Technische Universität Hamburg-Harburg
Institut für Numerische Simulation, E-10
Dr. Jens-Peter M. Zemke

Sommersemester 2008

Numerische Verfahren

Übungen und Lösungen, Blatt 2

Aufgabe 1: (Thema: Polynominterpolation.)

Es sind die folgenden vier verschiedenen Datensätze gegeben:

- $(x, y) = \{(0, 1), (1, 1), (2, 1), (3, 1), (4, 1)\}$,
- $(x, y) = \{(0, 0), (1, 0), (2, 1), (3, 0), (4, 0)\}$,
- $(x, y) = \{(0, 1), (1, 1), (2, 2), (3, 1), (4, 1)\}$,
- $(x, y) = \{(0, 1), (1, 2), (2, 5), (3, 10), (4, 17)\}$.

Skizzieren Sie (i.e., plotten Sie) die Datensätze zuerst, um ein Gefühl für den Verlauf der zugrunde liegenden Funktionen zu bekommen. Interpolieren Sie danach die Datensätze mittels Polynominterpolation

a) von Hand (mit allen erdenklichen Tricks und Abkürzungen),

b) mittels Matlab (Tipp: `help polyfit`, `help polyval`).

Welchen Weg zum jeweiligen Ergebnis würden Sie (nach etwas Nachdenken) bevorzugen?

Lösung zu Aufgabe 1:

Die Skizzen lassen Linearitäten oder Nullstellen erkennen. Insbesondere die Nullstellen in Knoten sind für Polynominterpolation in Lagrange-Form interessant. Auch verschobene Nullstellen können ausgenutzt werden. Der erste und letzte Datensatz zeigt, daß ein Polynom von niedrigerem Polynomgrad resultieren kann. Die Datensätze lassen sich wie folgt charakterisieren, ergo, schnell von Hand interpolieren:

- Der erste Datensatz beschreibt eine Gerade konstant Eins, also:

$$p_1(x) = 1.$$

- Der zweite Datensatz läßt sich schnell beschreiben, wenn wir an Polynominterpolation in der Lagrange-Form denken. Durch den Wert Null an vier der fünf Knoten fallen im Interpolationspolynom in der Lagrange-Form alle Lagrange-Polynome bis auf das dritte weg und müssen nicht berechnet werden. Der Faktor vor dem nicht wegfallenden Teil ist Eins. Es handelt sich also um das dritte Lagrange-Polynom $\ell_2(x)$ zu den Knoten $0, 1, 2, 3, 4$:

$$\begin{aligned} p_2(x) &= \ell_2(x) \\ &= \frac{(x-0)(x-1)(x-3)(x-4)}{(2-0)(2-1)(2-3)(2-4)} \\ &= \frac{1}{4}x^4 - 2x^3 + \frac{19}{4}x^2 - 3x. \end{aligned}$$

- Bei diesem Datensatz handelt es sich um den um Eins nach oben verschobenen zweiten Datensatz. Damit ist das Interpolationspolynom aber das LAGRANGE-Polynom $\ell_2(x)$ zu den Knoten $0, 1, 2, 3, 4$ plus Eins:

$$\begin{aligned} p_3(x) &= \ell_2(x) + 1 \\ &= \frac{1}{4}x^4 - 2x^3 + \frac{19}{4}x^2 - 3x + 1. \end{aligned}$$

- Bei diesem Datensatz handelt es sich um eine um Eins verschobene Parabel:

$$p_4(x) = x^2 + 1.$$

Die Interpolation mittels `polyfit` und `polyval` befindet sich in dem M-File `aufg02f01.m`. Nach Aufruf ohne Parameter werden die Datensätze und die zugehörigen Interpolationspolynome in einen Plot gezeichnet.

Wenn Sie alle Tricks selber erkannt haben, werden Sie sicherlich den Weg von Hand bevorzugen. Wenn es solche „Abkürzungen“ gibt, ist dieser numerisch auch sinnvoller, da bei kleinen Störungen sonst eine völlig andere Funktion resultieren kann. Denken Sie nur an den ersten Datensatz. Bei kleinen Störungen der Meßwerte würde garantiert ein Polynom vierten Grades resultieren. Hier ist es oft besser, durch `polyfit` erst einmal den Datensatz *approximierende* Polynome kleinen Grades zu finden.

Aufgabe 2: (Thema: Polynominterpolation – mittels Newtonscher Interpolation.)

