Data Warehousing and Data Mining: Time Series Analytics

Patrick Schäfer

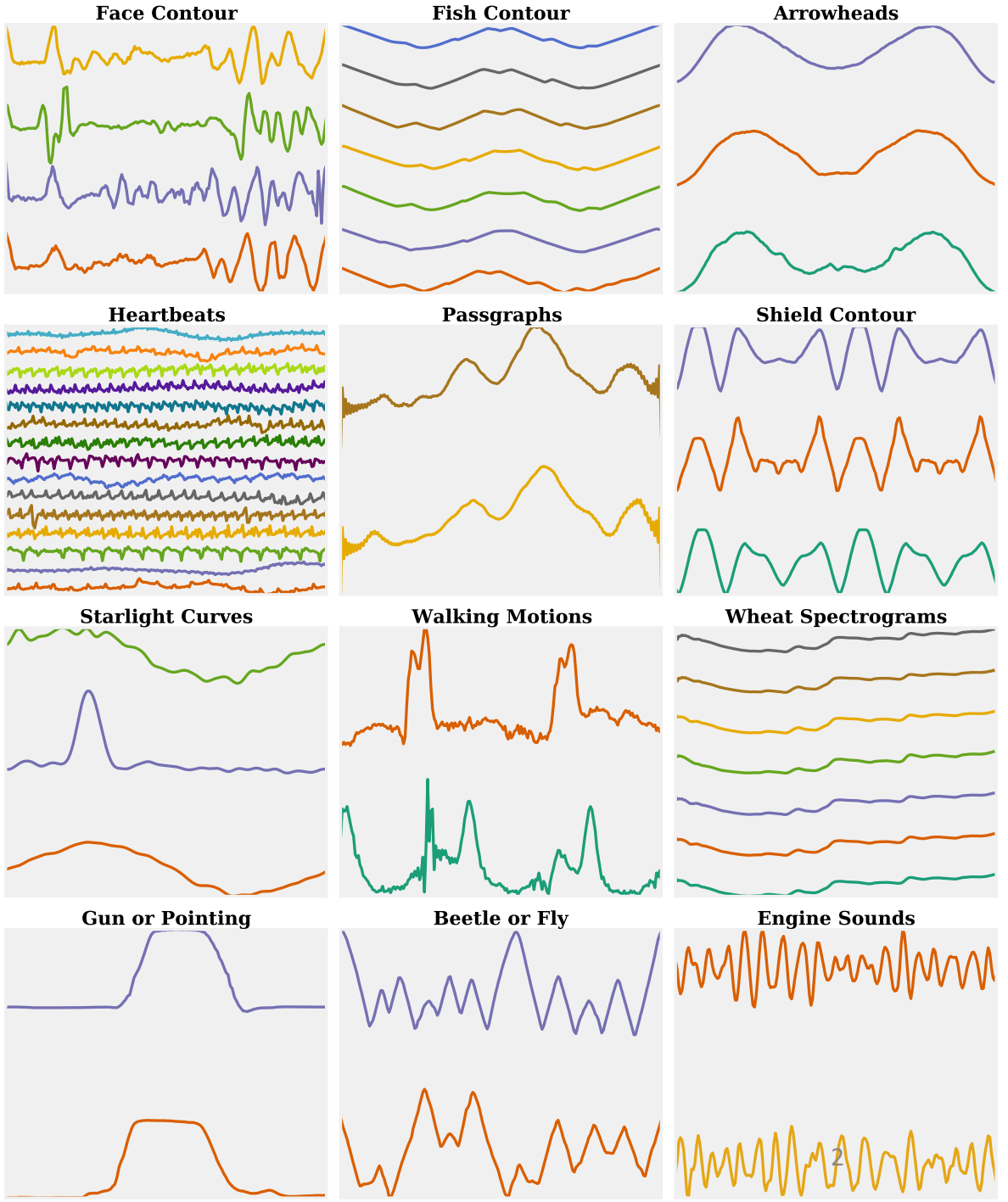
Berlin, February 5, 2018

Vorlesung: https://hu.berlin/vl_dwhdm17
Übung: https://hu.berlin/ue_dwhdm17

Humboldt-Universität
Informatik

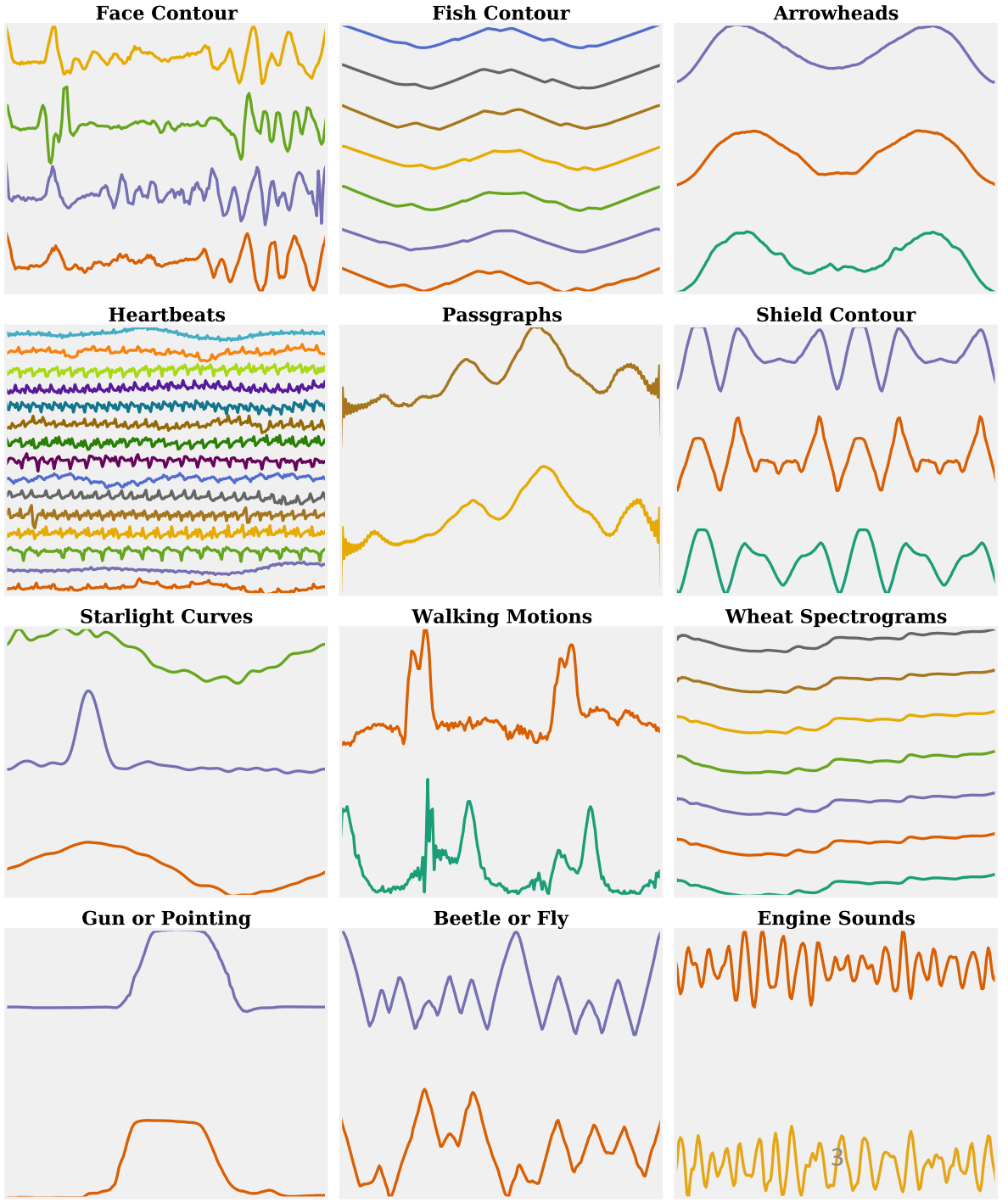
Motivation

- Temporal data is common in many data mining applications and data warehouses
  - There is a time dimension in all M/ER models
- Application domains range from:
  - Sensor data: environmental sensors measure temperature, pressure humidity
  - Medical devices: electrocardiogram (ECG) and electroencephalogram (EEG)
  - Financial market: stock prices, economic indicators, product sales
  - Meteorological data: sediments from drill holes, earth observation satellite data



Motivation

- Data analytics based on *similarity* is the tool for exploring these datasets
- Examples of similarity queries include
  - Find all stocks that show *similar* trends
  - Find the *most unusual* heartbeat in a patient's ECG recording
  - Find *frequent patterns* in a bird sound recording
  - Find the patients with the most *similar* ECG recording
  - Find products with a *similar* sales pattern



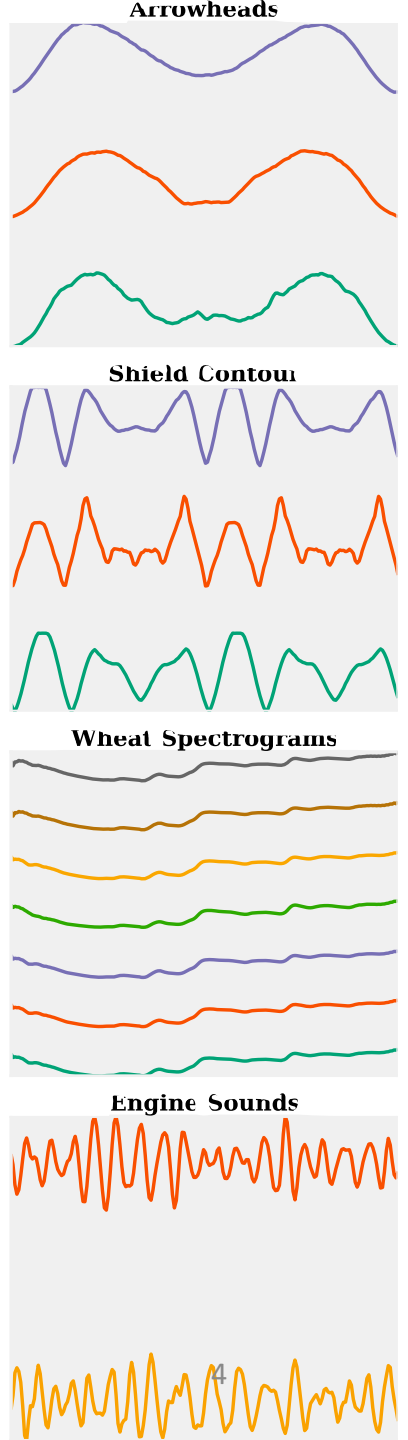
Time Series Definition

- Definition: A *Time Series* is a sequence (ordered collection) of n real values at time stamps $(\boldsymbol{t_1}, \ldots, \boldsymbol{t_n})$:

$$\boldsymbol{T} = (\boldsymbol{y_1}, \ldots, \boldsymbol{y_n})$$

- Time Series may be *univariate* or *multivariate*
  - Univariate: a single value y_i is associated with each time stamp t_i.
  - Multivariate: m values $y_i = (k_1, \ldots k_m)$ are associated with each time stamp t_i.

