UBC CPSC 302 Num Comp for Alg Problems, 2019W

Assignment 5

by j0k0b #42039157

Tips: Use the tips from Assignment 1 to get the most credit possible on this and future assignments. Replace the [Preferred Name or CS-ID] and [Student Number] above.

Table of Contents

UBC CPSC 302 Num Comp for Alg Problems, 2019W
Assignment 5
by j0k0b #42039157
Acknowledgments
Question 1 LU decomposition by steps
Question 1.1 Without pivoting
Answer 1.1 Without pivoting [2 marks]
Question 1.2 Partial pivoting
Answer 1.2 Partial pivoting [2 marks]
Question 2
Question 2.1 A modified matrix
Answer 2.1 A modified matrix [2 marks]
Question 2.2 Singular
Answer 2.2 Singular [2 marks]
Question 2.3 Find x(3)
Answer 2.3 Find x(3) [2 marks]
Question 3
Question 3.1 Norm
Answer 3.1 Norm [2 marks]
Question 3.2 Symmetric positive definite
Answer 3.2 Symmetric positive definite [2 marks]
Question 3.3 Condition number
Answer 3.3 Condition number [2 marks]
Question 4 Pentadiagonal solver
Answer 4 Pentadiagonal solver [8 marks]
5 My functions
5.1 penta
6 Provided functions
6.1 lustep
6.2 dispmat

Acknowledgments

You are welcome to work with other students on this assignment, but everything in this file must clearly be your own original work. Complete the following section to recognize your collaborators on this assignment.

Begin acknowledgments

Thanks to q5c1b for helping me with understand question 3 and 4.

Other than the contributions above, this work is my own.

— j0k0b

End acknowledgments

Question 1 LU decomposition by steps

Consider the 3×3 matrix

```
A1 = [1 2 3; 4 5 6; 7 8 10]
```

```
A1 = 3×3
     1     2     3
     4     5     6
     7     8    10
```

Question 1.1 Without pivoting

Find the LU decomposition of A when no pivoting is used. Show all intermediate steps, and express all row operations in terms of elementary matrices $M^{(k)}$. Make sure to show all the matrices that we have defined in class, particularly $M^{(1)}$, $M^{(2)}$, and the actual factors L and U.

Confirm that your resulting L and U decompose A.

You may do the calculations by hand, write your own MATLAB program, or use the provided functions lustep (from lesson 10-09 Wed "Pivoting strategies. Efficient implementation") and dispmat (below) to aid in these calculations.

Answer 1.1 Without pivoting [2 marks]

Enter your solution in the space below.

Begin answer

```
M_1 = lustep(A1, 1);
dispmat("M^(1) = ", M_1); % M^(1) = lu(A1, 1)
```

```
M^(1) =
      1            0            0
     -4            1            0
     -7            0            1
```

```
M_2 = lustep(M_1*A1, 2);
dispmat("M^(2) = ", M_2); % M^(2) = lu(M_1*A1, 2)
```

```
M^(2) =
      1            0            0
      0            1            0
      0           -2            1
```

```
dispmat("L = ", abs(M_1*M_2)); % L = abs(M_1*M_2)
```

```
L =
      1            0            0
      4            1            0
      7            2            1
```

```
dispmat("U = ", M_2*M_1*A1); % U = M_2*M_1*A1
```

```
U =
      1            2            3
      0           -3           -6
      0            0            1
```

End answer

Question 1.2 Partial pivoting

Find the LU decomposition of A when partial pivoting is used. Show all your intermediate steps, and express all row operations in terms of elementary matrices $M^{(k)}$. Similarly to part 1.1, show all matrices involved, and remember to show also the permutation matrices that you use.

Confirm that your resulting L, U, and P decompose A.

You may do the calculations by hand, write your own MATLAB program, or use the provided functions lustep and dispmat to aid in these calculations.

Answer 1.2 Partial pivoting [2 marks]

Enter your solution in the space below.

