

At Last! Time Series Joins, Motifs, Discords and Shapelets at Interactive Speeds



Eamonn Keogh

With

Yan Zhu, Chin-Chia Michael Yeh, Abdullah Mueen
with contributions from Zachary Zimmerman, Nader Shakibay Senobari,,
Gareth Funning, Philip Brisk, Liudmila Ulanova, Nurjahan Begum, Yifei
Ding, Hoang Anh Dau and Diego Silva

Outline

- In this talk I will introduce the *Matrix Profile*.
- I believe that the Matrix Profile will become the most *cited* and the most *used* time series data mining primitive introduced in the last decade.
- The Matrix Profile has implications for all shape-based time series data mining tasks, including: Classification, Clustering, Motif Discovery, Anomaly Detection, Joins, Density Estimation, Visualization, Semantic Segmentation and Rule Discovery.
- Among other things, the Matrix Profile allows time series batch operations to become truly *interactive* for the first time (Hench this talk)
- First, some boilerplate slides on time series...

The Ubiquity of Time Series

Humans measure *stuff*, and *stuff* keeps changing, thus we have time series everywhere.

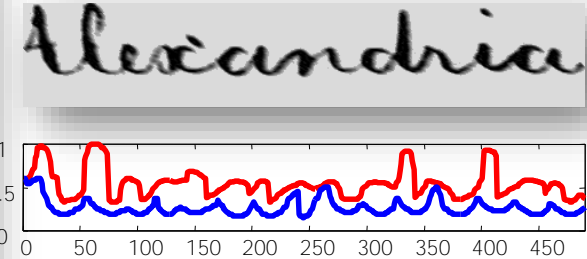
Sensors on machines



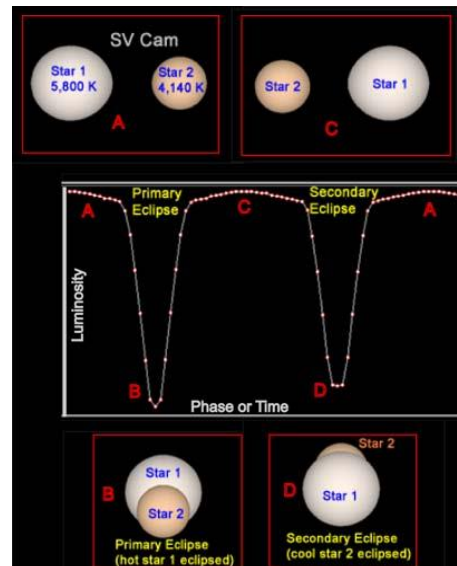
Stock prices



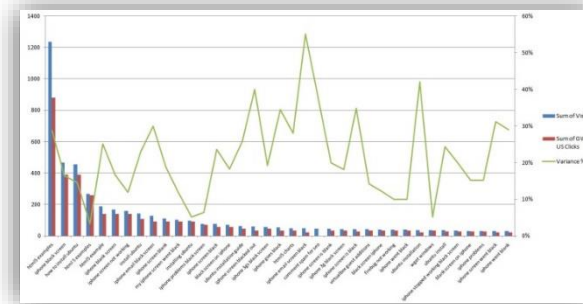
Hand writing



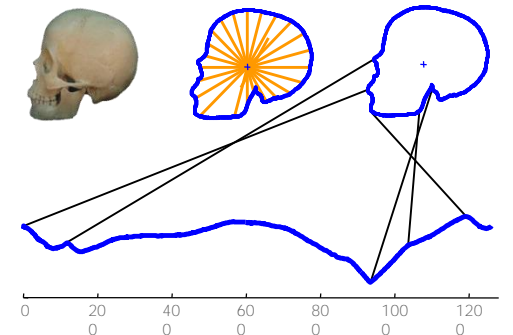
Astronomy:
star light curves



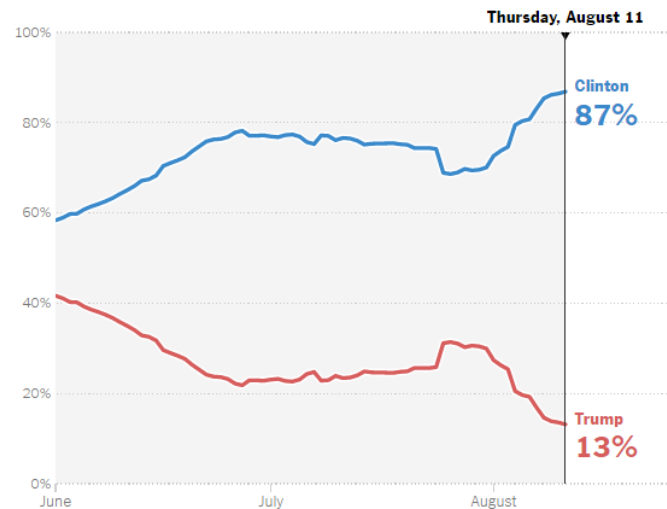
Web clicks



Shapes



Political Forecasts



Sound



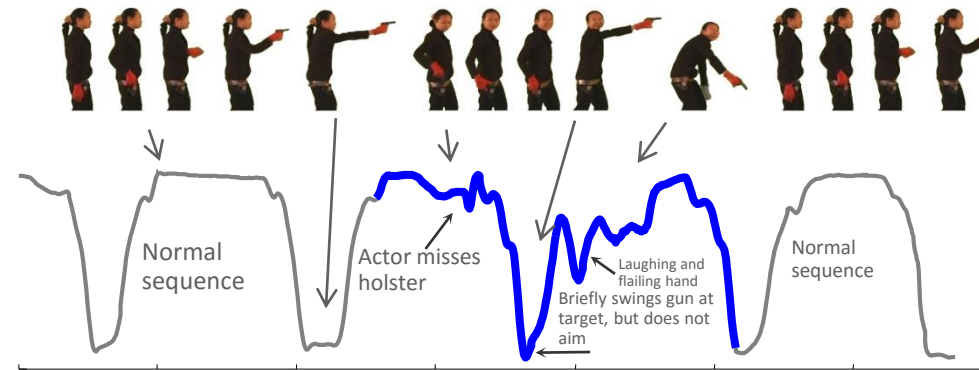
What do we want to do with all this Time Series?

The answer is... Everything!

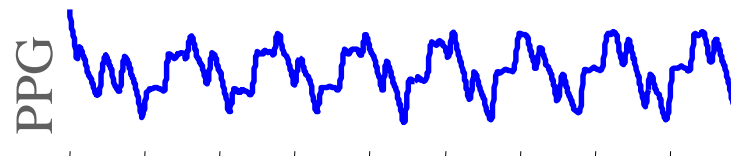
Classification, Clustering, Motif Discovery, Anomaly Detection, Joins, Density Estimation, Visualization, Semantic Segmentation and Rule Discovery.

In the last decade the community has come to the conclusion that if you can just measure similarity meaningfully for your domain, you can solve all these problems (possibly too slowly to be practical)

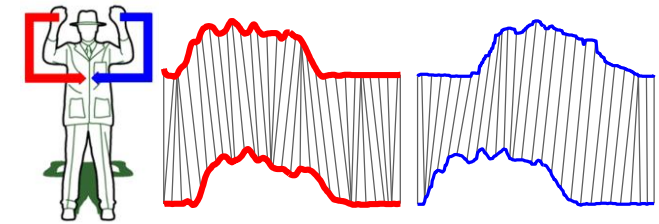
Therefore, computing similarity is typically the bottleneck for time series data mining.



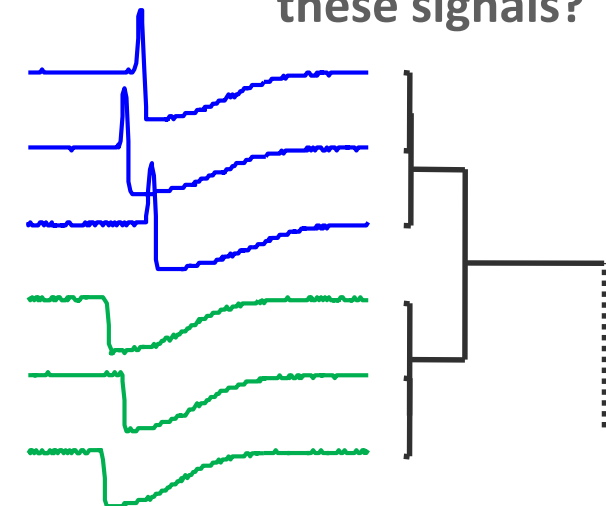
How is this man doing? (not well!)



What is the umpire signaling?



How should we group these signals?



Introduction to the Matrix Profile

With the context explained, let us take a first look at the Matrix Profile

We will begin by defining it (without discussing how we compute it)

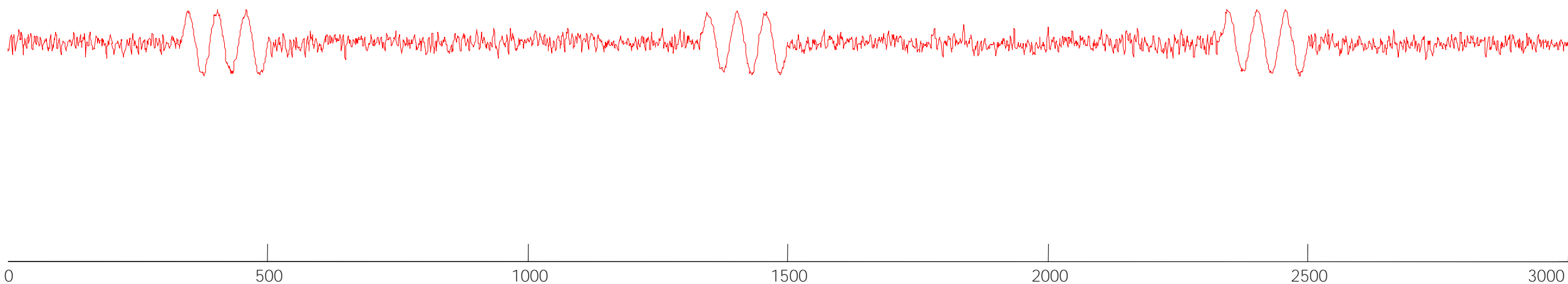
We will then show how it solves most time series problems

Finally, we will address the elephant in the room...


...the matrix profile seems to be much too expensive to compute to practical.



