Solution of Week 7 Lab (Prepared by Yuan Yin)

December 22, 2019

1 Exercise 1: LU Factorization:

(a).

Read the Tute sheet.

(b).

Read the Tute sheet.

(c).

The matrix, P, is the permutation matrix which performs pivoting. For example, if A is a 3×3 matrix and we want to switch the first and the third rows of A to get A', then

$$P = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix},$$

and P * A = A'.

Why we nee to use $y = L \setminus (P * b)$?

—— This is because $[L, U, P] = lu(A)$ returns a unit lower triangular matrix L, an upper triangular matrix U, and a permutation matrix P such that PA = LU.

$\Rightarrow P * A * x = P * b;$

$\Rightarrow L * U * x = P * b;$

$\Rightarrow L * y = P * b$ by setting $y = U * x$;

$\Rightarrow y = L \setminus P * b;$

$\Rightarrow x = U \setminus y.$

(d).

Note that p here is a permutation vector, and it permutes the rows of A as follows:

$$A = \begin{bmatrix} 1st\ row\ of\ A \\ 2nd\ row\ of\ A \\ \vdots \\ nth\ row\ of\ A \end{bmatrix} \Rightarrow A(p,:) = \begin{bmatrix} p_1th\ row\ of\ A \\ p_2th\ row\ of\ A \\ \vdots \\ p_nth\ row\ of\ A \end{bmatrix},\ where\ P = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix}.$$

2 Exercise 2: Operation Counts:

[3]:
```
%%file TestEfficiency.m

function TestEfficiency
%% PART A



n = 1000;
fprintf('When n is %d:\n', n);

% We construct A, x, b in the way such that Ax = b:
A = rand(n, n);
x = rand(n, 1);
b = A * x;

fprintf('\nUsing LU Factorisation: ');
tic
x_LU = A\b;
toc

fprintf('\nUsing Inverse Matrix: ');
tic
x_Inv = inv(A) * b;
toc


%% PART B
fprintf('\n\n------------------------------------------------------------\n\n');

nn = 1000;
fprintf('When n is %d:\n', nn);

% We construct AA, xx, bb in the way such that AA * xx = bb, also, AA needs
% to be positive definite since we want to use the command 'chol':
I = rand(nn);
AA = I' * I;
xx = rand(nn, 1);
bb = AA * xx;

fprintf('\nUsing Backslash: ');
tic
xx_LU = AA\bb;
toc

fprintf('\nUsing "LU" command: ')
tic
```

```
[L,U,P] = lu(AA);
xx_lu = U\(L\P * bb);
toc

fprintf('\nUsing Cholesky Factorization: ')
tic
R = chol(AA);
xx_chol = R\(R'\bb);
toc

end
```

```
Created file '/Users/RebeccaYinYuan/MAST30028 Tutorial Answers Yuan Yin/WEEK 7/TestEfficiency.m'.
```

[4]:
```
TestEfficiency
```

```
When n is 1000:

Using LU Factorisation: Elapsed time is 0.134665 seconds.

Using Inverse Matrix: Elapsed time is 0.090224 seconds.

------------------------------------------------------------

When n is 1000:

Using Backslash: Elapsed time is 0.032485 seconds.

Using "LU" command: Elapsed time is 0.068457 seconds.

Using Cholesky Factorization: Elapsed time is 0.020645 seconds.
```


3 Exercise 3: Backslash:

(a).

One way to generate symmetric, positive definite matrix, A, is multiplying one random matrix with its transpose. The MATLAB codes can be written as follows:

$n = 1000;$

$I = rand(n);$

$AA = I' * I;$

In 'CholScalar.m', A being symmetric means that it can be decomposed into $A = U'\Lambda U$, where U is an orthogonal matrix and Λ is a diagonal matrix with eigenvalues of A as its diagonal elements.

If A is not positive definite, its eigenvalues may be ZERO, and this will cause SINGULAR Cholesky factorisation.

(b).

Check out the MATLAB file "bslashtx.m".

4 Exercise 4: Vector and Matrix Norms:

(a).

known: $b = (3, -5, 2)$

$\Rightarrow ||b||_1 = |3| + |-5| + |2| = 10;$

$\Rightarrow ||b||_2 = \sqrt{3^2 + (-5)^2 + 2^2} = \sqrt{38} = 6.1644;$

$\Rightarrow ||b||_\infty = |-5| = 5;$

(b).

known:

$$A = \begin{bmatrix} 33 & -17 & 22 \\ 11 & -12 & -1 \\ 0 & -9 & 24 \end{bmatrix}$$

$\Rightarrow ||A||_1 = Max.Col.Sum = |22| + |-1| + |24| = 47$

$\Rightarrow ||A||_M = Max.Entry = 33$

$\Rightarrow ||A||_F = \sqrt{33^2 + (-17)^2 + 22^2 + 11^2 + (-12)^2 + (-1)^2 + 0^2 + (-9)^2 + 24^2} = 52.7731$

$\Rightarrow ||A||_\infty = Max.Row.Sum = |33| + |-17| + |22| = 72$

NOTE: (Run 'CheckNorms.m' to check your answers!)

[6]:

```
%%file CheckNorms.m

function CheckNorms
%% PART A: Check Vector norms:
clc

b = [3, -5, 2];

one_norm_vec = norm(b, 1)
two_norm_vec = norm(b)
inf_norm_vec = norm(b, Inf)


%% PART B: Check Matrix norms:
fprintf('\n\n---------------------------------\n\n');

A = [33, -17, 22; 11, -12, -1; 0, -9, 24];
```

```
one_norm_matrix = norm(A, 1)
max_norm_matrix = max(abs(A(:)))
frob_norm_matrix = norm(A, 'fro')
inf_norm_matrix = norm(A, Inf)

two_norm_matrix = norm(A)



end
```

```
Created file '/Users/RebeccaYinYuan/MAST30028 Tutorial Answers Yuan Yin/WEEK
7/CheckNorms.m'.
```

[7]:
```
CheckNorms
```

```
one_norm_vec =

    10


two_norm_vec =

    6.1644


inf_norm_vec =

     5



----------------------------------


one_norm_matrix =

    47


max_norm_matrix =

    33


frob_norm_matrix =
```

```
   52.7731


inf_norm_matrix =

    72


two_norm_matrix =

   48.0399
```