4.5 Spezielle Verfahren für Polynome

Ein Polynom vom Grad n hat n komplexe Nullstellen und häufig ist es nötig alle zu berechnen.

Strategie:

Berechne eine einzelne Nullstelle von $P_n(x)$, z.B. mit dem Newtonverfahren. Die gefundene Nullstelle kann herausdividiert werden und man erhält ein Polynom vom Grad $n-1$, welches alle übrigen Nullstellen enthält („Deflation")

Deflation

Division eines Polynomes durch einen Linearfaktor:

$$P_n(x) = a_0 + a_1 x + \ldots + a_n x^n$$
$$P_n(x_0) = 0$$
$$\Rightarrow (b_n x^{n-1} + \ldots + b_2 x + b_1)(x - x_0) = a_n x^n + \ldots + a_1 x + a_0$$

Koeffizientenvergleich:

$$b_n = a_n$$
$$b_{n-1} = x_0 b_n + a_{n-1}$$
$$\vdots$$
$$b_0 = x_0 b_1 + a_0 \quad [= 0, \text{ falls } x_0 \text{ Nullstelle ist}]$$

Hornerschema zur gleichzeitigen Auswertung der Ableitung:

$$P_n'(x) = a_1 + a_2 2x + \ldots + a_n n x^{n-1}$$

Das Hornerschema erzeugt:

$$b_n = a_n$$
$$b_{n-1} = a_n x + a_{n-1}$$
$$\vdots$$
$$b_0 = P_n(x)$$

Algorithmus

Input: x, a_i

$b = c = 0$

for $i = n, \dots, 1$

$\quad b = x b + a_i$

$\quad c = x c + b$

$b = x b + a_0$

Output: $P_n(x) = b$

$\quad P_n'(x) = c$

Laguerre's Methode:

Zur Bestimmung der Nullstellen schreibe:

$$P_n(x) = A(x - x_1)(x - x_2)\dots(x - x_n) = A \prod_{i=1}^{n} (x - x_i)$$

dann ist:

$$\ln(|P_n(x)|) = \ln(|A|) + \ln(|x - x_1|) + \dots + \ln(|x - x_n|)$$

Definiere:

$$G = \frac{d}{dx} \ln(|P_n(x)|) = \frac{1}{x-x_1} + \frac{1}{x-x_2} + \dots + \frac{1}{x-x_n}$$

und

$$H = \frac{d}{dx^2} \ln(|P_n(x)|) = \frac{1}{(x-x_1)^2} + \frac{1}{(x-x_2)^2} + \dots + \frac{1}{(x-x_n)^2}$$

Laguerre's Annahme ist nun, dass eine Nullstelle im Abstand von a zur momentanen Abschätzung liegt und alle anderen im Abstand b, also:

$$a = (x - x_1) \quad \text{und} \quad b = (x - x_i) \qquad i = 2, \dots, n$$

Dann folgt:

$$G = \frac{1}{a} + \frac{n-1}{b} \qquad \text{und} \qquad H = \frac{1}{a^2} + \frac{n-1}{b^2}$$

$$\Rightarrow \quad a = \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}}$$

Das Vorzeichen der Wurzel sollte so gewählt werden, dass der Nenner maximalen Betrag erhält.

Die neue Abschätzung ist dann durch

$$x = x - a$$

gegeben.

Algorithmus: Laguerre

Input: x, a, ε

while $(|P_n(x)| > \varepsilon)$

$$G = \frac{P'(x)}{P(x)}$$

$$H = G^2 - \frac{P''(x)}{P'(x)}$$

$$a = \frac{n}{G \pm \sqrt{(n-1)(nH - G^2)}}$$

$$x = x - a$$

Output: $x : P(x) \approx 0$

Nachtrag Hornerschema für die zweite Ableitung:

$d = xd + c$

Loop darf nur bis 2 laufen

$P''(x) = 2!\,d$

Bemerkungen

- Laguerres Methode konvergiert mit $K = 3$, wenn die Abschätzung nahe genug an einer Nullstelle liegt ($K = 1$ für mehrfache Nullstellen)
- Die Methode konvergiert fast immer gegen eine Nullstelle, im Gegensatz zu Newton.
- Möglicherweise ist noch eine Nachiteration mit Newton nötig.

