

# Introduction to Time Series Mining

Slides edited from Keogh  
Eamonn's tutorial:



**VLDB2006**  
32<sup>nd</sup> International Conference on Very Large Data Bases  
The Convention and Exhibition Center(COEX), Seoul, Korea

Your CD-rom contains:  
• VLDB 2006 time series tutorial.  
• More than 100 time series datasets.  
• Materials for teaching data mining.

... excellent tutorial concerning temporal mining ... Dr. Margaret Dunham, in her book, Data Mining Introductory and Advanced Topics.

DFT DWT SVD APCA PAA

NLP CLIPPED SAX CHEB PLA

Whyuse Dendograms? Definition: Let  $O$  and  $O'$  be two objects from the universe of possible objects. The distance (dissimilarity) is denoted by  $d(O, O')$ . Symmetry:  $d(A, B) = d(B, A)$ . Completing:  $d(A, B) \leq d(A, C) + d(C, B)$ . Consistency:  $d(A, B) \leq d(A, C) \leq d(A, B)$ .

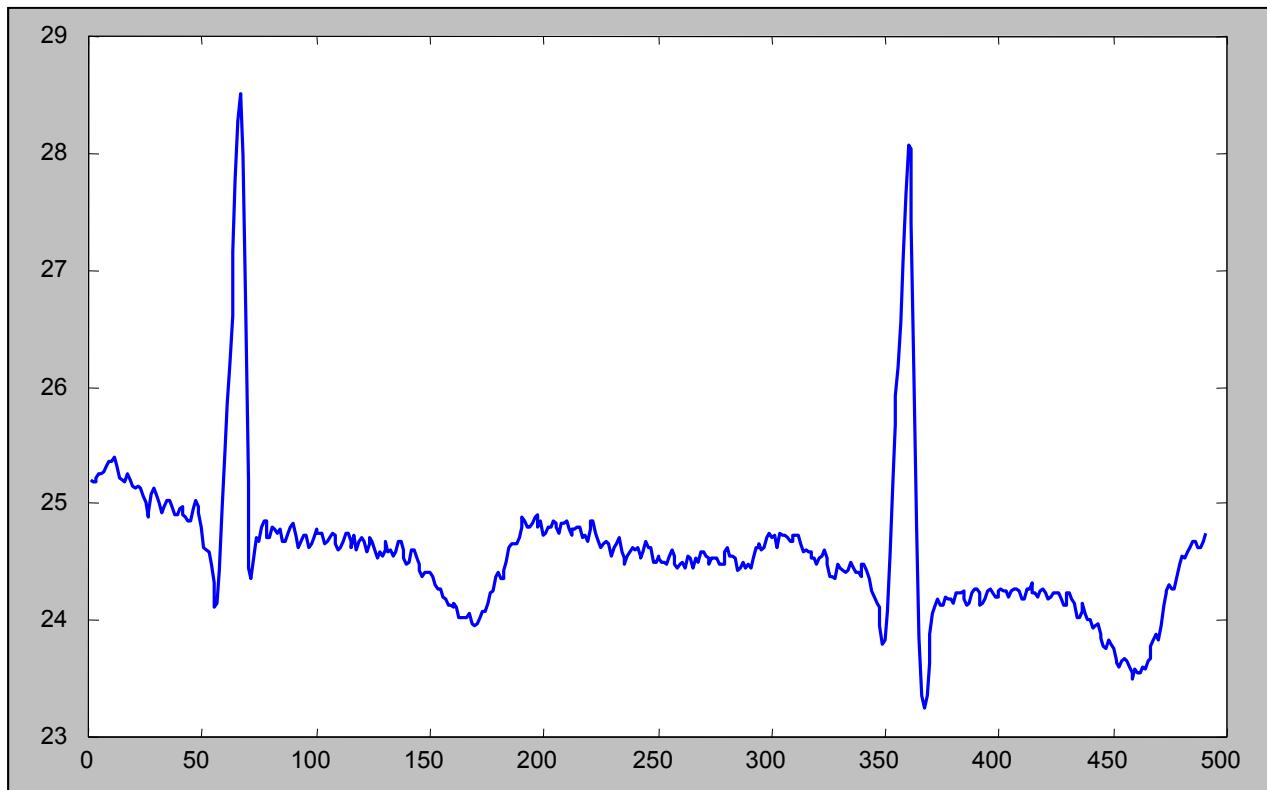
Properties of Distance Measures:  $d(A, B) \geq 0$ ,  $d(A, B) = 0 \Leftrightarrow A = B$ ,  $d(A, B) \leq d(A, C) + d(C, B)$ .

"Awesome tutorial!! It's just wonderful... playful AND deep! I couldn't stop looking at it, even though I've got other things to do... it was a well spent hour!" Dr Ben Shneiderman, Director of the Human-Computer Interaction Laboratory, University of Maryland at College Park.

25.1750  
25.2250  
25.2500  
25.2500  
25.2750  
25.3250  
25.3500  
25.3500  
25.4000  
25.4000  
25.3250  
25.2250  
25.2000  
25.1750  
..  
..  
24.6250  
24.6750  
24.6750  
24.6250  
24.6250  
24.6250  
24.6750  
24.7500

# What are Time Series?

A time series is a collection of observations made sequentially in time.



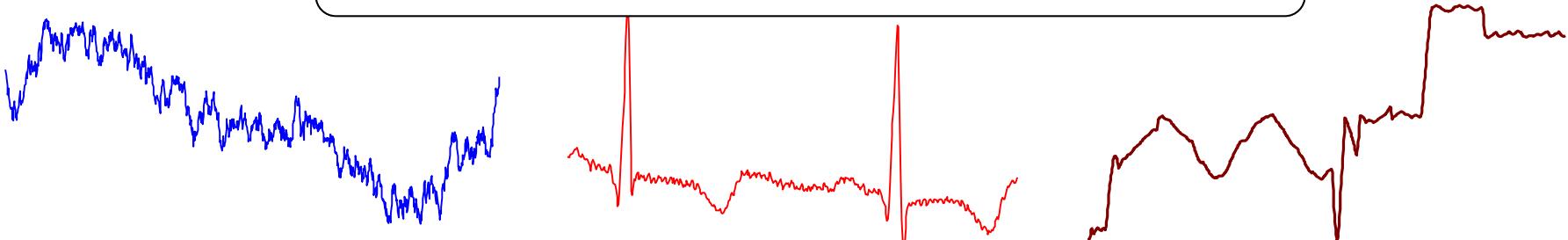
# Time Series are Ubiquitous! I

People measure things...



- *Their blood pressure*
- *George Bush's popularity rating*
- *The annual rainfall in Seattle*
- *The value of their Google stock*

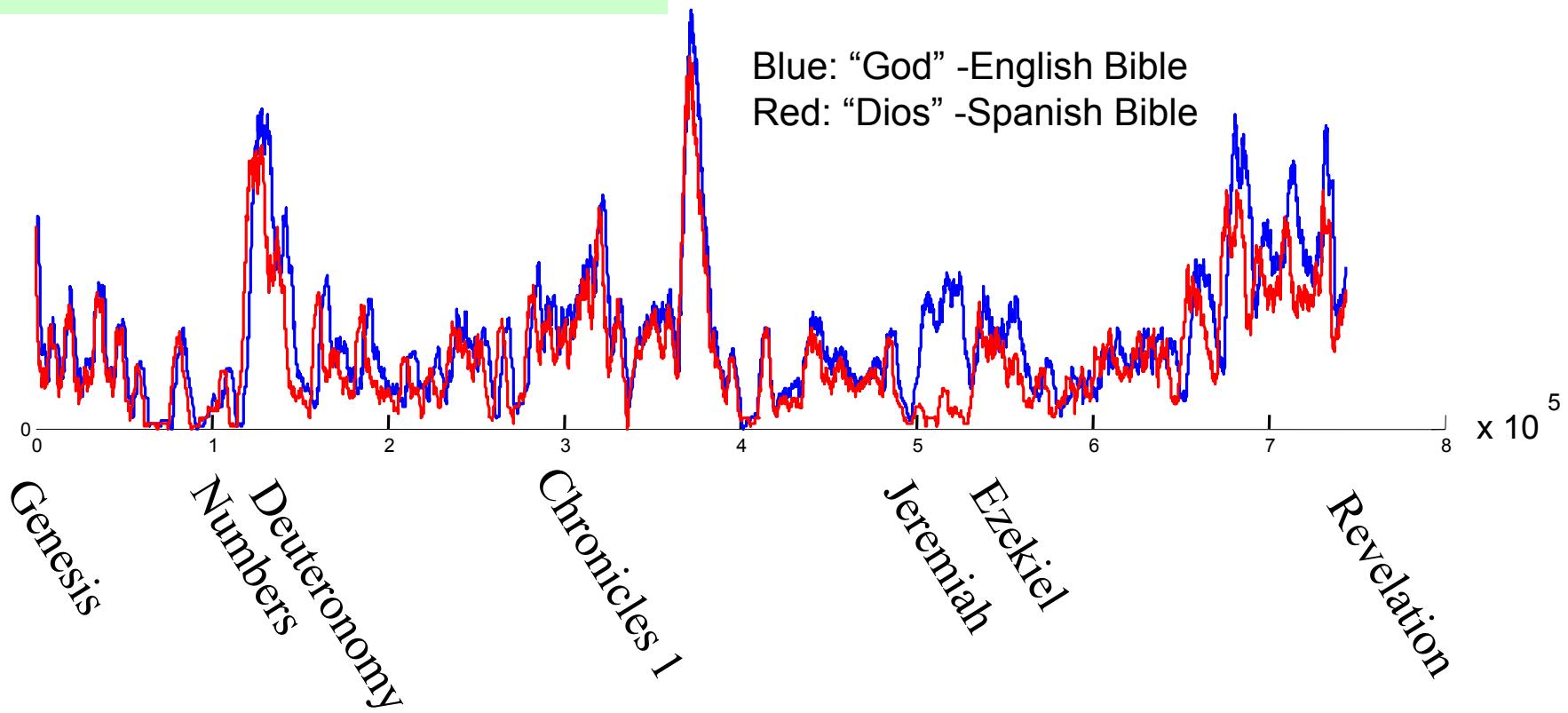
...and things change over time...



Thus time series occur in virtually every medical, scientific and businesses domain

# Text data, may best be thought of as time series...

The local frequency  
of words in the Bible



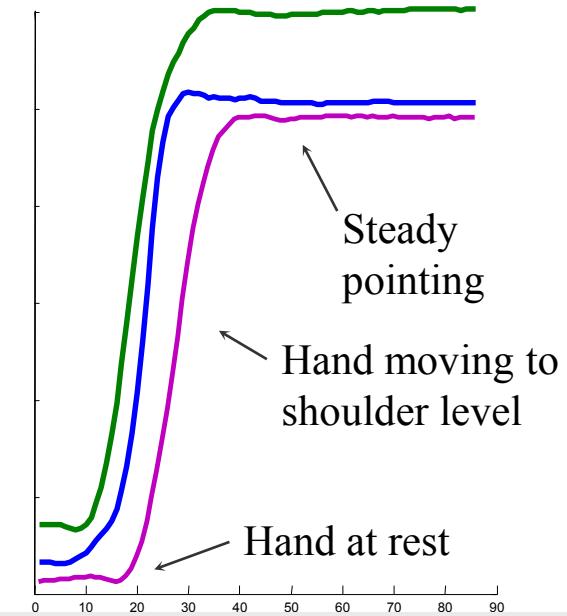
Gray: "El Señor" -Spanish Bible



# Video data, may best be thought of as time series...



**Point**



**Gun-Draw**

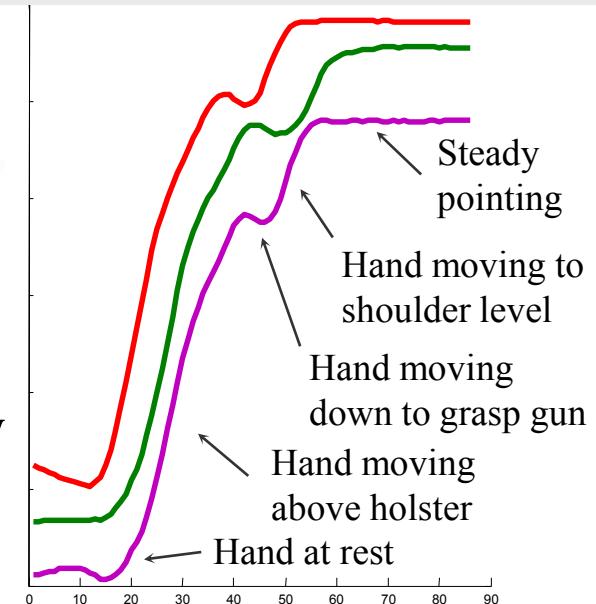
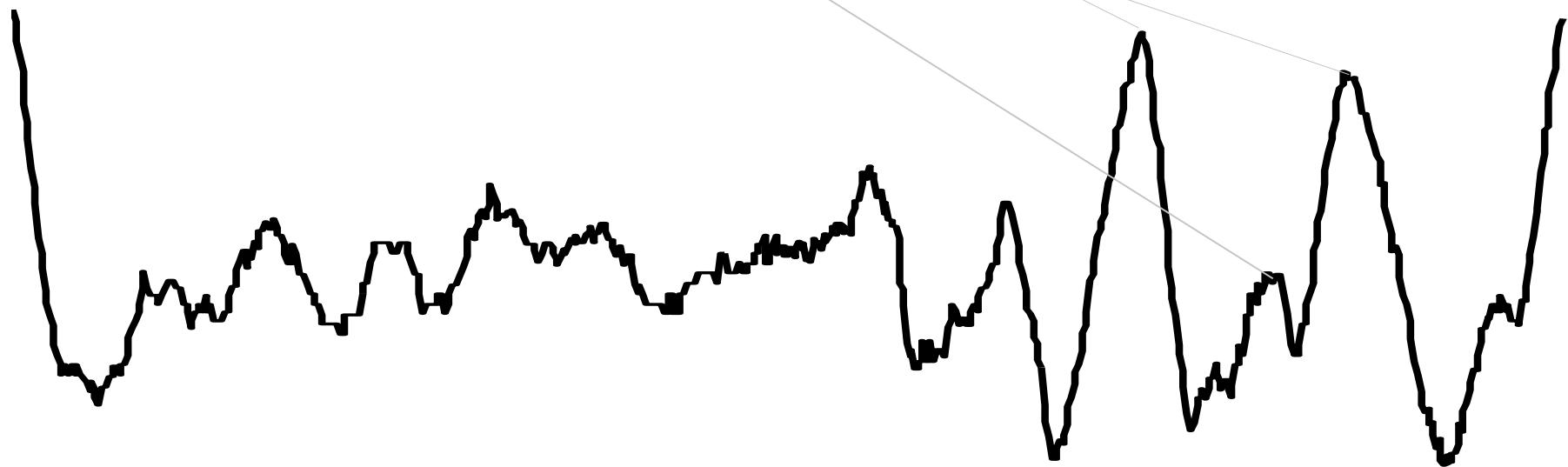
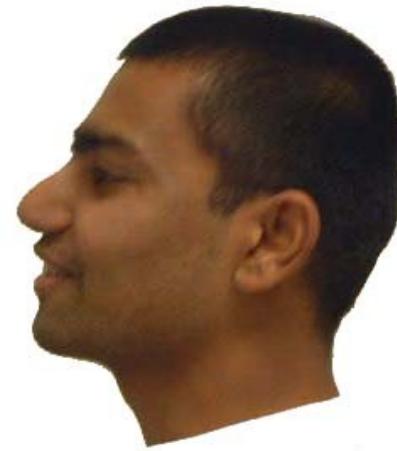
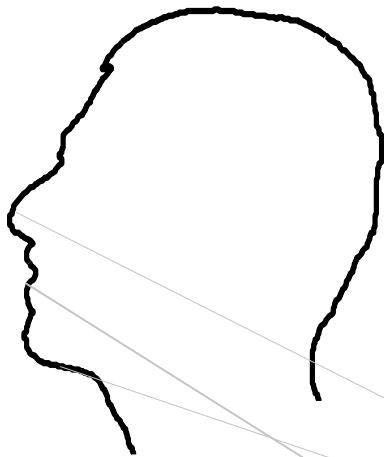
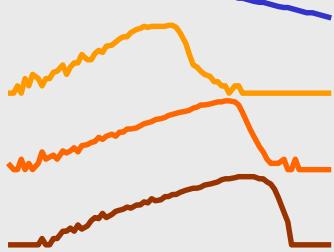


Image data, may best be thought of as time series...

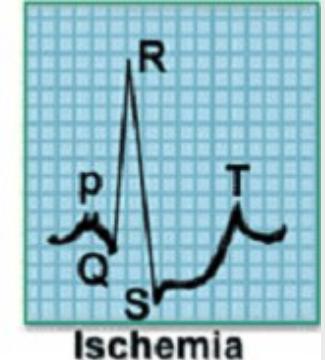
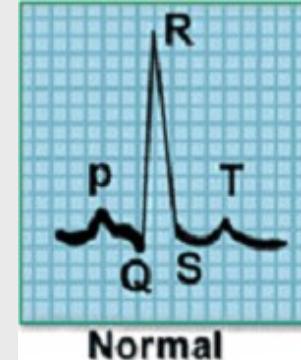


# What do we want to do with the time series data?

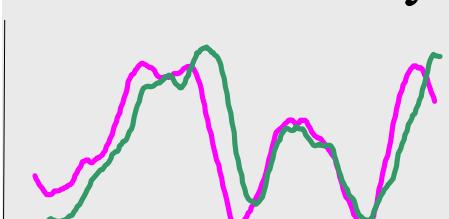
Clustering



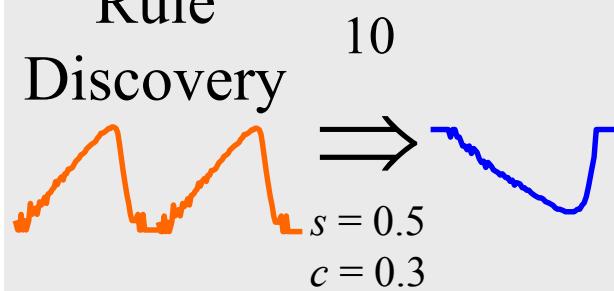
Classification



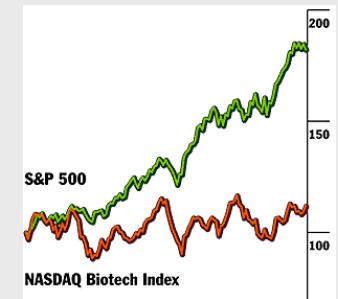
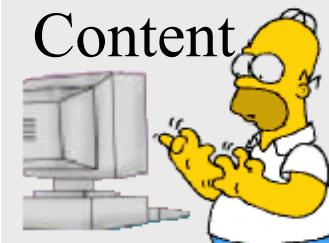
Motif Discovery



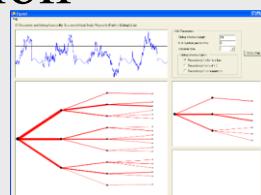
Rule Discovery



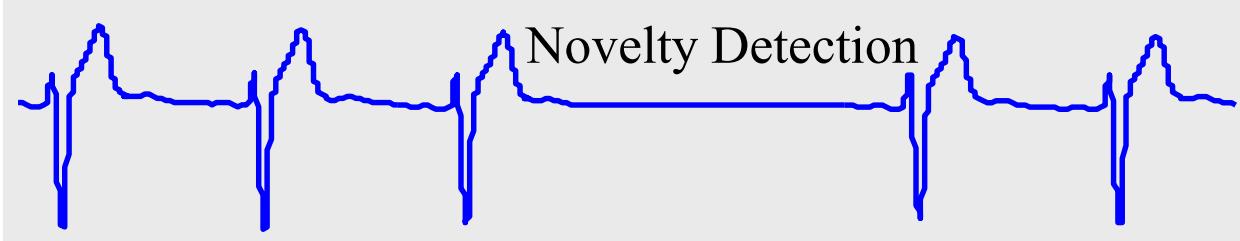
Query by Content



Visualization



Novelty Detection

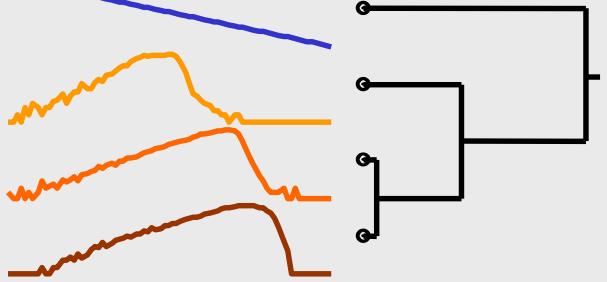


# Time series analysis tasks

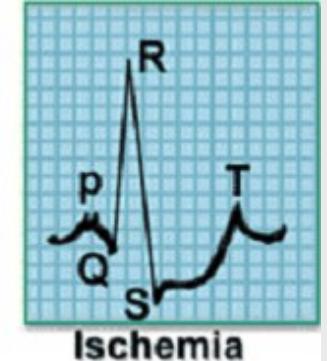
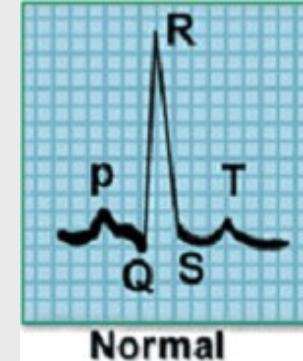
- Similarity-based tasks
  - Standard clustering, classification (KNN), etc.
- Outlier detection
- Frequent patterns (Motifs)
- Prediction

# All these problems require similarity matching

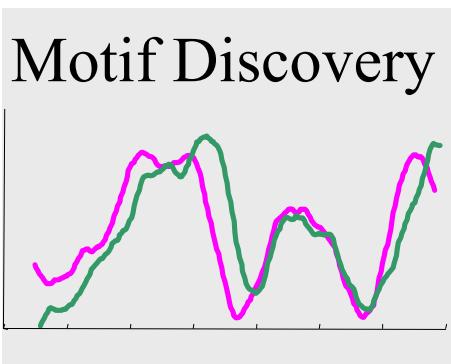
Clustering



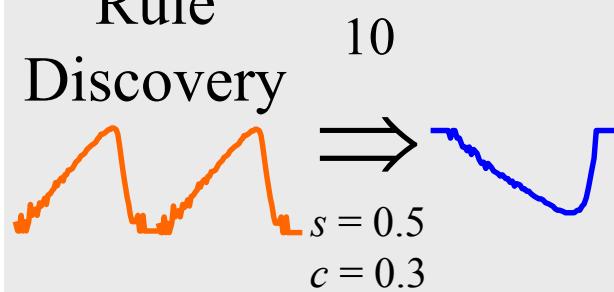
Classification



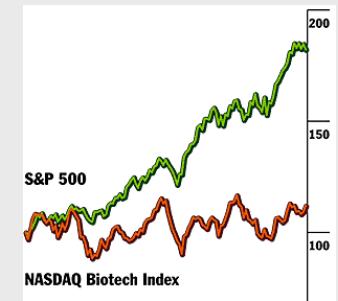
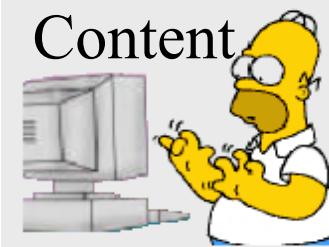
Motif Discovery