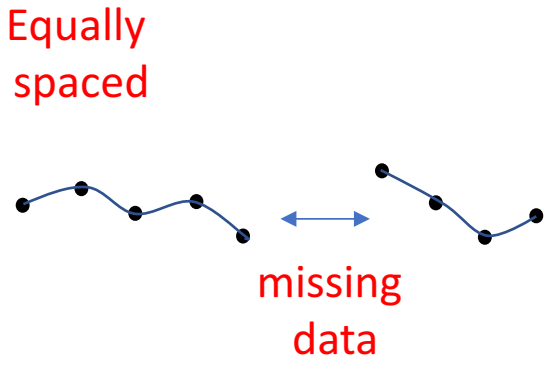
- The dimensionality of a time series refers to the number of values at each time stamp



Time Series vs Sequence Data

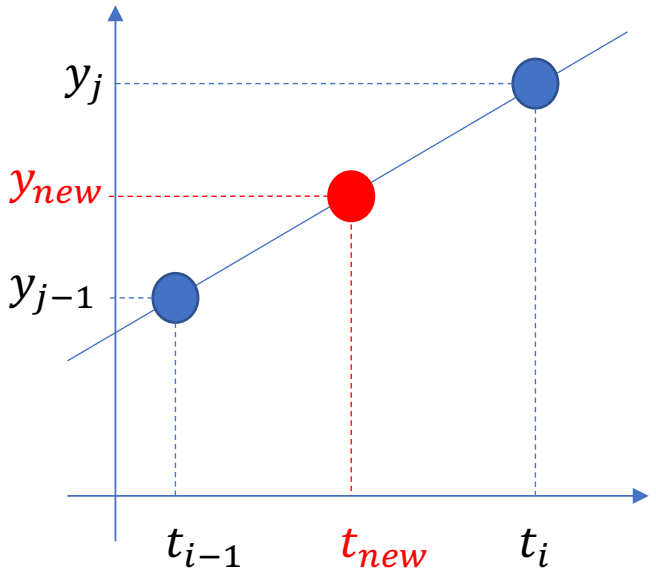
- Temporal data may be *discrete* or real-valued (continuous)
- *Time series* contain real values and *sequence* data (i.e. web click streams, DNA sequences, documents) refers to discrete data
- Different algorithms applicable for sequence data: Hashing, Tries, Naïve Bayes, Boyer-Moore, vector space model, tf-idf, word embeddings…
- In some cases a times series can be converted to a sequence by use of discretization, thereby make use of sequence mining algorithms

Pre-processing: Missing Data



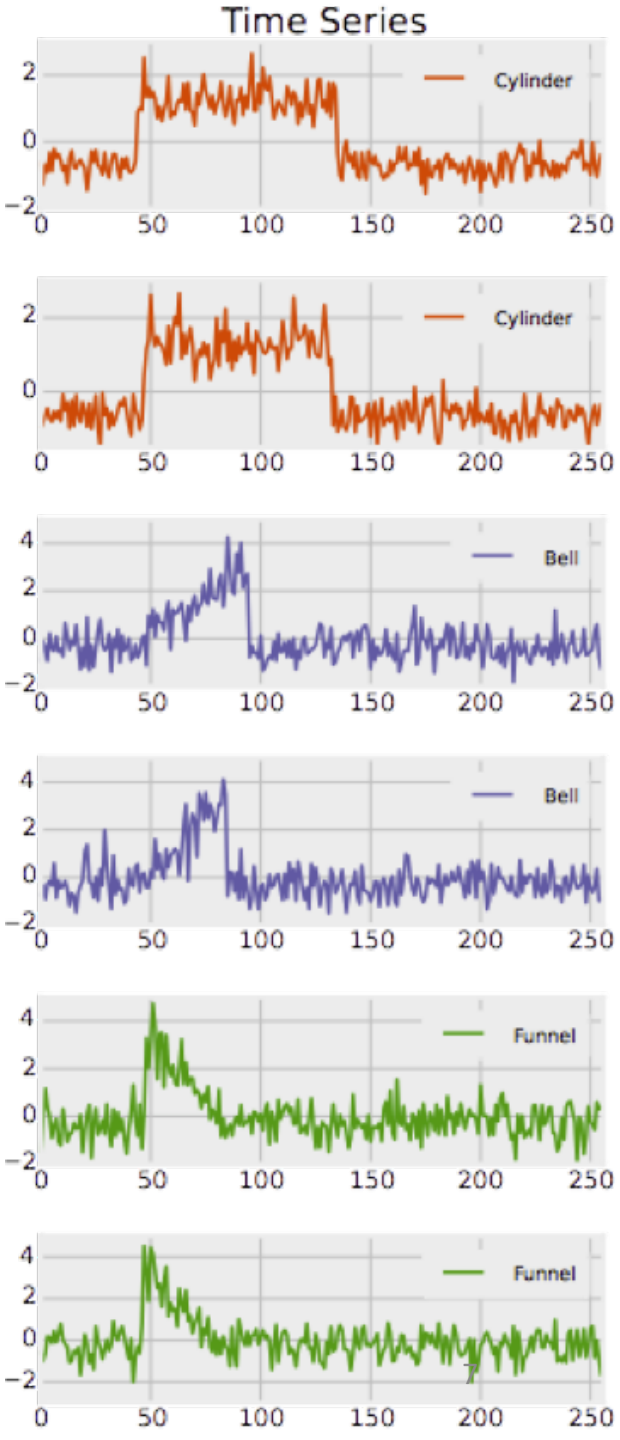
- It is convenient to have time series that are equally spaced and synchronized across time stamps
- However, it is common for time series to contain missing data
- One method is to apply linear, i.e. estimate the (missing) values at desired time stamps
- Linear interpolation: y_{j-1} and y_j are values of the time series at times t_{i-1} and t_i

$$y_{new} = y_{j-1} + \frac{(t_{new} - t_{i-1}) \cdot (y_j - y_{j-1})}{t_i - t_{i-1}}$$



Pre-processing: Noise Removal

- Removes short-term fluctuation and noise
- Binning:
  - Divide the data into *disjoint* intervals of size k. Then calculate the mean values $\mathrm{T} = (\bar{y}_1 \ldots \bar{y}_w)$, with $w = \frac{n}{k}$, in each interval: $\bar{y}_i = \frac{\sum_{j=k(i-1)+1}^{ki} y_j}{k}$
  - This reduces the number of points by a factor of k
- Smoothing (Moving-Averages)
  - Divide the data into *overlapping* intervals of size k over which the averages are calculated
  - Thus, the average is computed at each time stamp $[t_1, t_k], [t_2, t_{k+1}], \ldots$ rather than only at the interval boundaries $[t_1, t_k], [t_{k+1}, t_{2k}], \ldots$



Pre-processing: Normalization

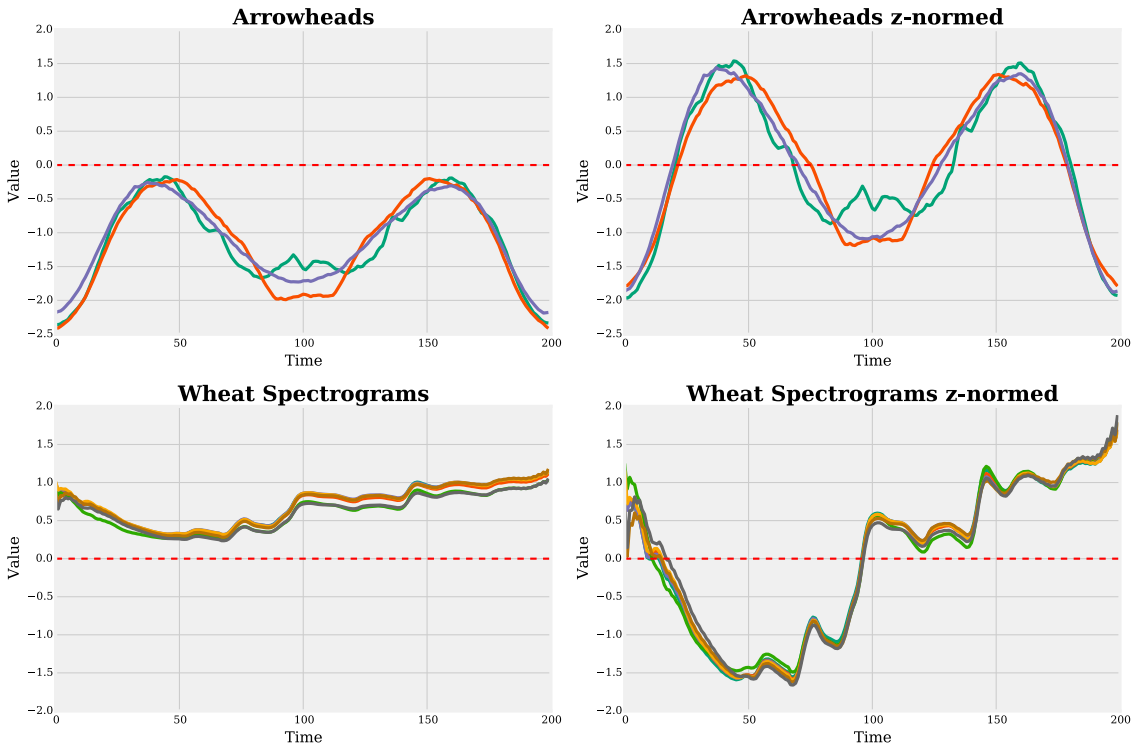
- Time series need to be normalized, especially when different sensors are used to record the data
- Normalization puts the data on the same scale to make comparisons meaningful (i.e. Fahrenheit und Celsius)
- Two methods are commonly used:
  - Z-normalization (is generally preferred)
  - Range-based normalization

Pre-processing: Z-Normalization

- Z-normalization:
  Let μ and σ be the mean and standard deviation of the time series $\boldsymbol{T} = (\boldsymbol{y_1}, \ldots, \boldsymbol{y_n})$, then

