Homework 3 Spectral Graph Theory

Exercise 1

a.

Script

```
library(igraph)
g1 = graph.full(5, directed = FALSE, loops = FALSE)
K5 = g1
K5_al = get.adjacency(K5)
```

Output:

```
[1,] . 1 1 1 1
[2,] 1 . 1 1 1
[3,] 1 1 . 1 1
[4,] 1 1 1 . 1
[5,] 1 1 1 1 .
```

b.

Script:

```
g2 = graph.full(3, directed = FALSE, loops = FALSE)
K5.3 = g1 + g2
K5.3_al = get.adjacency(K5.3)
```

Output:

```
[1,] . 1 1 1 1 . . .
[2,] 1 . 1 1 1 . . .
[3,] 1 1 . 1 1 . . .
[4,] 1 1 1 . 1 . . .
[5,] 1 1 1 1 . . . .
[6,] . . . . . 1 1
[7,] . . . . . 1 . 1
[8,] . . . . . 1 1 .
```

c.

Script:

```
K5.3e = K5.3 + edges(1,7)
K5.3e_al = get.adjacency(K5.3e)
```

Output:

```
[1,] . 1 1 1 1 . 1 .
[2,] 1 . 1 1 1 . . .
[3,] 1 1 . 1 1 . . .
[4,] 1 1 1 . 1 . . .
[5,] 1 1 1 1 . . . .
[6,] . . . . . 1 1
[7,] 1 . . . . 1 . 1
[8,] . . . . . 1 1 .
```

d.

Script:

```
B2.3 = graph.full.bipartite(2,3,directed = FALSE)
B2.3_al = get.adjacency(B2.3)
```

Output:

```
[1,] . . 1 1 1
[2,] . . 1 1 1
[3,] 1 1 . . .
[4,] 1 1 . . .
[5,] 1 1 . . .
```

e.

Script:

```
S5 = graph.star(n=5,mode="undirected")
S5_al = get.adjacency(S5)
```

Output:

```
[1,] . 1 1 1 1
[2,] 1 . . . .
[3,] 1 . . . .
[4,] 1 . . . .
[5,] 1 . . . .
```

f.

Script:

```
P5 = graph(c(1,2,2,3,3,4,4,5,5,6),directed=FALSE)
P5_al = get.adjacency(P5)
```

Output:

```
[1,] . 1 . . . .
[2,] 1 . 1 . . .
[3,] . 1 . 1 . .
[4,] . . 1 . 1 .
[5,] . . . 1 . 1
[6,] . . . . 1 .
```

Exercise 2

a.

Script:

```
K5_diag = diag(degree(K5))
```

Output:

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    4    0    0    0    0
[2,]    0    4    0    0    0
[3,]    0    0    4    0    0
[4,]    0    0    0    4    0
[5,]    0    0    0    0    4
```

Script:

```
K5.3_diag = diag(degree(K5.3))
```

Output:

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    4    0    0    0    0    0    0    0
[2,]    0    4    0    0    0    0    0    0
[3,]    0    0    4    0    0    0    0    0
[4,]    0    0    0    4    0    0    0    0
[5,]    0    0    0    0    4    0    0    0
[6,]    0    0    0    0    0    2    0    0
[7,]    0    0    0    0    0    0    2    0
[8,]    0    0    0    0    0    0    0    2
```

Script:
K5.3e_diag = diag(degree(K5.3e))

Output:

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]
[1,]	5	0	0	0	0	0	0	0
[2,]	0	4	0	0	0	0	0	0
[3,]	0	0	4	0	0	0	0	0
[4,]	0	0	0	4	0	0	0	0
[5,]	0	0	0	0	4	0	0	0
[6,]	0	0	0	0	0	2	0	0
[7,]	0	0	0	0	0	0	3	0
[8,]	0	0	0	0	0	0	0	2

Script:
B2.3_diag = diag(degree(B2.3))

Output:

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	3	0	0	0	0
[2,]	0	3	0	0	0
[3,]	0	0	2	0	0
[4,]	0	0	0	2	0
[5,]	0	0	0	0	2

Script:
S5_diag = diag(degree(S5))

Output:

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	4	0	0	0	0
[2,]	0	1	0	0	0
[3,]	0	0	1	0	0
[4,]	0	0	0	1	0
[5,]	0	0	0	0	1

Script:
P5_diag = diag(degree(P5))

Output:

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
[1,]	1	0	0	0	0	0
[2,]	0	2	0	0	0	0
[3,]	0	0	2	0	0	0
[4,]	0	0	0	2	0	0
[5,]	0	0	0	0	2	0
[6,]	0	0	0	0	0	1

Exercise 3

Script:

```
K5_laplacian = graph.laplacian(K5)
[1,]  4 -1 -1 -1 -1
[2,] -1  4 -1 -1 -1
[3,] -1 -1  4 -1 -1
[4,] -1 -1 -1  4 -1
[5,] -1 -1 -1 -1  4
K5.3_laplacian = graph.laplacian(K5.3)
[1,]  4 -1 -1 -1 -1  .  .  .
[2,] -1  4 -1 -1 -1  .  .  .
[3,] -1 -1  4 -1 -1  .  .  .
[4,] -1 -1 -1  4 -1  .  .  .
[5,] -1 -1 -1 -1  4  .  .  .
[6,]  .  .  .  .  .  2 -1 -1
[7,]  .  .  .  .  . -1  2 -1
[8,]  .  .  .  .  . -1 -1  2
K5.3e_laplacian = graph.laplacian(K5.3e)
[1,]  5 -1 -1 -1 -1  . -1  .
[2,] -1  4 -1 -1 -1  .  .  .
[3,] -1 -1  4 -1 -1  .  .  .
[4,] -1 -1 -1  4 -1  .  .  .
[5,] -1 -1 -1 -1  4  .  .  .
[6,]  .  .  .  .  .  2 -1 -1
[7,] -1  .  .  .  . -1  3 -1
[8,]  .  .  .  .  . -1 -1  2
B2.3_laplacian = graph.laplacian(B2.3)
[1,]  3  . -1 -1 -1
[2,]  .  3 -1 -1 -1
[3,] -1 -1  2  .  .
[4,] -1 -1  .  2  .
[5,] -1 -1  .  .  2
```

