Numerische Methoden

Naoki Pross – nproß@student.ethz.ch

27. März 2023

Zusammenfassung

These are my (kinda crappy) notes I take during the lecture of Numerical Method taught Dr. Roger Käppeli at ETH Zürich.

1 Numerische Quadratur

Ziel:

- Approximieren von bestimmten Integralen

$$Q[I] \approx \int_a^b f(x)dx$$

- Genauigkeit der Approximation
- Fundamentale Konzepte der Numerik
- Newton-Cates, Gauss, Adaptive Quadratur
- zweidimensionale Quadratur

Wozu: Oft ist $\int_a^b f(x)dx$ nicht exact berechenbar Idee:

1.1 Polynomiale Interpolation

Gegeben $n+1$ paarweise verschiedene Stützstellen / Knoten $x_0, x_1, \dots, x_n$ und zugehörige Stützwerte $y_0, y_1, \dots, y_n$: finde das Polynom n-ten Grades

$$p_1(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \quad \in \mathbb{P}_n,$$

welches die sogenannte Interpolationsbedingung (IB) erfüllt:

$$p_1(x_j) = y_j \qquad (j = 0, 1, \dots, n).$$

Die $n+1$ Koeffizienten $a_0, a_1, \dots, a_n$ des sogenannten Interpolationspolynom (IP) ergeben sich einfach aus der IB $\rightsquigarrow$ Lineares Gleichungssystem (LGS).

MATLAB: `p = polyfit(x, y, n)`, Auswertung mit `polyval()`.

Das IP kann man auch direkt mittels der Lagrange'schen Interpolationsformel (LI)

$$p_1(x) = \sum_{j=1}^{n} y_j L_j^n(x) \quad \text{wobei} \quad L_j^n(x) = \prod_{i=0, i\neq j}^{n} \frac{x - x_i}{x_j - x_i}$$

die sog. Lagrandge-Polynome (LP) sind. Die LP haben die folgende Eigenschaften:

- (LP1) $L_j^n(x)$ sind Polynome n-ten Grades
- (LP2) $L_j(x_k) = \delta_{jk}$

LP2 ist der Grund wieso die LI die IB erfüllt.

$$p_n(x_i) = \sum_{j=0}^{n} y_j L_j^n(x) = 0 + \cdots + 0 + y_i \underbrace{L_i^n(y_i)}_{1} + 0 + \cdots + 0 = y_i$$

1.2 Interpolationsfehler

Sei also $f : I = [0, L] \to \mathbb{R}$ und bezeichnen mit

$$p[f|x_0, \dots, x_n](x) \in \mathbb{P}_n$$

das IP welcher die IB

$$p[f|x_0, \dots, x_n](x_j) = f(x_j) \quad (j = 0, \dots, n)$$

erfüllt.

Für f $(n+1)$-mal stetig differenzierbar lässt sich zeigen, dass es für jedes $x \in I$ ein $\xi = \xi(x) \in I$ gilt mit

$$e(x) = f(x) - p[f|x_0, \dots, x_n](x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^{n} (x - x_j).$$

$e(x)$ ist eine Fehlerfunktion über das Ganze Interpolations-Intervall I. Oft ist man (nur) am grössten Fehler über I interessiert:

$$\|e\|_\infty = \max_{x\in I}|e(x)| = \max_{x\in I}\left|\frac{f^{(n+1)}(\xi(x))}{(n+1)!}\prod_{j=0}^{n}(x-x_j)\right| \le \left(\max_{x\in I}\left|\frac{f^{n+1}(\xi(x))}{(n+1)!}\right|\right)\left(\max_{x\in I}\left|\frac{f^{n+1}(\xi(x))}{(n+1)!}\right|\right)$$
$$= \frac{\|f^{(n+1)}\|_\infty}{(n+1)!}\left\|\prod_{j=0}^{n}\underbrace{(x-x_j)}_{\le b-a}\right\|_\infty \le \frac{\|f^{(n+1)}\|_\infty}{(n+1)!}(b-a)^{n+1}$$

Die Aussage "für f $(n+1)$-mal stetig differenzierbar" werden wir noch sehen: $f \in C^{n+1}[I]$, f genügend glatt (*smooth*), f genügend oft stetig differenzierbar[1].

1.3 Numerische Integration = Quadratur

Ziel: Approx von

$$I[f] = \int_a^b f(x)dx$$

Idee: Verwende polynomiale Interpolation um f zu approximieren und integriere $p[f|x_0, \dots x_n]$.

Definition 1. *Eine unendliche Rechenvorschrift der Form*

$$Q[f] = \sum_{j=0}^{n}\omega_j f(x_j)$$

zur Approximieren von $I[f]$ nennt man Quadraturregel (QR) Quadraturformel. Die $x_j \in [a,b]$ nennt man (Quadratur-) Knoten oder Integrationsstützstellen und die ω_j (Quadratur-) Gewichte.

