8 Numerik gewöhnlicher Differentialgleichungen

8.1 Grundlagen

In der Numerik von gewöhnlichen Differentialgleichungen werden vorwiegend Aufgaben folgender Art behandelt:

Definition 8.1.1 (Aufgabenstellung)

- *gegeben ist eine Anfangswertaufgabe:*
  - *explizites Differentialgleichungssystem* 1*–Ordnung:*
    $$\vec{y}' = \vec{f}(x, \vec{y}) \quad \textit{mit} \quad \vec{f} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^n$$
  - *Anfangswert:*
    $$\vec{y}(x_0) = \vec{y}^{\,0} = \left(y_1^0, \ldots, y_n^0\right)^T \in \mathbb{R}^n$$
- *gesucht ist der Funktionswert (–Vektor)* $\vec{y}(b)$ *an der rechts von* x_0 *liegenden Stelle* b*.*

Es ist also nicht die Lösungsfunktion als solche gesucht, sondern der Funktionswert der Lösung an einer bestimmten Stelle $b > x_0$.

Bemerkung 8.1.2

- *Unter vernünftigen Voraussetzungen an die Funktion* $\vec{f}$ *— etwa:*
  $\vec{f}$ *ist auf* $[x_0, b] \times \mathbb{R}^n$
  - *stetig*
  - *bzgl.* $\vec{y}$ *gleichmäßig lipschitzstetig, d.h. es gibt eine Konstante* $L > 0$ *mit*
    $$||\vec{f}(x, \vec{y}_1) - \vec{f}(x, \vec{y}_2)|| \leq L \cdot ||\vec{y}_1 - \vec{y}_2||$$
    für alle $x \in [x_0, b]$ *und* $\vec{y}_1, \vec{y}_2 \in \mathbb{R}^n$

  ist die Anfangswertaufgabe 8.1.1 eindeutig lösbar.
- *In vielen Anwendungen gilt* $n = 1$*, d.h. es handelt sich um eine explizite Differentialgleichung 1. Ordnung.*

Bemerkung 8.1.3 *Differentialgleichungen (oder auch Differentialgleichungssysteme) höherer Ordnung können immer auf ein äquivalentes Differentialgleichungssystem 1. Ordnung zurückgeführt werden:*
Die DGL
$$z^{(n)} = f(x, z, z', z'', \ldots, z^{(n-1)}) \tag{6}$$
n–ter Ordnung.

wird durch:

$$\begin{aligned} y_1 &= z \\ y_2 &= z' \\ &\vdots \\ y_{n-1} &= z^{(n-2)} \\ y_n &= z^{(n-1)} \end{aligned}$$

überführt in das Differentialgleichungssystem 1. Ordnung:

$$\begin{aligned} y_1' &= y_2 \\ y_2' &= y_3 \\ &\vdots \\ y_{n-1}' &= y_n \\ y_n' &= f(x, y_1, y_2, \ldots, y_n) \end{aligned} \tag{7}$$

Eine Lösung z der Differentialgleichung 6 der Ordnung n führt zur Lösung

$$\vec{y} = (z, z', \ldots, z^{(n-1)})^T$$

des Differentialgleichungssystems 7 und umgekehrt eine Lösung

$$\vec{y} = (y_1, y_2, \ldots, y_n)^T$$

des Differentialgleichungssystems 7 führt zu der Lösung

$$z = y_1$$

der Differentialgleichung 6 der Ordnung n.

Beispiel 8.1.4 (2–Körper–Problem)

Befindet sich im Punkt $\vec{P}_1(t)$ im Raum eine Masse M_1 und bewegt sich mit der Geschwindigkeit $\vec{P}_1(t)'$, sowie in Punkt $\vec{P}_2(t)$ eine Masse M_2 mit Geschwindigkeit $\vec{P}_2'(t)$, so beeinflussen sich die Massen gegenseitig aufgrund der Gravitation. Es gilt:

$$\begin{aligned} \vec{P}_1'' &= \gamma \cdot M_2 \cdot \frac{(\vec{P}_2 - \vec{P}_1)}{||\vec{P}_2 - \vec{P}_1||^3} \\ \vec{P}_2'' &= \gamma \cdot M_1 \cdot \frac{(\vec{P}_1 - \vec{P}_2)}{||\vec{P}_2 - \vec{P}_1||^3} \end{aligned}$$

(Ableitungen nach der Zeit t) wobei $\gamma = 6.674 \cdot 10^{-11} \frac{m^3}{Kg \cdot s^2}$ die Gravitationskonstante ist. Mit

$$\vec{P}_i(t) = \begin{pmatrix} P_{ix}(t) \\ P_{iy}(t) \\ P_{iz}(t) \end{pmatrix}, \ \vec{P}_i'(t) = \begin{pmatrix} P_{ix}'(t) \\ P_{iy}'(t) \\ P_{iz}'(t) \end{pmatrix} \text{ und } \vec{P}_i''(t) = \begin{pmatrix} P_{ix}''(t) \\ P_{iy}''(t) \\ P_{iz}''(t) \end{pmatrix}$$

für $i = 1, 2$ und

$$y_1(t) = P_{1x}(t),\ y_2(t) = P_{1y}(t),\ y_3(t) = P_{1z}(t),\ y_4(t) = P_{1x}'(t),\ y_5(t) = P_{1y}'(t),\ y_6(t) = P_{1z}'(t)$$

$y_7(t) = P_{2x}(t),\ y_8(t) = P_{2y}(t),\ y_9(t) = P_{2z}(t),\ y_{10}(t) = P'_{2x}(t),\ y_{11}(t) = P'_{2y}(t),\ y_{12}(t) = P'_{2z}(t)$
führt das auf das Differentialgleichungssystem 1. Ordnung:

$$
\begin{aligned}
y'_1 &= y_4 \\
y'_2 &= y_5 \\
y'_3 &= y_6 \\
y'_4 &= \gamma \cdot M_2 \cdot \frac{y_7 - y_1}{\left(\sqrt{(y_7-y_1)^2 + (y_8-y_2)^2 + (y_9-y_3)^2}\right)^3} \\
y'_5 &= \gamma \cdot M_2 \cdot \frac{y_8 - y_2}{\left(\sqrt{(y_7-y_1)^2 + (y_8-y_2)^2 + (y_9-y_3)^2}\right)^3} \\
y'_6 &= \gamma \cdot M_2 \cdot \frac{y_9 - y_3}{\left(\sqrt{(y_7-y_1)^2 + (y_8-y_2)^2 + (y_9-y_3)^2}\right)^3} \\
y'_7 &= y_{10} \\
y'_8 &= y_{11} \\
y'_9 &= y_{12} \\
y'_{10} &= \gamma \cdot M_1 \cdot \frac{y_1 - y_7}{\left(\sqrt{(y_7-y_1)^2 + (y_8-y_2)^2 + (y_9-y_3)^2}\right)^3} \\
y'_{11} &= \gamma \cdot M_1 \cdot \frac{y_2 - y_8}{\left(\sqrt{(y_7-y_1)^2 + (y_8-y_2)^2 + (y_9-y_3)^2}\right)^3} \\
y'_{12} &= \gamma \cdot M_1 \cdot \frac{y_3 - y_9}{\left(\sqrt{(y_7-y_1)^2 + (y_8-y_2)^2 + (y_9-y_3)^2}\right)^3}
\end{aligned}
$$

Satz 8.1.5 (Rückführung der Anfangswertaufgabe auf eine Integralgleichung)
Es sei eine Anfangswertaufgabe

$$\vec{y}\,' = \vec{f}(x,\vec{y}) \quad , \quad \vec{y}(x_0) = \vec{y}_0$$

gegeben.
Integriert man die Gleichung $\vec{y}\,' = \vec{f}(x,\vec{y})$, unter der Voraussetzung, dass $\vec{y}$ und $\vec{y}\,'$ eine Funktion von x ist, über dem Intervall $[x_0, x]$ (d.h. linker Randpunkt ist das feste x_0 und der rechte Randpunkt das "variable" x), so erhält man (wegen der oberen Grenze x wird hier t als Integrationsvariable eingeführt, es wird komponentenweise integriert):

$$
\begin{aligned}
\int_{x_0}^{x} \vec{y}\,'(t)\,dt &= \int_{x_0}^{x} \vec{f}(t,\vec{y}(t))\,dt \\
\Leftrightarrow \quad \vec{y}(x) - \vec{y}(x_0) &= \int_{x_0}^{x} \vec{f}(t,\vec{y}(t))\,dt \\
\Leftrightarrow \quad \vec{y}(x) &= \vec{y}(x_0) + \int_{x_0}^{x} \vec{f}(t,\vec{y}(t))\,dt \quad \textit{mit } \vec{y}(x_0) = \vec{y}_0 \\
\Leftrightarrow \quad \vec{y}(x) &= \vec{y}_0 + \int_{x_0}^{x} \vec{f}(t,\vec{y}(t))\,dt
\end{aligned}
$$

Die Anfangswertaufgabe ist also äquivalent zur Integralgleichung

$$\vec{y}(x) \;=\; \vec{y}_0 + \int_{x_0}^{x} \vec{f}(t,\vec{y}(t))\; dt \tag{8}$$

Diese Integralgleichung ist mathematisch gesehen genauso kompliziert wie die Anfangswertaufgabe, denn die Anfangswertaufgabe besteht (im Wesentlichen) aus einer Gleichung, in der eine unbekannte Funktion $\vec{y}$ und deren Ableitung $\vec{y}\,'$ vorkommen — in der Integralgleichung taucht dieselbe unbekannte Funktion $\vec{y}$ und ein Integral über diese Funktion auf.

Bemerkung 8.1.6 (Ansatz für numerische Verfahren)

- *Intervallaufteilung:*

$$x_0 < x_1 < \ldots < x_k < x_{k+1} < \ldots < x_n = b$$

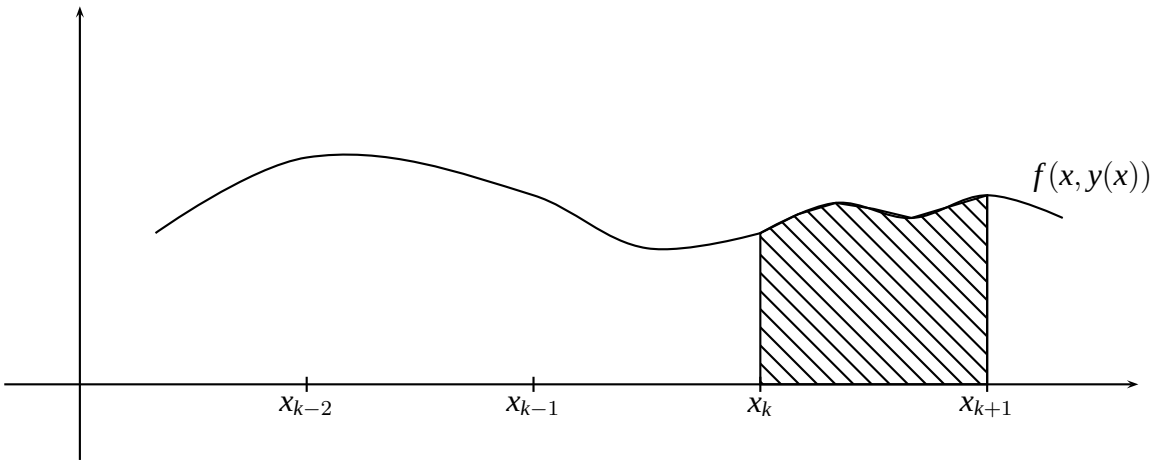
- *unter der Voraussetzung, dass man $\vec{y}_k = \vec{y}(x_k)$ schon kennt, die Integralgleichung 8 lokal für das Intervall von x_k bis x_{k+1} anwenden:*

$$\vec{y}(x_{k+1}) = \vec{y}_{k+1} \;=\; \vec{y}_k + \int_{x_k}^{x_{k+1}} \vec{f}(t,\vec{y}(t))\; dt$$

- *das in dieser Gleichung stehende Integral*

$$\int_{x_k}^{x_{k+1}} \vec{f}(t,\vec{y}(t))\; dt$$

näherungsweise mit einer (interpolatorischen) Quadraturformel berechnen.

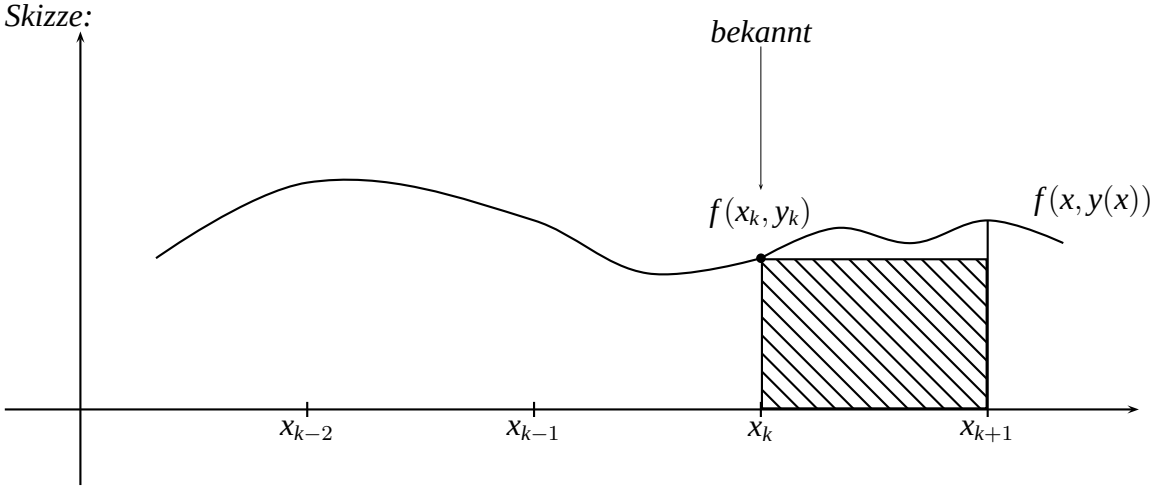


Beispiel 8.1.7 (explizites Euler–Verfahren)

Nimmt man zur näherungsweisen Berechnung des Integrals die (linksseitige) Rechteckregel*:*

$$\int_{x_k}^{x_{k+1}} \vec{f}(t,\vec{y}(t))\,dt \;\approx\; \vec{f}(x_k,\vec{y}_k)\cdot(x_{k+1}-x_k)$$

Skizze:



so erhält man die Formel:

$$\vec{y}_{k+1} \approx \vec{y}_k + h_k\cdot\vec{f}(x_k,\vec{y}_k) \quad \textit{mit} \quad h_k = x_{k+1}-x_k$$

Das auf dieser Formel beruhende Verfahren heißt **explizites Euler–Cauchy–Polygonzugverfahren**.

