Programmieren für Physiker

Interfakultatives Institut für Anwendungen der Informatik
Institut für Theoretische Teilchenphysik

Prof. Dr. M. Steinhauser, Dr. A. Mildenberger
http://comp.physik.kit.edu

SS 2011 – Blatt 07
Bearbeitungszeitraum: bis 01. Juni 2011

Aufgabe 17: Interpolation mit kubischen Splines **Pflichtaufgabe**

Gegeben seien die Stützstellen $\Delta = \{a = x_0 < x_1 < \ldots < x_n = b\}$ und die zugehörigen Funktionswerte $Y = \{y_0, y_1, \ldots, y_n\}$. In dieser Aufgabe soll eine kubische Splinefunktion $S_\Delta(Y; x)$ durch diese Punkte bestimmt werden. Bei kubischen Splines werden abschnittsweise Polynome dritter Ordnung durch je zwei aufeinanderfolgende Punkte gelegt, so dass die Polynome selbst und ihre ersten beiden Ableitungen stetig an den Stellen x_i, $i = 1, \ldots, n-1$ anschließen. Das Problem wird eindeutig lösbar, wenn zusätzlich zwei Randbedingungen vereinbart werden: In dieser Aufgabe soll gefordert werden, dass die zweite Ableitung der Splinefunktion an den Rändern $x = a$ und $x = b$ verschwindet.
Zentral bei der Berechnung der kubischen Splinefunktion sind die sogenannten „Momente“, die zweiten Ableitungen an den Stützstellen $M_j := S''_\Delta(Y, x_j)$. Für diese gilt das folgende tridiagonale $(n+1) \times (n+1)$ Gleichungssystem

$$\begin{pmatrix} 2 & \lambda_0 & 0 & \ldots & 0 \\ \mu_1 & 2 & \lambda_1 & \ddots & \vdots \\ 0 & \mu_2 & 2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \lambda_{n-1} \\ 0 & \ldots & 0 & \mu_n & 2 \end{pmatrix} \cdot \begin{pmatrix} M_0 \\ M_1 \\ M_2 \\ \vdots \\ M_n \end{pmatrix} = \begin{pmatrix} d_0 \\ d_1 \\ d_2 \\ \vdots \\ d_n \end{pmatrix}$$

mit den Größen $(j = 1, \ldots, n-1)$

$$\lambda_j = \frac{x_{j+1} - x_j}{x_{j+1} - x_{j-1}}\,, \qquad \mu_j = \frac{x_j - x_{j-1}}{x_{j+1} - x_{j-1}}\,, \qquad d_j = \frac{6}{x_{j+1} - x_{j-1}} \left(\frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}} \right)$$

und den Randbedingungen $\lambda_0 = \mu_n = d_0 = d_n = 0$.
In diesem System wird zunächst die untere Nebendiagonale eliminiert. Setze $\mu_0 = 2$ und führe für $i = 1, 2, \ldots, n$ nacheinander aus: $f := -\mu_i/\mu_{i-1}$,
$\mu_i := 2 + f \cdot \lambda_{i-1}$,
$d_i := d_i + f \cdot d_{i-1}$.
Die neu berechneten Werte $\mu_0, \ldots, \mu_n$ bilden dann die Diagonale des umgeformten Systems, die Nebendiagonale $\lambda_0, \ldots, \lambda_{n-1}$ bleibt unverändert.
Nun die Rücksubstitution: $M_n := d_n/\mu_n$ und dann für $i = n-1, \ldots, 0$: $M_i = (d_i - \lambda_i M_{i+1})/\mu_i$.
Mit den M_i sind die Koeffizienten von $S_\Delta(Y; x)$ vollständig bekannt:

$$S_\Delta(Y; x) = \alpha_j + \beta_j(x - x_j) + \gamma_j(x - x_j)^2 + \delta_j(x - x_j)^3 \quad \text{für} \quad x \in [x_j, x_{j+1}]$$

mit

$$\alpha_j = y_j, \qquad \beta_j = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{2M_j + M_{j+1}}{6}(x_{j+1} - x_j), \qquad \gamma_j = M_j/2, \qquad \delta_j = \frac{M_{j+1} - M_j}{6(x_{j+1} - x_j)}.$$