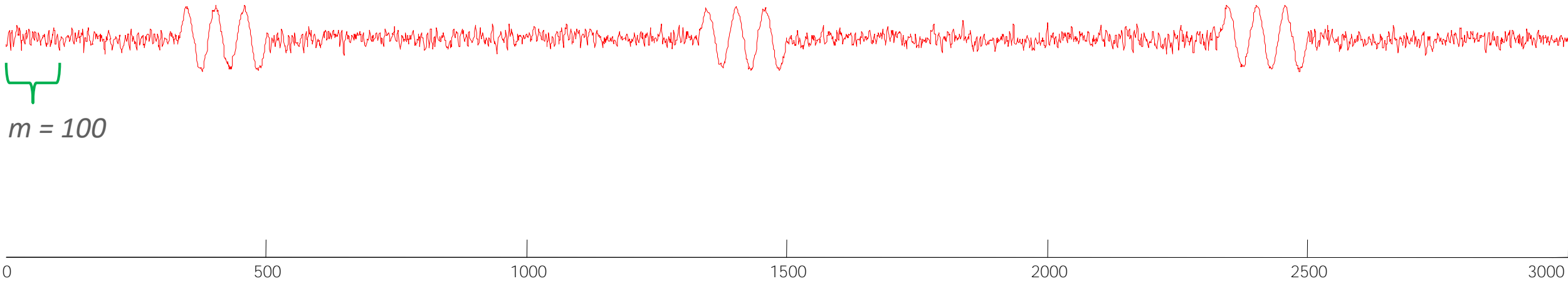
Intuition behind the Matrix Profile: Assume we have a time series T , lets start with a **synthetic one**...



$$|T| = n = 3,000$$

Note that for most time series data mining tasks, we are not interested in any *global* properties of the time series, we are only interested in small *local* subsequences, of this length, m 

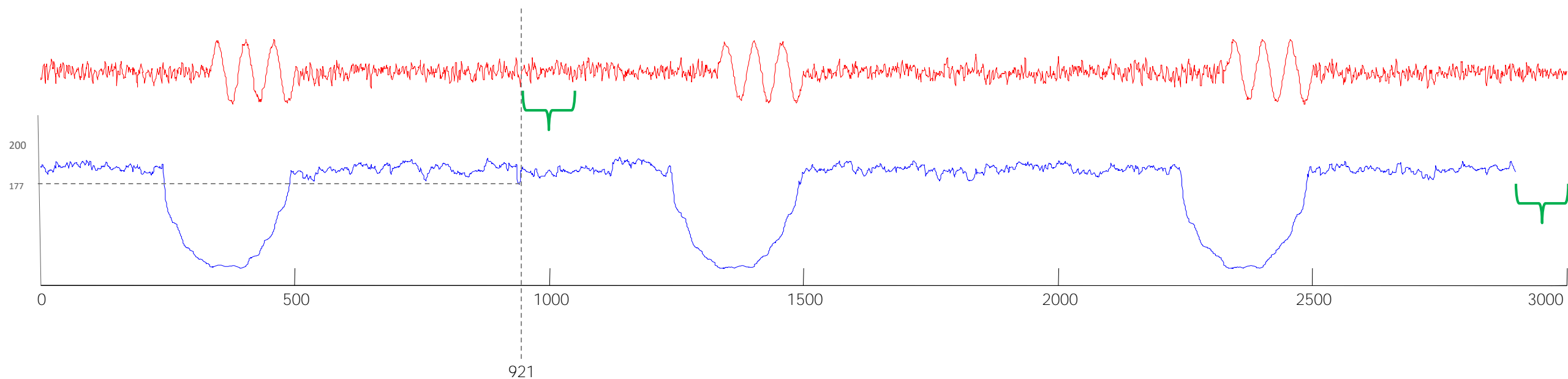
These subsequences might be about the length of individual heartbeats (for ECGs), individual days (for social media behavior), individual words (for speech analysis) etc



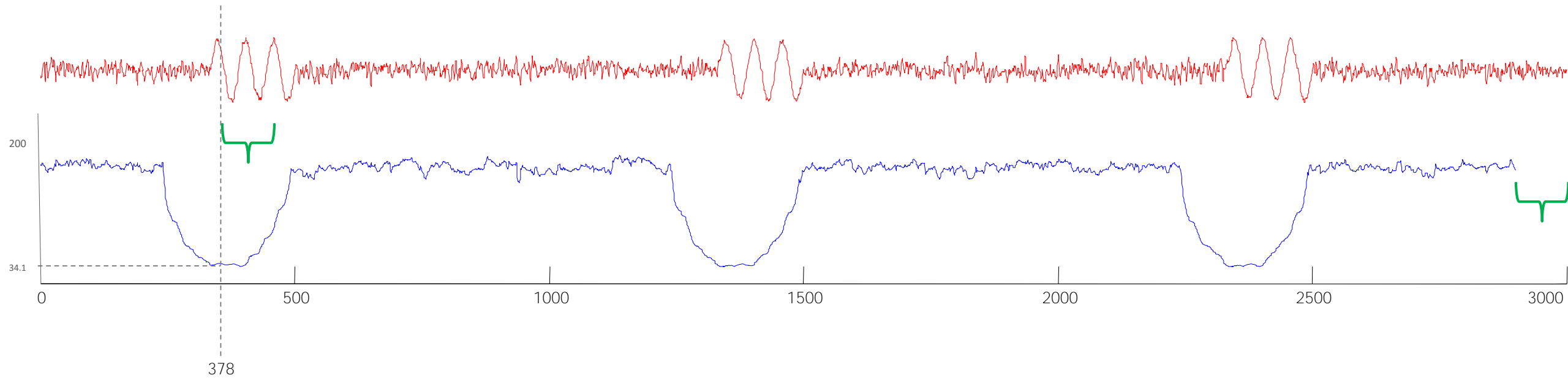
I have created a companion “time series”, called a [matrix profile](#) (or just profile).

The matrix profile at the i^{th} location records the distance of the subsequence in T , at the i^{th} location, to its nearest neighbor.

For example, in the below, the subsequence starting at 921 happens to have a distance of 177.0 to its nearest neighbor (wherever it is).

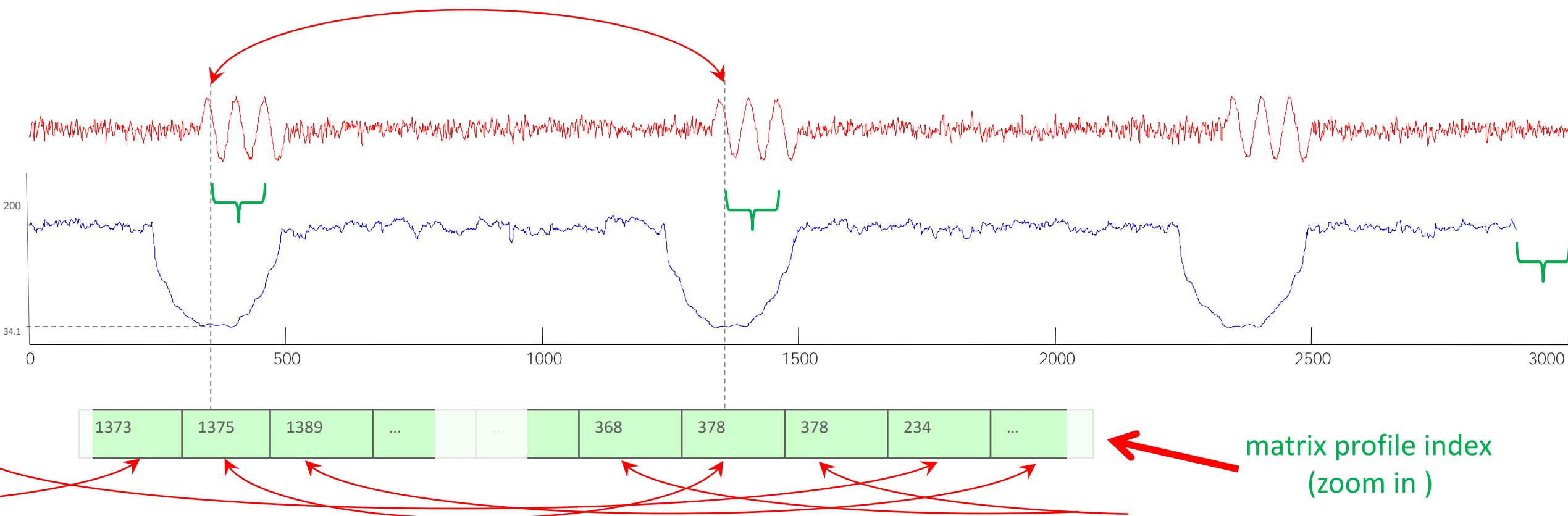


Another example. In the below, the subsequence starting at 378 happens to have a distance of 34.2 to its nearest neighbor (wherever it is).



I have created another companion sequence, called a **matrix profile index**.

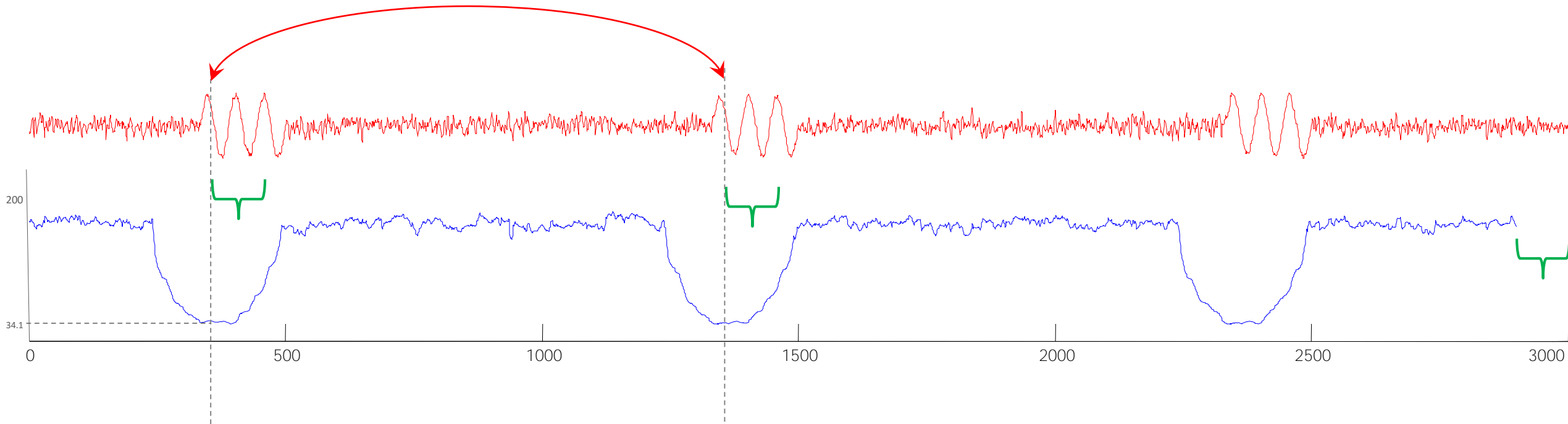
In the following slides I won't bother to show the **matrix profile index**, but be aware it exists, and it allows us to find the nearest neighbor to any subsequence in constant time.



You may have realized that computing the **matrix profile** is very expensive!

