ASSIGNMENT 3

-Meghana PV

-CH12B083

Note: Please find attached the code I've written for this assignment in the zipped folder as well.

Question 1

CODE:


```
clc
clear all
close all

Data=load('Inorfull.mat')
% Data structure consists of the follow attributes:
%           CONC: [130x3 double]
%           DATA: [130x176 double]
%         PureCo: [1x176 double]
%         PureCr: [1x176 double]
%         PureNi: [1x176 double]
%            WAV: [1x176 double]
%        stdDATA: [130x176 double]
%     PureCoCONC: 0.1720
%     PureCrCONC: 0.0764
%     PureNiCONC: 0.1965

PureComp=[Data.PureCo; Data.PureCr; Data.PureNi];
% Final Data: Absorbance spectra (176 pts.) measured for each sample
% (Total: 26 samples). , each consisting of 3 components with their
% concentrations noted.

% Model Data : 25 samples. Validation data: Left out sample. Calculate RMSE
% in predicting this conc.

% Goal is to predict concentrations given absorbance of mixture
```

```
Data =

          CONC: [130x3 double]
          DATA: [130x176 double]
```

```
        PureCo: [1x176 double]
        PureCr: [1x176 double]
        PureNi: [1x176 double]
           WAV: [1x176 double]
        stdDATA: [130x176 double]
    PureCoCONC: 0.1720
    PureCrCONC: 0.0764
    PureNiCONC: 0.1965
```

Picking one replicate measurement per sample- Random measurement

```
NewData=[];
NewStdData=[];
NewConc=[];

rand('seed',95) % So as to provide consistent measurements
for i=1:26
    dt_num = randi(5);
    istart = 5*(i-1)+dt_num;
    NewStdData = [NewStdData; Data.stdDATA(istart,:)];
     NewData = [NewData; Data.DATA(istart,:)];
    NewConc = [NewConc;Data.CONC(istart,:)];
end

X=NewData;
Y=NewConc;
```

Q1.e Average measurement - Uncomment this to solve for average case.

(Do so for 1f also)

```
% NewData=ones(26,176);
% NewStdData=ones(26,176);
% NewConc=ones(26,3);
%
% count=1;
% for i=1:5:130
%
NewConc(count,:)=(Data.CONC(i,:)+Data.CONC(i+1,:)+Data.CONC(i+2,:)+Data.CONC(i+3,:)+Data.CONC
(i+4,:))/5;
%
NewData(count,:)=(Data.DATA(i,:)+Data.DATA(i+1,:)+Data.DATA(i+2,:)+Data.DATA(i+3,:)+Data.DATA
(i+4,:))/5;
%
NewStdData(count,:)=(Data.stdDATA(i,:)+Data.stdDATA(i+1,:)+Data.stdDATA(i+2,:)+Data.stdDATA(i
+3,:)+Data.stdDATA(i+4,:))/5;
%     count=count+1;
% end

% X=NewData;

%Y=NewConc;
```

Q1. f: With end wavelengths removed- Uncomment and get results for Q. b and c

```
% X=X(:,26:151);
```

Q1a. OLS

```
NewData = X-(ones(26,1)*mean(X));
NewConc = Y - (ones(26,1)*mean(Y));

[nsamples , nvar] = size(NewData);
sumsqrerr = zeros(3,1);
RMSE=0;
for k=1:26
 ModelConc =[NewConc(1:k-1,:);NewConc(k+1:nsamples,:)];
    ModelData=[NewData(1:k-1,:);NewData(k+1:nsamples,:)];
     ModelStdData=[NewData(1:k-1,:);NewData(k+1:nsamples,:)];

     ValidateConc=NewConc(k,:);
    ValidateData=NewData(k,:);
     ValidateStdData=NewStdData(k,:);

 ModelX=ModelData;
 ModelY=ModelConc;
 %OLS - assuming errors in absorbance(ModelX)- Inverse OLS
 alphainv=pinv(ModelY'*ModelX)*ModelY'*ModelY;

PredictConc=ValidateData*alphainv;
err=0;
for i = 1:3
error = ValidateConc(i)-PredictConc(i);
sumsqrerr(i) = sumsqrerr(i) + error*error;
end

end

RMSE_1a = sqrt(sumsqrerr/nsamples)
RMSETotal_1a = sqrt(sum(sumsqrerr)/(nsamples*3))
```

```
RMSE_1a =

    0.0022
    0.0093
    0.0056


RMSETotal_1a =

    0.0064
```

Q1.d: Q1b and 1c with different no. of PCs - Change here from 1 to 6

```
nPC=3;
```

