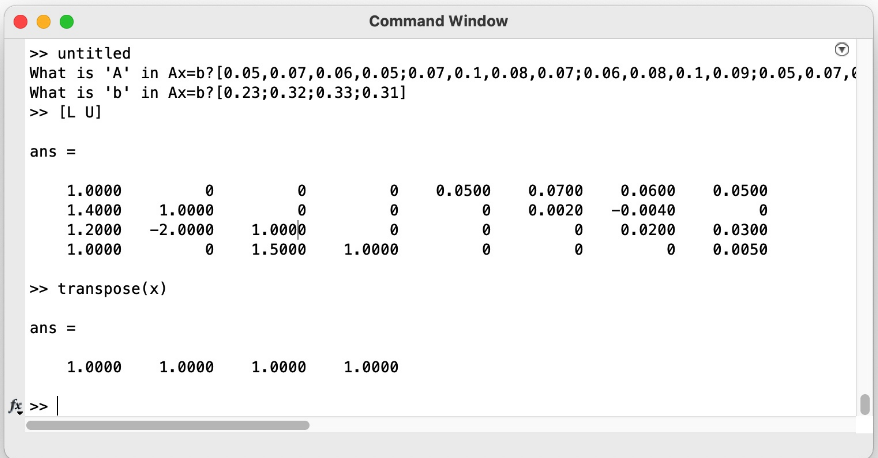
MATH 501 Problem Set 5

Qilin Ye

March 3, 2021

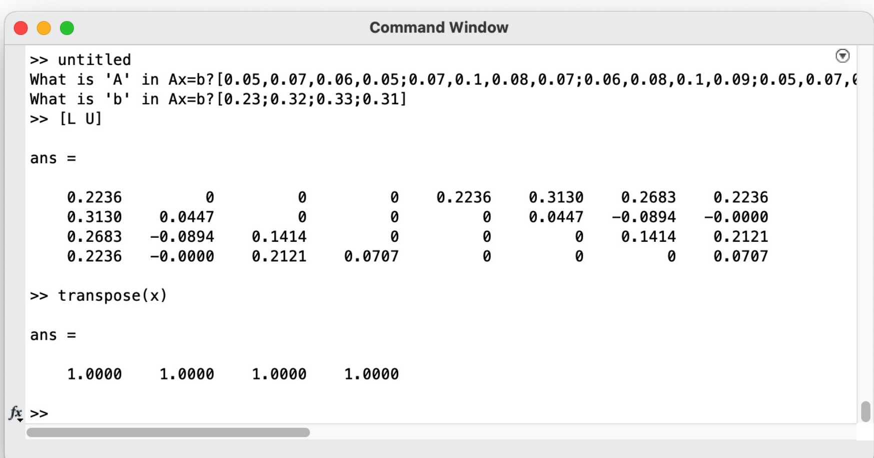
Pseudocode Programming

Doolittle's Factorization:



```
>> untitled
What is 'A' in Ax=b?[0.05,0.07,0.06,0.05;0.07,0.1,0.08,0.07;0.06,0.08,0.1,0.09;0.05,0.07,
What is 'b' in Ax=b?[0.23;0.32;0.33;0.31]
>> [L U]

ans =

    1.0000         0         0         0    0.0500    0.0700    0.0600    0.0500
    1.4000    1.0000         0         0         0    0.0020   -0.0040         0
    1.2000   -2.0000    1.0000         0         0         0    0.0200    0.0300
    1.0000         0    1.5000    1.0000         0         0         0    0.0050

>> transpose(x)

ans =

    1.0000    1.0000    1.0000    1.0000

>>
```