S5_laplacian = graph.laplacian(S5)

[1,] 4 -1 -1 -1 -1

[2,] -1 1 . . .

[3,] -1 . 1 . .

[4,] -1 . . 1 .

[5,] -1 . . . 1

P5_laplacian = graph.laplacian(P5)

[1,] 1 -1 . . . .

[2,] -1 2 -1 . . .

[3,] . -1 2 -1 . .

[4,] . . -1 2 -1 .

[5,] . . . -1 2 -1

[6,] . . . . -1 1

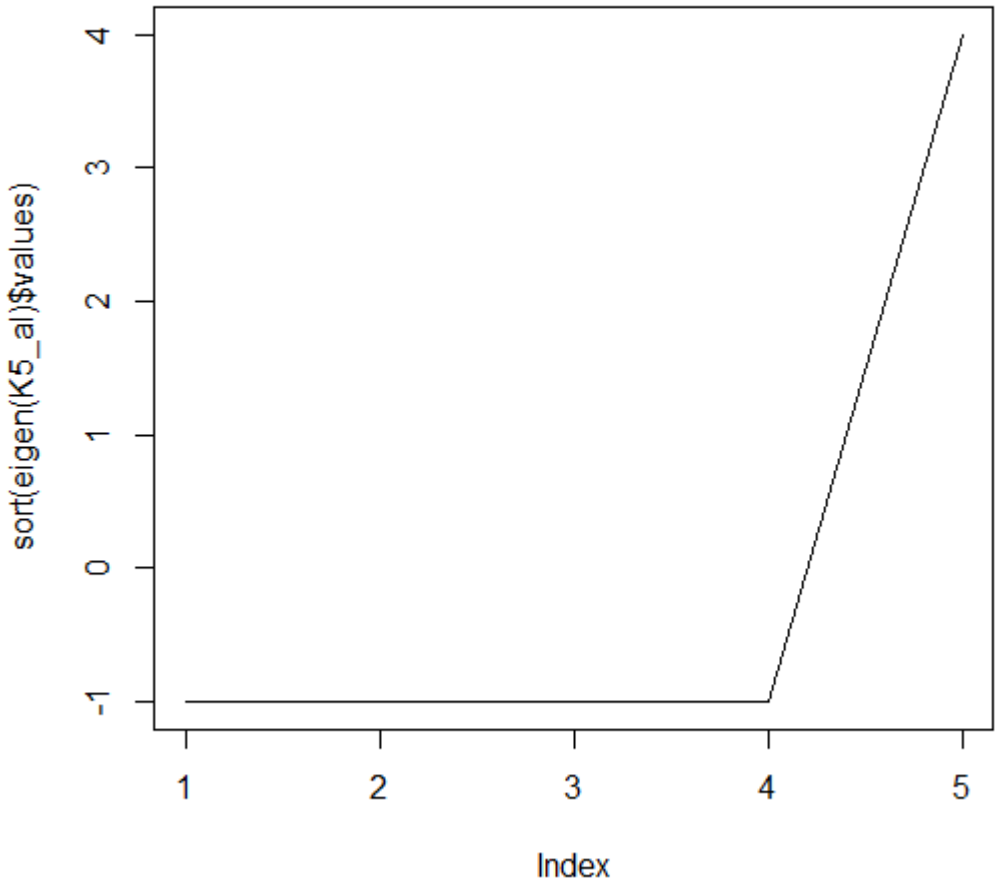
a. L is not a sparse matrix.
b. L is non-zero in the diagonal and at L[a][b] if there is an edge between a and b.
c. The value of non-diagonal and non-zero elements is -1.
d. L contains degree of vertex along it's diagonal.

