Principal Components Analysis of Commodities Data

ABSTRACT

PURPOSE: Use PCA in MATLAB to justify the classifications of the individualized datasets.

METHODS: PCA implemented in MATLAB using instructors functions and analyzed using code written.

RESULTS: Similar graphs for both z1 and z2

CONCLUSIONS:Dataset z1 is deduced to be food-related data and dataset z2 to be basic-material data

INTRODUCTION

The objective was to justify the classification of the commodity data using PCA in MATLAB. PCA stands for Principal Component Analysis and is a dimension-reduction tool that transforms possibly correlated variables into a smaller number of uncorrelated variables known as principal components. PCA allows us to present more from less, the small set can be reconstructed into the larger set and thus makes PCA great for making predictive models and data compression. The PCA code used was provided by the instructor as functions 'pcaprelim' and 'pcaapprox'. 'pcaprelim' returns the singular values and left singular vectors of the PCA which correlate to the eigenvalues/eigenvectors of the covariance matrix respectively as well as returns the mean signal (average vector of the data). 'pcaapprox' returns the reconstruction of data given as well as the principal components as a row vector of scalars. Using the information returned from these functions, we can determine the amount of principal components needed to capture most of the variation from the mean signal, as well as determine the error between the reconstructed signal and the original. By plotting the reconstructed signal against the original signal we can see first hand the effectiveness of PCA on large data sets.

METHODS

I began with loading in the dataset to be tested on and passing such dataset to the provided pcaprelim function. The pcaprelim function returns the eigenvalues of the covariance matrix, we can plot this vector to find a "knee" which indicates the number of principal components that should be used (k-value), however we can calculate that instead of hard coding values. We want the variation covered by the first k principal components to be about 50% but no more that 60%, we need to cycle through each "first k components" to find which k-value(s) fit our criteria. In this loop we must calculate the amount 'captured' at the k-value used, this value is checked to be around 50% (0.5) but no more than 60% (0.6). Calculating the 'captured' amount is the sum of the first-k singular values divided by the sum of all the singular values. Once a k-value fits the criteria, we can get to reconstructing data. The function pcaapprox provided by the instructor will do so for us, as long as it's passed the correct data. We must create a new loop that iterates through each original signal and with each signal it will be reconstructed using pcaapprox to which then we can calculate error, mean error (all signals) and plot the error by signal. In each iteration, pcaapprox takes the 'current signal', k-value, mean signal (from pcaprelim) and left singular vectors (from pcaprelim) to return the approximation. We can calculate the error of the approximation to the original signal as the square root of the sum of 1/100 the square difference between the approximation and original. We can put each error value in a matrix to which we can plot later, as well as accumulate the errors to find a mean after the loop is complete. Within this loop we also have two conditions to keep track of the reconstruction with the least and most error, which can be used for analysis. Upon finishing the signal loop, we can construct graphs of the signal with lowest error and highest error compared to its original, as well as calculate mean error and construct a graph for the errors of each signal. This procedure is repeated twice identically but with different datasets (z1, z2).

RESULTS

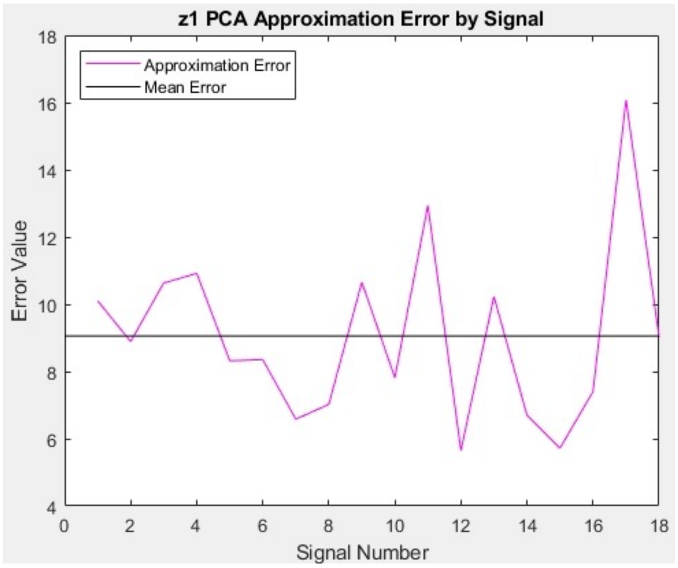

Figure 1. Error for each signal in z1.

