

# DATA MINING 2

## Time Series – Matrix Profile, Motifs & Discords

---

Riccardo Guidotti

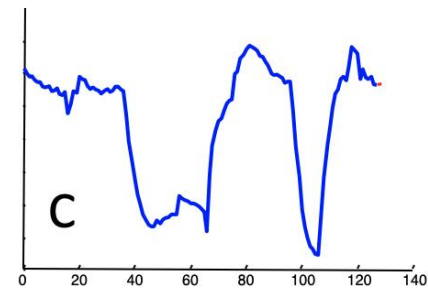
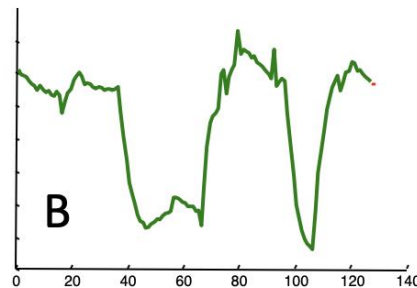
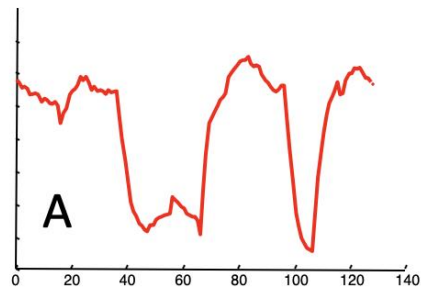
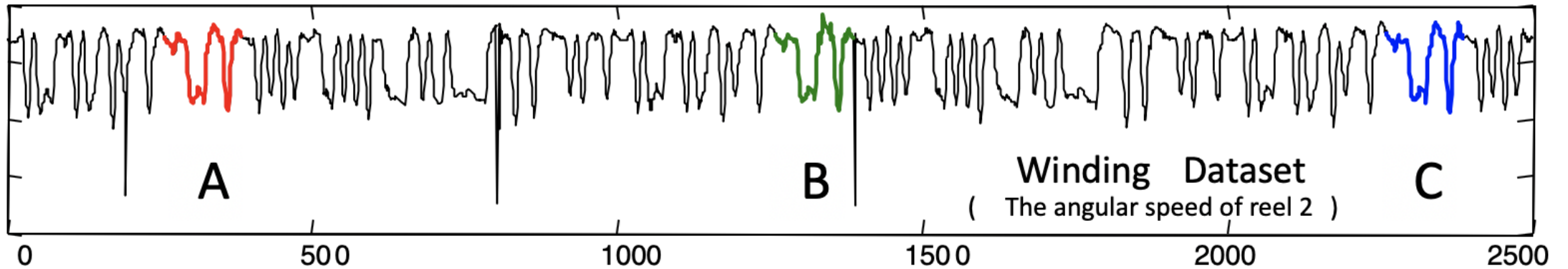
a.a. 2022/2023

Slides edited from Keogh Eamonn's tutorial



# Time Series Motif Discovery

- Finding repeated patterns, i.e., pattern mining.
- Are there any repeated patterns, of length  $m$  in the TS?



# Why Finding Motifs?

---

- Mining **association rules** in TS requires the discovery of motifs. These are referred to as primitive shapes and frequent patterns.
- Several **TS classifiers** work by constructing typical prototypes of each class. These prototypes may be considered motifs.
- Many **TS anomaly detection** algorithms consist of modeling normal behavior with a set of typical shapes (which we see as motifs) and detecting future patterns that are dissimilar to all typical shapes.

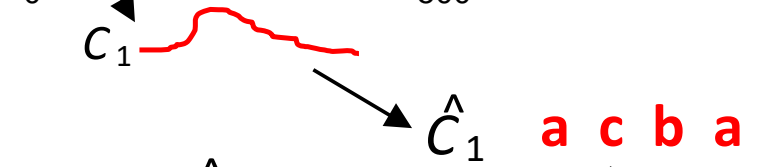
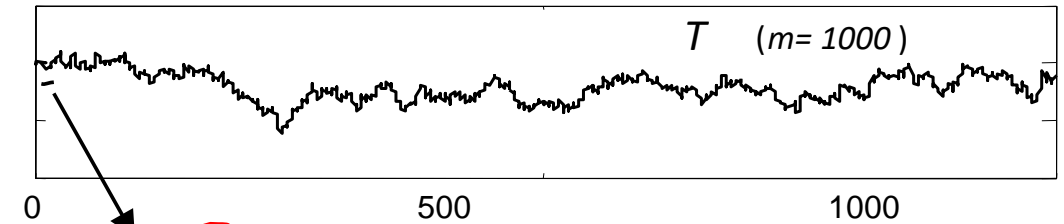
# How do we find Motifs?

---

- Given a predefined motif length  $m$ , a brute-force method searches for motifs from all possible comparisons of subsequences.
- It is obviously very slow and computationally expensive.
- The most referenced algorithm is based on a hot idea from bioinformatics, random projection\* and the fact that SAX allows use to lower bound discrete representations of TSs.
- J Buhler and M Tompa. Finding motifs using random projections. In RECOMB'01. 2001.

# Motif Discovery with Random Projections

- Assume that we have a time series  $T$  of length 1,000, and a motif of length 16, which occurs twice, at time  $T_1$  and time  $T_{58}$ .



$\hat{S}$

1	<b>a</b>	<b>c</b>	<b>b</b>	<b>a</b>
2	<b>b</b>	<b>c</b>	<b>a</b>	<b>b</b>
⋮	⋮	⋮	⋮	⋮
⋮	⋮	⋮	⋮	⋮
58	<b>a</b>	<b>c</b>	<b>c</b>	<b>a</b>
⋮	⋮	⋮	⋮	⋮
985	<b>b</b>	<b>c</b>	<b>c</b>	<b>c</b>

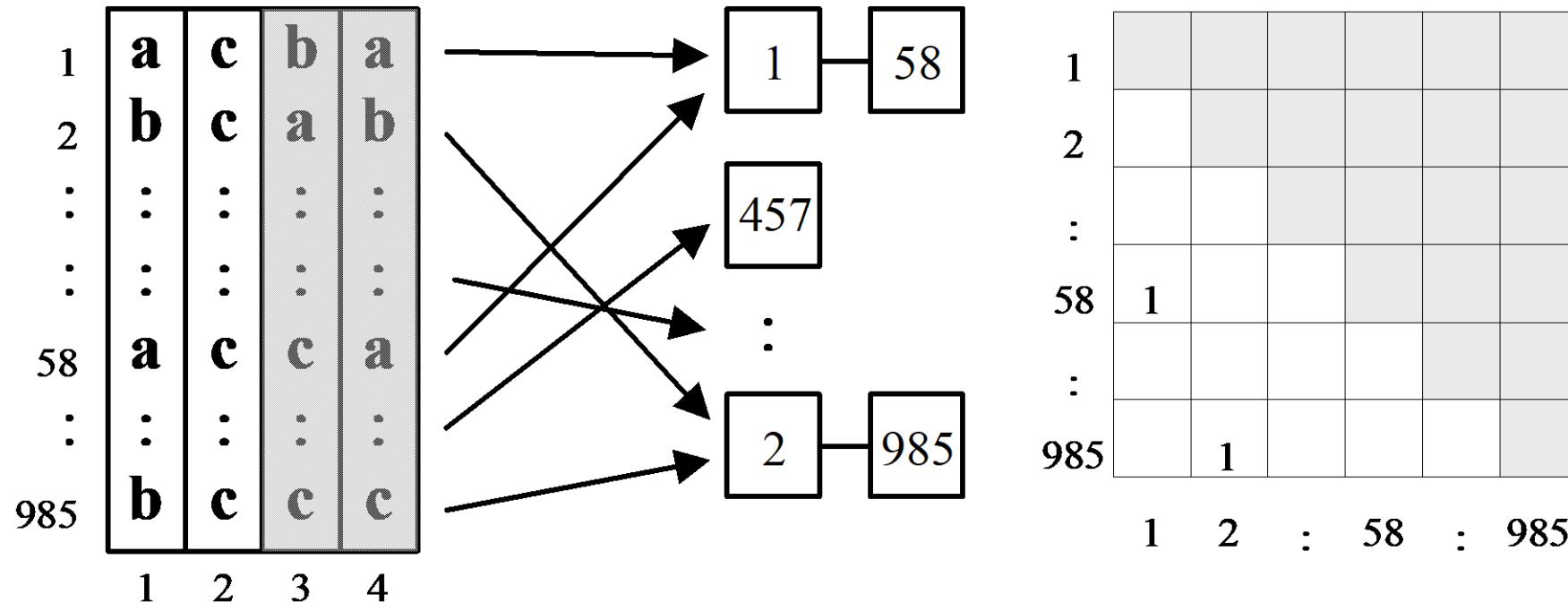
16

$\hat{C}_1$  **a c b a**

$a = 3$  {**a,b,c**} alphabet  
 $n = 16$  motif length  
 $w = 4$  sax window

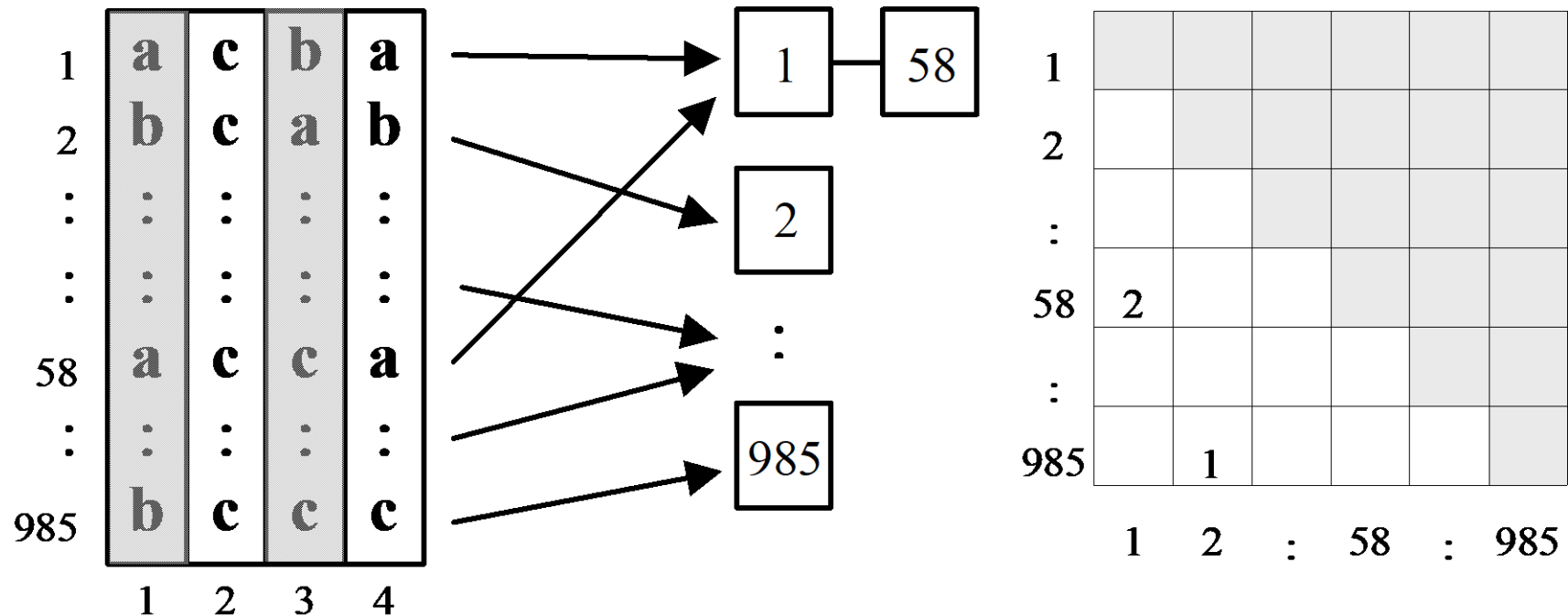
# Motif Discovery with Random Projections

- A mask  $\{1,2\}$  was randomly chosen, so the values in columns  $\{1,2\}$  were used to project matrix into buckets.
- Collisions are recorded by incrementing the appropriate location in the collision matrix.



# Motif Discovery with Random Projections

- A mask  $\{2,4\}$  was randomly chosen, so the values in columns  $\{2,4\}$  were used to project matrix into buckets.
- Once again, collisions are recorded by incrementing the appropriate location in the collision matrix.



# Motif Discovery with Random Projections

- At the end of the random perturbations consider the motifs observing the matrix in decreasing order of occurrences.
- For instance, this matrix indicates a high chance of having a motif starting at positions 1 and 58.
- The problem with this approach is that it is highly dependent from the approximation technique adopted.

