HOT SAX: Efficiently Finding the Most Unusual Time Series Subsequence

Eamonn Keogh
University of California, Riverside

*Jessica Lin
George Mason Univ

Ada Fu
Chinese Univ of Hong Kong

The 5th IEEE International Conference on Data Mining
Nov 27-30, Houston, TX

Anomaly (interestingness) detection

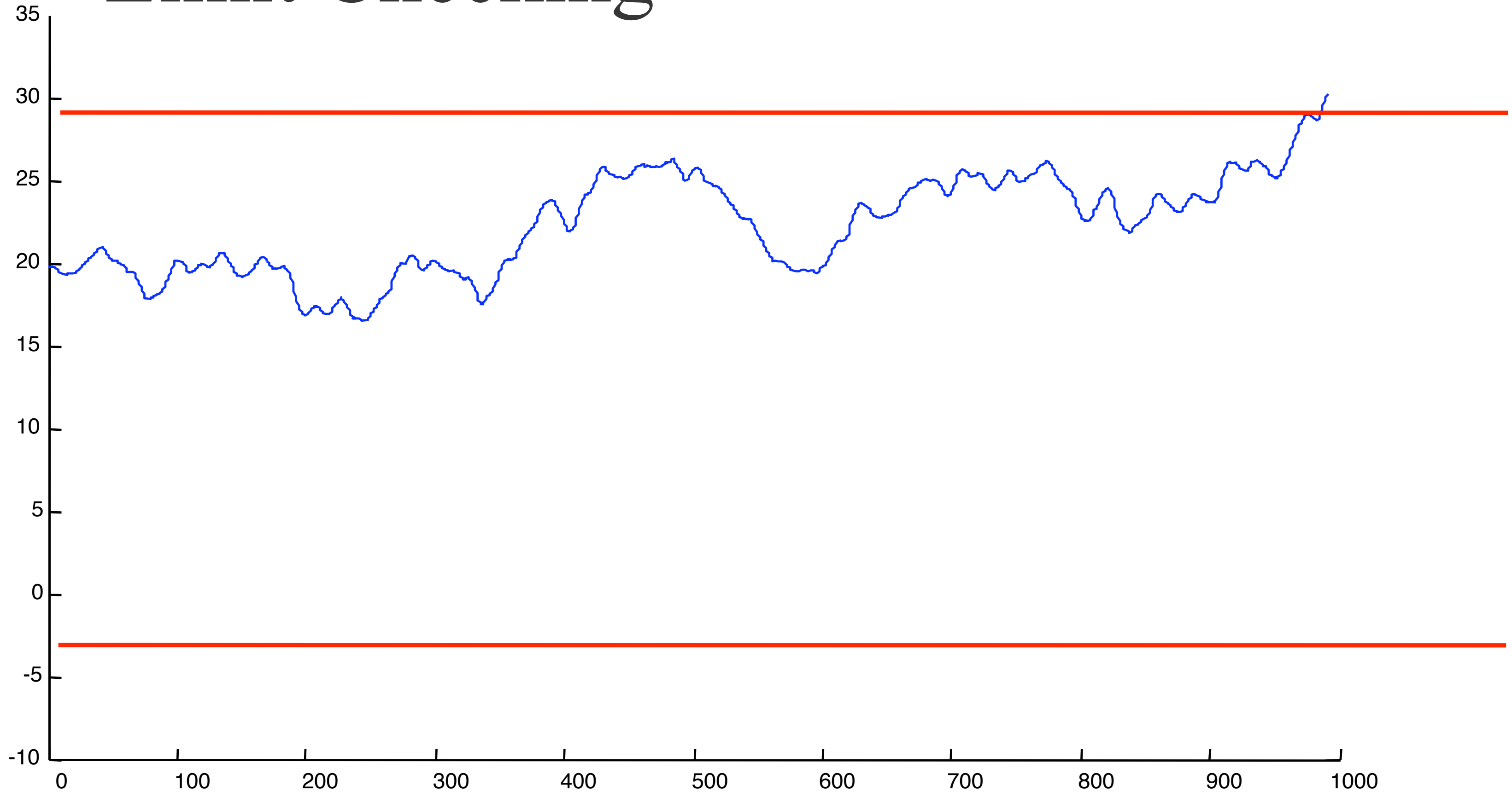
We would like to be able to discover surprising (unusual, interesting, anomalous) patterns in time series.

Note that we don't know in advance in what way the time series might be surprising

Also note that "surprising" is very context dependent, application dependent, subjective etc.