5. Lineare Gleichungssysteme

Wir betrachten nun den Fall von n linearen Gleichungen:

$$A\vec{x} = \vec{b} \quad \text{mit} \quad A \in \mathbb{R}^n \times \mathbb{R}^n \;,\; \vec{x}, \vec{b} \in \mathbb{R}^n$$

Gesucht ist x. Eine Lösung existiert dann, wenn A regulär ist, also $\det(A) \neq 0$.

Ansätze zur Lösung von $A\vec{x} = \vec{b}$

i) direkte Verfahren: exakte Lösungen in endlich vielen Schritten

ii) iterative Verfahren: Folge von Näherungslösungen

5.1 Direkte Verfahren

Das Prinzip dieses Verfahrens besteht in einer geeigneten Transformation des ursprünglichen Systems:

$$\underbrace{A}_{\mathbb{1}}\vec{x} = \vec{b} \qquad | \cdot Q \text{ von links}$$

$$\Rightarrow \underbrace{QAR}_{\hat{A}}\;\underbrace{R^{-1}\vec{x}}_{\hat{\vec{x}}} = \underbrace{Q\vec{b}}_{\hat{\vec{b}}}$$

mit den Transformationsmatrizen Q, R und so, dass $\hat{A}\hat{\vec{x}} = \hat{\vec{b}}$ einfacher oder gar trivial zu lösen ist. $\hat{A}$ diagonal, so ist das Offensichtlich der Fall. Auch Systeme in Dreiecksform sind leicht lösbar.

Def: L (R) heißt linke (rechte) oder untere (obere) Dreiecksmatrix, falls

$$L_{ij} = 0 \quad \forall_{j>i} \qquad (R_{ij} = 0 \quad \forall_{j<i})$$

L^{-1} (analog R^{-1}) kann leicht berechnet werden!
allgemein $\quad L\vec{x} = \vec{b}$ (Vorwärtssubstitution)

$$\begin{aligned}
l_0 x_0 \qquad\qquad\qquad &= b_0 \\
l_1 x_0 + l_1 x_1 \qquad\quad &= b_1 \\
\vdots \qquad\qquad\qquad & \quad \vdots \\
l_{n-1,0} x_0 + \ldots + l_{n-1,n-1} x_{n-1} &= b_{n-1}
\end{aligned}$$

$$\Rightarrow x_0 = \frac{b_0}{l_{00}}$$

$$x_1 = \frac{1}{l_{11}}(b_1 - l_{10} x_0)$$

$$\vdots$$

$$x_i = \frac{1}{l_{ii}}\left(b_i - \sum_{k=0}^{i-1} l_{ik} x_k\right)$$

Algorithmus: Vorwärtssubstitution

Input: L, $\vec{b}$
for $i = 0, \ldots, n-1$
$\quad x_i = \frac{1}{l_{ii}}\left(b_i - \sum_{k=0}^{i-1} l_{ik} x_k\right)$
return $\vec{x}$

Etwas allgemeiner kann auch die Inverse von L direkt berechnet werden, also X so, dass $LX = \mathbb{1}$, wobei

$X = (\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_{n-1})$

$\mathbb{1} = (\vec{e}_0, \vec{e}_1, \ldots, \vec{e}_{n-1})$

Algorithmus: Inverse einer L Matrix

Input: L

for $j = 0, \dots, n-1$

$\quad x_{jj} = \frac{1}{l_{jj}}$

$\quad$ for $i = j+1, \dots, n-1$

$\quad\quad x_{ij} = -\frac{1}{l_{ii}} \sum_{k=j}^{i-1} l_{ik} x_{kj}$

output: $X = L^{-1}$

Rechenaufwand zur Berechnung von $Lx = b$:

$\rightarrow$ i-te Zeile: je i Additionen und Multiplikationen
+ eine Division

n Zeilen insgesamt: $\sum_{i=0}^{n-1} i = \frac{1}{2} n(n-1)$ Mult. + Add.

$= 2\frac{1}{2}(n-1)n + n = O(n^2)$