Inhaltsverzeichnis

1 Norm und Skalarprodukt

1.1 Norm

Definitheit: $||x|| = 0 \Rightarrow x = 0$
absolute Homogenität: $||\alpha x|| = |\alpha| * ||x||$
Dreiecksungleichung: $||x + y|| \leq ||x|| + ||y||$

1.2 Skalarprodukt

¡x + y, z¿ = ¡x, z¿ + ¡y, z¿
¡x, y + z¿ = ¡x, y¿ + ¡x, z¿
TODO: Klammer
$< \lambda x, y > = \lambda$ ¡x, y¿
$< x, \lambda y > = \lambda$ ¡x, y¿
¡x, y¿ = ¡y, x¿
¡x, x¿ $\geq$ 0
¡x, x¿ = 0 $\Rightarrow$ x = 0

1.2.1 Vom Skalarprodukt induzierte Norm

$||x|| = \sqrt{< x, x >}$

1.2.2 Cauchy-Schwarzche Ungleichung

$| < x, y > | \leq ||x|| * ||y||$

2 Symmetrische, positiv definite Matrix

TODO: Matrizen

insbesonders: Diagonalmatrizen, Einheitsmatrizen

positiv definit: $x^t Ax > 0$ (beliebige Matrix)

alle EW ¿ 0 (symmetrische Matrix)

alle Haupt[TODO: ?] ¿ 0 (symetrische Matrix)

TODO: Matrix $\Rightarrow$ 3 Hauptminoren[?] = det(a), det(TODO: Matrix), det(TODO: Matrix)

2.1 Cholesky-Zerlegung

$A = GG^t$ G unter der Matrix, invertierbar (symmetrische Matrix)

2.2 [?] diagonaldominant und alle Diagonalelemente größer gleich 0

(symmetrische Matrix)

2.3 Eigenwerte

det(λ En - A) = 0

2.4 Eigenvektor

f(v) = λ v

3 Matrixnormen

3.1 Natürliche Matrixnorm

$||A||_\infty := max_{x \neq 0} \frac{||Ax||_\infty}{||x||_\infty} = max_{||x||=1} ||Ax||_\infty$

$||A|| = 0 \Rightarrow A = 0, ||\lambda A|| = |\lambda| * ||A||, ||A + B|| \leq ||A|| + ||B||, ||A * B|| \leq ||A|| * ||B||$

3.2 Verträglichkeit

$||Ax|| \leq ||A|| * ||x||$

3.3 Zeilensummennorm

= natürliche Matrixnorm

$||A||_\infty = max_{||x||_\infty=1} ||Ax||_\infty = max_{i=1,...,m} \sum_{j=1}^{n} |a_{ij}|$

$A = TODO : Matrix||A||_\infty$
$= max|1| + |-2| + |-3|, |2| + |3| + |-1|$
$= max6, 6 = 6$

3.4 Spaltensummennorm

$||A||_1 := max_{x \neq 0} \frac{||Ax||_1}{||x||_1} = max_{||x||_1=1} ||Ax||_1 = max_{j=1,...,n} \sum_{i=1}^{m} |a_{ij}|$

A = TODO: Matrix $||A||_1$ = max—1— + —2—, —-2— + —3—, —-3— + —-1— = max3, 5, 4 = 5

$||A^t||_1 = ||A||_\infty$

3.5 Spektralnorm

Die Spektralnorm wird definiert als

$A||_2 := max_{||x||_2=1} ||Ax||_2 // = max_{x \neq 0} \frac{||Ax||_2}{||x||_2} // = max_{||x||_2=1} < Ax, Ax > // = max_{||x||_2=1} < A^t Ax, x > // = max \sqrt{|\lambda|}, \lambda * EW von A^t A$

$A = TODO : Matrix, A^t A = TODO : Matrix det(\mu E_n - A^t A) = 0 \Leftrightarrow \mu_{1,2} = 16, 1$

$||A||_2 = \sqrt{max(\mu_1, \mu_2)} = \sqrt{\mu_1} = \sqrt{16} = 4$

4 Spektralradius, Konditionszahl einer Matrix

4.1 Spektralradius [phi]

$\varphi(A) = max : 1 \leq i \leq n|\lambda_i(A)| = spr(A)$ der betragsmäßig größte Eigenwert von A