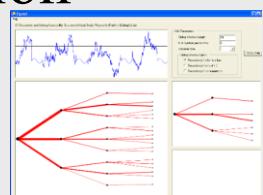
Rule Discovery



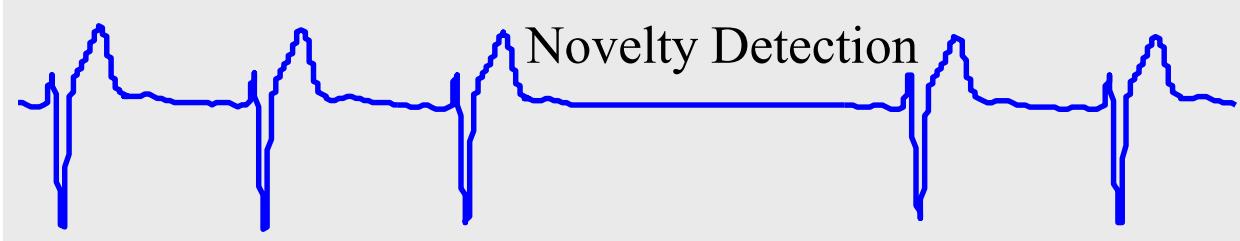
Query by Content



Visualization



Novelty Detection



# What is Similarity?

The quality or state of being similar; likeness; resemblance; as, a similarity of features. **Webster's Dictionary**



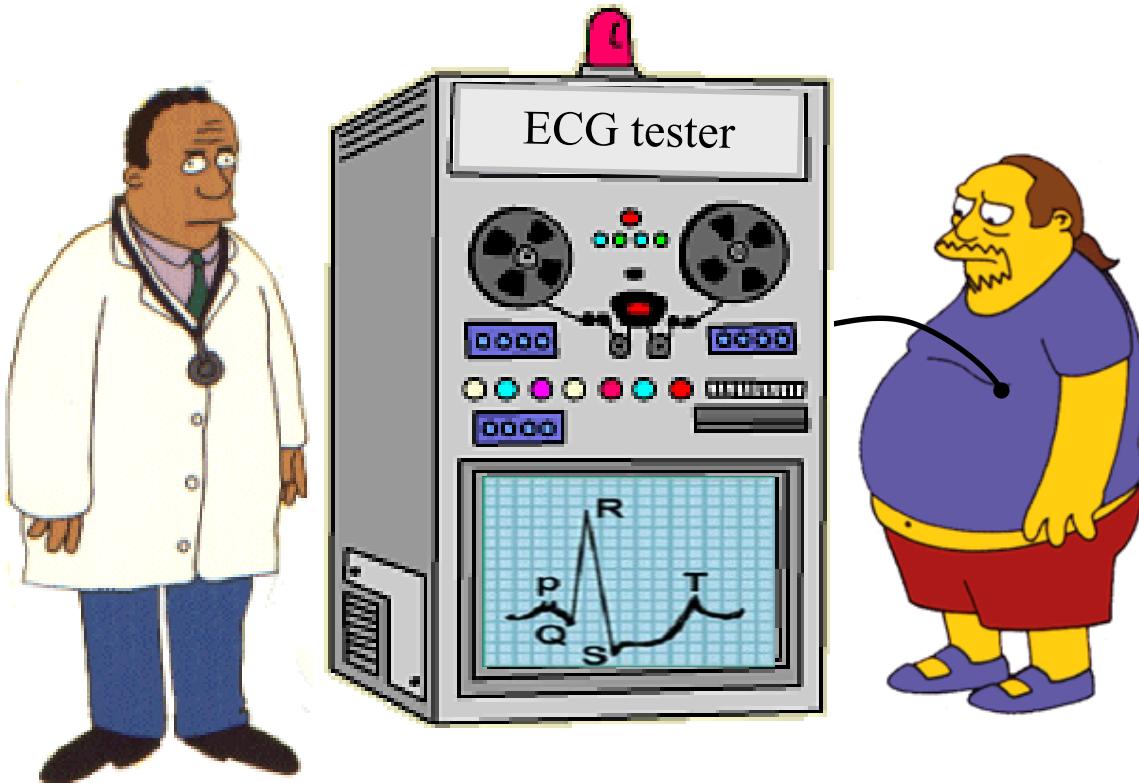
Similarity is hard to define, but...

*“We know it when we see it”*

The real meaning of similarity is a philosophical question.

We will take a more pragmatic approach.

# Here is a simple motivation for the first part of the tutorial



You go to the doctor because of chest pains. Your ECG looks strange...

Your doctor wants to search a database to find **similar** ECGs, in the hope that they will offer clues about your condition...

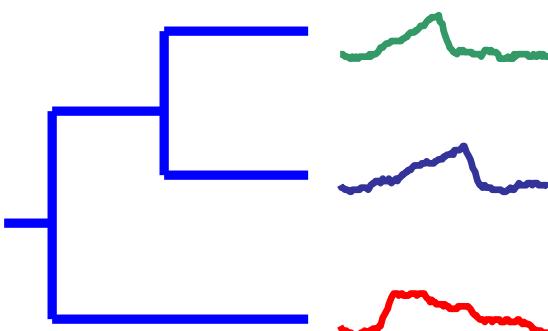
## Two questions:

- How do we define similar?
- How do we search quickly?

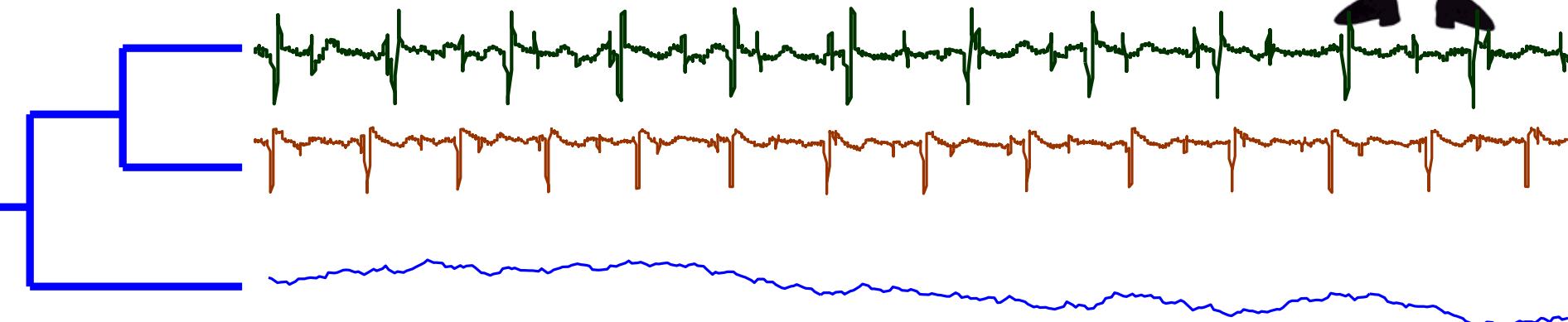
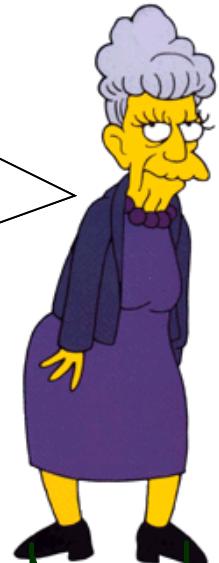
# Two Kinds of Similarity

time series

Similarity at  
the level of  
*shape*  
Next 40 minutes



Similarity at  
the *structural*  
level  
Another 10 minutes



# Euclidean Distance Metric

Given two time series:

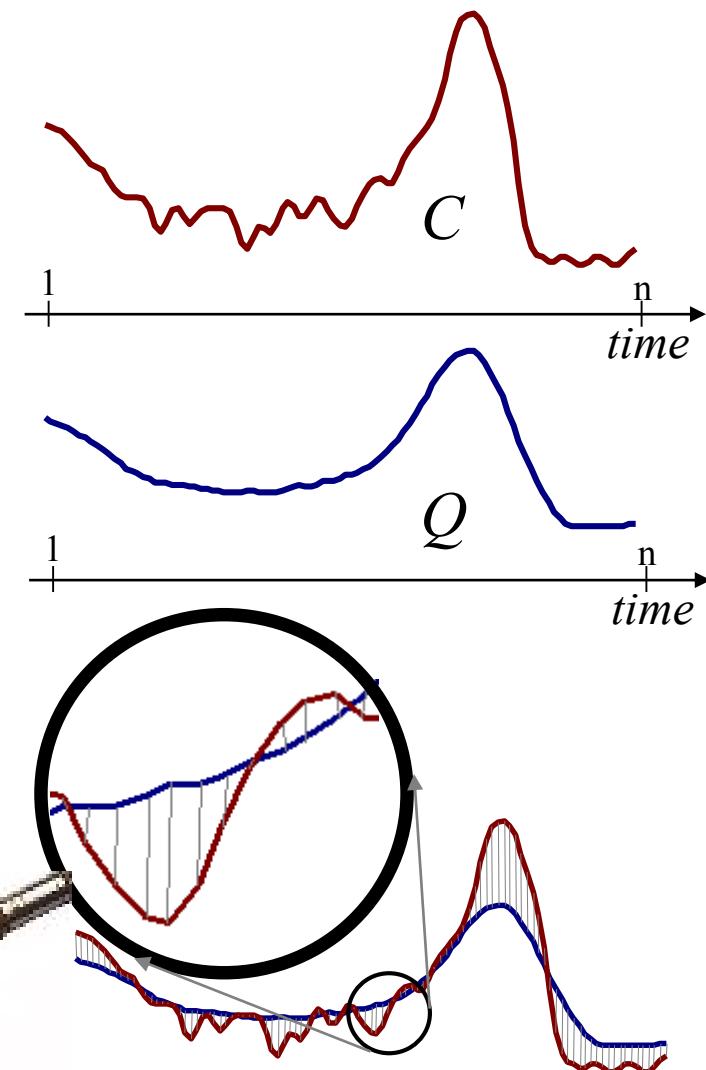
$$Q = q_1 \dots q_n \quad C = c_1 \dots c_n$$

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

Same meaning as in transaction data:

- schema: <age, height, income, tenure>
- T1 = < 56, 176, 110, 95 >
- T2 = < 36, 126, 180, 80 >

$$D(T1, T2) = \sqrt{[(56-36)^2 + (176-126)^2 + (110-180)^2 + (95-80)^2]}$$



About 80% of published work in data mining uses Euclidean distance

$$D(Q, C)$$

# Preprocessing the data before distance calculations



If we naively try to measure the distance between two "raw" time series, we may get very unintuitive results



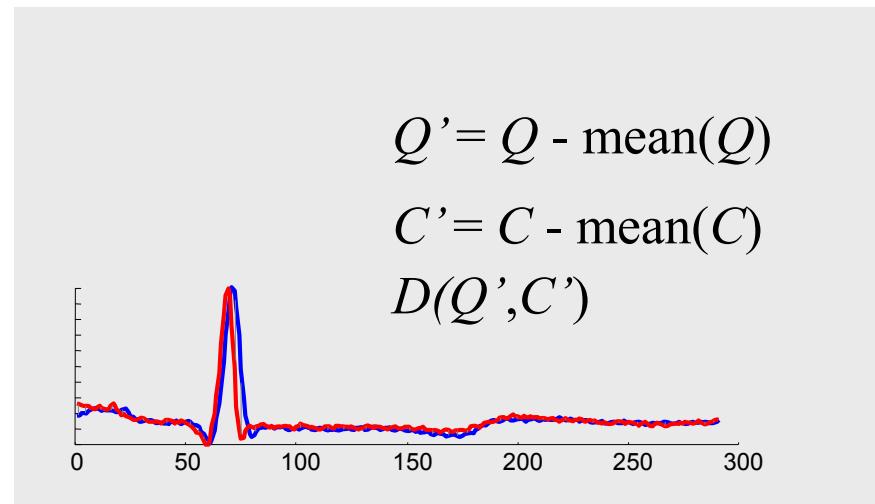
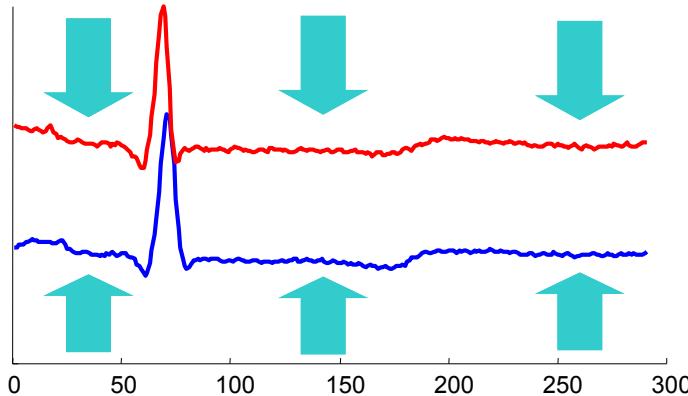
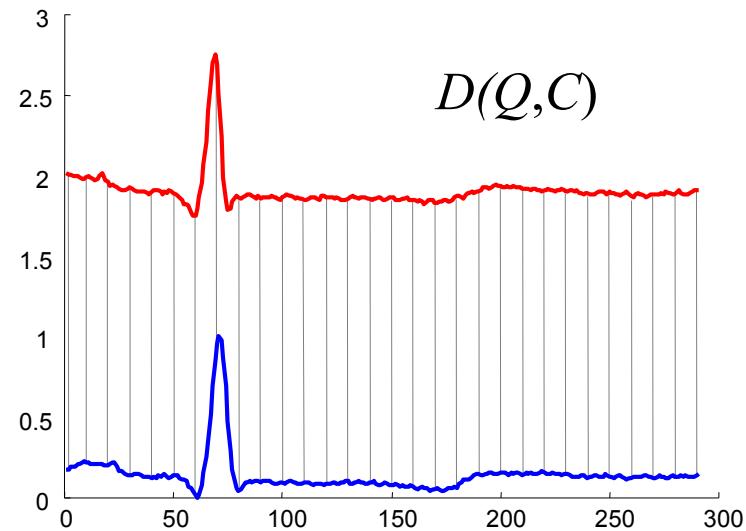
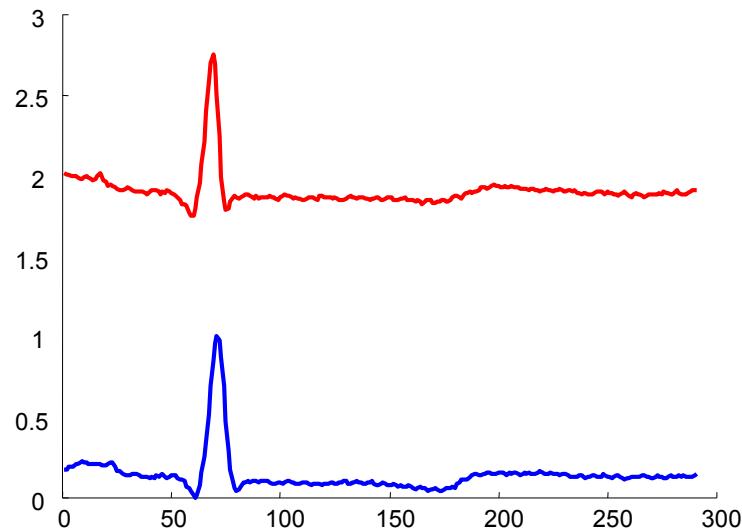
Euclidean distance is very sensitive to some "distortions" in the data.  
For most problems these distortions are not meaningful => should remove them



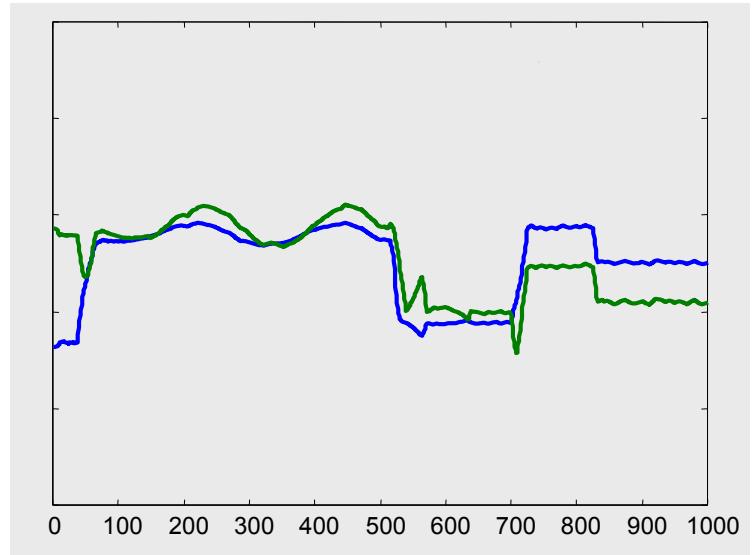
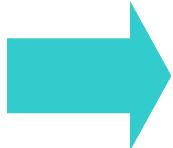
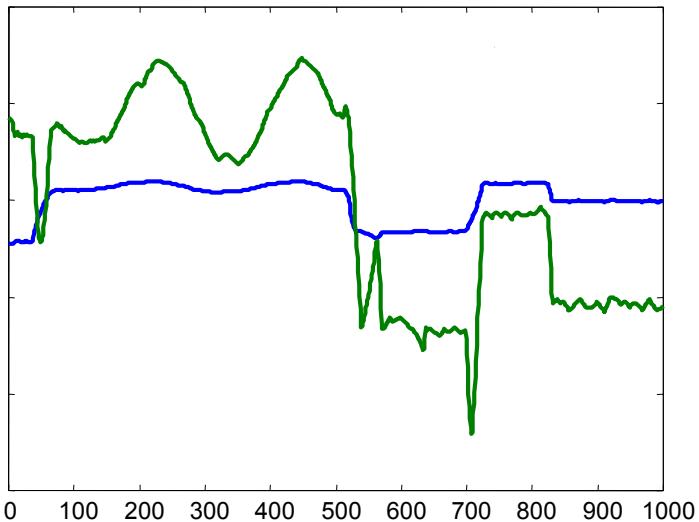
In the next few slides we will discuss the 4 most common distortions, and how to remove them

- Offset Translation
- Amplitude Scaling
- Linear Trend
- Noise

# Transformation I: Offset Translation



# Transformation II: Amplitude Scaling



Z-score of Q

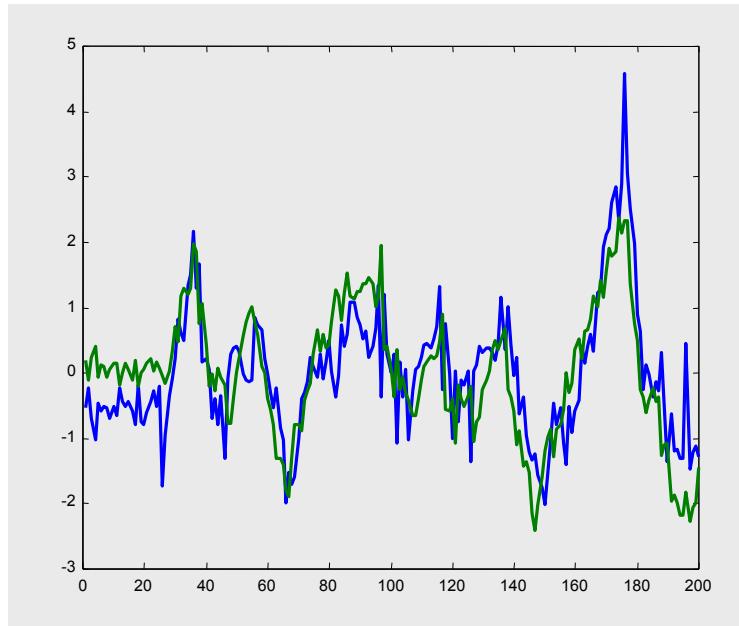
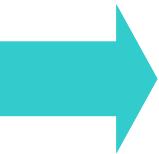
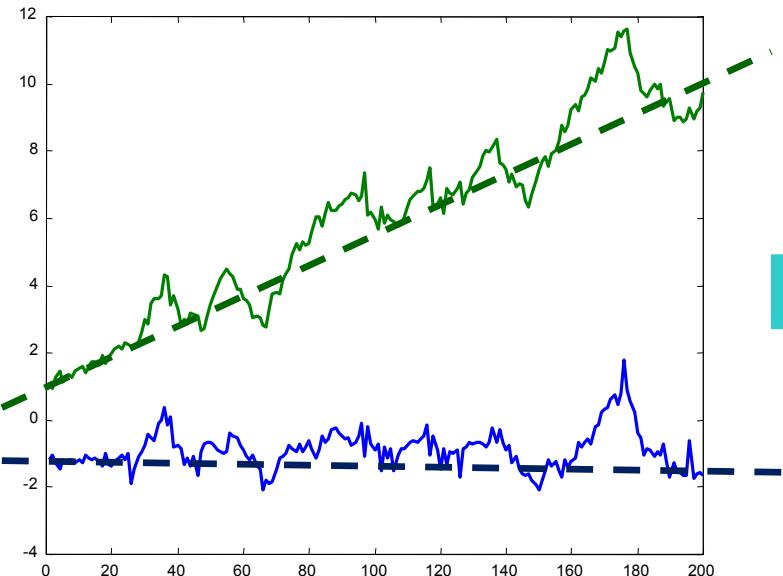


$$Q'' = (Q - \text{mean}(Q)) / \text{std}(Q)$$

$$C'' = (C - \text{mean}(C)) / \text{std}(C)$$

$$D(Q'', C'')$$

# Transformation III: Linear Trend



Removing linear trend:

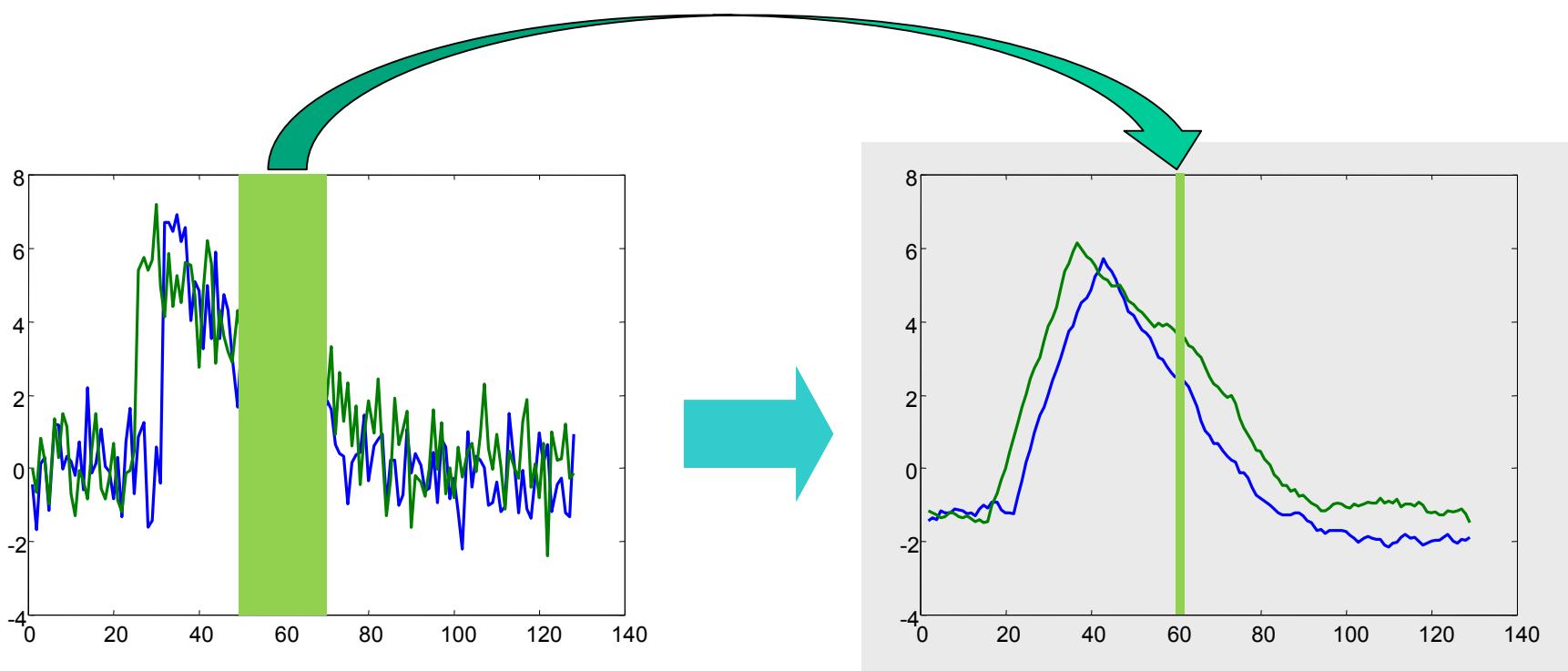
- fit the best fitting straight line to the time series, then
- subtract that line from the time series.

