

Mining Time Series:

Forecasting

Mining Massive Datasets

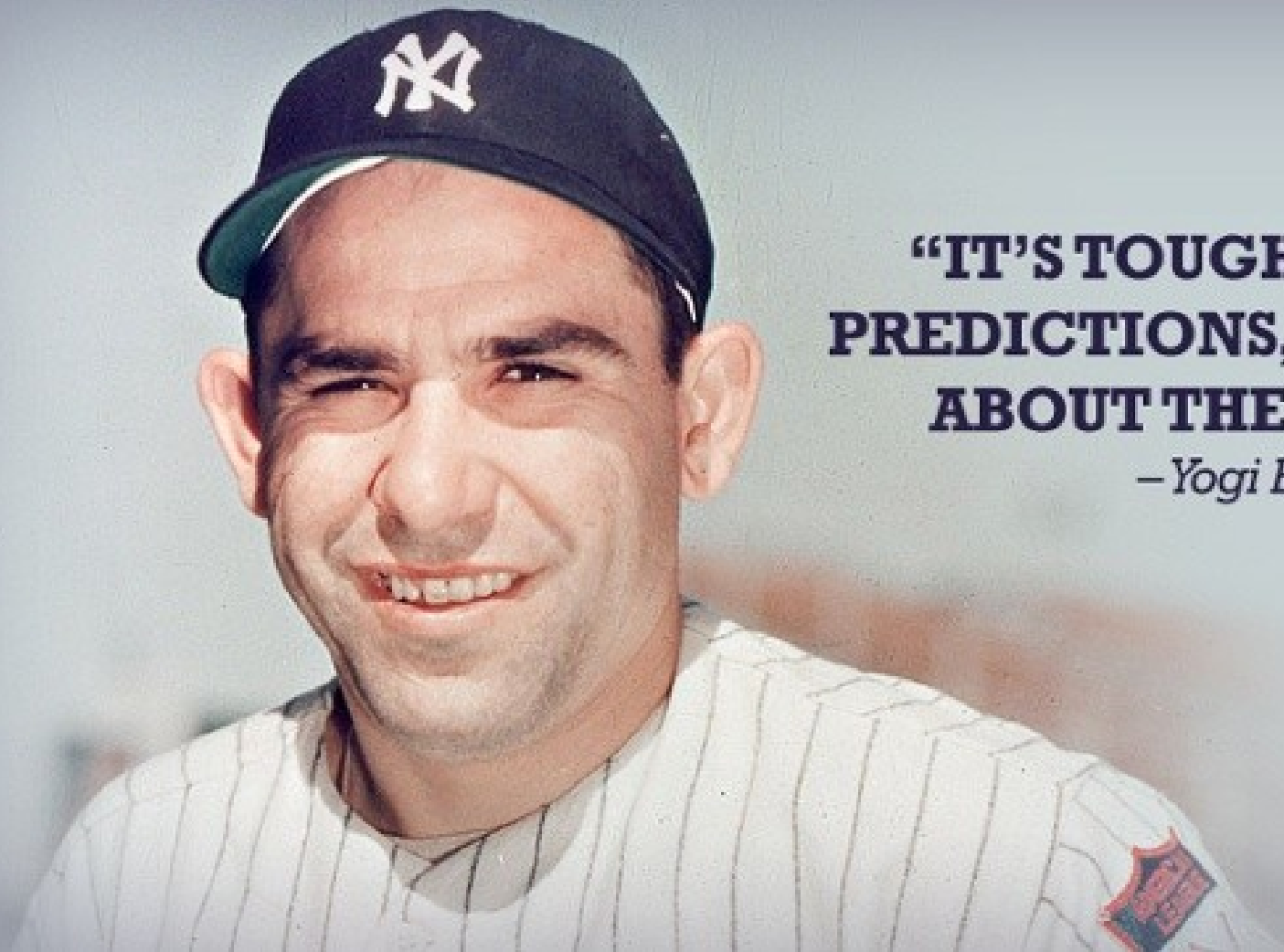
Prof. Carlos Castillo — <https://chato.cl/teach>



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



**“IT’S TOUGH TO MAKE
PREDICTIONS, ESPECIALLY
ABOUT THE FUTURE.”**

–Yogi Berra

(A similar phrase is attributed to Niels Bohr, Danish physicist and winner of the Nobel Prize in 1922)