Seien $x_0, x_1, \dots, x_n \in I = [a,b]$ (Integrationsintervall) uns gegeben, dann ist das IP einer Funktion f $p[f|x_0, \dots x_n](x) = \sum_{j=0}^{n} f(x_j)L_j^n(x)$. Da das IP die Funktion approximiert, so gilt

$$\begin{aligned}\int_a^b f(x)dx &\approx \int_a^b p[f|x_0,\dots,x_n](x)dx\\ &= \int_a^b \sum_{j=0}^{n} f(x_j)L_j^n(x)dx\\ &= \sum_{j=0}^{n} f(x_j)\int_a^b L_j^n(x)dx\\ &= \sum_{j=0}^{n} f(x_j)\omega_j = Q_n[f].\end{aligned}$$

Die Quadratur-Gewichte:

$$\int_a^b L_j^n(x)dx = \omega_j.$$

Beachte: Die Gewichte ω_j sind unabhängig von f. Also einmal ausrechnen und tabellieren.

Die MR (Mittelwert), TR (Trapez) und SR (Simpson) sind Netwon-Cotes (NC) Quadraturregeln. Bei diesen Quadraturregeln verteilt man knoten x_j äquidistant über das Intervall $I = [a,b]$:

$$n = 0 : \frac{a+b}{2}$$
$$n > 0 : x_j = a + \frac{b-a}{n}j, \quad j = 0, \dots, n.$$

Bemerkungen:

- TR und SR gehören zu den populärsten Quadraturregeln.
- NC QR mit $n > 6$ werden numerisch unbrauchbar (da negative Gewichte ω_j)

[1] Letzte 2 weniger präzis.

1.4 Quadraturfehler

Nun interessiern wir uns für den QR-Fehler:

Definition 2 (Quadraturfehler)**.** *Wir nennen $E[f] = |Q[f] - I[f]|$ den Quadraturfehler (QF).*

Als ein Mass der Genauigkeit einer QR definirern wir:

Definition 3 (Genauigkeitsgrad)**.** *Eine QR hat Genauigkeitsgrad (GG) $q \in \mathbb{N}$, falls sie alle Polynome bis und mit grad q exact integriert und q die grösstmögliche Zahlt ist.*

Definition 4. *Die Ordnung einer QR ist definiert durch $s = q + 1$.*

Dank der Linearität von des Integrals $I[f]$ und $Q[f]$ kann man den GG einfah bestimmen mit [2]

$$Q[x^k] = I[x^k], \quad k = 0, 1, \dots, q$$
$$Q[x^{q+1}] \neq I[x^{q+1}]$$

Eine weiter Verainfachung bei der Bestimmung des GG, dass man es nur für das Referenzintervall (RI) $[-1, +1]$ überprüfen muss. Weil durch einfache Variabelsubstitution lässt sich ein beliebiges Intervall $[a, b]$ auf $[-1, +1]$ transformier:

$$x = \frac{b-a}{2}t + \frac{a+b}{2}, \quad t \in [-1, +1]$$

$$I[f] = \int_a^b f(x)dx = \int_{-1}^{1} f\left(\frac{b-a}{2}t + \frac{a+b}{2}\right) \underbrace{\frac{b-a}{2}dt}_{dx}$$
$$= \frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}t + \frac{a+b}{2}\right) dt$$

Es ist klar, dass Newton-Cates Quadraturregeln midestens den GG der zugrundliegenden IPs haben. Falls der Grad n des IPs aber gerade ist, so gewinnt man ein GG gratis dazu.

Für den QF lässt sich zeigen

$$E[f] \leq \frac{\|f^{(q+1)}\|_\infty}{(q+1)!}(b-a)^{q+2}$$

Zusammengefasst: Je grösser der G, desto genauer ist eine QR, vorausgesetzt, das IP ist ene gute Approximatio von f[3].

1.5 Summiere Quadraturregeln

Um eine bessere Approximation von $I[f]$ zu erhalten benutzt man im Algemein eine gegebene QR nicht über das ganze Intervall sondern über Teilintervalle.

[2] da monome eine Basis in Polynomraum bilden

[3] Polynome approximiern schlecht unstetige funktionen $\rightsquigarrow$ Intervall zerstückeln.

Da Interall $I = [a, b]$ wird in N Teil-Intervalle $I_j = [x_{j-1}, x_j]$, $(j = 1, \dots, N)$ zerteilt mit

$$x_j = a + jh, \quad j = 0, 1, \dots, N \qquad \text{und} \qquad h = \frac{b-a}{N}.$$

Nun wendet man auf jedes Teil-Intervall eine QR und summiert.

Beispiel 1 (Summierte MR (SMR))**.**

$$Q_0^N[f] = \sum_{j=0}^{N} Q_0[f \text{ auf } I_j] = \sum_{j=1}^{N} hf\left(\frac{x_{j-1}+x_j}{2}\right)$$

Beispiel 2 (Summierte TR (STR))**.**

$$\begin{aligned} Q_1^N[f] = \sum_{j=0}^{N} Q_1[f \text{ auf } I_j] &= \sum_{j=1}^{N} \frac{h}{2}\left(f(x_{j-1}) + f(x_j)\right) \\ &= \frac{h}{2}f(x_0) + h\sum_{j=1}^{N-1} f(x_j) + \frac{h}{2}f(x_N) = \frac{h}{2}\left(f(a) + 2\sum_{j=1}^{N-1} f(x_j) + f(b)\right) \end{aligned}$$

