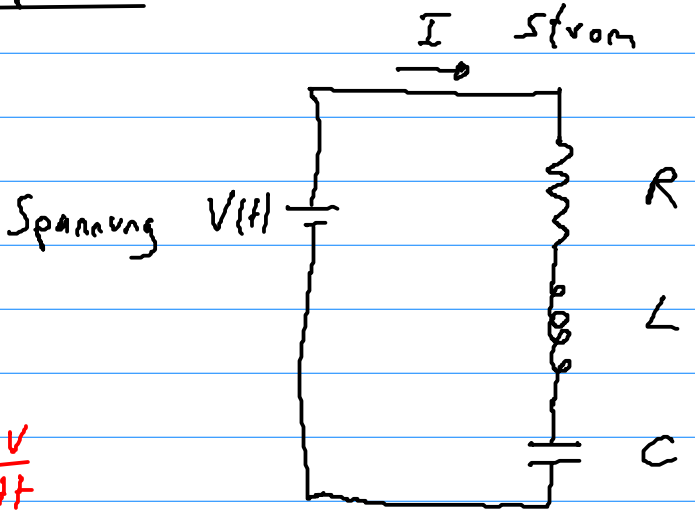
II. Explizite Einschrittverfahren

Ziele:
- Anfangswertproblem (gew. Diff.-Gl. + Anfangsbedingungen)
- Lösbarkeit?
- explizite Einschrittverfahren (Euler, Runge-Kutta)
- Genauigkeit (Diskretisierungsfehler)

Wozu: Viele Anwendungen ...

I.1 Motivation und Beispiele

(1) Schwingkreis



$$\underbrace{\ddot{I}}_{\frac{d^2I}{dt}} + \frac{R}{L}\underbrace{\dot{I}}_{\frac{dI}{dt}} + \frac{C}{L} I = \underbrace{\dot{V}}_{\frac{dV}{dt}}$$

(Skalare) gewöhnliche Differentialgleichung (DGL) 2. Ordnung für Strom $I(t)$

Oft: **ODE** für Ordinary Differential Equation

(2) Molekular-Dynamik (MD)

$$m_i \cdot \ddot{\vec{x}}_i = -\vec{\nabla}_i U(\vec{x}_1, \dots, \vec{x}_N), \quad i = 1, \dots, N$$

wobei N ... Anzahl Atome
$\vec{x}_i$ ... Position des i-ten Atom
U ... Potential (hängt von allen Atomen ab!)

System gewöhnlicher DGLen 2. Ordnung für die Positionen der Atome $\vec{x}(t)$ (Im Prinzip einfach nur Newton's Bewegungsgleichungen).

(3) Maxwell-Gleichungen

magnetische Feldstärke, Stromdichte, elektrische Flussdichte

$$\vec{\nabla} \times \vec{H} = \vec{j} + \frac{\partial \vec{D}}{\partial t}$$

magnetische Flussdichte

$$\vec{\nabla} \times \vec{E} = -\frac{\partial \vec{B}}{\partial t}$$

elektrische Feldstärke

$$\vec{\nabla} \cdot \vec{D} = \rho \quad \text{elektrische Ladungsdichte}$$

$$\vec{\nabla} \cdot \vec{B} = 0$$

System von partiellen DGL (weil partielle Ableitungen auftauchen!)

eigene Vorlesungen

Durch diskretisieren der Orts-Ableitungen (z.B. mit finiten Differenzen / Volumen / Elementen) erhält man ein (grosses) System von gewöhnlichen DGLen 1. Ordnung

← diese Vorlesung

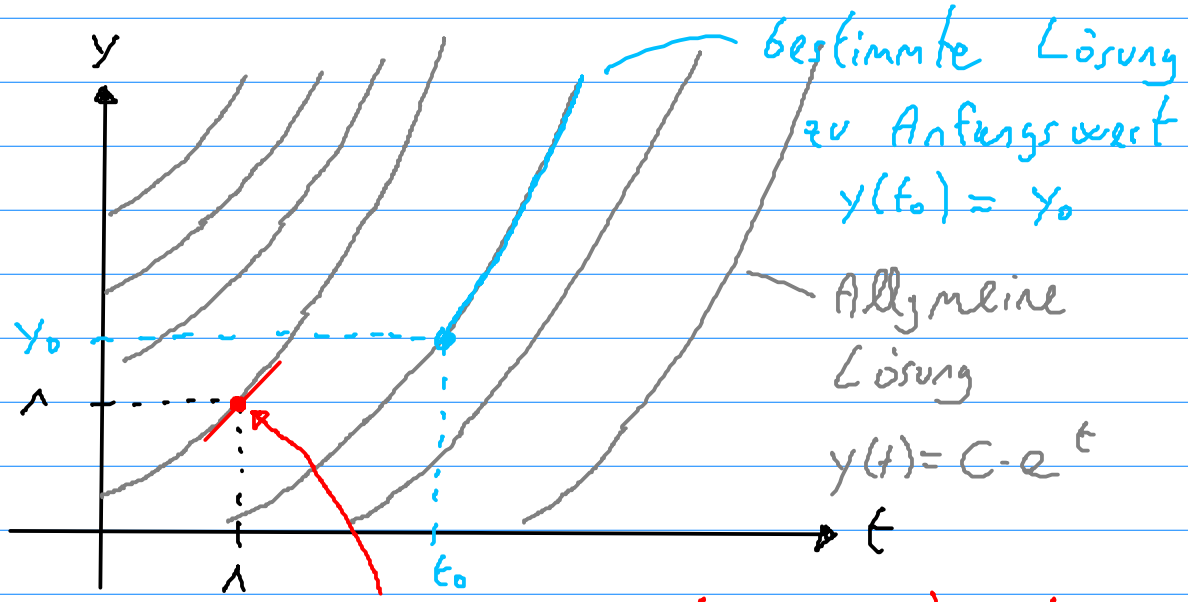
Oft: PDE für Partial Differential Equation

Zur Illustration betrachten wir eine (die?!) einfache skalare DGL

$$\dot{y}(t) = y(t) = f(t, y(t))$$

sog. rechte Seite der DGL

Eine (skalare) DGL lässt sich graphisch mit einen Richtungsfeld / Vektorfeld darstellen:

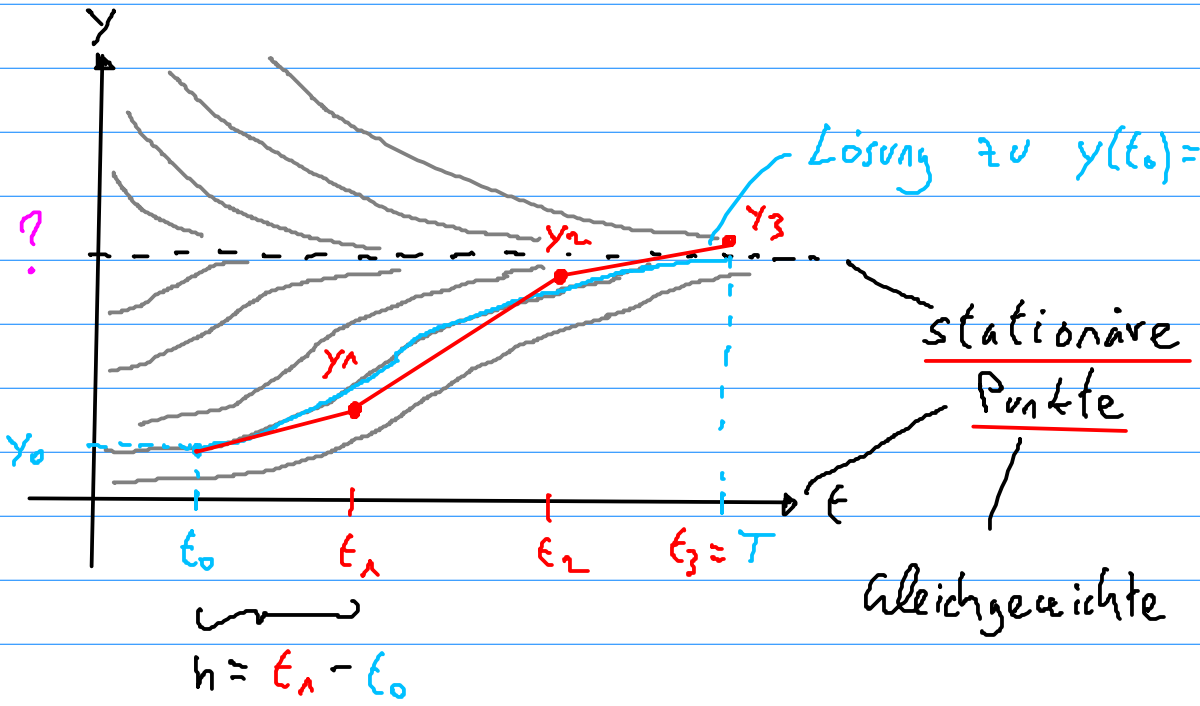


Steigung von $y(t)$ in $(t=1, y=1)$ ist 1

Eine etwas "interessantere" (skalare) DGL:

$$\dot{y}(t) = a \cdot y(t) - b \cdot y^2(t) = \left(a - b \cdot y(t)\right) y(t)$$

Graphisch:



Einfache Approximation der Lösung

① Diskretisiere das Zeitintervall $I = [t_0, T]$

$$t_k = t_0 + k \cdot h, \quad k = 0, 1, \ldots, N$$

mit $$h = \frac{T - t_0}{N}$$

② Setze Anfangswert $y_0 = y(t_0)$

③ Berechne für $k = 0, 1, \dots, N-1$

$$y_{k+1} = y_k + h \cdot f(t_k, y_k)$$

Dies ist das Euler Verfahren
oder explizites Euler
oder Vorwärts Euler
(Euler forward)

I.A. stellen sich nun folgende Fragen:

(i) Macht es Sinn (approx.) Lösungen zu suchen? (Bedingungen an die rechte Seite $f(t, y(t))$ für Existenz und Eindeutigkeit)

(ii) DGLen höherer Ordnung? (z.B. (1) & (2) oben)

(iii) Konvergiert das Euler Verfahren gegen die exakten Lösungen für $h \to 0$? Wie schnell?

(iv) Gibt's "bessere" Methoden?

II.2 Grundbegriffe

Def.: Ein skalares Anfangswertproblem (AWP) erster Ordnung:

Finde eine Funktion $y(t)$ einer Variablen (z.B. die Zeit) mit

$$\dot{y}(t) = f(t, y(t)) \qquad \text{(gew. DGL 1. Ordnung)}$$

auf dem Intervall $I = [t_0, T]$ und

$$y(t_0) = y_0 \qquad \text{(Anfangswert (AW))}$$

auch Anfangsbedingung

Oft hat man nicht nur eine DGL, sondern ein ganzes System

Def.: Ein (allgemeines) AWP erster Ordnung:

Finde die Funktionen $y_1(t), \dots y_n(t)$ einer Variablen (z.B. Zeit) mit

$$\begin{aligned} \dot{y}_1(t) &= f_1(t, y_1(t), \dots, y_n(t)) \\ &\vdots \\ \dot{y}_n(t) &= f_n(t, y_1(t), \dots, y_n(t)) \end{aligned}$$

(n gew. DGLen erster Ordnung)

auf dem Intervall $I = [t_0, T]$ und

$$\begin{aligned} y_1(t_0) &= y_{1,0} \\ &\vdots \\ y_n(t_0) &= y_{n,0} \end{aligned} \qquad (n \text{ AWe})$$

Kürzer: $\dot{\vec{y}}(t) = \vec{f}(t, \vec{y})$

$\vec{y}(t_0) = \vec{y}_0$

wobei $\vec{y}(t) = \begin{pmatrix} y_1(t) \\ \vdots \\ y_n(t) \end{pmatrix}$, $\vec{y}_0 = \begin{pmatrix} y_{1,0} \\ \vdots \\ y_{n,0} \end{pmatrix}$

und $\vec{f}(t, \vec{y}) = \begin{pmatrix} f_1(t, y_1(t), \dots, y_n(t)) \\ \vdots \\ f_n(t, y_1(t), \dots, y_n(t)) \end{pmatrix}$

Bsp.: (1) Lineares DGL System

$\dot{\vec{y}}(t) = A\,\vec{y}(t)$, $\vec{y} \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$ ($n \times n$ Matrix)

$\vec{y}(t_0) = \vec{y}_0$

Lösung: $\vec{y}(t) = e^{At}\,\vec{y}_0$ (Matrix Exp. Fkt.)

$$= \left(\sum_{i=0}^{\infty} \frac{1}{i!} A^i t^i \right) \vec{y}_0$$

Def.: Ein (skalares) AWP n-ter Ordnung:

finde die Funktion $y(t)$ einer Variablen (z.B. Zeit) mit

$$y^{(n)}(t) = f\left(t, y(t), \dot{y}(t), \ddot{y}(t), \dots, y^{(n-1)}(t)\right)$$

(skalare gew. DGL n-ter Ordnung)

auf dem Intervall $I = [t_0, T]$ und

$$\begin{aligned} y(t_0) &= y_0 \\ \dot{y}(t_0) &= \dot{y}_0 \\ \ddot{y}(t_0) &= \ddot{y}_0 \\ &\vdots \\ y^{(n-1)}(t_0) &= y_0^{(n-1)} \end{aligned}$$

(n AWe für y und die (n-1)-ten Ableitungen)

Bsp.: (S) Newton's Bewegungsgleichungen

$$m \cdot \ddot{x} = F$$

(Masse, Beschleunigung, Kraft)

(gew. DGL 2-ter Ordnung)

mit

$$x(t_0) = x_0 \quad \text{(Anfangs-Position)}$$

$$\dot{x}(t_0) = \dot{x}_0 \ (= v_0) \quad \text{(Anfangs-Geschw.)}$$

Im folgenden werden wir nur numerische Methoden für AWP erster Ordnung betrachten. Jedes AWP n-ter Ordnung lässt sich in ein System AWP erster Ordnung umschreiben:

Reduktion auf ein System erster Ordnung

Gegeben eine gew. DGL n-ter Ordnung

$$y^{(n)}(t) = f\left(t, \dot{y}, \ddot{y}, \dots, y^{(n-1)}\right)$$

Wir definieren

$$\begin{aligned} z_0(t) &= y(t) \\ z_1(t) &= \dot{y}(t) = \dot{z}_0(t) \\ z_2(t) &= \ddot{y}(t) = \dot{z}_1(t) \\ &\vdots \\ z_{n-1}(t) &= y^{(n-1)}(t) = \dot{z}_{n-2}(t) \end{aligned}$$

20.03.17 Beachte: $\dot{z}_{n-1}(t) = ?$

Dann können wir

$$y^{(n)}(t) = f\left(t, y(t), \dot{y}(t), \ddot{y}(t), \dots, y^{(n-1)}(t)\right)$$

$$\begin{cases} \dot{z}_{n-1}(t) = f\left(t, z_0(t), z_1(t), z_2(t), \dots, z_{n-1}(t)\right) \end{cases}$$

gew. DGL erster Ordnung!

Obige Definitionen sind auch $(n-1)$ gew. DGL erster Ordnung. Insgesamt haben wir also folgendes System von n gew. DGL erster Ordnung

$$\dot{\vec{z}}(t) = \vec{g}\left(t, \vec{z}(t)\right)$$

wobei

$$\vec{z}(t) = \begin{pmatrix} z_0(t) \\ z_1(t) \\ \vdots \\ z_{n-2}(t) \\ z_{n-1}(t) \end{pmatrix}$$

und

$$\vec{g}\left(t, \vec{z}(t)\right) = \begin{pmatrix} z_1(t) \\ z_2(t) \\ \vdots \\ z_{n-1}(t) \\ f\left(t, z_0(t), z_1(t), \dots, z_{n-1}(t)\right) \end{pmatrix}$$

Für die AWe ergibt sich

$$\vec{z}(t_0) = \vec{z}_0 = \begin{pmatrix} y_0 \\ \dot{y}_0 \\ \ddot{y}_0 \\ \vdots \\ y_0^{(n-1)} \end{pmatrix}$$

Bsp.: (6) $\ddot{y}(t) = f(t, y(t), \dot{y}(t))$

$y(t_0) = y_0$

$\dot{y}(t_0) = \dot{y}_0$

Reduktion auf System erster Ordnung:

$z_0(t) = y(t)$

$z_1(t) = \dot{y}(t) = \dot{z}_0(t)$

$z_2(t) = \ddot{y}(t) = \dot{z}_1(t) = f(t, z_0(t), z_1(t))$

Somit $\dot{\vec{z}} = \vec{g}(t, \vec{z}(t))$

wobei

$$\vec{z} = \begin{pmatrix} z_0(t) \\ z_1(t) \end{pmatrix}, \quad \vec{g}(t, \vec{z}(t)) = \begin{pmatrix} z_1 \\ f(t, z_0(t), z_1(t)) \end{pmatrix}$$

und AWe

$$\vec{z}(t_0) = \vec{z}_0 = \begin{pmatrix} y_0 \\ \dot{y}_0 \end{pmatrix}$$

Genau gleich kann man ein System von m gew. DGLen n-ter Ordnung auf ein System von $m \cdot n$ gew. DGLen erster Ordnung reduzieren.

Def.: Eine gew. DGL heisst **autonom**, falls die rechte Seite die Form $\vec{f} = \vec{f}(\vec{y}(t))$ hat (anstatt $\vec{f} = \vec{f}(t, \vec{y}(t))$).

Autonomisieren von DGLen

Betrachte folgende gew. DGL

$$\dot{\vec{y}} = \vec{f}(t, \vec{y}(t)) \quad , \quad \vec{y} \in \mathbb{R}^n$$

Durch einführen der neuen Variabel

$$\vec{z}(t) = \begin{pmatrix} \vec{y}(t) \\ t \end{pmatrix} = \begin{pmatrix} \vec{z}_1 \\ z_{n+1} \end{pmatrix}$$

und der rechten Seite

$$\vec{g}(\vec{z}(t)) = \begin{pmatrix} \vec{f}(t, \vec{y}(t)) \\ 1 \end{pmatrix} = \begin{pmatrix} \vec{f}(z_{n+1}, \vec{z}_1) \\ 1 \end{pmatrix}$$

erhält man eine autonome gew. DGL

$$\dot{\vec{z}} = \vec{g}(\vec{z})$$

Bsp.: (1) $\dot{y}(t) = y(t)^2 + t^2$

$$\vec{z}(t) = \begin{pmatrix} z_1(t) \\ z_2(t) \end{pmatrix} = \begin{pmatrix} y(t) \\ t \end{pmatrix}$$

$$\vec{g}(\vec{z}(t)) = \begin{pmatrix} y(t)^2 + t^2 \\ 1 \end{pmatrix} = \begin{pmatrix} z_1(t)^2 + z_2(t)^2 \\ 1 \end{pmatrix}$$

$$\rightsquigarrow \quad \dot{\vec{z}}(t) = \vec{g}(\vec{z}(t)) \quad \text{autonom } \checkmark.$$

II.2.1 Existenz, Eindeutigkeit und Stabilität

Im folgenden wollen wir uns ein paar theoretische Fragen bez. AWPen anschauen:

(i) Was muss erfüllt sein, dass es überhaupt Lösungen gibt? (Existenz)

(ii) Gibt's mehrere Lösungen? Unter welcher Bedingung gibt es nur eine? (Eindeutigkeit)

(iii) Wie hängt die Lösung vom AW ab? (Stabilität)

Betrachten wir folgendes (allgemeines) AWP

$$\vec{y}'(t) = \vec{f}(t, \vec{y}(t)) \qquad \text{(System von DGlen)}$$

$$\vec{y}(t_0) = \vec{y}_0 \qquad \text{(AW)}$$

wobei

$$\vec{y} : I = [t_0, T] \subset \mathbb{R} \to D \subset \mathbb{R}^n$$

'(Zeit)Intervall 'Zustands- oder Phasen-raum

$$\vec{f} : I \times D \to \mathbb{R}^n$$

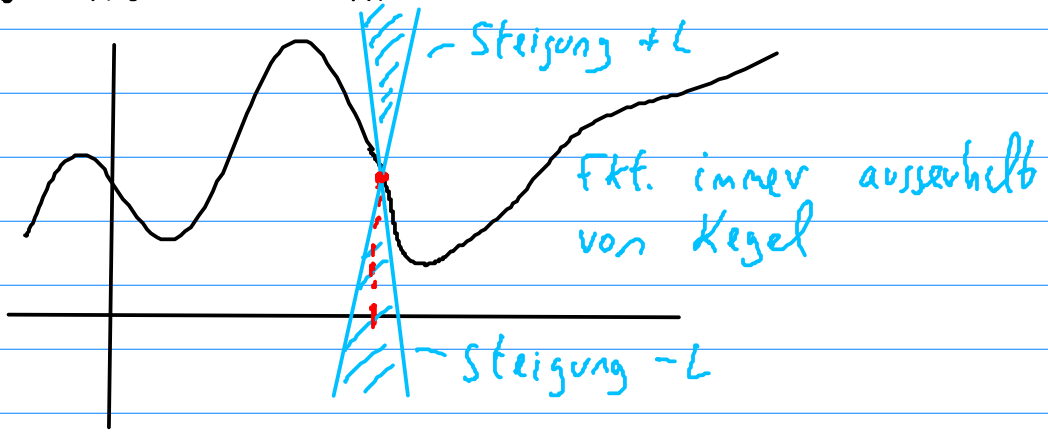
Um (i) und (ii) von oben sicherzustellen, benötigen wir einen etwas stärkeren Begriff der Stetigkeit