Gegeben sei die Funktion

$$w(x) := \sqrt{x}.$$

- Sie interpolieren $w(x)$ an den Knoten $0, 1, 4, 9$ mittels des Polynomes p, sind aber zunächst nur an einem Näherungswert für $w(2)$ interessiert. Wenden Sie zur Bestimmung dieser Näherung den Algorithmus von NEVILLE und AITKEN an und erstellen ein Tableau der Form, wie sie in Bemerkung 2.11 des Skriptes zu sehen ist.
- Warum ist die Fehlerabschätzung des Skriptes aus Bemerkung 2.22 hier zwar *an*wendbar, aber nicht wirklich *ver*wendbar?
- Nun scheint es so zu sein, dass Sie das Interpolationspolynom auch noch an anderen Stellen ausgewertet benötigen. Da Sie ja bereits Übung im Tableauaufstellen haben, entschließen Sie sich, die Dividierten Differenzen für eine Newtoninterpolation zu berechnen. Geben Sie Ihr Newtonsches Interpolationspolynom an. Erstellen Sie ein MatLab-Programm, welches die Dividierten Differenzen berechnet und anschließend das Newtonsche Interpolationspolynom an den Stellen $x \in \{\frac{1}{2}, 2, 4\}$ auswertet.

Lösung zu Aufgabe 2:

Das Tableau ergibt sich nach (2.5) des Skriptes zu:

$$\begin{array}{llll}
x_0 = 0, \quad y_0 = 0 & & & \\
 & \frac{1\cdot(2-0)-0\cdot(2-1)}{1-0} = 2 & & \\
x_1 = 1, \quad y_1 = 1 & & \frac{\frac{4}{3}\cdot(2-0)-2\cdot(2-4)}{4-0} = \frac{5}{3} & \\
 & \frac{2\cdot(2-1)-1\cdot(2-4)}{4-1} = \frac{4}{3} & & \frac{\frac{41}{30}\cdot(2-0)-\frac{5}{3}\cdot(2-9)}{9-0} = \frac{8}{5} \\
x_2 = 4, \quad y_2 = 2 & & \frac{\frac{8}{5}\cdot(2-1)-\frac{4}{3}\cdot(2-9)}{9-1} = \frac{41}{30} & \\
 & \frac{3\cdot(2-4)-2\cdot(2-9)}{9-4} = \frac{8}{5} & & \\
x_3 = 9, \quad y_3 = 3 & & &
\end{array}.$$

Die Fehlerabschätzung aus Bemerkung 2.22 des Skriptes, explizit gegeben durch

$$\|f - p\|_\infty \leqslant \frac{\|f^{(n+1)}\|_\infty}{(n+1)!}\|\omega\|_\infty,$$

ist zwar anwendbar, liefert aber nur die Information, daß der Fehler kleiner gleich ∞ ist, da die Wurzel im Punkt $(0,0)$ nicht differenzierbar ist und damit

$$\|f^{(n+1)}\|_\infty \equiv \|(\sqrt{x})^{(4)}\|_\infty := \sup_{x\in(0,9]} \left|-\frac{15}{16}x^{-7/2}\right| = \infty$$

gilt. Diese Information ist zwar nicht falsch, aber auch nicht *sonderlich* aussagekräftig.

Setzt man $x_0 = 0$, $x_1 = 1$, $x_2 = 4$, $x_3 = 9$, so berechnen sich die Dividerten Differenzen zur Newton Interpolation nach Algorithmus 2.14 des Skriptes zu:

$$\begin{array}{llllll}
x_0 = 0, & y_0 = 0 & & & & \\
 & & \frac{1-0}{1-0} = 1 & & & \\
x_1 = 1, & y_1 = 1 & & \frac{\frac{1}{3}-1}{4-0} = -\frac{1}{6} & & \\
 & & \frac{2-1}{4-1} = \frac{1}{3} & & \frac{-\frac{1}{60}+\frac{1}{6}}{9-0} = \frac{1}{60} & . \\
x_2 = 4, & y_2 = 2 & & \frac{\frac{1}{5}-\frac{1}{3}}{9-1} = -\frac{1}{60} & & \\
 & & \frac{3-2}{9-4} = \frac{1}{5} & & & \\
x_3 = 9, & y_3 = 3 & & & &
\end{array}$$

Und entsprechnd ergibt sich das Newtonsche Interpolationspolynom zu

$$\begin{aligned}
&\frac{1}{60}\cdot(x-4)(x-1)(x-0) - \frac{1}{6}\cdot(x-1)(x-0) + 1\cdot x + 0 \\
=&\frac{1}{60}(x-4)(x-1)x - \frac{1}{6}(x-1)x + x.
\end{aligned}$$

Die Umsetzung in MatLab finden Sie in der Datei `aufg02script02.m`.