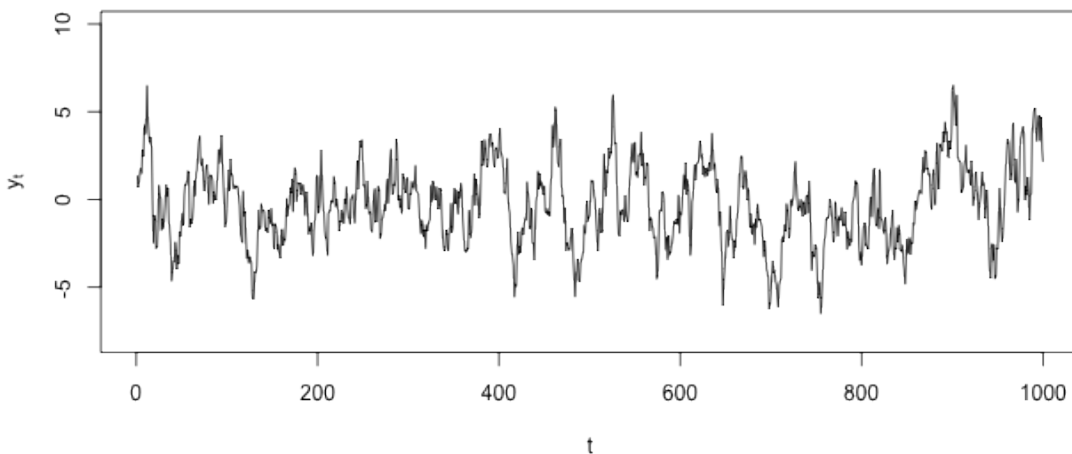
Forecasting

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

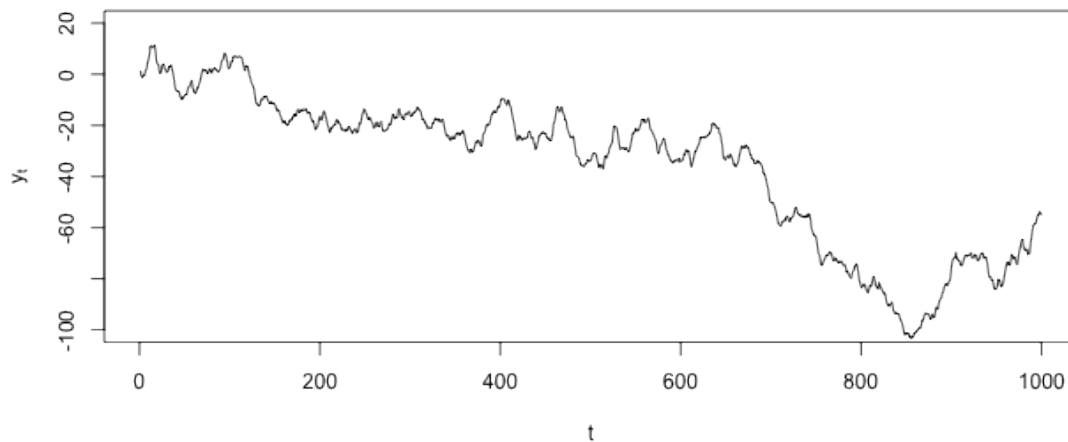
Stationary vs Non-Stationary processes

- **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

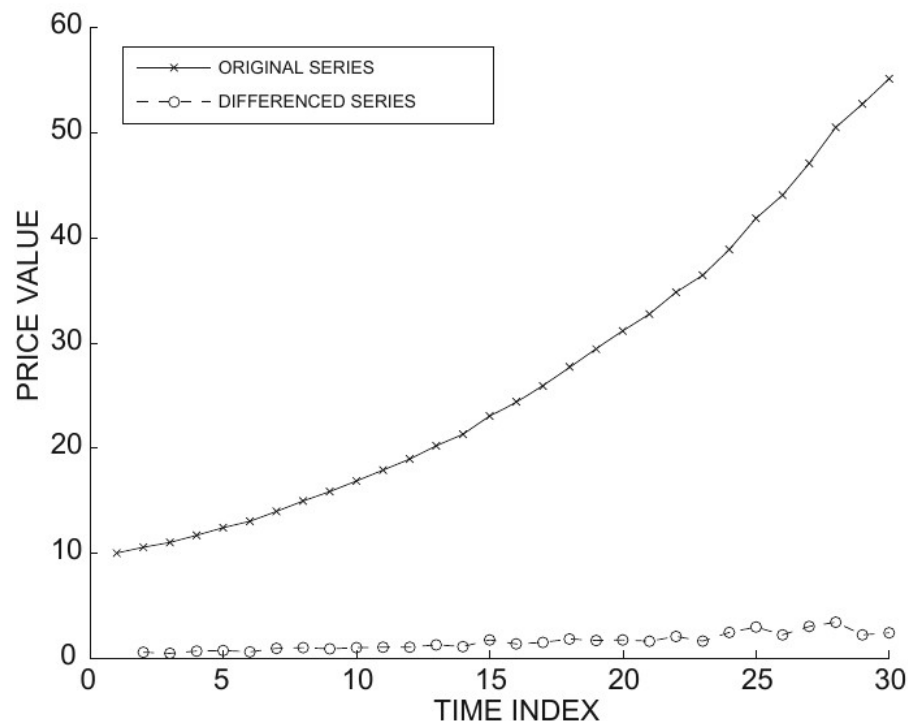
- In this case, current parameters (e.g., mean) are good predictors of future parameters

