vdots$

$\alpha_{23} = 4,314216594$

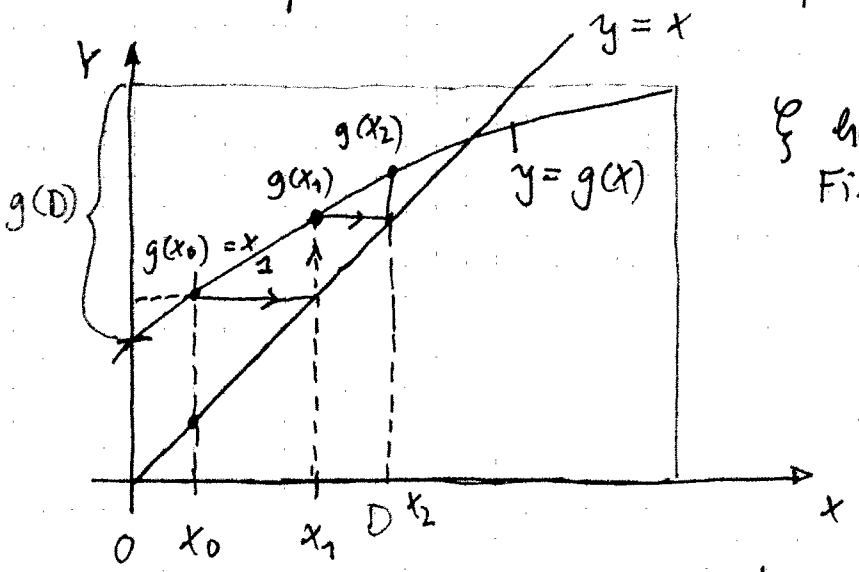
$\alpha_{24} = 4,314216594$

$\alpha_{24} \;=\; \text{Fixpunkt}$

$\alpha \cong 247°$

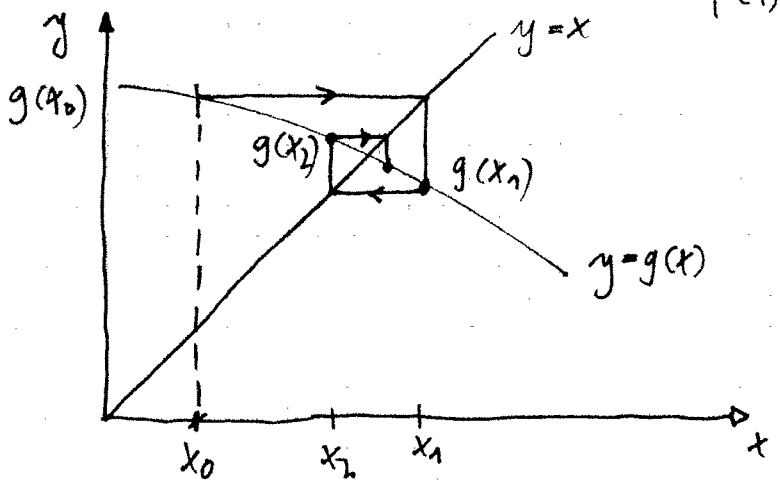
▷ 1 dim: Graphisch

$f'(x) > 0$



ξ heisst anziehender Fixpkt.

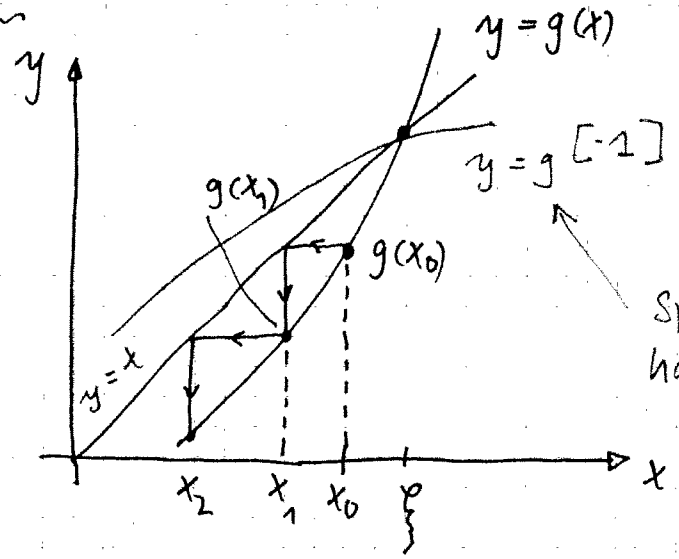
$f'(x) < 0$



ξ ist auch anziehend

$\xi = \lim_{k \to \infty} x_k$

Aber:



Abstossend bzw. instabie

Spiegelung an der Winkelhalbierenden.

Fixpunkt: Die Iteration $x_{k+1} = g(x_k)$ konvegiert nicht gegen ξ.

Hingegen konvergiert $x_{k+1} = g^{[-1]}(x_k)$ für $k = 0, 1, \ldots$

$g(x_{k+1}) = x_k$ (Rückwärtsiteration)

1.2. Fehlertheorie

Absoluten Fehler von x_k bzg. ξ

$$e_k := x_k - \xi$$

wobei $x_{k+1} = g(x_k)$

Idee: e_{k+1} durch e_k ausdrücken!

Taylorentwicklung im Punkt ξ

$$\Rightarrow e_{k+1} = x_{k+1} - \xi = g(e_k + \xi) - \xi$$

$$= g(\xi) + g'(\xi) e_k + \frac{1}{2} g''(\xi) e_k^2 - g(\xi)$$

$$\Rightarrow e_{k+1} = g'(\xi)\, e_k + \mathcal{O}(e_k^2)$$

▷ Asymptotisches Fehlergesetz

Ann: $x_k \to \xi$, $\{x_k\}$ konvergiert

$\Rightarrow \xi$ ist attraktiver Fixpunkt:

$e_k \to 0$ mit $e_{k+1} = g'(\xi)\, e_k + \mathcal{O}(e_k^2)$

▷ Def: Lineare Konvergenzgeschwindigkeit

Sei $\{x_k\}$ konvergent mit $\lim_{k\to\infty} x_k = \xi$,

$e_k = x_k - \xi$

$\{x_k\}$ heisst linear konvergent falls

$$\left|\frac{e_{k+1}}{e_k}\right| \le K < 1 \quad \text{für } k > N.$$

$$\lim_{k\to\infty} \left|\frac{e_{k+1}}{e_k}\right| = q < 1 \quad \text{mit } q := \text{Konvergenz-faktor}$$

▷ Zusammenfassung:

ξ Fixpunkt von g, d.h. $\xi = g(\xi)$, x_0 Startwert genügend nahe bei ξ. Fixpunktiteration

$$x_{k+1} = g(x_k)$$

▷ 2 Hauptfälle: (i) $|g'(\xi)| < 1 \Leftrightarrow \xi$ attraktiv

(ii) $|g'(\xi)| > 1 \Leftrightarrow \xi$ abstossend

▷ Bemerkung: Sei ξ abstossend, d.h. $|g'(\xi)| > 1$

dann gilt:

$$|g^{[-1]'}| = \left|\frac{1}{g'(\xi)}\right| < 1$$

ξ ist anziehend von Funktion $g^{[-1]}$

1.3. Banachscher Fixpunktsatz

▷ Satz: Annahme:

(1) Sei $g(D) \subset D$ (Selbstabbildung)

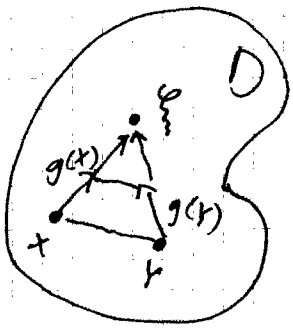
(2) Sei D abgeschlossen (Ränder sind noch in D enthalten)

g ist stetig in D

(3) g ist kontrahierend, d.h. es gibt eine Zahl L mit $0 < L < 1$, mit $x, y \in D$

$$|g(y) - g(x)| \leq L|x - y| \quad (*)$$

Bsp: 2 dim:



Behauptung:

g hat genau einen Fixpunkt ξ. Die Fixpunktiteration konvergiert nach ξ für jeden beliebigen Startwert $x_0 \in D$

Es gilt:

$$|x_k - \xi| \leq \frac{L^k}{1-L} |x_0 - x_1|$$

▷ Bemerkung 1:

(*) heisst Lipschitzbedingung, $L < 1$ garantiert Kontraktion, L heisst Lipschitzkonstante.

$$L = \max_{\substack{x,y \in D \\ x \neq y}} \left| \frac{g(y) - g(x)}{y - x} \right|$$

Maximale Sekantensteigung.

▷ Bemerkung 2:

Konvergenzgeschwindigkeit nahe bei ξ ist $L \cong g'(\xi)$

1.4. Newtonverfahren

Zweite Standardform einer Gleichung

$$f(x) = 0$$

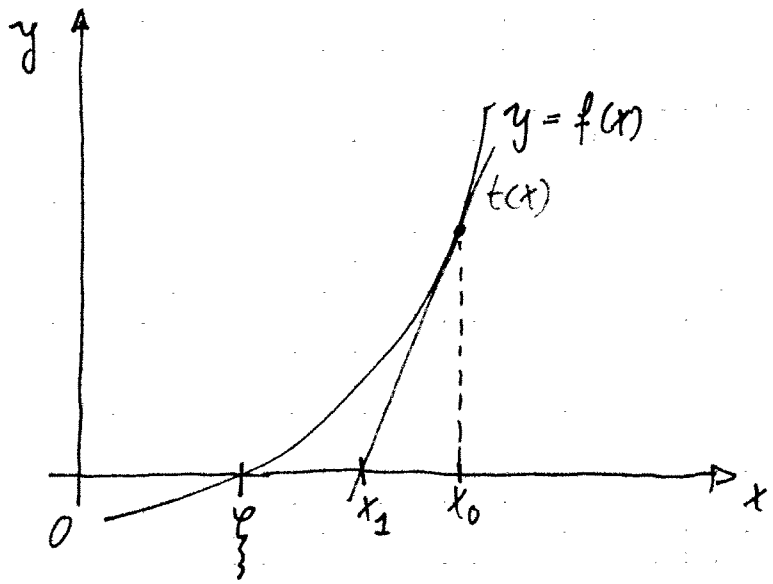
Gegeben: f ist differenzierbar

$$f: x \longmapsto f(x)$$

Gesucht: Nullstelle ξ; d.h. $f(\xi) = 0$

Nichtlineare Gleichung.

Idee: ersetze f durch lineare Funktion



$$t(x) = f(x_0) + f'(x_0)(x - x_0)$$

x_1 sodass $t(x_1) = 0$ linearisieren !!!

$$f(x_0) + f'(x_0)(x_1 - x_0) = 0$$

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$ Vorraussetzung: $f(x_0) \neq 0$

25.02.08

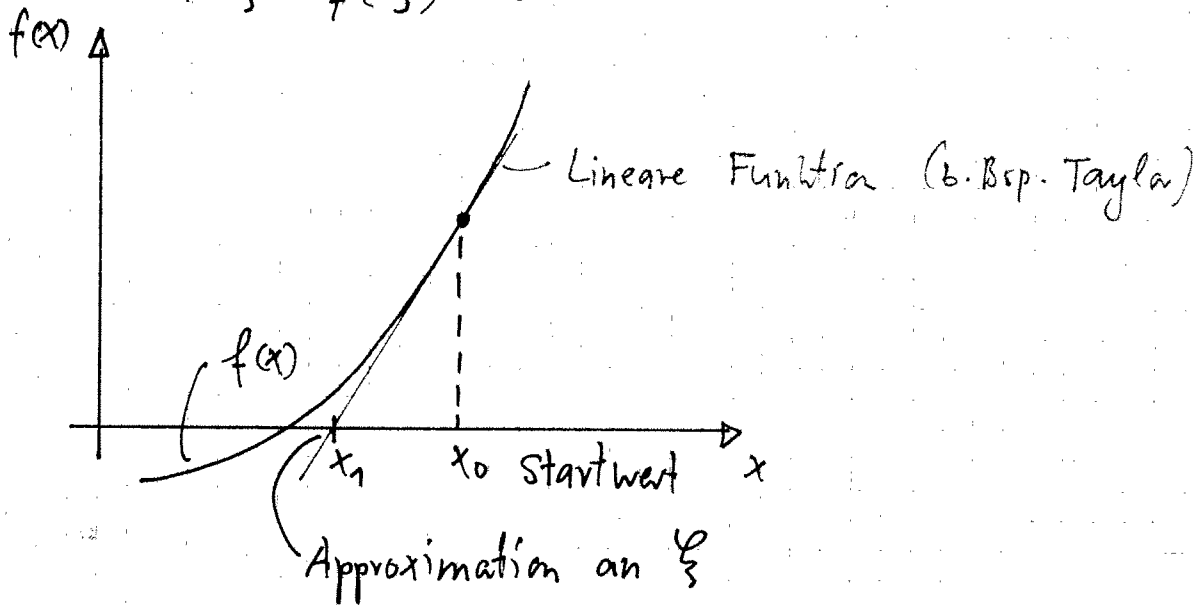
Newtonverfahren

1) $x \stackrel{?}{=} \cos(x)$; $f(x) = x - \cos(x)$

Nst von $f(x)$

2) $x^2 = 2$; $x = \sqrt{2}$; $f(x) = x^2 - 2$

Nst ξ $f(\xi) = 0$



Newton-Algorithmus

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$; x_0 ist Schätzer

$k = 0, 1, 2, \ldots$

Bsp: $f(x) = x^2 - 2$; $f'(x) = 2x$

$$x_{k+1} = x_k - \frac{x_k^2 - 2}{2x_k} = \frac{1}{2}x_k + \frac{1}{x_k} \qquad \forall\, x_k \neq 0$$

$x_0 = 2$

$x_1 = \frac{3}{2} = 1{,}5$

$x_2 = 1{,}41666666\ldots$

$x_3 = 1{,}41421568\ldots$

$x_4 = 1{,}4142135623\,7\,4$

Wie verringert sich der Fehler allgemein? (Konvergenztheorie)

$x_k = \xi + e_k$; e_k : Fehler

$x_{k+1} = \xi + e_{k+1}$

$$x_{k+1} \overset{\text{Verfahren}}{=} (\xi + e_k) - \frac{f(\xi + e_k)}{f'(\xi + e_k)}$$

$$\overset{\text{Taylor Entw.}}{=} \xi + \frac{f''(\xi)}{2f'(\xi)} e_k^2 + O(e_k^3) = \xi + e_{k+1}$$

$$\Rightarrow e_{k+1} = \frac{f''(\xi)}{2f'(\xi)} e_k^2 \quad \left.\right\} e_k = 10^{-3}, e_{k+1} \approx \bullet\, 10^{-6}$$

▷ Definition:

Falls für den Fehler $e_k \overset{k\to\infty}{\to} 0$ eines Verfahrens gilt

$e_{k+1} \leq C e_k^p$

mit C - const (unabhängig von k) und $p \geq 1$,

so heisst das Verfahren konvergent mit Ordnung p.

Bsp: Newton : $p = 2$

allg. Fixpunktiteration : $p = 1$ ($C < 1$: Kontraktionsbedingung)

Abbruchkriterium:

Eigentlich: $|x_{k+1} - \xi| < \delta$; δ: Toleranz

Wir kennen aber ξ nicht!

Ersatz: $|x_{k+1} - x_k| < \delta$ (absolute Fehler)

$$\frac{|x_{k+1} - x_k|}{|x_k|} < \delta \quad \text{(relative Fehler)}$$

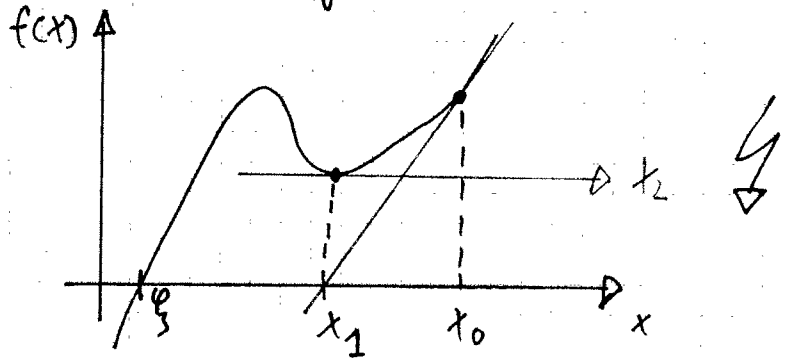
Kombination:

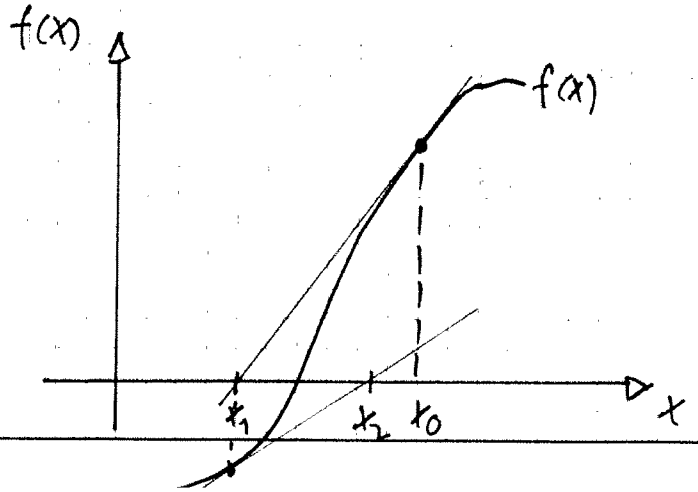
$$|x_{k+1} - x_k| < \delta_{rel} |x_{k+1}| + \delta_{abs}$$

Zusätzlich: Iteration $< 10 \ldots 15$

Konvergenzprobleme:

Newton konvergiert, falls der Schätzer x_0 "nahe genug" bei der Lösung ist. In der Praxis ist das heikel.





(schlechte Zeichnung siehe Skript)

zyklisch!

Oft ist $f'(x)$ mühsam zu berechnen:

Strategien:

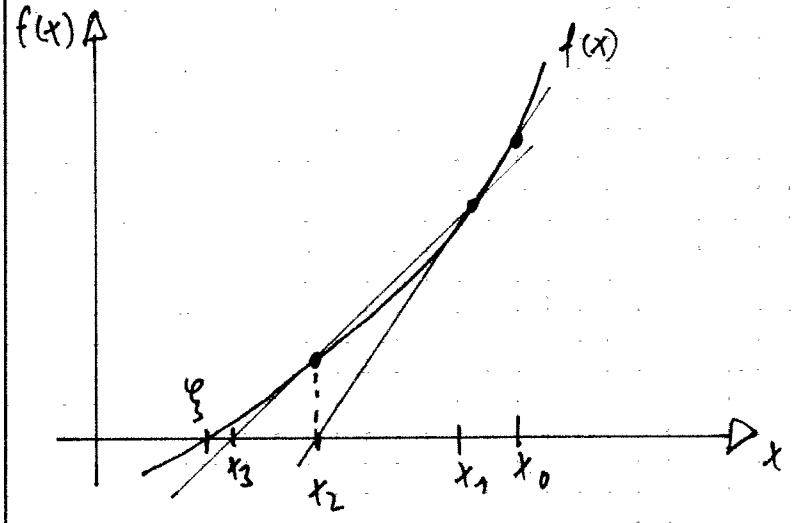
1) Wir approximieren $f'(x) \approx \frac{f(x+h) - f(x)}{h}$ mit h klein.

2) Oder $f'(x) \approx f'(x_0)$ "Quasi-Newton"
für alle h die gleiche Ableitung. $p = 1$

3) $f'(x_k) = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}$ "Sekantenverfahren"

Trick: Wir benützen die Auswertung $f(x_{k-1})$ aus einem früheren Schritt.

Dafür brauchen wir zwei Startwerte. x_0, $x_{-1} = x_0 - h$



Es lässt sich zeigen $p = \frac{\sqrt{5}+1}{2} \simeq 1{,}61\ldots$

Goldene Schnitt!

Also: $p_{Quasi} < p_{Sek.} < p_{Newton}$

Zusätzlich ist Sekanten Verf. effizienter als Newta Verf.

Effizienz pro Schritt $= \frac{\text{Fehlerreduktion}}{\text{Fehlerauswertung}}$ Hier: $E = p^{\frac{1}{m}}$

m: Funktionsauswertung!

