1495 UNIVERSITY OF ABERDEEN | CELEBRATING 525 YEARS 1495 – 2020

ABERDEEN 2040

Time Series – 1

Data Mining & Visualisation

Lecture 25

2025

Today...

- Time Series Data
- PAA
- SAX
- DTW

Types of Data

So far, we have covered various methods for analysing and mining datasets.

To demonstrate these methods, we've mostly been using reasonably clean, consistent, and tabular data.

However, not all data that you might want to analyse is tabular, or consistent, in nature.

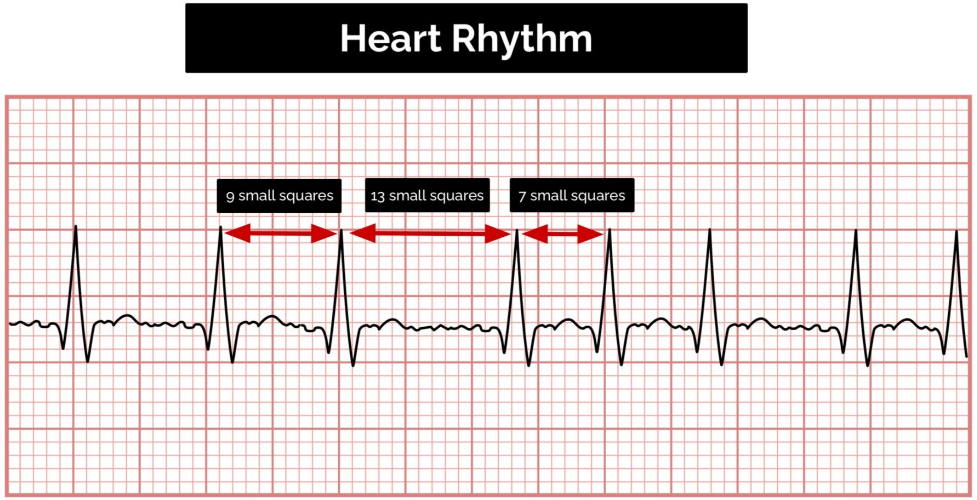
Types of Data

For the remaining lectures, we will explore analysis methods for a few different types of data that you might encounter.

In this lecture, we will start with *time series data*.

Time Series Data

Time Series Data



Time series data refers to a sequence of datapoints, collected or recorded at successive time intervals.

The key characteristic is that they are <u>time ordered</u>.

However, there will often be some correlations and patterns that emerge across successive observations.

Time Series Data

Time series data appears in many different contexts, across several domain areas:

Finance: Stock market analysis, economic forecasting, risk management.

Healthcare: Monitoring patient vital signs, predicting disease outbreaks.

Retail: Sales forecasting, inventory management, demand forecasting.

Meteorology: Weather forecasting, climate change modelling.

Manufacturing: Quality control, production planning, equipment maintenance schedules.

Time Series Data

There are various reasons that we might wish to be able to understand and analyse this data.

We might want to be able to **forecast** data, i.e. be able to predict future values based on past data.

We might want to understand patterns, to identify seasonal trends and cycles, to help in planning and decision-making.

We might want to perform **anomaly detection**, to identify unusual data that may indicate errors, extraordinary events, or opportunities for intervention.

Time Series Characteristics – Seasonality

There are several characteristics that we might see across time series data.

One such characteristic is its **seasonality**. I.e. is there a regularly repeating (cyclical) pattern of high and low values, related to the time?

This might be across business cycles, meteorological seasons, calendar months, days of the week, hours of the day, etc.

Time Series Characteristics – Seasonality

Example: Average daily temperature in London.

