Problem 1

Part 1:

In this problem, I first set up the matrix M by implement the data of the matches. Then, normalize every row in M if their sum is not 0. Then set up w0 with uniform distributed random variable. Then do the iteration.

For iteration, I set up a 1001 column matrix u, the first is used to store w0, then store w1 to w1000. After the 1000 iterations, according to the problem, I got u(:,11), u(:,101), u(:,201), u(:,1001) for iteration 10, 100, 200, 1000. For each of them, find the highest 20 possibilities, and display their name. The result is as followed:

t= 10 :

'UW-Whitewater' '\t\t' [0.0284]

'ColoradoSt-Pueblo' '\t\t' [0.0218]

'MountUnion' '\t\t' [0.0204]

'OhioState' '\t\t' [0.0195]

'MinnSt-Mankato' '\t\t' [0.0145]

'Oregon' '\t\t' [0.0143]

'Linfield' '\t\t' [0.0136]

'SouthernOregon' '\t\t' [0.0133]

'NorthDakotaSt' '\t\t' [0.0132]

'Wesley' '\t\t' [0.0129]

'Alabama' '\t\t' [0.0115]

'TCU' '\t\t' [0.0108]

'FloridaSt' '\t\t' [0.0108]

'Wartburg' '\t\t' [0.0097]

'Concord' '\t\t' [0.0097]

'Hobart' '\t\t' [0.0096]

'Amherst' '\t\t' [0.0096]

'MaryHardin-Baylor' '\t\t' [0.0090]

'IllinoisSt' '\t\t' [0.0089]

'FerrisSt' '\t\t' [0.0088]

t= 100 :

'UW-Whitewater' '\t\t' [0.2076]

'OhioState' '\t\t' [0.0934]

'ColoradoSt-Pueblo' '\t\t' [0.0585]

'Oregon' '\t\t' [0.0544]

'FloridaSt' '\t\t' [0.0332]

'SouthernOregon' '\t\t' [0.0322]

'CarrollMT' '\t\t' [0.0234]

'TCU' '\t\t' [0.0224]

'GeorgiaTech' '\t\t' [0.0176]

'Alabama' '\t\t' [0.0170]

'VirginiaTech' '\t\t' [0.0143]

'Arizona' '\t\t' [0.0141]

'Amherst' '\t\t' [0.0132]

'Baylor' '\t\t' [0.0116]

'UCLA' '\t\t' [0.0109]

'Georgia' '\t\t' [0.0106]

'BoiseSt' '\t\t' [0.0098]

'Harvard' '\t\t' [0.0098]

'FerrisSt' '\t\t' [0.0094]

'MichiganSt' '\t\t' [0.0092]

t= 200 :

'UW-Whitewater' '\t\t' [0.2206]

'OhioState' '\t\t' [0.1156]

'Oregon' '\t\t' [0.0667]

'FloridaSt' '\t\t' [0.0419]

'ColoradoSt-Pueblo' '\t\t' [0.0271]

'SouthernOregon' '\t\t' [0.0269]

'TCU' '\t\t' [0.0255]

'GeorgiaTech' '\t\t' [0.0217]

'CarrollMT' '\t\t' [0.0198]

'Alabama' '\t\t' [0.0196]

'VirginiaTech' '\t\t' [0.0177]

'Arizona' '\t\t' [0.0170]

'Amherst' '\t\t' [0.0132]

'Baylor' '\t\t' [0.0131]

'UCLA' '\t\t' [0.0130]

'Georgia' '\t\t' [0.0125]

'BoiseSt' '\t\t' [0.0114]

'MichiganSt' '\t\t' [0.0105]

'Harvard' '\t\t' [0.0100]

'Mississippi' '\t\t' [0.0097]

t= 1000 :

'UW-Whitewater' '\t\t' [0.2209]

'OhioState' '\t\t' [0.1345]

'Oregon' '\t\t' [0.0769]

'FloridaSt' '\t\t' [0.0494]

'TCU' '\t\t' [0.0286]

'GeorgiaTech' '\t\t' [0.0254]

'Alabama' '\t\t' [0.0222]

'VirginiaTech' '\t\t' [0.0207]

'Arizona' '\t\t' [0.0193]

'Baylor' '\t\t' [0.0147]

'UCLA' '\t\t' [0.0146]

'Georgia' '\t\t' [0.0143]

'Amherst' '\t\t' [0.0132]

'BoiseSt' '\t\t' [0.0125]

'Harvard' '\t\t' [0.0119]

'MichiganSt' '\t\t' [0.0118]

'Mississippi' '\t\t' [0.0109]

'ArizonaSt' '\t\t' [0.0101]

