# ECE 712

## Final Project

**Sara Jamil**
**001143947**
**jamils2@mcmaster.ca**
**1/16/2017**

# Table of Contents

## Introduction

The purpose of this project is to examine and analyse data using latent variable methods. These latent variable methods are very useful for fitting a model to a set of observed data and serve an important part of the machine learning framework. Using principal component analysis (PCA), partial least squares (PLS), and canonical correlation analysis (CCA) methods, the goal is to predict a set of response values from a previously unseen set of input variables. The applicability and effectiveness of these methods will be discussed.

Latent variable methods are used in many applications in signal processing, control theory, biomedical modelling, and many more. Latent variables are variables that are inferred from other variables that are observed or directly measured. An advantage of latent variable methods is that it reduces the dimensionality of the data such that a large number of observable variables can be aggregated to represent an underlying concept which makes the data easier to understand.

In this project we are given a set of independent variables X and their corresponding responses Y. Typically in these problems, X and Y are both highly rank deficient or very poorly conditioned and also very noisy. Therefore, the use of latent variable methods including principal component analysis, partial least squares, and canonical correlation analysis, which express the X and Y subspaces using only a few latent variables, are far more efficient at prediction than is ordinary least squares.

## Methods

In this project, we are given a set of independent variables X (500x10) and corresponding responses Y (500x6). To obtain these variables, an orthonormal basis T (10x2) was constructed and multiplied by a random matrix A (2x500) to obtain the true value $X_{tru}$, which is of rank 2. Then, $X_{tru}$ was multiplied by another random matrix B (6x10) to obtain the true value $Y_{tru}$. This true value of the responses was further modified by adding a factor Q (also multiplied by a random matrix C) which is orthogonal to $X_{tru}$ such that the rank of $Y_{tru}$ is 4.

$$X_{tru} = TA$$

$$Y_{tru} = X_{tru}B + QC$$

Once these true values of X and Y are obtained, noise is added to both the independent variables and the responses. The noise matrices are represented as $E_x$ and $E_y$.

$$X = X_{tru} + E_x$$

$$Y = Y_{tru} + E_y$$

After we obtain the X and Y matrices which represent the observed data, they are split up into a training set and a testing set. Of the 500 observations, the first 250 observations of X and Y are taken as the training set and rest is used as a test set ($X_{tst}$ and $Y_{tst}$). The test set is used for testing the model obtained from the training set to assess its accuracy.

Using the latent variable methods of PCA, PLS, and CCA, we will obtain models from the training set using varying rank values of r. These methods and their implementation will be explained below. Once the models from each of these methods are obtained, a set of Y values, $Y_{pred}$, will be predicted from the $X_{tst}$ variables. The accuracy will be assessed by calculating the differences between $Y_{pred}$ and $Y_{tst}$ as well as comparing $Y_{pred}$ to the true values $Y_{tru}$. Also, the training set accuracy will be calculated by using the model on the training data to obtain $Y_{train}$ and comparing it to Y and $Y_{tru}$. In all of these accuracy calculations, the relative error was obtained using the Frobenius norm. The equations for calculating all the testing and training errors are shown below.

$$Etest = \frac{\|Y_{pred} - Y_{tst}\|_F^2}{\|Y_{tst}\|_F^2}$$

$$Etest_{tru} = \frac{\|Y_{pred} - Y_{tru}\|_F^2}{\|Y_{tru}\|_F^2}$$

$$Etrain = \frac{\|Y_{train} - Y_{tst}\|_F^2}{\|Y_{tst}\|_F^2}$$

$$Etrain_{tru} = \frac{\|Y_{train} - Y_{tru}\|_F^2}{\|Y_{tru}\|_F^2}$$

## Principal Component Analysis (PCA)

Principal component analysis (PCA) is a method that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. PCA can be used as a tool for making predictive models or for dimensionality reduction. Dimensionality reduction is the process of reducing the number of random variables by obtaining a set of principal variables. The number of principal components must be less than or equal to the number of original variables. The first principal component has the largest variance and accounts for as much of the variability in the data as possible. Each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. This results in an uncorrelated orthogonal basis set.

The principal component solution is identical the pseudoinverse approach. Therefore we can use the singular value decomposition (SVD) of X to obtain the rank-r pseudoinverse of X.

$$X = U\Sigma V'$$

$$X_r = U_r\Sigma_r V_r'$$

Each term consists of only the first r components corresponding to those of X. The smallest singular values are removed in the rank-r approximation of X, $X_r$.

