Technische Universität München
Lehrstuhl für Informatik V
Dr. T. Neckel, Dr. D. Pflüger

SS 2011
Sheet 4
1.6.2011

Algorithms of Scientific Computing (Algorithmen des Wissenschaftlichen Rechnens)

Numerical Quadrature for One-dimensional Functions

This week we focus on the question how to determine the definite integral

$$F(f,a,b) = \int_a^b f(x)\,dx \quad \text{for functions } f : [a,b] \to \mathbb{R}.$$

It is not always possible to integrate the function f analytically, and so we use numerical quadrature. For the implementation of our algorithms we will use the Python language. The file *quadrature1d.py* already contains function templates and a program skeleton. Copy it to the same directory as *plotting.py*, edit it and run it to test your programs.

Excercise 1: Analytical Integration

Let

$$f(x) = -4\,x\,(x-1) \tag{1}$$

$$g(x) = \frac{8}{9}\cdot(-16\,x^4+40\,x^3-35\,x^2+11\,x) \tag{2}$$

the functions under consideration. Compute the antiderivatives and evaluate the integrals.

Hint: If not specified otherwise, the domain considered from now on is the unit interval $\Omega = [0,1]$.

Excercise 2: Composite Trapezoidal Rule

Write a function that approximates the integral via the Composite Trapezoidal Rule. Open the file *quadrature1d.py* and fill in the function template

```
def trapezoidalRule(f, a, b, n):
    '''
    Implements CTR to approximate the integral of given function.
    '''
    result = 0.0
    return result
```

with a,b and f according to the definition above and n being the number of trapezoids used.

Excercise 3: Composite Simpson Rule

Do the same as in Exercise 2, only use the Composite Simpson Rule now.

```
def simpsonRule(f, a, b, n):
    '''
    Implements CSR to approximate the integral of given function.
    '''
    result = 0.0
    return result
```

Excercise 4: Archimedes' Hierarchical Approach

In this exercise we will use Archimedes' approach to approximate the integral.

Let $\vec{u} = [u_0, \ldots, u_n]^T \in \mathbb{R}^n, n = 2^l - 1, l \in \mathbb{N}$ a vector of function values with $u_i = f(x_i = \frac{i+1}{2^l})$.

a) Write a function that transforms a given vector $\vec{u} \in \mathbb{R}^n$ to a similar vector $\vec{v} \in \mathbb{R}^n$ containing the hierarchical coefficients needed for Archimedes' quadrature approach. Use the function template from the file *main.py*

```
def hierarchize1d(u, l):
    '''
    Transforms the (2^l-1) func. values in u to hier. coeffs
    '''
    # you can also compute "in place" and return a modified u
    v = u[:]
    return v
```

Hint: Later in the lecture we will officially call this process "hierarchization".

b) Having computed the vector $\vec{v}$ with the hierarchical coefficients, implement a function evaluating the integral.

```
def archimedesRule(v, l):
    '''
    Implements Archimedes' approach to approximate the integral.
    '''
    result = 0.0
    return result
```

c) Write a function "dehierarchize1d" similar to "hierarchize1d" that computes the inverse of the transformation above.

Excercise 5: Thoughts about Adaptivity

Discuss how the previous methods could be extended in order to improve their approximation quality.