2. Übung, 23.05.2019

1 Numerische Genauigkeit

Ziel der Aufgabe ist, es ein Verständnis für die endliche Rechengenauigkeit bei der Berechnung mit Fließzahlen beim Programmieren zu entwickeln.

1.1 Grenzwertbildung

Berechnen Sie zunächst analytisch den Grenzwert der Funktion

$$f(x) = \frac{\sqrt{16+x}-4}{x}$$

mit Hilfe der Regel von de l'Hôpital für $x \to 0$.

Schreiben Sie ein Fortran Programm, das den Grenzwert numerisch berechnet. Das Programm soll folgende Anforderungen erfüllen:

- Die Genauigkeit der verwendeten Fließzahlen soll mit Hilfe der Funktion `Kind(Argument)` festgelegt werden. Wählen Sie zunächst die sgn. doppelte Genauigkeit für alle im Programm verwendeten Fließkommazahlen (64bit).
- Die Funktion $f(x)$ soll für ein Intervall $[a, b]$ berechnet werden, wobei die Anzahl der Stützpunkte im Intervall ebenfalls ein Eingabeparameter sein soll.
- Die Wertepaare $x, f(x)$ sollen für alle $x \in [a, b]$ in ein Datei geschrieben werden. Der Name der Datei soll ein Eingabeparameter sein. Verwenden Sie eine geeignete Formatierung der Ausgabe.

Übersetzen Sie Ihr Programm, und lassen Sie es für sinnvolle Angaben der Intervallgrenzen laufen (z.B. $[a, b] = [1d-20, 1d-10]$ und Anzahl der Stützstellen 1000.

Tragen Sie mit Hilfe von `gnuplot` das Ergebnis auf. Den Wertebereich der Ausgabe können Sie mit Hilfe des folgenden Kommandos einstellen:

```
gnuplot>  pl [0:1e-11][0.12:0.13] 'erg.dat' w l
```

Der angezeigte Wertebereich ist nun $x \in [0, 10^{-11}]$ und $y \in [0.12, 0.13]$. Alternativ können Sie auch die Maus zur Auswahl der Wertebereich verwenden.

Was beobachten Sie?
Welchen Wert für $f(x)$ erhält man für $x = 1.d-20$.

Wie verändert sich das Konvergenzverhalten, wenn Sie für alle Fließzahlen im Programm die einfache Genauigkeit verwenden?

Welche Schlussfolgerungen ziehen Sie aus den Beobachtungen?

1.2 Wertebereich von Zahlen

Schreiben Sie ein Programm, das den Wertebereich verschiedenen Zahlentypen (Integer, Real,...) auf dem Bildschirm (Standardausgabe) ausgibt (Siehe Folien der Vorlesung; `range,precision` Fortran Kommandos).

(1) Geben Sie die Werte von `selected_int_kind()` und `selected_real_kind` für verschiedene Argumente der Funktionen aus.
(2) Welcher Zahlenwert beschreibt die intrinsische Genauigkeit des Systems.
(3) Wie viele verschiedene Ganzzahl bzw. Fließzahldatentypen stehen zur Verfügung?

Beispiel für Anforderung: Fließzahl mit 12 Nachkommastellen und einen Wertebereich von $10^{-60}, .., 10^{60}$ abdecken. Ist dies in Fortran möglich? (Hinweis: `Select_real_kind()` .)

1.3 Auswertung Arithmetischer Operationen

Das nachfolgende Programm gibt numerische Werte einer Berechnung aus. Lesen Sie das Programm sorgfältig durch.

```
program calc
  implicit none
  print *," Werte:", 353/7/7*2
  print *," Werte:", 353./7/7*2
  print *," Werte:", 353/7/7.*2

end program calc
```

Aufgabe: Zunächst Überlegen – dann Programm laufen lassen!!
(1) Verdeutlichen Sie durch Setzen von Klammern, welche Berechnung durchgeführt wird.
(2) Ist das Endergebnis immer gleich?

Kopieren Sie den Quellcode in eine Datei und übersetzen Sie das Programm. Ist Ihre Vermutung zur vorherigen Frage (Endergebnis) richtig? Wie lautet das richtige Ergebnis?

2 Anwendungsbeispiel

2.1 Zufallszahlen

Unter Fortran gibt es die intrinsische Unterroutine `call random_number(argument)`, die "Pseudo"- Zufallszahlen erzeugt und über die Variable `argument` zurückgibt. Die Unterroutine kann im Argument mit Skalaren oder Feldvariablen aufgerufen werden. Sie liefert gleichverteilte Zufallszahlen im Bereich $[0, \ldots, 1]$.

(1) Schreiben Sie ein Programm, das `N` Zufallszahlen in eine Datei ausgibt. Die Anzahl der Zufallszahlen soll ein Eingabeparameter des Programms sein (Hinweis: Schleife oder Felder verwenden).

Visualisieren Sie die Zufallszahlen mit `gnuplot`.

(2) Schreiben Sie ein Programm, das pro Zeile jeweils zwei Zufallszahlen ausgibt. Tragen Sie nun Spalte 1 gegen Spalte 2 auf. Was beobachten Sie?

(3) Führen Sie das/die Programme mehrfach aus und verwenden Sie jeweils einen anderen Namen für die Ergebnisdatei. Vergleichen Sie den Inhalt der Datei.

Der zum Fortran Sprachumfang gehörende Zufallsgenerator kann mit Hilfe der Unterroutine `call RANDOM_SEED([seed=feld] [, andere optionale Argumente])` initialisiert werden. Informationen zur Verwendung der Initialisierung finden Sie unter `https://gcc.gnu.org/onlinedocs/gcc-4.5.4/gfortran/RANDOM_005fSEED.html`.