Next, we must find the linear relationship between X and Y, but using $X_r$ as an approximation to X. This removes the noise in X associated with the directions of smallest variation. We assume that $X_r$ and Y are related through a linear regression model.

$$Y = X_r A + E_y$$

$$A = {X_r}^{\dagger} Y$$

Therefore, the matrix A can be obtained using the pseudoinverse of the rank-r approximation of X. We can now predict a set of responses to a new set of input variables to test the accuracy of the model.

$$Y_{pred} = X_{tst} A$$

When obtaining a least squares solution, the matrix A may be full rank but poorly conditioned. These small singular values of A lead to large variances of the least squares solution. Also, small perturbations in A due to noise can result in large relative changes in the least squares solution, which suggests that ordinary least squares will be unstable if A is poorly conditioned. Therefore, using the principal component solution allows for the removal of the small singular values of A and thus reduces the variance of the solution, but it introduces bias. The principal component method is a trade-off between reduced variance and increased bias. Since the overall error is a combination of both variance and bias, the goal is to choose the rank r parameter which minimizes this error.

Note that in the implementation of PCA we didn't scale or mean center the data because of the nature of the input (i.e. X and Y were obtained using random matrices which are already scaled and mean centered). Once the model was obtained, the relative errors were calculated as previously discussed.

## Partial Least Squares (PLS)

Partial least squares (PLS) is also referred to as "projections onto latent structures" and is similar to PCA is some ways. PLS is used to find the fundamental relations between X and Y (i.e. a latent variable approach to modelling the covariance structures in these two spaces). A PLS model will try to find the multidimensional direction in the X space that explains the maximum multidimensional variance direction in the Y space.

The principal component method extracts latent variables which describe directions of major variation in X alone. In contrast, the PLS method extracts latent variables whose directions take into account the major correlations between X and Y, and major variation in both X and Y themselves. Therefore, PLS should be better at prediction than principal components, since it takes into account the relationship between X and Y.

In PLS, the objective function is shown below where *t = Xw.*

$$\max_{\mathrm{w}} w'X'YY'Xw \quad s.t. w'w = 1$$

Rather than using the deflation method, an alternate approach was used in the implementation. This approach uses the SVD of the covariance between X and Y, Y'X. W is obtained by using the right singular vectors of the covariance, however, only the first r vectors were used to obtain the matrix. Next, the model can be obtained using the following equations.

$$T = XW$$

$$C = (T'T)^{-1}T'Y$$

Then, the predicted values of Y can be obtained from the tests set $X_{tst}$ to determine the accuracy of the model.

$$T_{tst} = X_{tst}W$$

$$Y_{pred} = T_{tst}C$$

The relative errors of the test set and training set were then calculated using the equations previously described.

## Canonical Correlation Analysis (CCA)

Canonical Correlation Analysis (CCA) is the final latent variable method that will be examined. It also analyzes the covariance matrix between X and Y. If there are correlations among the X and Y variables, CCA will find a linear combination of X and Y which maximize the correlation with one another.

The major difference between CCA and the PLS approaches is that, unlike PLS, the CCA approach considers only the correlations between X and Y, whereas PLS takes into account both correlation and variance of the X and Y variables. The influence of variance in the CCA latent variables is supressed by the fact that X and Y are orthonormalized during the process through multiplication with the inverse square root factors.

In the implementation of CCA, we must define a matrix P of correlation coefficients.

$$P = S'_{xx}X'Y\,S_{yy}$$

$$S_{xx} = (X'X)^{-\frac{1}{2}} \quad S_{yy} = (Y'Y)^{-\frac{1}{2}}$$

Note that in the implementation, the Sxx and Syy matrices can be defined as the inverse of the cholesky factor of X'X and Y'Y, respectively. Both these definitions can be shown to provide the same solution. The problem can then be defined by the following objective function.

$$\max_{\mathrm{w,z}} w'P\,z \quad s.t.\ w'w = 1\,, z'z = 1$$

The joint soutions for w and z are $U_r$ and $V_r$, which are the first r columns of the left and right singular values of P, respectively.

We can now determine the CCA latent variables T and S which are a transformed version of X and Y, respectively.

$$T = X\,L = X\,(X'X)^{-\frac{T}{2}}\,U_r$$

$$S = Y\,M = Y\,(Y'Y)^{-\frac{T}{2}}\,V_r$$

In order to use CCA to solve the problem, we form rank-r approximations $X_r$ and $Y_r$ to X and Y, respectively.

$$X_r = TA$$

$$Y_r = SB$$

The coefficient matrices A and B are solved through standard least-squares regression. Once A and B are obtained, we can assume that $X_r$ and $Y_r$ are linearly related.