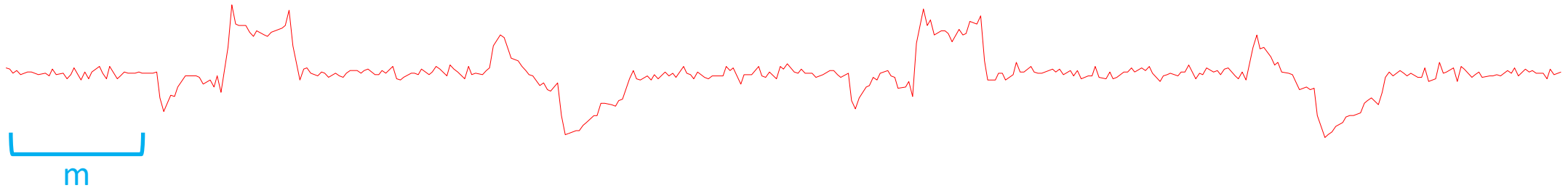
1						
2	2					
:	1	3				
58	<b>27</b>	2	1			
:	3	2	2	1		
985	0	1	2	1	3	
	1	2	:	58	:	985



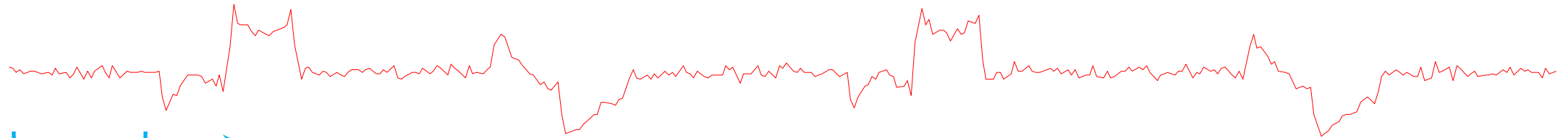
# Matrix Profile

---

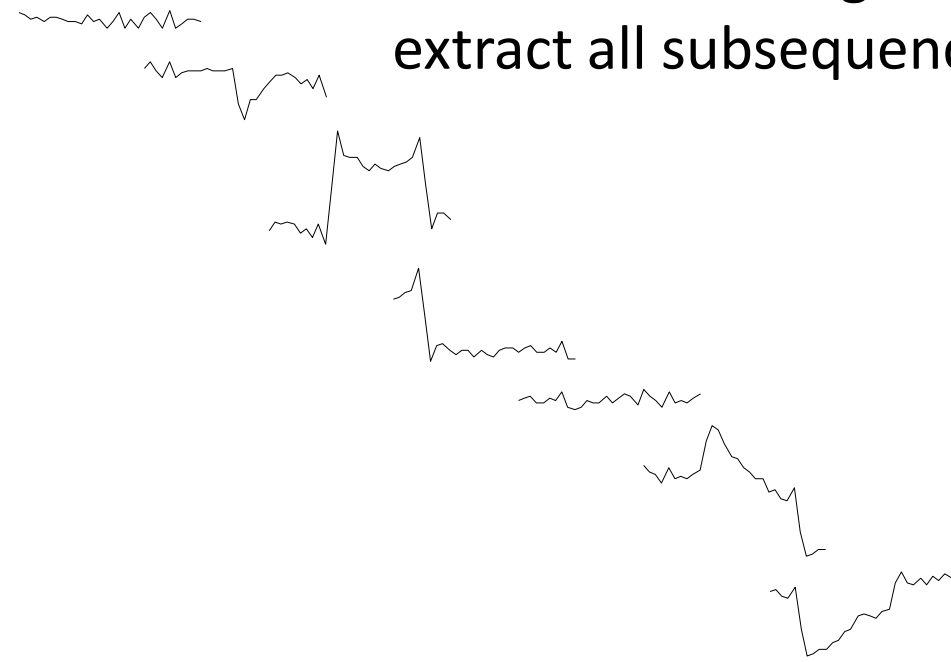
- The Matrix Profile (MP) is a data structure that annotates a TS and can be exploited for many purposes: e.g. efficient Motif Discovery.
- Given a time series,  $T$  and a desired subsequence length,  $m$ .



# Matrix Profile



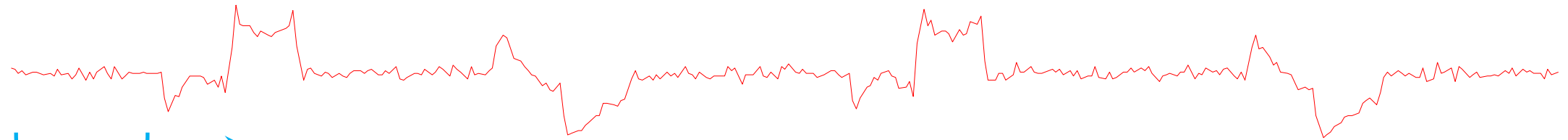
We can use sliding window of length  $m$  to extract all subsequences of length  $m$ .



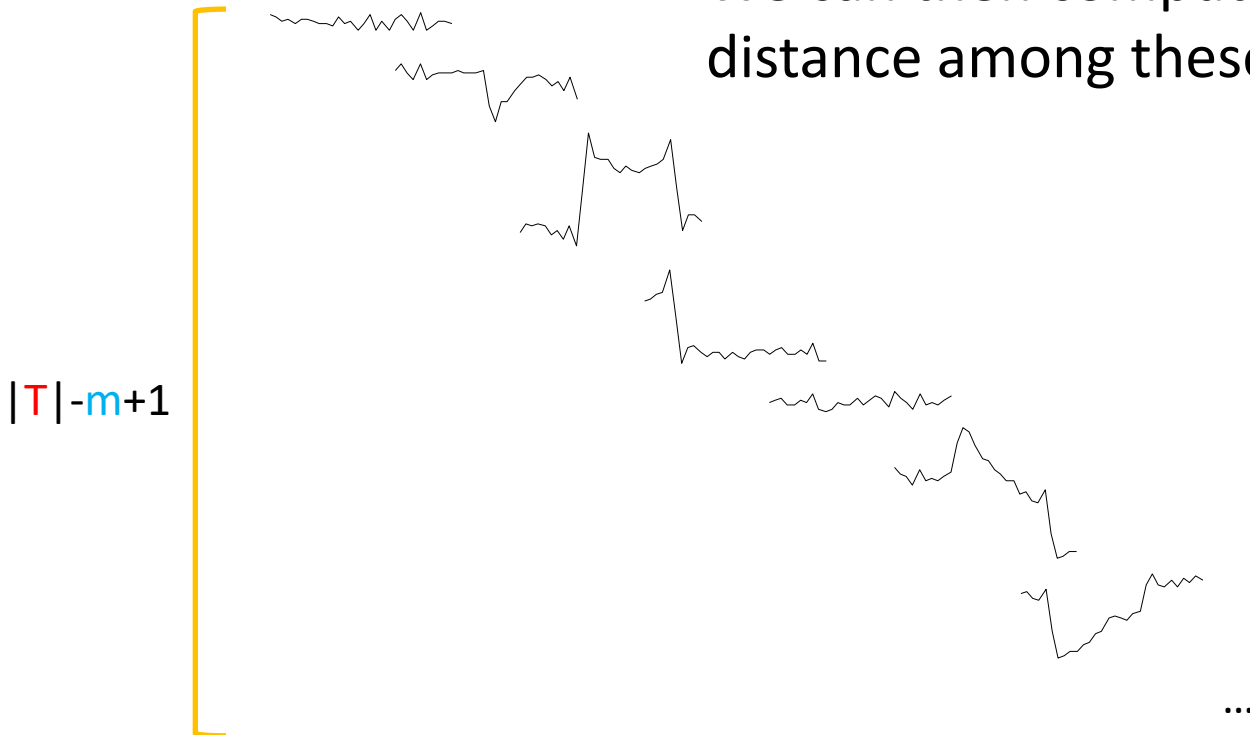
...

$|T| - m + 1$

# Matrix Profile



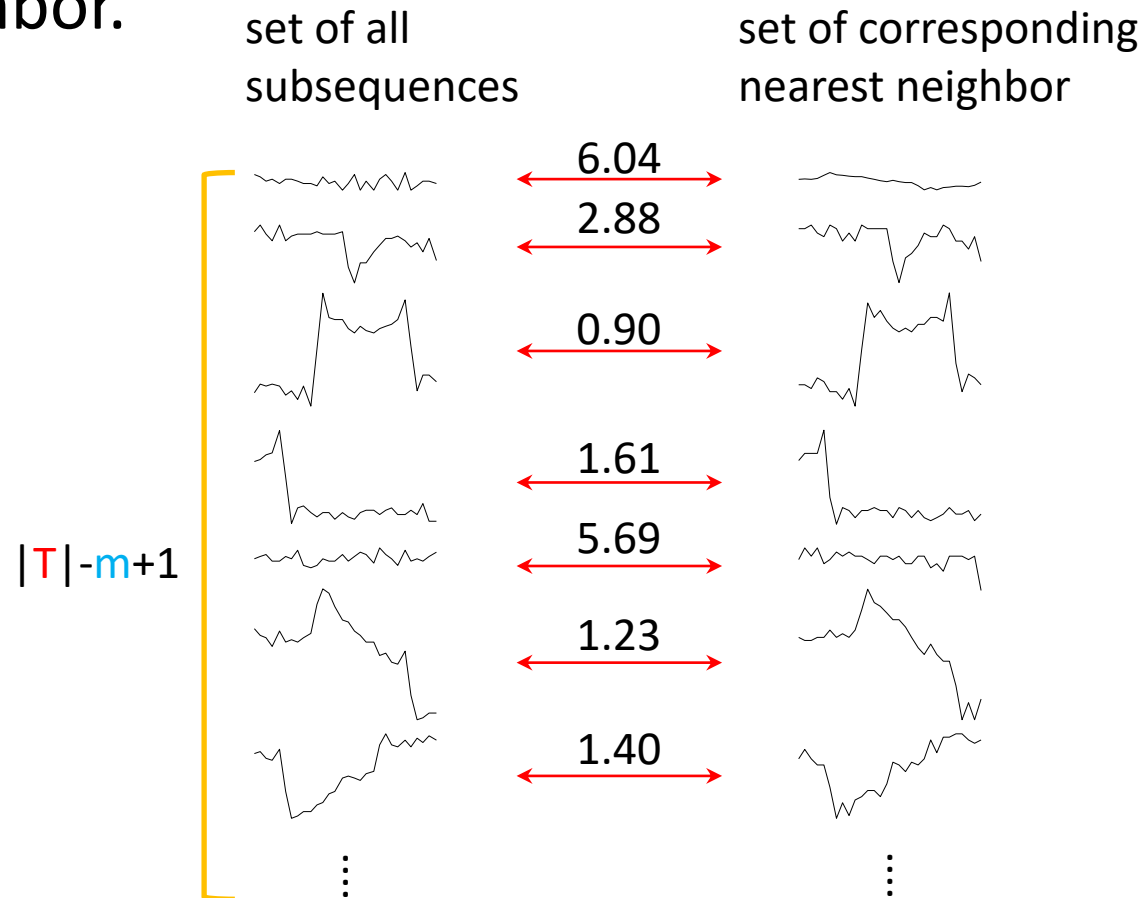
We can then compute the pairwise distance among these subsequences.



0	7.6952	7.7399	...
7.6952	0	7.7106	...
7.7399	7.7106	0	...
...	...	...	...

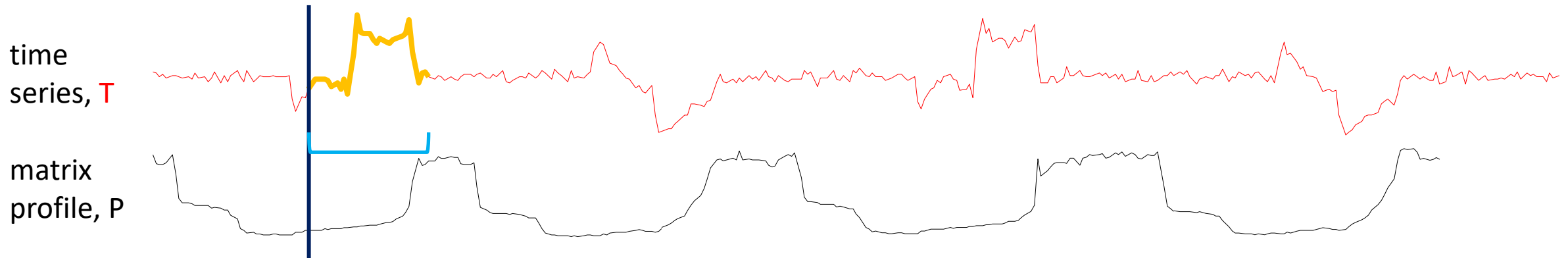
# Matrix Profile

- For each subsequence we keep only the distance with the closest nearest neighbor.