If a single Euclidian distance calculation takes 0.0001 seconds, then computing the **matrix profile** for tiny dataset below takes 7.5 minutes! We will come back to this issue later.

$$((3000 * 2999) / 2) * 0.0001 \text{ seconds} = 7.49 \text{ minutes}$$



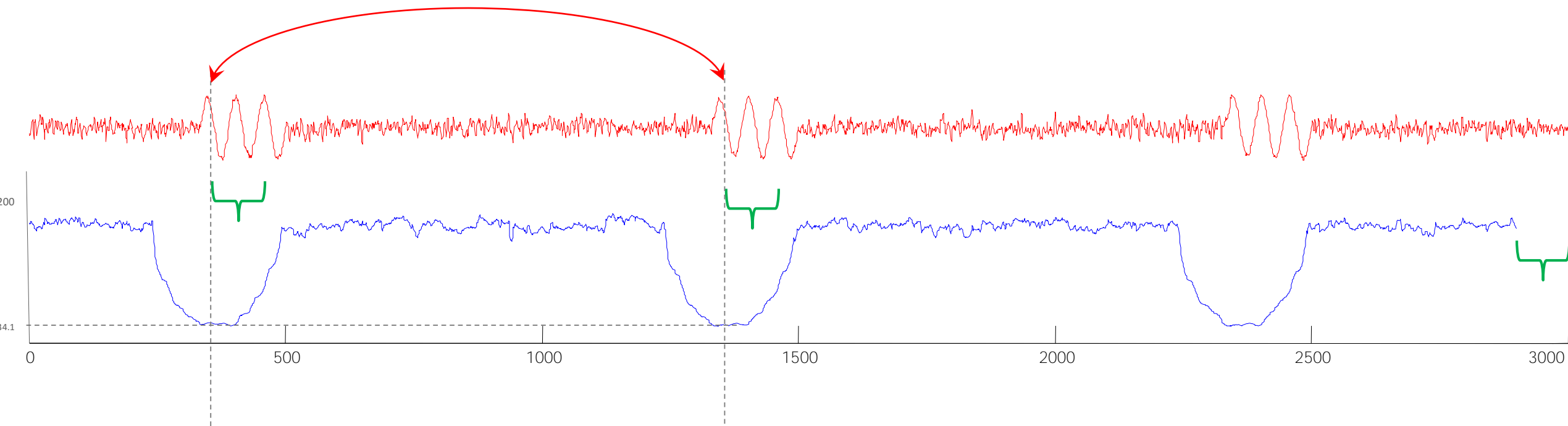
Overarching Claim

- *Given* the Matrix Profile, then virtually every time series data mining task is either trivial or easy.
- In next few slides I will show examples for...
 - Motif Discovery
 - Anomaly Detection (Discord Discovery)
 - Joins (Both self joins, and AB-Joins)
- ..but the same is true for Classification, Clustering, Semantic Segmentation, Visualization, Density Estimation and Rule Discovery.

The matrix profile has some interesting properties...

First, the pair of lowest values (it must be a tying pair) are the *time series motif*.

Other definitions of motif can be found quickly using the matrix profile (discussion omitted)



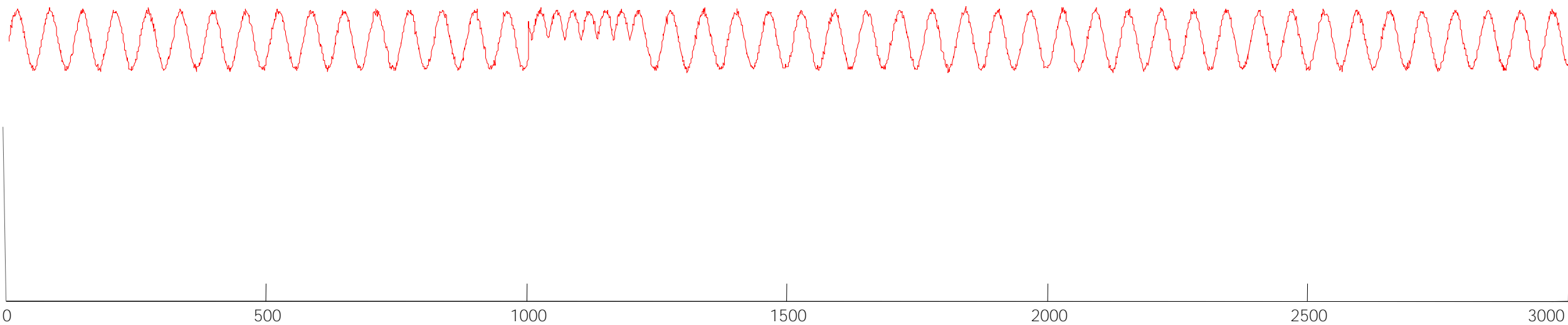
I will show some other, more exciting examples of motifs later...

The matrix profile has some interesting properties...

Second, the highest values corresponds to the **time series discord** (an anomaly)

To see this, let us consider another dataset. Below is a slightly noisy sine wave. I have added an anomaly by taking the absolute value in the region between 1,000 and 1,200.

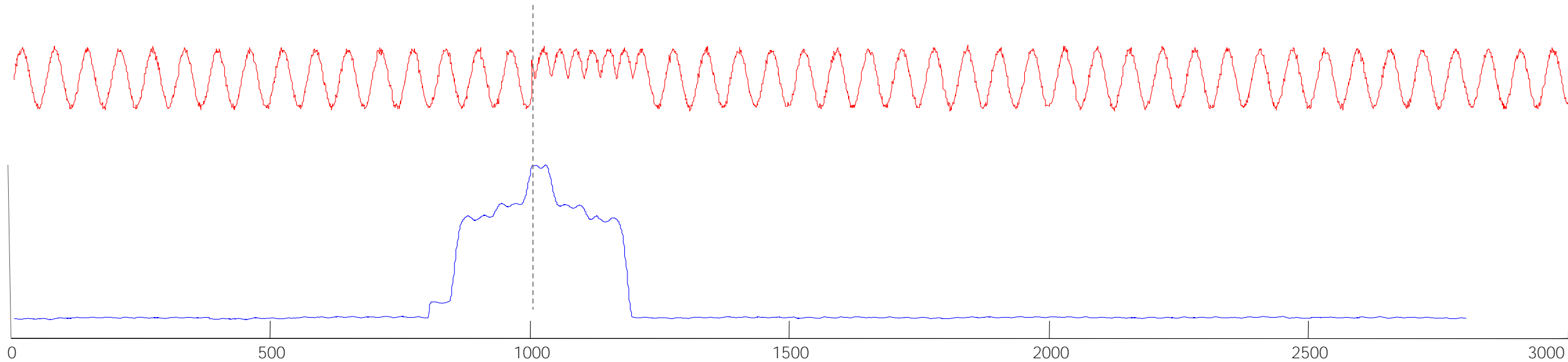
What would the matrix profile look like for this time series? (next slide).



The matrix profile has some interesting properties...

Second, the highest values corresponds to the **time series discord** (an anomaly).

The matrix profile strongly encodes (“peaks at”) the anomaly.

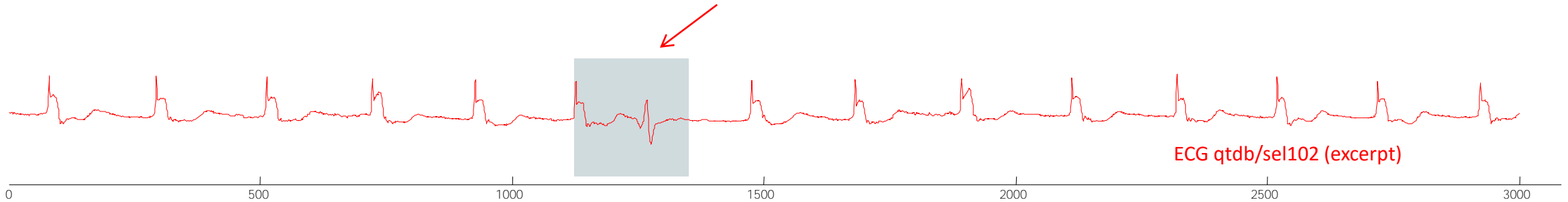


Before Moving On

- I want to show you that the nice intuitive properties of the matrix profile are not limited to clean synthetic data.
- Let quickly us see examples in real data....
 - one example of discords (ECG data)
 - one example motifs (Industrial data)

Matrix Profiles as Anomaly Detectors: 1 of 2

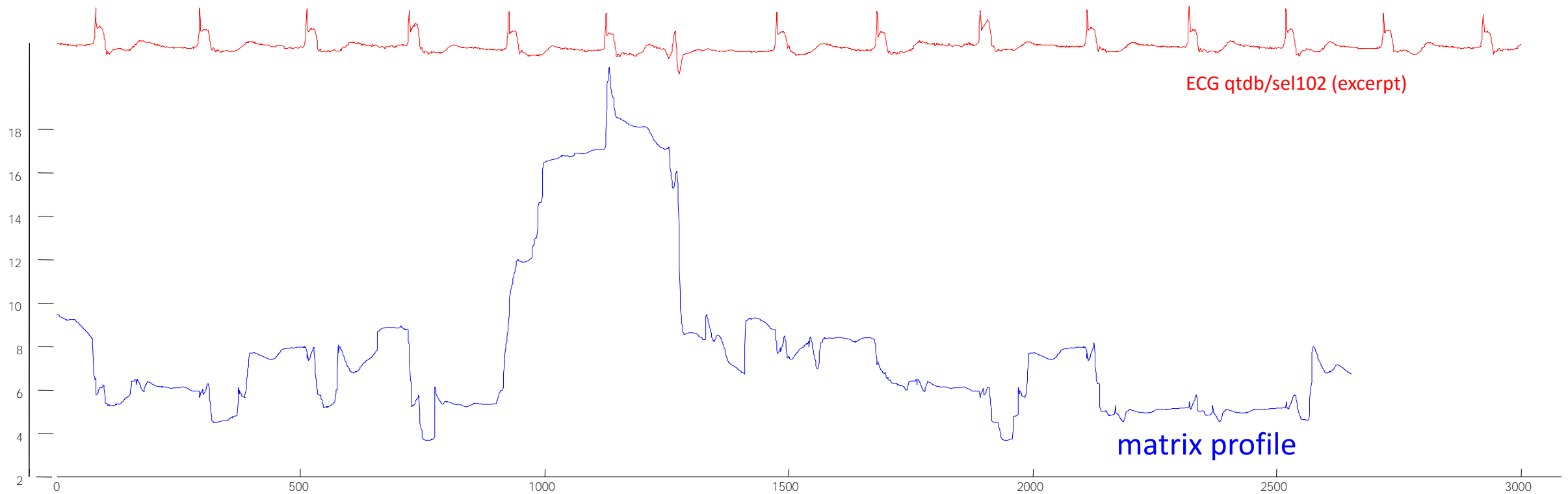
An anomaly, a premature ventricular contraction



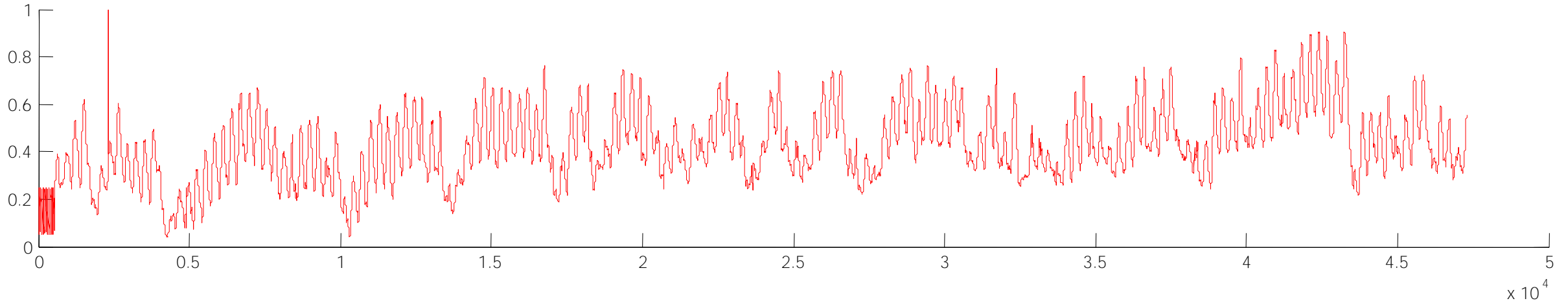
Let us use a [matrix profile](#) to see if we can spot this anomaly (next slide)

Matrix Profiles as Anomaly Detectors: 2 of 2

The alignment of the peak of the matrix profile and the ground truth is sharp and perfect!