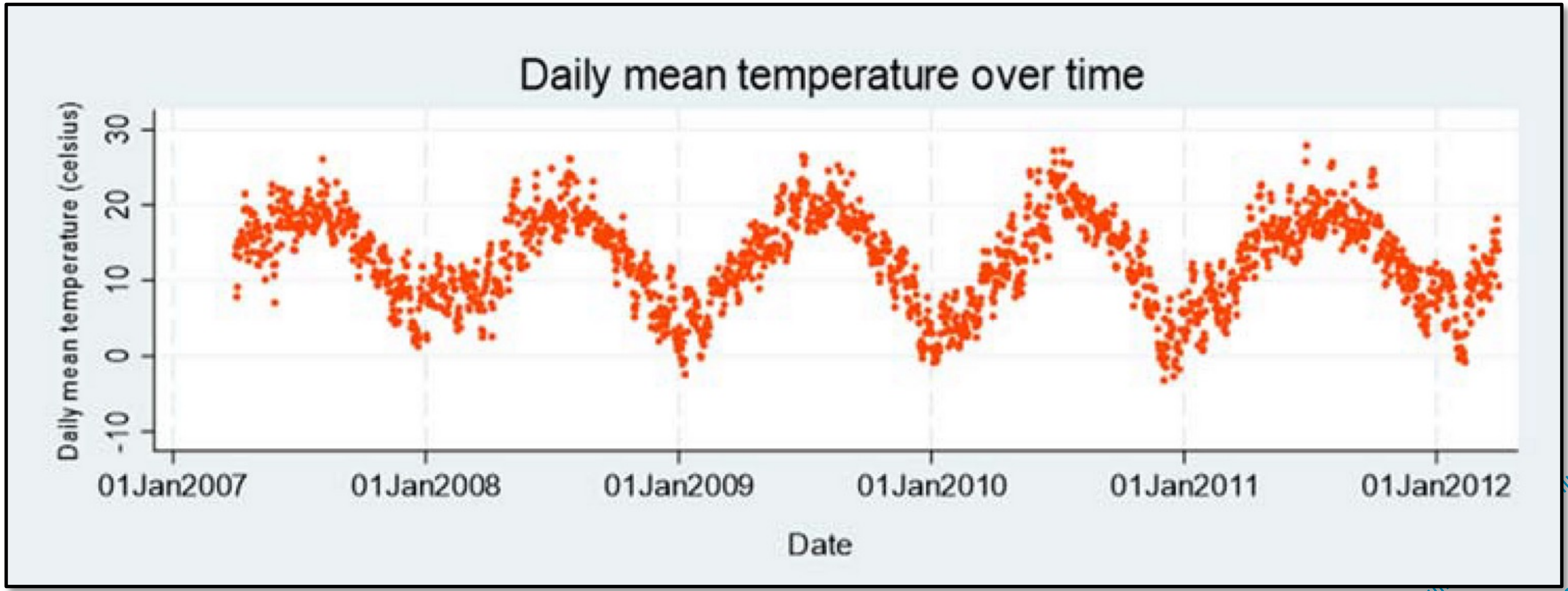


Image source: https://doi.org/10.3390/ijerph17061957

Time Series Characteristics – Trend

Another is the overall **trend** of the time series data.

Ignoring the *seasonality*, does the data trend upwards or downwards?

This gives us a picture of how a variable changes over time, ignoring seasonal fluctuations.

Time Series Characteristics – Trend

Example: Web searches for: ‘GPT’.

Source: Google Trends

Time Series Characteristics – Noise

Accounting for the *trend* and *seasonality* of time series data, how much **noise** (i.e. seemingly random variance) is exhibited within the data?

Time series data will often have some degree of noise, which can mask the underlying patterns that we are interested in.

Time Series Characteristics – Noise

Example: Heart rate data.