# Matrix Profile

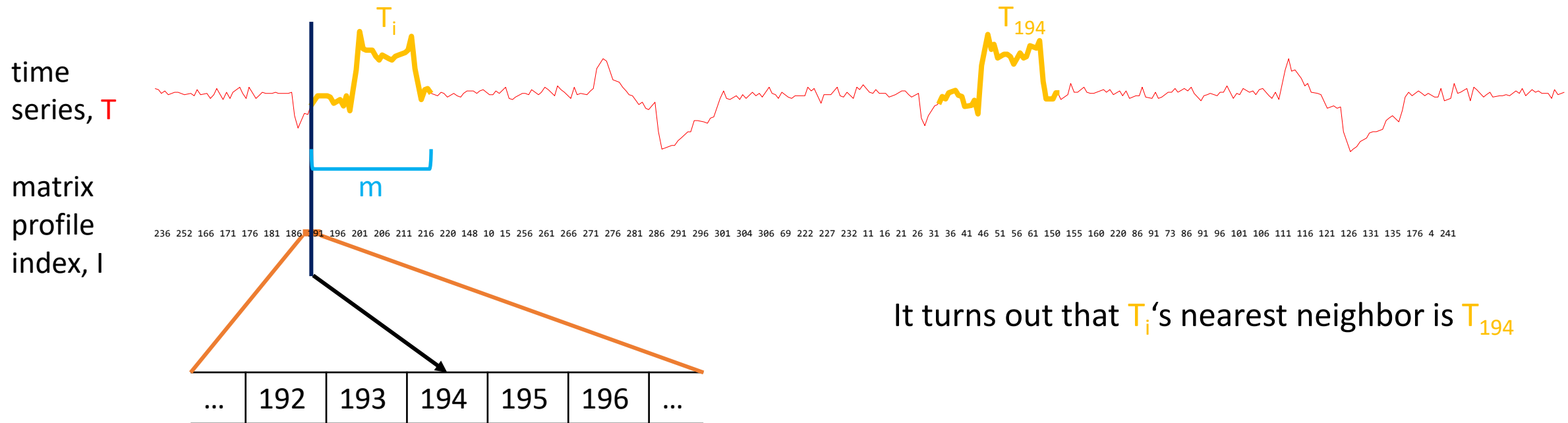
- The distance to the corresponding nearest neighbor of each subsequence can be stored in a vector called **matrix profile P**.



The matrix profile value at location  $i$  is the distance between  $T_i$  and its nearest neighbor

# Matrix Profile

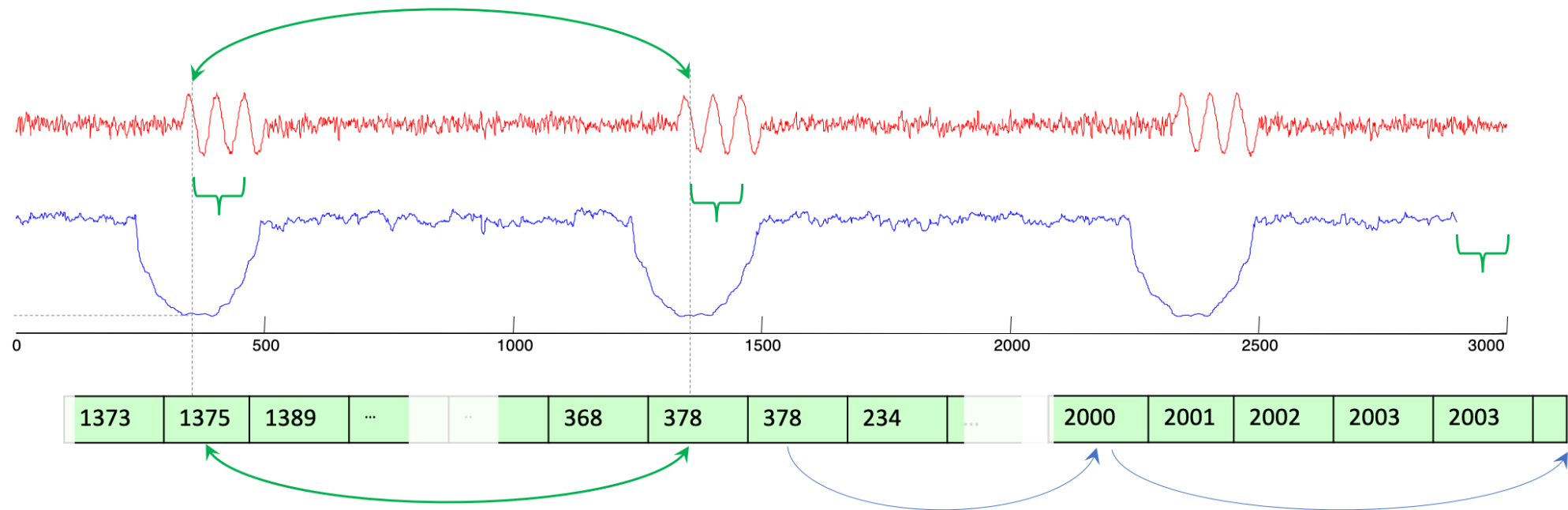
- The index of corresponding nearest neighbor of each subsequence is also stored in a vector called matrix profile index.



The matrix profile value at location  $i$  is the distance between  $T_i$  and its nearest neighbor

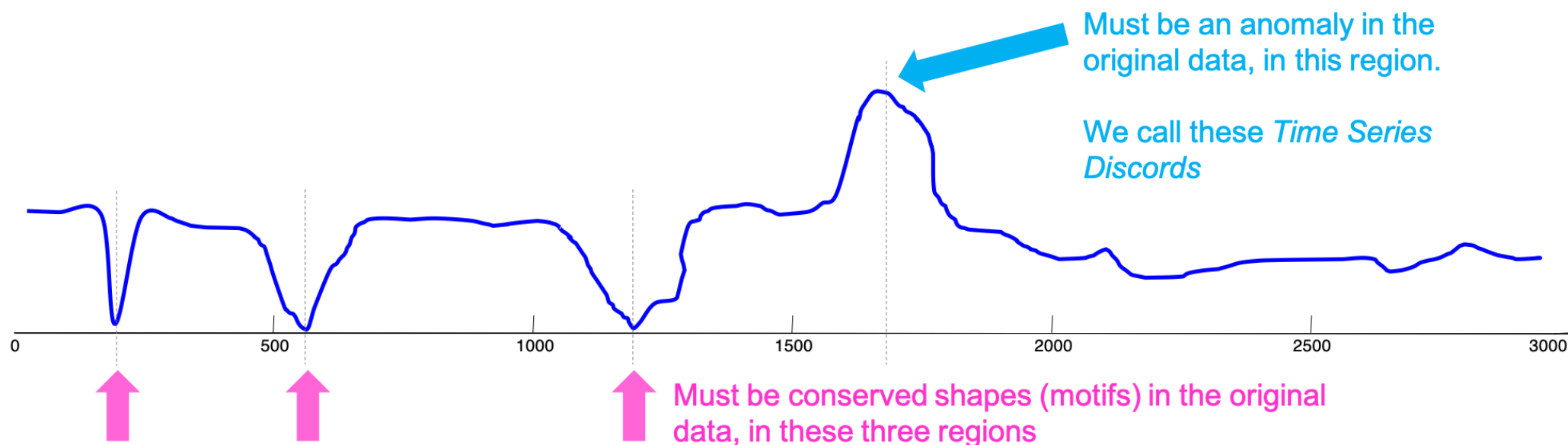
# Matrix Profile

- The MP index allows to find the nearest neighbor to any subsequence in constant time.
- Note that the pointers in the matrix profile index are not necessarily symmetric.
- If A points to B, then B may or may not point to A
- The classic TS motif: the two smallest values in the MP must have the same value, and their pointers must be mutual.



# How to “read” a Matrix Profile

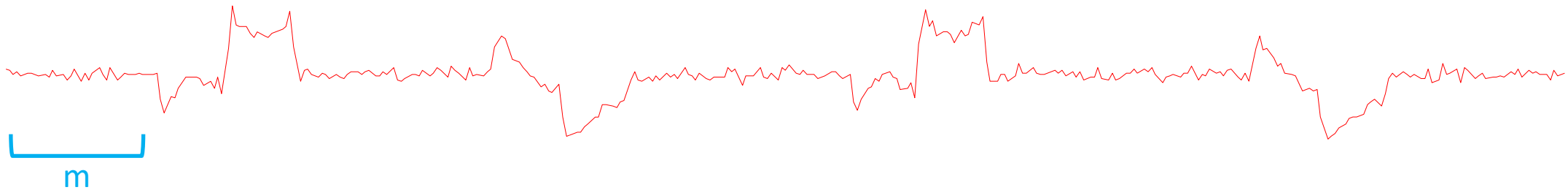
- For relatively low values, you know that the subsequence in the original TS must have (at least one) relatively similar subsequence elsewhere in the data (such regions are “motifs”)
- For relatively high values, you know that the subsequence in the original TS must be unique in its shape (such areas are anomalies).





# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



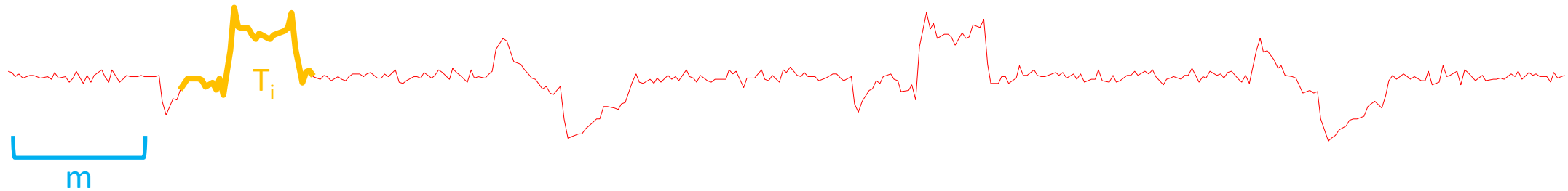
inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

Matrix profile is initialized as inf vector

This is just a toy example, so the values and the vector length does not fit the time series shown above

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .

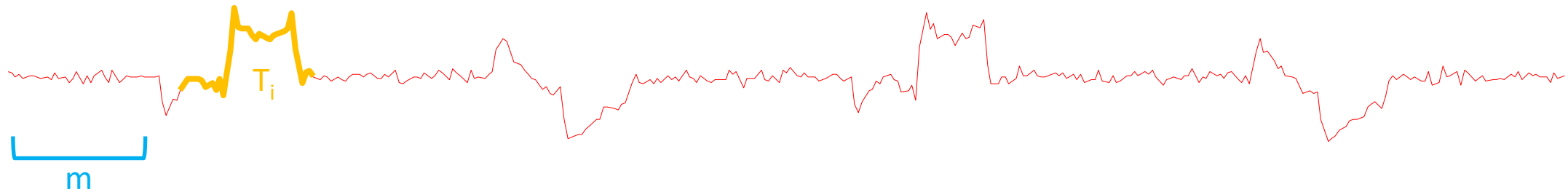


inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

At the first iteration, a subsequence  $T_i$  is randomly selected from  $T$

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

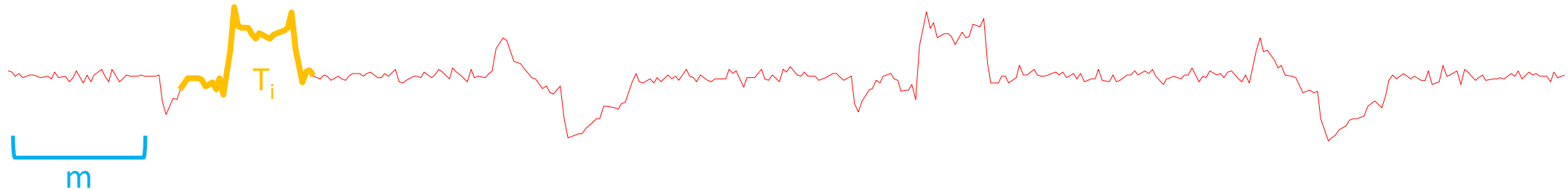
We compute the distances between  $T_i$  and every subsequences from  $T$  (time complexity =  $O(|T| \log(|T|))$ )  
We then put the distances in a vector based on the position of the subsequences

3	2	0	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

↖ The distance between  $T_i$  and  $T_1$  (first subsequence) is 3

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

We compute the distances between  $T_i$  and every subsequences from  $T$  (time complexity =  $O(|T| \log(|T|))$ )  
We then put the distances in a vector based on the position of the subsequences

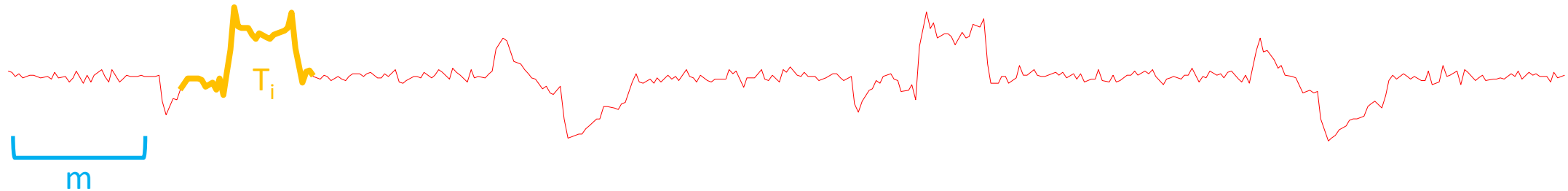
3	2	0	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---



Let say  $T_i$  happen to be the third subsequences, therefore the third value in the distance vector is 0

