Math 337 Homework 08

Andy Reagan

March 3, 2014

1. Use the Gerschgorin Circles Theorem and the fact that the eigenvalues of real symmetric matrices are real to obtain the best estimate for the location of the eigenvalues of the following tri-diagonal matrix:

$$A = \begin{pmatrix} a & -1 & 0 & . & . & . & 0 \\ -1 & a & -1 & 0 & . & . & 0 \\ 0 & -1 & a & -1 & 0 & . & 0 \\ & & & . & . & . & \\ 0 & . & . & 0 & -1 & a & -1 \\ 0 & . & . & . & 0 & -1 & a \end{pmatrix},$$

where a is a real number. In particular, what is the minimum distance between an eigenvalue of this matrix and zero?

Solution: By the Gerschgorin Circles Theorem, all of the eigenvalues are found in circles centered at a. The radius of these circles is 1, corresponding to the first and last rows, and 2 otherwise. Since the eigenvalues are real, we know that they fall in the interval $[a-2, a+2]$ on the real line.

In particular, the distance between the eigenvalues of this matrix and 0 is at least $a-2$ if $a > 2$ and $a+2$ is $a < 2$. If $|a| \leq 2$, the minimum distance is 0 (i.e., the eigenvalues can well be 0).

2. Consider a linear BVP

$$y'' + 2(2-x)y' = 2(2-x), y(0) = -1, y(6) = 5.$$

Discretize it using scheme (8.4) with $h = 1$.

(i) Verify that you obtain a linear system

$$\begin{pmatrix} -2 & 2 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 3 & -2 & -1 \\ 0 & 0 & 0 & 4 & -2 \end{pmatrix} \begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ -2 \\ -4 \\ 4 \end{pmatrix}.$$

(ii) Solve is using MATLAB. What do you obtain?

(iii) The result you have obtain in part (ii) occurs because one of the conditions of Theorem 8.3 is violated. What is the condition?

Solution:

(i) For this problem, equation (8.4) becomes

$$Y_0 = -1; \tag{1}$$
$$(1 + h(2 - x_n))Y_{n+1} - 2Y_n + (1 - h(2 - x_n))Y_{n-1} = 2h^2(2 - x_n), \quad 1 \leq n \leq 5 \tag{2}$$
$$Y_6 = 5. \tag{3}$$

Setting $h = 1$ in (4), this become

$$(3 - x_n)Y_{n+1} - 2Y_n + (-1 + x_n)Y_{n-1} = 4 - 2x_n; \tag{4}$$

We have the following 5 discrete equations for $Y_{1\ldots5}$, where Y_0 and Y_6 are the supplied BC.

$$\begin{aligned}
(3 - x_1)Y_2 - 2Y_1 + (-1 + x_1)Y_0 &= 4 - 2x_1; \\
(3 - x_2)Y_3 - 2Y_2 + (-1 + x_2)Y_1 &= 4 - 2x_2; \\
(3 - x_3)Y_4 - 2Y_3 + (-1 + x_3)Y_2 &= 4 - 2x_3; \\
(3 - x_4)Y_5 - 2Y_4 + (-1 + x_4)Y_3 &= 4 - 2x_4; \\
(3 - x_5)Y_6 - 2Y_5 + (-1 + x_5)Y_4 &= 4 - 2x_5.
\end{aligned}$$

Plugging in $x_i = i$ and $Y_0 = -1, Y_6 = 5$, and moving constants to the RHS, we are left

$$\begin{aligned}
2Y_2 - 2Y_1 &= 2; \\
Y_3 - 2Y_2 + Y_1 &= 0; \\
-2Y_3 + 2Y_2 &= -2; \\
-Y_5 - 2Y_4 + 3Y_3 &= -4; \\
-2Y_5 + 4Y_4 &= 4.
\end{aligned}$$

Intuitively obvious to the casual observer, this agrees with the desired linear system.

(ii) MATLAB obtain a vector of all NaN's, the matrix being singular to working precision.

```
% HW08 Problem 02

% set A,r from our problem
A = [-2,2,0,0,0;
    1,-2,1,0,0;
    0,2,-2,0,0;
    0,0,3,-2,-1;
    0,0,0,4,2];
r = [2;0;-2;-4;4];

% solve
x = A\r;
disp(x);
```

(iii) The requirement of Theorem 8.3 that $h\mathcal{P} \leq 2$ is violated. Here $P(x)$ achieves a magnitude of 8, at $x = 6$, and we had chose $h = 1$. Therefore, our $h\mathcal{P} = 8$.

3. (a) Give an operation count for finding L and U for a tridiagonal matrix, as per Eq. (8.21).
   (b) Give operation counts for solving the systems in (8.17), as per (8.22) and (8.23).
   (c) Given the total operations count for the Thomas algorithm.

