# HW5 ECS 230

Heqiao Ruan
email:hruan@ucdavis.edu

December 5, 2018

## 1 Problem 1:

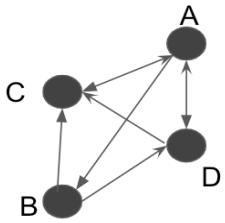
Here we know that $\lambda$ is the eigenvalue of a real square matrix A, so it satisfy $Ax = \lambda x$ for a vector x. Then due to the property of characteristic polynomial we know that $g(\lambda) = det(A - \lambda I)$ and the characteristic polynomial of $A^T$ is $g^{'}(\lambda) = det(A^T - \lambda I)$ where I is the identity matrix. Here as we know $det(A) = det(A^T)$, the $det(A - \lambda I) = det((A - \lambda I)^T) = det(A^T - \lambda I^T) = det(A^T - \lambda I) = g(\lambda)$, so we can see that $A$ and $A^T$ has the same characteristic polynomial. So we get $g(\lambda) = g^{'}(\lambda)$ which means $\lambda$ is also the eigenvalue of $A^T$.

## 2 Problem 2

Here after transforming the matrix into the link matrix, we print the matrix mapping in the c program and its screenshot is shown as below.
The result of the leading eigenvector and the result for each iteration as well as the link matrix is shown as below:

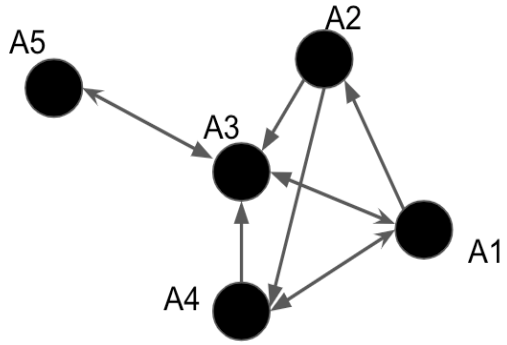
```
Iteration 0: the difference is 1.849750
Iteration 1: the difference is 0.804816
Iteration 2: the difference is 0.424316
Iteration 3: the difference is 0.382134
Iteration 4: the difference is 0.235707
Iteration 5: the difference is 0.076479
Iteration 6: the difference is 0.045061
Iteration 7: the difference is 0.038174
Iteration 8: the difference is 0.018949
Iteration 9: the difference is 0.004895
Iteration 10: the difference is 0.004434
Iteration 11: the difference is 0.003591
Iteration 12: the difference is 0.001377
Iteration 13: the difference is 0.000514
Iteration 14: the difference is 0.000470
Iteration 15: the difference is 0.000315
Iteration 16: the difference is 0.000104
Iteration 17: the difference is 0.000057
Iteration 18: the difference is 0.000049
Iteration 19: the difference is 0.000026
Iteration 20: the difference is 0.000007
Iteration 21: the difference is 0.000006
Iteration 22: the difference is 0.000005
Iteration 23: the difference is 0.000002
Iteration 24: the difference is 0.000001
Converge after 25 iterations, the dominant eigenvector:
0.387097
0.129032
0.290323
0.193548
PRINT MATRIX:A:
0.000000 0.000000 1.000000 0.500000
0.333333 0.000000 0.000000 0.000000
0.333333 0.500000 0.000000 0.500000
0.333333 0.500000 0.000000 0.000000
```

We can see that the ordinary power method it converge after 25 iterations and the result eigenvector is $[0.387097, 0.129032, 0.290323, 0.193548]^T$
Then we can visualize the toy network:




# 3 Problem 3

Here we use another network which can be visualized as below:




Then the running result is shown as below(result eigenvector and the link matrix)

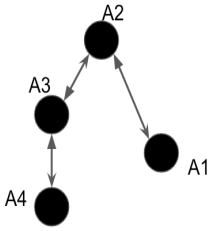
```
Converge after 47 iterations, the dominant eigenvector:
0.244898
0.081633
0.367347
0.122449
0.183674
```

```
PRINT MATRIX:A:
0.000000 0.000000 0.500000 0.500000 0.000000
0.333333 0.000000 0.000000 0.000000 0.000000
0.333333 0.500000 0.000000 0.500000 1.000000
0.333333 0.500000 0.000000 0.000000 0.000000
0.000000 0.000000 0.500000 0.000000 0.000000
```

Then the eigenvector here has the ranking of A3,A1,A5,A4,A2(in terms of weightt). So we can see that the boost page 3's score above that of page 1. So we verify the exercise 1.