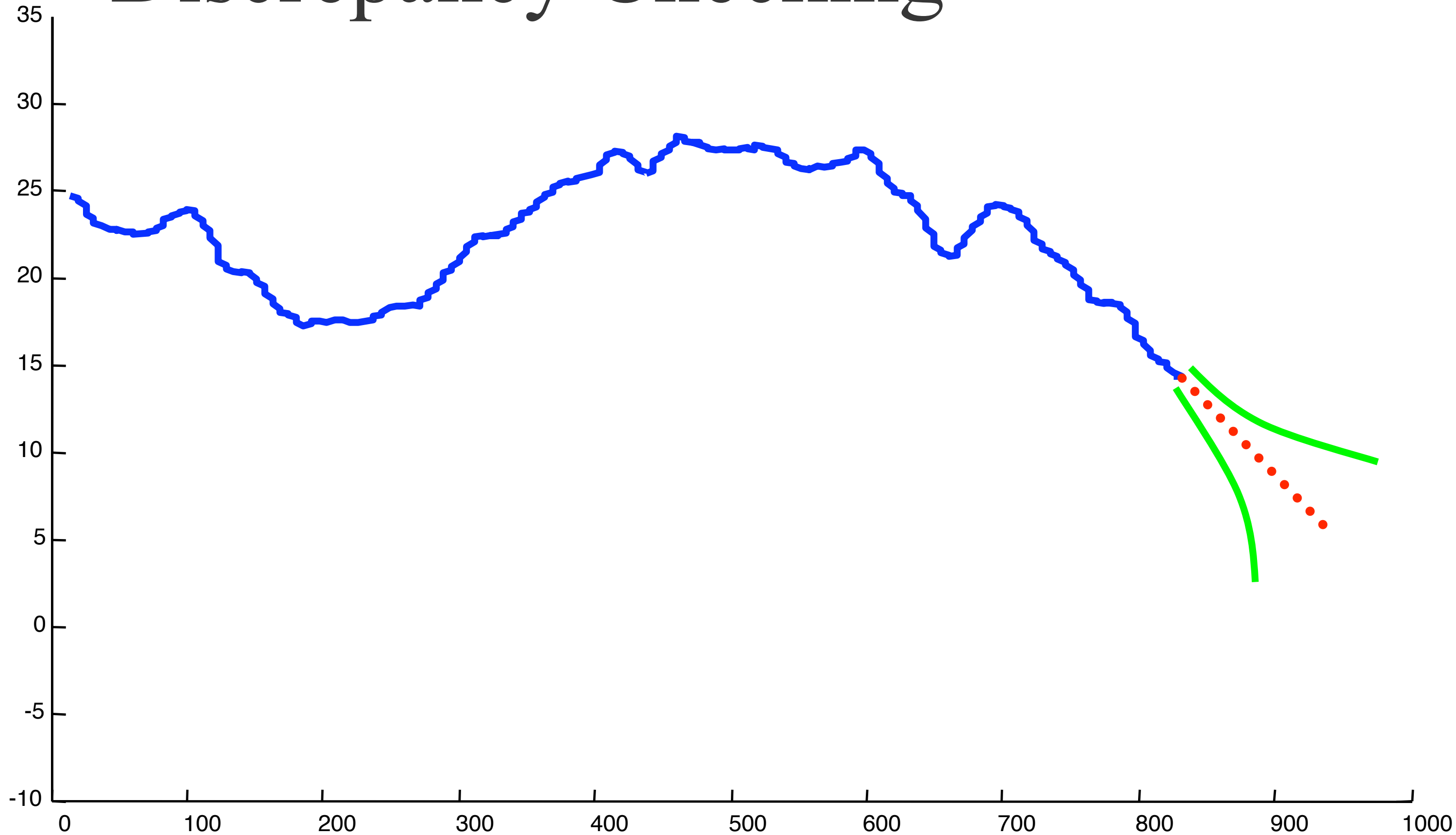
Simple Approaches I

Limit Checking

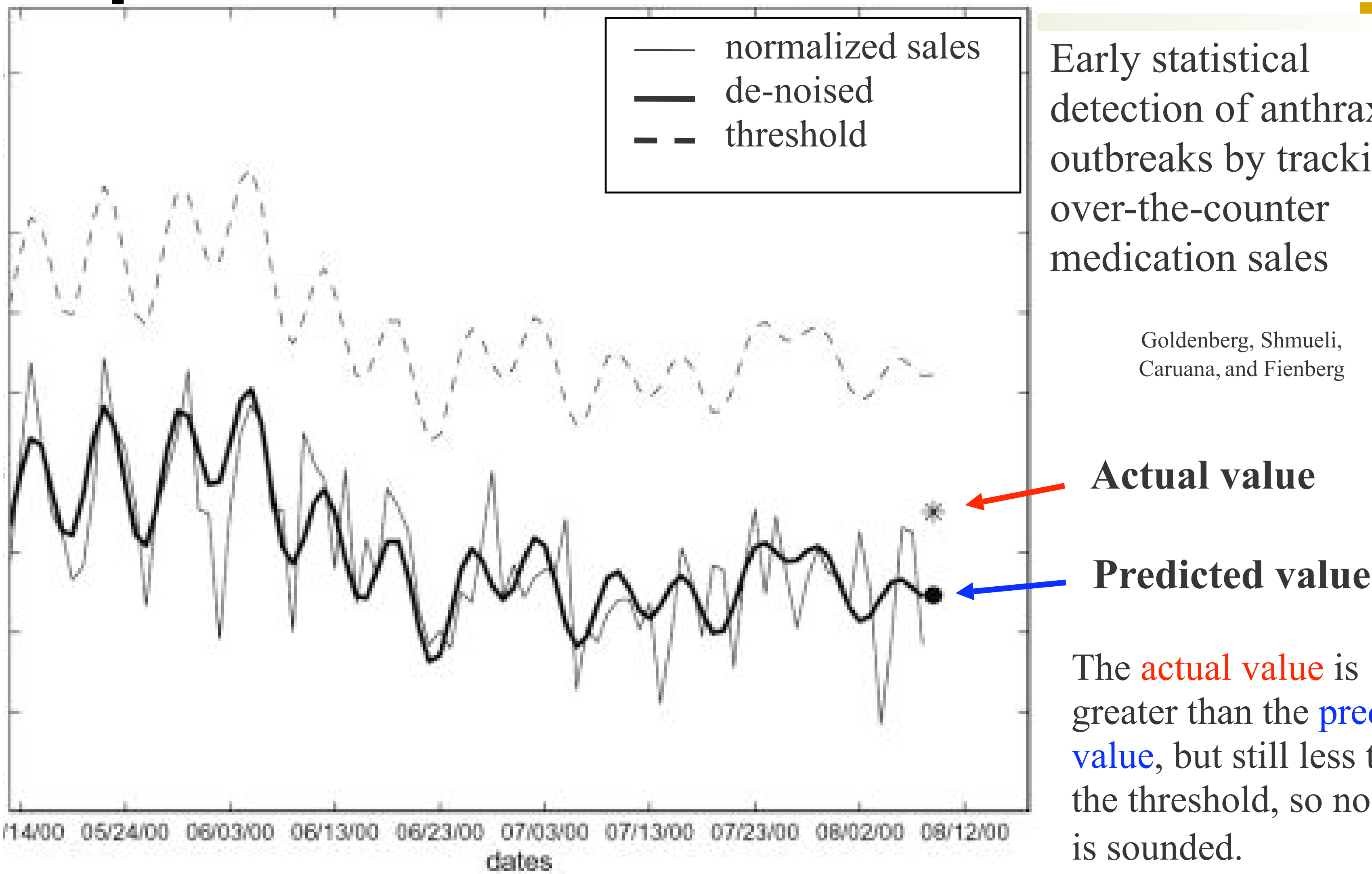


Simple Approaches II

Discrepancy Checking



Discrepancy Checking: Example



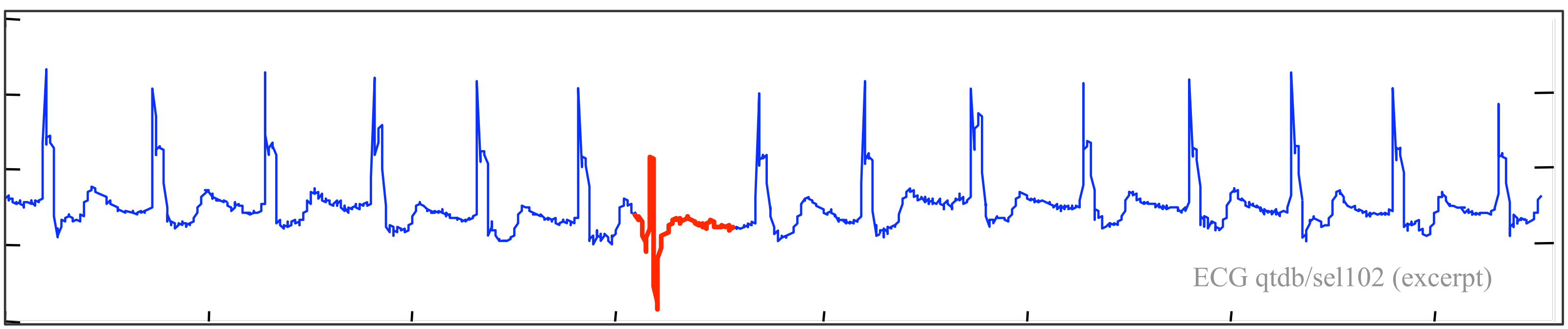
Early statistical detection of anthrax outbreaks by tracking over-the-counter medication sales

Goldenberg, Shmueli, Caruana, and Fienberg

The actual value is greater than the predicted value, but still less than the threshold, so no alarm is sounded.

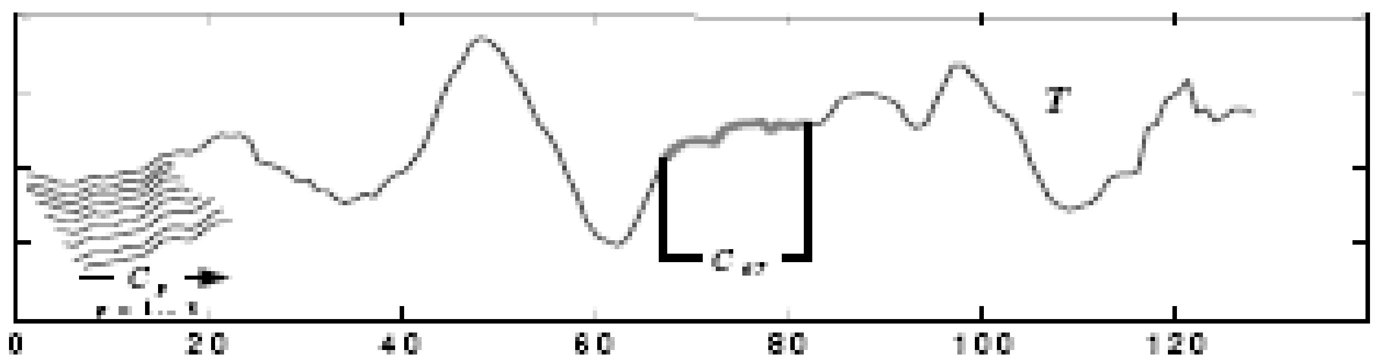
Time Series Discord

- Discord: subsequence that is *least* similar to other subsequences
- Applications:
  - Anomaly detection
  - Clustering
  - Data cleaning



Background – Sliding Windows

- Use a sliding window to extract subsequences



Time Series Discords

- Subsequence *C* of length *n* is said to be the discord if *C* has the largest distance to its nearest non-self match.
- K^{th} Time Series Discord

Non-self Match

- *Non-Self Match*: Given a time series T, containing a subsequence C of length n beginning at position p and a matching subsequence M beginning at q, we say that M is a non-self match to C at distance of $Dist(M,C)$ if $|p - q| \geq n$.

Why is the Notion of Non-self Match Important?

- Consider the following string:

  abcabcabcabcXXXabcabcabacabc

- Annotated string:

  $a_0 b_0 c_0 a_0 b_0 c_0 a_0 b_0 c_0 a_0 b_1 c_1 X_1 X_1 X_1 a_0 b_0 c_0 a_0 b_0 c_0 a_1 \mathbf{b_2 a_1 c_0} a\ b\ c$

- With Non-self match distance:

  $a_0 b_0 c_0 a_0 b_0 c_0 a_0 b_0 c_0 a_0 b_1 c_2 X_3 X_2 X_1 a_0 b_0 c_0 a_0 b_0 c_0 a_1 b_2 a_1 c_0 a\ b\ c$

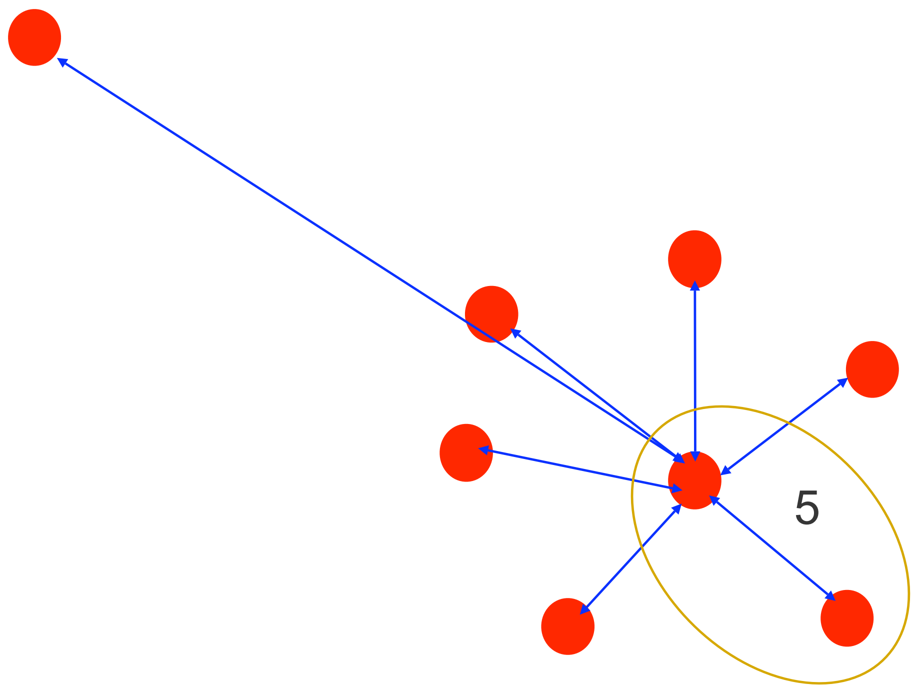
Time Series Discords

- Subsequence *C* of length *n* is said to be the discord if *C* has the largest distance to its nearest non-self match.
- K^{th} Time Series Discord