$||A|| \geq |\lambda|$ (für jede Matrixnorm, die mit einer Vektornorm verträglich ist)

4.2 Konditionszahl einer Matrix A

$cond(A) = ||A|| * ||A^{-1}||$

4.3 Sonderfall symmetrisch, positiv definite Matrix

$cond(A) = \frac{\lambda_{max}}{\lambda_{min}}$

5 Ähnlichkeitstransformation, Invarianz der Eigenwerte

y = Ax

$\overline{x} = Cx, \overline{y} = Cy \qquad (detC \neq 0), C \in GL$

$y = Ax \Rightarrow C^{-1}\overline{y} = AC^{-1}\overline{x} \Rightarrow \overline{y} = CAC^{-1}\overline{x} \Rightarrow \overline{y}\overline{A}\overline{x}$

$\overline{A} = CAC^{-1} \Rightarrow \overline{A} \sim A$

$\lambda EW, vEV zu A$

$\Rightarrow Av = C^{-1}\overline{A}Cv = \lambda v$

$\Rightarrow \overline{A} und A$ haben dieselben Eigenwerte, algebraisch und geometrische Vielfalten stimmen überein (Invarianz der Eigenwerte)

5.1 Reduktionsmethoden

A duch Ähnlichkeitstransformationen

$A = A^{(0)} = T_1^{-1}A^{-1}T_1 = Q... = T_i^{-1}A^{(i)}T_i = ...$

auf Form bringen, für welche EW und EV leicht zu berechnen sind (z.B. Jordan-Normalform)

6 Gleitkommazahlen, ...

6.1 Gleitkommazahl (normalisiert)

$b \in \mathbb{N}, b \geq 2, x \in \mathbb{R}$
$x = \pm m * b^{\pm e}$
Mantisse: $m = m_1 b^{-1} + m_2 b^{-2} + ... \in \mathbb{R}$
Exponent: $e = e_{s-1} b^{s-1} + ... + e_0 b^0 \in \mathbb{N}$
für $x \neq 0$ eindeutig

6.2 Gleitkommagitter

$A = A(b, r, s)$ größte Darstellbare Zahl: $(1 - b^{-r}) * b^{b^s - 1}$
mit b als Basis, r als Mantissenlänge, s als Exponentenlänge
$(b = 10) : 0,314 * 10^1 = 3,14$
$0,123 * 10^6 = 123.000$
Beispiel: konvertiere von Basis 8 zu Basis 10:
$x = (0,5731 * 10^5)_8 \in A(8, 5, 1)$
$x = (5 * 8^{-1} + 7 * 8^{-2} + 3 * 8^{-3} + 1 * 8^{-4}) * 8^5$
$x = 5 * 8^4 + 7 * 8^3 + 3 * 8^2 + 1 * 8^1 = 24.264 * 10^0$

6.3 Maschienengenauigkeit eps

$eps = \frac{1}{2} b^{-r+1}, IEEE : eps = \frac{1}{2} * 2^{-52} \approx 10^{-16}$

6.4 Rundungsfehler

$absolut : |x - rd(x)| \leq \frac{1}{2} b^{-r} b^e$
$relativ : |\frac{x - rd(x)}{x}| \leq \frac{1}{2} b^{-r+1} = eps$

7 Darstellung des Interpolationsfehlers

7.1 Fehler I

f$\in C^{n+1}$[a, b], $\forall$ x $\in$ [a, b] $\exists \xi_x \in (\overline{x_0, ..., x_n, x})$, wobei das Intervall das kleinst mögliche Intervall, das alle x_i enthällt, s.d.

$f(x) - p(x) = \frac{f^{(n+1)}(\xi x)}{(n+1)!} \prod_{j=0}^{n}(x - x_j)$

7.2 Fehler II

$f \in C^{n+1}[a, b], \forall x \in [a, b] \setminus x_0, ..., x_n gilt:$

$f(x) - p(x) = f[x_0, ..., x_n, x] \prod_{j=0}^{n}(x - x_j)$

mit $f[x_i, ..., x_{i+k}] = y[x_i, ..., x_{i+k}]$

und $f[x_0, ..., x_n, x] = \int_0^1 \int_0^{t_1} ... \int_0^{t_n} f^{n+1}(x_0 + t_1(x_1 - x_0) + ... + t_n(x_n - x_{n-1} + t(x - x_n))dtdt_n...dt_1$

für $x_0 = x_1 = ... = x_n$:

$f[x_0, ..., x_n] = \frac{1}{n!} f^{(n)}(x_0)$

$\frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{j=0}^{n}(x - x_j) = f(x) - p(x) = f[x_0, ..., x_n, x] \prod_{j=0}^{n}(x - x_j)$

$\Rightarrow f[x_0, ..., x_n, x] = \frac{f^{(n+1)}(\xi_x)}{(n+1)!}$

8 Konditionierung einer numerischen Aufgabe, Konditionszahlen

8.1 numerische Aufgabe

$x_j \in \mathbb{R} mit f(x_1, ..., x_m) \Rightarrow y_i = f_i(x_j)$
fehlerhafte Eingangsgrößen $x_i + \Delta y_i$
$|\Delta y_i|$ ist der absolute Fehler, $|\frac{\Delta y_i}{y_i}|$ ist der relative Fehler

8.2 Konditionszahl (relativ)

$k_{ij}(x) = \frac{\partial f_i}{\partial x_i}(x) \frac{\Delta x_j}{x_j}$
$\frac{\Delta y_i}{y_i} = \sum_{j=1}^{m} k_{ij}(x) \frac{\Delta x_j}{x_j}$
$|k_{ij}(x)| >> 1 \Rightarrow$ schlecht konditioniert
$|k_{ij}(x)| << 1 \Rightarrow$ gut konditioniert, ohne Fehlerverstärkung
$|k_{ij}(x)| > 1 \Rightarrow$ Fehlerverstärkung
$|k_{ij}(x)| < 1 \Rightarrow$ Fehlerdämpfung

9 Stabilität eines Algorithmus

9.1 stabiler Algorithmus

akkumulierte Fehler der Rechnung (Rundungsfehler, Auswertungsfehler, etc.) übersteigen den unvermeidbaren Problemfehler der Konditionierung der Aufgabe nicht. Aka Trotz Ungenauigkeiten bei den Eingabe Variablen erhalten wir fast sehr genaue Ergebnisse.

10 Auslöschung

Verlust von Genauigkeit bei der Subtraktion von Zahlen mit gleichem Vorzeichen

TODO: bei bedarf ein Beispiel

11 Horner-Schema*

$p(x) = a_0 + x(... + x(a_{n-1} + a_n x)...)$

11.1 Code

```
def horner(Ac, Ax, n, x):
    y = 0.0
    for i in reversed range(n):
        y = y * (x - Ax[i]) + Ac[i]
    return y
```

Ac: Vektor mit Koeffizienten, ist ein np Array
Ax: Stützstellen, ist ein np Array
n: Anzahl der Stützstellen, ist ein int
x: Auswertungspunkt, ist ein double
Immer Horner-Schema zur Auswertung von Polynomen verwenden.

11.2 Auswertung

TODO: subsection

12 Interpolation und Approximation

12.1 Grundproblem

Darstellung und Auswertung von Funktionen

12.2 Aufgabenstellung

f(x) nur auf Diskreter Menge von Argumenten $x_0, ..., x_n$ bekannt und soll rekonstruiert werden

analytisch gegebene Funktion soll auf Reelwerte dargestellt werden, damit jederzeit Werte zu beliebigen x berechnet werden können.

Einfach konstruierte Funktionen in Klassen P:

Polynome: $p(x) = a_0 + a_1 x + a_2 x^2 + ... + a_n x^n$

rationale Funktion: $r(x) = \frac{a_0 + a_1 x + ... + a_n x^n}{b_0 + b_1 x + ... + b_m x^m}$

trigonometrische Funktion: $t(x) = \frac{1}{2} a_0 + \sum_{k=1}^{n} (a_k cos(kx) + b_k sin(kx))$

Exponentialsummen: $e(x) = \sum_{k=1}^{n} a_k exp(b_k x)$

12.3 Interpolation

Zuordnung von $g \in P$ zu f durch Fixieren von Funktionswerten

$g(x_i) = y_i = f(x_i), i = 0, ..., n$

12.4 Approximation

$g \in P$ beste Darstellung, z.B.

