Project1: Condition Monitoring of Ball Bearing using Hjorth's parameters

Handong Global University

School of Mechanical & Control Engineering

Industrial AI & Automation 24-2

Team: HSN

Teammates: Yun-ki Noh 21800226, Si-on Seo 21800360, Ho-jin Jang 22100645

Date: 2024.10.22

Wind Turbine High-Speed Bearing Prognosis Dataset

1. Introduction

The fault of the bearing is the biggest cause of factory fault. When ball bearing faults happen, people have to stop factory system including ball bearing part. And as spending time to check which ball breaing was broken, people had to pay high cost from delay time of operating factory. In this reason, people tried to predict Remaining Useful Life(RUL) of ball bearing with many features from ball bearing's signal.

However, extracting many features and selecting specific features as a health indicator demand a lot of times. In this situation, to reduce consumming time for extracting & selecting features for health indicator, Detectivity can be used as a health indicator feature. From 'Detectivity: A combination of Hjorth's parameters for condition monitoring of ball bearings.' of '*Mechanical Systems and Signal Processing',* detectivity can be calculated like this: [1]

$$\rightarrow y = \text{ball bearing signal data}$$

$$\rightarrow \text{Activity} = \sigma^2,\ \text{Mobility} = \sqrt{\frac{\text{Activity}(\text{diff}(y))}{\text{Activity}(y)}},\ \text{Complexity} = \sqrt{\frac{\text{Mobility}(\text{diff}(y))}{\text{Mobility}(y)}}$$

$$\rightarrow A_{\text{ref}} = \text{mean}(\text{Activity}),\ M_{\text{ref}} = \text{mean}(\text{Mobility}),\ C_{\text{ref}} = \text{mean}(\text{Complexity})$$

$$\rightarrow \text{Act}_{(\text{dB})} = 10 * \log_{10}\left(\frac{\text{Act}}{\text{Act}_{\text{ref}}}\right), \text{Mob}_{(\text{dB})} = 10 * \log_{10}\left(\frac{\text{Mob}}{\text{Mob}_{\text{ref}}}\right), \text{Com}_{(\text{dB})} = 10 * \log_{10}\left(\frac{\text{Com}}{\text{Com}_{\text{ref}}}\right)$$

$$\rightarrow \text{Detectivity} = \text{Act}_{(\text{dB})} - \text{Mob}_{(\text{dB})} + \text{Com}_{(\text{dB})}$$

Hjorth's parameters(Activity, Mobility, and Complexity) are indicator as time domain features.[1]. In commen case, these features have been used for analysis of electroencephalography signals. From these three parameters, we calculated new parameter, 'Detectivity'[1]. Detectivity is indicator for detecting ball bearing fault from bearing signal. From the 'Detectivity' journal, we would able to reduce the time spent extracting and selecting features by using Detectivity as a health indicator for RUL prediction. To comfirm this

arguement, we planned to compare RUL prediction performance from Detectivity with traditional features. With WindTurbineProgonosis Dataset, we worked on the project like this:

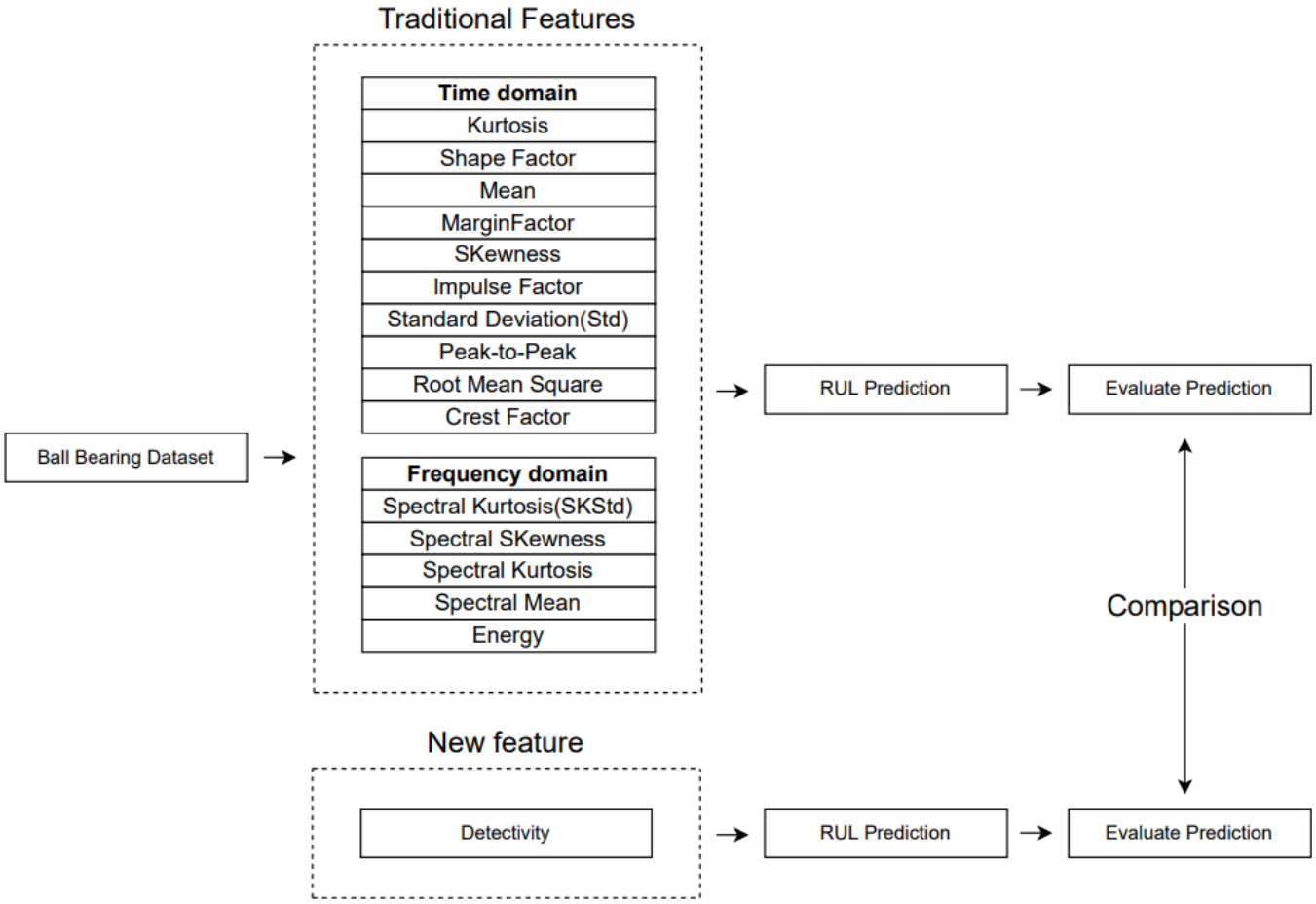

Figure 1. Overall Structure of Our Project

In the Case 1, we used "MATLAB's Wind Turbine Bearinng Prognosis Tutorial" to prepare detectivity's RUL prediction performance. [Link: 풍력 터빈 고속 베어링 예지진단 - MATLAB & Simulink - MathWorks 한국]