Finding Discords: Brute-force

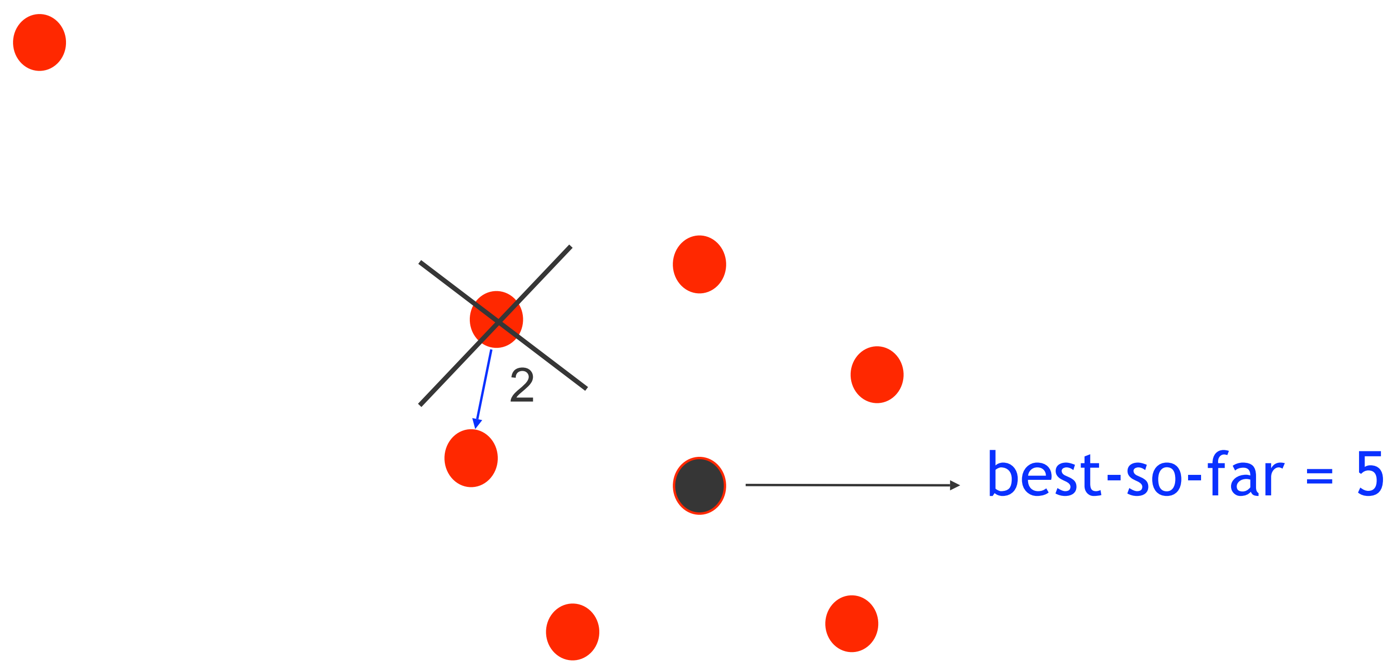
- [outer loop] For each subsequence in the time series, [inner loop] find the distance to its nearest match
- The subsequence that has the greatest such value is the discord (i.e. discord is the subsequence with the farthest nearest-neighbor)
- $O(m^2)$

Example

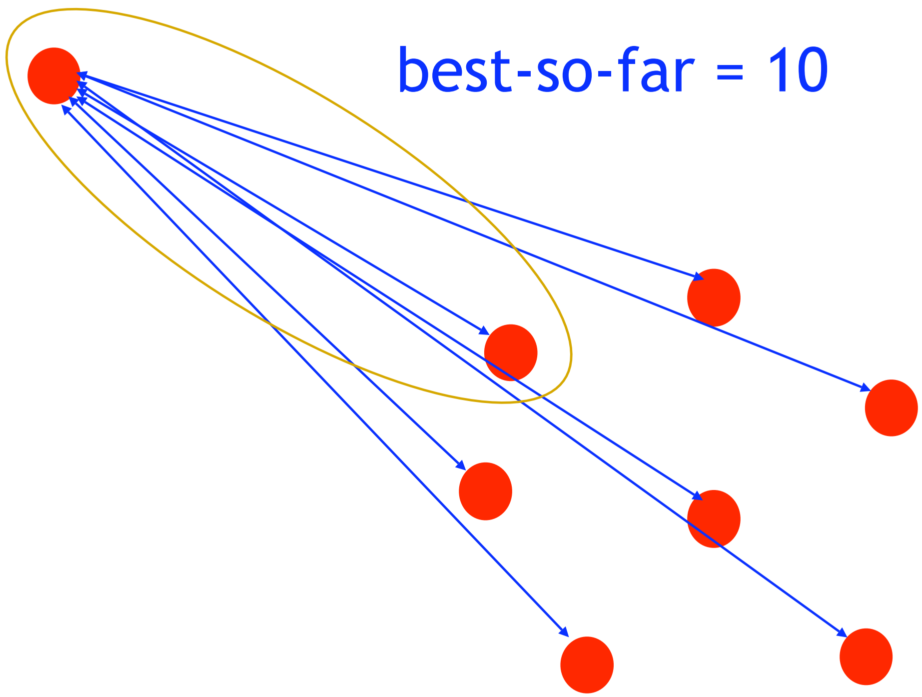


best-so-far = 5

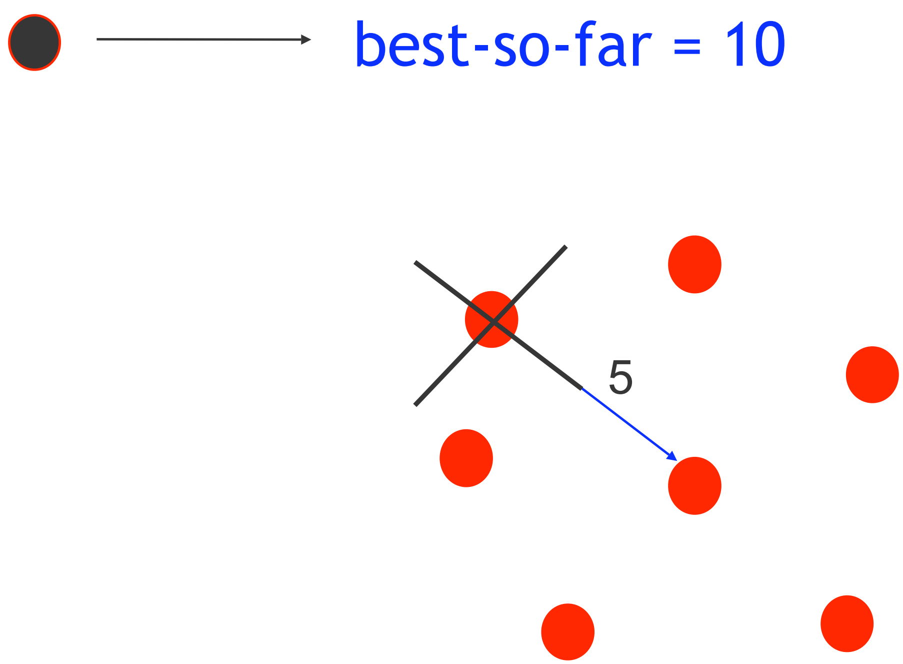
Example



Example - Optimal Ordering



Example - Optimal Ordering



Observations from Brute-Force Alg.