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?

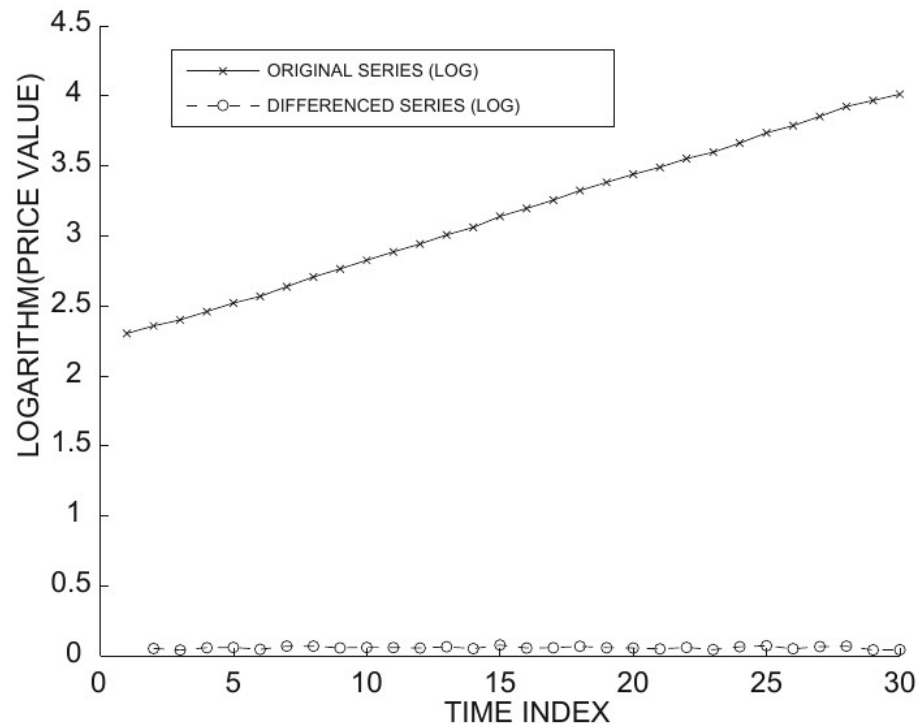


Differencing (cont.)

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

$$\begin{aligned}y_i'' &= y_i' - y_{i-1}' \\ &= y_i - 2 \cdot y_{i-1} + y_{i-2}\end{aligned}$$

- $y_i' = y_i - y_{i-m}$ differencing (m = 24 hours, 7 days, ...)