$$znorm(T) = \left(y'_1, \ldots, y'_n\right)$$

$$\text{with } y'_i = \frac{y_i - \mu}{\sigma}$$



Pre-processing: Range-based normalization

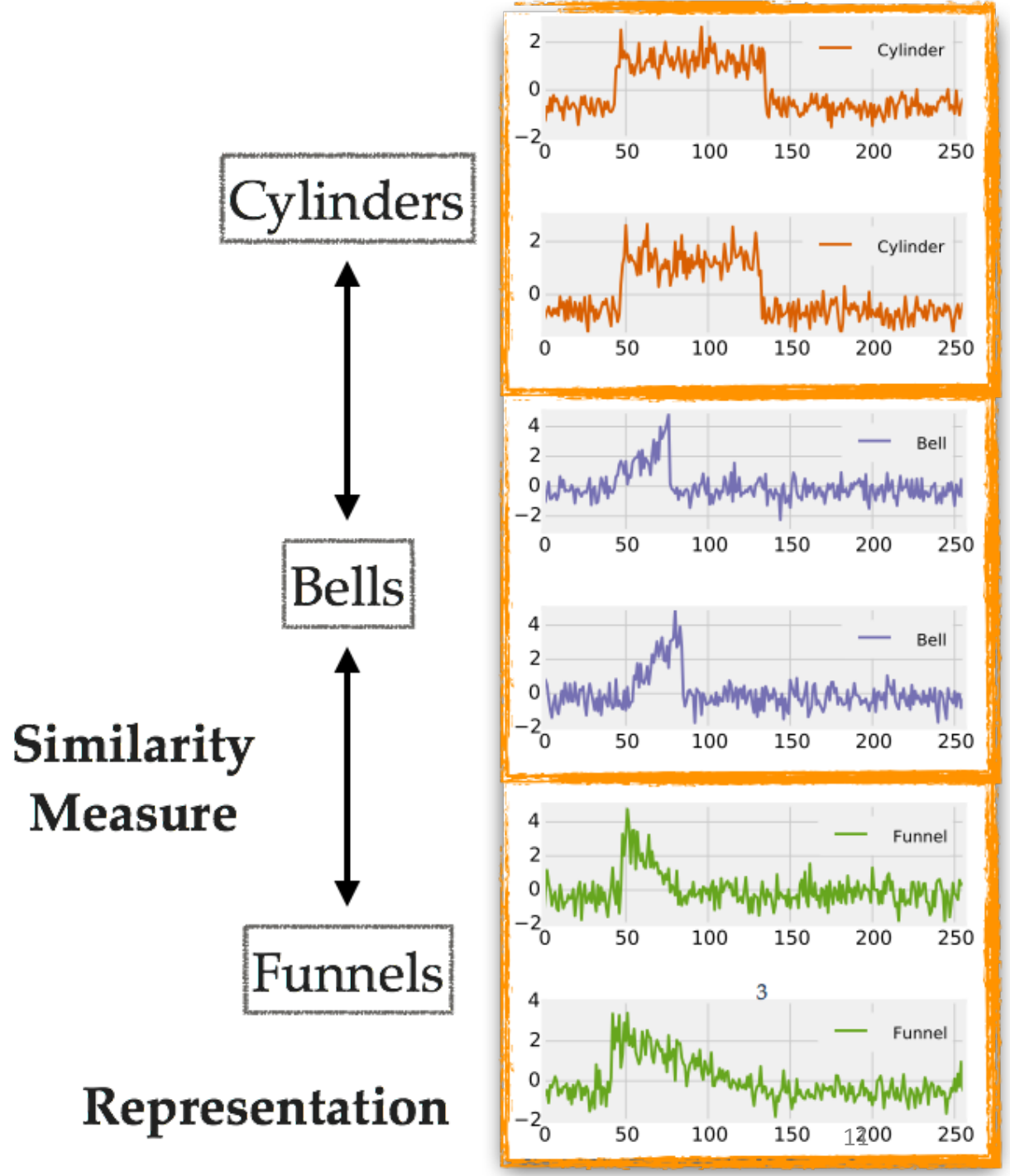
- Range-based normalization:
  The minimum and maximum values over all time series are determined, then each value of the time series $\boldsymbol{T} = (\boldsymbol{y_1}, \ldots, \boldsymbol{y_n})$is mapped to range (0,1) by:

$$range\_norm(T) = \left(y'_1, \ldots, y'_n\right)$$

$$\text{with } y'_i = \frac{y_i - min}{\max - min}$$

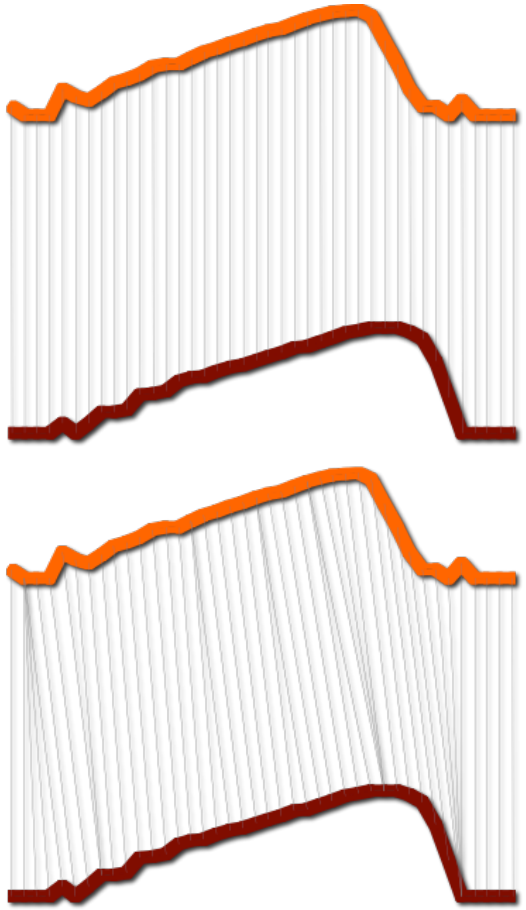
Time Series Similarity

- At the core of time series data analytics there are:
  - a **time series representation.**
  - a **similarity measure** to compare two time series.



Similarity Measures (Distance Measures)

- The *similarity* of two time series Q and T is expressed in terms of a real value using a *distance measure*: $D(Q,T) \rightarrow \mathbb{R}_0^+$
- A *similarity measure* is the inverse of the distance measure: it qualifies *similar (/dissimilar)* time series by a *small (/large)* value
- Time series similarity measures are designed with application specific goals
- Most common methods are *Euclidean distance (ED)* and *Dynamic Time Warping Distance (DTW)*
- Other common distance measures are Longest Common Subsequence or Edit Distance



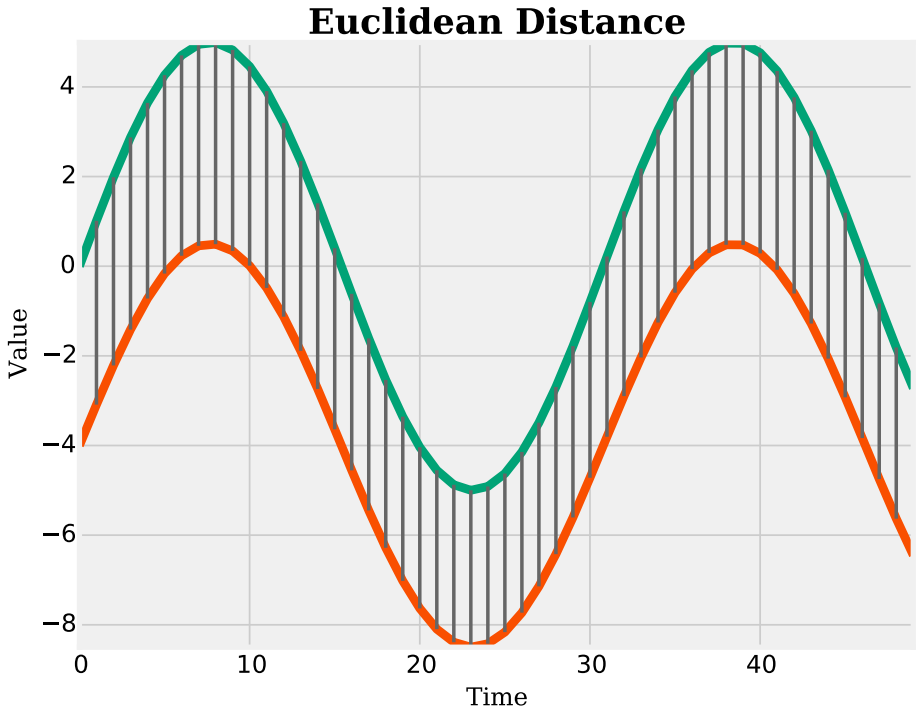
Euclidean Distance

DTW

Euclidean Distance (ED)

- Definition: The *Euclidean distance* between two time series $Q = (q_1, \ldots, q_n)$ and C = $(c_1, \ldots, c_n)$, both of length n, is defined as:

$$D_{ED}(Q, C) = \sqrt{\sum_i (q_i - c_i)^2}$$

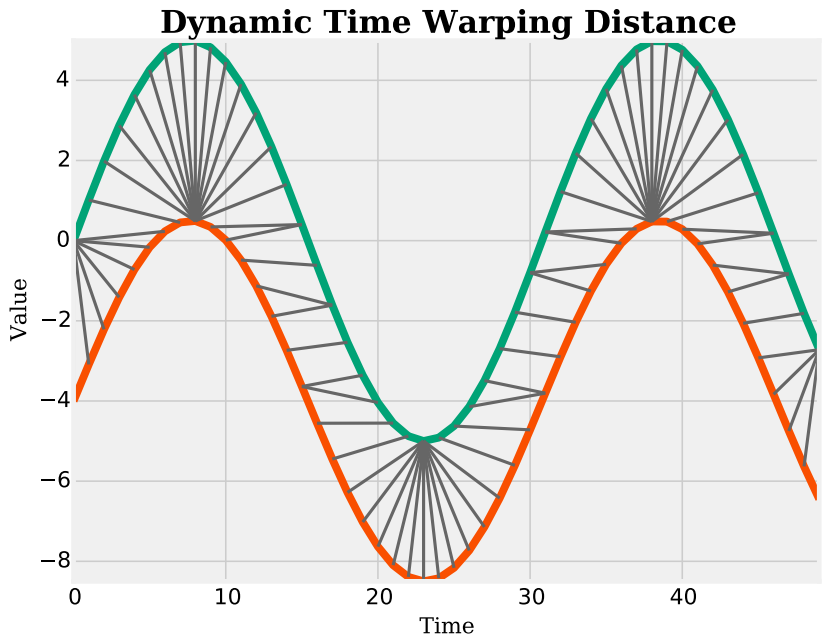


- The ED applies a linear alignment of the time axis
- ED cannot cope with variable length time series
- ED runtime is O(n)

Dynamic Time Warping (DTW)

- *Dynamic Time Warping* applies an elastic transformation of the time axis to detect similar shapes that have a different phase
- This is essentially a peak-to-peak and valley-to-valley alignment of two time series
- Intuition: DTW can be thought of as an extension of the ED, which uses two indices i and j representing both time axis
- These indices are incremented independently:

$$D_{DTW}(Q, C) = \sqrt{\sum_{(i,j)} (q_i - c_j)^2}$$

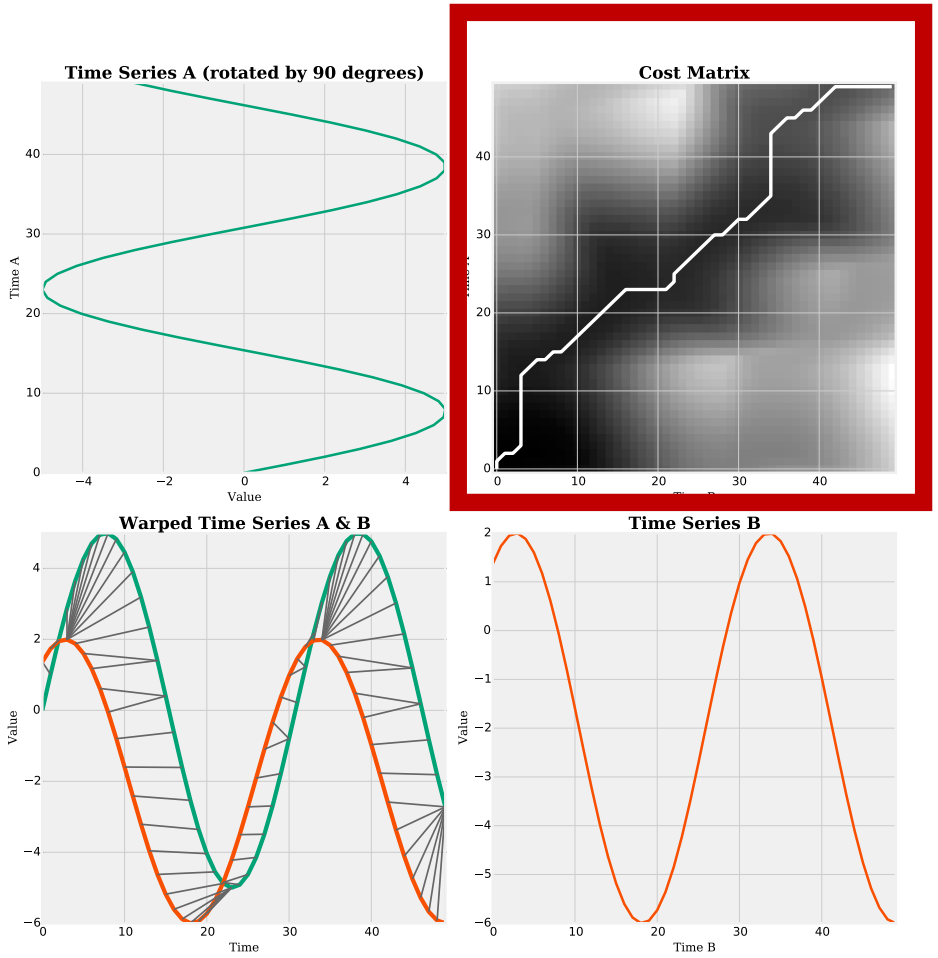


Dynamic Time Warping (Cost Matrix)

