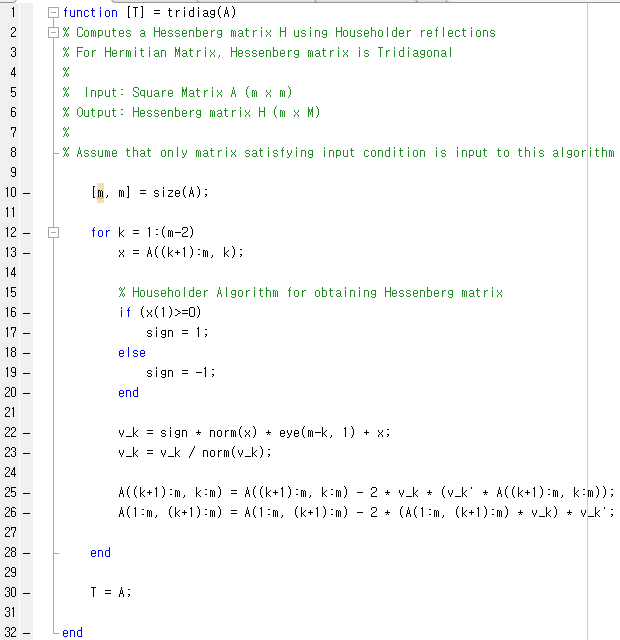
**MATLAB HW 9**

Dept: EE

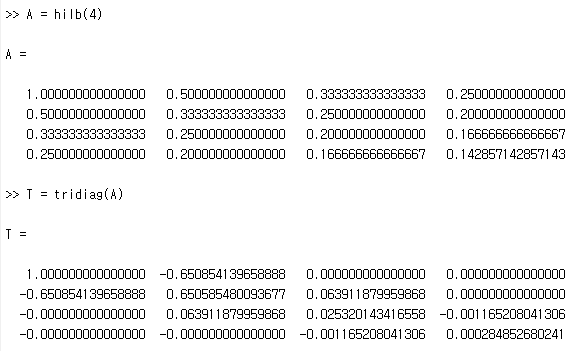
20150651 장강욱

1. (a)

MATLAB HW7에서 진행했었던 hessen 메소드를 그대로 옮겨 메소드 이름을 tridiag으로 변경했다.

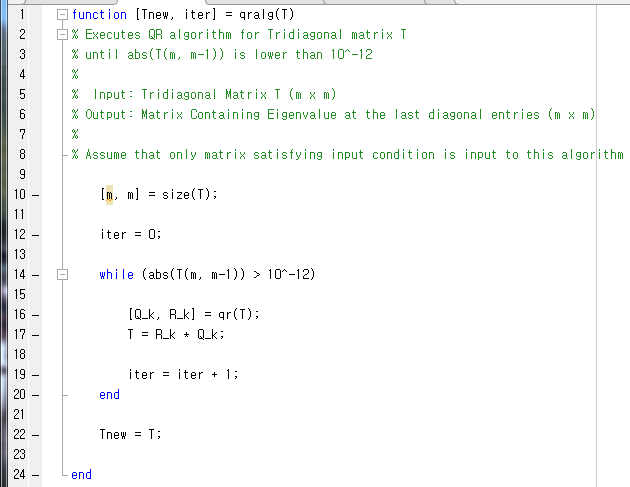


위 메소드를 A = hilb(4)에 적용하면 다음과 같은 결과이다.

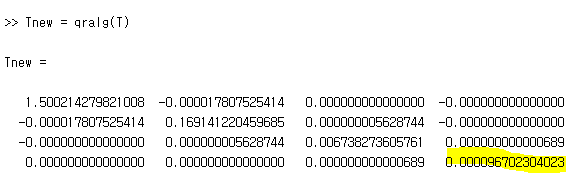


(b)

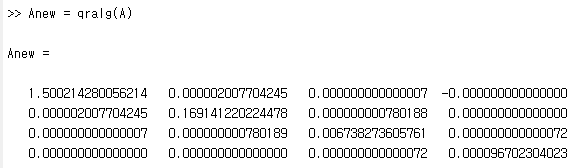
QR 알고리즘을 Matlab의 qr 메소드를 바탕으로 작성하였다.



이 메소드를 (a)의 출력인 Tridiagonal 행렬 T에 적용하면 다음과 같다.