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

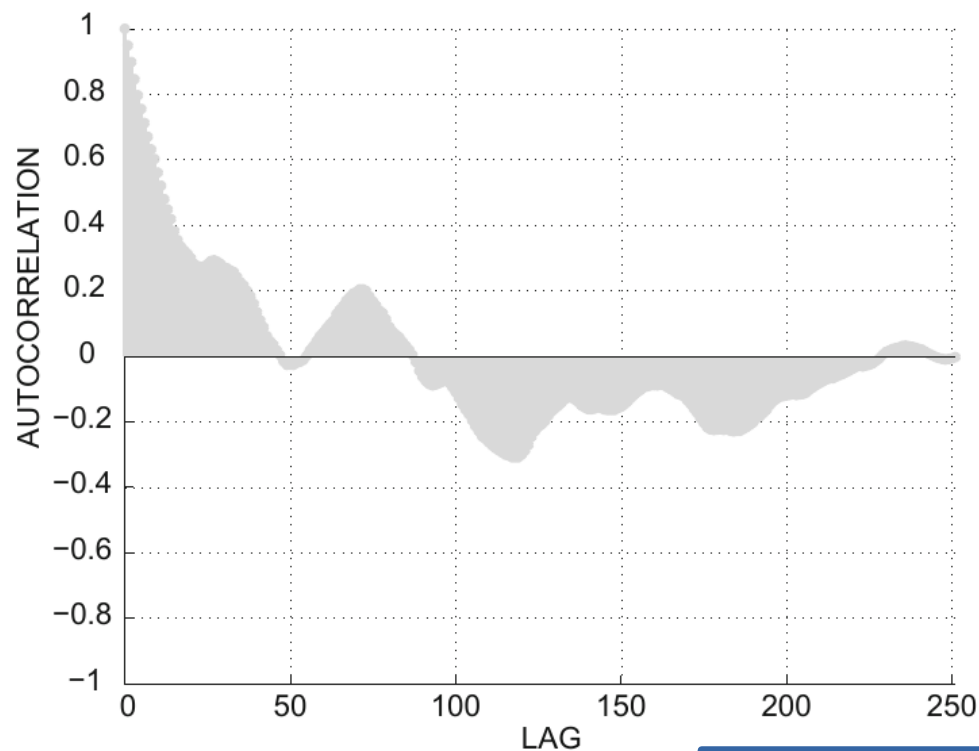
Autoregressive model AR(p)

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

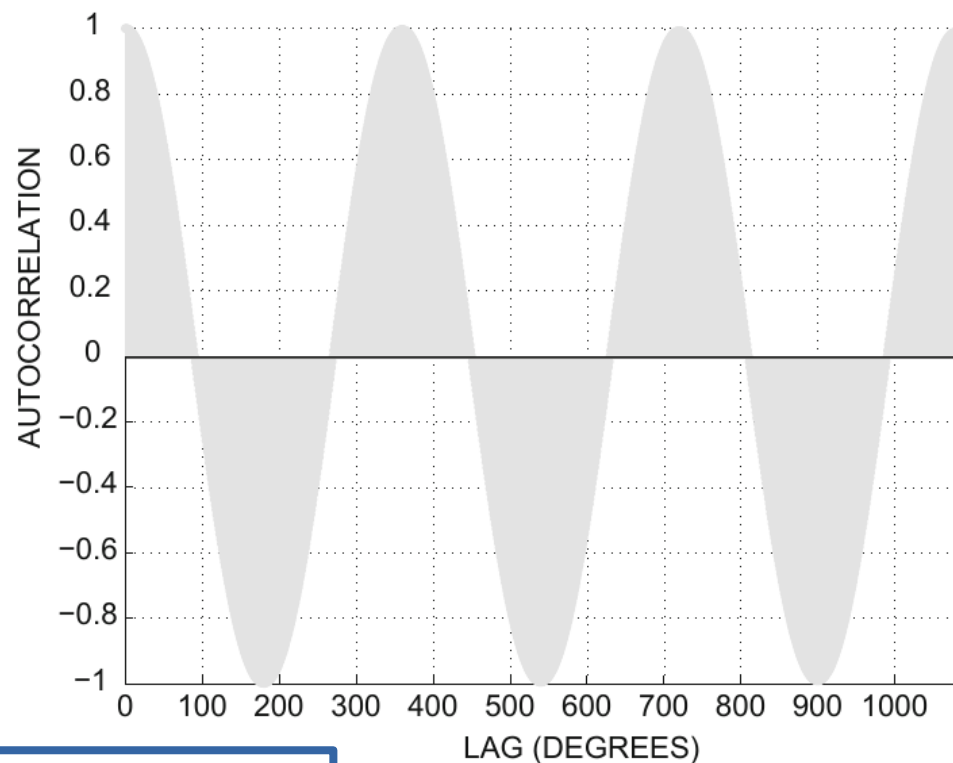
- Autocorrelation lines in $[-1,1]$
- High absolute values \Rightarrow predictability
- Autoregressive model of order p , AR(p):