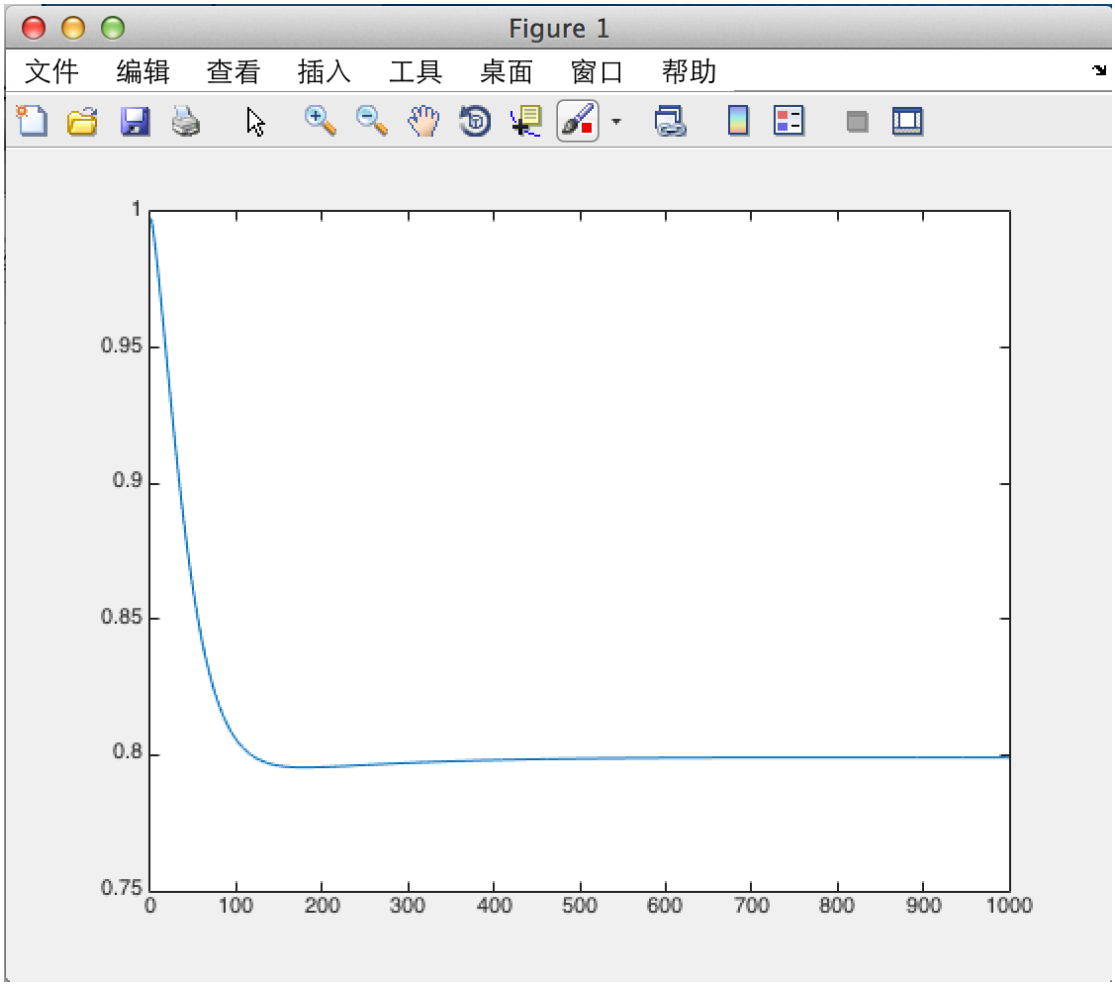
'Clemson' '\t\t' [0.0092]

'Duke' '\t\t' [0.0082]

Part 2:

Base on part 1, find the eigenvalue and eigenvector of M^T, then select the eigenvector that is corresponding to eigenvalue of 1,. Then normalize it so that its sum of value is 1. This is u1 in the formula.

After that, I calculate and record the value of the question formula, then plot it. The graph is as followed:



Also, the when we have iteration of 1000, the result of the formula is :

When t=1000, the result is :

0.7991

There is another question, I find that the u1 of this question has one ‘1’, and other dimension is all ‘0’. This means finally the players will all go to one team. Then I start to doubt whether it is right. So I find the team the ‘1’ indicates. The result is:

The only winner is :
    'UW-Whitewater'

And it is the same with the most top team in all iterations when t=10,100,200,1000. So I think that it may be possible. With large enough iteration, all players will go to the most powerful team. And when t is not large enough, the most powerful team is most likely to list at the top and goes to 1 gradually.

