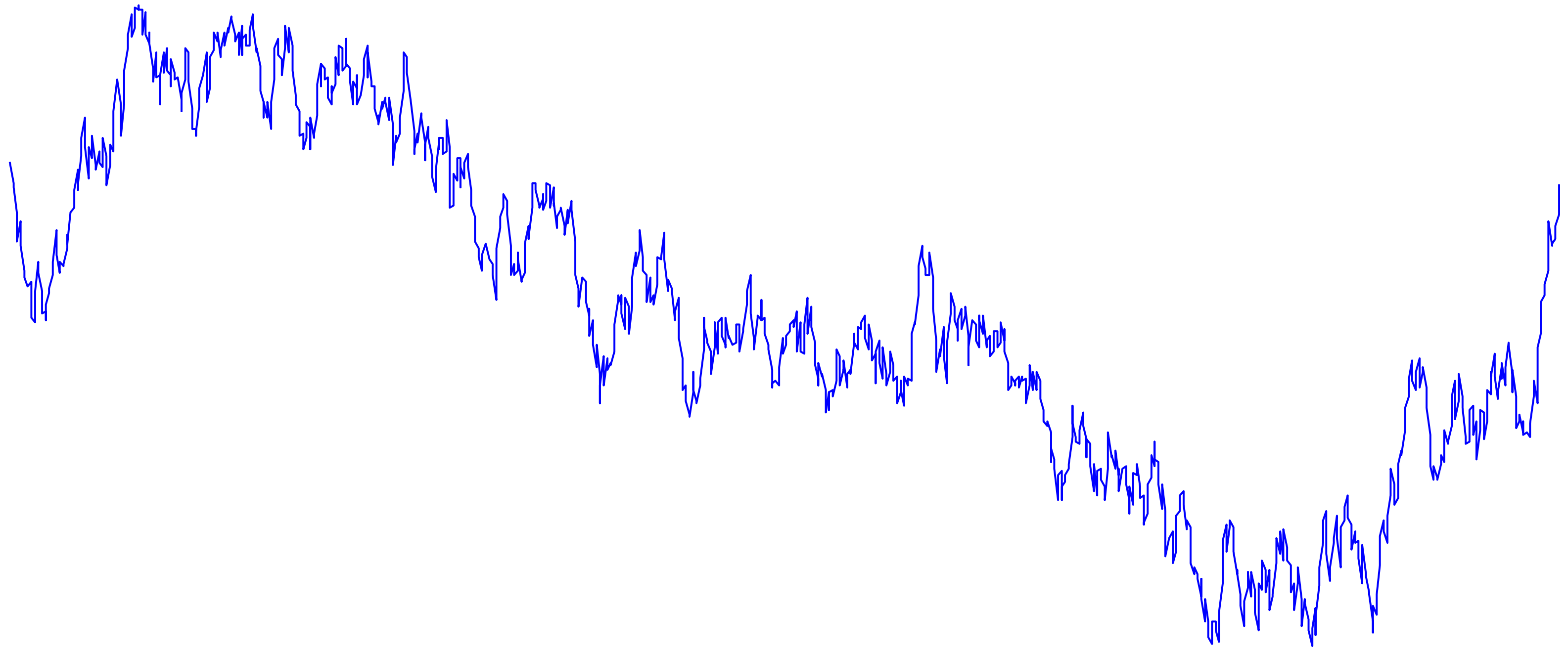
Time Series - Similarity, Distances, Transformations and Clustering

UNIVERSITÀ DI PISA

What is a Time Series?

- A time series is a collection of observations made sequentially in time, generally at constant time intervals.

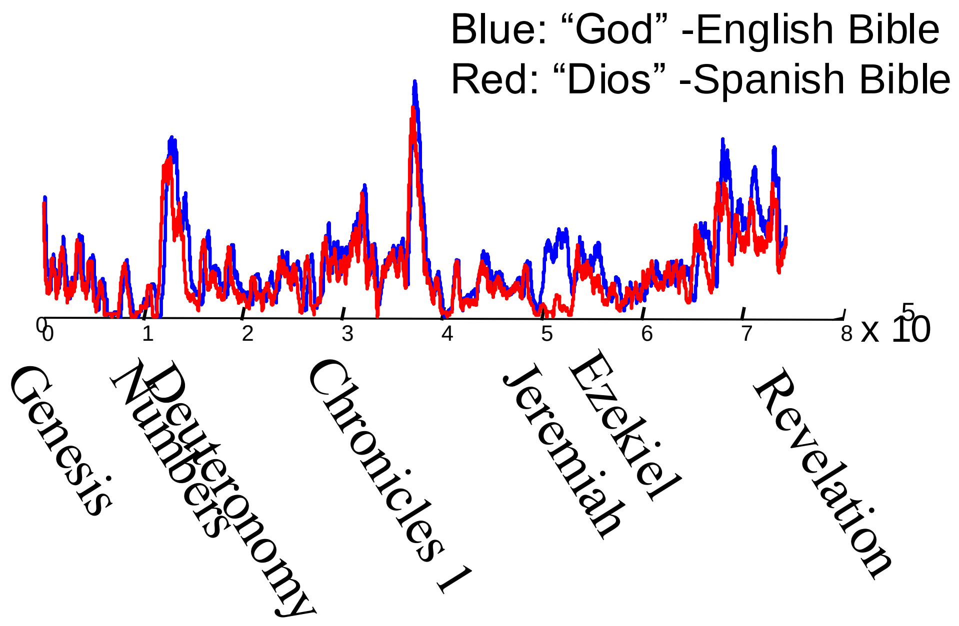


25.1750
25.2250
25.2500
25.2500
25.2750
25.3250
25.3500
25.3500
25.4000
25.4000
25.3250
25.2250
25.2000
25.1750
…
24.6250
24.6750
24.6750
24.6250
24.6250
24.6250
24.6750
24.7500

Time Series are Ubiquitous

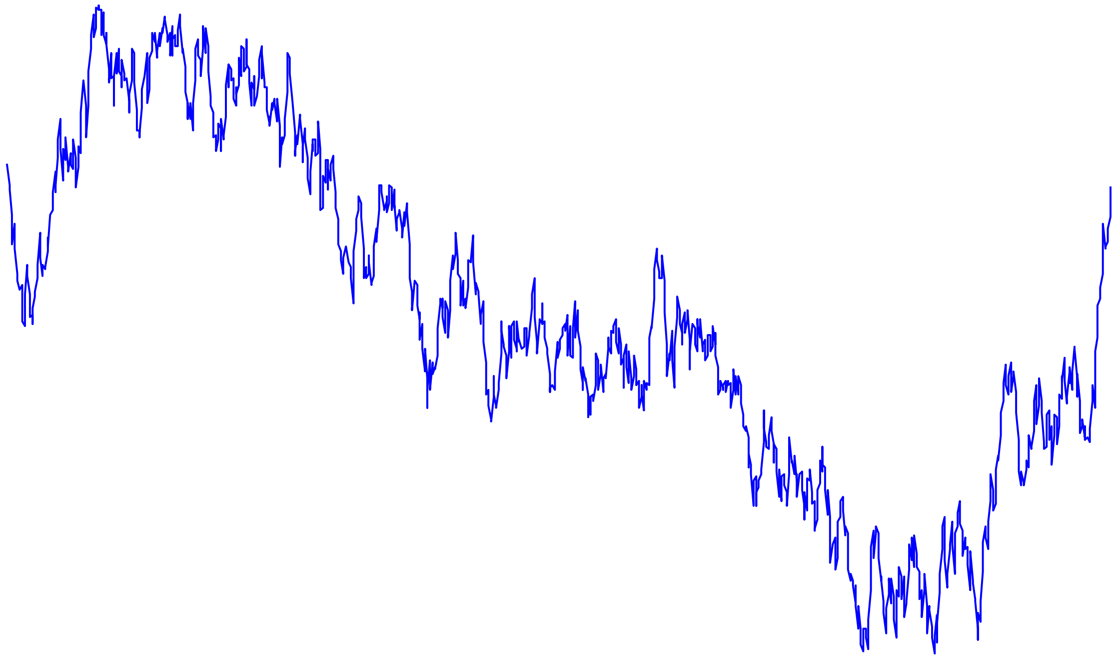
- You can measure many things … and things change over time.
  - Blood pressure
  - Donald Trump's popularity rating
  - The annual rainfall in Pisa
  - The value of your stocks

- In addition, other data type can be seen as time series
  - Text data: words count
  - Images: edges displacement
  - Videos: object positioning



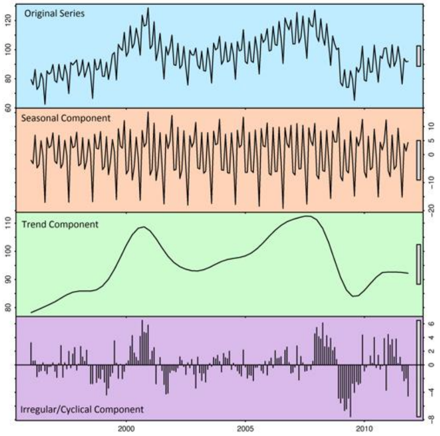
Problems in Working with Time Series

- Large amount of data.
- Similarity is not easy to estimate.
- Differing data formats.
- Differing sampling rates.
- Noise, missing values, etc.

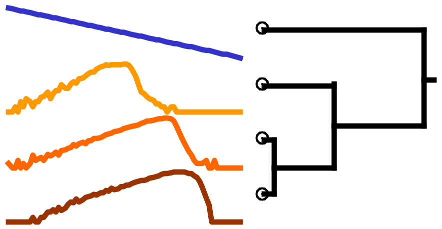


What We Can Do With Time Series?

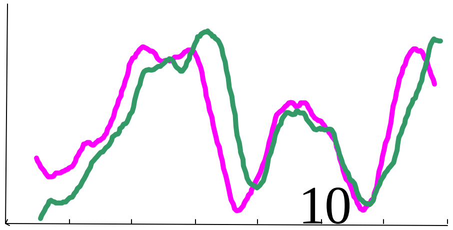
- Trends, Seasonality



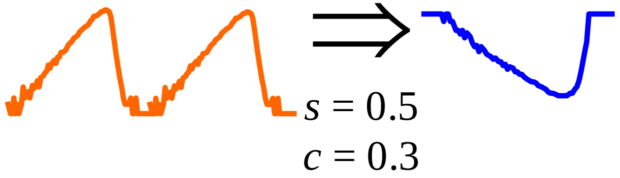
- Clustering



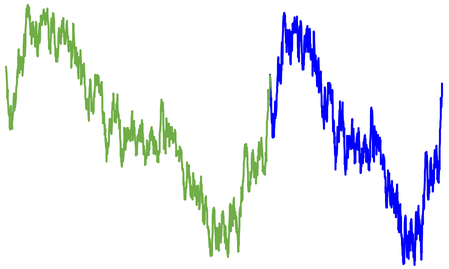
- Motif Discovery



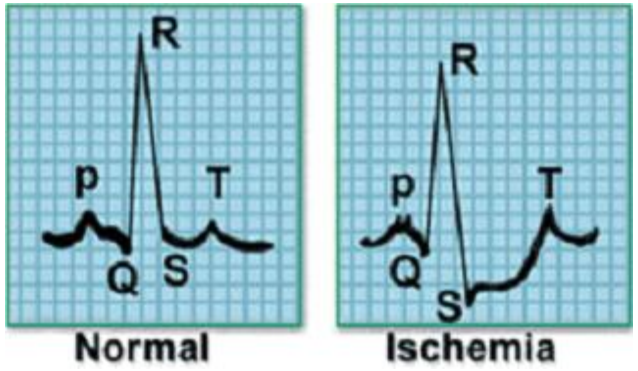
- Rule Discovery



- Forecasting

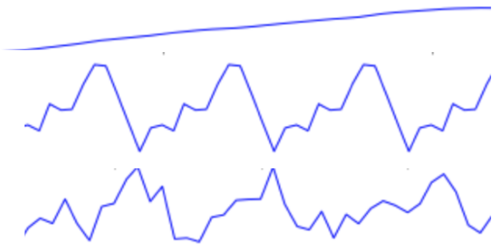


- Classification



Time Series Components

- A given TS consists of **three systematic** components including level, trend, seasonality, and **one non-systematic** component called noise.
  - **Level**: The average value in the series.
  - **Trend**: The increasing or decreasing value in the series.
  - **Seasonality**: The repeating short-term cycle in the series.
  - **Noise**: The random variation in the series.



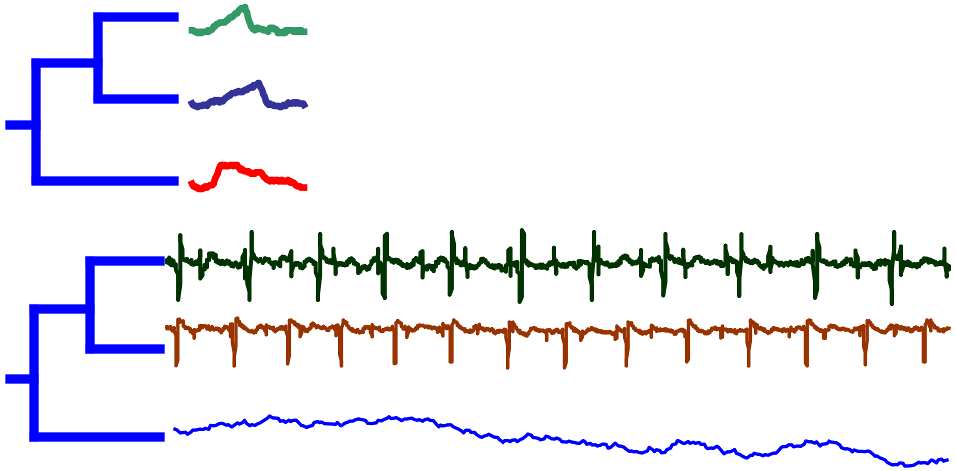
- A **systematic** component have consistency or recurrence and can be described and modeled.
- A **non-systematic** component cannot be directly modeled.

Similarity, Distances and Transformations

Similarity

- All these problems require similarity matching.

- What is Similarity?
  - It is the quality or state of being similar, likeness, resemblance, as a similarity of features.
- In time series analysis we recognize two kinds of similarity:
  - Similarity at the level of shape
  - Similarity at the structural level



Shape-based Similarities

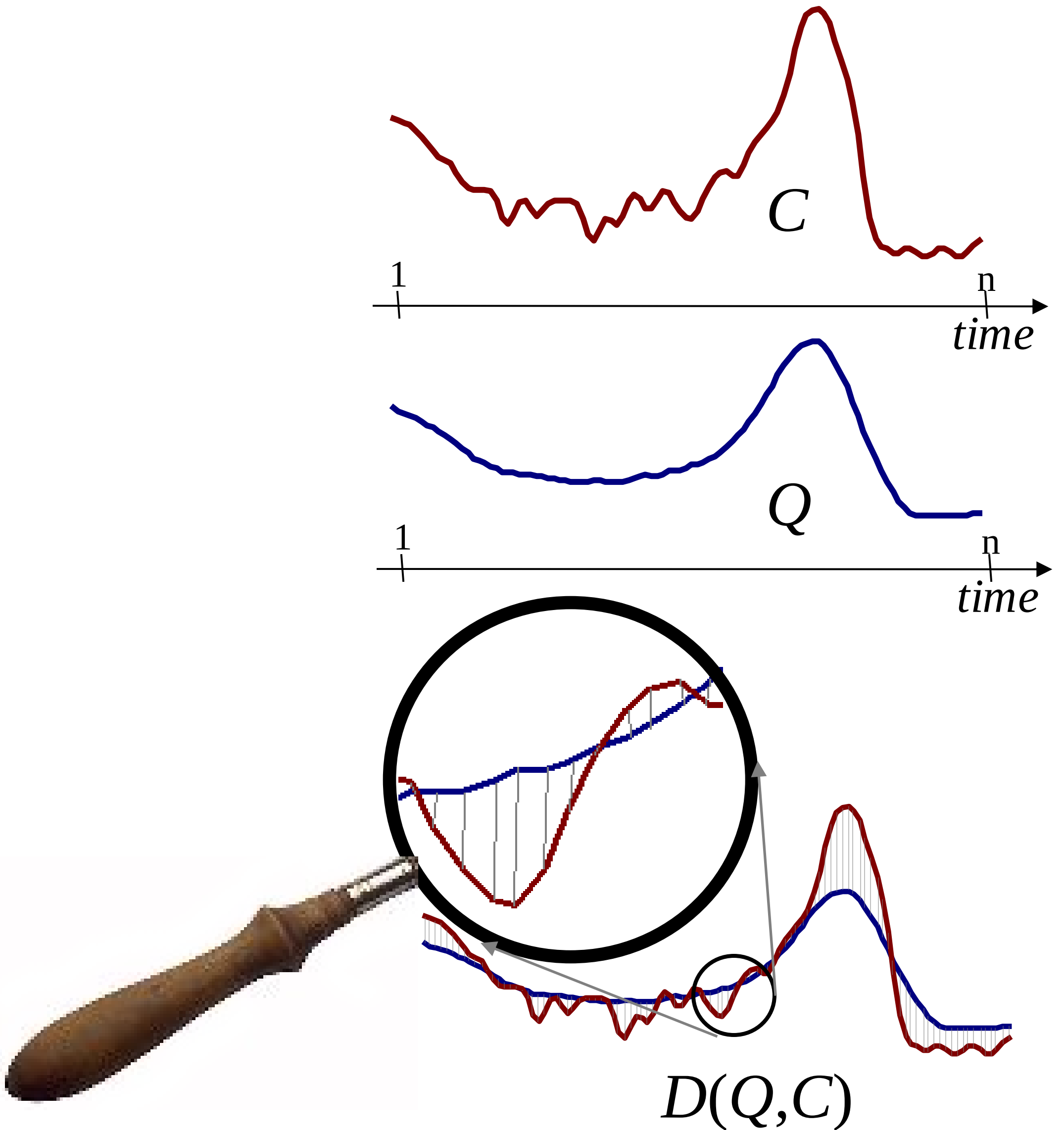
Euclidean Distance

- Given two time series:
  - $Q = q_1 \ldots q_n$
  - $C = c_1 \ldots c_n$

$$D(Q,C) \equiv \sqrt{\sum_{i=1}^{n}(q_i - c_i)^2}$$

- $T1 = < 56, \quad 176, \quad 110, \quad 95 >$
- $T2 = < 36, \quad 126, \quad 180, \quad 80 >$

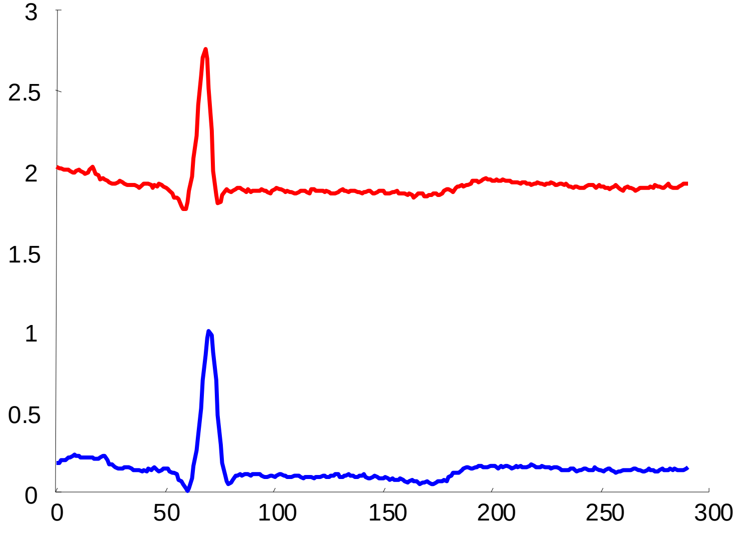
$D(T1,T2) = sqrt\ [\ (56\text{-}36)^2 + (176\text{-}126)^2 + (110\text{-}180)^2 + (95\text{-}80)^2\ ]$

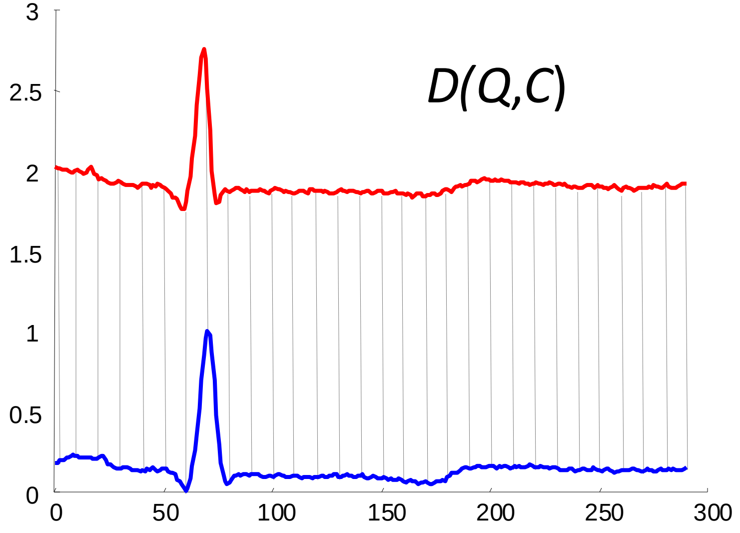


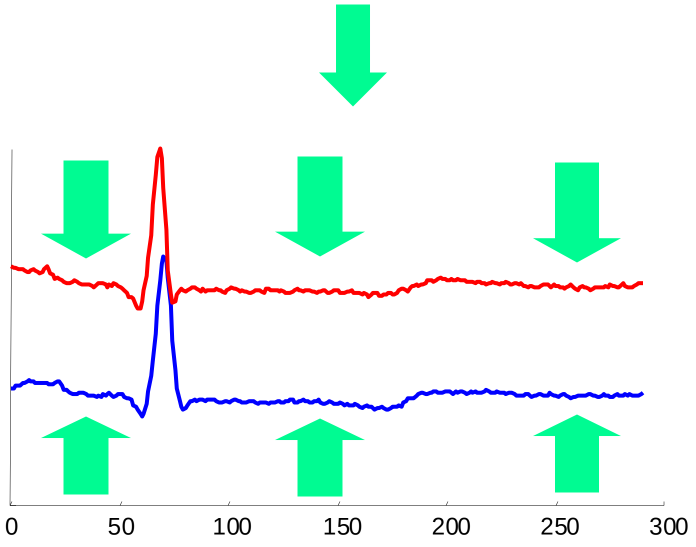
Problems with Euclidean Distance

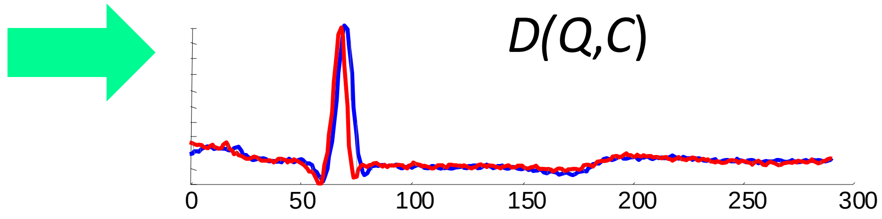
- Euclidean distance is very sensitive to “distortions” in the data.
- These distortions are dangerous and should be removed.
- Most common distortions:
  - Offset Translation
  - Amplitude Scaling
  - Linear Trend
  - Noise
- They can be removed by using the appropriate transformations.

