BRAE 428 Midterm 2021

Work independently.

DO NOT share your work with other people.

Due: 5:00 pm, May / 07 /2021

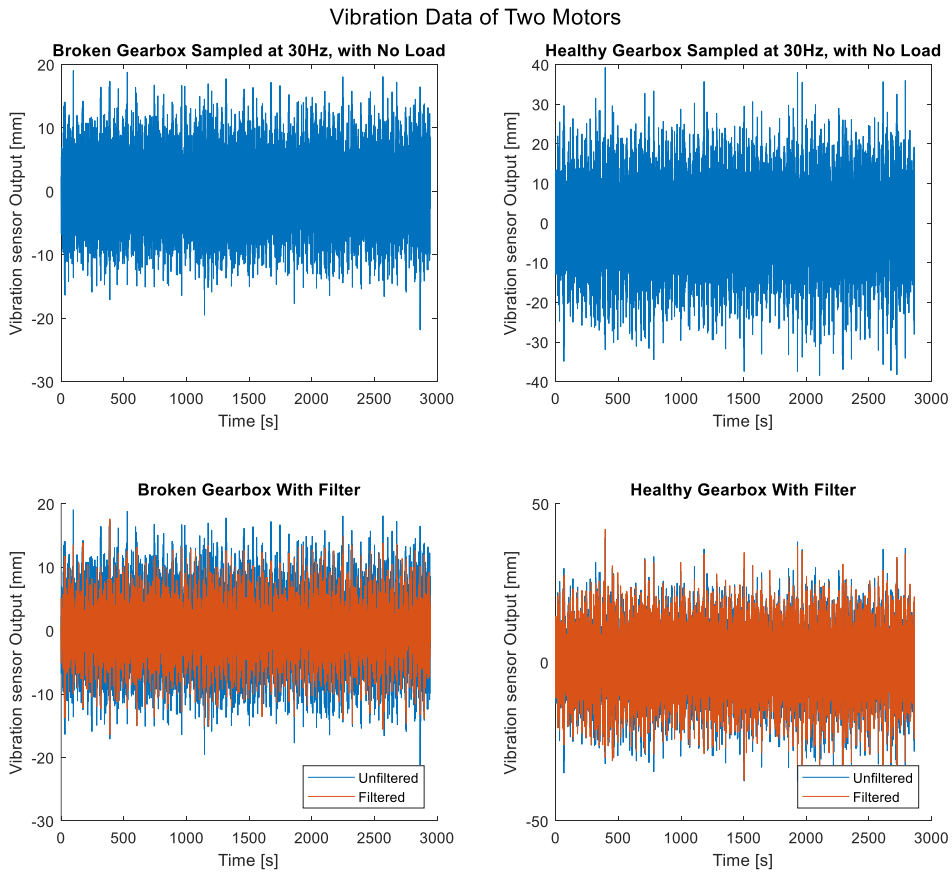
Problem 1. (30 points)

Two vibration sensors were placed on top of two gearboxes (one healthy condition and one broken-tooth condition) to record vibration data. The data was collected at 30 Hz with no load on the gearboxes.

Write a MATLAB program to analyze the data of these two gearboxes:

1. Plot the vibration data from both gearboxes in the time domain (x is seconds, y is mm)
2. Filter the noise signals from the collected data. We know that gearbox vibration frequencies are higher than 5 Hz.
3. What is the difference between the peak frequencies of the two vibration signals?

Comment your program and put it in a Word document. Your program should run without any problems when the Run button is pressed.