Def.: Eine Funktion $\vec{f}: I \times D \to \mathbb{R}^n$ ist Lipschitz-stetig in (Variabel) $\vec{y}$ mit der Lipschitz-Konstanten $L \geq 0$, wenn für alle $t \in I$ und $\vec{y}, \vec{z} \in D$ gilt

$$\| \vec{f}(t, \vec{y}) - \vec{f}(t, \vec{z}) \| \leq L \| \vec{y} - \vec{z} \|$$

Hier ist $\|\_\|$ eine Norm.
(1-, euklidische, max-Norm ...)

Bem.: (i) Lipschitz-Stetigkeit beschränkt wie stark eine Funktion um einen Punkt sich verändern kann



(ii) Stetige diff.'bare Funktionen sind Lipschitz-stetig: Setze $L = \max_{y \in (a,b)} |f'(y)|$

(iii) Auch nicht diff'bare Funktionen können Lipschitz-stetig sein

z.B.: $f(y) = |y|$

(iv) Manchmal auch Lipschitz-Bedingung genannt

Frage: Ist $f(y) = \sqrt{y}$, $y \in \mathbb{R}^+$, überall Lipschitz-stetig?

Der folgende Satz stellt nun (i) und (ii) sicher

Satz II.1: (Picard-Lindelöf)

Sei $\vec{f}$ stetig in $(t, \vec{y})$ und Lipschitz-stetig in $\vec{y}$ auf $[t_0, t_0 + \delta] \times D$, mit $\delta > 0$ und D eine Umgebung vom AW $\vec{y}_0$.

Dann existiert eine eindeutige Lösung $\vec{y}(t)$ des AWP

$$\dot{\vec{y}}(t) = \vec{f}(t, \vec{y}(t)), \quad \vec{y}(t_0) = \vec{y}_0$$

für zumindest eine kurze Zeit $[t_0, t_0 + \varepsilon]$, $\varepsilon > 0$.

Bsp.: (8) $\dot{y}(t) = 2\sqrt{|y(t)|}$

$y(0) = 0$

Lösungen: $y(t) = 0$

$y(t) = t \cdot |t|$

Grund: $f(t, y) = 2\sqrt{|y|}$ stetig, aber nicht Lipschitz stetig in $y = 0$!

(9) Bsp. (8) mit $y(1) = 1$

$\rightsquigarrow$ $y(t) = t \cdot |t|$ eindeutige Lösung !

(10) $\dot{y}(t) = y(t)^2$

$y(0) = y_0 > 0$

Lösung: $y(t) = \dfrac{y_0}{1 - y_0 \cdot t}$

Aber nur für $0 \leq t < \dfrac{1}{y_0}$!

(... "zumindest eine kurze Zeit" ...)

Grund: Lipschitz-Konstante L unbeschränkt

Der folgende Satz gibt Aufschluss über die eingehende Frage (iii) bez. der Abhängigkeit der Lösung von den AW:

Satz II.2: Die Funktion $\vec{f}$ sei Lipschitz-stetig mit Lipschitz-Konstanten L in einer Umgebung der AWe $\vec{y}_0$, $\vec{z}_0$ und seien $\vec{y}(t)$, $\vec{z}(t)$ die Lösungen der jeweiligen AWPs.
Dann gilt für $t \in [t_0, t_0 + \varepsilon]$

$$\|\vec{y}(t) - \vec{z}(t)\| \leq e^{L(t-t_0)} \|\vec{y}_0 - \vec{z}_0\|$$

Bem.: Satz II.2 stellt sicher, dass die Lösungen stetig vom AW abhängen.
Aber Lösungen können exponentiell in der Zeit auseinander driften

Bsp.: (11) $\dot{y}(t) = \lambda y(t)$, $y(t_0) = y_0 \rightsquigarrow y(t) = y_0 e^{\lambda(t-t_0)}$

$\dot{z}(t) = \lambda z(t)$, $z(t_0) = z_0 \rightsquigarrow z(t) = z_0 \cdot e^{\lambda(t-t_0)}$

Frage: Lipschitz-Konstante $L = ?$

$$\rightsquigarrow \quad y(t) - z(t) = e^{\lambda(t-t_0)} (y_0 - z_0)$$

↑ Vgl. Satz II.2

II.3 Runge-Kutta Verfahren

Betrachten wir das (skalare) AWP

$$\dot{y}(t) = f(t, y(t))$$

$$y(t_0) = y_0$$

Die Lösung können wir formal durch Integration erhalten

$$y(t) = y_0 + \int_{t_0}^{t} f(\tau, y(\tau)) \, d\tau$$

Integral-Gleichung... Nicht unbedingt "besser" wie mit der gew. DGL...

Idee: Verwende Quadratur für das Integral der rechten Seite f

Konkret, versuchen wir die Lösung ausgehend von t_0, y_0 zu einem Punkt $t_1, \tilde{y}_1$, wobei $t_1 = t_0 + h$, mittels Quadratur zu approx.:

$$y(t_1) = y_0 + \int_{t_0}^{t_1} f(\tau, y(\tau)) \, d\tau$$

$$= y_0 + h \cdot \int_0^1 f(t_0 + h\tau, y(t_0 + h\tau)) \, d\tau$$

... grenzen verschoben und skaliert ...

$$\approx y_0 + h \cdot \sum_{i=1}^{s} w_i \cdot f(t_0 + h \cdot c_i, y(t_0 + h \cdot c_i))$$

Quadratur Gewichte / Knoten

Problem: $y(t_0 + h \cdot c_i)$ immer noch unbekannt!

$\rightsquigarrow$ Müssen auch approx. werden ...

