



 POLITECNICO DI MILANO



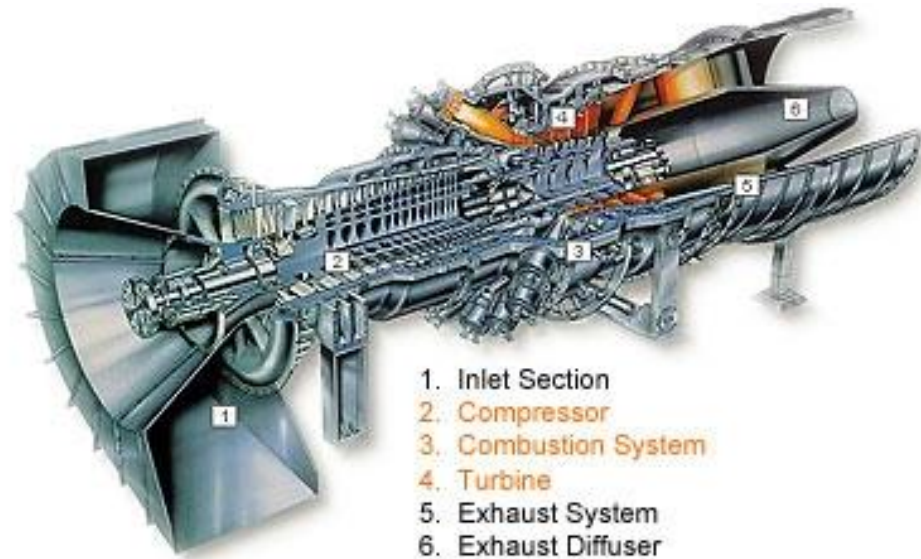
Exercises Session on Fault Detection

Giovanni Floreale
giovanni.floreale@polimi.it



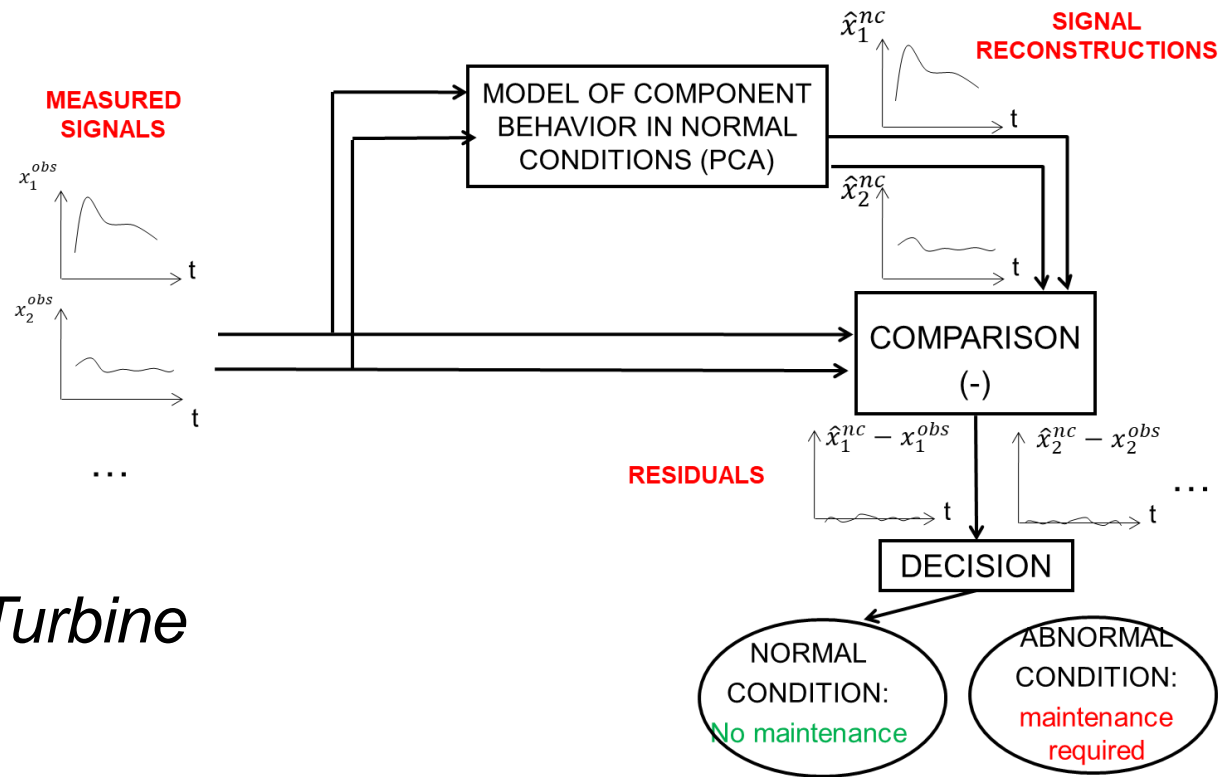
Exercise 1

Component: Gas Turbine



Courtesy of Siemens Westinghouse

Exercise 1



Method: PCA

Component: Gas Turbine

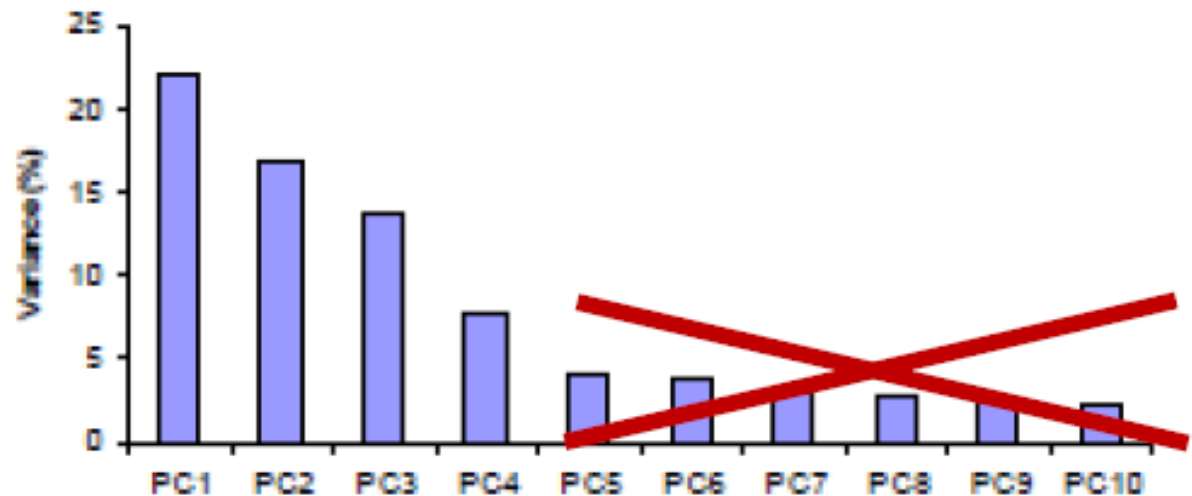


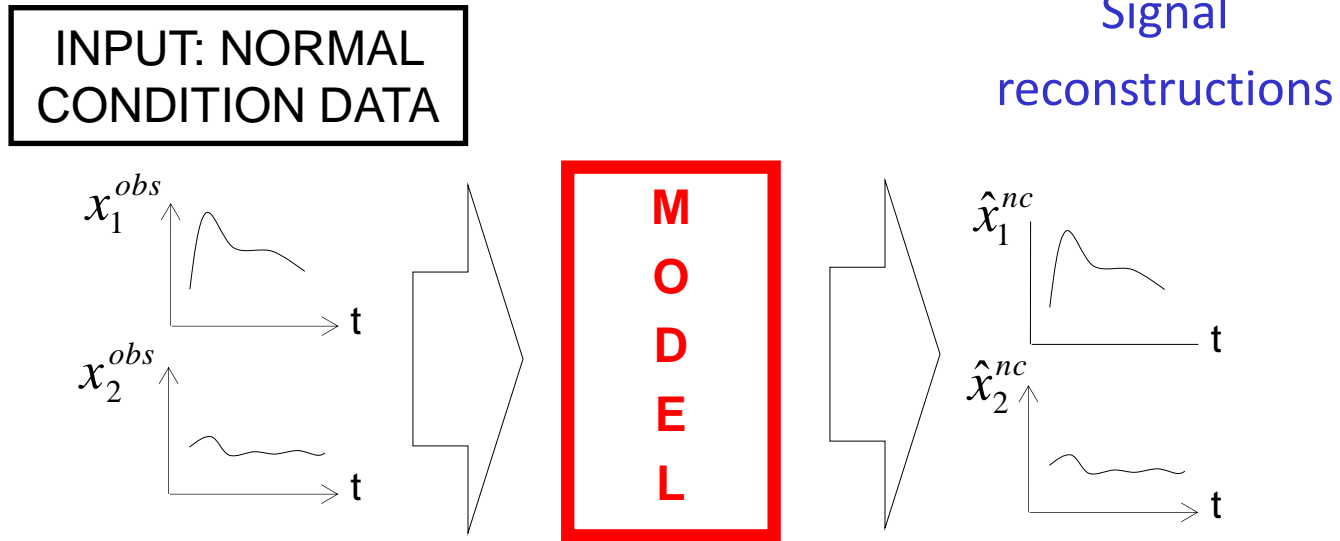
Method: PCA

Setting the model parameter:

- $p = \text{number of principal components}$
- Objective: ACCURATE and ROBUST reconstruction model

$$\%Var(PC_i) = \frac{\lambda_i}{\sum_{i=1, \dots, n} \lambda_i}$$





- **Accurate** reconstruction model if:

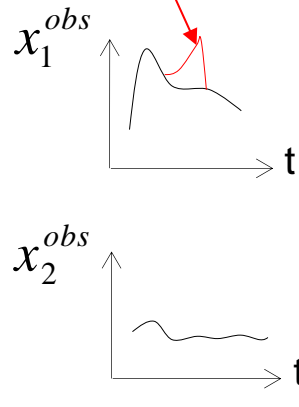
$$\vec{\hat{x}}^{nc} \cong \vec{x}^{obs}$$

- Metric to measure accuracy: Root Mean Square Error (*RMSE*):

$$RMSE = \frac{\sum_{j=1}^n \sqrt{\frac{\sum_{t=1}^{N_{valid}} [\hat{x}^{nc}(t,j) - x^{obs}(t,j)]^2}{N_{valid}}}}{n}$$

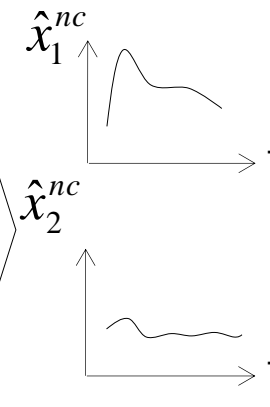


INPUT: **ABNORMAL
CONDITION DATA**
(Simulated)



**M
O
D
E
L**

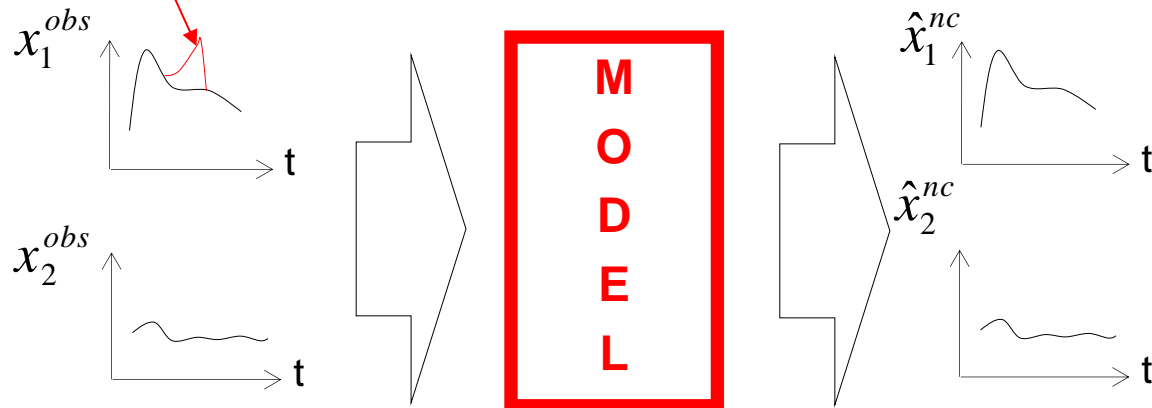
Signal
reconstructions



- Robust reconstruction if: $\vec{\hat{x}}^{nc} \cong \vec{x}^{obs-nc}$



INPUT: **ABNORMAL
CONDITION DATA**
(Simulated)



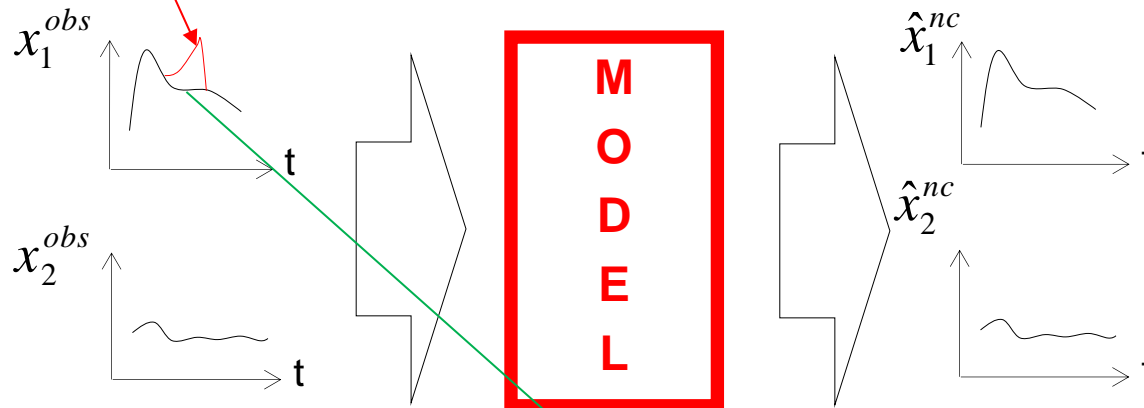
- **Robust** reconstruction if: $\vec{\hat{x}}^{nc} \cong \vec{x}^{obs-nc}$
- Error to measure Robustness:

$$\text{Robustness} = \frac{\sum_{t=1}^{N_{valid}} |\vec{x}_1^{obs-nc}(t) - \hat{x}_1^{nc}(t)|^2}{N_{valid}}$$



INPUT: **ABNORMAL
CONDITION DATA**
(Simulated)

Signal
reconstructions



- **Robust** reconstruction if: $\vec{\hat{x}}^{nc} \cong \vec{x}^{obs-nc}$

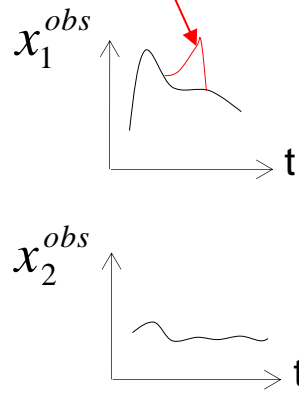
- Error to measure Robustness:

Normal condition data
(before anomaly simulation)

$$\text{Robustness} = \frac{\sum_{t=1}^{N_{valid}} \left| \vec{x}_1^{obs-nc}(t) - \hat{x}_1^{nc}(t) \right|^2}{N_{valid}}$$

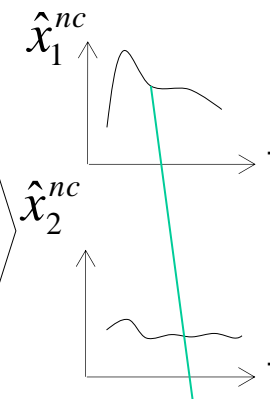


INPUT: **ABNORMAL
CONDITION DATA**
(Simulated)



**M
O
D
E
L**

Signal
reconstructions



- **Robust** reconstruction if: $\vec{\hat{x}}^{nc} \cong \vec{x}^{obs-nc}$
- Error to measure Robustness (only on the signal containing the anomaly):

Reconstruction of the anomaly

$$\text{Robustness} = \frac{\sum_{t=1}^{N_{valid}} \left| \vec{x}_1^{obs-nc}(t) - \hat{x}_1^{nc}(t) \right|^2}{N_{valid}}$$



Create a Colab file in your drive in the same folder where data are



```
from google.colab import drive
drive.mount("/content/drive") #mount the drive
```

```
[ ] import os
os.chdir("/content/drive/MyDrive/Fault Detection/Exercise 1") #set the current directory
```

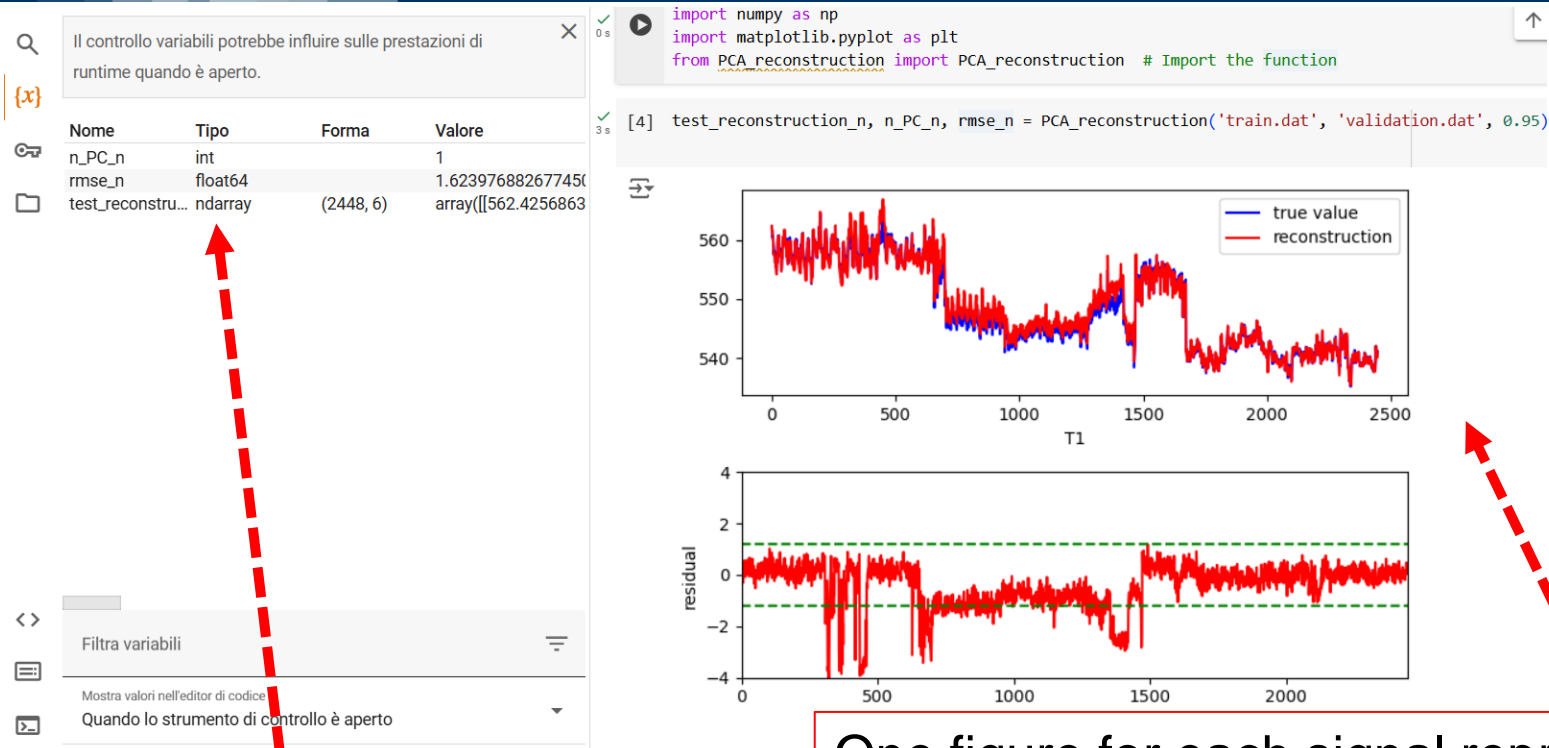
```
[ ] import numpy as np
import matplotlib.pyplot as plt
from PCA_reconstruction import PCA_reconstruction # Import the function
```

```
[ ] test_reconstruction_n, n_PC_n, rmse_n = PCA_reconstruction('train.dat', 'validation.dat', 0.95)
```

Import necessary functions and execute the code in one cell:

```
test_data_rec, n_PC, RMSE=PCA_reconstruction('train.dat', 'validation.dat', v)
```

v=minimum percentage of variance to be represented in the PCA space [e.g., 0.95]



- The reconstruction obtained using the PCA algorithm
- The number of principal components corresponding to the selected variance
- The RMSE



Consider the same data used in Exercise 1

1. Optimize the PCA reconstruction model to achieve:

- a. Accuracy
- b. Robustness

Hints:

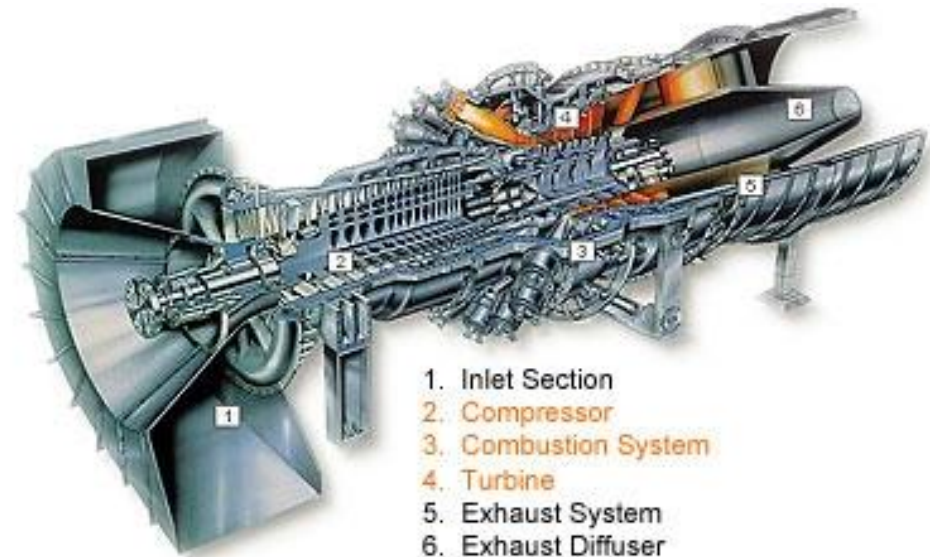
- Consider different numbers of Principal Components by trying different values for the parameter v =*minimal percentage of variance considered in the PCA space*
- Consider the root mean square error as a performance measure on test data under normal condition and the robustness metric on test data under simulated abnormal condition

```
[test_reconstruction,n_PC, RMSE]=PCA_reconstruction('train.dat','validation.dat',0.95)
```

* In python you must first import the function as: `from PCA_reconstruction import PCA_reconstruction`



Temperature location 1 (°C)
Temperature location 2 (°C)
Temperature location 3 (°C)
Temperature location 4 (°C)
Temperature location 5 (°C)
Temperature location 6 (°C)



1. Inlet Section
2. Compressor
3. Combustion System
4. Turbine
5. Exhaust System
6. Exhaust Diffuser

Courtesy of Siemens Westinghouse



train.dat → healthy condition data, 6 signals

validation.dat → healthy condition data, 6 signals

val_anomaly.dat → validation data modified with an anomaly on signal 1

validation_sim.dat → validation data modified with an anomaly on signal 6

test_1.dat, test_2.dat, test_3.dat, test_4.dat → test data of unknown health condition



```
from PCA_reconstruction import PCA_reconstruction
import numpy as np
import matplotlib.pyplot as plt

true_signal = np.loadtxt('validation.dat')
N, n = true_signal.shape

variance_levels = [0.9, 0.95, 0.98, 0.99, 0.995, 1.0]
n_PC = []
rmse = []

for v in variance_levels:
    test_reconstruction, n_pc, rmse_val = PCA_reconstruction('train.dat', 'validation.dat', v)
    n_PC.append(n_pc)
    rmse.append(rmse_val)
    plt.close('all')

plt.figure(1)
plt.plot(variance_levels, n_PC, '-o')
plt.xlabel('Percentage of Variance')
plt.ylabel('Number of Principal Components')
plt.legend(['Number of Principal Components'])
plt.grid(True)

plt.figure(2)
plt.plot(variance_levels, rmse, '-o')
plt.xlabel('Percentage of Variance')
plt.ylabel('RMSE')
plt.legend(['RMSE'])
plt.grid(True)

plt.figure(3)
plt.plot(n_PC, rmse, '-o')
plt.xlabel('Number of Principal Components')
plt.ylabel('RMSE')
plt.legend(['RMSE'])
plt.grid(True)
plt.show()
```

Load true signal (validation data)

Define the variance levels to test

Initialize lists to store results

Run PCA reconstruction for each variance level

Plot number of principal components vs. variance level

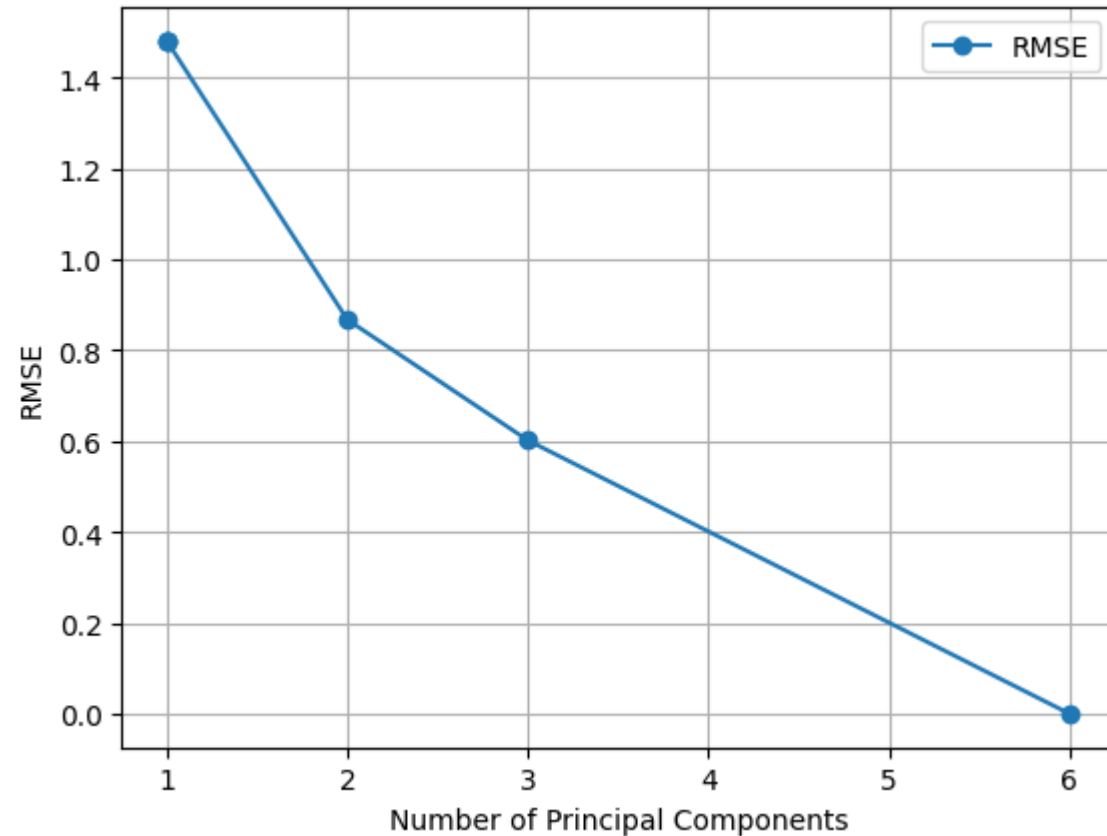
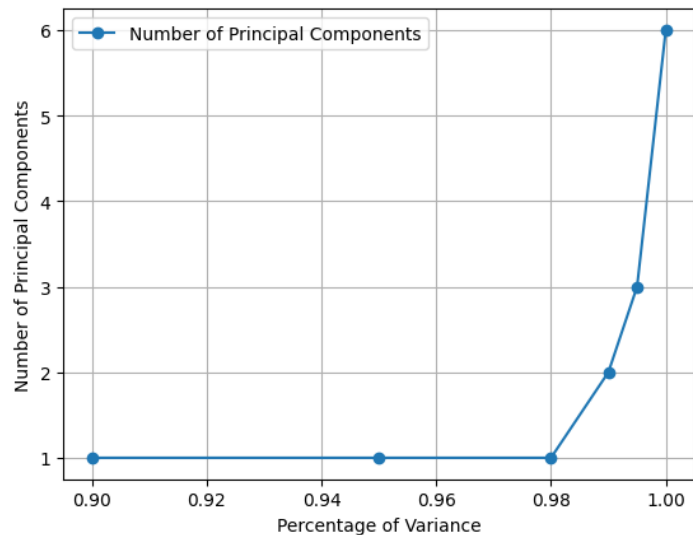
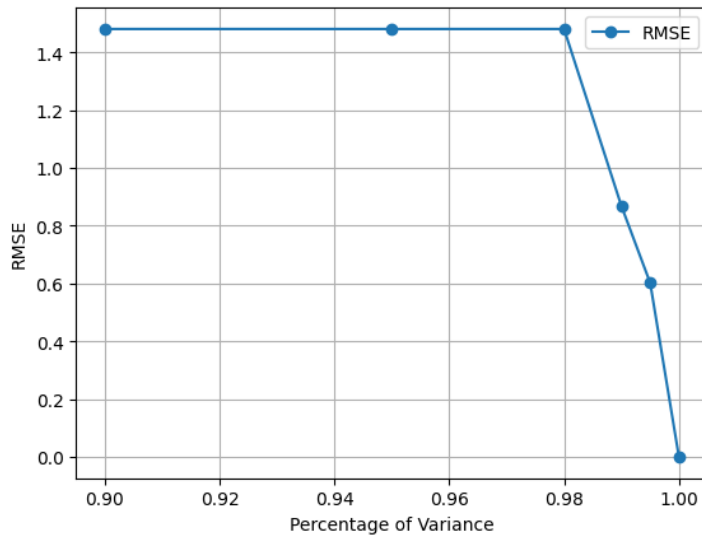
Plot RMSE vs. variance level

Plot RMSE vs. number of principal components



Optimal Choice of the number of Principal components (test for the accuracy)