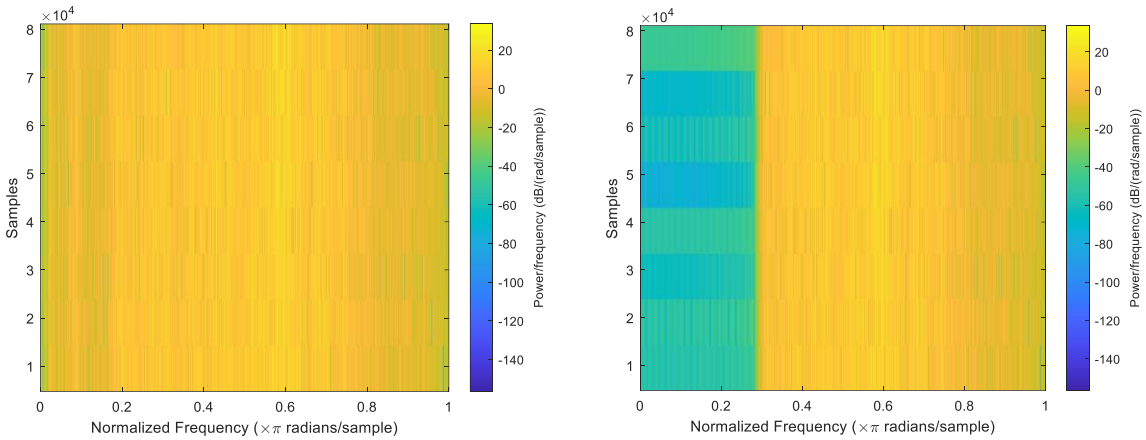


Figure 2. Showing Spectrogram of Filtered Healthy Motor

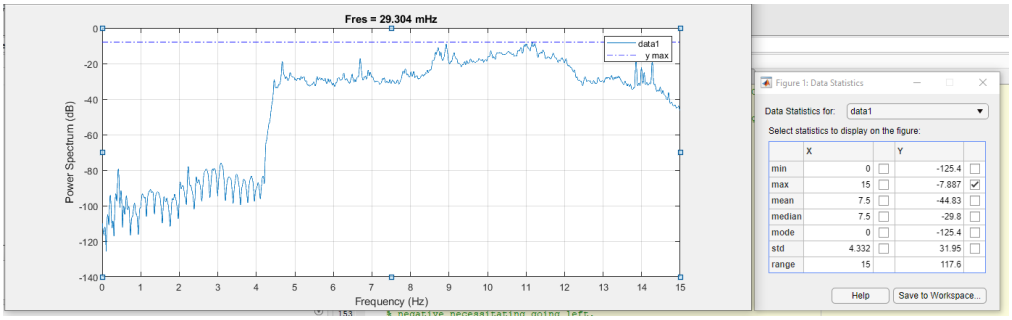


Figure 3. Power Spectrum of Broken Filtered Data

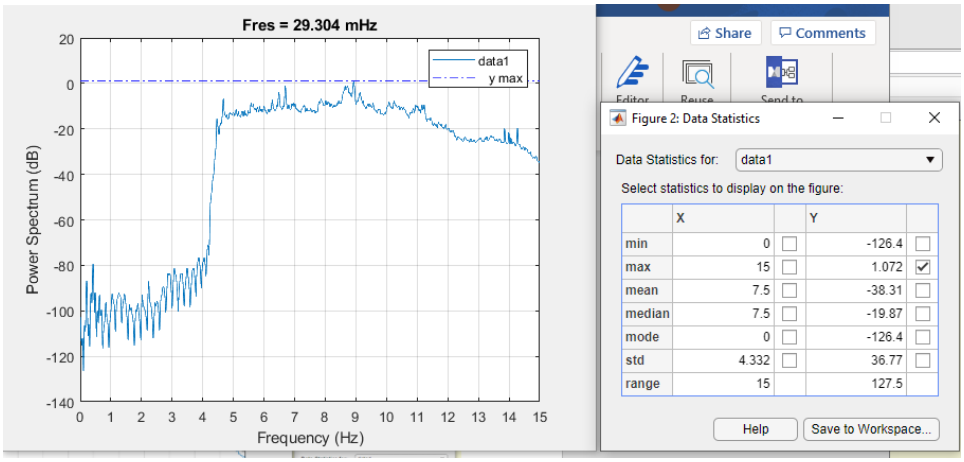


Figure 4. Power Spectrum of Healthy Filtered Data

Part 3.

```
% The peak frequency of the proken motor is roughly 11.25 Hz. It is
% operating at -7 dB. The peak frequency of the healthy motor is 9hz. It is
% operating at 1.072 Db. It makes sense that the healthy motor would have
% an increased power delivery.
```