Removed **linear trend**

Removed offset translation

Removed amplitude scaling

# Transformation IV: Noise

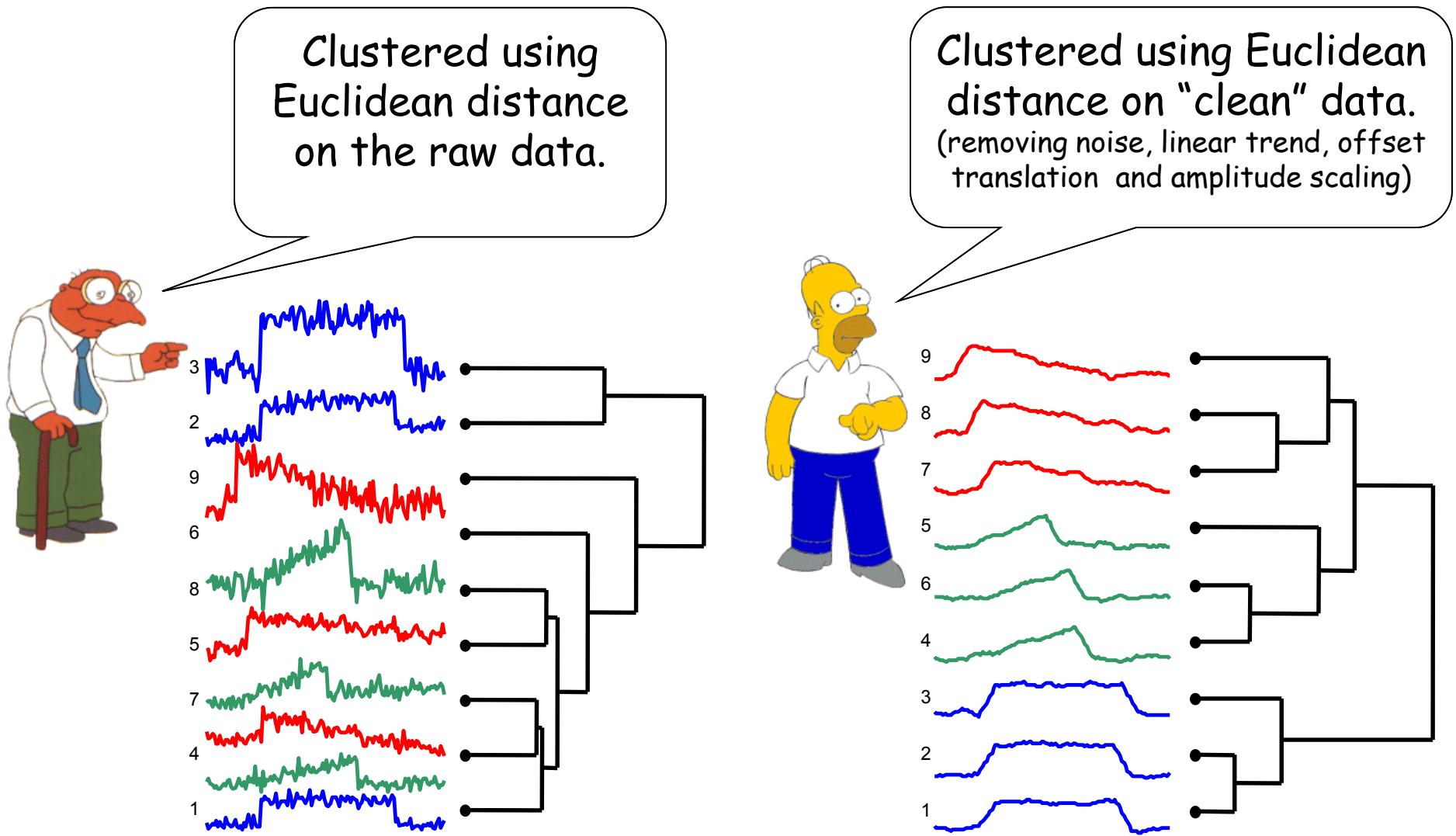


The intuition behind  
removing noise is...

Average each datapoints  
value with its neighbors.

$$\begin{aligned} Q' &= \text{smooth}(Q) \\ C' &= \text{smooth}(C) \\ D(Q', C') \end{aligned}$$

# A Quick Experiment to Demonstrate the Utility of Preprocessing the Data



# Summary of Preprocessing



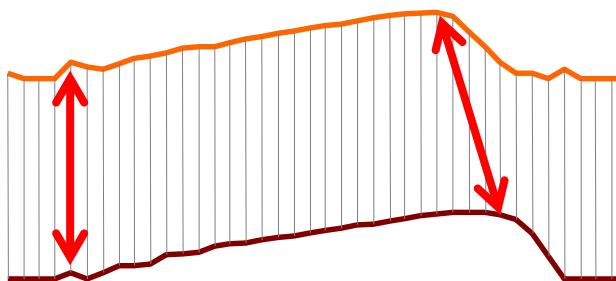
The “raw” time series may have distortions which we should remove before clustering, classification etc



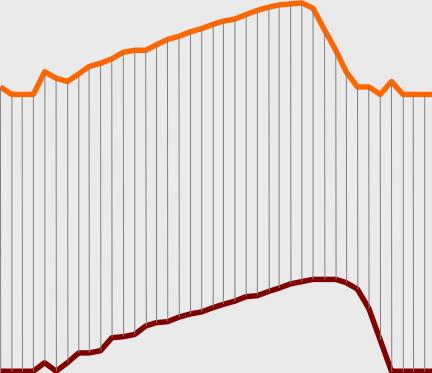
Of course, sometimes the distortions are the most interesting thing about the data, the above is only a general rule

# Beyond Euclidean: Dynamic Time Warping

- Sometimes two time series that are conceptually equivalent evolve at different speeds, at least in some moments



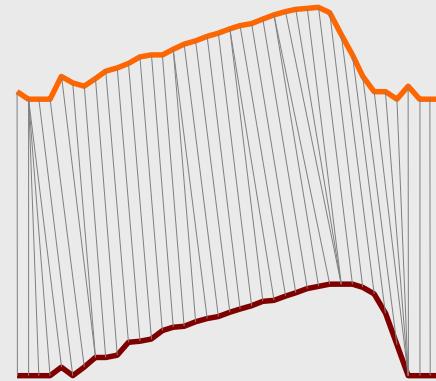
E.g. correspondence  
of peaks in two  
similar time series



**Fixed Time Axis**

*Sequences are aligned “one to one”.*

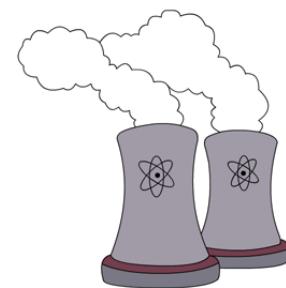
Greatly suffers from the misalignment in data.



**“Warped” Time Axis**

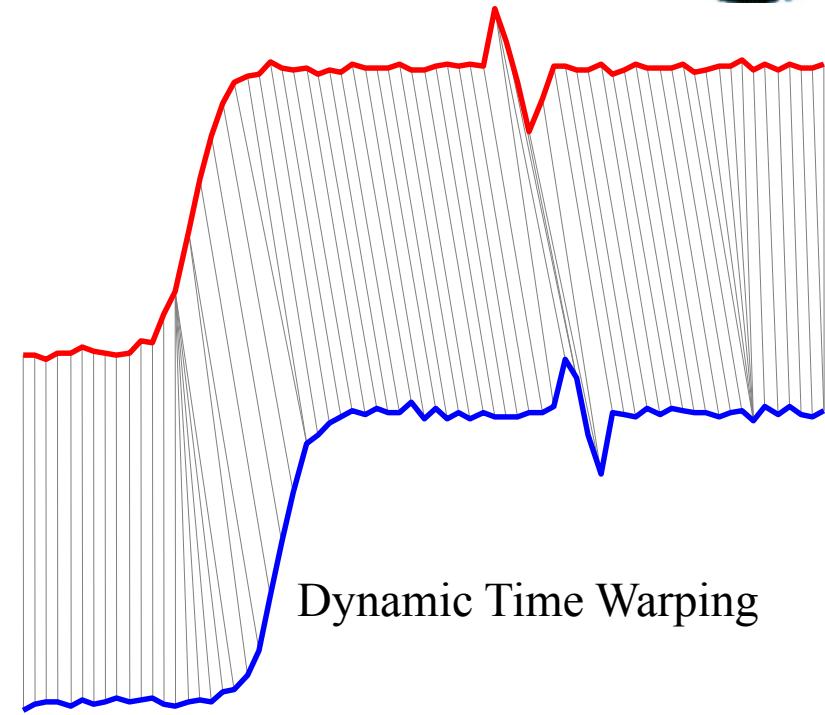
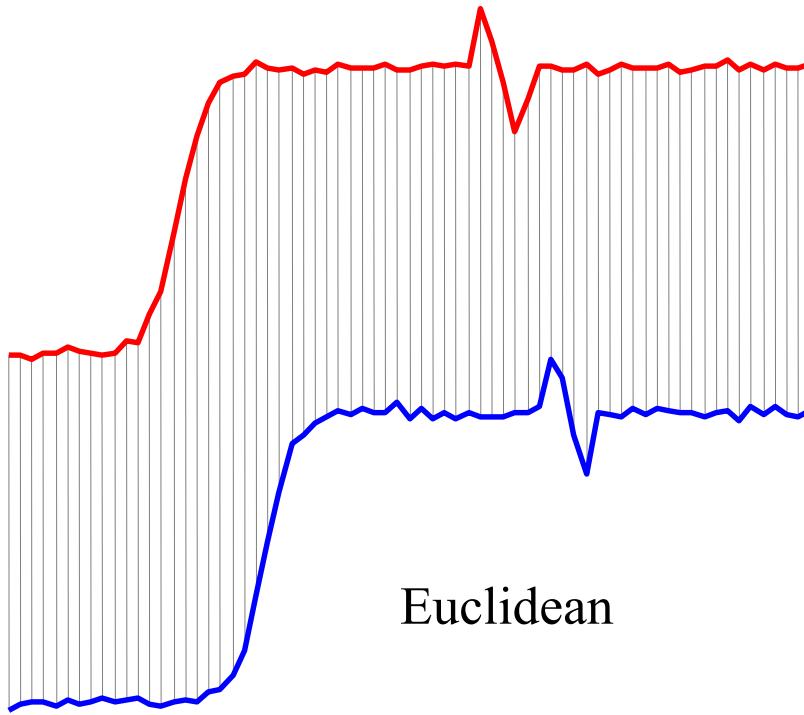
*Nonlinear alignments are possible.*

Can correct misalignments in data.

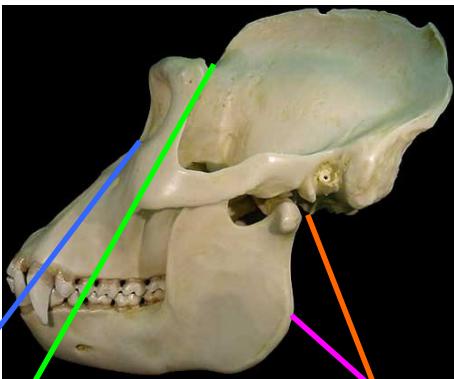


Here is another example on  
nuclear power plant trace data,  
to help you develop an intuition  
for DTW

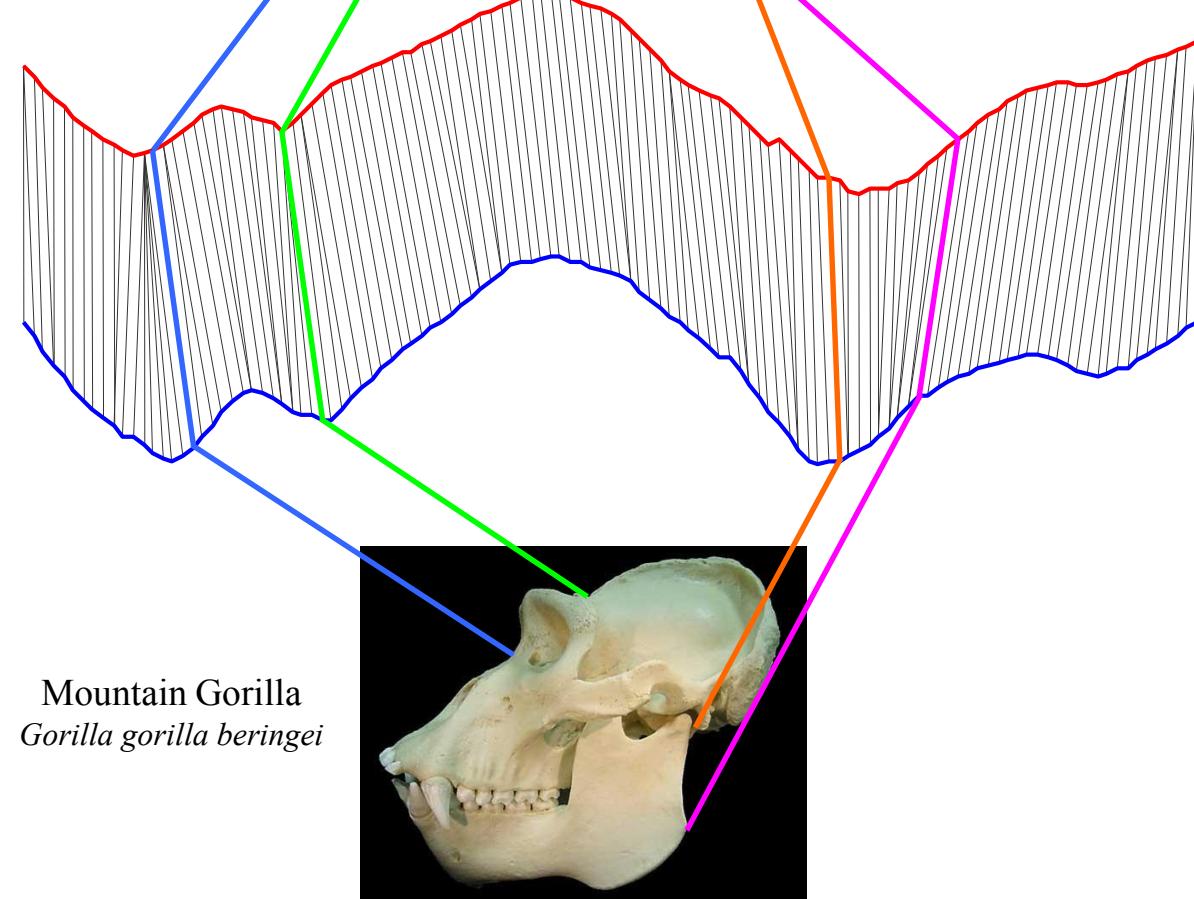
Nuclear  
Power  
Excellent!



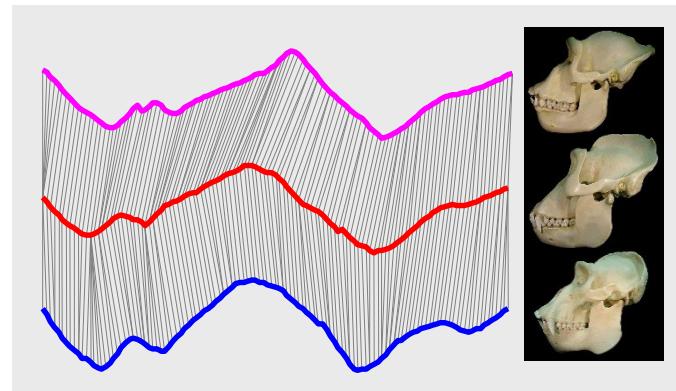
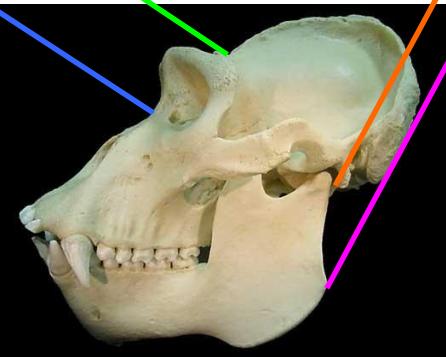
Lowland Gorilla  
*Gorilla gorilla graueri*



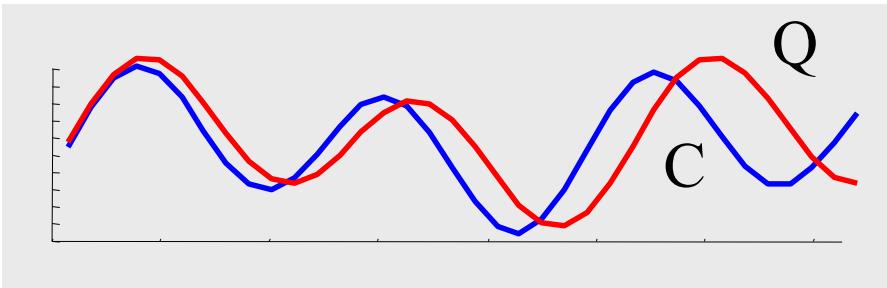
DTW is needed  
for most natural  
objects...



Mountain Gorilla  
*Gorilla gorilla beringei*



# How is DTW Calculated? I

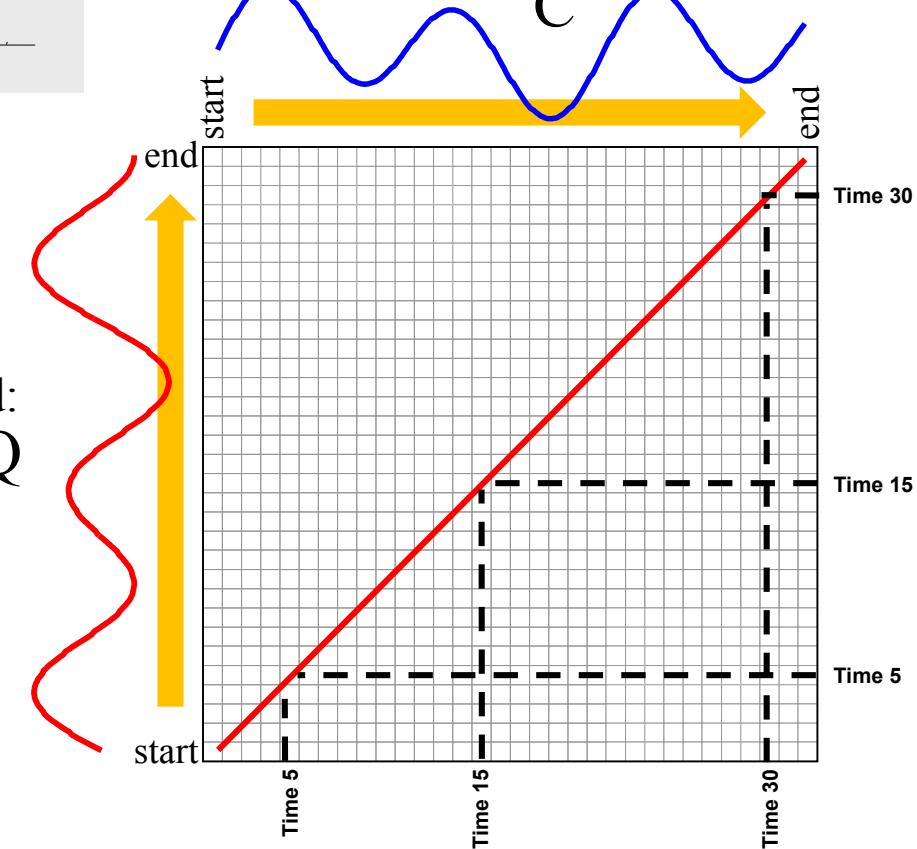


We create a matrix the size of  $|Q|$  by  $|C|$ , then fill it in with the distance between every pair of point in our two time series.

