Universität Basel

Prof. Dr. H. Harbrecht
M. Schmidlin

Numerik der Differentialgleichungen Herbstsemester 2020

Programmierblatt 1.

Besprechungswoche: **19–23.10.2020**

Auf diesem Programmierblatt betrachten wir die Lösung von gewöhnlichen Differentialgleichungen mit Hilfe von Einschrittverfahren. Vorgelegt sei dazu jeweils die gewöhnliche Differentialgleichung

$$\mathbf{y}'(x) = \mathbf{f}\big(x, \mathbf{y}(x)\big), \quad x \in [a, b] \tag{1}$$

mit $\mathbf{f}\colon [a, b] \times \mathbb{R}^d \to \mathbb{R}^d$ und dem Anfangswert $\mathbf{y}(a) = \mathbf{y}_a \in \mathbb{R}^d$.

θ-Schema

Kombinieren wir für $\theta \in [0, 1]$ den rechtsseitigen Differenzenquotienten mit dem linksseitigen mit einer Gewichtung von $1 - \theta$ respektive θ, so erhalten wir

$$\frac{\mathbf{y}(x+h) - \mathbf{y}(x)}{h} \approx (1-\theta)\mathbf{y}'(x) + \theta\mathbf{y}'(x+h) = (1-\theta)\mathbf{f}\big(x, \mathbf{y}(x)\big) + \theta\mathbf{f}\big(x+h, \mathbf{y}(x+h)\big).$$

Für eine Schrittweite $h > 0$ setzen wir $x_i := a + ih$ für $i = 0, 1, \ldots$ und $\boldsymbol{\eta}_0 := \mathbf{y}(a) = \mathbf{y}_a$. Wir können dann sukzessive Approximationen $\boldsymbol{\eta}_i$ von $\mathbf{y}(x_i)$ vermittels der Gleichung

$$\boldsymbol{\eta}_{i+1} = \boldsymbol{\eta}_i + h\big[(1-\theta)\mathbf{f}(x_i, \boldsymbol{\eta}_i) + \theta\mathbf{f}(x_{i+1}, \boldsymbol{\eta}_{i+1})\big] \tag{2}$$

bestimmen. Diese ist für $\theta \neq 0$ allerdings im allgemeinen nichtlinear und muss in jedem Schritt mit einer numerischen Methode gelöst werden. Für dieses Blatt wollen wir dazu das Newton-Verfahren mit einer Schrittweitensteuerung anwenden:

Algorithmus Newton-Verfahren mit Schrittweitensteuerung

Input: $\mathbf{g}\colon \mathbb{R}^d \to \mathbb{R}^d$, $\mathbf{g}'\colon \mathbb{R}^d \to \mathbb{R}^{d\times d}$, $\mathbf{z}_0 \in \mathbb{R}^d$, $\sigma \in (0, 1)$, $\texttt{tol} \ll 1$, $\texttt{maxiter} \in \mathbb{N}$

Output: $\mathbf{z}$ Approximation der Lösung von $\mathbf{g}(\mathbf{z}) \overset{!}{=} \mathbf{0}$

```
setze z := z_0
setze iter := 0
while iter < maxiter do
    löse Gleichungssystem g'(z)d = g(z)
    if ||d|| <= tol then
        breche ab
    else
        setze α := 1
        while ||g(z − αd)|| > ||g(z)||(1 − σα) do
            setze α := α/2
        end while
        setze z := z − αd
        setze iter := iter + 1
    end if
end while
```

Wir schreiben dazu Gleichung (2) um als

$$\mathbf{g}_i(\boldsymbol{\eta}_{i+1}) \overset{!}{=} \mathbf{0} \quad \text{mit} \quad \mathbf{g}_i(\mathbf{z}) := \mathbf{z} - \boldsymbol{\eta}_i - h\big[(1-\theta)\mathbf{f}(x_i, \boldsymbol{\eta}_i) + \theta\mathbf{f}(x_{i+1}, \mathbf{z})\big]$$

und bemerken, dass die Ableitung von $\mathbf{g}_i$ durch

$$\mathbf{g}_i'(\mathbf{z}) = \mathbf{I} - h\theta \mathbf{f}_{\mathbf{y}}(x_{i+1}, \mathbf{z}),$$

gegeben ist, wobei $\mathbf{I}$ die Einheitsmatrix bezeichnet. Als Startnäherung $\mathbf{z}_0$ für das Newton-Verfahren können wir dabei jeweils das $\boldsymbol{\eta}_i$ des vorhergenden Zeitschritts benutzen.