Solution:

(a) We compute $M - 1$ new α values, and $M - 1$ new β values. Each new α requires one operation, and each β requires 2. Therefore, we use $3(M - 1)$ operations.

(b) Solving for $\vec{z}$ requires exactly $2(M - 1)$ operations. The solving for $\vec{y}$ requires exactly $1 + 3(M - 1)$ operations. In total, this is $1 + 5(M - 1)$ operations.

(c) The total operation count is (adding the two previous counts) $1 + 8(M - 1)$.

4. Use the `thomas.m` function, posted under "Codes for examples and selected homework problems," to solve a tridiagonal system $A\vec{y} = \vec{r}$ where A has '2' on the main diagonal and '-1' on the two subdiagonals. Take $\vec{r} = [1, -1, 1, -1, \ldots]^T$ and $M = 1000$ and 5000. Now solve the same system using Matlab's solver. Here, you need to investigate *two* cases: One, when A is constructed as a regular (i.e., full) matrix and two, when it is constructed as a sparse matrix.

   Compare the computational times required to solve this system for your code and for the MATLAB's solver, in those two cases. In particular, comment on *how the computational times scale with* M in each of the three cases considered.

Solution: I compare with the following code, and plot how the computational times scale with M (using more values of M). I find that both the `thomas.m` code and MATLAB's sparse solver scale sub-linearly with M, with MATLAB's solver being faster.

For a non-sparse matrix, MATLAB's solver scales superlinearly with M, and becomes useless quickly.

```
% HW08 Problem 4

clear all;

Mvec=[100,1000,5000,10000,50000];
for i=1:length(Mvec)
    M = Mvec(i);
    a = -ones(M-1,1);
    b = 2.*ones(M,1);
    c = -ones(M-1,1);
    r = ones(M,1);
    r(2:2:M) = -ones(floor(M/2),1);

    tic;
    y = thomas(a,b,c,r);
    tom_t(i) = toc;



    % too slow for the bigger problem
    if i<5
```

```
        A = diag(a,-1) + diag(b) + diag(c,1);
        % check this is correct
        % A(1:10,1:10)
        tic;
        y = A\r;
        mat_t(i) = toc;
    end

    % make sparse
    a = -ones(M,1);
    c = -ones(M,1);
    A = spdiags(a,-1,M,M) + spdiags(b,0,M,M) + spdiags(c,1,M,M);
    % verify sparse storage
    % A(1:10,1:10)

    tic;
    y = A\r;
    mat_t_sp(i) = toc;
end


figure;
plot(Mvec,tom_t)
hold on;
plot(Mvec,mat_t_sp,'r')
legend('thomas algorithm','matlab sparse solver')
xlabel('M','FontSize',20)
ylabel('time','FontSize',20)

figure;
plot(Mvec(1:end-1),mat_t)
xlabel('M','FontSize',20)
ylabel('time','FontSize',20)
```

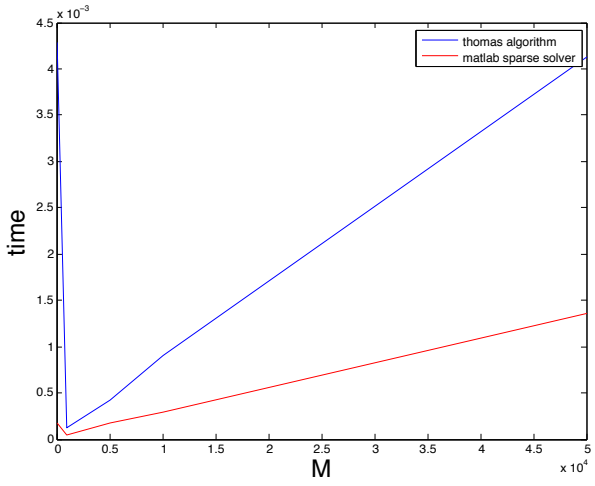

Figure 1: Scaling of computational time for thomas.m and MATLAB's built in solver, with a sparse matrix, versus M.

5. Redo Problem 4 of HW07 using the discretized BVP (8.4) with $h = 0.09$ (ste size $h = 0.1$ will not "fit" into the interval $[0, 1.62]$). Compare the result with that found in HW07. Which method, shooting or finite-difference discretization, is preferable for solving BVPs like this

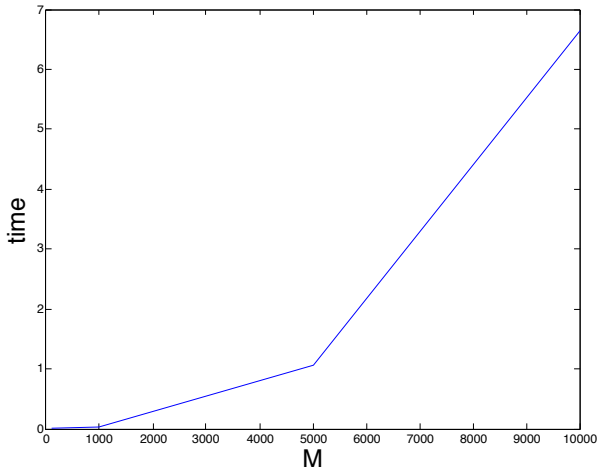

Figure 2: Scaling of computational time for MATLAB's built in solver, with a non-sparse matrix, versus M. It is impractical on a personal computer to extend M much further than a 10,000 by 10,000 matrix, which takes 7 second to solve.

one?

