DATA MINING 2

Time Series – Approximation & Clustering

Riccardo Guidotti

a.a. 2024/2025

Slides edited from Keogh Eamonn’s tutorial

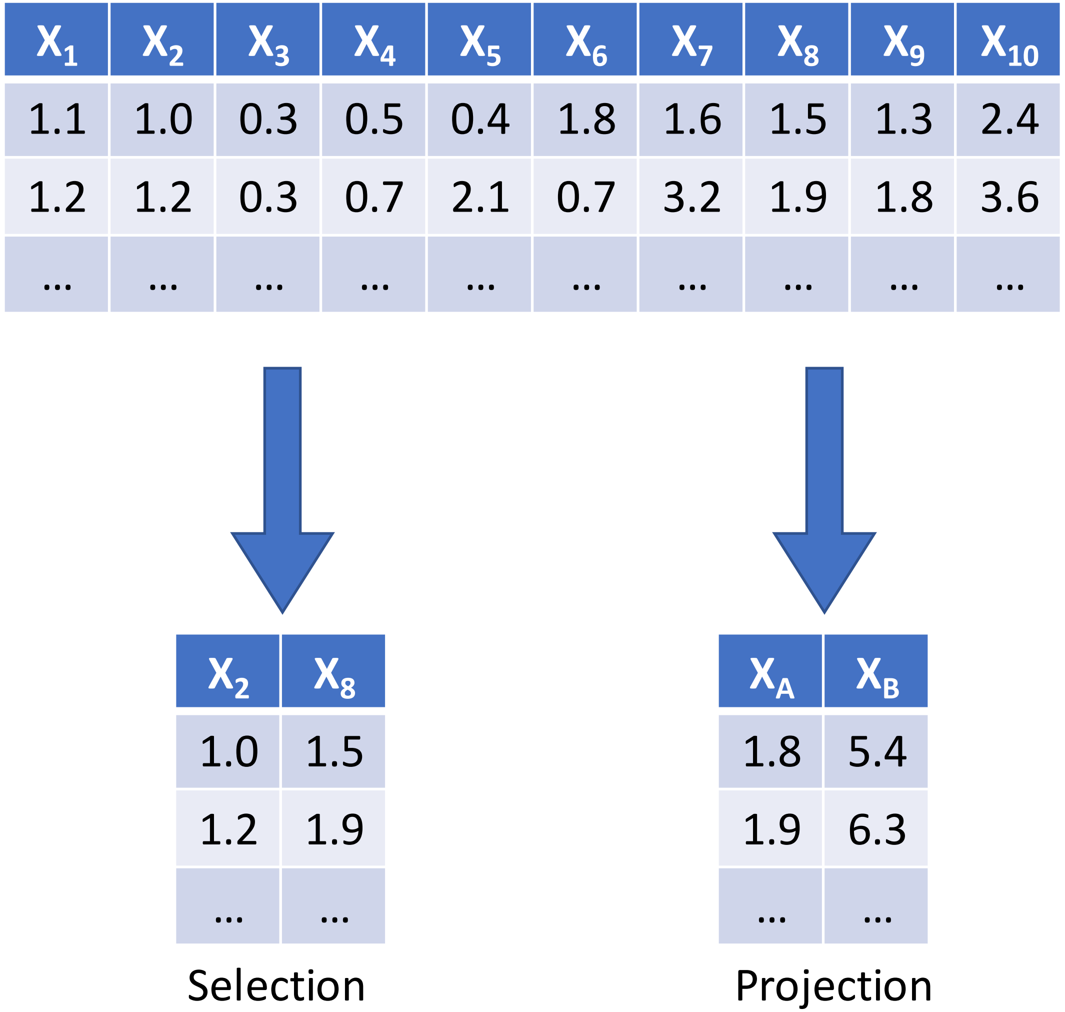
IN SUPREMÆ DIGNITATIS 1343

UNIVERSITÀ DI PISA

Time Series Approximations

Dimensionality Reduction

- Dimensionality reduction is the process of reducing the number of variables under consideration by obtaining a subset of principal variables.
- Dimensionality Reduction approaches can be divided into:
    - **Feature Selection:** variables are selected among the existing ones
    - **Feature Projection:** new variables are created to compactly represent the existing ones.



X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}
1.1	1.0	0.3	0.5	0.4	1.8	1.6	1.5	1.3	2.4
1.2	1.2	0.3	0.7	2.1	0.7	3.2	1.9	1.8	3.6
…	…	…	…	…	…	…	…	…	…

X_2	X_8
1.0	1.5
1.2	1.9
…	…

Selection

X_A	X_B
1.8	5.4
1.9	6.3
…	…

Projection

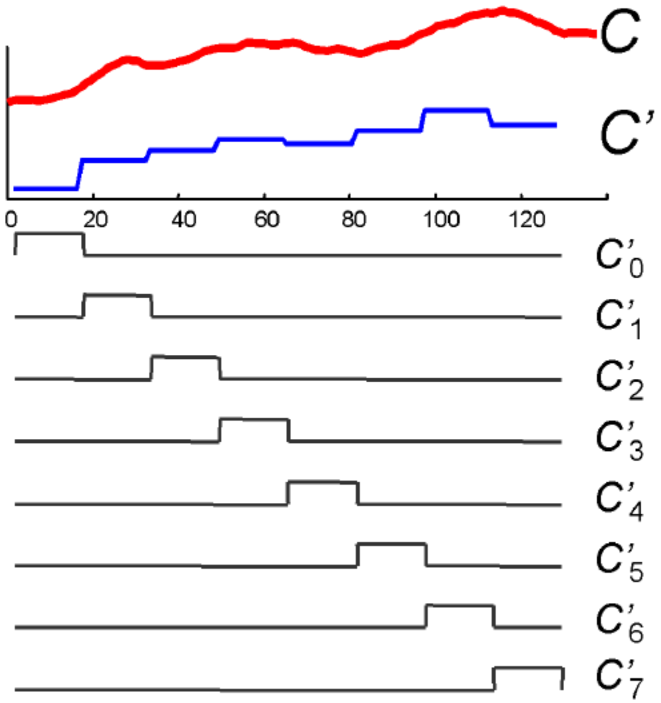
Time Series Approximation

- Time Series Approximation is a special form of Dimensionality Reduction specifically designed for TSs.
- Time Series Approximation consists in representing a TS into a smaller and simpler space that is later used for further calculus (e.g. DTW).
- Approximation vs Compression: the approximated space is always understandable, while the compressed space is not necessarily understandable.
- Approximated representations can be
    - **Time-Dependent**: the approximated values maintain a temporal ordering
    - **Time-Independent**: the approximated values loose the temporal ordering
    - **Instance-wise**: the approximation operation involves only the values of a single TS
    - **Dataset-wise**: the approximation operation involves the values of a dataset of TS

Piecewise Aggregate Approximation (PAA)

- PAA approximates a TS by dividing it into equal-length segments and using the mean value of the data points that fall within the segment as representation.
- PPA represent the TS as a sequence of box basis functions with each box of the same size.
- Given *T = {x$_1$, …, x$_n$},* PAA reduces *T* from a vector with *n* dimensions to a vector $\bar{T} = \{\bar{x}_1, \dots, \bar{x}_N\}$ with *N* dimensions (with *N* < *n*) by dividing *T* into *N* equi-sized ``frames'' with length *n/N* where the *i-th* element is calculated as

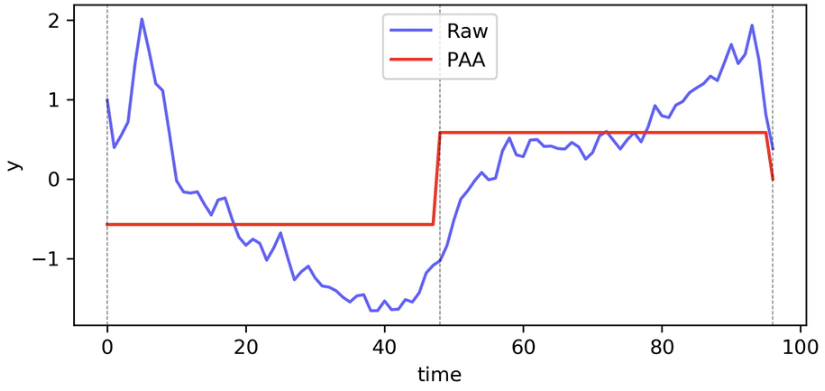
$$\bar{x}_i = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} x_j$$