Begin answer

```
T_1 = rref(A1);
P_1 = zeros(3,3);
P_1(1,:) = T_1(3,:);
P_1(2,:) = T_1(2,:);
P_1(3,:) = T_1(1,:);
dispmat("P^(1) = ",P_1); % P^(1) = P_1
```

```
P^(1) =
       0              0              1
       0              1              0
       1              0              0
```

```
dispmat("P^(1)A1 = ", P_1*A1); % P^(1)A1 = P_1*A1
```

```
P^(1)A1 =
       7              8             10
       4              5              6
       1              2              3
```

```
M_1 = lustep(P_1*A1, 1);
dispmat("M^(1)", M_1); % M^(1) = lustep(P_1*A1,1)
```

```
M^(1)
       1              0              0
     -4/7             1              0
     -1/7             0              1
```

```
dispmat("M^(1)P^(1)A1",M_1*P_1*A1); % "M^(1)P^(1)A1 = M_1*P_1*A1
```

```
M^(1)P^(1)A1
       7              8             10
       0             3/7            2/7
       0             6/7           11/7
```

```
P_2 = zeros(3);
P_2(1,:) = T_1(1,:);
P_2(2,:) = T_1(3,:);
P_2(3,:) = T_1(2,:);
dispmat("P^(2) = ", P_2); % P^(2) = P_2
```

```
P^(2) =
       1              0              0
       0              0              1
       0              1              0
```

```
dispmat("P^(2)M^(1)P^(1)A1 = ", P_2*M_1*P_1*A1); % P^(2)M^(1)P^(1)A1 = P_2*M_1*P_1*A1
```

```
P^(2)M^(1)P^(1)A1 =
       7              8             10
       0             6/7           11/7
       0             3/7            2/7
```

```
M_2 = lustep(P_2*M_1*P_1*A1, 2);
dispmat("M^(2) = ", M_2); % M^(2) = M_2
```

```
M^(2) =
       1              0              0
       0              1              0
       0            -1/2             1
```

```
dispmat("U = ", M_2*P_2*M_1*P_1*A1); % U = M_2*P_2*M_1*P_1*A1
```

```
U =
       7              8             10
       0             6/7           11/7
       0              0            -1/2
```

```
dispmat("Which is similar to ", M_2*P_2*M_1*(P_2)'*P_2*P_1*A1); % M_2*P_2*M_1*(P_2)'*P_2*P_1*A1
```

```
Which is similar to
       7              8             10
       0             6/7           11/7
       0              0            -1/2
```

```
dispmat("Therefore, P = P^(2)P^(1) = ", P_2*P_1); % P = P^(2)P^(1)
```

```
Therefore, P = P^(2)P^(1) =
       0              0              1
       1              0              0
       0              1              0
```

```
dispmat("L = ", abs(P_2*M_1*(P_2)'*M_2)); % L = abs(P_2*M_1*(P_2)'*M_2)
```

```
L =
       1              0              0
      1/7             1              0
      4/7            1/2             1
```

```
[L,U,P] = lu(A1)
```

```
L = 3×3
    1.0000         0         0
    0.1429    1.0000         0
    0.5714    0.5000    1.0000
U = 3×3
    7.0000    8.0000   10.0000
         0    0.8571    1.5714
         0         0   -0.5000
P = 3×3
     0     0     1
     1     0     0
     0     1     0
```

End answer

Question 2

Consider now a slight modification of the matrix from Question 1:

```
A2=A1; A2(3,3)=9
```

```
A2 = 3×3
     1     2     3
     4     5     6
     7     8     9
```

Question 2.1 A modified matrix

Use MATLAB's instruction

```
[L2,U2,P2]=lu(A2);
```

or apply a procedure as in Question 1.2 to find the LU decomposition with partial pivoting of this matrix. Compare your decomposition to your Answer 1.2.

Answer 2.1 A modified matrix [2 marks]

Enter your solution in the space below.

Begin answer

```
[L2,U2,P2] = lu(A2)
```

```
L2 = 3×3
    1.0000         0         0
    0.1429    1.0000         0
    0.5714    0.5000    1.0000
U2 = 3×3
    7.0000    8.0000    9.0000
         0    0.8571    1.7143
         0         0   -0.0000
P2 = 3×3
     0     0     1
     1     0     0
     0     1     0
```

As we can see, considering the previous question and this one, that P and L remain the same, while U has changed.

