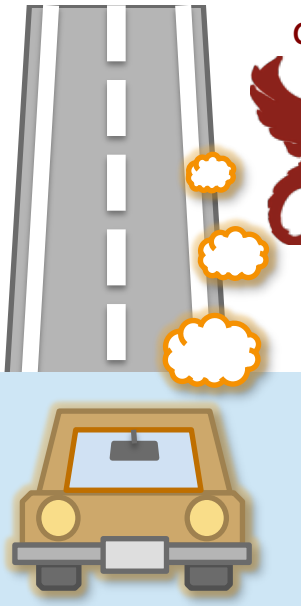
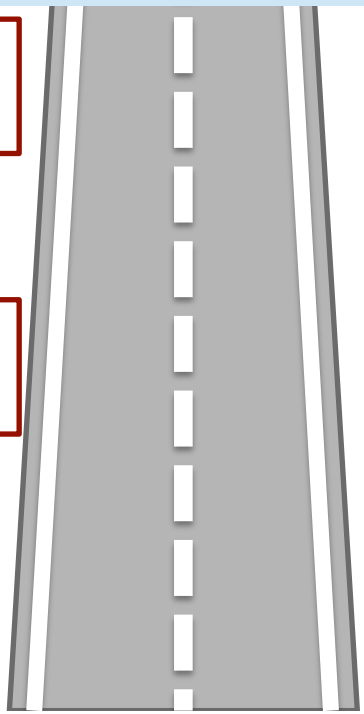
Mining and Forecasting of Big Time-series Data

