Vorlesung: Numerik 1 für Ingenieure

Version 28.5.2018

Michael Karow

11. Vorlesung

Thema: Lösung von ODE (ordinary differential equations)

Numerische Lösung von gewöhnlichen Differentialgleichungen (Kurzeinführung)

Eine explizite gewöhnliche Differentialgleichung (ordinary differential equation, kurz ODE) erster Ordnung ist eine Gleichung der Form $y'(t) = f(t, y(t))$.
Gibt man noch einen Anfangswert $y_0 \in \mathbb{R}^n$ vor, so hat man ein
Anfangswertproblem (AWP):

$$y'(t) = f(t, y(t)), \qquad y(t_0) = y_0, \qquad f : \mathbb{R} \times \mathbb{R}^n \supseteq G \to \mathbb{R}^n. \qquad (*)$$

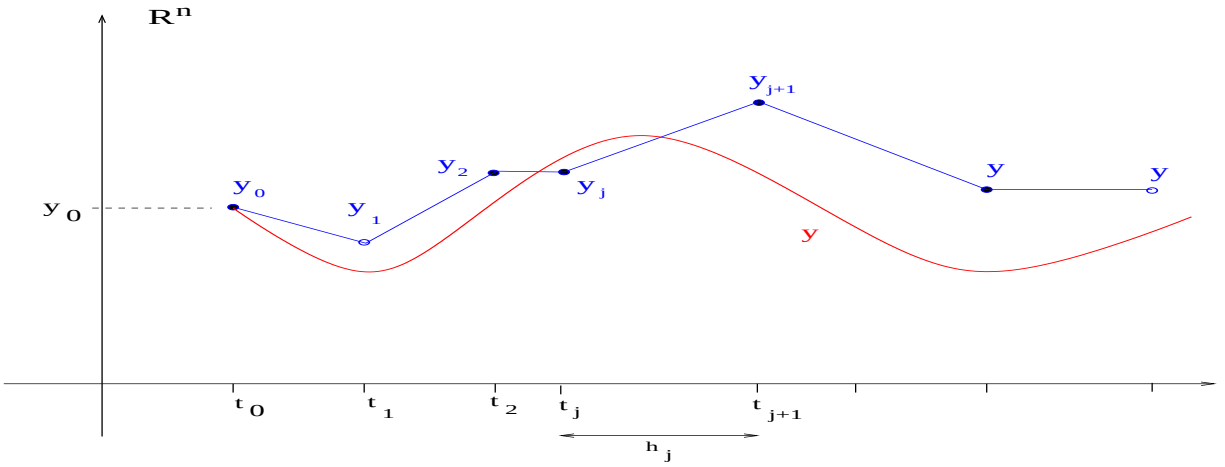
Gesucht ist die Funktion $y : [t_0, t_e] \to \mathbb{R}^n$, welche die Bedingungen $(*)$ erfüllt.

Solche Gleichungen sind oft nur numerisch lösbar. Sei y die Lösung von $(*)$ und sei $t_0 < t_1 < t_2 < \ldots < t_m = t_e$ eine Unterteilung des Intervalls $[t_0, t_e]$. Ziel ist es, für die Folge

$$y(t_0),\ y(t_1),\ y(t_2),\ y(t_3)\ldots,\ y(t_m)$$

auf numerischem Wege eine möglichst gute Näherungsfolge y_j zu berechnen:

$$y(t_0) = y_0,\quad y_1,\quad y_2,\quad y_3,\quad \ldots,\quad y_m$$



Bemerkungen zu gewöhnlichen Differentialgleichungen I

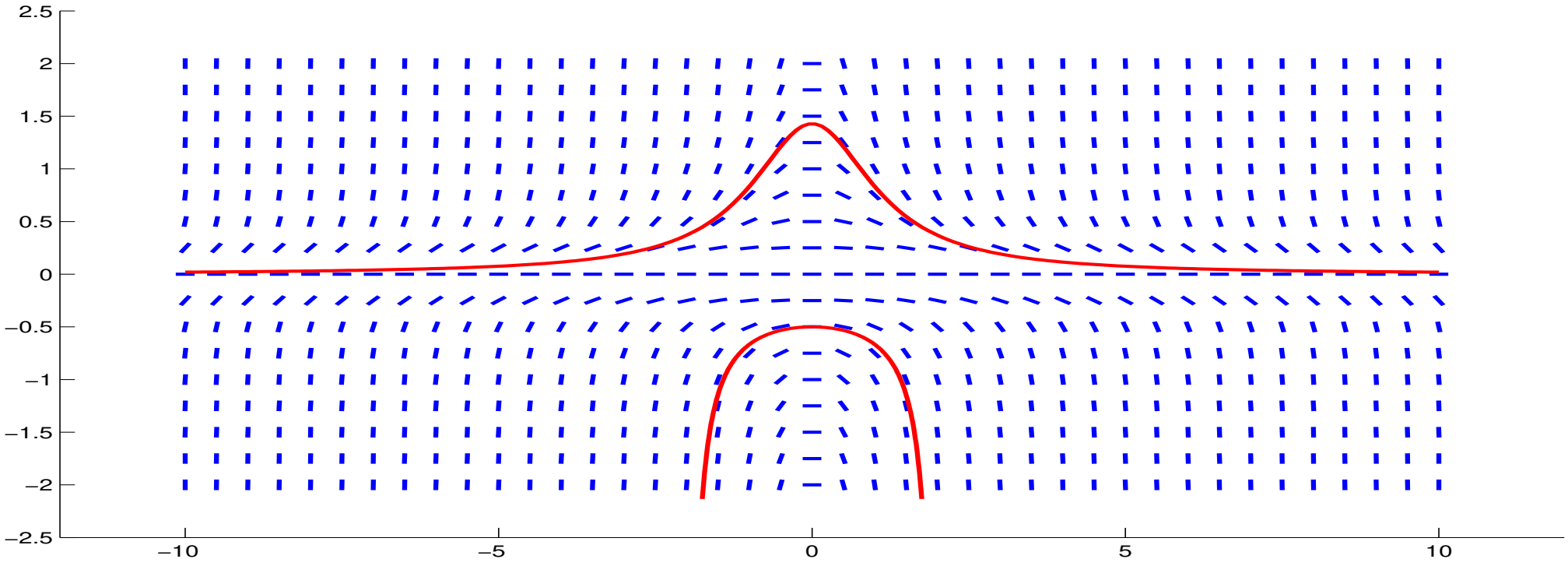
DGL (ODE), (hier für den skalaren Fall):

$$\underset{\uparrow}{y'(t)} = \underset{\uparrow}{f}(t, y(t))$$

Steigung der Lösung vorgegebene Steigung f am Punkt $(t, y(t))$

Die Funktion f ist ein Steigungsfeld (Richtungsfeld).

Beispiel: Das Steigungsfeld $f(t, y) = t\, y^2$ und zwei Lösungen der ODE.



Bemerkungen zu gewöhnlichen Differentialgleichungen II

Das einfachste nichttriviale Anfangwertproblem ist wohl

$$y'(t) = a\, y(t), \qquad y(0) = y_0.$$

Lösung:

$$y(t) = y_0\, e^{at}.$$

Dabei ist $e = 2.718281828459046...$ die **Eulersche Zahl**.

Eine weiteres Beispiel:

$$y'(t) = 1 + y(t)^2, \qquad y(0) = 0.$$

Lösung:

$$y(t) = \tan(t), \qquad t \in (-\pi/4, \pi/4).$$

Die Lösung hat ein **endliches Existenzintervall** ('endliche Lebensdauer').

Bemerkungen zu gewöhnlichen Differentialgleichungen III

Auf den folgenden Seiten ist die Gleichung

$$y'(t) = f(t, y(t))$$

meistens vektorwertig gemeint: $y(t) \in \mathbb{R}^n$.
Schreibt man diese Gleichung komponentenweise, so lautet sie

$$\begin{aligned} y_1'(t) &= f_1(t, y_1(t), y_2(t), \ldots, y_n(t)), \\ y_1'(t) &= f_2(t, y_1(t), y_2(t), \ldots, y_n(t)), \\ &\vdots \\ y_n'(t) &= f_n(t, y_1(t), y_2(t), \ldots, y_n(t)). \end{aligned}$$

Integrale auf den folgenden Seiten sind ebenfalls komponentenweise gemeint:

$$\int_{t_0}^{t_1} f(t, y(t))\, dt = \begin{bmatrix} \int_{t_0}^{t_1} f_1(t, y_1(t), y_2(t), \ldots, y_n(t))\, dt \\ \int_{t_0}^{t_1} f_2(t, y_1(t), y_2(t), \ldots, y_n(t))\, dt \\ \vdots \\ \int_{t_0}^{t_1} f_n(t, y_1(t), y_2(t), \ldots, y_n(t))\, dt \end{bmatrix}.$$

Bemerkungen zu gewöhnlichen Differentialgleichungen IV

Sei $f : G \to \mathbb{R}^n$ stetig. Dann ist die ODE

$$y'(t) = f(t, y(t)) \qquad (*)$$

äquivalent zur Integralgleichung

$$y(t) = y(t_0) + \int_{t_0}^{t} f(\tau, y(\tau))\, d\tau \qquad (**)$$

Sowohl der Beweis des Existenz- und Eindeutigkeitssatz für die Lösungen
eines Anfangswertproblems (siehe nächste Seite)
als auch die Ansätze für numerische Lösungsverfahren basieren auf der Integralgleichung.