Motif Discovery: Industrial Data:



This is *real* industrial data I have worked on. However, I have changed some details to comply with an NDA.

The data is about six months long, and is annotated (not shown) by the quality of the yield produced.

We ran the data through a tool that computes the matrix profile, then extracts the top three motifs sets, and the top three discords.

This is the original time series

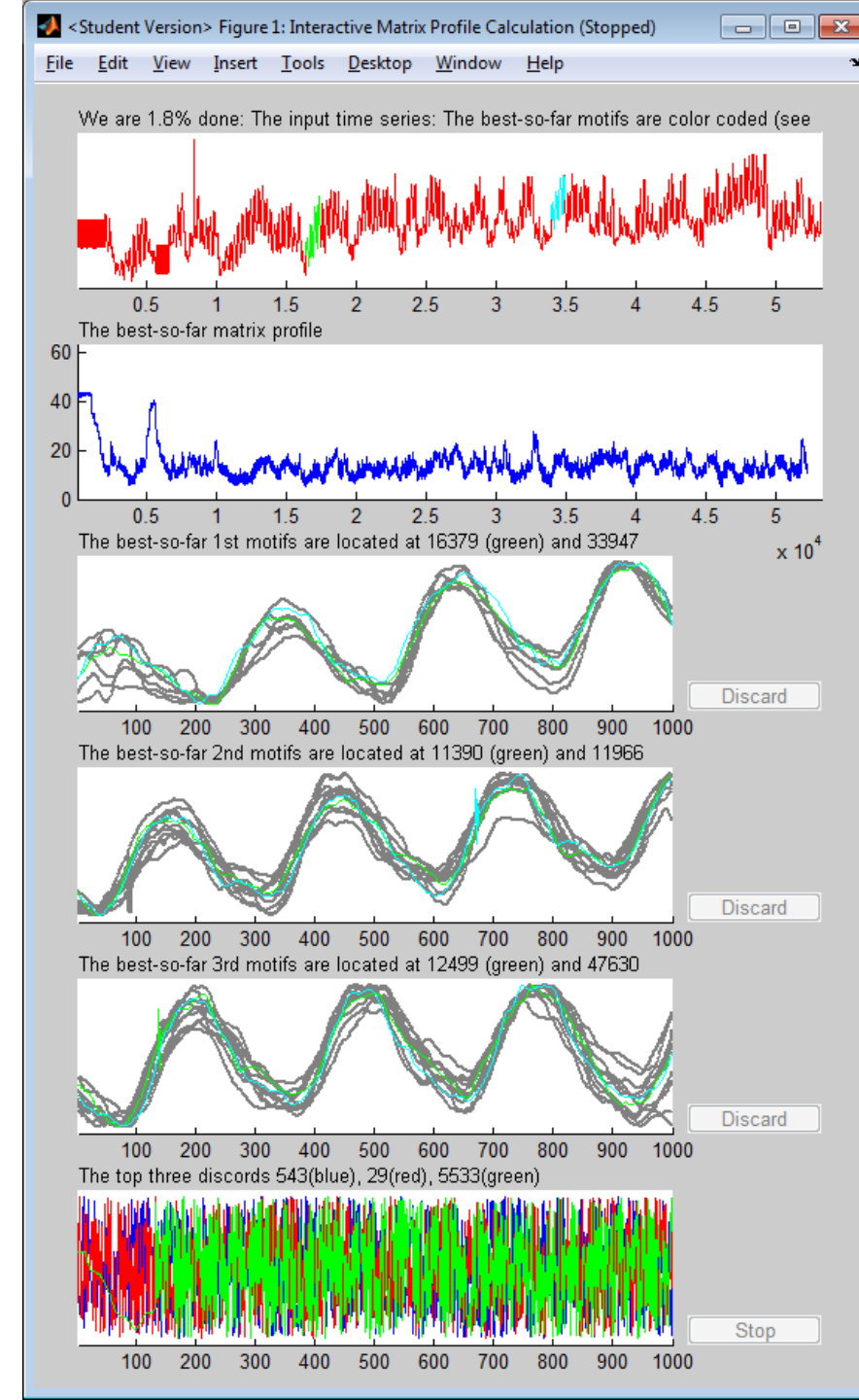
Here is the **matrix profile**

This is the top motif

This is the *second* motif

This is the *third* motif

There are the three most unusual patterns



Note that there appear to be three *regimes* discovered

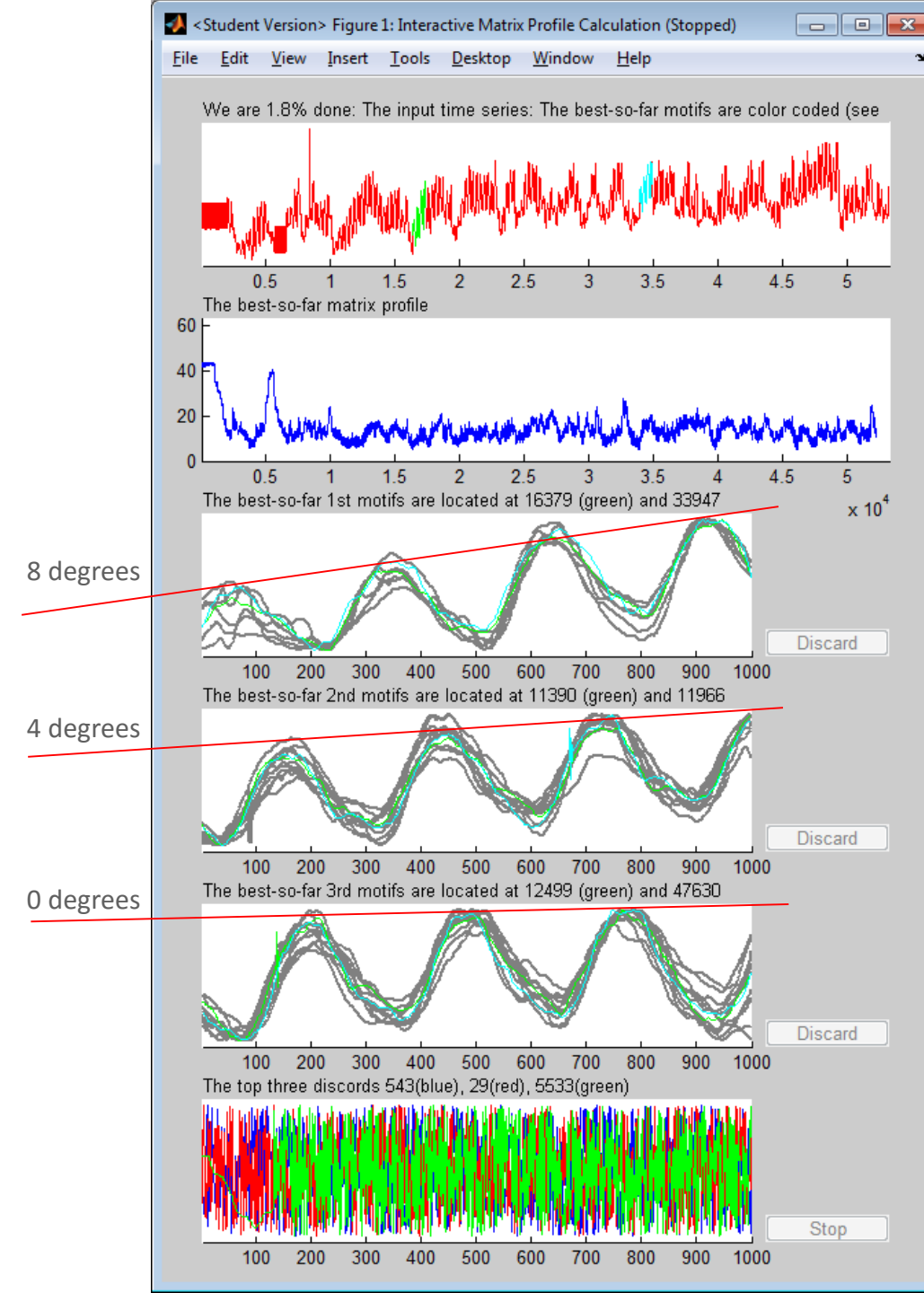
- A. An 8-degree ascending slope
- B. A 4-degree ascending slope
- C. A 0-degree constant slope

(everything above this line is **true**, below this line is speculation or obfuscated for privacy)

We can now ask are the regimes associated with yield quality, by looking up the yield numbers on the days in question.
We find..

A = {bad, bad, fair, bad, fair, bad, bad}
B = {bad, good, fair, bad, fair, good, fair}
C = {good, good, good, good, good, good, good}

So yes! This patterns appear to be precursors to the quality of yield (we have not fully teased out causality here). So now we can monitor for patterns “B” and “A” and sound an alarm if we see them, take action, and improve quality.