Transformation I: Offset Translation







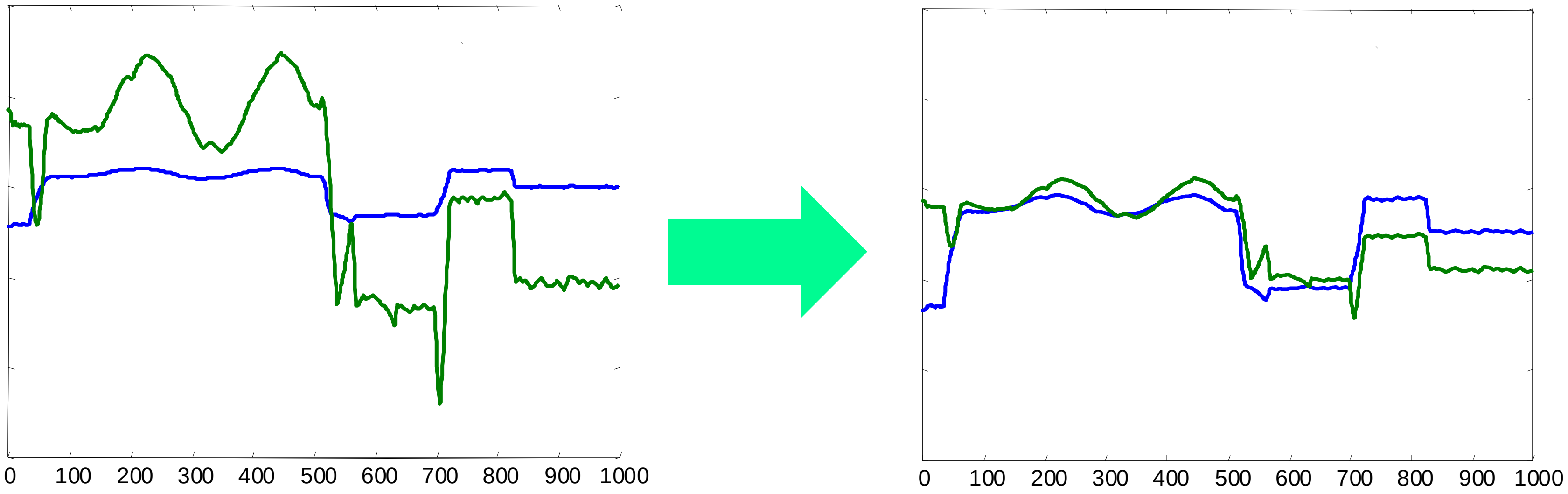


$Q = Q$ - mean(Q)

$C = C$ - mean(C)

$D(Q,C)$

Transformation II: Amplitude Scaling



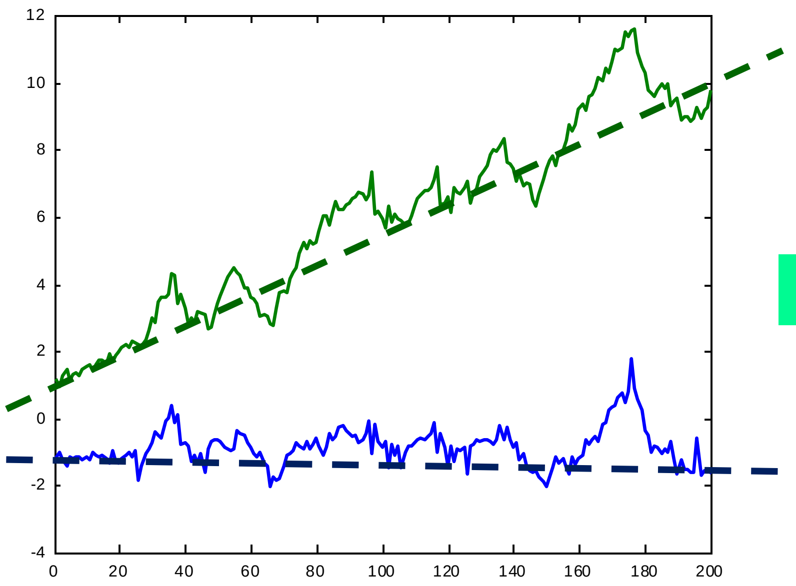
$$Q = (Q - \mathrm{mean}(Q)) \,/\, \mathrm{std}(Q)$$

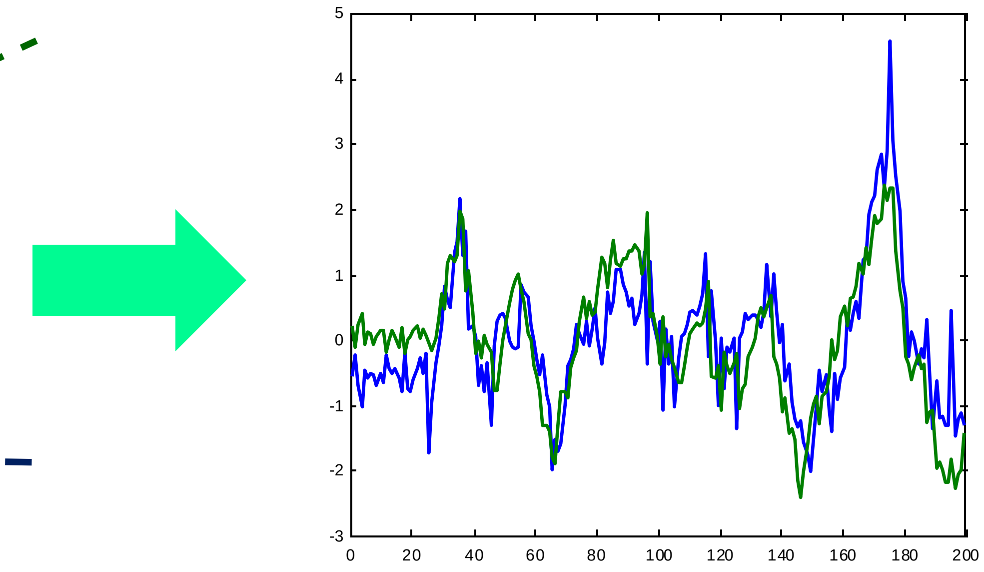
$$C = (C - \mathrm{mean}(C)) \,/\, \mathrm{std}(C)$$

$$D(Q,C)$$

Transformation III: Linear Trend

- Removing linear trend: fit the best fitting straight line to the time series, then subtract that line from the time series.

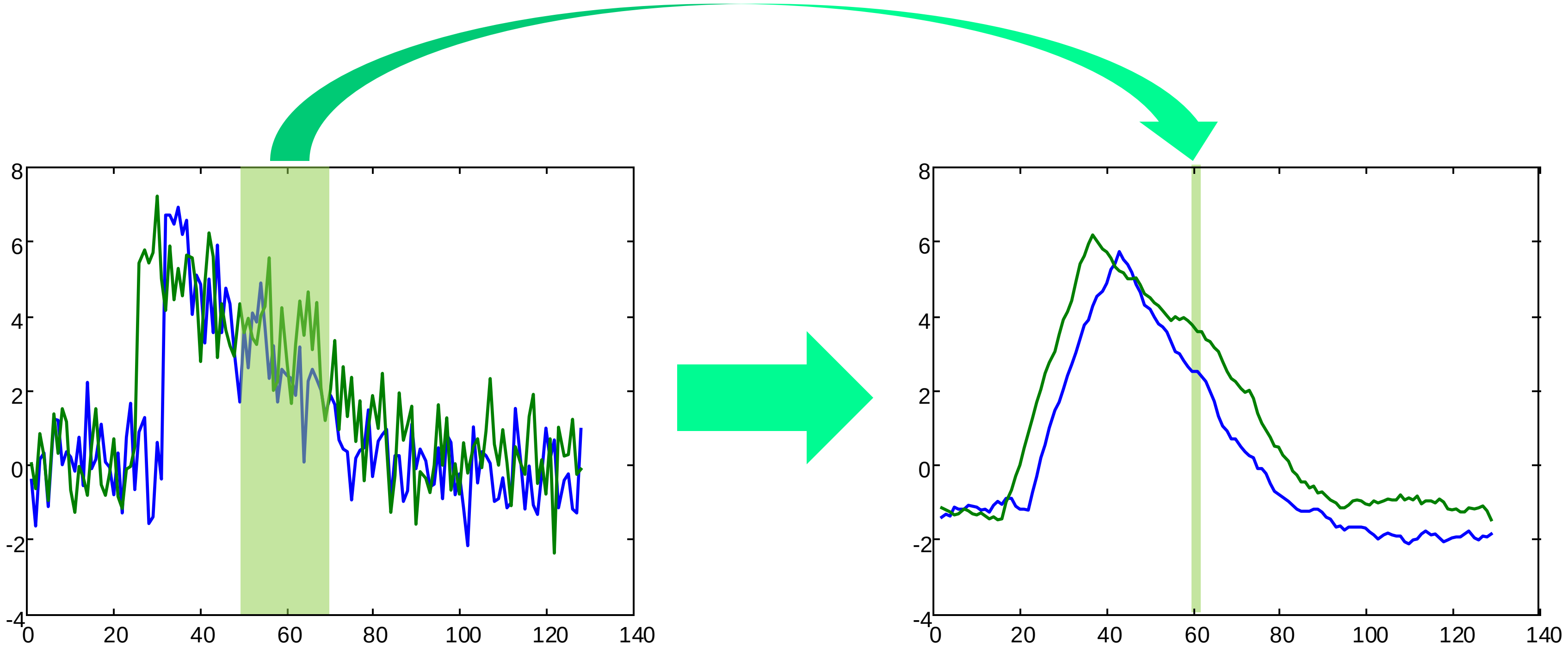




Removed linear trend,
offset translation,
amplitude scaling

Transformation IV: Noise

- The intuition behind removing noise is to average each datapoints value with its neighbors.



Q = smooth(Q)

C = smooth(C)

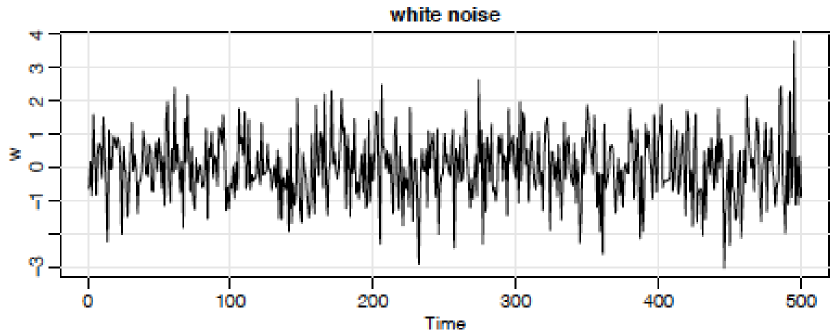
$D(Q,C)$

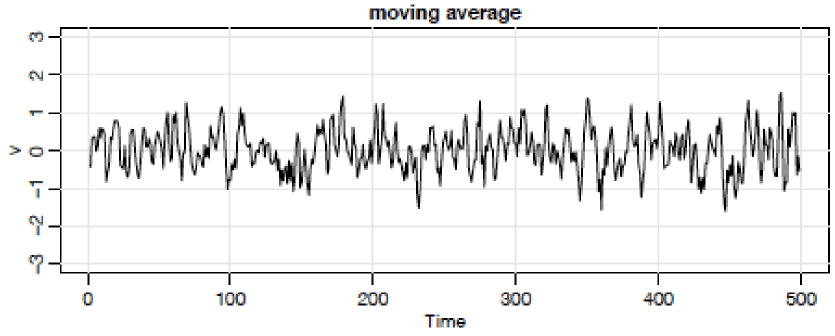
Moving Average

- Noise can be removed by a **moving average** (MA) that smooths the TS.
- Given a window of length *w* and a TS *t*, the MA is applied as follows
- $t_i = \frac{1}{w}\sum_{j=i-w/2}^{w/2} t_j$ for $i = 1, \ldots, n$
- For example, if w=3 we have
- $t_i = \frac{1}{3}\left(t_{i-1} + t_i + t_{i+1}\right)$

w=3

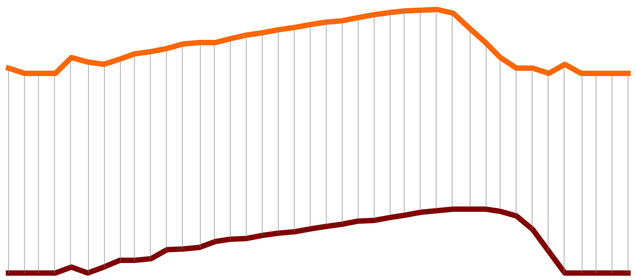
time	value	ma
t1	20	-
t2	24	22.0
t3	22	24.0
t4	26	24.3
t5	25	-



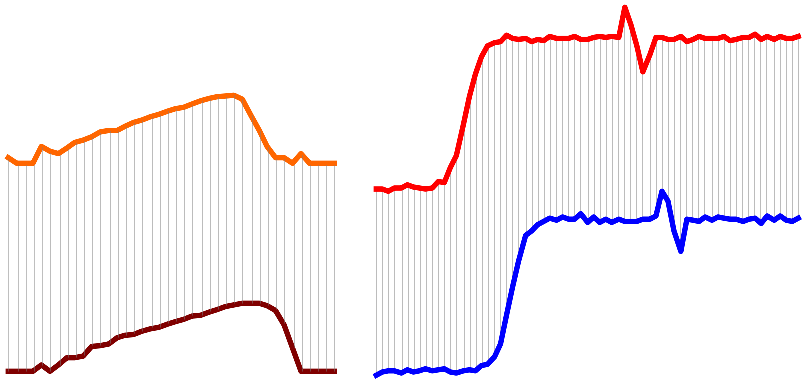


Dynamic Time Warping

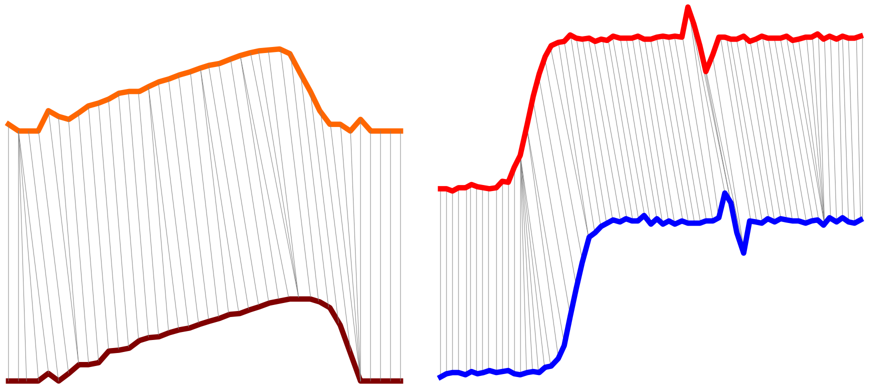
- Sometimes two time series that are conceptually equivalent evolve at different speeds, at least in some moments.



E.g. correspondence of peaks in two similar time series



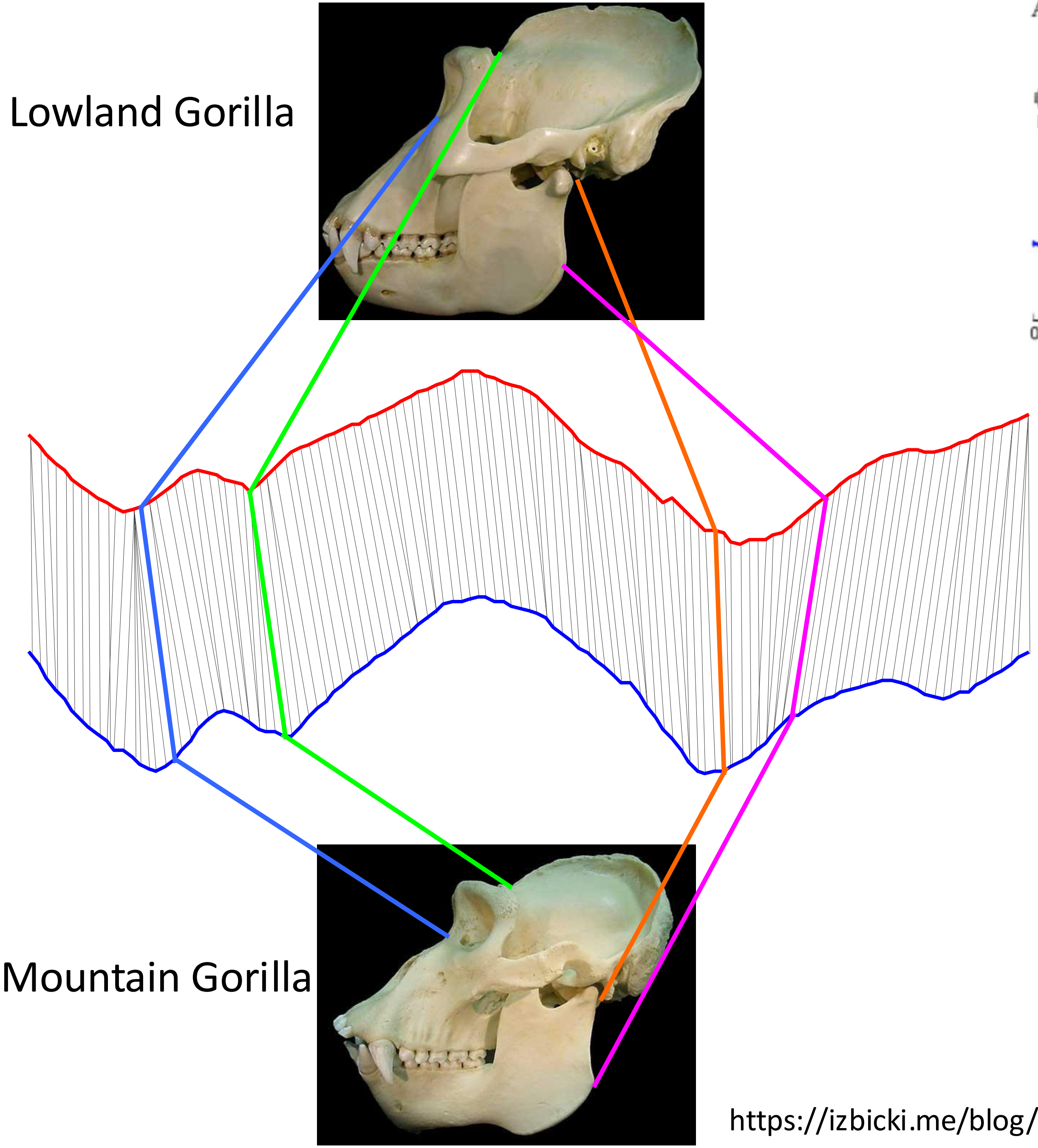
Fixed Time Axis. Sequences are aligned “one to one”. Greatly suffers from the misalignment in data. **Euclidean**.