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

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

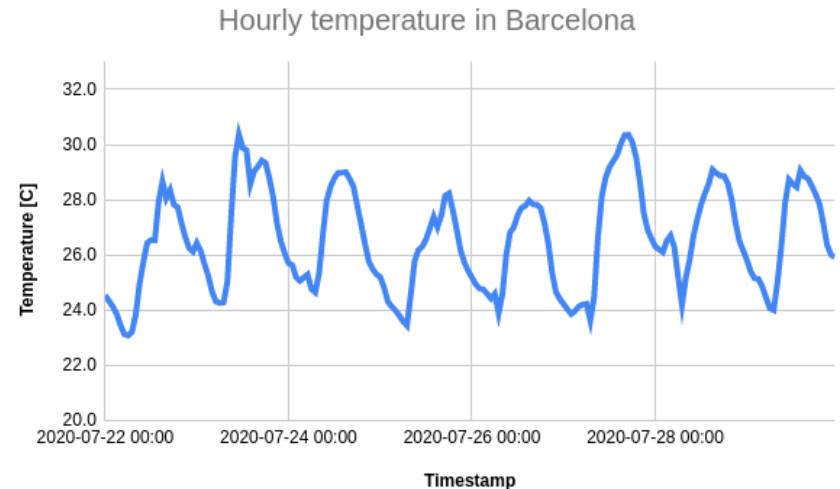
$$y_t^{\text{AR}} = \sum_{i=1}^p a_i \cdot y_{t-i} + c + \epsilon_t$$

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

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

Exercise: simple auto-regressive model

- Create a simple auto-regressive model for temperature in a city
- Use two lags:
 - 1 hour
 - 24 hours
- Compute the predicted series
(optionally: include it in the plot)
- Compute the maximum error



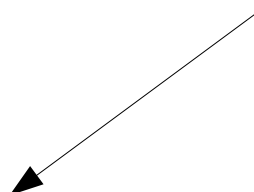
Spreadsheet links: <https://upfbarcelona.padlet.org/chato/hogch321o6pws1fd>

Moving average model MA(q)

- Focus on the variations (shocks) of the model, i.e., places where **change was unexpected**

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

- MA(q) model:
$$y_t^{\text{MA}} = \sum_{i=1}^q b_i \cdot \epsilon_{t-i} + c + \epsilon_t$$



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

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

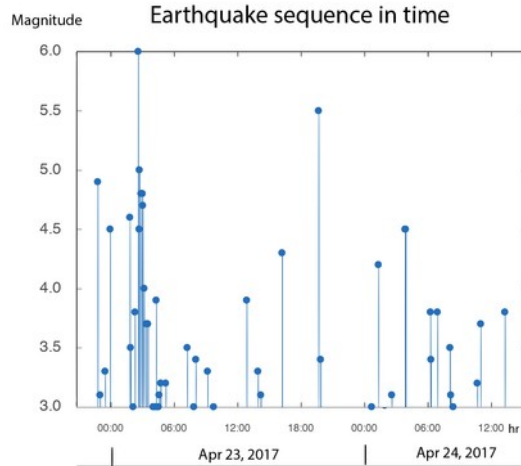
Note: this is an ARIMA(p,1,q) model as we're using first order differencing

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

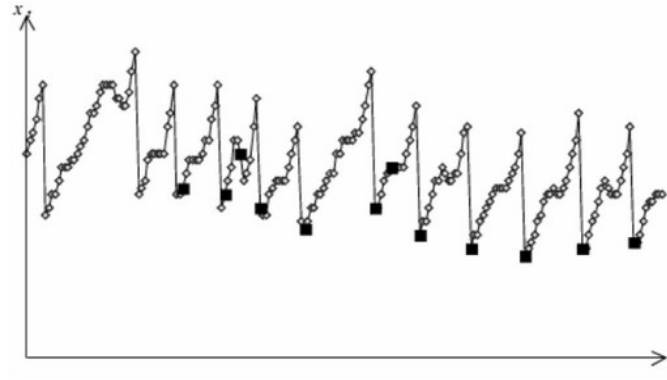
Event detection

(a simple framework)

