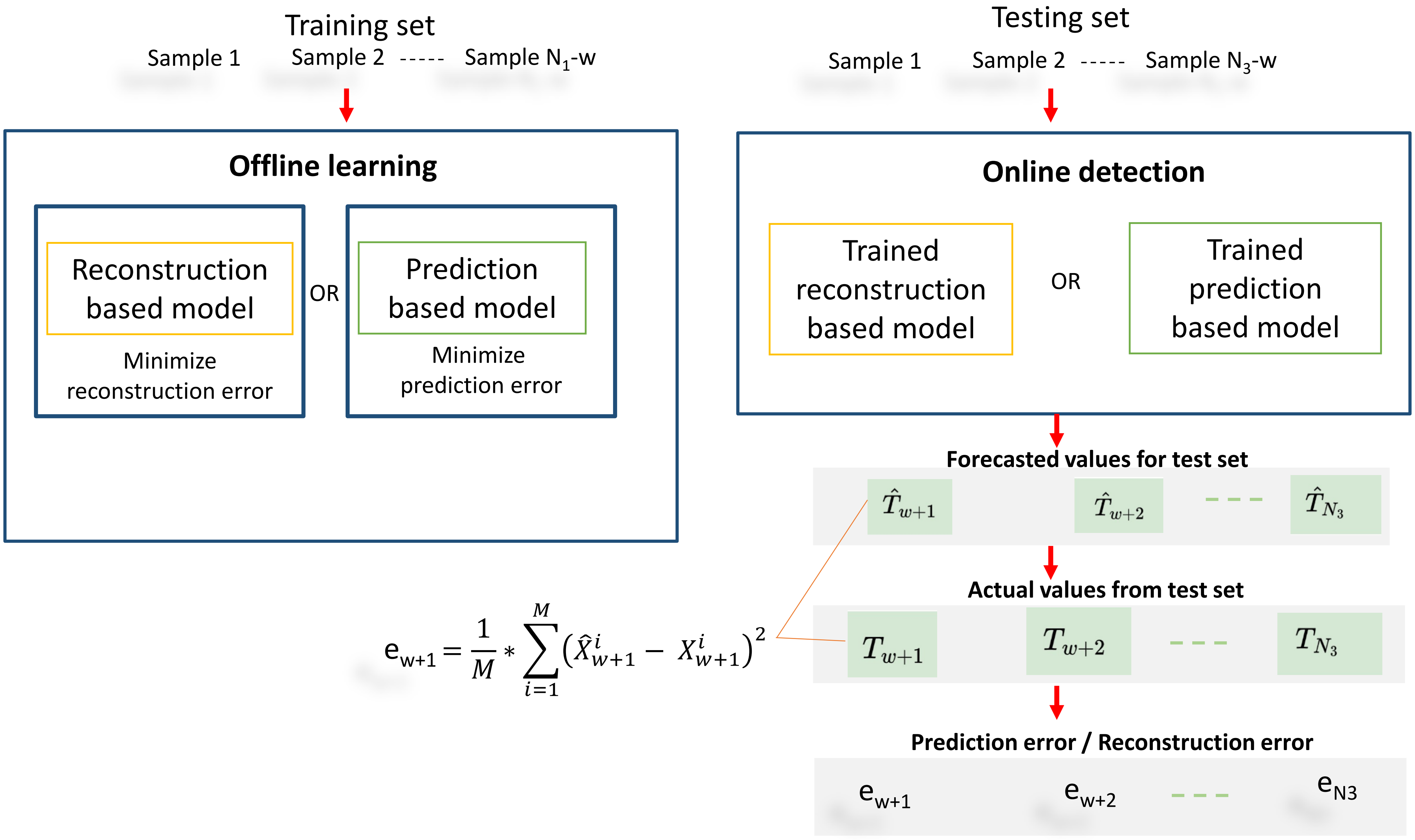
Introduction
to
Time Series

Class - 2

Semi-supervised learning approach for anomaly detection



$$e_{w+1} = \frac{1}{M} * \sum_{i=1}^{M} \left(\hat{X}_{w+1}^{i} - X_{w+1}^{i}\right)^2$$

Reconstruction/ Prediction error

Prediction error / Reconstruction error

e_{w+1} e_{w+2} - - - e_{N3}

Given ground truth for test set

Point

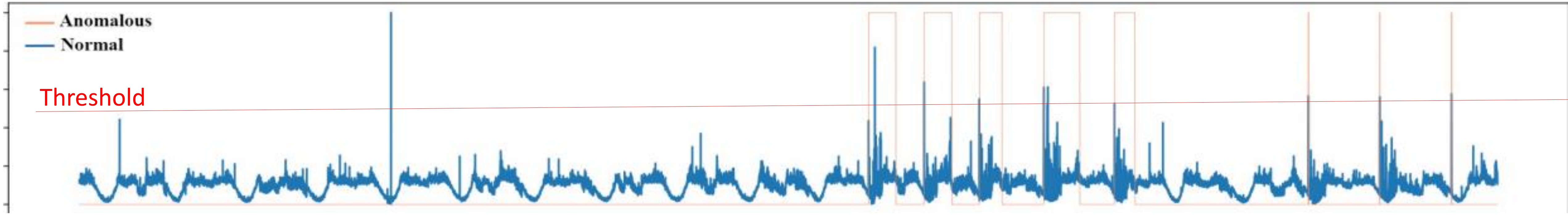
T_{w+1} - - - T_{N_3}

0 0 0 1 - - - 1 0

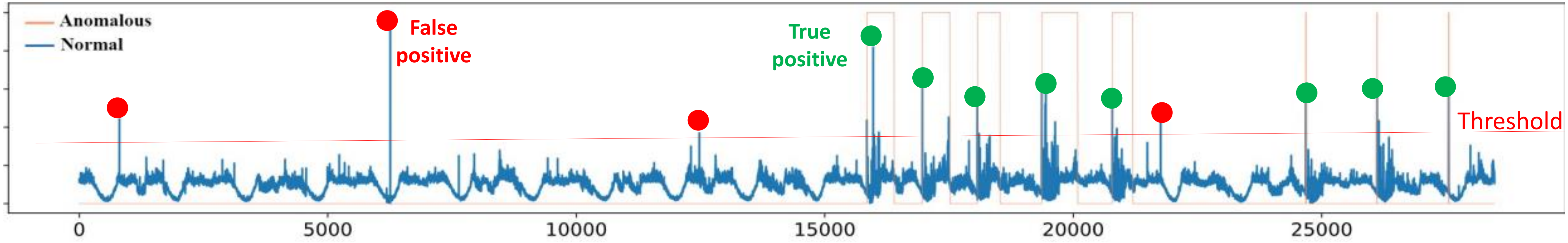
Sequence

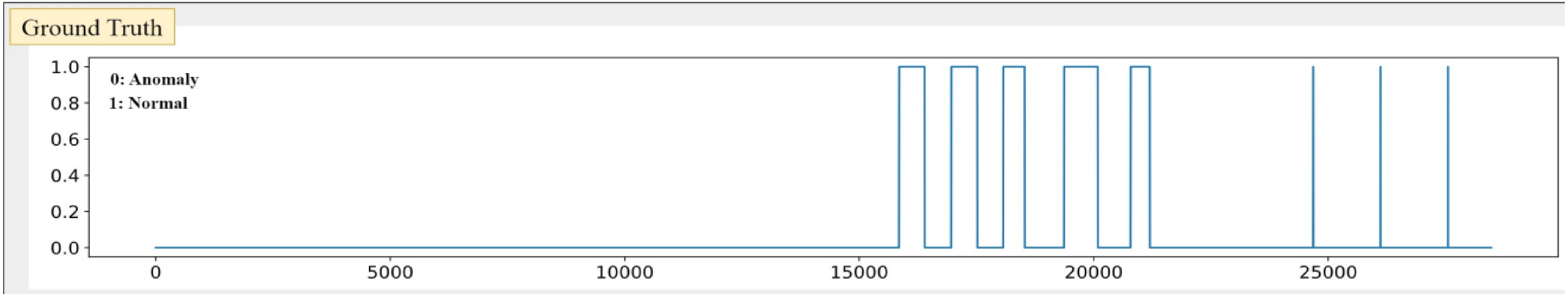
[[837, 858], [2959, 4174],…….]

