Exercise 3

Eigenvalue Decomposition (2 Points)

```
>> A = load_sparse_normal(filename);

>> [VEC, VAL] = eig(A);

>> VAL = diag(VAL);

>> min_ = min(VAL)

min_ = -2.6356
>> max_ = max(VAL)

max_ =  3.6742
>> VAL

VAL =

  -2.6356e+000
  -2.5296e+000
  -1.5243e+000
  -1.4142e+000
  -1.2797e+000
  -8.3928e-001
  -1.8364e-015
  -1.5105e-016
   7.2398e-016
   7.8539e-016
   2.2042e-001
   1.1944e+000
   1.4142e+000
   1.6391e+000
   2.0804e+000
   3.6742e+000
```

The lowest eigen value is **-2.6356**, and the largest one is **3.6742**. No eigen value of this vector is zero.

Centrality in Undirected Networks (4 Points)

Compute the eigenvector centrality of all users in the given Facebook graph using two methods:

By using the GNU Octave function eigs()

By using power iteration

```
function [val_final, vec_final] = eig_power(A)
  x = compute_degrees(A)';
  diff = 1e-10;
  vec_final = x;
  val_final = 0;
  while(1)
    vec_old = vec_final;
    val_old = val_final;
    vec_final = A*vec_final;
    val_final = norm(vec_final);
    vec_final = vec_final/val_final;
    if abs(val_final - val_old)<diff && norm(vec_final-vec_old)<diff
      break;
    end
  end
end
```

Compare the results of both calculations by computing the norm of the difference between the two vectors.

Are the two vectors equal?

```
>> difference = norm(vec_final- eg_vec)

difference =   3.3989e-010
```

The difference between the vectors is very small (Norm ~ e-10)

Which are the 10 users with the highest eigenvector centrality? Output the user IDs.

```
>> for i = [1:10]

  indicies(i) = find(vec_final == vec_final_s(i));

endfor

>> indicies

indicies =

   9904    2322    5170    5157    2362    7765    3943     133    2332    1902
```

What are the 10 users with highest degree in the network?

```
indicies_d =

   2332     471     554    2322      23     451    2208    9904    1463    3943
```

Do they correspond to the 10 users with highest eigenvector centrality? Why?

Corresponding nodes are:

Eigenvector number 1 and Degree number 8
Eigenvector number 2 and Degree number 4
Eigenvector number 7 and Degree number 10
Eigenvector number 9 and Degree number 1
Some of the nodes are in both top 10, but not all. This is because a high node degree centrality does not mean this node will have a high eigen vector centrality. But it could be a good indicator, and influence it.

What degree do the 10 users with largest eigenvector centrality have?

```
>> deg_top_eig = degrees(indicies)

deg_top_eig =

Compressed Column Sparse (rows = 1, cols = 10, nnz = 10 [100%])

   (1, 1) ->  722
   (1, 2) ->  797
   (1, 3) ->  537
   (1, 4) ->  504
   (1, 5) ->  606
   (1, 6) ->  457
   (1, 7) ->  618
   (1, 8) ->  523
   (1, 9) ->  1098
   (1, 10) ->  589
```

PageRank (4 Points)

Compute the PageRank for all users using power iteration, with the random teleportation

parameter = 0:15.

Which are the 10 users with the highest PageRank? Output the user IDs.

What are the 10 users with the most followers in the network? Do they correspond to the 10 users with highest PageRank? Why? How many followers do the 10 users with largest PageRank have?

Users with the most followers are the ones with **highest in degree**.

```
>> in_degree = sum(C,1);

>> in_degree_s = sort(in_degree, "descend");

>> for i = [1:10]

  find(in_degree == in_degree_s(i))

endfor

ans =  643
ans =

   1405   5486

ans =

   1405   5486

ans =  1379
ans =  1471
ans =  2748
ans =  2467
ans =  612
ans =  14047
ans =  5449
```

We have implemented page rank with power iteration, but we run out of memory when calculating P2.

```
%Task 3

C = load_sparse_directed("out.munmun_twitter_social");
alpha = 0.15;

c_deg = sum(C,2);
n=size(C)(1);

P1 = sparse(n,n);
P2 = sparse(n,n);

c_deg_inv = c_deg.^-1;
c_deg_inv(find(c_deg==0))=0;

P1 = (diag(c_deg_inv))*C;
c_deg(c_deg==0) = -1;
c_deg(c_deg>0) = 0;
c_deg(c_deg==-1) = 1;

P2 = (1/n)*c_deg*sparse(ones(1,n));

diff = 1e-5;
v = ones(1,n)/n;
while 1
  old_v = v;
  v = (v*(1-alpha)*P) + v*alpha*(ones(1,n)/n)';
  if norm(v-old_v)<diff
    break;
  endif
endwhile

max_page_rank = sort(v, "descend");
max_page_rank(1:10)
```

4 Centralities (Extra5)