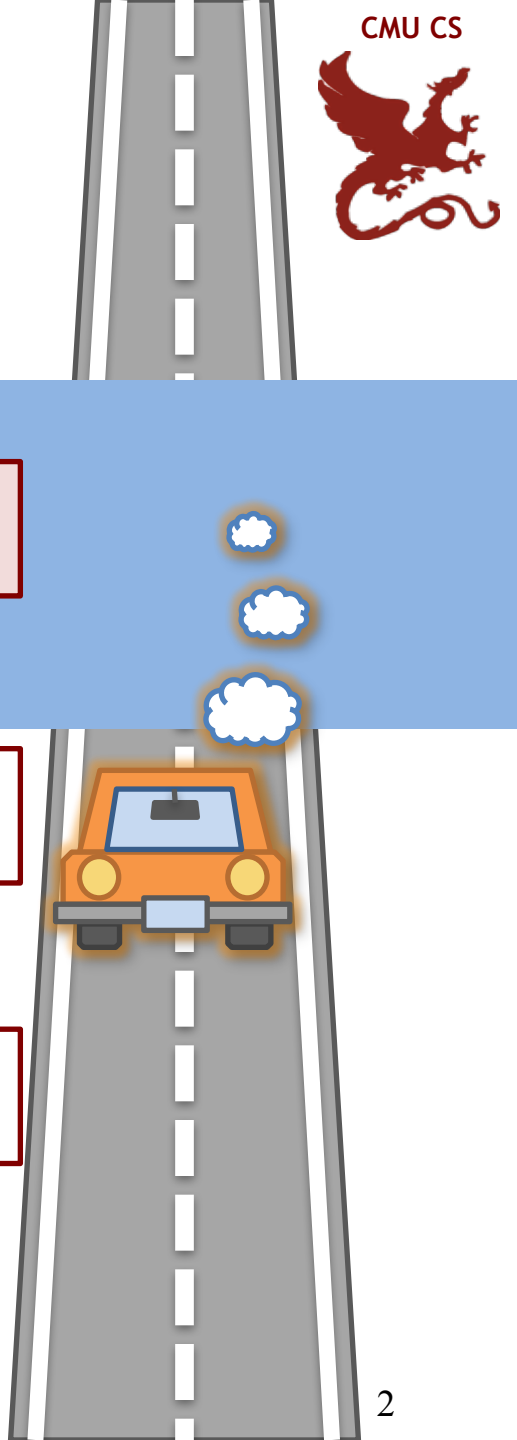
Smart Analytics for Big Time-series Data

Yasushi Sakurai (Kumamoto University)
Yasuko Matsubara (Kumamoto University)
Christos Faloutsos (Carnegie Mellon University)

Roadmap

- Motivation
- **Similarity search, pattern discovery and summarization** — **Part 1**
- Non-linear modeling and forecasting — **Part 2**
- Extension of time-series data: tensor analysis — **Part 3**



Part 1

Similarity search, pattern discovery and summarization

Yasushi Sakurai (Kumamoto University)
Yasuko Matsubara (Kumamoto University)
Christos Faloutsos (Carnegie Mellon University)

Part 1 - Roadmap

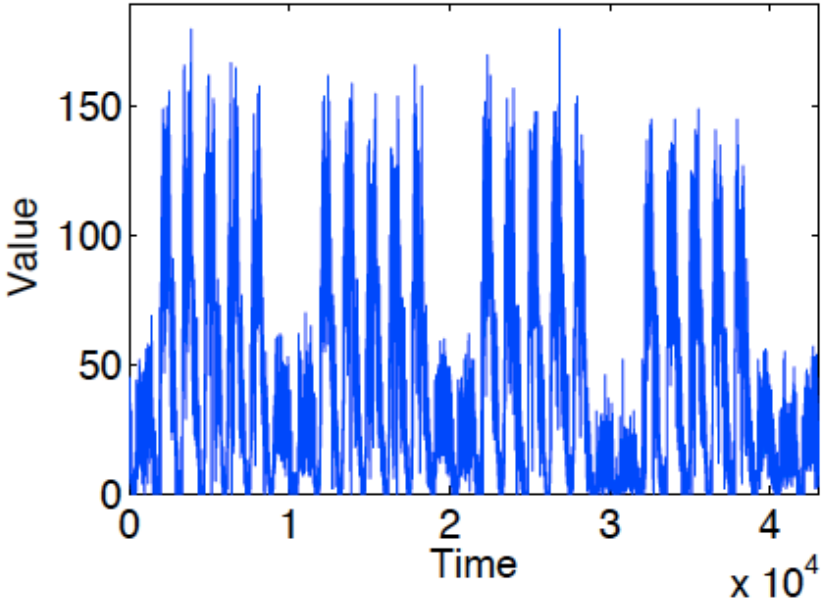
- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Motivation - Applications

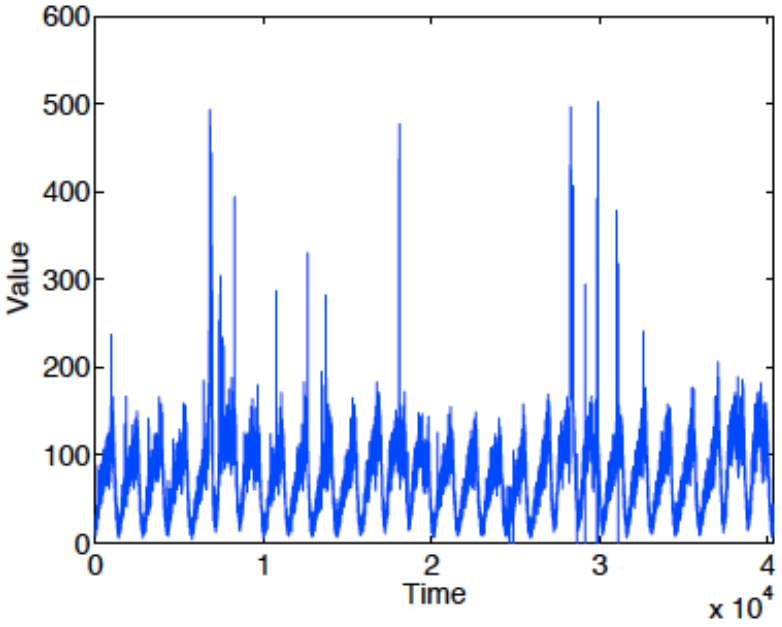
- Web online activities
  - Web access logs
  - Search volume
  - Online reviews
- IoT device data
- Medical, healthcare data

Motivation - Applications

- Web access logs



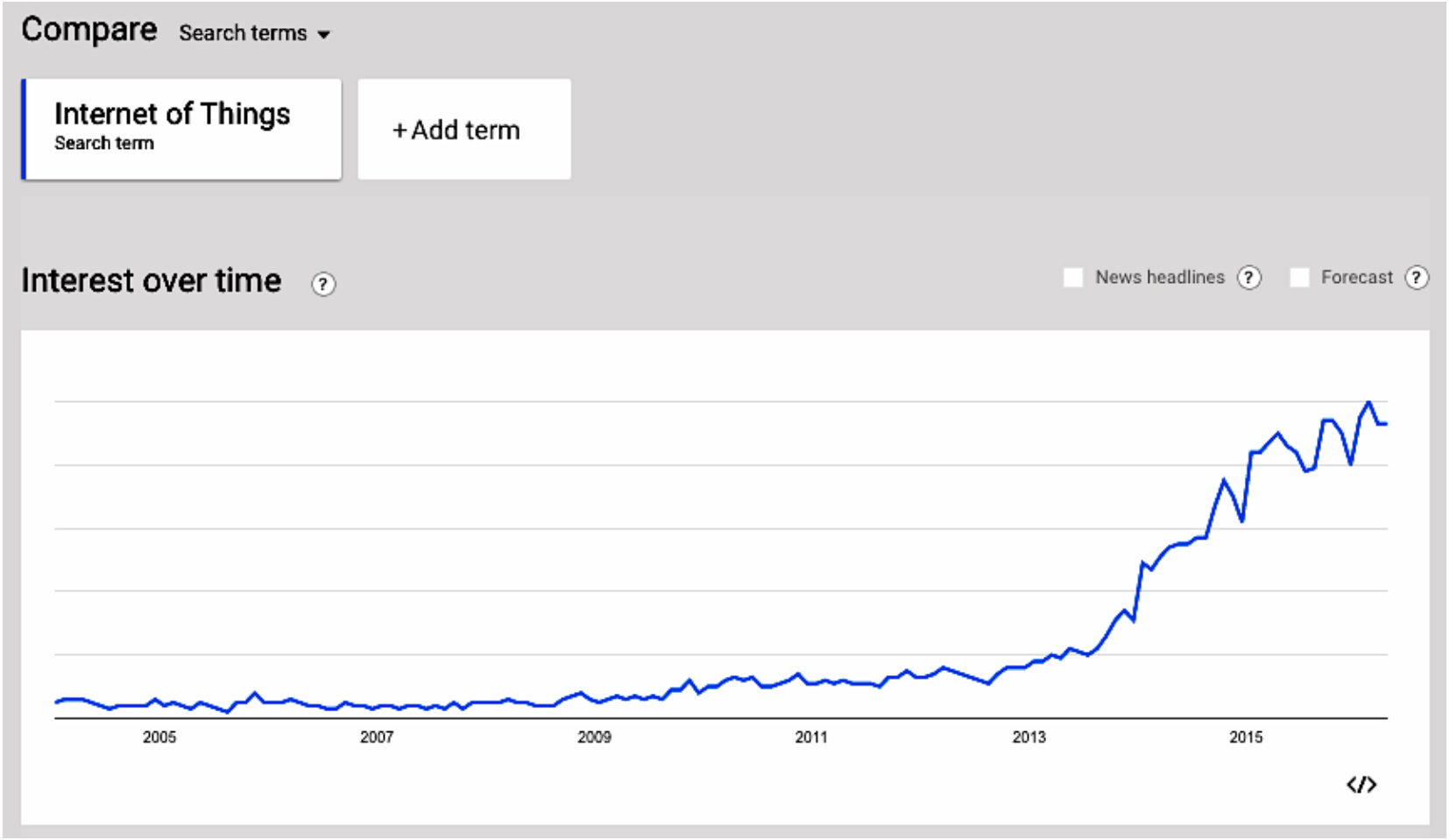
Web clicks (business news site)



Ondemand TV (access count of users)

Motivation - Applications

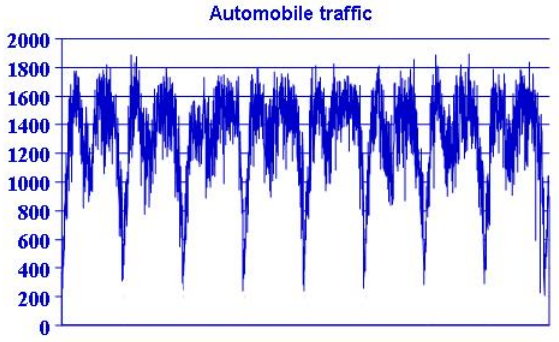
- Web search volume from Google trends



Motivation - Applications

- IoT (Internet of Things) device data
  - Civil/automobile infrastructure
  - Bridge vibrations [Oppenheim+02]
  - Road conditions / traffic monitoring
  - Environmental data (air/water pollutant monitoring)

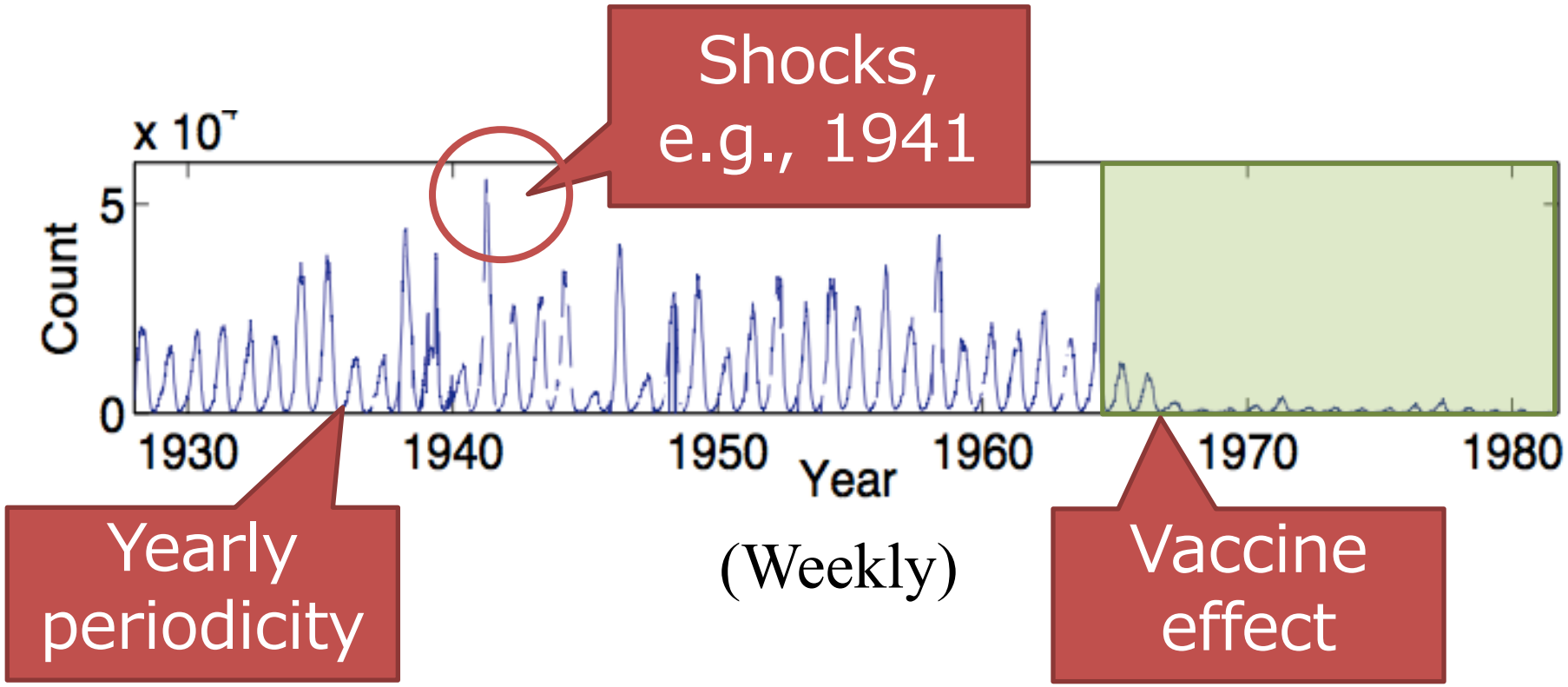
Tokyo Gate Bridge



Motivation - Applications

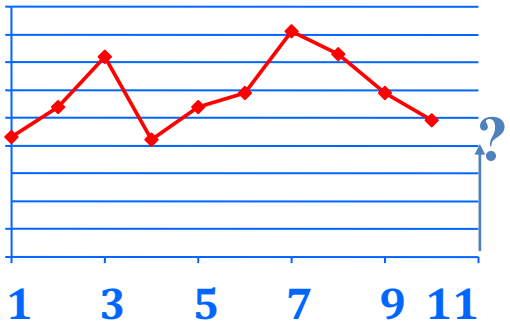
- Medical (epidemic) time-series data

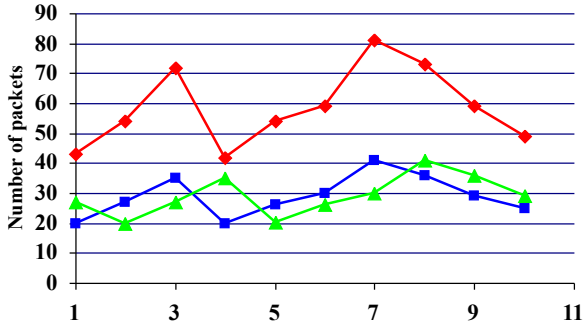
  e.g., measles cases in the U.S.



Wish list

- Problem 1: find patterns/**rules**
- Problem 2: **forecast**
- Problem 3: find patterns/rules/forecast, with **many** time sequences



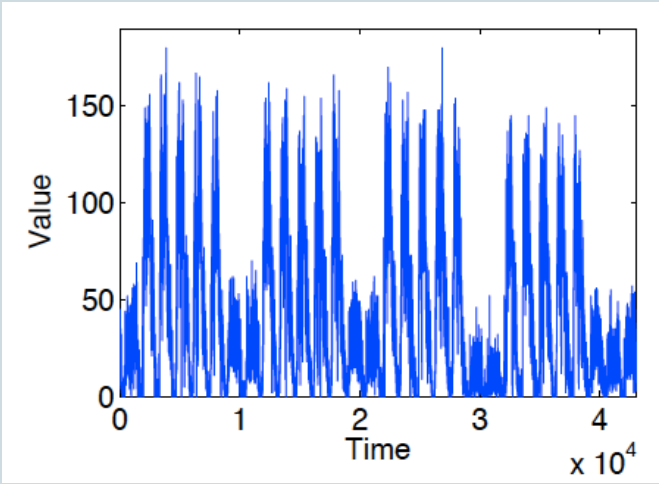


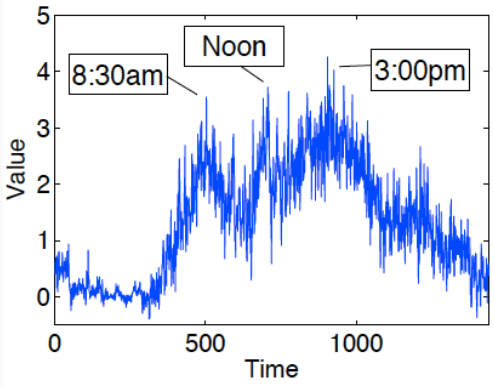
Problem #1

Given: time-series data (e.g., #clicks over time)

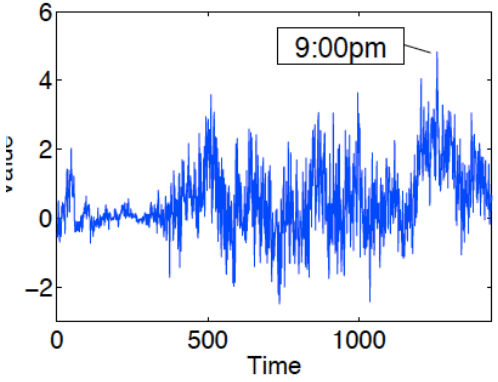
Find: patterns, periodicities, and/or compress

Original web-click sequence





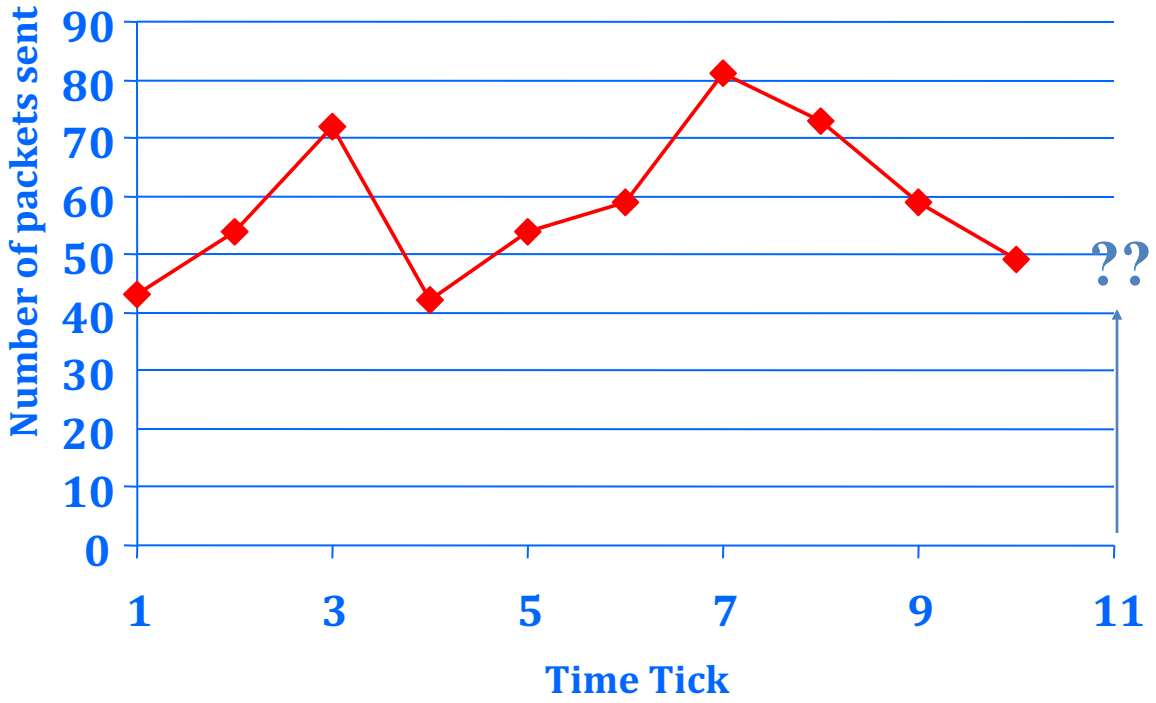
Weekday component



Weekend component

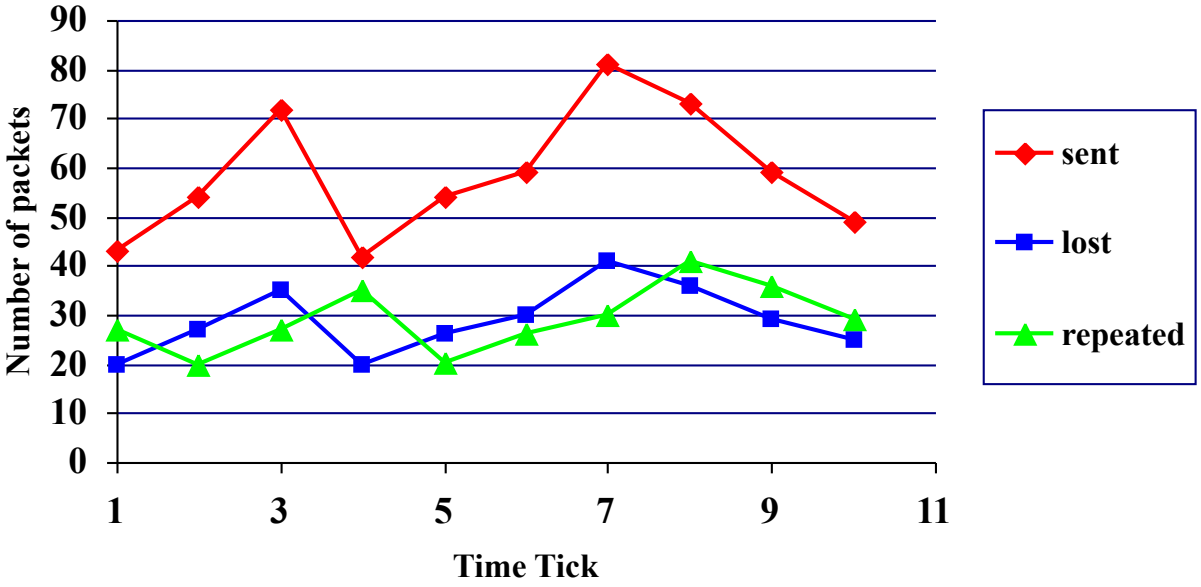
Problem #2

Given x_t, x_{t-1}, …, forecast x_{t+1}



Problem #3

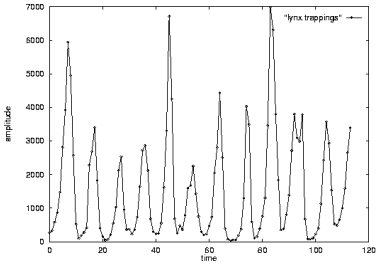
- **Given**: A set of **correlated** time sequences
- **Forecast** '**Repeated(t)**'



Important observations

Patterns, outliers, modeling, forecasting and similarity indexing are closely related:

- For forecasting, we need
  - patterns/rules/models
  - similar past settings
- For outliers, we need to have forecasts
  - (outlier = too far away from our forecast)



Important topics NOT in this tutorial:

- Continuous queries
  - [Babu+Widom] [Gehrke+] [Madden+]
- Categorical data streams
  - [Hatonen+96]
- Outlier detection (discontinuities)
  - [Breunig+00]

Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

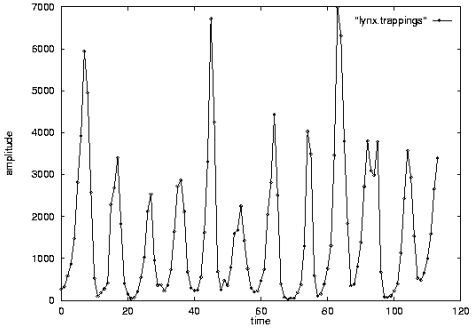
Roadmap

- Motivation
- Similarity Search and Indexing
  - distance functions: Euclidean, time-warping
  - indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Importance of distance functions

Subtle, but **absolutely necessary**:

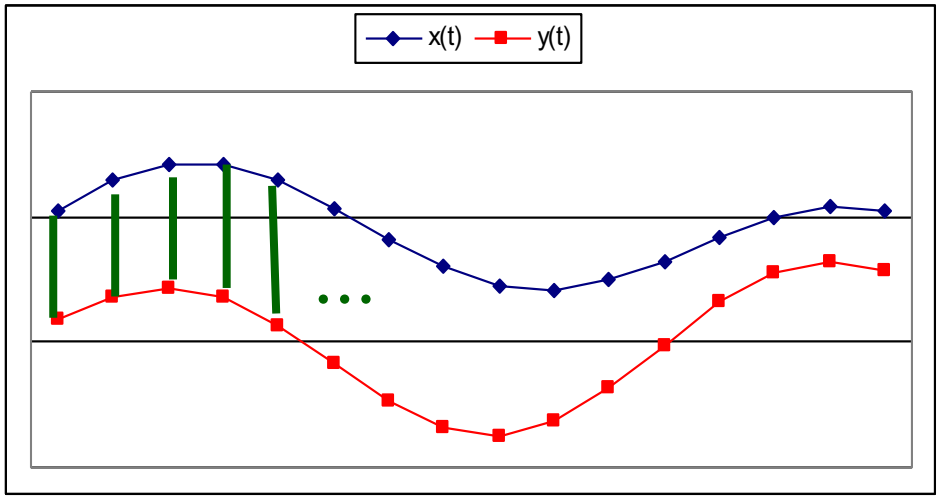
- A 'must' for similarity search, indexing and clustering



Two major families

- Euclidean and Lp norms
- Time warping and variations

Euclidean and Lp

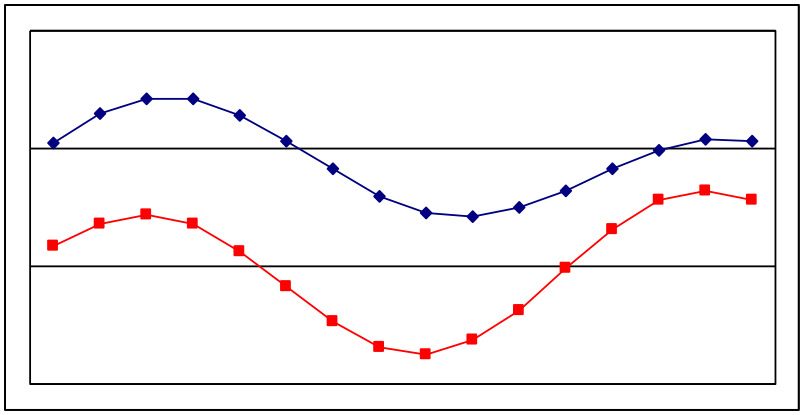


$$D(\vec{x},\vec{y}) = \sum_{i=1}^{n} (x_i - y_i)^2$$

$$L_p(\vec{x},\vec{y}) = \sum_{i=1}^{n} | x_i - y_i |^p$$

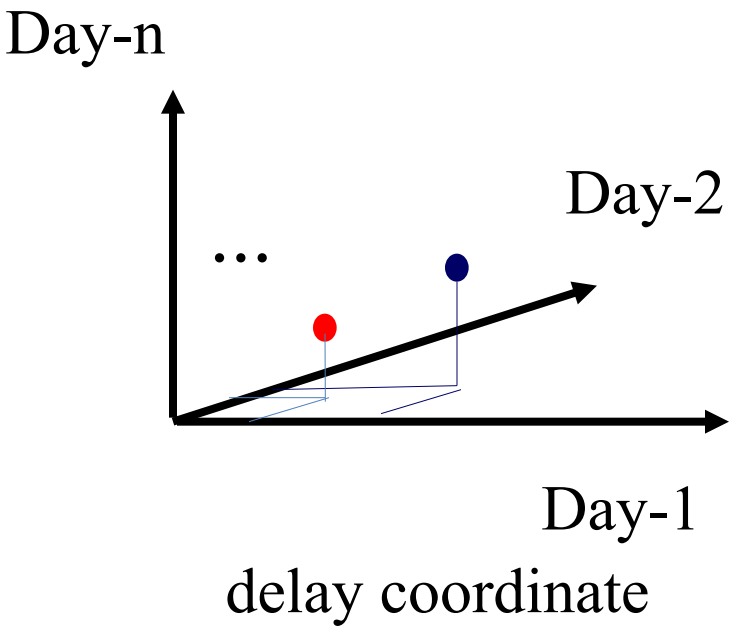
- L_1: city-block = Manhattan
- L_2 = Euclidean
- L_∞

Observation #1



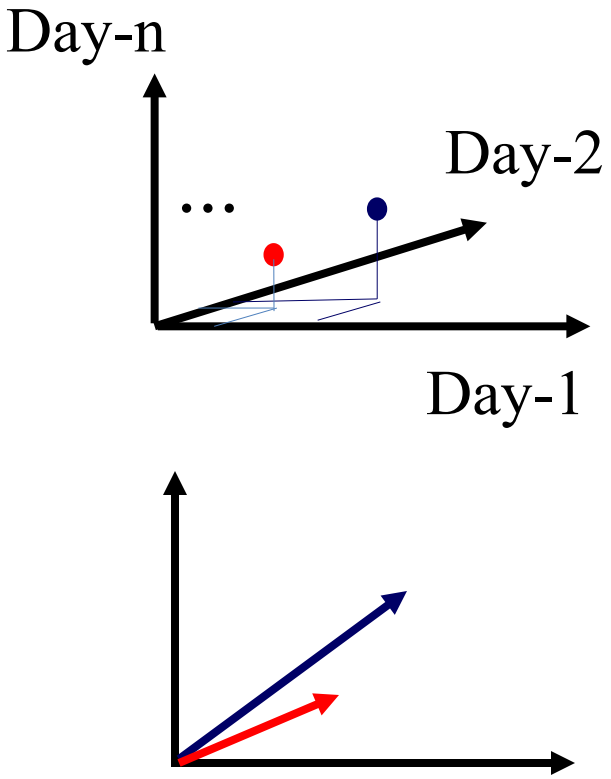
- Time sequence

  -> n-dimensional vector



Observation #2

- Euclidean distance is closely related to
  - cosine similarity
  - dot product
  - 'cross-correlation' function

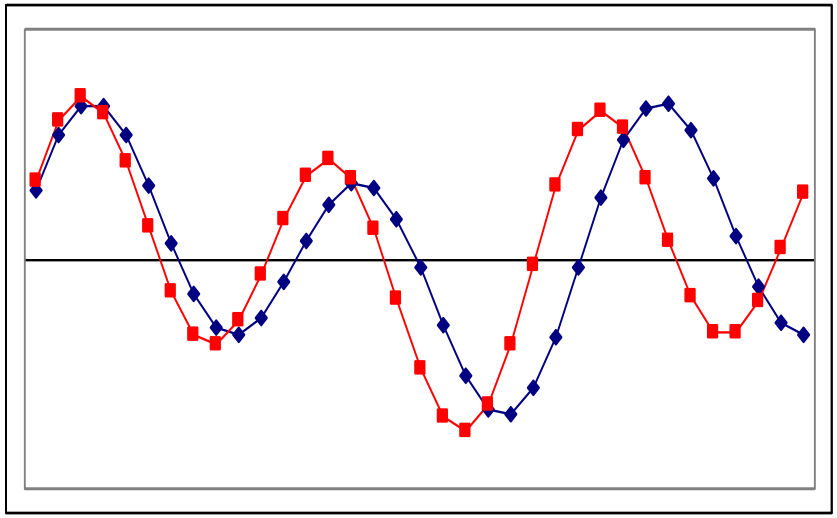


Time Warping

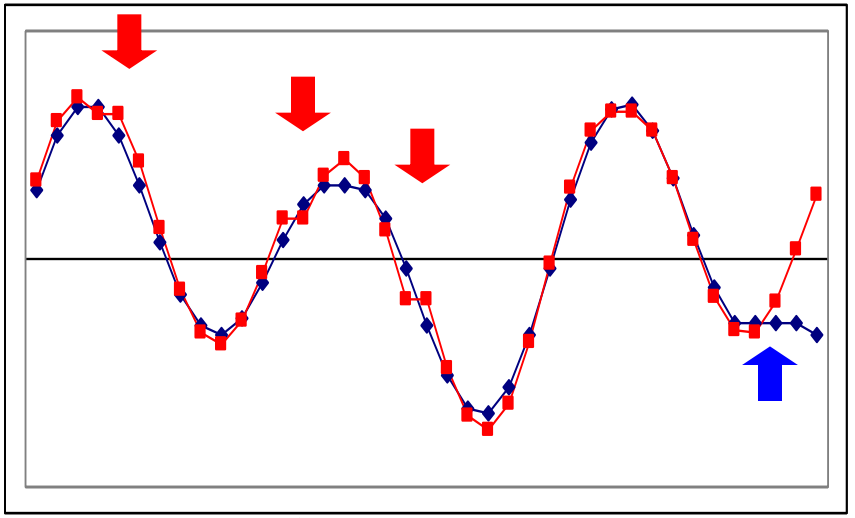
- allow accelerations - decelerations
  - (with or w/o penalty)
- THEN compute the (Euclidean) distance (+ penalty)
- related to the string-editing distance
- fast search methods [Yi+98] [Keogh+02] [Sakurai+05]

Time Warping

- Allow sequences to be stretched along the time axis
  1. minimize the distance of sequences
  2. insert 'stutters' into a sequence
  3. THEN compute the (Euclidean) distance



original

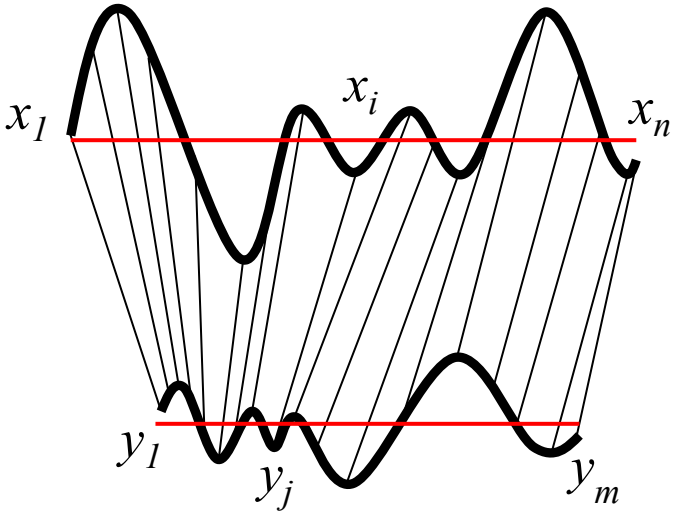


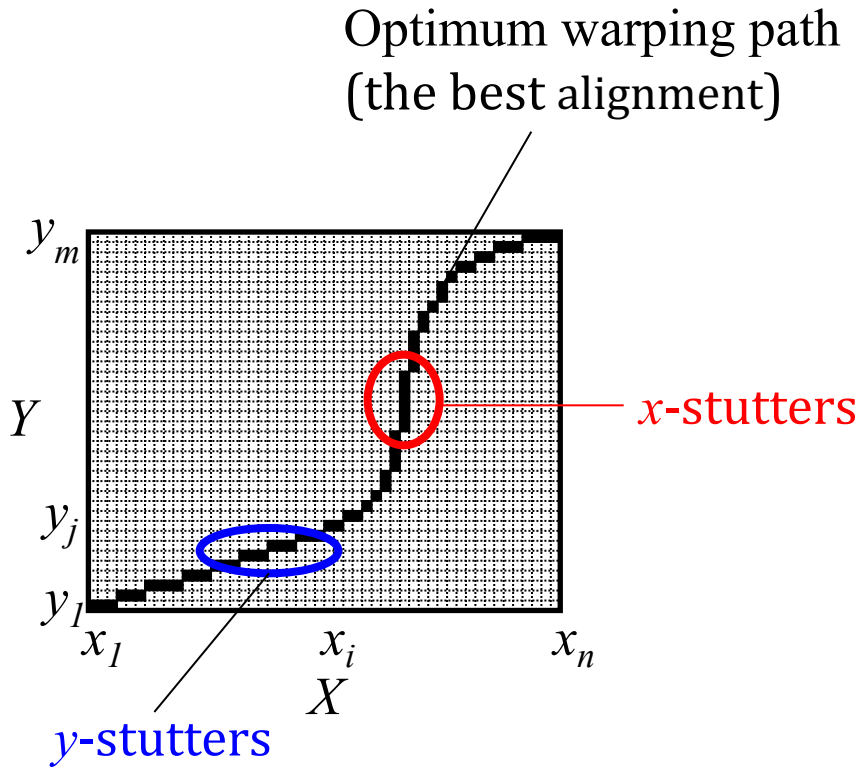
'stutters':

Time Warping

Q: how to compute it?

A: dynamic programming





Time Warping DETAILS

Q: how to compute it?

A: dynamic programming

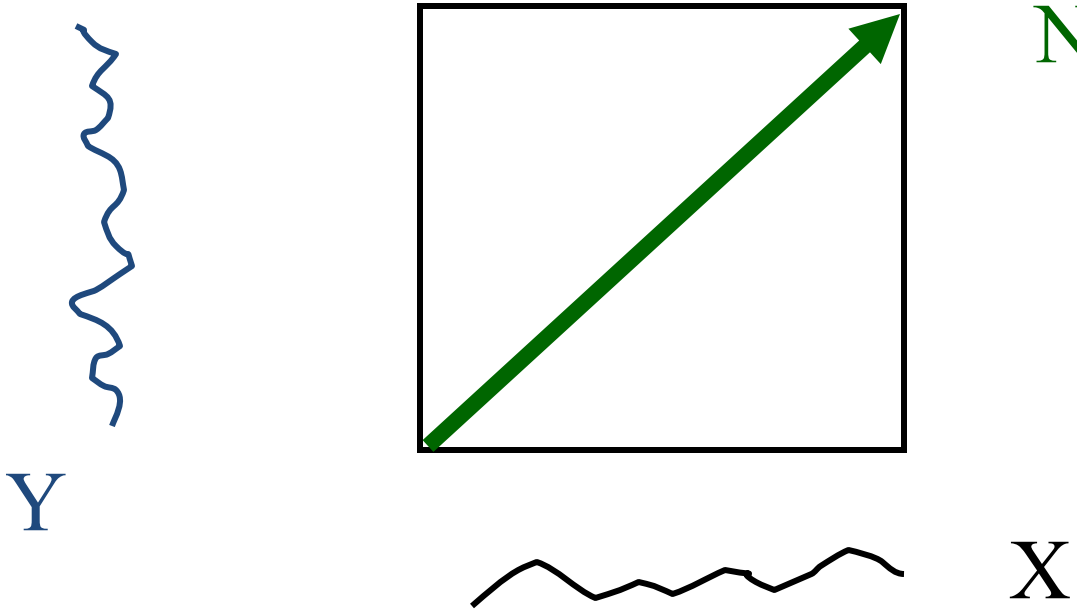
$$X=\{x_1, x_2, \ldots, x_i\},\ Y=\{y_1, y_2, \ldots, y_j\}$$

$$D_{dtw}(X,Y) = f(n,m)$$

$$f(i,j) = \left\|x_i - y_j\right\| + \min \begin{cases} f(i,j-1) & x\text{-stutter} \\ f(i-1,j) & y\text{-stutter} \\ f(i-1,j-1) & \text{no stutter} \end{cases}$$

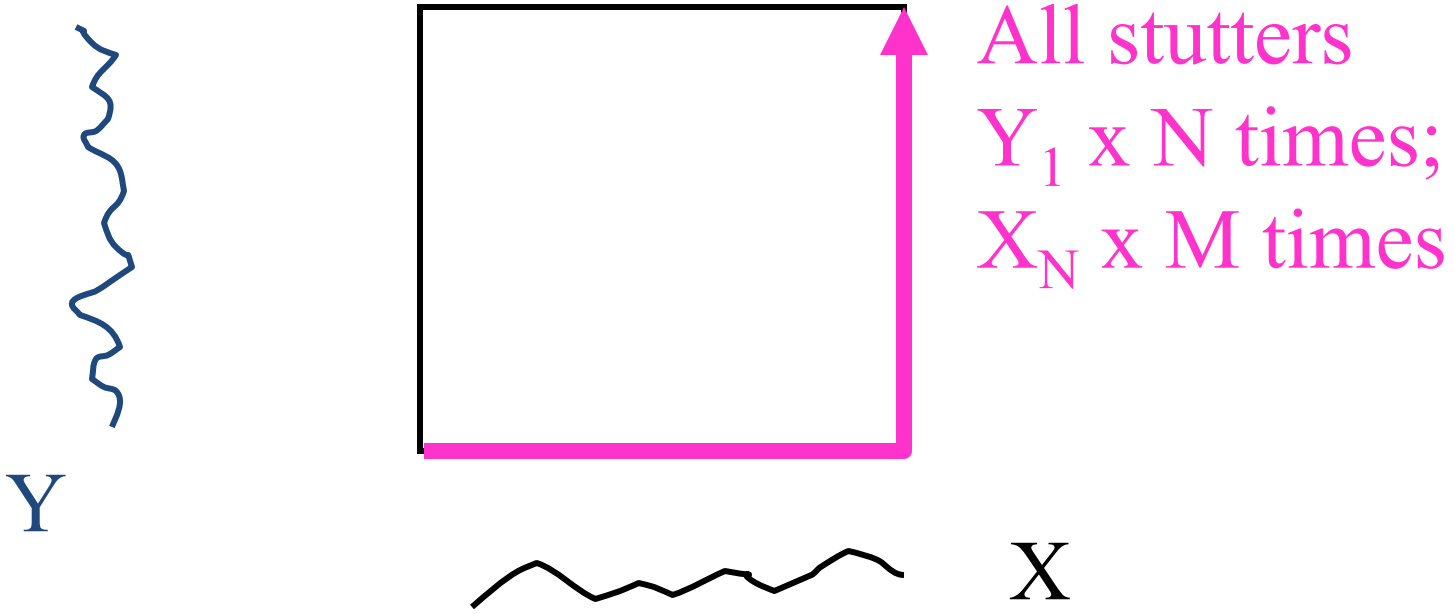
Time Warping

- Time warping matrix & optimal path:



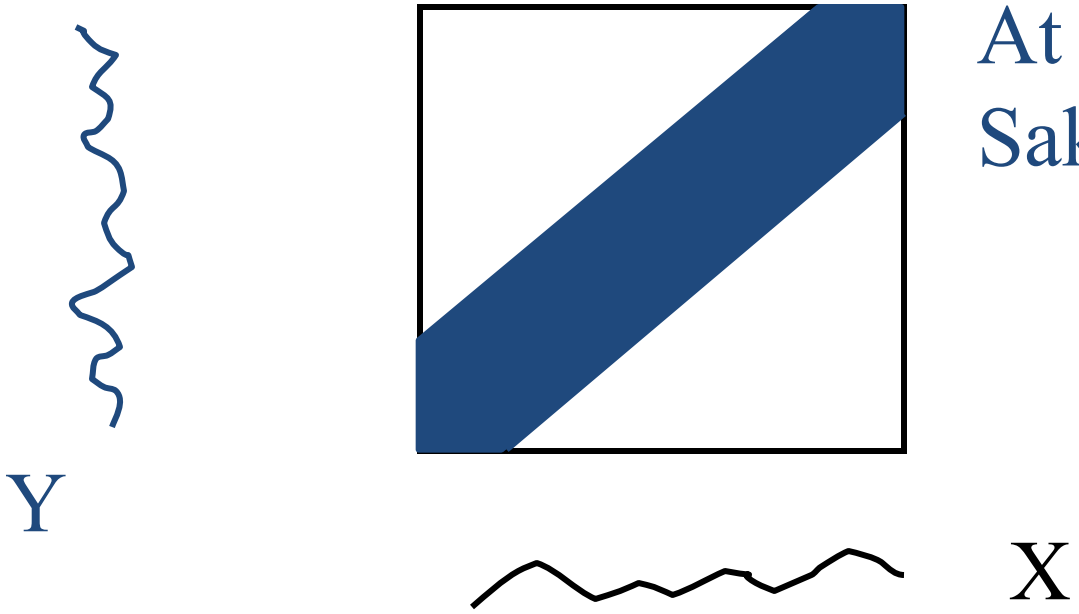
Time Warping

- Time warping matrix & optimal path:



Time Warping - variations

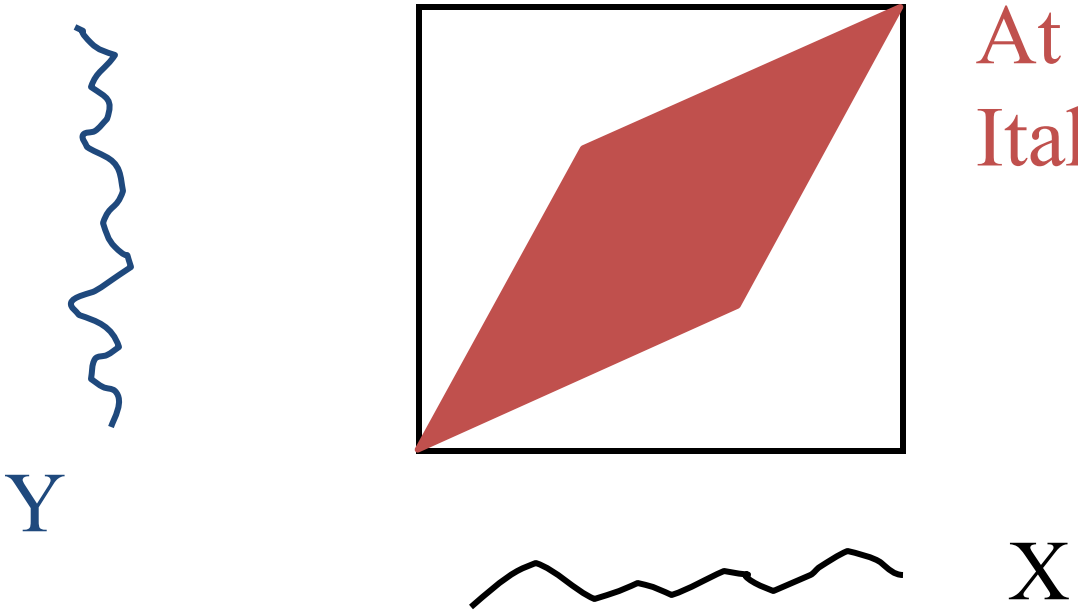
- Time warping matrix & optimal path:



At most k stutters:
Sakoe-Chiba band

Time Warping - variations

- Time warping matrix & optimal path:

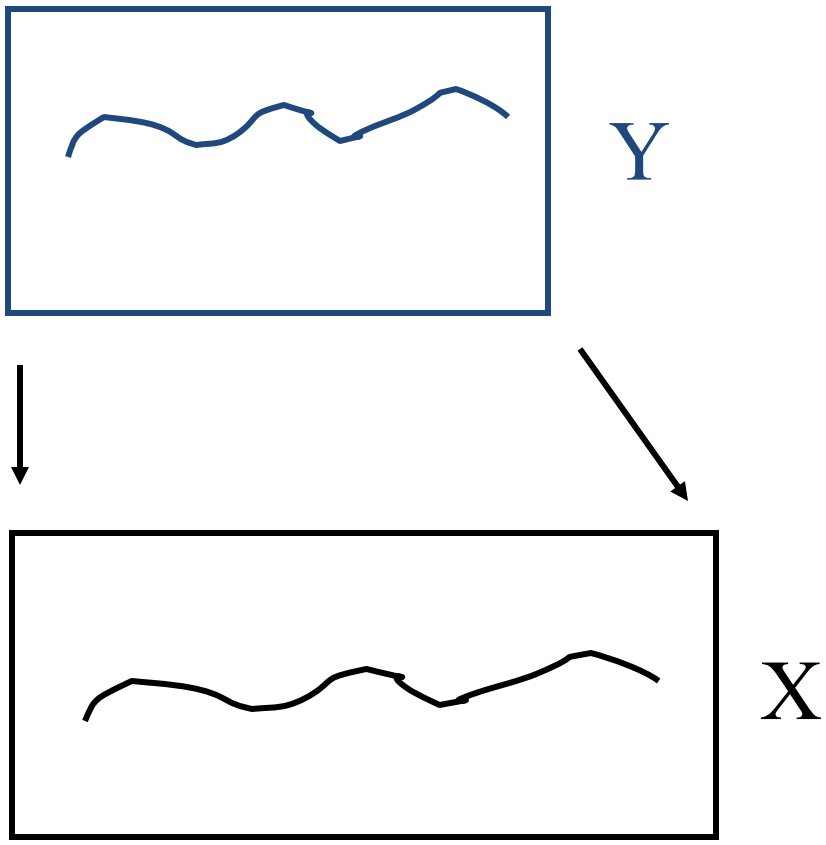


At most x% stutters:
Itakura parallelogram

Time warping

- Complexity: O(M*N) - quadratic on the length of the strings
- **Many** variations (penalty for stutters; limit on the number/percentage of stutters; …)
- popular in voice processing [Rabiner+Juang]

A variation: Uniform axis scaling



- Stretch / shrink time axis of Y, up to p%, for free
- THEN compute Euclidean distance
- [Keogh+, VLDB04]

Other Distance functions

- piece-wise linear/flat approx.; compare pieces [Keogh+01] [Faloutsos+97]
- 'cepstrum' (for voice [Rabiner+Juang])
  - do DFT; take log of amplitude; do DFT again!
- Allow for small gaps [Agrawal+95]

Related work

- Chen + Ng [vldb'04]: ERP 'Edit distance with Real Penalty': give a penalty to stutters
- Keogh+ [kdd'04]: VERY NICE, based on information theory: compress each sequence (quantize + Lempel-Ziv), using the **other** sequences' LZ tables
- Rakthanmanon+ [kdd' 12]: EXCELLENT Software, the UCR Suite for ultrafast subsequence search

Conclusions

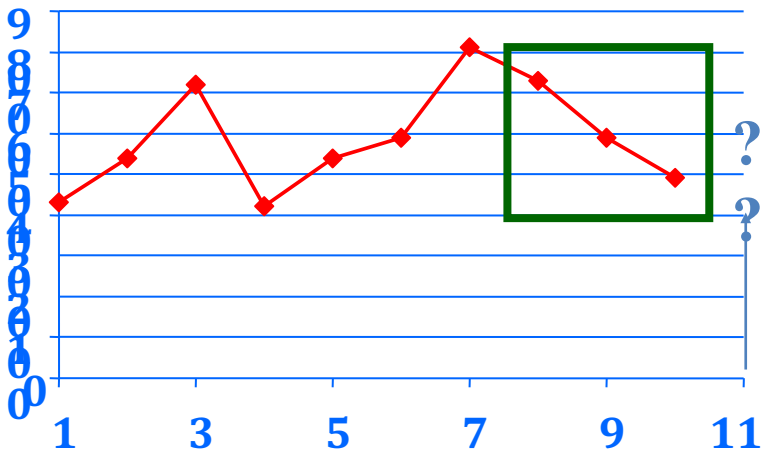
- Prevailing distances:
  - Euclidean and
  - time-warping

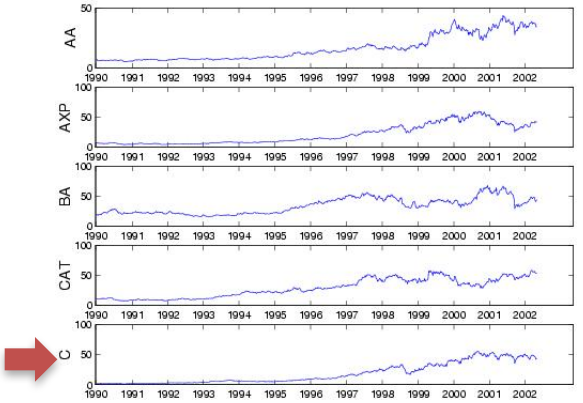
Roadmap

- Motivation
- Similarity Search and Indexing
  - distance functions: Euclidean, time-warping
  - indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

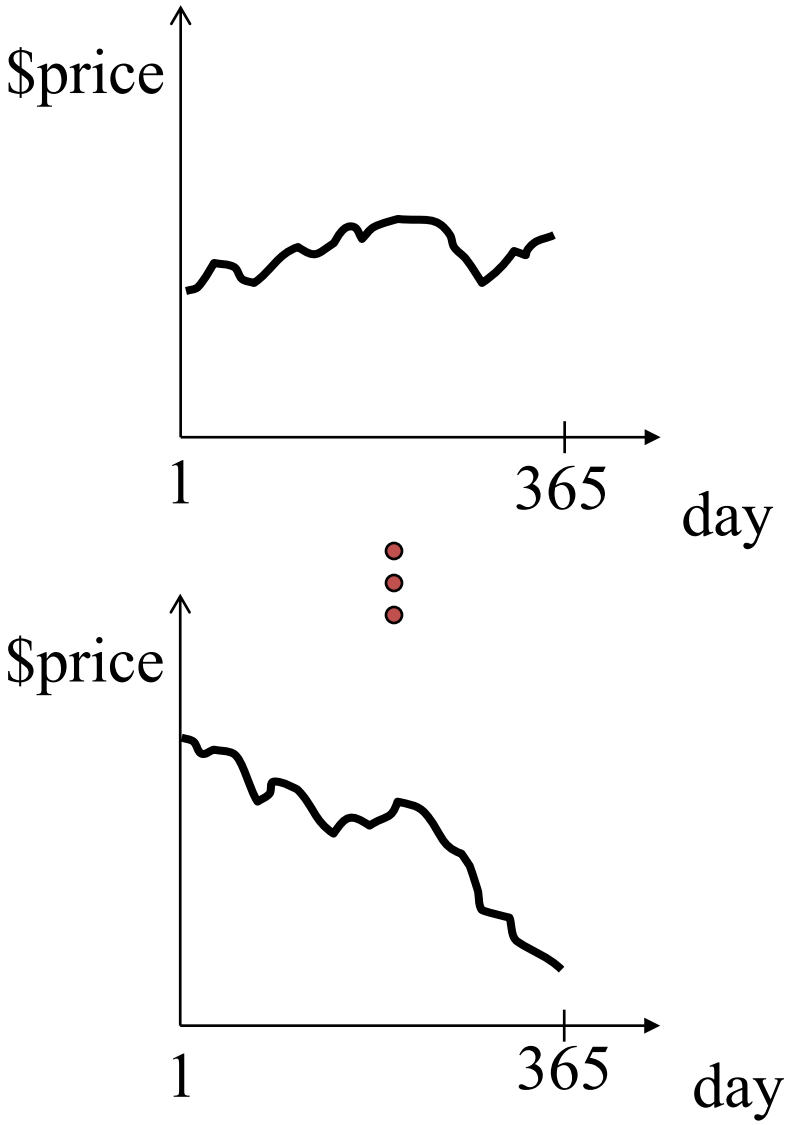
Indexing

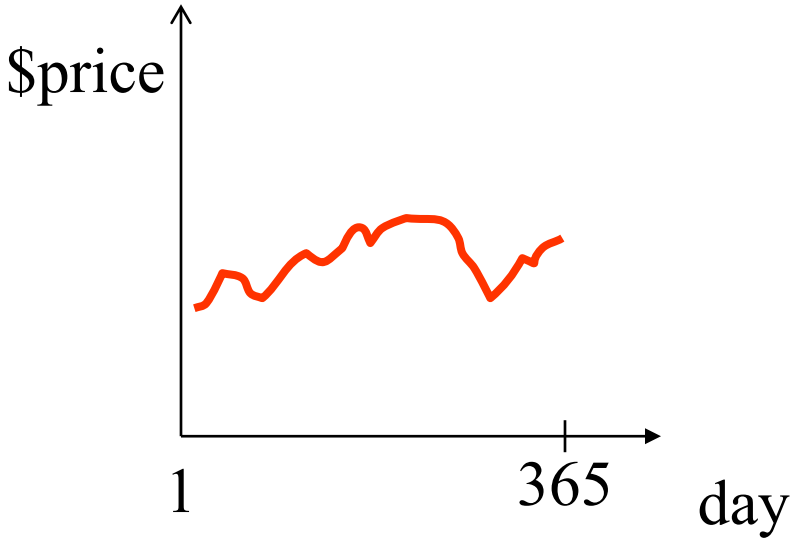
- Given a set of time sequences,
- Find the ones similar to a desirable query sequence





Indexing





distance function: by expert

(Euclidean; DTW; …)

Idea: 'GEMINI'

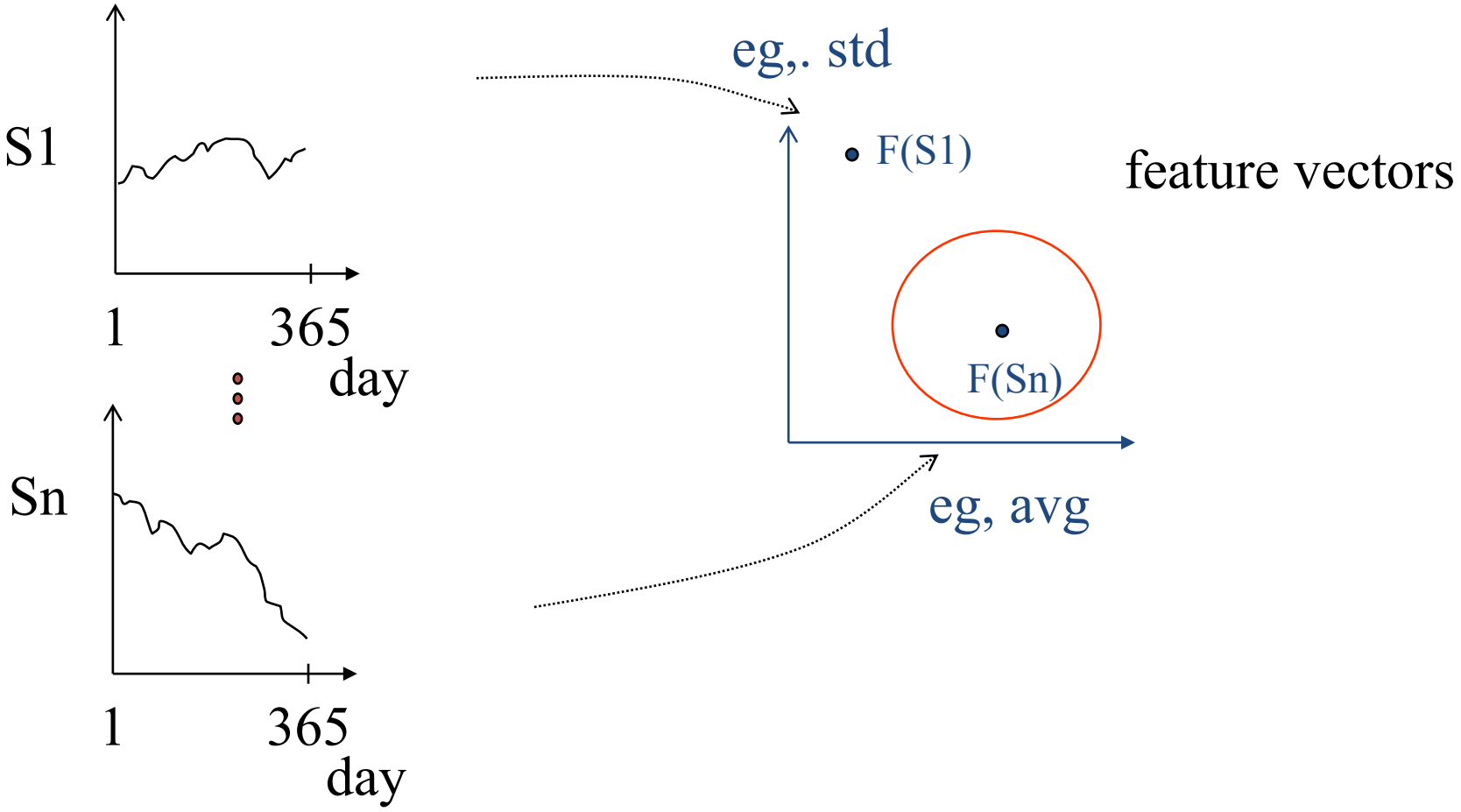
Eg., '*find stocks similar to MSFT*'

Seq. scanning: too slow

How to accelerate the search?