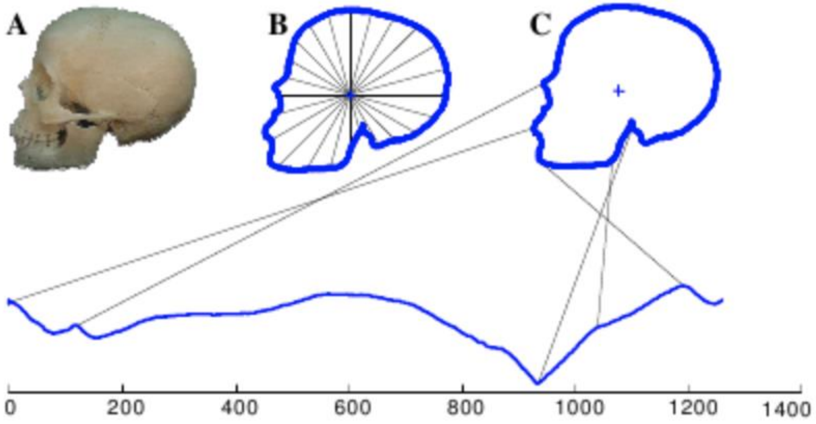
Warped Time Axis. Nonlinear alignments are possible. Can correct misalignments in data. **Dynamic Time Warping.**

IMAGE AS A TIME SERIES

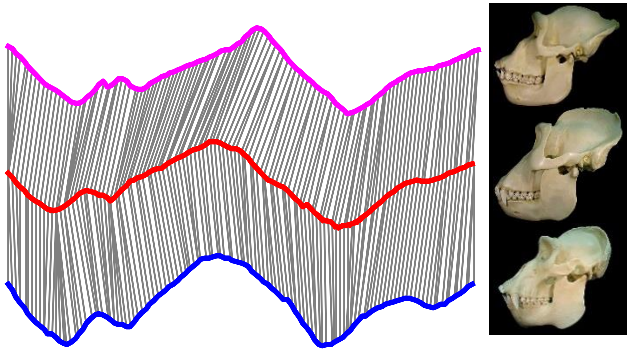
Lowland Gorilla



Mountain Gorilla



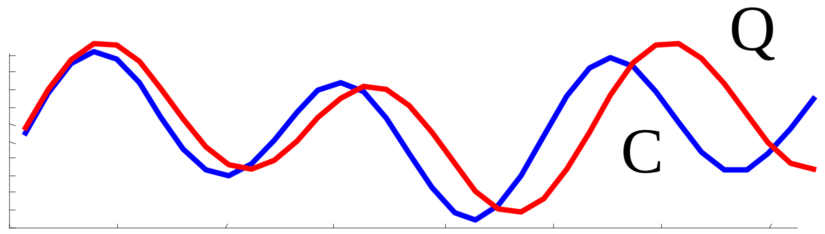
Radial Scanning

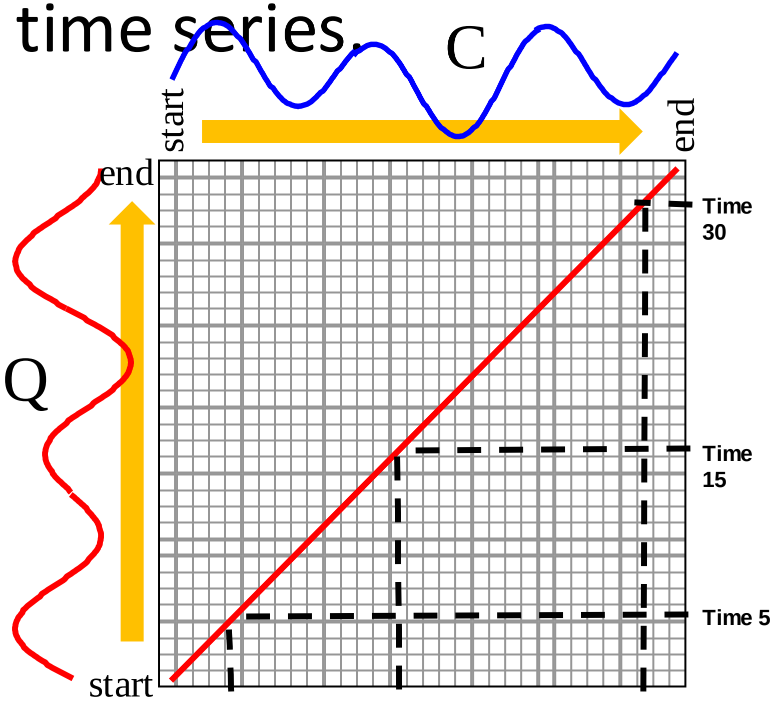


https://izbicki.me/blog/converting-images-into-time-series-for-data-mining.html

How is DTW Calculated?

- We create a matrix with size of |Q| by |C|, then fill it in with the distance between every pair of points in our two time series.



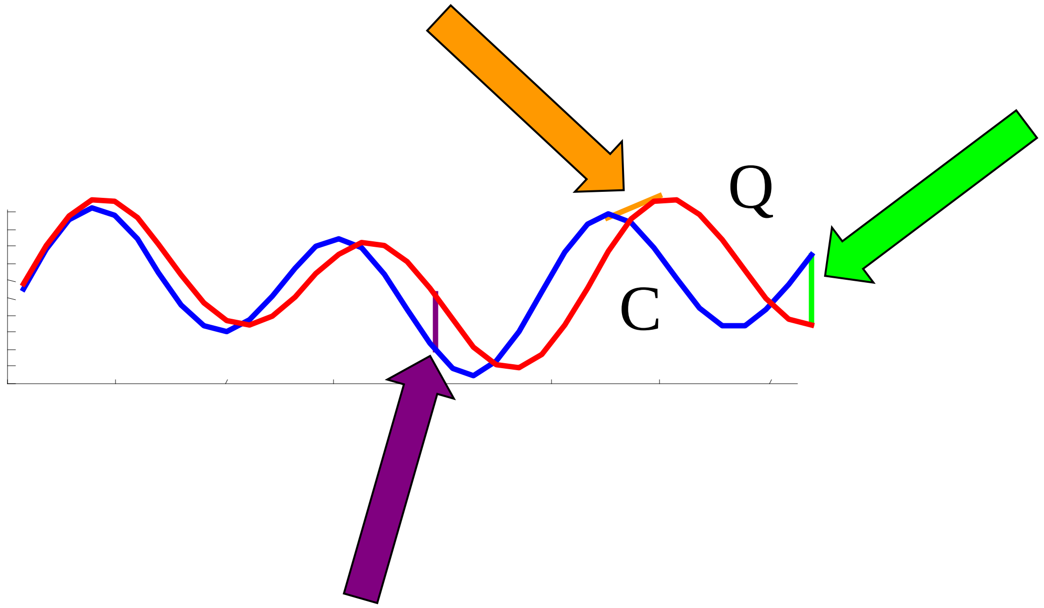


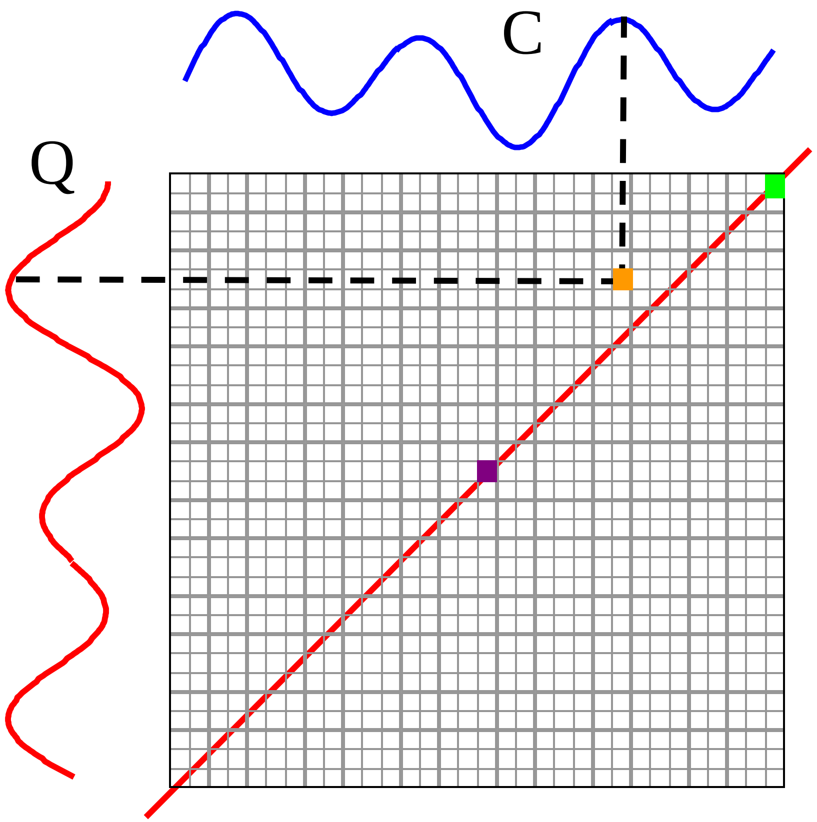
The Euclidean distance works only on the diagonal of the matrix. The sequence of comparisons performed:

- Start from pair of points *(0,0)*
- After point *(i,i)* move to *(i+1,i+1)*
- End the process on *(n,n)*

How is DTW Calculated?

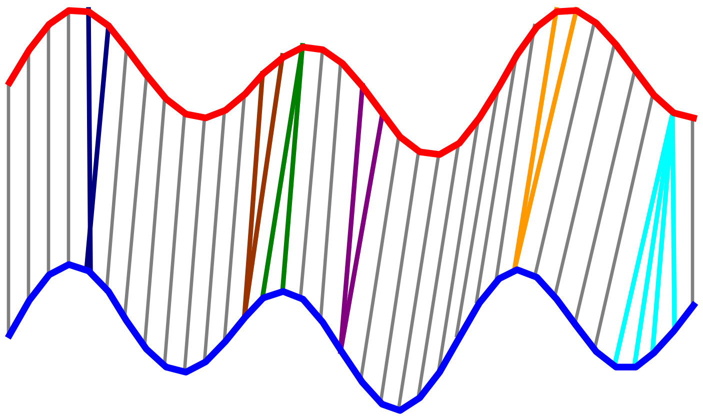
- The DTW distance can “freely” move outside the diagonal of the matrix
- Such cells correspond to temporally shifted points in the two time series

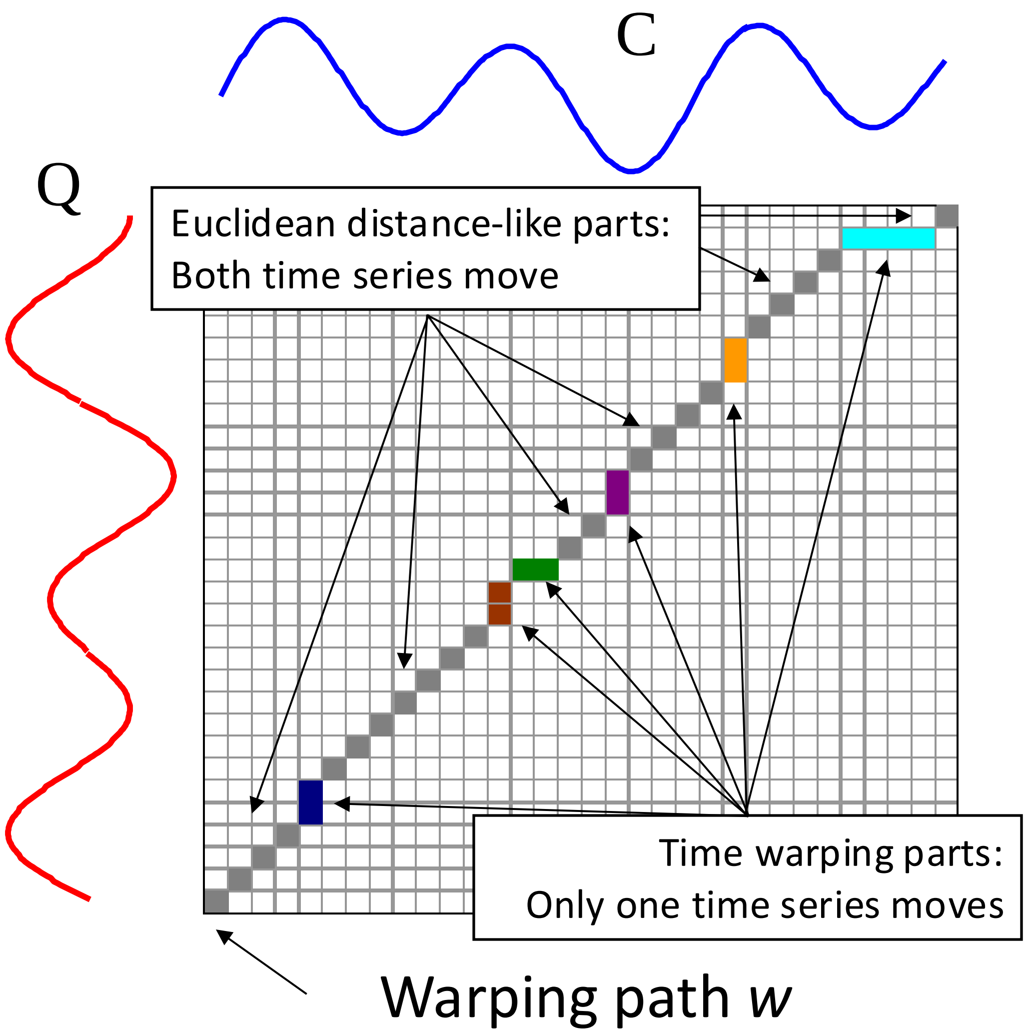




How is DTW Calculated?

- Every possible warping between two time series, is a path through the matrix.
- The constrained sequence of comparisons performed:
  - Start from pair of points *(0,0)*
  - After point *(i,j)*, either *i* or *j* increase by one, or both of them
  - End the process on *(n,n)*





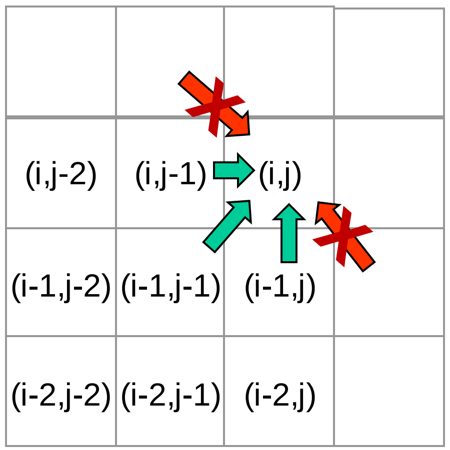
How is DTW Calculated?

- Every possible warping between two time series, is a path through the matrix.
- We find the best one using a recursive definition of the DTW:

$\gamma(i,j)$ = cost of best path reaching cell (i,j)

$= d(q_i,c_j) + \min\{ \gamma(i\text{-}1,j\text{-}1), \gamma(i\text{-}1,j), \gamma(i,j\text{-}1) \}$

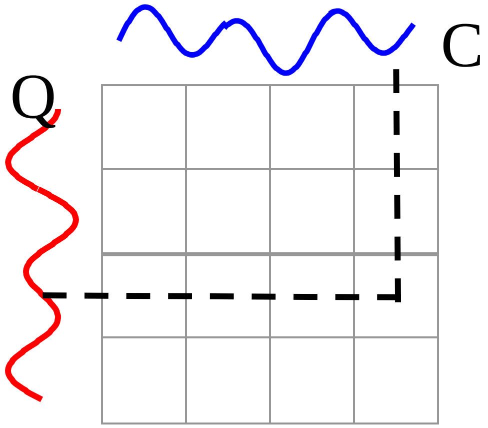
- Idea: best path must pass through (i-1,j), (i-1,j-1) or (i,j-1)



Dynamic Programming Approach

Step 1: compute the matrix of all $d(q_i,c_j)$

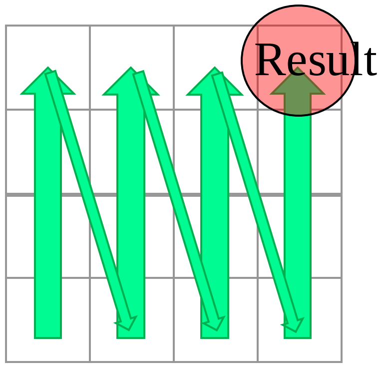
- Point-to-point distances $D(i,j) = | Q_i - C_j |$



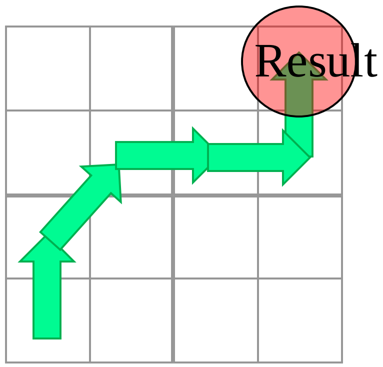
$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \}$$

Step 2: compute the matrix of all path costs $\gamma(i,j)$

- Start from cell *(1,1)*
- Compute *(2,1), (3,1), …, (n,1)*
- Repeat for columns *2, 3, …, n*
- Final result in last cell computed



Step 3: find the path with the lowest value (best alignment)



Dynamic Programming Approach

$\gamma(i,j) \;=\; d(q_i,c_j) + \min\{\, \gamma(i-1,j-1),\, \gamma(i-1,j),\, \gamma(i,j-1) \,\}$

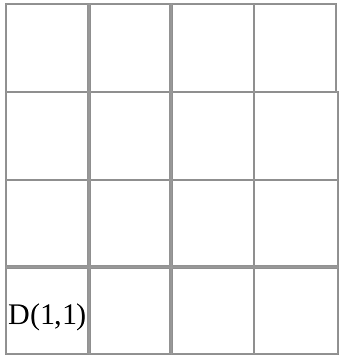
Step 2: compute the matrix of all path costs $\gamma(i,j)$

- Start from cell *(1,1)*

  $\gamma(1,1) = d(q_1,c_1) + \min\{\gamma(0,0), \gamma(0,1), \gamma(1,0)\}$

  $= d(q_1,c_1)$

  $= D(1,1)$

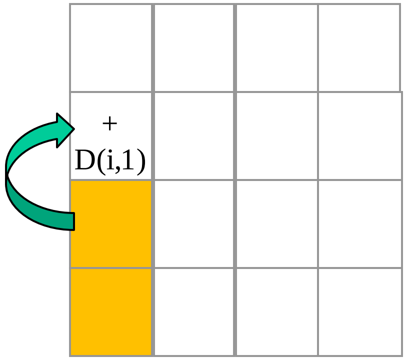


- Compute (2,1), (3,1), …, (n,1)

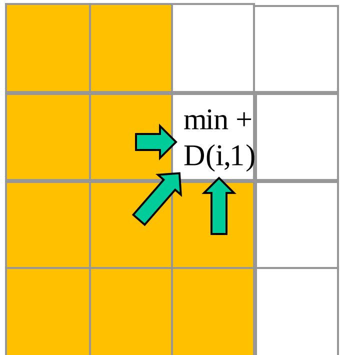
  $\gamma(i,1) = d(q_i,c_1) + \min\{\, \gamma(i-1,0), \gamma(i-1,1), \gamma(i,0) \,\}$

  $= d(q_i,c_1) + \gamma(i-1,1)$

  $= D(i,1) + \gamma(i-1,1)$



- Repeat for columns *2, 3, …, n*
  - The general formula applies

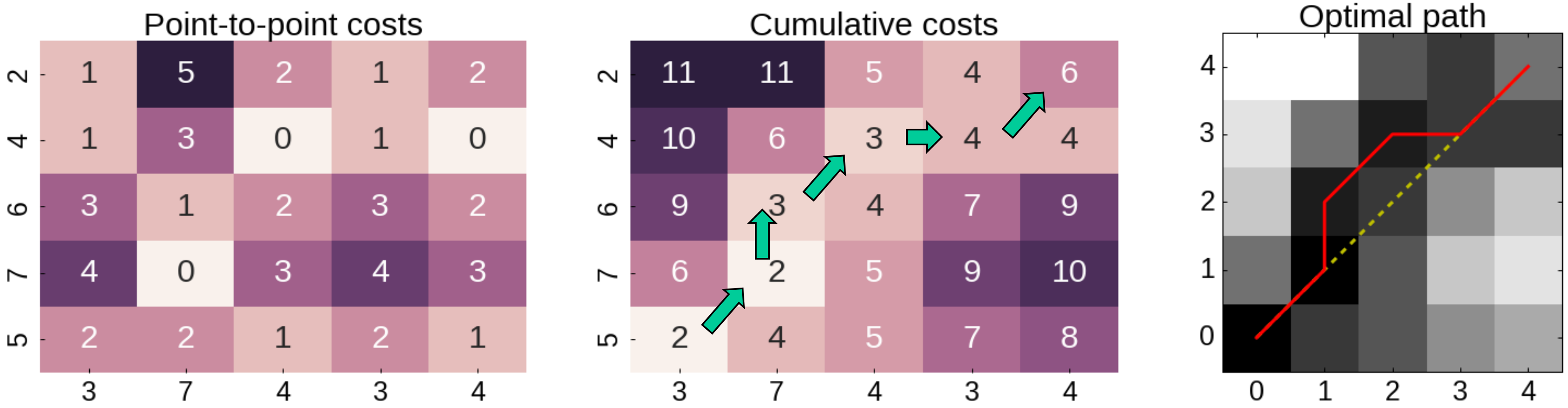


Dynamic Programming Approach

Example

- c = < 3, 7, 4, 3, 4 >
- q = < 5, 7, 6, 4, 2 >

$$\gamma(i,j) \; = \; d(q_i,c_j) + \min\{\, \gamma(i\text{-}1,j\text{-}1),\, \gamma(i\text{-}1,j\,),\, \gamma(i,j\text{-}1)\,\}$$



DTW – Exercise 1

- Given the following input time series:

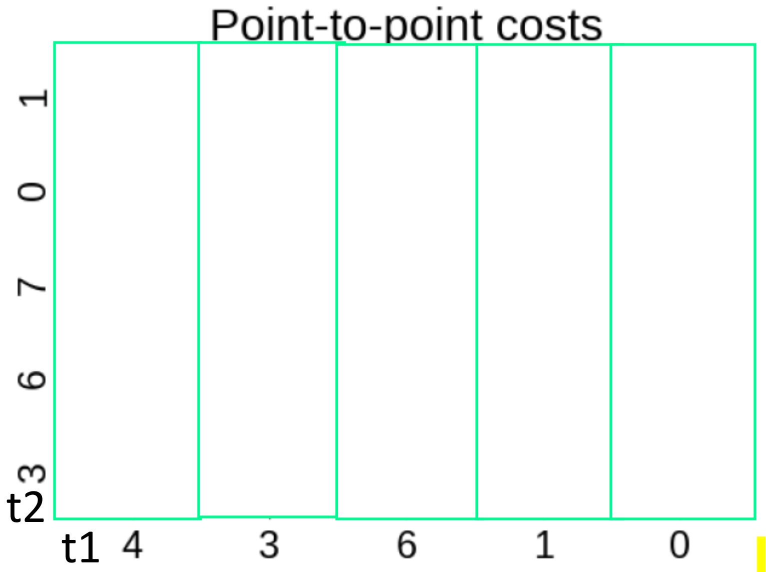
t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A) Compute the distance between "t1" and "t2", using the DTW with distance between points computed as d(x,y) = |x – y|.
- B) If we repeat the computation of point (A) above, this time with a Sakoe-Chiba band of size r=1, does the result change? Why?
- C) If we compute DTW(T1,T2), where T1 is equal to t1 in reverse order (namely T1=<0,1,6,3,4>) and similarly for T2 (namely T2=<1,0,7,6,3>), is it true that DTW(T1,T2) = DTW(t1,t2)? Discuss the problem without providing any computation.

