



Time Series Management

Michele Linardi Ph.D.

michele.linardi@orange.fr

Some of the slides of this course are taken from the **excellent Tutorial of Eammon Keogh**
A Decade of Progress in Indexing and Mining Large Time Series Databases. VLDB 2006.



Syllabus

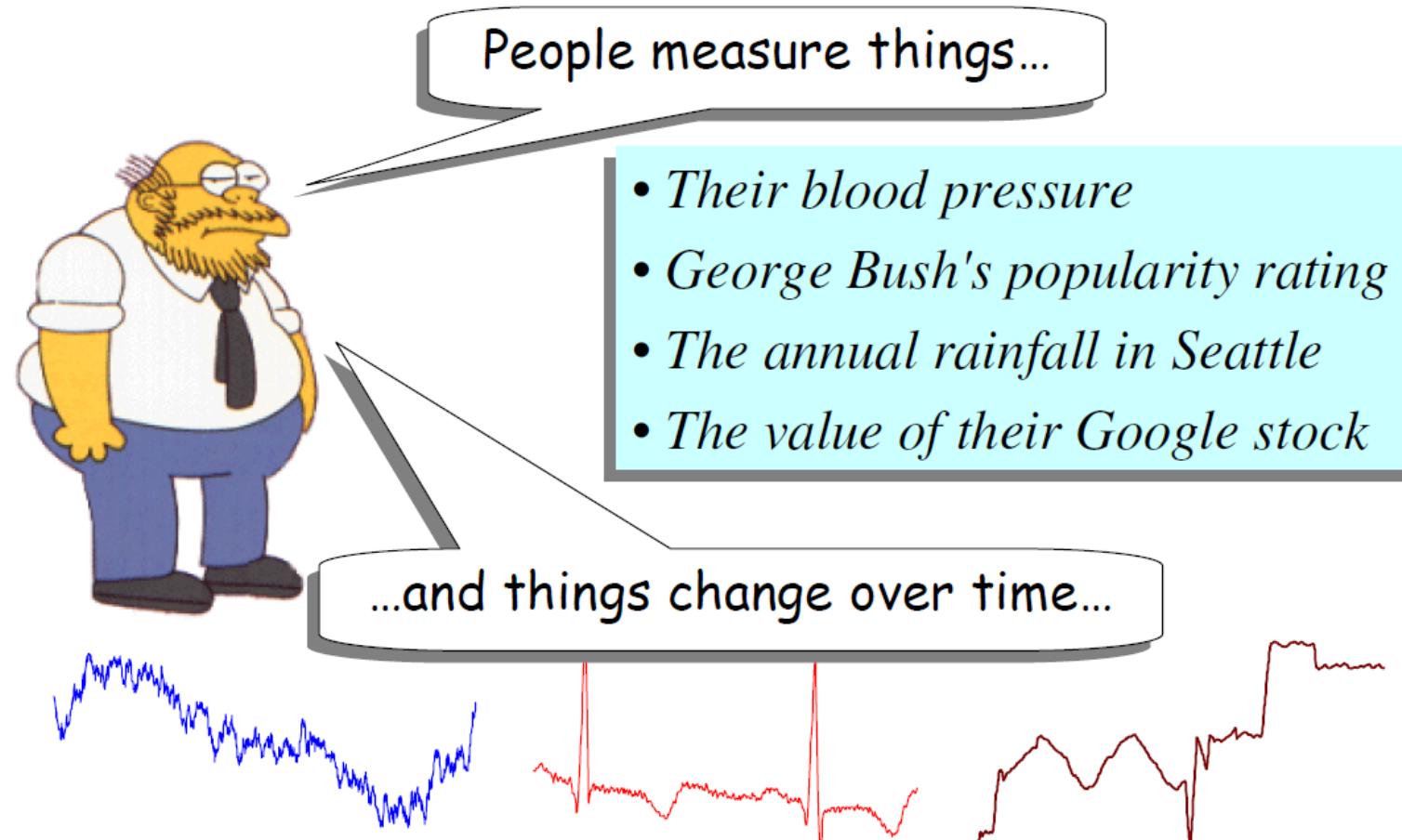
- Ubiquity of data series collections
- Time series data mining
- Similarity Search
- Metrics
- SAX : Time Series Symbolic Aggregaate approXimation.
- DTW Lower bounding
- Speed up computation by lower bounding true Euclidean distance over SAX representation.

Time series data... quick recap

- A **univariate time series** is a sequence of measurements of the same variable collected over time. Most often, the measurements are made at regular time intervals.



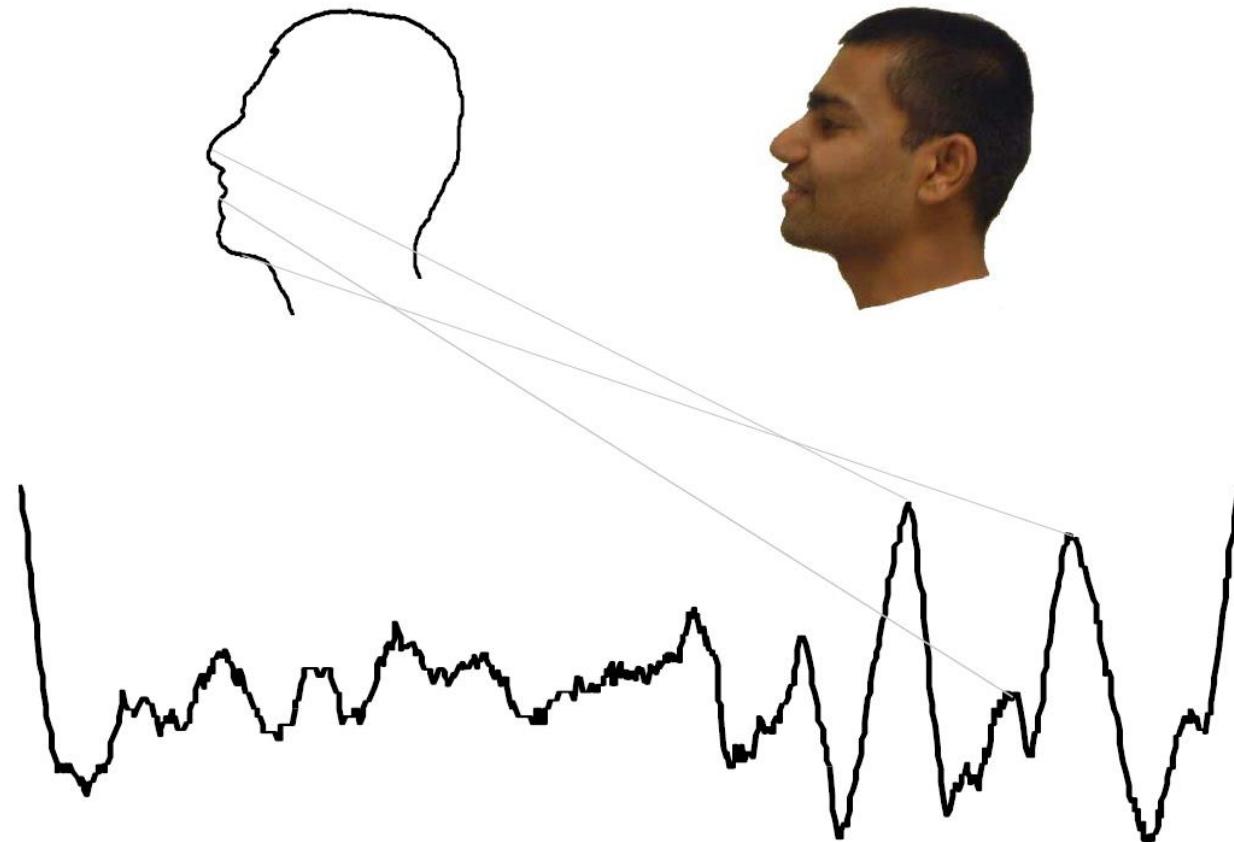
Time series are ubiquitous



Thus time series occur in virtually every medical, scientific and businesses domain

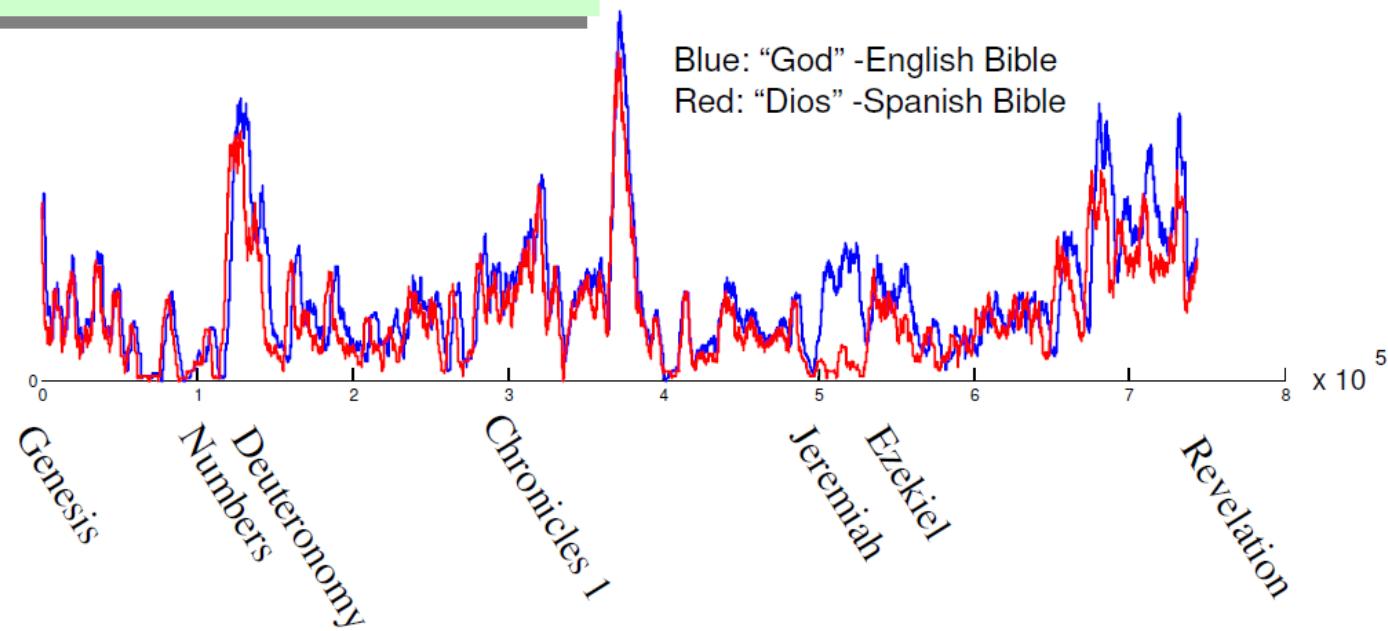
Time series are ubiquitous (1/4)

Image data, may best be thought of as time series...