Exercise 4

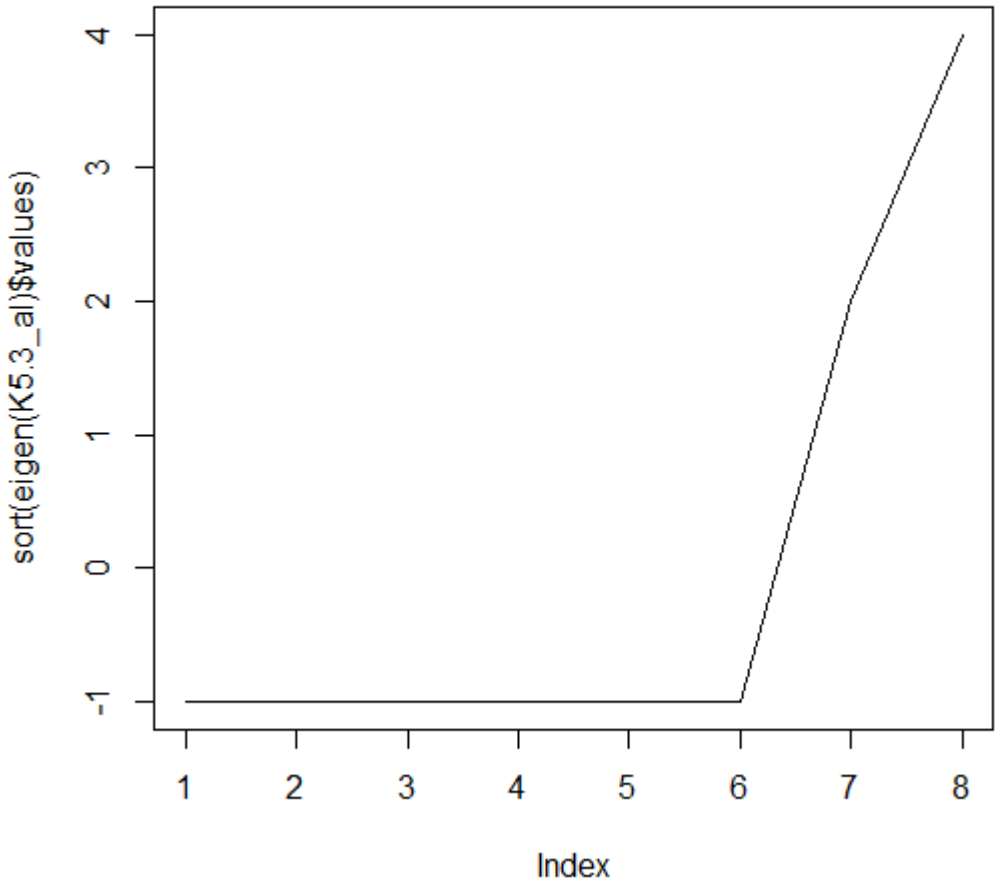
eigen(K5_al)$values

[1] 4 -1 -1 -1 -1

plot(sort(eigen(K5_al)$values),type="line")



```
eigen(K5.3_al)$values
[1]  4  2 -1 -1 -1 -1 -1 -1
plot(sort(eigen(K5.3_al)$values),type="line")
```