Aufgabe 3: (Thema: Fehler(schranken) der Polynominterpolation und der Interpolation mittels Splines.)

In der Materialsammlung finden Sie eine zip-Datei namens `intererror.zip`. Entpacken Sie diese und lesen Sie sich die Kommentare in der Datei `aufg02script01.m` durch. Plotten Sie die Funktion `exptestfunc.m` über dem Intervall `intval`, welche in dem Skript benutzt werden. Führen Sie nun das Skript aus, und diskutieren Sie mit ihren Kommilitonen über die Ergebnisse. Öffnen Sie die Datei `aufg02error.m` und vollziehen Sie mit Hilfe der Kommentare den logischen Aufbau der Funktion nach. Ändern Sie nun im Skript `aufg02script01.m` die Variablen nach eigenem Gutdünken, welche mit einem Kommentar der Form `<--- Hier dürfen Sie` beschriftet sind und diskutieren Sie die Resultate.

Lösung zu Aufgabe 3:

Im Grunde ist dies keine Lösung im klassischen Sinne, sondern beschreibt nur die Beobachtungen und mögliche Ursachen dieser, wenn man das Skript `aufg02script01.m` für verschiedende Werte von `n` und für ein beliebiges aber *festes* Intervall ablaufen lässt.

In exakter Rechnung ist für die Polynominterpolation und bei der Wahl von äquidistanten Knoten ein Verhalten der Fehlerkurven zu erwarten, wie es in der Abbildung 2.1 im Vorlesungsskript zu sehen ist. Fehler nehmen am Rand des Interpolationsintervalles zu, was im Grunde auf die Funktion

$$\omega(x) = \prod_{i=0}^{n}(x - x_i)$$

zurückführbar ist, welche nach Satz 2.21 des Vorlesungsskriptes in die Fehlerdarstellung der Polynominterpolation eingeht. Werden die inneren Knoten $x_1, \ldots, x_{n-1}$ des äquidistanten Gitters, wobei die Subintervallbreite mit h bezeichnet sei, um $h/2$ nach links respektive nach rechts verschoben, steht zu vermuten, dass die Fehlerkurven auf der Seite ausgedämpft werden, auf der nun ein Subintervall der Länge $h/2$ existiert. Im Gegenzug ist nun auf der anderen Seite ein Subintervall der Länge $\frac{3}{2}h$ vorhanden. Ergo ist dort ein lokaler Anstieg des Fehlers zu erwarten. Diese Vorhersagen werden für kleine `n` auch bestätigt durch die Fehlerkurven.

Man beachte, dass zwar in exakter Arithmetik `polyfit/polyval` das gleiche Interpolationspolynom wie Newton konstruiert und auswertet, numerisch jedoch auch bereits für moderat große `n` (z.B. `n=10` bei der Intervallwahl $[0,1]$) ob der verschiedenen Vorgehensweise sichtbar andere Ergebnisse liefert. Wählt man nun `n` immer größer, so wird die Matrix für das lineare Gleichungssystem, welches `polyfit` zu Grunde liegt, immer schlechter konditioniert, was an dieser Stelle erstmal nur heißen soll, dass kleine Störungen in der Eingabe bzw. im Aufstellen der Matrix große Störungen im numerischen Ergbnis relativ zur wahren Lösung ergeben können. Anschließendes Anwenden des Horner Schemas in `polyval` bringt weitere Rundungs- und Auslöschungsfehler ins Spiel, die sich je weiter kumulieren können desto höher der Polynomgrad ist. Und dieser ist für die gegebene Testfunktion tatsächlich für `n` Subintervalle auch gleich `n`. Ähnlich verhält es sich bei der Interpolation nach Newton. Hier wird zwar kein lineares Gleichungssystem gelöst, jedoch wird auch hier auf ein Horner Schema ähnliches Verfahren zur Auswertung des Interpolationspolynomes zurückgegriffen. Folglich kumulieren sich hier die Rundungs- und Auslöschungsfehler in sehr ähnlicher Weise. Neben den Rundungsfehlern hier gehen natürlich auch die Auslöschungsfehler der vorher berechneten Dividierten Differenzen in das Ergbnis ein. Die Interpolation mittels (3,2)-Splines besitzt nicht nur den Vorteil der theoretisch gesicherten gleichmäßigen Konvergenz sondern braucht auch zur Auswertung an einer festen Stelle $\hat{x}$ stets jeweils nur ein Polynom dritten Grades heranziehen. Exzessives Kumulieren von Rundungs- und Auslöschungsfehlern, wie es im Horner Schema bei der Polynominterpolation für große `n` auftritt, ist hier also nicht gegeben.