- DTW starts by computing a *cost matrix* M with the distances between all pairs of values in $Q = (q_1 \dots q_n)$ and $C = (c_1 \dots c_n)$
- This matrix has the dimensionality n^2 and is given by:

$$M_{i,j} = \left(q_i - c_j\right)^2, \qquad i, j \in [1 \dots n]$$

- DTW then searches for the optimal *warping path*

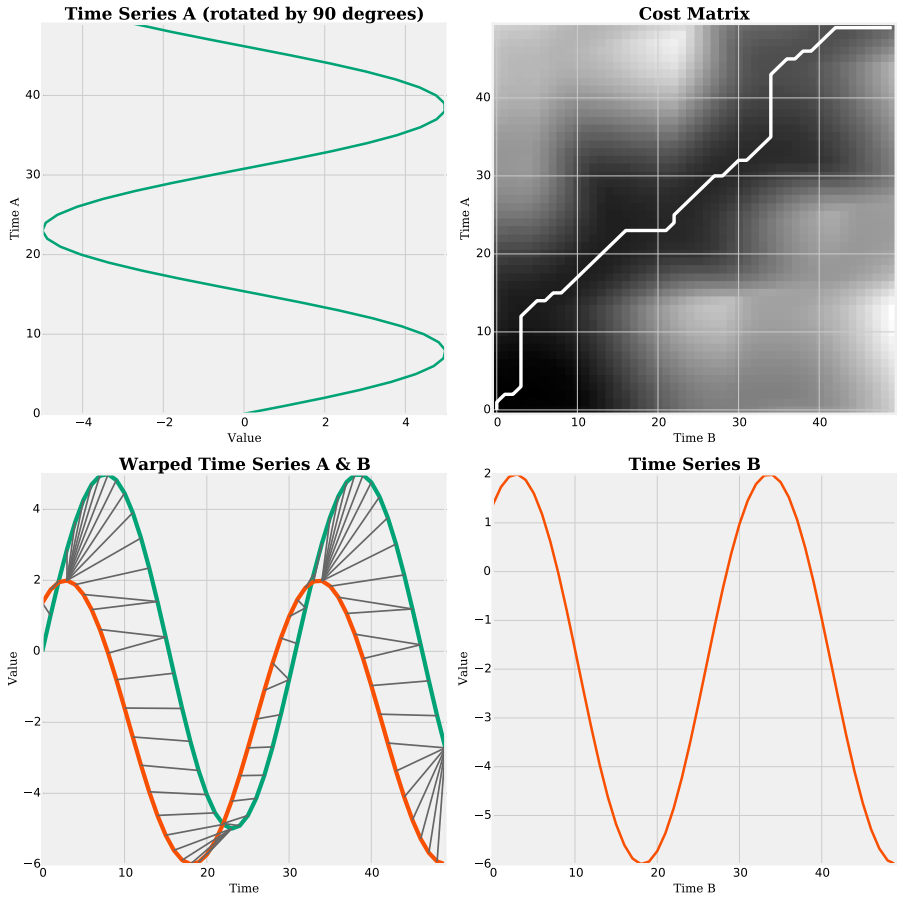


Dynamic Time Warping (Warping Path)

- Definition: A warping path p is defined as a set of tuples that defines a traversal of the cost matrix M whereas i and j represent indices of the values in $Q = (q_1 \dots q_n)$ and $C = (c_1 \dots c_n)$:

  $p_1 = \{(\mathbf{1}, \mathbf{1}), (2,2), \dots, (n-1, n-1), (\boldsymbol{n}, \boldsymbol{n})\}$
  $p_2 = \left\{(\mathbf{1}, \mathbf{1}), (1,2), \left(\underbrace{1}_{\boldsymbol{i}}, \underbrace{3}_{\boldsymbol{j}}\right), (2,3), \dots, (n-1, n), (\boldsymbol{n}, \boldsymbol{n})\right\}$

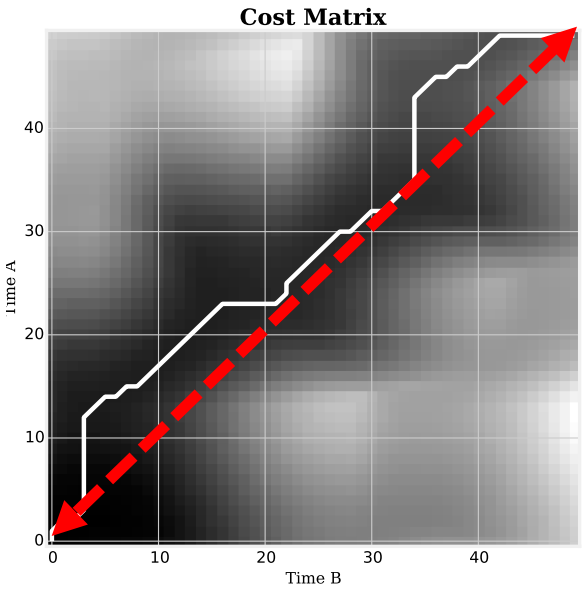
- A valid warping path has to satisfy two conditions:
  - The start has to be (1,1) and the end has to be (n,n)
  - The path may proceed by a maximum of one index: $0 \leq i_{a+1} - i_a \leq 1$ and $0 \leq j_{a+1} - j_a \leq 1$ for all $a < n$
- ED is the special case of the diagonal in the cost matrix



Dynamic Time Warping (Distance)

- Definition: The DTW distance is defined as the warping path with the minimal total cost through the cost matrix M

$$D_{DTW}(Q, C) = min\left\{\sum_{(i,j)\in p}(q_i - c_j)^2 \mid p \in M\right\}$$

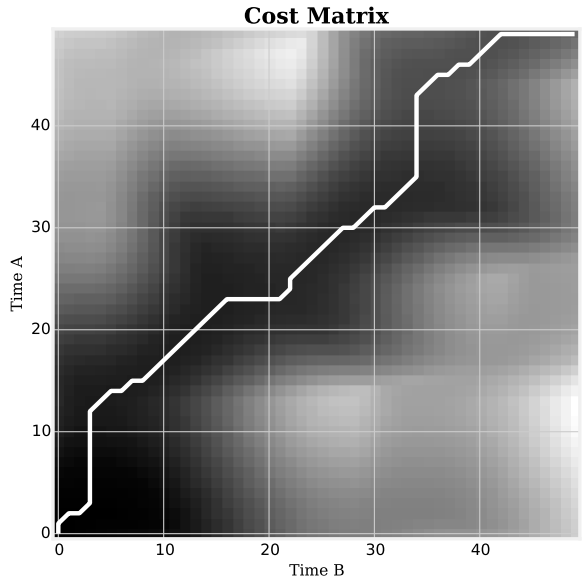


- As DTW is a peak-to-peak and valley-to-valley alignment of two time series, it may produce suboptimal results if there is a variable number of peaks and valleys
- DTW runtime is $O(n^2)$

Dynamic Time Warping (Algorithm)

```
int DTWDistance(
        Time Series q[1..n], TimeSeries c[1..n]) {
    // cost matrix
    M:= array [0..n, 0..n]
    for i := 1 to n
       M[i, 0] := infinity
    for i := 1 to n
       M[0, i] := infinity
    M[0, 0] := 0

    //find optimal path
    for i := 1 to n
        for j := 1 to n
            cost := D(q[i], c[j])    // measure distance
            M[i, j] := cost + minimum(
                  M[i-1, j ],       // insertion
                  M[i , j-1],       // deletion
                  M[i-1, j-1])    // match
    return M[n, n]
}
```

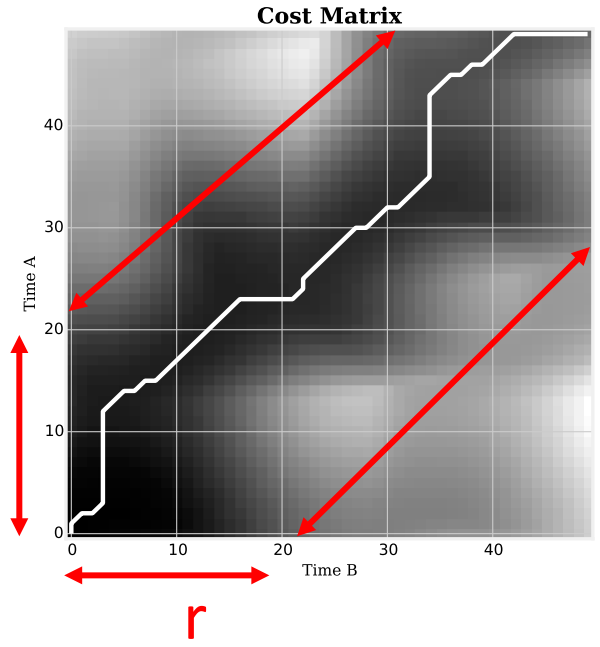


Dynamic Time Warping (Warping Window)

- A warping window constraint can be set to reduce the search space (computational complexity)
- It defines the amount of warping allowed between each pair of points on the warping path p:

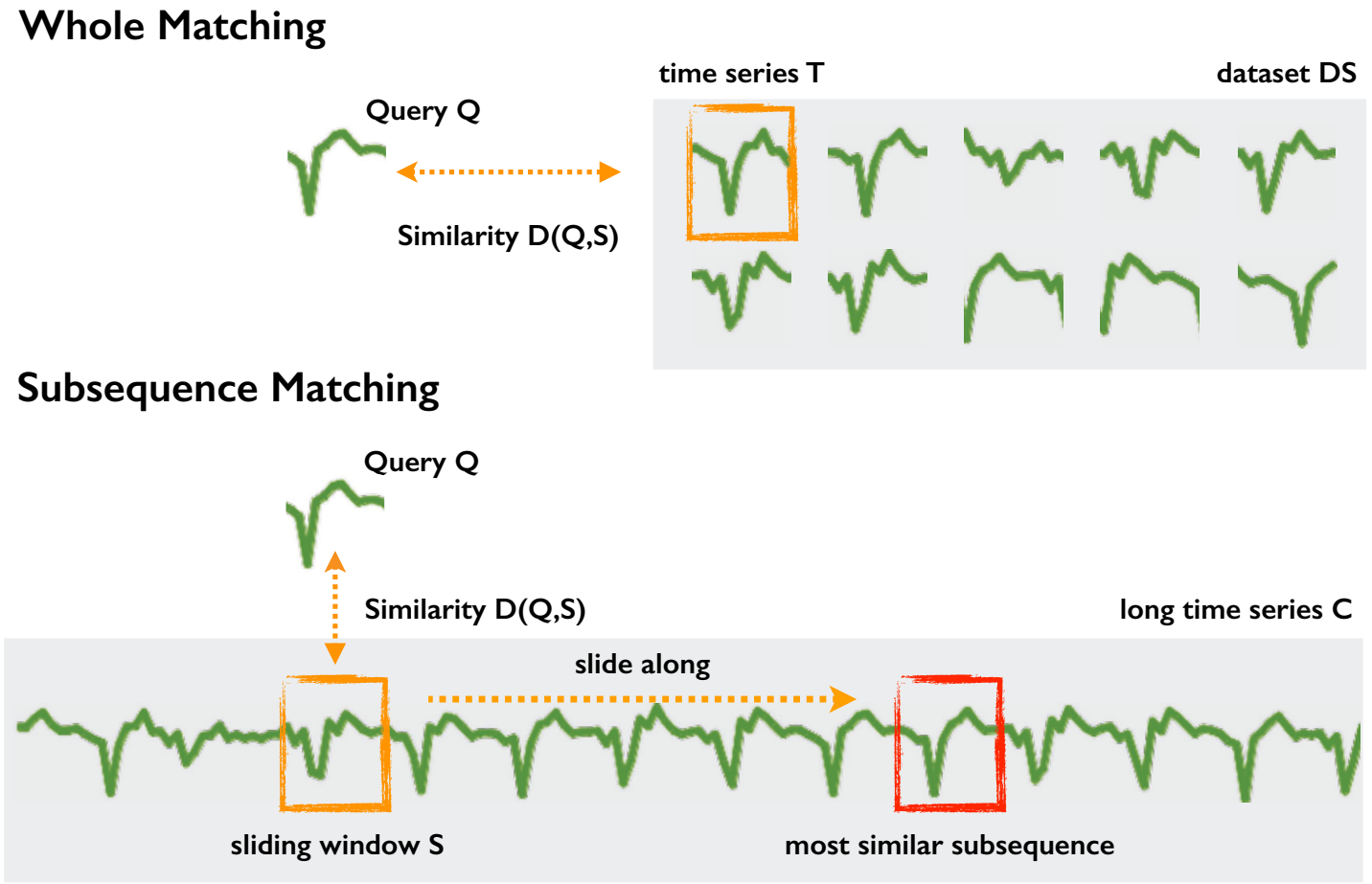
$$|i - j| \leq r \cdot n \qquad \forall (i, j)\ \epsilon\ p$$

- DTW with a warping window constraint r has a computational complexity of $O(nr)$



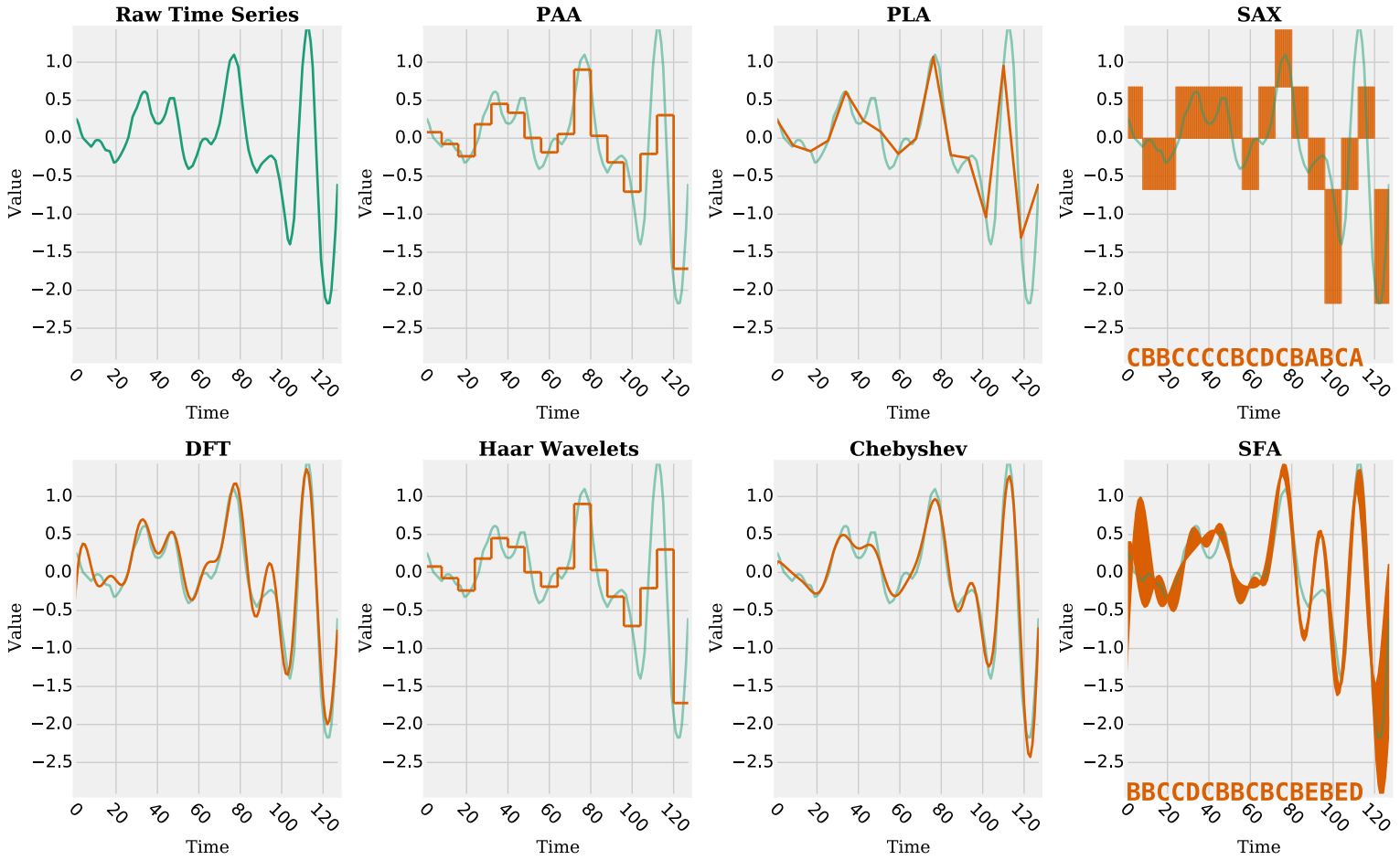
Similarity: Whole vs. Subsequence Matching

- *Whole matching*: the distance is calculated between two whole time series
- *Subsequence matching*: Searches for the best subsequence within a longer time series
- Complexity for time series length n and subsequence length w with Euclidean distance:
  - Whole matching: $O(n)$
  - Subsequence matching: $O\big(w(n-w)\big)$



Time Series Data Transformations

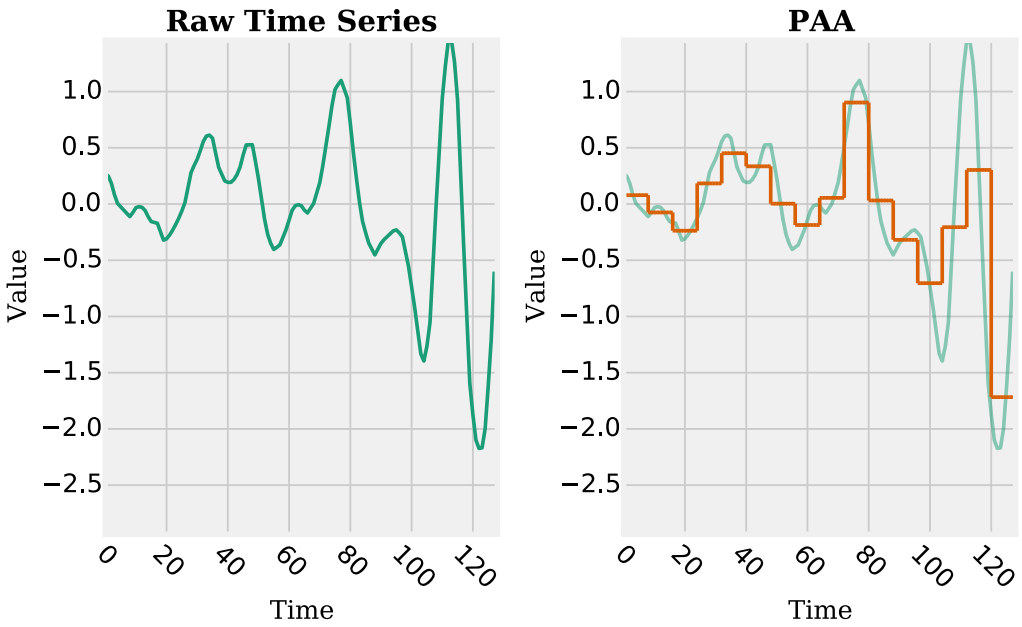
- A variety of methods exist for reducing the dimensionality of time series (i.e. for noise filtering, faster processing),
- *Real-valued* methods transform into a smaller number of numeric values and *symbolic* methods transform into discrete values (sequences)



Piecewise Aggregate Approximation (PAA)

