Numerische Integration: Adaptive Verfahren und Extrapolation

Alexander Schwanecke

Zusammenfassung: Bei näherungsweisen Integrationen war es uns bisher nur möglich, anhand von einer immer genauer werdenden äquidistanten Maschenstruktur die Genauigkeit zu erhöhen. Dabei wurde aber immer außer acht gelassen, daß eine Funktion sich in unterschiedlichen Bereichen unterschiedlich stark ändert. Ziel dieses Vortrags ist es, das Prinzip der sogenannten *adaptiven Verfahren* und schließlich auch der *Extrapolation* vorzustellen.
Hierfür werden die Grundlagen und Bezeichnungen der numerischen Integration wiederholt, um dann hieraus das sog. adaptive Simpson-Verfahren herzuleiten. Zur Erläuterung werden Bedeutung und Wirkung anhand von Grafiken und Programmierbeispiele aufgeführt. Abschließend wird das Romberg-Verfahren kurz erläutert.

1 Einleitung

Bisher wurden in der Vorlesung Verfahren behandelt, die sich der Funktion nur global anpassen und selbst immer direkt Flächen berechnen. Insbesondere waren dies die abgeschlossenen Newton-Cotes-Formeln (Kap. 2).
Zwei Strategien sollen nun zusätzlich eingeführt werden: 1. Die Teilung des Integrationsintervalls in Teilintervalle unterschiedlicher Breite, die an den lokalen Grad der Veränderung der Funktion angepaßt ist, als Idee der adaptiven Verfahren (Kap. 3) und 2. die Extrapolation als Versuch von vorhandenen Iterationen auf die asymptotische Entwicklung der genäherten Flächen bei Verkleinerung der Maschenweite zu schließen (Kap. 4).

2 Grundlagen

2.1 Grundbegriffe

Eine näherungsweise Berechnung eines Integrals wird als (numerische) *Quadratur* bezeichnet. Die Entwicklung und Auswahl solcher Quadraturen muß an die speziellen Aufgabenstellungen gekoppelt sein,– insbesondere an:

1. die *Besonderheiten* der zu integrierenden Funktion (Singularitäten oder Bereiche mit starker Oszillation),
2. die *Kenntnisse* über den Integranden (Wertetabelle oder tatsächliche Definition der Funktion),
3. die geforderte *Genauigkeit* und
4. die *Anzahl* und Art der zu betrachtenden Fälle.

Insbesondere auf die Punkte 1 und 3 wird noch detailliert eingegangen. Sie sind die Standbeine der adaptiven Verfahren.

Definition 2.1 *Seien $a, b \in \mathbb{R}$, $a < b$, $f : [a, b] \to \mathbb{R}$ und f stetig. Das Integral auf dem Intervall [a,b] von f wird bezeichnet mit*

$$I(f) := \int_a^b f(x)dx.$$

2.2 Standard-Verfahren

Die aus der Vorlesung bekannten Verfahren beruhen auf der Interpolation an *Stützstellen* von Funktionen. Im folgenden werden diese für eine Funktion f mit $(t_i, f_i) := (t_i, f(t_i))$ für $i = 0, \ldots, n$ bezeichnet. Hierbei nennt man die t_i *Knoten*. Die Quadraturformeln besitzen im allgemeinen die Form

$$\tilde{I}(f) = (b - a) \sum_{i=0}^{n} \lambda_i f(t_i), \tag{1}$$

wobei $t_0, \ldots, t_n$ die benutzten Knoten und $\lambda_0, \cdots, \lambda_n$ die durch das spezifische Verfahren bestimmten *Gewichte* sind.
Die Bestimmung der Gewichte beruht auf einem je anhand der Stützstellen konstruierten Interpolationspolynom, das als Ersatz für die eigentliche Funktion integriert wird. Für äquidistante Stützstellen ist dies die Idee der sog. *Newton-Cotes-Formeln*. Es eignet sich die Schreibweise

$$\tilde{I}(f) = \int_a^b P_f(x)\, dx,$$

wofür $P_f(x)$ das durch die Stützstellen festgelegte Polynom ist.

Nach der Lagrangeschen Interpolationsformel ist dies:

$$P_f(x) = \sum_{i=0}^{n} f_i L_{in}(x), \quad L_{in}(x) = \prod_{\substack{k=0 \\ k \neq i}}^{n} \frac{x - t_k}{t_i - t_k}. \tag{2}$$

mit den $n + 1$ Knoten:

$$t_i := a + i \cdot \frac{b - a}{n}, \quad i = 0, \ldots, n. \tag{3}$$

Lemma 2.2 *Für $n+1$ paarweise verschiedene Knoten $t_0, \ldots, t_n$ exisitiert genau eine Quadraturformel*

$$\tilde{I}(f) = (b-a) \sum_{i=0}^{n} \lambda_i f(t_i),$$

die für alle Polynome $P_f \in \mathbf{P}_n$ vom Grad kleiner gleich n exakt ist.