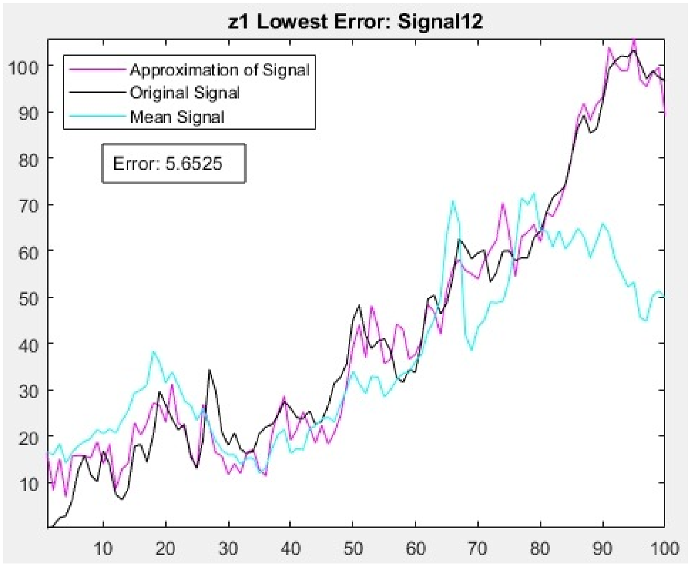

Figure 2. Signal in z1 with most accurate approximation (lowest error).

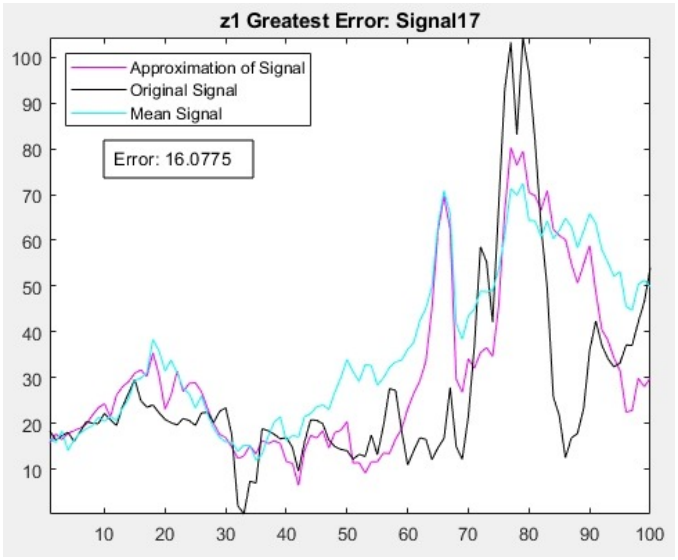

Figure 3. Signal in z1 with least accurate approximation, (greatest error).

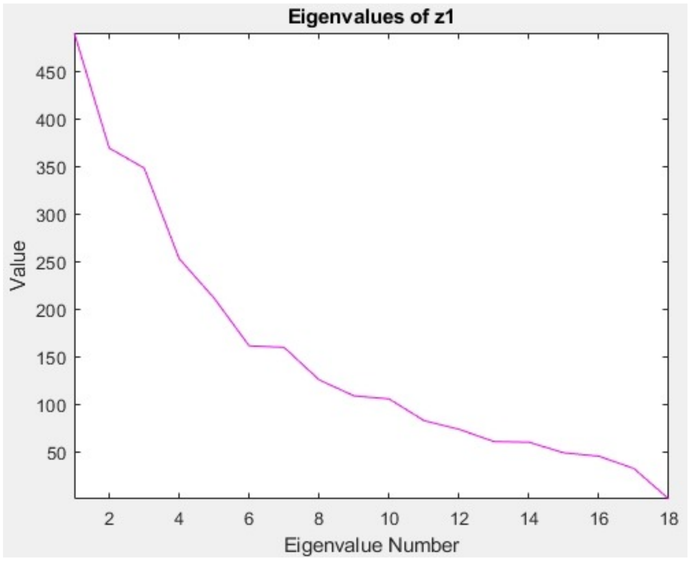

Figure 4. Eigenvalue plot for observing 'knee' in z1.