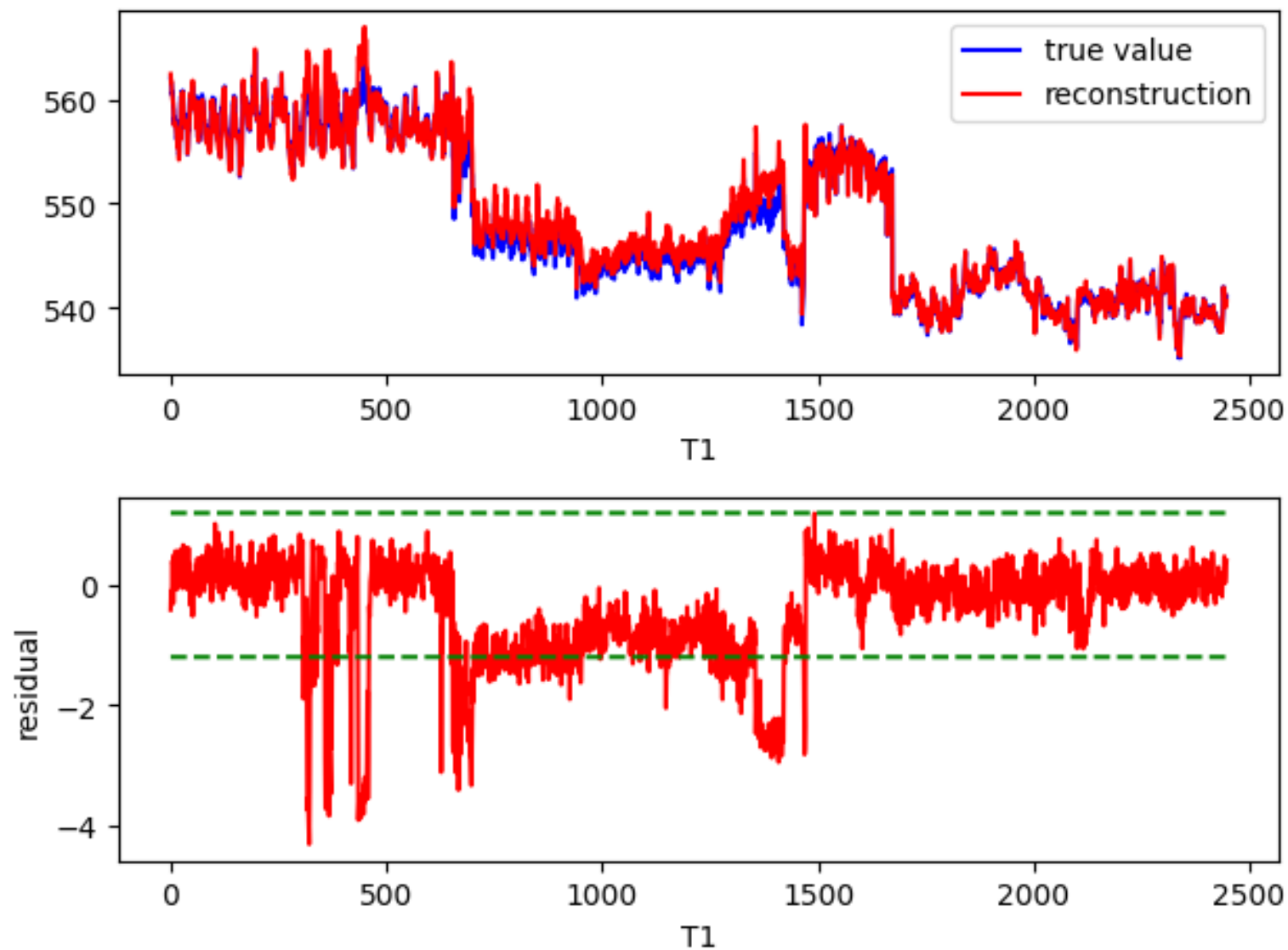
16

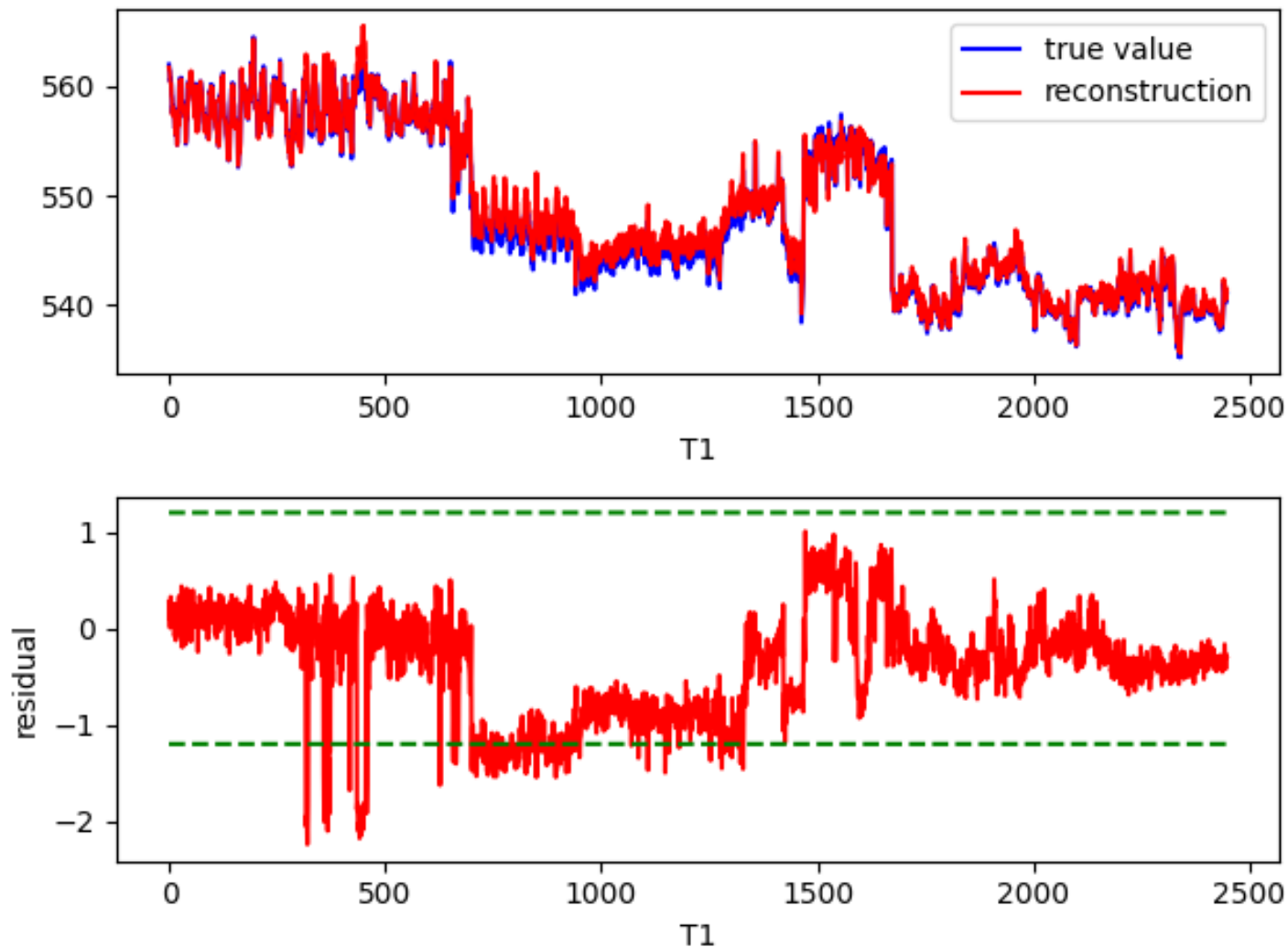


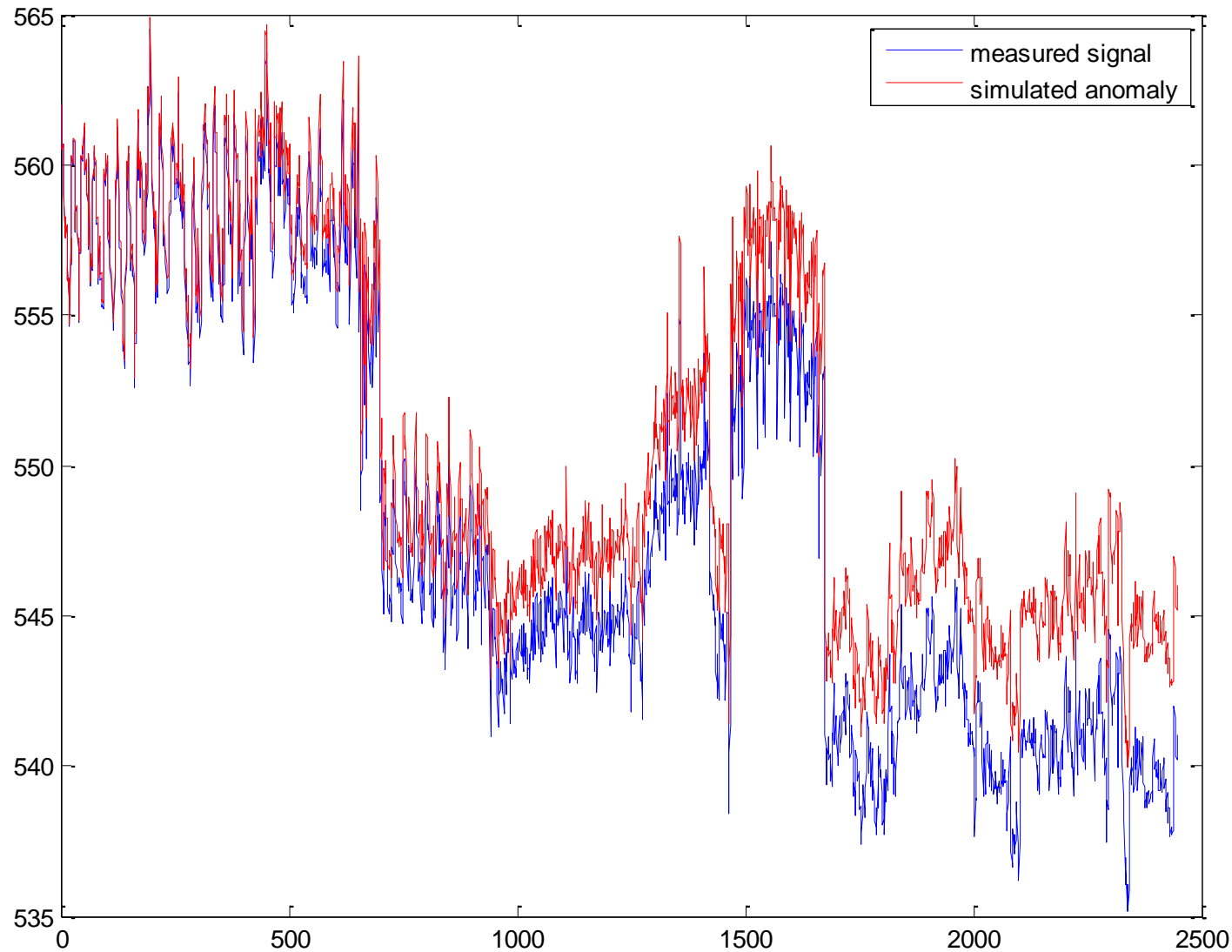


Reconstruction of the normal condition: 1 PC

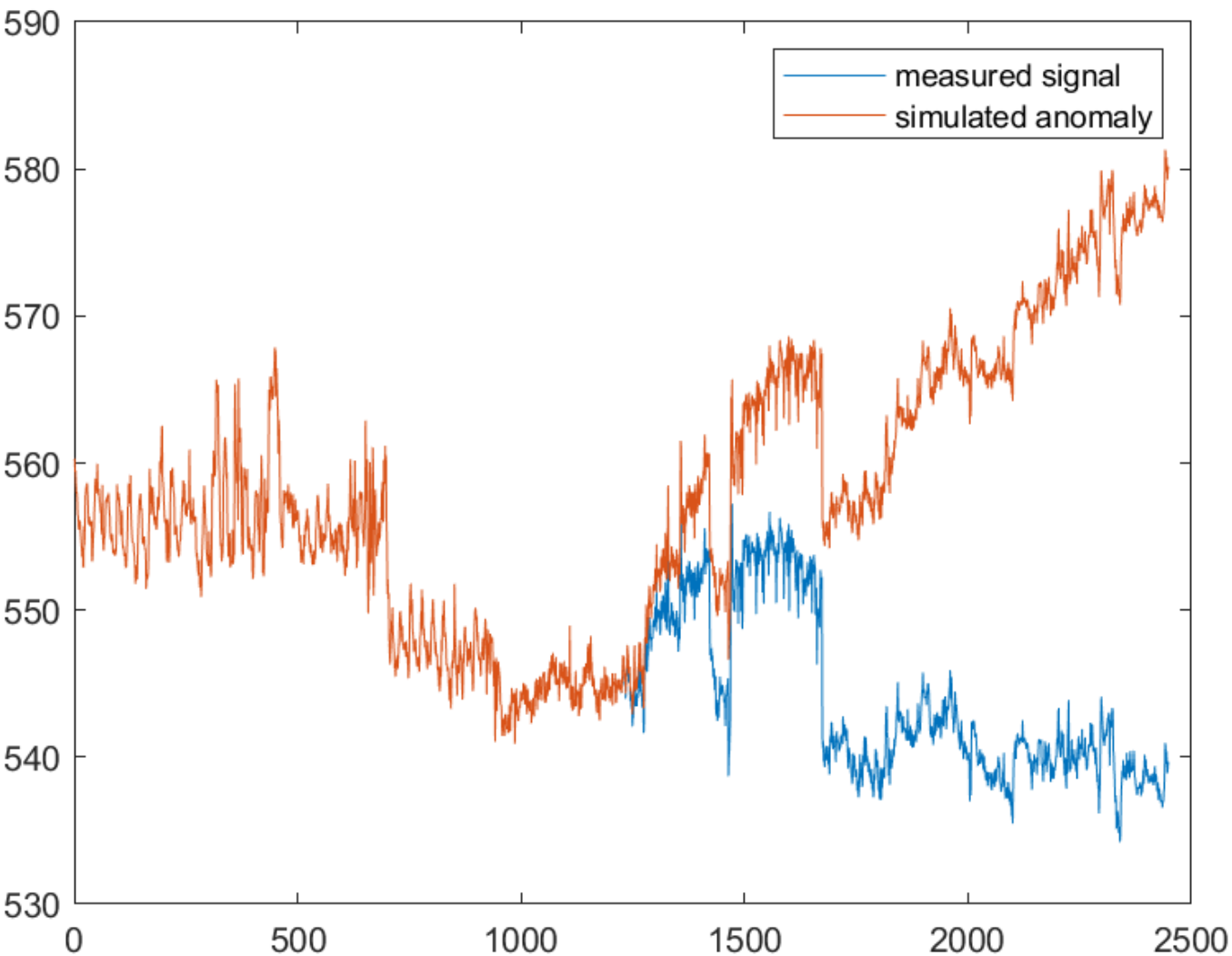
17







val_anomaly.dat
Signal 1



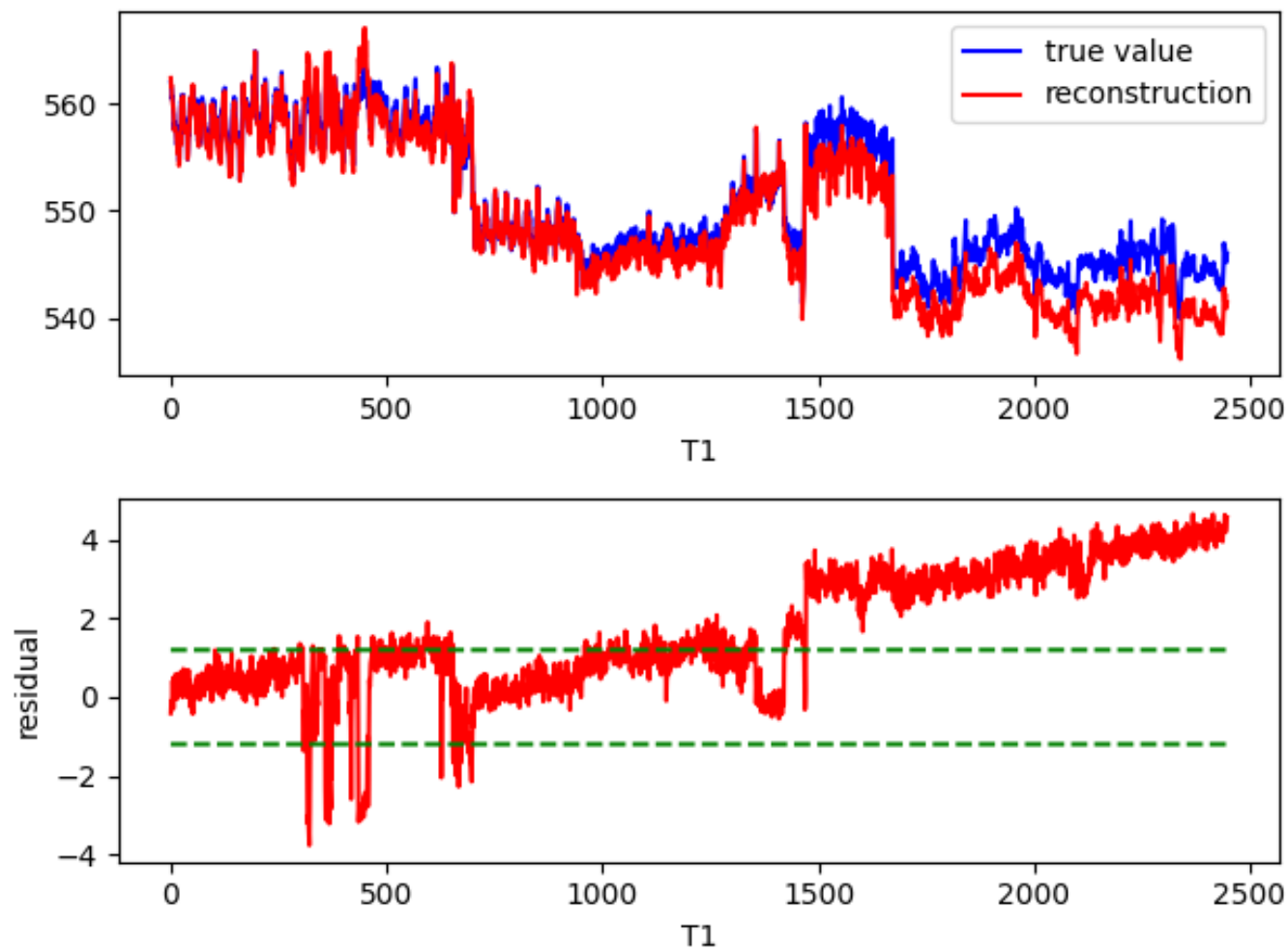
validation_sim.dat

Signal 6



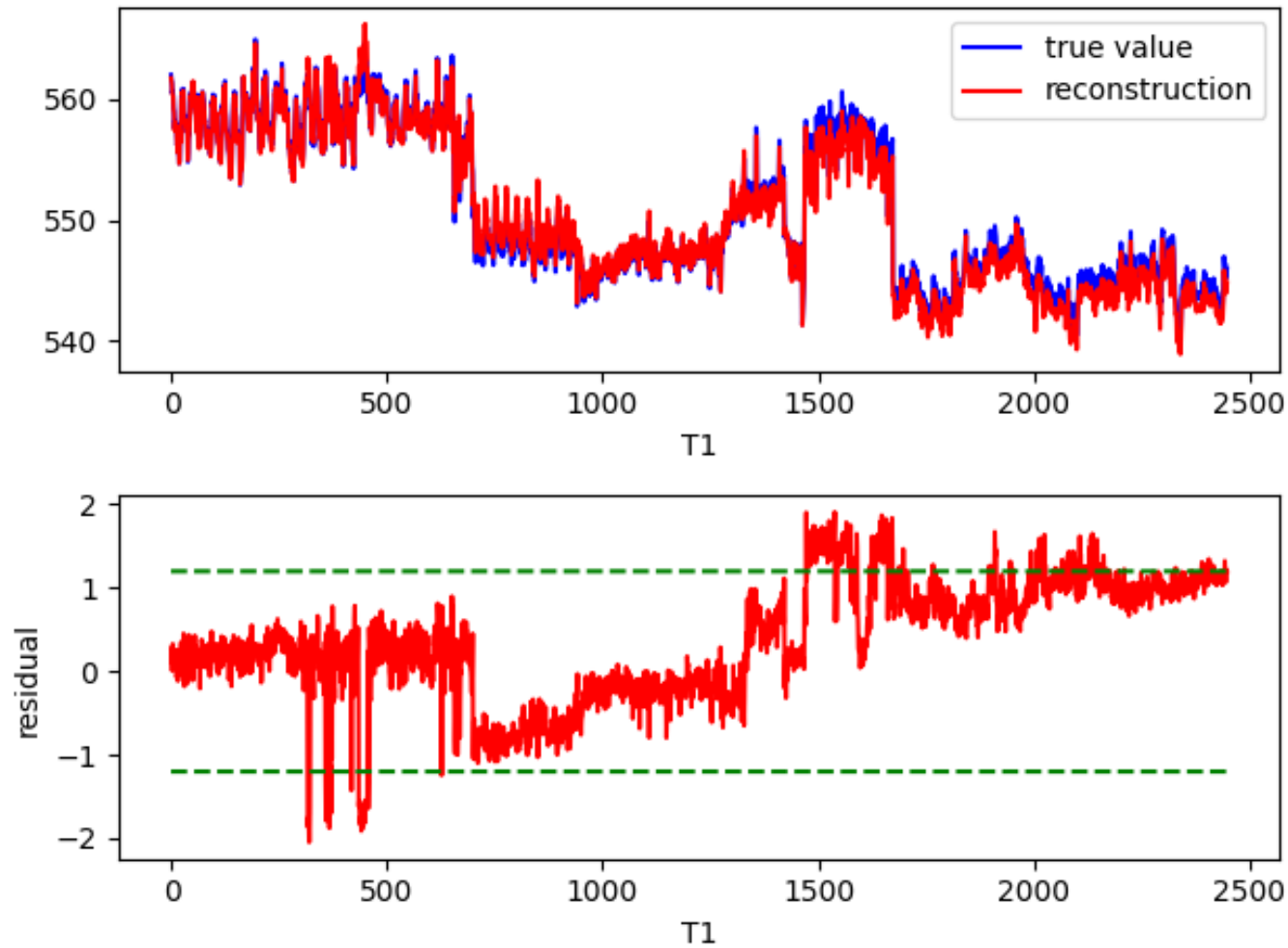
val_anomaly.dat

Signal 1





val_anomaly.dat
Signal 1





```
true_signal = np.loadtxt('validation.dat')
N, n = true_signal.shape
variance_levels = [0.9, 0.95, 0.98, 0.99, 0.995, 1.0]
rmse = []
rmse_sim = []
robustness = []
number_PC = []
for v in variance_levels:
    test_reconstruction_n, n_PC_n, rmse_n = PCA_reconstruction('train.dat', 'validation.dat', v)
    rmse.append(rmse_n)
    # test_reconstruction, n_PC, rmse_a = PCA_reconstruction('train.dat', 'val_anomaly.dat', v)
    test_reconstruction, n_PC, rmse_a = PCA_reconstruction('train.dat', 'validation_sim.dat', v)
    rmse_sim.append(rmse_sim)
    number_PC.append(n_PC)
    # robustness_val = np.sqrt(np.sum((test_reconstruction[:, 0] - true_signal[:, 0]) ** 2)) / N
    robustness_val = np.sqrt(np.sum((test_reconstruction[:, 5] - true_signal[:, 5]) ** 2)) / N
    robustness.append(robustness_val)

# Close any generated figures
plt.close('all')

plt.figure()
plt.plot(number_PC, rmse, marker='o', markerfacecolor='blue', label='Reconstruction Error')
plt.xlabel('Number of Principal Components')
plt.plot(number_PC, robustness, marker='o', markerfacecolor='red', label='Robustness')
plt.legend()
plt.show()
```

Load validation data

Define variance levels to test

Initialize lists to store results

Perform PCA reconstruction for each variance level

Perform PCA reconstruction on 'validation.dat'

Perform PCA reconstruction on 'validation_sim.dat'

Calculate robustness metric

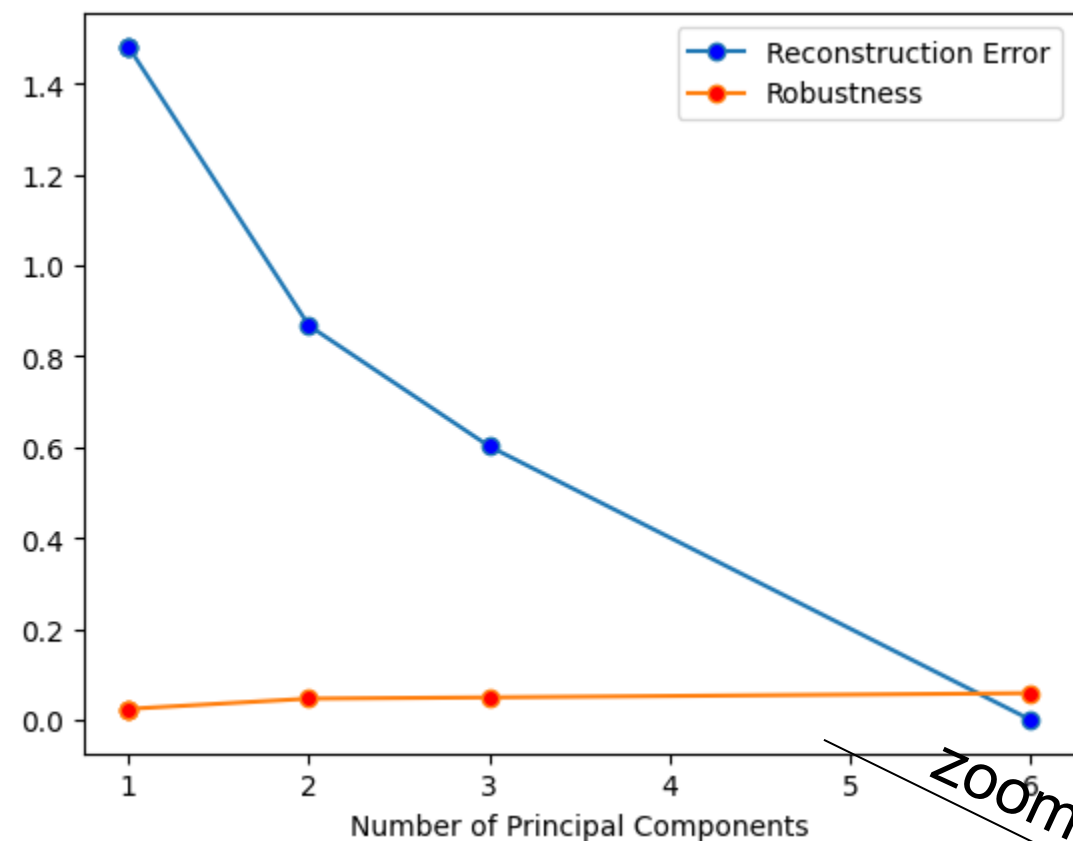
Plot reconstruction error and robustness against the number of principal components



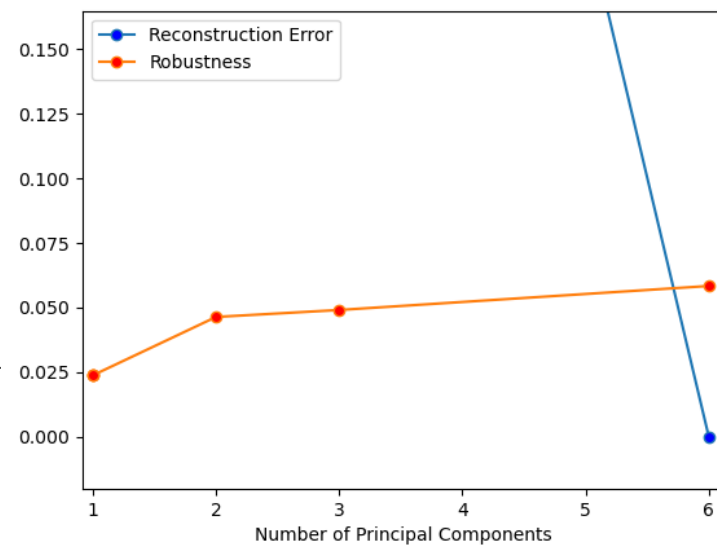
Both are to be minimized!

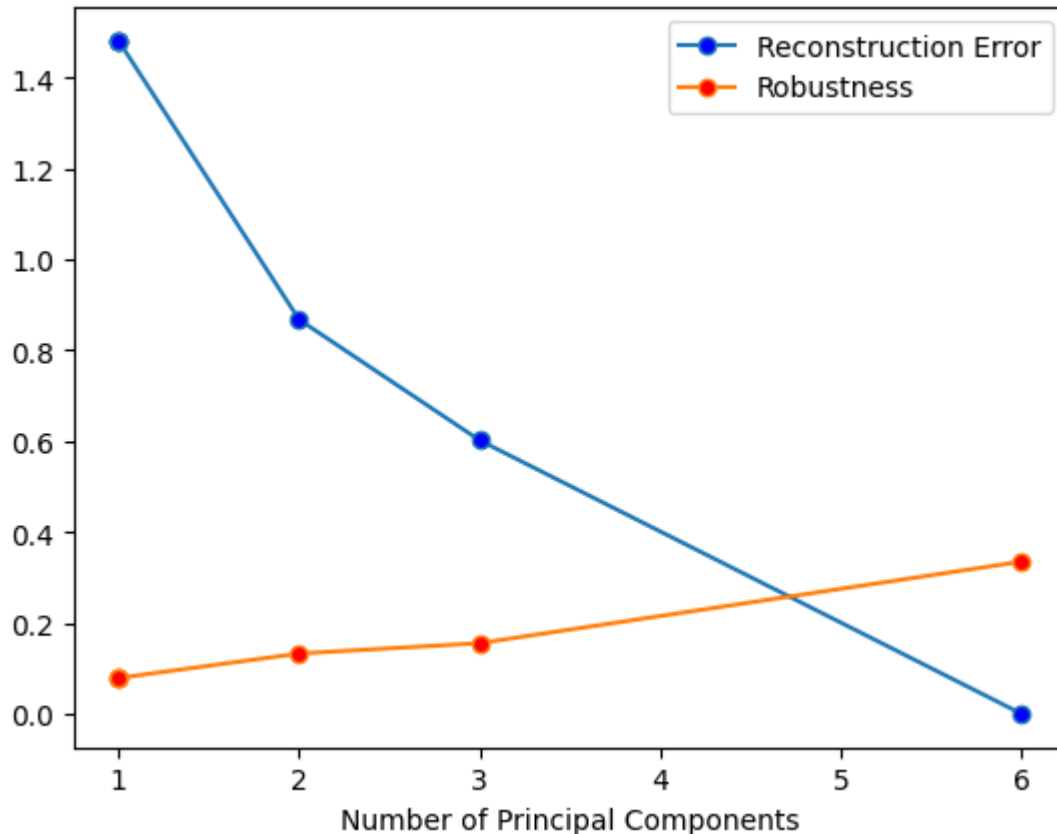
$$\text{Robustness} = \frac{\sum_{t=1}^{N_{\text{valid}}} |\vec{x}_1^{\text{obs-nc}}(t) - \hat{x}_1^{\text{nc}}(t)|^2}{N_{\text{valid}}}$$

$$\text{RMSE} = \frac{\sum_{j=1}^n \frac{\sum_{t=1}^{N_{\text{valid}}} |\hat{x}^{\text{nc}}(t,j) - \vec{x}^{\text{obs}}(t,j)|^2}{N_{\text{valid}}}}{n}$$



Zoom





→ Both are to be minimized!

$$\text{Robustness} = \frac{\sum_{t=1}^{N_{\text{valid}}} |\vec{x}_6^{\text{obs-nc}}(t) - \hat{x}_6^{\text{nc}}(t)|^2}{N_{\text{valid}}}$$

$$\text{RMSE} = \frac{\sum_{j=1}^n \frac{\sum_{t=1}^{N_{\text{valid}}} |\hat{x}^{\text{nc}}(t,j) - \vec{x}^{\text{obs}}(t,j)|^2}{N_{\text{valid}}}}{n}$$

Optimal value of *variance* (and thus of PCs) is a trade off between RMSE and robustness.