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

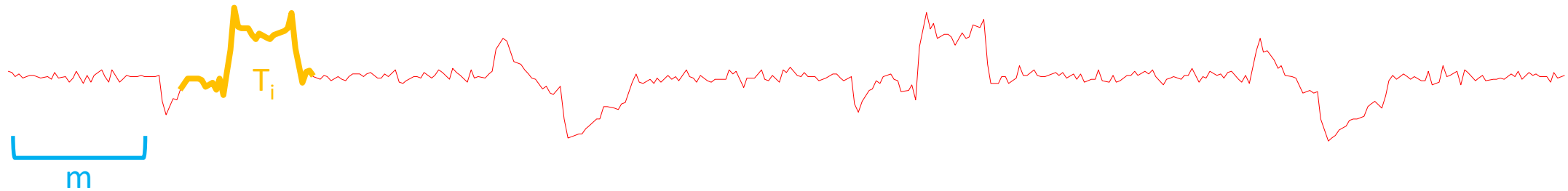
min

Matrix profile is updated by apply elementwise minimum to these two vectors

3	2	0	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



3	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
---	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

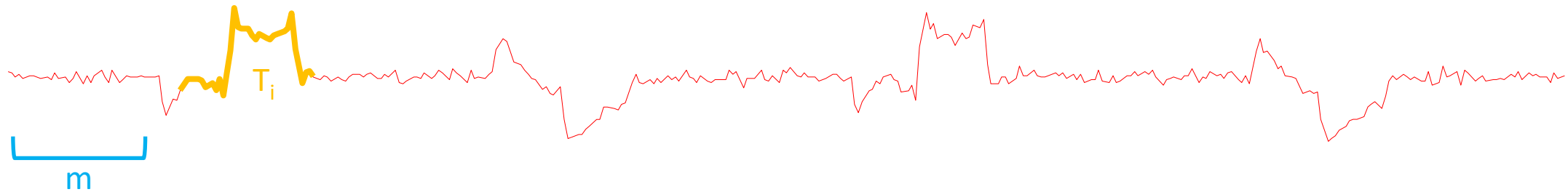
min

Matrix profile is updated by apply elementwise minimum to these two vectors

3	2	0	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



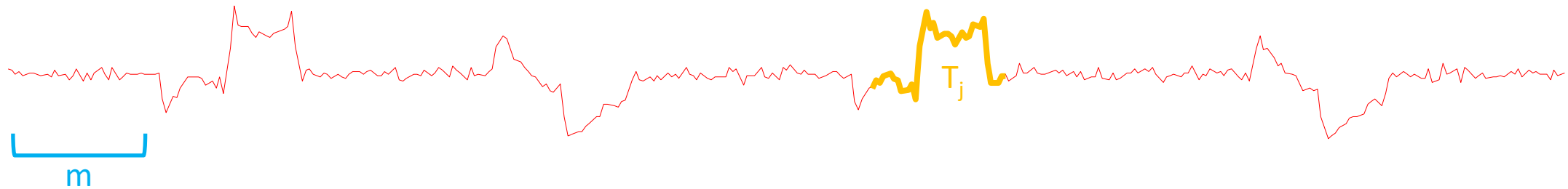
3	2	inf	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

After we finish update matrix profile for the first iteration

3	2	0	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



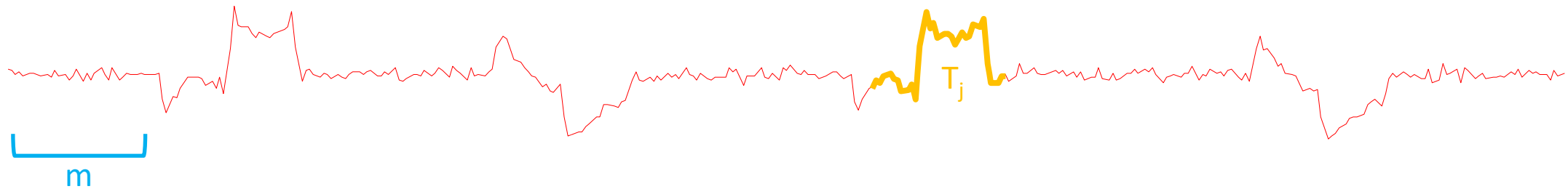
3	2	inf	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

In the second iteration, we randomly select another subsequence  $T_j$  and it happens to be the 12<sup>th</sup> subsequence



# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



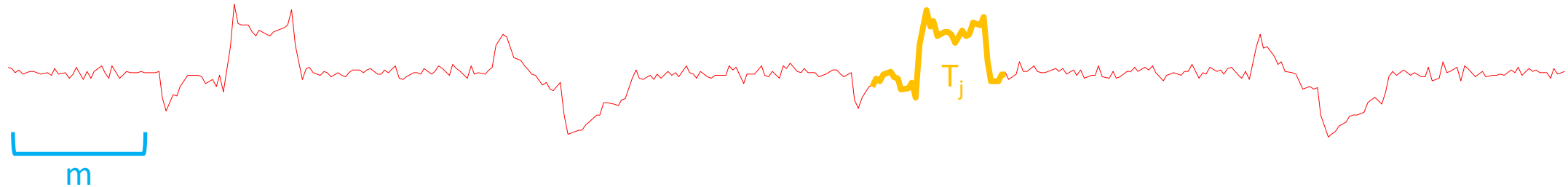
3	2	inf	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Once again, we compute the distance between  $T_j$  and every subsequences of  $T$

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



3	2	inf	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

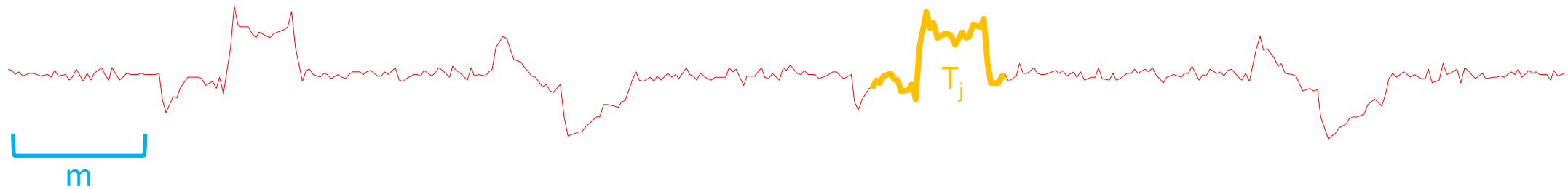
min

The same elementwise minimum

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



2	2	inf	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

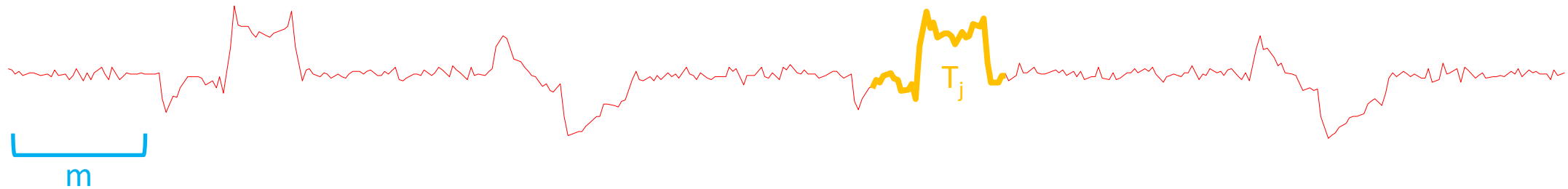
min  $\updownarrow$

The same elementwise minimum

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



2	2	inf	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	-----	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

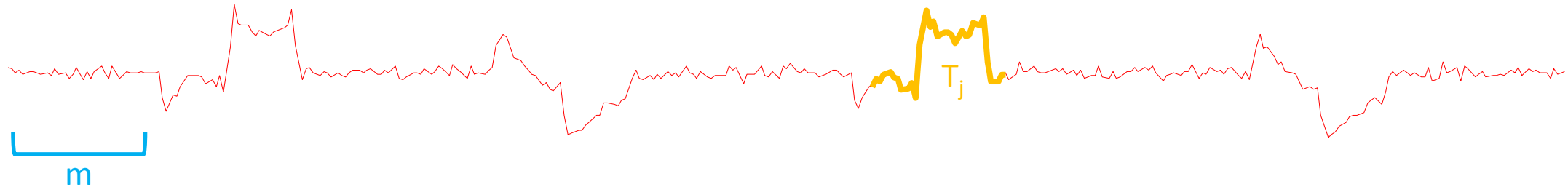
min

The same elementwise minimum

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



2	2	1	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

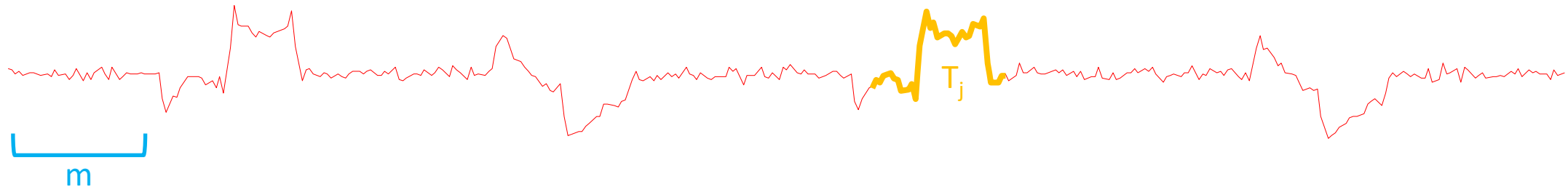
min

The same elementwise minimum

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



2	2	1	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

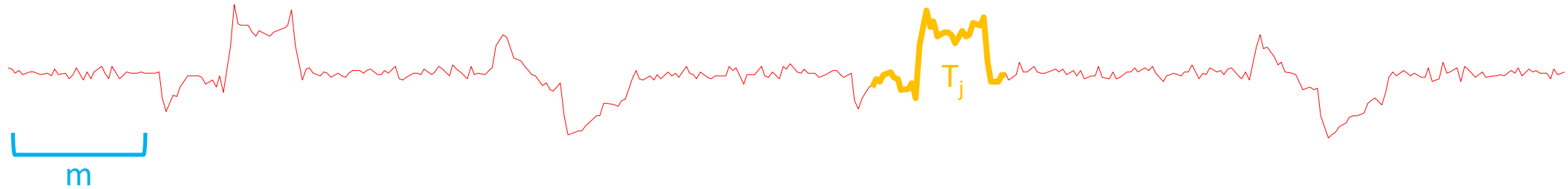
min

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

We repeat the two steps (distance computation and update) until we have used every subsequences. The different indexes are analyzed in parallel and the distance is calculated using the Mueen's Algorithm for Similarity Search (MASS) <https://www.cs.unm.edu/~mueen/FastestSimilaritySearch.html>

# How to Compute Matrix Profile?

- Given a time series,  $T$  and a desired subsequence length,  $m$ .



2	2	1	5	3	4	5	1	2	9	8	4	2	3	4	8	6	2	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

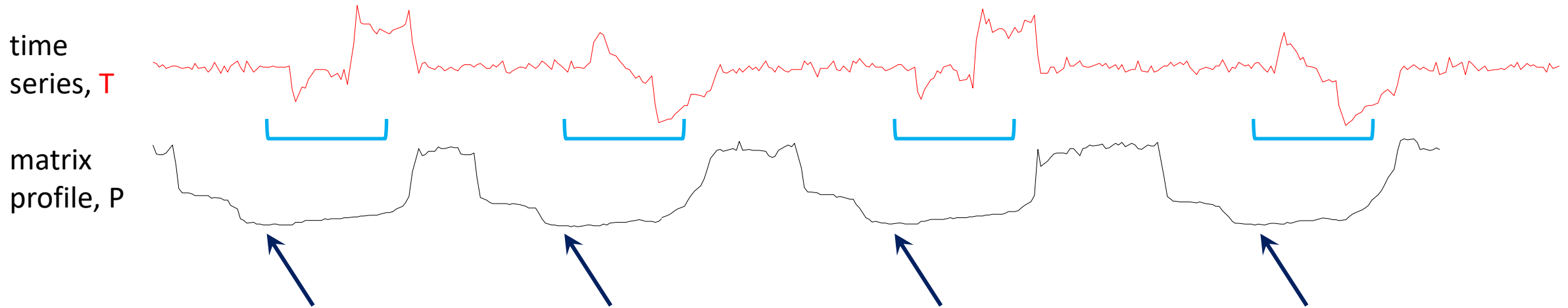
min  $\updownarrow$

2	3	1	4	4	3	6	2	1	5	8	0	2	3	5	9	4	2	2
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

There are  $|T|$  subsequences and the distance computation is  $O(|T|\log(|T|))$

