Mining time series data

Mining Massive Datasets

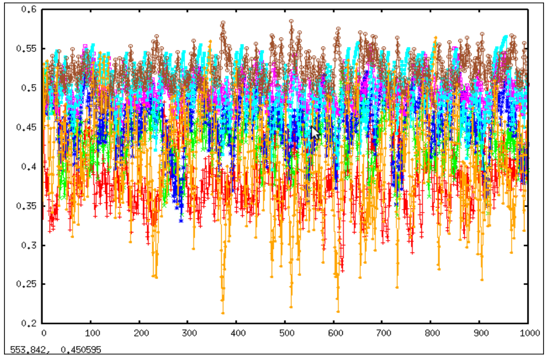
Carlos Castillo

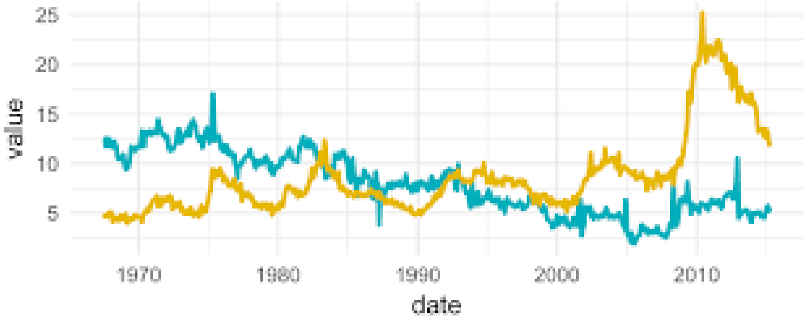
Topic 11

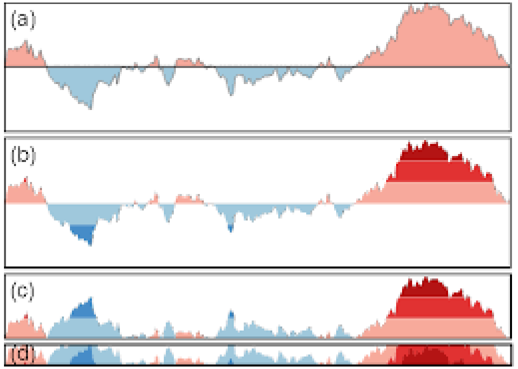
Universitat Pompeu Fabra Barcelona

Sources

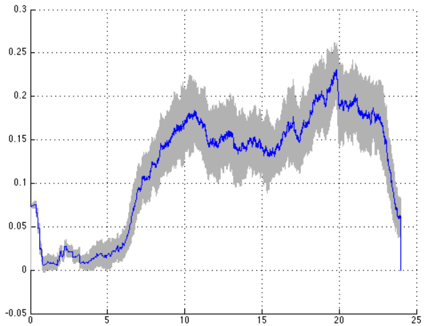
- Data Mining, The Textbook (2015) by Charu Aggarwal (chapter 14)
- Introduction to Time Series Mining (2006) tutorial by Keogh Eamonn [alt. link]
- Time Series Data Mining (2006) slides by Hung Son Nguyen

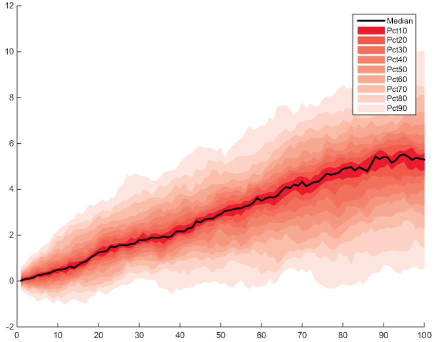


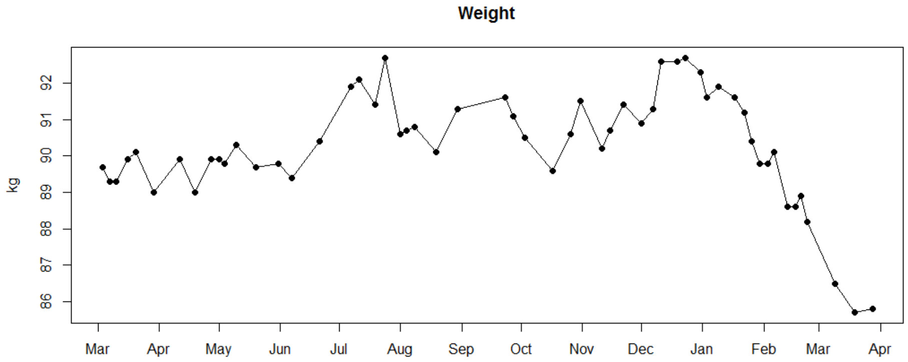




Why do we mine time series? Examples

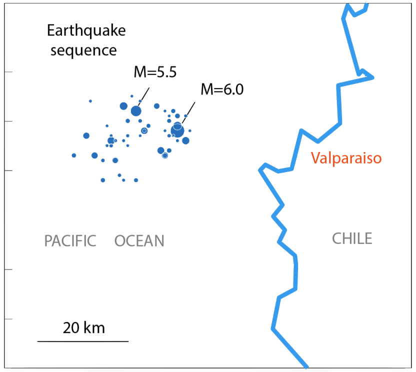


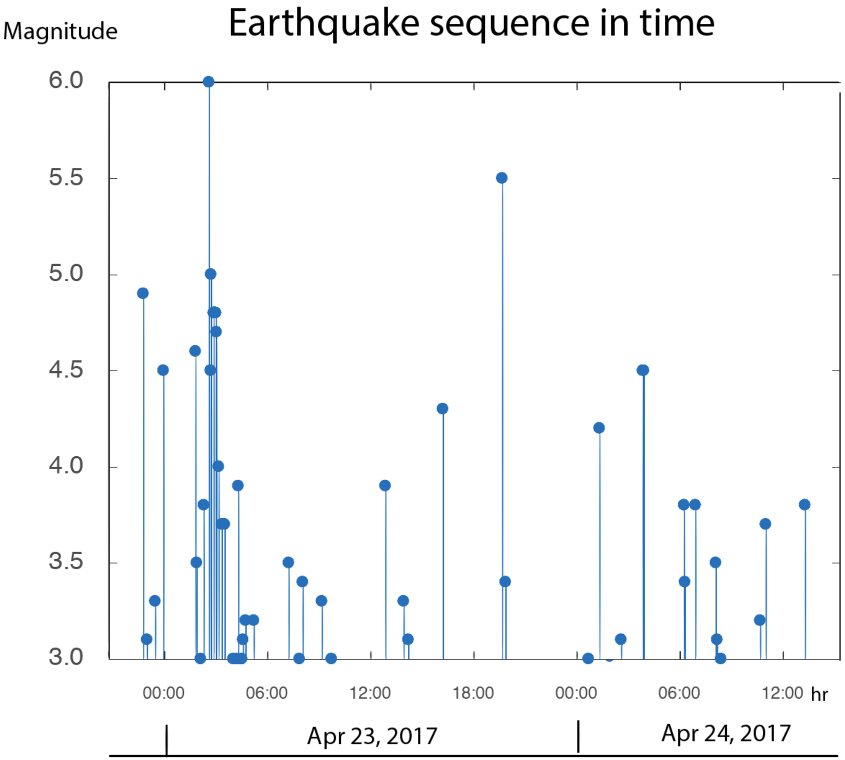




Seismic data

- Observations = earthquakes
- Goal: characterize when peeks occur



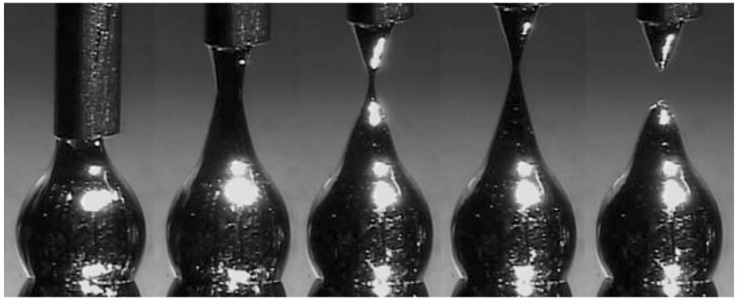


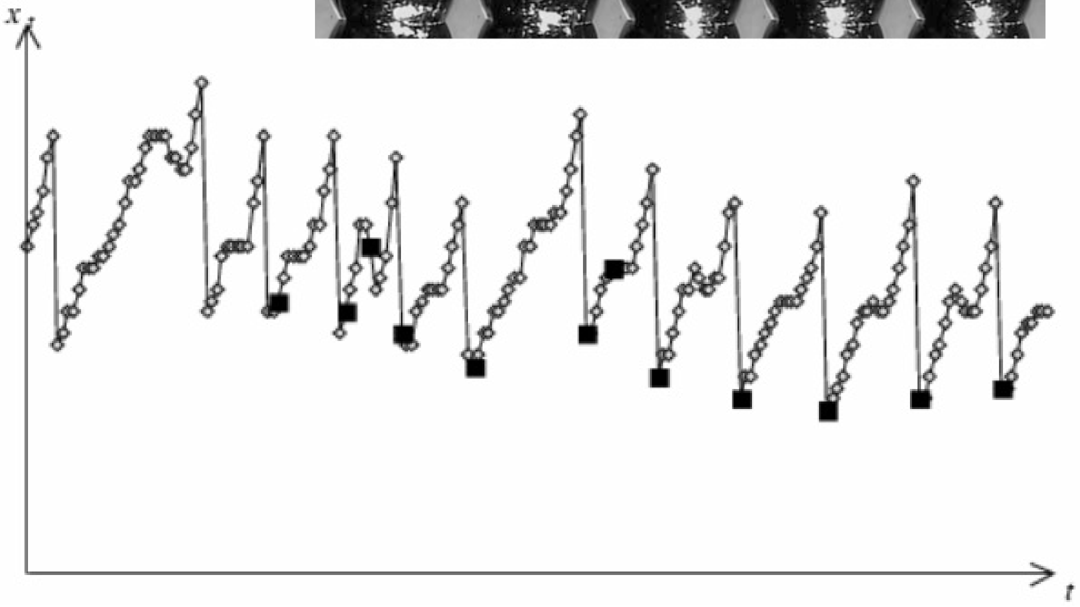
Liquid metal droplets

◇ = length of hot metal droplet

■ = droplet release (chaotic, noisy)

Goal: prediction of release





Stock prices

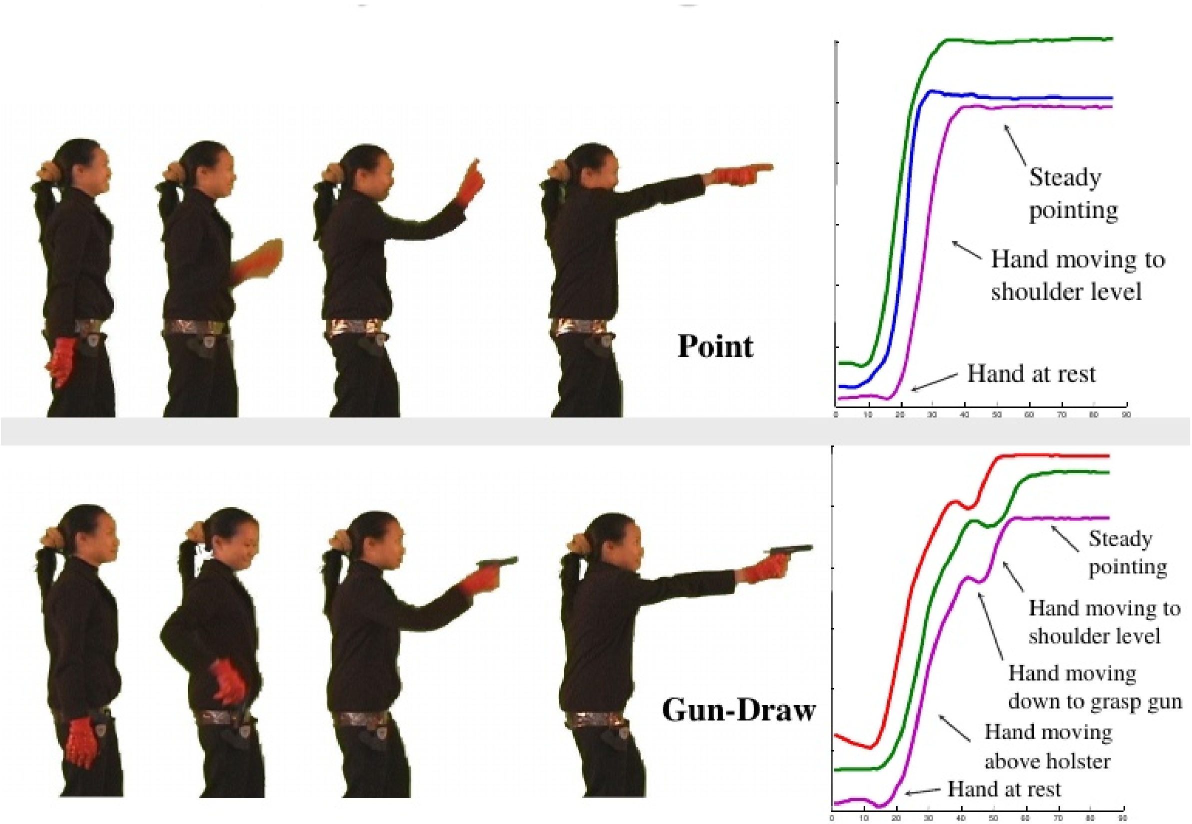
Price

Volume traded

Goal: find hidden patterns providing an advantage

Video data / gestures

- Series of angles of articulations in the body
- Temporal patterns can reveal gestures



