```
%%load the data
m=load('imr90chr2.mat');
data=m.imr90chr2;
%construct degree matrix
D=zeros(size(data,1));
for i1 = 1:size(D,1)
    for j1 = 1:size(D,1)
            D(i1,i1) = D(i1,i1)  + data(i1,j1);
    end
end

%k=6 because it is the first eigenvalue where
%the increasing of the eigenvalue decreased
k=6;

%%(a)
%timing for unnormalized clustering
tic;
L_un=D-data;
[V_un,D_un] = eigs(L_un,k, 'smallestabs');
idx_un = kmeans(V_un, k);
t_un = toc;
A_un = kMat(idx_un);
% visualize output
figure, imagesc(A_un), axis square
colormap(gca,'jet');
title(sprintf('unnormalized clustering kmeans, k=%i',k));


%timing for normalized_ng clustering
tic;
L_sym = eye(size(data,1)) - D^-0.5 * data * D^-0.5;
[V_sym,D_sym] = eigs(L_sym,k, 'smallestabs');
V_sym_k = normalize(V_sym,2);
idx_sym = kmeans(V_sym_k, k);
t_sym = toc;
A_sym = kMat(idx_sym);
% visualize output
figure, imagesc(A_sym), axis square
colormap(gca,'jet');
title(sprintf('normalized ng clustering kmeans, k=%i',k));

%timing for normalized_malik clustering
tic;
L_rw = eye(size(data,1)) - D^-1 * data;
[V_rw,D_rw] = eigs(L_rw,k, 'smallestabs');
idx_rw = kmeans(V_rw, k);
t_rw = toc;
A_rw = kMat(idx_rw);
% visualize output
figure, imagesc(A_rw), axis square
colormap(gca,'jet');
```

```
title(sprintf('normalized malik clustering kmeans, k=%i',k));


%%(b)
%malik method has the shortest time.
%accuracy for normalized k-means clustering is better than
 unnormalized
%clustering

%%(c)
%plot of smallest 20 eigenvalues using L_sym
eigen=eigs(L_un,20,'smallestabs');
figure;plot(eigen);
title(sprintf('eigenvalue'));
%k=6 because it is the first eigenvalue where
%the increasing of the eigenvalue decreased


%%(d)
%agglomerative clusters
T_un = clusterdata(V_un, k);
T_sym = clusterdata(V_sym_k, k);
T_rw = clusterdata(V_rw, k);

A_un_c = kMat(T_un);
% visualize output
figure, imagesc(A_un_c), axis square
colormap(gca,'jet');
title(sprintf('unnormalized agglo cluster k=%i',k));

A_sym_c = kMat(T_sym);
% visualize output
figure, imagesc(A_sym_c), axis square
colormap(gca,'jet');
title(sprintf('normalized ng agglo cluster k=%i',k));

A_rw_c = kMat(T_rw);
% visualize output
figure, imagesc(A_rw_c), axis square
colormap(gca,'jet');
title(sprintf('normalized malik agglo cluster k=%i',k));




function [A] = kMat(idx)

%kMat creates a matrix from k-means cluster output
%   if idx(i) == idx(j), A(i,j) = idx(i)
%   else A(i,j) = 0
%
%   Scott Ronquist, 3/14/19. scotronq@umich.edu
%
```

```
A = zeros(size(idx));
for iK = 1:max(idx)
    kIdxTemp = find(idx == iK);
    A(kIdxTemp,kIdxTemp) = iK;
end

end
```

```
Warning: First input matrix is close to singular or badly scaled.
 RCOND =
1.857747e-17. Results may be inaccurate.
Warning: First input matrix is close to singular or badly scaled.
 RCOND =
2.797518e-17. Results may be inaccurate.
Warning: First input matrix is close to singular or badly scaled.
 RCOND =
2.158832e-17. Results may be inaccurate.
Warning: First input matrix is close to singular or badly scaled.
 RCOND =
1.857747e-17. Results may be inaccurate.
```

Published with MATLAB® R2019a