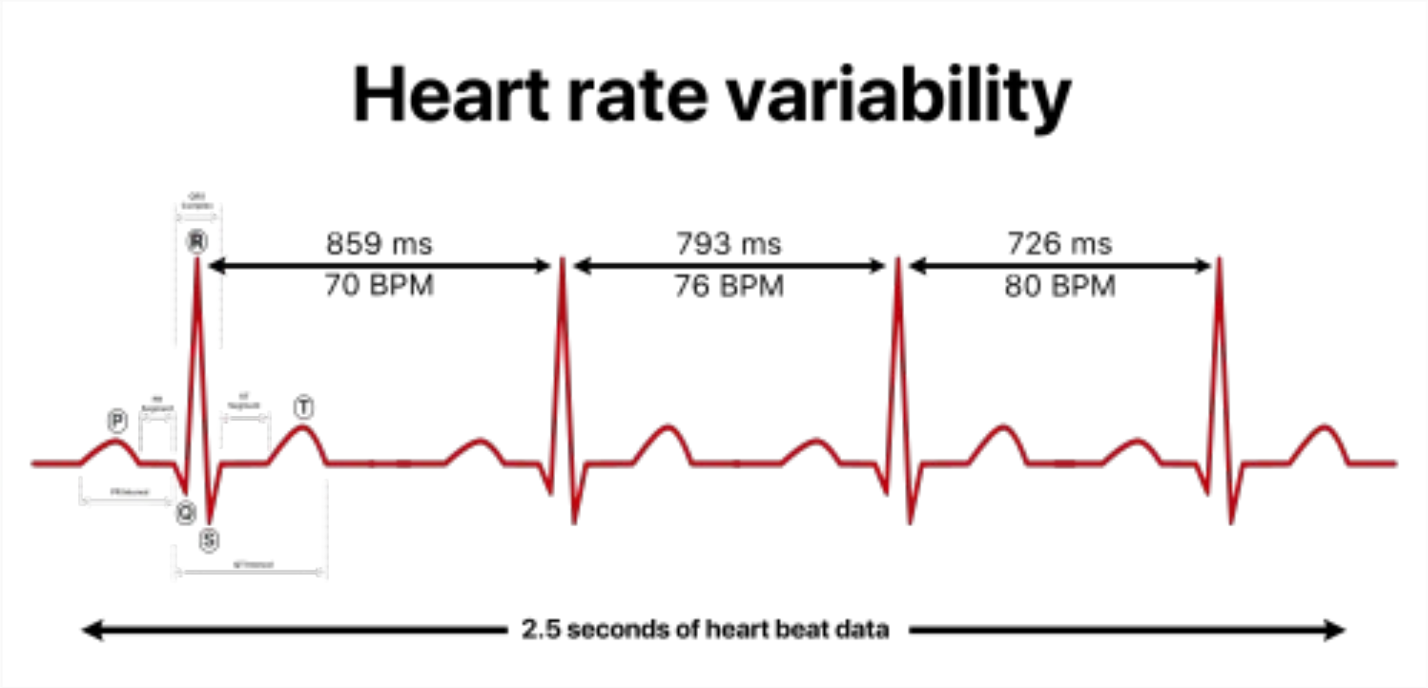


Image source: https://en.wikipedia.org/wiki/Heart_rate_variability

Time Series Characteristics

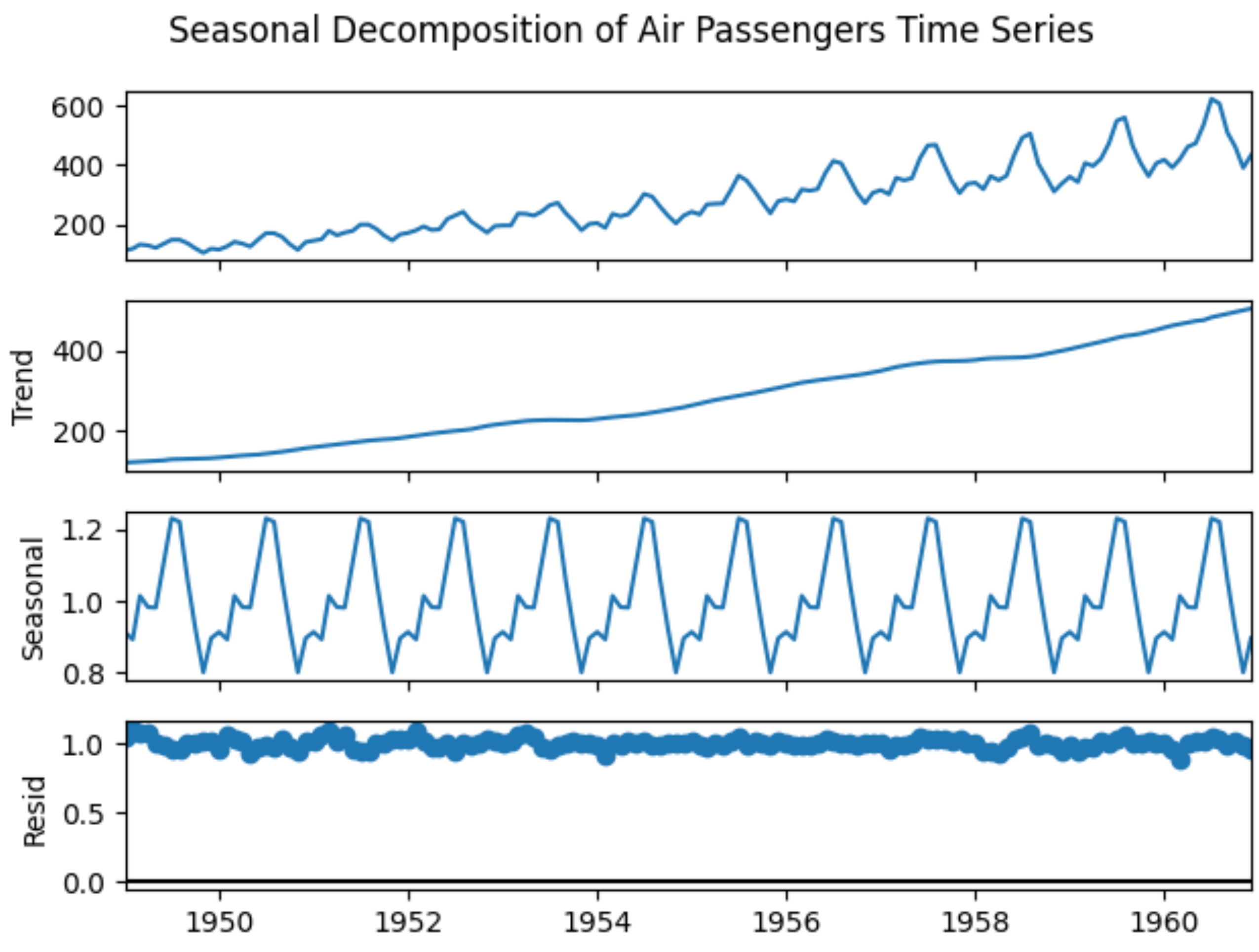


Image source: https://www.geeksforgeeks.org/seasonality-detection-in-time-series-data/

Time Series Characteristics – Stationarity

The **stationarity** of time series data refers to whether and how its statistical properties (e.g. mean, variance) change over time.

We refer to time series data as **stationary** when its statistical properties remain consistent over time.

Time series data exhibits **non-stationarity** when these statistical properties change over time.

Time Series Characteristics – Stationarity

Example: Stock prices over time.

Image source: https://www.cnbc.com/quotes/MSFT

Challenges of Analysing Time Series Data

These various factors mean that *effectively* analysing time series data can often be extremely challenging.

Complicating issues further is that time series data often exhibits **high-dimensionality**, in that it may have many thousands of time points.

Challenges of Analysing Time Series Data

Often, time series analysis is about accounting for this high-dimensionality.

In short, by performing some pre-processing, we can more easily analyse, compare, and understand time series data.

