III. Adaptive Schrittweitensteuerung

<u>Ziel:</u> Eine gewünschte Genauigkeit erreichen mit möglichst wenig (Rechen-) Aufwand (Effizienz!) $\rightsquigarrow$ adaptive Schrittweite
$h \rightsquigarrow h_j$

- Fehlerabschätzungen & Toleranzen (abs., rel.)
- Schrittweitenhalbierung & Eingebettete RK-Verfahren
- Adaptive Schrittweitensteuerung

<u>Wozu:</u> Praxis relevant!

III.1 Fehlerbetrachtungen und Toleranzen

Gegeben ein skalares AWP der Einfachheit halber!

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

$$y(t_0) = y_0$$

für $t \in [t_0, T]$.

Wir wollen dieses AWP näherungsweise bis auf eine Genauigkeit / Toleranz mit möglichst wenigen Integrationsschritten lösen

$$\varepsilon = \max_{j=0,\dots,N} |\underbrace{y(t_j)}_{\text{exakt}} - \underbrace{y_j}_{\text{approx.}}| < TOL$$

Lösung

Dieses Problem ist nicht mehr ganz so "einfach" zu behandeln wie bei der Quadratur weil gemachte Fehler sich verstärken können...

Satz II.3 gibt uns die Abschätzung

$$\varepsilon \leq \left(\underbrace{\cancel{|y(t_0) - y_0|}}_{\text{Fehler in AW}} + \underbrace{\sum_{j=1}^{N} |e_j|}_{\text{LDF}} \right) \cdot \underbrace{\cancel{e^{L(T-t_0)}}}_{\text{"vernachlässigen"}} < TOL$$

16.04.18

Also

$$\sum_{j=1}^{N} |e_j| < TOL$$

$$|e_1| + \ldots + |e_j| + \ldots + |e_N| < \underbrace{TOL}_{\text{global}}$$

__Idee:__ Wähle $\underbrace{|e_j| < tol}_{\text{lokal}}$

Damit $\displaystyle\sum_{j=1}^{N} |e_j| = \underbrace{N}_{?} \cdot tol < TOL$

__Problem:__ Wir kennen die Anzahl Schritte N nicht im voraus!

Bei äquidistanter Schrittweite gilt

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

Damit liegt nahe

$$\frac{T-t_0}{h_j} \cdot tol_j \approx TOL$$

Schrittweite im j-ten Schritt

$$tol_j \approx h_j \cdot \frac{TOL}{T-t_0}$$

Also wähle h_j so, dass

$$|e_j| \lesssim tol_j \qquad (TK1)$$

Soweit so gut, aber es stellt sich dennoch die Frage ob die Vernachlässigung des exponentiellen Faktors eine gute Idee war...

I.A. NEIN!

D.h. es ist i.A. schwierig (bzw. unmöglich) von den LDF_n auf die GDF_n zu schliessen

Deshalb könnte man auch einfach nur die lokalen Fehler mit lokalen Toleranzen kontrollieren, z.B.

$$|e_j| \lesssim atol \qquad \text{(absolute)} \qquad (TK2)$$

$$|e_j| \lesssim |y_{j-1}| \cdot rtol \qquad \text{(relative)} \qquad (TK3)$$

$$|e_j| \lesssim atol + |y_{j-1}| \cdot rtol \qquad \text{(abs. + rel.)} \qquad (TK4)$$

Bem.: Es gibt keine Garantie, dass die Näherungs-Lösung die gewünschte Genauigkeit hat!
In der Praxis liefert die adaptive Schrittweitensteuerung häufig dennoch gute Resultate.

III.2 Lokale Fehlerschätzer

Nun müssen wir die LDF e_j lokal schätzen. Für diese haben wir die a priori Fehlerschätzer (s. II.4)

$$|e_j| = |y(t_j) - y(t_{j-1}) - h_j \cdot \phi(t_{j-1}, y(t_{j-1}), h_j)|$$
$$\approx \mathcal{O}(h^{p+1}) = C \cdot h^{p+1}$$

für ein ESV der KO p.

In der Konstanten C stecken höhere Ableitungen der (uns unbekannten!) Lösung. Deshalb, wie schon bei der adaptiven Quadratur, sind diese a priori Fehlerschätzer nicht direkt brauchbar.

Idee: Vergleiche das Resultat eines Verfahrens mit dem Resultat eines genaueren Verfahren (sog. Kontroll-Verfahren (KV))

Wie bei der adaptiven Quadratur überlegt man sich folgende Möglichkeiten.