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

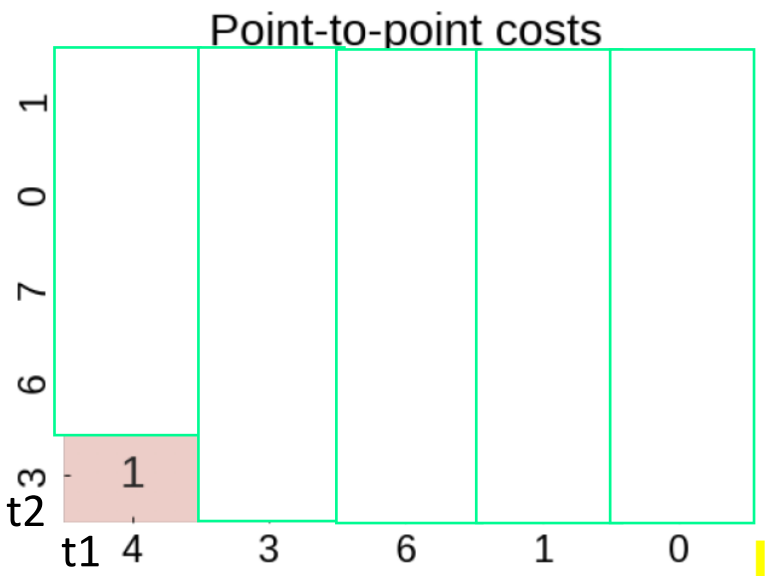


Result: 4

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

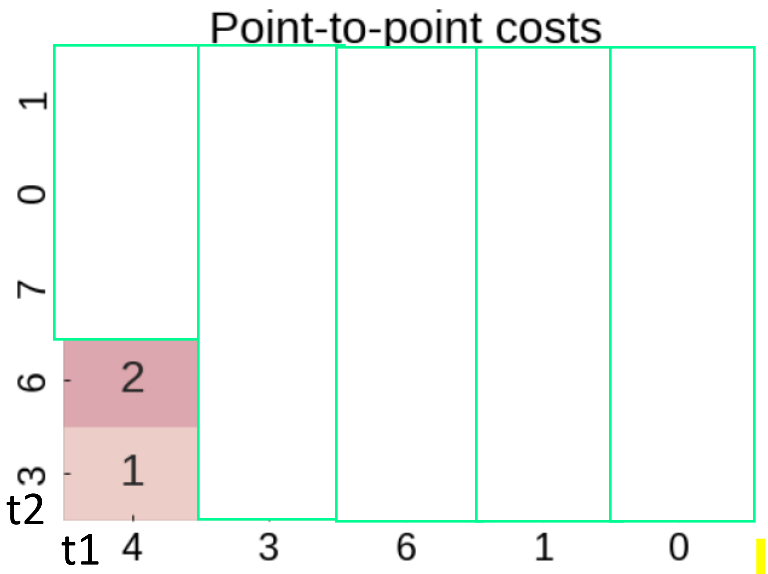


Result: 4

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

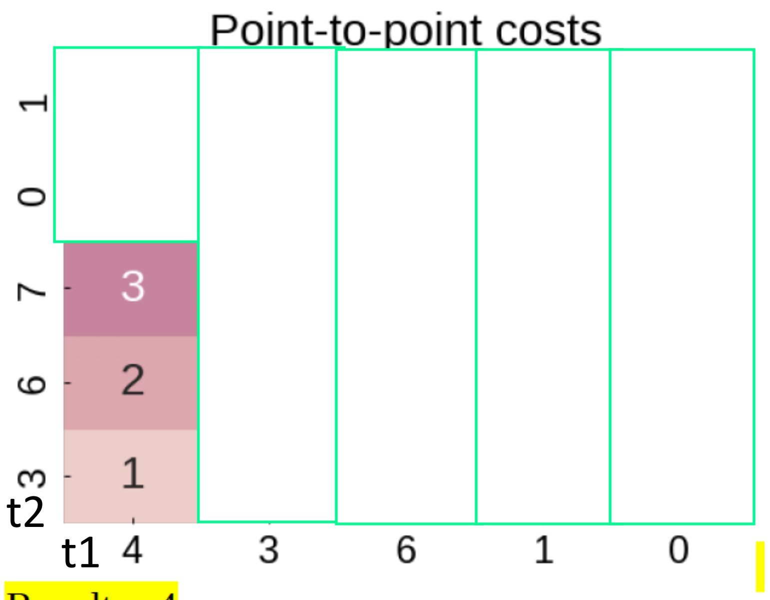


Result: 4

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

DTW – Exercise 1 - Solution

- A)

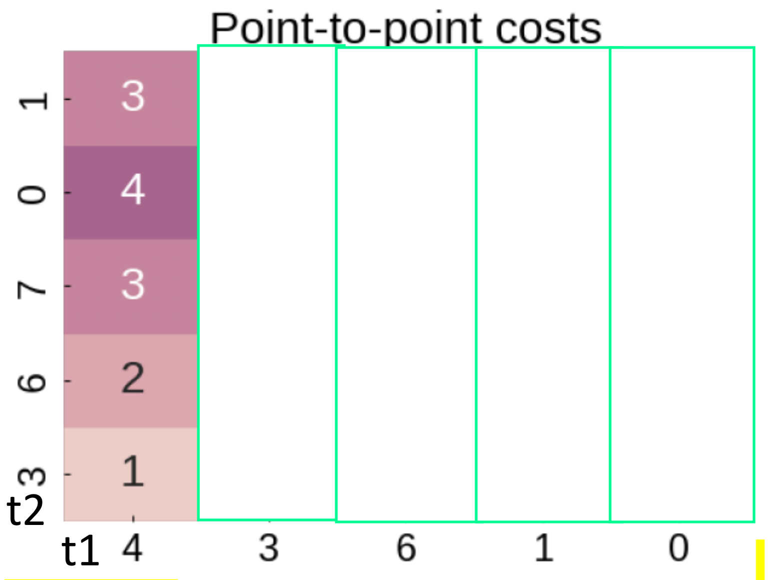


Result: 4

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

DTW – Exercise 1 - Solution

- A)

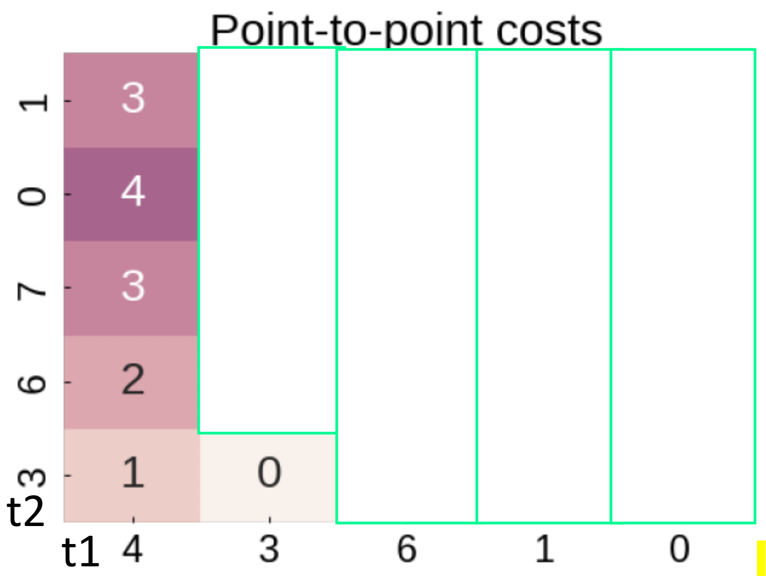


Result: 4

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

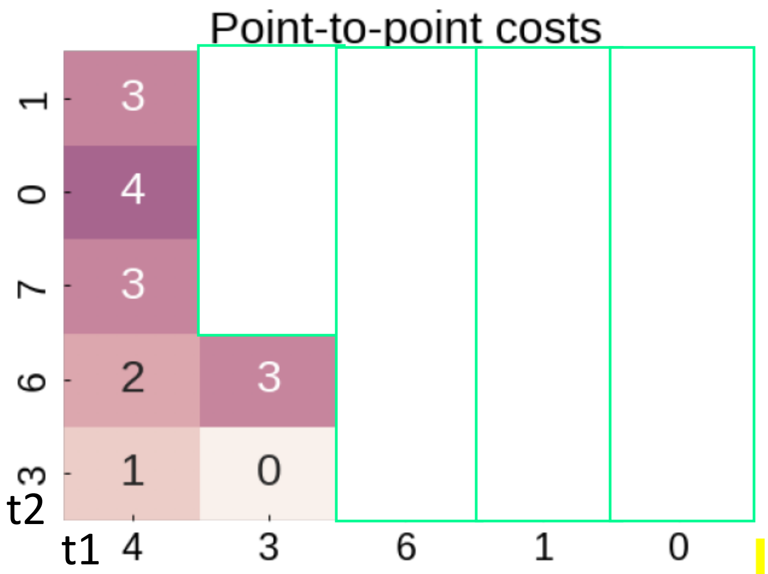


Result: 4

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

DTW – Exercise 1 - Solution

- A)

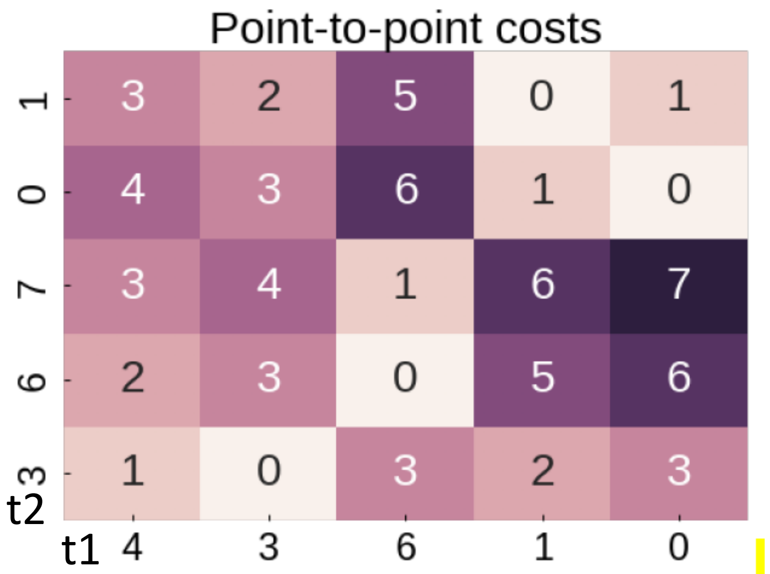


Result: 4

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

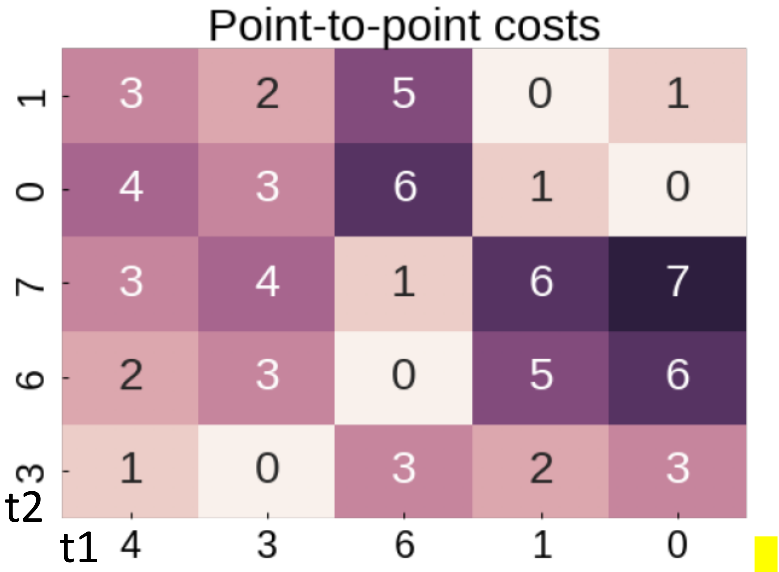


Result: 4

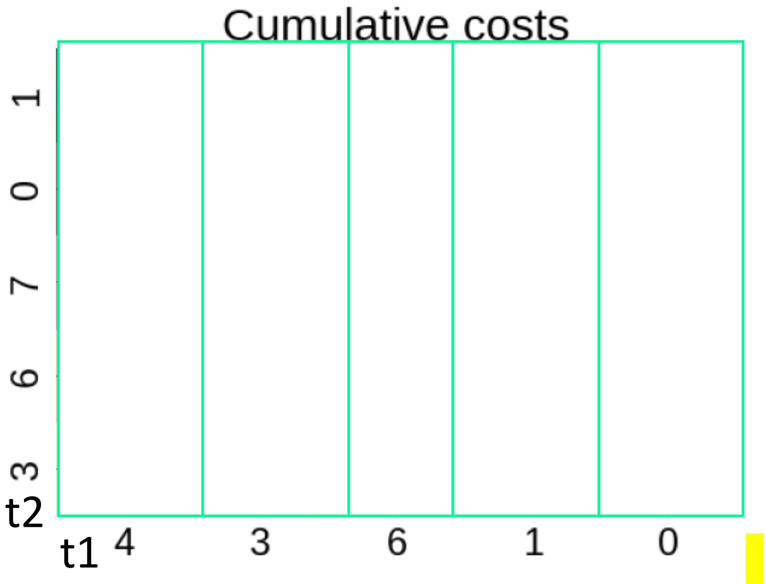
DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)



Result: 4

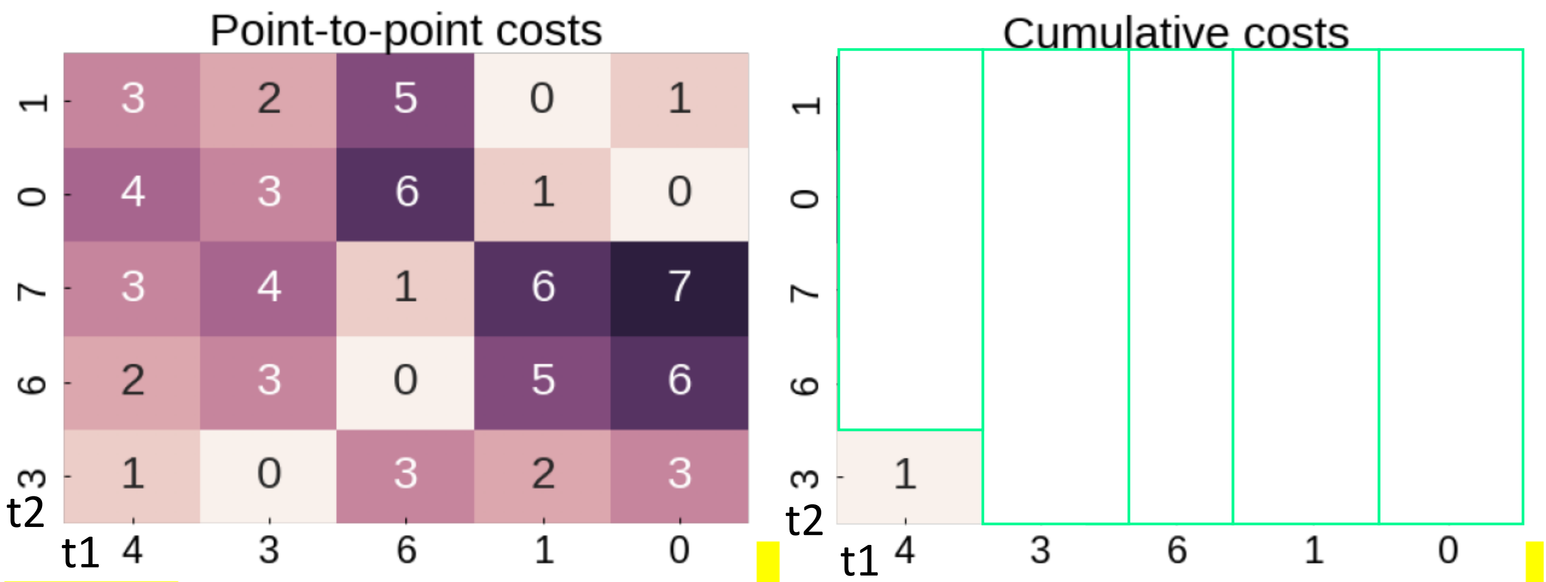


$$\gamma(i,j) \;=\; d(q_i,c_j) + \min\{\, \gamma(i-1,j-1),\, \gamma(i-1,j\,),\, \gamma(i,j-1) \,\}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)



Result: 4

$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

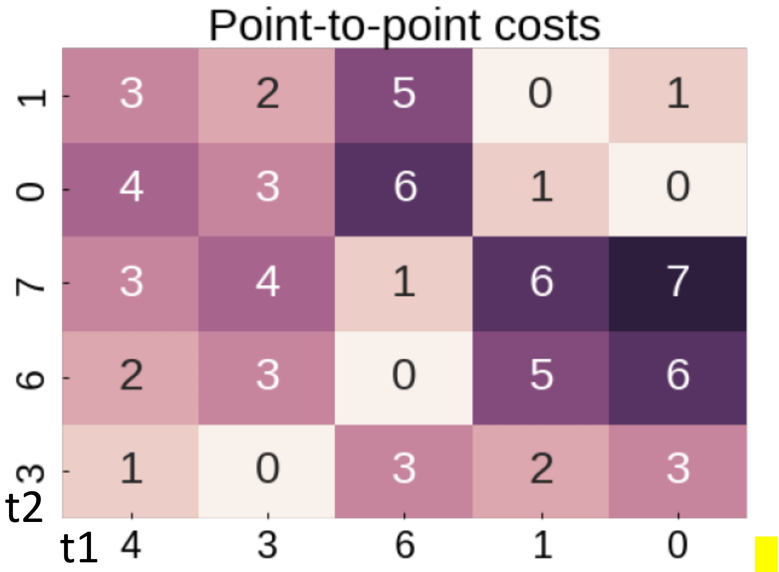
Result: 4

$$\gamma(i,j) = d(q_i,c_j) + \min\{\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)\}$$

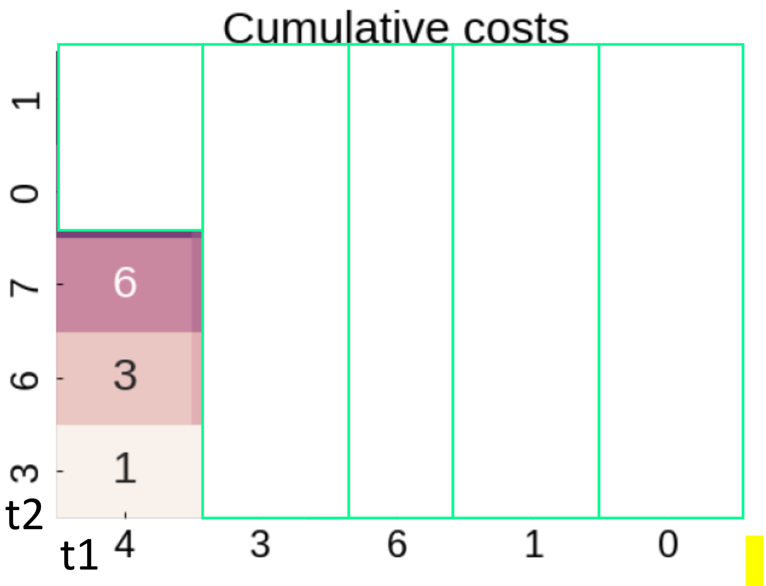
DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)



Result: 4

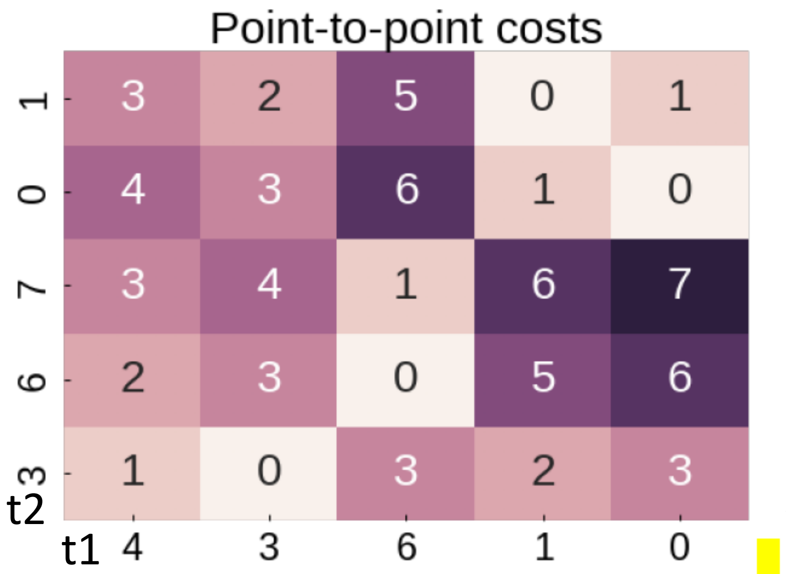


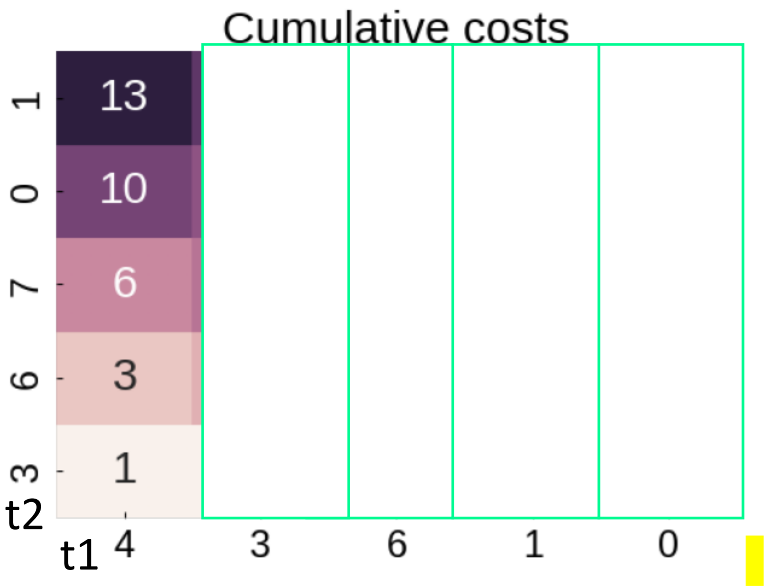
$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)