Q1.b: PCR with error free scores

```
NewData = X-(ones(26,1)*mean(X));
NewConc = Y - (ones(26,1)*mean(Y));
%Denoising the data using PCA
[DenoisedModelData,ScoresMatrix,SingularValues] = PCA(NewData',nPC);

NewData=ScoresMatrix;

[nsamples , nvar] = size(NewData);
sumsqrerr = zeros(3,1);
RMSE=0;
for k=1:26
 ModelConc =[NewConc(1:k-1,:);NewConc(k+1:nsamples,:)];
    ModelData=[NewData(1:k-1,:);NewData(k+1:nsamples,:)];
     ModelStdData=[NewData(1:k-1,:);NewData(k+1:nsamples,:)];

     ValidateConc=NewConc(k,:);
    ValidateData=NewData(k,:);
     ValidateStdData=NewStdData(k,:);

 ModelX=ModelData;
 ModelY=ModelConc;
 %OLS - assuming errors in concentrations(ModelY)-  OLS
 alphainv=pinv(ModelX'*ModelX)*ModelY'*ModelX;

PredictConc=ValidateData*alphainv;
err=0;
for i = 1:3
error = ValidateConc(i)-PredictConc(i);
sumsqrerr(i) = sumsqrerr(i) + error*error;
end

end

RMSE_1b = sqrt(sumsqrerr/nsamples)
RMSETotal_1b = sqrt(sum(sumsqrerr)/(nsamples*3))
```

```
RMSE_1b =

    0.0033

    0.0126

    0.0096


RMSETotal_1b =

     0.0094
```

Q1.c: PCR with error free concentrations

```
NewData = X-(ones(26,1)*mean(X));
NewConc = Y - (ones(26,1)*mean(Y));
```

```
%Denoising the data using PCA
[DenoisedModelData,ScoresMatrix,SingularValues] = PCA(NewData',nPC);

NewData=ScoresMatrix;

[nsamples , nvar] = size(NewData);
sumsqrerr = zeros(3,1);
RMSE=0;
for k=1:26
 ModelConc =[NewConc(1:k-1,:);NewConc(k+1:nsamples,:)];
    ModelData=[NewData(1:k-1,:);NewData(k+1:nsamples,:)];
     ModelStdData=[NewData(1:k-1,:);NewData(k+1:nsamples,:)];

     ValidateConc=NewConc(k,:);
    ValidateData=NewData(k,:);
     ValidateStdData=NewStdData(k,:);

 ModelX=ModelData;
 ModelY=ModelConc;
 %OLS - assuming errors in scores(ModelX) i.e absorbance- Inverse OLS
 alphainv=pinv(ModelY'*ModelX)*ModelY'*ModelY;

PredictConc=ValidateData*alphainv;
err=0;
for i = 1:3
error = ValidateConc(i)-PredictConc(i);
sumsqrerr(i) = sumsqrerr(i) + error*error;
end

end

RMSE_1c = sqrt(sumsqrerr/nsamples)
RMSETotal_1c = sqrt(sum(sumsqrerr)/(nsamples*3))
```

```
RMSE_1c =

    0.0904
    0.3290
    0.2499


RMSETotal_1c =

    0.2442
```

Published with MATLAB® R2014a

RESULTS:

a. OLS model with errors in absorbance. **RMSE values– Random case**

Co	Cr	Ni	Net RMSE
0.0022	0.0093	0.0056	0.0064

b. Results in d.

c. Results in d.

d. As there are 3 pure components and the absorbance spectra of the mixture is a linear combination of the pure component spectra for these 3, the true no. of principal components is 3.

PCR and OLS model with errors in concentrations:

No. of PCs	Co	Cr	Ni	Net RMSE
1	0.0036	0.0200	0.0117	0.0136
2	0.0032	0.0131	0.0090	0.0093
3	0.0033	0.0126	0.0096	0.0094
4	0.0034	0.0132	0.0099	0.0097
5	0.0034	0.0137	0.0079	0.0093
6	0.0041	0.0160	0.0103	**0.0084**

PCR and OLS model with errors in absorbance:

No. of PCs	Co	Cr	Ni	Net RMSE
1	0.0053	0.0308	0.0121	0.0193
2	0.0043	0.0175	0.0128	0.0128
3	0.0904	0.3290	0.2499	0.2442
4	0.0205	0.0756	0.0600	0.0570
5	0.0062	0.0237	0.0145	0.0164
6	0.0041	0.0160	0.0103	**0.0112**