III.2.1 Schrittweitenhalbierung

Gegeben ein Verfahren der KO p, berechne im j-ten Schritt ein Schritt mit Schrittweite h

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

und 2 Schritte mit halber Schrittweite $\frac{h}{2}$ (KV)

Hut für KV!

$$\hat{y}_{j+1/2} = y_j + \frac{h}{2} \cdot \phi(t_j, y_j, h/2)$$

$$\hat{y}_{j+1} = \hat{y}_{j+1/2} + \frac{h}{2} \phi(\underbrace{t_{j+1/2}}_{t_j + h/2}, \hat{y}_{j+1/2}, h/2)$$

Der LDF ist dann

$$|e_{j+1}| = |\underbrace{y(t_{j+1})}_{\text{Lösung mit AW } y_j} - y_{j+1}| = C \cdot h^{p+1} + \mathcal{O}(h^{p+2})$$

und

~Satz II.3

$$|\hat{e}_{j+1}| = |y(t_{j+1}) - \hat{y}_{j+1}| = 2 \cdot C \cdot \left(\frac{h}{2}\right)^{p+1} + \mathcal{O}(h^{p+2})$$

$$= \frac{1}{2^p} C \cdot h^{p+1} + \mathcal{O}(h^{p+2})$$

$$\approx \frac{|e_{j+1}|}{2^p}$$

Nun wollen wir $|e_{j+1}|$ und $|\hat{e}_{j+1}|$ in Verbindung mit der Differenz der Resultate $|\hat{y}_{j+1} - y_{j+1}|$ bringen:

$$|e_{j+1}| = |y(t_{j+1}) \overbrace{- \hat{y}_{j+1} + \hat{y}_{j+1}}^{+0 \text{ Trick}} - y_{j+1}|$$

$$\overset{\Delta\text{-Ug.}}{\leq} \underbrace{|y(t_{j+1}) - \hat{y}_{j+1}|}_{\frac{|e_{j+1}|}{2^p}} + |\hat{y}_{j+1} - y_{j+1}|$$

Auflösen nach $|e_{j+1}|$

$$\rightsquigarrow \quad |e_{j+1}| \approx \frac{2^p}{2^p - 1} |\hat{y}_{j+1} - y_{j+1}| = \varepsilon_{j+1}$$

Schätzungen!

und

$$|\hat{e}_{j+1}| \approx \frac{1}{2^p - 1} |\hat{y}_{j+1} - y_{j+1}| = \hat{\varepsilon}_{j+1}$$

Dies sind dann wieder sog. a posteriori Fehlerschätzer.

<u>Bsp.:</u> (1) Euler mit h + Euler mit $h/2$

(2) Heun " + Heun "

III.2.2 Eingebettete RK-Verfahren

Der Fehler wird geschätzt durch Vergleich eines Verfahrens der KO p mit einem Verfahren höherer KO, z.B. $p+1$, (das KV). Um dies effizient zu berechnen, (d.h. mit möglichst wenigen Auswertungen der rechten Seite f) verwendet man sog. eingebettete RK-Verfahren mit Butcher-Tableaux der Form

$$
\begin{array}{c|ccccc}
c_1 & & & & & \\
c_2 & a_{21} & & & & \\
c_3 & a_{31} & a_{32} & & & \\
\vdots & \vdots & \vdots & \ddots & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \cdots & b_{s-1} & b_s \\
\hline
 & \hat{b}_1 & \hat{b}_2 & \cdots & \hat{b}_{s-1} & \hat{b}_s
\end{array}
\quad \leftarrow \text{KV}
$$

Bsp.: (3) Euler + Heun Verfahren

$$
\begin{array}{c|cc}
0 & & \\
1 & 1 & \\
\hline
 & 1 & 0 \\
\hline
 & 1/2 & 1/2
\end{array}
\quad
\begin{array}{l}
\\ \\ \leftarrow \text{Euler} \\ \leftarrow \text{Heun}
\end{array}
$$

(4) Dormand und Prince Verfahren (MATLAB ode45)

