Computermathematik (für Informatik)

Hausübung | Jänner 2009

Aufgabenstellung: 3
Abgabetermin: 28. 02. 2009

Aufgabe 1 (Kurvendiskussion)**.**

(a) Führen Sie eine Kurvendiskussion der Funktion

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

mit Hilfe von Sage durch. Gehen Sie insbesondere auf die folgenden Punkte ein: Definitionsbereich, Stetigkeitsbereich, Nullstellen, Extrema, Wendepunkte, Monotonie, Krümmungsverhalten, Verhalten am Rande des Definitionsbereiches.

(b) Setzen Sie die Kurvendiskussion in LaTeX. Die einzelnen Rechenschritte sollen in vollständigen Sätzen beschrieben werden. Alle Zwischenschritte sind anzugeben. Fügen Sie ausserdem eine in Sage erzeugte Graphik hinzu, die alle wichtigen Eigenschaften der Funktion f darstellt.

Aufgabe 2. Berechnen Sie den folgenden Grenzwert und setzen Sie die Lösung in LaTeX. Die einzelnen Rechenschritte sollen in vollständigen Sätzen beschrieben werden. Alle Zwischenschritte sind anzugeben.

$$\lim_{x\to\frac{\pi}{2}^-} (\tan x)^{\cos x}.$$

Aufgabe 3 (Newton Interpolation)**.**
Das Interpolationspolynom zu $n+1$ paarweise verschiedene Stützstellen $x_0, x_1, \ldots, x_n$ und Werten $y_0, y_1, \ldots, y_n$ ist das Polynom p kleinsten Grades, für das gilt:

$$p(x_i) = y_i \quad \text{für alle } i = 0, \ldots, n$$

Eine Möglichkeit dieses Problem zu lösen, ist mit Hilfe des **Newton'schen Interpolationsverfahren**. Dafür wählt man den folgenden Ansatz für das Interpolationspolynom:

$$p(x) = c_0 + c_1(x-x_0) + c_2(x-x_0)(x-x_1) + \cdots + c_n(x-x_0)(x-x_1)\cdots(x-x_{n-1})$$

Die Koeffizienten $c_0, c_1, \ldots, c_n$ werden mit dem folgenden Differenzenschema berechnet:

$$\begin{array}{c|lllll}
x_0 & f[x_0] & & & \\
 & & f[x_0,x_1] & & \\
x_1 & f[x_1] & & f[x_0,x_1,x_2] & \\
 & & f[x_1,x_2] & & f[x_0,x_1,x_2,x_3] \\
x_2 & f[x_2] & & f[x_1,x_2,x_3] & \\
 & & f[x_2,x_3] & & \\
x_3 & f[x_3] & & & \\
\vdots & \vdots & & &
\end{array}$$

Dabei werden die Ausdrücke $f[x_i, x_{i+1}, \ldots, x_{i+k}]$ folgendermassen rekursiv berechnet:

$$
\begin{aligned}
f[x_k] &= y_k \\
f[x_i, x_{i+1}, \ldots, x_{i+k}] &= \frac{f[x_{i+1}, \ldots, x_{i+k}] - f[x_i, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}
\end{aligned}
$$

Die Koeffizienten des Interpolationspolynoms $p(x)$ sind dann gegeben als:

$$c_k = f[x_0, x_1, \ldots, x_k], \quad \text{für } k = 0, 1, \ldots, n$$

Beispiel: Man berechne das Interpolationspolynom $p(x)$ aus den Interpolationswerten:

x_k	-3	0	1	2	3
y_k	1	-1	0	1	0

$$
\begin{array}{r|lllll}
-3 & 1 & & & & \\
 & & \frac{-1-1}{0-(-3)} = -\frac{2}{3} & & & \\
0 & -1 & & \frac{1-(-\frac{2}{3})}{1-(-3)} = \frac{5}{12} & & \\
 & & \frac{0-(-1)}{1-0} = 1 & & \frac{0-\frac{5}{12}}{2-(-3)} = -\frac{1}{12} & \\
1 & 0 & & \frac{1-1}{2-0} = 0 & & \frac{-\frac{1}{3}-\left(-\frac{1}{12}\right)}{3-(-3)} = -\frac{1}{24} \\
 & & \frac{1-0}{2-1} = 1 & & \frac{-1-0}{3-0} = -\frac{1}{3} & \\
2 & 1 & & \frac{-1-1}{3-1} = -1 & & \\
 & & \frac{0-1}{3-2} = -1 & & & \\
3 & 0 & & & &
\end{array}
$$

Das Interpolationspolynom lautet somit:

$$
\begin{aligned}
p(x) &= 1 - \frac{2}{3}(x+3) + \frac{5}{12}(x+3)x - \frac{1}{12}(x+3)x(x-1) - \frac{1}{24}(x+3)x(x-1)(x-2) \\
&= -\frac{1}{24}x^4 - \frac{1}{12}x^3 + \frac{13}{24}x^2 + \frac{7}{12}x - 1
\end{aligned}
$$

Schreiben Sie eine Funktion `newton_interpolation(data)` die das Newton'sche Interpolationsverfahren implementiert.
Input: `data` – Liste mit Interpolationspunkten in der Form

$$[(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)].$$

Output: das Interpolationspolynom

Aufgabe 4 (Polynom Arithmetik).
Ein Polynom lässt sich darstellen als Liste von Koeffizienten. Also entspricht dem Polynom

$$p(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_n \cdot x^n$$

die Liste $[a_0, a_1, \ldots, a_n]$.
Sage-Polynome können mit der Methode `.list()` in diese Form konvertiert werden. Zum Beispiel:

```
sage: R.<t> = QQ[]
sage: p = 4 + 3*t^2 - t^4
sage: p.list()
[4, 0, 3, 0, -1]
```

Schreiben Sie Funktionen, welche die Grundrechenarten für Polynome in Listendarstellung implementieren. Gefordert sind die folgenden Funktionen:

- `poly_add(p1, p2)`
  Input: $p1$, $p2$ – Polynome in Listendarstellung
  Output: Die Summe von $p1$ und $p2$ in Listendarstellung

- `poly_sub(p1, p2)`
  Input: $p1$, $p2$ – Polynome in Listendarstellung
  Output: Die Differenz von $p1$ und $p2$ in Listendarstellung

- `poly_mult(p1, p2)`
  Input: $p1$, $p2$ – Polynome in Listendarstellung
  Output: Das Produkt von $p1$ und $p2$ in Listendarstellung

- `poly_div(p1, p2)`
  Input: $p1$, $p2$ – Polynome in Listendarstellung
  Output: Ein Paar (q, r) wobei q der ganzahlige Anteil und r der Rest bei Polynomdivision von $p1 : p2$ ist. q, r beide in Listendarstellung

- `poly_eval(p, x)`
  Input: p – Polynom in Listendarstellung
  x – eine Zahl
  Output: Der Wert des Polynoms p an der Stelle x.

- `poly_latex(p)`
  Input: p – Polynom in Listendarstellung
  Ouput: Ein String mit einer LaTeX-Darstellung des Polynoms p.

Die in Sage eingebauten Funktionen zur Polynomarithmetik dürfen **nicht** verwendet werden.