The **Euclidean distance** works only on the diagonal of the matrix

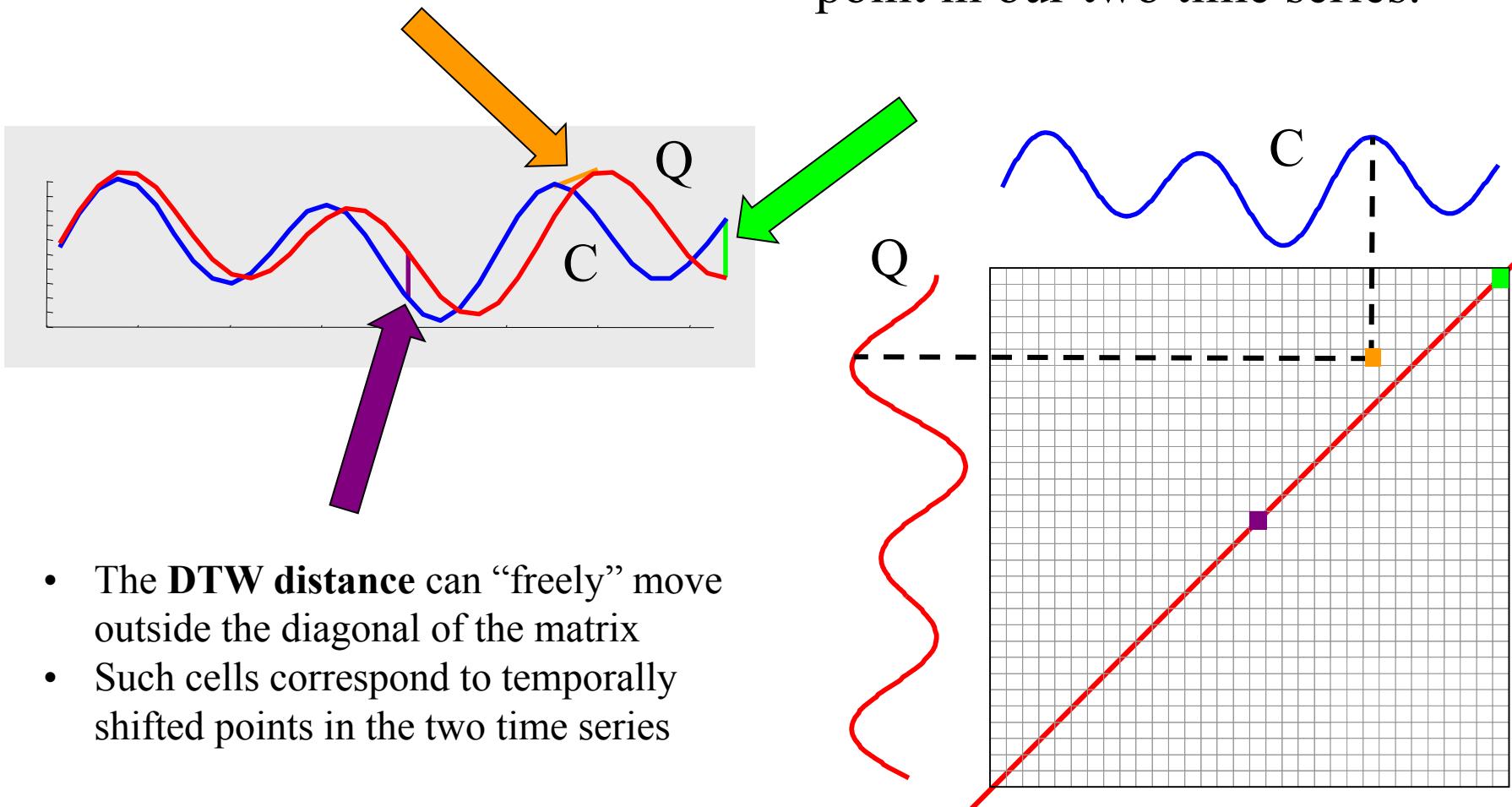
The sequence of comparisons performed:

- Start from pair of points  $(0,0)$
- After point  $(i,i)$  move to  $(i+1,i+1)$
- End the process on  $(n,n)$



# How is DTW Calculated? I

We create a matrix the size of  $|Q|$  by  $|C|$ , then fill it in with the distance between every pair of point in our two time series.

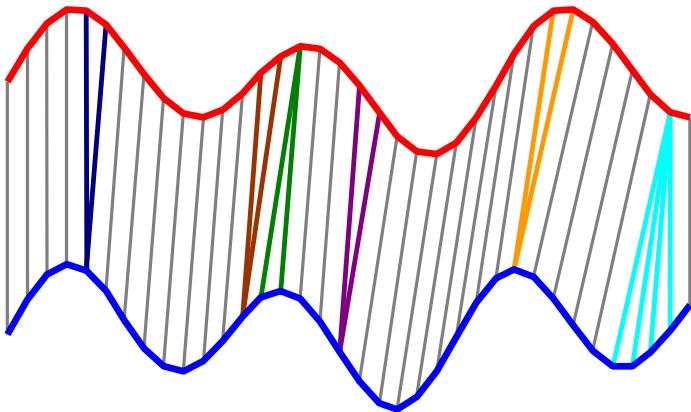


# How is DTW Calculated? II

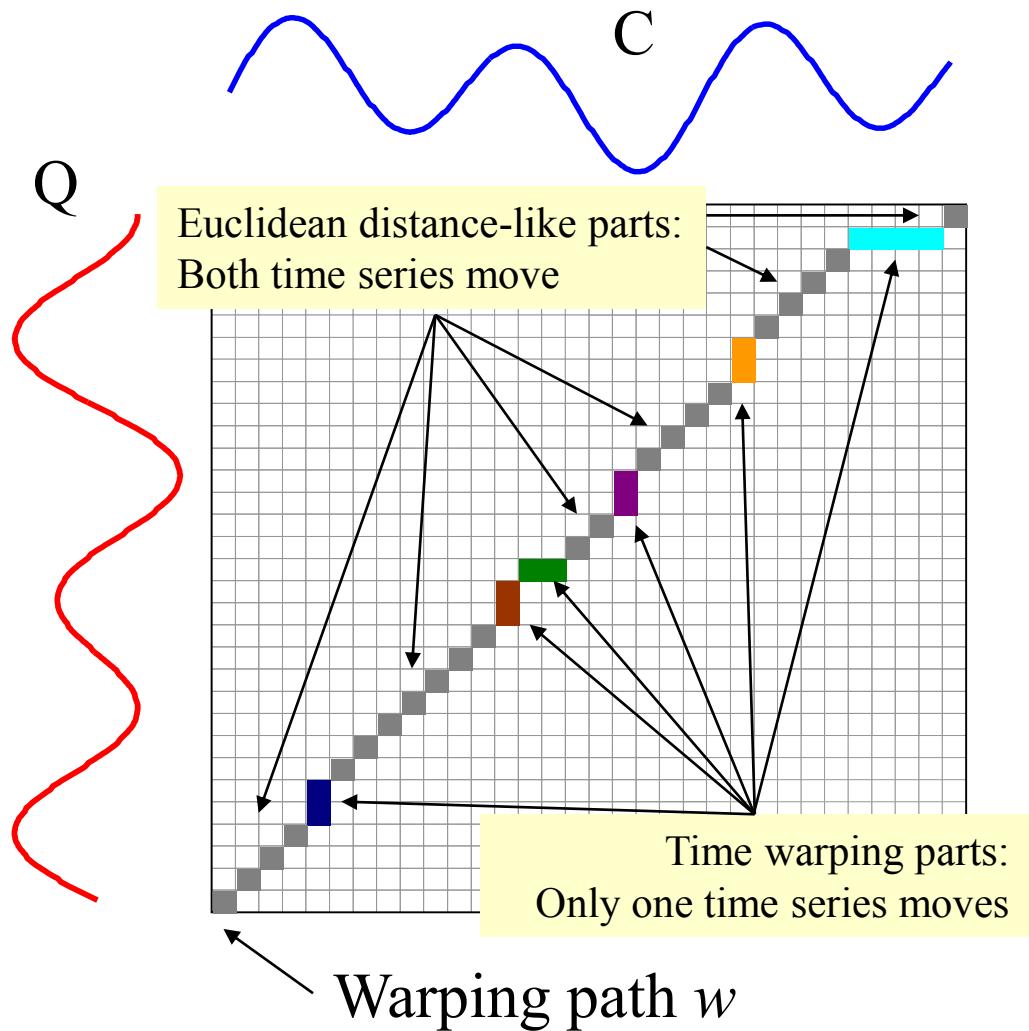
The **DTW distance** can “freely” move outside the diagonal of the matrix

The only constraints:

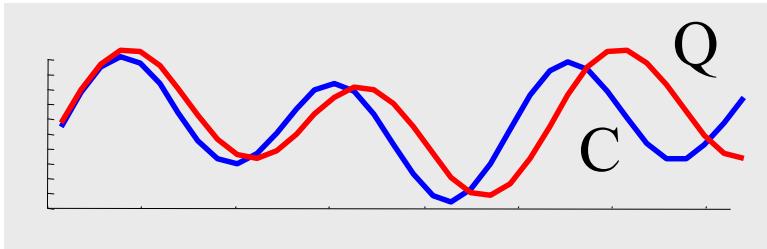
- Start from pair of points  $(0,0)$
- After point  $(i,j)$ , either “ $i$ ” or “ $j$ ” **increase by one, or both of them**
- End the process on  $(n,n)$



Every possible warping between two time series, is a path through the matrix.



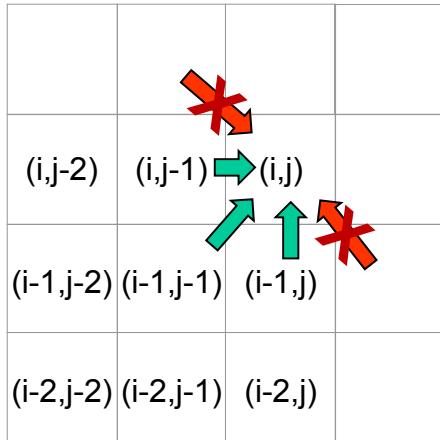
# How is DTW Calculated? II



Definition of DTW as recursive function:

$$\begin{aligned}\gamma(i,j) &= \text{cost of best path reaching cell } (i,j) \\ &= d(q_i, c_j) + \min \{ \gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1) \}\end{aligned}$$

Idea: best path must pass through  $(i-1, j)$ ,  $(i-1, j-1)$  or  $(i, j-1)$



We want the best warping path:

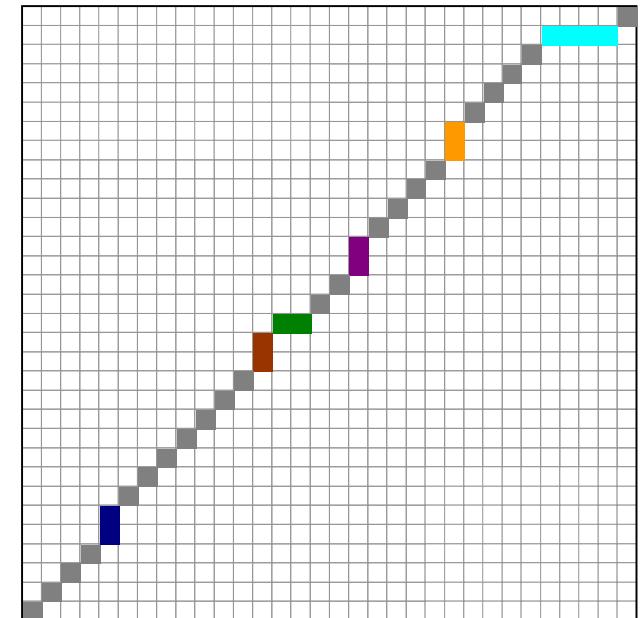
$$DTW(Q, C) = \min_{w \in PATHS} \sum_{k=1}^{|w|} w_k$$

$w_k$  = cost of the k-th points comparison

Alternatives:

- $w_k = |Q_i - C_j|$
- $w_k = [Q_i - C_j]^2$

$(i, j)$  = position of  $w_k$



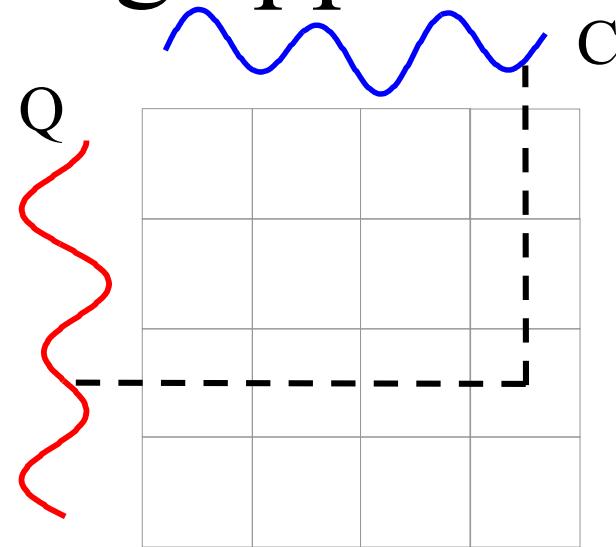
**Remark:**  $w$  can be longer than “n”

# Dynamic programming approach

$$\gamma(i,j) = d(q_i, c_j) + \min\{ \gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1) \}$$

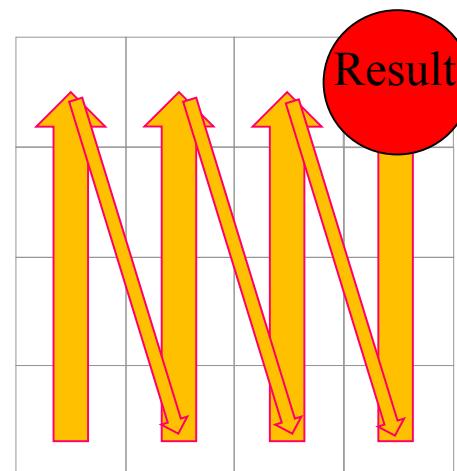
Step 1: compute the matrix of all  $d(q_i, c_j)$

- Point-to-point distances
- $D(i,j) = | Q_i - C_j |$



Step 2: compute the matrix of all path costs  $\gamma(i,j)$

- Start from cell (1,1)
- Compute (2,1), (3,1), ..., (n,1)
- Repeat for columns 2, 3, ..., n
- Final result in last cell computed



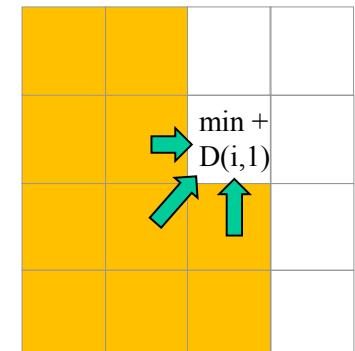
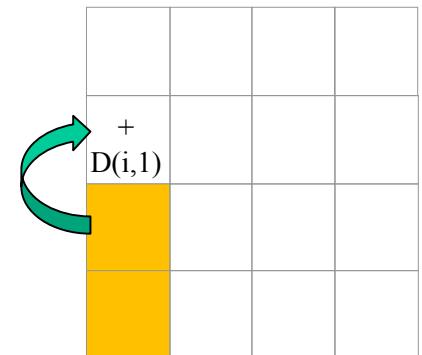
# Dynamic programming approach

$$\gamma(i,j) = d(q_i, c_j) + \min\{ \gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1) \}$$

Step 2: compute the matrix of all path costs  $\gamma(i,j)$

- Start from cell  $(1,1)$ 
  - $$\begin{aligned} \gamma(1,1) &= d(q_1, c_1) + \min\{ \gamma(0,0), \gamma(0,1), \gamma(1,0) \} \\ &= d(q_1, c_1) \\ &= D(1,1) \end{aligned}$$
- Compute  $(2,1), (3,1), \dots, (n,1)$ 
  - $$\begin{aligned} \gamma(i,1) &= d(q_i, c_1) + \min\{ \gamma(i-1,0), \gamma(i-1,1), \gamma(i,0) \} \\ &= d(q_i, c_1) + \gamma(i-1,1) \\ &= D(i,1) + \gamma(i-1,1) \end{aligned}$$
- Repeat for columns 2, 3, ..., n
  - The general formula applies

D(1,1)			



# Dynamic programming approach

## Example and How to infer the optimal path

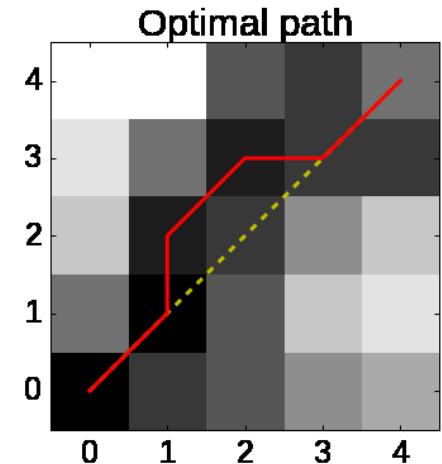
$$t = \langle 3, 7, 4, 3, 4 \rangle$$

$$q = \langle 5, 7, 6, 4, 2 \rangle$$

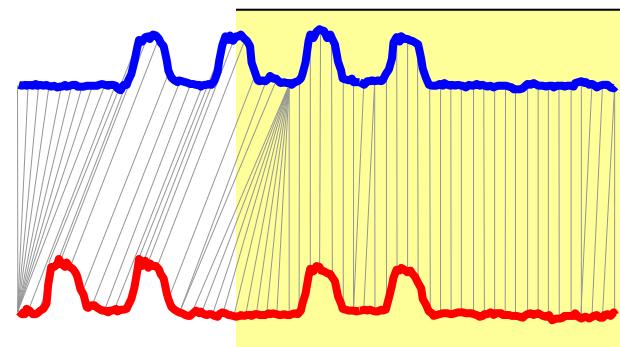
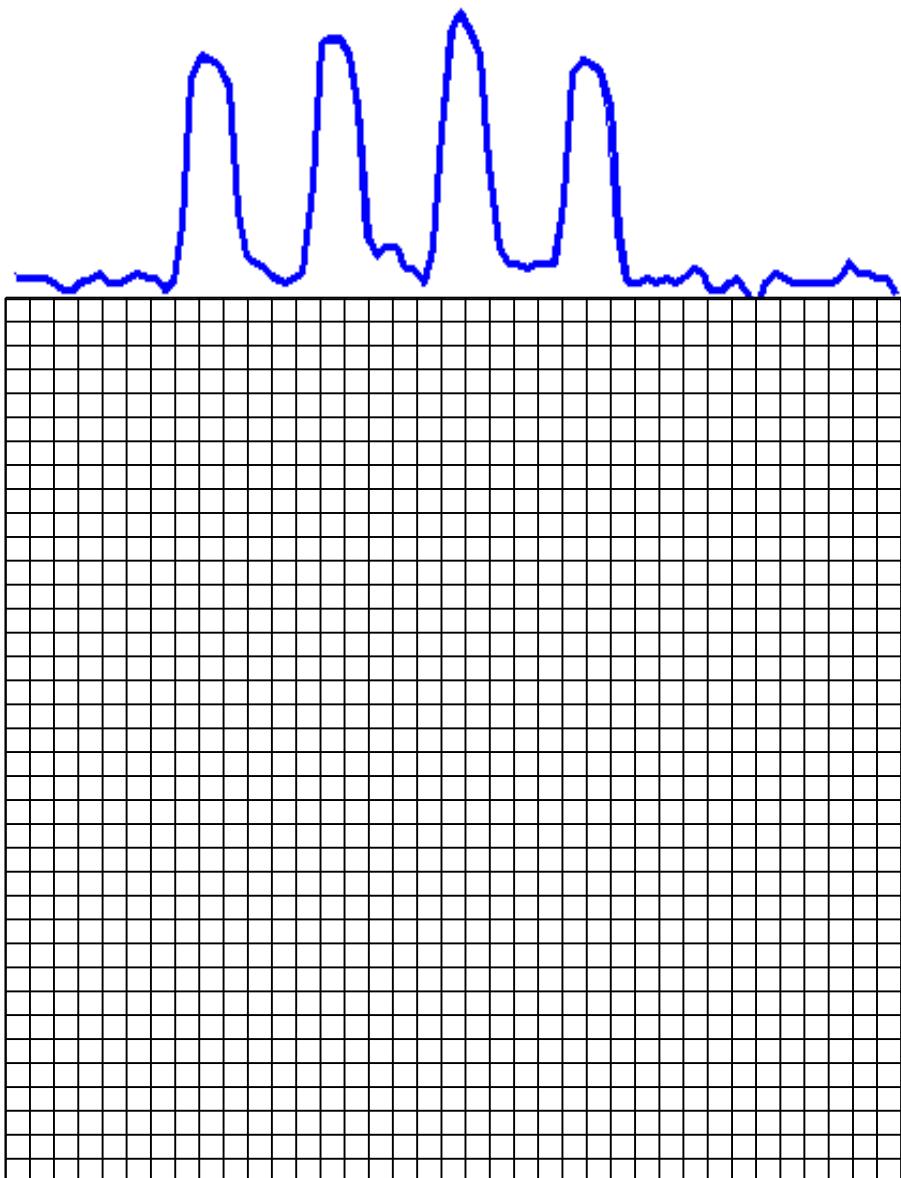
$$\gamma(i,j) = d(q_i, c_j) + \min\{ \gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1) \}$$

Point-to-point costs				
5	2	1	2	
3	1	2	3	2
4	0	3	4	3
2	2	1	2	1
3	7	4	3	4

Cumulative costs				
2	4	6	7	8
3	11	11	5	4
4	10	6	3	4
6	9	4	7	9
7	6	2	5	9
5	2	4	5	7
3	7	4	3	4



