

**Homework 4 - ENME 691**

Report - Group 3

March 11th, 2024

Brian O’Malley

Ashwin Sudarshan

Sai Dinesh Gelam

# 

**Table of Contents**

1. Introduction
2. Solution Methodology
3. Results
4. Discussion
5. Conclusions
6. Appendix
7. **Introduction**
8. **Background information and description about the problem**

**Rotor-Bearing System:** This system typically consists of a rotating shaft (the rotor) supported by bearings. Bearings are critical components that enable smooth rotation while minimizing friction and wear. Monitoring the health of these bearings is essential for ensuring the reliability and efficiency of rotating machinery.

**Unbalance Defects:** Unbalance occurs when the center of mass of a rotating component (such as the rotor in this case) is not aligned with the axis of rotation. This can lead to vibration and potential damage to the bearings and other components of the system. In the testbed, unbalanced defects were induced by adding screws to the disk of the shaft.

In this case, we have 2 levels of unbalance. Unbalance 1 being the case with 1 screw and Unbalance 2 representing the case with 2 screws added to the disk.

**Vibration Analysis:** Vibration signals provide valuable insights into the condition of rotating machinery. Changes in vibration patterns can indicate various faults or anomalies, including unbalance, misalignment, bearing defects, and more. Accelerometers are commonly used to measure vibration levels in rotating machinery.

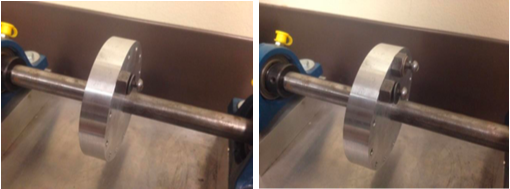
**Data Collection:** Vibration data were collected using an accelerometer mounted on the bearing block. The data were sampled at a high rate of 2560 Hz to capture detailed vibration characteristics. This high sampling rate is necessary to capture the fast-changing vibration signals accurately

1. **Raw data description**

A rotor-bearing testbed (Figure 1) was built to analyze the health condition of the shaft with unbalance defects.The rotating speed of the test-bed is 20 Hz, and screws were added on the disk of the shaft to induce unbalance defects as shown in Figure 2. An accelerometer was mounted on the bearing block and to measure the bearing's vibrations. The vibration data were collected under balanced and unbalanced conditions at a rate of 2560 Hz



**Figure 1**: Healthy rotor-bearing test beds



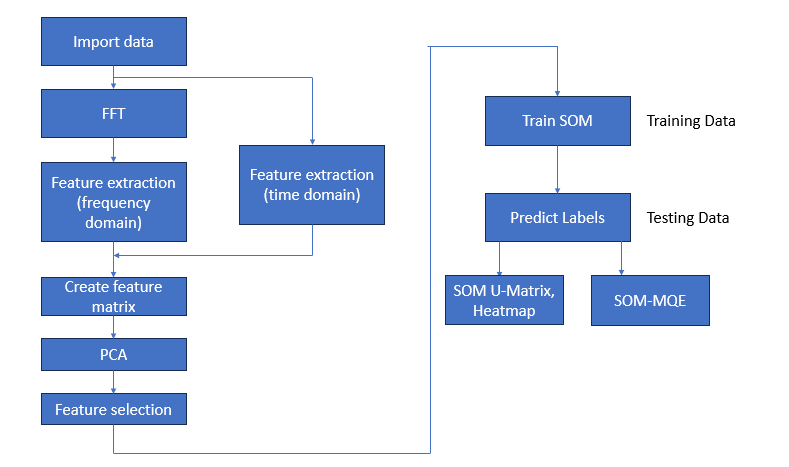
**Figure 2:** Unbalanced level 1(left) and level 2 (right) rotor-bearing test beds.

The resultant data was split into training and testing sets for the machine learning model being developed as a part of this work. Sixty tests were allocated for the training set and thirty for the testing set. The training set was composed of 40 faulty bearing tests (20 each for level 1 and 2) and 20 healthy bearing tests while the proportion of balanced to unbalanced tests in the testing set was unknown.