Algorithmus 8.1.8 (explizites Euler–Verfahren)

Gegeben: Anfangswertaufgabe $\vec{y}' = \vec{f}(x,\vec{y})$, $\vec{y}(x_0) = \vec{y}_0$.
Gesucht: "Funktionswert" (eigentlich: Vektor) $\vec{y}(b)$ *mit* $b > x_0$.
Vorgehen:
Unterteile das Intervall $[x_0,b]$ *in viele kleine Teilintervalle*

$$x_0 < x_1 < x_2 < \ldots < x_n = b$$

und setze $h_k = x_{k+1} - x_k$ *(Teilintervalllänge des k-ten Intervalls).*
Ausgehend vom Anfangswert $(x_0,\vec{y}_0)$ *berechne iterativ für* $k = 0,1,2,\ldots,n-1$*:*

$$\vec{y}_{k+1} = \vec{y}_k + h_k\cdot\vec{f}(x_k,\vec{y}_k)$$

Dann gilt:

$$\vec{y}_n \approx \vec{y}(x_n) = \vec{y}(b)$$

Man kann zeigen: Sei $h = \max\{\, h_k \mid 0 \le k < n \,\}$ *die größte der beteiligten Teilintervalllängen, so gilt für den Fehler:*

$$||\vec{y}_n - \vec{y}(b)|| \in \mathcal{O}(h)$$

d.h. das Euler–Cauchy–Verfahren hat die Ordnung 1.

Beispiel 8.1.9

gegeben: *Anfangswertaufgabe* $y' = \underbrace{\sqrt{x+y}}_{f(x,y)}, \quad y(0) = 1.0$

gesucht: $y(1.0)$

Es soll das Euler–Cauchy–Polygonzugverfahren mit konstanter Schrittweite $h = 0.1$ durchgeführt werden.

- *Anfangswert:* $x_0 = 0.0, y_0 = 1.0$
- $x_1 = 0.1, y_1 = y_0 + h \cdot f(x_0, y_0) = 1.1000000000$
- $x_2 = 0.2, y_2 = y_1 + h \cdot f(x_1, y_1) = 1.2095445115$
- $x_3 = 0.3, y_3 = y_2 + h \cdot f(x_2, y_2) = 1.3282687513$
- $x_4 = 0.4, y_4 = y_3 + h \cdot f(x_3, y_3) = 1.4558723857$
- $x_5 = 0.5, y_5 = y_4 + h \cdot f(x_4, y_4) = 1.5921027929$
- $x_6 = 0.6, y_6 = y_5 + h \cdot f(x_5, y_5) = 1.7367438242$
- $x_7 = 0.7, y_7 = y_6 + h \cdot f(x_6, y_6) = 1.8896079411$
- $x_8 = 0.8, y_8 = y_7 + h \cdot f(x_7, y_7) = 2.0505305294$
- $x_9 = 0.9, y_9 = y_8 + h \cdot f(x_8, y_8) = 2.2193656718$
- $x_{10} = 1.0, y_{10} = y_9 + h \cdot f(x_9, y_9) = 2.3959829323$

Ein Näherungswert für den gesuchten Funktionswert $y(1.0)$ ist $y_{10} = 2.3959829323$.

Bemerkung 8.1.10 (geometrische Beschreibung des Euler–Cauchy–Verfahrens)

Im eindimensionalen Fall kann das explizite Euler–Cauchy-Polygonzugverfahren geometrisch anhand des Richtungsfeldes der Differentialgleichung $y' = f(x,y)$ hergeletitet werden:
Zur Anfangswertaufgabe $y' = f(x,y), y(x_0) = y_0$ sei der Funktionswert der Lösung $y(x)$ an der Stelle $b > x_0$ gesucht.

Hierzu wird auf der x–Achse die Strecke von x_0 nach b in kleine Teilintervalle unterteilt:

$$x_0 < x_1 < x_2 < \ldots < x_n = b$$

Aufgrund des Anfangswertes kennen wir den Funktionswert y_0 der Lösung $y(x)$ an der Stelle x_0 — die Lösung $y(x)$ soll ja durch den Punkt (x_0, y_0) gehen.
Ausgehend von diesem Anfangswert (x_0, y_0) wird jetzt wie folgt ein Näherungswert y_1 für den Funktionswert $y(x_1)$ der Lösung $y(x)$ an der Stelle x_1 ermittelt:

1. *Man setzt den Punkt (x_0, y_0) in die rechte Seite der Differentialgleichung ein und erhält die Steigung $y'_0 = f(x_0, y_0)$ der (ansonsten unbekannten) Lösungskurve $y(x)$ im Punkt (x_0, y_0).*
2. *Hiermit ist die Tangente der Lösungskurve im Punkt (x_0, y_0) bekannt: diese geht durch den Punkt (x_0, y_0) und hat die Steigung $y'_0 = f(x_0, y_0)$.*

3. In einer (kleinen) Umgebung von x_0 wird die Lösung $y(x)$ ganz gut durch ihre Tangente angenähert:

Es gilt nach Taylor:

$$y(x) \approx \underbrace{y(x_0) + y'(x_0) \cdot (x - x_0)}_{\text{Tangente von } y(x) \text{ in } (x_0,y_0)}$$

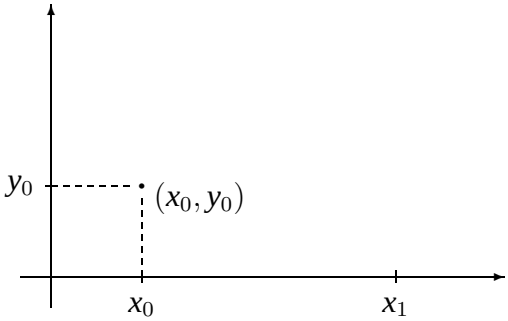
Durch Einsetzen von x_1 für x folgt mit $y(x_0) = y_0$ und $y'(x_0) = y'_0 = f(x_0, y_0)$:

$$y(x_1) \approx y_1 := y_0 + f(x_0, y_0) \cdot (x_1 - x_0)$$

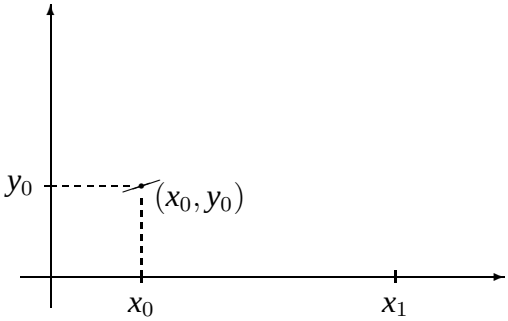
Mit dieser Formel kann also anhand des Anfangswertes (x_0, y_0) ein Näherungswert y_1 für den Funktionswert der (unbekannten) Lösung $y(x)$ an der Stelle x_1 ermittelt werden!

Skizzen:

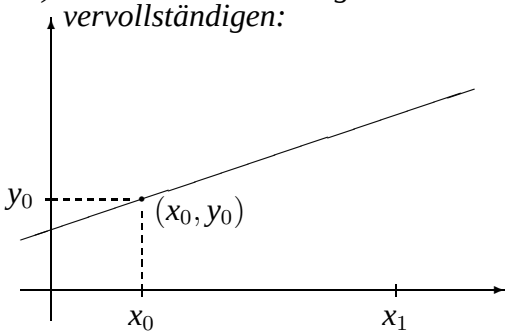
a.) Anfangswert:



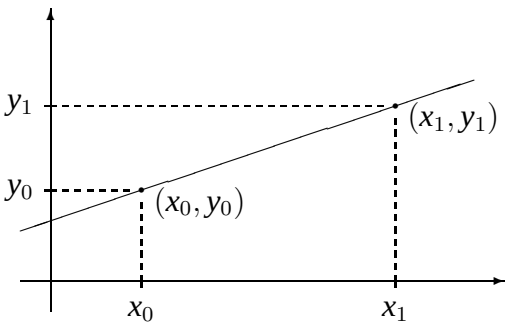
b.) Linienelement mit Steigung $f(x_0, y_0)$ ermitteln:



c.) Linienelement zu Tangente vervollständigen:



d.) Tangente an der Stelle x_1 auswerten:



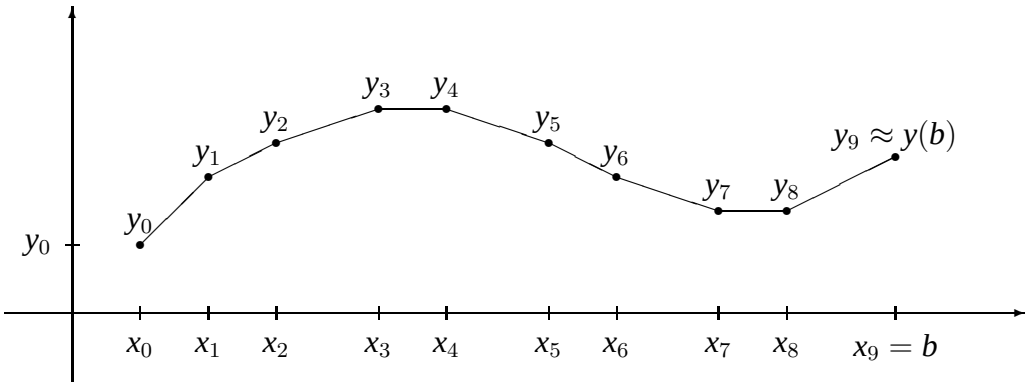
Mit diesem Verfahren bzw. der Formel

$$y_1 \;=\; y_0 + (x_1 - x_0) \cdot f(x_0, y_0)$$

ist es also gelungen, ausgehend vom Anfangspunkt (x_0, y_0) einen weiter rechts liegenden Punkt (x_1, y_1) zu ermitteln mit $y_1 \approx y(x_1)$.
Dies kann nun iterativ fortgesetzt werden:

$$\begin{array}{rcl}
y_2 &=& y_1 + (x_2 - x_1) \cdot f(x_1, y_1) \quad , \quad y_2 \approx y(x_2) \\
y_3 &=& y_2 + (x_3 - x_2) \cdot f(x_2, y_2) \quad , \quad y_3 \approx y(x_3) \\
&\vdots& \\
y_n &=& y_{n-1} + (x_n - x_{n-1}) \cdot f(x_{n-1}, y_{n-1}) \quad , \quad y_n \approx y(x_n) = y(b)
\end{array}$$

Die Linienelemente in den Punkten (x_k, y_k) legen die Geradenstücke von (x_k, y_k) nach (x_{k+1}, y_{k+1}) fest und man gelangt mit diesen Geradenstücken von einem Punkt zum nächsten:
Skizze:



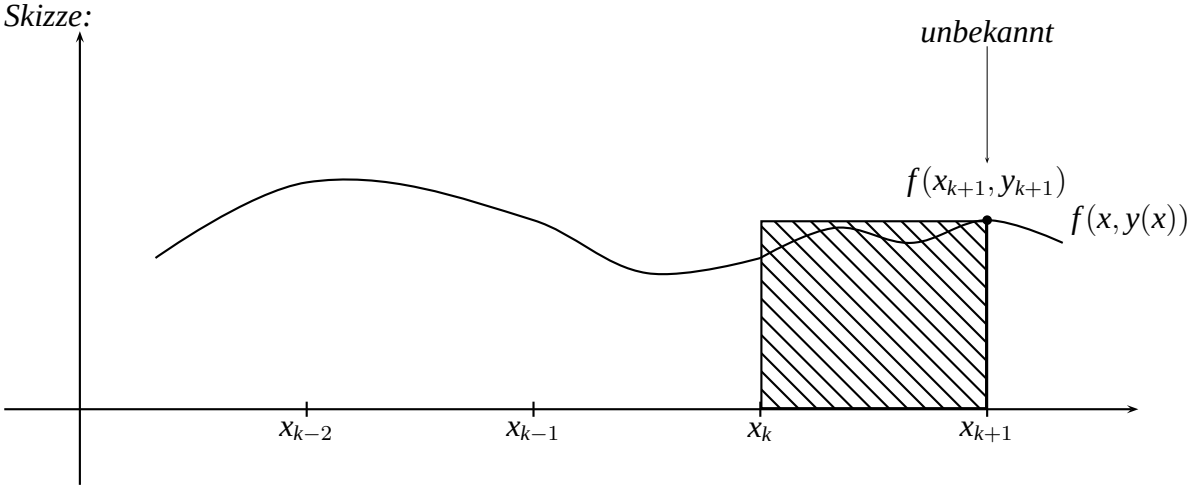
Diese Skizze motiviert auch den Namen Polygonzugverfahren, der in der Literatur ebenfalls für dieses Euler–Cauchy–Verfahren verwendet wird.