[Faloutsos96]

'GEMINI' - Pictorially



GEMINI

Solution: Quick-and-dirty' filter:

- extract *d* features (numbers, eg., avg., etc.)
- map into a point in the *d*-dimensional feature space
- organize points with off-the-shelf spatial access method ('SAM' – R-tree, etc)
- discard false alarms

Examples of GEMINI

- Time sequences: DFT (up to 100 times faster) [SIGMOD94];
- [Kanellakis+], [Mendelzon+]

Indexing - SAMs

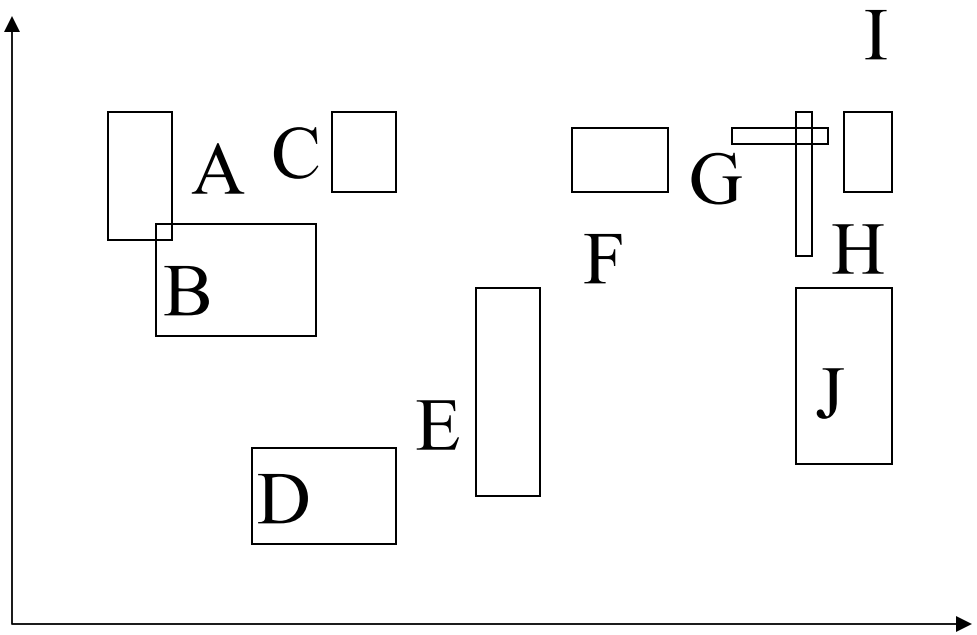
Q: How do Spatial Access Methods (SAMs) work?

A: they group nearby points (or regions) together, on nearby disk pages, and answer spatial queries quickly (‘range queries’, ‘nearest neighbor’ queries etc)

For example:

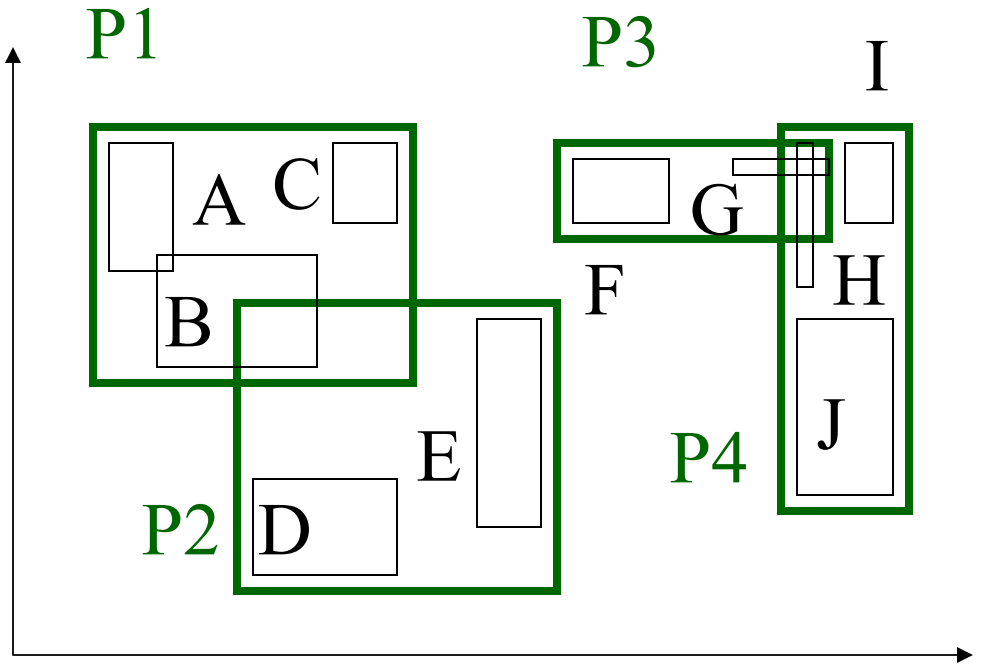
R-trees

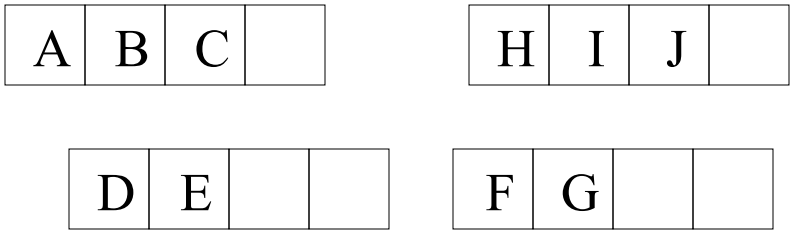
- [Guttman84] eg., w/ fanout 4: group nearby rectangles to parent MBRs; each group -> disk page



R-trees

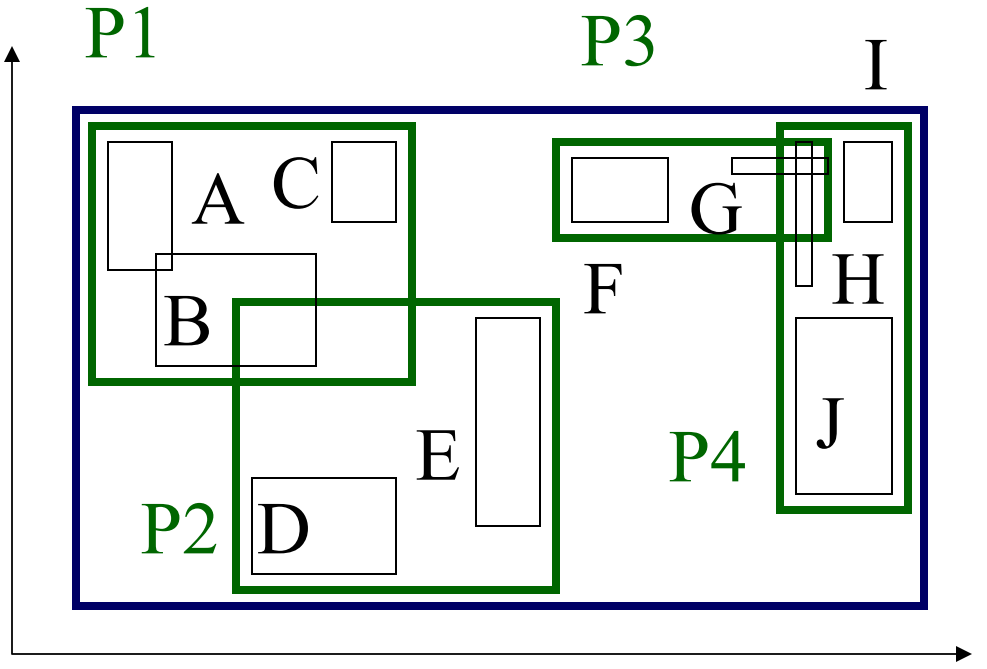
- eg., w/ fanout 4:

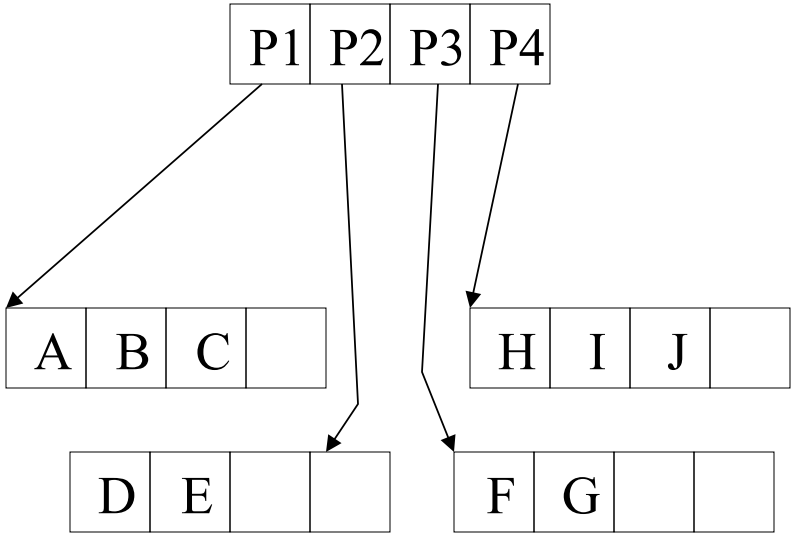




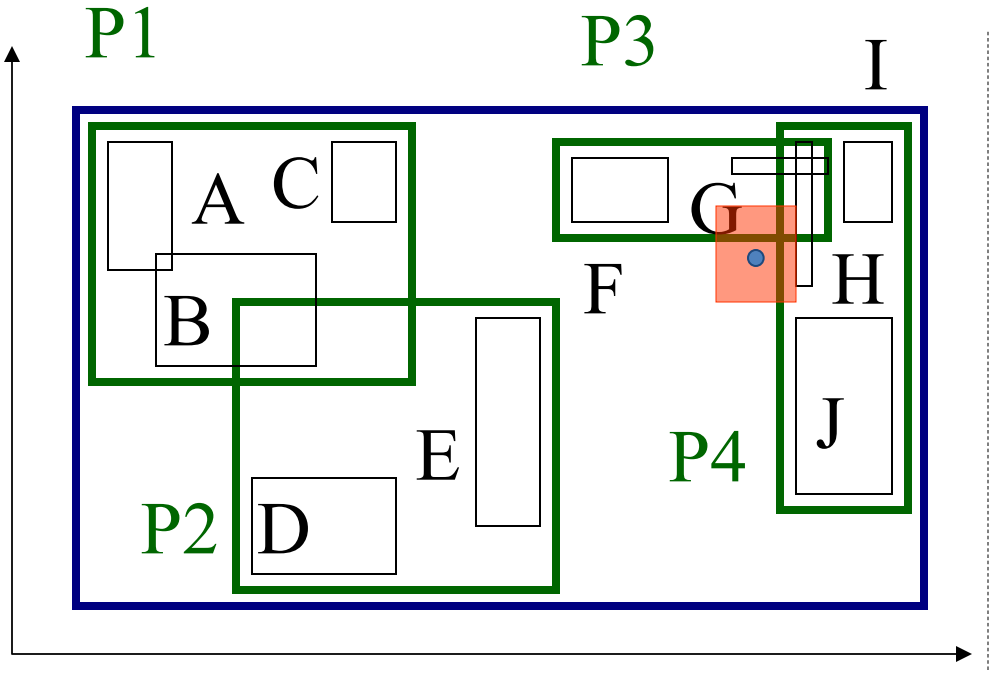
R-trees

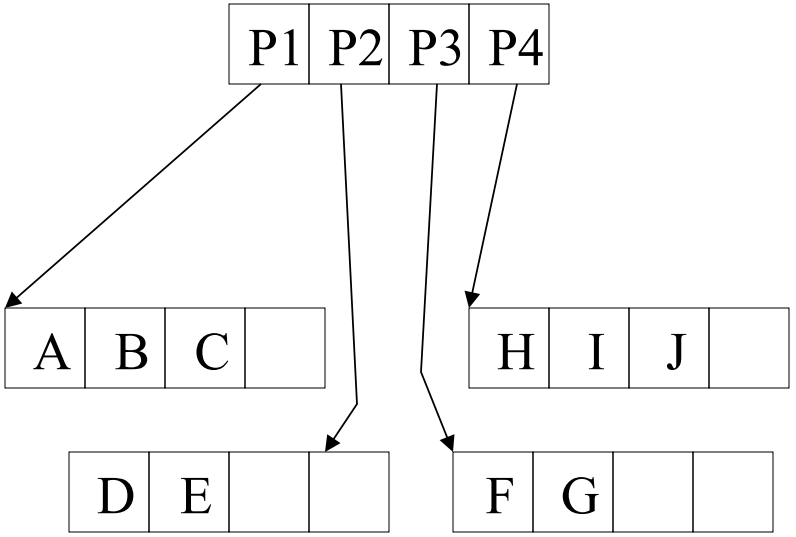
- eg., w/ fanout 4:



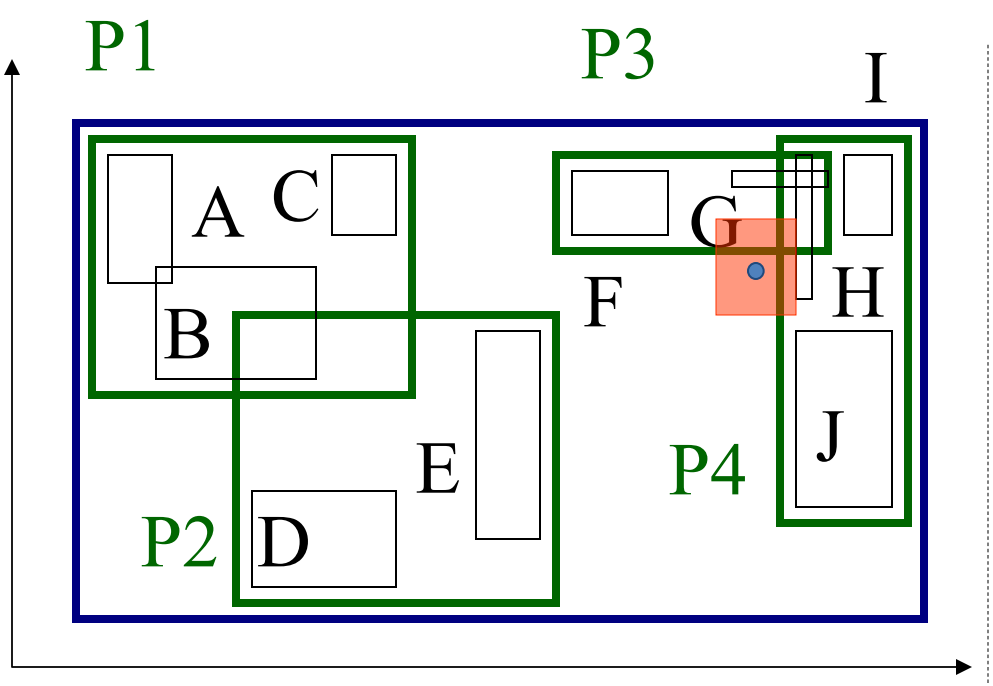


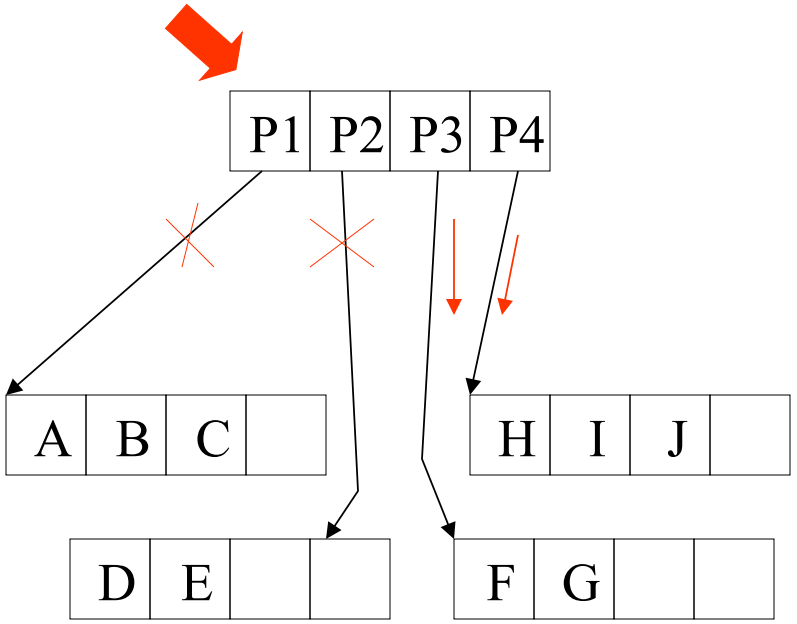
R-trees - range search?





R-trees - range search?





Conclusions

- Fast indexing: through GEMINI
  - feature extraction and
  - (off the shelf) Spatial Access Methods [Gaede+98]

Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
  - DFT, DWT (data independent)
  - SVD, ICA (data independent)
  - MDS, FastMap
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

DFT: definition

- For a sequence $x_0, x_1, \ldots x_{n-1}$
- the (**n-point**) Discrete Fourier Transform is
- $X_0, X_1, \ldots X_{n-1}$:

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j2\pi\, tf / n) \qquad f = 0, \ldots, n-1$$

$$(j = \sqrt{-1}\,)$$

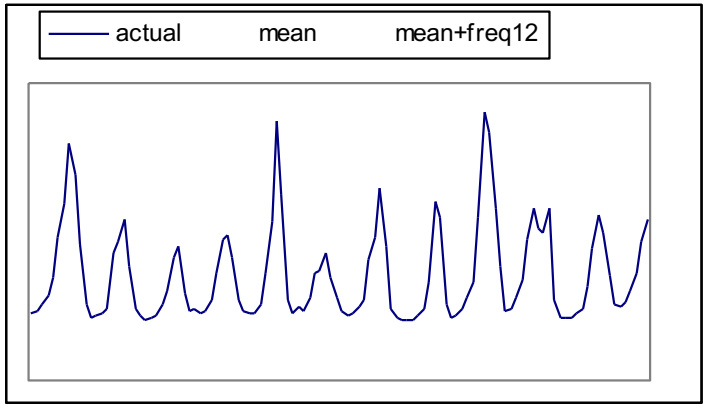
inverse DFT

$$x_t = 1/\sqrt{n} \sum_{t=0}^{n-1} X_f * \exp(+j2\pi\, tf / n)$$

DFT: Amplitude spectrum

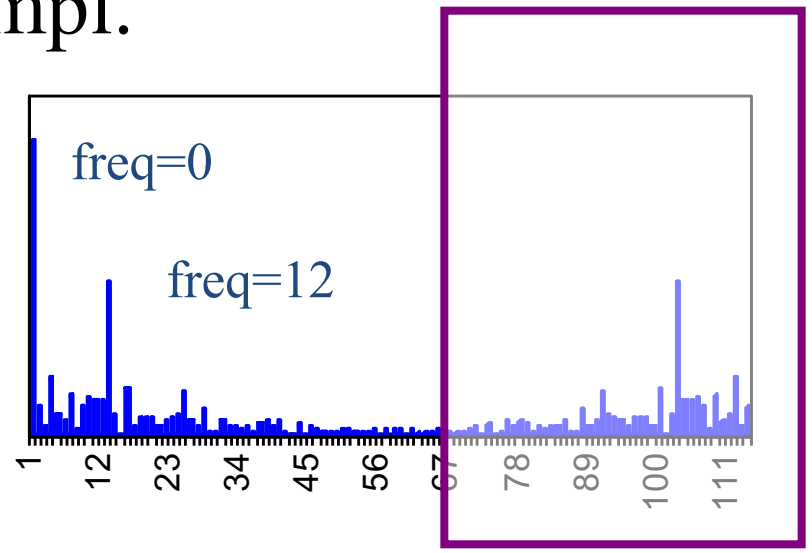
Amplitude: $A_f^2 = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



year

Ampl.

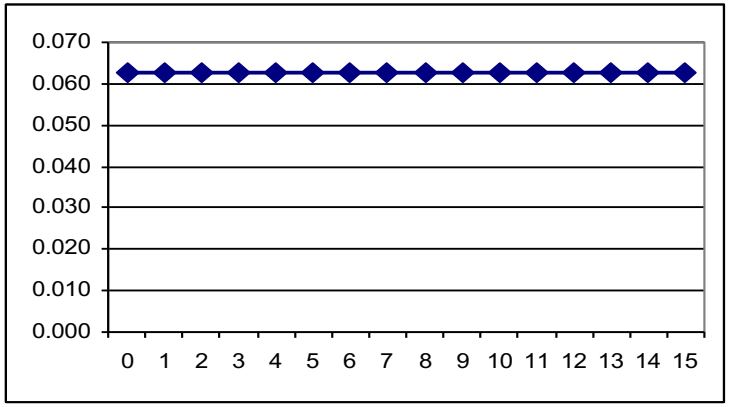


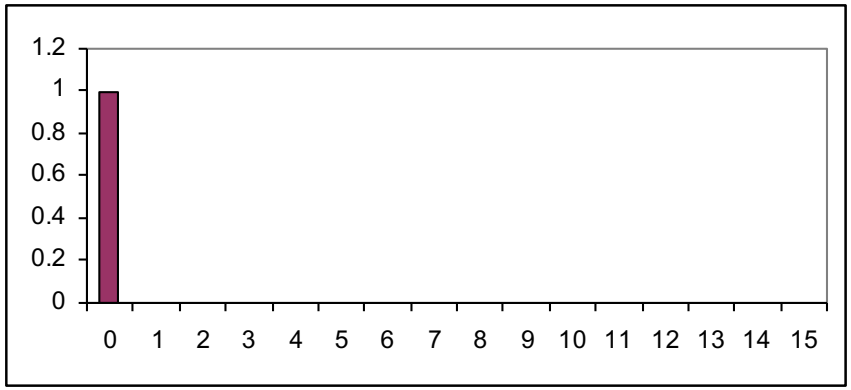
Freq.

DFT: examples

- Flat

Amplitude



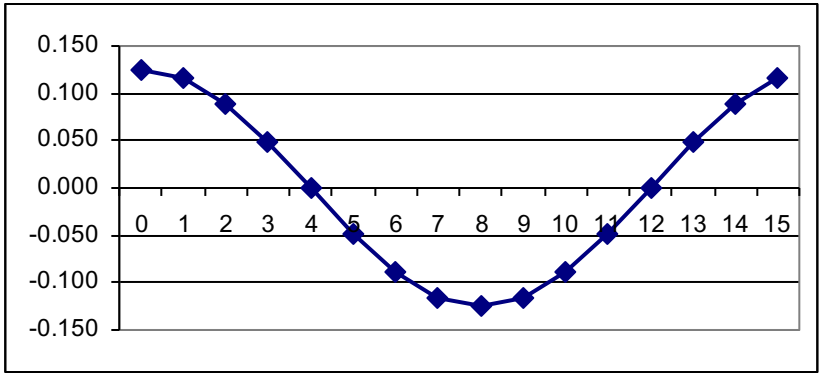


time

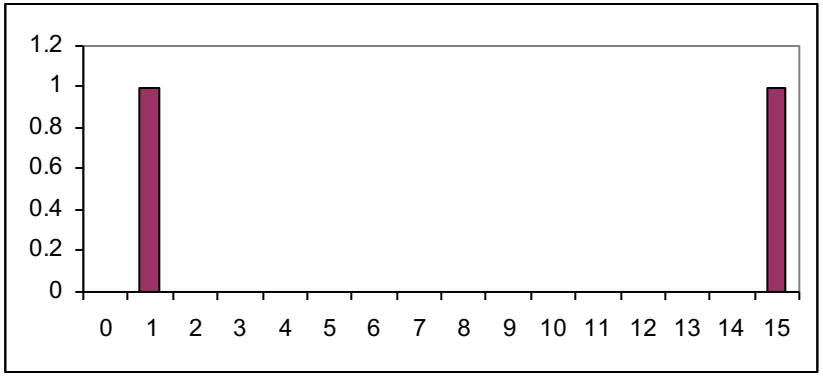
freq

DFT: examples

- Low frequency sinusoid



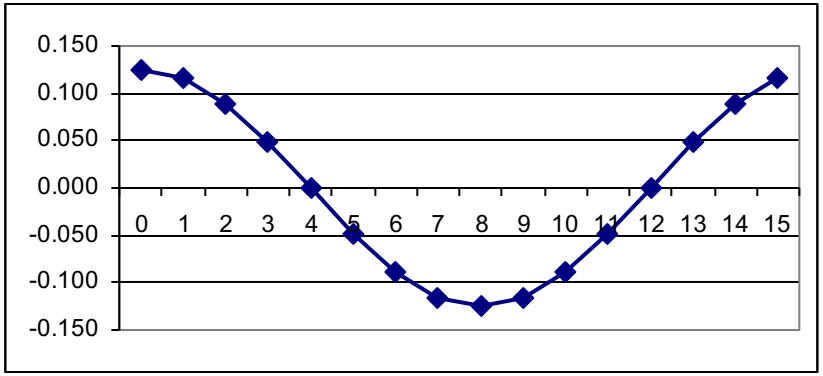
time



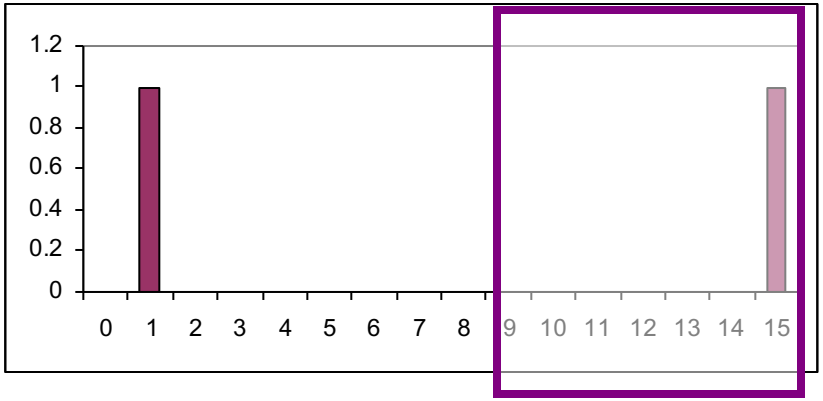
freq

DFT: examples

- Sinusoid - symmetry property: $X_f = X^*_{n-f}$



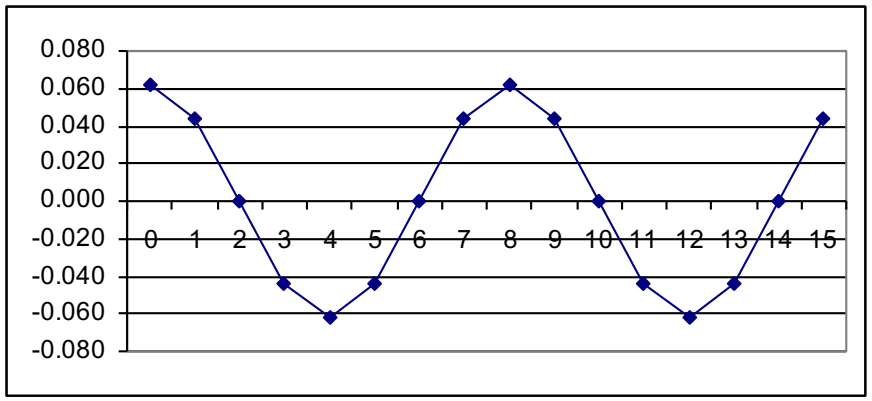
time



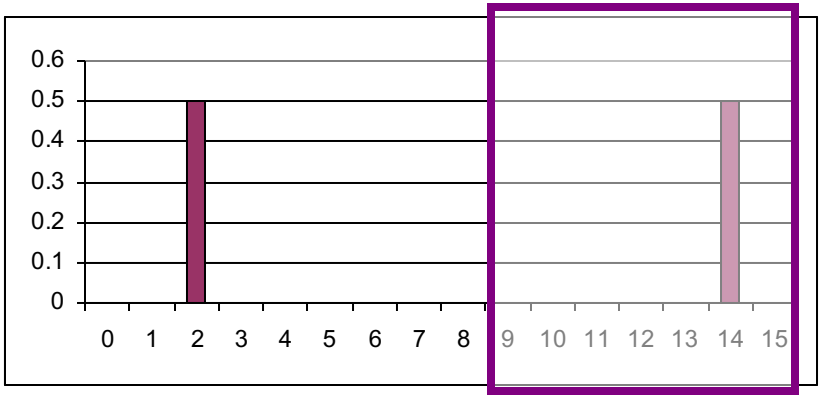
freq

DFT: examples

- Higher freq. sinusoid



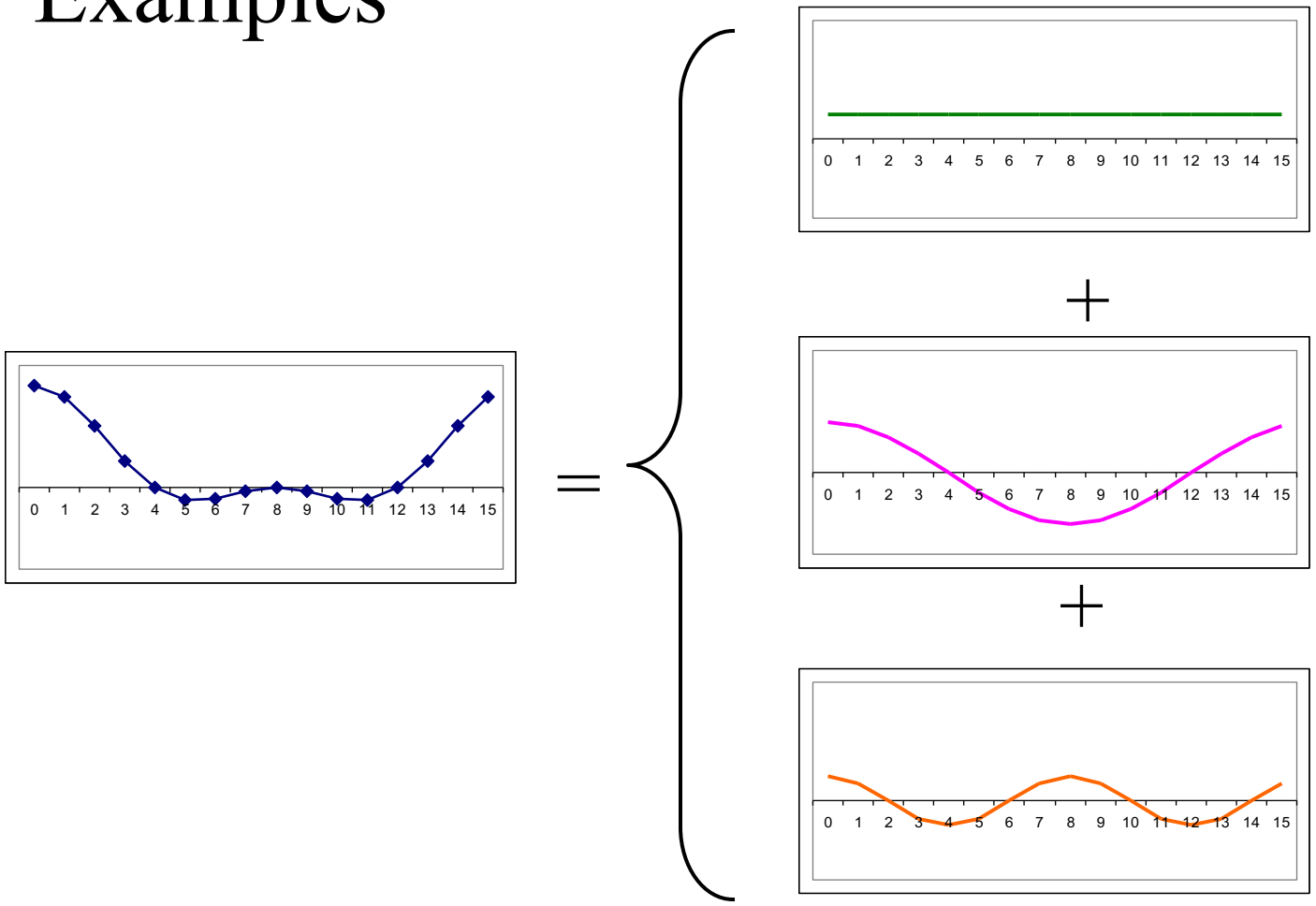
time



freq

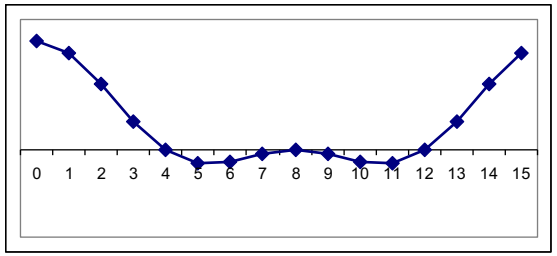
DFT: examples

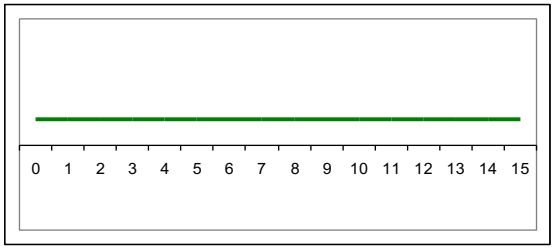
- Examples

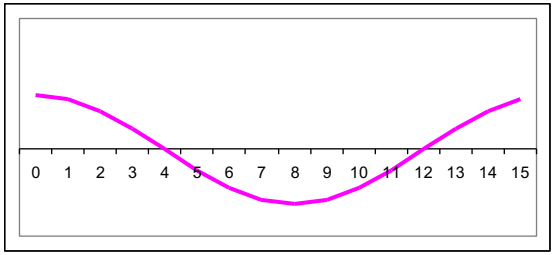


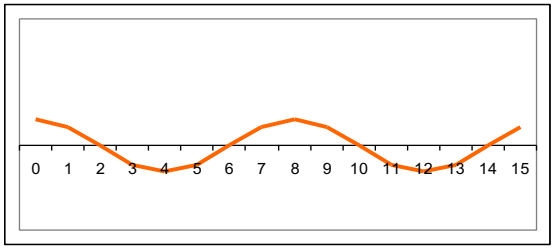
DFT: examples

- Examples

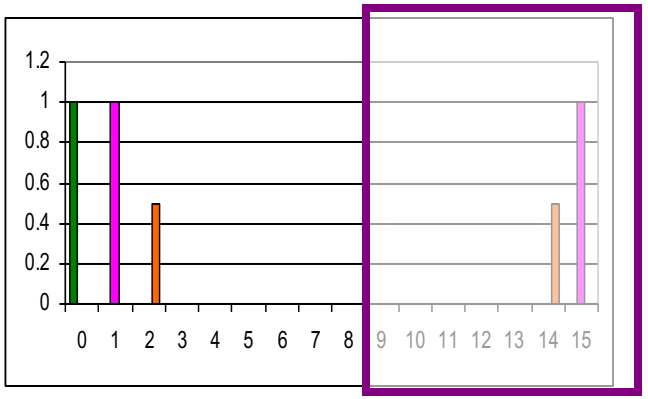








Ampl.

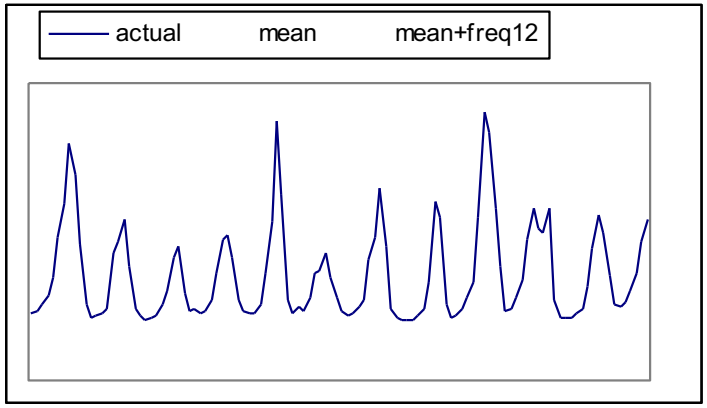


Freq.

DFT: Amplitude spectrum

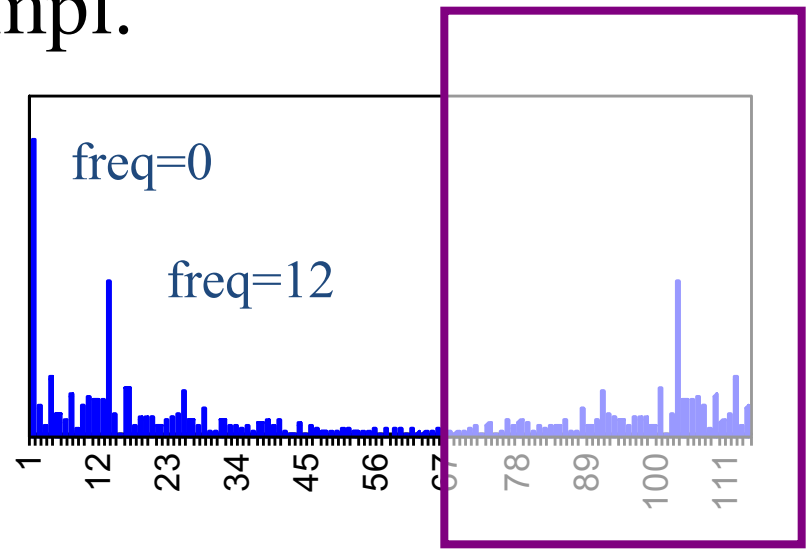
Amplitude: $A_f^2 = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



year

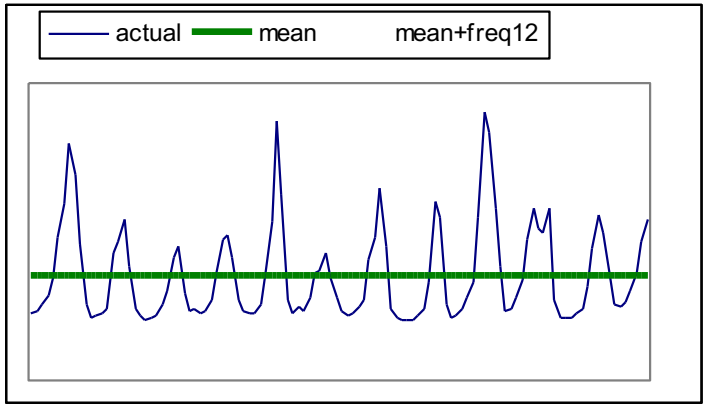
Ampl.



Freq.

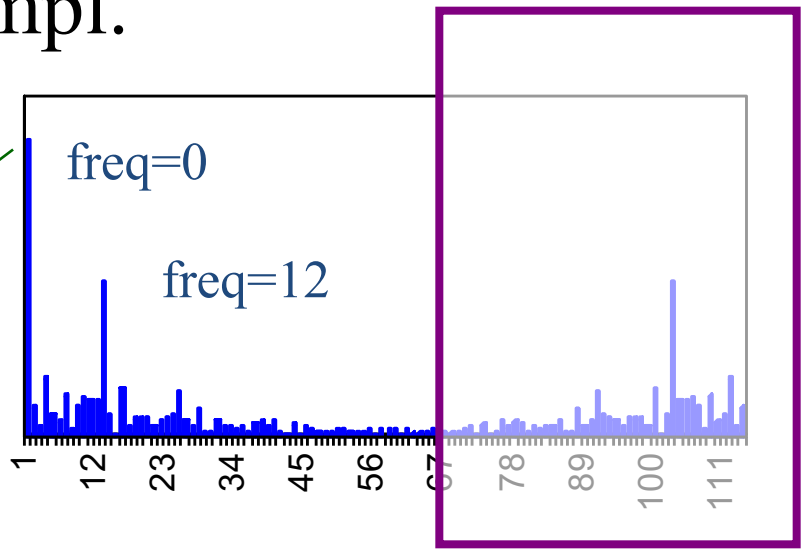
DFT: Amplitude spectrum

count



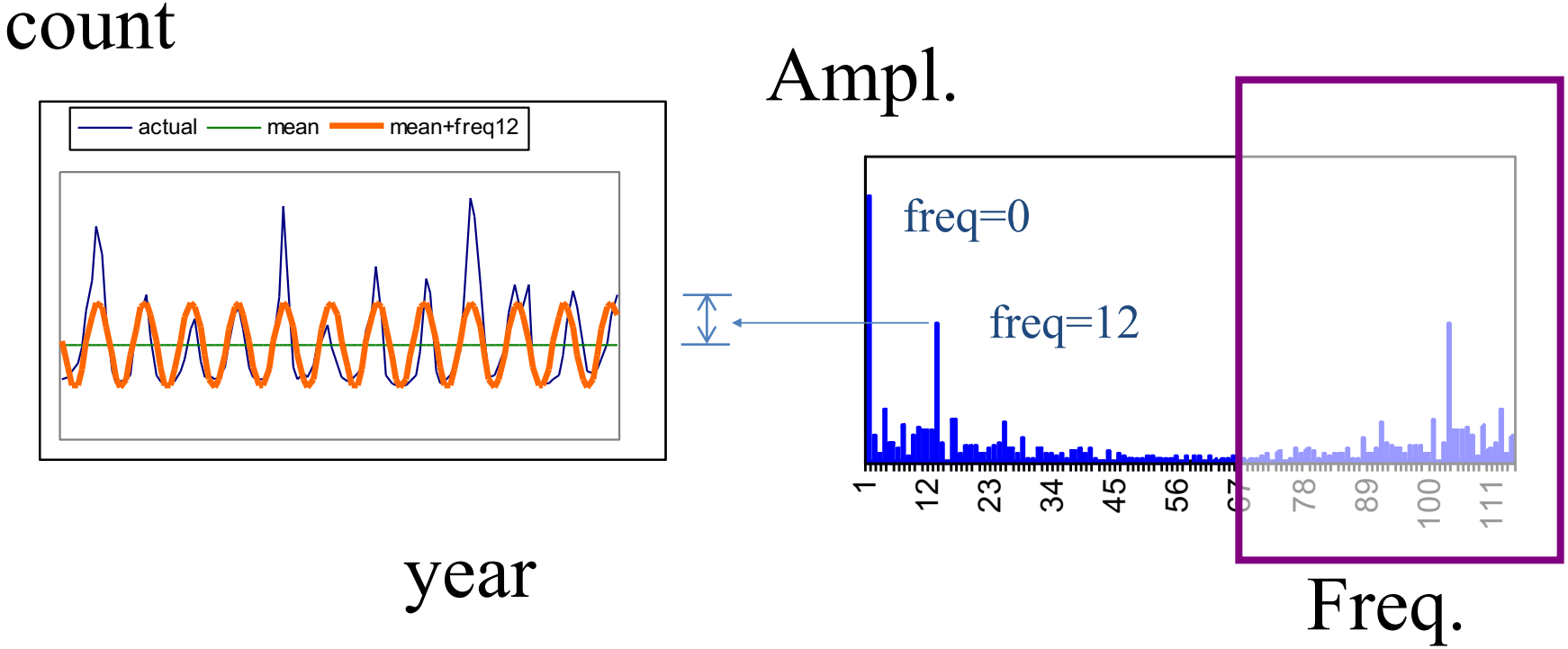
year

Ampl.



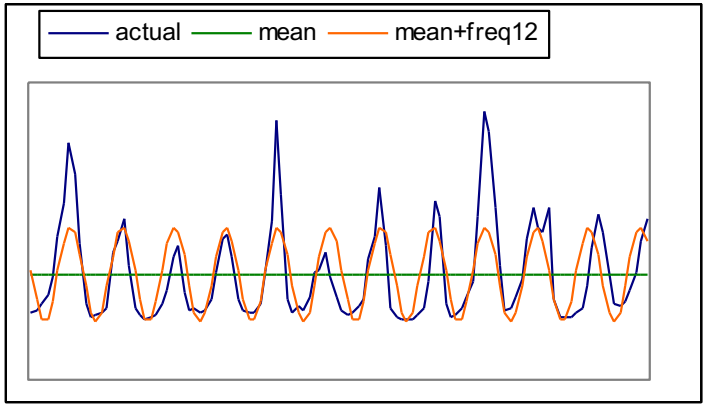
Freq.

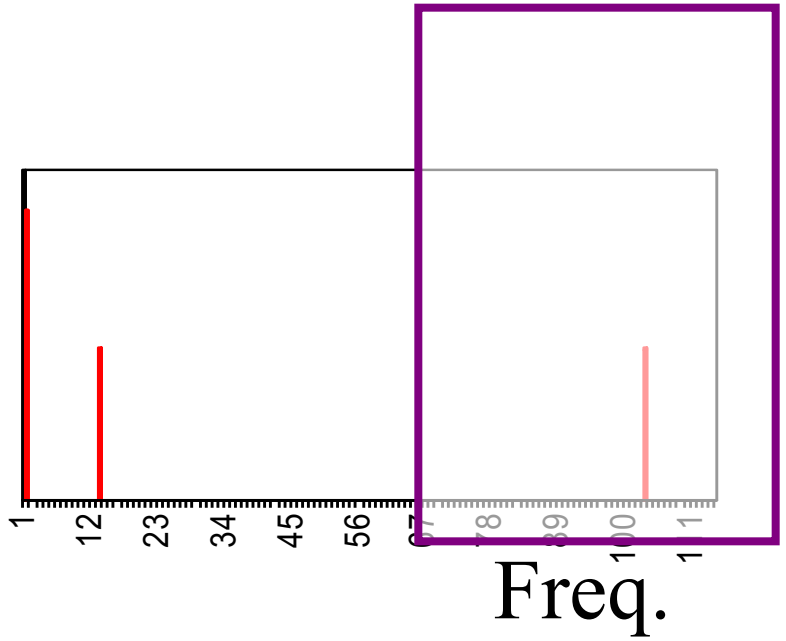
DFT: Amplitude spectrum



DFT: Amplitude spectrum

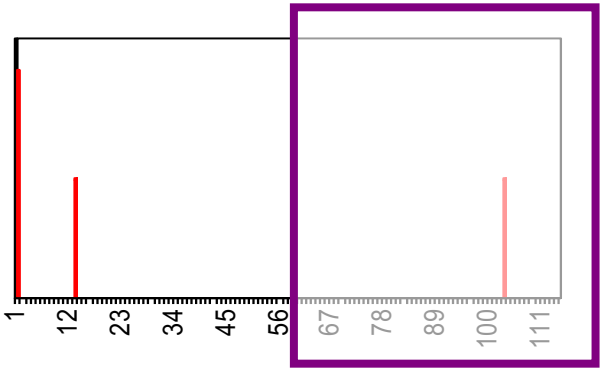
- Excellent approximation, with only 2 frequencies!
- So what?





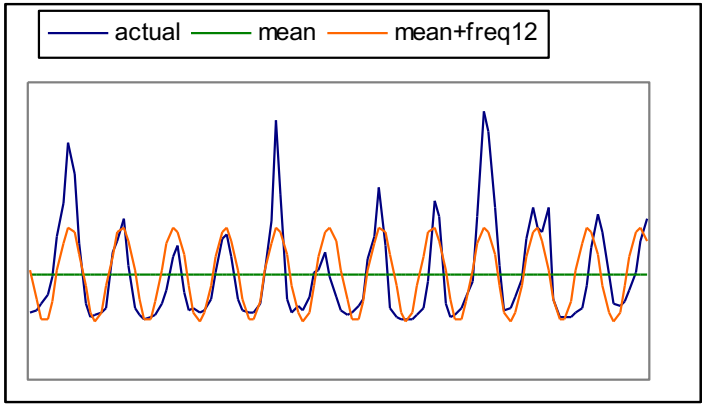
DFT: Amplitude spectrum

- Excellent approximation, with only 2 frequencies!
- So what?
- A1: **(lossy) compression**
- A2: pattern discovery



DFT: Amplitude spectrum

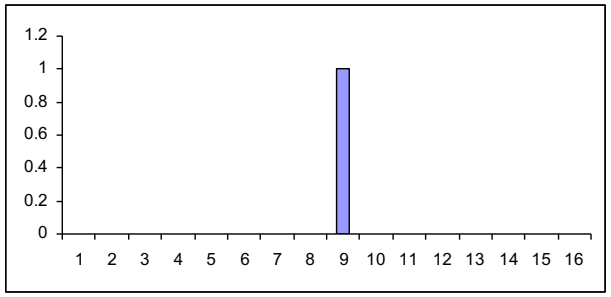
- Excellent approximation, with only 2 frequencies!
- So what?
- A1: (lossy) compression
- A2: **pattern discovery**



Wavelets - DWT

- DFT is great - but, how about compressing a spike?

value

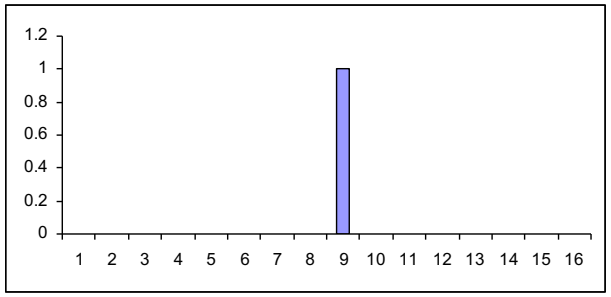


time

Wavelets - DWT

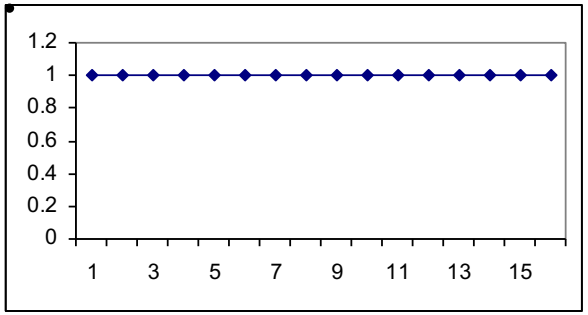
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



time

Ampl.

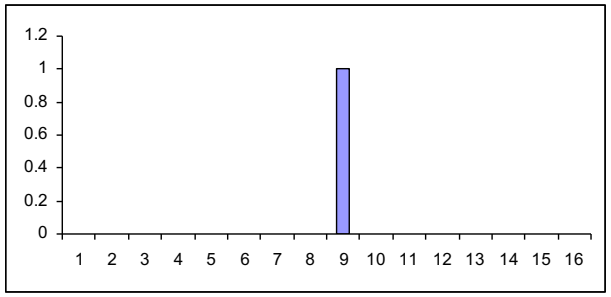


Freq.

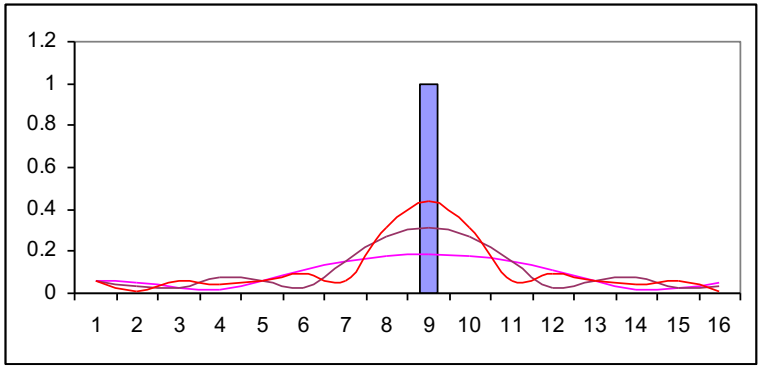
Wavelets - DWT

- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



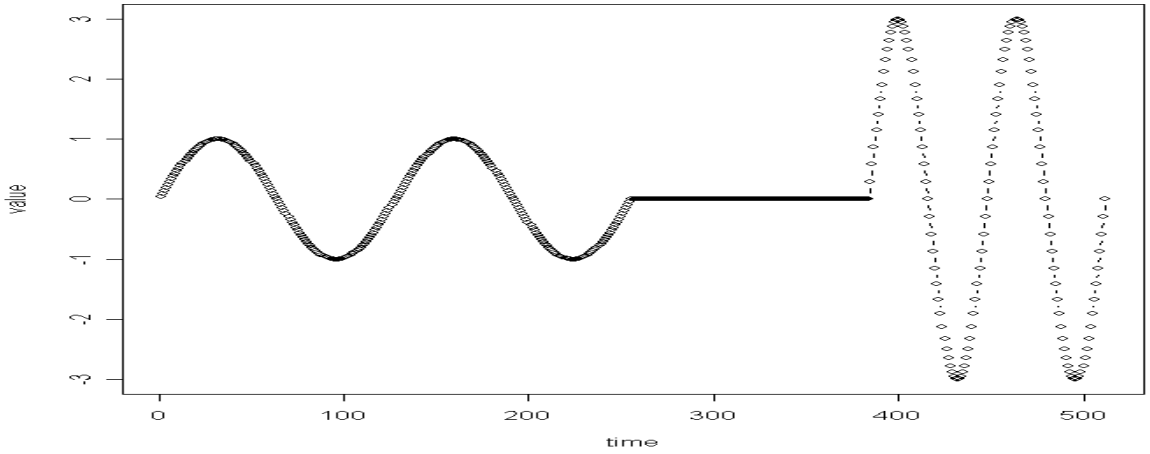
time



Wavelets - DWT

- Similarly, DFT suffers on short-duration waves (eg., baritone, silence, soprano)

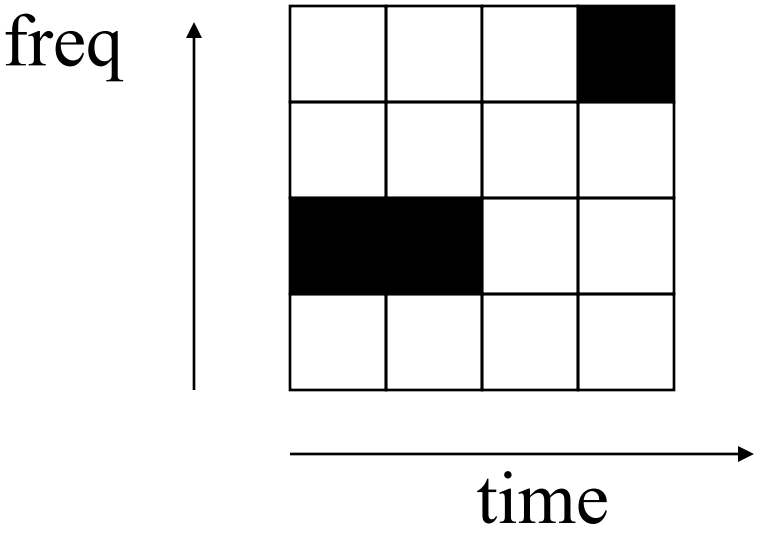
value

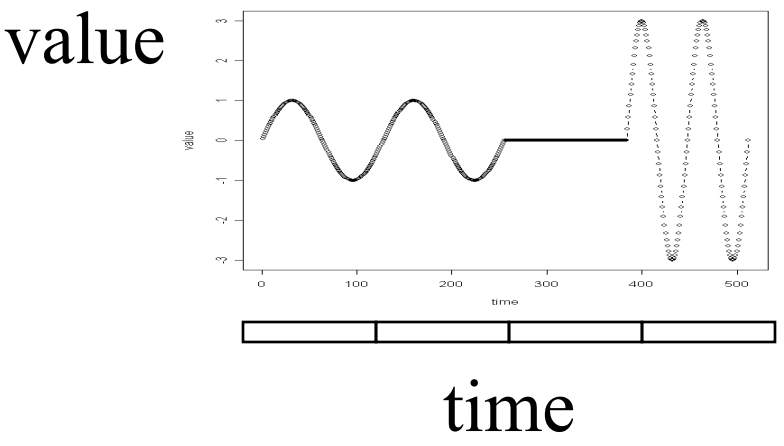


time

Wavelets - DWT

- Solution#1: Short window Fourier transform (SWFT)
- But: how short should be the window?





