Programming C++

Project **QD-Verfahren für Polynome**

Status: 25. Mai 2021, 19:36

Supervisor: Prof.Dr. G. Haase, `gundolf.haase@uni-graz.at`

Implementieren Sie das von Heinz Rutishauser [Rutishauser1954] zur Bestimmung aller Nullstellen eines Polynoms entworfene QD-Verfahren. (6 Pkt.)

- Definieren Sie eine Klasse `Polynomial`, die einen Polynom über $\mathbb{R}$ speichert und die den Ausgabeoperator `<<` implementiert.

- Machen Sie eine Methode `Polynomial shift(const double d)`, die einen neuen Polynom erzeugt und dabei die Substitution $x_{neu} = x_{alt} + d$ durchführt.

- Implementieren Sie nun eine Funktion `vector<double> roots(const Polynomial &p)`, die alle reellen Nullstellen eine Polynoms mithilfe des QD-Verfahren bestimmt [Hermann2001]:

  Gegeben sei ein Polynom

  $$P(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0, \tag{1}$$

  wobei $a_0 \neq 0$, $a_1 \neq 0$, $\ldots$, $a_n \neq 0$. Das QD-Verfahren zur Approximation der Wurzeln von (1) erzeugt zwei unterschiedliche Zahlenfolgen

  $$\left\{e_i^{(k)}\right\}_{i=1}^{\infty},\ k = 1(1)n+1,\ \text{und}\ \left\{q_i^{(k)}\right\}_{i=1}^{\infty},\ k = 1(1)n.$$

  Die Folgenelemente bestimmen sich wie folgt:

  $$\begin{aligned}
  e_i^{(1)} &= 0,\ i = 1,2,\ldots,\ e_i^{(n+1)} = 0,\ i = 1,2,\ldots,\\
  e_1^{(k)} &= \frac{a_{n-k}}{a_{n-k+1}},\ k = 2(1)n,\\
  q_1^{(1)} &= -\frac{a_{n-1}}{a_n},\ q_1^{(k)} = 0,\ k = 2(1)n,\\
  q_{i+1}^{(k)} &= e_i^{(k+1)} + q_i^{(k)} - e_i^{(k)},\ k = 1(1)n,\ i = 1,2,\ldots,\ \text{und}\\
  e_{i+1}^{(k)} &= \frac{q_{i+1}^{(k)} e_i^{(k)}}{q_{i+1}^{(k-1)}},\ k = 2(1)n,\ i = 1,2,\ldots
  \end{aligned}$$

  Man beachte, dass dass dieser Prozess wohldefiniert ist, falls die Nenner nirgendwo Null sind.

  Die Starttabelle sieht folgendermaßen aus:

i	$e_i^{(1)}$	$q_i^{(1)}$	$e_i^{(2)}$	$q_i^{(2)}$	$e_i^{(3)}$	$q_i^{(3)}$	$\ldots$	$e_i^{(n)}$	$q_i^{(n)}$	$e_i^{(n+1)}$
1	0	$-\frac{a_{n-1}}{a_n}$	$\frac{a_{n-2}}{a_{n-1}}$	0	$\frac{a_{n-3}}{a_{n-2}}$	0	$\ldots$	$\frac{a_0}{a_1}$	0	0
2	0									0
$\vdots$	$\vdots$									$\vdots$

Der nächste Schritt besteht in der Konstruktion der $q_2^{(k)}$-Elemente in der zweiten Zeile. Man bestimmt sie, indem das unmittelbar rechts darüber stehende Element $e_1^{(k+1)}$ zum direkt darüber stehenden Element $q_1^{(k)}$ addiert und das unmittelbar links darüber stehende Element $e_1^{(k)}$ davon substrahiert wird.

Jetzt können die Elemente $e_2^{(k)}$ eingetragen werden. Man bestimmt sie, indem das unmittelbar rechts daneben stehende Element $q_2^{(k)}$ mit dem direkt darüber stehenden Element $e_1^{(k)}$ multipliziert und das entstehende Resultat durch das unmittelbar links daneben stehende Element $q_2^{(k-1)}$ dividiert wird.

Der Prozess wird nun auf Basis dieser zwei Teilschritte analog weitergeführt.

– Gilt $\lim_{i\to\infty} e_i^{(k)} = \lim_{i\to\infty} e_i^{(k+1)} = 0$ für jedes $k = 1(1)n$, dann existiert $\lim_{i\to\infty} q_i^{(k)}$. Dieser Grenzwert stellt eine reelle Wurzel des Polynoms $P(x)$ dar.

– Konvergiert die Folge $\left\{e_i^{(k)}\right\}_{i=1}^{\infty}$ jedoch für mindestens ein $k \in \{1, 2, \ldots, n\}$ nicht gegen Null, dann konvergieren die Folgen $\{r_i\}_{i=1}^{\infty}$ und $\{s_i\}_{i=1}^{\infty}$ gegen die Grenzwerte $r^{(k)}$ bzw. $s^{(k)}$, wobei

$$\begin{aligned} r_i^{(k)} &= q_i^{(k-1)} + q_i^{(k)}, \; i = 1, 2, \ldots \text{ und} \\ s_i^{(k)} &= q_{i-1}^{(k-1)} q_i^{(k)}, \; i = 2, 3, \ldots. \end{aligned}$$

Der mit diesen Grenzwerten konstruierte Ausdruck $x^2 - r^{(k)}x + s^{(k)}$ ist ein quadratischer Faktor von $P(x)$, der zu einem Paar konjugiert komplexer Wurzeln gehört.

- Das oben beschriebene Verfahren funktioniert gut, solange im Nenner nirgendwo eine Null steht. Sollte es nicht der Fall sein, muss man die Substitution $x_{neu} = x_{alt} + d$ für ein geeignetes d derchführen. In der Praxis probiert man unterschiedliche Werte $d = 1, -1, 2, -2, 3, -3, \ldots$, bis das Verfahren problemlos durchläuft. Behandeln Sie diese Fälle in Ihrem Programm mithilfe der Funktion `shift`.

- Überprüfen Sie Ihr Programm. Benützen Sie dafür folgende Polynome:

  * $x^5 - 6x^4 + 12x^3 - 12x^2 + 11x - 6$
  * $10x^6 - 27x^5 + 45x^4 - 73x^3 + 18x^2 + 9x - 2$
  * $x^7 - 7x^6 + 21x^5 - 35x^4 + 35x^3 - 21x^2 + 7x - 1$
  * $x^4 + x^3 - x^2 + x - 2$

Zusatzaufg.: Das QD-Verfahren von Heinz Rutishauser [Rutishauser1954] kann zwar benützt werden, um alle reellen Nullstelle eines Polynoms zu bestimmen, in der Praxis sind die Ergebnisse aber oft zu ungenau. Daher wendet man genauere Verfahren an, bei denen man die erhaltenen Nullstellen als Startwerte einsetzt. Implementieren Sie das Newtonverfahren[1] und benützen Sie es, um die erhaltenen Nullstellen zu präzisieren. (+2 Pkt.)

Zusatzaufg.: Erweitern Sie Ihr Programm, sodass auch alle komplexen Nullstelle eines Polynoms bestimmt werden. (+1 Pkt.)

[1] `https://de.wikipedia.org/wiki/Newtonverfahren`

Literatur

[Rutishauser1954] Rutishauser, H. Der Quotienten-Differenzen-Algorithmus. Journal of Applied Mathematics and Physics (ZAMP) 5, 233–251 (1954).

[Hermann2001] Hermann, Martin. Numerische Mathematik. Oldenbourg Wissenschaftsverlag, S. 246-253 (2001).