Aufgabe 6

Bearbeitungszeit: 2 Wochen (bis Freitag, 20. Juli 2018)

Mathematischer Hintergrund: Eindimensionale adaptive Quadratur, Quadraturformeln

Elemente von C++: Rekursion, Einbinden einer Grafikbibliothek, Makefiles

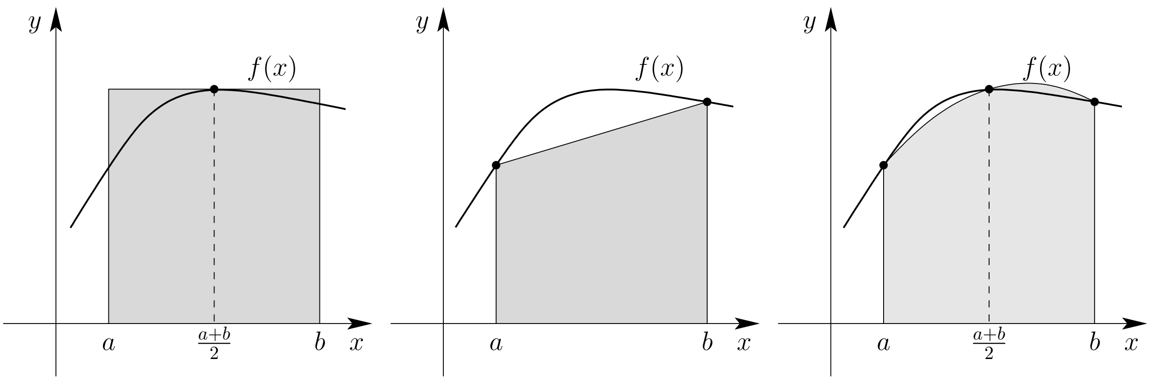
Aufgabenstellung

Gegeben seien $a, b \in \mathbb{R}$, $a < b$ und eine stetige Funktion $f : [a, b] \to \mathbb{R}$. Schreiben Sie ein Programm, welches das Integral $I(x) = \int_a^b f(x)\,\mathrm{d}x$ bis auf einen vorgegebenen Fehler $\varepsilon > 0$ berechnet. Verwenden Sie dafür eine adaptive Unterteilung des Intervalls $[a, b]$ und vergleichen Sie den benötigten Aufwand mit dem Aufwand im Falle einer äquidistanten Unterteilung von $[a, b]$.

Quadraturformeln

Da Integrale nur in den wenigsten Fällen analytisch geschlossen berechenbar sind, bemüht man numerische Methoden zur Berechnung von Integralen (*Quadratur*). Die resultierenden Näherungsformeln werden als *Quadraturformeln* bezeichnet. Hierfür legt man N Stützstellen $x_i \in [a, b]$, $1 \leq i \leq N$ fest, an denen die Funktion f ausgewertet wird.

In Abbildung 1 sind die drei einfachsten Quadraturformeln *Mittelpunktsregel*, *Trapezregel* und *Simpson-Regel* dargestellt. Diese drei Quadraturformeln sind allesamt sogenannte Newton-Cotes-Formeln, welche man erhält, indem man die Funktion f an den Stützstellen x_i durch ein Polynom interpoliert und dieses anschließend exakt integriert. [1]



(a) Mittelpunktsregel (b) Trapezregel (c) Simpson-Regel

Abbildung 1: Einfache Newton-Cotes-Quadraturformeln

Die entstehenden Newton-Cotes-Formeln kann man in allgemeiner Form als

$$\int_a^b f(x)\,\mathrm{d}x \approx (b-a) \sum_{i=1}^{N} \gamma_i f(x_i)$$

schreiben, wobei die $\gamma_i \in \mathbb{R}$ auch als Gewichte der Stützstellen bezeichnet werden. In Tabelle 1 stehen die Stützstellen und Gewichte der ersten Newton-Cotes-Quadraturformeln auf dem Intervall $[a, b]$.

Quadraturformel	**Stützstellen x_i**	**Gewichte γ_i**
Mittelpunktsregel	$\frac{a+b}{2}$	1
Trapezregel	a, b	$\frac{1}{2}, \frac{1}{2}$
Simpson-Regel	$a, \frac{a+b}{2}, b$	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$

Tabelle 1: Newton-Cotes-Quadraturformeln auf dem Intervall $[a, b]$

Exaktheitsgrad und Ordnung

Man klassifiziert Quadraturformeln danach, bis zu welchem Grad sie Polynome exakt integrieren, d.h. bis zu welchem $m \in \mathbb{N}$ gilt

$$\int_a^b p(x)\,\mathrm{d}x = (b-a) \sum_{i=1}^{N} \gamma_i \, p(x_i) \quad \text{für alle} \quad p \in P_m,$$

wobei P_m die Menge der Polynome vom Grad kleinergleich $m \in \mathbb{N}$ bezeichne. Man sagt in diesem Fall auch, die Quadraturformel ist exakt vom Grade m.