Newta $m = 2$ $p = 2$ $E = \sqrt{2}$

Sekante $m = 1$ $p = 1.6$ $E = 1{,}6$

Newton in N Dimensionen

Analog: Nullstelle von $\vec{f}: \mathbb{R}^N \to \mathbb{R}^N$;

$$\underline{x} \mapsto \underline{f}(\underline{x})$$

$$\underline{\xi} = (\xi_1, \xi_2, \dots, \xi_N)$$

d.h. $\underline{f}(\underline{\xi}) = 0$ also

$$f_1(\underline{\xi}) = 0$$
$$f_2(\underline{\xi}) = 0$$
$$\vdots$$
$$f_N(\underline{\xi}) = 0$$

Bsp: $x^2 - y^2 = 0$

$xy + 1 = 0$

$f_1(x,y) = x^2 - y^2$

$f_2(x,y) = xy + 1$

$\underline{\xi} \hat{=} (x,y)$: Lösung: $\underline{\xi} = (1,-1)$ oder $(-1,1)$

Approximation durch lineare Abb. (Taylor) $\rightsquigarrow$ Newton

$$f_i(\underline{x}_0 + \Delta\underline{x}) \overset{\text{Taylor}}{=} f_i(x_0) + \sum_{j=1}^{N} \frac{\partial f_i}{\partial x_j}(x_0)\,\Delta x + O(\|\Delta x\|^2)$$

$$= \underline{f} + J\,\Delta\underline{x} = 0$$

$i = 1, \dots, N$

$\underline{x}_0$ Schätzer

$\overset{!}{=} 0$ (schwierig), $\Delta x = ?$

$\overset{!}{=} 0$ (leicht), $\Delta x = ?$

Jacobi-Matrix: $J = Df = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots \\ \frac{\partial f_2}{\partial x_1} & \ddots & \\ \vdots & & \end{pmatrix}$

Newton: für $k = 0, 1, 2, \ldots$:

i) Löse lineares Gleichungssystem

$$J(x_k)\,\Delta x = -f(x_k) \quad \text{nach } \Delta x$$

ii) $x_{k+1} = x_k + \Delta x$

x_0 Schätzer

Schwierig: J ist teuer zu rechnen

typischerweise mit Differenzenquotient

Marke: ein nichtlineares Gleichungssystem $\rightsquigarrow$ wird zu einer Iteration von lin. Gleichungssystemen

Lineare Gleichungssysteme (LGS)

Die Lsg. von LGS ist Grundbaustein von sehr vielen Berechnungen.

Abstrakt: $Ax = b$

$A \in \mathbb{R}^{N \times N}$; $x, b \in \mathbb{R}^N$

Erinnere: A regulär ($\det \neq 0$)

$\Leftrightarrow$ genau eine Lösung x

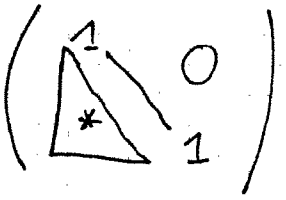
A singulär ($\det = 0$)

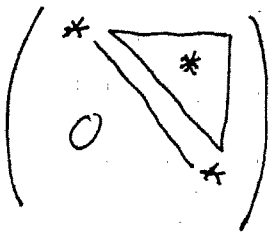
$\Leftrightarrow$ entweder keine oder unendlich viele Lsg.

Klassischer Algorithmus: Gauss-Elimination

numerische Varianten: LR - Zerlegung

Wir zerlegen ("faktorisieren") $A = L \cdot R$, $L, R \in \mathbb{R}^{N \times N}$

$L =$ 

; $R =$ 

Linke untere Dreiecksmatrix

Rechte obere Dreiecksmatrix

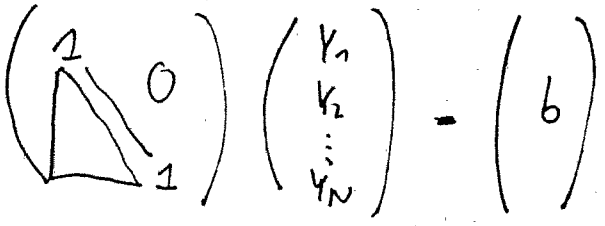
(Englisch: LU - Decompositia)

Wozu ? $Ax = b$

$\longleftrightarrow (LR)\, x = b$

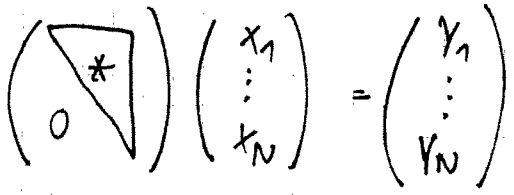
$\underbrace{\quad}_{y}$

Wir lösen $Ly = b$ (leicht!)



"Vorwärts" einsetzen.

$Rx = y$ (auch leicht)



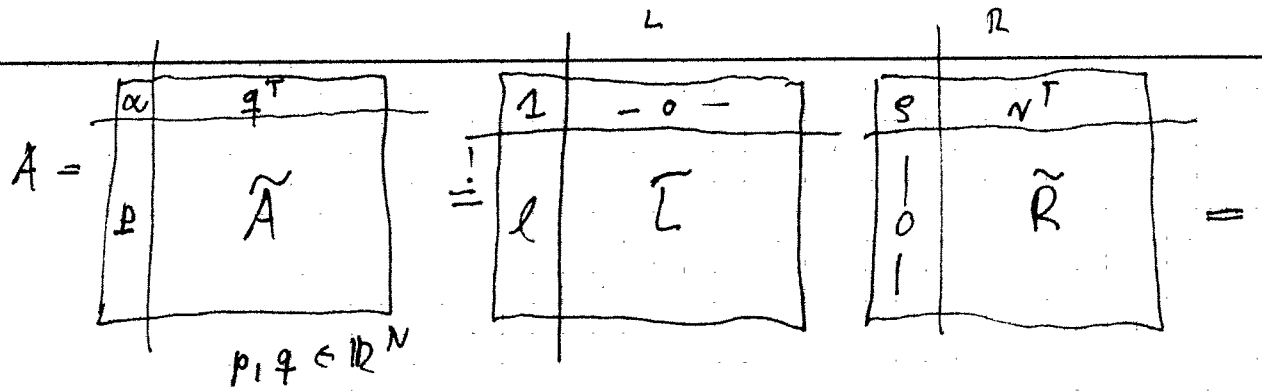
"Rückwärts" einsetzen

Das ist übrigens genau was Gauss macht !

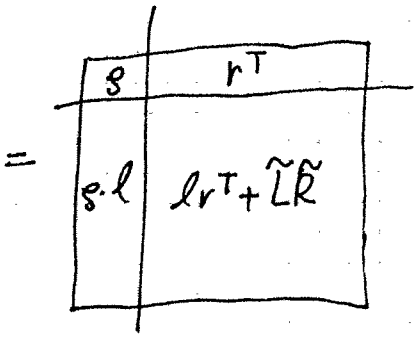
Die Zerlegung ist nützlich falls mehrere rechte Seiten (für)

gelöst werden muss.

$Ax = b_i$ mit $i = 1, 2, \ldots$



Wir iterieren über die Zeilen (Gauss)



$\underbrace{l, r}_{?} \in \mathbb{R}^N \qquad s \in \mathbb{R}$ (?)

Es folgt: $s := \alpha \qquad r := q$

$l := \frac{1}{s} f$

Matlab!

Problem $\alpha \neq 0$ notwendig

Ansonsten Zeilen vertauschen!

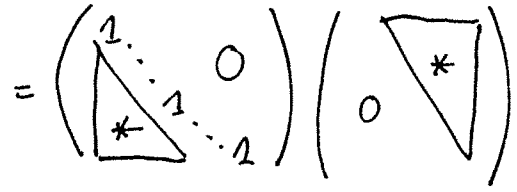
$\rightsquigarrow \quad PA = L \cdot R$

↑

Vertauschungsmatrix

$n \times n$ Matrix

$A = LR =$

$=$ 

Wesentlicher Schritt : rückwärtseinsetzen, dann vorwärtseinsetzen

Man kann mit "Blockmatrizen" rechnen wie mit Matrixelementen.

- Vektoren $\underline{x} = \begin{bmatrix} \\ \\ \end{bmatrix}$ (immer Spaltenvektoren)

$\underline{l} = \frac{1}{\alpha}\underline{f} \Rightarrow \alpha \neq 0$ notwendige Bedingung

↳ Pivot

Zerlege Untermatrix:

$\tilde{A} = \underline{l}\underline{r}^T + \tilde{L}\tilde{R}$

$\tilde{A} - \underline{l}\underline{r}^T = \tilde{L}\tilde{R}$ $(n-1) \times (n-1)$ Matrizen

- $(\tilde{A} - \underline{l}\underline{r}^T)_{11} \neq 0$ notwendig

Gleichss: $A\underline{x} = \underline{b}$

Zeilenvertauschen:

▷ Def: Permutationsmatrix $P \in \mathbb{R}^{n \times n}$

in jeder Zeile, jeder Spalte genau eine 1 sonst Oen.

Eigenschaften: 1)

$$P = \begin{bmatrix} 0\,0\ldots0\,1\,0\ldots0 \\ 0\,0\,0\,0\,1\ldots0\,0 \\ 0\ldots \\ 1\ldots \end{bmatrix}$$

i_1 Position

i_2 Posit.

$$PA = P\begin{pmatrix} \underline{a}_1^T \\ \underline{a}_2^T \\ \vdots \\ \underline{a}_n^T \end{pmatrix} = \begin{pmatrix} \underline{a}_{i_1}^T \\ \underline{a}_{i_2}^T \\ \vdots \end{pmatrix}$$

2) P ist orthogonal

$$P^{-1} = P^T$$

$$J = PP^{-1} = PP^T$$

▷ Satz: Für jede Matrix $A \in \mathbb{R}^{n \times n}$ existieren Permutationsmatrizen und Faktoren L und R mit

$$PA = LR$$

MATLAB: $[L, R, P] = lu(A)$ liefert die Matrizen L, P, und R

Bemerkungen:

i) A „singulär", Rang $r < n$

Es gibt: $PA = LR$

mit $\begin{pmatrix} R & \\ 0 & 0 \end{pmatrix}\}r$

ii) Determinante

= Volumen des Parallelflachs das von den Spaltenvektoren aufgespannt wird.

$$\det(A) = \det(PLR) = \underbrace{\det(P)}_{(-1)^\nu} \underbrace{\det(L)}_{1} \det(R)$$

$$= (-1)^\nu \cdot \prod_{i=1}^{n} r_{ii} \quad ; \quad \nu \equiv \text{Anzahl der Vertausch. } P$$

(iii) Vorteile der LR Zerlegung:

- Speicherung von L, R und P an Ort und Stelle in A zusammen mit Vektor der Länge n für P
- Zerlegungsaufwand:

$$\mu_{LR} = \frac{n^3}{3} + O(n^2)$$ Multiplik./Divis.

$$\mu_{solve} = n^2$$

$$Ax = b$$

$$x = A^{-1}b$$ Aufwand ebenfalls n^2

Mehrere rechte Seiten mit derselben Matrix:

$$X = [x_1, x_2, \ldots, x_L] \; ; \; B = [b_1, b_2, \ldots, b_L]$$

(iv) Ist A dünnbesetzte Matrix, so ist A^{-1} vollbesetzt (sparse)

sparse: Anzahl von 0 verschiedenen Elementen ist $k \cdot n$

Bei dünnbesetzten Matrizen sind i.a. die Faktoren L, R auch dünnbesetzt.

(E-Technik:

f kompakte Träger, Fouriertransf. geht von $-\infty, \infty$)

2.2. Kondition, Norm

$A\underline{x} = \underline{b}$; $\underline{b}, \underline{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^n$

A kann fast singulär sein.

Im Allgemeinen ist die Lösung $\tilde{x}$ dann ungenau.

Residuum: $\underline{r} := A\tilde{\underline{x}} - \underline{b}$

$\tilde{x}$ kann ungenau sein auch wenn $\underline{r}$ klein ist.

Bsp: $n = 2$: $A = \begin{pmatrix} 137 & 100 \\ 100 & 73 \end{pmatrix}$

$$A^{-1} = \begin{pmatrix} 73 & -100 \\ -100 & 137 \end{pmatrix}$$

Mit: $\underline{b} = \begin{pmatrix} 4{,}30 \\ 3{,}10 \end{pmatrix} \Rightarrow \underline{x} = \begin{pmatrix} 3{,}9 \\ -5{,}3 \end{pmatrix}$

Kleine Störung bei $\underline{b}$:

$\tilde{b} = \begin{pmatrix} 4{,}31 \\ 3{,}10 \end{pmatrix} = \underline{b} + \Delta$; $\Delta = \begin{pmatrix} 0{,}01 \\ 0{,}00 \end{pmatrix} \sim 2‰$

Lösung für $\tilde{b}$: $\tilde{x} = \begin{pmatrix} 4{,}63 \\ -6{,}20 \end{pmatrix} \sim$ Änderung 20%

Jemand behauptet $\tilde{x}$ löse $A\underline{x} = \underline{b}$.

Kontrolle: Residuum $\underline{r} = A\tilde{\underline{x}} - \underline{b} = \begin{pmatrix} 0{,}01 \\ 0 \end{pmatrix}$ scheint ok.

Trotzdem ist $\tilde{x}$ um 20% falsch.

Zum Verständnis brauchen wir die Vektornorm, Matrixnorm und Kondition.

Vektornormen:

Sei $x \in \mathbb{R}^{n \times n}$

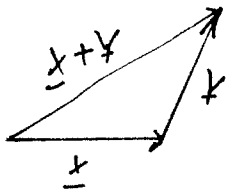
▷ Def: Die Zahl $\|x\| \in \mathbb{R}$ heisst eine Norm falls gilt:

i) $\|\underline{x}\| \geq 0 \quad \forall \underline{x} \in \mathbb{R}^n$, "$=0$" $\Longleftrightarrow \underline{x} = \underline{0}$

ii) $\alpha \in \mathbb{R}$: ~~$\|\alpha \mathbb{R}\|$~~

$$\|\alpha \underline{x}\| = |\alpha| \, \|\underline{x}\| \quad \forall \underline{x} \in \mathbb{R}^n$$

iii) $\|\underline{x} + \underline{y}\| \leq \|\underline{x}\| + \|\underline{y}\|$



▷ Bem: $\|.\|$ ist Abstandsbegriff zwischen Pkten $\underline{x}$ und $\underline{y}$

Bsp: Höldernorm, p-Norm

$$\underline{X} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

▷ Def: $\|\underline{x}\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$; $p \geq 1$

3 Spezialfälle:

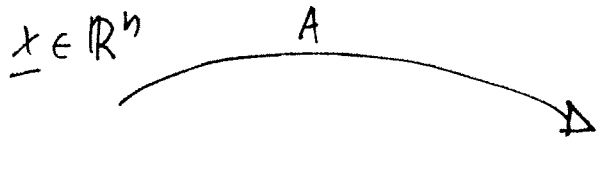
$p = 2$: $\|\underline{x}\|_2 = \sqrt{\left(\sum_{i=1}^{n} x_i^2 \right)}$ "Euklidische Norm"

$p = 1$: $\|\underline{x}\|_1 = \sum_{i=1}^{n} |x_i|$ "Summennorm"

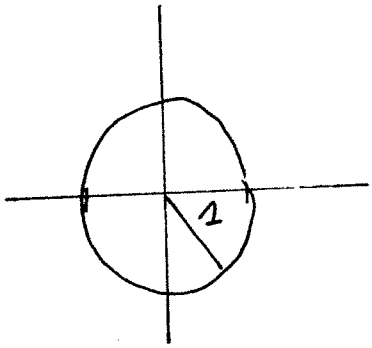
$p = \infty$: $\|\underline{x}\|_\infty = \max_{k=1,\ldots,n} |x_k|$ "Maximumnorm"

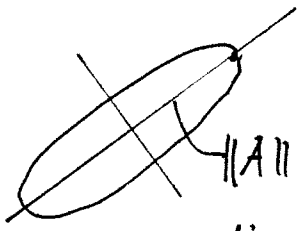
Die einer Vektornorm zugehörige Matrixnorm

▷ Def: $\|A\| := \max_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|$



$p = 2$





die grösste Streckung

1) Hauptsatz: $\|Ax\| \leq \|A\| \|x\|$ (1)

▷ Satz: $A = (a_{ij})$

$$\|A\|_1 = \max_j \sum_{i=1}^{n} |a_{ij}|$$

maximale ~~Kolonnen~~ Spalten Betrags-Summe

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}|$$

maximale Zeilen —"— —"—

$p = 2$

$\|A\|_2 = |\lambda_{max}|$, falls $A = A^T$

allgemein:

$\|A\|_2 = \sigma_{max}$ (grösster Singulärwert)

Kondition einer Matrix

▷ Einfluss Datenfehler !

Sei $A \in \mathbb{R}^{n \times n}$, regulär, $b \neq 0$

LGS:

ungestört: $A\underline{x} = \underline{b}$ (2)

gestört: $A(\underline{x} + \Delta\underline{x}) = \underline{b} + \Delta\underline{b}$ } −

$\Rightarrow A\,\Delta\underline{x} = \Delta\underline{b}$

$\Delta\underline{x} = A^{-1}\Delta\underline{b}$ (3) absolute Fehler

Wende (1) auf (2) u. (3) an:

$\|\underline{b}\| \leq \|A\| \|\underline{x}\|$

$\|\Delta\underline{x}\| \leq \|A^{-1}\| \|\Delta\underline{b}\|$

$\|\underline{b}\| \cdot \|\Delta\underline{x}\| \leq \|A\| \|A^{-1}\| \|\underline{x}\| \cdot \|\Delta\underline{b}\|$

$$\underbrace{\frac{\|\Delta\underline{x}\|}{\|\underline{x}\|}}_{\text{rel. Fehler}} \leq \|A\| \|A^{-1}\| \underbrace{\frac{\|\Delta\underline{b}\|}{\|\underline{b}\|}}_{\text{rel. Fehl. rechte Seite}}$$

▷ Def: $\mathrm{cond}(A) = \|A\| \|A^{-1}\|$

Kondition von A bezüglich $\|\cdot\|$.

▷ Satz:

$$\frac{\|\Delta\underline{x}\|}{\|\underline{x}\|} \leq \underbrace{\mathrm{cond}(A)}_{\text{maximale Verstärkungsfaktor mit dem der relative Fehler von } \underline{b} \text{ sich auf den rel. Fehler der Lösung überträgt}} \frac{\|\Delta\underline{b}\|}{\|\underline{b}\|} \quad (4)$$

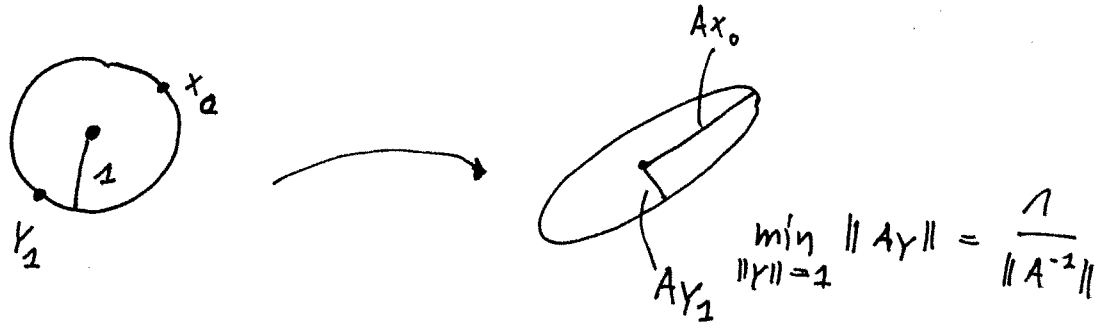
10.03.2008

▷ Bem: i)

cond(A) hängt von Norm ab

$$\text{cond}(A) = \frac{\max\limits_{\|x\|=1} \|Ax\|}{\min\limits_{\|y\|=1} \|Ay\|}$$

$$\underline{x} \longmapsto A\underline{x}$$



2. Bsp: $A = A^T$

$$\|.\|_2 \Longrightarrow \text{cond}_2(A) = \frac{\max|\lambda|}{\min|\lambda|}$$

λ := Eigenwerte

ii) MATLAB: cond(A, p)

Es wird die exakte cond(A) berechnet.

Fehleranalyse

Rundungsfehler bei LR-Zerlegung (Gauss) wirken sich aus wie eine Störung der rechten Seite $\underline{b}$

exakt: $A\underline{x} = \underline{b} \in \mathbb{R}^n$; $A \in \mathbb{R}^{n \times n}$

$\tilde{\underline{x}} = \underline{x} + \Delta\underline{x}$ numerisches Resultat

Residuum: $\underline{r} = A\tilde{\underline{x}} - \underline{b} \cong \Delta\underline{b}$

relatives Residuum: $\frac{\|\underline{r}\|}{\|\underline{b}\|} = \frac{\|\Delta\underline{b}\|}{\|\underline{b}\|}$

eps := Maschinengenauigkeit

eps := min (ε) mit $1 + \varepsilon > 1$ (auf Rechner)

i.a. erreicht ein guter Algorithmus unabhängig von n und von cond (A)

(5) $\frac{\|\underline{r}\|}{\|\underline{b}\|} \leq g \cdot \text{eps}$; $g \tilde{\leq} 1$

(4), (5) $\Rightarrow$ $\boxed{\frac{\|\Delta\underline{x}\|}{\|\underline{x}\|} \leq \text{cond}(A) \cdot g \cdot \text{eps}}$

▷ Bem:

iii) cond $(A) \geq 1$

$I = A^{-1}A$

$1 = \|I\| = \|A^{-1} \cdot A\| \leq \|A^{-1}\| \cdot \|A\|$

(iv) Fehler bei Multiplikation "exakte" Matrix mal

"gestörter" Vektor:

Geg: T reg. $T \in \mathbb{R}^{n \times n}$; $\underline{v} \in \mathbb{R}^n$

Ges: $\underline{w} = T\underline{v}$; $\underline{w} \in \mathbb{R}^n$

Ann: $\tilde{\underline{v}} = \underline{v} + \Delta\underline{v}$

Wie groß ist der Fehler von $\tilde{\underline{w}} = T\tilde{\underline{v}}$

Absolute Fehler: $\Delta\underline{w} = \tilde{\underline{w}} - \underline{w} = T\Delta\underline{v}$

$$\frac{\|\Delta\underline{w}\|}{\|\underline{w}\|} \leq \text{cond}(T) \frac{\|\Delta\underline{v}\|}{\|\underline{v}\|}$$

"Vorkonditionierer" (Aufgabe)

Wenn immer man mit Matrizen multipliziert, sollten Matrizen mit kleiner Konditionszahl gewählt werden.