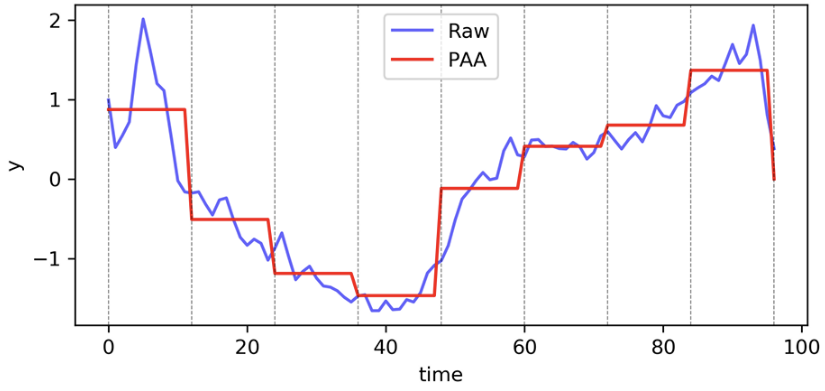
Pros

- Extremely fast to calculate
- Supports non-Euclidean measures
- Supports weighted Euclidean distance

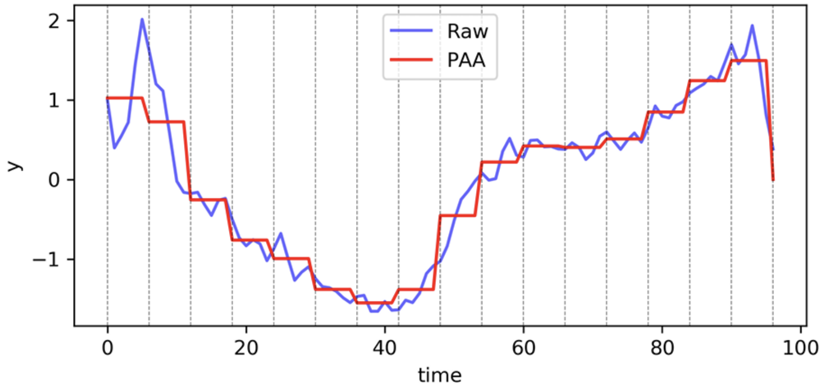
PAA - Example



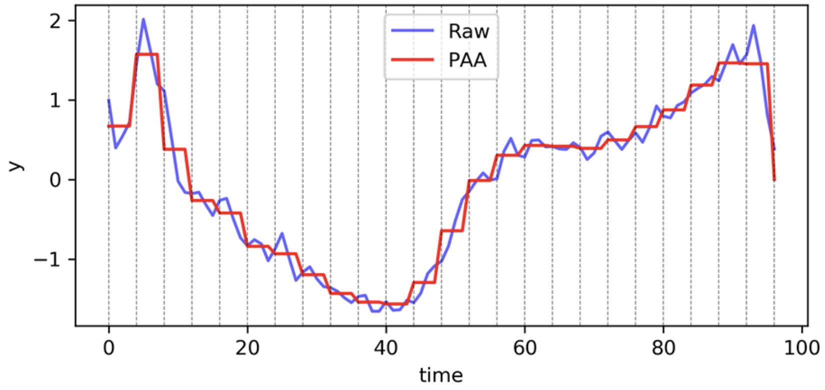
(a) segment $w = 2$.



(b) segment $w = 8$.

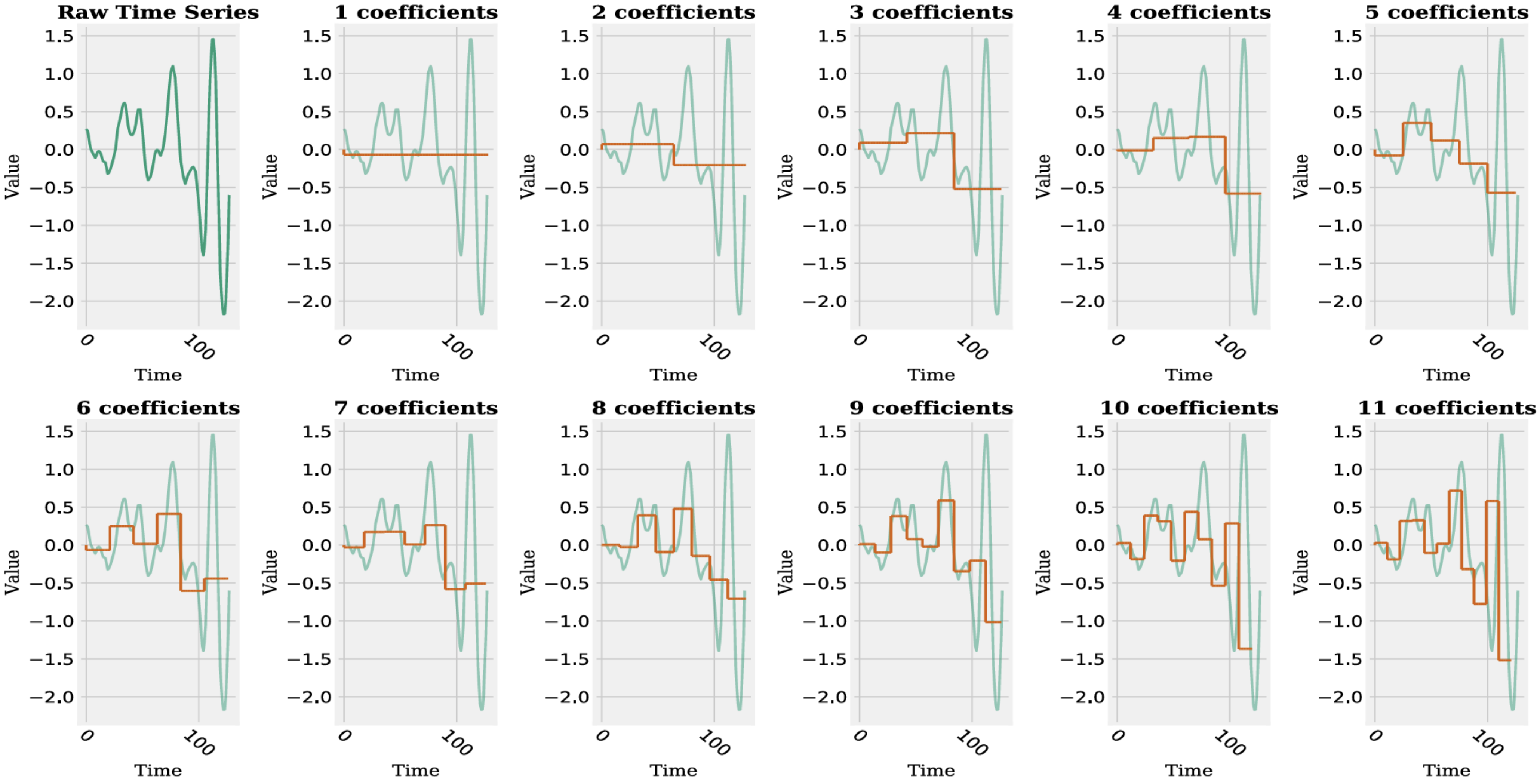


(c) segment $w = 16$.



(d) segment $w = 24$.

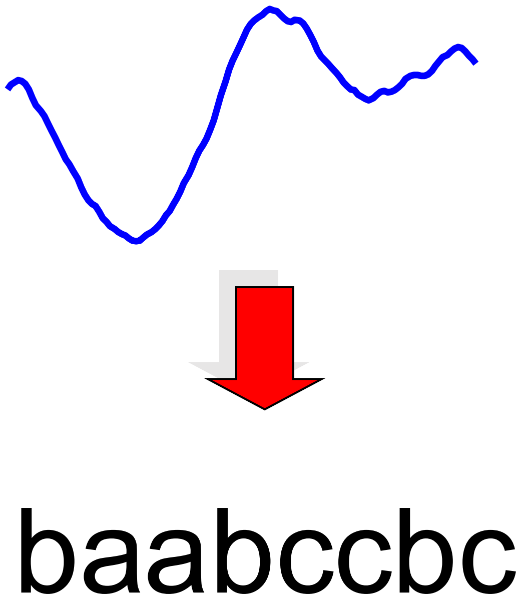
PAA - Example



Raw Time Series
1 coefficients
2 coefficients
3 coefficients
4 coefficients
5 coefficients
6 coefficients
7 coefficients
8 coefficients
9 coefficients
10 coefficients
11 coefficients
Value
Time
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
100

Symbolic Aggregate Approximation (SAX)