Implementieren Sie dieses Verfahren in einem C++-Programm. Verwenden Sie zum Testen die bereits von 6. Übungsblatt bekannte Datei `/home/ck11/daten/a15-interpol.dat` (steht auch

auf der WWW-Seite), die einige Wertepaare (x, y) enthält. Sie dürfen voraussetzen, dass die x-Werte aufsteigend geordnet sind.
Werten Sie die Splinefunktion an 300 äquidistanten x-Werten zwischen minimaler und maximaler Stützstelle aus und schreiben Sie jeweils x und den Wert der Splinefunktion zeilenweise in eine Datei. Die ursprüngliche Punktedatei und die von Ihnen erzeugte Ergebnisdatei schauen Sie sich bitte in einem geeigneten Plot-Programm an.

Aufgabe 18: Nullstellenbestimmung **Pflichtaufgabe**

Zwei Methoden, um Nullstellen einer (stetigen) Funktion numerisch zu bestimmen, sollen in dieser Aufgabe programmiert und verglichen werden: das Bisektions- und das Sekantenverfahren (in der sogenannten *regula falsi*-Variante).
Bei beiden Verfahren startet man mit einem Intervall $[x_0, x_1]$, das eine Nullstelle der gegebenen Funktion $f(x)$ einschließen soll. Lassen Sie also die Werte x_0 und x_1 eingeben und überprüfen Sie anhand der Vorzeichen von $f(x_0)$ und $f(x_1)$, ob eine Nullstelle im Intervall vorhanden ist.
Je nach gewähltem Verfahren wird nun eine neue Stützstelle x_{test} bestimmt:

$$\text{Bisektionsverfahren}: \quad x_{\text{test}} = \frac{x_0 + x_1}{2}$$
$$\text{Sekantenverfahren}: \quad x_{\text{test}} = \frac{x_0 f(x_1) - x_1 f(x_0)}{f(x_1) - f(x_0)}$$

Der Ausdruck beim Sekantenverfahren ergibt sich aus der Nullstelle derjenigen Geraden, die durch die Punkte $(x_0, f(x_0))$ und $(x_1, f(x_1))$ geht. An der neu gewonnen Stelle x_{test} ist die Funktion auszuwerten und abhängig vom Vorzeichen wird entweder die obere oder die untere Intervallgrenze neu gesetzt, so dass die Nullstelle stets innerhalb des betrachteten Intervalls bleibt. Wiederholen Sie diese Schritte, bis entweder die Intervallgröße kleiner als $\varepsilon_x = 10^{-4}$ oder der Funktionswert $|f(x_{\text{test}})| < \varepsilon_f = 10^{-3}$ wird.
Ihr Programm soll folgendermaßen strukturiert sein: Definieren Sie für die zu untersuchende Funktion $f(x)$ eine C++-Funktion und schreiben Sie für jedes der beiden Verfahren eine Funktion, die als Übergabewerte das Anfangsintervall und die gewünschten Genauigkeitswerte erhält. Geben Sie während der Iteration bei jedem Schritt die Schrittzahl, das Nullstellen-Intervall und den Funktionswert $f(x_{\text{test}})$ aus.

Testfunktionen: (i) $f(x) = \cos(x) - x$, $x_0 = -10$, $x_1 = 10$.
(ii) $f(x) = e^x - x^3$, $x_0 = 2$, $x_1 = 10$ (`double`-Arithmetik empfohlen).

Hinweis:
Auf den Webseiten befindet sich ein Informationsblatt, auf welchem die Namen und Bedeutungen von wichtigen mathematischen Funktionen in C++ aufgelistet sind.