Bei gleicher Initialisierung wird die gleiche Pseudozufallzahlenreihe generiert. Wieso ist dies sinnvoll? Was müssen Sie bei der Verwendung von Zufallszahlen sicherstellen?

2.2 Diffusion: Random Walk

Betrachten Sie ein Teilchen, das im Ursprung $x_0 = 0$ liegt. Im eindimensionalen Fall gilt folgende Regel für die Änderung der Position x_n:

$$x_{n+1} = x_n + s \quad \text{wobei} \quad x_0 = 0 \quad \text{und} \quad P(s = 1) = P(s = -1) = 0,5$$

wobei n den n-ten Sprung bezeichnet und P ist die Wahrscheinlichkeit für die Richtung bezeichnet. Beim Random-Walk sind die beiden Wahrscheinlichkeiten gleich.

Aufgabe: Schreiben Sie ein Programm, welches Ihnen die Trajektorie des Teilchen für $nmax$ Sprünge berechnet und in eine Datei ausgibt.

Sie können das Programm für eine bzw. zwei Dimensionen schreiben.

Zur Auswertung der Wahrscheinlichkeit des Sprungs P benötigen Sie einen Zufallsgenerator. Hierzu stellt Fortran die Unterroutine `call Random_number(val)` zur Verfügung. Die Unterroutine weist dem Argument val Zufallszahlen im Wertebereich $[0, 1]$ zu. val kann ein Skalar oder ein Feld sein. Bei Feldern werden allen Komponenten Zufallszahlen zugewiesen.

Betrachten Sie die Trajektorie des Teilchens mit Hilfe von z.B. `gnuplot`.

Aufgabe: Berechnen sie

- die mittlere Auslenkung $\langle \Delta x \rangle$ als Funktion der Schritte (Zeit)
- die mittlere quadratische Auslenkung $\left\langle (x - < x >)^2 \right\rangle$ als Funktion der Schritte (Zeit)

und tragen Sie die beiden Größen gegen die Schrittzahl auf. Was beobachten Sie? Welche Funktion beschreibt den Verlauf?

Berechnen Sie die Größen für 1000 Teilchen.

3 Numerische Integration der Sinus-Funktion

3.1 Numerische Quadratur mit der Mittelpunktsregel

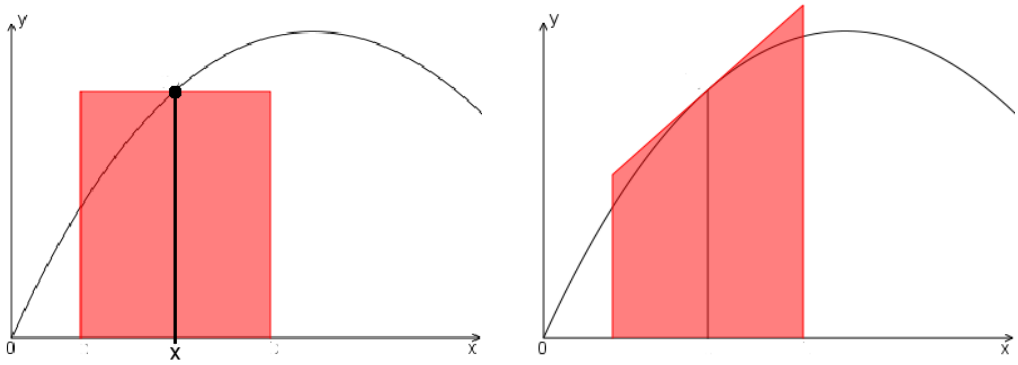

Figure 1: Schematische Darstellung der Mittelpunktsregel (links) und der Trapezregel (rechts)

① Approximieren Sie das Integral der $sin(x)$-Funktion durch eine Folge von konstanten Funktionen (siehe Fig. 1, links):
es sei x_i eine Stützstelle und die sin-Funktion wird im Intervall $[x_i - \frac{\Delta x}{2}, x_i + \frac{\Delta x}{2}]$ als konstant (nämlich $sin(x_i)$) angenommen werden. Daraus können sie schrittweise die Fläche der Kurve zusammensetzen.

$$F(x_i) = f(x_i) \cdot \Delta x \quad \text{f.a.} \quad x \in [x_i - \frac{\Delta x}{2} \ldots x_i + \frac{\Delta x}{2}]$$

② Schreiben Sie die Daten in eine Datei mit den drei Spalten x, $sin(x)$ und $\int sin(x)$ und plotten Sie diese.

③ Überprüfen Sie das Resultat, indem Sie als Anzahl der Halbwellen '2' eingeben.

④ Schreiben Sie eine 4 Spalte mit der analytischen Lösung für jede Stelle x_i und vergleichen Sie die Genauigkeit mit der numerischen Approximation. Was passiert, wenn Sie die Anzahl der Stützstellen erhöhen?

3.2 Numerische Quadratur mit der Trapezsregel

① Trapez-Regel:
Approximieren Sie das Integral der $sin(x)$-Funktion an durch eine Folge von linearen Funktionen (siehe Fig. 1, rechts).

$$F(x_i) = \frac{f(x_i + \frac{\Delta x}{2}) + f(x_i - \frac{\Delta x}{2})}{2} \cdot \Delta x$$

② Vergleichen Sie den absoluten Fehler der beiden Integrationsmethoden in einem Plot

Fortgeschrittene Lösung: In den obigen Aufgaben können Sie das Integrationsverfahren in einer Unterroutine realisieren und die zu integrierende Funktion in einer Unterfunktion definieren und diese der Integrationsroutine übergeben.