$max_{a \leq x \leq b} |f(x) - g(x)| minimal$

$(\int_a^b |f(x) - g(x)|^2 dx)^{\frac{1}{2}} minimal$

13 Lagransche Interpolationsaufgabe

13.1 Aufgabe

Finde zu n + 1 verschiedene Stützstellen/Knoten $x_0, ..., x_n \in \mathbb{R}$ und Werten $y_0, ..., y_n \in \mathbb{R}$ ein Polynom p $\in P_n mit p(x_i) = y_i$

13.2 Eindeutigkeit + Existenz

Die Lagransche Interpolationsaufgabe ist eindeutig lösbar

TODO: bei bedarf Beweis rein kopieren den Ich nicht verstanden hab

13.3 Lagransche Basispolynome

$L_i^{(n)}(x) = \prod_{j=0,j\neq i}^{n} \frac{x-x_j}{x_i-x_j} \in P_n, i = 0, ..., n$

13.4 Eigenschaften

ortogonal: es gilt $L_i^{(n)}(x_k) = d_{ik} =$ TODO: split over 2 lines 1, i = k 0, sonst

bilden Basis von P_n

haben Grad n

13.5 Lagransche Darstellung

$p(x) = \sum_{i=0}^{n} y_i L_i^{(n)}(x) \in P_n$ mit $p(x_j) = y_j$

Nachteil: Bei Hinzunahme von (x_{n+1}, y_{n+1}) ändert sich das Basispolynom komplett

TODO: Beispiel

14 Newtonsche Basispolynome...

14.1 Newton-Polynome

$N_0(x) = 1, N_i(x) = \prod_{j=0}^{i-1}(x - x_j)$ mit $p(x) = \sum_{i=0}^{n} a_i N_i(x)$

14.1.1 Auswertung

$y_0 = p(x_0) = a_0$

$y_1 = p(x_1) = a_0 + a_1 * (x_1 - x_0)$

$\vdots$

$y_n = p(x_n) = a_0 + a_1(x_1 - x_0) + ... + a_n(x_n - x_0) * ... * (x_n - x_{n-1})$

14.1.2 Vorteil

Bei Hinzunahme von (x_{n+1}, y_{n+1}) muss nur eine neue Rechnung durchgeführt werden, und nicht das gesamte Polynom neu berechnet werden

TODO: Beispiel

14.2 Newtonsche Darstellung(stabile Variante)

$p(x) = \sum_{i=0}^{n} y[x_0, ..., x_i] N_i(x)$

14.3 Dividierte Differenzen*

$y[x_i, ..., x_{k+1}] = \frac{y[x_{i+1}, ..., x_{k+1}] - y[x_i, ..., x_{i+k-1}]}{x_{i+k} - x_i}$ mit k = 1, ..., j und i = k - j

für beliebige [?] $\sigma : 0, ..., n \to 0, ..., n$ gilt $y[\tilde{x_0}, ..., \tilde{x_n}] = y[x_0, ..., x_n]$

15 Nevillsche Darstellung

$p_{jj}(x) = y_j \qquad j = 0, ..., n \qquad k = 1, ..., j \qquad i = k - j$

$p_{i,i+k}(x) = p_{i,i+k-1}(x) + (x - x_i)\frac{p_{i+1,i+k}(x) - p_{i,i+k-1}(x)}{x_{i+k} - x_i}$

15.1 Schema

$$\begin{array}{rrcrcrr}
x & k=0 & & k=2 & \dots & k=n-1 & k=n \\
x_0 & y_0 & \longrightarrow & p_{0,1} & \dots & p_{0,n-1} & p_{0,n} \\
x_1 & y_1 & \longrightarrow & p_{1,2} & \dots & p_{1,n} & \\
\vdots & \vdots & \vdots & \iddots & & & \\
x_{n-1} & y_{n-1} & \longrightarrow & p_{n-1,n} & & & \\
x_n & y_n & & & & &
\end{array}$$

TODO: add the diagonal arrows

Hinzunahme von (x_{n+1}, y_{n+1}) ist problemlos

Auswertung von $p_{0,n}(x)$ in $\xi \neq x_i$ ohne vorherige Bestimmung der Koeffizienden der Newton-Darstellung ist einfach und Numerisch stabil möglich