$$Y_r = X_r C + E$$

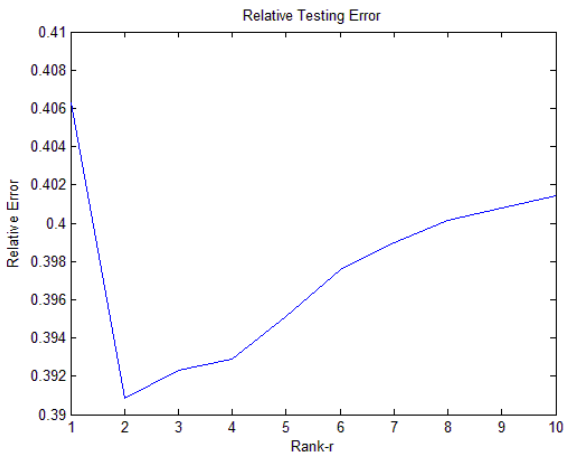
We can solve for C using an additional least-squares regression procedure, however since Xr is rank deficient, we must use the pseudoinverse technique to determine C. Once C is obtained, we can predict a new set of responses from the test input to assess the accuracy of the model.

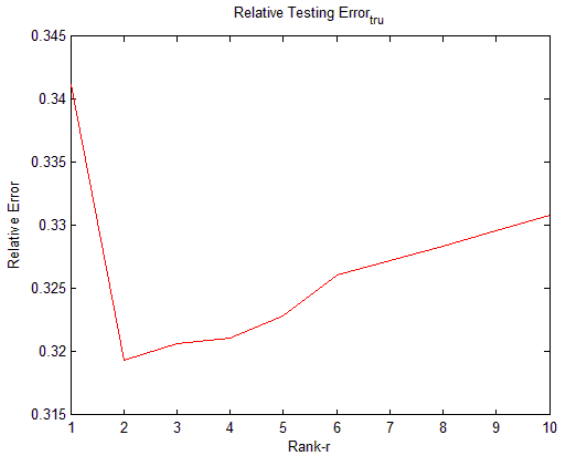
$$Y_{pred} = X_{tst} C$$

The relative errors are computed similarly to the methods previously described and is also computed for varying rank-r values.

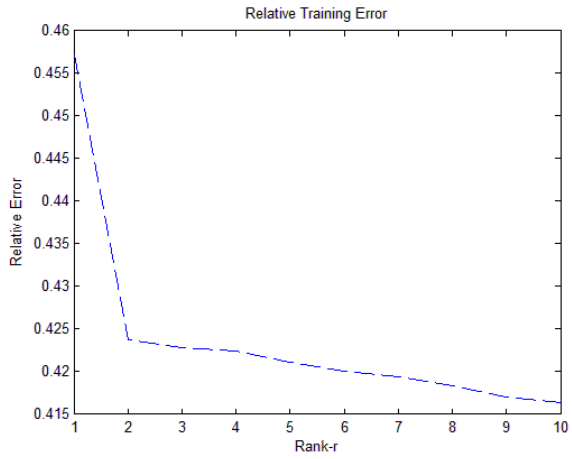
## Results

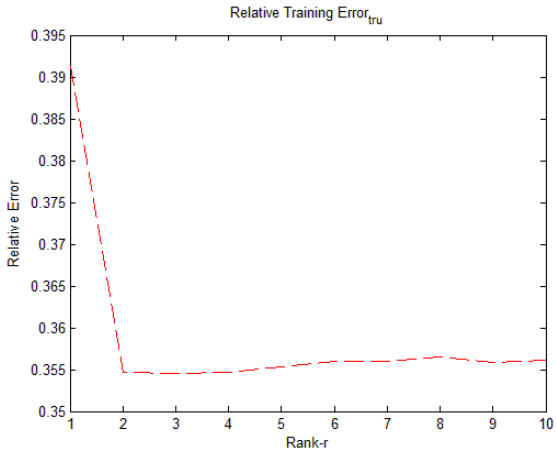
The results below show the accuracy of each of the methods in terms of relative error with respect to the rank r. The following plots show the results of PCA when comparing the predicted value of the test set $Y_{pred}$ to the testing value $Y_{tst}$ and the true value $Y_{tru}$.



The following plots show the results of PCA when comparing the predicted value of the training set $Y_{train}$ to the training value Y and the true value $Y_{tru}$.

