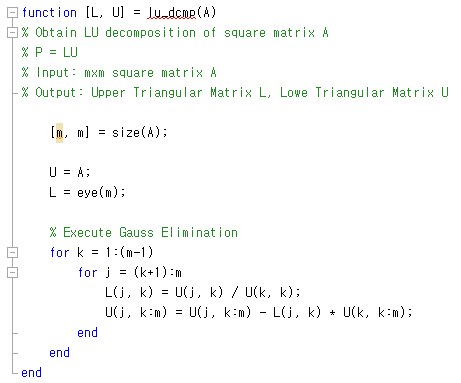
**MAS364 HW6**

20150651 장강욱

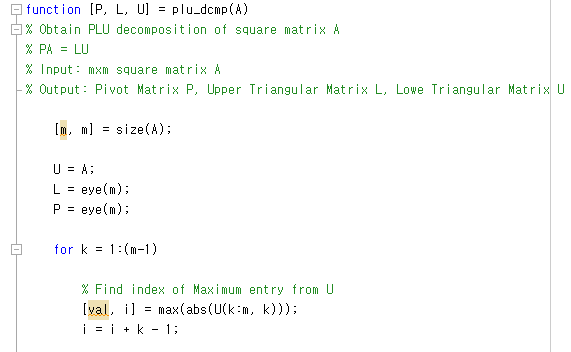
Dept: EE

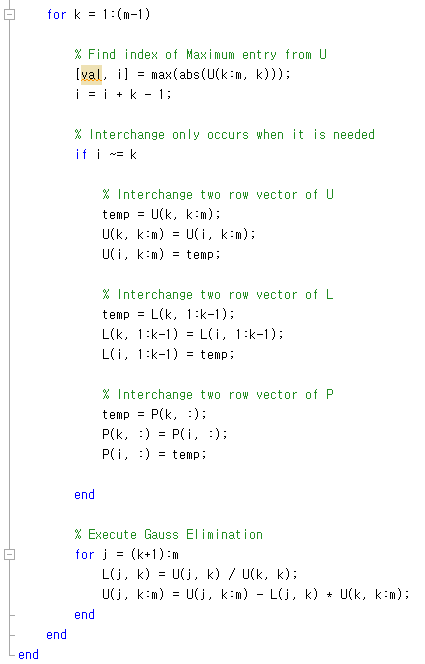
1. (a)

아래는 Pivoting이 없는 LU 분해에 대한 코드이다. 입력으로 들어오는 행렬은 항상 정사각행렬이라고 가정하였다.

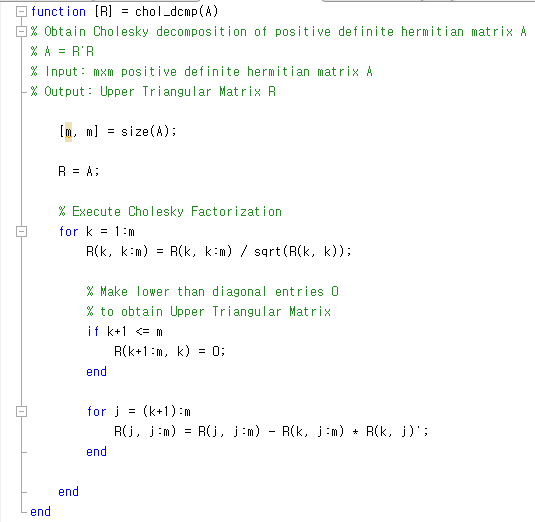


아래는 Pivoting이 있는 LU 분해(PLU 분해)에 대한 코드이다. 입력으로 들어오는 행렬은 항상 정사각행렬이라고 가정하였다. PLU 분해 중 행을 바꾸는 작업은 필요할 때에만 진행하여, 불필요한 연산을 줄였다.





아래는 Cholesky 분해에 대한 코드이다. 입력으로 들어오는 행렬은 항상 Positive Definite Hermitian 행렬이라 가정하였다. 결과행렬을 Upper Triangular 행렬로 만들기 위해, 대각 원소보다 낮은 부분에 있는 원소들을 0으로 만드는 작업이 포함되었다.