1. **Solution Methodology**

**a. Flowchart**

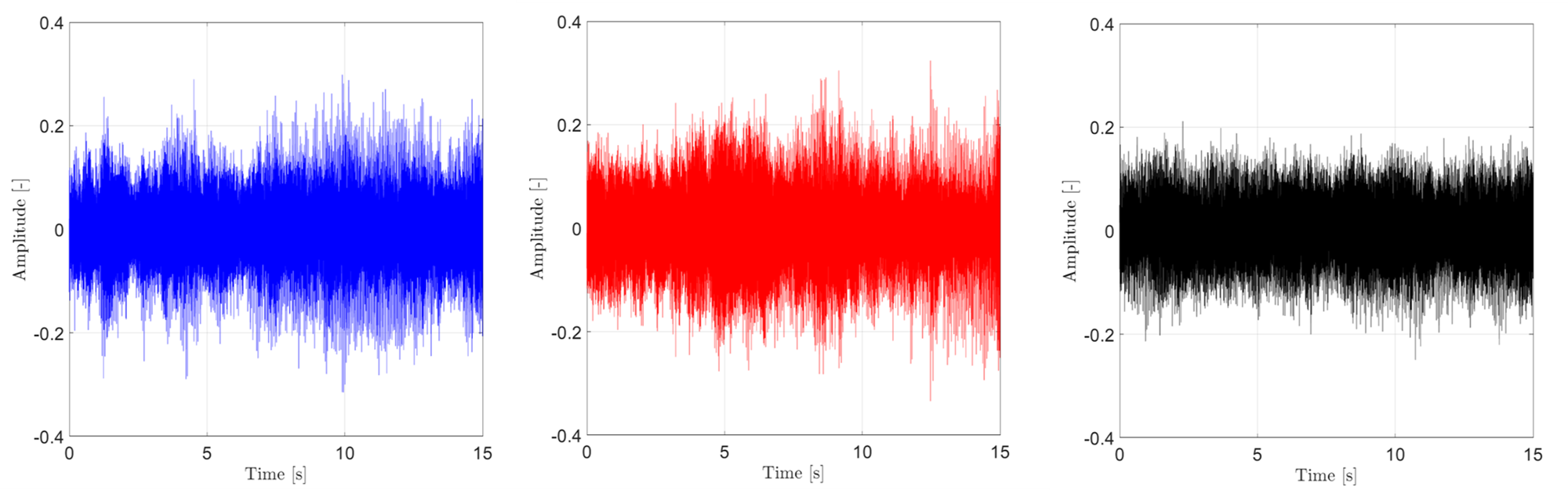
The workflow of the entire process including the data analysis, PCA, feature extraction, and model development can be seen in Figure 3.



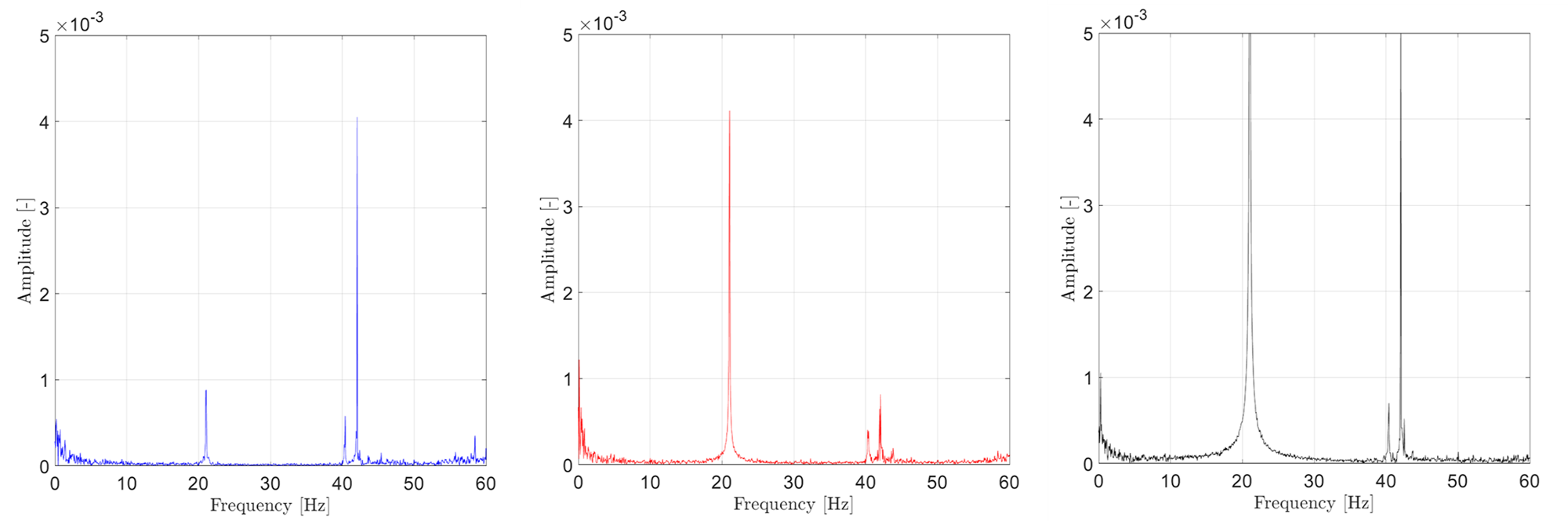
**Figure 3:** Workflow.

**b. Data Analysis**

The solution process begins with loading the vibration data from the healthy, faulty, and testing datasets. Time domain analysis is conducted by plotting healthy and both levels of faulty data sets across the data collection period, but the difference in magnitude is hard to discern as shown in Figures 4. The data is transformed using the fast fourier transform as shown in Figures 5, where the peak amplitudes are visible. The amplitude of the signal is scaled down by ½ the number of points in each signal and results are only shown up to 60 Hz. The difference in amplitude between the healthy and the two unbalanced samples is now clear. The same data analysis process is applied to the test data.