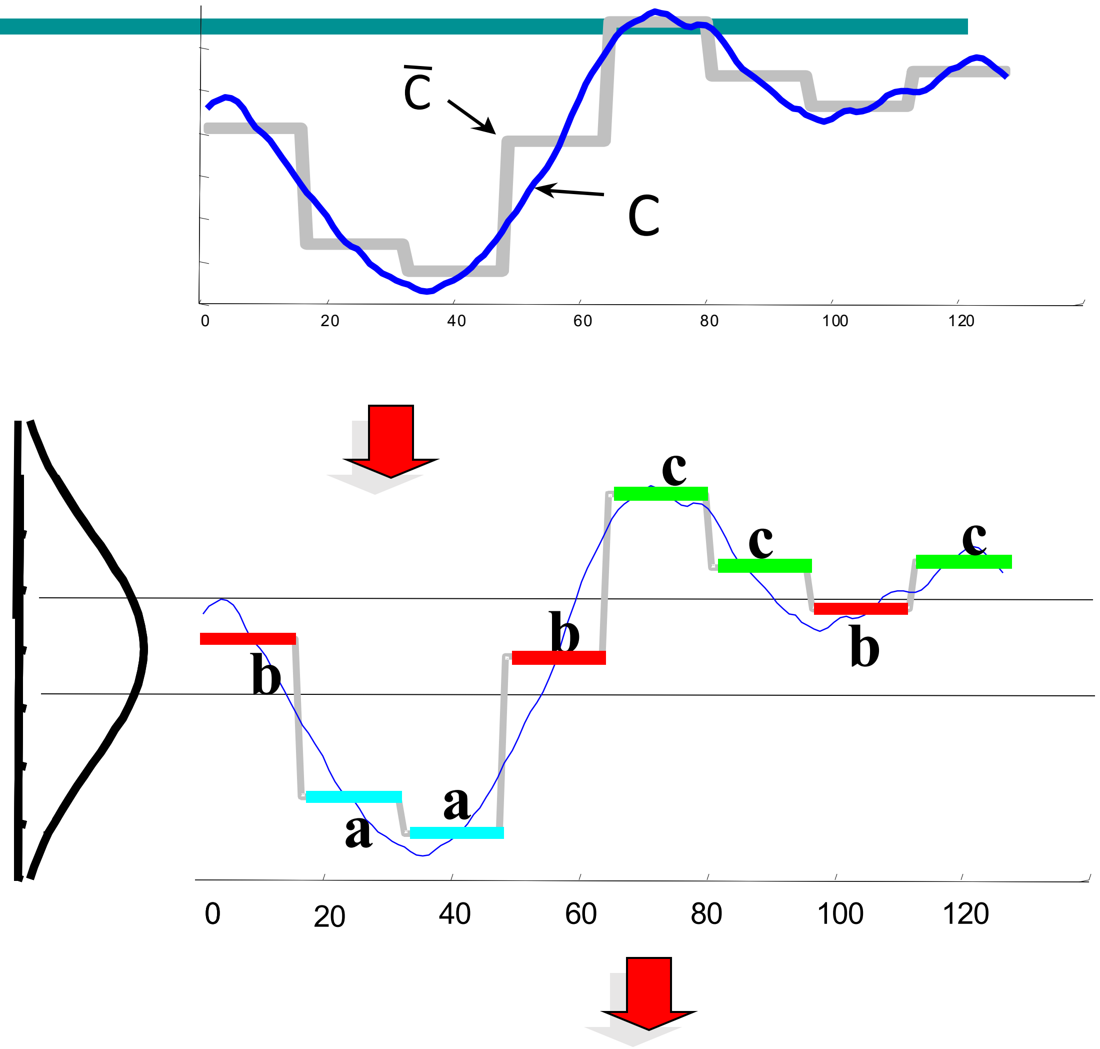
- SAX converts a TS into a discrete format using a small alphabet size such that every part of the representation contributes about the same amount of information about the shape of the TS.
- First converts the time series to PAA representation, then convert the PAA to symbols.



baabccbc

How do we obtain SAX?

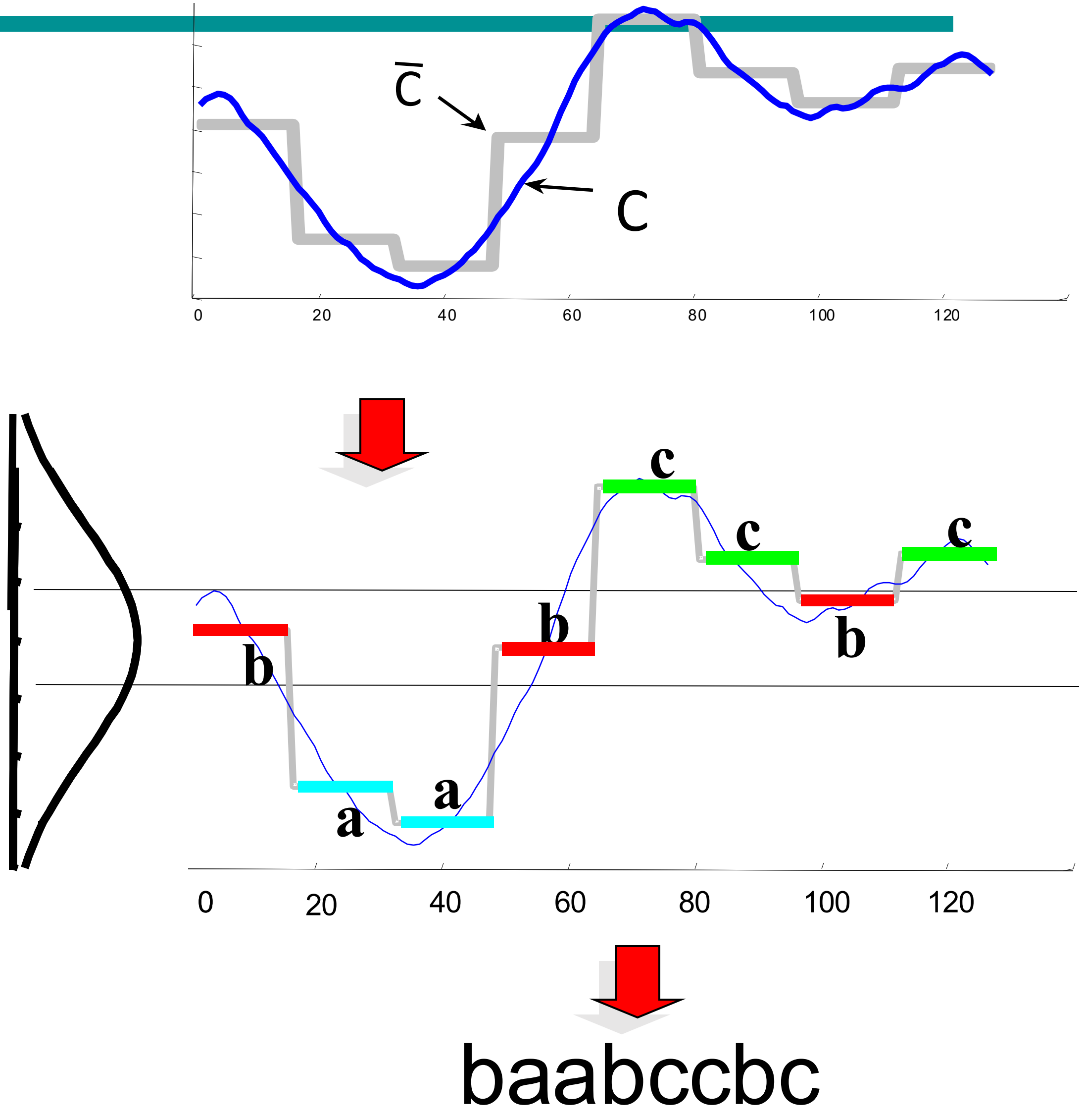
- A time series *T* of length *n* is divided into *w* equal-sized segments; the values in each segment are approximated and replaced by their average.
- Next, we determine the breakpoints that divide the distribution space into *a* equiprobable regions, where *a* is the alphabet size specified by the user.
- The breakpoints are determined such that the probability of a segment falling into any of the regions is approximately the same.