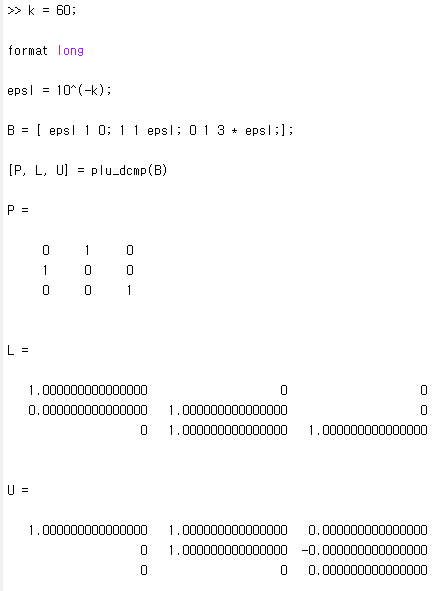
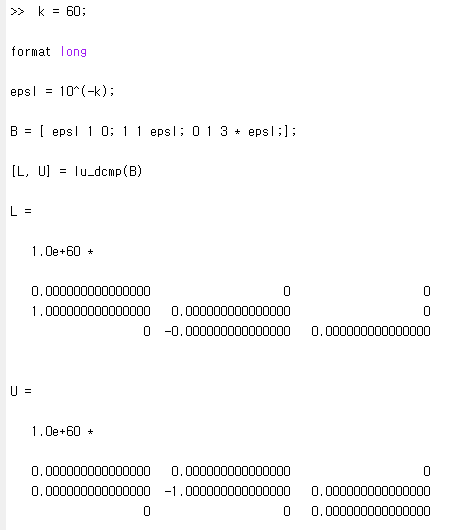
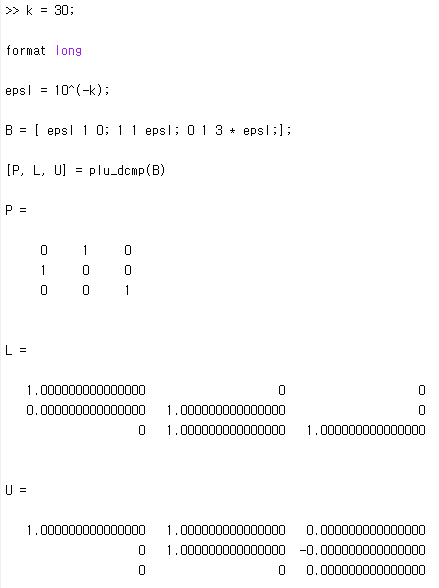
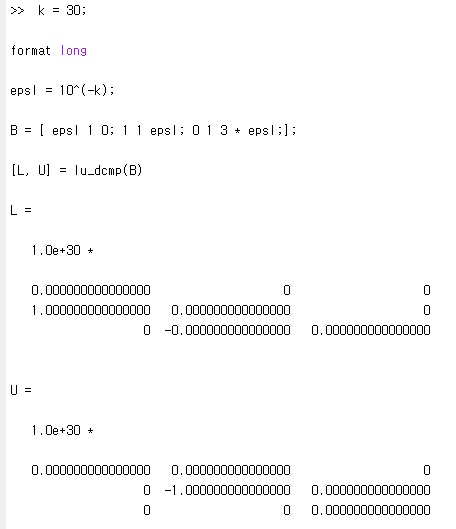
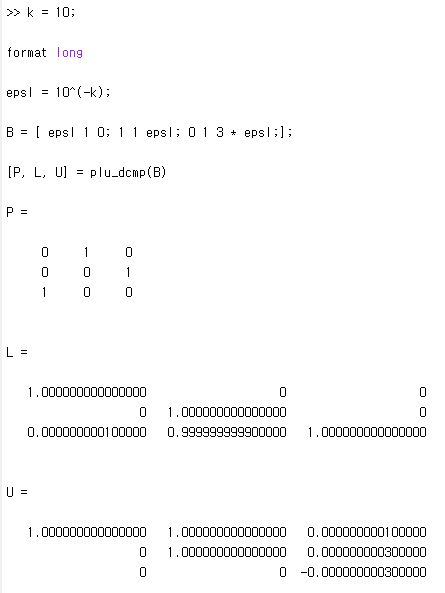
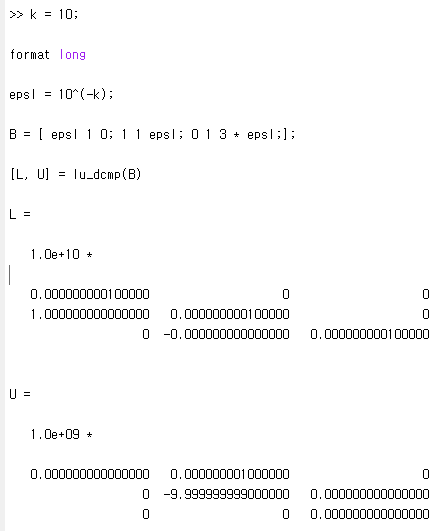