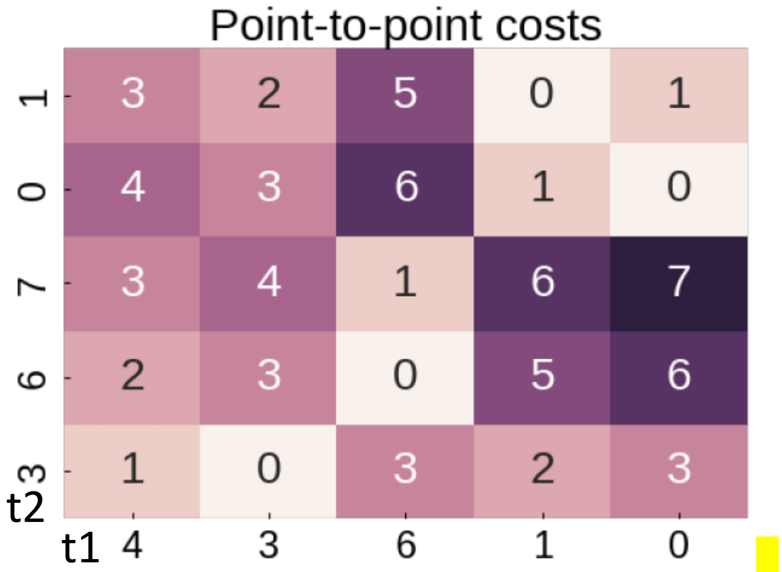
Result: 4

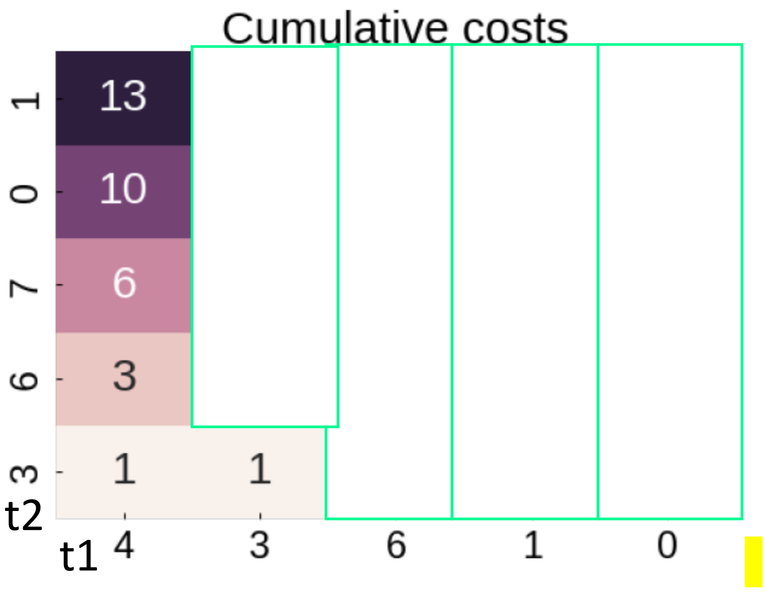
$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i\text{-}1,j\text{-}1), \gamma(i\text{-}1,j), \gamma(i,j\text{-}1) \}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)





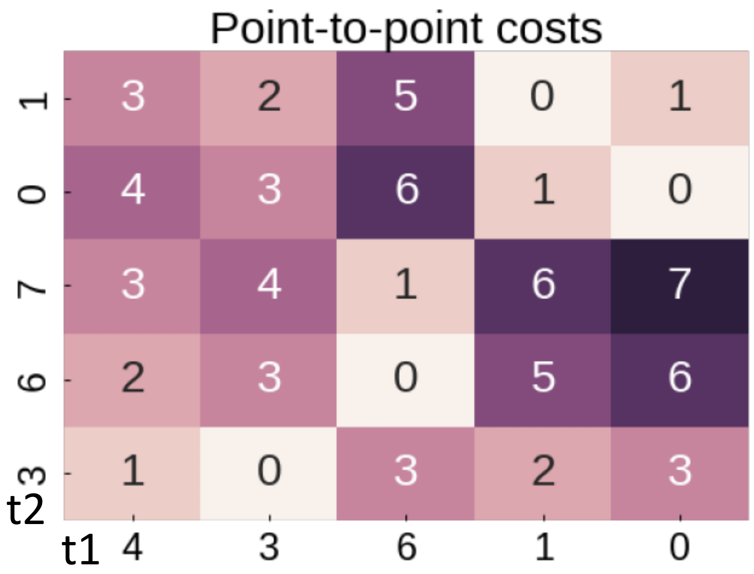
Result: 4

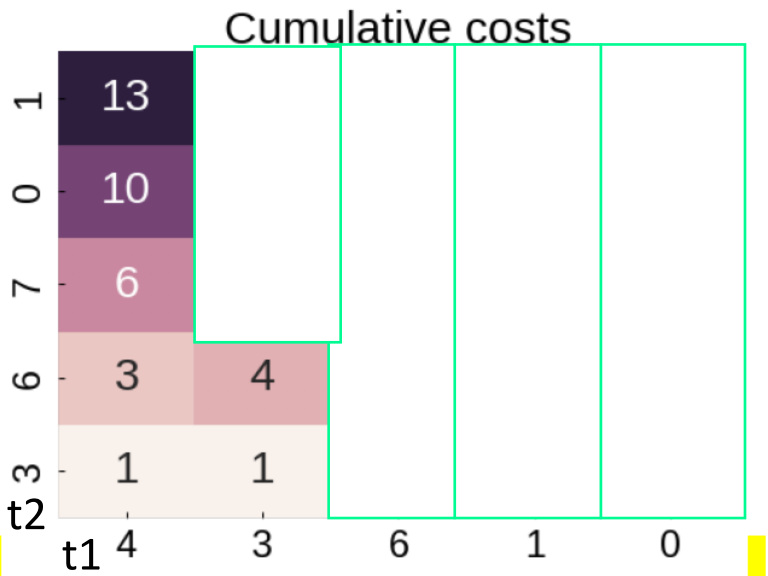
$$\gamma(i,j) = d(q_i,c_j) + \min\{\, \gamma(i-1,j-1),\ \gamma(i-1,j),\ \gamma(i,j-1) \,\}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)





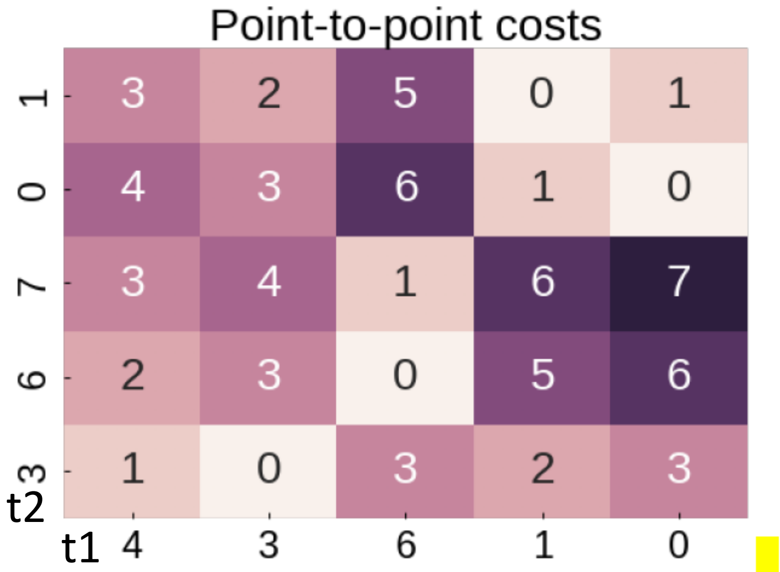
Result: 4

$$\gamma(i,j) \;=\; d(q_i,c_j) + \min\{\, \gamma(i\text{-}1,j\text{-}1),\ \gamma(i\text{-}1,j),\ \gamma(i,j\text{-}1) \,\}$$

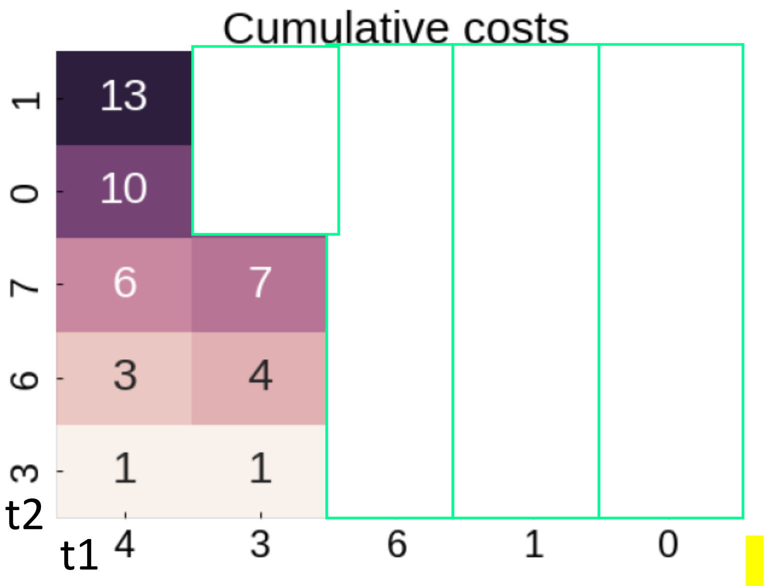
DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)



Result: 4

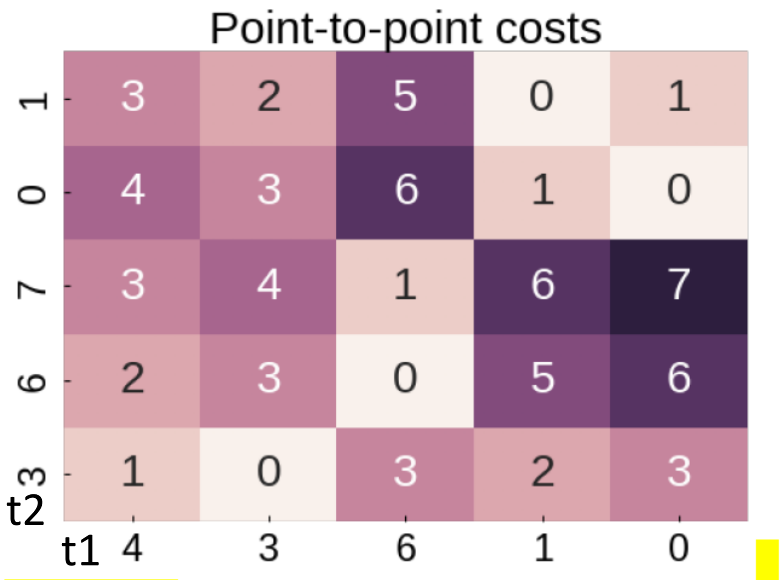


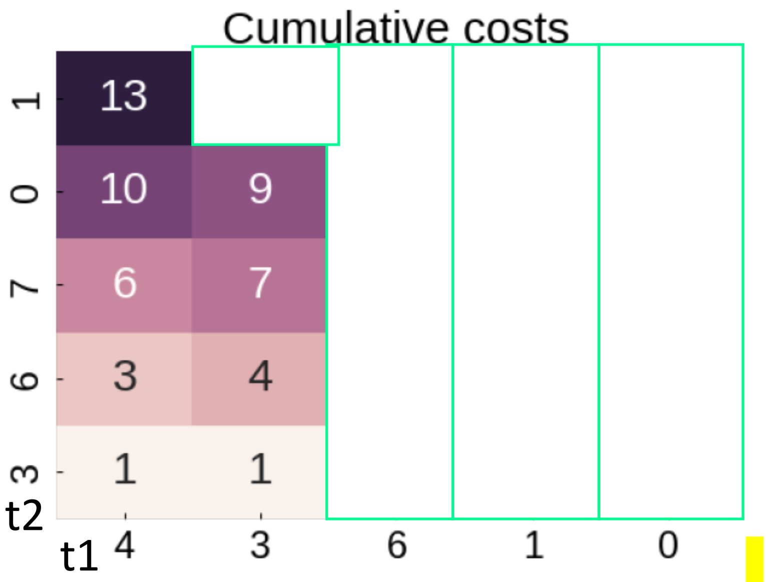
$$\gamma(i,j) \;=\; d(q_i,c_j) + \min\{\, \gamma(i-1,j-1),\, \gamma(i-1,j),\, \gamma(i,j-1) \,\}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)





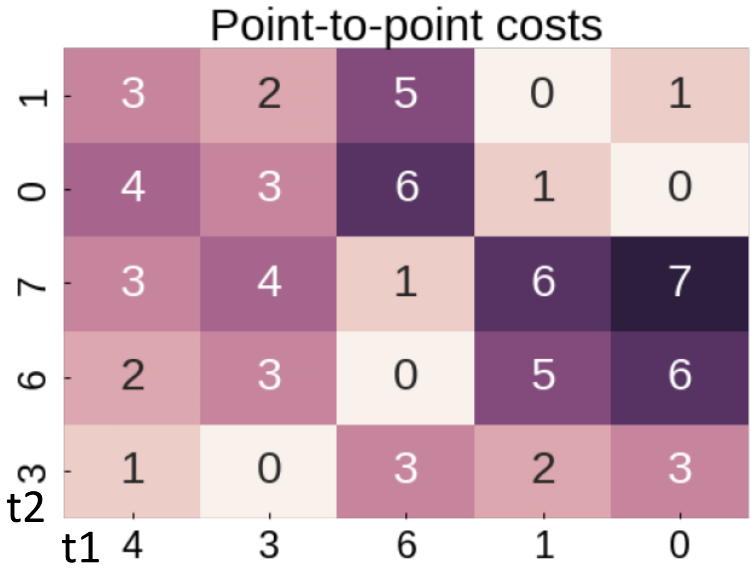
Result: 4

$$\gamma(i,j) = d(q_i,c_j) + \min\{\, \gamma(i-1,j-1),\ \gamma(i-1,j),\ \gamma(i,j-1) \,\}$$

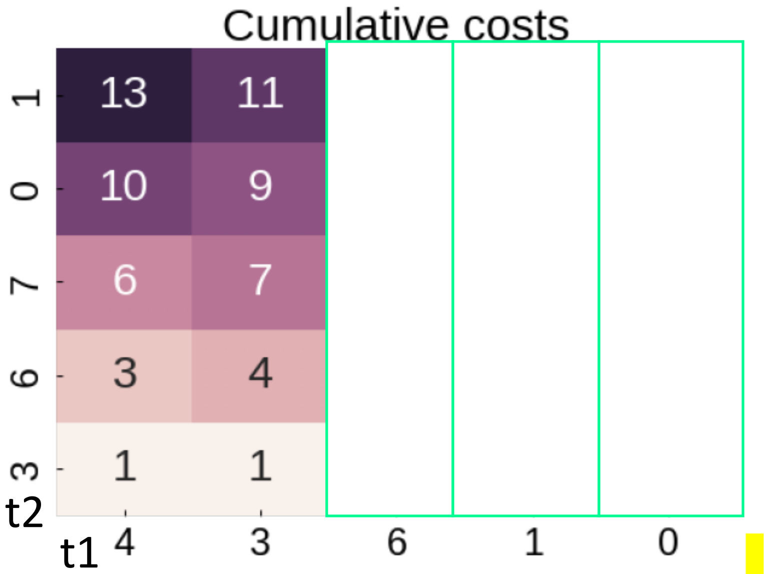
DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)



Result: 4

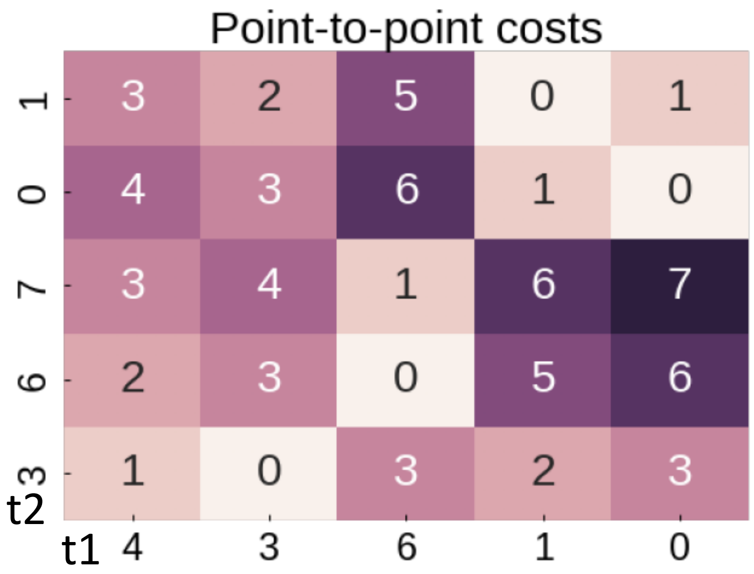


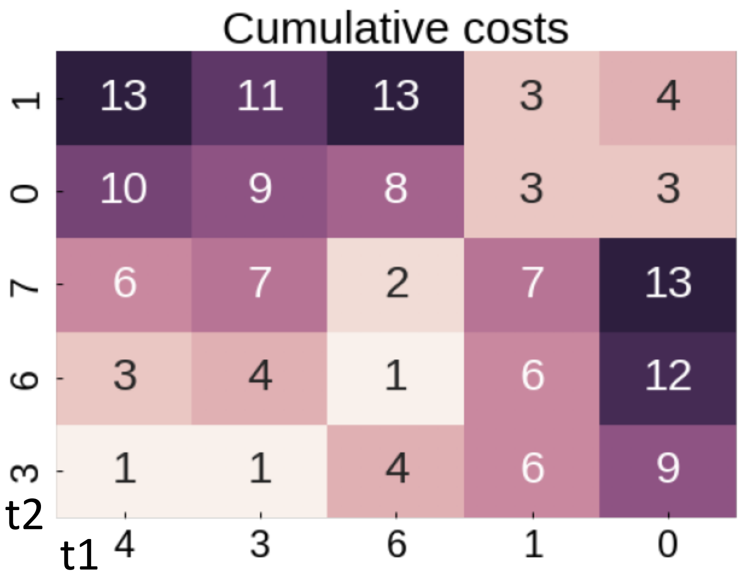
$$\gamma(i,j) = d(q_i, c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \}$$

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

DTW – Exercise 1 - Solution

- A)