Prediction error



Pointwise and sequence wise anomaly detection





Point wise detection :

- **Total anomalies** – 10600
- **True positives** - 5600
- **False positives** – 600
- **False negatives** – (10600-5600) =5000
- **Precision** – .87, **Recall** - .53 , **F1score** - .65

Sequence wise detection :

- **Total anomalies** – 8
- **True positives** - 8
- **False positives** – 4
- **False negatives** – (8-8) =0
- **Precision** – .67, **Recall** - 1 , **F1score** - .8

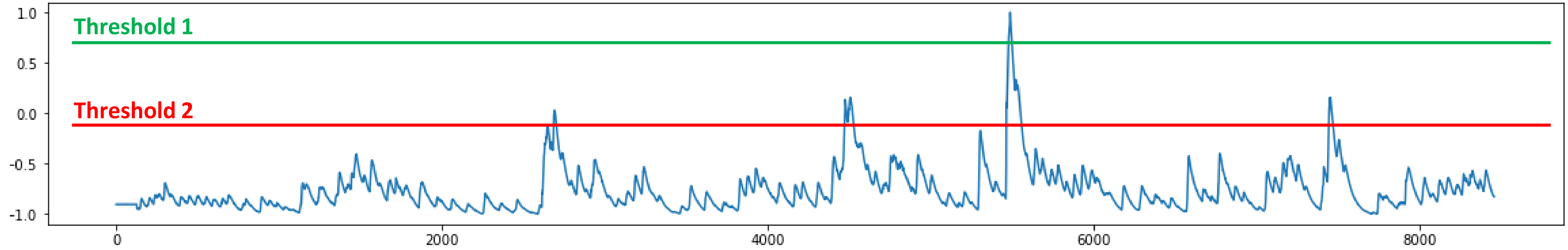
Thresholding techniques

- **Static thresholding**
  - Fixed threshold for test set to decide whether an observation is an anomaly
  - If the prediction error exceeds a certain threshold, then the observation is classified as an anomaly

- **Maximum of training error**

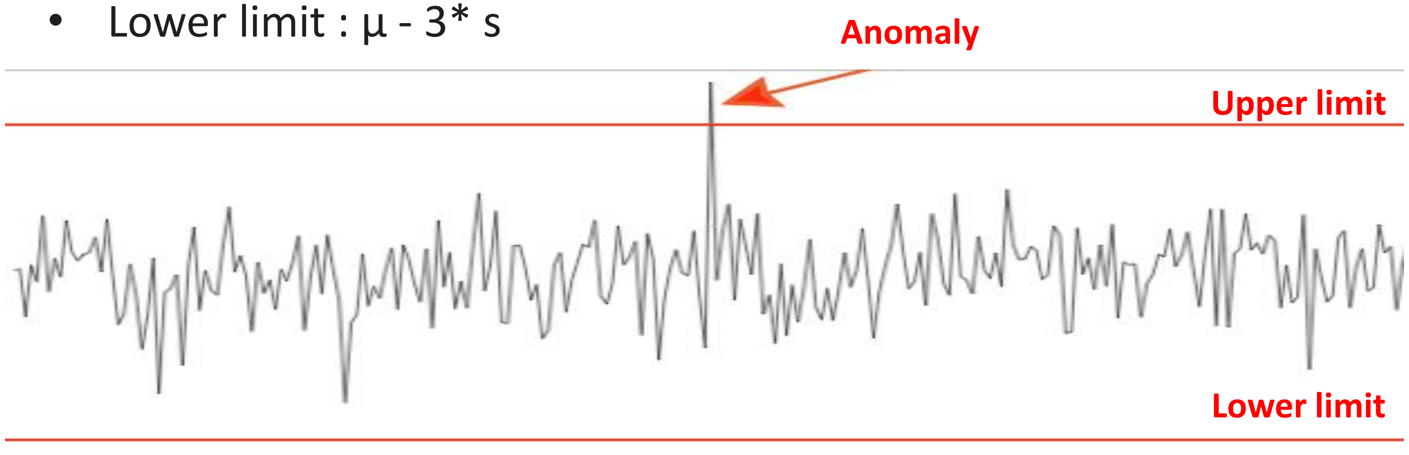
$$\text{Maximum}\,(e_{w+1},\ e_{w+2},\ ---,\ e_{N1})$$

- **Brute force search for best threshold value**
  - Start from highest error value and decrease gradually to find threshold value which gives best F1-score



Static thresholding

- **Using mean (μ) and standard deviation (s)**
  - Upper limit : μ + 3* s
  - Lower limit : μ - 3* s



- **Z-score with upper limit 3 and lower limit -3**

$$z = (x - \mu)/\sigma$$

- **Using percentile**
  - Lower limit : 2.5th percentile
  - Upper limit : 97.5th percentile

$$2.5 * \frac{N_{3-w} +1}{100} \text{th value}$$

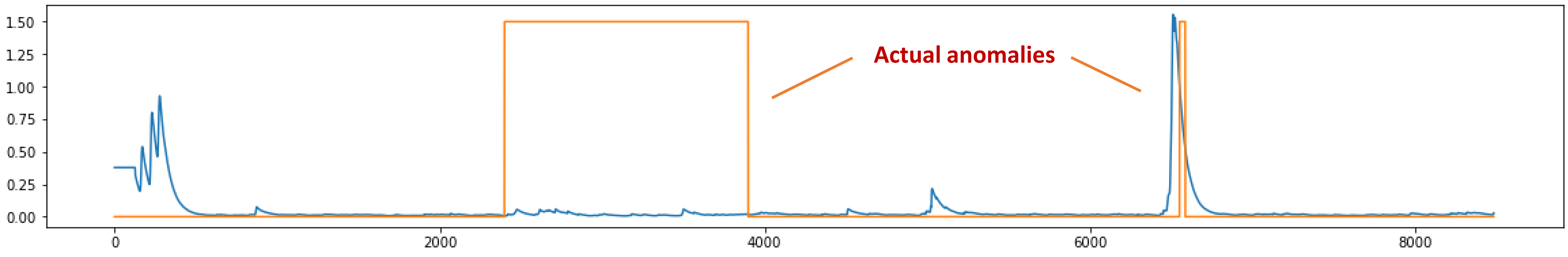
$$97.5 * \frac{N_{3-w} +1}{100} \text{th value}$$

Static thresholding

- **Using interquartile range :**
  - First quartile / one-fourth quartile [Q_{25}]: 25th percentile
  - Third quartile / three-fourth quartile [Q_{75}]: 75th percentile
  - Inter quartile range [IQR] : Q_{75} - Q_{25}
  - Lower limit : Q_{25} - 1.5 * IQR
  - Upper limit : Q_{75} + 1.5 * IQR

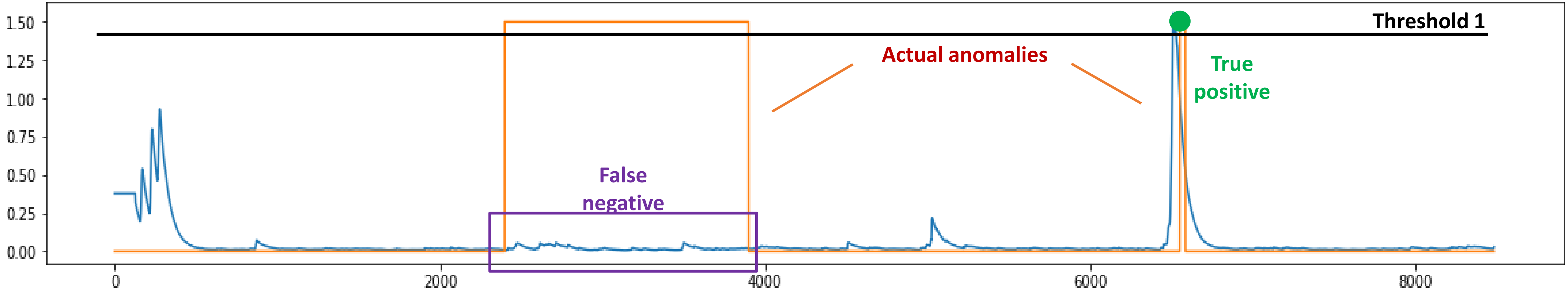
- **Drawback of static thresholding :**
  - Not able to adapt to changes in the data over time