Code for Problem 1

```
%Import Data
brokendata =
csvread("MidtermData2021\MidtermData2021\Problem1\BrokenGearbox30HzNoLoad.csv
");
healthydata =
csvread("MidtermData2021\MidtermData2021\Problem1\HealthyGearbox30HzNoLoad.cs
v");

%Info about data
size(brokendata);
size(healthydata);

%Generate corresponding time of each sample for each dataset
time = 0;
timearray = zeros(88320, 1);
for i = (1:88320)
    timearray1(i,1) = time;

    time = time + 1/30;
end
time = 0;
for i = (1:85873)
    timearray2(i,1) = time;

    time = time + 1/30;
end

%Plot default outputs
figure();
subplot(2,2,1);
plot(timearray1,brokendata);
title('Broken Gearbox Sampled at 30Hz, with No Load');
xlabel('Time [s]');
ylabel('Vibration sensor Output [mm]');


subplot(2,2,2);
%
plot(timearray2,healthydata);
title('Healthy Gearbox Sampled at 30Hz, with No Load');
xlabel('Time [s]');
ylabel('Vibration sensor Output [mm]');
sgtitle('Vibration Data of Two Motors');
%Eliminate the Noise

% Now Filter anything > 5Hz (corresponding to gearbox vibrations)

brokenfiltered = highpass(brokendata,5,30);
healthyfiltered = highpass(healthydata,5,30);
```

```
%Plot Filtered Broken Vs Unfiltered Broken


subplot(2,2,3);
hold on;
plot(timearray1,brokendata);
plot(timearray1,brokenfiltered);

legend({'Unfiltered', 'Filtered'},'Location','southeast');
title('Broken Gearbox With Filter');
xlabel('Time [s]');
ylabel('Vibration sensor Output [mm]');

%Plot Filtered Healthy Vs Unfiltered Healthy

subplot(2,2,4);
hold on;
plot(timearray2,healthydata);
plot(timearray2,healthyfiltered);
legend({'Unfiltered', 'Filtered'}, 'Location','southeast');
title('Healthy Gearbox With Filter');
xlabel('Time [s]');
ylabel('Vibration sensor Output [mm]');

% Find Peak Frequencies of Filtered Data


figure()
pspectrum(healthyfiltered,timearray2)
figure()
pspectrum(brokenfiltered,timearray1)

% The peak frequency of the proken motor is roughly 11.25 Hz. It is
% operating at -7 dB. The peak frequency of the healthy motor is 9hz. It is
% operating at 1.072 Db. It makes sense that the healthy motor would have
% an increased power delivery.
```

Problem 2. (40 points)

You are asked to work on the computer vision on a strawberry harvester robot. The first task of this computer vision system is to identify the strawberry bed right beneath the robot, so the robot can follow it later. A sample image from the robot's front camera view is shown below.

1. Write a MATLAB program to extract the strawberry bed areas from the sample image. You can do any data manipulation as you wish.
2. Find the strawberry bed right beneath the robot (assuming the camera is mounted near the front center of the robot)
3. Find the angle difference between the robot's heading and the strawberry bed right beneath the robot.

4. How can you use the angle difference found in step 3 to control the robot to follow the strawberry bed?

Comment your program and put it in a Word document. Your program should run without any problems when the Run button is pressed.

Figure 5. Isolated Middle Row

Code for Problem 2

```
% Import Image
I = imread('MidtermData2021\MidtermData2021\Problem2.jpg');
imshow(I)
BW = createMask(I); % Generated code using HSV color scheme in Color
Thresholder
imshow(BW)
[BW1,info] = filterRegions(BW); %Generated code using 'Image Region Analyzer'
s = info.Orientation;
```

```
if s > 0
    angle = 90-s;
elseif s == 0
    angle = 90;
else
    angle = -(180 + s - 90);
end
angle
% Since the Orientation is measured from the positive x axis,
% If the Orientation is negative, we ought to find the relative angle from
% the positive Y axis, this being the forward part of the robot will allow
% us to adjust to follow the line, positive necessitating going right, and
% negative necessitating going left.

%Thus in this case, it makes sense for the craft to understand it needs to
%adjust 20 degrees to the left, to adequately change course and finish the
%drive.
```

Function 1 for Problem 2

```
function [BW,maskedRGBImage] = createMask(RGB)
%createMask  Threshold RGB image using auto-generated code from
colorThresholder app.
%  [BW,MASKEDRGBIMAGE] = createMask(RGB) thresholds image RGB using
%  auto-generated code from the colorThresholder app. The colorspace and
%  range for each channel of the colorspace were set within the app. The
%  segmentation mask is returned in BW, and a composite of the mask and
%  original RGB images is returned in maskedRGBImage.

% Auto-generated by colorThresholder app on 06-May-2021
%------------------------------------------------------


% Convert RGB image to chosen color space
I = rgb2hsv(RGB);

% Define thresholds for channel 1 based on histogram settings
channel1Min = 0.165;
channel1Max = 0.309;

% Define thresholds for channel 2 based on histogram settings
channel2Min = 0.000;
channel2Max = 1.000;

% Define thresholds for channel 3 based on histogram settings
channel3Min = 0.000;
channel3Max = 1.000;

% Create mask based on chosen histogram thresholds
sliderBW = (I(:,:,1) >= channel1Min ) & (I(:,:,1) <= channel1Max) & ...
    (I(:,:,2) >= channel2Min ) & (I(:,:,2) <= channel2Max) & ...
    (I(:,:,3) >= channel3Min ) & (I(:,:,3) <= channel3Max);
```

```
BW = sliderBW;

% Initialize output masked image based on input image.
maskedRGBImage = RGB;

% Set background pixels where BW is false to zero.
maskedRGBImage(repmat(~BW,[1 1 3])) = 0;

end
```

