신호처리를 위한 행렬 계산 Programming Assignment 3

20180490 이재헌

1. Code implementation

a) LDL^T *Factorization.*

LDL decomposition에서 L과 D의 각 요소의 공식은 아래와 같다.

LDL decomposition [edit]

An alternative form, eliminating the need to take square roots when **A** is symmetric, is the symmetric indefinite factorization[17]

$$\mathbf{A} = \mathbf{LDL}^\mathrm{T} = \begin{pmatrix} 1 & 0 & 0 \\ L_{21} & 1 & 0 \\ L_{31} & L_{32} & 1 \end{pmatrix} \begin{pmatrix} D_1 & 0 & 0 \\ 0 & D_2 & 0 \\ 0 & 0 & D_3 \end{pmatrix} \begin{pmatrix} 1 & L_{21} & L_{31} \\ 0 & 1 & L_{32} \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} D_1 & & (\text{symmetric}) \\ L_{21}D_1 & L_{21}^2 D_1 + D_2 & \\ L_{31}D_1 & L_{31}L_{21}D_1 + L_{32}D_2 & L_{31}^2 D_1 + L_{32}^2 D_2 + D_3 . \end{pmatrix}.$$

The following recursive relations apply for the entries of **D** and **L**:

$$D_j = \underset{④}{A_{jj}} - \sum_{k=1}^{j-1} \underset{⑤}{L_{jk}^2 D_k},$$

$$L_{ij} = \frac{1}{\underset{③}{D_j}} \left(\underset{①}{A_{ij}} - \sum_{k=1}^{j-1} \underset{②}{L_{ik} L_{jk} D_k} \right) \quad \text{for } i > j.$$

This works as long as the generated diagonal elements in **D** stay non-zero. The decomposition is then unique. **D** and **L** are real if **A** is real.

이를 for문을 활용해 구현해주면 다음과 같은 코드로 완성할 수 있다.

```
n = size(x,1);
L = eye(n); % L is unit lower triangular
D = eye(n);
D(1,1) = x(1,1);
% get L(i,j) for i>j
for i=2:n
    for j=1:i-1
    ① L(i,j) = x(i,j);
        for k=1:j-1
  ② (       L(i,j) = L(i,j) - L(i,k)*L(j,k)*D(k,k);
        end
    ③ L(i,j) = L(i,j)/D(j,j);
    end
    % get Diagonal entry for D
  ④ D(i,i) = x(i,i);
    for j=1:i-1
⑤ (     D(i,i) = D(i,i) - L(i,j)^2*D(j,j);
    end
end
```

공식에 해당하는 코드의 부분을 숫자로 표기하였다.

b) $Cholesky\ Factorization(LL^T\ Factorization).$

Cholesky decomposition에서 L의 각 요소의 공식은 아래와 같다.

The Cholesky–Banachiewicz and Cholesky–Crout algorithms

If we write out the equation

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T = \begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} L_{11} & L_{21} & L_{31} \\ 0 & L_{22} & L_{32} \\ 0 & 0 & L_{33} \end{pmatrix}$$

$$= \begin{pmatrix} L_{11}^2 & & (\text{symmetric}) \\ L_{21}L_{11} & L_{21}^2 + L_{22}^2 & \\ L_{31}L_{11} & L_{31}L_{21} + L_{32}L_{22} & L_{31}^2 + L_{32}^2 + L_{33}^2 \end{pmatrix},$$

we obtain the following:

$$\mathbf{L} = \begin{pmatrix} \sqrt{A_{11}} & 0 & 0 \\ A_{21}/L_{11} & \sqrt{A_{22} - L_{21}^2} & 0 \\ A_{31}/L_{11} & (A_{32} - L_{31}L_{21})/L_{22} & \sqrt{A_{33} - L_{31}^2 - L_{32}^2} \end{pmatrix}$$

and therefore the following formulas for the entries of **L**:

$$L_{j,j} = (\pm)\sqrt{A_{j,j} - \sum_{k=1}^{j-1} L_{j,k}^2},$$

$$L_{i,j} = \frac{1}{L_{j,j}}\left(A_{i,j} - \sum_{k=1}^{j-1} L_{i,k}L_{j,k}\right) \quad \text{for } i > j.$$

이를 코드로 구현하면 아래와 같다.

```
n = size(x,1);
L = zeros(n);
L(1,1) = sqrt(x(1,1)); % initialize
%get L(i,j) for i>j
for i=2:n
    for j=1:i-1
    1   L(i,j) = x(i,j);
    2   for k=1:j-1
            L(i,j) = L(i,j) - L(i,k)*L(j,k);
        end
    3   L(i,j) = L(i,j)/L(j,j);
    end
    % get the diagonal enrty L(i,i)
4   L(i,i) = x(i,i);
5   for j=1:i-1
        L(i,i) = L(i,i) - L(i,j)^2;
    end
6   L(i,i) = sqrt(L(i,i));
end
```

마찬가지로 공식에 해당하는 코드의 부분을 숫자로 표시하였다.

c) *LU decomposition with partial pivioting*.

```
n = size(A,1);
Ak = A;
L = eye(n); % L is unit lower triangular
U = zeros(n);
P = eye(n);
for k = 1:n-1
    [~,r] = max(abs(Ak(k:end,k))); % r = the index of the maximum of a part of column
    r = n-(n-k+1)+r; % consider the relative index
    Ak([k r],:) = Ak([r k],:); % exchange the row
    P([k r],:) = P([r k],:);
    L([k r],:) = L([r k],:);
    for i = k+1:n
        L(i,k) = Ak(i,k) / Ak(k,k); % get tau
        for j = 1:n
            U(k,j) = Ak(k,j);
            Ak(i,j) = Ak(i,j) - L(i,k)*Ak(k,j); % subtraction of the rows below the pivot
        end
    end
end
U(:,end) = Ak(:,end);
```