Beweis: Wir setzen die zu den Knoten t_i gehörenden Lagrange-Polynome $L_{in} \in \mathbf{P}_n$ aus (2), die nach Vorraussetzung, ohne daß ein Fehler entsteht, integriert werden, da bei $n+1$ Stützstellen ein Polynom n–ten Grades eindeutig bestimmt ist, in die Quadraturformel ein:

$$I(L_{in}) = \tilde{I}(L_{in}) = (b-a) \sum_{j=0}^{n} \lambda_j L_{in}(t_j) = (b-a) \sum_{j=0}^{n} \lambda_j \delta_{ij} = (b-a)\lambda_i.$$

Dadurch sind die Gewichte als

$$\lambda_i = (b-a)^{-1} I(L_{in}) \tag{4}$$

eindeutig bestimmt. Insbesondere ist dies auf die Basiseigenschaften der Lagrange-Polynome zurückzuführen. □

2.3 Die Trapezregel

Als einfaches Beispiel für eine derartige Näherung einer Funktion bietet sich die Trapezregel an, die zwischen zwei Stützpunkten eine lineare Verbindung schafft (Abb.1). Die Fläche des entstehenden Trapezes wird als Näherung des Integrals benutzt,– daher die Namensgebung.
Anhand dieser Regel können Polynome bis zum Grad $n = 1$ korrekt integriert werden.

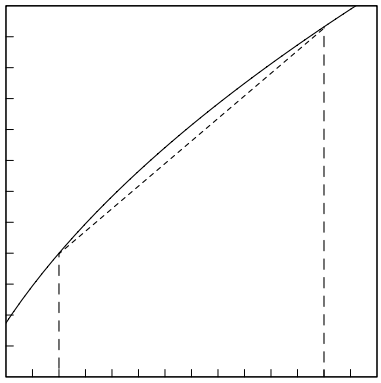

Abbildung 1: Funktionsweise der Trapezregel

Mit den Knoten $t_0 = a$ und $t_1 = b$ ergeben sich ihre Gewichte mit (2) und (4) sofort als

$$\begin{aligned} \lambda_0 &= \frac{1}{b-a} \int_a^b \prod_{k=1}^{1} \frac{x - t_k}{t_0 - t_k}\, dx = \frac{1}{b-a} \int_a^b \frac{x-b}{a-b}\, dx \\ &= \frac{-1}{(b-a)^2} \left[\frac{x^2}{2} - b \cdot x \right]_a^b = \frac{-1}{(b-a)^2} \left(\frac{-1}{2}(b^2 - 2ab + a^2) \right) = \frac{1}{2}. \end{aligned}$$

Ein analoge Rechnung ergibt $\lambda_1 = \frac{1}{2}$.

Somit ist die *Trapezregel* in der allgemeinen Form einer Quadratur (1) gegeben durch

$$\tilde{I}_T(f) = (b-a)\left(\frac{1}{2}\cdot f(a) + \frac{1}{2}\cdot f(b)\right) = \frac{t_1 - t_0}{2}\cdot(f_0 + f_1). \tag{5}$$

Ihre Anwendung ist in Abbildung 1 angedeutet. Das von den gestrichelten Linien umgebene Trapez nähert die Fläche unter der Kurve (durchgezogene Linie).

2.4 Verbesserung der Genauigkeit

Die Idee der abgeschlossenen Newton-Cotes-Formeln kann mit einer Erhöhung des Grades des Interpolationspolynoms bzw. – anders ausgedrückt – einer Erhöhung der Anzahl der Stützstellen verfeinert werden. Die Gewichte für die ersten entstehenden Formeln bei der Fortsetzung diesen Gedankens sind:

n	$\lambda_{0n}, \ldots, \lambda_{nn}$	Geläufige Namen
1	$\frac{1}{2}\quad\frac{1}{2}$	Trapezregel
2	$\frac{1}{6}\quad\frac{4}{6}\quad\frac{1}{6}$	Simpson-Formel, Keplersche Faßregel
3	$\frac{1}{8}\quad\frac{3}{8}\quad\frac{3}{8}\quad\frac{1}{8}$	Newtonsche 3/8-Regel, pulcherrima
4	$\frac{7}{90}\quad\frac{32}{90}\quad\frac{12}{90}\quad\frac{32}{90}\quad\frac{7}{90}$	Milne-Regel

Tabelle 1: Newton-Cotes-Formeln

Bei den abgeschlossenen Newton-Cotes-Formeln werden die Knoten in äquidistanten Intervallen an den Intervallanfängen bzw. –enden plaziert.

2.5 Simpson-Formel

Sie ist die Basis des herzuleitenden Simpson-Verfahrens und beruht als Newton-Cotes-Formel auf der Wahl von drei Knoten: $t_0 = a$, $t_1 = \frac{a+b}{2}$ und $t_2 = b$. Die Gewichte

werden, wie bereits bei der Trapezregel geschehen, mit (2) und (4) hergeleitet:

$$\begin{aligned}
\lambda_0 &= \frac{1}{b-a}\int_a^b \prod_{k=1}^{2} \frac{x-t_k}{t_0-t_k}\,dx \\
&= \frac{1}{b-a}\int_a^b \frac{x-t_1}{t_0-t_1}\cdot\frac{x-t_2}{t_0-t_2}\,dx \\
&= \frac{1}{b-a}\int_a^b \frac{x-\frac{a+b}{2}}{a-\frac{a+b}{2}}\cdot\frac{x-b}{a-b}\,dx \\
&= \frac{1}{-a^3+3a^2b-3ab^2+b^3}\int_a^b 2x^2+(-a-3b)x+(ab+b^2)\,dx \\
&= \frac{1}{-a^3+3a^2b-3ab^2+b^3}\left[\frac{2}{3}x^3+\frac{-a-3b}{2}x^2+(ab+b^2)x\right]_a^b \\
&= \frac{1}{6}\cdot\frac{-a^3+3a^2b-3ab^2+b^3}{-a^3+3a^2b-3ab^2+b^3} \\
&= \frac{1}{6}, \\
\lambda_1 &= \frac{1}{b-a}\int_a^b \frac{x-t_0}{t_1-t_0}\cdot\frac{x-t_2}{t_1-t_2}\,dx \\
&= \frac{1}{b-a}\int_a^b \frac{x-a}{\frac{a+b}{2}-a}\cdot\frac{x-b}{\frac{a+b}{2}-b}\,dx \\
&= \frac{4}{a^3-3a^2b+3ab^2-b^3}\int_a^b x^2-(a+b)x+ab\,dx \\
&= \frac{4}{a^3-3a^2b+3ab^2-b^3}\left[\frac{1}{3}x^3-\frac{a+b}{2}x^2+ab\cdot x\right]_a^b \\
&= \frac{4}{a^3-3a^2b+3ab^2-b^3}\cdot\frac{a^3-3a^2b+3ab^2-b^3}{6} \\
&= \frac{4}{6}
\end{aligned}$$