baabccbc

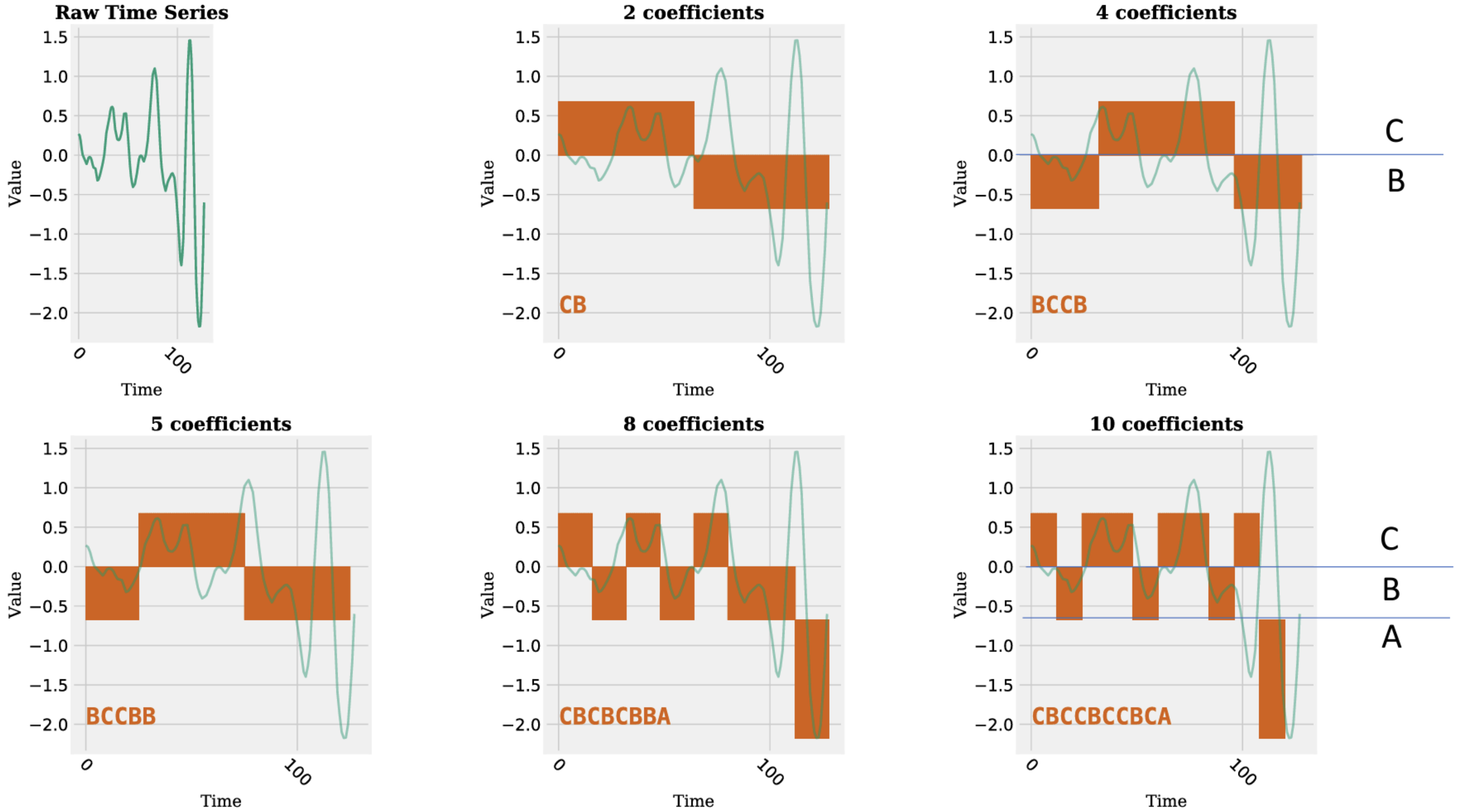
How do we obtain SAX?

- Once the breakpoints are determined, each region is assigned to a symbol and the PAA coefficients are mapped with the symbol corresponding to the region 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.

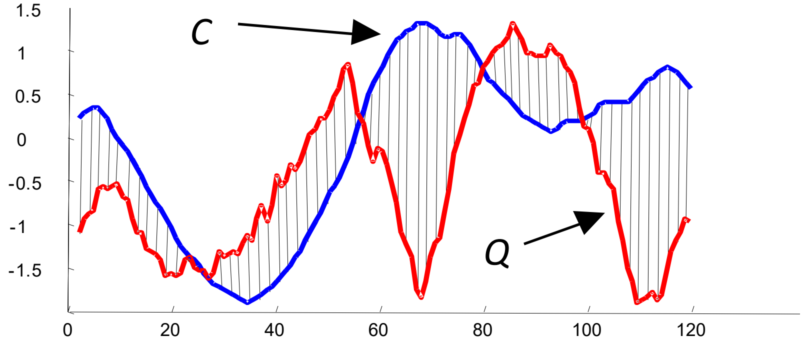


baabccbc

SAX - Example


Raw Time Series
2 coefficients
4 coefficients
5 coefficients
8 coefficients
10 coefficients
Value
Time
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0
100
CB
BCCB
BCCBB
CBCBCBBA
CBCCBCCBCA
C
B
C
B
A

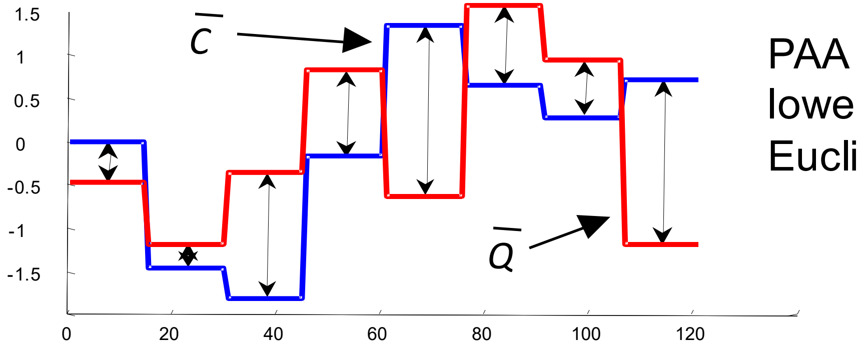
Distances and Approximations



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

Euclidean Distance

$$DR(\overline{Q},\overline{C}) \equiv \sqrt{\tfrac{n}{w}}\sqrt{\sum_{i=1}^{w} (\overline{q}_i - \overline{c}_i)^2}$$



PAA distance lower-bounds the Euclidean Distance

$\hat{C}$ = **baabccbc**

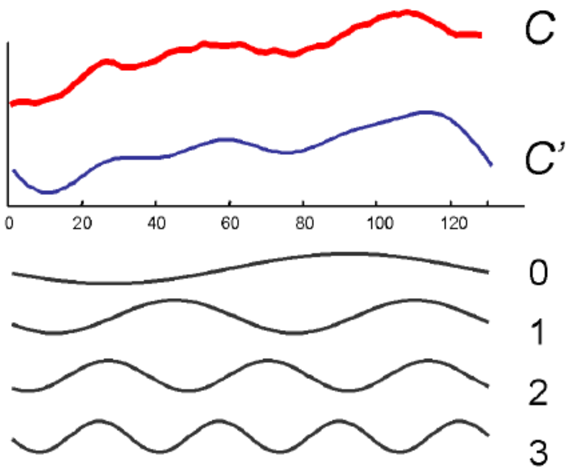
$\hat{Q}$ = **babcacca**

$$MINDIST(\hat{Q},\hat{C}) \equiv \sqrt{\tfrac{n}{w}}\sqrt{\sum_{i=1}^{w} \left(dist(\hat{q}_i,\hat{c}_i)\right)^2}$$

dist() can be implemented using a table lookup or by using for each symbol the mean value of its region.

Discrete Fourier Transform (DFT)

- Represent a TS of length *n* as a linear combination of *w* smooth periodic sinusoidal series (i.e., sines and cosines).
- Each wave is represented by a Fourier coefficient
- This representation is called Frequency Domain
- The DFT concentrates most of its energy in the first few Fourier coefficients
- Low-pass filter: approximate a TS by its first *w* Fourier coefficients.

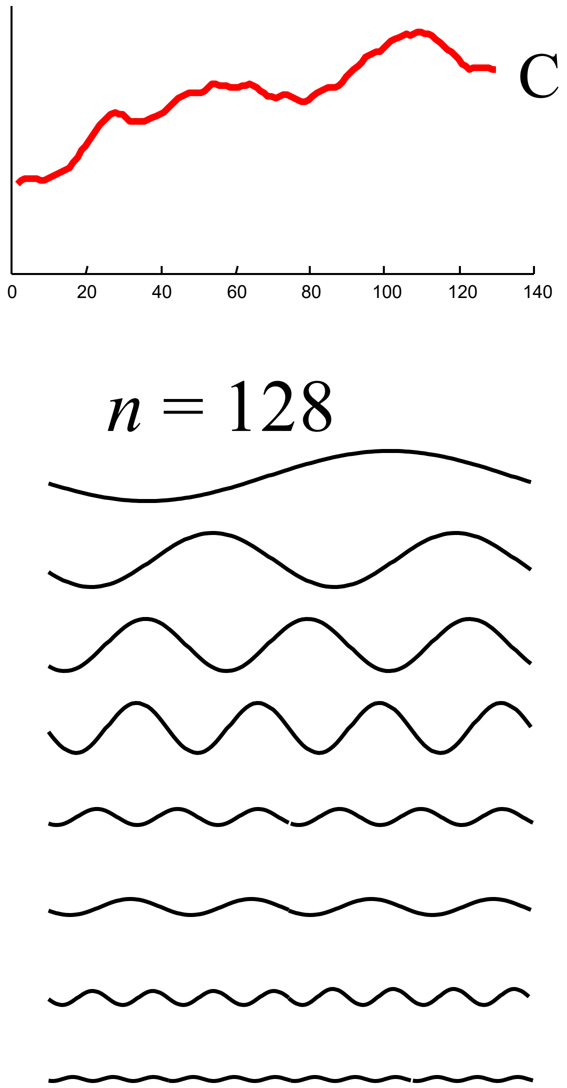


How do we obtain DFT?

