Aufgaben zur Vorlesung
Numerik II
Wintersemester 2012/13
Übungsblatt 7

W.-J. Beyn
D. Otten

Abgabe: Mittwoch, 28.11.2012, vor Beginn der Übung

Übung: Mi. 12:15–13:45, V5-148

Aufgabe 19 [Autonomisierung]

Wir betrachten die nicht-autonome Differentialgleichung

$$u' = f(t, u), \quad u(t_0) = u^0 \tag{1}$$

mit $f \in C(J \times \mathbb{R}^n, \mathbb{R}^n)$. Indem man die Zeit als künstliche Zustandsvariable auffasst, kann das Anfangswertproblem (1) autonomisiert werden, d.h. äquivalent umgeschrieben werden in

$$\begin{pmatrix} u \\ t \end{pmatrix}' = \begin{pmatrix} f(t,u) \\ 1 \end{pmatrix}, \quad \begin{pmatrix} u(0) \\ t(0) \end{pmatrix} = \begin{pmatrix} u^0 \\ t_0 \end{pmatrix} \tag{2}$$

Zeigen Sie, dass ein explizites Runge-Kutta-Verfahren genau dann invariant unter Autonomisierung ist, wenn für die Koeffizienten gilt

$$\sum_{j=1}^{m} \gamma_j = 1 \quad \text{und} \quad \sum_{j=1}^{m} \beta_{ij} = \alpha_i \quad \forall i = 1, \ldots, m.$$

Hinweis: Invariant unter Autonomisierung bedeutet, dass die numerische Lösung für u von (2) mit der von (1) übereinstimmt. Verwenden Sie $s \geqslant 0$ als unabhängige Variable in (2).

(6 Punkte)

Aufgabe 20: [Stabilität des impliziten Euler-Verfahrens]

Gegeben sei die Anfangswertaufgabe

$$u' = f(t, u), \, t \in J = [t_0, t_E], \quad u(t_0) = u^0 \in \mathbb{R}^n,$$

wobei $f \in C(J \times \mathbb{R}^n, \mathbb{R}^n)$ einer einseitigen Lipschitzbedingung genüge, vergleiche Aufgabe 5, d.h. es gebe ein $\alpha \in \mathbb{R}$ derart, dass

$$\langle f(t,v) - f(t,w), v - w \rangle \leqslant \alpha \, \|v - w\|_2^2 \quad \forall t \in J \, \forall v, w \in \mathbb{R}^n.$$

Man zeige, dass das implizite Euler-Verfahren

$$\frac{1}{h_j}\left(u^{j+1} - u^j\right) = f(t_{j+1}, u^{j+1}), \quad j = 0, \ldots, M-1, \quad h_{\max} = \max_{j=0,\ldots,M-1} h_j$$

unter der Voraussetzung $\alpha h_{\max} \leqslant q < 1$ mit $q \in]0,1[$ stabil ist bezüglich der Norm

$$\|u\|_{2,\infty} = \max_{j=0,\ldots,M} \|u(t_j)\|_2 \, , \; u \in (\mathbb{R}^n)^{\Omega_h} .$$

Hinweis: Es darf ohne Beweis angenommen werden, dass das Gleichungssystem

$$v - u = hf(t, v)$$

für alle $u \in \mathbb{R}^n$, $t \in J$ und $h \geqslant 0$ mit $\alpha h < 1$ genau eine Lösung $v = v(t, u, h)$ besitzt. Weiter orientiere man sich eng am Stabilitätsbeweis der Vorlesung, Satz 2.13, und verwende die einseitige Lipschitzbedingung anstelle der beidseitigen.

(6 Punkte)

Aufgabe 21: [Implizite Runge-Kutta-Verfahren]

Gegeben sei die Anfangswertaufgabe

$$u' = f(t, u), \quad u(t_0) = u^0.$$

Schreiben Sie ein Programm, das folgendes leistet:
Nach Eingabe der Anfangsdaten u^0, der rechten Seite f sowie deren Ableitung Df, des Start- und Endzeitpunktes t_0 und t_E, der Schrittweite h und eines m-stufigen Runge-Kutta-Tableaus

$$\begin{array}{c|ccc} \alpha_1 & \beta_{11} & \cdots & \beta_{1m} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_m & \beta_{m1} & \cdots & \beta_{m,m} \\ \hline & \gamma_1 & \cdots & \gamma_m \end{array}$$

soll das Programm das zugehörige (implizite) Runge-Kutta-Verfahren durchführen. Lösen Sie die impliziten Gleichungen im j-ten Schritt mit dem (mehrdimensionalen) Newton-Verfahren. Starten Sie dabei mit $k_i^0 = f(t_j, u^j)$ für $i = 1, \ldots, m$ und beenden Sie die Iteration, sobald

$$\max_{i=1,\ldots,m} \left\| k_i^{\nu+1} - k_i^{\nu} \right\|_\infty \leqslant 10^{-7}$$

erfüllt ist. Brechen Sie die Rechnung ab, falls das Newton-Verfahren nicht konvergiert (zu viele Iterationsschritte oder Overflow).
Verwenden Sie Ihr Programm zur numerischen Lösung der aus Aufgabe 14 bekannten Beispiele a) und b) auf dem Intervall $[0, 7]$ mit den angegebenen impliziten Runge-Kutta-Verfahren und der Schrittweite $h = 0.1$. Zeichnen Sie für a) und b) jeweils ein aussagekräftiges Diagramm, das die exakte Lösung und alle numerischen Lösungen enthält!
Runge-Kutta-Tableaus:

- $m = 1$ (mit Konsistenzordnung 2)

$$\begin{array}{c|c} \frac{1}{2} & \frac{1}{2} \\ \hline & 1 \end{array}$$

- $m = 2$ (mit Konsistenzordnung 4)

$$\begin{array}{c|cc} \frac{1}{6}\left(3 - \sqrt{3}\right) & \frac{1}{4} & \frac{1}{4} - \frac{\sqrt{3}}{6} \\ \frac{1}{6}\left(3 + \sqrt{3}\right) & \frac{1}{4} + \frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

(6 Punkte)