Function 2 for problem 2

```
function [BW_out,properties] = filterRegions(BW_in)
%filterRegions  Filter BW image using auto-generated code from
imageRegionAnalyzer app.
%  [BW_OUT,PROPERTIES] = filterRegions(BW_IN) filters binary image BW_IN
%  using auto-generated code from the imageRegionAnalyzer app. BW_OUT has
%  had all of the options and filtering selections that were specified in
%  imageRegionAnalyzer applied to it. The PROPERTIES structure contains the
%  attributes of BW_out that were visible in the app.

% Auto-generated by imageRegionAnalyzer app on 06-May-2021
%-----------------------------------------------------------

BW_out = BW_in;

% Fill holes in regions.
BW_out = imfill(BW_out, 'holes');

% Filter image based on image properties.
BW_out = bwpropfilt(BW_out, 'Area', [5000, 22000]);

% Get properties.
properties = regionprops(BW_out, {'Area', 'Eccentricity', 'EquivDiameter',
'EulerNumber', 'MajorAxisLength', 'MinorAxisLength', 'Orientation',
'Perimeter'});

% Uncomment the following line to return the properties in a table.
% properties = struct2table(properties);
end
```

Problem 3. (30 points)

Create a simple (shallow) artificial neural network that learns the model of the system (regression) shown below using **MATLAB Neutral Net Fitting** app. x is the input of the system, and y is the output of the system. The data is attached on Canvas.

$$X(\text{input}) \rightarrow \boxed{\text{System}} \rightarrow Y\ (\text{output})$$

1. Plot the performance chart
2. Test the trained network with some new inputs the model has never seen before.
3. The model of the system is very similar to which equation?

Comment your program and put it in the Word document. Answer the question in the Word document.