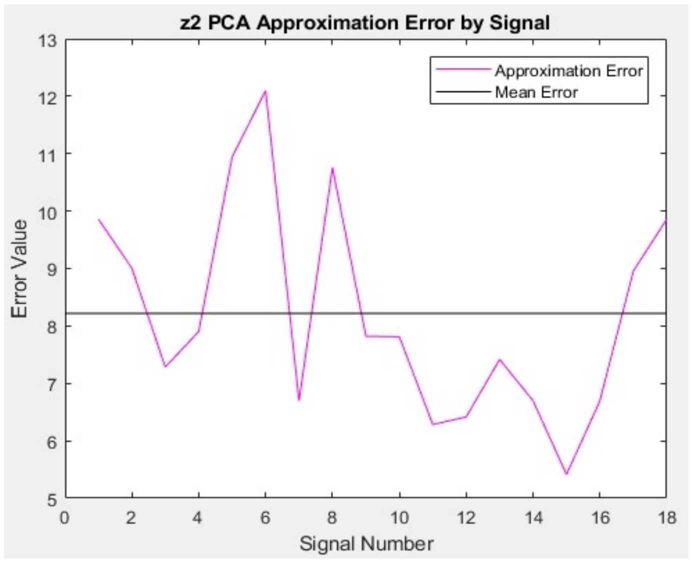

Figure 5. Error for each signal in z2.

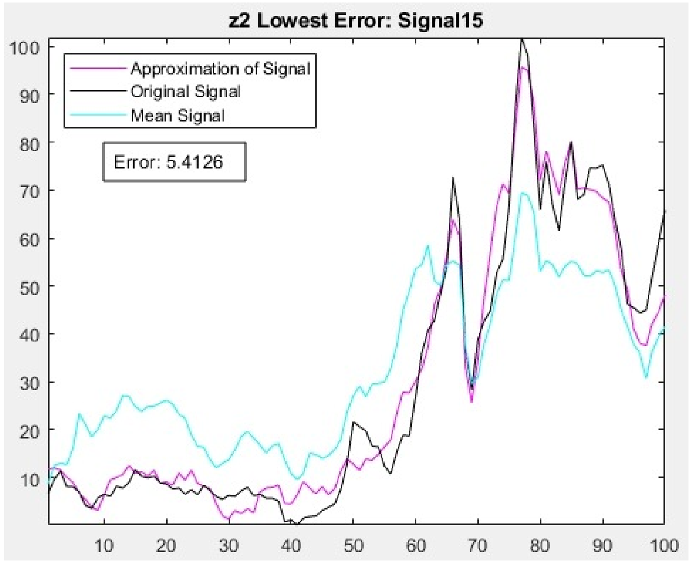

Figure 6. Signal in z2 with most accurate approximation (lowest error).