Time Series Analysis with PAA and SAX

ABERDEEN 2040

Time Series Analysis with PAA and SAX

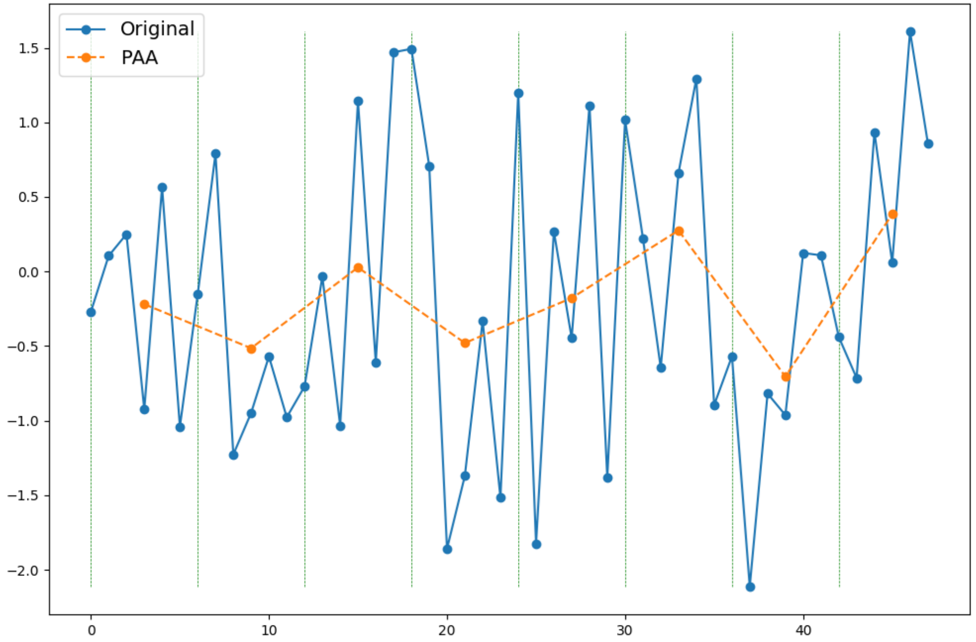
A relatively simple way of analysing time series data is through combining two concepts:

- Piecewise Aggregate Approximation (PAA), and
- Symbolic Aggregate Approximation (SAX)

Piecewise Aggregate Approximation (PAA)

Piecewise Aggregate Approximation (PAA) is a technique used to reduce the dimensionality of time series data.

It involves downsampling the data into fixed ‘segments’ such that, for each segment, the mean value is retained.



Piecewise Aggregate Approximation (PAA)

By reducing the dimensionality of the time series data, it makes it easier to analyze, visualize, and store.

By doing so, it offers:

- **Data Compression** – by reducing the size of the dataset while retaining its essential shape.
- **Noise reduction** – by smoothing out short-term fluctuations and highlighting underlying trends.
- **Efficiency** – by facilitating faster computation in analysis and pattern recognition tasks.

Piecewise Aggregate Approximation (PAA)

The PAA process is as follows:

Step 1: Divide the original time series into N equally sized segments.

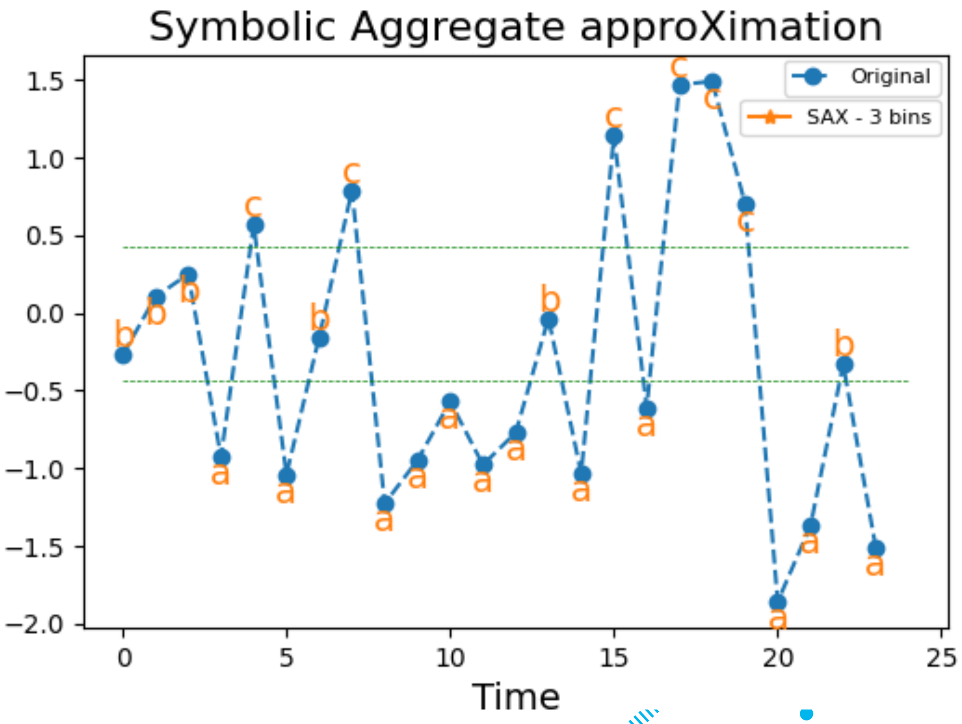
Step 2: Calculate the average value for each segment. These averages represent the PAA-transformed time series.