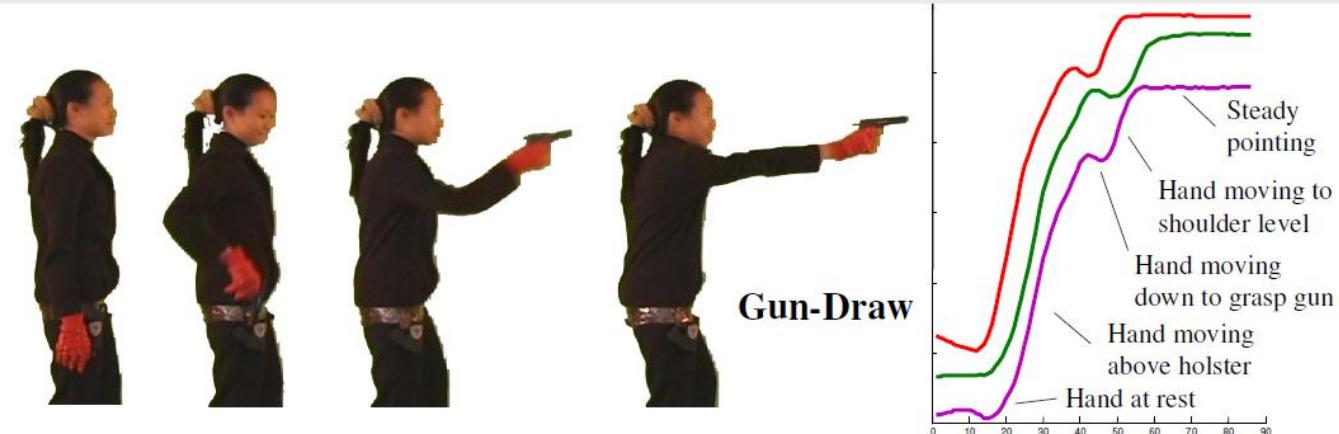
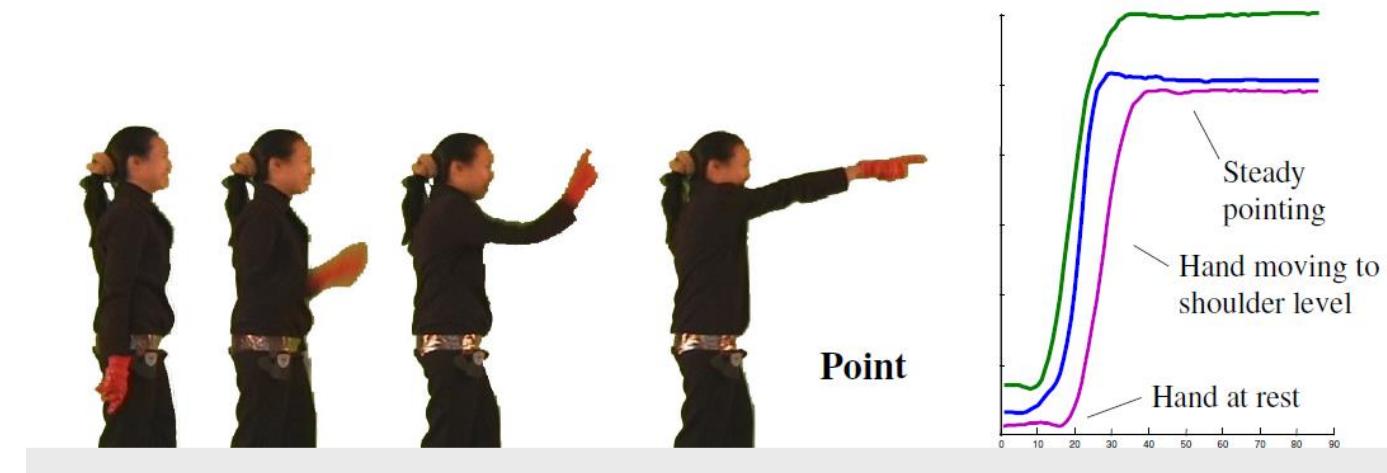
Time series are ubiquitous (2/4)

The local frequency
of words in the Bible



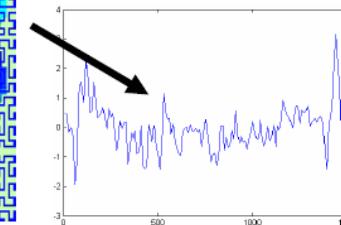
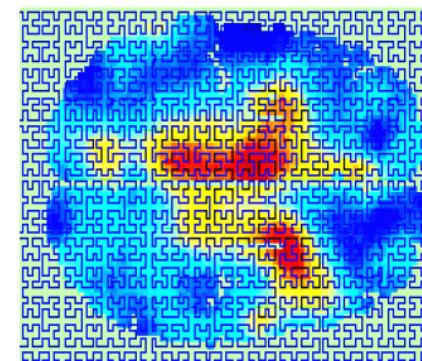
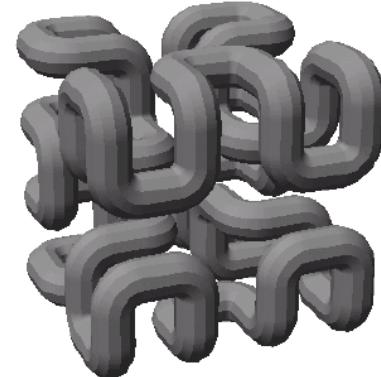
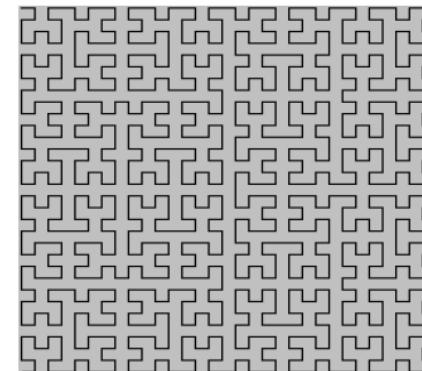
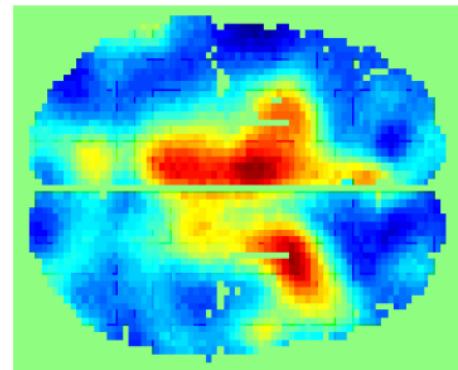
Time series are ubiquitous (3/4)

Video data, may best be thought of as time series...



Time series are ubiquitous (4/4)

Brain scans (3D voxels), may best be thought of as time series..



Why is Working With Time Series is so difficult? (1/3)

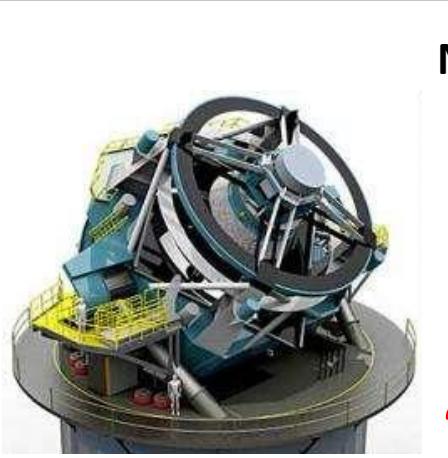
1 Hour of EKG data: 1 Gigabyte.

Typical Weblog: 5 Gigabytes per week.

Space Shuttle Database: 200 Gigabytes and growing.

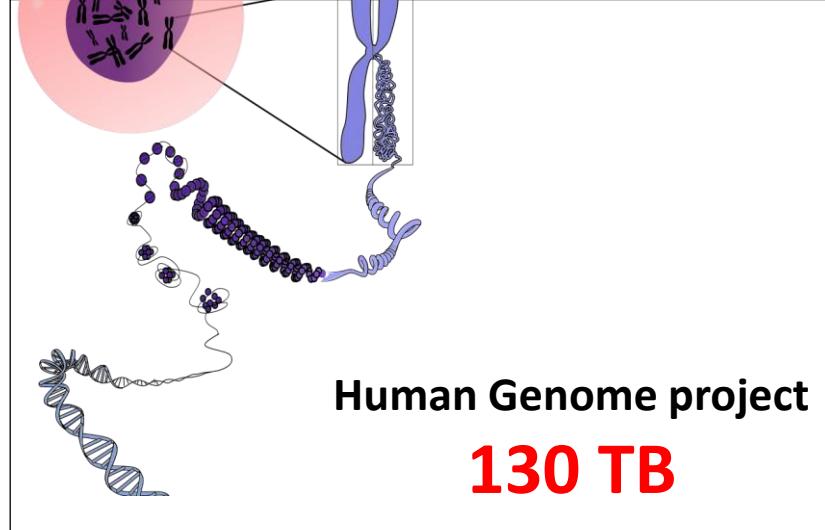
Macho Database (Canadian Astronomy Data Centre): 3 Terabytes, updated with 3 gigabytes a day.

Why is Working With Time Series is so difficult? (2/3)