The overall time complexity is  $O(|T|^2\log(|T|))$

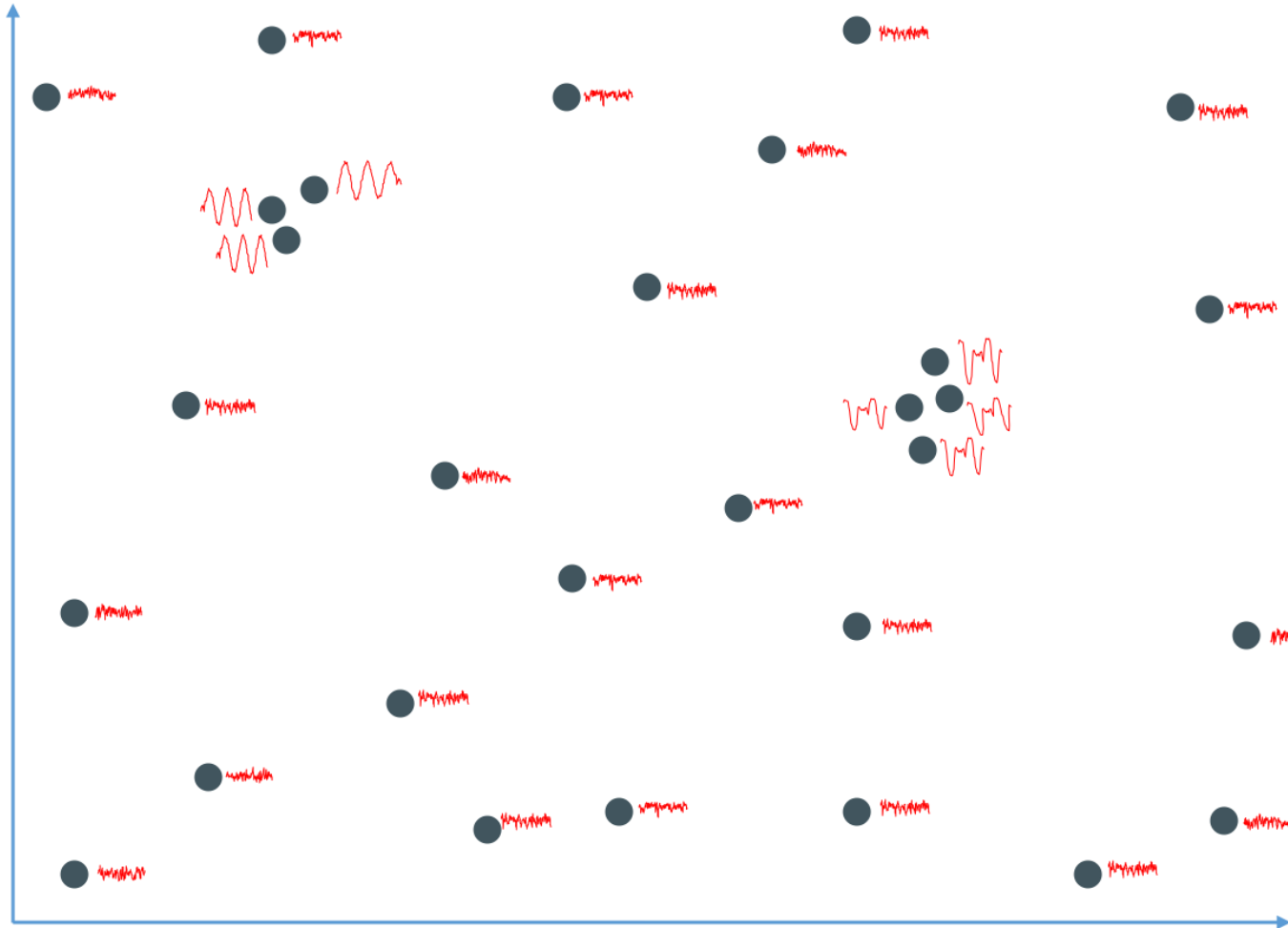
# Motif Discovery From Matrix Profile



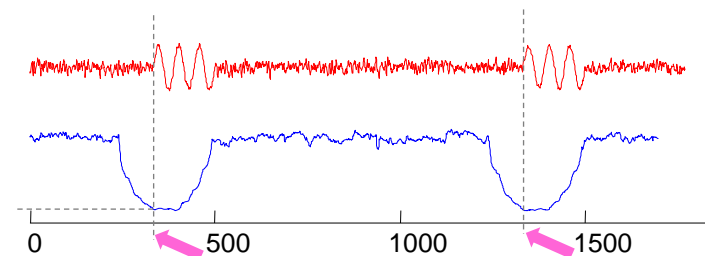
Local minimums are corresponding to motifs



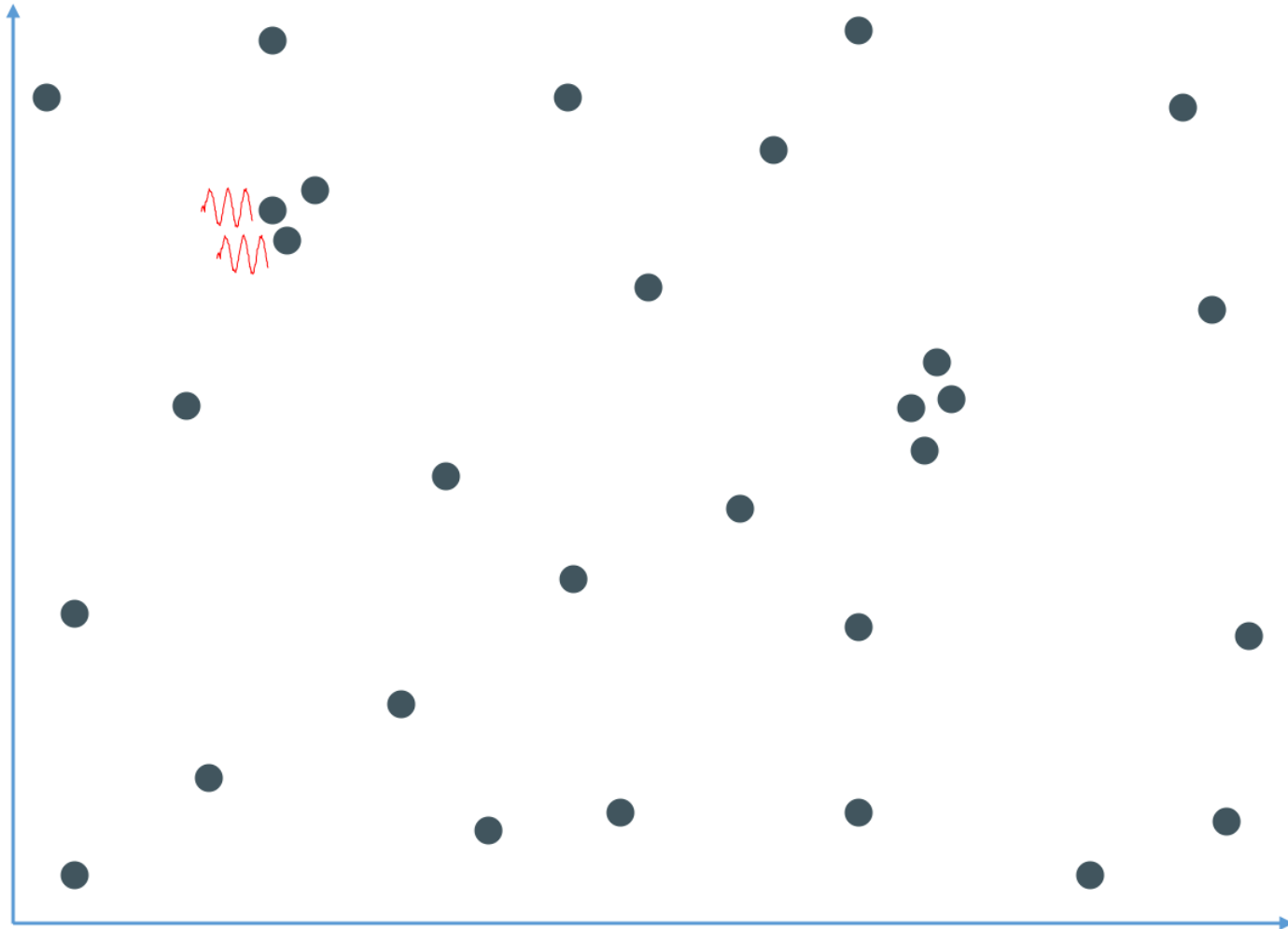
# Motif Discovery From Matrix Profile



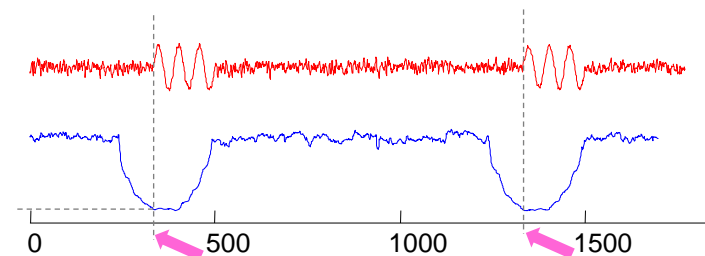
- It is sometime useful to think of time series subsequences as points in m-dimensional space.
- In this view, dense regions in the m-dimensional space correspond to regions of the time series that have a low corresponding MP.



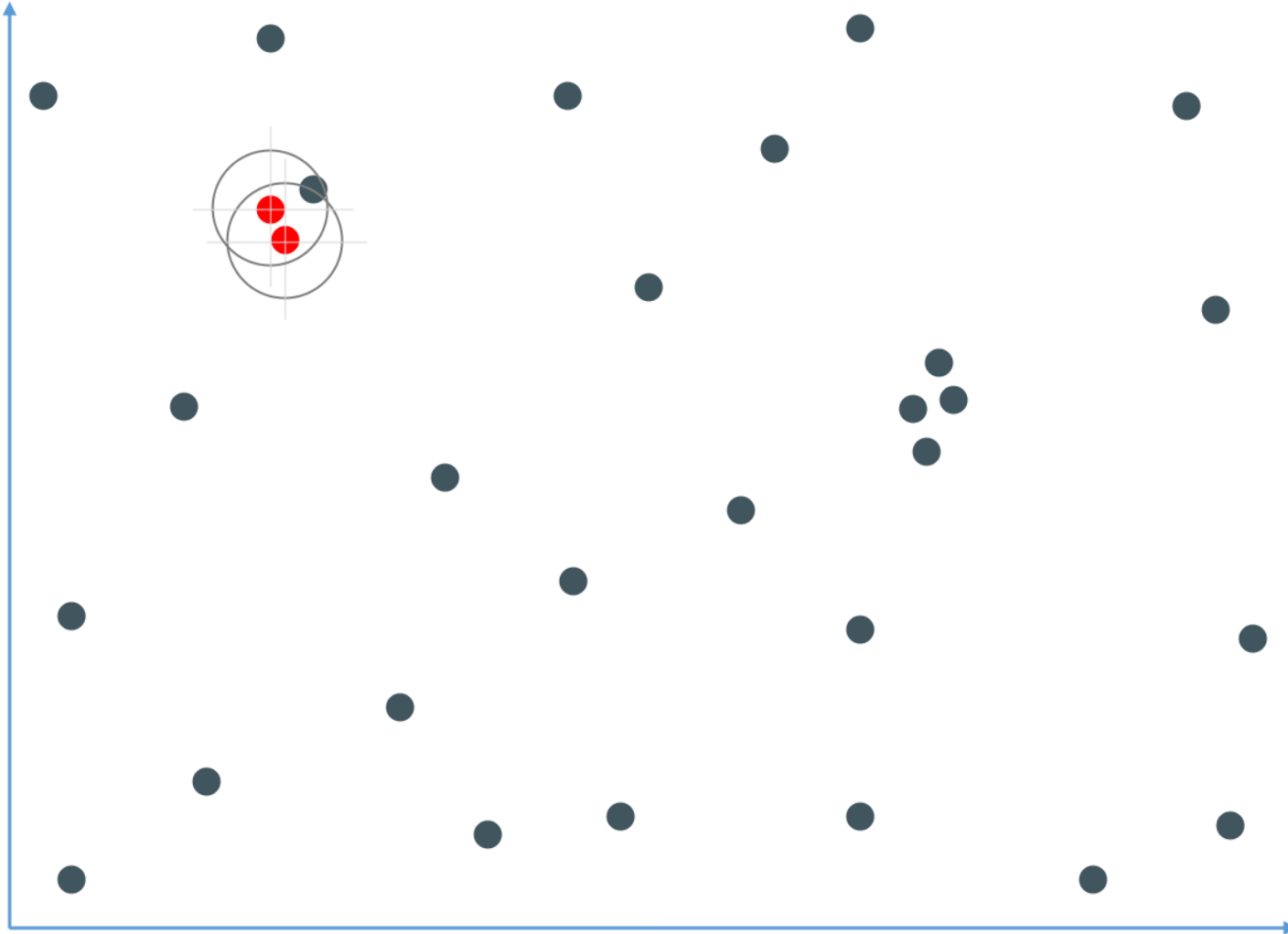
# Top-K Motifs



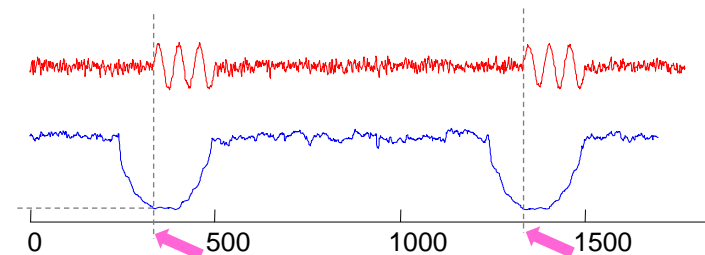
- We need a parameter  $R$ .
- $1 < R < (\text{small number, say } 3)$
- Let's make  $R = 2$  for now.
- We begin by finding the nearest pair of points, the *motif pair*....
- This the pair of subsequences corresponding to lowest pair of values in the MP