2.3. QR-Zerlegung

Ausgleichsrechnung nach kleinsten Quadraten.

Vorbereitung: orthogonale Matrizen

$Q \in \mathbb{R}^{m \times m}$ heisst orthogonal $\Leftrightarrow$ $Q^{-1} = Q^T$

Eigenschaften:

Normtreue in $\|.\|_2$: $\|Q\underline{x}\|_2 = \|x\|_2$; $\forall \underline{x}$

Bsp: Drehung, Spiegelungen

$\det(Q) = \pm 1$

- Euklidische Norm ist natürliche Norm

$\|Q\|_2 = 1$, $\text{cond}(Q) = 1$

▷ Satz: QR-Zerlegung

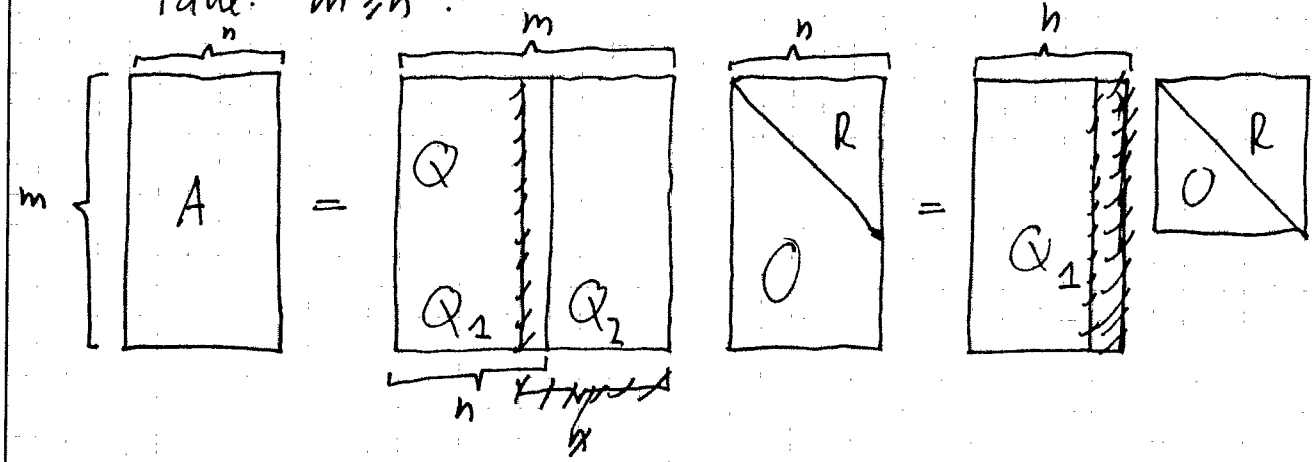
$A \in \mathbb{R}^{m \times n}$, beliebige Rechtecksmatrix.

Dann existiert eine orthogonale Matrix

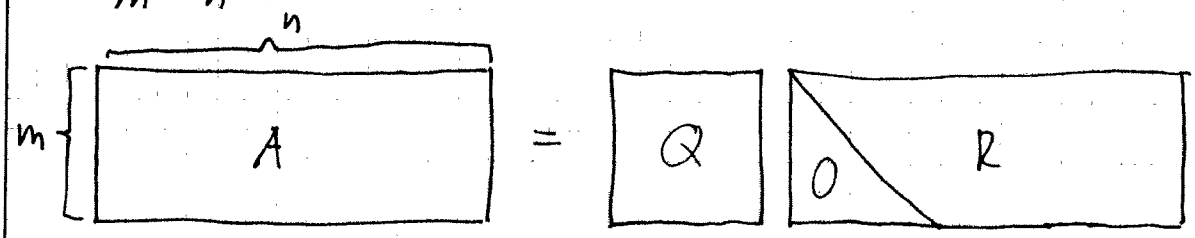
$Q \in \mathbb{R}^{m \times m}$ mit:

$A = QR$

Fälle: $m \geq n$:



$m < n$:



Beweis: Konstruktiv, Rechenverfahren

Idee: $R = Q^T A$

Schrittweiser Aufbau von Q als Produkt von Elementardrehungen.

▷ Def: Givens-Rotationen

Drehung in der (k,l)-Ebene des $\mathbb{R}^m$ um den Winkel φ.

$$\underline{x} \in \mathbb{R}^m \longmapsto Q_{kl}^T(\varphi)\,\underline{x} \quad ; k \neq l$$

mit $Q_{kl}^T :=$

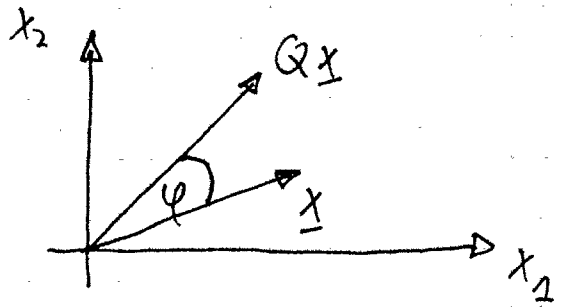
$$\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & \cos(\varphi) & & -\sin(\varphi) & & \\ & & & \ddots & & & \\ & & \sin(\varphi) & & \cos(\varphi) & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}$$

Zeile k (mit $-\sin(\varphi)$), Zeile l (mit $\cos(\varphi)$)

Spalte k Spalte l

▷ Bem: Ebene Drehung

$$\underline{x} \longmapsto Q\underline{x} \quad ; \quad Q = \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix}$$



Wirkung der Givensrotation auf A

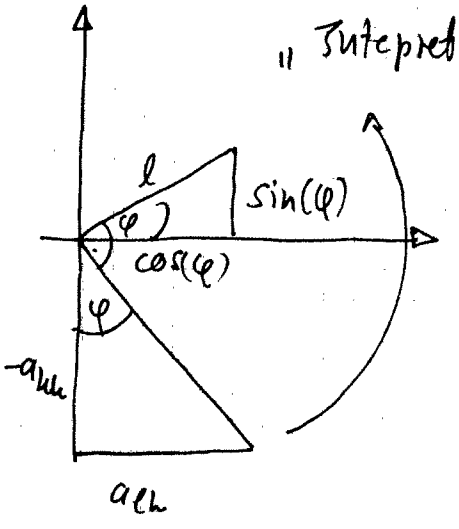
$$A_1 = Q_{k\ell}^T(\varphi)\, A \qquad \begin{pmatrix} \underline{a}_1^T \\ \underline{a}_2^T \\ \underline{a}_m^T \end{pmatrix} \quad ; A \in \mathbb{R}^{m\times n}$$

$$A_1 = \begin{pmatrix} \underline{a}_1^T \\ \underline{a}_2^T \\ \underline{a}_k^T \cos(\varphi) + \underline{a}_\ell^T \sin(\varphi) \\ \underline{a}_{k+1} \\ -\underline{a}_k^T \sin(\varphi) + \underline{a}_\ell^T \cos(\varphi) \\ \underline{a}_m^T \end{pmatrix} \begin{matrix} \\ \\ \leftarrow \text{Zeile } k \\ \\ \leftarrow \text{Zeile } \ell \\ \\ \end{matrix}$$

$$(A_1)_{\ell k} = -a_{kk}\sin(\varphi) + a_{\ell k}\cos(\varphi) \stackrel{!}{=} 0$$

$$\varphi = \text{angle}(a_{kk} + i\,a_{\ell k}) \qquad \text{"Matlab"}$$

"Interpretation"



QR:

Kolonnenweise alle Elemente $\neq 0$ des unteren Dreiecks von A "zu Null" machen.

$$R = Q_{m-1,n-1}^T \cdots Q_{23}^T\, Q_{1n}^T \cdots$$

$$\cdots Q_{13}^T\, Q_{12}^T(\varphi_{12})\, A$$

▷ Anwendung: Least squares approximation

Bsp: Ansatz: linear

$$K(t) = \frac{a_0}{2} + \sum_{k=1}^{N} a_k \cos\left(\frac{2\pi}{p} kt\right) + b_k \sin\left(\frac{2\pi}{p} kt\right)$$

Finde $a_0, a_1, \dots a_N, b_1, \dots, b_N$ so dass es passt

Typisches Problem:

Lineares Modell für eine Messreihe

Geg: m Zeitpunkte t_k, $k = 1, 2, \dots m$ exakt

m Messwerte f_k, $k = 1, 2, \dots, m$ fehlerbehaftet

Betrachte "lineares Modell"

Ansatz: $\phi(t) := \sum_{l=1}^{n} \phi_l(t)\, c_l$

ϕ_l Basisfkt $l = 1, \dots, n$

das die Messreihe approximiert

$\phi(t_k) \approx f_k$

($K(t) = \phi(t)$, Bsp.

Basisfkt: $n = 2N + 1$

$\phi_l = \{ \dots \}$

Ges: $\underline{c} = \begin{pmatrix} c_1 \\ \vdots \\ c_n \end{pmatrix}$, sodass "Abstand" klein

Lösung: Betrachte Residuenvektor

$\underline{r} = \begin{pmatrix} r_1 \\ \vdots \\ r_m \end{pmatrix} \in \mathbb{R}^m$; $r_k = \phi(t_k) - f_k$

Modellmatrix:

$$A = \left(\varphi_l(t_k) \right)_{\substack{k=1,\dots,m \\ l=1,\dots,n}}$$

$$\underline{r} = A\underline{c} - \underline{f} \quad ; \quad \underline{f} = \begin{pmatrix} f_1 \\ \vdots \\ f_m \end{pmatrix}$$

$\underline{c} \in \mathbb{R}^n$; $f \in \mathbb{R}^m$

Somit suche $\underline{c} \in \mathbb{R}^n$ mit $\|\underline{r}\| \stackrel{!}{=}$ minimal in geeigneter Norm. ⊛

17.03.2008

▷ Methode der kleinsten Quadrate heisst $\|.\| := \|.\|_2 =$

$= \sum_{k=1}^{m} r_k^2$; Gauss., least squares

⊛ Lösen mit der QR-Faktorisierung

$$\|\underline{r}\|_2^2 = \|A\underline{c} - \underline{f}\|_2^2$$

$$= \|QR\underline{c} - \underline{f}\|_2^2$$

$$= \|R\underline{c} - Q^T\underline{f}\|_2^2 \stackrel{!}{=} \min$$

$Q^{-1} = Q^T$ ↗

Residuum $\tilde{\underline{r}}$

$$\tilde{\underline{r}} := \begin{pmatrix} R \\ 0 \\ \hline 0 \end{pmatrix}_{\substack{m \\ n}} \cdot \underline{c} - Q^T\underline{f} = \begin{pmatrix} \tilde{\tilde{r}}_{oben} \\ \hline \tilde{\tilde{r}}_{unten} = (Q^T\underline{f})_{unten} \end{pmatrix}$$

Aus $\| \tilde{r} \|_2^2 = \underbrace{\| \tilde{r}_{oben} \|_2^2}_{=0} + \underbrace{\| \tilde{r}_{unten} \|_2^2}_{(Q^T f)_{unten} \text{ ist fest}} \stackrel{!}{=} \min$

$\tilde{r}_{oben} \stackrel{!}{=} 0$

$R\underline{c} - (Q^T \underline{f})_{oben} = 0$

$R\underline{c} = (Q^T \underline{f})_{oben}$ "Lösen durch Rückwärtseinsetzen"

MATLAB:

$A \in \mathbb{R}^{m \times n}$; $\underline{f} \in \mathbb{R}^m$

$m \geq n$

Befehl: `c = A \ f` löst $\| A\underline{c} - \underline{f} \|_2^2 \stackrel{!}{=} \min$ mit QR-Zerlegung.

2.4. Eigenwertprobleme

Rückblick:

$A \in \mathbb{R}^{n \times n}$; Finde $\underline{x} \neq 0$ mit $A\underline{x} = \lambda \underline{x}$; $\lambda \in \mathbb{C}$

"Eigenwertproblem"

$\underline{x}$ heisst Eigenvektor (EV)

λ -"- Eigenwert (EW)

▷ Charakteristisches Polynom: $(A - \lambda I)\, x = 0$ (1)

$\Rightarrow \underbrace{\det(A - \lambda I)}_{p(\lambda)} = 0$

$p(\lambda) :=$ char. Polynom

$$p(\lambda) = \lambda^n - \lambda^{n-1} \underbrace{\mathrm{tr}(A)}_{= \sum_{k=1}^{n} a_{kk}} + \ldots +$$

Allgemein $\lambda_k \in \mathbb{C}$.

Lin. Alg. $\Rightarrow$ bestimme $p(\lambda)$ — MATLAB: $\underline{c} = \mathrm{poly}(A)$

Nullstellen bestimmen — $\underline{\lambda} = \mathrm{roots}(\underline{c})$

unbrauchbar !!

Grund: λ_k sind schlecht bestimmt durch Koeff. eines Polynoms.

Bsp: $A = \begin{pmatrix} 1 & \varepsilon \\ \varepsilon & 1 \end{pmatrix}$; $p(\lambda) = \lambda^2 - 2\lambda + 1 - \varepsilon^2$

Sei MATLAB $\varepsilon = 5 \cdot 10^{-9}$

$\tilde{p}(\lambda) = \lambda^2 - 2\lambda + 1$

↑ mit Rechner

$p(\lambda) = 0 \Rightarrow \lambda_{1/2} = 1 \pm \varepsilon$ "Beide sind auf Rechner darstellbar"

$\tilde{p}(\lambda) = 0 \Rightarrow \lambda_{1/2} = 1$

"Nie Nullstellen von Polynomen ausrechnen, immer in Matrizen umwandeln und λ_k bestimmen."

▷ EW können zusammenfallen:

mehrfache EW : Anzahl des Auftretens nennt man die algebraische Vielfachheit.

▷ Diagonalform einer Matrix:

Lemma: Zu verschiedenen EW gehören linear unabhängige EV.

Ann: A hat n verschiedene EW.

EW: $\lambda_1, \lambda_2, \lambda_3, \dots, \lambda_n$

EV: $\underline{x}_1, \underline{x}_2, \underline{x}_3, \dots, \underline{x}_n \Rightarrow$ linear unabhängig

$A\underline{x}_k = \lambda x_k \quad ; \; k = 1, \dots, n$

$$A \underbrace{\begin{bmatrix} | & | & & | \\ \underline{x}_1 & \underline{x}_2 & \cdots & \underline{x}_n \\ | & | & & | \end{bmatrix}}_{T} = \begin{bmatrix} | & | & & | \\ \underline{x}_1 & \underline{x}_2 & \cdots & \underline{x}_n \\ | & | & & | \end{bmatrix} \underbrace{\begin{pmatrix} \lambda_1 & & 0 \\ & \lambda_2 & \\ 0 & & \lambda_n \end{pmatrix}}_{D}$$

$AT = TD$

Satz: $A \in \mathbb{R}^{n \times n}$; n linear unabhängige EV $\underline{x}_k$

$AT = TD \Rightarrow A = TDT^{-1}$

$D = T^{-1}AT$

} Ähnlichkeitstransformationen

MATLAB: [T, D] = eig (A)

▷ Eigenraum: (ER)

$\underline{x}_1, \underline{x}_2$ zwei Eigenvektoren zum selben EW λ:

$$A\underline{x}_2 = \lambda \underline{x}_1 \quad | \cdot \alpha_1$$
$$A\underline{x}_2 = \lambda \underline{x}_2 \quad | \cdot \alpha_2$$

$$A(\alpha_1 \underline{x}_1 + \alpha_2 \underline{x}_2) = \lambda(\alpha_1 \underline{x}_1 + \alpha_2 \underline{x}_2)$$

$\Rightarrow$ $\alpha_1 \underline{x}_2 + \alpha_2 \underline{x}_2$ ist auch EV

$\forall\, \alpha_1, \alpha_2$

Fälle: $\underline{x}_1, \underline{x}_2$ linear unabhängig

$\{\alpha_1 \underline{x}_1 + \alpha_2 \underline{x}_2\}$ bilden 2-dim. Eigenraum

$\underline{x}_1, \underline{x}_2$ linear unabhängig

$\underline{x}_2 = \gamma \underline{x}_1$

$\Rightarrow \{\alpha_1 \underline{x}_1 + \alpha_2 \underline{x}_2\} = \{c_1 \underline{x}_1\}$

1.-dim. Raum

Zusammenfassung:

▷ EV sind höchstens auf freien Faktor bestimmt. z. Bsp. $\|\cdot\| = 1$.

▷ alle EV zum Eigenwert λ bilden einen linearen Raum: ER zu λ

▷ Ein einfacher EW (alg. V = 1) hat immer eine 1-Dim ER.

Def: Die dim. des ER zu einem mehrfachen EW heißt die geom. Vielf. des EW (= Anzahl d. linear unabhäng. EV)

Satz: geometrische Vielfachheit $\leq$ algebraische Vielfachheit

Bsp: $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$; $\lambda_1, \lambda_2 = 1$ alg. $V = 2$

geom. $V = 2$

$A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$; $\lambda_1 = \lambda_2 = 1$ alg. $V = 2$

es existiert nur 1 EV ; $\underline{x}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \rightarrow$ geom $V = 1$

Satz: Matrix ist diagonalisierbar, wenn für jeden EW gilt geom V = algebr. V.

Folgerung: $A = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ ist nicht diagonalisierbar.

MATLAB: $[T, D] = eig(A)$

Falls A nicht diagonalisierbar :

- ein k-facher EW λ wird in k einfache EW aufgespalten.

$|\lambda_l - \lambda| \leq const(eps)^{\frac{1}{k}}$

$\Rightarrow$ Damit wird A "diagonalisierbar"

Die zugehörigen EV ; 3fach

▷ Symmetrische Matrizen $A = A^T$

Es gilt: Alle EW sind reell, $A = A^T \Rightarrow$ ist immer diagonalisierbar; T kann orthogonal gewählt werden.

$A \in \mathbb{C}$; $A = (\bar{A})^T = A^H$ ← Hermitsch.

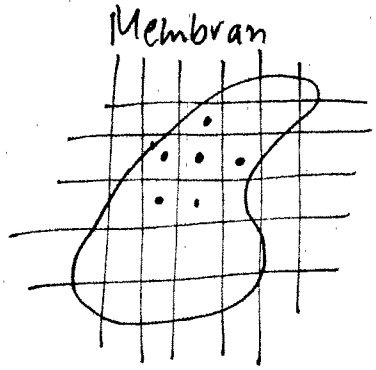
T unitär : $T^{-1} = T^H$

2.5. Numerik der Eigenwertprobleme

Aufgabe	Technik
a) Bestimmung einzelner EV und EW, vor allem des absolut grössten oder des absolut kleinsten	Vektoriteration mit Startvektor $\underline{x}_0$ $\underline{x}_{k+1} = A\,\underline{x}_k$
b) Bestimmung aller EW und EV	Matrixzerlegung $T^{-1}AT = D$

a) Vektoriteration

Typisches Problem: Schwingung eines Kontinuums (Hochspannungsleitungen, Membran) in diskreter Approximation.



$\Delta u = \lambda u$

n Felder $u = 0$ auf Rand

Diskrete Approximation.

Sei $\underline{x} \in \mathbb{R}^n$, x gibt Auslenkung der Mitte des Feldes an

$$A\underline{x} = \lambda \underline{x}$$

$$A \in \mathbb{R}^{n \times n}$$

Spezielle Situation:

- A, x sind gross, d.h. n gross
- A ist dünnbesetzte Matrix
- $A = A^T$
- Nur wenige EW am unteren Spektrum gesucht.

▷ Algorithmus: Wähle (fast) beliebigen Startvektor $\underline{x}_0$

Iterationen $\underline{x}_{k+1} = A\underline{x}_k$; $k = 0, 1, 2.$

Resultat: Folge $\{\underline{x}_k\}$ konvergiert nach EV zum grössten EW.

31.03.08

Kurze Wiederholung:

EW-Probleme : $A\underline{x} = \lambda \underline{x}$

Schlüsselbegriffe:
EW, EV, algebraische Vielfachheit, Diagonalform, ER, ...

Numerik der EW-Probleme

! Nicht das charakteristische Polynom bestimmen !

Grund: λ_k sind schlecht bestimmt durch die Koeffi. eines Polynoms. (siehe Bsp.)