NASA's Solar Observatory
1.5 TB per day

Large Synoptic Survey
Telescope (2019)
~30 TB per night



Human Genome project
130 TB



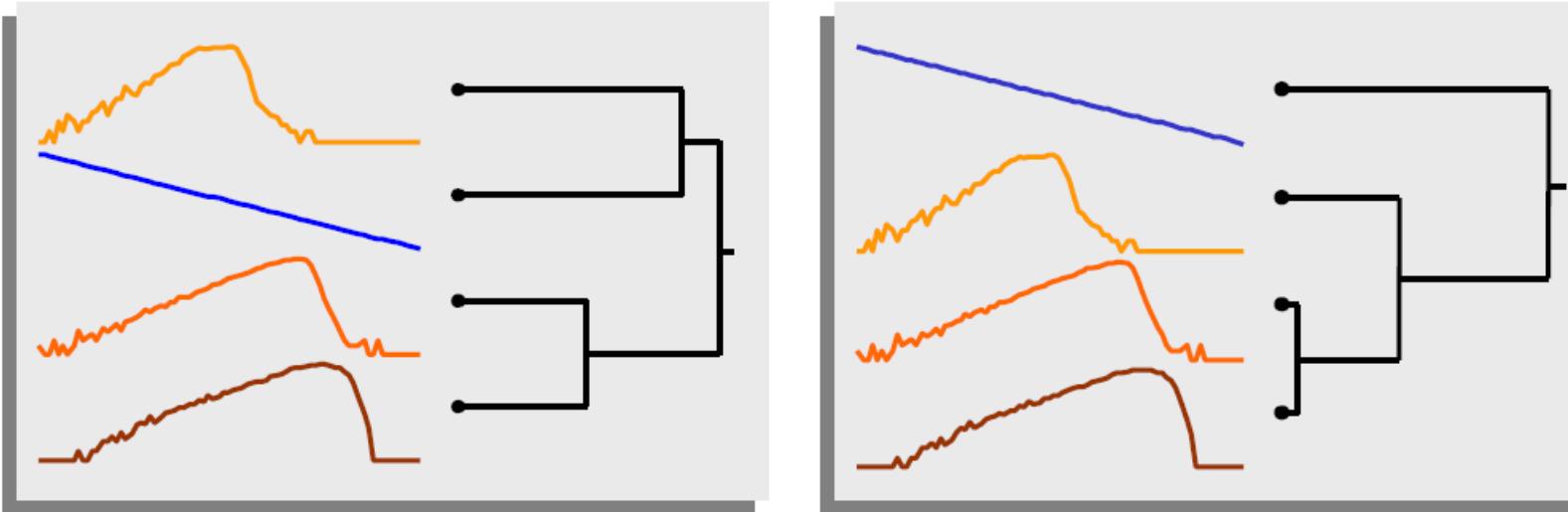
passenger aircrafts
20 TB per hour

data center and
services monitoring
2B data series
4M points/sec



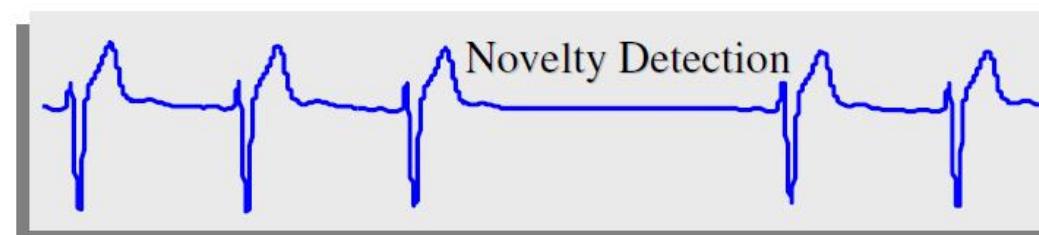
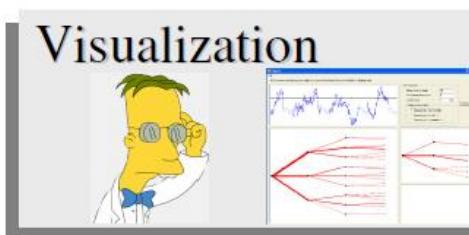
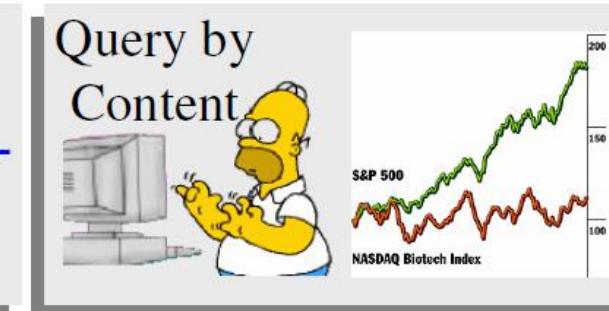
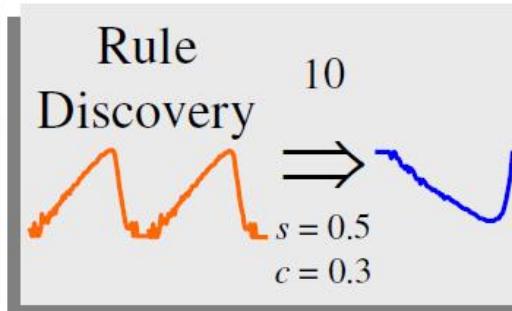
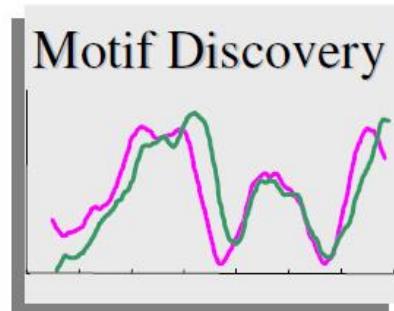
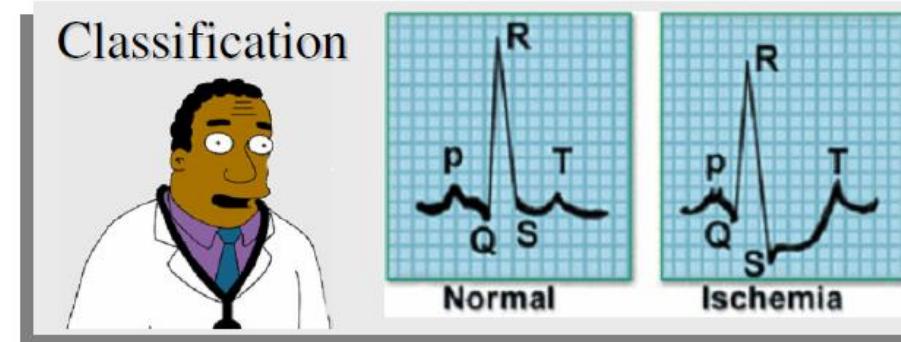
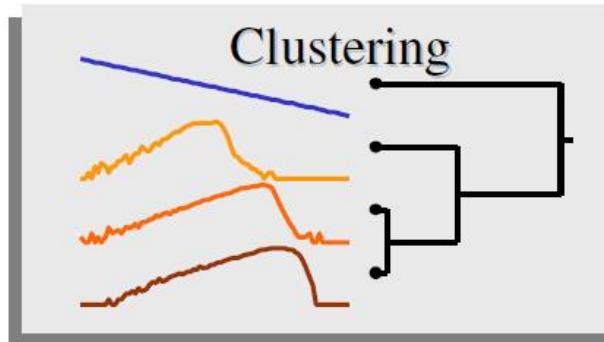
Why is Working With Time Series is so difficult? (3/3)

Answer: We are dealing with subjectivity



The definition of similarity depends on the user, the domain and the task at hand. We need to be able to handle this subjectivity.

Problems requiring Similarity Search



Important Data Mining questions



How do we define similarity ?



How do we search large time series collection quickly ?

What is similarity ?



What is similarity ?



The quality or state of being similar; likeness; resemblance; as, a similarity of features.

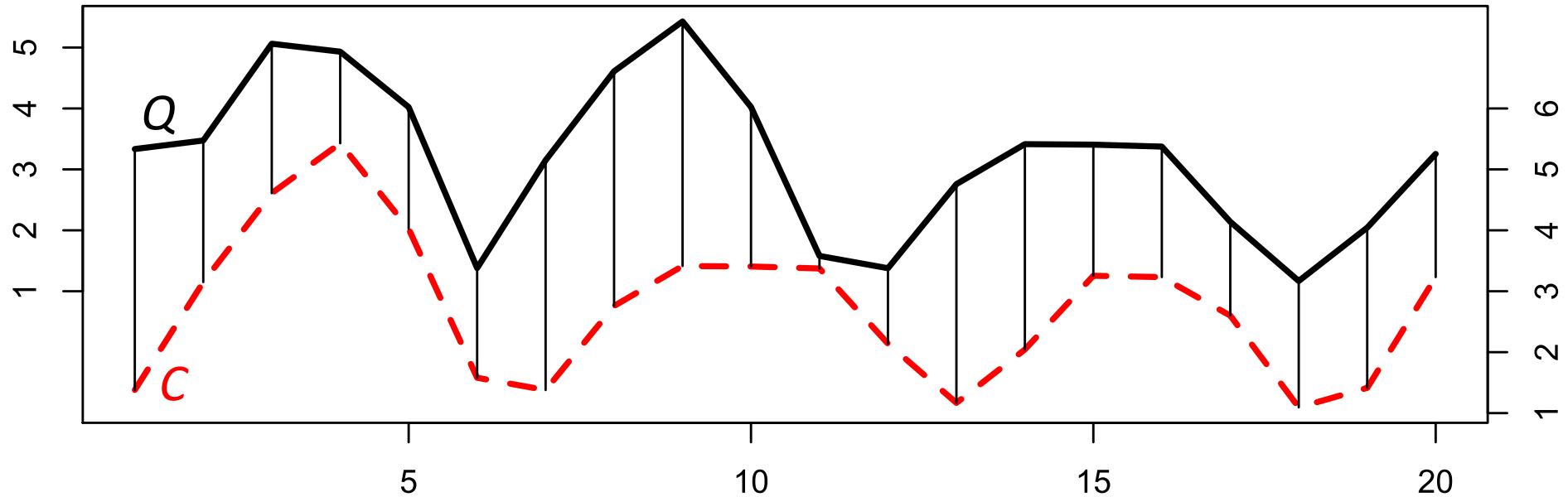


Similarity is hard to define, but...“We know it when we see it”



The real meaning of similarity is a philosophical question.

Similarity Measure: Euclidean



Given two time series $Q = q_1 \dots q_n$ and $C = c_1 \dots c_n$

their Euclidean distance is defined as:

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

Preprocessing the data before distance calculations

- Z-Normalization (Amplitude Scaling)
- Linear Trend
- Noise

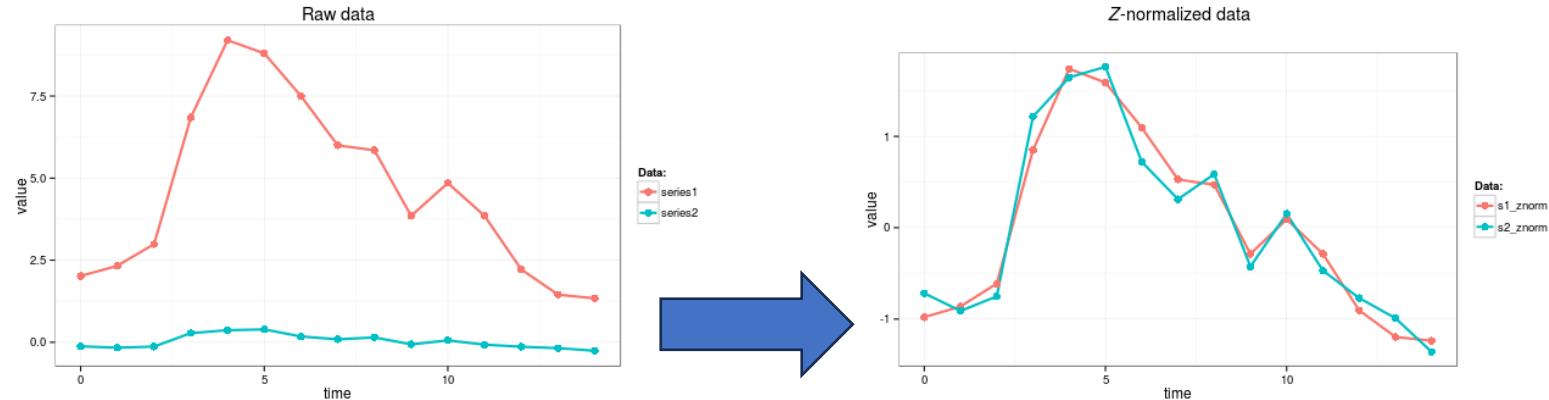
Z Normalization

Z-normalization, also known as

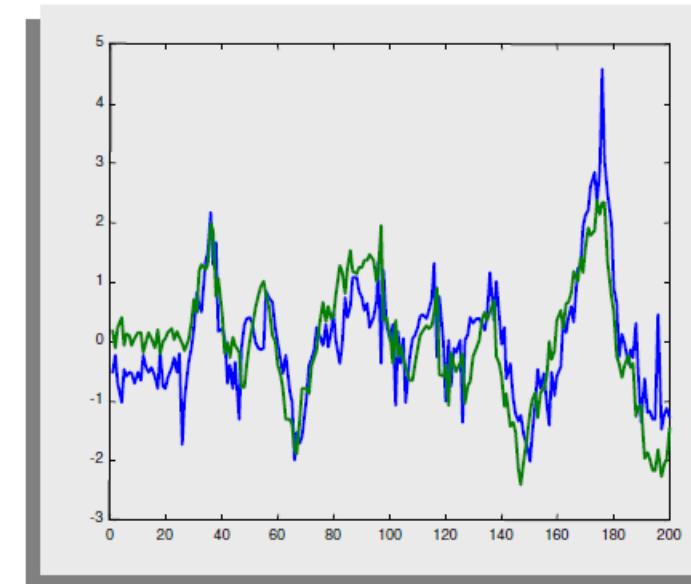
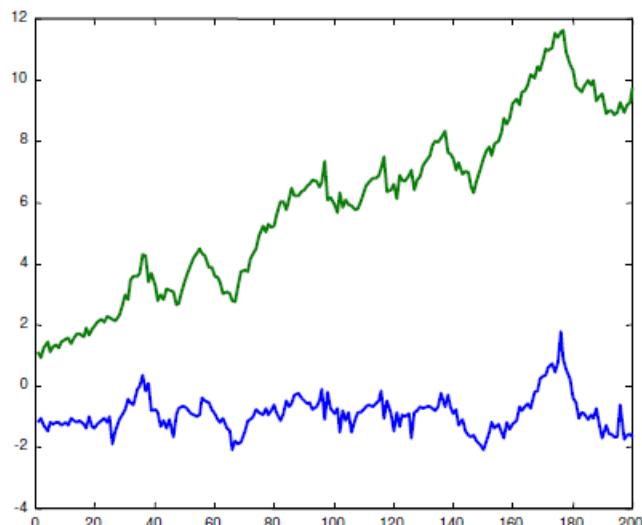
“Normalization to Zero Mean and Unit of Energy”.

The procedure ensures, that all elements of the input vector are transformed into the output vector whose mean is approximately 0 while the standard deviation is in a range close to 1.