- Our goal is to find the subsequence with the greatest distance to its nearest neighbor
  - We keep track of the best-so-far value
  - In the inner loop, as soon as we encounter a distance < best-so-far, we can terminate the loop
  - Such optimization depends on the orderings of subsequences examined in both the outer and the inner loop

Heuristic Discord Discovery

- Two heuristics:
  - One to determine the order in which the outer loop visits the subsequences
    - invoked once
    - need to be no larger than O(m)
  - One for the inner loop
    - takes the current candidate (from the outer loop) into account
    - invoked for every iteration of the outer loop
    - need to be O(1)

Three Possible Heuristics

- Magic – O(m)
  - Perfect ordering:
    - for outer loop, subsequences are sorted in descending order of non-self match distance to the NN.
    - for inner loop, subsequences are sorted in ascending order of distance to current candidate (from outer)
- Perverse – $O(m^2)$
  - Reverse of Magic
- Random - O(m) ~ $O(m^2)$
  - Random ordering for both outer/inner loops
  - works well in practice

Approximations to Magic

- For the outer loop, we don’t actually need the perfect ordering
  - Just need to ensure that among the first few subsequences examined, one of them has a large distance to its NN
- For the inner loop, we don’t need the perfect ordering either
  - Need to ensure that among the first few subsequences examined, one of them has a small distance to the current candidate (i.e. smaller than best-so-far)

Approximating the Magic Outer Loop

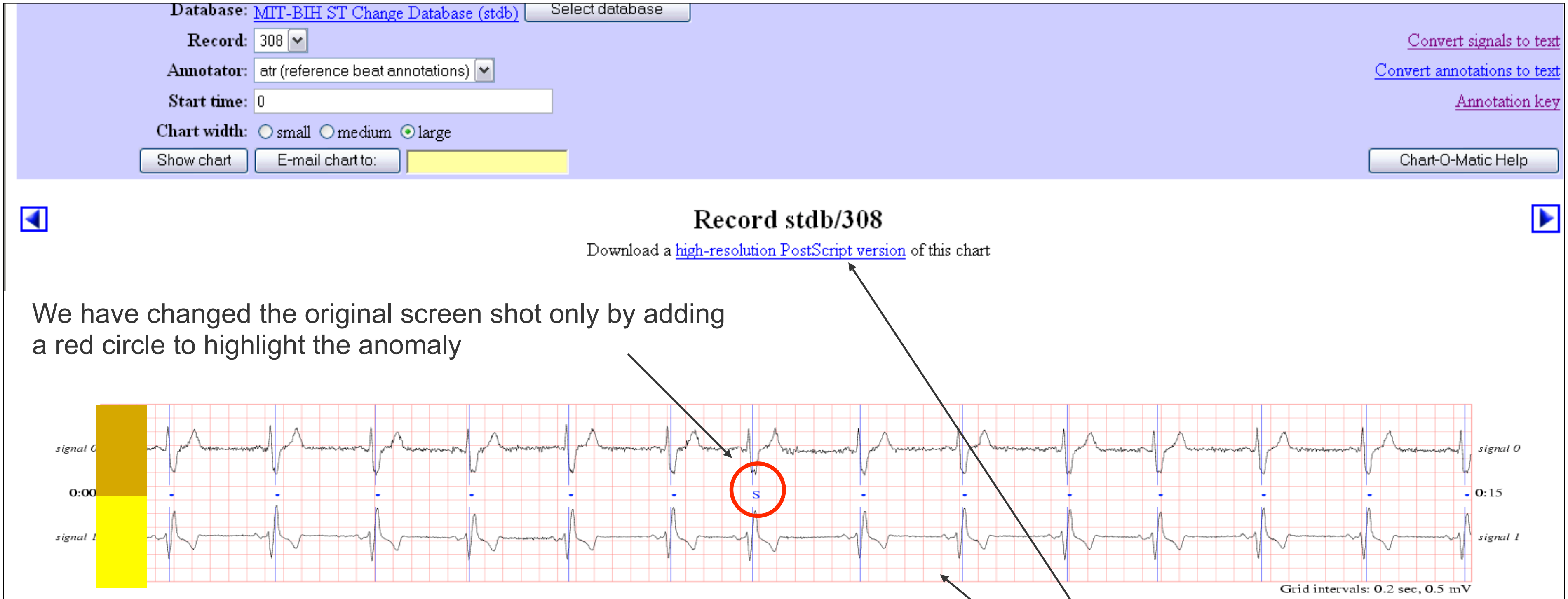
- Scan the counts of the array entries and find those with the smallest count (i.e. mincount = 1)
- Subsequences with such SAX strings (mincount = 1) are examined first in the outer loop
- The rest are ordered randomly
- Intuition: Unusual subsequences are likely to have rare or unique SAX strings

Approximating the Magic Inner Loop

- When candidate j is being examined in the outer loop
  - Look up its SAX string by examining the array
  - Visit the trie and find the subsequences mapped to the same string – these will be examined first
  - The rest are ordered randomly
- Intuition: subsequences that are mapped to the same SAX strings are likely to be similar

HOT SAX

- Because our algorithm works by using heuristics to order SAX sequences, we call it HOT SAX, short for **H**euristically **O**rder **T**ime series using **SAX**



We have changed the original screen shot only by adding a red circle to highlight the anomaly

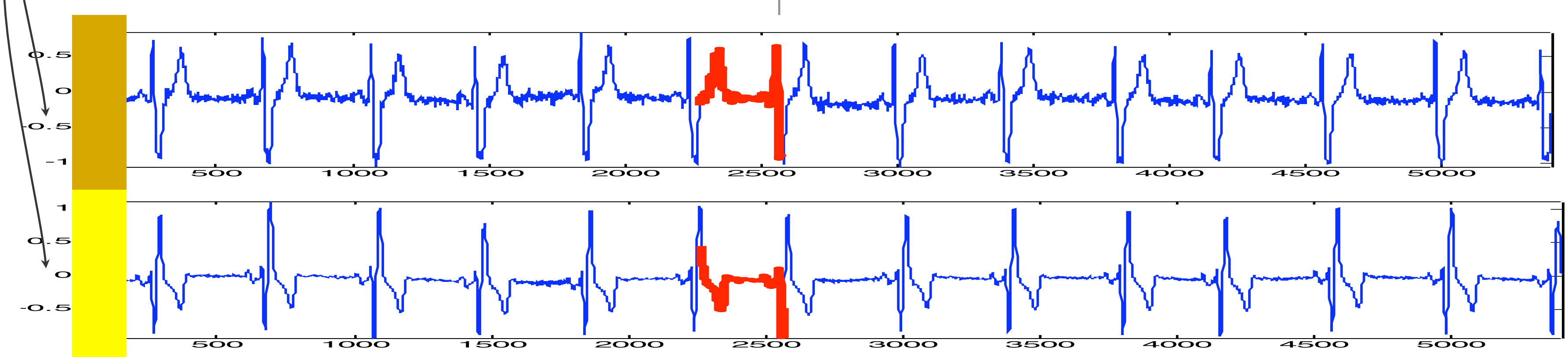
In all the examples below, we have included screen dumps of the MIT ECG server in order to allow people to retrieve the original data independent of us.

However, all data is also available from us in a convenient zip file.

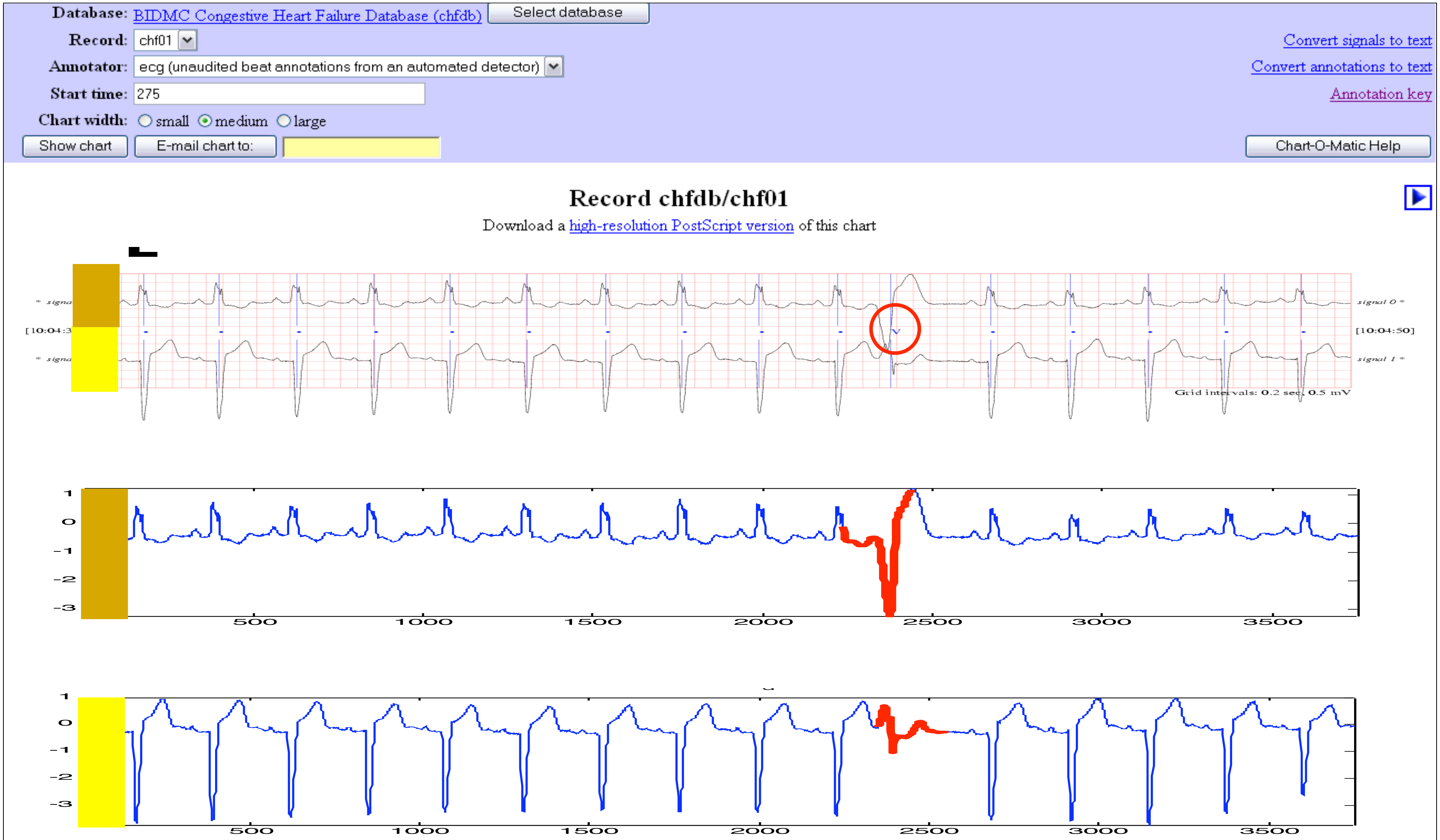
This is KEY only, the next 8 slides show examples in this format