Numerik II (WS 12/13)
Übungsblatt 07
Lösungen

Aufgabe 19: $f \in C(J \times \mathbb{R}^n, \mathbb{R}^n)$, $J = [t_0, t_E]$

- nicht-autonome Differenzialgleichung

$$(19.1) \qquad u'(t) = f(t, u(t)) \qquad , \quad u(t_0) = u^0$$

- autonome Differenzialgleichung

$$(19.2) \qquad \begin{pmatrix} \tilde{u} \\ \tilde{t} \end{pmatrix}'(s) = \begin{pmatrix} f(\tilde{t}(s), \tilde{u}(s)) \\ 1 \end{pmatrix} =: \tilde{f}\begin{pmatrix} \tilde{u}(s) \\ \tilde{t}(s) \end{pmatrix}, \qquad \begin{pmatrix} \tilde{u}(0) \\ \tilde{t}(0) \end{pmatrix} = \begin{pmatrix} u^0 \\ t_0 \end{pmatrix}$$

Zeige: Ein explizites RK-Verfahren ist genau dann invariant unter Autonomisierung, wenn für die Koeffizienten gilt

$$(19.3) \qquad \sum_{i=1}^{m} \gamma_i = 1 \qquad \text{und} \quad \sum_{l=1}^{m} \beta_{il} = \alpha_i \quad \forall i = 1, \dots, m$$

Lösung: • Autonomisierung einer DGL (siehe Beiblatt).

- Vorbemerkung:

zu (19.1):

$$u^{j+1} = u^j + h_j \sum_{i=1}^{m} \gamma_i K_i(t_j, u^j, h_j)$$

$$K_i(t_j, u^j, h_j) = f\left(t_j + \alpha_i h_j,\; u^j + h_j \sum_{l=1}^{i-1} \beta_{il} K_l(t_j, u^j, h_j)\right), \quad i = 1, \dots, m$$

zu (19.2):

$$\begin{pmatrix} \tilde{u}^{j+1} \\ \tilde{t}^{j+1} \end{pmatrix} = \begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix} + h_j \sum_{i=1}^{m} \gamma_i \tilde{K}_i\left(s_j, \begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix}, h_j\right)$$

$$\tilde{K}_i\left(s_j, \begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix}, h_j\right) = \tilde{f}\left(\begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix} + h_j \sum_{l=1}^{i-1} \beta_{il} \tilde{K}_l\left(s_j, \begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix}, h_j\right)\right), \quad i = 1, \dots, m$$

- "$\Leftarrow$": Seien die Bedingungen aus (19.3) erfüllt. Wir zeigen per Induktion über j:

$$(19.4) \qquad \boxed{\tilde{u}^j = u^j \quad \text{und} \quad \tilde{t}^j = t_j \quad \forall j = 0, 1, 2, \dots}$$

IA: $j = 0$: $\tilde{u}^0 = \tilde{u}(0) = u^0$, $\tilde{t}^0 = \tilde{t}(0) = t_0$ ✓

IS: $j \to j+1$:

$$\begin{pmatrix} \tilde{u}^{j+1} \\ \tilde{t}^{j+1} \end{pmatrix} = \begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix} + h_j \sum_{i=1}^{m} \gamma_i \underbrace{\tilde{K}_i\left(s_j, \begin{pmatrix} \tilde{u}^j \\ \tilde{t}^j \end{pmatrix}, h_j\right)}_{\overset{!}{\underset{\text{zz (mit IV)}}{=}} \begin{pmatrix} K_i(t_j, u^j, h_j) \\ 1 \end{pmatrix} \; \forall i = 1, \dots, m} = \begin{pmatrix} \tilde{u}^j + h_j \sum_{i=1}^{m} \gamma_i K_i(t_j, u^j, h_j) \\ \tilde{t}^j + h_j \sum_{i=1}^{m} \gamma_i \end{pmatrix}$$

$$(19.5) \qquad \underset{\substack{\text{IV} \\ \sum_{i=1}^{m} \gamma_i = 1 \\ \text{vgl. (19.3)}}}{=} \begin{pmatrix} u^j + h_j \sum_{i=1}^{m} \gamma_i K_i(t_j, u^j, h_j) \\ t_j + h_j \end{pmatrix} = \begin{pmatrix} u^{j+1} \\ t_{j+1} \end{pmatrix}.$$

Wir zeigen ~~über~~ per Induktion über i:

$$\boxed{\tilde K_i\left(s_j, \begin{pmatrix}\tilde u^j\\ \tilde t^j\end{pmatrix}, h_j\right) = \begin{pmatrix} K_i(t_j, u^j, h_j) \\ 1\end{pmatrix}, \quad i=1,\dots,m}$$

IA: $i=1$:

$$\tilde K_1\left(s_j,\begin{pmatrix}\tilde u^j\\ \tilde t^j\end{pmatrix},h_j\right) = \tilde f\begin{pmatrix}\tilde u^j\\ \tilde t^j\end{pmatrix} = \begin{pmatrix} f(\tilde t^j,\tilde u^j)\\ 1\end{pmatrix} \underset{\substack{\text{IV}\\ (\text{bzgl. } j)}}{=} \begin{pmatrix} f(t_j,u^j)\\ 1\end{pmatrix}$$

$$\underset{\substack{\sum_{l=1}^{0}\beta_{il} = \alpha_1 = 0 \\ \text{vgl. } (19.3)}}{=} \begin{pmatrix} K_1(t_j,u^j,h_j)\\ 1\end{pmatrix}.$$

IS: $i-1 \to i$:

$$\tilde K_i\left(s_j,\begin{pmatrix}\tilde u^j\\ \tilde t^j\end{pmatrix},h_j\right) = \tilde f\left(\begin{pmatrix}\tilde u^j\\ \tilde t^j\end{pmatrix} + h_j\sum_{l=1}^{i-1}\beta_{il}\tilde K_l\left(s_j,\begin{pmatrix}u^j\\ t^j\end{pmatrix},h_j\right)\right)$$

$$\underset{\substack{\text{IV}\\ (\text{bzgl. } i)}}{=} \tilde f\left(\begin{pmatrix}\tilde u^j + h_j\sum_{l=1}^{i-1}\beta_{il}K_l(t_j,u^j,h_j)\\ \tilde t^j + h_j\sum_{l=1}^{i-1}\beta_{il}\end{pmatrix}\right)$$

$$= \begin{pmatrix} f\left(\tilde t^j + h_j\sum_{l=1}^{i-1}\beta_{il},\ \tilde u^j + h_j\sum_{l=1}^{i-1}\beta_{il}K_l(t_j,u^j,h_j)\right)\\ 1\end{pmatrix}$$

$$\underset{\substack{\text{IV}\\ (\text{bzgl. } j)}}{=} \begin{pmatrix} f\left(t_j + h_j\sum_{l=1}^{i-1}\beta_{il},\ u^j + h_j\sum_{l=1}^{i-1}\beta_{il}K_l(t_j,u^j,h_j)\right)\\ 1\end{pmatrix}$$

(19.6)
$$\underset{\substack{\alpha_i = \sum_{l=1}^{i-1}\beta_{il},\ i=1,\dots,m\\ \text{vgl. } (19.3)}}{=} \begin{pmatrix} f\left(t_j + h_j\alpha_i,\ u^j + h_j\sum_{l=1}^{i-1}\beta_{il}K_l(t_j,u^j,h_j)\right)\\ 1\end{pmatrix}$$

$$= \begin{pmatrix} K_i(t_j,u^j,h_j)\\ 1\end{pmatrix}.$$

• $\Longrightarrow$: Es gelte (19.4).