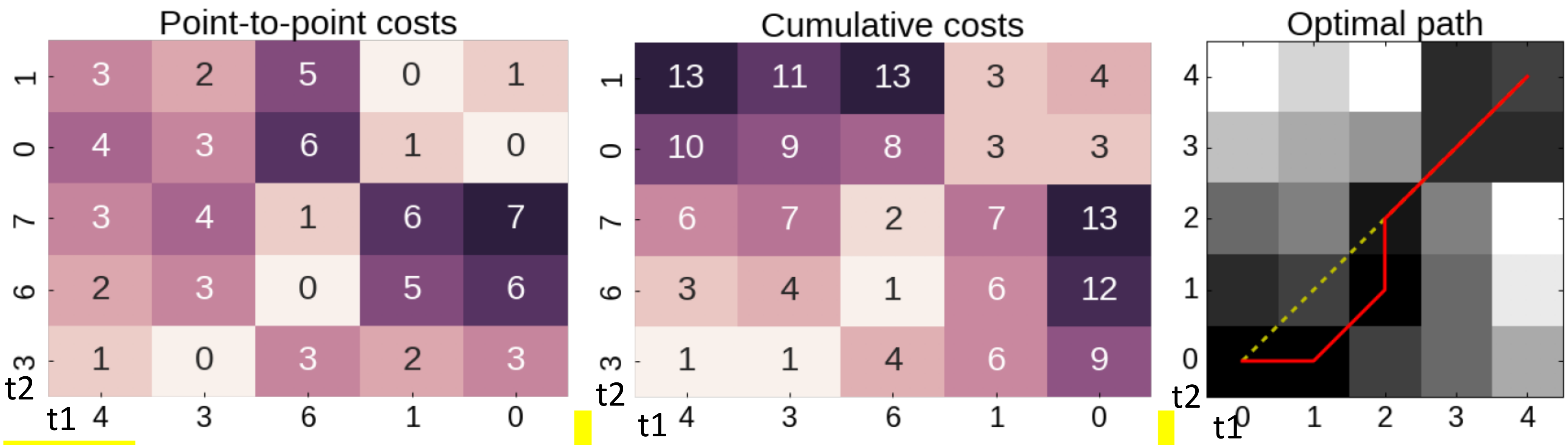
Result: 4

$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \}$$

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

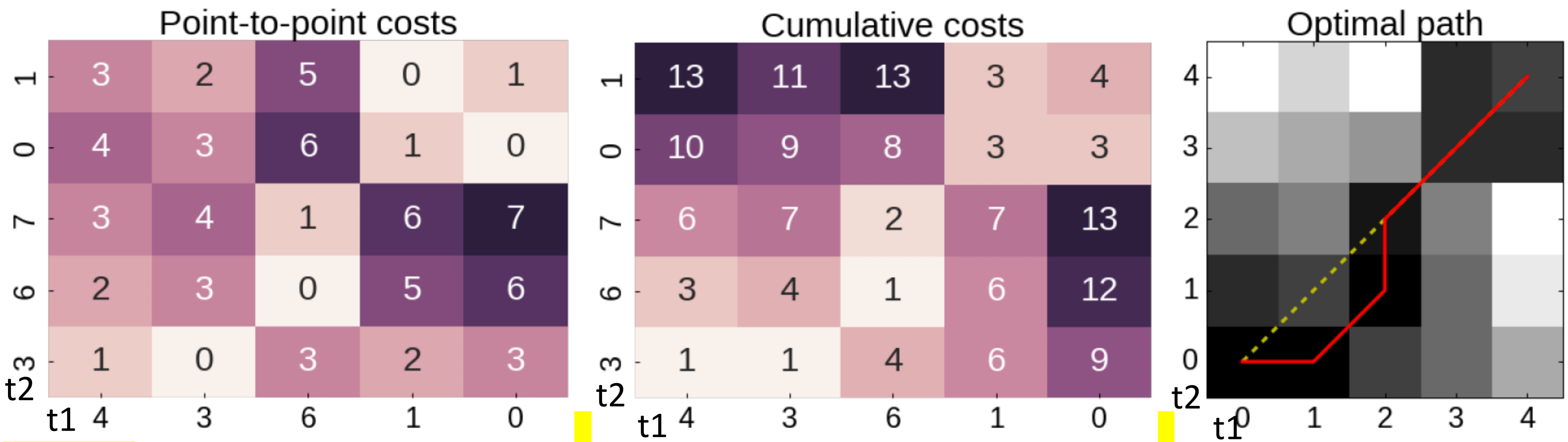


Result: 4

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)



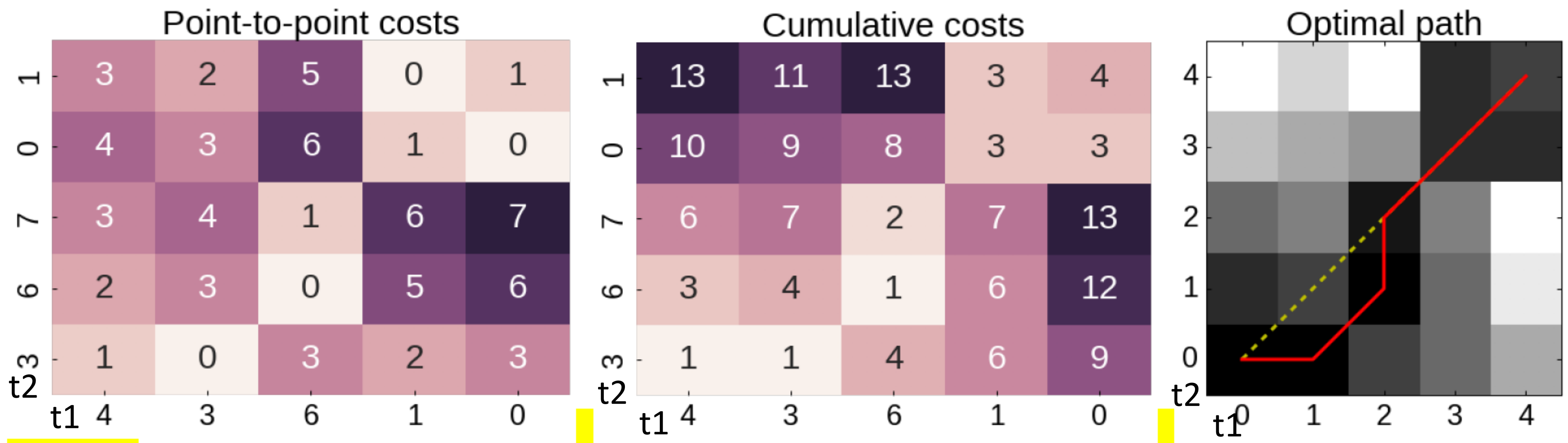
Result: 4

- B) No. Because the DTW optimal path remains inside the band of size r=1

DTW – Exercise 1 - Solution

t1	< 4, 3, 6, 1, 0 >
t2	< 3, 6, 7, 0, 1 >

- A)

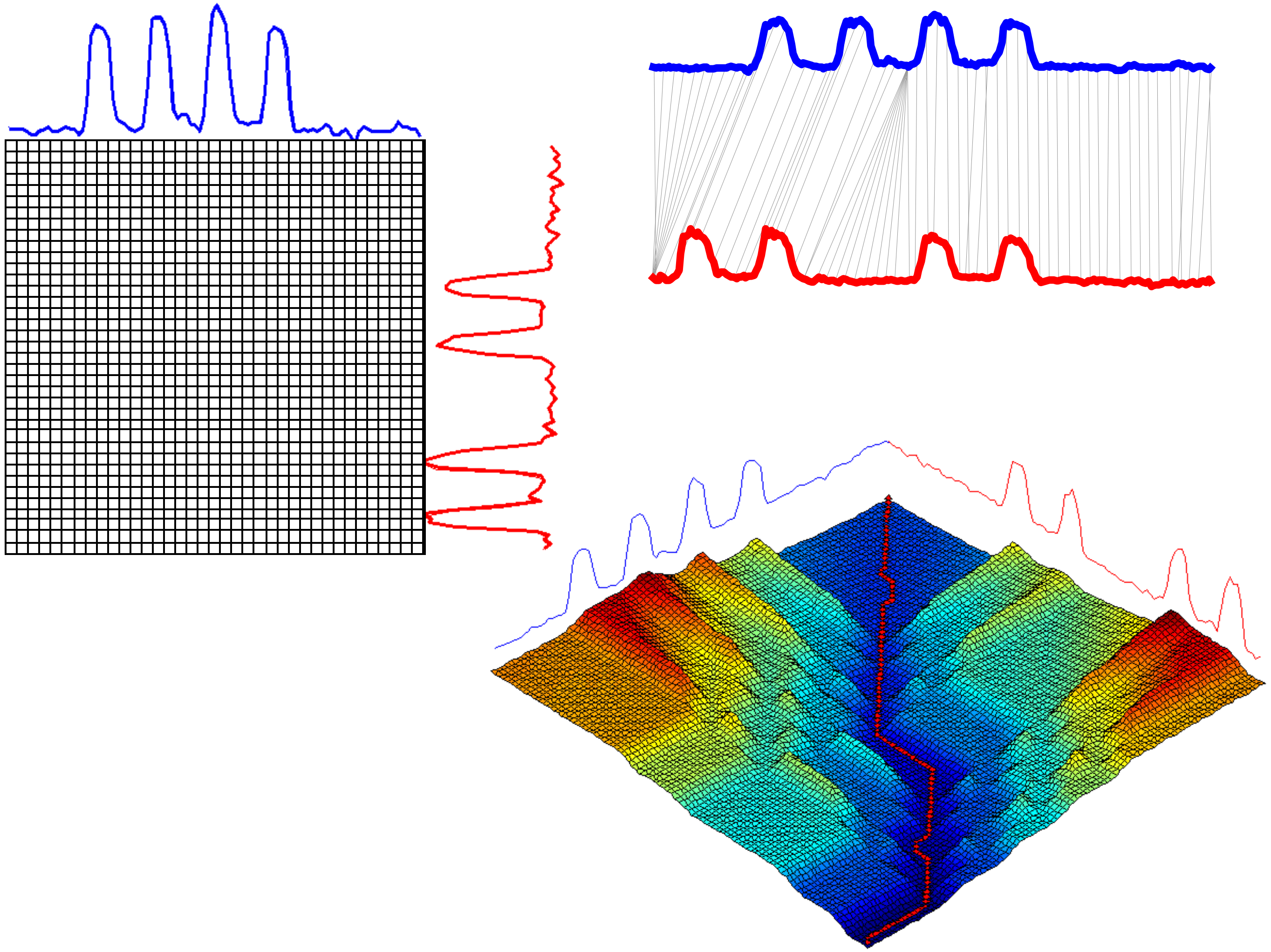


Result: 4

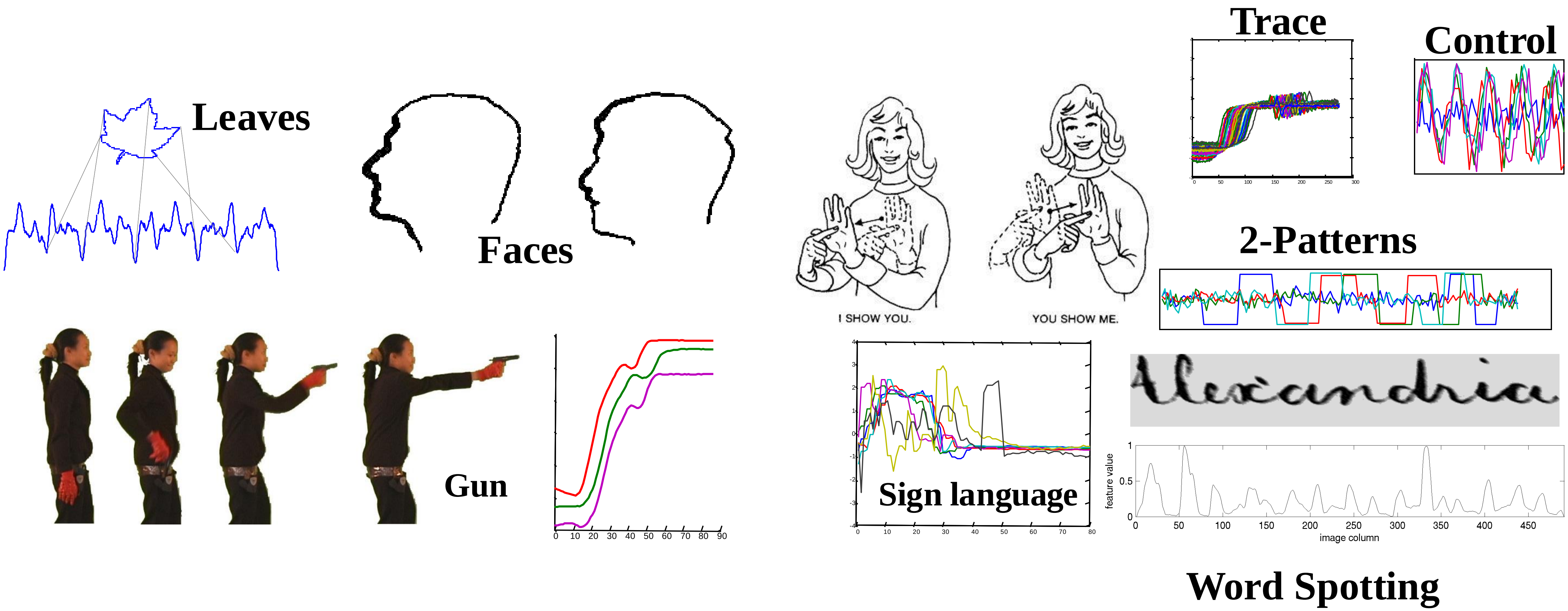
- B) No. Because the DTW optimal path remains inside the band of size r=1
- C) Yes. The optimal path in one direction is the same in the opposite direction. Though, the cumulative costs matrix might look different.

Dynamic Time Warping – A Real Example

- A Real Example
- This example shows 2 one-week periods from the power demand time series.
- Note that although they both describe 4-day work weeks, the blue sequence had Monday as a holiday, and the red sequence had Wednesday as a holiday.



Comparison of Euclidean Distance and DTW



Comparison of Euclidean Distance and DTW

- Classification using 1-NN
- Class(x) = class of most similar training object
- Leaving-one-out evaluation
- For each object: use it as test set, return overall average

Error Rate

Dataset	Euclidean	DTW
Word Spotting	4.78	1.10
Sign language	28.70	25.93
GUN	5.50	1.00
Nuclear Trace	11.00	0.00
Leaves#	33.26	4.07
(4) Faces	6.25	2.68
Control Chart*	7.5	0.33
2-Patterns	1.04	0.00

Comparison of Euclidean Distance and DTW

- Classification using 1-NN
- Class(x) = class of most similar training object
- Leaving-one-out evaluation
- For each object: use it as test set, return overall average
- DTW is two to three orders of magnitude slower than Euclidean distance.

Milliseconds

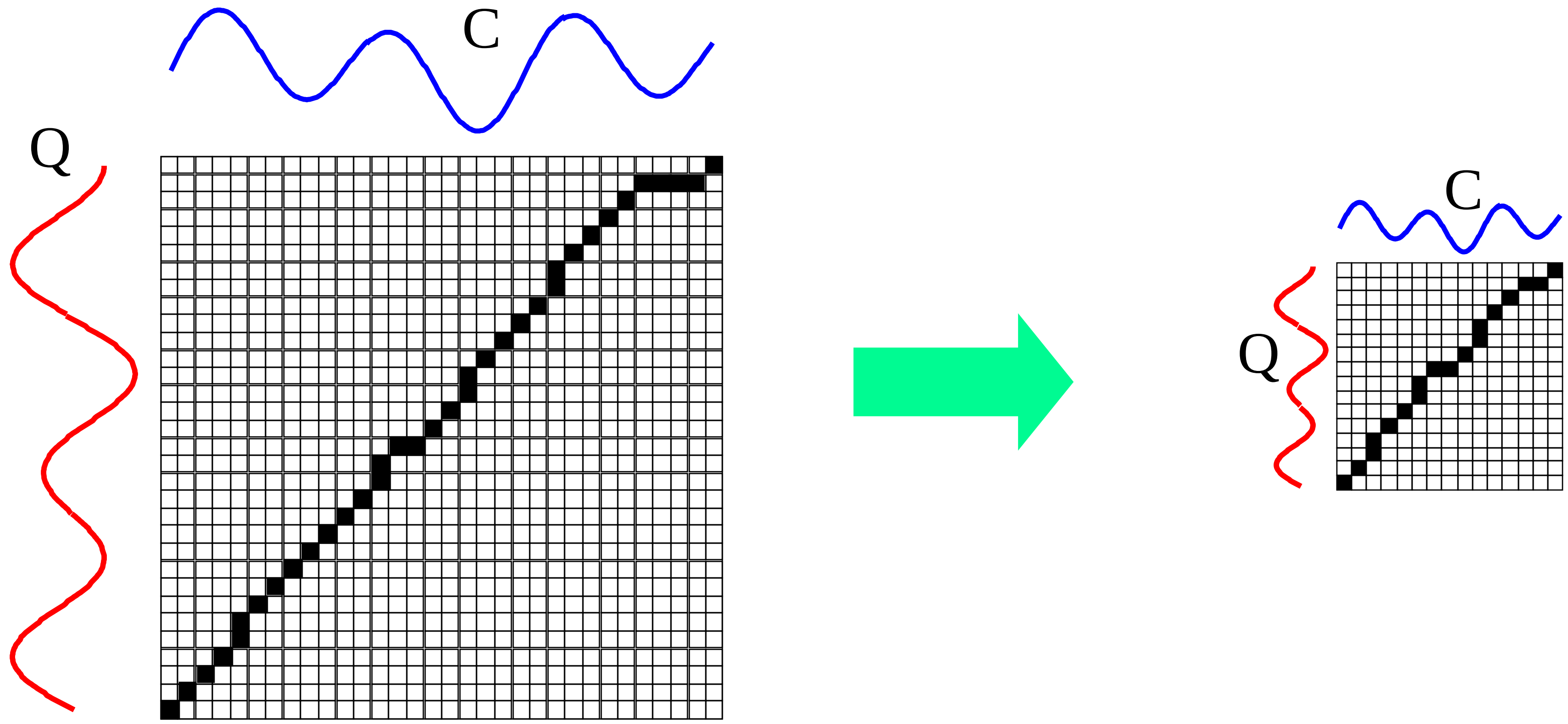
Dataset	**Euclidean**	**DTW**
Word Spotting	40	8,600
Sign language	10	1,110
GUN	60	11,820
Nuclear Trace	210	144,470
Leaves	150	51,830
(4) Faces	50	45,080
Control Chart	110	21,900
2-Patterns	16,890	545,123

What we have seen so far…

- Dynamic Time Warping gives much better results than Euclidean distance on many problems.
- Dynamic Time Warping is very very slow to calculate!
- Is there anything we can do to speed up similarity search under DTW?

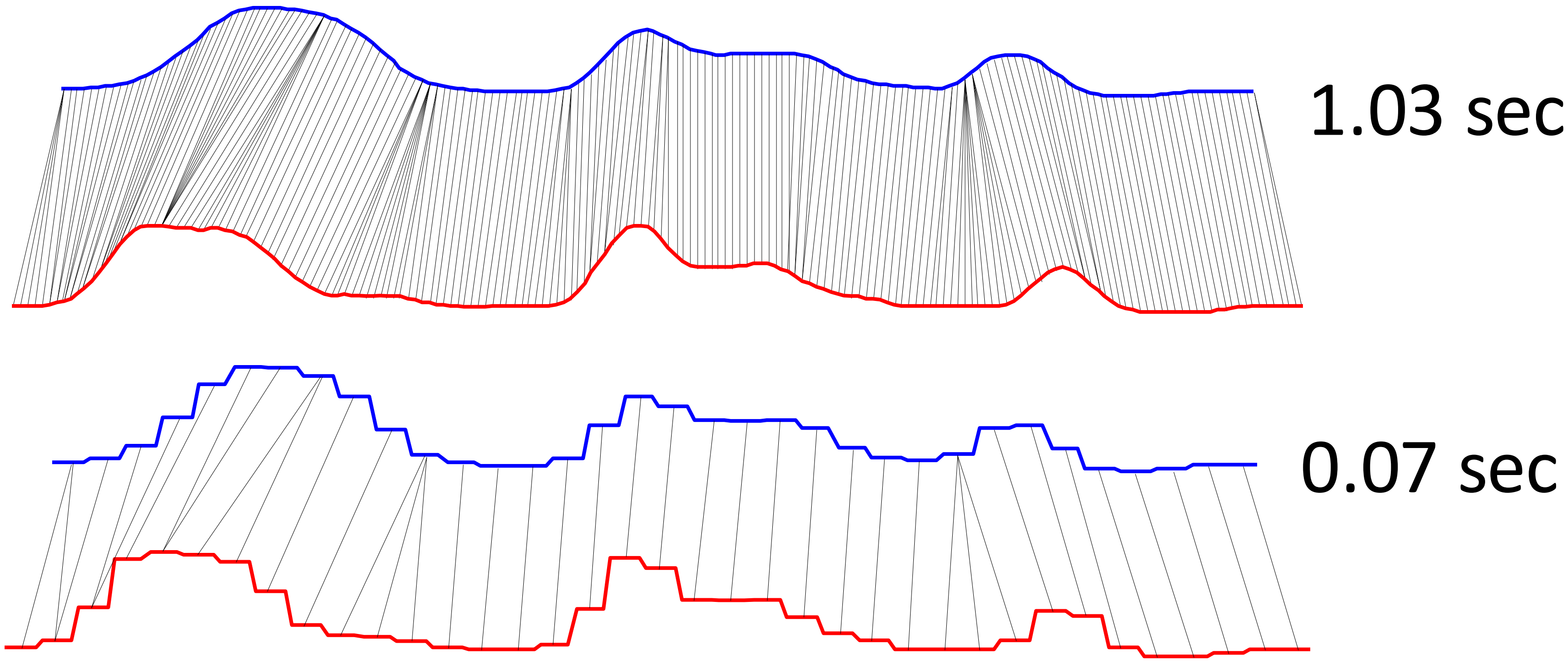
Fast Approximations to DTW

- Approximate the time series with some compressed or downsampled representation, and do DTW on the new representation.



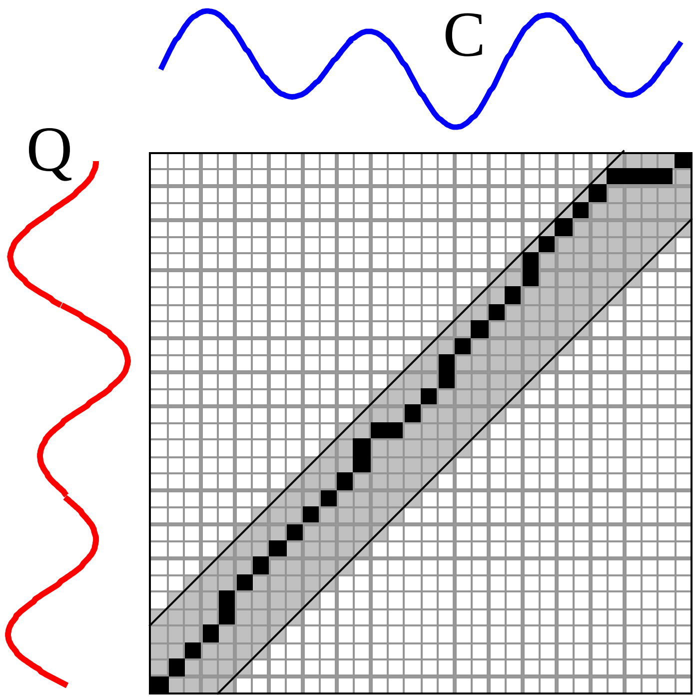
Fast Approximations to DTW

- There is strong visual evidence to suggests it works well
- In the literature there is good experimental evidence for the utility of the approach on clustering, classification, etc.