Beispiel 8.1.11 (implizites Euler–Verfahren)

Nimmt man zur näherungsweisen Berechnung des Integrals die (rechtsseitige) Rechteckregel*:*

$$\int_{x_k}^{x_{k+1}} \vec{f}(t, \vec{y}(t))\, dt \;\approx\; \vec{f}(x_{k+1}, \vec{y}_{k+1}) \cdot (x_{k+1} - x_k)$$

Skizze:



so erhält man die Formel:

$$\vec{y}_{k+1} \approx \vec{y}_k + h_k \cdot \vec{f}(x_{k+1}, \vec{y}_{k+1}) \quad \textit{mit} \quad h_k = x_{k+1} - x_k$$

In dieser Formel taucht der zu berechnende Wert (Vektor) $\vec{y}_{k+1}$ sowohl auf der linken Seite, als auch auf der rechten Seite als zweites Argument der Funktion $\vec{f}$ auf — es handelt sich somit um eine **implizite** *Gleichung für $\vec{y}_{k+1}$.*
Diese implizite Gleichung muss dann gelöst werden — im Allgemeinen iterativ.
Das auf dieser Formel beruhende Verfahren heißt **implizites Euler–Verfahren**.

Algorithmus 8.1.12 (implizites Euler–Verfahren)

Gegeben: Anfangswertaufgabe $\vec{y}' = \vec{f}(x, \vec{y})$, $\vec{y}(x_0) = \vec{y}_0$.
Gesucht: "Funktionswert" (eigentlich: Vektor) $\vec{y}(b)$ *mit* $b > x_0$.
Vorgehen:
Unterteile das Intervall $[x_0, b]$ *in viele kleine Teilintervalle*

$$x_0 < x_1 < x_2 < \ldots < x_n = b$$

und setze $h_k = x_{k+1} - x_k$ *(Teilintervalllänge des k-ten Intervalls).*
Ausgehend vom Anfangswert $(x_0, \vec{y}_0)$ *berechne iterativ für* $k = 0, 1, 2, \ldots, n-1$*:*

$$\vec{y}_{k+1} = \vec{y}_k + h_k \cdot \vec{f}(x_{k+1}, \vec{y}_{k+1})$$

Diese (für jeden Schritt von x_k *nach* x_{k+1} *anfallende)* **implizite** *Gleichung muss iterativ ausgehend von einem geeigneten Startvektor gelöst werden.*
Dann gilt:

$$\vec{y}_n \approx \vec{y}(x_n) = \vec{y}(b)$$

Auch dieses Verfahren hat die Ordnung 1*, d.h. mit* $h = \max\{\ h_k \mid 0 \leq k < n\ \}$ *gilt für den Fehler:*

$$||\vec{y}_n - \vec{y}(b)|| \in \mathcal{O}(h)$$

Bemerkung 8.1.13 (Durchführung der Fixpunktiteration in jedem Schritt)

Die in jedem Schritt des impliziten Euler–Verfahrens (iterativ) zu lösende Fixpunktgleichung:

$$\vec{y}_{k+1} = \vec{y}_k + h_k \cdot \vec{f}(x_{k+1}, \vec{y}_{k+1})$$

kann umgeschrieben werden zu

$$\underbrace{\vec{y}_{k+1} - \vec{y}_k}_{=:\ \vec{z}} = h_k \cdot \vec{f}(x_{k+1}, \vec{y}_k + \underbrace{\vec{y}_{k+1} - \vec{y}_k}_{=:\ \vec{z}})$$

also zu

$$\vec{z} = h_k \cdot \vec{f}(x_{k+1}, \vec{y}_k + \vec{z})$$

wobei $\vec{z} = \vec{0}$ *ein vernünftiger Startvektor für die Fixpunktiteration ist.*
Hierbei kann

- *die Gleichung*

  $$\vec{z} = h_k \cdot \vec{f}(x_{k+1}, \vec{y}_k + \vec{z})$$

  selbst als Fixpunktgleichung verwendet und mit dieser Gleichung die Fixpunktiteration durchgeführt werden.

- *die Gleichung umgeformt werden zu einer Nullstellengleichung:*

  $$\vec{g}(\vec{z}) = \vec{z} - h_k \cdot \vec{f}(x_{k+1}, \vec{y}_k + \vec{z}) = \vec{0}$$

  für welche dann das (ggf. mehrdimensionale) Newton–Verfahren zur Nullstellenbestimmung (mit Startvektor $\vec{0}$*) durchgeführt werden kann.*

Beispiel 8.1.14

gegeben: *Anfangswertaufgabe* $y' = \underbrace{\sqrt{x+y}}_{f(x,y)}, \quad y(0) = 1.0$

gesucht: $y(1.0)$

Es soll das implizite Euler–Verfahren mit konstanter Schrittweite $h = 0.1$ durchgeführt werden. Die in jedem Schritt zu lösende implizite Gleichung lautet:

$$z = h \cdot f(x_{k+1}, y_k + z) = h \cdot \sqrt{x_{k+1} + y_k + z}$$

a.) Jeweilige Berechnung der impliziten Gleichung mit dem allgemeinen Iterationsverfahren

b.) Jeweilige Berechnung der impliziten Gleichung mit dem Newton–Verfahren

Startwert jeweils 0 (die Iteration wird abgebrochen, sobald sich zwei iterierte Werte um weniger als 10^{-10} unterscheiden):

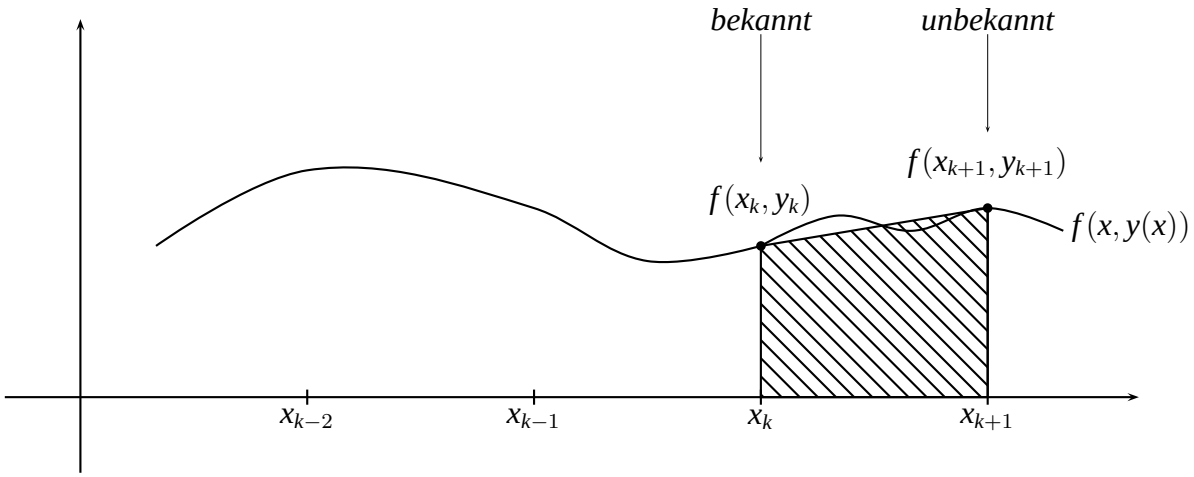
k	x_k	y_k	*Anzahl Iterationen*	
			a.)	*b.)*
0	0.000000	1.0000000000	–	–
1	0.100000	1.1100000000	8	4
2	0.200000	1.2295643924	8	4
3	0.300000	1.3583409811	8	3
4	0.400000	1.4960376646	8	3
5	0.500000	1.6424073100	8	3
6	0.600000	1.7972374575	8	3
7	0.700000	1.9603430380	8	3
8	0.800000	2.1315610586	8	3
9	0.900000	2.3107466223	8	3
10	1.000000	2.4977698797	7	3

Beispiel 8.1.15 (Trapezregel)

Die Berechnung des Integrals mit der Trapezregel:

$$\int_{x_k}^{x_{k+1}} \vec{f}(t, \vec{y}(t))\, dt \;\approx\; \frac{(x_{k+1} - x_k)}{2} \cdot \left(\vec{f}(x_k, \vec{y}_k) + \vec{f}(x_{k+1}, \vec{y}_{k+1}) \right)$$

Skizze:



führt mit $h_k = x_{k+1} - x_k$ auf die ebenfalls **implizite** *Gleichung:*

$$\vec{y}_{i+k} = \vec{y}_k + \frac{h_k}{2} \cdot \vec{f}(x_k, \vec{y}_k) + \frac{h_k}{2} \cdot \vec{f}(x_{k+1}, \vec{y}_{k+1})$$

für den zu berechnenden Wert $\vec{y}_{k+1}$.
Das auf dieser Formel beruhende (implizite) Verfahren für Differentialgleichungsn heißt Trapezregel.

Algorithmus 8.1.16 (Trapezregel)

Gegeben: Anfangswertaufgabe $\vec{y}' = \vec{f}(x, \vec{y})$, $\vec{y}(x_0) = \vec{y}_0$.
Gesucht: "Funktionswert" (eigentlich: Vektor) $\vec{y}(b)$ mit $b > x_0$.
Vorgehen:
Unterteile das Intervall $[x_0, b]$ in viele kleine Teilintervalle

$$x_0 < x_1 < x_2 < \ldots < x_n = b$$

und setze $h_k = x_{k+1} - x_k$ (Teilintervalllänge des k-ten Intervalls).
Ausgehend vom Anfangswert $(x_0, \vec{y}_0)$ berechne iterativ für $k = 0, 1, 2, \ldots, n-1$:

$$\vec{y}_{k+1} = \vec{y}_k + \frac{h_k}{2} \cdot \vec{f}(x_k, \vec{y}_k) + \frac{h_k}{2} \cdot \vec{f}(x_{k+1}, \vec{y}_{k+1})$$

Diese (für jeden Schritt von x_k nach x_{k+1} anfallende) **implizite** *Gleichung muss iterativ ausgehend von einem geeigneten Startvektor gelöst werden.*
Dann gilt:

$$\vec{y}_n \approx \vec{y}(x_n) = \vec{y}(b)$$

Dieses Verfahren hat die Ordnung 2, d.h. mit $h = \max\{\ h_k \mid 0 \le k < n\ \}$ gilt für den Fehler:

$$||\vec{y}_n - \vec{y}(b)|| \in \mathcal{O}(h^2)$$

Bemerkung 8.1.17
Mit $\vec{z} = \vec{y}_{k+1} - \vec{y}_k$ kann diese implizite Gleichung umgeschrieben werden zu

$$\vec{z} = \frac{h}{2} \cdot \vec{f}(x_k, \vec{y}_k) + \frac{h}{2} \cdot \vec{f}(x_{k+1}, \vec{y}_k + \vec{z})$$

und diese Gleichung kann (ausgehend vom Startvektor $\vec{0}$)

- *entweder direkt zur Iteration verwendet werden,*
- *oder (nach Umformung in eine Nullstellengleichung) mit dem Newton–Verfahren gelöst werden.*

Beispiel 8.1.18
gegeben: *Anfangswertaufgabe* $y' = \underbrace{\sqrt{x+y}}_{f(x,y)}, \quad y(0) = 1.0$

gesucht: $y(1.0)$

Es soll die Trapezregel mit konstanter Schrittweite $h = 0.1$ *durchgeführt werden.*
Die in jedem Schritt zu lösende implizite Gleichung lautet:

$$z = \frac{h}{2} \cdot f(x_k, y_k) + \frac{h}{2} \cdot f(x_{k+1}, y_k + z) = \frac{h}{2} \cdot \sqrt{x_k + y_k} + \frac{h}{2} \cdot \sqrt{x_{k+1} + y_k + z}$$

a.) Jeweilige Berechnung der impliziten Gleichung mit dem allgemeinen Iterationsverfahren

b.) Jeweilige Berechnung der impliziten Gleichung mit dem Newton–Verfahren

Startwert jeweils 0 *(die Iteration wird abgebrochen, sobald sich zwei iterierte Werte um weniger als* 10^{-10} *unterscheiden):*

k	x_k	y_k	*Anzahl Iterationen*	
			a.)	*b.)*
0	0.000000	1.0000000000	–	–
1	0.100000	1.1048835949	7	3
2	0.200000	1.2193351156	7	3
3	0.300000	1.3429926794	7	3
4	0.400000	1.4755578207	7	3
5	0.500000	1.6167790988	7	3
6	0.600000	1.7664410793	7	3
7	0.700000	1.9243566154	7	3
8	0.800000	2.0903612584	7	3
9	0.900000	2.2643090958	6	3
10	1.000000	2.4460695821	6	3

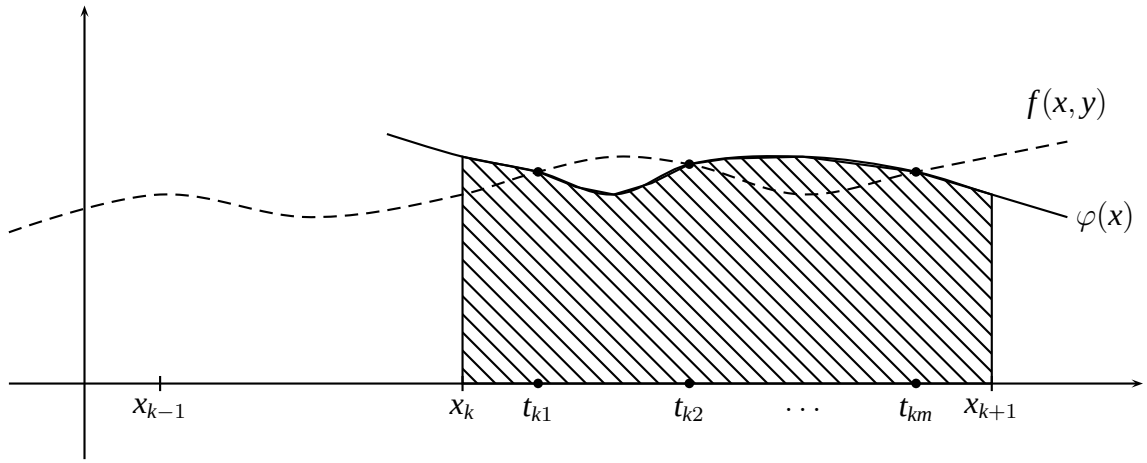
Definition und Bemerkung 8.1.19

Zur Berechnung des neuen Wertes $\vec{y}_{k+1}$ *wird in allen Verfahren das Integral*