The user defines the **segment size**. A larger number of segments captures more detail but reduces less dimensionality.

Symbolic Aggregate Approximation (SAX)

Symbolic Aggregate Approximation (SAX) is a technique used to further compress the data obtained from PAA.

It involves transforming numerical PAA values into symbols, representing a range of 'bucketed' values.



E.g. < -.5 = '**a**'; ≥ -0.5 & <0.5 = '**b**'; ≥0.5 = '**c**'

Symbolic Aggregate Approximation (SAX)

SAX reduces the dimensionality of the data even further while preserving essential patterns and trends.

By doing so, it offers:

- **Dimensionality Reduction** – by enabling the representation of long time series with a compact symbol sequence.
- **Pattern Recognition** – by having symbols capture important patterns and trends in the data.
- **Interpretability** – by allowing for easy interpretation and comparison of time series through Symbolic representation.

Symbolic Aggregate Approximation (SAX)

To transition from PAA to SAX:

Step 1: Divide the range of possible values into equal-sized intervals based on the standard deviation of the data.

Step 2: Assign a symbol to each interval. The number of intervals corresponds to the size of the alphabet.

Step 3: For each numerical value obtained from PAA, find the interval it falls into.

Step 4: Replace the numerical value with the corresponding symbol.

PAA and SAX – Example

Consider the following time series data of goals scored by Diego Maradona and Pele playing for their respective national teams.

The first column shows the years.

The second column shows the corresponding goals for Maradona in a normalized (scaled) form.

The third column shows Pele's goals in the normalized form.

Year	Maradona Goals Normalized	Pele Goals Normalized
1	-0.70	-0.80
2	-0.70	1.00
3	-0.10	1.60
4	0.70	-0.20
5	0.50	-1.40
6	0.30	0.80
7	0.70	0.60
8	0.70	-0.80
9	0.50	1.00
10	2.70	0.00
11	0.20	-1.40
12	-0.20	0.80

PAA and SAX – Example

a) Compute a **3-segment** PAA representations of the goals time series data for each player.

We know that we want 3 segments. Therefore, each segment will have 4 years (12 / 3 = 4).

Year	Maradona Goals Normalized	Pele Goals Normalized
1	-0.70	-0.80
2	-0.70	1.00
3	-0.10	1.60
4	0.70	-0.20
5	0.50	-1.40
6	0.30	0.80
7	0.70	0.60
8	0.70	-0.80
9	0.50	1.00
10	2.70	0.00
11	0.20	-1.40
12	-0.20	0.80

Maradona goals:

Segment 1 (Years 1—4): {-0.70, -0.70, -0.10, 0.70} Mean = -0.80 / 4 = -0.20

Segment 2 (Years 5—8): {0.50, 0.30, 0.70, 0.70} Mean = 2.20 / 4 = 0.55

Segment 3 (Years 9—12): {0.50, 2.70, 0.20, -0.20} Mean = 3.20 / 4 = 0.80

Maradona PAA: {-0.20, 0.55, 0.80}

PAA and SAX – Example

Year	Maradona Goals Normalized	Pele Goals Normalized
1	-0.70	-0.80
2	-0.70	1.00
3	-0.10	1.60
4	0.70	-0.20
5	0.50	-1.40
6	0.30	0.80
7	0.70	0.60
8	0.70	-0.80
9	0.50	1.00
10	2.70	0.00
11	0.20	-1.40
12	-0.20	0.80

a) Compute a **3-segment** PAA representations of the goals time series data for each player.

We know that we want 3 segments. Therefore, each segment will have 4 years (12 / 3 = 4).

Pele goals:

Segment 1 (Years 1—4): {-0.80, 1.00, 1.60, -0.20} Mean = 1.60 / 4 = 0.40

Segment 2 (Years 5—8): {-1.40, 0.80, 0.60, -0.80} Mean = -0.80 / 4 = -0.20

Segment 3 (Years 9—12): {1.00, 0.00, -1.40, 0.80} Mean = .40 / 4 = 0.10

Pele PAA: {0.40, -0.20, 0.10}

PAA and SAX – Example

b) For the two PAAs from (a), compute the SAX representations using the following break point data.

Alphabet	Breakpoint 1	Breakpoint 2
a	Negative Infinity	< -0.67
b	>= -0.67	< 0
c	>= 0	< 0.67
d	>= 0.67	Positive Infinity

PAA and SAX – Example

b) For the two PAAs from (a), compute the SAX representations using the following break point data.

Recall our previous PAA values. Simply map these to the corresponding SAX values.