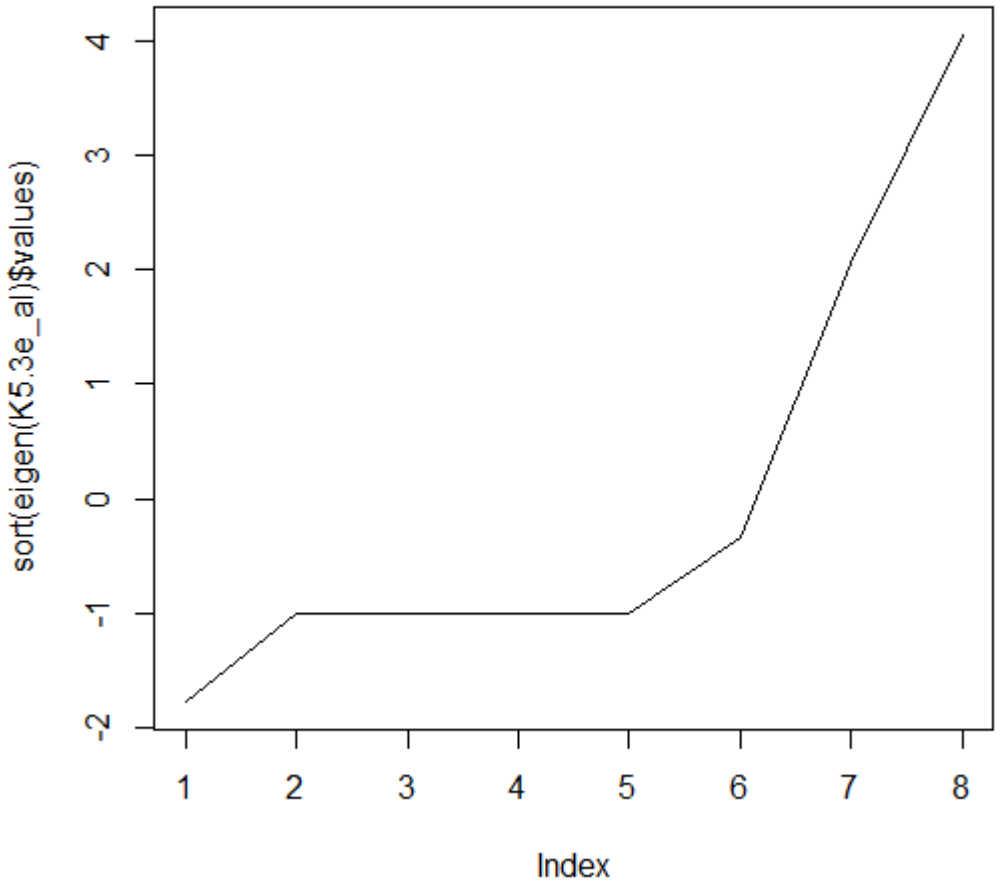


eigen(K5.3e_al) $values

[1] 4.0615344 2.0549225 -0.3365858 -1.0000000 -1.0000000 -1.0000000 -1.0000000

[8] -1.7798711

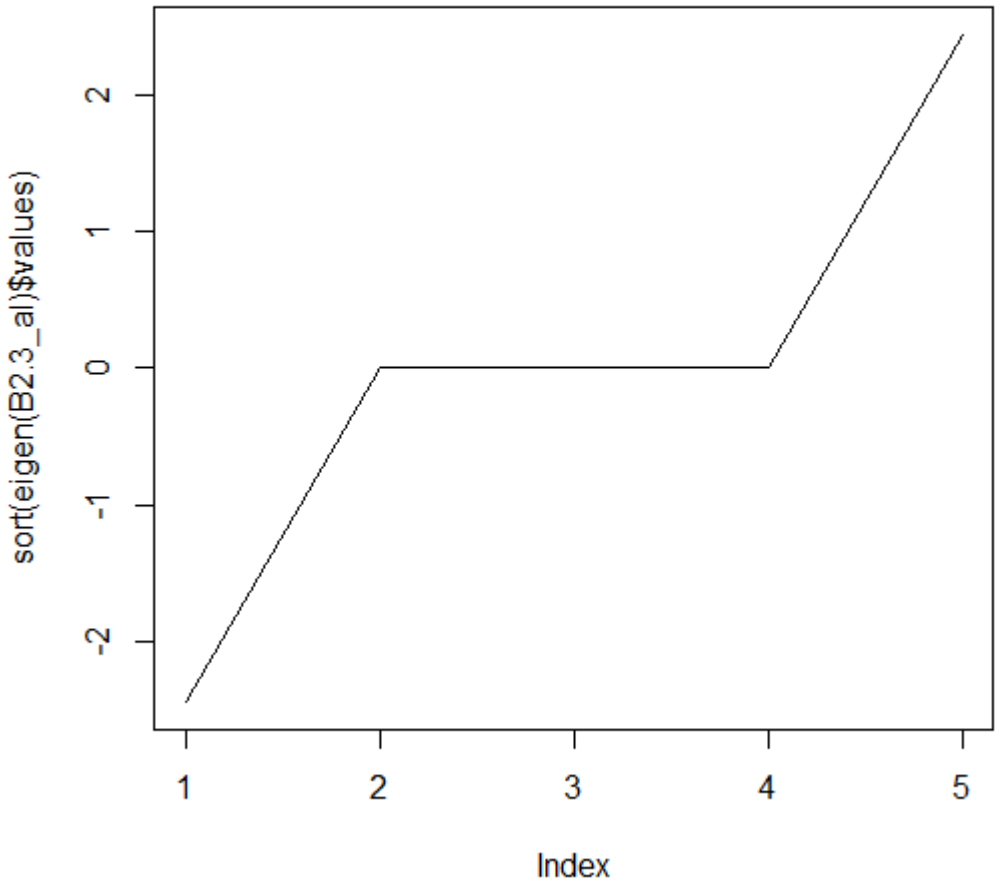
plot(sort(eigen(K5.3e_al)$values),type="line")



eigen(B2.3_al) $values

[1] 2.449490e+00 2.664535e-15 0.000000e+00 -9.860761e-32 -2.449490e+00

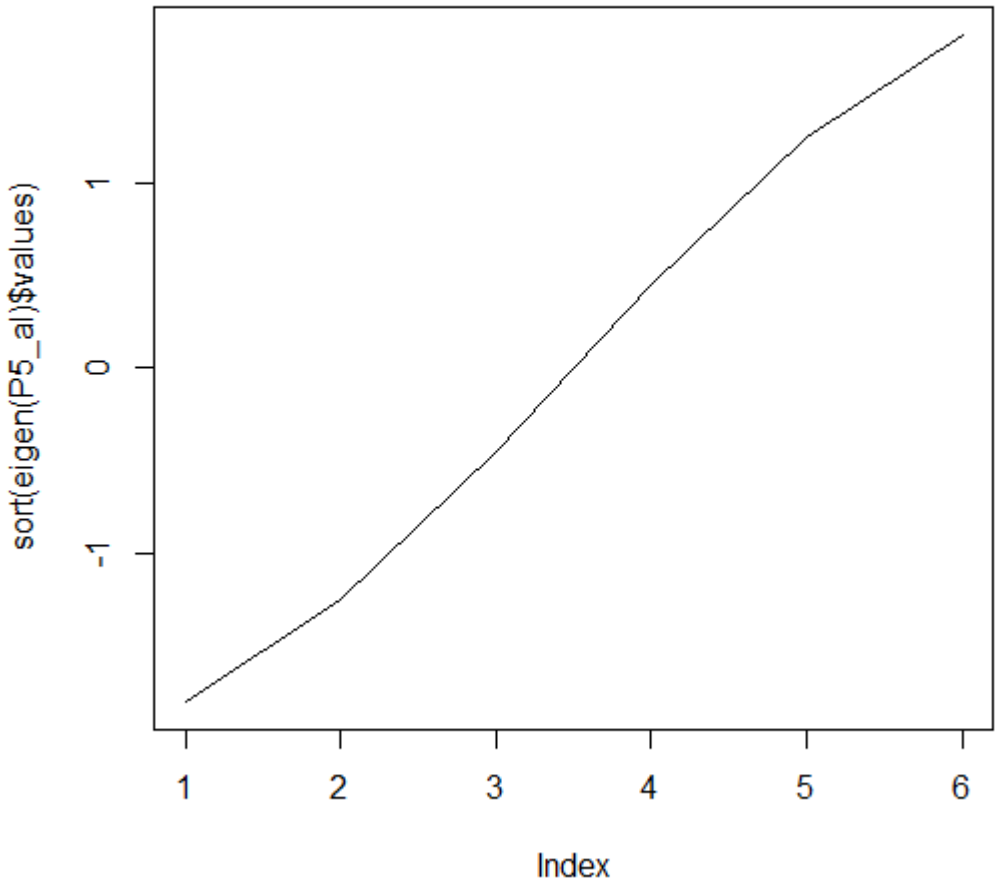
plot(sort(eigen(B2.3_al)$values),type="line")



eigen(P5_al) $values

[1] 1.8019377 1.2469796 0.4450419 -0.4450419 -1.2469796 -1.8019377

plot(sort(eigen(P5_al)$values),type="line")



a. Number of unique values: 2
Largest: 4, number of vertices -1
Smallest: -1
Multiplicity: 4 :- 1, -1 :- 4

b.
Graph Spectrum:
[1] 2.449490e+00 2.664535e-15 0.000000e+00 -9.860761e-32 -2.449490e+00
Highest eigen value in a general m n bipartite graph is sqrt(m*n) and lowest is -sqrt(m*n)