- The DFT decomposes a TS *T* of length *n* into a sum of *n* orthogonal basis functions using sinusoid waves represented with two numbers: amplitude and phase.
- A Fourier coefficient (sinusoid wave) is represented by the complex number: X_u = $(real_u, imag_u)$ $u = 0, 1, \ldots, n\text{-}1$
- where the amplitude $A_u = \sqrt{Re(X_u)^2 + Im(X_u)^2}$ and the phase $\varphi_u = atan^2(Re(X_u)^2, Im(X_u)^2)$
- The *n-th* point DFT of a TS $T = \{x_1, \ldots, x_n\}$ *is given by*
- $DFT(T) = X_0, \ldots, X_{n-1} = \{(real_0, imag_0), \ldots, (real_{n-1}, imag_{n-1})\}$
- with $X_u = \frac{1}{n} \sum_{i=1}^{n} x_i \, e^{-\frac{j2\pi i}{n}}$ $u \in [0, n)$, $j = \sqrt{-1}$
- The first Fourier coefficient ($X_0 = \frac{1}{n} \sum_{i=1}^{n} x_i \, e^{0}$) is equal to the mean of a TS and can be discarded to obtain offset invariance, i.e., level removal.

An Example of DFT

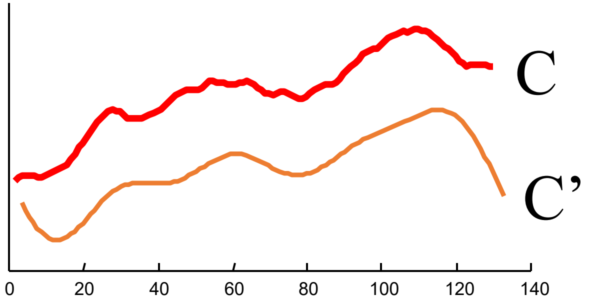
- We can decompose the data into 64 pure sine waves using the Discrete Fourier Transform (just the first few sine waves are shown).
- The Fourier Coefficients are reproduced as a column of numbers (just the first 30 or so coefficients are shown).
- Note that at this stage we have not done dimensionality reduction, we have merely changed the representation...



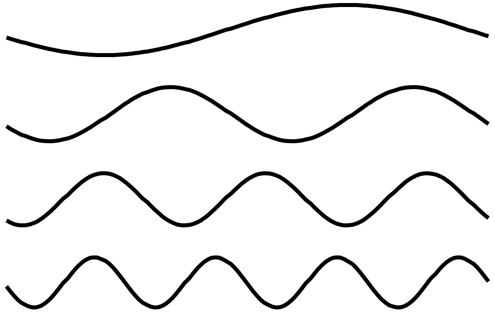
Fourier Coefficients	Raw Data
1.5698	0.4995
1.0485	0.5264
0.7160	0.5523
0.8406	0.5761
0.3709	0.5973
0.4670	0.6153
0.2667	0.6301
0.1928	0.6420
0.1635	0.6515
0.1602	0.6596
0.0992	0.6672
0.1282	0.6751
0.1438	0.6843
0.1416	0.6954
0.1400	0.7086
0.1412	0.7240
0.1530	0.7412
0.0795	0.7595
0.1013	0.7780
0.1150	0.7956
0.1801	0.8115
0.1082	0.8247
0.0812	0.8345
0.0347	0.8407
0.0052	0.8431
0.0017	0.8423
0.0002	0.8387
…	…

An Example DFT

- … however, note that the first few sine waves tend to be the largest (equivalently, the magnitude of the Fourier coefficients tend to decrease as you move down the column).
- We can therefore truncate most of the small coefficients with little effect.



$n = 128$



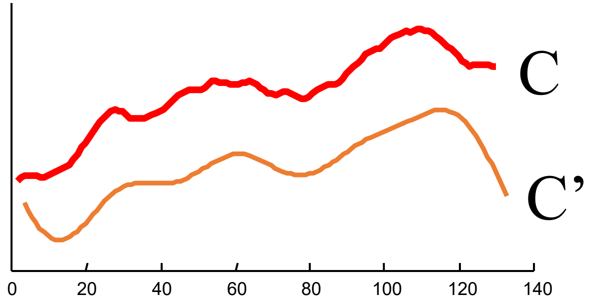
Truncated Fourier Coefficients	Fourier Coefficients	Raw Data
1.5698	1.5698	0.4995
1.0485	1.0485	0.5264
0.7160	0.7160	0.5523
0.8406	0.8406	0.5761
0.3709	0.3709	0.5973
0.4670	0.4670	0.6153
0.2667	0.2667	0.6301
0.1928	0.1928	0.6420
	0.1635	0.6515
	0.1602	0.6596
	0.0992	0.6672
	0.1282	0.6751
	0.1438	0.6843
	0.1416	0.6954
	0.1400	0.7086
	0.1412	0.7240
	0.1530	0.7412
	0.0795	0.7595
	0.1013	0.7780
	0.1150	0.7956
	0.1801	0.8115
	0.1082	0.8247
	0.0812	0.8345
	0.0347	0.8407
	0.0052	0.8431
	0.0017	0.8423
	0.0002	0.8387
	…	…

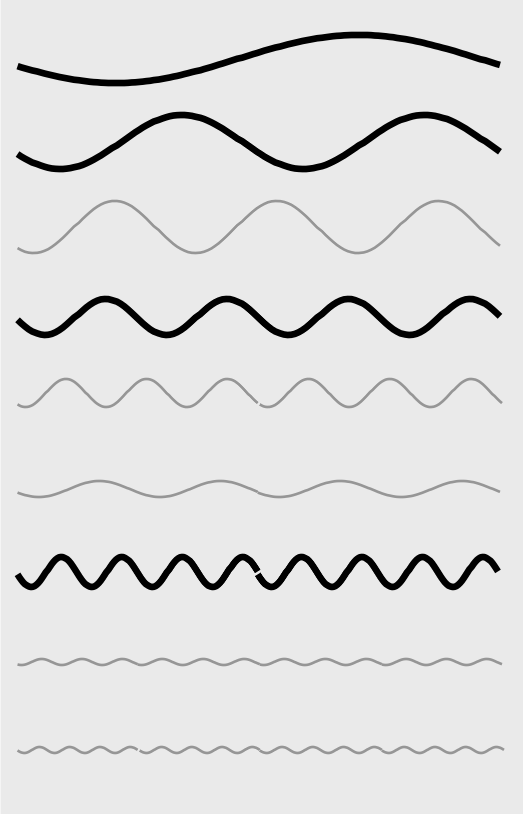
$n = 128$
$N = 8$
$C_{\text{ratio}} = 1/16$

We have discarded $\frac{15}{16}$ of the data.

An Example of DFT

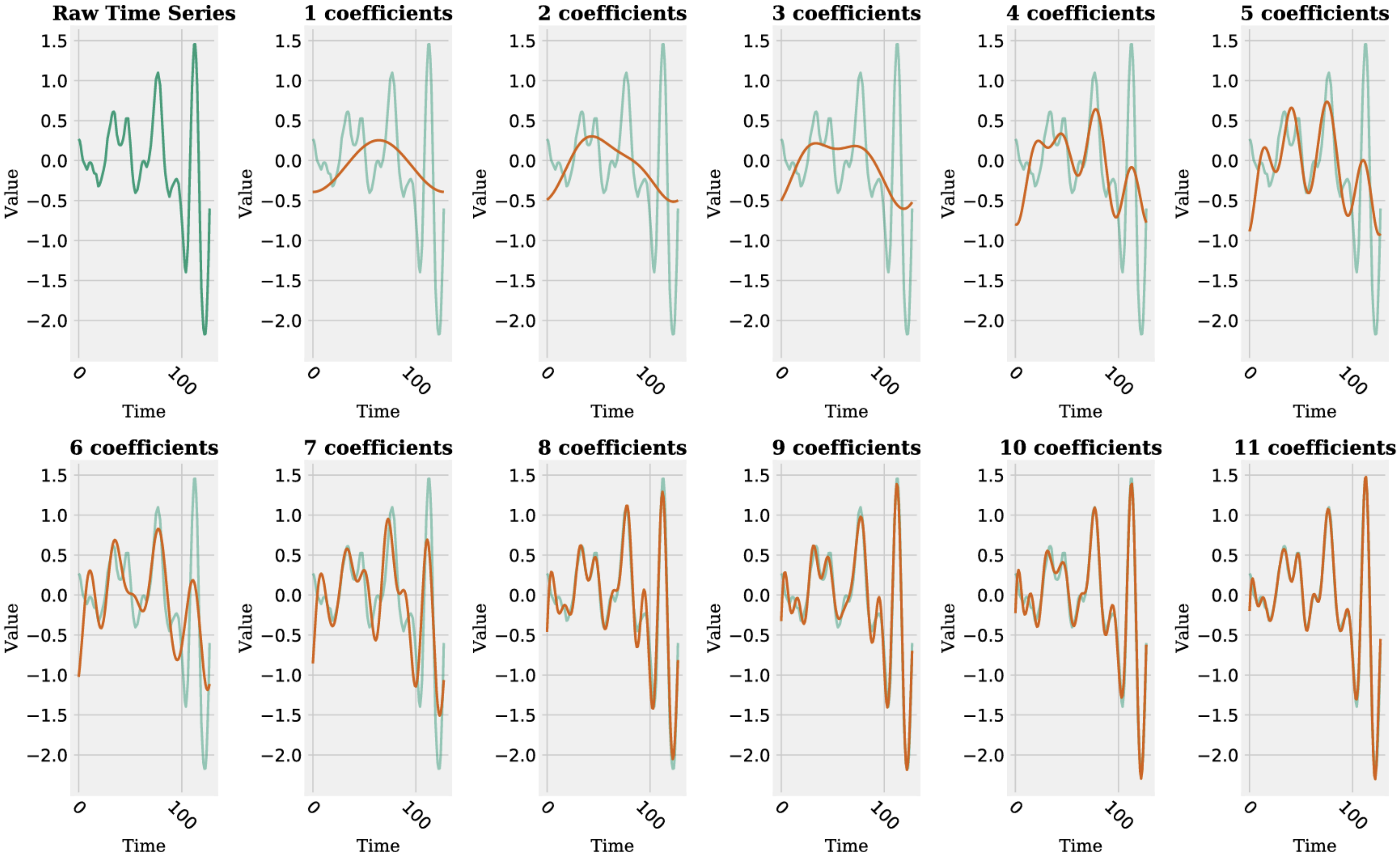
- Instead of taking the first few coefficients, we could take the best coefficients
- This can help greatly in terms of approximation quality but makes indexing hard.
- Note this applies also to Wavelets