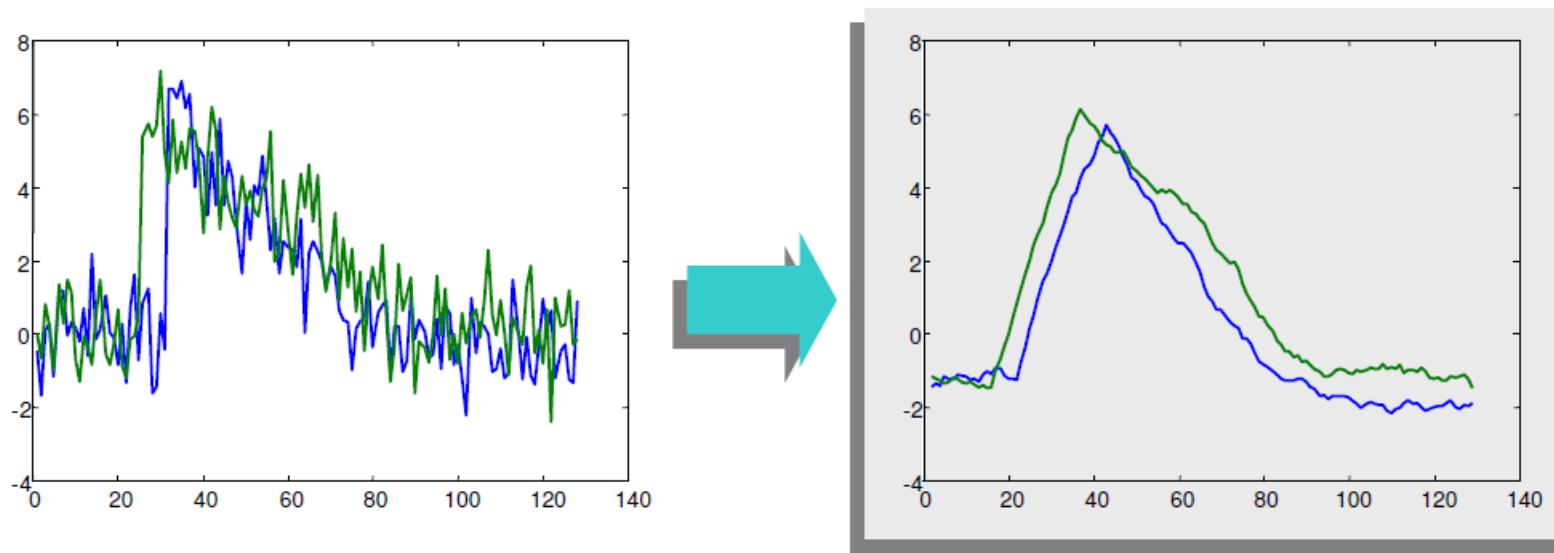
$$x'_i = \frac{x_i - \mu}{\sigma}, \text{ where } i \in \mathbb{N}$$



Remove linear trend



Removing Noise

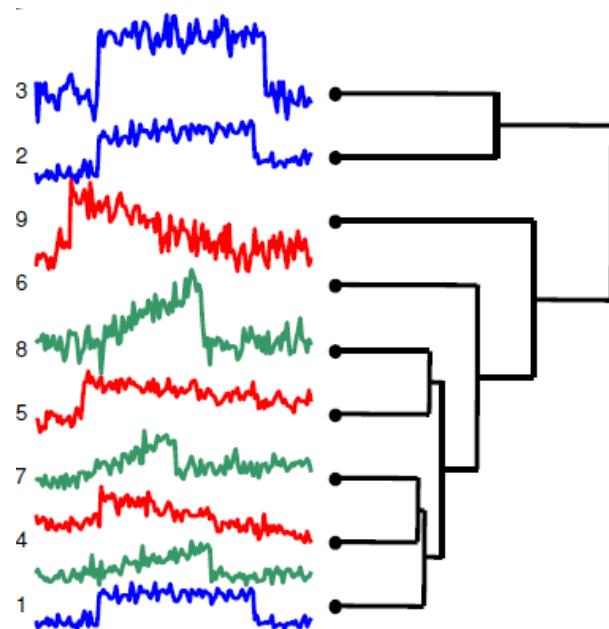


Smoothing function: Remove noise component

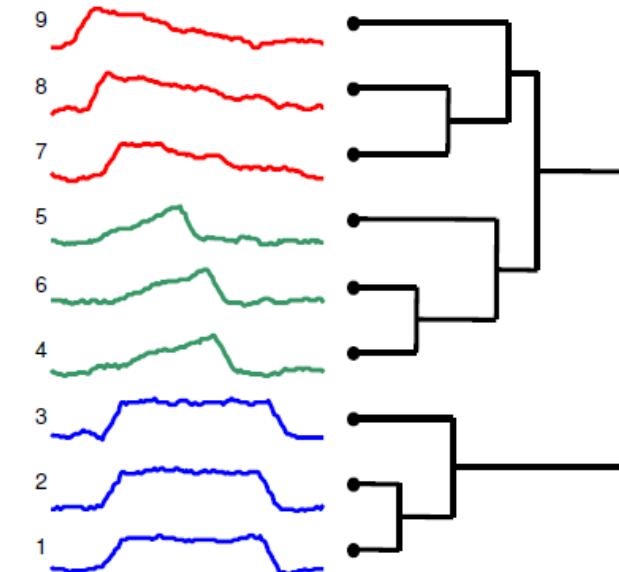
e.g., Average each datapoints value with its neighbors

Importance of data pre-processing

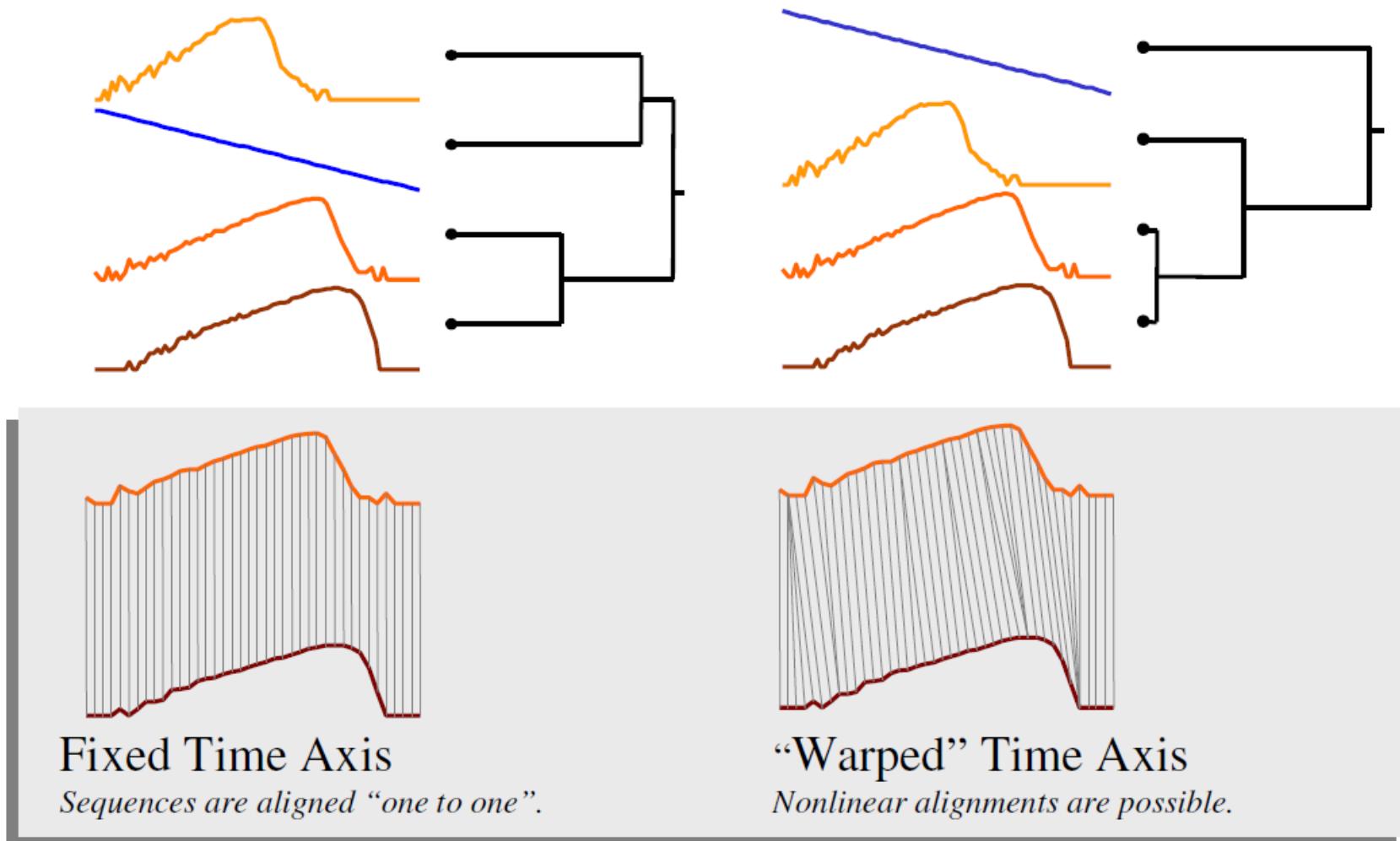
Clustered using Euclidean distance on the raw data.



Clustered using Euclidean distance, after removing noise, linear trend, and Z-Normalization