15.2 Code

```
def divDiffs(xi, yi, x):
    n = len(xi)
    p = n * [0]
    for k in range(n):
        for i in range(n - k):
            if k == 0:
                p[i] = yi[i]
            else:
                p[i] = ((x - xi[i + k]) * p[i] + (xi[i] - x) * p[i + 1]) / (xi[i] - xi[i + k])
    return p[0]
```

16 Hermite-Interpolation

16.1 Aufgabe

Gegeben: x_i i = 0, ..., m paarweise verschieden
$y_i^{(k)}$ i = 0, ..., m k = 0, ..., $\mu_i(\mu_i \geq 0)$

Gesucht: $p \in P_n$, n = m + $\sum_{i=0}^{m} \mu_i$: $p^{(k)}(x_i) = y_i^{(k)}$

x_i sind (μ_i + 1)-fache Stützstellen

$x_0 = -1$, $x_1 = 1$, $m = 1$, $y_0^{(0)} = 0$, $y_1^{(0)}$, $y_1^{([l?])} = 2$

$\Rightarrow \mu_0 = 0$, $\mu_1 = 1$

$\Rightarrow$ n = 1 + 0 + 1 = 2

$\Rightarrow$ p(x) = x^2

16.2 Existenz + Eindeutig

analog zur Lagrange-Interpolation

16.3 Fehler

$f \in C^{n+1}[a,b] : \forall x \in [a,b] \exists \xi_x \in (\overline{x_0, ..., x_m, x})$, s.d.

$f(x) - p(x) = f[x_0, ..., x_0, ..., x_m, ..., x_m, x] \prod_{i=0}^{m}(x - x_i)^{\mu_i + 1}$

$= \frac{1}{(n+1)!} f^{(n+1)}(\xi_x) \prod_{i=0}^{m}(x - x_i)^{\mu_i + 1}$

17 Extrapolation zum Limes + Fehler

17.1 Richardson-Extrapolation

nicht direkt berechenbare Größe

$a(0) = lim_{k \to 0} a(k), \qquad k \in \mathbb{R}_+$

berechne a(k_i) für gewisse k_i, i = 0, ..., n und [?] $p_n(0)$ des Interpolations Polynoms zu $(h_i, a(h_i))$ als Schätzung für a(0)

a(0) := $lim_{x \to 0^+} \frac{cos(x)-1}{sin(x)} \qquad (= 0)$

a(x) := $\frac{(cos(x)-1)}{sin(x)}$

Interpolation a(x) an Stützstellen k_i nahe bei 0:

$k_0 = \frac{1}{8} \qquad a(k_0) = -6,258151 * 10^{-2}$

$k_1 = \frac{1}{16} \qquad a(k_1) = -3,126018 * 10^{-2}$

$k_2 = \frac{1}{32} \qquad a(k_2) = -1,562627 * 10^{-2}$

17.2 Lagrange

$p_2(x) = a(k_0) \frac{(x-\frac{1}{16})(x-\frac{1}{32})}{(\frac{1}{8}-\frac{1}{16})(\frac{1}{8}-\frac{1}{32})} + a(k_1) \frac{(x-\frac{1}{8})(x-\frac{1}{32})}{(\frac{1}{16}-\frac{1}{8})(\frac{1}{16}-\frac{1}{32})} + a(k_2) \frac{(x-\frac{1}{8})(x-\frac{1}{16})}{(\frac{1}{32}-\frac{1}{8})(\frac{1}{32}-\frac{1}{16})}$

$\Rightarrow a(0) \sim p_2(0) = -1,02 * 10^{-5}$

17.3 Neville

$p_{i,i+k}(0) = p_{i,i+k-1}(0) + \frac{p_{i,i+k-1}(0) - p_{i+1,i+k}(0)}{\frac{x_{i+k}}{x_i - 1}}$, k = 1, 2

i	x_i	$p_{i,i}(0) = a(k_i)$	$p_{i,i+1}(0)$	$p_{i,i+2}(0)$
0	$x_0 = \frac{1}{8}$	$-6,258151 * 10^{-2}$	$6,115 * 10^{-5}$	$-1,02 * 10^{-5}$
1	$x_1 = \frac{1}{16}$	$-3,126018 * 10^{-2}$	$7,64 * 10^{-6}$	
2	$x_2 = \frac{1}{32}$	$-1,562627 * 10^{-2}$		