From the RMSEtotal values for both the case, we can see that least errors are there when no. of principal components = 6, although it is supposed to be 3. This is because the data is not scaled, and different variables have different variances.

e. **RMSE values– Average case**

PCR and OLS model with errors in concentrations:

No. of PCs	Co	Cr	Ni	Net RMSE
1	0.0035	0.0204	0.0116	0.0137
2	0.0032	0.0127	0.0079	0.0089

3	0.0005	0.0021	0.0025	0.0019
4	0.0005	0.0022	0.0019	0.0017
5	0.0005	0.0022	0.0019	0.0017
6	0.0005	0.0020	0.0016	**0.0015**

The RMSE values are much lower than before when compared to the random replicate case, as the effect of erroneous measurements is reduced. Although, the no. of PCs is 6 as before.

PCR and OLS model with errors in absorbance:

No. of PCs	Co	Cr	Ni	Net RMSE
1	0.0053	0.0320	0.0120	0.0200
2	0.0046	0.0182	0.0114	0.0127
3	0.0005	0.0021	0.0025	**0.0019**
4	0.0005	0.0023	0.0023	0.0019
5	0.0005	0.0024	0.0023	0.0019
6	0.0005	0.0023	0.0022	0.0019

The RMSE values are much lower than before when compared to the random replicate case, as the effect of erroneous measurements is reduced. This model is also more accurate and predicts the no. of PCs correctly as 3 !

f. Eliminating the end noisy variables: RMSE values for average case:

PCR and OLS model with errors in concentrations:

No. of PCs	Co	Cr	Ni	Net RMSE
1	0.0036	0.0197	0.0117	0.0134
2	0.0031	0.0114	0.0084	0.0084
3	1.0e-03 * 0.0866	1.0e-03 * 0.4194	1.0e-03 * 0.3059	**3.0386e-04**
4	1.0e-03 * 0.0901	1.0e-03 * 0.3846	1.0e-03 * 0.3057	2.8839e-04
5	1.0e-03 * 0.0912	1.0e-03 * 0.3838	1.0e-03 * 0.3071	2.8861e-04
6	1.0e-03 * 0.0933	1.0e-03 * 0.2759	1.0e-03 * 0.2900	2.3726e-04

PCR and OLS model with errors in absorbance:

No. of PCs	Co	Cr	Ni	Net RMSE
1	0.0053	0.0299	0.0120	0.0189
2	0.0044	0.0163	0.0120	0.0119
3	1.0e-03 * 0.0866	1.0e-03 * 0.4194	1.0e-03 * 0.3061	**3.0393e-04**
4	1.0e-03 * 0.0872	1.0e-03 * 0.4186	1.0e-03 * 0.3051	3.0326e-04
5	1.0e-03 * 0.0874	1.0e-03 * 0.4188	1.0e-03 * 0.3050	3.0336e-04
6	1.0e-03 * 0.0874	1.0e-03 * 0.4185	1.0e-03 * 0.3052	3.0328e-04

We can see that the model is very accurate as the RMSE errors are very small for no. of PCs = 3 and above. This is because we have removed the end wavelengths, which have high errors and thus very different variances. Now the unscaled data has only minor changes in variances between the variables and hence is good enough to predict the calibration model accurately.

Question 2

CODE:


```
clc
clear all
close all

F1=load('flowdata1.mat')
F2=load('flowdata2.mat')

[N,n]=size(F1.Fmeas); % N - no. of samples, n- no. of variables
X=(F1.Fmeas)';
Atrue=F1.Atrue;
STD=F1.std; % Standard deviation of errors in variables
```

```
F1 =

    Atrue: [3x5 double]
      std: [5x1 double]
    Fmeas: [1000x5 double]
    Ftrue: [1000x5 double]


F2 =

    Atrue: [3x5 double]
      std: [5x1 double]
    Fmeas: [1000x5 double]
    Ftrue: [1000x5 double]
```

Data Set2 - Uncomment and Run to get solutions for 2nd dataset

```
% X=(F2.Fmeas)';
% Atrue=F2.Atrue;
```

Q2a.