Maradona PAA: { -0.20, 0.55, 0.80 }

Maradona SAX: { b c d }

Pele PAA: { 0.40, -0.20, 0.10 }

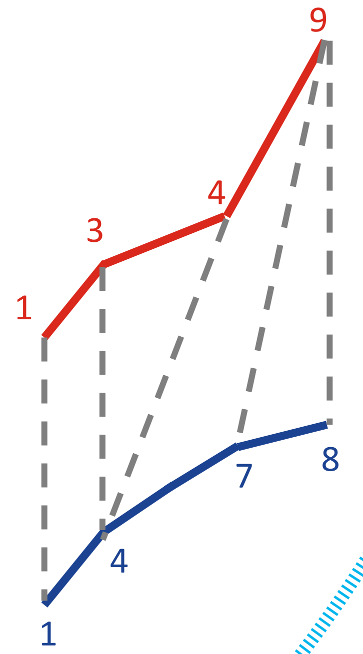
Pele SAX: { c b c }

Alphabet	Breakpoint 1	Breakpoint 2
a	Negative Infinity	< -0.67
b	>= -0.67	< 0
c	>= 0	< 0.67
d	>= 0.67	Positive Infinity

Dynamic Time Warping

Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW) is a method used to measure the similarity between two time series, even when they have different lengths or are warped in time.



Unlike traditional distance measures, like Euclidean distance or Pearson correlation, DTW accounts for the local temporal distortions between time series.

Dynamic Time Warping (DTW)

DTW is effective in comparing time series with variations in speed, phase shifts, or non-linear distortions.

It is widely used in speech recognition, gesture recognition, pattern matching, and time series clustering.

Dynamic Time Warping (DTW)

DTW has the following steps:

Step 1: We generate a 'distance matrix', by calculating the squared difference between each pair of elements across both time series.

Step 2: We generate an 'accumulated distance matrix', using the distance matrix from the previous step, using dynamic programming.

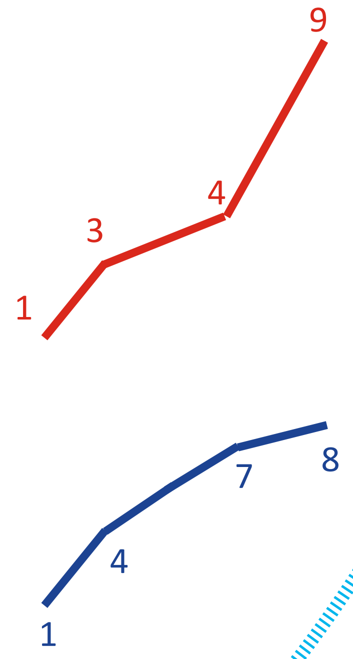
Step 3: We identify the optimal 'path' through the accumulated distance matrix, selecting the path with the minimum accumulated cost (distance).

Step 4: We use that path to determine which elements in which time series should be mapped to each other.

DTW – Example

Consider the following two time series:

$X = \{1, 3, 4, 9\}$; $Y = \{1, 4, 7, 8\}$



How similar are these time series sequences?

How well do the points map together, and how do we find the <u>best</u> mapping between the elements of each time series?

DTW – Step 1

Consider the following two time series:

X = {1, 3, 4, 9};   Y = {1, 4, 7, 8}

Step 1: We generate the **distance matrix**, using the following formula:

$$\text{dist}(i, j) = (X_i - Y_j)^2$$

9				
4				
3				
1				
	1	**4**	**7**	**8**

DTW – Step 1

Consider the following two time series:

$X = \{1, 3, 4, 9\}$; $Y = \{1, 4, 7, 8\}$

Step 1: We generate the **distance matrix**, using the following formula:

$$\mathrm{dist}(i, j) = (X_i - Y_j)^2$$

$\mathrm{Dist}(1, 1) = (1 - 1)^2 = 0^2 = 0$

9				
4				
3				
1	0			
	1	**4**	**7**	**8**

DTW – Step 1

Consider the following two time series:

X = {1, 3, 4, 9};   Y = {1, 4, 7, 8}

Step 1: We generate the **distance matrix**, using the following formula:

$$dist(i, j) = (X_i - Y_j)^2$$

$Dist(1, 4) = (1 - 4)^2 = -3^2 = 9$

9				
4				
3				
1	0	9		
	1	**4**	**7**	**8**

DTW – Step 1

Consider the following two time series:

X = {1, 3, 4, 9}; Y = {1, 4, 7, 8}

Step 1: We generate the **distance matrix**, using the following formula:

$$\text{dist}(i, j) = (X_i - Y_j)^2$$

$\text{Dist}(1, 7) = (1 - 7)^2 = -6^2 = 36$

9				
4				
3				
1	0	9	36	
	1	**4**	**7**	**8**

DTW – Step 1

Consider the following two time series:

X = {1, 3, 4, 9}; Y = {1, 4, 7, 8}

Step 1: We generate the **distance matrix**, using the following formula:

$$\text{dist}(i, j) = (X_i - Y_j)^2$$

Dist(1, 8) = $(1 - 8)^2$ = -7^2 = 49

9				
4				
3				
1	0	9	36	49
	1	**4**	**7**	**8**

DTW – Step 1

Consider the following two time series:

X = {1, 3, 4, 9}; Y = {1, 4, 7, 8}

Step 1: We generate the **distance matrix** using the following formula:

$$\text{dist}(i, j) = (X_i - Y_j)^2$$

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

Consider the following two time series:

X = {1, 3, 4, 9};   Y = {1, 4, 7, 8}

Step 2: We generate the **accumulated distance matrix**

9	∞				
4	∞				
3	∞				
1	∞				
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Note: To start with, our accumulated distance matrix is just an empty version of our distance matrix, but with one extra column and one extra row.