**Figure 4:** Time domain of the healthy (left), unbalanced level 1 (middle), and unbalanced level 2 (right) samples.

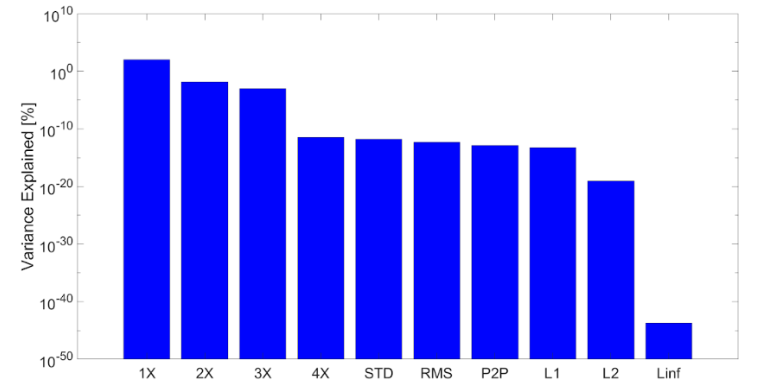


**Figure 5**: Frequency domain of the healthy (left), unbalanced level 1 (middle), and unbalanced level 2 (right) samples.

**c. Principal Component Analysis**

The most relevant sensor measurements were selected via Principal

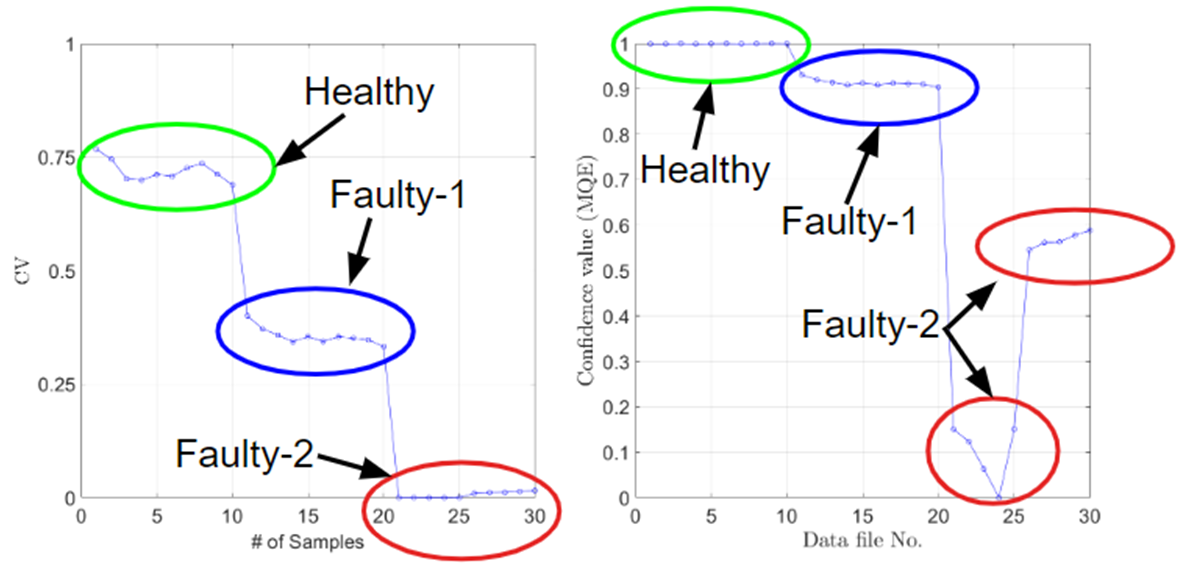
Components Analysis (PCA). From Figure 6, it can be observed that the variables which are of most significance are the 1X Harmonic and 2X Harmonic, closely followed by 3X Harmonic.We find from the figure that Rms and Std hardly have much significance. Hence, the two features we selected were 1X harmonic and 2X harmonic.

****

**Figure 6 :** Principal component analysis

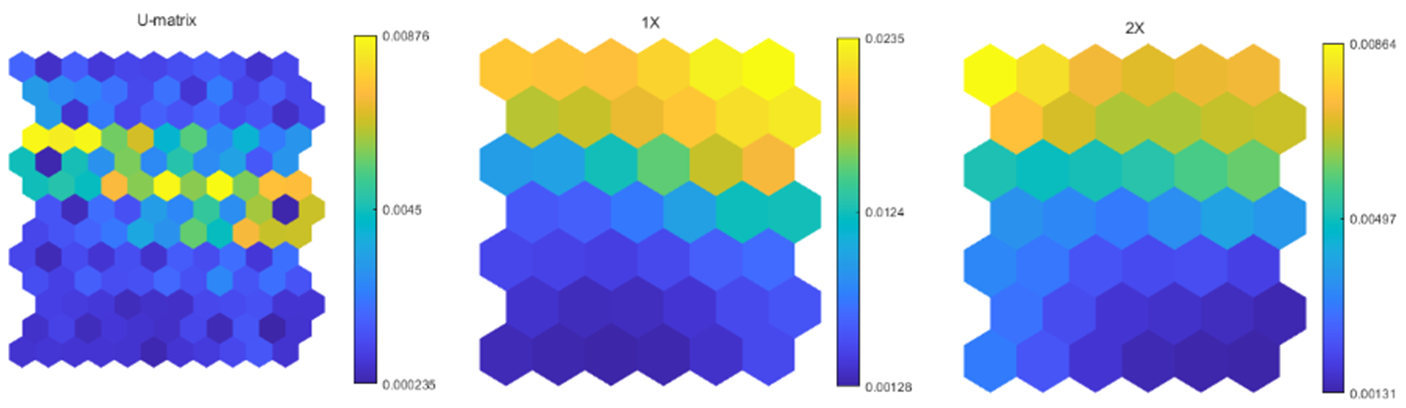
1. **Results and Discussion**

Figure 7 compares the results from Logistic Regression (LR )against those of the SOM-MQE using the 1X harmonic only for training. In both cases, distinct regions appear in the plots which correspond to the labels provided for the testing set. Looking at the left hand side of the figure it becomes clear that LR predicts a value of ~0.75 for healthy, ~0.375 for unbalanced 1, and ~0.0 for unbalanced 2. The results for SOM-MQE are somewhat less clear. The difference in predicted confidence value (CV) for healthy (CV =1.0) and unbalanced 1 (CV = 0.9) is only 0.1, while the CV for unbalanced 2 ranges from 0.0 to 0.6. While we know the correct labels for each sample in this testing set, the significant variation in the prediction for unbalanced 2 should caution its use.