The annotated ECG from PhysioBank (two signals)

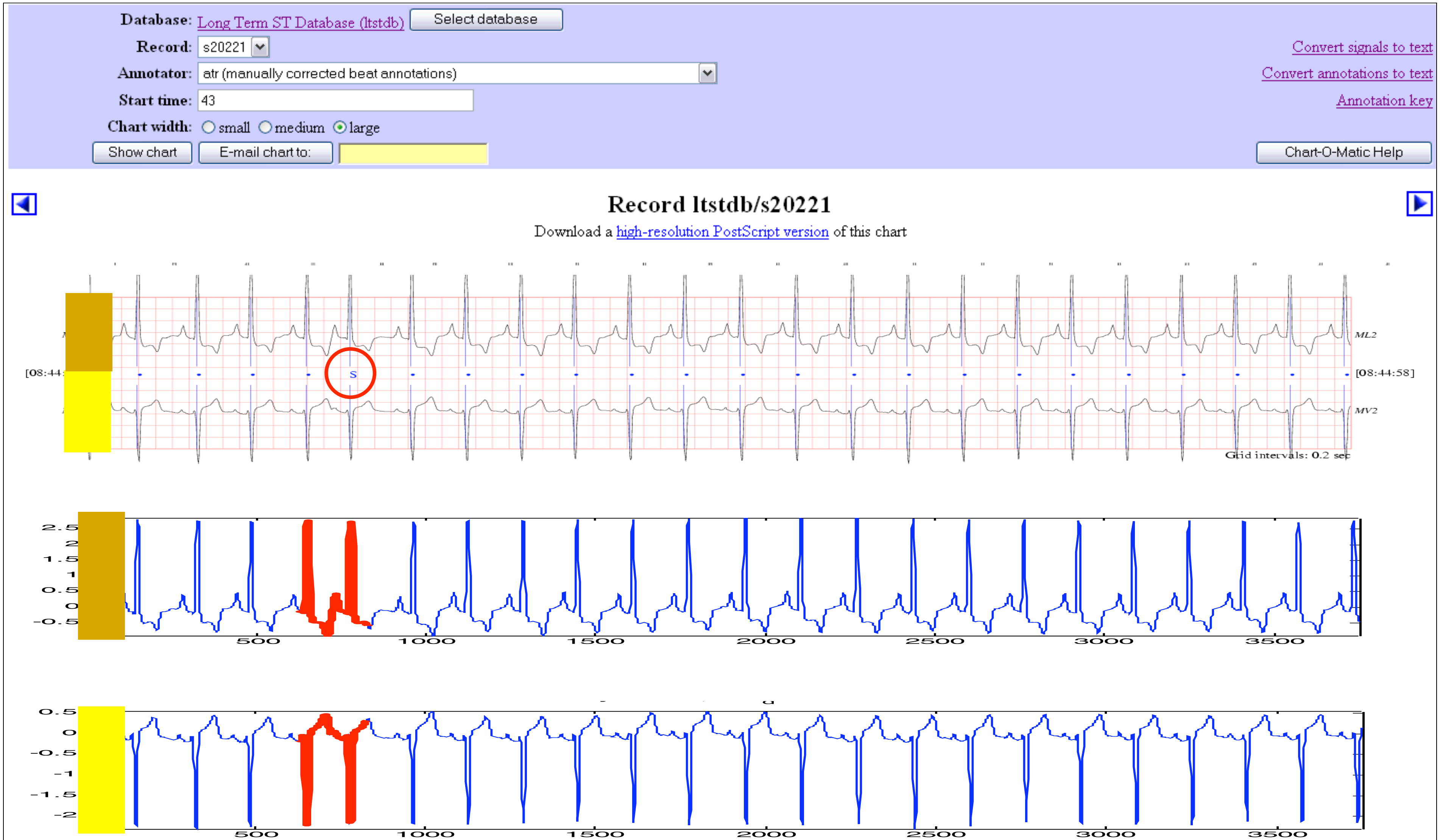
Anomalies (marked by red lines) found by the discord discovery algorithm. Each of the two traces were searched independently.



Each of the two traces were searched independently.



Each of the two traces were searched independently.

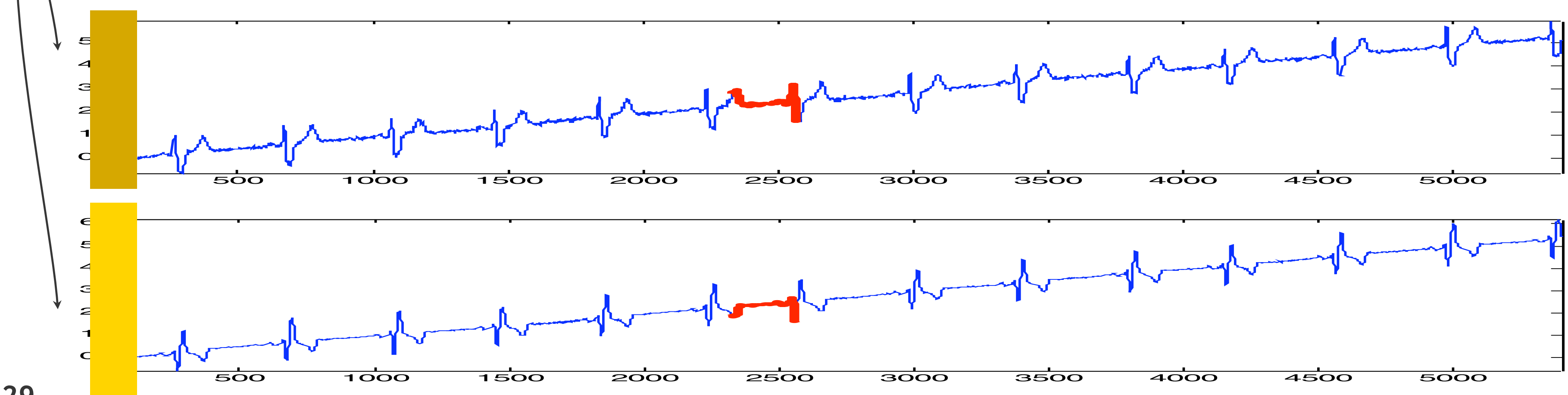


Each of the two traces were searched independently.

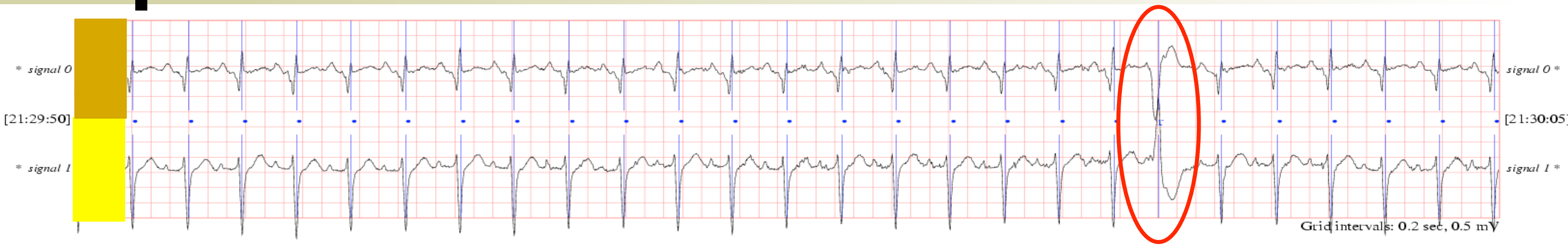
Adding Linear Trend

This is a dataset shown in a previous example

To demonstrate that the discord algorithm can find anomalies even with the presence of linear trends, we added linear trend to the ECG data on the top. The new data and the anomalies found are shown below. This is important in ECGs because of the *wandering baseline* effect.

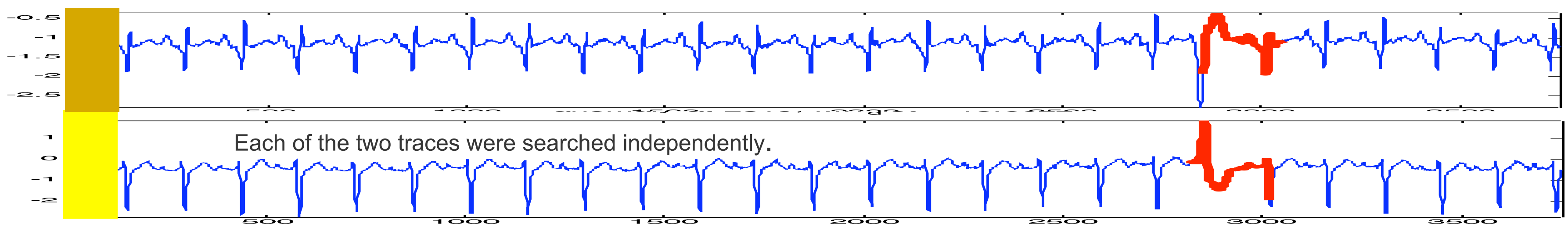


Window Size

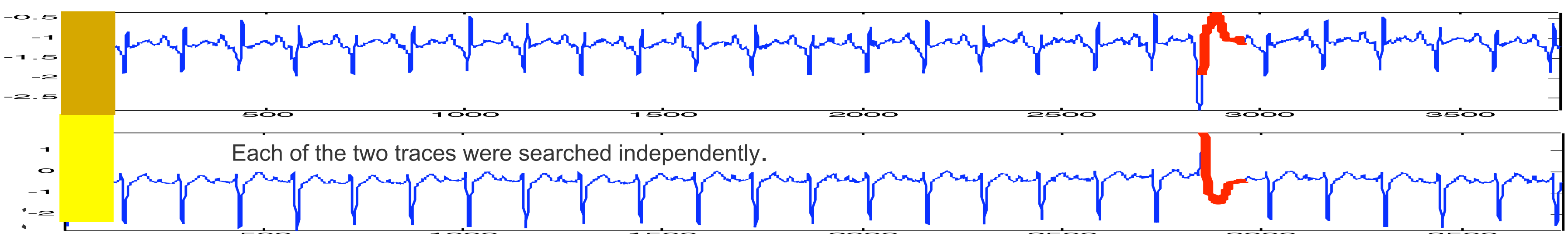


This example shows that the discord algorithm is not sensitive to the window size. In fact on all problems above, we can double or half the discord length and still find the anomalies. Below is just one example for clarity.

