MATH 5510 – Numerical Analysis 1. November 30, 2016

Final Project 1

Write a Matlab function

```
[U] = my_FD_two_point(a,b,c,N,f)
```

that solves the following two point boundary value problem

$$\begin{aligned} -a(x)u''(x) + b(x)u'(x) + c(x)u(x) &= f(x), \quad 0 < x < 1, \\ u(0) &= 0, \\ u'(1) &= 0. \end{aligned}$$

by using the finite difference scheme given in class. The input for the function is the following:

- $a(x), b(x), c(x)$ – inline functions of coefficients
- $N + 1$– number of the mesh points in the method
- $f(x)$ – inline function for the RHS.

The output should be the $(N + 1) \times 1$ solution vector U.

Your method should solve the resulting tridiagonal matrix equation (the matrix has never to be created) by using your own tridiagonal LU-decomposition. Use that any tridiagonal matrix can be decomposed as a product of two bidiagonal matrices as follows

$$\begin{pmatrix} a_1 & c_1 & & \\ b_1 & \ddots & \ddots & \\ & \ddots & \ddots & c_{n-1} \\ & & b_{n-1} & a_n \end{pmatrix} = \begin{pmatrix} 1 & & & \\ d_1 & \ddots & & \\ & \ddots & \ddots & \\ & & d_{n-1} & 1 \end{pmatrix} \begin{pmatrix} e_1 & f_1 & & \\ & \ddots & \ddots & \\ & & \ddots & f_{n-1} \\ & & & e_n \end{pmatrix}$$

Your need to write a Matlab function

```
[d,e,f] = my_tridiagonal_decomposition(a,b,c)
```

that performs this decomposition. The function's input should be vectors a, b, c that form diagonals of the original tridiagonal matrix and the function's output should be vectors d, e, f that form the diagonals of the decomposed matrices. No additional memory should be used. Then write your own MATLAB m-functions that solves the bi-diagonal systems $Ax = b$ and $Bx = b$, where

$$A = \begin{pmatrix} 1 & & & & \\ c_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & c_{n-2} & 1 & \\ & & & c_{n-1} & 1 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} d_1 & e_1 & & & \\ & d_2 & e_2 & & \\ & & \ddots & \ddots & \\ & & & d_{n-1} & e_{n-1} \\ & & & & d_n \end{pmatrix}$$

by back and front substitutions. Your functions should take as an input vectors

$$[c_1; \ldots; c_{n-1}], \quad [d_1; \ldots; d_n], \quad [e_1; \ldots; e_n], \quad [b_1; \ldots; b_n],$$

and output vector

$$[x_1; \ldots; x_n].$$

such that $Ax = b$ or $Bx = b$.

To check your program run the following script (can be downloaded from course website). E-mail me your function. Print and submit the script of your function, the tables and the figures produced from running the test script. Make sure that the names of your functions match.

```
% Final_project1_check

N = 20; % number of mesh points
a = inline('1+x','x'); % a(x) function
b = inline('1+x.^2','x'); % b(x) function
c = inline('x','x'); % c(x) function
f = inline('sin(pi/2*x).*(x+(1+x)*pi^2/4)+(1+x.^2).*cos(pi/2*x)*pi/2','x'); % RHS

% computing solution FD method
U = my_FD_two_point(a,b,c,N,f);

x = linspace(0,1,N+1);
% exact solution
U_ex = sin(pi/2*x);
% plotting exact and the approximate solution
figure(1)
plot(x,U(:),'r*', x,U_ex);


% check space rate of convergence
fprintf(['   dx     error        rate of convergence         \n'] )
   disp('-----------------------------------------------------------------')

error = zeros(10,1);
N = 10;
U = my_FD_two_point(a,b,c,N,f);
x = linspace(0,1,N+1)';
U_ex = sin(pi/2*x);
error(1) = norm(U(:)-U_ex(:),inf);
for j = 2:5
       N = 10*2^(j-1);
       U = my_FD_two_point(a,b,c,N,f);
       x = linspace(0,1,N+1)';
       U_ex = sin(pi/2*x);
       error(j) = norm(U(:)-U_ex(:),inf);
       dx = 2/(10*2^(j-1));
       fprintf([' %1.2e  %20.3e    %20.5f   \n'],...
       dx, error(j), log2(error(j-1)/error(j))  )
end
disp('-----------------------------------------------------------------')
```