We consider $v=0.99$
2 PC



Consider the same data used in Exercise 1

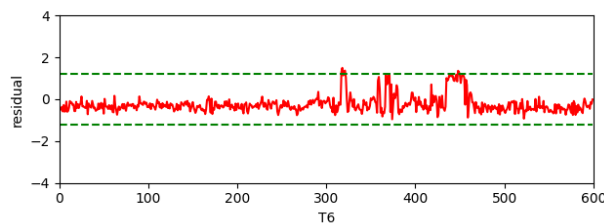
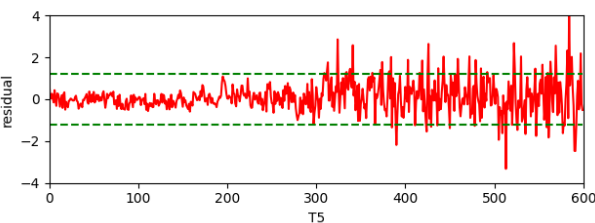
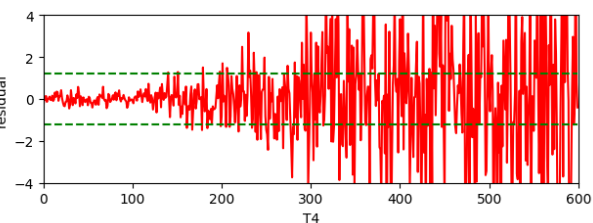
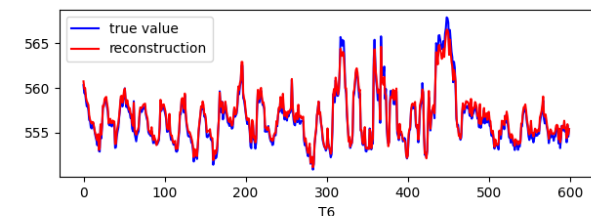
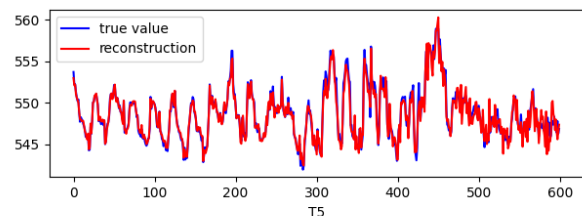
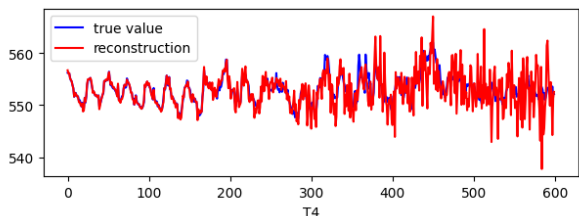
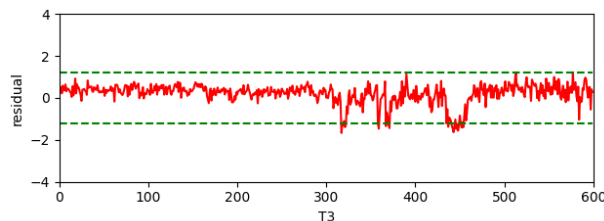
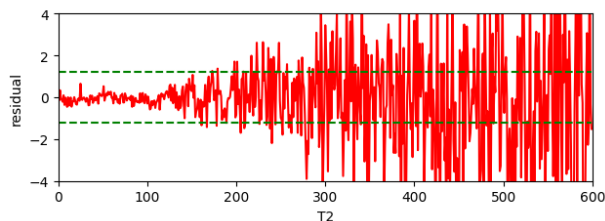
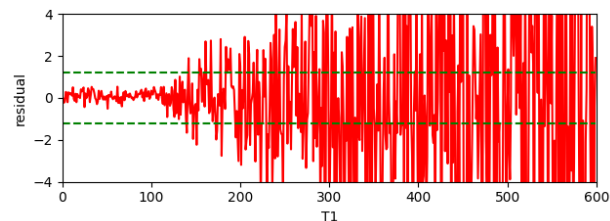
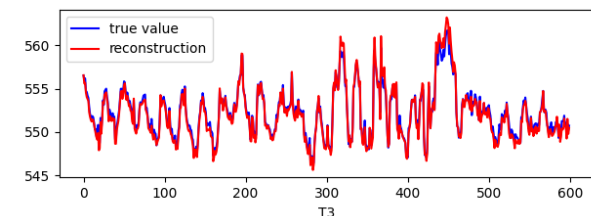
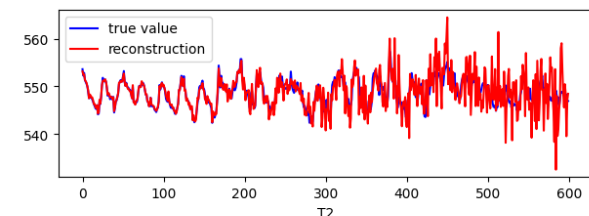
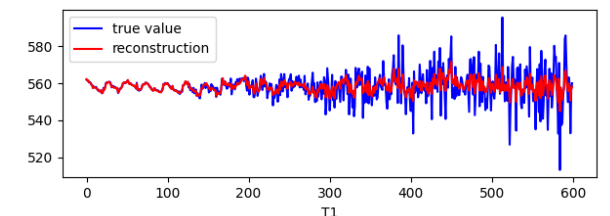
1. Optimize the PCA reconstruction model to achieve:
 - a. Accuracy
 - b. Robustness
2. Apply the PCA model on test datasets: test_1.dat, test_2.dat, test_3.dat and test_4.dat.

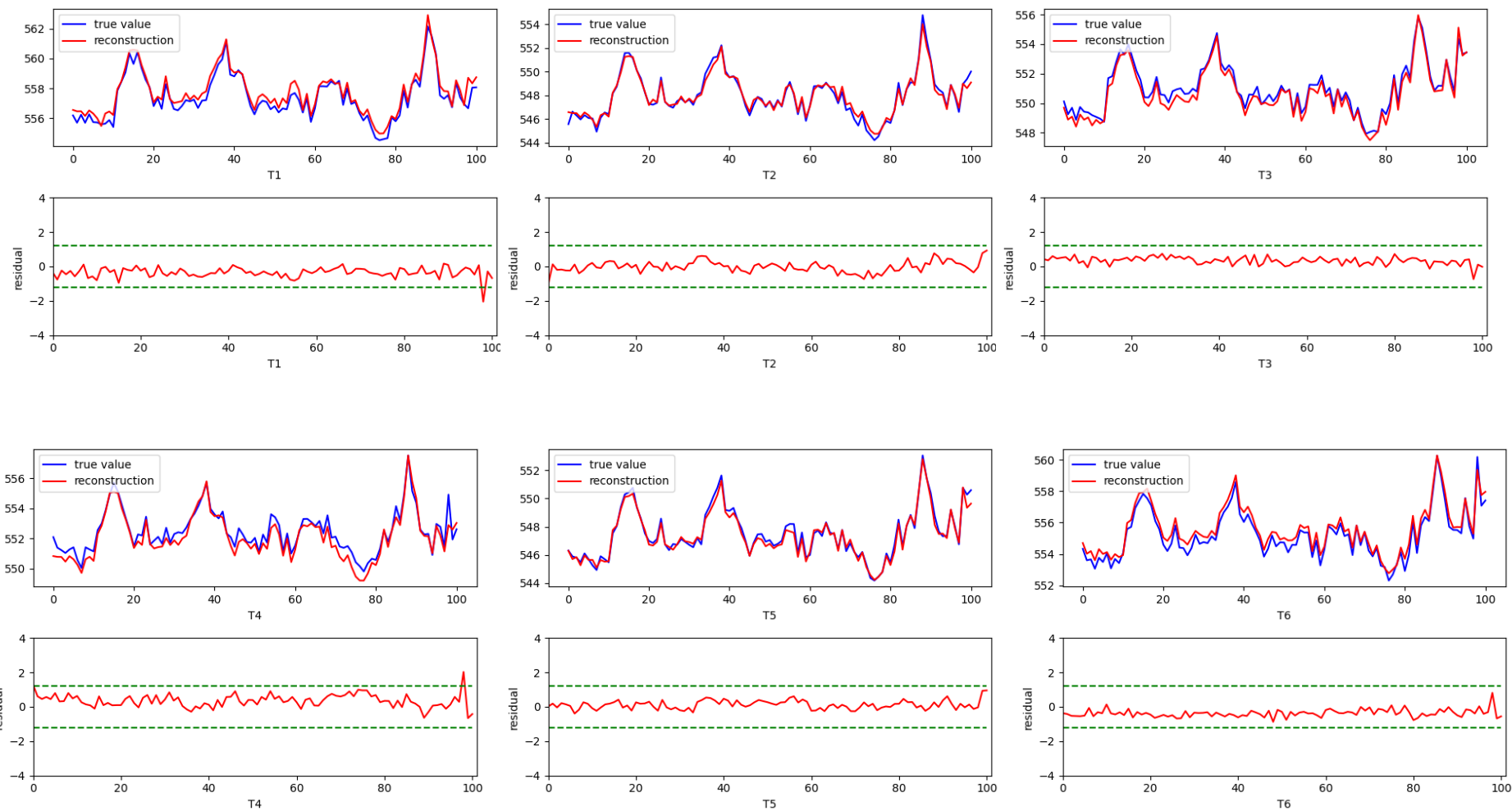
```
[test_reconstruction,n_PC, RMSE]=PCA_reconstruction('train.dat','validation.dat',0.95)
```



test_1.dat → anomaly on signal 1

27

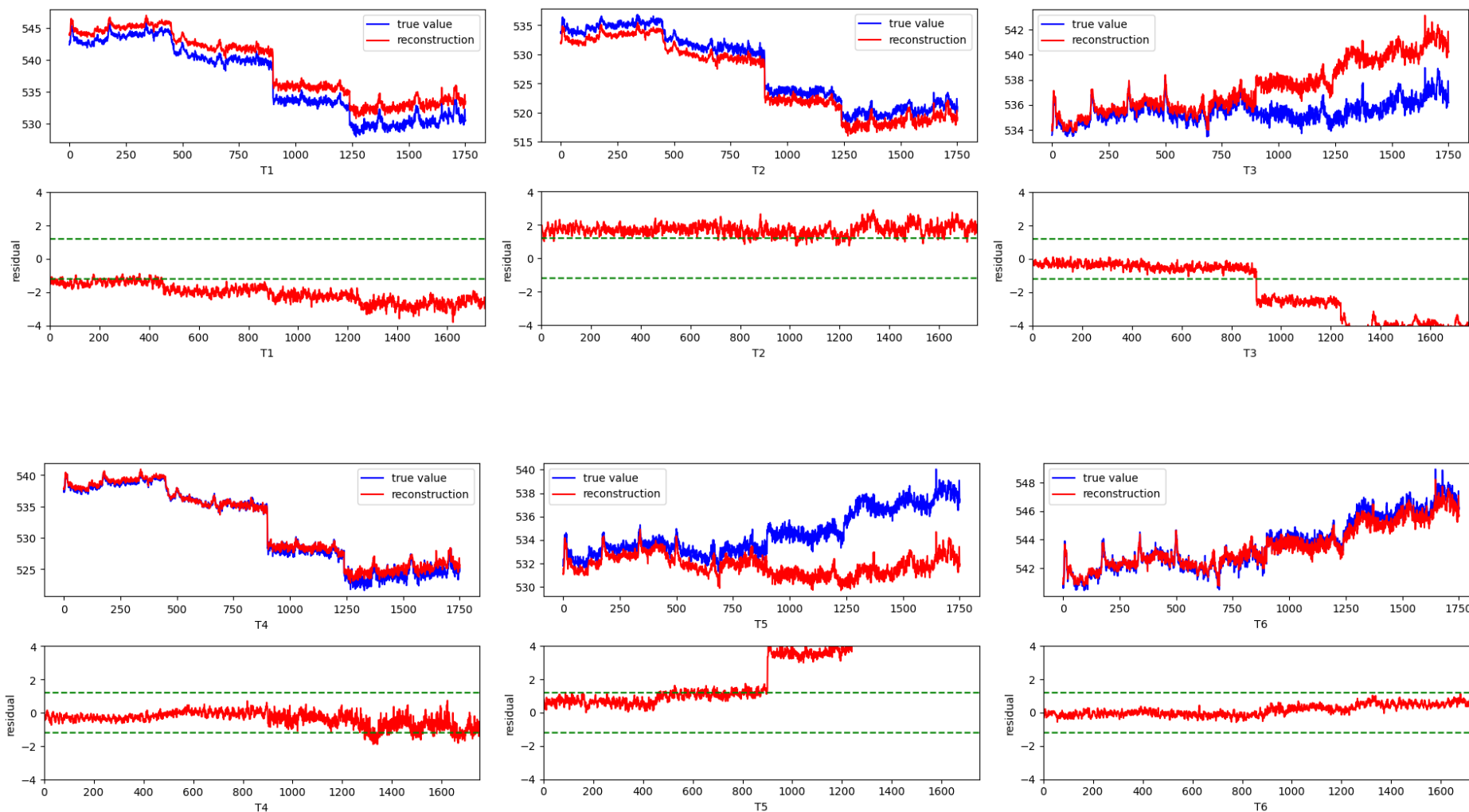


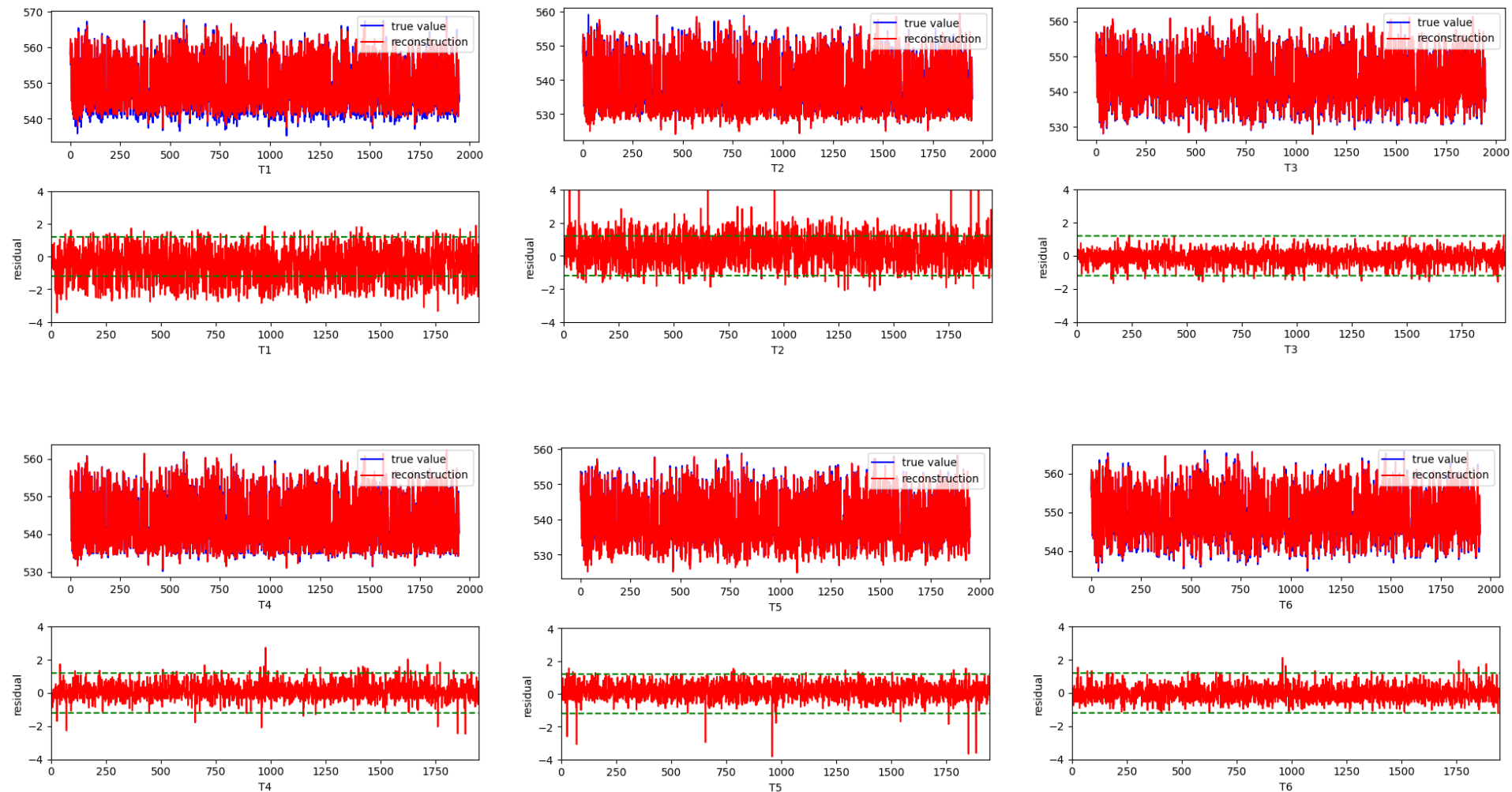




test_3.dat → Several Anomalous Signals

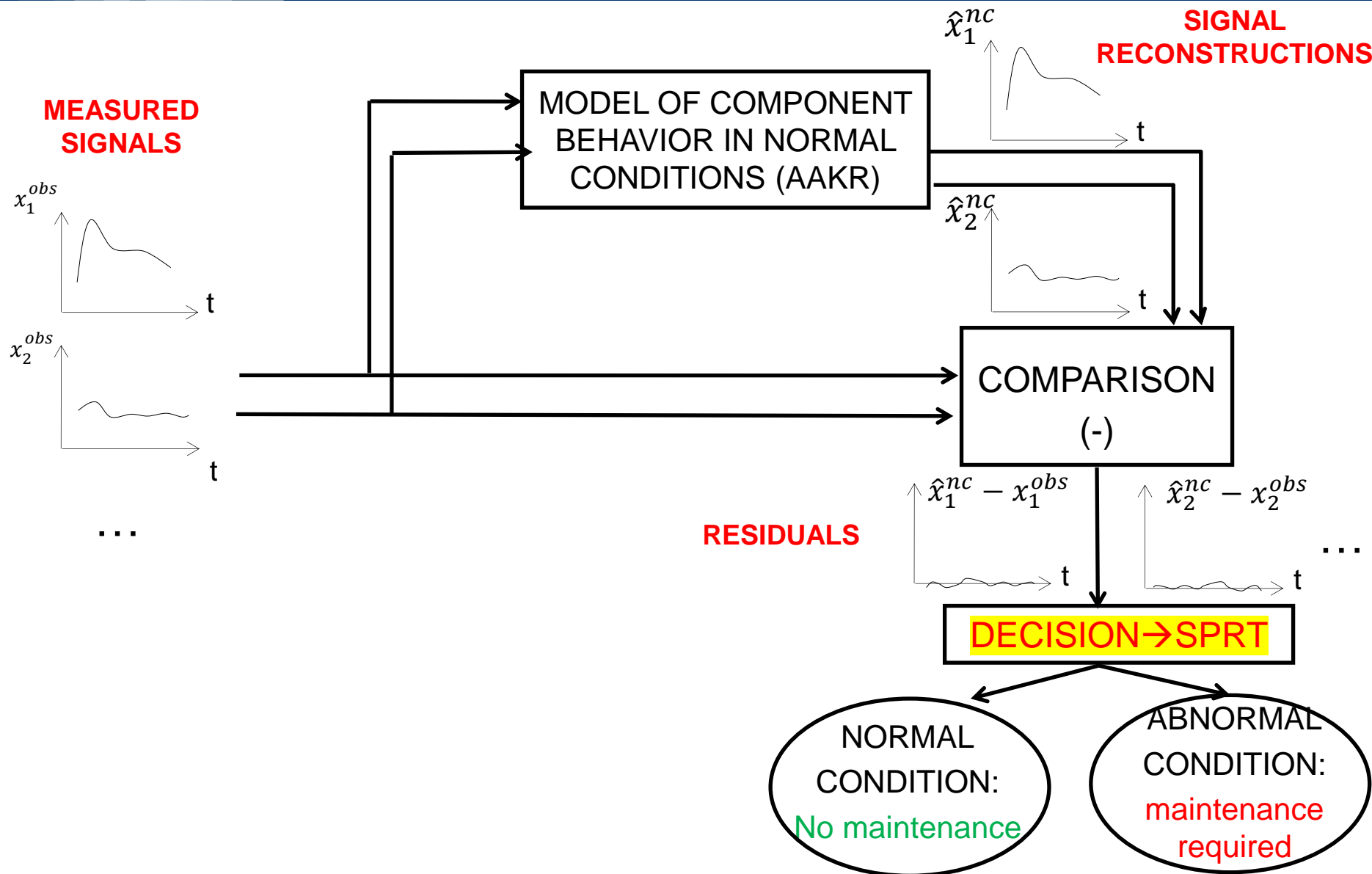
29

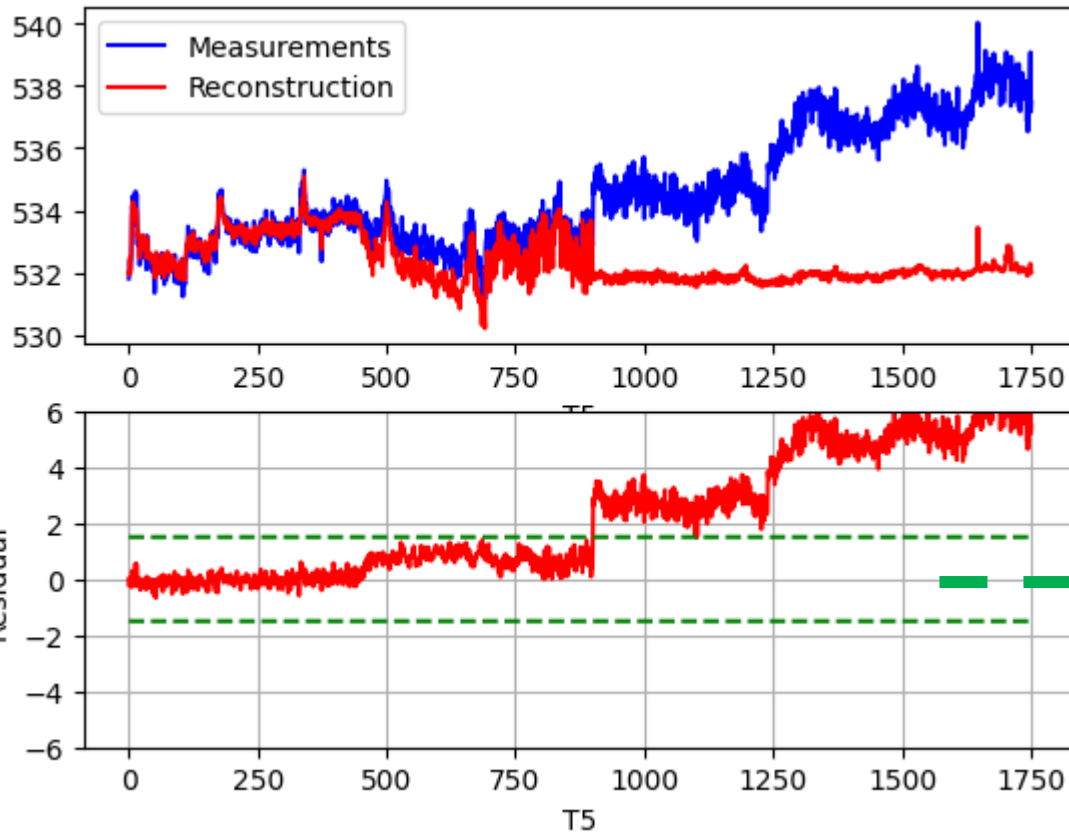






Sequential Probability Ratio Test (SPRT)





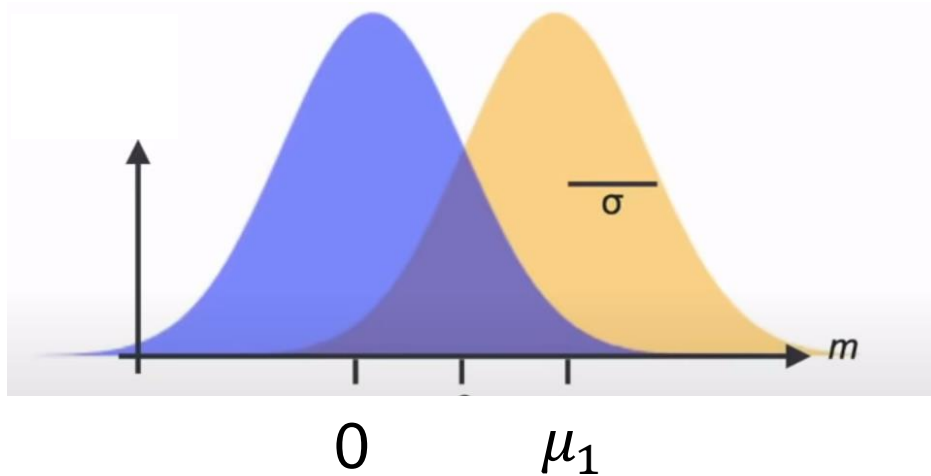
How to properly set the residuals thresholds to trigger an alarm?