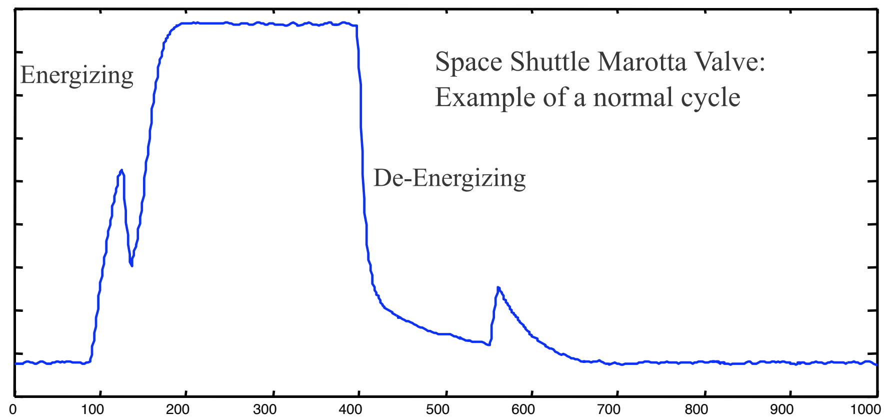
discord_{200}

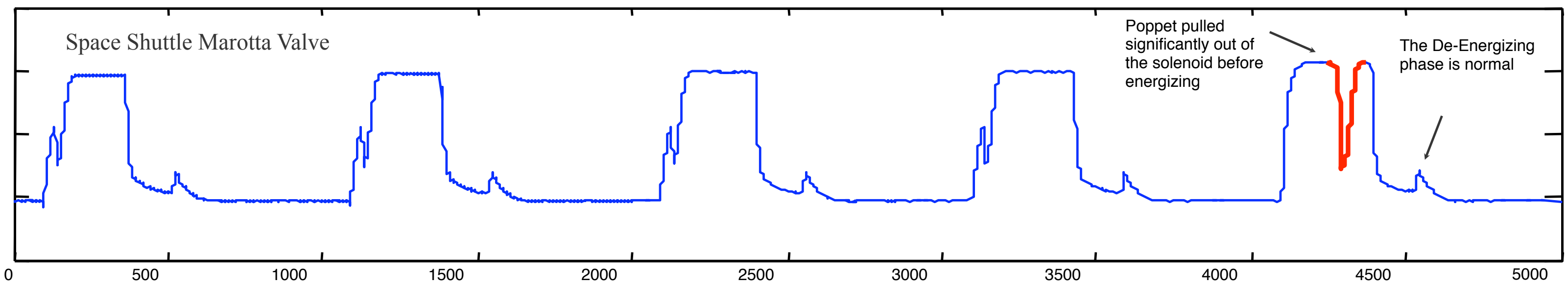


discord_{100}

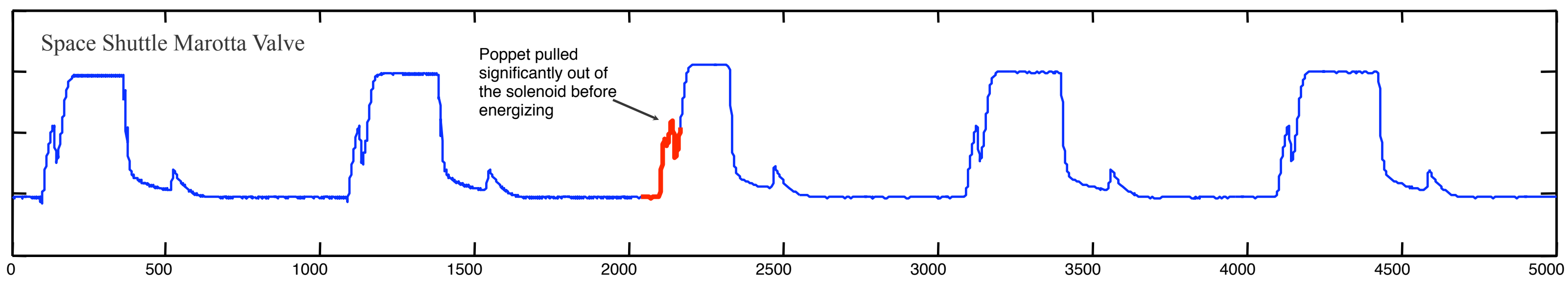


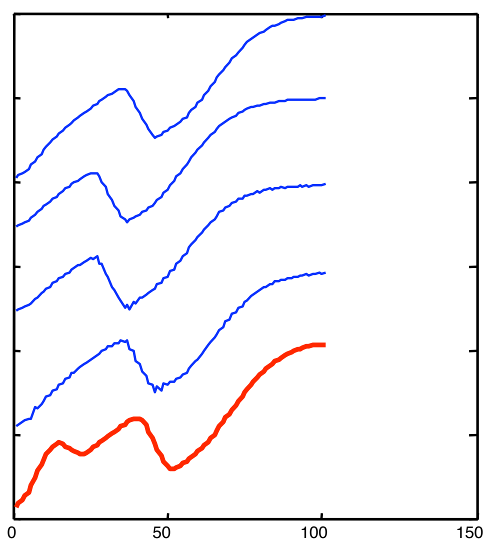
Space Shuttle Dataset

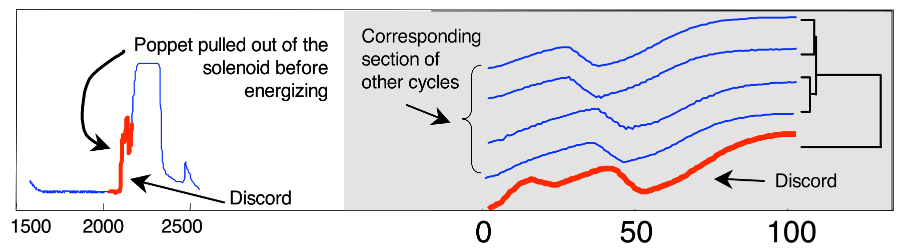


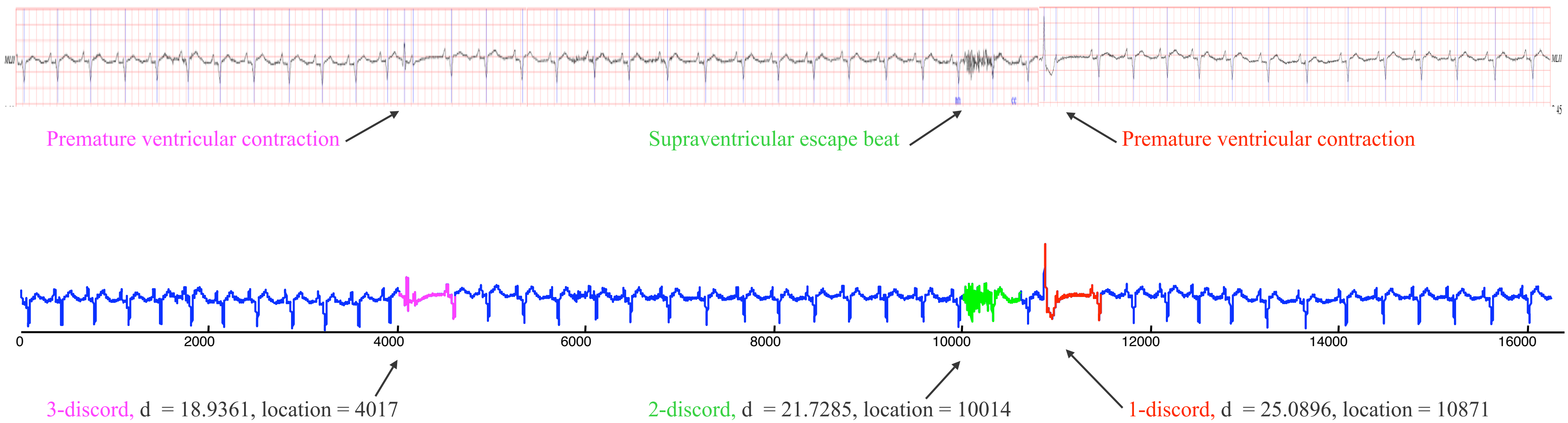


Space Shuttle - A More Subtle Problem



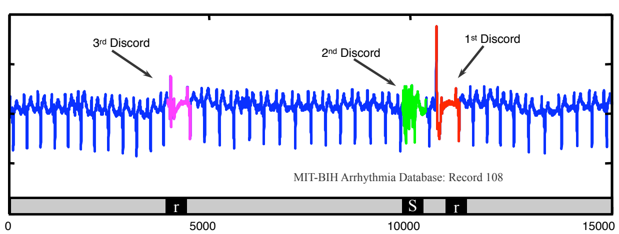


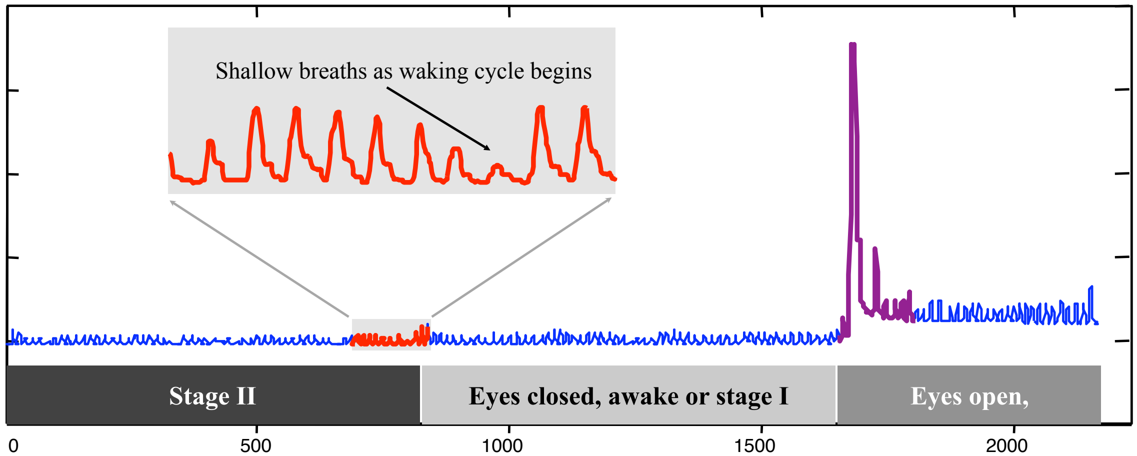




The time series is record mitdb/x_mitdb/x_108 from the PhysioNet Web Server (The local copy in the UCR archive is called mitdbx_mitdbx_108.txt). It is a two feature time series, here we are looking at just the MLII column.

Cardiologists from MIT have annotated the time series, here we have added colored makers to draw attention to those annotations. Here we show the results of finding the top 3 discords on this dataset. We chose a length of 600, because this a little longer than the average length of a single heartbeat.





A time series showing a patients respiration (measured by thorax extension), as they wake up. A medical expert, Dr. J. Rittweger, manually segmented the data. The 1-discord is a very obvious deep breath taken as the patient opened their eyes. The 2-discord is much more subtle and impossible to see at this scale. A zoom-in suggests that Dr. J. Rittweger noticed a few shallow breaths that indicated the transition of sleeping stages.

Institute for Physiology. Free University of Berlin.

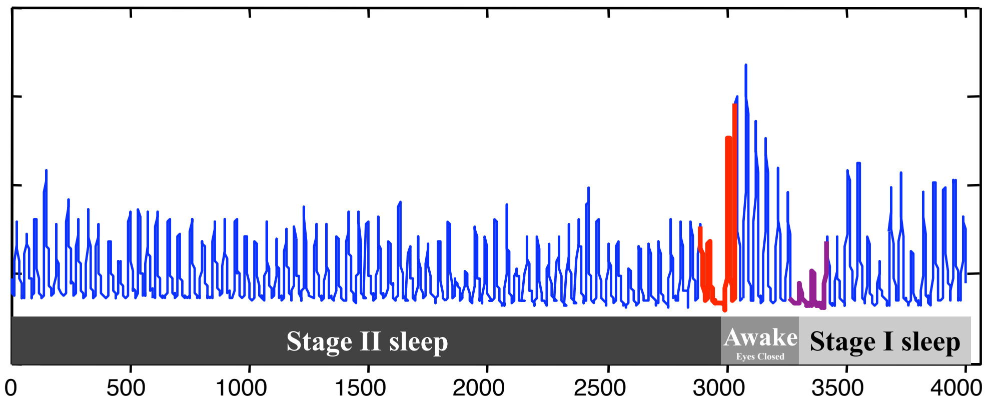
Data shows respiration (thorax extension), sampling rate 10 Hz.

This is Figure 9 in the paper.

This is dataset nprs44

Beginning at 15500
Ending at 22000

The beginning and ending points were chosen for visual clarity (given the small plot size) they do not effect the results



A time series showing a patients respiration (measured by thorax extension), as they wake up. A medical expert, Dr. J. Rittweger, manually segmented the data.

Institute for Physiology.Free University of Berlin.

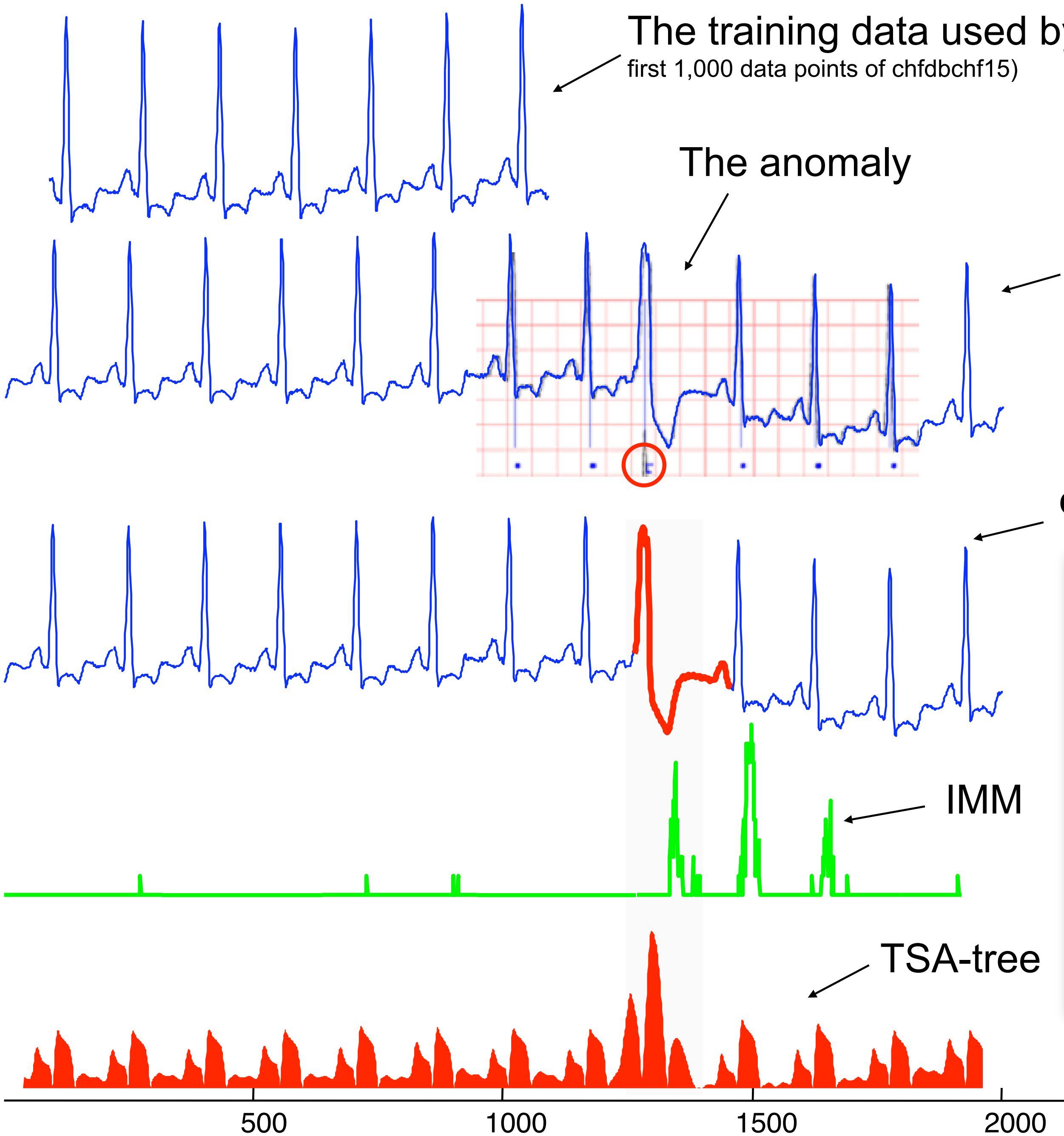
Data shows respiration (thorax extension), sampling rate 10 Hz.

This is Figure 10 in the paper.