Example:

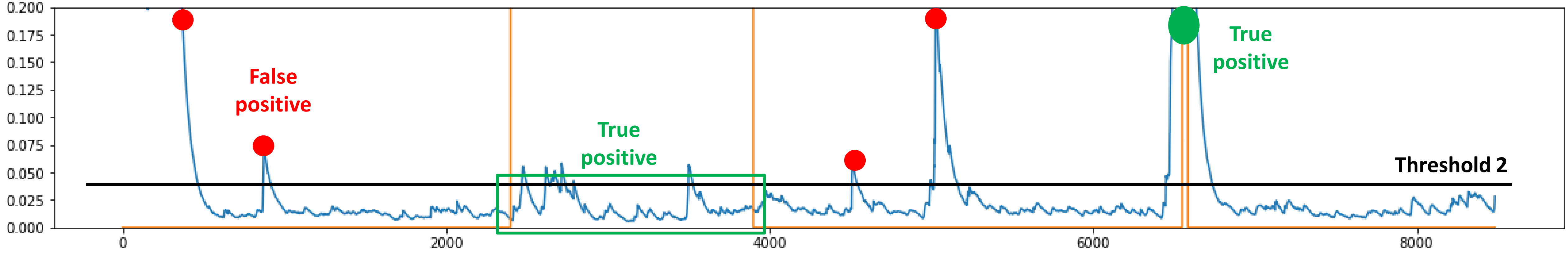


Rolling averaged prediction error for a multivariate test set

Static thresholding

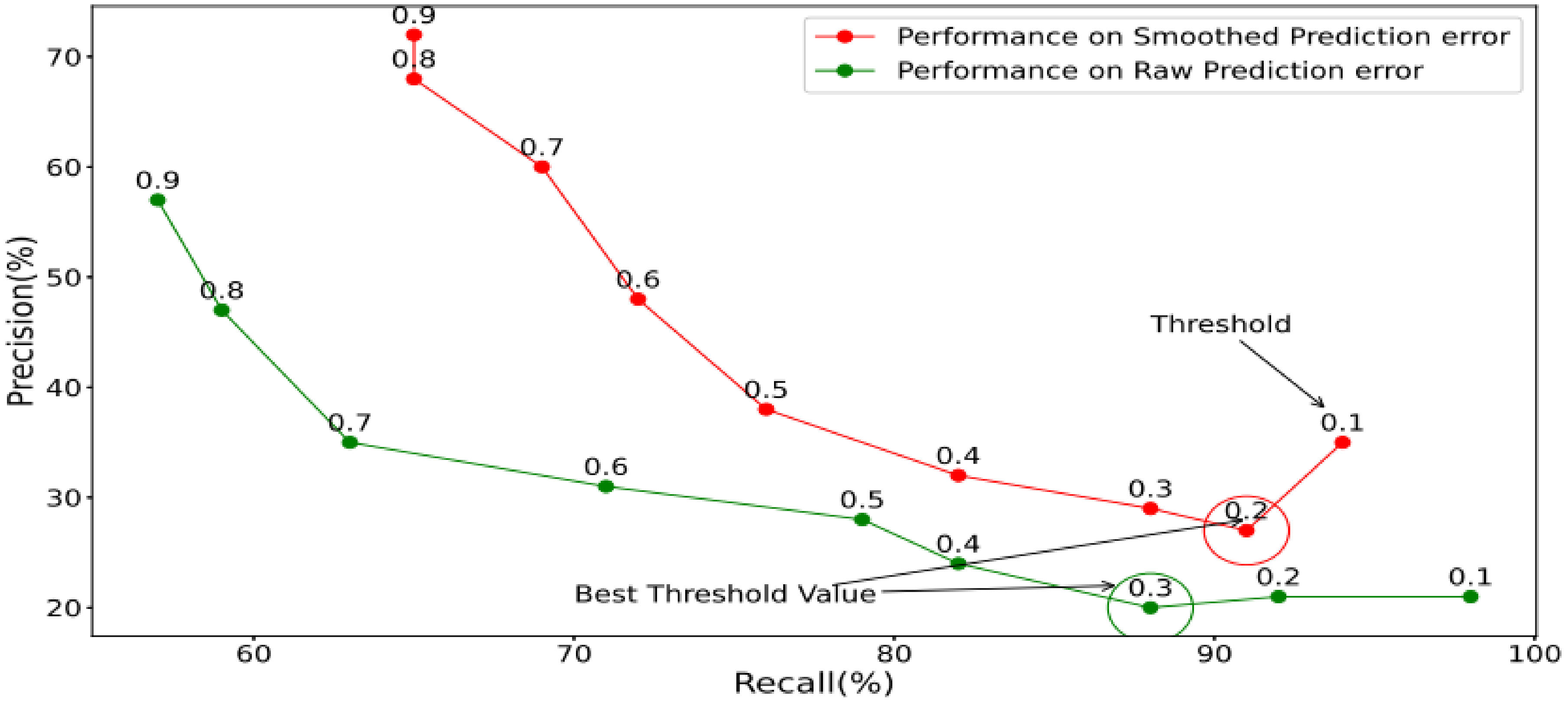


Zoom In



Static thresholding

- Best threshold value for brute force searching



Dynamic thresholding

Dynamic thresholding

- Apply thresholding on non overlapping window over time

- **Nonparametric dynamic thresholding**

$E = [\, e_{w+1} \quad e_{w+2} \quad - - - \quad e_{N3} \,]$ **Prediction error / Reconstruction error**

↓

$e = [\, E_{t-h} \quad - - - \quad e_t \,]$ **A subsequence of length h**

↓

Generate the smoothed errors (e_s)

↓

threshold ϵ selected from the set $\epsilon = \mu(\mathbf{e}_s) + \mathbf{z}\sigma(\mathbf{e}_s)$

Dynamic thresholding

Where ϵ is determined by:

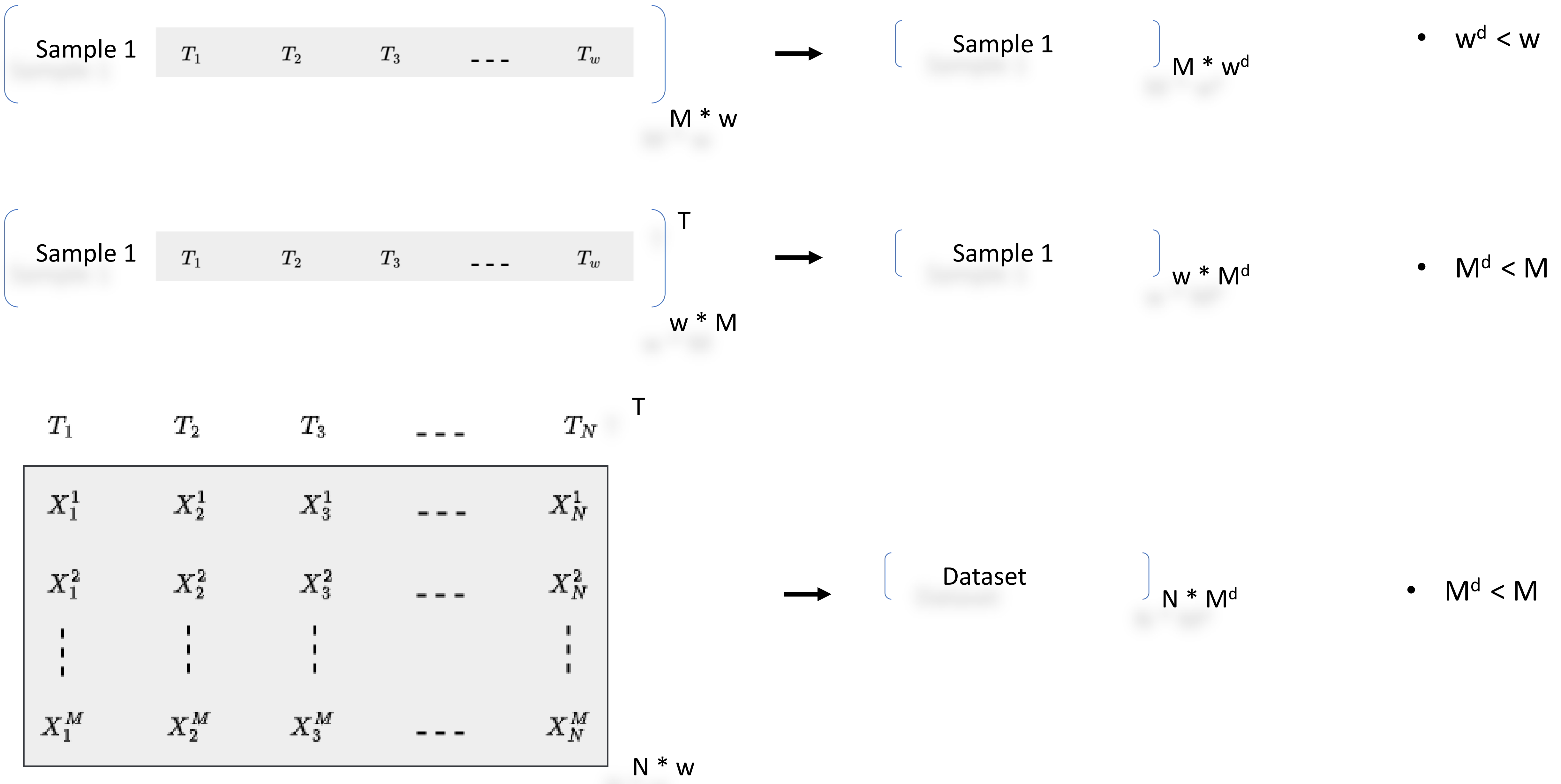
$$\epsilon = argmax(\epsilon) = \frac{\Delta\mu(\mathbf{e}_s)/\mu(\mathbf{e}_s)) + (\Delta\sigma(\mathbf{e}_s)/\sigma(\mathbf{e}_s)}{|\mathbf{e}_a| + |\mathbf{E}_{seq}|^2}$$

Such that:

$$\Delta\mu(\mathbf{e}_s) = \mu(\mathbf{e}_s) - \mu(\{e_s \in \mathbf{e}_s | e_s < \epsilon\})$$
$$\Delta\sigma(\mathbf{e}_s) = \sigma(\mathbf{e}_s) - \sigma(\{e_s \in \mathbf{e}_s | e_s < \epsilon\})$$
$$\mathbf{e}_a = \{e_s \in \mathbf{e}_s | e_s > \epsilon\}$$
$$\mathbf{E}_{seq} = \text{continuous sequences of } e_a \in \mathbf{e}_a$$

- A threshold is found that, if all values above are removed, would cause the greatest percent decrease in the mean and standard deviation of the smoothed errors

Dimensionality reduction



Principal component analysis

Data :

X	Y
1.4	0.3
1.6	0.2
1.4	0.2
1.5	0.2
1.4	0.2
4.7	1.4
4.5	1.5
4.9	1.5
4.0	1.3
4.6	1.5

Principal component analysis

Step 1- Compute Covariance Matrix

Covariance Matrix for X and Y :

$$\begin{pmatrix} Cov(x,x) = Var(x) & Cov(x,y) \\ Cov(y,x) & Cov(y,y) = Var(y) \end{pmatrix}$$

Where,

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N-1}$$

Principal component analysis

X	Y	A=X – Mean (X)	B =Y – Mean(Y)	AB	A^2	B^2
1.4	0.3	-1.6	-0.53	.848	2.56	.281
1.6	0.2	-1.4	-0.63	.882	1.96	.397
1.4	0.2	-1.6	-0.63	1.008	2.56	.397
1.5	0.2	-1.5	-0.63	.945	2.25	.397
1.4	0.2	-1.6	-0.63	1.008	2.56	.397
4.7	1.4	1.7	0.57	.969	2.89	.325
4.5	1.5	1.5	0.67	1.005	2.25	.449
4.9	1.5	1.9	0.67	1.273	3.61	.449
4.0	1.3	1.0	0.47	.47	1.0	.221
4.6	1.5	1.6	0.67	1.072	2.56	.449
Mean=3.0	**Mean=0.83**			**Sum=9.480**	**Sum=24.2**	**Sum=3.762**

Covariance Matrix :

$$\begin{pmatrix} A^2 / N-1 & AB / N-1 \\ BA / N-1 & B^2 / N-1 \end{pmatrix} = \begin{pmatrix} 2.689 & 1.053 \\ 1.053 & 0.418 \end{pmatrix}$$

Principal component analysis

Step 2 - Eigen Decomposition

Covariance Matrix (C) :

$$\begin{pmatrix} 2.689 & 1.053 \\ 1.053 & 0.418 \end{pmatrix}$$

Eigen value of covariance matrix = **Det (C – λ I) = 0**

$$=> \begin{pmatrix} 2.689 - \lambda & 1.053 \\ 1.053 & 0.418 - \lambda \end{pmatrix} = 0$$

$$=> \lambda^2 - 3.107\,\lambda + 0.015 = 0$$

$$=> \lambda_1 = 0.005 \qquad \lambda_2 = 3.102$$

Corresponding eigen vectors are =

$$V_1 = \begin{pmatrix} -.365 \\ .931 \end{pmatrix} \qquad V_2 = \begin{pmatrix} 0.931 \\ 0.365 \end{pmatrix}$$

Principal component analysis

Sort eigen values in decreasing order :

$\lambda_1 = 3.102 \qquad \lambda_2 = 0.005$

Corresponding eigen vectors :

$$V_1 = \begin{bmatrix} 0.931 \\ 0.365 \end{bmatrix} \qquad V_2 = \begin{bmatrix} -.365 \\ .931 \end{bmatrix}$$

Principal component analysis

Step 3 – Compute Principal component

First Principal Component : Dot product (Mean centered data , eigen vector ^{T})

A=X – Mean (X)	B =Y – Mean(Y)
-1.6	-0.53
-1.4	-0.63
-1.6	-0.63
-1.5	-0.63
-1.6	-0.63
1.7	0.57
1.5	0.67
1.9	0.67
1.0	0.47
1.6	0.67

$$\begin{bmatrix} 0.931 \\ 0.365 \end{bmatrix} =$$

First Principal Component
-1.683
-1.533
-1.72
-1.626
-1.72
1.791
1.641
2.013
1.103
1.734

Principal component analysis

To Get Original Data Back

Original data : **Dot product (Principal components , eigen vector) + Mean**

First Principal Component
-1.683
-1.533
-1.72
-1.626
-1.72
1.791
1.641
2.013
1.103
1.734

· **eigen vector** (0.931 0.365) =

X’	Y’
-1.567	-0.615
-1.427	-0.56
-1.601	-0.628
-1.514	-0.594
-1.601	-0.628
1.667	0.654
1.528	0.599
1.874	0.735
1.026	0.403
1.614	0.633