und

$$\begin{aligned}
\lambda_2 &= \frac{1}{b-a}\int_a^b \frac{x-t_0}{t_2-t_0}\cdot\frac{x-t_1}{t_2-t_1}\,dx \\
&= \frac{1}{b-a}\int_a^b \frac{x-a}{b-a}\cdot\frac{x-\frac{a+b}{2}}{b-\frac{a+b}{2}}\,dx = \ldots \\
&= \frac{1}{6}.
\end{aligned}$$

In der allgemeinen Form einer Quadratur (1) schreibt sich die *Simpson-Formel* demnach als:

$$\tilde{I}_S(f) = (t_2-t_0)\left(\frac{1}{6}f(t_0)+\frac{4}{6}f(t_1)+\frac{1}{6}f(t_2)\right) = \frac{h}{3}\cdot(f_0+4\cdot f_1+f_2), \qquad (6)$$

wobei hier die Maschenweite $h=(t_2-t_0)/2$ ist.

In Abbildung 2 kann man erkennen wie in eine sinusförmige Funktion (durchgezogene Linie) eine Parabel (feingestrichelte Linie) hineingelegt wird und das Integral anhand der drei Stützpunkte ausgewertet wird. Dies darf nicht mit einer zweifachen Anwendung der Trapezregel verwechselt werden. Die Gewichte λ_i, die gerade nicht je ein Drittel betragen, geben den Unterschied und bedeuten die quadratische Interpolation.

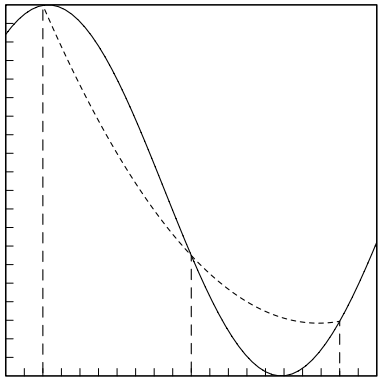

Abbildung 2: Funktionsweise der Simpson-Formel

Definition 2.3 *Damit führen wir den Begriff des Restglieds R ein:*

$$R(f) := I(f) - \tilde{I}(f).$$

Bemerkung 2.4 *Der Begriff der Genauigkeit sollte noch bekannt sein. Sie besagt bis zu welchem Grad r Polynome mit einem Verfahren noch ohne Fehler integriert werden. Es gilt bei den behandelten abgeschlossenen Newton-Cotes-Formeln $r = n$ für ungerades und $r = n + 1$ für gerades n. Für die Simpson-Formel soll dies nun explizit gezeigt werden:*

Lemma 2.5 *Die Simpson-Formel integriert nicht nur $f \in \mathbf{P}_2$, sondern auch $f \in \mathbf{P}_3$ fehlerlos.*

Beweis: Für $f(t) = \left(t - \frac{a+b}{2}\right)^3$ ist

$$I(f) = 0 \quad \Leftarrow \quad \int_a^{\frac{a+b}{2}} f(t)\,dt = -\int_{\frac{a+b}{2}}^b f(t)\,dt$$

und mit (6) folgt

$$\tilde{I}_S(f) = \frac{b-a}{6} \cdot \left(\left(a - \frac{a+b}{2}\right)^3 + 4\left(\frac{a+b}{2} - \frac{a+b}{2}\right)^3 + \left(b - \frac{a+b}{2}\right)^3 \right) = 0.$$

Da es also für eine Funktion, die genau den Grad 3 besitzt, erfüllt ist, läßt sich nun mit Hilfe der Eigenschaft der Integration als ein monotones, lineares Funktional auf dem Vektorraum der Riemann-integrierbaren Funktionen, zu denen die hier behandelten stetigen und mehrfach-differenzierbaren Funktionen gehören, dies auch für alle Vielfachen

und Linearkombinationen dieser Funktion mit Polynomen zweiten Grades und geringer schließen. Damit sind alle Polyome dritten Grades und geringer eingeschlossen. □

Im folgenden benötigen wir Konkreteres:

Satz 2.6 *Für die Simpson-Formel mit der Maschenweite* $h = \frac{b-a}{2}$ *bei einer* 4*–mal differenzierbaren Funktion* f *kann die Fehlerabschätzung*

$$|R_S(f)| \leq \frac{h^5}{90} \cdot \max_{\tau \in [a,b]} \left| f^{(4)}(\tau) \right|$$

benutzt werden.

Der Beweis findet sich z.B. im Vorlesungsskript zur Vorlesung Numerische Mathematik im Sommersemester 2001 von PD Dr. A. Meister an der Universität Hamburg. Er beruht auf einer Integration der Fehlerabschätzung für polynomiale Interpolation.