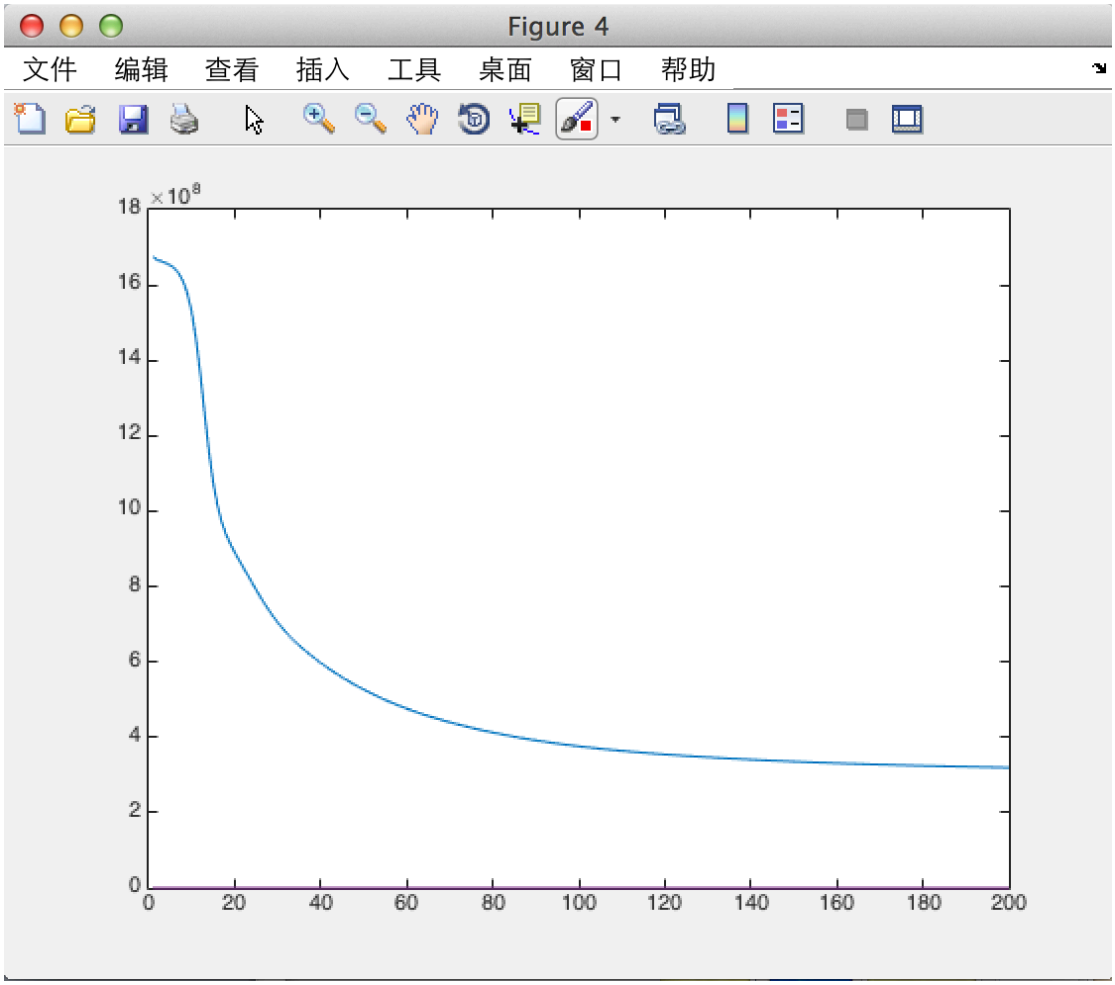
Also, with the graph we find, we can tell that the rate of concentration becomes slower and slower.

Problem 2

Part 1:

In this problem, matrix M is just the matrix imported by face.csv.

Then I do the iteration to update W and H, meanwhile record the objective function value for each iteration in an array. Then plot it, the graph is listed below:

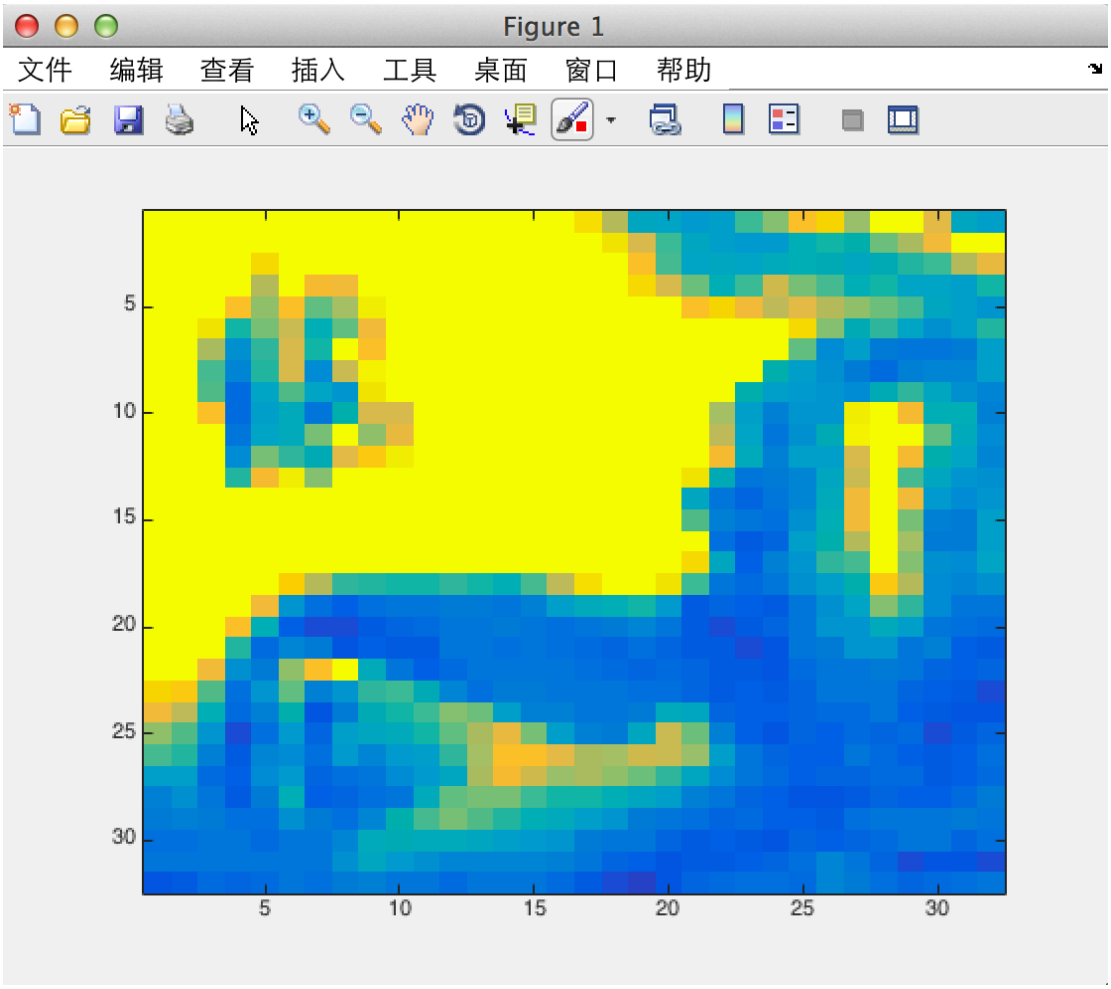


Then code to get the 3 graphs , output the graph. Find the largest corresponding W column index k1, k2, k3. The result is listed below:

Graph index 1 is 758, the corresponding k is

k1 =

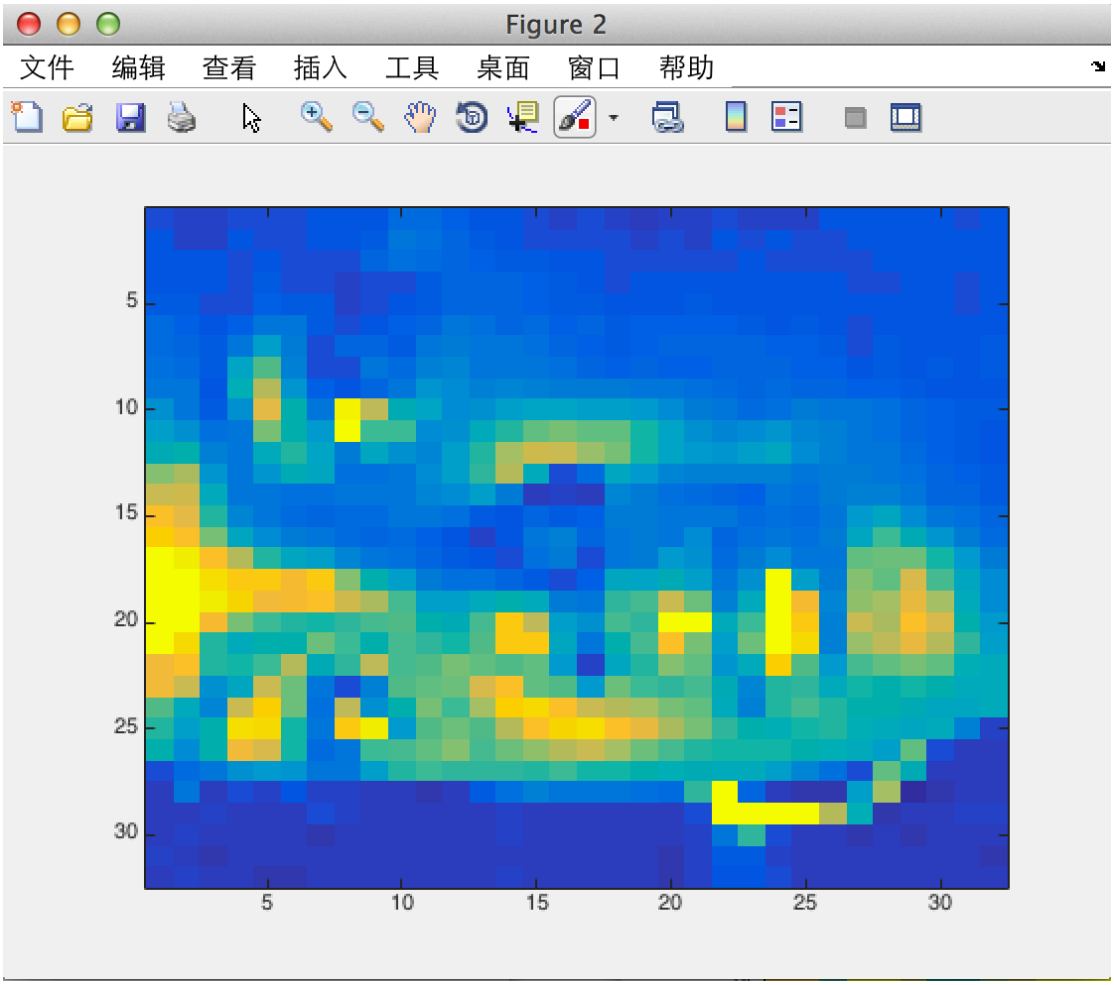
22



Graph index 2 is 832, the corresponding k is

k2 =

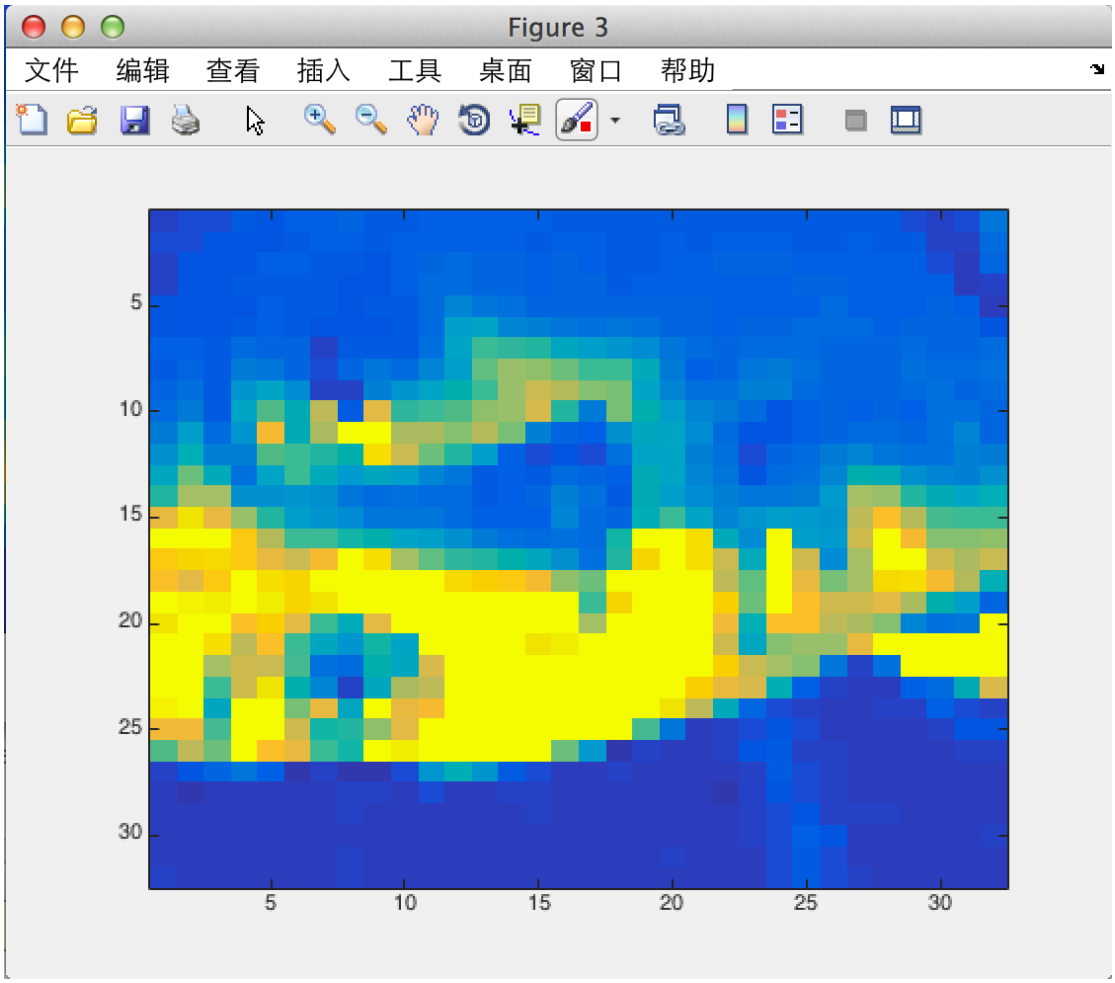
15



Graph index 3 is 396, the corresponding k is

k3 =

11

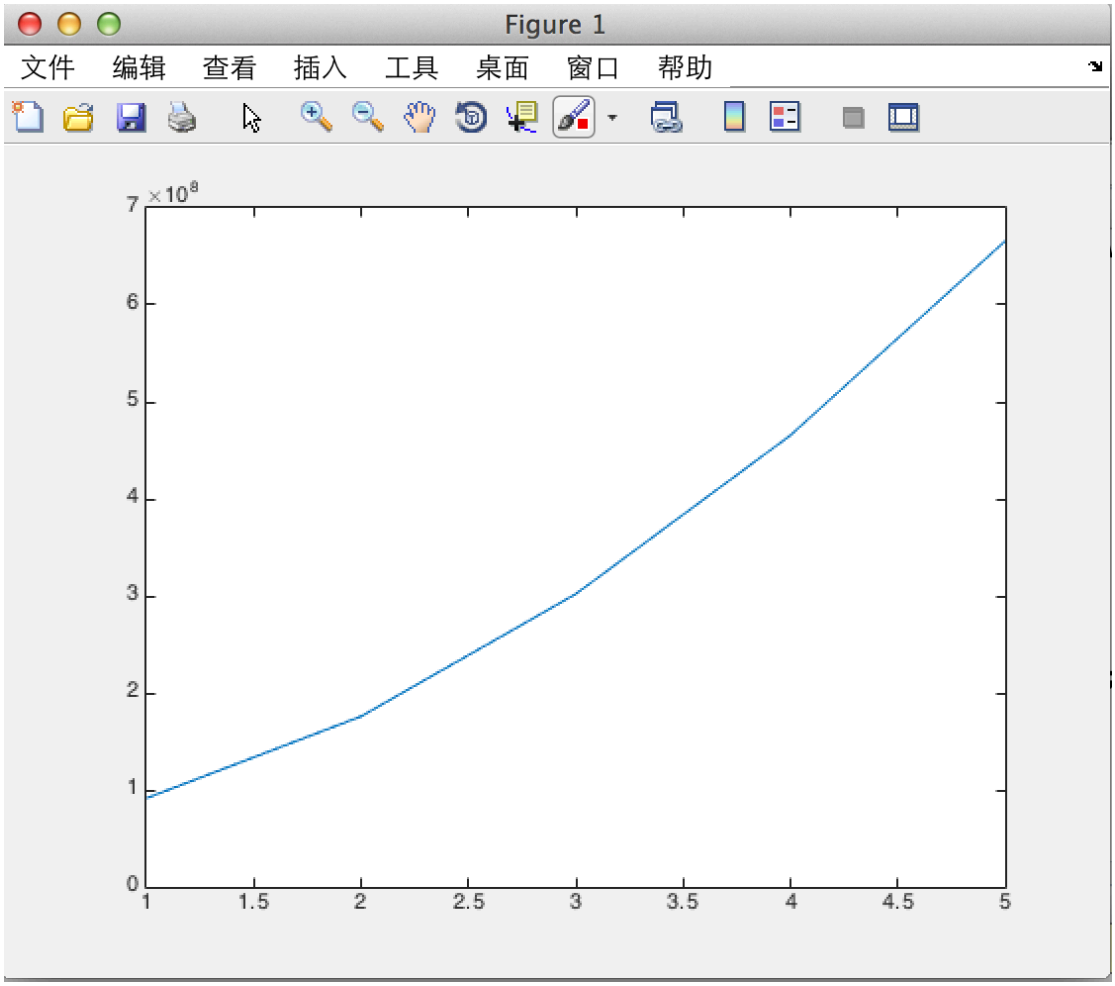


Then the columns of W with index k1=22, k2=15, k3=11 (they are different).

Part 2:

At first, I decompose the data imported from the text, then use these data to form the X matrix, with dimension (words, articles)=(3012, 8447). The value in each cell is the times that this word appear in this article. I successfully implement this matrix. The name is X.

Then I tried to implement the algorithm of divergence penalty. Yet after days of test, I still got problems at last. For one time, I got the graph of objective function is listed below.