These cells are all populated with infinity, except the bottom left cell, which is populated with 0. We'll explain the intuition behind this at the end.

9	∞				
4	∞				
3	∞				
1	∞				
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

Consider the following two time series:

X = {1, 3, 4, 9}; Y = {1, 4, 7, 8}

Step 2: We generate the **accumulated distance matrix** using the following formula:

$$AD(x_i, y_j) = \text{dist}(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

9	∞				
4	∞				
3	∞				
1	∞				
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Note: In other words, we populate each accumulated distance matrix cell with:
The distance matrix value for each cell **plus** whichever neighbouring cell, between:

$$AD[\,x_{i-1},\ y_j\,],\quad AD[\,x_i,\ y_{j-1}\,],\quad AD[\,x_{i-1},\ y_{j-1}\,]$$

has the **smaller** value.

$$AD(\,x_i, y_j\,) = \text{dist}(\,x_i, y_j\,) + \boldsymbol{min} \begin{cases} AD[\,x_{i-1},\ y_j\,] \\ AD[\,x_i,\ y_{j-1}\,] \\ AD[\,x_{i-1},\ y_{j-1}\,] \end{cases}$$

9	∞				
4	∞				
3	∞				
1	∞				
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	4	7	8

Note: In other words, we populate each accumulated distance matrix cell with: The distance matrix value for each cell **plus** whichever neighbouring cell, between:

$AD[x_{i-1}, y_j]$, $AD[x_i, y_{j-1}]$, $AD[x_{i-1}, y_{j-1}]$

has the **smaller** value.

$$AD(x_i, y_j) = dist(x_i, y_j) + \underline{\boldsymbol{min}} \begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

9	∞				
4	∞				
3	∞				
1	∞				
	0	∞	∞	∞	∞
		1	4	7	8

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	$\boxed{0}$	9	36	49
	1	**4**	**7**	**8**

Distance matrix

$$AD(x_i, y_j) = \mathrm{dist}(x_i, y_j) + \boldsymbol{min}\begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ \boxed{AD[x_{i-1}, y_{j-1}]} \end{cases}$$

To populate our first cell:

AD(1, 1) = 0 + 0 = 0

9	∞				
4	∞				
3	∞				
1	∞	$\boxed{}$			
	$\boxed{0}$	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

$$AD(x_i, y_j) = \text{dist}(x_i, y_j) + \textbf{min} \begin{cases} AD[x_{i-1}, y_j] \\ \boxed{AD[x_i, y_{j-1}]} \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

To populate the next cell:

AD(1, 4) = 9 + 0 = 9

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

9	∞				
4	∞				
3	∞				
1	∞	0			
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	$\boxed{36}$	49
	1	**4**	**7**	**8**

Distance matrix

$$AD(x_i, y_j) = \text{dist}(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ \boxed{AD[x_i, y_{j-1}]} \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

To populate the next cell:

AD(1, 7) = 36 + 9 = 45

9	∞				
4	∞				
3	∞				
1	∞	0	$\boxed{9}$	$\boxed{}$	
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

$$AD(x_i, y_j) = \text{dist}(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

To populate the next cell:

AD(1, 8) = 49 + 45 = 94

9	∞				
4	∞				
3	∞				
1	∞	0	9	45	
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

$$AD(x_i, y_j) = \mathrm{dist}(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

To populate the next cell:

AD(3, 1) = 4 + 0 = 4

9	∞				
4	∞				
3	∞				
1	∞	0	9	45	94
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	$\boxed{1}$	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

$$AD(x_i, y_j) = \text{dist}(x_i, y_j) + \boldsymbol{min}\begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ \boxed{AD[x_{i-1}, y_{j-1}]} \end{cases}$$

To populate the next cell:

AD(3, 4) = 1 + 0 = 1

9	∞				
4	∞				
3	∞	4	$\boxed{}$		
1	∞	$\boxed{0}$	9	45	94
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	4	7	8

Distance matrix

$$AD(x_i, y_j) = \text{dist}(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

To populate the next cell:

AD(3, 7) = 16 + 1 = 17

9	∞				
4	∞				
3	∞	4	1		
1	∞	0	9	45	94
	0	∞	∞	∞	∞
		1	4	7	8

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

$$AD(x_i, y_j) = dist(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ \boxed{AD[x_i, y_{j-1}]} \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

To populate the next cell:

AD(3, 8) = 25 + 17 = 42

9	∞				
4	∞				
3	∞	4	1	17	
1	∞	0	9	45	94
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

ABERDEEN 2040

DTW – Step 2

9	64	25	4	1
4	9	0	9	16
3	4	1	16	25
1	0	9	36	49
	1	**4**	**7**	**8**

Distance matrix

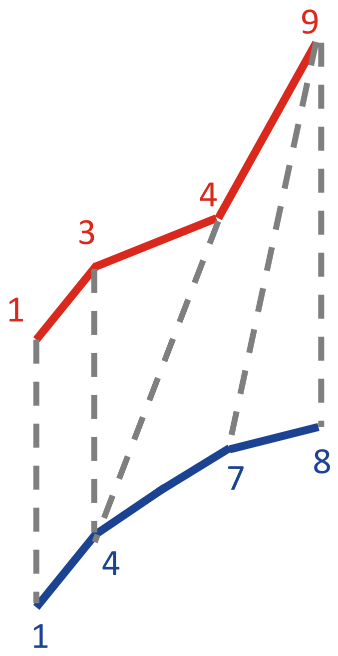
$$AD(x_i, y_j) = \mathrm{dist}(x_i, y_j) + \boldsymbol{min} \begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

We repeat this process, populating the whole accumulated distance matrix.