Beweis der Äquivalenz:

$(**) \Rightarrow (*)$ folgt aus dem Hauptsatz der Differential- und Integralrechnung.

$(*) \Rightarrow (**)$. Variablenwechsel $t \to \tau$ ergibt zunächst $y'(\tau) = f(\tau, y(\tau))$.
Integrieren ergibt dann

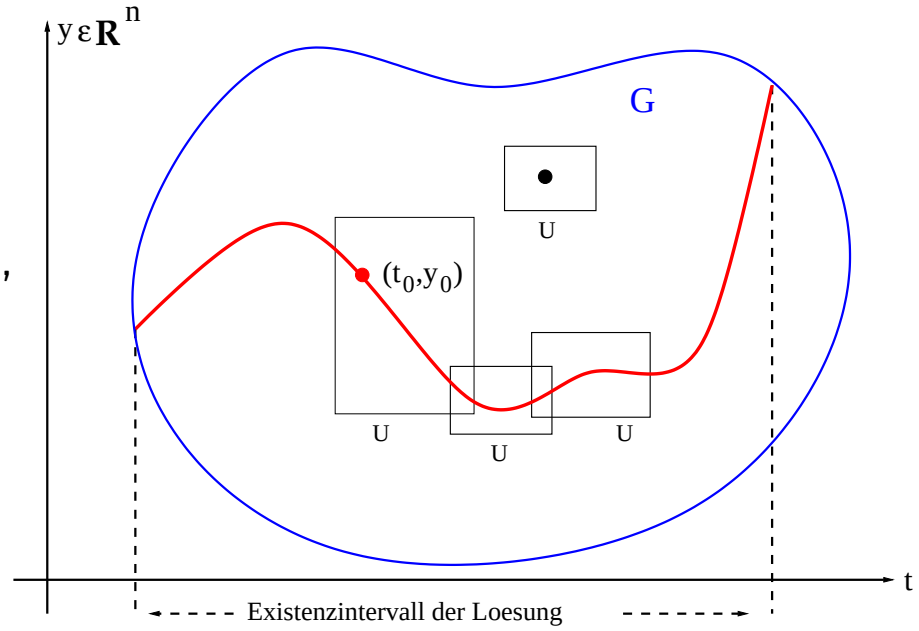
$$y(t) - y(t_0) = \int_{t_0}^{t} y'(\tau)\, d\tau = \int_{t_0}^{t} f(\tau, y(\tau))\, d\tau$$

Nun addiert man $y(t_0)$ auf beiden Seiten und bekommt $(**)$.

Bemerkungen zu gewöhnlichen Differentialgleichungen V

Existenz und Eindeutigkeitssatz von Picard-Lindelöf (1890):

Sei $G \subseteq \mathbb{R} \times \mathbb{R}^n$ ein offenes Gebiet
und sei $f : G \to \mathbb{R}^n$ stetig und lokal **lipschitzstetig** im zweiten Argument$^{(*)}$.
Dann existiert zu jedem Anfangspunkt $(t_0, y_0) \in G$ genau eine Lösung der
ODE $y'(t) = f(t, y(t))$ mit $y(t_0) = y_0$, wobei der Graph der Lösung
nicht innerhalb des Gebiets G endet.



$(*)$ Lokale **Lipschitzstetigkeit** bedeutet:

Um jeden Punkt $(\tilde{t}, \tilde{y}) \in G$ gibt es eine
Umgebung $U \subseteq G$, und eine Konstante $L \geq 0$,
so dass

$$\|f(t, y_1) - f(t, y_2)\| \leq L \, \|y_1 - y_2\|$$

für alle $(t, y_1), (t, y_2) \in U$. Dabei ist $\|\cdot\|$ eine
beliebige Norm auf $\mathbb{R}^n$ und $L \geq 0$ eine von U
abhängige Konstante.

Lipschitzstetigkeit ist erfüllt, wenn f nach y stetig partiell differenzierbar ist.

Beweisidee: Fixpunktiteration mit Funktionenfolge y_k, $k = 0, 1, 2, \ldots$,

$$y_0(t) :\equiv y_0, \qquad y_{k+1}(t) := y_0 + \int_{t_0}^{t} f(\tau, y_k(\tau)) \, d\tau, \quad k = 1, 2, \ldots$$

Lipschitzstetigkeit impliziert Konvergenz $y_k(t) \to y(t)$, wenn $|t - t_0|$ (bzw. U)
hinreichend klein. Globale Lösung entsteht durch 'zusammenkleben' lokaler Lösungen.

Bemerkungen zu gewöhnlichen Differentialgleichungen VI

ODE höherer Ordnung können als ODE erster Ordnung geschrieben werden, indem man die niederen Ableitungen als Variablen einführt.

Beispiel: Betrachte die ODE $x''(t) = g(t, x(t), x'(t))$. Setze $v(t) = x'(t)$. Dann ist

$$\frac{d}{dt} \underbrace{\begin{bmatrix} x(t) \\ v(t) \end{bmatrix}}_{y(t)} = \underbrace{\begin{bmatrix} v(t) \\ g(t, x(t), v(t)) \end{bmatrix}}_{f(t,y(t))}.$$

Numerische Lösung von gewöhnlichen Differentialgleichungen 1. Ordnung

Eine explizite gewöhnliche Differentialgleichung (ordinary differential equation, kurz ODE) erster Ordnung ist eine Gleichung der Form $y'(t) = f(t, y(t))$.
Gibt man noch einen Anfangswert $y_0 \in \mathbb{R}^n$ vor, so hat man ein
Anfangswertproblem (AWP):

$$y'(t) = f(t, y(t)), \qquad y(t_0) = y_0, \qquad f : \mathbb{R} \times \mathbb{R}^n \supseteq G \to \mathbb{R}^n. \qquad (*)$$

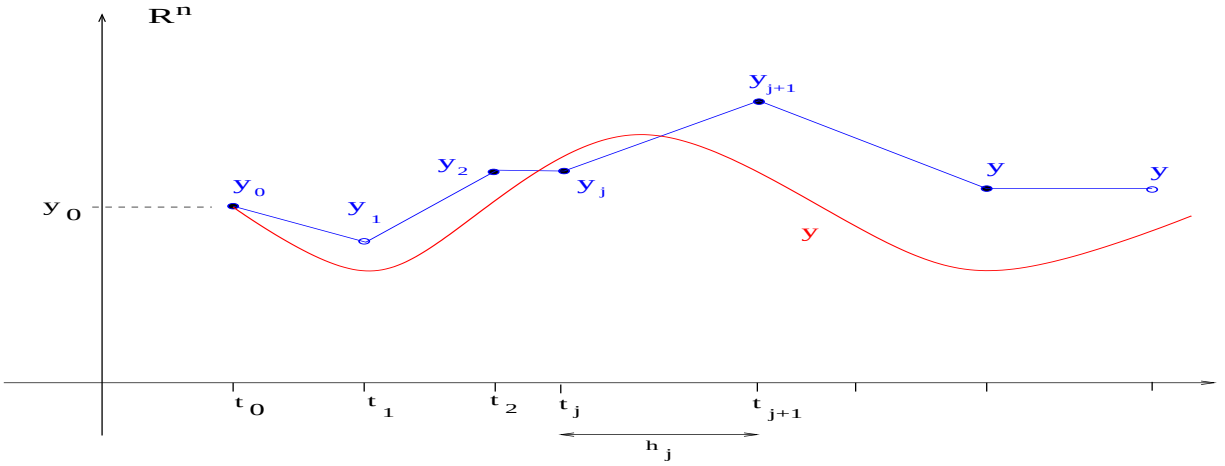
Gesucht ist die Funktion $y : [t_0, t_e] \to \mathbb{R}^n$, welche die Bedingungen $(*)$ erfüllt.

Solche Gleichungen sind oft nur numerisch lösbar. Sei y die Lösung von $(*)$ und sei $t_0 < t_1 < t_2 < \ldots < t_m = t_e$ eine Unterteilung des Intervalls $[t_0, t_e]$. Ziel ist es, für die Folge

$$y(t_0),\ y(t_1),\ y(t_2),\ y(t_3)\ldots,\ y(t_m)$$

auf numerischem Wege eine möglichst gute Näherungsfolge y_j zu berechnen:

$$y(t_0) = y_0,\quad y_1,\quad y_2,\quad y_3,\quad \ldots,\quad y_m$$



Verfahrenstypen

Anfangswertproblem:

$$y'(t) = f(t, y(t)), \qquad y(t_0) = y_0.$$

Ziel: Nähere die Folge

$$y(t_0),\ y(t_1),\ y(t_2),\ y(t_3)\ldots,\ y(t_m)$$

durch eine leicht zu berechnende Folge

$$y(t_0) = y_0,\ y_1,\ y_2,\ y_3,\ \ldots,\ y_m$$

Notation: $h_j = t_{j+1} - t_j$ (Schrittweite)

Die Folgenglieder y_j werden rekursiv berechnet.