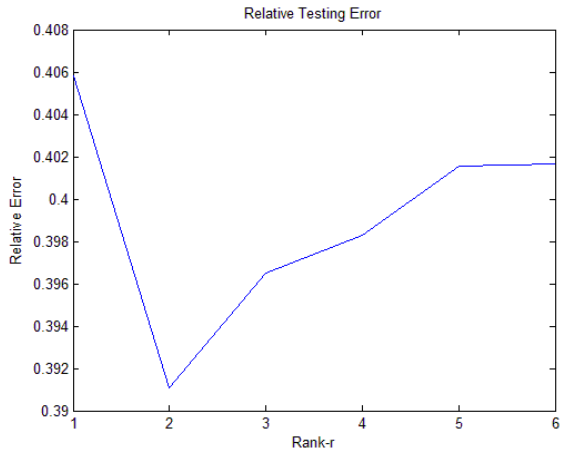


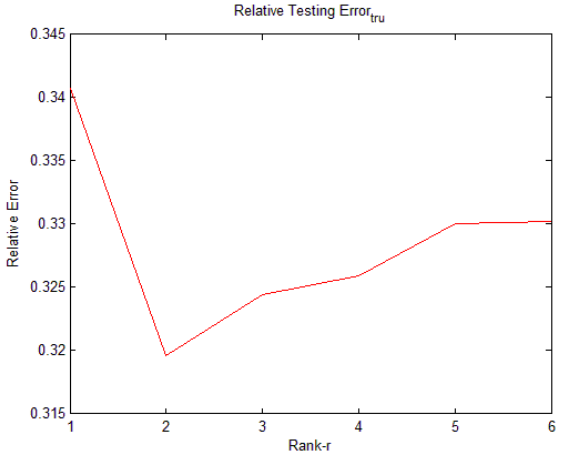




Note that the case where the rank r = 10 is equivalent to the ordinary least squares (OLS) solution. The following results show the values of the OLS method.

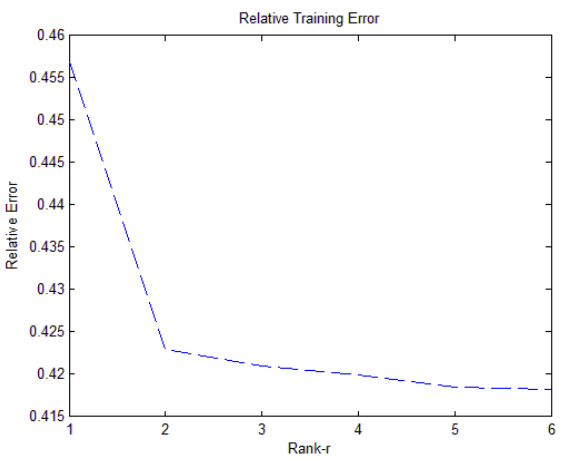
```
EOLStest =

    0.4015


EOLStest_tru =

    0.3309


EOLStrain =

    0.4164


EOLStrain_tru =

    0.3561
```

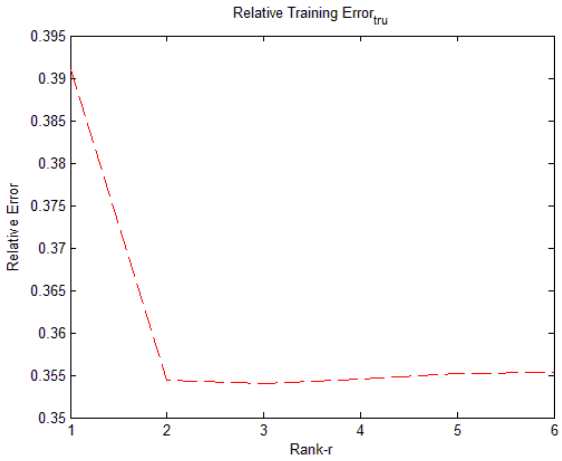
The following plots show the results of the PLS method on test set.



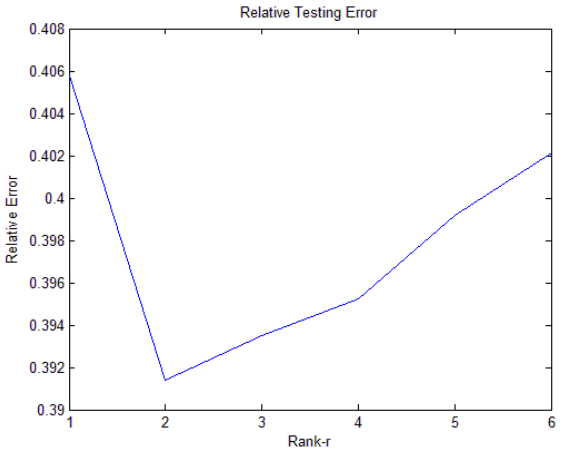



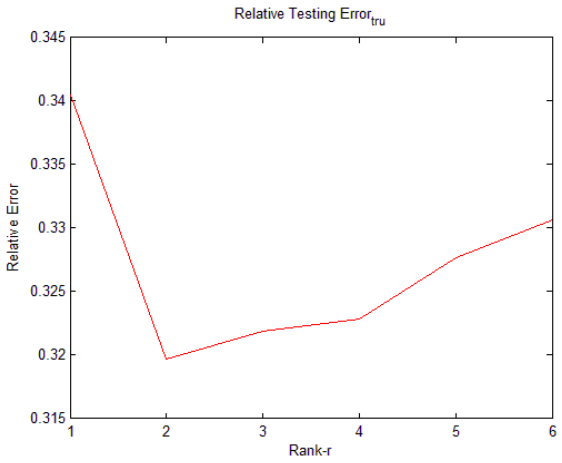
The following plots show the results of the PLS method on the training set.