Sequential Probability Ratio Test



Assumption: **residuals** come from two normal distributions with same variance:

$$P(R|H_0) = N(0, \sigma)$$
$$P(R|H_1) = N(\mu_1, \sigma)$$



Parameters to be set:

- the residual variance in normal and abnormal condition (σ)
- the expected offset amplitude (μ_1)

Our desiderata:

- the maximum acceptable false alarm rate (α)
- the maximum acceptable missing alarm rate (β)



Sequentially compute loglikelihood:

$$L_0 = 1 \rightarrow \ln(L_0) = 0$$

$$\ln(L_1) = \ln(L_0) + \frac{\mu_1}{\sigma^2} \left(r^{(1)} - \frac{\mu_1}{2} \right)$$

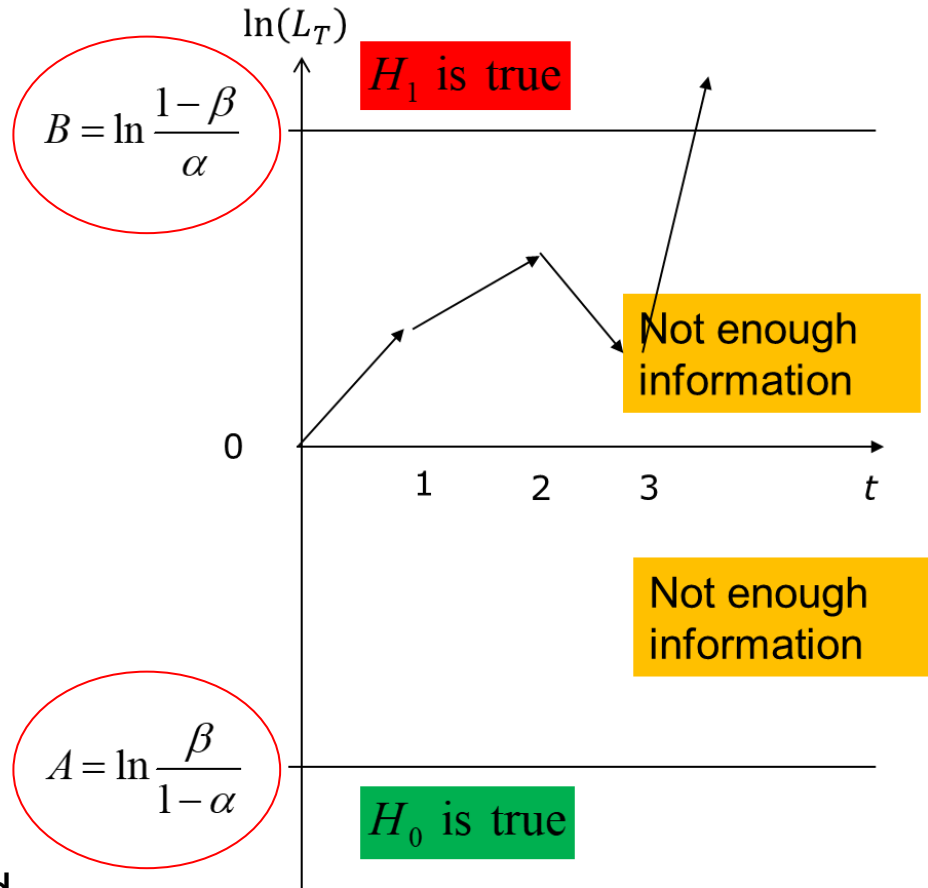
$$\ln(L_2) = \ln(L_1) + \frac{\mu_1}{\sigma^2} \left(r^{(2)} - \frac{\mu_1}{2} \right)$$

...

Set decision boundaries:

$$A = \ln \frac{1 - \beta}{\alpha}$$

$$B = \ln \frac{\beta}{1 - \alpha}$$



You are sure you will meet requirements on false and missed alarms (α and β)!



$$[SPRT_{Index}] = SPRT(test_{name}, test\ rec_{name}, \alpha, \beta, \mu_{H_1}, \sigma)$$

OUTPUT:

- $SPRT_{Index} = N$ -by-1 vector containing the $SPRT$ index

INPUTS:

- $test_{name} = N$ -by-1 data matrix containing N sequential measurements of one signal
- $test\ rec_{name} = N$ -by-1 data matrix containing the N reconstructions of the data in $test_{name}$.
- α = Desired false alarm rate
- β = Desired missed alarm rate.
- μ_{H_1} = the mean of the Gaussian distribution of H_1 hypothesis (the component is working in abnormal conditions).
- $\sigma = \sigma_{H_0} = \sigma_{H_1}$ = variance of Gaussian distribution

Note that the function receives the names of the files containing the data. First you have to save the signal you want to apply SPRT to and then, call it by its name.

Example:

```
save('test_4A_rec_S5.dat', 'test_reconstruction(:,5)', '-ascii')  
[SPRT_Index] = SPRT('test_4A_meas_S5.dat', 'test_4A_rec_S5.dat', 0.01, 0.01, 1.5, 0.2)
```

Python:

```
train_data = np.loadtxt(train_data_name); np.savetxt('test_4A_rec_S5.dat', test_reconstruction[:, 4], fmt='%.6f')  
SPRT_Index = SPRT_PCA('test_4B_meas_S5.dat', 'test_4B_rec_S5.dat', alpha, beta, mu1, sigma)
```

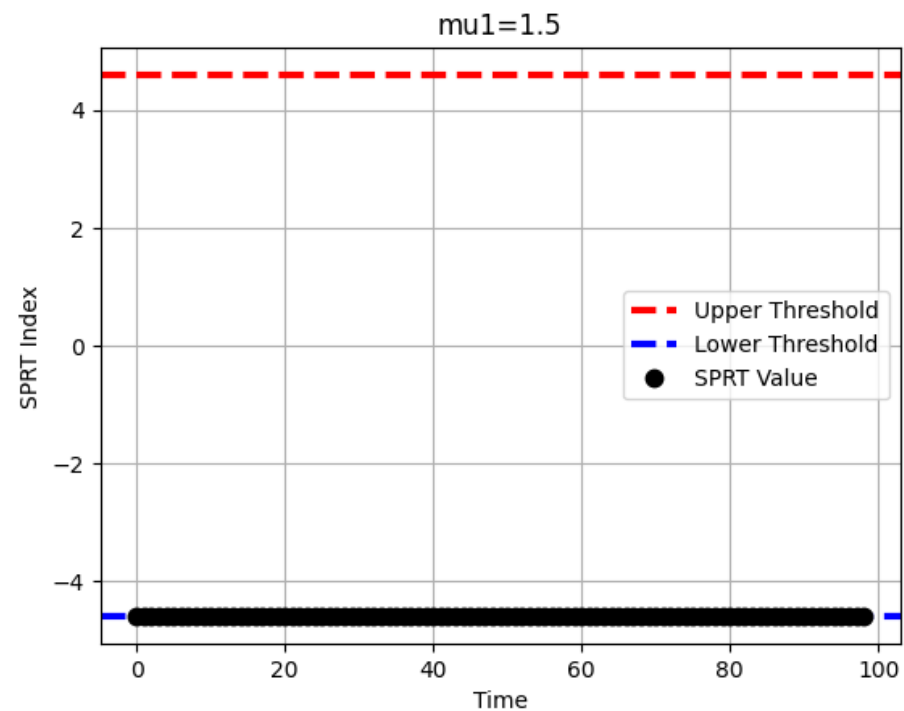
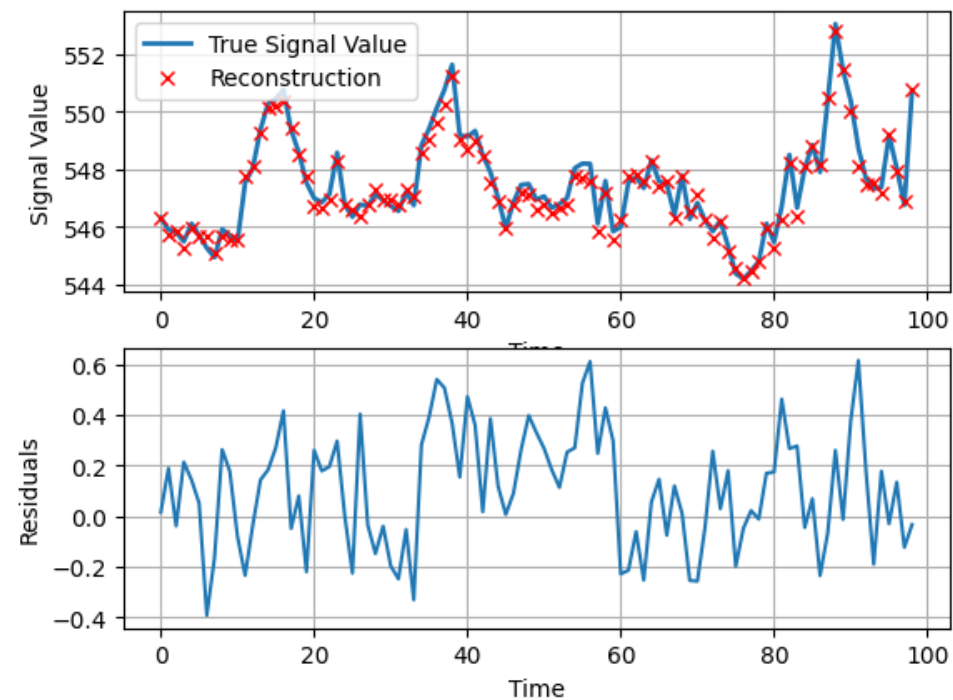


- Apply the SPRT to detect abnormal conditions to the measurements of signal 5 in files “test_4A.dat” and “test_4B.dat”. The following setting of the SPRT parameters is suggested:
 - $\alpha = 0.01$
 - $\beta = 0.01$
 - trade off among the two + risk of having many points in the “not enough information” region
 - Assume Gaussian distributions for the two hypothesis H_0 and H_1 with parameters:
 - $\mu_{H_0} \approx 0, \sigma_{H_0} = 0.2 \rightarrow$ could be computed from historical healthy data!
 - $\mu_{H_1} = 1.5, \sigma_{H_1} = 0.2 \rightarrow$ if no abnormal data are available, we must set these param by assumptions. Hint: residuals will grow as the system gets more abnormal \rightarrow Offset on the mean and same variance.
- Do you detect the occurrence of abnormal conditions? When?
- Compare the results with those of a threshold-based method with threshold = 1.
- Draw your conclusions on the possibility of using the SPRT in decision-making for fault detection.



Signal 5, test set 4A

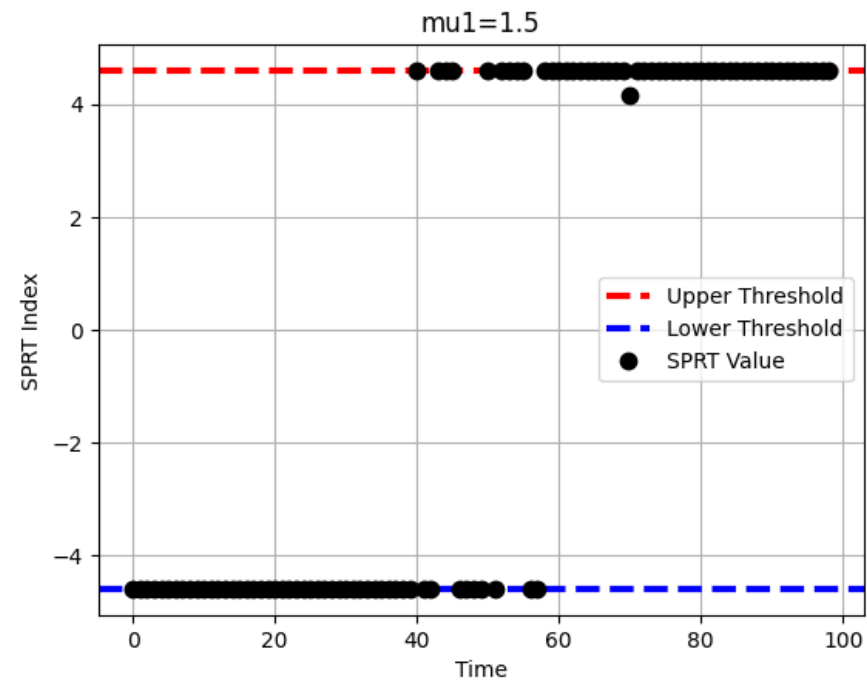
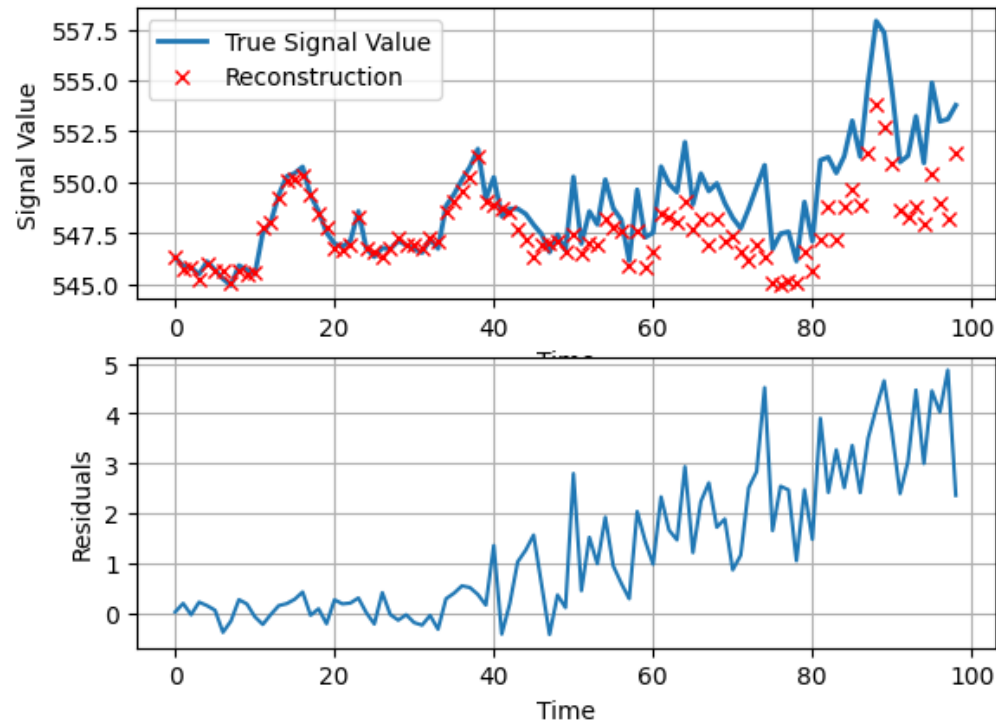
41

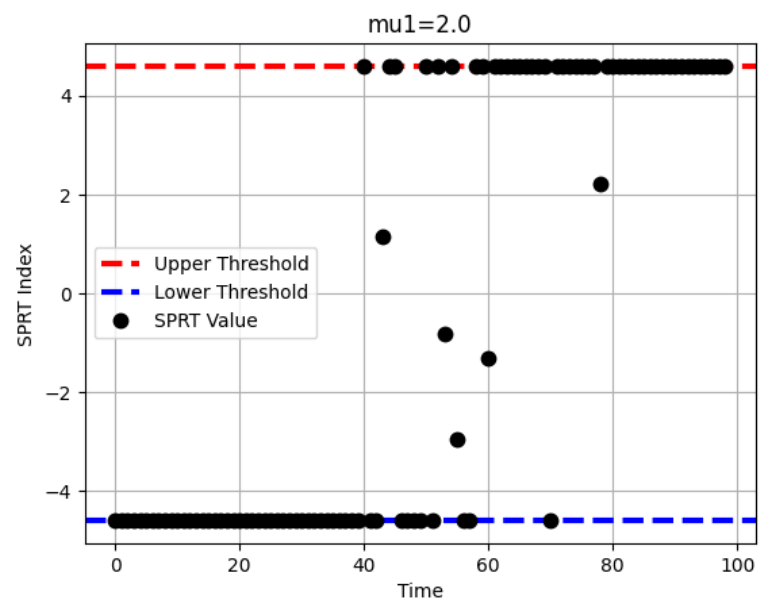
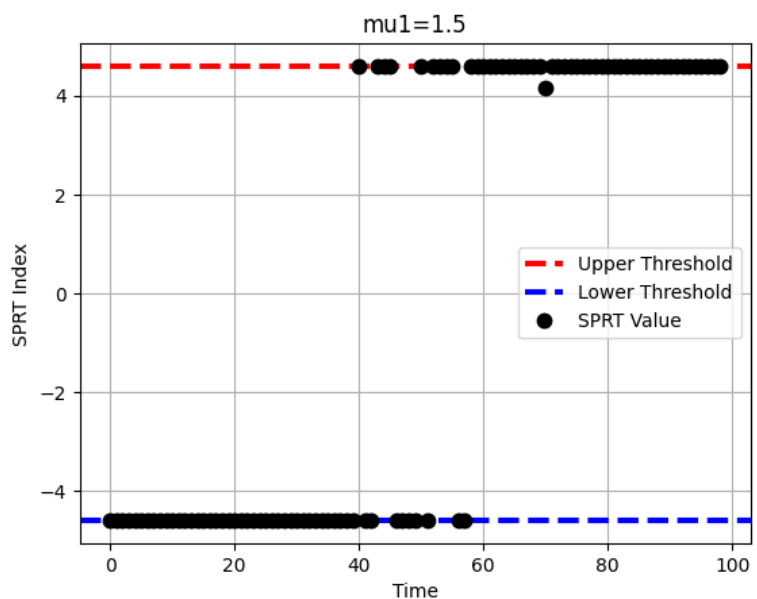
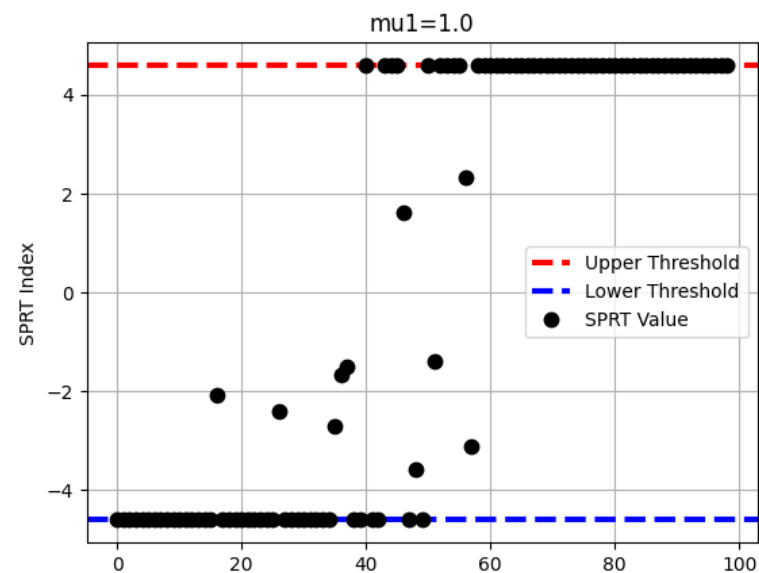
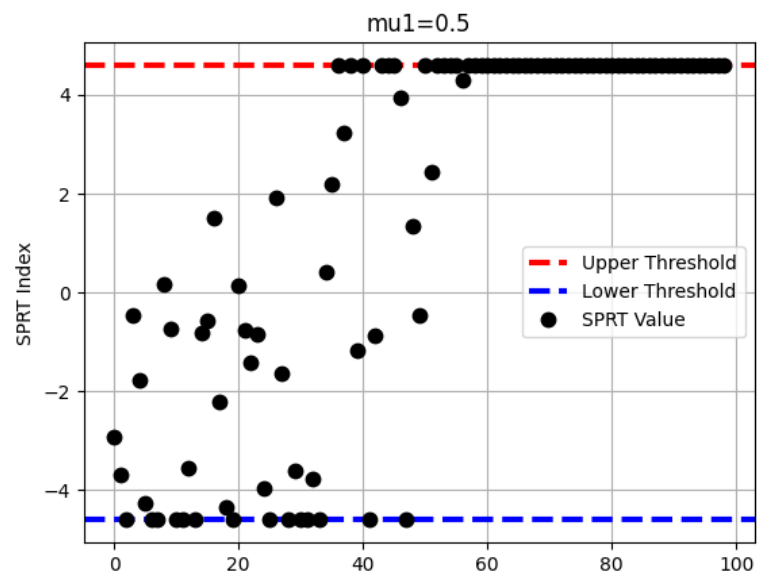