Aufgabenstellung (mündlich wiederholen)

bzw.
Fragestellung

a) Vektoriteration

Bsp. : Schwingende Membran (Zeichnung verteilen)

kontinuierlich:

$$\begin{cases} \Delta u = \lambda u & \text{in } G \\ u = 0 & \text{auf } \partial G \end{cases}$$

diskret:

$$\begin{cases} A\underline{x} = \lambda \underline{x} & \text{in } G \\ \underline{x} = 0 & \text{auf } \partial G \end{cases}$$

$\underline{x} \equiv$ Vektor der Auslenkungen der Gitterpunkte. wobei $\lambda = \omega^2$ (ω: Eigenfrequenz)

Algorithmus: (direkte Vektoriteration)

▷ Standard: $\underline{x}_{k+1} = A\underline{x}_k$; $k = 0, 1, 2, \ldots$
mit (fast) beliebigen Startvektor $\underline{x}_0$

▷ Normierung: $x_{k+1} = \dfrac{A\underline{x}_k}{\|A\underline{x}_k\|}$

▷ Rayleigh-Quotient: $\lambda_{max} = \lim\limits_{k \to \infty} \dfrac{\underline{x}_k^T A \underline{x}_k}{\underline{x}_k^T \underline{x}_k}$ (Behr's Serie)

▷ Satz: Ann: λ_1 ein einfacher EW von $A \in \mathbb{R}^{n\times n}$ $(A = A^T)$ und $|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|$; desweiteren stehe $\underline{x}_0 \in \mathbb{R}^n$ nicht senkrecht auf dem ER von λ_1.

Folge: $\underline{y}_k := \frac{\underline{x}_k}{\|\underline{x}_k\|}$ mit $\underline{x}_{k+1} = A\underline{x}_k$ konvergiert gegen einen normierten EV von A zum EW λ_1

▷ Beweis:

$\underline{v}_1, \underline{v}_2, \dots, \underline{v}_n$ bilden eine Orthonormalbasis von Eigenvektoren von A mit $A\underline{v}_i = \lambda_i \underline{v}_i$.
Dann gilt $\underline{x}_0 = \sum_{i=1}^{n} \alpha_i \underline{v}_i$ mit $\alpha_1 = \langle \underline{x}_0, \underline{v}_1 \rangle \neq 0$
Folglich ist:

$$\underline{x}_k = A^k \underline{x}_0 = A^k \sum_{i=1}^{n} \alpha_i \underline{v}_i = \sum_{i=1}^{n} \alpha_i \lambda_i^k \underline{v}_i$$

$$= \alpha_1 \lambda_1^k \underbrace{\left[\underline{v}_1 + \sum_{i=2}^{n} \frac{\alpha_i}{\alpha_1} \left(\frac{\lambda_i}{\lambda_1} \right)^k \underline{v}_i \right]}_{=: \underline{z}_k}$$

Da $|\lambda_i| < |\lambda_1| \quad \forall i = 2, \dots, n$,
gilt
$\lim_{k\to\infty} \underline{z}_k = \underline{v}_1$ und deshalb

$$\underline{y}_k = \frac{\underline{x}_k}{\|\underline{x}_k\|} = \frac{\underline{z}_k}{\|\underline{z}_k\|} \xrightarrow[k\to\infty]{} \underline{v}_1$$

□

Wir haben gezeigt die Folge $y_k = \frac{x_k}{\|x_k\|}$ konvergiert der Richtung nach gegen EV zum grössten EW.

▷ Bemerkungen:

- erhalten nur den EV zum grössten EW (N1)
- Berechnung ist „günstig"; A muss nicht gespeichert werden
- falls $\alpha_1 \cong 0$, konvergiert y_k dennoch gegen v_1 (Rundungsfehler)
- Konvergenzgeschwindigkeit hängt vom Quotienten $\left|\frac{\lambda_2}{\lambda_1}\right|$ ab
  ⇒ Langsame Konvergenz wenn $|\lambda_2|$ und $|\lambda_1|$ dicht zusammen liegen (N2)

- Bestimmung von λ_{min}: Inverse Vektoriteration

Allgemein:
Angenommen, wir hätten einen Schätzer $\mu \approx \lambda_i$ ($\to 0$) eines beliebigen EW der Matrix A, so dass

$$|\lambda_i - \mu| < |\lambda_j - \mu| \quad \forall\, j \neq i.$$

Dann ist $(\lambda_i - \mu)^{-1}$ der betragsgrösste EW der Matrix $(A - \mu I)^{-1}$.

Iterationsvorschrift
⇒
$$(A - \mu I)\, x_{k+1} = x_k \qquad \text{für } k = 0, 1, \ldots$$

▷ Bei jedem Iterationsschritt wird das LGS gelöst, jedoch nur für verschiedene rechte Seiten x_k.
⇒ $(A - \mu I)$ muss nur einmal zerlegt werden!

Zusammenfassung:

vielleicht wegfassen

Matrix	EW	EV	Grund
A	λ_i	$\underline{v}_i$	$A = TDT^{-1}$
A^{-1} $(\mu = 0)$	λ_i^{-1}	$\underline{v}_i$	$A^{-1} = TD^{-1}T^{-1}$
$(A - \mu I)$	$(\lambda_i - \mu)$	$\underline{v}_i$	$(A-\mu I) = T(D-\mu I)\,T^{-1}$

- **Konvergenzfaktor:** (linear)

$$\max_{j \neq i} \left| \frac{\lambda_i - \mu}{\lambda_j - \mu} \right| < 1$$

für besonders gute Schätzer $\ll 1$

- **Simultane Vektoriteration (Unterraum-Iteration)**

Ziel: p kleinste EW + dazugehörige EV bestimmen, die paarweise orthonormal sein sollten.

Algorithmus:

▷ p orthonormale Startvektoren: $X_0 = \left(x_0^{(1)}, \dots, x_0^{(p)}\right) \in \mathbb{R}^{n \times p}$

▷ Iteration

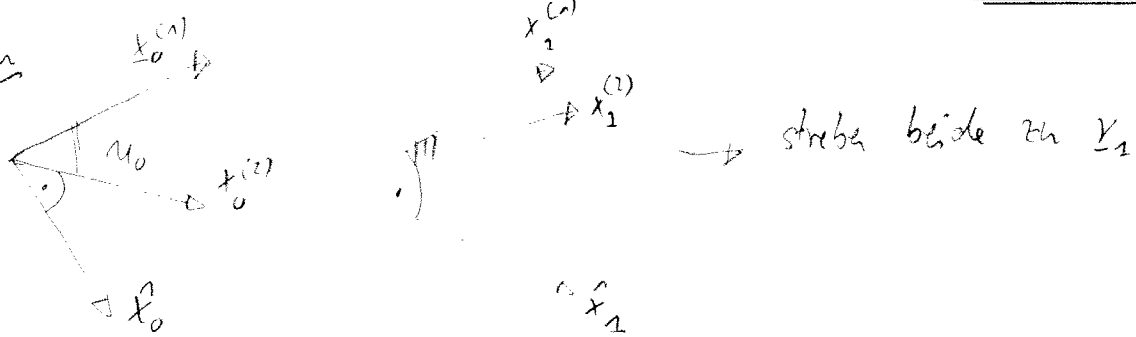
Matlab: $[Q_k, R_k] = qr(X_k)$; $X_k = [X_k^{(1)} \dots$

$$X_{k+1} = A X_k \quad \Rightarrow Q_k = \left[q_k^{(1)}, \dots, q_k^{(p)}\right]$$

$$\langle q_k^{(i)}, q_k^{(j)} \rangle = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

- Vektoren $x_k^{(i)}$, $i = 1, \dots, p$ werden nicht mehr orthonormal sein
- Nach jeder Iterationsschritt eine orthog. Basis $x_k^{(1)}, x_k^{(2)}, \dots x_k^{(p)}$ im Unterraum $U_k \in \mathbb{R}^k$ konstruieren mit Hilfe der QR-Zerlegung

Warum



→ streben beide zu $\underline{v}_1$

b) QR-Algorithmus

Gebräuchlichste Methode zur Approximation aller EW und EV einer allgemeinen Matrix $A \in \mathbb{R}^{n \times n}$

Sei $A_0 := A$

$$\begin{cases} \text{zerlege } A_0 = Q_0 R_0 = \boxed{Q_0}\,\boxed{R_0} \;;\; Q_0^T Q_0 = I \\ \text{bilde } A_1 = R_0 Q_0 \end{cases}$$

Wegen $R_0 = Q_0^{-1} A_0 \Rightarrow A_1 = Q_0^{-1} A_0 Q_0 = \begin{pmatrix} \tilde{\tilde{R}}_{11} & & * \\ & \ddots & \\ 0 & & \tilde{\tilde{R}}_{nn} \end{pmatrix}$

$\Rightarrow A_1$ ist „ähnlich" (orthogonal) zu A_0

$\Rightarrow A_1$ hat dieselben EW wie A_0

Schursche Normalform (Satz von Schur erwähnen)

Algorithmus:

Gegeben $A_0 = A \in \mathbb{R}^{n \times n}$

$$A_k = Q_k R_k \;;\; A_{k+1} = R_k Q_k \;,\; k = 0, 1, \ldots$$

Satz: (wird nicht bewiesen)

Falls A_0 lauter EW von verschiedenem Betrag hat, konvergiert die Matrixfolge A_k gegen eine obere Dreiecksmatrix $\tilde{\tilde{R}}$.

Bemerkung:

(i) EW von A erscheinen als Einer- und Zweierblöcke auf der Diagonalen von $\tilde{\tilde{R}}$.

(ii) Bei betragsgleichen EW, entstehen „Kästchen" auf der Diagonalen, die nicht konvergiere.

(iii) EV: $\underline{x}^{(i)} = Q \underline{x}^{(i)}$, $\underline{x}^{(i)}$ sind EV von A_k und $Q := Q_1 \cdot \ldots \cdot Q_k$

function QR_ZerlegungFuerEW mit Matlab zeigen:

MATLAB Beispiel

```
clc;
A = [0,0,4,-5,2; -2,1,8,-10,4;-2,-2,14,-15,6;...
     -2,-2,14,-17,8;-2,-2,15,-20,10];

eig(A)
n = 40
for i=1:n
    [Q,R] = qr(A);
    Aneu = R*Q;
    A = Aneu;
end;
A
return;
```

```
ans =

   2.0000
   1.0000
   3.0000
   1.0000 + 1.0000i
   1.0000 - 1.0000i

n =

    10


A =

    2.0010    5.9357    8.3191   40.1465  -11.7702
   -0.0001    3.0005    1.5153    9.5280   -1.3649
   -0.0000    0.0000    1.4992   -4.2026    1.0293
    0.0000   -0.0003    0.2850    0.4908    0.1333
    0.0002   -0.0012   -0.0485   -0.0496    1.0084

n =

    40

A

    2.0000    5.9386   28.0212   32.0702   -2.3863
   -0.0000    3.0000    6.1357    7.5277    0.8402
    0.0000   -0.0000   -0.5000   -3.7528    0.0000
    0.0000   -0.0000    0.8660    2.5000   -0.0000
    0.0000   -0.0000    0.0000    0.0000    1.0000

>> eig([-0.5 -3.7528; 0.8660, 2.500])

ans =

   1.0000 + 1.0000i
   1.0000 - 1.0000i
```

▷ EV $\hat{y}^{(3)}$ zum EW $\hat{\lambda}_3 \approx 3{,}0$ von A_{40}:

$$(A_{40} - \hat{\lambda}_3 \mathbb{1})\, \hat{y}^{(3)} = 0$$

$$\hat{x}^{(3)} = Q \hat{y}^{(3)} \quad \text{mit } Q = Q_1 \cdot Q_2 \cdot \ldots Q_{40}$$

07.04.2008

2.6. Singulärwertzerlegung (SVD)

G. Golub, Stanford

$A \in \mathbb{R}^{m \times n}$; Rang $s \leq \ell := \min(m,n)$

(1) $\underline{x} \in \mathbb{R}^n \longmapsto \underline{y} = A\underline{x}$

Neu Koordinaten: $\underline{\xi} \in \mathbb{R}^n$, $V \in \mathbb{R}^{n \times n}$ orthogonal $V^T V = \mathbb{1}_{n \times n}$

$\underline{x} = V\underline{\xi}$ (2a)

Bildraum: $\underline{\eta} \in \mathbb{R}^m$: $U \in \mathbb{R}^{m \times m}$ ortho. $U^T U = \mathbb{1}_{m \times m}$

(2b) $\underline{y} = U\underline{\eta}$

$$U\underline{\eta} = \underline{y} \overset{(1)}{=} A\underline{x} \overset{(2a)}{=} AV\underline{\xi}$$

$$\underline{\eta} = \underbrace{U^T A V}_{:=S}\underline{\xi}$$

Ziel: Wähle U und V, sodass S "einfach" wird.

Satz: Zu jeder Matrix $A \in \mathbb{R}^{m \times n}$ gibt es orthogonale Matrizen U, V, sodass

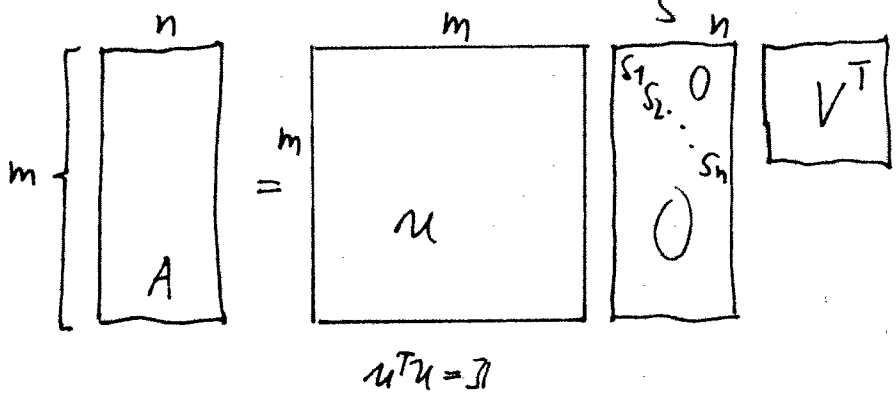
$$A = USV^T ;$$

dabei ist S diagonal mit nicht negativen Diagonalelementen

$$s_1 \geq s_2 \geq s_3 \geq \ldots \geq s_\ell \geq 0$$

$s_1, s_2, \ldots, s_\ell$ heissen Singulärwerte.

Bsp: $m \geq n$, $l = n$



$U^T U = \mathbb{1}$

Bem:

(i) Guter, stabiler Algorithmus zur Erzeugung von SVD bekannt. Aufwand ca. $10\,m^3$ Operationen.

→ nicht anfällig für Rundungsfehler

(ii) MATLAB: ~~SVD~~ $[U, S, V] = \text{svd}(A)$

(iii) Zusammenhang mit symmetrischen EWP

$$A^T A = V S^T S V^T$$

$$\text{diag}(s_1^2, s_2^2, \ldots, s_n^2)$$

$\Rightarrow s_k^2 = \lambda_k$; λ_k EW von $A^T A$

$s_k = \sqrt{\lambda_k}$

▷ Anwendungen:

(i) Rang einer Matrix $\text{rank}(A) = p < \min(m, n)$

$\Rightarrow S = \text{diag}(s_1, s_2, \ldots, s_p, 0, \ldots, 0)$

Praktisch werden die s_k mit $k > p$ zwar klein, aber nicht 0.

$$\frac{s_k}{s_1} \approx eps$$

|

Maschinengenauigkeit

"Numerischer Rang" hängt von der Rechnergenauigkeit ab.

$$s_{rech} = \max\{k\}$$

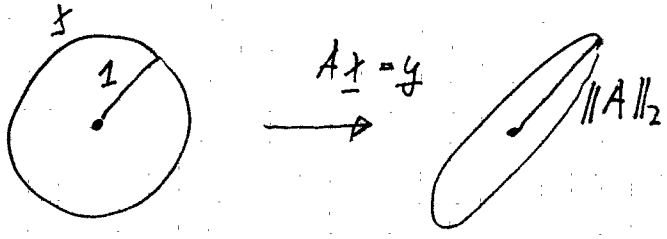
$$s_k > eps\, s_1$$

(ii) Euklidische Norm und Kondition

$$A \in \mathbb{R}^{n \times n} \; ; \; A = USV^T$$

$$S = diag(s_1, \dots, s_n)$$

$\|A\|_2$ - maximale Länge des Bildes eines Einheitsvektors.



$$\Rightarrow s_1 = \|A\|_2$$

$$\|A^{-1}\|_2 = \frac{1}{s_n} \qquad \left| \quad cond_2(A) = \frac{s_1}{s_n} \right.$$

$$A^{-1} = VS^{-1}U^T$$

$$\downarrow$$

$$diag\left(\frac{1}{s_1}, \frac{1}{s_2}, \dots, \frac{1}{s_n}\right)$$

(iii) Lineare Abbildung:

Kern, Bild ansehen.

Sei $A \in \mathbb{R}^{m \times n}$; $m \geq n$

$\operatorname{rank}(A) = s < n$

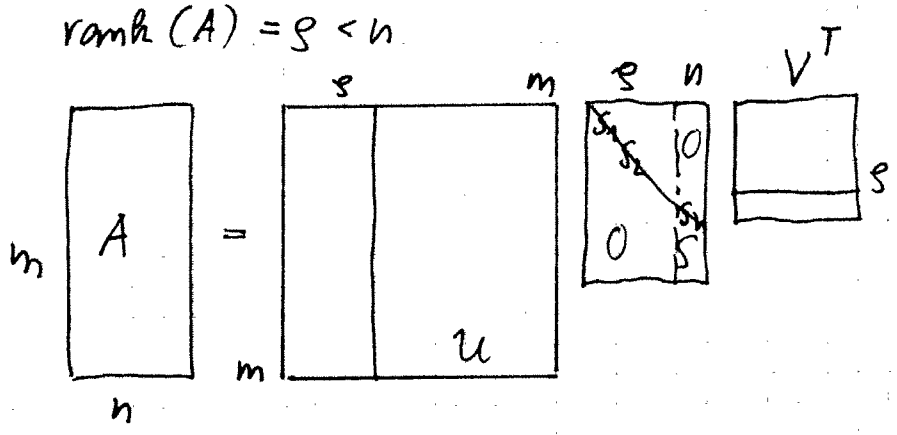
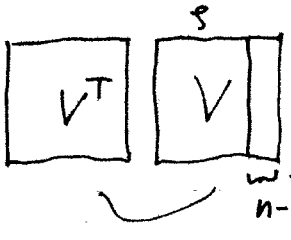


Abbildung: $\underline{x} \longmapsto \underline{y} = A\underline{x}$



orthogonale Basis von K_A

$\exists$

▷ Kern: $K_A = \{ \underline{x} \in \mathbb{R}^n \mid A\underline{x} = 0 \}$

$\dim K_A = n - s$ orthogonale Basis von K_A

= Spaltenvektoren $s+1, s+2, \ldots, n$ von V.

▷ Bild: (Range)

$R_A := \{ \underline{y} \in \mathbb{R}^m \mid \exists\, \underline{x} \in \mathbb{R}^n \text{ mit } A\underline{x} = \underline{y} \}$

$\dim R_A = s$; ortho. Basis: Spalten $1, 2, \ldots, s$ von U

(iv) Ausgleichsproblem mit nicht vollem Rang

Sei $A \in \mathbb{R}^{m \times n}$; $m \geq n$, $\operatorname{rank}(A) = p < n$

Residuum : $\underline{r} = A\underline{c} - \underline{f}$

Gesucht $\underline{c}$, sodass $\|\underline{r}\|_2 \stackrel{!}{=} \min$

Sei SVD bekannt. $A = U S V^T$

$S = \operatorname{diag}(s_1, \dots, s_p, 0 \dots, 0)$

Transformiertes Residuum :

$$\underline{\tilde{r}} = U^T \underline{r} = S(\underbrace{V^T \underline{c}}_{\underline{y}}) - \underbrace{U^T \underline{f}}_{\underline{g}} \qquad \underline{y} = V^T \underline{c}$$

Gesucht : $\underline{y} = V^T \underline{c}$

Komponentenweise :

$\tilde{r}_1 = s_1 y_1 - g_1 = 0 \qquad y_1 = g_1 / s_1$

$\tilde{r}_2 = s_2 y_2 - g_2 = 0$

$\tilde{r}_p = s_p y_p - g_p = 0$

$\tilde{r}_{p+1} = 0\, y_{p+1} - g_{p+1}$

$\vdots$

$\tilde{r}_n = 0\, y_n - g_n$

$\tilde{r}_{n+1} = 0\, y_{n+1} - g_{n+1}$

Lösung:

$$y_k = \begin{cases} g_k / s_k & k = 1, 2, \dots p \\ \text{beliebig} & k = p+1, \dots, n \end{cases}$$

$\Rightarrow$ es gibt ∞-viele Lösungen.

$\underline{c} = V \underline{y}$

Auswahl ausgezeichneter Lösung:

Wähle $y_{s+1}, \dots, y_n$

sodass $\| \underline{z} \|_2 \stackrel{!}{=} \min$

V orthogonal.

$\Rightarrow \| \underline{z} \|_2 = \| \underline{y} \|_2$

$\Rightarrow$ Minimum-Norm-Lösung

$0 = y_{s+1} = y_{s+2} = \dots = y_n$

3. Iterative Methoden für LGS

(1) Problem: $A\underline{x} = \underline{b}$

$A \in \mathbb{R}^{n \times n}$, „regulär" ; $b \in \mathbb{R}^n$

Speziell : n sehr gross . 10^c

A dünnbesetzt (sparse)

Gesucht: $\underline{x}$

A sparse „$\Rightarrow$" A^{-1} vollbesetzt

Nie A^{-1} berechnen !!

Benutze nur die Abbildung $\underline{x} \mapsto A\underline{x}$.