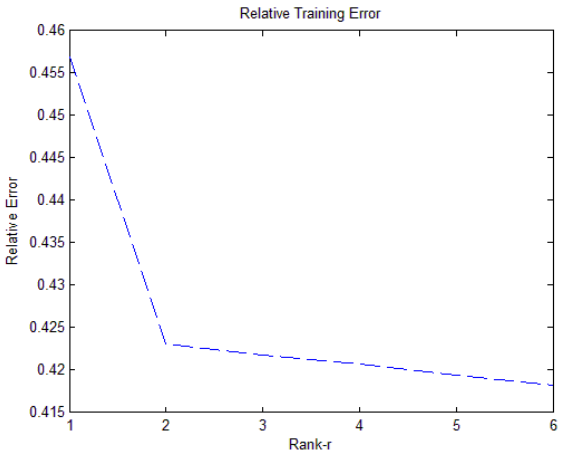



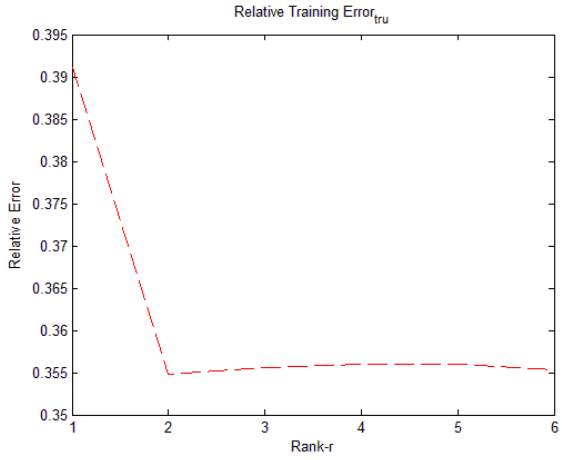

The following plots show the results of the CCA method on the test set.



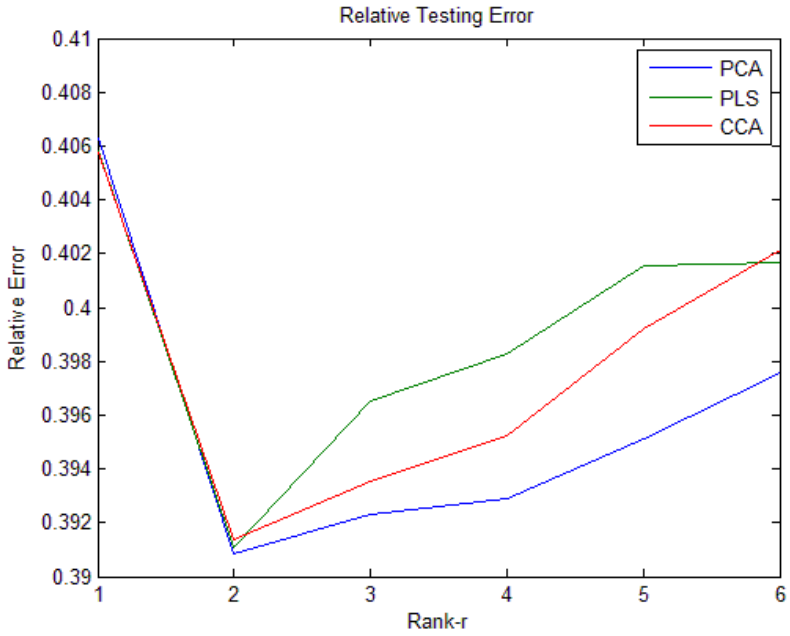

The following plots show the results of the CCA method on the training set.