2.6 Zusammengesetzte Quadraturformeln

Um, ohne den Interpolationsgrad n des Verfahrens wachsen zu lassen, die Genauigkeit zu erhöhen, besteht die Möglichkeit, die Zahl der Stütstellen wachsen zu lassen, die bereits bekannten Newton-Cotes-Formeln auf je $n+1$ der Stützstellen anzuwenden und die Werte der einzelnen Integrale aufzusummieren. Je nach Verfahren, werden also $N = i \cdot n + 1$, $(i = 1, 2, \ldots)$ Stützstellen benutzt.

Für die Simpsonformel erhält man für $5 = 2 \cdot 1 + 1$ Knoten im Abstand h, also eine Aufsummierung in zwei Teilintegralen, die zusammengesetzte Simpsonformel mit (6):

$$\begin{aligned} \tilde{I}_{ZS} := \tilde{I}_{ZS5} &= \frac{t_4 - t_0}{12} \cdot (f_0 + 4 \cdot f_1 + 2 \cdot f_2 + 4 \cdot f_3 + f_4) \\ &= \frac{h}{3} \cdot (f_0 + 4 \cdot f_1 + 2 \cdot f_2 + 4 \cdot f_3 + f_4) \end{aligned} \tag{7}$$

bzw. in allgemeiner Form für $N = i \cdot n + 1$ Knoten:

$$\tilde{I}_{ZSN} = \frac{h}{3} \cdot \left(f_0 + 2 \sum_{k=1}^{N/2-1} f_{2k} + 4 \sum_{k=1}^{N/2} f_{2k-1} + f_N \right).$$

Satz 2.7 *Für die zusammengesetzte Simpsonformel* $\tilde{I}_{ZS}$ *(7) angewandt auf eine* 4*–mal differenzierbare Funktion* f *gilt die Fehlerabschätzung*

$$|R_{ZS}(f)| \leq \frac{h^5}{45} \cdot \max_{\tau \in [a,b]} \left| f^{(4)}(\tau) \right| \qquad \text{mit} \quad h = \frac{t_4 - t_0}{4}.$$

Dieser Satz läßt sich direkt aus Satz 2.6 durch Summation und Abschätzung von τ für das Gesamtintervall herleiten.

Da die Maschenweite bei der zusammengesetzten Simpsonformel halb so groß wie bei der einfachen Simpsonformel ist, liegt die Schranke dort um den Faktor 2^{-4} niedriger.

3 Adaptives Verfahren nach Simpson

3.1 Funktionsweise eines adaptiven Verfahrens

Bisher haben wir nur mit äquidistanten Knoten gearbeitet und so ein einfaches und klares Formelrepertoire erhalten. Wollten wir höhere Genauigkeiten, so wurden einfach der Grad oder die Anzahl der Zusammensetzungen erhöht, um eine geringere Maschenbreite und einen als geringer abschätzbaren Fehler zu erhalten. Die Verfahren sind zwar recht einfach, doch rechnen diese bei schon längst vorhandener hoher Genauigkeit der Iteration immer noch weiter. Die Besonderheiten der Funktion, sei es ihre Linearität oder auch ihr großer Veränderungsgrad werden nur global berücksichtigt und rufen, wegen kleinen Bereichen starker Veränderung evtl. eine sehr enge Maschenbreite bei vorgegebenem Fehler hervor.

Der Gedanke der adaptiven Verfahren ist also gerade die lokale Anpassung an das Verhalten der Funktion. Dies entspricht in erster Linie einer lokal individuellen Veränderung der Maschenweite und benötigt Werkzeug zur daraus resultierenden globalen Fehlerabschätzung. Als erstes werden Abbruchkriterien gesucht, die zeigen, wann eine Verringerung (vorschlagsweise: Halbierung) der Maschenweite ausreicht.

Das Verfahren nach *Gander* [2] beschreibt die Anwendung einer Newton-Cotes-Formel $\tilde{I}_1(f)$ und der darauf beruhenden zusammengesetzten Formel $\tilde{I}_2$ für das gleiche Teilintervall und läßt abbrechen, wenn in Maschinenarithmetik

$$I_S + \tilde{I}_1 = I_S + \tilde{I}_2$$

gilt, wobei I_S ein Schätzert für das insgesamt zu erhaltende Integral ist. So läßt sich die Rechengenauigkeit des Computers ausreizen. Für vorgegebene relative Genauigkeiten gibt es darauf aufbauende Methoden.

Der Schätzwert ist natürlich im vornherein zu ermitteln; doch wie erhält man einen sinnvollen, wenn nicht obere oder untere Schranken der Funktion bekannt sind, und wie geht man mit um den Nullpunkt schwankenden Funktionen um, die eventuell ein Integral sehr geringer Größenordnung besitzen? Im weiteren wird für das Simspon-Verfahren, ein noch eleganteres Abbruchkriterium entwickelt.

3.2 Simpson-Verfahren

Dieses Verfahren, der zentrale Punkt des Vortrags, beruht auf einer Auswertung des Integranden in Teilstücken mit der Simpsonformel $\tilde{I}_S$ (6) und der zusammengesetzten Simpsonformel $\tilde{I}_{ZS}$ (7) zur Fehlerabschätzung. Um dies zu verstehen, wird noch weiteres Rüstzeug benötigt. Hierfür möchte ich eine zusammenfassende Fehlerabschätzung motivieren:

Bemerkung 3.1 *Wählt man beim Simpson-Verfahren die Knotenabstände h_i so, daß für die genäherten Integrale jedes zugehörigen Intervalls $[a_i, b_i]$*

$$\left|\tilde{I}_{ZS}(f) - \tilde{I}_S(f)\right| \leq \frac{15 h_i \epsilon}{b-a}$$

gilt, dann läst sich als Restgliedabschätzung für das auf L Teilintervalle mit je drei Knoten für die Simpsonformel und je fünf Knoten für die zusammengesetzte Simpsonformel angewandte und aufsummierte Simpson-Verfahren folgendes näherungsweise verwenden:

$$|R_{SV}(f)| = \left|I(f) - \tilde{I}_{SV}(f)\right| = \left|I(f) - \sum_{i=1}^{L} \tilde{I}_{ZS}(f,[a_i,b_i])\right| \leq \epsilon.$$