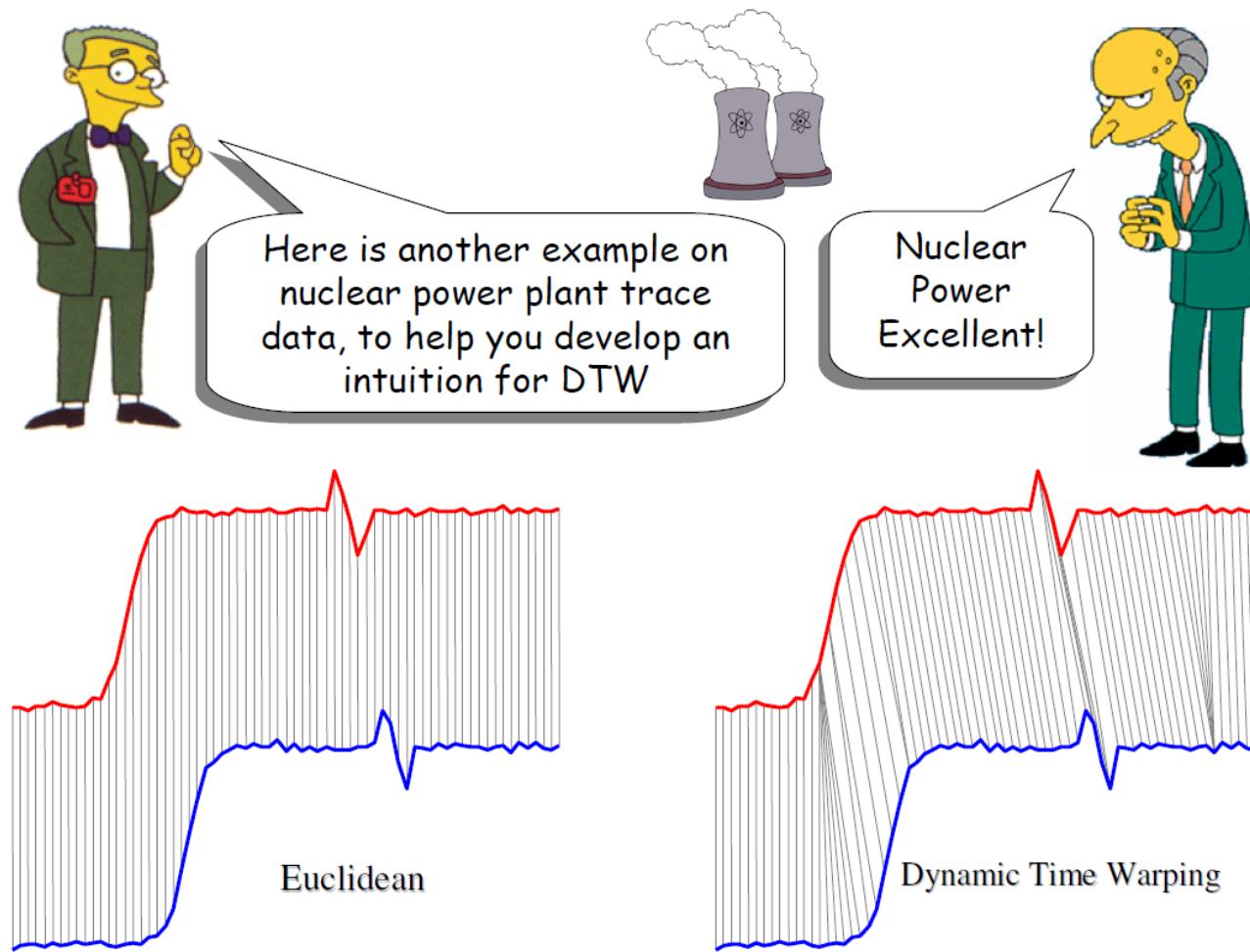


Dynamic Time Warping

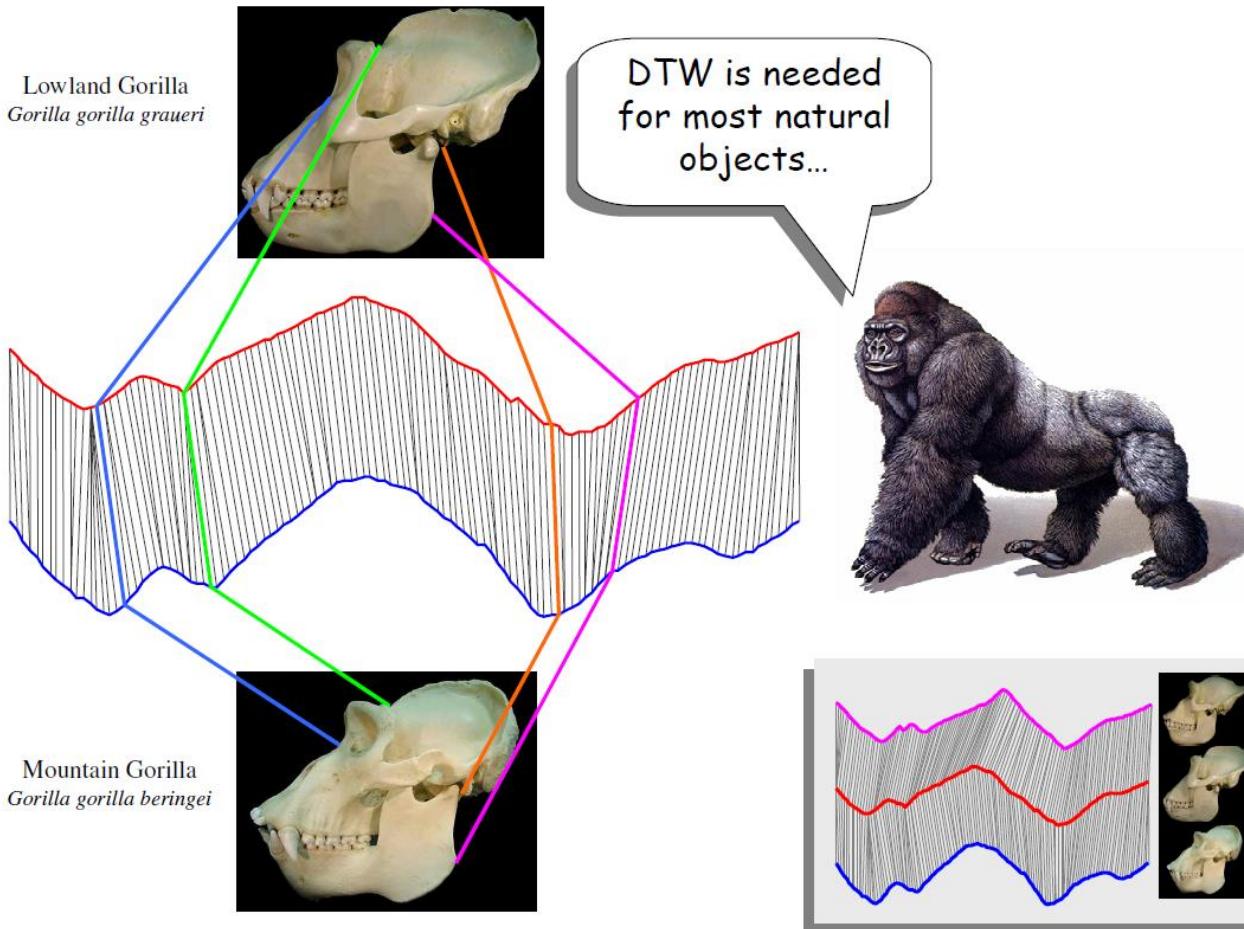


Note: We will first see the utility of DTW, then see how it is calculated.

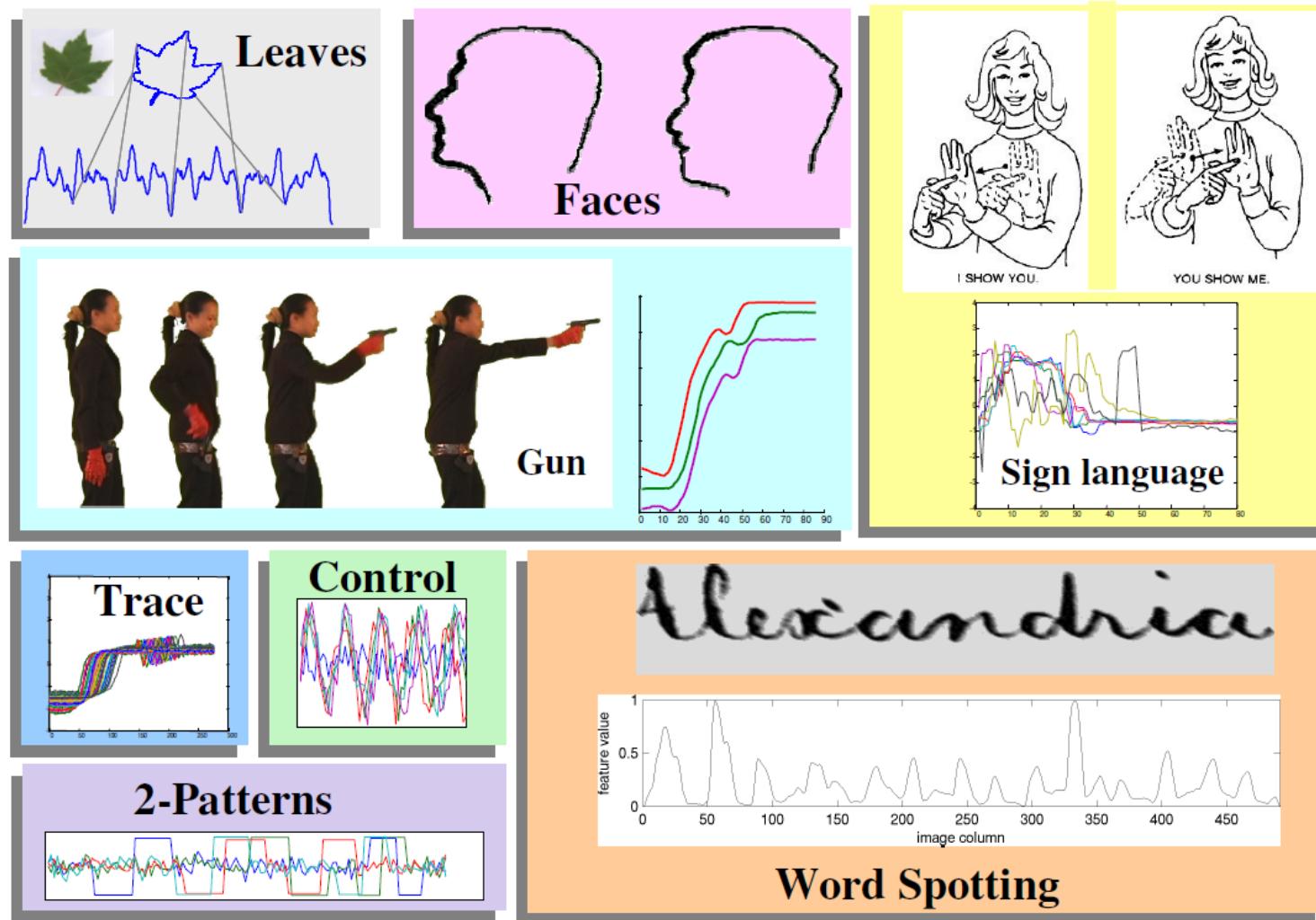
Some real-world example (1/3)



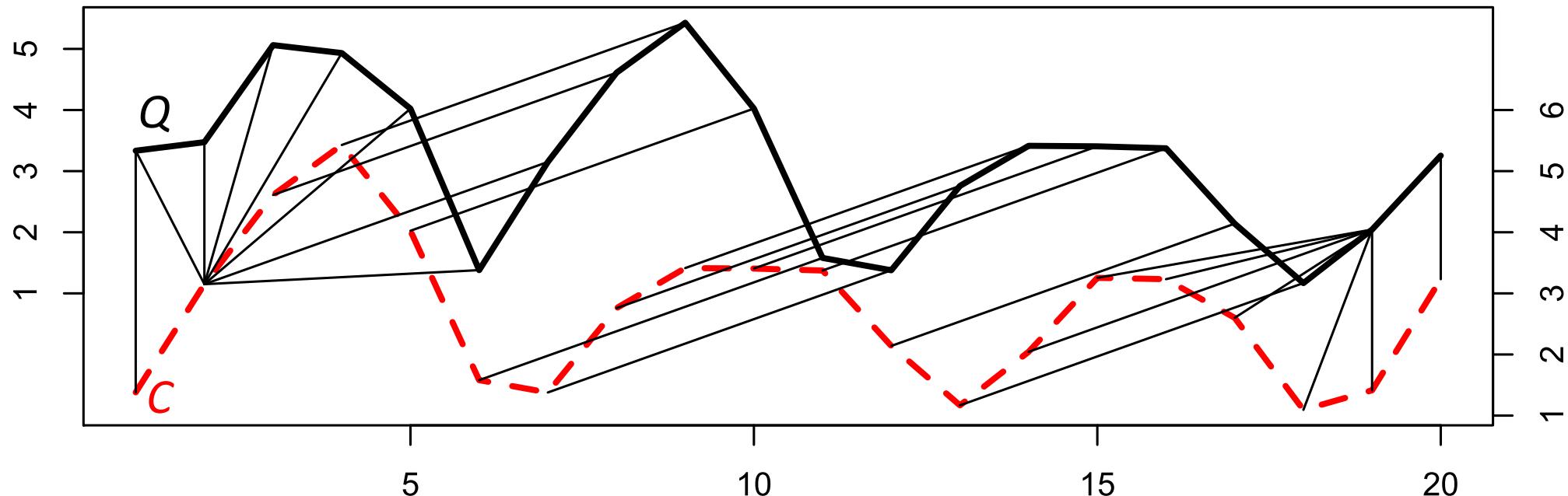
Some real-world example (2/3)



Some real-world example (3/3)