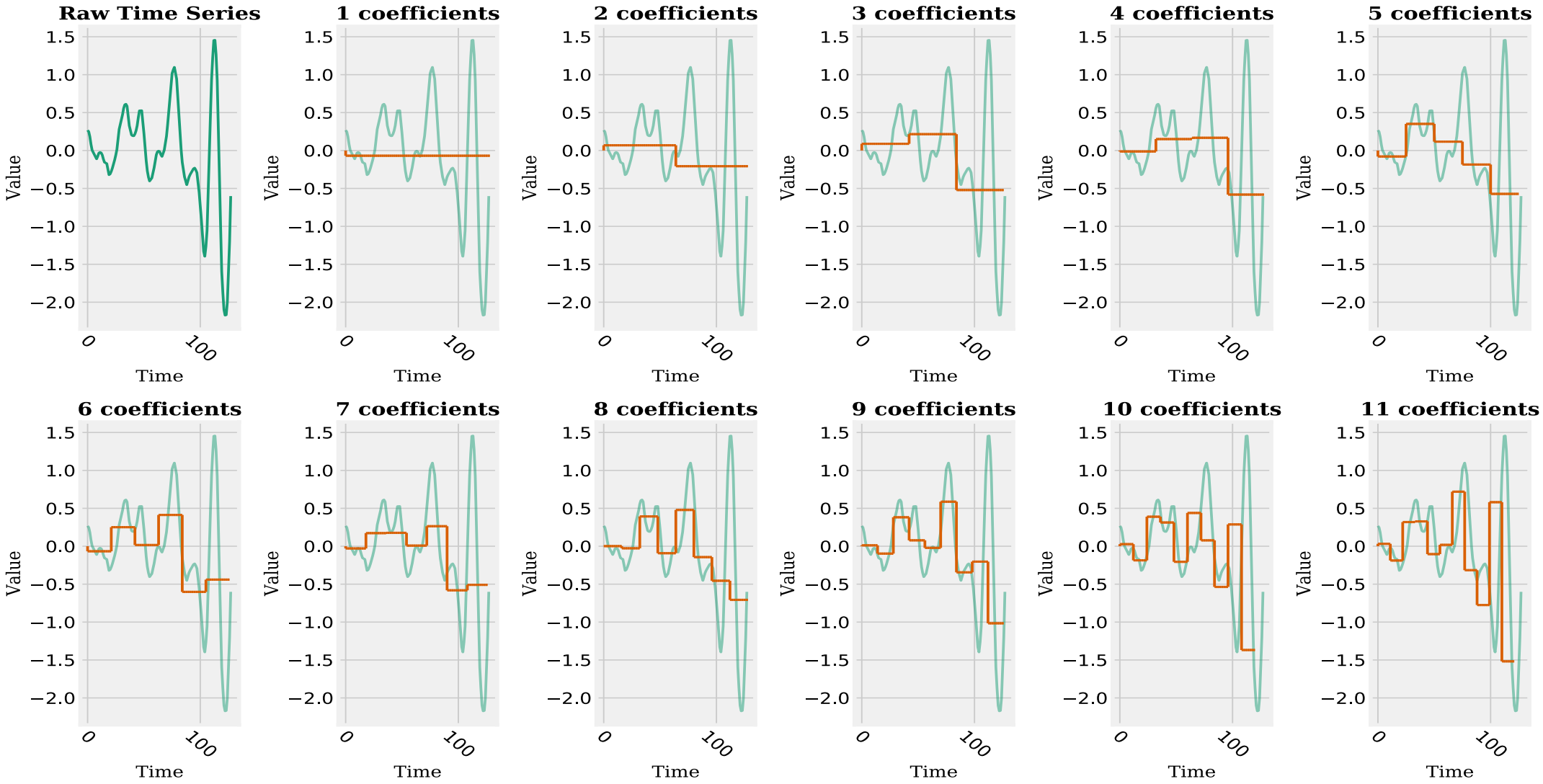
- Intuition: to represent a time series of length n the data is divided into w equal-sized intervals and the mean value in each interval is calculated
- PAA: a time series $T = (y_1, \ldots, y_n)$ of length n is represented by a w-dimensional sequence of mean values $\mathrm{C} = (\overline{y_1}, \ldots, \overline{y_w})$, where i-th element is calculated as:

$$\overline{y_i} = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} y_j$$



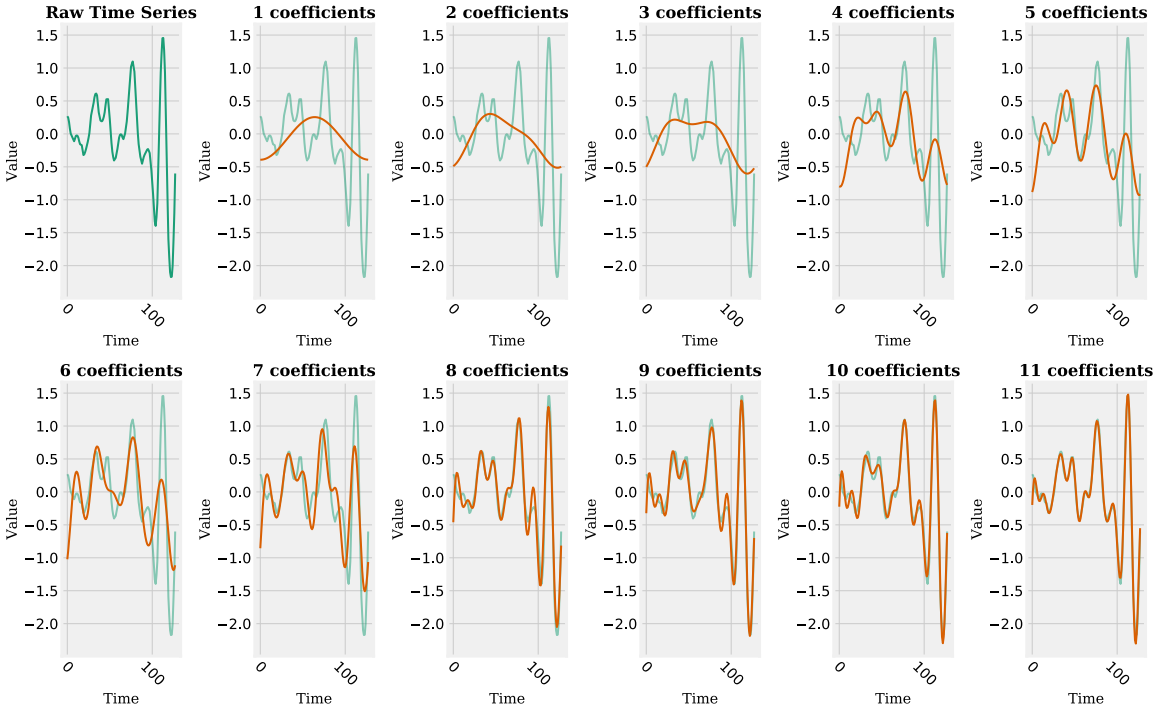
PAA: 0,08 -0,08 -0,24 0,18 0,45 0,34 0,00 -0,19 0,05 0,90 0,03 -0,32 -0,70 -0,21 0,30 -1,72

Piecewise Aggregate Approximation (PAA)



Discrete Fourier Transform (DFT)

- Intuition: each series of length n can be expressed by a linear combination of smooth periodic sinusoidal series
- Each wave is represented by a complex number (*Fourier Coefficient*)
- This representation is called *Frequency Domain*
- The DFT concentrates most of its energy in the *first few* Fourier coefficients
- Low-pass filter: One can approximate a time series by its first *w Fourier coefficients*



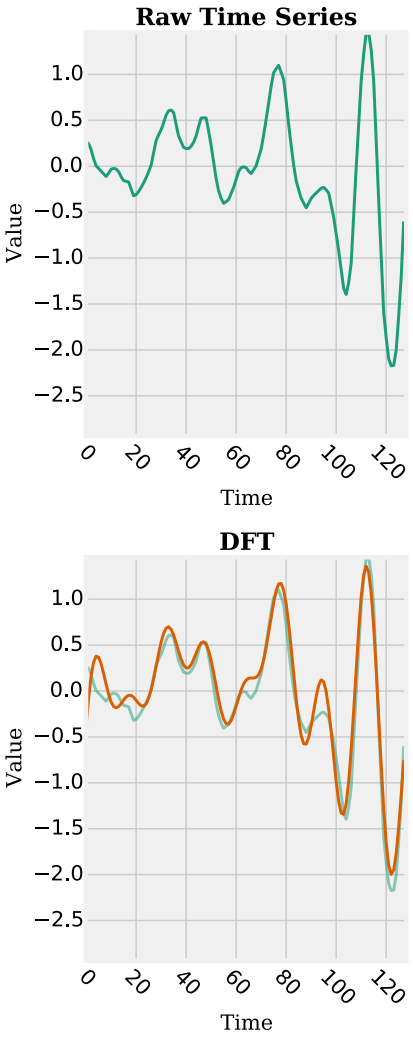
DFT : 0 -8.81 -20.7 -11.9 -6.28 -8.02 -0.67 15.31 -18.7 -18.36 -5.67 16.84 -8.919 -23.80 10.70 -21.92 25.255-1.321 …

Discrete Fourier Transform (DFT)

- The DFT decomposes a time series T of length n into a sum of n orthogonal basis functions using sinusoid waves
- A Fourier coefficient (sinusoid wave) is represented by the complex number: $X_u = (real_u, imag_u), \quad for\ u = 0,1 \dots, n-1$

- The n-point DFT of a time series $T = (y_1 \dots y_n)$ is then given by: $DFT(T) = X_0, \dots, X_{n-1} = (real_0, imag_0, \dots, real_{n-1}, imag_{n-1})$

- with $X_u = \frac{1}{n}\sum_{i=1}^{n} y_i \cdot e^{-\frac{j2\pi ui}{n}}, for\ u\epsilon[0,n), j = \sqrt{-1}$

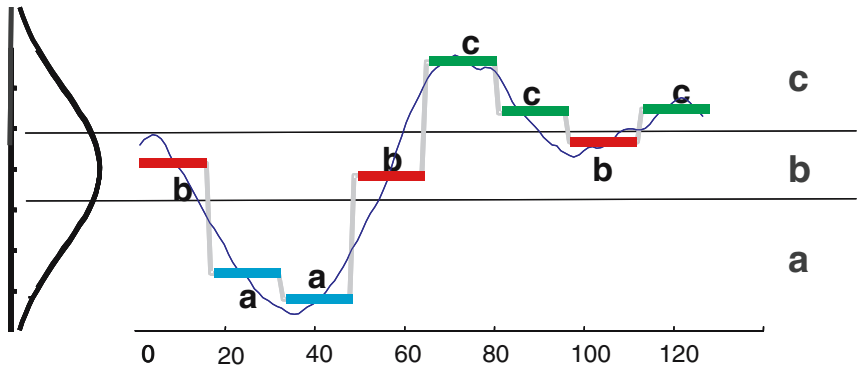


Discrete Fourier Transform (DFT)

- The *first Fourier coefficient* is equal to the mean of a time series and can be discarded to obtain offset invariance: $X_0 = \frac{1}{n}\sum_{i=1}^{n} y_i \cdot e^0$
- Using only the first few Fourier coefficients is equal two low-pass filtering (smoothening) a time series
- Computational Complexity: The *Fast Fourier Transform* has a computational complexity of $O(n \log n)$ to compute the DFT of a time series of length n

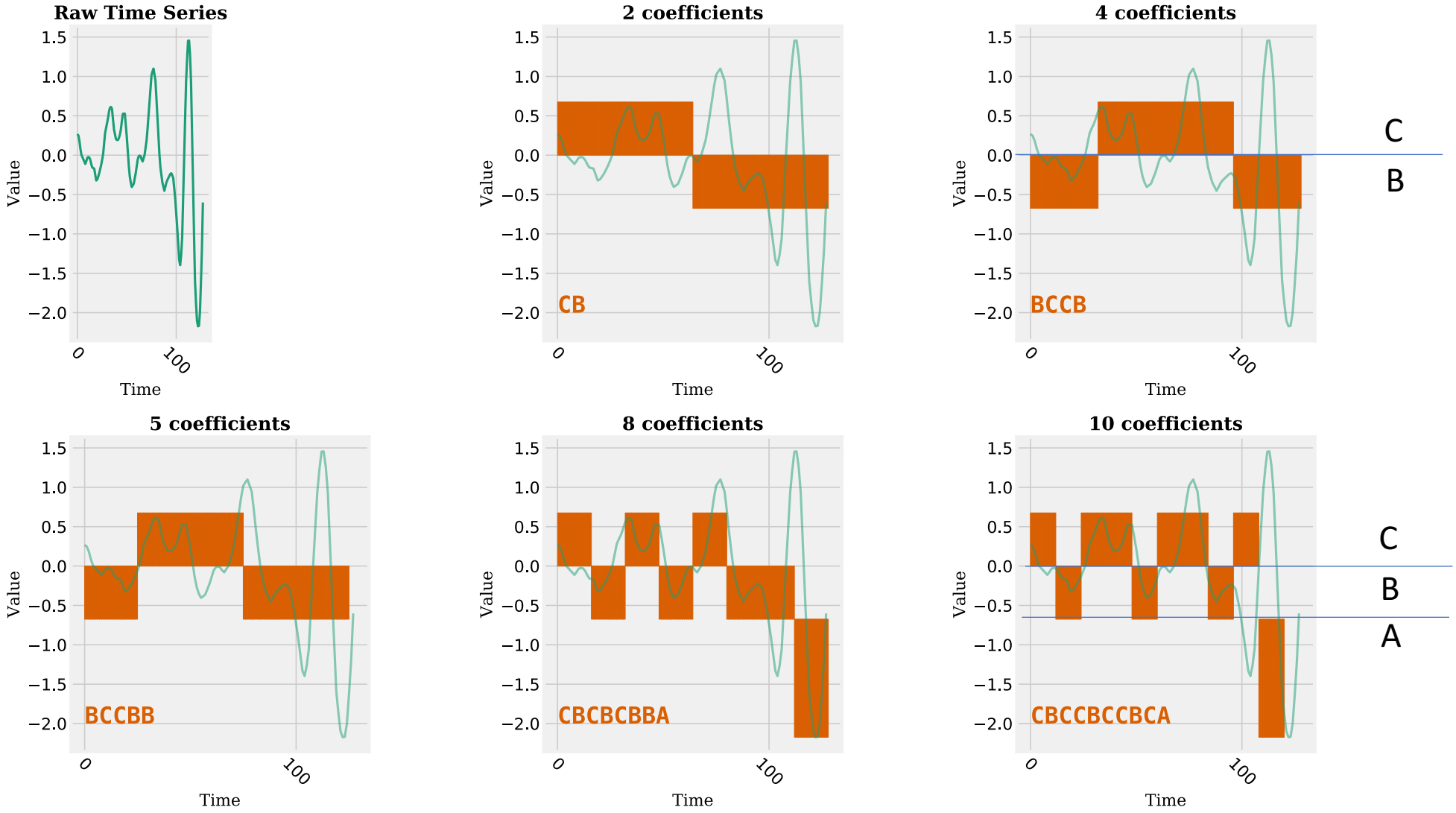
Symbolic Aggregate Approximation (SAX)

