Hochschule für Technik, Wirtschaft und Kultur Leipzig (FH)
Fachbereich Informatik, Mathematik und Naturwissenschaften

Skriptum

zur Vorlesung

Numerik II

von
Bernd Engelmann

Inhalt:

	Seite
6. Interpolation mit Polynomen und Splines	3
6.1. Die Interpolationsaufgabe	3
6.2. Eindeutige Lösbarkeit der Interpolationsaufgabe	4
6.3. Lagrangesche Interpolationsformel	4
6.4. Algorithmus von Aitken-Neville und Extrapolation	5
6.5. Newtonsche Interpolationsformel	7
6.6. Hermite-Interpolation	10
6.7. Fehler der Polynominterpolation	11
6.8. Splines	20
6.9. Interpolierende kubische Splines	21
6.10. Approximationseigenschaften kubischer Splines	24
6.11. Kubische "not-a-knot"-Splines	26
7. Numerische Integration	27
7.1. Trapezformel, Trapezsumme	27
7.2. Keplersche Fassregel, Simpson-Formel	28
7.3. Interpolatorische Quadratur	29
7.4. Trapezsummenextrapolation, Romberg-Verfahren	33
7.5. Gauss-Legendre-Integration	37
8. Numerische Differenziation	40
8.1. Finite Differenzen als Ableitungsnäherungen	40
8.2. Extrapolation für Ableitungen	42
8.3. Weitere Ableitungsapproximationen	44
9. Numerische Lösung von Anfangswertaufgaben gewöhnlicher Differenzialgleichungen	45
9.1. Einschrittverfahren (ESV)	45
9.1.1. Eulersches Polygonzugverfahren	45
9.1.2. ESV höherer Konsistenzordnung	48
9.1.3. Konstruktion von ESV höherer Konsistenzordnung.	49
9.1.4. Übertragung auf Systeme von Differenzialgleichungen	51
9.1.5. Einfluss von Rundungsfehlern	52
9.1.6. Runge-Kutta-Verfahren und Schrittweitesteuerung	52

Literatur:

Locher, F.: Numerik für Informatiker. Springer-Verlag, Berlin u.a. 1993.

Schwarz, H.R.: Numerische Mathematik. Teubner-Verlag, Stuttgart 1993.

Stoer,J.: Numerische Mathematik, Bd.1. Springer-Verlag, Berlin u.a. 1989.

Stoer,J.; Bulirsch,R.:Numerische Mathematik, Bd.2. Springer-Verlag,

Berlin u.a. 1990.

6. Interpolation mit Polynomen und Splines

Polynome und rationale Funktionen spielen in den Anwendungen eine entscheidende Rolle. Dafür sprechen folgende Gründe :

1. Polynome sind die einfachsten Funktionen, die sich allein mit Hilfe der arithmetischen Grundoperationen berechnen lassen.
2. Nach dem Satz von Taylor kann das Verhalten komplizierter Funktionen mit Hilfe von Polynomen angenähert werden, d.h. Polynome erlauben eine Approximation von Funktionen.

6.1. Die Interpolationsaufgabe

Bei der Interpolation wird eine Funktion bzw. eine Kurve gesucht, die durch gegebene Punkte (x_i, y_i) (Stützstellen) verläuft.

Grundlegende Typen der Problemstellung:

(a) Durch gegebene Punkte (x_i, y_i) eine möglichst glatte Kurve legen (CAD);

(b) Zu gegebenen Punkte (x_i, y_i) , die durch Funktionswerte $y_i = f(x_i)$ einer (komplizierten, nicht explizit darstellbaren) Funktion erzeugt werden, suche man eine (einfache, bekannte) Funktion die durch die Stützstellen verläuft (Approximation von f(x)).

Allgemein:
Es liegen (n+1) Wertepaare (Stützstellen) (x_i, y_i) i = 0, 1, ..., n vor und es sollen (n+1) Parameter $a_0, a_1, ..., a_n$ so bestimmt werden , dass für eine Interpolationsfunktion $\Phi(x; a_0, ..., a_n)$ gilt

$$\Phi(x_i; a_0, ..., a_n) = y_i \quad i = 0, 1, ..., n.$$

Lineare Interpolation: $\Phi = a_0\Phi_0(x) + ... + a_n\Phi_n(x)$

Φ hängt linear von Parametern a_i ab
(Polynominterpolation, trigonometrische Interpolation, Spline-Interpolation).

Nichtlineare Interpolation: Φ hängt nichtlinear von Parametern a_i ab, z.B.

$$\Phi(x, a_0, a_1, a_2) = a_0 e^{a_1 x} + e^{a_2 x} \quad .$$

(häufig in naturwissenschaftlichen Problemen, z. B. bei der Analyse radioaktiver Zerfallsreihen). In der numerischen Mathematik wird rationale Interpolation z. B. zur Beschleunigung der Konvergenz von Algorithmen verwendet.

Zusammenhang Ausgleichsrechnung u. Interpolation: In Problemen der Ausgleichsrechnung soll die Abweichung $\Phi(x_i, a_0, ..., a_n) - y_i$ möglichst klein sein, gemittelt über alle Messpunkte i = 0, ..., m. Es liegen i. a. aber viel mehr Messungen m als Parameter $a_0, ..., a_n$ vor. Bei der Interpolation liegen genau so viele Stützstellen (Messpunkte) vor, wie zu bestimmende Parameter.

Gesichtspunkt der Genauigkeit: Wenn Messwerte mit stärkeren Fehlern behaftet sind, so sollte eine Ausgleichsaufgabe mit eventuell weniger Parametern vorgezogen werden; es erfolgt ein statistischer Ausgleich der Fehler. Wenn die Messwerte hochgenau sind, dann liefert Interpolation i.a. die bessere Näherung für die Funktionswerte zwischen den Stützstellen.

6.2. Eindeutige Lösbarkeit der Interpolationsaufgabe

Wir bezeichnen im folgenden mit Π_n die Menge aller reellen Polynome P vom Grad ≤ n.

Interpolationsaufgabe:

geg: Stützknoten x_i i = 0, 1, ...,n ($x_j \neq x_k$ für $j \neq k$)
Stützwerte y_i

ges: Interpol. Polynom $P \in \Pi_n$ mit
$P(x_i) = y_i$ i = 0, 1, ..., n. (6.1)

Die Gleichungen (6.1) sind die sogenannten Interpolationsbedingungen, diese sind äquivalent dem linearen Gleichungssystem für $a_0, a_1, ..., a_n$

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots x_0^n \\ 1 & x_1 & x_1^2 & \cdots x_1^n \\ \vdots & & & \\ 1 & x_n & x_n^2 & \cdots x_n^n \end{pmatrix} * \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix} \quad \text{bzw. } V\,\vec{a} = \vec{y} \tag{6.2}$$

V ist die Vandermondesche Matrix

Satz 6:1. (Existenz und Eindeutigkeit der Polynominterpolation)
Es gibt genau ein Polynom P ε $\prod_n$, welches die Interpolationsaufgabe löst.

Beweis: Es gilt $\det V = \prod_{0 \le j < k \le n} (x_k - x_j)$.

Wegen $x_j \neq x_k$ für $j \neq k$ ist det V ≠ 0. Also ist (6.2) eindeutig lösbar. ◊

6.3. Lagrangesche Interpolationsformel

Def.6.2: Die Polynome

$$L_k(x) = \prod_{\substack{j=0 \\ j \neq k}}^{n} \left(\frac{x - x_j}{x_k - x_j} \right) = \frac{(x-x_0)\cdots(x-x_{k-1})(x-x_{k+1})\cdots(x-x_n)}{(x_k-x_0)(x_k-x_{k-1})(x_k-x_{k+1})\cdots(x_k-x_n)}$$
$$= \frac{w(x)}{(x-x_k)w'(x_k)} \tag{6.3}$$

heißen Lagrangesche Basispolynome zu $x_0, x_1, \cdots, x_n$. und $w(x) = \prod_{j=0}^{n} (x - x_j)$

das Knotenpolynom.

Die Basispolynome haben die Eigenschaft, dass gilt

$$L_k(x_j) = \delta_{kj} = \begin{cases} 0 & j \neq k \\ 1 & j \neq k \end{cases}$$

Satz 6.3: (Lagrangesche Interpolationsformel). Die eindeutige Lösung der Interpolationsaufgabe ist gegeben durch

$$P(x) = \sum_{k=0}^{n} y_k L_k(x) \qquad (6.4)$$

Beweis: $L_k \; \varepsilon \; \prod_n \; und$

$$P(x_j) = \sum_{k=0}^{n} y_k L_k(x_j) = y_j \cdot 1; \quad j=0,1,\cdots,n. \; \Diamond$$

Bemerkungen:

1. Satz 6.3 belegt konstruktiv die Existenz einer Lösung der Interpololationsaufgabe.
2. Die Lagrangesche Interpolationsformel ist meist nur von theoretischem Interesse. Sie ist ungünstig, wenn die Zahl der Stützstellen erweitert wird, da sich alle Lagrangeschen Basispolynome ändern.

6.4. Algorithmus von AITKEN - NEVILLE und Extrapolation

Bezeichnung: Mit

$P_j^{(K)} \varepsilon \pi_K$

werde das interpolierende Polynom der Aufgabe (6.1) zu den Stützstellen $x_{j-K},\cdots,x_j \; (0 \le K \le j \le n)$ bezeichnet.

Also: $P_{j-1}^{(K-1)}$ interpoliert an den Stellen $x_{j-k},\cdots,x_{j-2},x_{j-1}$

$P_j^{(K-1)}$ interpoliert an den Stellen $x_{j-k+1},\cdots,x_{j-1},x_j$

$P_j^{(K)}$ interpoliert an den Stellen $x_{j-k},\cdots,x_{j-2},x_{j-1},x_j$

Satz 6.4: Es gilt

$$P_j^{(K)}(x) = \frac{x_j - x}{x_j - x_{j-K}} P_{j-1}^{(K-1)}(x) + \frac{x - x_{j-K}}{x_j - x_{j-K}} P_j^{(K-1)}(x) \qquad (6.5)$$

Wiederholte Anwendung der Beziehung (6.5) liefert das Interpolationstableau:

k=0	k=1	k=2	.....	k=n	
$y_0 \equiv P_{0,}^{(0)}(x)$					
$y_1 \equiv P_{1,}^{(0)}(x)$	$P_1^{(1)}(x)$				
$y_2 \equiv P_{2,}^{(0)}(x)$	$P_2^{(1)}(x)$	$P_2^{(2)}(x)$			
$\vdots$					
$y_n \equiv P_{n,}^{(0)}(x)$	$P_n^{(1)}(x)$	$P_n^{(2)}(x)$		$P_n^{(n)}(x)$	interpoliert über alle Stützstellen

Mit Hilfe des Tableaus kann $P_n^{(n)}(x)$ an einer (oder mehreren) Stellen $x = \bar{x}$ berechnet werden. Statt das Tableau spaltenweise von links nach rechts aufzubauen, kann man es auch Zeile für Zeile aufbauen:

Algorithmus von Aitken-Neville

geg: Stützstellen $x_0, x_1, \ldots, x_n$ und Stützwerte $y_0, y_1, \ldots, y_n$; x Neustelle

ges.: $P_n^{(n)}(x)$

j := 0, 1, ... , n:

$$P_j^{(0)}(x) := y_j$$

K = 1, ... , j

$$P_j^{(K)}(x) := P_j^{(K-1)}(x) + \frac{x - x_j}{x_j - x_{j-K}} \left(P_j^{(K-1)}(x) - P_{j-1}^{(K-1)}(x) \right) \qquad (6.6)$$

Extrapolation

Liegt die Neustelle $\bar{x}$ nicht innerhalb der Stützstellen $x_0, x_1, \ldots, x_n$, so spricht man bei der Berechnung von $P_n(\bar{x})$ von Extrapolation. Das Schema von Aitken-Neville wird in der Regel zur Extrapolation verwendet (z.B. Trendberechnungen, wenn x die Zeit bezeichnet und $\bar{x}$ in der Zukunft liegt; Grenzwertberechnungen, wenn $\bar{x}$ eine Grenzstelle bezüglich x ist). Mittels der Transformation $x' = x - \bar{x}$ kann die Neustelle $\bar{x}'=0$ gesetzt werden. Wir nehmen an, daß diese Transformation bereits durchgeführt ist, dann folgt mit $x = \bar{x} = 0$ für die extrapolierten Werte $P_j^k = P_j^k(0)$ aus (6.6)

$$P_j^{(K)} := P_j^{(K-1)} - \frac{x_j}{x_j - x_{j-K}} \left(P_j^{(K-1)} - P_{j-1}^{(K-1)} \right) \qquad (6.7)$$

In den Anwendungen (z.B. bei Grenzwertberechnungen) wird häufig statt x die Bezeichnung h (Schrittweite) verwendet und es ist für eine Größe T(h) der Grenzwert für $x = h \to \bar{h} = 0$ zu berechnen. Die Startwerte P_j^0 im Neville Schema sind dann mit den Werten $T(h_j)$ zu besetzen und die Werte x_j mit h_j. Häufig entstehen die Schrittweiten h_j durch fortlaufende Halbierung einer Anfangsschrittweite h_0, d.h. es gilt

$$h_j = h_{j-1} / 2 = h_0 / 2^j .$$

Dann folgt für die Werte P_j^k, welche den gesuchten Grenzwert T(0) auf der Basis der Stützwerte $P_j^0 = T(h_j)$ extrapolieren, die Berechnungsformel

$$P_j^{(K)} := P_j^{(K-1)} + \frac{1}{2^k - 1} \left(P_j^{(K-1)}(x) - P_{j-1}^{(K-1)}(x) \right) \quad k = 1,2,..., j \qquad (6.8)$$

Die Diagonalwerte P_k^k des Aitken-Neville Schemas konvergieren tatsächlich gegen den gesuchten Grenzwert T(0), wenn T(h) in einer Umgebung von h=0 eine Entwicklung der Form

$$T(h) = \alpha_0 + \alpha_1 h + \alpha_2 h^2 + \ldots + \alpha_m h^m + R_m(h) h^{m+1} \qquad (6.9)$$

mit beschränktem $R_m(h)$ besitzt.

Bemerkung: Setzt man $h_j = 1 / n_j$ j=0,1,... , so können mittels des Neville-Schemas auch Grenzwerte von Zahlenfolgen $\{a_n\}$ für $n \to \infty$ bzw. Partialsummenfolgen unendlicher Reihen extrapoliert werden. Als Startwerte P_j^0 des Tableaus sind dabei die Werte a_{n_j} der Zahlenfolge bzw. s_{n_j} der Partialsummenfolge zu verwenden. Bei fortlaufender Verdoppelung der n_j erhält man wieder die Berechnungsvorschrift (6.8) für die Werte des Neville-Schemas.

6.5. Newtonsche Interpolationsformel

Falls man das interpolierende Polynom selbst bzw. Werte an mehreren Stellen benötigt, so ist die Lagrangesche Form bzw. der Algorithmus von Aitken-Neville nicht vorteilhaft. Insbesondere beim Einfügen einer neuen Stützstelle ändern sich alle Lagrangeschen Basispolynome.

Gegeben: Stützknoten x_i, Stützwerte y_i $(i = 0,1,...,n)$

Newtonsche Form der interpolierende Polynome:

$P_1(x) = a_0 + a_1(x - x_1)$ interpol. über $(x_0, y_0), (x_1, y_1)$

$P_2(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1)$ interpol. über $(x_0, y_0), (x_1, y_1), (x_2 - y_2)$

:

$P_n(x) = a_0 + a_1(x - x_0) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$ interpol. über alle Stützstellen .

Offenbar besteht zwischen P_{K-1}, P_K die Beziehung

$$P_K(x) = P_{K-1}(x) + a_K(x - x_0)(x - x_1)\cdots(x - x_{K-1}) \qquad (6.10)$$

d. h. die Polynome sind rekursiv aufgebaut; die Hinzunahme einer Stützstelle (x_K,y_K) führt dazu, dass ein neuer Summand mit dem Koeffizienten a_K hinzugefügt wird.

Berechnung der Koeffizienten

Die Koeffizienten genügen einem Bildungsgesetz, welches mit sogenannten dividierten Differenzen zusammenhängt:

Def. 6.5.: Die rekursiv definierten Größen

$$f[x_j] := y_j,$$

$$f[x_j, x_{j,1}, \cdots, x_K] = \frac{f[x_{j+1}, \cdots, x_K] - f[x_j, \cdots, x_{K-1}]}{x_K - x_j} \qquad \text{für } 0 \le j < K < n \qquad (6.11)$$

heißen dividierte Differenzen; $f[x_{j+1}, \cdots, x_K]$ heißt (K-j)te-dividierte Differenz.

Rekursionsschema zur Berechnung dividierter Differenzen

$$\begin{array}{llll}
x_0 & y_0 = f[x_0] =: a_0 & & \\
x_1 & y_1 = f[x_1] & f[x_0, x_1] =: a_1 & \\
x_2 & y_2 = f[x_2] & f[x_1, x_2] & f[x_0, x_1, x_2] =: a_2 \\
\vdots & & & \\
\vdots & & & \\
x_{n-2} & y_{n-2} = f[x_{n-2}] & & \\
x_{n-1} & y_{n-1} = f[x_{n-1}] & f[x_{n-2}, x_{n-1}] & \\
x_n & y_n = f[x_n] & f[x_{n-1}, x_n] & f[x_0, \cdots, x_n] := a_n
\end{array}$$

Praktische Berechnung: Von oben Zeile für Zeile möglich. Hinzufügen einer weiteren Stützstelle ist unproblematisch; das Schema wird um (x_{n+1}, y_{n+1}) erweitert und eine weitere Zeile berechnet.

Zusammenhang mit den Koeffizienten des Interpolationspolynoms

Satz 6.6.: Die Lösung der Interpolationsaufgabe (6.1) ist gegeben durch das Interpolationspolynom in Newtonscher Form

$$P_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1})$$

mit den Koeffizienten

$$a_0 = f[x_0], \quad a_1 = f[x_0, x_1], \quad a_2 = f[x_0, x_1, x_2], \ \cdots, \ a_n = f[x_0, \cdots, x_n],$$

d. h. diese bilden die obere Schrägzeile des Schemas der dividierten Differenzen.

HORNER-artiges Auswertungsschema:

Nach der rekursiven Berechnung der dividierten Differenzen kann das Polynom wie folgt ausgewertet werden

$$P_n(x) = \{[a_n(x - x_{n-1}) + a_{n-1}](x - x_{n-2}) + \cdots + a_1\}(x - x_0) + a_0$$

bzw. mit Hilfe des Schemas

	$x - x_{n-1}$	$x - x_{n-2}$		$x - x_1$	$x - x_0$
	$a_n (x - x_{n-1})$	$b_{n-1}(x - x_{n-2})$			$b_1 (x - x_0)$
a_n	$+a_{n-1}$	$+ a_{n-2}$	.....	$+a_1$	$+a_0$
	b_{n-1}	b_{n-2}		b_1	$b_0 = P_n(x)$

Eigenschaften der dividierten Differenzen:

1.) Die dividierten Differenzen sind symmetrische Funktionen der x_i , d. h. sind $x_{i_0}, \ldots, x_{i_K}$ irgendwelche Permutation von $x_0, \ldots, x_K$, so gilt

$$f[x_0, \ldots, x_K] = f[x_{i_0}, \ldots, x_{i_K}] \ . \qquad (6.12)$$

Das folgt aus der eindeutigen Lösbarkeit der Interpolationsaufgabe, d. h. alle Koeffizienten der entsprechenden Polynome müssen übereinstimmen. Die Eigenschaft belegt, dass Stützstellen beliebig angeordnet sein können.

2.) Ist f(x) ein Polynom von Grad N und sind $y_i = f(x_i)$
i = 0, 1, 2, ... Stützstellen, so gilt $f[x_0, \cdots, x_K] = 0$ für K > N.