If \lambda is the highest eigen value of the bipartite graph, then -\lambda will be the lowest eigen value of the graph. This is not true for other eigen values other then highest and lowest values.

c. Largest eigen value of star graph S5: 2
   For a generalized star graph, the largest eigen value is sqrt(N-1) where N is the number of vertices in the graph. It is equivalent to sqrt(degree of center vertex).

d. Largest eigenvalue of path graph p5 is 1.8019
   As the path length increases, the largest eigen value also increases and lowest decreases.

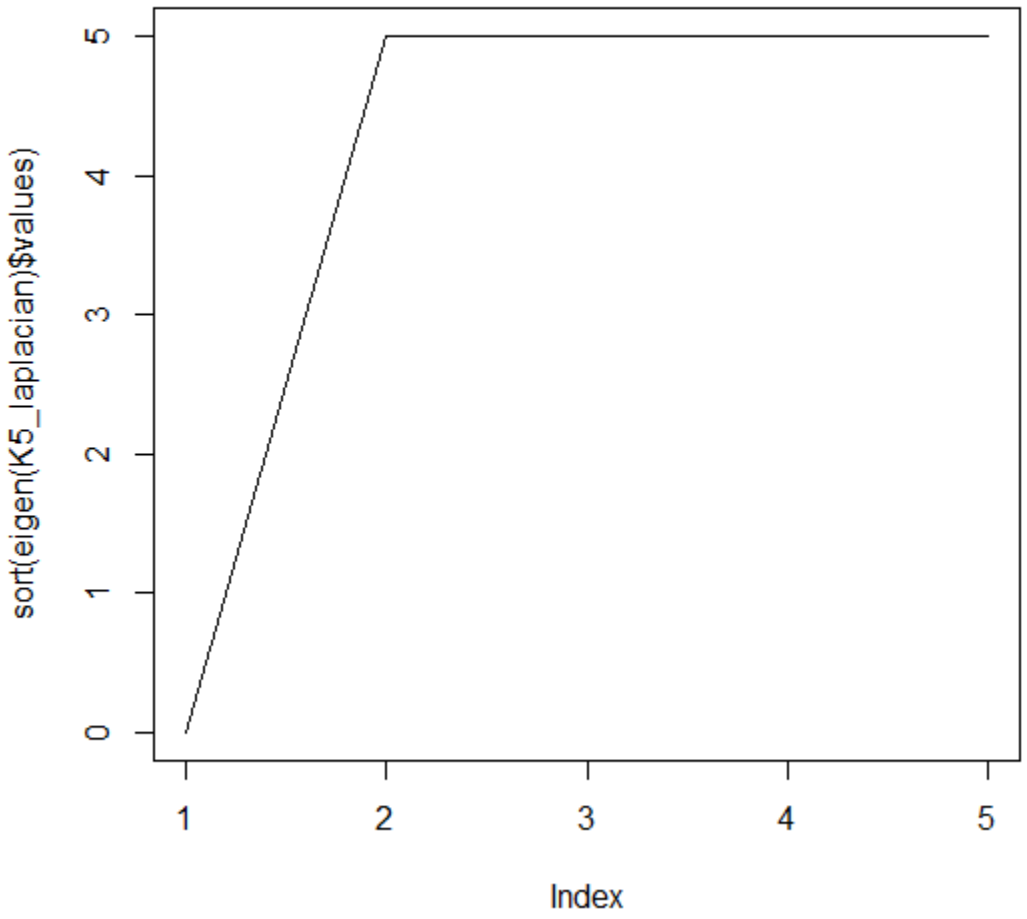
e. Largest eigen value of clique > largest eigen value of star > largest eigen value of path