Mittelpunktsregel

Versuchen wir es mit der MR:

$$y(t_1) \approx y_0 + h \cdot f\left(t_0 + \frac{h}{2}, y\left(t_0 + \frac{h}{2}\right)\right)$$

Wie könnte man eine Approx. von $y(t_0 + \frac{h}{2})$ berechnen? ⇝ z.B. mit dem expliziten Euler-Verfahren!

$$y\left(t_0 + \frac{h}{2}\right) \approx y_0 + \frac{h}{2} f(t_0, y_0) = y_{1/2}$$

$\uparrow$

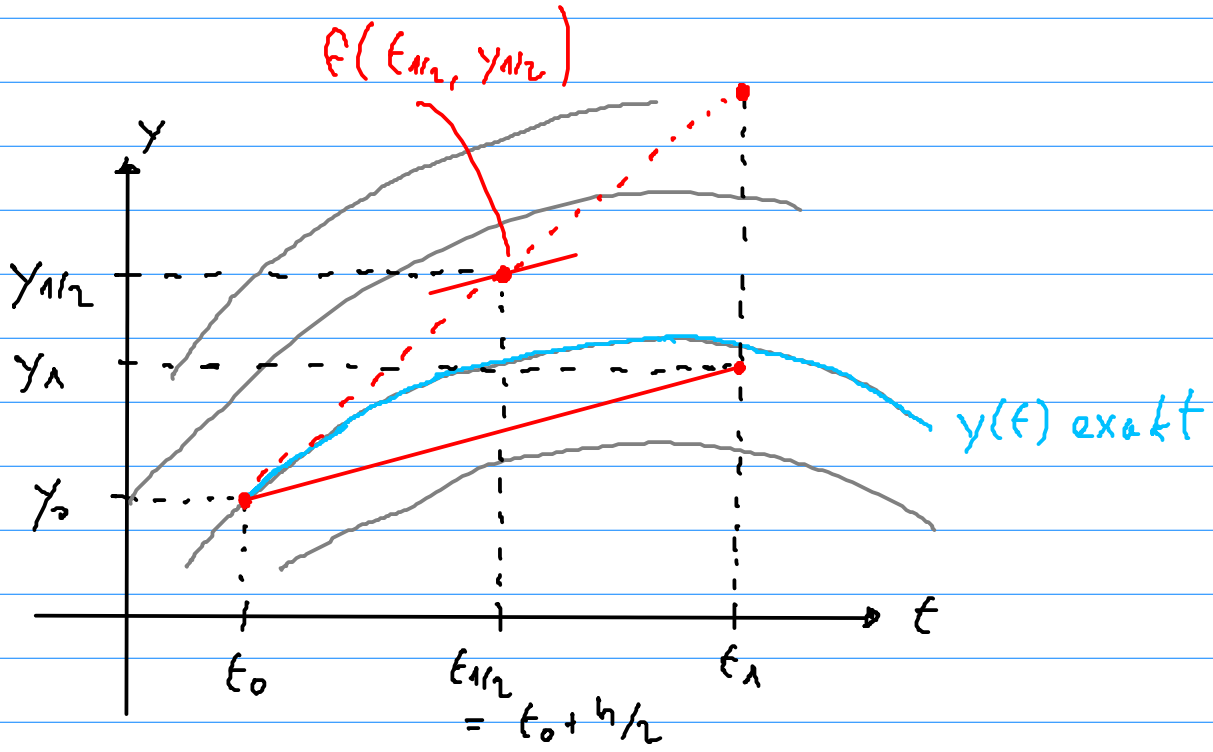
$y(t_0) = y_0$ AW

Setzen wir dies oben ein erhalten wir

$$y(t_1) \approx y_0 + h \cdot f\left(t_0 + \frac{h}{2}, y_{1/2}\right) = y_1$$

27.03.17

Graphisch:



Zusammenfassend, erhalten wir folgende Methode für AWPe:

$$k_1 = f(t_j, y_j)$$

$$k_2 = f(t_j + h/2,\ y_j + h/2 \cdot k_1)$$

$$y_{j+1} = y_j + h \cdot k_2$$

für $j = 0, 1, \ldots, N-1$. Dieses Verfahren ist bekannt als die verbesserte Polygonzug-Methode von Euler.

Oder auch: modifiziertes Euler Verfahren
explizite Mittelpunkts-Methode

Eine andere Möglichkeit wäre als QR die TR zu verwenden

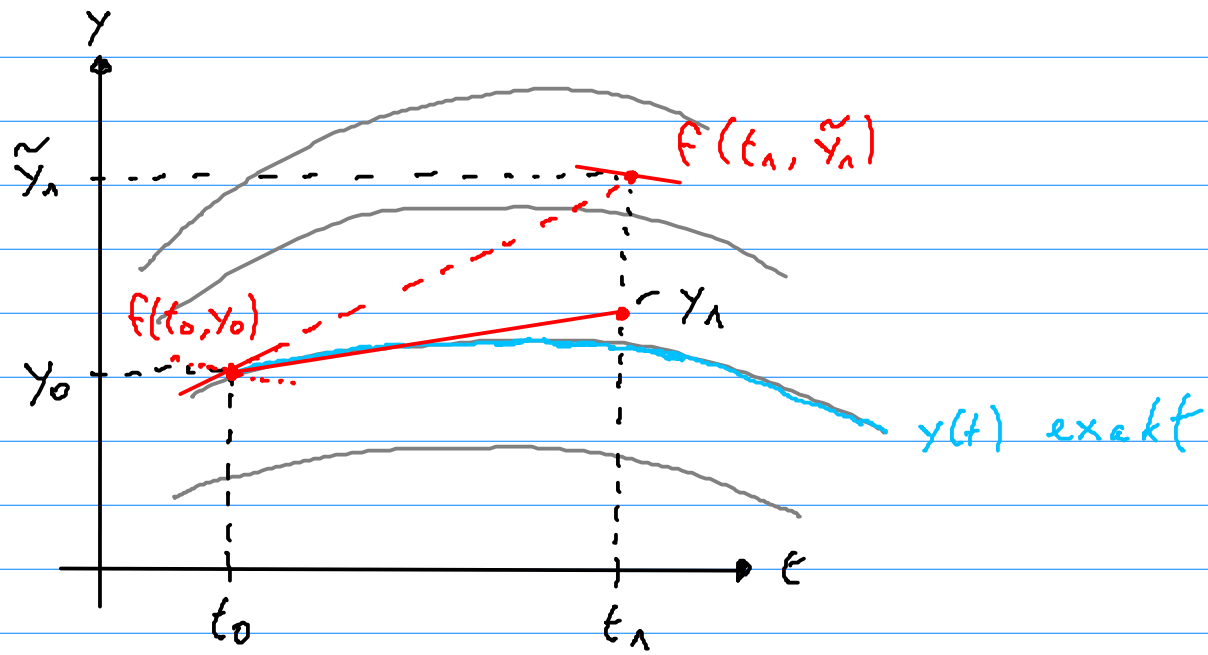
$$y(t_1) \approx y_0 + \frac{h}{2}\left(f(t_0, y_0) + f(t_1, y(t_1))\right)$$

$y(t_1)$?

Wie vorhin, approx. wir $y(t_1)$ mit dem Euler-Verfahren

$$y(t_1) \approx y_0 + h \cdot f(t_0, y_0) = \tilde{y}_1$$

Graphisch:



Wir erhalten folgendes Verfahren

$$k_1 = f(t_j, y_j)$$

$$k_2 = f(t_j + h, y_j + h \cdot k_1)$$

$$y_{j+1} = y_j + \frac{h}{2}(k_1 + k_2)$$

für $j = 0, 1, \ldots, N-1$. Dies ist das Verfahren von Heun.

Oder auch explizite Trapez-Methode.