Wavelets - DWT

- Answer: **multiple** window sizes! -> DWT

  `Multi-scale windows': brilliant idea that we'll see several times in this tutorial (BRAID, TriMine, etc)

Wavelets - DWT

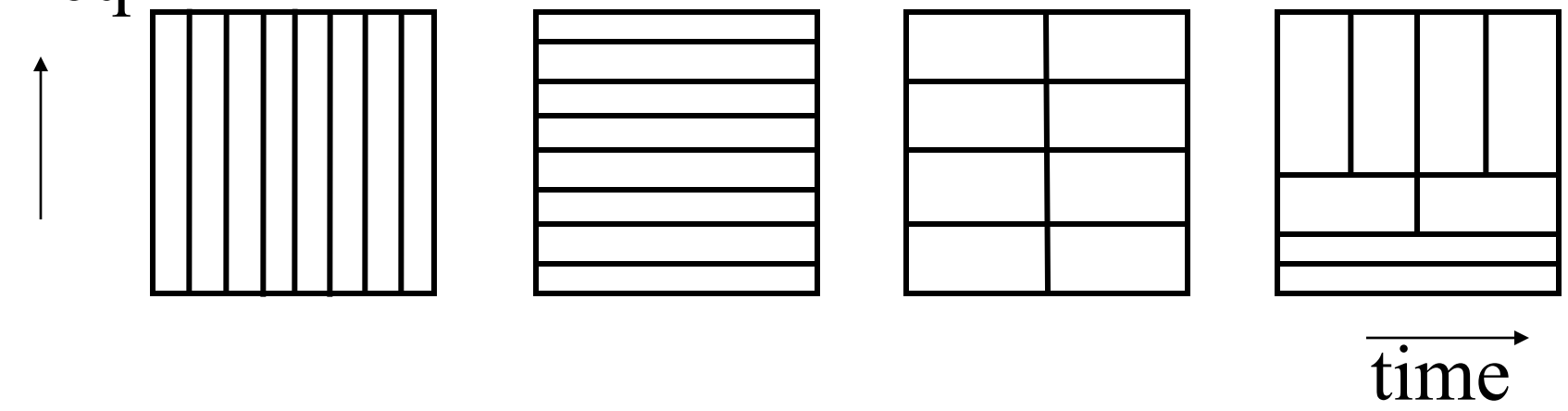
- Answer: **multiple** window sizes! -> DWT

Time domain

Multi-scale windows

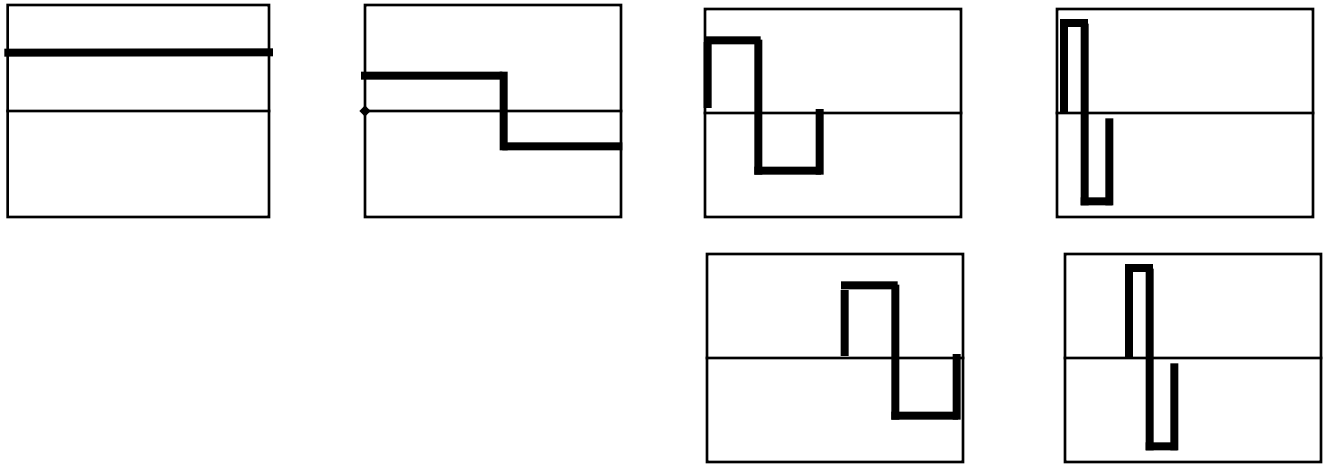
DFT SWFT DWT

freq



Haar Wavelets

- subtract sum of left half from right half
- repeat recursively for quarters, eight-ths, ...

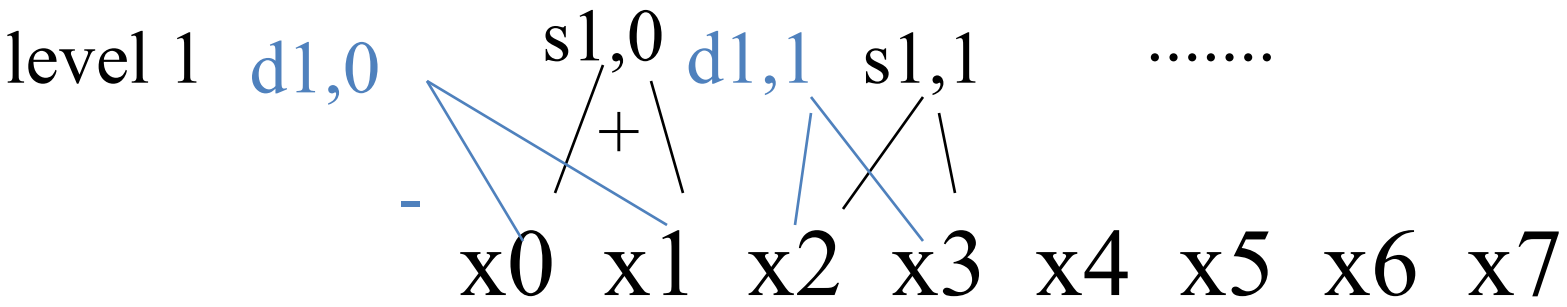


Wavelets - construc

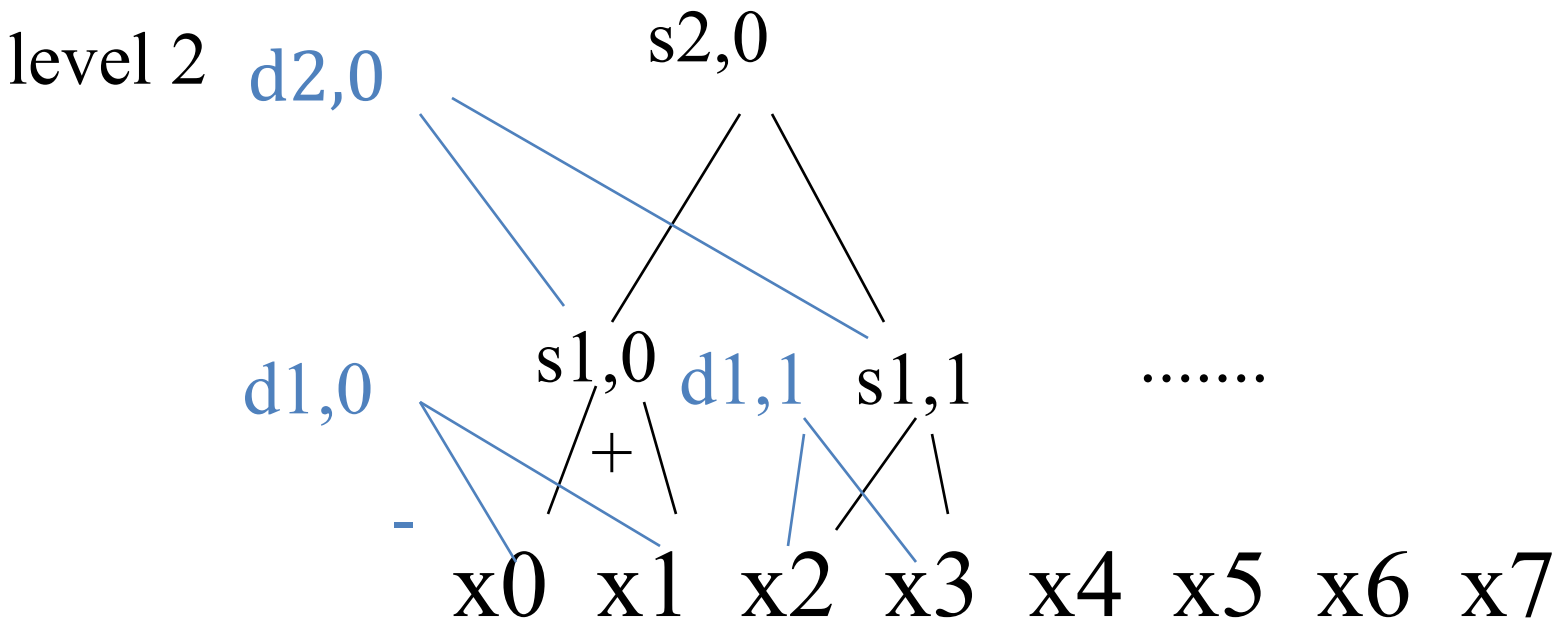
DETAILS

x0 x1 x2 x3 x4 x5 x6 x7

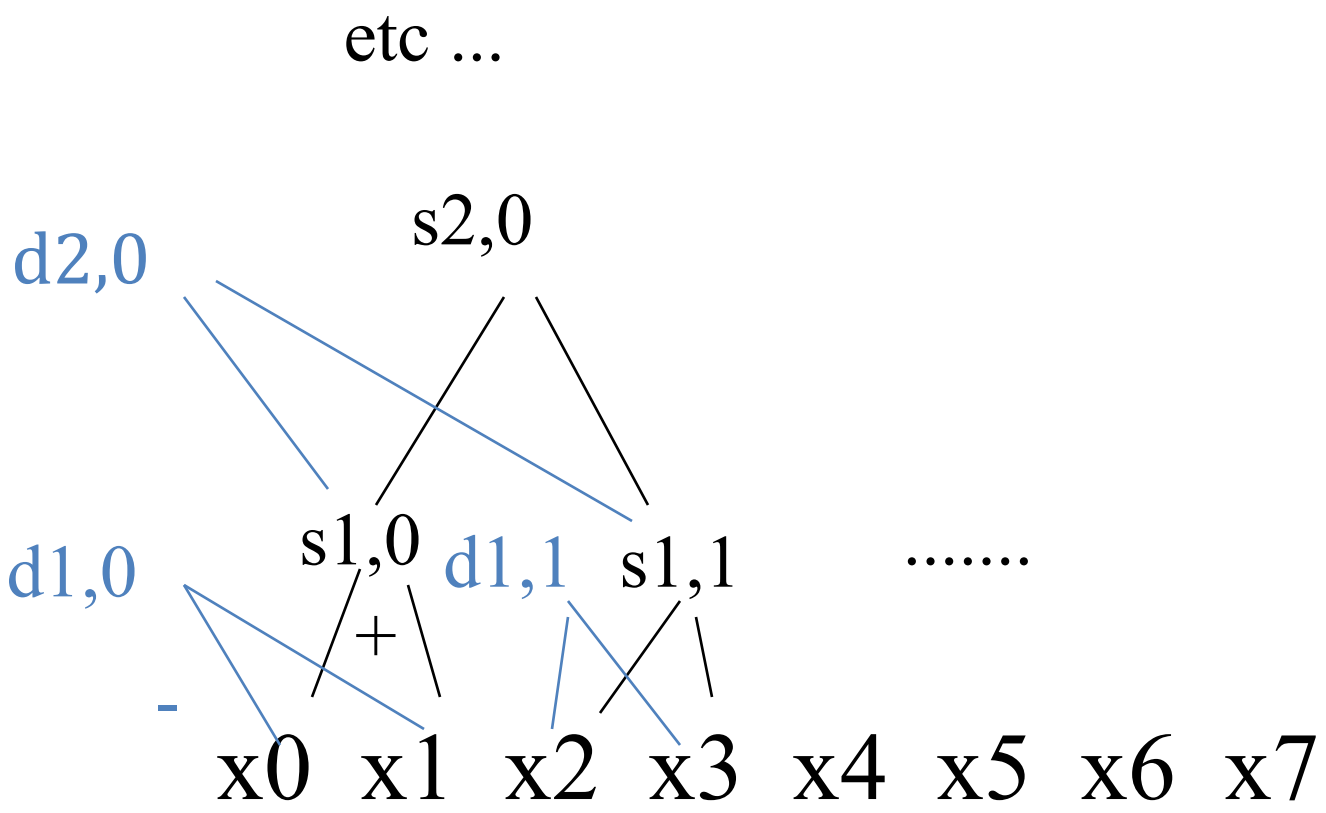
Wavelets - construc DETAILS



Wavelets - construc DETAILS



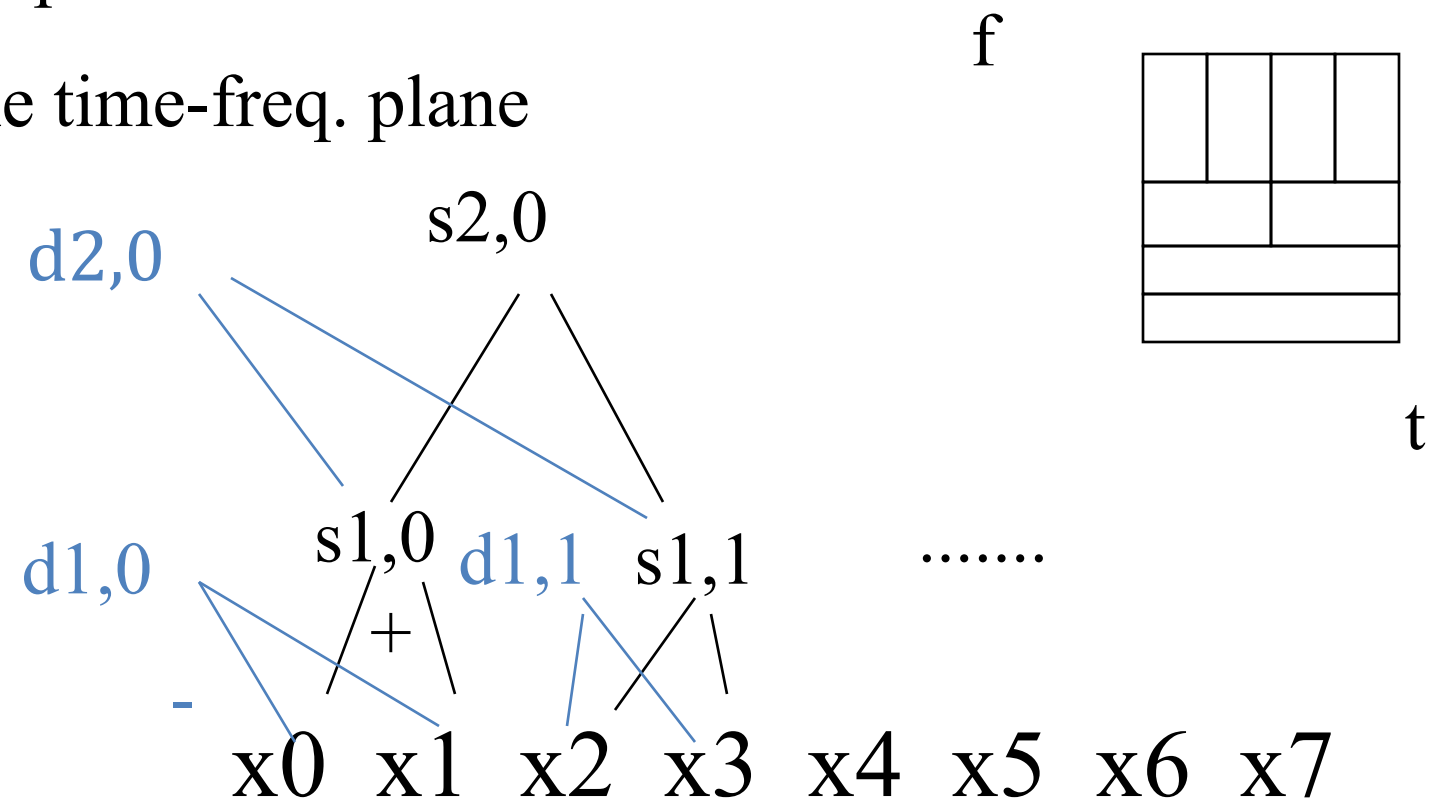
Wavelets - construc DETAILS



Wavelets - construc DETAILS

Q: map each coefficient

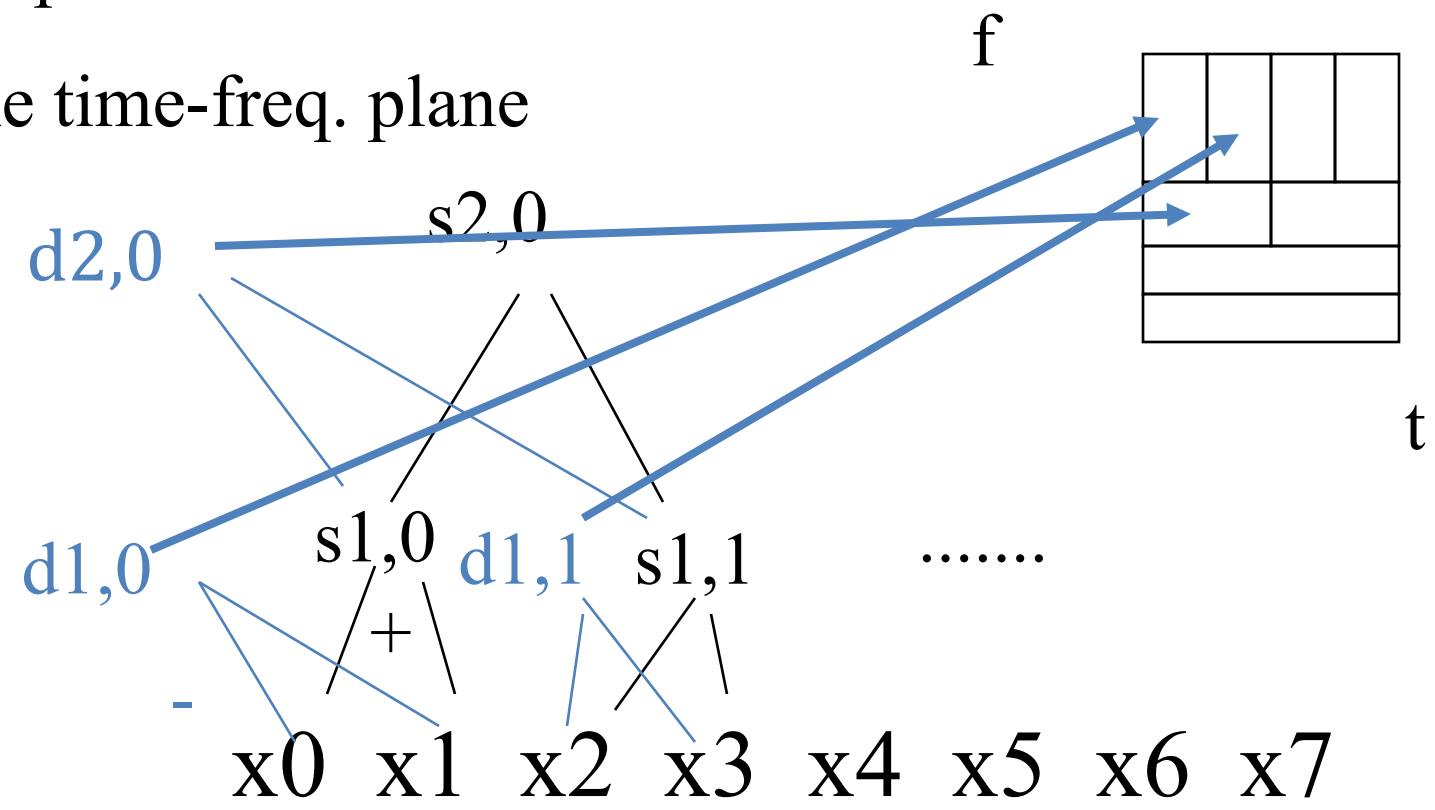
on the time-freq. plane



Wavelets - construc DETAILS

Q: map each coefficient

on the time-freq. plane



Wavelets - construc DETAILS

Observation1:

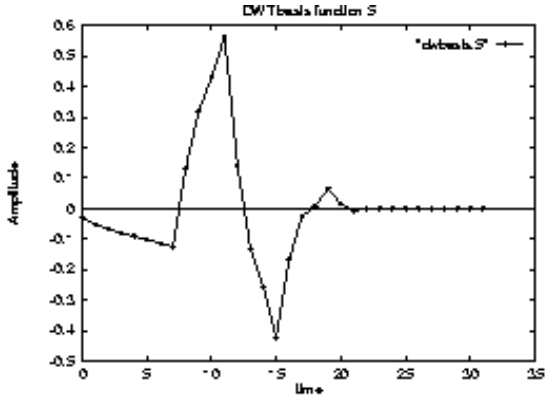
‘+’ can be some weighted addition

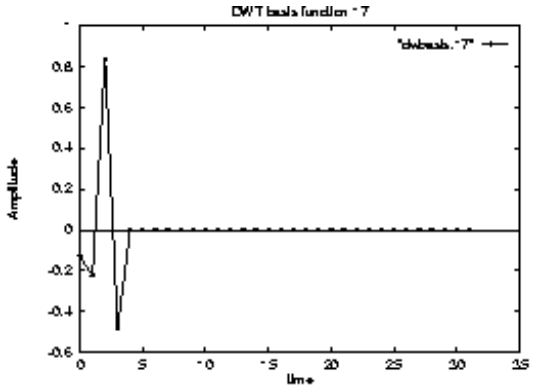
‘-’ is the corresponding weighted difference (‘Quadrature mirror filters’)

Observation2: unlike DFT/DCT,

there are *many* wavelet bases: Haar, Daubechies-4, Daubechies-6, Coifman, Morlet, Gabor, ...

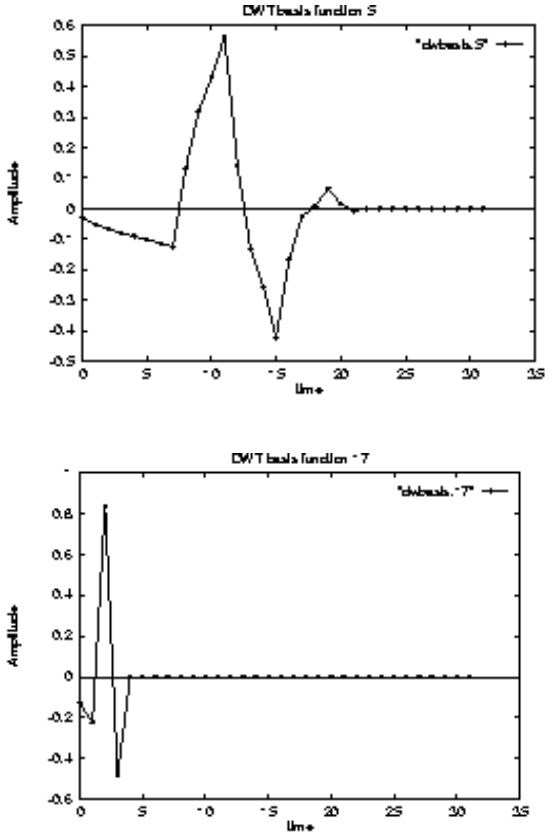
Wavelets - how do they look like?



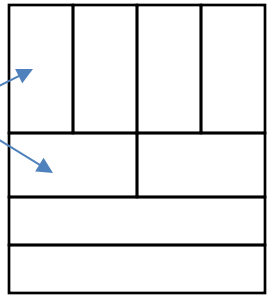


- E.g., Daubechies-4

Wavelets - how do they look like?

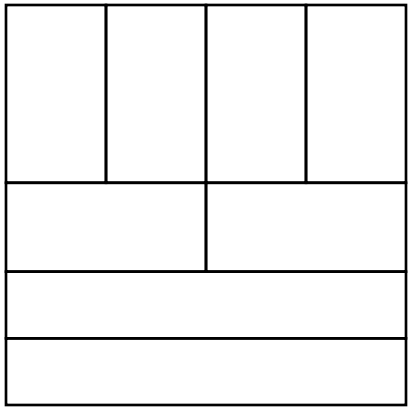


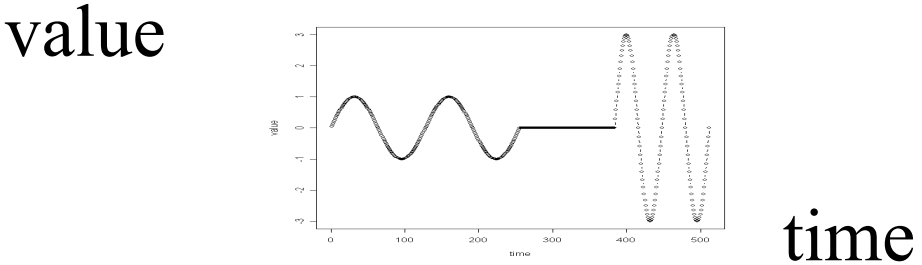
- E.g., Daubechies-4



Wavelets - Drill#1:

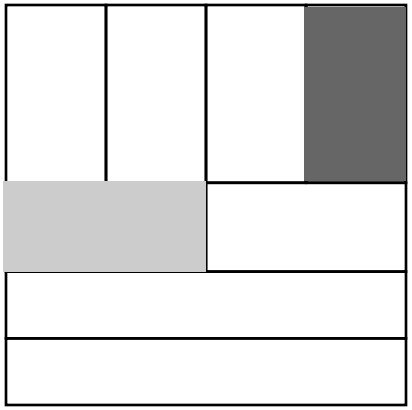
- Q: baritone/silence/soprano - DWT?

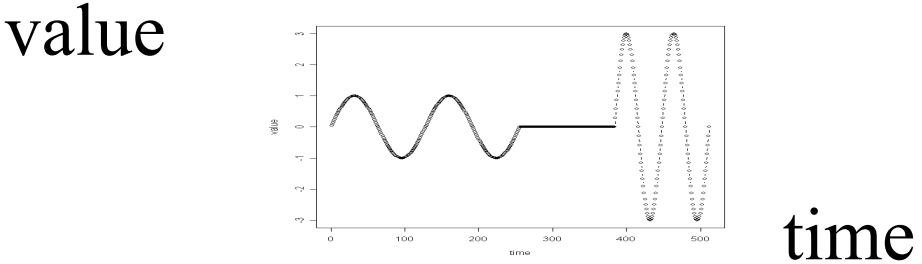




Wavelets - Drill#1:

- Q: baritone/silence/soprano - DWT?

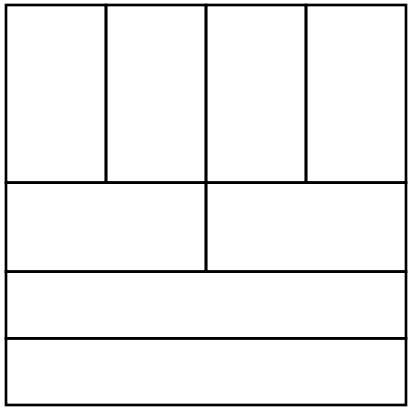




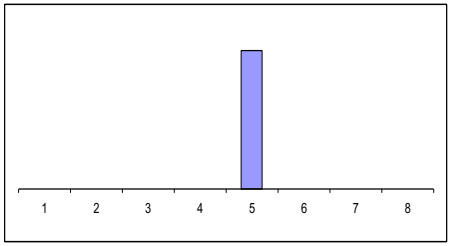
Wavelets - Drill#2:

- Q: spike - DWT?

f



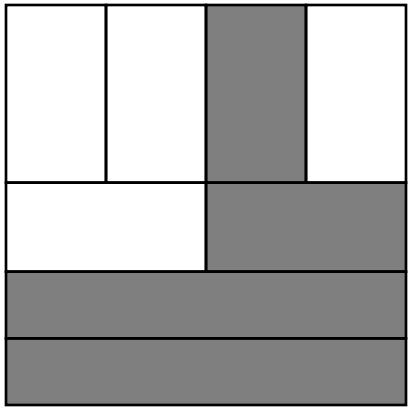
t

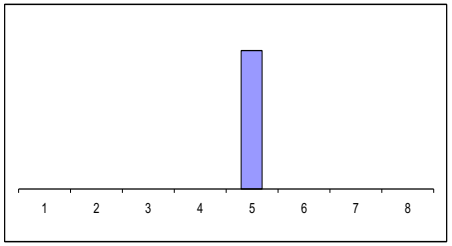


Wavelets - Drill#2:

- Q: spike - DWT?

f





t

0.00 0.00 **0.71** 0.00

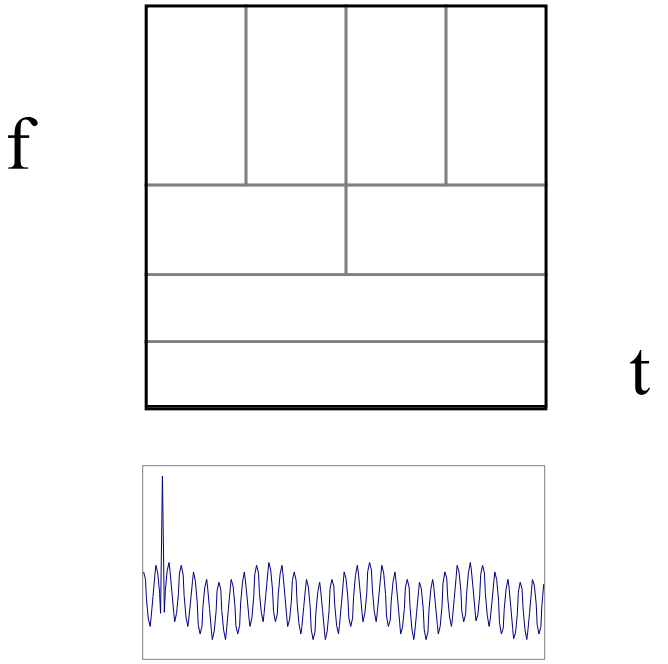
0.00 **0.50**

-0.35

0.35

Wavelets - Drill#3:

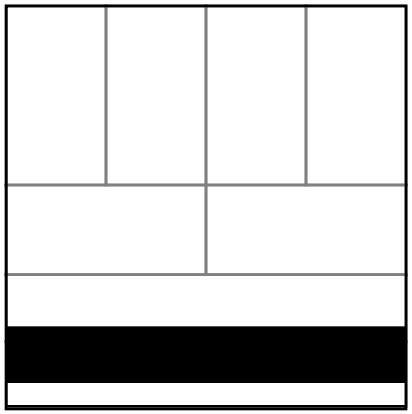
- Q: weekly + daily periodicity, + spike - DWT?



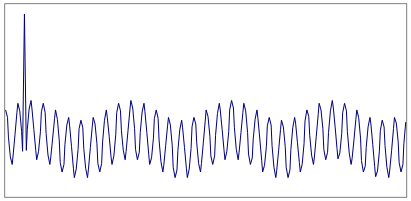
Wavelets - Drill#3:

- Q: **weekly** + daily periodicity, + spike - DWT?

f

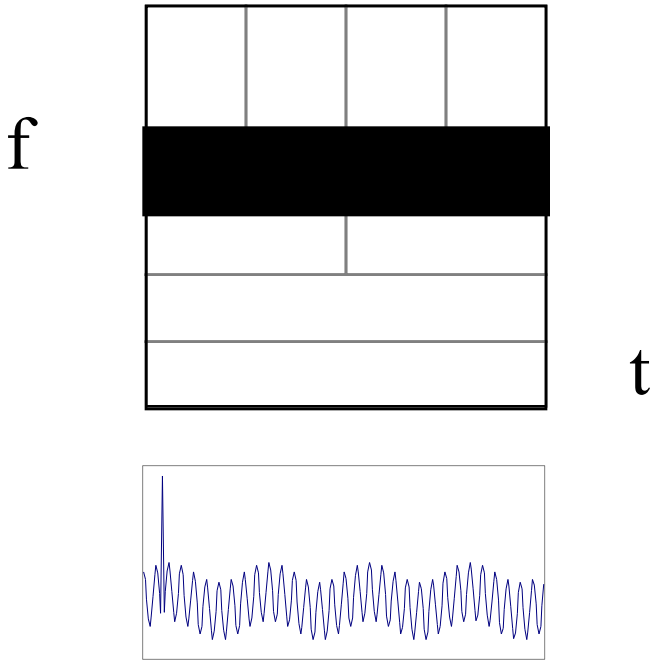


t



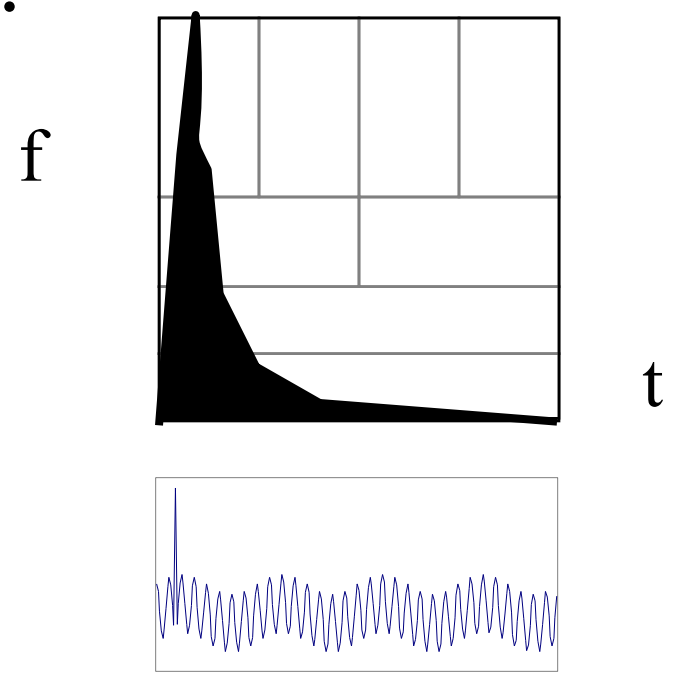
Wavelets - Drill#3:

- Q: weekly + **daily** periodicity, + spike - DWT?



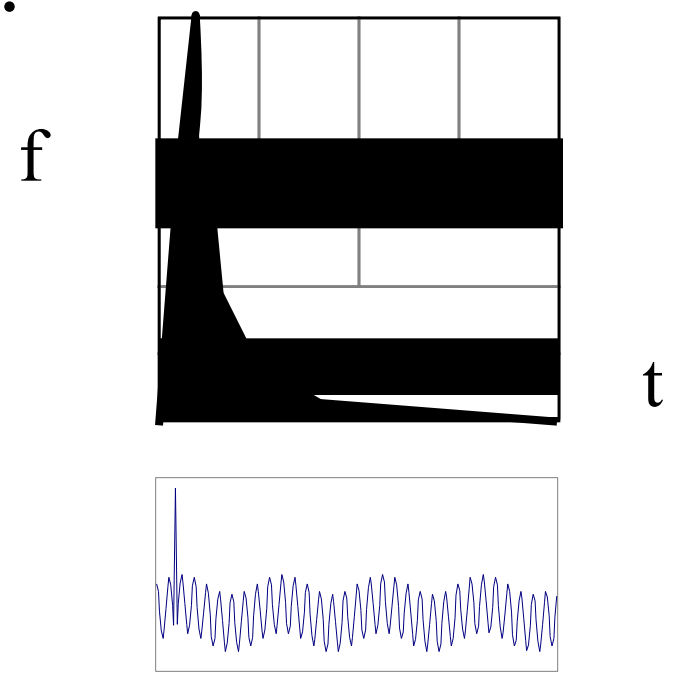
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + **spike** - DWT?



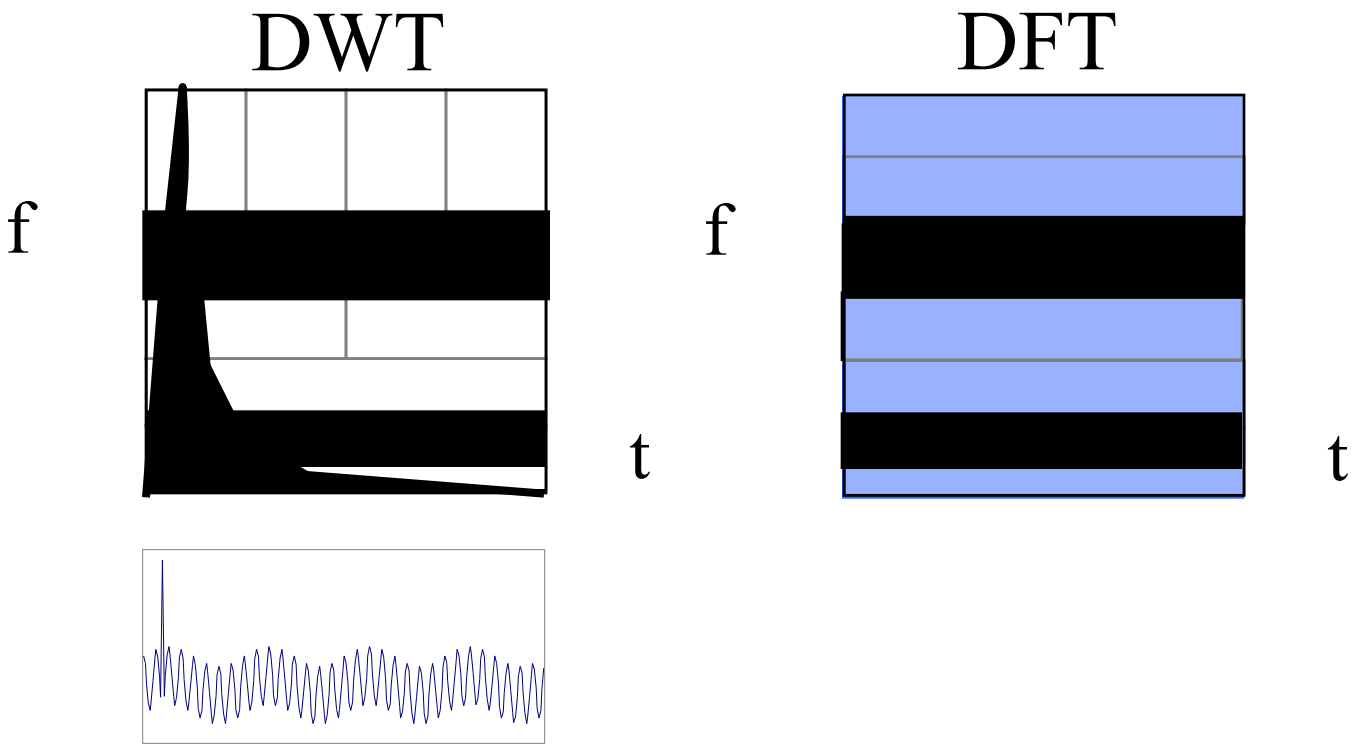
Wavelets - Drill#3:

- Q: weekly + daily periodicity, + spike - DWT?



Wavelets - Drill#3:

- Q: DFT?



Advantages of Wavelets

- Better compression (better RMSE with same number of coefficients - used in JPEG-2000)
- fast to compute (usually: O(n)!)
- very good for ‘spikes’

DFT & DWT: conclusions

- **DFT** spots periodicities (with the '**amplitude spectrum**')
  - can be quickly computed (O(n log n)), thanks to the FFT algorithm.
  - **standard** tool in signal processing (speech, image etc signals)
  - (closely related to DCT and JPEG)

DFT & DWT: conclusions

- **DWT**: multi-resolution
  - very suitable for self-similar traffic
  - used for summarization of streams [Gilbert+01], db histograms, etc

- DFT&DWT: powerful tools for **compression**, **pattern detection** in real signals
  - included in math packages (matlab, 'R', mathematica, … - often in spreadsheets!)

Resources - software and urls

- http://www.dsptutor.freeuk.com/jsanalyser/FFTSpectrumAnalyser.html : Nice java applets for FFT
- http://www.relisoft.com/freeware/freq.html voice frequency analyzer (needs microphone)

Resources: software and urls

- *xwpl:* open source wavelet package from Yale, with excellent GUI
- http://monet.me.ic.ac.uk/people/gavin/java/waveletDemos.html : wavelets and scalograms

Books

- William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery: *Numerical Recipes in C*, Cambridge University Press, 1992, 2nd Edition. (Great description, intuition and code for DFT, DWT)
- C. Faloutsos: *Searching Multimedia Databases by Content*, Kluwer Academic Press, 1996 (introduction to DFT, DWT)

Additional Reading

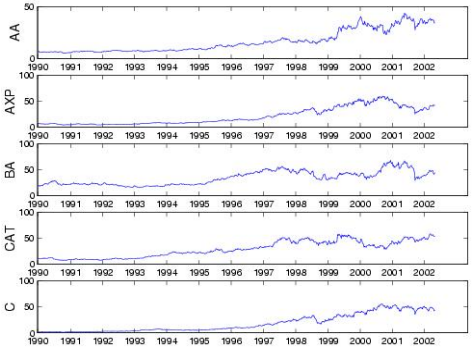
- [Gilbert+01] Anna C. Gilbert, Yannis Kotidis and S. Muthukrishnan and Martin Strauss, *Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries*, VLDB 2001

Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
  - DFT, DWT, DCT (data independent)
  - SVD, ICA (data independent)
  - MDS, FastMap
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

SVD

- Singular Value Decomposition
- THE optimal method for dimensionality reduction
  - (under the Euclidean metric)



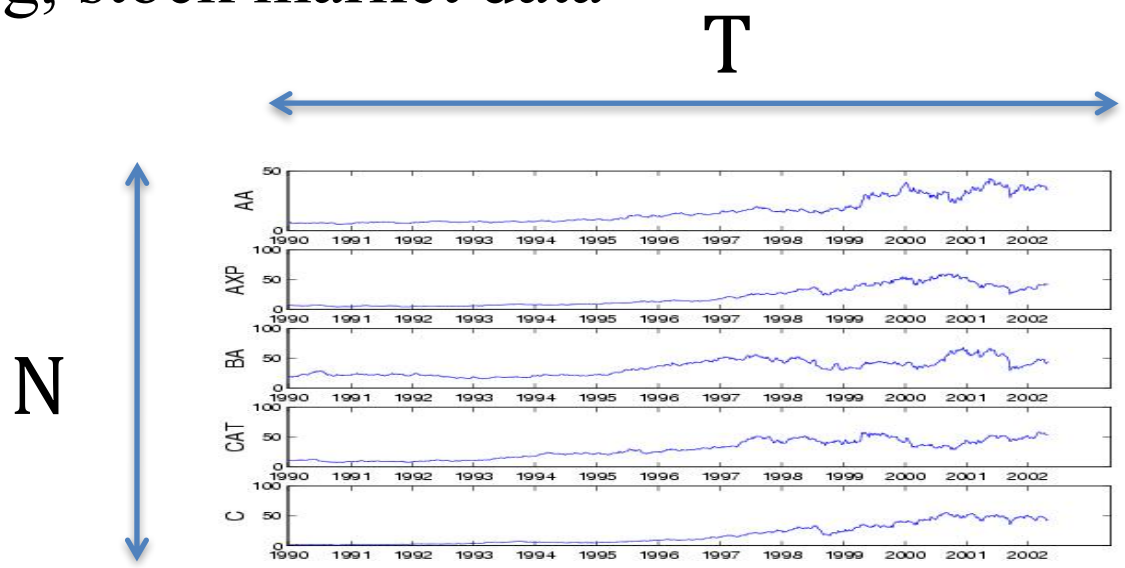
- Given: many time sequences
- Find: the latent ('hidden') variables

SVD

Two (equivalent) interpretations:

- Geometric (each sequence -> point in T-d space)
- Matrix algebra (N x T matrix)

eg, stock market data

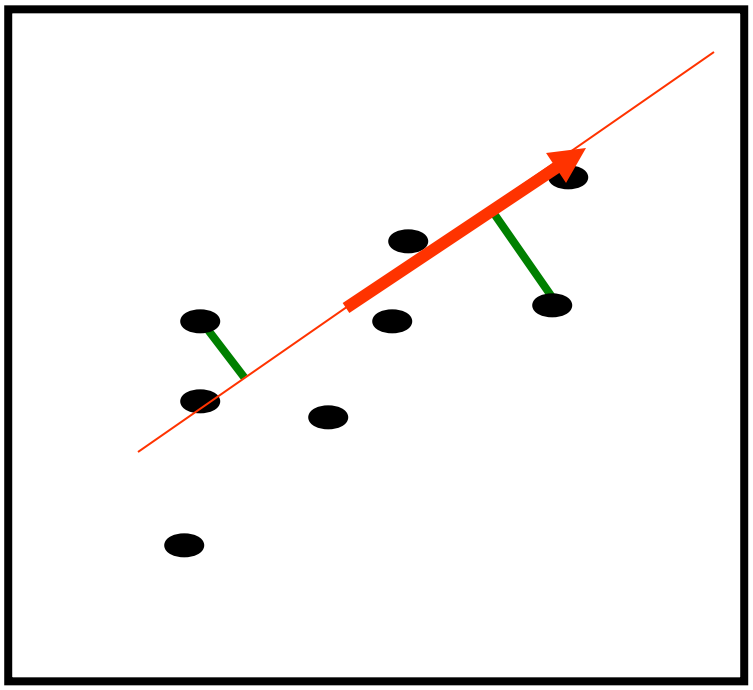


AA: Alcoa
AXP: American Express
BA: Boeing
CAT: Caterpillar
C: Citi Group

Singular Value Decomposition (SVD)

- SVD (~LSI ~ KL ~ PCA ~ spectral analysis...) – Geometric interpretation

day2



day1

LSI: S. Dumais; M. Berry

KL: eg, Duda+Hart

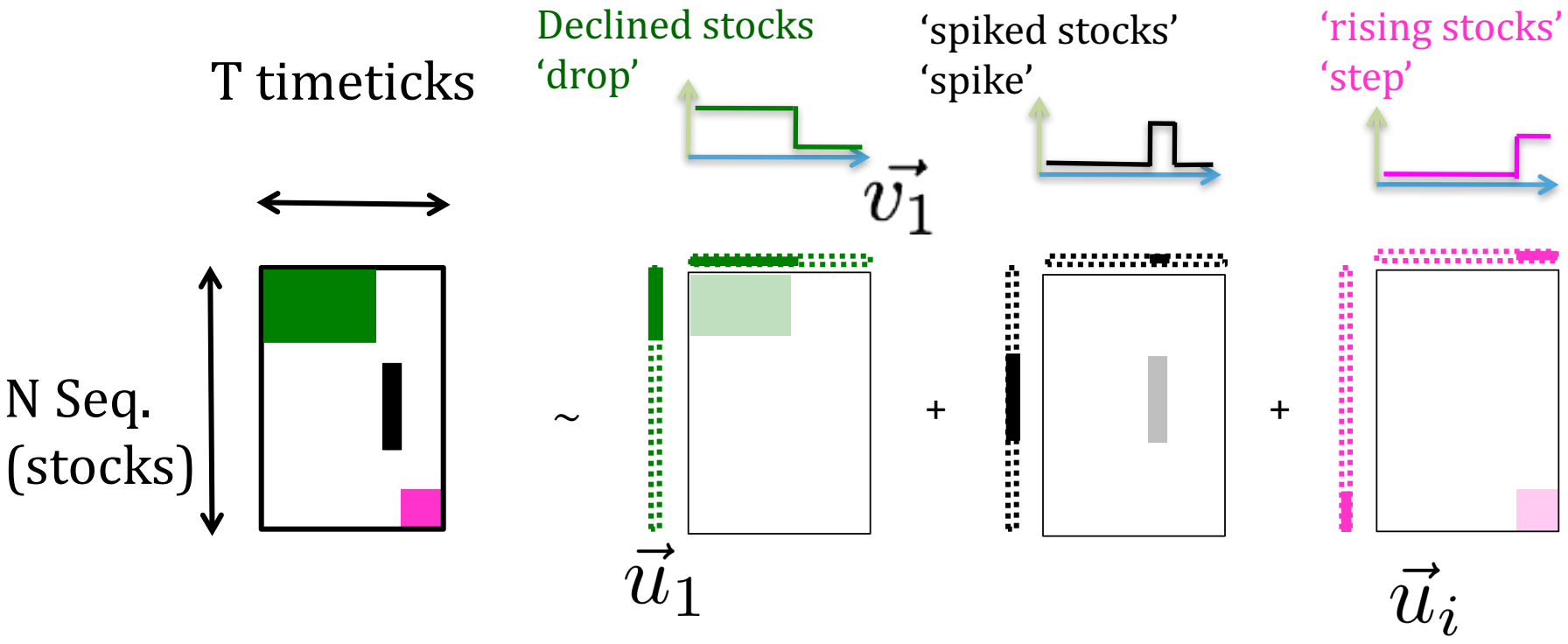
PCA: eg., Jolliffe

Details: [Press+],

[Faloutsos96]

SVD – matrix interpretation

- SVD -> matrix factorization: finds blocks



SVD

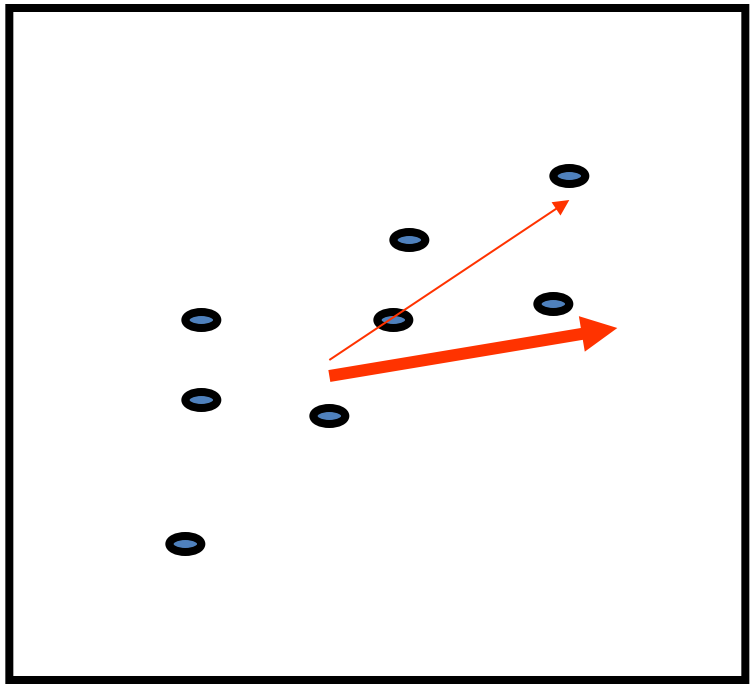
- **Extremely** useful tool
  - (also behind PageRank/google and Kleinberg's algorithm for hubs and authorities)

SVD

- **Extremely** useful tool
  - (also behind PageRank/google and Kleinberg's algorithm for hubs and authorities)
- But may be slow: O($N * M * M$) if $N>M$
- any approximate, faster method?