Ziel : Verfahren die nur die Abbildung benützen (Operatorprinzip)

3.1. Jacobi - Iteration

Sei $A = C - B$

Aus (1) folgt

$C\underline{x} - B\underline{x} = b$

$C\underline{x} = B\underline{x} + \underline{b}$

$\underline{x} = C^{-1}(B\underline{x} + \underline{b})$

Fixpunktform :

Wähle Startvektor x_0

$$\boxed{\underline{x}_{k+1} = C^{-1}(B\underline{x}_k + \underline{b})}$$

▷ Voraussetzungen:

C^{-1} regulär

C sodass, $C\underline{x}_{k+1} = B\underline{x}_k + \underline{b}$ (3) leicht lösbar.

▷ Fehlertheorie:

(2) − (3)

$C(\underline{x}_{k+1} - \underline{x}) = B(\underline{x}_k - \underline{x})$

▷ Def: Fehler: $\underline{e}_k = \underline{x}_k - \underline{x}$

$\Rightarrow C\underline{e}_k = B\underline{e}_k$

$\underline{e}_{k+1} = C^{-1}B\underline{e}_k$

Normen: $\|\underline{e}_{k+1}\| \leq \|C^{-1}B\| \|\underline{e}_k\|$

Satz: Hinreichend für die Konvergenz ist, dass

$\|C^{-1}B\| < 1$

für mindestens eine Norm

(i) Jacobi-Iteration

$A = (a_{ij})$

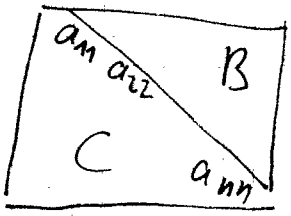
Sei A "diagonal dominant", d.h.

$|a_{ii}| > \sum_{k \neq i} |a_{ik}|$ (4)

Wähle $C = (c_{ij}) = \operatorname{diag}(a_{11}, a_{22}, \dots, a_{nn})$

(4) $\Rightarrow \|C^{-1}B\|_\infty < 1$

(ii) Gauss-Seidel

$A =$ 

konvergiert leicht schneller als Jacobi.

3.2. Konjugierte Gradienten (cg)

M. Hestenes, E. Stiefel ~51, ~52

Sei $A \in \mathbb{R}^{n\times n}$, symmetrisch $A^T = A$

und A positiv definit, d.h. $Q(x) := x^T A x > 0 \quad \forall x \neq 0$

Bem: (0) A hat lauter, positive EW

Beweis: λ EW, v EV $\Rightarrow Av = \lambda v$

$$0 < Q(v) = v^T A v = v^T v \lambda = \underbrace{v^T v}_{\|v\|_2^2} \lambda$$

$\lambda > 0$

Wiederholung:

Iterative Methoden für LGS

additive Zerlegung: $A = C - B$

Algorithmus: Wähle Startvektor $\underline{x}_0$

$$C\underline{x}_{k+1} = B\underline{x}_k + \underline{b} \quad ; k = 0, 1, 2, \ldots$$

$$\left(\underline{x}_{k+1} = C^{-1}B\underline{x}_k + C^{-1}\underline{b} \right)$$

↯ – nie Inverse berechnen

Bsp: für additive Zerlegung: $A = \mathbb{1} - G$

$$\Rightarrow \underline{x}_{k+1} = G\underline{x}_k + \underline{b} \quad \text{(Serie)}$$

Fehlertheorie: $\| \underline{e}_{k+1} \| \leq \underbrace{\| G \|}_{<1} \cdot \| \underline{e}_k \|$

Bsp:

$$\begin{pmatrix} 4 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 5 \\ 6 \\ 5 \end{pmatrix}$$

$4x_1 + x_2 = 5$

$$\Rightarrow x_1^{(k+1)} = (-x_2^{(k)} + 5) \cdot \frac{1}{4}$$

$$x_2^{(k+1)} = (-x_1^{(k)} - x_3^{(k)} + 6) \frac{1}{4}$$

$$x_3^{(k+1)} = (-x_2^{(k)} + 5) \frac{1}{4}$$

Jacobi-Verfahren

besser

Gesamtschrittverfahren

▷ Matrix Darstellung

$$\underline{x}^{(k+1)} = \underbrace{\frac{1}{4}\begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & -1 & 0 \end{pmatrix}}_{\|G\| < 1} \underline{x}^{(k)} + \frac{1}{4}\begin{pmatrix} 5 \\ 6 \\ 5 \end{pmatrix}$$

▷ Gauss-Seidel-Verfahren (Einzelschrittverfahren)

$$x_1^{(k+1)} = (-x_2^{(k)} + 5)\,\frac{1}{4}$$

$$x_2^{(k+1)} = (-x_1^{(k+1)} - x_3^{(k)} + 6)\,\frac{1}{4}$$

$$x_3^{(k+1)} = (-x_2^{(k+1)} + 5)\,\frac{1}{4}$$

$$x_3^{(k+1)} = \left[(+x_1^{(k+1)} + x_3^{(k)} + 6)\,\frac{1}{4} + 5\right]\frac{1}{4}$$

3.2. Das Verfahren der konjugierten Gradienten

▷ Motivation:

Unter Berücksichtigung der Spezialstruktur der Matrix (Besetzungsstruktur: sparsity pattern) das LGS iterativ und möglichst "direkt" zu einer bestimmten Genauigkeit zu lösen.

(Bsp. Serie 5: Schwingende Membran: Laplace Glch. führte auf eine Block-Tridiagonalmatrix)

▷ Ziel: Löse LGS $A\underline{x} + \underline{b} = 0$ mit $A \in \mathbb{R}^{n\times n}$
$A^T = A$ und positiv definit: $\underline{x}^T A \underline{x} > 0$

▷ Bem:

(i) Alle EW von A sind positiv

(ii) Wir suchen ein Minimum der Energie (Energienorm) (Bilder)

▷ Satz: Die Lösung $\underline{x}$ von $A\underline{x} + \underline{b} = 0$ ($A^T = A$, $\underline{x}^T A \underline{x} > 0$, $A \in \mathbb{R}^{n\times n}$) ist das Minimum der quadratischen Funktion

$$F(\underline{u}) := \frac{1}{2}\underline{u}^T A \underline{u} + \underline{u}^T b \qquad (*)$$

▷ Beweis:

$$\operatorname{grad}_{\underline{u}} F(\underline{u}) = \left(\frac{\partial F}{\partial u_1}, \frac{\partial F}{\partial u_2}, \dots, \frac{\partial F}{\partial u_n}\right)^T = A\underline{u} + \underline{b} =: \underline{r}$$

$\underline{r}$ = Residuumvektor zum Vektor $\underline{u}$.

$\operatorname{grad}_{\underline{u}} F(\underline{u}) = 0$ ist notwendige Bedingung für ein Extremum

Minimaleigenschaft:

$\underline{u}, \underline{x} \in \mathbb{R}^n$, $\underline{x} \neq 0$

$$F(\underline{u}+\underline{x}) \overset{(*)}{=} \frac{1}{2}(\underline{u}+\underline{x})^T A(\underline{u}+\underline{x}) + (\underline{u}+\underline{x})^T \underline{b}$$

$$= \frac{1}{2}\left[\underline{u}^T A \underline{u} + \underbrace{\underline{u}^T A \underline{x}}_{\underline{u}^T A^T \underline{x} = (A^T \underline{x})^T \underline{u} = \underline{x}^T A \underline{u}} + \underline{x}^T A \underline{u} + \underline{x}^T A \underline{x}\right] + \underline{u}^T \underline{b} + \underline{x}^T \underline{b}$$

$$= F(\underline{u}) + \underline{x}^T \underbrace{(A\underline{u}+\underline{b})}_{=0 \text{ Extremum}} + \underbrace{\frac{1}{2}\underline{x}^T A \underline{x}}_{>0 \text{ positiv def.}}$$

$$> F(\underline{u})$$

□

▷ Prinzip um Minimum zu finden

Zu einem gegebenen Näherungsvektor (Startvektor) und einem gegebenen, „geeignet festzulegenden Richtungsvektor" $\underline{p} \neq 0$ die Funktion $F(\underline{x})$ in dieser Richtung zu minimieren.

▷ 2D - Minimierung (vergleiche Bilder)

Geg.: Startpunkt $\underline{x}_0$
Suchrichtung $\underline{p}_0$

$$F(\underline{x}_0 + s\underline{p}_0) \overset{!}{=} \min$$

Ges: Parameter $s = s_0 \in \mathbb{R}$, so dass

$$F(\underline{x}_0 + s\underline{p}_0) = \frac{1}{2}(x_0 + sp_0)^T A (x_0 + sp_0) + (x_0 + sp_0)^T b$$

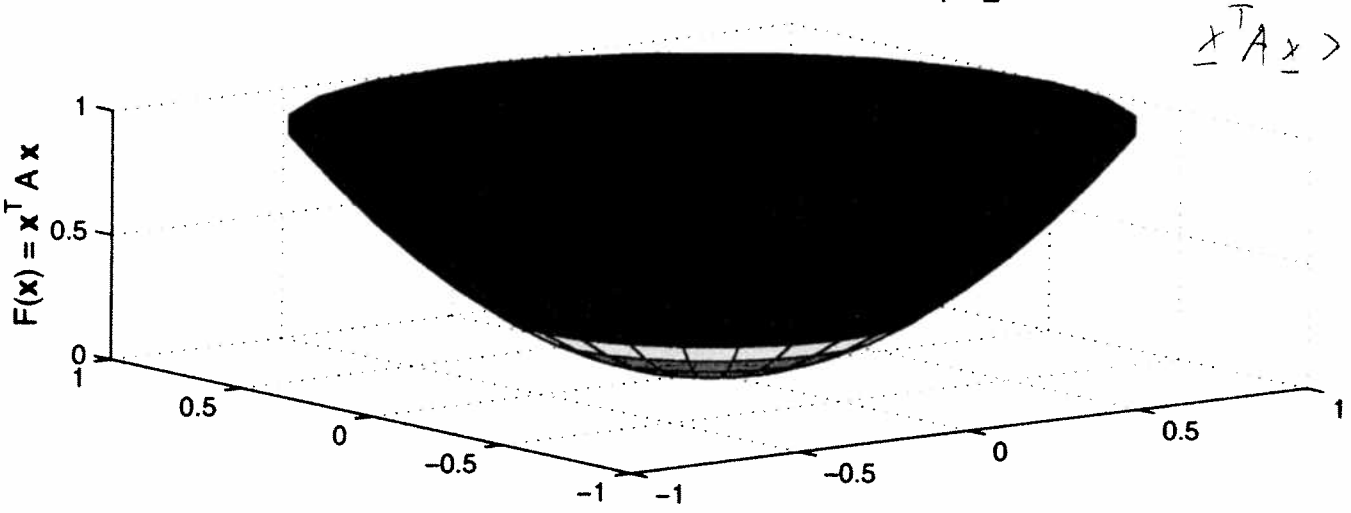
$$= F(x_0) + \frac{1}{2} s^2 p_0^T A p_0 + s p_0^T \underbrace{(Ax_0 + b)}_{= r_0} \equiv F^*(s)$$

nicht ganz wie Paraboloid sollte aber Ellipsoid sein

Bilder ④

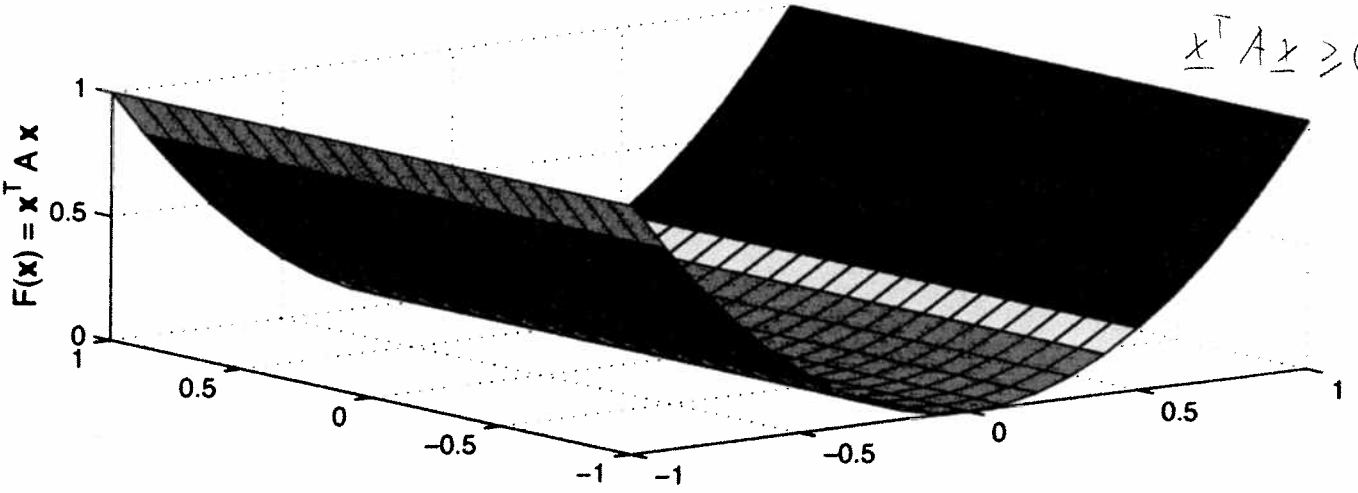
Quadratische Formen in $x = (x_1, x_2)^T$

$\underline{x}^T A \underline{x} > 0$



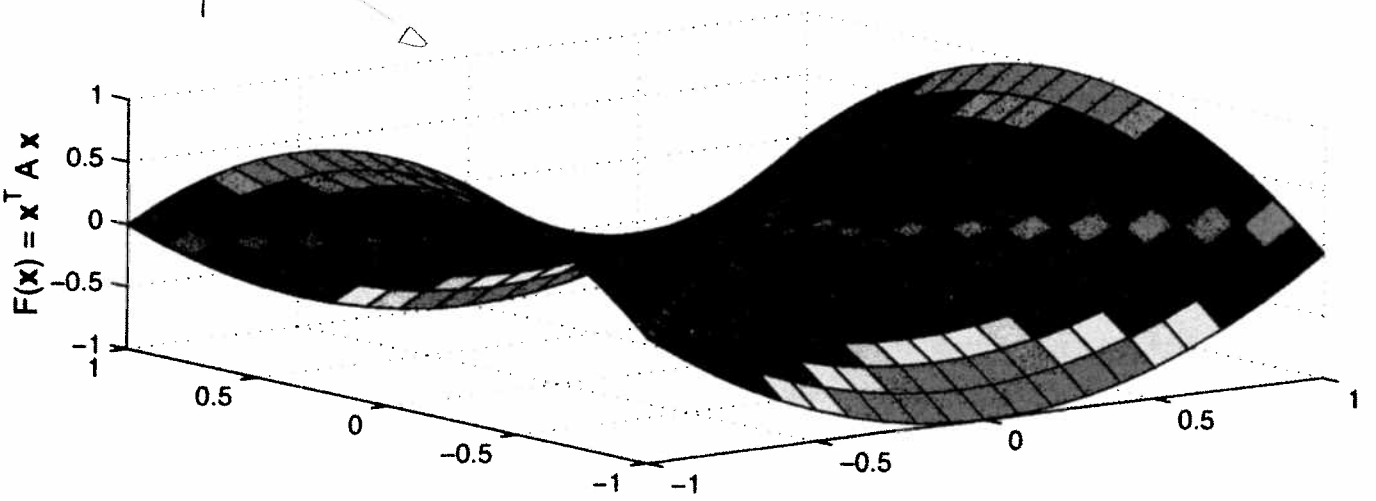
$A = A^T$ positiv definit

$\underline{x}^T A \underline{x} \geq 0$



$A = A^T$ positiv semidefinit

Sattelpunkt



$A = A^T$ indefinit

Bild (2)

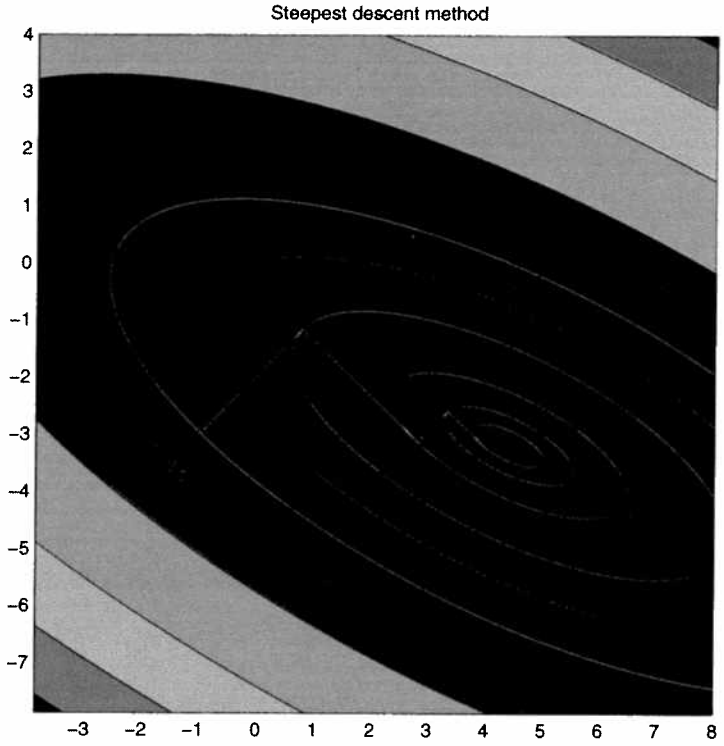


Figure 1: The steepest descent method for $N = 2$.

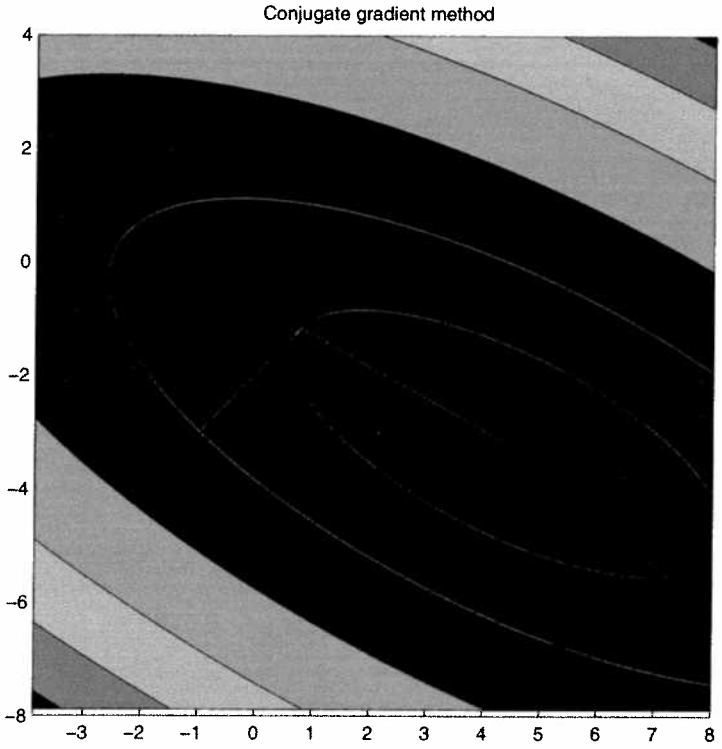


Figure 2: Conjugate directions — the CG method for $N = 2$.

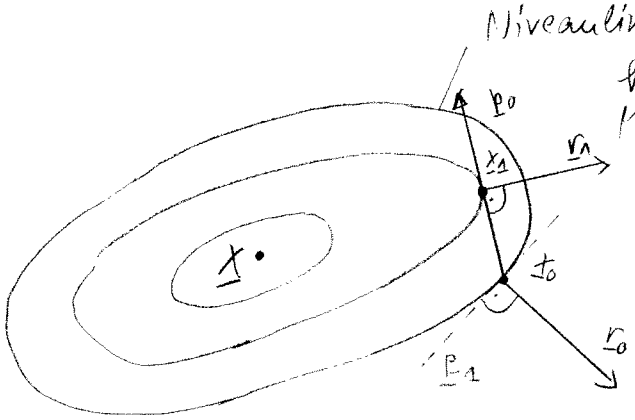
$$\frac{d}{d\varrho} F^*(\varrho) = \varrho\, p_0^T A p_0 + p_0^T r_0 \overset{!}{=} 0$$

$$\Rightarrow \boxed{\varrho_0 = -\frac{p_0^T r_0}{p_0^T A p_0}} \quad \leftarrow \text{Somit haben wir die Länge } \varrho \text{ festgelegt.}$$

$$\rightsquigarrow x_1 = x_0 + \varrho_0 p_0$$

$$\frac{d^2}{d\varrho^2} F^*(\varrho) = p_0^T A p_0 > 0 \quad \text{"Minimum"}$$

▷ Geometrische Interpretation eines Iterationsschrittes



$$x_1 = x_0 + \varrho_0 p_0$$

$$r_1 = A x_1 + b$$

p_1 ist Tangente an die Niveaulinie durch x_1

▷ Wie wählt man den Richtungsvektor p_k

a) $p_k := -r_k = -\operatorname{grad} F(x_k)$

- naheliegend, denn steilster Abstieg, jedoch nicht immer optimal. (siehe Bild) ⟶ Langsame Konvergenz
- langgestreckte Ellipsen (entsprechend cond(A) gross) viele Schritte, obwohl in jeden Iterationsschritt die Richtung gewählt wird, welche die stärkste Abnahme von $F(x)$ garantiert.