Alle Verfahren die wir bis jetzt kennengelernt haben sind Teil einer grossen Familie von Einschrittverfahren (ESV):

Runge-Kutta Verfahren

Ein s-stufiges Runge-Kutta (Einschritt-) Verfahren (RK-ESV) ist definiert durch

$$y_{j+1} = y_j + h \cdot \sum_{i=1}^{s} b_i \cdot k_i$$

wobei die Stufen / Steigungen

$$k_i = f\left(t_j + c_i \cdot h,\ y_j + h \cdot \sum_{l=1}^{s} a_{il} \cdot k_l\right)$$

sind. Weiter nennt man

- s ... Anzahl Stufen
- c_i ... Knoten
- b_i ... Gewichte
- a_{il} ... Runge-Kutta Matrix / Koeffizienten

RK Verfahren schreibt man am besten in einem sog. **Butcher-Tableau (BT)**

$$\begin{array}{c|ccc} c_1 & a_{11} & a_{12} & \cdots & a_{1s} \\ c_2 & a_{21} & a_{22} & \cdots & a_{2s} \\ \vdots & & & & \\ c_s & a_{s1} & a_{s2} & \cdots & a_{ss} \\ \hline & b_1 & b_2 & \cdots & b_s \end{array} = \begin{array}{c|c} \vec{c} & A \\ \hline & \vec{b}^T \end{array}$$

$\vec{b}^T$: transponiert

Bsp.: (12) BT der verbesserten Polygonzug-Methode von Euler

2 Stufen, $s=2$

$$\begin{array}{c|cc} \vec{c} & A & \\ 0 & 0 & 0 \\ 1/2 & 1/2 & 0 \\ \hline & 0 & 1 \\ & & \vec{b}^T \end{array}$$

die Nullen der RK-Matrix A schreibt man (oft) nicht

$$k_1 = f\left(t_j + \overset{0}{\overset{\|}{c_1}} \cdot h,\ y_j + h \cdot \left(\overset{0}{\overset{\|}{a_{11}}} \cdot k_1 + \overset{0}{\overset{\|}{a_{12}}} \cdot k_2\right)\right)$$

$$= f(t_j, y_j)$$

$$k_2 = f\left(t_j + \overset{1/2}{\overset{\|}{c_2}} \cdot h,\ y_j + h \cdot \left(\overset{1/2}{\overset{\|}{a_{21}}} \cdot k_1 + \overset{0}{\overset{\|}{a_{22}}} \cdot k_2\right)\right)$$

$$= f\left(t_j + h/2,\ y_j + h/2 \cdot k_1\right)$$

$$y_{j+1} = y_j + h \cdot \left(\overset{0}{\overset{\shortparallel}{b_1}} \cdot k_1 + \overset{1}{\overset{\shortparallel}{b_2}} \cdot k_2 \right)$$

$$= y_j + h \cdot k_2$$

(13) BT der Methode von Heun

0	0	0
1	1	0
	1/2	1/2

(14) BT des Euler-Verfahrens

?	?	
		?

Obwohl wir bisher nur skalare AWPe betrachtet haben, kann man die RK-Verfahren sehr einfach auf allgemeine AWPe (also Systeme) verallgemeinern:

$$y \rightsquigarrow \vec{y}$$

$$f \rightsquigarrow \vec{f}$$

Desweiteren ist die Def. der RK-Verfahren sehr allgemein. Sie umfasst auch sog. implizite Verfahren

Bsp.: (1S) Implizite Mittelpunkts-Methode

$1/2$	$1/2$
	1

$$k_1 = f\left(t_j + \frac{h}{2}, y_j + \frac{h}{2} \cdot k_1\right)$$

implizite Def. von k_2

$$y_{j+1} = y_j + h \cdot k_1$$

d.h. man muss eventuell ein i.A. nicht-lineares Gl.-System lösen!

Implizite Methoden benötigt man bei sog. steifen Problemen $\rightsquigarrow$ Kap. V.

Explizite RK-Verfahren haben ein BT der Form

c_1	0	0	⋯	0
c_2	a_{21}	0	⋯	0
⋮	⋮	⋱	⋱	⋮
c_s	a_{s1}	a_{s2}	⋯ $a_{s,s-1}$	0
	b_1	b_2	b_{s-1}	b_s

D.h. A ist eine Untere-Dreiecksmatrix mit Nullen auf der Diagonalen.

Das wohl bekannteste RK-Verfahren ist das sog. klassische RK-Verfahren

0	0	0	0	0
$1/2$	$1/2$	0	0	0
$1/2$	0	$1/2$	0	0
1	0	0	1	0
	$1/6$	$2/6$	$2/6$	$1/6$

Graphisch ⇝ Slides

Oft wird es auch nur als DAS RK-Verfahren oder RK4 bezeichnet.

↖ 4 Stufen

II.4 Fehlerbetrachtungen für ESV

Hier wollen wir die Genauigkeit von ESV untersuchen. Dazu betrachten wir ein allgemeines ESV der Form

$$y_{j+1} = y_j + h \cdot \phi(t_j, y_j, h)$$

wobei ϕ die sog. Verfahrens- oder Inkrements-Funktion ist.

Bem.: (i) Bei RK-ESV ist

$$\phi(t_j, y_j, h) = \sum_{i=1}^{s} b_i \cdot k_i$$

(ii) Bei einem expliziten ESV kann man ϕ einfach durch einsetzen berechnen.
Bei einem impliziten ESV muss man um ϕ zu berechnen i.A. nicht-lineare Gleichungssysteme lösen.

Im folgenden betrachten wir das (skalare) AWP

$$\dot{y}(t) = f(t, y(t)) \qquad \text{(DGL)}$$

$$y(t_0) = y_0 \qquad \text{(AW)}$$

auf dem Intervall $[t_0, T]$.

Anfangs- End-Zeit

Mit $y(t)$ bezeichnen wir die exakte Lösung.
Wir approx. die Lösung durch diskretisieren des Zeit-Intervalls in N Teil-Intervalle

$$t_j = t_0 + j \cdot h$$

mit der/dem Schrittweite / Zeitschritt

$$h = \frac{T - t_0}{N}$$

Die approx. Lösung zur Zeit t_j bezeichnen wir mit y_j.

Also

$$y(t_j) \approx y_j$$

Def.: Der **globale Diskretisierungsfehler** (GDF) zur Zeit t_j ist definiert durch

$$e_j = y(t_j) - y_j$$

($y(t_j)$: exakte Lösung zur Zeit t_j; y_j: approx. Lösung zur Zeit t_j)

D.h. e_j ist der Fehler (die Differenz zwischen der exakten und approx. Lösung) nach j Schritten.

Def.: Ein ESV heisst **konvergent von der Ordnung** p (oder hat **Konvergenz-Ordnung** (KO) p), wenn gilt

$$\epsilon = \max_{j=0,\dots,N} \underbrace{|y(t_j) - y_j|}_{e_j} = \mathcal{O}(h^p)$$

für h klein genug.