- SAX converts the time series into a discrete sequence of symbols by
  - X-axis: Applies dimensionality reduction using the PAA transformation
  - Y-axis: Transforms each PAA value using discretization to a symbol
- Values sampled from a z-normalized time series follow Gaussian distribution
- SAX applies discretization based on Gaussian distribution to produce equi-probable symbols
- A real valued time series is then represented by a word



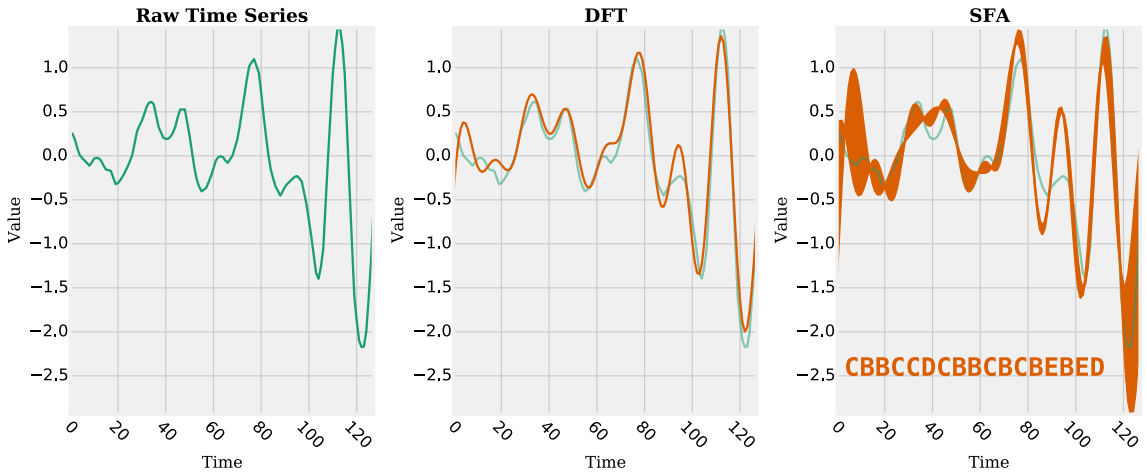
from: Lin et al.: **Experiencing SAX: a novel symbolic representation of time series**

Symbolic Aggregate Approximation (SAX)



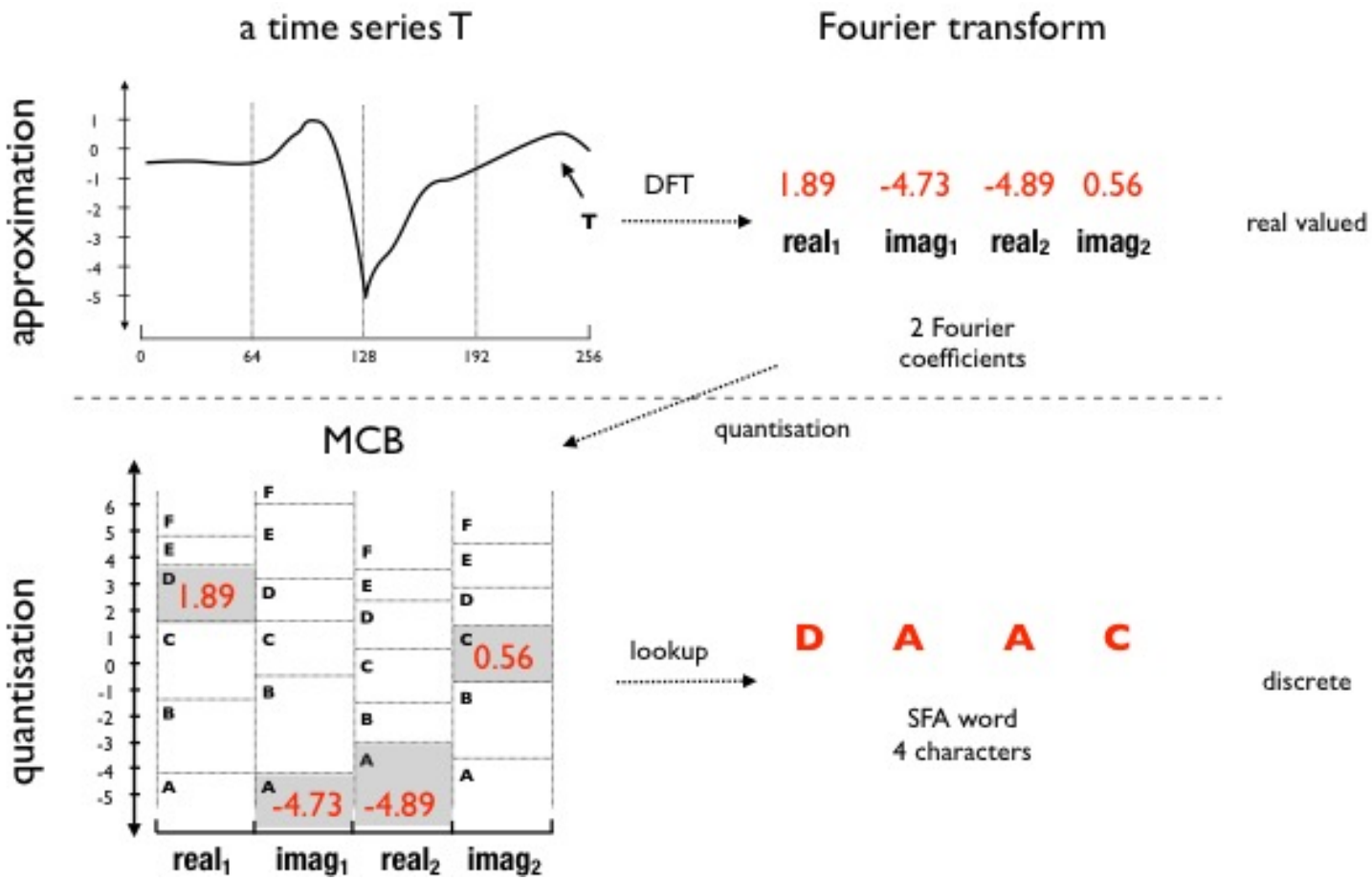
Symbolic Fourier Approximation (SFA)

- SFA represents each real valued time series by a word
- SFA is composed of
  - *a) approximation* using the Fourier transform and
  - b) a data adaptive *discretization*
- The discretization intervals are learned from the Fourier transformed data distribution rather than using fixed intervals



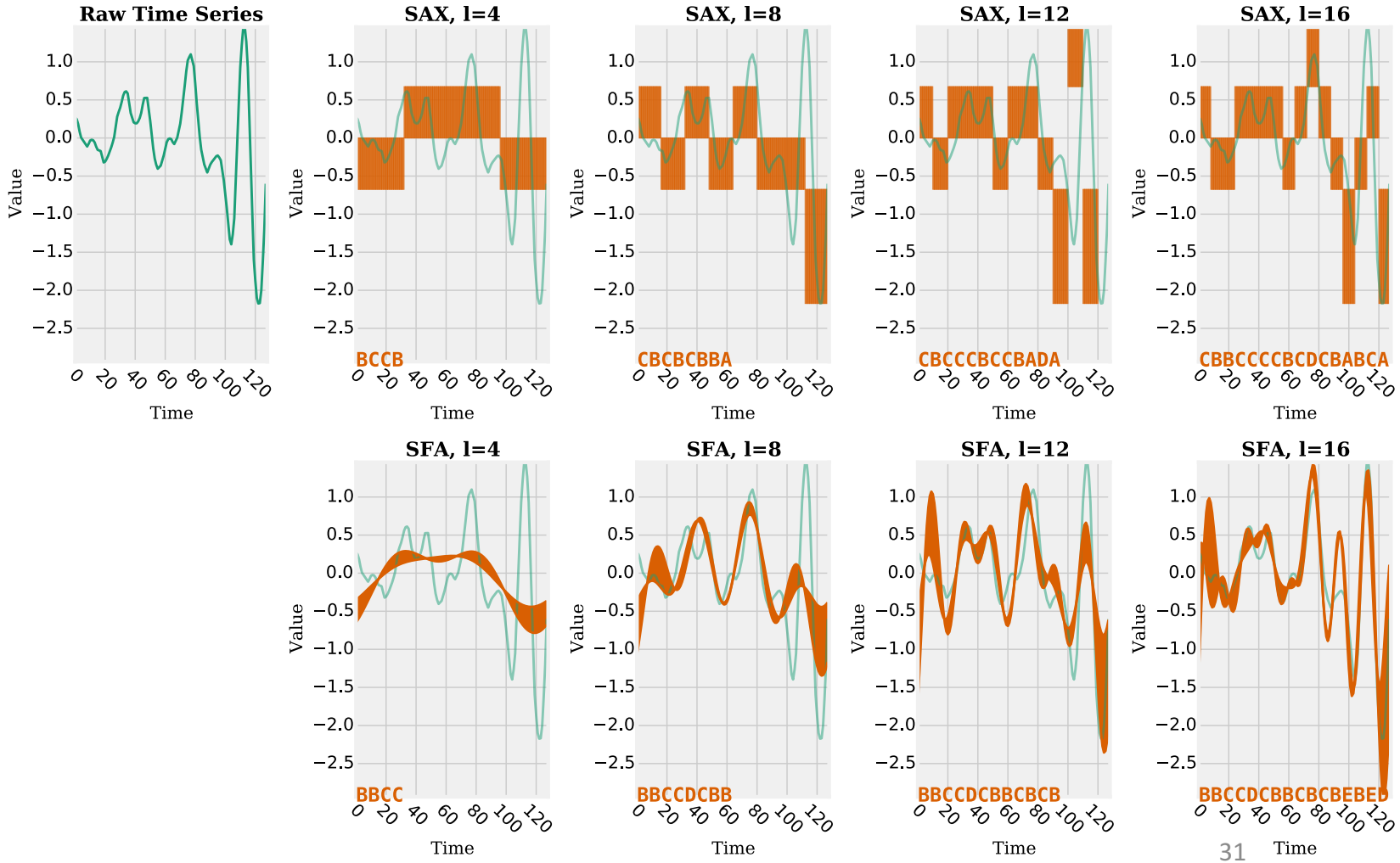
Raw:	DFT	Discretization
0.2679	0	C
0.2480	-8.81	B
0.1828	-20.7	B
0.0817	-11.9	C
0.0051	-6.28	C
-0.023	-8.02	D
-0.052	-0.67	C
-0.082	15.31	B
-0.111	-18.7	B
-0.075	-18.36	C
-0.032	-5.67	B
-0.022	-16.84	C
-0.029	-8.919	B
[...]	[...]	[...]