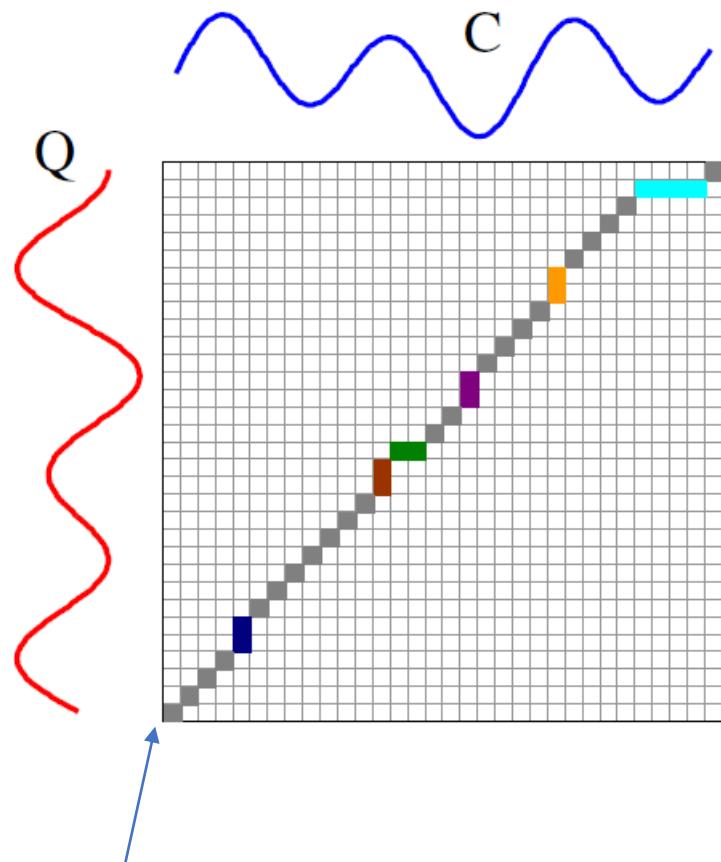
Distance Measure: Dynamic Time Warping (DTW)



Given two time series $Q = q_1 \dots q_n$ and $C = c_1 \dots c_n$

We want to find a pairwise points alignment $w \in (\mathbb{N} \times \mathbb{N})^n$, which minimize the pairwise points distance: $\text{DTW}(Q, C) = \operatorname{argmin}_P \left(\sum_{i=1}^{|P|} (q_{P_i[0]} - c_{P_i[1]})^2 \right)$

DTW calculation

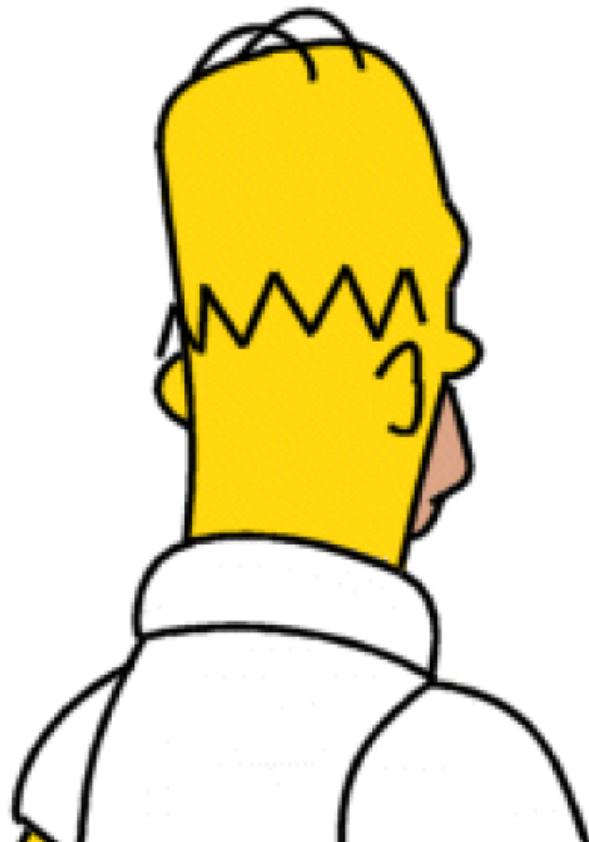


Recursive function of minimum cost path

$$\gamma(P_i) = \sqrt{(q_{P_i[0]} - c_{P_i[1]})^2 + \min\{\gamma(P_{i[0]} - 1), \gamma(P_{i[1]} - 1), \gamma(P_{i[0]} - 1, P_{i[1]} - 1)\}}$$

$P_0 = (0, 0)$ and $P_n = (n, n)$ are always the first and the last element of the warping path respectively.

Time complexity – $O(n^2)$

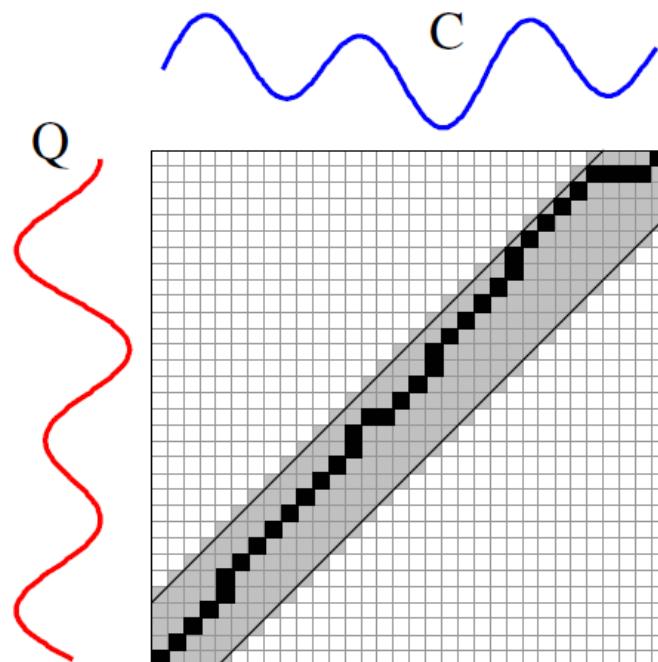


- Dynamic Time Warping gives **much better** results than Euclidean distance on virtually all problems.
- Dynamic Time Warping is very very slow to calculate!

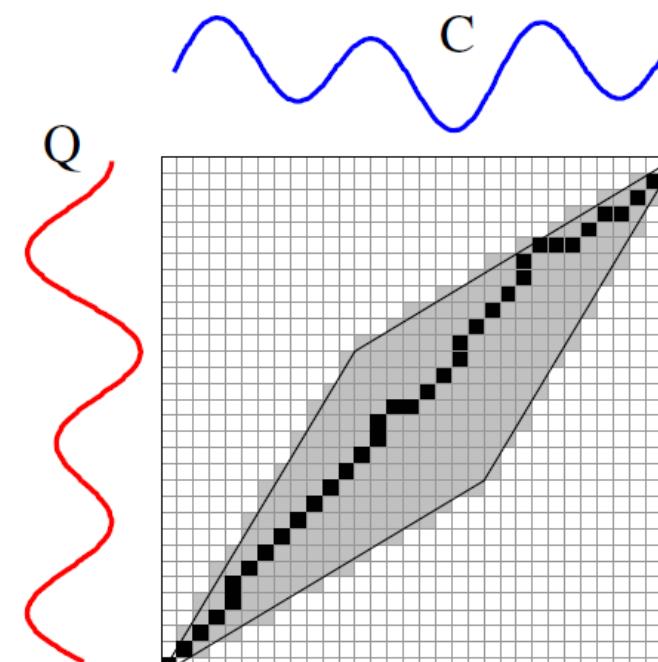
Is there anything we can do to speed up similarity search under DTW?

Global warping path constraints

- Slightly speed up the calculations
- Prevent pathological warpings

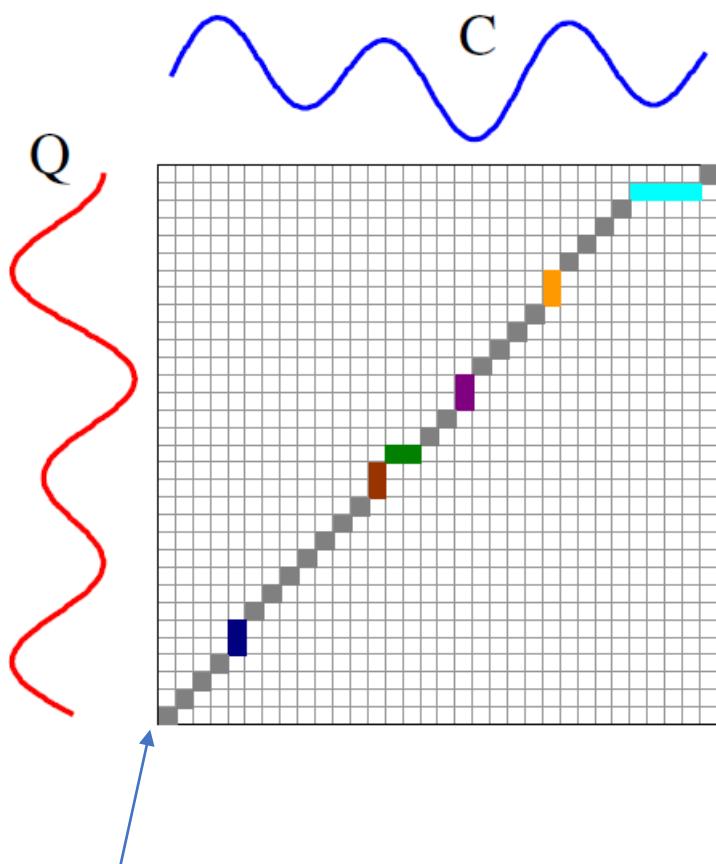


Sakoe-Chiba Band



Itakura Parallelogram

DTW calculation - pseudocode



Warping path $P \in (\mathbb{N} \times \mathbb{N})^n$

```
int DTWDistance(Q: array [1..n], C: array [1..m], warConstr:  
int) {  
    DTW := array [0..n, 0..m]  
  
    warConstr := max(warConstr, abs(n-m)) // adapt window size  
  
    for i := 0 to n  
        for j:= 0 to m  
            DTW[i, j] := infinity  
  
    DTW[0, 0] := 0  
    for i := 1 to n  
        for j := max(1, i-warConstr) to min(m, i+warConstr)  
            DTW[i, j] := 0  
  
    for i := 1 to n  
        for j := max(1, i-warConstr) to min(m, i+warConstr)  
            cost := d(Q[i], C[j])  
            DTW[i, j] := cost + minimum(DTW[i-1, j ],  
                                         DTW[i , j-1],  
                                         DTW[i-1, j-1])  
    return DTW[n, m]  
}
```

Lower bounding

Assume that we have two functions:

- $\text{DTW}(A, B)$
- $\text{lower_bound_distance}(A, B)$

The true DTW
function is very
slow...

The *lower
bound* function
is very fast...

By definition, for all A, B , we have

$$\text{lower_bound_distance}(A, B) \leq \text{DTW}(A, B)$$

Speed up search with lower bounding

We can speed up similarity search under DTW
by using a lower bounding function

```
Algorithm Lower_Bounding_Sequential_Scan(Q)
1. best_so_far = infinity;
2. for all sequences in database
3.   LB_dist = lower_bound_distance(Ci, Q);
4.   if LB_dist < best_so_far
5.     true_dist = DTW(Ci, Q);
6.     if true_dist < best_so_far
7.       best_so_far = true_dist;
8.       index_of_best_match = i;
9.     endif
10.    endif
11.  endfor
```

Try to use a cheap
lower bounding
calculation as
often as possible.