Aufgabe 1 (Newton-Verfahren mit Schrittweitensteuerung)**.**

Schreiben Sie eine Funktion

```
function z = newtonIterationSWS(g, gp, z0, sigma, tol, maxiter),
```

welche das nichtlineare Gleichungssystem $\mathbf{g}(\mathbf{z}) = \mathbf{0}$ für $\mathbf{g}\colon \mathbb{R}^d \to \mathbb{R}^d$ mit Hilfe des Newton-Verfahrens mit Schrittweitensteuerung löst. Die Funktionen $\mathbf{g}$ und $\mathbf{g}'$ sollen dabei als *Function Handles* `g` und `gp` übergeben werden.

Hinweis. *Testen Sie Ihre Funktion selbstständig mit einfachen, nichtlinearen Gleichungen, wie z.B. Nullstellen von (multivariaten) Polynomen oder trigonometrischen Funktionen.*

Aufgabe 2 (θ-Schema)**.**

Schreiben Sie eine Funktion

```
function [x, y] = thetaSchema(f, fy, a, b, ya, n, theta, ...
                              sigma, tol, maxiter),
```

welche das Anfangswertproblem (1) mit Hilfe des θ-Schemas löst. Dabei soll die Schrittweite als $h = (b-a)/n$ gewählt werden. Die Funktionen $\mathbf{f}$ und $\mathbf{f}_{\mathbf{y}}$ sollen dabei als *Function Handles* `f` und `fy` übergeben werden. Die Funktions-Outputs sollen der Form $\mathtt{x} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{\mathrm{n}} \end{bmatrix}$ und $\mathtt{y} = \begin{bmatrix} \boldsymbol{\eta}_0 & \boldsymbol{\eta}_1 & \cdots & \boldsymbol{\eta}_{\mathrm{n}} \end{bmatrix}$ sein.

Explizite Runge-Kutta-Verfahren

Als weitere Klasse von Verfahren betrachten wir explizite Runge-Kutta-Verfahren. Wählen wir eine Schrittweite $h > 0$ und setzen $x_i := a + ih$ für $i = 0, 1, \ldots$ sowie $\boldsymbol{\eta}_0 := \mathbf{y}(a) = \mathbf{y}_a$, dann sind die sukzessiven Approximationen $\boldsymbol{\eta}_i$ von $\mathbf{y}(x_i)$ durch

$$\begin{aligned} \mathbf{k}_j^{(i)} &= \mathbf{f}\left(x_i + h\alpha_j, \boldsymbol{\eta}_i + h\sum_{\ell=1}^{j-1} \beta_{j,\ell}\mathbf{k}_\ell^{(i)}\right), \quad j = 1, \ldots, m \\ \boldsymbol{\eta}_{i+1} &= \boldsymbol{\eta}_i + h\sum_{j=1}^{m} \gamma_j \mathbf{k}_j^{(i)} \end{aligned} \tag{3}$$

gegeben (siehe Kapitel 2.4 der Vorlesung). Dabei sind die Gewichte im *Butcher-Tableau*

$$\begin{array}{c|cccc} \alpha_1 & 0 & 0 & \ldots & 0 \\ \alpha_2 & \beta_{2,1} & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \alpha_m & \beta_{m,1} & \ldots & \beta_{m,m-1} & 0 \\ \hline & \gamma_1 & \gamma_2 & \cdots & \gamma_m \end{array}$$

zusammengefasst. Die Form dieser Verfahren erlaubt es, anstatt für jedes, durch ein Butcher-Tableau definierte Verfahren, eine Implementierung zu schreiben, eine Implementierung zu schreiben, welche das Butcher-Tableau als einen Input nimmt. Des Weiteren kann man dann für die expliziten Runge-Kutta-Verfahren die Schrittweitensteuerung mit eingebettetem Kontrollverfahren (siehe Kapitel 2.5) umsetzen.