# 4 Problem 4

Here let's construct a linear chain and the network is visualized as below:




Then we can see that initialize from $[1, 1, 1, 1]$, the result is that the eigenvector corresponding

to the leading eigenvalue is $[0.166667, 0.333333, 0.333333, 0.166667]^T$ and it converges in 21 iterations.We can also see it visualized below:

```
Converge after 21 iterations, the dominant eigenvector:
0.166667
0.333333
0.333333
0.166667
```

This eigenvector make sense indeed as it has the ranking of page of $(A_1, A_3) -> (A_1, A_4)$.However we can see that this matrix indeed has 2 same leading eigenvalues 1 and -1 so potentially for different initialization may have some questions.

However, if we initialize it as $[1, 2, 3, 4]^T$, it's not converge even after 150 iterations.We can see the result visualized below. What's more the link matrix now is also shown.

```
Iteration 148: the difference is 0.744208
Iteration 149: the difference is 0.744208
Iteration 150: the difference is 0.744208

 Still not converge!
Can't converge After 151 iterations to vectors:
The vector now is:
0.372104
0.496139
0.744208
0.248069
```

```
PRINT MATRIX:A:
0.000000 0.500000 0.000000 0.000000
1.000000 0.000000 0.500000 0.000000
0.000000 0.500000 0.000000 1.000000
0.000000 0.000000 0.500000 0.000000
```

For the second initialization we can see that it may trapped in a local optimal, as there are 2 dominating eigenvectors corresponding to the leading eigenvalue. So it may explain why the initialization $[1, 2, 3, 4]^T$ doesn't converge well.

# 5 Problem 5

We can solve the above problem by the shifted power method. To implement the shifted power method, we can translate the A: $B = A + \Lambda I$ where I is the identity matrix. Then the eigenvalue problem becomes $Bv = (A + \Lambda I)v = Av + \Lambda v = (\lambda + \Lambda)v$ where $\Lambda$ is the amount of translation and $\lambda$ is the eigenvalue.

Here I set the shift amount $\Lambda$ as 0.3 so that making the leading eigenvalue 1.3 and make the leading one unique(the absolute value of the second eigenvalue is 0.7).

Then we try this trick and get the following result:

```
Iteration 27: the difference is 0.000001
Converge after 28 iterations, the dominant eigenvector:
0.166666
0.333333
0.333334
0.166667
```

We can see that it converges to well after 27 iterations.

# 6 Problem 6

Here using the shifted power method, we can see that the convergence rate becomes much more faster(27 compared larger than 150). What's more the eigenvectors we solve are invariant under shift.
However, there maybe some issues caused by shifted power method when there are disconnected webs. It may compromise the network's structure(after inference there may be links after shift where there's supposed be no link).
Then we have anothermodified method using the matrix M calculated by $M = (1-m)A+m*S$ with the default ratio m=0.15 as specified in the paper where S is the matrix with all entries $\frac{1}{n}$. Here we can see that this method can ensure that the matrix is non-singular so that we have a web with no isolated block. However, if n is too large, this effect can be compromised which only introduces more time complexity.**The result is shown as below:**

```
Iteration 74: the difference is 0.000004
Iteration 75: the difference is 0.000004
Iteration 76: the difference is 0.000003
Iteration 77: the difference is 0.000003
Iteration 78: the difference is 0.000002
Iteration 79: the difference is 0.000002
Iteration 80: the difference is 0.000002
Iteration 81: the difference is 0.000001
Iteration 82: the difference is 0.000001
Iteration 83: the difference is 0.000001
Converge after 84 iterations, the dominant eigenvector:
0.175439
0.324561
0.324561
0.175439
```

We can see that the modified method converge much slower than the shifted method.So we can see that we should have selected a good amount of shift in applying the shift method. So we may argue that given smart choice of shift amount for fully connected web, shift method is preferred. However for the matrix for web with disconnected blocks which is pretty common in real life application, we may need to use the modified version to make sure the matrix non-singular.