

Math 493: Homework 3a – Anderson – Summer 2009

DUE: THURSDAY, JUNE 11, 2009

1. Consider the one dimensional wave equation on the bounded spatial domain

$$\begin{aligned}\frac{\partial^2 u}{\partial t^2} &= \frac{\partial^2 u}{\partial x^2}, \\ u &= 0 \quad \text{on } x = 0, 1, \\ u &= f(x) \quad \text{on } t = 0, \\ \frac{\partial u}{\partial t} &= 0 \quad \text{on } t = 0,\end{aligned}$$

We shall be interested in the solution to this problem for $0 \leq x \leq 1$ and $0 \leq t \leq t_F$. You should examine numerical solutions of this problem for the three different functions $f(x)$ specified below.

The numerical code that you should implement is a method of lines approach in which the spatial interval is discretized into N equal intervals of width $h = 1/N$ (giving grid points $x_j = (j - 1)/N$ for $j = 1, \dots, N + 1$) and the integration in time for each of the unknowns in space is accomplished using an ODE solver for the resulting system of first order ODEs (I recommend Matlab's Runge-Kutta based `ode45`). In particular, the system of ODEs is

$$\begin{aligned}\frac{du_j}{dt} &= v_j, \quad \text{for } j = 2, \dots, N, \\ \frac{dv_j}{dt} &= (u_{j+1} - 2u_j + u_{j-1})/h^2, \quad \text{for } j = 2, \dots, N,\end{aligned}$$

and $u_1 = 0$, $u_{N+1} = 0$, $v_1 = 0$ and $v_{N+1} = 0$. Initial conditions are $u_j(t = 0) = f(x_j)$ and $v_j(t = 0) = 0$ for all j .

Note that you can set the tolerances for Matlab's `ode45` solver using the following 'options' definition:

```
RelTolVal=10(-8);
AbsTolVal=10(-8);
options = odeset ('RelTol',RelTolVal,'AbsTol',AbsTolVal);
%
[t,uv] = ode45 (@myfun, tspan, uv_init,options, parameter1, parameter2);
```

where `@myfun` is a Matlab function file `myfun.m` in which the right-hand sides of the above ODEs are defined, `tspan` is a vector containing the time values at which you would like the solution of the ODE to be returned (if you specify only a beginning and an ending value

Matlab will return the solution at all point at which it computes the solution), `uv_init` is a vector of initial conditions for $u_2, u_3, \dots, u_N, v_2, v_3, \dots, v_N$, `options` is as defined above, for example, and `parameter1`, `parameter2` are parameter values that you may wish to supply to the function `myfun.m`.

(a) Obtain a numerical solution approximation for the case $f(x) = \sin(\pi x)$ for $t_F = 1.25$. Note that in this case the exact solution to the PDE is given by $u_{exact}(x, t) = \cos(\pi t) \sin(\pi x)$ so the purpose of this is to verify that your numerical code can reproduce to some approximation the correct result. In particular, define a relative error as

$$e = \|\vec{u}^i - \vec{U}^i\|_2 / \|\vec{U}^i\|_2$$

where \vec{U}^i is the exact solution vector at the final time $t = t_F$ that has components $u_{exact}(x_j, t_F)$ for $j = 2, \dots, N$ and \vec{u}^i is the corresponding approximate solution vector at the final time obtained from your calculations (note that this vector only includes the u part of the solution and not the v part). These norms can be computed in Matlab using the `norm` command. Make a table of values of this error for $N = 25$, $N = 50$, $N = 100$, $N = 200$ and $N = 400$ and discuss your observations.

(b) Repeat a similar procedure using $t_F = 1.25$ for the case where

$$f(x) = \begin{cases} x & x < 1/2 \\ 1 - x & x \geq 1/2 \end{cases}$$

Here the exact solution (which can be obtained by separation of variables and superposition) is given by

$$u_{exact}(x, t) = \sum_{n=0}^{\infty} \frac{4(-1)^n}{\pi^2(2n+1)^2} \cos[(2n+1)\pi t] \sin[(2n+1)\pi x].$$

Examine the exact solution in Matlab by replacing the infinite sum with a finite one with sufficiently many terms. Note that since you know the solution at $t = 0$ you can obtain a reasonable idea of how many terms you'll need by comparing this truncated 'exact' solution with the initial function $f(x)$. Turn in a plot of the numerical solution based on the above method of lines algorithm and 'exact' solution at $t = 1.25$.

(c) Identify a numerical solution for the case

$$f(x) = \exp[-(x - x^*)^2/w]$$

for $x^* = 0.5$ and $w = 0.005$. Plot the numerical solution $u(x, t)$ at times $t = 0$, $t = 0.25$, $t = 0.5$, $t = 0.75$ and $t = 1$. Make sure that you use sufficiently resolved grids in space and time to be sure you are seeing a reasonable approximation.

ADDITIONAL COMMENTS: If you store the solution for a collection of time values and space values you can make a surface plot of the entire solution for all x and t values computed. See Matlab commands `meshgrid`, `surf`, `mesh` to do this.

The following is an excerpt from my code for this problem that plots the surface u (stored as a matrix) over the x and t domains. It is more than you need for this problem but it shows some clues for interesting plotting features in Matlab.

```
[XX,TT]=meshgrid(x,t);
surf(XX,TT,u);
xlabel('x')
ylabel('t')
shading interp
grid on
lightangle(-45,75)
set(gcf,'Renderer','zbuffer')
set(findobj(gca,'type','surface'),...
'FaceLighting','phong',...
'AmbientStrength',.3,'DiffuseStrength',.8,...
'SpecularStrength',.9,'SpecularExponent',25,...
'BackFaceLighting','unlit')
colormap([0.5 0.6 0.9])
view([60,45])
```

Here x is a vector of length $N + 1$, t is a vector of length $M + 1$ and u is a matrix of size $M + 1$ by $N + 1$ (whose rows contain the solution for a given time level).