⟹ Mehr Information benötigt:

zwei (lin. unabh.) Suchrichtungen

b) Konjugierte Gradienten

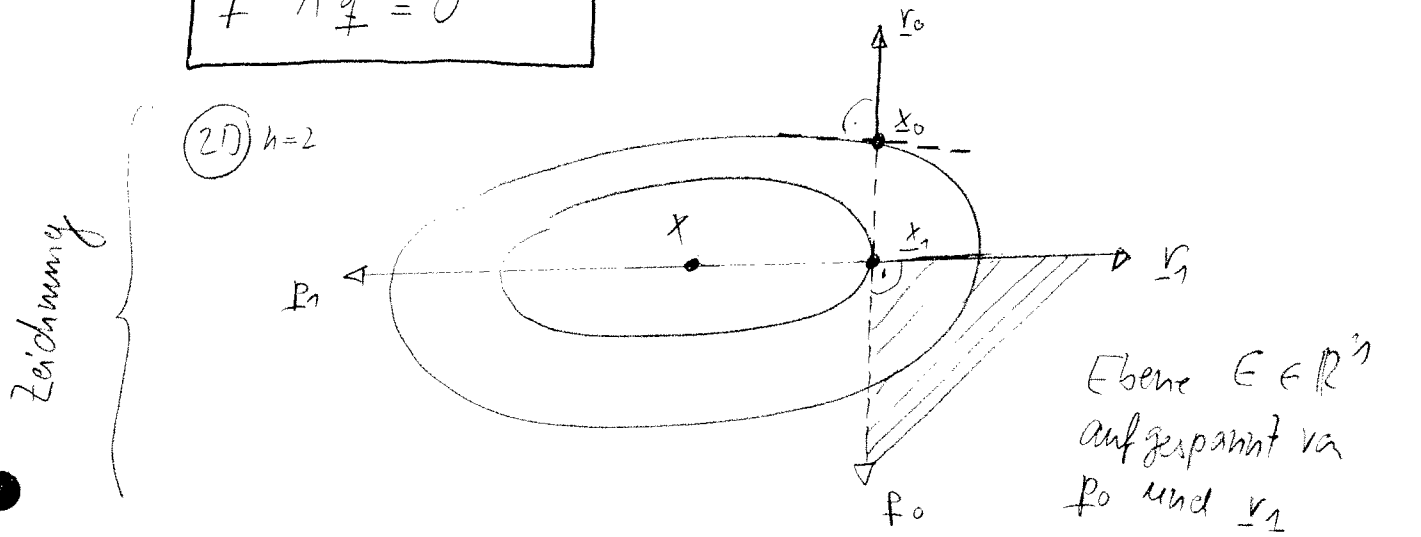
▷ Geometrisch:

Die Richtung p, welche vom Punkt x_k den Mittelpkt x^* der Ellipse trifft, ist mit der Tangentenrichtung im Punkt x_k im Sinn der Kegelschnittgleichung konjugiert.

▷ Def:

Zwei Vektoren (Richtungen) $p, q \in \mathbb{R}^n$ heissen konjugiert (A-orthogonal), falls für A (mit $x^T A x > 0$ und $A^T = A$) gilt

$$p^T A q = 0 \qquad (*)$$



Ebene $E \in \mathbb{R}^n$ aufgespannt v. p_0 und r_1

$p_1 \in \text{span}(p_0, r_1)$

▷ Ziel:

Im k-ten Iterationsschritt das Minimum von $F(x_k)$ bezüglich der Ebene E zu ermitteln.

⇒ Richtungsvektor p_{k+1} muss konjugiert sein zu p_k bezüglich der Schnittellipse und somit auch bezüglich des Ellipsoids $F(x_k)$

Aufgabe { (*) bei gegebenem p_k finde eine Tangentenrichtung p_{k+1} an Ebene E (Niveaufläche $p_{k+1}^T A p_k = \text{const}$)

▷ **Frage:** Wie sieht die $(k+1)$-te Suchrichtung aus?

▷ **Ansatz:**

$$p_1 = -r_1 + \sigma_1 p_0 \; ; \; \sigma_1 = ?$$

p_0, p_1 konjugiert $\Rightarrow$ $0 = p_1^T A p_0 \overset{A=A^T}{=} p_0^T A p_1$

$$= p_0^T A(-r_1 + \sigma_1 p_0) = \sigma_1 p_0^T A p_0 - p_0^T A r_1 = 0$$

$$\Rightarrow \boxed{\sigma_1 = \frac{p_0^T A r_1}{p_0^T A p_0}}$$ (damit haben wir auch p_1 bestimmt)

Gesucht ist aber $x \cong x_2 = x_0 + \underbrace{\rho_0 p_0}_{x_1} + \rho_1 p_1$

aus $\frac{d}{d\rho} F^*(\rho) = 0$ und p_0, p_1 konjugiert

$$\boxed{\rho_1 = -\frac{p_1^T r_1}{p_1^T A p_1}}$$

▷ **cg-Algorithmus**

- **Start:** Wahl von x_0 $(=0)$; $r_0 = A x_0 + b$; $p_{-1} \overset{\text{falls } x_0 = 0}{=} 0$
- **Iteration:** $k = 0, 1, \ldots$

(1)
$$\sigma_k = \begin{cases} \dfrac{p_{k-1}^T A r_k}{p_{k-1}^T A p_{k-1}} & \text{falls } k > 0 \\ 0 & \text{falls } k = 0 \end{cases}$$

Sonst definiere wir ihn uns als $p_{-1} := 0$ ⇒ dann ist gerade $\sigma_0 = 0$

(2)
$$p_k = \sigma_k p_{k-1} - r_k$$

(3)
$$\rho_k = -\frac{p_k^T r_k}{p_k^T A p_k}$$

(4) $\underline{x}_{k+1} = \underline{x}_k + \varrho_k \underline{f}_k$

(5) $\underline{r}_{k+1} = A\underline{x}_{k+1} + \underline{b} = \underline{r}_k + \varrho_k A \underline{f}_k$

▷ Satz: Orthogonalitätsrelationen

- $\underline{r}_{k+1} \perp \underline{f}_k$ (kann man mittels vollständiger)
  $\underline{r}_{k+1} \perp \underline{f}_{k-1}$ (vielleicht in Serie)
  $\underline{r}_{k+1} \perp \underline{r}_k$

⇒ Suchrichtungen $\underline{f}_0, \underline{f}_1$ paarweise konjugiert

⇒ Residuen $\underline{r}_0, \underline{r}_1$ ($\nabla F(\underline{x}_0), \nabla F(\underline{x}_1)$) paarweise orthogonal.

(▷ Satz: Die Residuenvektoren $\underline{r}_k$ bilden ein Orthogonalsystem, und die Richtungsvektoren $\underline{f}_k$ sind paarweise konjugiert. Für $\ell \geq 2$ gelten

$\underline{r}_{k+1}^T \underline{r}_k = 0 \qquad k = 0, 1, \ldots, \ell-2$

$\underline{r}_{k+1}^T \underline{f}_k = 0 \qquad k = 1, 2, \ldots, \ell-1$

$\underline{f}_\ell^T A \underline{f}_k = 0 \qquad k = 1, 2, \ldots, \ell-1$)

⇒ $\underline{r}_0, \underline{r}_1, \ldots, \underline{r}_{n-1} \in \mathbb{R}^n$ bilden ein Orthogonalsystem in $\mathbb{R}^n$

d.h. es kann höchstens n viele Null verschiedene Vektoren enthalten

⇒ $\underline{r}_n = 0$ ⇒ $\underline{x}_n = \underline{x}$ exakte Lösung

c) Vorkonditionierung

▷ **Def:** $\underline{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$, $\underline{x}^T A \underline{x} > 0$

A-Norm von $\underline{x}$: $\| \underline{x} \|_A := \sqrt{\underline{x}^T A \underline{x}}$ "Energienorm"

(siehe S. 10)

▷ **Satz:** Konvergenzabschätzung des cg-Verfahrens (ohne Beweis)

$\underline{e}_k = \underline{x}_k - \underline{x}$;

$$\frac{\| \underline{e}_k \|_A}{\| \underline{e}_0 \|_A} \leq 2 \left(\frac{\sqrt{\mathrm{cond}_2(A)} - 1}{\sqrt{\mathrm{cond}_2(A)} + 1} \right)^k \qquad (*)$$

Annahme: $\| \underline{e}_k \|_A / \| \underline{e}_0 \|_A \leq \varepsilon$

$\overset{(*)}{\Longrightarrow}$ $\boxed{k \leq \frac{1}{2} \sqrt{\mathrm{cond}_2(A)} \ln\left(\frac{2}{\varepsilon}\right) + 1}$

$\rightsquigarrow$ cond (A) entscheidend für Konvergenzgüte !

▷ **Skalierung (Vorkonditionierung)**

neues LGS:

$$\tilde{A} \tilde{\underline{x}} + \tilde{\underline{b}} = 0 \qquad (1)$$

$$\left.\begin{aligned} \tilde{A} &= C^{-T} A C^{-1} \\ \tilde{\underline{x}} &= C \underline{x} \\ \tilde{\underline{b}} &= C^{-T} \underline{b} \end{aligned}\right\} \quad (2)$$

C regulär und „günstig"

▷ **Ziel:** $\mathrm{cond}_2(\tilde{A}) \approx 1$

cond ≡ κ

Damit $\kappa_2(\tilde{A}) = \kappa_2(C^{-T} A C^{-1}) \ll \kappa_2(A)$ ist,

erhalten wir aus der Ähnlichkeitstransformation von $\tilde{A}$:

$$C^{-1} \tilde{A} C = C^{-1} C^{-T} A \underbrace{C^{-1} C}_{= I} = \underbrace{(C^T C)^{-1}}_{:= M} A$$

$M = C^T C$ ist symmetrisch und positiv definit

↑ Vorkonditionierungsmatrix

Wegen Ähnlichkeit:

$$\kappa_2(\tilde{A}) = \frac{\lambda_{max}(M^{-1} A)}{\lambda_{min}(M^{-1} A)}$$

▷ cg-Algorithmus mit Vorkonditionierung (siehe Skript)

Einführung einer zusätzlichen Vektorfolge $\underline{z}_k : \underline{r}_k$

$$\underline{z}_k = M^{-1} \underline{r}_k$$

▷ Nachtrag zur Energienorm:

$$\| \underbrace{\underline{x}_k - \underline{x}}_{\underline{e}_k} \|_A^2 = (x_k - x)^T A (x_k - x)$$

$$= x_k^T A x_k - 2 x_k^T A x + x^T A x$$

$$= x_k^T A x_k + 2 x_x^T b + (A^{-1} b)^T b$$

$$= 2 F(x_k) + (A^{-1} b)^T b$$

d.h. die Iterierte $\underline{x}_k$ minimiert den Fehler $\underline{e}_k$ in der Energienorm!

4. Gewöhnliche Differentialgleichungen (ODE)

4.1. Problemstellung und Bsp

i) Radioaktiver Zerfall

$m(t)$:= Menge radioaktiven Materials zur Zeit t

$$\boxed{\frac{dm(t)}{dt} = -\lambda m(t)} \quad ; t \in \mathbb{R}$$

(λ: Proportionalitätsfaktor)

Lösungen: $m(t) = A e^{-\lambda t}$ (allgemein)

$m(t_0) = m_0$ (vorhandene Menge des Materials zum Zeitpkt. t_0)

$$m(t) = m_0 e^{-\lambda(t-t_0)} \quad \text{(speziell)}$$

▷ Def: Anfangswertaufgabe $y : t \in \mathbb{R} \longrightarrow y(t) \in \mathbb{R}$

Gesucht ist eine Funktion $y(t)$, so dass gilt

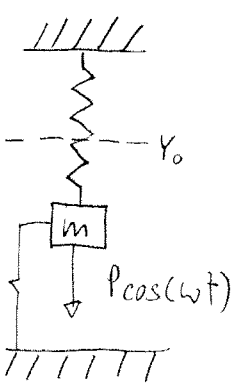
$$\begin{cases} \frac{dy(t)}{dt} = f(t, y(t)), & \text{(AWP)} \\ y(t_0) = y_0 \end{cases}$$

($\frac{dy(t)}{dt} = y'(t)$) DGL 1. Ordnung

ii) Erzwungene Schwingung eines stark gedämpften Schwingers

$$\boxed{m \frac{d^2y}{dt^2} + g \frac{dy}{dt} + c(y - y_0) = P\cos(\omega t)}$$

(ω: Kreisfrequenz)



- m: Masse
- $g \frac{dy}{dt}$: Dämpfungskraft
- $c(y - y_0)$: Rückstellkraft
- y_0: Ruhelage
- $P\cos(\omega t)$: Störkraft

⟿ DGL 2ter Ordnung

Spezielle Werte für m, δ, c, y_0, P und ω:

$$\Rightarrow \text{(AWP)} \begin{cases} \frac{d^2y}{dt^2} + 200\,\frac{dy}{dt} + 156.25\,y = 80\cos(t) + 156.25 \\ \text{Anfangszustand (Bedingung)} \\ y(0) = 5 \\ y'(0) = -100 \end{cases}$$

Lösungsmethode: Variablensubstitution um die DGL höherer Ordnung auf System 1. Ord. zurückzuführen.

$y_1 := y$
$y_2 := y'$ mit $\underline{y}(t) = \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} \in \mathbb{R}^2$

$$\Rightarrow \begin{cases} y_1'(t) = y_2(t) \\ \frac{dy_2}{dt} = -156.25\,y_1(t) - 200\,y_2(t) + 80\cos(t) + 156.25 \\ y_1(0) = 5 \\ y_2'(0) = -100 \end{cases}$$

Allgemein wird eine DGL n-ter Ordnung in ein System erster Ordnung umgeschrieben.

(iii) Autonome DGL

$$y' = f(y)$$

↑ t kommt nicht vor, d.h. $\frac{\partial f}{\partial t} = 0$

▷ Satz: Für $y'(t) = f(y)$ gilt:

Mit $y(t)$ ist auch $y(t - t_0)$ eine Lösung.

▷ zurückführen ein allgemeines AWP auf ein autonomes

$$\begin{cases} y'(t) = f(t, y(t)) \;;\; y \in \mathbb{R}^n & \text{"Nicht-autonom"} \\ y(t_0) = y_0 \end{cases}$$

Definition: $\underline{Y} := \begin{pmatrix} t \\ y \end{pmatrix} \in \mathbb{R}^{n+1}$; $\underline{F}(\underline{Y}) := \begin{pmatrix} 1 \\ f(t, y) \end{pmatrix}$

$$\begin{cases} \frac{d}{d\tau} \underline{Y}(\tau) = \underline{F}(\underline{Y}(\tau)) & \text{"} \qquad \text{"} \\ \underline{Y}(0) = \begin{bmatrix} t_0 \\ y_0 \end{bmatrix} \end{cases}$$

neues autonomes AWP mit unabhängiger Variable: $\tau = t - t_0$

(iv) Existenzsatz für Differentialgleichungen

▷ Satz: (Existenz) $[a,b] \times \mathbb{R}^n$

Es sei $f(t, y) : D_f \subset \mathbb{R}^{n+1} \to \mathbb{R}^n$ eine stetige Fkt.

Dann besitzt die AWAufgabe (AWP) eine Lösung, die bis an den Rand von D_f fortgesetzt werden kann.

▷ Satz: (Eindeutigkeit)

Es sei $f(t, y) : D_f \subset \mathbb{R}^{n+1} \to \mathbb{R}^n$ stetig in D_f und $f(t, y)$ erfülle eine Lipschitzbedingung bezüg y, d.h.

es gibt eine Konstante $L > 0$, so dass:

$$\boxed{\| f(t,y) - f(t,\tilde{y}) \| \leq L \| y - \tilde{y} \|}$$

$\forall (t,y) \in D_f$ und $(t,\tilde{y}) \in D_f$. Dann hat das AWP genau eine Lösung.

Bsp:

$$\begin{cases} y'(t) = \sqrt{y} \\ y(0) = 0 \end{cases}$$ erfüllt nicht Lipschitzbed.

$$\Rightarrow \begin{cases} y_1(t) = 0 \\ y_2(t) = t^2 \end{cases}$$ keine Eindeutigkeit!

4.2. Taylor- und Runge-Kutta-Verfahren

a) Das Euler-Verfahren

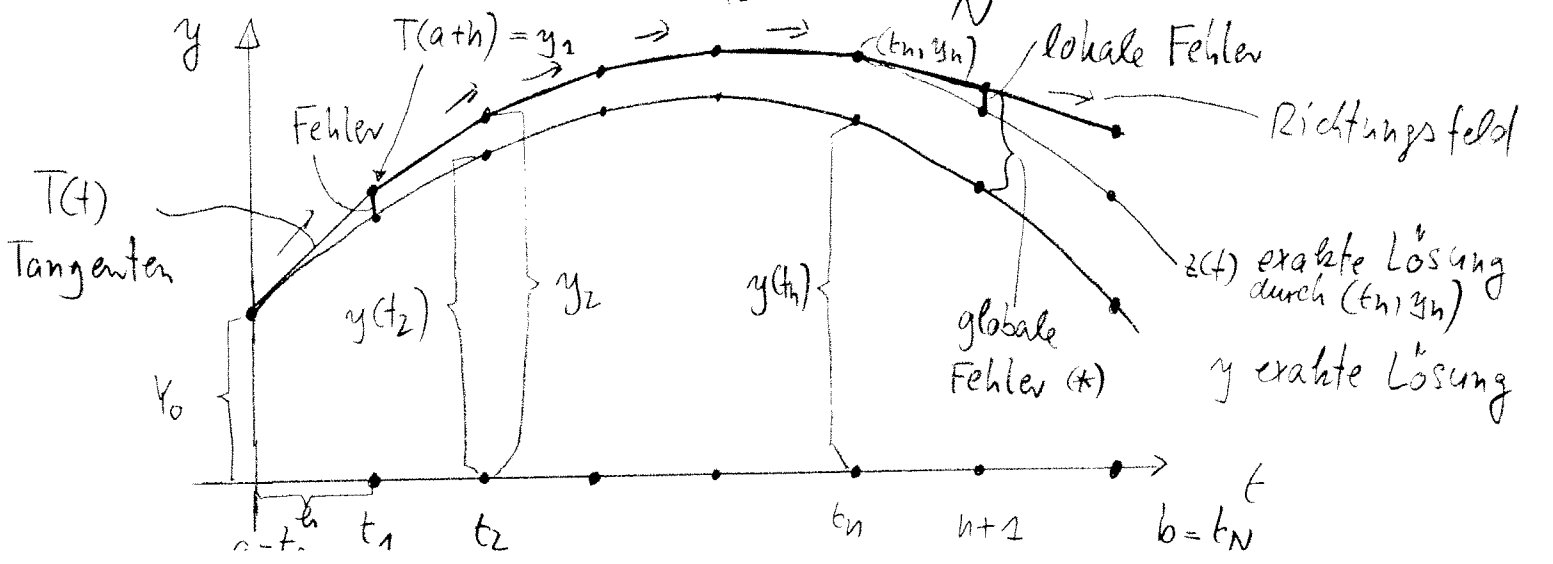
$$\begin{cases} y'(t) = f(t, y(t)) \; ; t \in [a,b] \\ y(t_0) = y_0 \end{cases}$$

(*) gibt die Fehlerordnung an

Diskretisation:

$$t_i = t_0 + i \cdot h \qquad i = 0, 1, \dots, N$$

$\uparrow$ Schrittweite $h = \frac{b-a}{N} = \text{const}$



- Die exakte Lösung $y(t_i)$ wird durch numerische Werte y_i approximiert.