2. Dataset

- WindTurbineProgonosis Dataset:

- Hardware: 20-tooth pinion gea / 2MW
- Sampling time: 6s per day
- Samplng period: 50 days
- Sampling frequency: 97656Hz
- Data number: 585,936 data of each 50 files
- link: GitHub - mathworks/WindTurbineHighSpeedBearingPrognosis-Data: Data set for Wind Turbine High-Speed Bearing Prognosis example in Predictive Maintenance Toolbox

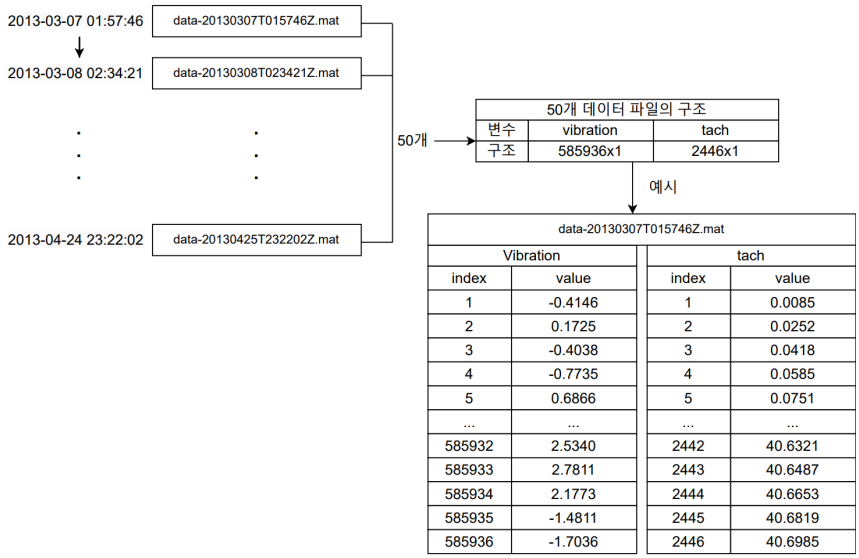

Figure 2. Structure of WindTurbineProgonsis Dataset

2-1. Import Dataset

```
clear all

% Import WT dataset
hsbearing = fileEnsembleDatastore('Full-Data', '.mat');

timeUnit = 'day';

hsbearing.DataVariables = ["vibration", "tach"];
hsbearing.IndependentVariables = "Date";
hsbearing.SelectedVariables = ["Date", "vibration", "tach"];
hsbearing.ReadFcn = @helperReadData;
hsbearing.WriteToMemberFcn = @helperWriteToHSBearing;
tall(hsbearing);  % Show Elements of Selected Variables.

fs = 97656; % Hz
```

2-2. Plot Raw Dataset

This dataset include wind turbine vibrations signal. To understand the whold trend of data, We plotted vibration data from 0 to 50 days.

```
% Reset hsbearing
reset(hsbearing);
tstart = 0;

% Create and center figure
figure('Units', 'normalized', 'Position', [0.3, 0.3, 0.4, 0.6]); % Center the figure

hold on;
while hasdata(hsbearing)
    data = read(hsbearing);

    v = data.vibration{1};
    t = tstart + (1:length(v)) / fs;
    plot(t(1:10:end), v(1:10:end));
    tstart = t(end);
end
hold off;

% Labeling and formatting the plot
xlabel('Time (s), 6 seconds per day, 50 days in total');
ylabel('Acceleration (g)');
title('Vibration Data Over Time');
grid on;
```

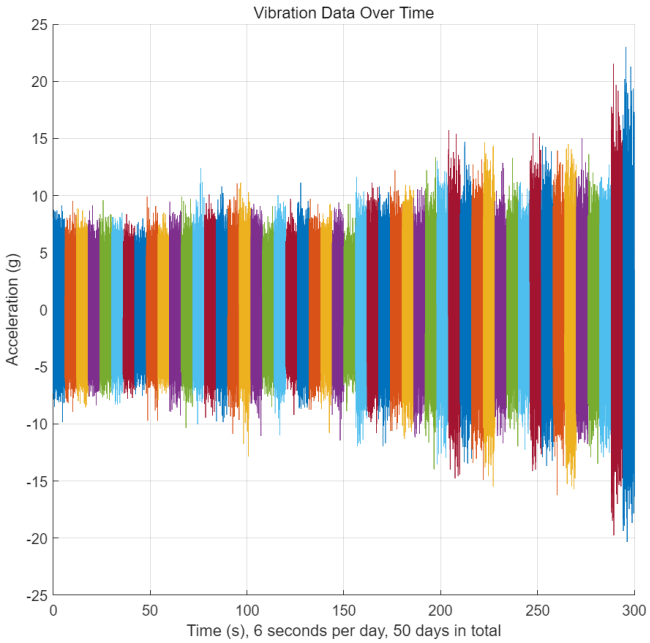

Figure 3. Graph of WindTurbineProgonsis Dataset, Vibration Dataset

```
% Reset hsbearing
```

```
reset(hsbearing);
tstart = 0;

% Plot tachometer's dataset
figure;
hold on;
while hasdata(hsbearing)
    data = read(hsbearing);

    tach = data.tach{1};
    t = tstart + (1:length(tach))/fs;
    plot(t, tach);
    tstart = t(end);
end
hold off;
xlabel('Time (s), 6 seconds per day, 50 days in total');
ylabel('RPM');
title('Tachometer Data Over Time');
grid on;
```

