Anwesenheitsübung 3 zur Vorlesung ‘Numerische Methoden der Physik’ SS 2014

Bastian Knippschild, Christian Jost und Mitarbeiter

Bearbeitung in den Übungen am 22. – 24. 04. 2014

Approximation von Funktionen 2

Falls Sie es noch nicht getan haben, schreiben Sie Ihre Lösung der letzten Anwesenheitsübung so um, dass die Polynominterpolation eine eigenständige Funktion ist. Dieser Funktion sollten die Stützstellen, die Funktionswerte an den Stützstellen und ein Array, in dem die Koeffizienten gespeichert werden, übergeben werden.

Interpolieren Sie nun die Runge-Funktion

$$f(x) = \frac{1}{1 + b^2 x^2}$$

für $b \in \{1,2,5\}$ mit verschiedener Anzahlen von Stützstellen im Intervall $x \in [-1,1]$.

Nach der Berechnung der Newton-Koeffizienten können Sie die Anzahl der Stützstellen in dem Intervall erhöhen und das Interpolationspolynom an den neuen Stützstellen auswerten. Lassen Sie sich die Stützstellen und das ausgewertete Polynom als Tabelle auf den Bildschirm ausgeben. Plotten Sie die so erhaltene Tabelle, zum Beispiel mit dem Programm `gnuplot`. Was sehen Sie?

Approximation von Funktionen 3

In dieser Aufgabe sollen die Polynominterpolation mit der Tschebyschow-Interpolation verglichen werden. Dazu soll die Runge-Funktion mit Hilfe der Tschebyschow-Polynome im Intervall $x \in [-1,1]$ interpoliert werden.

Die n Koeffizienten c_k des Tschebyschow-Polynoms $T_n(x)$ vom Grad n können wie folgt konstruiert werden:

$$c_k = \frac{2}{n} \sum_{i=0}^{n} \cos\left(k\pi \frac{i + 0.5}{n}\right) f(x_i).$$

Hierbei sind x_i die Nullstellen von $T_n(x)$.

Um das Tschebyschow-Polynom auszuwerten, bietet sich der Clenshaw-Algorithmus an. Für gegebene Tschebyschow-Koeffizienten c_k und ein $x \in [-1,1]$ wird das Polynom wie folgt ausgewertet:

```
tmp = b1 = b2 = 0
for i = n-1, n-2, ..., 1; do
  tmp = 2 * x * b1 - b2 + c[i]
  b2 = b1
  b1 = tmp
end for
return x * b1 - b2 + c[0] / 2
```

Plotten Sie Ihr Ergebniss wie in der vorhergehenden Aufgabe und vergleichen Sie die beiden Ergebnisse. Was fällt Ihnen auf?

Rationale Interpolation

In der Vorlesung ist die rationale Interpolation besprochen worden. Dazu wird eine Funktion $f(x)$ durch ein rationales Polynom

$$\phi^{\mu,\nu}(x) = \frac{a_0 + a_1 x + \cdots + a_\mu x^\mu}{b_0 + b_1 x + \cdots + b_\nu x^\nu}$$

approximiert. Mit der Methode nach Neville ist es möglich, das Interpolationspolynom auszuwerten, ohne dass die Koeffizienten $\{a_i\}$ und $\{b_i\}$ bekannt sind. Dazu müssen $n = \mu + \nu + 1$ Stützstellen und die Werte der Funktion an den Stützstellen bekannt sein. Die Interpolation erfolgt durch folgenden Code:

```
for i = 0, 1, ..., n-1; do
  for j = 0, 1, ..., n-1; do
    T[i][j] = 0
  end for
end for
a = b = c = t = 0
for i = 0, 1, ..., n-1; do
  T[i][0] = f[i]
  for j = 1, 2, ..., i; do
    if (x - x[i]) ~ 0; then
      T[i][j] = T[i][j-1]
      continue
    else if (x[i] - x[i-j]) ~ 0; then
      a = 1
    else
      a = (x - x[i-j]) / (x - x[i])
    end if

    if j = 1; then
      t = 0
    else
      t = T[i-1][j-2]
    end if

    c = T[i][j-1] - T[i-1][j-1]

    if (T[i][j-1] - t) ~ 0; then
      T[i][j] = T[i][j-1]
      continue
    else if (c - (T[i][j-1] - t)) ~ 0; then
      b = 0
    else
      b = 1 - c / (T[i][j-1] - t)
    end if
    T[i][j] = T[i][j-1] + c/ (a * b - 1)
  end for
end for
return T[n-1][n-1]
```

Hierbei entspricht `(x - x[i]) ~ 0` $x - x_i \approx 0$. Implementieren Sie die rationale Interpolation und vergleichen Sie das Ergebnis mit den vorherigen Aufgaben.