17.4 Extrapolationsfehler

a(n) habe die Entwickling:

$a(h) = a_0 + \sum_{j=1}^{n} a_j h^{jq} + a_{n+1}(h) h^{(n+1)q}$ mit q ¿ 0, Koeffizienten a_j

und $a_{n+1}(h) = a_{n+1} + a(1[?????])$

$(h_k)_{k \in \mathbb{N}}$ erfülle:

$0 \leq \frac{h_{k+1}}{h_k} \leq p < 1$ ($\Rightarrow h_k$ positiv monoton fallend)

Dann gilt für $p_1^{(k)} \in P_n$ (in h^q) durch $(h_k^q, a(h_k))$, ..., $(h_{k+n}^q, a(h_{k+1}))$

$a(0) - p_n^{(k)}(0) = O(h_k^{(n+1)q}) \qquad (k \to \infty)$

18 Spline-Interpolation

18.1 Interpolationsnachteil

Starke Oszillation von Polynomen höheren Grades

18.2 Abhilfe

Spline-Interpolation, d.h. stückweise polynomielle Interpolation mit (n - 1)-mal stetig diff.baren Knoten

18.3 Lineare Spline

alle Abschnitt-Splines sind lineare Funktionen

18.4 Kubischer Spline

$s_n : [a, b] \to \mathbb{R}$ kubischer Spline bezüglich $a = x_0 < x_1 < ... < x_n = b$, wenn gillt

1. $s_n \in C^2[a, b]$
2. $S_n|_{I_i} \in P_3$, i = 1, ..., n

natürlicher Spline:

3. $s_n''(a) = s_n''(b) = 0$

18.5 Existenz

Der interpolierende kubische Spline existiert und ist eindeutig bestimmt durch zusammen Vorgabe von $s_n''(a), s_n''(b)$

für natürlichen Spline s_n durch $x_0, ..., x_n$, $y_0, ..., y_n$ gilt:

$\int_a^b |s'(x)|^2 dx \leq \int_a^b |g''(x)|^2 dx$

bezüglich g $\in C^2[a, b]$ mit g$(x_i) = y_i$, i = 1, ..., n

18.6 Approximationsfehler

$f \in C^4[a, b]$, $s_1''(a) = f''(a) \wedge s_n''(b) - f''(b)$:

$max_{x \in [a,b]} |f(x) - s : n(x)| \leq \frac{1}{2} h^4 max_{x \in [a,b]} |f^{(4)}(x)$

19 Gauß-Approximation

$< f, g >:= \int_a^b f(t)\overline{g(t)}dt \qquad ||f|| = \sqrt{< f, f >}$

H Prähilbertraum, $\delta \subset H$ endlich Dimensional

$\exists f \in H$ eindeutig bestimmte "beste Approximation" $g \in S$

$||f - g|| = min_{\varphi \in S}||f - \varphi||$

bes. einfache Lösung, wenn $\{\varphi_1, ..., \varphi_n\}$ eine ONB ist, d.h.

$(\varphi_i, \varphi_j) = \delta_{i,j} \Rightarrow \alpha_i =< f, \varphi_i >$ i = 1, ..., n

$\Rightarrow g = \sum_{i=1}^{n} < f, \varphi_i > \varphi_i$ ist beste Approximation

20 Gram-Schmidt-Algorithmus

$w_1 := \frac{v_1}{||v_1||}$ $\tilde{w_k} := v_k - \sum_{i=1}^{k-1} \gamma < v_k, w_i > w_i$, $w_k := \frac{\tilde{w_k}}{||\tilde{w_k}||}$

20.1 Code

```
n = size(v, 1)
    k = size(v, 2)
    u = np.zeros(n, k)
    u[:, 1] = v[:, 1]/sqrt(v[:, 1] * v[:, 1])
    for i in range(2, k):
          u[:, i] = v[:, i]
          for j in range(1, i - 1):
                u[:, i] = u[:, i] - (u[:, i] * u[:, j]) / (u[:, j] * u[:, j]) * u[:, j]
          u[:, i] = u[:, i] / sqrt(u[:, i] * u[:, i])
```

21 Interpolatorische Quadraturformeln