Event: an important occurrence



Earthquake or
aftershock

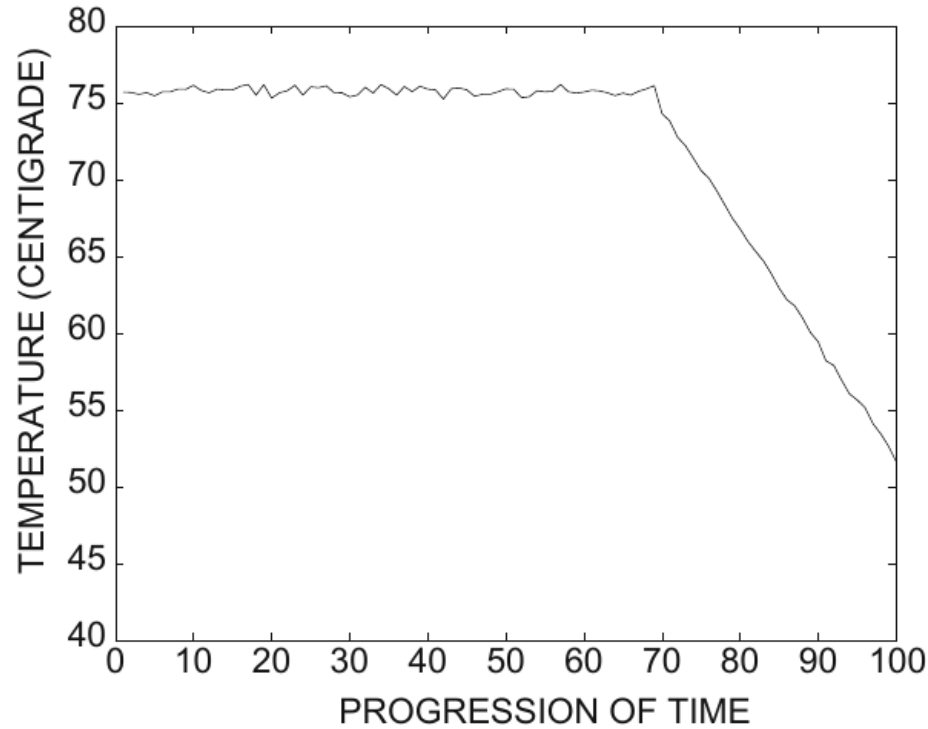


Droplet
release

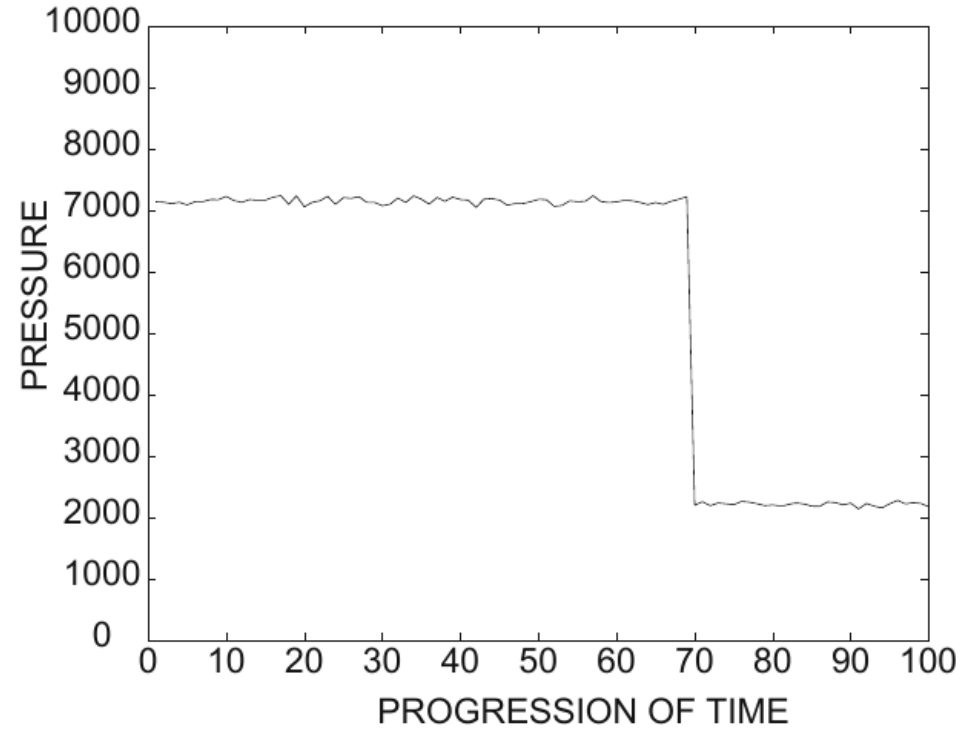