+ Mean →

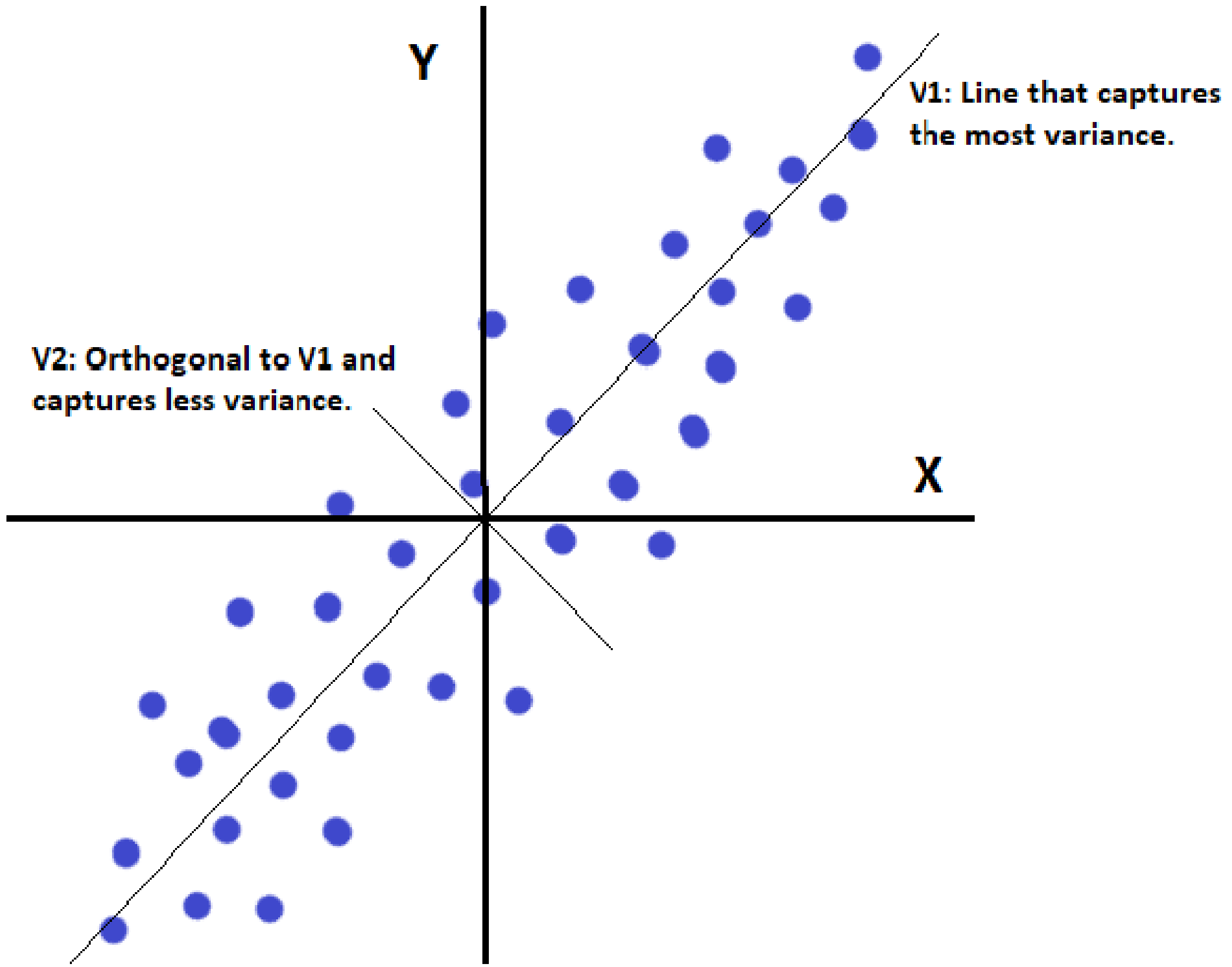
X’ + Mean (x)	Y’ + Mean (Y)
1.433	.215
1.573	.27
1.399	.202
1.486	.236
1.399	.202
4.667	1.484
4.528	1.429
4.874	1.565
4.026	1.233
4.614	1.463

Original data

Reconstruction Error :

Mean square error = [.001 .004]

Principal component analysis



Singular value decomposition

Singular Value Decomposition :

$$A = U\,S\,V^T \longrightarrow$$

Dimensionality reduction	Size of U	Size of S	Size of V
None	(m, n)	(n, n)	(n, m)
1D	(m, 1)	(1, 1)	(1, m)
2D	(m, 2)	(2, 2)	(2, m)
...	...	...	...

- **Given matrix :**

$$A = \begin{pmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

- **Compute U :**

$$AA^T = \begin{pmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 & 0 & 0 \\ 4 & 3 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 20 & 14 & 0 & 0 \\ 14 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Eigenvalues of the matrix AA^T = $\lambda=0$, $\lambda=0$; $\lambda = 29.883$; $\lambda = 0.117$

Eigen vectors of the matrix AA^T (U) = $\begin{pmatrix} .82 & -.58 & 0 & 0 \\ .58 & .82 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

- **Compute V :**

$$A^T A = \begin{pmatrix} 2 & 1 & 0 & 0 \\ 4 & 3 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 4 \\ 1 & 3 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 5 & 11 \\ 11 & 25 \end{pmatrix}$$

Eigen vectors of the matrix $A^T A$ (V) = $\begin{pmatrix} .40 & -.91 \\ 91 & .40 \end{pmatrix}$

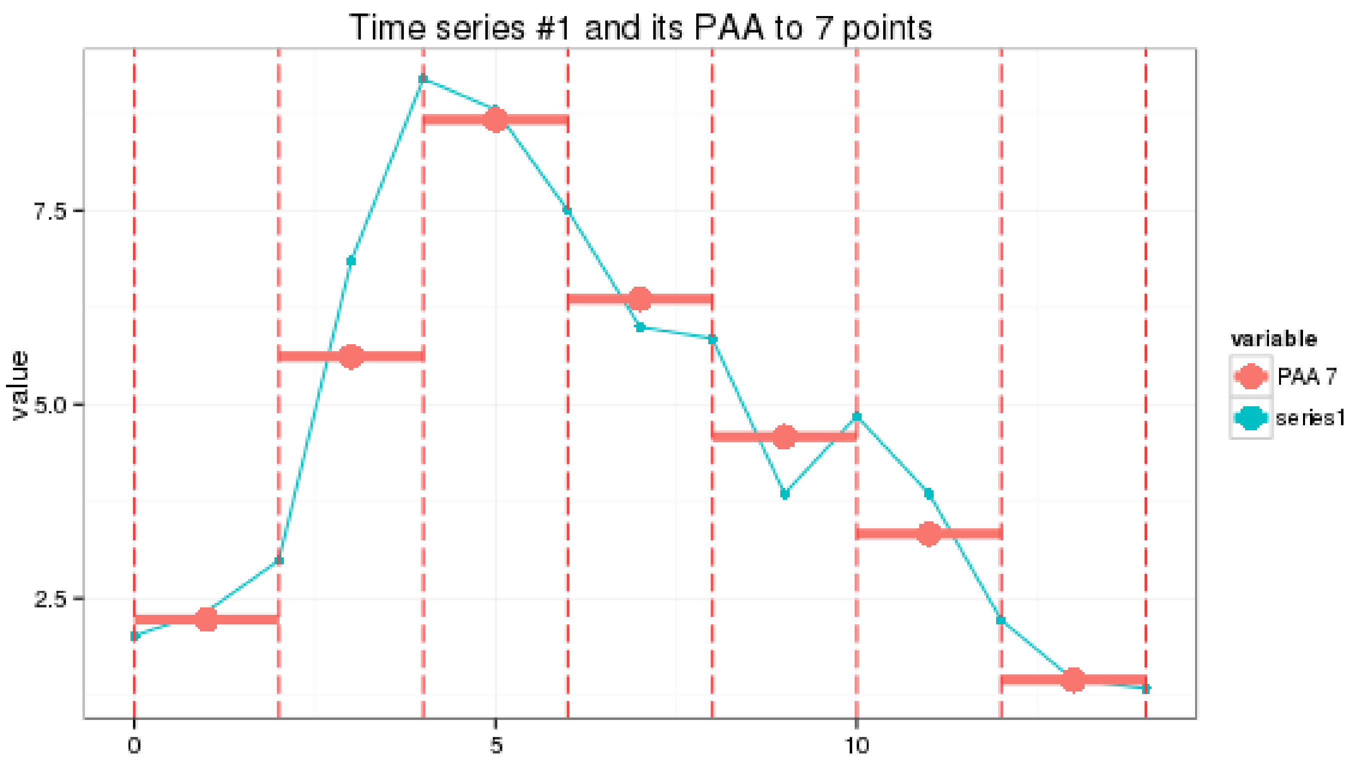
- **Compute S :**

  Square root of the eigenvalues :

$$\begin{pmatrix} 5.47 & 0 \\ 0 & 0.37 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}$$