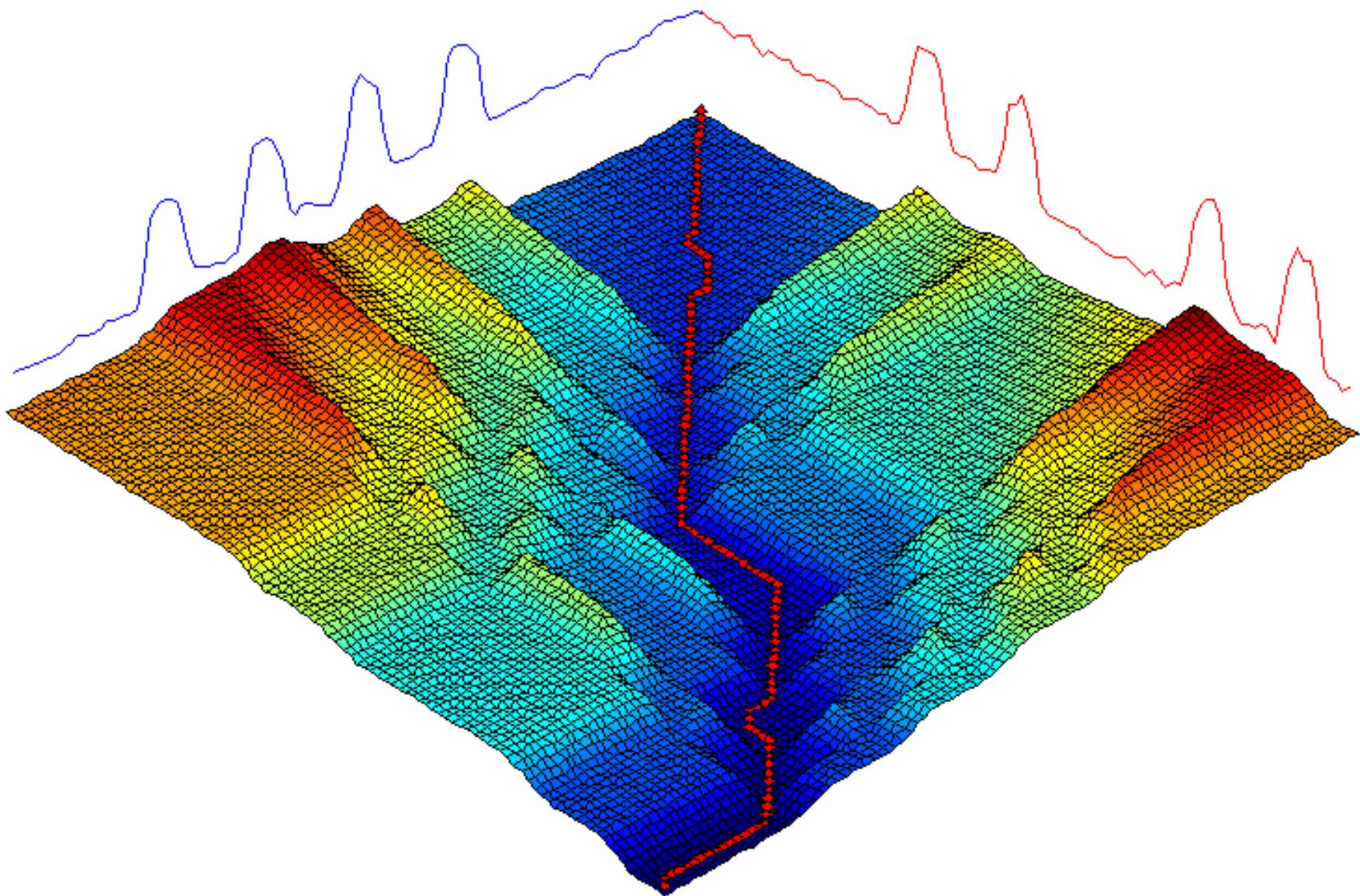
# Let us visualize the cumulative matrix on a real world problem I



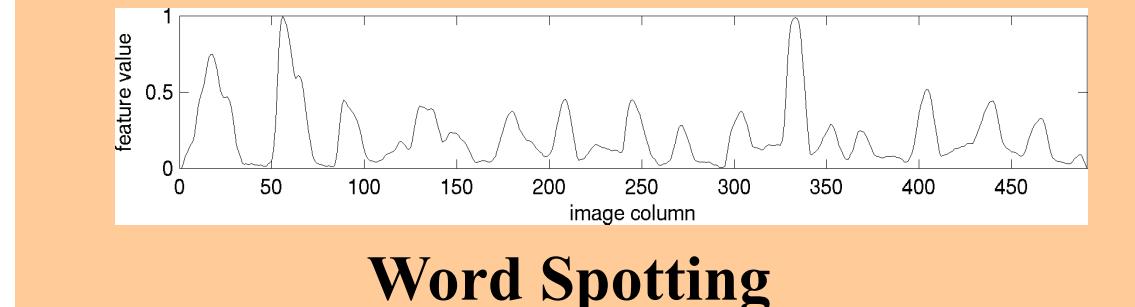
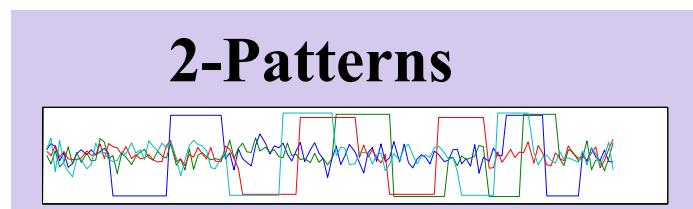
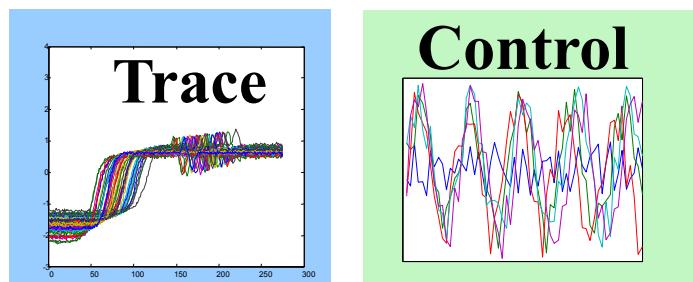
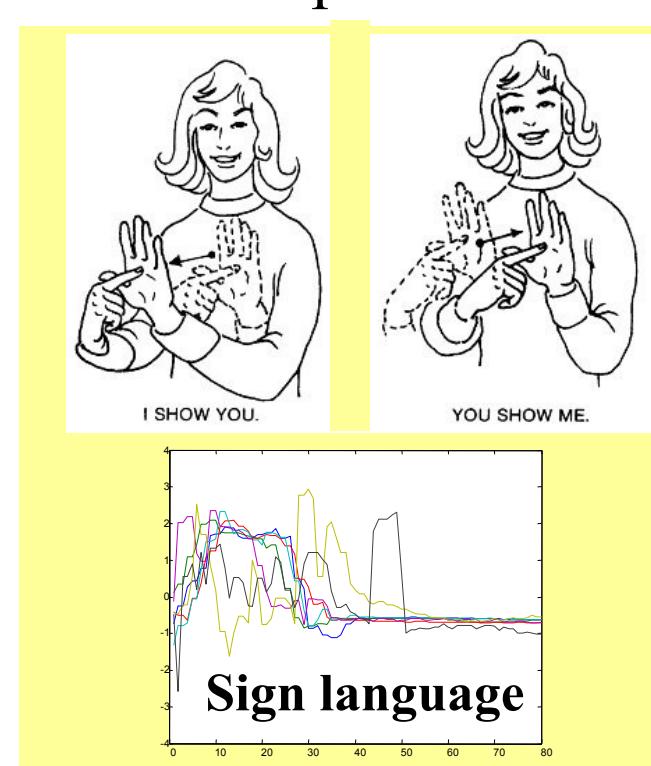
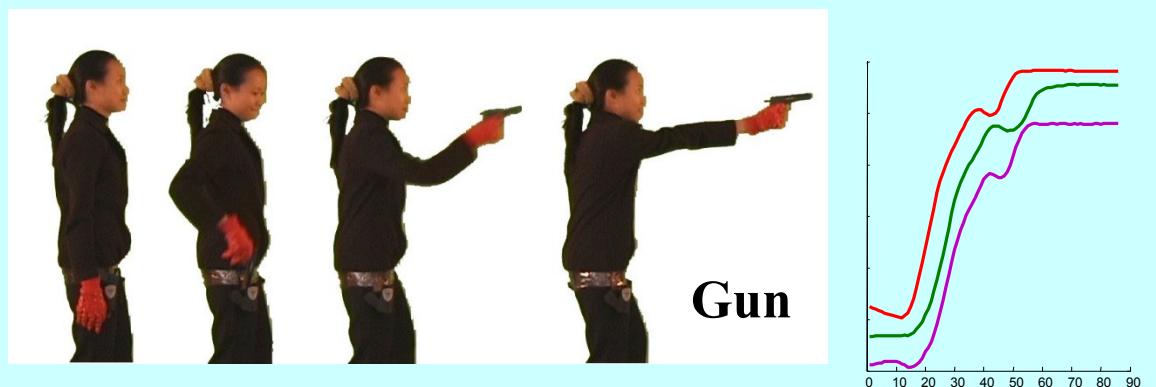
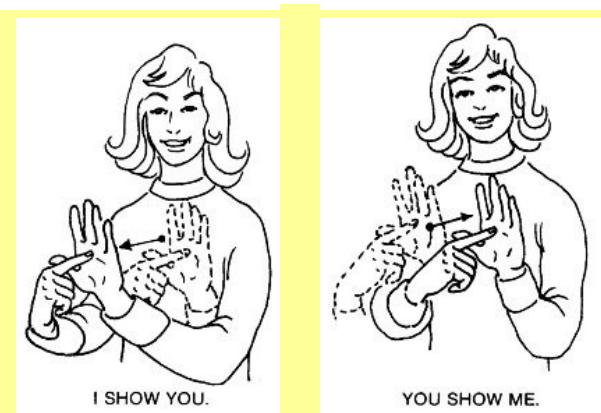
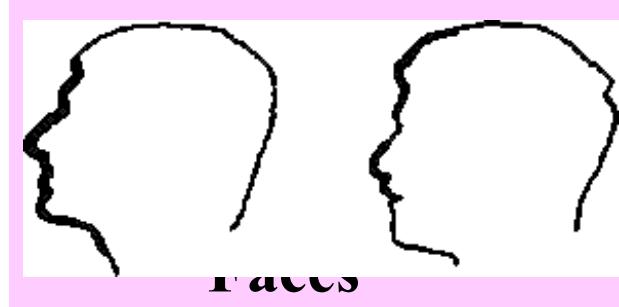
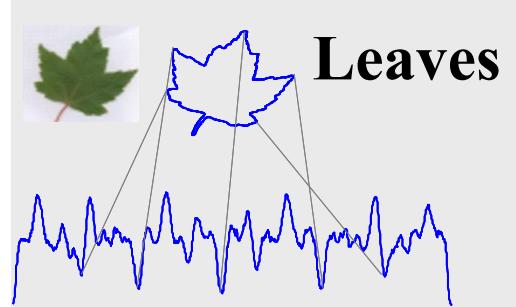
This example shows 2 one-week periods from the power demand time series.

Note that although they both describe 4-day work weeks, the blue sequence had Monday as a holiday, and the red sequence had Wednesday as a holiday.

Let us visualize the cumulative matrix on a real world problem II



# Let us compare Euclidean Distance and DTW on some problems



# Results: Error Rate

Classification using **1-nearest-neighbor**

- $\text{Class}(x)$  = class of most similar training object

Leaving-one-out evaluation

- For each object: use it as test set, return overall average

Dataset	Euclidean	DTW
Word Spotting	4.78	1.10
Sign language	28.70	25.93
GUN	5.50	1.00
Nuclear Trace	11.00	0.00
Leaves <sup>#</sup>	33.26	4.07
(4) Faces	6.25	2.68
Control Chart*	7.5	0.33
2-Patterns	1.04	0.00



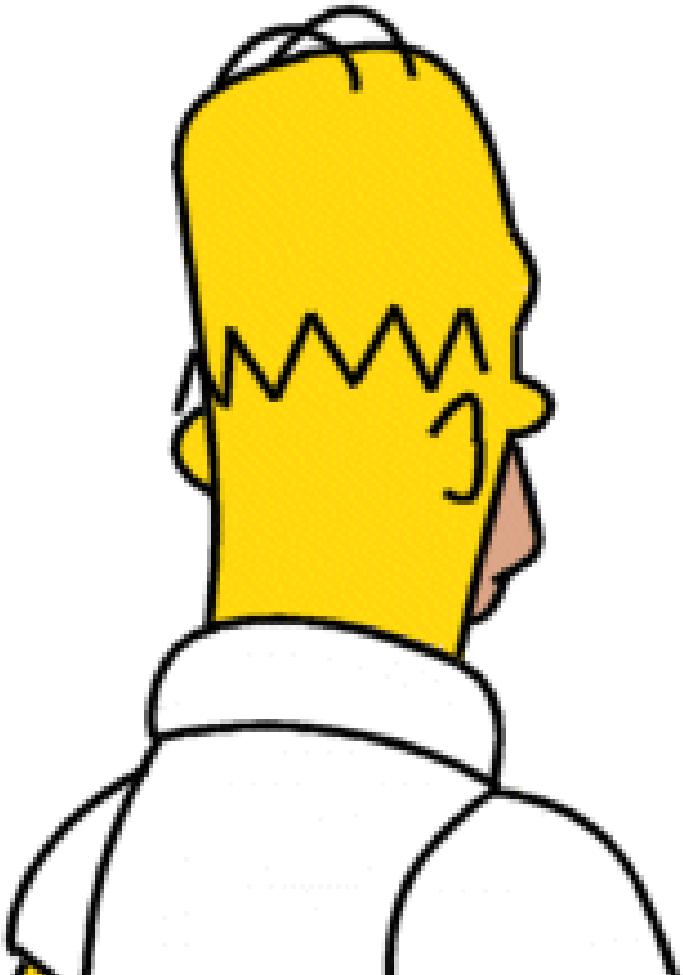
# Results: Time (msec )

Dataset	Euclidean	DTW
Word Spotting	40	8,600
Sign language	10	1,110
GUN	60	11,820
Nuclear Trace	210	144,470
Leaves	150	51,830
(4) Faces	50	45,080
Control Chart	110	21,900
2-Patterns	16,890	545,123

DTW is  
two to  
three  
orders of  
magnitude  
slower  
than  
Euclidean  
distance



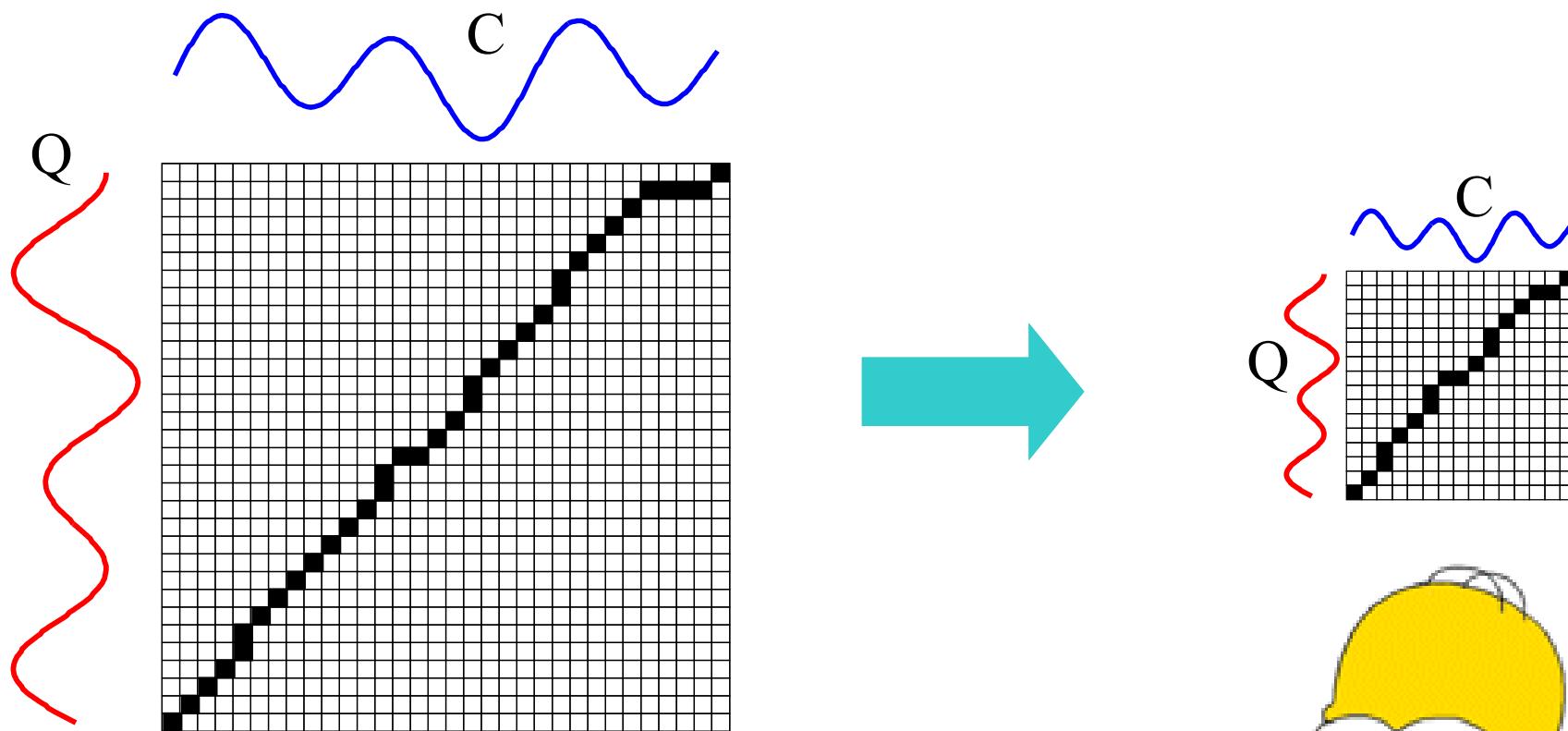
# What we have seen so far...



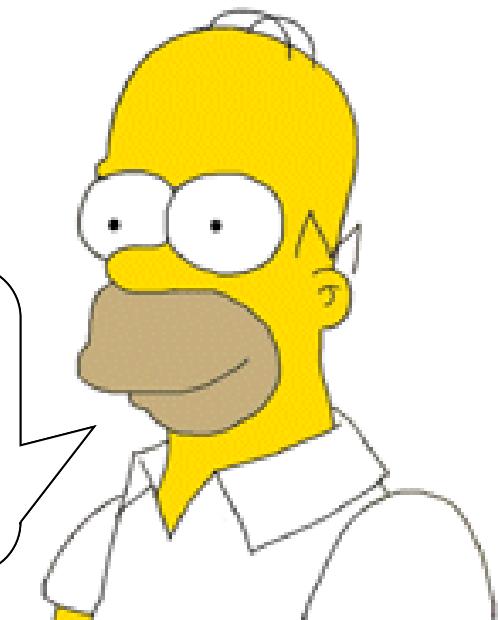
- Dynamic Time Warping gives **much better** results than Euclidean distance on virtually all problems.
- Dynamic Time Warping is very very slow to calculate!

Is there anything we can do to speed up similarity search under DTW?

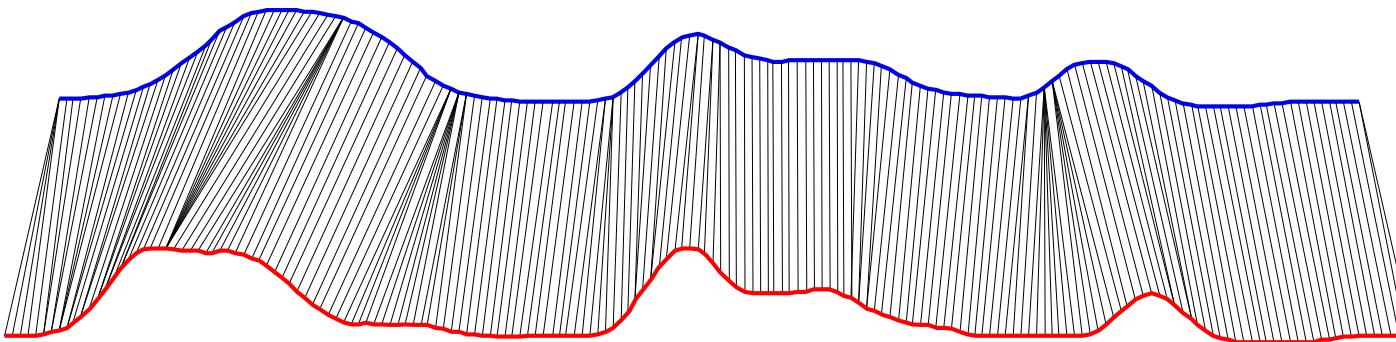
# Fast Approximations to Dynamic Time Warp Distance I



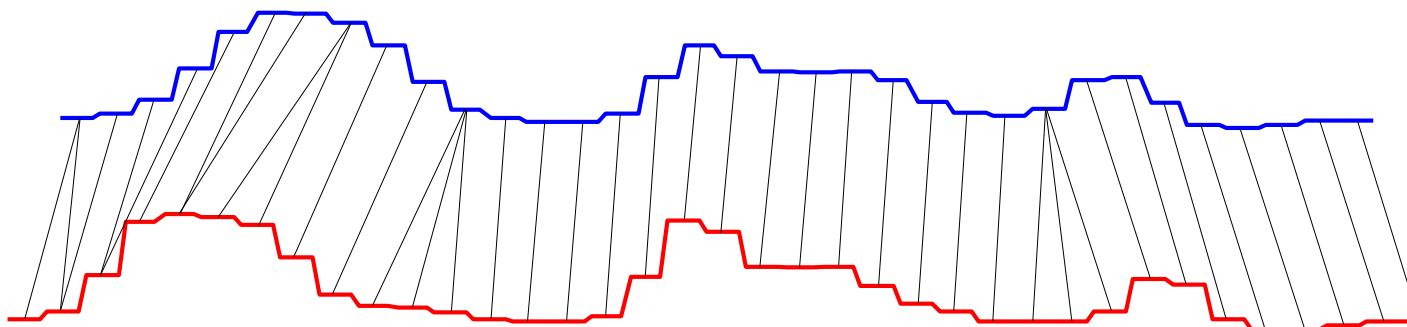
Simple Idea: Approximate the time series with some compressed or downsampled representation, and do DTW on the new representation. How well does this work...