Cholesky's Factorization:



```
>> untitled
What is 'A' in Ax=b?[0.05,0.07,0.06,0.05;0.07,0.1,0.08,0.07;0.06,0.08,0.1,0.09;0.05,0.07,
What is 'b' in Ax=b?[0.23;0.32;0.33;0.31]
>> [L U]

ans =

    0.2236         0         0         0    0.2236    0.3130    0.2683    0.2236
    0.3130    0.0447         0         0         0    0.0447   -0.0894   -0.0000
    0.2683   -0.0894    0.1414         0         0         0    0.1414    0.2121
    0.2236   -0.0000    0.2121    0.0707         0         0         0    0.0707

>> transpose(x)

ans =

    1.0000    1.0000    1.0000    1.0000

>>
```

```matlab
A = input("What is A in Ax=b?");
n = size(A);
n = n(1);
b = input("What is b in Ax=b?");

%Doolittle, solve A=LU
L = zeros(n);
U = zeros(n);

for k = 1:n
    L(k,k) = 1;
    for j = k:n
        A_kj_Old = A(k,j);
        for s = 1:k-1
            A_kj_Old = A_kj_Old - L(k,s) * U(s,j);
        end
        U(k,j) = A_kj_Old;
    end

    for i = k+1:n
        A_ik_Old = A(i,k);
        for s = 1:k-1
            A_ik_Old = A_ik_Old - L(i,s) * U(s,k);
        end
        L(i,k) = A_ik_Old / U(k,k);
    end
end


%Fwd substitution, solve Lz=b
z = zeros(n,1);

for i=1:n
   b_i_Old = b(i);
   for j = 1:i-1
       b_i_Old = b_i_Old - L(i,j) * z(j);
   end
   z(i) = b_i_Old;
end

%Bwd substitution, solve Ux=z
x = zeros(n,1);

for i = 0:n-1
    z_i_Old = z(n-i);
    for j = n+1-i:n
        z_i_Old = z_i_Old - U(n-i,j) * x(j);
    end
    x(n-i) = z_i_Old / U(n-i,n-i);
end

clearvars -except A L U x b
```

```matlab
A = input("What is A in Ax=b?");
n = size(A);
n = n(1);
b = input("What is b in Ax=b?");

%Cholesky, solve A=LU
L = zeros(n);

for k = 1:n
    a_kk_Old = A(k,k);
    for s=1:k-1
        a_kk_Old = a_kk_Old - L(k,s)^2;
    end
    L(k,k) = sqrt(a_kk_Old);

    for i = k+1:n
        a_ik_Old = A(i,k);
        for s = 1:k-1
            a_ik_Old = a_ik_Old - L(i,s) * L(k,s);
        end
        L(i,k) = a_ik_Old / L(k,k);
    end
end

U = transpose(L);

%Fwd substitution, solve Lz=b
z = zeros(n,1);

for i=1:n
   b_i_Old = b(i);
   for j = 1:i-1
       b_i_Old = b_i_Old - L(i,j) * z(j);
   end
   z(i) = b_i_Old / L(i,i);
end

%Bwd substitution, solve Ux=z
x = zeros(n,1);

for i = 0:n-1
    z_i_Old = z(n-i);
    for j = n+1-i:n
        z_i_Old = z_i_Old - U(n-i,j) * x(j);
    end
    x(n-i) = z_i_Old / U(n-i,n-i);
end

clearvars -except A L U x b
```

The entries of L^{-1} are given by the following formula (and it is not hard to verify: start from the columns or rows with least nonzero entries and gradually move on. I will omit the verification):