Applications

- Clustering
- Classification
- Motif discovery
- Event detection
- …

1) All require a reasonable definition of the **similarity** between two time series

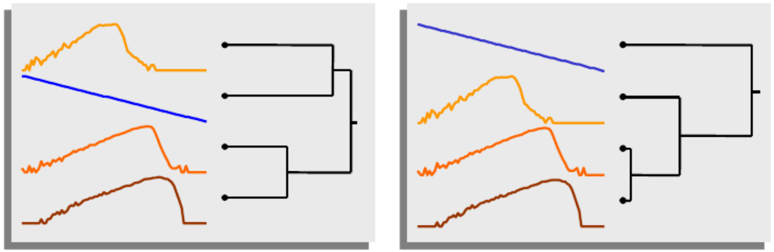
2) All can be done in **real-time** or **retrospectively**

Context vs Behavior

- **Contextual attribute(s)**
  - $x(i) = t_i$ = timestamp is the typical one
  - Sometimes other attributes providing context
- **Behavioral attribute(s)**
  - $y^j(i)$ = temperature, angle, price, sensor reading, …
    $j \in 1 \ldots d$

What are the difficulties?

- High sampling rate of many series over extended periods of time means …
  - Tons of data
  - Things are bound to fail at several points (missing data, noisy data)
- Subjectivity



Introduction to Time Series Mining (2006) tutorial by Keogh Eamonn [alt. link]

Preparing a time series

Notation: multivariate time series

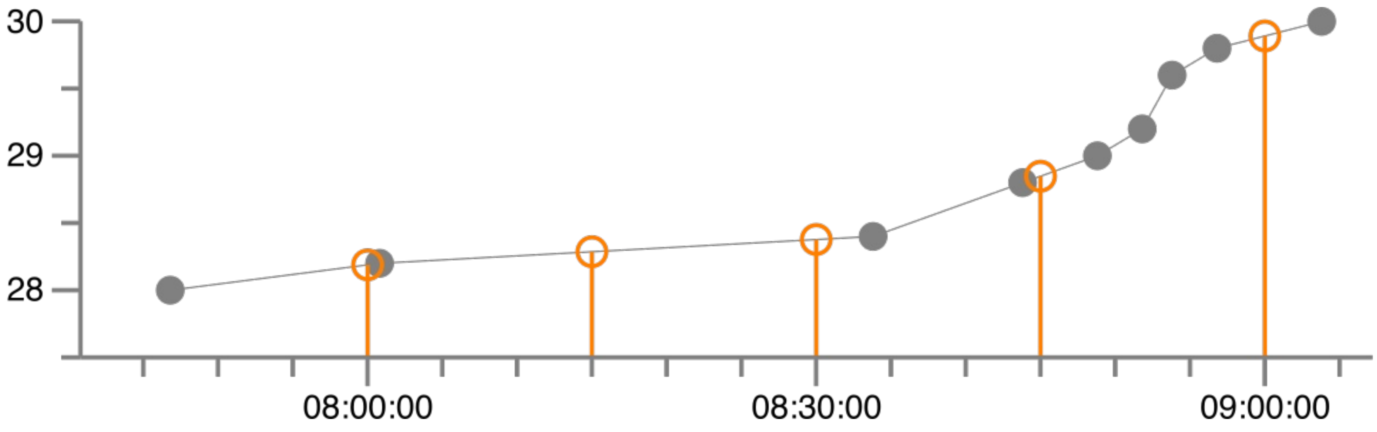
- Length n, timestamps $t_1, t_2, \ldots, t_n$
- Values at time t_i : $(y_i^1, y_i^2, \ldots, y_i^d)$
- If series is univariate we drop the superscript

Missing values: linear interpolation

- Let $t_i < t_x < t_j$

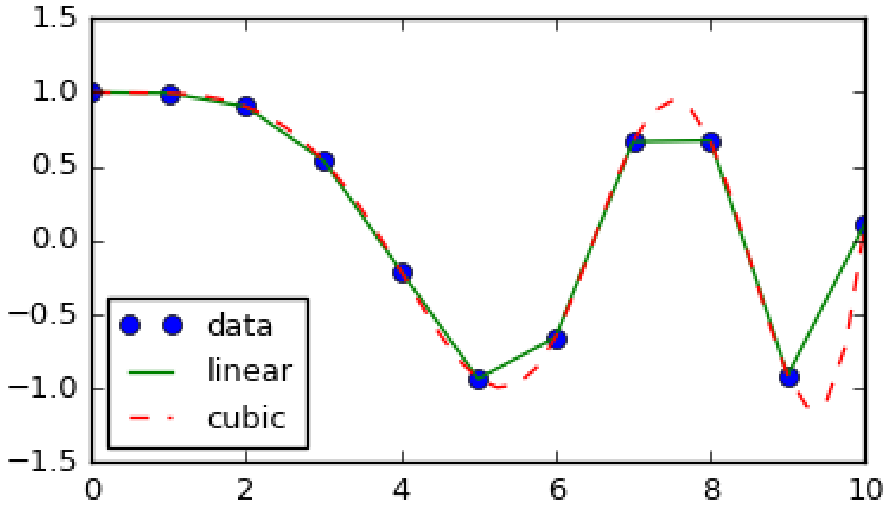
$$y_x = y_i + \left(\frac{t_x - t_i}{t_j - t_i}\right) \cdot (y_j - y_i)$$

- Example: make an irregular series regular



Missing values: splines

Cubic polynomials between y_i, y_{i+1} that have the same slope at those points as the original curve.

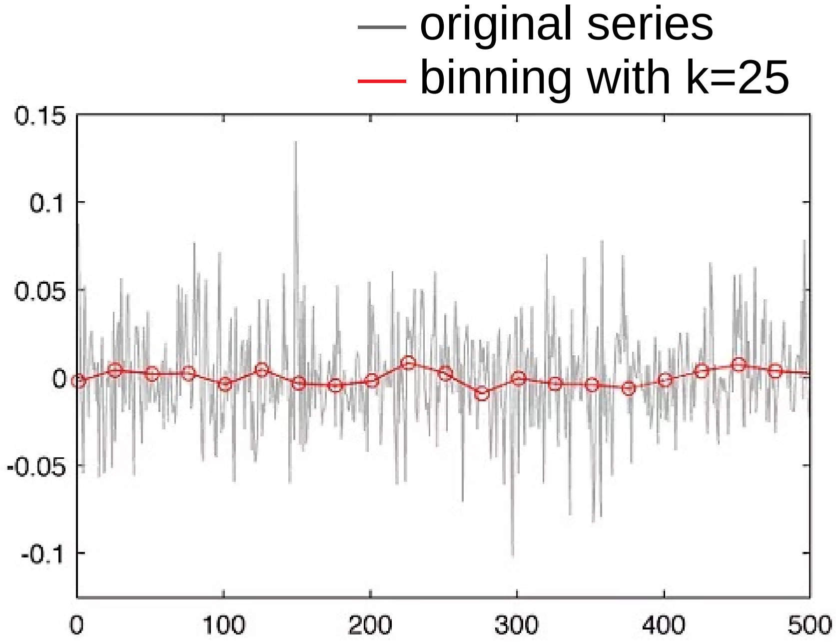


http://lagrange.univ-lyon1.fr/docs/scipy/0.17.1/tutorial/interpolate.html

Noise removal: binning

- Replace series by average of values in bins (subsequences) of length k

$$y'_{i+1} = \frac{1}{k}\sum_{r=1}^{k} y_{i\cdot k+r}$$



http://www.quantatrisk.com/2013/03/22/rebinning-of-financial-time-series/

Noise removal:
moving average smoothing

- Equivalent to overlapping bins

$$y'_i = \frac{1}{k} \sum_{r=1}^{k} y_{i-r+1}$$

- Larger k leads to smoother series, but losses more information
- Use smaller k for first k-1 items

https://www.fidelity.com/learning-center/trading-investing/technical-analysis/technical-indicator-guide/sma

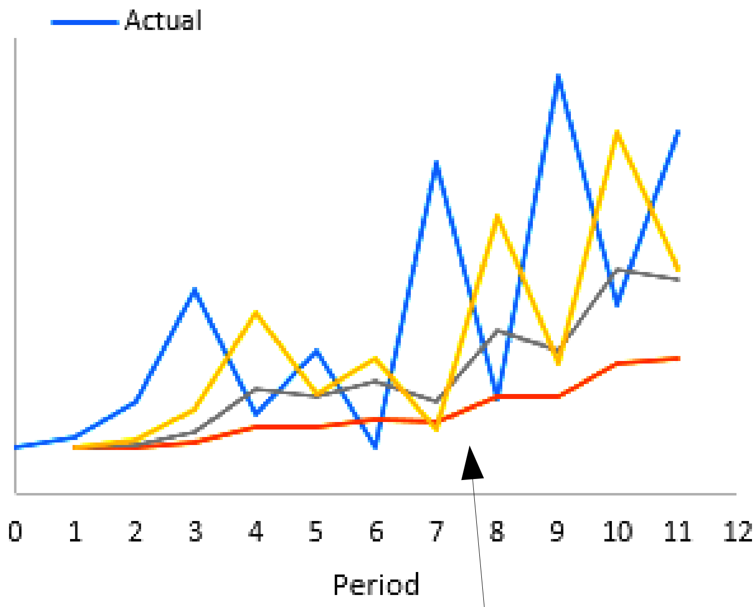
Noise removal: exponential smoothing

- Combine previously smoothed point with current point

$$y'_i = \alpha \cdot y_i + (1 - \alpha) \cdot y'_{i-1}$$

- Recursively substituting

$$y'_i = (1 - \alpha)^i \cdot y'_0 + \alpha \sum_{j=1}^{i} y_j \cdot (1 - \alpha)^{i-j}$$



Which y' has the larger alpha?