Sudden
price change

Example: pipe rupture



(a) Temperature (pipe rupture scenario)

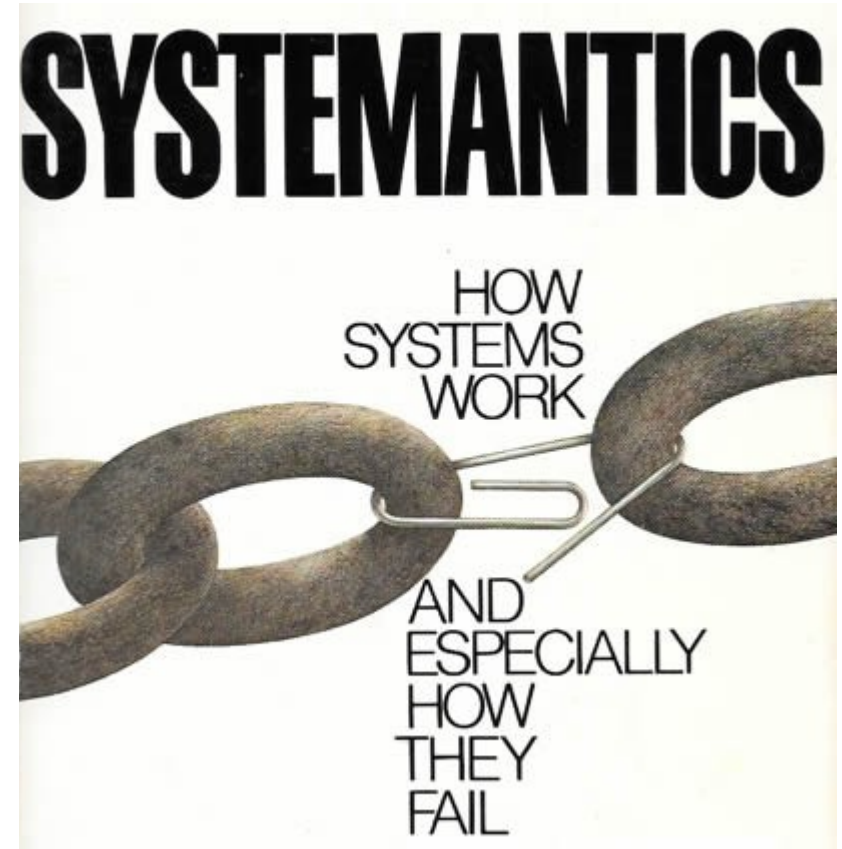


(b) Pressure (pipe rupture scenario)