Signal 5, test set 4B

42





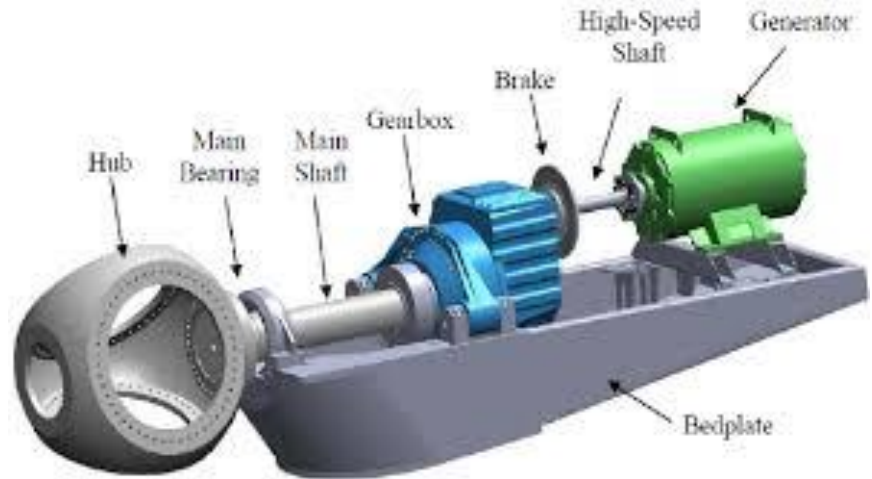
Exercise 2

Method: you choose
Component: Wind Turbine





Wind turbine farm

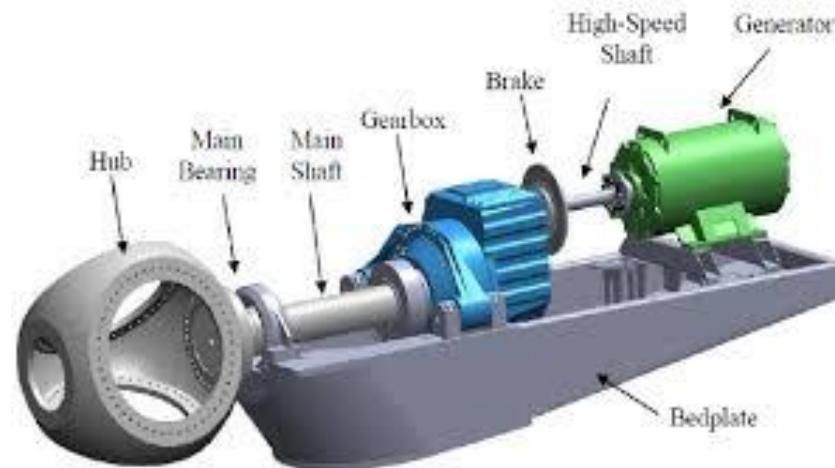


- Main Bearing+ Planetary Gear box + Gearbox + Generator
- Monitoring system: 6 accelerometers and 1 sensor measuring the rotating speed.



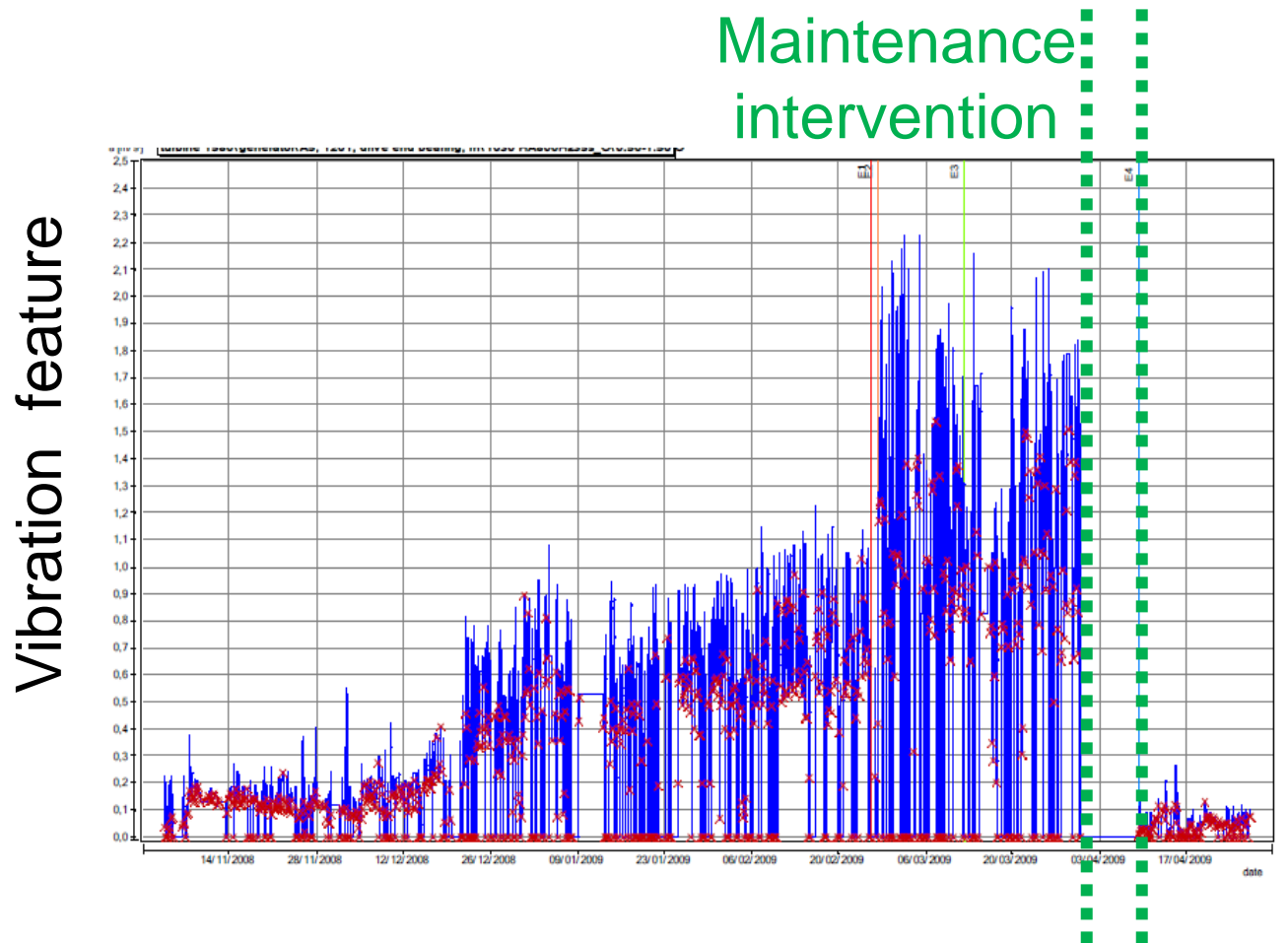
Some possible failures modes

- a) Crashing of the highspeed shaft of the gear box
- b) Breaking of a teeth in the planetary stage
- c) Misalignment between the generator and the gearbox shafts
- d) Wearing of the rear bearing housing of the generator



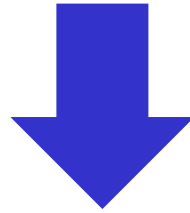
Example

- Wearing of the rear bearing housing of the generator





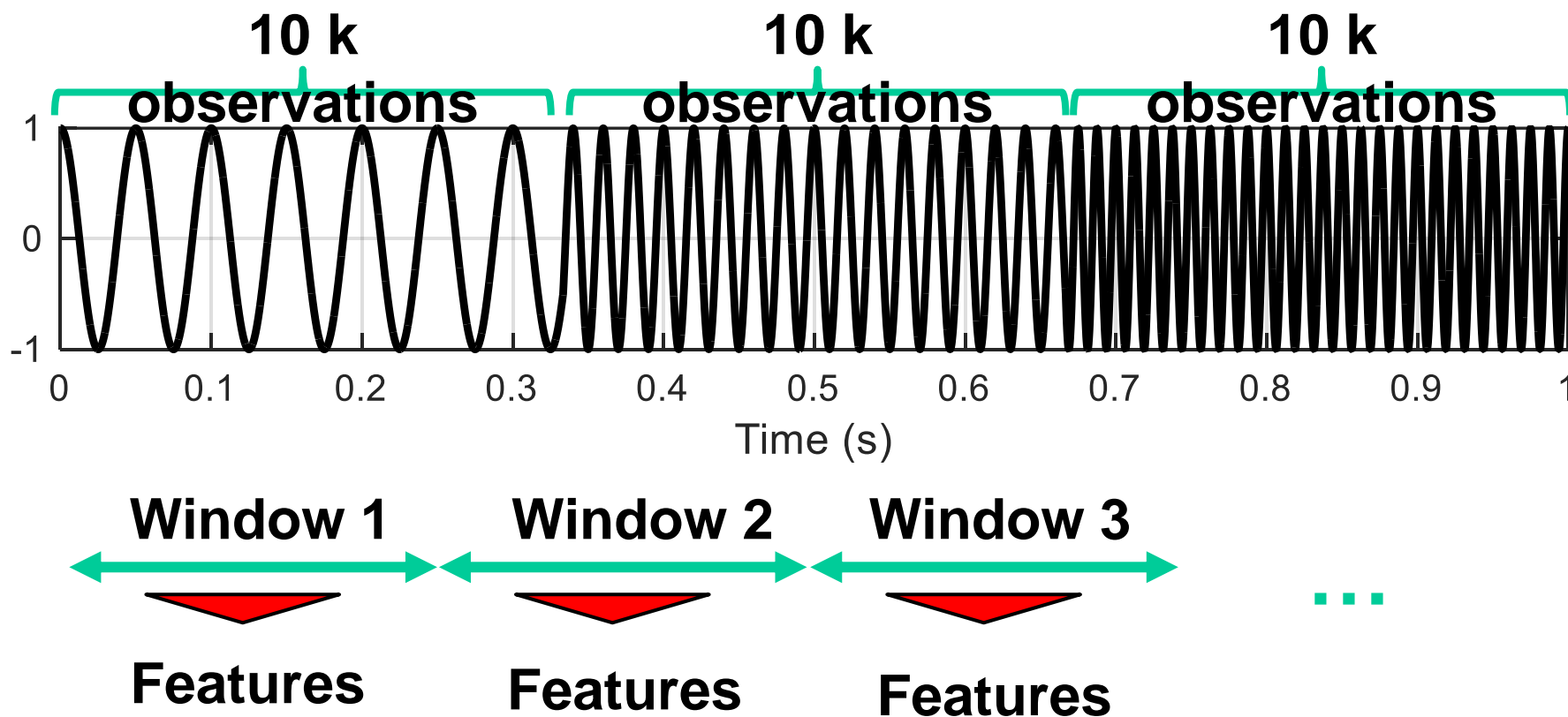
Gearbox-Generator Failure

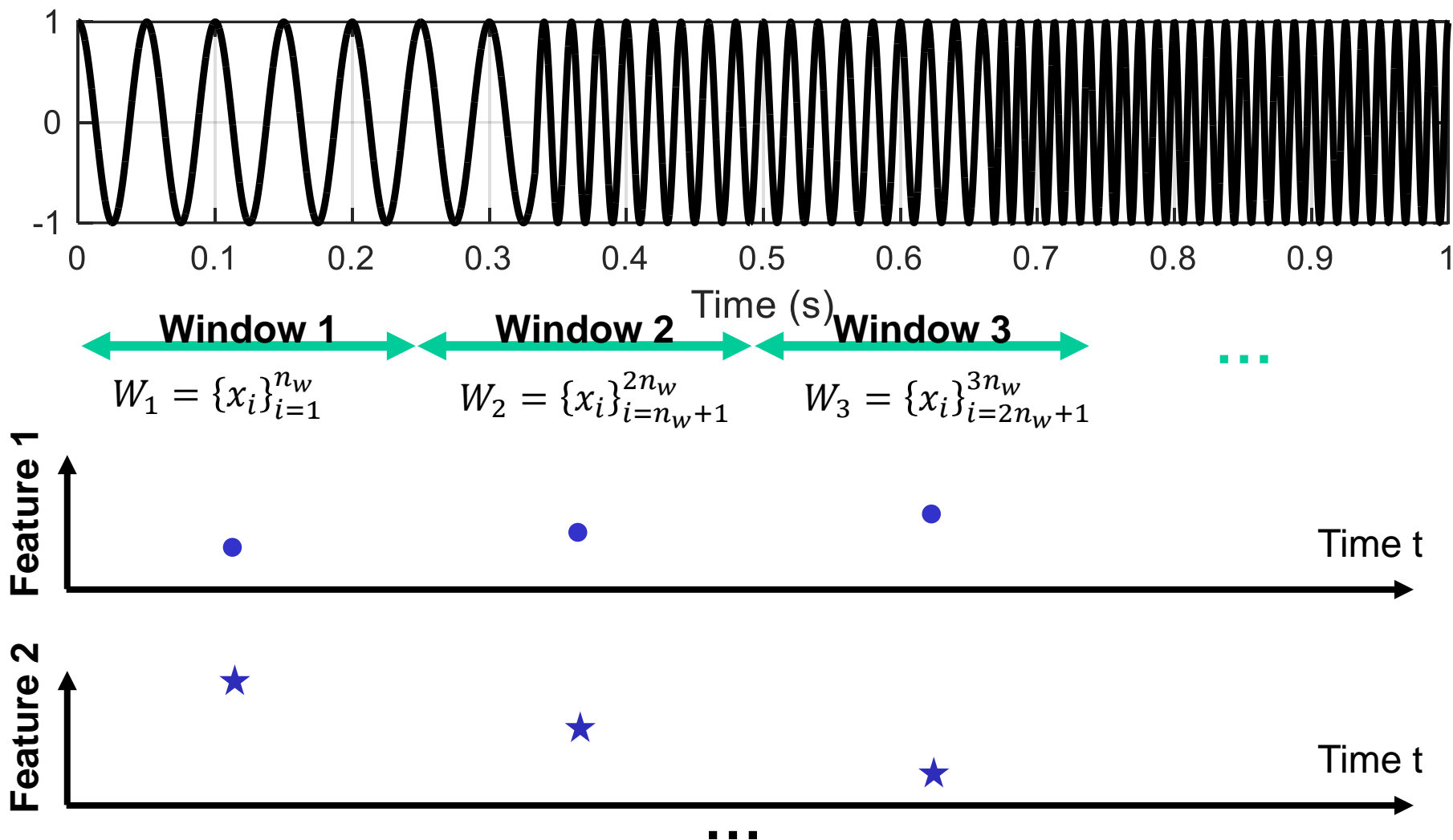


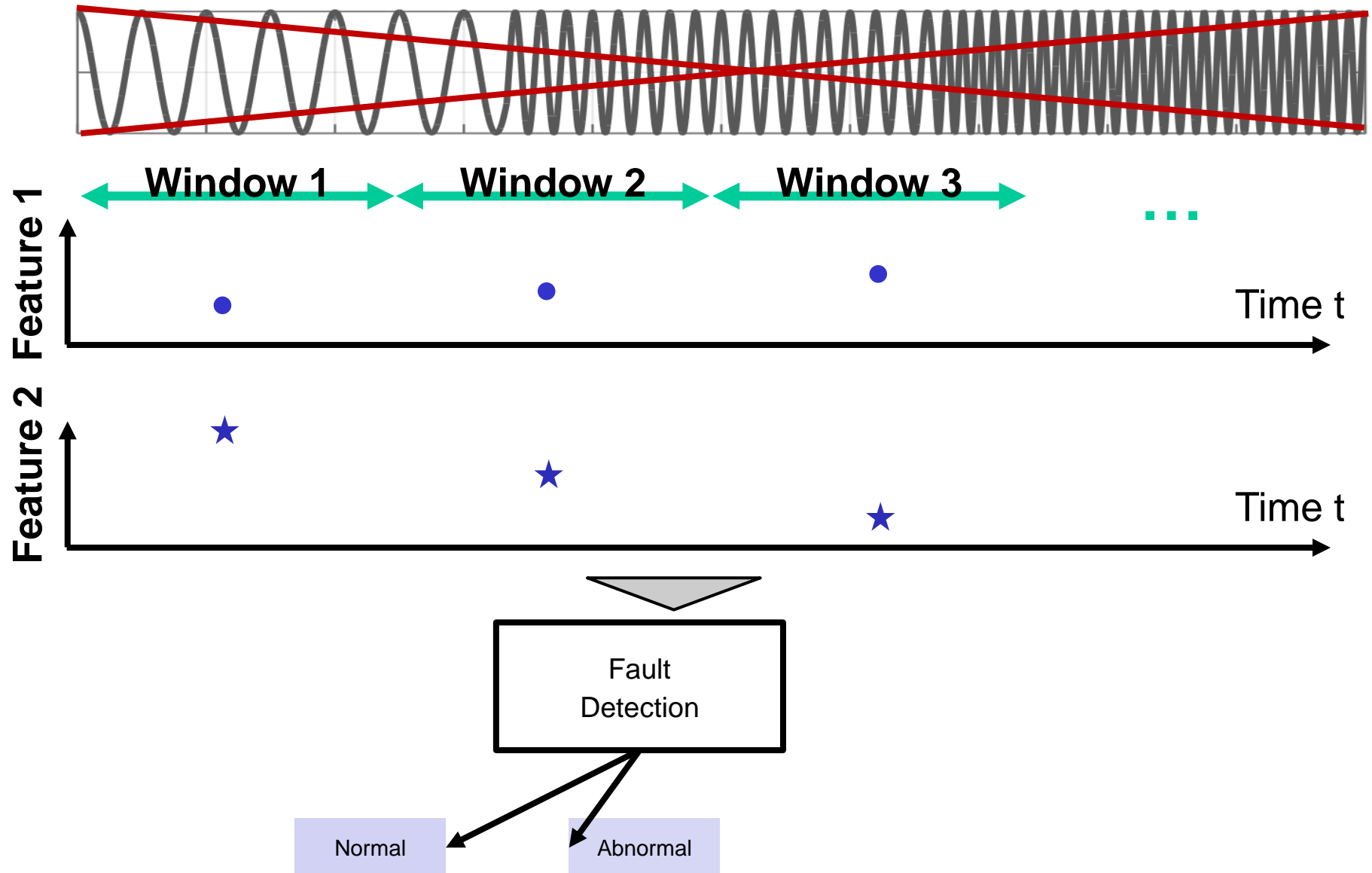
- **Long turbine downtime [up to some months]**
 - **Intervention of expensive tools**



High frequency sensor:









- Each 5 sec signal data are divided in 40 segments(windows) in order to extract features
 - The following statistical features are considered:
Mean, standard deviation, Kurtosis, Skewness, min, max, 2nd moment, 3rd moment
 - A window contains N values $W = \{x_i\}_{i=1}^N$, the corresponding Statistical Features are:

Sample Mean:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$$

Sample Variance:

$$\sigma^2 \approx s^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2$$

Standard Deviation: σ

k -th Moments:

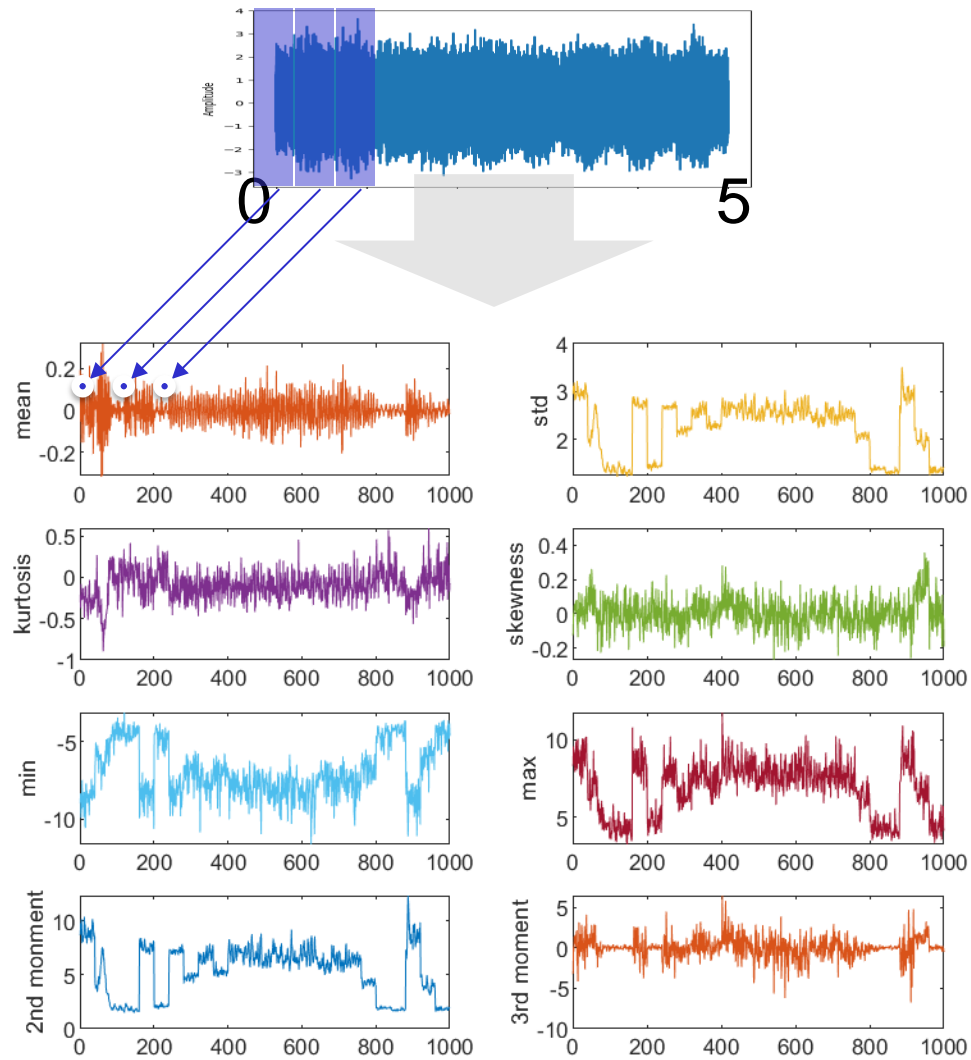
$$\mu_k \approx \frac{1}{N} \sum_{i=1}^N x_i^k$$

Skewness → “asymmetry”
(the third standardized moment)

$$\frac{\mu_3}{\sigma^3}$$

Kurtosis → “peak”
(the fourth standardized moment)

$$\frac{\mu_4}{\sigma^4}$$





Exercise 2: Assignment

In folder Exercise 2 you find:

- 1) The file 'train.dat' which contains the features already extracted in normal condition
- 2) The file 'validation.dat' which contains the features already extracted in normal condition
- 3) The file 'validation_sim.dat' which contains a simulated abnormal condition on signal 3
- 4) The files for testing 'test1.dat' and 'test2.dat'

Each of these files contains the following signals:

mean std kurtosiss skewness min max 2nd Moment 3rd Moment

You are required to:

- 1) **Develop and optimize** a fault detection tool for the turbine generator;
- 2) **Apply the developed tool** to the data in the file "test.mat" and Identify possible abnormal conditions period.

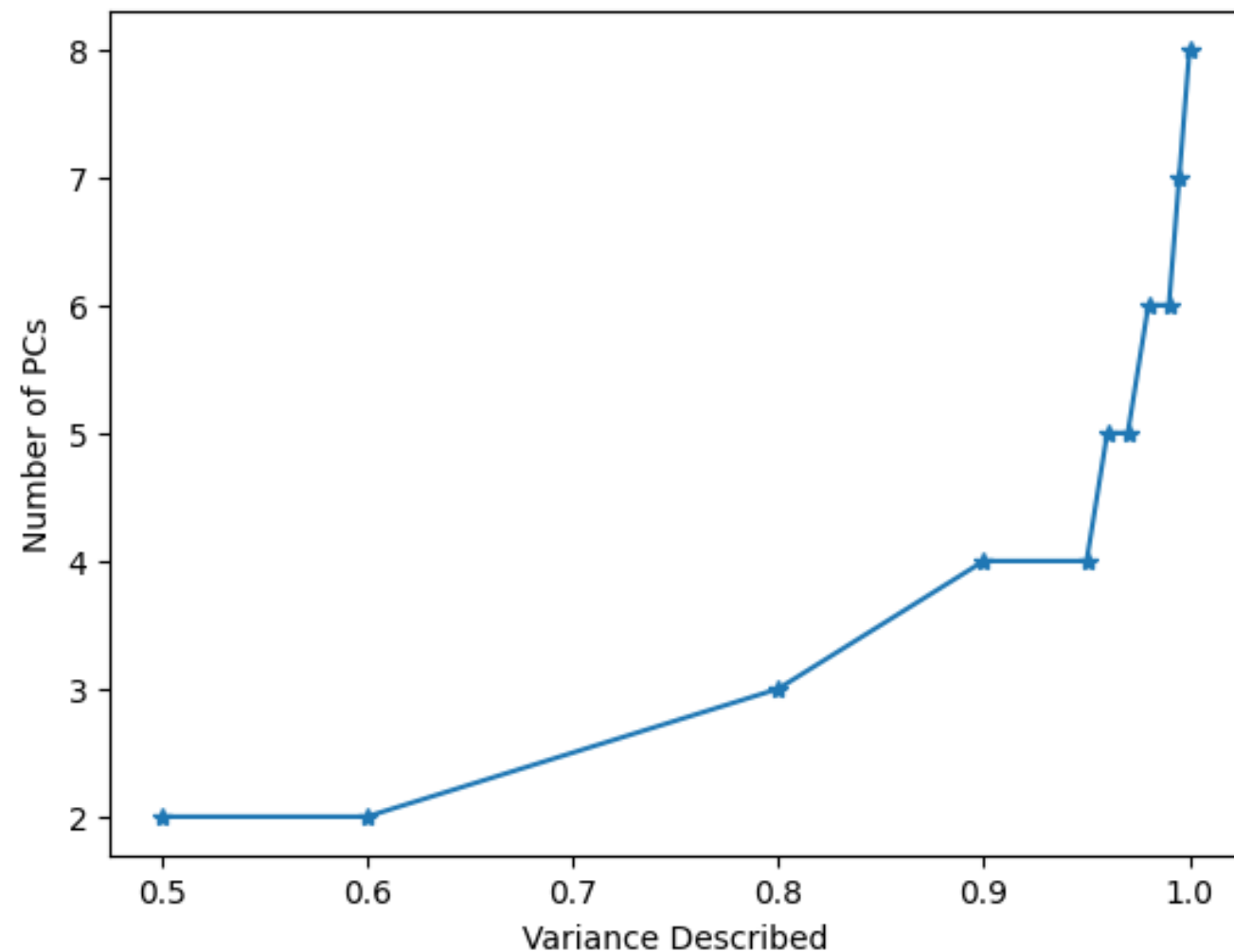


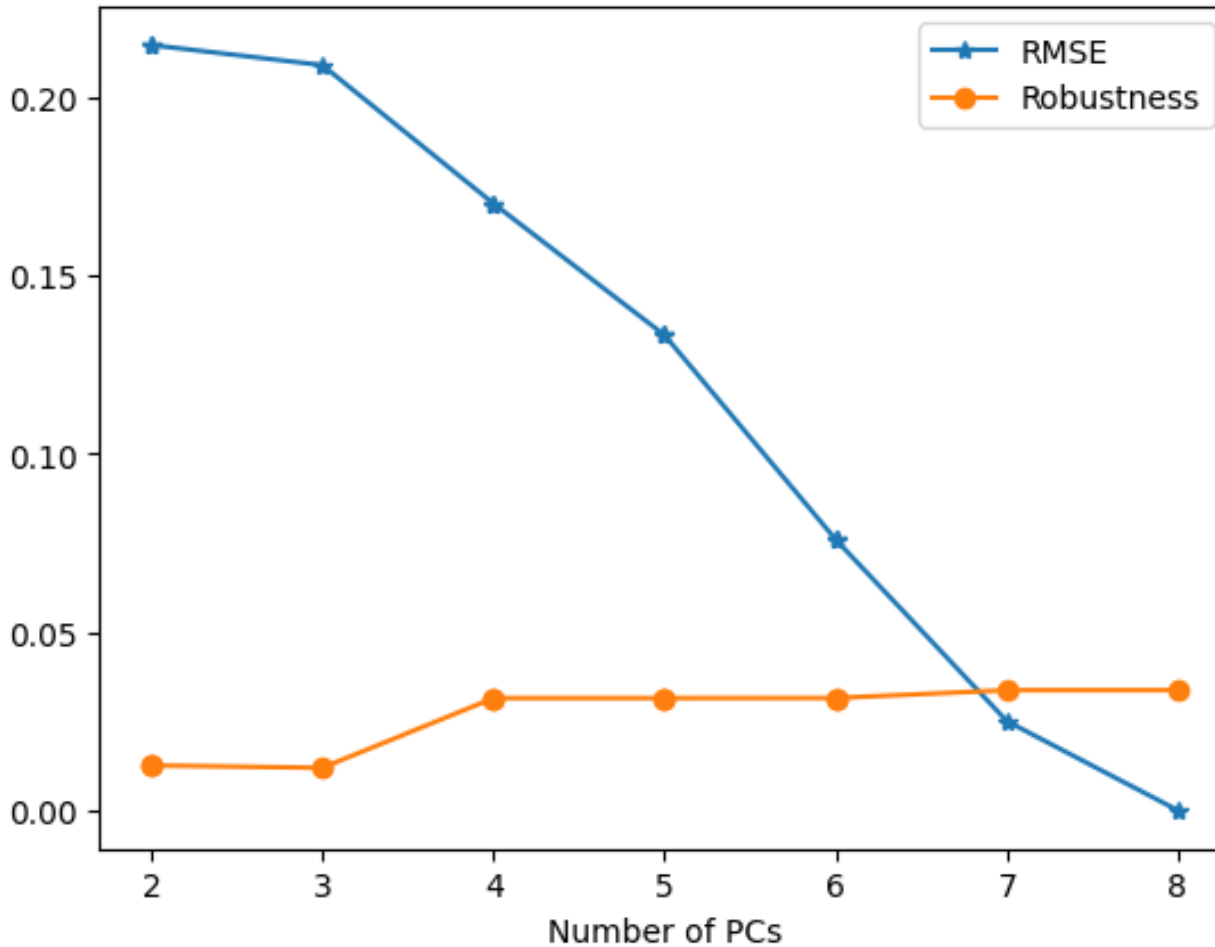
Solutions



Exercise 2

Fault detection with PCA
Component: Wind Turbine

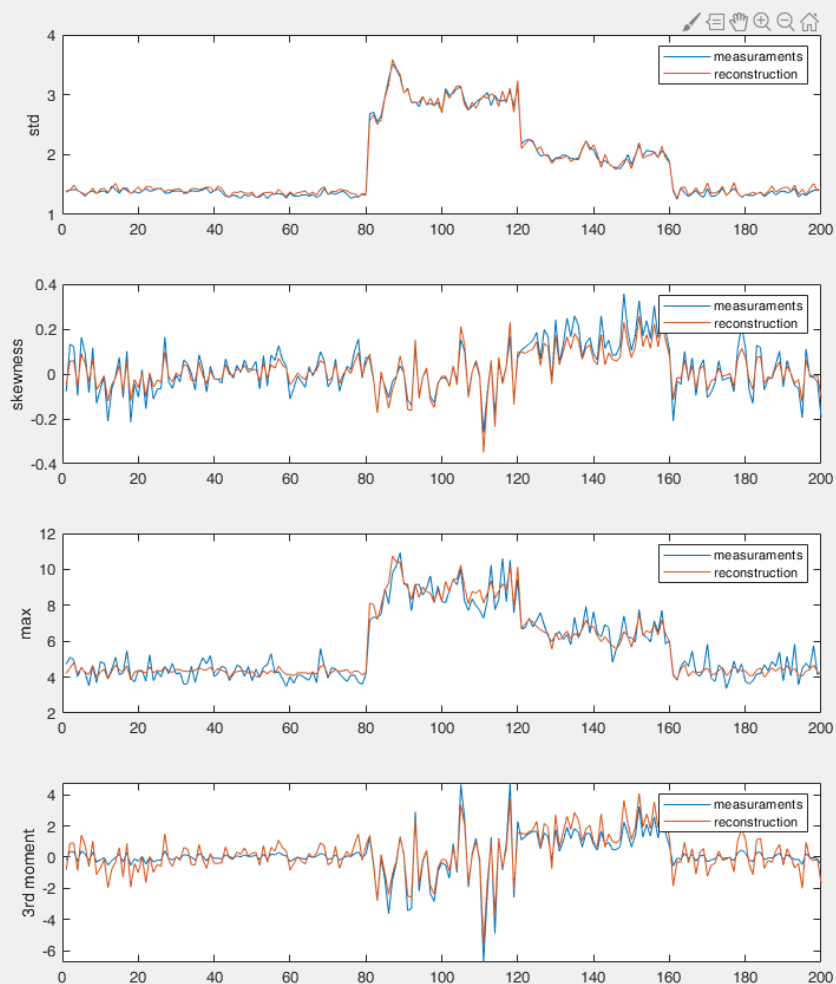
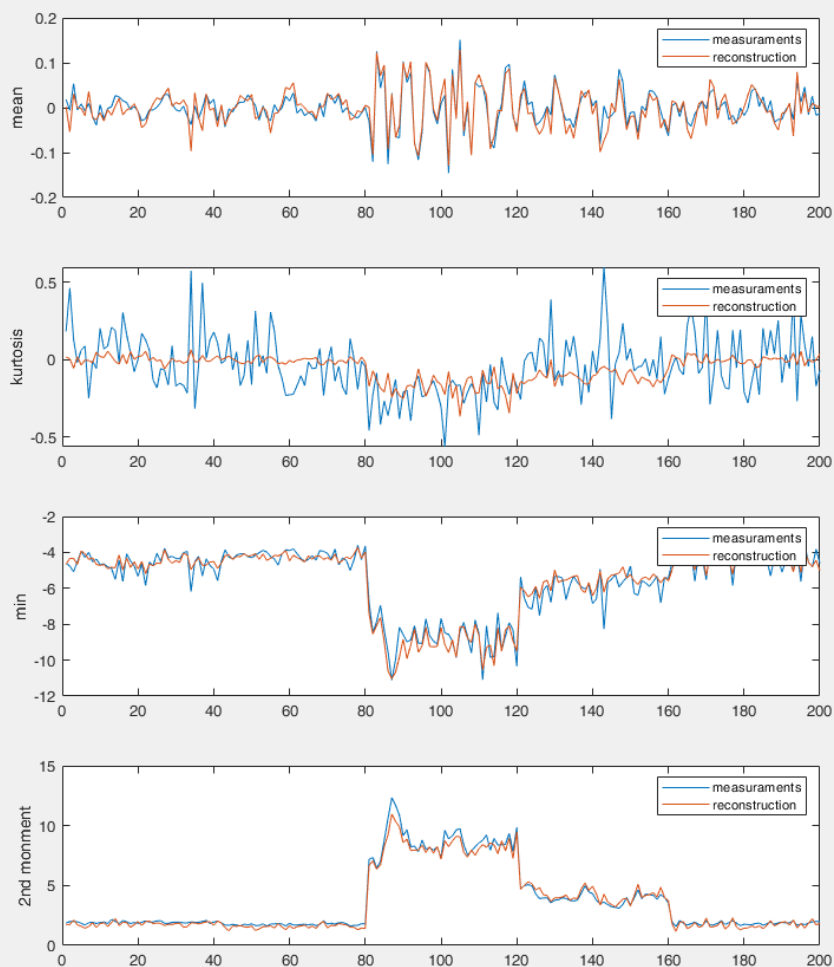




RMSE computed on all signals from the healthy validation data.

Robustness computed on a single signal containing a simulated anomaly

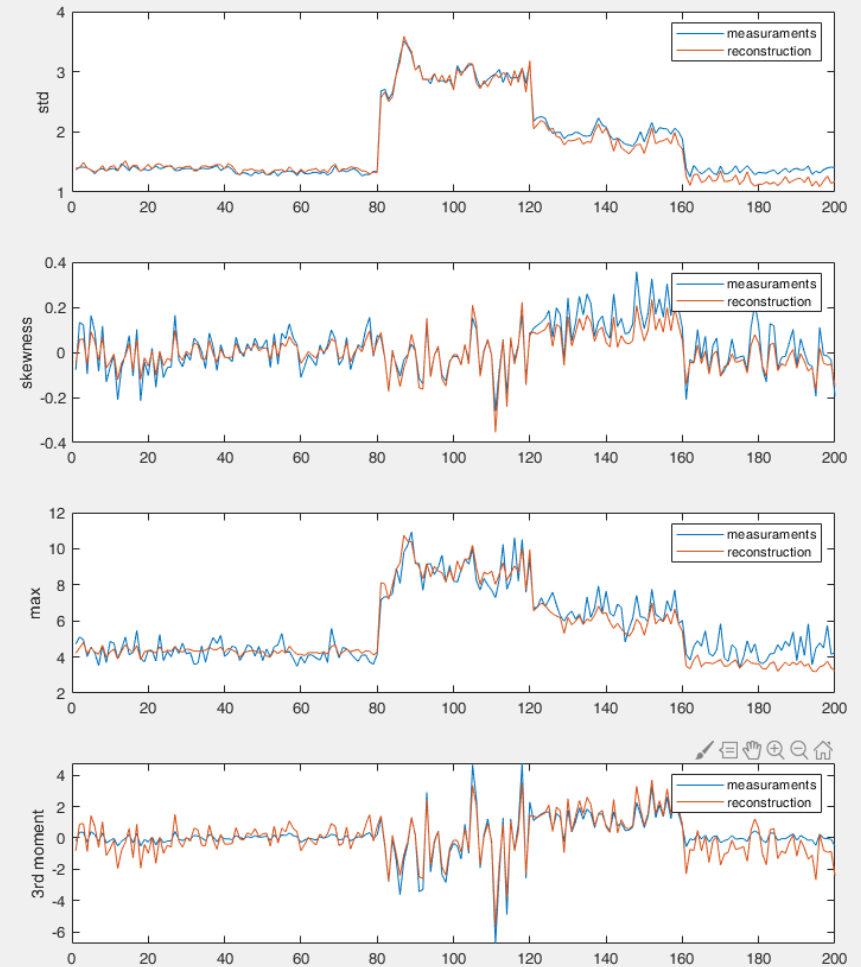
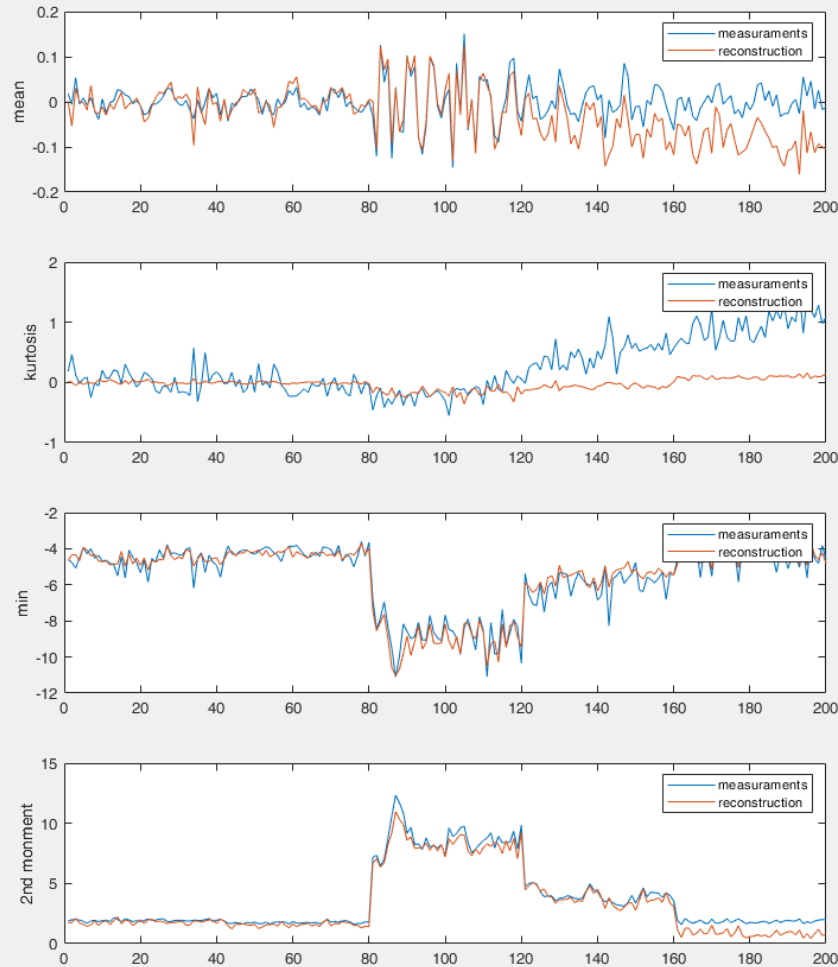
3 or 4 PCs seem the best