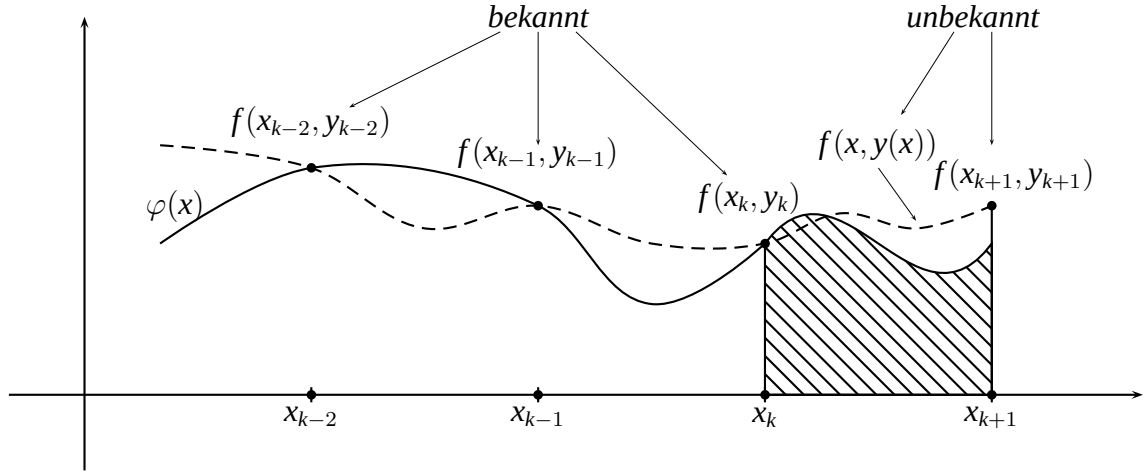
$$\int_{x_k}^{x_{k+1}} \vec{f}(t, \vec{y}(t))\, dt$$

über ein Interpolationspolynom $\vec{\varphi}(t)$ *für die unbekannte Funktion* $\vec{f}(t, \vec{y}(t))$ *näherungsweise berechnet.*
Je nachdem, wo die Stützstellen dieses Interpolationspolynoms liegen, unterscheidet man die Verfahrenstypen:

Einschrittverfahren: *Bei den Einschrittverfahren liegen* **alle** *Stützstellen des Interpolationspolynoms $\vec{\varphi}$ im aktuellen Integrationsintervall $[x_k, x_{k+1}]$:*



Mehrschrittverfahren *Bei den Mehrschrittverfahren sind die Stützstellen des Interpolationspolynoms ausschließlich unter den bereits durch die Intervallaufteilung gegebenen x–Werten:*



Je nachdem, ob x_{k+1} zu den Stützstellen des Interpolations dazugehört oder nicht, spricht man von einem **expliziten** *(gehört nicht dazu) oder einem* **impliziten** *(gehört dazu) Mehrschrittverfahren.*

8.2 Einschrittverfahren

Definition und Bemerkung 8.2.1 (Runge–Kutta–Verfahren)

Zur Berechnung des nächsten Näherungswertes $\vec{y}_{k+1}$ für $\vec{y}(x_{k+1})$ ausgehend vom aktuellen *Wert $\vec{y}_k$ werden im Intervall $[x_k, x_{k+1}]$ — die Länge des Interalls sei $h = x_{k+1} - x_k$ — die m Stützstellen $x_{kj} = x_k + a_j \cdot h$ für das Interpolationspolynom zugrunde gelegt.*

Die systematische Berechnung des Interpolationspolynoms $\vec{\varphi}(x)$ (mit zum Teil noch unbekann-

ten Stützwerten) und die anschließende näherungsweise Berechnung des Integrals

$$\int_{x_k}^{x_{k+1}} \vec{f}(t,\vec{y}(t))\,dt$$

über dieses Interpolatinspolynom $\vec{\varphi}(x)$ führt auf Gleichungen folgender Form:

$$\begin{aligned}
\mathbf{K}_1 &= \vec{f}\Big(x_k + a_1 h, \vec{y}_k + h \cdot \sum_{j=1}^{m} b_{1j} \cdot \mathbf{K}_j\Big) \\
\mathbf{K}_2 &= \vec{f}\Big(x_k + a_2 h, \vec{y}_k + h \cdot \sum_{j=1}^{m} b_{2j} \cdot \mathbf{K}_j\Big) \\
&\vdots \\
\mathbf{K}_m &= \vec{f}\Big(x_k + a_m h, \vec{y}_k + h \cdot \sum_{j=1}^{m} b_{mj} \cdot \mathbf{K}_j\Big)
\end{aligned} \tag{9}$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \sum_{j=1}^{m} c_j \cdot \mathbf{K}_j$$

mit fest vorgegebenen, von den konkreten Stützstellen (und vom Verfahren) abhängenden Konstanten a_i, c_i und b_{ij}, $1 \le i,j \le m$.
Das auf diesen Formeln beruhende Verfahren heißt **$\mathbf{m}$–stufiges Runge–Kutta–Verfahren**.
Die Koeffizienten, welche dieses Verfahren vollständig beschreiben, werden häufig in einer Tabelle, dem sogenannten **Butcher–Diagramm**, *angegeben:*

$$\begin{array}{c|cccc}
a_1 & b_{11} & b_{12} & \dots & b_{1m} \\
a_2 & b_{21} & b_{22} & \dots & b_{2m} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
a_m & b_{m1} & b_{m2} & \dots & b_{mm} \\
\hline
 & c_1 & c_2 & \dots & c_m
\end{array}$$

Bemerkung 8.2.2

Die bislang kennengelernten Verfahren sind von diesem Typ:

1. *explizites Euler–Verfahren:*

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \underbrace{\vec{f}(x_k,\vec{y}_k)}_{=:\ \mathbf{K}_1}$$

Stufenzahl m ist 1.

$$\mathbf{K}_1 = \vec{f}(x_k + 0 \cdot h, \vec{y}_k + h \cdot 0 \cdot \mathbf{K}_1)$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot 1 \cdot \mathbf{K}_1$$

Butcher–Diagramm für das explizite Euler–Verfahren:

$$\begin{array}{c|c}
0 & 0 \\
\hline
 & 1
\end{array}$$

2. *implizites Euler–Verfahren:*

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \underbrace{\vec{f}(x_{k+1}, \vec{y}_{k+1})}_{=:\ \mathbf{K}_1}$$

Stufenzahl m ist 1.

$$\mathbf{K}_1 = \vec{f}(x_k + 1 \cdot h, \vec{y}_k + h \cdot 1 \cdot \mathbf{K}_1)$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot 1 \cdot \mathbf{K}_1$$

Butcher–Diagramm für das implizite Euler–Verfahren:

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

3. *Trapezregel:*

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \Big(\frac{1}{2} \cdot \underbrace{\vec{f}(x_k, \vec{y}_k)}_{=:\ \mathbf{K}_1} + \frac{1}{2} \cdot \underbrace{\vec{f}(x_{k+1}, \vec{y}_{k+1})}_{=:\ \mathbf{K}_2}\Big)$$

Stufenzahl m ist 2.

$$\begin{aligned} \mathbf{K}_1 &= \vec{f}\Big(x_k + 0 \cdot h, \vec{y}_k + h \cdot (0 \cdot \mathbf{K}_1 + 0 \cdot \mathbf{K}_2)\Big) \\ \mathbf{K}_2 &= \vec{f}\Big(x_k + 1 \cdot h, \vec{y}_k + h \cdot (\tfrac{1}{2} \cdot \mathbf{K}_1 + \tfrac{1}{2} \cdot \mathbf{K}_2)\Big) \end{aligned}$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot (\frac{1}{2} \cdot \mathbf{K}_1 + \frac{1}{2} \cdot \mathbf{K}_2)$$

Butcher–Diagramm für die Trapezregel:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & \frac{1}{2} & \frac{1}{2} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Definition 8.2.3

1. *Ein Runge–Kutta–Verfahren, bei dem die Matrix der Koeffizienten b_{ij} eine* **strikte untere Dreiecksmatrix** *ist:*

$$\begin{array}{c|ccccc} a_1 & 0 & 0 & 0 & \dots & 0 \\ a_2 & b_{21} & 0 & 0 & \dots & 0 \\ a_3 & b_{31} & b_{32} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ a_m & b_{m1} & b_{m2} & \dots & b_{m(m-1)} & 0 \\ \hline & c_1 & c_2 & \dots & c_{m-1} & c_m \end{array}$$

heißt (m–stufiges) **explizites Runge–Kutta–Verfahren**.

Bei den meisten expliziten Runge–Kutta–Verfahren gilt darüber hinaus, dass $a_0 = 0$ *gilt.*

In diesem Fall werden die Einträge gleich 0 *im Butcher–Diagramm fortgelassen, so dass ein solches explizites Runge–Kutta–Verfahren dann durch das Schema*

$$\begin{array}{c|ccccc} a_2 & b_{21} & & & & \\ a_3 & b_{31} & b_{32} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ a_m & b_{m1} & b_{m2} & \dots & b_{m(m-1)} & \\ \hline & c_1 & c_2 & \dots & c_{m-1} & c_m \end{array}$$

beschrieben wird.

2. *Ein Runge–Kutta–Verfahren, bei dem die Matrix der Koeffizienten* b_{ij} *eine* **untere Dreiecksmatrix** *ist:*

$$\begin{array}{c|cccc} a_1 & b_{11} & 0 & \dots & 0 \\ a_2 & b_{21} & b_{22} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 0 \\ a_m & b_{m1} & b_{m2} & \dots & b_{mm} \\ \hline & c_1 & c_2 & \dots & c_m \end{array}$$

heißt (m–stufiges) **diagonal–implizites Runge–Kutta–Verfahren**.

Sind darüberhinaus alls Diagonalelemente b_{jj} *gleich, so spricht man von einem* **einfach diagonal–impliziten Runge–Kutta–Verfahren**.

3. *Ist die Matrix* b_{ij} **keine** *untere Dreiecksmatrix, so spricht man von einem (m–stufigen)* **voll impliziten Runge–Kutta–Verfahren**.

Bemerkung 8.2.4

- *Bei einem* **expliziten** *Runge–Kutta–Verfahren ist das System 9 zur Bestimmung der* $\mathbf{K}$*'s ein explizites Gleichungssystem, d.h.*

  1. *zuerst kann* $\mathbf{K}_1$ *ausgerechnet werden,*
  2. *anschließend kann* $\mathbf{K}_2$ *mit Hilfe von* $\mathbf{K}_1$ *berechnet werden,*
  3. *dann kann* $\mathbf{K}_3$ *mit Hilfe von* $\mathbf{K}_1$ *und* $\mathbf{K}_2$ *berechnet werden*
  4. *usw.*

  Nach der Berechnung aller $\mathbf{K}$*'s wird* $\vec{y}_{k+1}$ *mit Hilfe der* $\mathbf{K}$*'s berechnet.*

- *Bei einem* **impliziten** *Runge–Kutta–Verfahren ist das Gleichungssystem 9 zur Bestimmung der* $\mathbf{K}$*'s ein* **implizites** *Gleichungssystem, welches iterativ zu lösen ist.*

  Hierbei kann

- *entweder das System 9 selbst zur Fixpunktiteration verwendet,*
- *oder (nach Umformung in eine Nullstellengleichung) ein Newton–Verfahren angewendet werden.*

Als Startwert kann man jeweils $\mathbf{K}_j = \vec{f}(x_k, \vec{y}_k)$ *für alle j verwenden.*

Es kann gezeigt werden, dass bei vernünftigen Aufgabenstellungen und hinreichend kleinen Schrittweiten h das System 9 für die $\mathbf{K}$*'s eine eindeutige Lösung hat.*

Definition 8.2.5 (Konsistenzordnung eines Einschrittverfahrens)
Sei $\vec{y}_{k+1}$ *der mittels des Einschrittverfahrens ausgehend von der Stelle* x_k *und exaktem Anfangswert* $\vec{y}_k = \vec{y}(x_k)$ *berechnete Näherungswert für die exakte Lösung* $\vec{y}(x_{k+1})$ *an der Stelle* $x_{k+1} = x_k + h$ *und gilt:*

$$\frac{\vec{y}(x_{k+1}) - \vec{y}_{k+1}}{h} \in \mathcal{O}(h^p)$$

so heißt p die Konsistenzordnung *des Verfahrens und das Verfahren heißt* **konsistent**, *falls* $p \geq 1$ *gilt.*
(Der Bruch in obiger Gleichung ist der durch h geteilte lokale Fehler*, also der Fehler, der im Punkt* x_{k+1} *vorliegt, falls* $\vec{y}_k$ *im Punkt* x_k *noch fehlerfrei war.)*

Definition und Bemerkung 8.2.6 (Konvergenzordnung eines Einschrittverfahrens)
Sei $\vec{y}_n$ *der mittels Intervallaufteilung*

$$x_0 < x_1 < \ldots < x_n = b$$

ausgehend von dem Anfangswert $\vec{y}(x_0) = \vec{y}_0$ *bestimmte Näherungswert für die Lösung* $\vec{y}(x_n) = \vec{y}(b)$*, so heißt p die* **Konvergenzordnung** *des Verfahrens, wenn*

$$(\vec{y}(x_n) - \vec{y}_n) \in \mathcal{O}(h^p)$$

*gilt (*globaler Fehler*).*
Man kann zeigen, dass bei einem Einzelschrittverfahren die Konvergenzordnung und die Konsitenzordnung gleich sind!

