Name: PARMESH YADAV

Roll No: 2020319

SCIENTIFIC COMPUTING (MTH373)

HOMEWORK – 3

Problem – 1

A1) To prove: Eigenvalues of a triangular matrix $A \in R^{m \times m}$ are exactly its diagonal elements.

Proof: Consider $A \in R^{m \times m}$ be an upper triangular matrix.

Therefore, A is of the form,
$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ 0 & a_{22} & \cdots & a_{2m} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & a_{mm} \end{bmatrix}$$

Calculating char. polynomial,
$$A - \lambda I = \begin{bmatrix} a_{11}-\lambda & a_{12} & \cdots & a_{1m} \\ 0 & a_{22}-\lambda & \cdots & a_{2m} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & a_{mm}-\lambda \end{bmatrix}$$

$$p_m(\lambda) = \det(A - \lambda I) = (a_{11}-\lambda)(a_{22}-\lambda)\cdots(a_{mm}-\lambda)$$
since, determinant of triangular matrix is the product of diagonal elements

Hence, the eigenvalues are $\{a_{11}, a_{22}, \ldots, a_{mm}\}$ i.e. exactly its diagonal elements.

Therefore, it can be similarly proved for lower triangular as well.

Problem – 2

A2) To prove : Rank of a non-defective matrix $A \in R^n$ is equal to no. of non-zero eigenvalues of A.

Proof : We know that,

$$A \text{ is non-defective}$$
$$\Rightarrow A \text{ is diagonalizable}$$
$$\Rightarrow AM = GM \text{ for all } \lambda \text{ of } A$$
$$\Rightarrow \text{No. of eigenvalues of } A = n$$

[AM : Algebraic Multiplicity
GM : Geometric Multiplicity]

By rank - nullity theorem,

$$\text{Rank}(A) + \text{Nullity}(A) = n,$$

and,

$$\text{Nullity}(A) = \dim[\text{Ker}(A)]$$

We know that elements of kernel of A are linear combinations of its eigenvectors corresponding to the 0 eigenvalue.

This implies, $\text{Rank}(A) = n - \text{Nullity}(A)$

Therefore, $\text{Nullity}(A) =$ Total no. of zero eigenvalues.

$\Rightarrow n - \text{Nullity}(A) =$ " " " non-zero "

$\Rightarrow \text{Rank}(A) =$ Total no. of non-zero eigenvalues.

Problem – 3

A3) We know, $u^T \cdot v = 1$ and $u, v \in R^m$

$$\Rightarrow u^T \cdot v \cdot v^{-1} = v^{-1}$$
$$u^T = v^{-1}$$
$$\Rightarrow u = (v^{-1})^T$$

Now, acc. to the question, $A = u \cdot v^T$

$$\Rightarrow A = (v^{-1})^T \cdot v^T$$
$$= [v^{-1} \cdot v]^T$$
$$= (I)^T$$
$$\Rightarrow A = I$$

Therefore, we know that the identity matrix has one eigenvalue $= 1$ with algebraic multiplicity $= m$.

Power method won't converge to the dominant eigenvalue - eigenvector pair because at every step, the vector will remain the same.

Problem – 4

A4) To prove: Show all eigenvalues of a real symmetric positive definitive matrix are all real and strictly > 0.

Proof: First, we have to prove that all eigenvalues of a symmetric matrix are real.

Therefore, let (λ, u) be an eigenvalue, eigenvector pair for A:

$$\Rightarrow Au = \lambda u$$

Let us compute, $\langle Au, u\rangle = \langle \lambda u, u\rangle$, now assuming $u \in \mathbb{C}^n$.

We have, $u^H A . u = u^H . \lambda . u$ $\quad \left[\because \langle u, v\rangle = v^H . u \quad \forall u, v \in \mathbb{C}^n\right]$

$$u^H A u = \lambda u^H u \text{ --- } (1)$$

$$\Rightarrow (u^H A u)^H = (\lambda u^H u)^H$$
$$u^H A^H u = u^H u \lambda^*$$
$$u^H A u = \lambda^* u^H u \text{ --- } (2) \quad [\because A^H = A]$$

By (1) and (2), $\lambda u^H u = \lambda^* u^H u$

$$\Rightarrow \lambda = \lambda^*$$
$$\Rightarrow \lambda \text{ is real.}$$

Now, by definition of positive definitive matrix,

$$\forall x \in \mathbb{R}^n, \; x^T A x > 0 \text{ --- } (3)$$

Let (λ, u) be an eigenvalue, eigenvector pair for A:

$$\Rightarrow Au = \lambda u$$
$$\Rightarrow u^T . Au = u^T \lambda u$$
$$\Rightarrow u^T . Au = \lambda u^T . u$$
$$\Rightarrow u^T . A . u = \lambda \|u\|_2^2 \quad \left[\because x^T . x = \|x\|_2^2\right]$$

Now, by (3), $u^T . Au > 0$ and $\|u\|_2^2$ is also positive

$$\Rightarrow \lambda \text{ is also positive.}$$

Therefore, all eigenvalues of a real symmetric positive definitive matrix are all real and strictly > 0.