End answer

Question 2.2 Singular

A is in fact singular. Explain why, either by using your result in Question 2.1 or by other means.

Answer 2.2 Singular [2 marks]

Enter your solution in the space below.

Begin answer

There are two claims that we can make, and both of them should prove that A is a singular matrix. Firstly, let's observe what happens when we subtract the second and the third rows from the first in A.

```
T = zeros(3);
T(1,:) = A2(1,:);
T(2,:) = A2(2,:) - A2(1,:);
T(3,:) = A2(3,:) - A2(1,:);
dispmat("Resulting matrix = ", T);
```

```
Resulting matrix =
        1               2               3
        3               3               3
        6               6               6
```

Above, we can see that the Third Row is twice of that of the Second Row. $\therefore$ We can claim that A must be a singular matrix.

Secondly, from the previous part (2.1), we can see that the value in the 3rd Row and 3rd Column of the Matrix U (i.e., $U2$) is 0. We already know that matrix A is a product of U and other matrices, and U is already a singular matrix ($u_{3,3} = 0$).

End answer

Question 2.3 Find x(3)

Let us denote $\mathbf{y} = L^{-1}P\mathbf{b}$. Given a right hand side vector $\mathbf{b}$ such that $y_3 = 1$ (and arbitrary values y_1 and y_2), and attempting to solve $A\mathbf{x} = \mathbf{b}$, what is x_3 in exact arithmetic? What result does MATLAB compute for x_3?

Answer 2.3 Find x(3) [2 marks]

Enter your solution in the space below.

Begin answer

Using the formula, we can say that $b = (L^{-1}P)^{-1}y$. And we also know that $Ax = b$. $\therefore$ For any value of b, such that $y_3 \neq 0$, $x_3 = \frac{y_3}{0}$ in exact arithmetic. Following is the MATLAB code for solving this with $y_1 = 3, y_2 = 2, y_3 = 1$. We get $x_3 = -0.6305 \times 10^{16}$ where $u_{33} = -1.586 \times 10^{-16}$.

```
y = [3;2;1];
warning off
U2(3,3)
```

```
ans = -1.5860e-16
```

```
x = U2\y
```

```
x = 3×1
10^16 ×
   -0.6305
    1.2610
   -0.6305
```

```
warning on
```

End answer

Question 3

Given that a and b are two real positive numbers (parameters), the eigenvalues of the symmetric tridiagonal matrix

$$A = \begin{pmatrix} a & b & & \\ b & \ddots & \ddots & \\ & \ddots & \ddots & b \\ & & b & a \end{pmatrix}$$

of size $n \times n$ are $\lambda_j = a + 2b\cos(\pi j/(n+1))$, $j = 1, \dots, n$.

Aside:

A non-sparse version of this matrix can be obtained in MATLAB with the instruction

```
A = diag(a*ones(n,1),0) + diag(b*ones(n-1,1),1) + diag(b*ones(n-1,1),-1)
```

Type `help spdiags` for the scoop on the sparse version.

Question 3.1 Norm

Find $||A||_\infty$ (algebraically, for any $a, b > 0$ and n).

Answer 3.1 Norm [2 marks]

Enter your solution in the space below.

Begin answer

We know that $||A||_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |a_{ij}|$, where a_{ij} is an element of matrix$A$, and i is the row and j is the column, i.e, $i, j \in 1, \dots, n$. Secondly, we know that $\lambda_j = a + 2b\cos(\pi j/(n+1))$. So, max of $|a + 2b\cos(\pi j/(n+1))|$ for any n and j is $a + 2b$.

$\therefore ||A||_\infty = a + 2b$.

End answer

Question 3.2 Symmetric positive definite

Assuming that $a > 2b$, show that A is symmetric positive definite.

You may use the fact that eigenvectors from different eigenvalues are orthogonal for real symmetric matrices.

Answer 3.2 Symmetric positive definite [2 marks]

Enter your solution in the space below.

Begin answer

We are give a symmetric tridiagonal matrix, so we already know that the eigenvalues are real.