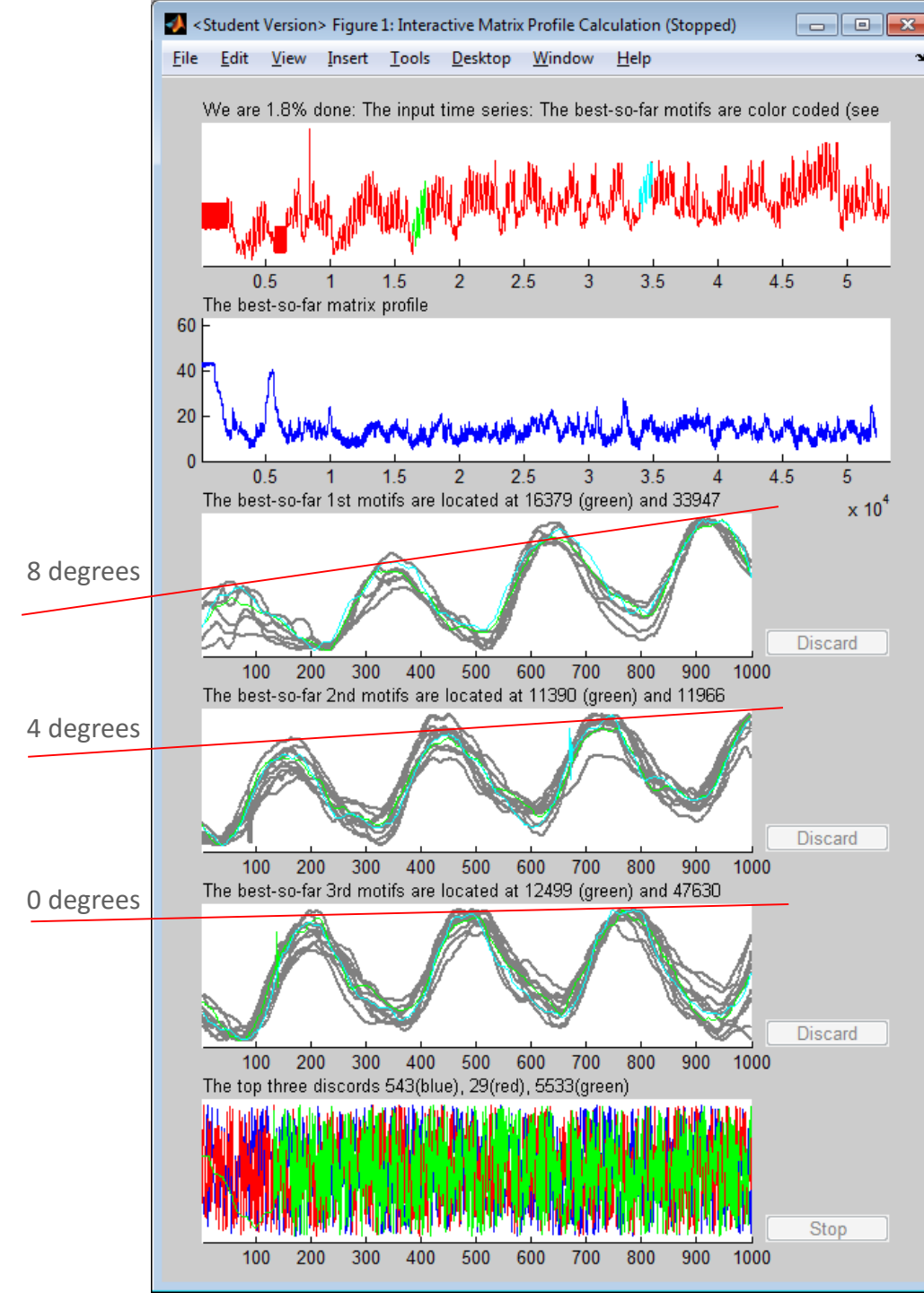


In passing, how long does this take?

If done in a brute-force manner, doing this would take 144 days.

Say each Euclidean distance comparison takes 0.0001 seconds.


$(500000 * ((500000 - 1) / 2) * 0.0001) * \text{seconds} = 144.67 \text{ days}$

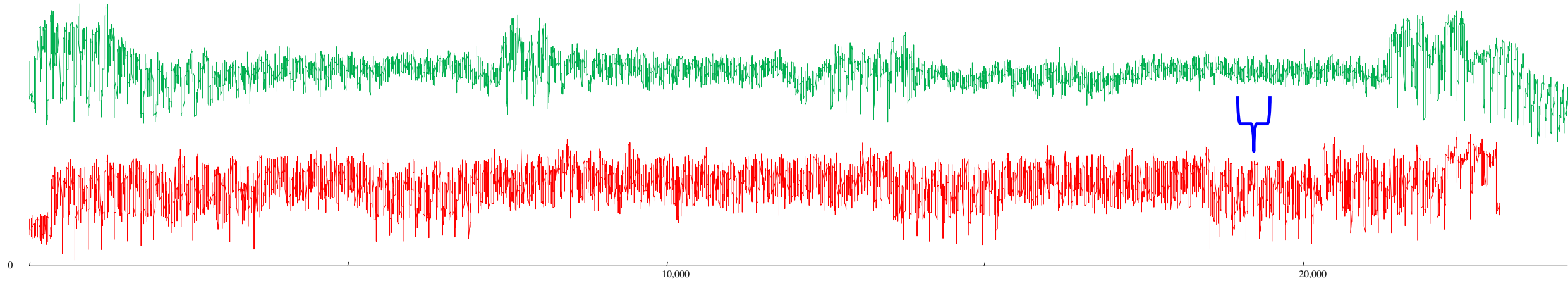


Generalizing to Joins

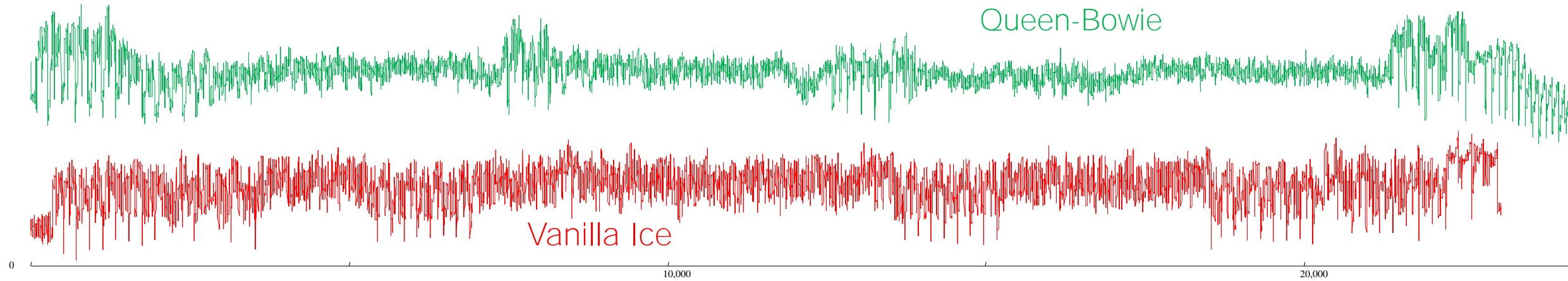
- A Matrix Profile can be seen as a self-join
- It is trivial to generalize it to an AB-join
 - For every subsequence in A, find its closest subsequence in B
 - Note that this is not symmetric in general
- Surprisingly, there is almost no work on time series joins.
- Let us see some trivial examples, then discuss useful applications

Can you see any common structure between the two time series below?

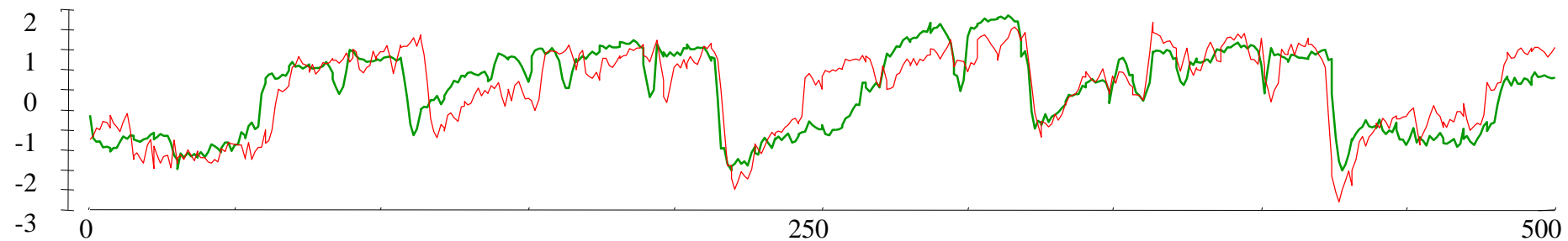
Hint, it is probably about this length 




The data is the 2nd MFCC of two songs, *Under Pressure* and *Ice Ice Baby*

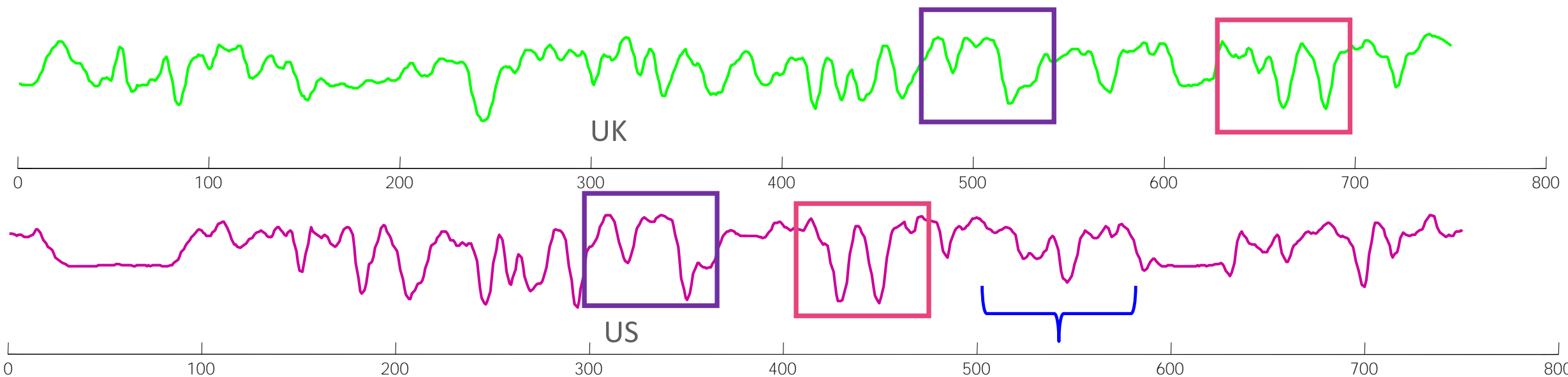


A zoom-in of the best conserved region between the two time series
(similarity join)

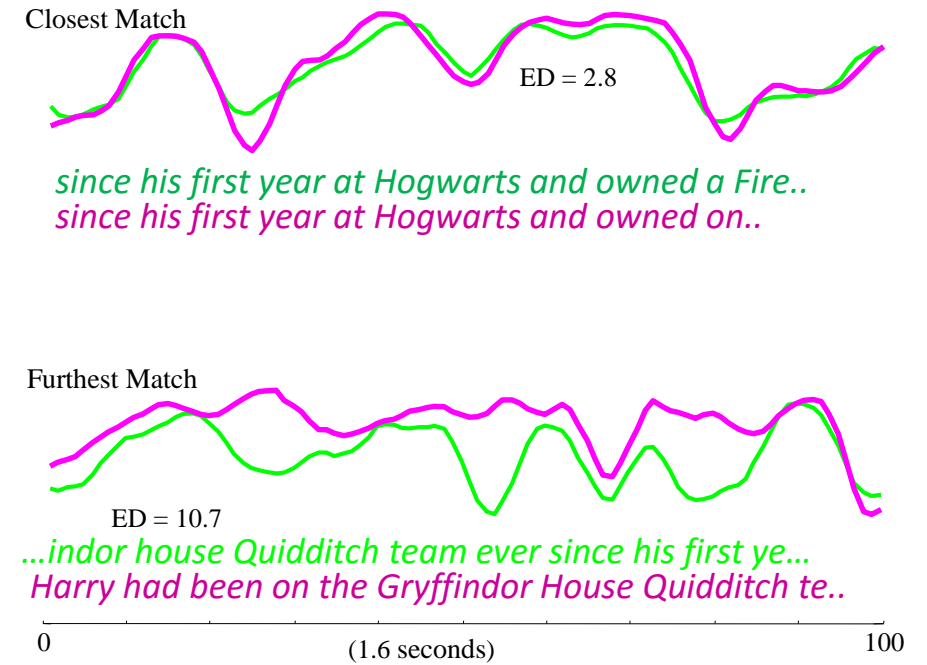


In the previous example I asked you to find “*common structure between the two time series*”
Now I am going to ask you the opposite question.
What is *different* between the two time series?