K를 증가시켜가며 pivot의 column의 값을 decreasing order로 정렬하기 위해 행들의 위치를 서로 바꿔준다. 이 행의 교환을 Ak, P, L에 모두 적용해주어야 된다. 그 후 pivot의 값을 이용해 Gaussian transformation을 하듯이 τ_k의 값들을 구해 이를 pivot의 row에 곱해 각 row에서 빼주면 그 행이 0으로 맞춰지고, τ_k는 L의 요소가 된다.

2. Test.m Screenshot.

1.

```
LDL factorization
L:
    1.0000         0         0         0
    0.4749    1.0000         0         0
    0.7865    0.5143    1.0000         0
    0.6894    0.8821    0.9700    1.0000

D:
    1.7177         0         0         0
         0    0.7643         0         0
         0         0    0.4969         0
         0         0         0    0.0057

L * D * L':
    1.7177    0.8158    1.3509    1.1841
    0.8158    1.1518    1.0347    1.2366
    1.3509    1.0347    1.7615    1.7600
    1.1841    1.2366    1.7600    1.8842
```

```
x:
    1.7177    0.8159    1.3509    1.1841
    0.8158    1.1518    1.0347    1.2366
    1.3509    1.0347    1.7615    1.7600
    1.1841    1.2366    1.7600    1.8842
```

2.

```
Cholesky factorization
L:
    1.3106         0         0         0
    0.6225    0.8743         0         0
    1.0307    0.4496    0.7049         0
    0.9035    0.7712    0.6838    0.0753

L * L':
    1.7177    0.8158    1.3509    1.1841
    0.8158    1.1518    1.0347    1.2366
    1.3509    1.0347    1.7615    1.7600
    1.1841    1.2366    1.7600    1.8842

x:
    1.7177    0.8159    1.3509    1.1841
    0.8158    1.1518    1.0347    1.2366
    1.3509    1.0347    1.7615    1.7600
    1.1841    1.2366    1.7600    1.8842
```

3.

```
LU factorization
L:
    1.0000         0         0         0
    0.4749    1.0000         0         0
    0.7865    0.5142    1.0000         0
    0.6894    0.8821    0.9700    1.0000

U:
    1.7177    0.8159    1.3509    1.1841
         0    0.7643    0.3931    0.6742
         0   -0.0000    0.4969    0.4820
         0         0         0    0.0057

L * U:
    1.7177    0.8159    1.3509    1.1841
    0.8158    1.1518    1.0347    1.2366
    1.3509    1.0347    1.7615    1.7600
    1.1841    1.2366    1.7600    1.8842

P * x:
    1.7177    0.8159    1.3509    1.1841
    0.8158    1.1518    1.0347    1.2366
    1.3509    1.0347    1.7615    1.7600
    1.1841    1.2366    1.7600    1.8842
```

3. Uniqueness of LDL^T, LL^T

1) LL^T

Cholesky Decomposition is unique up to the sign of its diagonal entries.

Pf) Assume $A = LL^T = MM^T$ are two Cholesky decompositions of A.

$$I = L^{-1}MM^TL^{-T} = (L^{-1}M)(L^{-1}M)^T$$

$L^{-1}M$ is a lower triangular matrix and thus $(L^{-1}M)^T$ is an upper triangular matrix.

$$D \coloneqq L^{-1}M = (L^{-1}M)^T$$

D is both lower triangular and upper triangular. Therefore, D is diagonal.

$$D^2 = I \to D = diag(\pm 1, \pm 1, \ldots, \pm 1)$$

$$M = LD$$

L and D differ by the sign of their columns.

Therefore, if we restrict the diagonal entries of L to be positive, L is unique.

2) LDL^T

From the lecture, we know the following.

LDL^T Factorization

If $A \in \boldsymbol{R}^{n\times n}$ is **symmetric,**
and $\det(A(1{:}k, 1{:}k)) \neq 0$, for $k = 1{:}n-1$
then there exist a **unique unit lower triangular** matrix
L, and a **unique diagonal** matrix $D = \mathbf{diag}\{d_1, \cdots, d_n\}$
such that $A = LDL^T$.

We can prove that D is unique.

suppose
$A = L_1D_1L_1^* = L_2D_2L_2^*$

where D_1, D_2 are diagonal matrices and L_1, L_2 are lower triangular with 1's on the diagonal.
$D_2 = L_2^{-1}L_1D_1L_1^*(L_2^*)^{-1} = \left(L_2^{-1}L_1\right)D_1\left(L_2^{-1}L_1\right)^*$
$= R^*D_1R$
with $R := \left(L_2^{-1}L_1\right)^*$ which is upper triangular with ones on the diagonal

$\implies D_2R^{-1} = R^*D_1$
the LHS is uppper triangular while the RHS is lower triangular, which in fact implies each side is diagonal. And since R has ones on the diagonal:
$\implies D_2 = D_2R^{-1} = R^*D_1 = D_1$

alternatively, looking at $D_2R^{-1} = R^*D_1$ and examining component (k,k) of each side
$\implies d_{k,k}^{(2)} \cdot 1 = d_{k,k}^{(1)} \cdot 1 \implies D_2 = D_1$