Beispiel 3 (Summierte SR (SSR))**.**

$$\begin{aligned} Q_2^N[f] = \sum_{j=0}^{N} Q_2[f \text{ auf } I_j] &= \sum_{j=1}^{N} \frac{h}{6}\left(f(x_{j-1}) + 4f\left(\frac{x_{j-1}+x_j}{2}\right) + f(x_j)\right) \\ &= \frac{h}{6}\left[f(a) + 2\sum_{j=1}^{N-1} f(x_j) + 4\sum_{j=1}^{N} f\left(\frac{x_{j-1}+x_j}{2}\right) + f(b)\right] \end{aligned}$$

Wie verhilt sih der QF von SQRn? Der QF eine SQR ist (offensichtlich) die Summe der gemachten Fehler auf jeden Teil-Intervall:

$$\begin{aligned} E^N[f] = |I[f] - Q_n^N[f]| &= \left|\sum_{j=1}^{N} I[f \text{ auf } I_j] - Q_n[f \text{ auf } I_j]\right| \le \sum_{j=1}^{N} |I[f] \text{ auf } I_j] - Q_n[f \text{ auf } I_j]| \\ &\le \sum_{j=1}^{N} \frac{\max_{x\in I_j} |f^{(q+1)}(x)|}{(q+1)!} \underbrace{|x_j - x_{j-1}|}_{h}^{q+2} \le \frac{\|f^{(q+1)}\|_\infty}{(q+1)!} h^{q+1} \underbrace{\sum_{j=1}^{N} h}_{Nh=b-a}. \end{aligned}$$

Zusammengefast:

$$E^N[f] \le \frac{\|f^{(q+1)}\|_\infty}{(q+1)!}(b-a)h^{q+1} = \frac{\|f^{(s)}\|_\infty}{s!}(b-a)h^s.$$

Lanudau Symbol: $e = \mathcal{O}(h^p)$ falls $|e| \le Ch^p$ für positive Konstanten C, p gilt alle h klein genug. Für SQR $E^N[f] = \mathcal{O}(h^{q+1}) = \mathcal{O}(h^s)$.

Bemerkgungen:

- Die Ordnung s kann sehr einfash in einem log-log-Plot ablesen;
- Um volle Ordnung ($s = q+1$) zu erhalten muss die zu integrierende Funktion glatt genug sein (genügend of stetig differenzierbar);

1.6 Gauss-Quadratur

Idee: wähle die $n+1$ Knoten so, dass der grösstmögliche GG erreicht wird. Hoffnung: GG mit $q \approx n+n+1 = 2n+1$ (n vom IP n-ten Grades, $n+1$ von der Knoten x_j)

Frage: Was ist der grösstmögliche GG der man überhaupt erreichen kann? Betrachte Folgendes Polynom vom Grad $2n+2$auf dem RI

$$p(x) = \prod_{i=0}^{n}(x - x_i) \in \mathbb{P}_{2n+2}.$$

Klar: $I[p] = \int_{-1}^{+1} p(x)dx > 0$. Aber mit Quadratur

$$Q[p] = \sum_{j=0}^{n} \omega_j \underbrace{p(x_j)}_{0} = 0.$$

Also der grösstmögliche GG den man erreichen kann ist $q = 2n+1$!

Betrachten wir hierzu den Interpolationsfehler $e(x) = f(x) - p[f|x_0, \dots, x_n](x) = K(x)\prod_{i=0}^{n}(x - x_i)$. Sei nun $f(x) = x^m$ ein Monom mit $m \geq 0$ ganzzahlig. Dann ist

$$e(x) = x^m - \underbrace{p[x^m|x_0, \dots, x_n](x)}_{\text{Polynom von Grad } n} = K(x) \quad \underbrace{\prod_{i=0}^{n}(x - x_i)}_{\text{Polynom von Grad } n+1}$$

($K(x)$ muss ein Polynom von grad $max\{m-n-1, 0\}$ sein) mit

$$K(x) = \begin{cases} 0 & \text{für } m \leq n, \\ r(x) \in \mathbb{P}_{m-n-1} & \text{für } m > n. \end{cases}$$

Integrieren wir nun $e(x)$ über das RI:

$$\begin{aligned} \int_{-1}^{1} e(x)dx &= \int_{-1}^{1} x^m dx - \int_{-1}^{1} p[x^m|x_0, \dots, x_n](x)dx = I[x^m] - Q[x^m] \\ &= \int_{-1}^{1} K(x)\prod_{i=0}^{n}(x - x_i)dx = \begin{cases} 0 & \text{für } m \leq n, \\ \int_{-1}^{1} r(x)\prod_{i=0}^{n}(x - x_i)dx & \text{für } m > n. \end{cases} \end{aligned}$$