Algorithmus 8.2.7 (verbessertes Euler–Verfahren)
Butcher–Diagramm:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \\ \hline & 0 & 1 \end{array}$$

(es handelt sich somit um ein 2*–stufiges* **explizites** *Verfahren!)*
In Formeln:

$$\begin{aligned} \mathbf{K}_1 &= \vec{f}(x_k, \vec{y}_k) \\ \mathbf{K}_2 &= \vec{f}(x_k + \tfrac{h}{2}, \vec{y}_k + \tfrac{h}{2}\mathbf{K}_1) \end{aligned}$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \mathbf{K}_2$$

oder kürzer in einer Formel:

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \vec{f}(x_k + \frac{h}{2}, \vec{y}_k + \frac{h}{2} \cdot \vec{f}(x_k, \vec{y}_k))$$

Die Konvergenzordnung des verbesserten Euler–Verfahrens ist 2*.*

Algorithmus 8.2.8 (Verfahren von Heun)
Butcher–Diagramm:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

(es handelt sich somit um ein **explizites** *Verfahren, die Stufenzahl m ist* 2*!)*
In Formeln:

$$\begin{aligned} \mathbf{K}_1 &= \vec{f}(x_k, y_k) \\ \mathbf{K}_2 &= \vec{f}(x_k + h\ ,\ \vec{y}_k + h \cdot \mathbf{K}_1) \end{aligned}$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot \frac{1}{2} \cdot (\mathbf{K}_1 + \mathbf{K}_2)$$

Die Ordnung des Verfahrens ist 2*.*

Algorithmus 8.2.9 (Klassisches Runge–Kutta–Verfahren)
Butcher–Diagramm:

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

(Wiederum ein explizites *Verfahren, die Stufenzahl m ist* 4*!)*
In Formeln:

$$\begin{aligned} \mathbf{K}_1 &= \vec{f}(x_k, \vec{y}_k) \\ \mathbf{K}_2 &= \vec{f}(x_k + \tfrac{h}{2}\ ,\ \vec{y}_k + \tfrac{h}{2} \cdot \mathbf{K}_1) \\ \mathbf{K}_3 &= \vec{f}(x_k + \tfrac{h}{2}\ ,\ \vec{y}_k + \tfrac{h}{2} \cdot \mathbf{K}_2) \\ \mathbf{K}_4 &= \vec{f}(x_k + h\ ,\ \vec{y}_k + h \cdot \mathbf{K}_3) \end{aligned}$$

und

$$\vec{y}_{k+1} = \vec{y}_k + h \cdot (\frac{1}{6} \cdot \mathbf{K}_1 + \frac{1}{3} \cdot \mathbf{K}_2 + \frac{1}{3} \cdot \mathbf{K}_3 + \frac{1}{6} \cdot \mathbf{K}_4)$$

Das klassische Runge–Kutta–Verfahren hat die Ordnung 4*.*

Beispiel 8.2.10

gegeben: *Anfangswertaufgabe* $y' = \underbrace{\sqrt{x+y}}_{f(x,y)}, \quad y(0) = 1.0$

gesucht: $y(1.0)$

Es soll das klassische Runge–Kutta–Verfahren mit konstanter Schrittweite $h = 0.2$ durchgeführt werden.

k	x_k	y_k	K's
0	0.0000000000	1.0000000000	$K_1 = 1.0000000000$ $K_2 = 1.0954451150$ $K_3 = 1.0997929403$ $K_4 = 1.1916201526$
1	0.2000000000	1.2194032088	$K_1 = 1.1913870944$ $K_2 = 1.2800554356$ $K_3 = 1.2835142198$ $K_4 = 1.3697102076$
2	0.4000000000	1.4756777625	$K_1 = 1.3695538553$ $K_2 = 1.4534899890$ $K_3 = 1.4563745265$ $K_4 = 1.5384903860$
3	0.6000000000	1.7666035383	$K_1 = 1.5383769168$ $K_2 = 1.6187776963$ $K_3 = 1.6212591736$ $K_4 = 1.7002515617$
4	0.8000000000	2.0905602789	$K_1 = 1.7001647799$ $K_2 = 1.7778011016$ $K_3 = 1.7799832553$ $K_4 = 1.8564904874$
5	1.0000000000	2.4463010782	

Der mit dem klassischen Runge–Kutta–Verfahren berechnete Näherungswert lautet:

$$y(1.0) \approx 2.4463010782$$

Algorithmus 8.2.11 (Gauß–Verfahren der Ordnung 4)
Butcher–Diagramm:

$\frac{3-\sqrt{3}}{6}$	$\frac{1}{4}$	$\frac{3-\sqrt{3}}{12}$
$\frac{3+\sqrt{3}}{6}$	$\frac{3+\sqrt{3}}{12}$	$\frac{1}{4}$
	$\frac{1}{2}$	$\frac{1}{2}$

Es handelt sich somit um ein 2–stufiges implitzites Verfahren.
In Formeln:

$$\begin{aligned}
\mathbf{K}_1 &= \vec{f}(x_k + \frac{3-\sqrt{3}}{6} \cdot h \ , \ \vec{y}_k + h \cdot (\frac{1}{4} \cdot \mathbf{K}_1 + \frac{3-2\sqrt{3}}{12} \cdot \mathbf{K}_2)) \\
\mathbf{K}_2 &= \vec{f}(x_k + \frac{3+\sqrt{3}}{6} \cdot h \ , \ \vec{y}_k + h \cdot (\frac{3+2\sqrt{3}}{12} \cdot \mathbf{K}_1 + \frac{1}{4} \cdot \mathbf{K}_2))
\end{aligned}$$

und

$$\vec{y}_{k+1} = \vec{y}_k + \frac{h}{2} \cdot (\mathbf{K}_1 + \mathbf{K}_2)$$

Algorithmus 8.2.12 (Gauß–Verfahren der Ordnung 6)
Butcher–Diagramm:

$\frac{1}{2} - \frac{\sqrt{15}}{10}$	$\frac{5}{36}$	$\frac{2}{9} - \frac{\sqrt{15}}{15}$	$\frac{5}{36} - \frac{\sqrt{15}}{30}$
$\frac{1}{2}$	$\frac{5}{36} + \frac{\sqrt{15}}{24}$	$\frac{2}{9}$	$\frac{5}{36} - \frac{\sqrt{15}}{4}$
$\frac{1}{2} + \frac{\sqrt{15}}{10}$	$\frac{5}{36} + \frac{\sqrt{15}}{30}$	$\frac{2}{9} + \frac{\sqrt{15}}{15}$	$\frac{5}{36}$
	$\frac{5}{18}$	$\frac{4}{9}$	$\frac{5}{18}$

Es handelt sich somit um ein 3–stufiges implitzites Verfahren.
(Auf eine explizite Angabe der Formelm wird hier verzichtet!)

8.3 Schrittweitensteuerung

Ist eine Anfangswertaufgabe

$$\vec{y}' = \vec{f}(x, \vec{y}) \quad , \quad \vec{y}(x_0) = \vec{y}_0$$

gegeben und ist der Funktionswert $\vec{y}(b)$ der Lösung $\vec{y}(x)$ an der Stelle $b > x_0$ gesucht, so muss man, um mit einem numerischen Verfahren die Lösung einigermaßen genau zu bestimmen, das Intervall in Teilintervalle aufteilen:

$$x_0 < x_1 < x_2 < \ldots < x_n = b$$

um dann ausgehend vom Funktionswert $\vec{y}_0$ an der Anfangsstelle x_0 schrittweise über $\vec{y}_1 \approx \vec{y}(x_1)$, $\vec{y}_2 \approx \vec{y}(x_2)$, ... zum gewünschten Näherungswert $\vec{y}_n \approx \vec{y}(x_n) = \vec{y}(b)$ zu gelangen.
Wählt man (zu) viele kleine Teilintervalle, so ist das Ergebnis zwar sehr genau, dafür ist der Rechenaufwand aber sehr hoch.
Wählt man (zu) wenige große Teilintervalle, ist der Rechenaufwand sehr gering, dafür dürfte das Ergebnis aber zu ungenau sein.
Wie bei der numerischen Integration (Adaptive Quadratur) ist es anzustreben, bei einer vorgegebenen Genauigkeitsschranke $(b - x_0) \cdot \varepsilon > 0$ die Schrittweiten der Intervallaufteilung jeweils so groß zu wählen, dass die vorgegebene Genauigkeitsschranke gerade noch unterboten wird.
Tunlichst sollte die Wahl der "passenden" Schrittweiten dem numerischen Verfahren selbst überlassen werden, so dass das Verfahren selbständig die lokalen Schrittweiten den lokalen Bedürfnissen anpasst.
Bei den *adaptiven* Verfahren ist es in jedem Berechnungsschritt (Berechnung von $\vec{y}_{k+1}$ ausgehend von x_k und $\vec{y}_k$) erforderlich, den hierbei gemachten *lokalen Fehler* zu schätzen, um entscheiden zu können,

- ob das Ergebnis genau genug ist und zum nächsten Teilintervall übergegangen werden kann

- oder ob das Ergebnis zu ungenau ist und ausgehend von x_k mit einer kleineren Schrittweite neu gerechnet werden muss.

Es gibt unterschiedliche Möglichkeiten, den *lokalen Fehler* zu schätzen:

Bemerkung 8.3.1 (Schätzung des lokalen Fehlers)

1. *Rechnung mit einem Verfahren und unterschiedlichen Schrittweiten:*

   Sei V ein Verfahren mit Konsistenzordnung (=Konvergenzordnung) q, weiter sei

   - $\tilde{\mathbf{Y}}$ *der mittels dieses Verfahrens und Schrittweite* $h = x_{k+1} - x_k$ *in einem Schritt mit Schrittweite h ausgehend von* $(x_k, \vec{y}_k)$ *berechnete Näherungswert für* $\vec{y}(x_{k+1})$*,*
   - $\mathbf{Y}$ *der mittels dieses Verfahrens mit Schrittweite* $\frac{h}{2}$ *in zwei Schritten ausgehend von* $(x_k, \vec{y}_k)$ *berechnete Näherungswert für* $\vec{y}(x_{k+1})$

   so ist

   $$\frac{1}{2^q - 1}(\mathbf{Y} - \tilde{\mathbf{Y}})$$

   eine Schätzung für den lokalen Fehler der (besseren) Näherung $\mathbf{Y}$ *und*

   $$\mathcal{F} = \frac{1}{2^q - 1}||(\mathbf{Y} - \tilde{\mathbf{Y}})||$$

   ist ein Schätzwert für den Betrag eben dieses (lokalen) Fehlers.

2. *Rechnung mit zwei unterschiedlichen Verfahren und gleicher Schrittweite:*

   Sei V ein Verfahren der Konsistenzordnung (=Konvergenzordnung) q und $\tilde{V}$ *ein Verfahren mit Konsistenzordnung mindestens* $q + 1$*, weiter sei*

   - $\mathbf{Y}$ *der ausgehend von* $(x_k, \vec{y}_k)$ *mit dem Verfahren V und Schrittweite* $h = x_{k+1} - x_k$ *berechnete Näherungswert für* $\vec{y}(x_{k+1})$*, und*
   - $\tilde{\mathbf{Y}}$ *der ausgehend von* $(x_k, \vec{y}_k)$ *mit dem Verfahren* $\tilde{V}$ *und derselben Schrittweite h berechnete Näherungswert für* $\vec{y}(x_{k+1})$*,*

   so ist die Differenz

   $$\mathbf{Y} - \tilde{\mathbf{Y}}$$

   eine Schätzung für den lokalen Fehler der (schlechteren) Näherung $\mathbf{Y}$ *und*

   $$\mathcal{F} = ||\mathbf{Y} - \tilde{\mathbf{Y}}||$$

   ist ein Schätzwert für den Betrag eben dieses (lokalen) Fehlers.

Ein möglicher Ansatz zur Schrittweitensteuerung ist dann (nach dem Buch *Numerik–Algorithmen* von Frau Prof. Dr. G. Engeln–Müllges) der folgende:

Algorithmus 8.3.2 (Schrittweitensteuerung)
Es sei eine Fehlerschranke $(b - x_0) \cdot \varepsilon > 0$ vorgegeben.
In folgendem Algorithmus wird sichergestellt, dass in jedem Integrationsschritt der Intervalllänge h der lokale Fehler ungefähr gleich $h \cdot \varepsilon$ ist.

Ausgehend vom aktuellen Punkt $(x_k, \vec{y}_k)$ und Schrittweite $h = x_{k+1} - x_k$ werden zwei Näherungswerte $\mathbf{Y}$ bzw. $\tilde{\mathbf{Y}}$ für den nächsten Wert $\vec{y}(x_{k+1})$ berechnet,
entweder

- *mit ein und demselben Verfahren V der Ordnung q und den beiden Schrittweiten h (Näherung $\tilde{\mathbf{Y}}$ und $\frac{h}{2}$ (Näherung $\mathbf{Y}$)*
- *oder mit zwei Verfahren V der Ordnung q (Näherung $\mathbf{Y}$) und $\tilde{V}$ der Ordnung mindestens $q + 1$ (Näherung $\tilde{\mathbf{Y}}$)*

und der Betrag des hierbei gemachten Fehlers $\mathcal{F}$ nach Bemerkung 8.3.1 geschätzt.
Mit dieser Fehlerschätzung $\mathcal{F}$ wird die folgende Steuergröße berechnet:

$$S = \sqrt[q]{\frac{h \cdot \varepsilon}{\mathcal{F}}}$$

- *Ist $S \geq 1$, so wird der Fehler als klein genug angesehen, der Wert $\mathbf{Y}$ wird als Näherungswert für $y(x_{k+1})$ akzeptiert, man geht zum nächsten x–Wert x_{k+1} über und wählt als neue Schrittweite:*

$$h = \min\{\, 2,\, S \,\} \cdot h$$

  d.h. die Schrittweite wird nächsten Schritt geringfügig vergrößert,
- *Ist $S < 1$, so wird der Fehler als zu groß angesehen, der Schritt ausgehend von x_k muss mit der kleineren Schrittweite:*

$$h = \max\{\, \frac{1}{2},\, S \,\} \cdot h$$

  und neuem $x_{k+1} = x_k + h$ wiederholt werden.

