Prof. Dr. Dmitri Kuzmin
Dipl.-Technomath. Christopher Basting
Numerische Mathematik für Physiker und Ingenieure

Sommersemester 2016
Übungsblatt 4

Abgabe der **Theorieaufgaben bis Mittwoch, 25.05.2016, 16:15 Uhr** (einzeln oder zu zweit) in den Briefkasten Ihrer Übungsgruppe. Bitte vermerken Sie auf Ihrer Abgabe Ihre Übungsgruppe sowie Name und Matrikelnummer. Heften Sie mehrere Blätter zusammen.
Abgabe der **Programmieraufgaben bis Donnerstag, 26.05.2016, 16:15 Uhr** digital im Moodle-Arbeitsraum der Veranstaltung. Bei Abgabe zu zweit bitte nur einmal einreichen und im Kommentar den Namen und E-Mail Adresse des/der Koautors/Koautorin nennen. Sie können die Aufgaben in MATLAB bzw. OCTAVE schreiben. Alternativ kann auch in C oder C++ abgegeben werden, allerdings ohne mögliche Hilfestellung. Es kann nur lauffähiger Code bewertet werden.
Bei Fragen zu Vorlesung und Übungen können Sie neben der Übungsgruppe auch das **Tutorium** nutzen, jeweils **Mittwochs von 13:15 bis 14:45** Uhr im CIP-Pool der Mathematik (M946).

Aufgabe 4.1 (Zusammengesetzte Quadraturformeln | 5+3 Punkte)

a) Bestimmen Sie eine Näherung für

$$\ln 2 = \int_0^1 \frac{1}{1+x}\, dx$$

durch numerische Quadratur mit Hilfe der summierten Simpsonregel in einer Genauigkeit von $4 \cdot 10^{-5}$.

b) Wie viele Funktionsauswertungen sind erforderlich, um die gleiche Genauigkeit bei Verwendung der summierten Trapezregel garantieren zu können?

Aufgabe 4.2 (Gauss-Quadratur | 6+6 Punkte)

Bestimmen Sie Näherungen für folgende Integrale mit Hilfe der Gaußschen Quadraturformel mit 2 Punkten ($n = 1$). Überprüfen Sie mit Hilfe der Fehlerabschätzung der entsprechenden Quadraturformel, ob der Quadraturfehler geringer als $1/1000$ ist.

a) $\displaystyle\int_0^1 \cosh x \, \mathrm{d}x\,, \qquad \left(\cosh x := \frac{e^x + e^{-x}}{2}\right),$

b) $\displaystyle\int_2^3 \frac{1}{x} \, \mathrm{d}x\,.$

Programmieraufgabe 4.1 (Zusammengestzte Quadratur | 4 + 4 Punkte)

a) Implementieren Sie eine Funktion `[v]=myQuadraturSum1D(f,w,p,a,b,N)`, die eine zusammengestzte (summierte) Quadraturformel umsetzt. Dabei soll für die Eingabeargumente gelten:

- `f` ein `function_handle` der zu integrierenden Funktion;
- `w` ein Vektor der Dimension R welcher die Quadraturgewichte enthält;
- `p` ein Vektor der Dimension R welcher die Stützstellen der Quadraturformel auf dem Einheitsintervall enthält;
- `a` die untere Integrationsgrenze;
- `b` die obere Integrationsgrenze;
- `N` die Anzahl der Teilintervalle I_i, $i = 1, \dots, N$ mit $h = \frac{b-a}{N}$ und

$$I_i := [a + (i-1) \cdot h, a + i \cdot h].$$

Auf jedem Teilintervall soll dann die durch `w` und `p` spezifizierte Quadraturformel umgesetzt und somit ein Näherung für

$$\int_a^b f(x)\mathrm{d}x = \sum_{i=1}^N \int_{I_i} f(x)\mathrm{d}x$$

berechnet werden.

b) Testen Sie Ihre Implementierung für folgende Quadraturformeln:

- Trapezregel
- Simpson-Regel
- Milne-Regel

und für die numerische Integration der Runge-Funktion

$$f(x) := \frac{1}{1 + 25x^2}$$

aus Programmieraufgabe 2.1 auf dem Interval $[-1, 1]$. Erstellen Sie dazu einen aussagekräftigen Plot, welcher die drei Quadraturverfahren hinsichtlich der des Fehlers gegenüber der exakten Lösung für $n = 1, 2, \dots, 100$ vergleicht. Tragen Sie dabei den Fehler logarithmisch auf (mit dem Matlab-Befehl `semilogy`).
Speichern Sie Ihren Test in einem Skript `myQuadraturSum1DTest.m` und zusätzlich den durch das Skript erstellten Plot als `myQuadraturSum1DPlot.fig`.