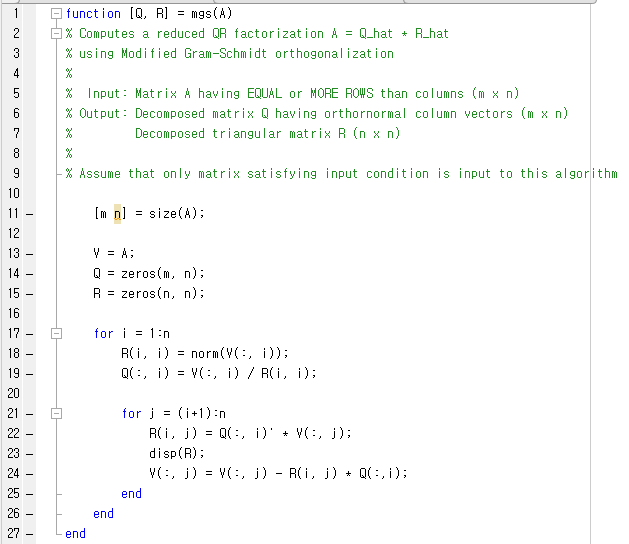
**MAS364 MATLAB HW #3**

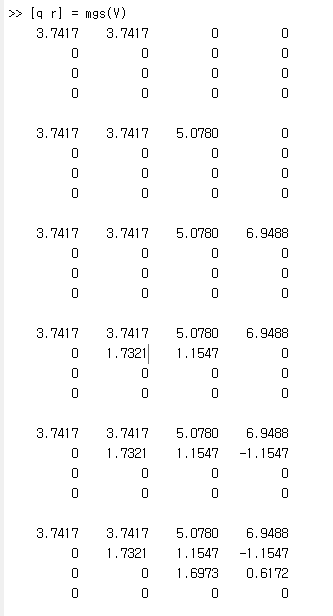
EE 20150651 장강욱

1.

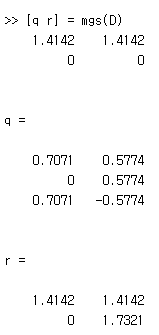
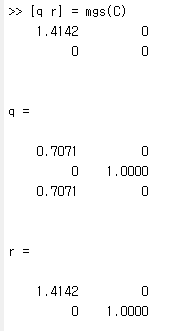
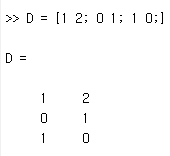
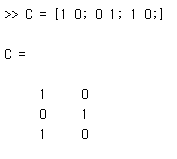
아래는 문제에 해당하는 코드이다. 문제에서 열보다 행의 개수가 더 많은 행렬에 대해서 QR 분해를 하라 하였으므로, 입력되는 행렬은 모두 주어진 조건을 만족하는 행렬만 입력한다 가정했다. 따라서 코드 내에서 Error Handling을 하지 않았다. 실제 코드에서는 23번째 줄의 disp(R); 행렬은 주석 처리하였다.



아래는 이 코드에 대한 결과이다. 상삼각행렬의 원소가 Modified Gram-Schmidt가 의도한 대로 채워짐을 확인할 수 있다.