8.3.1 Einbettungsformeln

erleichtern die Schrittweitensteuerung, wenn pro Schritt mit zwei unterschiedlichen Verfahren Näherungen berechnet werden.
Hierbei ist es hilfreich, wenn die beiden verwendeten Verfahren V und $\tilde{V}$ so in einem Zusammenhang stehen, dass etwa die Zwischen– und Hilfswerte, welche zur Ausführung eines Schrittes mit dem Verfahren V berechnet werden, bei der Durchführung des Schrittes mit Verfahren $\tilde{V}$ wiederverwendet werden können, so dass für $\tilde{V}$ nicht alles neu berechnet werden muss.
Solche Paare zusammenhängender Verfahren V und $\tilde{V}$ heißen *Einbettungsformeln*.
Einbettungsformeln sind vor allem bei expliziten Runge–Kutta–Verfahren weit verbreitet.

Beispiele hierzu sind das verbesserte Polygonzugverfahren (Ordnung 2, übernimmt die Rolle von Verfahren V) mit dem Koeffizientenschema:

$$\begin{array}{c|cc} \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array}$$

ausgeschrieben:

$$\begin{aligned} K_1 &= f(x_i, y_i) \\ K_2 &= f(x_i + \tfrac{1}{2} \cdot h, y_i + h \cdot \tfrac{1}{2} \cdot K_1) \end{aligned}$$

und

$$Y = y_i + h \cdot (0 \cdot K_1 + 1 \cdot K_2)$$

und folgendes Runge–Kutta–Verfahren der Ornung 3 (übernimmt die Rolle von Verfahren $\tilde{V}$), Koeffizientenschema:

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

ausgeschrieben:

$$\begin{aligned} K_1 &= f(x_i, y_i) \\ K_2 &= f(x_i + \tfrac{1}{2} \cdot h, y_i + h \cdot \tfrac{1}{2} \cdot K_1) \\ K_3 &= f(x_i + 1 \cdot h, y_i + h \cdot (-1 \cdot K_1 + 2 \cdot K_2)) \end{aligned}$$

und

$$\tilde{Y} = y_i + h \cdot (\tfrac{1}{6} \cdot K_1 + \tfrac{2}{3} \cdot K_2 + \tfrac{1}{6} \cdot K_3)$$

Wie man sieht, werden in beiden Verfahren V und $\tilde{V}$ die Größen K_1 und K_2 gleich berechnet und verwendet — bei Verfahren $\tilde{V}$ kommt zusätzlich ein K_3 hinzu.
Bei den Formeln für die Näherungswerte Y bzw. $\tilde{Y}$ treten dann unterschiedliche Koeffizienten auf.
Man kann die Koeffizienten beider Verfahren in <u>ein</u> Schema hineinschreiben:

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

Bemerkung 8.3.3 (Koeffizientenschema für Einbettungsformeln)
Das Koeffizientenschema einer allgemeinen (expliziten) Einbettungsformel lautet:

$$\begin{array}{c|ccccc} a_2 & b_{21} & & & & \\ a_3 & b_{31} & b_{32} & & & \\ \vdots & \vdots & \ddots & \ddots & & \\ a_m & b_{m1} & \dots & \dots & b_{m,m-1} & \\ \hline V & c_1 & c_2 & \dots & c_{m-1} & c_m \\ \tilde{V} & \tilde{c}_1 & \tilde{c}_2 & \dots & \tilde{c}_{m-1} & \tilde{c}_m \end{array}$$

(i. Allg. sind die letzten Koeffizienten c_m, c_{m-1},... gleich 0)

In jedem Schritt mit diesen Verfahren werden die Hilfsgrößen wie folgt berechnet:

$$\begin{aligned} K_1 &= f(x_i, y_i) \\ K_2 &= f(x_i + a_2 \cdot h, y_i + h \cdot b_{21} \cdot K_1) \\ \vdots &\quad \vdots \quad \vdots \\ K_m &= f(x_i + a_m \cdot h, y_i + h \cdot \sum_{l=1}^{m-1} b_{ml} K_l) \end{aligned}$$

und anschließend mit denselben Hilfsgrößen die beiden Näherungen:

$$Y = y_i + h \cdot \sum_{i=1}^{m} c_i \cdot K_i$$

und

$$\tilde{Y} = y_i + h \cdot \sum_{i=1}^{m} \tilde{c}_i \cdot K_i$$

berechnet.

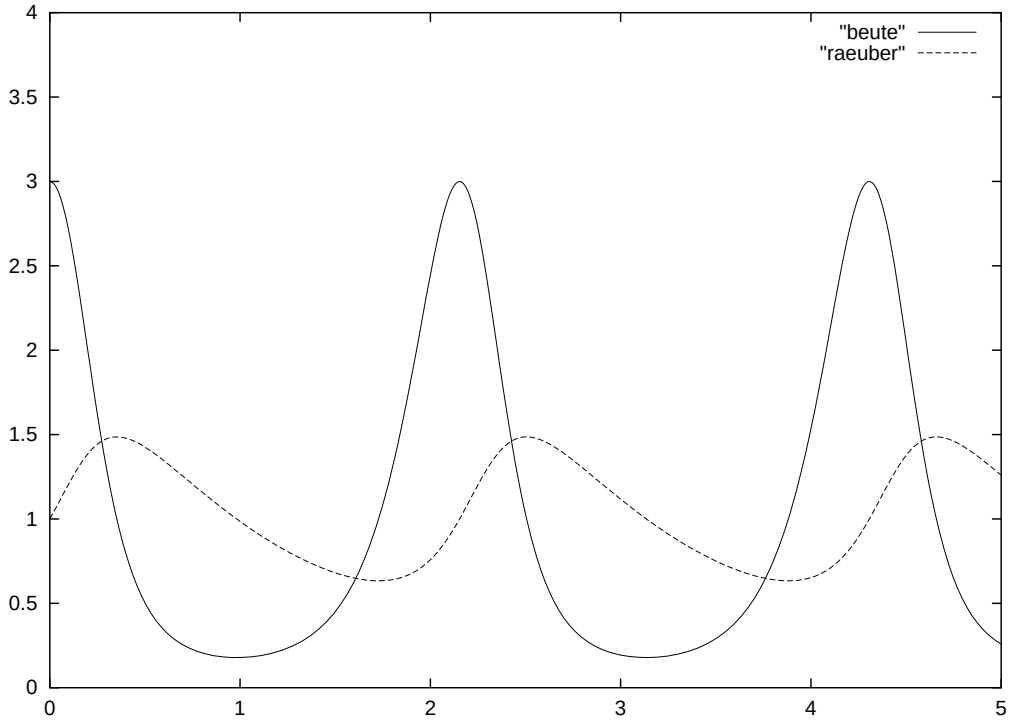
Bemerkung 8.3.4 (Einbettungsformel von England)
Eine bekanntes Formelpaar von diesem Typ sind die England–Formeln mit Ordnung $q = 4$ für V und 5 für $\tilde{V}$. Sie haben folgendes Koeffizientenschema:

$\frac{1}{2}$	$\frac{1}{2}$					
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$				
1	0	-1	2			
$\frac{2}{3}$	$\frac{7}{27}$	$\frac{10}{27}$	0	$\frac{1}{27}$		
$\frac{1}{5}$	$\frac{28}{625}$	$-\frac{125}{625}$	$\frac{546}{625}$	$\frac{54}{625}$	$-\frac{378}{625}$	
V	$\frac{1}{6}$	0	$\frac{4}{6}$	$\frac{1}{6}$		
$\tilde{V}$	$\frac{14}{336}$	0	0	$\frac{35}{336}$	$\frac{162}{336}$	$\frac{125}{336}$

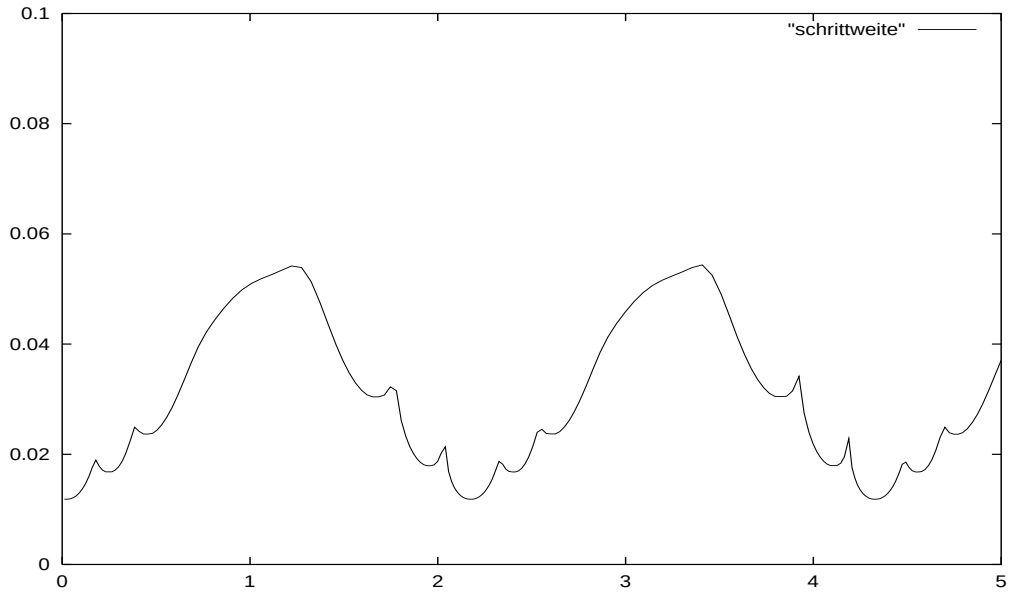
Beispiel 8.3.5 (Beispiel zur Schrittweitensteuerung)
Beute–Räuber–Differentialgleichungssystem (Lotka–Volterra):

$$\begin{aligned} y_1'(t) &= a \cdot y_1(t) \cdot (1 - y_2(t)) \\ y_2'(t) &= y_2(t) \cdot (y_1(t) - 1) \end{aligned}$$

$y_1(t) =$ "Beute" und $y_2(t) =$ "Räuber".
Zeitliche Entwicklung der Populationen, berechnet mit der Formel von England ($a = 10.0$, $y_1(0) = 3.0$, $y_2(0) = 1.0$):



Entwicklung der Schrittweiten:



8.4 Stabilität, steife Differentialgleichungen

In der Praxis auftretende Differentialgleichungssysteme haben häufig die Eigenschaft, dass sich die Lösungen aus "*abklingenden*" *e*–Funktionen $e^{\lambda t}$ zusammensetzen mit $\lambda \in \mathbb{C}$ und negativem Realteil $\mathrm{Re}(\lambda) < 0$.
Beachte: ist $\lambda = -a + b \cdot i$ mit $a, b \in \mathbb{R}$ und $a > 0$, so ist:

$$e^{\lambda t} = e^{-at} \cdot (\cos(bt) + i \cdot \sin(bt))$$

eine oszillierende, aber wegen $a > 0$ betragsmäßig abklingende (komplexe) Funktion.
Man möchte natürlich, dass auch eine numerisch bestimmte Näherungslösung für das Differentialgleichungssystem genauso "abklingende" Bestandteile hat.
Hierzu sind je nach Verfahren an die zu verwendende Schrittweite *h* Bedingungen zu knüpfen!

Definition 8.4.1 (Testproblem)
Das Anfangswertproblem

$$y'(x) = \lambda \cdot y(x) \quad , \quad y(0) = 1$$

($\lambda \in \mathbb{C}$, $Re(\lambda) < 0$) heißt **Testproblem**.

Die exakte Lösung des Testproblems lautet bekanntlich

$$y(x) = e^{\lambda x}$$

Da $Re(\lambda) < 0$ handelt es sich bei der Lösung um eine betragsmäßig monoton fallende Funktion. Ist $x_{k+1} = x_k + h$, so gilt für die exakte Lösung:

$$y(x_{k+1}) = e^{\lambda x_{k+1}} = e^{\lambda(x_k + h)} = e^{\lambda h} \cdot e^{\lambda x_k} = e^{\lambda h} \cdot y(x_k)$$

d.h. die exakte Lösung $y(x_{k+1})$ an der Stelle x_{k+1} erhält man aus der exakten Lösung $y(x_k)$ an der Stelle x_k durch Multiplikation mit dem Faktor $e^{\lambda h}$.
Wegen $Re(\lambda) < 0$ ist der Betrag dieses Faktors $e^{\lambda h}$ kleiner als 1.