I also implement the code for the last part of the problem, finding 10 words with largest possibilities. The output of code for the last question is list below, which can find the 10 maximum value and return their index. We can see that all the result below is with possibility 1, since I got the algorithm part some mistake. Although this part works, the result is wrong.

The column index is 1, the corresponding word and possibilities are

Record =

1.0e+03 *

2.8920	0.0010
0.0280	0.0010
0.8590	0.0010
0.2760	0.0010

3.1010 0.0010
1.4950 0.0010
3.0110 0.0010
1.5500 0.0010
2.4780 0.0010
1.3490 0.0010

The column index is 5, the corresponding word and possibilities are

Record =

1.0e+03 *

2.7340 0.0010
0.4970 0.0010
1.5830 0.0010
1.6860 0.0010
1.8360 0.0010
1.4520 0.0010
1.4500 0.0010
0.2240 0.0010
2.6560 0.0010
1.3810 0.0010

The column index is 9, the corresponding word and possibilities are

Record =

1.0e+03 *

0.6760 0.0010
2.7470 0.0010
0.8720 0.0010
0.2340 0.0010
2.5420 0.0010
0.7220 0.0010
1.7200 0.0010
0.0300 0.0010
2.6160 0.0010
2.2350 0.0010

The column index is 16, the corresponding word and possibilities are

Record =

1.0e+03 *

0.3040	0.0010
0.4420	0.0010
1.6870	0.0010
0.5600	0.0010
3.1000	0.0010
1.2460	0.0010
1.3150	0.0010
1.8940	0.0010
0.8450	0.0010
0.5520	0.0010

The column index is 24, the corresponding word and possibilities are

Record =

1.0e+03 *

0.3110	0.0010
0.6330	0.0010
0.5110	0.0010
2.4180	0.0010
1.3080	0.0010
0.3180	0.0010
2.8800	0.0010
1.0130	0.0010
2.5420	0.0010
1.8030	0.0010

I listed all the code for this part in the attachment part. There's a lot of code is in form of annotation, which stands for the test and trying of the work. I believe the code have some mistake in algorithm part, yet finally I can't figure all of them.

Source Code Attachment

For problem 2, the part 2 and part 3 code must have the result in part 1 in working region. Or we can just put code 1,2,3 one by one and we can get all results.