Exercise 5

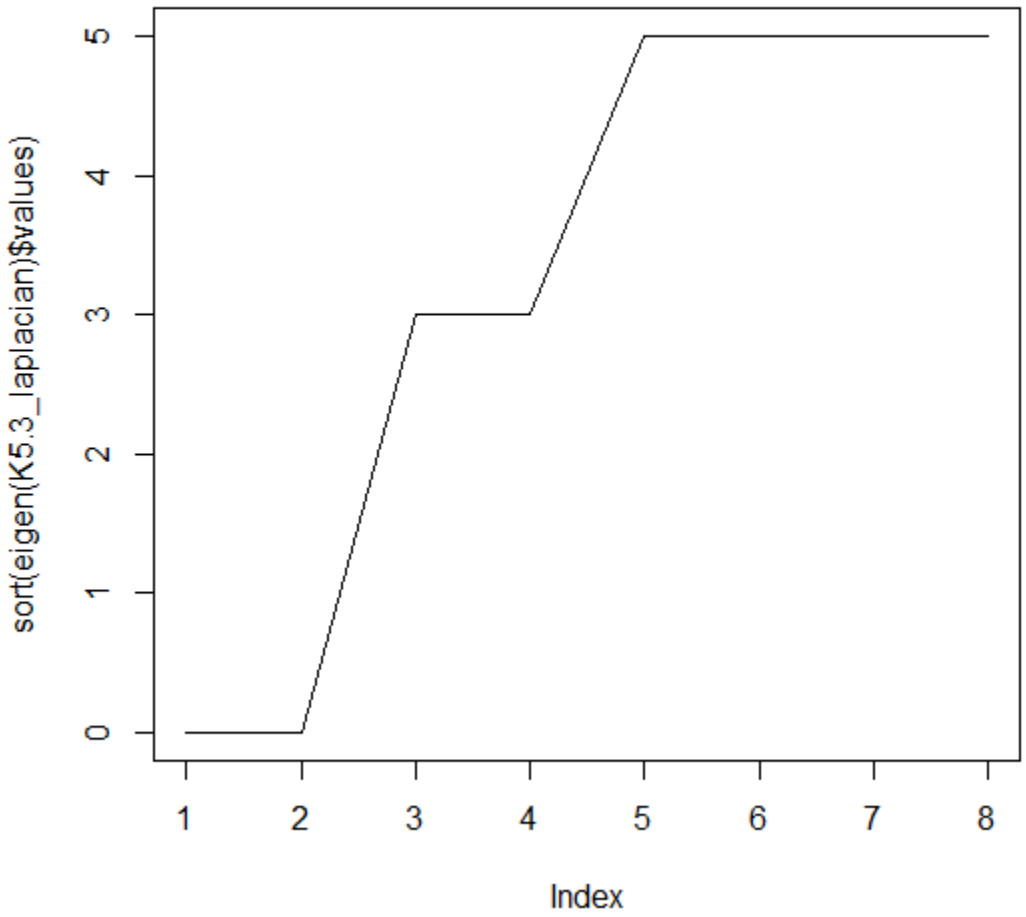
```
K5_laplacian = graph.laplacian(K5)
K5.3_laplacian = graph.laplacian(K5.3)
K5.3e_laplacian = graph.laplacian(K5.3e)
B2.3_laplacian = graph.laplacian(B2.3)
S5_laplacian = graph.laplacian(S5)
P5_laplacian = graph.laplacian(P5)

eigen(K5_laplacian)$values
[1] 6.000000e+00 6.000000e+00 6.000000e+00 6.000000e+00 6.000000e+00
[6] 7.105427e-15
eigen(K5.3_laplacian)$values
[1] 6.000000e+00 6.000000e+00 6.000000e+00 6.000000e+00 6.000000e+00
[6] 3.000000e+00 3.000000e+00 7.105427e-15 2.220446e-15
eigen(K5.3e_laplacian)$values
[1] 7.087406e+00 6.000000e+00 6.000000e+00 6.000000e+00 6.000000e+00
[6] 3.555434e+00 3.000000e+00 3.571598e-01 1.509903e-14
eigen(B2.3_laplacian)$values
[1] 5.000000e+00 3.000000e+00 2.000000e+00 2.000000e+00 2.664535e-15
eigen(P5_laplacian)$values
[1] 3.732051e+00 3.000000e+00 2.000000e+00 1.000000e+00 2.679492e-01
[6] 1.332268e-15
```