1. Angenommen $\sum_{i=1}^{m}\gamma_i \neq 1$, dann gilt (vgl. (19.5))

$$t_{j+1} := t_j + h_j \underset{(19.4)}{=} \tilde t^j + h_j \underset{\sum_{i=1}^{m}\gamma_i \neq 1}{\neq} \tilde t^j + h_j\sum_{i=1}^{m}\gamma_i = \tilde t^{j+1} \quad ↯ \text{ zu } (19.4)$$

$$\Rightarrow \sum_{i=1}^{m}\gamma_i = 1$$

2. Angenommen $\sum_{l=1}^{i-1}\beta_{il} \neq \alpha_i$ für ein $i \in \{1,\dots,m\}$. Da (19.4) für beliebige f's gilt (unabhängig von der Wahl von f), können wir $g(t)$ anstelle von $f(t,u)$ betrachten. In diesem Fall ist (19.4) nach unserer Annahme nicht erfüllt (da in (19.6) keine Gleichheit vorliegt) ↯ zu (19.4).

$$\Rightarrow \sum_{l=1}^{i-1}\beta_{il} = \alpha_i \quad \forall i=1,\dots,m$$

Autonomisierung einer DGL:

- Nicht-autonome DGL

$$(1)\quad \left\{\begin{array}{l} u'(t) = f(t, u(t)) \\ u(t_0) = u^0 \end{array}\right\}$$

hier: $u'(t) = \frac{d}{dt} u(t)$

- Autonomisierung:

$\tilde{t}(s) := s + t_0$, dann $\tilde{t}(0) = t_0$ und $\frac{d}{ds}\tilde{t}(s) = 1$

$\tilde{u}(s) := u(s+t_0) = u(\tilde{t}(s))$, dann $\tilde{u}(0) = u(t_0) = u^0$ und

$$\frac{d}{ds}\tilde{u}(s) = \frac{d}{ds} u(s+t_0) = \left(\frac{d}{dt}u\right)(s+t_0)\cdot 1 \underset{(1)}{=} f(\underbrace{s+t_0}_{\tilde{t}(s)}, \underbrace{u(s+t_0)}_{\tilde{u}(s)})$$

Wir erhalten

Kein zusätzliches s Argument mehr vorhanden.

$$\begin{pmatrix} f(\tilde{t}(s), \tilde{u}(s)) \\ 1 \end{pmatrix} =: \tilde{f}\left(\begin{pmatrix} \tilde{u}(s) \\ \tilde{t}(s) \end{pmatrix}\right)$$

- Autonome DGL:

$$(2)\quad \left\{\begin{array}{l} \begin{pmatrix} \tilde{u}'(s) \\ t'(s) \end{pmatrix} = \tilde{f}\left(\begin{pmatrix} \tilde{u}(s) \\ \tilde{t}(s) \end{pmatrix}\right) \\ \begin{pmatrix} \tilde{u}(0) \\ t(0) \end{pmatrix} = \begin{pmatrix} u^0 \\ t_0 \end{pmatrix} \end{array}\right\}$$

hier: $\begin{pmatrix} \tilde{u}'(s) \\ \tilde{t}'(s) \end{pmatrix} = \frac{d}{ds}\begin{pmatrix} \tilde{u}(s) \\ \tilde{t}(s) \end{pmatrix}$

Bemerkung:

- Die Lösungen autonomer DGL'en sind invariant unter Zeittranslation, d.h.

$x = x(t)$ Lösung von $\frac{d}{dt}x(t) = f(x(t)) \Rightarrow z(t) := x(t+c)$ Lösung von $\frac{d}{dt}z(t) = f(z(t))$

Aufgabe 20:

Gegeben sei die AWA

$$u' = f(t,u) \quad , t \in J = [t_0, t_E] ,$$
$$u(t_0) = u^0 \in \mathbb{R}^n,$$

wobei

- $f \in C(J \times \mathbb{R}^n, \mathbb{R}^n)$,
- f genüge Lipschitzbedingung im 2. Argument der Form
$$\exists \alpha \in \mathbb{R} : \langle f(t,v) - f(t,w), v-w \rangle \leq \alpha \|v-w\|_2^2 \quad \forall t \in J \ \forall v,w \in \mathbb{R}^n.$$

Zeige: Das implizite Euler-Verfahren $\left(m=1, \begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}\right)$

$$\frac{1}{h_j}(u^{j+1} - u^j) = f(t_{j+1}, u^{j+1}) \quad , j = 0, \dots, M-1 \quad , h_{max} := \max_{j=0,\dots,M-1} h_j$$

mit
$$\alpha h_{max} \leq q < 1 \quad , q \in]0,1[$$
ist stabil bzgl. der Norm
$$\|u\|_{2,\infty} := \max_{j=0,\dots,M} \|u(t_j)\|_2 \quad , u \in (\mathbb{R}^n)^{\Omega_h} =: X_h$$

d.h.
$$\exists C > 0 \ \exists \bar{h} > 0 : \|u_1 - u_2\|_{2,\infty} \leq C \cdot \|T_h(u_1) - T_h(u_2)\|_{2,\infty} \quad \forall u_1, u_2 \in X_h \quad \forall h \leq h_{max} \leq \bar{h}.$$

$\|.\|_{2,\infty}$ wird im Skript mit $\|.\|_\infty$ und $(\mathbb{R}^n)^{\Omega_h}$ mit X_h bezeichnet.