Following is the more concrete way to prove that the given matrix is symmetric.

```
% random matrix example for Question 3
n = randi(5); % random number between 0 and 5
a = randi(5); % random number between 0 and 5
b = randi(5); % random number between 0 and 5
A = diag(a*ones(n,1),0) + diag(b*ones(n-1,1),1) + diag(b*ones(n-1,1),-1); % non sparse
A_t = transpose(A); % transpose
dispmat("A = ", A); % display A
```

```
A =
     3     3     0     0     0
     3     3     3     0     0
     0     3     3     3     0
     0     0     3     3     3
     0     0     0     3     3
```

```
dispmat("Transpose of A", A_t); % this should be the same matrix
```

```
Transpose of A
     3     3     0     0     0
     3     3     3     0     0
     0     3     3     3     0
     0     0     3     3     3
     0     0     0     3     3
```

```
dispmat("Subtracting matrices", A - A_t); % this should be a zero matrix of size n x n
```

```
Subtracting matrices
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

Now, we are given that $\lambda_j = a + 2b\cos(\pi j/(n+1))$. Since, we know that both a and b are real positive numbers, we just need to consider about the cos function. Using our knowledge of trigonometric functions we can also claim that $-1 \leq cos(x) \leq 1$ for any real x.

$\therefore (a - 2b) \leq \lambda_j \leq (a + 2b)$

And, since $a > b$, based on our assumption; $\lambda_j \geq a - 2b > 0$ for all j. Hence, this proves that$A$ is a symmetric positive definite.

End answer

Question 3.3 Condition number

Continuing to assume that $a > 2b$, find the condition number $\kappa_2(A)$. What happens when a gets close to $2b$ and n is large?

Answer 3.3 Condition number [2 marks]

Enter your solution in the space below.

Begin answer

We can optimize the function for λ_j for better calculation as follows:

$cos(\frac{\pi n}{n+1}) = cos(\pi - \frac{\pi}{n+1}) = cos(\pi)cos(\frac{\pi}{n+1}) + sin(\pi)sin(\frac{\pi}{n+1}) = -cos(\frac{\pi}{n+1})$.

And we know that $-1 \leq cos(\frac{\pi}{n+1}) \leq 1$, therefore,

$$\kappa_2(A) = \frac{\max_j\{\lambda_j\}}{\min_j\{\lambda_j\}} = \frac{a + 2b\cos(\frac{\pi}{n+1})}{a - 2b\cos(\frac{\pi}{n+1})}.$$

Now, we know that only thing varying is the $\frac{\pi}{n+1}$. When n is large, i.e., very large $lt_{n->\infty}\frac{\pi}{n+1} \approx \frac{\pi}{\infty} \approx 1$.

$\therefore \kappa_2(A) \approx \frac{a+2b}{a-2b}$. As a gets closer to $2b$, $(a-2b)$ gets closer to 0, so, $\kappa_2(A) \propto \frac{1}{a-2b}$.

End answer

Question 4 Pentadiagonal solver

Write a MATLAB function `penta` below that solves pentadiagonal systems of equations of size n. A pentadiagonal matrix is a banded one with $\text{bandwidth} = 5$ and $p = q = 3$. Assume that no pivoting is needed, but do not assume that the matrix A is symmetric. Your program should expect as input six vectors:

- `md` = main diagonal vector of A (length n)
- `ld` = lower diagonal of A (length $n-1$)
- `lld` = next lower diagonal of A (length $n-2$)
- `ud` = upper diagonal of A (length $n-1$)
- `uud` = next upper diagonal of A (length $n-2$)
- `b` = right hand side vector (length n)

It should calculate and return $\mathbf{x} = A^{-1}\mathbf{b}$ using a Gaussian elimination variant that required $\mathcal{O}(n)$ flops and consumes no additional space as a function of n (i.e., in total $7n$ storage locations are required).

Try your program on the matrix defined by $n = 100,000$, $a_{i,i+2} = a_{i+2,i} = i$, $a_{i,i+1} = a_{i+1,i} = -2i$, and $a_{i,i} = 8i$, for all i such that the relevant subscripts fall in the range 1 to n. Derive a right hand side vector $\mathbf{b} = A\mathbf{x}_{exact}$ using $\mathbf{x}_{exact} = (1, 1, \ldots, 1)^T$. Then solve for $\mathbf{x}$ given this $\mathbf{b}$ and record $\|\mathbf{x}_{exact} - \mathbf{x}\|_2$ to verify your solver.

Answer 4 Pentadiagonal solver [8 marks]

Enter your solution in the space below.

Begin answer