Hint, it is probably about this length 



Here the difference is due to a unique phrase that only appears in the USA version of the Harry Potter books.

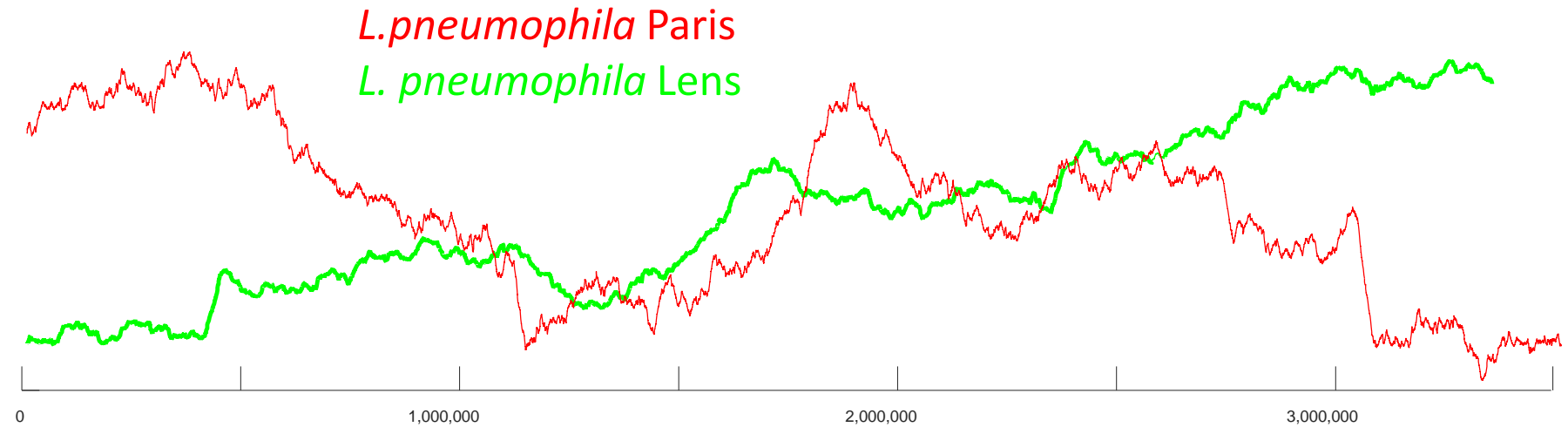


UK version : *Harry was passionate about Quidditch. He had played as Seeker on the Gryffindor house Quidditch team ever since his first year at Hogwarts and owned a Firebolt, one of the best racing brooms in the world...*

USA version : *Harry had been on the Gryffindor House Quidditch team ever since his first year at Hogwarts and owned one of the best racing brooms in the world, a Firebolt.*

It is possible to convert DNA to time series.
Here we converted two of the 180 known strains of Legionella, *L. pneumophila* Paris and *L. pneumophila* Lens, which consist of 3,503,504 and 3,345,567 bp respectively.

On a hunch, lets flip one of them left to right, then join them... (next slide)

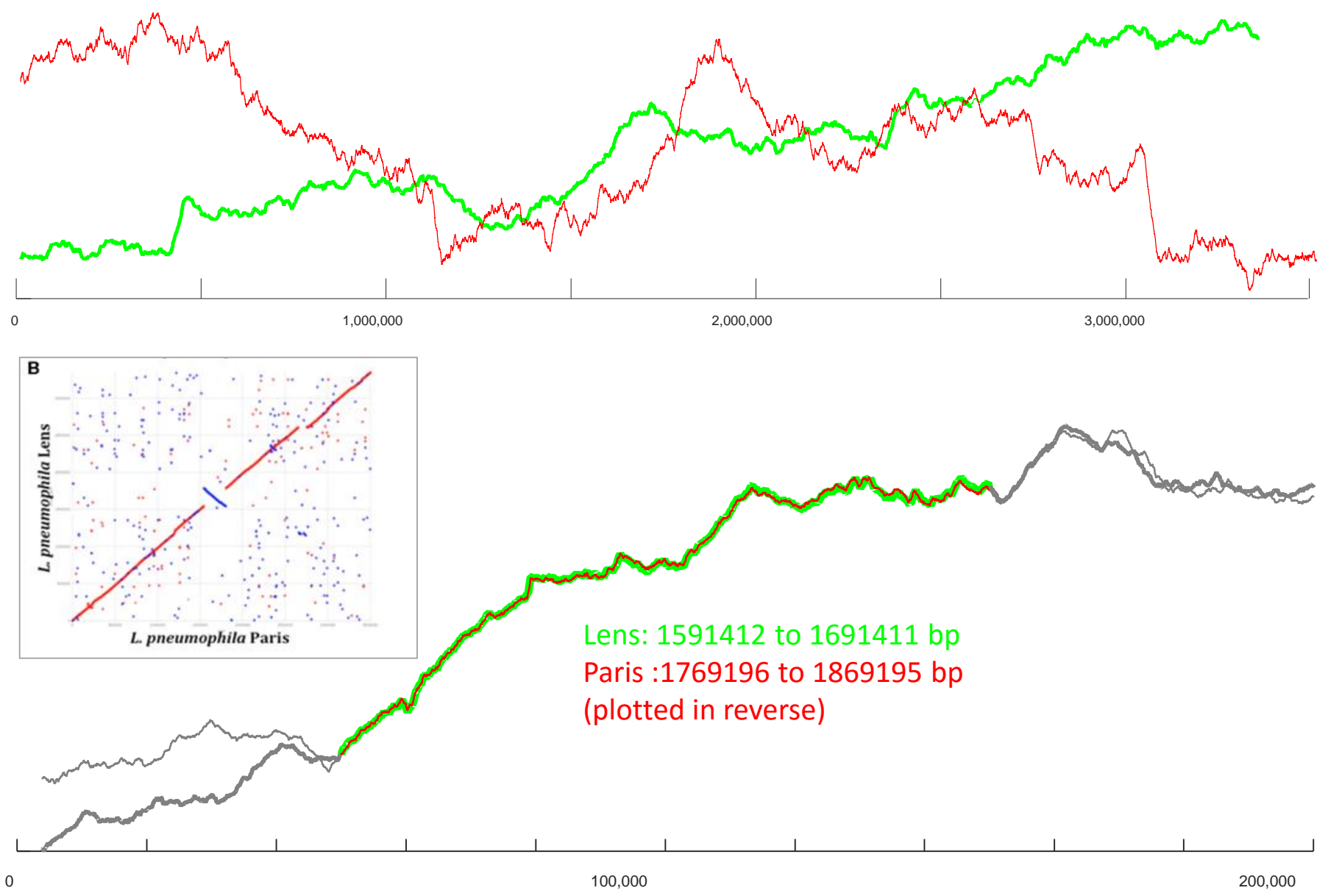


Real-valued similarity joins normally scale very poorly in dimensionality. A dimensionality of 40 is much harder than a dimensionality of 20.

Here the dimensionality was 100,000!!

Moreover, they scale poorly on dataset size, here the data sizes are of 3,503,504 and 3,345,567.

How was this possible?



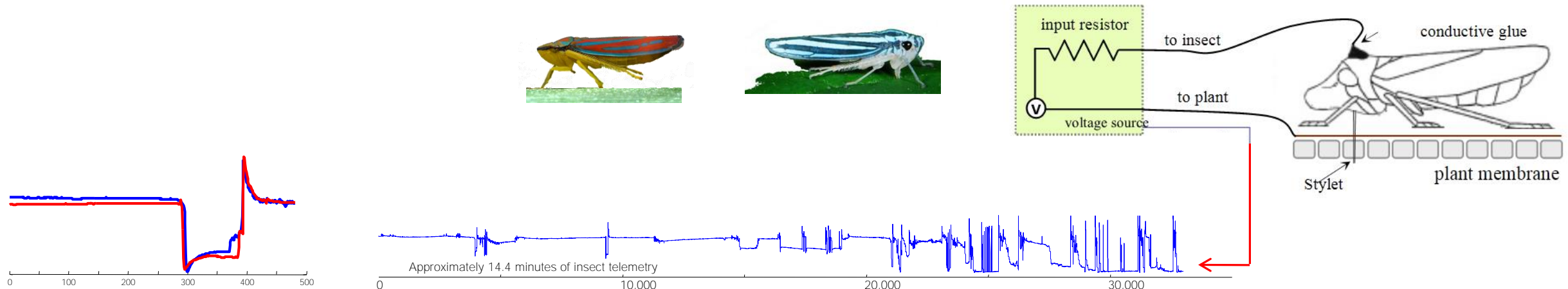
The Utility of Joins

I believe that time series joins are *the* killer app

- Given two insects..

or two patients, or two processing runs, or two space shuttle launches, or two golf swings, or two ad campaigns, or two medical interventions...

- What is conserved, what is different?



Computing the Matrix Profile

Computing the Matrix Profile with a brute force algorithm takes $O(n^2m)$

We have an algorithm, STOMP, that takes $O(n^2)$.

Because (recall the DNA example) m can be 100,000, this is a significant speed-up.