Lösung: (vgl. Beweis von Satz 2.13, globale Stabilität von Einschrittverfahren)

Definiere den Operator
$$T_h u := \left(u(t_0) - u^0 , \left(\frac{1}{h_j}(u^{j+1} - u^j) - f(t_{j+1}, u^{j+1}) \right)_{j=0,\dots,M-1} \right).$$

Stabilitätsungleichung für T_h:

Seien $u_1, u_2 \in X_h$, $v_1 := T_h(u_1)$, $v_2 := T_h(u_2)$, $\varsigma := \|v_1 - v_2\|_{2,\infty}$, $z := u_1 - u_2$

1. • Für $j = 0, \dots, M-1$ gilt:
$$\frac{1}{h_j}\left(z(t_{j+1}) - z(t_j) \right) = \left[\frac{1}{h_j}\left(u_1(t_{j+1}) - u_1(t_j)\right) - \overbrace{\varphi(t_j, u_1(t_j), h_j)}^{= f(t_{j+1}, u_1(t_{j+1}))} \right]$$
$$- \left[\frac{1}{h_j}\left(u_2(t_{j+1}) - u_2(t_j)\right) - \underbrace{\varphi(t_j, u_2(t_j), h_j)}_{= f(t_{j+1}, u_2(t_{j+1}))} \right]$$
$$+ \underbrace{\varphi(t_j, u_1(t_j), h_j)}_{= f(t_{j+1}, u_1(t_{j+1}))} - \underbrace{\varphi(t_j, u_2(t_j), h_j)}_{= f(t_{j+1}, u_2(t_{j+1}))}$$

$$\boxed{\|z(t_{j+1})\|_2 \leq \left(\frac{1}{1 - \alpha h_j} \right) \cdot \left[\|z(t_j)\|_2 + h_j \varsigma \right] \quad , j = 0, \dots, M-1}$$

Multiplikation mit h_j und Addition von $z(t_j)$ liefert

$$z(t_{j+1}) = z(t_j) + h_j \cdot \underbrace{\left[\frac{1}{h_j}\big(u_1(t_{j+1})-u_1(t_j)\big) - f(t_{j+1},u_1(t_{j+1}))\right]}_{=v_1(t_{j+1})},$$

$$- h_j \underbrace{\left[\frac{1}{h_j}\big(u_2(t_{j+1})-u_2(t_j)\big) - f(t_{j+1},u_2(t_{j+1}))\right]}_{=v_2(t_{j+1})},$$

$$+ h_j \big(f(t_{j+1},u_1(t_{j+1})) - f(t_{j+1},u_2(t_{j+1}))\big)$$

$$= z(t_j) + h_j \left[v_1(t_{j+1}) - v_2(t_{j+1}) + f(t_{j+1},u_1(t_{j+1})) - f(t_{j+1},u_2(t_{j+1}))\right]$$

Multiplikation (von rechts) mit $z(t_{j+1})$ liefert weiter

$$\|z(t_{j+1})\|_2^2 = \langle z(t_{j+1}), z(t_{j+1})\rangle$$

$$= \langle z(t_j), z(t_{j+1})\rangle + h_j \langle v_1(t_{j+1}) - v_2(t_{j+1}), z(t_{j+1})\rangle$$
$$+ h_j \cdot \langle f(t_{j+1},u_1(t_{j+1})) - f(t_{j+1},u_2(t_{j+1})), \underbrace{z(t_{j+1})}_{=u_1(t_{j+1})-u_2(t_{j+1})}\rangle$$

f Lipschitz stetig im 2. Argument

$$\leq \langle z(t_j), z(t_{j+1})\rangle + h_j \langle v_1(t_{j+1}) - v_2(t_{j+1}), z(t_{j+1})\rangle + \alpha h_j \|z(t_{j+1})\|_2^2$$

Subtraktion von $\alpha h_j \|z(t_{j+1})\|_2^2$ und Division durch $1-\alpha h_j > 0$ liefert nun

$$\|z(t_{j+1})\|_2^2 \leq \left(\frac{1}{1-\alpha h_j}\right) \cdot \left[\langle z(t_j), z(t_{j+1})\rangle + h_j \langle v_1(t_{j+1}) - v_2(t_{j+1}), z(t_{j+1})\rangle\right]$$

$$\overset{\text{Cauchy-Schwarz}}{\leq} \left(\frac{1}{1-\alpha h_j}\right) \cdot \left[\|z(t_j)\|_2 + h_j \|v_1(t_{j+1}) - v_2(t_{j+1})\|_2\right] \|z(t_{j+1})\|_2$$

Division durch $\|z(t_{j+1})\|_2$ liefert

$$\|z(t_{j+1})\|_2 \leq \left(\frac{1}{1-\alpha h_j}\right) \left[\|z(t_j)\|_2 + h_j \underbrace{\|v_1(t_{j+1}) - v_2(t_{j+1})\|_2}_{\substack{\leq \max_{j=0,\dots,M-1} \|v_1(t_{j+1}) - v_2(t_{j+1})\|_2 \\ = \|v_1 - v_2\|_{2,\infty} =: \rho}}\right].$$

$$\leq \left(\frac{1}{1-\alpha h_j}\right) \left[\|z(t_j)\|_2 + h_j \rho\right].$$

2. Dies ist eine rekursive Abschätzung für $\|z(t_{j+1})\|_2$. Durch Induktion finden wir

$$\boxed{\|z(t_j)\|_2 \leq \prod_{\nu=0}^{j-1} \left(\frac{1}{1-\alpha h_\nu}\right) \cdot \|z(t_0)\|_2 + \sum_{i=0}^{j-1} h_i \rho \prod_{\nu=i+1}^{j-1} \left(\frac{1}{1-\alpha h_\nu}\right), \quad j=0,\dots,M.}$$

Induktionsanfang (IA): $j=0$. $\prod_\emptyset = 1$, $\sum_\emptyset = 0$.

$$\|z(t_0)\|_2 = \|u_1(t_0) - u_2(t_0)\|_2 = \|u^0 - u^0\|_2 = 0 \leq 0 = \prod_\emptyset(\dots) + \sum_\emptyset h_i \rho_i \prod_\emptyset(\dots). \checkmark$$

Induktionsschritt (IS): $j \to j+1$

$$\|z(t_{j+1})\|_2 \underset{\text{Teil 1}}{\leq} \left(\frac{1}{1-\alpha h_j}\right)\cdot\left[\|z(t_j)\|_2 + h_j\rho\right]$$

$$\underset{\text{IV}}{\leq} \left(\frac{1}{1-\alpha h_j}\right)\cdot\left[\prod_{\nu=0}^{j-1}\left(\frac{1}{1-\alpha h_\nu}\right)\|z(t_0)\|_2 + \sum_{i=0}^{j-1} h_i\rho\prod_{\nu=i+1}^{j-1}\left(\frac{1}{1-\alpha h_\nu}\right) + h_j\rho\right]$$

$$= \prod_{\nu=0}^{j}\left(\frac{1}{1-\alpha h_\nu}\right)\|z(t_0)\|_2 + \sum_{i=0}^{j} h_i\rho\prod_{\nu=i+1}^{j}\left(\frac{1}{1-\alpha h_\nu}\right) \checkmark$$