- Richtungsfeld tangential zur Lösungskurve

$$y(t_1) = y(a+h) \sim y_1 = y_0 + h y_0' = y_0 + h f(t_0, y_0)$$

$y(t_1)$ ist nicht bekannt, also beim nächsten Schritt muss man von (t_1, y_1) ausgehen.

$$y(t_2) \sim y_2 := y_1 + h f(t_1, y_1)$$

$\Rightarrow$ Euler-Verfahren für Vekt-DGL

geg: y_0 (AW)

Für $i = 0, 1, \dots N$ berechne man

$$\underline{y}_{i+1} = \underline{y}_i + h \underline{f}(t_i, y_i)$$

$$t_{i+1} = t_i + h$$

28.04.2008

Numerik der gewöhnlichen Diff-Glch

$$y'(t) = F(t, y(t))$$

↖ gegeben, „rechte Seite"

$y(t=0) = y_0$ gegeben

$y(t) = ?$

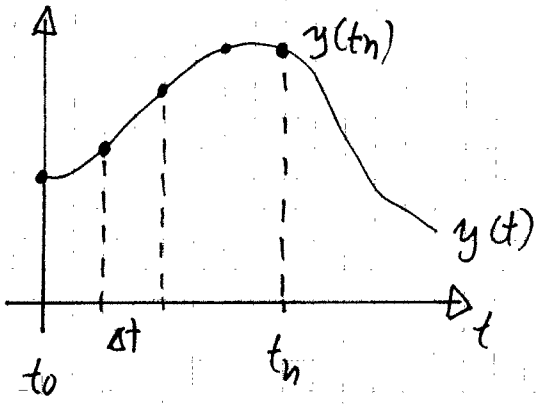
Idee (Numerik): Diskretisierung $t \in [0, T_{end}]$

$t_0 = 0$; $t_{n+1} = t_n + \Delta t_n$

Δt_n: Schrittweiten

Hier $\Delta t_n = \Delta t$ ($\forall n$, konstante Schrittweite)

$y(t)$ wird approximiert durch $\{y(t_n)\}_{n \in [0,1,2,\dots]}$

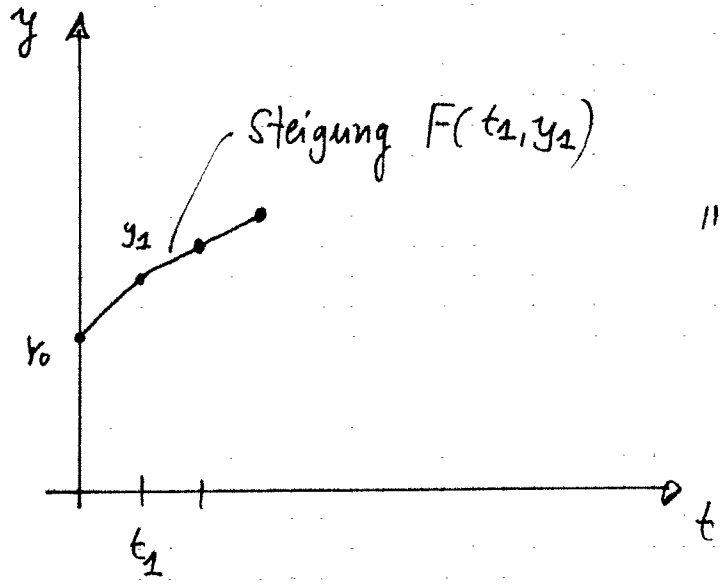


$y_n := y(t_n)$

(1) Euler Methode

$$\underbrace{\frac{y_{n+1} - y_n}{\Delta t}}_{y'(t_n)} = F(t_n, y_n)$$

$$\rightsquigarrow \boxed{y_{n+1} = y_n + \Delta t\, F(t_n, y_n)}$$



"explizites Euler"

(2) $$\frac{y_{n+1} - y_n}{\Delta t} = F(t_{n+1}, y_{n+1})$$ "implizites Euler"

$$\boxed{y_{n+1} = y_n + \Delta t\, F(t_{n+1}, y_{n+1})}$$

Bestimmungsgleichung für y_{n+1}. Nicht linear!

(Newton-Verfahren)

(3) $$\frac{y_{n+1} - y_n}{\Delta t} = F(t_{n+1/2}, y_{n+1/2})$$ "Mittelpunkts-regel"

$$\| \quad \frac{1}{2}(y_n + y_{n+1})$$

$$\boxed{y_{n+1} = y_n + \Delta t\, F\left(t_{n+1/2}, \tfrac{1}{2}(y_n + y_{n+1})\right)}$$ "implizit"

(4) $$\begin{cases} y^{(*)}_{n+\frac{1}{2}} = y_n + \frac{\Delta t}{2} F(t_n, y_n) \\ y_{n+1} = y_n + \Delta t\, F(t_{n+1/2}, y^{(*)}_{n+1/2}) \end{cases}$$

"explizite Mittelpunktsregel"

(5) $F(t_{n+1/2}, y_{n+1/2}) \hat{=} F_{n+1/2} = \frac{1}{2}\left[F(t_n, y_n) + F(t_{n+1}, y_{n+1})\right]$

$$y_{n+1} = y_n + \Delta t \frac{1}{2}\left[F(t_n, y_n) + F(t_{n+1}, y_{n+1})\right]$$

Trapezregel!

Struktur: Runge - Kutta - Methoden

$$k_i = F\left(t_n + c_i \Delta t,\ y_n + \Delta t \sum_{j=1}^{s} a_{ij} k_j\right)$$

$$y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i k_i$$

Jedes Verfahren ist durch die Werte a_{ij}, b_i, c_i gegeben.

Butcher - Tabelle

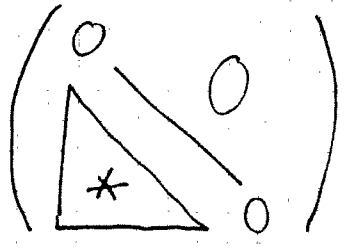
c_1	a_{11}	a_{12}
$\vdots$	a_{21}	a_{22} $\ddots$
	$\vdots$	
c_s	a_{s1}	$\cdots$
	b_1	$\cdots$ b_s

z. Bsp.

0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0
	0	1

(4)

„s" ist die Anzahl F-Auswertungen, Stufen.

$(a_{ij}) =$  $\Longleftrightarrow$ Verfahren explizit

Bsp:

0	0	0	0	0
1/2	1/2	0	0	0
1/2	0	1/2	0	0
1	0	0	1	0
	1/6	1/3	1/3	1/6

Vierstufige RK-Verfahren

▷ Konvergenz (globale Fehler)

$g_n := y_n - y(t_n)$ (am Ende $t_n = n\Delta t = T_{end}$)

y_n: approx. $y(t_n)$: exakt

▷ Def: Das Verfahren konvergiert falls

$$\lim_{\Delta t \to 0} \| g_n \| = 0$$

(Achtung: $n = \frac{T_{end}}{\Delta t} \to 0$)

Konvergenzordnung $p \iff \| g_n \| = \sigma(\Delta t^p) \quad \Delta t \to 0$

▷ Satz: Konsistenz + Stabilität ⇒ Konvergenz

- Konsistenz gibt an wie gut ein einzelner Schritt ist.
- Stabilität gibt an ob ein Fehler für viele Schritte klein bleibt.

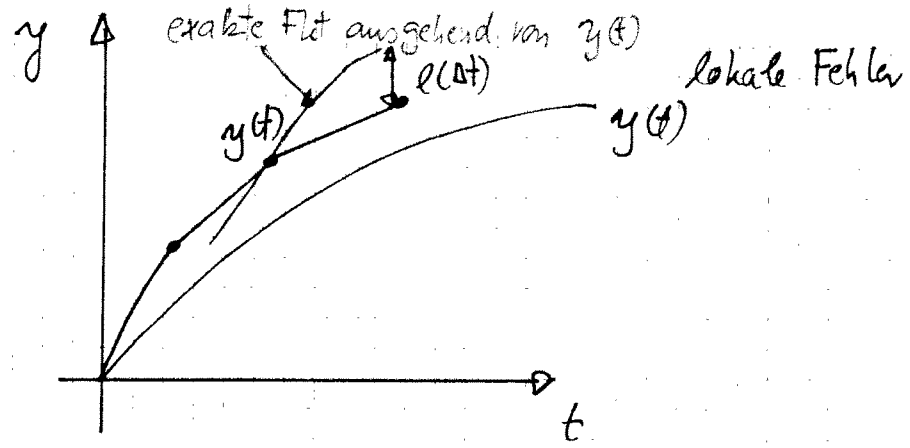
▷ Konsistenz: Abstrahtes Verfahren:

$$y_{n+1} = y_n + \Delta t \, \underbrace{\Phi(t_n, y_n, \Delta t)}_{\text{Einschritt-Methode}}$$

Φ heisst Verfahrensvorschrift.

▷ Def: (Lokaler Fehler) $y(t)$ ist exakt

$$l(t, \Delta t) = \underbrace{y(t) + \Delta t \, \Phi(t, y(t), \Delta t)}_{\text{Ein Schritt ausgehend von } y(t)} - \underbrace{y(t+\Delta t)}_{\text{exakt}}$$



▷ lokaler Fehler

$$l(t,\Delta t) = y(t) + \Delta t\, \Phi(t, y(t), \Delta t) - y(t+\Delta t)$$

$$\| l(t,\Delta t) \| = \sigma(\Delta t^{p+1}) \iff p \text{ Konsistenzordnung}$$

Bsp:

$$y_{n+1} = y_n + \Delta t\, F(y_{n+1/2}) \quad ; \quad y_{n+1} = y_n + \frac{\Delta t}{2} F(y_n)$$

angenommen $y(t)$ ist exakte Lsg. $\boxed{y'(t) = F(y(t))}$

$$l(t,\Delta t) = \underbrace{y(t)}_{\hat{=} y_n} + \Delta t\, F\Big(\underbrace{y(t)}_{\hat{=} y_n} + \frac{\Delta t}{2} F\big(\underbrace{y(t)}_{\hat{=} y_n}\big)\Big) - y(t+\Delta t)$$

Taylor Δt klein

$$l(t,\Delta t) = y(t) + \Delta t\Big[F(y(t)) + F'(y(t)) \cdot \frac{\Delta t}{2} F(y(t)) + \sigma(\Delta t^2)\Big] - y(t+\Delta t)$$

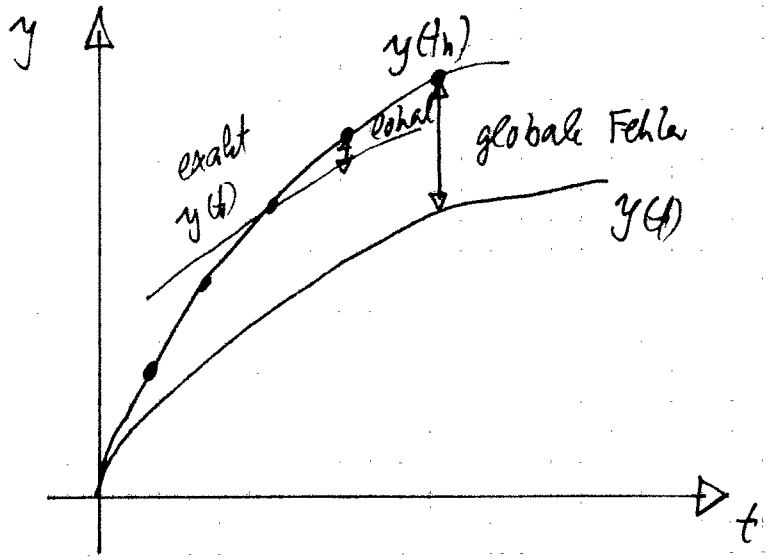
$$y(t+\Delta t) = y(t) + \underbrace{y'(t)}_{\| F(y(t))} \Delta t + \underbrace{y''(t)}_{\| F'(y(t)) \cdot F(y(t))} \frac{\Delta t^2}{2} + \sigma(\Delta t^3)$$

$\rightsquigarrow \quad l(t, \Delta t) = \mathcal{O}(\Delta t^3)$

Konsistenzordnung : $p = 2$; Euler : $p = 1$

Runge-Kutta : $p = 4$

Falls die Konsistenzordnung p ist und das Verfahren stabil, dann ist die Konvergenzordnung auch p.



Der lokale Fehler summiert sich auf.
Deshalb $l = \mathcal{O}(\Delta t^{p+1})$
aber $g = \mathcal{O}(\Delta t^p)$

▷ Stabilität:

Wir betrachten die lineare Test Gleichung.

$y' = \lambda y$ (z. Bsp. durch Linearisierung)

$y' = F(y) = \underbrace{F'(0)}_{\hat{=}\lambda} y$; $y \ll 1$ $\quad \lambda \in \mathbb{C}$ erlaubt

▷ im Systemfall ist λ Eigenwert der Jacobi $F'(y_0)$

Lsg: $y(t) \sim e^{\lambda t}$

Weiter: Wir wählen ein Verfahren auf $y' = \lambda y$ an.

$$y_{n+1} = R(\lambda \Delta t)\, y_n$$

$z := \lambda \Delta t$

$R(z)$ heisst Verstärkungsfaktor

R verstärkt oder dämpft auch den Fehler $\rightsquigarrow$ Stabilität

▷ Def: (Stabilität)

Falls, für eine abfallende Lsg. d.h. $Re(\lambda) < 0$, für den Verstärkungsfaktor $R(z)$ die Bedingung $\| R(z) \| \leq 1$ erfüllt, so ist das Verfahren (linear) stabil.

▷ Bsp: Euler (explizit)

$$y_{n+1} = y_n + \Delta t\, F(y_n) \quad ; \; y' = \underbrace{\lambda y}_{F(y)}$$

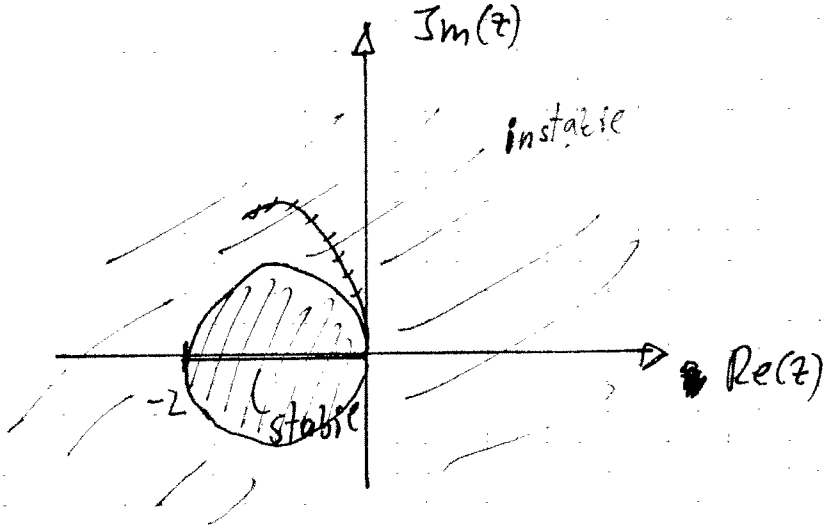
$$= y_n + \Delta t\, \lambda y_n$$

$$= \underbrace{(1 + \Delta t \lambda)}_{R(\lambda \Delta t)}\, y_n$$

$\rightsquigarrow R(z) = 1 + z$;

▷ Bedingung: Für $\lambda < 0$ muss $|R(z)| \leq 1$

$$|1 + z| \leq 1 \quad ; \; -2 \leq z \leq 0$$



Wähle: $\lambda \Delta t < 0$ ✓

$\lambda \Delta t > -2 \quad \rightsquigarrow \quad \Delta t < \frac{2}{|\lambda|}$

✱ Ist nur stabil für hinreichend kleine Zeitschritte.

▷ Def: (Stabilitätsgebiet)

$$S := \{ z \in \mathbb{C} \mid |R(z)| \leq 1 \}$$

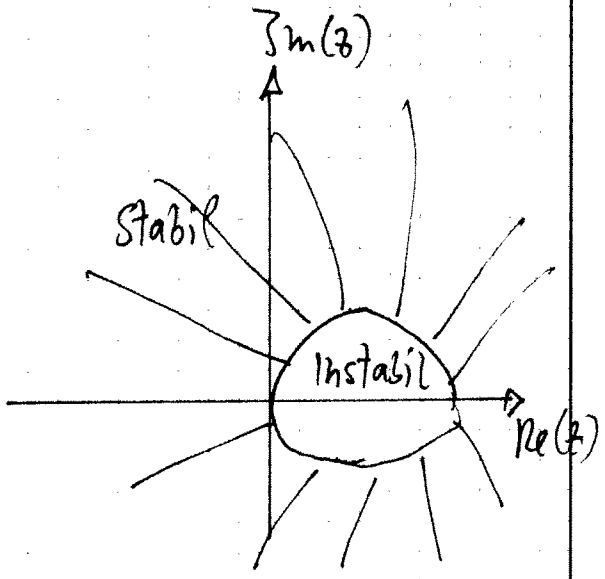
Bsp: „impliziter Euler"

$$y_{n+1} = y_n + \Delta t \lambda y_{n+1}$$

$$y_{n+1} = \frac{1}{1 - \Delta t \lambda} y_n$$

$$\rightsquigarrow R(z) = \frac{1}{1-z}$$

Hier keine Zeitschrittbeschränkung.



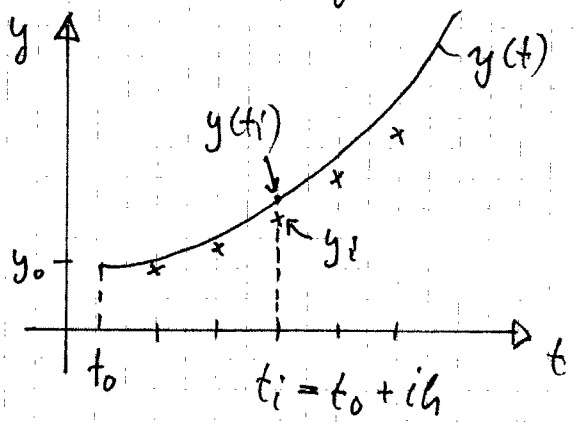
AWA gew. Diffgl.

$$y' = f(t, y(t)) \qquad (1)$$

Anfangsbedingung $y(t_0) = y_0$ (2)

Ges: $y(t)$ mit

Diskretisierung



$y_i \sim y(t_i)$

num. appr. exakt

h Schrittweite

lokaler Fehler: $l(t_0, h) = y(t_0+h) - y_h = \mathcal{O}(h^{p+1})$

p = Ordnung

Konsistenz: $p \geq 1$

4.4. Steife Diffgl.

Testgleichung: $y' = qy \qquad q \in \mathbb{C}$

Einschrittverfahren:

$$y_{n+1} = \underbrace{V(qh)}_{\text{Verstärkungsfaktor}} y_n \quad ; \quad qh \hat{=} z \qquad n = 0, 1, 2, \ldots \infty$$

Stabilitätsgebiet:

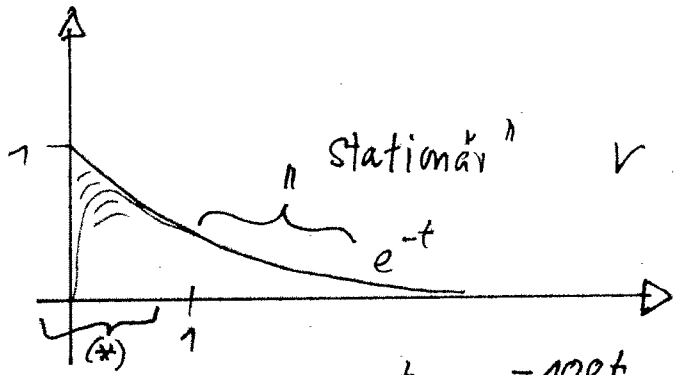
$$S = \{ z \in \mathbb{C} \mid |V(z)| \leq 1 \}$$

$$= \{ z \in \mathbb{C} \mid \{y_n\}_{n=0,1,\ldots} \text{ beschränkt} \}$$

Warum?

a) Einführungsbsp:

$$\begin{cases} y'' + 101\, y + 100\, y = 0 \\ y(0) = 0 \\ y'(0) = 99 \end{cases}$$



ν völlig unwichtig

Exakt: $y(t) = e^{-t} - e^{-100t}$

(*) transiente Phase.

umschreiben in ein System

$y = y$
$z = y'$

$$\begin{pmatrix} y \\ z \end{pmatrix}' = \begin{pmatrix} 0 & 1 \\ -101 & 100 \end{pmatrix} \begin{pmatrix} y \\ z \end{pmatrix}$$

EW von $\begin{pmatrix} 0 & 1 \\ 101 & 100 \end{pmatrix}$

$q_1 = -100$; $q_2 = -1$

Diagonalisierbar:

$u' = -u$: $h, \emptyset$

$v' = -100v$: $h, \emptyset$

Num. Lösung mit Euler:

$y' = qy$; $q_1 = -100$; $q_2 = -1$

$y_{(41} = (1 + hq)^{(}$

Stabilität:

$$\left.\begin{array}{l} |1 + hq_1| \leq 1 \\ |1 + hq_2| \leq 1 \end{array}\right\} \Rightarrow \quad 0 < h \leq \frac{2}{-q_1}$$

$$y(0,2) = e^{-0,2} - e^{-20} = \underbrace{0,81873\ldots}_{u} \underbrace{- 2 \cdot 10^{-9}}_{v}$$

(i) Für $t \geq 0,2$ möchte man mit grösserem Schritt rechnen, angepasst an $-q_2 h_2 = 0,1 \rightarrow h_2^* = 0,1$.

Sobald man das macht wird $1 + q_1 h^* = -9$.

$\Rightarrow$ Der unwesentliche Beitrag zur Lösung erzwingt

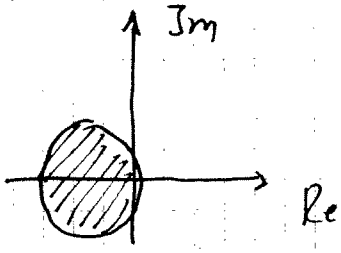
viel zu kleinen Schritt. ⇒ Steifheit.

⇒ Stabilitätsgebiet.

