Technische Universität München
Lehrstuhl für Informatik V
Univ.-Prof. Dr. M. Bader
M.Sc. Kilian Röhner, M.Sc. D. Jarema

SS 2015
Sheet 5
20.5.2015

Algorithms of Scientific Computing (Algorithmen des Wissenschaftlichen Rechnens)

Archimedes Quadrature and Haar Wavelets

Excercise 1: Archimedes' Hierarchical Approach

Again we focus on the question how to determine the definite integral

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

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. **Hint:** Later in the lecture we will officially call this process "hierarchization", thus the function name.

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

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

Excercise 2: Thoughts about Adaptivity

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

Excercise 3: The Haar Wavelet Basis

We derive the *mother wavelet* ψ as well as *orthonormal wavelet bases* $\{\ \psi_{l,m}\ \}$ with

$$\begin{aligned} \psi_{l,k}(t) &= \psi(2^l\, t - k) \\ span\{\ \psi_{l,k}\ \} &= W_l,\ < \psi_{l,k}, \psi_{l,m} >= \delta_{k,m}\ \ k,m \in \mathbb{Z}. \end{aligned} \tag{1}$$

In this exercise we want to compute the 1-d wavelet transform for the Haar wavelet family and apply it to a signal vector $\vec{s}$ of length $m = 2^n$. The transform can be implemented very efficiently as a "pyramidal algorithm" taking $\mathcal{O}(m)$ steps. For educational purpose we focus on the $\mathcal{O}(m^2)$ matrix-based algorithm.

(i) Write a function that constructs the transformation matrix M consisting of the basis vectors $\psi_{l,k},\ l \leq n,\ 0 \leq k \leq 2^n - 1$.

(ii) Use Python's package *numpy.linalg* to invert the matrix.

(iii) Use the program to compute the transform $M\vec{s} = \vec{d}$ as well as the reconstructed signal $M^{-1}\vec{d} = \vec{s}'$ of the vector

$$\vec{s} = [1, 2, 3, -1, 1, -4, -2, 4]^T$$

(iv) Verify the program's output tracing the steps by hand.