Herleitung dieses Gedankengangs
Werden auf ein Teilintervall $[a_i, b_i]$ der Breite $2 \cdot h_i = b_i - a_i$ Simpsonformel und zusammengesetzte Simpsonformel auf eine 4–mal differenzierbare Funktion f angewandt, so gelten für die entstehenden Fehler mit (2.6) und (2.7) die folgenden Abschätzungen:

$$\begin{aligned}\left|R_S^{(i)}(f)\right| &\leq \frac{h_i^5}{90} \cdot \max_{\tau\in[a_i,b_i]} \left|f^{(4)}(\tau)\right|, \\ \left|R_{ZS}^{(i)}(f)\right| &\leq \frac{h_i^5}{2^5 \cdot 45} \cdot \max_{\tau\in[a_i,b_i]} \left|f^{(4)}(\tau)\right|.\end{aligned}$$

Damit gilt in guter Näherung bezogen auf das Intervall $[a_i, b_i]$ und auf die Funktion f

$$\begin{aligned}16 \cdot \left|R_{ZS}^{(i)}\right| &= \left|R_S^{(i)}\right| \quad \Rightarrow \\ 16 \cdot \left|I^{(i)} - \tilde{I}_{ZS}^{(i)}\right| &= \left|I^{(i)} - \tilde{I}_S^{(i)}\right| \quad \Rightarrow \\ 16 \cdot \left|I^{(i)} - \tilde{I}_{ZS}^{(i)}\right| &= \left|I^{(i)} - \tilde{I}_{ZS}^{(i)} + \tilde{I}_{ZS}^{(i)} - \tilde{I}_S^{(i)}\right| \quad \Rightarrow \\ 16 \cdot \left|I^{(i)} - \tilde{I}_{ZS}^{(i)}\right| &\leq \left|I^{(i)} - \tilde{I}_{ZS}^{(i)}\right| + \left|\tilde{I}_{ZS}^{(i)} - \tilde{I}_S^{(i)}\right| \quad \Rightarrow \\ 15 \cdot \left|I^{(i)} - \tilde{I}_{ZS}^{(i)}\right| &\leq \left|\tilde{I}_{ZS}^{(i)} - \tilde{I}_S^{(i)}\right|\end{aligned}$$

und aufsummiert über alle Teilintervalle:

$$\begin{aligned}\left|\sum_{i=1}^{L}\left(I^{(i)} - \tilde{I}_{ZS}^{(i)}\right)\right| &\leq \sum_{i=1}^{L}\left|I^{(i)} - \tilde{I}_{ZS}^{(i)}\right| \leq \frac{1}{15}\sum_{i=1}^{L}\left|\tilde{I}_{ZS}^{(i)} - \tilde{I}_S^{(i)}\right| \quad \Rightarrow \\ |R_{SV}| &= \left|I - \sum_{i=1}^{L}\tilde{I}_{ZS}^{(i)}\right| \leq \frac{1}{15}\frac{15\epsilon}{b-a}\sum_{i=1}^{L} h_i = \epsilon.\end{aligned}$$

3.3 Algorithmus

Mit (3.1) ist nun der letzte Baustein gegeben, der einen Algorithmus ermöglicht, der bei einem vorgegebenen Integranden f und den Intervallgrenzen $[a, b]$ mit einer Toleranz ϵ integriert.

In einem Metacode könnte dies so aussehen:

function $SV(t,h)\{$
  $\tilde{I}_S = h/3 \cdot (f(t) + 4 \cdot f(t+h) + f(t+2h))$
  $\tilde{I}_{ZS} = h/6 \cdot (f(t) + 4 \cdot f(t+h/2) + 2 \cdot f(t+2h) + 4 \cdot f(t+3/2h) + f(t+2h))$
  if $\left(\left|\tilde{I}_{ZS} - \tilde{I}_S\right| \leq (15 \cdot h \cdot \epsilon)/(b-a)\right)\{$
    $I = I + \tilde{I}_{ZS}$
    $SV(t+2h, (b-(t+2h))/2)$
  $\}$
  else $SV(t, h/2)$
$\}$

Für die Vorführung habe ich einen rekursiven Algorithmus gewählt. Die Prozedur kann mit $SV(a,(b-a)/2)$ aufgerufen werden. Es werden ihr der Startpunkt t und die Maschenweite der anzuwendenden Simpsonformel h übergeben.
Entsprechend (6) und (7) wird der betrachtete Bereich $[t, t+2h]$ mit der Simpsonformel und der zusammengesetzten näherungsweise integriert.
Über (3.1) wird entschieden, ob die Genauigkeit reicht und der verbleibende Intervall der Funktion nun angegangen werden soll, oder ob für eine Präzisierung mit der halben Maschenweite der gleiche Startwert nochmals benutzt werden soll.
In der Variablen I wird das Gesamtintegral aufsummiert.

Zusätzlich ist nun ein lauffähiges C++ Programm abgebildet.

```
// Simpson.cpp ---- Ein kleines C++ - Programm
#include <iostream>
#include <cmath>
using namespace std;

double a = 0.;                // Linke Grenze.
double b = 1.;                // Rechte Grenze.
double e = 1e-7;              // Erlaubter Gesamtfehler.
double g = 15. * e / (b-a);   // Spart Berechnungen.
double I = 0.;                // Init.

double f(double x) {    // Integrand - hier aus Beispiel 1.
 return (sin(20*x*x));
}

void SV(double t, double h) {
 if (t >= b) return;    // Erforderlicher Bereich integriert?
 // Die Simpsonregel.
 double IS  = h / 3 * ( f(t) + 4 * f(t+h) + f(t+2*h) );
 // Die zusammengesetzte Simpsonregel.
 double IZS = h / 6 * ( f(t) + 4 * f(t+h/2) +
                        2 * f(t+h) + 4 * f(t+3*h/2) + f(t+2*h) );
```

```
 if (abs(IZS-IS) <= (g * h)) { // Die Fehlerabschaetzung.
  I += IZS;                    // Aufsummierung.
  SV(t+2*h, (b - (t+2*h))/2);  // Restlicher Intervall.
 } else SV(t,h/2);             // Verfeinerung.
 return;
}

int main() {
 SV(a,(b-a)/2); // Start der rekursiven Schleife
 cout << "\nI = (" << I << " +- " << e << ")\n";
}
```

3.4 Beispiele

Mit einer etwas komplexeren Variante dieses Programms, entstanden einige Beispiele. In den zugehörigen Abbildungen sind die kleinen Dreiecke im unteren Bereich der Zeichnungen die vom Verfahren gewählten Orte der Teilintervallgrenzen.

In Abbildung 3 ist $\int_0^1 \sin(20 \cdot x^2)\,dx$ für $\epsilon = 1 \cdot 10^{-5}$ approximiert. Hierbei wurde die Funktion SV insgesamt 172–mal aufgerufen. Das Programm lieferte mit $I = 0.129376047$ statt 0.129376027 sogar ein höhere als die geforderte Genauigkeit.

In Abbildung 4 ist über $\int_0^1 f(x)\,dx$ eine verschachtelte Funktion

$$f(x) = \left\{ \begin{array}{lll} -200(x-0.2)^2 + 9 & für & 0.0 \le x < 0.2 \\ 9 & für & 0.2 \le x < 0.4 \\ 9\cos\left(\frac{8\pi}{3}(0.7-x)\right) & für & 0.4 \le x < 0.7 \\ 9\frac{0.7}{x}\cos\left(\frac{50\pi}{3}(0.7^2-x^2)\right) & für & 0.7 \le x \le 1.0 \end{array} \right\}$$

für $\epsilon = 1 \cdot 10^{-3}$ approximiert. Hierbei wurde die Funktion SV insgesamt 136–mal aufgerufen. Das Programm lieferte $I = 3.13064$ statt 3.13153 und bleibt damit im zuvor gesteckten Rahmen.

Man erkennt gut, wie der Algorithmus bei Unterschreitung der entsprechenden Teilintervallgrenzen für den quadratischen wie den linearen Anteil der Funktion keine weiteren Stützstellen benötigt. Die trotzdem zusätzlich aufgetretenen Stützstellen beruhen lediglich darauf, daß die Halbierung der Breite des untersuchten Intervalls nicht genau die Grenzen der Teilabschnitte der Funktion trifft.

4 Extrapolation: Verfahren nach Romberg

4.1 Idee

Im Romberg-Verfahren geht man davon aus, daß sich bei sinkender Maschenweite die näherungsweise berechneten Integrale dem tatsächlichen Wert asymptotisch annähern.