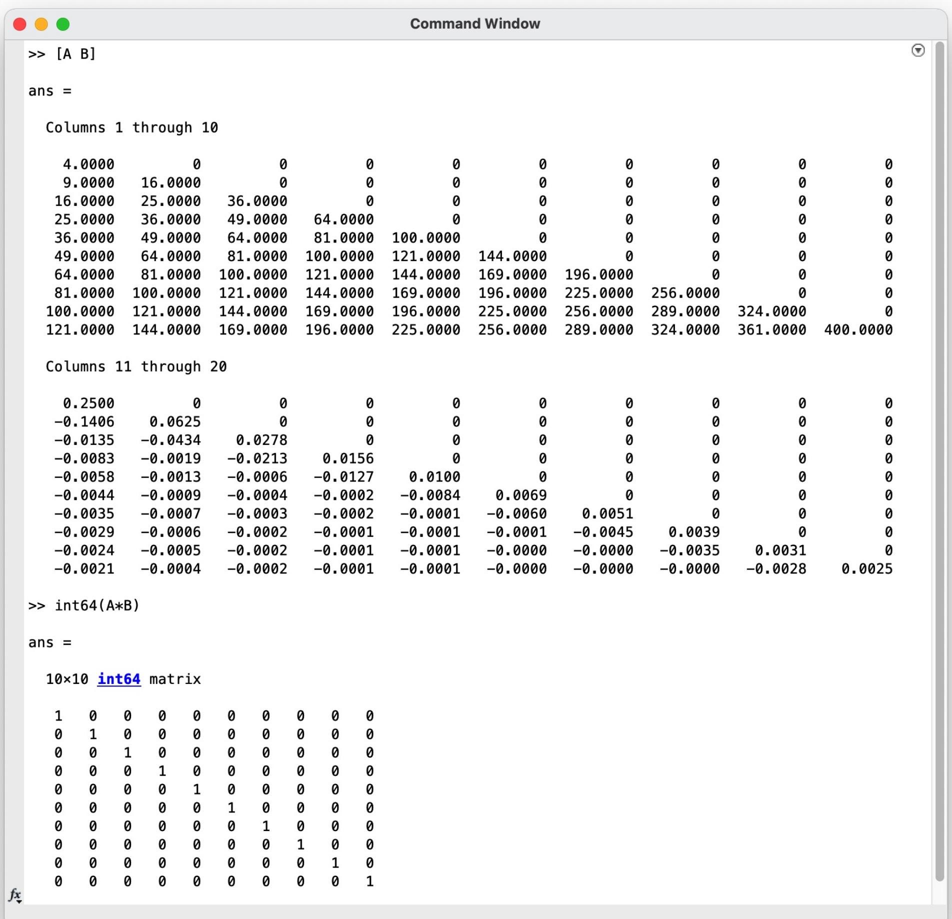
$$L^{-1}_{ij} = \begin{cases} 0 & i < j \\ 1/L_{ii} & i = j \\ -\left[\sum_{k=j}^{i-1} L_{ik} L^{-1}_{kj}\right] / L_{ii} & i > j. \end{cases}$$

```
A = input("What is the matrix?");
n = size(A);
B = zeros(n);

for j = 1:n
    B(j,j) = 1 / A(j,j);
    for i = j+1:n
        B(i,j) = - sum(A(i,j:i-1).* B(j:i-1,j).) /
            A(i,i);endend
```

I finally realized that I don't need another `for` loop to describe something like $x_i \leftarrow \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j\right) / a_{ii}$; all I needed instead was a dot product, i.e.,

```
x(i) = b(i) - sum(A(i,1:i-1) .* x(1:i-1).') / A(i,i);
```

Anyway, the result is as follows:



```
Command Window
>> [A B]

ans =

  Columns 1 through 10

    4.0000         0         0         0         0         0         0         0         0         0
    9.0000   16.0000         0         0         0         0         0         0         0         0
   16.0000   25.0000   36.0000         0         0         0         0         0         0         0
   25.0000   36.0000   49.0000   64.0000         0         0         0         0         0         0
   36.0000   49.0000   64.0000   81.0000  100.0000         0         0         0         0         0
   49.0000   64.0000   81.0000  100.0000  121.0000  144.0000         0         0         0         0
   64.0000   81.0000  100.0000  121.0000  144.0000  169.0000  196.0000         0         0         0
   81.0000  100.0000  121.0000  144.0000  169.0000  196.0000  225.0000  256.0000         0         0
  100.0000  121.0000  144.0000  169.0000  196.0000  225.0000  256.0000  289.0000  324.0000         0
  121.0000  144.0000  169.0000  196.0000  225.0000  256.0000  289.0000  324.0000  361.0000  400.0000

  Columns 11 through 20

    0.2500         0         0         0         0         0         0         0         0         0
   -0.1406    0.0625         0         0         0         0         0         0         0         0
   -0.0135   -0.0434    0.0278         0         0         0         0         0         0         0
   -0.0083   -0.0019   -0.0213    0.0156         0         0         0         0         0         0
   -0.0058   -0.0013   -0.0006   -0.0127    0.0100         0         0         0         0         0
   -0.0044   -0.0009   -0.0004   -0.0002   -0.0084    0.0069         0         0         0         0
   -0.0035   -0.0007   -0.0003   -0.0002   -0.0001   -0.0060    0.0051         0         0         0
   -0.0029   -0.0006   -0.0002   -0.0001   -0.0001   -0.0001   -0.0045    0.0039         0         0
   -0.0024   -0.0005   -0.0002   -0.0001   -0.0001   -0.0000   -0.0000   -0.0035    0.0031         0
   -0.0021   -0.0004   -0.0002   -0.0001   -0.0001   -0.0000   -0.0000   -0.0000   -0.0028    0.0025

>> int64(A*B)

ans =

  10×10 int64 matrix

   1   0   0   0   0   0   0   0   0   0
   0   1   0   0   0   0   0   0   0   0
   0   0   1   0   0   0   0   0   0   0
   0   0   0   1   0   0   0   0   0   0
   0   0   0   0   1   0   0   0   0   0
   0   0   0   0   0   1   0   0   0   0
   0   0   0   0   0   0   1   0   0   0
   0   0   0   0   0   0   0   1   0   0
   0   0   0   0   0   0   0   0   1   0
   0   0   0   0   0   0   0   0   0   1
```