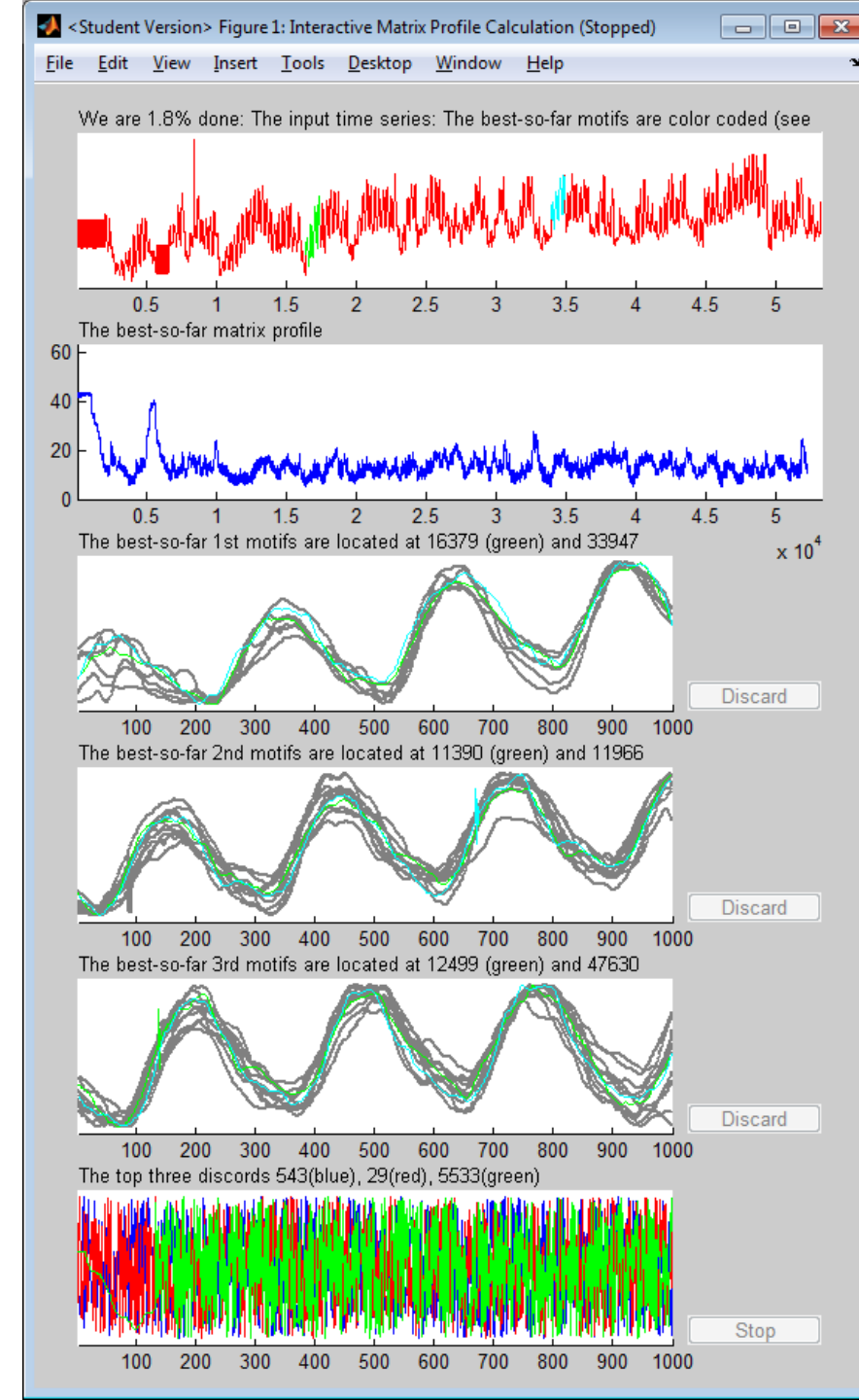
But wait! There's more!

- We can cast our algorithm in an *anytime* framework, making it even faster by a factor of about 100.
- Once the Matrix profile is computed, we can maintain it at 20Hz-plus forever (as an implication, this means we have invented the first *exact* online motif discovery algorithm, the first *exact* online discord discovery algorithm)
- It can trivially exploit hardware, such as GPUs, cloud computing etc.
- Lets put all this into perspective (next few slides)

Remember this example?

We said it would take 144 days, if done in a brute-force manner. We did this in 4 seconds (cheap desktop machine). We can do 99% of the datasets people care about *interactively*.

As the time series has about 500,000 datapoints, to produce the matrix profile we have to **compute one hundred twenty-four billion, nine hundred ninety-nine million, seven hundred fifty thousand** pairwise calculations.

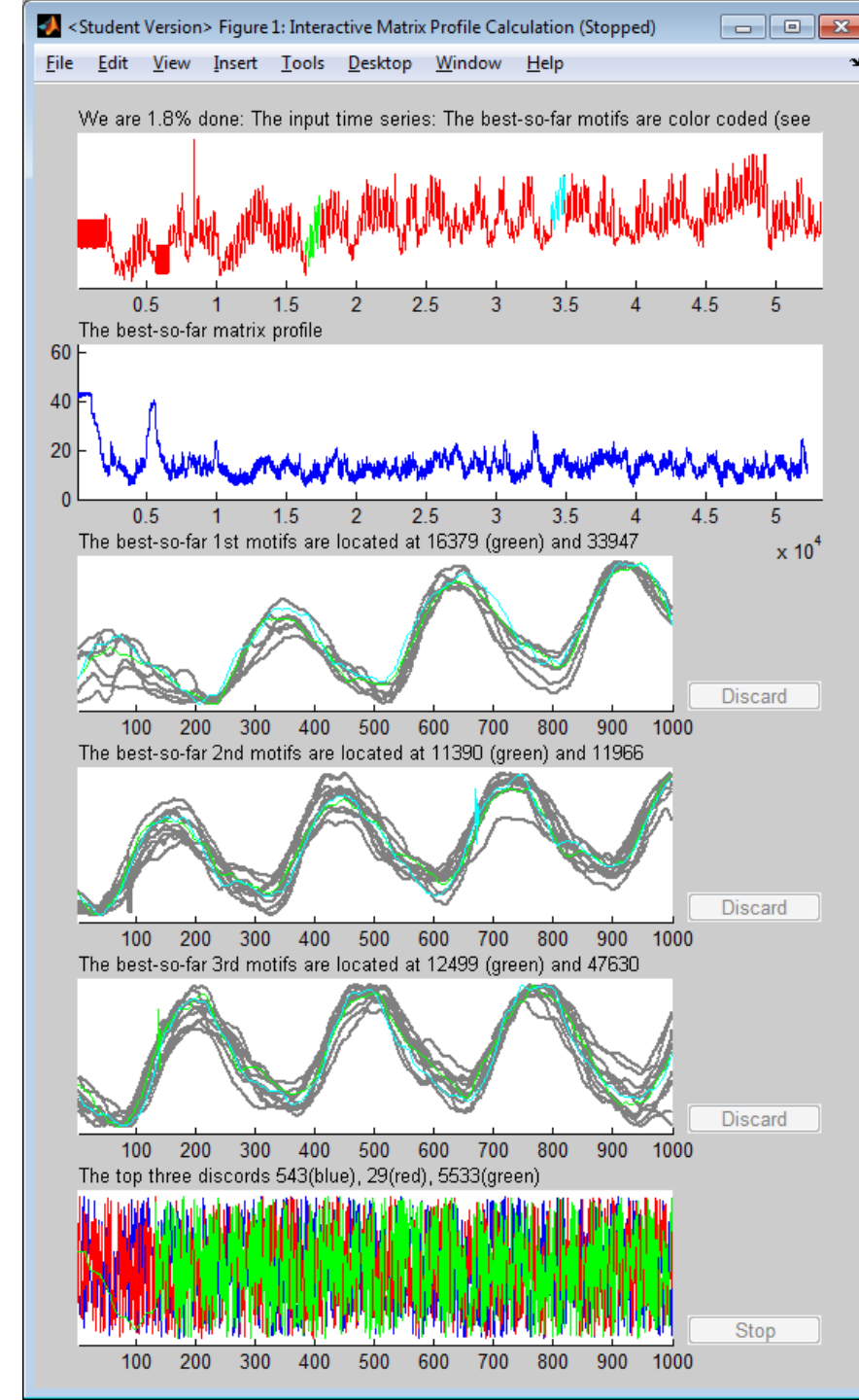


Remember this example?

We said it would take 144 days, if done in a brute-force manner, we did this in 4 seconds (cheap desktop). We can do 99% of the datasets people care about *interactively*.

As the time series has about 500,000 datapoints, to produce the matrix profile we have to **compute one hundred twenty-four billion, nine hundred ninety-nine million, seven hundred fifty thousand** pairwise calculations.

That sounds like a lot, but we have recently done **four hundred ninety-nine quadrillion, nine hundred ninety-nine trillion, nine hundred ninety-nine billion, five hundred million** pairwise comparisons. This is surely the largest *exact* join ever attempted.



Conclusions (to be followed by one more example)

We have introduced a new data structure for time series, the Matrix Profile.

We have heard the overarching claim: *Once you have the Matrix Profile, all time series data mining tasks become trivial.*

We have seen examples in Motif Discovery, Anomaly Detection and Joins, and you will take my word for Classification, Clustering,, Density Estimation, Visualization, Semantic Segmentation and Rule Discovery (or ask the see the cool examples)

We heard (in passing) about STAMP, STOMP and STAMP_i, a family of algorithms for computing the Matrix Profile *very* quickly.

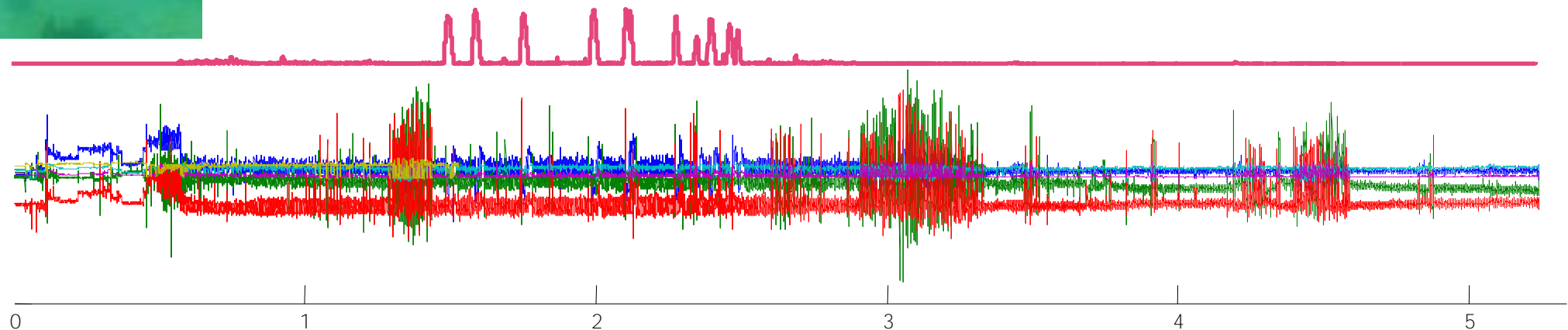
Let us conclude with one final example, that highlights typical interactions with data....

Penguin Telemetry Case Study

This is a very common situation. We are given a data “dump”, with almost no context.

How can we make sense of this data?

We can interactively explore it with our Matrix Profile tool....