Der a posteriori Fehlerschätzer ist dann

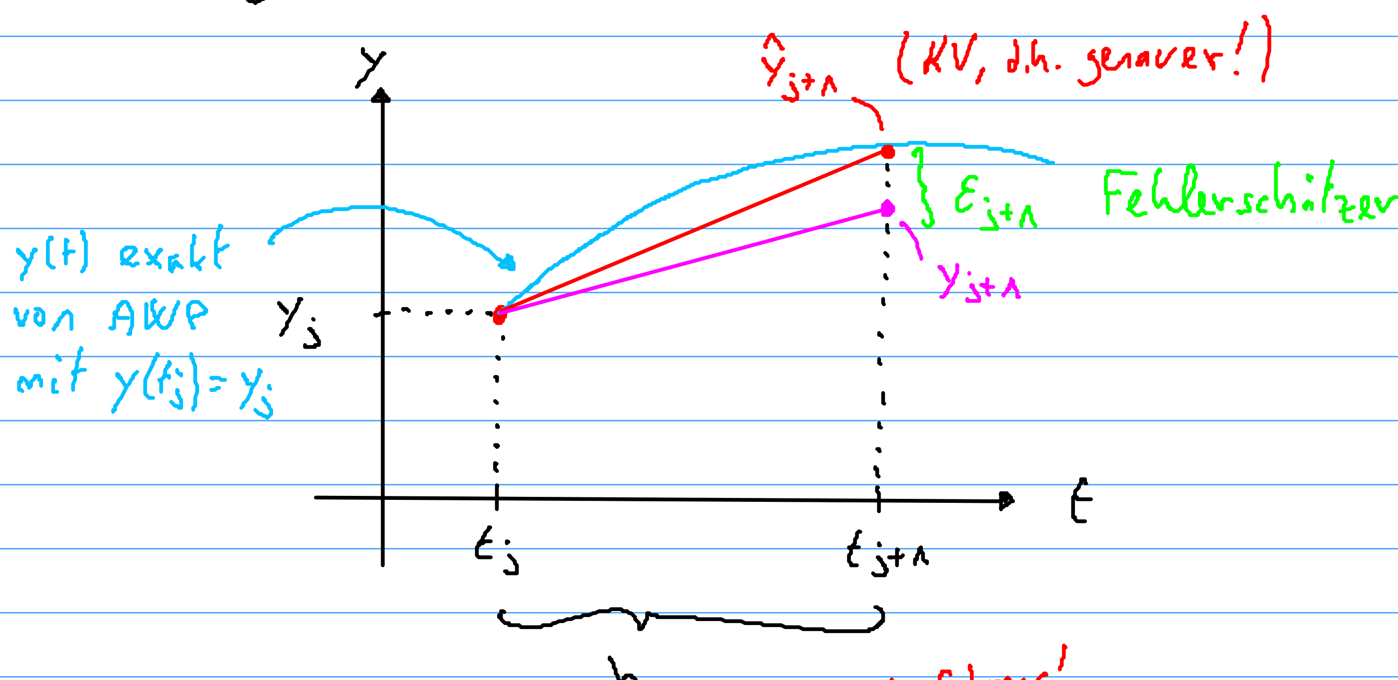
$$|e_{j+1}| \approx |\hat{y}_{j+1} - y_{j+1}| = \varepsilon_{j+1}$$

Bem.: Hier haben wir nur ein Fehlerschätzer für das weniger genaue Verfahren (D.h. kein Schätzer für $\hat{e}_{j+1}$ des KVs!)

III.3 Adaptive Schrittweitensteuerung

Ziel: Wähle Schrittweite klein genug um gewünschte lokale Präzision zu erreichen und gross genug um effizient zu sein

Ausgangslage im j-ten Schritt



KO des Verfahrens!

Haben: $\varepsilon_{j+1} \approx C \cdot h^{p+1} \rightsquigarrow C \approx \dfrac{\varepsilon_{j+1}}{h^{p+1}}$

Wollen: $tol_{j+1} \approx C \cdot H^{p+1} \rightsquigarrow H \approx h \cdot \left(\dfrac{tol_{j+1}}{\varepsilon_{j+1}} \right)^{1/(p+1)}$

H ist ein Schrittweiten-Vorschlag um

$$\varepsilon_{j+1} \approx tol_{j+1}$$

zu erreichen.

Frage: ① Wenn $\varepsilon_{j+1} > tol_{j+1}$, ist

$$H \lesseqgtr h \text{ ?}$$

② Wenn $\varepsilon_{j+1} < tol_{j+1}$?

Daraus überlegt man sich folgende adaptive Schrittweiten-Wahl

"Sicherheits"-Faktor

$$H = h \cdot \min\left(\underbrace{facmax}_{\text{begrenze Vergrösserung}},\ \max\left(\underbrace{facmin}_{\text{begrenze Verkleinerung}},\ fac \cdot \left(\frac{tol}{\varepsilon}\right)^{1/p+1} \right)\right)$$

23.08.18

Adaptive Schrittweitensteuerung (~MATLAB Pseudo-Code)

function $[t, y] =$ adapt_ode$(\dots)$

$j = 0$

$h = h0$

$tol = atol + |y_0| \cdot rtol$

$\varepsilon = tol$

while $(t_j < T)$

Parameter

$$h = h \cdot \min\left(facmax,\ \max\left(facmin,\ fac \cdot \left(\frac{tol}{\varepsilon}\right)^{1/p+1}\right)\right)$$

("Details" zu h → Übung)

$y_{j+1} = y_j + \dots$

$\hat{y}_{j+1} = y_j + \dots$ mit h-Halb.-

$\varepsilon = |y_{j+1} - \hat{y}_{j+1}|$ oder $\hat{\varepsilon} = \dots$ / Methode

$tol = atol + \max(|y_j|, |\hat{y}_{j+1}|) \cdot rtol$ (TK5)

if $(\varepsilon < tol)$ (TK1...4 auch möglich)

$t_{j+1} = t_j + h$

$y_{j+1} = \hat{y}_{j+1}$ ← Nehmen das KV!

$j = j+1$

end

end

Bem.: (i) Es wird stets das genauere Resultat $\hat{y}_{j+1}$ genommen. Obwohl man im Fall eines eingebetteten RK-Verfahren eigentlich keinen Fehlerschätzer $\hat{E}_{j+1}$ hat.

(ii) Es gibt keine Garantie, dass der globale Fehler $E < tol$ ist!