Einige Grundtypen:

y_{j+1}	$=$	$\psi(f, t_j, y_j,\ h_j)$	explizite Einschrittverfahren
y_{j+1}	$=$	$\psi(f, t_j, y_{j+1}, y_j,\ h_j)$	implizite Einschrittverfahren
y_{j+1}	$=$	$\psi(f, t_j, y_j, \ldots, y_{j-p},\ h_j)$	explizite Mehrschrittverfahren
y_{j+1}	$=$	$\psi(f, t_j, y_{j+1},\ y_j, \ldots, y_{j-p},\ h_j)$	implizite Mehrschrittverfahren

Implizite Verfahren sind in der Regel numerisch stabiler als explizite. Manche ODE lassen sich mit expliziten Verfahren nur sehr schlecht lösen. Schwierigkeit bei impliziten Verfahren: man muss (nichtlineare) Gleichungen lösen.

Herleitung von Einschrittverfahren. Grundideen

$$\begin{array}{ccc} y'(t) & = & f(t, y(t)) \\ \downarrow & \text{ersetze} & \downarrow \\ \frac{y_{j+1}-y_j}{h_j} & = & \phi(f, t_j, y_j, y_{j+1}, h_j) \\ \text{Diff.quotient} & & \text{Näherung für } f \end{array}$$

Umstellen ergibt:

$$\boxed{y_{j+1} = y_j + h_j\, \phi(f, t_j, y_j, y_{j+1}, h_j)}$$

Einfachste Beispiele:

$\phi(f, t_j, y_j, y_{j+1}, h_j) = f(t_j, y_j)$ **explizites Eulerverfahren**

$\phi(f, t_j, y_j, y_{j+1}, h_j) = f(t_{j+1}, y_{j+1})$ **implizites Eulerverfahren**

$\phi(f, t_j, y_j, y_{j+1}, h_j) = \frac{f(t_j,y_j)+f(t_{j+1},y_{j+1})}{2}$ **Trapezregel**

Herkunft der Bezeichnung 'Trapezregel': Lösung der ODE erfüllt

$$y(t_{j+1}) = y(t_j) + \int_{t_j}^{t_{j+1}} f(t, y(t))\, dt.$$

Ersetze hierin $y(t_{j+1}), y(t_j)$ durch y_{j+1}, y_j und das Integral durch Trapezregelnäherung

$$\int_{t_j}^{t_{j+1}} f(t, y(t))\, dt \approx h_j\, \frac{f(t_j, y_j) + f(t_{j+1}, y_{j+1})}{2}.$$

Merke: Integralapprox. ist Grundlage für Herleitung von Verfahren höherer Ordnung.

Die Fehlerordnung

Das Ziel bei der numerischen Lösung einer DGL ist natürlich, dass der Fehler $\|y_j - y(t_j)\|$ möglichst klein ist. Dieser Fehler hängt u.a. von den Schrittweiten $h_j = t_{j+1} - t_j$ ab.

Man sagt, dass ein Verfahren von der **Konsistenzordnung** p ist, wenn

$$\|y_{j+1} - y(t_{j+1})\| = \mathcal{O}(h^{p+1}) \qquad \text{falls} \quad y_j = y(t_j).$$

Dabei ist $h = t_{j+1} - t_j$ die Schrittweite.
(Fehler nach einem Schritt, falls die Startwerte y_{j-1}, $y(t_{j-1})$ übereinstimmen.
Lokaler Fehler)

Man sagt, dass ein Verfahren von der **Konvergenzordnung** p ist, wenn

$$\max_{t_j \in [t_0, t_e]} \|y_j - y(t_j)\| = \mathcal{O}(h^p), \qquad \text{wobei } h = \max_j (t_{j+1} - t_j).$$

h ist die maximale Schrittweite.

Unter schwachen Voraussetzungen an die ODE und das Verfahren ist

Konvergenzordnung=Konsistenzordnung

Einfache Einschrittverfahren

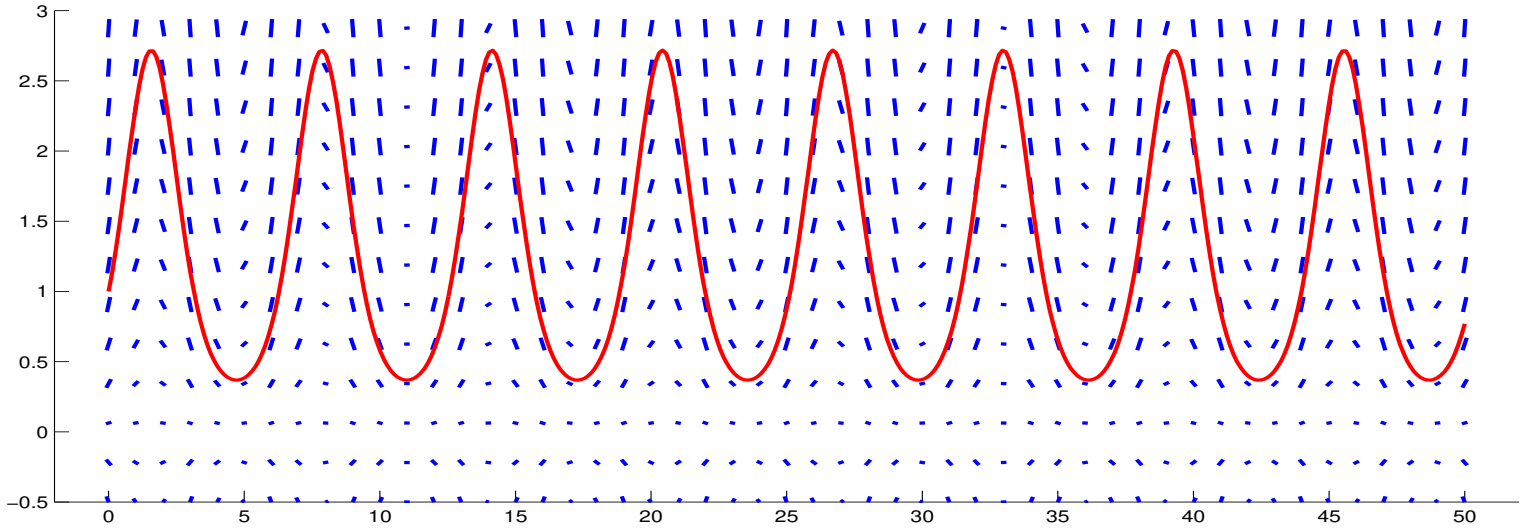
Bezeichnung	Vorschrift	Ordnung
Euler explizit	$y_{j+1} = y_j + h_j\, f(t_j, y_j)$	1
Euler implizit	$y_{j+1} = y_j + h_j\, f(t_{j+1}, y_{j+1})$	1
Trapezregel	$y_{j+1} = y_j + h_j\, \dfrac{f(t_j, y_j) + f(t_{j+1}, y_{j+1})}{2}$	2
Heun	$y_{j+1} = y_j + h_j\, \dfrac{f(t_j, y_j) + f(t_j + h_j,\, y_j + h_j f(t_j, y_j))}{2}$	2
impl. Mit.punktsregel	$y_{j+1} = y_j + h_j\, f\left(t_j + \dfrac{h_j}{2}, \dfrac{y_j + y_{j+1}}{2}\right)$	2
Collatz (mod. Euler)	$y_{j+1} = y_j + h_j\, f\left(t + \frac{h_j}{2},\; y_j + \frac{h_j}{2} f(t_j, y_j)\right)$	2

Verfahren von Heun in übersichtlicher Schreibweise:

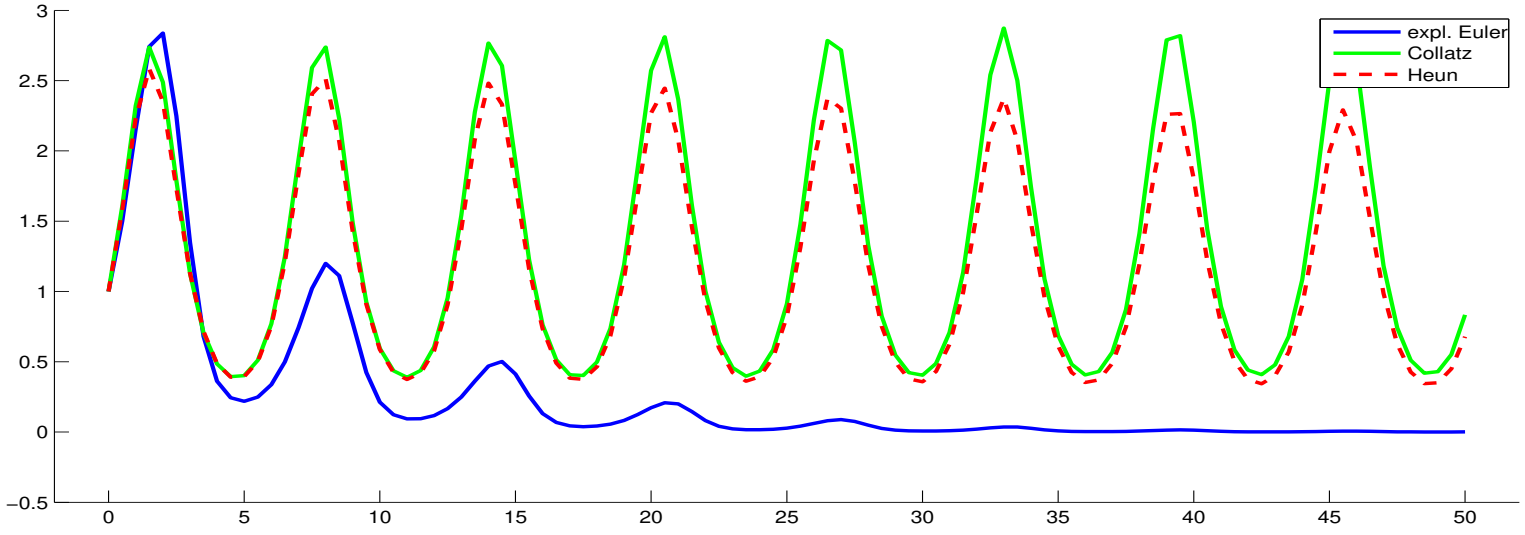
$$k_1 := f(t_j, y_j), \qquad k_2 := f(t_j + h_j, y_j + h_j\, k_1), \qquad y_{j+1} = y_j + h_j \left(\frac{1}{2} k_1 + \frac{1}{2} k_2\right).$$

Beispiel: Numerische Lösungen des Anfangswertproblems $y'(t) = \cos(t)\, y(t)$, $y(0) = 1$.