3. Aus

$$\|z(t_0)\|_2 = \|u_1(t_0)-u_2(t_0)\|_2 = \|(u_1(t_0)-u^0)-(u_2(t_0)-u^0)\|_2 = \|v_1(t_0)-v_2(t_0)\|_2 \leq \rho$$

und der Induktion erhalten wir

$$\|z(t_j)\|_2 \leq \left[\prod_{\nu=0}^{j-1}\left(\frac{1}{1-\alpha h_\nu}\right) + \sum_{i=0}^{j-1} h_i \prod_{\nu=i+1}^{j-1}\left(\frac{1}{1-\alpha h_\nu}\right)\right]\cdot\rho, \quad j=0,\dots,M$$

Weiter gilt nun

$$\frac{1}{1-\alpha h_\nu} = 1+\left(\frac{1}{1-\alpha h_\nu}-1\right) = 1+\frac{\alpha h_\nu}{1-\alpha h_\nu} \underset{1+x\leq e^x}{\leq} e^{\frac{\alpha h_\nu}{1-\alpha h_\nu}}$$

- für $\alpha \geq 0$ verwende nun (nach Voraussetzung gilt $h_\nu\alpha \leq h_{max}\alpha \leq q<1$):

$$h_\nu\alpha \leq q < 1 \Rightarrow -h_\nu\alpha \geq -q > -1 \Rightarrow 1-h_\nu\alpha \geq 1-q > 0$$

$$\Rightarrow 0 \overset{!}{<} \frac{1}{1-h_\nu\alpha} \leq \frac{1}{1-q} \quad \forall \alpha\in\mathbb{R}.$$

- für $\alpha\leq 0$ fordere zusätzlich $h_\nu|\alpha| < \frac{q}{1-q}$:

$$(1-q)(-\alpha)h_\nu \leq (1-q)|\alpha|h_\nu \leq q \Rightarrow -\alpha h_\nu \leq \frac{q}{1-q} \Rightarrow 1-\alpha h_\nu \leq \frac{1}{1-q}$$

$$\Rightarrow \frac{1}{1-h_\nu\alpha} \geq 1-q \underset{q\in]0,1[}{>} 0$$

Dies liefert

$$e^{\frac{\alpha h_\nu}{1-\alpha h_\nu}} \leq \begin{cases} e^{\frac{\alpha h_\nu}{1-q}}, & \alpha\geq 0 \;\&\; h_\nu\alpha\leq q<1 \\ e^{\alpha h_\nu(1-q)}, & \alpha \leq 0 \;\&\; h_\nu|\alpha| \leq \frac{q}{1-q}\end{cases} =: e^{\alpha h_\nu C_\alpha}$$

Daraus schließen wir jetzt: $\left(C_\alpha := \begin{cases}\frac{1}{1-q}, & \alpha\geq0 \;\&\; h_{max}\alpha\leq q<1\\ 1-q, & \alpha\leq 0 \;\&\; h_{max}|\alpha|\leq\frac{q}{1-q}\end{cases}\right)$

$$\|z(t_j)\|_2 \leq \left[\prod_{\nu=0}^{j-1} e^{\alpha h_\nu C_\alpha} + \sum_{i=0}^{j-1} h_i \prod_{\nu=i+1}^{j-1} e^{\alpha h_\nu C_\alpha}\right]\cdot\rho$$

$$=\left[e^{\alpha C_\alpha\sum_{\nu=0}^{j-1}h_\nu} + \sum_{i=0}^{j-1} h_i e^{\alpha C_\alpha \sum_{\nu=i+1}^{j-1} h_\nu}\right]\cdot\rho$$

$$=\left[e^{\alpha C_\alpha(t_j-t_0)} + \sum_{i=0}^{j-1} h_i e^{\alpha C_\alpha(t_j-t_{i+1})}\right]\cdot\rho$$

$$= e^{\alpha C_\alpha(t_j-t_0)}\cdot\left[1+\sum_{i=0}^{j-1} h_i \underbrace{e^{\alpha C_\alpha(t_0-t_{i+1})}}_{\leq \begin{cases}1, &\alpha\geq0\\ e^{\alpha C_\alpha(t_0-t_j)}, &\alpha\leq0\end{cases}}\right]\cdot\rho \quad \text{falls } j\geq1$$

$$\leq e^{\alpha C_\alpha (t_j - t_0)} \cdot \left[1 + \underbrace{\left(\sum_{i=0}^{j-1} h_i\right)}_{= t_j - t_0 \;\; (h_j := t_{j+1} - t_j)} \cdot \begin{cases} 1 & , \alpha \geq 0 \\ e^{\alpha C_\alpha (t_0 - t_j)} & , \alpha \leq 0 \end{cases} \right] \cdot \rho$$

$$\leq e^{\alpha C_\alpha (t_j - t_0)} \cdot \left[1 + (t_j - t_0) \cdot \begin{cases} 1 & , \alpha \geq 0 \\ e^{\alpha C_\alpha (t_0 - t_j)} & , \alpha \leq 0 \end{cases} \right] \cdot \rho \quad , j = 1, 2, \dots, M$$

Für $j = 0$ ~~haben~~ haben wir bereits $\|z(t_0)\|_2 \leq \rho$ oben eingesehen.

4. Nehmen wir nun das Maximum über $j = 0, \dots, M$, so erhalten wir

$$\|u_1 - u_2\|_{2,\infty} = \|z\|_{2,\infty} = \max_{j=0,\dots,M} \|z(t_j)\|_2$$

$$\leq \max_{j=0,\dots,M} e^{\alpha C_\alpha (t_j - t_0)} \left[1 + (t_j - t_0) \cdot \begin{cases} 1 & , \alpha \geq 0 \\ e^{\alpha C_\alpha (t_0 - t_j)} & , \alpha \leq 0 \end{cases} \right] \cdot \rho$$

$$\underset{t_E = t_M}{\leq} \begin{cases} e^{\alpha C_\alpha (t_E - t_0)} & , \alpha \geq 0 \\ 1 & , \alpha \leq 0 \end{cases} \cdot \left[1 + (t_E - t_0) \cdot \begin{cases} 1 & , \alpha \geq 0 \\ e^{\alpha C_\alpha (t_0 - t_E)} & , \alpha \leq 0 \end{cases} \right] \cdot \rho$$

$$= \begin{cases} e^{\alpha C_\alpha (t_E - t_0)} \cdot (1 + t_E - t_0) \cdot \rho & , \alpha \geq 0 \\ e^{-\alpha C_\alpha (t_E - t_0)} \cdot \big(\underbrace{e^{\alpha C_\alpha (t_E - t_0)}}_{\leq 1 \text{, da } \alpha \leq 0.} + t_E - t_0 \big) \cdot \rho & , \alpha \leq 0 \end{cases}$$

$$\leq \underbrace{e^{|\alpha| C_\alpha (t_E - t_0)} \cdot (1 + t_E - t_0)}_{=: C} \cdot \|T_h(u_1) - T_h(u_2)\|_{2,\infty} \quad \forall u_1, u_2 \in X_h \quad \forall h \leq h_{max} \leq \bar{h}$$

