Institut für Numerische und Angewandte Mathematik
FB Mathematik und Informatik der Universität Münster
Prof. Dr. Christian Engwer, Dipl. Math. Dipl. Inf. Sebastian Westerheide

17.04.2012

Übung zur Vorlesung

Wissenschaftliches Rechnen

SS 2012 — Blatt 1

Abgabe: 24.04.2012, 10:00 Uhr, Briefkasten 89
Code zusätzlich per e-mail an `sebastian.westerheide@uni-muenster.de`

Achtung: Achten Sie darauf, ihre Programme ordentlich zu formatieren und gut zu kommentieren. Die Form wird mit in die Bewertung eingehen.

Aufgabe 1 (Nullstellenbestimmung mittels Bisektion) (4 Punkte)

Nach dem Zwischenwertsetz besitzt eine stetige Funktion $f : [a, b] \to \mathbb{R}$ mit $f(a) \cdot f(b) < 0$ (d.h. $f(a) < 0$, $f(b) > 0$ oder $f(a) > 0$, $f(b) < 0$) mindestens eine Nullstelle in $[a, b]$. Anschaulich muss der Graph von f mindestens einmal die x-Achse kreuzen, da er wegen der Stetigkeit keine Sprünge macht. Der Beweis des Zwischenwertsatzes liefert ein einfaches numerisches Verfahren zur Bestimmung von Nullstellen stetiger Funktionen, das Bisektionsverfahren. In Pseudocode lässt es sich folgendermaßen formulieren:

```
x1 := a; x2 := b;
while abs(x2 - x1) > eps do
  m := (x1 + x2) / 2;
  if f(m) * f(x1) >= 0 then
    x1 := m;
  else
    x2 := m;
  end;
end;
return (x1 + x2) / 2;
```

Zu einer gegebenen Fehlerschranke `eps` liefert es bis auf einen maximalen Fehler $\pm\frac{\texttt{eps}}{2}$ eine beliebige Nullstelle in $[a, b]$.

(a) Implementieren Sie das Bisektionsverfahren in C++.

- Benutzen Sie folgendes Interface `Function` für Funktionen $f : \mathbb{R} \to \mathbb{R}$

```cpp
class Function {
public:
  virtual double evaluate (double x) const = 0;
};
```

- Schreiben Sie eine Prozedur

```
double bisection_method (const Function& f, double a, double b,
                         double eps);
```

(b) Testen Sie Ihre Implementierung mit `eps` $= 10^{-8}$ an folgenden Funktionen:

- $f_1(x) = x^2 - 1$ auf $[0.6, 1.2]$ (Nullstelle: $x = 1$)
- $f_2(x) = \cos(x) - x$ auf $[0, 1]$ (Nullstelle: $x = 0.739085133215$)

Leiten Sie dazu jeweils mittels Vererbung eine Implementierung der Funktion vom Interface `Function` ab. Schreiben Sie ein Hauptprogramm, das Instanzen der Funktionen erzeugt, jeweils den Bisektionsalgorithmus aufruft und ermittelte Nullstellen sowie zugehörige Funktionswerte auf der Konsole ausgibt.

(c) Bestimmen Sie mit Ihrer Implementierung ferner eine Nullstelle der Funktionen

- $f_3(x) = \sin(2x) \cdot x$ auf $[-2, 7]$
- $f_4(x) = x^2 - 2$ auf $[-1.5, 1.5]$

Aufgabe 2 (Mathematisches Pendel) (4 Punkte)

In der Vorlesung haben wir ein mathematisches Modell für ein Pendel hergeleitet, das sogenannte mathematische Pendel (Kapitel 1.1.2). Dieses wird durch gewöhnliche, nichtlineare Differentialgleichung $\varphi''(t) + \frac{g}{l}\sin\big(\varphi(t)\big) = 0$ beschrieben, die sich für kleine Auslenkungswinkel $\varphi \ll 1$ durch die Näherung $\sin\varphi \approx \varphi$ weiter vereinfachen lässt. Dadurch ergibt sich folgendes lineare Anfangswertproblem:

$$\varphi''(t) + \frac{g}{l}\varphi(t) = 0 \qquad \text{auf} \qquad I := [0, \infty), \tag{1a}$$

$$\varphi(0) = \varphi_0, \qquad \varphi_0 \in \mathbb{R}, \tag{1b}$$

$$\varphi'(0) = v_0, \qquad v_0 \in \mathbb{R} \tag{1c}$$

(a) Bestimmen Sie die analytische Lösung des Anfangswertproblems (1).
Tipp: Bestimmung des Fundamentalsystems zu (1a)*, Darstellung der Lösung von* (1) *als Linearkombination, Einarbeiten der Anfangsbedingungen* (1b), (1c).