Aufgabe 3 (explizite Runge-Kutta-Verfahren).

Schreiben Sie eine Funktion

```
function [x, y] = explizitRK(f, a, b, ya, n, alpha, beta, gamma),
```

welche das Anfangswertproblem (1) mit Hilfe eines expliziten Runge-Kutta-Verfahrens löst. Dabei soll das Butcher-Tableau als Spaltenvektoren `alpha` und `gamma` sowie Matrix `beta` übergeben werden. Die Funktion $\mathbf{f}$ soll dabei als *Function Handle* `f` übergeben werden. Die Funktions-Outputs sollen der Form $\mathtt{x} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{\mathtt{n}} \end{bmatrix}$ und $\mathtt{y} = \begin{bmatrix} \boldsymbol{\eta}_0 & \boldsymbol{\eta}_1 & \cdots & \boldsymbol{\eta}_{\mathtt{n}} \end{bmatrix}$ sein.

Aufgabe 4 (schrittweitengesteuerte, explizite Runge-Kutta-Verfahren).

Schreiben Sie eine Funktion

```
function [x, y] = explizitRKadapt(f, a, b, ya, h0, alpha, beta, ...
                                  gamma, gammah, p, tau, epsilon),
```

welche das Anfangswertproblem (1) mit Hilfe eines expliziten Runge-Kutta-Verfahrens mit einer Schrittweitensteuerung durch ein eingebettetes Runge-Kutta-Verfahren löst. Dabei wird das erweiterte Butcher-Tableau als Spaltenvektoren `alpha`, `gamma` und `gammah` sowie Matrix `beta` übergeben, wobei das Verfahren mit den γ-Werten `gamma` die Ordnung `p` und das Kontrollverfahren mit den $\widehat{\gamma}$-Werten `gammah` die Ordnung $\mathtt{p}+1$ haben sollen. Die anderen In- und Outputs sind analog zu jenen der Funktion `explizitRK`.

Hinweis. *Es ist hilfreich, die Funktion* `explizitRK` *von Aufgabe 3 als Startpunkt für die Implementation zu benutzen.*

Numerische Beispiele

Mit diesen Verfahren zuhanden betrachten wir nun einige Beispiele aus der Vorlesung und aus naturwissenschaftlichen oder technischen Anwendungen.

Aufgabe 5 (Beispiel I: Testproblem).

Vorgelegt sei das Anfangswertproblem $y' = \lambda y$ für $x \in [1, 3]$ mit $y(1) = 1$.

(a) Benutzen Sie Ihre Funktionen `thetaSchema` um das Problem numerisch mit $n = 2^3, 2^4, \ldots, 2^{13}$ und $\theta = 0, 0.49, 0.499, 0.4999, 0.5, 0.5001, 0.501, 0.51, 1$ zu lösen. Berechnen Sie zu jedem `n` für jedes θ den Absolutbetrag des Fehlers zwischen der analytischen Lösung und der jeweiligen Approximation an der Stelle $x = 3$ und plotten Sie diese Fehler zu jedem θ gegen n in einen `loglog`-Plot. Plotten Sie zum Vergleich ebenso die Raten n^{-1} und n^{-2} in den gleichen Plot. Wählen Sie dafür einmal $\lambda = 1$ und $\lambda = -1$ und benutzen Sie dabei `sigma = 0.5`, `tol = 1e-12` und `maxiter = 42`.

(b) Wiederholen Sie die Teilaufgabe (a), wobei sie stattdessen Ihre Funktionen `explixitRK` benutzen, um die Lösung numerisch mit dem expliziten Euler-Verfahren, dem Verfahren von Heun und dem klassischen Runge-Kutta-Verfahren zu berechnen. Plotten Sie hier zum Vergleich die Raten n^{-1}, n^{-2} und n^{-4}.

(c) Wiederholen Sie die Teilaufgabe (a), wobei sie stattdessen Ihre Funktionen `explixitRKadapt` benutzen, um die Lösung numerisch mit dem Runge-Kutta-Fehlberg-Verfahren RKF2(3) und dem Dormand-Prince-Verfahren DOPRI zu berechnen. Sie benutzen hierfür $\varepsilon = 2^{-1}, 2^{-2}, \ldots, 2^{-20}$, $h_0 = 0.5$ und $\tau = 0.9$. Plotten Sie hier den Fehler gegen ε anstelle gegen n und, zum Vergleich, die Raten ε, $\varepsilon^{2/3}$ und $\varepsilon^{4/5}$.