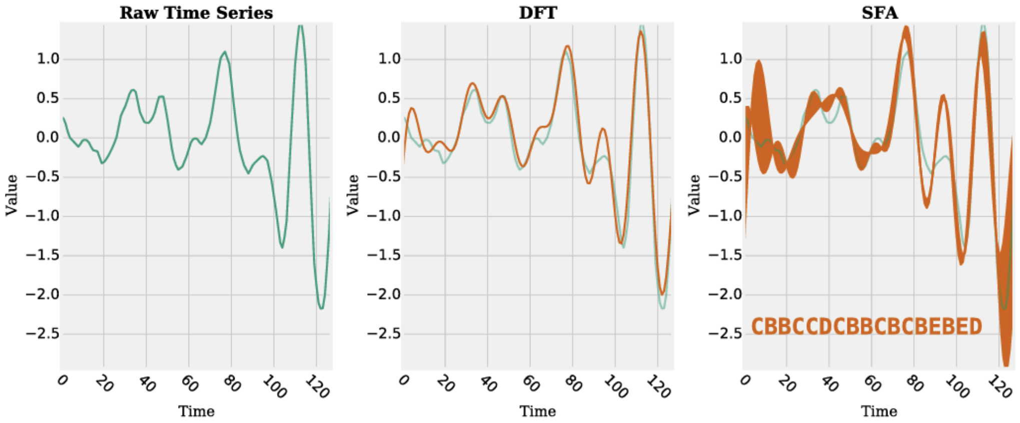
Truncated Fourier Coefficients	Fourier Coefficients	Raw Data
1.5698	1.5698	0.4995
1.0485	1.0485	0.5264
0.7160	0.7160	0.5523
0.8406	0.8406	0.5761
0.3709	0.3709	0.5973
0.4670	0.4670	0.6153
0.2667	0.2667	0.6301
0.1928	0.1928	0.6420
	0.1635	0.6515
	0.1602	0.6596
	0.0992	0.6672
	0.1282	0.6751
	0.1438	0.6843
	0.1416	0.6954
	0.1400	0.7086
	0.1412	0.7240
	0.1530	0.7412
	0.0795	0.7595
	0.1013	0.7780
	0.1150	0.7956
	0.1801	0.8115
	0.1082	0.8247
	0.0812	0.8345
	0.0347	0.8407
	0.0052	0.8431
	0.0017	0.8423
	0.0002	0.8387
	…	…

DFT - Example


Raw Time Series
1 coefficients
2 coefficients
3 coefficients
4 coefficients
5 coefficients
6 coefficients
7 coefficients
8 coefficients
9 coefficients
10 coefficients
11 coefficients
Value
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
Time
0
100

Symbolic Fourier Approximation (SFA)

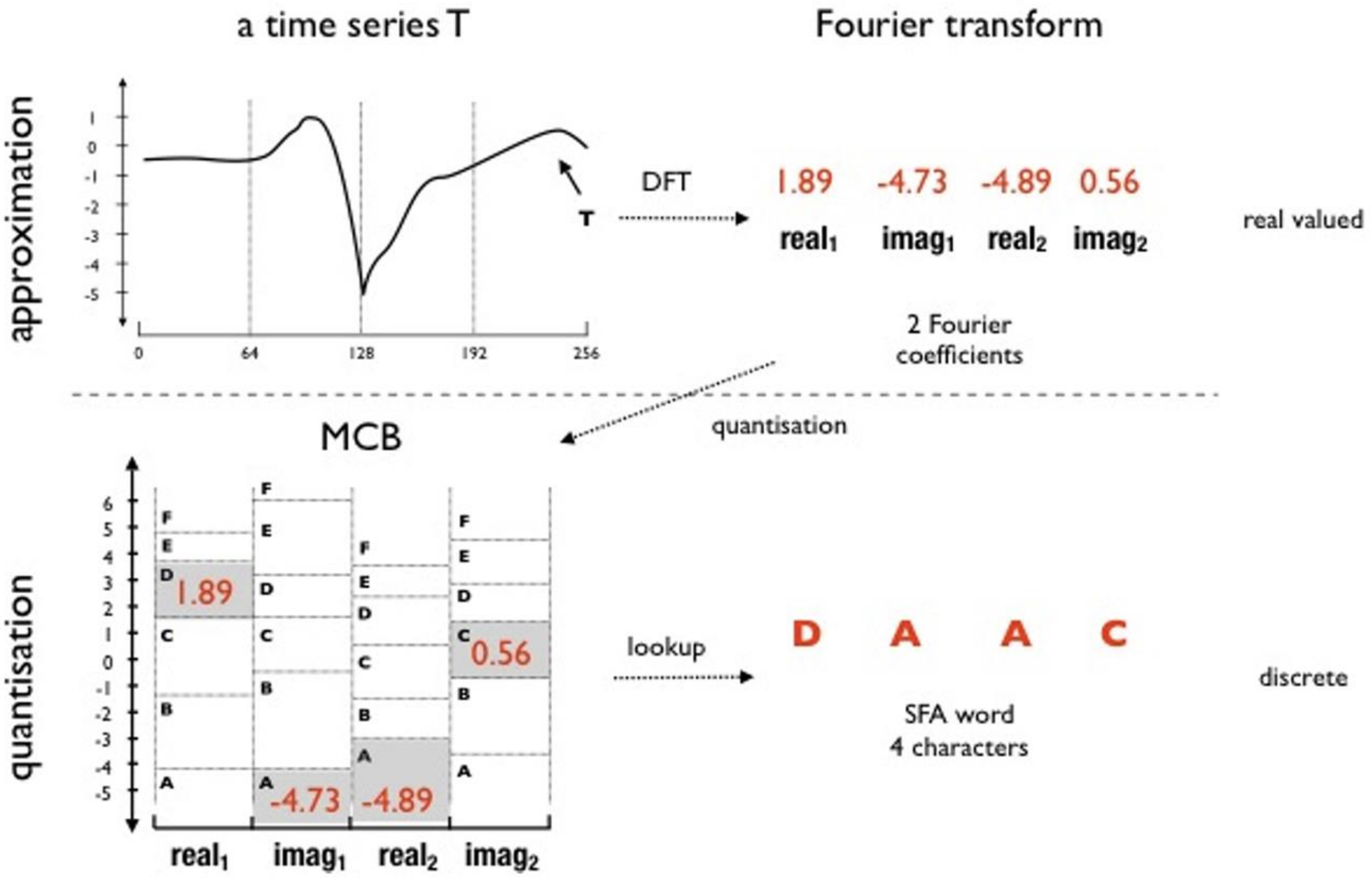
- SFA represents a TS with a word
- SFA is composed of
  - a) approximation using the Fourier transform
  - b) a data adaptive discretization
- The discretization intervals are learned from the Fourier transformed data distribution rather than using fixed intervals with Multiple Coefficient Binning (MCB)



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
[...]	[...]	[...]