Aber viel mehr noch: Wenn wir $n+1$ Knoten mit

$$\int_{-1}^{1} \underbrace{r(x)}_{\in \mathbb{P}_n} \underbrace{\prod_{i=0}^{n}(x - x_i)}_{\in \mathbb{P}_{n+1}} dx = 0$$

für $n < m < 2n+2$ bestimmen können, so erhalten wir den grösstmöglichen GG. Aus der linearern Algebra ist bekannt

$$\langle f, g \rangle = \int_{-1}^{1} f(x)g(x)dx$$

ein Skalaproduct in $C[-1, 1]$ definiert. Wenn $\langle f, g \rangle = 0$, so sagt man dass f und g orthogonal zueinander sind. Also

$$\left\langle r(x), \prod_{i=0}^{n}(x - x_i) \right\rangle = 0$$

sagt uns wir suchen Orthogonalpolynome! Dies führ uns zu den Legendre-Polynomen, welche durch folgende Rekusions-Formel gegeben sind

$$\begin{aligned} P_0(x) = 1, \qquad P_1(x) = x \\ P_{j+1}(x) = \frac{2j+1}{j+1}xP_j(x) - \frac{j}{j+1}P_{j-1}(x), j \leq 1. \end{aligned}$$

Die $P_j(x)$ bilden eine orthogonale Basis von $\mathbb{P}_n$:

$$\langle P_i(x), P_j(x) \rangle = \int_{-1}^{1} P_i(x)P_j(x)dx = 0, \quad i \neq j.$$

Um den maximalen GG zu erhalten, wählen wir die $n+1$ Knoten so dass

$$\prod_{i=0}^{n}(x-x_i) \sim P_{n+1}(x)$$

ein (skalares) vielfaches von $(n+1)$-Legendre Polynom ist. $\rightsquigarrow$ Wähle die Knoten x_i als die Nullstellen von $P_{n+1}(x)$!

Bemerkungen:

- DIe Gewichte der GLQ sind stets positiv.
- Für n "nicht zu gross' sind die GLQ tabelliert. Für grosse n werden die GLQ numerisch bestimmt (z.B. wie Übung S03 `gauleg.m`)
- Es gilt stets: hohe Ordnung bedeutet nicht zwingend hohe Genaugkeit! Dies gilt nur wenn f glatt genug ist.
- Allgemeiner betrachtet man

  $$I[f] = \int_a^b w(x)f(x)dx$$

  wobei $w(x)$ eine nichtnegative Gewichtsfunktion ist:
  - $w(x) = 1 \rightsquigarrow$ Gauss-Legendre
  - $w(x) = 1/\sqrt{1-x^2} \rightsquigarrow$ Gauss-Tschebyscheff
  - $w(x) = e^{-x^2} \rightsquigarrow$ Gauss-Hermite
- Manchmal beginnt der Summationsindex bei 1

  $$G[f] = \sum_{j=1}^{n} \omega_j f(x_j).$$

  Dann ist GG $q = 2n-1$ und Ordnung $2n$.

1.7 Adaptive Quadratur

Ziel: Berechne $I[f] = \int_a^b f(x)dx$ bis auf eine vogegebene Toleranz

$$|Q[f] - I[f]| < \text{ Tol.}$$

so effizient[4] wie möglich.

Idee:

Dazu benötien einen lokalen Fehler-Schätzer.

Idee: Vergleiche das Resultat einer Methode mit dem Resultat einer genaueren Methode.

Beispiel 4. *Slides (15) $\rightsquigarrow$ Zeigt dass obige einfache Idee durchaus brauchbare Fehler-Schätzer liefert.*

[4] D.h. mit so wenig Funktionauswertungen wie möglich.

Untersuche wir den sogenannte Intervall-Halbierung Fehler-Schätzer etwas genauer. Betrachten wir ei Intervall $I = [a, b]$ und den QF einer QR Ordnung s

$$E[f] = |Q[f] - I[f]| \overset{\text{QF1}}{\leq} \frac{\|f^{(s)}\|_\infty}{s!}(b-a)^{s+1} = K(b-a)^{s+1}$$

Dies nennt man einer *a-priori* Fehlerschätzer. Dieser ist natürlich (gewochnlich) nicht direkt brauchbar (weil K bzw. $\|f^{(s)}\|$ unbekannt ist). Für den QF git:

$$E^1[f] = |Q^1[f] - I[f]| \approx K(b-a)^{s+1}$$
$$E^2[f] = |Q^2[f] - I[f]| \approx \underbrace{K\left(\frac{b-a}{2}\right)^{s+1} + K\left(\frac{b-a}{2}\right)^{s+1}}_{\text{Fehler auf halben Teil-Intervallen.}} = \frac{K}{2^s}(b-a)^{s+1} = \frac{E^1[f]}{2^s}$$

Nun

$$E^1[f] = |Q^1[f] \underbrace{-Q^2[f] + Q^2[f]}_{=0 \text{ Trick}} - I[f]| \leq |Q^1[f] - Q^2[f]| + \underbrace{|Q^2[f] - I[f]|}_{E^1[f]/2^s}$$
$$\rightsquigarrow E^1[f] \approx \frac{2^s}{2^s - 1}|Q^1[f] - Q^2[f]|$$
$$E^2[f] \approx \frac{1}{2^s - 1}|Q^1[f] - Q^2[f]|$$

Dies sind sogenannten *a-posteriori* Fehler-Schätzer.