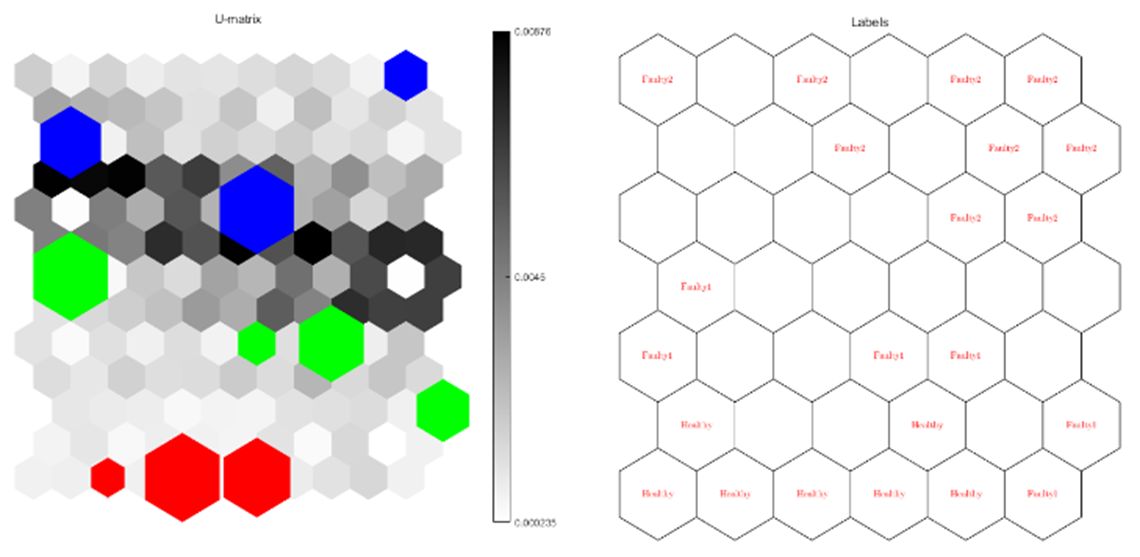
**Figure 7:** Results for logistic regression (left) and SOM-MQE (right) using the 1X harmonic only.

The SOM was trained on both the 1X and 2X harmonic and the corresponding U-matrices can be seen in Figures 8 and 9. The maximum and minimum distances for each harmonic are located in different corners of the map for each harmonic and provide a broader coverage than either would on their own. For example, training the SOM with only the 1X harmonic resulted in only 20/30 testing samples being correctly labeled, but when the 2X harmonic is included this increases to 25/30. The complete U-matrix (on the left hand side of the figure) shows the maximum distance values are located horizontally along the upper portion of the matrix.



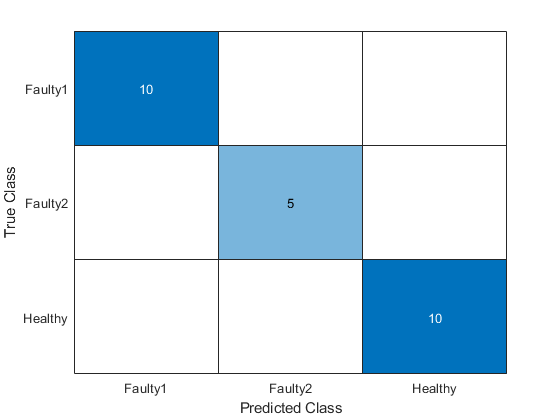
**Figure 8**: SOM U-Matrix Full (left), 1X harmonic (center), and 2X harmonic (right)

The actual labels of the testing set can be seen in Figure 9 along with the designation of the testing samples indicated by various colors (red for healthy, green for unbalanced 1, and blue for unbalanced 2). Here it's clear that when the SOM classifies the data, it does so correctly.



**Figure 9**: SOM U-Matrix with labels. Red indicates healthy, green indicates unbalanced 1, and blue indicates unbalanced 2.

Unfortunately, there are 5 points in the testing data set which the SOM will not classify, leaving us with a prediction score of 83.3% (25/30) as shown in Figure 10. Alternative features including the 3X, RMS, STD, and peak-to-peak of the signal were considered. At best, the prediction score stayed the same (25/30) and in some cases adding in these features degraded the prediction performance.