Textbook Problems

4.2.1 (a) Recall the Gaussian-Jordan elimination.

$$\left[\begin{array}{cccc|cccc} u_{11} & u_{12} & \cdots & u_{1n} & 1 & & & \\ & u_{22} & \cdots & u_{2n} & & 1 & & \\ & & \ddots & \vdots & & & \ddots & \\ & & & u_{nn} & & & & 1 \end{array}\right]$$

If we apply it to an invertible upper triangular matrix, then we automatically begin with the back substitution stage. For the matrix on the right, it is impossible for any entry below the diagonal to become nonzero, as we are always one row by another row below it by definition of back substitution. Therefore throughout the Gaussian-Jordan process, the right matrix remains upper triangular and that, of course, includes the final step where U on the left becomes I and I on the right becomes U^{-1}. □

(b) Following a similar argument above it's immediate that the Gaussian-Jordan elimination of an invertible lower triangular matrix is lower triangular, so it suffices to show that, if L is unit lower triangular then L^{-1} has 1's along its entries. Indeed, if $LL^{-1} = I$, then looking at I_{ii} (i.e., the diagonal entries of I) gives

$$\sum_{k=1}^{n} L_{ik} L_{ki}^{-1} = L_{ii} L_{ii}^{-1} = 1 \implies L_{ii}^{-1} = 1/L_{ii} = 1,$$

since all other terms of the summation become 0 because either $k > i$ or $k < i$. The claim then follows. □

(c) WLOG assume A, B are $n \times n$ upper triangular matrices (the lower-triangular case is highly analogous). Suppose $AB = C$. It follows that

$$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}.$$

Notice that $A_{ik} = 0$ when $i > k$ and $B_{kj} = 0$ when $k > j$. If $i > j$ then there is no k satisfying $i \leqslant k$ and $k \leqslant j$, so each term $A_{ik}B_{kj}$ is inevitably 0, i.e., $C_{ij} = 0$. Therefore C is upper triangular. □

4.2.6 Suppose A is factorizable with

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ & U_{22} \end{bmatrix}.$$

Immediately we see that $L_{11}U_{11} = 0$ so either $L_{11} = 0$ or $U_{11} = 0$. If it is the former case then we have a contradiction that $L_{11}U_{11} + 0 \cdot U_{22} = A_{12} = 1$, and if it's latter case we have another contradiction that $L_{11}U_{11} + L_{21} \cdot 0 = A_{21} = 0$. Therefore A does not admit an LU-factorization.

4.2.16 Suppose $A_{n\times n} = LU$ and is invertible. Immediately we see that L and U have no zero diagonal entry, so they are invertible. Let A_k, L_k, U_k be the $k \times k$ leading principal minors of these matrices, respectively. Immediately we see that L_k, U_k have no zero diagonal entries so they are invertible. It remains to notice that $A_k = L_k U_k$:

$$\begin{bmatrix} L_{11} & & & \\ L_{21} & L_{22} & & \\ \vdots & \vdots & \ddots & \\ L_{n1} & L_{n2} & \cdots & L_{nn} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} & \cdots & U_{1n} \\ & U_{22} & \cdots & U_{2n} \\ & & \ddots & \cdots \\ & & & U_{nn} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{nn} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{bmatrix}.$$

Notice that

$$A_{ij} = \sum_{m=1}^{k} L_{im} U_{mk} = \sum_{m=1}^{\min\{i,j\}} L_{im} U_{mk}$$

so indeed the value of A_{ij} is determined only by entries of L_k and U_k. Since L_k and U_k are both invertible, A_k also is, and the claim follows. $\square$

4.2.22 For $\Longrightarrow$: suppose A is symmetric, real, and positive definite. By the Cholesky Theorem $A = LL^T$ for some L with positive diagonal. Therefore the row vectors of L are linearly independent, and this is precisely the set of vectors we are looking for.