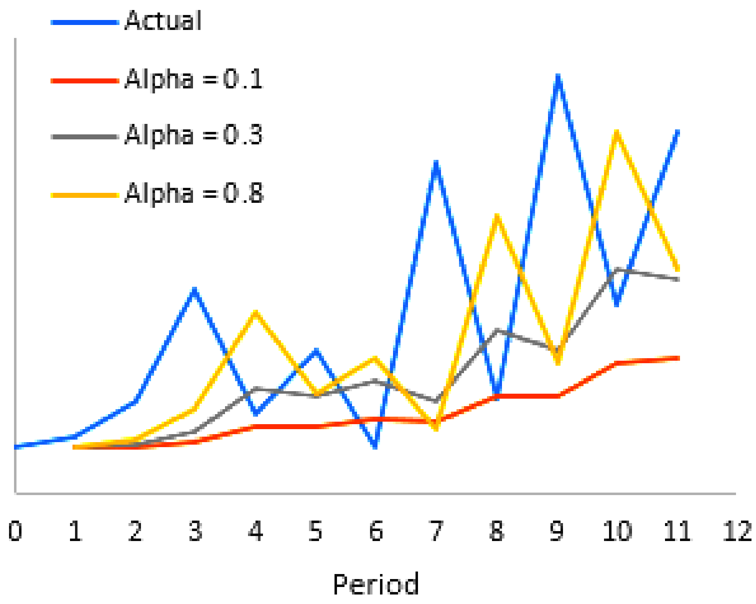
Noise removal: exponential smoothing

- Combine previously smoothed point with current point

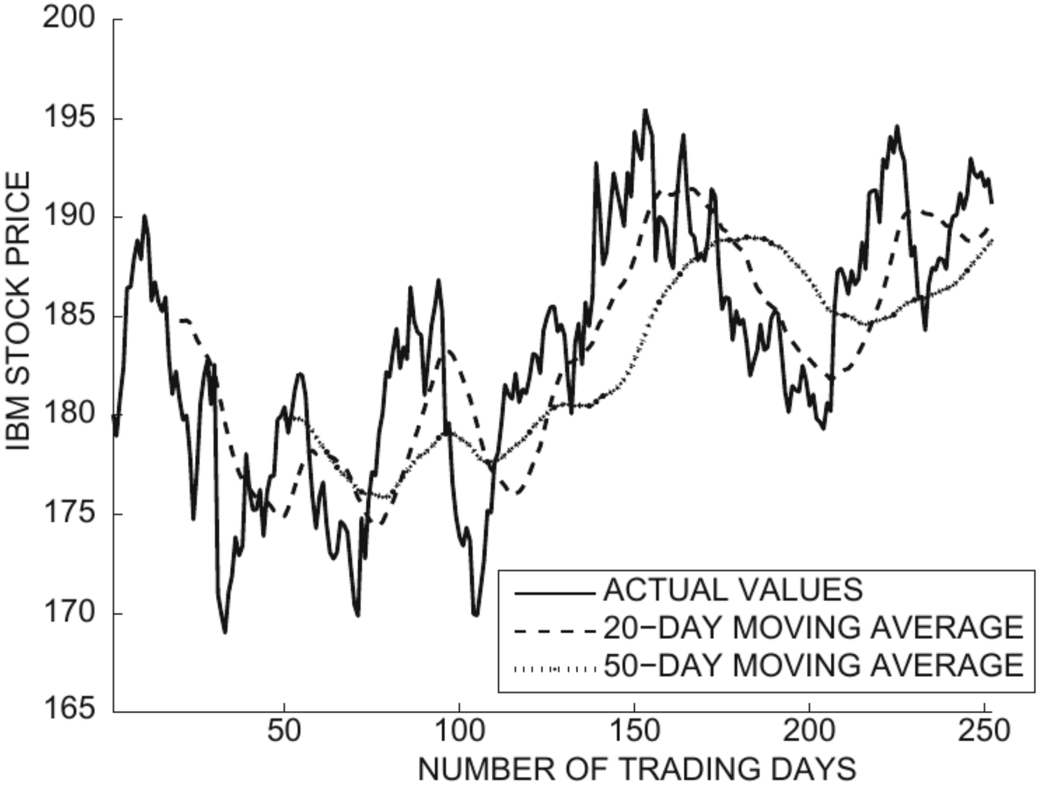
$$y'_i = \alpha \cdot y_i + (1-\alpha) \cdot y'_{i-1}$$

- Recursively substituting

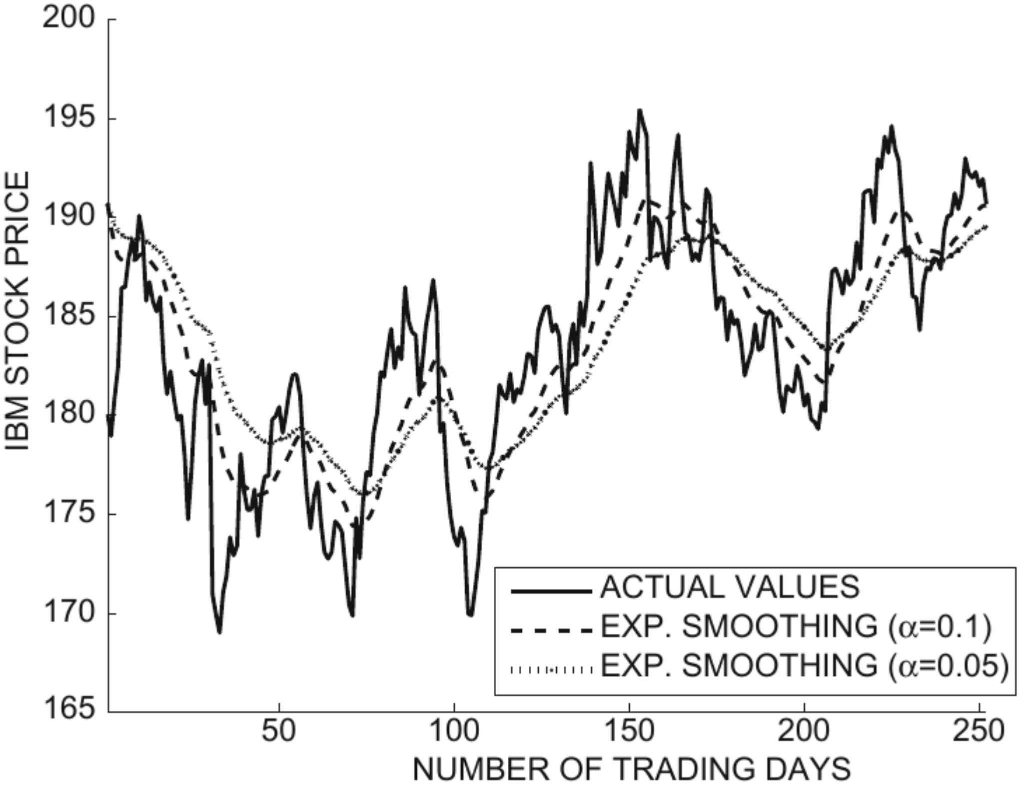
$$y'_i = (1-\alpha)^i \cdot y'_0 + \alpha \sum_{j=1}^{i} y_j \cdot (1-\alpha)^{i-j}$$



Moving average vs exponential smoothing



(a) Moving average smoothing



(b) Exponential smoothing

Try it!

- Given the following series:

t	1	2	3	4	5	6	7	8	9	10
y_t	2	4	12	3	1	-2	0	15	3	2
y_t'										
y_t''										

- y_t': moving average with k=3
- y_t'': exponential average with alpha=0.5

Using Euclidean distance on time series

Euclidean distance for time series

- Euclidean distance between series y and z

$$d(y,z) = \sqrt{\sum_{i=1}^{n} (y_i - z_i)^2}$$

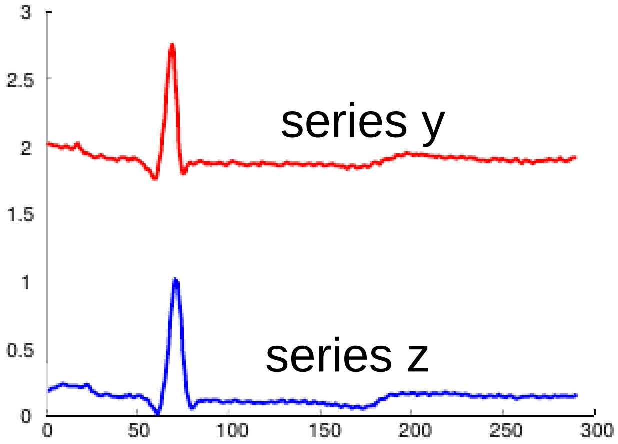
python

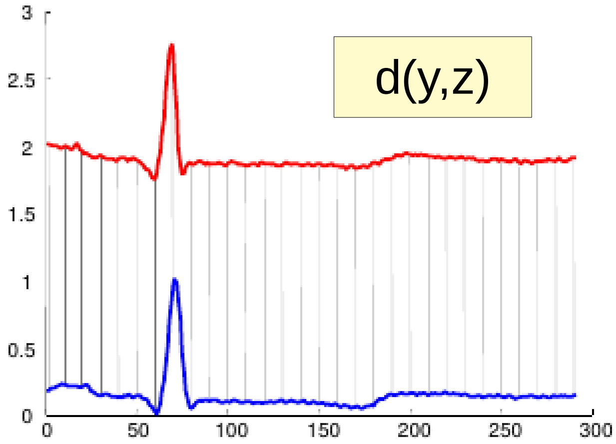
```
numpy.linalg.norm(y-z)
```

- Sensitive to **noise** (see previous slides on how to fix this)
- Sensitive to different **offsets**, **amplitudes**, and **trends**

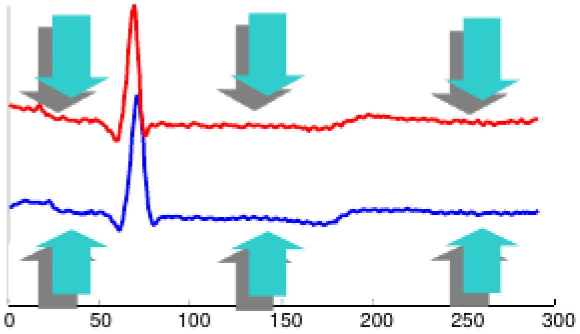
Slides on this section from: Introduction to Time Series Mining (2006) tutorial by Keogh Eamonn [alt. link]