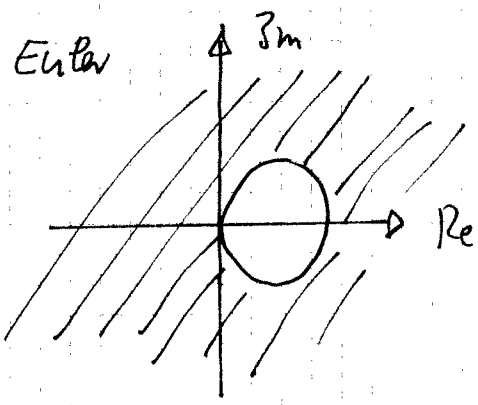
b) θ - Verfahren

$$y_{n+1} = y_n + h\left[(1-\theta)\, f(t_n, y_n) + \theta\, f(t_{n+1}, y_{n+1})\right]$$

$\theta = 0$, Euler



$\theta = 1$, imp. Euler



Genauigkeit:

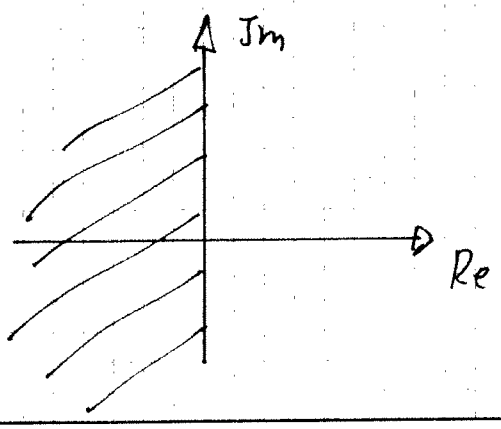
$$(1 + hq) \sim e^{hq}$$

anwenden auf $\theta - \frac{1}{2}$ (Trapezregel)

$$y_{n+1} = y_n + \frac{h}{2}\left[\underbrace{f(t_n, y_n)}_{q y_n} + \underbrace{f(t_{n+1}, y_{n+1})}_{q y_{n+1}}\right]$$

auf $y' = q\lambda$

$$y_{n+1} = \underbrace{\frac{1 + z/2}{1 - z/2}}_{V(z)}\, y_n$$



c) Auswirkung des Stabilitätsgebietes auf die Wahl der Schrittweite

$$y' = Ay$$

$A \in \mathbb{R}^{m \times m}$ diagonalisierbar

d.h. $y' = T^{-1} \underbrace{\Lambda}_{\text{diag}(q_1, q_2, \dots, q_m)} T y \qquad (1)$

zur Analyse umtransformieren

$$z = Ty$$

$$z' = Ty' = \underbrace{TT^{-1}}_{\mathbb{I}} \Lambda \underbrace{Ty}_{z} \qquad (2)$$

$$z' = \Lambda z$$

$$\left.\begin{aligned} z^{1\prime} &= q_1 z^{-1} \\ z^{2\prime} &= q_2 z^{2} \end{aligned}\right\} \text{entkoppelt}$$

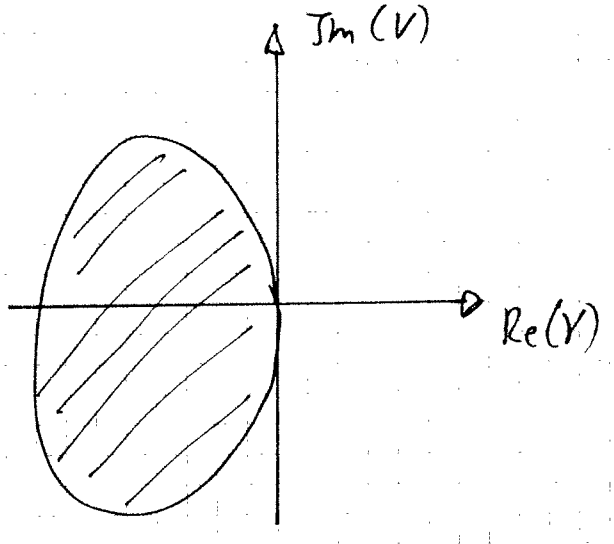
(1) $\xrightarrow{z = Ty}$ (2)

$\downarrow$ Wende Verf. Φ an $\qquad \downarrow$

$\{y_n\} \xrightarrow{\tilde{z}_n = T y_n} \{\tilde{z}_n\} \quad \{z_n\}$

Bei praktisch allen Verfahren gilt $\{\tilde{z}_n\} \equiv \{z_n\}$

Typisches Stab.gebiet eines expliziten Verfahrens



Im der transienten Phase müssen alle Komponenten "genau" berechnet werden. Später ist $z^{(1)}$ unwichtig. Trotzdem muss $q_1 h \in S$ liegen.

Steife Probleme : $\Rightarrow$ verwende implizite Verfahren.

A - Stabilität:

$$\mathbb{C}^- = \{ z \in \mathbb{C} \mid Re(z) \leq 0 \} \subset S$$

Imp. Euler : A - Stabil ; $p = 1$

Trapezregel : - " - ; $p = 2$

d) Rückwärtsdifferentiations Formeln (BDF)

Wähle Interpolationspolynom $P(t)$ durch die Pkte.

(t_i, y_i) ; $i = -k+1, \ldots, 1$

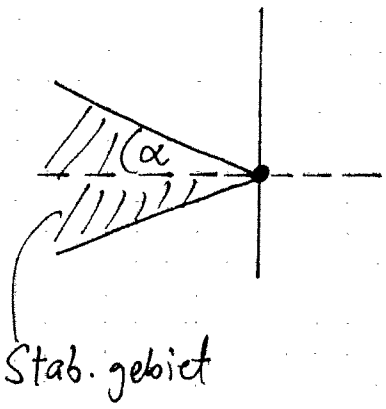
$P'(t_{n+1}) = f(t_{n+1}, P(t_{n+1}))$

$\Rightarrow$ BDFormeln

$k = 1$: impliziter Euler

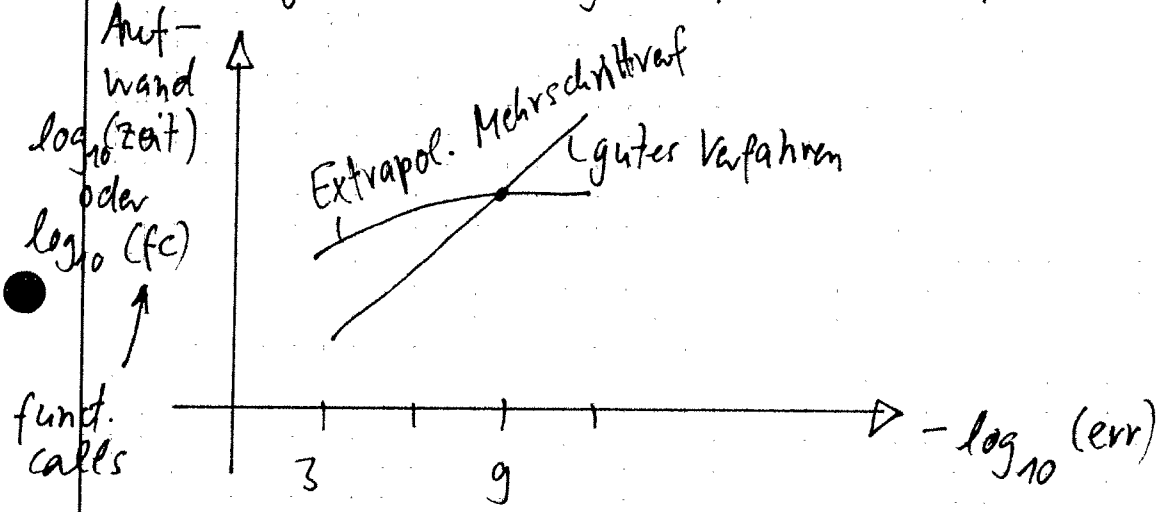
$$\sum \alpha_{ki}\, y_{n+i} = f(t_{n+1}, y_{n+1})$$

$A(\alpha)$ - stabil



e) Zur Praxis

Diagram: Genauigkeit, versus Aufwand



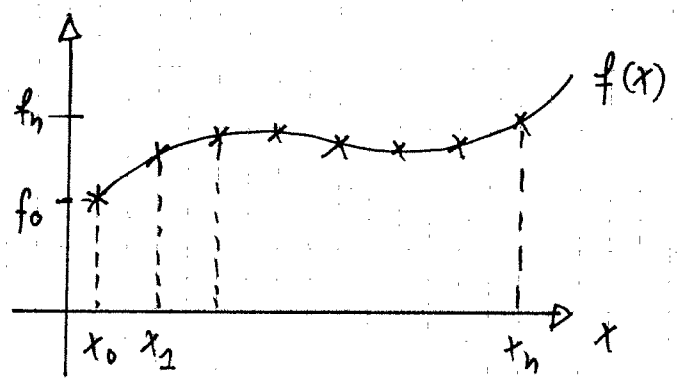
6. Interpolation

19.05.2008

Motivation : Vorallem zum Entwickeln von Verfahren, z. Bsp. Rückwärtsdifferentiations Formeln.

6.1. Problemstellung

Rekonstruktion, Approximation einer Funktion aus Funktionswerten.



$f(x_i) \hat{=} f_i$
diskrete Werte von $f(x)$

Gegeben: $n+1$ Stützstellen x_i

$n+1$ Funktionswerte, f_i ; $i = 0, 1, \ldots, n$

Gesucht: Approximation $p(x)$ für $f(x)$

mit : $\boxed{p(x_i) = f_i}$, (1) $i = 0, 1, 2, \ldots n$

heisst : Interpolationsbedingungen

Es gibt ∞-viele Lösungen. Wähle Funktionsklasse aus.

z. Bsp. trigonometrisches Polynom

Einfachster Fall :

$p(x)$ Polynom vom Grad $\leq n$

$n+1$ Parameter

$$\boxed{P(x) := a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n} \quad (2)$$

Gibt es eine eindeutige Lösung?

6.2. Lagrangepolynom

Wie soll man es *nicht* machen.

Interpolationsbedingung:

$$\Rightarrow \begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ & & & & \\ 1 & x_i & x_i^2 & \dots & x_i^n \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_i \\ \vdots \\ f_n \end{pmatrix}$$

$\hat{=}$ Van der Mondematrix V

Bem: • V ist oft schlecht konditioniert.

• Gausselim. $\frac{n^3}{3}$-Operationen. (zu teuer)

Gute Methode: Basisfunktionen von Lagrange. $B_k(x)$

$$(3) \quad B_k(x_\ell) = \begin{cases} 1 & \text{für } k = \ell \\ 0 & \text{sonst} \end{cases}$$

$$(4) \quad B_k(x) = \prod_{\substack{\ell=0 \\ \ell \neq k}}^{n} (x - x_\ell) \Big/ \prod_{\substack{\ell=0 \\ \ell \neq k}}^{n} (x_k - x_\ell)$$

Grad n

Dann erfüllt $p(x) = \sum_{k=0}^{n} f_k B_k(x)$ (5)

dann erfüllt die Interpol. beding. (1)

Bem: Grad $\leq n$

Existenz gesichert.

Eindeutigkeit ? Gl. system regulär.

Satz: Für $n+1$ verschiedene Stützstellen x_k, $k = 0, 1, \dots n$ und beliebige Stützwerte f_k, $k = 0, 1, \dots, n$ gibt es genau ein Interpol. polynom. vom Grad $\leq n$ mit

$$p(x_k) = f_k \; .$$

6.3. Die barycentrische Formel

Effiziente Auswertung von p an gegebener "neuer" Stelle x.

$$L(x) := \prod_{i=0}^{n} (x - x_i) \quad ; \text{ Produkt aller linear Faktoren}$$

$$w_k := \frac{1}{\prod\limits_{\substack{i=0 \\ i \neq k}}^{n} (x_k - x_i)} \quad ; \; k = 0, 1, \dots, n$$

„Stützkoeff"

$$(4) \Rightarrow B_k(x) = L(x) \cdot \underbrace{\frac{w_k}{(x - x_k)}}_{:= c_k(x)} = L(x) \cdot c_k(x)$$

$$(5) \Rightarrow p(x) = L(x) \sum_{k=0}^{n} f_k \cdot c_k(x) \qquad (6)$$

(6) gilt für beliebige Wahl von $\{f_k\}$

z.B. $f_k = 1$, $k = 0, 1, \dots, n \Rightarrow p(x) = 1$

in (6) $\Rightarrow$

$$1 = L(x) \sum_{k=0}^{n} c_k(x) \rightarrow L(x) = \frac{1}{\sum\limits_{k=0}^{n} c_k(x)}$$

Algorithmus zum Auswerten von $p(x)$

(i) Berechne Stützkoeff. $w_k = \dfrac{1}{\prod_{\substack{i=0 \\ i \neq k}}^{n} (x_k - x_i)}$; $k = 0, 1, \ldots, n$

Aufwand : $O(n^2)$ (unabhängig von f_k und x)

(ii) Für Neustellen x

$$c_k := \frac{w_k}{x - x_k} \quad ; \quad k = 0, 1, \ldots, n$$

$$p(x) = \frac{\sum_{k=0}^{n} c_k f_k}{\sum_{k=0}^{n} c_k}$$

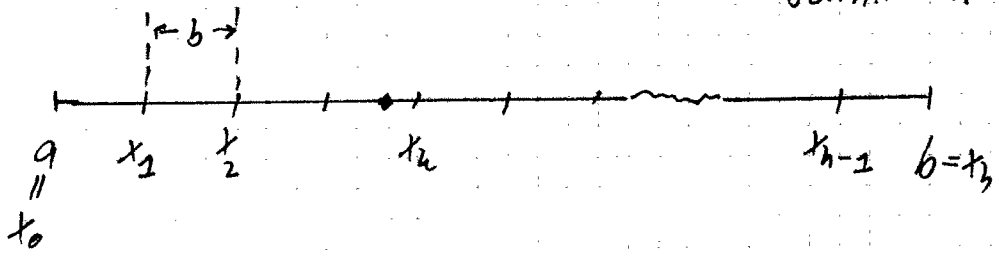
Aufwand : $O(n)$ ($2n + 3$ Punktoper.)

Die Stützkoeffizienten

Zwei wichtige Fälle

a) Äquidistante Stützstellen

Schritt $h > 0$



$$w_k = \frac{1}{h^n k(k-1)(k-2) \cdot 2 \cdot 1 \cdot (-1)^{n-k} \cdot 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (n-k)}$$

$$w_k^* = (-h)^n n!\, w_k = (-1)^k \binom{n}{k} \quad ; \quad k = 0, 1, \ldots, n$$

Binomialkoeff. mit altern. Vorzeichen.

Bsp: $n = 3$

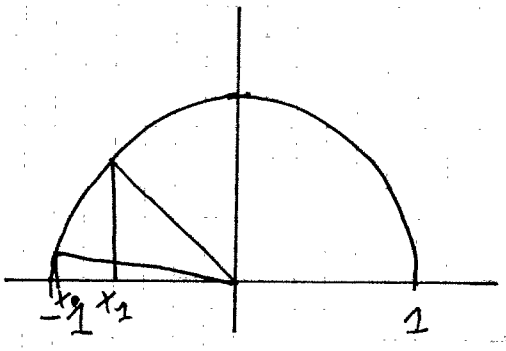
$w_k^* = \{1, -3, 3, -1\}$

b) Tschebyscher – Punkte

$$x_k := \frac{a+b}{2} + \frac{a-b}{2} \cos\left(\frac{(k+\frac{1}{2})\pi}{n+1}\right)$$

O. b. d. A. $a = 1, b = -1$

$$x_k = \cos\underbrace{\left(\frac{(k+\frac{1}{2})\pi}{n+1}\right)}_{\varphi_k}$$



$$\frac{1}{w_k} = \prod_{\substack{i=0 \\ i \neq k}}^{n} (x_k - x_i) = \lim_{x \to x_k} \prod_{\substack{i=0 \\ i \neq k}}^{n} (x - x_i) \cdot \frac{x - x_k}{x - x_k}$$

$$= \lim_{x \to x_k} \frac{L(x) - L(x_k)}{x - x_k}$$

$$w_k = \frac{1}{L'(x_k)} \quad (7)$$

Berechne: $L(x)$ für T-Punkte

Setze: $x = \cos(\varphi)$

Beh: $L(x) = c \cdot \cos((n+1) \cdot \varphi)$ (8)

↳ const.

denn: Muss gelten: $L(x_k) = 0$

$$L(x_k) = c \cdot \cos((n+1)\varphi_k) = c \cdot \cos((k+\tfrac{1}{2})\pi) = 0$$

Differenziere (8) nach φ

$$(8) \Rightarrow -L'(x)\sin(\varphi) = (-1)\, C \sin((n+1)\varphi)\,(n+1)$$

$$w_k = \frac{1}{L'(x_k)} = (-1)^k (-1) \sin\left(\frac{(k+\tfrac{1}{2})\pi}{n+1}\right)$$

6.4. Problematik bei der Polynominterpolation

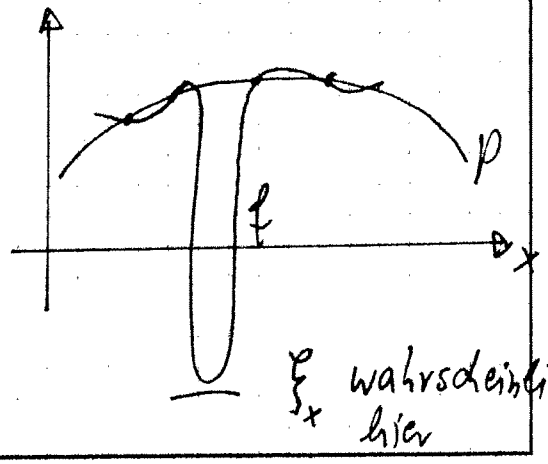
Satz: Sei f auf dem Intervall $J := [a,b]$, $n+1$ stetig diffbar. „a" „b"
und $p(x)$ das zu $x_0 < x_1 < x_2 < \ldots < x_n$
gehörige Interpolationspolynom vom Grad $\leq n$
mit $p(x_k) = f(x_k) := f_k$
Dann existiert zu einer beliebigen Nennstelle
$x \in J$ ein $\xi_x \in J = [a,b]$ mit

$$f(x) - p(x) = \frac{1}{(n+1)!} L(x) f^{(n+1)}(\xi_x)$$

Folgerung: Grosse Interpolationsfehler können 2 Ursachen haben.

(i) $\left| f^{(n+1)}(\xi_k) \right|$ gross

Es liegt an f.



Abhilfe : Lege Stützstellen dichter.

(t') $|L(x)|$ gross

Viele äquidistante Stützstellen sind ungünstig

Besser verdichte Stützstellen am Intervallrand z.B. : T-Punkte

Numerische Methoden – Zusammenfassung

(1) Fixpunktiteration

Gleichung, Form, $x_{k+1} = g(x_k)$ für $k = 0, 1, 2, \ldots$

Konvergenz, abstossend, anziehend $|g'(\xi)| < 1$ (linear)

Banachscher Fixpunktsatz,

Nullstellensuche: (skalar + N Dimensionen)

$f(x) = 0$, Newton-, Sekantenverfahren, Quasinewton

$$x_{k+1} = x_k - \frac{f(x_k)}{\underbrace{f'(x_k)}_{\neq 0}}$$ „linearisierung"

Konvergenz: Ordnung, Abbruchkriterium (absoluten, relativen Fehler)

Effizienz, Jacobi-Matrix

(2) Lineare Gleichungssysteme

regulär, singulär

Gauss, faktorisierung, LU-Zerlegung (LR)

Permutationsmatrix

Normen (Höldersche, p-Norm); Matrixnorm: $\|A\| := \max_{x \neq 0} \frac{\|Ax\|}{\|x\|}$

Kondition, Einfluss der Fehler

QR-Zerlegung (Ausgleichsrechnung), Givensrotation. (Least squares)

(3) Eigenwertprobleme

Definitionen und Numerik (schwingende Membran) $\begin{cases} \Delta u = \lambda u \\ u = 0 \end{cases}$

Vektoriteration, Rayleigh-Quotient

(Inverse) und QR für alle EW und EV

(4) Singulärwertzerlegung

für Rang einer Matrix, Lineare Abb., Ausgleichsproblem

(5) Iterative Methoden für LGS

Jacobi-Iteration, CG: Minimierung einer quadratischen Funktion

$A = A^T$ positiv definit

A-orthogonal (konjugiert), A-Norm $\|x\|_A = \sqrt{x^T A x}$

Algorithmus

Vorkonditionierung

(6) Numerik der ODEs

DGl 1. und 2. Ordnung, AWP, Autonome DGl, Systeme

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases}$$

Euler, (implizit), Mittelpunktsregel, ...

RK-Verfahren: $P(\hat{s}) \leq \hat{s} - 1$

$$\begin{cases} k_i = f\left(t_n + c_i \Delta t, y_n + \Delta t \sum_{j=1}^{P(\hat{s})} a_{ij} k_j\right) \\ y_{n+1} = y_n + \Delta t \sum_{i=1}^{s} b_i k_i \end{cases}$$

globaler, lokaler Fehler, Konsistenz

Konsistenz + Stabilität $\Longrightarrow$ Konvergenz

$\| l(t, \Delta t) \| = O(\Delta t^{p+1}) \Longleftrightarrow$ p Konsistenz (Konvergenz) Ordnung

(Taylor)

Stabilität: $y' = \lambda y$, $\lambda \in \mathbb{C}$, $y(t) \sim e^{\lambda t}$

$y_{n+1} = R(\lambda \Delta t)\, y_n$

Steife Diffgl. $A(\alpha)$, A, Stabilität

(7) Interpolation

Bedingung: $p(x_i) = f_i$

Lagrangepolynom; baryzentrische Formel

Stützkoeffizienten, Tschebyscheff-Punkte