ETHZ, D-MATH

Prüfung

Numerische Methoden D-PHYS, FS 2012

Dr. V. Gradinaru

24.08.2012

Prüfungsdauer: 180 Minuten

Maximal erreichbare Punktzahl: 120

1 Aufgabe (20 Punkte): Fixpunktiteration

Die Lösungen der Gleichung

$$x^5 + 3x + 4 = e^{\cos x} + 8x^2 \tag{1.1}$$

sollen in Intervall $[-1, 2]$ mit Hilfe von Fixpunktiterationen bestimmt werden.
Arbeiten Sie in den folgenden Teilaufgaben mit dem Template `aufgabe01/fixpoint.py`.

(i) Die Funktionen

$$\begin{aligned}
\Phi_1(x) &= \frac{e^{\cos x} - 4}{x^4 - 8x + 3} \\
\Phi_2(x) &= \frac{e^{\cos x} - 3x - 4}{x^4 - 8x} \\
\Phi_3(x) &= \frac{e^{\cos x} - 3x - 4}{x^4 + 2x^3 + 4x^2} + 2
\end{aligned}$$

definieren drei Fixpunktiterationen $x^{(k+1)} = \Phi_i(x^{(k)})$, $i = 1, 2, 3$. Zeigen Sie, dass diese konsistent mit der Gleichung (1.1) sind.

(ii) Implementieren Sie die `PYTHON` Funktion `fixIter(F,x0,tol,maxit)` innerhalb von `fixpoint.py`, die eine Fixpunktiterationen `F`

$$x^{(k+1)} = F(x^{(k)}) \ , \ k = 0, 1, 2, ...$$

mit dem Startwert $x_0 =$ `x0` bis zur Genauigkeit `tol` oder bis `maxit` Iterationen ausführt. Die Rückgabewerte sind dabei ein Vektor mit allen Iterationswerten x_i, ein Konvergenzflag `Converged` (`Converged = True` falls die Fixpunktietration konvergiert und `Converged = False` sonst) und die Anzahl der benötigten Iterationen um die gewünschte Genauigkeit zu erreichen $\mid x_{i+1} - x_i \mid <$ `tol`.

(iii) Verwenden Sie die Funktion `fixIter` für folgende Fixpunktiterationen und Startwerte Kombinationen:

(a) Φ_1 mit $x_0 = 0$

(b) Φ_2 mit $x_0 = 0.5$

(c) Φ_3 mit $x_0 = 1$

Benutzen Sie als Abbruchkriterien `tol` $= 10^{-13}$ und `maxit` $= 100$.
Geben Sie für jede Fixpunktiteration die 5 ersten Iterationswerte und der letzte Iterationswert (wenn die gewünschte Genauigkeit erreicht wurde) aus.

(iv) Bestimmen Sie numerisch die Konvergenzordnung der drei Fixpunktiterationen.

2 Aufgabe (16 Punkte): Methode der kleinsten Quadrate

Um die Amplitude A und den Phasenwinkel ϕ einer Schwingung

$$b(t) = A\sin(2t + \phi) \tag{2.1}$$

zu bestimmen, sind an vier Zeitpunkten t_k die Auslenkungen b_k gemessen worden:

t_k	0	$\pi/4$	$\pi/2$	$3\pi/4$
b_k	1.6	1.1	-1.8	0.9

Arbeiten Sie in den folgenden Teilaufgaben in der Datei `aufgabe02/leastsquares.py`.

(i) Benutzen Sie die trigonometrische Formel

$$\sin(A + B) = \sin(A)\cos(B) + \cos(A)\sin(B)$$

und führen Sie die neuen Unbekannten $\alpha = A\cos(\phi)$ und $\beta = A\sin(\phi)$ ein, um ein lineares Ausgleichsproblem zu erhalten.

(ii) Bestimmen Sie α und β im Sinne der kleinsten Fehlerquadrate. Verwenden Sie hierzu jeweils die Normalengleichungen und die QR Orthogonaltransformation. Geben Sie zu beiden Methoden den Residuenvektor an.

(iii) Berechnen Sie aus (ii) A und ϕ und erstellen Sie einen Plot welcher die Daten und (2.1) darstellt. Verwenden Sie hierzu die Datei `leastsquares.py`.

3 Aufgabe (32 Punkte): Ordnung ist nicht alles

Arbeiten Sie in den folgenden Teilaufgaben mit dem Template `aufgabe03/ordnungistnichtalles.py`.

(i) Berechnen Sie die analytische Lösung des Anfangswertproblems

$$\dot{y} = |1.1 - y| + 1, \quad y(0) = 1. \tag{3.1}$$

(ii) Wir wollen nun die Lösung von (3.1) numerisch berechnen. Implementieren Sie hierzu folgende Verfahren in `ordnungistnichtalles.py`:

- ein explizites Euler-Verfahren,
- ein Runge–Kutta-Verfahren der Ordnung 3 gegeben durch das Butcher–Tableau

$$
\begin{array}{c|ccc}
0 & & & \\
1/2 & 1/2 & & \\
1 & -1 & 2 & \\
\hline
 & 1/6 & 2/3 & 1/6
\end{array}
$$

Beide Integrations-Routinen sollen als Input die rechte Seite $F(y)$ der Differentialgleichung $\dot{y} = F(y)$, die Anfangsbedingung `y0`, die Startzeit `tStart`, die Stopzeit `tEnd` und die Anzahl Schritte `N` entgegen nehmen. Der Output soll der Loesungsvektor `y` zu den Zeiten im Vektor `t` sein.

(iii) Für das Anfangswertproblem (3.1) plotten Sie die Fehler des expliziten Euler und Runge–Kutta-Verfahren am Endzeitpunkt T als Funktion der Schrittweiten $h = 1/2^k$, $k = 4, \ldots, 13$ in einer doppeltlogarithmischen Skala. Nehmen Sie für den Endzeitpunkt jeweils $T = 0.09$ und $T = 0.1$. Ergänzen Sie hierzu die PYTHON Datei `ordnungistnichtalles.py`. Hinweis: die exakte Lösung des Anfangswertproblems (3.1) ist $y(t = 0.09) = 1.094675696201649$ und $y(t = 0.1) = 1.104700834614225$.

(iv) Welche Konvergenzordnungen werden theoretisch erwartet? Welche Konvergenzordnung wird bei $T = 0.09$ und bei $T = 0.1$ jeweils beobachtet? Erklären Sie das Beobachtete.

4 Aufgabe (28 Punkte): Numerische Quadratur

Es soll folgendes Integral

$$I = \int_0^2 \frac{e^x - 1}{e - 1} \mathrm{d}x = \frac{e^2 - 3}{e - 1}$$

numerisch berechnet werden.
Arbeiten Sie in den folgenden Teilaufgaben mit dem Template `aufgabe04/quad.py`.

(i) Implementieren Sie Ihre eigene `PYTHON` Routine `simpson(func,a,b,N)` in der Vorlage `quad.py` welche das Integral einer skalaren Funktion `func` auf dem Intervall $[a, b]$ mit der zusammengesetzten Simpson-Regel auf N äquidistanten Teilintervallen approximiert.

(ii) Werten Sie das Integral I mit Ihrer Implementierung aus (i) für $N = 2, ..., 500$ aus. Bestimmen Sie aus Ihren Ergebnissen die Ordnung p der algebraischen Konvergenz. Verwenden Sie die Funktion `polyfit` aus `numpy` um die Konvergenzordnung p zu bestimmen.

(iii) Das Integral I soll nun mittels Monte-Carlo Quadratur numerisch ausgewertet werden. Implementieren Sie hierzu die Funktion `mc(func,a,b,N)` in der Vorlage `quad.py` welche das Integral einer skalaren Funktion `func` auf dem Intervall $[a, b]$ mit `N` gelichmässig verteilten Samples berechnet.

(iv) Werten Sie das Integral I mit Ihrer Implementierung aus (iii) für $N = 2, ..., 500$ aus. Bestimmen Sie aus Ihren Ergebnissen die Ordnung p der algebraischen Konvergenz. Verwenden Sie die Funktion `polyfit` aus `numpy` um die Konvergenzordnung p zu bestimmen.

(v) Was ist die genaue Bedeutung der durch die Monte-Carlo Quadratur mit N Samples berechneten Approximation des Integrals I?

(vi) Es soll nun die Methode der Varianzkontrolle (control variates) für die Bestimmung des Integrals I benutzt werden. Approximieren Sie hierzu die Exponentialfunktion mittels einer Taylor-Reihe mit Entwicklungspunkt am Ursprung bis (und inklusive) dritter Ordnung.

Bestimmen Sie die algebraische Konvergenzordnung p für $N = 2, ..., 500$ mit Ihrer Implementierung der Monte-Carlo Quadratur mit Varianzkontrolle. Verwenden Sie die Funktion `polyfit` aus `numpy` um die Konvergenzordung p zu bestimmen.

5 Aufgabe (24 Punkte): Steif oder nicht steif?

Bestimmen Sie ob die Differentialgleichung

$$y''(t) = -201y' - 200y^2 + 2 \ , \ t \in [0, 20]$$

steif ist. Nennen Sie ein entsprechendes Verfahren für die numerische Integration der obigen Differentialgleichung.
Falls notwendig, arbeiten Sie in der Datei `aufgabe05/steif.py`.