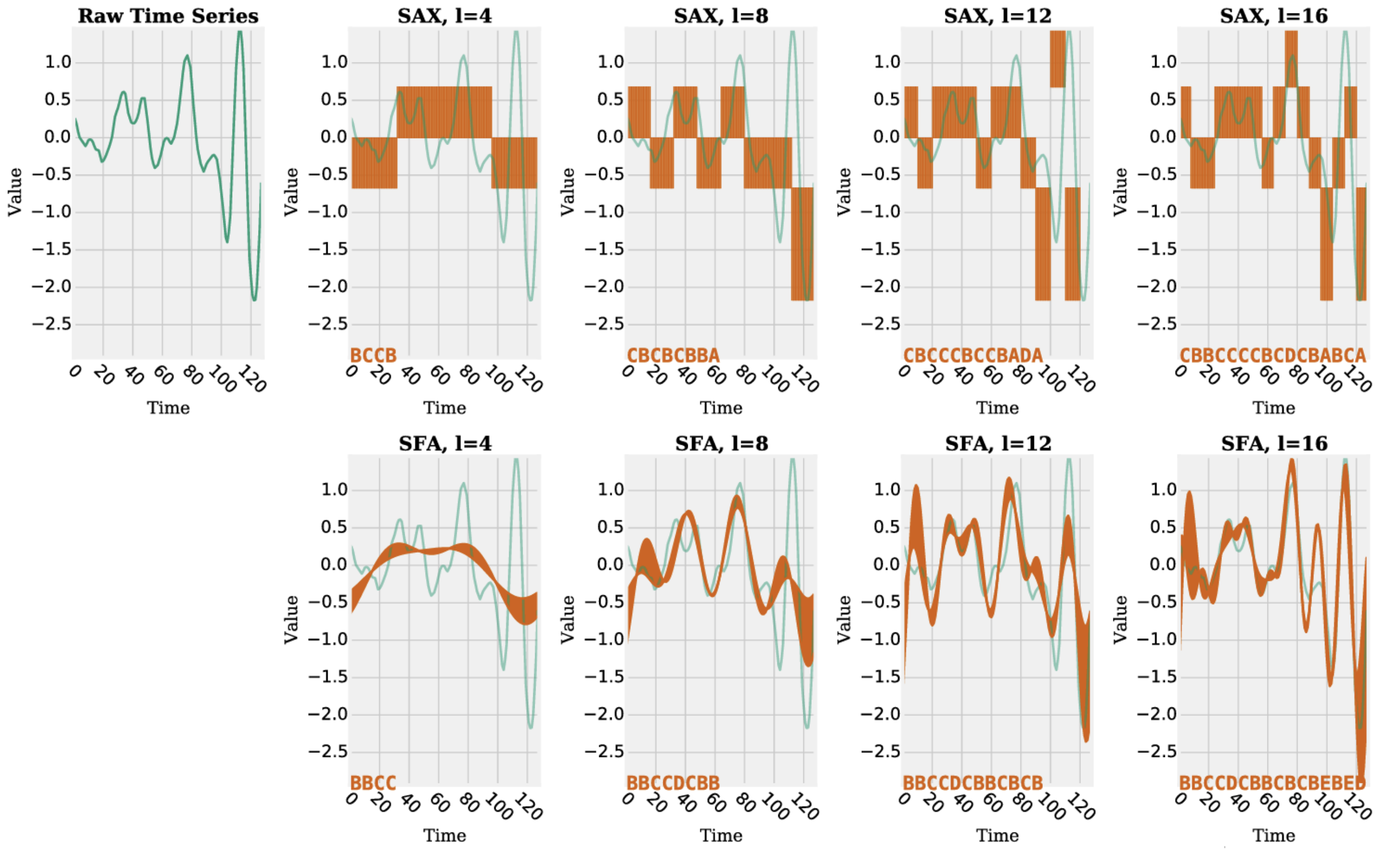
SFA Discretization

- The objective of MCB is to minimize the loss of information introduced by discretization.
- This is achieved by applying a discretization for each coefficient.
- MCB uses different breakpoints for each symbol on each coefficient.



Comparison of SFA and SAX

- Roughly we can say
  - SAX = PAA + Discretization
  - SFA = DFT + Discretization
- Approximation cause a loss of information
- Discretization augments the level of information loss.
- The higher the number of symbols and the alphabet size, the more exact is the representation.

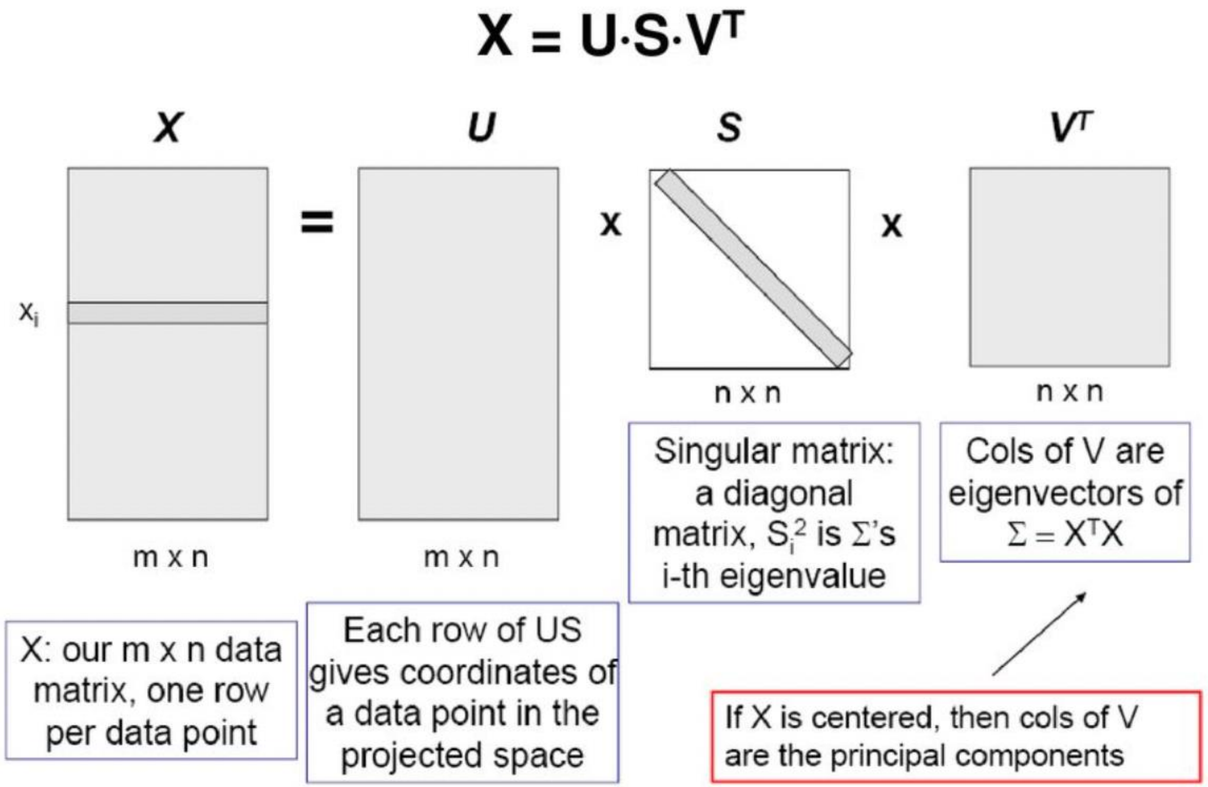


Properties of Symbolic Representations

- Noise removal: discretization
- String representations: allows for string domain algorithms like hashing or the bag-of-words to be applied
- Dimensionality reduction: allow for indexing high dimensional data
- Storage reduction: sequences have a much lower memory footprint than real-valued time series

Singular Value Decomposition (SVD)

- SVD is a factorization method that decomposes a matrix into three other matrices: U, S, and V^T (transpose of V):
- $X = U\ S\ V^T$
- Input matrix X (m x n)
- U (m x m) contains orthogonal columns that represent the left singular vectors.
- S (m x n) is a diagonal matrix containing the singular values.
- V^T (n x n) contains orthogonal rows representing the right singular vectors.



X: our m x n data matrix, one row per data point

Each row of US gives coordinates of a data point in the projected space

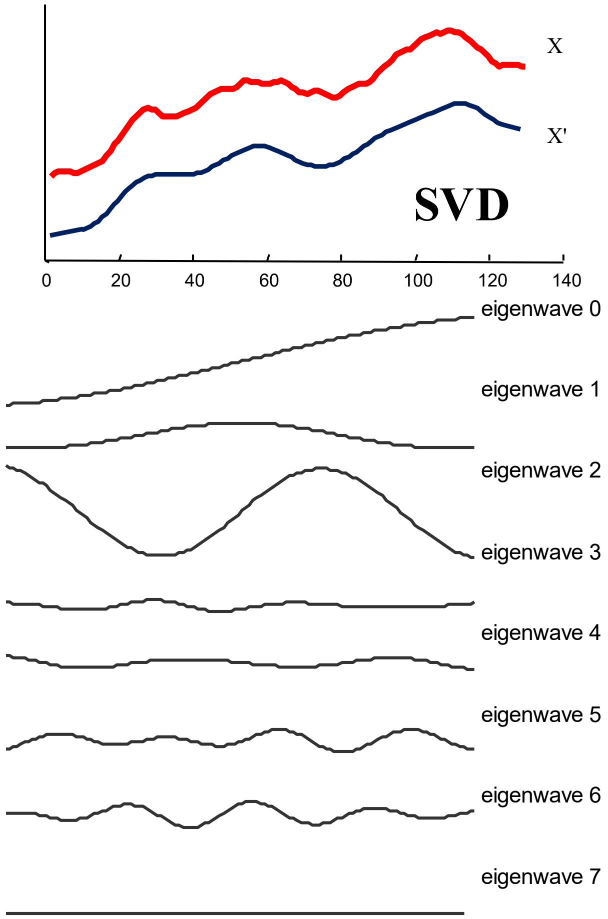
Singular matrix: a diagonal matrix, S_i^2 is Σ's i-th eigenvalue