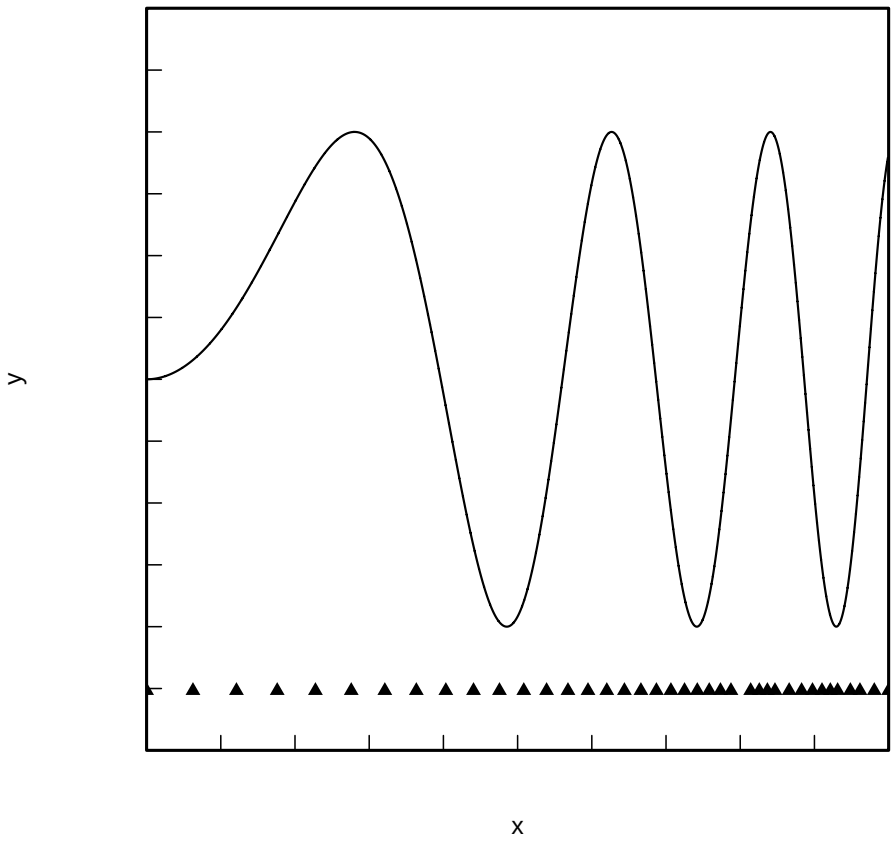

Abbildung 3: Beispiel einer oszillierenden Funktion

Die *Euler-Maclaurinsche Summenformel* gibt für diese Annahme den Anlaß. Sie besagt, daß sich die Iterationen mit einer Verringerung der Maschenweite dem exakten Wert in Form einer polynomialen Funktion mit geraden Koeffizienten nähern. Das Ziel ist nun, den Achsenabschnitt dieser Funktion $\tilde{I}(h)$ zu finden. Denn hier ist die Maschenweite $h = 0$ und damit müßten wir einen fehlerlosen Wert des Integrals $I(f)$ bestimmt haben. Ganz so gut funktioniert das Verfahren nicht, aber immerhin kann es in vielen Fällen einen guten Näherungswert für $I(f)$ bestimmen.

4.2 Grundlagen des Verfahrens

Struktur und Aufbau der Euler-Maclaurinschen Summenformel sind komplex, daher werden nur die nötigen Spezialfälle eingeführt. Zudem benutzen wir als Verfahren lediglich die Trapezregel. Eine Herleitung findet sich z.B. in [5].

Für die Trapezregel (5) läßt sich die Euler-Maclaurinsche Summenformel umstellen und nach der Maschenweite entwickeln, so daß die Trapezregel für eine Funktion $f \in C^{2m}[a,b]$ bei einer Maschenweite $h = \frac{b-a}{n}$ auf dem Intervall $[a,b]$ folgende Form annimmt

$$\tilde{I}_T(f,h) = I(f) + c_1 h^2 + c_2 h^4 + \ldots + c_{m-1} h^{2m-2} + \alpha_m(h) \cdot h^{2m}. \tag{8}$$

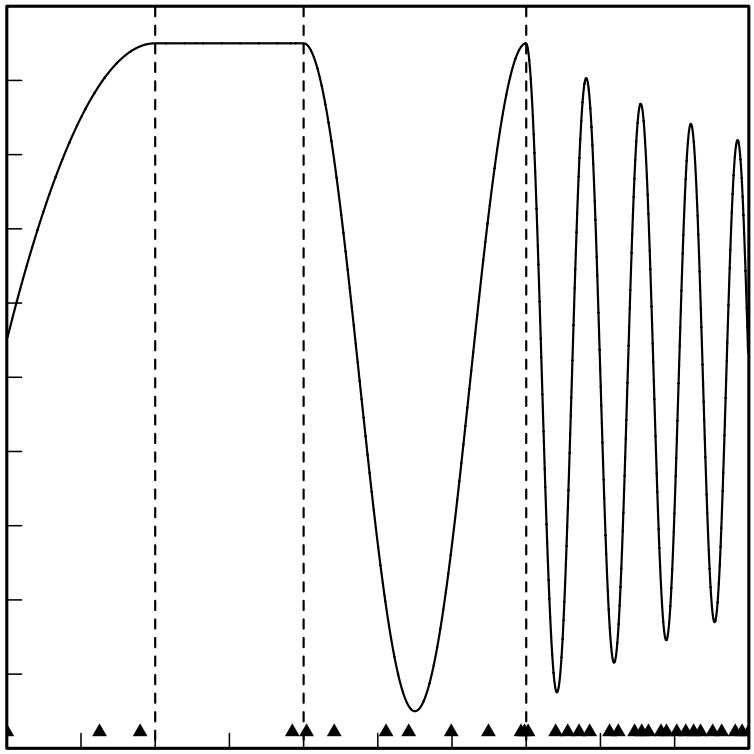

Abbildung 4: Beispiel einer zusammengesetzten Funktion

Zwar sind die c_k, $k = 1, 2, \ldots, m$ definiert, doch ist für uns nur wichtig, daß sie unabhängig von der Maschenweite h sind, während zwar der sogenannte Restgliedkoeffizient $\alpha_m(h)$ hiervon abhängt, aber beschränkt ist. Als kurzer Ausblick und um die kommenden Schlußfolgerungen zu motivieren (Beweis siehe Lehrbücher):

$$|\alpha_m(h)| \leq \left| \frac{B_{2m}}{(2m)!}(b-a) \right| \max_{x \in [a,b]} \left| f^{(2m)}(x) \right|, \quad B_k \,\text{Bernoulli} - \text{Zahl}.$$

Bei einer wachsenden Zahl an Ausgangswerten m nimmt also die Schranke mit hoher Geschwindigkit (Fakultät) ab.
Daher bestimmen wir dasjenige Polynom, das auf den Stützstellen $(h_k^2, \tilde{I}_T(f, h_k))$, $k = 1, 2, \ldots, m$ beruht, als

$$T_{m,m}(h) = a_0 + a_1 h^2 + a_2 h^4 + \ldots + a_{m-1} h^{2m-2}.$$

An dieser Stelle wird ersichtlich, daß nun nur noch die Maschenweite zu Null gesetzt werden muß, um mit $T_{m,m}(0)$ einen Näherungswert für $I(f)$ zu erhalten.