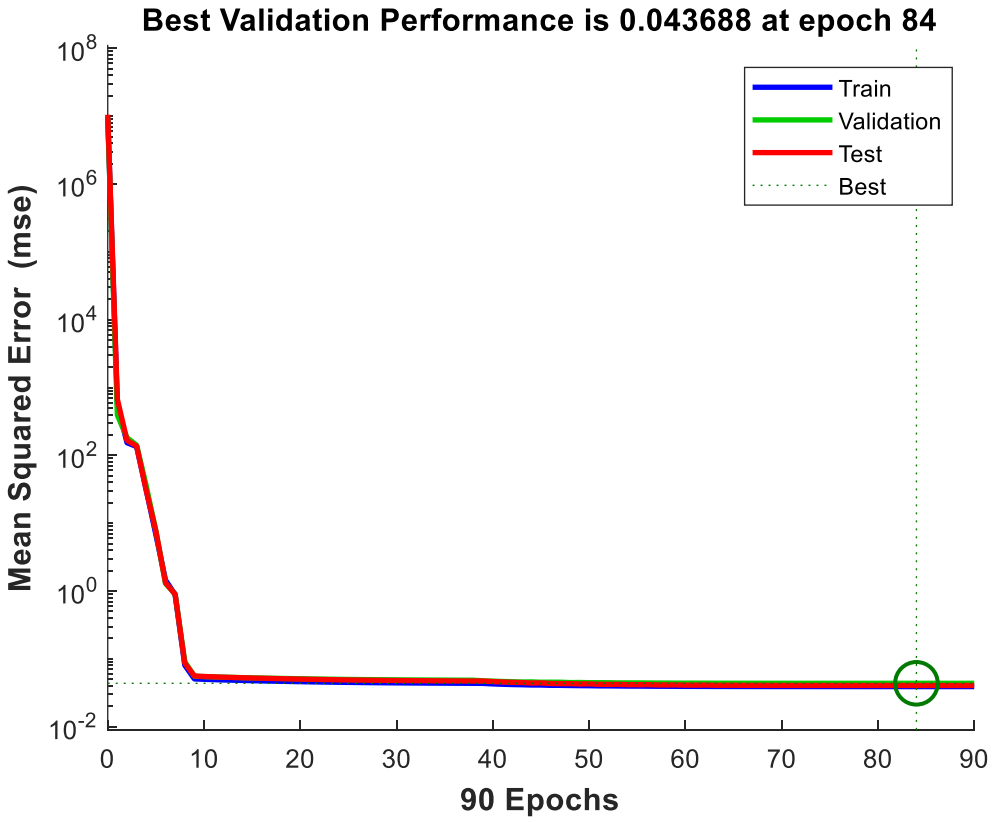


Figure 6. Performanace Chart

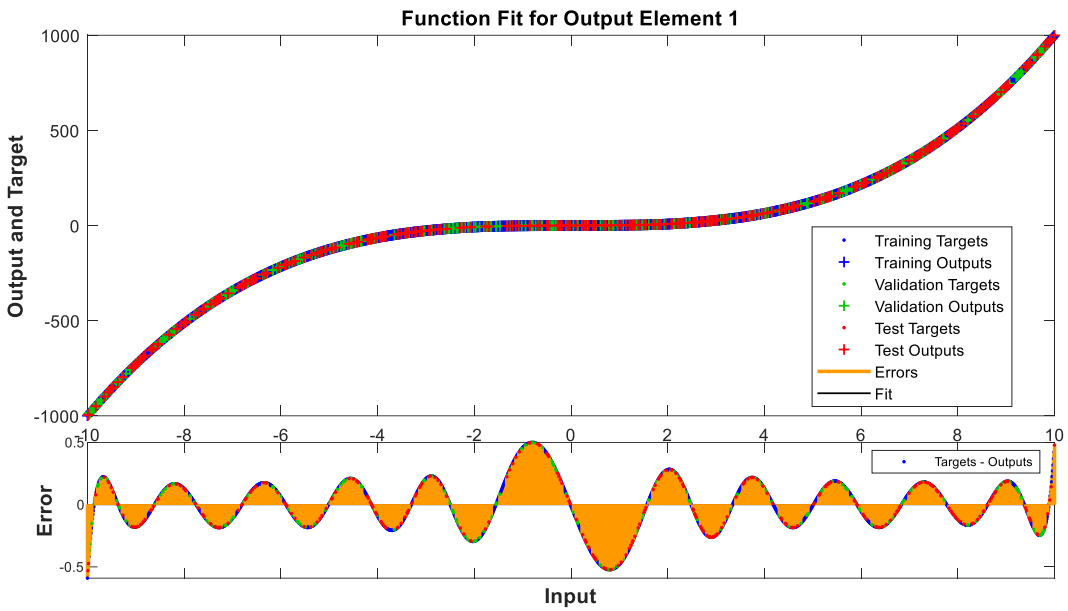


Figure 7. Test & Validation data plotted together with Error

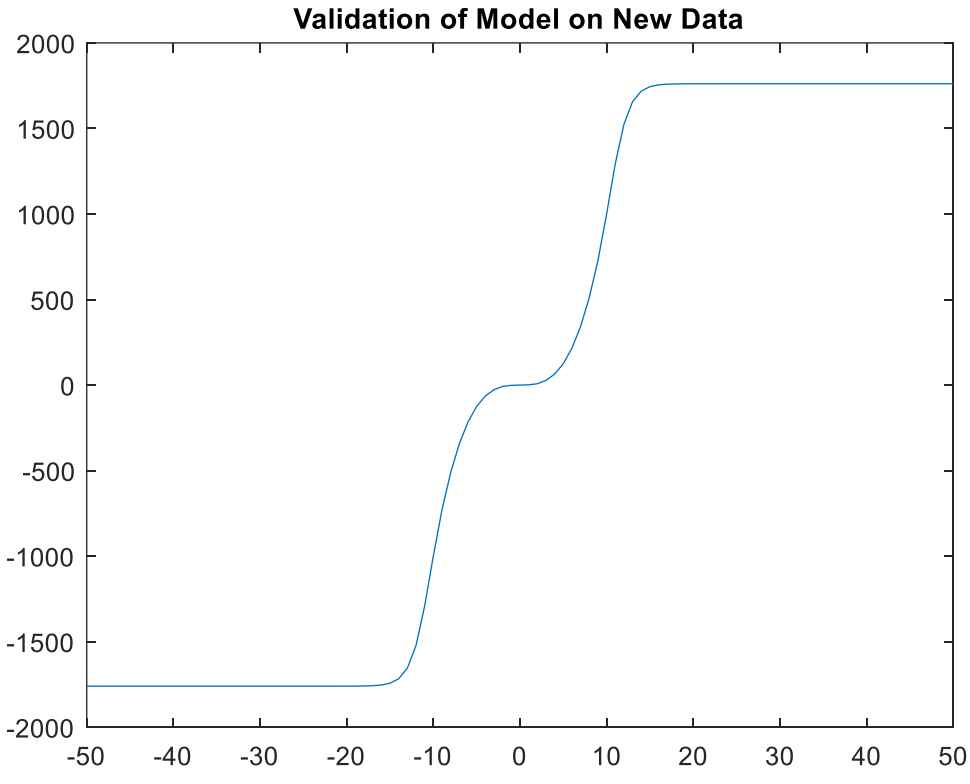


Figure 8. Plot Showing Data Well Beyond the Training Set

Code for Problem 3

```
%First Import Data

data = load("MidtermData2021\MidtermData2021\Problem3.mat");

% Data Preparation

labels  =data.y;

Features = data.x;

%Test new datapoints
newx = [];
newy = [];
for i = (-50:1:50)
    newoutput = myNeuralNetworkFunction(i);
    newy= [newy; newoutput];
    newx= [newx; i];
end

%Plot new data

plot(newx,newy)
title("Validation of Model on New Data")
% Interestingly, the model does not continue the trend towards infinity
```

Function 1 for Problem 3

```
function [y1] = myNeuralNetworkFunction(x1)
%MYNEURALNETWORKFUNCTION neural network simulation function.
%
% Auto-generated by MATLAB, 06-May-2021 23:06:00.
%
% [y1] = myNeuralNetworkFunction(x1) takes these arguments:
%   x = 1xQ matrix, input #1
% and returns:
%   y = 1xQ matrix, output #1
% where Q is the number of samples.

%#ok<*RPMT0>

% ===== NEURAL NETWORK CONSTANTS =====

% Input 1
x1_step1.xoffset = -10;
x1_step1.gain = 0.1;
x1_step1.ymin = -1;
```

```
% Layer 1
b1 = [-4.9554802440954794918;-4.1764353047517319695;3.7439172519214873888;-
3.3392687325695615819;-2.868244171067389825;-2.8797479940162982182;-
3.3260946959464745554;3.7261212417737765712;4.1473854049044449614;4.9371017183