Die Mittelpunktsregel und die Trapezregel sind exakt vom Grade 1. Die Simpson-Regel ist sogar exakt vom Grade 3. [1]

Ist eine Quadraturformel exakt vom Grade m und ist f eine $(m+1)$-mal stetig differenzierbare Funktion, so kann man die Funktion f bei der Quadratur durch ihre Taylor-Entwicklung

$$f(x) = \sum_{k=0}^{m} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k + \mathcal{O}((x-x_0)^{m+1}).$$

ersetzen. Die Quadraturformel integriert dann das Taylor-Polynom m-ten Grades exakt. Der Fehler bei der Quadratur hängt dann von der Länge des Integrationsintervalls in $(m+1)$-ter Potenz ab, ist also von Ordnung $m+1$.

Summierte Quadraturformeln

Zur Steigerung der Genauigkeit wendet man eine Quadraturformel meist nicht direkt auf das Intervall $[a, b]$ an, sondern unterteilt $[a, b]$ zunächst in Teilintervalle und wendet die Quadraturformel dann auf die Teilintervalle an. Auf diese Weise erhält man sogenannte *summierte Quadraturformeln.*

Unterteilt man das Intervall $[a, b]$ äquidistant in n Teilintervalle der Länge $h := \frac{b-a}{n}$, so erhält man die Stützstellen $\xi_j = a + jh$ für $0 \leq j \leq n$, d.h. $\xi_0 = a$ und $\xi_n = b$. Wendet man auf jedem Teilintervall $[\xi_j, \xi_{j+1}]$ die Trapezregel

$$I_T(f, [\xi_j, \xi_{j+1}]) := \frac{1}{2}\,(\xi_{j+1} - \xi_j)\,[f(\xi_j) + f(\xi_{j+1})] = \frac{h}{2}\,[f(\xi_j) + f(\xi_{j+1})]$$

an, so erhält man die summierte Trapezregel

$$\begin{aligned} S_{T,n}(f, [a, b]) &:= \sum_{j=0}^{n} I_T(f, [a + jh, a + (j+1)h]) \\ &= \sum_{j=0}^{n} \left[\frac{h}{2} f(\xi_j) + \frac{h}{2} f(\xi_{j+1})\right] = \frac{h}{2} f(a) + h \sum_{j=1}^{n-1} f(a + jh) + \frac{h}{2} f(b). \end{aligned}$$

Adaptivität

In vielen Fällen ist es zum Erreichen der gewünschten Genauigkeit des Integrals nicht notwendig, das gesamte Intervall in kleine Teilintervalle zu zerlegen. Verhält sich die Funktion f z.B. auf einem großen Teilintervall wie eine Gerade, so wird sie hier bereits durch die Mittelpunktsregel exakt integriert. In einem anderen Abschnitt kann es jedoch nötig sein, das Intervall in viele Teilintervalle zu unterteilen.

Liegt eine Möglichkeit vor, den Fehler der aktuellen Näherung auf einem Teilintervall zu schätzen, so kann man die weitere Unterteilung in Abhängigkeit dieser Fehlerschätzung steuern. Diese Fähigkeit eines Verfahrens, sich selbst an das gegebene Problem anzupassen, wird als *Adaptivität* bezeichnet.

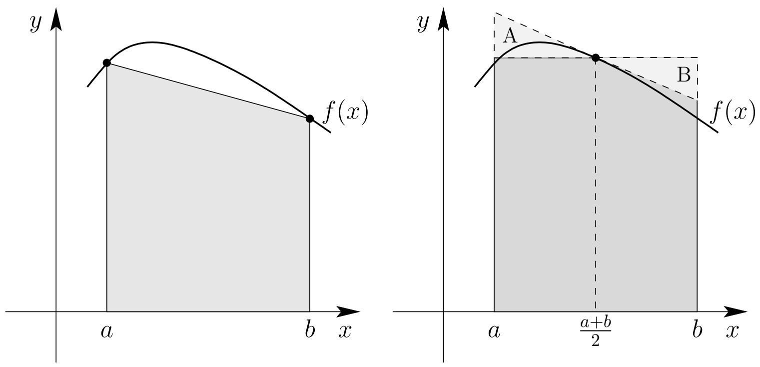
Einschluss der Lösung bei konvexen oder konkaven Funktionen

Ist die gegebene Funktion f auf dem Intervall $[a, b]$ differenzierbar und zudem konvex oder konkav, so schließen die durch die Mittelpunktsregel und Trapezregel numerisch bestimmten Näherungen des Integrals I_M und I_T den Wert des exakten Integrals $I = \int_a^b f(x)\,\mathrm{d}x$ ein, d.h. es ist

$$\min\{I_M, I_T\} \leq I \leq \max\{I_M, I_T\}\,. \tag{1}$$

Beweis: O.E. sei f konkav auf dem Intervall $[a, b]$ (ansonsten ist f konvex und somit $-f$ konkav und man betrachte $-f$, $-I_M$ und $-I_T$).