Beispiel 5. *Slides (16). Algorithmus kann mit eine zu klein Toleranz nie fertig rechnen (unendliche Rekursion). Oder (17) nach ein Schritt fertig sein.*

Bemerkung:

- Adaptive Quadratur kann *oft* gute Resultate liefern.
- Adaptive quadratur kann auch schief gehen.
- In MATLAB: `quad()`, `integral()`

1.8 Zwei-Dimensionale Quadratur

Bis jezt haben wir nur 1D Quadratur behandelt. Diese Methoden kann man relativ einfach aufh 2 oder 3 Dimensionen verellgemeinen. In 2D:

$$I[f] = \int_a^b \int_c^d f(x,y)\,dxdy \approx \sum_{i=0}^{n} \omega_i^x \int_c^d f(x_i, y)\,dy \approx \sum_{i=0}^{n} \omega_i^x \sum_{j=0}^{m} \omega_j^y f(x_i, y_j) = \sum_{i=0}^{n}\sum_{j=0}^{m} \omega_i^x \omega_j^y f(x_i, y_j)$$

Dies kann man auch etwas verallgemeinen

$$I[f] = \int_a^b \int_{\phi_1(x)}^{\phi_2(x)} f(x,y)\,dxdy$$

Ziel:

- Approximieren von Integrale.
- Genauigkeit der Approximation
- Fundamentale Konzepte der Numerik
- Newton-Cotes, Gauss, Adaptive Quadratur
- 2D Quadratur

2 Explizite Einschrittverfahren

Ziele:

- Anfangswertprobleme (gew. DGL + Anfangswert)
- Lösbarkeit
- Explizite Einschrittverfahren (Euler, Runge-Kutta)
- Genauigkeit (Disktretisierungsfehler)

Wozu: Viele Anwendungen!

2.1 Motivation

Schwingkreis

$$\ddot{I} + \frac{R}{L}\dot{I} + \frac{1}{LC}I = 0$$

(Skalare) gewöhnliche Differenzialgleichung 2. Ordnung für den Strom $I(t)$.

[1]

Molekular-Dynamik (MD)

$$m_i \ddot{\boldsymbol{x}}_i = -\nabla U(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N)$$

wobei

- N Anzahl Atome / Moleküle
- $\boldsymbol{x}_i$ Position des i-ten Atom / Molekül
- U Potential

Maxwell-Gleichungen

$$\nabla \times \boldsymbol{H} = \boldsymbol{J} + \partial_t \boldsymbol{D} \qquad \nabla \times \boldsymbol{E} = \partial_t \boldsymbol{B}$$
$$\nabla \cdot \boldsymbol{D} = \rho \qquad \nabla \cdot \boldsymbol{B} = 0$$

System von partiellen DGL.

Zur Illustration betrachten wir (die) einfache skalare DGL

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

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

Eine etwas "interessante" (skalare) DGL:

$$\dot{y}(t) = ay(t) - by^2(t) = (a - by(t))y(t)$$

Graphisch:

Einfache Approximation der Lösung

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

$$t_k = t_0 + kh, \qquad k = 0, 1, \dots, N \qquad \text{mit} \quad h = \frac{T - t_0}{N}$$

(2) Setzen Anfangswert $y_0 = y(t_0)$

(3) Berechne für $k = 0, 1, \dots, N - 1$

$$y_{k+1} = y_k + hf(t_k, y_k)$$

(4) Dies ist das *Euler Verfahren* oder expliziter Euler oder Vorwärts Euler.

In allgemein stellen sich nun folgende Fragen:

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

(ii) DGL'en höherer Ordnung?

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

(iv) Gibt es "bessere" Methoden?

2.2 Grundbegriffe

Definition 5. *Ein skalares Anfangswertproblem (AWP) erster Ordnung: Find eine Funktion $y(t)$ einer Variablen (z.B. die Zeit) mit*

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

auf dem Intervall $[t_0, T_0]$ und $y(t_0) = y_0$ (Anfangswert, AW, auch Anfangsbedingung).

Im allgemein sind $\boldsymbol{y}$, $\dot{\boldsymbol{y}}$, $\boldsymbol{f}$, $\boldsymbol{y}_0$ Vektoren:

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

und die Anfangsbedingungen sind $y_1(t_0) = y_{1,0}, \dots, y_n(t_0) = y_{n,0}$.

Beispiel 6. *Lineares DGL System*

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

Lösung: $\boldsymbol{y}(t) = e^{At}\boldsymbol{y}_0$

Definition 6. *Ein (skalares) AWP n-ter Ordnung: Finde die Funktion $y(t)$ einer Variable (z.B. Zeit) mit*

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

auf dem Intervall $I = [t_0, T]$, $y(t_0) = y_0$, $\dot{y}(t) = \dot{y}_0$, $\dots, y^{(n-1)}(t) = y_0^{(n-1)}$.

Beispiel 7 (Newton'schen Bewegungsgleichung)**.**

$$\textit{(Masse)(Beschleunigung)} = m\dot{x} = \textit{(Kraft)} = F$$

mit $x(t_0) = y_0$ (Anfangs-Position), $\dot{x}(t_0) = \dot{y}_0 = v_0$ (Anfangs-Geschwindigkeit).