Aufgabe 6 (Beispiel II: Lotka-Volterra-Modell).

Benutzen Sie Ihre Funktionen, um das Räuber-Beute-Modell aus Beispiel 1.2 aus der Vorlesung, welches durch

$$\mathbf{y}' = \mathbf{f}(x, \mathbf{y}) = \begin{bmatrix} \alpha y_1(1 - y_2) \\ y_2(y_1 - 1) \end{bmatrix}, \quad \alpha > 0,$$

gegeben ist, für $\alpha = 10$ und $a = 0$, $b = 5$ und $\mathbf{y}(a) = [3, 1]^\mathsf{T}$ approximativ zu lösen. Plotten Sie dabei jeweils y_1 und y_2 gegen x und markieren Sie ebenfalls die vom Verfahren benutzten Stellen $(x_i, 0)$.

- Für `thetaSchema` benutzen Sie $\theta = 0.5001$ und $n = 2^6$ mit `sigma = 0.5`, `tol = 1e-12` und `maxiter = 42`.
- Für `explixitRK` benutzen Sie das klassische Runge-Kutta-Verfahren ebenfalls mit $n = 2^6$.
- Für `explixitRKadapt` benutzen Sie das Dormand-Prince-Verfahren mit $\varepsilon = 10^{-5}$, $h_0 = 1$ und $\tau = 0.9$.

Aufgabe 7 (Beispiel III: Van-der-Pol-Oszillator).

Ein Van-der-Pol-Oszillator wird durch die gewönliche Differentialgleichung

$$\begin{bmatrix} v' \\ u' \end{bmatrix} = \begin{bmatrix} \epsilon^{-1}(u - v^3/3 + v) \\ -v \end{bmatrix}$$

beschrieben. Benutzen Sie Ihre Funktionen, um diese für $\epsilon = 0.03$ und $a = 0$, $b = 10$ und $[v(0), u(0)]^\mathsf{T} = [0, 2]^\mathsf{T}$ approximativ zu lösen. Plotten Sie dabei jeweils v und u gegen x.

- Für `thetaSchema` benutzen Sie $\theta = 0.6$ und $n = 2^8$ mit `sigma = 0.5`, `tol = 1e-12` und `maxiter = 42`.
- Für `explixitRK` benutzen Sie das klassische Runge-Kutta-Verfahren ebenfalls mit $n = 2^8$.
- Für `explixitRKadapt` benutzen Sie das Dormand-Prince-Verfahren mit $\varepsilon = 10^{-6}$, $h_0 = 0.1$ und $\tau = 0.9$.

Was fällt auf?

Aufgabe 8 (Beispiel IV: SEIRS-Modell).

Eine Variante eines SEIRS-Modell aus der mathematischen Epidemiologie ist durch die gewöhnliche Differentialgleichung

$$\begin{aligned} R' &= r_R I - r_S R, \\ S' &= r_S R - r_{E,E} SE - r_{E,I} SI, \\ E' &= r_{E,E} SE + r_{E,I} SI - r_I E, \\ I' &= r_I E - r_R I, \end{aligned}$$

gegeben. Benutzen Sie Ihre Funktionen, um diese für $a = 0$, $b = 12$ und $[R(0), S(0), E(0), I(0)]^\mathsf{T} = [0, 0.999999, 0.000001, 0]^\mathsf{T}$ approximativ zu lösen. Plotten Sie dabei jeweils R, S, E und I gegen x als MATLAB *area plot*, `area(x.', y.')`. Wählen Sie dafür die gleichen Parameter wie in Aufgabe 7 und betrachten Sie folgende drei Situationen:

Fall	r_R	r_S	$r_{E,E}$	$r_{E,I}$	r_I
0	2	0.3	4	6	3
1	2	0.3	4	6 für $x < 2$, 0.5 für $x \geq 2$	3
2	2	0.3	4 für $x < 3$, 2 für $x \geq 3$	6 für $x < 2$, 0.5 für $x \geq 2$	3