Für die Sekante s von f über dem Intervall $[a, b]$ gilt aufgrund der Konkavität $s(x) \leq f(x)$ für alle $x \in [a, b]$. Folglich ist $I = \int f(x)\,\mathrm{d}x \geq \int s(x)\,\mathrm{d}x = I_T$.



(a) Trapezregel (b) Mittelpunktsregel

Abbildung 2: Einschluss der Lösung bei konkaver Funktion f

Es sei t die Tangente an den Funktionsgraphen von f im Punkt $x = \frac{a+b}{2}$ (siehe Abbildung 2(b)). Dann entspricht die Fläche unter t über dem Intervall $[a, b]$ der von der Mittelpunktsregel berechneten Fläche, da die Dreiecke A und B in Abbildung 2(b) gleich groß sind. Aufgrund der Konkavität kann f jedoch nicht oberhalb von t liegen und es gilt $I = \int f(x)\,\mathrm{d}x \leq \int t(x)\,\mathrm{d}x = I_M$. $\square$

Die Abschätzung (1) kann man bei konvexen oder konkaven Funktionen nutzen und $|I_M - I_T|$ als Schätzung des Quadraturfehlers verwenden. In der Praxis fällt diese Schätzung allerdings eher pessimistisch aus.

Wurde die Funktion f bereits an den Stützstellen der Mittelpunkts- und der Trapezregel ausgewertet, so kann mithilfe der Simpson-Regel leicht die Approximation höheren Exaktheitsgrades bestimmt werden, da die Stützstellen der Simpson-Regel gerade die Stützstellen der Mittelpunkts- und der Trapezregel sind.

Umsetzung im Programm

Für das Programm verfolgen wir folgende Strategie: Soll das Integral der Funktion f im Intervall $[a, b]$ mit einem absoluten Fehler von höchstens ε berechnet werden, so wenden wir zunächst die Mittelpunkts- und die Trapezregel an und berechnen den Fehlerschätzer $|I_M - I_T|$. Ist die gewünschte Genauigkeit erreicht, so geben wir das mit der Simpson-Regel berechnete Ergebnis zurück.

Wurde die Genauigkeit ε verfehlt, so wird das Intervall in zwei Teilintervalle gleicher Länge zerlegt und das Verfahren auf jedem Teilintervall wiederholt, wobei nun jeweils die Genauigkeit $\frac{\varepsilon}{2}$ verlangt wird.

Der Aufwand einer Quadratur wird oft über die Anzahl der Funktionsauswertungen gemessen. Da eine Funktionsauswertung sehr aufwändig sein kann[1], sollten Sie vermeiden, die Funktion f mehrmals an der gleichen Stützstelle auszuwerten.

Um die Vorzüge der adaptiven Rechnung beobachten zu können, vergleichen Sie die Anzahl der benötigten Funktionsauswertungen mit jener bei der summierten Trapezregel mit äquidistanter Schrittweite. Gehen Sie bei der summierten Trapezregel von der minimalen Schrittweite, welche im adaptiven Algorithmus nötig ist, aus.

Funktionsauswertungen und Zielgenauigkeit

Fragen Sie zu Beginn Ihres Programmes ab, welches Beispiel gerechnet werden soll. Fangen Sie mögliche Fehler bei der Eingabe ab. Rufen Sie anschließend die Routine

```
void Start(int Bsp, double &a, double &b, double &epsilon,
           bool Grafik=true, int Pause=300);
```

auf, welche Ihnen die Grenzen des Intervalls $[a, b]$ sowie die gewünschte Zielgenauigkeit des Ergebnisses `epsilon` zurückgibt. Die optionale Grafikausgabe können Sie mit `Grafik` aktivieren. Der Parameter `Pause` steuert die Ausgabegeschwindigkeit.

Die zu integrierende Funktion können Sie anschließend durch einen Aufruf der Funktion

```
double f(double x);
```

auswerten.

Geben Sie am Ende ihres Programmes das Resultat Ihrer numerischen Quadratur durch einen Aufruf von

```
bool Ergebnis(double I);
```

zurück. Es wird dann mit dem Wert des exakten Integrals verglichen und Ihr Ergebnis wird bewertet.

Literatur

[1] Dahmen, W. und A. Reusken: *Numerische Mathematik für Ingenieure und Naturwissenschaftler*. Springer Verlag, Heidelberg, 2. Auflage, 2008.

[1] Ist der gesuchte Funktionswert beispielsweise der Auftrieb einer Flugzeugtragfläche in Abhängigkeit eines Parameters, so kann zur Berechnung eine sehr komplexe Strömungssimulation notwendig sein.