Die exakte Lösung ist die periodische Funktion $y(t) = e^{\sin(t)}$:

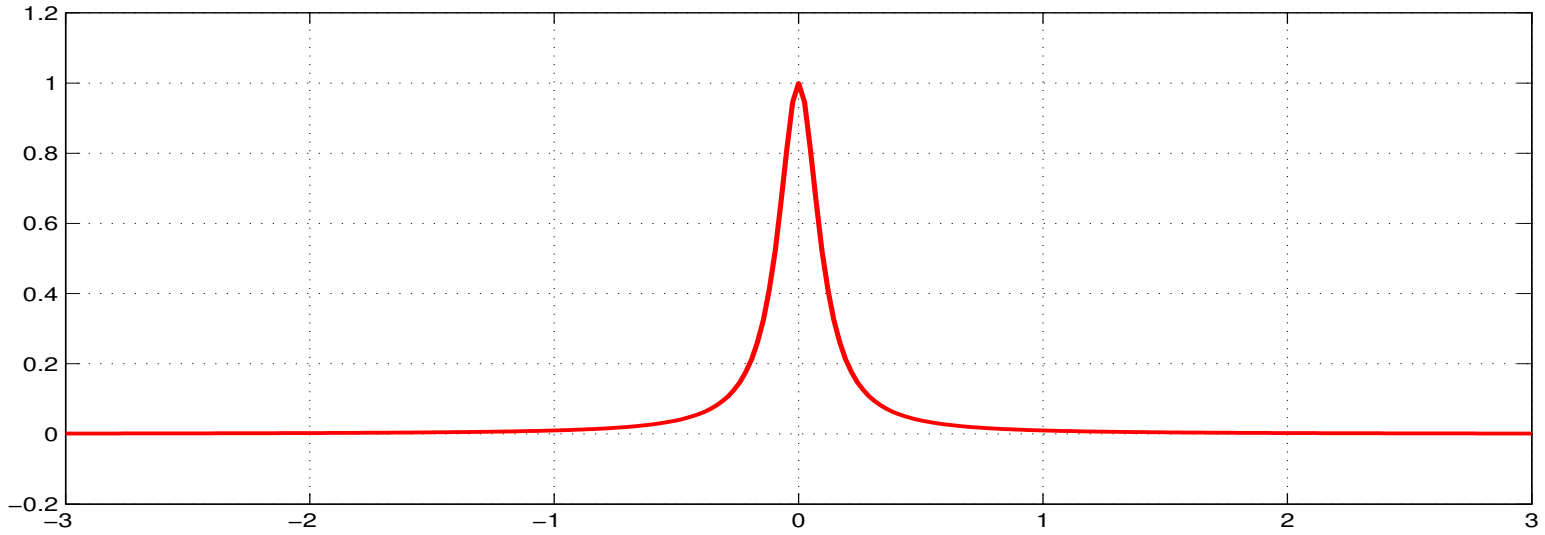


Hier sind die Ergebnisse verschiedener Löser für die (grobe) Schrittweite $h = 0.5$:

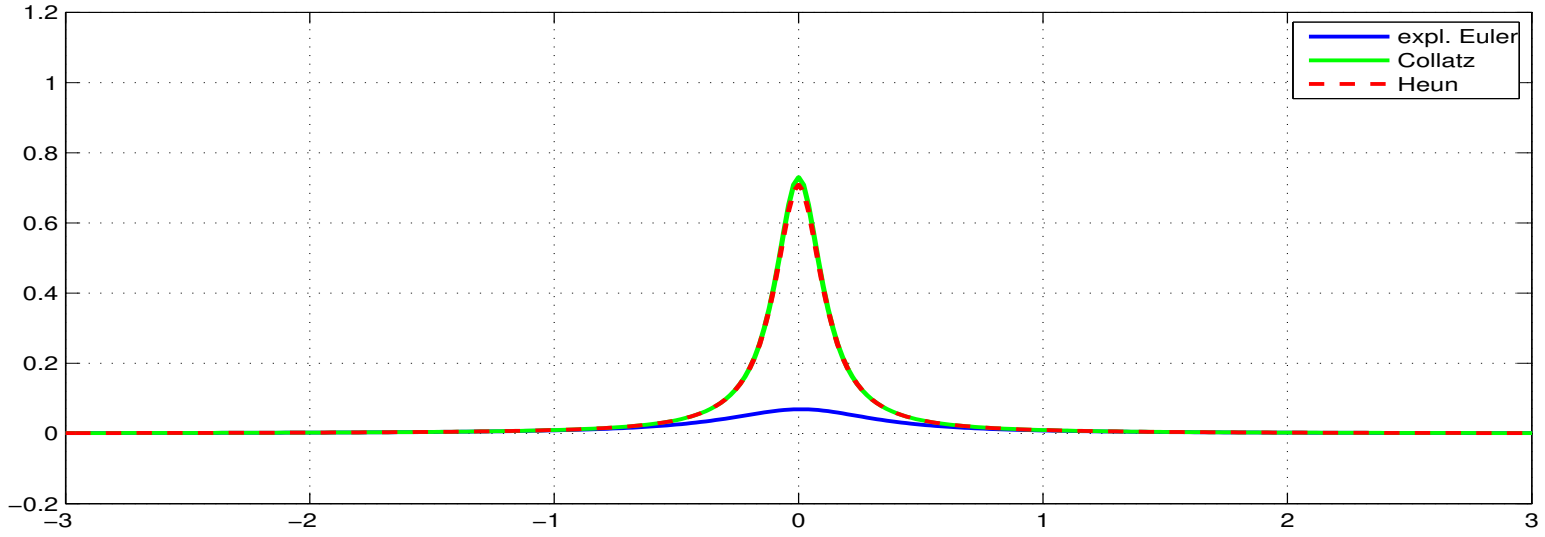


Beispiel: Numerische Lösungen der ODE $y'(t) = -2\,a\,t\,y(t)^2$.

Die exakten Lösungen sind von der Form $y(t) = 1/(c + a\,t^2), \;\; c \in \mathbb{R}$.
Hier ist die Lösung für $a = 100$, $c = 1$:



Hier sind die Ergebnisse verschiedener Löser für die Schrittweite $h = 0.02$:



Allgemeines s-stufiges Runge-Kutta-Schema:

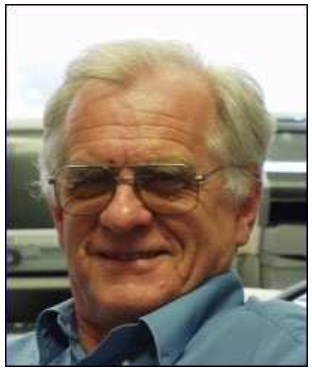
$$
\begin{aligned}
k_1 &= f(t_j + c_1\, h_j,\ y_j + h_j(a_{11}\, k_1 + a_{12}\, k_2 + \ldots + a_{1s} k_s)\ ) \\
k_2 &= f(t_j + c_2\, h_j,\ y_j + h_j(a_{21}\, k_1 + a_{22}\, k_2 + \ldots + a_{2s}\, k_s)\ ) \\
&\vdots \qquad\quad \vdots \\
k_s &= f(t_j + c_s\, h_j,\ y_j + h_j(a_{s1}\, k_1 + a_{s2}\, k_2 + \ldots + a_{ss}\, k_s)\ )
\end{aligned}
$$

$$
y_{j+1} = y_j + h_j\,(b_1\, k_1 + b_2\, k_2 + \ldots + b_s\, k_s)
$$

Daten in Butcher-Tabelle:

c_1	$a_{11}\ \ a_{12} \ldots a_{1s}$
c_2	$\vdots \qquad\quad \vdots$
$\vdots$	$\vdots \qquad\quad \vdots$
c_s	$a_{s1}\ \ a_{s2} \ldots a_{ss}$
	$b_1 \quad b_2 \quad \ldots \quad b_s$

Verfahren ist explizit, wenn $[a_{ij}]$ strikte untere Dreiecksmatrix ist.



John C. Butcher (1933-), Auckland, Neuseeland.
Tabelle im Artikel von 1963

Klassisches Runge-Kutta-Verfahren (Ordnung 4, publiziert 1901):

$$
\begin{aligned}
k_1 &= f(t_j,\ y_j) \\
k_2 &= f(t_j + (1/2)h_j,\ y_j + h_j\,(1/2)\,k_1) \\
k_3 &= f(t_j + (1/2)h_j,\ y_j + h_j\,(1/2\,)k_2) \\
k_4 &= f(t_j + h_j,\ y_j + h_j\,k_3)
\end{aligned}
$$

$$
y_{j+1} = y_j + h_j\left(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}\right).
$$

Butcher-Tabelle:

0				
1/2	1/2			
1/2	0	1/2		
1	0	0	1	
	1/6	1/3	1/3	1/6

C.D.T.Runge
(1856 – 1927)

W.M.Kutta
(1867 – 1944)

Verfahren basiert auf Simpson-Regel (Kep. Fassregel) zur Approx. von $\int_{t_j}^{t_{j+1}} f(t, y(t))\,dt$.