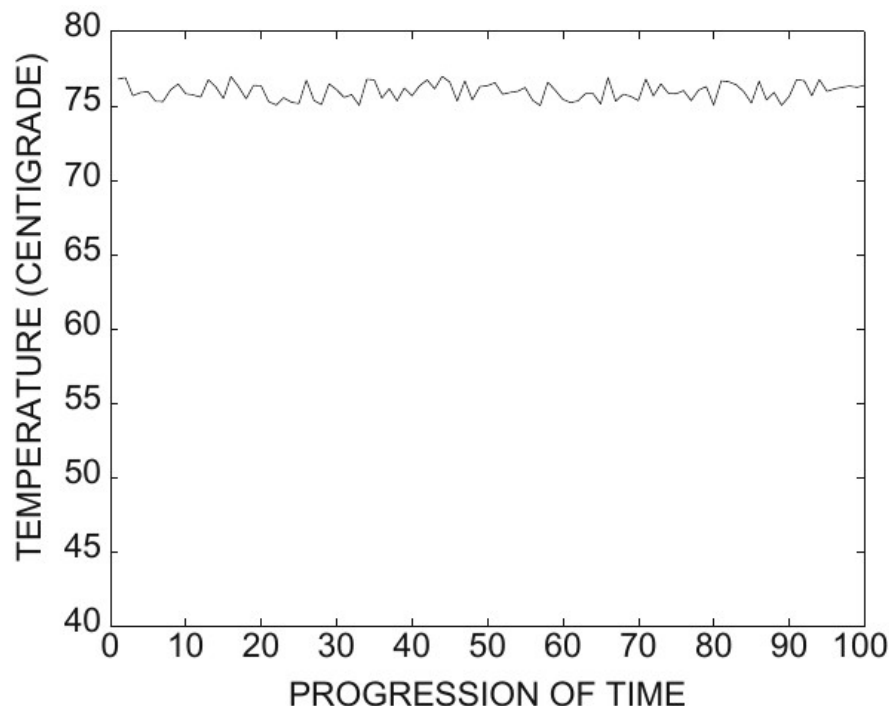
(But what if sensors fail? ...)

- “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”

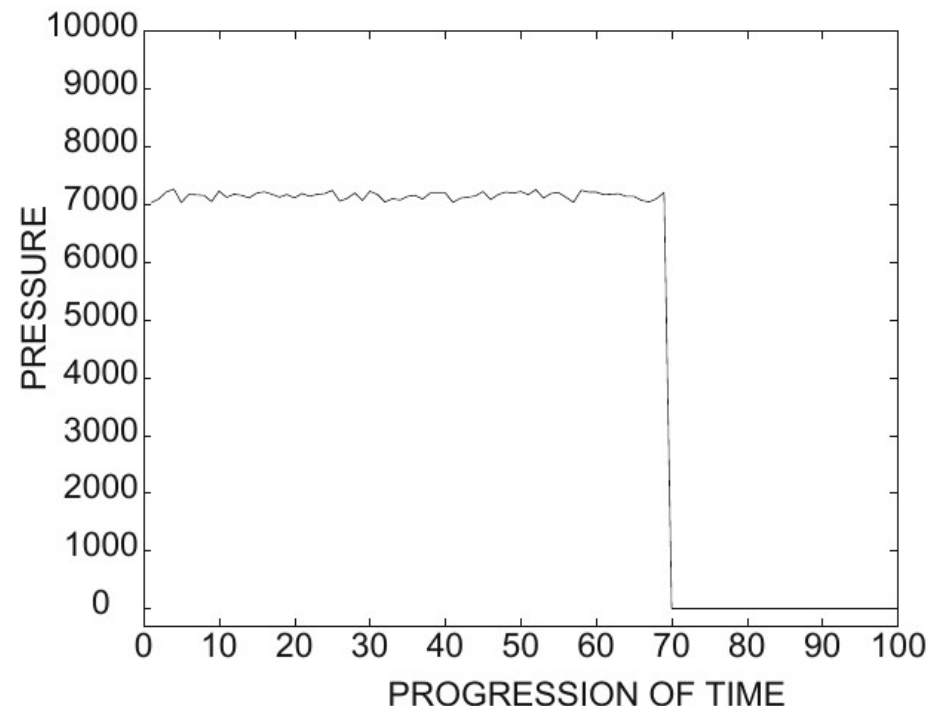
Gall, John. Systemantics: the underground text of systems lore: how systems really work and especially how they fail. Ann Arbor, MI, 1975.



... Can we still 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, \dots, T_r be times at which an event has been observed in the past
- (Offline) Learn coefficients $\alpha_1, \alpha_2, \dots, \alpha_d$ 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 α_1 ,

$$\alpha_2, \dots, \alpha_d$$

- Average alarm level for events

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

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

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

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

Learning discrimination coefficients

$\alpha_1, \alpha_2, \dots, \alpha_d$ (cont.)

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

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

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

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

Use any off-the-shelf iterative optimization solver

Summary

Things to remember

- Time series forecasting
- Event detection

Exercises for TT27-TT29

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