Reduktion auf ein System erster Ordnung

Gegeben eine DGL n-ter Ordnung

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

Wir definieren $((n-1)$ DGL 1. Ordnung)

$$z_0(t) = y(t), \quad z_1(t) = \dot{y}(t) = \dot{z}_0, \quad \dots, \quad z_{n-1} = y^{(n-1)}(t) = \dot{z}_{n-2}$$

und (1. DGL 1. Ordnung)

$$\dot{z}_{n-1} = y^{(n)}(t) = f(t, y(t), \dot{y}(y), \dots, y^{(n-1)}(t)) = f(t, z_0(t), z_1(t), \dots, z_{n-1}(t)).$$

Diese n DGL 1. Ordnung schreiben wir als System

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

wobei $\boldsymbol{z}(t) = (z_0(t), z_1(t), \dots, z_n(t))^\mathsf{T}$, $\boldsymbol{g}(t, \boldsymbol{z}(t)) = (z_1(t), z_2(t), \dots, f(t, z_0(t), z_1(t), \dots z_{n-1}(t)))^\mathsf{T}$. Für die AWV $\boldsymbol{z}(t_0) = \boldsymbol{z}_0 = (y_0, y_1, \dots, y_{n-1})^\mathsf{T}$.

Beispiel 8.

$$\ddot{y} = f(t, y(t), \dot{y}(t)), \qquad y(t_0) = y_0, \quad \dot{y}(t_0) = \dot{y}_0.$$

$$\begin{aligned} z_0(t) &= y(t) \\ z_1(t) &= \dot{y}(t) = \dot{z}_0(t) \\ z_2(t) &= \ddot{y}(t) = \dot{z}_1(t) = f(t, y(t), \dot{y}(t)) \end{aligned}$$

somit

$$\dot{\boldsymbol{z}} = \begin{bmatrix} \dot{z}_0(t) \\ \dot{z}_1(t) \end{bmatrix} = \boldsymbol{g}(t, \boldsymbol{z}(t)) = \begin{bmatrix} z_1 \\ f(t, z_0(t), z_1(t) \end{bmatrix}$$

und die AW'e

$$\boldsymbol{z}(t_0) = \boldsymbol{z}_0 = \begin{bmatrix} y_0 \\ \dot{y}_0 \end{bmatrix}$$

Definition 7. *Eine gewöhnliche DGL heisst autonom, falls die rechtes Seite die Form $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{y}(t))$ hat (anstatt $\boldsymbol{f} = \boldsymbol{f}(t, \boldsymbol{y}(t))$).*

Autonomisieren von DGL'en

Betrachte folgende DGL

$$\dot{\boldsymbol{y}} = \boldsymbol{f}(t, \boldsymbol{y}(t)), \qquad \boldsymbol{y} \in \mathbb{R}^n.$$

Durch einführen der neuen Variablen

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

und die rechte Seite

$$\boldsymbol{g}(\boldsymbol{z}(t)) = \Big[\boldsymbol{f}(t, \boldsymbol{y}(t)), 1\Big] = \Big[\boldsymbol{f}(z_{n+1}, \boldsymbol{z}_1), 1\Big].$$

Damit

$$\dot{\boldsymbol{z}} = \boldsymbol{g}(\boldsymbol{z}(t)).$$

Beispiel 9.

$$\dot{y} = y(t)^2 + t^2, \quad \boldsymbol{z}(t) = \begin{bmatrix} z_1(t) \\ z_2(t) \end{bmatrix} = \begin{bmatrix} y(t) \\ t \end{bmatrix}, \quad \boldsymbol{g}(\boldsymbol{z}(t)) = \begin{bmatrix} z_1(t)^2 + z_2(t)^2 \\ 1 \end{bmatrix} \quad \rightsquigarrow \quad \dot{\boldsymbol{z}} = \boldsymbol{g}(\boldsymbol{z}(t))$$

Bemerkung: Allgemein bezeichnet man DGL höheren Ordnung als autonom, falls die rechte der DGL nicht explizit von der Zeit abhängt.

2.2.1 Existenz, Eindeutigkeit und Stabilität

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

(ii) Gibt es mehrere Lösungen? Under welchen Bedingungen (Eindeutigkeit)

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

Betrachten wir Folgendes (allgemeines) AWP

$$\begin{aligned} \dot{\boldsymbol{y}}(t) &= \boldsymbol{f}(t, \boldsymbol{y}(t)) && \text{(System DGL)} \\ \boldsymbol{y}(t_0) &= \boldsymbol{y}_0 && \text{(AW)} \end{aligned}$$

wobei

$$\boldsymbol{y} : I = [t_0, T] \subset \mathbb{R} \to D \subset \mathbb{R}^n, \qquad \boldsymbol{f} : I \times D \to \mathbb{R}^n.$$

D heisst Zustands- oder Phasenraum.

Um (i) und (ii) sicherzustellen, benötigt:

Definition 8. *Eine Funktion $\boldsymbol{f} : I \times D \to \mathbb{R}^n$ ist Lipschitz-stetig in (Variable) $\boldsymbol{y}$ mit Lipschitz-Kontante $C \leq 0$, wenn für alle $t \in I$ und $\boldsymbol{y}, \boldsymbol{z} \in D$ gilt*

$$\|\boldsymbol{f}(t, \boldsymbol{y}) - \boldsymbol{f}(t, \boldsymbol{z})\| \leq C\|\boldsymbol{y} - \boldsymbol{z}\|.$$

Hier ist $\|\cdot\|$ eine Norm.