Bsp.: (16) Empirische KO der bisher angetroffenen RK-ESV

$\rightsquigarrow$ Slides

Euler $\sim \mathcal{O}(h^?)$

verb. Euler $\sim \mathcal{O}(h^?)$

Heun $\sim \mathcal{O}(h^?)$

RK4 $\sim \mathcal{O}(h^?)$

Um die KO einer ESV theoretisch zu untersuchen benötigen wir:

Def.: Der **lokale Diskretisierungsfehler** (LDF) zur Zeit t_j ist definiert durch

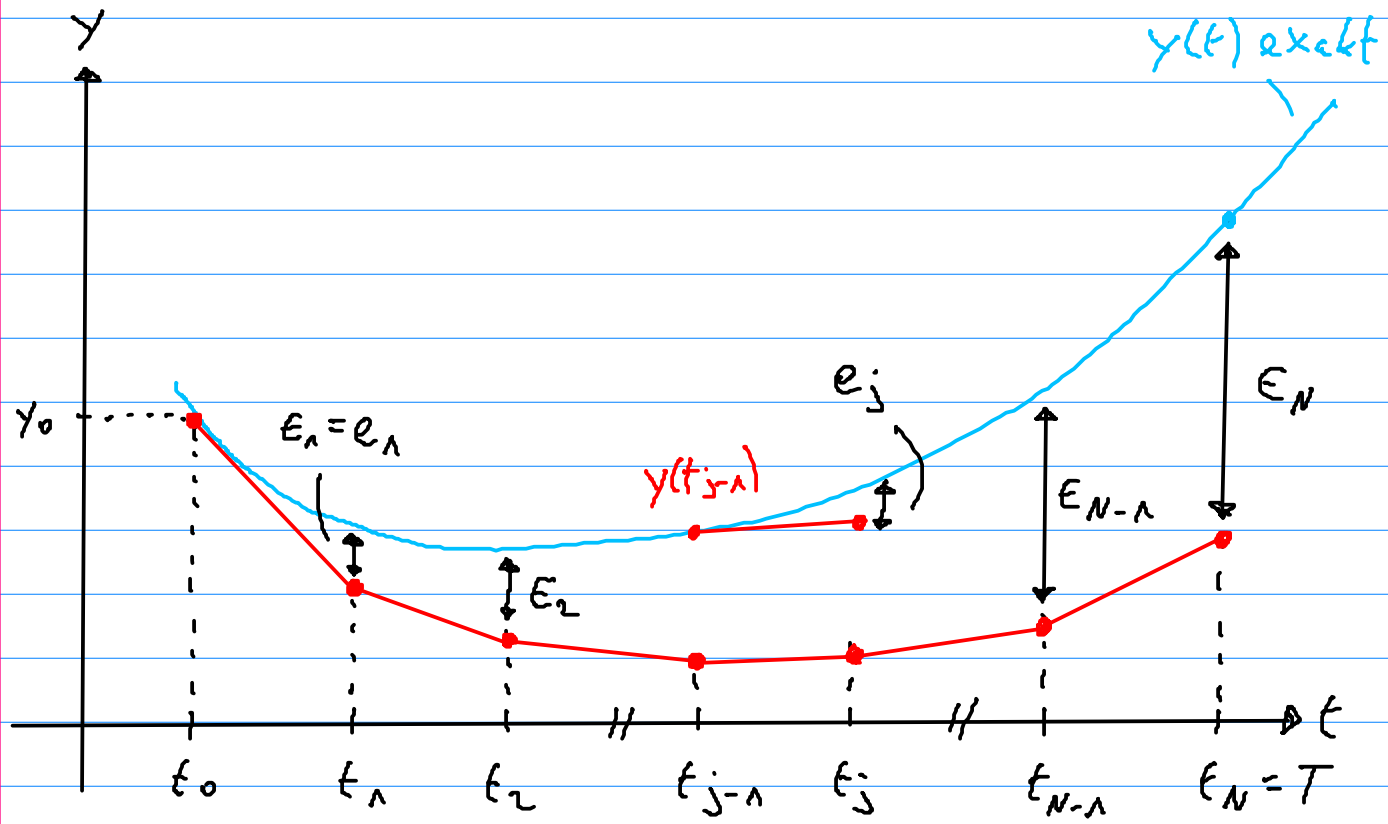
$$e_j = y(t_j) - \left(y(t_{j-1}) + h \cdot \phi(t_{j-1}, y(t_{j-1}), h) \right)$$

exakte Lösung bei t_{j-1} !

D.h. e_j ist der Fehler bei t_j nach einem Schritt ausgehend von der exakten Lösung bei t_{j-1}.

03.04.17

Graphisch:



Def.: Der Konsistenzfehler (KF) zur Zeit t_j ist definiert durch

$$\tau_j = \frac{e_j}{h}$$

$$= \frac{y(t_j) - y(t_{j-1})}{h} - \phi(t_{j-1}, y(t_{j-1}), h)$$

Def.: Ein ESV heisst **konsistent von der Ordnung** p (oder hat **Konsistenzordnung** p), wenn gilt

$$\tau = \max_{j=0,\dots,N} |\tau_j| = \mathcal{O}(h^p)$$

für h klein genug.
Ein ESV heisst **konsistent** falls $p \geq 1$.

LDF, KF und Konsistenzordnung sind lokale Grössen. Diese kann man (relativ) einfach mittels Taylor-Entwicklungen bestimmen:

$$\tau_{j+1} = \frac{y(\underbrace{t_j + h}_{t_{j+1}}) - y(t_j)}{h} - \phi(t_j, y(t_j), h)$$

$$= \frac{y(t_j) + h\cdot\dot{y}(t_j) + \frac{h^2}{2}\cdot\ddot{y}(t_j) + \dots - y(t_j)}{h}$$

$$- \left(\phi(t_j, y(t_j), 0) + h\cdot\dot{\phi}(t_j, y(t_j), 0) + \frac{h^2}{2}\ddot{\phi}(t_j, y(t_j), 0) + \dots \right)$$

$$= \dot{y}(t_j) - \phi(t_j, y(t_j), 0)$$

$$+ \frac{h}{2}\left(\ddot{y}(t_j) - 2\cdot\dot{\phi}(t_j, y(t_j), 0) \right)$$

$$+ \frac{h^2}{6}\left(\dddot{y}(t_j) - 3\cdot\ddot{\phi}(t_j, y(t_j), 0) \right)$$

$$+ \dots + \mathcal{O}(h^p) + \dots$$

Berechnen wir zunächst die Ableitungen der Lösung

$$\dot{y}(t) \overset{\text{DGL}}{=} f(t, y(t))$$

$$\ddot{y}(t) = \frac{d}{dt} f(t, y(t))$$

$$\overset{\text{Ketten-Regel!}}{=} \frac{\partial f}{\partial t}(t, y(t)) + \frac{\partial f}{\partial y}(t, y(t)) \cdot \underbrace{\dot{y}(t)}_{= f(t, y(t)) \text{ DGL!}}$$

$$= \frac{\partial f}{\partial t}(t, y(t)) + \frac{\partial f}{\partial y}(t, y(t)) \cdot f(t, y(t))$$