$\rho = \|v_1 - v_2\|_{2,\infty}$
$= \|T_h(u_1) - T_h(u_2)\|_{2,\infty}$

$\alpha \geq 0$: $\bar{h}$ such that $\bar{h} \cdot \alpha \leq q < 1$. $(q \in]0,1[)$

$\alpha \leq 0$: $\bar{h}$ such that $\bar{h} |\alpha| \leq \frac{q}{1-q}$.

Aufgabe 21:

```
%% Aufgabenteil (a)
f = @(t,v) v-2*sin(t);
Dvf = @(t,v) 1;
u_init=1;
t_init=0;
t_end=7;
h=0.1;
t_fine=t_init:h/100:t_end;
u_bar = @(t) cos(t)+sin(t);

%% Explizite Runge-Kutta-Verfahren
% Runge-Kutta-Tableau (m=1) Explizites Euler-Verfahren
alpha=[0];
beta=[0];
gamma=[1];

[tn_euler,un_euler]=rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,gamma);

% Runge-Kutta-Tableau (m=2) Heun-Verfahren
alpha=[0;1/2];
beta=[0 0; 1/2 0];
gamma=[0 1];

[tn_heun,un_heun]=rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,gamma);

% Runge-Kutta-Tableau (m=4) Klassisches Runge-Kutta-Verfahren
alpha=[0 1/2 1/2 1];
beta=[0 0 0 0;1/2 0 0 0;0 1/2 0 0;0 0 1 0];
gamma=[1/6 1/3 1/3 1/6];

[tn_rkclassic,un_rkclassic]=rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,
gamma);

%% implizite Runge-Kutta-Verfahren
% Runge-Kutta-Tableau (m=1, p=2)
alpha=[1/2];
beta=[1/2];
gamma=[1];

[tn_m1p2,un_m1p2]=rk_implicit(f,Dvf,u_init,t_init,t_end,h,alpha,beta,gamma);

% Runge-Kutta-Tableau (m=2, p=4)
alpha=[(3-sqrt(3))/6;(3+sqrt(3))/6];
beta=[1/4 1/4-sqrt(3)/6; 1/4+sqrt(3)/6 1/4];
gamma=[1/2 1/2];

[tn_m2p4,un_m2p4]=rk_implicit(f,Dvf,u_init,t_init,t_end,h,alpha,beta,gamma);

%% Graphische Ausgabe
figure;
subplot(1,2,1);
hold on;
plot(tn_euler,un_euler,'*b');
plot(tn_heun,un_heun,'*r');
plot(tn_rkclassic,un_rkclassic,'*g');
plot(t_fine,u_bar(t_fine),'k');
```

```
legend('explicit Euler: m=1, p=1','Heun: m=2, p=2','classical RK: m=4,
p=4','exact','Location','Best');
hold off
subplot(1,2,2);
hold on;
plot(tn_m1p2,un_m1p2,'*b');
plot(tn_m2p4,un_m2p4,'*r');
plot(t_fine,u_bar(t_fine),'k');
legend('m=1, p=2','m=2, p=4','exact','Location','Best');
hold off;

%% Aufgabenteil (b)
f = @(t,v) 1/10*[-6 2;2 -9]*v;
Dvf = @(t,v) 1/10*[-6 2;2 -9];
u_init=[1;1];
t_init=0;
t_end=7;
h=0.1;
t_fine=t_init:h/100:t_end;
u_bar = @(t) 1/5*[-exp(-t)+6*exp(-t/2);2*exp(-t)+3*exp(-t/2)];

%% Explizite Runge-Kutta-Verfahren
% Runge-Kutta-Tableau (m=1) Explizites Euler-Verfahren
alpha=[0];
beta=[0];
gamma=[1];

[tn_euler,un_euler]=rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,gamma);

% Runge-Kutta-Tableau (m=2) Heun-Verfahren
alpha=[0;1/2];
beta=[0 0; 1/2 0];
gamma=[0 1];

[tn_heun,un_heun]=rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,gamma);

% Runge-Kutta-Tableau (m=4) Klassisches Runge-Kutta-Verfahren
alpha=[0 1/2 1/2 1];
beta=[0 0 0 0;1/2 0 0 0;0 1/2 0 0;0 0 1 0];
gamma=[1/6 1/3 1/3 1/6];

[tn_rkclassic,un_rkclassic]=rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,
gamma);

%% implizite Runge-Kutta-Verfahren
% Runge-Kutta-Tableau (m=1, p=2)
alpha=[1/2];
beta=[1/2];
gamma=[1];

[tn_m1p2,un_m1p2]=rk_implicit(f,Dvf,u_init,t_init,t_end,h,alpha,beta,gamma);

% Runge-Kutta-Tableau (m=2, p=4)
alpha=[(3-sqrt(3))/6;(3+sqrt(3))/6];
beta=[1/4 1/4-sqrt(3)/6; 1/4+sqrt(3)/6 1/4];
gamma=[1/2 1/2];
```

```
[tn_m2p4,un_m2p4]=rk_implicit(f,Dvf,u_init,t_init,t_end,h,alpha,beta,gamma);

%% Graphische Ausgabe
u_exact_sol=u_bar(t_fine);
space_dim=length(u_init);
for i=1:space_dim
  figure;
  subplot(1,2,1)
  hold on;
  plot(tn_euler,un_euler(i,:),'*b');
  plot(tn_heun,un_heun(i,:),'*r');
  plot(tn_rkclassic,un_rkclassic(i,:),'*g');
  plot(t_fine,u_exact_sol(i,:),'k');
  legend('explicit Euler: m=1, p=1','Heun: m=2, p=2','classical RK: m=4,
p=4','exact','Location','Best');
  hold off
  subplot(1,2,2);
  hold on;
  plot(tn_m1p2,un_m1p2(i,:),'*b');
  plot(tn_m2p4,un_m2p4(i,:),'*r');
  plot(t_fine,u_exact_sol(i,:),'k');
  legend('m=1, p=2','m=2, p=4','exact','Location','Best');
  hold off;
end
```

```
function [tn,un] = rk_implicit(f,Dvf,u_init,t_init,t_end,h,alpha,beta,gamma)
%RK_EXPLICIT loest Anfangswertprobleme der Form
%                u'(t) = f(t,u(t)), u(t_init) = u_init
%             mit expliziten Runge-Kutta Verfahren (alpha,beta,gamma)
%             zur konstanten Zeitschrittweite h auf dem endlichen
%             Zeitintervall [t_init,t_end].

%% Initialisierung
  m=length(gamma);                   % Stufe des RK-Verfahrens
  tn=t_init:h:t_end;                 % diskretes Zeitintervall
  time_steps=length(tn);             % Anzahl der Zeitschritte
  space_dim=length(u_init);          % Dimension der DGL, Raumdimension
  un=zeros(space_dim,time_steps);    % Speicherreservierung fuer RK-Iteration
  un(:,1)=u_init;                    % Initialisierung der Anfangsdaten
  k=zeros(space_dim,m);              % Speicherreservierung der k_i's

%% Implizites Runge-Kutta-Verfahren
  for j=2:time_steps

    % 1. Berechnung der k_i's durch Loesen eines nichtlinearen
    %    Gleichungssystems
    % Startwert fuer das Newton-Verfahren: (im j-ten Schritt
    % k_i^0=f(t_j,u_j) fuer i=1..m)
    for i=1:m
        k(:,i)=f(tn(j-1),un(:,j-1));
    end
    % Mehrdimensionales Newton-Verahren
    eps=10^(-7);
    max_newton_steps=20;
    n = 0;
    xn = k(:);
    yn = F(f,alpha,beta,xn,tn(j-1),un(:,j-1),h);
    while max(abs(yn))>eps
      n = n+1;
      zn = xn - DF(Dvf,alpha,beta,xn,tn(j-1),un(:,j-1),h)\yn;
      if norm(zn-xn)==Inf
        error('Newton method has not converged: Overflow.')
      end
      xn=zn;
      yn = F(f,alpha,beta,xn,tn(j-1),un(:,j-1),h);
      if n>max_newton_steps
        error('Newton method has not converged: Maximal number of iterations
reached.')
      end
    end
    k(:)=xn;

    % 2. Berechnung der u^j's
    temp_phi=zeros(space_dim,1);
    for i=1:m
      temp_phi=temp_phi+gamma(i)*k(:,i);
    end
    un(:,j)=un(:,j-1)+h*temp_phi;
  end
end
```

```
%% Function F
  function z = F(f,alpha,beta,k,t,v,s)
    d=length(v);
    m=length(alpha);
    z=zeros(d*m,1);
    for i=1:m
      % Berechnung der Argumente von f
      temp_phi=zeros(d,1);
      for j=1:m
        temp_phi=temp_phi+beta(i,j)*k((j-1)*d+1:j*d);
      end
      % Berechnung von F_i
      z((i-1)*d+1:i*d,1)=f(t+alpha(i)*s,v+s*temp_phi)-k((i-1)*d+1:i*d);
    end
  end

%% Function DF
  function A = DF(Df,alpha,beta,k,t,v,s)
    m=length(alpha);
    d=length(v);
    A=zeros(d*m,d*m);
    for i=1:m
      % Berechnung der Argumente von Df
      temp_phi=zeros(d,1);
      for j=1:m
        temp_phi=temp_phi+beta(i,j)*k((j-1)*d+1:j*d);
      end
      % Berechnung von DF_ij
      for j=1:m
        if i==j
          A((i-1)*d+1:i*d,(j-1)*d+1:j*d)=Df(t+alpha(i)*s,v+s*temp_phi)*s*beta↙
(i,j)-eye(d,d);
        else
          A((i-1)*d+1:i*d,(j-1)*d+1:j*d)=Df(t+alpha(i)*s,v+s*temp_phi)*s*beta↙
(i,j);
        end
      end
    end
  end

%% Newton-Verfahren:
    function [nst,n]=newton(F,DF,x0,eps,newton_einfach)
    %NEWTON1D Eindimensionales Newton-Verfahren zur Berechnung von Nullstellen
    %         einer Funktion F
    %    F              : Funktion
    %    dF             : Ableitung der Funktion
    %    x0             : Startwert
    %    eps            : Genauigkeit
    %    newton_einfach : 0 = Newton-Verfahren
    %                     1 = vereinfachtes Newton-Verfahren
    %    nst            : Nullstelle der Funktion
    %    n              : Anzahl der benoetigten Iterationen

      n = 0;
      xn = x0;
      yn = F(xn);
```