- **Dimensionality reduction**

Piecewise aggregation approximation



Feature extraction

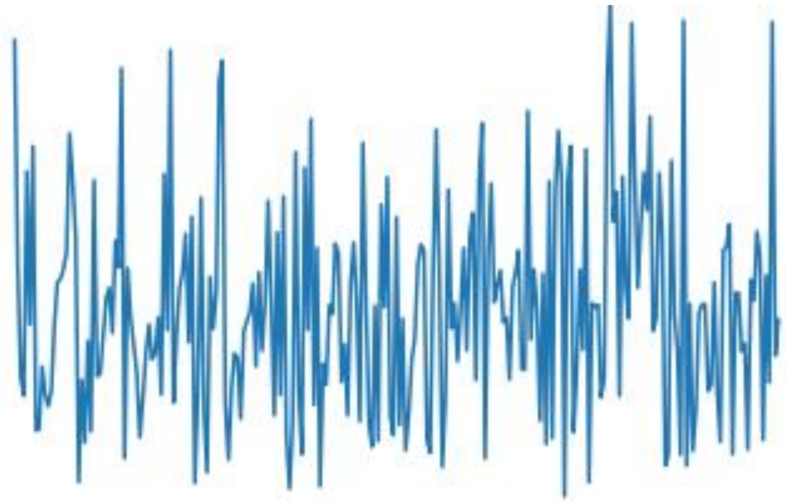
- Time domain features

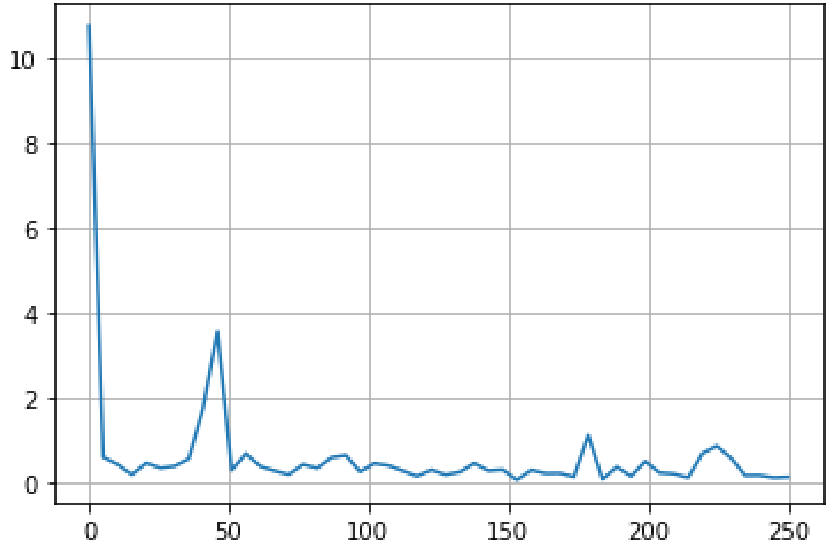
1	Mean	$T_m = \frac{1}{n}\sum_{i=1}^{n} x_i$	7	Shape factor	$T_{sf} = \frac{T_{rms}}{\bar{x}}$
2	Root mean square	$T_{rms} = \left[\frac{1}{n}\sum_{i=1}^{n} x_i^2\right]^{1/2}$	8	Crest factor	$T_{cf} = \frac{x_{\max}}{x_{rms}}$
3	Root	$T_r = \left[\frac{1}{n}\sum_{i=1}^{n} \lvert x_i\rvert^{1/2}\right]^2$	9	Impulse factor	$T_{if} = \frac{x_{\max}}{\bar{x}}$
4	Standard deviation	$T_{sd} = \left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2\right]^{1/2}$	10	Clearance factor	$T_{clf} = \frac{x_{\max}}{x_r}$
5	Skewness	$T_{sk} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^3}{(n-1)T_{sd}^3}$	11	Skewness factor	$T_{skf} = \frac{T_{sk}}{T_{rms}^3}$
6	Kurtosis	$T_{ku} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^4}{(n-1)T_{sd}^4}$	12	Kurtosis factor	$x_{kuf} = \frac{T_{ku}}{T_{rms}^4}$

- x - a univariate time series window of length *n* with timestamp index *i.*

Feature extraction

- Frequency domain features
  - Fourier transformation

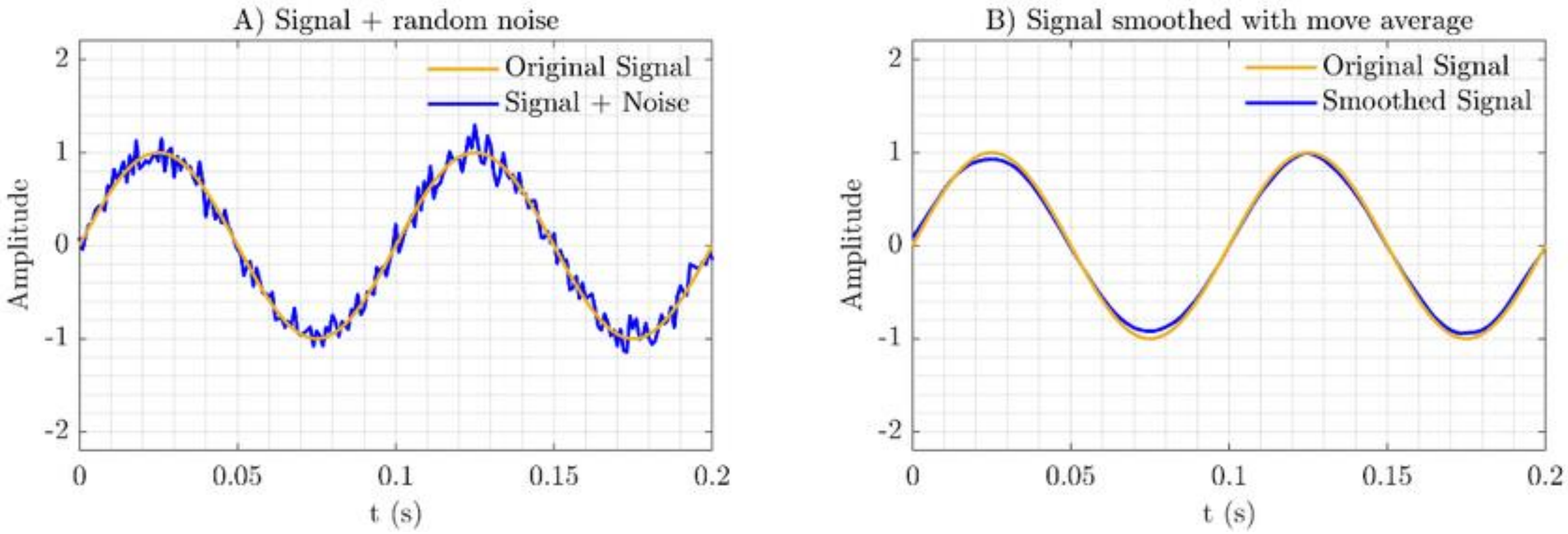




  - Peak power of frequency spectrum
  - Spectrogram of series

Removing noise from series

- $\hat{X}_1^i = \frac{1}{Ws} * \sum_{k=1}^{Ws} X_k^i$



- **Exponential smoothing :** Smoothed value = (Smoothing factor * Current value) + (1 - Smoothing factor) * Previous smoothed value. [Smoothing factor = a value between [0 to 1]]

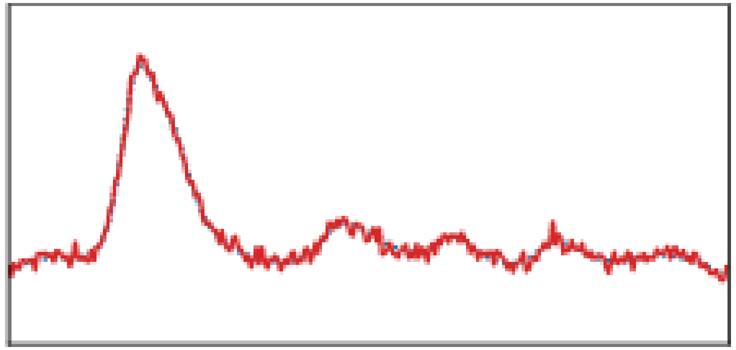
- **Median Filtering:**

Replace each data point with the median value within a local window [remove outliers] .