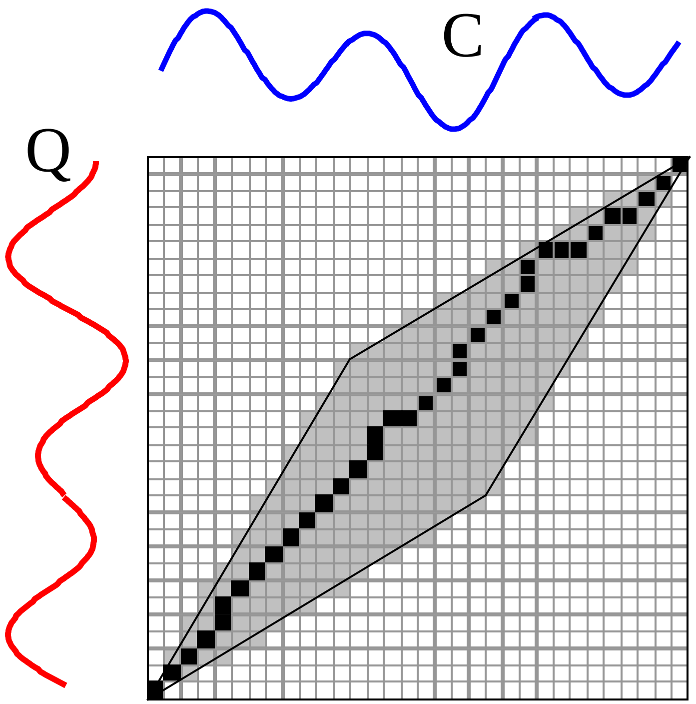


Global Constraints

- Slightly speed up the calculations
- Prevent pathological warpings



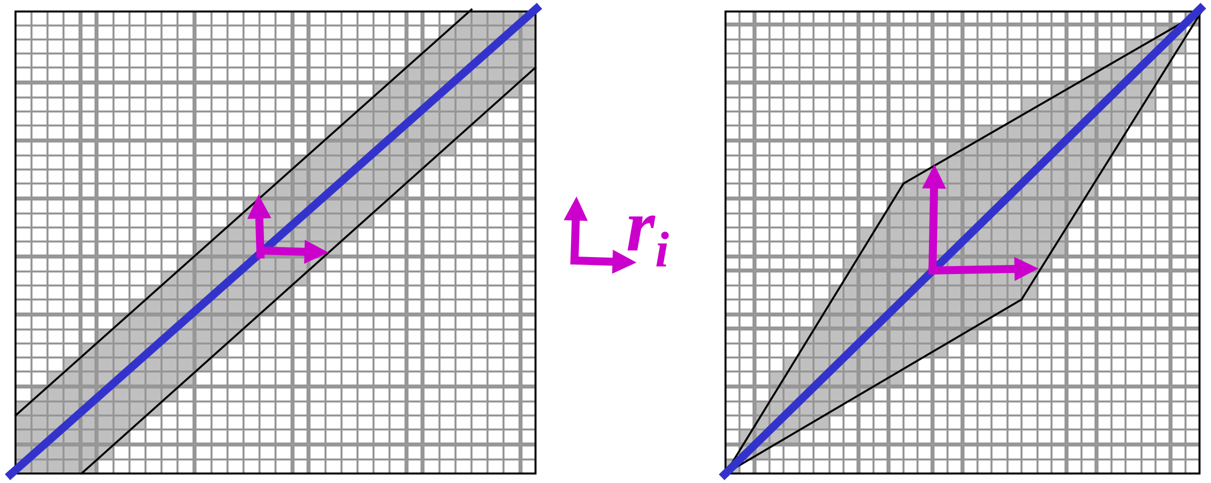
Sakoe-Chiba Band



Itakura Parallelogram

Global Constraints

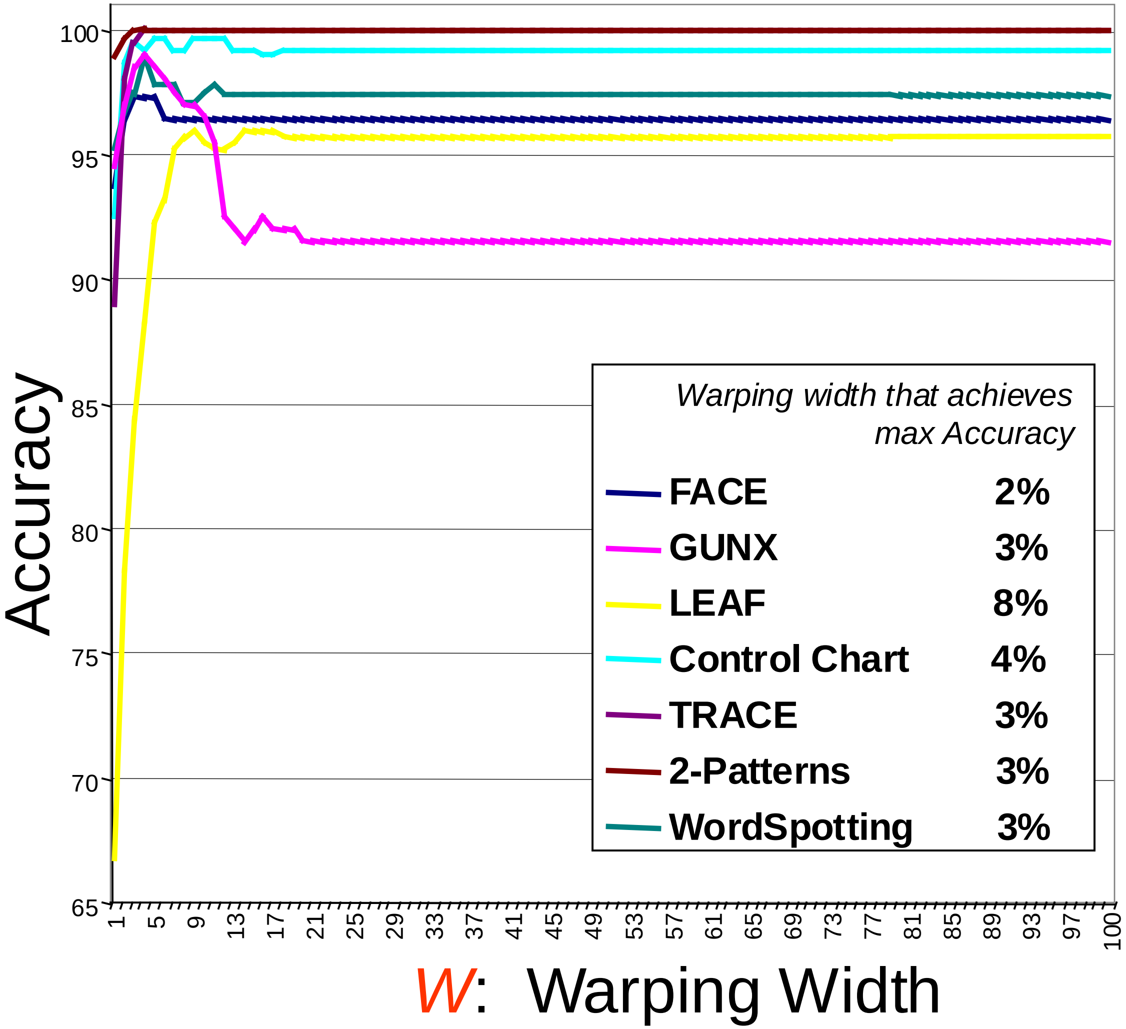
- A global constraint constrains the indices of the warping path $w_k = (i,j)_k$ such that $j\text{-}r \leq i \leq j+r$, where *r* is a term defining allowed range of warping for a given point in a sequence.
- r can be considered as a *window* that reduces the number of calculus.

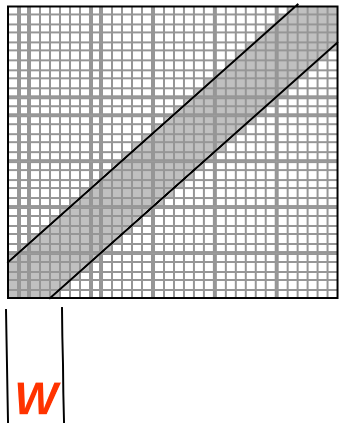


Sakoe-Chiba Band

Itakura Parallelogram

Accuracy vs. Width of Warping Window

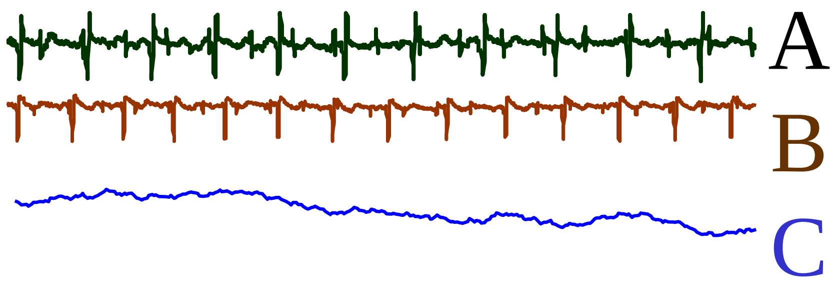




Structural-based Similarities

Structure or Model Based Similarity

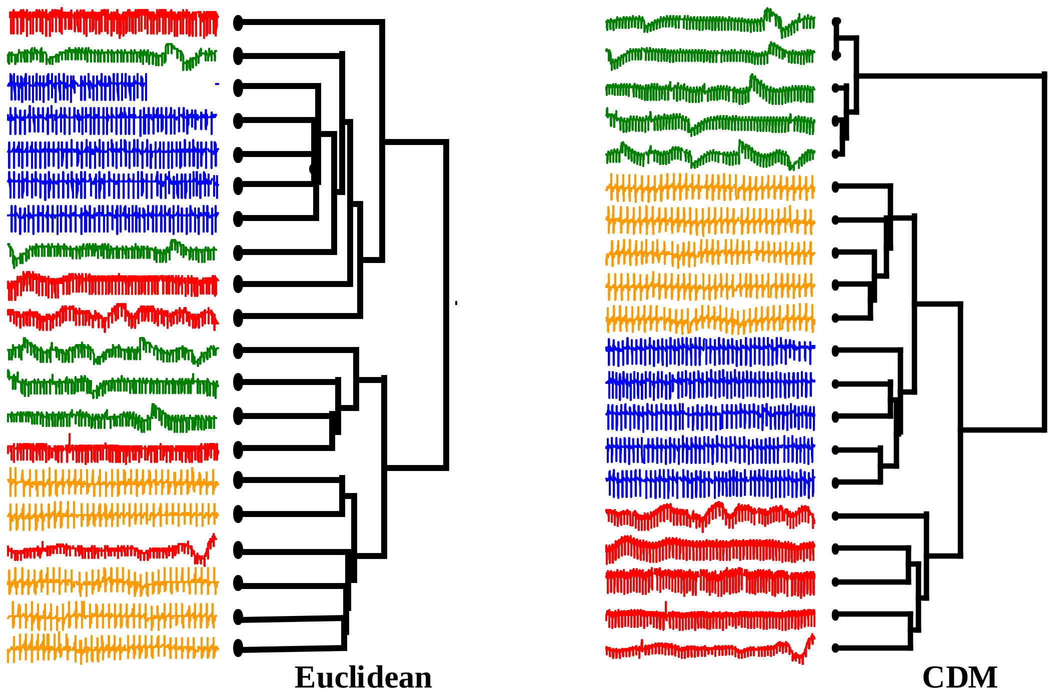
- For long time series, shape-based similarity give very poor results.
- We need to measure similarly based on high level structure.
- The basic idea is to:
    1. extract *global* features from the time series,
    2. create a feature vector, and
    3. use it to measure similarity and/or classify
- Example of features:
    - mean, variance, skewness, kurtosis,
    - 1^{st} derivative mean, 1^{st} derivative variance, …
    - parameters of regression, forecasting, Markov model



Feature\Time Series	A	B	C
Max Value	11	12	19
Mean	5.3	6.4	4.8
Min Value	3	2	5
Autocorrelation	0.2	0.3	0.5
…	…	…	…

Compression Based Dissimilarity

- Use as features whatever structure a given compression algorithm finds.
- $d(x, y) = CDM(x, y) = \frac{C(x,y)}{C(x)+C(y)}$
- Time series can be compressed using various transformations:
  - Piecewise Linear Approximation
  - Adaptive Piecewise Constant Approximation
  - Symbolic Aggregate Approximation



Time Series Approximation

Time Series Approximation

- **Approximation**: represent a TS into a new smaller and simpler space and use this novel representation for computing.

- **Approximation** is a special form of **Dimensionality Reduction** specifically designed for TSs.

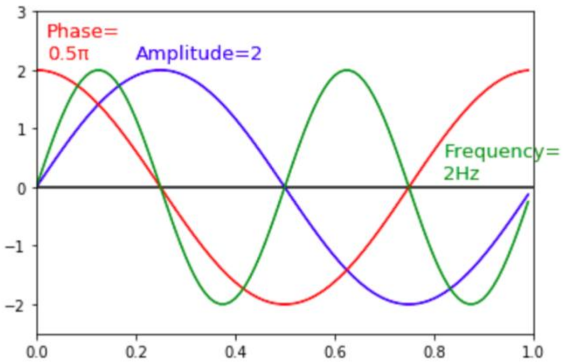
- **Approximation vs Compression**:
  - the approximated space is always understandable
  - the compressed space is not necessarily understandable.

Discrete Fourier Transform (DFT)

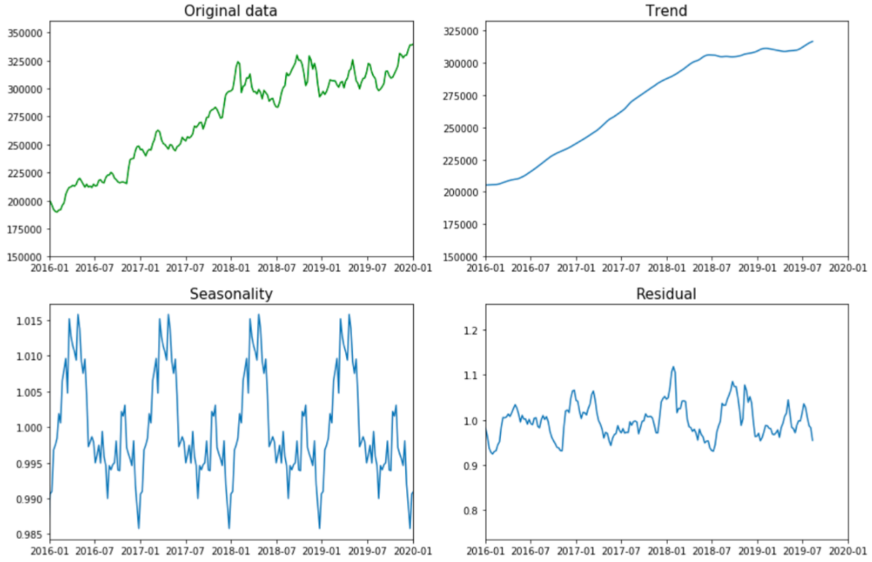
Jean Fourier: 1768-1830

- Apply a spectral decomposition of a signal
- DTF is a method to decompose functions depending on **time** into functions depending on **frequency**
- TS is a function depending on time
  - we have a value for temperature for each point in time.

Frequency is the number of complete cycles

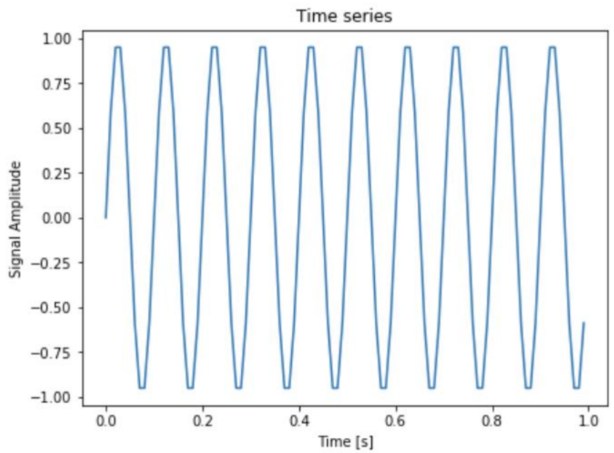


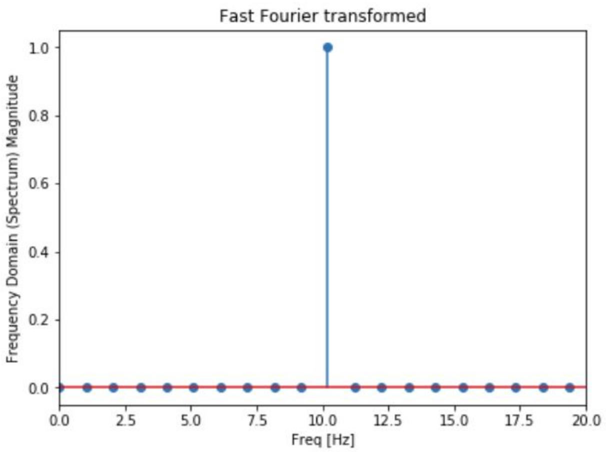
TS: a combination of seasonality, trend, and noise



- DFT extracts **different** seasonality patterns from a single time series variable
- Example: Given an hourly temperature data set, DFT can detect the presence of **day/night variations and summer/winter variations**
  - it will tell you that those two seasonality (frequencies) are present in your data.

Discrete Fourier Transform (DFT)

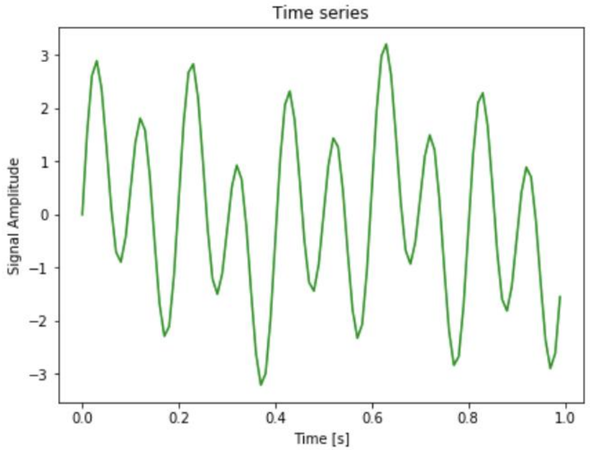


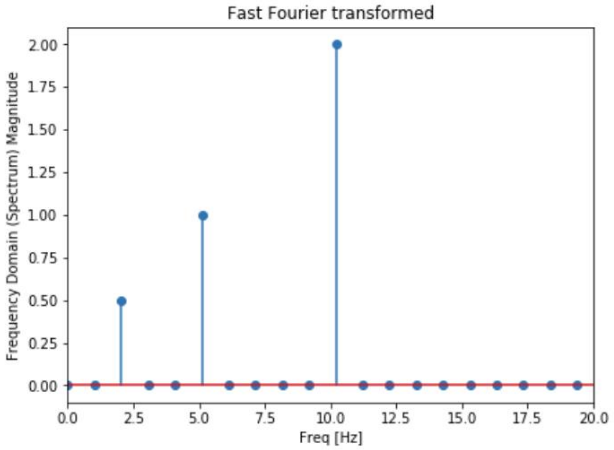


- A peak value at 10 Hz with a magnitude of one while all other frequencies are around zero.

- The original TS where has 10 complete cycles in a second with an amplitude of one.

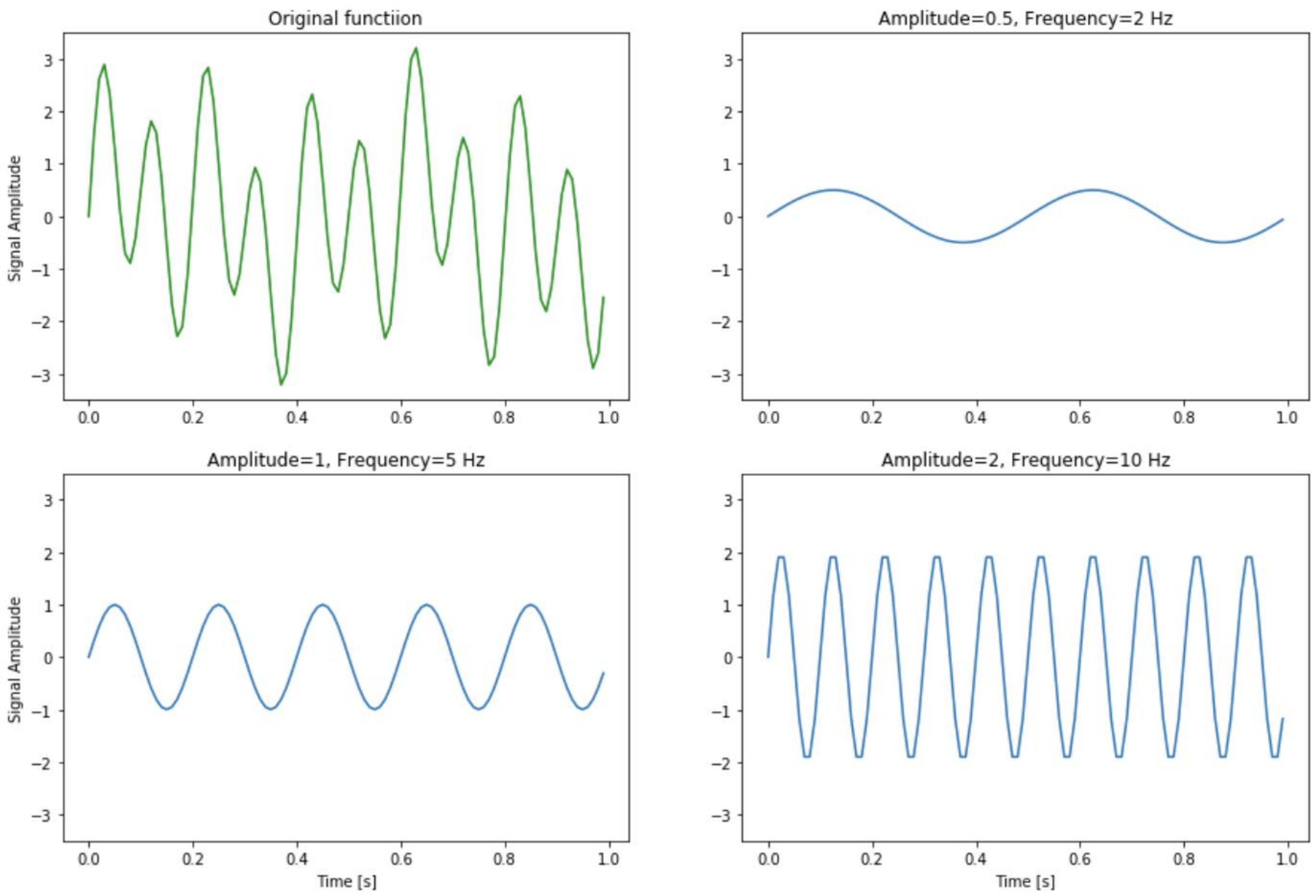
Jean Fourier: 1768-1830





- Data comprises of 3 different elementary components with 3 different frequencies (2, 5 and 10 Hz) at 3 different amplitudes (0.5, 1 and 2).