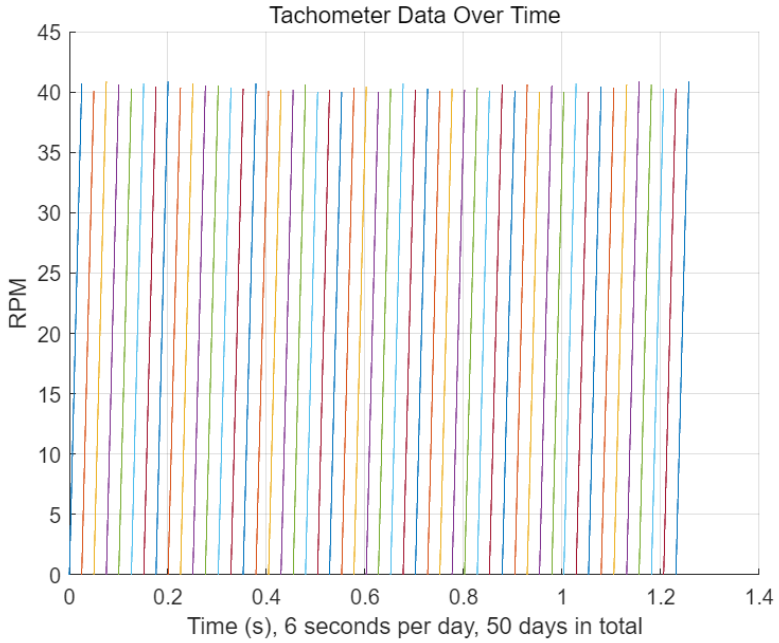

Figure 4. Graph of WindTurbineProgonsis Dataset, Tach Dataset

Vibration signal seems to increase as closing to the fault time. But, Tachometer signal shows the periodic RPM change during 50 days. Because vibration signal include fault traits, we selected vibration dataset to predict RUL.

2-3. Exract Only Vibration Dataset

We extracted vibration signal from dataset to get detectivity.

```
% Specify the data variables to be processed
```

```
hsbearing.DataVariables = ["vibration","tach"];
colors = parula(50);
reset(hsbearing)
day = 1;
vibration_data = zeros(585936,50);
tachometer_data = zeros(2446,50);
count = 1;

% Extract vibration data of 50 files
while hasdata(hsbearing)
    data = read(hsbearing);
    data2add = table;

    % Get the vibration signal
    v = data.vibration{1};
    vibration_data(:,count) = v;

    % Count untill 50 days
    day = day + 1;

    count = count+1;
end
```

3. Data Processing: Feature Extraction & Selection

3-1. Extract Detectivity

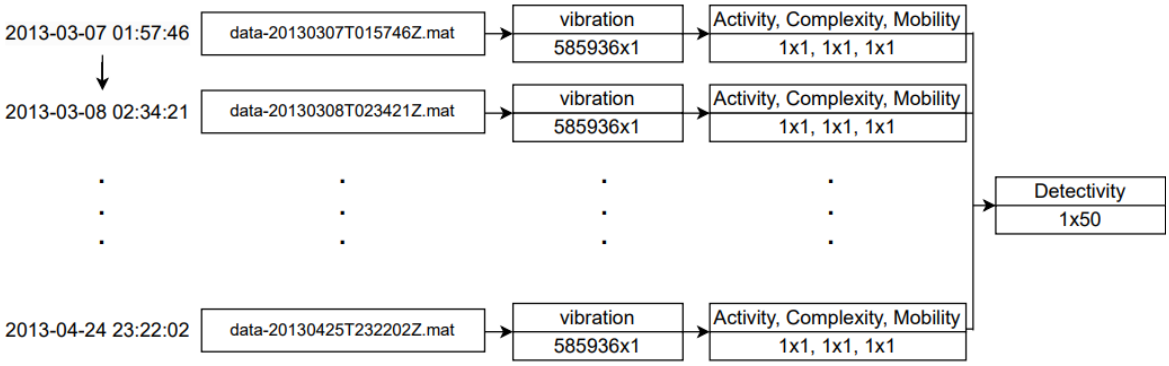

Figure 5. Extracting 1x50 Detectivity from Vibration Dataset

In our project, detectivity and Traditional features should be extracted to compare usefulness of detectivity as a RUL model feature. At first, we extracted activity, mobility, and complexity from 585,936 data of each 50 files. After calculating each 50 numbers of three parameters, we calculated detectivity. As a results of this, we could make 50 numbers of detectivity.

```
% Update DataVariables to include detectivity
hsbearing.DataVariables = [hsbearing.DataVariables; "detectivity"];

hsbearing.SelectedVariables = "vibration";
reset(hsbearing);
```

```
% Total number of detectivity calculations
num_cycles = 50;

% Initialize an array to store 50 four parameter values
activities = zeros(1, num_cycles);
mobilities = zeros(1, num_cycles);
complexities = zeros(1, num_cycles);
detectivitiy = zeros(1, num_cycles);

% Repeat 50 times to extract detectivity
for cycle_index = 1:num_cycles
    if ~hasdata(hsbearing)
        reset(hsbearing); % If no more data, reset to read again
    end

    features = table;

    % Extract 50 numbers of activity, mobility, and complexity
    for j = 1:num_cycles
        % Calculate activity
        v = vibration_data(:,j);
        activities(j) = var(v);

        % Calculate mobility
        first_derivative = diff(v);
        mobilities(j) = sqrt(var(first_derivative) / activities(j));

        % Calculate complexity
        second_derivative = diff(first_derivative);
        complexities(j) = sqrt(var(second_derivative) / var(first_derivative))/
sqrt(var(first_derivative) / var(v));
    end

    % Reference values
    A_ref = mean(activities);
    M_ref = mean(mobilities);
    C_ref = mean(complexities);

    % Calculate dB values of each three parameters
    activity_j_dB = 10 * log10(activities / A_ref);
    mobility_j_dB = 10 * log10(mobilities / M_ref);
    complexity_j_dB = 10 * log10(complexities / C_ref);

    % Calculate detectivity
    detectivity = activity_j_dB - mobility_j_dB + complexity_j_dB;
end

% Store all 50 detectivity values in the features table
features.detectivity = detectivity
```

Detectivity Dataset								
1	2	3	4	...	47	48	49	50
-0.3775	-0.6092	-0.4759	-0.8803	...	0.1405	0.6789	2.6225	1.9205

Table 1. 1x50 Detectivity Dataset

After extracting detectivity, combine this feature with date.

```
% Write the derived features to the corresponding file in hsbearing
data = read(hsbearing);
writeToLastMemberRead(hsbearing, features);

% Select the "Date" variable from hsbearing
hsbearing.SelectedVariables = ["Date"];
reset(hsbearing);

% Gather the date data
dateTable = gather(tall(hsbearing));
% Create a table combining Date and detectivity
combinedTable = table(dateTable.Date, features.detectivity', 'VariableNames',
{'Date', 'Detectivity'});
featureTable = table2timetable(combinedTable)
```

featureTable = *50×1 timetable*

	Date	Detectivity
1	2013-03-07 01:57:46	-0.3775
2	2013-03-08 02:34:21	-0.6092
3	2013-03-09 02:33:43	-0.4759
4	2013-03-10 03:01:02	-0.8803
5	2013-03-11 03:00:24	-0.8404
6	2013-03-12 06:17:10	-0.6599
7	2013-03-13 06:34:04	-1.0142
8	2013-03-14 06:50:41	-1.5716
9	2013-03-15 06:50:03	-0.8542

