Rachel King

ECE 3522: Stochastic Processes in Signals and Systems

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 1912

# Problem Statement

This assignment has us dealing with a large set of data, unlike the much smaller sets which we normally deal with; the data “ratings” has 329 cities which follow 9 different criteria. The different criteria are what make this assignment difficult to deal with, considering we will need to find some sort of way to weight it. We will be dealing with a “whitening transform” which will test our linear algebra skills, as well as doing many comparisons.

# Approach and Results

*Compute the covariance matrix for the raw data (contained in the data matrix “ratings”).*



Figure - Covariance Matrix

load cities

c\_m = cov(ratings);

*Do an eigenvalue/eigenvector analysis of the covariance matrix, and display both these matrices. Discuss what you observe.*



Figure - Eigen Vector Matrix



Figure - Eigen Value Matrix

[V, D] = eig(c\_m);

As you can see, the Eigen value analysis yields a matrix with only values on the diagonal, and the Eigen vector matrix contains all the non-diagonal skew data

*Compute the whitening transform we discussed in class, and compute a matrix of transformed data.*



Figure - Snapshot of 329x329 whitening matrix

W = ratings\*V\*diag(1./(diag(D)+1).^(1/2))\*V';

Y = cov(W);

*Demonstrate that the covariance of the transformed data is an identity matrix by computing the covariance of the transformed data matrix.*



Figure - Covariance of whitening matrix, which is a 329x329 identity matrix

*Plot the percent of the variance accounted for by the first n eigenvalues for n = [1,9]. Make sure the eigenvalues are sorted from largest to smallest.*



Figure - Plot of percent of variance

for n = 1:9

 p(n) = (ev(n)/sum(ev));

end

plot(p)

% end

%highest two variances

%ev(8) -> recreation

%ev(9) -> economics

It seems as if the last two have the highest variances.

*Euclidean distances and ratings:*

%use a weighted sum to find a number that

%best represents each city

for i = 1:329

 for j = 1:9

 X = ev(j)\*p(j);

 Y = Y+X;

 end

 Z(i) = Y;

end

Z = Z';

First, make a weighted sum (shown in the code above) using the percentages which we calculated. This gives each city one signal number which is representative of each of the 9 categories.

Find the Euclidean distance between each city:

%compares each city, storing the distance

%in a new matrix, A

for i = 1:329

 for j = 1:329

 G1 = Z(i);

 G2 = Z(j);

 A(i, j) = sqrt(sum((G1 - G2) .^ 2));

 end

end

The results are stored in the matrix, A. Now, we need to find the minimum distance of the bottom triangle of the matrix (so that we are only doing each comparison twice):
%we only need either the bottom or top half

%of the distance matrix A, so we will first shorten the matrix

B = zeros(size(A));

for i = 1:329

 B(i+1:329, i) = A(i+1:329, i);

end

B(B==0) = nan;

%find the distance that is the smallest, and return

%the index of that city

[E, I] = min(B);

The minimum value will be what is returned in E, and the minimum index will be I, telling us which cities are closest together

# MATLAB Code

%Rachel King

clear all; clc;

% function CA\_12

load cities

c\_m = cov(ratings);

%V = eigenvectors

%D = eigenvalues

[V, D] = eig(c\_m);

W = ratings\*V\*diag(1./(diag(D)+1).^(1/2))\*V';

Y = cov(W);

Z = eye(329);

ev = sort(diag(D), 'descend');

for n = 1:9

 p(n) = (ev(n)/sum(ev));

end

plot(p)

% end

%highest two variances

%ev(8) -> recreation

%ev(9) -> economics

R = ratings;

Y = 0;

%use a weighted sum to find a number that

%best represents each city

for i = 1:329

 for j = 1:9

 X = ev(j)\*p(j);

 Y = Y+X;

 end

 Z(i) = Y;

end

Z = Z';

%compares each city, storing the distance

%in a new matrix, A

for i = 1:329

 for j = 1:329

 G1 = Z(i);

 G2 = Z(j);

 A(i, j) = sqrt(sum((G1 - G2) .^ 2));

 end

end

%we only need either the bottom or top half

%of the distance matrix A, so we will first shorten the matrix

B = zeros(size(A));

for i = 1:329

 B(i+1:329, i) = A(i+1:329, i);

end

B(B==0) = nan;

%find the distance that is the smallest, and return

%the index of that city

[E, I] = min(B);

# Conclusions

This computer assignments is incomplete only due to time, but was my favorite so far because we get to work with what felt like real world data. Computing the whitening transform removed a good bit of noise from the system I think. Using the Eigen analysis, we were able to make a find the percent of the variance that each rating represented of the overall system, which allowed us to make a weighted sum, which was really important because each criteria was on a different scale. I used that weighted sum to created one value that represented each city, allowing us to find the Euclidean distance between each of the cities. However, due to my poor time management, I was never able to see the results of my work, nor was I able to see the effect that using the whitening transform had on my data. However, I did learn a good bit about comparing large amounts of data and storing that.