Bemerkung 8.4.2 (numerische Verfahren für die Testaufgabe)
Wendet man ein Runge–Kutta–Verfahren auf die Testdifferentialgleichung $y' = \lambda y$ an, so geht ebenfalls der Näherungswert y_{k+1} für $y(x_{k+1})$ durch Multiplikation mit einem Faktor $F(\lambda h)$ aus dem Näherungswert y_k für $y(x_k)$ hervor, d.h. es gibt eine (komplexe) Funktion $F(z)$, so dass dieser Faktor gleich $F(\lambda h)$ ist:

$$y_{k+1} = F(\lambda h) \cdot y_k$$

Diese (komplexe) Funktion $F(z)$ heißt **Stabilitätsfunktion** *des numerischen Verfahrens.*

Bemerkung 8.4.3 (Explizites Euler-Verfahren für das Testproblem)
Wendet man das explizite Euler–Verfahren mit Schrittweite h auf das Testproblem an, so erhält man

$$y(x_{k+1}) \approx y_{k+1} = y_k + h \cdot f(x_k, y_k) = y_k + h\lambda \cdot y_k = (1 + \lambda h) \cdot y_k$$

d.h. die Stabilitätsfunktion des expliziten Euler–Verfahrens ist $F(z) = 1 + z$.
*Damit das qualitative Verhalten der exakten Lösung (*betragsmäßig abklingend*) auch für die numerisch bestimmte Näherungslösung gilt, ist zu fordern, dass auch der beim Euler–Verfahren verwendete Faktor* $F(\lambda h) = (1 + \lambda h)$ *betragsmäßig kleiner als* 1 *ist, es ist also zu fordern:*

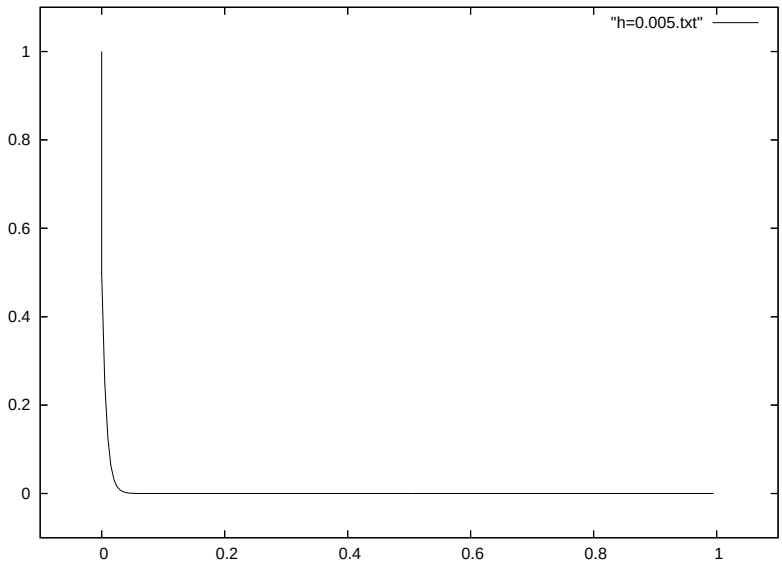
$$|F(\lambda h)| < 1 \quad \textit{mit} \quad F(z) = 1 + z$$

Ist diese (vom Wert λ *abhängende) Bedingung für die verwendete Schrittweite* h *verletzt, so kommt mit dem Euler–Verfahren ein unsinniges Ergebnis heraus!*

Beispiel 8.4.4 *Es wird das Testproblem mit* $\lambda = -100$ *und das explizite Euler–Verfahren zugrundegelegt.*
Die exakte Lösung $y(x) = e^{-100x}$ *ist eine extrem schnell abklingende e–Funktion.*
Die "Stailitätsbedingung" für das Euler–Verfahren $|1 + \lambda h| < 1$ *führt mit* $\lambda = -100$ *auf*

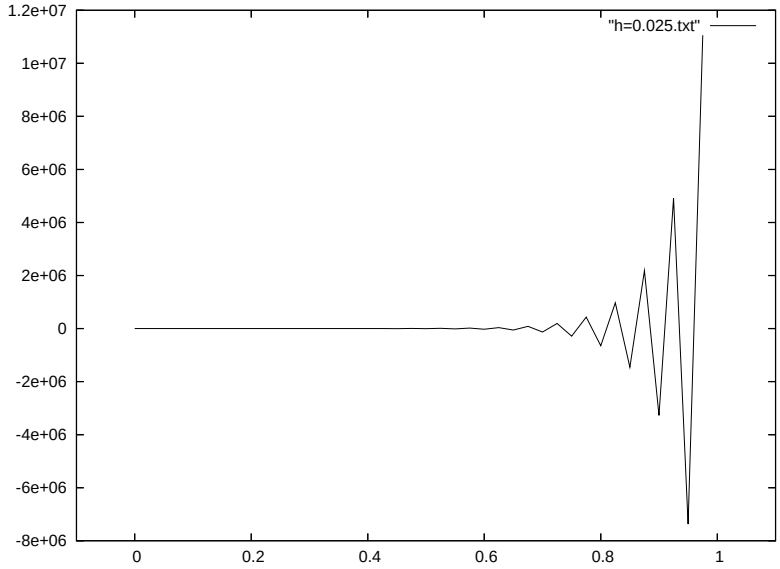
$$h < 0.02$$

- *Wählt man die Schrittweite* $h = 0.005$*, so ist diese Bedingung erfüllt und die für das Intervall* $[0, 1]$ *mit dem Euler–Verfahren berechneten Lösung sieht wie folgt aus:*



Wie man sieht, stimmt die Lösung mit der exakten Lösung sehr gut überein.

- *Wählt man als Schrittweite* $h = 0.025$*, so ist diese Bedingung verletzt und die für das Intervall* $[0, 1]$ *mit dem Euler–Verfahren berechnete Näherungslösung*



hat mit der exakten Lösung nicht viel zu tun.

Bemerkung 8.4.5 (Klassisches Runge–Kutta–Verfahren für das Testproblem)
Wendet man das klassische Runge–Kutta–Verfahren auf die Testdifferentialgleichung $y' = \lambda y$ *an, so erhält man die Formel:*

$$y_{k+1} = \left(1 + (\lambda h) + \frac{(\lambda h)^2}{2} + \frac{(\lambda h)^3}{3!} + \frac{(\lambda h)^4}{4!}\right) \cdot y_k$$

D.h. die Stabilitätsfunktion für das klassiche Runge–Kutta–Verfahren lautet:

$$F(z) = 1 + z + \frac{z^2}{2} + \frac{z^3}{3!} + \frac{z^4}{4!}$$

Um auch hier abklingendes *Verhalten der Näherungslösung zu bekommen, muss für die zu verwendende Schrittweite h*

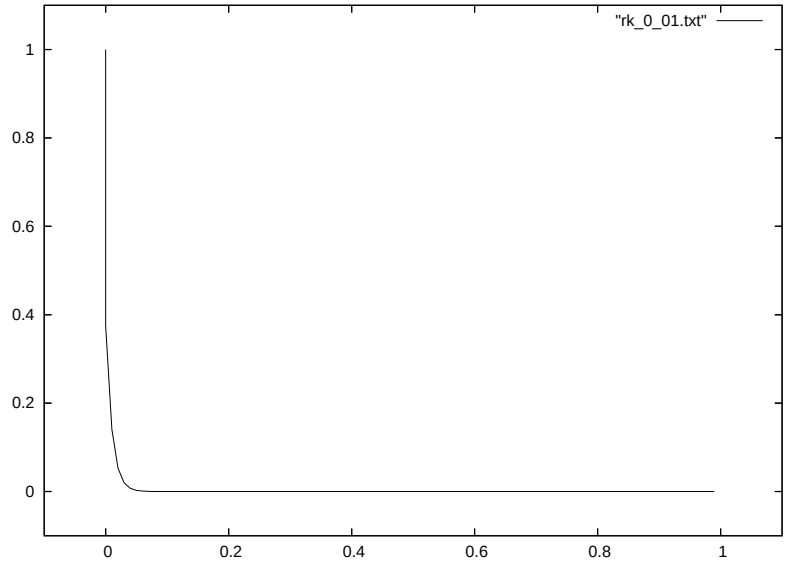
$$|F(\lambda h)| < 1$$

gefordert werden.

Beispiel 8.4.6 *Es wird das Testproblem mit* $\lambda = -100$ *und das explizite klassiche Runge–Kutta–Verfahren zugrundegelegt.*
Die exakte Lösung $y(x) = e^{-100x}$ *ist eine extrem schnell abklingende e–Funktion.*
Die "Stailitätsbedingung" für das klassiche Runge–Kutta–Verfahren führt mit $\lambda = -100$ *auf*

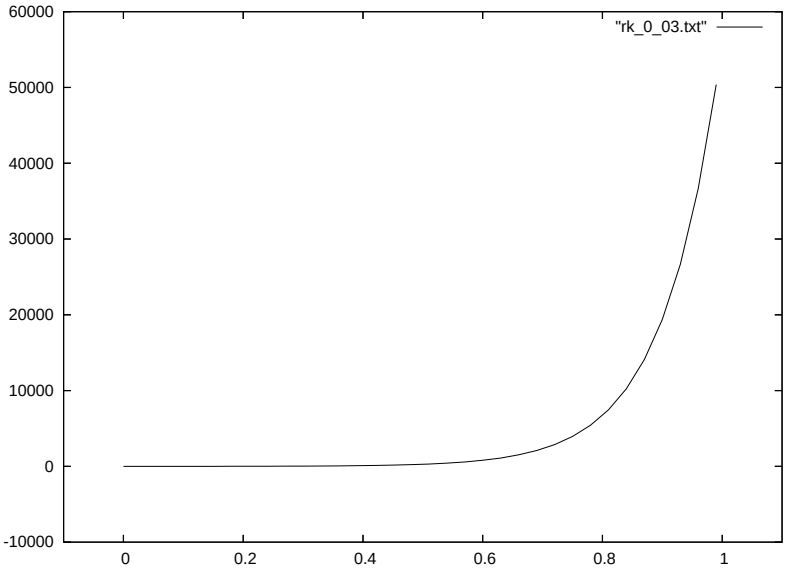
$$h < 0.0278$$

- *Wählt man die Schrittweite* $h = 0.01$*, so ist diese Bedingung erfüllt und die für das Intervall* $[0, 1]$ *mit dem klassichen Runge–Kutta–Verfahren berechnete Lösung sieht wie folgt aus:*



Wie man sieht, stimmt die Lösung mit der exakten Lösung sehr gut überein.

- *Wählt man als Schrittweite* $h = 0.03$*, so ist diese Bedingung verletzt und die für das Intervall* $[0, 1]$ *mit dem klassichen Runge–Kutta–Verfahren berechnete Näherungslösung*



hat mit der exakten Lösung nicht viel zu tun.

Bemerkung 8.4.7 *In den Anwendungen bei Differentialgleichungsen ist das Problem, dass sich die Lösungen häufig aus abklingenden e–Funktionen mit unterschiedlichen "Abklingkonstanten"* λ_1 *und* λ_2 *zusammensetzen, etwa:*

$$y(t) = c_1 \cdot \underbrace{e^{-1 \cdot t}}_{y_1(t)} + c_2 \cdot \underbrace{e^{-100 \cdot t}}_{y_2(t)} + \dots$$

hier hat man $\lambda_1 = -1$ *und* $\lambda_2 = -1000$*.*

Im Vergleich zu $y_1(t)$ ist der Bestandteil $y_2(t)$ "extrem schnell abklingend" und schon für kleine Werte für t spielt $y_2(t)$ in der Lösung $y(t)$ keinerlei Rolle mehr.
Damit der von $\lambda_2 = -1000$ herrührende Bestandteil auch in der numerischen Lösung keine Rolle spielt, muss dennoch für dass gewählte Verfahren mit Stabilitätsfunktion $F(z)$:

$$|F(\lambda_2 h)| < 1$$

gelten, ansonsten sind die Ergebnisse unbrauchbar.

Definition 8.4.8 (steifes Differentialgleichungssystem)
Ein Differentialgleichungssystem heißt **steif***, falls in der Lösung abklingende e–Funktionen mit stark unterschiedlichen Abklingkonstanten λ_k, $k = 1, 2, \ldots$ beteiligt sind.*
Quantitativ: die Zahl

$$St = \frac{\max\{|Re(\lambda_k)|\}}{\min\{|Re(\lambda_k)|\}}$$

heißt **Steifheitsmaß***.*
Das System heißt **steif***, falls das Steifheitsmaß St größer als 1000 ist.*

Beispiel 8.4.9 *In der Lösung des Differentialgleichungssystems:*

$$\begin{pmatrix} y_1' \\ y_2' \end{pmatrix} = \begin{pmatrix} -500.5 & 499.5 \\ 499.5 & -500.5 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} + \begin{pmatrix} 50\sin(100t) \\ 3 \end{pmatrix}$$

mit Anfangswerten $y_1(0) = -1$ und $y_2(0) = 2$ tauchen die e–Funktionen $e^{-1 \cdot t}$ (also $\lambda_1 = -1$) und $e^{-1000 \cdot t}$ (also $\lambda_2 = -1000$) auf.
Löst man dieses System mit dem Euler–Cauchy–Polygonzugverfahren, so muss

- *für den "Hauptbestandteil" $e^{-1 \cdot t}$ der Lösung*

  $$|F(\lambda_1 \cdot h)| < 1 \quad \textit{also} \quad |1 - 1 \cdot h| < 1 \quad \textit{also} \quad h < 2$$

  gelten.

  Damit also der Hauptbestandteil $e^{-1 \cdot t}$ der Lösung qualitativ richtig erfasst wird, reicht eine Schrittweite $h < 2$ aus.

- *für den (eigentlich in der Lösung keine große Rolle spielenden) Bestandteil e^{-1000t} muss*

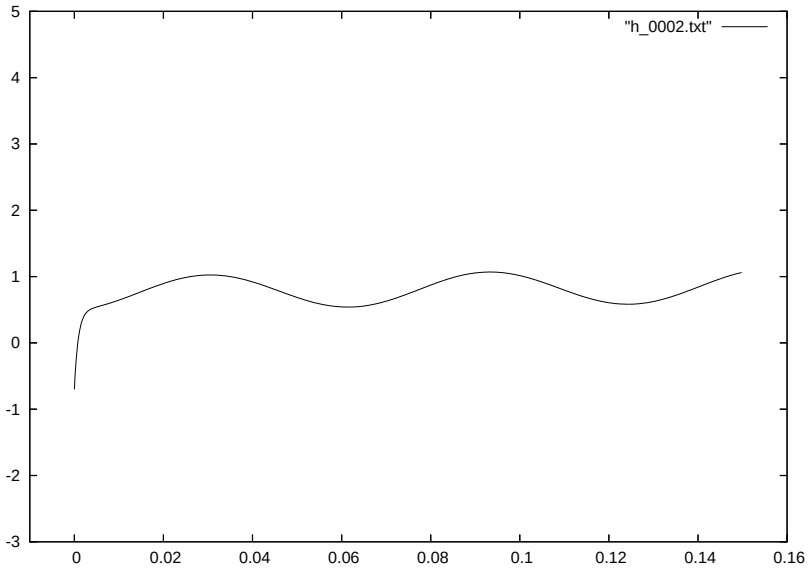
  $$|F(\lambda_2 h)| < 1 \quad \textit{also} \quad |1 - 1000h| < 1 \quad \textit{also} \quad h < 0.002$$

  gelten.

