

AMATH 301
Homework 4: Spring 2009

DUE: see website for exact time and date. No late assignments accepted.

- I Consider the following boundary value problem (which could describe an undamped harmonic oscillator with an external sinusoidal driving mechanism, with driving amplitude 12 and driving frequency $\omega = 1$)

$$\frac{d^2y(t)}{dt^2} + 4y(t) = 12\cos(t)$$

over the domain $t \in [0, 1]$ and corresponding boundary conditions

$$y(0) = 0,$$

$$y(1) = 4.$$

Solve this boundary value problem using the shooting method. Use bisection to iteratively find the value of $A = \frac{dy(0)}{dt}$ that will lead to the solution satisfying the boundary conditions. As your initial lower and upper limits for the bisection method use $\frac{dy(0)}{dt} = -2$ and $\frac{dy(0)}{dt} = 2$ if you are implementing the bisection similar to the code from lecture 3, if instead you use a code similar to lecture 16 then you will obtain the same results by using your initial value of $A = 2$ and your initial step $dA = 1$. Use ODE45 with the default settings (that is do not use `odeset`). Use a tolerance of 10^{-5} for the bisection method (the tolerance on the right boundary condition being met to break out of the search loop).

ANSWER: Your numerical solution value for $\frac{dy(0)}{dt}$ as A1.dat. Your numerical solution value for $y(1)$ as A2.dat. The values of your numerical solutions evaluated at $t = 0 : .01 : 1$ should be saved as a 101x1 column vector in A3.dat.

- II The deflection of a uniformly loaded, long rectangular plate under an axial tension force is governed by a second-order differential equation. Let S represent the axial forces and q the intensity of the uniform load. The deflection W along the elemental length is given by

$$\frac{d^2W}{dx^2} - \frac{S}{D}W = \frac{-ql}{2D} + \frac{q}{2D}x^2,$$

$$W(0) = W(l) = 0$$

The parameter l is the length of the plate and D is the flexural rigidity of the plate. Let $q = 100 \frac{\text{lb}}{\text{in}^2}$, $S = 80 \frac{\text{lb}}{\text{in}}$, $D = 8.8 \times 10^7 \frac{\text{lb}}{\text{in}}$, and $l = 50\text{in}$. The plate will damage if the maximal deflection value along the elemental length is above 0.2 inches.

1. Approximate the deflection at 1 inch intervals. To do this, solve the differential equation with the shooting method using a bisection solver and a tolerance of 10^{-6} . Use `ode45` to solve the differential equations. Find the maximal deflection value. Is it under the threshold? How far down the plate does the maximum deflection occur?

ANSWERS: Save the numerical solution as a 51x1 column vector in A4.dat. Save the maximal deflection value in A5.dat, and the inches down the plate in which this maximal deflection occurs in A6.dat.

2. Approximate the deflection at 1 inch intervals. To do this, use the *direct* approach. Find the maximal deflection value. Is it under the threshold? How far down the plate does the maximum deflection occur?

ANSWERS: Save the numerical solution as a 51x1 column vector in A7.dat. Save the maximal deflection value in A8.dat, and the inches down the plate in which this maximal deflection occurs in A9.dat.

- III Solve the following Poisson problem using finite differences. Consider the Poisson equation

$$-\left(\frac{d^2u(x,y)}{dx^2} + \frac{d^2u(x,y)}{dy^2}\right) = 1$$

on the domain $[0, 1] \times [0, 1]$ with boundary conditions $u(x^*, y^*) = 0$ for (x^*, y^*) lying on the boundary of the domain.