Table 2. TimeTable of Date and Detectivity

3-2. Plot Detectivity

Detectivity shows upward trend. But, there are many fluctuation of going up and down. This fluctuation would make bad prediction performance, because smooth trend of feature is necessary to get exact prediction.

```
% Plot the calculated detectivity values
figure;
plot(featureTable.Detectivity, '-o');
title('Detectivity Plot');
xlabel('Sample Number');
ylabel('Detectivity (dB)');
grid on;
```

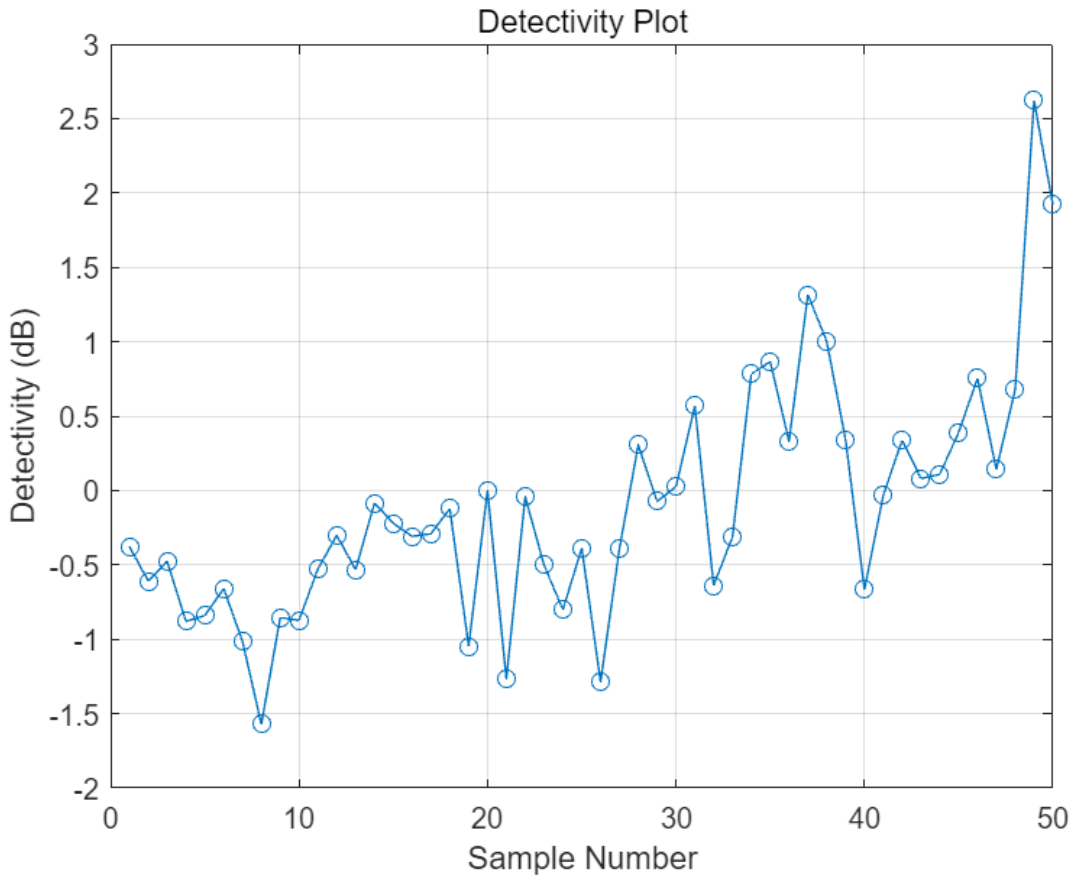

Figure 6. Graph of Detectivity

To make detectivity more smoothly, we applied smooth filter to detectivity signal.

```
variableNames = featureTable.Properties.VariableNames;
featureTableSmooth = varfun(@(x) movmean(x, [5 0]), featureTable);
featureTableSmooth.Properties.VariableNames = variableNames;
figure
hold on
plot(featureTable.Date, featureTable.Detectivity)
plot(featureTableSmooth.Date, featureTableSmooth.Detectivity)
hold off
xlabel('Time')
```

```
ylabel('Feature Value')
legend('Before smoothing', 'After smoothing')
title('Detectivity')
```

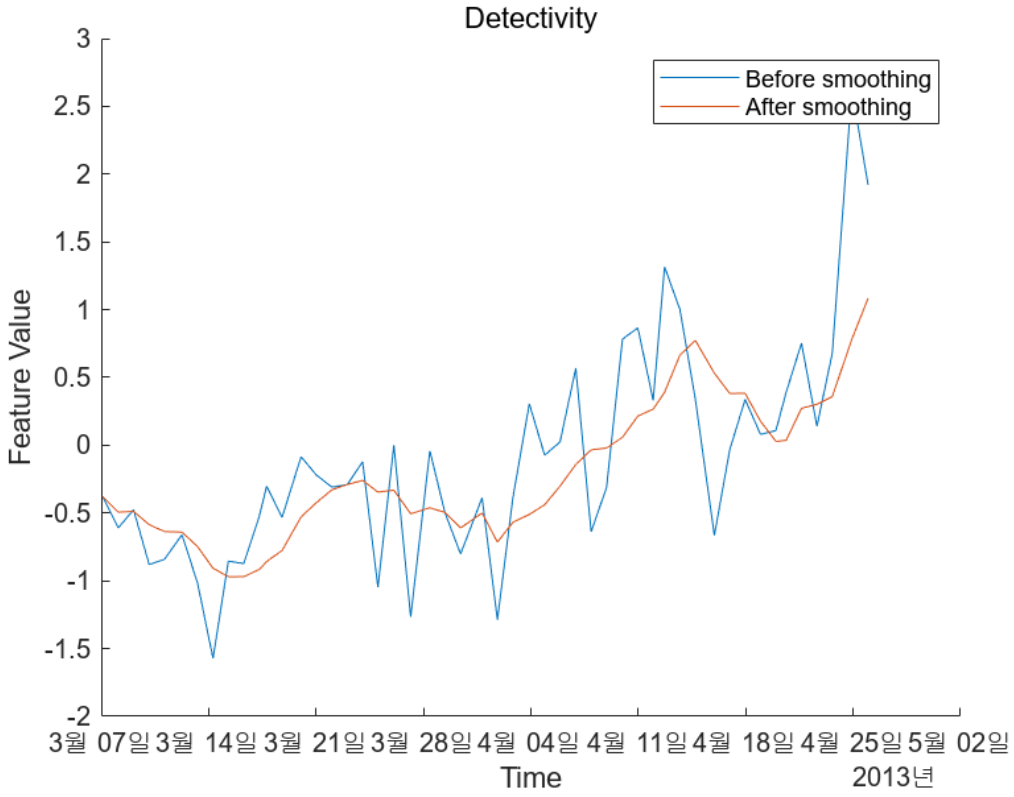

Figure 7. Graph of Smoothing Detectivity

3-3. Extract Traditional Features

To compare detectivity with Traditional features, we extracted time domain and frequency domain's features: Kurtosis, ShapeFactor, Mean, MarginFactor, SKStd etc.