# Fast Approximations to Dynamic Time Warp Distance II



1.03 sec



0.07 sec

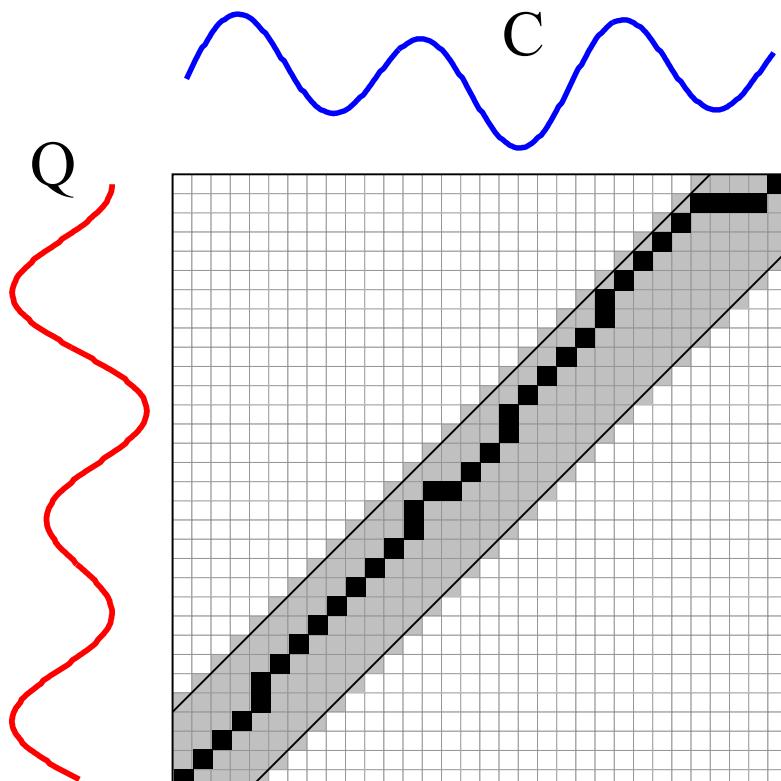


... there is strong visual evidence to suggests it works well

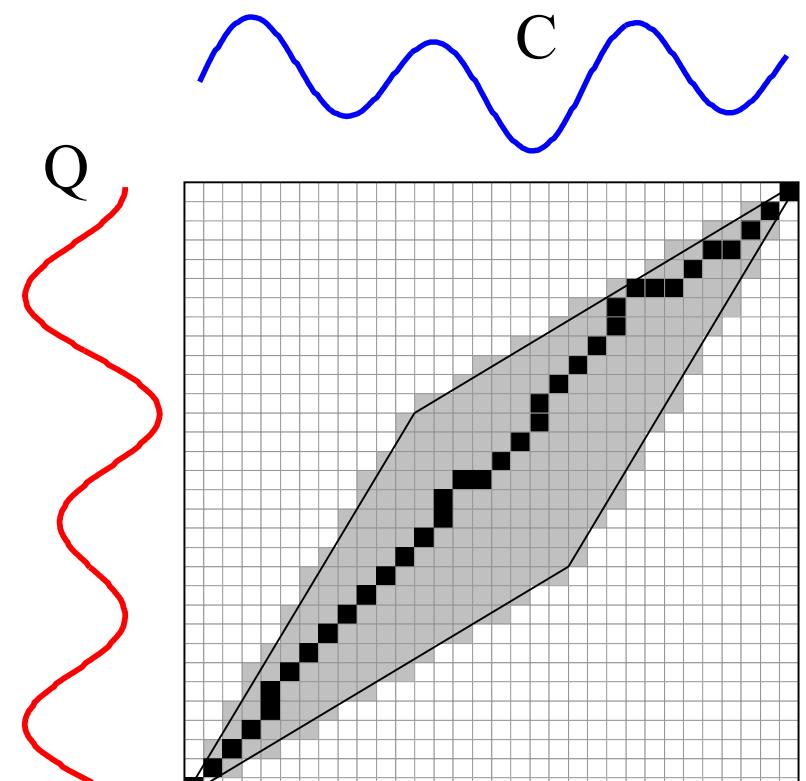
There is good experimental evidence for the utility of the approach on clustering, classification, etc

# Global Constraints

- Slightly speed up the calculations
- Prevent pathological warpings



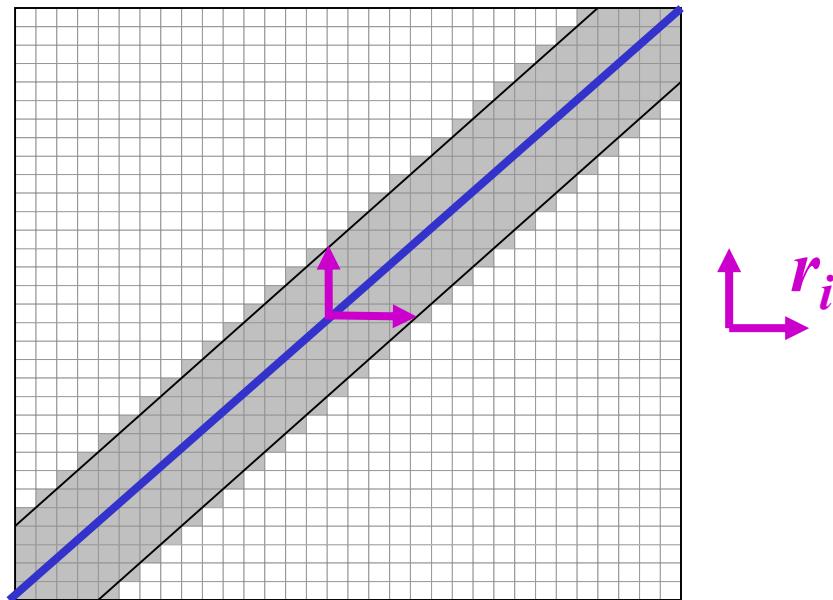
Sakoe-Chiba Band



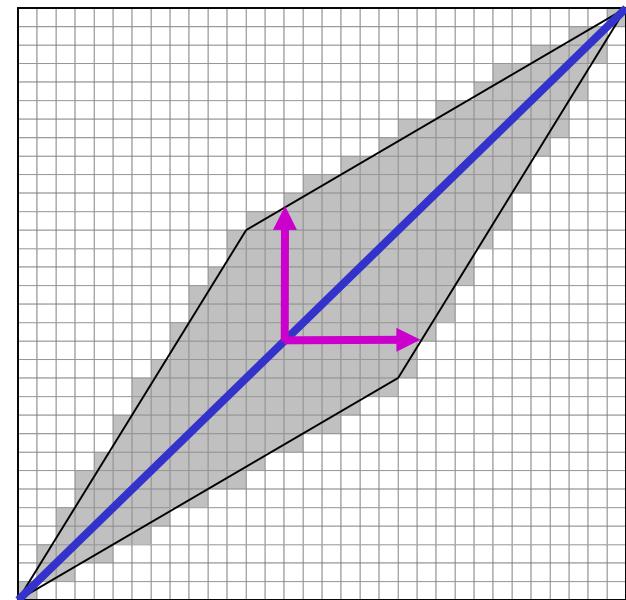
Itakura Parallelogram

A global constraint constrains the indices of the warping path  $w_k = (i, j)_k$  such that  $j-r \leq i \leq j+r$

Where  $r$  is a term defining allowed range of warping for a given point in a sequence.

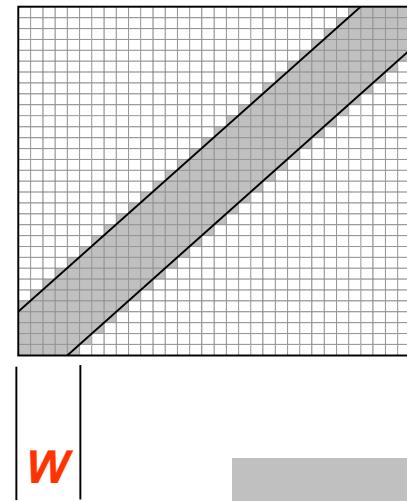
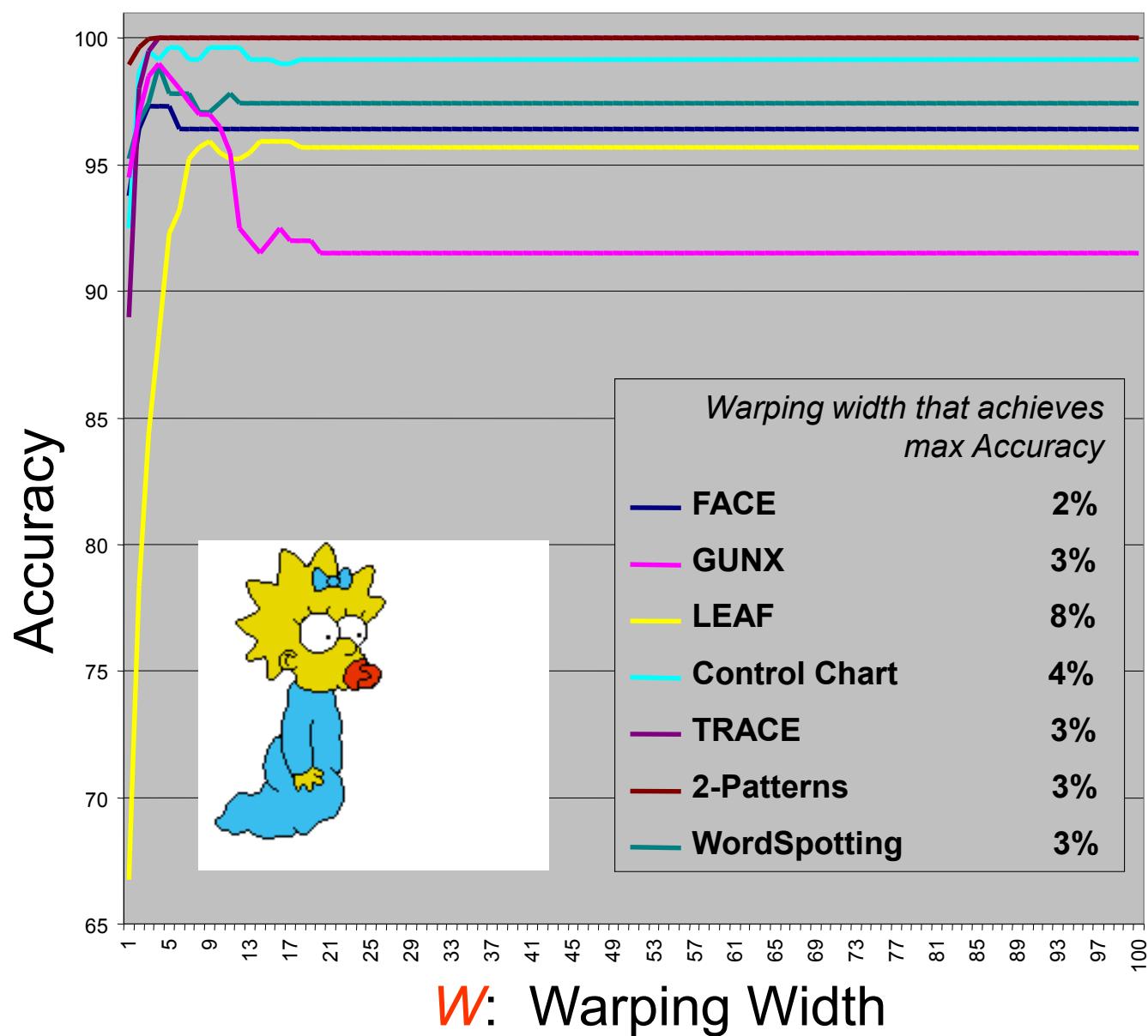


Sakoe-Chiba Band



Itakura Parallelogram

# Accuracy vs. Width of Warping Window

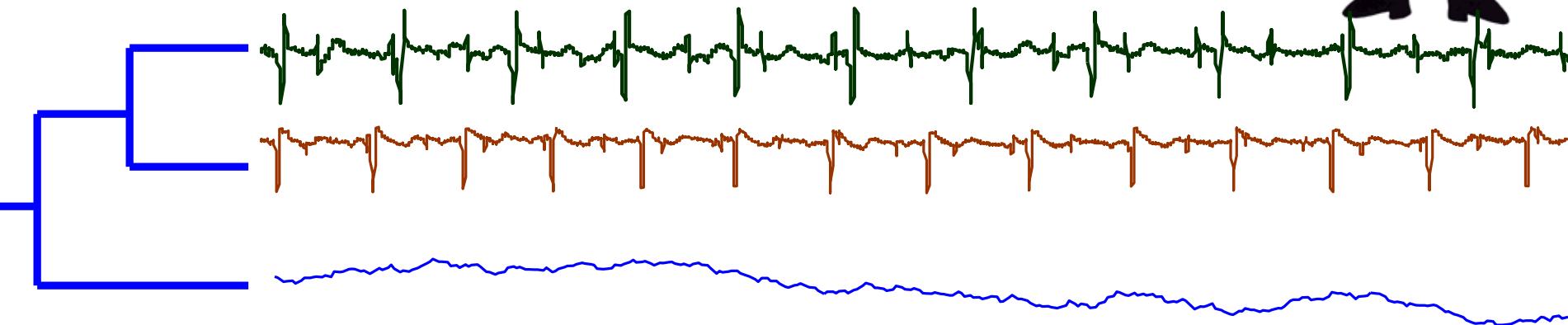


We are  
done with  
*shape*  
similarity

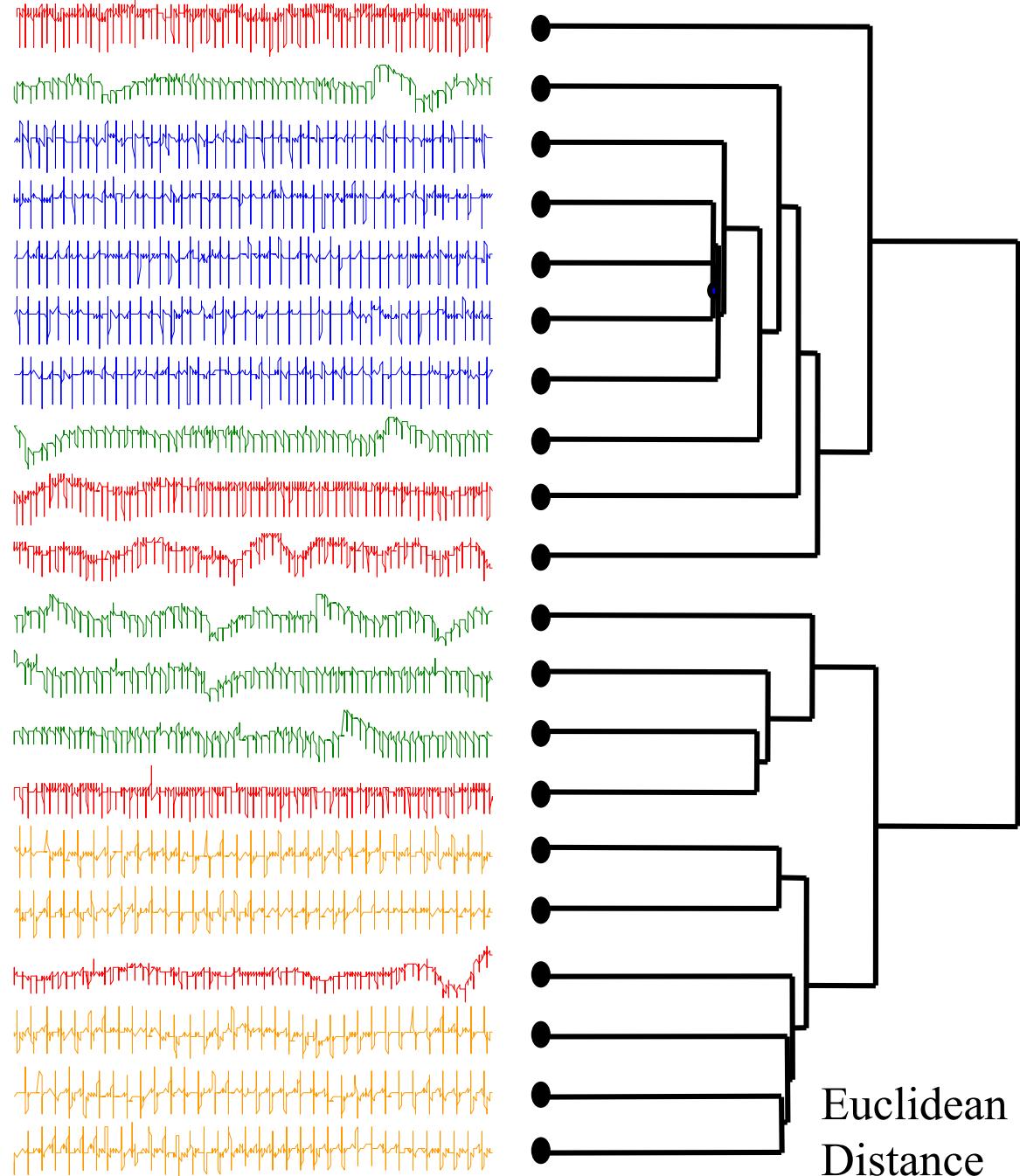
# Two Kinds of Similarity



Let us consider  
similarity at  
the *structural*/  
level for the  
next 10 minutes



For long time series, shape based similarity will give very poor results. We need to measure similarly based on high level structure



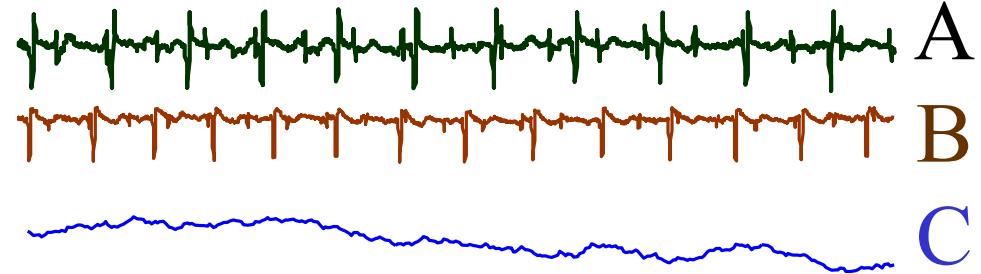
Euclidean Distance

# Structure or Model Based Similarity

The basic idea is to extract *global* features from the time series, create a feature vector, and use these feature vectors to measure similarity and/or classify



But which  
• **features?**  
• **distance measure/learning algorithm?**



Feature \ Time Series	A	B	C
Max Value	11	12	19
Autocorrelation	0.2	0.3	0.5
Zero Crossings	98	82	13
ARIMA	0.3	0.4	0.1
...	...	...	...

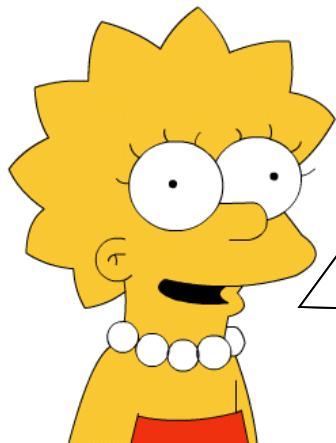
# Feature-based Classification of Time-series Data

Nanopoulos, Alcock, and Manolopoulos

- features?
- distance measure/  
learning algorithm?

## Learning Algorithm

multi-layer perceptron neural network



Makes sense, but when  
we looked at the *same*  
dataset, we found we  
could be better  
classification accuracy  
with Euclidean  
distance!

## Features

mean

variance

skewness

kurtosis

mean (1<sup>st</sup> derivative)

variance (1<sup>st</sup> derivative)

skewness (1<sup>st</sup> derivative)

kurtosis (1<sup>st</sup> derivative)

# Learning to Recognize Time Series: Combining ARMA Models with Memory-Based Learning

Deng, Moore and Nechyba

- features?
- distance measure/  
learning algorithm?

## Distance Measure

Euclidean distance (between coefficients)

- Use to detect drunk drivers!
- Independently rediscovered and generalized by Kalpakis et. al. and expanded by Xiong and Yeung

### Features

The parameters of the Box Jenkins model.

More concretely, the coefficients of the ARMA model.

*“Time series must be invertible and stationary”*

# Deformable Markov Model Templates for Time Series Pattern Matching

Ge and Smyth

## Part 1

- features?
- distance measure/  
learning algorithm?

## Distance Measure

“Viterbi-Like” Algorithm

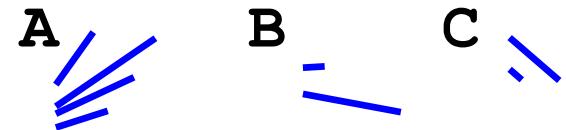
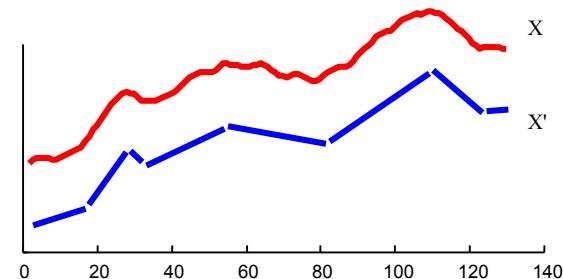
Variations independently developed by Li and Biswas, Ge and Smyth, Lin, Orgun and Williams etc

There tends to be lots of parameters to tune...

	A	B	C
A	0.1	0.4	0.5
B	0.4	0.2	0.2
C	0.5	0.2	0.3

The parameters of a Markov Model

The time series is first converted to a piecewise linear model

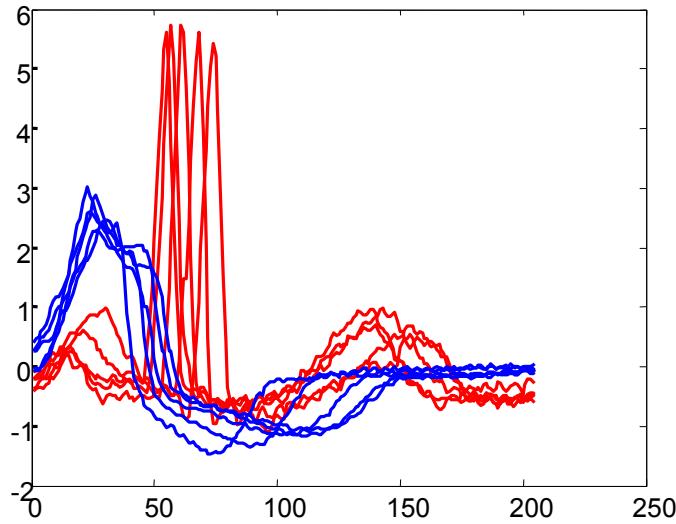


# Deformable Markov Model Templates for Time Series Pattern Matching

Ge and Smyth

## Part 2

On this problem  
the approach  
gets 98%  
classification  
accuracy\*...



### Features

The parameters of a  
Markov Model

The time series is first  
converted to a piecewise  
linear model



But Euclidean distance  
gets 100%! And has no  
parameters to tune, and  
is tens of thousands  
times faster...