Bemerkungen:

- Lipschitz-Stetigkeit breschränkt wie stark eine funktion um einen Punkt sich verändern kann:

- Stetige differenzierbare Funktionen sind Lipschitz-Stetig: Setze $C = \max_{y\in[a,b]} |f'(y)|$
- Auch nicht differenzierbare Funktionen können Lipschitz-Stetig sein, z.B. $f(y) = |y|$.
- Mancham auch Lipschitz-Bedingung genannt.

Ist $f(y) = \sqrt{y}$, $y \in \mathbb{R}^+$ Lipschitz-Stetig? Nicht bei 0.

Satz 1 (Picard-Lindelöf)**.** *Sei $\boldsymbol{f}$ stetig in $(t, \boldsymbol{y})$ und Lipschitz-Stetig in $\boldsymbol{y}$ auf $[t_0, t_0 + \delta] \times D$, mit $\delta > 0$ und D soll eine Umgebung von $\boldsymbol{y}_0$ dem AW.*

Dann existier eine eindeutige Lösung $\boldsymbol{y}(t)$ des AWP

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

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

Beispiel 10.

$$\dot{y}(t) = 2\sqrt{|y(t)|}, \qquad y(0) = 0.$$

Lösungen: $y(t) = 0$, $y(t) = t|t|$ (Zwei Lösungen). Grund: $f(t, y(t)) = 2\sqrt{|y|}$ nicht Lipschitz-Stetig bei $y = 0$

Beispiel 11. *Bsp. 10 (oben) mit $y(1) = 1 \rightsquigarrow y(t) = t|t|$ eindeutige Lösung.*

Beispiel 12.

$$\dot{y}(t) = y^2(t), \qquad y(0) = y_0 > 0.$$

Lösungen:

$$y(t) = \frac{y_0}{1 - y_0 t}$$

aber nur für $0 < t < 1/y_0$ ("zumindest für eine kurze Zeit"). Grund: Lipschitz-Kontante unbeschränkt.

Satz 2. *Die Funktion $\boldsymbol{f}$ sei Lipschitz-stetig mit die Lipschitz-Kinstante C in einer Umgebung der AW'e $\boldsymbol{y}_0, \boldsymbol{z}_0$ und seien $\boldsymbol{y}(t), \boldsymbol{z}(t)$ die Lösungen des jeweiligen AWP's.*

Dann gilt für $t \in [t_0, t_0 + \varepsilon]$:

$$\|\boldsymbol{y}(t) - \boldsymbol{z}(t)\| \leq e^{C(t-t_0)} \|\boldsymbol{y}_0 - \boldsymbol{z}_0\|.$$

Bemerkung: Satz 2 stellt sicher dass die Lösungen stetig vom AW abhängen. Aber Lösungen können exponentiell in der Zeit ausseinanderlaufen.

Beispiel 13.

$$\begin{aligned} \dot{\boldsymbol{y}} &= \lambda \boldsymbol{y}(t), \boldsymbol{y}(t_0) = \boldsymbol{y}_0 && \rightsquigarrow \boldsymbol{y}(t) = \boldsymbol{y}_0 e^{\lambda(t-t_0)} \\ \dot{\boldsymbol{z}} &= \lambda \boldsymbol{z}(t), \boldsymbol{z}(t_0) = \boldsymbol{z}_0 && \rightsquigarrow \boldsymbol{z}(t) = \boldsymbol{z}_0 e^{\lambda(t-t_0)} \end{aligned}$$

$$\boldsymbol{y}(t) - \boldsymbol{z}(t) = e^{\lambda(t-t_0)}(\boldsymbol{y}_0 - \boldsymbol{z}_0).$$

2.3 Runge-Kutta Verfahren

Betrachten

$$\dot{y}(t) = f(t, y(t)), \qquad y(t_0) = y_0.$$

Formal (Integral-Gleichung, nicht unbedingt einfacher):

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

Idee: Verwende Quadratur für das Integral der Funktion f. Konkret: t_0, y_0 und integrieren bis t_1, y_1, wobei $t_1 = t_0 + h$, mit Quadratur

$$\begin{aligned} y(t_1) &= y_0 + \int_{t_0}^{t_1} f(\tau, y(\tau))\, d\tau = y_0 + h \int_0^1 f(t_0 + h\tau, y(t_0 + h\tau))\, d\tau \\ &= y_0 + h \sum_{i=1}^{s} \omega_i f(t_0 + hc_i, y(\underbrace{t_0 + hc_i}_{\text{Knoten}})) \end{aligned}$$

Problem: $y(t_0 + hc_i)$ immer noch unbekannt $\rightsquigarrow$ approximieren. Versuchen wir es mit der Mittelpunktregel

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

Kennen wir $y(t_0, h/2)$ nicht, aber approx mit dem Euler Verfahren

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

Eingesetzt:

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

Zusammenfassend:

$$k_1 = f(t_j, y_j), \qquad k_2 = f\left(t_j + \frac{h}{2}, y_j + \frac{h}{2} k_1\right), \qquad y_{j+1} = y_j + hk_2$$

für $j = 0, 1, \ldots, N-1$. Dieser ist bekannt als *verbesserte Polygonzug-Methode von Euler* (auch modifizierte Euler Verfahren, oder explizite Mittelpunkts-Methode).