Algorithmus: N- Interpolation; Berechnung des Polynoms an einer Stelle x
Speicherplatzsparende Variante

Eingabe: n ... Grad von P_n
$x_0, ..., x_n; y_0, ..., y_n$ Stützstellen
x ... Stelle, an der P_n auszuwerten ist (Neustelle)

```
i = 0, 1, ..., n
    a [i]  := y [i]                    Vorbesetzung der Koeffizienten

j = 1, 2, ..., n
      K = n, n-1, ..., j
                                       Berechnung der Koeffizienten
      a [K]  := ( a [K] - a [K-1 ]) / ( x [K] - x [K-1])

P  := a [n]
l = n-1, ..., 1, 0

      P := P * ( x - x [l] ) + a [l]   Berechnung des Funktionswertes
```

Spezialfall äquidistanter Stützstellen:

Sind die Stützknoten äquidistant verteilt mit Schrittweite h, d. h. $x_i = x_0 + i * h \quad i = 0, 1, ..., n$ so kann man die dividierten Differenzen durch vorwärts genommene Differenzen ersetzen:

$$\Delta^0 f(x_i) = Y_i$$

$$\Delta^1 f(x_i) = Y_{i+1} - Y_i = \Delta^0 f(x_i) - \Delta^0 f(x_{i+1}) - \Delta^0 f(x_i)$$

$$\Delta^K f(x_i) = \Delta^{K-1} f(x_{i+1}) - \Delta^{K-1} f(x_i)$$

Diese Differenzen unterscheiden sich von den dividierten Differenzen dadurch, dass nicht durch die Differenz der Stützknoten dividiert wird. Es gilt also der Zusammenhang

$$f[x_i] = \Delta^0 f(x_i)$$

$$f[x_i, x_{i+1}] = \frac{\Delta^1 f(x_i)}{h}$$

$$\vdots$$

$$f[x_i, \cdots, x_{i+k}] = \frac{\Delta^k f(x_i)}{k!\, h^k}.$$

Inverse Interpolation

Ein Typ von Interpolationsproblemen kann wie folgt formuliert werden:

Gegeben: Eine Tabelle von Stützknoten und Stützwerten $(x_i, y_i) \quad i = 0,1,\ldots,n$, wobei die Werte y_i durch eine Funktion $y = f(x)$ erzeugt sind.
Gesucht: Eine Stelle $x = \bar{x}$, für welche die Funktion f(x) einen bestimmten Funktionswert $y = \bar{y}$ annimmt (z.B. $\bar{y} = 0$, dann entspricht $\bar{x}$ einer Nullstelle von f(x)).

Interpoliert man die Funktion $y = f(x)$ mit Hilfe eines Polynoms $P_n(x)$, so ist zur Bestimmung einer Näherung von $\bar{x}$ die Gleichung $P_n(x) = \bar{y}$ zu lösen. Die Lösung der i.a. nichtlinearen Polynomgleichung kann umgangen werden, wenn statt der Funktion $y = f(x)$ die Umkehrfunktion $x = f^{-1}(y)$ interpoliert werden kann (es wird vorausgesetzt, dass diese

im Bereich der Interpolationsstellen existiert). Dazu sind lediglich die Werte der Ausgangstabelle zu vertauschen und in gewöhnlicher Weise die Interpolation durchzuführen. Bezeichnet $x = Q_n(y)$ das entsprechende Interpolationspolynom, so erhält man wegen $\bar{x} = f^{-1}(\bar{y})$ durch Berechnung von $Q_n(\bar{y})$ eine Näherung für die gesuchte Stelle $\bar{x}$. Die Berechnung kann z.B. wieder mit Hilfe des Schemas von Aitken-Neville erfolgen.

6.6. Hermite - Interpolation

Kennzeichen: An einem oder mehreren Stützknoten x_i werden außer dem Funktionswert y_i auch Ableitungswerte y_i' , ..., $y_i^{(K)}$ vorgegeben.

Man führt dann die modifizierte Folge von Stützknoten ein:
Sind in Knoten x_i der Funktionswert y_i und Ableitungen y_i' , ..., $y_i^{(K)}$ vorgegeben, so tritt der Knoten in der modifizierten Knotenfolge K + 1 mal auf. Ist

$$a = x_0 \le x_1 \le x_2 \le \cdots \le x_n = b$$

diese Folge, so lautet die Aufgabe der Hermite-Interpolation:

Aufg: Gesucht ist ein Polynom P ε π_n so,dass $P^{(d_i)}(x_i) = f^{(d_i)}(x_i)$

$$d_i = \max\{ j \quad x_i = x_{i-j} \}$$

Existenz und Eindeutigkeit des Polynoms P(x) können wieder nachgewiesen werden; wir haben n+1 unabhängige Bedingungen für n+1 gesuchte Koeffizienten.

Praktische Berechnung: P(x) wird in der Newtonschen Darstellung bestimmt, wobei die Koeffizienten mit Hilfe dividierter Differenzen berechnet werden. Gegebene Größen werden in das Schema der dividierten Differenzen eingetragen (jetzt ist nicht nur die 1. Spalte besetzt, sondern es gibt Eintragungen von $f^{(K)}(x_i)$ in der k-ten Spalte, wobei die Bedingung (6.13) zu beachten ist).

$$P(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + a_n(x - x_0)\cdots(x - x_{n-1}) \ .$$

Bem.: Im Abschnitt 6.7. werden wir den Zusammenhang zwischen dividierten Differenzen und Ableitungen der Funktion f herstellen:

$$f[x_0, \cdots, x_k] = \frac{f^{(k)}(\xi)}{k!} \qquad \xi \text{ eine Zwischenstelle aus } (x_0, x_K)$$

Folgerung: Sind alle Stützstellen gleich, d. h. $x_0 = x_1 = \cdots = x_n$ und es sind die Stützwerte $f(x_0), f'(x_0), f''(x_0), ..., f^{(n)}(x_0)$ vorgegeben, so gilt $a = b = x_0$ und $\xi = x_0$ d. h. es ist

$$f[x_0, \cdots, x_0] = \frac{f^{(k)}(x_0)}{k!} = a_k \qquad k = 0,1,\cdots,n \tag{6.13}$$

die k-te dividierte Differenz und

$$P_n(x) = a_0 + a_1(x - x_0) + ... + a_n(x - x_0)^n = \sum_{K=0}^{n} \frac{f^{(K)}(x_0)}{K!}(x - x_0)^K$$

ist das Taylorpolynom ohne Restglied.

Beachte: Vorbesetzung des Interpolynomsschemas: Ist $f^{(K)}(x_i)$ vorgegeben, so ist $\frac{f^{(k)}(x_i)}{k!}$ in das Schema der dividierten Differenzen einzutragen.

Spezialfall der kubischen Hermiteinterpolation: Gegeben sind 2 Knoten x_0, x_1 und die Stützwerte y_0, y_0', y_1, y_1' .Dann ist das kubische Interpolationspolynom eindeutig bestimmt.

Darstellung von P(x) durch kubische die kubische Hermitebasis H_0, H_1, H_2, H_3

$$P(x)=y_0 H_0(x)+y_0' H_1(x)+y_1 H_2(x)+y_1' H_3(x).$$

Die kubischen Hermite-Basispolynome erfüllen dabei die Bedingungen

$$H_0(x_0)=1,\ H_0'(x_0)=0,\ H_0(x_1)=0,\ H_0'(x_1)=0$$
$$H_1(x_0)=0,\ H_1'(x_0)=1,\ H_1(x_1)=0,\ H_1'(x_1)=0$$
$$H_2(x_0)=0,\ H_2'(x_0)=0,\ H_2(x_1)=1,\ H_2'(x_1)=0$$
$$H_3(x_0)=0,\ H_3'(x_0)=0,\ H_3(x_1)=0,\ H_0'(x_1)=1$$

Übertragung auf Knoten $x_0, x_1, x_2, \ldots, x_n$:
Für ein Teilintervall $\lfloor x_i, x_{i+1} \rfloor$ $(i=0,1,\cdots,n-1)$ wird jeweils $P_i(x)$ als kubisches Polynom bestimmt. Die zusammengesetzte Funktion ist dann stetig differenzierbar über dem Gesamtintervall $[a,b] = [x_0, x_n]$, da die Ableitungen der aneinanderstoßenden kubischen Polynomstücke gleich sind (Lokal kubische Hermite-Interpolation).

6.7. Fehler der Polynominterpolation, Approximation von Funktionen

6.7.1 Fehlerformel der Interpolation

Ausgangspunkt: Die gegebenen Wertepaare $(x_i, y_i)\ i = 0,1,..,n$ sind Abszisse und Funktionswert einer häufig komplizierten oder nur implizit gegebenen Funktion $y = f(x)$. Ersetzt man diese Funktion im Intervall $[a,b]$ durch das Inter-polationspolynom $P_n(x)$, so entsteht natürlich ein Fehler $e_n(x) = f(x) - P_n(x)$ für Werte $x \neq x_i$. Ohne Voraussetzungen an f(x) kann dieser Fehler beliebig groß werden. Ist jedoch die Funktion f(x) genügend oft differenzierbar, so kann der Interpolationsfehler berechnet werden.

Interpolationsaufgabe: Gegeben sind

$$x_i \in [a,b] \quad i = 0,1,\ldots,n$$
$$y_i = f(x_i) \quad i = 0,1,\ldots,n$$

sowie das Interpolatonspolynom $P_n(x) \in \Pi_n$ mit $P_n(x_i) = y_i$.

Voraussetzung: $f(x)$ soll in $[a,b]$ (n+1)-mal differenzierbar sein.
ges : $e_n(x) = f(x) - P_n(x)$ für $x \in [a,b],\ x \neq x_i$.

Satz 6.7.: Unter den obigen Voraussetzungen existiert für jedes feste $x \in (a,b)$ ein Wert $\xi = \xi(x) \in (a,b)$ mit

$$f(x) - P_n(x) = e_n(x) = \omega(x)\frac{f^{(n+1)}(\xi)}{(n+1)!} \tag{6.14}$$

und

$$\omega(x) = (x-x_0)(x-x_1)\ldots(x-x_n) \text{ (Knotenpolynom).}$$

Anwendung: Ist die (n+1)-te Ableitung von $f(x)$ auf $[a,b]$ beschränkt, d.h

$$\left|f^{(n+1)}(x)\right| \le M_{n+1} \quad x \in [a,b]$$

so gilt

$$|e_n(x)| \le \frac{M_{n+1}}{(n+1)!}\max_{x\in[a,b]}|\omega(x)|\,. \tag{6.15}$$

Im Fall äquidistanter Stützstellen $x_{i+1} - x_i = h$ gilt für $x \in [x_0, x_n] = [a,b]$

$$|\omega(x)| = |(x-x_0)(x-x_1)\ldots(x-x_n)| \le h\cdot h\cdot(2h)\ldots(nh) < (n+1)!h^{n+1}$$

d.h.

$$|e_n(x)| \le M_{n+1}h^{n+1} = O(h^{n+1})\,.$$

Damit gilt: Der Maximalfehler der Interpolation nimmt mit Verkleinerung von h (bzw. Vergrößerung von n) ab, falls die Funktion f(x) Ableitungen besitzt, die gleichmäßig beschränkt sind durch eine Konstante M, d.h $M_i \le M \;\forall i$. Je zentraler x liegt, um so kleiner wird das Maximum von $\omega(x)$ sein, d.h der kleinste Interpolationsfehler entsteht für zentrale Lage von x bezüglich der Stützstellen $x_0,\ldots,x_n$.

Andere Darstellung des Interpolationsfehlers:

Es sei $x = \bar{x}$ fest in $[a,b]$ gewählt und wir führen $x_{n+1} = \bar{x},\; y_{n+1} = f(\bar{x})$ als neue Stützstelle ein.

$P_n(x)$ soll über $x_0,\ldots,x_n$ und

$P_{n+1}(x)$ soll über $x_0,\ldots,x_n,x_{n+1}$ interpolieren.

Dann gilt:

$$P_n(x) = a_0 + a_1(x-x_0)+\ldots+a_n(x-x_0)\ldots(x-x_{n-1})$$
$$P_{n+1}(x) = a_0 + a_1(x-x_0)+\ldots+a_n(x-x_0)\ldots(x-x_{n-1}) + a_{n+1}(x-x_0)\ldots(x-x_n)$$

sowie

$$P_{n+1}(\bar{x}) - P_n(\bar{x}) = a_{n+1}(\bar{x}-x_0)\ldots(\bar{x}-x_n) = a_{n+1}\omega(\bar{x}).$$

Wegen

$$P_{n+1}(\bar{x}) = y_{n+1} = f(\bar{x})\;;\; a_{n+1} = f[x_0,\ldots,x_n,\bar{x}]$$

erhält man

$$f(\bar{x}) - P_n(\bar{x}) = e_n(\bar{x}) = f[x_0,\ldots,x_n,\bar{x}]\omega(\bar{x}).$$

Da $\bar{x}$ beliebig war, folgt durch Vergleich mit (6.14)

$$e_n(\bar{x}) = f[x_0,\ldots,x_n,\bar{x}]\omega(\bar{x}) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega(\bar{x})$$

d.h. für ein $\xi = \xi\left(\bar{x}\right) \in I$ gilt

$$\boxed{f\left[x_0,\ldots,x_n,\bar{x}\right] = \frac{f^{(n+1)}(\xi)}{(n+1)!}} \qquad (6.16)$$

wobei I das kleinste Intervall ist, welches $x_0,\ldots,x_n$ und $\bar{x}$ enthält.

Damit: Für dividierte Differenzen und Ableitungen gilt der Zusammenhang (6.16) mit entsprechenden Ableitungen von $f(x)$ an einer Zwischenstelle .

6.7.2. Einfluß von Datenfehlern und Rungesches Phänomen

Wir nehmen an, dass die Stützwerte y_k fehlerbehaftet sind (resultierend aus Messungen u.a.), d.h. $\tilde{y}_k$ sind Näherungen mit

$$y_k - \tilde{y}_k = \varepsilon_k \quad , \ |\varepsilon_k| \le \varepsilon \qquad , \ k = 0,1,\ldots,n .$$

$P_n(x)$ sei das Interpolationspolynom zu exakten Daten y_k und $\tilde{P}_n(x)$ das zu gestörten Daten $\tilde{y}_k$. Dann gilt nach der Lagrange-Formel

$$P_n(x) = \sum_{k=0}^{n} y_k L_k(x) \ ; \ \ \tilde{P}_n(x) = \sum_{k=0}^{n} \tilde{y}_k L_k(x)$$

und für das Fehlerpolynom $E(x)$ folgt

$$E_n(x) = P_n(x) - \tilde{P}_n(x) = \sum_{k=0}^{n} (y_k - \tilde{y}_k) L_k(x) = \sum_{k=0}^{n} \varepsilon_k L_k(x) \qquad (6.17)$$

bzw.

$$|E_n(x)| \le \sum_{k=0}^{n} |\varepsilon_k| |L_k(x)| \le \varepsilon \sum_{k=0}^{n} |L_k(x)| = \varepsilon . \mathrm{L}_n(x).$$

$\mathrm{L}_n(x)$ wird als Lebesque-Funktion bezeichnet und $\Lambda_n = \max\limits_{x\in[a,b]} \mathrm{L}_n(x)$ als Lebesque-Konstante.

Satz 6.8.: Durch die Datenfehler wird ein Polynomfehler $E_n(x)$ hervorgerufen mit

$$|E_n(x)| \le \varepsilon \, \mathrm{L}_n(x) \quad \text{bzw.} \ \max_{x\in[a,b]} |E_n(x)| \le \varepsilon \cdot \Lambda_n \qquad (6.18)$$

Beachte: Λ_n ist eine nur von der Lage der Knoten abhängige Fehlerkonstante. Für gleichabständige Stützstellen wächst Λ_n sehr schnell mit n an.

Damit können kleine Fehler in y_k u.U. große Fehler im Interpolationspolynom hervorrufen. Die Konditionen der Interpolationsaufgabe verschlechtert sich somit mit wachsendem n.

Wir betrachten eine Funktion $f(x)$ auf dem Intervall $[-1,1]$ und interpolieren diese auf der Basis gleichabständiger Stützstellen. Für Funktionen, deren sämtliche Ableitungen durch eine Konstante M gleichmäßig beschränkt sind, geht der Fehler $e_n(x) = f(x) - P_n(x)$ für $n \to \infty$ gegen Null.

Das folgende Beispiel zeigt aber, dass es Funktionen gibt, die beliebig oft differenzierbar sind und diese Bedingung nicht erfüllen:

Beisp.:

$$y = f(x) = \frac{1}{1+12x^2} \quad , \quad x \in [-1,1]$$

Die Funktion soll mit Hilfe äquidistanter Stützknoten und der entsprechenden Interpolationspolynome angenähert werden. Das Bild zeigt die Verhältnisse im Fall n=10.

Die schlechten Approximationseigenschaften von $P_n(x)$ im obigen Beispiel werden als Rungesches Phänomen bezeichnet und resultieren vor allem aus der Verwendung äquidistanter Stützstellen.

6.7.3. Tschebyscheff-Polynome und Lagrange-Tschebyscheff-Approximation

Bisher hatten ein Polynom n-ten Grades in Monom-Darstellung, Lagrange- oder Newtonscher Darstellung betrachtet. Im Vektorraum der Polynome vom Grad ≤ n kann man auch eine andere Basis zur Polynomdarstellung wählen.
Eine wichtige Basis bilden die Tschebyscheff - Polynome erster und zweiter Art, die ebenfalls linear unabhängig sind und jedes Polynom n-ten Grades erzeugen können. Der Grund für die Wahl der Tschebyscheff-Polynome liegt darin, daß sie durch eine 3-Term-Rekursion bestimmt sind und gewisse Orthogonalitätsbedingungen erfüllen.

Def. 6.9.: Die durch

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_{k+1}(x) &= 2xT_k(x) - T_{k-1}(x) \qquad , \; k = 1,2,\ldots,n-1 \end{aligned} \tag{6.19}$$

definierten Polynome $T_0, T_1, \ldots, T_n$ heißen Tschebyscheff-Polynome 1.Art .

Die durch

$$\begin{aligned} U_0(x) &= 1 \\ U_1(x) &= 2x \\ U_{k+1}(x) &= 2xU_k(x) - U_{k-1}(x) \quad , \; k = 1,2,\ldots,n-1 \end{aligned}$$

definierten Polynome $U_0, U_1, \ldots, U_n$ heißen Tschebyscheff-Polynome 2.Art .

Eigenschaften von Tschebyscheff-Polynomen erster Art

1. Darstellung

Ist $x \in [-1,1]$, so kann man den Zusammenhang mit trigonometrischen Funktionen nutzen. Additionentheoreme liefern:

$$\cos(k+1)t = \cos kt \cos t - \sin kt \sin t$$
$$\underline{\cos(k-1)t = \cos kt \cos t + \sin kt \sin t}$$
$$\cos(k+1)t + \cos(k-1)t = 2\cos kt \cos t$$

Ersetzt man $\varphi_k(t) = \cos kt$ so gilt $\varphi_0(t) = 1, \varphi_1(t) = \cos t$ und es folgt

$$\varphi_{k+1}(t) = 2\cos t \varphi_k(t) - \varphi_{k-1}(t)$$

d.h. die Rekursionsbeziehung für φ_{k+1} ähnelt der von $T_{k+1}(t)$.
Die Transformationen

$$x = \cos t \qquad t \in [0, \pi] \quad \text{bzw.} \quad t = \arccos x \qquad x \in [-1,1]$$

sind umkehrbar eindeutig, so daß man für $x \in [-1,1]$ wegen der identischen Rekursionsbeziehung und der gleichen Startfunktionen erhält:

$$T_k(x) = \cos(k \cdot \arccos x) \qquad x \in [-1,1] \tag{6.20}$$

2.Verlauf der Tschebyscheff-Polynome $T_k(x)$ im Intervall $[-1,1]$

(a) $|T_k(x)| \le 1$ für $x \in [-1,1]$ folgt unmittelbar aus (6.20)

(b) Nullstellen von $T_k(x)$:

$$\boxed{x = \cos\left(\frac{\pi}{k}\frac{2l-1}{2}\right) \quad , l = 1,2,\ldots,k-1,k} \tag{6.21}$$

einfache Nullstellen (Tschebyscheff-Knoten)

(c) Extremalpunkte: $T_k(x) = \pm 1$ d.h.

$$x = \cos\left(\frac{l\pi}{k}\right) \quad , l = 0,1,\ldots,k$$

(d) Im Intervall $(0,\pi)$ ist $\cos t$ monoton fallend, so daß für die $\cos$ - Werte die Ungleichung gilt:

$$\cos\left(\frac{2l+1}{k+1}\frac{\pi}{2}\right) < \cos\left(\frac{2l-1}{k}\frac{\pi}{2}\right) < \left(\frac{2l-1}{k+1}\frac{\pi}{2}\right)$$

Damit: Zwischen zwei aufeinanderfolgenden Nullstellen von T_{k+1} liegt genau eine Nullstelle von T_k .

(e) Symmetrieeigenschaft
Für $T_0(x)=1$ gilt: $T_0(x)$ ist eine gerade Funktion. Für $T_1(x)=x$ gilt: $T_1(x)$ ist eine ungerade Funktion. Durch Induktion folgt mit der Rekursionsformel:
$T_k(x)$ ist gerade für geraden Index k und ungerade für ungerades k.

Bem.: Die Tschebyscheff-Knoten (6.21), d.h. die Nullstellen des T-Polynoms $T_k(x)$ für k=n+1 spielen eine besondere Rolle bei der Interpolation.

Der Clenshaw-Algorithmus

Ein Polynom $p_n(x)$ soll jetzt in Tschebyscheff-Darstellung gegeben sein, wobei als Basis-Polynome die Tschebyscheff-Polynome $T_0(x),T_1(x),\ldots,T_n(x)$ auftreten:

$$p_n(x)=a_0T_0(x)+a_1T_1(x)+\ldots+a_nT_n(x).$$

Zur Berechnung des Funktionswertes ist ein Algorithmus analog dem Horner-Schema bei Monom-Darstellung des Polynoms zu entwickeln. Wir verwenden dazu die erzeugende 3-Term-Rekursion

$$T_{k+1}(x):=2xT_k(x)-T_{k-1}(x) \qquad , k=1,2,\ldots \quad .$$

Dann gilt

$$\begin{aligned}
p_n(x) &= \underbrace{a_n}_{:=b_n}T_n(x)+a_{n-1}T_{n-1}(x)+\ldots a_1T_1(x)+a_0T_0(x)\\
&\qquad \downarrow T_n(x):=2xT_{n-1}(x)-T_{n-2}(x)\\
&= \underbrace{(a_n2x+a_{n-1})}_{:=b_{n-1}}T_{n-1}(x)+(-a_n+a_{n-2})T_{n-2}(x)+\ldots+a_0T_0(x)\\
&\vdots\\
&= b_1T_1(x)+(-b_2+a_0)T_0(x)\\
&= xb_1+(-b_2+a_0)\cdot 1:=b_0
\end{aligned}$$

Algorithmus:

$$\begin{aligned}
b_n &:= a_n\\
b_{n-1} &:= 2xb_n+a_{n-1}\\
b_{n-k} &:= 2xb_{n-k+1}+a_{n-k}-b_{n-k+2} \qquad ,k=2,\ldots,n-1\\
b_0 &:= xb_1+a_0-b_2\\
&= p_n(x)
\end{aligned}$$

Rechenschema:

	a_n	a_{n-1}	a_{n-2}	...	a_2	a_1	a_0
	÷	÷	$-b_n$	...	$-b_4$	$-b_3$	$-b_2$
x	÷	$2xb_n$	$2xb_{n-1}$	...	$2xb_3$	$2xb_2$	xb_1
	b_n	b_{n-1}	b_{n-2}	...	b_2	b_1	$\underline{\lvert b_0=p_n(x)}$

Mit Hilfe von Tschebyscheff-Polynomen können jetzt Funktionen approximativ dargestellt werden, ohne dass das Rungesche Phänomen auftritt.

Zielstellung: Der Fehlerterm der Polynominterpolation ist durch

$$e_n(x) = \omega(x)\frac{f^{(n+1)}(\xi)}{(n+1)!} .$$

gegeben und es soll der maximale Fehler minimiert werden. Insbesondere soll das Knotenpolynom $\omega(x) = (x-x_0)(x-x_1)\ldots(x-x_n)$ betragsmäßig kleine Werte annehmen durch geeignete Wahl der Stützstellen x_i .

Wir betrachten zunächst als Basisintervall das Intervall $x \in [-1,1]$.

Satz 6.10.: Sei n fest und $x \in [-1,1]$. Bei beliebiger Wahl der Knoten $\{x_0, x_1, ..., x_n\}$ aus $[-1,1]$ ist das Polynom $T(x)$ mit

$$T(x) = \frac{1}{2^n}T_{n+1}(x) = (x-x_0)(x-x_1)....(x-x_n)$$

das einzige mit führendem Koeffizienten 1, für welches gilt

$$\frac{1}{2^n} = \max_{x\in[-1,1]}|T(x)| \le \max_{x\in[-1,1]}|\omega(x)| .$$

Damit: Der Fehlerterm $e_n(x)$ der Polynominterpolation wird minimal, wenn $\omega(x) = T(x) = \frac{1}{2^n}T_{n+1}(x)$ ist, d.h. die Knoten $x_0, x_1, \ldots, x_n$ sind die Nullstellen des Tschebyscheffpolynoms $T_{n+1}(x)$ (Tschebyscheff-Knoten).

Allgemein kann man zeigen, dass die Interpolationspolynome auf der Basis von Tschebyscheffknoten mit wachsenden n verbesserte Approximationseigenschaften besitzen :

Satz 6.11.: Sind $\{x_0, x_1, ..., x_n\}$ Stützstellen in $[-1,1]$, $y_i = f(x_i)$ gegebene Stützwerte ($i = 0,\ldots,n$) und sind $f(x), f'(x)$ stetig auf $[-1,1]$, dann liefert die Interpolation mit Tschebyscheff-Knoten eine Folge $\{P_n(x)\}$ von Polynomen, welche in $[-1,1]$ für $n \to \infty$ gleichmäßig gegen $f(x)$ konvergiert.

Lagrange-Tschebyscheff-Approximation in $[-1,1]$

Statt der (n+1) Basispolynome $L_0, \ldots, L_n$ wird das System der (n+1) linear unabhängigen Tschebyscheff-Polynome $T_0, \ldots, T_n$ als Basis von Π_n verwendet.

Aufgabe: geg.: $x_0, x_1, \ldots, x_n$ Tschebyscheff-Knoten in $[-1,1]$
$y_0, \ldots, y_n$ Stützwerte ; $y_i = f(x_i)$
ges.: $P_n(x) = c_0T_0(x) + c_1T_1(x) + \ldots + c_nT_n(x)$ (6.21)
Interpolationspolynom in Tschebyscheff-Darstellung.

Die Koeffizienten $c_0, \ldots, c_n$ sind jetzt nicht wie in der Lagrangeschen Interpolationsformel gegeben, da für die $T_i(x)$ nicht die Eigenschaft der Lagrangepolynome gilt $L_i(x_j) = \delta_{ij}$.

Berechnung der Koeffizienten: Wir nutzen Orthogonalitätseigenschaften der Tscheby-scheff-Polynome bezüglich der Tschebyscheff-Knoten $x_i = \cos\left(\pi \frac{2i+1}{2n+2}\right)$ $(i = 0,1,...,n)$ aus:

$$\sum_{i=0}^{n} T_0(x_i)^2 = n+1, \quad \sum_{i=0}^{n} T_j(x_i)^2 = \frac{n+1}{2}, \quad j \neq 0$$

$$\sum_{i=0}^{n} T_k(x_i) T_j(x_i) = 0 \qquad \text{für} \quad j \neq k \tag{6.22}$$

Durch Multiplikation des Ansatzes (6.21) mit $T_0(x)$, einsetzen von $x_0, x_1, ..., x_n$ und Summation der entstehenden Beziehungen folgt:

$$\sum_{i=0}^{n} y_i T_0(x_i) = c_0(n+1) + c_1 \cdot 0 + ... + c_n \cdot 0 \ .$$

Analog erhält man durch Multiplikation von (6.21) mit $T_k(x)$ den Koeffizienten c_k und es folgt für die Koeffizienten:

$$c_0 = \frac{1}{n+1} \sum_{i=0}^{n} y_i T_0(x_i) = \frac{1}{n+1} \sum_{i=0}^{n} y_i$$

$$c_k = \frac{2}{n+1} \sum_{i=0}^{n} y_i T_k(x_i) = \frac{2}{n+1} \sum_{i=0}^{n} y_i \cos\left(k\pi \frac{2i+1}{2n+2}\right)$$

$$k = 1,2,...,n \tag{6.23}$$

Die Auswertung von $P_n(x) = c_0 T_0(x) + ... + c_n T_n(x)$ für beliebiges $x \in [-1,1]$ kann mit Hilfe des Clenshaw-Algorithmus erfolgen.

Allgemeines Intervall $[a,b]$:

Nicht immer wird eine Funktion $f(t)$ im Intervall $[-1,1]$ zu approximieren sein, sondern in einem allgemeineren Intervall $[a,b]$. Wir transformieren dann die Variable t so, dass die neue Variable x in $[-1,1]$ ist:

$$t = \frac{b-a}{2} x + \frac{a+b}{2} \qquad \text{bzw.} \quad x = 2\frac{t-a}{b-a} - 1. \tag{6.24}$$

Sind jetzt $x_0, x_1, ..., x_n$ die Tschebyscheff-Knoten bezüglich $[-1,1]$ (Nullstellen von $T_{n+1}(x)$)

$$x_i = \cos\frac{(2i+1)\pi}{2(n+1)} \qquad (\ i = 0,1,2,...,n\ ), \quad \text{so sind} \quad t_i = \frac{b-a}{2} x_i + \frac{a+b}{2}$$

die Tschebyscheff-Knoten bezüglich $[a,b]$.

Algorithmus (Lagrange-Tschebyscheff-Approximation von $f(t)$ über $[a,b]$):

S0: $a, b, f(t)$ und n gegeben, sowie die Stelle $t \in [a,b]$ an der das Interpolationspolynom $Q_n(t)$ für $f(t)$ auszuwerten ist (Neustelle).

S1: Berechnung der Tschebyscheff-Knoten $x_i \quad (i = 0,1,...,n)$, der Werte $t_i \in [a,b]$ und der Stützwerte $y_i = f(t_i)$.

S2: Berechnung der Koeffizienten $c_0, c_1, ..., c_n$ aus (6.23).

S3: Auswertung des Interpolationspolynoms an der Stelle t:

Berechnung von $x := 2\dfrac{t-a}{b-a} - 1$ und Auswertung von $P_n(x)$ mittels Clenshaw-Algorithmus. Dann gilt $Q_n(t) := P_n(x)$.

6.7.4. Padé-Approximation

Die Approximation einer Funktionen $f(x)$ kann häufig bedeutend verbessert werden, wenn nicht Polynome, sondern rationale Funktionen verwendet werden, d.h.

$$f(x) = \frac{P_n(x)}{Q_m(x)} + e_{n,m}(x) \quad , \quad x \in [a,b]. \tag{6.25}$$

Dabei ist $R_{n,m}(x) = \dfrac{P_n(x)}{Q_m(x)}$ die approximierende rationale Funktion und $e_{n,m}(x)$ der Approximationsfehler.

Voraussetzung: $f(x)$ und die Ableitungen der Funktion sind stetig in $x = 0$.
(Durch Variablentransformation kann jedes Intervall so transformiert werden, dass $x = 0$ enthalten ist.)

Konstruktionsprinzip für die approximierende rationale Funktion $R_{n,m}(x) = \dfrac{P_n(x)}{Q_m(x)}$:

$$P_n(x) = p_0 + p_1 x + ... + p_n x^n$$
$$Q_m(x) = 1 + q_1 x + ... + q_m x^m$$

werden so bestimmt, dass $f(x)$ und $\dfrac{P_n(x)}{Q_m(x)}$ in $x = 0$ übereinstimmen bis zur Ableitung der Ordnung $n+m$.

Wahl von n,m: Bei fester Summe $n+m$ ist der Fehler am kleinsten, wenn $m = n$ ist oder es gilt n=m+1, d.h. der Zähler ist höchstens um einen Grad höher als der Nenner.

Berechnung der Koeffizienten:

Sind m, n festgelegt, so gibt es $n+m+1$ unbekannte Größen $p_0, p_1, ..., p_n, q_1, ..., q_m$.
Wir nehmen an, dass $f(x)$ analytisch ist, d.h. die Funktion besitzt eine Taylorentwicklung. Dann gilt:

$$f(x) = a_0 + a_1 x + a_2 x^2 + ... + a_k x^k + ... = \frac{P_n(x)}{Q_m(x)} + e_{n,m}(x)$$

bzw.

$$f(x) Q_m(x) - P_n(x) = e_{n,m}(x) Q_m(x) = Z(x).$$

Da die ersten $n+m$ Ableitungen von $f(x)$ und $\frac{P_n(x)}{Q_m(x)}$ in $x=0$ übereinstimmen sollen, darf die Entwicklung des Fehlers $Z(x)$ erst mit der Potenz $n+m+1$ beginnen.

$$f(x)Q_m(x)-P_n(x)=\left(\sum_{i=0}^{\infty}a_i x^i\right)\left(\sum_{j=0}^{m}q_j x^j\right)-\sum_{k=0}^{n}p_k x^k=\sum_{l=n+m+1}^{\infty}c_l x^l$$

Ausmultiplizieren und Koeffizientenvergleich für die Koeffizienten von $x^0, x^1, x^2, \ldots, x^{n+m}$ liefert wegen $q_0=1$ das System von Gleichungen:

$$\begin{aligned}
&x^0: && a_0-p_0=0\\
&x^1: && q_1a_0+a_1-p_1=0\\
&x^2: && q_2a_0+q_1a_1+a_2-p_2=0\\
&\vdots\\
&x^n: && q_ma_{n-m}+q_{m-1}a_{n-m+1}+\ldots+a_n-p_n=0\\
&\text{---------} && \text{---------}\\
&x^{n+1}: && q_ma_{n-m+1}+q_{m-1}a_{n-m+2}+\ldots+q_1a_n+a_{n+1}=0\\
&\vdots && \vdots\\
&x^{n+m}: && q_ma_n+q_{m-1}a_{n+1}+\ldots+q_1a_{n+m-1}+a_{n+m}=0
\end{aligned} \qquad (6.26)$$

Kennzeichen: 1. Die Summe der Indizes der Faktoren entspricht immer der x-Potenz.
2. Die letzten m Gleichungen sind ein lineares Gleichungessystem in den Unbekannten $q_1,\ldots,q_m$. Sind diese Größen berechnet, so können $p_0,\ldots,p_n$ aus den ersten n Gleichungen direkt berechnet werden.

6.8. Splines

Ziel: Wir wollen eine möglichst glatte Kurve durch Stützstellen (x_i, y_i) legen und dazu Polynome niederen Grades verwenden.
Eine Möglichkeit war die kubische Hermiteinterpolation; jetzt sollen spezielle stückweise Polynome verwendet werden.

Def. 6.12: Es sei $\Delta=\{x_0,\ldots,x_n\}$ eine Menge von $n+1$ verschiedenen Stützknoten $a=x_0<x_1<\ldots<x_n=b$ und $s(x)\in C^{q-1}[a,b]$. Stimmt $s(x)$ auf jedem Intervall $[x_i, x_{i+1}]$ mit einem Polynom $s_i(x)$ vom Grad q überein, so heißt $s(x)$ eine Spline vom Grad q .
Den linearen Vektorraum der Splines vom Grad q bezüglich der Knotenmenge Δ bezeichnen wir mit $S_{q,\Delta}$.

Besonders wichtig sind die Fälle:

- q=1 $s(x)\in C^0[a,b]$, d.h. $s(x)$ ist stetig und linear auf jedem Teilintervall (Polygonzug oder lineare Spline)
- q=2 $s(x)\in C^1[a,b]$, d.h. $s(x)$ ist stetig differenzierbar und quadratisch (quadrat. Spline)

q=3 $s(x) \in C^2[a,b]$, d.h. $s(x)$ ist 2 mal stetig differenzierbar auf $[a,b]$ und kubisch auf jedem Teilintervall (kubische Spline).

Kubische Splines sind besonders interessant für Graphikanwendungen, Lösung von Differenzialgleichungen u.a..

Lokale Darstellungsform einer Spline $s(x) \in S_{q,\Delta}$:

Nach Definition 6.12. stimmt $s(x)$ auf jedem Intervall $[x_i, x_{i+1}]$ mit einem Polynom $s_i(x)$ vom Grad q überein. Damit erhält man als lokale Darstellung von $s(x)$ in $[x_i, x_{i+1}]$:

$$s(x) = s_i(x) = a_0^{(i)} + a_1^{(i)}(x - x_i) + a_2^{(i)}(x - x_i)^2 + \ldots + a_q^{(i)}(x - x_i)^q \quad , i = 0,\ldots,n-1 \qquad (6.27)$$

Es sind somit $(q+1)\cdot n$ Koeffizienten $a_j^{(i)}$ zu bestimmen. Da $s(x)$ über $[a,b]$ $(q-1)$-mal stetig differenzierbar sein soll, müssen die folgenden Übergangsbedingungen an den inneren Knoten $x_1,\ldots,x_{n-1}$ von $[a,b]$ erfüllt sein:

$$\left.\begin{array}{c} s_{i-1}(x_i) = s_i(x_i) \\ s'_{i-1}(x_i) = s'_i(x_i) \\ \vdots \\ s_{i-1}^{(q-1)}(x_i) = s_i^{(q-1)}(x_i) \end{array}\right\} \quad i = 1,\ldots,n-1 \qquad (6.28)$$

Damit: Für die $(q+1)\cdot n$ Koeffizienten müssen $q(n-1)$ Übergangsbedingungen erfüllt sein, es sind somit noch $n+q$ Bedingungen frei um die Spline eindeutig zu bestimmen (die freien Bedingungen entsprechen der Anzahl der Basiselemente von $S_{q,\Delta}$).Stellt man (n+1) Interpolationsbedingungen, so bleiben (q-1) Zusatzbedingungen zu stellen.

6.9. Interpolierende kubische Splines

Dieser Fall (q=3) ist der wichtigste für die Anwendungen: Die Spline $s(x)$ interpoliert über n+1 Stützstellen und ist in $[a,b]$ zweimal stetig differenzierbar. Da für das menschliche Auge noch Unstetigkeiten der 1. Ableitung erkennbar sind, erscheint die Funktion $s(x)$glatt daraus resultiert ihre Verwendung in Graphik - und CAD - Algorithmen.
Wir berechnen $s(x)$ in der lokalen Darstellung als stückweise kubische Funktion

$$s(x) = s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 \quad x \in [x_i, x_{i+1}] \quad i = 0,1,\ldots,n-1 \qquad (6.29)$$

Wir spezialisieren den Ansatz (6.29) durch die Einführung von Hilfsgrößen
$y'_i,\ y''_i \quad (i = 0,1,\ldots,n)$

$$s_i(x) = y_i + y'_i(x - x_i) + \frac{y''_i}{2}(x - x_i)^2 + \frac{y''_{i+1} - y''_i}{6h_i}(x - x_i)^3 \quad x \in [x_i, x_{i+1}] \qquad (6.30)$$

Dann gilt für die Ableitungen

$$s_i'(x) = y_i' + y_i''(x-x_i) + \frac{y_{i+1}''-y_i''}{2h_i}(x-x_i)^2$$
$$s_i''(x) = y_i'' + \frac{y_{i+1}''-y_i''}{h_i}(x-x_i)$$

Bedingung für 2. Ableitungen:

$$s_i''(x_i) = y_i'' \ ;$$
$$s_{i-1}''(x_i) = y_{i-1}'' + \frac{y_i''-y_{i-1}''}{h_{i-1}}h_{i-1} = y_i'' \quad i = 1,2,...,n-1$$

d.h. die Stetigkeit der 2. Ableitungen an den inneren Punkten ist durch den Ansatz erfüllt, wobei $h_i = x_{i+1} - x_i$ bezeichnet.

Bedingung für 1. Ableitungen:

$$s_i'(x_i) = y_i' \ ;$$
$$s_{i-1}'(x_i) = y_{i-1}' + y_{i-1}''h_{i-1} + (y_i''-y_{i-1}'')\frac{h_{i-1}}{2} \ ,$$

d.h. Stetigkeit der 1. Ableitungen an inneren Knoten, wenn gilt

$$y_i' = y_{i-1}' + y_{i-1}''h_{i-1} + (y_i''-y_{i-1}'')\frac{h_{i-1}}{2} = y_{i-1}' + (y_i''+y_{i-1}'')\frac{h_{i-1}}{2} \quad i = 1,2,...,n-1 \qquad (6.31)$$

Bedingung für Funktionswerte:

$s_i(x_i) = y_i \quad i = 0,1,\ldots,n-1$, d.h. die Interpolationsbedingungen sind erfüllt durch den Ansatz.

Stetigkeit des Übergangs und Interpolationsbedingung (für $i = n$)

$$s_{i-1}(x_i) = y_i = y_{i-1} + y_{i-1}'h_{i-1} + \frac{y_{i-1}''}{2}h_{i-1}^2 + \frac{y_i''-y_{i-1}''}{6h_{i-1}}h_{i-1}^3 \quad i = 1,\ldots,n \qquad (6.32)$$

Aus der Beziehung (6.32) kann y_{i-1}' eliminiert werden; durch Einsetzen in die Beziehung (6.31) für die ersten Ableitungen erhalten wir ein System in welchem nur noch die Hilfsgrößen y_i'' vorkommen:

$$y_{i-1}' = \frac{y_i - y_{i-1}}{h_{i-1}} - \frac{1}{3}y_{i-1}''h_{i-1} - \frac{1}{6}y_i''h_{i-1} \quad i = 1,...,n$$

bzw.

$$y_i' = \frac{y_{i+1} - y_i}{h_i} - \frac{1}{3}y_i''h_i - \frac{1}{6}y_{i+1}''h_i \quad i = 0,...,n-1 \qquad (6.33)$$

und aus (6.31) folgt nach Multiplikation mit dem Faktor 6:

$$y_{i-1}''h_{i-1} + (2\,h_{i-1} + 2\,h_i)y_i'' + y_{i+1}''h_i = 6\left(\frac{y_{i+1}-y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}\right) \qquad (6.34)$$
$$i = 1,2,...,n-1$$

Das System (6.34) stellt ein lineares Gleichungssystem dar von n-1 Gleichungen für n+1

Unbekannte $y_0'', y_1'', \ldots, y_n''$. Wir bezeichnen es als **System der Grundgleichungen**. Um die Unbekannten eindeutig bestimmen zu können, müssen somit noch zwei Gleichungen als Zusatzbedingungen gestellt werden. Diese treten in der Regel in Form von **Randbedingungen** auf:

(I) Natürliche Randbedingungen:

geg.: $s''(x_0) = y_0'' = 0$;
$s''(x_n) = y_n'' = 0$ (6.35)

Das System (6.34) ist dann ein System von n-1 Gleichungen für n-1 Unbekannte $y_1'', \ldots, y_{n-1}''$. Nach Multiplikation mit dem Faktor 6 folgt:

$$\begin{pmatrix} 2(h_0+h_1) & h_1 & & & 0 \\ h_1 & 2(h_1+h_2) & h_2 & & \\ & \ddots & \ddots & \ddots & \\ & & & & h_{n-2} \\ 0 & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix} \begin{pmatrix} y_1'' \\ y_2'' \\ \vdots \\ \\ y_{n-1}'' \end{pmatrix} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \\ \\ \gamma_{n-1} \end{pmatrix} \tag{6.36}$$

$$\gamma_i = 6\left[\frac{y_{i+1}-y_i}{h_i} - \frac{y_i - y_{i-1}}{h_{i-1}}\right] \quad i = 1, \ldots, n-1 \tag{6.37}$$

Lösung von (6.36): Das System ist symmetrisch, tridiagonal, diagonal dominant und die Matrix ist somit positiv definit. Eine effektive Lösung kann mit Hilfe des Cholesky-Verfahrens erfolgen.

Bem.: Statt Nullrandwerte können auch Werte $s''(x_0) = y_0''$, $s''(x_n) = y_n''$ beliebig vorgegeben sein, dann sind γ_1 und γ_{n-1} zu verändern: $\gamma_1 := \gamma_1 - h_0 y_0''$, $\gamma_{n-1} := \gamma_{n-1} - h_{n-1} y_n''$.

(II) Hermite Randbedingungen:

geg.: $s'(x_0) = y_0'$;
$s'(x_n) = y_n'$ (6.38)

Die fehlenden 2 Gleichungen erhält man aus den Randbedingungen:

(6.33) für i=0 :

$$2h_0 y_0'' + h_0 y_1'' = 6\left[\frac{y_1 - y_0}{h_0} - y_0'\right] = \gamma_0 \tag{6.39a}$$

(6.31) für i=n mit (6.33) für i=n-1:

$$h_{n-1} y_{n-1}'' + 2h_{n-1} y_n'' = 6\left[y_n' - \frac{y_n - y_{n-1}}{h_{n-1}}\right] = \gamma_n \tag{6.39b}$$

Elimination liefert :

$$h_0 y_0'' = \frac{1}{2}\gamma_0 - \frac{h_0}{2} y_1'' \qquad h_{n-1} y_n'' = \frac{1}{2}\gamma_n - \frac{h_{n-1}}{2} y_{n-1}''$$

Durch Einsetzen in (6.34) für i=1 bzw. i=n-1 erhält man das System

$$\begin{pmatrix} \frac{3}{2}h_0+2h_1 & h_1 & & & & 0 \\ h_1 & 2(h_1+h_2) & h_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} & \\ 0 & & & h_{n-2} & 2h_{n-2}+\frac{3}{2}h_{n-1} & \end{pmatrix} \begin{pmatrix} y_1'' \\ y_2'' \\ \vdots \\ y_{n-1}'' \end{pmatrix} = \begin{pmatrix} \gamma_1 - \frac{1}{2}\gamma_0 \\ \gamma_2 \\ \\ \gamma_{n-2} \\ \gamma_{n-1} - \frac{1}{2}\gamma_n \end{pmatrix} \quad (6.40)$$

Ferner gilt für die fehlenden zweiten Momente:

$$y_0'' = \frac{1}{2h_0}\gamma_0 - \frac{1}{2}y_1'' \qquad y_n'' = \frac{1}{2h_{n-1}}\gamma_n - \frac{1}{2}y_{n-1}''.$$

(III) Periodische Randbedingungen : Als Voraussetzung sollte $y_0 = y_n$ erfüllt sein, damit das Interpolationsintervall $[a,b] = [x_0, x_n]$ ein volles Periodenintervall darstellt.

geg.: $s'(x_0) = y_0' = s'(x_n) = y_n'$;

$$s''(x_0) = y_0'' = s''(x_n) = y_n'' \quad (6.41)$$

Man beachte, dass nur die Gleichheit der Werte gefordert wird, aber kein konkreter Wert vorgeschrieben ist. Das System (6.34) der Grundgleichungen sind n-1 Gleichungen für n Unbekannte $y_1'',\ldots,y_n''$. Der Wert y_0'' kann wegen $y_0'' = y_n''$ eliminiert werden.

(6.32) für i=0 ergibt mit (6.31) für i=n und (6.32) für i=n-1 die Zusatzgleichung:

$$\tilde{\gamma}_n = 6\left[\frac{y_1 - y_0}{h_0} - \frac{y_n - y_{n-1}}{h_{n-1}}\right] = y_1''h_0 + 2(h_0 + h_{n-1})y_n'' + h_{n-1}y_{n-1}'' \quad (6.42)$$

Das lineare Gleichungssystem zur Berechnung von $y_1'',\ldots,y_n''$ erhält jetzt die Form:

$$\begin{pmatrix} 2(h_0+h_1) & h_1 & & & & h_0 \\ h_1 & 2(h_1+h_2) & h_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} & \\ h_0 & & & h_{n-1} & 2(h_0+h_{n-1}) & \end{pmatrix} \begin{pmatrix} y_1'' \\ y_2'' \\ \vdots \\ y_{n-1}'' \\ y_n'' \end{pmatrix} = \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \\ \gamma_{n-1} \\ \tilde{\gamma}_n \end{pmatrix} \quad (6.43)$$

Die tridiagonale Struktur der Systemmatrix ist gestört, sie bleibt symmetrisch, diagonal dominant und damit positiv definit.

6.10. Approximationseigenschaften kubischer Splines

Wir nehmen jetzt an , daß die Stützstellen $(x_i, y_i) \quad i = 0,1,\ldots,n$ durch die Funktionswerte einer Funktion $y = f(x) \in C^2[a,b]$ gegeben sind, d.h.

$$y_i = f(x_i) \quad i = 0,1,\ldots,n$$

Frage: Wie approximiert die Spline $s(x)$ für $x \in [a,b]$ die Funktion $f(x)$?

Vor.: Sowohl $s(x)$ als auch $f(x)$ erfüllen den gleichen Typ von Randbedingungen, d.h.

(I) $s''(x_0) = s''(x_n) = 0$ und $f''(x_0) = f''(x_n) = 0$ oder

(II) $s'(x_0) = y'_0$, $s'(x_n) = y'_n$ und $f'(x_0) = y'_0$, $f'(x_n) = y'_n$ oder

(III) $s'(x_0) = s'(x_n)$, $s''(x_0) = s''(x_n)$ und $f'(x_0) = f'(x_n)$, $f''(x_0) = f''(x_n)$.

Satz 6.13. Unter allen zweimal stetig differenzierbaren Funktionen $y = \varphi(x)$ auf dem Intervall $[a,b]$, welche die obigen Interpolationsbedingungen und die entsprechenden Randbedingungen erfüllen, ist die kubische Splinefunktion $s(x)$ die glatteste, d.h. es gilt

$$\int_a^b s''(x)^2\,dx \le \int_a^b \varphi''(x)^2\,dx \ .$$

Bemerkung:

$\kappa(x) = \left(\dfrac{\varphi''(x)}{\sqrt{\left(1+\varphi'(x)^2\right)^3}} \right)$ ist die Krümmung von $\varphi(x)$ im Punkt x und $\dfrac{1}{\kappa(x)}$ der Krümmungsradius. Die Größe $E = \int_a^b \kappa(x)^2\,dx$ misst dann z.B. die Biegeenergie einer Leiste, die in der Form $\varphi(x)$ verformt wird.

Damit: $s(x)$ ist die glatteste Funktion in Sinne einer minimalen Gesamtkrümmung über $[a,b]$, welche die Interpolation und Randbedingungen erfüllt.

Approximationseigenschaften von $s(x)$ bezüglich der Funktion $f(x)$ von welcher die Funktionswerte y_i bekannt sind:

Satz 6.14: Es sei $f \in C^4[a,b]$ und $s(x)$ sei die interpolierende kubische Spline. Dann gilt für $x \in [x_i, x_{i+1}]$ $\quad i = 0,1,\ldots,n-1$

$$|f(x) - s(x)| \le c_1 H^2 \cdot h_i^2$$

$$|f'(x) - s'(x)| \le c_2 H^2 \cdot h_i$$

$$|f''(x) - s''(x)| \le c_3 H^2$$

$$|f'''(x) - s'''(x)| \le c_4 \frac{H^2}{h_i}$$

wobei $H = \max\limits_{i \in \{0,\ldots,n-1\}} h_i$ ist und die c_i sind von $f(x)$ abhängige Konstanten.

Damit: $s(x)$ approximiert eine 4-mal stetig differenzierbare Funktion $f(x)$ sowohl in den Funktionswerten als auch den Ableitungen bis zur 3. Ordnung. Für $h_i \to 0$ kann der Approximationsfehler beliebig klein gemacht werden.

Praktische Bedeutung kubischer Splines:

- einfache Berechnung über numerisch stabile Gleichungssysteme;
- minimale Gesamtkrümmung, d.h. sehr glatt ohne Oszillationen wie Polynome;
- Dämpfung von Störungen in Stützstellen, d.h. Änderungen eines Stützwertes y_i wirken sich nur lokal in Umgebung der Stützstelle aus;
- invariant gegenüber anderer Skalierung der Variablen.

6.11. Kubische "not-a-knot"-Splines

Wenn unbekannt ist, welche Randbedingungen die zu approximierende Funktion f(x) erfüllt, so wird i.a. die interpolierende Spline s(x) nicht die Randbedingungen von f(x) treffen. In diesen Fällen gilt die Fehlerabschätzung von Satz 6.14. nicht mehr. Bei äquidistanten Stützstellen und Verwendung natürlicher Randbedingungen kann man zeigen, daß der Fehler $|f(x)-s(x)|$ für $h \to 0$ nur wie h^2 abnimmt und nicht wie zu erwarten ist wie die Potenz h^4.
Eine kubische Splineapproximation der Fehlerordnung h^4 erhält man mit den von de Boor eingeführten "not-a-knot" Spline bzw. mit extrapolierten Randbedingungen. Statt der bisherigen Bedingungen in Randpunkten werden 2 Zusatzbedingungen gestellt, dass in x_1 *und* x_{n-1} auch die 3.Ableitungen der kubischen Polynomstücke gleich sein sollen:

(IV) Extrapolierte Randbedingungen: $s_0'''(x_1) = s_1'''(x_1)$, $s_{n-2}'''(x_{n-1}) = s_{n-1}'''(x_{n-1})$.

Wegen des Ansatzes (6.30) folgt

$$s_0'''(x_1) = \frac{y_1'' - y_0''}{h_0} = \frac{y_2'' - y_1''}{h_1} = s_1'''(x_1)$$

bzw. eine analoge Beziehung für die zweite Zusatzbedingung. Umformung der Gleichungen liefert die folgende Form der zwei Zusatzgleichungen zum System (6.34) der Grundgleichungen

$$\begin{aligned} -h_1 y_0'' + (h_0 + h_1) y_1'' - h_0 y_2'' &= 0, \\ -h_{n-1} y_{n-2}'' + (h_{n-2} + h_{n-1}) y_{n-1}'' - h_{n-2} y_n'' &= 0. \end{aligned} \qquad (6.44)$$

Bemerkung zur Bezeichnung der Spline:

1. Bezeichnet

$$g(x) = y_1'' + (x - x_1)\frac{y_2'' - y_1''}{h_1}$$

die lineare Funktion, welche bezüglich der zweiten Ableitungen die Interpolationsbedingungen erfüllt: $g(x_1) = y_1''$, $g(x_2) = y_2''$, so wird y_0'' extrapolierter Wert genannt, wenn $y_0'' = g(x_0)$ gesetzt wird:

$$y_0'' = g(x_0) = y_1'' - \frac{h_0}{h_1}(y_2'' - y_1'').$$

Dies entspricht gerade der ersten Zusatzgleichung. Analog lässt sich die zweite Zusatzgleichung als Extrapolation des Ableitungswerts y_n'' mittels y_{n-2}'', y_{n-1}'' deuten.

2. Betrachtet man die kubischen Teilfunktionen $s_0(x), s_1(x)$, so gilt wegen der Konstruktion der Spline

$$s_0(x_1) = s_1(x_1), \quad s_0'(x_1) = s_1'(x_1), \quad s_0''(x_1) = s_1''(x_1)$$

und wegen der Zusatzbedingung gilt $s_0'''(x_1) = s_1'''(x_1)$. Diese 4 Bedingungen definieren eine eindeutige kubische Funktion, d.h. es gilt $s_0(x) \equiv s_1(x)$ *auf* $[x_0, x_2]$. Analog erhält man $s_{n-2}(x) \equiv s_{n-1}(x)$ *auf* $[x_{n-2}, x_n]$. Die Knoten x_1, x_{n-1} sind somit keine echten Splineknoten, woraus die Bezeichnung "not-a-knot" Spline resultiert.

7. Numerische Integration

Ziel: Numerische Berechnung eines bestimmten Integrals $I = \int_a^b f(x)dx$ ohne die Stammfunktion F (x) des Integranden zu kennen. Wegen der geometrischen Deutung des Integrals als Flächeninhalt wird die Integralberechnung auch als Quadratur bezeichnet, ihre Notwendigkeit ergibt sich aus mehreren Gründen:

- Die Stammfunktion F(x) ist nicht geschlossen angebbar, z. B.
  elliptische Integrale, Gaußsches Fehlerintegral $\int_0^\infty \exp(-\frac{t^2}{2})dt$.
- Der Integrand liegt nur durch eine Wertetabelle vor (Messungen, punktweise Berechnungen aus kompliziertem Zusammenhang).

Verfahren: Die Auswahl des Verfahrens richtet sich nach Eigenschaften des Integranden (Glattheit, Singularitäten) und der gewünschten Genauigkeit. I. a. ist die numerische Quadratur ein sehr stabiler Prozess, der nur Funktionswerte des Integranden benutzt.

Bem: Unbestimmte Integrale werden zweckmäßig als AWP gewöhnlicher DGL behandelt, denn es gilt

$$y = f(t) = \int_a^t g(x)dx \quad gdw. \quad y_0 = f(a) = 0, \quad y' = g(t) \quad (t \geq a).$$

7.1. Trapezformel, Trapezsumme

Die Trapezmethode ist eine der einfachsten Quadraturformeln.
Das Integrationsintervall [a, b] wird in N Teilintervalle zerlegt durch Einführung von Stützstellen $a = x_0 < x_1 < x_2 < \ < x_N = b$ und die Funktion f(x) wird durch eine Linearisierung (Polygonzug) f*(x) ersetzt.

N = 1

$$I \approx \int_a^b f^*(x)dx = \frac{f(a) + f(b)}{2}(b-a)$$

bzw.

$$I = \frac{b-a}{2}(y_0 + y_1) + E_T \qquad (7.1)$$

Trapezformel, E_T Fehler

N > 1:

$$I \approx \sum_{\nu=1}^{N} \int_{x_{\nu-1}}^{x_\nu} f^*(x)dx = \sum_{\nu=1}^{N} \frac{f(x_{\nu-1}) + f(x_\nu)}{2}(x_\nu - x_{\nu-1}).$$

Häufig wird mit äquidistanter Unterteilung von [a, b] gearbeitet :

$$H=\frac{b-a}{N}, \quad x_k=a+kH, \; k=0,1,2,\ldots,N$$

$$I\approx\int_a^b f^*(x)dx=\left[\frac{f(x_0)+f(x_1)}{2}+\frac{f(x_1)+f(x_2)}{2}+\cdots\frac{f(x_{N-1})+f(x_N)}{2}\right]H$$

$$I=\frac{b-a}{N}\left[\frac{y_0+y_N}{2}+y_1+y_2+\cdots+y_{N-1}\right]+E_{TS} \qquad \text{(Trapezsumme)} \qquad (7.2)$$

Anwendung: 1. Im allgemeinen langsame Konvergenz. Vorteile bei periodischen Funktionen für die das Integrationsintervall [a, b] einem Periodenintervall entspricht. Auf unendlichen Intervallen nur dann, wenn der Intergrand schnell abklingt (etwa exponentiell).

2. Für praktische Anwendungen ist es sinnvoll die Trapezsumme mit einer weiteren Formel zu koppeln, um aus den Vergleich beider Werte eine Schätzung des Fehlers und damit eine Steuerung der Genauigkeit zu erhalten. Hier bietet sich im Vergleich die Mittelpunktsregel an, da die Trapezsumme zur halben Schrittweite $\frac{H}{2}$ auf die Funktionswerte des Integranden zugreift, die für die Mittelpunktsregel zur Schrittweite H benötigt werden:

$$I=\frac{b-a}{N}\left[f(a+\frac{H}{2})+f(a+3\frac{H}{2})+\ldots+f(a+(2N-1)\frac{H}{2})\right]+E_{MR} \qquad (7.3)$$

7.2. Keplersche Fassregel, SIMPSON-Formel

f(x) wird näherungsweise durch eine interpolierende Parabel ersetzt: $f^*(x)=c_0+c_1x+c_2x^2$ die in den Knoten $x_0=a, \quad x_1=\frac{a+b}{2}, \quad x_2=b$ Interpolationsbedingungen erfüllt:

$$f^*(a)=y_0, \quad f^*\left(\frac{a+b}{2}\right)=y_1, \quad f^*(b)=y_2.$$

Dann gilt

$$\int_a^b f^*(x)dx=\int_a^b\left(c_0+c_1x+c_2x^2\right)dx=c_0x+c_1\frac{x^2}{2}+c_2\frac{x^3}{3}\Big|_a^b$$

$$=\frac{1}{6}(b-a)\left[\left(c_0+c_1a+c_2a^2\right)+\left(c_0+c_1b+c_2b^2\right)+4\left(c_0+c_1\frac{a+b}{2}+c_2\left(\frac{a+b}{2}\right)^2\right)\right]$$

$$=\frac{1}{6}(b-a)\left[f^*(a)+f^*(b)+4f^*\left(\frac{a+b}{2}\right)\right]$$

$$\boxed{I=\frac{1}{6}(b-a)[y_0+4y_1+y_2]+E_K}$$ Keplersche Fassregel.

Summierte Fassregel, Simpson-Formel:

Das Intervall $[a,b]$ wird nun in N Doppelstreifen der Länge $H = 2h = \frac{b-a}{N}$ zerlegt d. h. $h = \frac{b-a}{2N}$ und man wendet auf jeden Doppelstreifen die Fassregel an:

$$I = \int_{x_0}^{x_2} f dx + \int_{x_2}^{x_4} f dx + \cdots + \int_{x_{2N-2}}^{x_{2N}} f dx$$

$$\approx \frac{1}{6} H\left[y_0 + 4y_1 + y_2\right] + \frac{1}{6} H\left[y_2 + 4y_3 + y_4\right] + \cdots \frac{1}{6} H\left[y_{2N-2} + 4y_{2N-1} + y_{2N}\right]$$

$$\boxed{I = \frac{b-a}{6N}\left[y_0 + y_{2N} + 2\left(y_2 + y_4 + \cdots y_{2N-2}\right) + 4\left(y_1 + y_3 + \cdots y_{2N-1}\right)\right] + E_{SR}} \qquad (7.4)$$

SIMPSON-Regel, Summierte Fassregel

7.3. Interpolatorische Quadratur

Allgemeines zur Theorie von Quadraturformeln

Betrachtet wird das Integral $I(f) = \int_a^b f(x)dx$ und eine Näherung von I von der Form

$$I_n(f) = \sum_{j=0}^{n} c_j f(x_j) \qquad (7.5)$$

mit $x_0 < x_1 < \ldots < x_n$. Die Werte c_j (abhängig von n) heißen **Koeffizienten der Näherungsformel,** die x_j **Stützstellen**. Die Größe $E_n(f) = I(f) - I_n(f)$ heißt **Fehler der Näherungsformel.** Die Koeffizienten $\{c_j\}$ werden so gewählt, daß für alle Funktionen aus einer gewissen Testklasse der Integrationsfehler $E_n(f) = 0$ ist gilt. Ist diese Testklasse von Funktionen f eine Teilklasse der Polynome, so spricht man von von einer **interpolatorischen Quadraturformel.**

Def.7.1: Eine Folge $\{I_n(f)\}$ von Quadraturformeln heißt **konvergent** für die Funktion f(x), wenn gilt $\lim_{n\to\infty} E_n(f) = 0$.

Satz 7.2: (Stecklov 1916)
Für die Stützstellen gelte $x_j \in [a,b]$ *für* $j = 0,1,\ldots,n$. Eine Folge von Quadraturformeln ist konvergent für jede Funktion $f \in C[a,b]$ genau dann, wenn sie konvergent für alle Polynome ist und für die Koeffizienten gilt für jedes n die Bedingung $\sum_{j=0}^{n} |c_j| < K$ mit einer von n unabhängigen Konstanten K.

Zur Charakterisierung der Genauigkeit einer Quadraturformel betrachtet man die Teilklasse aller Polynome für die der Fehler $E_n(f)$ Null ist.

Def. 7.3: Eine Quadraturformel besitzt den **Genauigkeitsgrad q**, wenn gilt $E_n(f) = 0$ für alle Polynome f vom Grad kleiner oder gleich q und es gilt $E_n(x^{q+1}) \neq 0$.
Eine Folge von Quadraturformeln der Form (5) besitzt die **Ordnung r**, wenn für alle genügend glatten Funktionen f gilt

$$E_n(f) = O(h^r) \quad \textit{für} \quad h \to 0, \quad \text{mit} \quad h = \max_{1\le j\le n}(x_j - x_{j-1}).$$

Die geschlossenen Newton-Cotes-Formeln

Wir wollen jetzt systematisch Grundformeln für die Integration entwickeln. Es bietet sich die bisherige Vorgehensweise an: f(x) wird durch ein interpolierendes Polynom P(x) = f*(x) ersetzt und wir nutzen eine äquidistante Intervallunterteilung.

Äquidistante Intervallteilung:

Im Fall $x_0 = a, x_n = b$ nennt man die entstehenden Formeln **geschlossen**, im Fall $a < x_0, x_n < b$ **offen** und wenn nur eine der Stützstellen mit dem Randpunkt übereinstimmt halboffen.

Wir zeigen die Entwicklung der geschlossenen Newton-Cotes-Formeln mit Stützknoten

$$h=\frac{b-a}{n}, \quad x_i=a+ih, \; i=0,1,2,\ldots,n$$

$P_n(x)$ sei interpolierendes Polynom vom Grad $\leq$ n mit

$$P_n(x_i)=y_i=f(x_i) \quad i=0,1,\cdots,n\,.$$

Unter Nutzung der Lagrangeschen Interpolationsformel:

$$P_n(x)=\sum_{i=0}^{n} y_i L_i(x)$$

mit

$$L_i(x)=\prod_{\substack{k=0\\k\neq i}}^{n}\frac{x-x_k}{x_i-x_k}=\frac{(x-x_0)(x-x_1)\cdots(x-x_{i-1})(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)(x_i-x_1)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)}.$$

sowie der Substitution:

$$x=a+sh, \quad 0\leq s\leq n \quad dx=hds$$
$$x-x_k=h(s-k), \quad x_i-x_k=h(i-k)$$

d. h. $$L_i(x)=\varphi_i(s)=\frac{(s-0)(s-1)\cdots(s-i+1)(s-i-1)\cdots(s-n)}{(i-0)(i-1)\cdots(i-i+1)(i-i-1)\cdots(i-n)}. \qquad (7.6)$$

folgt für die Integralberechnung:

$$I=h\sum_{i=0}^{n} y_j\alpha_j+\mathrm{E}_n(f) \quad mit \quad \alpha_j=\int_0^n \varphi_j(s)ds \quad j=0,1,\cdots,n \qquad (7.7)$$

α_i … Gewichte der Quadraturformel, nicht abhängig von der zu integrierenden Funktion f(x), d. h. sie können berechnet und tabelliert werden und sind für alle Integranden gültig.

Satz 7.4.: Der Genauigkeitsgrad der Quadraturformel (7.7) ist mindestens n, d. h. die Formel ist exakt für Funktionen f(x) die im Polynom vom Grad $\leq$ n sind: $E_n(f) = 0$.

Berechnung der Gewichte: Die Gewichte hängen davon ab, welcher Grad das Polynom $P_n(x)$ über [a, b] besitzt.

n = 1: $h=\frac{b-a}{n}=(b-a)$ $\quad \alpha_0=\int_0^1 \frac{s-1}{0-1}ds=\frac{1}{2}$ $\quad \alpha_1=\int_0^1 \frac{s-0}{1-0}ds=\frac{1}{2}$

$$I = h\,(\frac{1}{2}y_0 + \frac{1}{2}y_1) + E_1(f)$$

n = 2: $h=\frac{b-a}{n}=\frac{(b-a)}{2}$ $\quad \alpha_0=\frac{4}{3}$, $\quad \alpha_1=\frac{4}{3},\ldots$ $\quad \alpha_2=\frac{1}{3}$,

$$I=h\left[\frac{1}{3}y_0+\frac{4}{3}y_1+\frac{1}{3}y_2\right]+E=\frac{b-a}{6}\left[y_0+y_1+y_2\right]+E_2(f)$$

(Keplersche Fassregel)

Allgemein:

Für alle natürlichen Zahlen n erhält man Integrationsformeln, die für $f(x)=P_n(x)$ (Polynom vom Grad ≤ n) exakt sind.

Speziell:

$f(x)=P_0(x)\equiv 1$

$$\int_a^b 1dx=(b-a)=\frac{b-a}{n}\sum_{i=0}^{n}\alpha_i \qquad \boxed{\sum_{i=0}^{n}\alpha_i = n} \qquad (7.8)$$

Bedingung für Gewichte.

Fehlerformel:

Da f(x) in [a, b] durch $P_n(x)$ angenähert wird, gilt für den Integrationsfehler

$$E_n(f)=\int_a^b [f(x)-P_n(x)]dx \quad =\int_a^b (x-x_o)(x-x_1)\cdots(x-x_n)\frac{f^{n+1}(\xi(x))}{(n+1)!}dx;\ \xi(x)\in(a,b)$$

wegen der Fehlerformel für die Polynominterpolation.

Bezeichnung: $w(x)=(x-x_0)(x-x_1)\cdots(x-x_n)$

Satz 7.5.:

(a) Ist n gerade und $f\in C^{n+2}[a,b]$, dann gilt

$$E_n(f)=\frac{k_n}{(n+2)!}f^{(n+2)}(\xi) \quad mit \quad \begin{matrix} a<\xi<b \\ k_n=\int_a^b xw(x)dx \end{matrix} \qquad (7.9a)$$

(b) Ist n gerade und $f\in C^{n+2}[a,b]$, dann gilt

$$E_n(f)=\frac{k_n}{(n+1)!}f^{(n+1)}(\xi) \quad mit \quad \begin{matrix} a<\xi<b \\ k_n=\int_a^b w(x)dx \end{matrix} . \qquad (7.9b)$$

Schlußfolgerung:

Für eine geschlossene Newton-Cotes-Formel auf der Basis von (n+1) Stützstellen gilt somit: Für gerades n ist die Intergrationsformel exakt vom Grad n+1 und für ungerades n exakt vom Grad n. In der nachfolgenden Tabelle sind die Gewichte der geschlossene Newton-Cotes-Formel für n=1 bis n=7 aufgeführt.

Berechnung der Fehlerkonstanten k_n:

n = 1: $$k_1 = \int_a^b (x-a)(x-b)dx = -\frac{(b-a)^3}{6} = -\frac{h^3}{6}$$

d.h. $$|E_1(f)| \le \frac{(b-a)^3}{12} M_2, \quad M_2 = \max_{x\in[a,b]} |f''(x)|$$

n = 2: $$K_2 = \int_a^b x(x-a)\left(x-\frac{a+b}{2}\right)(x-b)dx = -\frac{4}{15}h^5 = -\frac{4}{15}\frac{(b-a)^5}{2^5}$$

$$|E_2(f)| \le \frac{4}{15\cdot 4!}h^5 M_4 = \frac{1}{90}h^5 M_4 = \frac{1}{2880}(b-a)^5 M_4 \quad \text{mit } M_4 = \max_{x\in[a,b]} |f^4(x)| \ .$$

Problem: Zur Erreichung einer vorgegebenen Genauigkeit könnte man nun n genügend hoch wählen und damit eine Integratonsregel n-ter Ordnung entwickeln.Ab n=8 treten jedoch auch negative Gewichte auf. Dies führt dazu, dass wegen des Zusammenhangs $c_j = h * \alpha_j$ die Bedingung $\sum |c_j| < K$ in Satz 7.2.nicht erfüllt ist. Die Newton-Cotes Formeln liefern somit keine konvergente Folge von Quadraturformeln. Durch den Übergang zu summierten Formeln über immerkleinere Teilintervalle kann dieser Nachteil ausgeglichen werden: Wie bei der SIMPSON-Regel wird dabei zunächst [a, b] in N Grobintervalle der Länge $H = \frac{b-a}{N}$ geteilt und jedes der Grobintervalle durch n Stützstellen unterteilt mit Abstand $h = \frac{H}{n} = \frac{b-a}{N\cdot n}$. Damit entstehen summierte Formeln. Für jedes Teilintervall der Länge H gilt dann die Fehlerformel (lokaler Fehler). Der globale Fehler ist die Summe der lokalen Fehler über alle Grobintervalle.

Globale Fehlerabschätzung

Für den lokalen Fehler gilt immer eine Abschätzung der Form

$$E^i = \left| \int_x^{x+H} (f(x) - P_n(x))dx \right| = |E_n(f)| \le h^{p+1}\cdot K \cdot |f^{(p)}(\xi)| \qquad \text{mit} \qquad \xi \in (x, x+H)$$

dann folgt für den globalen Fehler

$$E \le \sum_{i=1}^N E^i = \sum_{i=1}^N h^{p+1} K \cdot \max_{x\in I_i} |f^{(p)}(x)| \ \le h^p K \cdot \max_{x\in[a,b]} |f^{(p)}| \sum_{i=1}^N h = h^p \cdot K \cdot M_p \cdot \frac{b-a}{n}$$

$$\boxed{E \le \frac{b-a}{n} K \cdot M_p h^p, \ \ M_p = \max_{x\in[a,b]} |f^{(p)}(x)|} \qquad (7.10)$$

Dabei ist p durch jeweilige h-Potenz des lokalen Fehlers bestimmt, d. h. der globale Fehler ist um eine Ordnung niedriger als der lokale Fehler.

Bemerkung: Die praktische Anwendung von (7.10) zur Bestimmung des möglichen Maximalfehlers scheitert häufig an der Berechnung oder Abschätzung des Maximums M_p der p-ten Ableitung des Integranden. In diesen Fällen wird in der Regel eine Schätzung des Fehlers auf der Grundlage des **Vergleichs verschiedener Formeln** (siehe die Bemerkung zur Trapezregel und Mittelpunktregel) oder des **Vergleichs von Werten der gleichen Formel zu verschiedenen Schrittweiten** vorgenommen. Für die Simpson-Regel erhält man z. B. wegen der Beziehung $E = O(h^4)$ für den globalen Fehler aus dem Vergleich der Werte $I_{SR}(h)$ und $I_{SR}(2h)$ der Regel zur Schrittweite h und 2h die Beziehung

$$\left|I - I_{SR}(h)\right| \approx \frac{1}{15}\left|I_{SR}(h) - I_{SR}(2h)\right| \qquad (7.11)$$

zur Schätzung des Fehlers.

7.4. Trapezsummenextrapolation, Romberg-Verfahren

In 7.1. war zur näherungsweisen Berechnung des Integrals $I=\int_a^b f(x)dx$ die Trapezsumme benutzt worden:

$$T(h)=h\left[\frac{f(a)}{2}+f(a+h)+f(a+2h)+\cdots+f(b-h)+\frac{f(b)}{2}\right] \quad . \qquad (7.12)$$

Diese kann als Funktion der Schrittweite $h = \frac{b-a}{N}$ aufgefasst werden. Für $h \to 0$ wird T(h) das Integral I darstellen; allerdings scheitert das praktisch daran, dass in T(h) die Anzahl der Summanden unbeschränkt anwächst und T(h) dann durch Rundungsfehler stark verfälscht wird.

Extrapolation zur Bestimmung von $\lim_{h\to 0} T(h)$:

Wir wenden die Methode der Extrapolation aus 6.4 an.

- T(h) wird für verschiedene Schrittweiten h_0 , h_1 , $\cdots$, h_n berechnet.

- Zu den Wertpaaren $(h_0,T(h_o)),\cdots,(h_n,T(h_n))$ wird ein Interpolationspolynom P(h) aufgestellt.

- Der Wert P(0) des Polynoms stellt eine Näherung für den gesuchten Wert $\lim_{h\to 0} T(h)$ dar.

Der Erfolg der Methode wird dadurch begründet, dass T(h) eine asymptotische Entwicklung in Abhängigkeit von h besitzt:

Satz 7.6.: Für eine in [a, b] (2m + 2)-mal stetig differenzierbare Funktion f(x) gilt die Entwicklung

$$T(h)=I+\tau_1 h^2+\tau_2 h^4\cdots\tau_m h^{2m}+\alpha(h)h^{2m+2}$$

Die τ_i sind unabhängig von h und es gibt M > 0 so dass $|\alpha(h)| \le M$ unabhängig von h gilt.

Die Entwicklung wird als Euler-MacLaurinsche Summenformel bezeichnet, die τ_i sind Zahlenwerte, die mit den sogenannten Bernoulli-Zahlen zusammenhängen und allein durch Funktions- und Ableitungswerte von f(x) in den Randpunkten x=a und x=b bestimmt sind.

Beispiel: Berechnet man T(h) zu 2 verschiedenen Schrittweiten h_0, h_1, so gilt

$$T(h_0) = I + \tau_1 h_0^2 + \tau_2 h_0^4 + O(h_0^6) = I + O(h_0^2) \qquad \downarrow -\frac{h_1^2}{h_0^2}$$

$$T(h_1) = I + \tau_1 h_1^2 + \tau_2 h_1^4 + O(h_1^6) = I + O(h_1^2)$$

$$T(h_1) - \frac{h_1^2}{h_0^2} T(h_0) = I\left(1 - \frac{h_1^2}{h_0^2}\right) + \tau_i 0 + \tau_2\left(h_1^4 - h_1^2 h_0^2\right) + O(h^6)$$

wenn h die Größenordnung von h_0, h_1 bezeichnet. Damit gilt:

$$T(h_0,h_1) = \frac{T(h_1) - \frac{h_1^2}{h_0^2} T(h_0)}{1 - \frac{h_1^2}{h_0^2}} = I + 0\tau_1 - \tau_2\left(h_0^2 h_1^2\right) + O(h^6)$$

d. h. der Wert $T(h_0,h_1)$ besitzt die Fehlerordnung $O(h^4)$, die um zwei h-Potenzen besser ist als $T(h_0)$ und $T(h_1)$ selbst. Während man also mit der Trapezmethode durch Halbierung von h den Fehler nur auf $\frac{1}{4}$ reduzieren kann, ist durch Berechnung des sogenannten extrapolierenten Wertes $T(h_0,h_1)$ die Fehlerordnung entscheidend verkleinert worden.

Andere Darstellung für $T(h_0,h_1)$:

$$\boxed{T(h_0,h_1) = T(h_1) - \frac{1}{1 - \frac{h_0^2}{h_1^2}}\left(T(h_1) - T(h_0)\right)} \qquad (7.13)$$

Analog: Für eine (2m + 2) mal stetig differenzierbare Funktion f(x) können durch Kombination von (m + 1) Trapezsummen alle Terme mit $\tau_1, \tau_2 \cdots, \tau_m$ eliminiert werden und man erhält $T(h_0,h_1,\cdots,h_m)$ als Näherung für I(f) mit einer Fehlerordnung $O(h^{2m+2})$.

Wird die Elimination von $\tau_1, \cdots, \tau_m$ wie oben auf der Basis des Gauß-Algorithmus durchgeführt, so spricht man von der Richardson-Extrapolation. Numerisch günstiger ist das Romberg-Verfahren, welches auf der Verwendung eines Interpolationspolynoms und des Aitken-Neville-Schemas beruht.

Romberg-Verfahren

1. Zu einer Folge von Schrittweiten
   $$h_0, \quad h_1 = \frac{h_0}{n_1}, \quad h_2 = \frac{h_0}{n_2}, \quad \cdots, \quad h_m = \frac{h_0}{n_m}$$
   werden die Trapezsummen berechnet $T(h_i) \quad i = 0,1,\cdots,m$
2. Es wird ein Interpolationspolynom $\tilde{P}(h)$ bestimmt
   $\tilde{P}(h) = a_0 + a_1 h^2 + a_2 h^4 + \cdots + a_m h^{2m}$ bzw.
   $P(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_m x^m \left(x = h^2 \text{ wegen Satz 7.6.}\right)$ mit $\tilde{P}(h_i) = T(h_i)$.
3. Der extrapolierte Wert $P(0) = a_0$ ist ein verbesserter Näherungswert für I.

Praktische Durchführung mittels Neville-Schema

Das Polynom $\tilde{P}(h)$ wird nicht explizit berechnet, sondern mit dem Neville-Schema nur der Wert des Polynoms an der Stelle h = 0.

Schema von Aitken-Neville: $x_o = h_0^2, \cdots, x_n = h_n^2$ Stützstellen, $\bar{x}$ Auswertungsstelle des Polynoms. Nach Satz 6.4., Formel (6.5), gilt

$$P_j^{(k)}(\bar{x}) = P_j^{(k-1)}(\bar{x}) + \frac{\bar{x} - x_j}{x_j - x_{j-k}} \left(P_j^{(k-1)}(\bar{x}) - P_{j-1}^{(k-1)}(\bar{x}) \right)$$

Wir ersetzen x durch h^2 und wollen auf $\bar{x} = \bar{h}^2 = 0$ extrapolieren. Die Startwerte des Schemas sind

$$P_j^{(0)} = T(h_j) \quad j = 0,1,\cdots,m .$$

Dann gilt wegen $x_j = h_j^2, \quad x_{j-k} = h_{j-k}^2$

$$P_j^{(k)} := P_j^{(k-1)} - \frac{h_j^2}{h_j^2 - h_{j-k}^2} \left(P_{j-1}^{(k-1)} - P_{j-1}^{(k-1)} \right) = P_j^{(k-1)} - \frac{1}{1 - \frac{h_{j-k}^2}{h_j^2}} \left(P_j^{(k-1)} - P_{j-1}^{(k-1)} \right) \quad k = 1, \cdots, j \qquad (7.14)$$

Setzt man j = 1, k = 1 und $P_1^{(0)} = T(h_1), P_0^{(0)} = T(h_0)$ so stimmt $P_1^{(1)}$ mit $T(h_0, h_1)$ in (7.13) überein.

Berechnung der Startwerte:

Schrittweitewahl: $h_0 = b - a$, $h_j = \frac{h_0}{n_j} \qquad j = 1, 2, 3, \cdots$

(a) $n_j = 2, 4, 8, 16, 32, \ldots$ Romberg - Folge

(b) $n_j = 2, 3, 4, 6, 8, 12, 16, \ldots$ Bulirsch – Folge $(2^{k-1}, bzw. 3 * 2^k .)$

Vorteil der Bulirsch-Folge: Die h_i gehen nicht zu stark gegen Null, der zusätzliche Aufwand zur Berechnung von $T(h_i)$ besteht im wesentlichen in der Berechnung nur weniger neuer Funktionswerte.

Berechnung der Trapezsumme: Nur die neu auftretenden Funktionswerte müssen berechnet werden. Für die Romberg-Folge: $n_j = 2n_{j-1} \quad bzw. \; h_j = \frac{h_{j-1}}{2}$ erhält man

$$T(h_j) = T\left(\frac{1}{2} h_{j-1}\right) = \frac{1}{2} h_{j-1} \left[\frac{f(a)}{2} + f(a + h_j) + f + (a + 2h_j) + f(a + 3h_j) + \cdots + f(b - h_j) + \frac{f(b)}{2} \right]$$

$$= \frac{1}{2} T(h_{j-i}) + h_j \left[f(a + h_j) + f(a + 3h_j) + \cdots + f(b - h_j) \right]$$

(neu zu berechnende Funktionswerte)

Abbruch des Verfahrens:

Statt Vorgabe einer festen Anzahl m von Extrapolationsschritten kann eine relative Genauigkeit *tol* für das Integral vorgegeben werden. Es wird abgebrochen, wenn die

Differenz $\left|P_j^{(j)}-P_{j-1}^{(j-1)}\right|$ kleiner als $tol\cdot\tilde{I}$ wird. Die Größe $P_j^{(j)}$ wird dann als Integralnäherung akzeptiert. $\tilde{I}$ ist dabei eine obere Grenze für $|I|$ (kann z. B. bei der Berechnung der Startwerte $P_j^{(0)}=T(h_j)$ mitgewonnen werden, indem alle Funktionswerte positiv genommen werden, d. h.

$$\tilde{I}\approx\int_a^b|f(x)|dx)$$

Solange die Abbruchbedingungen nicht erfüllt sind, kann das Schema um eine weitere Zeile erweitert werden, d. h. es ist ein neuer Startwert $P_{j+1}^{(0)}=T(h_{j+1})$ zu bestimmen.

Algorithmus:

$h_0 = b - a,$

j = 0, 1, 2, 3, ...

$$h_j = \frac{h_0}{n_j},\ P_j^{(0)}=T(h_j)$$

k = 1, 2, ... , j

$$P_j^{(k)}:=P_j^{(k-1)}-\frac{1}{1-(h_{j-k}/h_j)^2}\left(P_j^{(k-1)}-P_{j-1}^{(k-1)}\right)$$

Abbruch, wenn $\left|P_j^{(j)}-P_{j-1}^{(j-1)}\right| < tol * \tilde{I}$

Speziell für die Romberg-Folge gilt

$$\left(\frac{h_{j-k}}{h_j}\right)^2=\left(\frac{h_0/2^{j-k}}{h_0/2^j}\right)^2=\frac{4^j}{4^{j-k}}=4^k \quad \text{d.h.} \quad P_j^{(k)}:=P_j^{(k-1)}+\frac{1}{4^k-1}\left(P_j^{(k-1)}-P_{j-1}^{(k-1)}\right)$$

Konvergenz und Fehlerabschätzung:

Satz 7.7.: Ist $f(x)\in C^{2m+2}[a,b]$ und $T(h_i)i=0,\cdots,m$ sind die Trapezsummen. Dann gilt für die mit Hilfe des Schemas erzeugten Werte.

$$\left|P_m^m-\int_a^b f(x)dx\right|=O\left(h_0^2h_1^2\cdots h_m^2\right)\ .$$

Für genügend glatte Integranden $\left(f\in C^{2m+2},\text{m groß}\right)$ kann superlineare Konvergenz nachgewiesen werden, d. h. entlang der Diagonalen des Tableaus wird die Konvergenz beobachtet

$$\lim_{j\to\infty}\frac{P_j^{(j)}-I}{P_{j-1}^{(j-1)}-I}=0\ .$$

Vorsicht bei nicht genügend glatten Integranden: z. B. $\int_0^1 x^{3/2}dx$

Hier bringt nur die erste Spalte für k = 1 Vorteile wegen der Unstetigkeit der 2. Ableitung in x = 0.

Transformationen:

Bei Singularitäten des Integranden bzw. der Ableitungen sollte das Integral transformiert werden (Substitutionsregel) um reguläre Integranden zu erhalten. Mit der Substitution $x = t^2$ erhält man z.B. $\int_0^1 x^{3/2}\,dx = 2\int_0^1 t^4\,dt$.

7.5. Gauss-Legende-Integration

Die allgemeine Form einer Integrationsregel ist nach (7.5)

$$I = \int_a^b f(x)dx = \sum_{i=0}^{n} c_i\, f(x_i) + E_n(f)\,. \qquad (7.15)$$

Bisher hatten wir die (n+1) Stützstellen x_i äquidistant vorgegeben. Die Formeln waren exakt für Polynome vom Grad n+1 bzw. n (bei ungeradem n).

Jetzt werden die 2n+2 Größen x_i, c_i (i = 0, , n) in (7.15) als frei betrachtet und sollen so bestimmt werden, dass die Formeln exakt sind für Polynome möglichst hohen Grades.

Vermutung: Es sind 2n+2 Größen frei; d. h. es können Polynome mit maximal 2n+2 Koeff. exakt integriert werden. Die Polynome sind somit maximal vom Grad 2n+1.

Beispiel: Dass ein Polynom vom Grad 2n+2 nicht mehr exakt integriert werden kann, zeigt die Funktion

$$f(x) = P_{2n+2}(x) = (x-x_0)^2 (x-x_1)^2 \ldots (x-x_n)^2 \;.$$

Es gilt für b > a

$$I = \int_a^b f(x)dx > 0\,.$$

Wegen $f(x_j) = 0$ gilt aber $\sum_{i=0}^{n} c_i\, f(x_i) = 0$, d. h. Formel (7.15) kann für f(x) nicht exakt sein.

Quadraturformeln auf der Basis von (n+1) Stützstellen, mit denen Polynome vom Grad 2n+1 exakt integriert werden, heißen Gauss-Legende-Formeln.

Basisintervall [-1, 1]: Wir betrachten zunächst eine Integration über dem Intervall [-1, 1] :

Satz 7.8: Es gibt genau eine Quadraturformel

$$I = \sum_{i=0}^{n} c_i\, f(x_i) + E_n(f), \quad x_i \in [-1,1]$$

mit n+1 Stützstellen, die exakt ist vom Grad 2n+1. Die Stützstellen x_i sind dann die Nullstellen des Legendre-Polynoms P_{n+1} und die Integrationsgewichte sind

$$c_i = \int_{-1}^{1} \prod_{\substack{j=0 \\ j\neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right)^2 dx \quad i = 0,1,\ldots,n \;.$$

Beweis: H.R. Schwarz S. 360/361.

Bemerkung: Die Legendre-Polynome sind spezielle Orthogonalpolynome und genügen der Beziehung $P_n(x) = \frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n$.

Charakteristisch:

- Die Nullstellen x_i liegen symmetrisch zu x = 0;
- Gewichte $c_i > 0$;
- Gewichte zu einer positiven Nullstelle x_i und der entsprechenden negativen NST x_j sind gleich, d. h. $c_i = c_j$ wenn $x_i = -x_j$;
- Um mit Gauss-Formeln arbeiten zu können, müssen die x_i, c_i (i = 0, ..., n) berechnet (tabelliert) werden.

Berechnung der x_i, c_i :

Aufgrund des Zusammenhangs des Legendre Polynoms P_{n+1} (x) mit der Determinante einer 3-Bandmatrix (n+1)ter Ordnung können die NST x_i als Eigenwerte der symmetrischen Tridiagonalmatrix J_{n+1} stabil berechnet werden.

$$J_{n+1} = \begin{pmatrix} 0 & \beta_1 & & & & \\ \beta_1 & 0 & \beta_2 & & & \\ & \beta_2 & 0 & & & \\ & & & 0 & \beta_{n-1} & \\ & & & \beta_{n-1} & 0 & \beta_n \\ & & & & \beta_n & 0 \end{pmatrix} \qquad \beta_i = \frac{i}{\sqrt{4i^2-1}} \qquad i = 1, \cdots, n. \qquad (7.16)$$

Die Gewichte c_i sind durch die ersten Komponenten v_1^i der entsprechenden normierten Eigenvektoren bestimmt:

$$c_i = 2\left(v_1^i\right)^2 \quad .$$

Allgemeines Intervall [a, b]: Ist f(t) für t ∈ [a, b] zu integrieren, so kann die Variablentransformation verwendet werden:

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

Es gilt

$$I = \int_a^b f(t)dt = \frac{b-a}{2}\int_{-1}^{1} g(x)dx \qquad \text{mit} \quad g(x) = f\left(\frac{b-a}{2}x + \frac{a+b}{2}\right) .$$

Damit folgt

$$\boxed{\begin{aligned} I &= \frac{b-a}{2}\sum_{i=0}^{n} c_i\, g(x_i) + E_n = \frac{b-a}{2}\sum_{i=o}^{n} c_i\, f(t_1) + E_n \\ t_i &= \frac{b-a}{2}x_1 + \frac{a+b}{2} \qquad i = 0, \cdots, n \end{aligned}} \qquad (7.17)$$

Dabei sind x_i, c_i die oben berechneten Werte.

Vorteile:

- Hohe Genauigkeit bei wenig Funktionswertberechnungen des Integranden
- Die Folge der Gauss-Legendre-Formeln ist konvergent für $n \to \infty$.

Nachteile:

- Bei vorgegebener Genauigkeit ist es schwer a-priori ein geeignetes n zu finden, so dass $|E_n| <$ eps ist. Die Formeln werden solange angewendet, bis die Abweichung aufeinanderfolgende Werte kleiner als eps ist, bzw. es wird das Gesamtintervall solange in Teilintervalle zerlegt, bis die geforderte Genauigkeit erreicht wird.
- Bereits berechnete Funktionswerte können nicht bei größerem n verwendet werden, da die x_i sich unterscheiden (Verbesserung durch Kronrod-Formeln).
- Die Formeln der Gauss- Integration sind offene Formeln. Ist der Einsatz geschlossener Formeln erforderlich (z. B. bei der Lösung von Integralgleichungen), dann ist ein Übergang zu Radan-Formeln bzw. Lobatto-Formeln nötig. Die Integrationsformeln sind dann nur noch exakt vom Grad 2n bzw. 2n-1.

8. Numerisches Differenzieren

Geschlossene Lösungen von Differenzialgleichungen können in der Praxis selten berechnet werden, insbesondere bei nichtlinearen Systemen von Differenzialgleichungen. Darum ist es erforderlich, numerische Näherungslösungen zu bestimmen. In verschiedenen Diskretisierungsverfahren sind Ableitungen durch geeignete Approximationen zu ersetzen.

8.1. Finite Differenzen als Näherungen für Ableitungen

Numerische Approximationen für Ableitungen enthalten 2 Fehlerarten:
(a) Diskretisierungsfehler
(b) Rundungsfehler.

Wir werden uns hier vor allem mit dem Diskretisierungsfehler beschäftigen: Zunächst wird gezeigt, wie man systematisch Diskretisierungen von Ableitungen erhält.

Vor: I = [a, b] sei ein Intervall, f(x) eine Funktion über I und f(x) sei m+1-mal stetig differenzierbar. Weiter seien $x_1, \ldots, x_n$ gewisse Stützstellen in I und $x = \bar{x}$ ein beliebiger Punkt aus I.

Ziel: $f^{(k)}(\bar{x})$ näherungsweise bestimmen

Methode: Es wird das zu $x_1, \ldots, x_n$ gehörende Interpolationspolynom P(x) gebildet und $f^{(k)}(\bar{x})$ durch $P^{(k)}(\bar{x})$ ersetzt.

Bezeichnungen: m+1....Grad der Differenzierbarkeit von f(x)
n.....Zahl der verwendeten Stützstellen für die Ableitungsapproximation
k.....Grad der zu approximierenden Ableitung
Bedingung: $m \geq n > k$

Wir setzen $h_i = x_i - \bar{x} \quad i = 1, \cdots, n$ und entwickeln $f(x) \text{ in } x = \bar{x}$ in einer Taylorentwicklung

$$f(\bar{x}+h_j) = f(x_i) = f(\bar{x}) + h_i f'(\bar{x}) + \cdots + \frac{h_i^m}{m!} f^{(m)}(\bar{x}) + \frac{h_i^{m+1}}{(m+1)!} f^{(m+1)}(\bar{x}+\vartheta_i h_i)$$

$$i = 1, \ldots, n \quad (0 < \vartheta_i < 1).$$

Diese n Beziehungen multiplizieren wir mit Gewichten α_i und summieren über i, setzen $f(x_i) = f_i$ zur Abkürzung:

$$\sum_{i=1}^{n} \alpha_i f_i = \left(\sum_{i=1}^{n} \alpha_i \right) f(\bar{x}) + \left(\sum_{i=1}^{n} \alpha_i h_i \right) f'(\bar{x}) + \cdots + \frac{1}{m!} \left(\sum_{i=1}^{n} \alpha_i h_i^m \right) f^{(m)}(\bar{x}) +$$

$$+ \frac{1}{(m+1)!} \left(\sum_{i=1}^{n} \alpha_i h_i^{m+1} f^{(m+1)}(\bar{x}+\vartheta_i h_i) \right). \qquad (8.1)$$

Die α_i sind so bestimmen, dass rechts in (8.1) die ersten n Summanden wegfallen bis auf den Summanden mit $f^{(k)}(\bar{x})$. Dazu stellen wir die folgenden n Bedingungen.:

$$\sum_{i=1}^{n} \alpha_i = 0, \quad \sum_{i=1}^{n} \alpha_i h_i = 0, \quad \cdots, \quad \frac{1}{k!} \sum_{i=1}^{n} \alpha_i h_i^k = 1, \quad \cdots, \quad \sum_{i=1}^{n} \alpha_i h_i^{n-1} = 0$$

bzw. $$\boxed{\sum_{i=1}^{n} \alpha_i h_i^{\nu} = \nu!\ \delta_{\nu k} \qquad \nu = 0, 1, \cdots, n-1} \tag{8.2}$$

Dabei ist $\delta_{\nu k} = 0$ für $\nu \neq k$, $\delta_{\nu k} = 1$ für $\nu = k$.

(8.2) stellt ein lineares Gleichungssystem für die Gewichte α_i dar. Die Koeffizientenmatrix hat die Form

$$\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ h_i & h_2 & h_3 & \cdots & h_n \\ h_1^2 & h_2^2 & h_3^2 & \cdots & h_n^2 \\ h_1^{n-1} & h_2^{n-1} & h_3^{n-1} & \cdots & h_n^{n-1} \end{pmatrix}, \tag{8.3}$$

Dies ist eine Vandermondesche Matrix deren Determinante ≠ 0 ist, wenn alle h_i verschieden sind, d. h. wegen $h_i = x_i - \bar{x}$ müssen die Stützstellen der Interpolation verschieden sein. Damit hat (8.2) für paarweise verschiedene Stützstellen genau eine Lösung und es gilt

$$\boxed{f^{(k)}(\bar{x}) = \sum_{i=1}^{n} \alpha_i f_i + E_n} \tag{8.4}$$

Approximationsfehler En:

Wegen (8.1) und (8.2) gilt für den Fehler E_n (Diskretisierungsfehler der Ableitung)

$$\begin{aligned} E_n &= f^{(k)}(\bar{x}) - \sum_{i=1}^{n} \alpha_i f_i = \\ &= -\frac{1}{n!}\left(\sum_{i=1}^{n} \alpha_i h_i^n \right) f^{(n)}(\bar{x}) \cdots - \frac{1}{(m+1)!}\left(\sum_{i=1}^{n} \alpha_i h_i^{m+1} f^{(m+1)}(\bar{x} + \vartheta_j h_j) \right) \end{aligned} \tag{8.5}$$

d.h. E_n ist klein, wenn die h_i klein sind.
Darüber hinaus gilt $f^{(n)}(\bar{x}) = 0, f^{(m)}(\bar{x}) = 0, f^{(m+1)}(\bar{x} + \vartheta_i h_i) = 0$, wenn f(x) selbst ein Polynom vom Grad kleiner als n ist, d. h. die Approximation (8.4) ist dann exakt.

Satz 8.1.: Ist f(x) ein Polynom vom Grad kleiner als n und $\bar{x} \in I$ ein Wert zwischen den n Stützstellen $x_1, \cdots, x_n$, dann werden sämtliche Ableitungen von f(x) in $\bar{x}$ bis zur Ordnung k = n-1 durch (8.4) (mit $E_n = 0$) exakt dargestellt, wenn die α_i gemäß (8.2) bestimmt werden.

Konkrete finite Differenzenformeln:

1.) $m \geq n = 2; \quad k = 1$ (1. Ableitung zu approximieren)
$\bar{x} = x_1, \quad h_2 = x_2 - \bar{x} = h > 0, \quad h_1 = x_1 - \bar{x} = 0:$

Aus (2) folgt $\alpha_1 = \frac{1}{h}, \quad \alpha_2 = -\frac{1}{h}$, d. h.

$$\boxed{f'(\bar{x}) = \frac{1}{h}\left(f(\bar{x}+h) - f(\bar{x}) \right) + 0(h)} \tag{8.6}$$

(vorwärts genommener Differenzenquotient)

2.) $m \geq 3, n = 2 \quad k = 1$ (1. Ableitungen zu approximieren)

$\bar{x} = \frac{1}{2}(x_1 + x_2) \quad h_1 = x_1 - \bar{x} = -h \quad x_1 = \bar{x} - h$

$h_2 = x_2 - \bar{x} = h > 0 \quad x_2 = \bar{x} + h$

Aus (8.2) folgt $\alpha_1 = \frac{1}{2h}, \quad \alpha_2 = -\frac{1}{2h}$, d. h.

$$\boxed{f'(\bar{x}) = \frac{1}{2h}[f(\bar{x}+h) - f(\bar{x}-h)] + 0(h^2)} \qquad (8.7)$$

(zentraler Differenzenquotient)

3.) m = 4, n = 3, k = 2 (2. Ableitung zu approximieren)

$\bar{x} = x_2, \quad h_1 = x_1 - \bar{x} = -h, \quad h_2 = 0, \quad h_3 = x_3 - \bar{x} = h$

Aus (2) folgt $\alpha_3 = \alpha_1 = \frac{1}{h^2} \quad , \quad \alpha_2 = -\frac{2}{h^2}$ d. h.

$$\boxed{f''(\bar{x}) = \frac{1}{h_2}[f(x_1) - 2f(x_2) + f(x_3)] + 0(h^2)} \qquad (8.8)$$

Bemerkung: Zur Bildung von Differenzenquotienten für die k-te Ableitung benötigt man mindestens k+1 Interpolationspunkte. Die Genauigkeit der Approximation ist um so besser je zentraler der Punkt $\bar{x}$ bezüglich der Stützstellen liegt, daher werden in der Praxis zentrale Differenzenquotienten bevorzugt.

Fehlerabschätzung: Man kann mit Hilfe der Formel für den Fehler der Polynominterpolation (Satz 6.7) nachweisen, dass gilt:

Satz 8.2.: Sei f(x) n-mal stetig differenzierbar und $P_{n-1}(x)$ das zu $x_1, \cdots, x_n$ gehörende Interpolationspolynom, $n > k \geq 0$. Dann gilt für beliebige $\bar{x} \in [x_1, \cdots, x_n]$.

$$\left| f^{(k)}(\bar{x}) - P_{n-1}^{(k)}(\bar{x}) \right| \leq \frac{h^{n-k}}{(n-k)!} \max_{x \in [x_1, x_n]} \left| f^{(n)}(x) \right| \qquad \text{mit } h = \max_{i=1,..n} |x_j - \bar{x}|$$

8.2. Extrapolation für Ableitungen

Ziel: Verbesserung der Genauigkeit von Differenzenapproximationen für Ableitungen; Erreichen einer höheren Fehlerordnung.

Methode: Extrapolation auf h = 0:

Der Differenzenquotient $D_k(h)$ als Näherung für $f^{(k)}(\bar{x})$ wird für verschiedene h-Werte berechnet:

$$d_0 = D_k(h_0), d_1 = D_k(h_1), \cdots, d_n = D_k(h_n)$$

und für die Wertepaare $(h_i, d_i) \quad i = 0, \cdots, n$ wird ein Interpolationspolynom n-ten Grades aufgestellt in Abhängigkeit von h: $d = P_n(h)$. Der Wert $P_n(0)$ stellt dann i.a. ein bessere Näherung für die Ableitung $f^{(k)}(\bar{x})$ dar als es die d_i sind.

Beispiel: Ableitungsapproximation $f'(x_o)$ auf der Grundlage der vorderen Differenzenquotienten

Unter der Voraussetzung $f \in C^4$ gilt für $\bar{x} = x_o$ die Taylorentwicklung

$$f(x_o+h)=f(x_o)+f'(x_o)h+\frac{f''(x_o)}{2!}h^2+\frac{f'''(x_o)}{3!}h^3+\frac{f^{(4)}(x_o+\vartheta h)}{4!}h^4+o(h^4)$$

und für den vorderen Differenzenquotienten folgt

$$D_1(h)=\frac{f(x_o+h)-f(x_o)}{h}=f'(x_o)+\frac{f''(x_o)}{2!}h+\frac{f'''(x_o)}{3!}h^2+O(h^3).$$

Wir setzen

$x_0 + h = x_1$ d. h. $h_1 = h$

$x_0 + \frac{h}{2} = x_2$ d. h. $h_2 = h/2$

$x_0 + \frac{h}{4} = x_3$ d. h. $h_3 = h/4$

und erhalten

$$D_1(h_1)=\frac{f(x_1)-f(x_o)}{h_1}=f'(x_o)+\frac{f''(x_o)}{2!}h+\frac{f'''(x_o)}{3!}h^2+O(h^3) \qquad \downarrow -\frac{1}{2}$$

$$D_1(h_2)=\frac{f(x_2)-f(x_o)}{h_2}=f'(x_o)+\frac{f''(x_o)}{2!}\frac{h}{2}+\frac{f'''(x_o)}{3!}\frac{h^2}{4}+O\left(\frac{h^3}{8}\right) \qquad \downarrow -\frac{1}{2}$$

$$D_1(h_3)=\frac{f(x_3)-f(x_o)}{h_3}=f'(x_o)+\frac{f''(x_o)}{2!}\frac{h}{4}+\frac{f'''(x_o)}{3!}\frac{h^2}{16}+O\left(\frac{h_3}{64}\right)$$

Durch Subtraktion des $\left(-\frac{1}{2}\right)$ fachen der 1. von der 2. Beziehung bzw. der 2. von der 3.Beziehung und Division durch $\left(1-1/2\right)$ erhalten wir extrapolierte Diffenenzenquotienten 1.Stufe.

$$D_1(h_1,h_2)=\frac{D_1(h_2)-\frac{1}{2}D_1(h_h)}{1-\frac{1}{2}}=f'(x_0)+\frac{f'''(x_0)}{3!}\cdot\left(-\frac{1}{2}h^2\right)+O(h^3) \qquad \downarrow -\frac{1}{4}$$

$$D_1(h_2,h_3)=\frac{D_1(h_3)-\frac{1}{2}D_1(h_2)}{1-\frac{1}{2}}=f'(x_0)+\frac{f'''(x_0)}{3!}\left(-\frac{1}{8}h^2\right)+O(h^3)$$

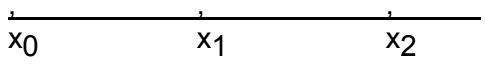
Daraus erhält man einen Differenzenquotienten 2. Extrapolationsstufe

$$D_1(h_1,h_2,h_3)=\frac{D_1(h_2,h_3)-\frac{1}{4}D_1(h_1,h_2)}{1-\frac{1}{4}}=f'(\bar{x})+O(h^3).$$

Während $D_1(h)$ die Fehlerordnung $O(h)$ besitzt , haben die extrapolierten Quotienten 1. Stufe die Fehlerordnung $O(h^2)$ und der Quotient 2. Stufe die Ordnung $O(h^3)$.

Bemerkungen: 1.Die h_i müssen nicht notwendig durch fortlaufende Halbierung einer Schrittweite h entstehen.
2.Verbesserungen der Fehlerordnung um 2 in jeder Extrapolationsstufe erhält man unter Verwendung des zentralen Differenzenquotienten.

Mann kann auf diese Weise auch explizite Formeln für Differenzenquotienten höherer Fehlerordnung entwickeln:



$$\begin{aligned} x_0 + h &= x_1 \qquad h_1 = h \\ x_0 + 2h &= x_2 \qquad h_2 = 2h \end{aligned} \quad \Rightarrow q = \frac{h_2}{h_1} = 2$$

Unter Verwendung des vorderen Differenzenquotienten für die Approximation der 1. Ableitung folgt dann

$$D_1(h,2h) = \frac{D_1(2h) - 2D_1(h)}{1-2} = \frac{\frac{f(x_2)-f(x_0)}{2h} - 2\frac{f(x_1)-f(x_0)}{h}}{-1}$$

$$= \frac{-(f_2-f_0)+4(f_1-f_0)}{2h} = \frac{-3f_0+4f_1-f_2}{2h}$$

(vorderer Differenzenquotient der Fehlerordnung $O(h^2)$).

Praktische Durchführung der Extrapolation:

Es bietet sich wieder das Neville-Schema an, wobei x durch h zu ersetzen ist und der Wert des Interpolationspolynoms an der Stelle $\bar{h} = o$ zu berechnen ist.

Damit sind die Werte $D_k(h_i) = P_i^{(0)}$ Startwerte des Ausgangstableaus und es ist die Rekursion durchzuführen

$$P_j^{(k)} = P_j^{(k-1)} + \frac{O - h_k}{h_k - h_j}\left[P_j^{(k-1)} - P_{j-1}^{(k-1)}\right] \quad j = 0,1,2,\ldots; \quad k = 1,..j.$$

8.3. Weitere Möglichkeiten zur Approximation von Ableitungen

Aufgrund der Eigenschaften von Splines, können auch diese zur Approximation von Ableitungen verwendet werden. So liefern die Momente y_i'' aus dem System der Grundgleichungen (6.34) mit den Zusatzgleichungen entsprechend der gewählten Randbedingungen zur Berechnung kubischer Splines Approximationen von zweiten Ableitungen in den Stützstellen der Splineinterpolation. Bezüglich der Güte der Approximation gilt Satz 6.11. bei Hermite-Randbedingungen bzw. der Berechnung einer „not-a-knot“ Spline.

9. Numerische Lösung von Anfangswertproblemen gewöhnlicher Differenzialgleichungen

Zielstellung: In diesem Kapitel entwickeln wir Verfahren zur numerischen Lösung von Anfangsproblemen des Typs:

$$\left.\begin{array}{l} \dot{y}_i = f_i(t, y_1, y_2, \cdots, y_n) \\ y_i(a) = y_i^0, \quad a \le t \le b \end{array}\right\} \quad i = 1,2,\cdots,n. \tag{9.1}$$

Da sowohl Differenzialgleichungen höherer Ordnung als auch Systeme von DGL höherer Ordnung in ein Kanonisches System 1. Ordnung überführt werden können, ist durch (9.1) tatsächlich eine sehr allgemeine Problemstellung gegeben. Bedingungen für die Existenz einer eindeutigen Lösung von (9.1) sind durch den Existenz- und Eindeutigkeitssatz für AWP gegeben, welcher im wesentlichen die Lipschitzstetigkeit der rechten Seite f bezüglich der y-Variablen fordert.

9.1. Einschrittverfahren (ESV)

Problemstellung:

$$\begin{array}{l} \dot{y} = f(t, y) \quad a \le t \le b \\ y(a) = y_0 \end{array} \tag{9.2}$$

Lösungsmethode: Das Intervall [a, b], in welchem die Lösungsfinktion y(t) gesucht wird, zerlegt man in N Teilintervalle

$$a = t_0 < t_1 < t_2 < \cdots < t_N = b .$$

Die numerische Lösung von (9.2) besteht darin, in den Stützstellen t_k Näherungswerte y_k für den Funktionswert $y(t_k)$ der Lösungsfunktion zu bestimmen. Dazu ist die Differenzialgleichung in geeigneter Weise zu diskretisieren.

9.1.1. Das Eulersche Polygonzugverfahren

Aus der Taylorentwicklung der Lösungsfunktion $y(t)$ im Punkt $t = t_k$ folgt

$$y(t) = y(t_k) + (t - t_k)\dot{y}(t_k) + \frac{(t - t_k)^2}{2!}\ddot{y}(z_k), \quad z_k \in (t, t_k) .$$

Setzt man $t = t_{k+1} = t_k + h$ und beachtet die Gültigkeit von $\dot{y}(t_k) = f(t_k, y(t_k))$, so erhält man unter Verwendung der Bezeichnungen y_k, y_{k+1} für die Näherungswerte von $y(t_k), y(t_{k+1})$ das numerische Schema

$$\boxed{\begin{array}{ll} y_{k+1} := y_k + h f(t_k, y_k) & k = 0,1,\cdots,N-1 \\ y_0 = y(a) & \end{array}} \quad .$$

Dabei wird eine äquidistante Unterteilung von $[a, b]$ vorausgesetzt, d. h. es gilt

$$t_k = a + k * h, \quad h = \frac{b - a}{N} \quad .$$

Geometrisch: Die Lösungsfunktion y(t) wird im Intervall $\left[t_k\,,\,t_{k+1}\right]$ durch die Tangente in t_k ersetzt

Aufwand: Das Verfahren ist einfach zu programmieren, pro Schritt ist eine Auswertung der rechten Seite f(t, y) der DGL erforderlich.

Fehlerordnung des Verfahrens:

Es spielen 2 Fehlerarten eine entscheidende Rolle:

(a) Diskretisierungsfehler, die durch die Ersetzung von Ableitungen durch finite Ausdrücke entstehen;
(b) Rundungsfehler bei der Auswertung der arithmetischen Ausdrücke.

Wir betrachten zunächst nur die erste Fehlerart, da dieser bei Verwendung einer geeigneten Gleipunktarithmetik i. a. den Hauptteil des Fehlers repräsentiert.

Def. 9.1: Die Größe

$$E(h)=\max_{1\le k\le N}\left|y_k-y(t_k)\right| \tag{9.4}$$

bezeichnet den globalen Diskretisierungsfehler. Mit

$$L(t,h)=\frac{1}{h}\left(y(t+h)-y(t)\right)-f\left(t,y(t)\right) \tag{9.5}$$

bezeichnen wir den lokalen Diskretisierungsfehler.

Der globale Diskretisierungsfehler misst die Abweichung der Näherungswerte von den exakten Lösungswerten. Wir erwarten, daß E(h) $\to$ 0 strebt für h $\to$ 0, d. h. für feinere Unterteilung von [a, b].

Der lokale Fehler misst dagegen die Abweichung von $f\left(t,y(t)\right)$ und dem Differenzenquotienten $\left(y(t+h)\right)-y(t)/\,h$.

Wir setzen $$L(h)=\max_{a\le t\le b-h}\left|L(t,h)\right| \quad , \tag{9.6}$$

dann folgt aus der Taylorentwicklung von y(t)

$$y(t+h)-y(t)=h\dot{y}(t)+\frac{h^2}{2}\ddot{y}(z),\quad z\in(t,t+h)\ .$$

Ist die zweite Ableitung der Lösungsfunktion beschränkt, d.h.

$$\left|\ddot{y}(t)\right|\le M\quad \forall t\in[a,b],$$

so hält man wegen $\dot{y}=f(t,y(t))$

$$\boxed{|L(h)| \le \frac{h}{2} M = O(h).} \tag{9.7}$$

Def.9.2: Eine Funktion g(h) heißt für $h \to 0$ vom Typ $0(h^p)$, wenn gilt

$$\lim_{h\to 0} \left| \frac{g(h)}{h^p} \right| = K > 0 ,$$

d. h. g(h) verhält sich für $h \to 0$ wie die Funktion Kh^p mit einer positiven Konstanten K.

Verbindung von lokalem und globalem Diskretisierungsfehler:

Mit $e_k = y(t_k) - y_k$ folgt

$$\begin{aligned} e_{k+1} = y(t_{k+1}) - y_{k+1} &= y(t_k) + hf(t_k, y(t_k)) + hL(t_k, h) - \left[y_k + hf(t_k, y_k) \right] \\ &= e_k + h\left[f(t_k, y(t_k)) - f(t_k, y_k) \right] + hL(t_k, h). \end{aligned}$$

Unter der Voraussetzung der Lipschitz-Stetigkeit von f bezüglich y folgt

$$|f(t_k, y(t_k)) - f(t_k, y_k)| \le \gamma \, |y(t_k) - y_k|$$

bzw.

$$|e_{k+1}| \le |e_k| + h\gamma |y(t_k) - y_k| + h|L(h)| \ . \tag{9.8}$$

Wir setzen $c = 1 + h\gamma$ und erhalten durch Fortführung des Prozesses die Abschätzung

$$|e_{k+1}| \le h \, |L(h)| \left(c^k + c^{k+1} + \cdots + c + 1 \right) \ . \tag{9.9}$$

Wegen $c = 1 + h\gamma > 1$ ist c^k monoton wachsend und $k \le N = \frac{b-a}{h}$, d.h. es gilt auch.

$$|e_k| \le h \, |L(h)| \, N \cdot c^N . \quad \forall k$$

c^N bleibt auch für $h \to 0$ (d. h. $N \to \infty$) beschränkt, denn für $h \to 0$ folgt

$$c^N = \left[(1 + \gamma h) \right]^{b-a} \xrightarrow[h\to 0]{} e^{\gamma(b-a)} \ .$$

Damit gilt: $|e_k| \le (b-a) e^{-\gamma(b-a)} |L(h)| = O(h) \quad \forall k$.

Satz 9.3.: Ist f(t, y) Lipschitzstetig bezüglich y und besitzt die Lösung y(t) des AWP eine beschränkte 2. Ableitung, dann gilt für den globalen Diskretisierungsfehler des Euler-Verfahrens

$$E(h) = O(h). \tag{9.10}$$

Die Konvergenz von $E(h) \to 0$ für $h \to 0$ ist für praktische Probleme zu gering, so dass Verfahren mit verbesserter Konvergenz interessieren.

9.1.2. ESV mit höherer Konvergenzordnung

Allgemein kann man explizite ESV durch die Verfahrensvorschrift charakterisieren:

$$\boxed{\begin{aligned} y_{k+1} &= y_k + h\,\Phi(t_k, y_k, h), \quad y_0 = y(a) \\ t_{k+1} &= t_k + h \qquad k = 0,1,\cdots \end{aligned}} \tag{9.11}$$

Lokaler Diskretisierungsfehler:

$$L(t,h) = \frac{1}{h}\big(y(t+h) - y(t)\big) - \Phi(t,y,h)\ . \tag{9.12}$$

Für eine sinnvolles Verfahren ist zu fordern, dass gilt $\lim\limits_{h\to 0} L(t,h) = 0$.

Allgemeiner definieren wir:

Def. 9.4.: Ein Verfahren heißt konsistent, wenn für alle stetig differenzierbaren Funktionen $f(t,y)$ und alle $t \in [a,b],\ y \in R$ gilt

$$\lim_{h\to 0} |L(t,h)| = 0\ .$$

Gilt darüber hinaus für $p \geq 1$

$$L(h) = O(h^p)\ ,$$

dann heißt p die Konsistenzordnung des Verfahrens.

Konsistenz und Konsistenzordnung charakterisieren ein Verfahren bezüglich der erreichbaren Genauigkeit. Neben dem lokalen Diskretisierungsfehler soll für sinnvolle Verfahren auch der globale Diskretisierungsfehler klein sein.

Def. 9.5.: Ein Verfahren heißt konvergent, wenn gilt

$$\lim_{h \to o} E(h) = 0\ .$$

Das Verfahren besitzt die Konvergenzordnung p, wenn gilt

$$E(h) = O(h^p)\ .$$

Def. 9.6.: Ein Verfahren heißt null-stabil (stabil für $h \to 0$), wenn für genügend kleine h eine kleine Störung des Verfahrens nur kleine Störungen der Näherungswerte bewirkt.

Die Null-Stabilität verbindet Konsistenz und Konvergenz, es gilt die Aussage:

Satz 9.7.: Ein konsistentes Verfahren der Ordnung p, welche null-stabil ist, ist auch konvergent von der Ordnung p.

Ein einfaches Kriterium, welches die Null-Stabilität des Verfahrens sichert, ist die Lipschitz-Stetigkeit der Verfahrensfunktion $\Phi(t,y,h)$ bezüglich y, d. h.

$$|\Phi(t,y_1,h) - \Phi(t,y_2,h)| \leq L\,|y_1 - y_2| \qquad \begin{matrix} \forall t \in [a,b],\ h>0 \\ \forall y_1, y_2 \end{matrix} \tag{9.13}$$

Bei Gültigkeit dieser Bedingung kann eine Abschätzung des globalen Diskretisierungsfehler vorgenommen werden:

Satz 9.8.: Das Einschrittverfahren sei konsistent mit der Ordnung p d. h.

$$|L(h)| \le K \cdot h^p$$

und die Verfahrensfunktion Φ erfülle eine Lipschitzbedingung (9.13) bezüglich y. Dann gilt für jeden Index k ($1 \le k \le N$)

$$|e_k| = |y_k - y(t_k)| \le K h^p \cdot G(t_k - a)$$

mit

$$G(t) = \begin{cases} \frac{1}{L}\left(e^{Lt} - a\right) & \text{für L>0} \\ t & \text{für L=0} \end{cases} .$$

Bem: Die Fehlerabschätzung zeigt, dass der globale Fehler die gleiche Ordnung wie der lokale Fehler besitzt. Die Abschätzung wächst exponentiell mit t >0, sie überschätzt den Fehler häufig erheblich.

9.1.3. Konstruktion von ESV höherer Konsistenzordnung

Zur Konstruktion von ESV höherer Konsistenzordnung wird für die Verfahrensfunktion $\Phi(t, y, h)$ ein geeigneter Ansatz verwendet. Die im Ansatz auftretenden freien Parameter werden so bestimmt, dass der lokale Diskretisierungsfehler mit einer möglichst hohen h-Potenz beginnt.

Beispiel: Verwendet man den Ansatz

$$\Phi(t,y,h) = a_1 f(t,y) + a_2 f\left(t + b_1 h,\ y + b_2 h f(t,y)\right) \tag{9.14}$$

und entwickelt man den lokalen Fehler L(t, h) in einer Taylorentwicklung, so können durch geeignete Wahl der Parameter a_1, a_2, b_1, b_2, die Koeffizienten der niederen h-Potenzen annulliert werden, so dass L(t, h) mit einer hohen h-Potenz beginnt (Konsistenzordnung).

Verfahren von HEUN: Für $a_1 = a_2 = \frac{1}{2}$, $b_1 = b_2 = 1$ erhält man das Verfahren

$$\boxed{\begin{aligned} k_1 &= f(t_k, y_k) \\ k_2 &= f(t_k + h, y_k + h k_1) \\ y_{k+1} &= y_k + h\left(\frac{1}{2}k_1 + \frac{1}{2}k_2\right) \end{aligned}} \tag{9.15}$$

welches als Verfahren von Heun bezeichnet wird. Pro Schritt werden zwei Auswertungen der rechten Seite der DGL nötig (Berechnung der Steigungen k_1, k_2) und es gilt L(t, h) = $O(h^2)$, d. h. es liegt ein Verfahren zweiter Ordnung vor.

Verbessertes Euler-Verfahren: Für $a_1 = 0,\ a_2 = 1,\ b_1 = b_2 = \frac{1}{2}$ erhält man das Verfahren

$$\boxed{\begin{aligned} k_1 &= f(t_k, y_k) \\ k_2 &= f\left(t_k + \frac{h}{2}, y_k + \frac{1}{2} h k_1\right) \\ y_{k+1} &= y_k + h k_2 \end{aligned}} \tag{9.16}$$

welches als verbessertes Euler-Verfahren bezeichnet wird. Dieses kann auch durch Extrapolation zweier Euler-Schritte begründet werden und besitzt ebenfalls die Fehlerordnung L(t, h) = O(h^2),

Anwendung von Extrapolation

Eine Verbesserung der Konsistenzordnung eines Verfahrens kann man wie erwähnt auch mittels Extrapolation erhalten. Wir zeigen dies am Beispiel des Euler-Verfahrens.

Mit den Schrittweiten h bzw. $\frac{h}{2}$ erhält man:

$$h: \quad t_k = a + kh = \bar{t} \qquad y_k^{(h)} = y(\bar{t}) + c_1 h + O(h^2)$$
$$\frac{h}{2}: \quad t_{2k} = a + 2k \cdot \frac{h}{2} = \bar{t} \qquad y_{2k}^{\frac{h}{2}} = y(\bar{t}) + c_1 \frac{h}{2} + O(h^2) \quad .$$

Extrapolationsschritt: $2y_{2k}^{\left(\frac{h}{2}\right)} - y_k^{(h)} = y(\bar{t}) + O(h^2)$

d.h. der Fehlerterm 1. Ordnung wird annulliert.

Die Extrapolation kann in jedem Schritt wie folgt realisiert werden, Ausgangspunkt ist (t_k, y_k):

(a) Berechne $y_{k+1}^{(1)}$ mit Schrittweite h

(b) Führe einen Doppelschritt mit Schrittweite $\frac{h}{2}$ aus: $y_{k+\frac{1}{2}}^{(2)}$, $y_{k+1}^{(2)}$

(c) Extrapoliere $y_{k+1}^{(1)}$ und $y_{k+2}^{(2)}$

$$y_{k+1} = 2_{k+1}^{(2)} - y_{k+1}^{(1)} = = y_k + h f\left(t_k + \frac{h}{2},\ y_k + \frac{h}{2} f(t_k, y_k)\right).$$

Damit erhält man das verbesserte Euler-Verfahren (9.16)

Das Verfahren von Heun wird auch als ein Verfahren vom Runge-Kutta-Typ bezeichnet. Das bekannteste Klassische Verfahren vom Runge-Kutta-Typ ist durch die Verfahrensvorschrift

$$\boxed{\begin{aligned} k_1 &= f(t_k, y_k) \\ k_2 &= f\left(t_k + \frac{h}{2}, y_k + \frac{1}{2} h k_1\right) \\ k_3 &= f\left(t_k + \frac{h}{2}, y_k + \frac{1}{2} h k_2\right) \\ k_4 &= f\left(t_k + \frac{h}{2}, y_k + \frac{1}{2} h k_3\right) \\ y_{k+1} &= y_k + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \end{aligned}} \qquad (9.17)$$

gegeben. Das Verfahren besitzt die Fehlerordnung $O(h^4)$ und benötigt die Berechnung von 4 Steigungen (d. h. 4 Auswertungen der rechten Seite $f(t,y)$ der DGl).

Bemerkung:

1. Extrapolation kann zur Genauigkeitserhöhung jedes anderen Verfahrens eingesetzt werden. Bei Verwendung eines Runge-Kutta-Verfahrens der Ordnung 4 besitzt der extrapolierte Wert

$$y_{k+1}=\frac{1}{15}\left(16y_{k+1}^{\frac{1}{2}}-y_{k+1}^{(h)}\right) \tag{9.18}$$

die Fehlerordnung $O\left(h^5\right)$. Der Aufwand ist allerdings hoch, denn die Berechnung von $y_{k+1}^{\left(\frac{h}{2}\right)}$ erfordert 2 Schritte des Verfahrens mit der Schrittweite $\frac{h}{2}$, d. h. 8 zusätzliche Auswertungen von f(t, y). Der Gewinn von einer Fehlerordnung ist eher als gering einzuschätzen.

2. Die bei der Trapezsummenintegration beobachtete Steigerung der Fehlerordnung um 2 mit jedem Extrapolationsschritt gelingt mit einer Methode von GRAGG. Allerdings ist das Verfahren kein ESV mehr.

9.1.4. Übertragung auf Systeme von DGL

ESV lassen sich in einfacher Weise auf Systeme von DGL übertragen. Dazu schreiben wir das System 1. Ordnung

$$\begin{aligned}
&\dot{y}_1=f_1\left(t,y_1,y_2,\cdots,y_n\right) \quad y_1(a)=y_1^0\\
&\dot{y}_2=f_2\left(t,y_1,y_2,\cdots,y_n\right) \quad y_2(a)=y_2^0 \quad a\le t\le b\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&\dot{y}_n=f_n\left(t,y_1,y_2,\cdots,y_n\right) \quad y_n(a)=y_n^0
\end{aligned}$$

in vektorieller Form:

$$\dot{y}=\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\\ \vdots\\ \dot{y}_n\end{pmatrix}=f(t,y) \quad y(a)=y^0 \quad a\le t\le b.$$

Da die Funktion f(t,y) jetzt eine Vektorfunktion mit n Komponenten ist, sind die Steigungen k_i Vektoren, welche mittels der Funktionswerte von f berechnet werden.

Für das Euler-Verfahren gilt dann

$$\begin{aligned}
&y^{k+1}=y^k+hf\left(t_k,y^k\right) \quad y^0=y(a)\\
&t_{k+1}=t_k+h \qquad k=0,1.....
\end{aligned} \tag{9.19a}$$

bzw. in Komponentenschreibweise

$$\begin{pmatrix}y_1^{k+1}\\ \vdots\\ y_n^{k+1}\end{pmatrix}=\begin{pmatrix}y_1^k\\ \vdots\\ y_n^k\end{pmatrix}+h\begin{pmatrix}f_1\left(t_k,y_1^k,\cdots,y_n^k\right)\\ \vdots\\ f_n\left(t_k,y_1^k,\cdots,y_n^k\right)\end{pmatrix}, \quad y^0=\begin{pmatrix}y_1^0\\ \vdots\\ y_n^0\end{pmatrix} \tag{9.19b}$$

Analog erfolgt die Übertragung für die anderen Verfahren.

9.1.5. Einfluß von Rundungsfehlern

Beim Euler-Verfahren

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

besitzt der Rundungsfehler 2 Bestandteile

(a) Fehler ε bei der Berechnung von f
(b) Fehler η bei der Auswertung der Formel , d. h.
$\tilde{y}_{k+1}=y_k+h[f(t_k,y_k)+\varepsilon]+\eta$

Intuitiv ist klar: Da der globale Diskretisierungsfehler für h $\to$ 0 gegen Null strebt, kommt er bei mehrfacher Verkleinerung von h in die Größenordnung des Rundungsfehlers. Das wird dadurch beschleunigt, dass mit Verkleinerung von h die Zahl N der Schriite und damit der Gesamtrundungsfehler wachsen.

Somit existiert eine optimale Schrittweite h_{0pt}, für welche der Gesamtfehler minimal ist. Diese Schrittweite ist a priori kaum zu bestimmen und i. a. klein. Man versucht in der Regel eine Genauigkeit tol zu erreichen, die deutlich größer als das Rundungsfehlerniveau ist, so dass der Diskretisierungsfehler den Gesamtfehler dominiert und entscheidend für die Wahl der Schrittweite ist.

9.1.6. Runge-Kutta-Verfahren und Schrittweitesteuerung

Runge-Kutta-Verfahren sind Einschrittverfahren der Form

$$y_{k+1}=y_k+h\Phi(t_k,y_k,h) \tag{9.20}$$

mit

$$\Phi \;=\; \sum_{l=1}^{m}\gamma_l k_l \;=\; \gamma_1 k_1+\gamma_2 k_2+\cdots+\gamma_m k_m \tag{9.21}$$

und

$$k_l=f\left(t_k+\alpha_l h,\; y_k+h\sum_{s=1}^{m}\beta_{ls}k_s\right) \qquad l = 1, ..., m. \tag{9.22}$$

Offenbar kann k_1 explizit berechnet werden, wenn $\beta_{l1} = \beta_{l2} = \cdots \beta_{lm} = 0$ sind, analog k_2 wenn $\beta_{l2} = \beta_{l3} = \cdots \beta_{lm} = 0$ sind usw.

Die Verfahren heißen:

- explizit wenn $\beta_{ls} = 0$ für $s \geq l$;

- halbimplizit wenn $\beta_{ls} = 0$ für $s > l$ (k_l nur implizit gegeben durch $k_l=f(t_k+\alpha_l h,y_k+\beta_{lk}k_l)$

- vollimplizit sonst.

m heißt Stufe des Verfahrens: Es gehen
m^2+2m Parameter α_l, γ_l, β_{sl} $(l,s=1,\cdots m)$ ein. Die Runge-Kutta-Parameter kann man übersichtlich in einem Schema anordnen

	Koeff. von	k_1		k_m
Koeff.	α_1	β_{11}	. . . .	β_{1m}
von h	:			
	:			
	α_m	β_{m1}	. . . .	β_{mm}
		γ_1	. . . .	γ_m

Bei expliziten Verfahren erhält man ein Dreieckschema.

Beispiel: Das R -K-Verfahren (9.17) der Ordnung 4 und Stufe m = 4 ist ein explizites Verfahren mit

$$y_{k+1} := y_k + \frac{h}{6}\left(k_1 + 2k_1 + 3k_1 + k_4\right)$$

und dem Koeffizientenschema

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

Schrittweitesteuerung

Insbesondere für nichtlineare DGL ist eine feste Schrittweite h nicht sinnvoll, da die rechte Seite f(t, y) und damit auch die Lösungsfunktion y(t) unterschiedliche Variation besitzen:

- f besitzt starke Variation: kleine Schrittweite wählen
- f besitzt schwache Variation: große Schrittweite wählen.

Eine Schrittweitesteuerung benötigt möglichst realistische Schätzung des Fehlers. Man gibt sich dazu mit einer Schätzung des lokalen Diskretisierungsfehler zufrieden. Eine solche Schätzung kann auf unterschiedlichen Wegen erhalten werden:

1. Weg: Benutzung von 2 Verfahren unterschiedlicher Ordnung zur Berechnung einer Nähung von $y\left(t_{k+1}\right)$, d.h. y_{k+1} und z_{k+1}. Die Fehlersteuerungstoleranz tol sei vorgegeben. Dann wird die Strategie verwendet:

$$\left\|z_{k+1} - y_{k+1}\right\| = \begin{cases} >> tol \rightarrow h & \text{reduzieren und Schritt wiederholen} \\ \approx tol \rightarrow h & \text{beibehalten} \\ << tol \rightarrow h & \text{vergrößern im folgenden Schritt} \end{cases} .$$

Ein Beispiel für diese Vorgehensweise sind die Runge-Kutta-Fehlberg Verfahren. Das Hauptproblem besteht darin, möglichst wenig zusätzlichen Aufwand zu haben, insbesondere bezüglich der Auswertung der rechten Seite der DGL.

Man nutzt dabei aus, dass die RK-Parameter noch Freiheitsgrade zulassen, so dass man 2 Verfahren unterschiedlicher Ordnung konstruieren kann, die im wesentlichen die gleichen k_i - Werte verwenden.

Koppelung bzw. Einbettung von 2 Runge-Kutta-Verfahren

a) Verfahren 5. Stufe, 4. Ordnung
b) Verfahren 6. Stufe, 5. Ordnung

Koeffizientenschema von RKF4/5

		k_1	k_2	k_3	k_4	k_5	k_6
k_1	0						
k_2	$\frac{1}{4}$	$\frac{1}{4}$					
k_3	$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				
k_4	$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			
k_5	1	$\frac{439}{216}$	- 8	$\frac{3680}{513}$	$-\frac{845}{4104}$		
Verfahren 4. Ordnung		$\frac{25}{216}$	0	$\frac{1408}{2565}$	$\frac{2197}{4104}$	$-\frac{1}{5}$	
k_6	$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	
Verfahren 5. Ordnung		$\frac{16}{135}$	0	$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$

Es gilt:

$$y_{k+1} := y_k + h\left(\gamma_1 k_1 + \gamma_2 k_2 + \cdots + \gamma_5 k_5\right),$$
$$z_{k+1} := y_k + h\left(\bar{\gamma}_1 k_1 + \bar{\gamma}_2 k_2 + \cdots + \bar{\gamma}_6 k_6\right),$$
$$err = h\left\{-\frac{1}{360}k_1 + \frac{128}{4275}k_3 + \frac{2197}{75240}k_4 - \frac{1}{50}k_5 - \frac{2}{55}k_6\right\}.$$

Berechnung der Schrittweite

Eine Schätzung des lokalen Diskretisierungsfehler ist durch die Größe

$err := y_{k+1} - z_{k+1}$

gegeben. Vom Benutzer muss eine Größe tol als Grundlage der Schrittweitesteuerung vorgegeben sein. Auf der Grundlage der Größe vor err wird die Schrittweite verdoppelt oder halbiert. Um zu kleine oder zu große Schrittweiten auszuschließen werden Schranken H_{min} bzw. H_{max} gesetzt.

Schrittweitesteuerung: z_{k+1}, y_{k+1}, $nerr := \left\| z_{k+1} - y_{k+1} \right\|$ berechnen.

(a) Wenn nerr > tol dann $h := \frac{h}{2}$, für $h < H_{min}$ Fehlerausgang.
Wiederholung des Schrittes, d. h. Neuberechnung von z_{k+1}, y_{k+1}.

(b) Wenn $\frac{1}{10} tol < nerr \leq tol$ dann Schrittweite h beibehalten, $y_{k+1} := z_{k+1}$.

(c) Wenn $nerr \leq \frac{1}{10} tol$ dann $h := 2 * h$, wenn $h > H_{max}$

$h := H_{max}$., $y_{k+1} := z_{k+1}$.

Kontinuierliche Schrittweitesteuerung: Aus der Fehleranalyse, d. h. der Darstellung von err mit Hilfe der Taylorentwicklung kann eine defizilere Schrittweitesteuerung abgeleitet werden:

$s := \left(\frac{h * tol}{nerr} \right)^{\frac{1}{p}}$ Schrittweitekoeffizient.

$s<1$: Wiederholung des Schrittes mit $h := h * \max\left\{\frac{1}{2}; s\right\}$

$s \geq 1$: Schritt akzeptiert, neue Schrittweite $h := h * \min\{2; s\}$

wenn 2 Verfahren p-ter und p + 1-ter Ordnung gekoppelt werden.

Bem.: Vorgabe von $tol \geq k * eps_M$ wobei k mindestens der Größe von $\max|y(t)|$ für t aus dem Integrationsintervall $\left[t_k , t_k + h\right]$ entsprechen soll.

2. Weg: Koppelung von 2 Verfahren gleicher Ordnung aber unterschiedlicher Stufe. Aufgrund des Unterschiedes der Fehlerterme kann eine Fehlerschätzung vorgenommen werden.

Beisp.: RK Merson-Verfahren koppelt 2 Verfahren 4. Ordnung von 4. und 5.Stufe

		k_1	k_2	k_3	k_4	k_5
k_1	0					
k_2	$\frac{1}{3}$	$\frac{1}{3}$				
k_3	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{6}$			
k_4	$\frac{1}{2}$	$\frac{1}{8}$	0	$\frac{3}{8}$		
k_5	1	$\frac{1}{2}$	0	$-\frac{3}{2}$	2	
4. Stufe y_{k+1}		$\frac{1}{2}$	0	$-\frac{3}{2}$	2	
5. Stufe z_{k+1}		$\frac{1}{6}$	0	0	$\frac{2}{3}$	$-\frac{1}{6}$

Fehlerschätzung:

$$y(t_{k+1}) - z_{k+1} \approx \frac{h}{5}(z_{k+1} - y_{k+1}) = \frac{1}{30}(-2k_1 + 9k_3 - 8k_4 + k_5) .$$

Mit $err := \frac{h}{5} \| z_{k+1} - y_{k+1} \|$ kann wieder eine Schrittweitesteuerung aufgebaut werden.