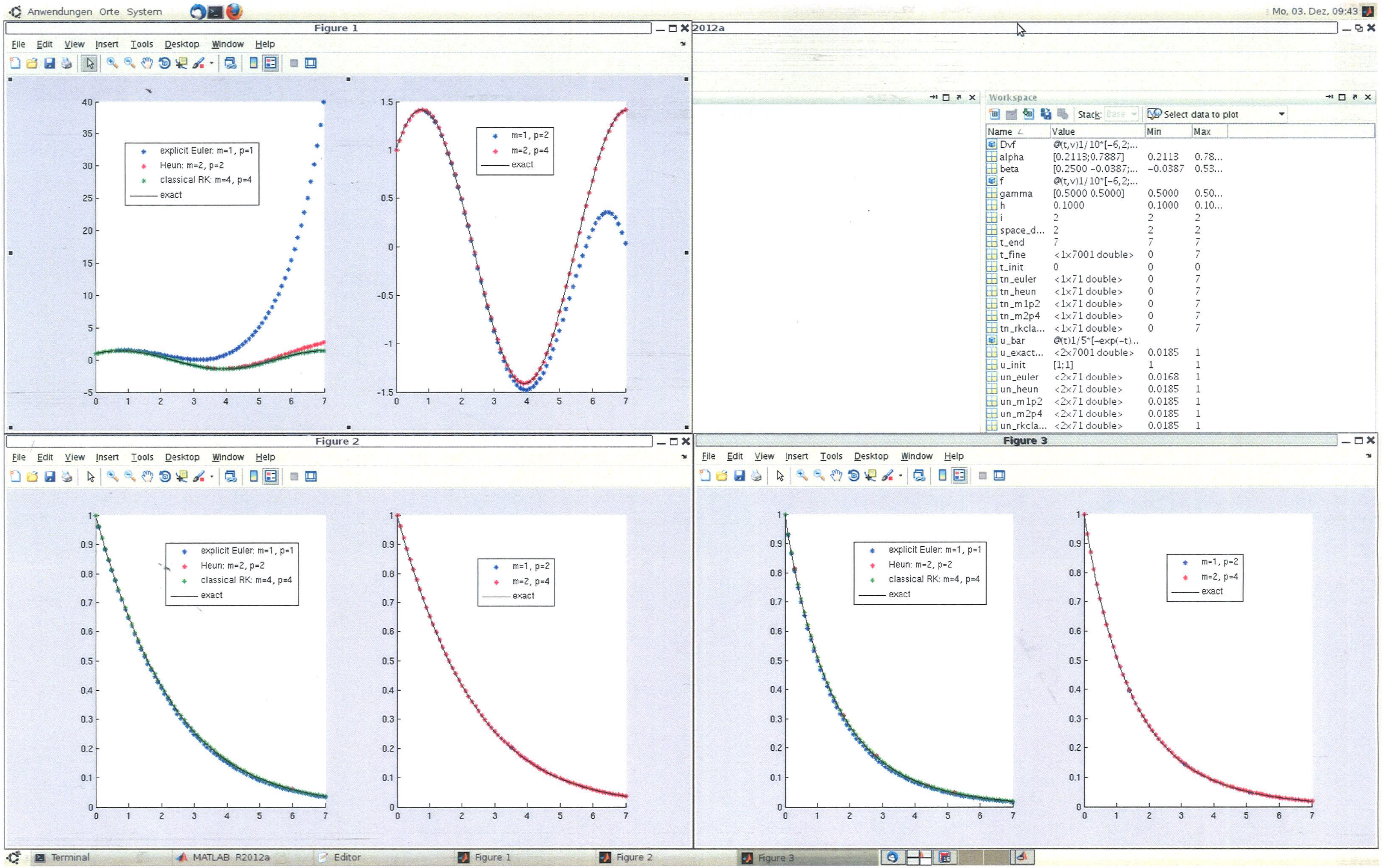
```
        if newton_einfach
            A0=DF(x0);
            while max(abs(yn))>eps
                n = n+1;
                %xn = xn - GSV(A0,yn,10^(-10));
                %xn = xn - LR(A0,yn,1);
                xn = xn - A0\yn;
                yn = F(xn);
            end
        else
            while max(abs(yn))>eps
                n = n+1;
                %xn = xn - GSV(dF(xn),yn,10^(-10));
                %xn = xn - LR(dF(xn),yn,1);
                xn = xn - DF(xn)\yn;
                yn = F(xn);
            end
        end
        nst=xn;
    end
```

```
function [tn,un] = rk_explicit(f,u_init,t_init,t_end,h,alpha,beta,gamma)
%RK_EXPLICIT loest Anfangswertprobleme der Form
%               u'(t) = f(t,u(t)), u(t_init) = u_init
%             mit expliziten Runge-Kutta Verfahren (alpha,beta,gamma)
%             zur konstanten Zeitschrittweite h auf dem endlichen
%             Zeitintervall [t_init,t_end].

%% Initialisierung
  m=length(gamma);                   % Stufe des RK-Verfahrens
  tn=t_init:h:t_end;                 % diskretes Zeitintervall
  time_steps=length(tn);             % Anzahl der Zeitschritte
  space_dim=length(u_init);          % Dimension der DGL, Raumdimension
  un=zeros(space_dim,time_steps);    % Speicherreservierung fuer RK-Iteration
  un(:,1)=u_init;                    % Initialisierung der Anfangsdaten
  k=zeros(space_dim,m);              % Speicherreservierung der k_i's

%% Explizites Runge-Kutta-Verfahren
  for j=2:time_steps
    % 1. Rekursive Berechnung der k_i's
    for i=1:m
      temp_k=zeros(space_dim,1);
      for l=1:i-1
        temp_k=temp_k+beta(i,l)*k(:,l);
      end
      k(:,i)=f(tn(j-1)+alpha(i)*h,un(:,j-1)+h*temp_k);
    end

    % 2. Berechnung der u^j's
    temp_phi=zeros(space_dim,1);
    for i=1:m
      temp_phi=temp_phi+gamma(i)*k(:,i);
    end
    un(:,j)=un(:,j-1)+h*temp_phi;
  end
end
```