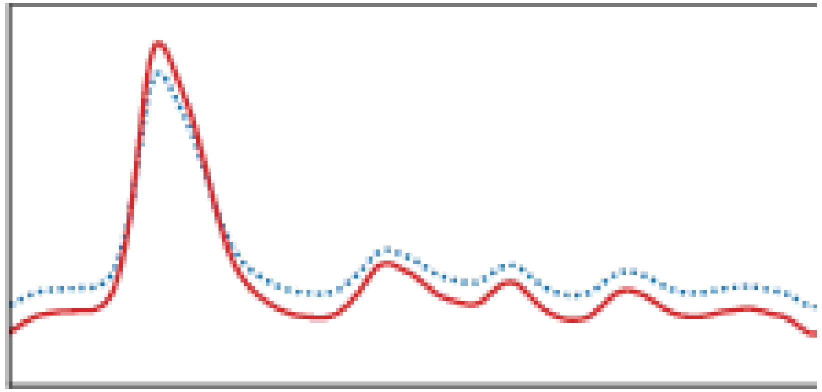
- **Autoencoders**

Data augmentation

- **Jittering :** Series of length n **+** random noise [$x_i = x_i + \mathcal{N}(0, \sigma).$]



- **Scaling :** Scaled value = Original value * Scale factor

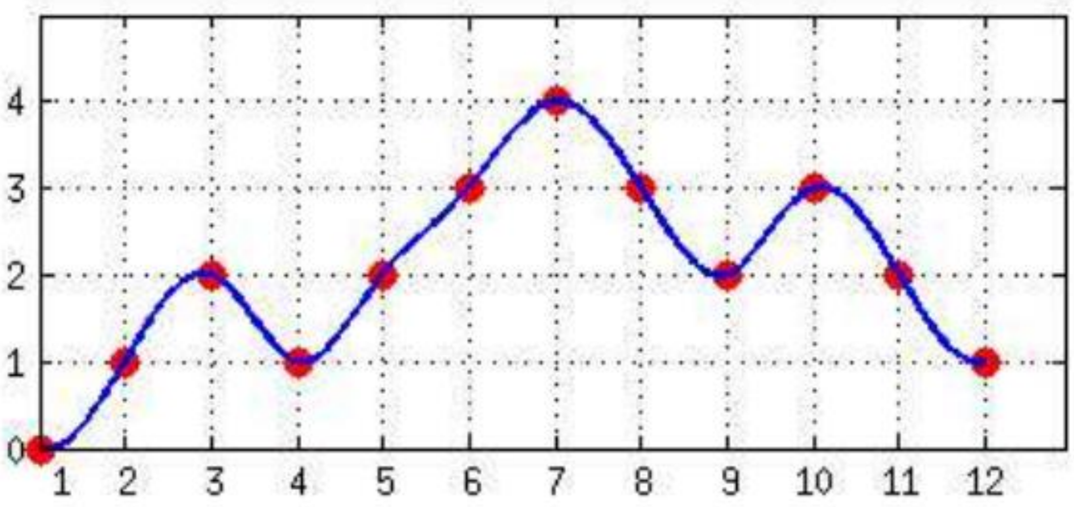


- **Shuffle / slice and shuffle**

Time series to image

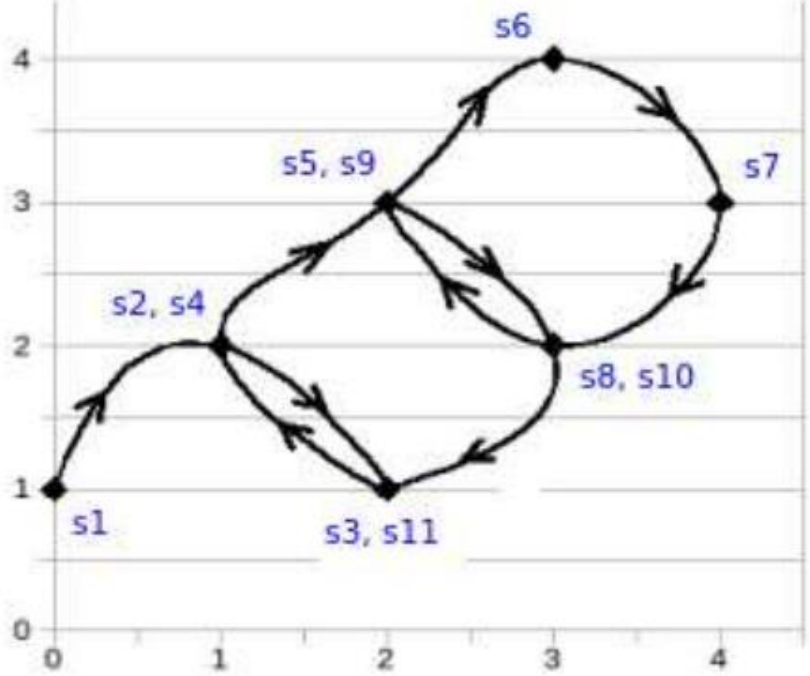
- **Recurrence plot**

Step 1



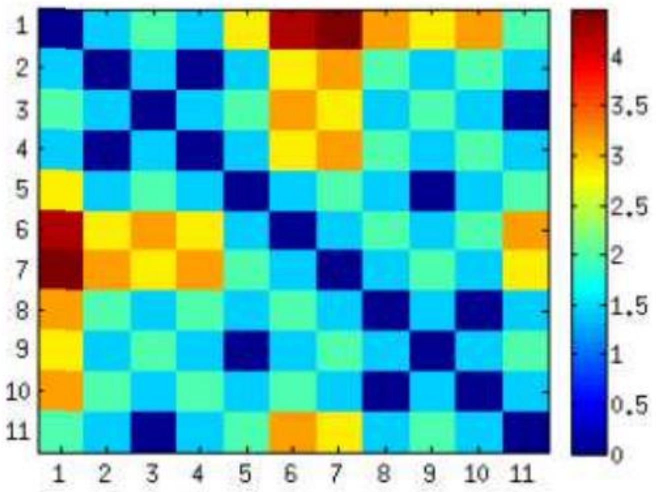
Time series (x) with 12 data points

Step 2



2D phase space trajectory of x (τ = 1)