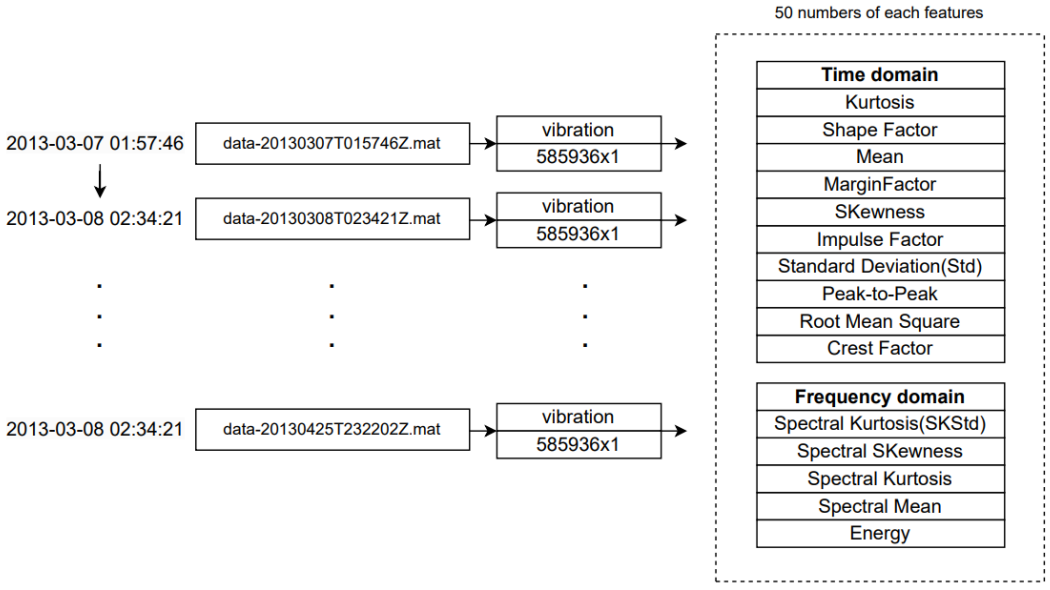

Figure 8.Extracting Traditional Features from Vibration Dataset

At the same time, we applied smoothing filter to Traditional features.

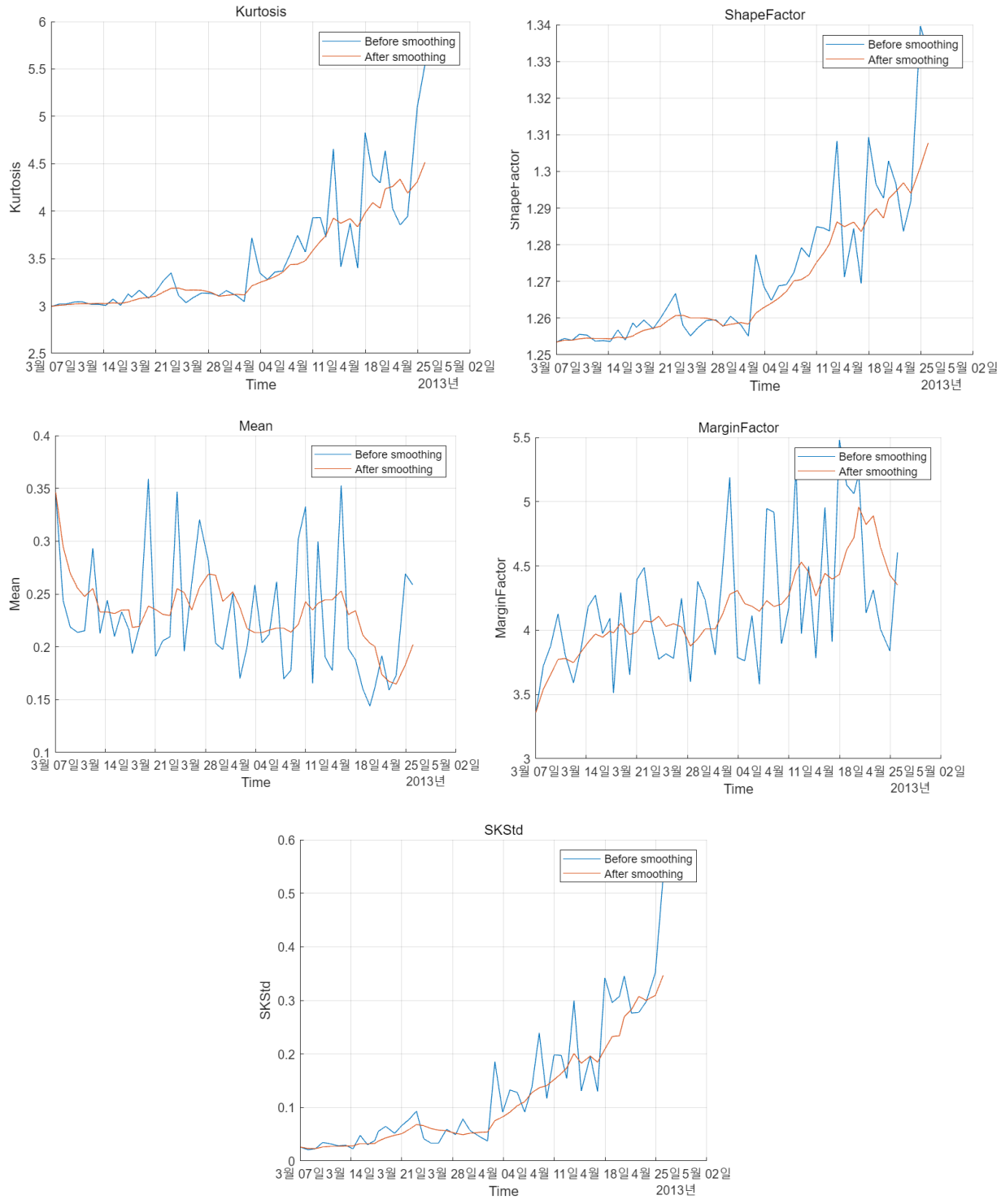

Figure 9. Smoothing Traditional Features

3-4. Select Traditional Features

After applying smoothing fiter,we calculated monotonicity to select health indicator features. We set threshold of monotonicity value as 0.3: Kurtosis, ShapeFactor, Mean, MarginFactor, and SKStd are over 0.3.

