D-MATH
Dr. Vasile Gradinaru
Marco Petrella

Serie 2

Abgabedatum: Di./Mi. 5.3/5.3
Koordinatoren: Marco Petrella, HG J 45, `marco.petrella@sam.math.ethz.ch`
Webpage: `http://metaphor.ethz.ch/x/2019/fs/401-1662-10L/`

1. *Konvergenzraten der summierten Quadraturregeln*

Wir wollen die folgende Quadraturregeln

- zusammengesetzte Mittelpunktregel
- zusammengesetzte Trapezregel
- zusammengesetzte Simpsonregel

verwenden, um das Integral

$$I = \int_0^1 f_i(x)\mathrm{d}x$$

von $f_i(x)$ auf N Teilintervallen oder mit n Funktionsauswertungen zu berechnen.
Die beiden Funktionen sind durch

$$f_1(x) := \frac{1}{1+5x^2} \qquad f_2(x) := \sqrt{x}\,. \tag{1}$$

gebeben.

a) Herunterladen Sie das Template `quadrature.py` und am Anfang des Files schreiben Sie die Funktionen `mittelpunkt(f,a,b,N)`, `trapezoid(f,a,b,N)` und `simpson(f,a,b,N)`. Dabei sollen die Funktion `f`, welche integriert werden soll, die untere und obene Grenzen `a` und `b` und die Anzahl der Teilintervalle in der zusammengesetzten Regel `N` eingegeben werden. Jede Funktion soll das Wert `I` ausgeben.

b) Schreiben Sie die Funktion `quadrature_errr(quadrature_rule, f, exact)`: dabei sollen die Quadraturregel `quadrature_rule`, die integrierende Funktion `f` und genaues Wert ihres Integrals `exact` eigegeben werden. Die Funktion `quadrature_errr` soll den Fehler `error` und die Anzahl Teilintervalle `n_chunks`$= 2^k$ wobei $k \in \{3, \dots, 10\}$ ausgegeben werden. **Tipp**: Die Funktion `enumerate` kann nützlich sein.

c) Implementieren Sie die Funktionen $f_i(x)$, $i \in \{1, 2\}$.

d) Unkommentieren Sie die Funktionen `plot_convergence(n_evals, errors, labels, title)` und `convergence_experiment(f, exact, title, filename)`. Vollständingen Sie dann die Funktion `convergence_experiment`: dabei sind `errors_l` und `n_l` den Fehler und die Anzahl der Funktionsauswertungen der Quadraturregel `l`, wobei `l`$\in \{$`mp`, `tr`, `si`$\}$.

Bitte wenden!

e) Unkommentieren Sie den letzten Teil des Templates und testen Sie den Code: zwei Plot sollen generiert werden, die die Konvergenzraten zeigen. Welche Methode verwendet man sinnvollerweise?

f) Ändern Sie den Code, um ein anderes Plot zu generieren, das die Konvergenz jeder Quadraturregel unter Berechnung des Integrals der Funktion

$$f(x) = x^5(1 - x^4) \tag{2}$$

zeigt.

Siehe nächstes Blatt!

2. *Homogen geladenes Quadrat in kartesischen Koordinaten*

Betrachten Sie ein quadratisches Gebiet in der x-y-Ebene welches eine konstante elektrische Ladungsdichte ρ_0 aufweist

$$\rho(x,y) = \begin{cases} \rho_0, & (x,y) \in [-1,1]^2 \\ 0, & \text{sonst.} \end{cases}$$

Das elektrostatische Potential φ an einem Punkt (x_p, y_p) ausserhalb des geladenen Quadrats ist dann durch Integration über die geladene Region gegeben

$$\varphi(x_p, y_p) = \frac{\rho_0}{4\pi\varepsilon_0} \int_{-1}^{+1} \int_{-1}^{+1} \frac{1}{\sqrt{(x-x_p)^2 + (y-y_p)^2}} \mathrm{d}x\mathrm{d}y \,.$$

Der Einfachheit halber setzen Sie $\frac{\rho_0}{4\pi\varepsilon_0} = 1$.

Implementieren Sie die Trapez- und die Simpson-Regel in zwei Dimensionen und berechnen Sie dann $\varphi(x_p, y_p)$ für $x_p = y_p = 2, 10, 20$. Schauen Sie sich den Fehler genau an. Was ist erstaunlich daran? Wie erklären sie sich dieses Verhalten?

Hinweis: Verwenden Sie das Template `potential.py`

Bitte wenden!

3. *Neues in* `Python`

In Python können Sie Funktionen als Funktionsargumente übergeben.

```
import numpy as np

def apply(f, x):
    return f(x)

def square(x):
    return x*x

x = np.random.random((3, 4))

apply(print, x)
print(apply(np.sin, x))
print(apply(lambda x: np.sum(x, axis=-1), x))
print(apply(square, x))
```

Einen Graph erstellen Sie wie folgt

```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(1e-5, 1.2, 10000)
fx = np.sin(2.0*np.pi/x)
plt.plot(x, fx, label="sin")
plt.legend()
plt.xlabel("x-Label")
plt.ylabel("y-Label")

plt.savefig("sin.png")
plt.savefig("sin.eps")
plt.show()
```

Siehe nächstes Blatt!