Numerical Algorithms for Visual Computing II

PD Dr. Michael Breuß and M.Sc. Kai Uwe Hagenburg

Released: 31.10.2008
Assigned to: Tutorial at 13.11.2008

Assignment 2 (4 Exercises) – BasicBasics™ of Schemes

Exercise No. 1 – Explicit coding

Consider the 1-D Laplace equation, see (2.10), over the open domain $\Omega = (0, 10)$ together with the boundary conditions

$$u(0) = 1\ , \quad u(10) = 4\,.$$

For computing the numerical solution of the arising BVP, consider a grid with the points

$$j, \quad j = 0, \ldots, 10,$$

and

$$u_0 = u(0)\ , \quad u_{10} = u(10)\,,$$

i.e., $h := \Delta x = 1$.

(a) Write a program solving the BVP by employing the method of artificial time. Use the discretisation (2.6) from the lecture as a basis for the spatial discretisation. Derive an explicit time-marching scheme. **(6 pts)**

(b) Write a program solving the BVP by means of Jacobi-iterations. For this, specify useful parameters for the scheme from (a). **(2 pts)**

Choose as initial states for the implementations point values of the function

$$f(x) \;=\; \begin{cases} 2x+1 & : \quad x \leq 5 \\ -\dfrac{7}{5}x + 18 & : \quad x > 5 \end{cases} \;.$$

Plot the initial state together with the results after 1, 10 and 20 iterations.

Hint. The derivation of schemes should always be done using pen and paper.

Exercise No. 2 – Direct coding

Write a program solving the elliptic BVP directly. Write down the arising linear system before coding.

Hint. You may use the Thomas algorithm, see e.g.
$http : //en.wikipedia.org/wiki/Tridiagonal\_matrix\_algorithm$ **(6 pts)**

Exercise No. 3 – Implicit coding

Consider the same problem as in Exercise No. 2, but employ an implicit time marching scheme this time.

(a) Write down the arising linear system. **(4 pts)**

(b) Compare the explicit and the implicit scheme. For this, experiment with the time step size (coding!). Give a conclusion on the experiments. **(4 pts)**

Exercise No. 4 – Jacobi strikes back

Consider again the one-dimensional Laplace equation.

(a) Derive a discretisation of u_{xx} by use of backward differences, see (2.4). **(2 pts)**

(b) Derive a Jacobi-iteration scheme from the new formula from (a). **(2 pts)**

(c) Discuss the usefulness of the new iterative scheme from (b). Are there problems with the application of the new scheme? (You don't need to code the method to give an answer, however, it may help if you don't have a clue.) **(4 pts)**