SVD shorcuts

- random projections (Johnson-Lindenstrauss thm [Papadimitriou+ pods98])

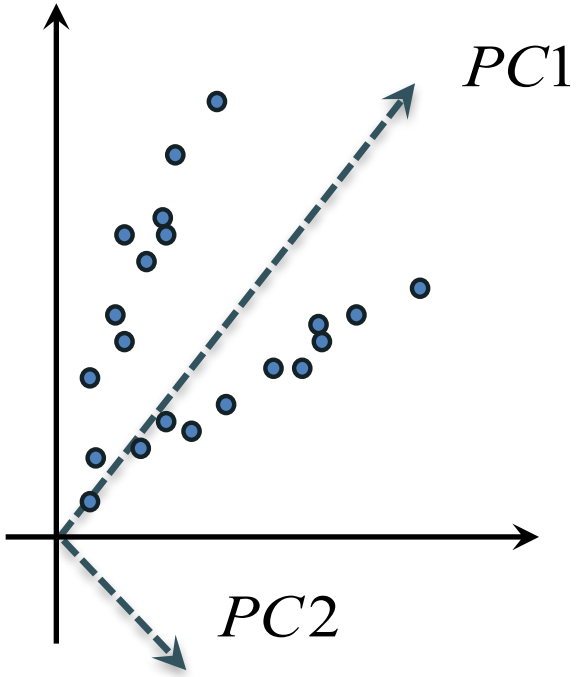


Random projections

- pick ‘enough’ random directions (will be ~orthogonal, in high-d!!)
- distances are preserved probabilistically, within epsilon
- (also, use as a pre-processing step for SVD [Papadimitriou+ PODS98])

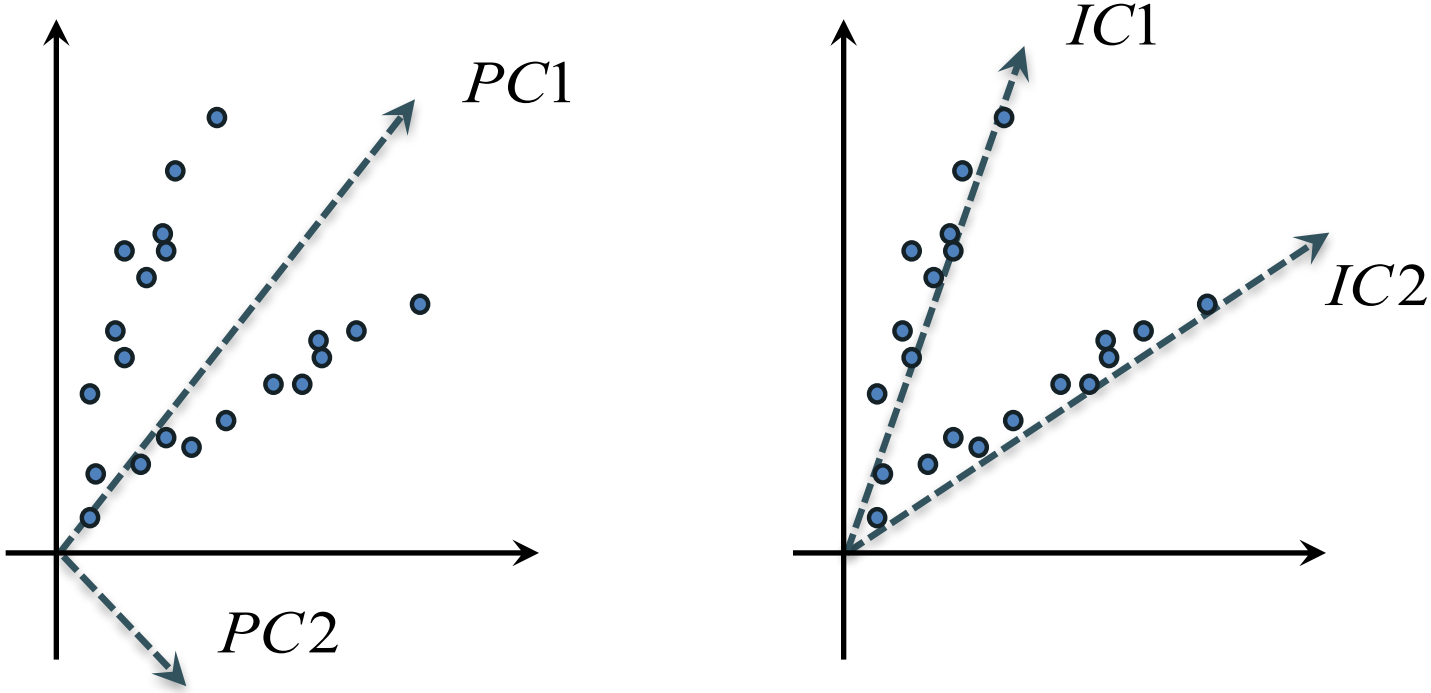
SVD & improvement

- Q: Can we do even better?
- A: sometimes, yes – by shooting for sparsity



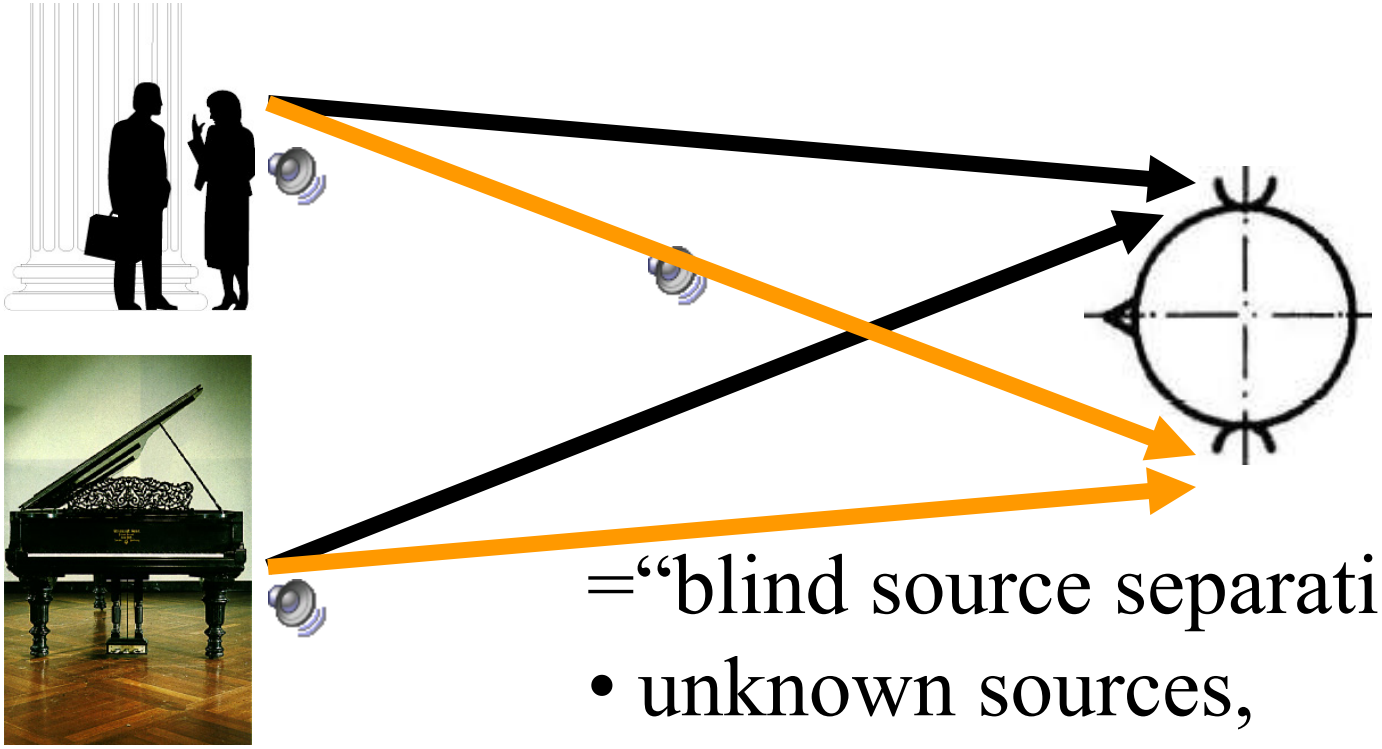
Independent Component Analysis (ICA)

- PCA (or SVD) sometimes misses essential features
  - PCA vs. ICA



A.k.a.: BSS = cocktail party problem
Find hidden variables

- Untangle two sound sources

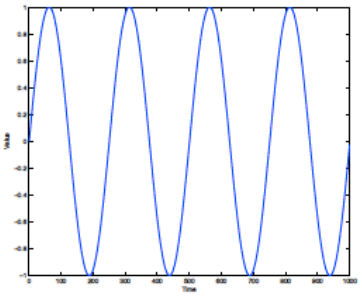


="blind source separation"
- unknown sources,
- unknown mixing

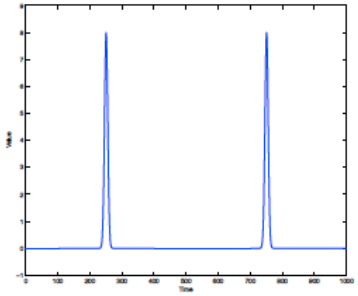
ICA

- Why not PCA

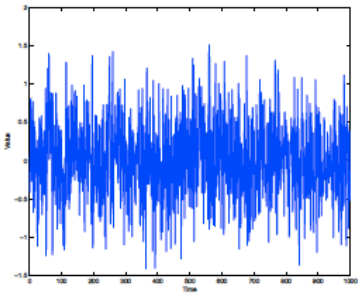
Source



Source #1

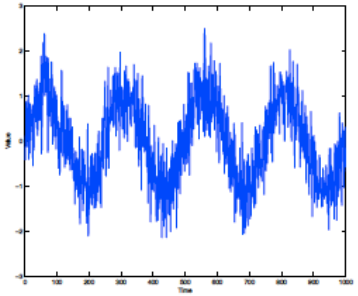


Source #2

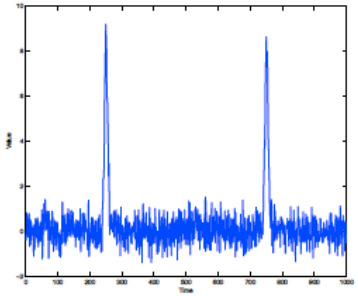


Source #3

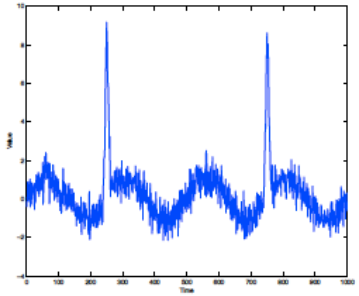
Mix



Sequence #1
(Sources #1 & #3)



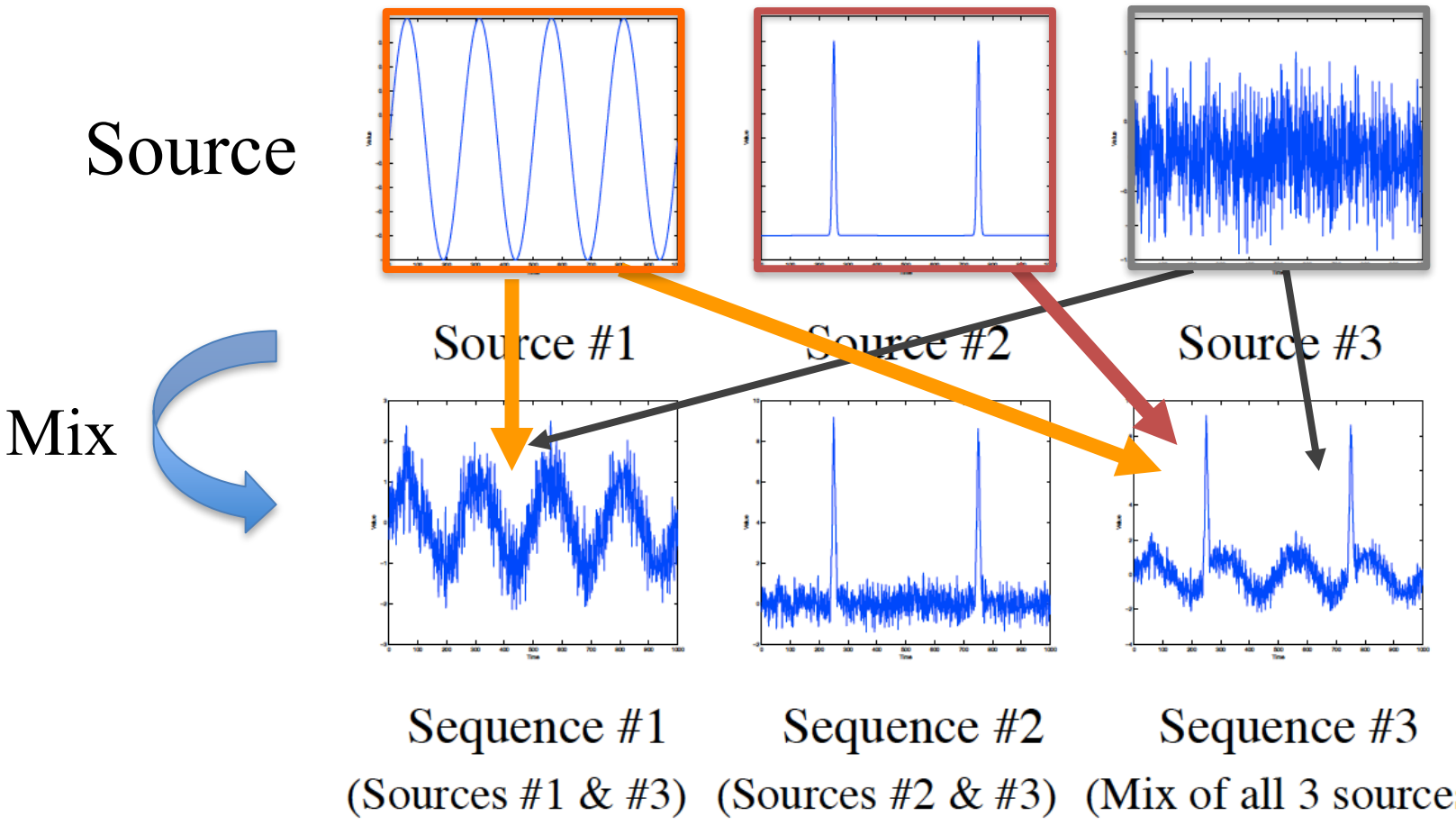
Sequence #2
(Sources #2 & #3)



Sequence #3
(Mix of all 3 sources)

ICA

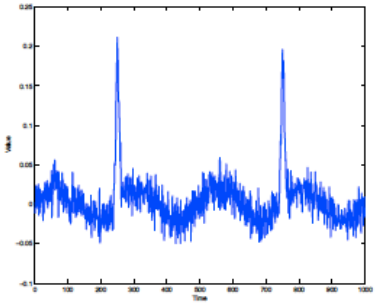
- Why not PCA



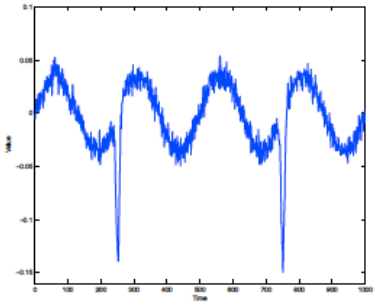
ICA

- Why not PCA

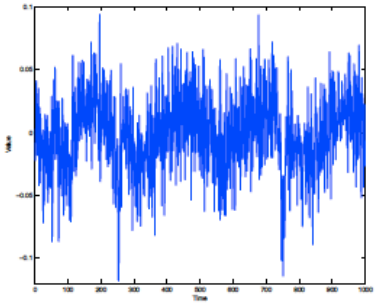
PCA



PC1

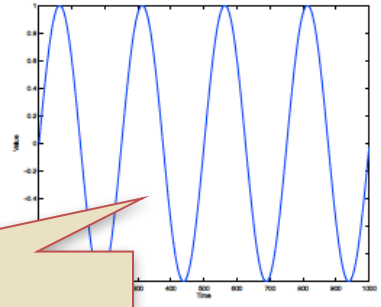


PC2

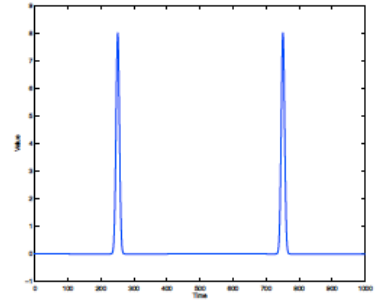


PC3

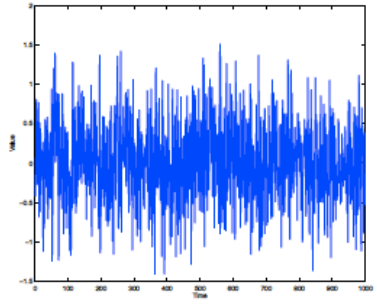
ICA



IC1



IC2

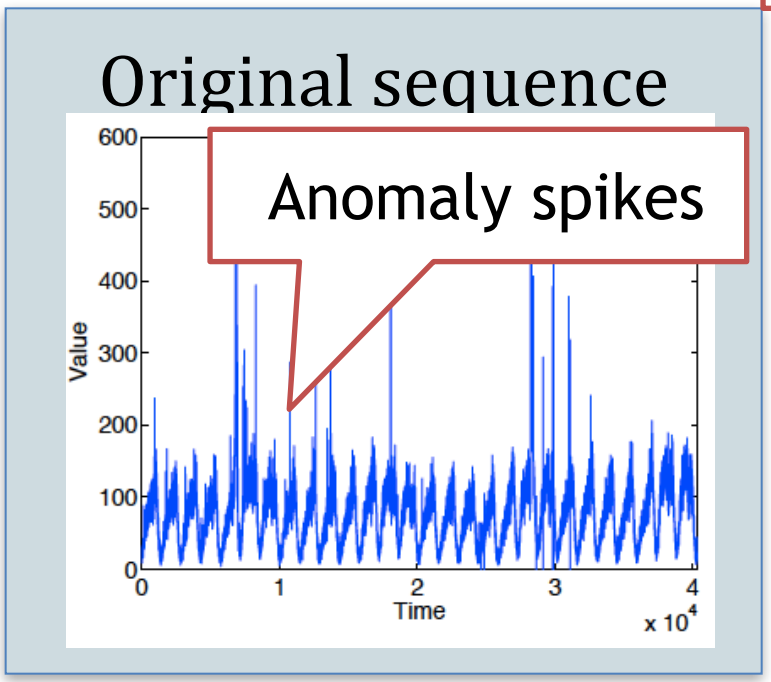


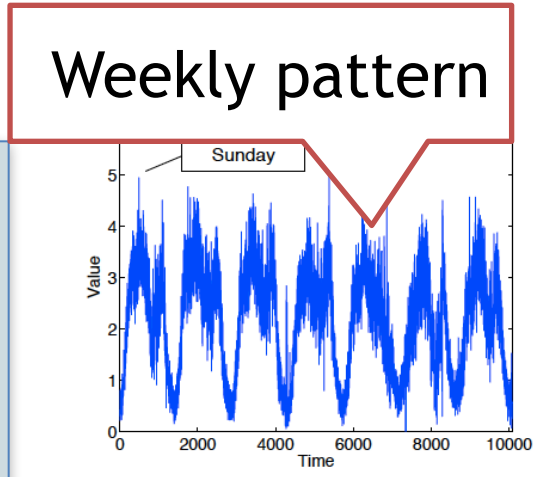
IC3

ICA recognizes the components successfully and separately

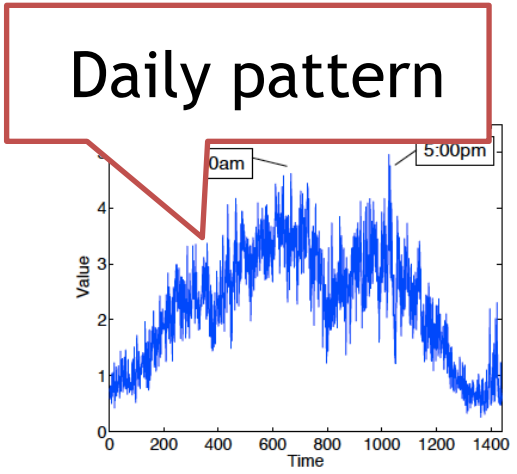
Hidden variables

- Local component analysis [Sakurai+11]

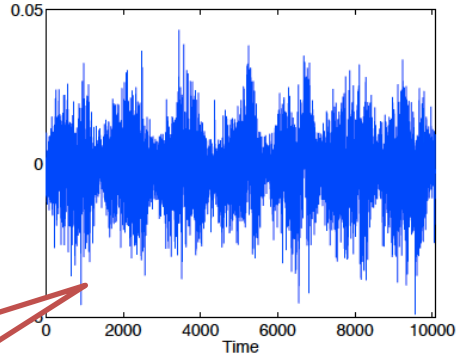




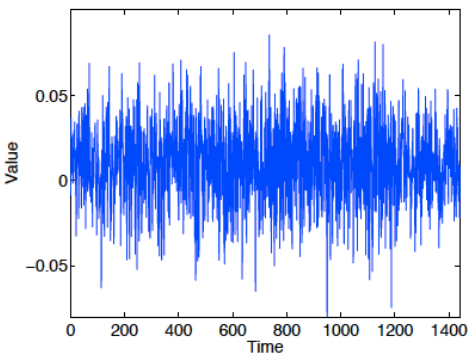
(b) Weekly pattern (*WindMine*)



(c) Daily pattern (*WindMine*)



(d) Weekly pattern (PCA)



(e) Daily pattern (PCA)

Hidden variables

Alcoa

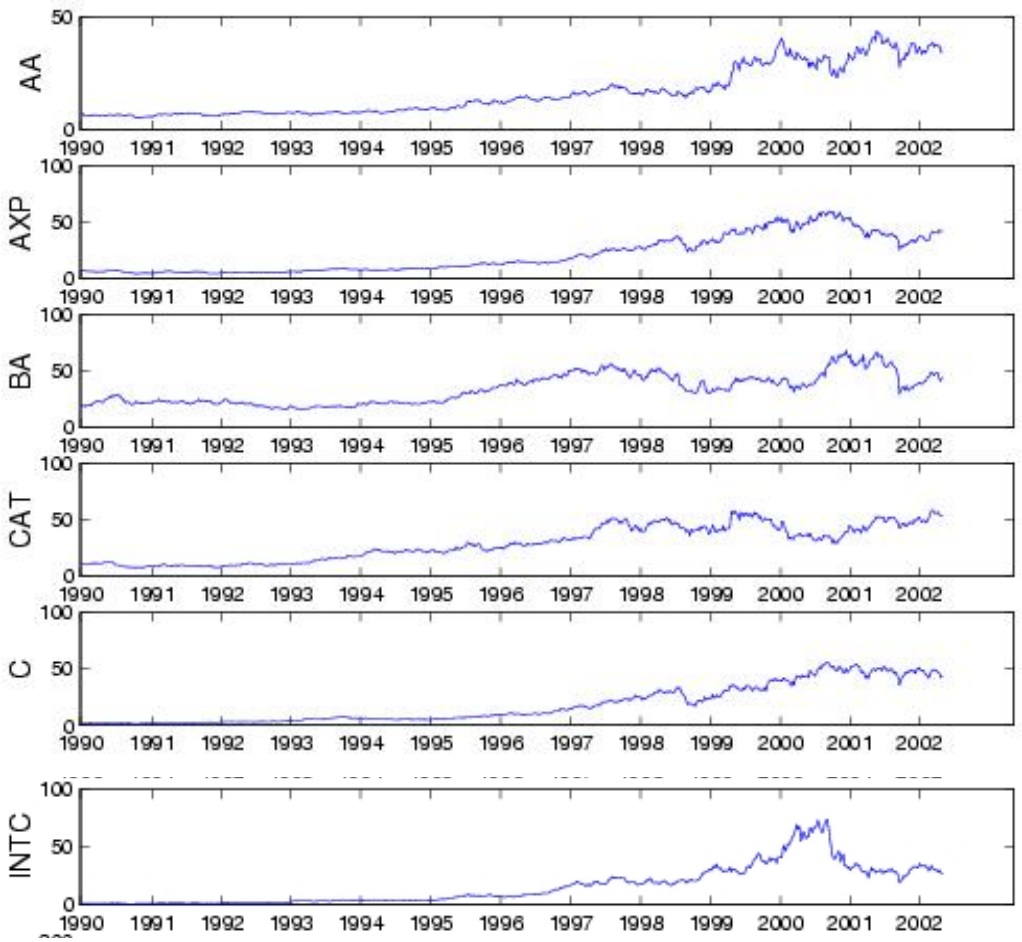
American Express

Boeing

Caterpillar

Citi Group

Intel

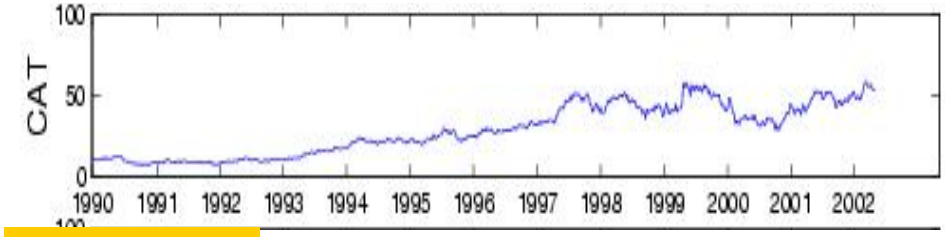


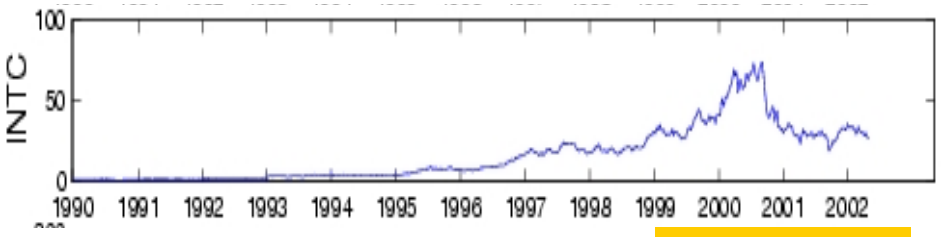
Dow Jones Industrial Average

Find common hidden variables, and weights.

Hidden variables

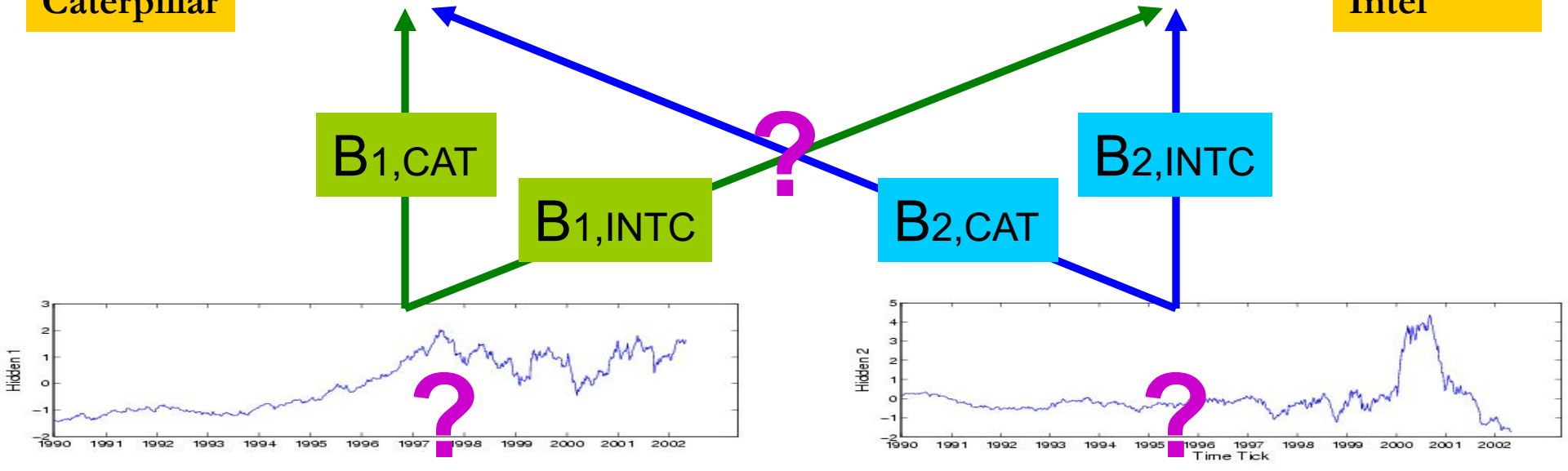
- Find hidden variables [Pan+04]





Caterpillar

Intel

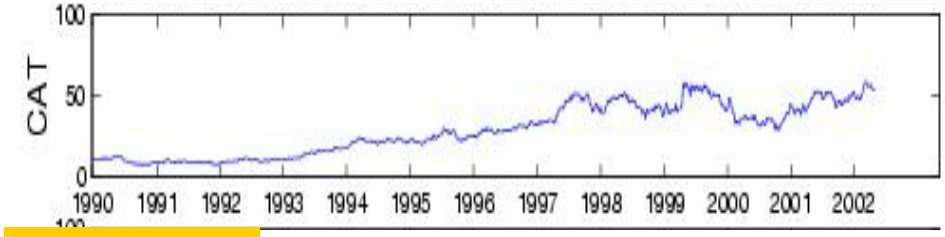


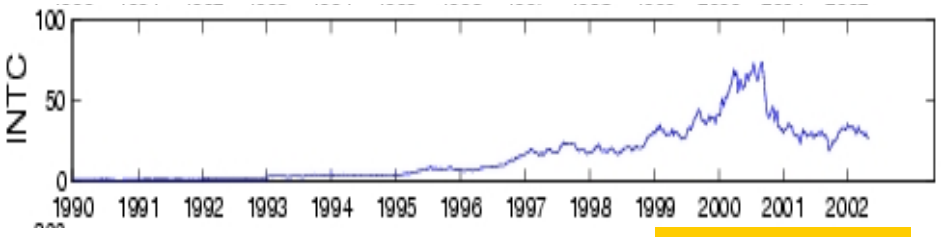
Hidden variable 1

Hidden variable 2

Hidden variables

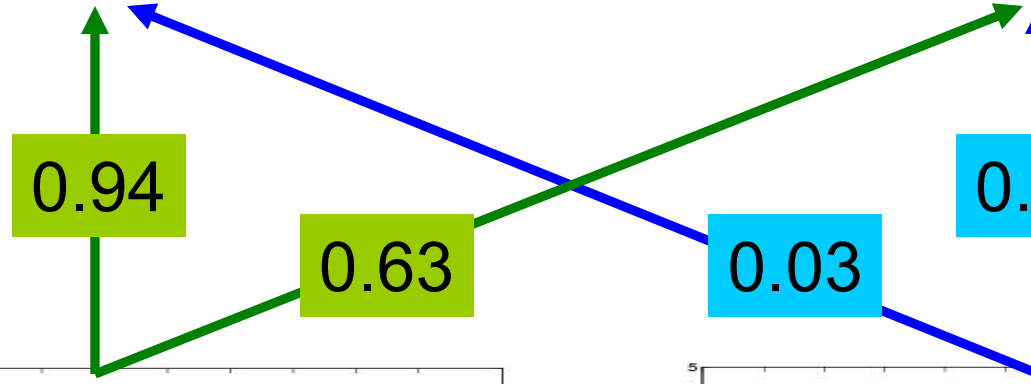
- Find hidden variables [Pan+04]

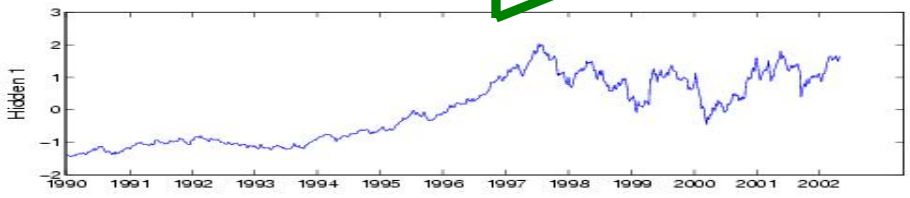


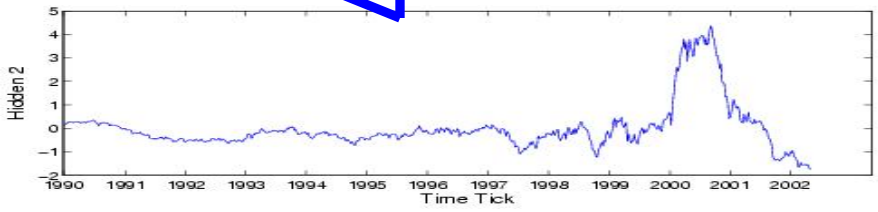


Caterpillar

Intel





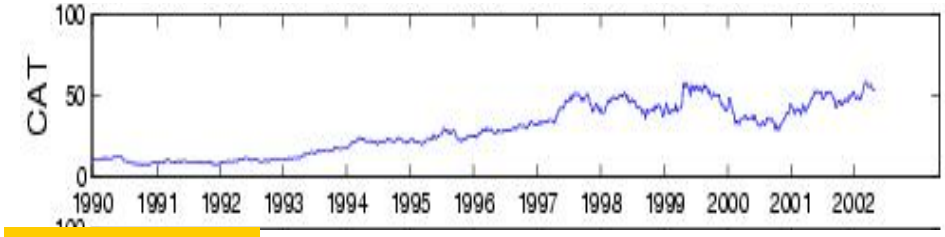


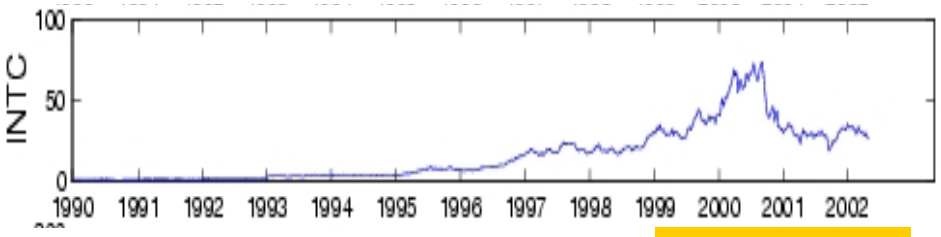
"Hidden variable 1"

"Hidden variable 2"

Hidden variables

- Find hidden variables [Pan+04]





Caterpillar

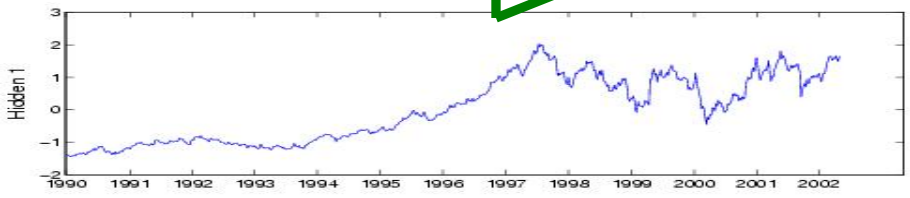
Intel

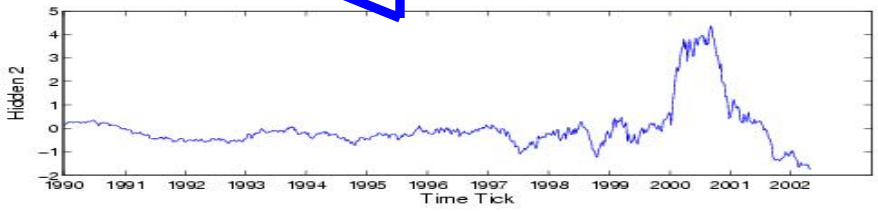
0.94

0.63

0.03

0.64





"General trend"

"Internet bubble"

Motivation: Find hidden variables

- ICA: also known as ‘Blind Source Separation’
- ‘cocktail party problem’
  - in a party, we can hear two concurrent conversations,
  - but separate them (and tune-in on one of them only)
- http://www.cnl.salk.edu/~tewon/Blind/blind_audio.html
- (in stocks: one ‘discussion’ is the general economy trend; the other ‘discussion’ is the tech-stock boom

Citation

- *AutoSplit: Fast and Scalable Discovery of Hidden Variables in Stream and Multimedia Databases,* Jia-Yu Pan, Hiroyuki Kitagawa, Christos Faloutsos and Masafumi Hamamoto, PAKDD 2004, Sydney, Australia.
- *WindMine: Fast and Effective Mining of Web-click Sequences*, Yasushi Sakurai, Lei Li, Yasuko Matsubara, Christos Faloutsos, SDM 2011, Mesa, Arizona.

Roadmap

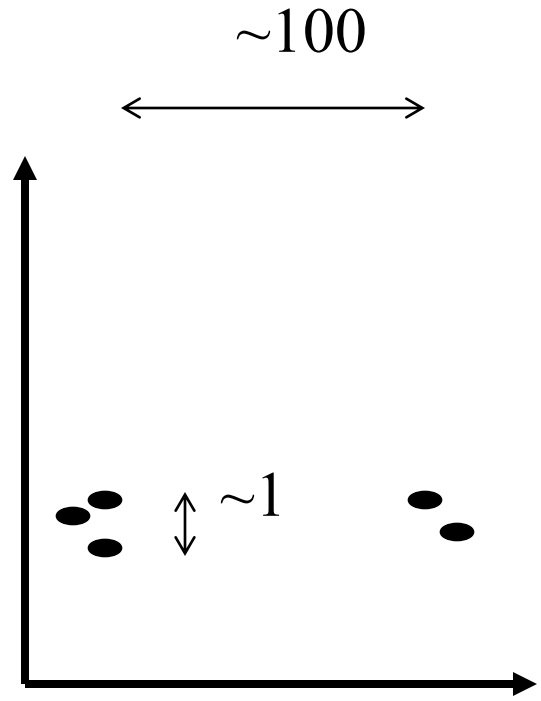
- Motivation
- Similarity Search and Indexing
- Feature extraction
  - DFT, DWT, DCT (data independent)
  - SVD, ICA (data independent)
  - MDS, FastMap
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

MDS / FastMap

- but, what if we have NO points to start with?
  (eg. Time-warping distance)
- A: Multi-dimensional Scaling (MDS) ; FastMap

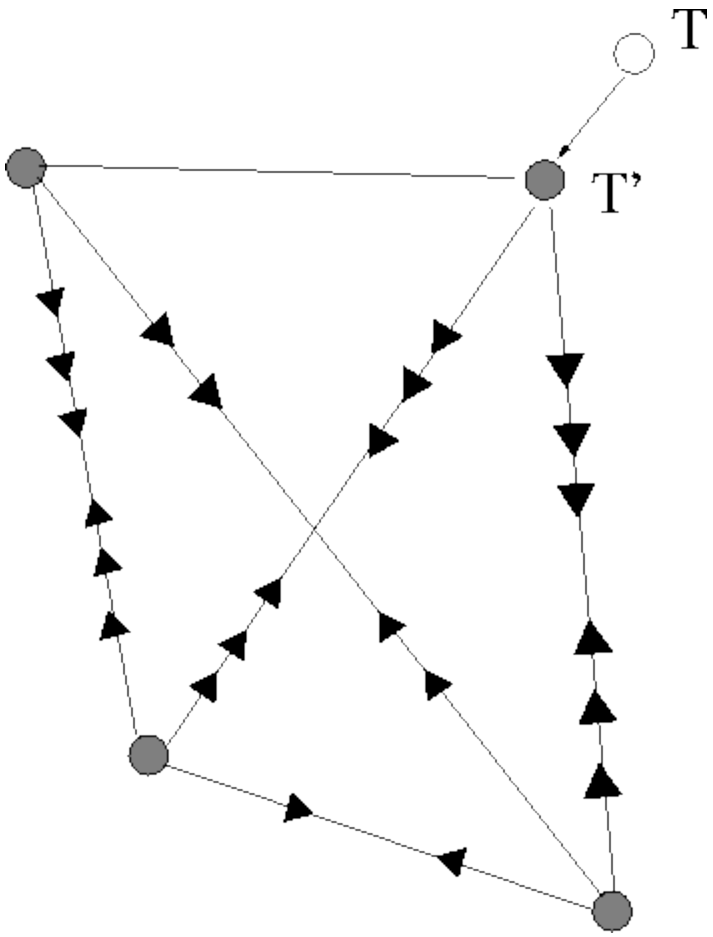
MDS/FastMap

	O1	O2	O3	O4	O5
O1	0	1	1	100	100
O2	1	0	1	100	100
O3	1	1	0	100	100
O4	100	100	100	0	1
O5	100	100	100	1	0



MDS

Multi Dimensional Scaling

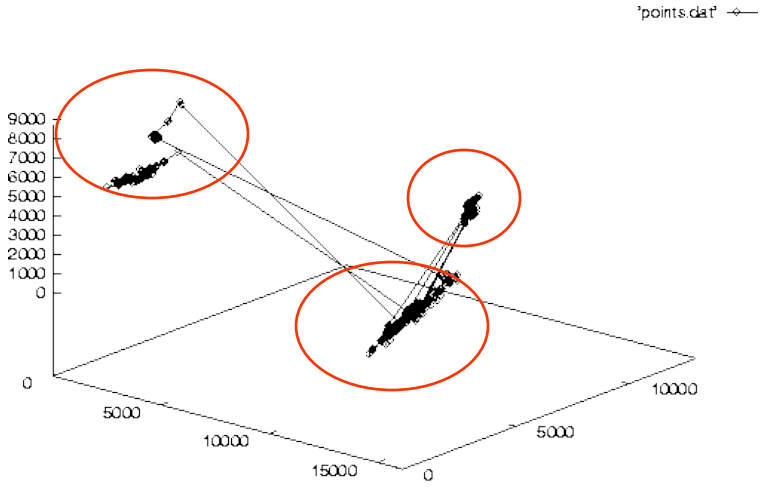


FastMap

- Multi-dimensional scaling (MDS) can do that, but in O(N**2) time
- FastMap [Faloutsos+95] takes O(N) time

FastMap: Application

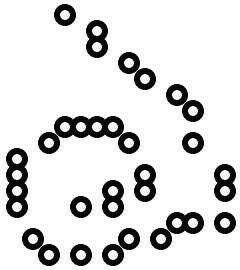
VideoTrails [Kobla+97]



scene-cut detection (about 10% errors)

Variations

- Isomap [Tenenbaum, de Silva, Langford, 2000]
- LLE (Local Linear Embedding) [Roweis, Saul, 2000]
- MVE (Minimum Volume Embedding) [Shaw & Jebara, 2007]



Conclusions - Practitioner's guide

Similarity search in time sequences

1) establish/choose distance (Euclidean, time-warping,…)

2) extract features (SVD, ICA, DWT), and use an SAM (R-tree/variant, or a Metric Tree M-tree)

2') for high intrinsic dimensionalities, consider sequential scan (it might win…)

Books

- William H. Press, Saul A. Teukolsky, William T. Vetterling and Brian P. Flannery: *Numerical Recipes in C*, Cambridge University Press, 1992, 2nd Edition. (Great description, intuition and code for SVD)
- C. Faloutsos: *Searching Multimedia Databases by Content*, Kluwer Academic Press, 1996 (introduction to SVD, and GEMINI)

References

- Agrawal, R., K.-I. Lin, et al. (Sept. 1995). Fast Similarity Search in the Presence of Noise, Scaling and Translation in Time-Series Databases. Proc. of VLDB, Zurich, Switzerland.
- Babu, S. and J. Widom (2001). "Continuous Queries over Data Streams." SIGMOD Record 30(3): 109-120.
- Breunig, M. M., H.-P. Kriegel, et al. (2000). LOF: Identifying Density-Based Local Outliers. SIGMOD Conference, Dallas, TX.
- Berry, Michael: http://www.cs.utk.edu/~lsi/

References

- Ciaccia, P., M. Patella, et al. (1997). M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. VLDB.
- Foltz, P. W. and S. T. Dumais (Dec. 1992). "Personalized Information Delivery: An Analysis of Information Filtering Methods." Comm. of ACM (CACM) 35(12): 51-60.
- Guttman, A. (June 1984). R-Trees: A Dynamic Index Structure for Spatial Searching. Proc. ACM SIGMOD, Boston, Mass.

References

- Gaede, V. and O. Guenther (1998). "Multidimensional Access Methods." Computing Surveys 30(2): 170-231.
- Gehrke, J. E., F. Korn, et al. (May 2001). On Computing Correlated Aggregates Over Continual Data Streams. ACM Sigmod, Santa Barbara, California.

References

- Gunopulos, D. and G. Das (2001). Time Series Similarity Measures and Time Series Indexing. SIGMOD Conference, Santa Barbara, CA.
- Eamonn J. Keogh, Themis Palpanas, Victor B. Zordan, Dimitrios Gunopulos, Marc Cardle: Indexing Large Human-Motion Databases. VLDB 2004: 780-791

References

- Hatonen, K., M. Klemettinen, et al. (1996). Knowledge Discovery from Telecommunication Network Alarm Databases. ICDE, New Orleans, Louisiana.
- Jolliffe, I. T. (1986). Principal Component Analysis, Springer Verlag.

References

- Keogh, E. J., K. Chakrabarti, et al. (2001). Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. SIGMOD Conference, Santa Barbara, CA.
- Kobla, V., D. S. Doermann, et al. (Nov. 1997). VideoTrails: Representing and Visualizing Structure in Video Sequences. ACM Multimedia 97, Seattle, WA.

References

- Oppenheim, I. J., A. Jain, et al. (March 2002). A MEMS Ultrasonic Transducer for Resident Monitoring of Steel Structures. SPIE Smart Structures Conference SS05, San Diego.
- Papadimitriou, C. H., P. Raghavan, et al. (1998). Latent Semantic Indexing: A Probabilistic Analysis. PODS, Seattle, WA.
- Rabiner, L. and B.-H. Juang (1993). Fundamentals of Speech Recognition, Prentice Hall.

References

- Traina, C., A. Traina, et al. (October 2000). Fast feature selection using the fractal dimension,. XV Brazilian Symposium on Databases (SBBD), Paraiba, Brazil.

References

- Dennis Shasha and Yunyue Zhu *High Performance Discovery in Time Series: Techniques and Case Studies* Springer 2004
- Yunyue Zhu, Dennis Shasha ``*StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time*'' VLDB, August, 2002. pp. 358-369.
- Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. *The Design of an Acquisitional Query Processor for Sensor Networks*. SIGMOD, June 2003, San Diego, CA.

References

- Lawrence Saul & Sam Roweis. *An Introduction to Locally Linear Embedding* (draft)
- Sam Roweis & Lawrence Saul. *Nonlinear dimensionality reduction by locally linear embedding*. Science, v.290 no.5500 , Dec.22, 2000. pp.2323--2326.
- B. Shaw and T. Jebara. "*Minimum Volume Embedding*" . Artificial Intelligence and Statistics, AISTATS, March 2007.

References

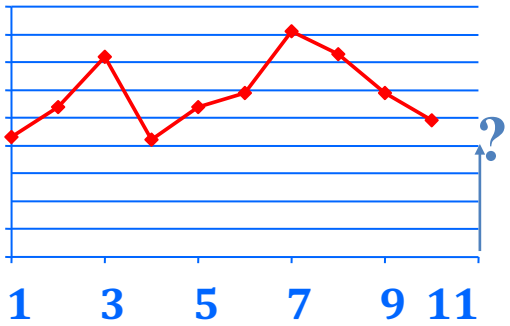
- Josh Tenenbaum, Vin de Silva and John Langford. *A Global Geometric Framework for Nonlinear dimensionality Reduction*. Science 290, pp. 2319-2323, 2000.

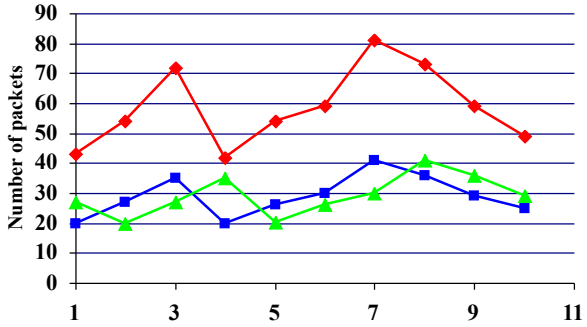
Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Wish list

- Problem 1: find patterns/**rules**
- Problem 2: **forecast**
- Problem 3: find patterns/rules/forecast, with **many** time sequences





Forecasting

"Prediction is very difficult, especially about the future." - Niels Bohr

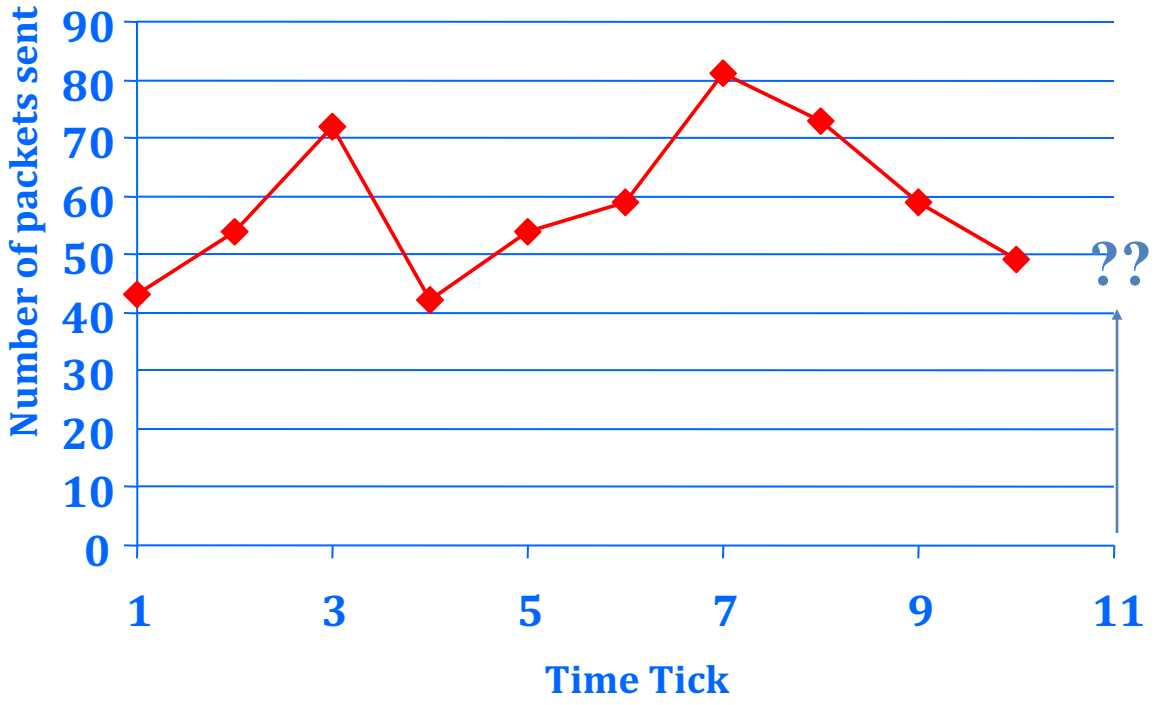
http://www.hfac.uh.edu/MediaFutures/thoughts.html

Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
- Streaming pattern discovery
- Automatic mining

Problem: Forecasting

- Example: give x_{t-1}, x_{t-2}, …, forecast x_t

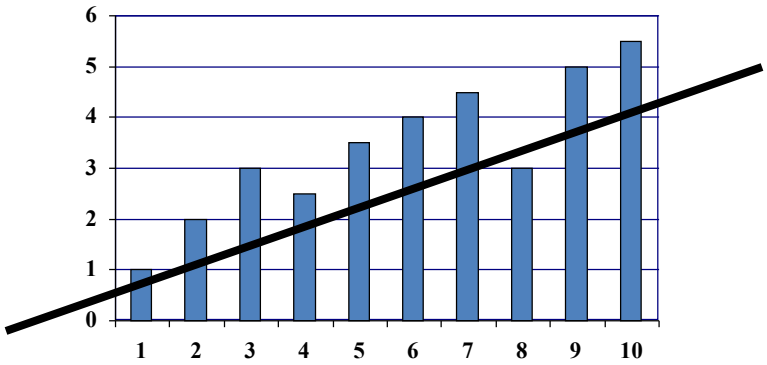


Forecasting: Preprocessing

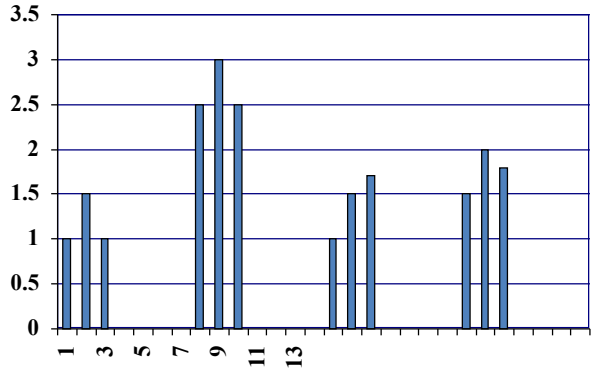
MANUALLY:

remove trends
periodicities

spot
7 days



time



time

Problem: Forecast

- Solution: try to express

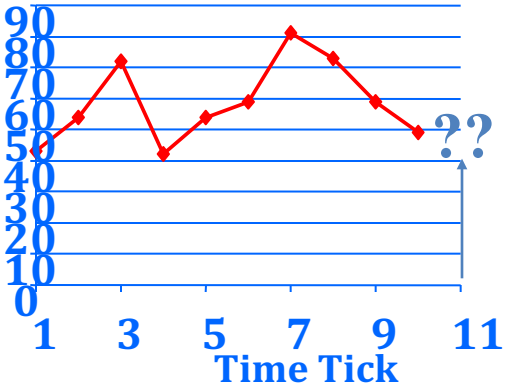
  x_t

  as a linear function of the past: x_{t-2}, x_{t-2}, …,

  (up to a window of w)

Formally:

$$x_t \approx a_1 x_{t-1} + \ldots + a_w x_{t-w} + noise$$



(if we **know** it is a non-linear model, see Part 2)

(Problem: Back-cast; interpolate)

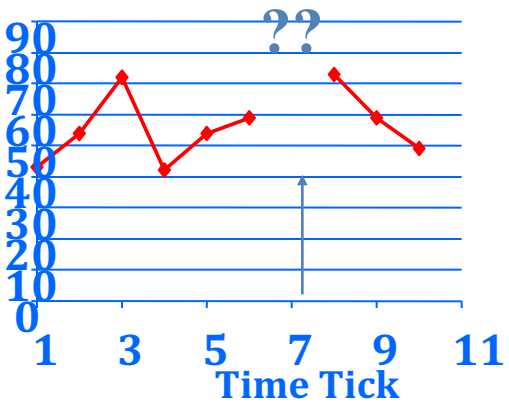
- Solution - interpolate: try to express

  x_t

  as a linear function of the past AND the future:

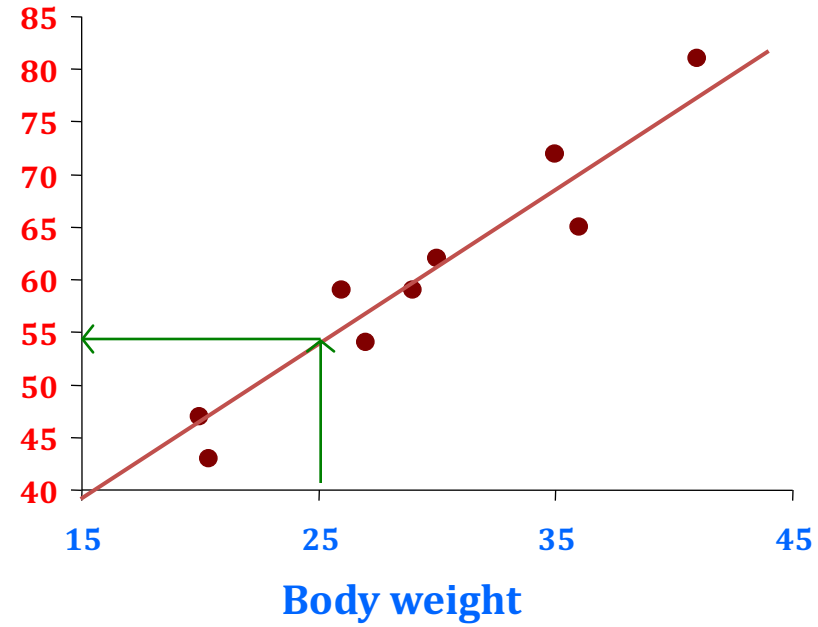
  $x_{t+1}, x_{t+2}, \ldots x_{t+w\ future}; x_{t-1}, \ldots x_{t-w\ past}$

  (up to windows of w_{past} , w_{future})
- EXACTLY the same algo's



Background: Linear Regression

patient	*weight*	*height*
1	27	43
2	43	54
3	54	72
…	…	…
N	25	??



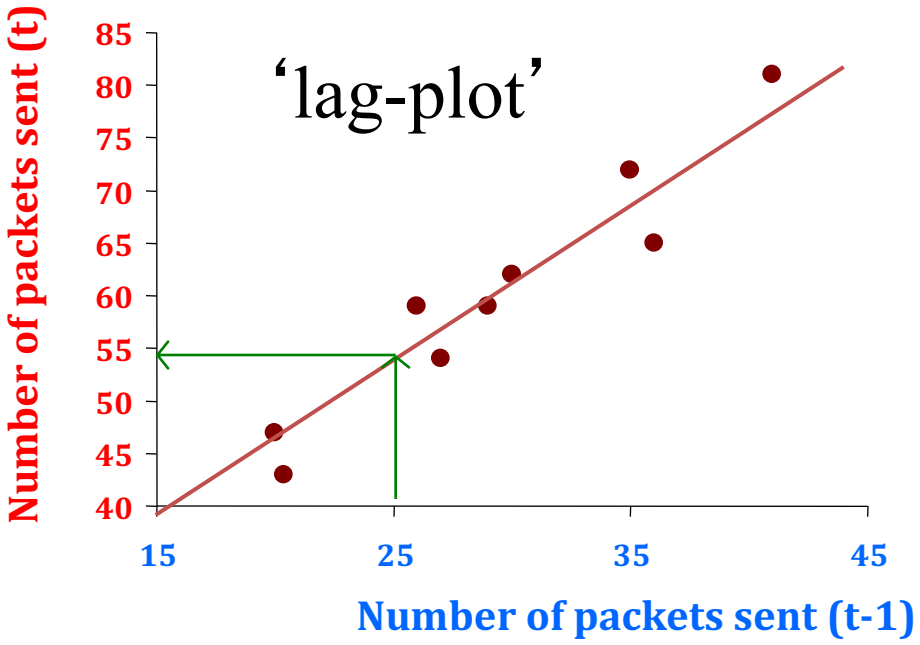
- express what we don't know (= 'dependent variable')
- as a linear function of what we know (= 'indep. variable(s)')

Linear Auto Regression:

Time	*Packets Sent(t)*
1	43
2	54
3	72
…	…
N	??

Linear Auto Regression:

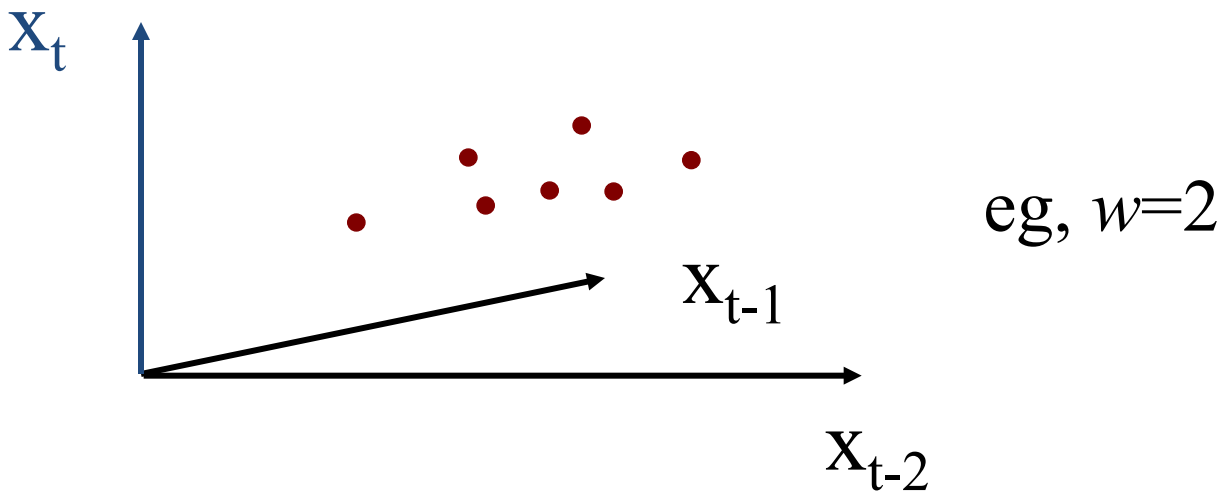
Time	*Packets Sent (t-1)*	*Packets Sent(t)*
1	-	43
2	43	54
3	54	72
	…	
…		…
N	25	??



- lag w=1
- Dependent variable = # of packets sent (S [t])
- Independent variable = # of packets sent (S[t-1])

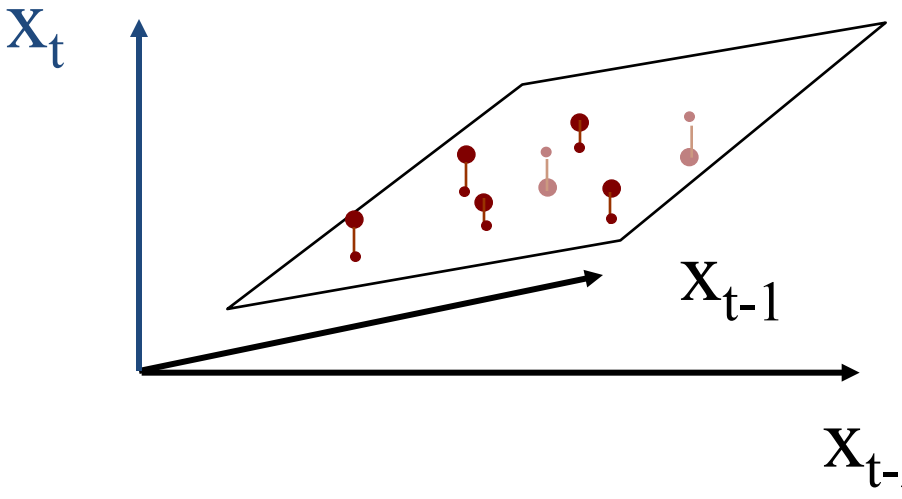
More details:

- Q1: Can it work with window $w>1$?
- A1: YES!



More details:

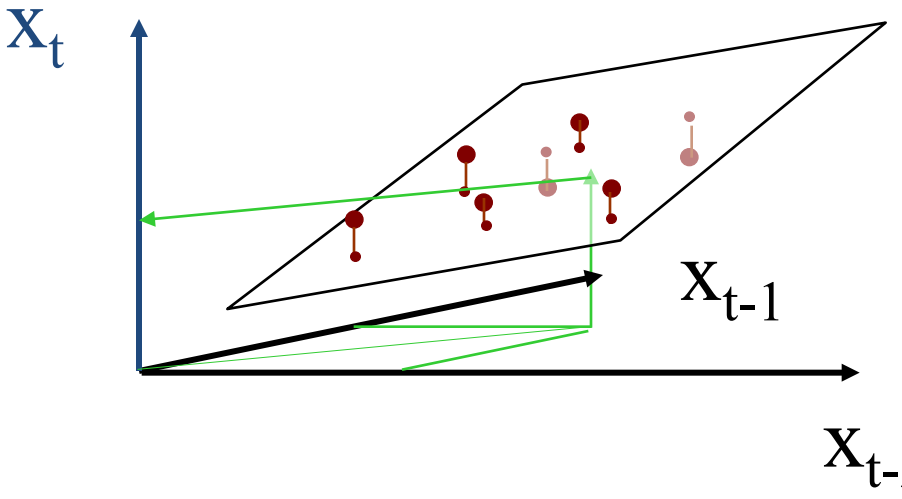
- Q1: Can it work with window $w>1$?
- A1: YES! (we'll fit a hyper-plane, then!)



eg, $w=2$

More details:

- Q1: Can it work with window $w>1$?
- A1: YES! (we'll fit a hyper-plane, then!)



eg, $w=2$

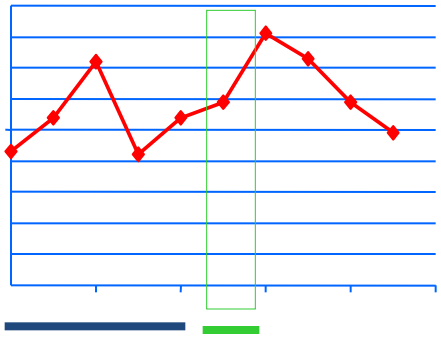
More details: DETAILS

- Q1: Can it work with window w>1?
- A1: YES! The problem becomes:

$$\mathbf{X}_{[N \times w]} \times \mathbf{a}_{[w \times 1]} = \mathbf{y}_{[N \times 1]}$$

- OVER-CONSTRAINED
  - **a** is the vector of the regression coefficients
  - **X** has the N values of the w indep. variables
  - **y** has the N values of the dependent variable

More details: DETAILS



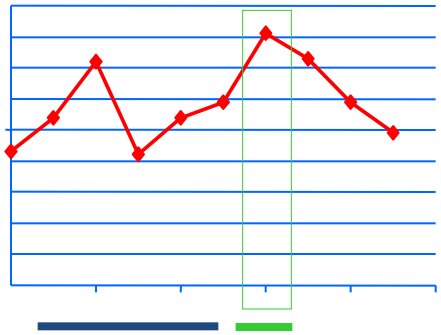
- $\mathbf{X}_{[N \times w]} \times \mathbf{a}_{[w \times 1]} = \mathbf{y}_{[N \times 1]}$

Ind-var1 Ind-var-w

time

$$\begin{bmatrix} X_{11}, X_{12}, \ldots, X_{1w} \\ X_{21}, X_{22}, \ldots, X_{2w} \\ \vdots \\ \vdots \\ \vdots \\ X_{N1}, X_{N2}, \ldots, X_{Nw} \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_w \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ \vdots \\ y_N \end{bmatrix}$$

More details: DETAILS



- $\mathbf{X}_{[N \times w]} \times \mathbf{a}_{[w \times 1]} = \mathbf{y}_{[N \times 1]}$

Ind-var1 Ind-var-w

time

$$\begin{bmatrix} X_{11}, X_{12}, \ldots, X_{1w} \\ X_{21}, X_{22}, \ldots, X_{2w} \\ \vdots \\ \vdots \\ \vdots \\ X_{N1}, X_{N2}, \ldots, X_{Nw} \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_w \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ \vdots \\ y_N \end{bmatrix}$$

More details

DETAILS

- Q2: How to estimate $a_1, a_2, \ldots a_w = \mathbf{a}$?
- A2: with Least Squares fit

$$\mathbf{a} = (\mathbf{X}^T \times \mathbf{X})^{-1} \times (\mathbf{X}^T \times \mathbf{y})$$

- (Moore-Penrose pseudo-inverse)
- **a** is the vector that minimizes the RMSE from **y**

Even more details

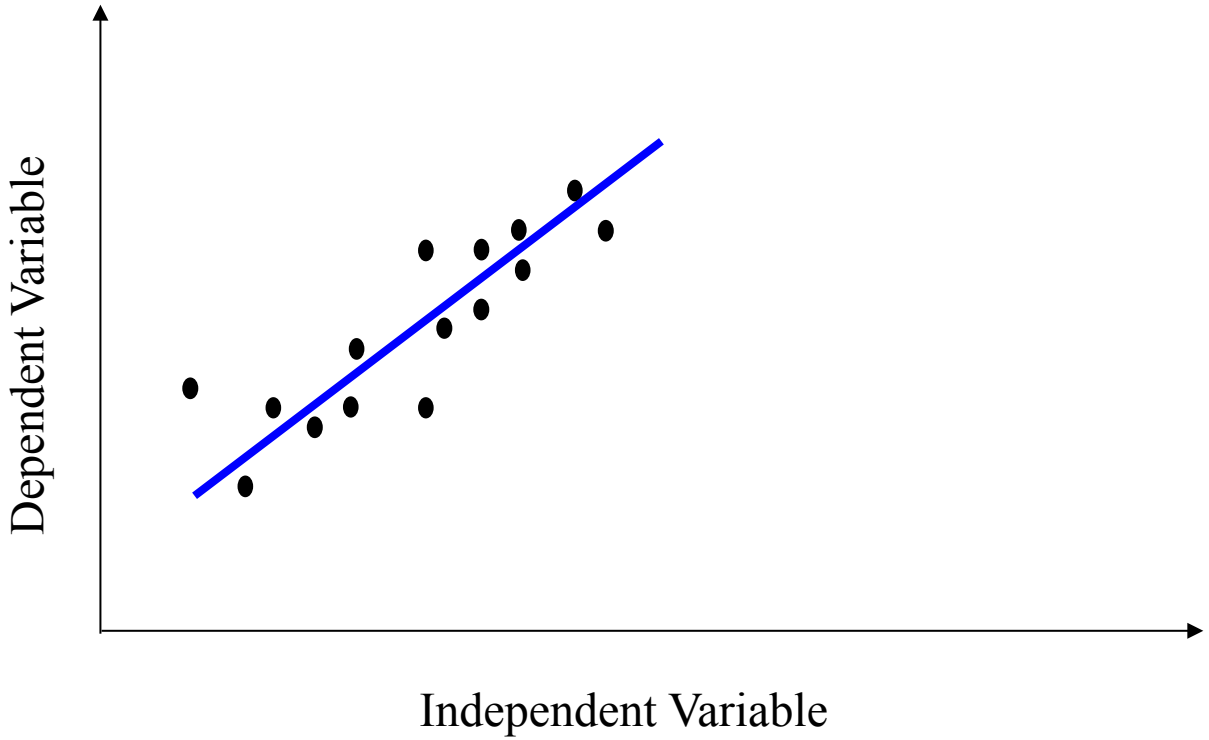
DETAILS

- Q3: Can we estimate **a** incrementally?
- A3: Yes, with the brilliant, classic method of 'Recursive Least Squares' (RLS) (see, e.g., [Yi+00], for details) - pictorially:

[Yi+00] Byoung-Kee Yi et al.: *Online Data Mining for Co-Evolving Time Sequences*, ICDE 2000.

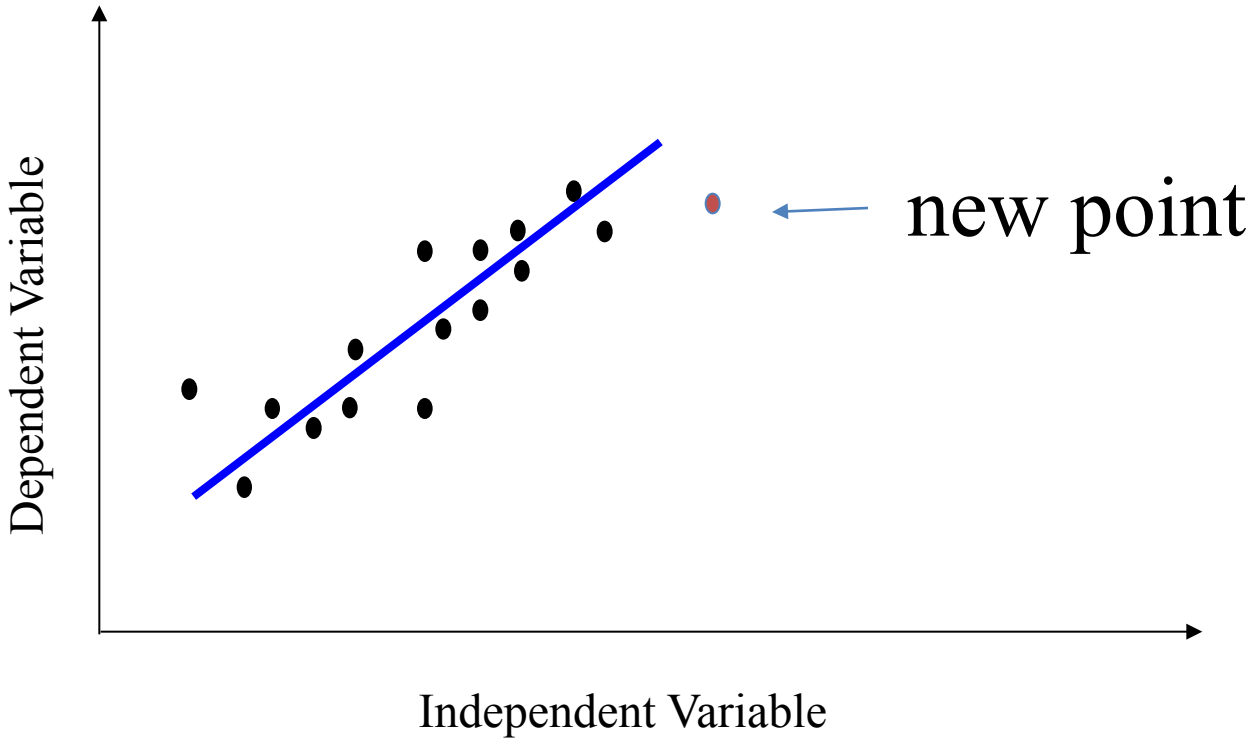
Even more details

- Given:



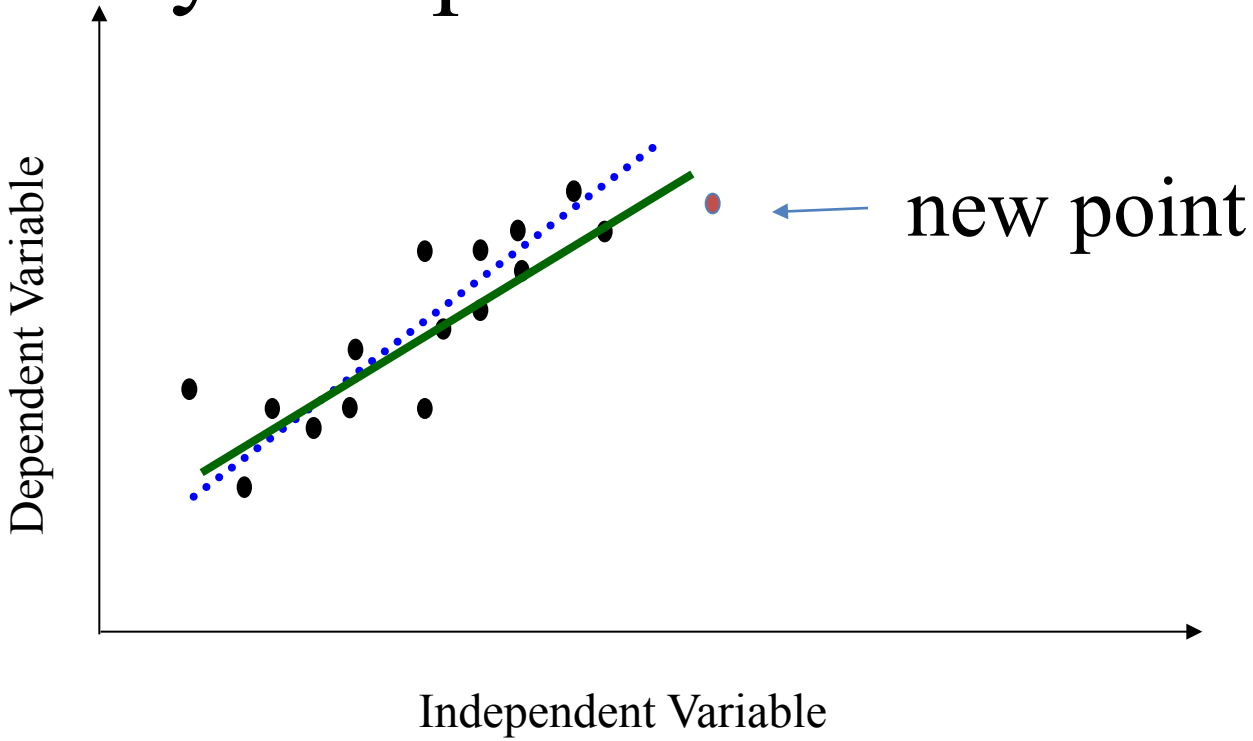
Even more details

- Given:



Even more details

Recursive Least Squares (RLS): quickly compute new best fit



Even more details

- Straightforward Least Squares
  - Needs huge matrix (**growing** in size) $O(N \times w)$
  - Costly matrix operation $O(N \times w^2)$
- Recursive LS
  - Need much smaller, fixed size matrix $O(w \times w)$
  - Fast, incremental computation $O(1 \times w^2)$

49,000,000 ⟷ 49

$N = 10^6, \quad w = 1\text{-}100$

Even more details

- Straightforward Least Squares
  - Needs huge matrix (**growing** in size) $O(N \times$
  - Costly matrix operati $O(N \times w^2)$
- Recursi
  - aller, fixed
  - $\times w)$
  - Fast, incremental computation $O(1 \times w^2)$

49,00 → 49

$N = 10^6, \quad w = 1\text{-}100$

RLS: GREAT for streams

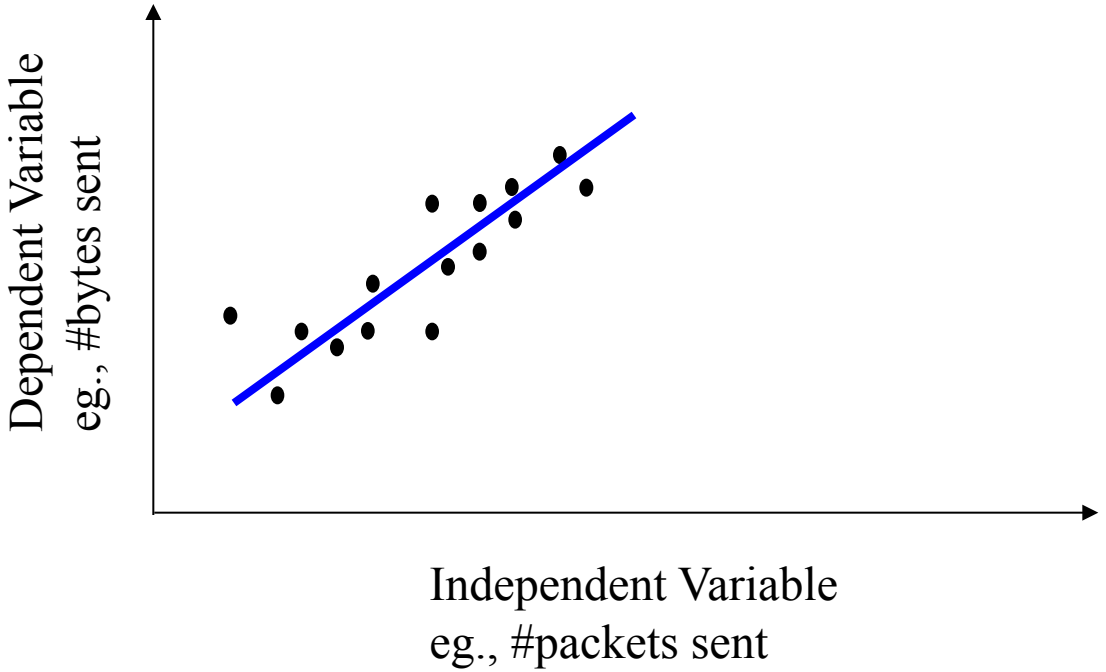
Even more detail

DETAILS

- Q4: can we ‘forget’ the older samples?
- A4: Yes - RLS can easily handle that [Yi+00]:

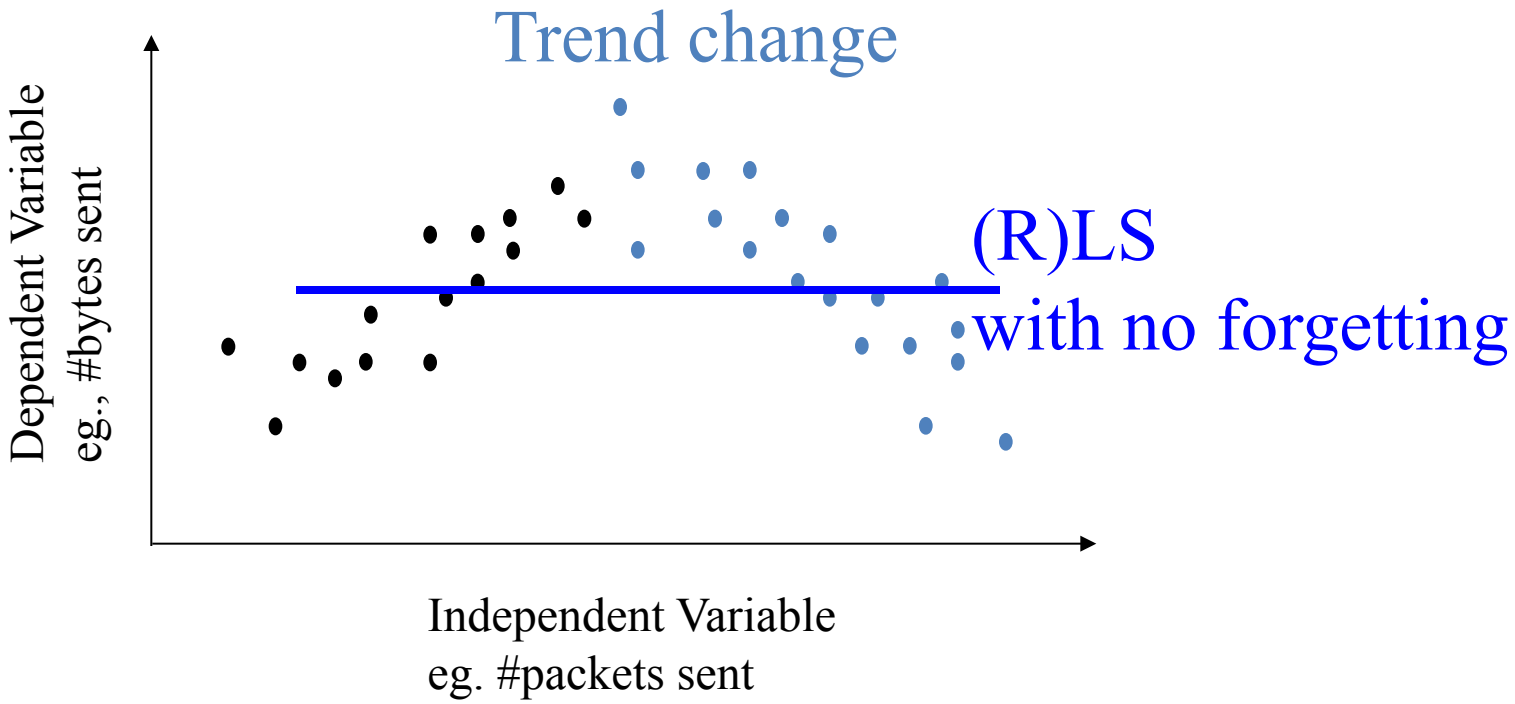
Adaptability - 'forgetting'

DETAILS



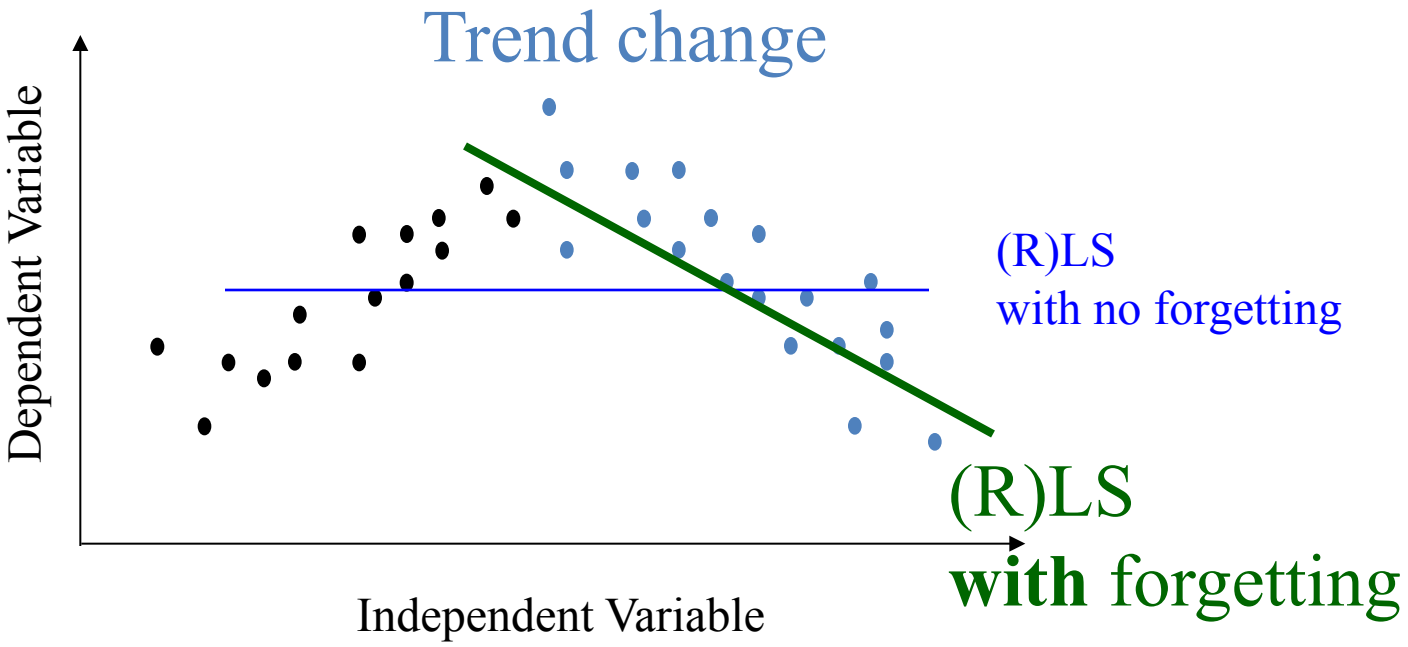
Adaptability - 'forgetting'

DETAILS



Adaptability - 'forgetting'

DETAILS



- RLS: can *trivially* handle 'forgetting'

How to choose ‘*w*’?

- Quick & dirty answer: w=1 or w=2
- Better answer: Model selection (say, with BIC or MDL – see later)
- Even better answer: **multi-scale windows** [Papadimitriou+, vldb2003]

Spiros Papadimitriou, Anthony Brockwell and Christos Faloutsos *Adaptive, Hands-Off Stream Mining* VLDB 2003, Berlin, Germany, Sept. 2003

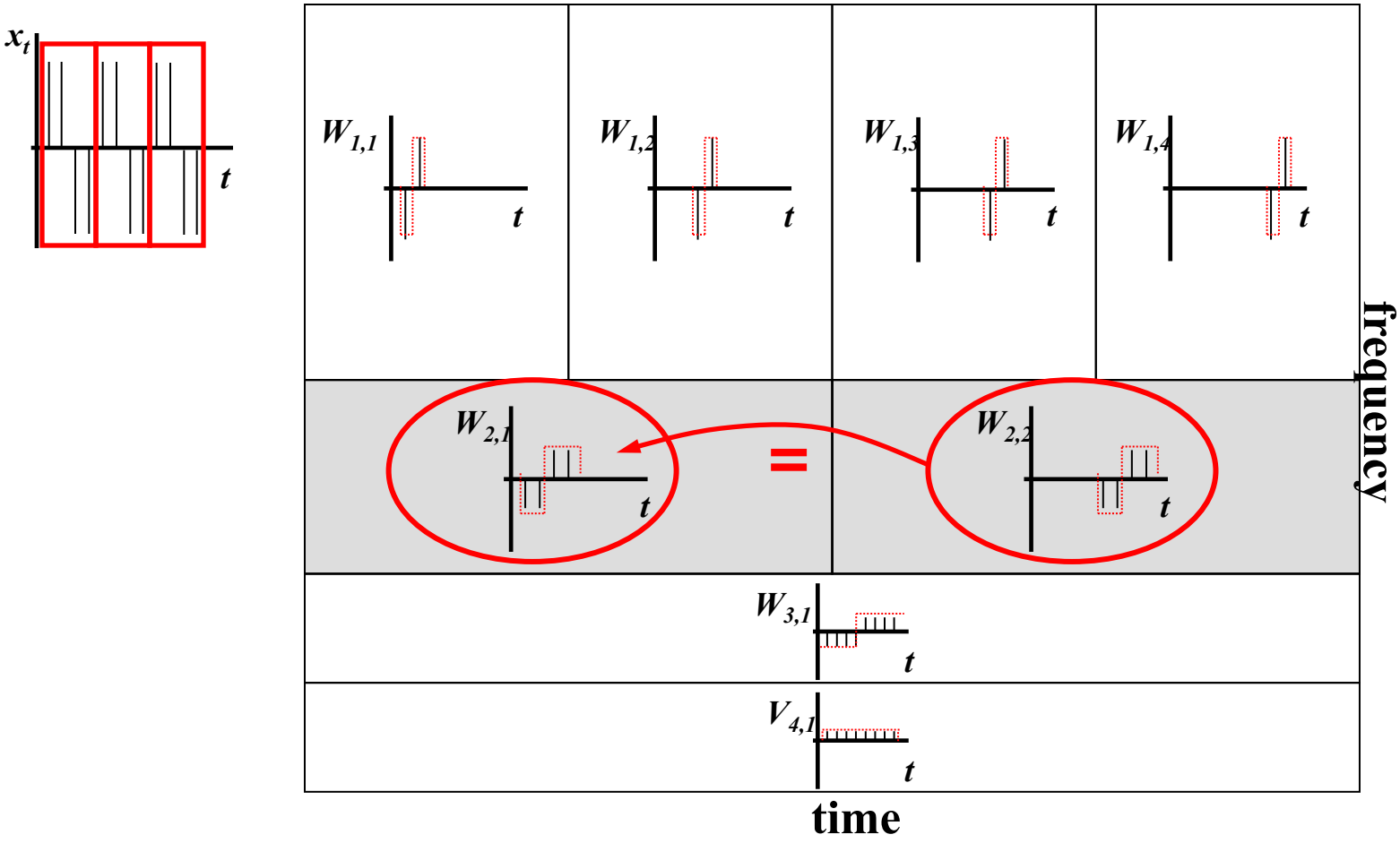
How to choose ‘*w*’ ?

- goal: capture arbitrary periodicities
- with NO human intervention
- on a semi-infinite stream

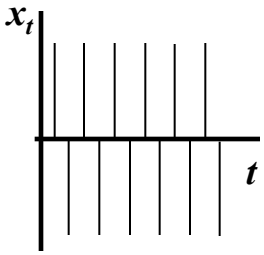
Answer:

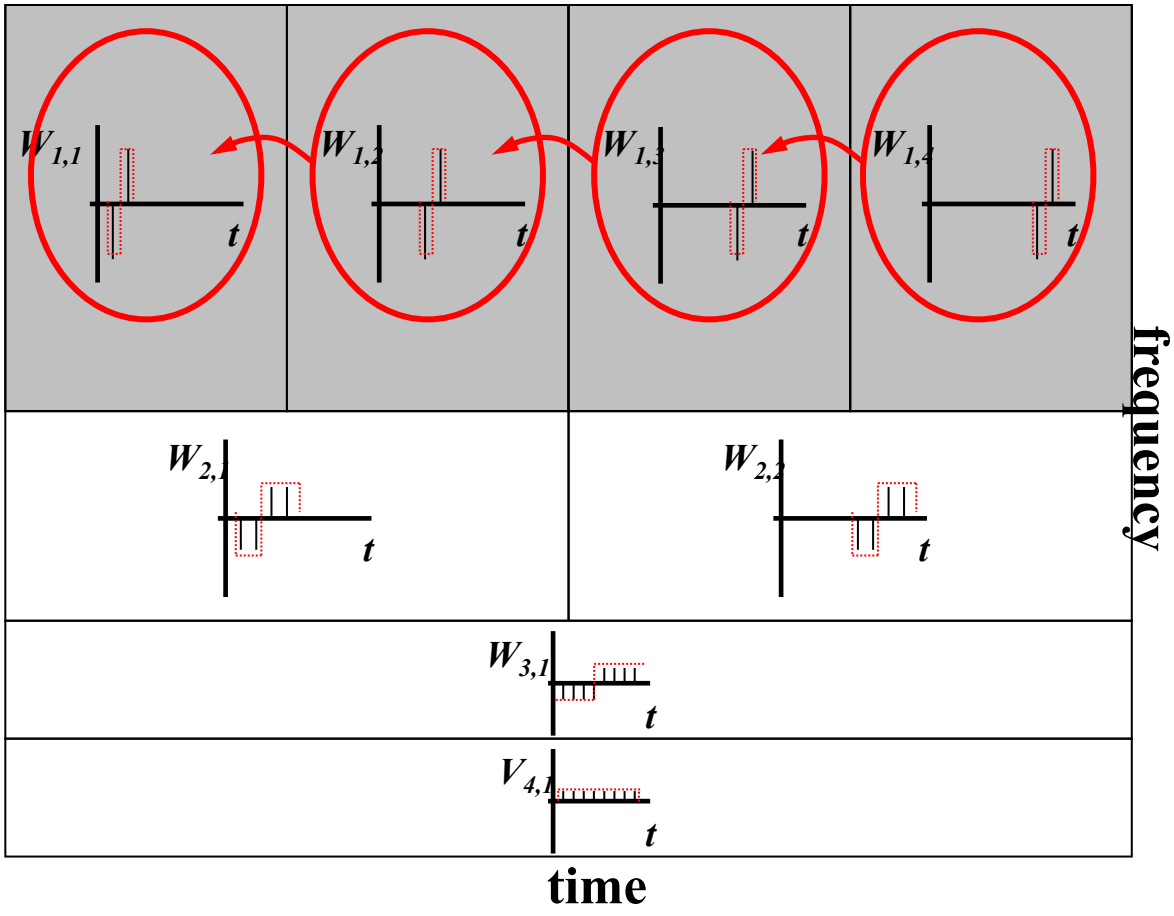
- ‘AWSOM’ (Arbitrary Window Stream fOrecasting Method) [Papadimitriou+, vldb2003]
- idea: do AR on each wavelet level
- in detail:

AWSOM

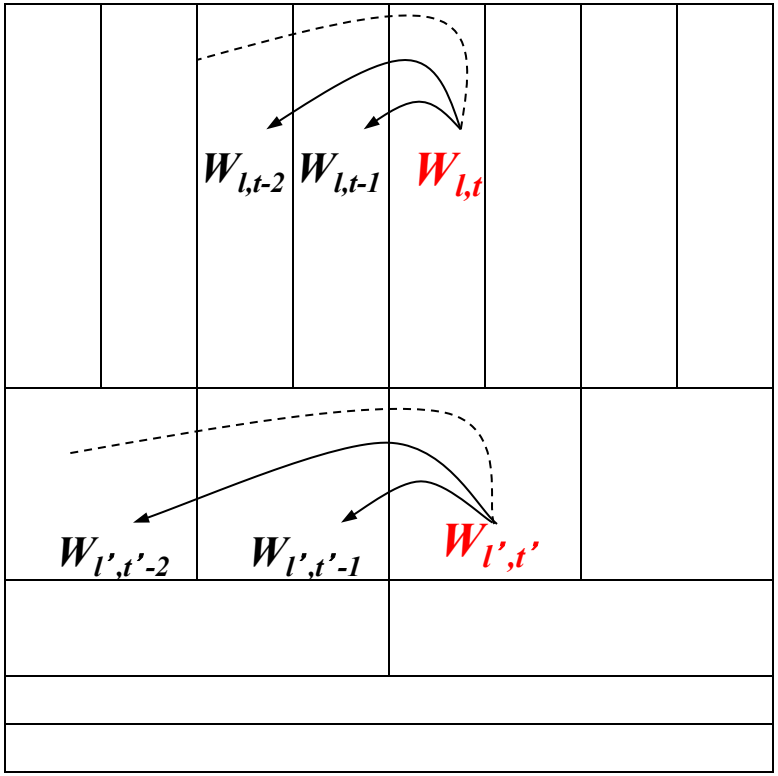


AWSOM





AWSOM - idea



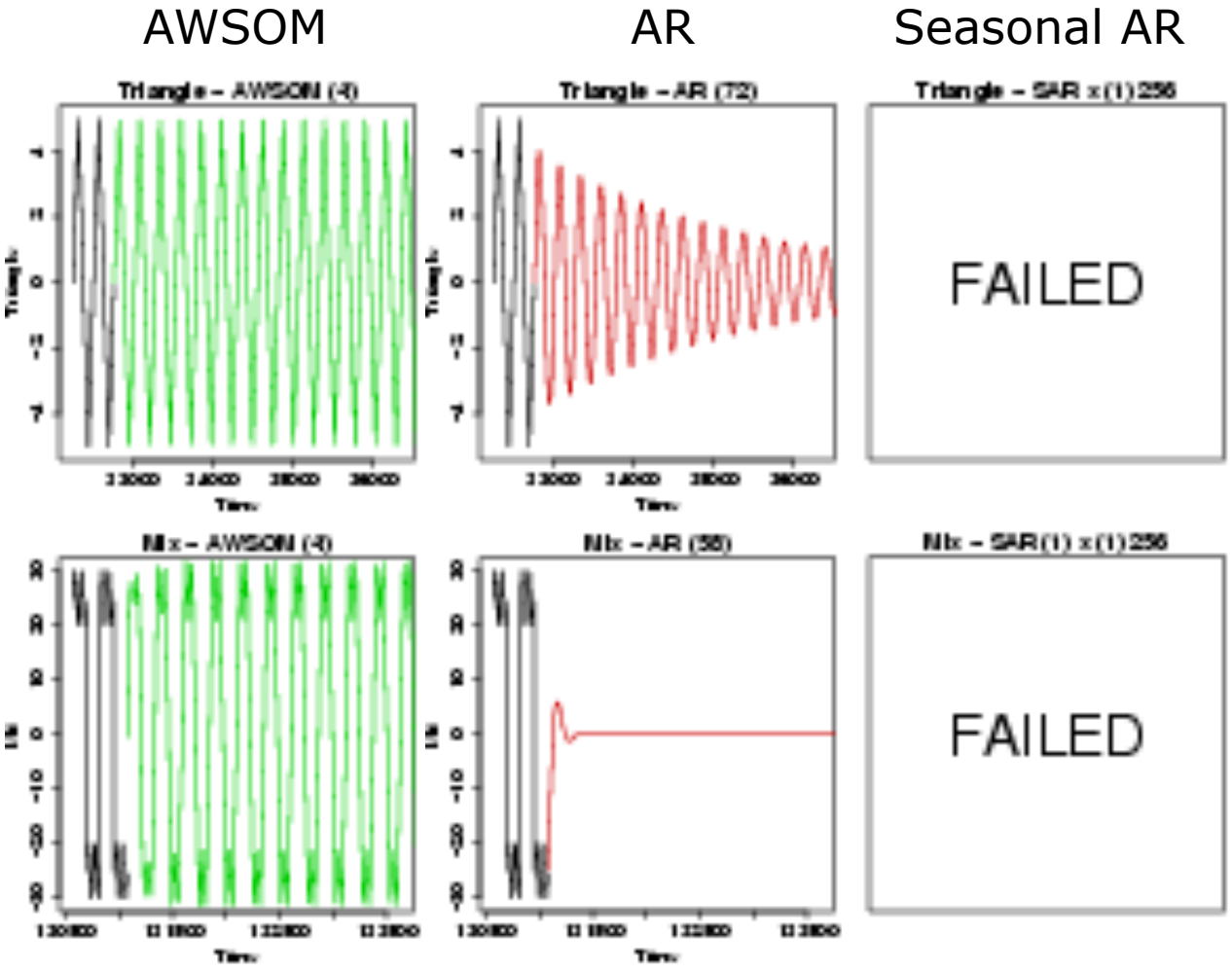
$$W_{l,t} = \beta_{l,1} W_{l,t-1} + \beta_{l,2} W_{l,t-2} + \ldots$$

$$W_{l',t'} = \beta_{l',1} W_{l',t'-1} + \beta_{l',2} W_{l',t'-2} + \ldots$$

More details…

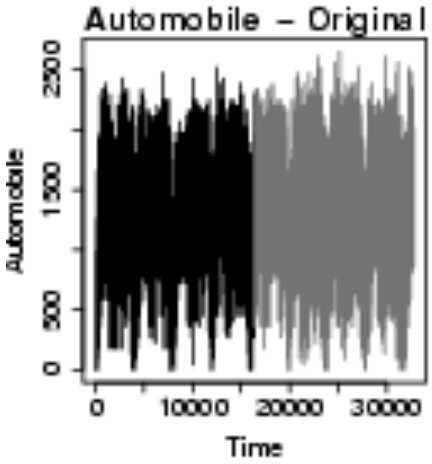
- Update of wavelet coefficients (incremental)
- Update of linear models (incremental; RLS)
- Feature selection (single-pass)
  - Not all correlations are significant
  - Throw away the insignificant ones ("noise")

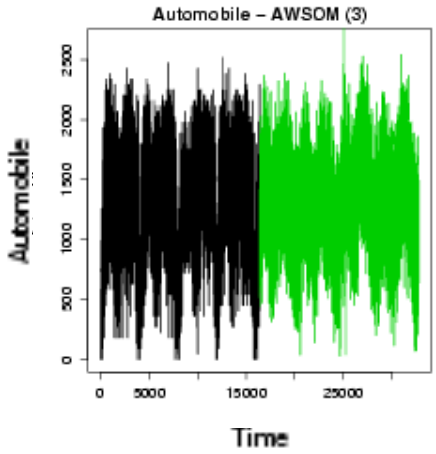
Results - Synthetic data

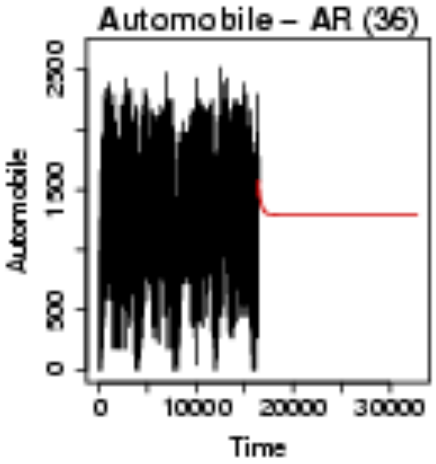


- Triangle pulse
- Mix (sine + square)
- AR captures wrong trend (or none)
- Seasonal AR estimation fails

Results - Real data

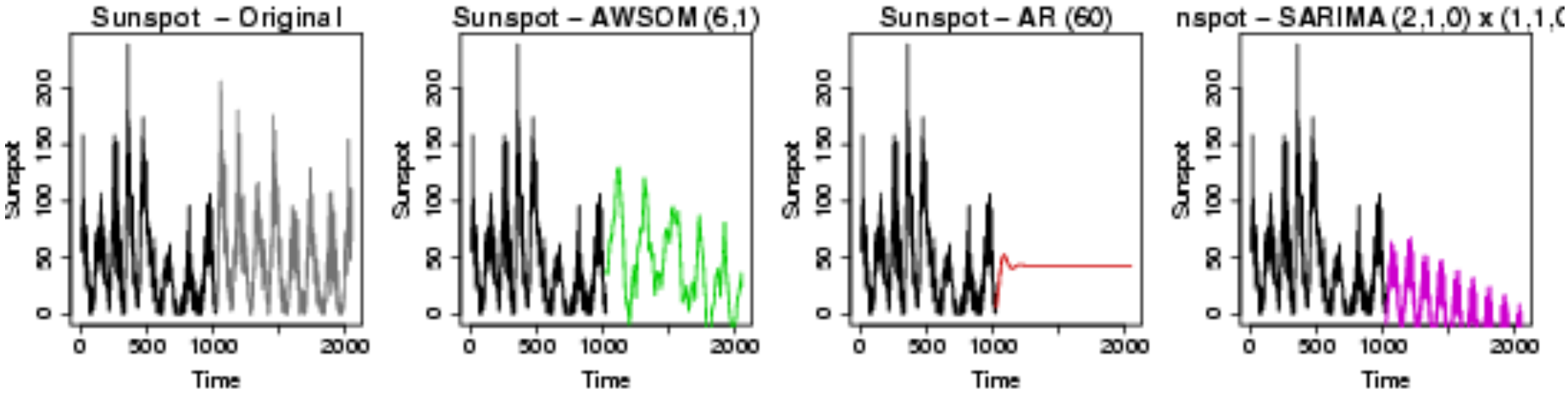






- Automobile traffic
  - Daily periodicity
  - Bursty "noise" at smaller scales
- AR fails to capture any trend
- Seasonal AR estimation fails

Results - real data



- Sunspot intensity
  - Slightly time-varying "period"
- AR captures wrong trend
- Seasonal ARIMA
  - wrong downward trend, despite help by human!

Complexity

- Model update

  Space: $O(lgN + mk^2) \approx O(lgN)$

  Time: $O(k^2) \approx O(1)$

- Where
  - N: number of points (so far)
  - k: number of regression coefficients; fixed
  - m: number of linear models; $O(lgN)$

Roadmap

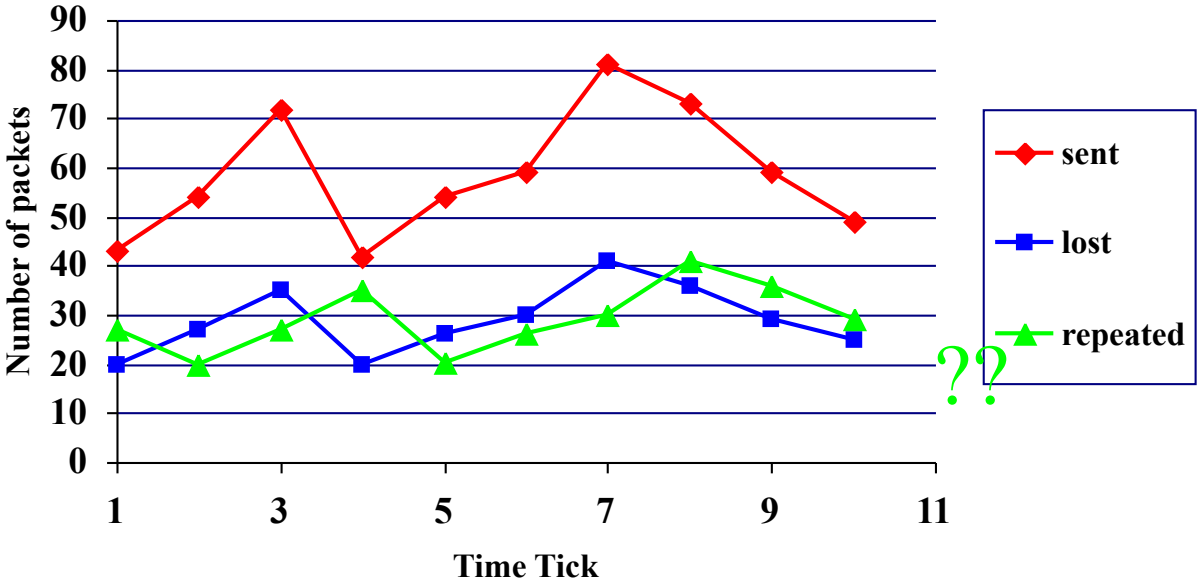
- Motivation
- Similarity Search and Indexing
- Feature extraction
- Streaming pattern discovery
- Linear forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
- Automatic mining

Roadmap

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
- Streaming pattern discovery
- Automatic mining

Co-Evolving Time Sequences

- Given: A set of **correlated** time sequences
- Forecast '**Repeated(t)**'



Solution:

Q: what should we do?

A: Least Squares, with

- Dep. Variable: Repeated(t)
- Indep. Variables:

  Sent(t-1), …, Sent(t-w);

  Lost(t-1), …, Lost(t-w);

  Repeated(t-1), ...
- (named: ‘MUSCLES’ [Yi+00])

Practitioner's guide

- AR(IMA) methodology: prevailing method for linear forecasting
- Brilliant method of Recursive Least Squares for fast, incremental estimation.
- See [Box-Jenkins]

Resources: software and urls

- MUSCLES: Prof. Byoung-Kee Yi:
  http://www.postech.ac.kr/~bkyi/
  or **christos@cs.cmu.edu**
- free-ware: 'R' for stat. analysis
  (clone of Splus)
  http://cran.r-project.org/

Books

- George E.P. Box and Gwilym M. Jenkins and Gregory C. Reinsel, *Time Series Analysis: Forecasting and Control*, Prentice Hall, 1994 (the classic book on ARIMA, 3rd ed.)
- Brockwell, P. J. and R. A. Davis (1987). Time Series: Theory and Methods. New York, Springer Verlag.

Additional Reading

- [Papadimitriou+ vldb2003] Spiros Papadimitriou, Anthony Brockwell and Christos Faloutsos *Adaptive, Hands-Off Stream Mining* VLDB 2003, Berlin, Germany, Sept. 2003
- [Yi+00] Byoung-Kee Yi et al.: *Online Data Mining for Co-Evolving Time Sequences*, ICDE 2000. (Describes MUSCLES and Recursive Least Squares)

Outline

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Stream mining

- Applications
  - Sensor monitoring
  - Network analysis
  - Financial and/or business transaction data
  - Web access and media service logs
  - Moving object tracking
  - Industrial manufacturing

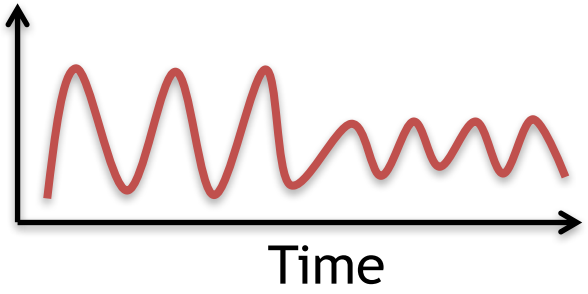
Stream mining

- Requirements
  - **–Fast**

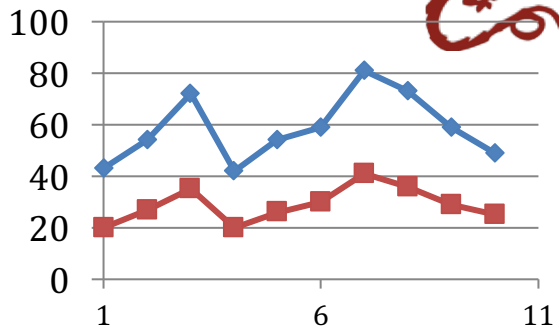
    high performance and quick response
  - **–Nimble**

    low memory consumption, single scan
  - **–Accurate**

    good approximation for pattern discovery and feature extraction



Monitoring data streams



- Correlation coefficient

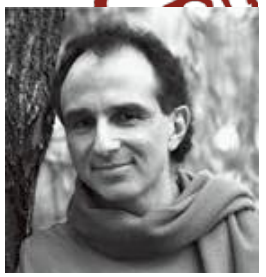
$$\rho = \frac{\sum_{t=1}^{n}(x_t - \bar{x})\cdot(y_t - \bar{y})}{\sigma(x)\cdot\sigma(y)}$$

$$\sigma(x) = \sqrt{\sum_{t=1}^{n}(x_t - \bar{x})^2}$$

- Correlation coefficient and the (Euclidean) distance

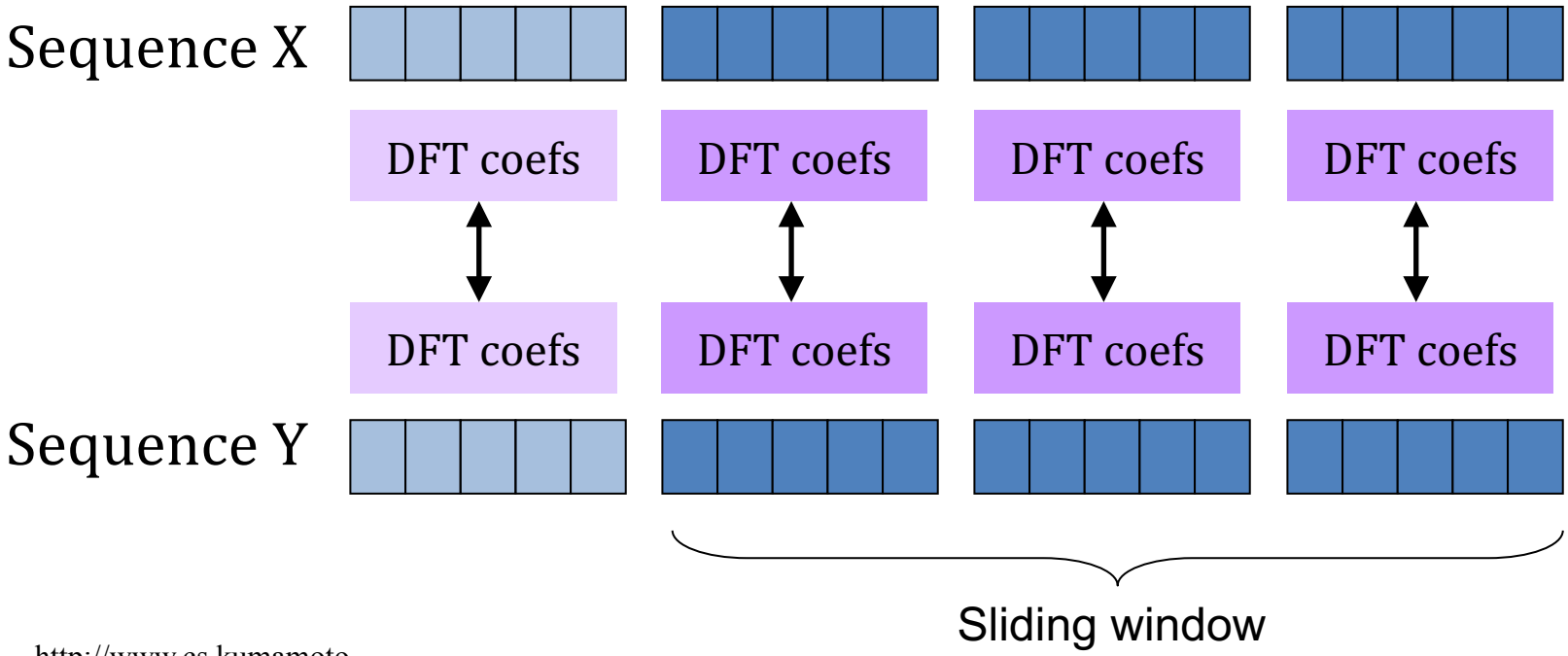
$$\rho = 1 - \frac{1}{2}\sum_{t=1}^{n}(\hat{x}_t - \hat{y}_t)^2 \qquad \hat{x}_t = (x_t - \bar{x})/\sigma(x)$$

Monitoring data streams



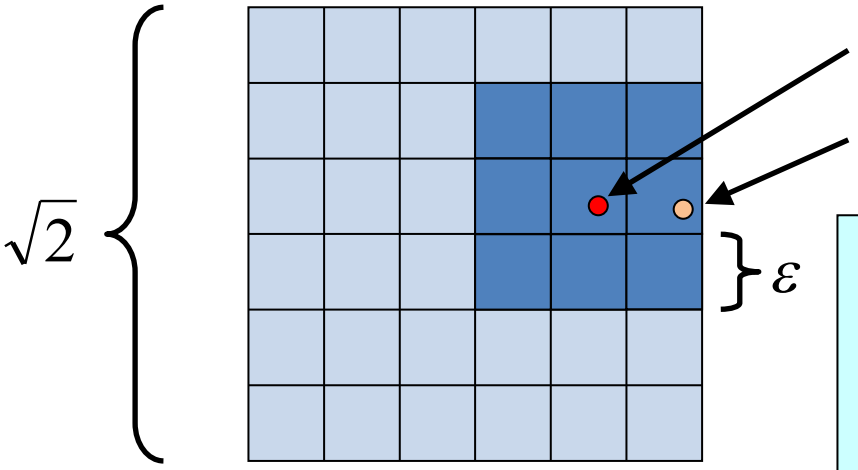
Dennis Shasha

- Correlation monitoring [Zhu+, vldb02]
  - DFT coefficients for each basic window
  - Correlation coefficient of each sliding window computed from the `sketch' (DFT coefs)



Monitoring data streams

- Grid structure (to avoid checking all pairs)
  - DFT coefficients yields a vector
  - High correlation -> closeness in the vector space

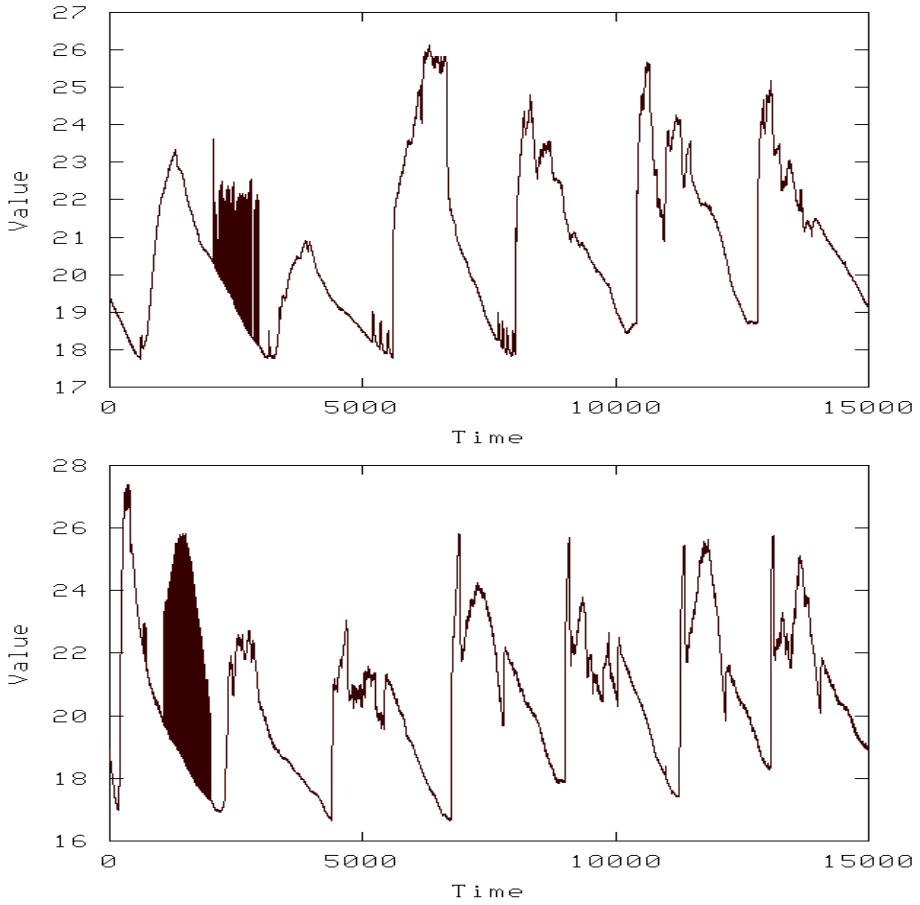


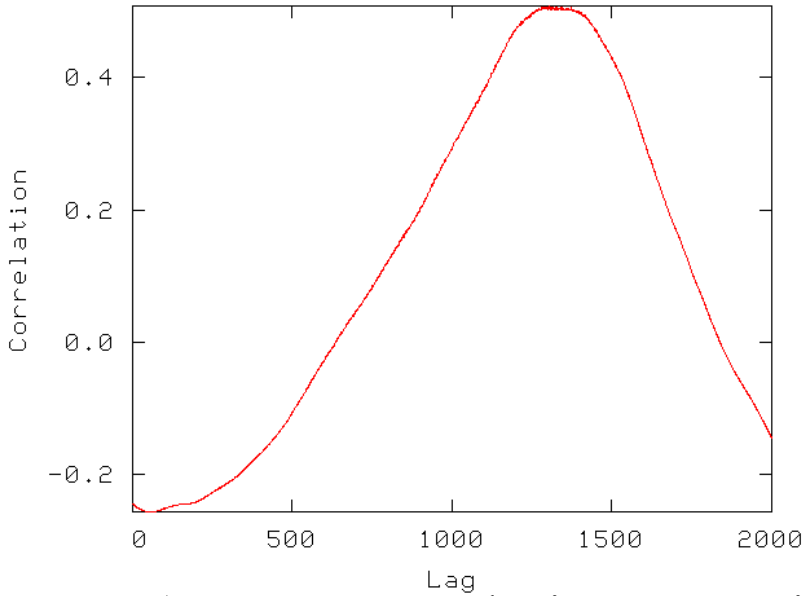
Correlation coefficients and the Euclidean distance

$$\rho = 1 - \frac{1}{2}\sum_{t=1}^{n}(\hat{x}_t - \hat{y}_t)^2$$

Monitoring data streams

- Lag correlation [Sakurai+, sigmod05]

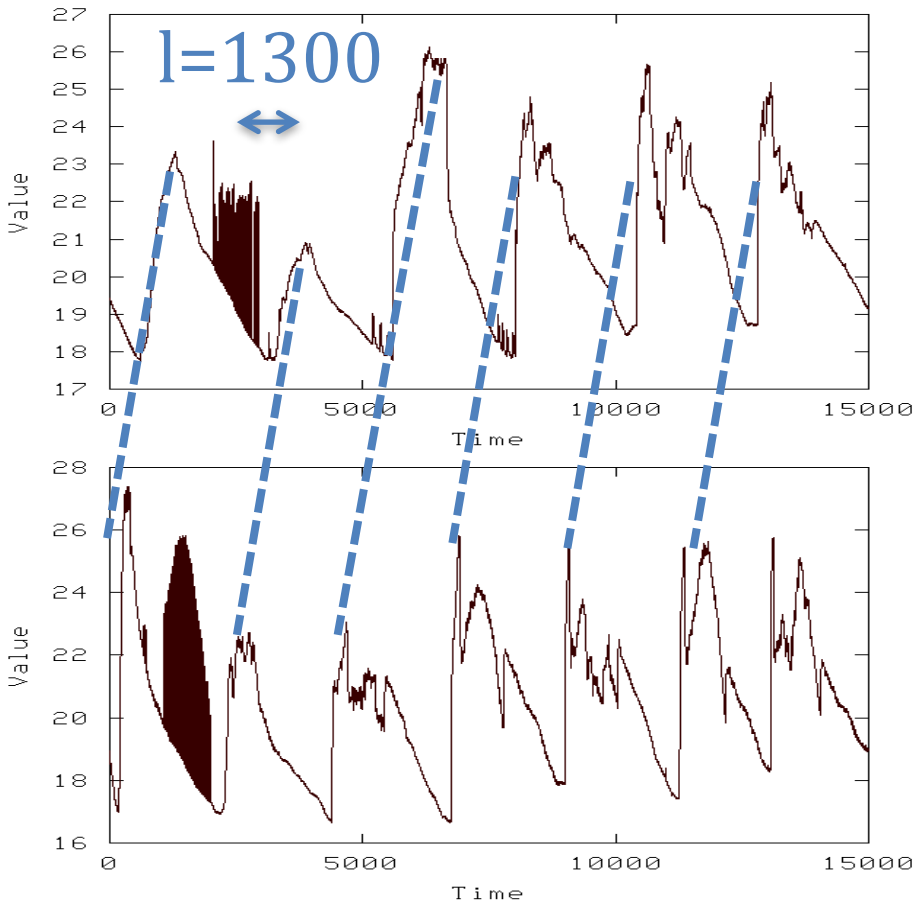




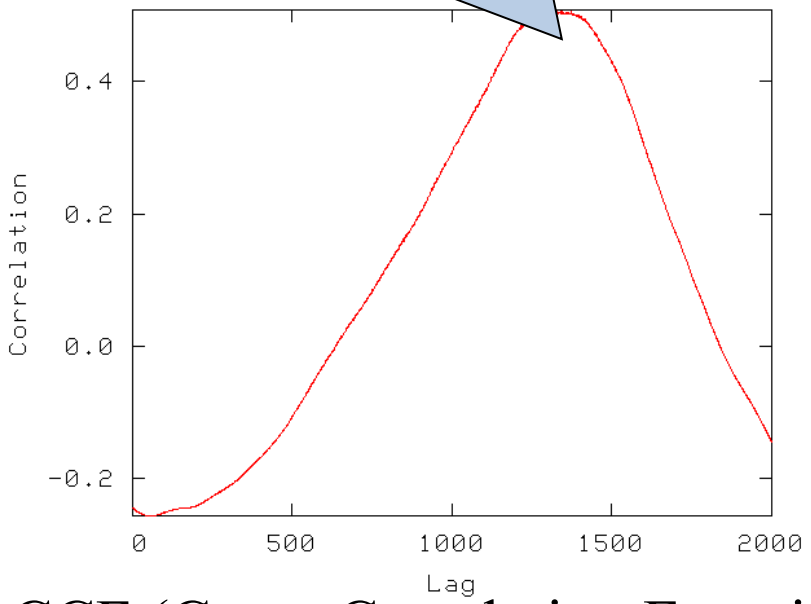
CCF (Cross-Correlation Function)

Monitoring data streams

- Lag correlation [Sakurai+, sigmod05]



correlated with lag *l*=1300



CCF (Cross-Correlation Function)

Lag correlation

- Definition of 'score', absolute value of $R(l)$

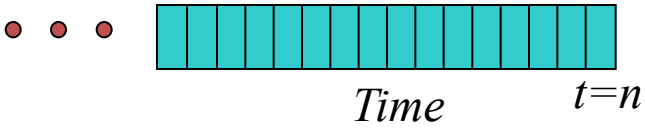
$$score(l) = |R(l)| \qquad R(l) = \frac{\sum_{t=l+1}^{n}(x_t - \bar{x})(y_{t-l} - \bar{y})}{\sqrt{\sum_{t=l+1}^{n}(x_t - \bar{x})^2}\sqrt{\sum_{t=1}^{n-l}(y_t - \bar{y})^2}}$$

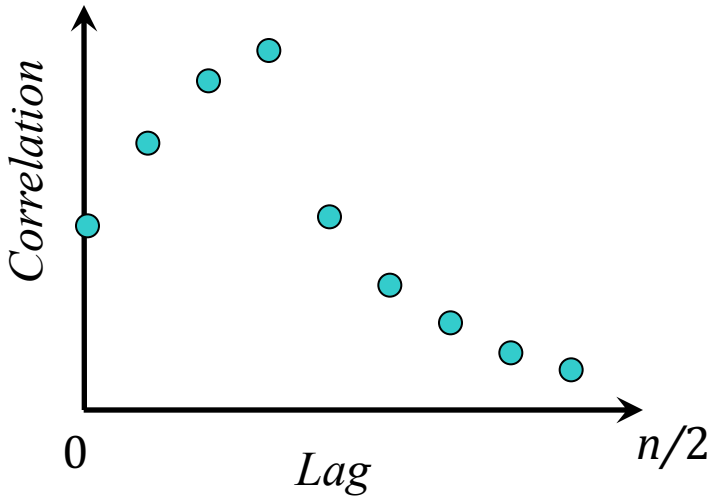
- Lag correlation
  - Given a threshold γ, $score(l) > \gamma$
  - A local maximum
  - The earliest such maximum, if more maxima exist

Lag correlation

- Why not naïve?
  - Compute correlation coefficient for each lag

    $l = \{0, 1, 2, 3, \ldots, n/2\}$
- But
  - $O(n)$ space
  - $O(n^2)$ time
  - or $O(n \log n)$ time w/ FFT

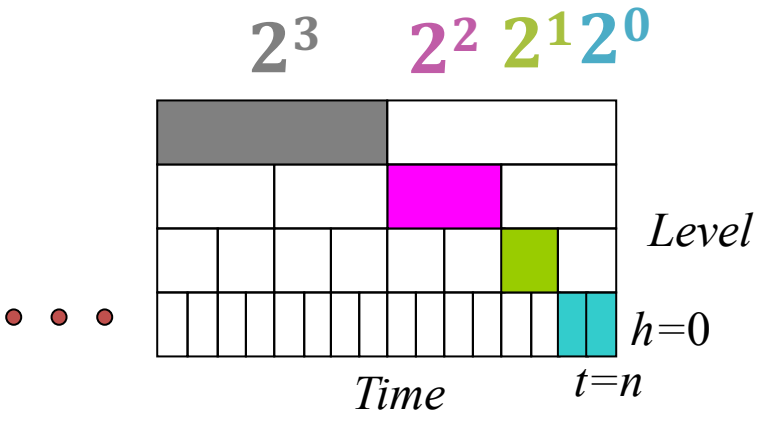


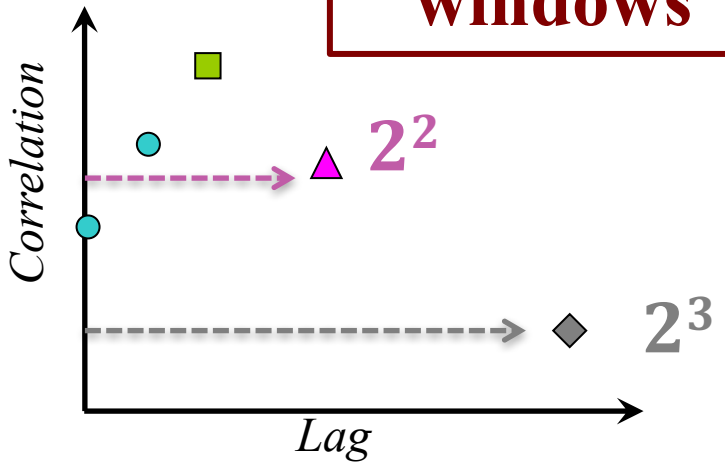


Lag correlation

- BRAID
  - Geometric lag probing + smoothing
  - Use colored windows
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

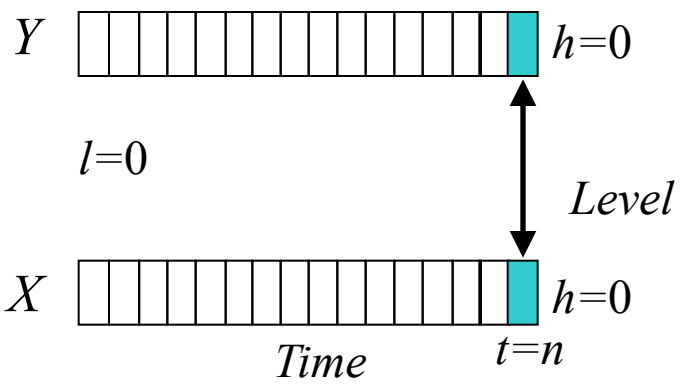
Multi-scale windows

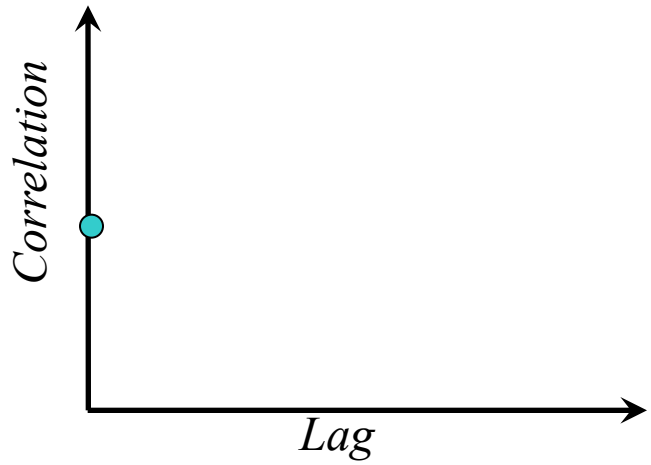




Lag correlation

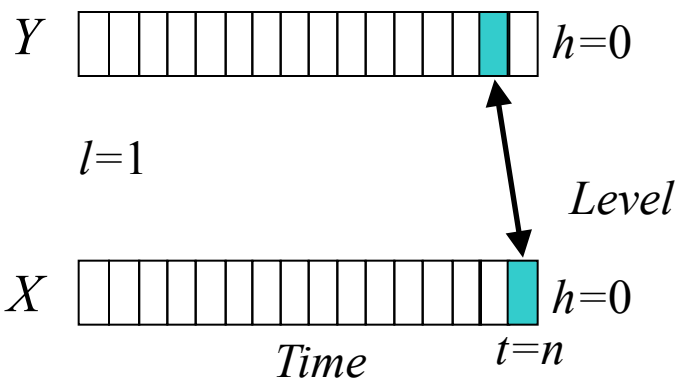
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

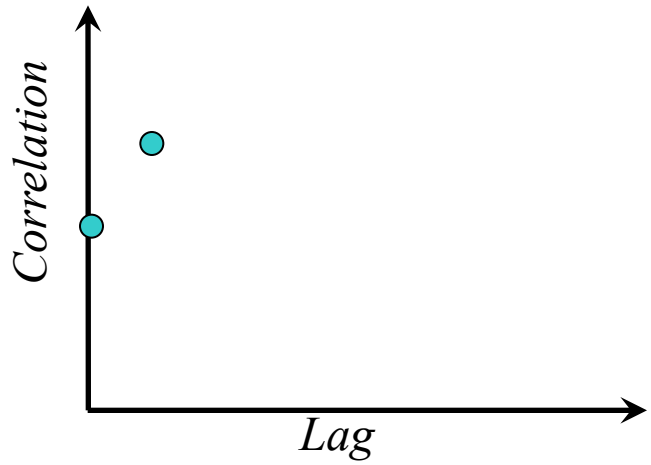




Lag correlation

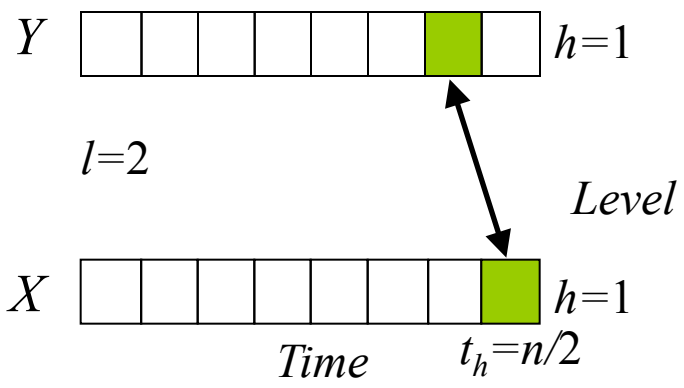
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

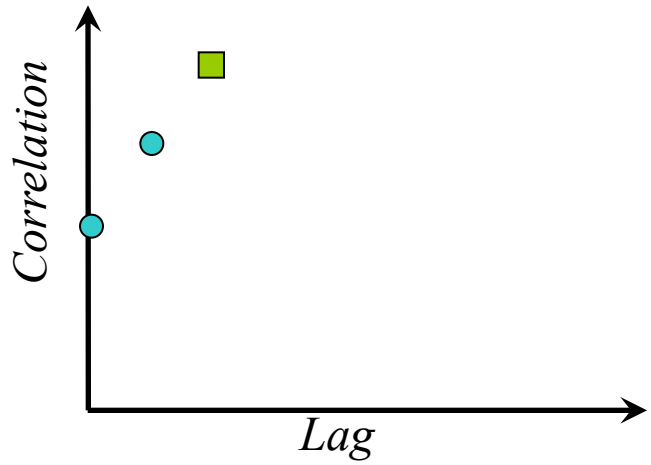




Lag correlation

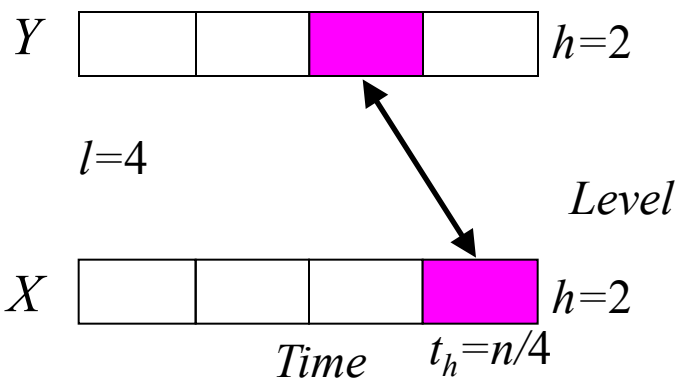
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

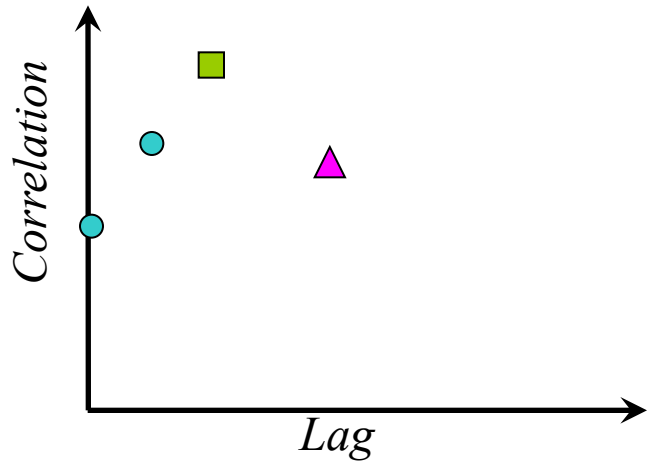




Lag correlation

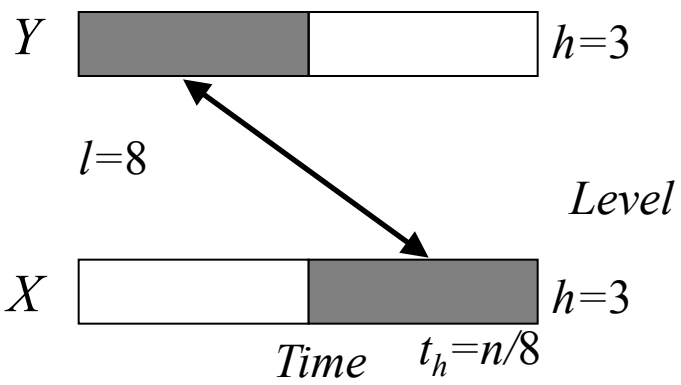
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

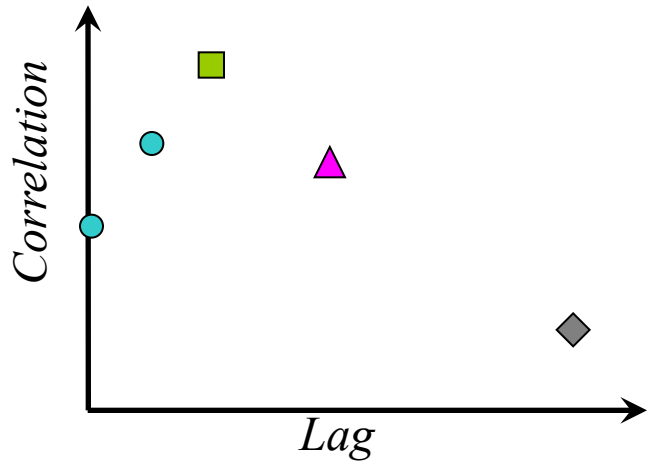




Lag correlation

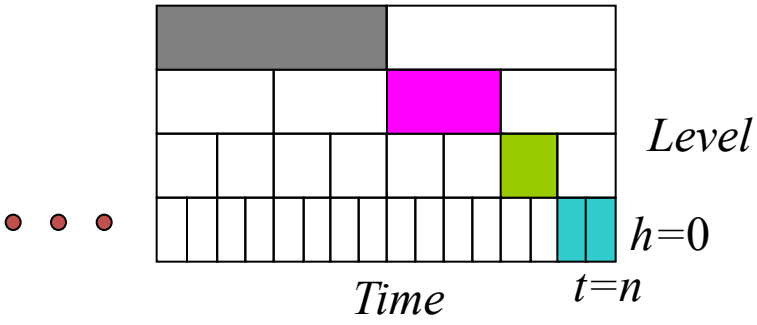
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

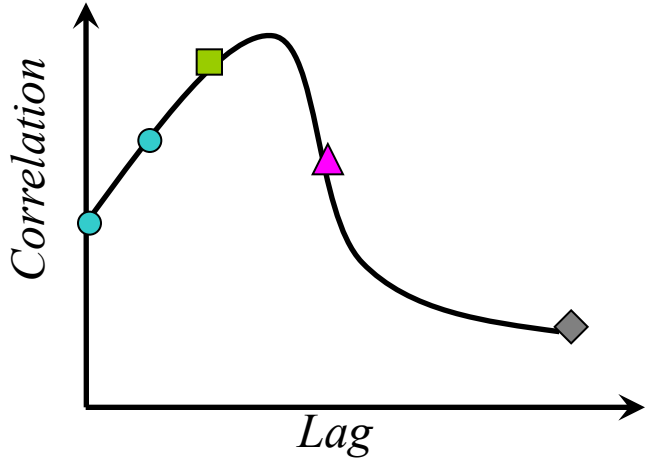




Lag correlation

- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$
  - Use a cubic spline to interpolate

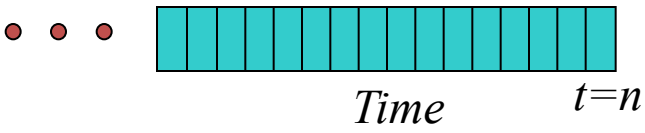




Lag correlation

- Why not naïve?
  - Compute correlation coefficient for each lag

    $l = \{0, 1, 2, 3, \ldots, n/2\}$
- But
  - $O(n)$ space
  - $O(n^2)$ time
  - or $O(n \log n)$ time w/]



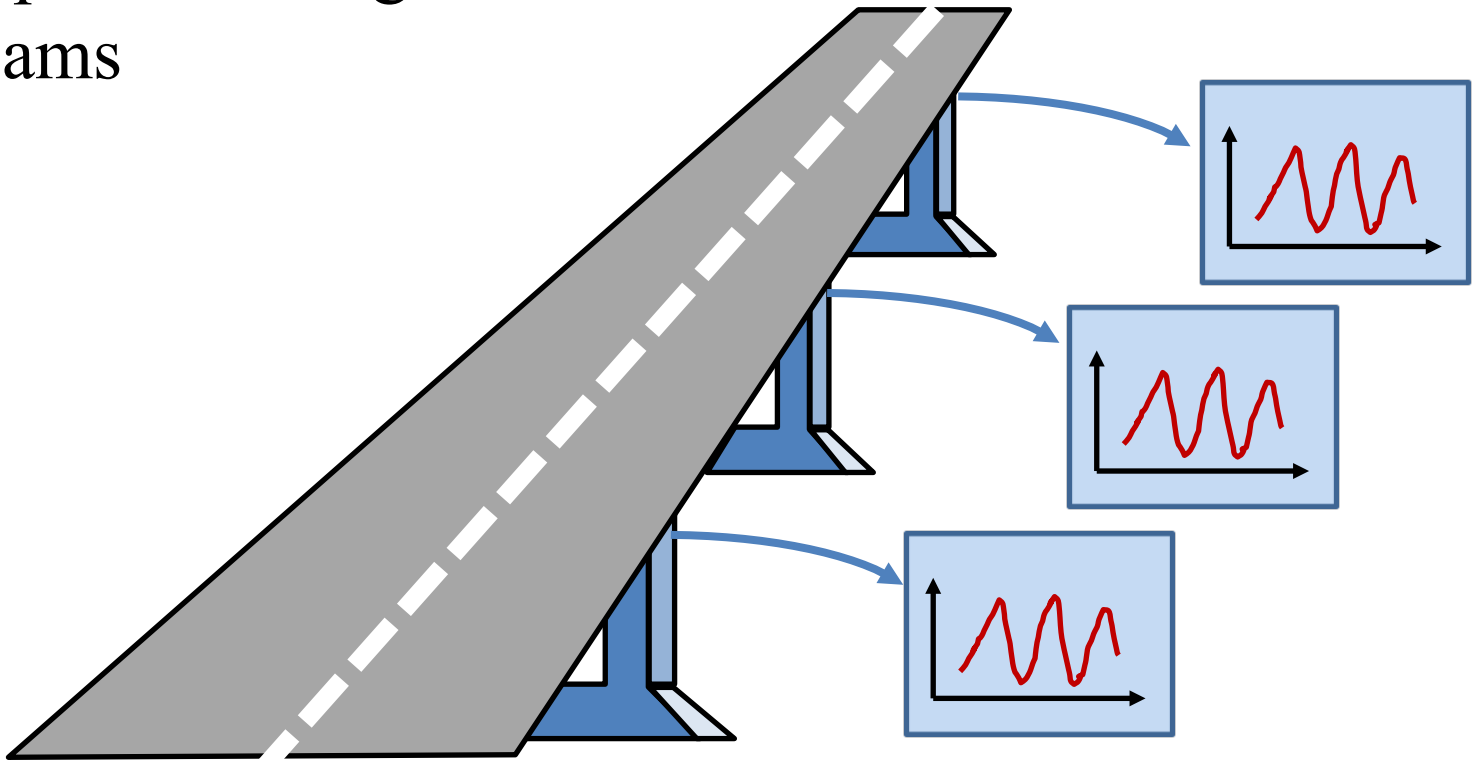
BRAID

- $O(log\ n)$ space
- $O(1)$ time

Multi-scale windows

BRAID in the real world

- Bridge structural health monitoring
  - Structural monitoring using vibration/shock sensors
  - Keep track of lag correlations for sensor data streams



BRAID in the real world

- Bridge structural health monitoring
  - Goal: real-time anomaly detection for disaster prevention
  - Several thousands readings (per sec) from several hundreds sensor nodes

Structural health monitoring

Vibration/shock sensor

- Uses BRAID
- Metropolitan Expressway (Tokyo, Japan)

BRAID in the real world

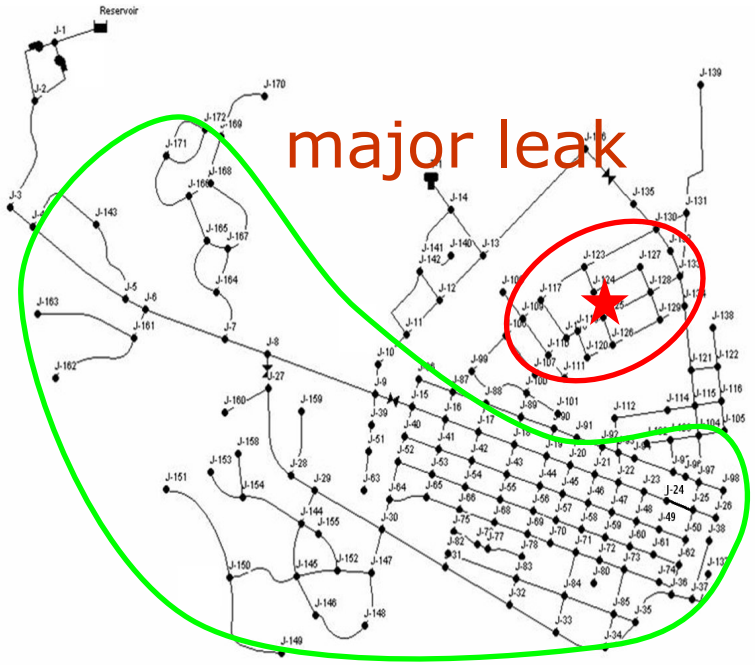
- Bridge structural health monitoring with BRAID

Metropolitan Expressway (Tokyo, Japan)

Tokyo Gate Bridge (Tokyo, Japan)

Can Tho Bridge (Vietnam)

Feature extraction from streams

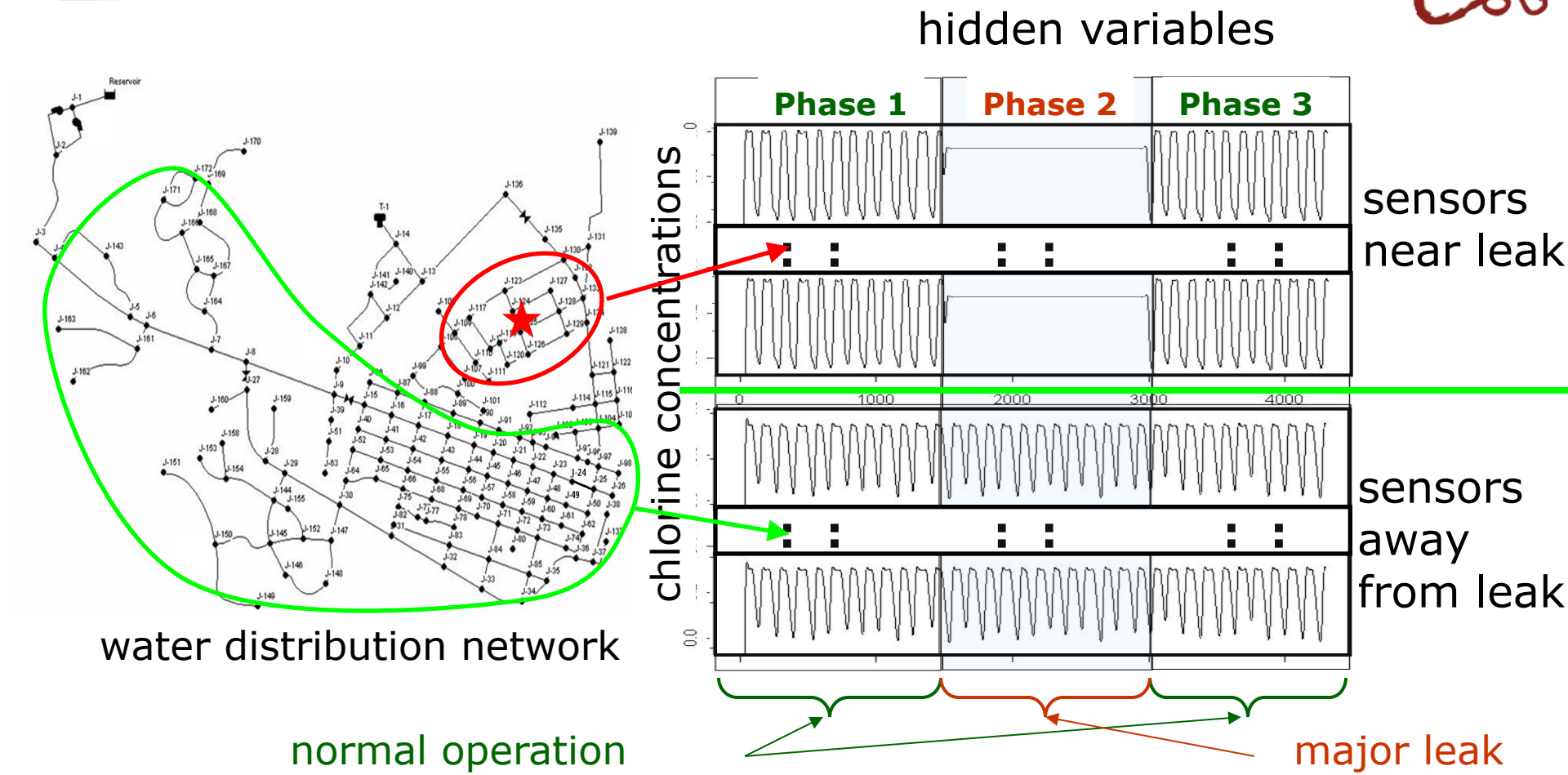


water distribution network

- Find hidden variables from streams [Papadimitriou+, vldb2005]

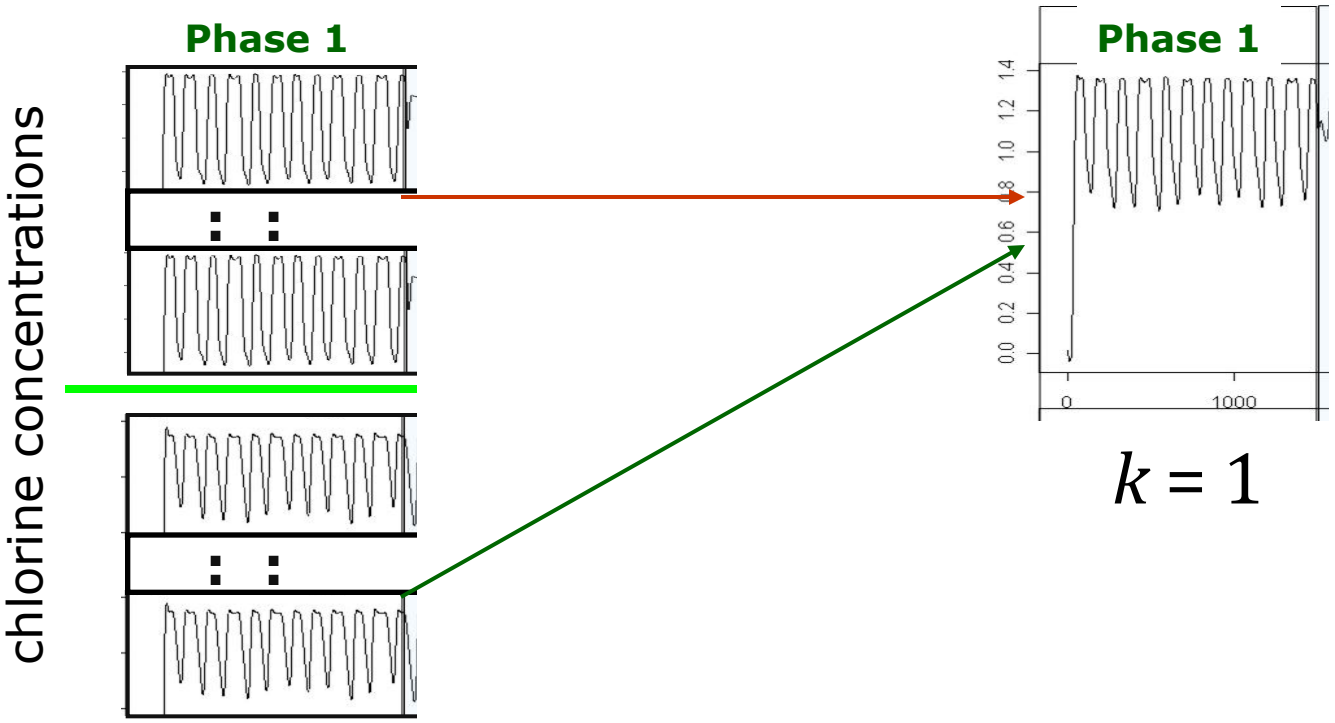
May have hundreds of measurements, but it is **unlikely they are completely unrelated**!

Feature extraction from streams



May have hundreds of measurements, but it is **unlikely they are completely unrelated**!

Motivation

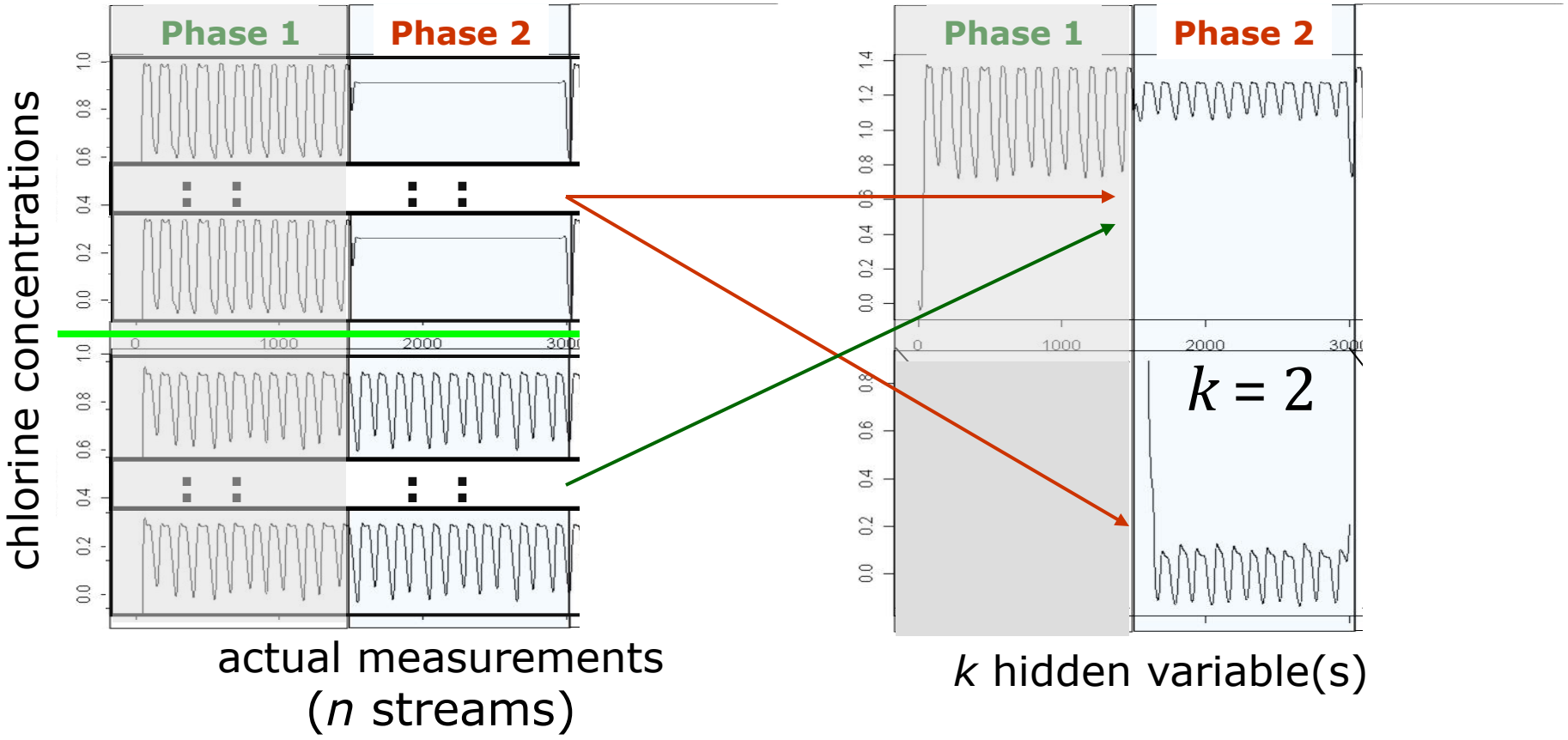


actual measurements
(*n* streams)

k hidden variable(s)

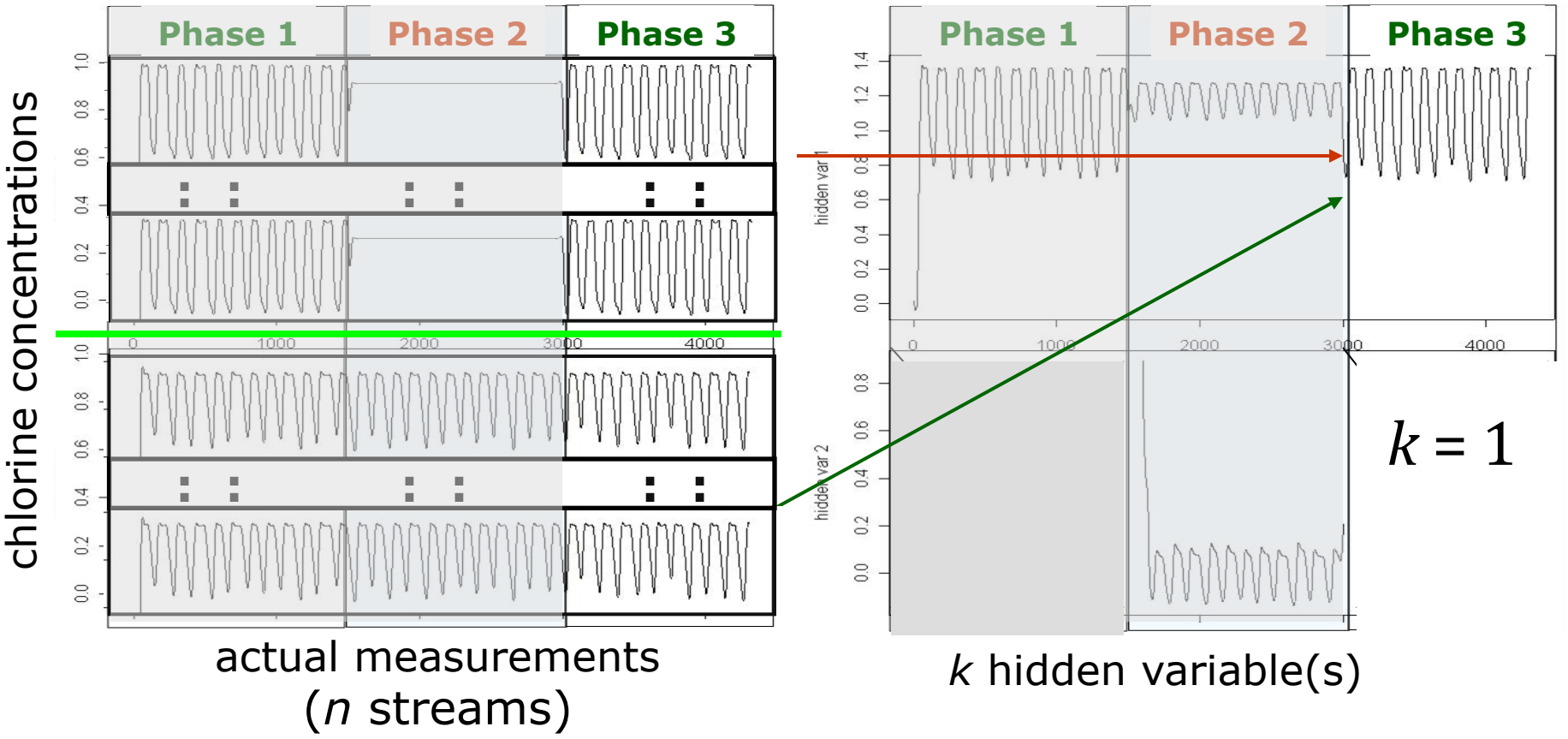
We would like to discover a few "hidden (latent) variables" that summarize the key trends

Motivation



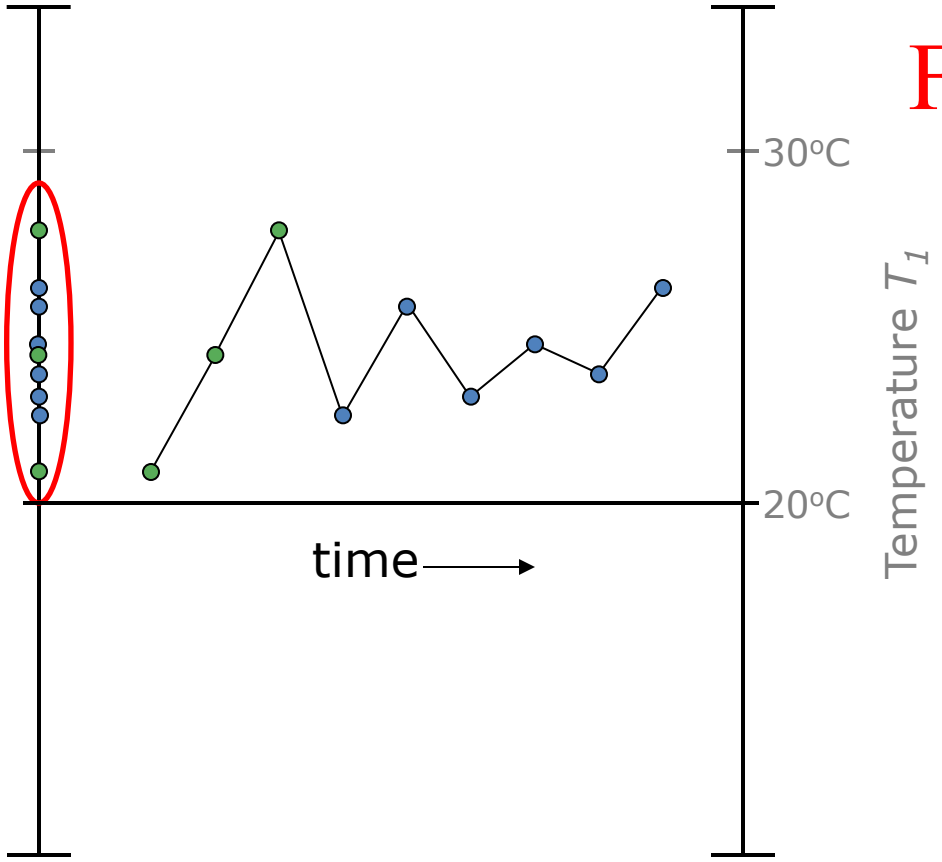
We would like to discover a few "hidden (latent) variables" that summarize the key trends

Motivation



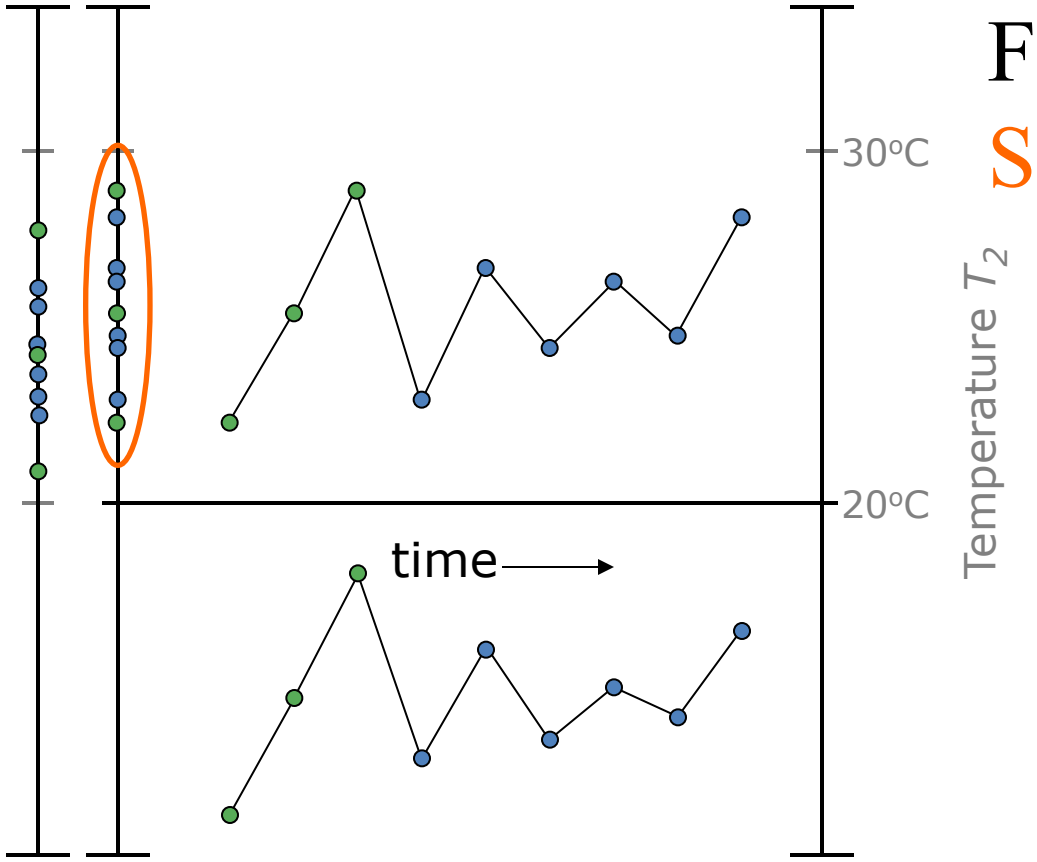
We would like to discover a few "hidden (latent) variables" that summarize the key trends

How to capture correlations?



First sensor

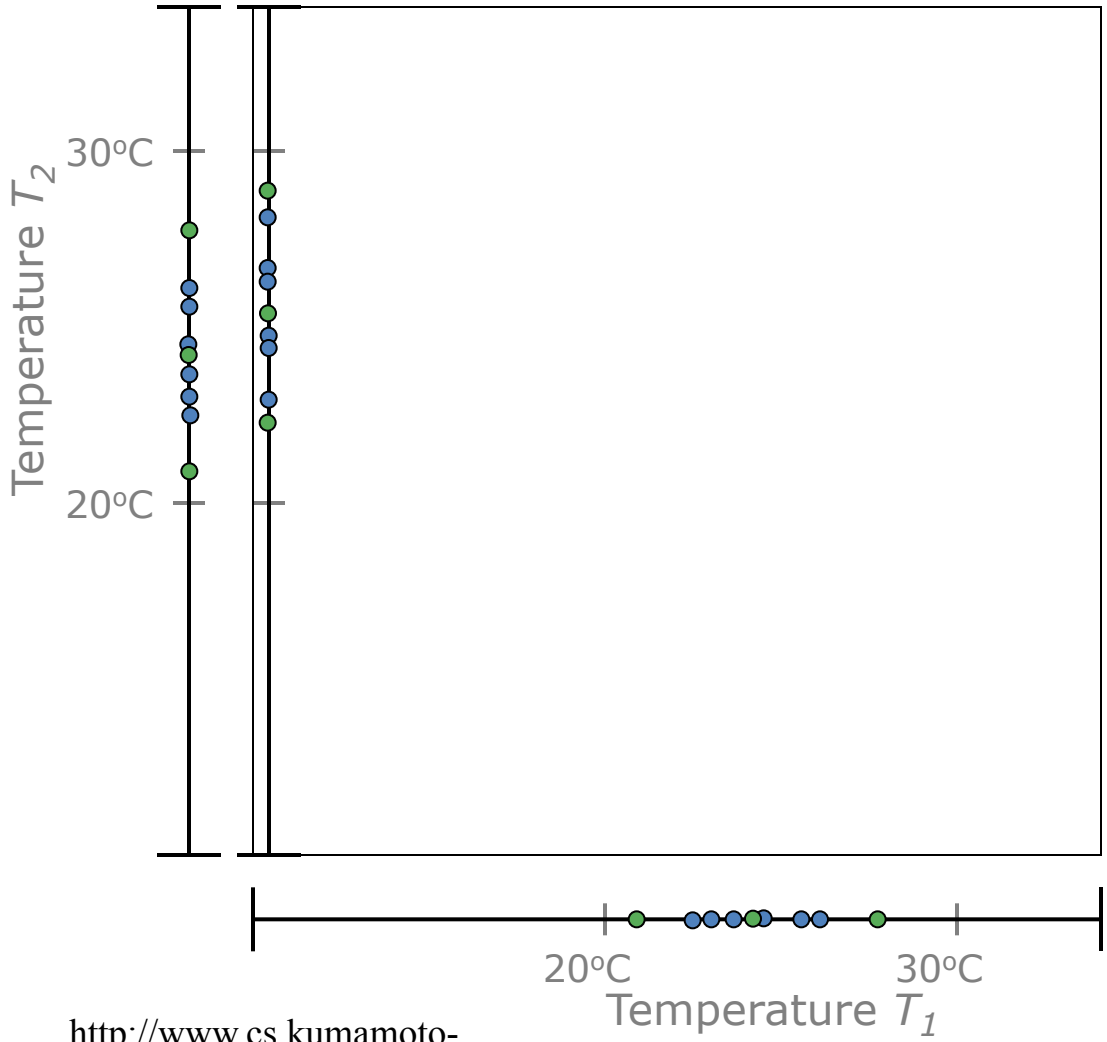
How to capture correlations?



First sensor

Second sensor

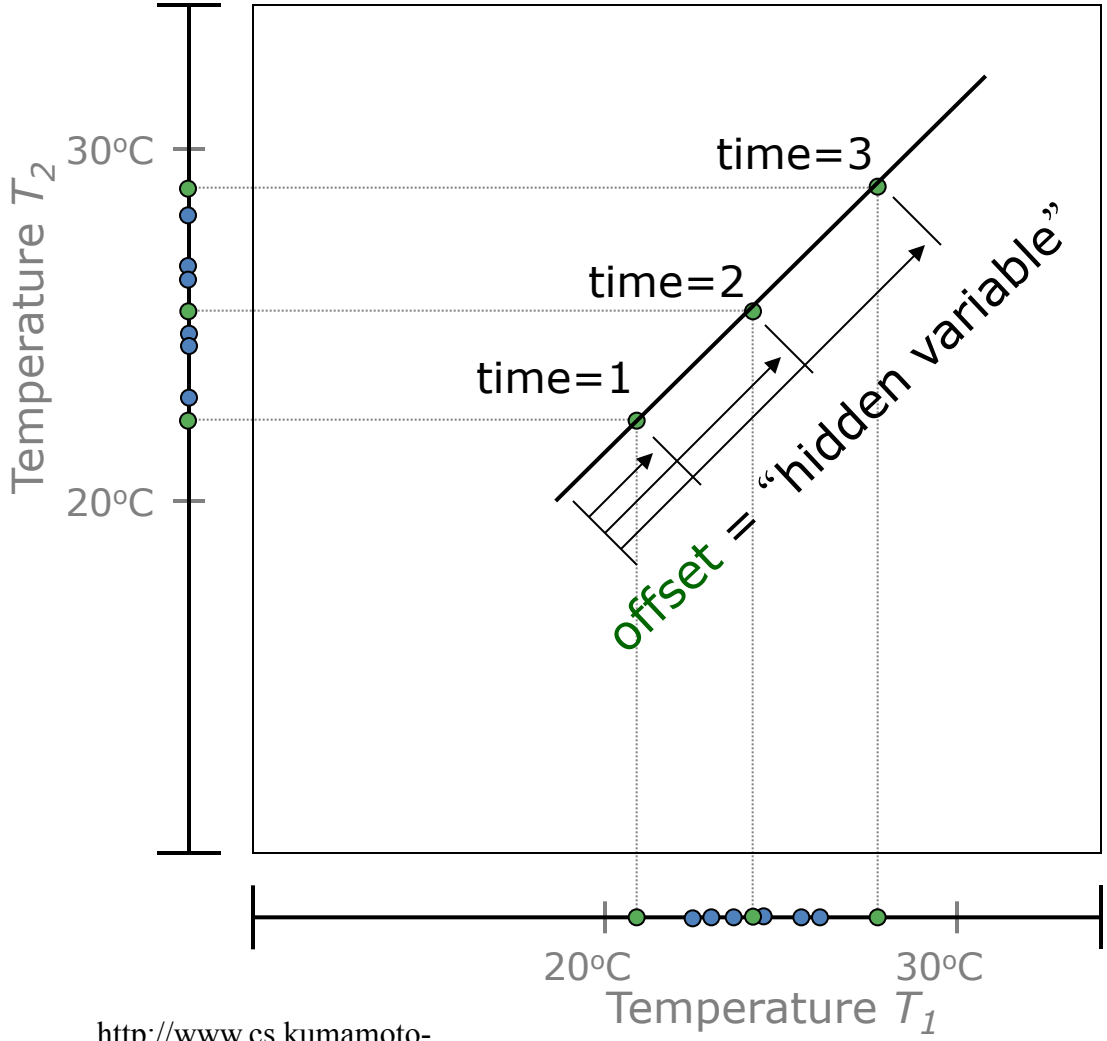
How to capture correlations?



Correlations:

Let's take a closer look at the first three value-pairs…

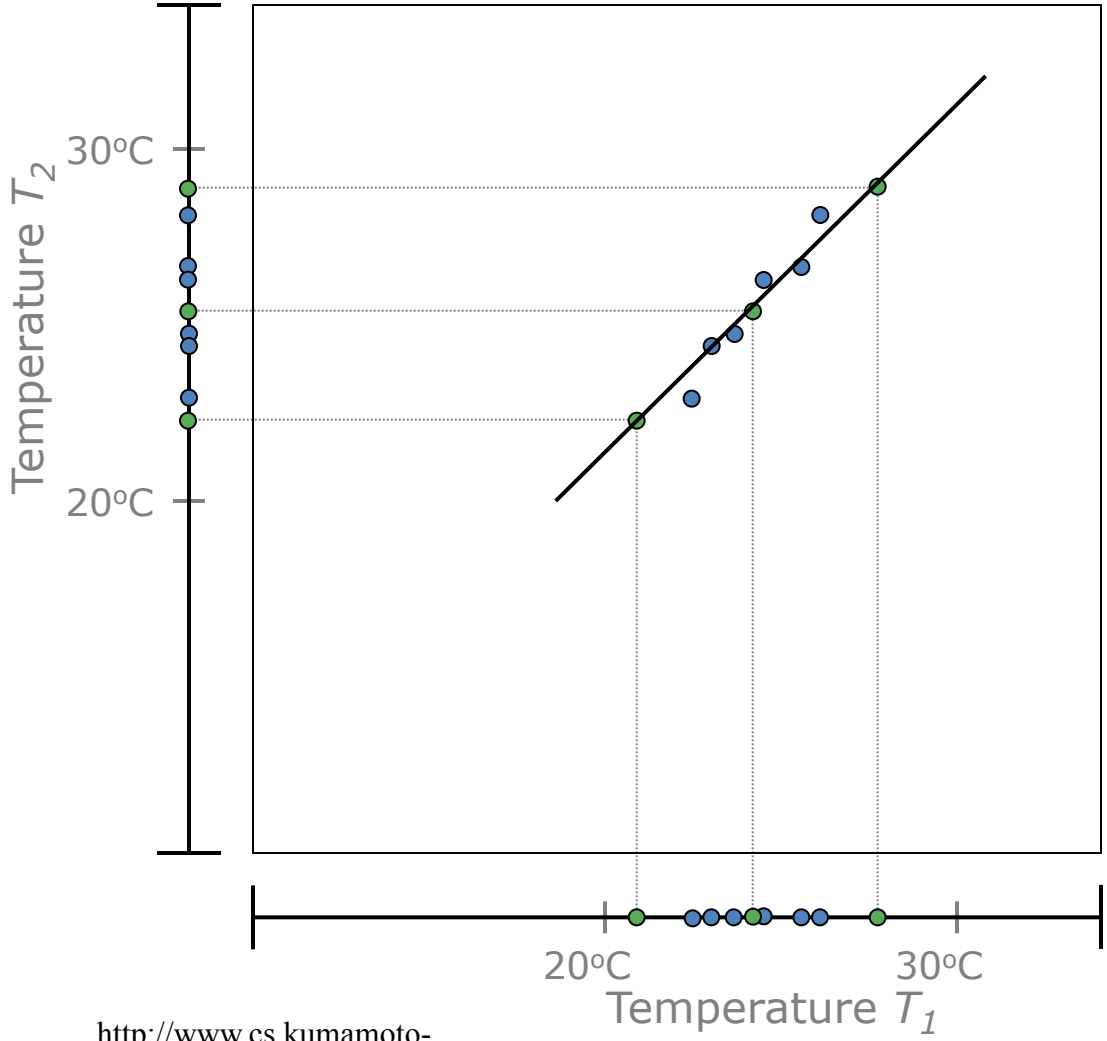
How to capture correlations?



First three lie (almost) on a line in the space of value-pairs…

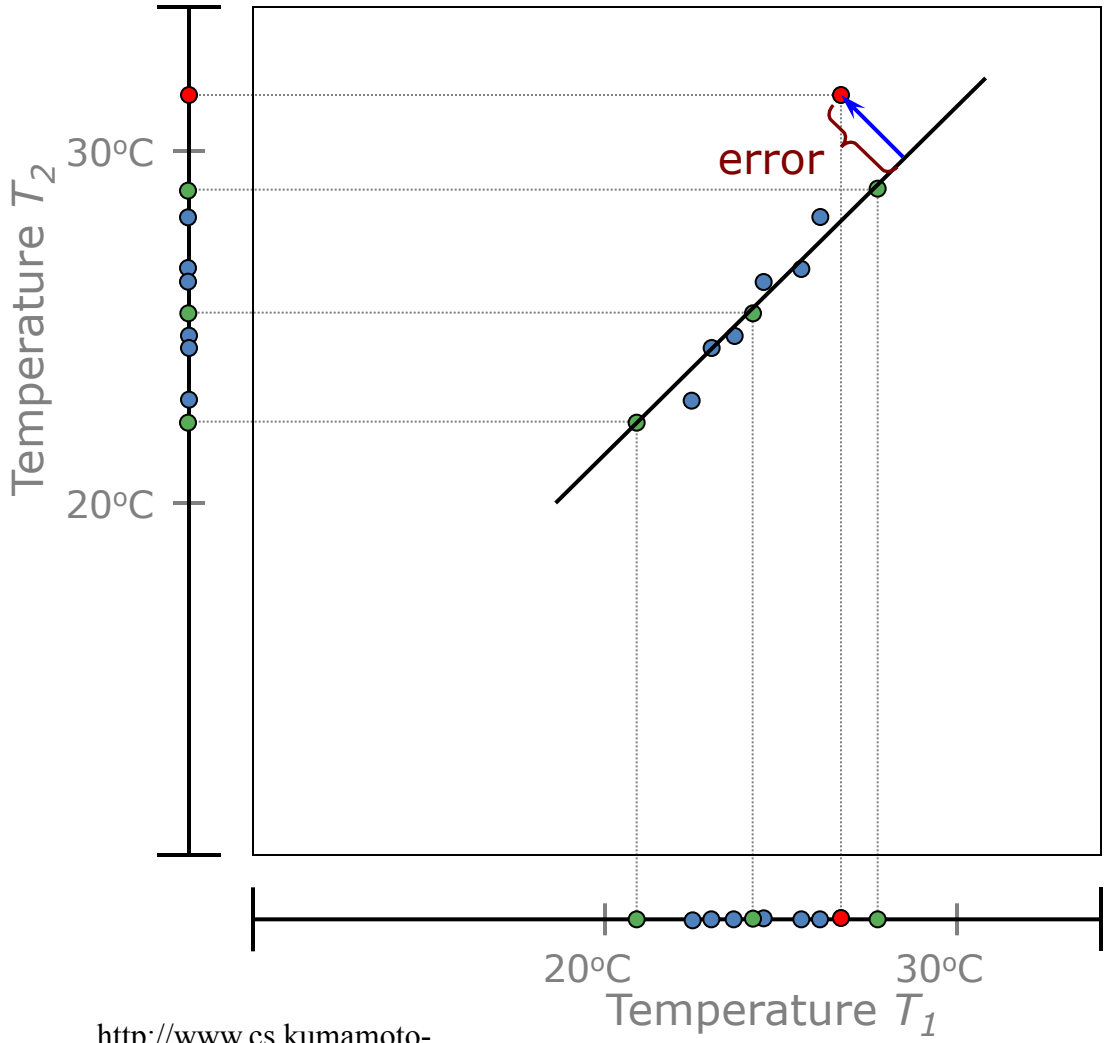
- O(*n*) numbers for the slope, and
- *One* number for each value-pair (offset on line)

How to capture correlations?



Other pairs also follow the same pattern: they lie (approximately) on this line

Incremental update

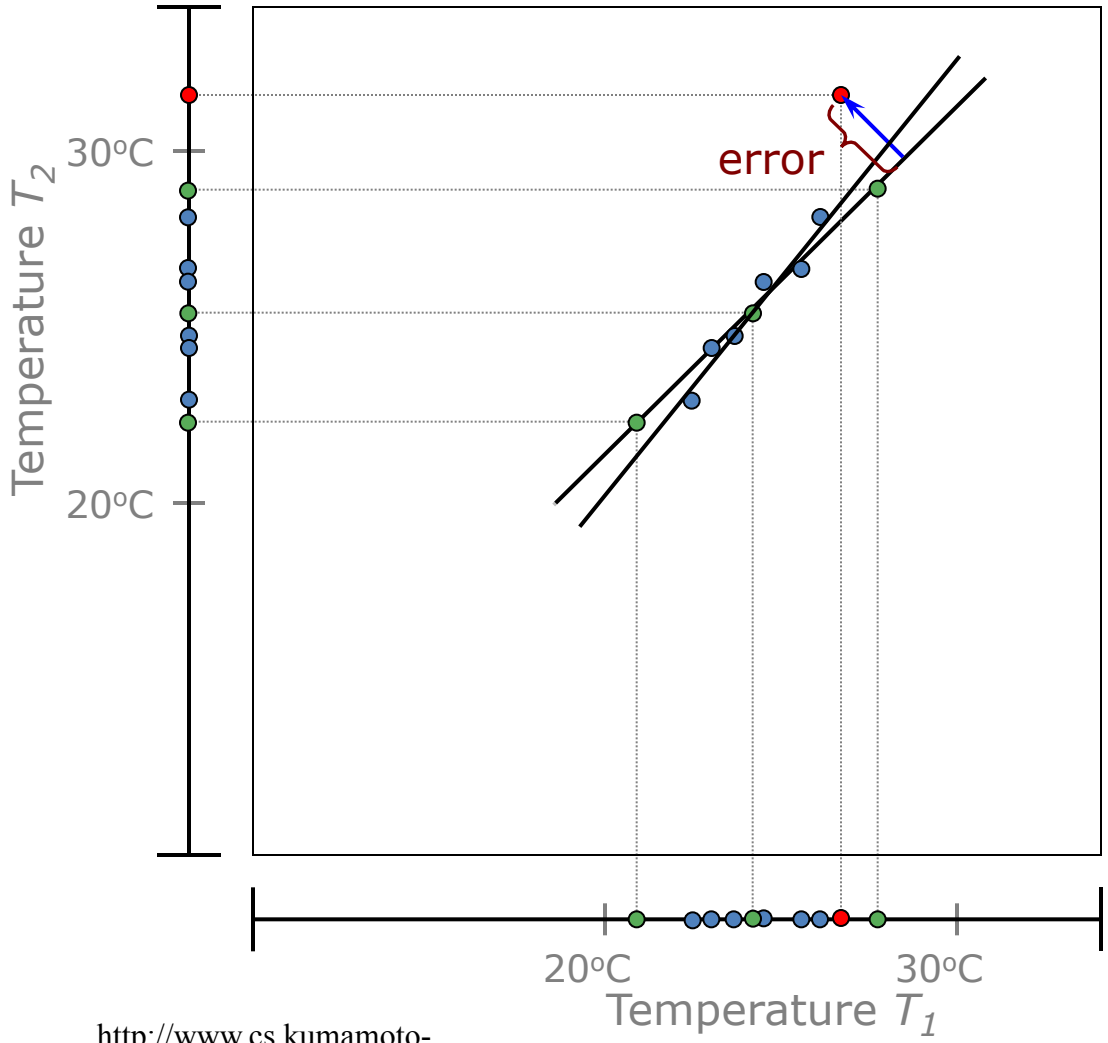


For each new point

- Project onto current line
- Estimate error

New value

Incremental update

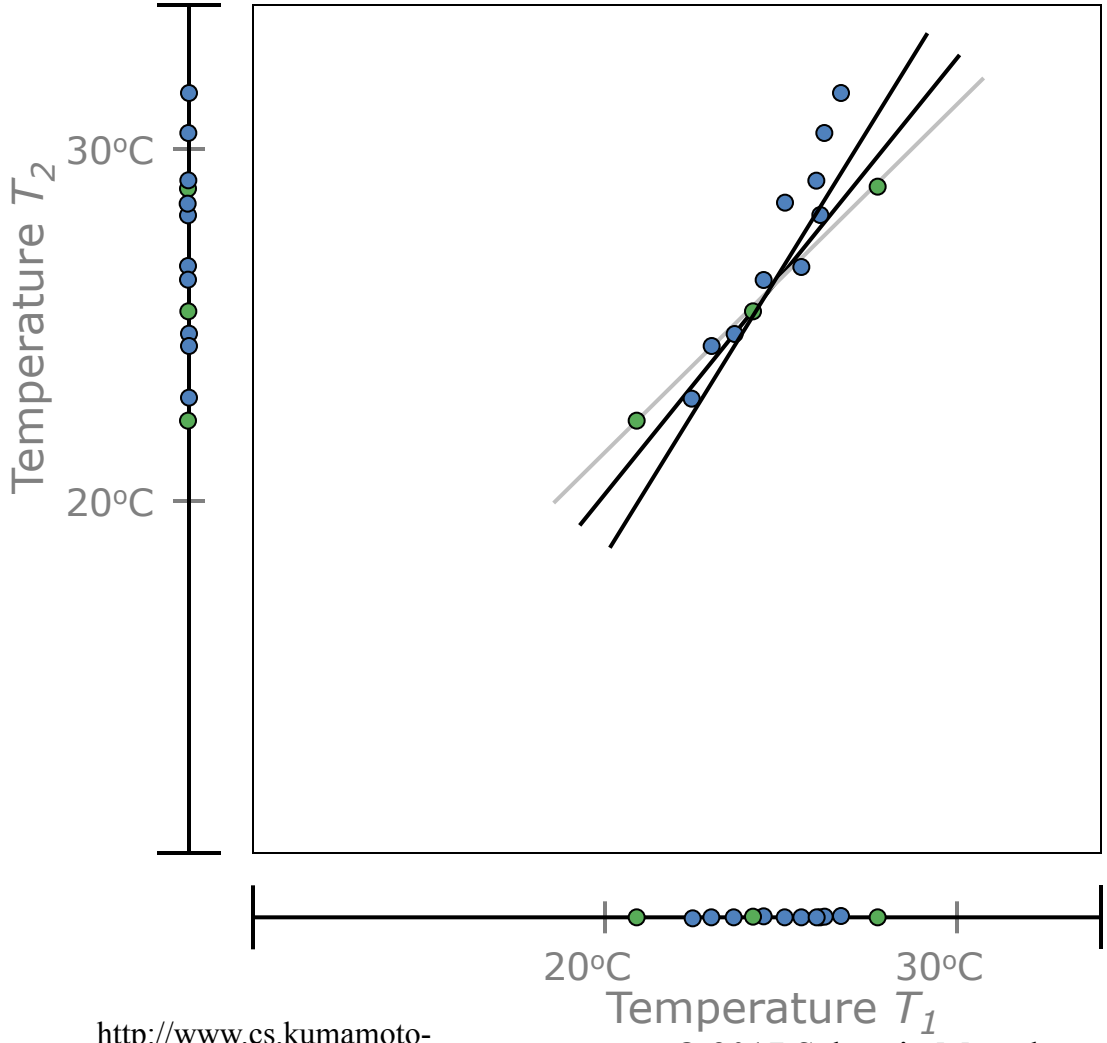


For each new point

- Project onto current line
- Estimate error
- Rotate line in the direction of the error and in proportion to its magnitude

➔O(n) time

Incremental update



For each new point

- Project onto current line
- Estimate error
- Rotate line in the direction of the error and in proportion to its magnitude

Related work

- Wavelet over streams [Gilbert+,vldb01] [Guha+,vldb04]
- Fourier representations [Gilbert+, stoc02]
- KNN [Koudas+, 04] [Korn+, vldb02]
- Histograms [Guha+, stoc01]
- Clustering [Guha+, focs00] [Aggarwal+, vldb03]
- Sketches [Indyk+, vldb00] [Cormode+, J. Algorithms 05]
- …
- …

Related work

- Heavy hitters [Cormode+, vldb03]
- Data embedding [Indyk+, focs00]
- Burst detection [Zhu+, kdd03]
- Segmentation [Keogh+, icdm01]
- Multiple scale analysis [Papadimitriou+, sigmod06]
- Fractal [Korn+, sigmod06]
- Time warping [Sakurai+, icde07]…
- …

Outline

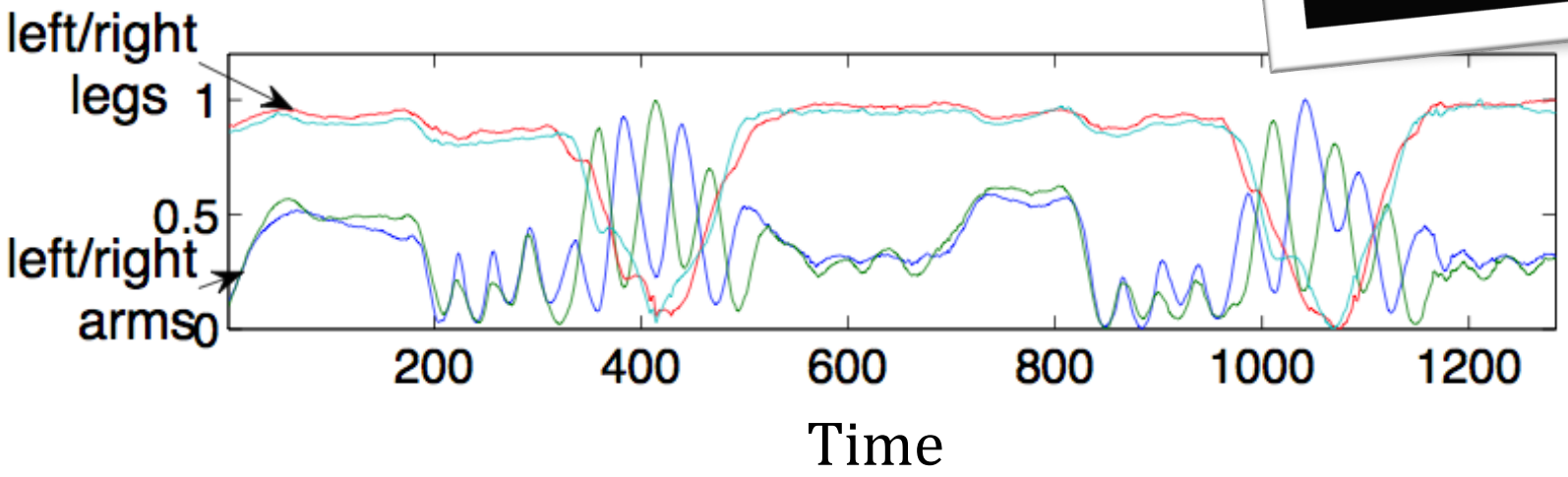
- Motivation
- Similarity Search and Indexing
- Feature extraction
- Streaming pattern discovery
- Linear forecasting
- Automatic mining

Motivation

Given: co-evolving time-series

– e.g., MoCap (leg/arm sensors)

"Chicken dance"

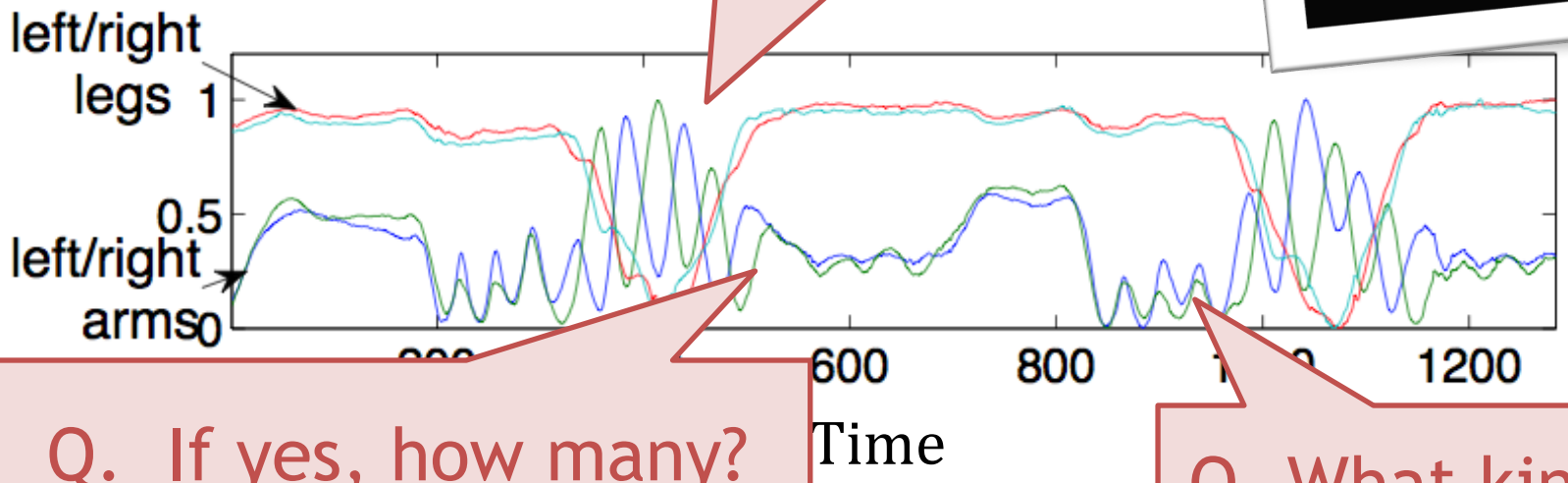


Motivation

Given: co-evolving time-series

– e.g., MoCap (leg/arm sensors)

"Chicken dance"



Q. Any distinct patterns?

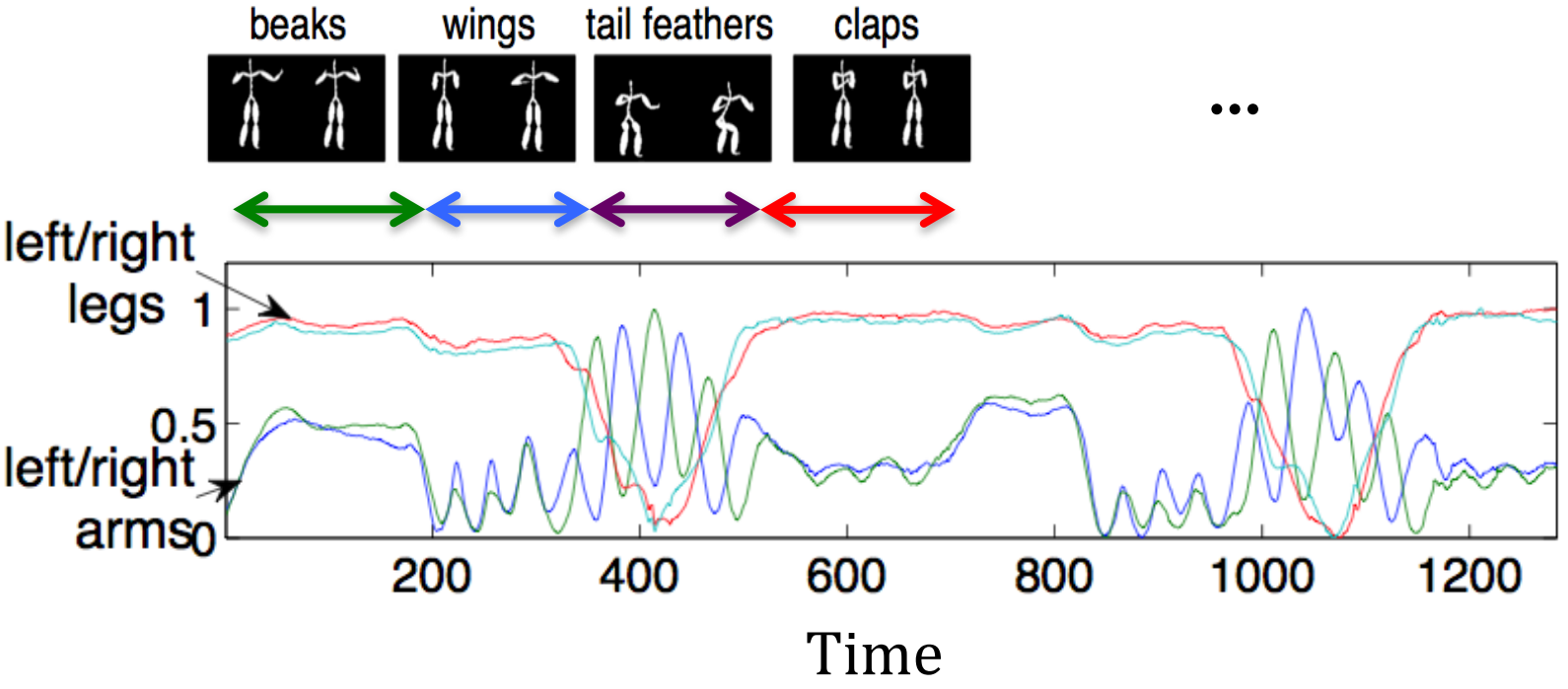
Q. If yes, how many?

Q. What kind?

Motivation

Challenges: co-evolving sequences

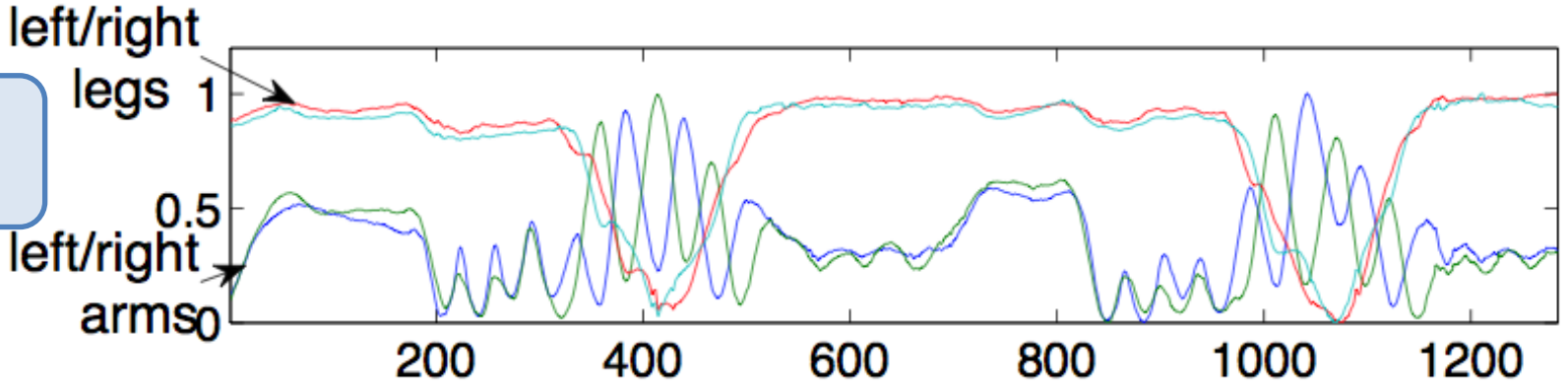
- Unknown # of patterns (e.g., beaks)
- Different durations



Motivation

Goal: find patterns that agree with human intuition

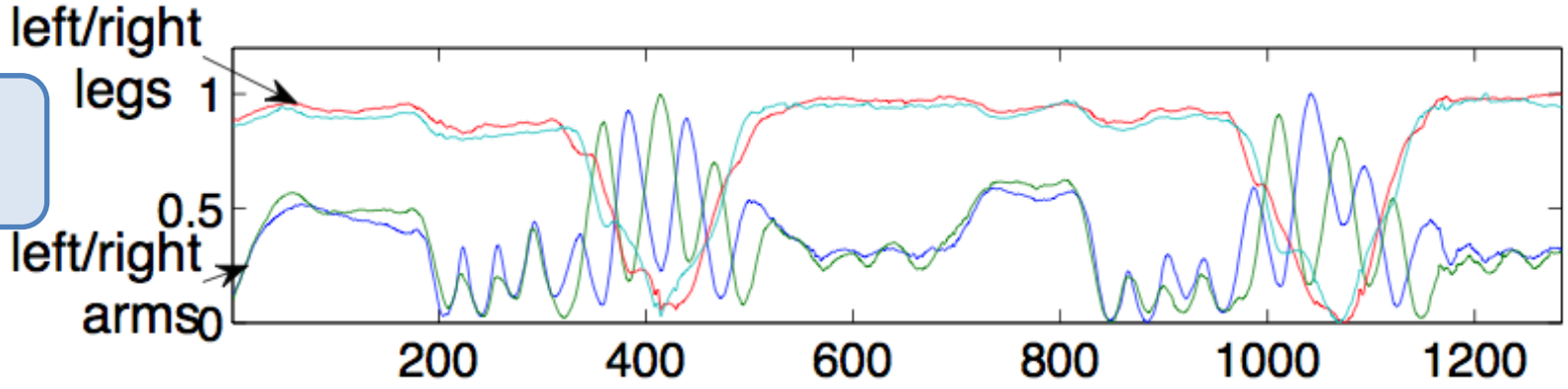
Input



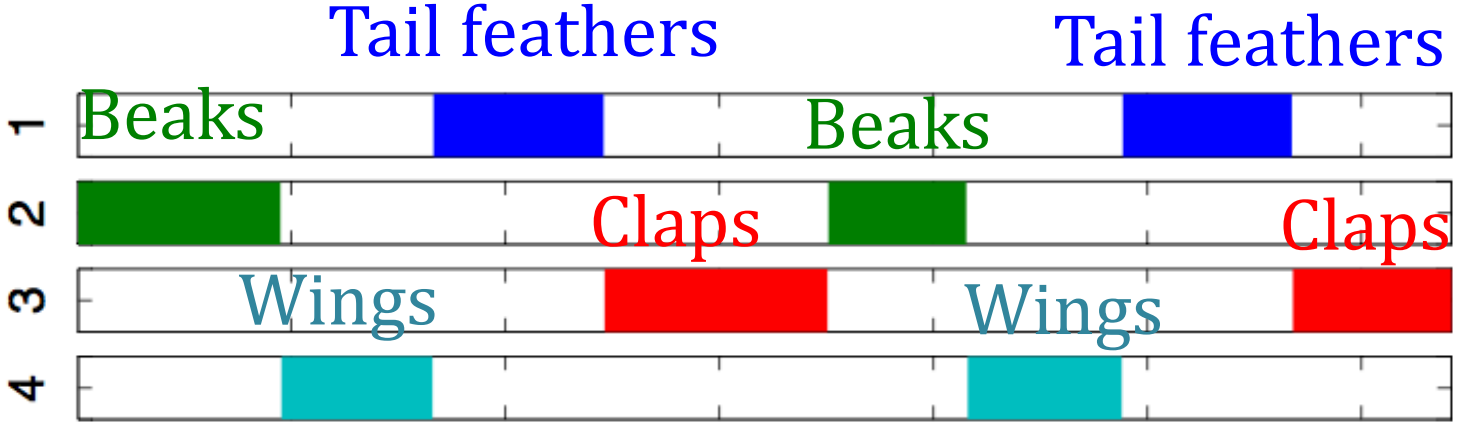
Motivation

Goal: find patterns that agree with human intuition

Input



Output



Motivation

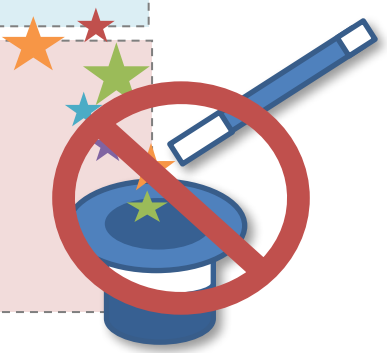
Goal: find patterns that agree with human intuition

Why: Automatic mining

No magic numbers! ... because,

Manual (use magic)

- sensitive to the parameter tuning
- long tuning steps (hours, days, …)



Automatic (no magic numbers)

- no expert tuning required

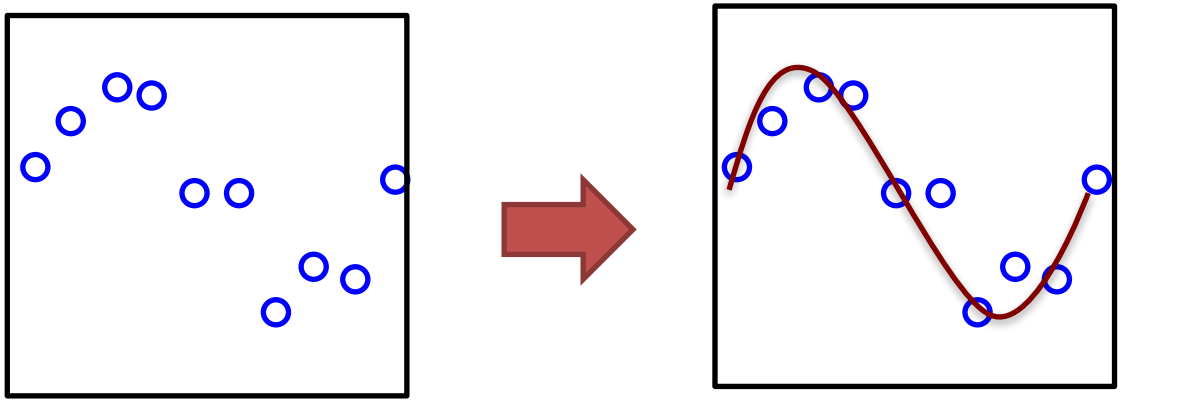
Big data mining:

-> we cannot afford human intervention!!

How: Automatic mining

Goal: fully-automatic modeling

- Given: **data X**
- Find: a compact description **(model M)** of X



Data (X)

Ideal model (M)

Q. How can we find the best model M?

How: Automatic mining

Goal: fully-automatic modeling

- Given: **data X**
- Find: a compact description **(model M)** of X

Answer: MDL!

Data (X) Ideal model (M)

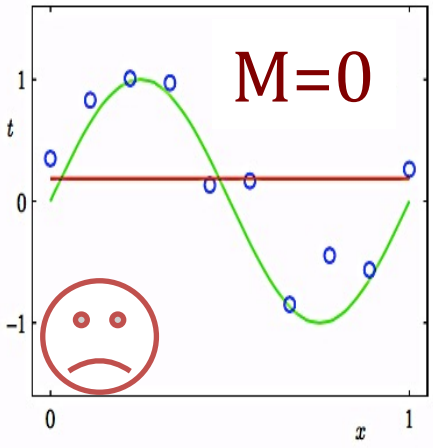
Q. How can we find the best model M?

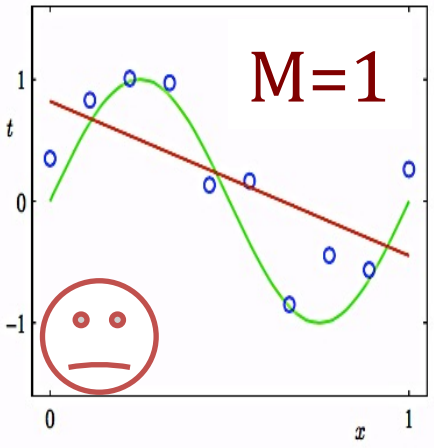
Solution: MDL (Minimum description length)

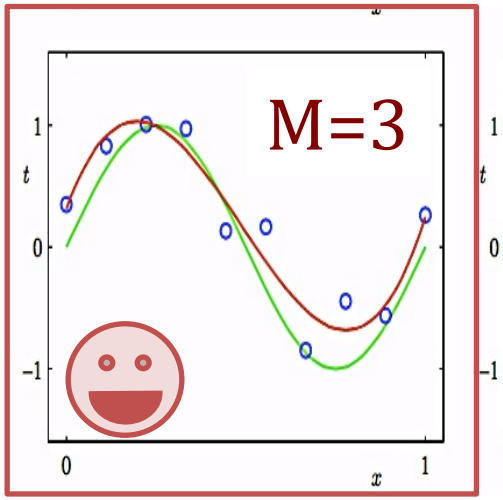
Solution: Minimize total encoding cost $!

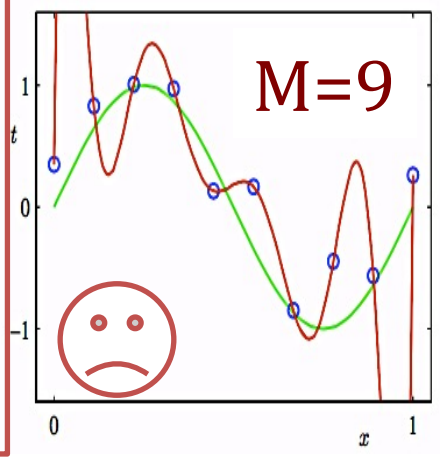
- Occam's razor (i.e., law of parsimony)
- **Fully automatic** parameter optimization
- No over-fitting

Ideal model









[Bishop: PR&ML]

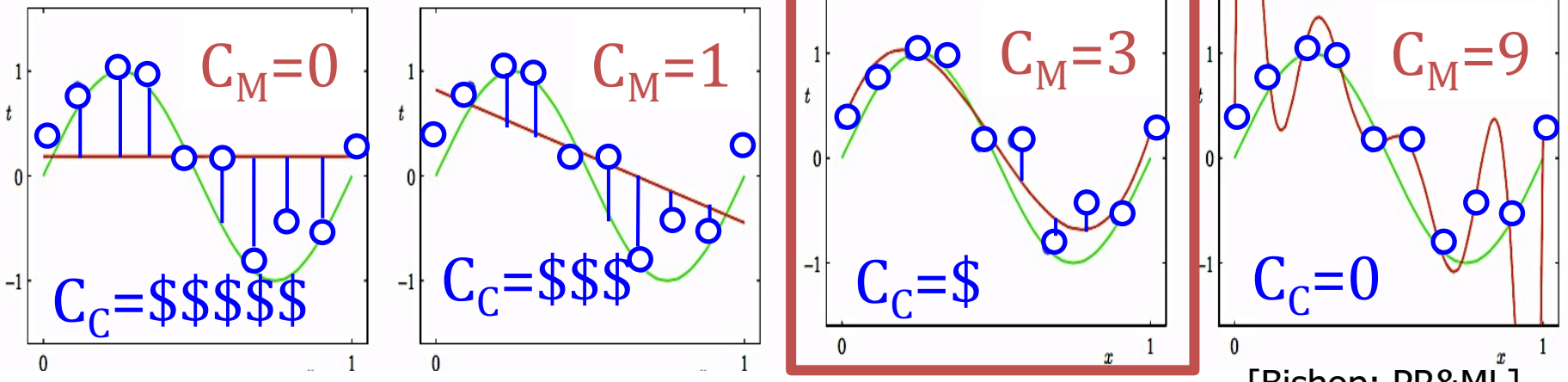
Solution: MDL (Minimum description length)

Solution: Minimize total encoding cost $!

$$\text{Cost}_T(X;M) = \min\left(\boxed{\text{Cost}_M(M)} + \boxed{\text{Cost}_c(X|M)} \right)$$

Total cost | Model cost | Coding cost (error)

$$$** **$$** **$ (Ideal!) **$$$$**



[Bishop: PR&ML]



[Matsubara+ SIGMOD'14]

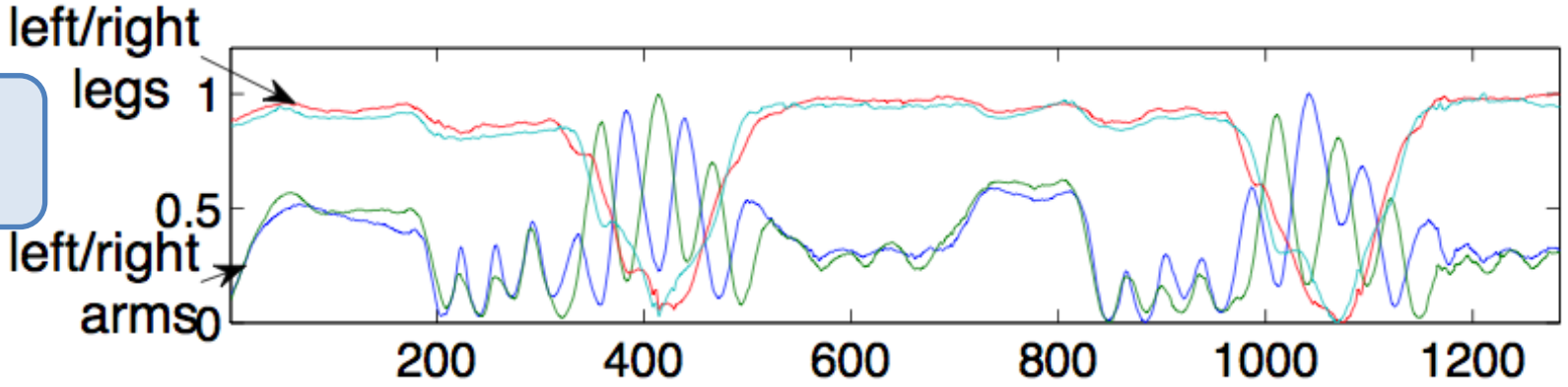
AutoPlait: Automatic Mining of Co-evolving Time Sequences

Yasuko Matsubara (Kumamoto University)
Yasushi Sakurai (Kumamoto University),
Christos Faloutsos (CMU)

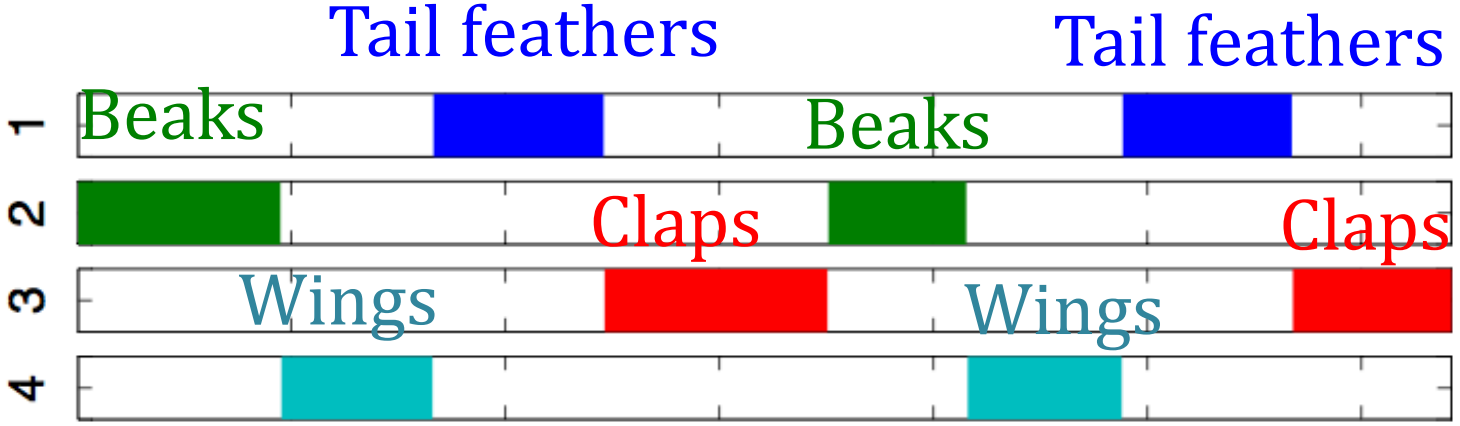

Problem definition

Goal: find patterns that agree with human intuition

Input



Output



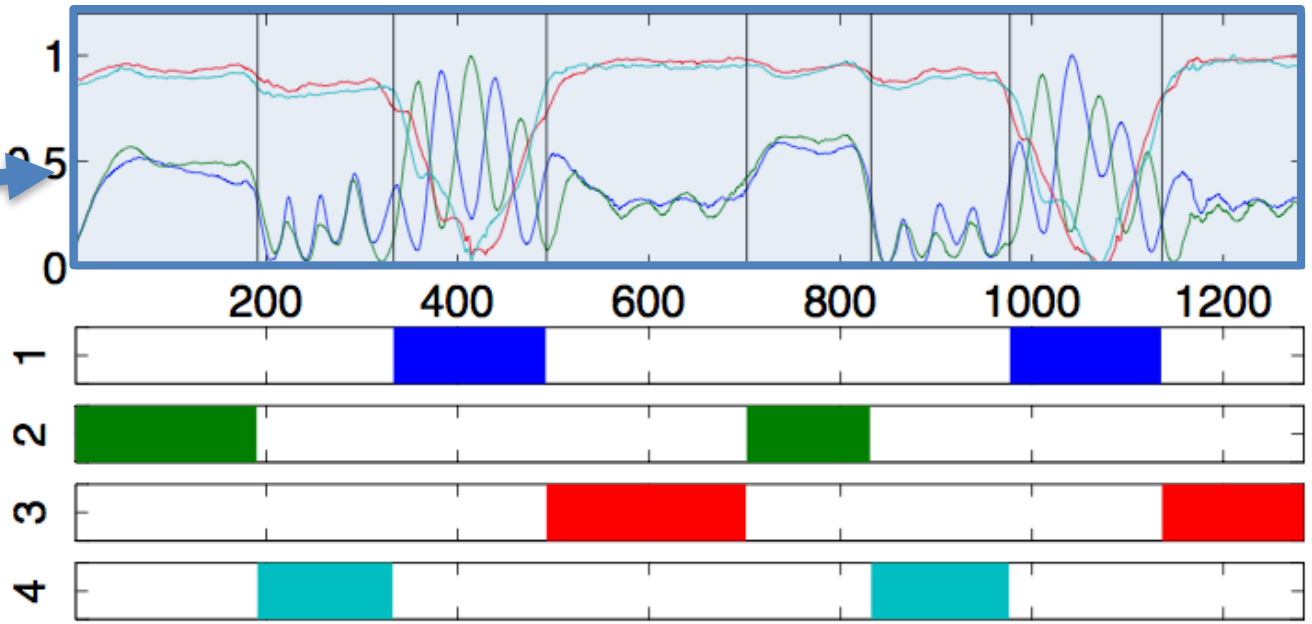
Problem definition

- Bundle : set of d co-evolving sequences

given

$$\underset{d \times n}{X} = \{x_1, \ldots, x_n\}$$

Bundle X (d=4)



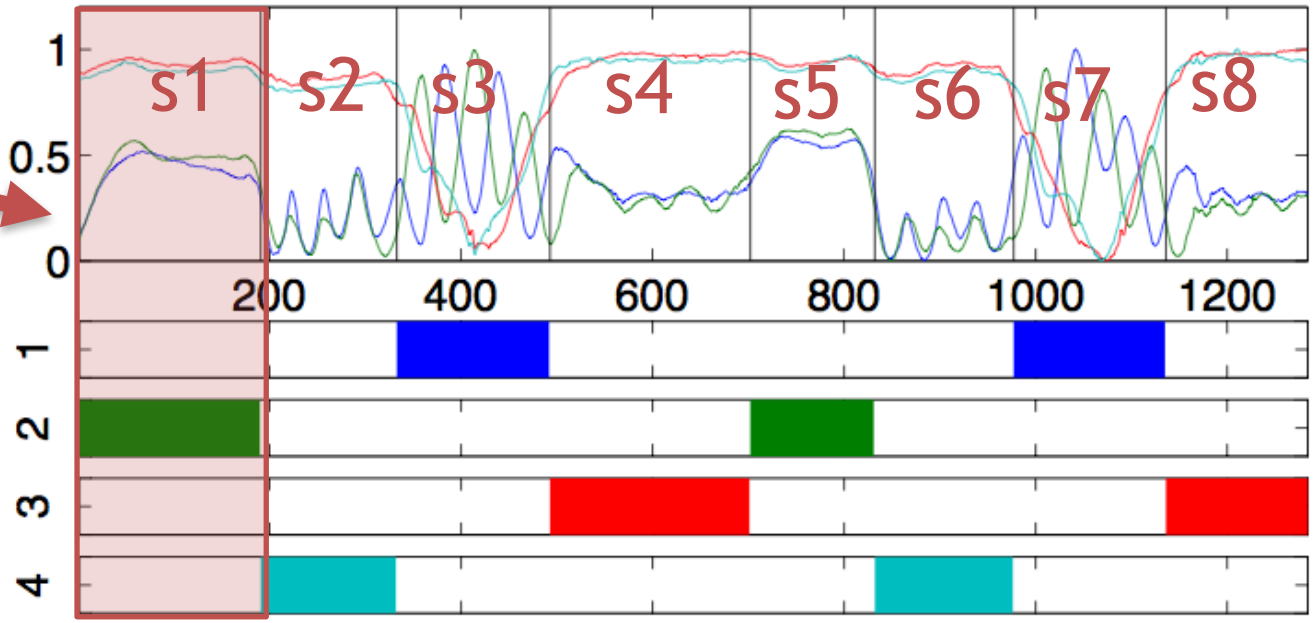
Problem definition

- Segment: convert X -> m segments, S

hidden

$$S = \{s_1, \ldots, s_m\}$$

Segment (m=8)



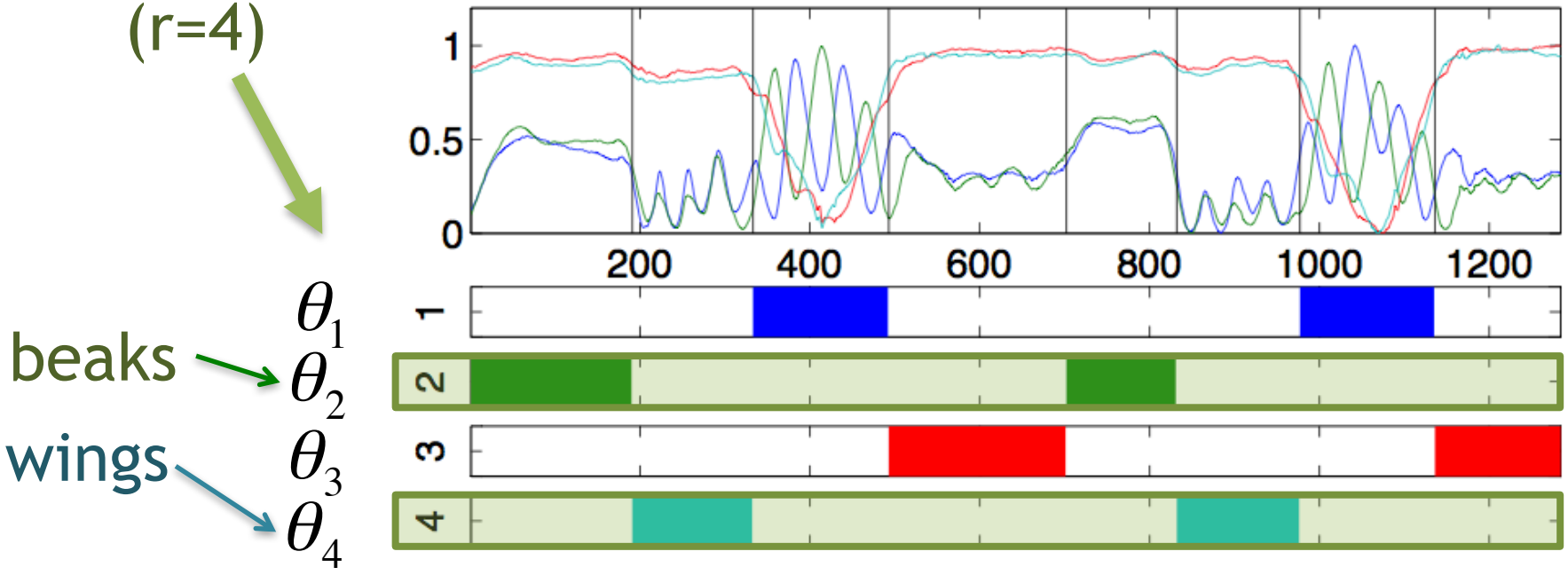
Problem definition

- Regime: segment groups: $\Theta = \{\theta_1, \theta_2, \ldots \theta_r, \Delta_{r \times r}\}$

hidden

Regimes (r=4)

θ_r : model of regime r

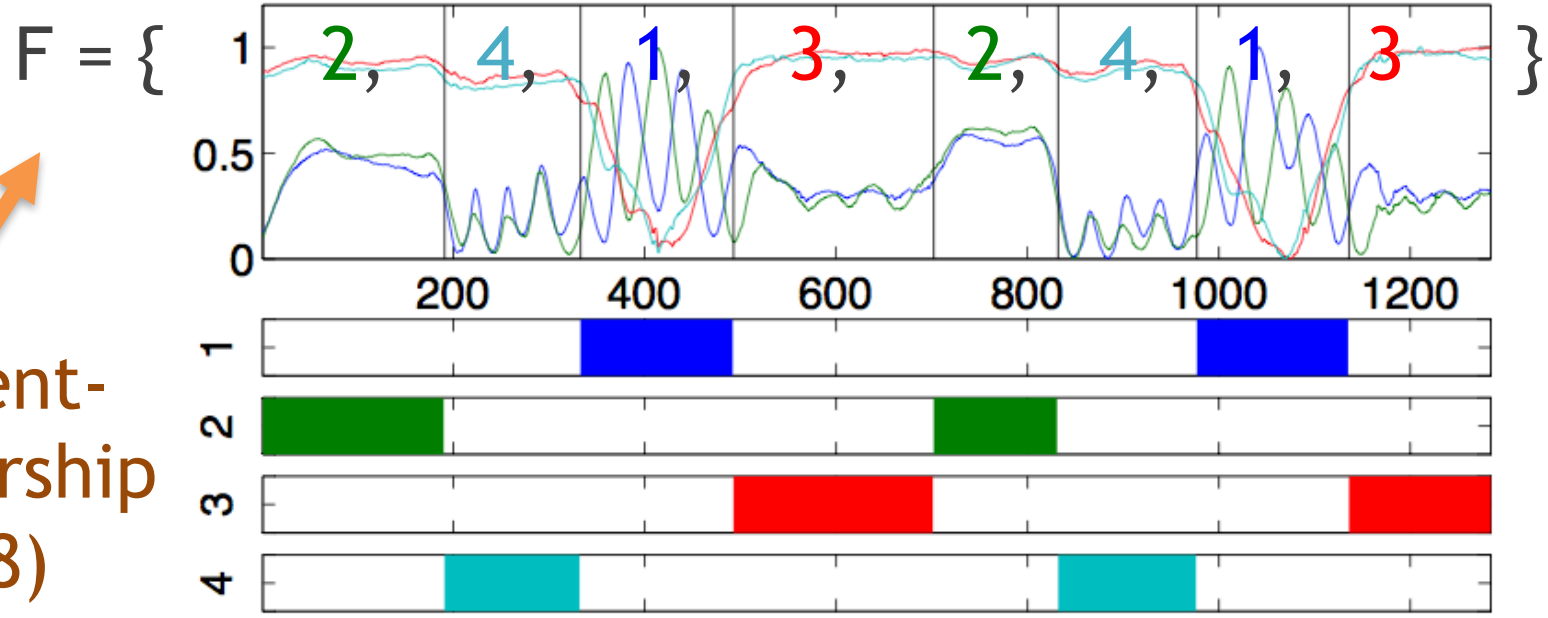


Problem definition

- Segment-membership: assignment

hidden

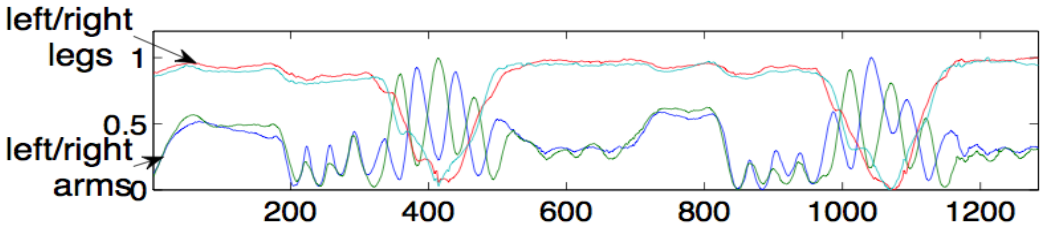
$$F = \{f_1, \ldots, f_m\}$$



Problem definition

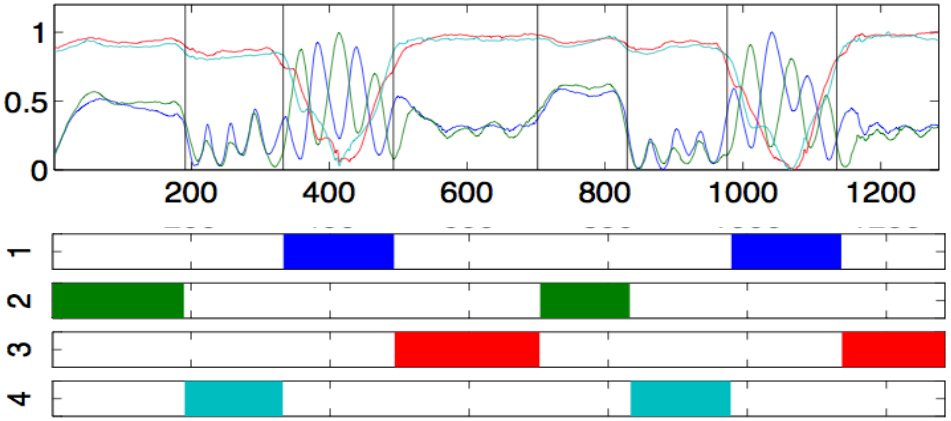
- Given: bundle X

$$X = \{x_1, \ldots, x_n\}$$



- Find: compact description C of X

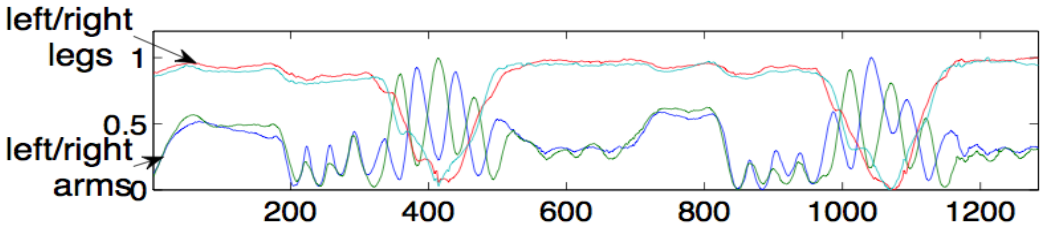
$$C = \{m, r, S, \Theta, F\}$$



Problem definition

- Given: bundle X

$$X = \{x_1, \ldots, x_n\}$$



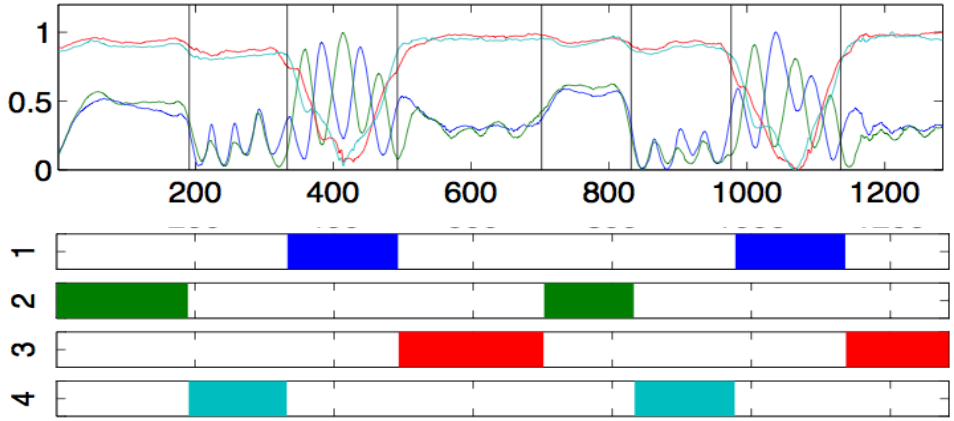
- Find: compact description C of X

$$C = \{m, r, S, \Theta, F\}$$

m segments

r regimes

Segment-membership



Main ideas

Goal: compact description of X

$$C = \{m, r, S, \Theta, F\}$$

without user intervention!!

Challenges:

Q1. How to generate 'informative' regimes ?

Q2. How to decide # of regimes/segments ?

Main ideas

Goal: compact description of X

$$C = \{m, r, S, \Theta, F\}$$

without user intervention!!

Challenges:

Q1. How to generate 'informative' regimes ?

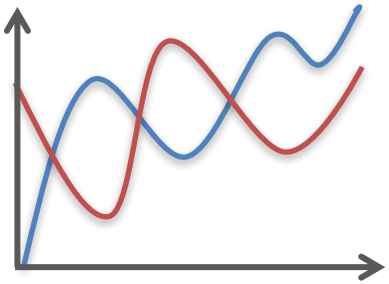
Idea (1): Multi-level chain model

Q2. How to decide # of regimes/segments ?

Idea (2): Model description cost

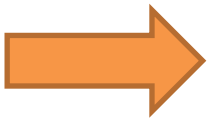
Idea (1): MLCM: multi-level chain model

Q1. How to generate 'informative' regimes ?



Sequences

Model



beaks

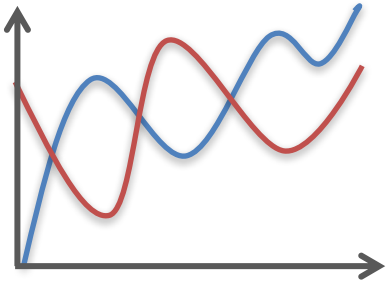
claps

wings

Regimes

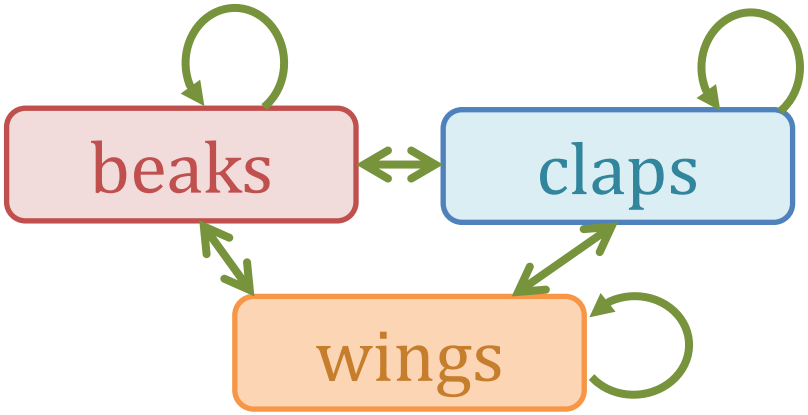
Idea (1): MLCM: multi-level chain model

Q1. How to generate 'informative' regimes ?



Sequences

Model



Regimes

Idea (1): Multi-level chain model

– HMM-based probabilistic model

– with "across-regime" transitions

Details

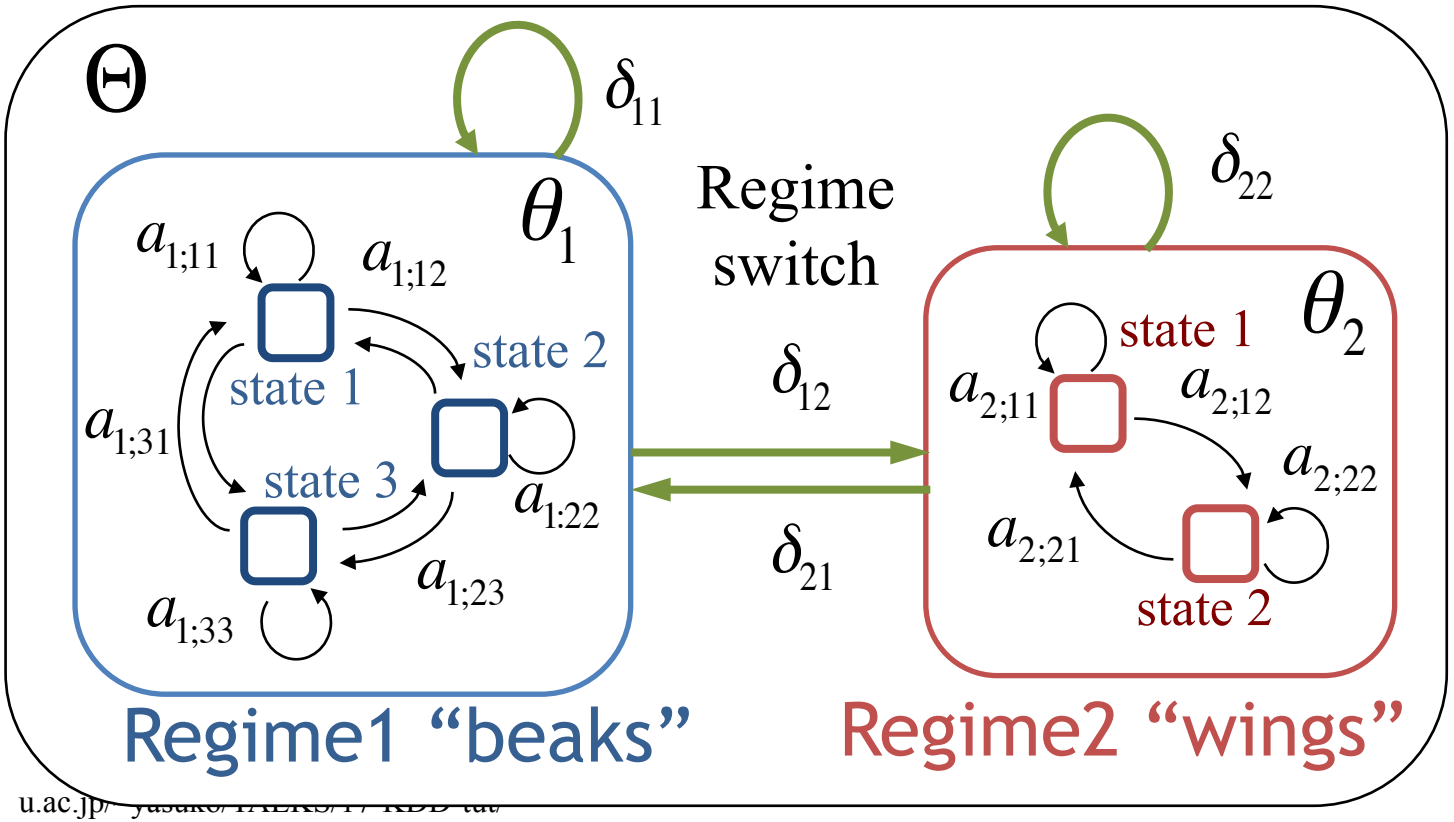
Idea (1): MLCM: multi-level chain model

$$\Theta = \{\theta_1, \theta_2, \ldots \theta_r, \Delta_{r\times r}\} \qquad (\theta_i = \{\pi, A, B\})$$

r regimes (HMMs)

across-regime transition prob.

Single HMM parameters



Regimes r=2

Regime 1 (k=3)

Regime 2 (k=2)

Idea (2): model description cost

Q2. How to decide # of regimes/segments ?

Idea (2): Model description cost

- Minimize encoding cost
- find "optimal" # of segments/regimes

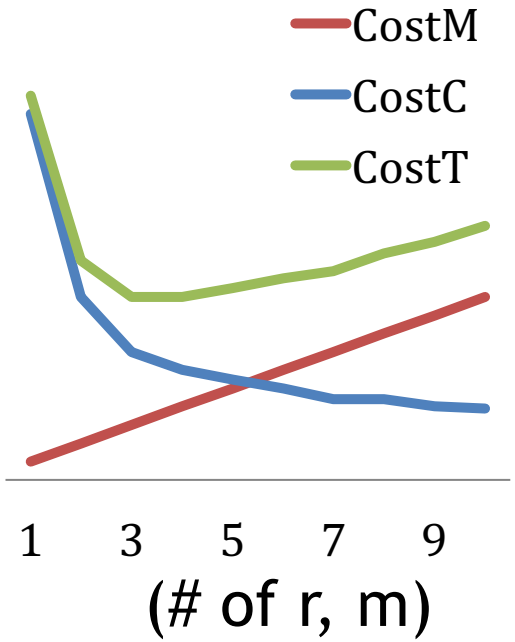
Idea (2): model description cost

Idea: Minimize encoding cost!

$$\min \left(\boxed{\text{Cost}_M(M)} + \boxed{\text{Cost}_c(X \mid M)} \right)$$

Model cost

Coding cost



Good compression  Good description

Details

Idea (2): model description cost

CMU CS

Total cost of bundle X, given C

$$C = \{m, r, S, \Theta, F\}$$

$$\begin{aligned} Cost_T(\boldsymbol{X}; \mathcal{C}) &= Cost_T(\boldsymbol{X}; m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F}) \\ &= \log^*(n) + \log^*(d) + \log^*(m) + \log^*(r) + m\log(r) \\ &\quad + \sum_{i=1}^{m-1} \log^* |s_i| + Cost_M(\boldsymbol{\Theta}) + Cost_C(\boldsymbol{X}|\boldsymbol{\Theta}) \end{aligned} \quad (6)$$

Details

Idea (2): model description cost

CMU CS

Total cost of bundle X, given C

$$C = \{m, r, S, \Theta, F\}$$

duration/dimensions

# of segments/regimes

segment-membership F

$$Cost_T(\mathbf{X}; \mathcal{C}) = Cost_T(\mathbf{X}; m, r, \mathcal{S}, \mathbf{\Theta}, \mathcal{F})$$

$$= \log^*(n) + \log^*(d) + \log^*(m) + \log^*(r) + m \log(r)$$

$$+ \sum_{i=1}^{m-1} \log^* |s_i| + Cost_M(\mathbf{\Theta}) + Cost_C(\mathbf{X}|\mathbf{\Theta}) \quad (6)$$

segment lengths

Model description cost of Θ

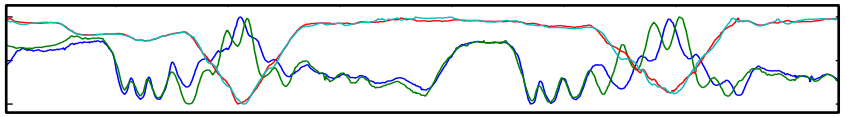
Coding cost of X given Θ

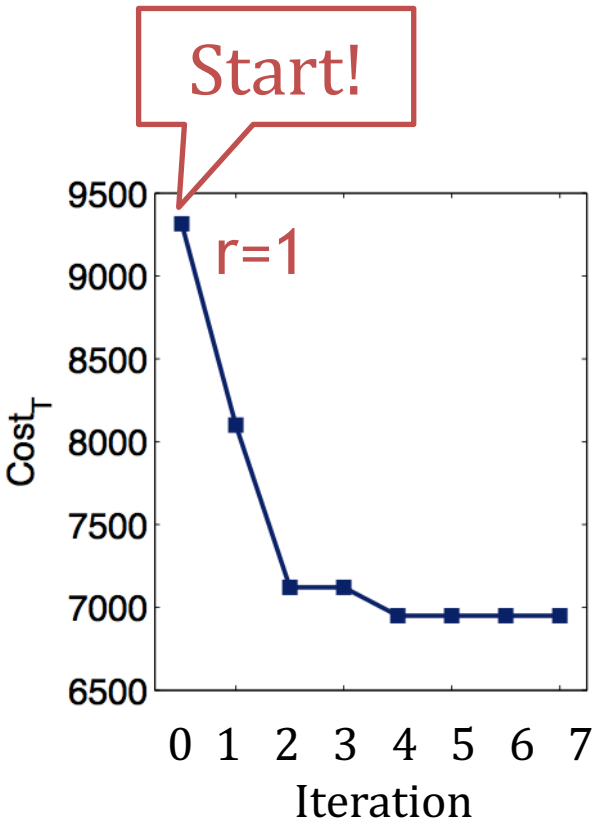
AutoPlait

Overview

Iteration 0
r=1, m=1

X

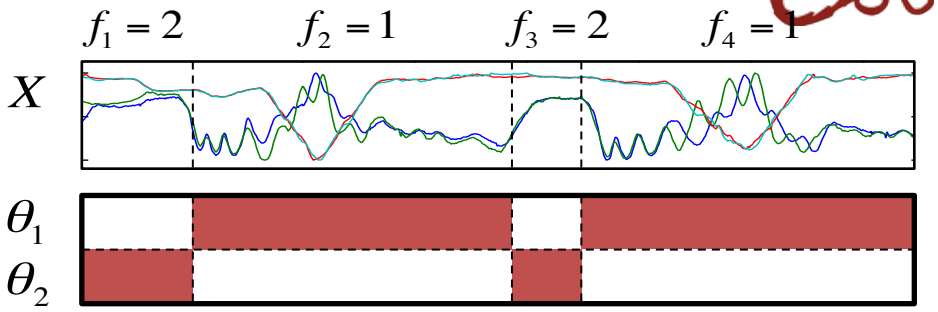


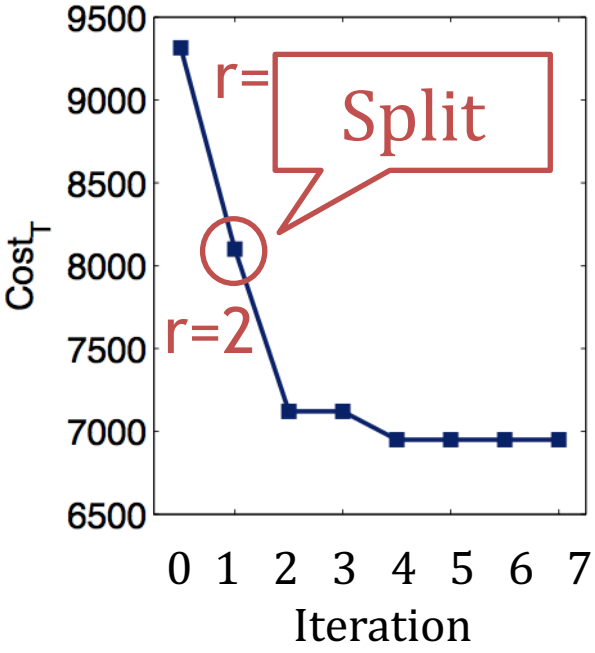


AutoPlait

Overview

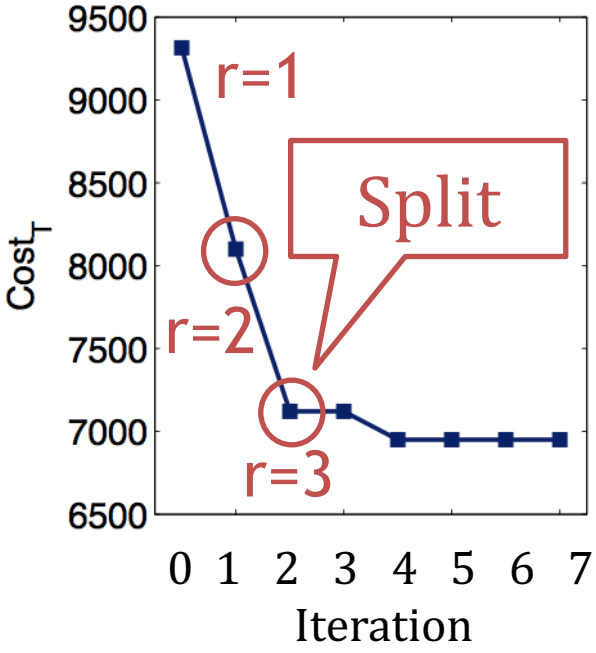
Iteration 1
r=2, m=4



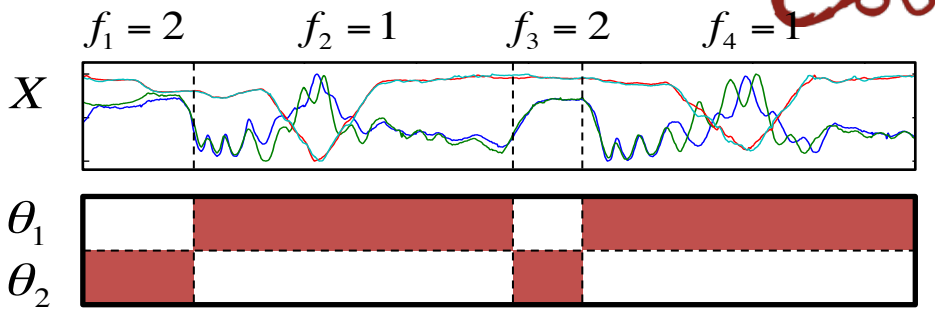


AutoPlait

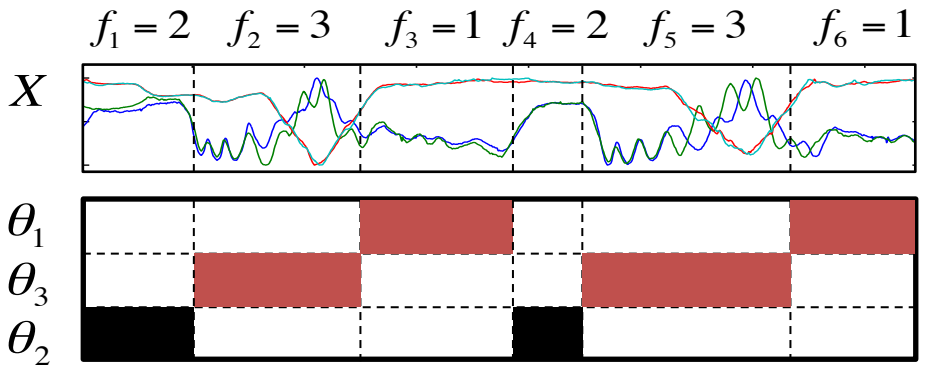
Overview



Iteration 1
r=2, m=4

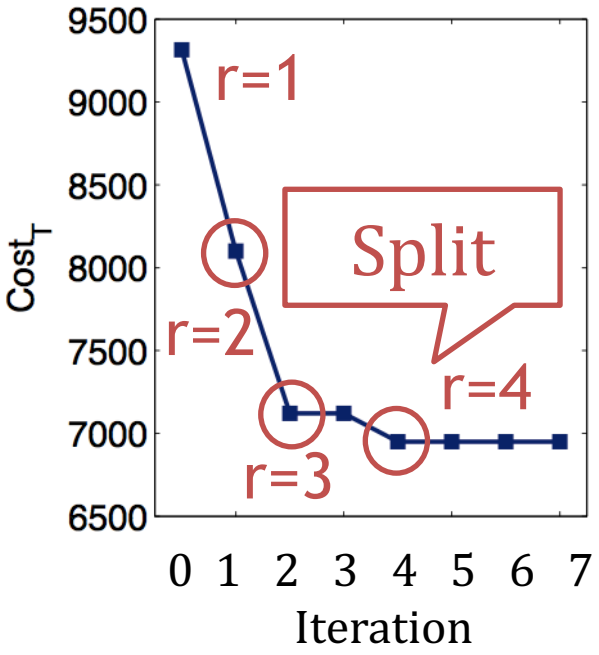


Iteration 2
r=3, m=6



AutoPlait

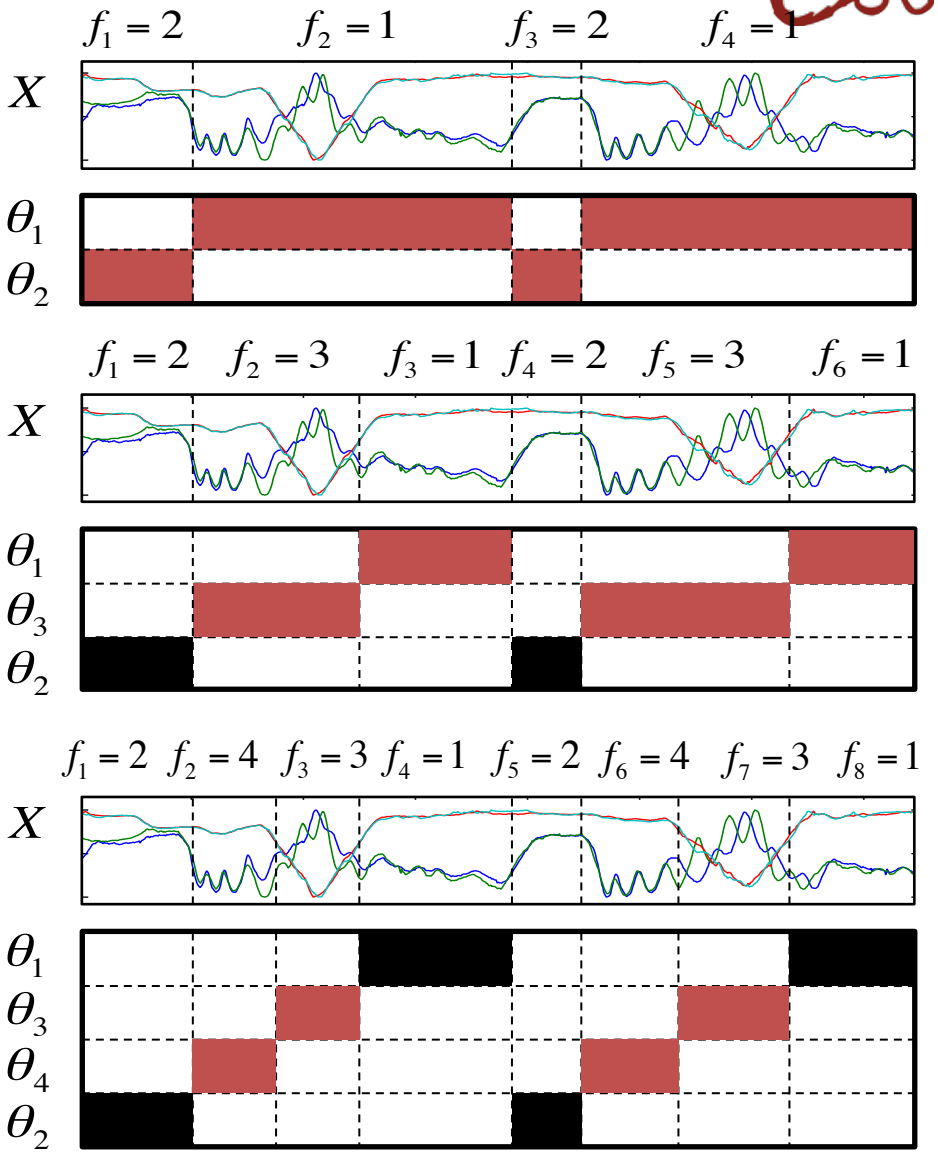
Overview



Iteration 1
r=2, m=4

Iteration 2
r=3, m=6

Iteration 4
r=4, m=8



AutoPlait

Algorithms

1. CutPointSearch — Inner-most loop

   Find good cut-points/segments

2. RegimeSplit — Inner loop

   Estimate good regime parameters **Θ**

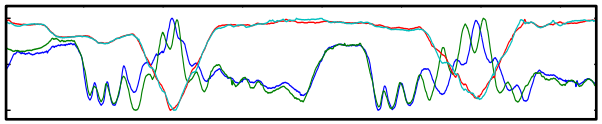
3. AutoPlait — Outer loop

   Search for the best number of regimes (r=2,3,4…)

1. CutPointSearch

Inner-most loop

Given:

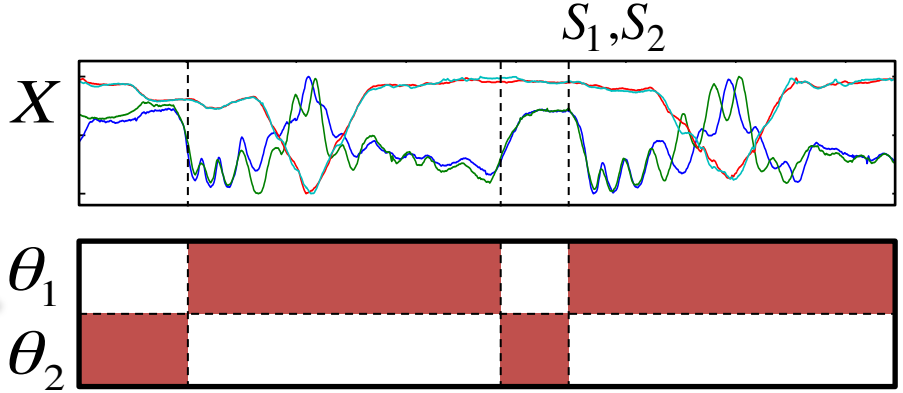
- bundle X 
- parameters of two regimes $\Theta = \{\theta_1, \theta_2, \Delta\}$

Find: cut-points of segment sets S_1, S_2,

$$\{S_1, S_2\} = \operatorname{argmax} P(X \mid S_1, S_2, \Theta)$$

X

$\{\theta_1, \theta_2, \Delta\}$

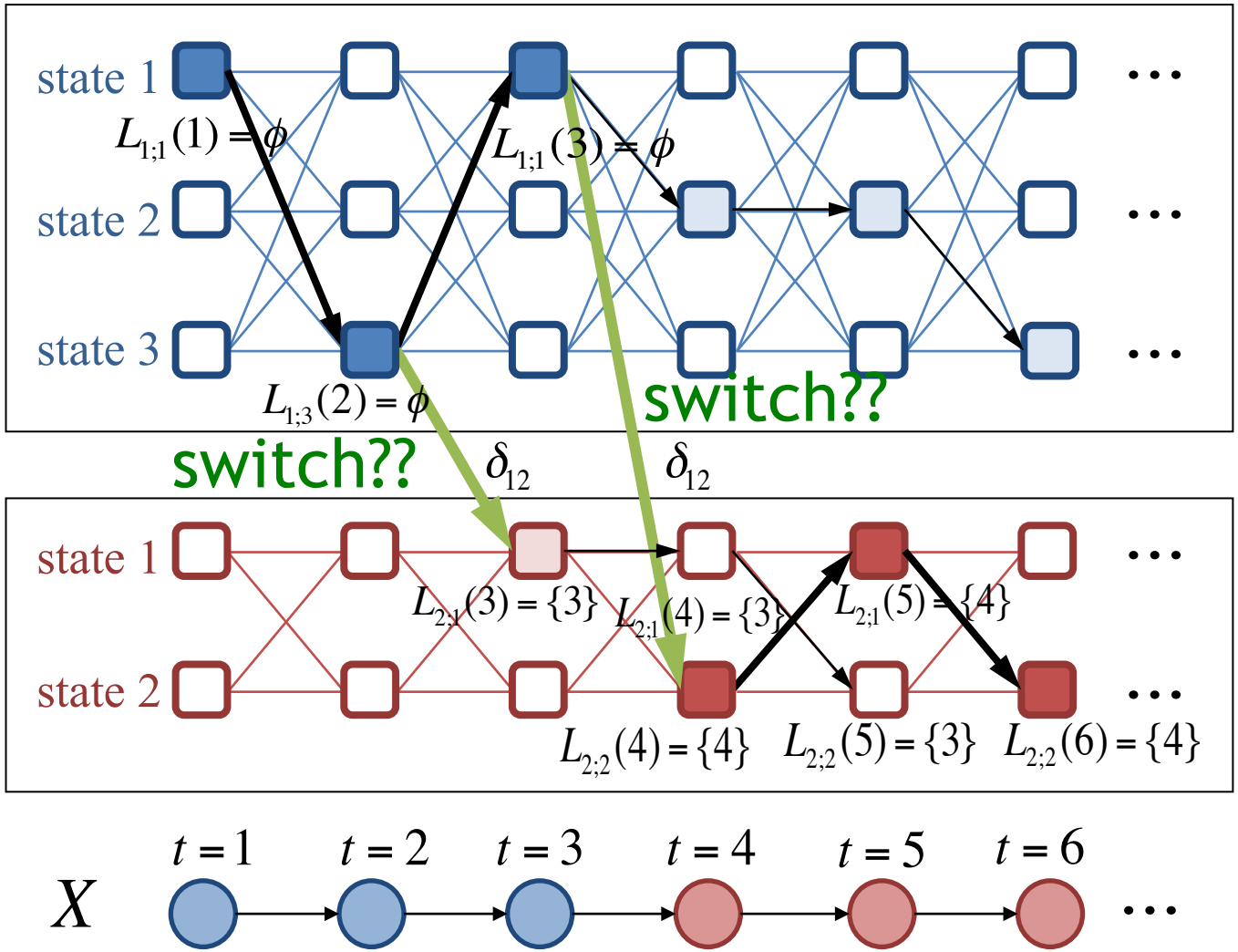


$S_1 = \{s_2, s_4\}$
$S_2 = \{s_1, s_3\}$

1. CutPointSearch

DP algorithm to compute likelihood:

$P(X \mid \Theta)$



1. CutPointSearch

Theoretical analysis

Scalability

- It takes $O(ndk^2)$ time (only single scan)
  - n: length of X
  - d: dimension of X
  - k: # of hidden states in regime

Accuracy

It guarantees the optimal cut points

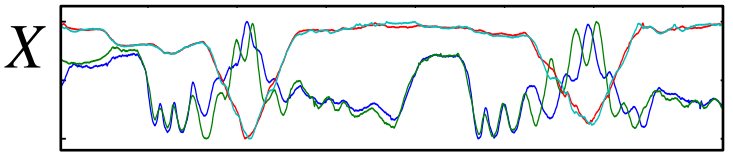
- (Details in paper)

2. RegimeSplit

Inner loop

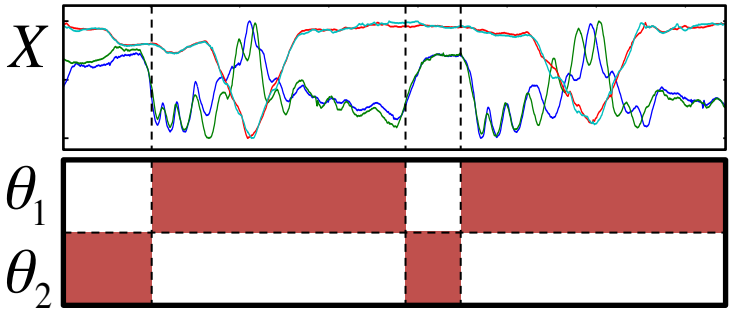
Given:

- bundle X



Find: two regimes

1. find cut-points of segment sets: S_1, S_2
2. estimate parameters of two regimes:

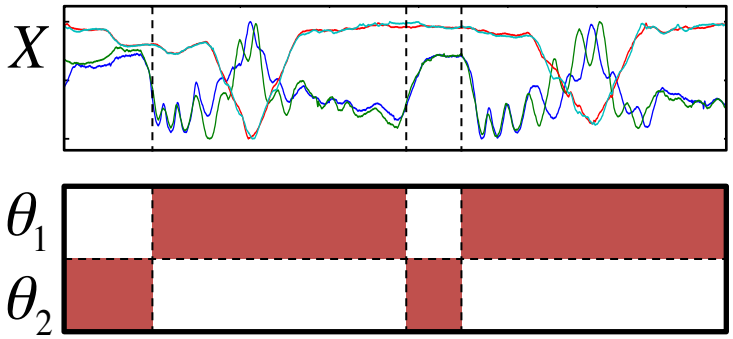


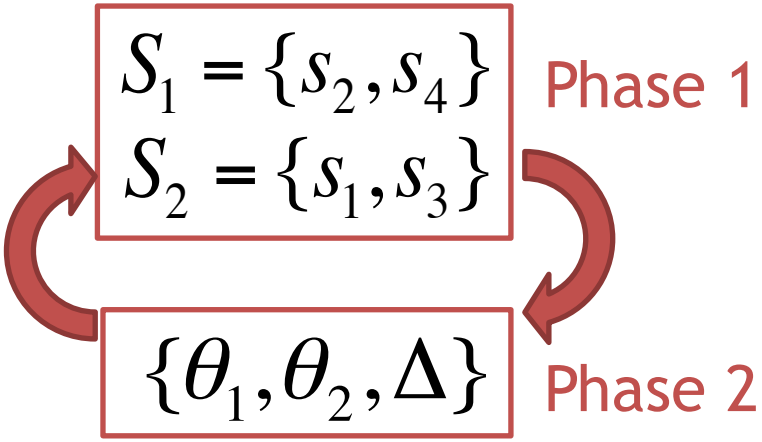
$$\Theta = \{\theta_1, \theta_2, \Delta\}$$

2. RegimeSplit

Two-phase iterative approach

- Phase 1: (CutPointSearch)
  - Split segments into two groups : S_1, S_2
- Phase 2: (BaumWelch)
  - Update model parameters: $\Theta = \{\theta_1, \theta_2, \Delta\}$



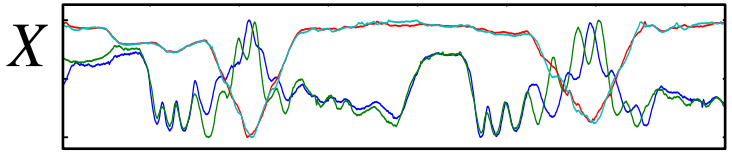


3. AutoPlait

Outer loop

Given:

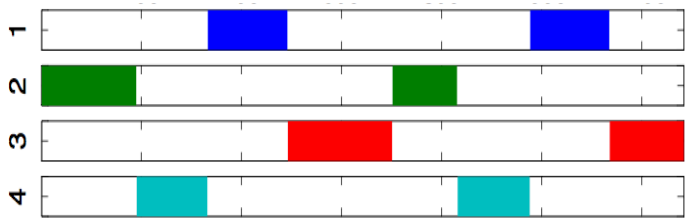
- bundle X



Find: r regimes (r=2, 3, 4, …)

- i.e., find full parameter set

$$C = \{m, r, S, \Theta, F\}$$



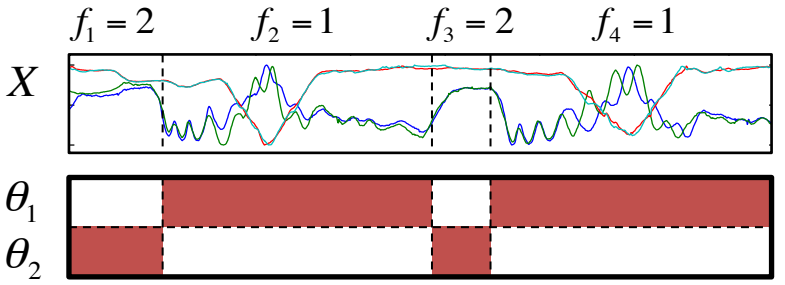
3. AutoPlait

Split regimes r=2,3,…, as long as cost keeps decreasing

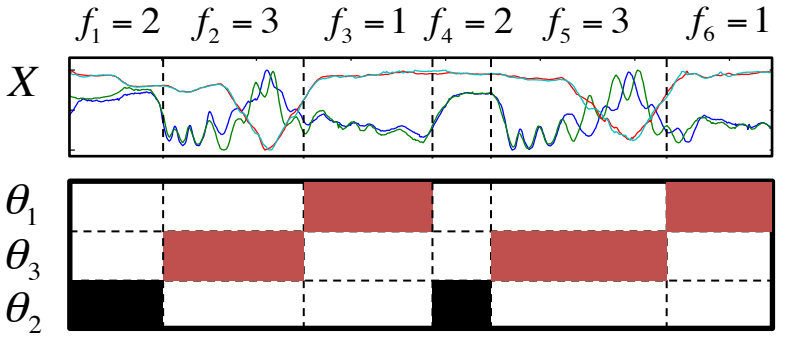
- Find appropriate # of regimes

$$r = \min_{r} Cost_T(X; m, r, S, \Theta, F)$$

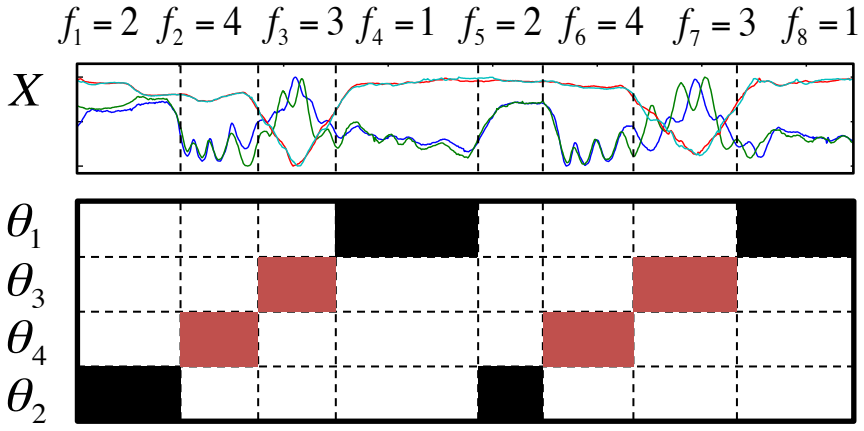
r=2, m=4



r=3, m=6



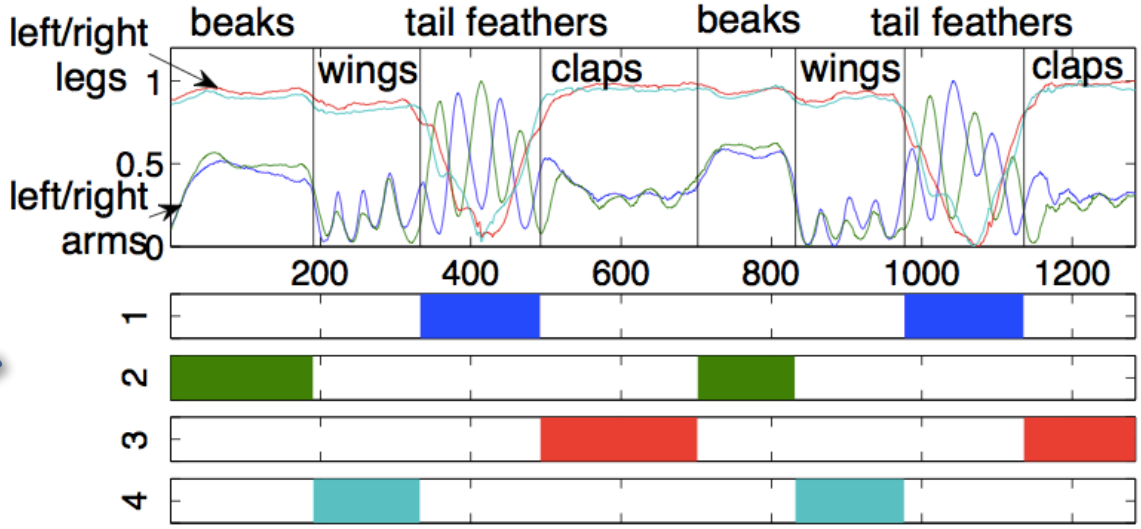
r=4, m=8



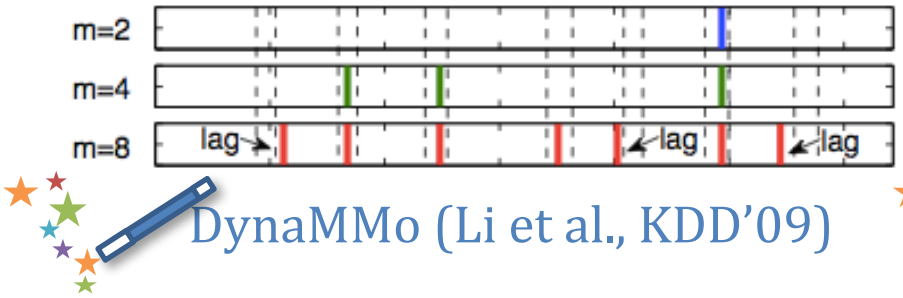
Sense-making

MoCap data

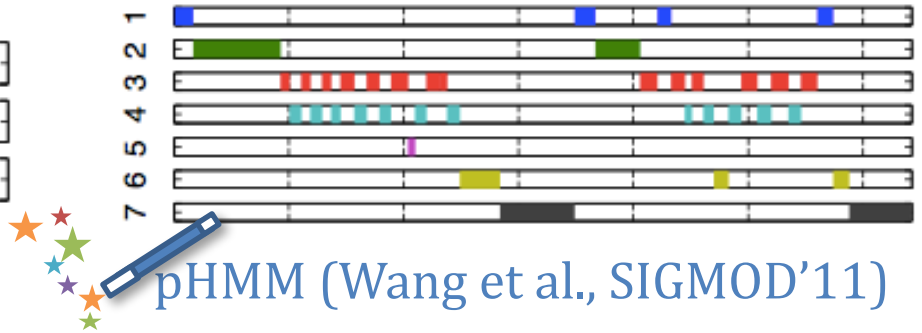
AutoPlait (NO magic numbers)



(a) AUTOPLAIT (**no** user defined parameters)



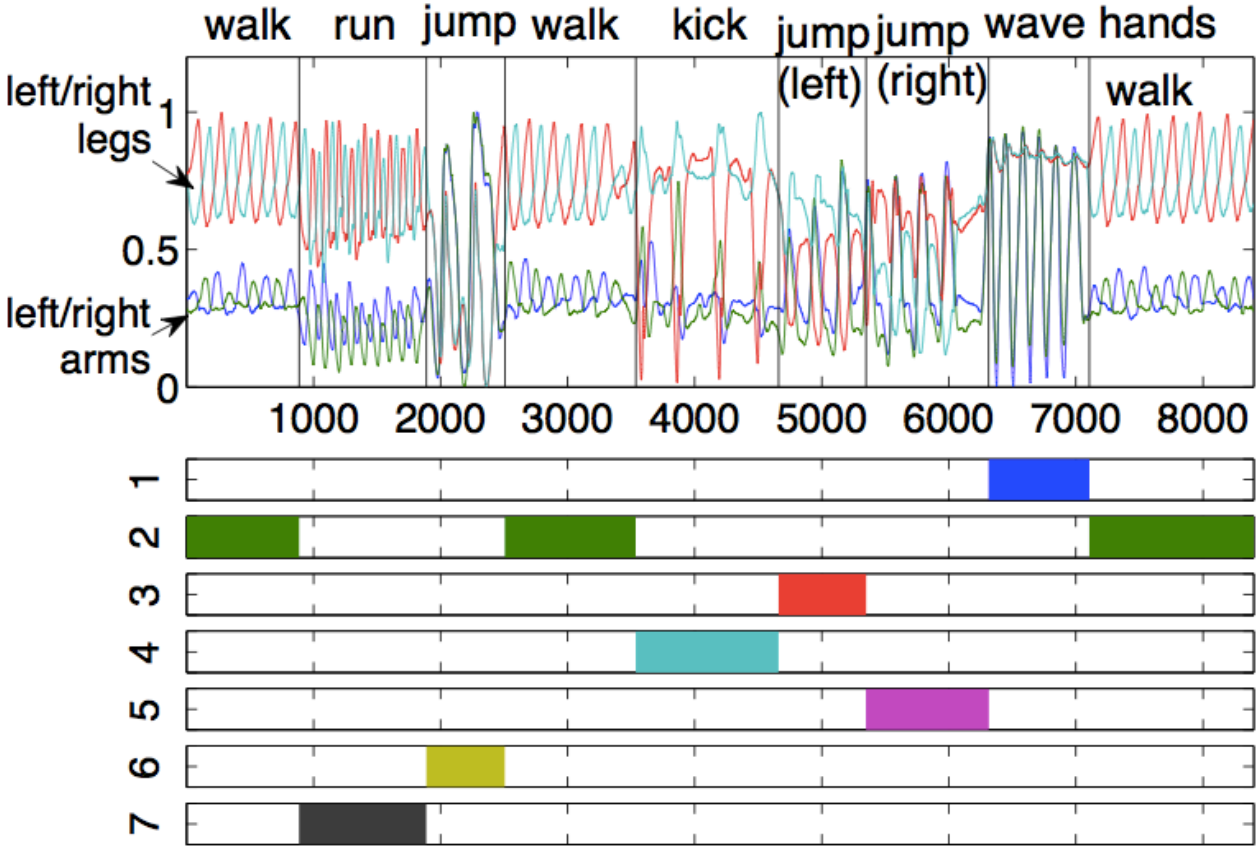
DynaMMo (Li et al., KDD'09)



pHMM (Wang et al., SIGMOD'11)

Sense-making

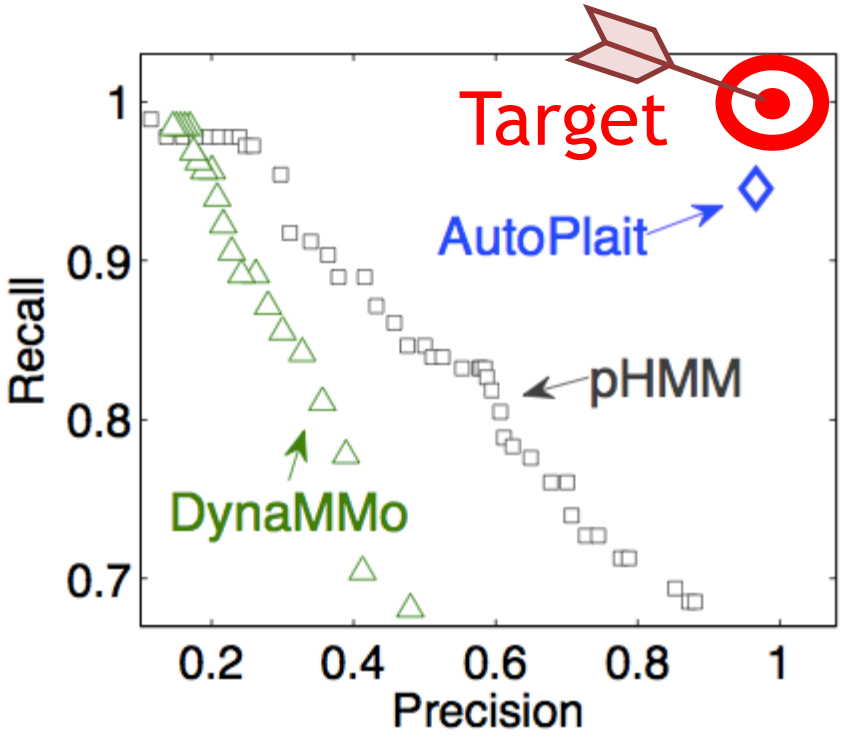
MoCap data



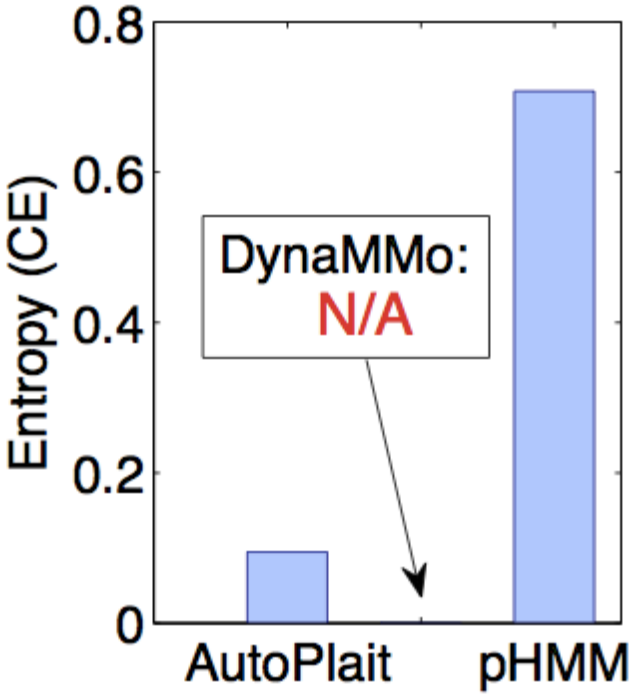
AutoPlait (NO magic numbers)

Q2. Accuracy

(a) Segmentation



(b) Clustering



(a) Precision and recall (higher is better)

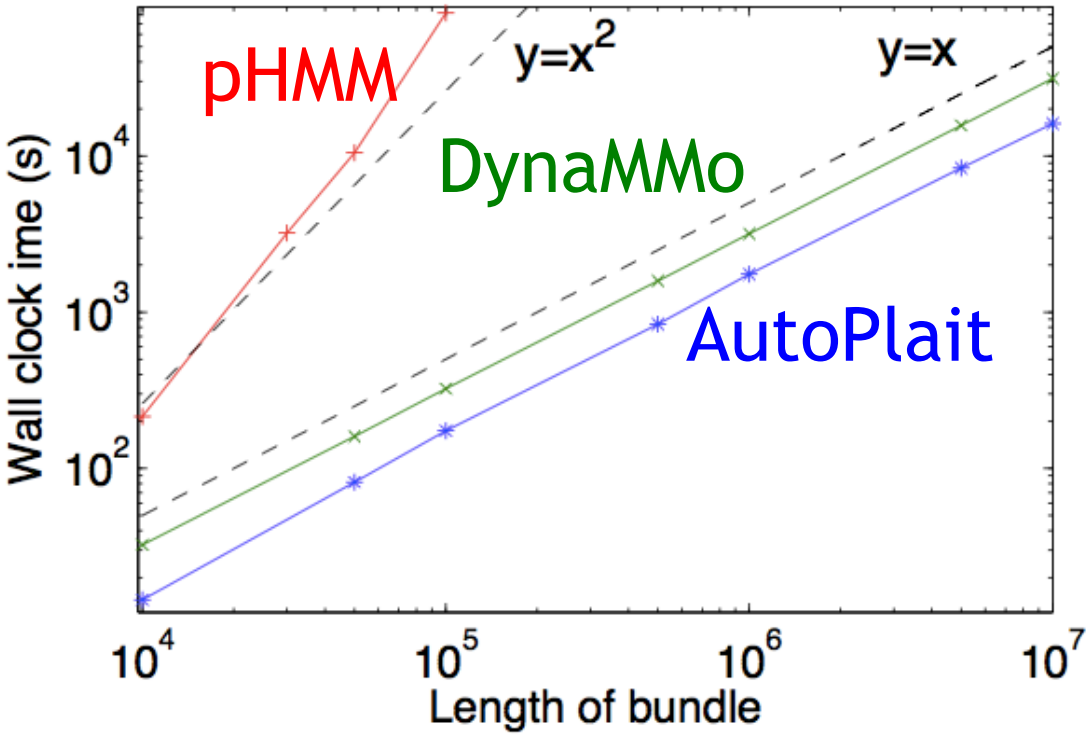
(b) CE score (lower is better)

AutoPlait needs "no magic numbers"

Q3. Scalability

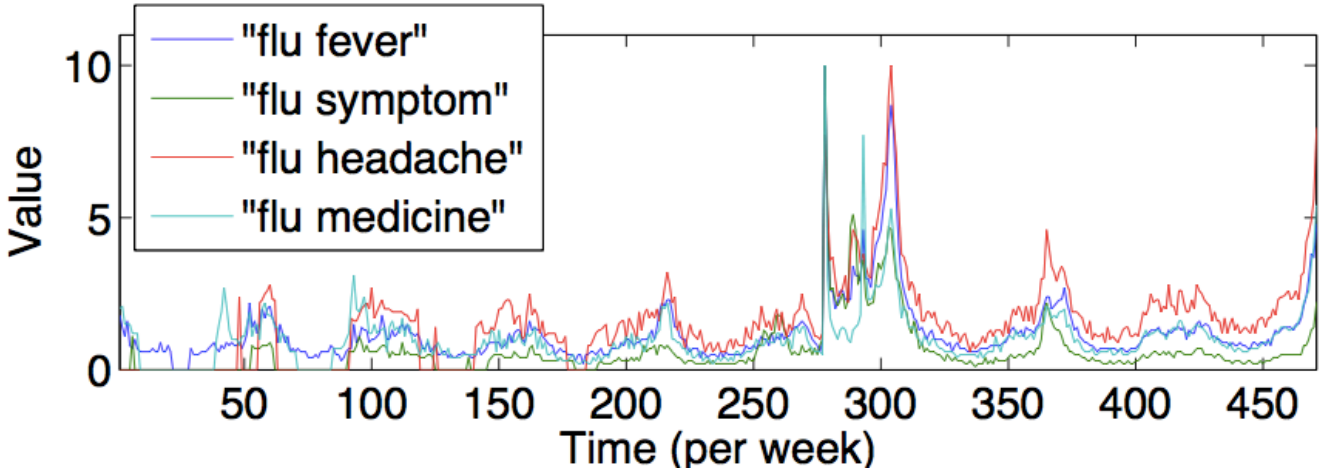
Wall clock time vs. data size (length) : n

AutoPlait scales linearly, i.e., O(n)



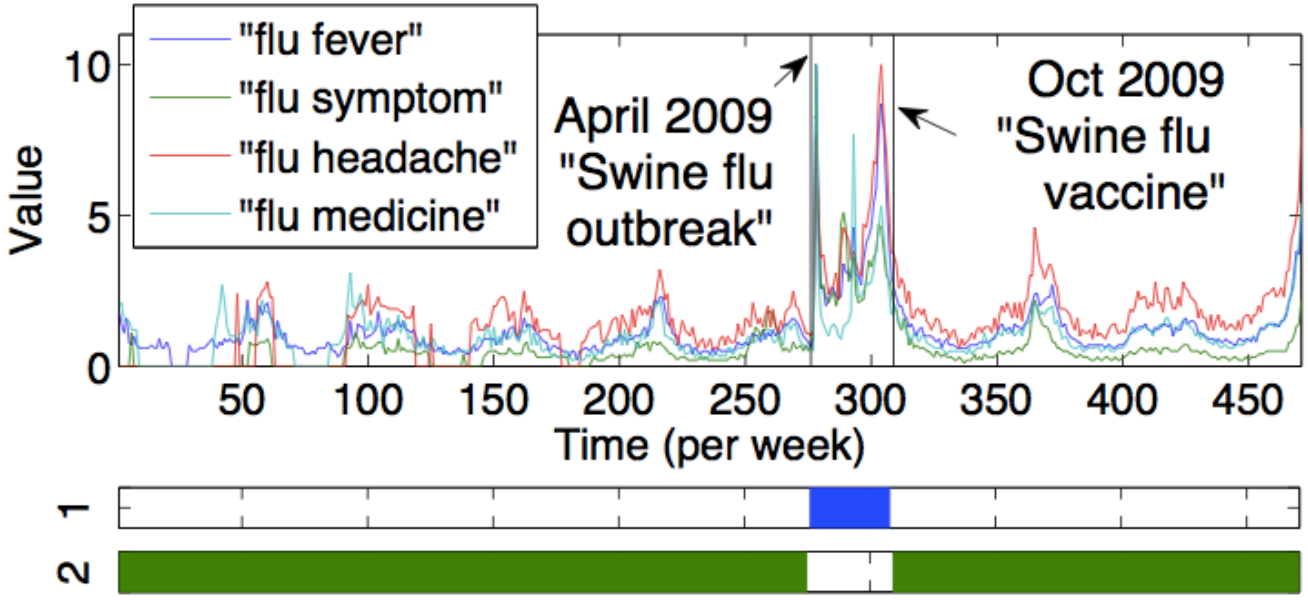
App. Event discovery (GoogleTrend)

Anomaly detection (flu-related topics,10 years)



App. Event discovery (GoogleTrend)

Anomaly detection (flu-related topics,10 years)

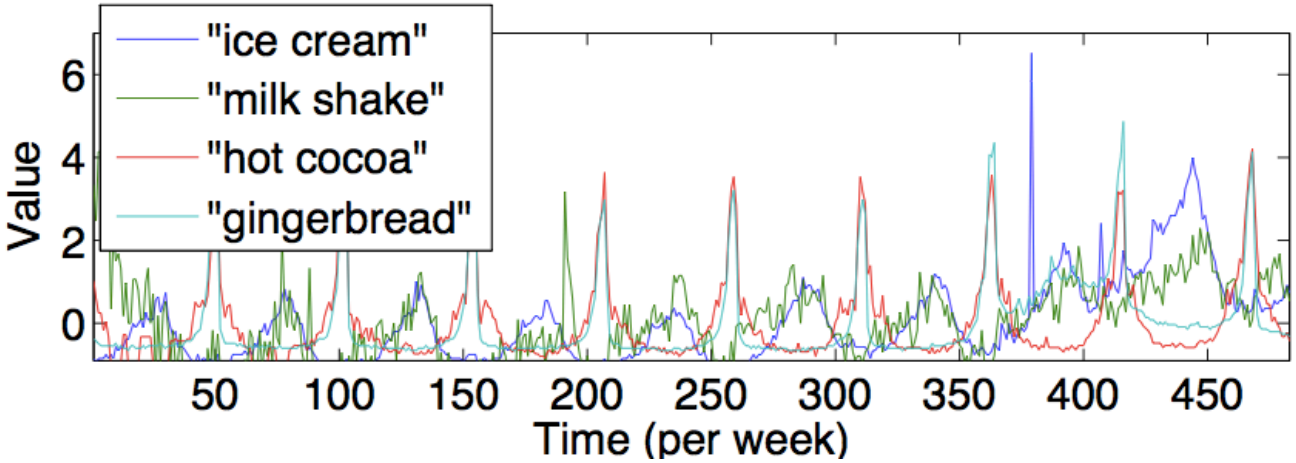


(a) Flu-related topics (regimes $r = 2$)

AutoPlait detects 1 unusual spike in 2009 (i.e., swine flu)

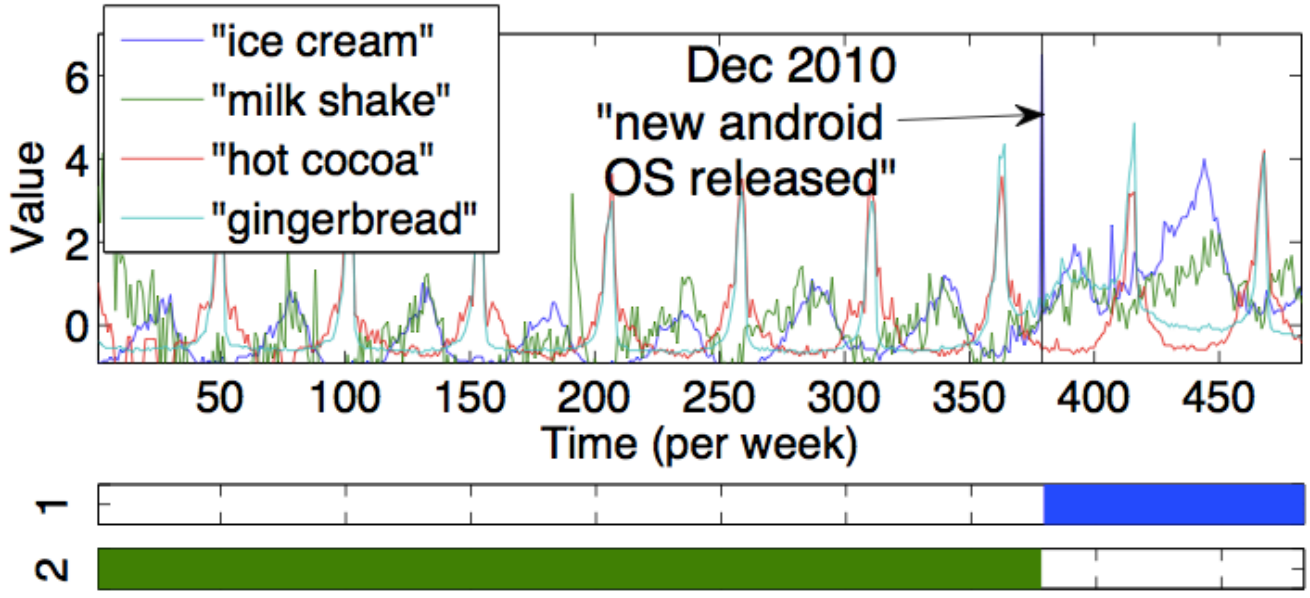
App. Event discovery (GoogleTrend)

Turning point detection (seasonal sweets topics)



App. Event discovery (GoogleTrend)

Turning point detection (seasonal sweets topics)

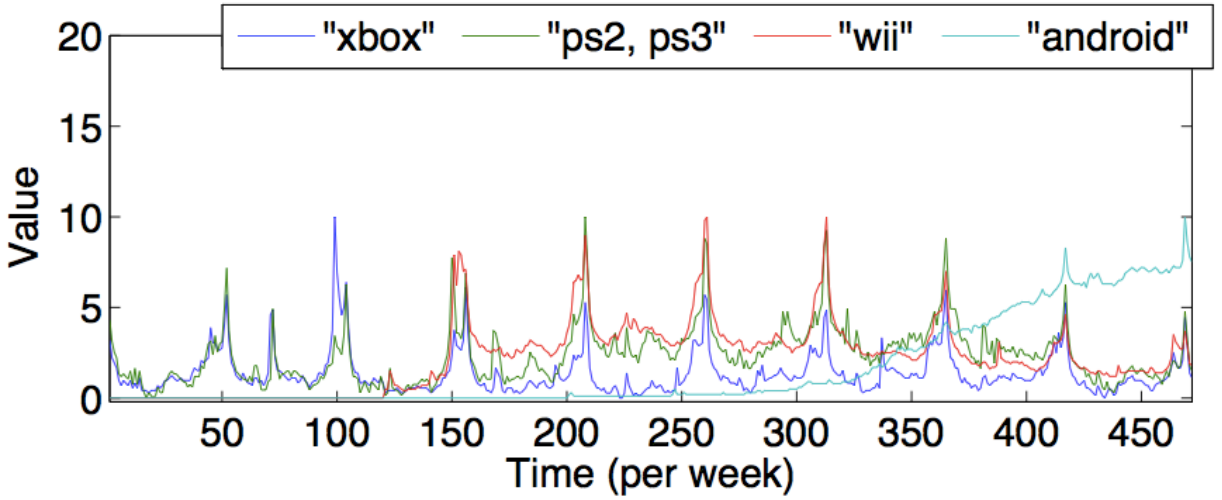


(b) Seasonal sweets topics (regimes $r = 2$)

Trend suddenly changed in 2010 (release of android OS "Ginger bread", "Ice Cream Sandwich")

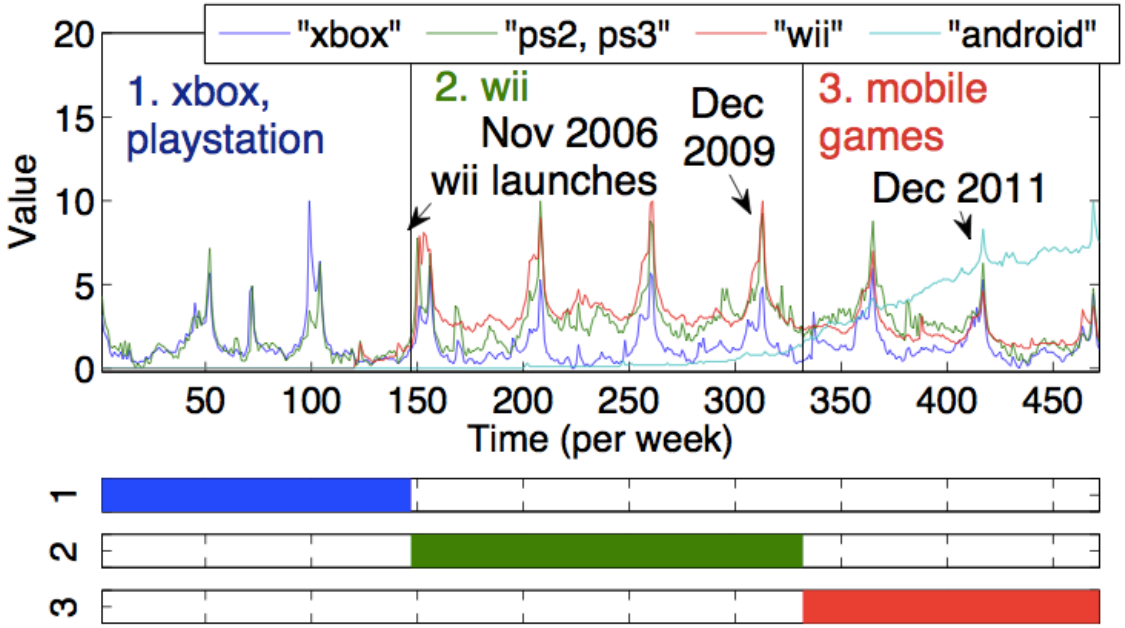
App. Event discovery (GoogleTrend)

Trend discovery (game-related topics)



App. Event discovery (GoogleTrend)

Trend discovery (game-related topics)



(c) Game-related topics (regimes $r = 3$)

It discovers 3 phases of "game console war" (Xbox&PlayStation/Wii/Mobile social games)

Industrial contribution

- Automobile sensor data
  - location, velocity, longitudinal/lateral acceleration

Code at

- http://www.cs.kumamoto-u.ac.jp/~yasuko/software.html

Part 1 – Conclusions

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Part 1 – Conclusions

- Motivation
- Similarity Search and Indexing
  - Euclidean/time-warping
  - extract features
  - index (SAM, R-tree)
- Feature extraction
  - SVD, ICA, DFT, DWT (multi-scale windows)

Part 1 – Conclusions

- Linear forecasting
  - AR, RLS
- Streaming pattern discovery
  - RLS, "incremental" wavelet transform
  - Multi-scale windows
- Automatic mining
  - MDL

References

- Yunyue Zhu, Dennis Shasha ``*StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time*'' VLDB, August, 2002. pp. 358-369.
- Spiros Papadimitriou, Jimeng Sun, Christos Faloutsos. *Streaming Pattern Discovery in Multiple Time-Series*. VLDB 2005.
- Yasushi Sakurai, Spiros Papadimitriou, Christos Faloutsos. *BRAID: Stream Mining through Group Lag Correlations*. SIGMOD 2005.
- Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, Martin Strauss. *Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries*. VLDB 2001.
- Sudipto Guha, Nick Koudas, Amit Marathe, Divesh Srivastava. *Merging the Results of Approximate Match Operations*. VLDB 2004.
- Anna C. Gilbert, Sudipto Guha, Piotr Indyk, S. Muthukrishnan, Martin Strauss. *Near-optimal sparse fourier representations via sampling*. STOC 2002.

References

- Nick Koudas, Beng Chin Ooi, Kian-Lee Tan, Rui Zhang. *Approximate NN queries on Streams with Guaranteed Error/performance Bounds*. VLDB 2004.
- Flip Korn, S. Muthukrishnan, Divesh Srivastava. *Reverse Nearest Neighbor Aggregates Over Data Streams*. VLDB 2002.
- Sudipto Guha, Nick Koudas, Kyuseok Shim. *Data-streams and histograms*. STOC 2001.
- Sudipto Guha, Nina Mishra, Rajeev Motwani, Liadan O'Callaghan. *Clustering Data Streams*. FOCS 2000.
- Charu C. Aggarwal, Jiawei Han, Jianyong Wang, Philip S. Yu. *A Framework for Clustering Evolving Data Streams*. VLDB 2003.

References

- Piotr Indyk, Nick Koudas, S. Muthukrishnan. *Identifying Representative Trends in Massive Time Series Data Sets Using Sketches*. VLDB 2000.
- Graham Cormode, S. Muthukrishnan. *An improved data stream summary: the count-min sketch and its applications*. J. Algorithms 55 (1), 2005.
- Graham Cormode, Flip Korn, S. Muthukrishnan, Divesh Srivastava. *Finding Hierarchical Heavy Hitters in Data Streams*. VLDB 2003.
- Piotr Indyk. *Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation*. FOCS 2000.
- Yunyue Zhu, Dennis Shasha. *Efficient elastic burst detection in data streams*. KDD 2003.

References

- Eamonn J. Keogh, Selina Chu, David M. Hart, Michael J. Pazzani. *An Online Algorithm for Segmenting Time Series*. ICDM 2001.
- Spiros Papadimitriou, Philip S. Yu. *Optimal multi-scale patterns in time series streams*. SIGMOD 2006.
- Flip Korn, S. Muthukrishnan, Yihua Wu. *Modeling skew in data streams*. SIGMOD 2006.
- Yasushi Sakurai, Christos Faloutsos, Masashi Yamamuro. *Stream Monitoring under the Time Warping Distance*. ICDE 2007.

Part 1

Similarity search, pattern discovery and summarization

Yasushi Sakurai (Kumamoto University)
Yasuko Matsubara (Kumamoto University)
Christos Faloutsos (Carnegie Mellon University)