Prof. Dr. Karsten Urban
Dipl.-Math. oec. Klaus Stolle
Institut für Numerische Mathematik
Universität Ulm

WiSe 2015/2016
Blatt 10
08.01.2016

Angewandte Numerik 2

Abgabetermin: Freitag, 15.01.2016, **vor der Übung**

Für dieses Übungsblatt gibt es 8 Theorie- und 14 Matlab-Punkte, sowie 3 Theorie- und 9 Matlab-Zusatzpunkte. Punkte, die mit einem * gekennzeichnet sind, sind Zusatzpunkte.
Die 50-Prozent-Grenzen liegen aktuell (inklusive Blatt 10) bei 79,5 Theoriepunkten und 94 Matlabpunkten.

Aufgabe 37 *(Stabilität)* (8T+3T* Punkte)

a) Berechnen Sie das Stabilitätsgebiet des aus der Trapezregel resultierenden impliziten Verfahrens nach Beispiel 3.2.3 (c) des Skripts (vgl. Aufgabe 26 Teil f)).

b) Ist dieses Verfahren A-stabil?

Hinweise:

i) Wenden Sie zunächst die Verfahrensvorschrift des aus der Trapezregel resultierenden Verfahrens auf das skalare Modellproblem $y' = \lambda y, \;\; y(0) = 1$ an.

ii) Bringen Sie die so erhaltene Gleichung auf die Form $y_h(t_{j+1}) = R(\lambda h)\, y_h(t_j)$.

iii) Berechnen Sie das Stabilitätsgebiet $S = \{z \in \mathbb{C} : |R(z)| \leq 1\}$.

Aufgabe 38 *(Programmieraufgabe: Implizite Runge-Kutta-Verfahren)* (6M+2M+6M+3M*+6M* Punkte)

a) Schreiben Sie analog zu Aufgabe 32 von Blatt 8 eine Matlabfunktion `yk = rungeKuttaImplizit(f, y0, tk, bt, anzFixpktIt)`, die für einen gegebenen Startwert $y^0 \in \mathbb{R}^n$ eine Lösung $y : \mathbb{R} \to \mathbb{R}^n$, $n \in \mathbb{N}$ der Anfangswertaufgabe

$$y' = f(t, y), \qquad y(t_0) = y^0$$

mit einem impliziten Runge-Kutta-Verfahren berechnet.

Sie dürfen dazu Ihre Funktion `yk = rungeKutta(f, y0, tk, bt)` aus Aufgabe 32 erweitern. Die Parameter sind wie in Aufgabe 32: `f` ist die Funktion f als *function handle*, `y0` ist der Startwert $y^0 \in \mathbb{R}^n$ und `tk` ist ein Gitter mit den diskreten Zeitpunkten t_k. Der Rückgabewert `yk` ist der Vektor der einzelnen Näherungswerte y^k.

Verwenden Sie zur Lösung des impliziten Gleichungssystems, mit dem die Funktionen $k_1(t, h, y), \ldots, k_m(t, h, y)$ berechnet werden, eine Fixpunktiteration mit einer festen Anzahl Schritten und Startwerten $k_j(t, h, y) = 0$ $(j = 1, \ldots, m)$. Der zusätzliche Parameter `anzFixpktIt` gibt die Anzahl der durchzuführenden Iterationen an.

Ihre Matlabfunktion `yk = rungeKuttaImplizit(f, y0, tk, bt, anzFixpktIt)` soll (völlig analog zu Aufgabe 32) den Algorithmus eines impliziten Runge-Kutta-Verfahrens unabhängig von den konkreten Werten für α, β und γ realisieren. Die konkreten Werte für α, β und γ, also das Butcher-Tableau, soll ihre Matlabfunktion `yk = rungeKuttaImplizit(f, y0, tk, bt, anzFixpktIt)` über den Parameter `bt` erhalten. `bt` soll dabei eine Struktur mit den Komponenten

i) `bt.m` für die Stufenanzahl des Runge-Kutta-Verfahrens,
ii) `bt.alpha` für den Spaltenvektor $\alpha = (\alpha_1, \ldots, \alpha_m)$ des Runge-Kutta-Verfahrens,
iii) `bt.beta` für die Matrix $\beta = (\beta_{i,j})$ $(i, j = 1, \ldots, m)$ des Runge-Kutta-Verfahrens und
iv) `bt.gamma` für den Zeilenvektor $\gamma = (\gamma_1, \ldots, \gamma_m)$ des Runge-Kutta-Verfahrens

sein. Die Matrix β kann jetzt, anders als beim expliziten Runge-Kutta-Verfahren aus Aufgabe 32, auf und oberhalb der Diagonalen auch Werte $\beta_{i,j} \neq 0$ enthalten.

b) Schreiben Sie eine Matlabfunktion `bt = rK3Implizit`, die das 3-stufige Butcher-Tableau

$$\begin{array}{c|ccc} 0 & \frac{1}{6} & -\frac{1}{6} & 0 \\ \frac{1}{2} & \frac{1}{6} & \frac{1}{3} & 0 \\ 1 & \frac{1}{6} & \frac{5}{6} & 0 \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array}$$

eines impliziten Runge-Kutta-Verfahrens zurück gibt. Der Rückgabewert `bt` soll also vom Typ der oben beschriebenen Struktur mit den Komponenten `bt.m`, `bt.alpha`, `bt.beta` und `bt.gamma` sein.

c) Testen Sie Ihre Matlabfunktionen mit der Anfangswertaufgabe

$$y'(t) = -2ty(t)^2, \quad y(0) = 1,$$

deren exakte Lösung durch $y(t) = \frac{1}{t^2+1}$ gegeben ist.

Wählen Sie `anzFixpktIt` $= 1, 2, 3, 4, 5$ und berechnen Sie jeweils mit den Schrittweiten $h := \frac{1}{N}$, $N = 10, 20, 40, 80, 160, 320, 640$ die Fehler $e_N = |y_N - y(1)|$ an der Stelle $t = 1$. Plotten Sie jeweils den Fehler e_N über N in einem doppelt logarithmischen Plot.

Was beobachten Sie?

d) Berechnen Sie auch die Konvergenzordnungen

$$p_N = \frac{\ln\left(\frac{e_{N/2}}{e_N}\right)}{\ln(2)} \quad (N = 20, 40, 80, 160, 320, 640),$$

die sich aus Ihren numerischen Werten ergeben.

e) Testen Sie Ihre Routinen auch für das Anfangswertproblem

$$\begin{array}{rcrcrcrl} y_1'(t) & = & & & y_2(t) & - & y_3(t), & \qquad y_1(0) = 1, \\ y_2'(t) & = & -2y_1(t) & + & 3y_2(t) & - & y_3(t), & \qquad y_2(0) = -1, \\ y_3'(t) & = & -y_1(t) & + & y_2(t) & + & y_3(t), & \qquad y_3(0) = 2, \end{array}$$

dessen exakte Lösung durch

$$y(t) = e^t \begin{pmatrix} 1 - 4t \\ 1 - 4t - 2e^t \\ 4 - 2e^t \end{pmatrix}$$

gegeben ist.

Hinweise:
Die Programmieraufgaben sind in Matlab zu erstellen. Senden Sie alle Matlab-Files in einer E-mail mit dem Betreff **Loesung-Blatt10** an **angewandte.numerik@uni-ulm.de** (Abgabetermin jeweils wie beim Theorieteil). Drucken Sie zusätzlich allen Programmcode sowie die Ergebnisse aus und geben Sie diese vor der Übung ab. Der Source Code sollte strukturiert und, wenn nötig, dokumentiert sein.