1. (b)

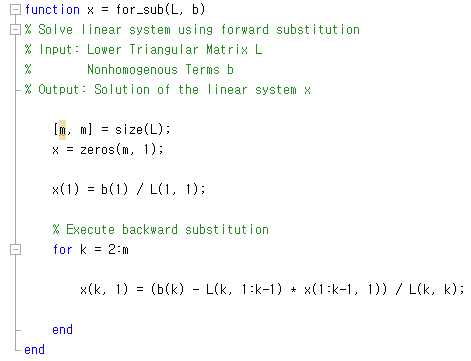
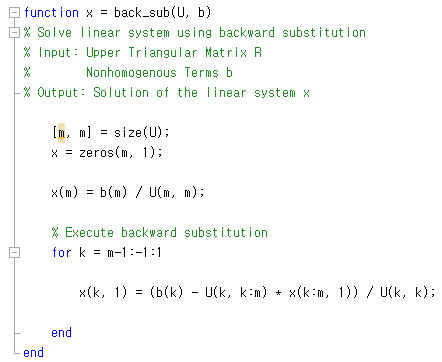
아래는 행렬 [DRW000019ac4c02 1 0; 1 1 3DRW000019ac4c04 ; 0 1 3DRW000019ac4c06 ;], DRW000019ac4c08=10^-k에서 k가 10, 30, 60일 때에 대해서 LU 및 PLU 분해의 결과이다. 왼쪽이 LU 분해, 오른쪽이 PLU 분해이며, 내려가는 순서로 k가 각각 10, 30, 60이다.

LU 분해의 결과는 굉장히 불안정한 것을 확인할 수 있다. 특히, DRW000019ac4c0a는 모든 k에 대해서 굉장히 큰 값(약 DRW000019ac4c0c)을 가진다. 또 k=10인 경우에는 DRW000019ac4c0e와 DRW000019ac4c10가 1임을 확인할 수 있지만, k=30, 60인 경우에는 컴퓨터의 Truncation으로 해당 원소가 0이라는 잘못된 결과가 나온다. 이는 결과 L과 U 앞에 큰 값이 곱해져 있어, 컴퓨터의 부동 소수점으로는 이를 정확히 표현할 수 없기 때문이다.

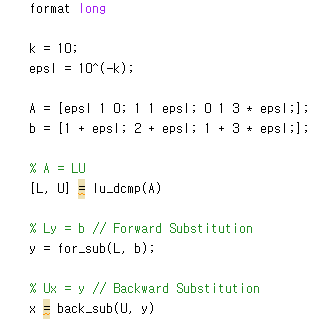
그러나 PLU 분해의 결과는 LU에 비해 k가 달라져도 일관적인 올바른 결과가 나옴을 확인할 수 있다. 이는 행렬의 Pivoting을 통해, 매우 작은 숫자로 나누는 연산을 피했기 때문이다.