3PC Simulated anomaly on signal 3 (kurtosis)

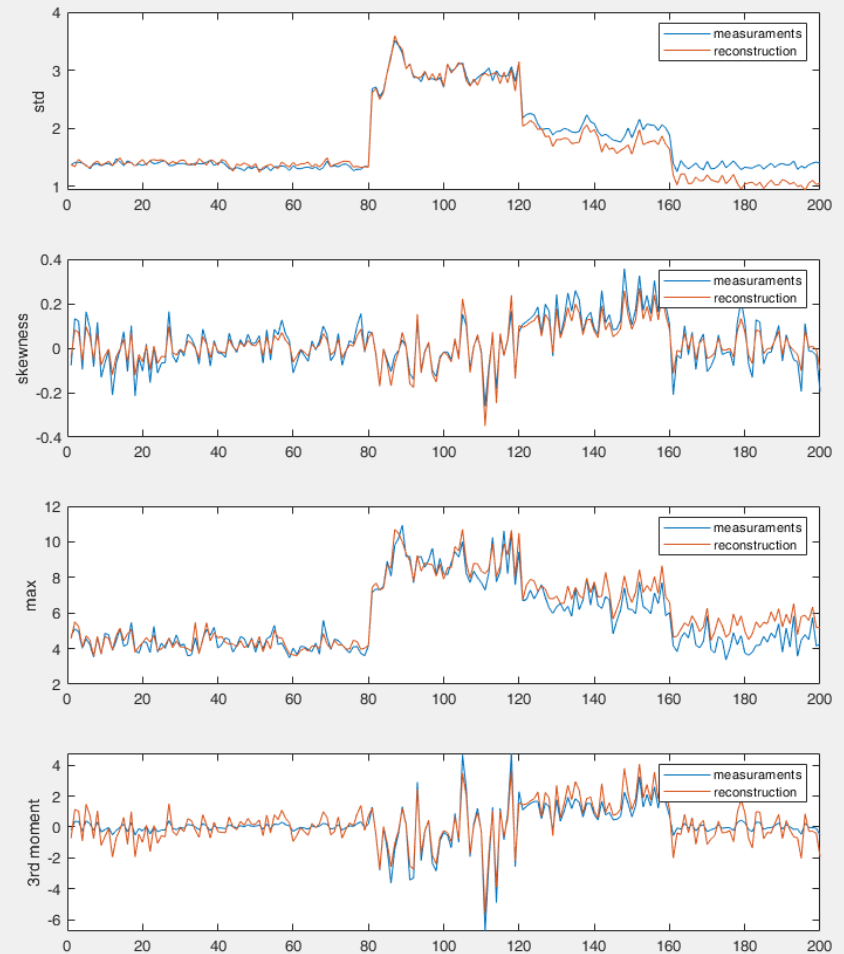
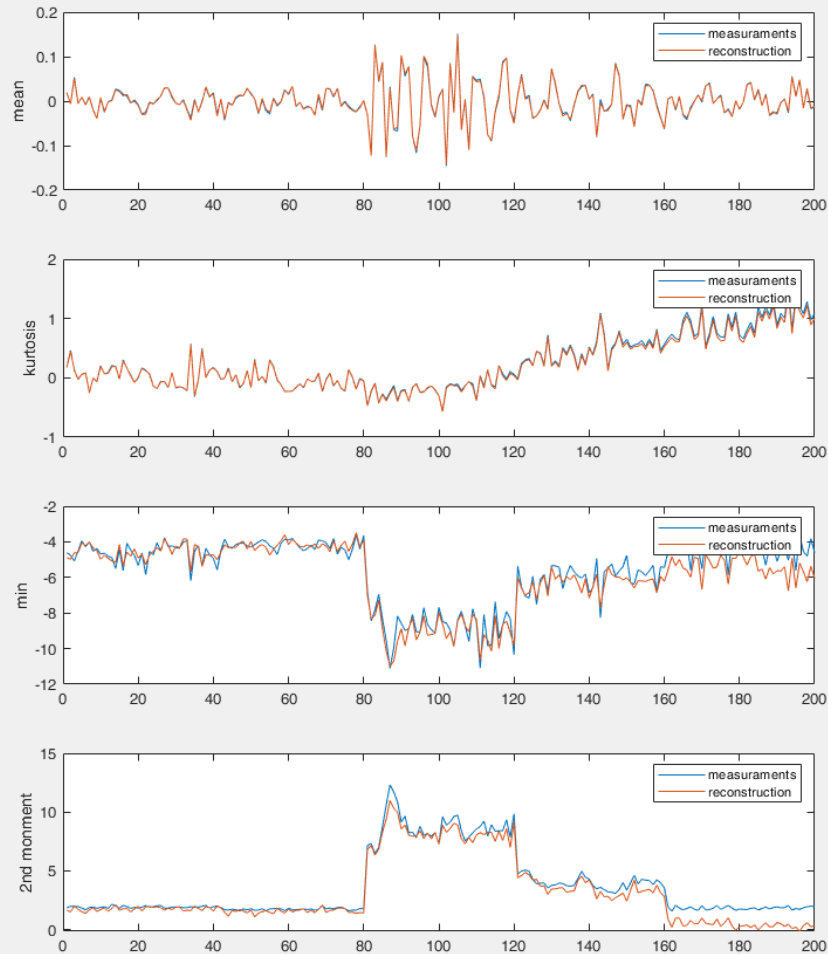
63





4PC Simulated anomaly on signal 3 (kurtosis)

64





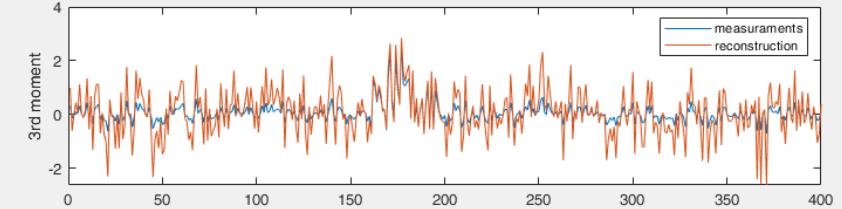
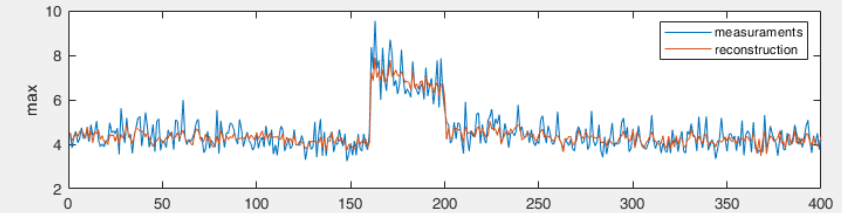
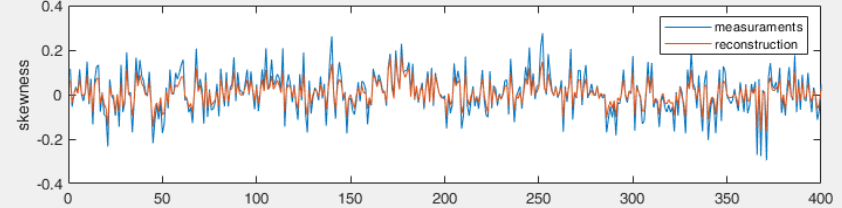
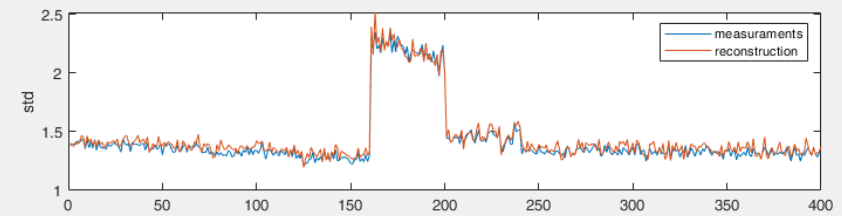
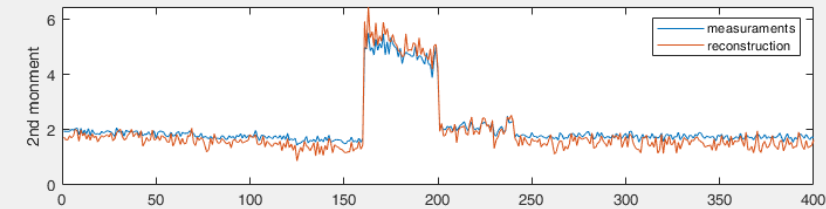
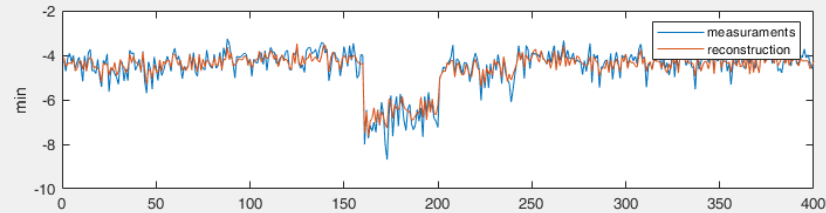
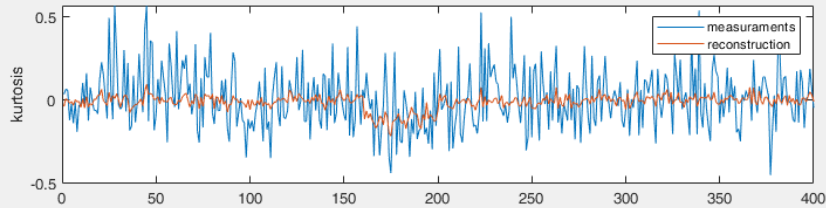
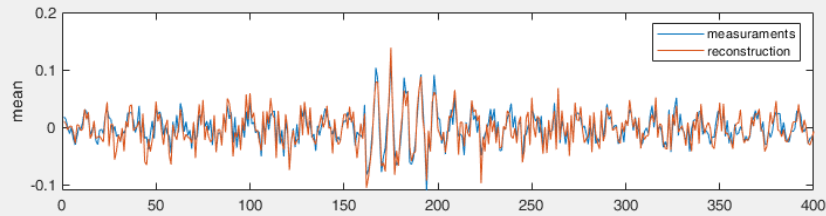
Apply the developed tool to the data:

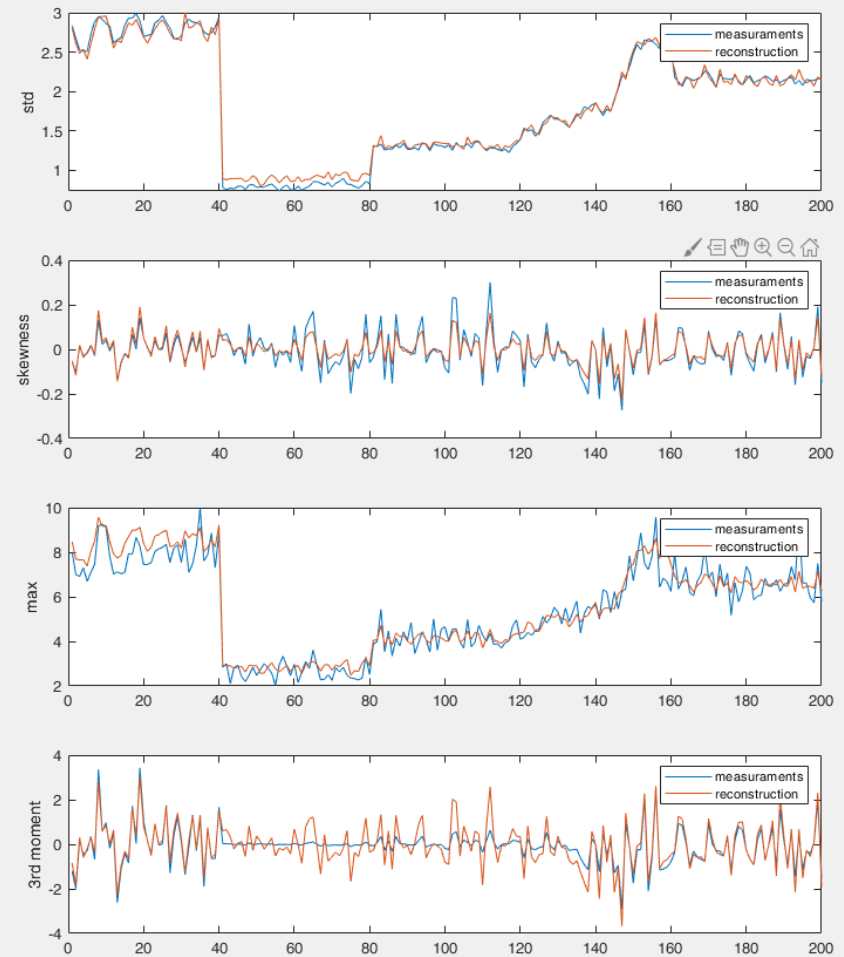
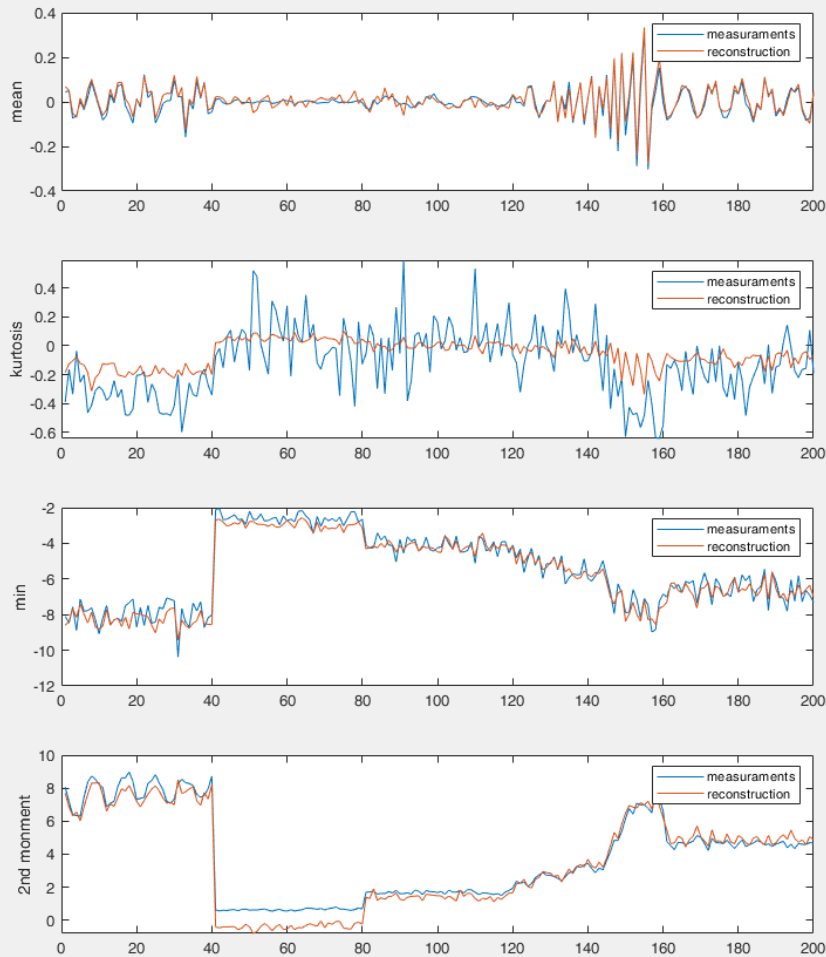
- test1
- test2



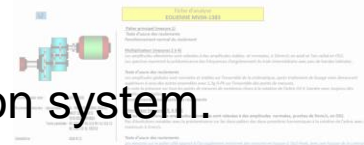
Test 1 → Normal Conditions

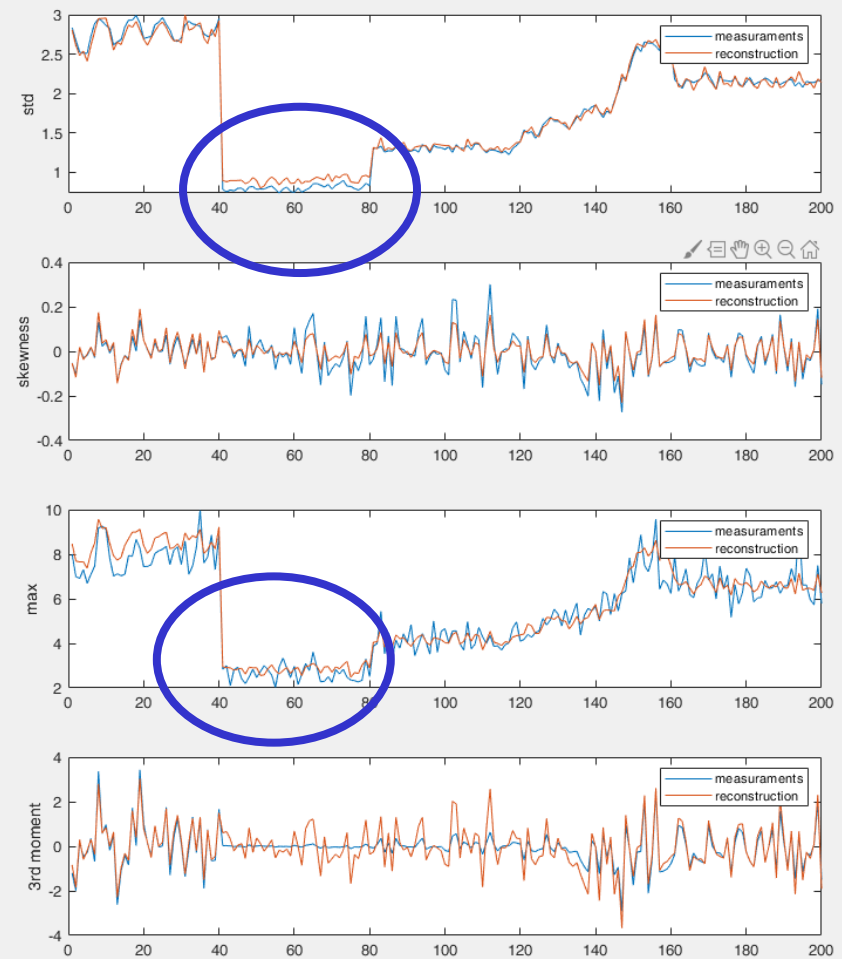
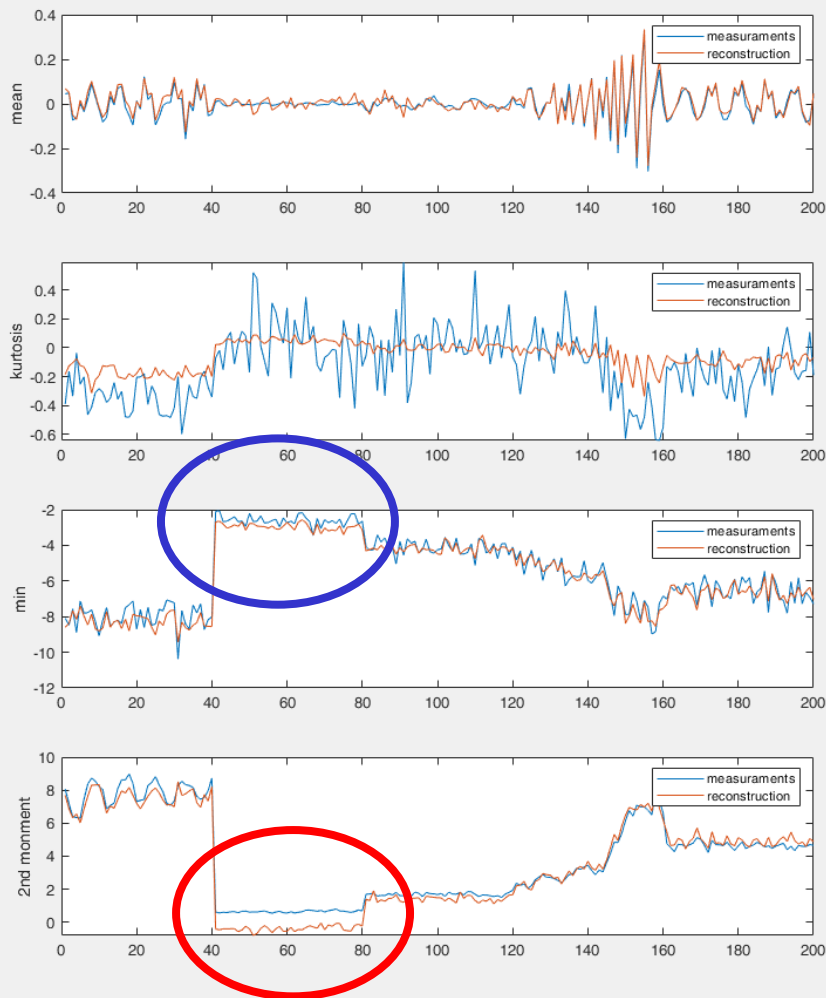
66



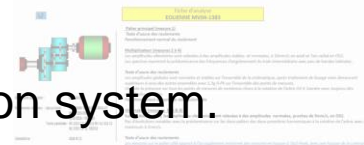


According to the report:
Acceptable operation of bearings but needs check of the lubrication system.





According to the report:
Acceptable operation of bearings, but needs check of the lubrication system.





SPRT: 2nd moment

69

SPRT('signal7_test2_meas.dat','signal7_test2_rec.dat', 0.01,0.01,1.5,0.2)