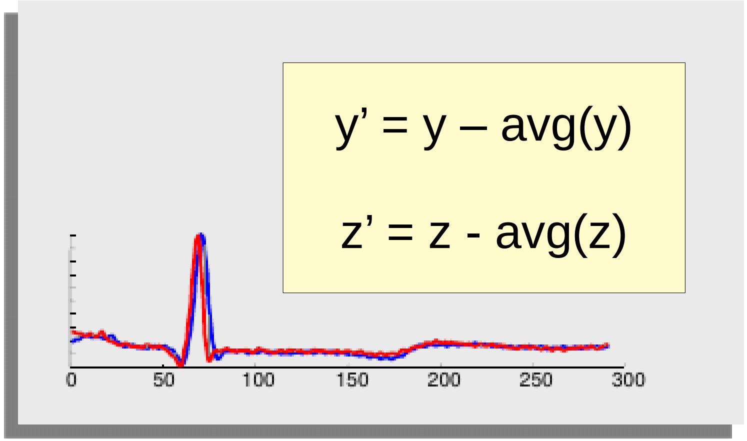
Offset translation: subtract the mean





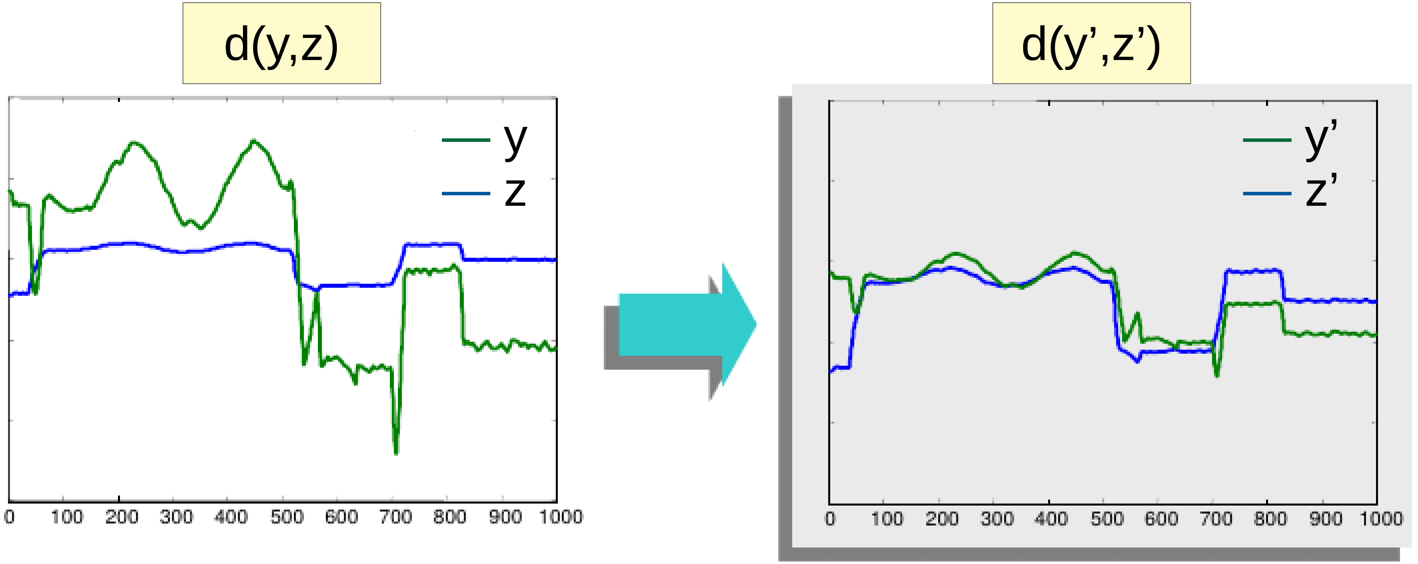
- Series look different





- Series look similar

Amplitude scaling: normalize



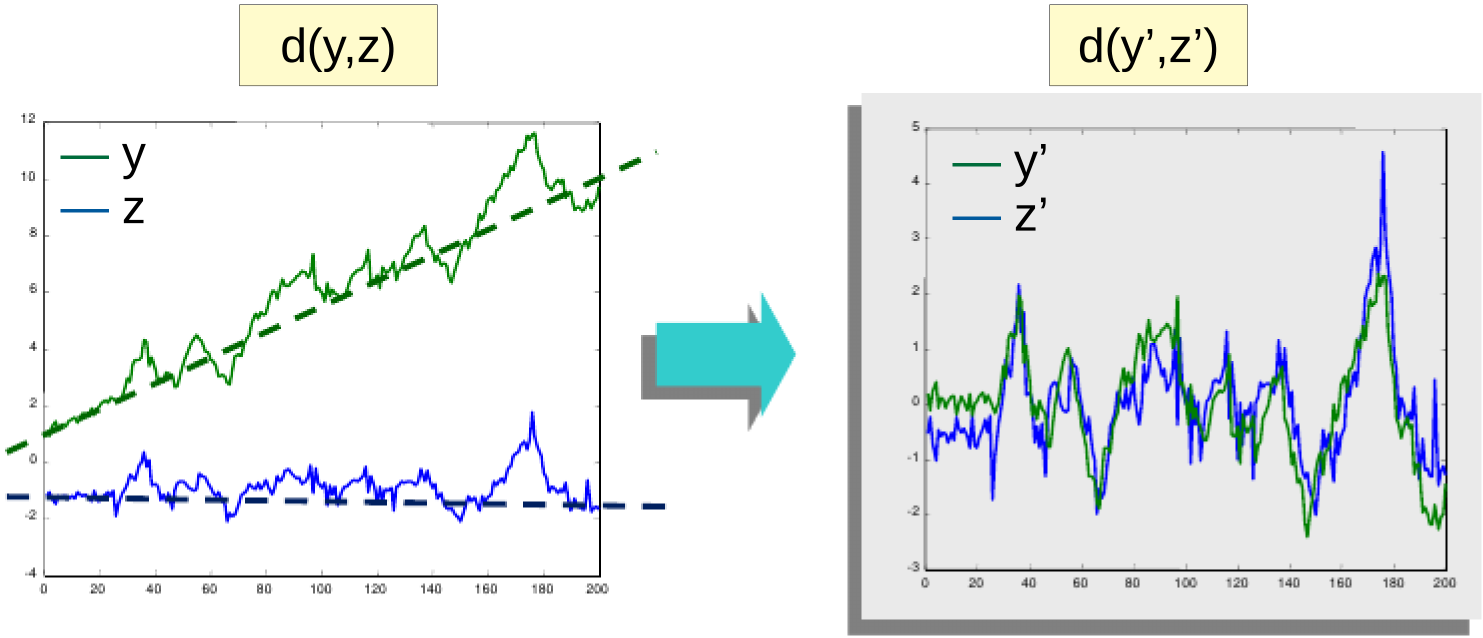
- Standardization

$$y_i' = \frac{y_i - \mathrm{avg}(y)}{\mathrm{std}(y)}$$

- Range-based normalization

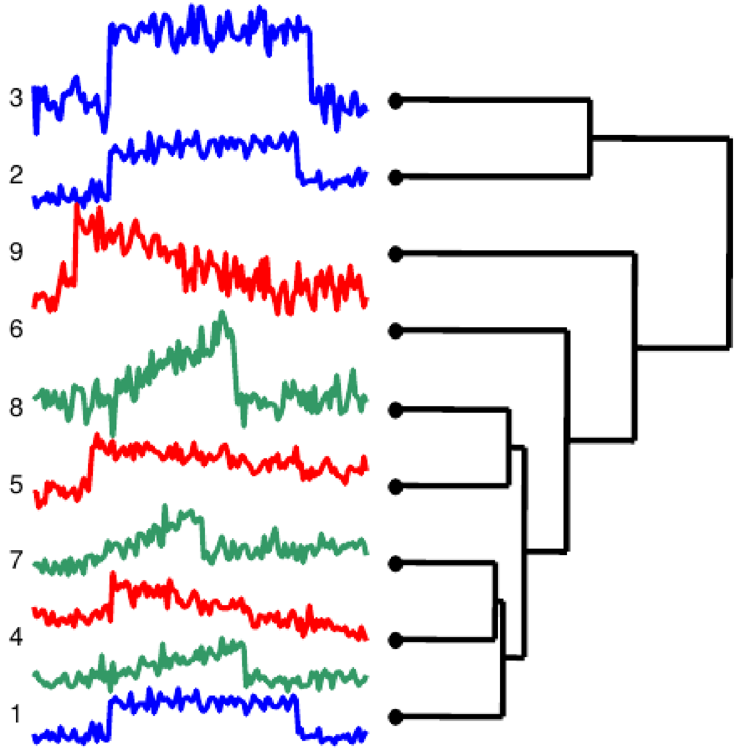
$$y_i' = \frac{y_i - \min(y)}{\max(y) - \min(y)}$$

Trend removal: remove linear trend

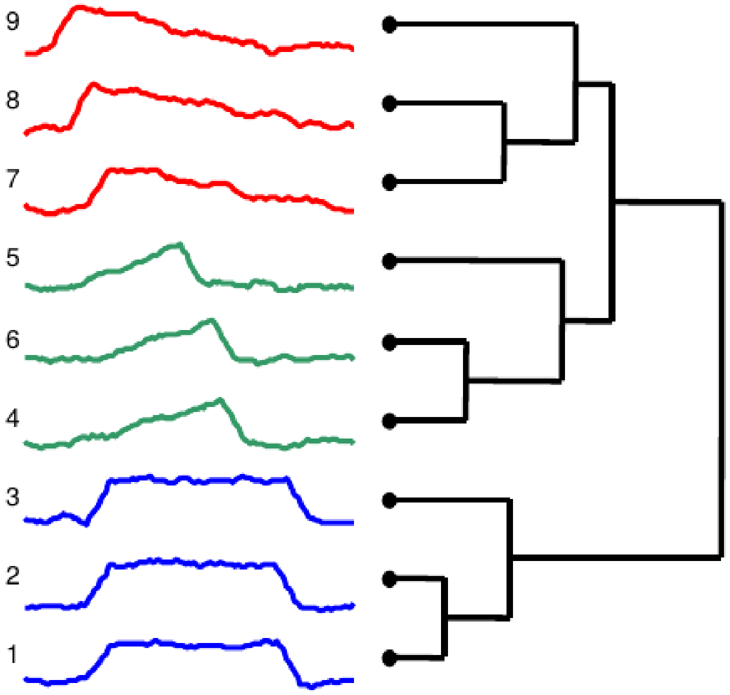


1. Find best straight line fitting data
2. Subtract that line from the data

Example: clustering of time series after using smoothing, offset translation, amplitude scaling, and trend removal



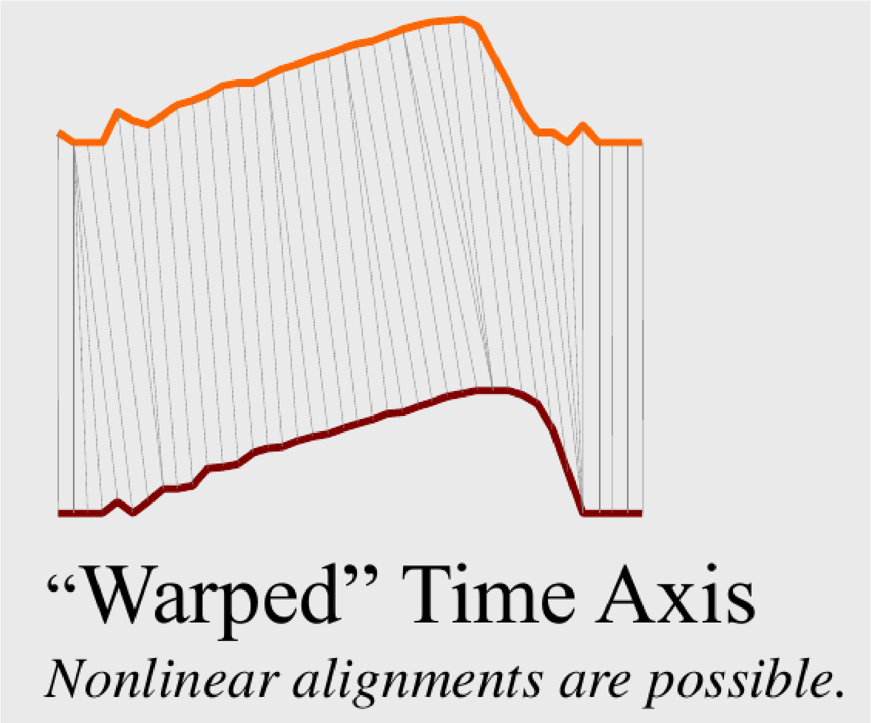
Clustering using euclidean distance on original series