For $\Longleftarrow$, suppose we have a set of linearly independent vectors. We can then form a matrix M whose row vectors are these vectors. It follows that $A = M^T M$. Then, A is positive definite because, for any $x \in \mathbb{R}^n$, $x^T Ax = x^T M^T Mx = (Mx)^T(Mx)$ which equals 0 if and only if $Mx = 0$. If $x^T Ax = 0$, since M is invertible (because it has full row rank), we have $Mx = 0 \Longrightarrow x = 0$. Hence A is positive definite. $\square$

4.2.27 For $\Longrightarrow$: if A is positive definite and B nonsignular, then $x^T B$ is a nonzero vector for any nonzero vector x. Then $x^T BABx = (Bx)^T A(Bx) > 0$, as desired. (Of course if $x = 0$ then $x^T BAB^T x = 0$.)

For $\Longleftarrow$: suppose BAB^T is nonsingular. Clearly B is nonsingular; otherwise for some nonzero x we have $Bx = 0$ and $x^T BAB^T x = (Bx)^T A(Bx) = 0$, contradicting BAB^T's positive definiteness. Once again, since B is nonsingular, so is B^T, and thus for any nonzero vector v there exists some y such that $B^T y = v$. Then,

$$v^T Av = (B^T y)^T A(B^T y) = y^T BAB^T y > 0.$$

Therefore A is positive definite. $\square$

4.2.34 If A admits a Cholesky factorization, then $\det(A) = \det(L)\det(L^T) = \det(L)^2$ for some nonsingular lower triangular L. Hence $\det(A) > 0$.

4.2.40 suppose $A = LL^T = MM^T$. First notice that the inverse of the transpose is the transpose of the inverse of a matrix, should they exist, i.e., for nonsingular A we have $(A^T)^{-1} = (A^{-1})^T$. For convenience we denote this by A^{-T}. Then

$$I = L^{-1}LL^T L^{-T} = L^{-1}MM^T L^{-T} = (L^{-1}M)(L^{-1}M)^T \Longrightarrow (L^{-1}M) = (L^{-1}M)^{-T}.$$

Notice that $L^{-1}M$ is lower triangular (cf. problem 1) whereas $(L^{-1}M)^{-T}$ is upper triangular! Therefore they have to be diagonal matrices and since $(L^{-1}M)(L^{-1}M)^T = I$, the diagonal entries must be ± 1. Since $M = L(L^{-1}M)$, one concludes that the entries of M differ from those of L by at most signs, but since we are only looking at Cholesky factorization with positive diagonals, $L = M$, as claimed. $\square$

4.2.52 No. Consider $\begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}$, clearly a symmetric matrix with minors 0 and 0. However, $\begin{bmatrix} \epsilon & 0 \\ 0 & \epsilon - 1 \end{bmatrix}$ has determinant $\epsilon(\epsilon - 1)$ which is negative for small ϵ. Therefore this property is <u>not</u> preserved.

4.2.54 If A is symmetric positive semidefinite, then $|a_{ij}| \leqslant \sqrt{a_{ii}a_{jj}}$: consider $v \in \mathbb{R}^n$ with entries 0 with the exception of $v_i = x$ and $v_j = 1$. Then $v^T Av = a_{ii}x^2 + 2a_{ij}x + a_{jj}$ ($a_{ji} = a_{ij}$ by symmetry). Since A is positive semidefinite, this quadratic equation has at most one root and thus $2a_{ij}^2 \leqslant a_{ii}a_{jj}$ and $|a_{ij}| \leqslant \sqrt{a_{ii}a_{jj}}$. Therefore, if a diagonal element of A is zero, the corresponding row and column must also be 0, and we can simply skip the original steps involved in Cholesky factorization. Other than that, carrying out the Cholesky factorization would still give us $A = LL^T$, the only difference being that L may have zero diagonal entries. $\square$

4.2.57 No. Consider again $\begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}$: nonnegative leading principal minors but not positive semidefinite:

$$\begin{bmatrix} a \\ b \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = -b^2 \leqslant 0.$$