Lesenswerter Artikel: John Butcher. A history of Runge-Kutta methods.
Applied Numerical Mathematics Volume 20, Issue 3, March 1996,

Maximale Ordnung von expliziten Runge-Kutta-Verfahren

Die folgende Tabelle zeigt, wie groß die maximale Ordnung $p(s)$ ist, die mit einem expliziten s-stufigen Verfahren für ein bestimmtes s erreicht werden kann.

Stufe s	1	2	3	4	5	6	7	8	9	≥ 9
Ordnung $p(s)$	1	2	3	4	4	5	6	6	7	$< s-2$

Ein 2-stufiges implizites Runge-Kutta-Verfahren der Ordnung 4 (Gauss-Verfahren)

$$
\begin{aligned}
k_1 &= f(t_j + c_1\, h_j,\ y_j + h_j(a_{11}\, k_1 + a_{12}\, k_2)) \\
k_2 &= f(t_j + c_2\, h_j,\ y_j + h_j(a_{21}\, k_1 + a_{22}\, k_2)) \\
\\
y_{j+1} &= y_j + h_j(b_1\, k_1 + b_2\, k_2)
\end{aligned}
$$

mit Butcher-Tabelle:

$$
\begin{array}{c|cc}
c_1 & a_{11} & a_{12} \\
c_2 & a_{21} & a_{22} \\
\hline
 & b_1 & b_2
\end{array}
=
\begin{array}{c|cc}
1/2 - 1/(2\sqrt{3}) & 1/4 & (1/4) - 1/(2\sqrt{3}) \\
1/2 + 1/(2\sqrt{3}) & (1/4) + 1/(2\sqrt{3}) & 1/4 \\
\hline
 & 1/2 & 1/2
\end{array}
$$

Idee hinter den Runge-Kutta-Verfahren

Für eine Lösung der ODE $y'(t) = f(t, y(t))$ gilt

$$y(t_{j+1}) = y(t_j) + \int_{t_j}^{t_{j+1}} f(\tau, y(\tau))\, d\tau.$$

Sei $h_j = t_{j+1} - t_j$. Zur Approximation des Integrals wähle eine Quadraturformel mit Stützstellen τ_i und Gewichten b_i:

$$y(t_{j+1}) \approx y(t_j) + h_j(\underbrace{b_1\, f(\tau_1, y(\tau_1)) + b_2\, f(\tau_2, y(\tau_2)) + \ldots + b_s\, f(\tau_s, y(\tau_s))}_{\text{gewichtete Summe von Steigungen}})$$

Leider sind die Stützwerte $f(\tau_i, y(\tau_i))$ nicht bekannt.
Finde geeignete Näherungen für die Steigungen

$$k_i \approx f(\tau_i, y(\tau_i))$$

und setze diese ein:

$$y(t_{j+1}) \approx y(t_j) + h_j(\ \underbrace{b_1\, k_1 + b_2\, k_2 + \ldots + b_s\, k_s}_{\text{gewichtete Summe von Steigungen}}\ )$$

Mit diesem Vorgehen bekommt man:

Mittelpunktregel $\Rightarrow$ Collatz

Trapezregel $\Rightarrow$ Heun

Simpsonregel $\Rightarrow$ klassisches Runge-Kutta-Verfahren

Bestimmung der Ordnung eines Verfahrens durch Taylor-Entwicklung

Anfangswertproblem: $y'(t) = f(t, y(t)) \qquad y(t_0) = y_0$

Ergebnis des numerischen Einschrittverfahrens nach einem Zeitschritt:

$$y_1 = y_1(h) = \psi(f, t_0, y_0, h)$$

Lokaler Diskretisierungsfehler

$$\|y(t_0 + h) - y_1(h)\|$$

Beispiel: Das Collatz-Verfahren.

Beim Collatz-Verfahren hat man

$$y_1(h) = y_0 + h\, f\left(t_0 + \frac{h}{2},\ y_0 + \frac{h}{2} f(t_0, y_0)\right).$$

Die Ableitungen an der Stelle $h = 0$ sind

$$y_1'(0) = f(t_0, y_0), \qquad y_1''(0) = \frac{\partial f}{\partial t}(t_0, y_0) + \frac{\partial f}{\partial y}(t_0, y_0)\ f(t_0, y_0).$$

Die Taylor-Entwicklung ist daher

$$y_1(h) \quad = \quad y_0 + f(t_0, y_0)\, h + \left[\frac{\partial f}{\partial t}(t_0, y_0) + \frac{\partial f}{\partial y}(t_0, y_0)\ f(t_0, y_0)\right](h^2/2) + \mathcal{O}(h^3)$$

Für die exakte Lösung hat man wegen $y'(t) = f(t, y(t))$, $y''(t) = \frac{d}{dt} f(t, y(t)) = \frac{\partial f}{\partial t}(t, y(t)) + \frac{\partial f}{\partial y}(t, y(t))\, y'(t)$ dieselbe Taylor-Entwicklung :

$$y(t_0 + h) = y_0 + f(t_0, y_0)\, h + \left[\frac{\partial f}{\partial t}(t_0, y_0) + \frac{\partial f}{\partial y}(t_0, y_0)\ f(t_0, y_0)\right](h^2/2) + \mathcal{O}(h^3)$$

Also ist $\|y_1(h) - y(t_0 + h)\| = \mathcal{O}(h^3)$. $\Rightarrow$ Konsistenzordnung ist mindestens 2.

Eingebettete Runge-Kutta-Verfahren

Ziel: Reduktion des Rechenaufwands durch Schrittweitensteuerung

Die Schrittweite h_j wird in jedem Schritt so vergrößert/verkleinert, dass eine vorgegebene Genauigkeit erreicht wird.

Gegeben: Zwei Runge-Kutta-Verfahren mit denselben Koeffizienten a_{ij}, c_i, aber verschiedenen Gewichten b_i und unterschiedlichen Ordnungen.

Verfahren der Ordnung p: $\quad y_{j+1} = y_j + h_j(b_1\,k_1 + b_2\,k_2 + \ldots + b_s\,k_s)$

Verfahren der Ordnung $p+1$: $\quad \widehat{y}_{j+1} = y_j + h_j(\widehat{b}_1\,k_1 + \widehat{b}_2\,k_2 + \ldots + \widehat{b}_s\,k_s)$

$\widehat{y}_{j+1}$ approximiert die exakte Lösung genauer als y_{j+1}.

Die Zahl $\|y_{j+1} - \widehat{y}_{j+1}\|$ schätzt den Abstand von y_{j+1} zur exakten Lösung.

Wähle eine Toleranz tol und bestimme in jedem Schritt die Schrittweite h_j so, dass

$$err := \frac{\|y_{j+1} - \widehat{y}_{j+1}\|}{h_j(1 + \|y_j\|)\,tol} \approx 1$$

Den Algorithmus dafür findet man z.B. in Bollhöfer/Mehrmann.

In der Literatur findet man noch andere (komplizertere) Bildungsvorschriften für die Fehlerfunktion err.

Eingebettete Runge-Kutta-Verfahren. Beispiele

1) Verfahren von Bogacki/Shampine (Matlab-Solver `ode23`)

0				
$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{3}{4}$	0	$\frac{3}{4}$		
1	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$	
	$\frac{2}{9}$	$\frac{1}{3}$	$\frac{4}{9}$	0
	$\frac{7}{24}$	$\frac{1}{4}$	$\frac{1}{3}$	$\frac{1}{8}$