Eine andere Möglichkeit mit TR:

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

Wieder mit dem Euler Verfahren $y(t_0 + h)$ approximieren[5]:

$$y(t_1) \approx y_0 + hf(t_0, y_0) = \tilde{y}_1.$$

Graphisch:

[5] Tilde weil in prinzip $\tilde{y}_1$ nur ein Zwischenwert ist.

Wir erhalten

$$k_1 = f(t_j, y_j), \qquad k_2 = f(t_j + h, y_j + hk_1), \qquad y_{j+1} = y_j + \frac{h}{2}(k_1 + k_2)$$

für $j = 0, 1, \dots, N-1$. Dies Verfahren ist bekannt als das *Verfahren von Heun* (auch explizite Trapez-Methode).

Alle Verfahren bis jetzt sind Teil der Runge-Kutta (RK) Einschrittverfahren (ESV).

Beispiel 14. *Butcher Tableau (BT) der verbesserten Polygon-Zug Methode von Euler.*

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

Dann

$$\begin{aligned} k_1 &= f(t_j + c_1 h, y_j + h(a_{11}k_1 + a_{12}k_2) = f(t_j, y_j) \\ k_2 &= f(t_j + c_2 h, y_j + h(a_{21}k_1 + a_{22}k_2) = f(t_j + h/2, y_j + hk_1/2) \\ y_{j+1} &= y_j + h(b_1 k_1 + b_2 k_2) = y_j + hk_2 \end{aligned}$$

Beispiel 15. *BT des Euler Verfahrens:*

$$k_1 = f(t_j, y_j), \qquad y_{j+1} = y_+ j + hk_1, \qquad \begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$$

Genau gleich für Systeme: $y \to \boldsymbol{y}$, $f \to \boldsymbol{f}$. RK sind sehr allgemein. E gibt sogenannten implizite Varfahren

Beispiel 16 (Implizite Mitteplutsk-Methode)**.**

$$\begin{array}{c|c} 1/2 & 1/2 \\ \hline & 1 \end{array} \qquad k_1 = f(t_j + h/2, y_j + hk_1/2), \quad y_{j+1} = y_j + hk_1.$$

Implizit weil k_1 auf beide seiten vorkommt (kann eine Nichtlineare Beziehung haben).

Implizite Methode benötigt man bei sogenannten steifen Problemen $\rightsquigarrow$ Kapitel 5. Explizite RK-Verfahren haben ein Butcher Tableau

$$\begin{array}{c|ccccc} c_1 & 0 & 0 & \cdots & 0 & 0 \\ c_2 & a_{21} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & 0 \\ \hline & b_1 & b_2 & \cdots & b_{s-1} & b_s \end{array}$$

D.h. $\boldsymbol{A}$ ist eine untere Dreiecksmatrix mit Nullen auf der Diagonale. Das wohl bekanntetste RK ist das sogennante *klassische RK-Verfahren* (auch DAS RK, oder RK4, da es 4 Stufen hat):

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

2.4 Fehlerbetrachtungen für ESV

Fehler untersuchen vor ESV:

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

wobei ϕ die sogenannte Verfahrens- oder Inkrement-Funktion ist.

Bemerkungen:

(i) Bei RK-ESV ist $\phi(t_j, y_j, h) = \sum_{i=1}^{s} b_i k_i$.

(ii) Bei einem expliziten ESV kann man ϕ durh einsetzen einfach berechnen. Bei einem implizten ESV muss in Allgemein nicht-lineare Gleichungssysteme lösen um ϕ zu berechnen.

Im folgenden, betrachten das skalare AWP

$$\dot{y}(t) = f(t, y(t)), \quad y(t_0) = y_0$$

auf Intervall $t \in [t_0, T]$. Mit $y(t)$ bezeichnen wir stets die exakte Lösung. Wir approximieren diese Lösung durch diskretisieren des Zeitintervalls in N Teilintervalle

$$t_j = t_0 + jh, \qquad \text{mit der Schrittweite / Zeitschritt} \quad h = \frac{T - t_0}{N}.$$

Die approximierte Lösung zur t_j bezeichnen mit y_j. Also $y(t_j) \approx y_j$.

Definition 9. *Der globale Diskretisierungsfehler (GDF) zur zeit t_j ist definiert durch*

$$\epsilon_j = y(t_j) - y_j$$

D.h. ϵ_j ist der Fehler nach j Schritten.

Definition 10. *Ein ESV heisst konvergent von Ordnung p (oder hat Konvergenzordnung p), falls filt*

$$\epsilon = \max_{j=0,1,\ldots,N} \underbrace{|y(t_j) - y_j|}_{\epsilon_j} = \mathcal{O}(h^p)$$

für h klein genug.

Beispiel 17. *Empirische KO der berets gesehen Verfahren berechnet $\rightsquigarrow$ Slides*

Euler	$\sim \mathcal{O}(h^1)$
verb. Euler	$\sim \mathcal{O}(h^2)$
Heun	$\sim \mathcal{O}(h^2)$
RK4	$\sim \mathcal{O}(h^4)$

Um die KO eines ESV theoretisch zu untersuchen, benötigt man

Definition 11. *Der lokale Diskretisierungsfehler (LDF) zur Zeit t_j ist definiert durch*

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

D.h. e_j ist der Fehler nach einem Schritt aussehend von der exakten Lösung.

Graphisch: