Temporal Analysis

Introduction to Temporal Analysis and Visualization

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Describe temporal analysis

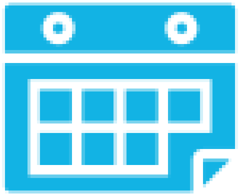
Temporal Analysis

"Time is an outstanding dimension reflected by Shneiderman's Task by Data Type Taxonomy."

W. Aigner, S. Miksch, W. Muller, H. Schumann, C. Tominski, "Visual Methods for Analyzing Time-Oriented Data", IEEE Trans. On Visualization and Computer Graphics, Vol. 14, No. 1, Jan.-Feb. 2008, pp. 47-60.

Time-Oriented Data

- Time oriented data is ubiquitous
  - Stock markets
  - Movie trends
  - Business
  - Medicine



- Each data case is likely an event of some kind, with one variable being the date and time

Time Series

“A random selection of 4000 graphs from 15 newspapers and magazines worldwide showed that between 1974 and 1980, 75% of these graphs were time series.”

E. Tufte, “The Visual Display of Quantitative Information,” 1983.

Time Series

What questions can we ask of these visuals?

- Does a data object exist at a certain time?
- When does a certain data object exist?
- How long does a data object exist?
- How fast and how much does the data object change?
- What order to objects appear/disappear?
- Is there a cyclical pattern to appearances?
- Which objects exist simultaneously?

E. Tufte, “The Visual Display of Quantitative Information,” 1983.

Time is…

Ordered

Continuous

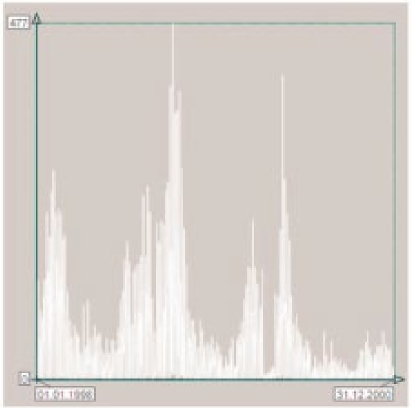
Cyclical

Independent of location

Linear vs. Cyclical Time

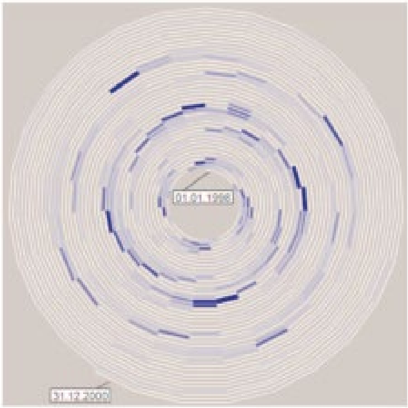
Linear time

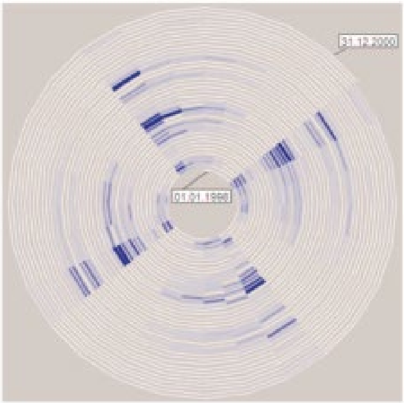
- One time point precedes another
- Time being ordered is closely bound to notion of causality



Cyclical Time

- The ordering of points in a cyclic time domain would be meaningless
- Winter comes before summer, but also after summer





Temporal Analysis

Temporal Visualization

Ira A. Fulton Schools of Engineering
Arizona State University

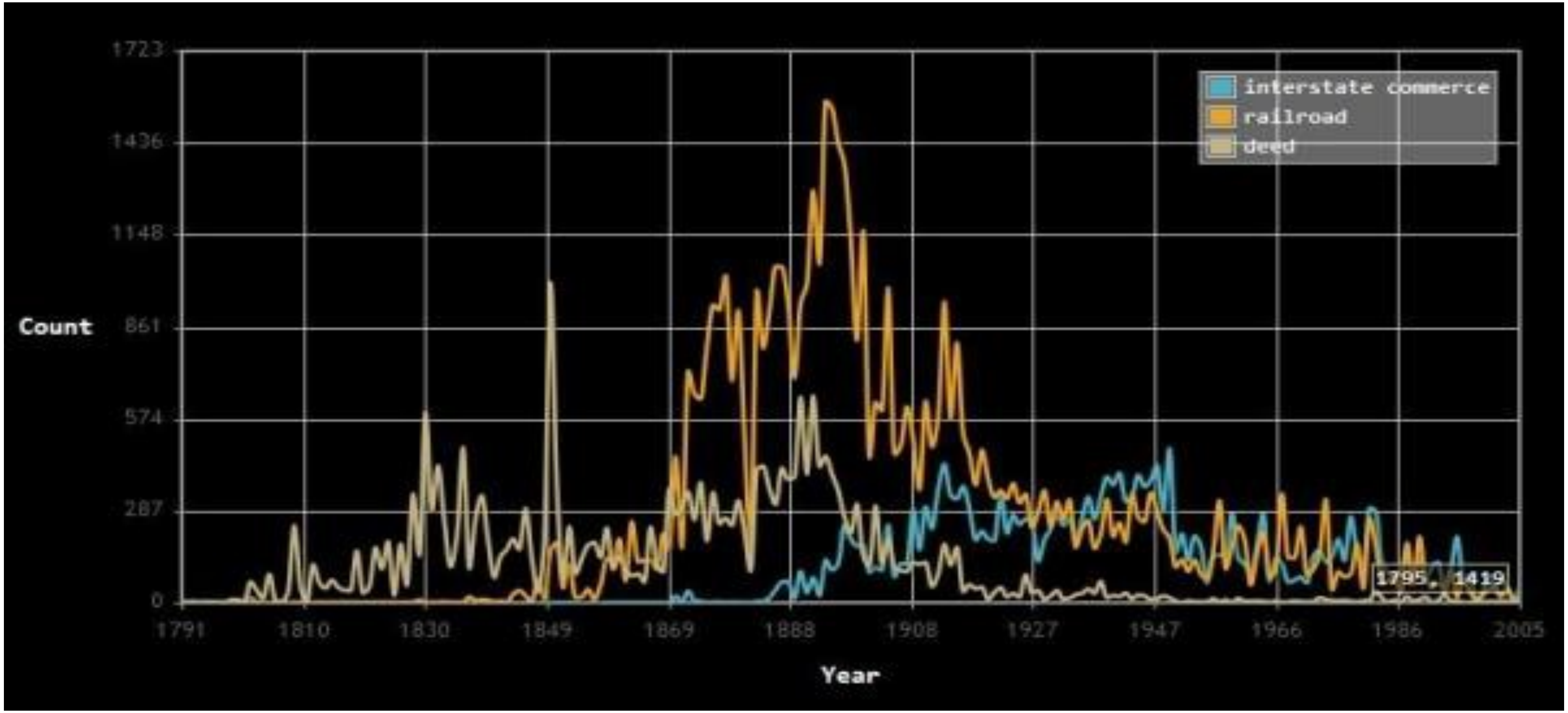
Objective

Objective

Apply methods of temporal analysis

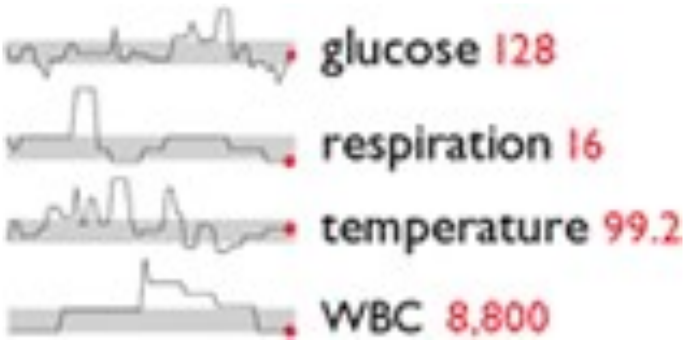
Linear Time

2D line graph with time on x-axis and value on y-axis



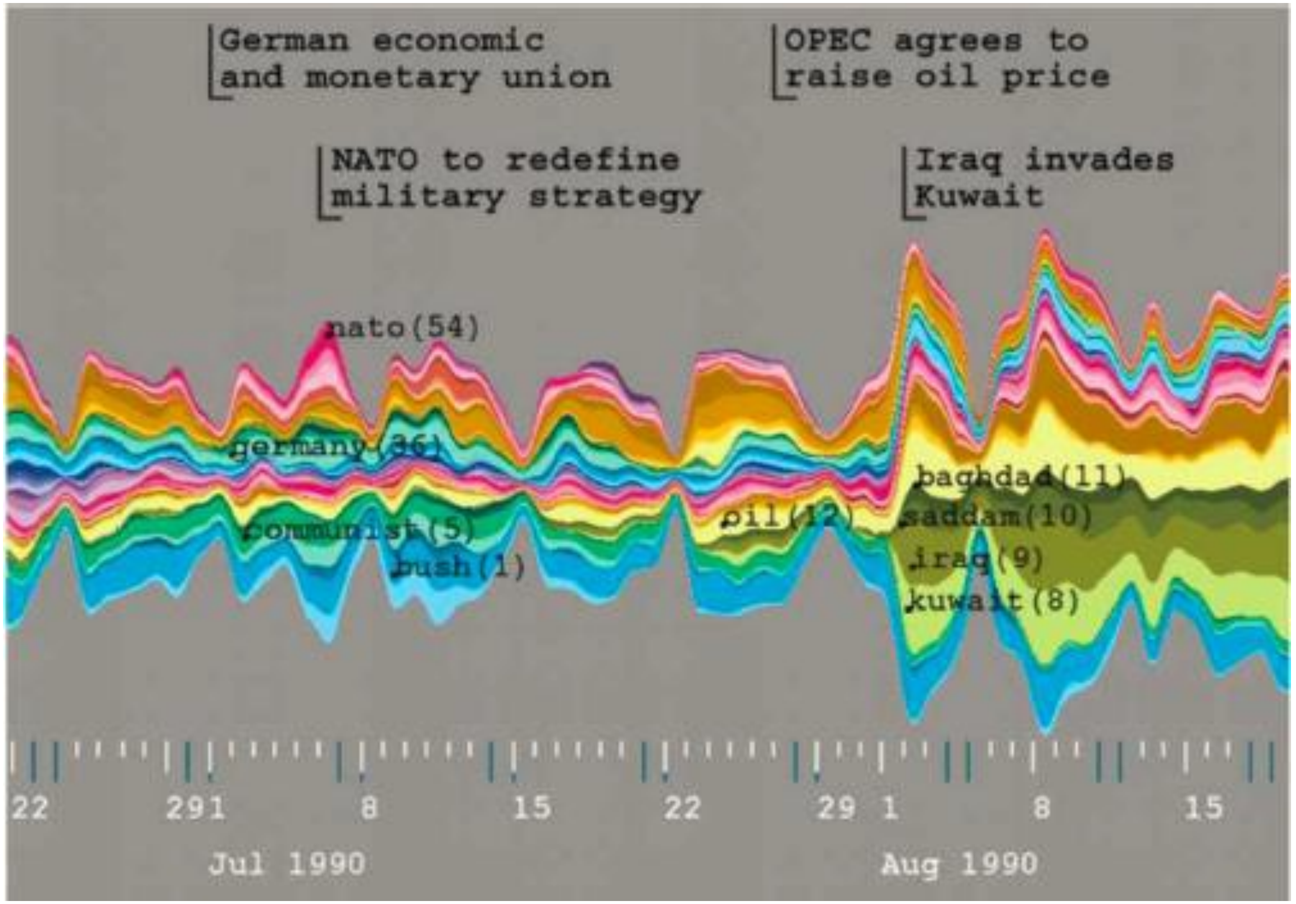
New Database Creates Time-Series Plots of Phrases in U.S. Supreme Court Opinions December 16, 2011 | Posted by: Scott C. Idleman: http://law.marquette.edu/facultyblog/2011/12/16/new-database-creates-time-series-plots-of-phrases-in-u-s-supreme-court-opinions/

“Intense, simple, wordlike graphics”



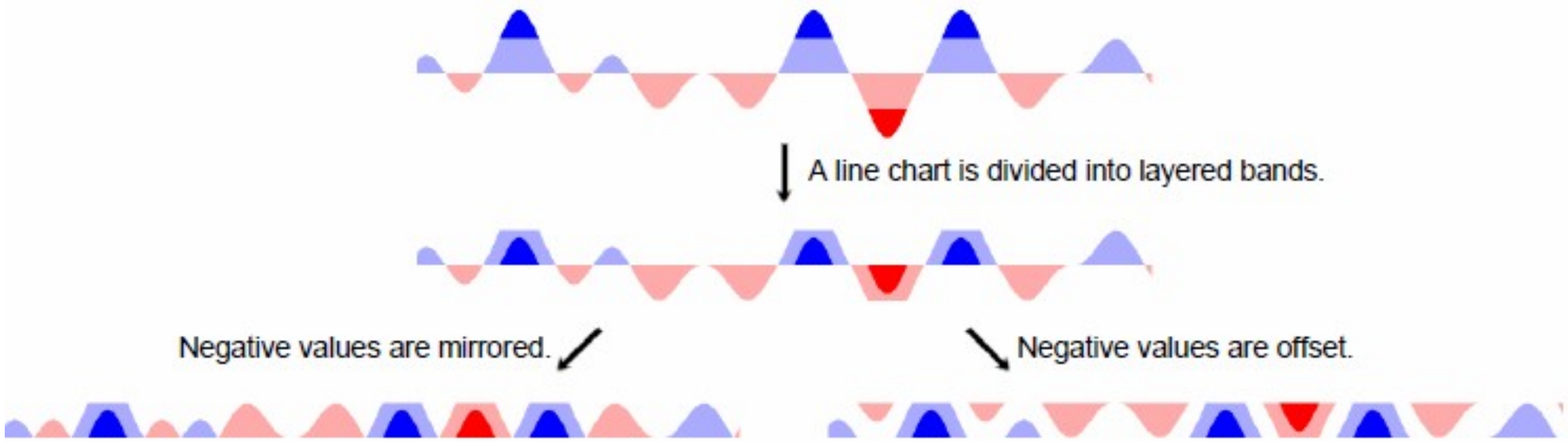
http://sparkline.org/

Theme River



1– S. Havre, E. Hetzler, P. Whitney, and L. Nowell. Themeriver: Visualizing thematic changes in large document collections. IEEE Transactions on Visualization and Computer Graphics 8(1):9-20, 2002.* - Theme River Image from: http://www.nytimes.com/interactive/2008/02/23/movies/20080223_REVENUE_GRAPHIC.html#

Sizing the Horizon



Jeffrey Heer, Nicholas Kong, and Maneesh Agrawala. 2009. Sizing the horizon: the effects of chart size and layering on the graphical perception of time series visualizations. In *Proceedings of the 27th international conference on Human factors in computing systems* (CHI '09). ACM, New York, NY, USA, 1303-1312.

Linear Vs. Cyclical

Trends are easily seen in a linear plot, but what about repeating patterns?

- Use spirals as they easily represent idea of repetition
- Finding periodicity may be easier than a bar chart

Spiral Graph

Archimedean Spiral

$$r = a + b\theta$$

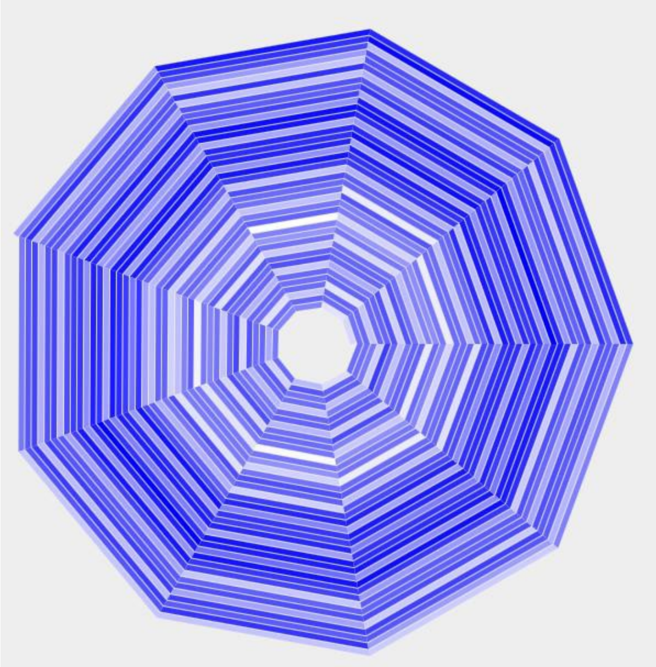
Logarithmic Spiral

$$r = ae^{b\theta}$$

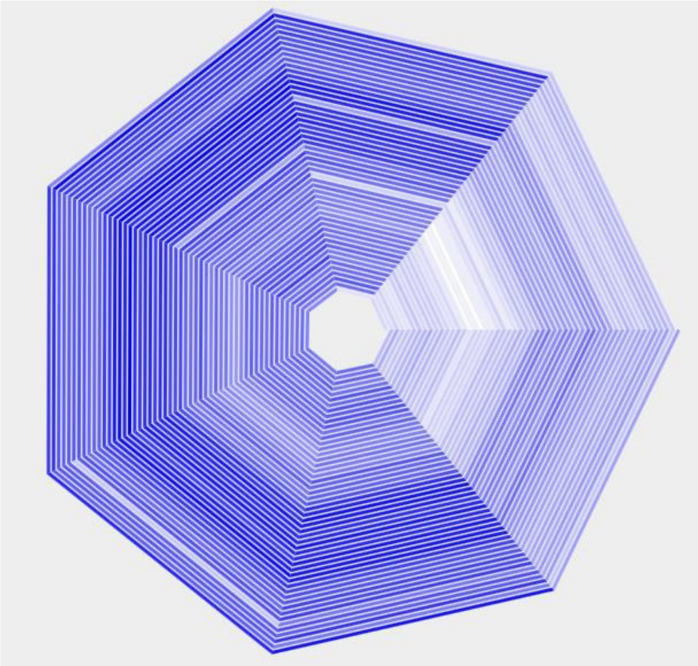
$$\theta = \frac{1}{b}\ln\left(\frac{r}{a}\right)$$

Spiral Graph

9 Day Spiral



7 Day Spiral

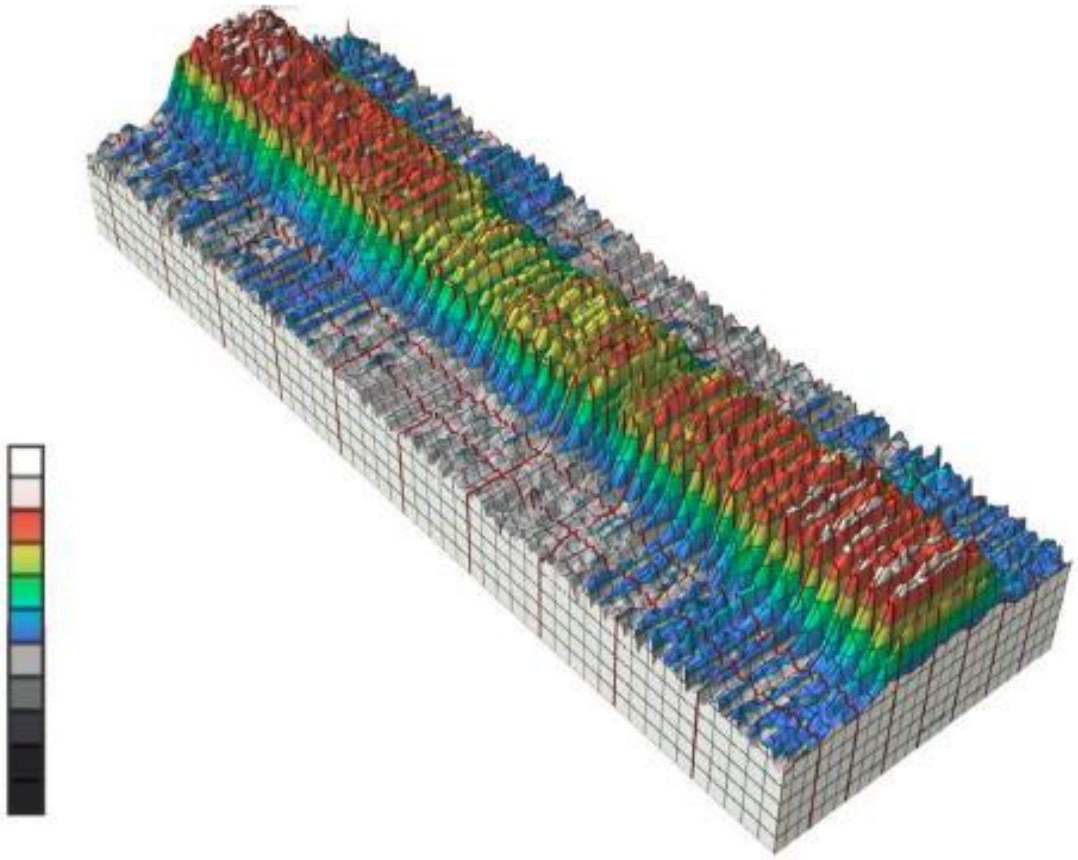


1 - John V. Carlis and Joseph A. Konstan, Interactive Visualization of Serial Periodic Data, Proceedings User Interface Software and Technology (UIST), pp. 29- 1998

2 - M. Weber, M. Alexa, and W. Mu¨ ller, “Visualizing Time-Series on Spirals,” Proceedings of the IEEE Symposium on Information Visualization (InfoVis ’01), 7-14, Oct. 2001.

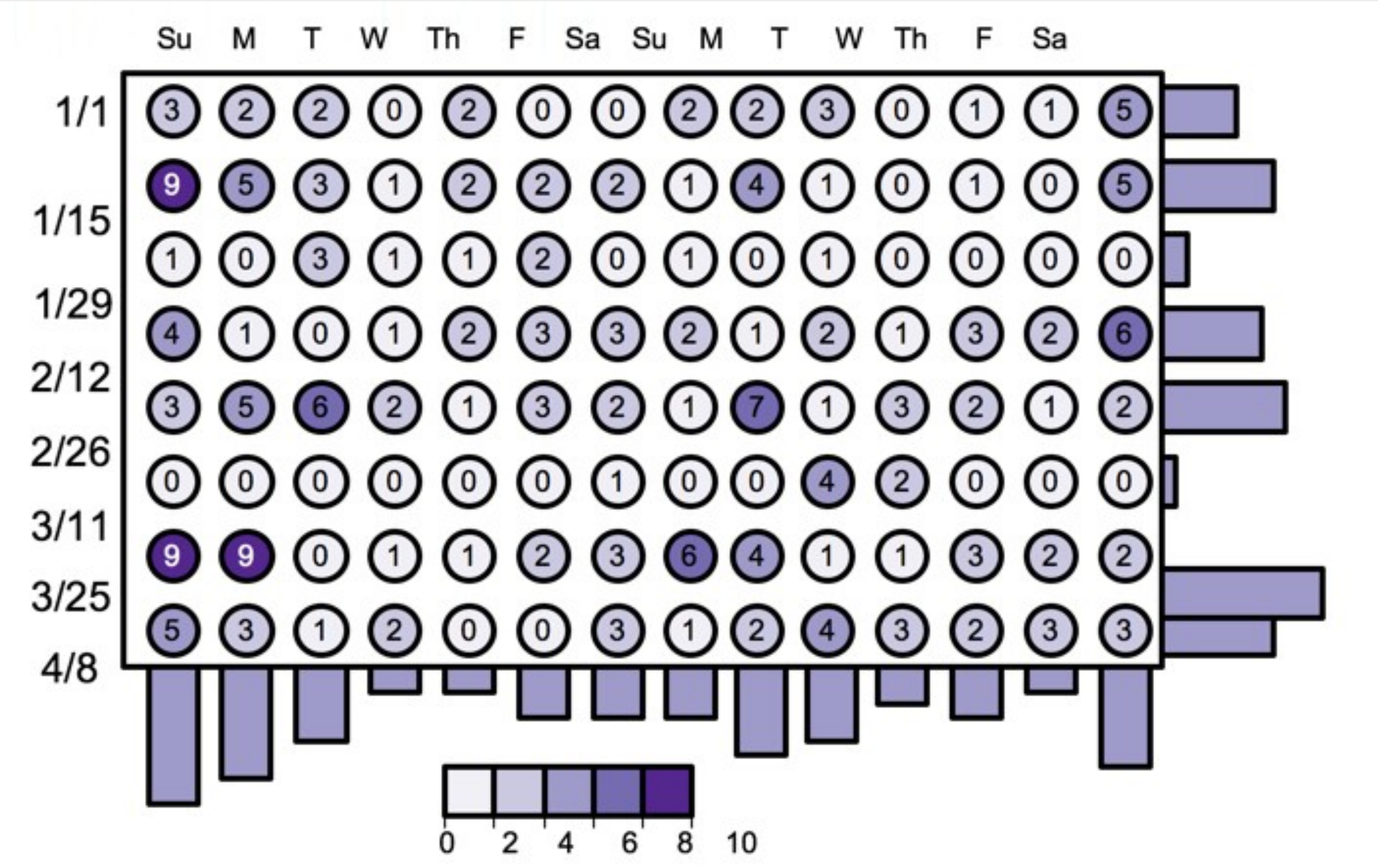
3 - Spirals for Periodic Data By Robert Kosara On August 7, 2011: http://eagereyes.org/techniques/spirals

Calendar Based Visualization



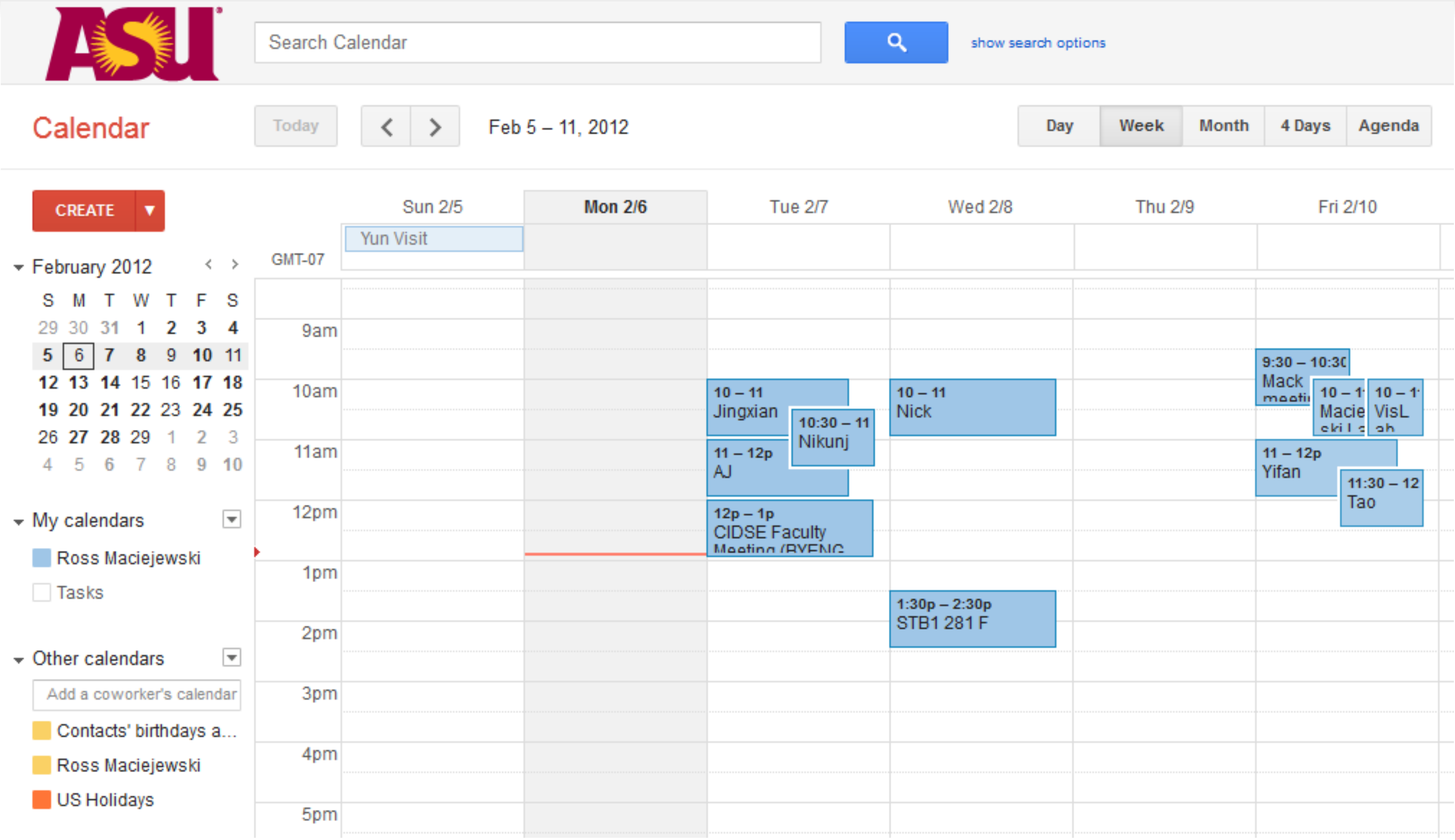
1 – J. J. Van Wijk and E. R. Van Selow. *Cluster and calendar based visualization of time series data*. Proceedings of the 1999 IEEE Symposium on Information Visualization, 1999.

Calendar Based Visualization



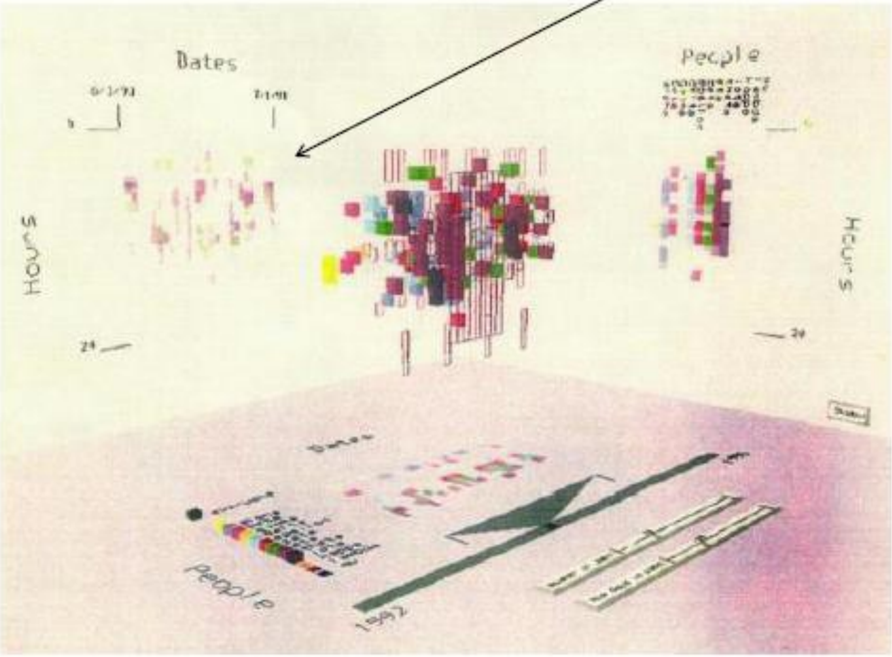
1 – J. J. Van Wijk and E. R. Van Selow. *Cluster and calendar based visualization of time series data*. Proceedings of the 1999 IEEE Symposium on Information Visualization, 1999.

Calendar Visualization



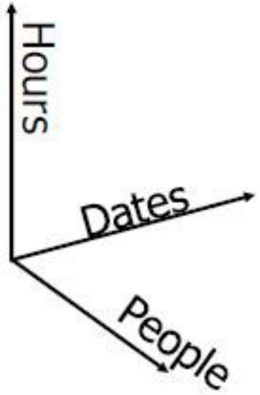
Spiral Calendar

Empty spots on back wall show good times



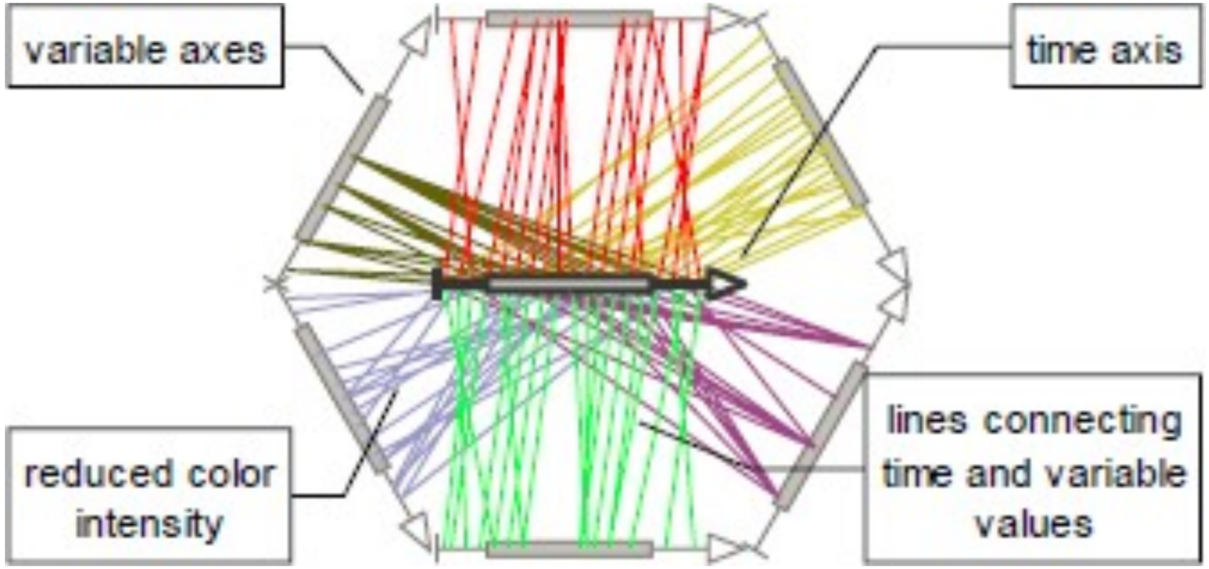
Uses projected shadows on walls

Time Lattice



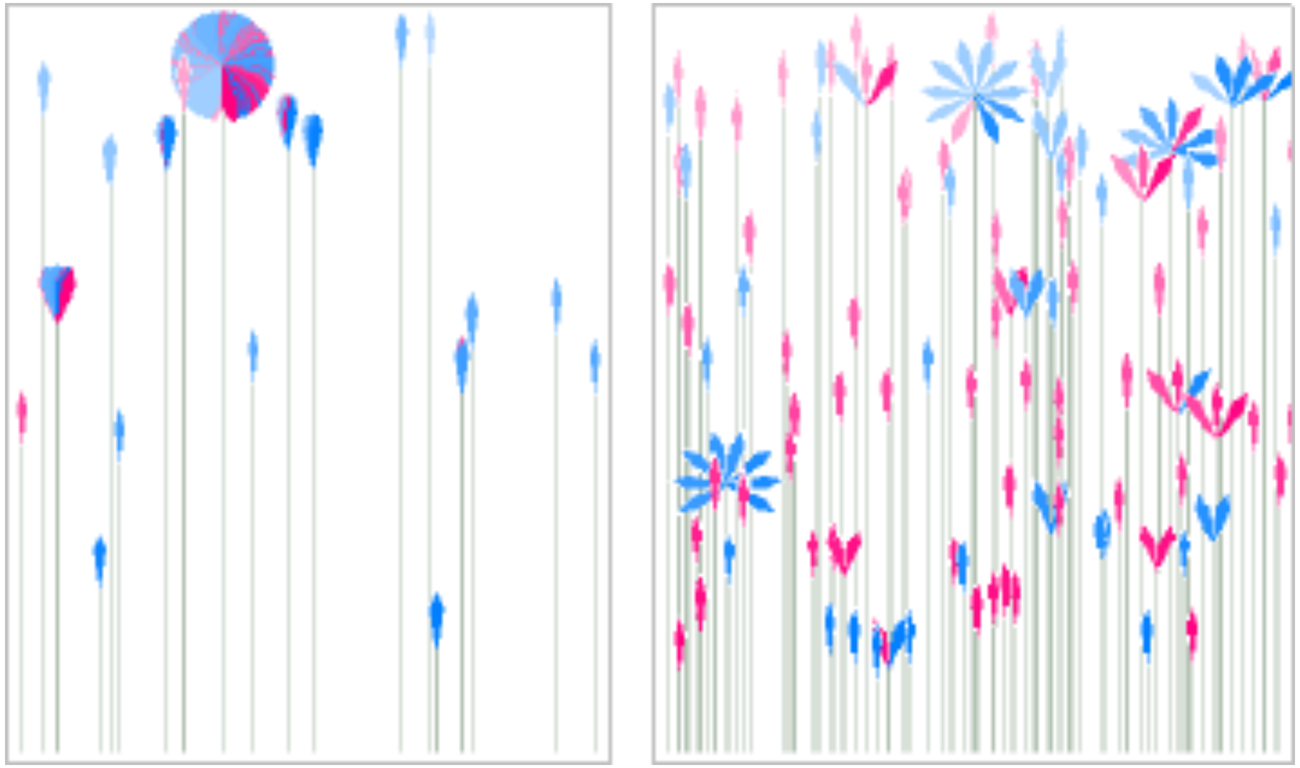
Mackinlay, J., Robertson, G. G., and DeLine, R., "Developing Calendar Visualizers for the Information Visualizer," UIST 1994.

Time Wheel



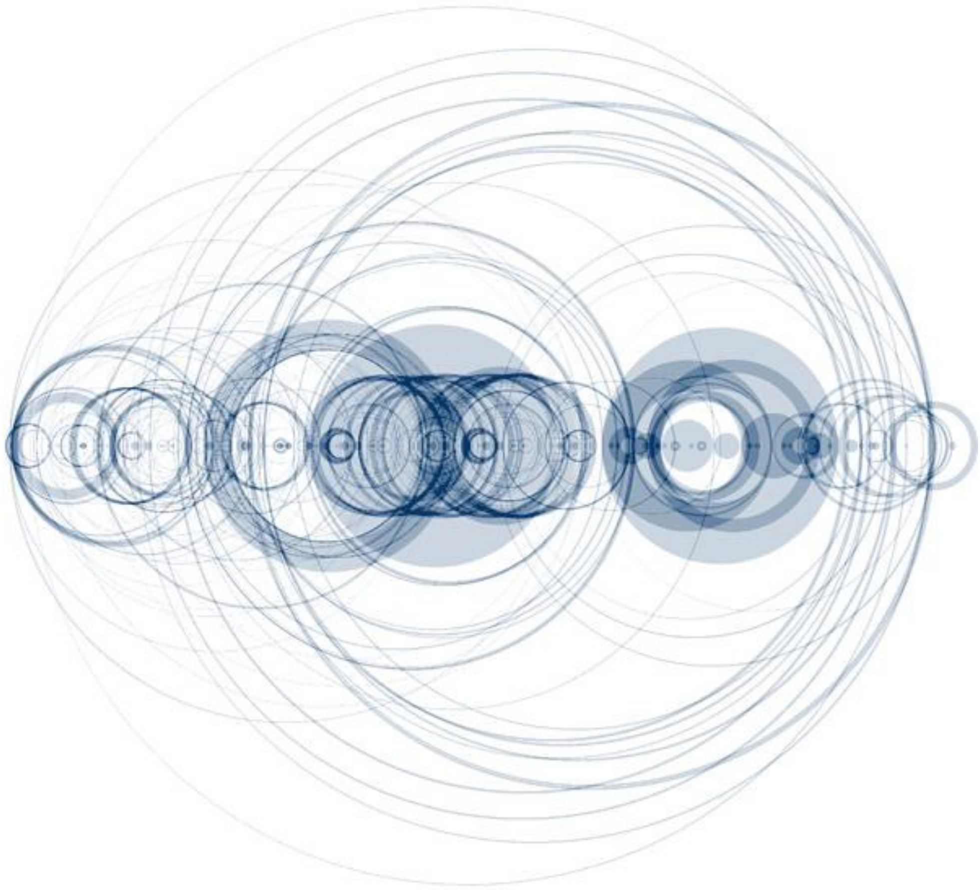
C. Tominski, J. Abello, and H. Schumann, “Axes-Based Visualizations with Radial Layouts,” Proceedings of the ACM Symposium on Applied Computing, pp. 1242-1

People Gardens



Rebecca Xiong and Judith Donath. 1999. PeopleGarden: creating data portraits for users. In *Proceedings of the 12th annual ACM symposium on User interface software and technology* (UIST '99). ACM, New York, NY, USA, 37-44.

Arc Diagrams



1– M. Wattenberg, "Arc Diagrams: Visualizing Structure in Strings," InfoVis 2002. http://www.bewitched.com/song.html

Ordered Time vs. Branching Time

Ordered time

domains consider things that happen one after another

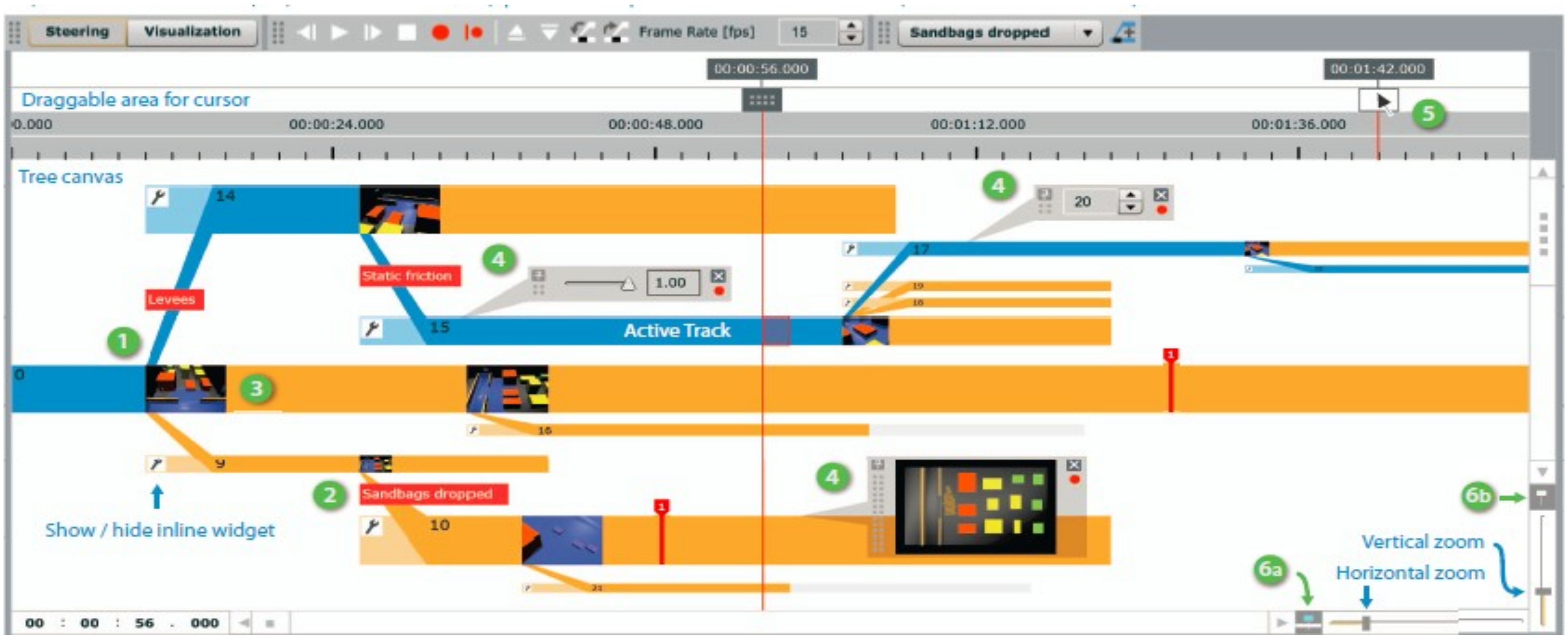
Branching time

considers multiple what-if scenarios, allowing comparison of alternate scenarios

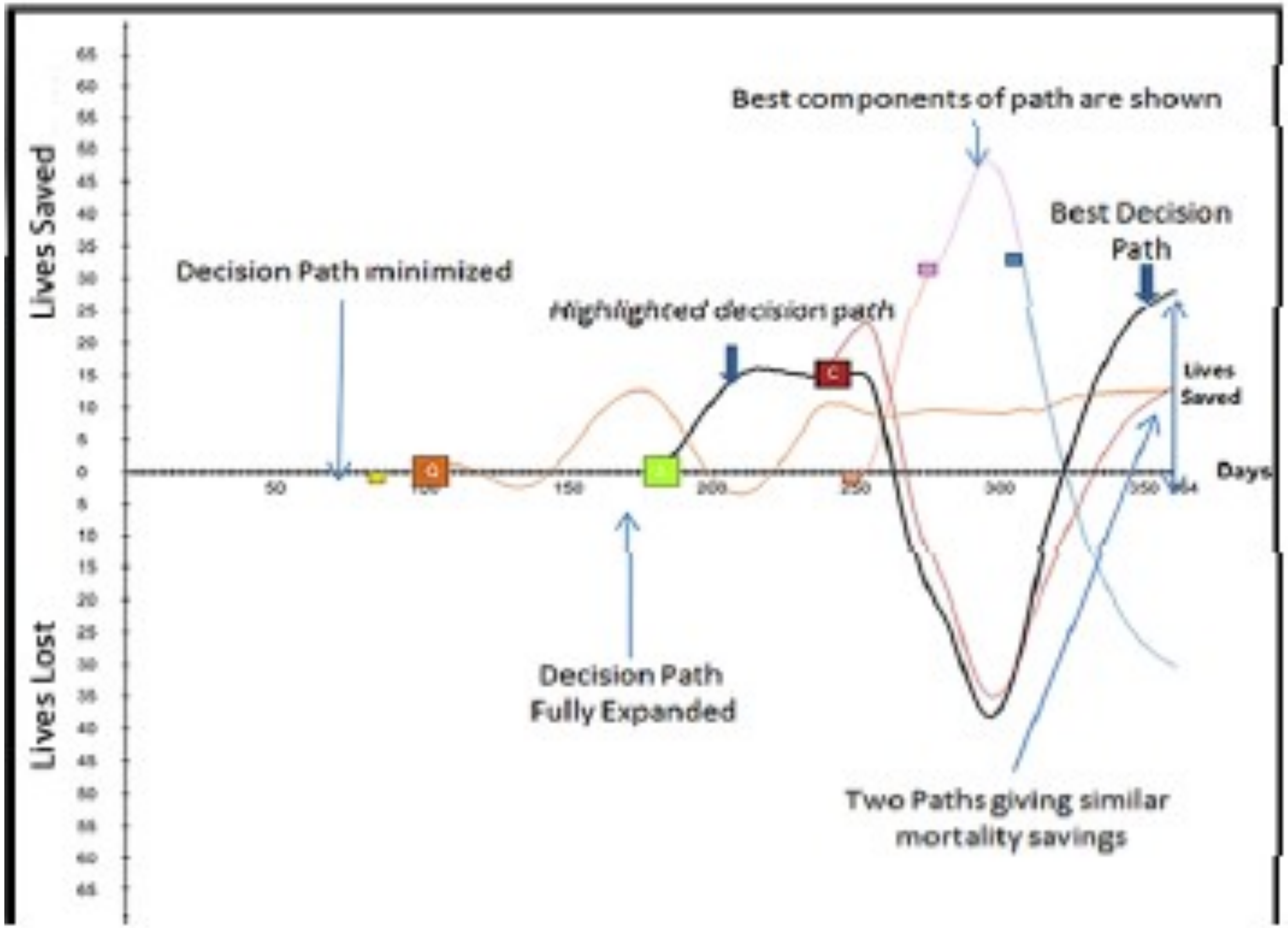
W. Aigner, S. Miksch, W. Muller, H. Schumann, C. Tominski, "Visual Methods for Analyzing Time-Oriented Data", IEEE Trans. On Visualization and Computer Graphics, Vol. 14, No. 1, Jan.-Feb. 2008, pp. 47-60.

World Lines

Creating new simulation tracks through temporal branching



Waser, J.; Fuchs, R.; Ribičič, H.; Schindler, B.; Blöschl, G.; Gröller, E.; , "World Lines," *IEEE Transactions on Visualization and Computer Graphics,* vol.16, no.6, pp.1458-1467, Nov.-Dec. 2010



Shehzad Afzal, Ross Maciejewski, and David S. Ebert. Visual Analytics Decision Support Environment for Epidemic Modeling and Response Evaluation. IEEE Conference on Visual Analytics Science and Technology (VAST), 2011.

Design Principles

- Show familiar visual representations whenever possible
- Provide side-by-side comparisons of small multiple views
- Spatial position is strongest visual cue
- Multiple views are more effective when coordinated through explicit linking
- Avoid abrupt visual change

Control Chart Analysis

Temporal Visualization

Ira A. Fulton Schools of Engineering
Arizona State University

Objective

Objective

Apply methods of temporal analysis

Data Mining

- Data mining domain has techniques for examining time series. Looking for
  - patterns
  - anomalies
- Enhance the visualizations
  - show what is important
- Used in exploratory analysis
  - “I think this looks interesting, show me similar trends.”

Typical Time Series Analysis

- Trend analysis
  - A company's linear growth in sales over the years
- Seasonality
  - Sales are higher in summer than winter
- Forecasting
  - What is expected sales next quarter?

Control Chart Overview

Control Chart Components

- For temporal data, we can find anomalies using control chart methods
- Control charts consist of a statistic representing some measurement in time

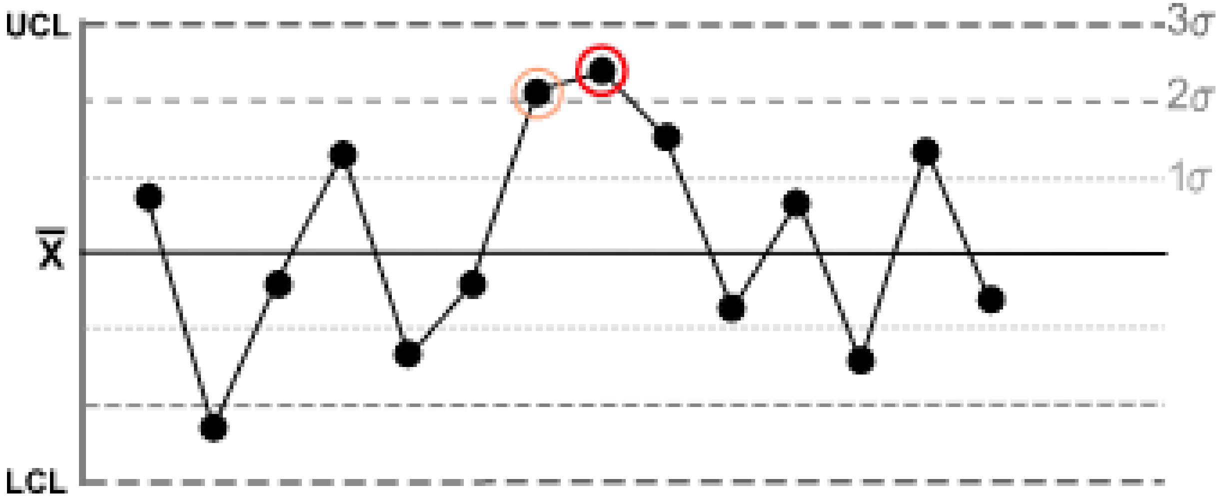
Calculations

- The mean and standard deviation of the statistic is calculated given all the available samples
- If the current value is greater than some pre-set number of standard deviations from the mean, then an alert is generated

What is a control chart?

- A graph used to study how a process changes over time
- Data are plotted in time order
- Always has a central line for average, an upper line for upper control limit and a lower line for lower control limit
- Lines are determined from historical data

Control Chart



When to use a control chart?

- Controlling ongoing processes by finding and correcting problems as they occur
- Predicting the expected range of outcomes from a process.
- Determining whether a process is stable (in statistical control).
- Analyzing patterns of process variation from special causes or common causes.
- Determining whether the quality improvement project should aim to prevent specific problems or to make fundamental changes to the process.

Control Chart Model

Upper Control Limit

$\mu + k\sigma$

Center Line

μ

Lower Control Limit

$\mu - k\sigma$

Moving Average/Range Charts

Moving Average Chart

- monitors the process location over time
- generally used for detecting small shifts in the process mean
- control limits are derived from average range on Range Chart

Range Chart

- monitors the process variation over time
- should be reviewed before Moving Average Chart

Moving Average: Stock Market Closing Example

Daily Closing Prices:

11,12,13,14,15,16,17

First day of 5-day SMA:

(11 + 12 + 13 + 14 + 15) / 5 = 13

Second day of 5-day SMA:

(12 + 13 + 14 + 15 + 16) / 5 = 14

Third day of 5-day SMA:

(13 + 14 + 15 + 16 + 17) / 5 = 15

Exponentially Weighted Moving Average

SMA:

10 period sum / 10

Multiplier:

(2 / (Time periods + 1)) = (2 / (10 + 1)) = 0.1818 (18.18%)

EMA:

{Close - EMA(previous day)} x multiplier + EMA(previous day)

The Lag Factor

Shorter Moving

- nimble and quick to change

Longer Lag

- Longer the moving average, more the lag

Longer Moving

- Longer moving - slow to change

Differences between simple moving averages and exponential moving averages, one is not necessarily better than the other

Length of your moving average depends on your analytical goal

Sequences and Time Series

Introduction to Sequences

K. Selçuk Candan, Professor of Computer Science and Engineering

CASCADE
ASU Center for Assured and SCAlable Data Engineering

Ira A. Fulton Schools of Engineering
Arizona State University

Strings, sequences, time series

- A *string* or *sequence*, $S = (c_1, c_2, \ldots, c_N)$, is a finite sequence of symbols.

abcbbbaabbaabcbbbaaabbc

- A *time series*, $T = (d_1, d_2, \ldots, d_N)$, is a finite sequence of data values.

Interest in the term “big data”

https://trends.google.com/trends/explore?date=2008-12-19%202018-01-19&q=big%20data

String/sequence matching and search

- Prefix search:
  - Find all strings that start with “tab”:
    - “table”; “tabular”; “tablet”; ….

- Subsequence search:
  - Find all strings that contain the subsequence “ark”:
    - “marketing”; “spark”; “quark”
  - Find all occurrences of “acd”:
    - “aabacdcdabdcababdacddcab.”

- Sequence similarity:
  - “table” vs. “cable”?
  - “table” vs. “tale”?
  - “table” vs. “tackle”?

String/sequence matching and search

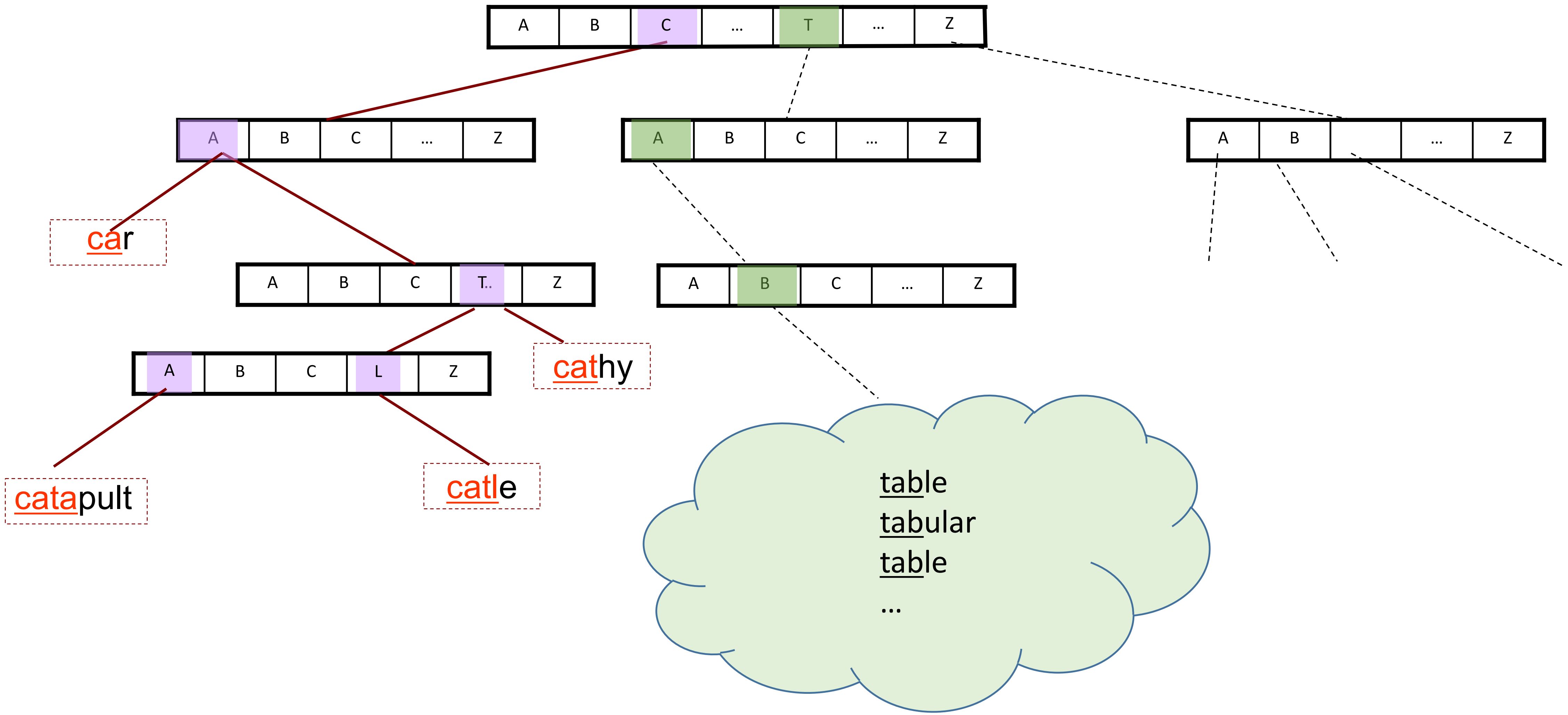
- Prefix search:
  - Find all strings that start with “tab”:
    - “<u>tab</u>le”; “<u>tab</u>ular”; “<u>tab</u>let”; ….

- Subsequence search:
  - Find all strings that contain the subsequence “ark”:
    - “m<u>ark</u>eting”; “sp<u>ark</u>”; “qu<u>ark</u>”

  - Find all occurrences of “acd”:
    - “aab<u>acd</u>cdabdcababd<u>acd</u>dcab.”

- Sequence similarity:
  - “table” vs. “cable”?
  - “table” vs. “tale”?
  - “table” vs. “tackle”?

Trie data structure



String/sequence matching and search

- Prefix search:
  - Find all strings that start with "tab":
    - "table"; "tabular"; "tablet"; ….
- Subsequence search:
  - Find all strings that contain the subsequence "ark":
    - "marketing"; "spark"; "quark"
  - Find all occurrences of "acd":
    - "aabacdcdabdcababdacddcab."
- Sequence similarity:
  - "table" vs. "cable"?
  - "table" vs. "tale"?
  - "table" vs. "tackle"?

Subsequence/Pattern Search

data: abcbbbaabbaabcbbbaaabbccbbbaabbaacbbbaabbccbcbbbaabcbbbaabab

pattern: abbcc

- Brute force approach:
  - scan the sequence, while aligning the pattern for each position in the sequence
- Given a sequence of length N, and pattern of length M
  - Cost: O(NxM)
  - For the above example, cost: 60 x 5

Suffix Trees and Arrays

- Tries work well if we search for a prefix

- Suffix trees and suffix arrays
  - Input text: a single long string
  - each position in the text gives a suffix

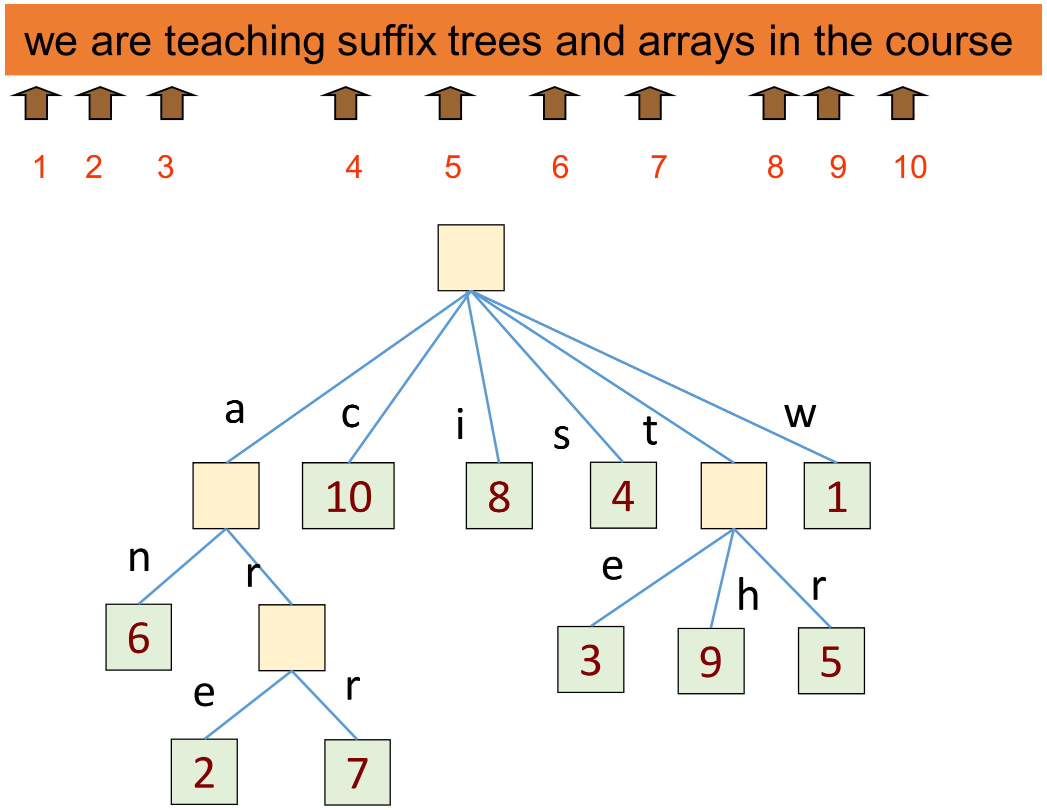
we are teaching suffix trees and arrays in the course

  - alternatively, start of each word in the text gives a suffix

we are teaching suffix trees and arrays in the course

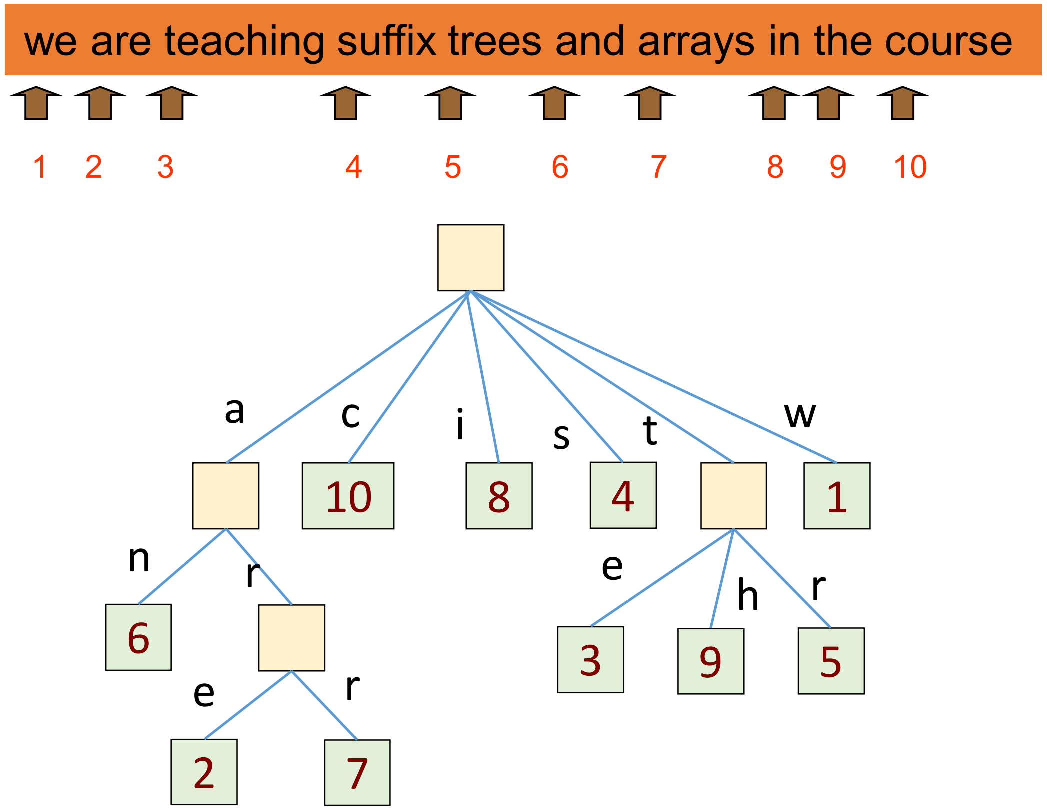
Suffix Trees

- Suffix trees
  - Input text: a single long string
  - each word position in the text gives a suffix



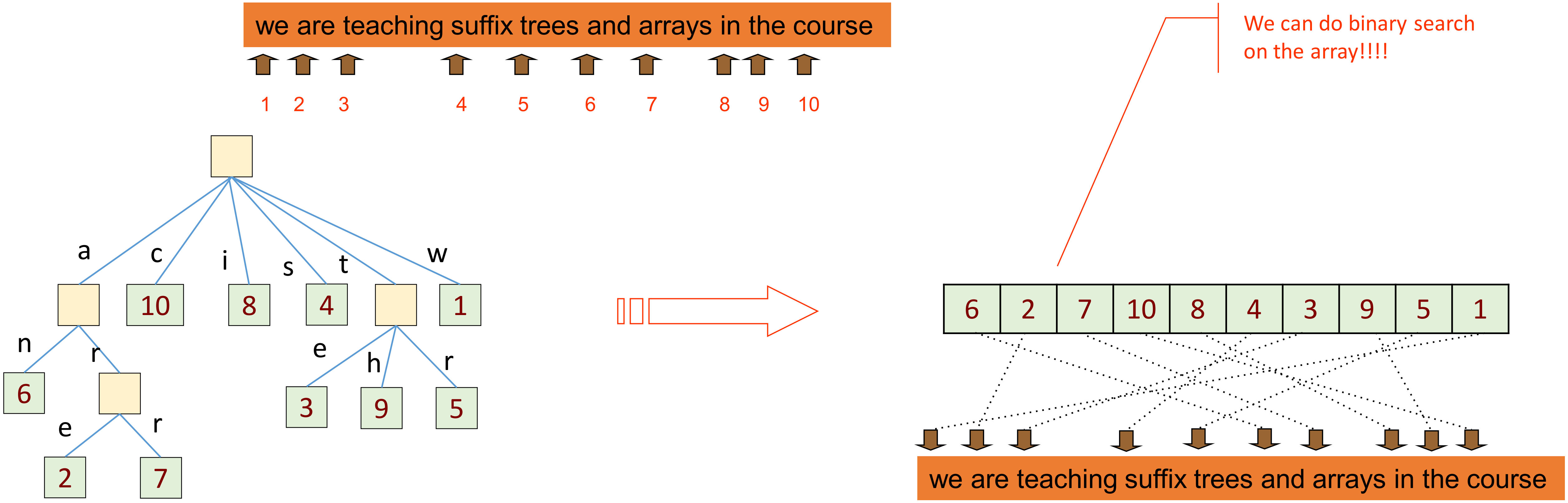
Suffix Trees

- Suffix trees
  - Input text: a single long string
  - each word position in the text gives a suffix



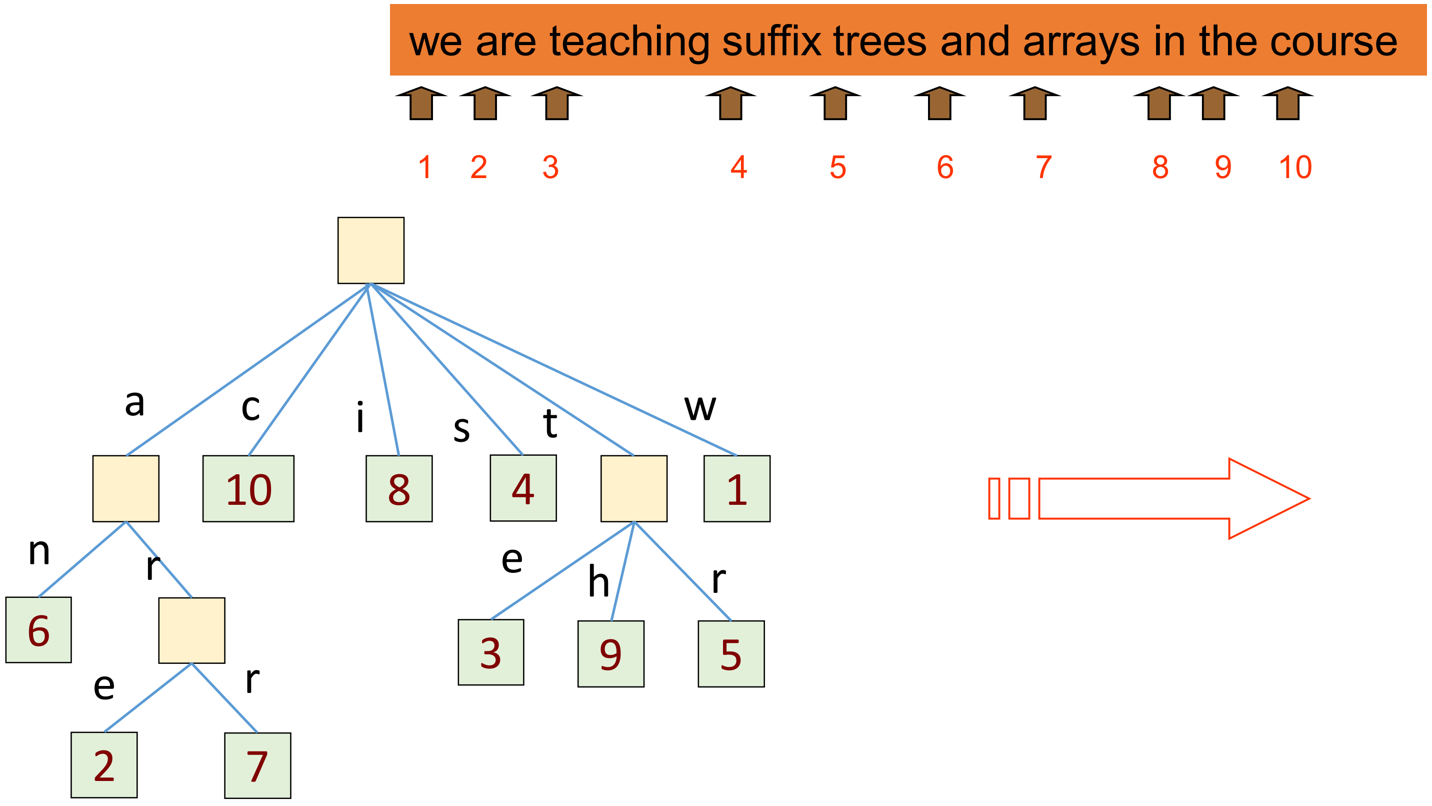
Suffix Arrays

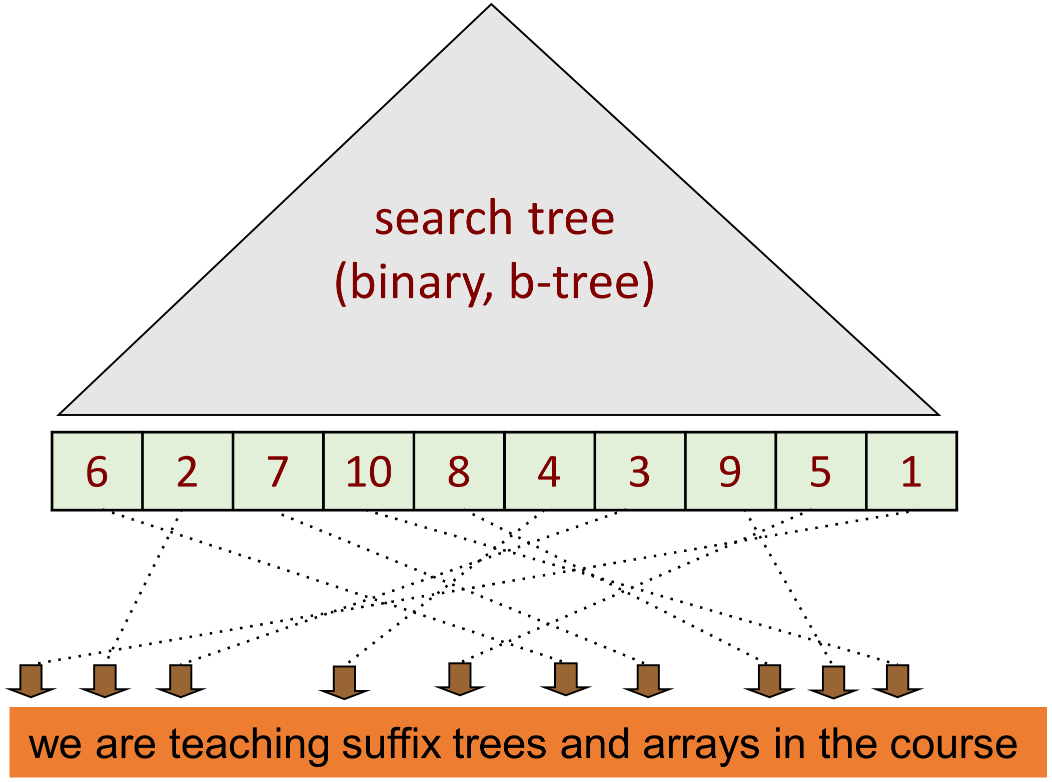
- Suffix arrays
  - Input text: a single long string
  - each word position in the text gives a suffix



Suffix Arrays

- Suffix arrays
  - Input text: a single long string
  - each word position in the text gives a suffix





Subsequence/Pattern Search

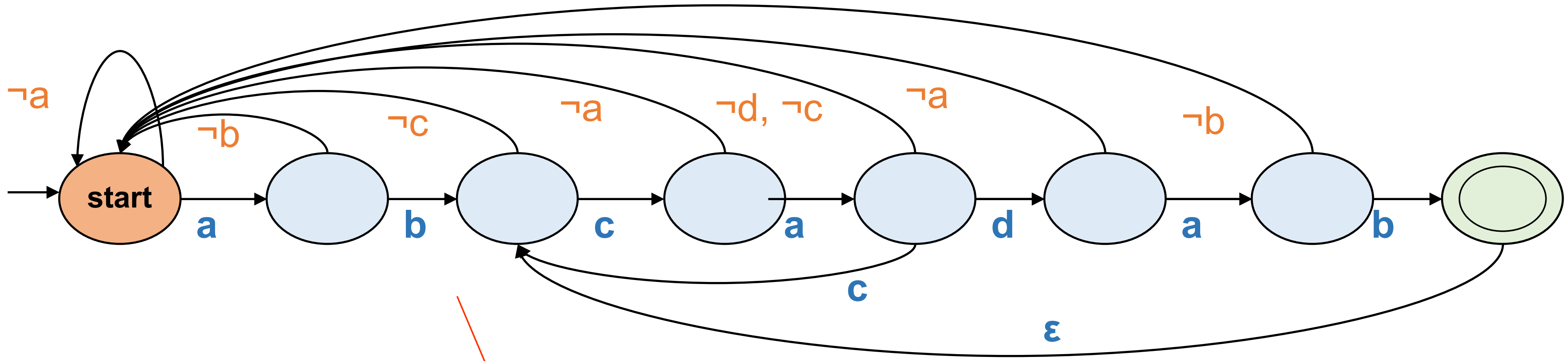
- What if we are not given the sequence in advance; can we do search without a data structure on the sequence?

  - Yes, scan the sequence…
  - ..but, we have seen that this is expensive
    - Given a sequence of length N, and pattern of length M
    - Cost: O(NxM)

- If we are given the pattern in advance, can we create a data structure on the pattern, instead?

Knuth-Morris-Pratt (KMP)

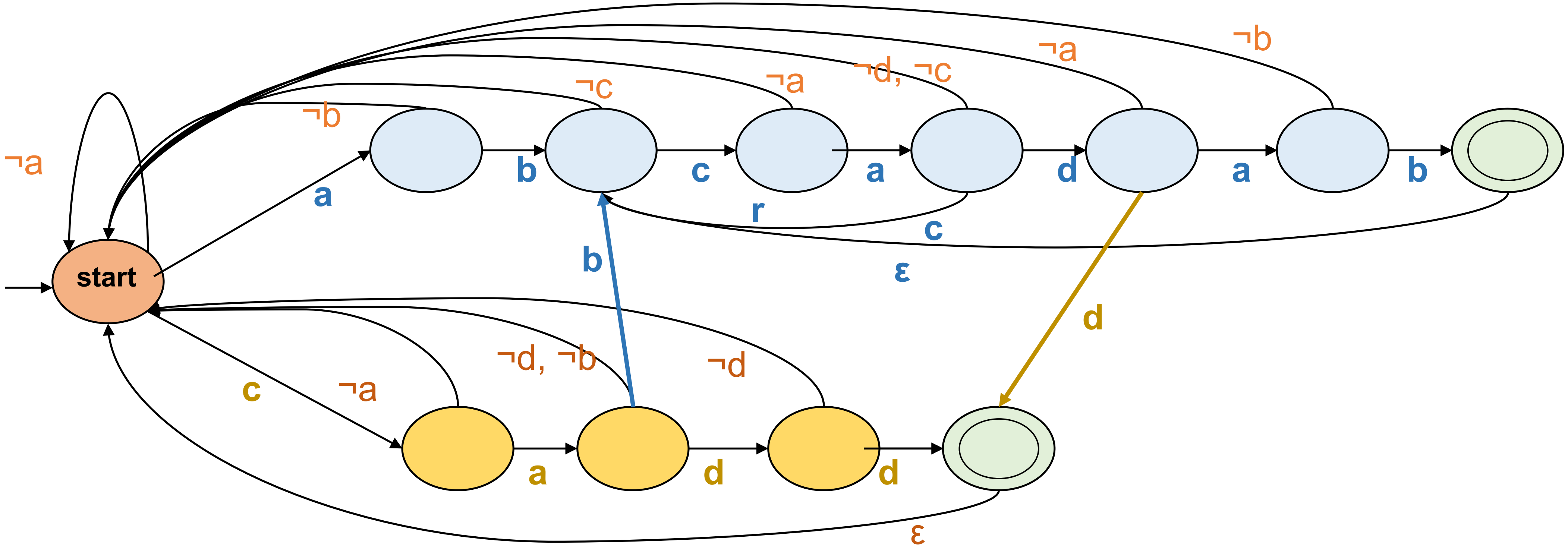
- Given a sequence of length N, and pattern of length M
- Knuth-Morris-Pratt: O(N)
- Example
  - Pattern: abcadab



A data structure on the query!!!!

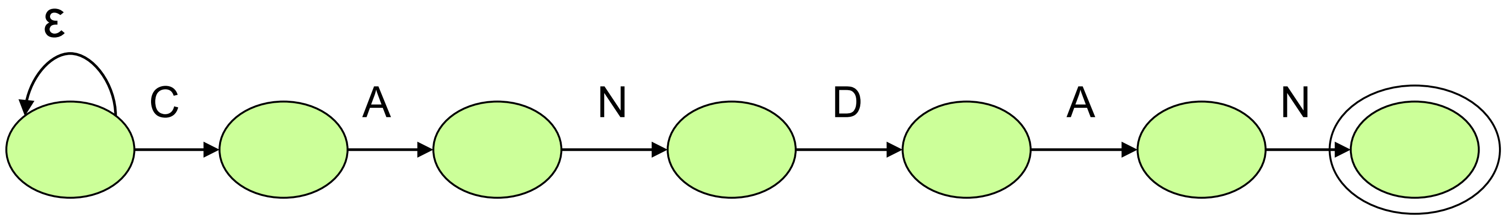
Aho-Corasick Trie

- What is we are given multiple patterns to search simultaneously
- Given a sequence of length N, and patterns of length M_1 and M_2
- Aho-Corasick Trie: O(N)
- Example
  - Patterns: abcadab; cadd

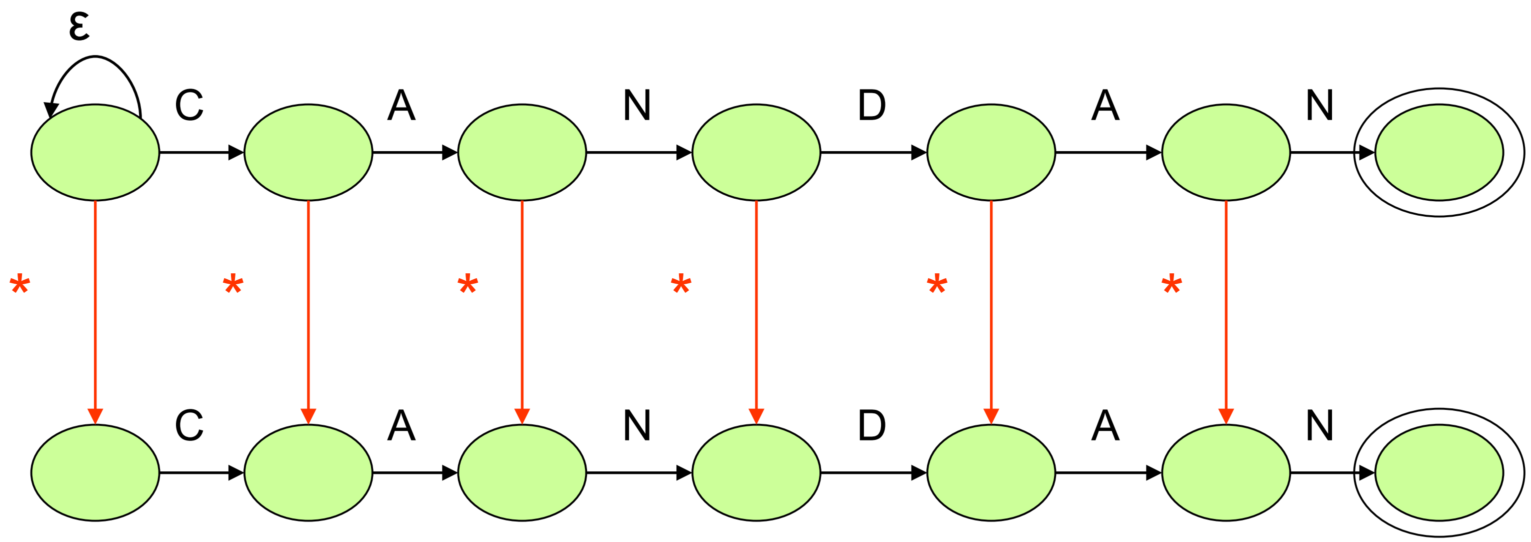


What about approximate matches?

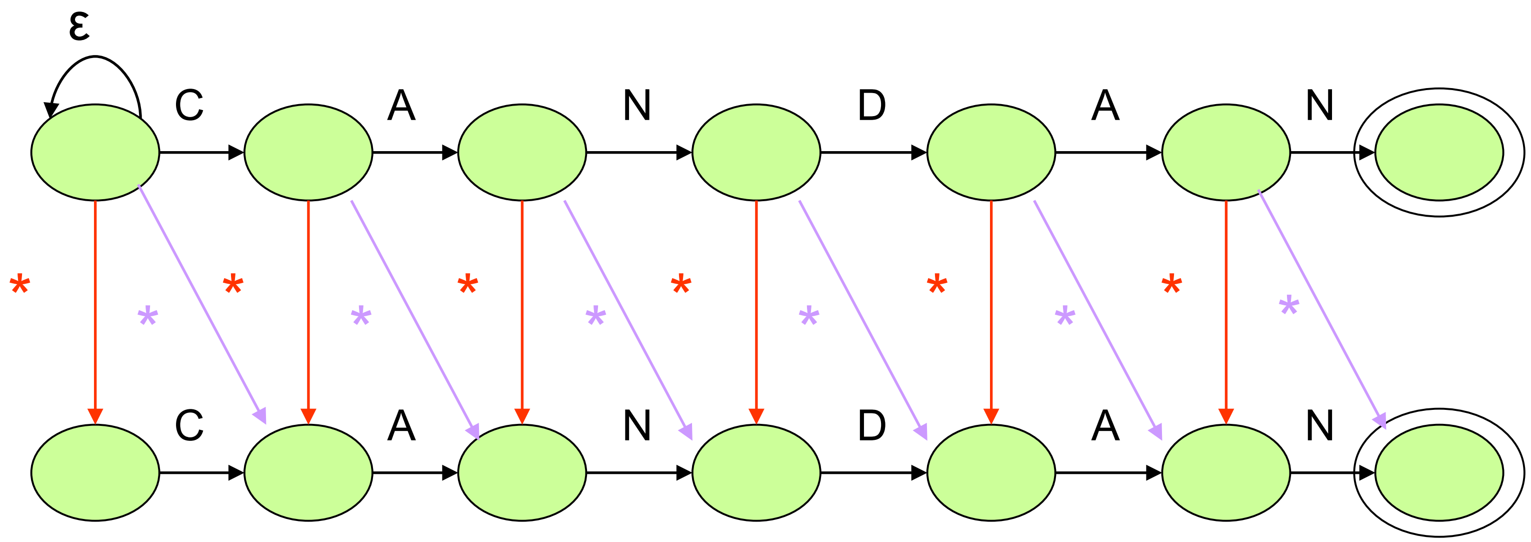
- Example
  - Pattern: CANDAN



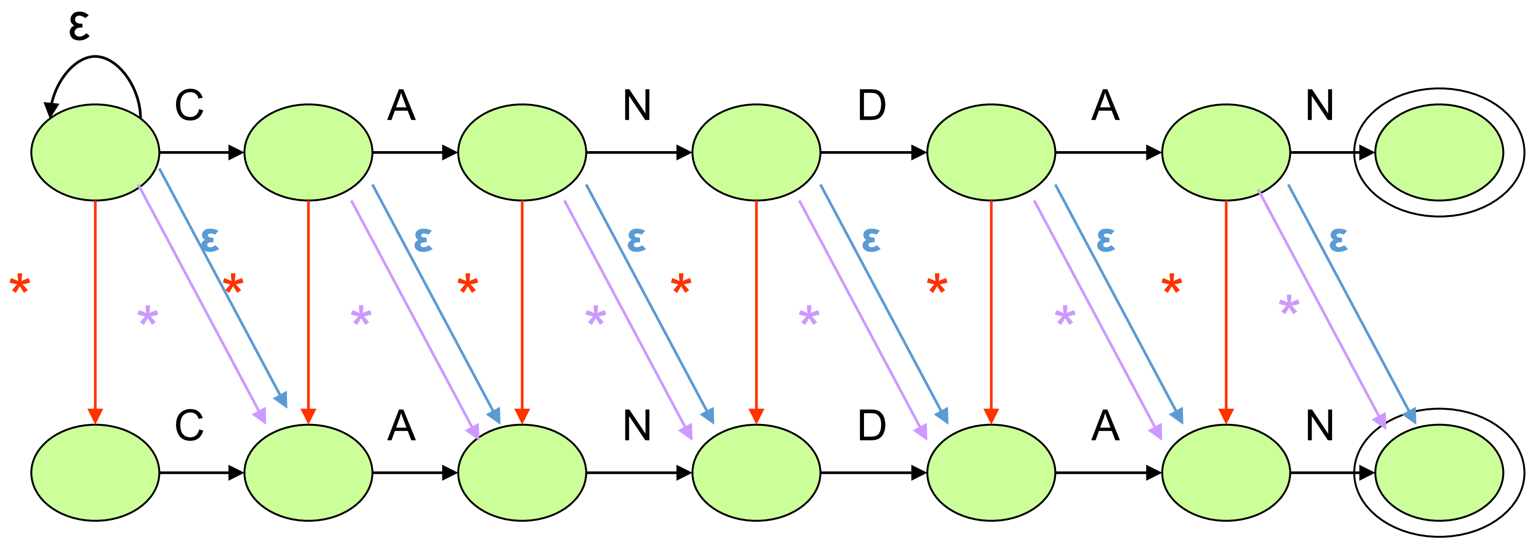
NFA…upto 1 insertion



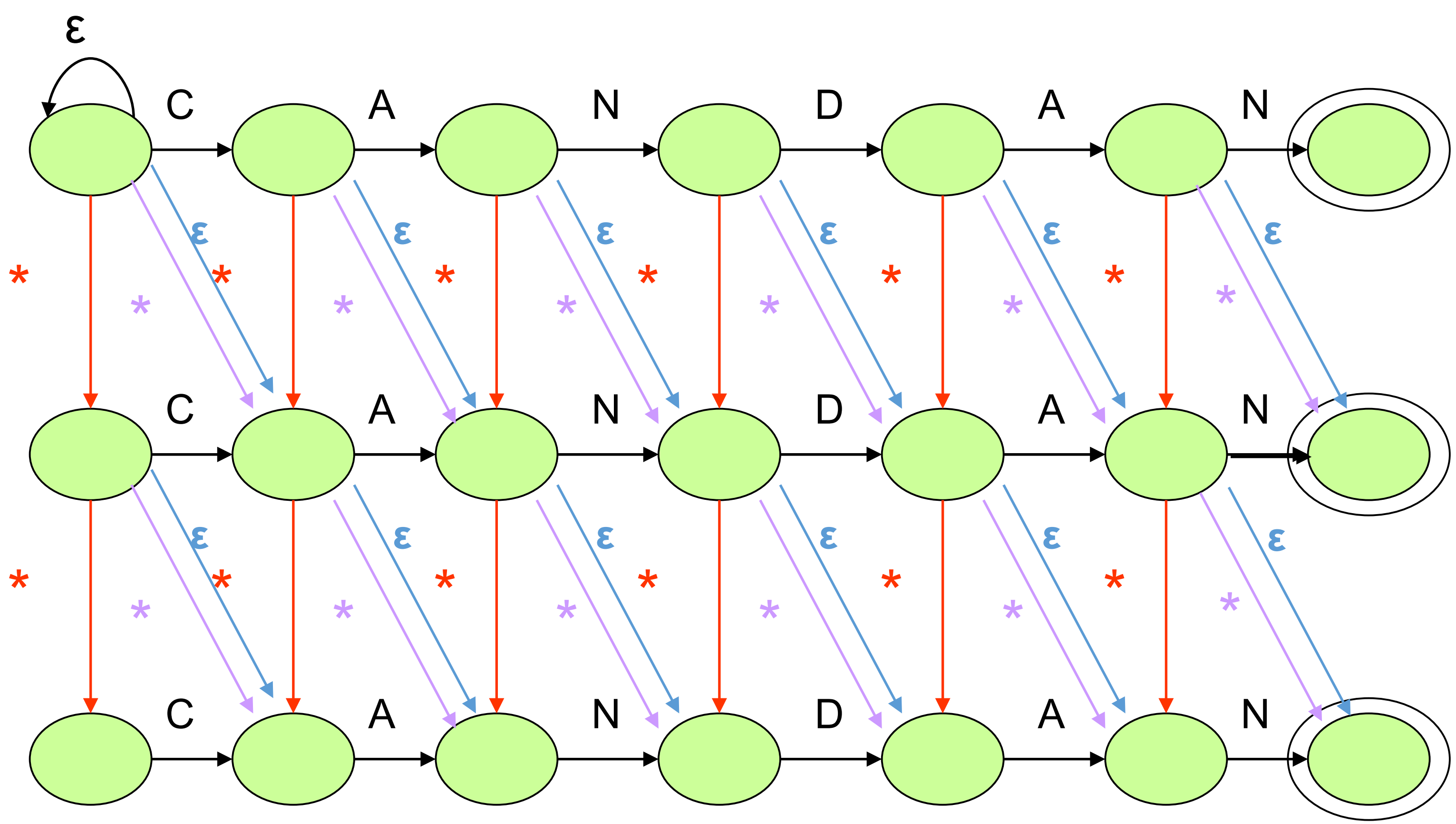
NFA…upto 1 insertion\replacement



NFA…upto 1 ins.\rep.\deletion



NFA…upto 2 ins.\rep.\deletion



Summary

- Prefix based sequence exploration:
  - Trie data structure helps prune the candidate set
- Subsequence search and exploration
  - Suffix trees and suffix arrays helps focus on the part of a long sequence
- Pattern matching
  - Non-deterministic finite automata can be used to support exact and approximate pattern matching

Sequences and Time Series

Edit Distance

K. Selçuk Candan, Professor of Computer Science and Engineering

CASCADE
ASU Center for Assured and SCAlable Data Engineering

Ira A. Fulton Schools of Engineering
Arizona State University

Strings, sequences, time series

| A *string* or *sequence*, $S = (c_1, c_2, \ldots, c_N)$, *is a finite sequence of symbols.*

abcbbbaabbaabcbbbaaabbc

- Prefix search:
  - Find all strings that start with "tab":
    - "<u>tab</u>le"; "<u>tab</u>ular"; "<u>tab</u>let"; ….
- Subsequence search:
  - Find all strings that contain the subsequence "ark":
    - "m<u>ark</u>eting"; "sp<u>ark</u>"; "qu<u>ark</u>"
  - Find all occurrences of "acd":
    - "aab<u>acd</u>cdabdcababd<u>acd</u>dcab."
- Sequence similarity:
  - "table" vs. "cable"?
  - "table" vs. "tale"?
  - "table" vs. "tackle"?

Approximate string match

- Sequence distance/similarity:
  - “table” vs. “cable”?
  - “table” vs. “bale”?

- Edit distance:
  - “table” vs. “cable”: 1 (replace “t” with “c”)
  - “table” vs. “bale”: 3 (delete “t”; replace “a” and “b ”; replace “b” and “a”)
- Common edit operations
  - Replacement:
  - a ->b
  - Deletion:
  - a ->λ
  - Insertion:
  - λ ->a

Edit cost

- Let E be a sequence of edit operations to convert one string to another
- Let us associate a cost, C, to each edit operation

> - Costs of edit operations can be different from each other
>   - Type of the operation (replace, delete, insert)
>   - Symbols involved in the operation
>   - Position of the edit operation

- Given a sequence of edit operations, E

$$C(E) = \sum_{e_i \in E} C(e_i)$$

- Edit Distance:

$$D(String_1, String_2) = \min_{E \text{ takes String}_1 \text{ to String}_2} \{C(E)\}$$

Edit distance

- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

D[i,j] = # of edits from length-i prefix of P to length-j prefix of Q

Edit distance

- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

D[i,j] = # of edits from length-i prefix of P to length-j prefix of Q

- D[0,j] = j

P | empty

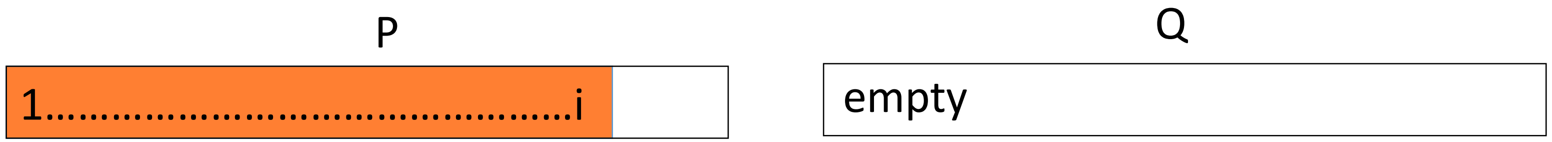
Q | 1 ……………………………………..j

Edit distance

- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

D[i,j] = # of edits from length-i prefix of P to length-j prefix of Q

- D[0,j] = j
- D[i,0] = i



Edit distance

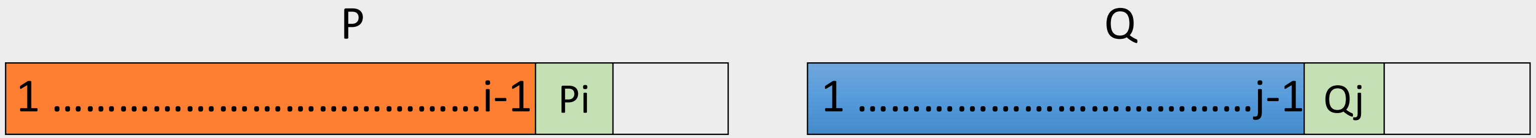
- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

D[i,j] = # of edits from length-i prefix of P to length-j prefix of Q

- D[0,j] = j
- D[i,0] = i

for all (i<=N) and (j <= M)

- if(P_i=Q_j)
  - D[i,j] = D[i-1,j-1]



Edit distance

- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

D[i,j] = # of edits from length-i prefix of P to length-j prefix of Q

- D[0,j] = j
- D[i,0] = i
- if(P_i=Q_j) D[i,j] = D[i-1,j-1]

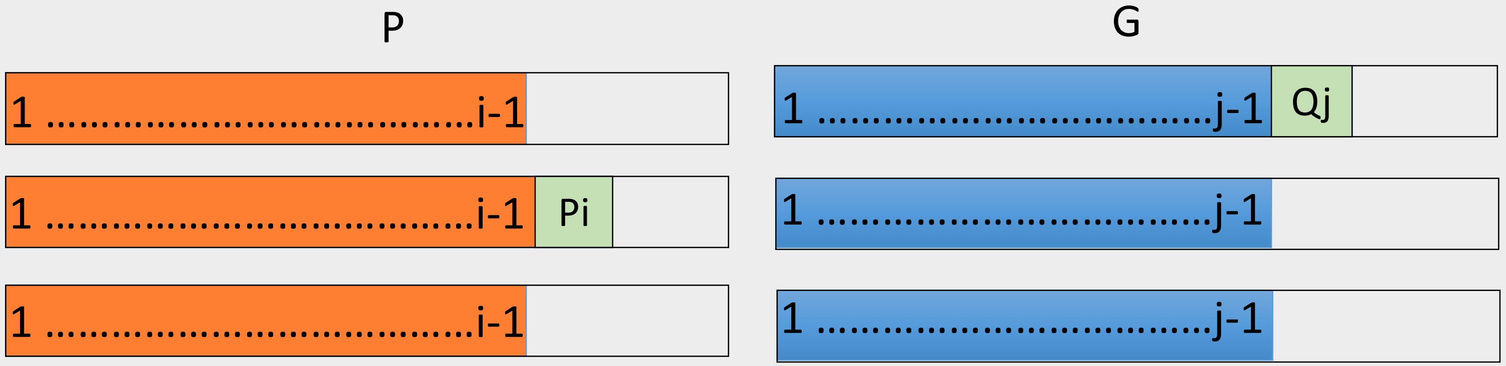
for all (i<=N) and (j <= M)

else D[i,j] = 1 + min{

insert Qj D[i-1,j] ,

delete Pi D[i,j-1] ,

replace Pi with Qj D[i-1,j-1] }



Edit distance

- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

D[i,j] = Cost of edits from length-i prefix of P to length-j prefix of Q

- D[-1,j] = infinity; D[i,-1] = infinity
- D[0,0] = 0
- if$(P_i=Q_j)$ D[i,j] = D[i-1,j-1] for all (i<=N) and (j <= M)

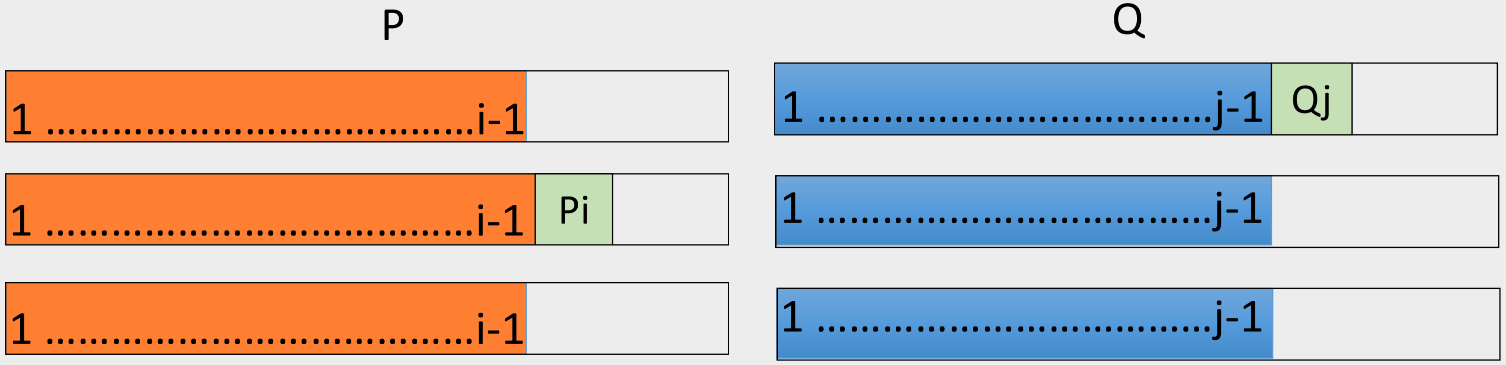
else D[i,j] = min{

insert Gj $C_{ins}(Q_j)$ + D[i-1,j] ,

delete Pi $C_{del}(P_i)$ + D[i,j-1] ,

replace Pi with Gj $C_{rep}(P_i, Q_j)$ + D[i-1,j-1]

}



Edit distance

- Let us be given two strings, P and Q, of lengths N and M
- Let us assume that all edit operations have cost = 1

O(N*M)

D[i,j] = Cost of edits from length-i prefix of P to length-j prefix of Q

- D[-1,j] = infinity; D[i,-1] = infinity
- D[0,0] = 0
- if(P_i=Q_j) D[i,j] = D[i-1,j-1] for all (i<=N) and (j <= M)

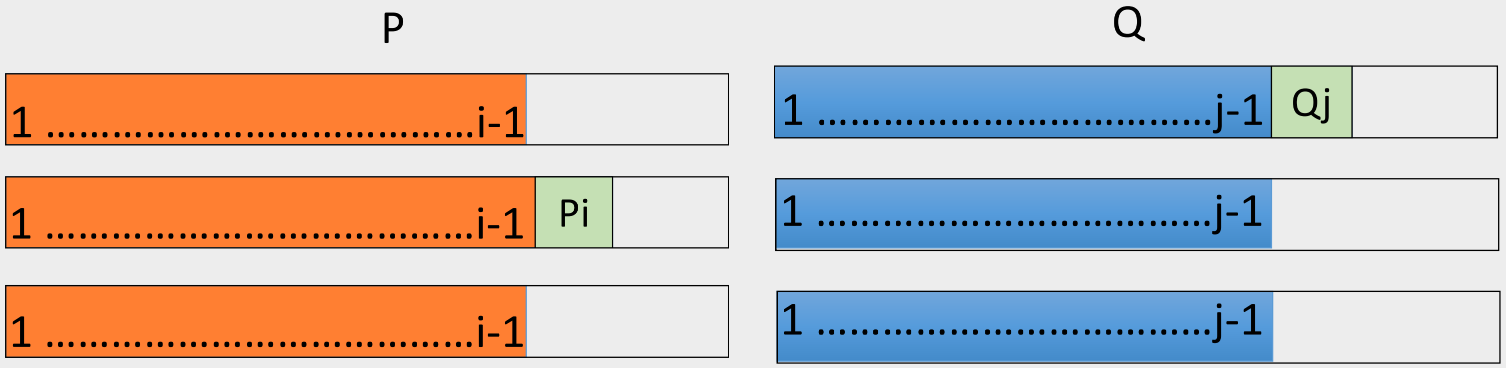
else D[i,j] = min{

insert Gj $C_{ins}(Q_j)$ + D[i-1,j] ,

delete Pi $C_{del}(P_i)$ + D[i,j-1] ,

replace Pi with Gj $C_{rep}(P_i, Q_j)$ + D[i-1,j-1]

}



Summary

- Edit distance can be used to assess how similar or different two strings are

- Problem: Edit distance can be costly for matching long strings.

Sequences and Time Series

W-Grams and Other Approaches to Filtering

K. Selçuk Candan, Professor of Computer Science and Engineering

CASCADE
ASU Center for Assured and SCAlable Data Engineering

Ira A. Fulton Schools of Engineering
Arizona State University

Strings, sequences, time series

| A *string* or *sequence*, $S = (c_1, c_2, \ldots, c_N)$, *is a finite sequence of symbols.*

abcbbbaabbaabcbbbaaabbc

- Prefix search:
  - Find all strings that start with "tab":
    - "table"; "tabular"; "tablet"; ....
- Subsequence search:
  - Find all strings that contain the subsequence "ark":
    - "marketing"; "spark"; "quark"
  - Find all occurrences of "acd":
    - "aabacdcdabdcababdacddcab."
- Sequence similarity:
  - "table" vs. "cable"?
  - "table" vs. "tale"?
  - "table" vs. "tackle"?

Reminder: Edit Distance

- Let P (of size N) and Q (of size M) be two sequences.

- Given a sequence of edit operations, E

$$C(E) = \sum_{e_i \in E} C(e_i)$$

- Edit distance

$$D(String_1, String_2) = \min_{E \text{ takes } \mathrm{String}_1 \text{ to } \mathrm{String}_2} \{C(E)\}$$

- Cost: O(N*M)

Cross-parsing Distance

- Let P (of size N) and Q (of size M) be two sequences
- c(P|Q): cost of parsing P with respect to Q
    1. Find the longest (possibly empty) prefix of P that appears as a string somewhere in Q
    2. Restart from the very next position in P and continue until P is completely parsed.

Cross-parsing Distance

- Let P (of size N) and Q (of size M) be two sequences
- c(P|Q): cost of parsing P with respect to Q
  1. Find the longest (possibly empty) prefix of P that appears as a string somewhere in Q
  2. Restart from the very next position in P and continue until P is completely parsed.

$$D_{crossparse}(P,Q) = \frac{\left(c(P\,|\,Q)-1\right)+\left(c(Q\,|\,P)-1\right)}{2}$$

- Linear time if the strings are indexed using a suffix tree

Compression Distance

- C(P) is the compressed size of P,
- C(Q) is the compressed size of Q, and
- C(P:Q) is the compressed size of the sequence obtained by concatenating P and Q.

$$D_{NCD}(P,Q) = \frac{C(P:Q) - \min\{C(P),C(Q)\}}{\max\{C(P),C(Q)\}}$$

Filtering

- Given a string P of (length N) and a pattern Q (of length M), determine whether the string P may contain an approximate match to Q

Filtering

- Given a string P of (length N) and a pattern Q (of length M), determine whether the string P may contain an approximate match to Q with at most k errors

- Approach 1: Given a maximum error rate, k,
  - cut the pattern Q into k + 1 pieces
  - verify that at least one piece of Q exists in P exactly

  This is because k errors cannot affect more than k pieces.

Filtering

- Given a string P of (length N) and a pattern Q (of length M), determine whether the string P may contain an approximate match to Q with at most k errors

- Approach 2: Given a maximum error rate, k,
  - slide a window of length M over the string P and count the number of symbols that are included in the pattern Q
  - only windows that have at least M – k matching symbols need to be considered

Fingerprinting with w-grams

- **w-grams:** Given a sequence P, its w-grams are obtained by sliding a window of length w over P.

Common w-gram counting

- Given a string P of (length N) and a string Q (of length M), determine whether the two strings may match each other with at most k errors
  - **w-grams:** Given a sequence P, its w-grams are obtained by sliding a window of length w over P.

- Approach (common w-gram counting):
  - Identify $(M - w + 1)$ w-grams of the query string Q
  - Each mismatch between Q and P can affect w many w-grams
    - given an upper bound of k errors, at least $(M - w + 1 - kw)$ w-grams must match
  - Search for these matches using a suffix tree (in linear time)

String kernels

- Given a string P of (length N) and a string Q (of length M), determine whether the strings <u>may approximately match each other</u>
  - **w-grams:** Given a sequence P, its w-grams are obtained by sliding a window of length w over P.

- <u>Approach (string kernels):</u>
  - Identify all w-grams of the query sequence Q; create a counting vector, q
  - Identify all w-grams of the data sequence P; create a counting vector, p
  - Measure the (dot product) similarity of the two counting vectors
    - If the dot product similarity is low, then P is not likely to match Q

Min-sampling similarity

- Given a string P of (length N) and a string Q (of length M), determine whether the strings may approximately match each other
  - **w-grams:** Given a sequence P, its w-grams are obtained by sliding a window of length w over P.
- Approach (min-sampling similarity):
  - Consider r random hash orders of the w-grams of P and Q
  - For each order o_i
    - Find the smallest B w-grams of string P based on the chosen order
    - Find the smallest B w-grams of string Q based on the chosen order
    - If they agree, $match_i(P, Q) = 1$
  - After computing the match for all r orders

$$sim(P,Q) = \frac{\sum_{o_i} match_i(P,Q)}{r}$$

Summary

- Edit distance can be costly for matching long strings
  - Cross-Parsing and Compression-Distance can be used to approximate the edits distance comparison

- W-grams (more commonly known as n-grams) can be used to help filter unpromising candidates before more costly distance computations

Sequences and Time Series

Time Series Modeling

K. Selçuk Candan, Professor of Computer Science and Engineering

CASCADE
ASU Center for Assured and SCAlable Data Engineering

Ira A. Fulton Schools of Engineering
Arizona State University

Strings, sequences, time series

- A *string* or *sequence*, $S = (c_1, c_2, \ldots, c_N)$, is a finite sequence of symbols.

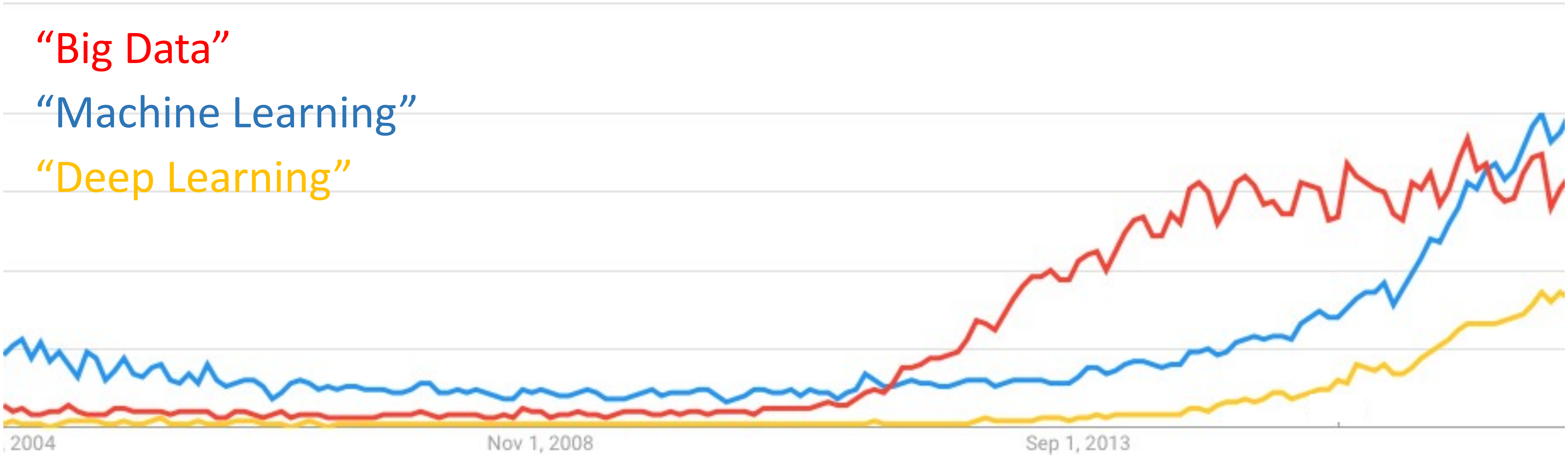
abcbbbaabbaabcbbbaaabbc

- A *time series*, $T = (d_1, d_2, \ldots, d_N)$, is a finite sequence of data values.

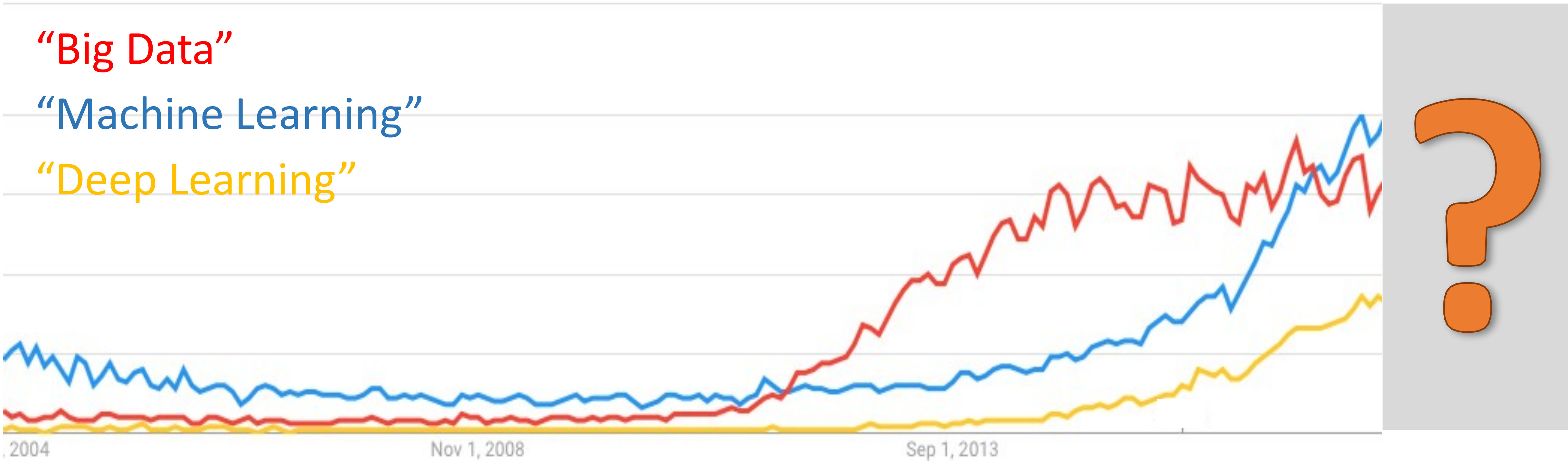
Interest in the term “big data”

https://trends.google.com/trends/explore?date=2008-12-19%202018-01-19&q=big%20data

Comparing time series



Forecasting time series

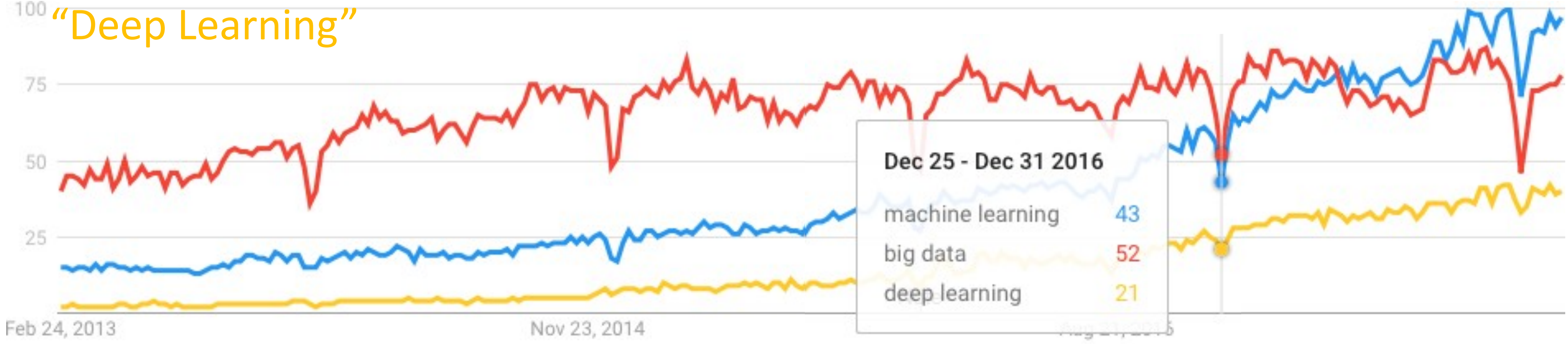


Motifs

“Big Data”

“Machine Learning”

“Deep Learning”



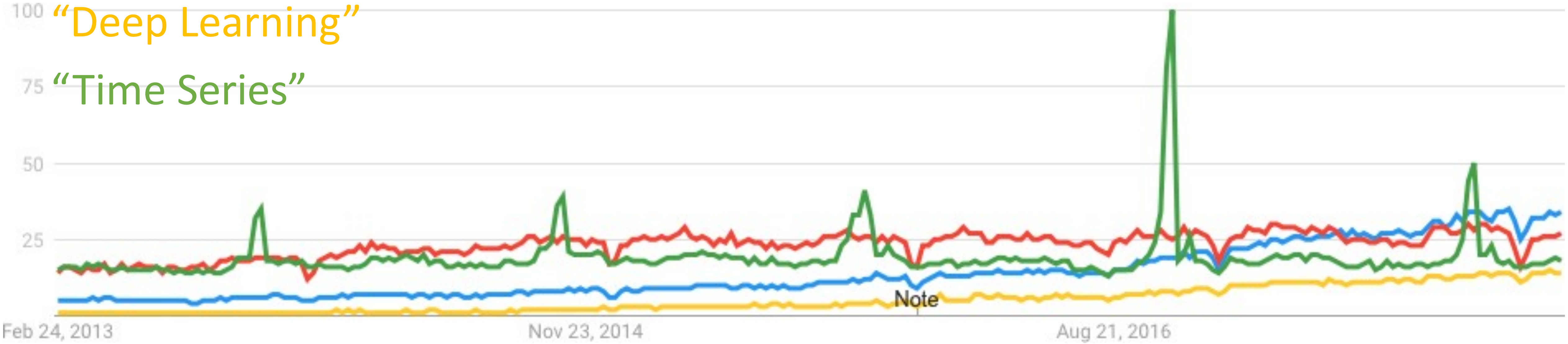
Motifs

“Big Data”

“Machine Learning”

“Deep Learning”

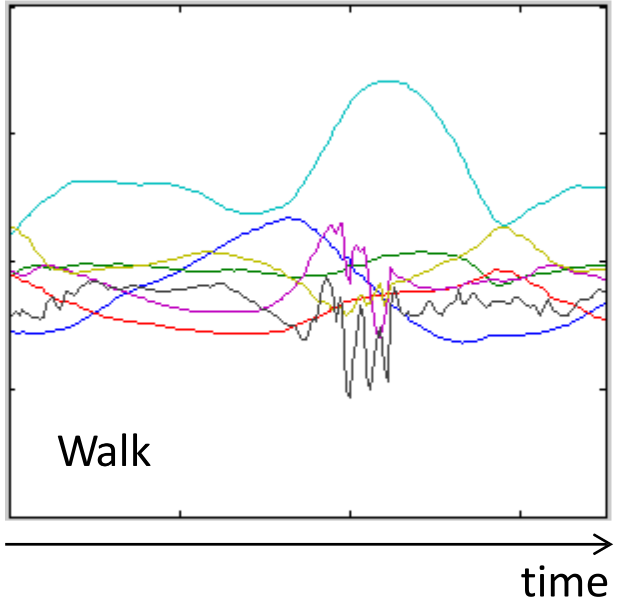
“Time Series”

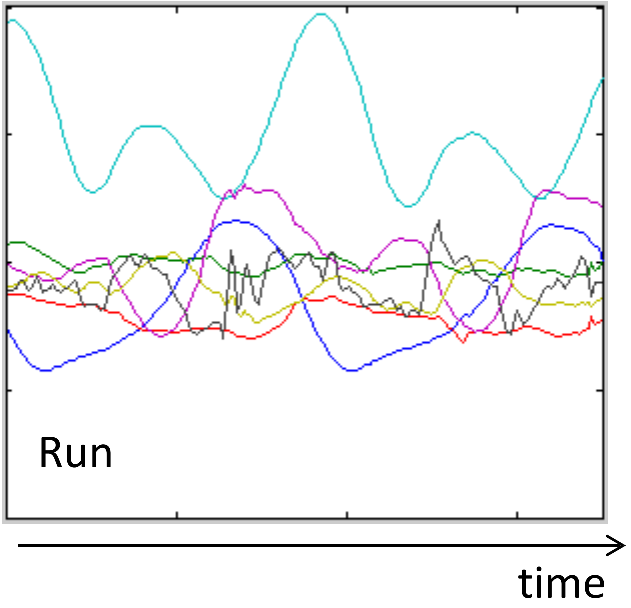


Classification

Markers on human body [1]

Multi-variate time series tracking 7 markers on subject's left leg





1. https://en.wikipedia.org/wiki/Motion_capture

Time Series Models

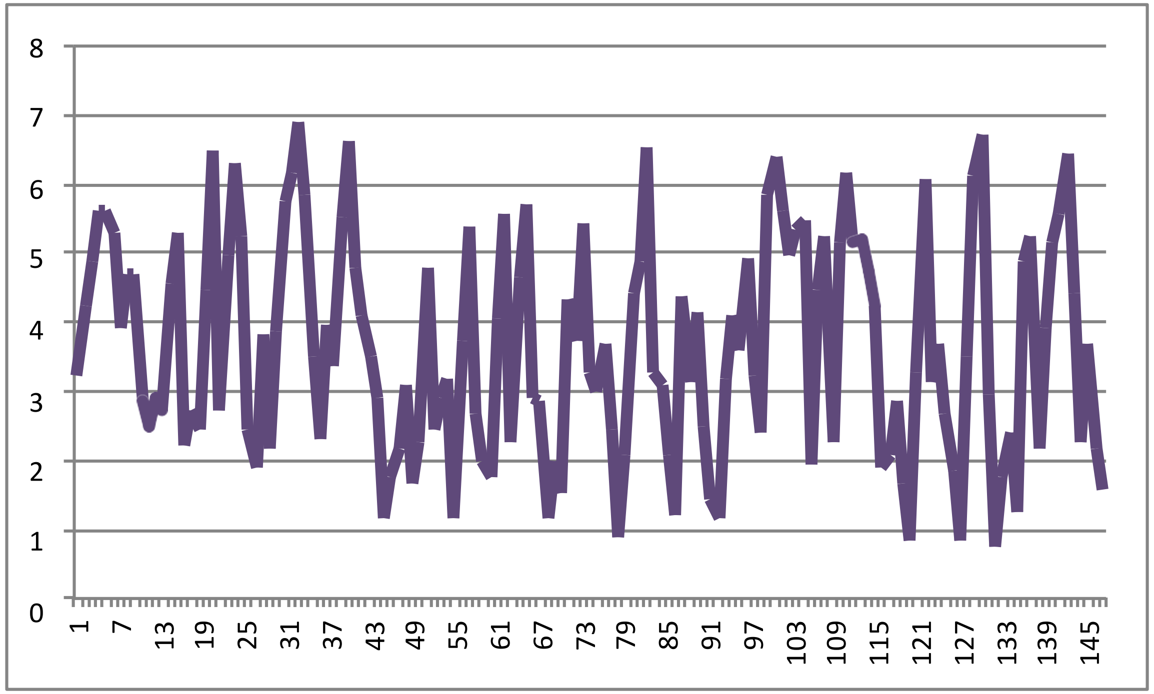
- Question: Can we discover a closed form formula (or a model) that describes the time series?

- Simpler Question: Can we characterize high-level properties of a given time series?

Types of Time Series

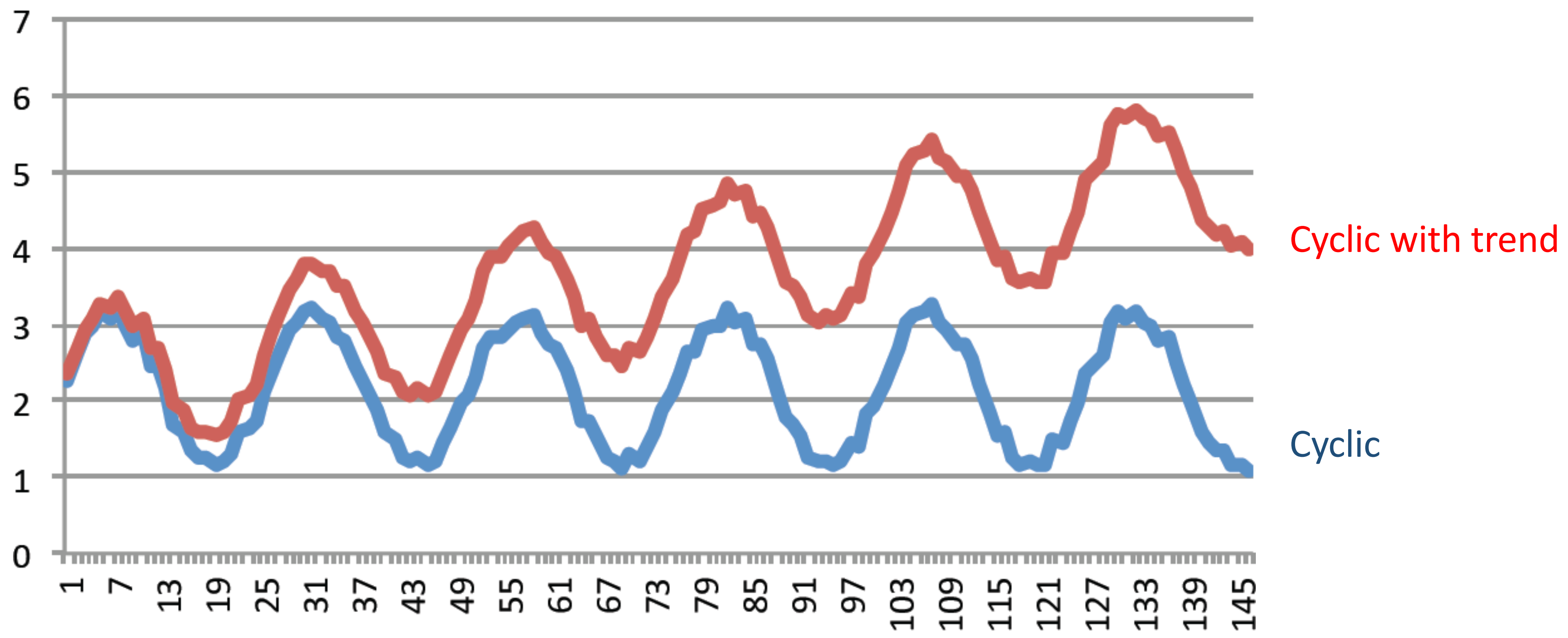
Stationary Series

- Stationary series
  - statistical properties, such as mean and variance, are constant over time.



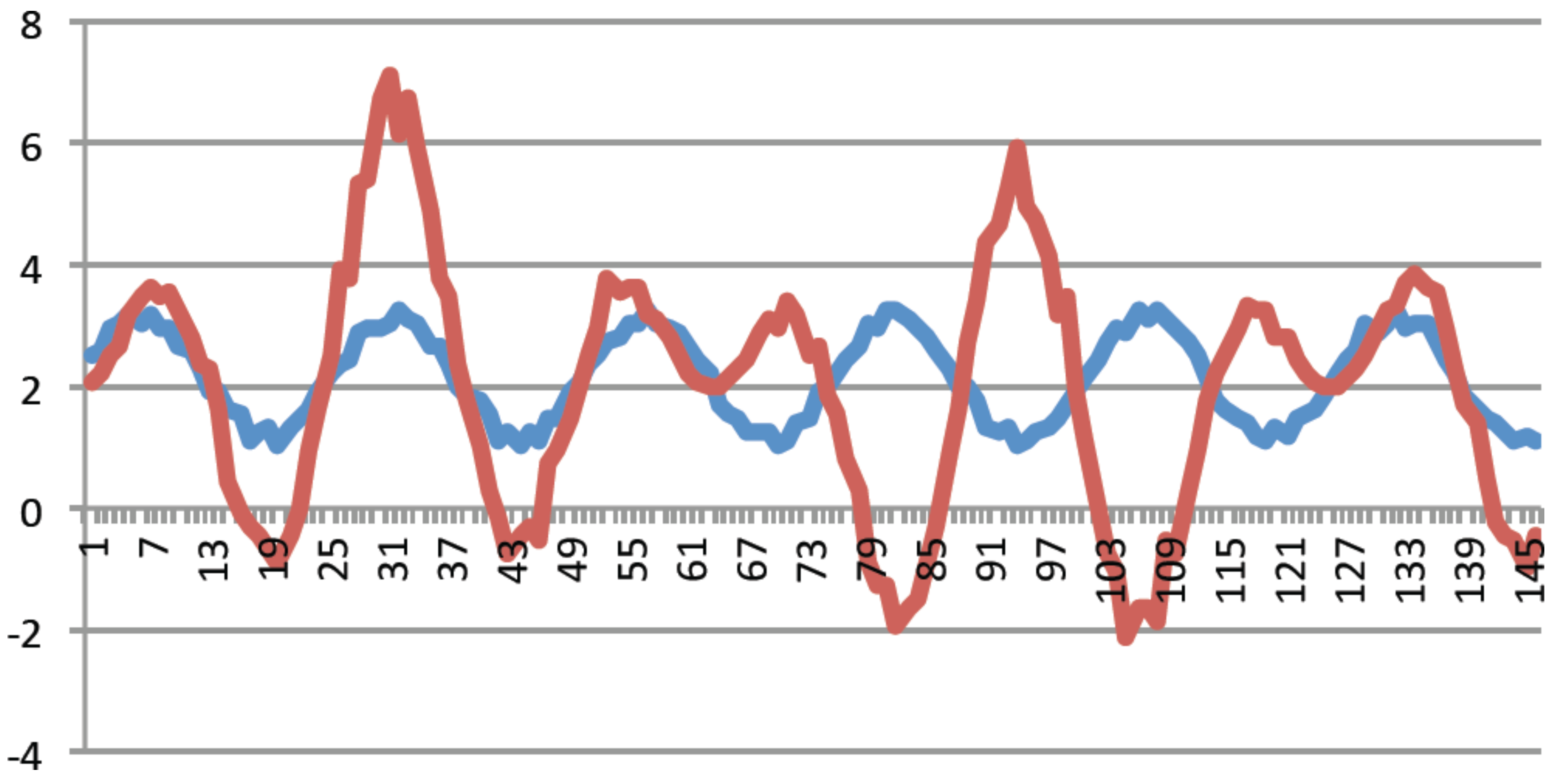
Non-Stationary Series

- Non-stationary series
  - statistical properties change over time



Non-Stationary Series

- Non-stationary series
  - statistical properties change over time

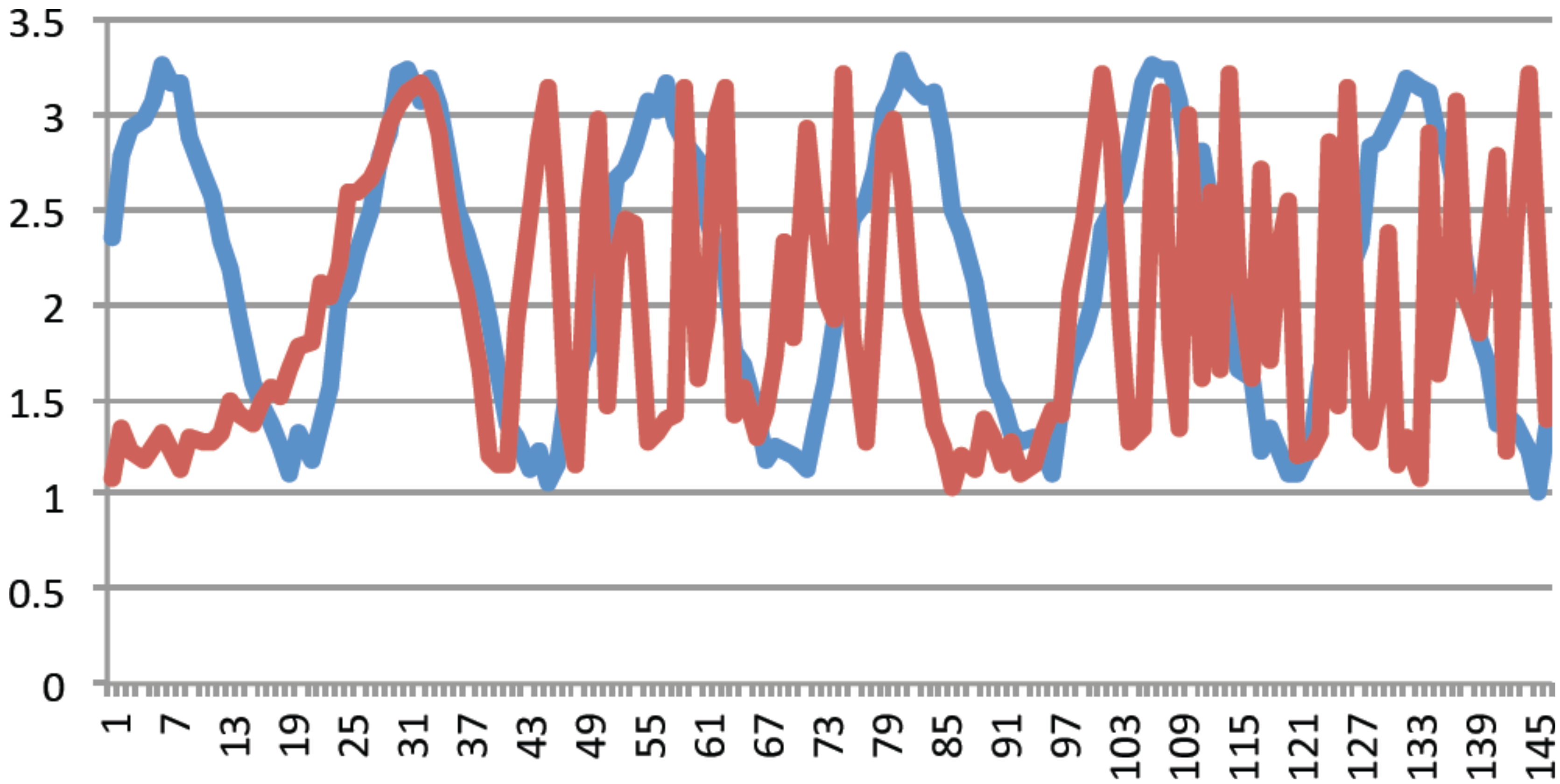


Cyclic with constant variance

Cyclic with time-dependent variance

Non-Stationary Series

- Non-stationary series
  - statistical properties change over time



Cyclic with time-dependent speed

Cyclic with constant speed

Why do these matter?

- Stationary series
  - statistical properties, such as mean and variance, are constant over time.
- Non-stationary series
  - statistical properties change over time
- Why important?
  - most statistical forecasting methods assume that the series can be rendered (approximately) stationary through mathematical transformations

Models of Time Series

Random Time Series Model

- Current observations only reflect current (random) error:

$$X_t = E_t \qquad E_t \sim N(0, \sigma^2)$$

- Intuitively, the "random error" represents a stochastic process that does not depend on the past

Autoregressive Time Series Model

- Current observations only reflect current (random) error and has no dependence on the past observations:

$$X_t = E_t$$

$$E_t \sim N(0, \sigma^2)$$

- AR(1): the current observation depends only the previous time instance and the current error

$$X_t = \alpha X_{t-1} + E_t + \lambda$$

Autoregressive Time Series Model

- Current observations only reflect current (random) error and has no dependence on the past observations:

$$X_t = E_t \qquad E_t \sim N(0, \sigma^2)$$

- AR(1): the current observation depends only the previous time instance and the current error

$$X_t = \alpha X_{t-1} + E_t + \lambda$$

- AR(2): the current observation depends only the previous 2 time instances and the current error

$$X_t = \alpha_1 X_{t-1} + \alpha_2 X_{t-1} + E_t + \lambda$$

Moving Average Time Series Model

- MA(1): The current observation depends only to the <u>error in the previous time instance</u> and the current error

$$X_t = \beta E_{t-1} + E_t + \lambda$$

- MA(2): The current observation depends only to the <u>error in the previous 2 time instances</u> and the current error

$$X_t = \beta_1 E_{t-1} + \beta_2 E_{t-2} + E_t + \lambda$$

ARMA model

- ARMA(a,m):
  - AR(a): The model has a auto-regressive terms
  - MA(m): The model has m moving-average terms

$$X_t = \left(\alpha_1 X_{t-1} + .. + \alpha_a X_{t-a}\right) + \left(\beta_1 E_{t-1} + .. + \beta_m E_{t-m}\right) + E_t + \lambda$$

- Shortcoming of ARMA models
  - These models cannot models where the current value is determined by taking into account the <u>speed</u> or <u>degree of acceleration</u> observed in the past.

Models with Differencing

- "differencing" enables models to also consider speed of change and degree of acceleration in determining the current value:
  - Order-1 differencing:

$$X_t^{(1)} = X_t - X_{t-1}$$

speed of change

  - Order-2 differencing:

$$X_t^{(2)} = X_t^{(1)} - X_{t-1}^{(1)}$$

degree of acceleration

  - …
  - Order-d differencing:

$$X_t^{(d)} = X_t^{(d-1)} - X_{t-1}^{(d-1)}$$

ARIMA model

- ARIMA(a,d,m)
  - AR(a): The model has a auto-regressive terms
  - MA(m): The model has m moving-average terms
  - Integrated with order-d differencing

$$X_t = \left(\alpha_1 X_{t-1} + .. + \alpha_a X_{t-a}\right) + \left(\beta_1 E_{t-1} + .. + \beta_m E_{t-m}\right) + E_t$$
$$+ \lambda + \left(\theta_1 X_{t-1}^{(d)} + .. + \theta_m X_{t-m}^{(d)}\right)$$

Models with Seasonal Differencing

- Seasonal "differencing" enables models to also consider speed of changes of values across a lag:
  - Lag "s" seasonal differencing:

$$s_t^{(s)} = x_t - x_{t-5}$$

Models with Seasonal Differencing

- Seasonal "differencing" enables models to also consider speed of changes of values across a lag:
  - lag "s" seasonal differencing:

$$s_t^{(s)} = x_t - x_{t-5}$$

- Seasonal ARIMA models also incorporate seasonal terms
  - additive seasonal models

$$X_t = \mathrm{ARIMA}(a,d,m) + \mathrm{SEASONAL}_s(A,D,M)$$

  - Multiplicative seasonal models

$$X_t = \mathrm{ARIMA}(a,d,m) \times \mathrm{SEASONAL}_s(A,D,M)$$

  - A= # seasonal autoregressive terms, D= # seasonal differences, M = # seasonal moving average terms

Model Discovery

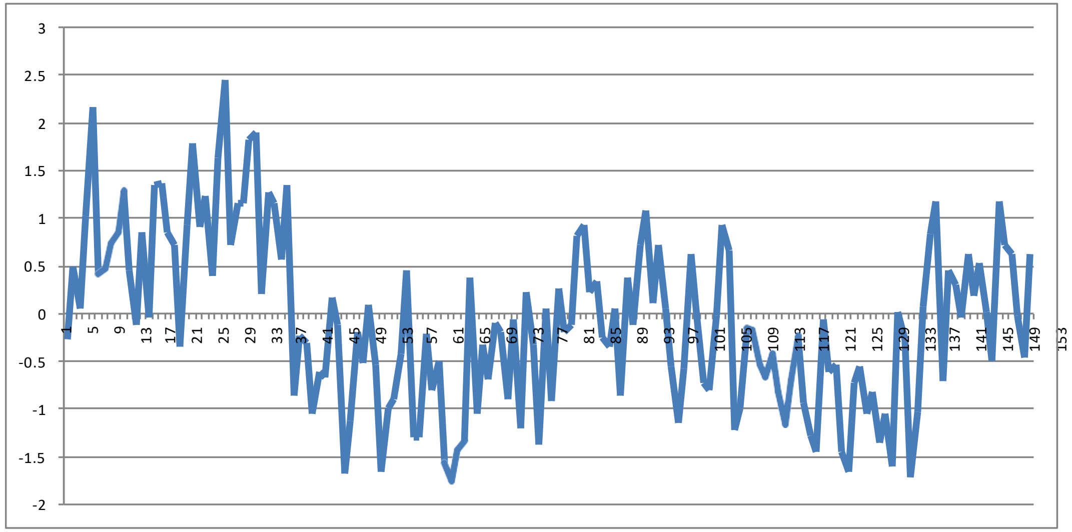
(Box-Jenkins Procedure)

Model based Time Series Analysis

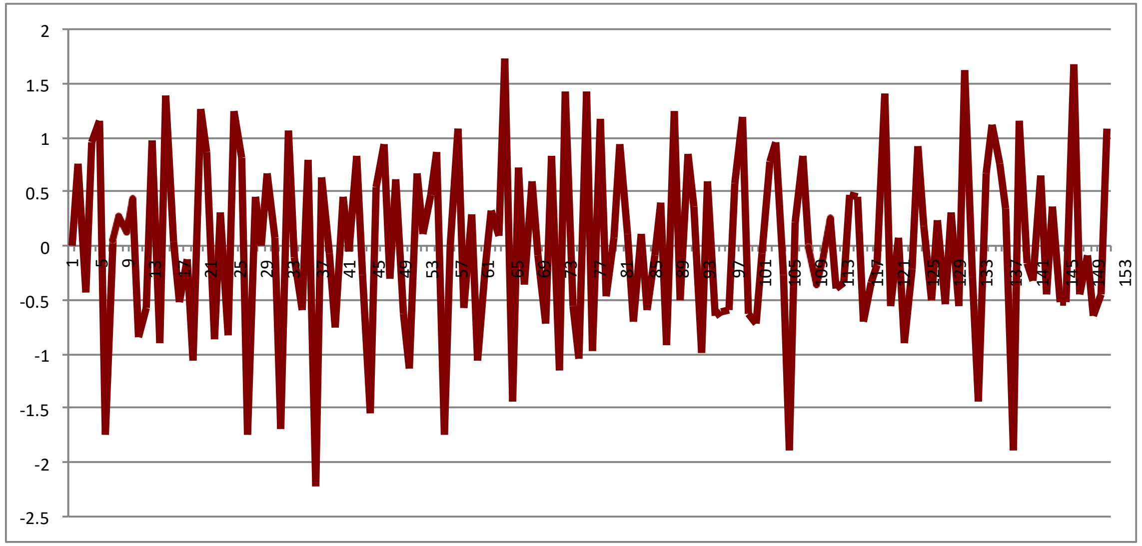
- Find the parameters of the model
  - Model fitting
    - the model should be as simple as possible (contain as few terms as possible)
    - the fit to historic data should be as good as possible

Box-Jenkins procedure

- Remove any seasonal patterns and deterministic trends that may hide valuable information and patterns through differencing
- When the mean trend is stochastic, differencing the series may yield a stationary stochastic process and, thus, may help convert a non-stationary series to a stationary one



Original Series



Differenced Series

Box-Jenkins procedure

- Remove any seasonal patterns and deterministic trends that may hide valuable information and patterns through differencing
- When the mean trend is stochastic, differencing the series may yield a stationary stochastic process and, thus, may help convert a non-stationary series to a stationary one
- Series that can be rendered stationary by D time differencing are called "integrated processes of order D"

Model based Time Series Analysis

- Find the parameters of the model
  - Model fitting
    - the model should be as simple as possible (contain as few terms as possible)
    - the fit to historic data should be as good as possible

Model based Time Series Analysis

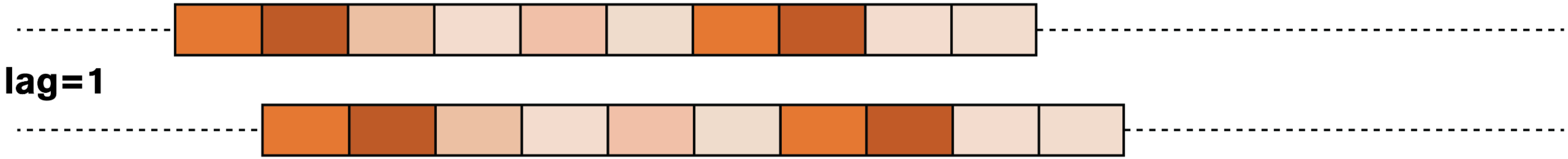
- Find the parameters of the model
  - Model fitting
  - Plot analysis
    - Autocorrelation Function (ACF) helps observe linear relationships between lagged values of a time series

$$\mathrm{ACF}(X, lag) = \frac{E\left[\left(X_t - \mu\right)\left(X_{t+lag} - \mu\right)\right]}{\sigma^2} = \frac{\mathrm{Covar}(X_t, X_{t+lag})}{\sigma^2}$$

Model based Time Series Analysis

- Find the parameters of the model
  - Model fitting
  - Plot analysis
    - Autocorrelation Function (ACF) helps observe linear relationships between lagged values of a time series

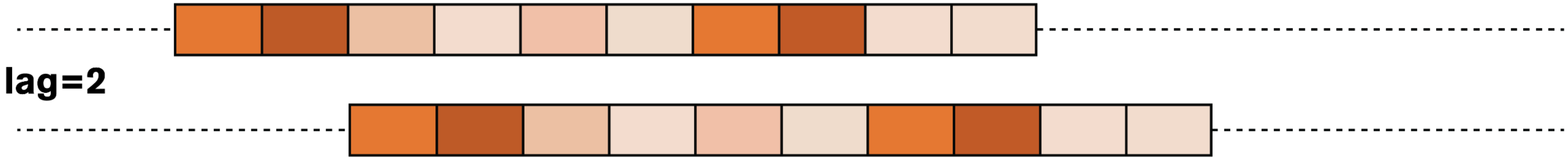
$$\mathrm{ACF}(X, lag) = \frac{E\left[(X_t - \mu)(X_{t+lag} - \mu)\right]}{\sigma^2} = \frac{\mathrm{Covar}(X_t, X_{t+lag})}{\sigma^2}$$



Model based Time Series Analysis

- Find the parameters of the model
  - Model fitting
  - Plot analysis
    - <u>Autocorrelation Function (ACF)</u> helps observe linear relationships between lagged values of a time series

$$\mathrm{ACF}(X, lag) = \frac{E\left[\left(X_t - \mu\right)\left(X_{t+lag} - \mu\right)\right]}{\sigma^2} = \frac{\mathrm{Covar}(X_t, X_{t+lag})}{\sigma^2}$$



Model based Time Series Analysis

- Find the parameters of the model
  - Model fitting
  - Plot analysis
    - Autocorrelation Function (ACF) helps observe linear relationships between lagged values of a time series

$$\mathrm{ACF}(X, lag) = \frac{E\left[\left(X_t - \mu\right)\left(X_{t+lag} - \mu\right)\right]}{\sigma^2} = \frac{\mathrm{Covar}(X_t, X_{t+lag})}{\sigma^2}$$

Model based Time Series Analysis

- Find the parameters of the model
  - Model fitting
  - Plot analysis
    - Autocorrelation Function (ACF) helps observe linear relationships between lagged values of a time series

$$\mathrm{ACF}(X, lag) = \frac{E\left[(X_t - \mu)(X_{t+lag} - \mu)\right]}{\sigma^2} = \frac{\mathrm{Covar}(X_t, X_{t+lag})}{\sigma^2}$$

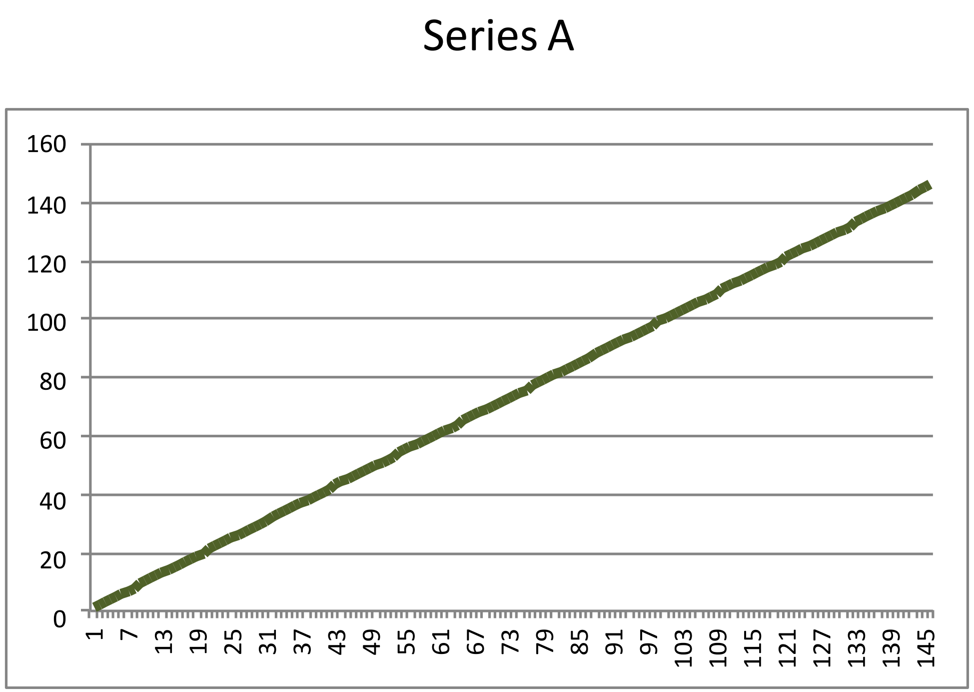
    - Partial Autocorrelation Function (PACF) discount the impact of intermediate values

$$\mathrm{PACF}(X, lag) = \frac{\mathrm{Covar}(X_t, X_{t+lag} \mid X_{t+1}, \ldots, X_{t+lag-1})}{\sigma^2}$$

Key Properties of ACF and PACF

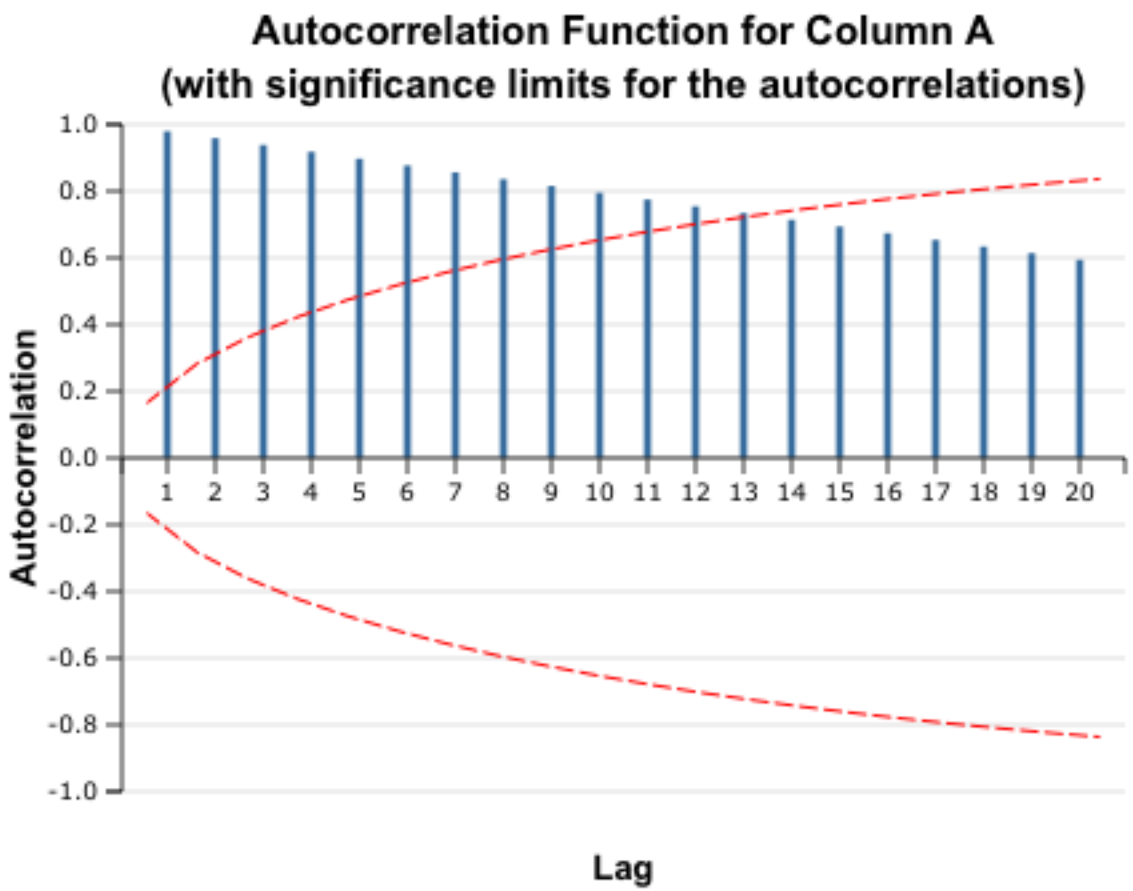
- For trend series autocorrelation function (ACF) slowly decays.

Autoregressive model



$$X_t = X_{t-1} + E_t$$

An AR(1) model



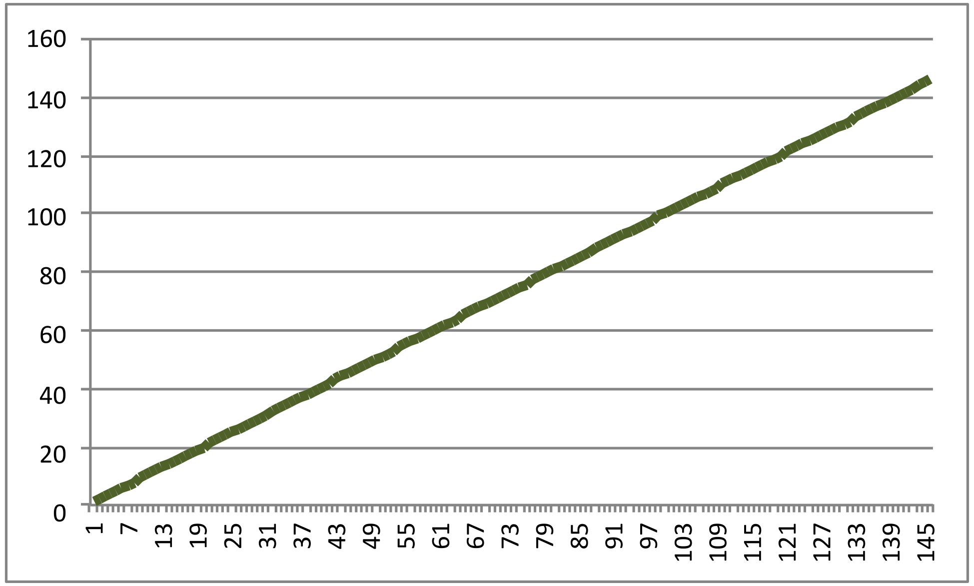
- Slowly decaying ACF indicates a trend
- If the series has positive autocorrelations for high number of lags, then it may need differencing to remove the trend or seasonal patterns

Key Properties of ACF and PACF

- For trend series autocorrelation function (ACF) slowly decays.
- For an autoregressive, AR(a), series, the partial autocorrelation function (PACF) gives 0 at lag >= a + 1

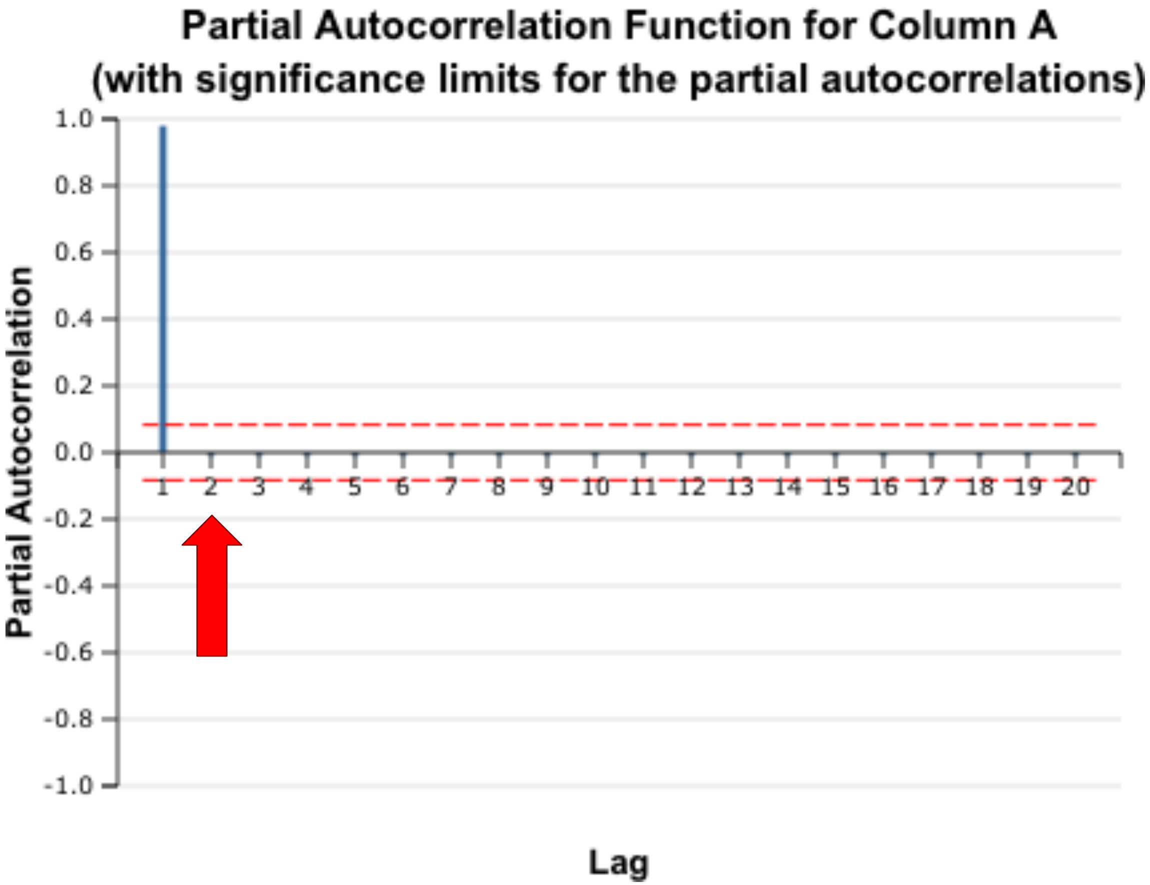
Autoregressive model

Series A



$$X_t = X_{t-1} + E_t$$

An AR(1) model



- Partial autocorrelation of an AR(a) process is zero at lag >= a + 1

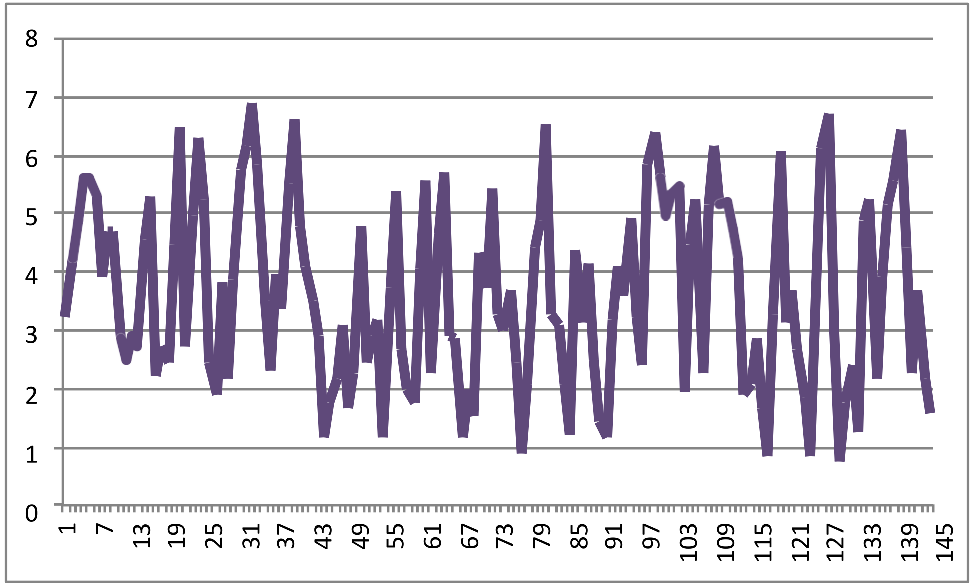
Key Properties of ACF and PACF

- For an autoregressive, AR(a), series, the partial autocorrelation function (PACF) gives 0 at lag >= a + 1

- For a moving average, MA(m), series,
  - the partial autocorrelation function (PACF) does not “shut off” at a fixed lag, but moves toward 0.

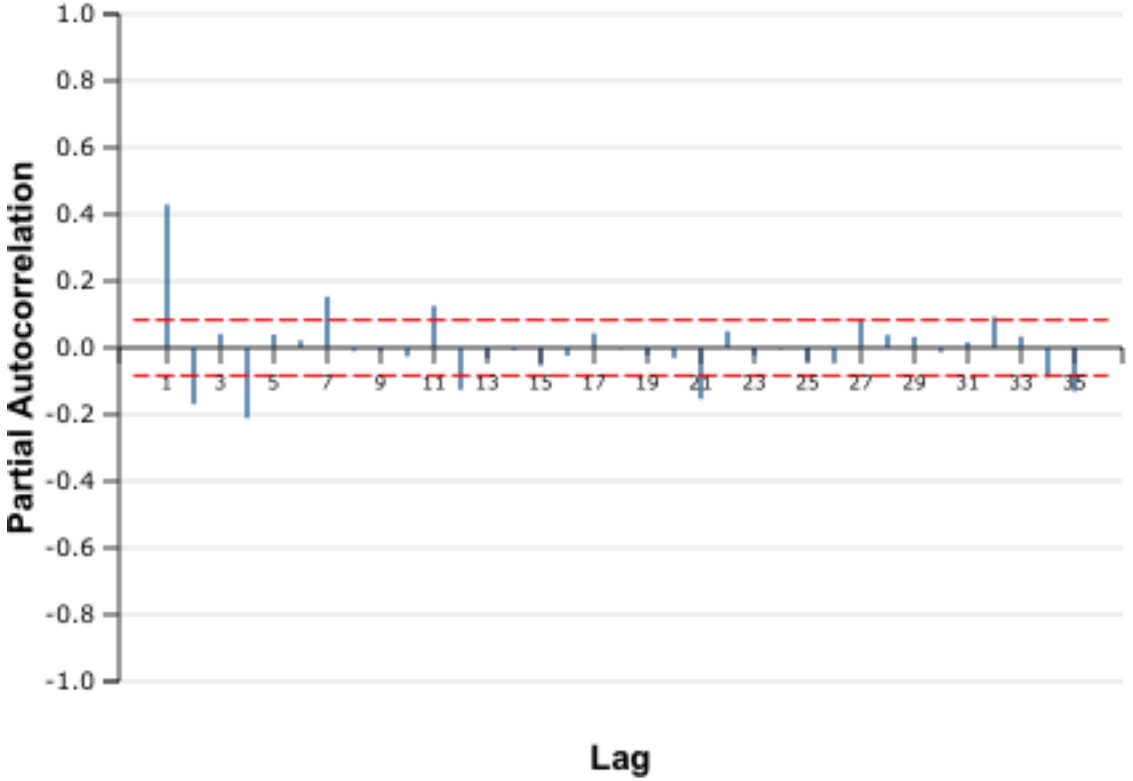
Moving average model

Series B



$$X_t = 0.5E_{t-1} + E_t$$

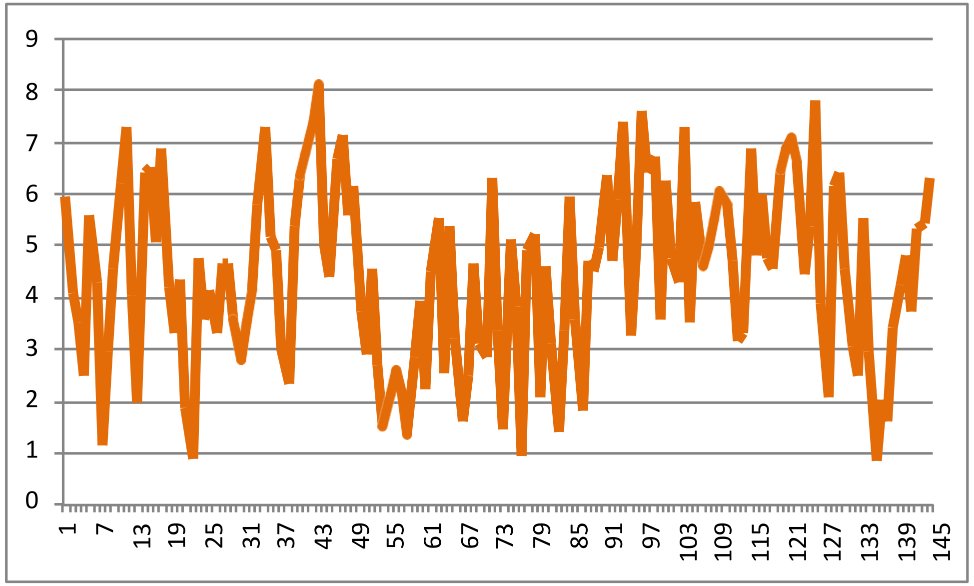
An MA(1) model



- Partial autocorrelation of an MA(m) process does not "shut off" at a fixed lag, but moves toward 0

Moving average model

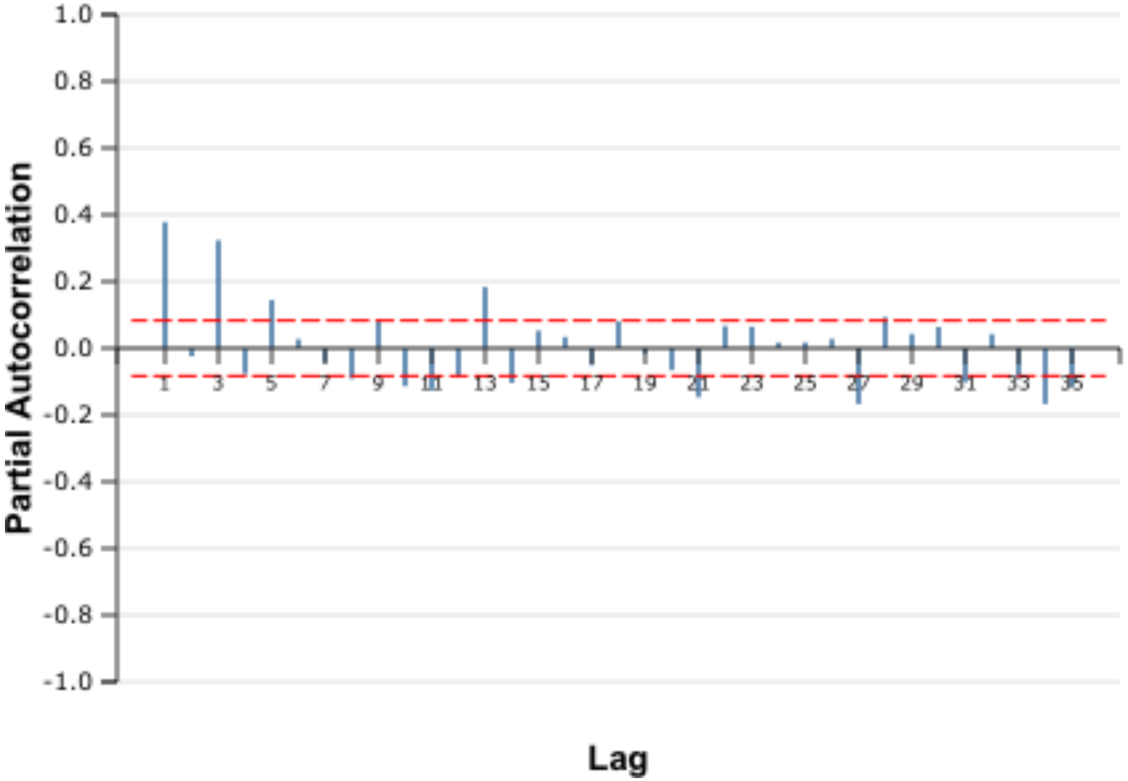
Series B



$$X_t = 0.5E_{t-1} + 0.3E_{t-3} + E_t$$

An MA(3) model

Partial Autocorrelation Function for Column B
(with significance limits for the partial autocorrelations)



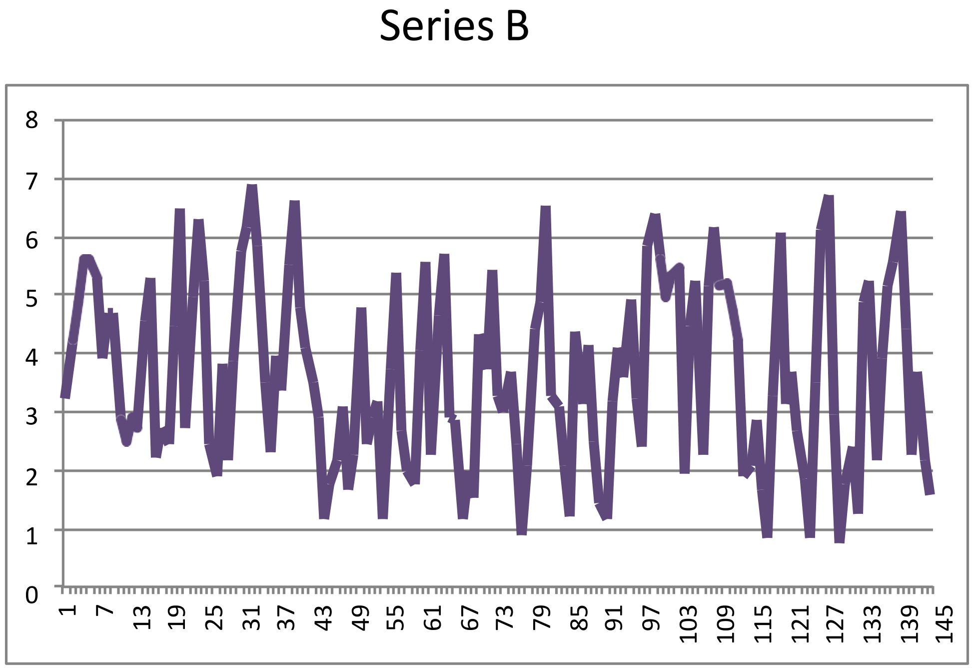
- Partial autocorrelation of an MA(m) process does not "shut off" at a fixed lag, but moves toward 0

Key Properties of ACF and PACF

- For an autoregressive, AR(a), series, the partial autocorrelation function (PACF) gives 0 at lag >= a + 1

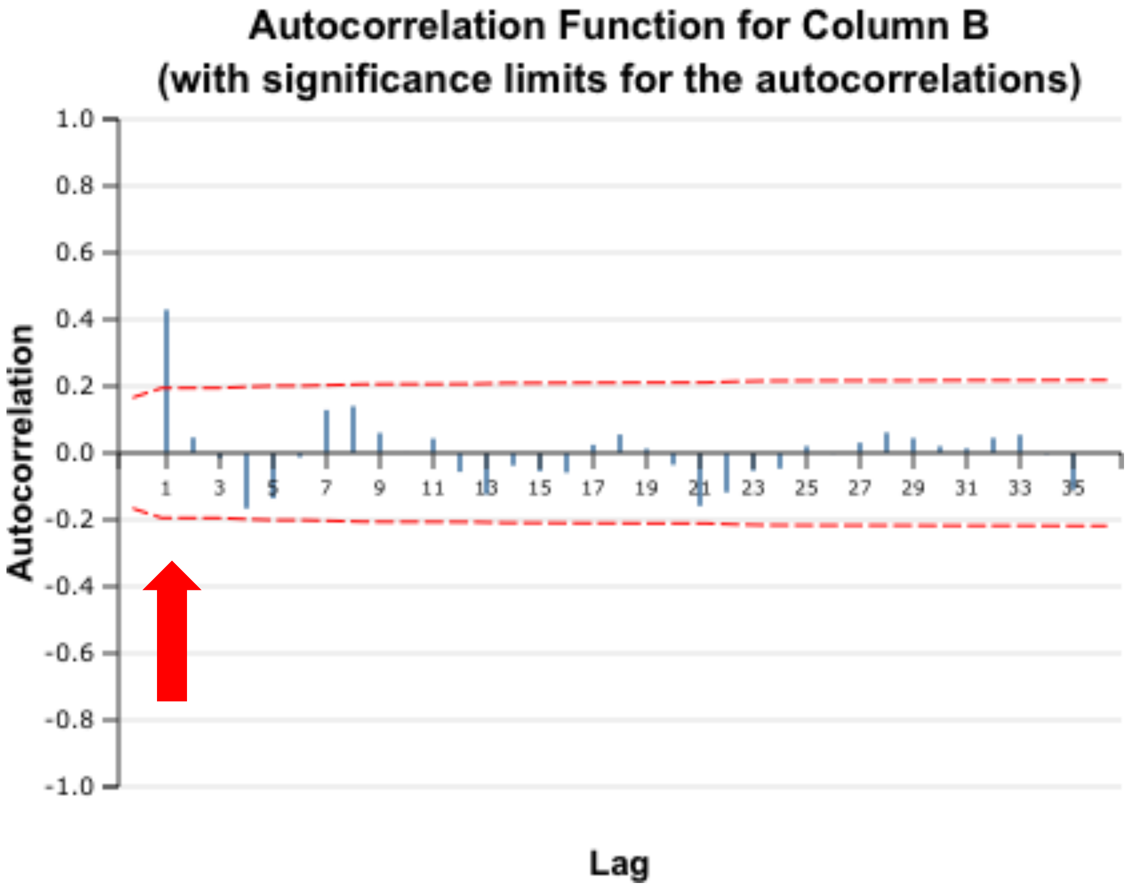
- For a moving average, MA(m), series,
  - the partial autocorrelation function (PACF) does not “shut off” at a fixed lag, but moves toward 0.
  - the autocorrelation function (ACF) has non-zero autocorrelation only at the lags of the model.

Moving average model



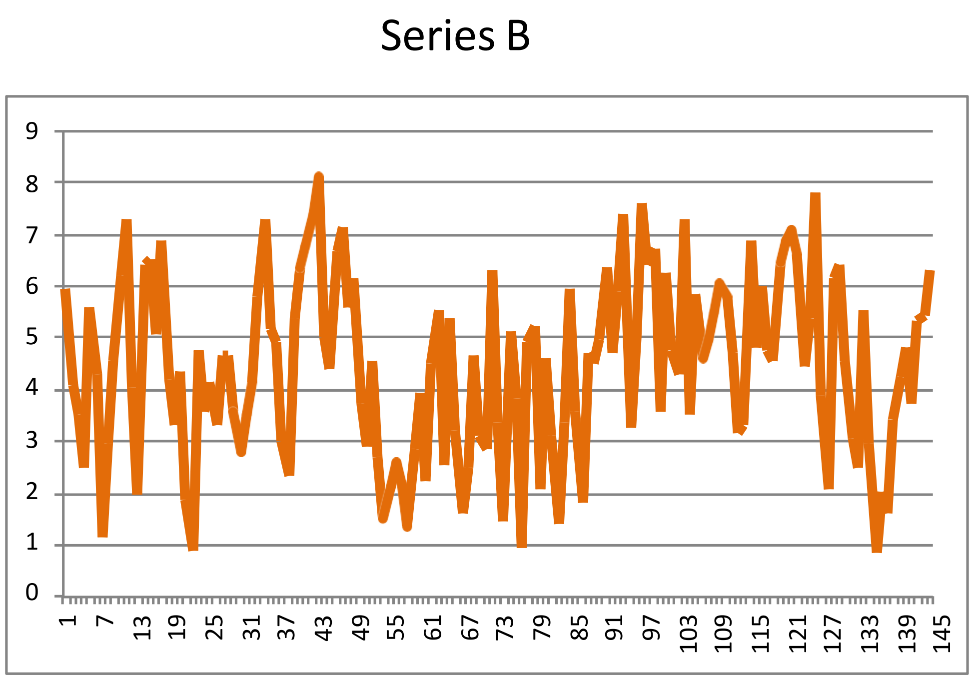
$$X_t = 0.5E_{t-1} + E_t$$

An MA(1) model



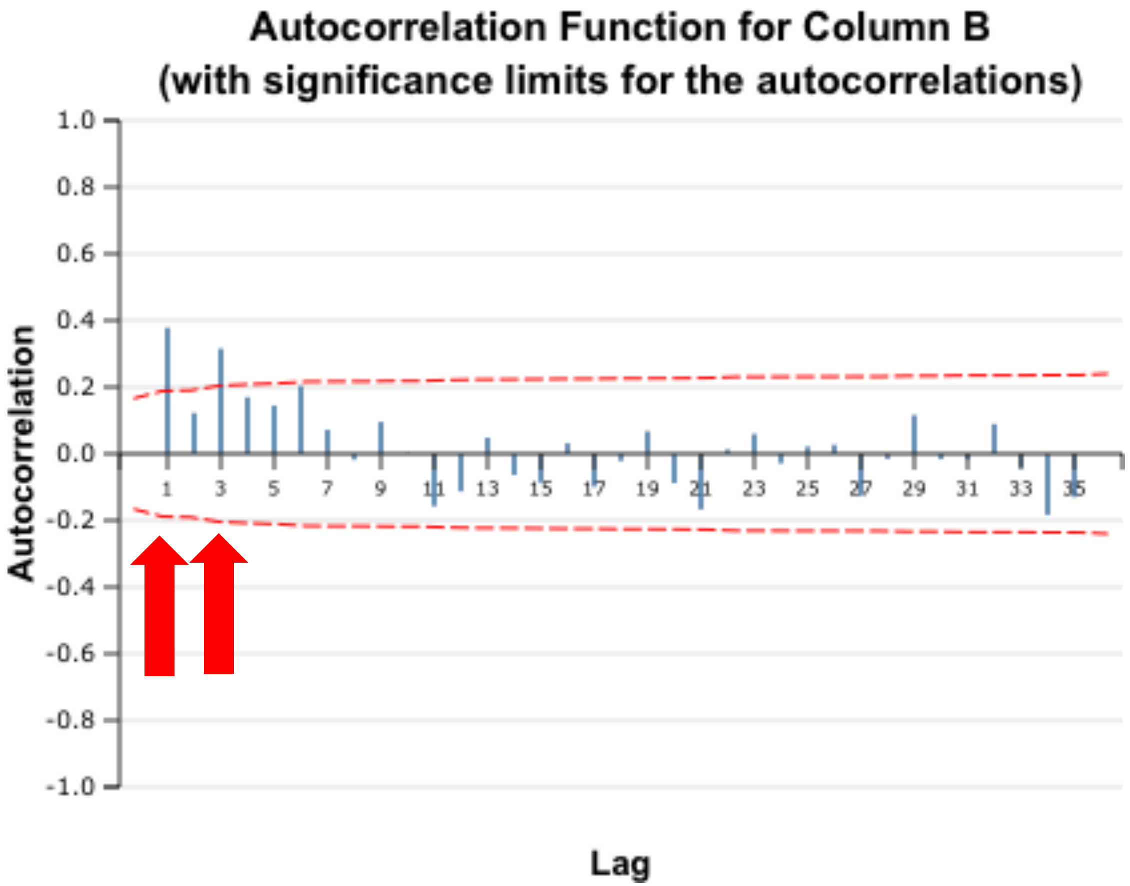
- Autocorrelation of an MA(m) process has non-zero autocorrelation only at the lags of the model

Moving average model



$$X_t = 0.5E_{t-1} + 0.3E_{t-3} + E_t$$

An MA(3) model



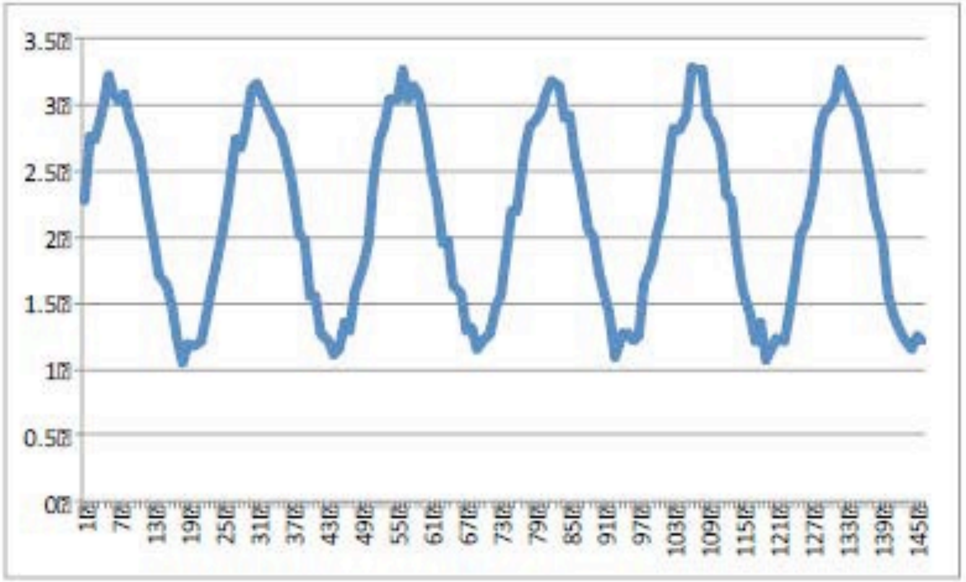
- Autocorrelation of an MA(m) process has non-zero autocorrelation only at the lags of the model

Key Properties of ACF and PACF

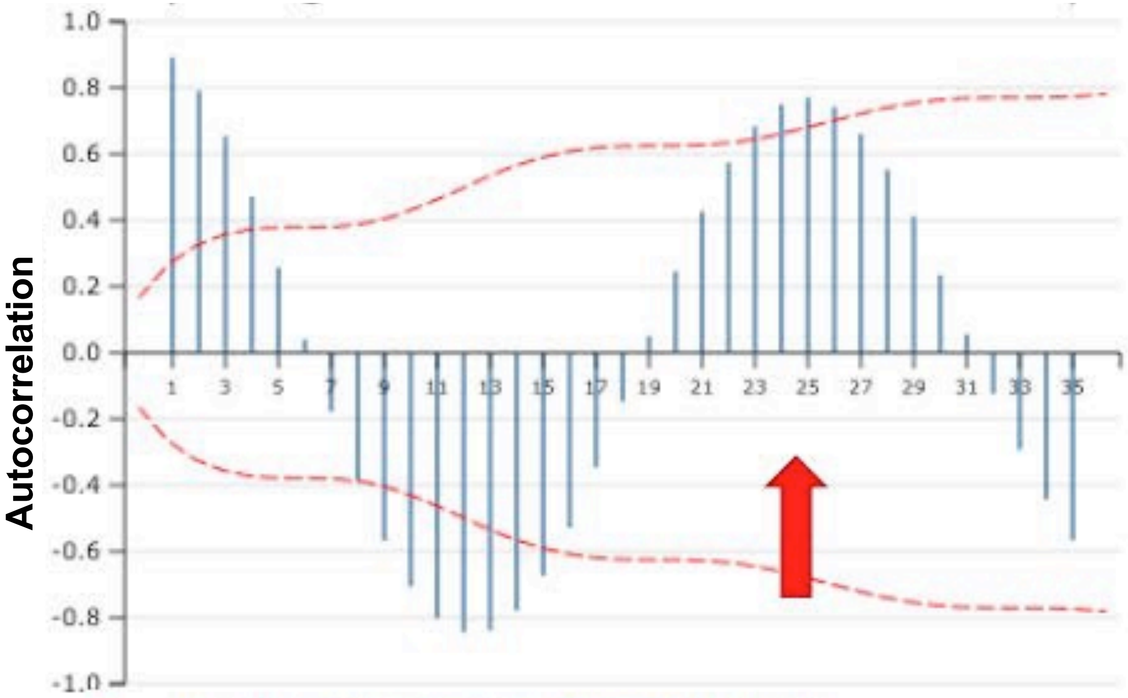
- For trend series autocorrelation function (ACF) slowly decays.
- For an autoregressive, AR(a), series, the partial autocorrelation function (PACF) gives 0 at lag >= a + 1
- For a moving average, MA(m), series,
  - the partial autocorrelation function (PACF) does not "shut off" at a fixed lag, but moves toward 0.
  - the autocorrelation function (ACF) has non-zero autocorrelation only at the lags of the model.
- For **seasonal series** autocorrelation function **(ACF) at the seasonal lag will be large and positive**.

Non-stationary (cyclic) model

Series B

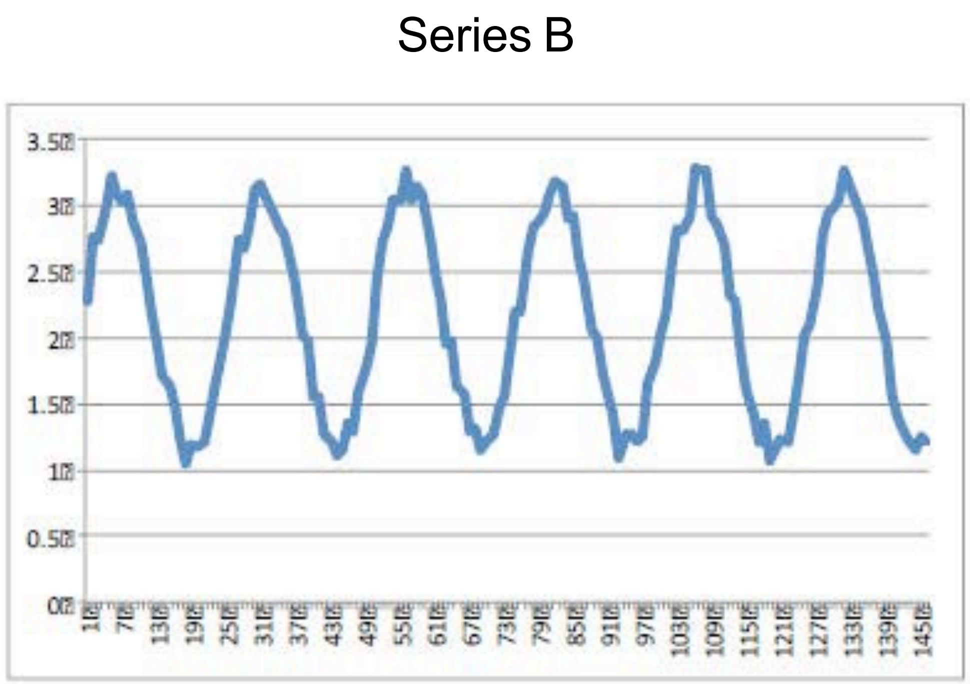


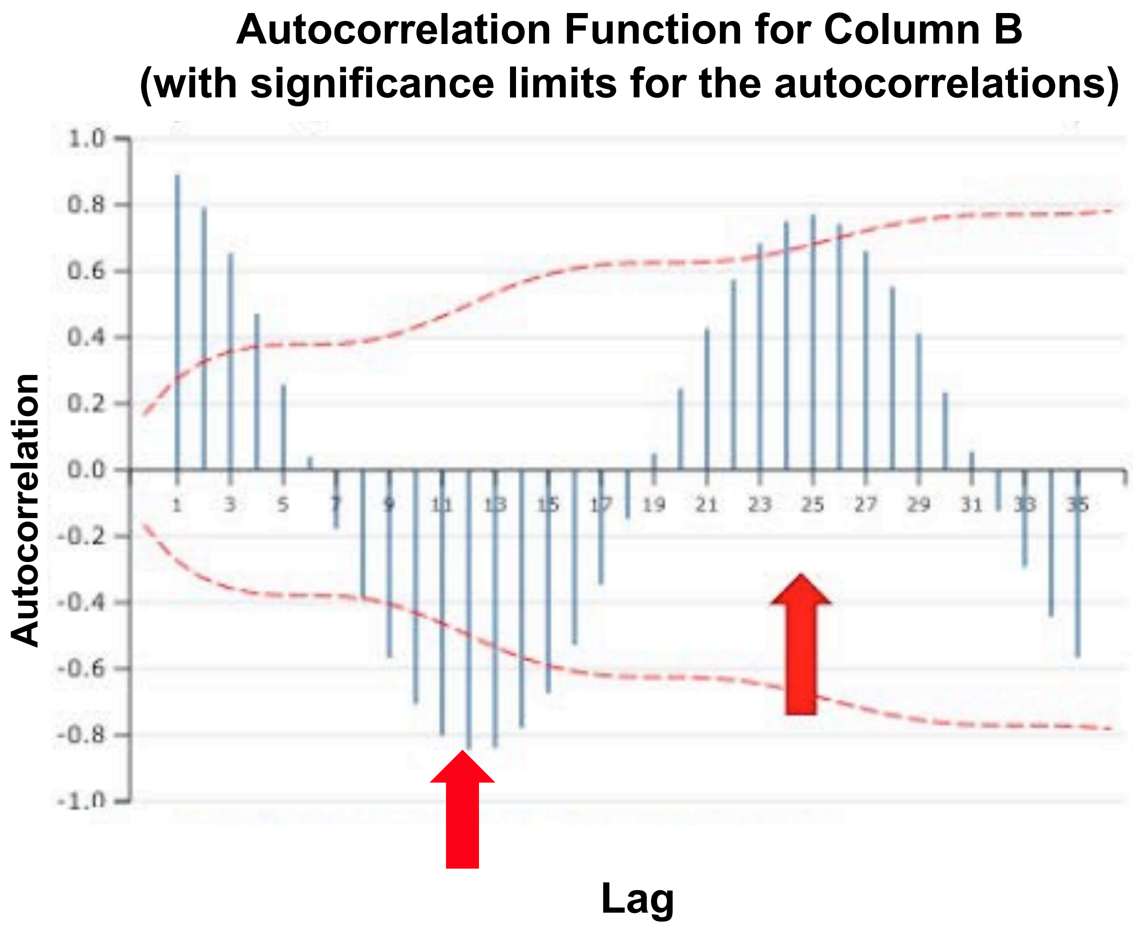
Autocorrelation Function for Column B
(with significance limits for the autocorrelations)



- No decay to zero – not stationary
- If the series has positive autocorrelations for high number of lags, then it may need differencing to remove the trend or seasonal patterns
- High values at fixed interval – seasonal autoregressive term
- (seasonal pattern with period of 24)

Non-stationary (cyclic) model

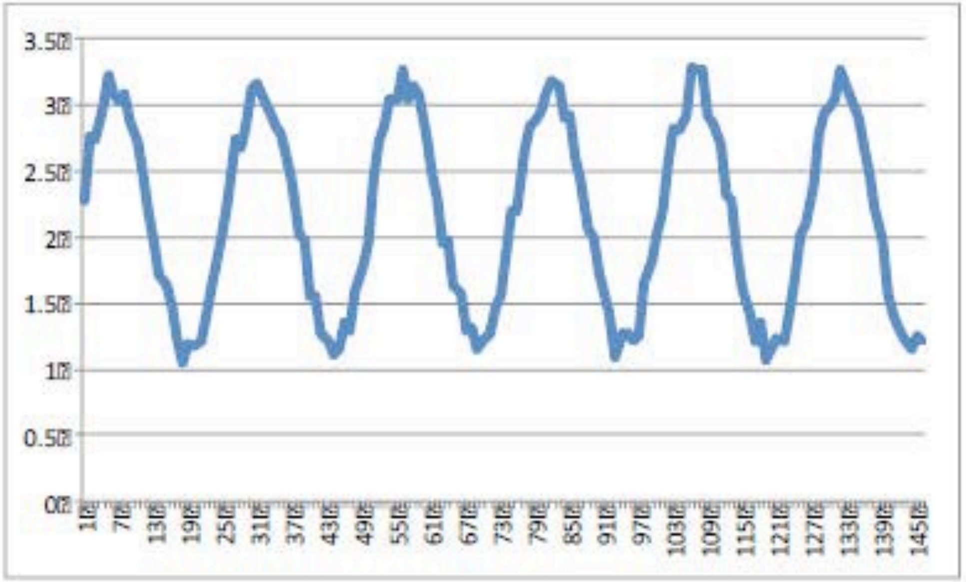




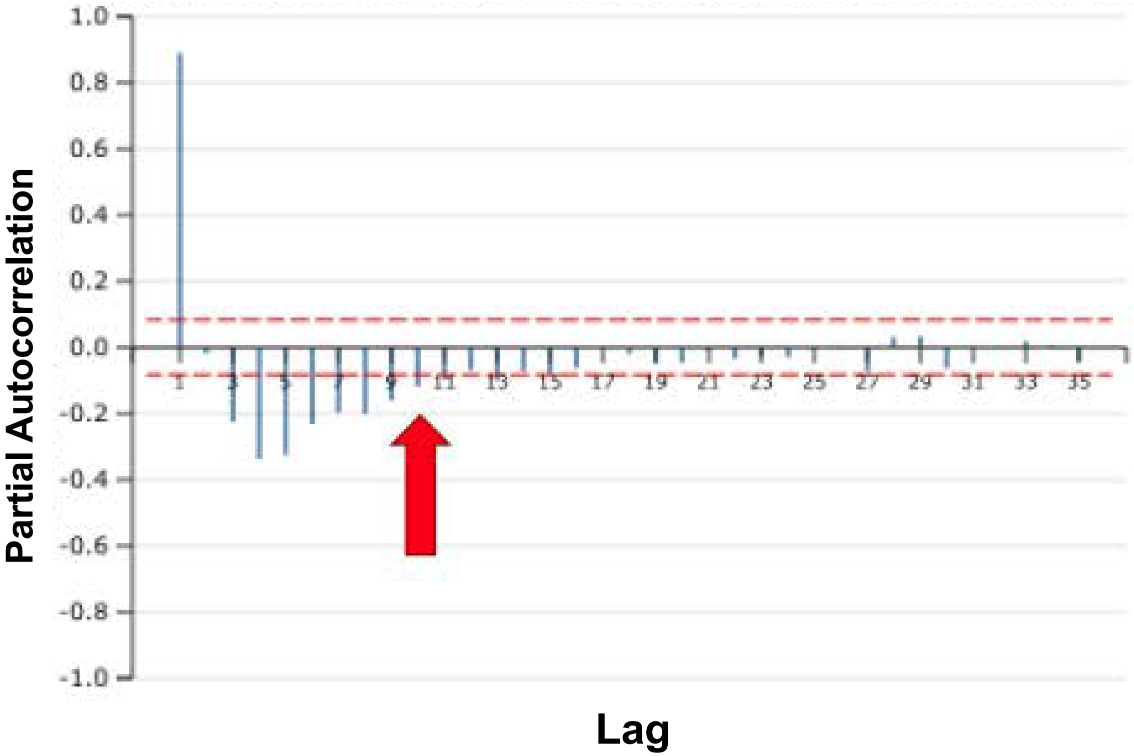
- The overall autocorrelation pattern indicates a cyclic pattern with period of 24

Non-stationary (cyclic) model

Series B

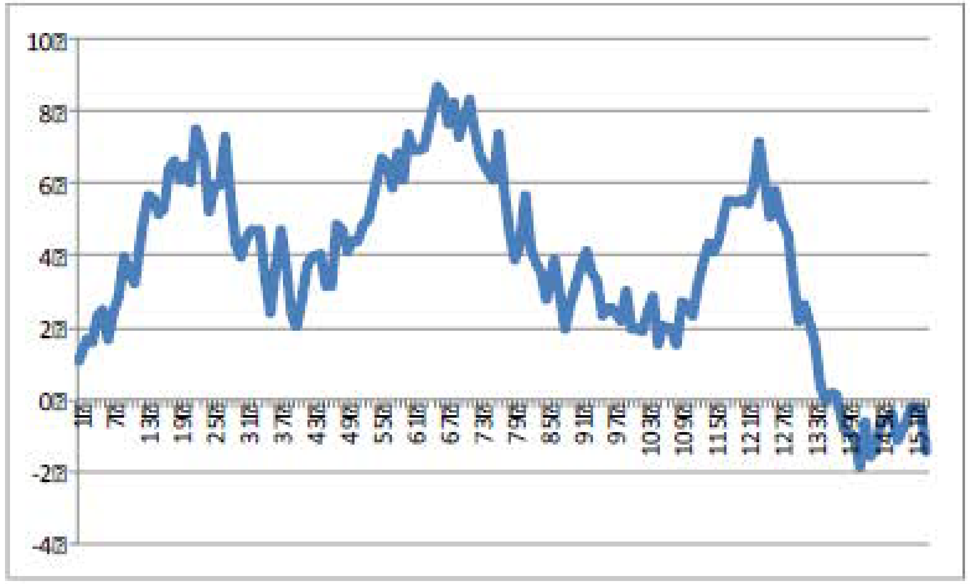


Partial Autocorrelation Function for Column B
(with significance limits for the partial autocorrelations)

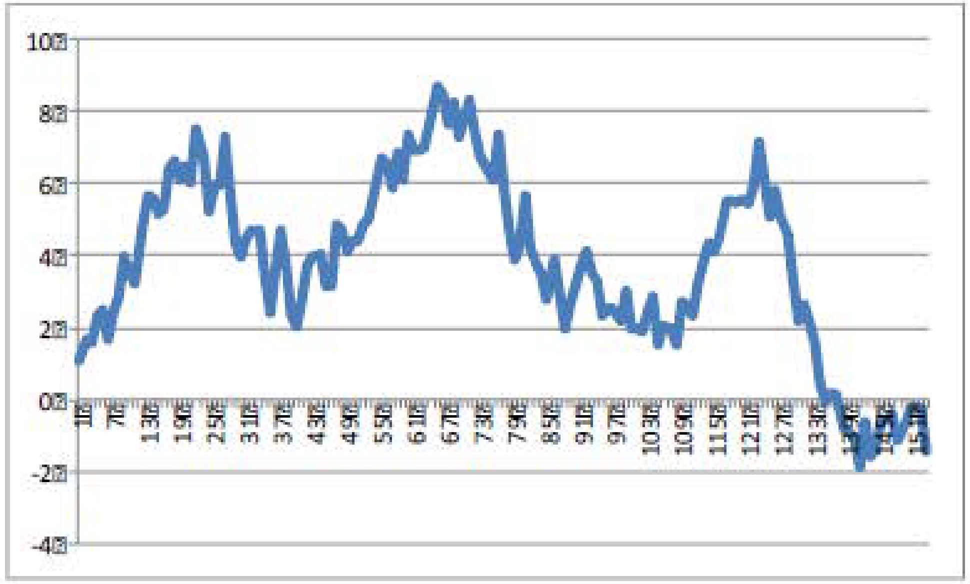


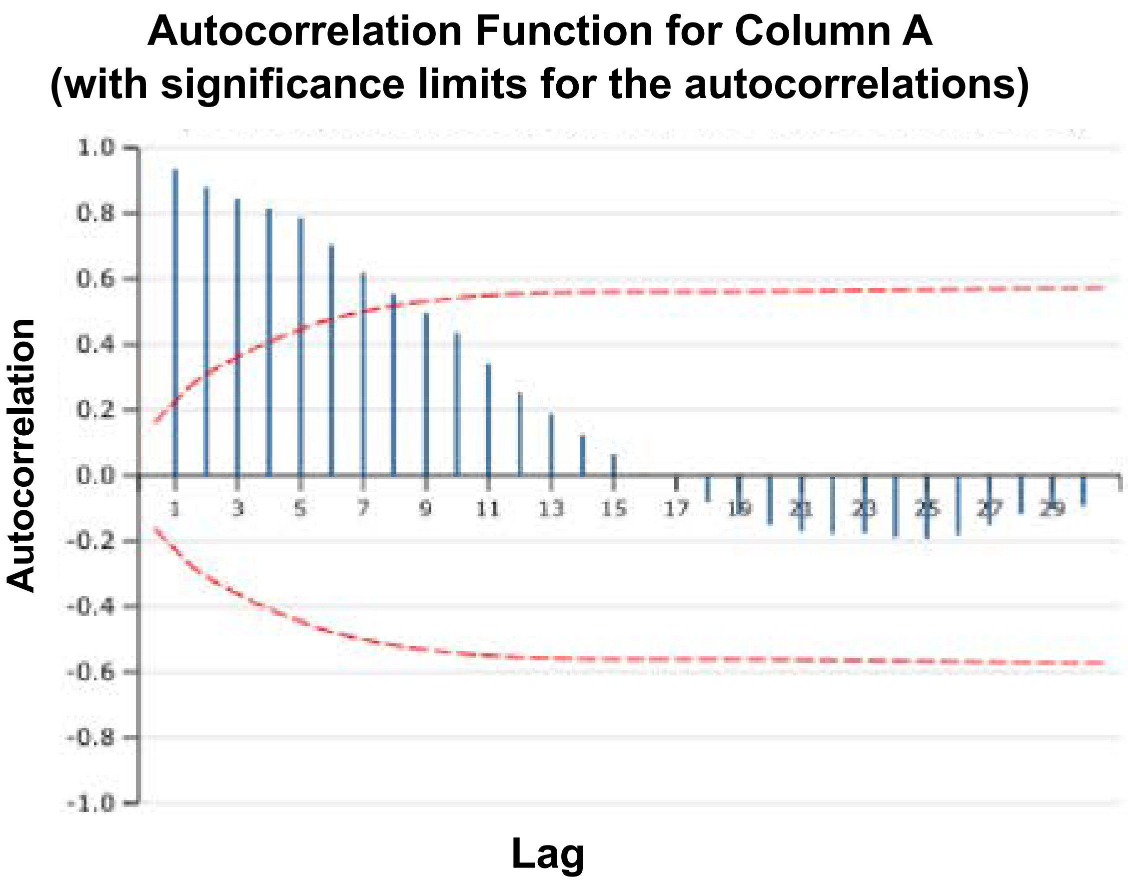
- The partial autocorrelation pattern may indicate a cyclic pattern with 12 unit half lag

Complex model



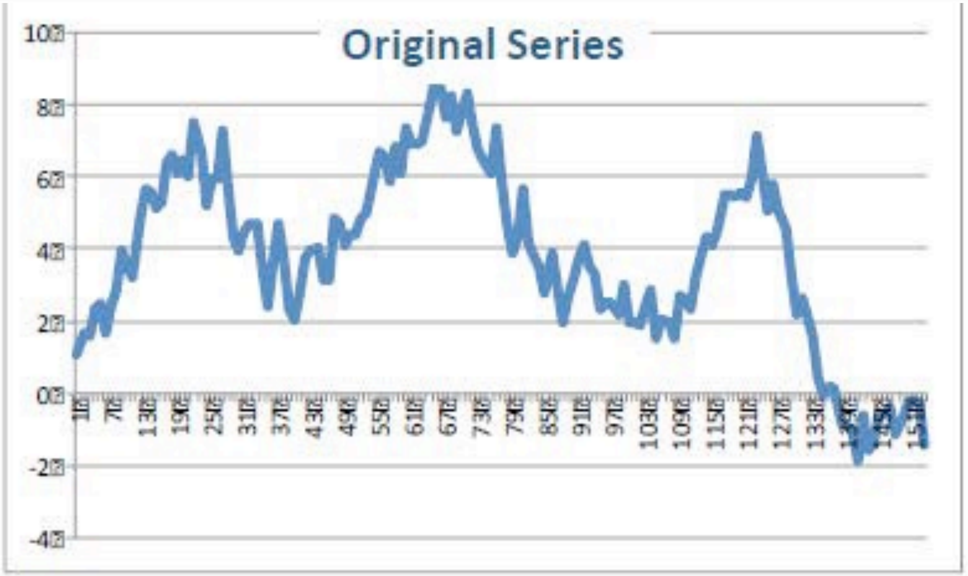
Complex model

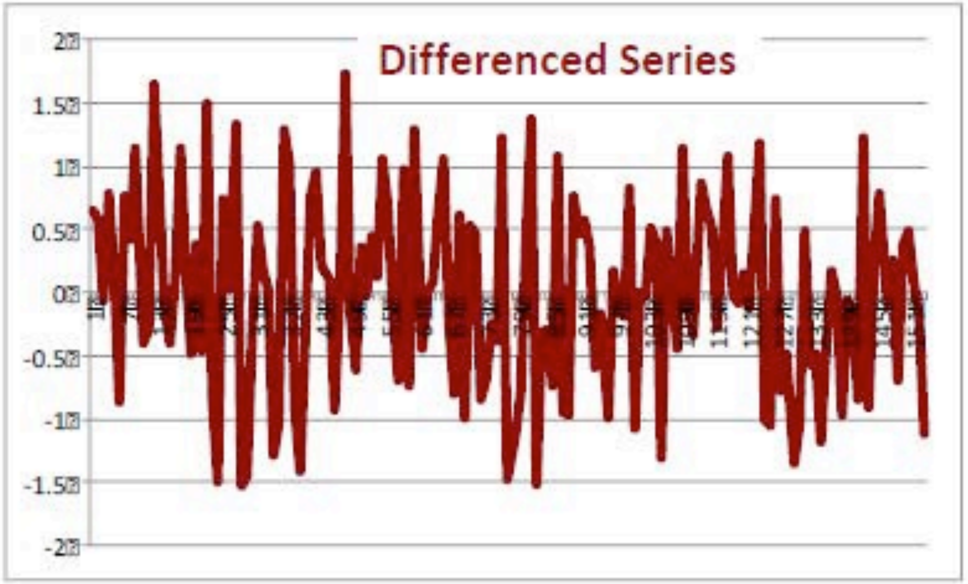


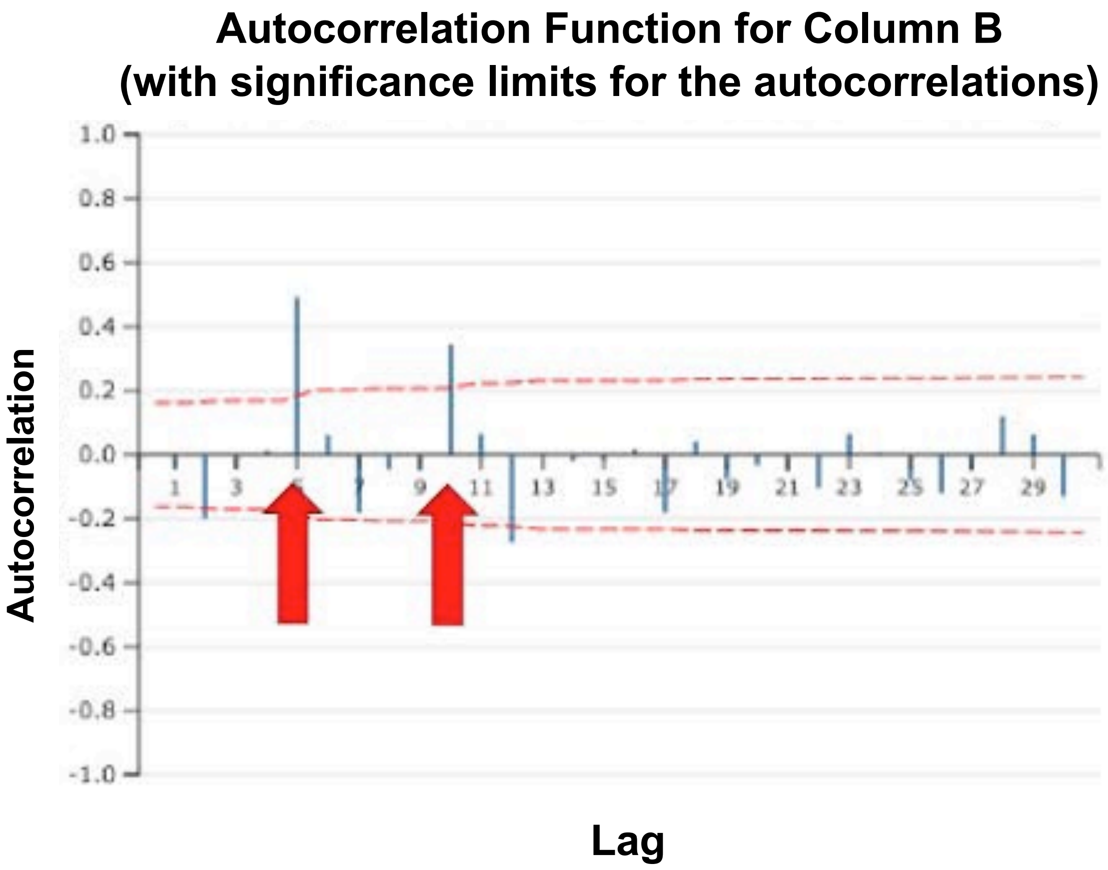


- Slowly decaying ACF indicates a trend
- if the series has positive autocorrelation for high number of lags, then it may need differencing to remove the trend or seasonal patterns

Complex model

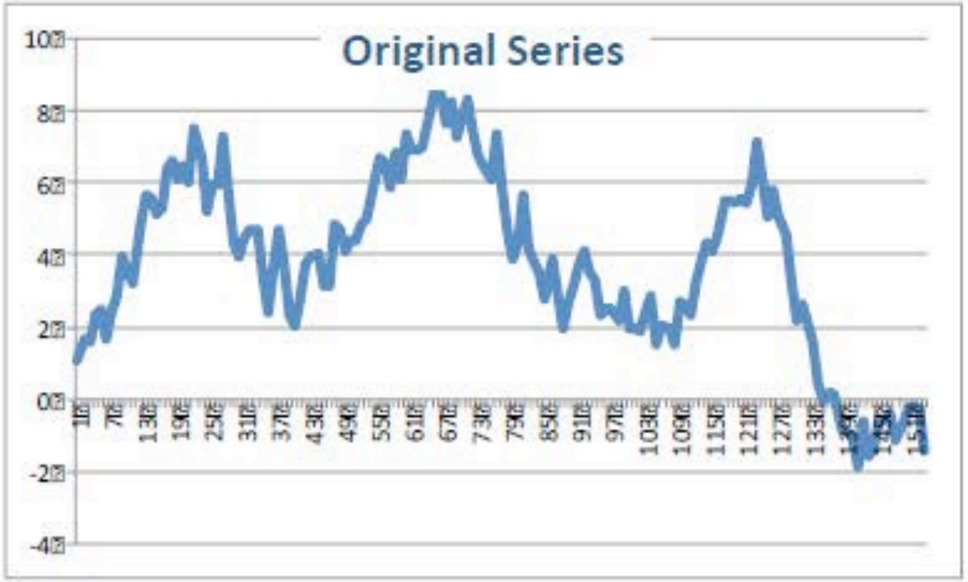


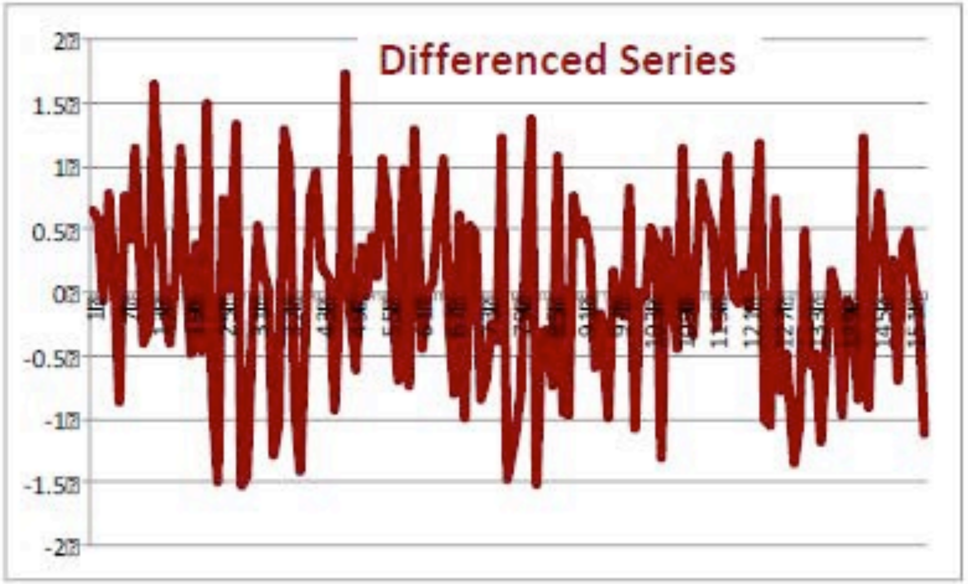




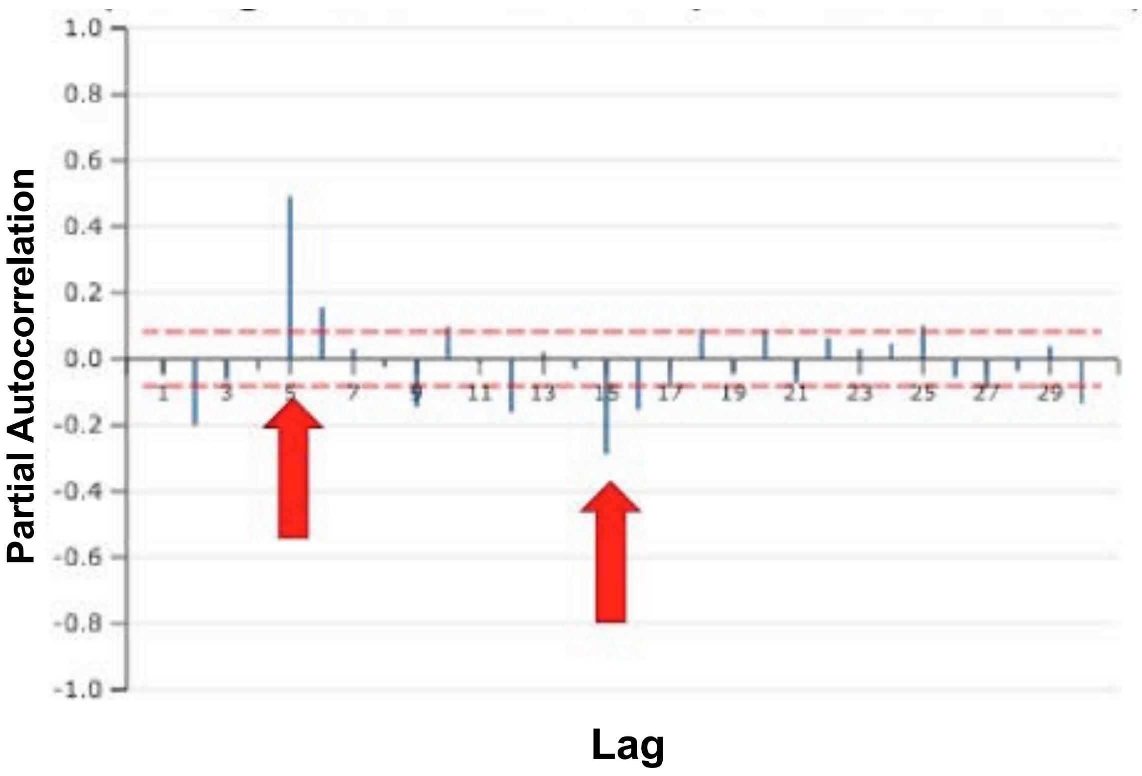
- Autocorrelation of an MA(m) process has non-zero autocorrelation only at the lags of the model
- Indicates MA(5) and MA(10) components.

Complex model



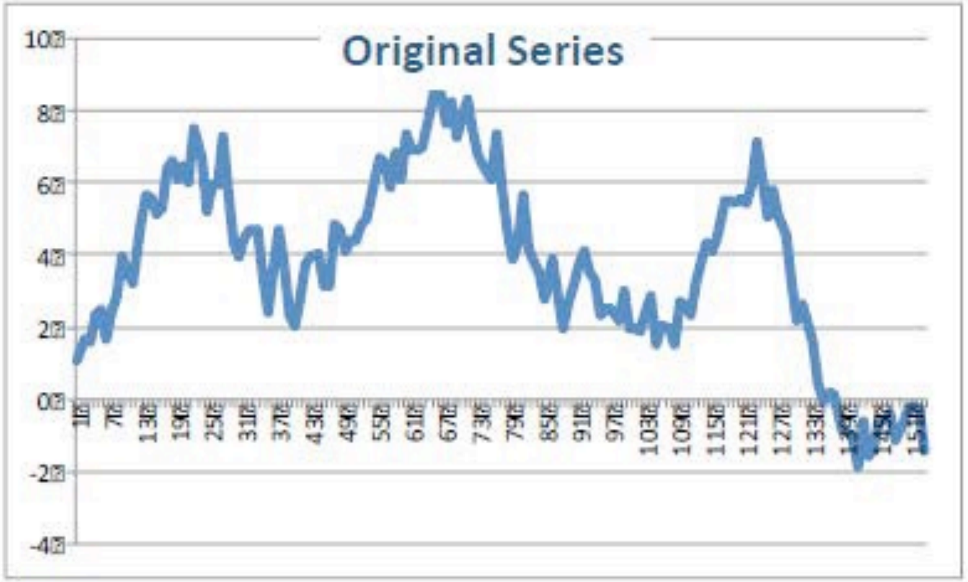


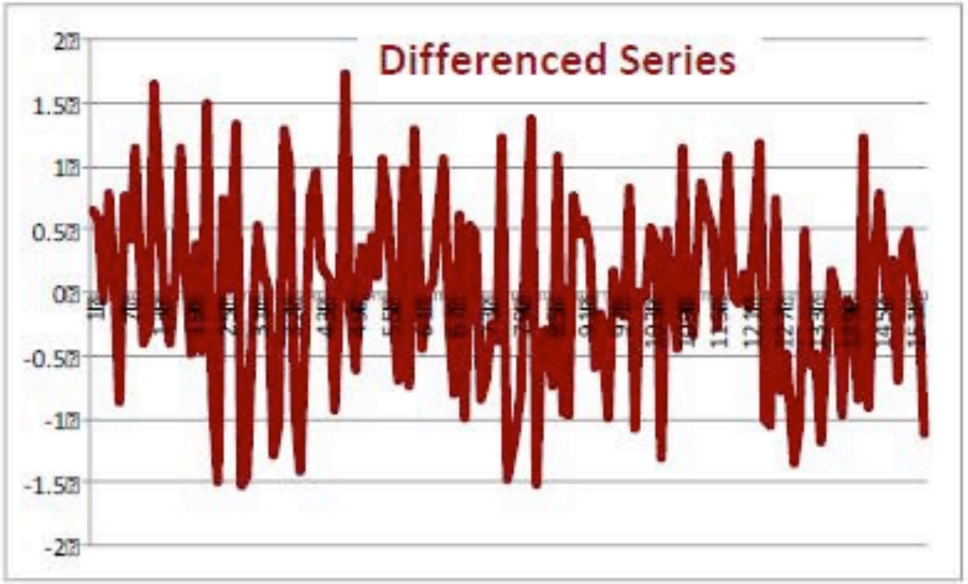
Partial Autocorrelation Function for Column B (with significance limits for the partial autocorrelations)



- Partial autocorrelation of an MA(m) process does not “shut off” at a fixed lag, but moves toward 0

Complex model





- The differenced series appears to be a moving average series with MA(5) and MA(10) components.

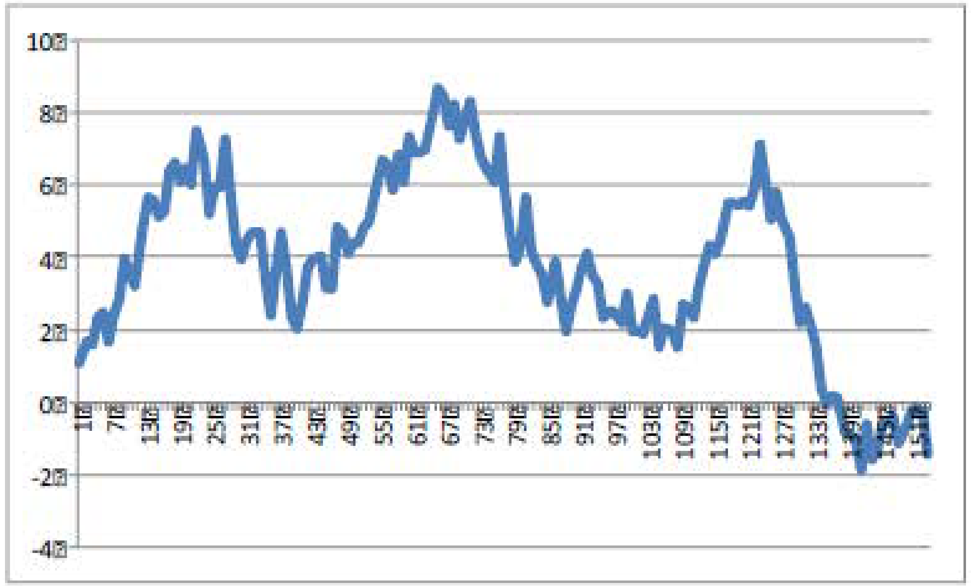
$$D_t = X_t - X_{t-1} = \alpha E_{t-5} + \beta E_{t-10} + E_t$$

$$X_t = X_{t-1} + \alpha E_{t-5} + \beta E_{t-10} + E_t$$

$$X_t = X_{t-1} + 0.5 E_{t-5} + 0.5 E_{t-10} + E_t$$

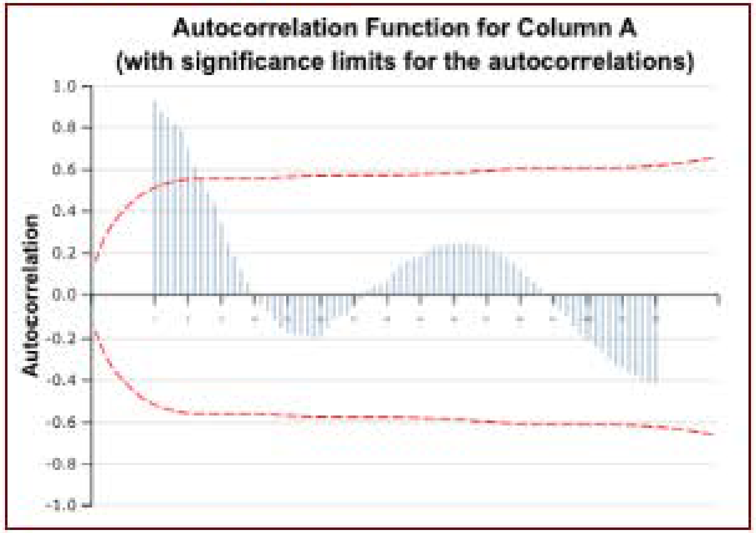
Moving average model

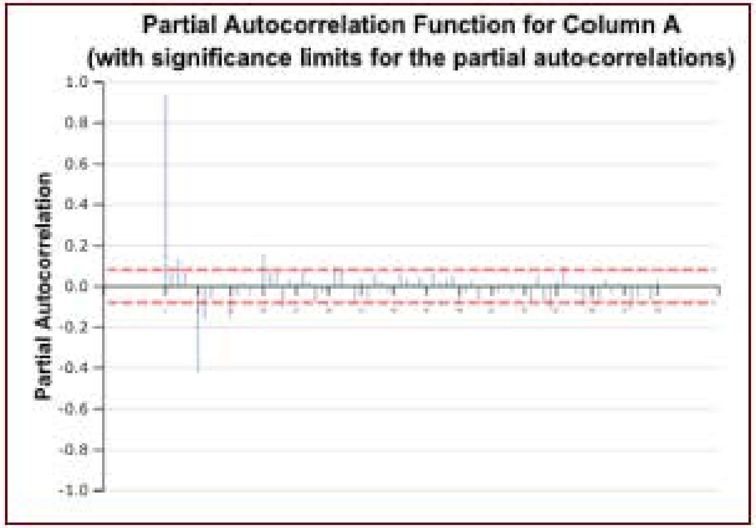
Original Series



$$X_t = X_{t-1} + 0.5E_{t-5} + 0.5E_{t-10} + E_t$$

- But isn't this series cyclic?





Sequences and Time Series

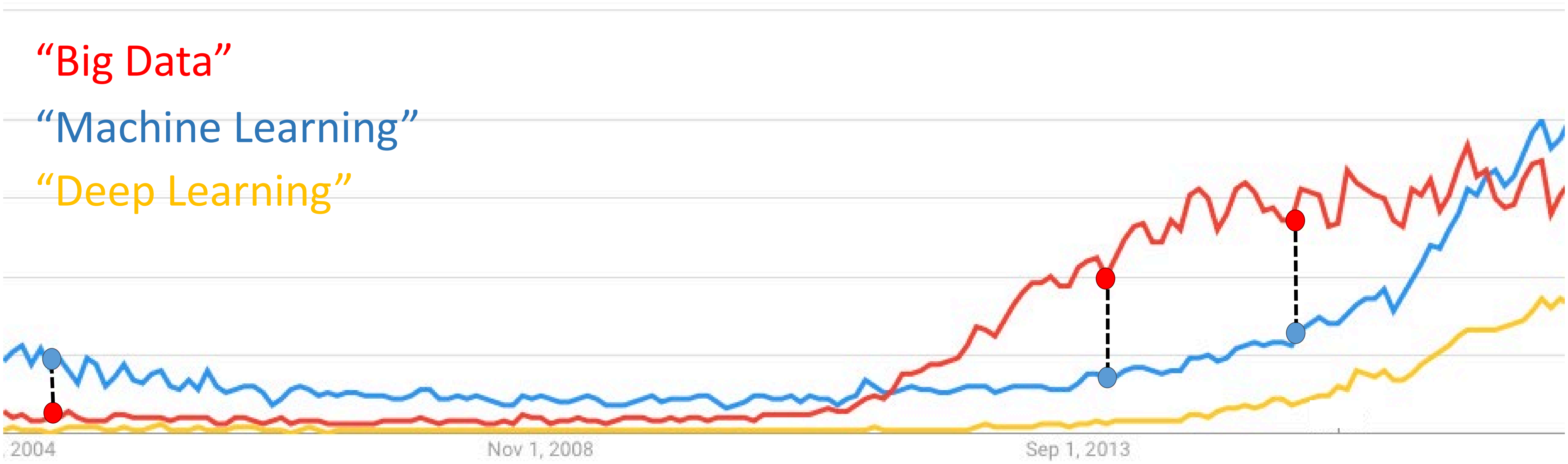
Time Series Matching

K. Selçuk Candan, Professor of Computer Science and Engineering

CASCADE
ASU Center for Assured and SCAlable Data Engineering

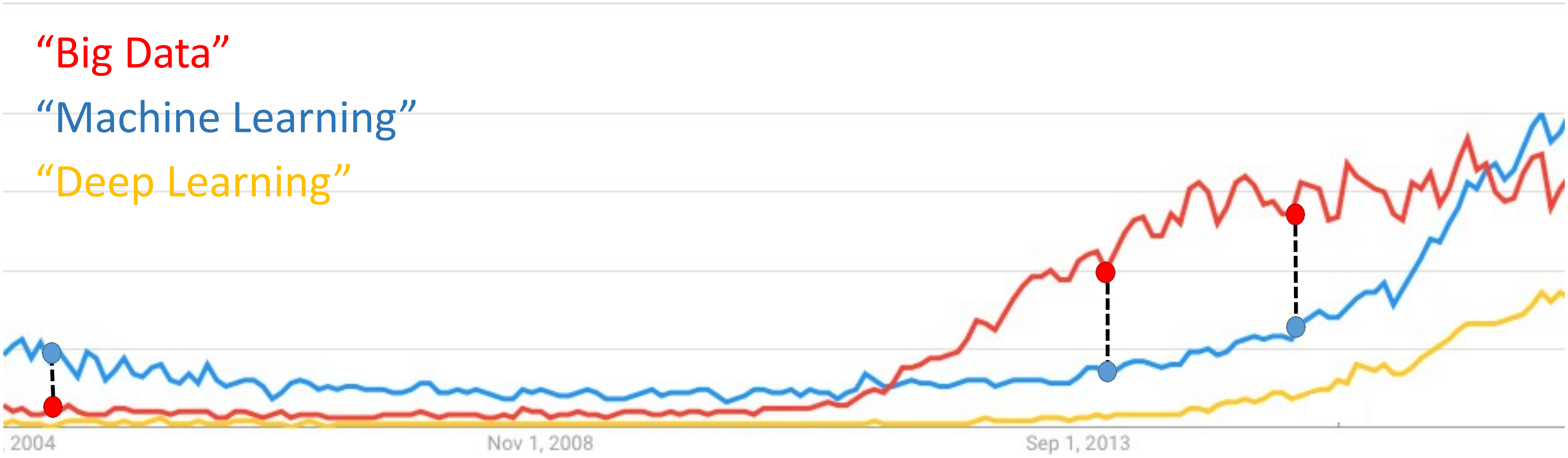
Ira A. Fulton Schools of Engineering
Arizona State University

Euclidean Distance



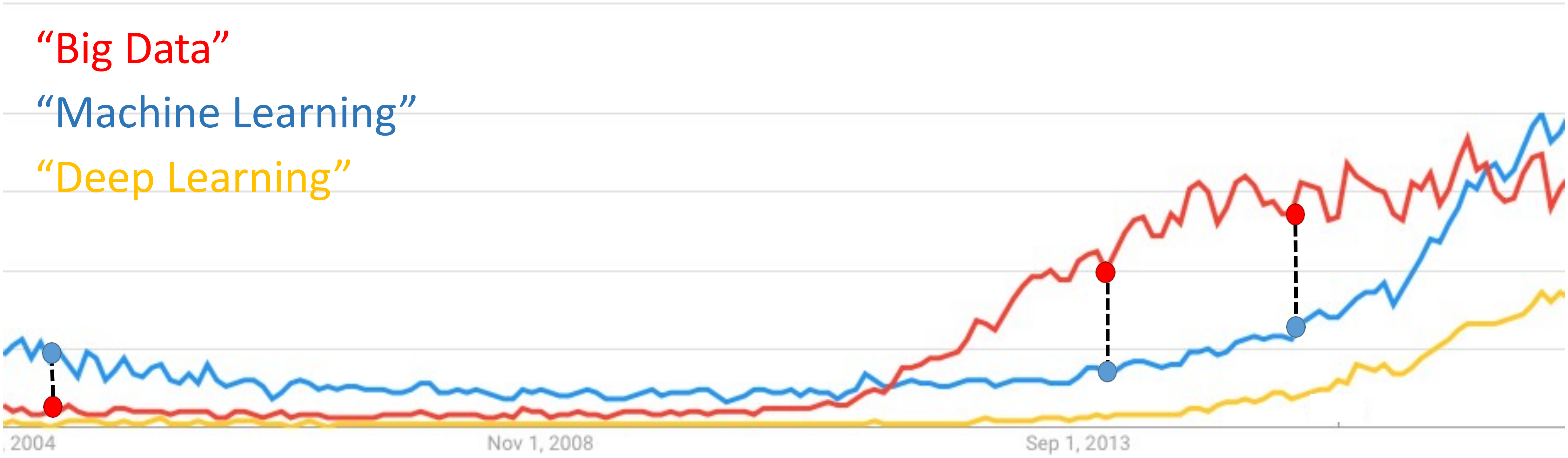
$$\Delta_{Euc}(B,M) = \sqrt{\sum_{i=1..N} \left(B_i - M_i\right)^2}$$

Euclidean Distance



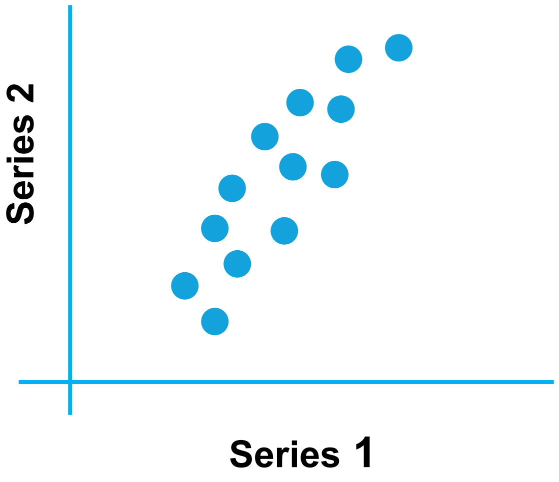
$$\Delta_{Euc}(B, M) = \sqrt{\sum_{i=1..N} (B_i - M_i)^2}$$

Correlation Similarity

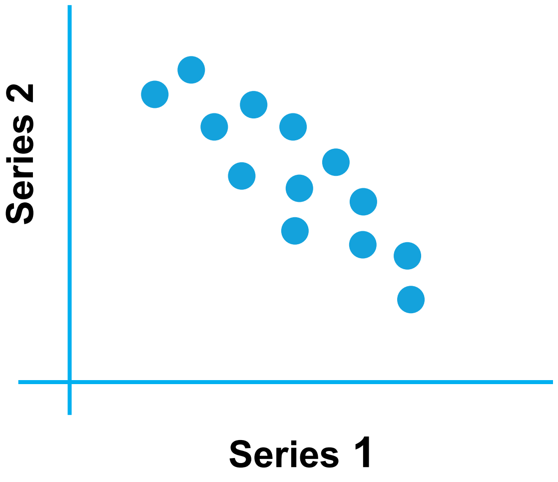


$$\mathrm{Sim}_{correl}(B, M) = \frac{E\left[\left(B - \mu_B\right)\left(M - \mu_M\right)\right]}{\sigma_B \sigma_M}$$

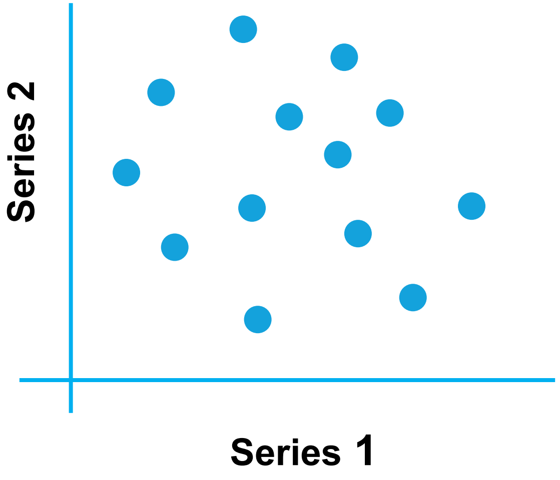
Correlation



positively correlated

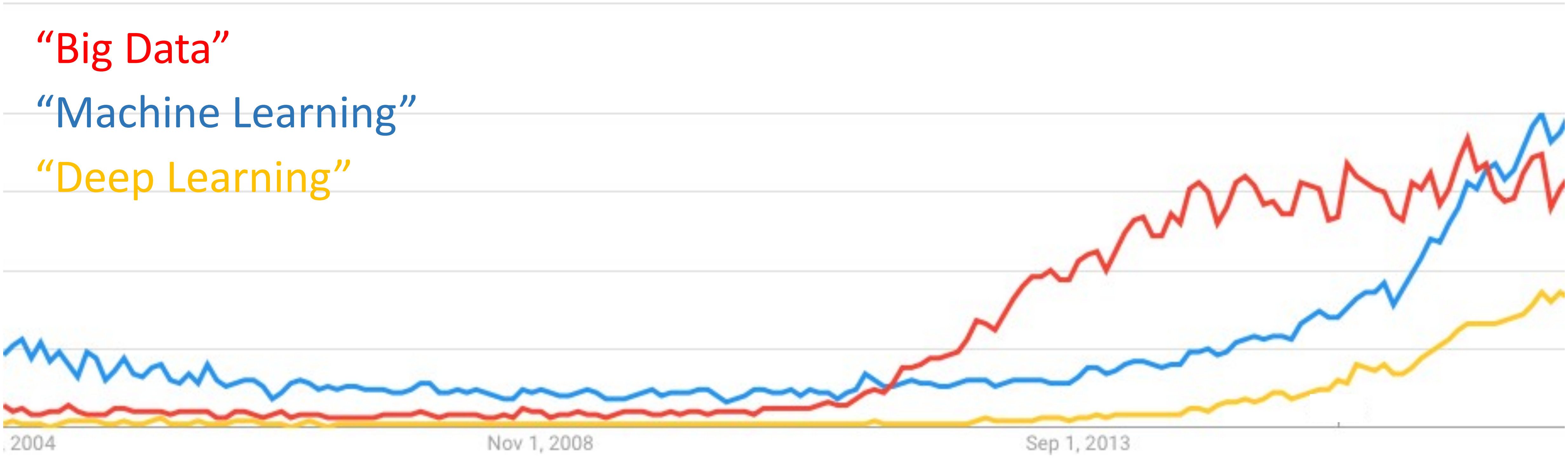


negatively correlated

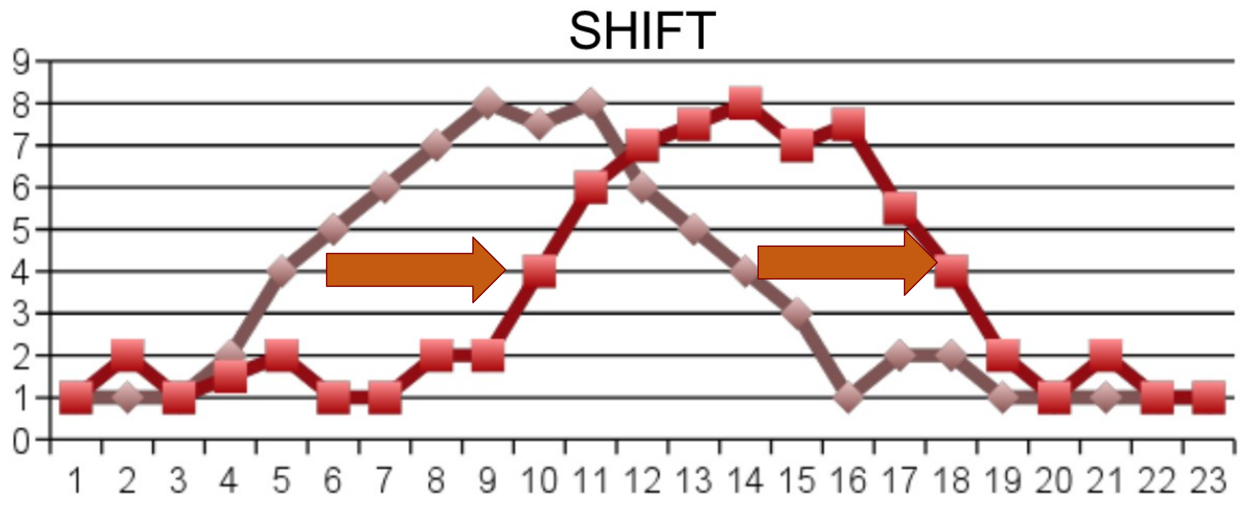


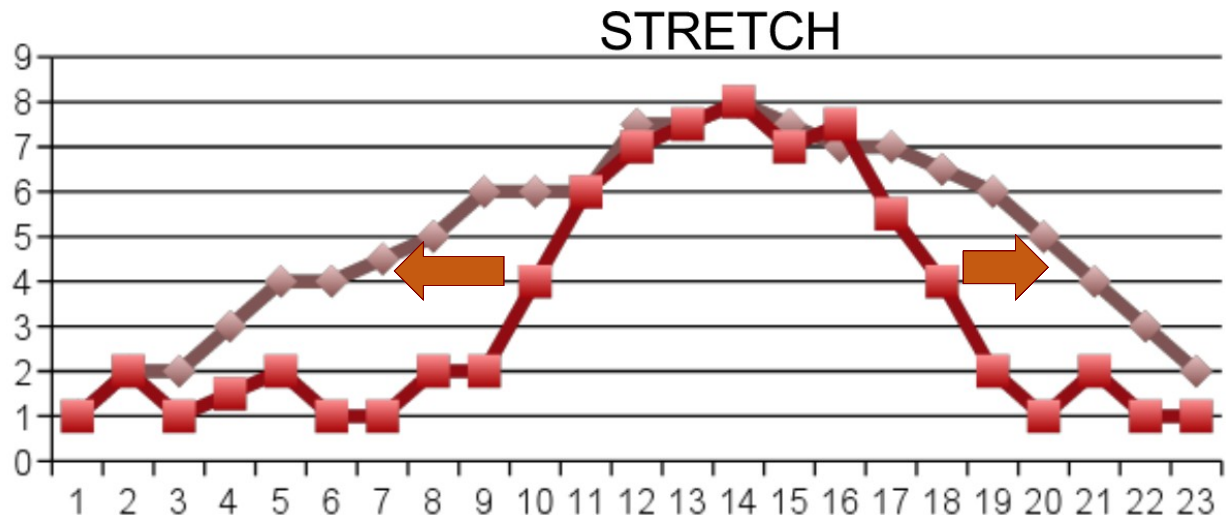
uncorrelated

Comparing time series

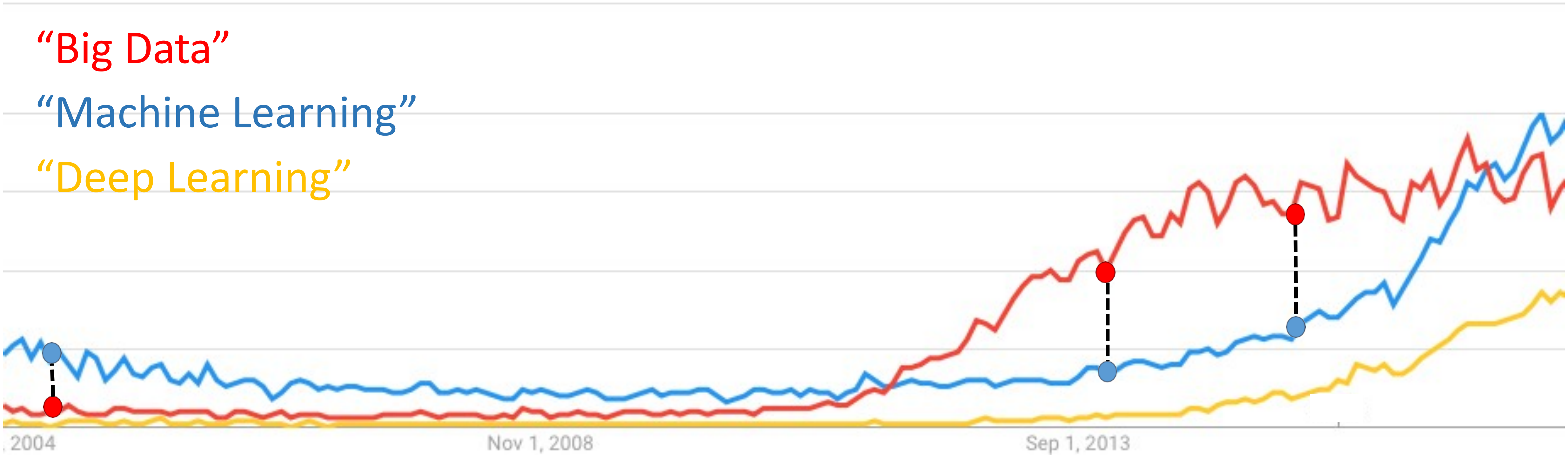


Issues with Synchronized Measures



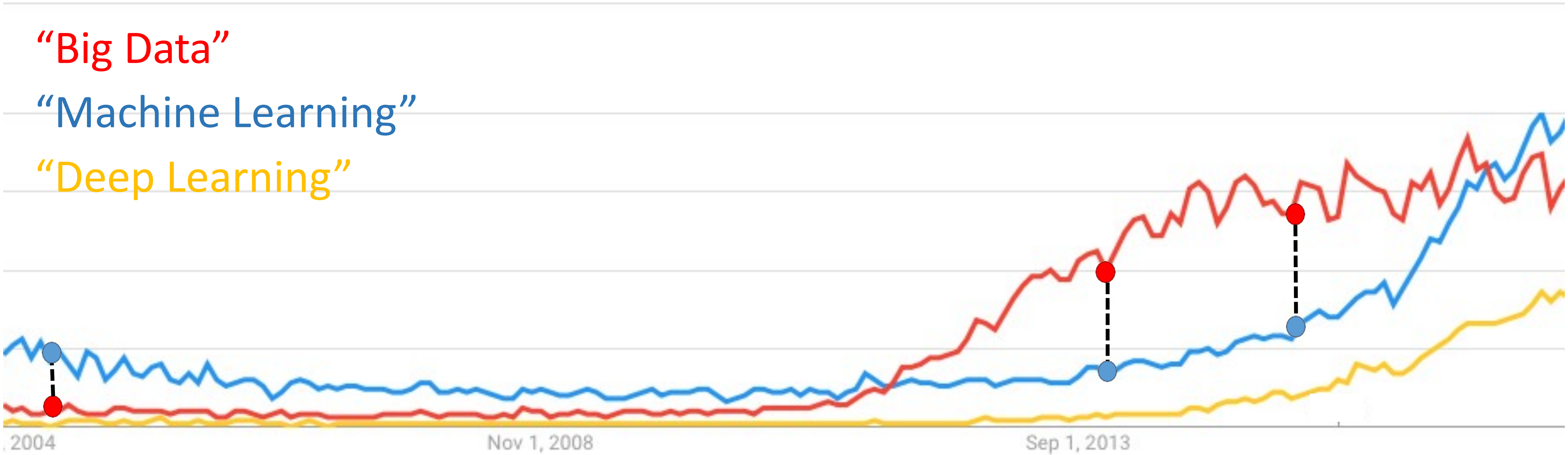


Euclidean Distance



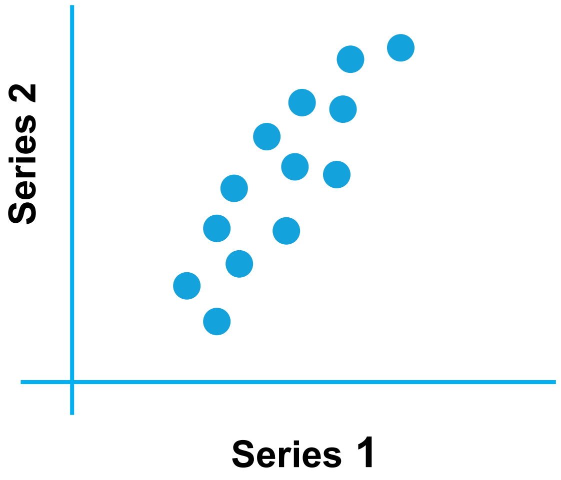
$$\Delta_{Euc}(B, M) = \sqrt{\sum_{i=1..N} (B_i - M_i)^2}$$

Correlation Similarity

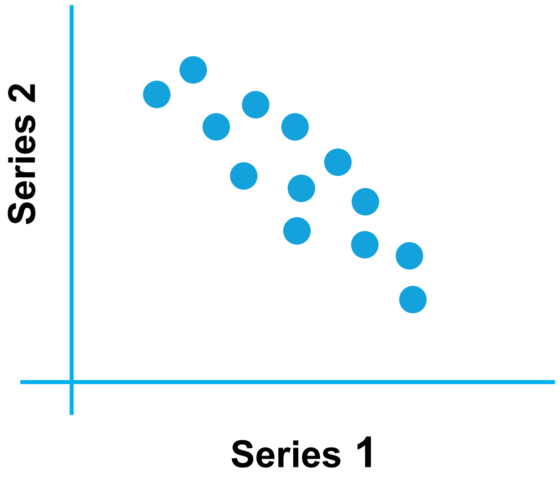


$$\mathrm{Sim}_{correl}(B, M) = \frac{E\left[(B-\mu_B)(M-\mu_M)\right]}{\sigma_B\sigma_M}$$

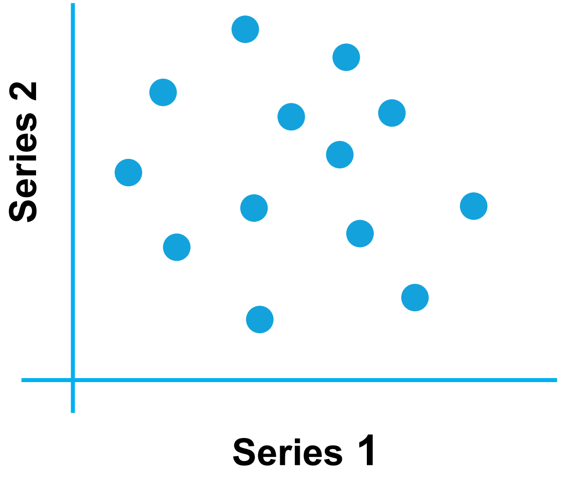
Correlation



positively correlated



negatively correlated



uncorrelated

Reminder: Edit cost

- Let E be a sequence of edit operations to convert one string to another
- Let us associate a cost, C, to each edit operation

> - Costs of edit operations can be different from each other
>   - Type of the operation (replace, delete, insert)
>   - Symbols involved in the operation
>   - Position of the edit operation

- Given a sequence of edit operations, E

$$C(E) = \sum_{e_i \in E} C(e_i)$$

Dynamic Time Warping

- Let us be given two time series, P and Q, of lengths N and M

| D[I,j] = # of edits from length-i prefix of P to length-j prefix of Q

- D[0,j] = infinity; D[i,0] = infinity
- D[0,0] = 0

else $D[i,j] = abs(P_{i,} - Q_j) + \min \{$

$D[i-1,j]$,

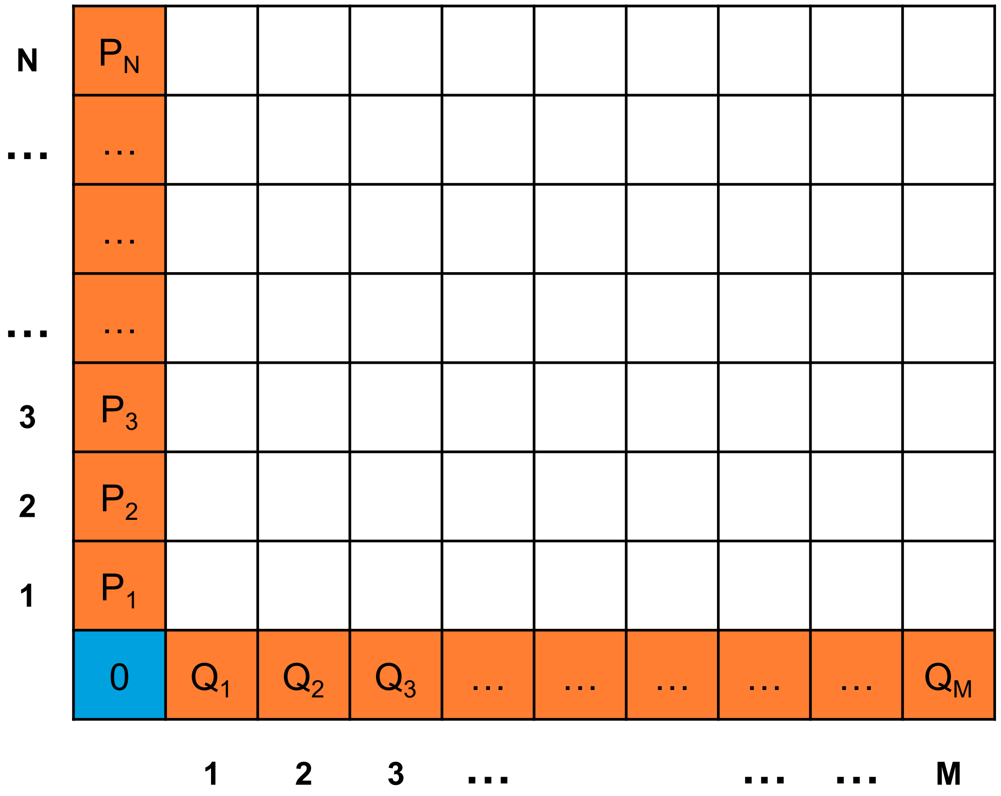
$D[i,j-1]$,

$D[i-1,j-1]$

$\}$

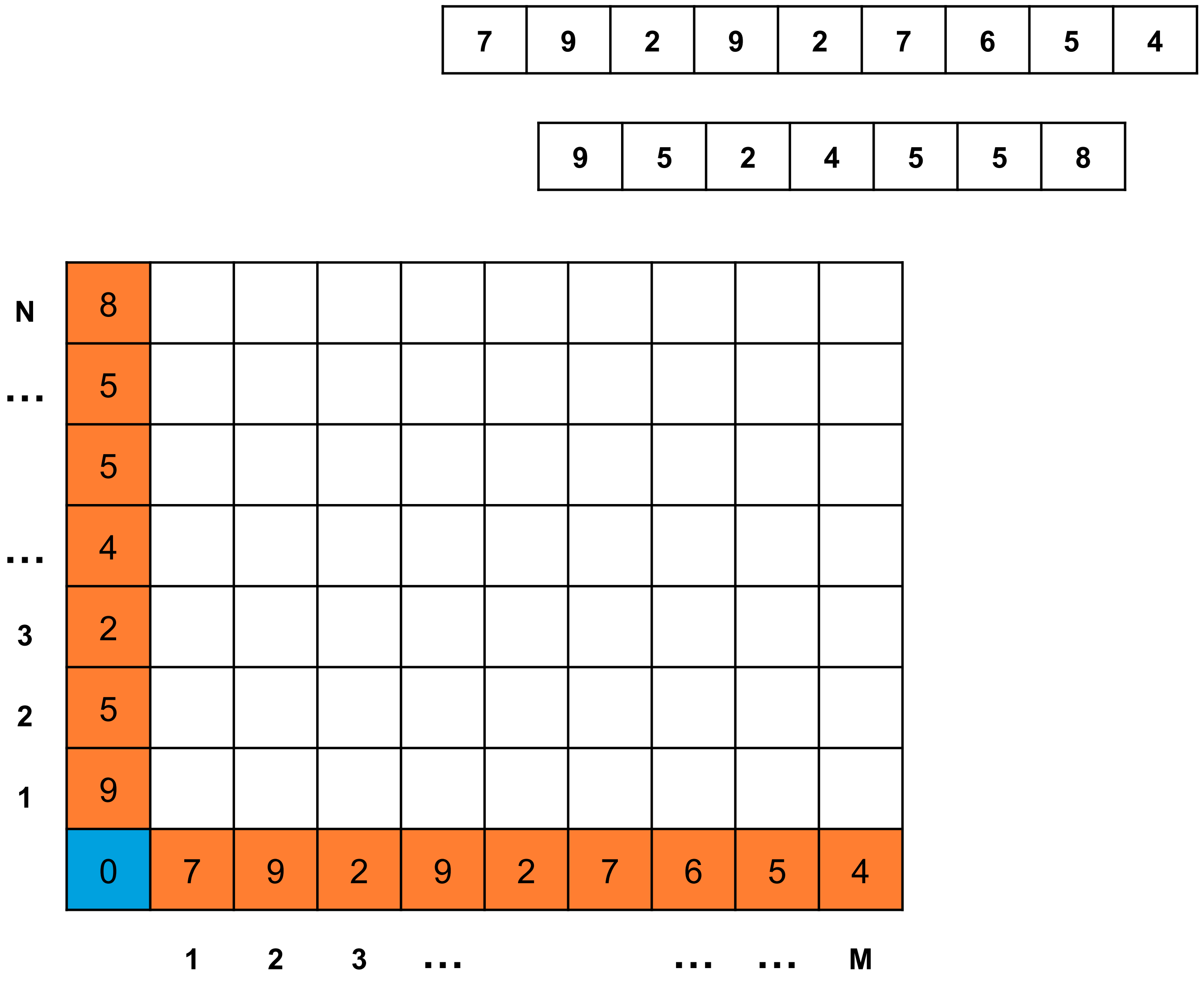
Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation



Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation



Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation

7	9	2	9	2	7	6	5	4

9	5	2	4	5	5	8

N	8	1	1	6	1	6	1	2	3	4
...	5	2	4	3	4	3	2	1	0	1
	5	2	4	3	4	3	2	1	0	1
...	4	3	5	2	5	2	3	2	1	0
3	2	5	7	0	6	0	5	4	3	2
2	5	2	4	3	4	3	2	1	0	1
1	9	2	0	7	0	7	2	3	4	5
	0	7	9	2	9	2	7	6	5	4
		1	2	3	...			...	...	M

Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation

7	9	2	9	2	7	6	5	4

9	5	2	4	5	5	8

N	8	17	1	6	1	6	1	2	3	4
...	5	16	4	3	4	3	2	1	0	1
	5	14	4	3	4	3	2	1	0	1
...	4	12	5	2	5	2	3	2	1	0
3	2	9	7	0	6	0	5	4	3	2
2	5	4	4	3	4	3	2	1	0	1
1	9	2	2	9	9	16	18	21	25	30
	0	7	9	2	9	2	7	6	5	4
		1	2	3	...			...	...	M

Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation

7	9	2	9	2	7	6	5	4

9	5	2	4	5	5	8

N	8	17	1	6	1	6	1	2	3	4
…	5	16	4	3	4	3	2	1	0	1
	5	14	4	3	4	3	2	1	0	1
…	4	12	5	2	5	2	3	2	1	0
3	2	9	7	0	6	0	5	4	3	2
2	5	4	6	3	4	3	2	1	0	1
1	9	2	2	9	9	16	18	21	25	30
	0	7	9	2	9	2	7	6	5	4
		1	2	3	…			…	…	M

Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation

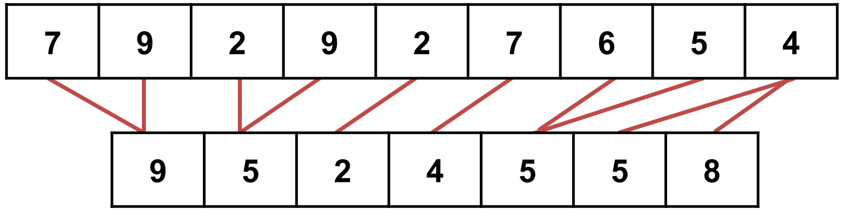
7	9	2	9	2	7	6	5	4

9	5	2	4	5	5	8

N	8	17	17	19	14	20	15	16	17	18
…	5	16	18	13	14	14	15	14	14	14
	5	14	16	10	11	13	13	13	13	14
…	4	12	14	7	10	11	12	14	15	15
3	2	9	11	5	11	9	14	18	18	20
2	5	4	6	5	9	12	14	15	15	16
1	9	2	2	9	9	16	18	21	25	30
	0	7	9	2	9	2	7	6	5	4
		1	2	3	…			…	…	M

Dynamic Time Warping (DTW)

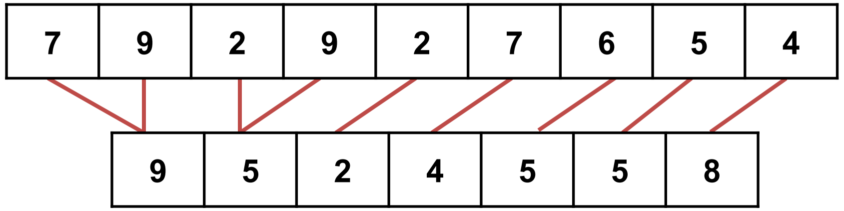
- Complexity: O(M,N)
- Dynamic Programming based implementation



N	8	17	17	19	14	20	15	16	17	18
...	5	16	18	13	14	14	15	14	14	14
	5	14	16	10	11	13	13	13	13	14
...	4	12	14	7	10	11	12	14	15	15
3	2	9	11	5	11	9	14	18	18	20
2	5	4	6	5	9	12	14	15	15	16
1	9	2	2	9	9	16	18	21	25	30
	0	7	9	2	9	2	7	6	5	4
		1	2	3	...			...	...	M

Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation

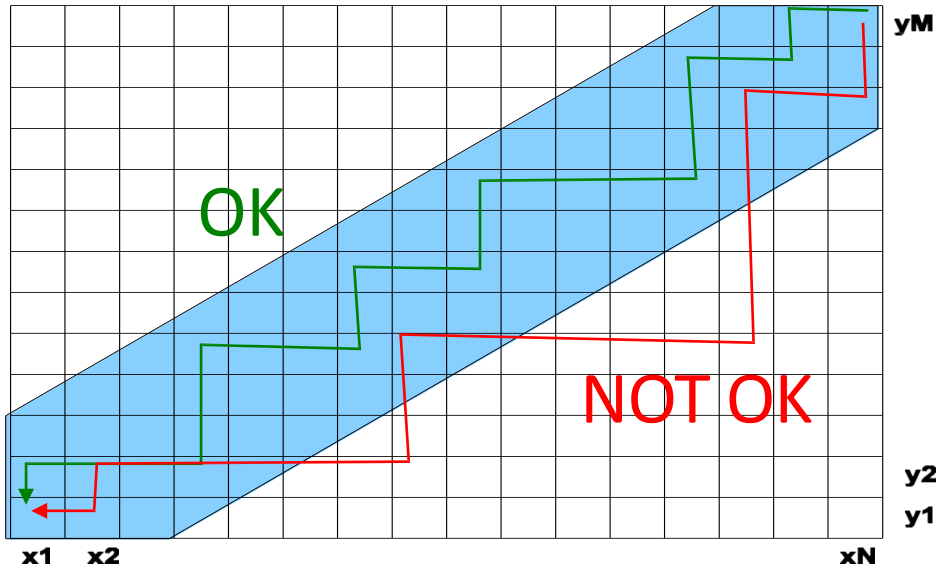


N	8	17	17	19	14	20	15	16	17	18
...	5	16	18	13	14	14	15	14	14	14
	5	14	16	10	11	13	13	13	13	14
...	4	12	14	7	10	11	12	14	15	15
3	2	9	11	5	11	9	14	18	18	20
2	5	4	6	5	9	12	14	15	15	16
1	9	2	2	9	9	16	18	21	25	30
	0	7	9	2	9	2	7	6	5	4
		1	2	3	...			...	...	M

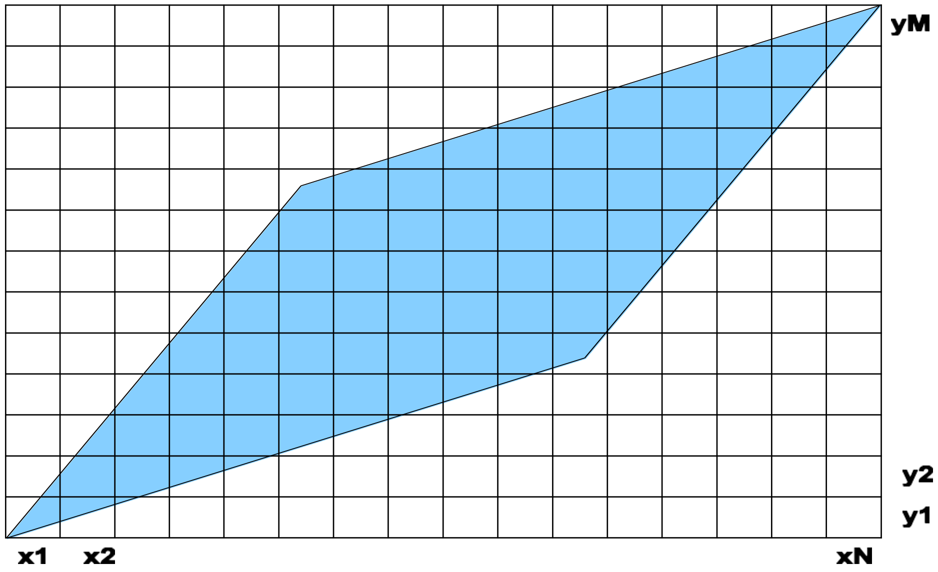
Reducing the Cost of DTW

To reduce the O(NM) cost of filling the grid various heuristics impose constraints on the grid regions through which the warp paths can pass.

Sakoe-Chiba band [1]



Itakura parallelogram [2]

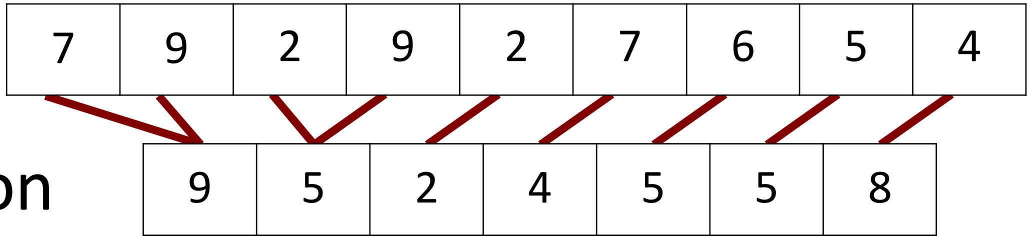


1 Dynamic Programming Algorithm Optimisation for Spoken Word Recognition, 1978

2 F. Itakura. Minimum prediction residual principle applied to speech recognition, 1975

Dynamic Time Warping (DTW)

- Complexity: O(M,N)
- Dynamic Programming based implementation



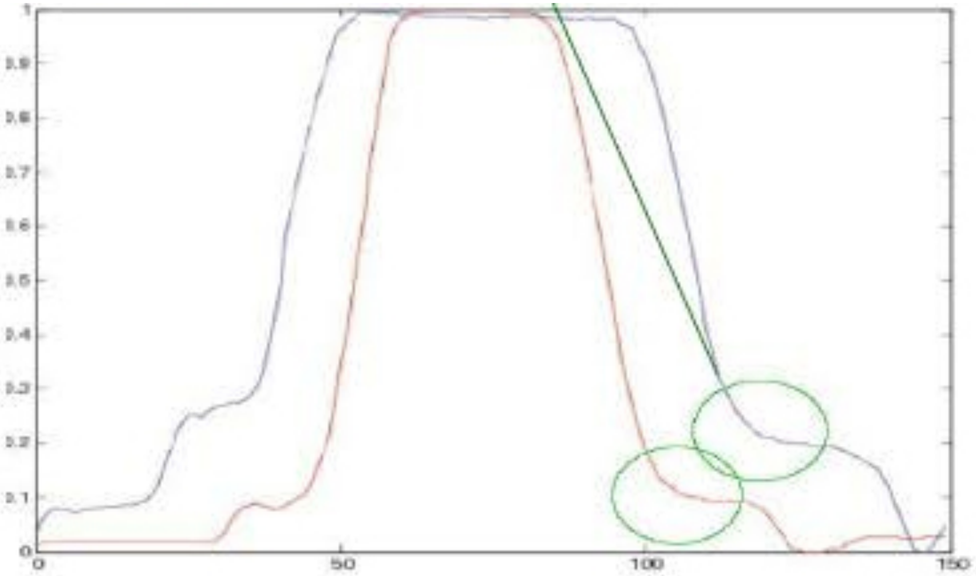
N	**8**	17	17	19	14	20	15	16	17	18
…	**5**	16	18	13	14	14	15	14	14	14
	5	14	16	10	11	13	13	13	13	14
…	**4**	12	14	7	10	11	12	14	15	15
3	**2**	9	11	5	11	9	14	18	18	20
2	**5**	4	6	5	9	12	14	15	15	16
1	**9**	2	2	9	9	16	18	21	25	30
	0	**7**	**9**	**2**	**9**	**2**	**7**	**6**	**5**	**4**
		1	**2**	**3**	**…**			**…**	**…**	**M**

sDTW

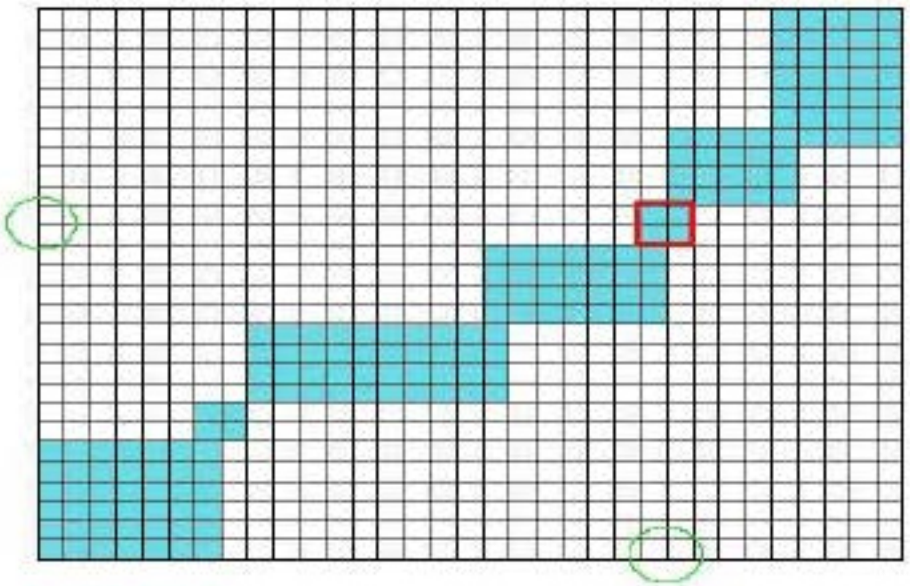
K.S.

Time series often carry **temporal features** that can be used for identifying **locally relevant constraints** to eliminate redundant work in an **adaptive** manner.

Sample (matching) structural features of two time series



Two time series



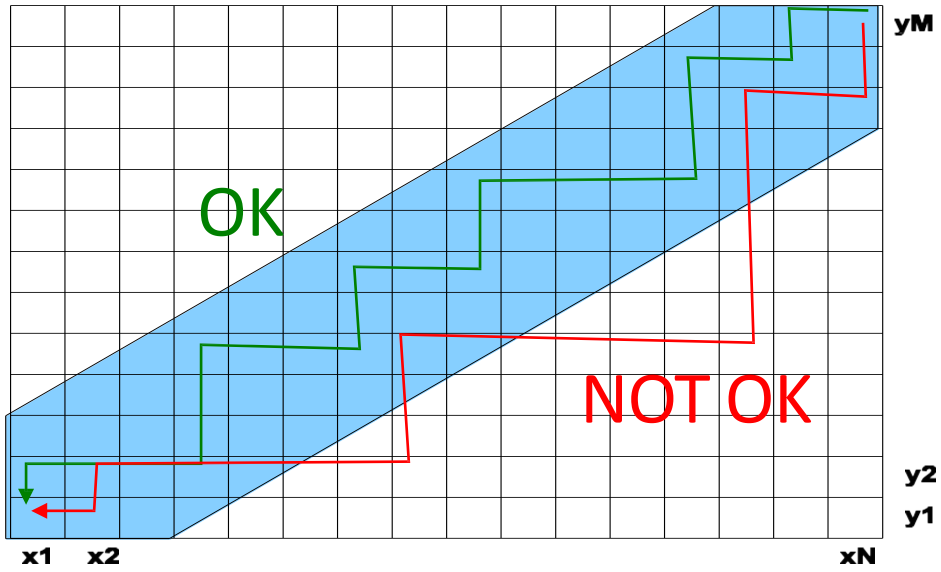
Adaptive constraints on the DTW grid

1 Dynamic Programming Algorithm Optimisation for Spoken Word Recognition, 1978
2 F. Itakura. Minimum prediction residual principle applied to speech recognition, 1975

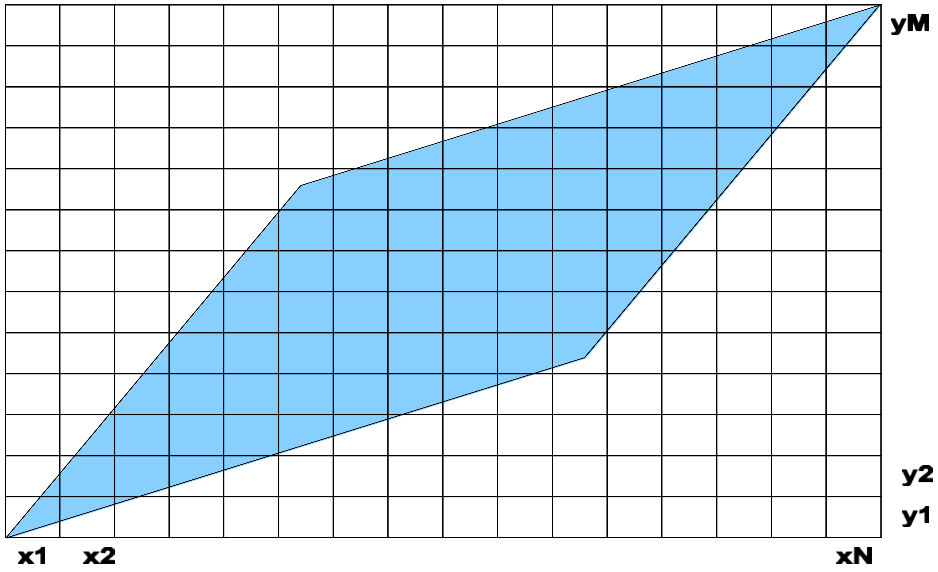
Reducing the Cost of DTW

To reduce the O(NM) cost of filling the grid various heuristics impose **constraints** on the grid regions through which the warp paths can pass.

Sakoe-Chiba band [1]



Itakura parallelogram [2]



1 Dynamic Programming Algorithm Optimisation for Spoken Word Recognition, 1978
2 F. Itakura. Minimum prediction residual principle applied to speech recognition, 1975

Time series are similar to sequences

- A *string* or *sequence*, $S = (c_1, c_2, \ldots c_N)$, *is a finite sequence of symbols*.

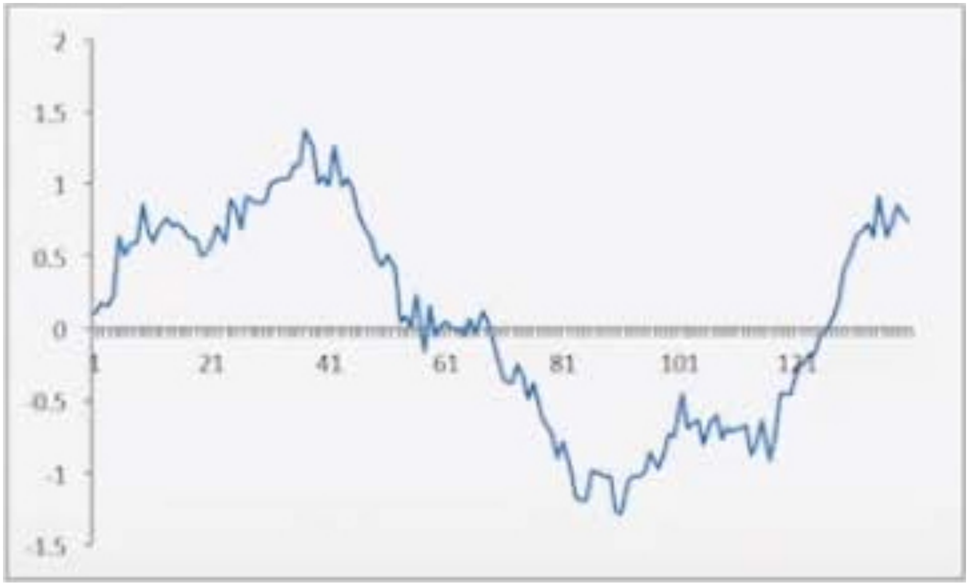
abcbbbaabbaabcbbbaaabbc

- A *time series*, $T = (d_1, d_2, \ldots, d_N)$, *is a finite sequence of data values*.

https://trends.google.com/trends/explore?q=bg%20data

SAX (Symbolic Aggregate ApproXimation

- Time series are similar to sequences
- Can we transform a time series into a compact sequence representation?

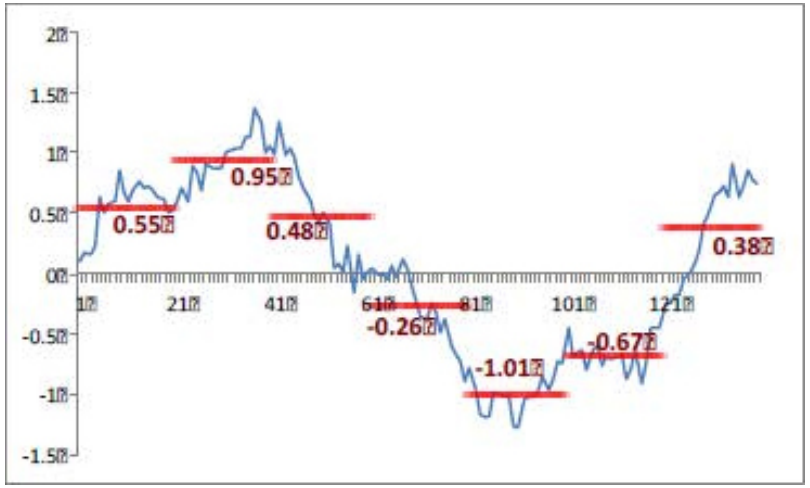


SAX (Symbolic Aggregate ApproXimation

Time series are similar to sequences

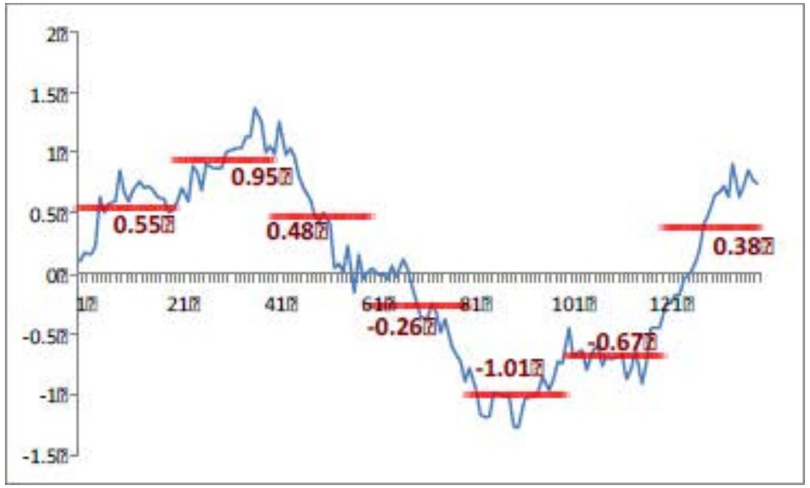
Transform a time series into a compact sequence representation

- Divide the time series into w-length (non-overlapping) windows
- For each window,
  - compute the average amplitude



SAX (Symbolic Aggregate ApproXimation

- Time series are similar to sequences
- Transform a time series into a compact sequence representation
  - Divide the time series into w-length (non-overlapping) windows
  - For each window,
    - compute the average amplitude
    - assign a symbol from a dictionary with s symbols representing this average amplitude



Range	Symbol
[0.9,2]	A
[0.3,0.9]	B
[0.0,0.3]	C
[0.0,-0.3]	D
[-0.3,-0.9]	E
[-0.9,-2]	F

SAX (Symbolic Aggregate ApproXimation

Time series are similar to sequences

Transform a time series into a compact sequence representation

- Divide the time series into w-length (non-overlapping) windows
- For each window,
  - compute the average amplitude
  - assign a symbol from a dictionary with s symbols representing this average amplitude

Note that, SAX reduces

- **Temporal resolution**, by dividing the string into windows (length ~ N/w)
- **Amplitude resolution**, by using only one of the s symbols per window

Summary

- Synchronized measures (Euclidean distance, correlation) are relatively cheap to compute, but may not account for shifts, stretches, and other temporal misalignments
- Asynchronous approaches (edit distance, DTW) can account for misalignments, but are expensive. Solution include
  - constraining the search space
  - creation of reduced representations of the time series

Sequences and Time Series

Time Series Motifs

K. Selçuk Candan, Professor of Computer Science and Engineering

CASCADE
ASU Center for Assured and SCAlable Data Engineering

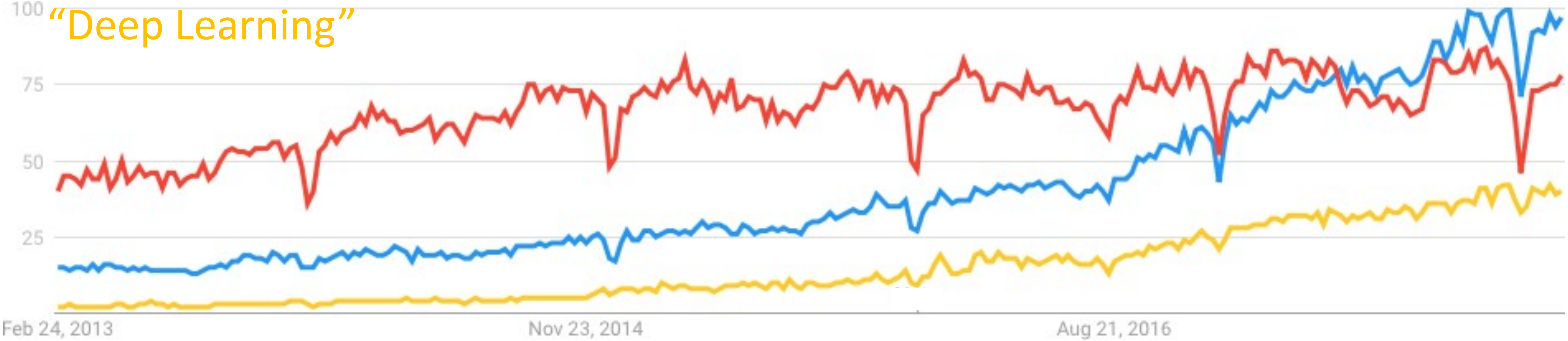
Ira A. Fulton Schools of Engineering
Arizona State University

Motifs

"Big Data"

"Machine Learning"

"Deep Learning"

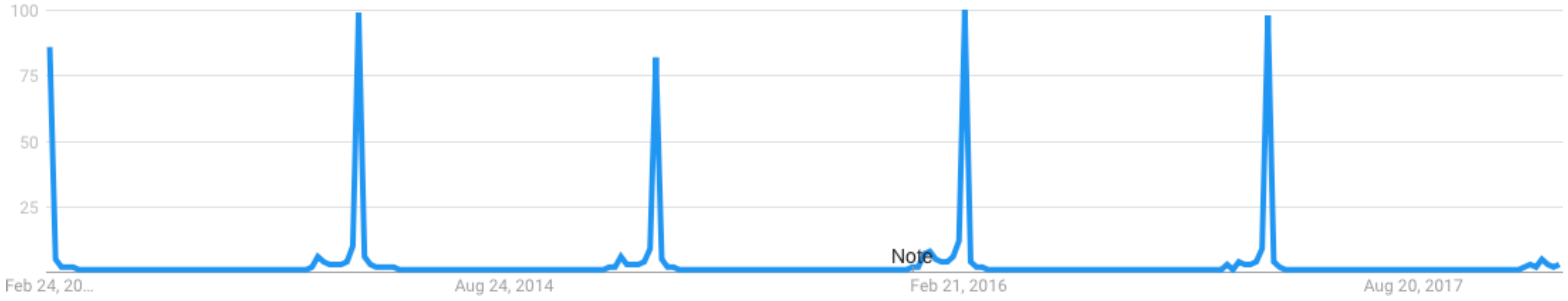


Motif Search Algorithm

- General approach
  - Enumerate subsequences of the time series (with a given length)
  - Apply a clustering algorithm to identify groups of similar subsequences

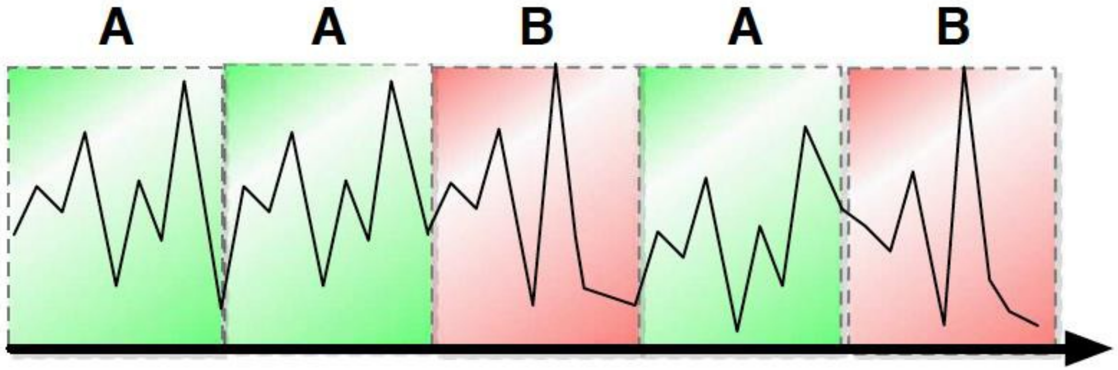
Motif Quality

"Oscars"

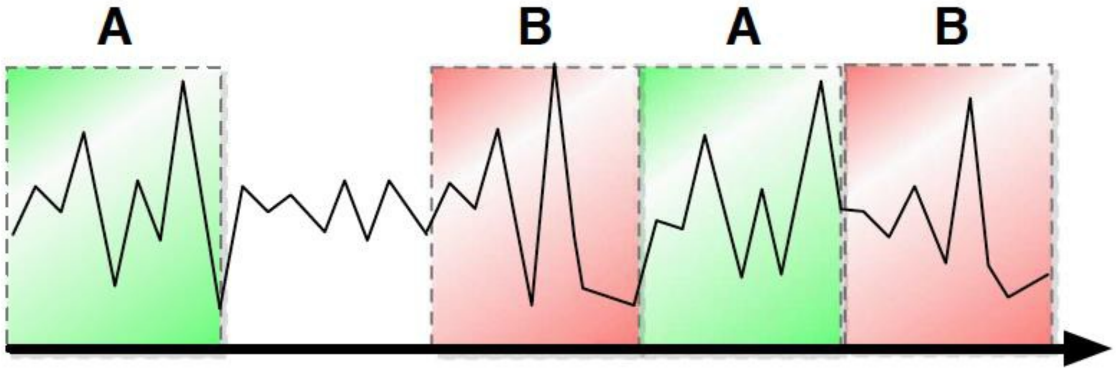


"Politics"

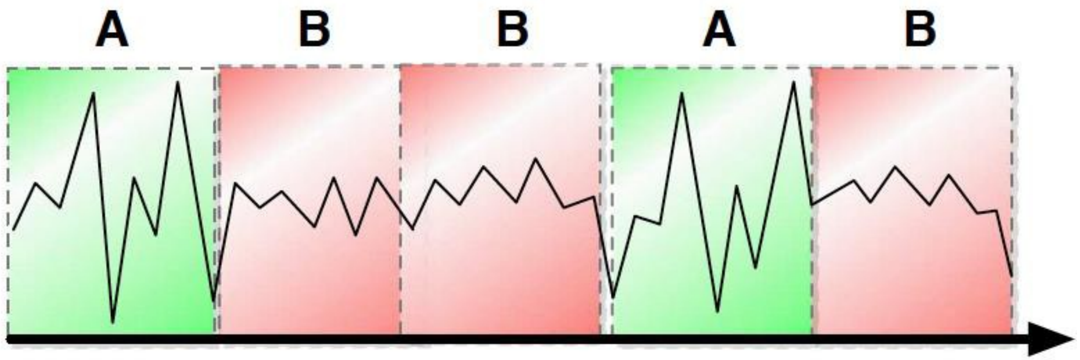
Motifs



(a) motif A has a higher support



(b) motif A has smaller distortion/higher purity



(c) motif A has higher dynamicity

Motif Search Algorithm

- General approach
  - Enumerate subsequences of the time series (with a given length)
  - Apply a clustering algorithm to identify groups of similar subsequences
  - Eliminate clusters with
    - too few subsequences (not enough support)
    - too imperfect matches (not well defined motifs)
    - too low dynamicity (patterns that do not carry much useful information or are not interesting)

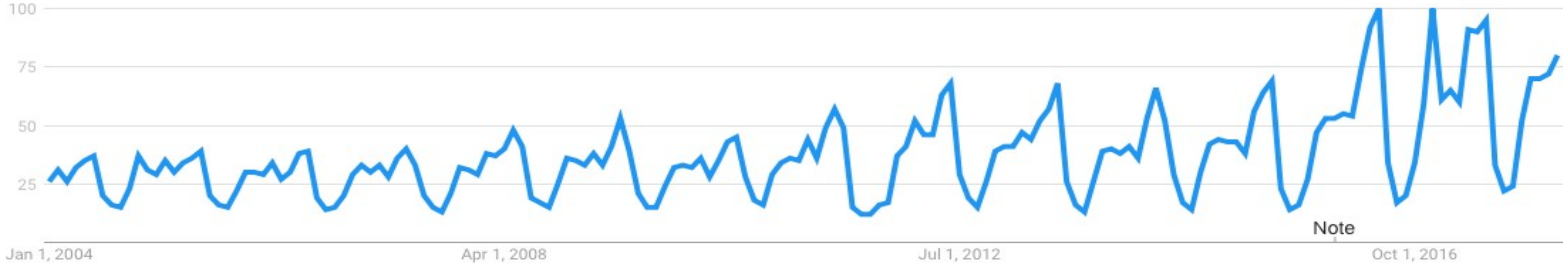
Motif Search Algorithm

- General approach
  - Enumerate subsequences of the time series (with a given length)
    - all subsequences
    - subsequences that are sufficiently different from their immediate neighborhood
    - subsequences that have high dynamicity
    - subsequences that have at least one other similar subsequence in the series
  - Apply a clustering algorithm to identify groups of similar subsequences
  - Eliminate clusters with
    - too few subsequences (not enough support)
    - too imperfect matches (not well defined motifs)
    - too low dynamicity (patterns that do not carry much useful information or are not interesting)

Other questions

- Do all instances of the motif need to be of the same (average) amplitude?

“NBA”



“Olympics”

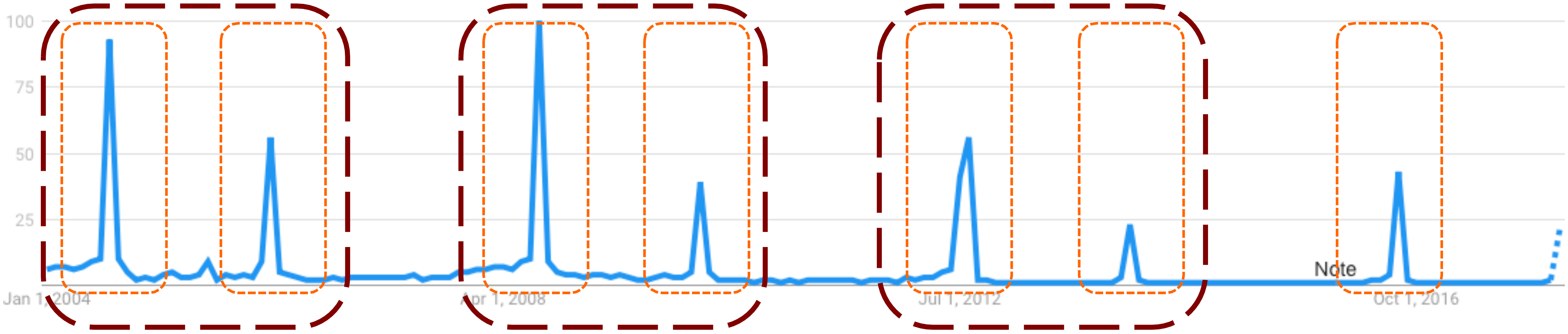
Motif Search Algorithm

- General approach
  - Enumerate subsequences of the time series (with a given length)
    - all subsequences
    - subsequences that are sufficiently different from their immediate neighborhood
    - subsequences that have high dynamicity
    - subsequences that have at least one other similar subsequence in the series
  - Apply a clustering algorithm to identify groups of similar subsequences
    - allow for amplitude differences
  - Eliminate clusters with
    - too few subsequences (not enough support)
    - too imperfect matches (not well defined motifs)
    - too low dynamicity (patterns that do not carry much useful information or are not interesting)

Other questions

- What is the motif length?
- Do all instances of a motif need to have the same length?

“Olympics”



Motif Search Algorithm

- General approach
  - Enumerate subsequences of the time series (with different lengths)
    - all subsequences
    - subsequences that are sufficiently different from their immediate neighborhood
    - subsequences that have high dynamicity
    - subsequences that have at least one other similar subsequence in the series
  - Apply a clustering algorithm to identify groups of similar subsequences
    - allow for amplitude differences
    - account for length differences
  - Eliminate clusters with
    - too few subsequences (not enough support)
    - too imperfect matches (not well defined motifs)
    - too low dynamicity (patterns that do not carry much useful information or are not interesting)

Summary

- Motif search
  - Divide series into sub-sequences
  - Eliminate un-interesting sub-sequences
  - Apply clustering algorithm
  - Eliminate un-interesting clusters

- Challenge
  - There are many sub-sequences
  - How to define what is an interesting sub-sequence?
  - How to define what is an interesting cluster?