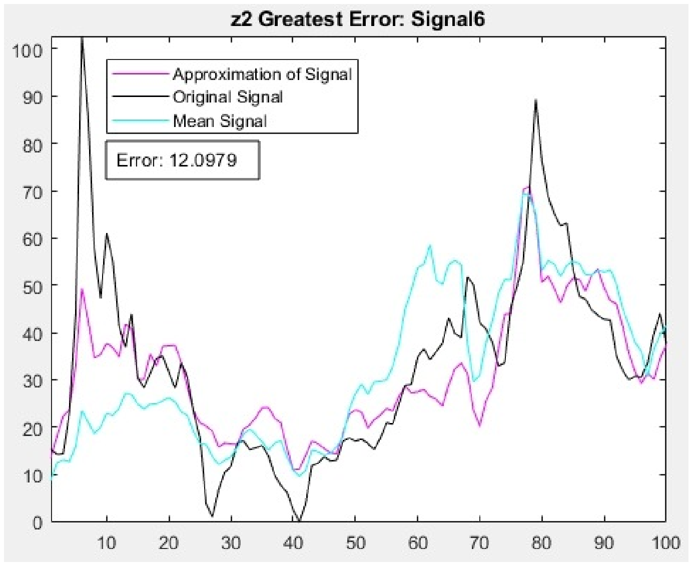

Figure 7. Signal in z2 with least accurate approximation, (greatest error).

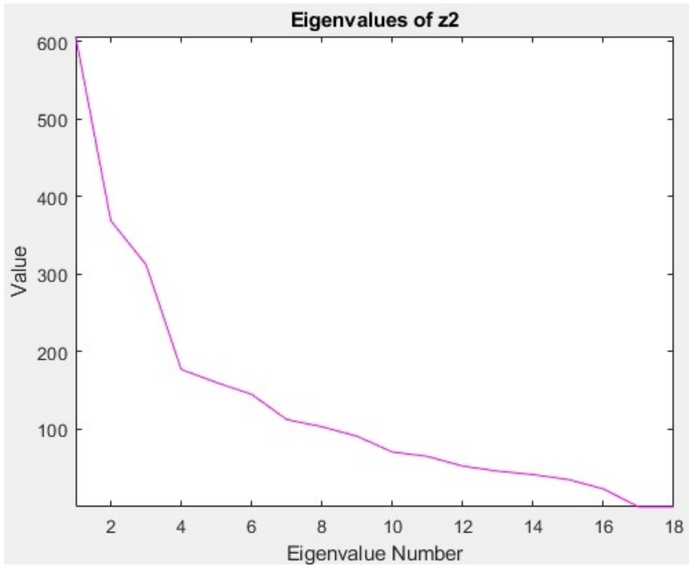

Figure 8. Eigenvalue plot for observing 'knee' in z2.

Mean Error for z1: 9.0574
Mean Error for z2: 8.2172
K-value for z1: 4
K-value for z2: 3

DISCUSSION

From the results above, we can see a lot of similarity between the graphs of z1 and z2 produced by the MATLAB code. From figures 2 and 6, we can see that the lowest error approximations of a signal are ~5.5 for both z1 and z2. We also can see from figures 4 and 8 that both datasets have a 'knee' at the k-value calculated from the code, which were 4 and 3 for z1 and z2 respectively. From figure 2, we can see that towards the end of the x-axis there is a significant difference between the mean and approximation, we can see this occur in figure 6 as well, however more prominent difference throughout. They match the original signal more so, which makes sense because they're the lowest error, but what's interesting is the the greater error signal of figures 3 and 7 match the mean signal a lot more than the original signal. It seems the signals with more 'extreme' data are harder to approximate, this is because PCA is a tool for predicting models, if the data within the signal is very distant from the mean ('extreme') then the approximation will be distant from the original since the mean is data used to approximate. From figures 1 and 5 we can notice something interesting, there is a very low error at signal 15 for both z1 and z2. This could mean the values of signal 15 are the most mean values for both z1 and z2 as they both would produce approximations very close to their original. Overall both z1 and z2 have a similar error for reconstruction, 9.0574 and 8.2172 respectively, this means for both z1 and z2, the error of the reconstructed matrix will be very similar and thus present an almost equally skewed set of data as a whole. In conclusion, by comparing figures 1 and 5 and taking into account the assumption that greater errors indicates extreme values, I deduce that z2 is the commodity of basic-materials. We can compare the mean signals between z1 and z2 and its fluctuations, for z2 we see very stock market like graphs in figures 6 and 7, prices of basic-materials fluctuate a lot as they can be traded on the stock market. The mean signal for z1 is very inflation-like, prices of food-related commodities have only increased over time and gradually.