```
n = 100000;
i = [1:n]';
md = 8*i;
ld = -2*i(1:n-1);
ud = ld;
lld = i(1:n-2);
uud = lld;
x = ones(n,1);
b(1) = md(1)*x(1) + ud(1).*x(2)+uud(1)*x(3);
b(2) = ld(1)*x(1) + md(2).*x(2)+ud(2).*x(3)+uud(2)*x(4);
b(3: n-2) = lld(1: n-4).*x(1:n-4)+ ld(2:n-3).*x(2:n-3)+md(3:n-2)+ud(3:n-2).*x(4:n-1) + uud(3:n-2).*x(5:n);
b(n-1) = lld(n-3).*x(n-3)+ld(n-2).*x(n-2)+md(n-1).*x(n-1)+ud(n-1).*x(n);
b(n) = lld(n-2).*x(n-2)+ld(n-1).*x(n-1)+md(n)*x(n);
x_p = penta(md,ld,ud,lld,uud,b);
norm(x-x_p) %error in ||x_{exact}-x||_2
```

```
ans = 4.3355e-14
```

End answer

5 My functions

MATLAB requires all functions to be at the end of the script. Enter functions you wrote for this assignment in the space below.

Tips: You only need to complete this section if you developed and used your own MATLAB function(s) in your code above. If you do define your own functions, organize your code; use indenting and reasonable variables (e.g. no unused variables or ambiguous names). Document your code clearly and effectively with comments (`% This is a comment`). Note the specific purpose of each function; indicate input requirements and output results.

Begin functions

5.1 penta

```
function [x] = penta(md,ld,ud,lld,uud,b)
%
%  Solve Ax = b for a pentadiagonal A
%  md  = main diagonal vector of A (length n)
%  ld  = lower diagonal of A (length (n-1))
%  lld = next lower diagonal of A (length (n-2))
%  ud  = upper diagonal of A (length (n-1))
%  uud = next upper diagonal of A (length (n-2))
%  b   = right hand side vector (length n)
l = length(md);
k = 1;
while (k < l) % LU
    ld(k) = ld(k)/md(k);
    b(k+1) = b(k+1) - ld(k)*b(k);
    md(k+1) = md(k+1) - ld(k)*ud(k);

    if (k < (l-1))
        ud(k+1) = ud(k+1) - ld(k)*uud(k);
        lld(k) = lld(k)/md(k);
        ld(k+1) = ld(k+1) - lld(k)*ud(k);
        md(k+2) = md(k+2) - lld(k)*uud(k);
        b(k+2) = b(k+2) - lld(k)*b(k);
    end
    k= k+1;
end % LU
x = b; % backward
x(l) = b(l)/md(l);
x(l-1) = (b(l-1)-ud(l-1)*x(l))/md(l-1);
j = l-2;
while (j > 0)
    x(j) = (b(j) - ud(j)*x(j+1)- uud(j)*x(j+2))/md(j);
    j = j-1;
end % backward
x = shiftdim(x);

end % penta
```

End functions

6 Provided functions

Additional utility functions are provided here. You should not need to modify these functions and may use them "as-is" in your code.

6.1 lustep

```
function M = lustep(A, k)
% Performs the k-th step of LU-decomposition by generating the elementary
% matrix M^(k).

n = size(A,1);
```

```
M = eye(n);

% generate elements of M below the main diagonal in column k
for i=k+1:n
    % l(i,k) = A(i,k)/A(k,k);
    % and M_ik = -l_ik for i>k
    M(i,k) = - A(i,k)/A(k,k);
end

end % lustep
```

6.2 dispmat

```
function out = dispmat(name,in)
% Evaluates a matrix, displaying its name and elements
% as rational approximations
%
% E.g. C = dispmat('C = ', A*B); % displays C, which is A*B
out = in;
disp(name);
disp(rats(out));

end % dispmat
```