Problem 1,

```
clear;
legend=importdata('legend.txt');
cfb=importdata('cfb2014.csv');

fid = fopen('cfb2014.csv');
A= textscan(fid, '%s %f %s %f','delimiter',',');
team1=A{1};
score1=A{2};
team2=A{3};
score2=A{4};
fclose(fid);
[matches,y]=size(team1);

[teams,y]=size(legend);
M=zeros(teams,teams);
sumRecord=zeros(teams,1);

for i=1:matches
    name1=char(team1(i));
    name1(find(isspace(name1)))=[];
    [index1,y]=find(strcmp(legend,name1));
    name2=char(team2(i));
    name2(find(isspace(name2)))=[];
    [index2,y]=find(strcmp(legend,name2));

    if (score1(i)>score2(i))

M(index1,index1)=M(index1,index1)+score1(i)/(score1(i)+score2(i));

M(index2,index1)=M(index2,index1)+score1(i)/(score1(i)+score2(i));

sumRecord(index1,1)=sumRecord(index1,1)+score1(i)/(score1(i)+score2(i
));

sumRecord(index2,1)=sumRecord(index2,1)+score1(i)/(score1(i)+score2(i
));
    end
    if (score1(i)<score2(i))

M(index2,index2)=M(index2,index2)+score2(i)/(score2(i)+score1(i));

M(index1,index2)=M(index1,index2)+score2(i)/(score2(i)+score1(i));
```

```
sumRecord(index1,1)=sumRecord(index1,1)+score2(i)/(score1(i)+score2(i
));

sumRecord(index2,1)=sumRecord(index2,1)+score2(i)/(score1(i)+score2(i
));
    end
end

for i=1:teams
    if (sumRecord(i,1)~=0)
        M(i,:)=M(i,:)/sumRecord(i,1);
    end
end

u=zeros(teams,1001);

for j=1:1001
    if (j==1)
        for i=1:teams
            u(i,j)=1/teams;
        end
    end
    if (j~=1)
        u(:,j)=transpose(M)*u(:,j-1);
    end
end

tmpU=u;
    disp('t= 10 : ');
    %disp();
    for i=1:20
        [x,y]=max(tmpU(:,11));
        disp([legend(y),'\t\t',tmpU(y,11)]);
        %disp(tmpU(y,11));
        tmpU(y,11)=0;
    end

tmpU=u;
    disp('t= 100 : ');
    %disp();
    for i=1:20
        [x,y]=max(tmpU(:,101));
        disp([legend(y),'\t\t',tmpU(y,101)]);
```

```
        %disp(tmpU(y,11));
        tmpU(y,101)=0;
    end

tmpU=u;
    disp('t= 200 : ');
    %disp();
    for i=1:20
        [x,y]=max(tmpU(:,201));
        disp([legend(y),'\t\t',tmpU(y,201)]);
        %disp(tmpU(y,11));
        tmpU(y,201)=0;
    end

tmpU=u;
    disp('t= 1000 : ');
    %disp();
    for i=1:20
        [x,y]=max(tmpU(:,1001));
        disp([legend(y),'\t\t',tmpU(y,1001)]);
        %disp(tmpU(y,11));
        tmpU(y,1001)=0;
    end



[V,D]=eig(transpose(M));
findIndex=0;
for i=1:teams
    if D(i,i)==1
        findIndex=i;
    end
end
findIndex
D(findIndex,:)
u1=V(:,findIndex);

u1Sum=0;
for i=1:teams
    u1Sum=u1Sum+u1(i,1);
end
wInfinity=u1/u1Sum;

graph=zeros(teams,2);
```

```
for t=1:1000
    graph(t,1)=t;
    y=0;
    for i=1:teams
        y=y+(u(i,t+1)-wInfinity(i,1))*(u(i,t+1)-wInfinity(i,1));
    end
    graph(t,2)=sqrt(y);
end
%graph
plot(graph(:,2));

disp('When t=1000, the result is : ');
disp(graph(1000,2));

%find
[x,y]=max(wInfinity(:));
disp('The only winner is : ');
disp(legend(y));
```

Problem 2

(1) matlab code for part 1

```
clear;
faces=importdata('faces.csv');
[dimensions,objects]=size(faces);

X=faces;
rank=25;
W=rand(dimensions,rank);
H=rand(rank,objects);
w0=rand(dimensions,rank);
h0=rand(rank,objects);
a=X;
maxiter=200;
size(w0);
size(w0');
obGraph=zeros(maxiter);
for j=1:maxiter
    % Multiplicative update formula
    numer = w0'*a;
```

```
    %h = h0 .* (numer ./ ((w0'*w0)*h0 + eps(numer)));
    h = h0 .* (numer ./ ((w0'*w0)*h0 ));
    numer = a*h';
    %w = w0 .* (numer ./ (w0*(h*h') + eps(numer)));
    w = w0 .* (numer ./ (w0*(h*h')));
    w0 = w; h0 = h;
    E=w*h;
    for x=1:dimensions
        for y=1:objects
            obGraph(j)=obGraph(j)+(X(x,y)-E(x,y))*(X(x,y)-E(x,y));
        end
    end
end
figure(4);
plot(obGraph);

% %disp(tmpGraph);
% figure(tmpGraph)
% image(tmpGraph)
% colorbar

indexList=zeros(dimensions,3);

% 1st graph
index1=758;
tmpGraphData=faces(:,index1);

tmpGraph=zeros(32,32);
for i=1:32
    for j=1:32
        tmpGraph(i,j)=tmpGraphData(32*(i-1)+j);
    end
end

figure(1);
image(tmpGraph);
%colorbar
[x,k1]=max(h(:,index1));
disp('Graph index 1 is 758, the corresponding k is');
k1
indexList(:,1)=w(:,k1);
%disp(w(:,k1));
```

```
% 2nd graph
index2=832;
tmpGraphData=faces(:,index2);

tmpGraph=zeros(32,32);
for i=1:32
    for j=1:32
        tmpGraph(i,j)=tmpGraphData(32*(i-1)+j);
    end
end

figure(2);
image(tmpGraph);
%colorbar
[x,k2]=max(h(:,index2));
disp('Graph index 2 is 832, the corresponding k is');
k2
indexList(:,2)=w(:,k2);
%disp(w(:,k2));


% 3rd graph
index3=396;
tmpGraphData=faces(:,index3);

tmpGraph=zeros(32,32);
for i=1:32
    for j=1:32
        tmpGraph(i,j)=tmpGraphData(32*(i-1)+j);
    end
end

figure(3);
image(tmpGraph);
%colorbar
[x,k3]=max(h(:,index3));
disp('Graph index 3 is 396, the corresponding k is');
k3
indexList(:,3)=w(:,k3);
%disp(w(:,k3));

disp('The three corresponding column in W is');
indexList
```