**Figure 10**: Confusion matrix for the SOM trained on the 1X and 2X harmonic

1. **Conclusions**

PCA was utilized to identify the most relevant features for distinguishing between healthy and faulty conditions. The analysis revealed that the 1X harmonic and 2X harmonic are the most significant variables, while other features such as RMS and standard deviation had minimal significance. SOM analysis was performed to classify the testing data into different health states. The results showed that the SOM model accurately classified the samples into healthy, unbalance 1, and unbalance 2 conditions based on the extracted features. Logistic regression was also employed (on the same data set )as a classification algorithm to predict the health state of the machinery based on the selected features. The model effectively classified the testing data into healthy and faulty conditions, with clear distinctions between different levels of unbalanced defects. Both SOM and logistic regression models demonstrated promising performance in classifying the health state of the rotor-bearing system. The analysis provided valuable insights into the effectiveness of different machine learning techniques for fault diagnosis in rotating machinery.

1. **Appendix**

**Feature Extraction:**

Brian O'Malley

%ENME 691 - Industrial AI

%HW3

% Spring 2024

clc;clear;close;

%% Top Matter

format long; format compact;

set(0,'defaultTextInterpreter','latex'); %trying to set the default

sz = 60; %Marker Size

szz = sz/35;

lw = 1;

ms=8;

fs=25;

txtsz = 30;

txtFactor = 0.8;

ax = [0.9,1.4,0.0,2.0];

loc = 'southwest';

pos = [218,114,1478,796];

plotFlag = false;

% txtsz = 24;

%%

%load in the data

dirTrainH = "F:\NOTES\Classes\Industrial AI\HW2\HW2\Homework 2\Training\Training\Healthy";

dirTrainF1 = "F:\NOTES\Classes\Industrial AI\HW3\Training\Faulty\Unbalance 1";

dirTrainF2 = "F:\NOTES\Classes\Industrial AI\HW3\Training\Faulty\Unbalance 2";

dirTest = "F:\NOTES\Classes\Industrial AI\HW2\HW2\Homework 2\Testing\Testing";

filesH = dir(dirTrainH +'\\*.txt');

filesF1 = dir(dirTrainF1 +'\\*.txt');

filesF2 = dir(dirTrainF2 +'\\*.txt');

filesT = dir(dirTest +'\\*.txt');

shaftspeed = 20;% Hz

Fs = 2560; % Sampling frequency

T = 1/Fs; % Sampling period

L = 38400; % Length of signal

time = (0:L-1)\*T; % Time vector

FsRange = Fs/L\*(0:L-1);

%set the range of interest (1x, 2x, 3x, 4x, harmonics)

low1X = find(FsRange==15);

up1X = find(FsRange==25);

low2X = find(FsRange==35);

up2X = find(FsRange==45);

low3X = find(FsRange==55);

up3X = find(FsRange==65);

low4X = find(FsRange==75);

up4X = find(FsRange==85);

%%

%transform healthy data

for i=1:length(filesH)

tempStr = dirTrainH +"\"+filesH(i).name;

trainingDataH{i} =readtable(tempStr);

%get additional data

t=abs(fft(trainingDataH{i}.Date3\_26\_2014));

peakToPeakH(i) = peak2peak(t);

L1H(i) = norm(t,1);

L2H(i) = norm(t,2);

LinfH(i) = norm(t,inf);

% plot(FsRange,fft(trainingDataH{i}.Date3\_26\_2014),'o')

transFFT\_H{i} = t; clear t;

end

%%

%transform faulty-1 data

for i=1:length(filesF1)

tempStr = dirTrainF1 +"\"+filesF1(i).name;

trainingDataF1{i} =readtable(tempStr);

t=abs(fft(trainingDataF1{i}.Date3\_26\_2014));

%get additional data

peakToPeakF1(i) = peak2peak(t);

L1F1(i) = norm(t,1);

L2F1(i) = norm(t,2);

LinfF1(i) = norm(t,inf);

transFFT\_F1{i} = t; clear t;

end

%%

%transform faulty-2 data

for i=1:length(filesF2)

tempStr = dirTrainF2 +"\"+filesF2(i).name;

trainingDataF2{i} =readtable(tempStr);

t=abs(fft(trainingDataF2{i}.Date3\_26\_2014));

%get additional data

peakToPeakF2(i) = peak2peak(t);

L1F2(i) = norm(t,1);

L2F2(i) = norm(t,2);

LinfF2(i) = norm(t,inf);

transFFT\_F2{i} = t; clear t;

end

%%

%transform testing data

for i=1:length(filesT)

tempStr = dirTest +"\"+filesT(i).name;

testData{i} =readtable(tempStr);

t=abs(fft(testData{i}.Date3\_26\_2014));

%get additional data

peakToPeakT(i) = peak2peak(t);

L1T(i) = norm(t,1);

L2T(i) = norm(t,2);