Ordnung 2

Ordnung 3

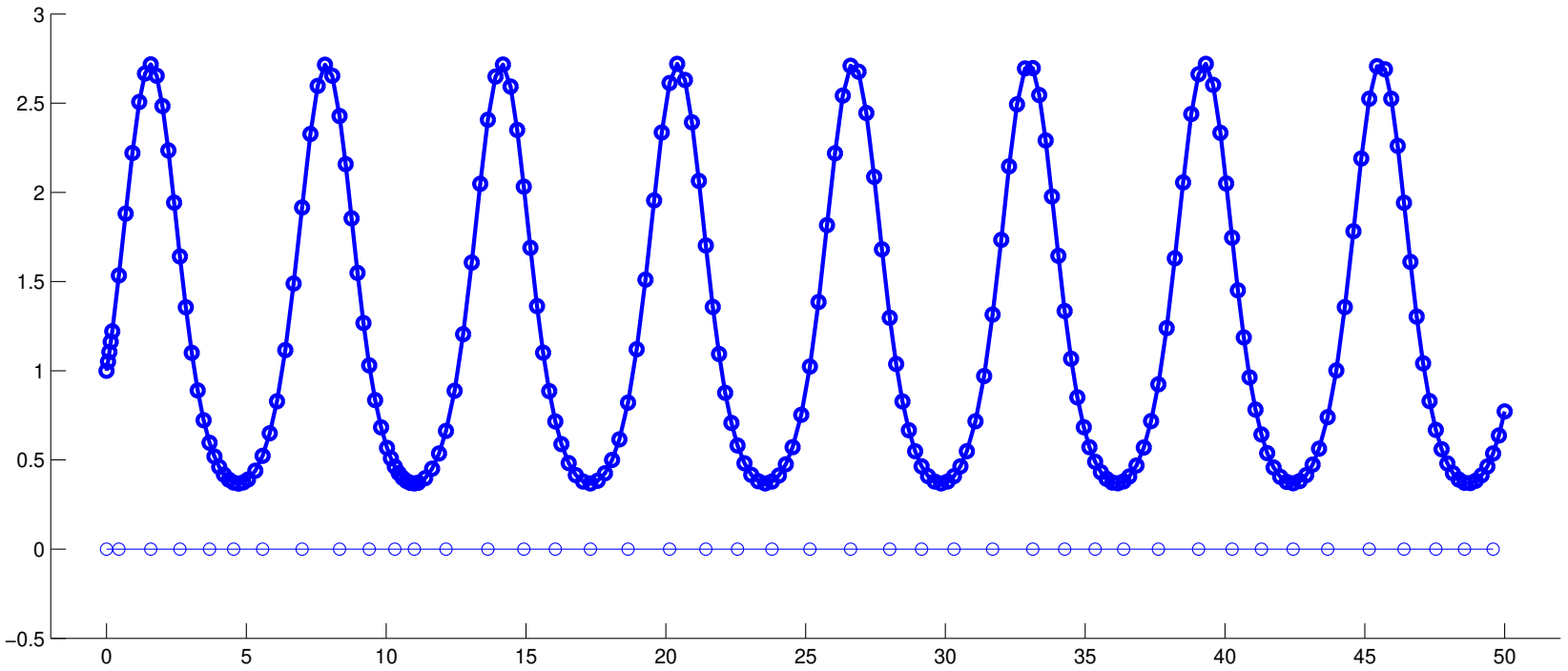
2) Verfahren von Dormand/Prince (Matlab-Solver `ode45`)

0							
$\frac{1}{5}$	$\frac{1}{5}$						
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$					
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$				
$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$			
1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$		
1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	
	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	0
	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$

Ordnung 4

Ordnung 5

Beispiel: Lösung des Anfangswertproblems $y'(t) = \cos(t)\, y(t)$, $y(0) = 1$ mit `ode45`



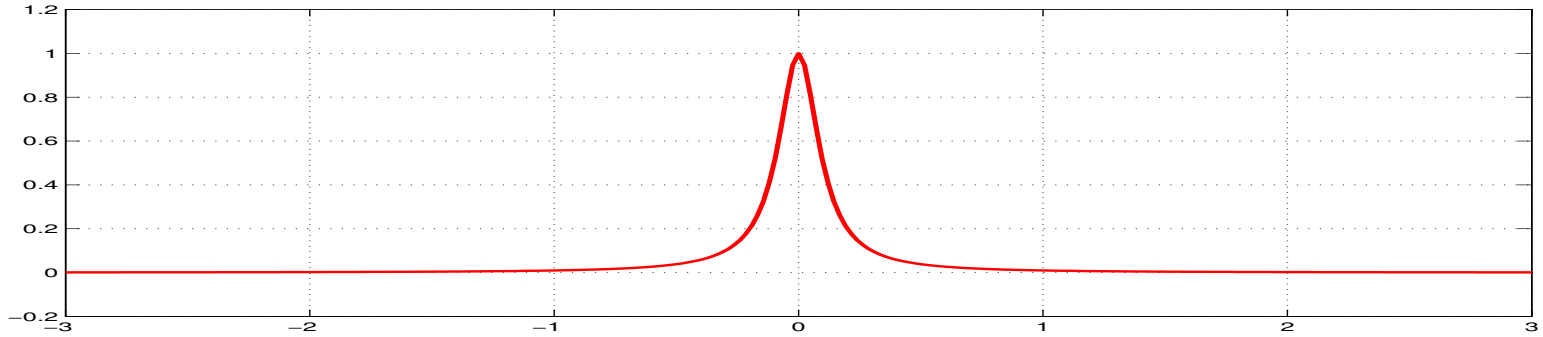
Auf der Nulllinie ist jede fünfte Stützstelle t_j dargestellt.

Matlab-Code:

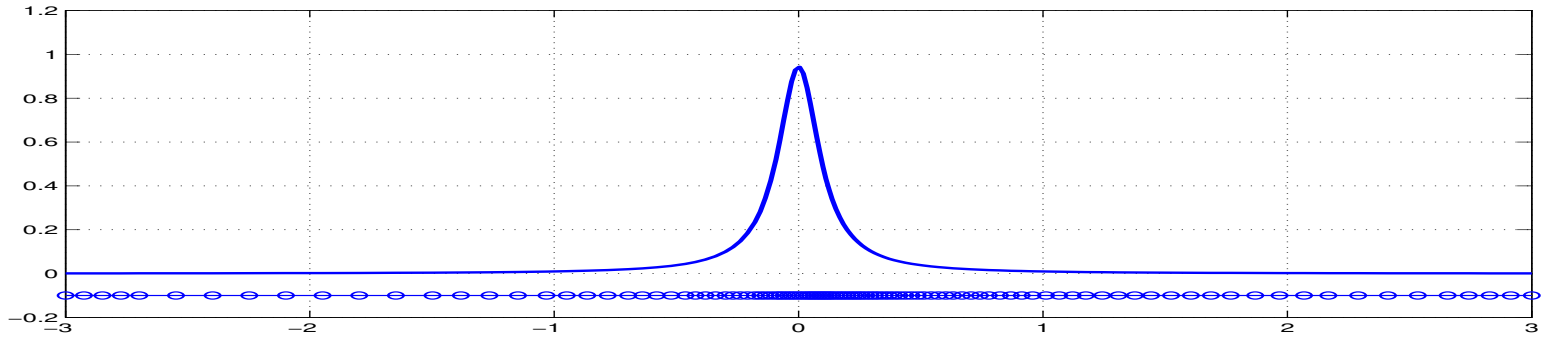
```
[T,Y] = ode45(@(t,y)(cos(t).*y),[0,50],1);
clf
hold on
axis([-2,52, -.5,3])
plot(T,Y,'-o','linewidth',2)
plot(T(1:5:length(T)),0*(1:5:length(T)),'-o')
```

Beispiel: Lösungen der ODE $y'(t) = -2\,a\,t\,y(t)^2$ mit MATLAB.

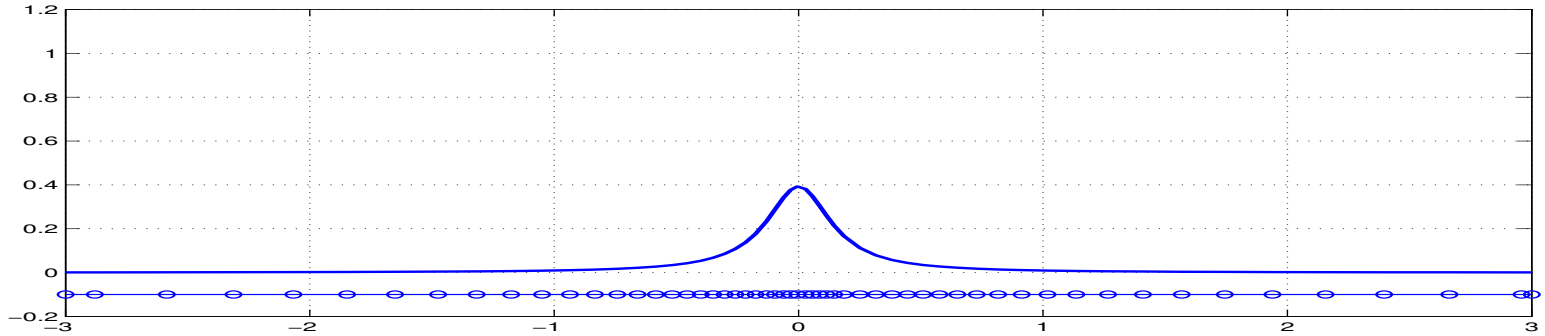
Die exakten Lösungen sind von der Form $y(t) = 1/(c + a\,t^2), \;\; c \in \mathbb{R}$.
Hier ist die Lösung für $a = 100$, $c = 1$:



Numerische Lösung mit `ode45`:



Numerische Lösung mit `ode23`:



Die Mehrschrittverfahren von Adams-Bashforth

Seien $y_j, y_{j-1}, y_{j-2}, \ldots, y_{j-p+1}$ bereits berechnete Näherungswerte für die Lösung des Anfangswertproblems. Schrittweite=$const$=h.

Sei P das Interpolationspolynom durch die Punkte

$$(t_i, f(t_i, y_i)), \qquad i = j, j-1, j-2, \ldots j-p+1.$$

Approximiere

$$y(t_{j+1}) \;=\; y_j + \int_{t_j}^{t_{j+1}} f(\tau, y(\tau))\, d\tau$$

durch

$$y_{j+1} \;=\; y_j + \int_{t_j}^{t_{j+1}} P(\tau)\, d\tau.$$

Die so konstruierten Verfahren haben Ordnung p

Beachte: Mehrschrittverfahren müssen mit Einschrittverfahren gestartet werden, weil man die Werte für mehrere zückliegende Zeitpunkte braucht.