  Obwohl dieser Bestandteil e^{-1000t} in der exakten Lösung keine große Rolle spielt, muss im numerischen Verfahren dennoch eine winzig kleine Schrittweite zugrunde gelegt werden, damit dieser auch in der numerischen Lösung keine Rolle spielt!

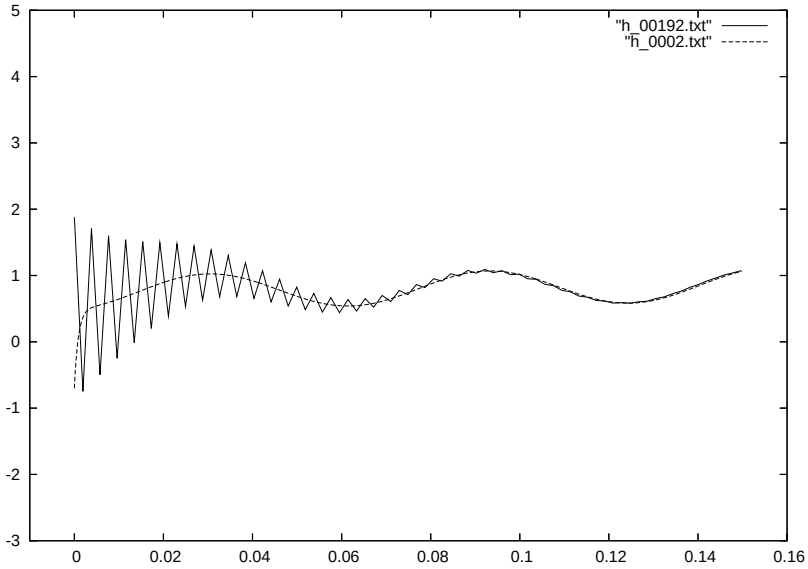
Je nachdem, ob und wie diese Bedingung eingehalten wird, sehen die berechneten Lösungen (etwa für $y_1(t)$) aus:

1. *Rechnung mit $h = 0.0002$, die Stabilitätsbedingung ist bei weitem erfüllt:*



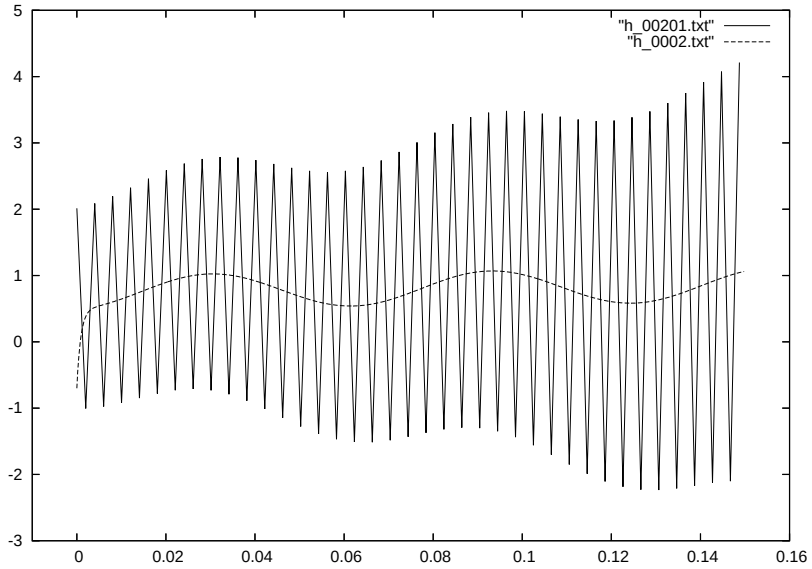
   so in etwa sieht die exakte Lösung für $y_1(t)$ aus!

2. *Rechnung mit $h = 0.00192$, die Stabilitätsbedingung ist so gerade noch erfüllt:*



   erst nach einiger Zeit klingt der von $\lambda_2 = -1000$ herrührende, stark oszilierende Bestandteil der numerischen Lösung ab — die berechnete Lösung nähert sich mehr und mehr der exakten Lösung.

3. *Berechnung mit $h = 0.00201$, die Stabilitätsbedingung ist knapp nicht mehr erfüllt:*



der von $\lambda_2 = -1000$ herrührende, stark oszilierende Bestandteil der numerischen Lösung schaukelt sich auf!

Die ermittelte Lösung stimmt immer weniger mit der exakten Lösung überein.

Definition 8.4.10 (Stabilitätsgebiet, Stabilitätsintervall)
Sei V ein numerisches Einschrittverfahren mit Stabilitätsfunktion $F(z)$, d.h. bei Anwendung auf die Testdifferentialgleichung $y' = \lambda y$ erhält man die Iterationsvorschrift:

$$y_{k+1} = F(\lambda h) \cdot y_k$$

- *Die Menge komplexer Zahlen:*

$$S := \{ z \in \mathbb{C} |\ |F(z)| < 1 \}$$

heißt das **Stabilitätsgebiet** *des Verfahrens V.*

Ist bei der Lösung eines Differentialgleichungssystems eine (abklingende) e–Funktion mit Abklingkonstante $\lambda \in \mathbb{C}$ ($Re(\lambda) < 0$) beteiligt, muss die Schrittweite so bemessen sein, dass $\lambda \cdot h$ im Stabilitätsgebiet liegt!

- *Die Menge reeller Zahlen*

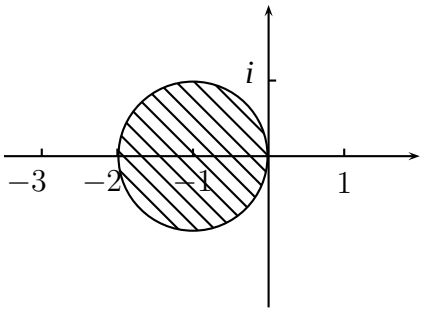
$$I := \{ x \in \mathbb{R} |\ |F(x)| < 1 \}$$

heißt das **Stabilitätsintervall** *des Verfahrens V (es handelt sich meißt wirklich um ein Intervall!).*

Bemerkung 8.4.11 (Stabilitätsgebiet/Stabilitätsindervall einfacher Verfahren)

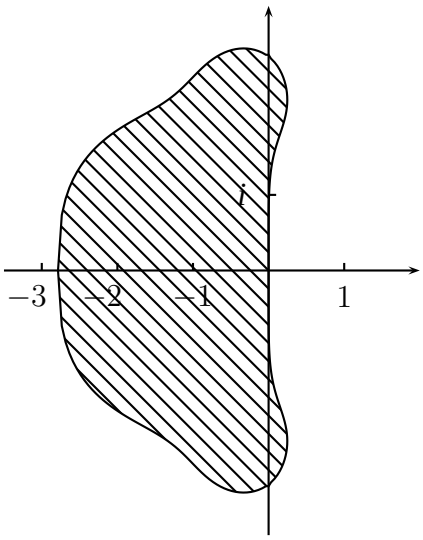
1. *das Stabilitätsgebiet des Euler–Cauchy–Polygonzugverfahrens ist das Innere des Kreises um -1 mit Radius 1:*

Skizze:



Das Stabilitätsintervall ist $(-2, 0)$.

2. *Skizze des Stabilitätsgebietes des klassischen Runge–Kutta–Verfahrens:*



Das Stabilitätsintervall ist $(-2.78, 0)$.

Man kann zeigen:

Bemerkung 8.4.12 *Für alle expliziten Runge–Kutta–Verfahren ist die Stabilitätsfunktion* $F(z)$ *ein Polynom in z, das Stabilitätsgebiet ist ein (links des Nullpunktes) liegendes beschränktes Gebiet und das Stabilitätsintervall ist ein endliches Intervall der Form* $(-a, 0)$ *mit* $a > 0$.

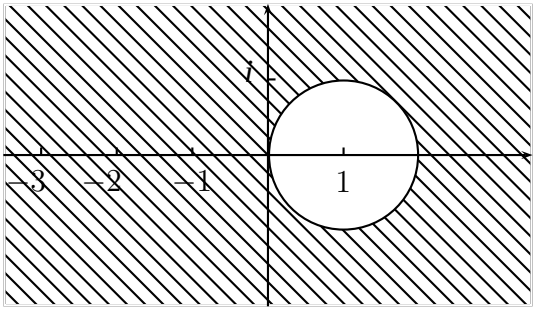
Bemerkung 8.4.13 (Stabilitätsfunktion/–Gebiet/–Interval des impliziten Euler–Verfahrens) *Die Anwendung des impliziten Euler–Verfahrens auf die Testdifferentialgleichung* $y' = \lambda y$ *führt auf die explizite Gleichung*

$$y_{k+1} = \frac{1}{1 - \lambda h} \cdot y_k$$

Die Stabilitätsfunktion lautet somit:

$$F(z) = \frac{1}{1 - z}$$

Das Stabilitätsgebiet ist das Äußere des Kreises um 1 *mit Radius* 1*:*



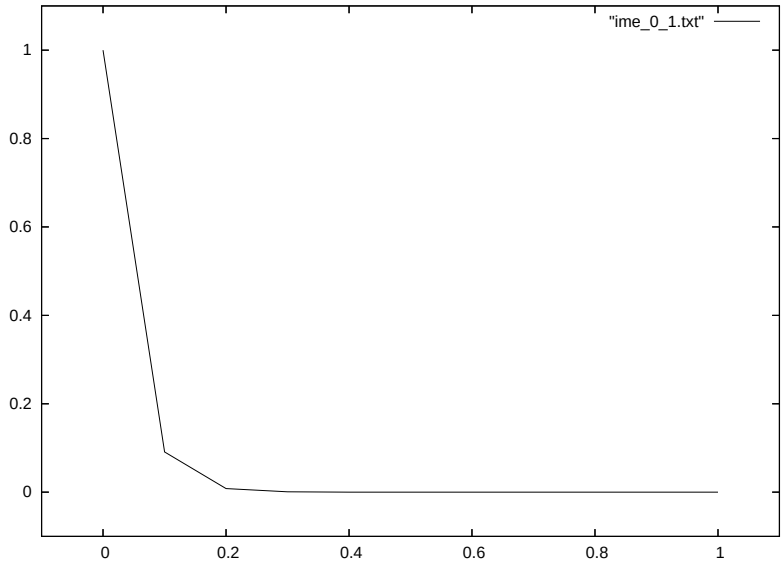
Definition 8.4.14 (A–Stabilität)
Ein Runge–Kutta–Verfahren mit Stabilitätsgebiet S heißt **A–stabil** *oder* **absolut stabil**, *falls*

$$\mathbb{C}^- = \{z \in \mathbb{C} \mid Re(z) < 0\} \subset S$$

gilt.
Bei einem absolut–stabilen Verfahren werden ohne Einschränkung für die Schrittweite alle abklingenden e–Funktionen qualitativ richtig berechnet.

Das implizite Euler–Verfahren ist absolut–stabil.

Beispiel 8.4.15 *Es wird das Testproblem mit* $\lambda = -100$ *und das implizite Euler–Verfahren zugrundegelegt.*
Die exakte Lösung $y(x) = e^{-100x}$ *ist eine extrem schnell abklingende e–Funktion.*
Bereits mit der recht großen Schrittweite $h = 0.1$ *erhält man qualitativ die richtige Lösung:*



Es liegt $\lambda \cdot h = -100 \cdot 0.1 = -10$ *im Stabilitätsgebiet.*
Für jede positive Schrittweite liegt $\lambda \cdot h$ *im Stabilitätsgebiet!*

Bemerkung 8.4.16 (Stabilitätsfunktion/–Gebiet/–Intervall der Trapezregel)
Wird die (eindimensionale) Trapezregel

$$y_{k+1} = y_k + \frac{h}{2} f(x_k, y_k) + \frac{h}{2} f(x_{k+1}, y_{k+1})$$

auf die Testdifferentialgleichung $y' = \lambda y$ angewendet, so erhält man

$$y_{k+1} = y_k + \frac{\lambda h}{2} y_k + \frac{\lambda h}{2} y_{k+1}$$

also die (explizite) Gleichung:

$$y_{k+1} = \frac{1 + \frac{\lambda h}{2}}{1 - \frac{\lambda h}{2}} \cdot y_k$$

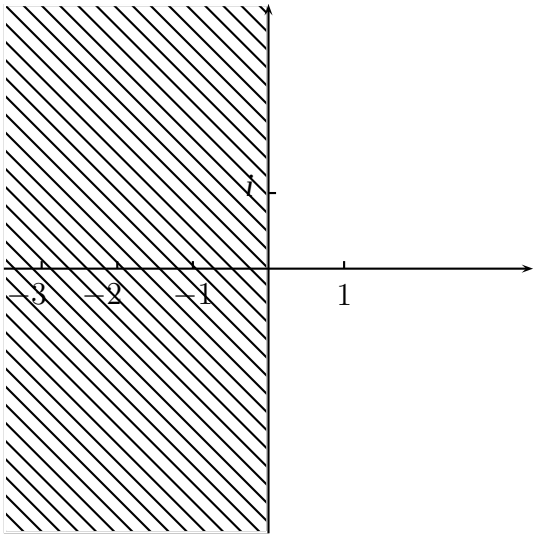
Die Stabilitätsfunktion ist somit

$$F(z) = \frac{1 + \frac{z}{2}}{1 - \frac{z}{2}} = \frac{2 + z}{2 - z}$$

Die Bedingung

$$|F(z)| < 1$$

ist für alle z mit $Re(z) < 1$ erfüllt,
Skizze des Stabilitätsgebietes der Trapezregel:



Somit folgt: auch die (ebenfalls implizite) Trapezregel ist absolut stabil.

Man kann zeigen:

Bemerkung 8.4.17 (Stabilität von Runge–Kutta–Verfahren)

- *Alle* **impliziten** *Runge–Kutta–Verfahren sind absolut stabil.*
- *Ein* **explizites** *Runge–Kutta–Verfahren kann nicht absolut stabil sein!*

Bemerkung 8.4.18

Für steife Differentialgleichungssysteme darf man kein explizites Runge–Kutta–Verfahren verwenden!