Bonus (a) Plot the error of your numerical solution. Explain the result.

Bonus (b) Repeat the problem with $h = 0.01$ and plot the error. Explain why it is greater than that for $h = 0.09$.

Solution: The problem is

$$y'' = 30^2(y - 1 + 2x), \quad y(0) = 1, \quad y(1.62) = -2.24.$$

Discretizing using (8.4) is

$$\begin{aligned} &Y_0 = 1; \\ &Y_{n+1} - Y_n(2 - 30^2h^2) + Y_{n-1} = 30^2h^2(2x_n - 1), \quad 1 \le n \le N - 1; \\ &Y_N = -2.24 \end{aligned}$$

I build this matrix in MATLAB and solve it using the built in solver.

```
% HW08 Problem 5
%
%

h = 0.09;
x = 0:h:1.62;
y0 = 1;
yf = -2.24;
% set r
r = (30^2*h^2.*(2.*x(2:end-1)-1))';
% account for BC
r(1) = r(1) - y0;
```

```
r(end) = r(end) - yf;
% build A
A = diag((2-30^2*h^2)*ones(length(x)-2,1))+diag(ones(length(x)-3,1),-1)...
    +diag(ones(length(x)-3,1),1);

% check sizes
size(A)
size(r)

% solve
y = A\r;

% plot
plot(x',[y0;y;yf]);
xlabel('x','FontSize',20)
ylabel('y','FontSize',20)
set(gcf, 'units', 'inches', 'position', [1 1 10 10])
set(gcf,'PaperPositionMode','auto')
print('-depsc2','-zbuffer','-r200',sprintf('andy_hw08_prb05_%02g.eps',1))
system(sprintf('epstopdf andy_hw08_prb05_%02g.eps',1));
```

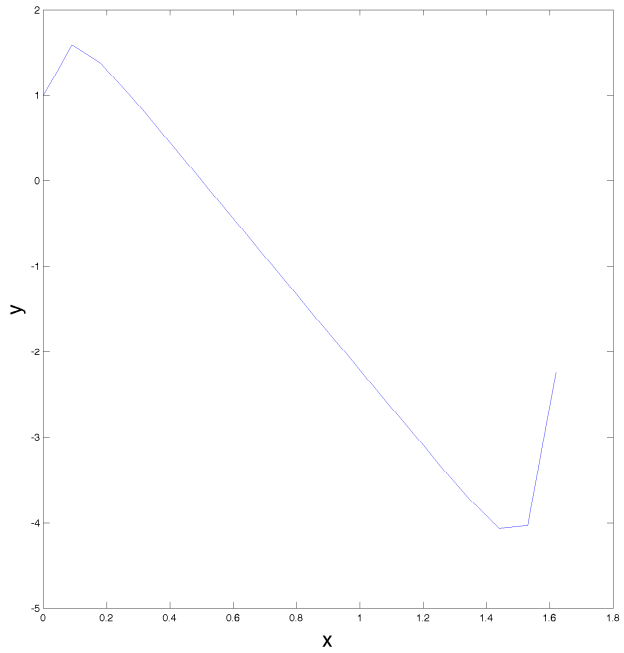

Figure 3: Solution the HW07 problem 4, using the finite difference scheme given by Eq (8.4).

6. Solve the BVP

$$(1+x)^2 y'' = 2y - 4, \qquad y(0) = 0, \qquad y(1) + 2y'(1) = 2$$

using the second-order accurate discretization (8.4) (for $n = 1, \ldots, N-1$) of this BVP. Use Method 1 of Sec. 8.4 modified in such a way that it can handle the mixed type BC at the *right* end point of the interval.

Confirm that you numerical solution has the second order of accurarcy by comparing it at different h with the exact solution $y_{\text{exact}} = 2x/(1+x)$. For this, do the following:

(i) Run your code with $h = 0.05$ and $h = 0.025$;

(ii) Plot the error as a function of x;

(iii) Confirm that the maximum error scales as $O(h^2)$.