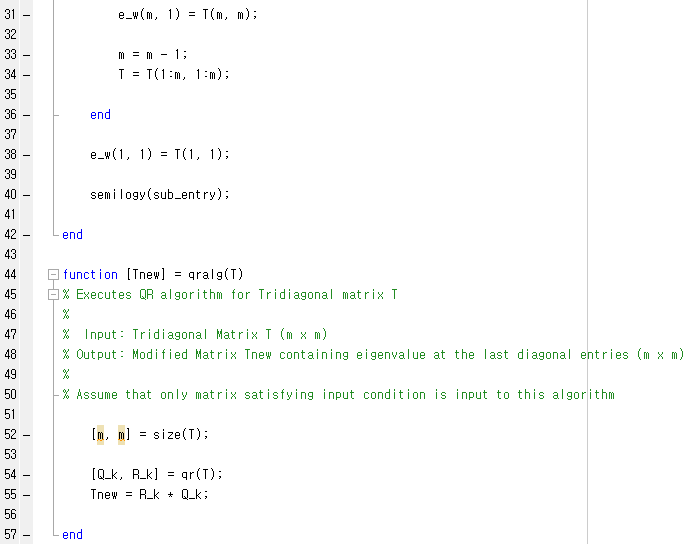
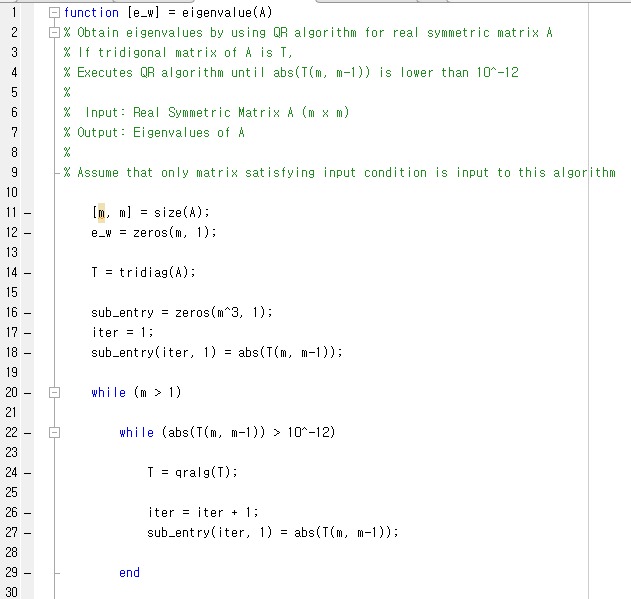


이 메소드를 A = hilb(4)에 적용하면 다음과 같다.

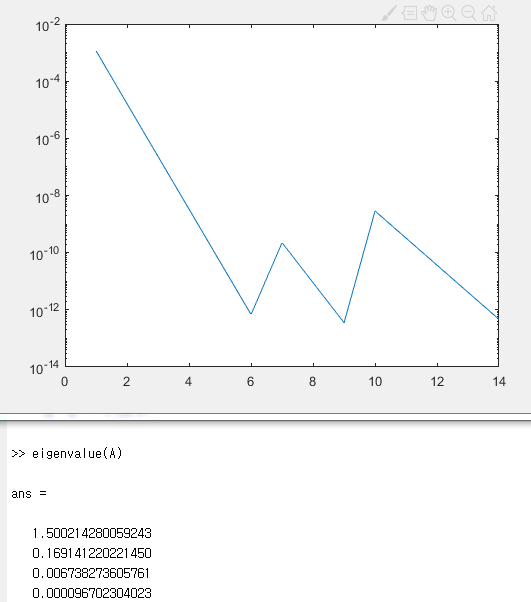


©

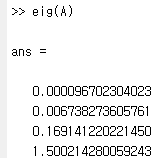
위 메소드들을 eigenvalue 메소드로 묶었다. tridiag 메소드는 (a)에서 그대로 가져왔고, qralg는 while문 조건을 없앤 채, 수정을 거쳐 아래에 첨부해 두었다. 없앤 while 문 조건은 eigenvalue 함수에 포함시켰다.



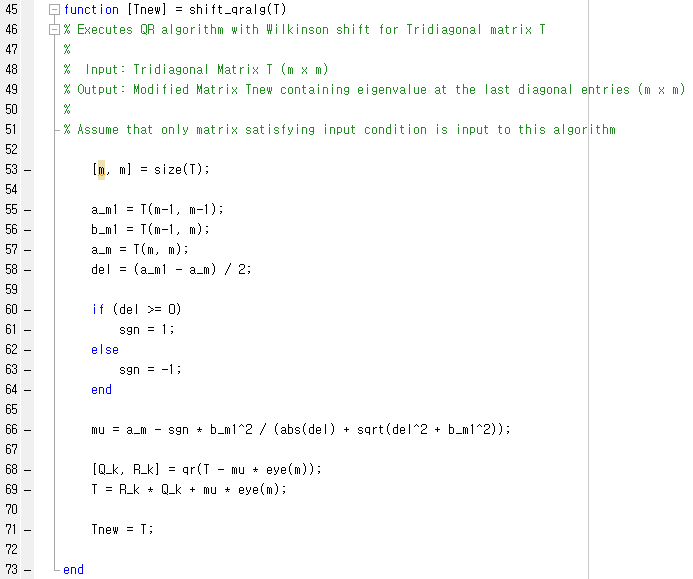
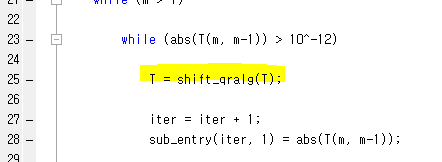
아래는 위 메소드를 실행한 결과이다.