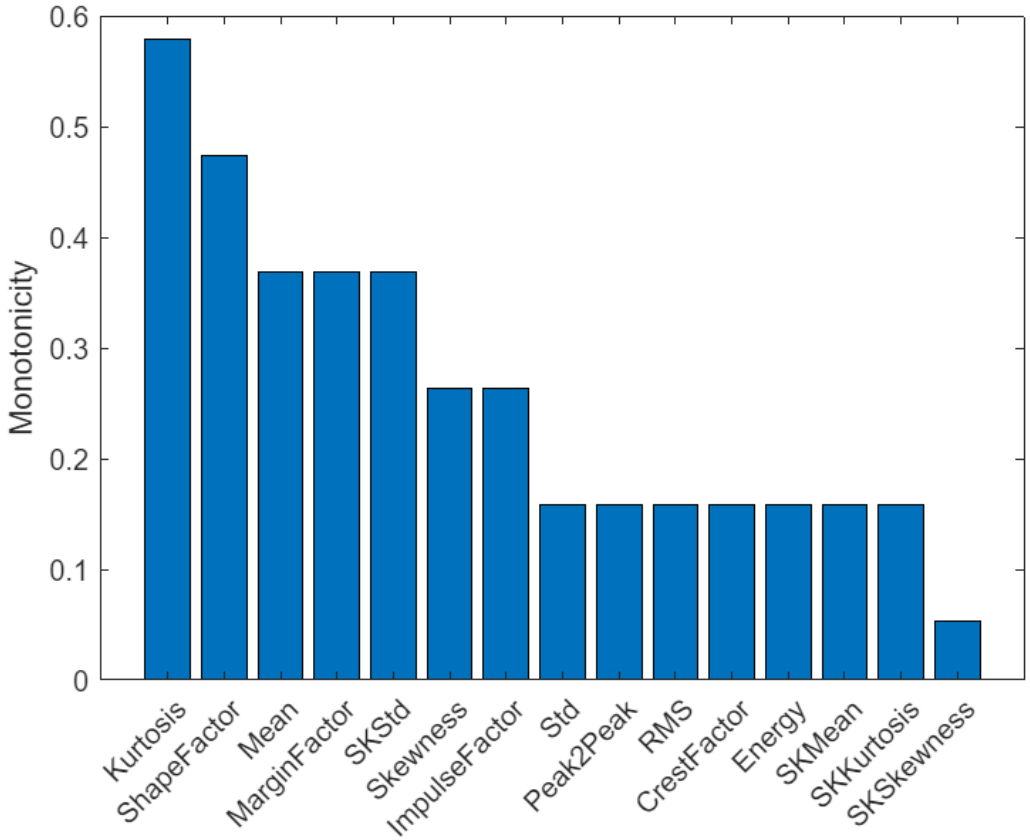

Figure 10. Monotonicity of Traditional Features

After selecting five features, we extracted PCA value to reduce the channel of features and organize all features into one feature.

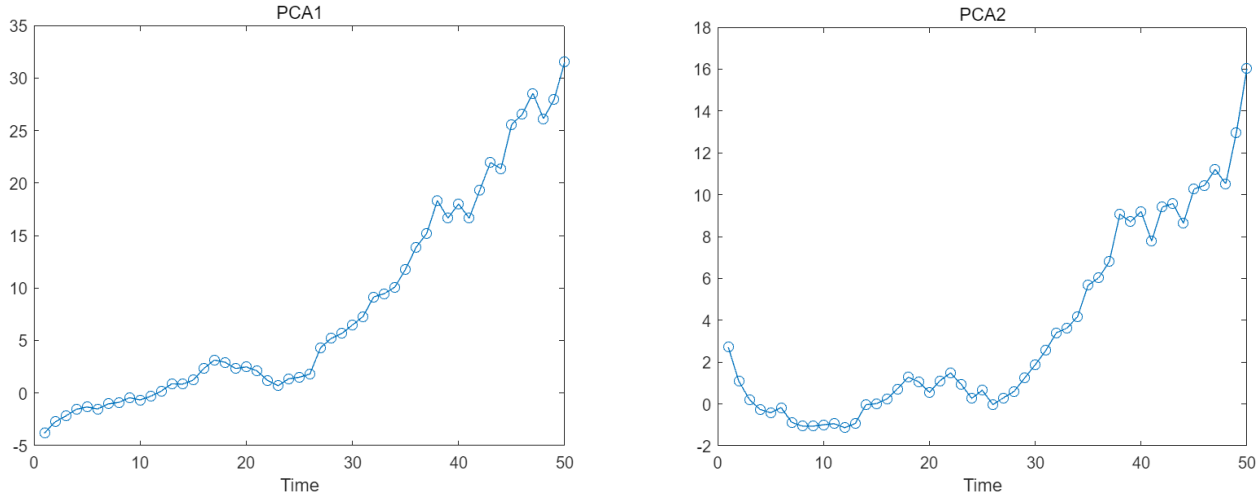

Figure 11. Two PCA results of five selected features: Kurtosis, ShapeFactor, Mean, MarginFactor, and SKStd

Because PCA1 shows more high upward monotonicity, we decided to use PCA1 as a final feature.

4. RUL Prediction & Evaluation

4-1. Build Model

Now, we got two tpyes of health indicator features: detectivity, and PCA. To train RUL prediction model with these features, we divided dataset into 2:5, train dataset:test dataset. In this chapter, we dealt with only detectivity code, and used pre-extracted Traditional features' result.

```
% Divide Detectivity dataset into train and test.
breaktime = datetime(2013, 3, 27);
breakpoint = find(featureTableSmooth.Date < breaktime, 1, 'last');
trainData = featureTableSmooth(1:breakpoint, :);
```

To predcit RUL, we used fitting model: Exponential degradation mode(EDM). [3]

$$h(t) = \phi + \theta \exp\left(\beta t + \epsilon - \frac{\sigma^2}{2}\right)$$

EDM shows the degradation level of system by using the upwarding fitting graph. θ and β are charge of the volatility of model. ϕ and ϵ mean the axis position of fitting model and σ^2 contains gaussian noise. Every trainning of fitting, this model find the optimum value to predict true RUL.

```
% healthIndicator of detectivity
healthIndicator = featureTableSmooth{:,:};
healthIndicator = healthIndicator - healthIndicator(1);

% Threshold for failure
threshold = healthIndicator(end);

% Fitting model
mdl = exponentialDegradationModel(...
    'Theta', 1, ...
    'ThetaVariance', 1e6, ...
    'Beta', 1, ...
    'BetaVariance', 1e6, ...
    'Phi', -1, ...
    'NoiseVariance', (0.1*threshold/(threshold + 1))^2, ...
    'SlopeDetectionLevel', 0.05);
```