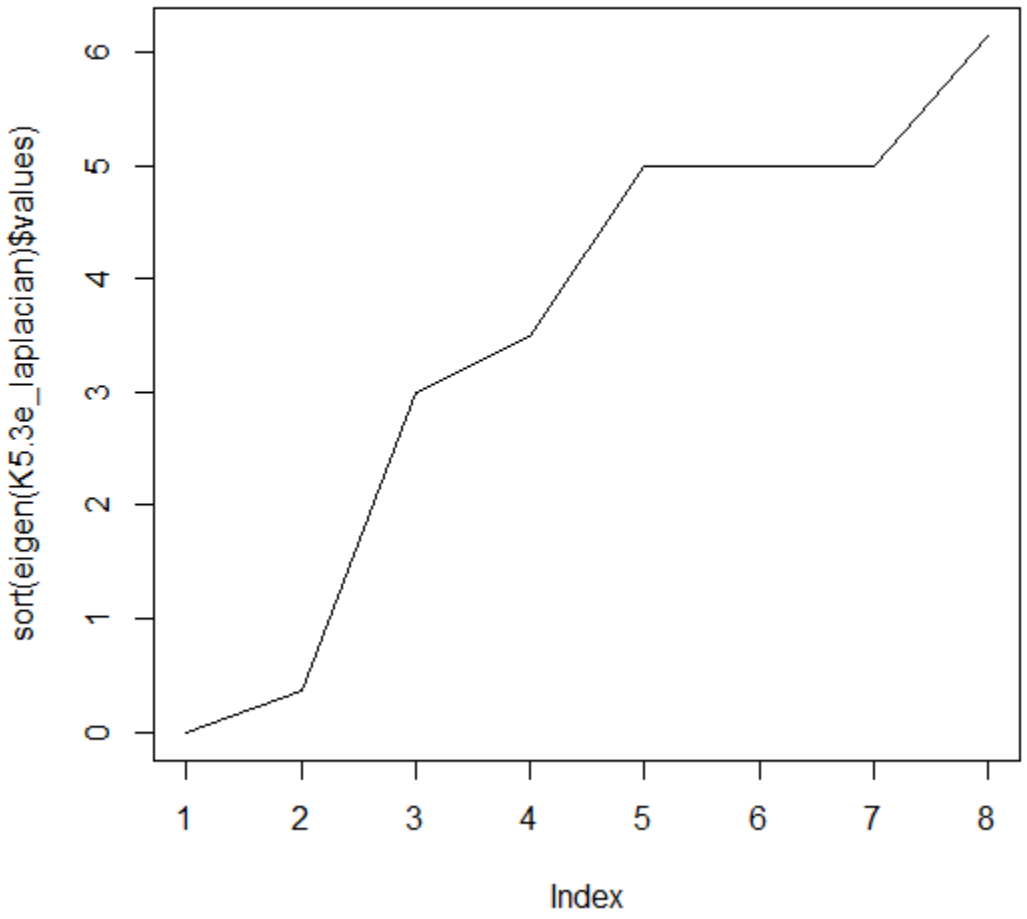
K5 - Clique



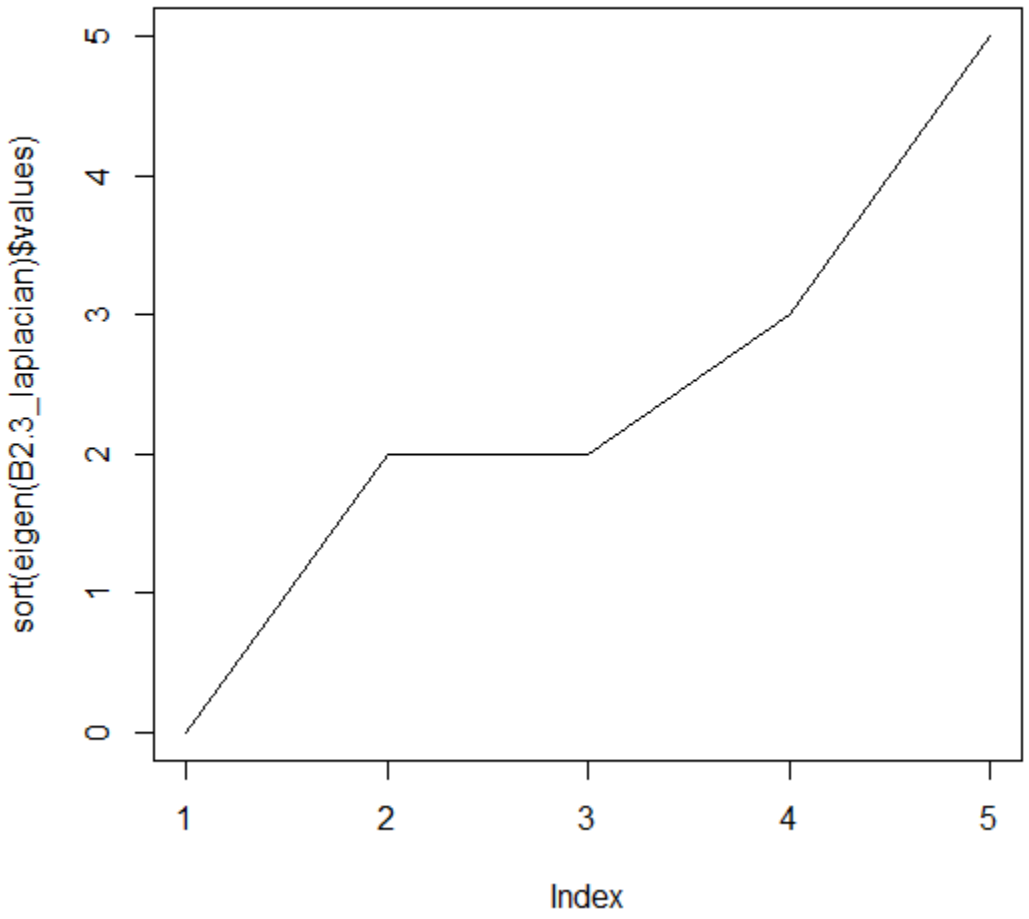
K5.3 graph



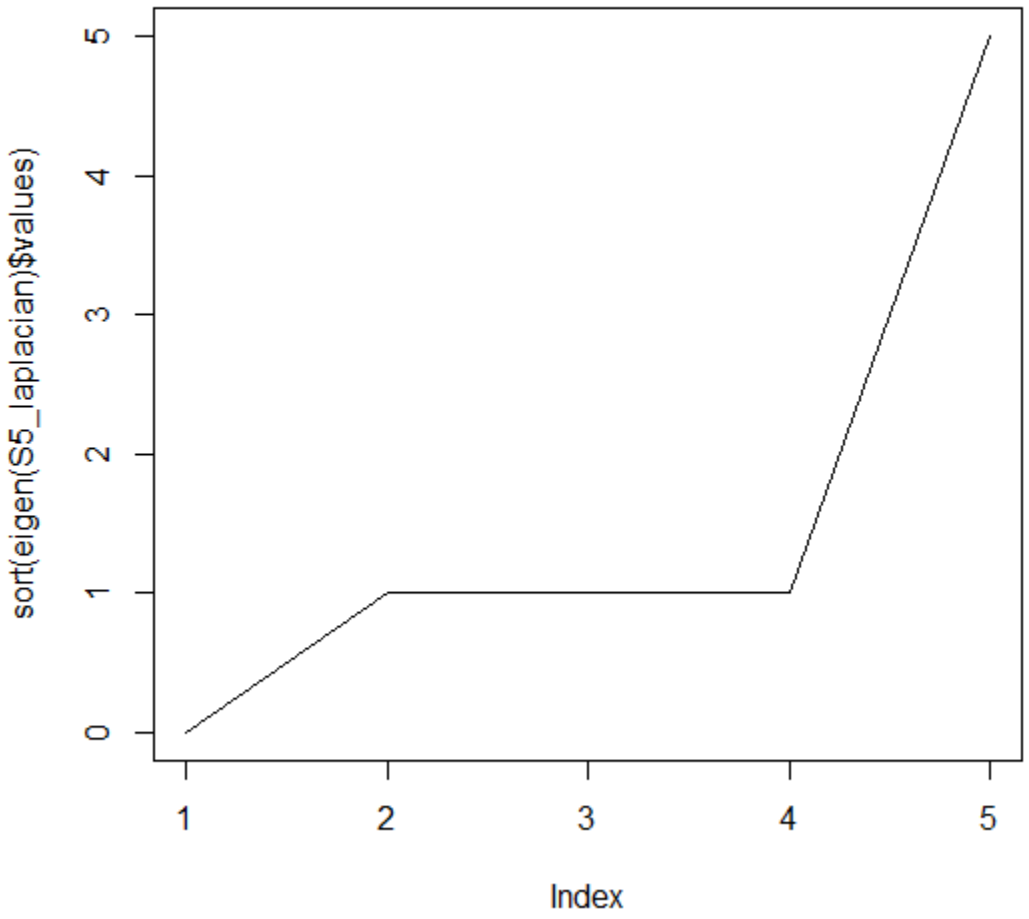
K5.3e



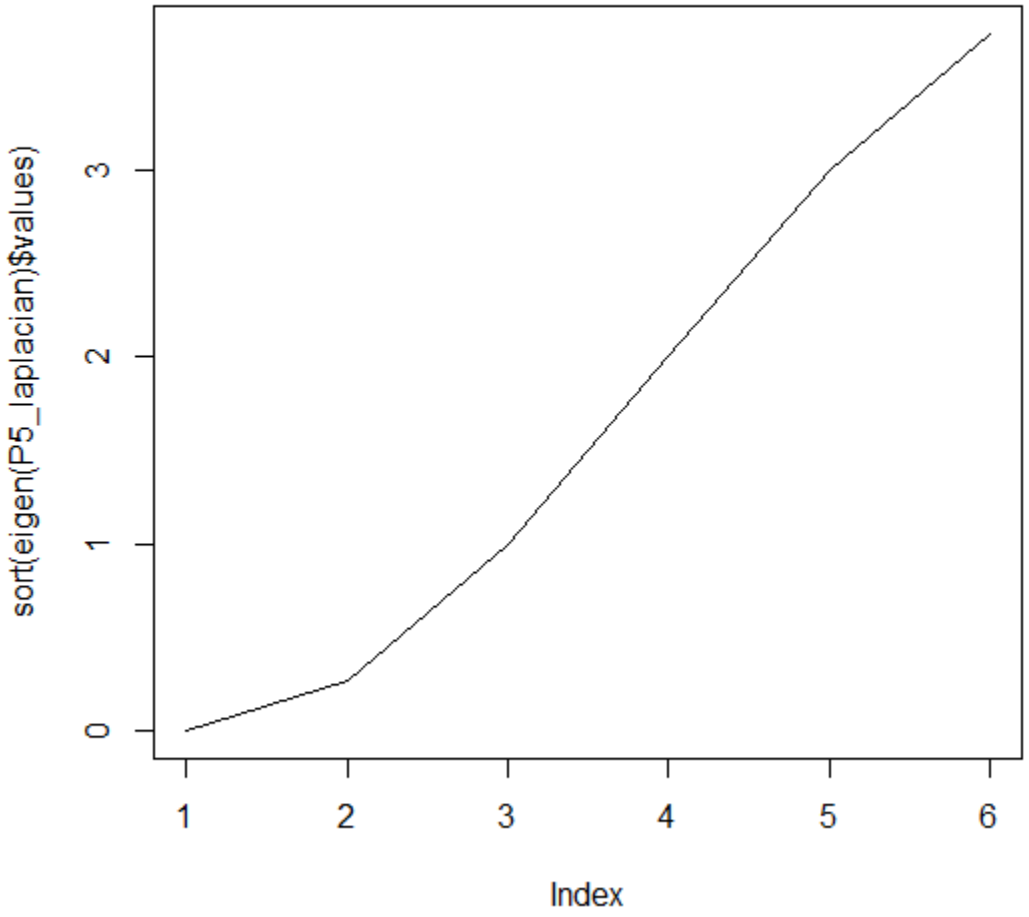
B2.3



S5



P5



a.

Graph	Maximum	Minimum
Clique	5, No of vertices	0
Disconnected Cliques	5, clique with higher no of vertices	0
Connected cliques	6.14,	0
Bi-partite graph	5, n+m	0
Star Graph	5, No of vertices	0
Path	3.73	0

b.

Graph	Multiplicity of 0 eigenvalues
Clique	1

Disconnected Cliques	2, no of cliques c
Connected cliques	1
Bi-partite graph	1
Star Graph	1
Path	1

c. The multiplicity of zero eigen values is equivalent to the number of cliques c.
d. The multiplicity of zero eigen values in a connected graph is 1 as the connected component pf graph is 1. The multiplicity of zero eigen values is equal to the number of connected components of the graph.
e. Second smallest value is 2.
f. The eigen value corresponding to eigen vector with all 1's is 0. It is for a clique.
g. The multiplicity of zero eigen values is equal to the number of connected components of a graph. So if multiplicity is k then there are k connected components in the graph.