# Compression Based Dissimilarity

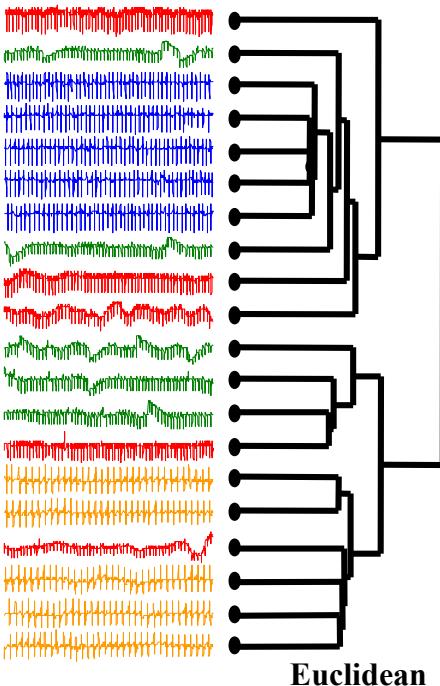
(In general) Li, Chen, Li, Ma, and Vitányi: (For time series) Keogh, Lonardi and Ratanamahatana

- features?
- distance measure/  
learning algorithm?

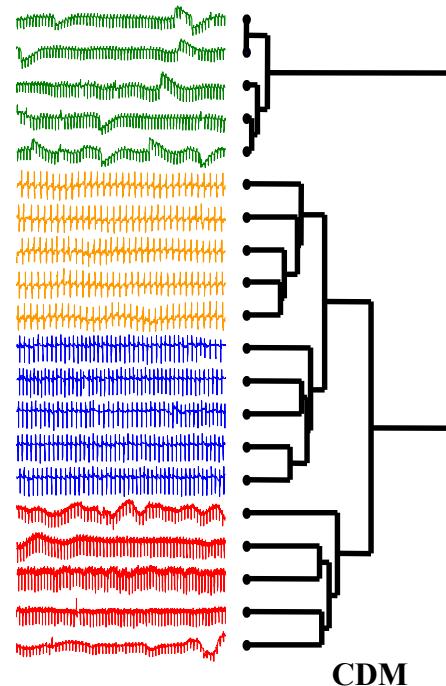
## Distance Measure

### Co-Compressibility

$$CDM(x, y) = \frac{C(xy)}{C(x) + C(y)}$$



Euclidean



CDM

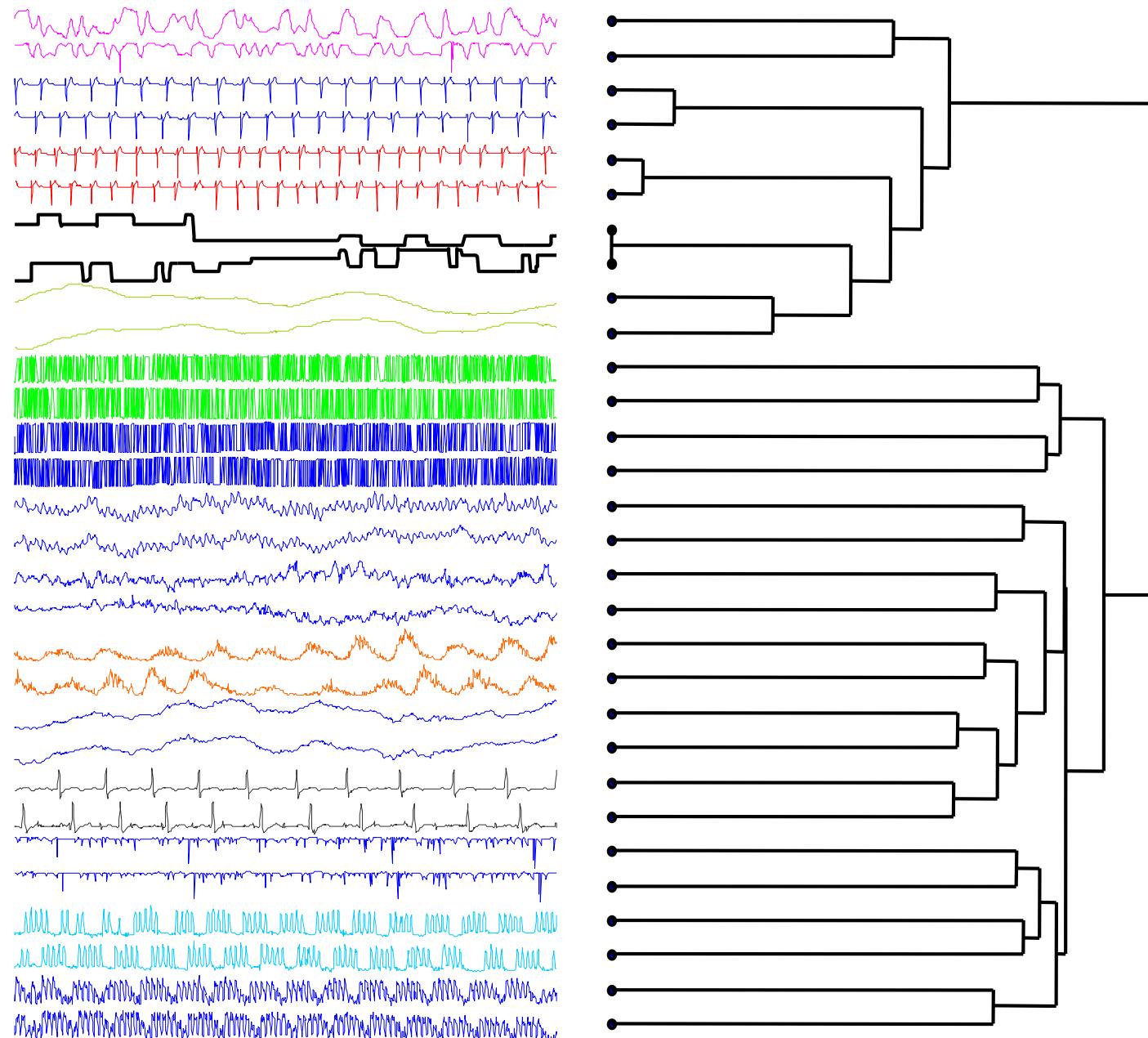
## Features

Whatever structure  
the compression  
algorithm finds...

The time series is first converted  
to the SAX symbolic  
representation\*

# Compression Based Dissimilarity

Reel 2: Tension  
Reel 2: Angular speed  
Koski ECG: Fast 2  
Koski ECG: Fast 1  
Koski ECG: Slow 2  
Koski ECG: Slow 1  
Dryer hot gas exhaust  
Dryer fuel flow rate  
Ocean 2  
Ocean 1  
Evaporator: vapor flow  
Evaporator: feed flow  
Furnace: cooling input  
Furnace: heating input  
Great Lakes (Ontario)  
Great Lakes (Erie)  
Buoy Sensor: East Salinity  
Buoy Sensor: North Salinity  
Sunspots: 1869 to 1990  
Sunspots: 1749 to 1869  
Exchange Rate: German Mark  
Exchange Rate: Swiss Franc  
Foetal ECG thoracic  
Foetal ECG abdominal  
Balloon2 (lagged)  
Balloon1  
Power : April-June (Dutch)  
Power : Jan-March (Dutch)  
Power : April-June (Italian)  
Power : Jan-March (Italian)



# Summary of Time Series Similarity

- If you have *short* time series
  - use DTW after searching over the warping window size
- If you have *long* time series,
  - and you know nothing about your data => try compression based dissimilarity
  - if you do know something about your data => extract features

# Time series analysis tasks

- Similarity-based tasks
  - Standard clustering, classification (KNN), etc.



# 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

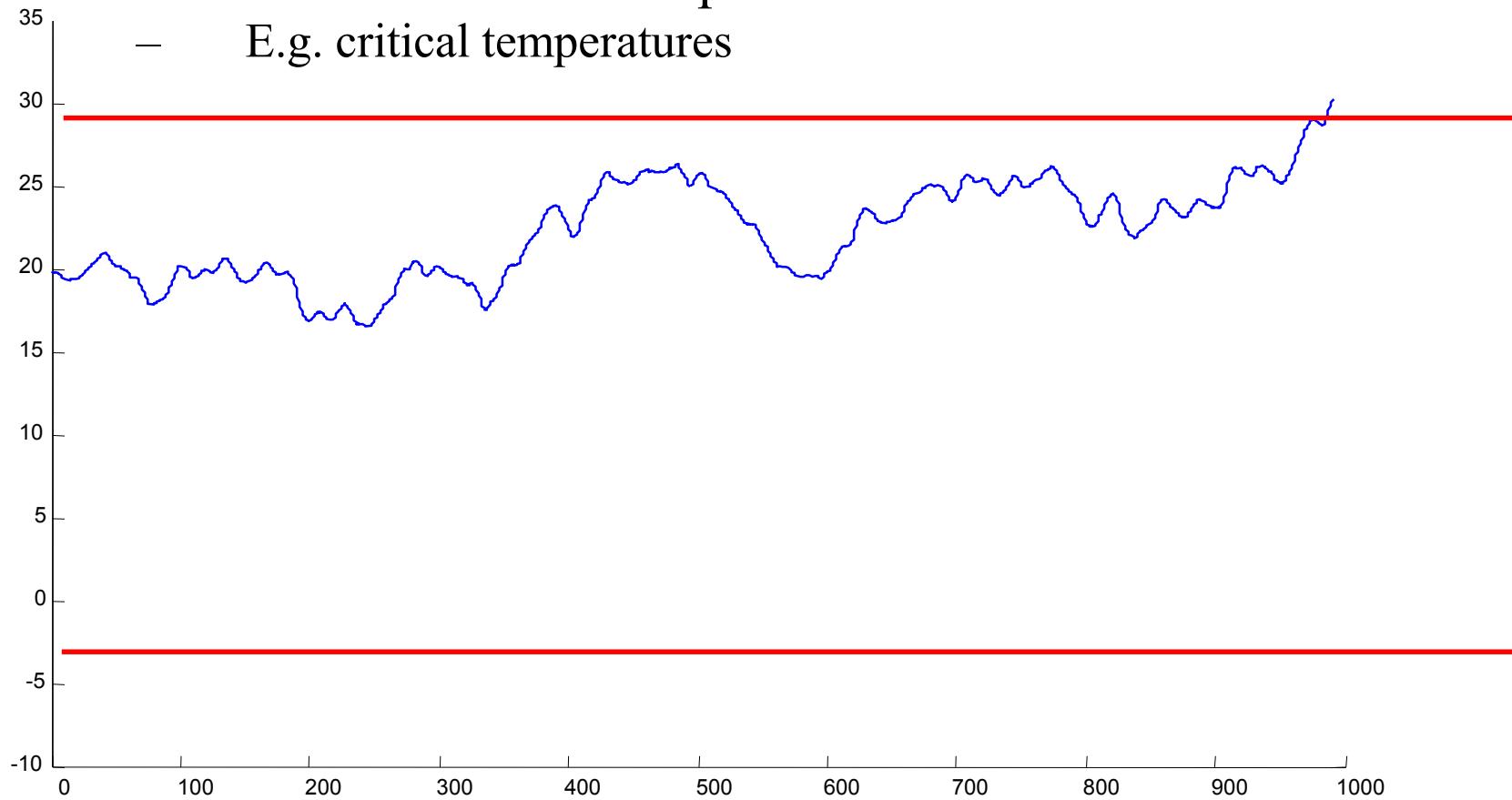
Also note that “surprising” is very context dependent, application dependent, subjective etc.



# Simple Approaches I

## Limit Checking

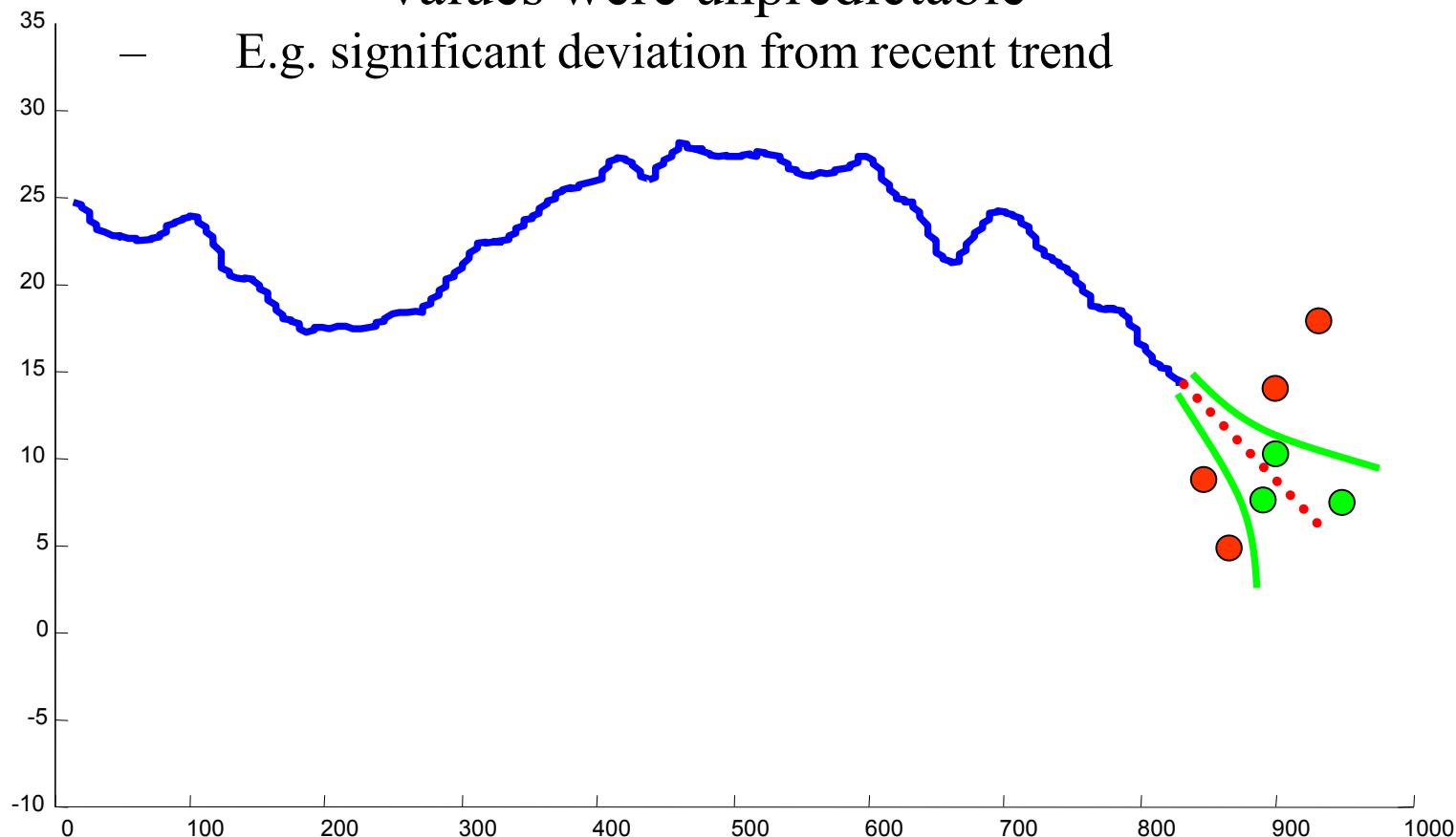
- Outliers = values outside pre-defined boundaries
  - E.g. critical temperatures



# Simple Approaches II

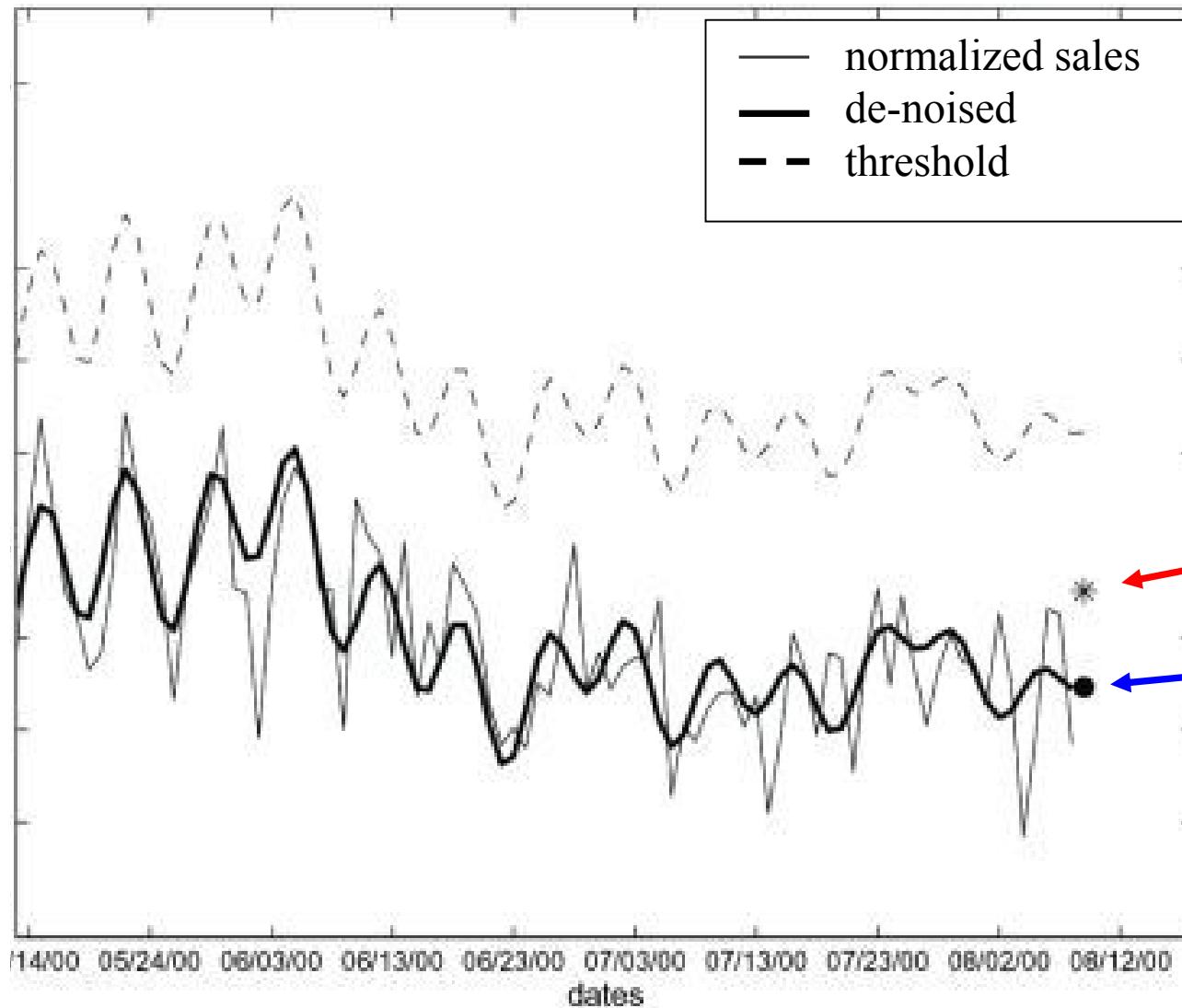
## Discrepancy Checking

- Outliers = values far from those predicted by a model  
= values were unpredictable  
E.g. significant deviation from recent trend



# Discrepancy Checking: Example

## Simple averaging predictor



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

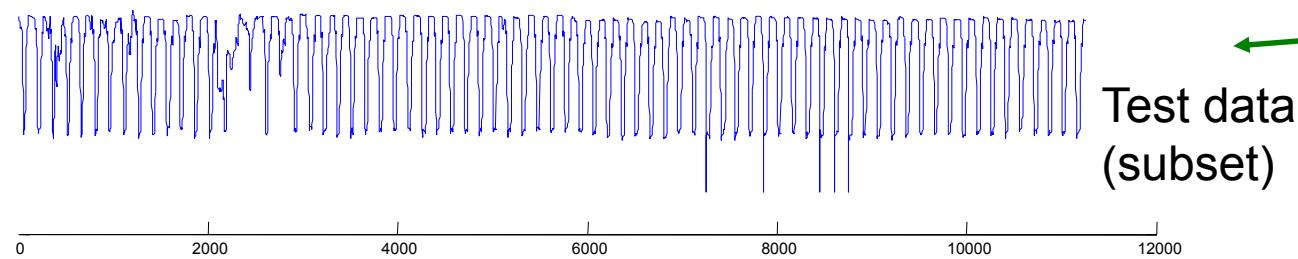
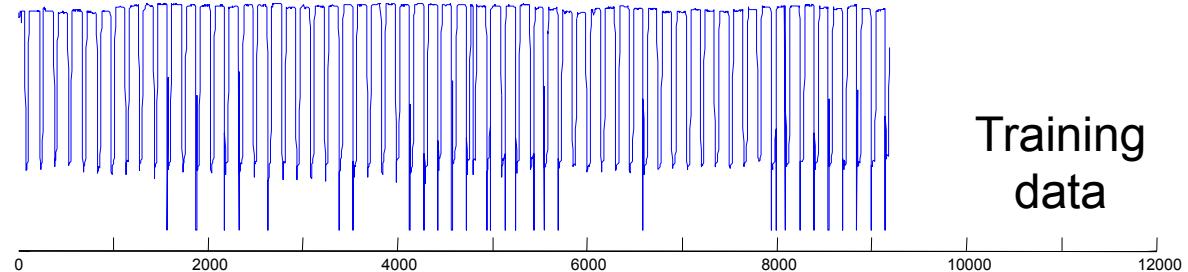
Goldenberg, Shmueli, Caruana, and Fienberg

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

# Discrepancy Checking with Markov models

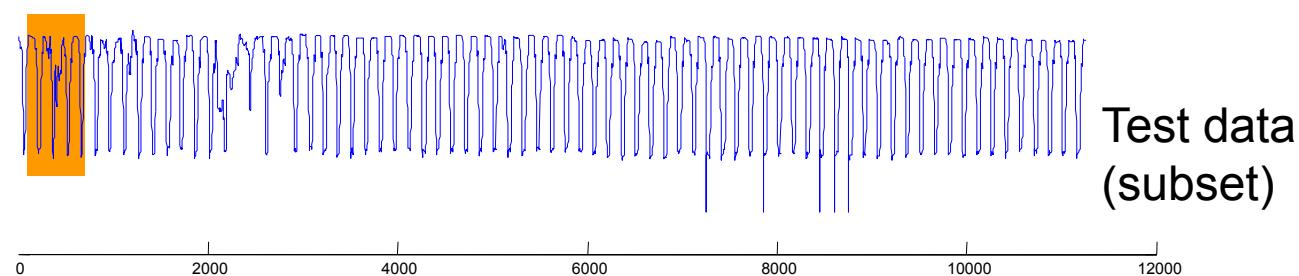
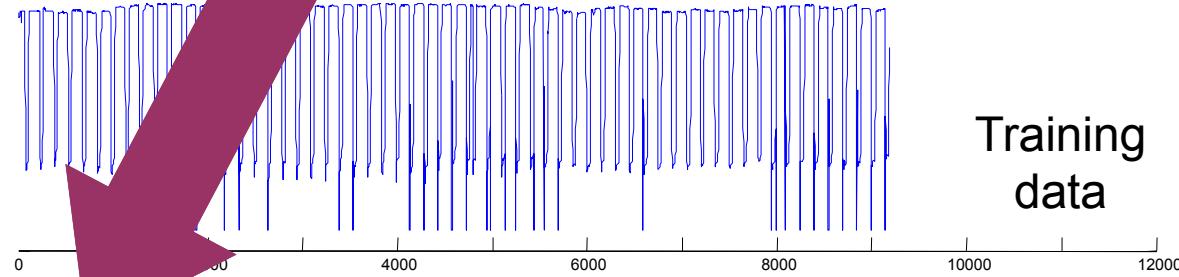
- Typical approach for text strings:
  - Take a set of text which has been labeled “normal”
  - Learn a Markov model for it
  - Any future data that is not modeled well by the Markov model you annotate as surprising.
- Time series can be easily converted to text
  - Discretization of numerical values
- We can use Markov models to find surprises in time series...