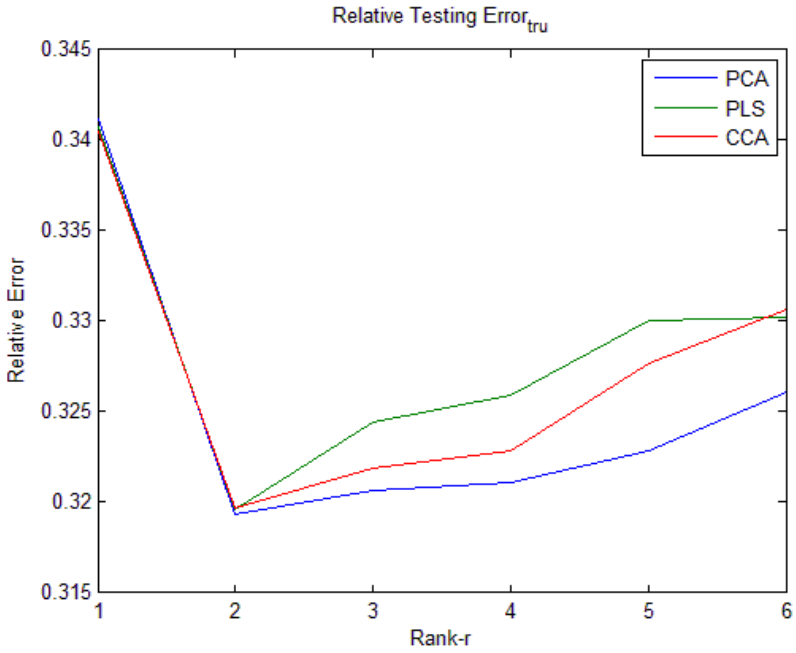



After evaluating all of the methods, they were compared in the following plots. Note that because the rank values are restricted to 6 for the PLS and CCA methods, only the first 6 rank values of the PCA method was plotted.

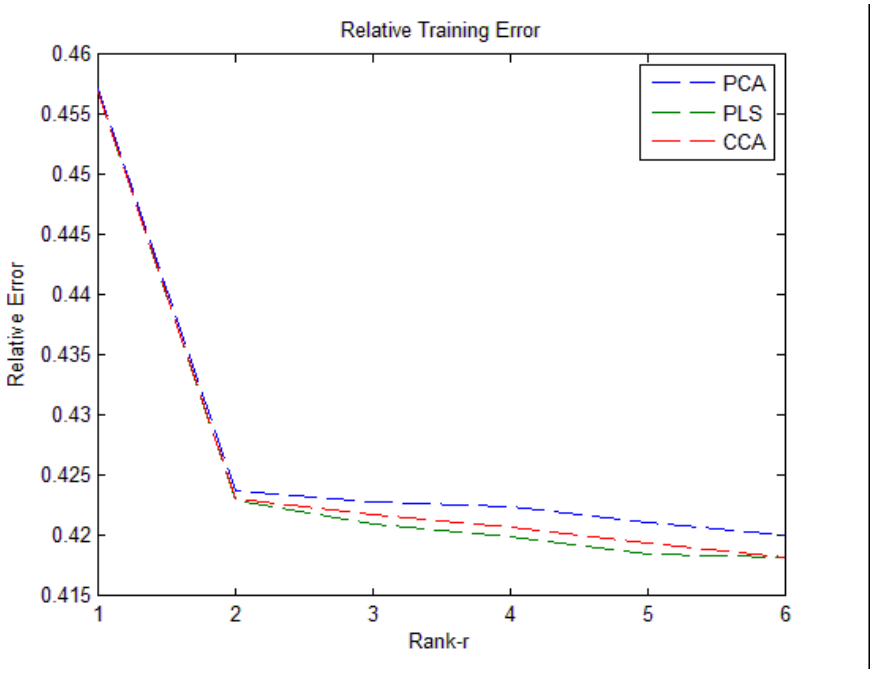
The following plot shows the relative testing error for all the methods.



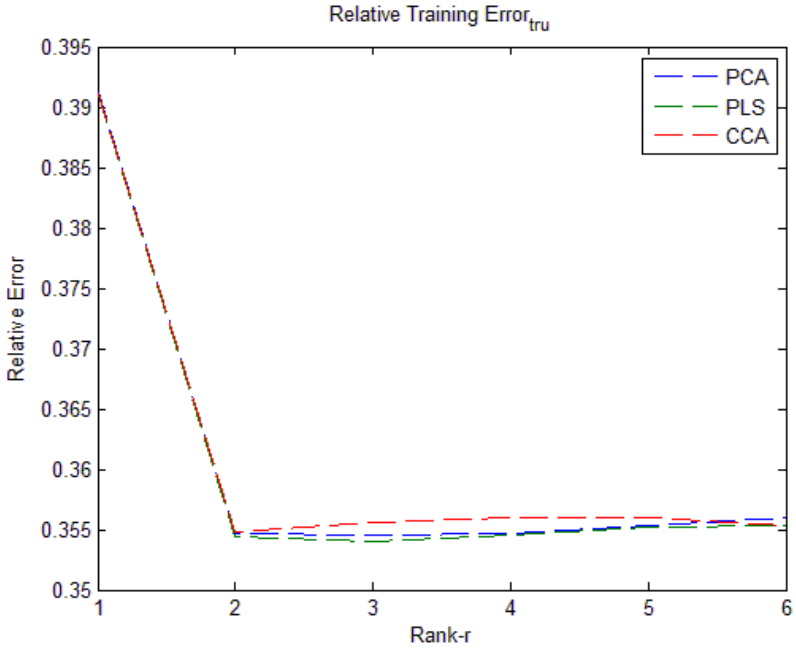

The following plot shows the relative testing error as compared to the true values for all the methods.