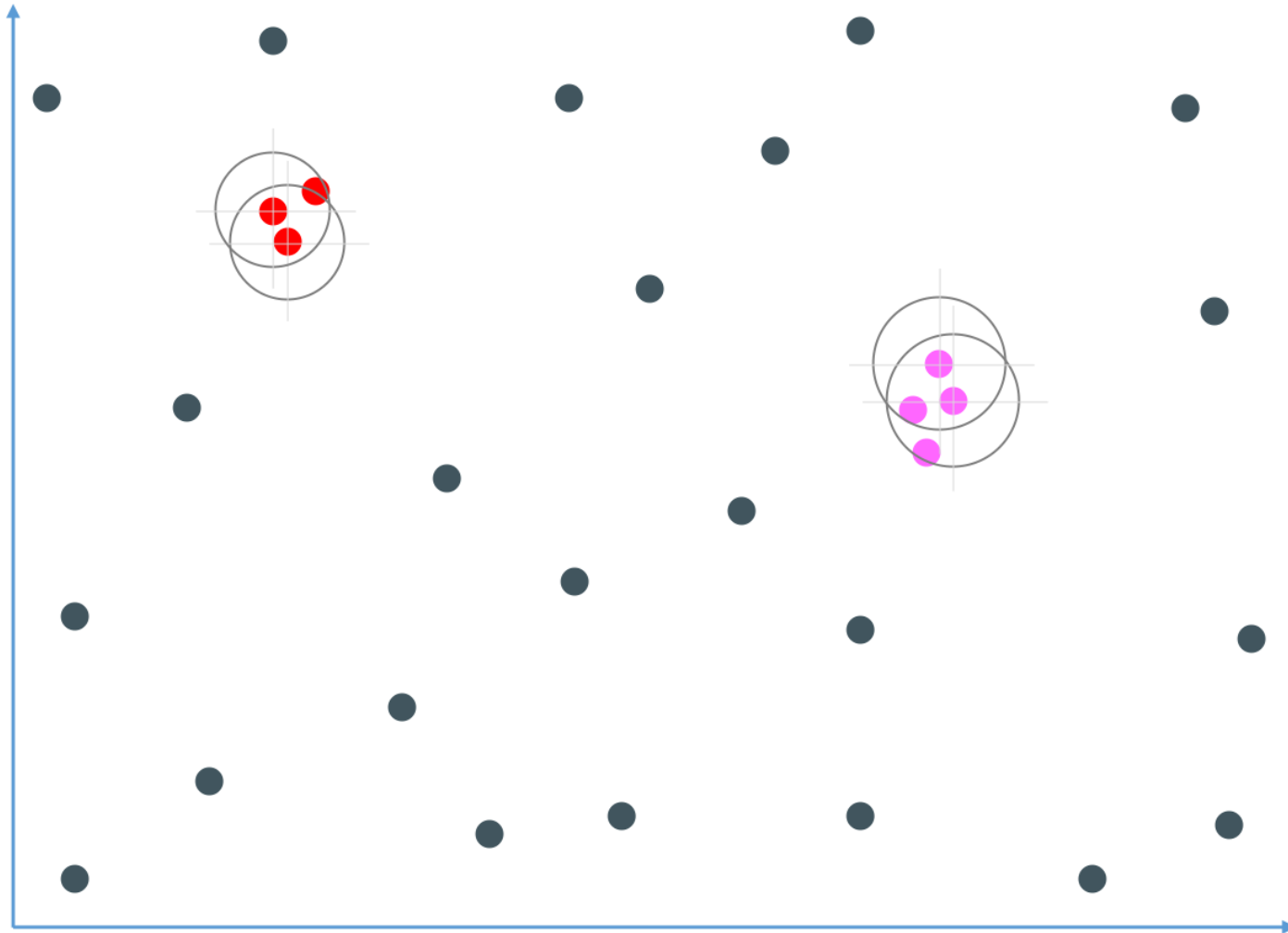
# Top-K Motifs



- We find the nearest pair of points are  $D1$  apart.
- Let's draw a circle,  $D1$  times  $R$ , around both points.
- Any points that are within either of these circles, are added to this motif, in this case just one.
- The Top-1 motif has three members, it is done.



# Top-K Motifs



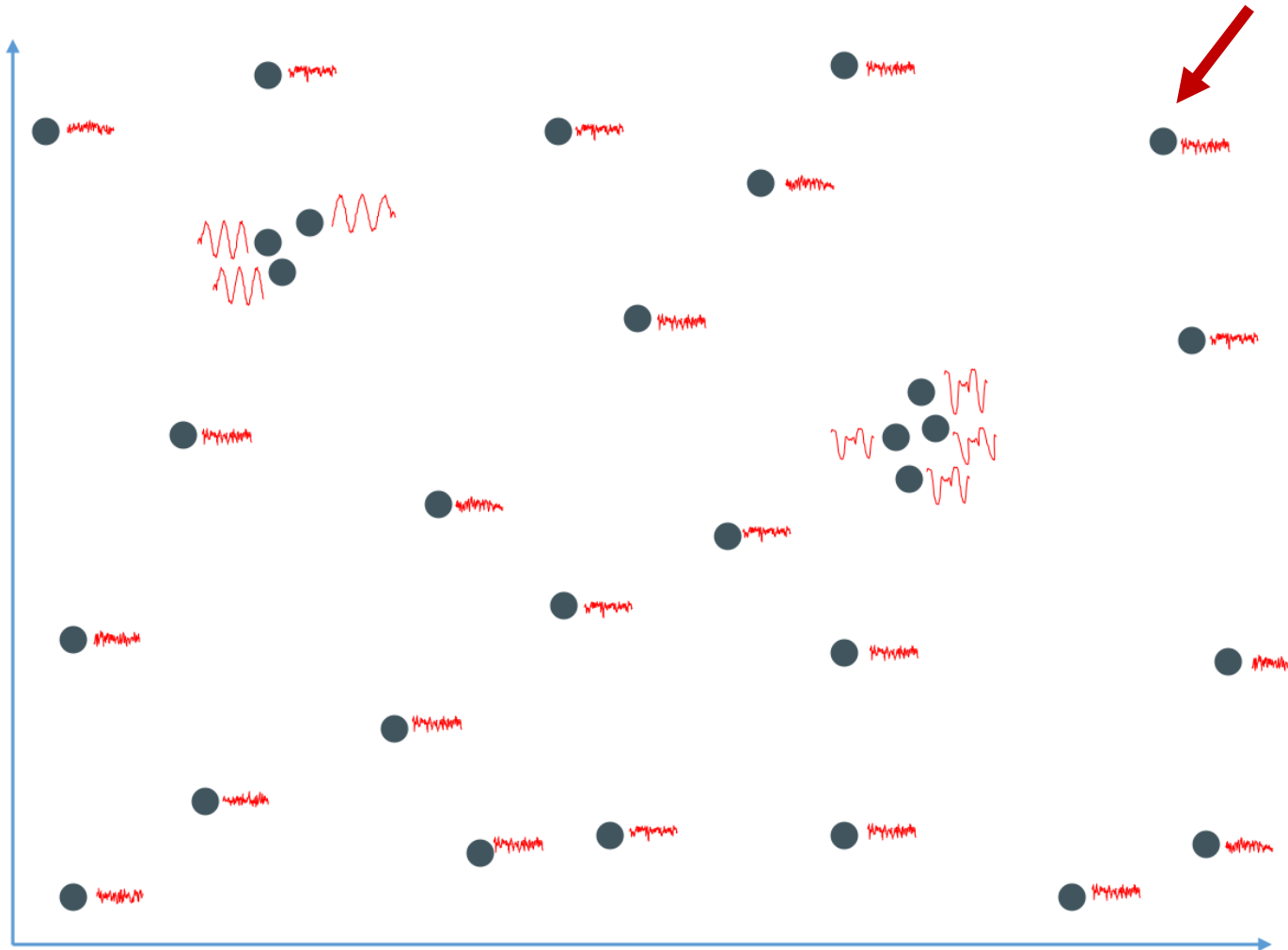
- Now let's find the Top-2 motif. We find the **nearest pair of points**, excluding anything from the top motif.
- The nearest pair of points are  $D_2$  apart.
- Let's draw a circle  $D_2$  times  $R$ , around both points.
- Any points that are within either of these circles, is added to this motif, in this case there are two for a total of four items in the Top-2 Motif

# Top-K Motifs

---

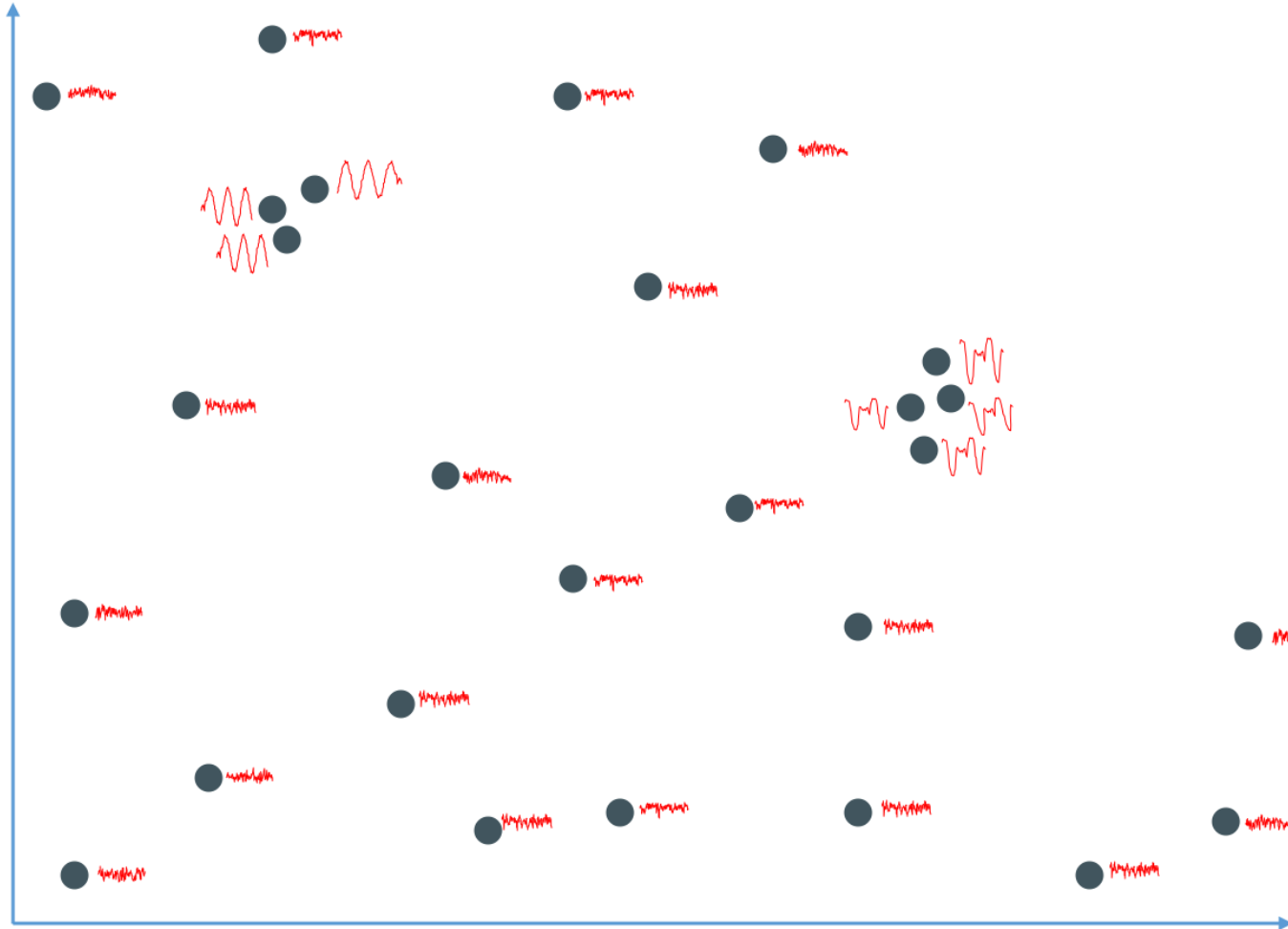
- We are done with the Top-2 Motif
- Note that we will always have:
  - $D_1 < D_2 < D_3 \dots D_K$
- **When to stop?** (what is K?)
- We could use MDL or a predefined K.

# Anomaly Discovery From Matrix Profile



- We need a parameter  $E$  of subsequences to exclude in the vicinity of the anomaly.
- Lets make  $E = 2$  for now.
- We begin by finding the subsequence with the highest distance in the MP
- This corresponding to biggest anomaly

# Top-K Anomaly



- Then we look for the  $E$  closest subsequences to the anomaly.
- We remove all of them.
- We can use a predefined  $K$  or the MDL to stop.

# References

- Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets. Chin-Chia Michael Yeh et al. 1997
- Time Series Shapelets: A New Primitive for Data Mining. Lexiang Ye and Eamonn Keogh. 2016.
- Josif Grabocka, Nicolas Schilling, Martin Wistuba, Lars Schmidt-Thieme (2014): Learning Time-Series Shapelets, in Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD 2014
- Deep learning for time series classification: a review. Hassan Ismail Fawaz et al. 2019.

## Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View that Includes Motifs, Discords and Shapelets

Chin-Chia Michael Yeh, Yan Zhu, Lindmila Ulanova, Nurjahan Begum, Yifei Ding, Hong Anli Dou, Diego Furtado Silva, Abdullatif Mosen, and Eamonn Keogh  
University of California, Riverside, Universidade de São Paulo, University of New Mexico  
(myn003, yzho01, hlan001, yzho07, hlan001)@ucr.edu, dergafob@unm.edu, unoz01@unm.edu, eamonn@cs.ucr.edu

**Abstract**—The all-pairs-similarity-search (or similarity join) problem has been extensively studied for text and a handful of other domains. However, surprisingly little progress has been made on similarity joins for time series subsequences. The lack of progress probably stems from the daunting nature of the problem. For even modest sized datasets the obvious nested-loop algorithm can take months, and typical speed-up techniques in this domain (i.e., indexing, lower-bounding, rectangular-approximate pruning and early abandonment) do not produce one or two orders of magnitude speeding. In this work we introduce a novel scalable algorithm for time series subsequence all-pairs-similarity-search. For exceptionally large datasets, the algorithm can be trivially cast as an anytime algorithm and produce high-quality approximate solutions in reasonable time. The exact similarity join algorithm compares the answer to the time series motif and time series discord problem in a side-effect, and our algorithm incidentally provides the fastest known algorithm for both these extensively-studied problems. We demonstrate the utility of our ideas for many time series data mining problems, including motif discovery, anomaly discovery, shapelet discovery, semantic segmentation, density estimation, and contrast set mining.

**Keywords**—Time Series; Similarity Joins; Motif Discovery

### 1. INTRODUCTION

The all-pairs-similarity-search (also known as similarity join) problem comes in several variants. The basic task is: Given a collection of data objects, retrieve the nearest neighbor for each object in the text domain. The algorithm has applications in a host of problems, including community discovery, duplicate detection, collaborative filtering, clustering, and query refinement [1]. While virtually all text processing algorithms have matured in time series data mining, there has been surprisingly little progress on Time Series subsequence All-Pairs-Similarity-Search (TSAPSS).

We believe that this lack of progress stems not from a lack of interest in this useful primitive, but from the daunting nature of the problem. Consider the following example that reflects the needs of an industrial collaborator. A boiler at a chemical refinery reports pressure once a minute. After a year, we have a time series of length 521,600. A plant manager may wish to do a similarity self-join on this data with week-long subsequences (10,080) to discover operating regimes (constant vs. winter vs. light distillate vs. heavy distillate, etc). The obvious nested loop algorithm requires  $112,800,002,200$  Euclidean distance computations. If we assume each one takes 0.0001 seconds, then the join will take 11.8 days. The core combination of this work is to show that we can reduce this time to 4.3 hours, using an off-the-shelf desktop computer. Moreover, we show that this join can be computed and/or updated incrementally. This would maintain the join essentially forever on a standard

This is the author's version of an article published in Data Mining and authenticated version is available online at: <https://doi.org/10.1007/s1>

## Deep learning for time series classification

Hassan Ismail Fawaz<sup>1</sup> · Germain Forestier<sup>1,2</sup> · Jonathan W. Hussane Idmouhar<sup>1</sup> · Pierre-Alain Muller<sup>1</sup>

**Abstract** Time Series Classification (TSC) is an important and challenging task. With the increase of time series data availability, hundreds of TSC algorithms have been proposed. Among these methods, only a few have considered Deep Neural Network (DNN) architectures. This is surprising as deep learning has seen very successful applications in various time series domains under a unified taxonomy: data such as text and audio can also be processed with DNNs to learn for document classification and speech recognition. In this article, we present the performance of deep learning algorithms for TSC by present most recent DNN architectures for TSC. We give an overview of the applications in various time series domains under a unified taxonomy: provide an open source deep learning framework to the TSC community of the compared approaches and evaluated them on a univariate TS archive) and 12 multivariate time series datasets. By training 5,779 time series datasets, we propose the most exhaustive study of DNNs

**Keywords** Deep learning · Time series · Classification · Review

## 1 Introduction

During the last two decades, Time Series Classification (TSC) has been considered as one of the most challenging problems in data mining (Yang and Wu, 2006; Falng and Agon, 2012). With the increase of temporal data availability (Silva et al., 2018), hundreds of TSC algorithms have been proposed since 2015 (Bagnall et al., 2017). Due to their natural temporal ordering, time series data are present in almost every task that requires some sort of human cognitive process (Längkvist et al., 2014). In fact, any classification problem, using data that is registered taking into account some notion of ordering, can be cast as a TSC problem (Cristian Borges Goncalves, 2017). Time series are encountered in many real-world applications ranging from electronic health records (Rajkumar et al., 2018) and human activity recognition (Nweke et al., 2018; Wang et al., 2018) to acoustic scene classification (Nwe et al., 2017) and cyber-security (Susto et al., 2018). In addition, the diversity of the datasets' types in the UCR/UEA archive (Chen et al., 2015b; Bagnall et al., 2017) (the largest repository of time series datasets) shows the different applications of the TSC problem.

© H. Ismail Fawaz  
E-mail: hassan.ismail.fawaz@uniba.fr  
<sup>1</sup>IRDMAS, Université Haute Alsace, Mulhouse, France  
<sup>2</sup>Faculty of IT, Monash University, Melbourne, Australia

## Time Series Shapelets: A New Primitive for Data Mining

Lexiang Ye  
Dept. of Computer Science & Engineering  
University of California, Riverside, CA 92521  
lyxiang@cs.ucr.edu

Eamonn Keogh  
Dept. of Computer Science & Engineering  
University of California, Riverside, CA 92521  
eamonn@cs.ucr.edu

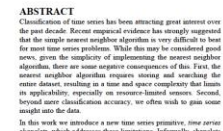


Figure 1: A diagram showing two plants, Urtica dioica and Verbena officinalis, with their corresponding time series plots. The Urtica dioica plot shows a series of peaks and valleys, while the Verbena officinalis plot shows a series of peaks and valleys with a different shape.

Figure 1: A diagram showing two plants, Urtica dioica and Verbena officinalis, with their corresponding time series plots. The Urtica dioica plot shows a series of peaks and valleys, while the Verbena officinalis plot shows a series of peaks and valleys with a different shape.

**Abstract** Classification of time series has been attracting great interest over the past decade. Recent empirical evidence has strongly suggested that the nearest neighbor algorithm is very difficult to beat for more time series problems. While this may be considered good news, given the complexity of implementing the nearest neighbor algorithm, there are some negative consequences of this. First, the nearest neighbor algorithm requires storing and searching the entire dataset, resulting in a time and space complexity that limits its applicability to large-scale time series. Second, beyond near classification accuracy, we often wish to gain some insight into the data.

In this work we introduce a new time series primitive: time series shapelets, which address these limitations. Informally, shapelets are time series subsequences which act as some more meaningful representative of a class. As we shall show with extensive empirical evaluation on diverse datasets, algorithms based on the time series shapelet primitive can be interpretable, more accurate and significantly faster than state-of-the-art classifiers.

**Categories and Subject Descriptors** H.2.8 Database Management; Database Applications; Data Mining

**General Terms** Algorithms; Experimentation

**1. INTRODUCTION** While the last decade has seen a huge interest in time series classification, to date the most accurate and robust method is the simple nearest neighbor algorithm (Hilb et al., 2014). While the nearest neighbor algorithm has the advantages of simplicity and not requiring extensive parameter tuning, it does have several important disadvantages. Chief among these are its space and time requirements, and the fact that it does not tell us anything about why a particular object was assigned to a particular class.

In this work we present a novel time series data mining primitive called time series shapelets. Informally, shapelets are time series subsequences which act as some more meaningful representative of a class. While we believe that shapelets can have many uses in data mining, one obvious application of them is to mitigate the two weaknesses of the nearest neighbor algorithm.

In order to make digital or hard copies of all or part of this work for personal or classroom use, is granted without fee provided that copies are made in the name of the publisher and the full citation on the first page. To copy other than for personal or classroom use, is prohibited. To request more information, contact the publisher at the address below. Copyright 2009 ACM 978-1-60559-950-9/09 \$5.00.

Because we are defining and solving a new problem, we will take some time to consider a detailed motivating example. Figure 1 shows some examples of leaves from two classes, Urtica dioica (Stinging nettle) and Verbena officinalis. These two plants are commonly confused, hence the colloquial name "bite tender" for Verbena officinalis.



Figure 2: A diagram showing a single leaf from Urtica dioica and a single leaf from Verbena officinalis. The Urtica dioica leaf is shown with its characteristic serrated edges and small hairs, while the Verbena officinalis leaf is shown with its smooth edges and larger size.

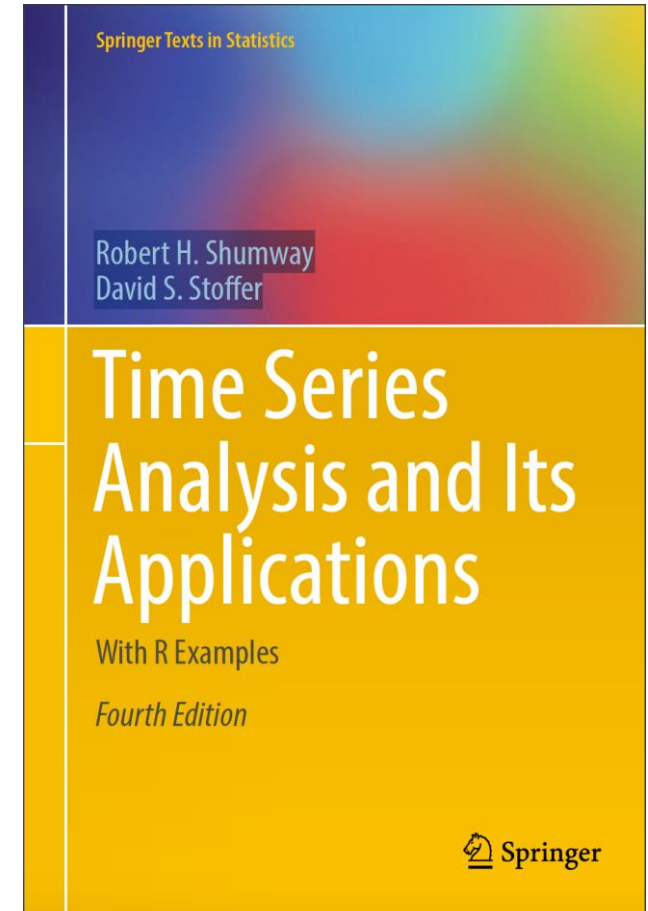
Suppose we wish to build a classifier to distinguish these two plants, what feature should we use? Since the main variability of color and size within each class completely dwarfs the inter-variability between classes, we first begin a search for the shapelets of the leaves. However, as we can see in Figure 1, the differences in the global shape are very subtle. Furthermore, it is very common for leaves to have dentations or "toothiness" due to insect damage, and these are likely to confuse any global measures of shape. Instead we attempt the following. We first convert each leaf into a one-dimensional representation as shown in Figure 2.

Such representations have been successfully used for the classification, clustering and outlier detection of shapes in recent years [8]. However, here we find that using a nearest neighbor classifier with either the (position-invariant) Euclidean distance or Dynamic Time Warping (DTW) distance does not significantly outperform random guessing. The reason for the poor performance of these otherwise very competitive classifiers seems to be due to the fact that the data is somewhat noisy (i.e. insect holes, and different stem lengths), and this noise is enough to swamp the subtle differences in the shapelets.



# 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. 4<sup>th</sup> 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](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.



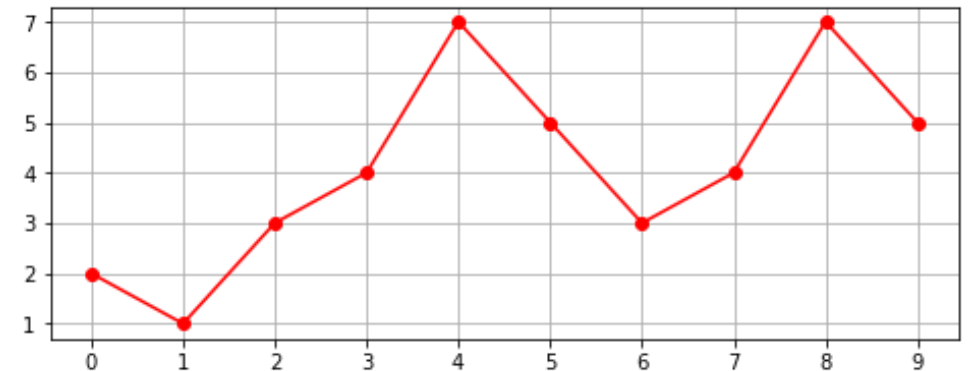
# Exercises Matrix Profile

---

# Matrix Profile

Given the TS  $x = \langle 2, 1, 3, 4, 7, 5, 3, 4, 7, 5 \rangle$

1. Build the Matrix Profile for  $x$  with  $m=4$  using the Manhattan distance as distance function between subsequences.
2. Draw the Matrix Profile
3. Identify the motifs with distance equals 0 and length equals to  $m$
4. Which is a correct value for  $m$  that would have retrieved more motifs with distance equals to 0?