Clustering using euclidean distance on processed series

Dynamic time warping

Dynamic time warping



Dynamic time warping example

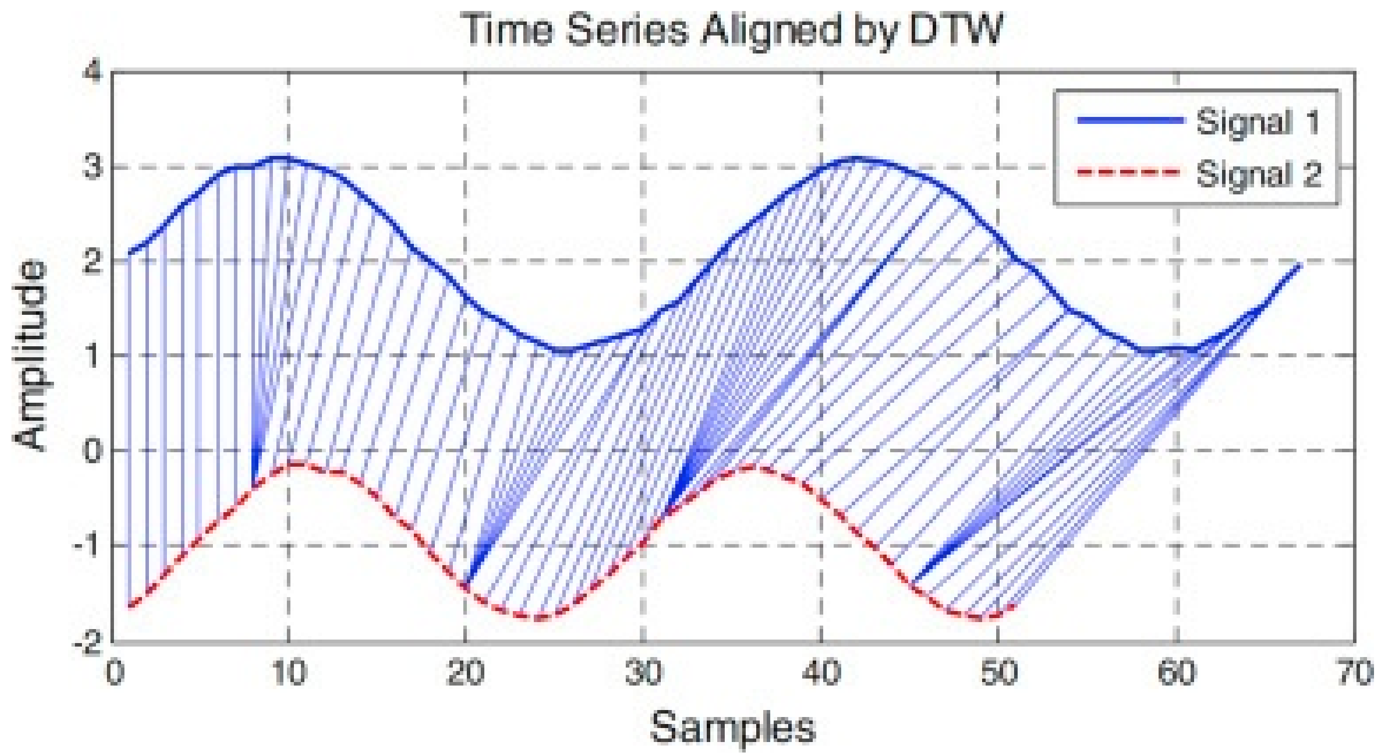
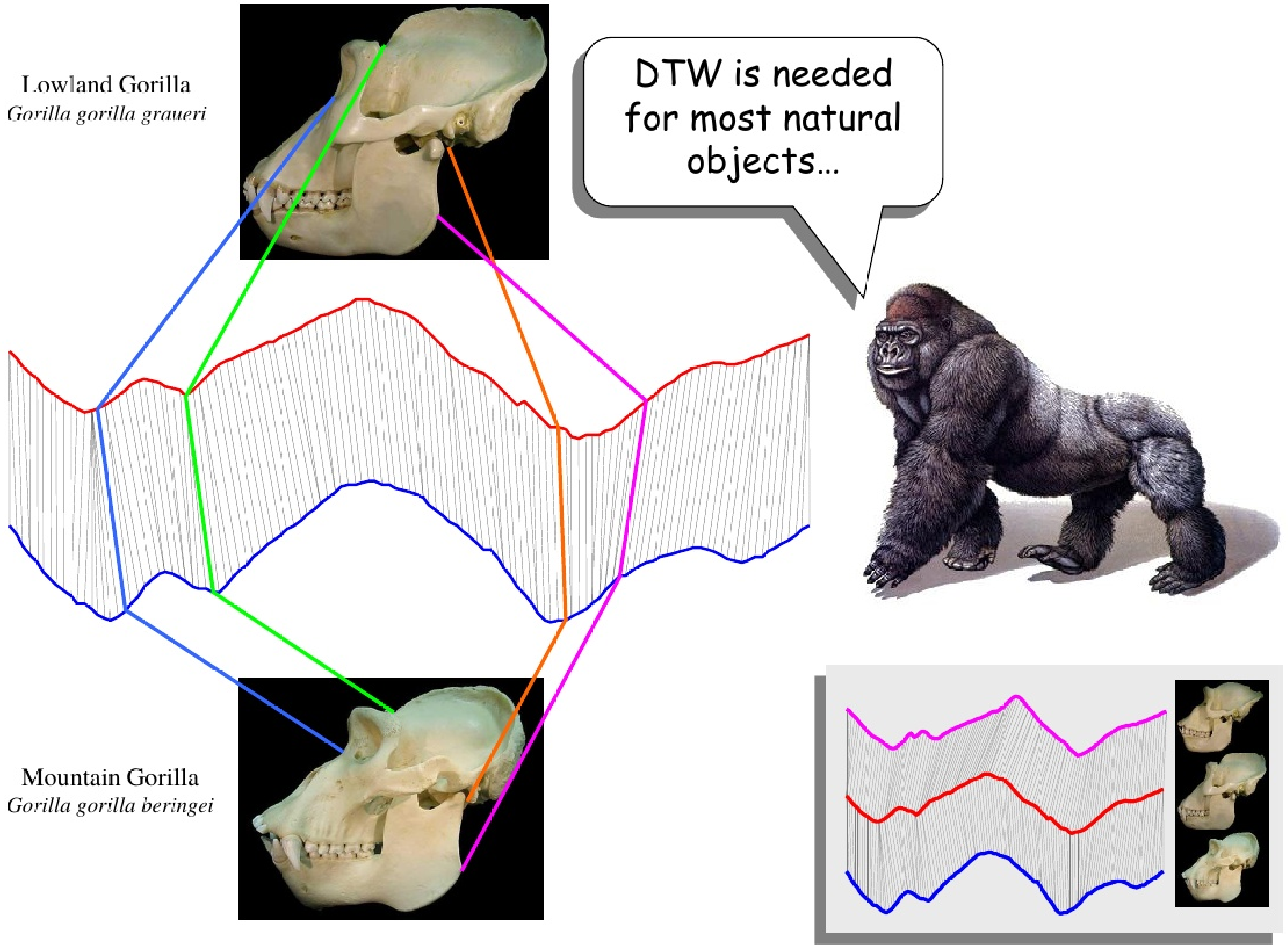
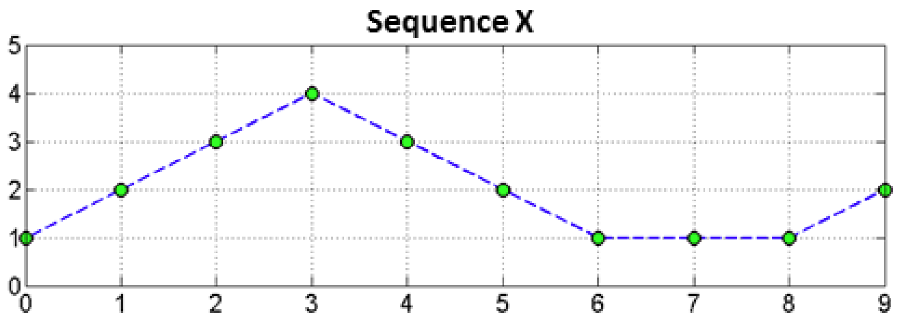


Image credits: Lu et al. 2016

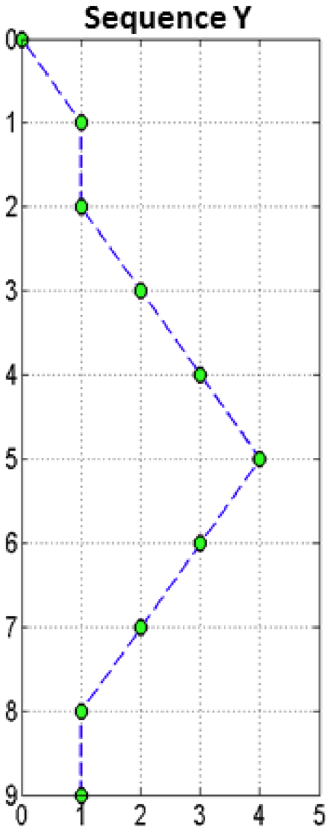

Lowland Gorilla
Gorilla gorilla graueri
DTW is needed for most natural objects…
Mountain Gorilla
Gorilla gorilla beringei

Computing DTW(X,Y)



1) Create a matrix M of size |X|×|Y|

2) Fill-in the matrix using dynamic programming



1	5	14	30
1	2	6	15

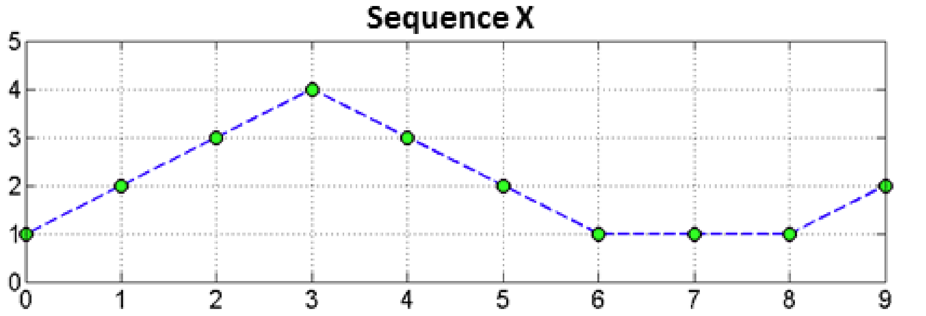
$$
\begin{aligned}
M(i,j) = \quad & d(y_i, x_j) + \\
& \min\{M(i-1, j-1), \\
& \quad M(i-1, j), \\
& \quad M(i, j-1)\}
\end{aligned}
$$

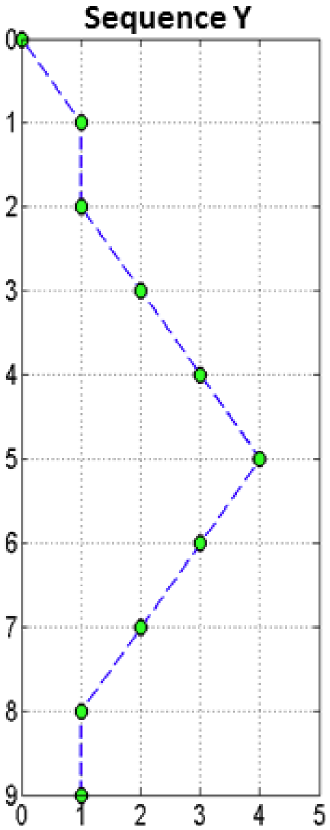
[Source]

Computing DTW(X,Y) (cont.)

1) Create a matrix M of size |X|×|Y|
2) Fill-in the matrix using dynamic programming
3) Find lighter path
4) Cell (a,b) in path ⇒ points a,b should be aligned

[Source]





1	5	14	30	39	43	44	45	46	50
1	2	6	15	19	20	20	20	20	21
1	2	6	15	19	20	20	20	20	21
2	1	2	6	7	7	8	9	10	10
6	2	1	2	2	3	7	11	13	11
15	6	2	1	2	6	12	16	20	15
19	7	2	2	1	2	6	10	14	15
20	7	3	6	2	1	2	3	4	4
20	8	7	12	6	2	1	1	1	2
20	9	11	16	10	3	1	1	1	2

Try it!

