446-1
Numerical Solution of Partial Differential Equations
Fall 2000
Hermann Riecke

Project 2

Due December 4, 2000

1. **Burgers Equation with Diffusion**

   Burgers equation describes the dynamics of nonlinear waves for which the wave speed depends on the amplitude of the wave. With an additional diffusion term the equation is given by
   $$\partial_t u = 2\gamma u \partial_x u + D\partial_x^2 u. \tag{1}$$
   Because of the shocks that can occur for $D = 0$ it is better to write the equation in conservation form,
   $$\partial_t u = \partial_x \left(\gamma u^2 + D\partial_x u\right). \tag{2}$$
   Solve (2) numerically with Dirichlet boundary conditions, $u(0,t) = 0$ and $u(L,t) = 0$. Use two different schemes: a Crank-Nicholson scheme of order (2,2) and a (2,2)-MacCormack scheme[1].

   For the Crank-Nicholson treat the nonlinear term by expanding it around the solution at time step t_n, i.e. write $u^{(n+1)} = u^{(n)} + \left\{u^{(n+1)} - u^{(n)}\right\}$ and expand in $u^{(n+1)} - u^{(n)}$. This scheme will lead to a tridiagonal matrix problem. Within Matlab this is solved without too much difficulty. Make sure you make use of the sparseness of the matrix. Below is a short list of commands and functions you may need for the Matlab code. You probably will have to read in the manual or the online help as well.

   For the MacCormack, there is no need to alternate between the FB- and the BF-version, since the coefficients in the differential equation (2) are constant.

   (a) Test the linear part of the Crank-Nicholson scheme and of the MacCormack scheme by performing a convergence study and compare the numerical results to the exact solution[2] of the diffusion equation
   $$\partial_t u = \partial_x \left(\partial_x u\right) \tag{3}$$
   with initial condition $u(x,0) = \sin(x/2)$ on the interval $[0, 2\pi]$. Measure the error of the numerical solution at $x = \pi$ and $t = 0.95$ for a sequence of values of Δt and of Δx

[1] It is in general good practice to write codes in a modular fashion and provide a control menu to change the parameters. For illustration purposes there is a simple program on the class web site that solve the Kuramoto-Sivashinsky equation with forward Euler (see below).

[2] To calculate the exact solution you could use a Fourier ansatz in space and time.

which demonstrates that the code in fact achieves the expected convergence rates. (Use suitable double-logarithmic plots).
Study numerically also the stability limits of both schemes. How do they depend on Δx? How does this compare with your expectations?

(b) Solve now the full equation (2) using the Crank-Nicholson and the MacCormack scheme. Choose as initial condition

$$u(x,0) = -2\sin(x) + \sin(x/2). \tag{4}$$

Attempt to resolve the solution graphically for $D = 0.001$, $\gamma = 1$ at the times $t = 0.25$ as well as for $t = 0.3$ and $t = 1$. Comment on the solution. What happens? What makes the resolution of the solution difficult?
vergleiche Verfahren. explizit verlangen

2. **Kuramoto-Sivashinsky Equation**

The Kuramoto-Sivashinsky equation describes among other things flame fronts and the evolution of the wavenumber of a traveling wave. It is given by

$$\partial_t u = D\partial_x^2 u - \partial_x^4 u + u\partial_x u \tag{5}$$

and can be considered to be regularization of the Burgers equation for the case $D < 0$. Use periodic boundary conditions.

(a) Solve (5) using a simple (1,2) forward Euler scheme. Also implement a (2,2) Crank-Nicholson scheme (with linearization of the nonlinearity as in the Burgers equation). Note that the periodic boundary conditions lead to additional entries in the matrix compared to the Dirichlet boundary conditions used before. for (5).

(b) For $D = -0.4$ and a system length $L = 10$ use both schemes to determine $u(x = 2, t = 30)$ with 10% and with 1% error. As initial condition use $u(x,0) = 0.5\,\sin(2\pi\, x/L)$. Which scheme is more efficient in the two cases? Can you reasonably also obtain 0.1% accuracy with both schemes?

(c) To get an impression of the dynamics that are possible in the Kuramoto-Sivashinsky equation use the scheme of your choice to solve (5) with random initial conditions for $L = 20\pi$ and $D = -0.6$. Evolve the equation at least till $t_{max} = 100$.

Various commands and functions in Matlab
Taking transposes of matrices and vectors: $Y = y'$
shifted vectore for derivatives: $yp = [y(2:nx)\ y(1)]$
vector of ones: $e = ones(nx,1)$
vector of zeroes: $z = zeros(nx,1)$
tridiagonal matrix with rows ....1, −2, 1...: $tridiag = spdiags([e\ \ -2*e\ e], -1:1, nx, nx)$
the function spdiag creates a matrix in sparse form, i.e. makes use of the sparseness. For comparison (i.e. with respect to speed) you can also try the full matrix version.
identity matrix: $Id = spdiags([e], 0, nx, nx)$

matrix with 1 in an off-diagonal; $spdiag([e], j, nx, nx)$ with $-nx+1 < j < nx-1$.
square each entry in the vector Y: $S = Y. * Y$
to solve the matrix equation $Ax = b$: $x = A\backslash b$
where x and b are column vectors (obtained from row vectors by transposition).