Problem – 5

A5) Acc. to the question, exponential of the matrix,

$$e^{A} = \sum_{K=0}^{\infty} \frac{1}{K!} \cdot A^{K}, \text{ till convergence}$$

its given that this sum is unconditionally convergent.

We can write,

$$e^{A} = I + A + \frac{A^2}{2} + \cdots \frac{A^m}{m!} + \cdots \text{ till convergence} \quad (1)$$

We know that A is symmetric and positive definitive,

$\Rightarrow$ A is diagonalizable

$\Rightarrow$ A can be written as $X \Lambda X^{-1}$ where X is non-singular and Λ is a diagonal matrix where each diagonal element is an eigenvalue corresponding to each column of X (eigenvectors).

$\Rightarrow$ $A \sim X \Lambda X^{-1}$ — (2)

$\Rightarrow$ $A^m \sim X \Lambda^m X^{-1}$

Plug (2) in (1), $e^{A} = I + X\Lambda X^{-1} + \frac{X\Lambda^2 X^{-1}}{2!} + \cdots$

$$e^{A} = I + X\left[\Lambda + \frac{\Lambda^2}{2} + \cdots\right] X^{-1}$$

$$e^{A} = X e^{\Lambda} X^{-1}$$

Now, as v is an eigenvector of A,

$$e^{A} \cdot v = \left[I + A + \frac{A^2}{2} + \cdots\right] v$$

$$= \left[Iv + Av + \frac{A^2 v}{2} + \cdots\right]$$

$$= \left[v + \lambda v + \frac{\lambda^2 v}{2} + \cdots\right] \quad \left[\because A^m \cdot v = \lambda^m \cdot v\right]$$

$$= \left[I + \lambda + \frac{\lambda^2}{2} + \cdots\right] v$$

$$e^{A} \cdot v = e^{\lambda} \cdot v$$

Problem – 6

```
[Running] python -u "d:\College\sem 5\Scientific Computing\HomeWork\HomeWork 3\problem_6.py"
The matrix is:
 [[ 2  3  2]
 [10  3  4]
 [ 3  6  1]]
The starting vector is:
 [[0]
 [0]
 [1]]
-------------------------
The normalized power iteration is:
The largest eigenvalue is:
 11.000000000000002
The largest eigenvector is:
 [[0.37139068]
 [0.74278135]
 [0.55708601]]
-------------------------
The inverse iteration is:
The smallest eigenvalue is:
 0.5000000000000001
The smallest eigenvector is:
 [[ 0.18257419]
 [ 0.36514837]
 [-0.91287093]]
Using real eigensystem library routine:
(array([11., -2., -3.]), array([[ 3.71390676e-01,  1.82574186e-01, -4.13692033e-16],
       [ 7.42781353e-01,  3.65148372e-01, -5.54700196e-01],
       [ 5.57086015e-01, -9.12870929e-01,  8.32050294e-01]]))
```

As we can clearly see in the output that the result for largest magnitude eigenvalue-eigenvector pair is same using normalized power iteration and using a general real eigenvalue library routine.

In the inverse iteration, values may differ from the library routine output because the inverse iteration that we implemented gives the smallest magnitude eigenvalue but the library gives smallest eigenvalue w.r.t sign i.e. – [Negative]

Problem –7

```
[Running] python -u "d:\College\sem 5\Scientific Computing\HomeWork\HomeWork 3\problem_7.py"
The matrix is:
 [[6 2 1]
 [2 3 1]
 [1 1 1]]
The starting vector is:
 [[0]
 [0]
 [1]]
-------------------------
The shifted inverse iteration is:
The smallest eigenvalue is:
 0.43706443871540673
The smallest eigenvector is:
 [[0.86643225]
 [0.45305757]
 [0.20984279]]
-------------------------
Using real eigensystem library routine:
(array([0.57893339, 2.13307448, 7.28799214]), array([[-0.0431682 , -0.49742503, -0.86643225],
       [-0.35073145,  0.8195891 , -0.45305757],
       [ 0.9354806 ,  0.28432735, -0.20984279]]))
```

As we can clearly see in the output that the result for smallest magnitude eigenvalue-eigenvector pair using shifted inverse iteration with sigma = 5 and is close to the result generated using a general real eigenvalue library routine.