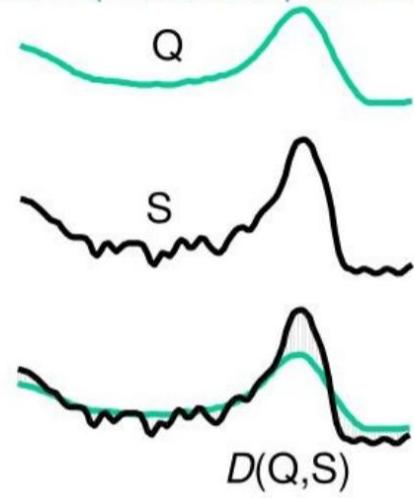


Only do the
expensive, full
calculations when
it is absolutely
necessary



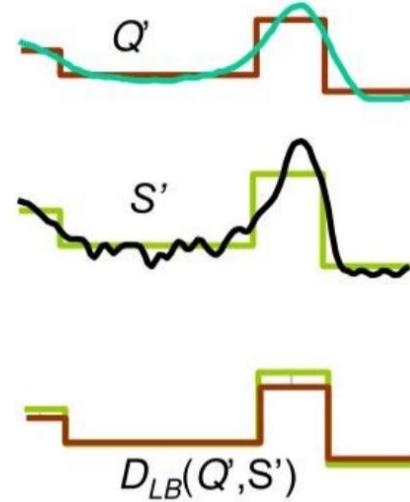
Lower Bounding Euclidean distance

Exact (Euclidean) distance $D(Q, S)$



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

Lower bounding distance $D_{LB}(Q, S)$

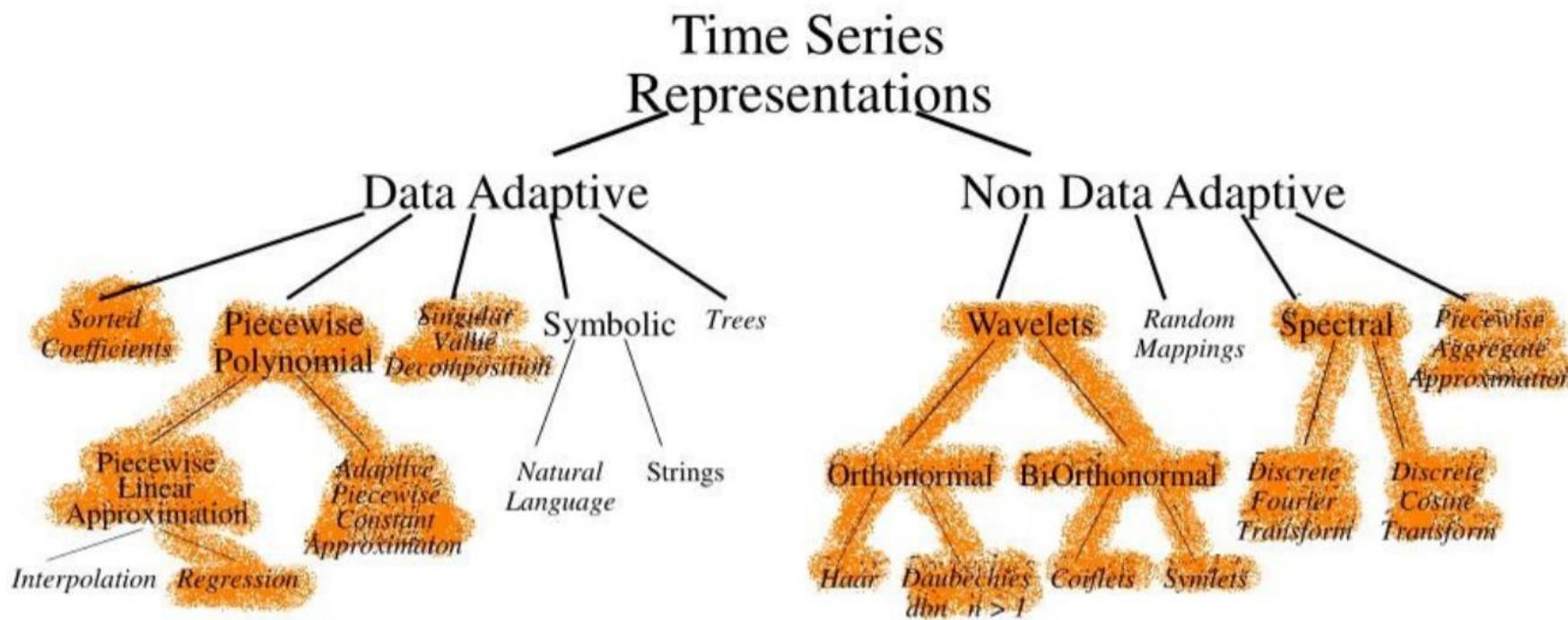


$$D_{LB}(Q', S') \equiv \sqrt{\sum_{i=1}^M (sr_i - sr_{i-1})(qv_i - sv_i)^2}$$

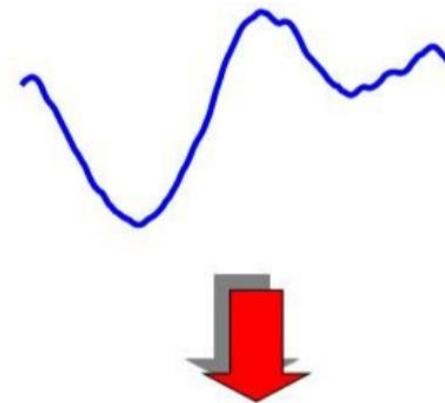
Lower bounding means that for all Q and S , we have...

$$D_{LB}(Q', S') \leq D(Q, S)$$

Lower bounding over discrete TS representation



Symbolic Aggregate approXimation (SAX)



baabccbc

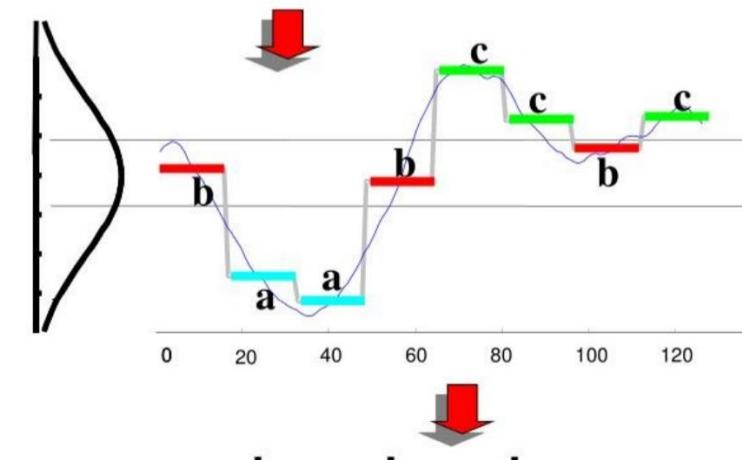
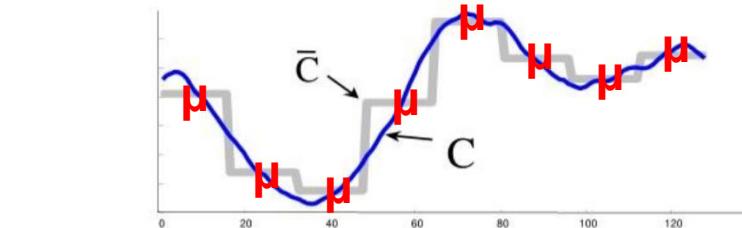
Symbolic Aggregate approXimation (SAX)



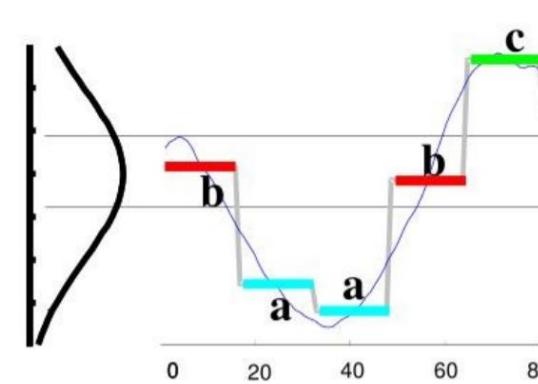
How do we obtain SAX?

First convert the time series to PAA representation, then convert the PAA to symbols

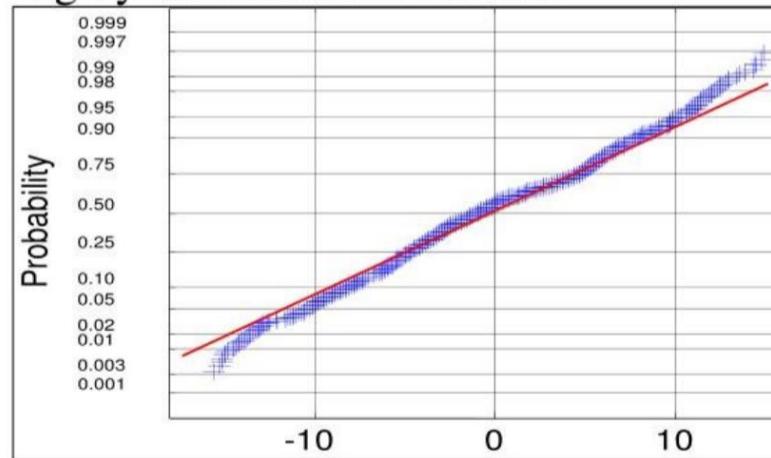
It take linear time



Symbolic Aggregate approXimation (SAX)



Time series subsequences tend to have a highly Gaussian distribution

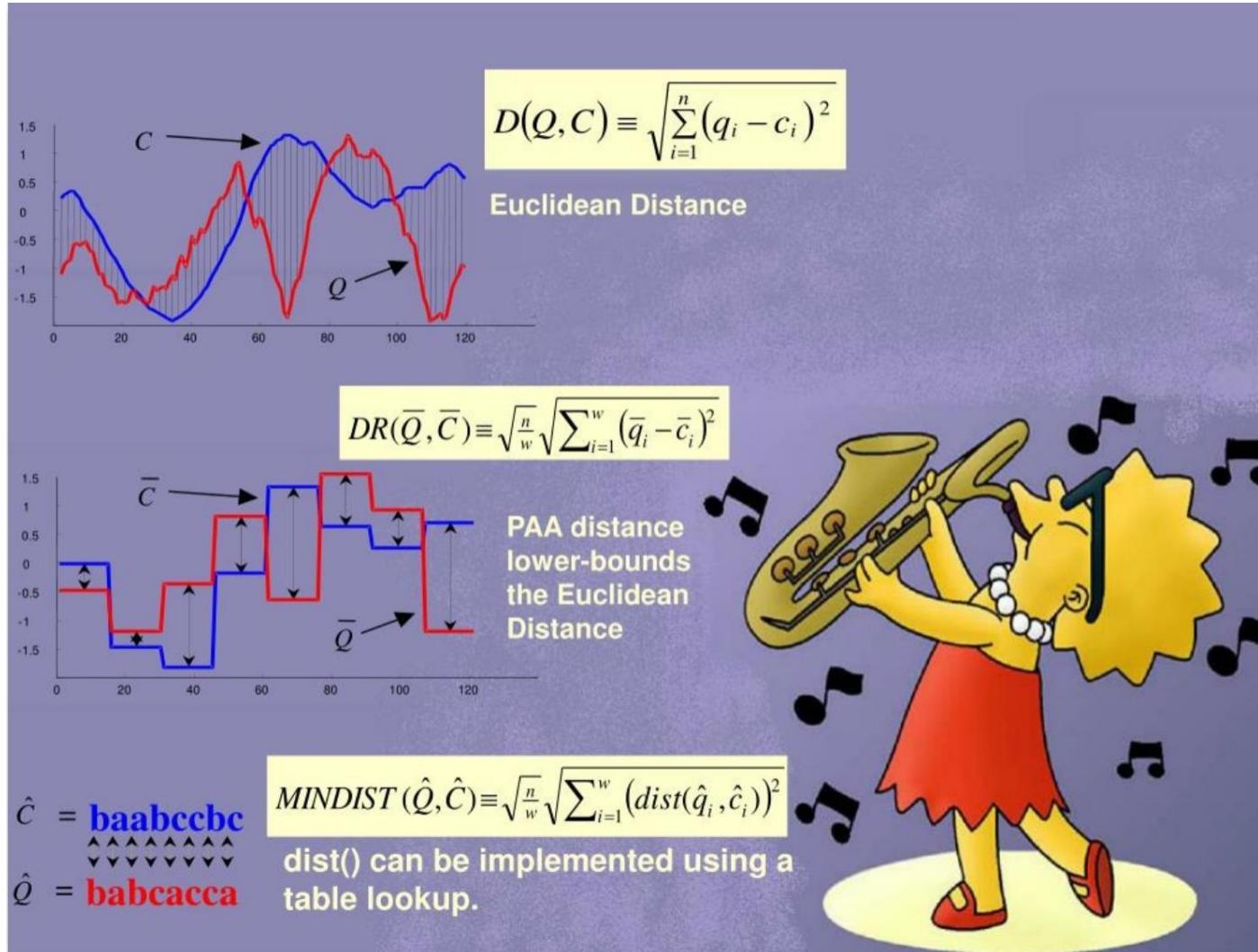


A normal probability plot of the (cumulative) distribution of values from subsequences of length 128.