- Compute the DTW for these two series

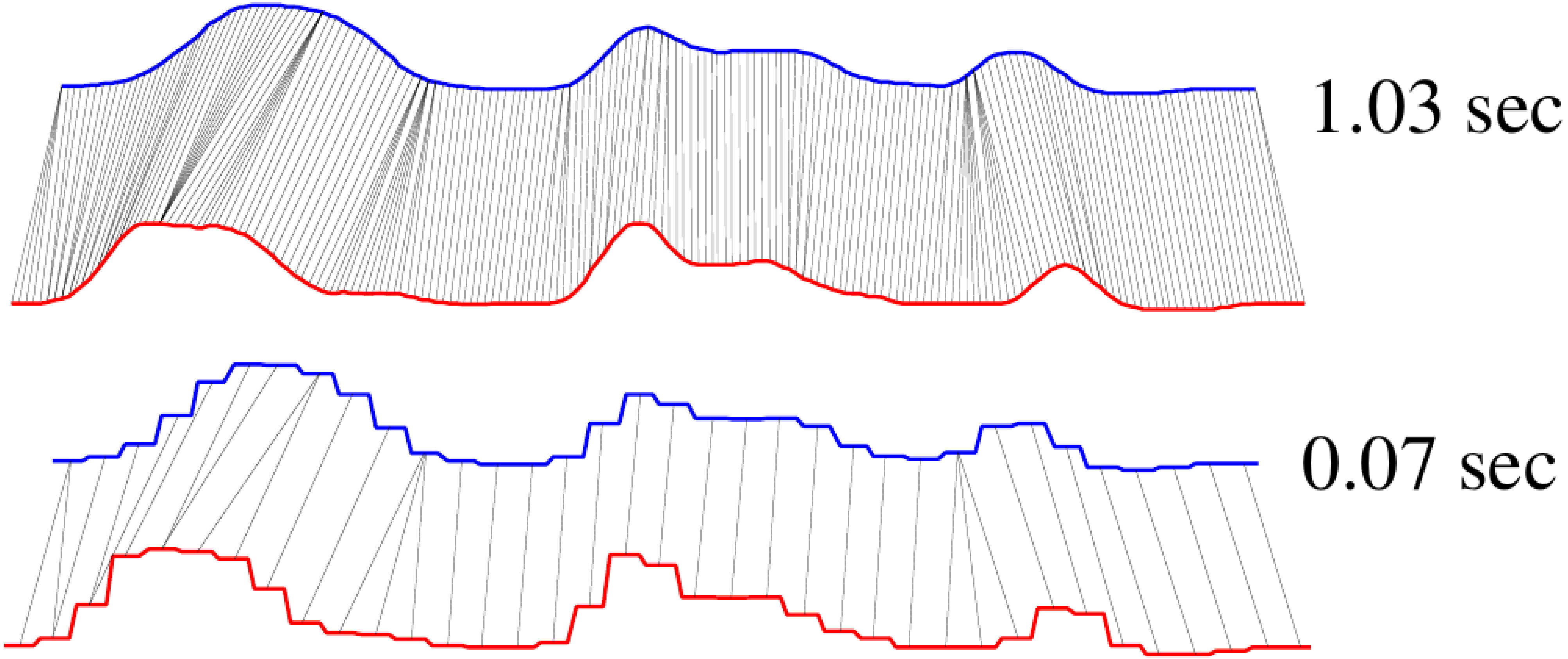
t	1	2	3	4	5	6
Y_t	2	5	2	5	3	--
X_t	0	3	6	0	6	1

Ko et al. 2008

Faster DTW through size reduction

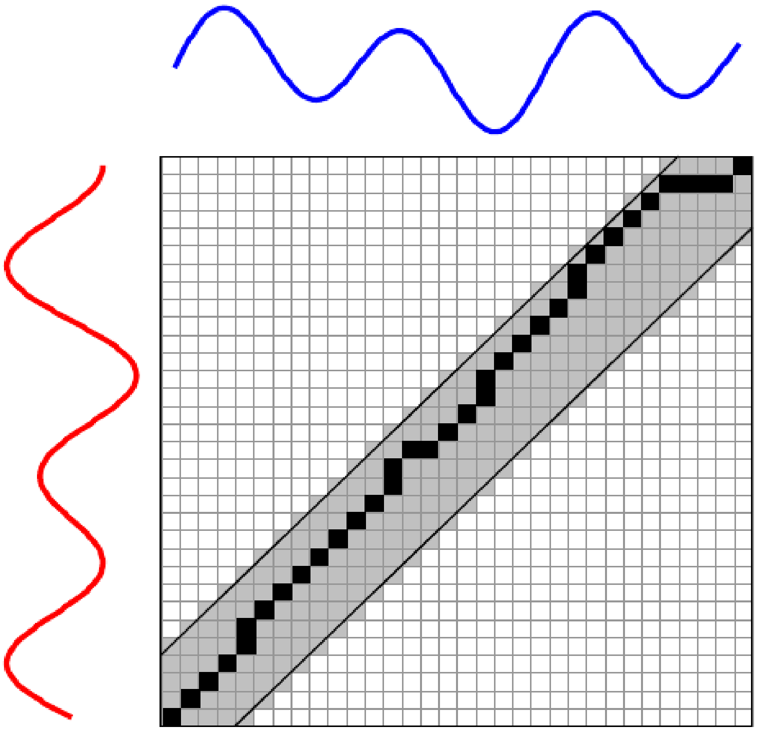
- How to avoid having a large matrix?
- Use less points
  - Sub-sample from original series
  - Bin the original series
- If sampling was done, after doing DTW:
  - Interpolate warpings for intermediate points

Example: faster DTW through sub-sampling

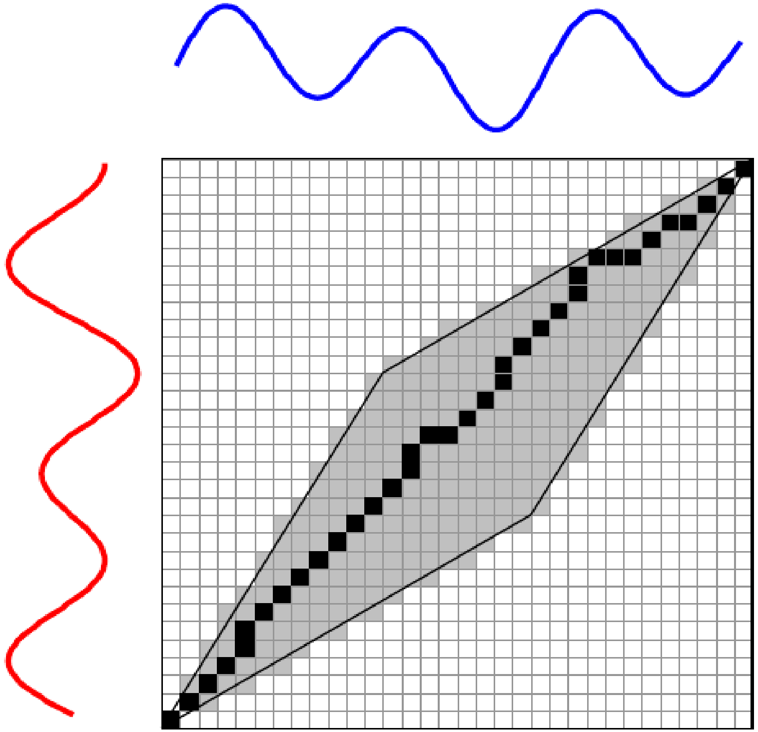


How to avoid pathological warpings?

Assume original series cannot be so far apart from each other, using domain knowledge



Sakoe-Chiba Band



Itakura Parallelogram

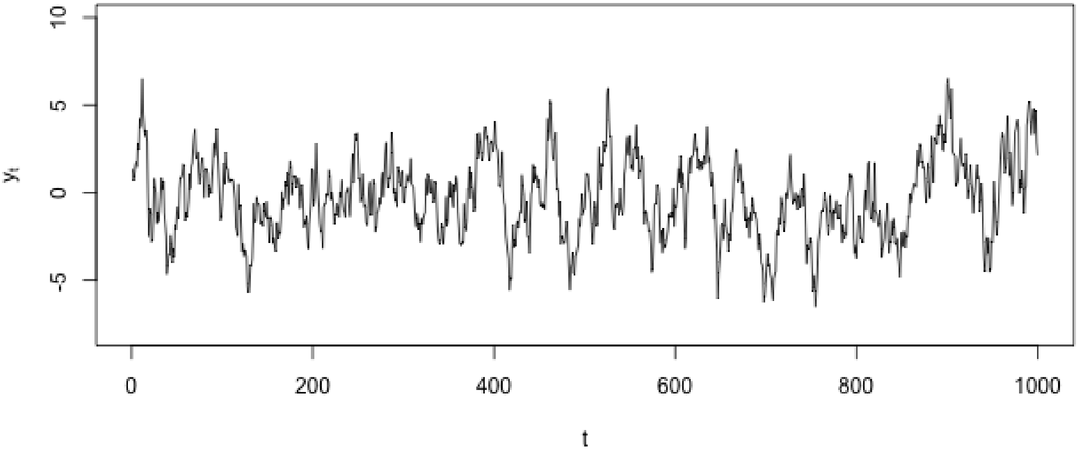
Forecasting

(AR, MA, ARMA, ARIMA, ...)

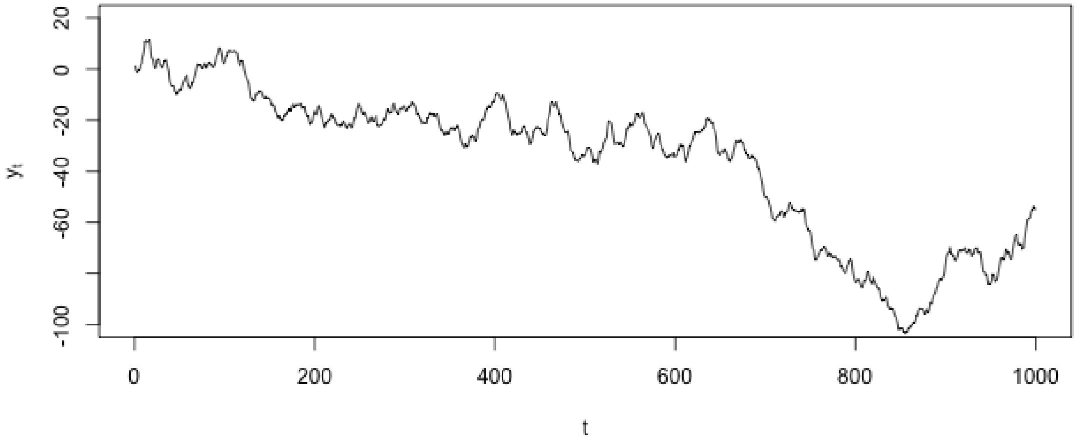
Stationary vs **Non-Stationary** processes

- **Stationary process**
  - Parameters do not change over time
  - E.g., *White noise* has zero mean, fixed variance, and zero covariance between y_t and y_{t+L} for any lag L
- **Non-stationary process**
  - Parameters change over time
  - E.g., price of oil, height of a child, glucose level of a patient, ...

Stationary process



Non-stationary process



Strictly stationary time series

A **strictly stationary time series** is one in which the distribution of values in any time interval $[a,b]$ is identical to that in $[a+L, b+L]$ for any value of time shift (lag) L

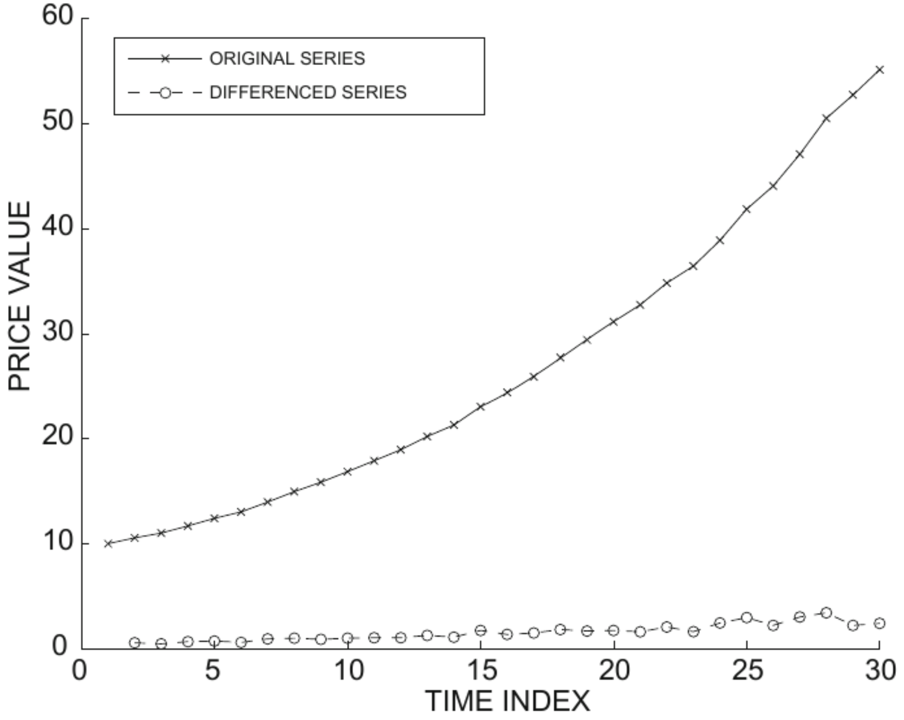
- In this case, current parameters (e.g., mean) are good predictors of future parameters

Differencing

First order differencing

$$y_i' = y_i - y_{i-1}$$

In this first example, if the original series is superlinear, the differenced series is stationary or non-stationary?