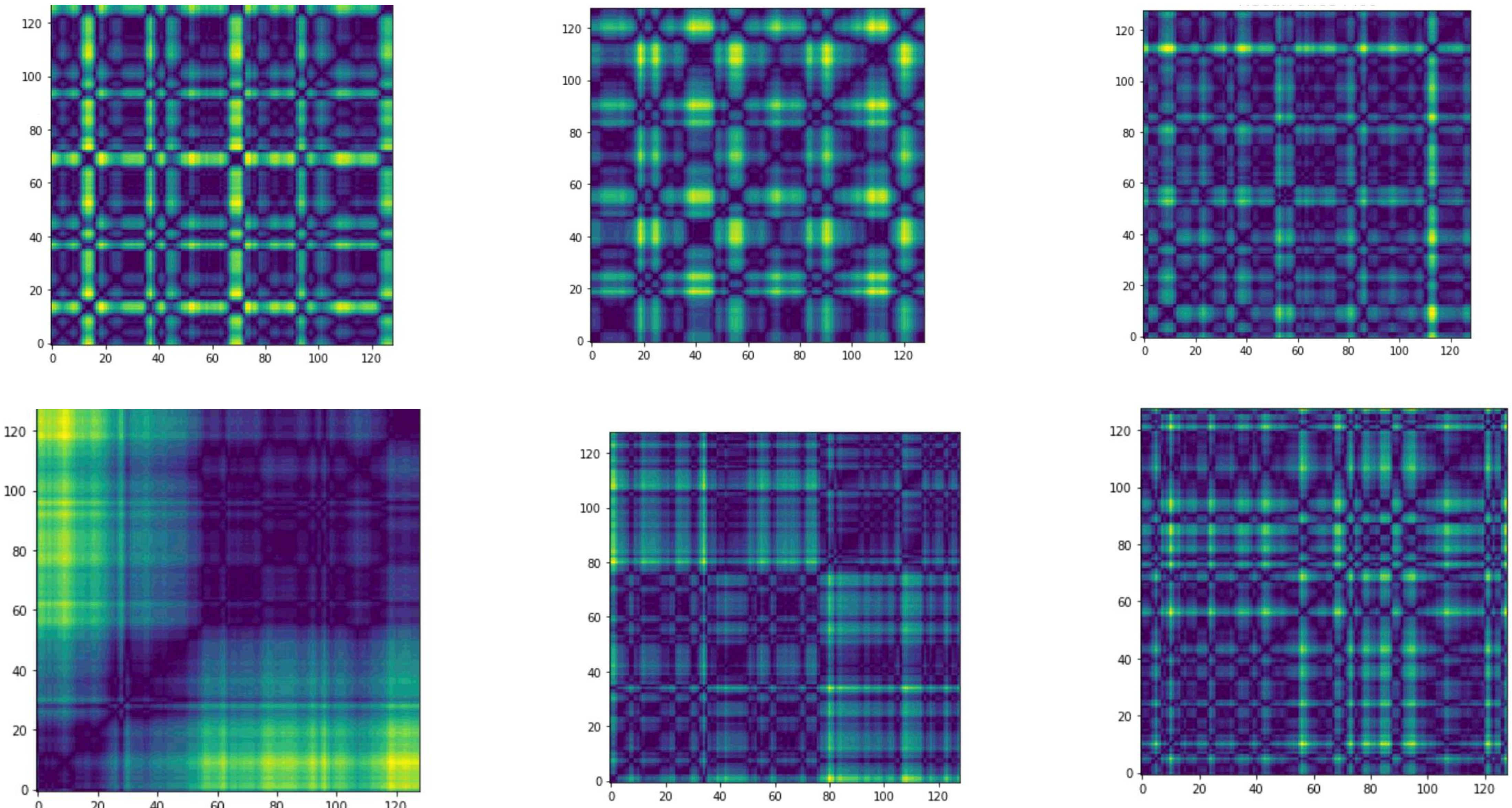
Step 3



11 × 11 square matrix (R) with Ri,j = dist(si,sj)

Time series to image

- **Recurrence plot for 6 different activity (acceleration data from human activity recognition dataset)**



Time series to image

- **Recurrence plot (variant)**

- consider a time series $X=\{x_1,x_2,\ldots\ldots\ldots,x_N\}$
- Create a Hankel matrix.

$$\begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ x_2 & x_3 & \cdots & x_{n+1} \\ \vdots & \vdots & & \vdots \\ x_m & x_{m+1} & \cdots & x_N \end{pmatrix}$$

- Dimensionality reduction to k dimension
- Create phase space trajectory of the k dimensional time series
- Create a recurrence plot (m x m matrix)

$$R_{i,j} = \theta(\epsilon - \|\vec{s}_i - \vec{s}_j\|),$$

Where,

ε = threshold distance, $\Theta(\cdot)$ = Heaviside function , $\|\vec{s}_i - \vec{s}_j\|$ = Euclidean norm

s = state , i, j = 1,2,……m

Time series to image

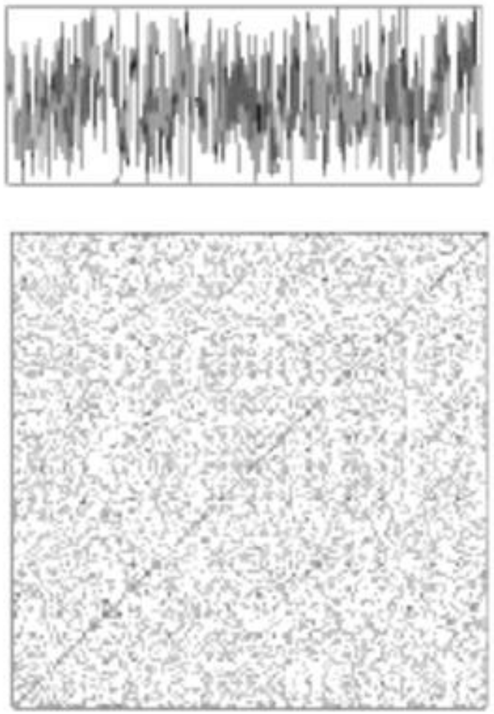


Fig.1- White noise

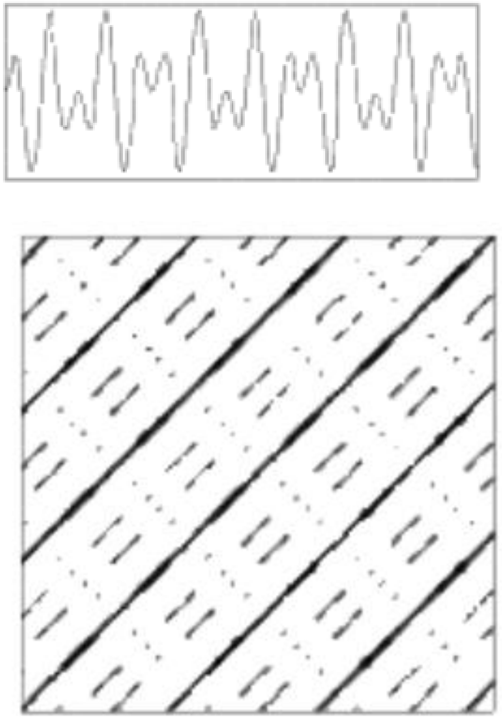


Fig.2- periodic recurrent structures

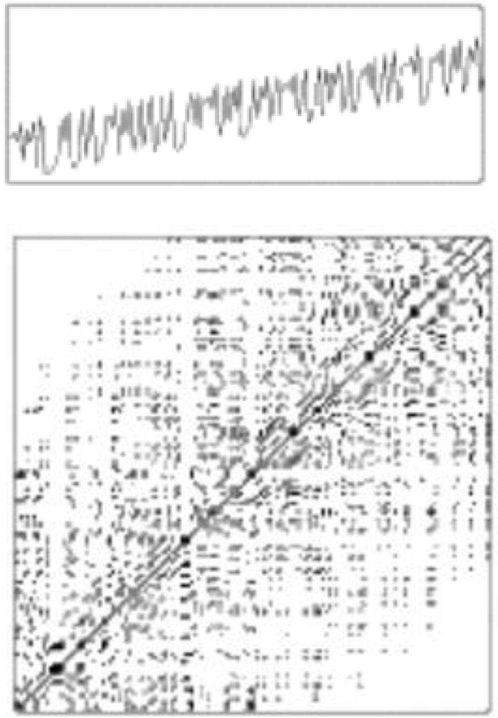


Fig.3- Non-stationary

Thank You