Why a Gaussian?



SAX lowerbounding (MINDIST)



Implementation (PYTS library)



A Python Package for Time Series Classification

Star 1,615

Navigation

[Getting Started](#)

[Installation, testing and development](#)

[Contributing guide](#)

Documentation

[User guide](#)

[API Documentation](#)

[Scikit-learn compatibility](#)

Tutorial - Examples

[Introductory examples](#)

[Approximating time series](#)

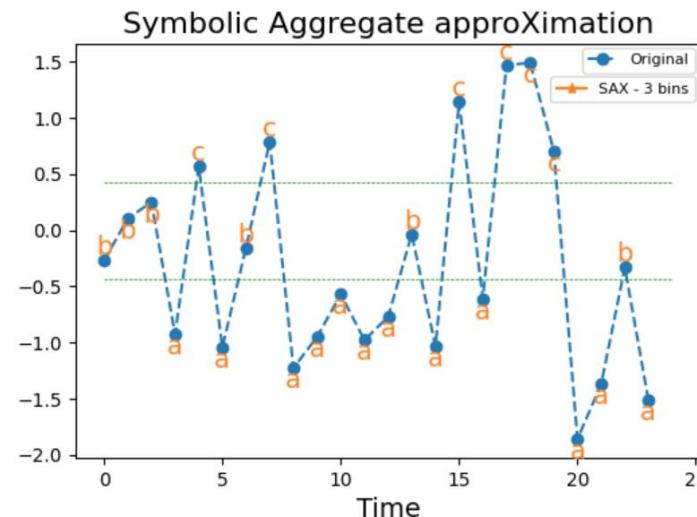
[Bag-of-words transformation](#)

[Classification algorithms](#)

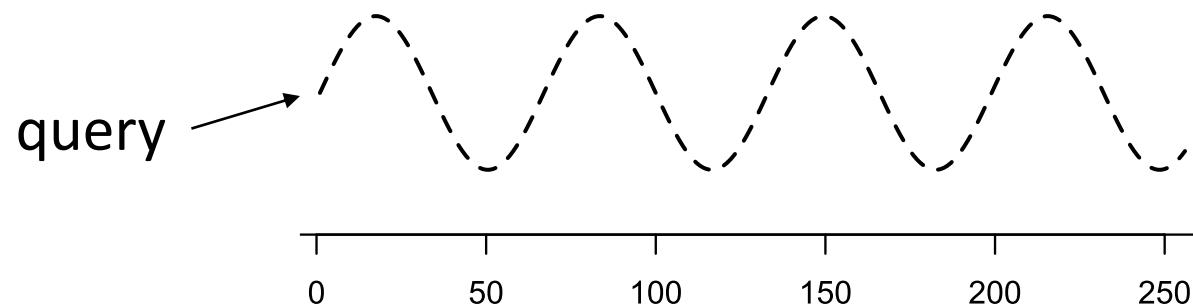
Note: Click here to download the full example code

Symbolic Aggregate approXimation

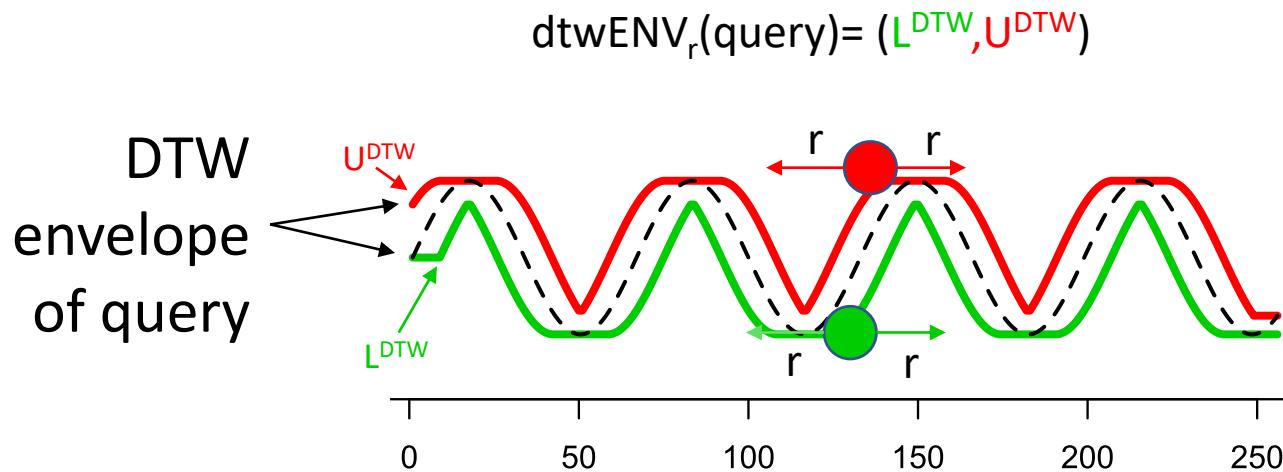
Binning continuous data into intervals can be seen as an approximation that reduces noise and captures the trend of a time series. The Symbolic Aggregate approXimation (SAX) algorithm bins continuous time series into intervals, transforming independently each time series (a sequence of floats) into a sequence of symbols, usually letters. This example illustrates the transformation. It is implemented as `pyts.approximation.SymbolicAggregateApproximation`.



Lower bounding DTW

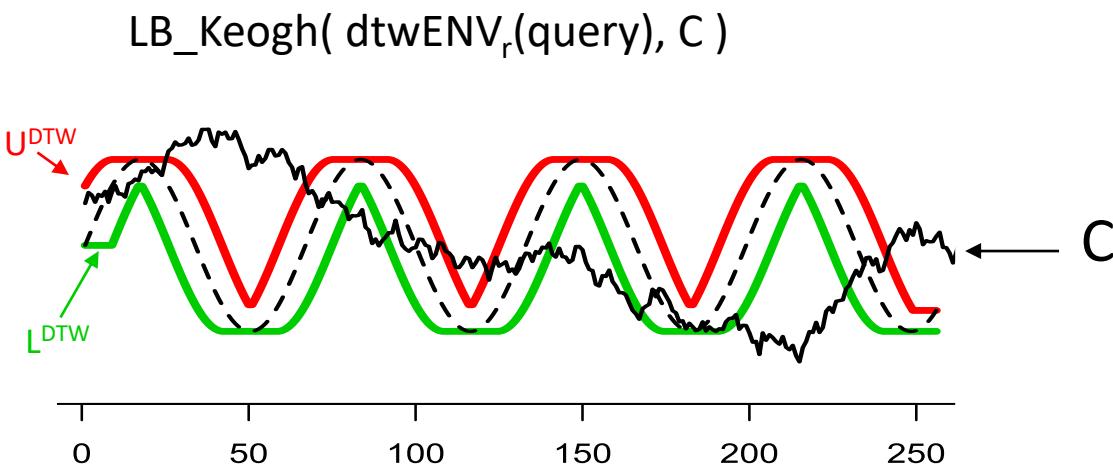


Lower bounding DTW



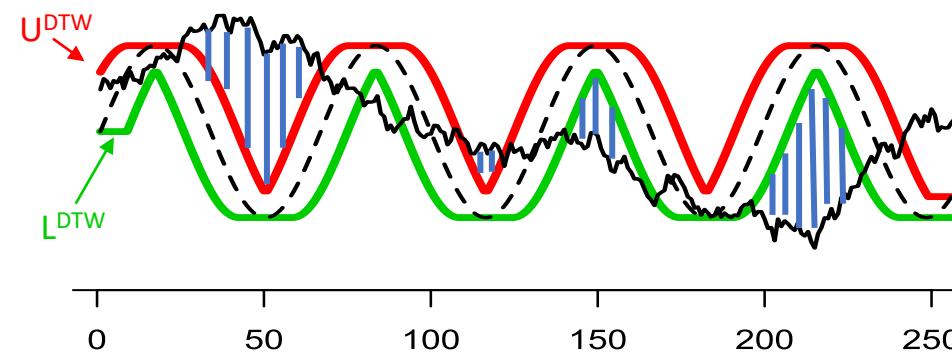
- For each query position, dtwENV bounds the values that can be aligned to a single point in the data series candidates (**warping window r**).

Lower bounding DTW



Lower bounding DTW

$\text{LB_Keogh(dtwENV}_r(\text{query}), \text{candidate})$



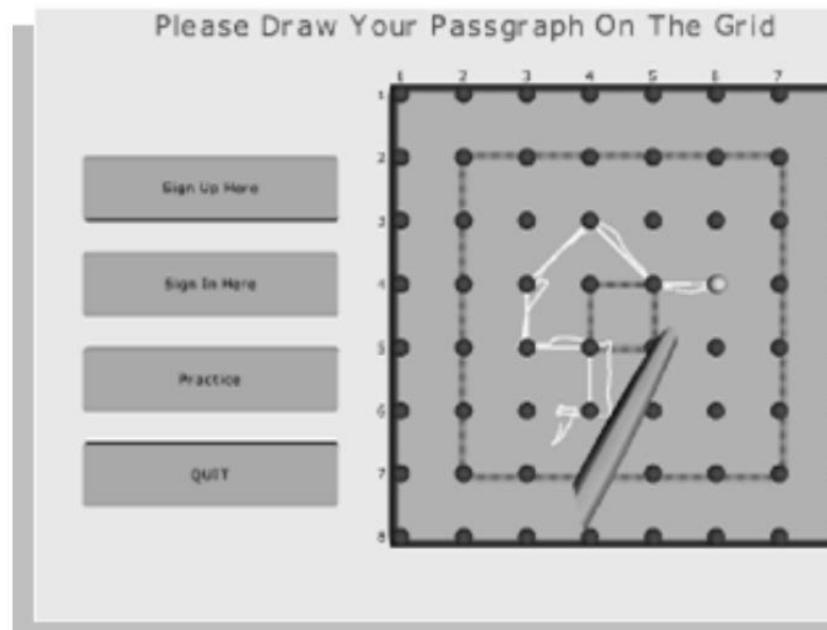
- Lower bounding of the true DTW distance between Query and Candidate
[$O(n)$ time]

A black and white aerial photograph showing a large agricultural field. The field is divided into several rectangular plots by a network of dirt roads. Within these plots, there are intricate, light-colored geometric patterns resembling crop circles or ancient markings. The patterns include straight lines, circles, and more complex shapes like a four-pointed star. The overall scene is vast and repetitive.

Presentation du TD

Haptics DATA

Data are taken from 5 people entering their passgraph (a code to access a system protected by a graphical authentication system) on a touchscreen. The data are the x-axis movement only.



Notebook

- Open the file (Python Notebook):

TS_SimilaritySearch.ipynb

- Instruction are contained in the notebook
- You can look at the data source to find similarity search hyperparameter :

[https://www.cs.ucr.edu/%7Eeamonn/time series data 2018/](https://www.cs.ucr.edu/%7Eeamonn/time_series_data_2018/)

References

- Eamonn J. Keogh. A Decade of Progress in Indexing and Mining Large Time Series Databases. VLDB 2006: 1268
- Eamonn J. Keogh, Li Wei, Xiaopeng Xi, Sang-Hee Lee, Michail Vlachos: **LB_Keogh Supports Exact Indexing of Shapes under Rotation Invariance with Arbitrary Representations and Distance Measures.** VLDB 2006: 882-893
- Chotirat Ann Ralanamahatana, Jessica Lin, Dimitrios Gunopoulos, Eamonn Keogh, Michail Vlachos & Gautam Das **Mining Time Series Data**
https://link.springer.com/chapter/10.1007/0-387-25465-x_51#Abs1
- <https://tslearn.readthedocs.io/> (tslearn package documentation)