LinfT(i) = norm(t,inf);

transFFT\_T{i} = t; clear t;

end

%%

%pick out the peak for the healthy data

for i=1:20

% temp = transFFT\_H{i};

temp = transFFT\_H{i}/L\*4;

healthy1X(i) = max(temp(low1X:up1X));

healthy2X(i) = max(temp(low2X:up2X));

healthy3X(i) = max(temp(low3X:up3X));

healthy4X(i) = max(temp(low4X:up4X));

rmsH(i) = rms(temp);

stdH(i) = std(temp);

clear temp;

end

%%

%pick out the peak for the fault data

for i=1:20

% temp = transFFT\_F1{i};

temp = transFFT\_F1{i}/L\*4;

faulty11X(i) = max(temp(low1X:up1X));

faulty12X(i) = max(temp(low2X:up2X));

faulty13X(i) = max(temp(low3X:up3X));

faulty14X(i) = max(temp(low4X:up4X));

rmsF1(i) = rms(temp);

stdF1(i) = std(temp);

clear temp;

end

%%

%pick out the peak for the fault data

for i=1:20

% temp = transFFT\_F2{i};

temp = transFFT\_F2{i}/L\*4;

faulty21X(i) = max(temp(low1X:up1X));

faulty22X(i) = max(temp(low2X:up2X));

faulty23X(i) = max(temp(low3X:up3X));

faulty24X(i) = max(temp(low4X:up4X));

stdF2(i) = std(temp);

rmsF2(i) = rms(temp);

clear temp;

end

%%

%pick out the peak for the Testing data

for i=1:30

temp = transFFT\_T{i}/L\*4;

% temp = transFFT\_T{i};

testing1X(i) = max(temp(low1X:up1X));

testing2X(i) = max(temp(low2X:up2X));

testing3X(i) = max(temp(low3X:up3X));

testing4X(i) = max(temp(low4X:up4X));

stdT(i) = std(temp);

rmsT(i) = rms(temp);

clear temp;

end

%%

%all components

tempH = [healthy1X',healthy2X',healthy3X',healthy4X',stdH',rmsH',peakToPeakH',L1H',L2H',LinfH'];

tempF1 = [faulty11X',faulty12X',faulty13X',faulty14X',stdF1',rmsF1',peakToPeakF1',L1F1',L2F1',LinfF1'];

tempF2 = [faulty21X',faulty22X',faulty23X',faulty24X',stdF2',rmsF2',peakToPeakF2',L1F2',L2F2',LinfF2'];

tempT = [testing1X',testing2X',testing3X',testing4X',stdT',rmsT',peakToPeakT',L1T',L2T',LinfT'];

%load components into feature matrix

FeatMat\_train = [tempH;tempF1;tempF2];

FeatMat\_test = tempT;

cMatTest = [1\*ones(10,1);2\*ones(10,1);3\*ones(10,1)];

[coeff,score,latent,tsquared,explained,mu] = pca(FeatMat\_train);

%clear low impact components (only need 1st and 2nd harmonic)

% FeatMat\_train(:,3:end) =[];

% FeatMat\_test(:,3:end) = [];

save("FeatMat\_trainAllL4Scaled.mat","FeatMat\_train");

save("FeatMat\_testAllL4Scaled.mat","FeatMat\_test");

% save("cMatTest.mat","cMatTest");

return

%%

%plots of interest follow, 1 and 2 are the time domain amplitudes of

%the healthy and faulty data, 3 and 4 are the frequency domain plots

%plot 5 demonstrates the feature extraction

if (plotFlag == true)

figure(1)

grid on; hold on; box on;

% axis square;

ax=gca;

ax.FontSize = fs;

pbaspect([2 1 1])

xlim([0 15]);

ylim([-0.4 0.4]);

yticks([-0.4 -0.2 0 0.2 0.4]);

temp = table2array(trainingDataH{1});

plot(time,temp,'-b');

xlabel('Time [s]','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

clear temp;

% legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

figure(2)

grid on; hold on; box on;

% axis square;

pbaspect([2 1 1])

ax=gca;

ax.FontSize = fs;

xlim([0 15]);

ylim([-0.4 0.4]);

yticks([-0.4 -0.2 0 0.2 0.4]);

temp = table2array(trainingDataF1{1});

plot(time,temp,'-r');

xlabel('Time [s]','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

clear temp;

% legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

figure(3)

grid on; hold on; box on;

% axis square;

pbaspect([2 1 1])

ax=gca;

ax.FontSize = fs;

xlim([0 60]);

ylim([0 0.005]);

yticks([ 0 0.001 0.002 0.003 0.004 0.005]);

temp = transFFT\_H{1}/L\*2;

plot(FsRange, temp,'-b');

xlabel('Frequency [Hz]','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

clear temp;

figure(4)

grid on; hold on; box on;

% axis square;

pbaspect([2 1 1])

ax=gca;

ax.FontSize = fs;

xlim([0 60]);

ylim([0 0.005]);

yticks([ 0 0.001 0.002 0.003 0.004 0.005]);

temp = transFFT\_F1{1}/L\*2;

plot(FsRange, temp,'-r');

xlabel('Frequency [Hz]','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

clear temp;

% legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

% end

figure(5)

grid on; hold on; box on;

% axis square;

pbaspect([2 1 1])

ax=gca;

ax.FontSize = fs;

xlim([0 60]);

ylim([0 0.005]);

yticks([ 0 0.001 0.002 0.003 0.004 0.005]);

temp = transFFT\_F2{1}/L\*2;

plot(FsRange, temp,'-r');

xlabel('Frequency [Hz]','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

clear temp;

% legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

% end

figure(6)

grid on; hold on; box on;

% axis square;

ax=gca;

ax.FontSize = fs;

yticks([ 0 0.01 0.02 0.03]);

pbaspect([2 1 1])

xlim([0 20]);

ylim([0 0.03]);

plot(healthy1X,'-ob');

plot(faulty11X,'-or');

plot(faulty21X,'-ok');

legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

xlabel('# of Samples','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

figure(7)

grid on; hold on; box on;