Symbolic Fourier Approximation (SFA)



Comparison of SFA and SAX

- Discretization and approximation cause loss of information
- The higher the number of symbols and the alphabet size, the more exact is the representation

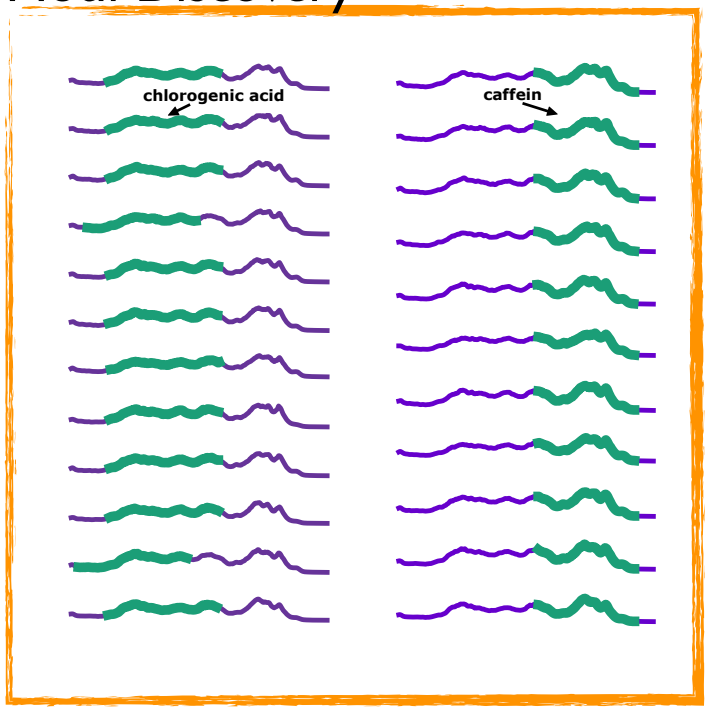


Properties of Symbolic Representations

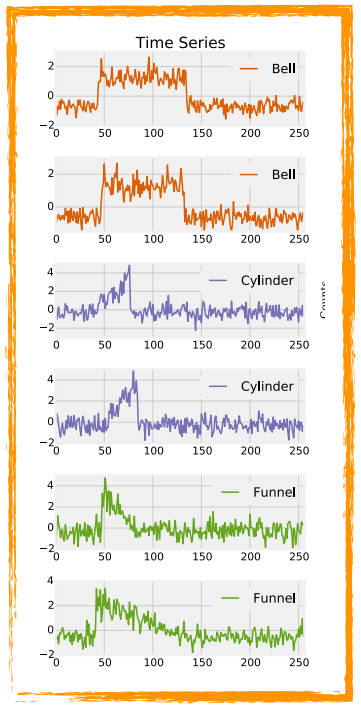
- Noise removal
  - SAX: Using PAA and quantization.
  - SFA: Using the DFT (low-pass filter) and quantization.
- String representation
  - Allows for string domain algorithms like hashing or the bag-of-words to be applied
- Dimensionality Reduction
  - Allow for indexing high dimensional data using the iSAX index (SAX) or the SFA trie (SFA)
- Storage reduction
  - Sequences have a much lower memory footprint than real-valued time series, i.e. less than 1 byte (“Char”) vs 8 byte (“Double”) for each time stamp

Time Series Data Analytics Tasks

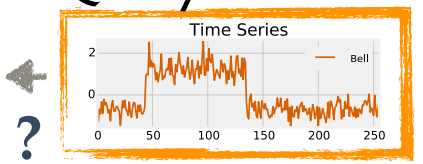
Motif Discovery



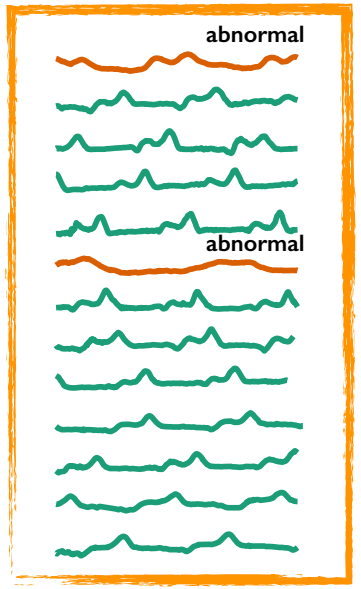
Classification



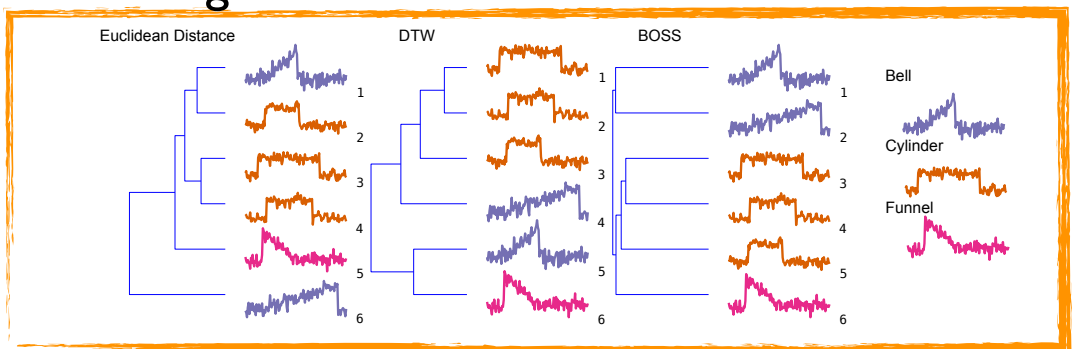
Query



Discords

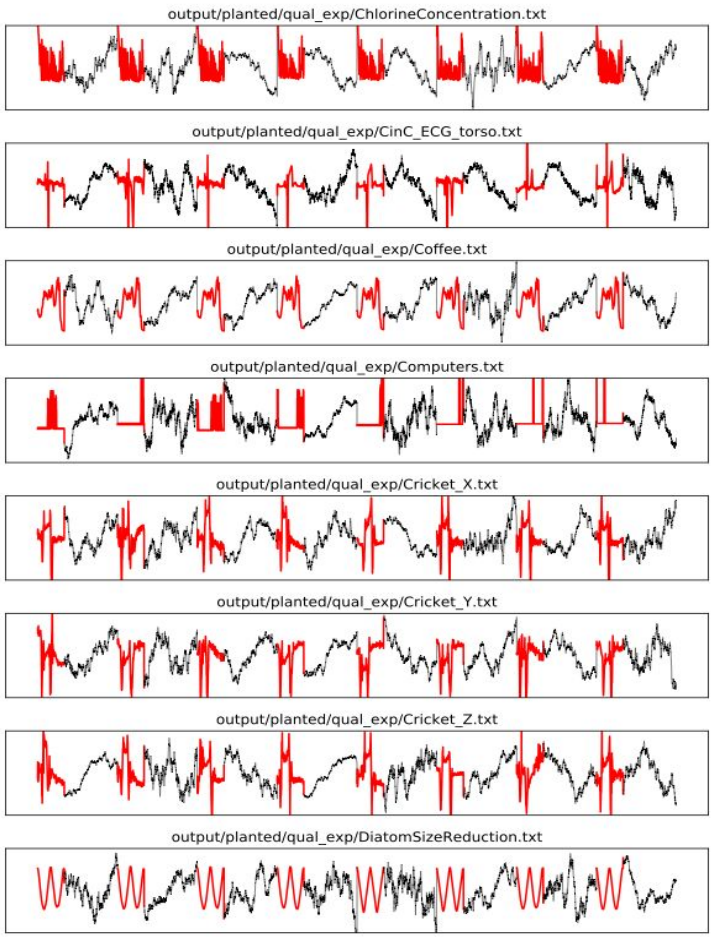


Clustering



Motifs

- A Motif is frequently occurring pattern or shape in a time series
- Distance-based (“exact” Motifs)
  - A subsequence of a time series is said to support a motif, if the distance between the subsequence and the motif is less than a threshold
- Sequential pattern mining (“approximate” Motifs)
  - Discretization is applied to convert the time series into a sequence. Motifs discovery lends methods from sequential pattern mining



From: Jonas Spenger’s Bachelor Thesis

Distance-based Motifs

- These are commonly defined as contiguous sequences of a time series
- Approximate distance match
  - A motif $(s_1, \ldots, s_w)$ is said to approximately match a contiguous subsequence of time series $\mathrm{T} = (y_1 \ldots y_n)$ at position i with $w \leq n$, if the distance between $(s_1 \ldots s_w)$ and $(y_i \ldots y_{i+w-1})$ is at most ϵ.
  - The most common distance measure used is the Euclidean distance

- Motif count: The number of matches within threshold ϵ of a motif to the time series is defined as the number of matches
- A typical goal is to find the most frequent motifs

Naïve Algorithm

```
Motif findBestMotif(
        Time Series (y1,…,yn), WindowLength w, Threshold epsilon)
Begin
    for i := 1 to n-w+1 do
        candidate_motif = (yi,…,yi+w-1);
        for j:=1 to n-w+1 do
            D = computeDistance(candidate_motif, (yj,…,yj+w-1));
            If (D < epsilon) and (non-trivial match) then
                  increment count of candidate_motif by 1
            end if
        end for
        if (candidate_motif has highest count so far) then
             update best_candidate to candidate_motif;
        end if
    return best_candidate;
end
```

- The naïve algorithm extracts all candidate motifs of length w from a time series and computes the distance to all offsets
- The number of matches is counted
- Trivial matches are overlapping sub-sequences (i.e. i = j)
- The approach requires a nested-loop and the number of operations is equal to the size of the time series, thus $O(n^2)$ distance computations
- Total complexity for Euclidean distance $O(n^2 w)$

Ideas to speed-up Motif discovery

- Compute a fast lower bound for the distance
- If the lower bound is greater than epsilon then the real distance does not have to be computed

- Approaches:
  - Use PAA and compute distance only for mean values

$$Dist(X, Y) \geq \sqrt{m} \cdot Dist(X', Y')$$

  - Use SAX and a precomputed lookup table for distances between symbols

Relation to Sequential Pattern Mining

- First, convert the time series into a sequences using for example SAX
- Now apply String mining algorithms
- Itemset Mining: discover frequent itemsets using association rule mining algorithms (ARM). The A-Priori algorithm will be presented in the lecture

Questions?