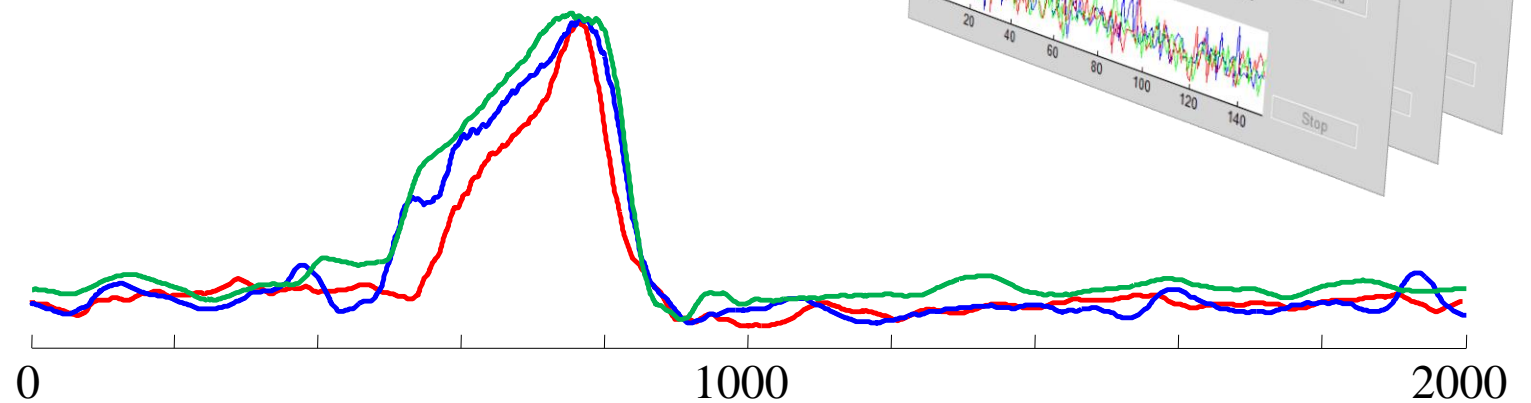


Penguin Telemetry

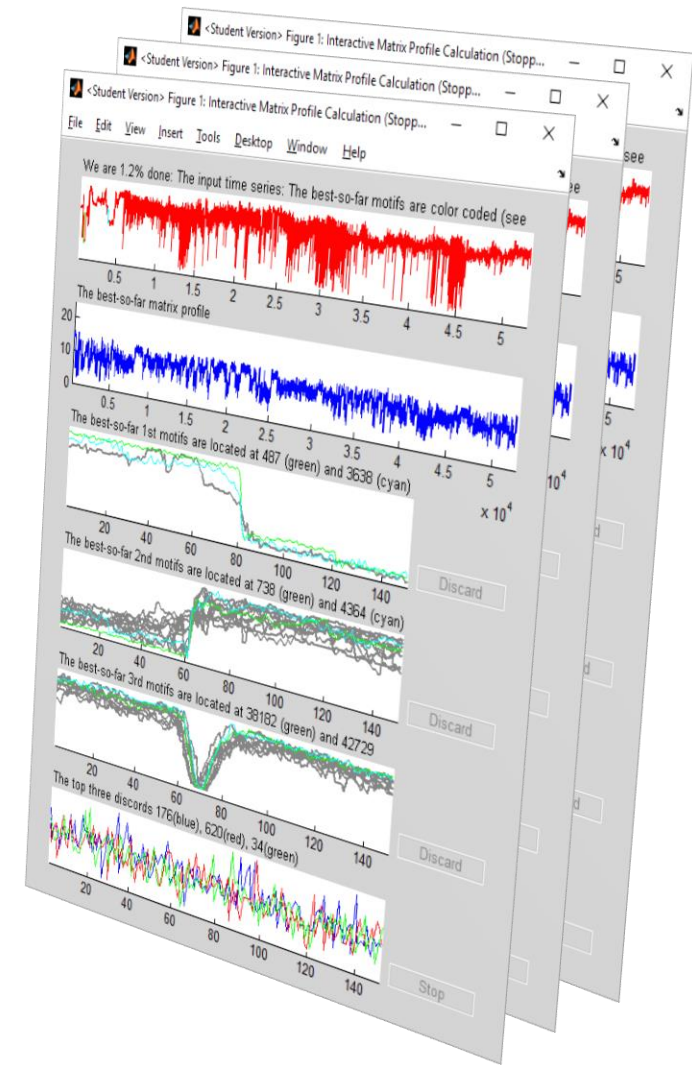
With just five minutes of “playing” with the data, I found a stunning regularity.

A few seconds before each dive, the penguin performs this “shark-fin” like behavior.

Thus, I have found a precursor rule...



Now I am done



Questions?



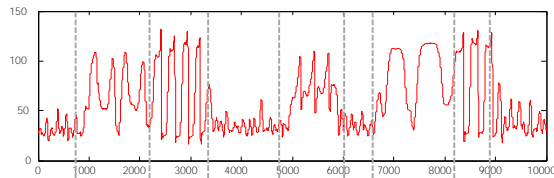
“Regime Change” (Semantic Segmentation)

There is a problem that has seen a lot of interest in recent years (especially in human behavior domains).

Given a time series (usually multidimensional, but here 1D for simplicity) segment it into discrete behaviors (waking, running, eating etc).

However you have no model of behavior and no domain knowledge ahead of time (maybe just some unlabeled training data).

Claim: We can use the matrix profile/matrix profile index to solve this problem. .. (next slide)



Ground truth



W – walking

S – stretching

P – punching

C – chopping

T – turning

D- drinking

CMU MoCap Subject 86, recording 4 (dimension 30, right radius)

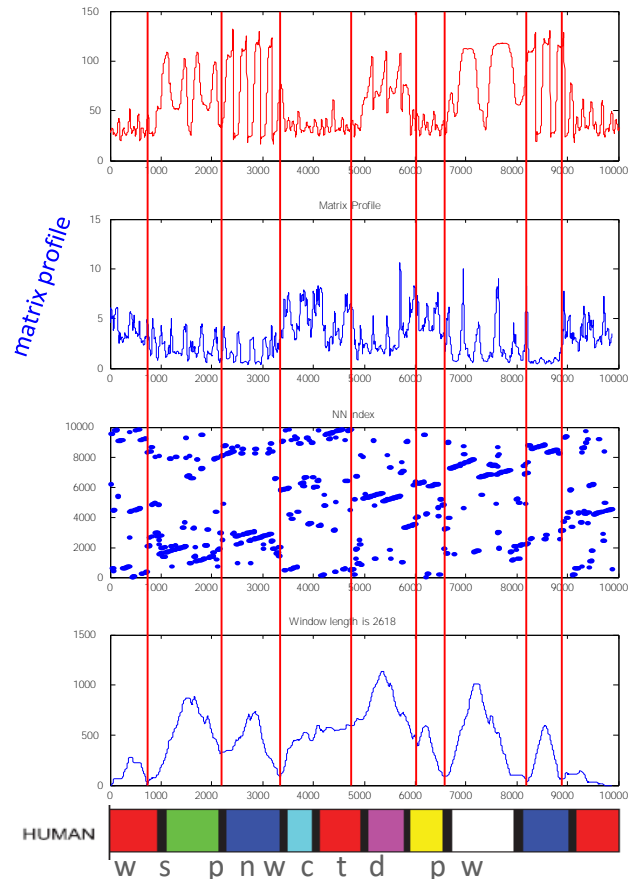
“Regime Change” (Semantic Segmentation)

Our results are *VERY* good compared to the literature.

This is all the more surprising because:

- 1) We are only using a single time series (for now), the right radius. Most paper naturally use many time series (15 or more).
- 2) Most papers train on “Joe”, then test on “John’. In contrast we are doing *no* training. We don’t know if the data is people or machines or autos etc.
- 3) We are parameter-free
- 4) We are faster (at least for this short example)

How do we do it? Next slide



Our solution

We claim that the bottom of valleys correspond to regime changes.

“Regime Change” (Semantic Segmentation)

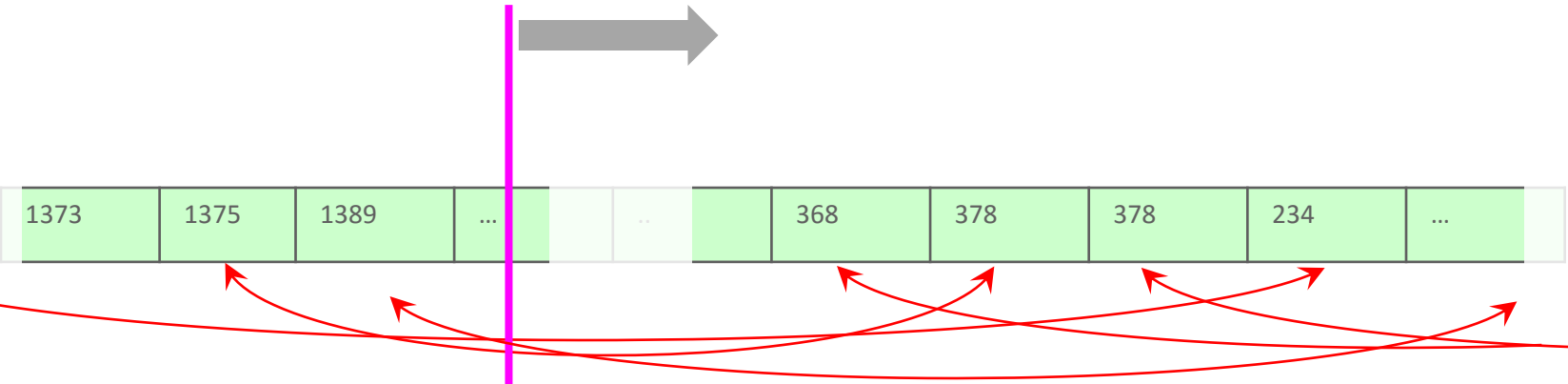
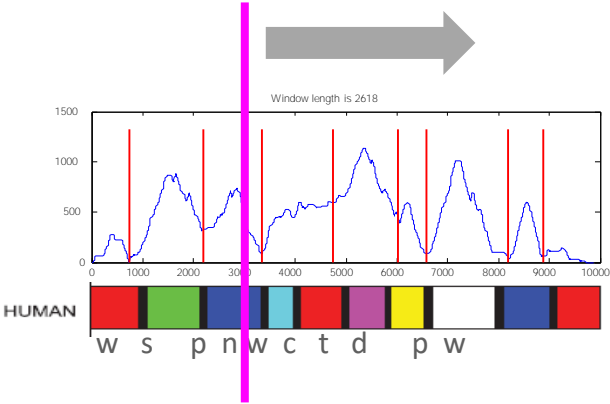
How do we do it?

We did it with an incredibly simple and fast algorithm (*fast* given that we have computed the matrix profile and the index)

Slide the pink line across the index, the number of “arrows” that cross it is the height of the Regime Change curve.

Why does this work?

walkwalkwalkwalkwalktrottrottrottrottrotfallfallfallfallfall



matrix profile index
(zoom in)