MATH 517 PROJECT 1: BURGERS' EQUATION, SHOCK AND RAREFACTION

Turn in on Blackboard by the end of Monday 9/8.

Goal: Solve the Burgers equation $u_t = -uu_x$ numerically, using the initial conditions from the written homework:

$$u(x,0) = \begin{cases} 4, & -3 < x < 3, \\ 0 & \text{otherwise} \end{cases}$$

Recommended software: Matlab (available in SU computer clusters) or its free alternative Scilab (available for download from `www.scilab.org`).

Answer the following questions, based on output:

(1) When (approximately) does the rarefaction wave catches up with the shock wave?
(2) What happens to the maximum of function u after the catch-up moment?
(3) What (approximately) is the speed of shock propagation initially and at the end of computation?

You can include the answers as comments when submitting the file on Blackboard.

RELEVANT MATLAB / SCILAB INFORMATION

Commands can be entered directly into command window, in which case they are executed one by one. You can end a command with semicolon ; to suppress its output. Usually, a sequence of commands is put together into a script, created with the built-in editor (from which the script can be executed). See the video tutorials posted on Blackboard.

`x = a:h:b` creates a row vector (array of numbers) whose components begin with a and increase in multiples of h, until they exceed b. For example, `x = 1:3:11` assigns the vector `[1,4,7,10]` to x.

`u = zeros(m,n)` creates a matrix with m rows and n columns, filled with zeros.

`u(i,:)` returns the row number i of the matrix. Similarly, `u(:,j)` is the column number j.

`plot(x,y)` plots x against y, where x and y must be vectors of the same size.

Sample program. First, set parameters:

```
a = -5;                -- left end of space interval
b = 25;               -- right end of space interval
t = 15;                -- end of time interval (it begins with 0)
timeStep = 0.01;      -- time resolution
spaceStep = 0.1;      --  space resolution
```

Calculate the size of matrix required to hold function u, and create the matrix:

```
numTimeSteps = t/timeStep + 1;
numSpaceSteps = (b-a)/spaceStep + 1;
u = zeros(numTimeSteps, numSpaceSteps);
```

Impose the initial and boundary conditions (boundary: 0 velocity at the left edge).

```
for j = ((-3-a)/spaceStep) : ((3-a)/spaceStep)
    u(1,j) = 4;
end


for i = 1 : numTimeSteps
    u(i,1) = 0;
end
```

Run the double loop that calculates the solution using an appropriate difference scheme.

```
for i = 1 : numTimeSteps-1
    for j = 2 : numSpaceSteps
         u(i+1,j) =  ...  most interesting part is here (see the class notes)
    end
end
```

Plot the solution. To avoid clutter, the loop below plots every 50th time step. What is the time interval between the consecutive plots? You'll need this to answer the questions.

```
hold on                  .....     omit this if using Scilab
for i = 1 : numTimeSteps/50
    plot(a:spaceStep:b, u(1+50*(i-1),:))
    pause(.5)         .....       replace with xpause(.5e6)  if using Scilab
end
```