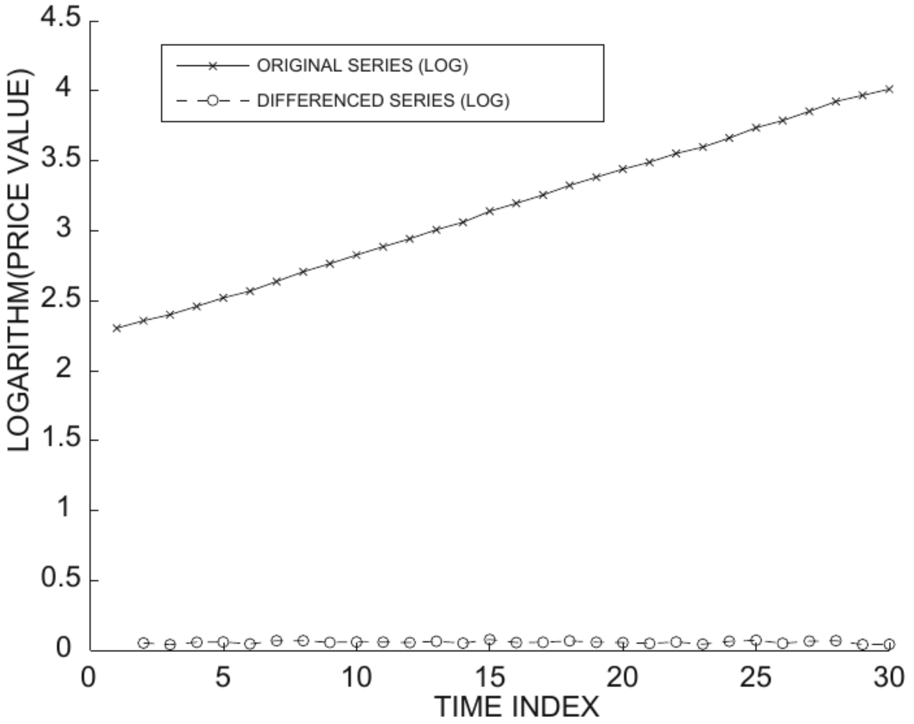


Differencing (cont.)

First order differencing

$$y_i' = y_i - y_{i-1}$$

In this second example, where the series is linear, the differenced series is stationary or non-stationary?



Other differencing operations

- Second-order differencing

$$y_i'' = y_i' - y_{i-1}'$$
$$= y_i - 2 \cdot y_{i-1} + y_{i-2}$$

- Seasonal differencing (m = 24 hours, 7 days, …)

$$y_i' = y_i - y_{i-m}$$

If you find a differencing that yields a stationary series, the forecasting problem is basically solved.

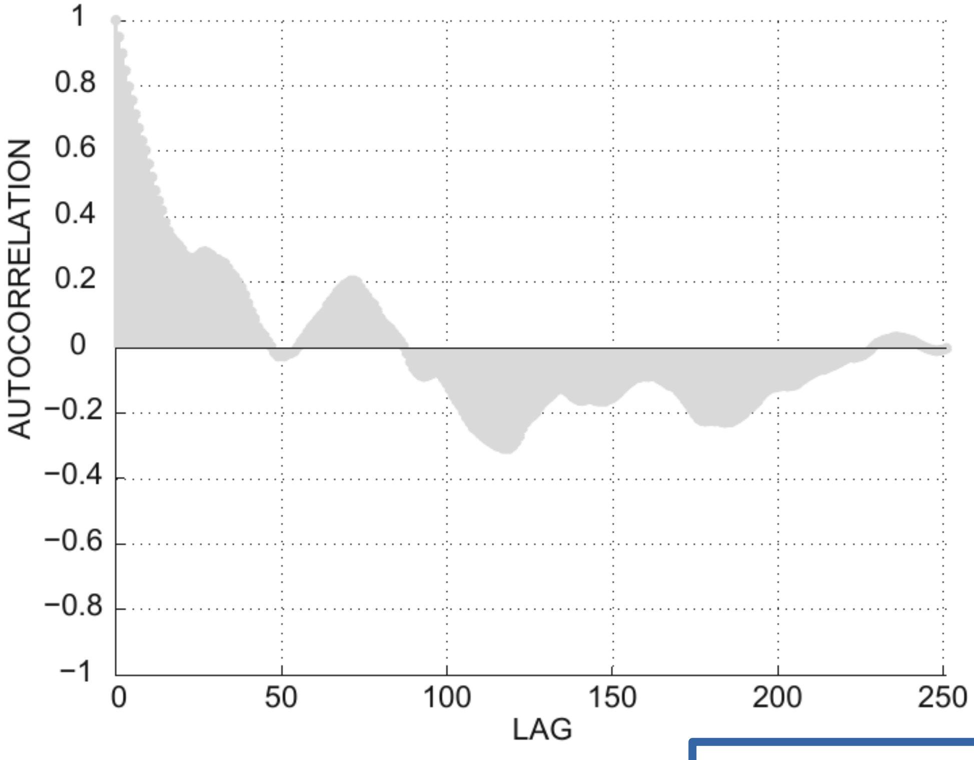
Autoregressive model **AR(p)**

$$\text{Autocorrelation}(L) = \frac{\text{Covariance}_t(y_t, y_{t+L})}{\text{Variance}_t(y_t)}$$

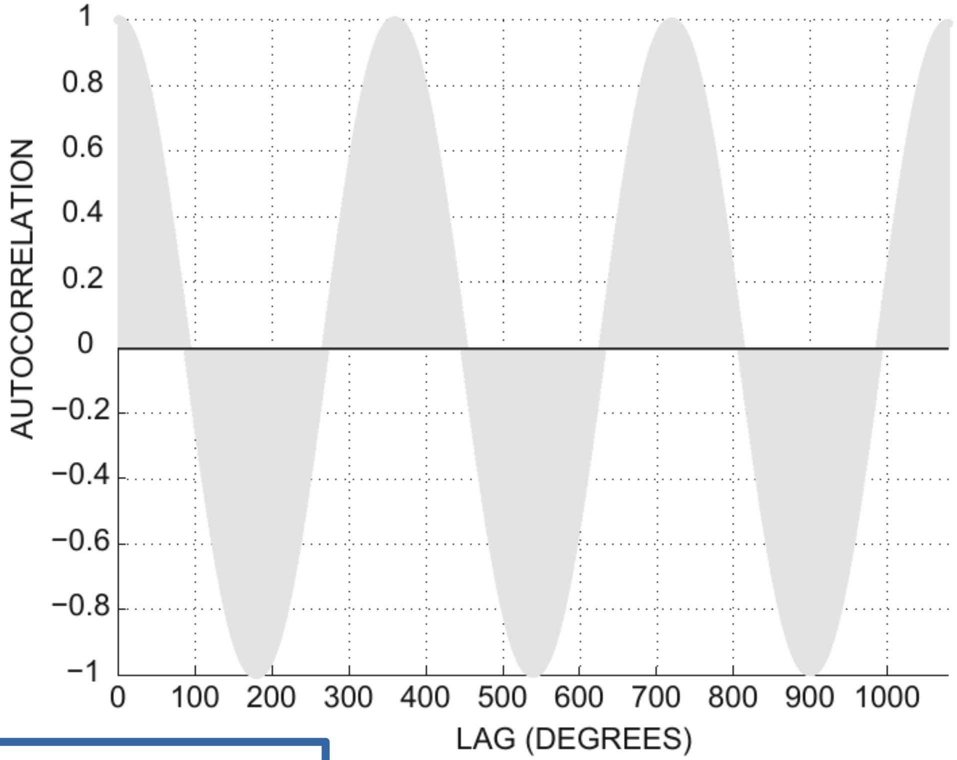
- Autocorrelation lines in [-1,1]
- High absolute values ⇒ predictability
- Autoregressive model of order p, AR(p):

$$y_t^{\text{AR}} = \sum_{i=1}^{p} a_i \cdot y_{t-i} + c + \epsilon_t$$

How to decide p? Autocorrelation plots



(a) IBM stock



(b) Sine wave

What is a reasonable range for p in these cases?

Finding coefficients and evaluating

- Each data point is a training element

$$y_t^{\mathrm{AR}} = \sum_{i=1}^{p} a_i \cdot y_{t-i} + c + \epsilon_t$$

- Coefficients found by least-squares regression
- Best models have $R^2 \to 1$

$$R^2 = 1 - \frac{\mathrm{Mean}_t(\epsilon_t^2)}{\mathrm{Variance}_t(y_t)}$$

Moving average model **MA(q)**

- Focus on the variations (shocks) of the model, i.e., places where change was unexpected
- AR(p) model: $$y_t^{\mathrm{AR}} = \sum_{i=1}^{p} a_i \cdot y_{t-i} + c + \epsilon_t$$
- MA(q) model: $$y_t^{\mathrm{MA}} = \sum_{i=1}^{q} b_i \cdot \epsilon_{t-i} + c + \epsilon_t$$

Autoregressive moving average model **ARMA(p,q)**

- Combines both the autoregressive and the moving average model

$$y_t^{\text{ARMA}} = \sum_{i=1}^{p} a_i \cdot y_{t-i} + \sum_{i=1}^{q} b_i \cdot \epsilon_{t-i} + c + \epsilon_t$$

- Select small p, q, to avoid overfitting

Autoregressive integrated moving average model **ARIMA(p,q)**

- Combines both the autoregressive and the moving average model on differenced series

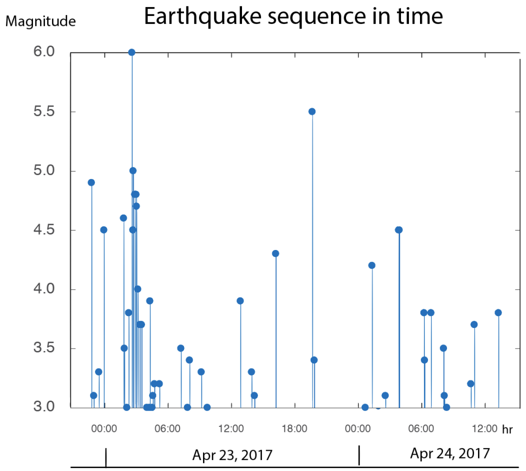
$$y_t^{\text{ARIMA}} = \sum_{i=1}^{p} a_i \cdot (y_{t-i} - y_{t-i-1}) + \sum_{i=1}^{q} b_i \cdot \epsilon_{t-i} + c + \epsilon_t$$

Note: this is an ARIMA(p,1,q) model as we're using first order differencing

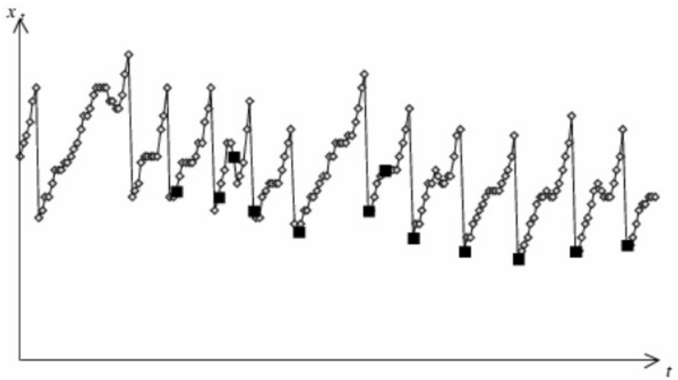
See also: [ARIMA end-to-end project in Python] by Susan Li (2018)

Event detection
(a simple framework)

