MA2501 Numerical Methods

Assignment 11

Tutorial: 4/5.

Exercise 1

Given the initial value problem

$$x'''(t) = e^{x(t)}$$

$$x(0) = 1, \;\; x'(0) = 1, \;\; x''(0) = 1.$$

a) Write this as a system of first order differential equations.

b) Find an approximation to $x(0.1)$ and $x(0.2)$ by using an 2. order Runge-Kutta method. Use the stepsize $h = 0.1$.

Oppgave 2

The solution of the partial differential equation (Poisson's equation)

$$u_{xx} + u_{yy} = -1$$

in the domain D, where $u(x, y)$ is given at the boundary of D, are to be approximated by a difference method. The Domain D is given by

$$D = \{(x, y) \mid 0 < x < 1, 0 < y < 1 - x\} \,,$$

$u(x, y) = 1$ on the boundary. Use the stepsize $h = 0.25$, and let u_{ij} be the approximation to $u(i \cdot h, j \cdot h)$. See the figure on the next page.

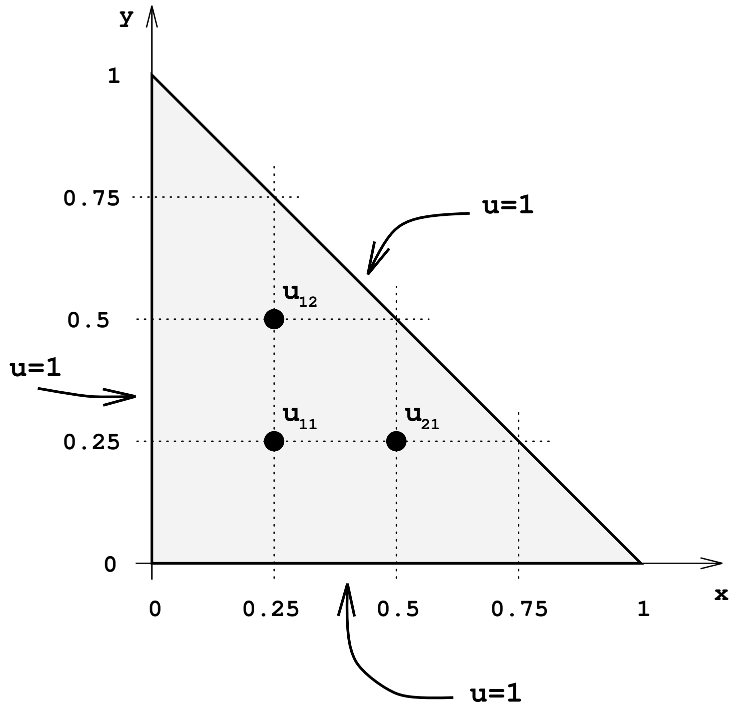
a) Find the three equations which is needed to find the approximations u_{11}, u_{12} and u_{21} when a central difference scheme is used to approximate the derivatives.

The system of equations in a) can be written as

$$A\vec{u} = \vec{b}$$

where A is a 3×3 matrix, and $\vec{u}$ and $\vec{b}$ are vectors.

b) Let $\vec{u} = (u_{11}, u_{21}, u_{12})^T$. Find A and $\vec{b}$. Solve the system for u_{11}, u_{21} and u_{12}.



Oppgave 3

Solve Burgers' equation

$$\begin{cases} u_t + \left(\dfrac{u^2}{2}\right)_x & = & \mu u_{xx} \\ \\ u(x,0) & = & 1.5 \cdot x \cdot (1-x)^2 \\ u(0,t) & = & u(1,t) = 0 \end{cases}$$

on $0 \leq x \leq 1$ and $0 \leq t \leq 6$.

Use the discretization

$$\frac{u_{i,j+1} - u_{i,j}}{k} + \frac{u^2_{i+1,j+1} - u^2_{i-1,j+1}}{4h} = \mu \frac{u_{i+1,j+1} - 2u_{i,j+1} + u_{i-1,j+1}}{h^2}, \qquad (1)$$
$$i = 1, 2, \cdots, n, \quad j = 0, 1, \cdots, m$$

where $u_{i,j} \approx u(x_i, t_j)$, $x_i = i \cdot h$ and $t_j = j \cdot k$. k and h are the step lengths in the t and x-direction respectively, and $n = 1.0/h$, $m = 6.0/k$. Remember $u_{0,j}$, $u_{n,j}$ and $u_{i,0}$ are given by the boundary and the initial value conditions.

The discretization is implicit, thus a nonlinear system of equations has to be solved with respect to $u_{i,j+1}$, $i = 1, \cdots, n-1$ for each timestep. This can be solved by Newtons method, using the values from the previous step as starting values for the iterations.

Make a MATLAB-script solving Burgers' equation by the discretization given in (1), and plot the solution. But before starting to do so, so

a) make sure you are certain what the discretization (1) means,

b) write up the nonlinear system to be solved for each step. In particular, what is the Jacobi-matrix for the equation.

Use $h = k = 0.01$ and $\mu = 0.001$, and start to program. If you like, you can use the skeleton on the next page:

Hint: In MATLAB it is usually smart to work directly on vectors or matrices. Thus

```
u(2:n-1) = p*(u(1:n-2)-2*u(2:n-1)+u(3:n)));
```

is exactly the same as

```
for i=2:n-1
  u(i) = p*(u(i-1)-2*u(i)+u(i+1));
end;
```

but the first alternative is much faster.

You might also like to make an animation of the solution.

```
% Preparation
%-------------
h=0.01;      % Stepsizes in x- and t- direction.
k=0.01;
xend = 1;
tend = 6;
n = xend/h+1;   % The number of discretization points in x- and t-
m = tend/k+1;   % direction respectively, including the boundary values.
mu=1.e-3
u = zeros(n,m); % A matrix for the solution

x=[0:h:xend]';
t=[0:k:tend];
u(:,1) = 1.5*x.*(1-x).^2; % Put the initial values in the first column
                          % of the solution matrix.

% Hovedloekke
%-------------
for j=1:m-1
  u(:,j+1) = u(:,j); %  The starting values for the Newton
                     %  iterations.
  du = ones(n-2,1);

  % Newton-iterasjoner.
  while norm(du,2) > 1.e-4
%=====================================================================
%    This you have to do yourselv. You should
%     Evaluate the function f(u):
%     Evaluate the jacobi-matrix J :
%     Let: du = -J\f (solve the system) and let u(2:n-1,j+1) =
%          u(2:n-1,j+1) + du;
%=====================================================================
  end;
end;

% Plot the solution
mesh(x,t,u');
view( -15,35);
xlabel('x');
ylabel('t');
```