Discrete Fourier Transform (DFT)



- Data comprises of 3 different elementary components with 3 different frequencies (2, 5 and 10 Hz) at 3 different amplitudes (0.5, 1 and 2).

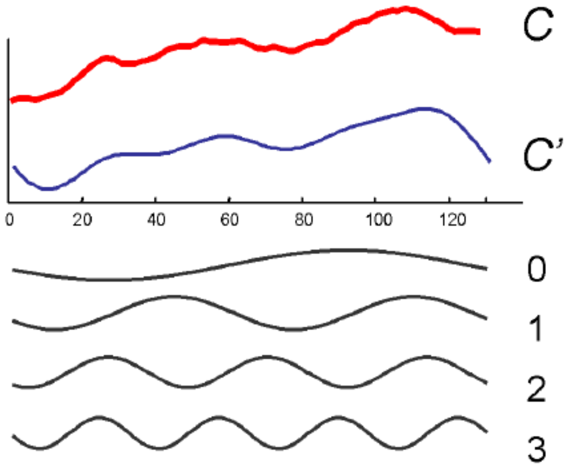
- Sine functions of the different components.

Discrete Fourier Transform (DFT)

- The basic idea of spectral decomposition is that any signal can be represented by the super position of a finite number of sine/cosine waves
- Each wave is represented by a single complex number known as a Fourier coefficient as a linear combination of sines and cosines
- Keep only the first n/2 coefficients
- Many of the Fourier coefficients have very low amplitude and thus contribute little to reconstructed signal.
- These low amplitude coefficients can be discarded without much loss of information thereby saving storage space.
- Pros
  - Good ability to compress most natural signals.
  - Fast, off the shelf DFT algorithms exist. O(nlog(n)).
- Cons
  - Difficult to deal with sequences of different lengths.
  - Cannot support weighted distance measures.

Jean Fourier

1768-1830



Piecewise Linear Approximation (PLA)

Karl Friedrich Gauss

1777 - 1855

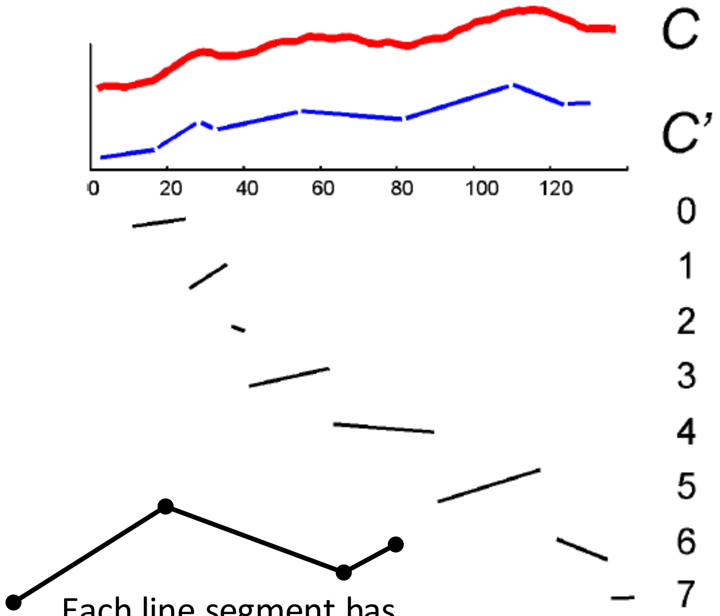
- Represent the time series as a sequence of straight lines.
- Lines could be connected or disconnected

- In the literature there are numerous algorithms available for **segmenting** time series.

- An open question is how to best choose K, the “optimal” number of segments used to represent a particular time series.

- This problem involves a tradeoff between accuracy and compactness, and clearly has no general solution.

- Pros:
  - data compression
  - noise filtering
  - able to support some interesting non-Euclidean similarity measures

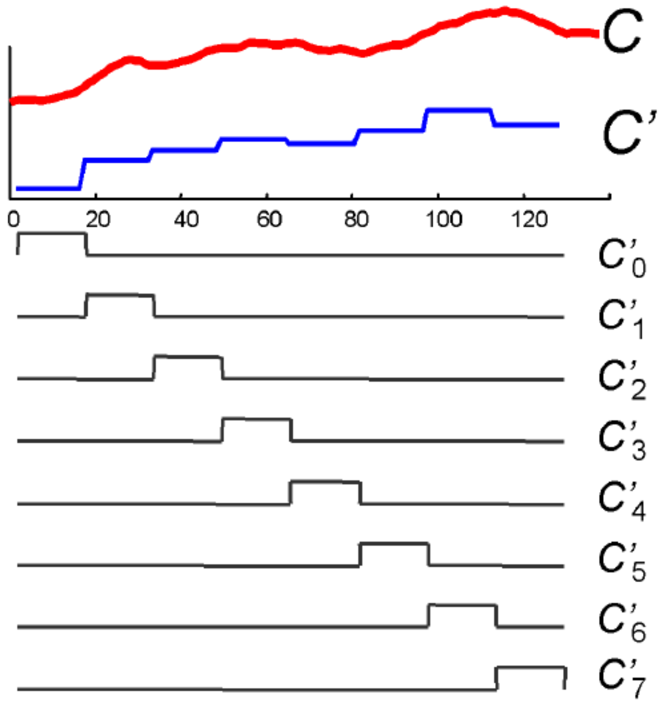


Each line segment has
- `length`
- `left_height`

(`right_height` can be inferred by looking at the next segment)

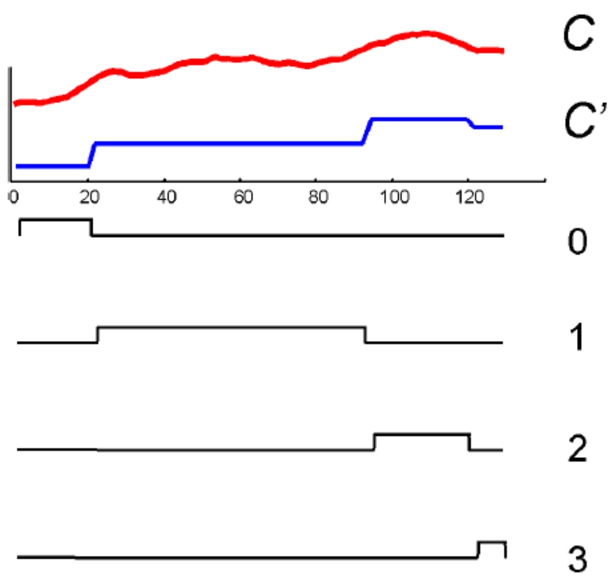
Piecewise Aggregate Approximation (PAA)

- Represent the time series as a sequence of box basis functions with each box of the same size.
- It approximates a TS by dividing it into equal-length segments and recording the **mean value of the data points** that fall within the segment.
- It reduces the data from *n* dimensions to *M* dimensions by dividing the time series into *M* equi-sized ``frames''.
- The **mean value of the data falling within a frame** is calculated, and a vector of these values becomes the data reduced representation.
- Pros
    - Extremely fast to calculate
    - Supports non Euclidean measures
    - Supports weighted Euclidean distance



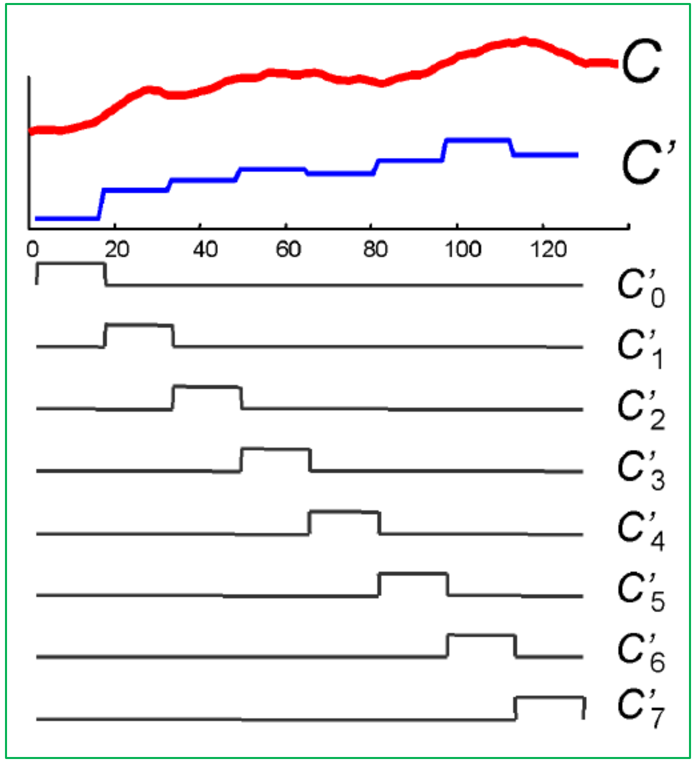
Adaptive Piecewise Constant Approximation (APCA)

- It allows the **segments to have arbitrary lengths**, which in turn needs two numbers per segment.
- The first number records the mean value of all the data points in segment, and the second number records the length of the segment.
- APCA has the advantage of being able to place a single segment in an area of low activity and many segments in areas of high activity.
- In addition, one has to consider the structure of the data in question.
- Pros:
  - Fast to calculate O(n)
  - Supports non Euclidean measures
  - Supports weighted Euclidean distance



Symbolic Aggregate Approximation (SAX)

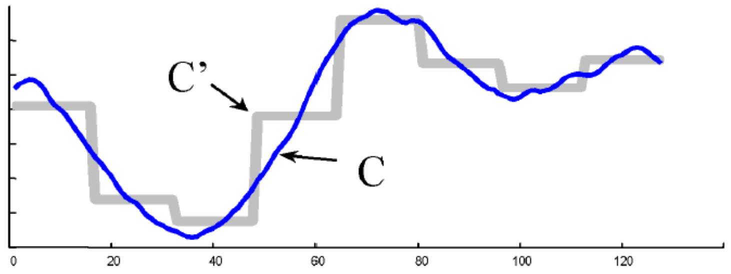
- Convert the data into a discrete format, with a small alphabet size.
- A time series ***T*** of length ***n*** is divided into ***w*** equal-sized segments; the values in each segment are then approximated and replaced by a single coefficient, which is their average.
- **Aggregating these *w* coefficients form the PAA representation of T.**
- Next, we determine the breakpoints that divide the distribution space into ***a*** equiprobable regions, where ***a*** is the alphabet size specified by the user
- The breakpoints are determined such that the probability of a segment falling into any of the regions is approximately the same.
- If the symbols are not equi-probable, some of the substrings would be more probable than others. Consequently, we would inject a probabilistic bias in the process.

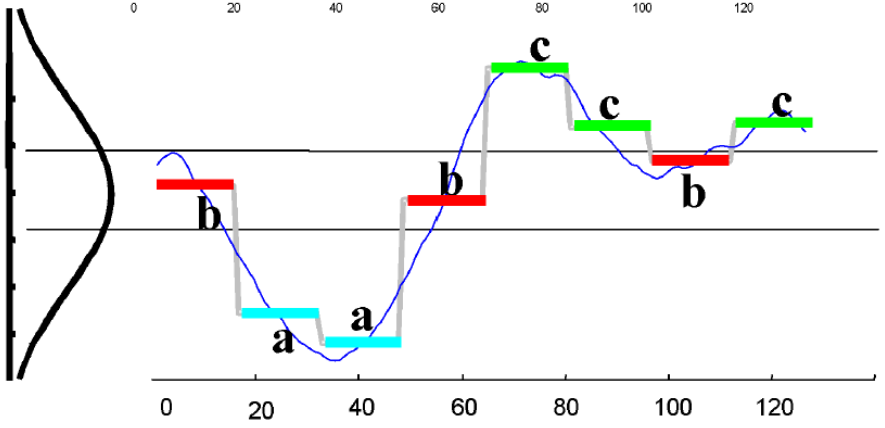


Symbolic Aggregate Approximation (SAX)

- Once the breakpoints are determined, each region is assigned a symbol.
- The PAA coefficients can then be easily mapped to the symbols corresponding to the regions in which they reside.
- The symbols are assigned in a bottom-up fashion, i.e., the PAA coefficient that falls in the lowest region is converted to “a”, in the one above to “b”, and so forth.

baabccbc



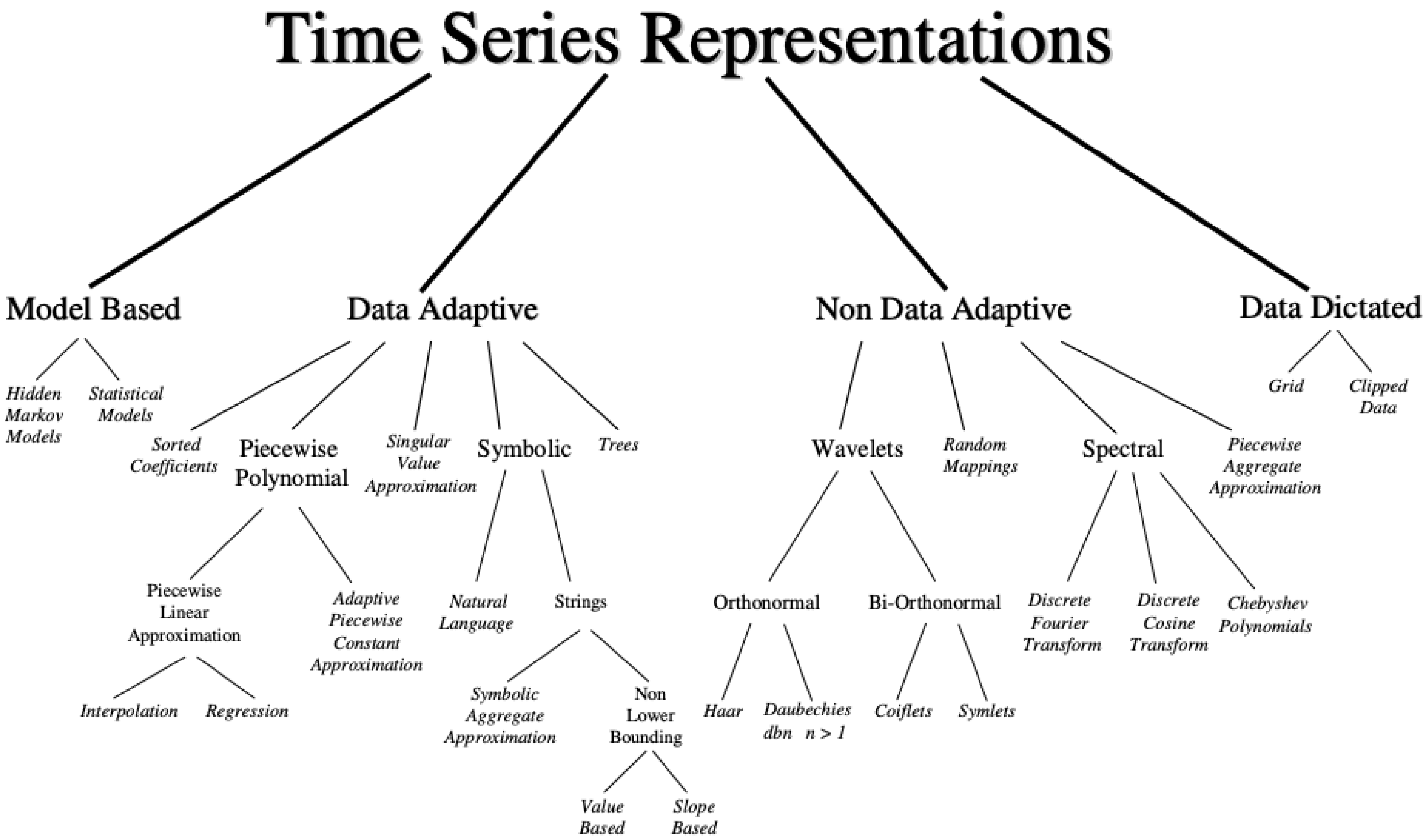


Summary of Time Series Similarity

- If you have short time series
  - use DTW after searching over the warping window size

- If you have long time series
  - if you do know something about your data =>
    extract features
  - and you know nothing about your data =>
    try compression/approximation based dissimilarity

Summary of Time Series Representation



Clustering

Clustering Time Series

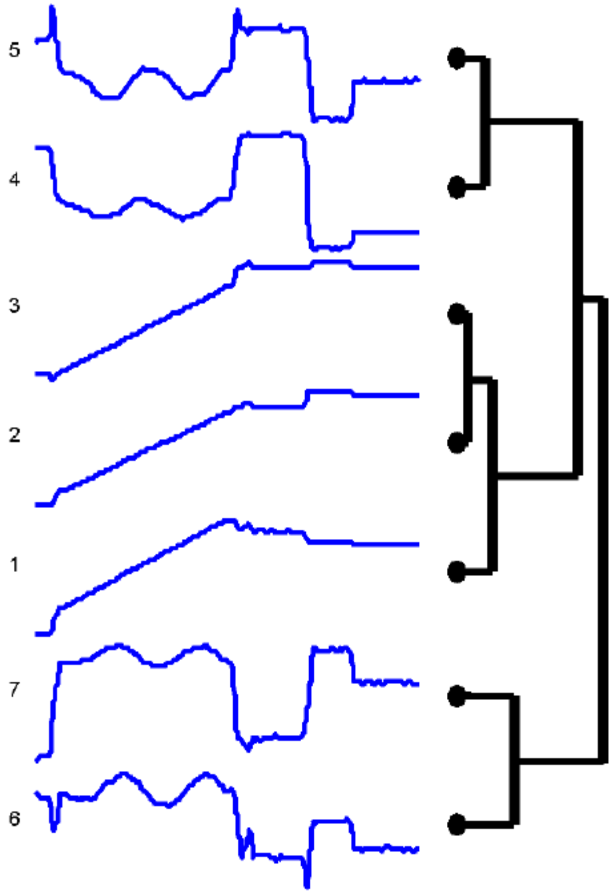
- It is based on the similarity between time series.
- The most similar data are grouped into clusters, but the clusters themselves should be dissimilar.
- These groups to find are not predefined, i.e., it is an unsupervised learning task.
- The two general methods of time series clustering are
  - Partitional Clustering and
  - Hierarchical Clustering

Types of Time Series Clustering

- ***Whole clustering***: similar to that of conventional clustering of discrete objects. Given a set of individual time series data, the objective is to group similar time series into the same cluster.
- ***Features-based clustering***: extract features, or time series motifs (see next lectures) as the features and use them to cluster time series.
- ***Compression-based clustering***: compress time series and run clustering on the compressed versions.
- ***Subsequence clustering***: given a single time series, subsequence clustering is performed on each individual time series extracted from the long time series with a sliding window.

Hierarchical Clustering

- It ***computes pairwise distance***, and then merges similar clusters in a bottom-up fashion, without the need of providing the number of clusters
- It is one of the best tools to data evaluation, by creating a dendrogram of several time series from the domain of interest.
- Its application is limited to small datasets due to its quadratic computational complexity.



Partitional Clustering

- Typically uses the K-Means algorithm (or some variant) to optimize the objective function by minimizing the sum of squared intra-cluster errors.
- K-Means is perhaps the most commonly used clustering algorithm in the literature, one of its shortcomings is the fact that the number of clusters, K, must be pre-specified.
- Also the ***distance function plays a fundamental role*** both for the quality of the results and for the efficiency.

References

- Forecasting: Principles and Practic. Rob J Hyndman and George Athanasaopoulus. (https://otexts.com/fpp2/)
- Time Series Analysis and Its Applications. Robert H. Shumway and David S. Stoffer. 4th edition.(http://www.stat.ucla.edu/~frederic/221/W21/tsa4.pdf)
- Mining Time Series Data. Chotirat Ann Ratanamahatana et al. 2010. (https://www.researchgate.net/publication/227001229_Mining_Time_Series_Data)
- Dynamic Programming Algorithm Optimization for Spoken Word Recognition. Hiroaki Sakode et al. 1978.
- Experiencing SAX: a Novel Symbolic Representation of Time Series. Jessica Line et al. 2009
- Compression-based data mining of sequential data. Eamonn Keogh et al. 2007.