Anwendungen Orte System
Mo, 03. Dez, 09:43
Figure 1
File Edit View Insert Tools Desktop Window Help
explicit Euler: m=1, p=1
Heun: m=2, p=2
classical RK: m=4, p=4
exact
m=1, p=2
m=2, p=4
exact
2012a
Workspace
Stack: Base
Select data to plot
Name Value Min Max
Dvf @(t,v)1/10*[-6,2;...
alpha [0.2113;0.7887] 0.2113 0.78...
beta [0.2500 -0.0387;... -0.0387 0.53...
f @(t,v)1/10*[-6,2;...
gamma [0.5000 0.5000] 0.5000 0.50...
h 0.1000 0.1000 0.10...
i 2 2 2
space_d... 2 2 2
t_end 7 7 7
t_fine <1x7001 double> 0 7
t_init 0 0 0
tn_euler <1x71 double> 0 7
tn_heun <1x71 double> 0 7
tn_m1p2 <1x71 double> 0 7
tn_m2p4 <1x71 double> 0 7
tn_rkcla... <1x71 double> 0 7
u_bar @(t)1/5*[-exp(-t)...
u_exact... <2x7001 double> 0.0185 1
u_init [1;1] 1 1
un_euler <2x71 double> 0.0168 1
un_heun <2x71 double> 0.0185 1
un_m1p2 <2x71 double> 0.0185 1
un_m2p4 <2x71 double> 0.0185 1
un_rkcla... <2x71 double> 0.0185 1
Figure 2
File Edit View Insert Tools Desktop Window Help
explicit Euler: m=1, p=1
Heun: m=2, p=2
classical RK: m=4, p=4
exact
m=1, p=2
m=2, p=4
exact
Figure 3
File Edit View Insert Tools Desktop Window Help
explicit Euler: m=1, p=1
Heun: m=2, p=2
classical RK: m=4, p=4
exact
m=1, p=2
m=2, p=4
exact
Terminal
MATLAB R2012a
Editor
Figure 1
Figure 2
Figure 3