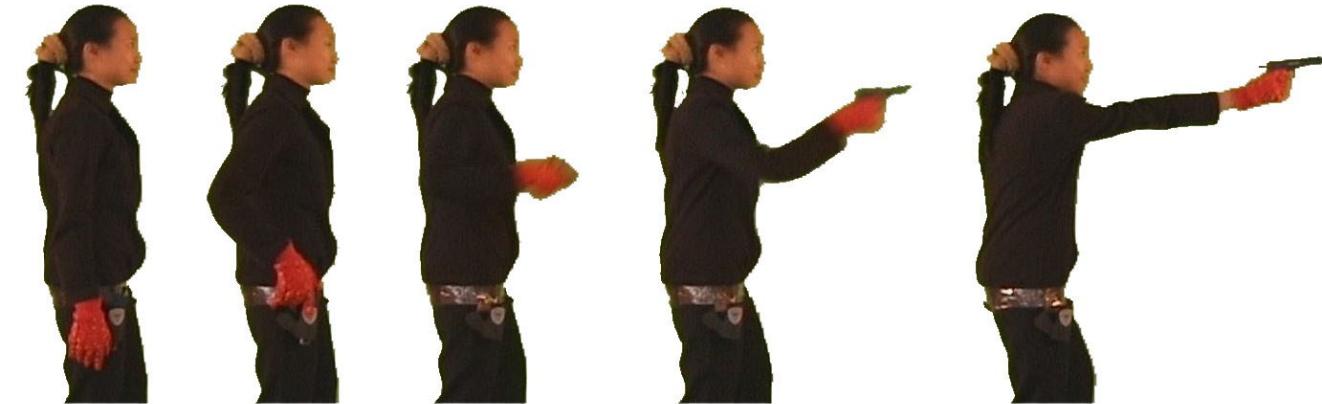
# Discrepancy Checking with Markov models



These were converted to the symbolic representation.  
I am showing the original data for simplicity

In the next slide we will zoom in on this subsection, to try to understand why it is surprising

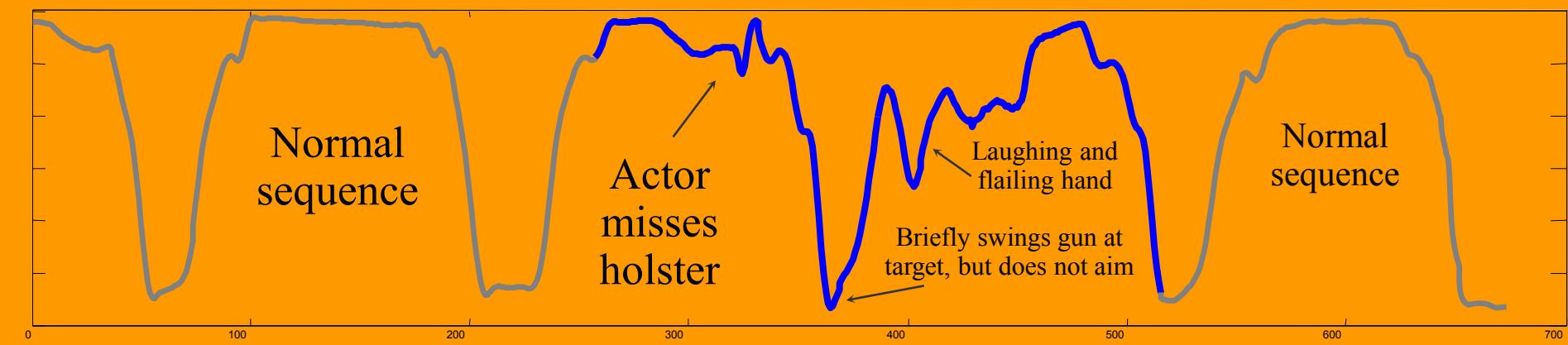




Normal Time Series



Surprising Time Series



# Anomaly (interestingness) detection

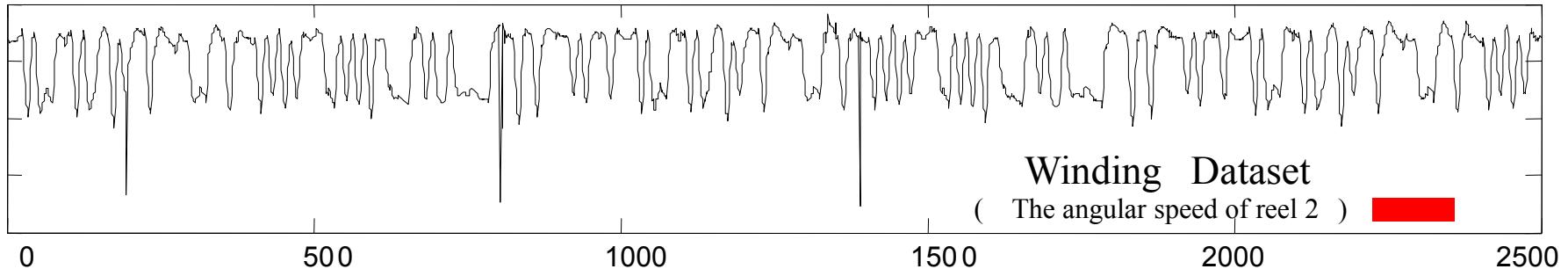
In spite of the nice example in the previous slide, the anomaly detection problem is wide open.

How can we find interesting patterns...

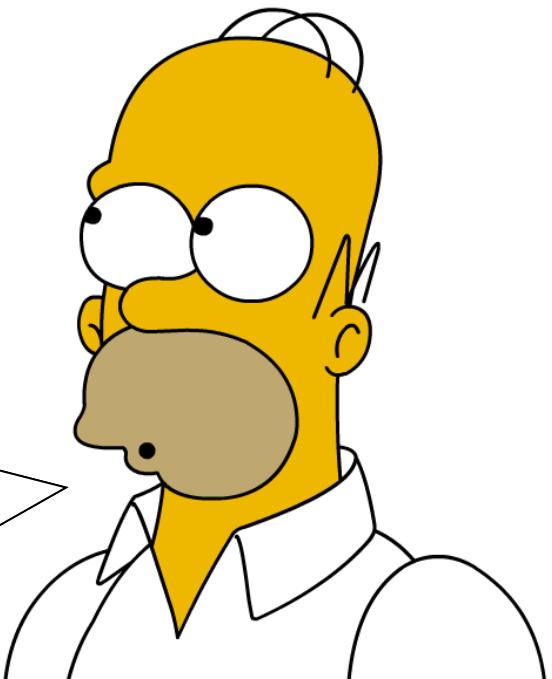
- Without (or with very few) false positives...
- In truly massive datasets...
- In the face of concept drift...
- With human input/feedback...
- With annotated data...

# Time Series Motif Discovery

(finding repeated patterns)

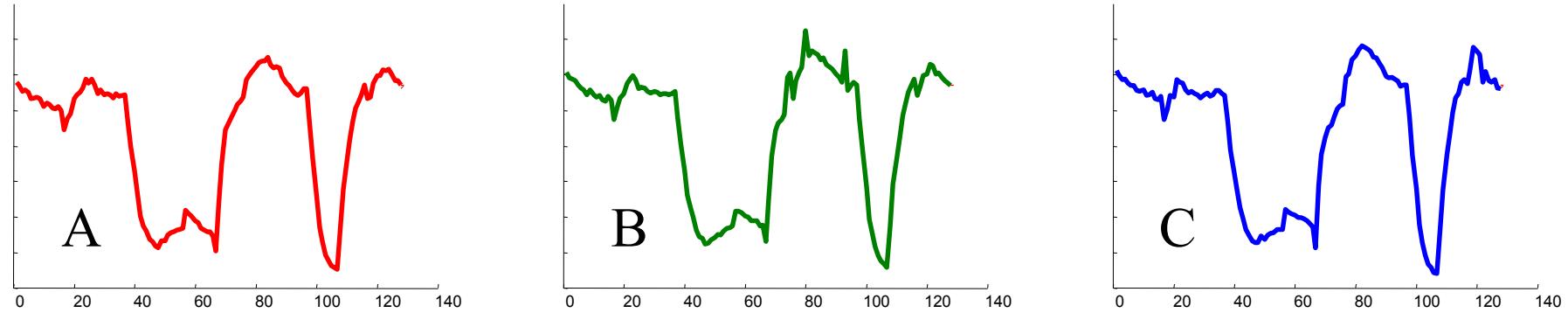
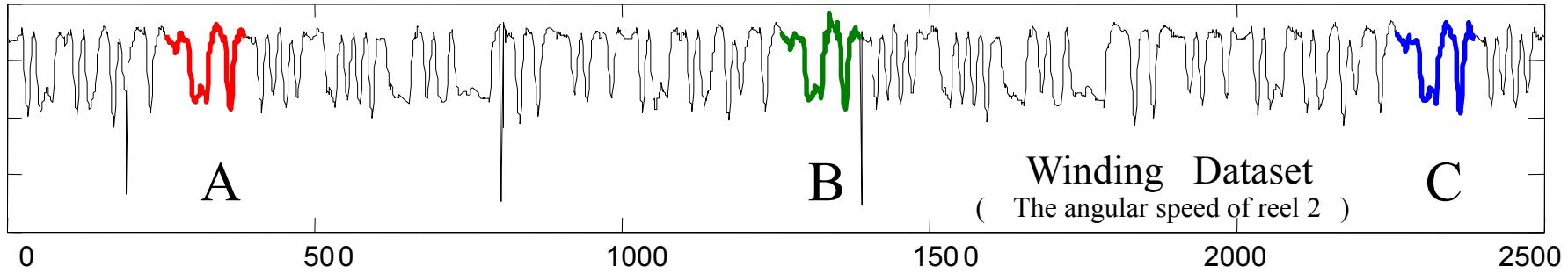


Are there any repeated patterns, of about this length — in the above time series?



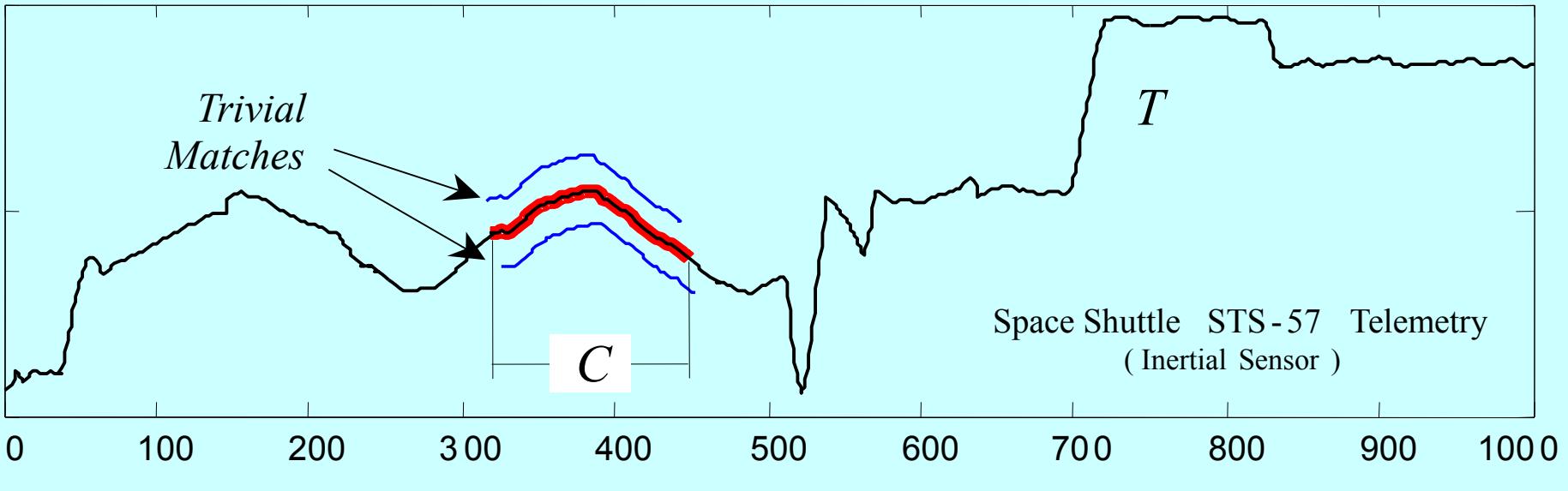
# Time Series Motif Discovery

(finding repeated patterns)



# Why Find Motifs?

- Mining **association rules** in time series requires the discovery of motifs. These are referred to as *primitive shapes* and *frequent patterns*.
- Several time series **classification algorithms** work by constructing typical prototypes of each class. These prototypes may be considered motifs.
- Many time series **anomaly/interestingness detection** algorithms essentially 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.
- In **robotics**, Oates et al., have introduced a method to allow an autonomous agent to generalize from a set of qualitatively different *experiences* gleaned from sensors. We see these “*experiences*” as motifs.
- In **medical data mining**, Caraca-Valente and Lopez-Chavarrias have introduced a method for characterizing a physiotherapy patient’s recovery based of the discovery of *similar patterns*. Once again, we see these “*similar patterns*” as motifs.
- **Animation and video capture...** (Tanaka and Uehara, Zordan and Celly)



**Definition 1. Match:** Given a positive real number  $R$  (called *range*) and a time series  $T$  containing a subsequence  $C$  beginning at position  $p$  and a subsequence  $M$  beginning at  $q$ , if  $D(C, M) \leq R$ , then  $M$  is called a *matching* subsequence of  $C$ .

**Definition 2. Trivial Match:** Given a time series  $T$ , containing a subsequence  $C$  beginning at position  $p$  and a matching subsequence  $M$  beginning at  $q$ , we say that  $M$  is a *trivial match* to  $C$  if either  $p = q$  or there does not exist a subsequence  $M'$  beginning at  $q'$  such that  $D(C, M') > R$ , and either  $q < q' < p$  or  $p < q' < q$ .

**Definition 3.  $K$ -Motif( $n, R$ ):** Given a time series  $T$ , a subsequence length  $n$  and a range  $R$ , the most significant motif in  $T$  (hereafter called the  $1$ -Motif( $n, R$ )) is the subsequence  $C_1$  that has highest count of non-trivial matches (ties are broken by choosing the motif whose matches have the lower variance). The  $K^{\text{th}}$  most significant motif in  $T$  (hereafter called the  $K$ -Motif( $n, R$ )) is the subsequence  $C_K$  that has the highest count of non-trivial matches, and satisfies  $D(C_K, C_i) > 2R$ , for all  $1 \leq i < K$ .

# OK, we can define motifs, but how do we find them?

The obvious brute force search algorithm is just too slow...

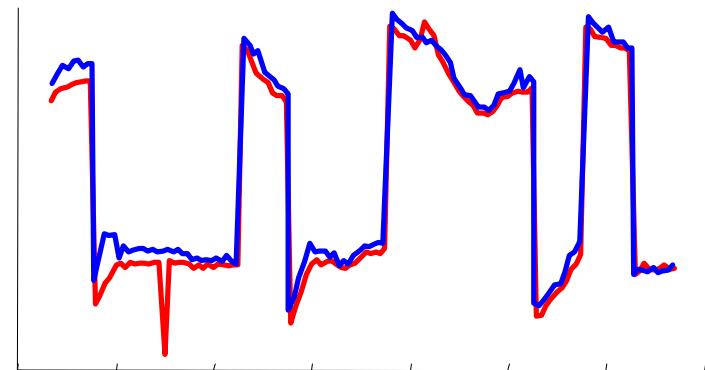
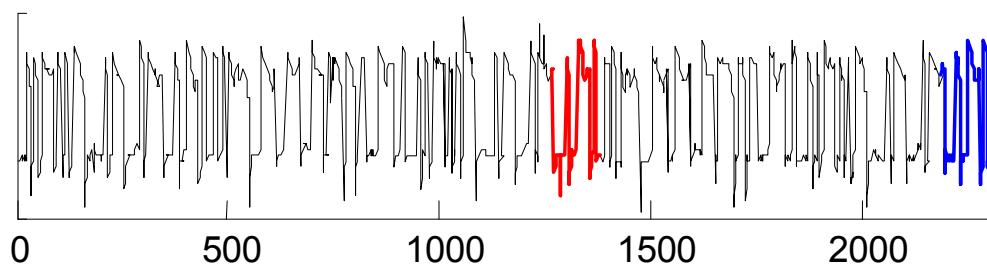
Many algorithms realized, main ones are based on *random projection*\* and SAX discretized representation of times series (useful to compute quick lower bound distances).



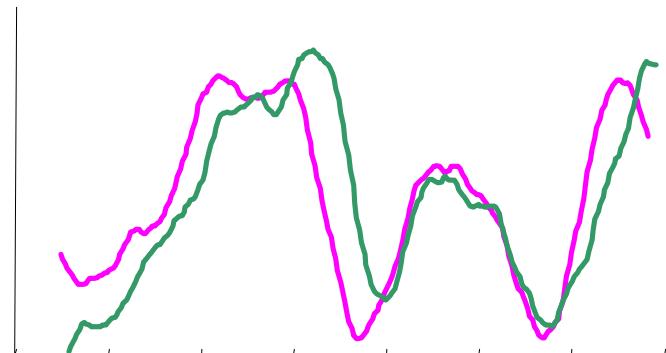
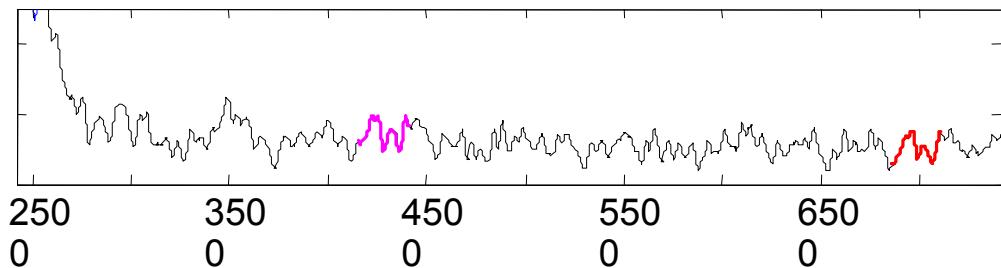
\* J Buhler and M Tompa. *Finding motifs using random projections*. In RECOMB'01. 2001.

# Some Examples of Real Motifs

Motor 1 (DC Current)



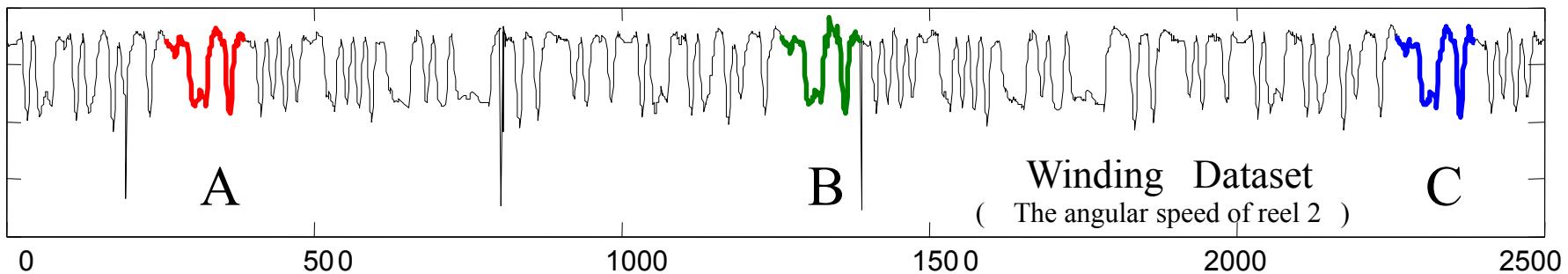
Astrophysics (Photon Count)



# Motifs Discovery Challenges

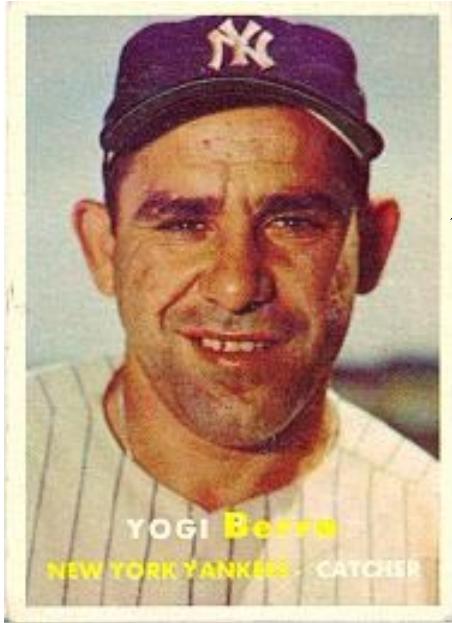
How can we find motifs...

- Without having to specify the length/other parameters
- In massive datasets
- While ignoring “background” motifs (ECG example)
- Under time warping, or uniform scaling
- While assessing their significance



Finding these 3 motifs requires about 6,250,000 calls to the Euclidean distance function

# Time Series Prediction



Yogi Berra  
1925 - 2015

Prediction is hard, especially about the future

There are two kinds of time series prediction

- **Black Box:** Predict tomorrow's electricity demand, given *only* the last ten years electricity demand.
- **White Box (side information):** Predict tomorrow's electricity demand, given the last ten years electricity demand *and* the weather report, *and* the fact that the world cup final is on and...

# Black Box Time Series Prediction

- Very difficult task, at least in longer-term prediction
- Several solutions are claiming great results, yet they are mostly still unreliable:
  - A paper in SIGMOD 04 claims to be able to get better than 60% accuracy on black box prediction of financial data (random guessing should give about 50%). The authors later agreed to test blind on a external and again got more than 60% -- though it was completely random data (quantum-mechanical random walk)...
  - A paper in SIGKDD in 1998 did black box prediction using association rules, more than twelve papers extended the work... but then it was proved that the approach *could* not work\*!

# End of Time series module