% axis square;

ax=gca;

ax.FontSize = fs;

yticks([ 0 0.01 0.02 0.03]);

pbaspect([2 1 1])

xlim([0 20]);

ylim([0 0.03]);

plot(healthy2X,'-ob');

plot(faulty12X,'-or');

plot(faulty22X,'-ok');

legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

xlabel('# of Samples','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

figure(8)

grid on; hold on; box on;

% axis square;

ax=gca;

ax.FontSize = fs;

yticks([ 0 0.01 0.02 0.03]);

pbaspect([2 1 1])

xlim([0 20]);

ylim([0 0.03]);

plot(healthy3X,'-ob');

plot(faulty13X,'-or');

plot(faulty23X,'-ok');

legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

xlabel('# of Samples','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

figure(9)

grid on; hold on; box on;

% axis square;

ax=gca;

ax.FontSize = fs;

yticks([ 0 0.01 0.02 0.03]);

pbaspect([2 1 1])

xlim([0 20]);

ylim([0 0.03]);

plot(healthy4X,'-ob');

plot(faulty14X,'-or');

plot(faulty24X,'-ok');

legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

xlabel('# of Samples','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

figure(10)

grid on; hold on; box on;

% axis square;

ax=gca;

ax.FontSize = fs;

yticks([ 0 0.01 0.02 0.03]);

pbaspect([2 1 1])

% xlim([0 20]);

% ylim([0 0.03]);

plot(stdH,'-ob');

plot(stdF1,'-or');

plot(stdF2,'-ok');

legend('Healthy','Faulty','Location','Northwest','FontSize',fs);

xlabel('# of Samples','FontSize',fs);

ylabel('Amplitude [-]','FontSize',fs);

end

**Logistic Regression:**

%E-Manufacturing 2013

% modified by Brian O'Malley (2024)

%Logistic Regression

clc;clear;close;

%% Top Matter

format long; format compact;

set(0,'defaultTextInterpreter','latex'); %trying to set the default

sz = 60; %Marker Size

szz = sz/35;

lw = 1;

ms=8;

fs=25;

txtsz = 30;

txtFactor = 0.8;

ax = [0.9,1.4,0.0,2.0];

loc = 'southwest';

pos = [218,114,1478,796];

% txtsz = 24;

FeatMatTrain = struct2array(load("FeatMat\_train.mat"));

FeatMatTest= struct2array(load("FeatMat\_test.mat"));

%only need the 1st harmonic for this

FeatMatTrain(:,2) = [];

FeatMatTest(:,2) = [];

FeatureMatrix = [FeatMatTrain ; FeatMatTest];

%% Select Training Portion and PCA (front/back)

%GoodSampleIndex (in this example I assume the baseline data is first 100

%samples

GoodSampleIndex=1:20;

DegradedSampleIndex=21:60;

%Baseline Data

BaselineData=FeatureMatrix(GoodSampleIndex,:);

DegradedData=FeatureMatrix(DegradedSampleIndex,:);

%% Train LR Model

%Label Vector (0.95 for good samples, 0.05 for bad samples

Label=[ones(size(BaselineData,1),1)\*0.95; ones(size(DegradedData,1),1)\*0.05];

%fit LR Model (glm-fit)

beta = glmfit([BaselineData; DegradedData],Label,'binomial');

%% Calculating Health Value (using LR Model)

TestFeatureMatrix=FeatureMatrix(61:90,:);

%calculate CV (Health Value)

CV\_Test = glmval(beta,TestFeatureMatrix,'logit') ; %Use LR Model

%%

figure(1)

grid on; hold on; box on;

% axis square;

ax=gca;

pbaspect([2 1 1])

ax.FontSize = fs;

plot(CV\_Test,'-ob');

xlabel('# of Samples','FontSize',fs);

ylabel('CV','FontSize',fs);

ylim([0 1]);

yticks([0 0.25 0.5 0.75 1.0]);

**SOM-MQE:**

%% SOM-MQE

clc

clear;close;

%% Top Matter

format long; format compact;

set(0,'defaultTextInterpreter','latex'); %trying to set the default

sz = 60; %Marker Size

szz = sz/35;

lw = 1;

ms=8;

fs=25;

txtsz = 30;

txtFactor = 0.8;

ax = [0.9,1.4,0.0,2.0];

loc = 'southwest';

pos = [218,114,1478,796];

% txtsz = 24;

%%

% Load the data

load FeatMat\_test.mat

load FeatMat\_train.mat

addpath('F:\NOTES\Classes\Industrial AI\HW4\HW4\HW4\Useful Code\SOM-Toolbox-master\som');% add the path

%only need 1st harmonic

FeatMat\_train(:,2:end) = []

FeatMat\_test(:,2:end) = []

% load FeatMat\_test.mat

% load FeatMat\_train.mat

%% Training of SOM with Normal condition data

%

TrainData=FeatMat\_train(1:20);

TestData=FeatMat\_test;

sM=som\_make(TrainData);

%% Calculate the MQE values for the testing data set

S=size(TestData);

S=S(1);

for ii=1:S

qe=som\_quality(sM,TestData(ii,:)); % calculate MQE value for each sample

MQEt(ii)=qe;

end

MQEtn=(1-(MQEt)./(max(MQEt))); % normalize MQE

MQEtn=MQEtn';