True Regression Matrix

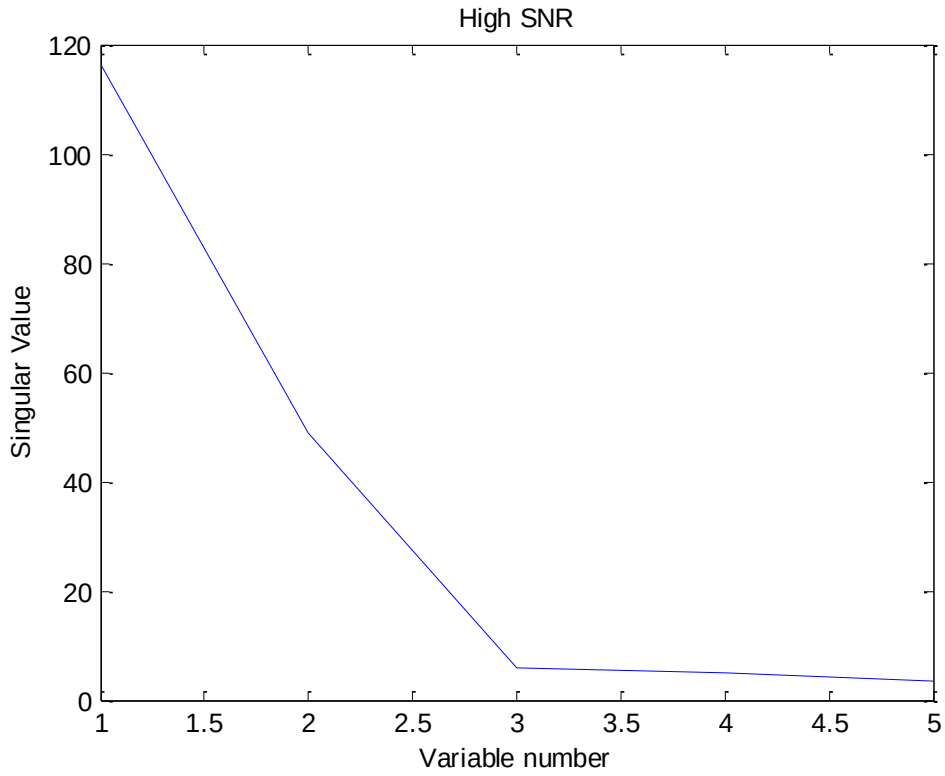
```
Ad=[Atrue(:,3) Atrue(:,4) Atrue(:,5)];
Ai=[Atrue(:,1) Atrue(:,2)];

RegressionMatrix=-inv(Ad)*Ai
```

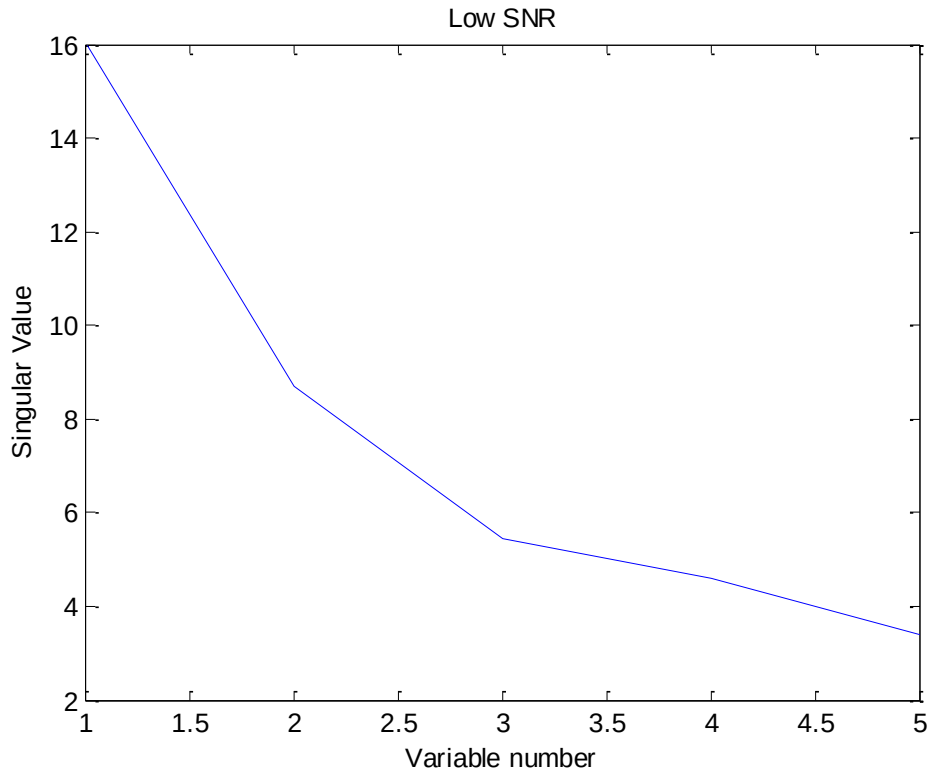
```
% Applying PCA to estimate Regression Matrix
k=2;
avg=mean(X,2);
Xs=X-repmat(avg,1,N);
[U S V]=svd(Xs,'econ');
Ahat=(U(:,k+1:n))';
Adhat=[Ahat(:,3) Ahat(:,4) Ahat(:,5)];
Aihat=[Ahat(:,1) Ahat(:,2) ];

RegressionMatrixEst=-inv(Adhat)*Aihat

RegError=RegressionMatrixEst-RegressionMatrix;
MaxAbsErrorPCA=max(max(abs(RegError)))
```

```
RegressionMatrix =

     1     1
     1     1
     1     0


RegressionMatrixEst =

    0.9975    0.9999
    0.9934    0.9933
    1.0112   -0.0035


MaxAbsErrorPCA =

    0.0112
```