Event: an important occurrence



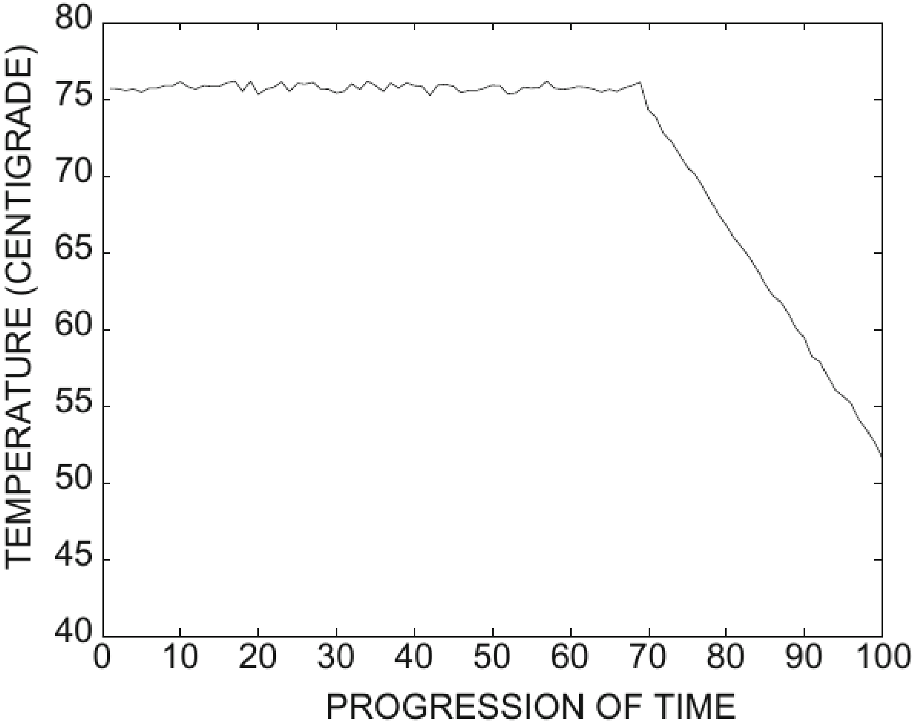
Earthquake or aftershock



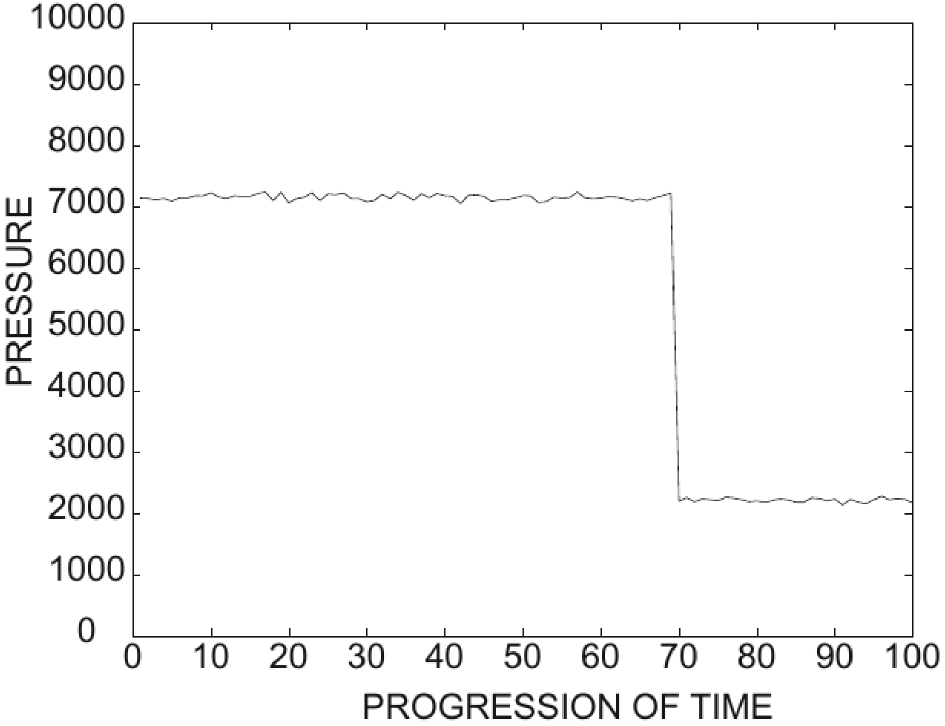
Droplet release

Sudden price change

Example: pipe rupture



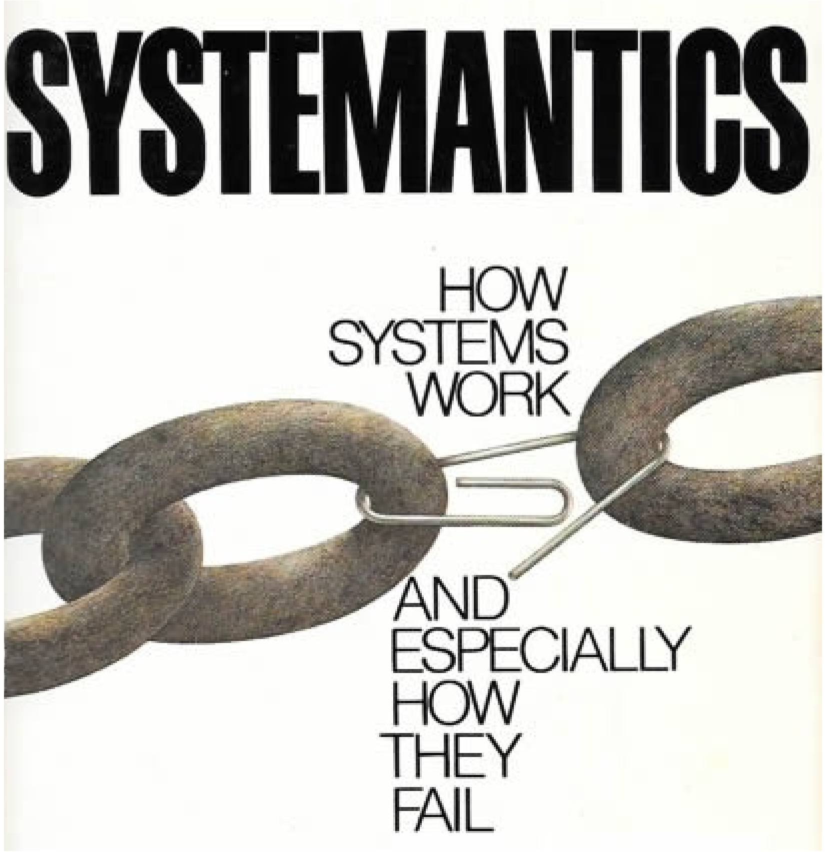
(a) Temperature (pipe rupture scenario)



(b) Pressure (pipe rupture scenario)

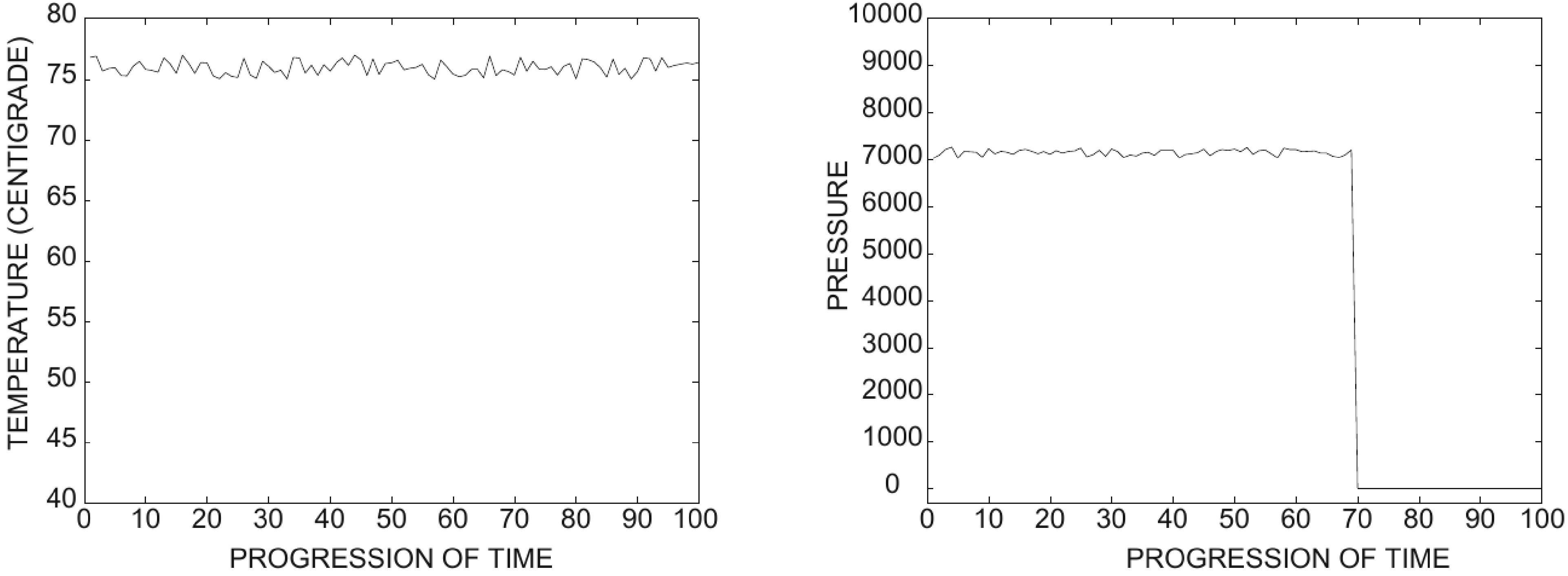
(… but what if sensors fail? …

- "Systems in general work poorly or not at all"
- **"In complex systems, malfunction and even total non-function may not be detectable for long periods, if ever"**



Gall, John. Systemantics: the underground text of systems lore:
how systems really work and especially how they fail. Ann Arbor, MI, 1975.

… can we detect failure? …)



(c) Temperature (sensor failure scenario) (d) Pressure (sensor failure scenario)

A general scheme for event detection in multivariate time series

- Let $T_1, T_2, \ldots, T_r$ be times at which an event has been observed in the past
- (Offline) Learn coefficients $\alpha_1, \alpha_2, \ldots, \alpha_d$ to distinguish between event times and non-event times
- (Online) Observe series and determine deviation of every stream i at timestamp t as z_t^i
- (Online) Compute composite alarm level $Z_t = \sum_{i=1}^{d} \alpha_i \cdot z_t^i$

Learning discrimination coefficients

$$\alpha_1, \alpha_2, \ldots, \alpha_d$$

$$Z_t = \sum_{i=1}^{d} \alpha_i \cdot z_t^i$$

- Average alarm level for events

$$Q^{\text{event}}(\alpha_1, \ldots, \alpha_d) = \frac{1}{r} \sum_{i=1}^{r} Z_{T^i}$$

- Average alarm level for non-events (we assume most points are non-events)

$$Q^{\text{normal}}(\alpha_1, \ldots, \alpha_d) = \frac{1}{N} \sum_{i=1}^{N} Z_t$$

Learning discrimination coefficients $\alpha_1, \alpha_2, \ldots, \alpha_d$ (cont.)

- For events $$Q^{\text{event}}(\alpha_1, \ldots, \alpha_d) = \frac{1}{r}\sum_{i=1}^{r} Z_{T^i}$$

- For non-events $$Q^{\text{normal}}(\alpha_1, \ldots, \alpha_d) = \frac{1}{N}\sum_{i=1}^{N} Z_t$$

Maximize $$Q^{\text{event}}(\alpha_1, \ldots, \alpha_d) - Q^{\text{normal}}(\alpha_1, \ldots, \alpha_d)$$

subject to $$\sum_{i=1}^{d} \alpha_i^2 = 1$$

Use any off-the-shelf iterative optimization solver

Summary

Things to remember

- Series preparation
  - Interpolation
  - Smoothing
- Dynamic time warping
- Time series forecasting
- Event detection

Exercises for this topic

- Data Mining, The Textbook (2015) by Charu Aggarwal
  - Exercises 14.10 → 1-6