%% Plot the calculated MQE values

% observe the difference between normal condition and faluty conditions

plot(MQEtn,'-o','Color','Blue');

xlabel('Data file No.','FontSize',fs);

ylabel('Confidence value (MQE)','FontSize',fs);

% title('Health Assessment Plot','FontSize',fs);

figure(1)

grid on; hold on; box on;

% axis square;

ax=gca;

pbaspect([2 1 1])

ax.FontSize = fs;

ylim([0 1]);

**SOM:**

clc;clear;close

%% Load paths for SOM toolbox

addpath('F:\NOTES\Classes\Industrial AI\HW4\HW4\HW4\Useful Code\SOM-Toolbox-master\dijkstra');

addpath('F:\NOTES\Classes\Industrial AI\HW4\HW4\HW4\Useful Code\SOM-Toolbox-master\som');%

%% load the data

load FeatMat\_train.mat

load FeatMat\_test.mat

%using both 1st and 2nd harmonics for this one

% FeatMat\_train(:,2) =FeatMat\_train(:,1) ;

% FeatMat\_test(:,2) = FeatMat\_test(:,1) ;

%% Create SOM data structure

sDBear = som\_data\_struct(FeatMat\_train,'name','Rotor-Bearing',...

'comp\_names',{'1X','2X'});

%%

for i = 1:length(FeatMat\_train)/3

labelTrain{i} = 'Healthy';

labelTrain{i+20} = 'Faulty1';

labelTrain{i+40} = 'Faulty2';

end

%% Add labels to the data

sDBear = som\_label (sDBear,'add',[1:20],labelTrain{1});

sDBear = som\_label (sDBear,'add',[21:40],labelTrain{21});

sDBear = som\_label (sDBear,'add',[41:60],labelTrain{41});

%% Initialization and training of the maps

sMap = som\_make(sDBear);

sMap = som\_autolabel(sMap,sDBear,'vote');

%% Visualize the maps

som\_show(sMap);

% U-matrix with labels

figure;

som\_show(sMap,'umat','all','empty','Labels');

som\_show\_add('label',sMap,'Textsize',8,'TextColor','r','Subplot',2)

%% See the hit points for healthy and faulty sets

colormap(1-gray)

som\_show(sMap,'umat','all','empty','Labels');

% Add labels to the map

som\_show\_add('label',sMap,'Textsize',8,'TextColor','r','Subplot',2)

% sampels for test

h1 = som\_hits(sMap,sDBear.data(1,:)); % data from healthy (Red)

h2 = som\_hits(sMap,sDBear.data(21,:)); % data faulty 1 (Green)

h3 = som\_hits(sMap,sDBear.data(41,:)); % data faulty 2 (Blue)

% diagnosis result

h1\_label = sMap.labels(h1==1);

h2\_label = sMap.labels(h2==1);

h3\_label = sMap.labels(h3==1);

% Use hit point to show diagnosis result

som\_show\_add('hit',[h1, h2, h3],'MarkerColor',[1 0 0 ; 0 1 0 ; 0 0 1 ],'Subplot',1)

%% Results for testing data

colormap(1-gray)

som\_show(sMap,'umat','all','empty','Labels');

% Add labels to the map

som\_show\_add('label',sMap,'Textsize',8,'TextColor','r','Subplot',2)

h1 = som\_hits(sMap,FeatMat\_test(1:10,:)); % test data healthy - red

h2 = som\_hits(sMap,FeatMat\_test(11:20,:)); % test data faulty 1 - green

h3 = som\_hits(sMap,FeatMat\_test(21:30,:)); % test data faulty 2 - blue

% diagnosis result

h1\_label = sMap.labels(h1==1);

h2\_label = sMap.labels(h2==1);

h3\_label = sMap.labels(h3==1);

% Use hit point to show diagnosis result

som\_show\_add('hit',[h1, h2, h3],'MarkerColor',[1 0 0 ; 0 1 0 ; 0 0 1 ],'Subplot',1)

%%

clear h1 h2 h3 h1\_label h2\_label h3\_label

for i=1:10

h1 = som\_hits(sMap,FeatMat\_test(i,:)); % test data healthy - red

h2 = som\_hits(sMap,FeatMat\_test(i+10,:)); % test data faulty 1 - green

h3 = som\_hits(sMap,FeatMat\_test(i+20,:)); % test data faulty 2 - blue

h1\_label{i} = sMap.labels(h1==1);

h2\_label{i} = sMap.labels(h2==1);

h3\_label{i} = sMap.labels(h3==1);

temp = convertCharsToStrings(cell2mat(h1\_label{i}));

predicted\_result(i) = temp; clear temp;

temp = convertCharsToStrings(cell2mat(h2\_label{i}));

predicted\_result(i+10) = temp; clear temp;

temp = convertCharsToStrings(cell2mat(h3\_label{i}));

predicted\_result(i+20) = temp; clear temp;

end

%%

for i=1:10

testLabels(i) = "Healthy";

testLabels(i+10) = "Faulty1";

testLabels(i+20) = "Faulty2";

end

%% Confusion Matrix

%need to exclude 5 data points that are unlabeled for the formatting to

%work

cm = confusionchart(testLabels(1:25),predicted\_result(1:25))

% cm = confusionchart(testLabels,predicted\_result)