This is dataset nprs43

Beginning at 1
Ending at 4000

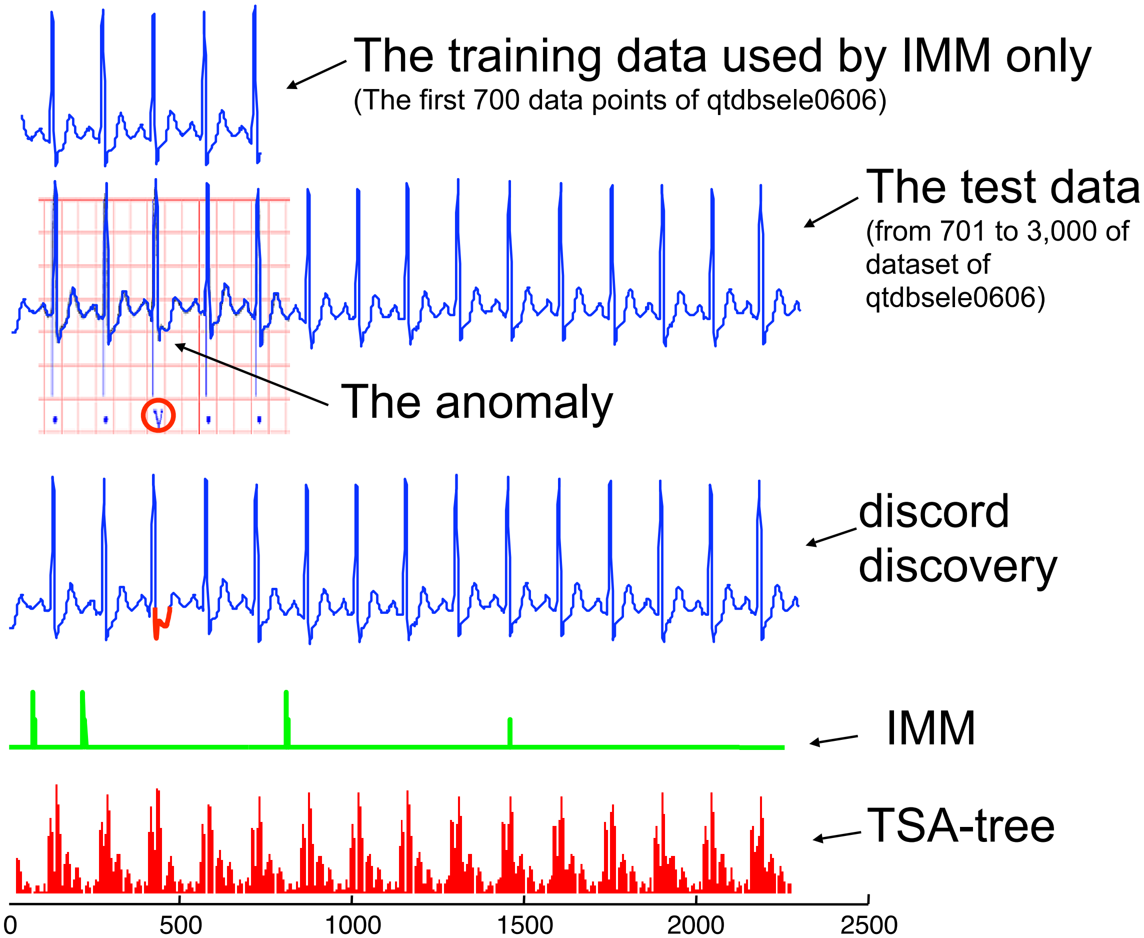
The beginning and ending points were chosen for visual clarity (given the small plot size) they do not effect the results



In this experiment, we can say that all the algorithms find the anomaly. The IMM approach has a slightly higher peak value just after the anomaly, but that may simply reflect the slight discretization of the time axis.

In the next slide, we consider more of the time series…

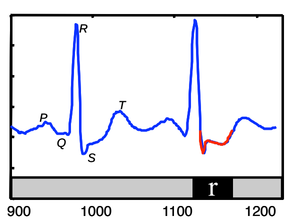
We can see here that the IMM approach has many false positives, in spite of very careful parameter tuning. It simply cannot handle complex datasets. Both the other algorithms do well here.
Note that this problem is in Figure 11 in paper

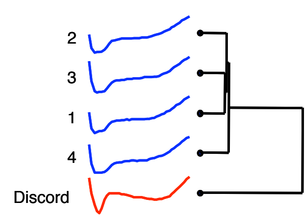


This example is Figure 12/13 in the paper.

Recall that we discussed this example above, it is interesting because the anomaly is extremely subtle.

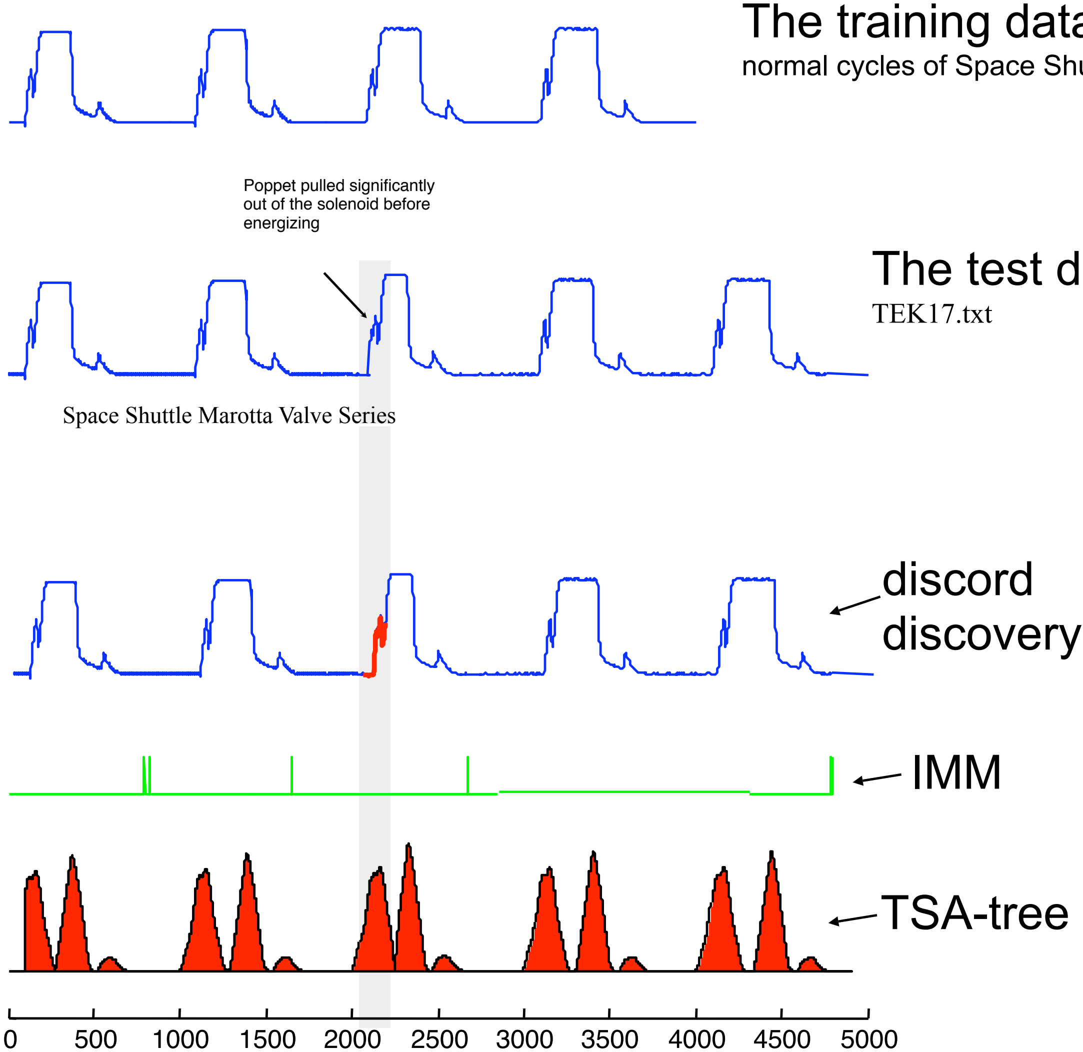
Here only the discord discovery algorithm can find the anomaly.





How was the discord able to find this very subtle Premature ventricular contraction? Note that in the normal heartbeats, the ST wave increases monotonically, it is only in the Premature ventricular contractions that there is an inflection.NB, this is not necessary true for all ECGS

The training data used by IMM only 4 normal cycles of Space Shuttle Marotta Valve Series

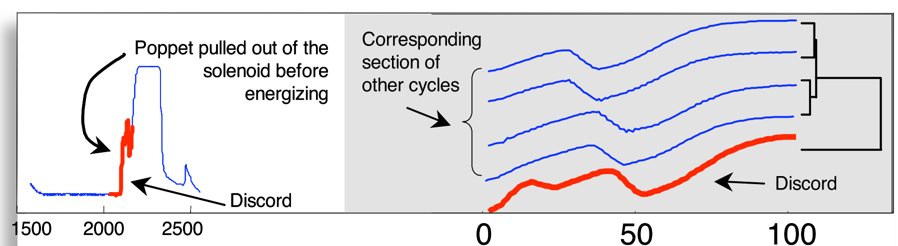


This example is Figure 7/8 in the paper.

Here the anomaly very subtle.

Only the discord discovery algorithm can find the anomaly.

A reminder of the cause of the anomaly



Conclusion & Future Work

- We define time series discords
- We introduce the HOT SAX algorithm to efficiently find discords and demonstrate its utility in various domains
- Future direction includes
  - multi-dimensional data
  - streaming data