Solution: To account for the mixed BC,

$$A_1 y(b) + A_2 y'(b) = \beta,$$

we introduce the point Y_{N+1} and approximate $y'(b)$ with a second order approximation

$$y'(b) = \frac{Y_{N+1} - Y_{N-1}}{2h} + O(h^2).$$

The mixed BC is therefore discretized as

$$A_1 Y_N + A_2 \frac{Y_{N+1} - Y_{N-1}}{2h} = \beta. \tag{5}$$

The ODE discretized at x_N is

$$(1 + P_N) Y_{N+1} - \left(2 - h^2 Q_N\right) Y_N + \left(1 - \frac{h}{2} P_N\right) Y_{N-1} = h^2 R_N \tag{6}$$

Solving for Y_{N+1} in (**??**) into (**??**) we have

$$-\left(2 - h^2 Q_N + 2h \frac{A_1}{A_2}\left(1 + \frac{h}{2} P_N\right)\right) Y_N + 2 Y_{N-1} = h^2 R_N - \left(1 + \frac{h}{2} P_N\right) \frac{2h\beta}{A_2} \tag{7}$$

where the first two terms on the LHS and the last term on the RHS make these matrix diagonal entries different. I solve this in MATLAB, and plot the error versus h. We find that the maximum error increases fourfold when h is doubled, confirming that the method is $O(h^2)$ accurate.

```
% HW08 Problem 6
%
%

hvec = fliplr([0.2;0.1;0.05;0.025;0.0125]);
for i=1:length(hvec)
    h = hvec(i);
    x = (0:h:1)';
    y0 = 0;
    yf = 2;
    % next two for mixed BC
    a1 = 1;
    a2 = 2;
    % set r, N of them
    r = -4*h^2./((1+x(2:end)).^2);
    % account for BC
    r(1) = r(1) - y0;
    r(end) = r(end)-2*h*yf/a2;
    % build A
    A = diag(-2-h^2*2./((1+x(2:end)).^2))... % main diagonal
        +diag(ones(length(x)-2,1),-1)... % sub
        +diag(ones(length(x)-2,1),1); % super
    % account for mixed BC
    A(end,end-1:end) = A(end,end-1:end)+[1,-2*h*a1/a2];
```

```
    % disp(A);

    % check sizes
    % size(A)
    % size(r)

    % solve
    yNum = [y0;A\r];
    yExact = 2.*x./(1+x);

    maxError = max(abs(yExact-yNum));
    errorVec(i) = maxError;
end

errorVec = errorVec';
disp(hvec);
disp(errorVec);
plot(hvec,errorVec);
xlabel('h','FontSize',20)
ylabel('max error','FontSize',20)
set(gcf, 'units', 'inches', 'position', [1 1 10 10])
set(gcf,'PaperPositionMode','auto')
print('-depsc2','-zbuffer','-r200',sprintf('andy_hw08_prb06_%02g.eps',1))
system(sprintf('epstopdf andy_hw08_prb06_%02g.eps',1));
```

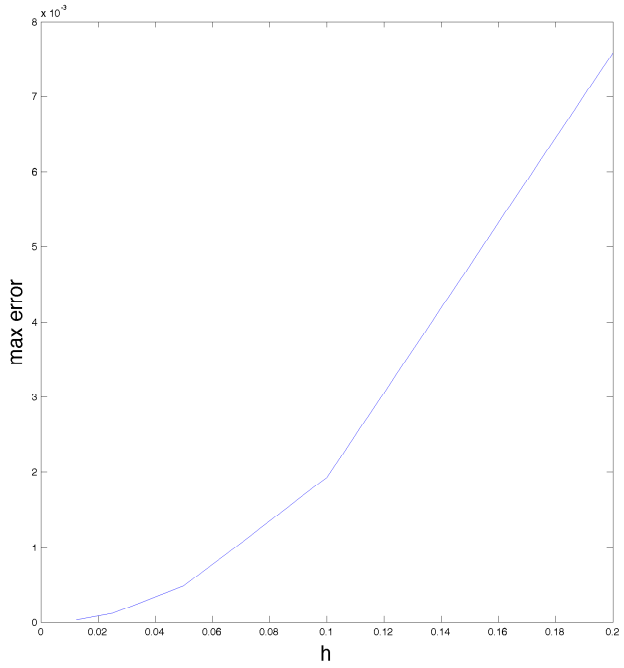

Figure 4: Solution to the BVP with mixed BC at $x = b$, using Method 1 of Sec 8.4.

7. Show that if condition (8.42) and the two conditions stated one line below it hold, then the coefficient matrix in Method 2 based on Eq. (8.39) is SDD.

   Bonus part: Equations (8.36) and (8.39) each lead to a second-order accurate method. Therefore, solutions obtained by those methods must differ by $O(h^3)$. Show *analytically* that this is indeed the case.

Solution:

Appendix 1: ODE Functions

Appendix 2: Numerical Methods