$$\dddot{y}(t) = \frac{d}{dt}\left(\frac{\partial f}{\partial t}(t, y(t)) + \frac{\partial f}{\partial y}(t, y(t)) \cdot f(t, y(t)) \right)$$

$$= \frac{\partial^2 f}{\partial t^2}(t, y(t)) + \frac{\partial^2 f}{\partial y \partial t}(t, y(t)) \cdot \dot{y}(t) \quad \left(= f(t, y(t))\right)$$

$$\left(\frac{\partial^2 f}{\partial t \partial y} = \frac{\partial^2 f}{\partial y \partial t}\right) + \left(\frac{\partial^2 f}{\partial t \partial y}(t, y(t)) + \frac{\partial^2 f}{\partial y^2}(t, y(t)) \cdot \underbrace{\dot{y}(t)}_{= f(t, y(t))} \right) \cdot f(t, y(t))$$

$$+ \frac{\partial f}{\partial y}(t, y(t)) \cdot \left(\frac{\partial f}{\partial t}(t, y(t)) + \frac{\partial f}{\partial y}(t, y(t)) \cdot \underbrace{\dot{y}(t)}_{= f(t, y(t))} \right)$$

$$= \frac{\partial^2 f}{\partial t^2}(t, y(t)) + 2 \cdot \frac{\partial^2 f}{\partial t \partial y}(t, y(t)) \cdot f(t, y(t))$$

$$+ \frac{\partial^2 f}{\partial y^2}(t, y(t)) \cdot f(t, y(t))^2 + \frac{\partial f}{\partial t}(t, y(t)) \cdot \frac{\partial f}{\partial y}(t, y(t))$$

$$+ \left(\frac{\partial f}{\partial y}(t, y(t)) \right)^2 \cdot f(t, y(t))$$

Usw.

Zur Berechnung der Entwicklung der Verfahrens-Funktion benötigt man die zwei-dimensionale Taylor-Reihe:

$$f(t+\Delta t, y+\Delta y) = f(t,y) + \frac{\partial f}{\partial t}(t,y)\cdot\Delta t + \frac{\partial f}{\partial y}(t,y)\cdot\Delta y + \frac{1}{2}\frac{\partial^2 f}{\partial t^2}(t,y)\cdot\Delta t^2 + \frac{\partial^2 f}{\partial t\partial y}(t,y)\cdot\Delta t\cdot\Delta y + \frac{\partial^2 f}{\partial y^2}(t,y)\cdot\Delta y^2 + \dots$$

$\sim h$

Am besten ein paar Beispiele

Bsp.: (17) Euler-Verfahren

$$\phi(t_j, y(t_j), h) = f(t_j, y(t_j))$$

$$\rightsquigarrow \quad \phi(t_j, y(t_j), 0) = f(t_j, y(t_j)) = \dot{y}(t_j) \;\checkmark$$

$$\dot{\phi}(t_j, y(t_j), 0) = \;?$$

Und damit Konsistenzordnung

$$p = \;?$$

D.h. $\tau = \mathcal{O}(h^{?})$ wie gemessen in Bsp. (16)!

10.04.17

(18) Verbesserter Euler

$$k_1 = f(t_j, y(t_j))$$

$$k_2 = f(t_j + h/2,\ y(t_j) + h/2\, k_1)$$

$$y_{j+1} = y(t_j) + h \cdot k_2$$

Entwickeln der Verfahrens-Funktion

$$\phi(t_j, y(t_j), h) = f(t_j + \tfrac{h}{2},\ y(t_j) + \tfrac{h}{2} k_1)$$

$$= f(t_j, y(t_j)) \leftarrow \phi(t_j, y(t_j), 0)$$

$$\dot\phi(t_j, y(t_j), 0) \cdot h \Big\{ + \frac{\partial f}{\partial t}(t_j, y(t_j)) \cdot \frac{h}{2} + \frac{\partial f}{\partial y}(t_j, y(t_j)) \cdot \frac{h}{2} k_1$$

$$\ddot\phi(t_j, y(t_j), 0) \cdot \frac{h^2}{2} \Big\{ + \frac{1}{2}\frac{\partial^2 f}{\partial t^2}(t_j, y(t_j)) \cdot \left(\frac{h}{2}\right)^2 + \frac{\partial^2 f}{\partial t \partial y}(t_j, y(t_j)) \cdot \left(\frac{h}{2}\right)^2 \cdot k_1 + \frac{1}{2}\frac{\partial^2 f}{\partial y^2}(t_j, y(t_j)) \cdot \left(\frac{h}{2} k_1\right)^2$$

$$+ \dots$$

Durch Vergleich

$$\phi(t_j, y(t_j), 0) = f(t_j, y(t_j)) = \dot y(t_j) \quad \checkmark$$

2. $$\dot\phi(t_j, y(t_j), 0) = \left(\frac{\partial f}{\partial t}(t_j, y(t_j)) + \frac{\partial f}{\partial y}(t_j, y(t_j)) \cdot f(t_j, y(t_j)) \right)$$

$$= \ddot y(t_j) \quad \checkmark$$

3. $$\ddot\phi(t_j, y(t_j), 0) = \dots \quad \times$$

Und damit Konsistenzordnung $p = 2$

D.h. $\tau = \mathcal{O}(h^2)$ wie gemessen in Bsp. (16)!

Fazit: die lokalen Fehlermasse (LDF, KF, Konsistenzordnung) lassen sich mittels Taylor-Reihenentwicklungen abschätzen

└ Dabei wird natürlich immer stillschweigend vorausgesetzt, dass die rechte Seite der DGL ist genügend glatt! ← f

Aber eigentlich sind wir ja an globalen Fehlergrössen (GDF & KO) interessiert. Bsp. (16), (17) und (18) deuten bereits an, dass zwischen den lokalen und globalen Grössen ein Zusammenhang besteht. Der folgende Satz untermauert dies:

Satz II.3: Falls die rechte Seite der DGL $\vec{f}(t,\vec{y})$ und die Verfahrens-Funktion $\vec{\phi}(t,\vec{y},t)$ Lipschitz-stetig in $\vec{y}$ sind, dann gilt für das ESV folgende (globale) Fehlerabschätzung

$$\epsilon = \max_{j=0,\dots,N} \|\vec{y}(t_j) - \vec{y}_j\|$$

$$\leq \left(\|\vec{y}(t_0) - \vec{y}_0\| + \sum_{j=1}^{N} \|\vec{e}_j\| \right) \cdot e^{\tilde{L}(t_N - t_0)}$$

AW fehler

Fehler in jeden Schritt summieren sich schlimmstenfalls

wobei $\tilde{L}$ die Lipschitz-Konstante der Verfahrens-Funktion $\vec{\phi}$ ist.

Obige Aussage impliziert

Konsistenz der Ordnung p

$\Rightarrow$ Konvergenz der Ordnung p

Bem.: (i) Satz II.3 macht eine Aussage über die Stabilität von ESV, d.h. eine kontrollierte Fehlerfortpflanzung (AW + LDF ✓)

(ii) $\tilde{L}$ hängt über $\vec{\phi}$ mit der rechten Seite $\vec{f}$ zusammen