(b) Bestimmen Sie die mit Hilfe der analytischen Lösung die Periodendauer T, d.h. die Dauer einer kompletten Schwingung (hin und zurück).

(c) Bestimmen Sie für Anfangswerte $\varphi_0 \neq 0$, $v_0 \neq 0$ die Winkelgeschwindigkeit $\varphi'(t)$ beim Durchgang der Ruheposition und den maximalen Auslenkungswinkel $\hat{\varphi}$ (die Winkelamplitude). Dabei werden Sie folgende Formeln gebrauchen können:

- $\sin\big(\arctan(x)\big) = \frac{x}{\sqrt{1+x^2}}$
- $\cos\big(\arctan(x)\big) = \frac{1}{\sqrt{1+x^2}}$

Aufgabe 3 (Mathematisches Pendel, Computermodell 1) (6 Punkte)

$$\varphi''(t) + \frac{g}{l}\sin\big(\varphi(t)\big) = 0 \qquad \text{auf} \quad I := [0, T], \qquad T \in \mathbb{R}^+, \tag{2a}$$

$$\varphi(0) = \varphi_0, \qquad \varphi_0 \in \mathbb{R}, \tag{2b}$$

$$\varphi'(0) = v_0, \qquad v_0 \in \mathbb{R}. \tag{2c}$$

Wir wollen ein erstes Computermodell für das mathematische Pendel (2) herleiten und mit diesem eine Simulation durchführen. Mit dem Ansatz $\varphi'(t) = v(t)$ lässt sich die gewöhliche Differentialgleichung (2a) in ein System von zwei gewöhnlichen Differentialgleichungen 1. Ordnung transformieren:

$$\varphi'(t) = v(t), \quad v'(t) = -\frac{g}{l}\sin\big(\varphi(t)\big) \quad \text{auf} \quad I, \tag{3a}$$

$$\varphi(0) = \varphi_0, \qquad v(0) = v_0. \tag{3b}$$

(a) Diskretisieren Sie das System (3) mit dem expliziten Eulerverfahren, indem Sie das Verfahren auf beide Gleichungen getrennt anwenden.

(b) Implementieren Sie das resultierende Computermodell in C++.

- Realisieren Sie die Konstanten $g, l, T, \Delta t$ und φ_0, v_0 im Hauptprogramm durch
```
const double g = 9.80665;   const double l = 1.0;
const double T = ...;
const double dt = ...;
const double phi_0 = ...;
const double v_0 = ...;
```
- Lagern Sie die Iterationsvorschrift des expliziten Eulerverfahrens in eine Prozedur aus. Mit einem Interface `TimeFunction` für zeitabhängige Funktionen (analog zu Aufgabe 1) ist eine mögliche Signatur beispielsweise
```
double explicit_euler_step (double t_k, double y_k, double dt,
                            const TimeFunction& f);
```
- Lassen Sie sich t_k und den Auslenkungswinkel $\varphi_k \approx \varphi(t_k)$ in jedem Zeitschritt auf der Konsole ausgeben. Um φ_k später plotten zu können, soll die Ausgabe zeilenweise in folgendem Format erfolgen (wobei ␣ ein Leerzeichen darstellt): t_k ␣ φ_k

(c) Führen Sie mit Ihrem Programm eine Simulation durch und beschreiben Sie Ihre Beobachtungen:

- Wählen Sie die Anfangswerte $\varphi_0 = 0.5$, $v_0 = 0$ und die maximale Zeit $T = 5$
- Wählen Sie verschiedene Schrittweiten $\Delta t = 2^{-i}, \;\; i = 0, 1, 2, ...$
- Leiten Sie jeweils die Ausgabe Ihres Programms mit

```
./myprogramname > datafilename.txt
```

in eine Textdatei `datafilename.txt` um und plotten Sie das Ergebnis in `gnuplot` mit dem Befehl `plot "./datafilename.txt" smooth csplines`