9	∞	77	26	5	6
4	∞	13	1	10	26
3	∞	4	1	17	42
1	∞	0	9	45	94
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Step 3 & 4

Step 3: We identify the optimal 'path' through the accumulated distance matrix, selecting the path with the minimum accumulated cost (from bottom left to top right).



9	∞	77	26	5	6
4	∞	13	1	10	26
3	∞	4	1	17	42
1	∞	0	9	45	94
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**

DTW – Intuition

$$AD(x_i, y_j) = \mathrm{dist}(x_i, y_j) + \boldsymbol{min}\begin{cases} AD[x_{i-1}, y_j] \\ AD[x_i, y_{j-1}] \\ AD[x_{i-1}, y_{j-1}] \end{cases}$$

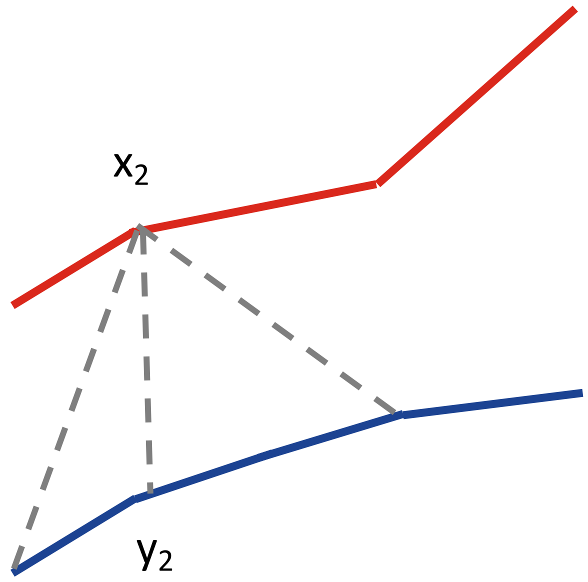
If you think about it, this formula is actually doing something fairly simple…

For a given point (e.g. x_2)

should it be connected to y_2?

should it be connected to y_1?

should it be connected to y_3?



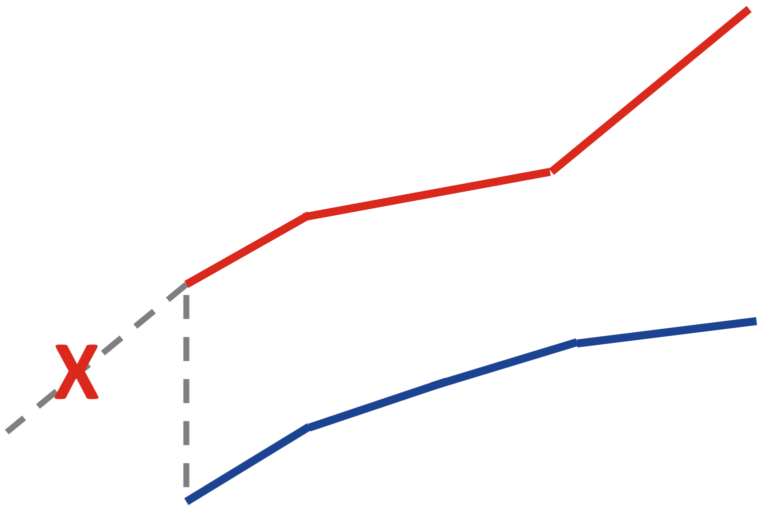
DTW – Intuition

And pre-filling our matrix with the 0 and row and column of ∞ values is just a weighting that makes sure that:

- We always start by joining X1 to Y1
- And we never go out of bounds

Accumulated distance matrix *D*

9	∞				
4	∞				
3	∞				
1	∞				
	0	∞	∞	∞	∞
		1	**4**	**7**	**8**



DTW – Advantages

A major advantage of DTW is its **flexibility**, in that it accommodates variations in time series alignment, making it suitable for analyzing data with temporal variations.

Another is its **robustness**, since DTW can handle noisy data and missing values more effectively than traditional distance measures.

DTW – Disadvantages

However, DTW suffers from **computational complexity**; it can be computationally intensive, especially for long time series or large datasets.

Furthermore, it **may require careful tuning**, since the choice of DTW parameters, such as the *warping window size*, can affect the results.

Summary of Methods

Technique	Applications
PAA	- Dimensionality reduction - Time series averaging - Data pre-processing
SAX	- Pattern recognition - Symbolic sequence analysis - Anomaly detection
DTW	- Time series alignment - Similarity measurement - Speech recognition