4-2. Predict RUL

By using matlab function [3], we predicted RUL and evaluated the prediction results.

```
% Detectivity
% Keep records at each iteration
totalDay = length(healthIndicator) - 1;
estRULs = zeros(totalDay, 1);
trueRULs = zeros(totalDay, 1);
```

```
CIRULs = zeros(totalDay, 2);
pdfRULs = cell(totalDay, 1);

% Create figures and axes for plot updating
figure
ax1 = subplot(2, 1, 1);
ax2 = subplot(2, 1, 2);
for currentDay = 1:totalDay

    % Update model parameter posterior distribution
    update(mdl, [currentDay healthIndicator(currentDay,:)])

    % Predict Remaining Useful Life
    [estRUL, CIRUL, pdfRUL] = predictRUL(mdl, ...
                                         [currentDay healthIndicator(currentDay)],
...
                                         threshold);
    trueRUL = totalDay - currentDay + 1;

    % Updating RUL distribution plot
    helperPlotTrend(ax1, currentDay, healthIndicator, mdl, threshold, timeUnit);
    helperPlotRUL(ax2, trueRUL, estRUL, CIRUL, pdfRUL, timeUnit)

    % Keep prediction results
    estRULs(currentDay) = estRUL;
    trueRULs(currentDay) = trueRUL;
    CIRULs(currentDay, :) = CIRUL;
    pdfRULs{currentDay} = pdfRUL;

    % Pause 0.1 seconds to make the animation visible
    pause(0.1)
end
```

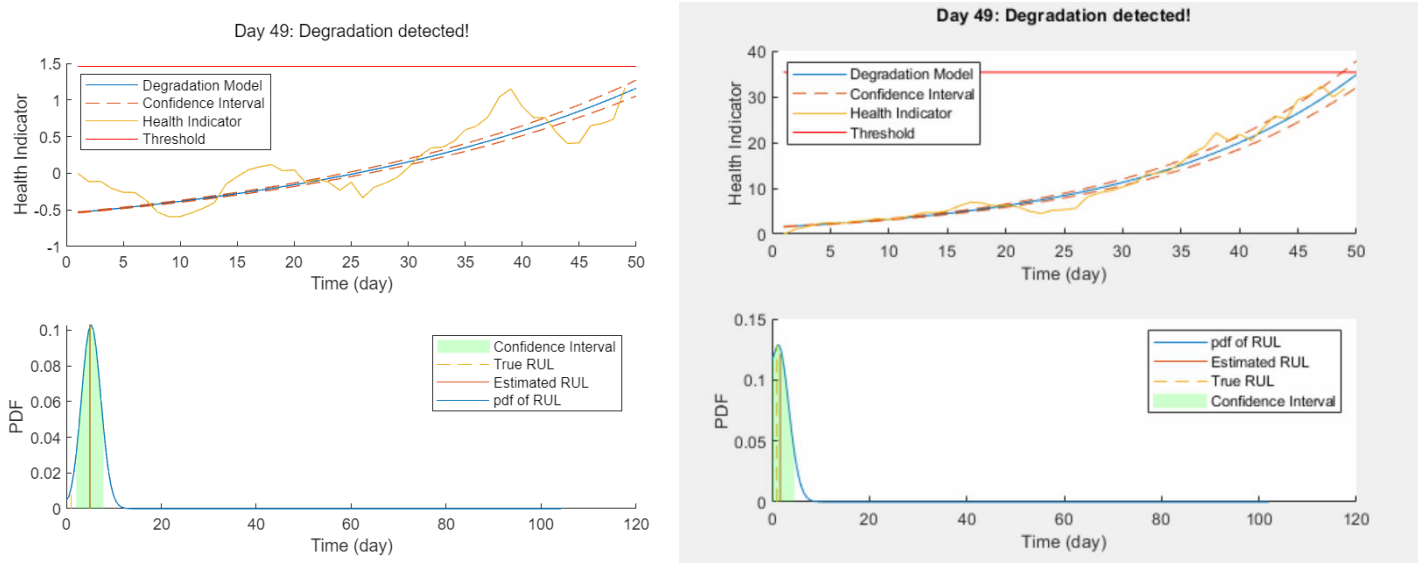

Figure 12. RUL Prediction Results: Left is from Detectivity, Right is from Ohter Features [3]

4-3. RUL Evaluation

α-λ plot is visual RUL diagnosis for prediction evaluation. Blue line is true RUL, and the blue area around the true RUL is the allowable range of change(20% possibility of discorrect). Red lines are predicted RUL after detecting degradation. The red area around the prediction RUL is allowable range of change(20% possibility of discorrect). By checking the shared area between true and prediction, we can know that when this model predict correct RUL.

As broad as possibile between true and predction RUL area, the model can be determined as high performance of RUL prediction model. When comapring with fitting model from detectivity and from Traditional features, it seems that two types of model show similar performance. However, we can compare performance of RUL prediction by chekcing the probability of RUL.

```
alpha = 0.2;
detectTime = mdl.SlopeDetectionInstant;
prob = helperAlphaLambdaPlot(alpha, trueRULs, estRULs, CIRULs, ...
    pdfRULs, detectTime, breakpoint, timeUnit);
title('\alpha-\lambda Plot')
```

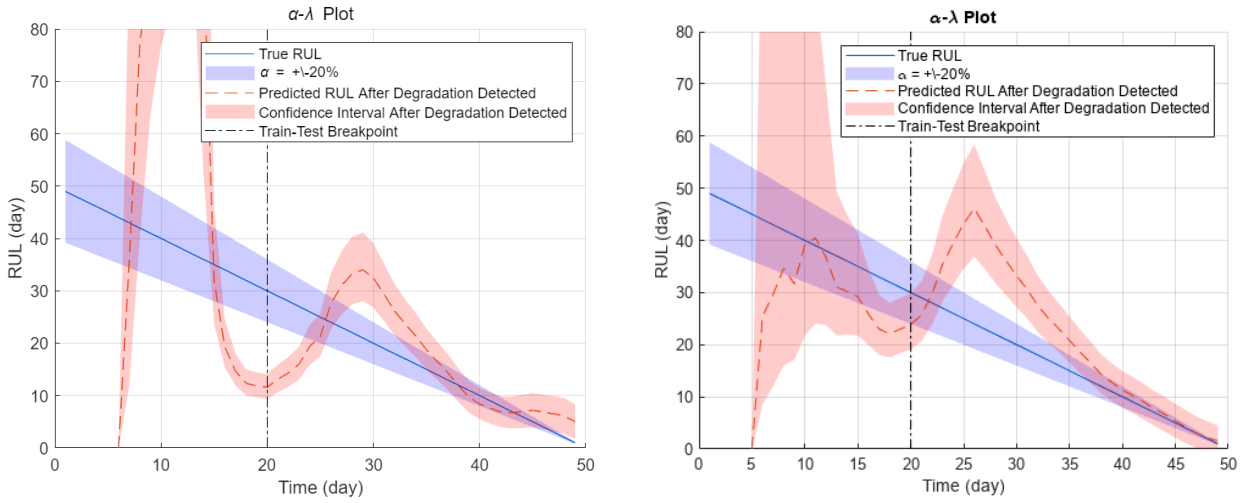

Figure 13. α-λ plot, Left is from Detectivity, Right is from Ohter Features [2]