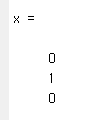
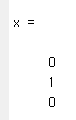
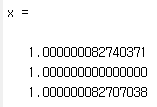
다음은 LU 분해와 PLU 분해에 Backward Substitution과 Forward Substitution 알고리즘을 더하여 선형 시스템의 해를 구하였다. 아래는 구현한 Backward Substitution과 Forward Substitution 알고리즘이다. 오른쪽이 Backward Substitution이고, 왼쪽이 Forward Substitution이다.



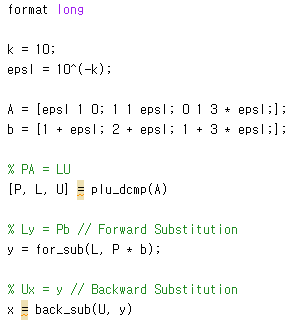
이 알고리즘에 조건을 바꾸어가며 LU 분해에 대한 해를 구하였다. 아래는 k=10, LU 분해로 해를 구한 예시 코드이다. 아래 코드에서 k 값을 다르게 하며 해 벡터 x를 구하였다.



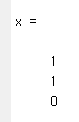
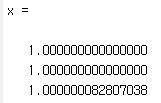
아래는 위 코드에 대한 결과이다. 왼쪽부터 k=10, 30, 60이다. k가 30, 60인 경우에 대해서 부정확한 해가 나왔음을 알 수 있다.



PLU 분해의 경우에는 b에 P를 곱해야 한다. 이를 고려한 예시코드가 아래에 있다.



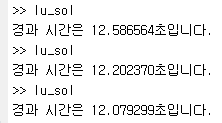
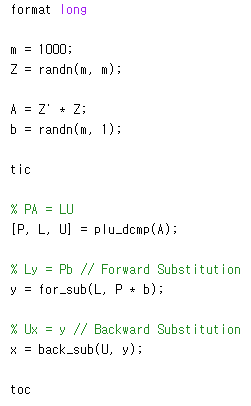
위 코드에 대한 해 벡터는 아래와 같다. 왼쪽부터 k=10, 30, 60이다. LU 분해와는 다르게, k가 30, 60인 경우에 대해서도 정확한 해가 나왔음을 알 수 있다.

EMB000019ac4bfd

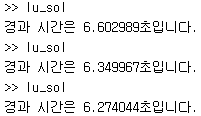
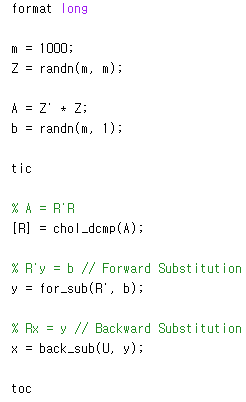
LU와 PLU 분해로 행렬을 직접 분해 해본 것과, 이들로부터 해를 구한 결과들로부터 Pivoting이 수반된 LU 분해가 더 정확함을 확인할 수 있다.

1. (c)

이 문제는 PLU 분해와 Cholesky 분해 중 어떤 것이 더 빠른가를 물어보고 있다. 아래는 문제에서 주어진 조건에 대해 PLU 분해를 진행하는 코드이다. tic, toc 구문으로 연산에 필요한 코드의 실행 시간을 측정하였다. 행렬의 크기가 1000x1000인 관계로 각 실행에 대한 실제 행렬의 분해 및 선형 시스템 해의 결과는 생략하였다.



아래는 문제에서 주어진 조건에 대해 Cholesky 분해를 진행하는 코드이다. tic, toc 구문으로 연산에 필요한 코드의 실행 시간을 측정하였다.



Cholesky 분해가 PLU 분해에 비해 소요시간이 약 절반 정도임을 확인할 수 있었다. 행렬 분해의 속도 면에서 Cholesky 분해가 PLU 분해보다 나음을 확인할 수 있다.