Cols of V are eigenvectors of $\Sigma = X^TX$

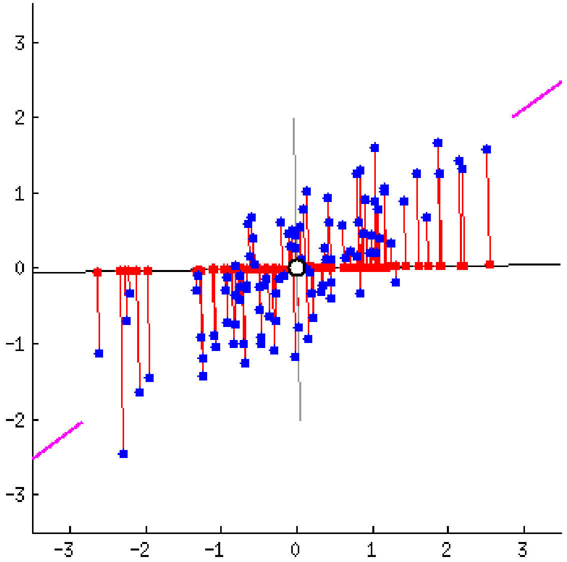
If X is centered, then cols of V are the principal components

SVD for Time Series Approximations

- SVD is similar to DFT and SFT in that it represents the shape in terms of a linear combination of basis shapes.
- DFT and SFT are ***individual approximations*** as they examine one TS at a time. They are completely independent of the rest of the data.
- SVD is a ***global approximations*** as the entire dataset is examined and is then rotated such that the first axis has the maximum possible variance, the second axis has the maximum possible variance orthogonal to the first, the third axis has the maximum possible variance orthogonal to the first two, etc.
- The global nature of SVD is both a weakness and a strength.



Principal Component Analysis (PCA)



- PCA is a statistical procedure that aims to transform data into a new coordinate system where the axes are the principal components. These components are orthogonal and capture the maximum variance in the data.

1. **Standardize the Data**: Normalize the data *X* to have zero mean and unit variance matrix *C*.
2. **Calculate Covariance Matrix**: Calculate the covariance matrix *Σ* = C^TC of the standardized data.
3. **Compute Eigenvectors and Eigenvalues**: Compute the eigenvectors and eigenvalues of the covariance matrix *Σ*.
4. **Select Principal Components**: Sort eigenvalues in descending order and choose the top-k eigenvalues to form principal components.
5. **Transform Data**: Project the original data *X* into the principal components to create a lower-dimensional representation.

Relationships between SVD and PCA

Dealing with data

- PCA primarily deals with the covariance structure of the data.
- SVD does not rely on a covariance matrix. It is a factorization that decomposes the original data without computing covariance.

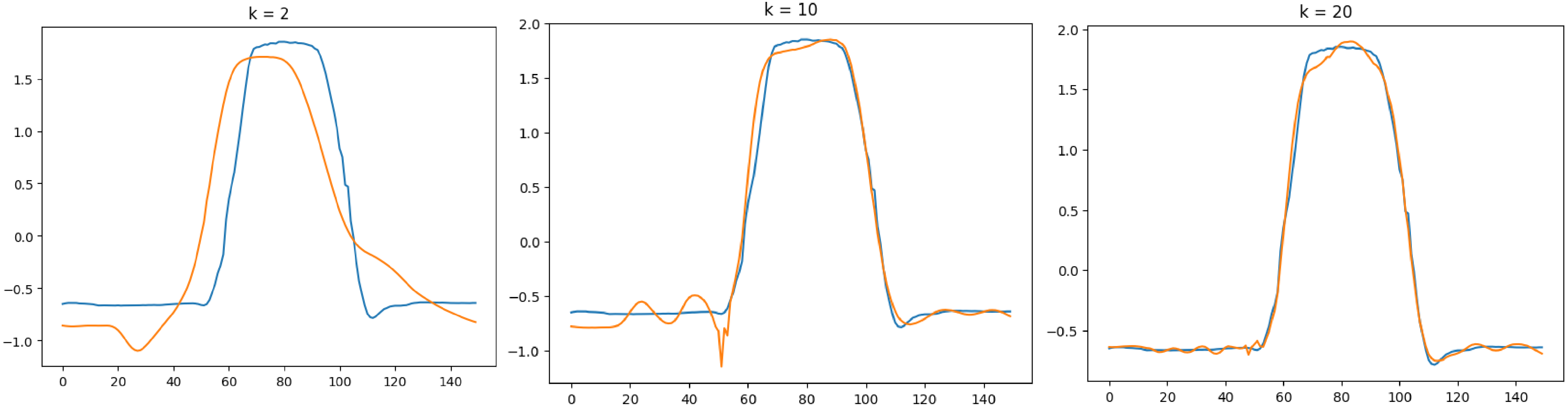
Computations

- Both PCA and SVD involve eigen-decomposition.
- PCA performs eigen-decomposition on the covariance matrix of the data which is a square symmetric matrix of size *m x m* where *m* is the number of time stamp.
- SVD performs eigen-decomposition on the data matrix itself of size *n x m* where *n* is the number of TS and *m* is the number of number of time stamps.

Relationships between SVD and PCA

- **PCA is a specific application of SVD**, primarily used for dimensionality reduction, while SVD is a more general matrix decomposition technique with broader applications in linear algebra and data analysis.
- PCA can be solved using SVD
- PCA focuses on the covariance structure and tries to maximize variance along orthogonal axes
- SVD focuses on matrix factorization and can handle cases where data is missing.
- **From an application perspective they are used interchangeably**.

PCA - Example


k = 2
1.5
1.0
0.5
0.0
−0.5
−1.0
0
20
40
60
80
100
120
140
k = 10
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
0
20
40
60
80
100
120
140
k = 20
2.0
1.5
1.0
0.5
0.0
−0.5
0
20
40
60
80
100
120
140

Remarks on Approximations

- None of the representations can be superior for all tasks.
- No research has proved how one should choose the best representation for the problem at hand and data of interest.
- The literature is not even consistent on nomenclature.
    - Example: Piecewise Aggregate Approximation
        - Piecewise Flat Approximation (Faloutsos et al., 1997)
        - Piecewise Constant Approximation
        - Segmented Means

Approximations, Distances and Normalizations

- It does not make any sense to use a distance function accounting for time like DTW if a Time-Independent approximation is used.
- Thus, do not use DTW after DTF, SFA, SVD, PCA.
- It does make sense to use DTW after Time-Dependent approximations.
- Thus, you can use DTW after PAA or SAX.
- Normalizations can be applied before and/or after approximations depending on the objective of your TSA task.
- Time series normalizations does not make any sense after a Time-Independent approximation.
- In that case traditional “column-wise” normalizations like Min-Max scaling or Z-Score normalization should be used.

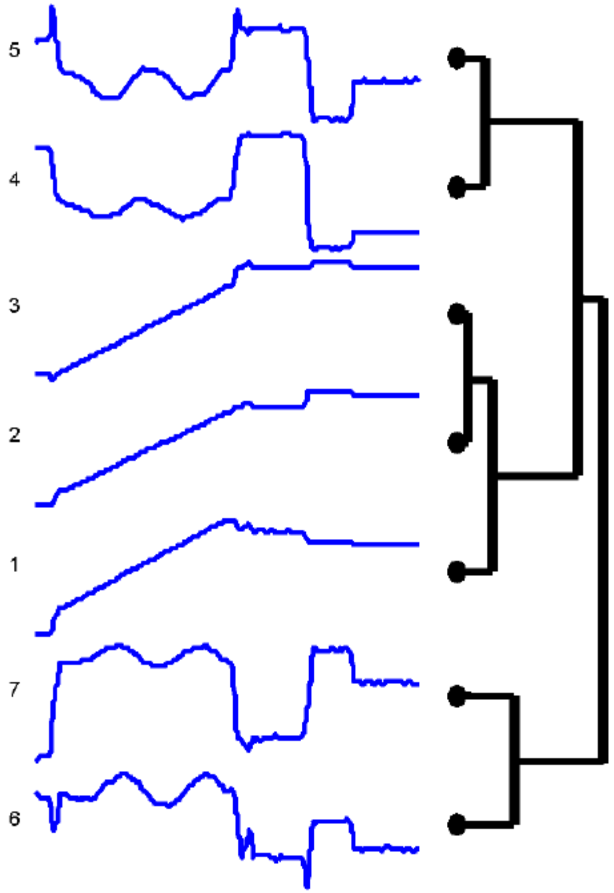
Clustering

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

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 K-Means (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.

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.
- ***Approximated-based clustering***: approximate 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.

References

- Selective review of offline change point detection methods. Truong, C., Oudre, L., & Vayatis, N. (2020). Signal Processing, 167, 107299.
- Time Series Analysis and Its Applications. Robert H. Shumway and David S. Stoffer. 4th edition.(https://www.stat.pitt.edu/stoffer/tsa4/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.