762502256];
IW1_1 = [4.7135001397620470698;5.3308276491539832165;-
6.3971215015507949531;8.1031712549590437789;11.241496909932402914;-
11.358769730595426495;-
8.1235329542446965689;6.3917180950397023409;5.3041476808984082684;4.6986851183
021993705];

% Layer 2
b2 = 0.00077088815464440585152;
LW2_1 = [0.58894845839409382116 0.17855961244481047268 -
0.075370382499736959803 0.029480010991924533031 0.008280235737933435216 7 -
0.007959902663809937176 1 -0.028954289234045509777 0.074478540823433636575
0.17903257868243846462 0.58946089150446734362];

% Output 1
y1_step1.ymin = -1;
y1_step1.gain = 0.001;
y1_step1.xoffset = -1000;

% ===== SIMULATION ========

% Dimensions
Q = size(x1,2); % samples

% Input 1
xp1 = mapminmax_apply(x1,x1_step1);

% Layer 1
a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*xp1);

% Layer 2
a2 = repmat(b2,1,Q) + LW2_1*a1;

% Output 1
y1 = mapminmax_reverse(a2,y1_step1);
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function
function y = mapminmax_apply(x,settings)
y = bsxfun(@minus,x,settings.xoffset);
y = bsxfun(@times,y,settings.gain);
y = bsxfun(@plus,y,settings.ymin);
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n,~)
a = 2 ./ (1 + exp(-2*n)) - 1;
end
```

```
% Map Minimum and Maximum Output Reverse-Processing Function
function x = mapminmax_reverse(y,settings)
x = bsxfun(@minus,y,settings.ymin);
x = bsxfun(@rdivide,x,settings.gain);
x = bsxfun(@plus,x,settings.xoffset);
end
```