Q2b.

```
figure(1)
plot([1:5],diag(S))
xlabel('Variable number')
ylabel('Singular Value')
```





Q2. c : PCA with Scaling

```
%Cov=diag(STD.*STD)
%L=chol(Cov,'lower')
Linv=inv(diag(STD));
```

```
Xs=Linv*Xs;

[U S V]=svd(Xs,'econ');
Ahat=(U(:,k+1:n))';
Adhat=[Ahat(:,3) Ahat(:,4) Ahat(:,5)];
Aihat=[Ahat(:,1) Ahat(:,2) ];

RegressionMatrixEst=-inv(Adhat)*Aihat

RegError=RegressionMatrixEst-RegressionMatrix;
MaxAbsErrorPCA=max(max(abs(RegError)))
```

```
RegressionMatrixEst =

    0.6647    0.5326
    0.4954    0.3963
    0.5573   -0.0010


MaxAbsErrorPCA =

    0.6037
```

Q2 d: Performing PCA and Estimating different regression matrices with different choice of independent variables

```
ind=[1,4] % Choice of independent variables
dep=[2,3,5];
% True Regression Matrix
Ad=[Atrue(:,dep(1)) Atrue(:,dep(2)) Atrue(:,dep(3))];
Ai=[Atrue(:,ind(1)) Atrue(:,ind(2))];

RegressionMatrix=-inv(Ad)*Ai

% Applying PCA to estimate Regression Matrix
k=2;
avg=mean(X,2);
Xs=X-repmat(avg,1,N);
[U S V]=svd(Xs,'econ');
Ahat=(U(:,k+1:n))';
Adhat=[Ahat(:,dep(1)) Ahat(:,dep(2)) Ahat(:,dep(3))];
Aihat=[Ahat(:,ind(1)) Ahat(:,ind(2)) ];
Ad_determinant=det(Adhat) % If det is close to 0 , matrix is singular, and variables are
dependant

RegressionMatrixEst=-inv(Adhat)*Aihat

RegError=RegressionMatrixEst-RegressionMatrix;
MaxAbsErrorPCA=max(max(abs(RegError)))
```

```
ind =

     1     4
```

```
RegressionMatrix =

    -1     1
     0     1
     1     0

Ad_determinant =

    0.3504


RegressionMatrixEst =

   -1.0002    1.0068
   -0.0026    1.0067
    1.0148   -0.0036


MaxAbsErrorPCA =

    0.0148
```

Published with MATLAB® R2014a

RESULTS:

a. Result for independent flows (1,2) tabulated in 2e.

b. From the graphs, we can see that there is a change in slope at x=3, signifying that there are 2 independent equations (i.e the ones corresponding to the much higher singular values at x=1,2)

c. The error obtained after auto-scaling is more than with the data with scaling (0.6037 for scaled,0.0112 for unscaled case for the independent variables 1,2)

d. Procedure to determine which set of variables is independent / dependant is found by trying all combinations and finding the determinant of the Ad matrix. For dependant variables, this matrix will be close to singular, as the true Ad matrix does not have full rank and is singular and so its inverse does not exist.

   This is the case for the combinations: (1,5) and (3,4) making them dependant, so we cannot use these sets to determine the flows.

   The best possible combination is the one with minimum **error in the regression matrix**.

Combination of independent variables	**Error for High SNR**	**Error for Low SNR**
1,2	0.0112	0.2638
1,3	0.0148	0.4384
1,4	0.0148	0.4453
2,3	0.0173	0.6776

2,4	0.0179	0.6212
2,5	0.0176	0.3963
3,5	0.0170	0.4039
4,5	0.0146	0.3832

Thus we can see that (1,2) is the best combination as it has the least error in both cases.