The following plot shows the relative training error for all the methods.




The following plot shows the relative training error as compared to the true values for all the methods.



## Discussion

This section will discuss the results previously shown and analyse some of the emerging trends as well as examine the effectiveness of the methods.

For each of the methods, the training error was slightly larger than the testing error. This is unexpected but may be due to the nature of the input (i.e. since it was comprised of random matrices) and it may be due to the testing procedure (i.e. using only 50% of the data to train).

For each of the methods, the relative error when compared to the true values was lower than when compared to the noisy data. That is, when comparing training error or the test error to the true values of Y, it provided lower error than when it was compared to the noisy version of Y. This means that all of these methods have proven to be effective at capturing the patterns in the original noise free data rather than modelling the noise in the data.

For all of the methods, the test error was at a minimum when rank r = 2. This is the expected result since the original X data was of rank 2, as explained. Thus, using these latent variable methods to reduce the dimensionality should provide the best results when the chosen rank is also 2. Although the error may not have been at a minimum at rank r = 2 for the training set, it still shows a large decrease from the rank r = 1 case which cannot capture the information in X, as expected.

When comparing the performance of PCA to OLS, it is clear that PCA is able to outperform OLS especially in the rank r = 2 case. All of the latent variable methods have shown an improvement from the ordinary least squares solution. These methods are able to capture the underlying trends in the data since only the principal components are used and can significantly reduce the effect of the noise in the data.

When comparing the performance of PCA, PLS, and CCA, the main measure of performance would be testing the model on previously unseen data. The relative error in the test set shows that all the methods produced very similar values at rank r = 2 when the error was at a minimum. When comparing these predicted outputs to the test data, it shows that PCA is able to slightly outperform PLS and CCA. The same statement holds when comparing the predicted outputs to the true responses. However, when comparing the methods on the training set performance, PLS seems to slightly outperform the other methods at the minimum error when rank r = 2.

It is interesting to note that PLS and CCA did not seem to significantly outperform the PCA method. PLS and CCA take into consideration the covariance between X and Y, while PCA only considers the variance in X. It would be expected that these methods provide good results when rank r = 4, due to the true Y output being of rank 4, however this was not found to be the case.

In conclusion, these latent variable methods all proved to be better than the ordinary least squares solution to the problem. All of these methods provided better accuracy of the models and were able to perform similarly well in the training and testing errors when the rank was equivalent to the original rank of X.