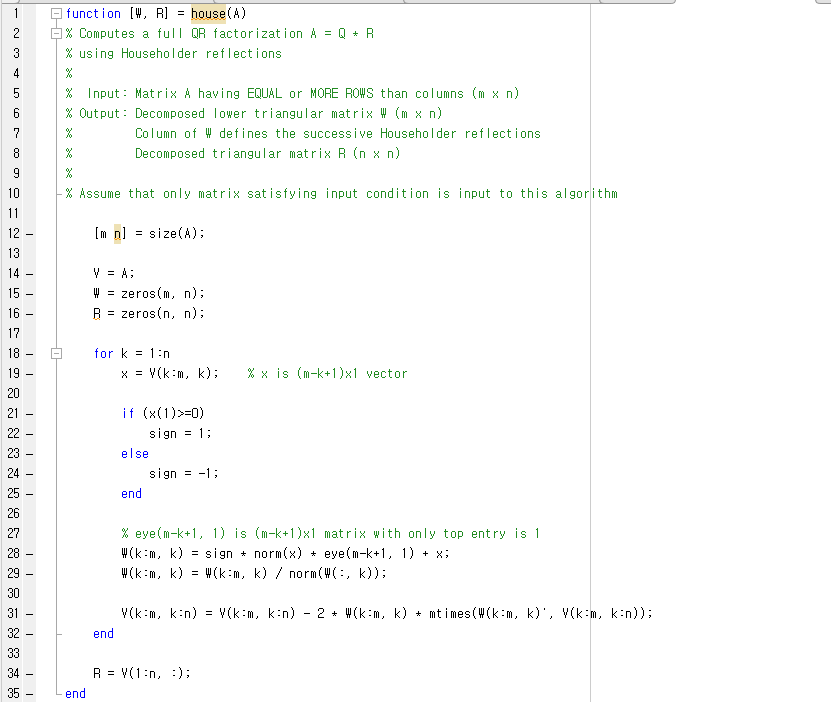


아래는 지난 과제(Ex.6.4)에서 등장했던 행렬들에 대해 직접 적용하였다. 코드의 결과가 직접 계산한 것과 같음을 확인할 수 있었다. C와 D는

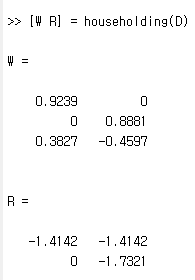
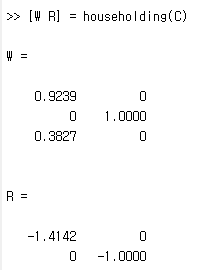
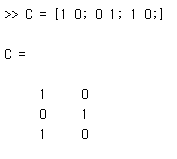
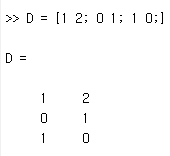


2. (a)

아래는 문제에 해당하는 코드이다. 문제에서 열보다 행의 개수가 더 많은 행렬에 대해서 QR 분해를 하라 하였으므로, 입력되는 행렬은 모두 주어진 조건을 만족하는 행렬만 입력한다 가정했다. 따라서 코드 내에서 Error Handling을 하지 않았다.

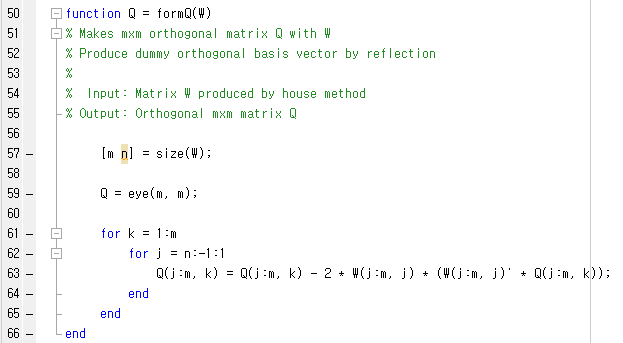


1에서 예시로 사용한 행렬 2개를 이 코드에도 사용해보았다. 아래는 그 결과이다.

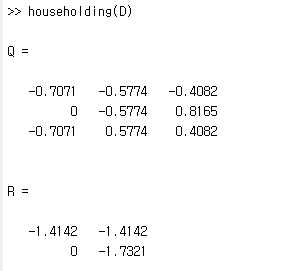
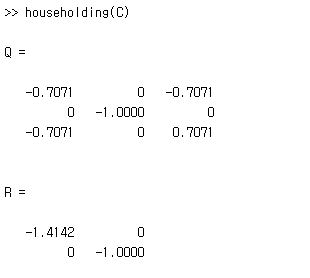
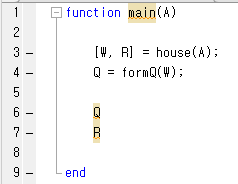


2. (b)

(a)에서 얻은 W 행렬에 Dummy Orthogonal 벡터를 추가하는 문제이다. 이에 대한 코드가 아래에 제시되어 있다.