(2) matlab code for part 2

```
clear;
nyt_data=importdata('nyt_data.txt');
%[dimensions,objects]=size(nyt_data);

%[articles,y]=size(nyt_data);
%articles;
%whos nyt_data;
X=zeros(3012,8447);

% nyt_data{1}
% S = regexp(nyt_data{1,1}, ',', 'split');
% [x,size]=size(S);
% %for i=1:size
% tmp=regexp(S(1,1), ':', 'split');
% word=str2double(tmp{1}(1));
% times=str2double(tmp{1}(2));
% %word=str2num(word)
% %times=str2num(times)
% X(word,1)=X(word,1)+times;
% % tmp=cell2mat(nyt_data{1});
% % tmp

for i=1:8447
    %nyt_data{i};
    S = regexp(nyt_data{i}, ',', 'split');
    [x,y]=size(S);
    for j=1:y
        tmp=regexp(S(1,j), ':', 'split');
        word=str2double(tmp{1}(1));
        times=str2double(tmp{1}(2));
        X(word,i)=X(word,i)+times;
%         if (i==1 && j==1)
%             word
%             times
%             %X(word,i)
%         end
    end
end

%X(1946,1)
```

```
rank=25;
dimensions=3012;
objects=8447;
%W=rand(dimensions,rank);
%H=rand(rank,objects);
w0=rand(dimensions,rank);
h0=rand(rank,objects);
%a=X;
maxiter=200;
obGraph=zeros(maxiter,1);
epsilon=1e-16;
Wrecord=zeros(dimensions,rank,maxiter);
for j=1:5
    % Multiplicative update formula
    numer = X./ (w0*h0+epsilon);
    j
    w0(dimensions,rank)
    h0(rank,objects)
%     if j==10 || j==25 || j==50 || j==100 || j==200
%         w0(dimensions,rank)
%     end

    % normalize w
    %norW0=transpose(w0);
    norW0=w0';
    for a=1:rank
        tmpSum=sum(norW0(a,:));
        %a
        %tmpSum
        if tmpSum~=0
            norW0(a,:)=norW0(a,:)/tmpSum;
        end
        %sum(norW0(a,:))
    end
    %norW0(2,3)
    h0;
    %X(3100,:)
    h = h0 .* (norW0*numer);

    numer = X./ (w0*h+epsilon);
    % normalize h
    %norH=transpose(h);
```

```
    norH=h';

    for a=1:rank
        tmpSum=sum(norH(:,a));
        %a
        %tmpSum
        if tmpSum~=0
            norH(:,a)=norH(:,a)/tmpSum;
        end
        if j==3
            %norW0(a,:)
        end
        %sum(norH(:,a))
        %norH
    end
    %w0;
    %w = w0 ./ (numer*norH+epsilon);
    %norH
    %numer
    %tt=numer*norH;
    %numer(3000,:)
    %w0;
    %tt(304,16);
    w = w0 ./ (numer*norH+epsilon);
    %w(3012,:)

    w0 = w; h0 = h;
    Wrecord(:,:,j)=w0;
    E=w*h;
    for x=1:dimensions
        for y=1:objects
            obGraph(j,1)=obGraph(j,1)+E(x,y)-X(x,y)*log(E(x,y));
        end
    end
end
figure(1);
plot(obGraph(:,1));



% normalize W
finalW=w;
for a=1:rank
    tmpSum=0;
```

```
    for b=1:dimensions
        tmpSum=tmpSum+finalW(b,a);
    end
    if tmpSum~=0
        finalW(:,a)=finalW(:,a)/tmpSum;
    end
end

Record=zeros(10,2);
Indexes=zeros(10);
Possibilities=zeros(10);
% 1st column
index1=1;
column=finalW(:,index1);

for i=1:10
    [x,y]=max(column(:));
    Record(i,1)=y;
    Record(i,2)=x;
    Indexes(i)=y;
    Possibilities(i)=x;
    column(y)=0;
end

disp('The column index is 1, the corresponding word and possibilities
are');
Record
%Indexes
%Possibilities

% 2nd column
index2=5;
column=finalW(:,index2);

for i=1:10
    [x,y]=max(column(:));
    Record(i,1)=y;
    Record(i,2)=x;
    Indexes(i)=y;
    Possibilities(i)=x;
    column(y)=0;
end
```

```
disp('The column index is 5, the corresponding word and possibilities
are');
Record
%Indexes
%Possibilities

% 3rd column
index3=9;
column=finalW(:,index3);

for i=1:10
    [x,y]=max(column(:));
    Record(i,1)=y;
    Record(i,2)=x;
    Indexes(i)=y;
    Possibilities(i)=x;
    column(y)=0;
end

disp('The column index is 9, the corresponding word and possibilities
are');
Record
%Indexes
%Possibilities

% 4th column
index4=16;
column=finalW(:,index4);

for i=1:10
    [x,y]=max(column(:));
    Record(i,1)=y;
    Record(i,2)=x;
    Indexes(i)=y;
    Possibilities(i)=x;
    column(y)=0;
end

disp('The column index is 16, the corresponding word and possibilities
are');
Record
%Indexes
%Possibilities
```

```
% 5th column
index5=24;
column=finalW(:,index5);

for i=1:10
    [x,y]=max(column(:));
    Record(i,1)=y;
    Record(i,2)=x;
    Indexes(i)=y;
    Possibilities(i)=x;
    column(y)=0;
end

disp('The column index is 24, the corresponding word and possibilities
are');
Record
%Indexes
%Possibilities
```