One way to solve this ODE is by using finite differences. The Laplacian term $\frac{d^2u}{dx^2} + \frac{d^2u}{dy^2}$ can be approximated using second order finite differences on a structured grid as

$$\frac{d^2u}{dx^2} + \frac{d^2u}{dy^2} \approx \frac{u(x + \Delta x, y) - 2u(x, y) + u(x - \Delta x, y)}{\Delta x^2} + \frac{u(x, y + \Delta y) - 2u(x, y) + u(x, y - \Delta y)}{\Delta y^2}$$

If further we use the finite difference approximation on a uniform grid where the domain $[0, 1] \times [0, 1]$ has been subdivided into $(n + 1) \times (n + 1)$ points (including the boundary points) we obtain a system of equations with $(n - 1)^2$ unknowns: $u(x_i, y_j)$, where $i = 1, \dots, (n - 1)$ and $j = 1, \dots, (n - 1)$.

This can easily be solved in MATLAB by writing it as a system of equations $\mathbf{A}\mathbf{u} = \mathbf{b}$ and solving for \mathbf{u} using the backslash command, where \mathbf{A} is the discretization matrix constructed using the finite difference approximation above. If we organize the unknown vector \mathbf{u} as follows:

$$\mathbf{u} = \begin{bmatrix} u_{11} \\ u_{12} \\ \cdot \\ \cdot \\ u_{1(n-1)} \\ u_{21} \\ u_{22} \\ \cdot \\ \cdot \\ u_{(n-1)(n-2)} \\ u_{(n-1)(n-1)} \end{bmatrix}$$

we can write the following function in MATLAB to create the discretization matrix \mathbf{A} corresponding to the negative of the Laplacian $-\left(\frac{d^2u(x,y)}{dx^2} + \frac{d^2u(x,y)}{dy^2}\right)$ as a function of n :

```

function A = lap(n)
    I = speye(n-1)
    D = -n^2*toeplitz([-2 1 zeros(1,n-3)])
    A = kron(I,D) + kron(D,I)

```

Note, to inspect the matrix **A** using the above function you need to use the function 'full(A)' because the function has used sparse matrix function definitions 'speye', 'toeplitz' and 'kron'.

1. Solve the poisson problem defined above using finite differences (you may use the function 'lap' or construct the matrix **A** yourself. Solve the problem on a grid (with $n = 21$) for the $(n - 1)^2$ unknowns **u**. Once you have obtained the solution vector **u**, use the function 'reshape' (type 'help reshape' in MATLAB) to reshape the vector **u** into a matrix defined on the 20^2 grid points. Use surf to plot your solution in 2D by typing 'surf(u)' once **u** has been reshaped into a matrix.

ANSWER: The 20×20 solution matrix **u** should be saved as a 20×20 matrix in A10.dat.

2. Use a similar code to part 1 to solve the problem

$$\frac{d^2u(x, y)}{dx^2} + \frac{d^2u(x, y)}{dy^2} = e^{-(x^2+y^2)}$$

on the domain $[0, 1] \times [0, 1]$ with boundary conditions $u(x^*, y^*) = 0$ for (x^*, y^*) lying on the boundary of the domain. (Note that the Laplace term in this equation does not have a negative in front of it).

ANSWER: The 20×20 solution matrix **u** should be saved as a 20×20 matrix in A11.dat

IV Read in the noisy data `noisydata.dat`. This consists of values of a function $f(t)$ sampled at 128 time points equally spaced between $t = -5$ and $t = 5 - dt$, where the spacing $dt = 10/128$ (the total time interval divided by the number of samples). Specifically, $\tau = -5 : 10/128 : (5 - 10/128)$. The unit of time t is seconds, so the total duration of the sampled signal is $T = 10$ sec.

1. Denoise the signal by setting to zero the Fourier coefficients for frequencies with absolute values greater than or equal to 1 Hz. This requires fft and ifft commands.

ANSWER: The denoised signal $f_{denoised}(t)$, evaluated at the same 128 time points equally spaced between $t = -5$ and $t = 5 - dt$, should be saved as a 128x1 column vector A12.dat.

2. Repeat, but this time setting to zero the Fourier coefficients for frequencies with absolute values greater than or equal to 0.5 Hz.

ANSWER: The new denoised signal $f_{denoised}(t)$, evaluated at the same 128 time points again, should be saved as a 128x1 column vector A13.dat.