Das einfachste Adams-Bashforth-Verfahren

Das Interpolationspolynom durch die Punkte

$$(t_j, f(t_j, y_j)), \quad (t_{j-1}, f(t_{j-1}, y_{j-1})), \qquad t_j - t_{j-1} = h$$

ist die Gerade

$$P(t) = f(t_{j-1}, y_{j-1}) + \frac{f(t_j, y_j) - f(t_{j-1}, y_{j-1})}{h}(t - t_{j-1}).$$

Integration ergibt

$$y_{j+1} = y_j + \int_{t_j}^{t_{j+1}} P(\tau)\, d\tau = y_j + h \left(\frac{3}{2} f(t_j, y_j) - \frac{1}{2} f(t_{j-1}, y_{j-1}) \right).$$

Programmieraufgabe 2 Gegeben seien zwei Massen m_1, m_2 die sich zur Zeit t an den Orten $\vec{x}_1(t), \vec{x}_2(t)$ befinden. Nach dem Newtonschen Graviationsgesetz zieht die Masse m_2 die Masse m_1 mit der Kraft

$$\vec{F}_1(\vec{x}_1(t), \vec{x}_2(t)) = \frac{\gamma\, m_1\, m_2}{\|\vec{x}_2(t) - \vec{x}_1(t)\|_2^3}(\vec{x}_2(t) - \vec{x}_1(t))$$

an. Dabei ist γ die Gravitationskonstante. Die Anziehungskraft, welche m_1 auf m_2 ausübt, ist $\vec{F}_2 = -\vec{F}_1$. Sei $\vec{v}_k(t) = \vec{x}_k'(t)$ die Geschwindigkeit von Masse k zum Zeitpunkt t. Dann gilt

$$m_k\, \vec{v}_k'(t) = \vec{F}_k(t), \quad k = 1, 2 \qquad \text{(Kraft=Masse mal Beschleunigung).}$$

Insgesamt hat man für die Bewegung der Massen die folgende Differentialgleichung 1. Ordnung:

$$\frac{d}{dt} \underbrace{\begin{bmatrix} \vec{x}_1(t) \\ \vec{x}_2(t) \\ \vec{v}_1(t) \\ \vec{v}_2(t) \end{bmatrix}}_{=:\vec{y}(t)} = \underbrace{\begin{bmatrix} \vec{v}_1(t) \\ \vec{v}_2(t) \\ \vec{F}_1(\vec{x}_1(t), \vec{x}_2(t))/m_1 \\ \vec{F}_2(\vec{x}_1(t), \vec{x}_2(t))/m_2 \end{bmatrix}}_{\vec{f}(\vec{y}(t))} \qquad (*)$$

Schreibe eine MATLAB-Funktion `doppelstern`(m1,m2,x1,x2,p,h) welche die Differentialgleichung $(*)$ mit Hilfe des klassischen Runge-Kutta-Verfahrens löst und einen Film erzeugt, der die Bewegung der Massen zeigt. Ein Beispiel wird in der Vorlesung vorgeführt. Dabei sind alle Vektoren 2-dimensional (ebene Bewegung). Der Parameter `h` ist die Zeitschrittweite, der Parameter `p` bestimmt die Anfangsgeschwindigkeiten:

$$\vec{v}_1(t_0) = \begin{bmatrix} 0 \\ p/m_1 \end{bmatrix}, \quad \vec{v}_2(t_0) = \begin{bmatrix} 0 \\ -p/m_2 \end{bmatrix}$$

Die Gravitationskonstante soll 1 sein: $\gamma = 1$.

Tipp: Schreibe zuerst eine Funktion `y_neu=ruku_schritt(f,y,h)`, die für eine Differentialgleichung mit beliebiger rechter Seite f (welche aber nicht explizit von der Zeit abhängt) einen Runge-Kutta-Schritt mit Schrittweite h ausführt.

Die Funktion f ist im vorliegenden Fall nicht explizit von der Zeit abhängig (d.h. wir haben $f(y)$ und nicht $f(t, y)$).

Das klassische Runge-Kutta-Verfahren lautet in diesem Fall:

$$
\begin{aligned}
k_1 &= f(y_j) \\
k_2 &= f(y_j + h_j\,(1/2)\,k_1) \\
k_3 &= f(y_j + h_j\,(1/2\,)k_2) \\
k_4 &= f(y_j + h_j\,k_3) \\
\\
y_{j+1} &= y_j + h_j \left(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} \right).
\end{aligned}
$$

Tipps und Vorschläge zur Lösung der Programmieraufgabe.

Drei Funktionen in eine Datei schreiben:

Hauptfunktion: `doppelstern(M1,M2,x1,x2,p,h)`

Unterfunktionen: `y_neu=ruku_schritt(f,y,h), f=gravi(y)`

`gravi(y)` ist die rechte Seite der ODE. Diese in der Funktion `doppelstern` in `ruku_schritt` einsetzen: `y=ruku_schritt(@gravi,y,h)`)

Anschließend die Ortskomponenten von y plotten.

Achtung: bei jedem Plot des Films muss das ganze Bild neu aufgebaut werden (mit clf, hold on, axis ...).

Den Plot jedes Bildes mit dem Kommando `drawnow` beenden.

Massen und Gravitationskonstante als globale Variablen behandeln.

Vorschlag für den Aufbau von `doppelstern`:

```
function doppelstern(M1,M2,x1,x2,p,h)

global gamma
global m1
global m2

m2=M2;
m1=M1;
gamma=1;

v1=[0; p/m1];
v2=[0; -p/m2];
y=[x1; x2; v1; v2];

    Hier Zaehlschleife fuer den Plot der Bilder einfuegen

end
```

Die Kommandos

```
global gamma
global m1
global m2
```

müssen auch in der Funktion gravi vorkommen.

Stabilität und Steifheit

Die Testgleichung

Sei $\lambda = \alpha + i\omega \in \mathbb{C}$ eine vorgegebene Zahl.
Wir betrachten die lineare ODE (Testgleichung)

$$y'(t) = \lambda\, y(t)$$

Die exakten Lösungen sind

$$y(t) = y(0)\, e^{\lambda t}, \qquad y(0) \in \mathbb{C} \text{ beliebig.}$$

Der Absolutbetrag der Lösung ist

$$|y(t)| = |y(0)|\, e^{\mathrm{Re}(\lambda)\, t} = |y(0)|\, e^{\alpha\, t}$$

Daraus folgt

1. Wenn $\alpha = \mathrm{Re}(\lambda) < 0$, dann $\lim_{t\to\infty} y(t) = 0$.
2. Wenn $\alpha = \mathrm{Re}(\lambda) > 0$, dann $\lim_{t\to\infty} |y(t)| = \infty$.
3. Wenn $\mathrm{Re}(\lambda) = 0$, also $\lambda = i\omega$, dann $|y(t)| = const = |y(0)|$.

Eine numerische Lösung der Testgleichung sollte auch diese Eigenschaften haben, damit sie brauchbar ist. Wir werden sehen, dass dazu in der Regel Bedingungen an die Schrittweite erfüllt sein müssen.

Bemerkung: Im Fall 1 ist 0 ein stabiler Gleichgewichtspunkt der ODE.

Lösung der Testgleichung $y'(t) = \lambda\, y(t)$ mit dem expliziten Euler

Das explizite Euler-Verfahren ergibt für die Testgleichung die Folge

$$y_{j+1} = y_j + h\,\lambda y_j = (1 + h\,\lambda)\, y_j, \qquad y_0 = y(0).$$

Also:

$$y_j = (1 + h\,\lambda)^j\, y(0)$$

Folglich:

1. Wenn $|1 + h\,\lambda| < 1$, dann $\lim_{j\to\infty} y_j = 0$
2. Wenn $|1 + h\,\lambda| > 1$, dann $\lim_{j\to\infty} |y_j| = \infty$
3. Wenn $|1 + h\,\lambda| = 1$, dann $|y_j| = const = |y(0)|$.

Das Verhalten der num. Lösung hängt also nicht von $\mathrm{Re}(\lambda)$ sondern von $|1 + h\,\lambda|$ ab.

Wir betrachten nun den einfachen Fall $\lambda = \alpha < 0$.
Dann ist $\lim_{t\to\infty} y(t) = \lim_{t\to\infty} |y(0)|\, e^{\lambda t} = 0$.
Für die numerische Lösung gilt aber

$$\lim_{j\to\infty} y_j = 0 \quad \Leftrightarrow \quad |1 + h\,\lambda| < 1 \quad \Leftrightarrow \quad -1 < 1 + h\,\lambda = 1 - h\,|\lambda| \quad \Leftrightarrow \quad h < 2/|\lambda|$$

Wenn $h \geq 2/|\lambda|$ dann gibt die numerische Lösung das Langzeitverhalten der exakten Lösung überhaupt nicht wider. Außerdem: wenn $h > 1/|\lambda|$ dann oszilliert die numerische Lösung (weil dann $1 + h\,\lambda < 0$) während die exakte Lösung das nicht tut.

Lösung der Testgleichung $y'(t) = \lambda\, y(t)$ mit dem impliziten Euler

Das implizite Euler-Verfahren ergibt für die Testgleichung die Folge

$$y_{j+1} = y_j + h\,\lambda\, y_{j+1} \qquad \Rightarrow \qquad y_{j+1} = \frac{1}{1 - h\,\lambda}\, y_j.$$

Also:

$$y_j = \left(\frac{1}{1 - h\,\lambda}\right)^j y(0)$$

Folglich:

1. Wenn $|1 - h\,\lambda|^{-1} < 1$, dann $\lim_{j\to\infty} y_j = 0$
2. Wenn $|1 - h\,\lambda|^{-1} > 1$, dann $\lim_{j\to\infty} |y_j| = \infty$
3. Wenn $|1 - h\,\lambda|^{-1} = 1$, dann $|y_j| = const = |y(0)|$.