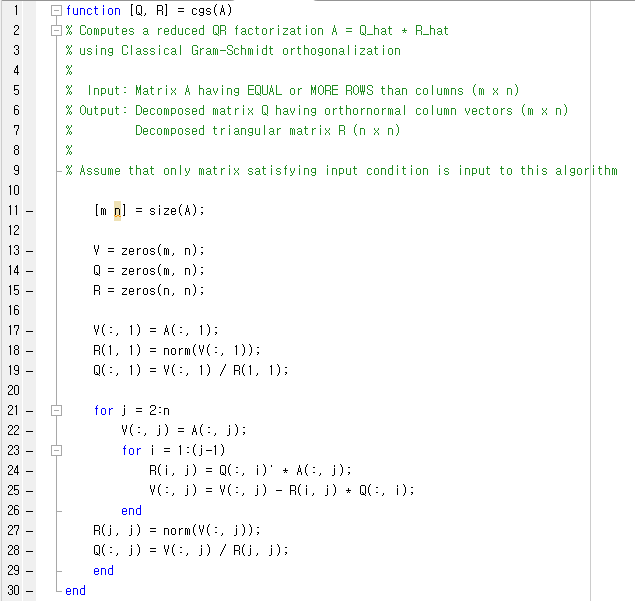


아래는 (a)와 (b)에서 구현한 모든 함수들을 실행하는 main 함수이며, 2.(a)의 행렬을 이에 적용했다.

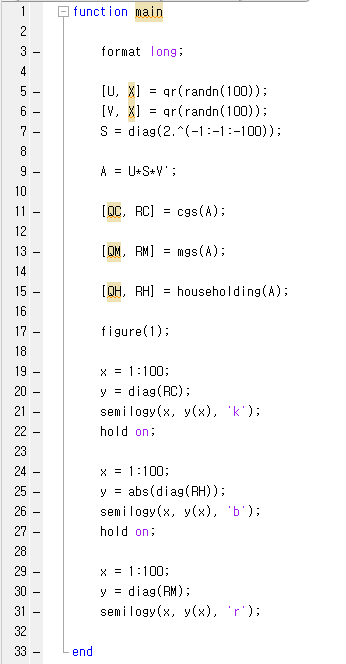


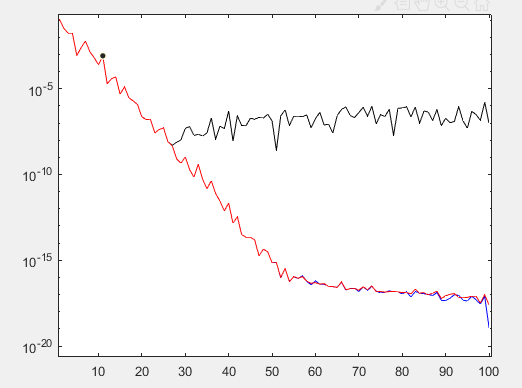
3.

1과 2에서 Modified Gram-Schmidt와 Householder 알고리즘은 각각 구현하였지만, Classical Gram-Schmidt는 구현하지 않았으므로, 먼저 구현하였다. 아래는 Classical Gram-Schmidt 알고리즘에 해당하는 코드이다.



구현한 세 알고리즘을 바탕으로, 책에 제시된 Fig 9.1을 그려보았다. 아래는 이에 필요한 코드와 결과 그래프이다.





검은색은 Classical Gram-Schmidt, 빨간색은 Modified Gram-Schmidt, 파란색은 House-holder 알고리즘이다. 그래프를 관찰했을 때, 안정성은 Classical Gram-Schmidt > Modified Gram-Schmidt & House-holder 알고리즘이다. 이것은 대각 성분이 로그 스케일의 어느 즈음에서 수렴하는지 확인함으로써 알 수 있다. 즉, Modified Gram-Schmidt & House-holder 알고리즘이 더욱 안정하다.