If we increase/decrease the value of sigma, the closeness of eigenvalue increases and eigenvector decreases and vice-versa.

The value may also be off due to the library routine handling error/edge cases and also handling rounding off errors which will occur in our implementation of shifted inverse.

Problem –8

```
[Running] python -u "d:\College\sem 5\Scientific Computing\HomeWork\HomeWork 3\problem_8.py"
The matrix is:
 [[ 2  3  2]
 [10  3  4]
 [ 3  6  1]]
The starting vector is:
 [[0.22021988]
 [0.68894051]
 [0.14874822]]
-------------------------
The Rayleigh quotient iteration is:
The rate of convergence is:
 [[0.8309955]]
The rate of convergence is:
 [[0.65228048]]
The rate of convergence is:
```

```
The rate of convergence is:
 [[0.6437782]]
The rate of convergence is:
 [[0.6437782]]
The rate of convergence is:
 [[0.6437782]]
The rate of convergence is:
 [[0.6437782]]
The closest eigenvalue to the rayleigh quotient is:
 4878899596318037.0
The closest eigenvector to the rayleigh quotient is:
 [[-0.37139068]
 [-0.74278135]
 [-0.55708601]]
-------------------------
Using real eigensystem library routine:
(array([-8.05778972, -1.34550956, 15.40329928]), array([[-0.63829307, -0.47912245,  0.60251443],
       [ 0.72204791, -0.1012749 ,  0.6843904 ],
       [-0.26688721,  0.87188593,  0.41059243]]))

[Done] exited with code=0 in 0.277 seconds
```

As we can see in the output that the result of our own coded Rayleigh Quotient Iteration of the eigenvalue-eigenvector pair is not equal to the one calculated by the real eigensystem library.

This is because that the library routine handled error/edge cases and also handles rounding off errors which will occur in our implementation of the same.

The rate of convergence is also calculated at every step.

Problem –9

```
[Running] python -u "d:\College\sem 5\Scientific Computing\HomeWork\HomeWork 3\problem_9.py"
Problem 6 matrix
The matrix is:
 [[ 2  3  2]
 [10  3  4]
 [ 3  6  1]]
The starting vector is:
 [[0]
 [0]
 [1]]
-------------------------
The modified QR iteration is:
The eigenvalues are:
 [11. -3. -2.]
The eigenvectors are:
 [[-1.00000000e+000 -1.02348309e-108  0.00000000e+000]
 [-1.02348309e-108  1.00000000e+000  0.00000000e+000]
 [-0.00000000e+000  0.00000000e+000  1.00000000e+000]]
-------------------------
Using real eigensystem library routine:
(array([11., -2., -3.]), array([[ 3.71390676e-01,  1.82574186e-01, -4.13692033e-16],
       [ 7.42781353e-01,  3.65148372e-01, -5.54700196e-01],
       [ 5.57086015e-01, -9.12870929e-01,  8.32050294e-01]]))
-------------------------
Problem 7 matrix
The matrix is:
 [[6 2 1]
 [2 3 1]
 [1 1 1]]
The starting vector is:
 [[0]
 [0]
 [1]]
-------------------------
The modified QR iteration is:
The eigenvalues are:
 [7.28799214 2.13307448 0.57893339]
The eigenvectors are:
 [[-1.00000000e+00  4.45619872e-64  0.00000000e+00]
 [ 4.45619872e-64  1.00000000e+00  0.00000000e+00]
 [-0.00000000e+00  0.00000000e+00  1.00000000e+00]]
-------------------------
Using real eigensystem library routine:
(array([0.57893339, 2.13307448, 7.28799214]), array([[-0.0431682 , -0.49742503, -0.86643225],
       [-0.35073145,  0.8195891 , -0.45305757],
       [ 0.9354806 ,  0.28432735, -0.20984279]]))

[Done] exited with code=0 in 0.682 seconds
```

As we can clearly see in the output, the eigenvalue-eigenvector pair generated by modified QR iteration is exactly the same as the one given by the real eigensystem library routine.

This proves that modified QR iteration converges the fastest amongst all the methods and gives the most accurate result. The order of values may be different, but they are same.