Das Verhalten der num. Lösung hängt also nicht von Re(λ) sondern von $|1 - h\,\lambda|^{-1}$ ab.

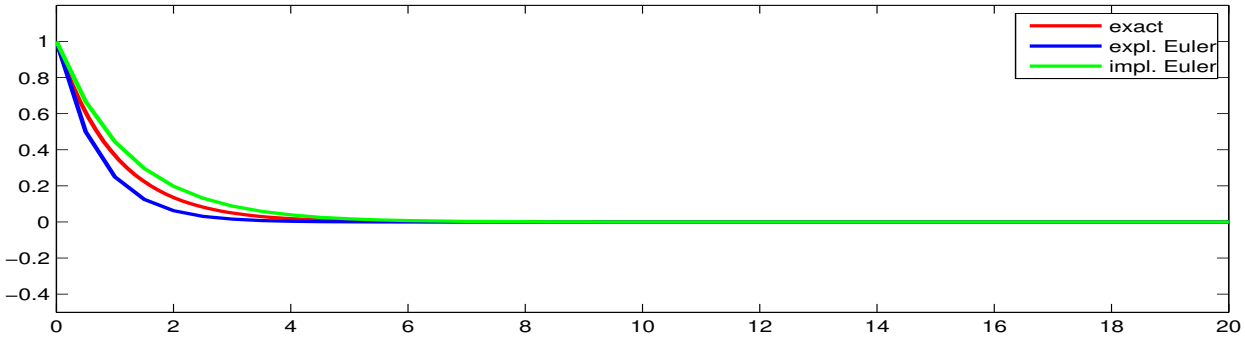
Wir betrachten wieder den einfachen Fall $\lambda = \alpha < 0$.
Dann ist $\lim_{t\to\infty} y(t) = \lim_{t\to\infty} |y(0)|\, e^{\lambda t} = 0$.

Die numerische Lösung konvergiert ebenfalls gegen 0, und zwar für **jede** Schrittweite, denn

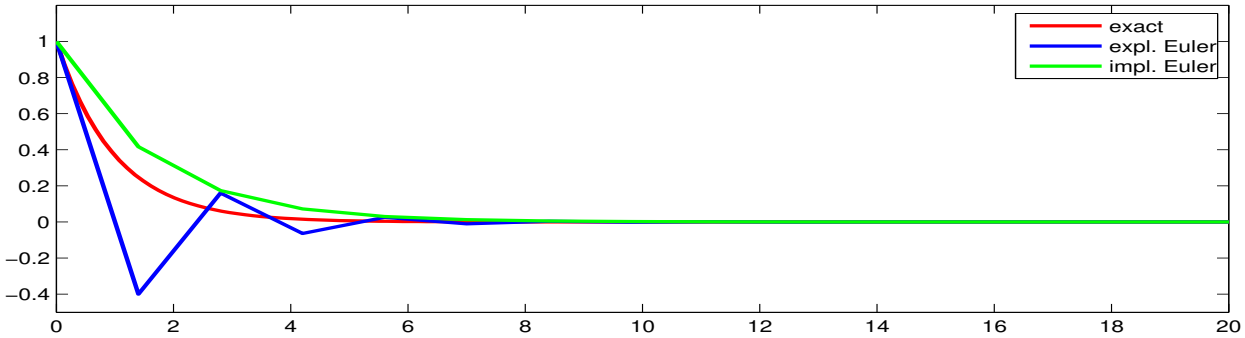
$$\lambda < 0 \quad \Rightarrow \quad 1 - h\lambda > 1 \quad \Rightarrow \quad |1 - h\lambda|^{-1} < 1.$$

Vergleich expliziter/impliziter Euler für die ODE $y'(t) = -y(t)$

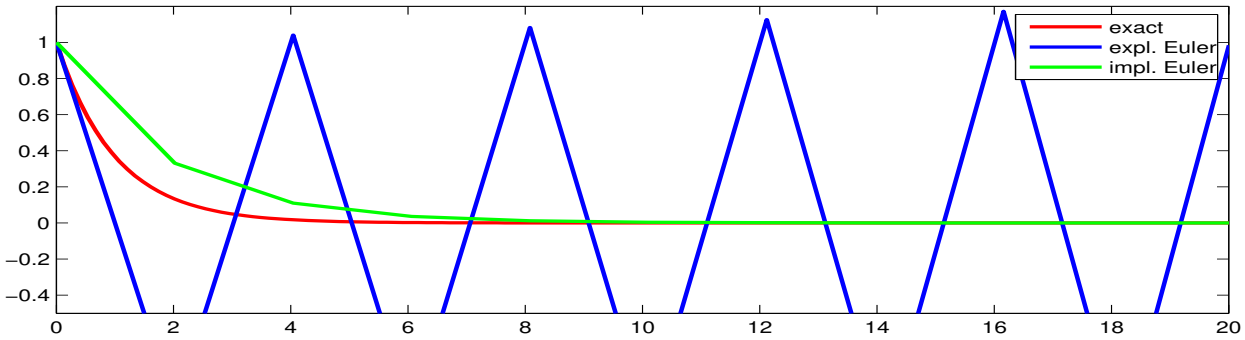
Schrittweite $h = 0.5$:



Schrittweite $h = 1.4$:



Schrittweite $h = 2.02$:



Stabiltitätsgebiete beim expliziten/impliziten Euler-Verfahren

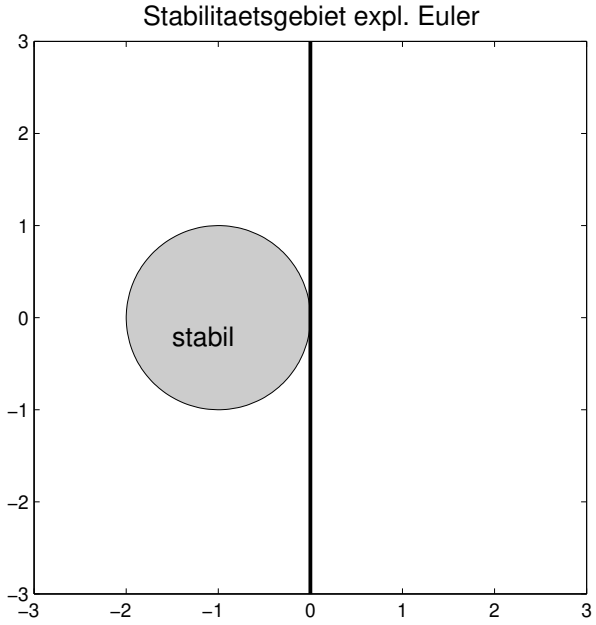
Wir haben gesehen, dass

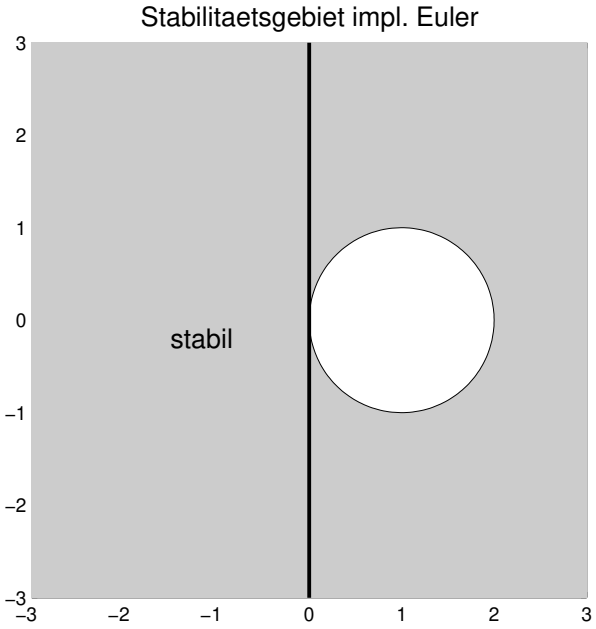
beim expliziten Euler: $\lim_{j\to\infty} y_j = 0 \quad \Leftrightarrow \quad |1 + h\,\lambda| < 1, \qquad (*)$

beim impliziten Euler: $\lim_{j\to\infty} y_j = 0 \quad \Leftrightarrow \quad |1 - h\,\lambda|^{-1} < 1. \qquad (**)$

Diese Bedingungen lassen sich graphisch veranschaulichen:

Sei $z = h\,\lambda \in \mathbb{C}$. Dann besagt die Ungleichung $(*)$, dass z im inneren des Kreises um -1 mit Radius 1 liegt. Die Ungleichung $(**)$ besagt, dass z außerhalb des Kreises um 1 mit Radius 1 liegt. Die Gebiete in denen die Ungleichungen $(*)$ bzw. $(**)$ erfüllt sind, heißen Stabilitätsgebiete des jeweiligen Verfahrens.





Noch zu schreiben:

1. Stabilitätsfunktion und Stabilitätsgebeit bei allgemeinen Runge-Kutta-Verfahren

2. A-Stabilität. Impl. Mittelunktsregel und Trapezregel sind A-stabil.

3. Verallgemeinerung auf vektorwertige lineare ODE. Zusammenhang mit Eigenwerten.