We extract probability of two cases: Detectivity and Traditional Features.

```
figure
t = 1:totalDay;
hold on
plot(t, prob)
plot([breakpoint breakpoint], [0 1], 'k-.')
hold off
xlabel(['Time (' timeUnit ')'])
ylabel('Probability')
legend('Probability of predicted RUL within \alpha bound', 'Train-Test Breakpoint')
title(['Probability within \alpha bound, \alpha = ' num2str(alpha*100) '%'])
```

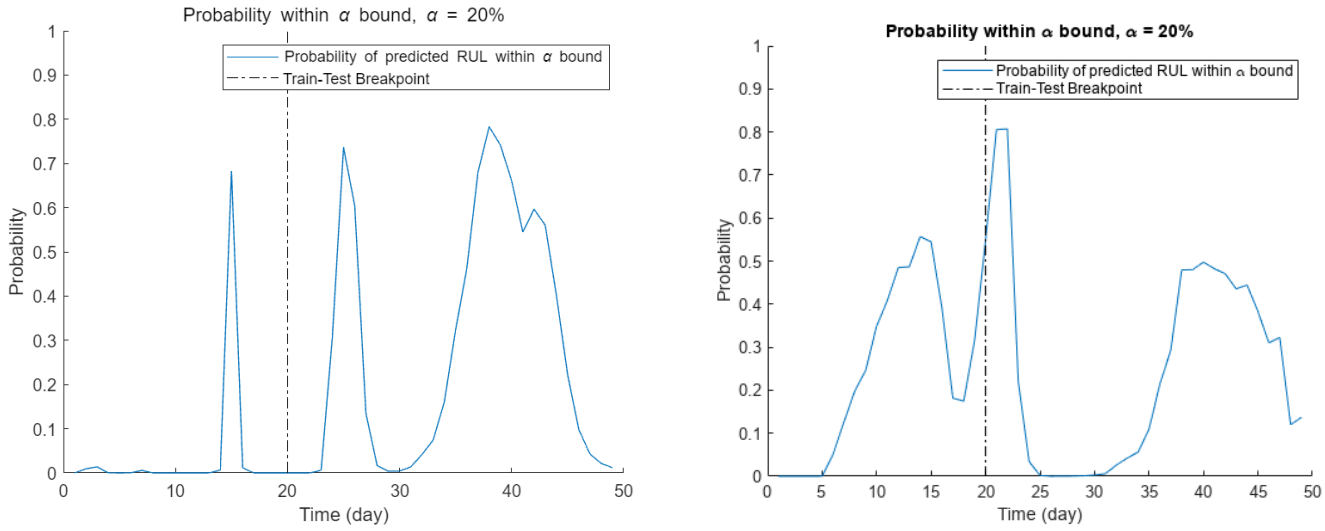

Figure 14. Probability graph, Left is from Detectivity, Right is from Traditional Features [2]

Probability graphs show different performance results of two methods. The first graph is from detectivity's RUL prediction. According to this graph, detectivity could not predict date of breakpoint. In contrast of this, probabilites of RUL prediction from Traditional features predict well date of breakpoint. In addition to this, as closing to the end of date, detectivity became not able to predict RUL.

This result show us that detectivity could not work better than Traditional features as a RUL prediction material. This low quality of RUL predction may occure from high volatility of detectivity. Although PCA from Traditional features was increased relatively steadily, detectivity was a lot of fluctuation. This unstability would have caused a relatively low performance prepared with Traditional features.

However, the simple method of extracting detectivity from raw data can be a strong point compared with method of using Traditional features. Although detectivity did not show good performance in this case1, we could check the possibility of detectivity as a feature for RUL prediction.

5. Conclusion

We tried to predict RUL by using machine learning with WindTurbineProgonosis Dataset. Because traditional method of extracting & selecting features takes a long time, we planned to use Detectivity as a health indicator. And to comfirm Detectivity's availability as a health indicator for RUL prediction, we compared Detectivity's RUL prediction performance with traditional features's performance.

The criteria of evaluating RUL prediction performance was that how much the predicted RUL and real RUL were sharing same area. As date becomes to be closed to 0 days of RUL, probability of both method showed low accuracy of RUL prediction. However, RUL prediction of Detectivity showed almost zero probability. In addition to this, at the fault date, Detectivity could not predict ball bearing fault. These results mean Detectivity can not work as health indicator for RUL prediction.

We concluded that using Detectivity would not guarantee high performance of RUL prediction. We thought that this low performance was happened from the limitation of machine learning. Although Detectivity showed the upward trend of ball bearing signal, because flucuation of Detectivity graph caused non-linearity, Detectivity

could not show high accuracy of RUL prediction with machin learning. In this reason, we proposed to use deep learning model as a RUL prediction model. With deep learning model, we would able to overcome curve fitting model's limitation for non-linearity.

References

[1] Cocconcelli, M., Strozzi, M., Camargo Molano, J. C., & Rubini, R. (2022). Detectivity: A combination of Hjorth's parameters for condition monitoring of ball bearings. *Mechanical Systems and Signal Processing, 164*, 108247. https://doi.org/10.1016/j.ymssp.2021.108247

[2] Saxena, A., Celaya, J., Balaban, E., Goebel, K., Saha, B., Saha, S., & Schwabacher, M. (2008). Metrics for evaluating performance of prognostic techniques. 2008 International Conference on Prognostics and Health Management (PHM), 1-8. https://doi.org/10.1109/PHM.2008.4711436

[3] MathWorks. (n.d.). Wind Turbine High-Speed Bearing Prognosis. MathWorks. Retrieved October 28, 2024, from https://kr.mathworks.com/help/predmaint/ug/wind-turbine-high-speed-bearing-prognosis.html