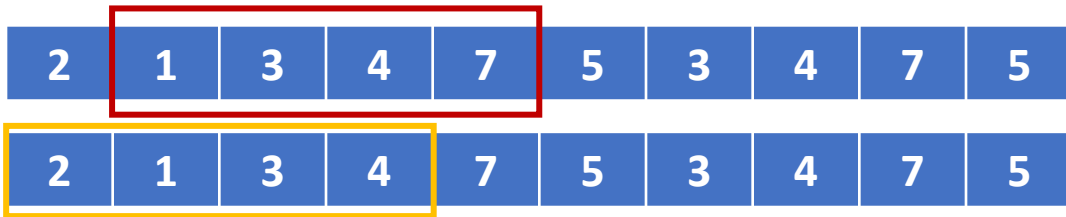


2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

inf									

m = 4



inf	7					

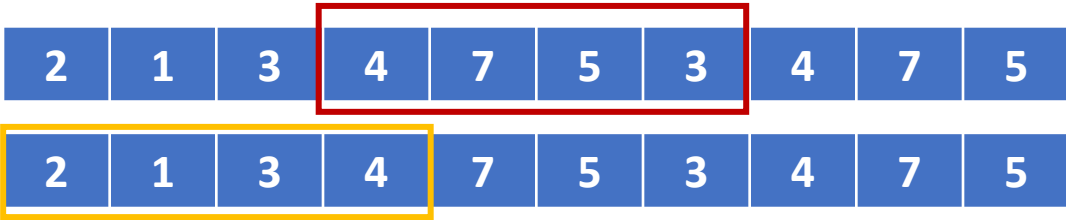
m = 4

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

inf	7	9				

m = 4



inf	7	9	11			

m = 4

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

inf	7	9	11	9		

m = 4



2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

inf	7	9	11	9	9	9

m = 4

2	1	3	4	7	5	3	4	7	5
2	1	3	4	7	5	3	4	7	5

inf	7	9	11	9	9	9

m = 4

2	1	3	4	7	5	3	4	7	5
2	1	3	4	7	5	3	4	7	5

inf	7	9	11	9	9	9
7						

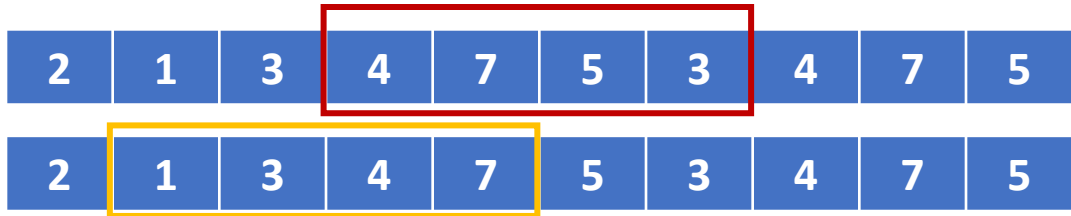
m = 4

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

inf	7	9	11	9	9	9
7	inf	8				

m = 4



inf	7	9	11	9	9	9
7	inf	8	12			

m = 4

2 1 3 4 7 5 3 4 7 5



2 1 3 4 7 5 3 4 7 5



inf	7	9	11	9	9	9
-----	---	---	----	---	---	---

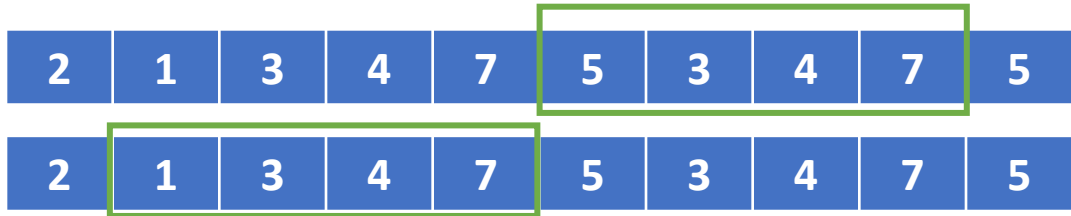
7	inf	8	12	12		
---	-----	---	----	----	--	--


m = 4

2	1	3	4	7	5	3	4	7	5
2	1	3	4	7	5	3	4	7	5

inf	7	9	11	9	9	9
7	inf	8	12	12	4	8

m = 4



inf	<b>7</b>	9	11	9	9	9
7	inf	8	12	12	<b>4</b>	8

m = 4



2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

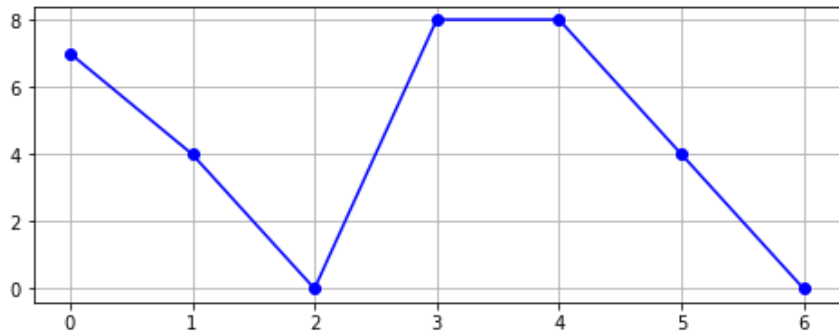
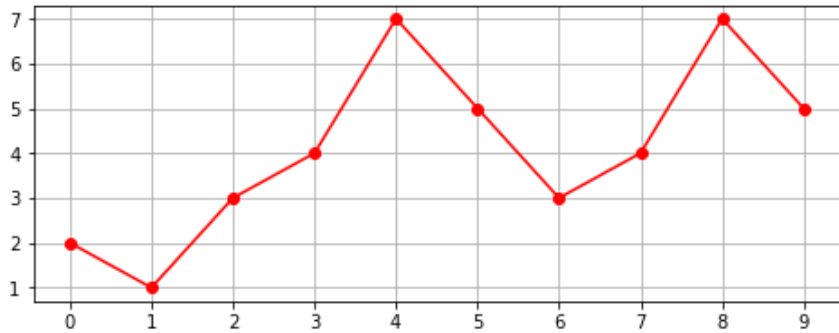
2	1	3	4	7	5	3	4	7	5
---	---	---	---	---	---	---	---	---	---

inf	<b>7</b>	9	11	9	9	9
7	inf	8	12	12	<b>4</b>	8
9	10	inf	8	9	8	<b>0</b>

m = 4

# Matrix Profile

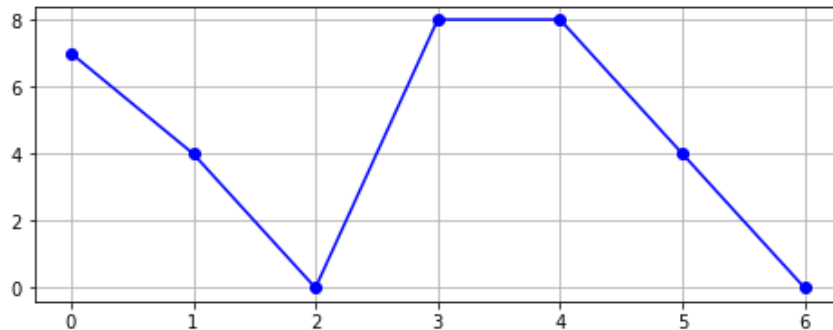
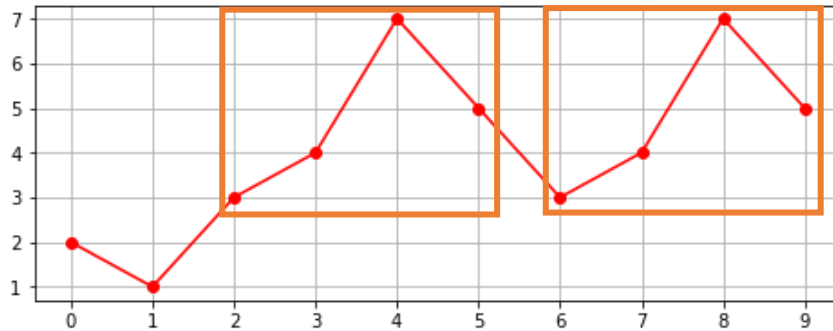
- $x = \langle 2, 1, 3, 4, 7, 5, 3, 4, 7, 5 \rangle$
- $mp = \langle 7, 4, 0, 8, 8, 4, 0 \rangle$



# Matrix Profile

- $x = \langle 2, 1, 3, 4, 7, 5, 3, 4, 7, 5 \rangle$
- $mp = \langle 7, 4, 0, 8, 8, 4, 0 \rangle$

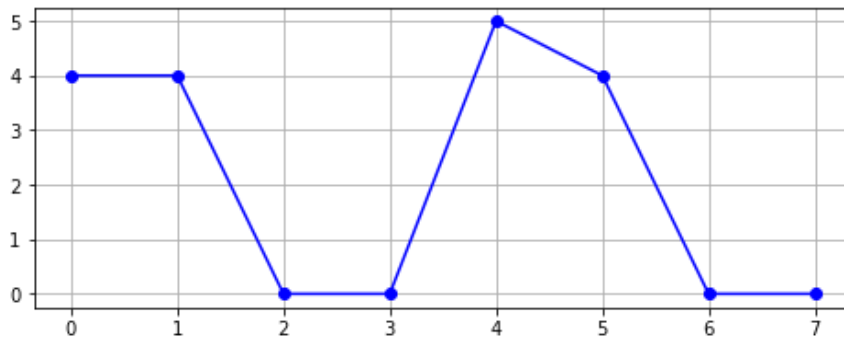
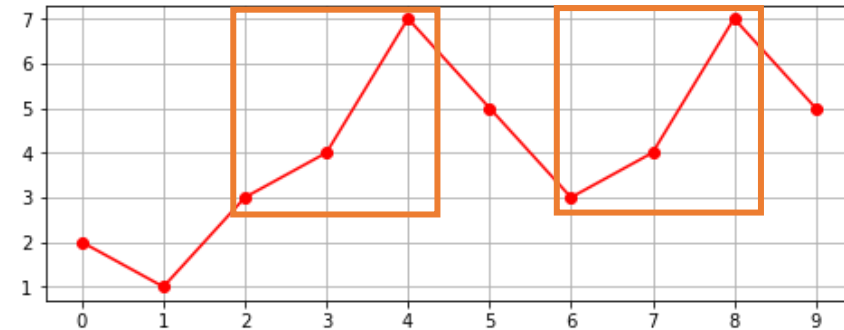
$m=4$



# Matrix Profile

- $x = \langle 2, 1, 3, 4, 7, 5, 3, 4, 7, 5 \rangle$
- $mp = \langle 4, 4, 0, 0, 5, 4, 0, 0 \rangle$

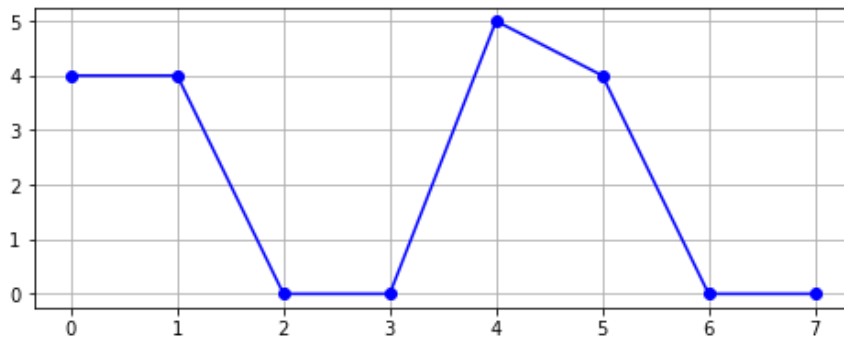
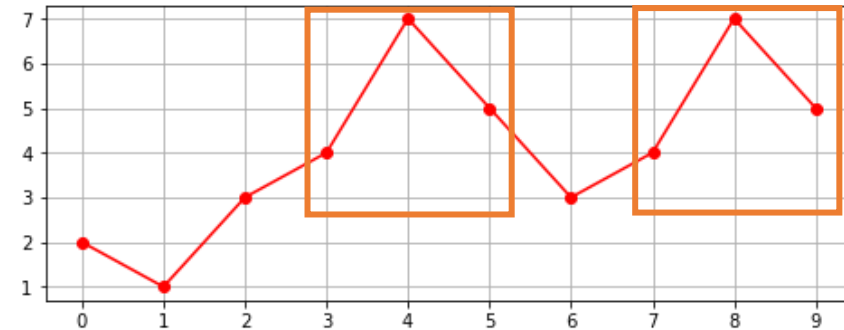
$m=3$



# Matrix Profile

- $x = \langle 2, 1, 3, 4, 7, 5, 3, 4, 7, 5 \rangle$
- $mp = \langle 4, 4, 0, 0, 5, 4, 0, 0 \rangle$

$m=3$



# Matrix Profile

---

Given the TS  $x = \langle 5, 5, 3, 5, 5, 1 \rangle$

1. Build the Matrix Profile for  $x$  with  $m=2$  using the Manhattan distance as distance function between subsequences.
2. Draw the Matrix Profile
3. Identify the motifs with distance equals 0 and length equals to  $m$