# Appendix

```
%%%% ECE 712
% Final Project
% Sara Jamil
clear; close all; clc;

load('project.mat');

%% OLS - Ordinary Least Squares

Aols = X\Y;

% predicting with Xtst
Ypred = Xtst*Aols;

% test set error
EOLStest = (norm(Ypred-Ytst,'fro')^2)/(norm(Ytst,'fro')^2);
EOLStest_tru = (norm(Ypred-Ytru(251:500,:),'fro')^2)/(norm(Ytru(251:500,:),'fro')^2);

% training set error
Ytrain = X*Aols;
EOLStrain = (norm(Ytrain-Y,'fro')^2)/(norm(Y,'fro')^2);
EOLStrain_tru = (norm(Ytrain-Ytru(1:250,:),'fro')^2)/(norm(Ytru(1:250,:),'fro')^2);


%% PCA - Principal Component Analysis

for r = 1:10
    Xr = pseudoinv(X,r);
    Apca = Xr*Y;

    % predicting with Xtst
    Ypred = Xtst*Apca;

    % test set error
    EPCAtest(r,:) = (norm(Ypred-Ytst,'fro')^2)/(norm(Ytst,'fro')^2);
    EPCAtest_tru(r,:) = (norm(Ypred-Ytru(251:500,:),'fro')^2)/(norm(Ytru(251:500,:),'fro')^2);

    % training set error
    Ytrain = X*Apca;

    EPCAtrain(r,:) = (norm(Ytrain-Y,'fro')^2)/(norm(Y,'fro')^2);
    EPCAtrain_tru(r,:) = (norm(Ytrain-Ytru(1:250,:),'fro')^2)/(norm(Ytru(1:250,:),'fro')^2);
end

figure(1)
plot(EPCAtest);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error');

figure(2)
plot(EPCAtest_tru,'r');
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error_t_r_u');

figure(3)
plot(EPCAtrain,'--');
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error');

figure(4)
plot(EPCAtrain_tru,'r--');
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error_t_r_u');


%% PLS - Partial Least Squares

for r = 1:6
```

```
    [U,S,V] = svd(Y'*X,'econ');
    s = diag(S);
    W = V(:,1:r);

    T = X*W;
    C = inv(T'*T)*T'*Y;

    % predicting with Xtst
    Ttst = Xtst*W;
    Ypred = Ttst*C;

    % test set error
    EPLStest(r,:) = (norm(Ypred-Ytst,'fro')^2)/(norm(Ytst,'fro')^2);
    EPLStest_tru(r,:) = (norm(Ypred-Ytru(251:500,:),'fro')^2)/(norm(Ytru(251:500,:),'fro')^2);

    % training set error
    Ytrain = T*C;

    EPLStrain(r,:) = (norm(Ytrain-Y,'fro')^2)/(norm(Y,'fro')^2);
    EPLStrain_tru(r,:) = (norm(Ytrain-Ytru(1:250,:),'fro')^2)/(norm(Ytru(1:250,:),'fro')^2);
end

figure(5)
plot(EPLStest);
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error');

figure(6)
plot(EPLStest_tru,'r');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error_t_r_u');

figure(7)
plot(EPLStrain,'--');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error');

figure(8)
plot(EPLStrain_tru,'r--');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error_t_r_u');


%% CCA - Canonical Correlation Analysis

for r = 1:6
%     % This method for finding T and S
%     % provides the same solution as below
%     Sxx = (X'*X)^(-.5);
%     Syy = (Y'*Y)^(-.5);
%     Pxy = Sxx'*X'*Y*Syy;
%     [U,E,V] = svd(Pxy,'econ');
%     U = U(:,1:r);
%     V = V(:,1:r);
%     T = X*Sxx*U;
%     S = Y*Syy*V;

    chx = inv(chol(X'*X));
    chy = inv(chol(Y'*Y));

    P = chx'*X'*Y*chy;

    [U,~,V] = svd(P,'econ');
    U = U(:,1:r);
    V = V(:,1:r);

    T = X*chx*U;
```

```
    S = Y*chy*V;

    A = T\X;
    B = S\Y;

    Xr = T*A;
    Yr = S*B;

    C = pseudoinv(Xr,r)*Yr;

    % predicting with Xtst
    Ypred = Xtst*C;

    % test set error
    ECCAtest(r,:) = (norm(Ypred-Ytst,'fro')^2)/(norm(Ytst,'fro')^2);
    ECCAtest_tru(r,:) = (norm(Ypred-Ytru(251:500,:),'fro')^2)/(norm(Ytru(251:500,:),'fro')^2);

    % training set error
    Ytrain = X*C;

    ECCAtrain(r,:) = (norm(Ytrain-Y,'fro')^2)/(norm(Y,'fro')^2);
    ECCAtrain_tru(r,:) = (norm(Ytrain-Ytru(1:250,:),'fro')^2)/(norm(Ytru(1:250,:),'fro')^2);
end

figure(9)
plot(ECCAtest);
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error');

figure(10)
plot(ECCAtest_tru,'r');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error_t_r_u');

figure(11)
plot(ECCAtrain,'--');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error');

figure(12)
plot(ECCAtrain_tru,'r--');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error_t_r_u');


%% Comparing methods

figure(13)
plot(1:6,EPCAtest(1:6),1:6,EPLStest,1:6,ECCAtest);
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error');
legend('PCA','PLS','CCA');

figure(14)
plot(1:6,EPCAtest_tru(1:6),1:6,EPLStest_tru,1:6,ECCAtest_tru);
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Testing Error_t_r_u');
legend('PCA','PLS','CCA');

figure(15)
plot(1:6,EPCAtrain(1:6),'--',1:6,EPLStrain,'--',1:6,ECCAtrain,'--');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error');
legend('PCA','PLS','CCA');
```

```
figure(16)
plot(1:6,EPCAtrain_tru(1:6),'--',1:6,EPLStrain_tru,'--',1:6,ECCAtrain_tru,'--');
set(gca,'XTick',1:6);
xlabel('Rank-r'); ylabel('Relative Error');
title('Relative Training Error_t_r_u');
legend('PCA','PLS','CCA');
```

```
function Xp = pseudoinv(X,r)
% Xp = pseudoinv(X,r)
% This function calculates the rank-r pseudoinverse of X
% where 1 <= r <= n

[U,S,V] = svd(X,'econ');
s = diag(S);

V = V(:,1:r);
U = U(:,1:r);
s = s(1:r);
s = 1./s(:);
s = diag(s);
% X = bsxfun(@times,V,s.')*U'; %pinv
Xp = V*s*U'; %pseudoinverse of X

end
```