Nun wird die rekursive Variante des *Nevilleschen Algorithmus* aus dem Bereich der Interpolation wieder interessant. Setzt man den Parameter zu Null entsteht eine vereinfachte und für uns genau passende Variante, die wie folgt lautet:

$$\begin{aligned} T_{i,0} &= \tilde{I}_i(f, h) \\ T_{i,k} &= T_{i,k-1} + \frac{T_{i,k-1} - T_{i-1,k-1}}{\left(\frac{h_{i-k}}{h_i}\right)^2 - 1}, \quad 1 \leq k \leq i \leq m. \end{aligned}$$

Diese Art der Berechnung aufgrund der beiden jeweils links benachbarten Elemente ist aus dem dreieckigen Nevilleschema bekannt.

Die Lehrbücher schlagen zur Veränderung der Maschenweite die sogenannte Rombergfolge vor: $h_0 = b - a$, $h_k = h_{k-1}/2$, $k = 1, 2, \ldots$
Hierüber läßt sich also

$$\left(\frac{h_{i-k}}{h_i}\right)^2 = \left(\frac{b-a}{2^{i-k-1}}\frac{2^{i-1}}{b-a}\right)^2 = 2^{2k}$$

vereinfachen. Damit sind nun alle Vorarbeiten für einen Algorithmus getätigt:

4.3 Algorithmus

Im Feld `T[i][k]` werden als Ausgangsbasis die ersten `m` Iterationen mit der Trapezregel `Trapez(h,w)` an den Positionen `T[i][0]` gespeichert, wobei die Funktion `Trapez(h,w)` über `w` Trapeze der Maschenweite `h` integriert. Das Neville-Schema in der mit dem Dreieck nach rechts zeigenden Form wird von der Funktion `Romberg()` in senkrechten Streifen von links nach rechts abgearbeitet.

```
// Romberg.cpp ---- Ein kleines C++ - Programm
#include <iostream>
#include <cmath>
using namespace std;

double a = 0.;                // Linke Grenze.
double b = 1.;                // Rechte Grenze.
int    m = 9;                 // Anzahl der Maschenweitehalbierungen.

double f(double x) { return (sin(20*x*x)); }  // Definition der Funktion.

double Trapez(double h, int w) {  // Trapezregel.
 double temp = 0.;                           // Zwischenrergebnisse.
 for (int v = 0; v < w; v++) {               // Teilintervalle.
  temp += h * ( f(a+v*h) + f(a+(v+1)*h) ) / 2;   // Aufsummierung,
 }
 return temp;
}

void Romberg() {  // Interpolationsalgorithmus anch Neville.
 double T[m+1][m+1]; // Feld der Zwischenergebnisse.
 for (int i=0; i <= m; i++) {  // Trapezregel fuer versch. "h".
  T[i][0] = Trapez( (b-a)/(pow(2.,i)) , int(pow(2.,i)) );
 }
 cout << endl;
 for (int k = 1; k <= m; k++) { // Fuellen aller Positionen im Schema.
```

```
  for (int i = k; i <= m; i++) {
   T[i][k] = T[i][k-1] + ( T[i][k-1] - T[i-1][k-1] ) / ( pow(2.,2.*k)-1 );
  }
 } // Ausgabe aller Iterationen
 for (int i = 0; i <= m; i++) { cout <<"T["i<<"]["<<i<<"] = "<<T[i][i]<<"\n"; }
 return;
}

int main() {
 cout << "Romberg-Verfahren...\nIterationen:\n";
 cout.precision(10);
 Romberg(); // Start des Verfahrens.
}
```

4.4 Beispiel

Das im vorigen Kapitel benutzte erste Beispiel $\int_0^1 \sin(20 \cdot x^2)\, dx$ ruft z.B. folgende Ausgabe hervor:

```
T[0][0] = 0.4564726254
T[1][1] = -0.487125308
T[2][2] = -0.0635424738
T[3][3] = 0.3239419392
T[4][4] = 0.1026121748
T[5][5] = 0.1303773102
T[6][6] = 0.1293661422
T[7][7] = 0.1293760499
T[8][8] = 0.1293760268
T[9][9] = 0.1293760268
```

bei einem korrekten Wert von $I(f) = (0.1293760267 \pm 10^{-10})$.

5 Zusammenfassung

Dieser Vortrag stellte zwei verschiedene Verfahren vor, die vom Weg der sturen Anwendung der aus der Vorlesung bereits bekannten Newton-Cotes-Formeln abweichen. Auf der einen Seite ist dies die Idee der Adaption. Sie paßt die Maschenweite dem Aussehen der Funktion an und kann sich sogar in unserem Anwenungsbeispiel mit der zusammengesetzten Simpsonregel eine absolute Genauigkeit vorgeben.
Auf der anderen Seite verfolgt die Extrapolation die Idee, den Weg von Iterationen mit immer feiner werdender Maschenstruktur vorherzusagen und daraus zu schliessen, was die Iteration bei unendlich kleiner Maschenweite hervorbringen würde.

Literatur

[1] G. Opfer. *Numerische Mathematik für Anfänger*, 2. Auflage, S. 83ff. Vieweg, Braunschweig, Wiesbaden, 1994.

[2] H. R. Schwarz. *Numerische Mathematik*, S. 319ff. Teubner, Stuttgart, 1986.

[3] R. Schaback, H. Werner. *Numerische Mathematik*. Springer, Berlin, 1992.

[4] P. Deuflhard, A. Hohmann. *Numerische Mathematik I.* de Gruyter Lehrbuch, Berlin, 1993.

[5] J. Stoer. *Numerische Mathematik 1.* Springer, Berlin, 1989.

[6] W. Törnig, P. Spellucci. *Numerische Mathematik für Ingenieure und Physiker*, Band 2. Springer, Berlin, 1990.

[7] O. Forster. *Analysis 1*, 5. Auflage. Vieweg, Braunschweig, Wiesbaden, 1999.