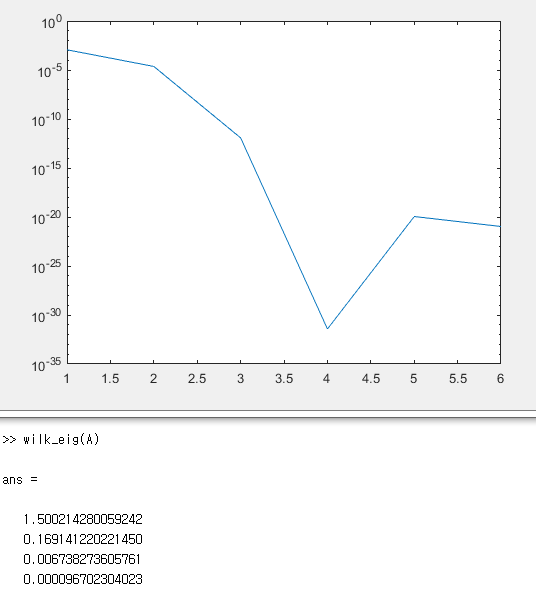
이 메소드를 Matlab의 eig 함수로 고윳값을 구하면 아래와 같으며, 완전히 같음을 알 수 있다. 결과 그래프 역시, 문제에서 예견한 것처럼 Sawtooth plot임을 알 수 있다.



(d) qralg 메소드를 Wilkinson shift를 이용하여 shif\_qralg 메소드로 수정하였다. 나머지는 ©에서와 같다. 아래는 본 코드 일부와, 수정한 qralg 메소드인 shift\_qralg 메소드이다.

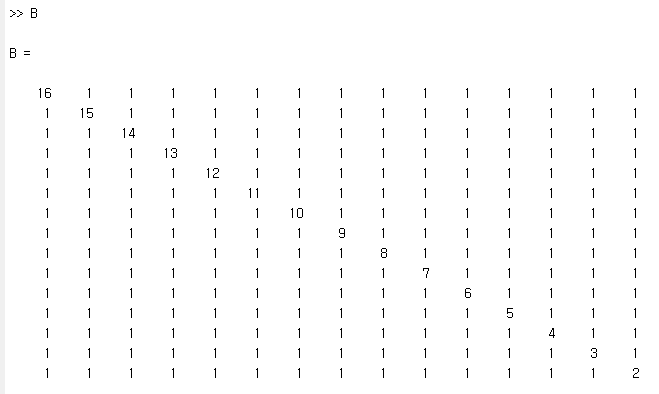


아래는 이 메소드의 그 결과이다. ©에서보다 반복횟수가 절반 이하로 줄어들었음을 확인할 수 있다. 구한 고윳값 역시 matlab eig 메소드로 구한 것과 완벽히 같다.

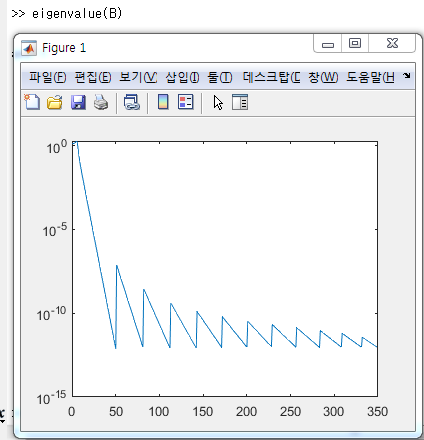


(e)

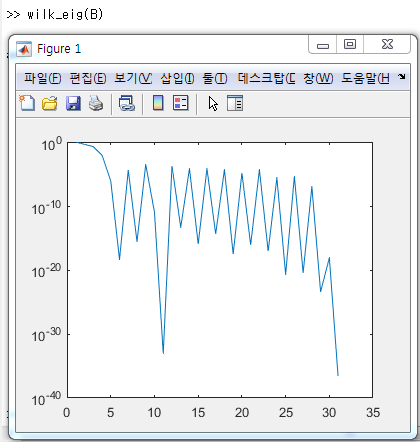
문제에서 제시한 행렬을 표현하면 아래와 같다.



아래는 일반 QR 알고리즘으로 구현한 메소드의 결과이다.



아래는 Shifted QR 알고리즘으로 구현한 메소드의 결과이다.



두 그래프에서 확인할 수 있듯이, Wilkinson Shift로 구한 고윳값은 반복 횟수가 현저히 적음을 알 수 있다. Pure QR 분해는 수렴 비율이 Linear이며, Wilkinson Shift는 Cubic이다. 고윳값 하나당 QR 분해의 반복횟수라는 지표는 해당 알고리즘의 수렴 비율이 선형일 때에만 의미가 있다. 따라서 Pure QR 분해에서는 의미가 있을 수 있겠지만, Wilkinson Shift에서는 의미가 없을 것이다.