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

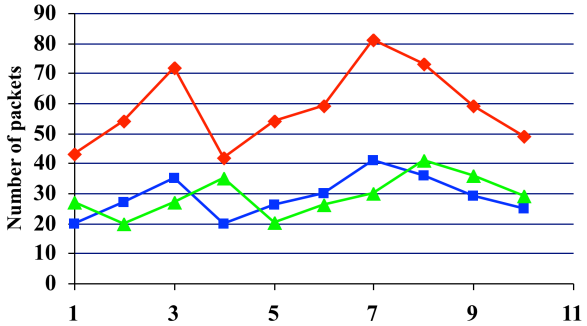
Problem definition

- Given: one or more sequences

  $x_1, x_2, \ldots, x_t, \ldots$

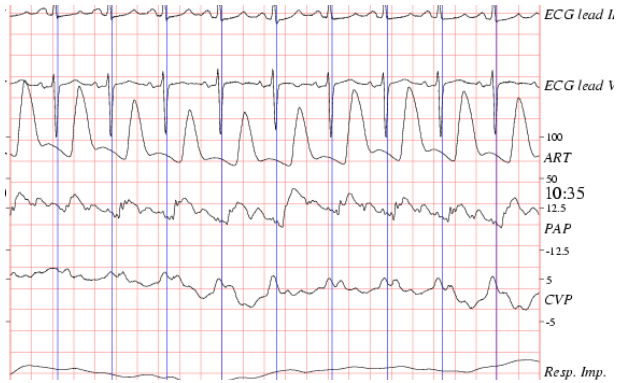
  $(y_1, y_2, \ldots, y_t, \ldots)$

- Find
  - similar sequences; forecasts
  - patterns; clusters; outliers

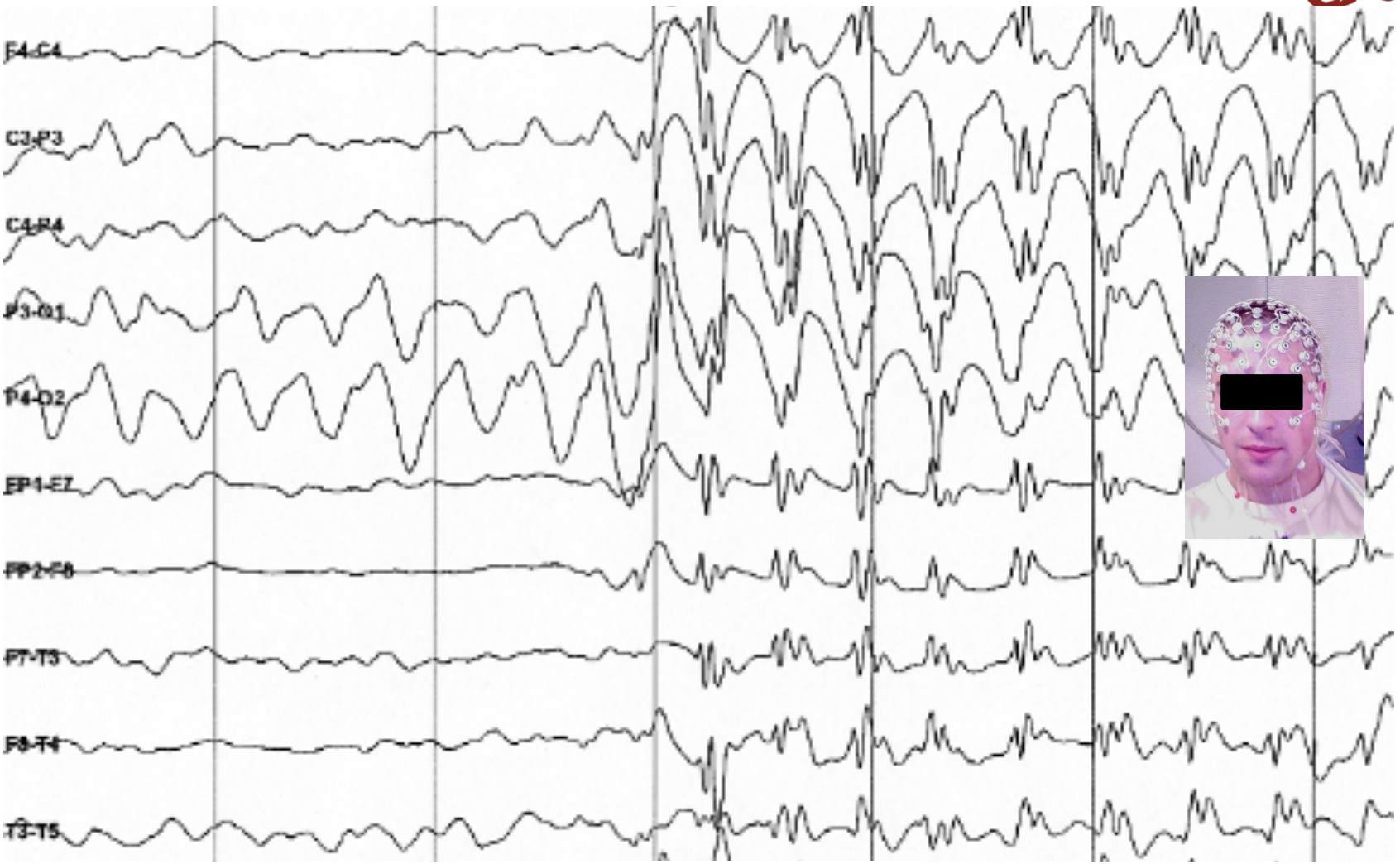


Motivation - Applications

- Financial, sales, economic series
- Medical
  - reactions to new drugs
  - elderly care
  - ECG ('physionet.org')



EEG - epilepsy



from wikipedia

Motivation - Applications (cont'd)

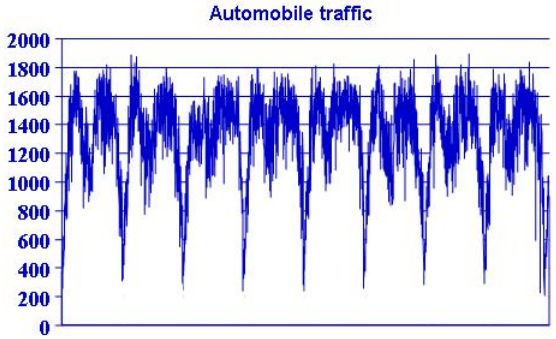
- 'Smart house'
  - sensors monitor temperature, humidity, air quality
- Video surveillance

Motivation - Applications (cont'd)

- Civil/automobile infrastructure
  - bridge vibrations [Oppenheim+02]

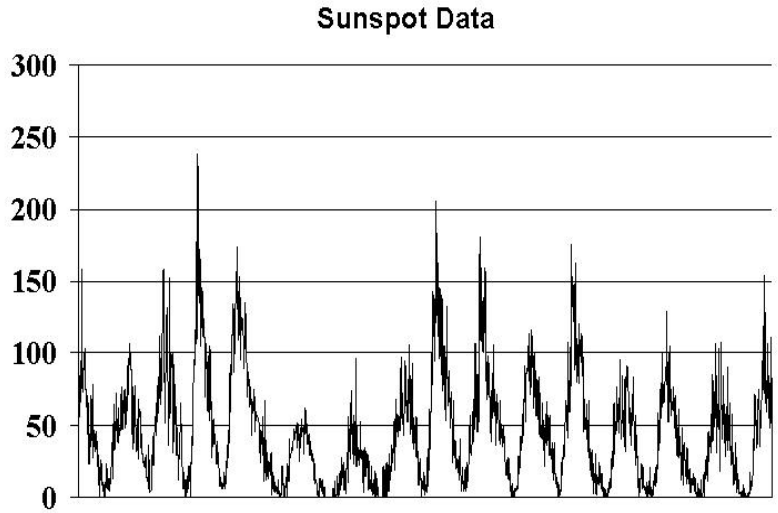
Tokyo Gate Bridge

  - road conditions / traffic monitoring



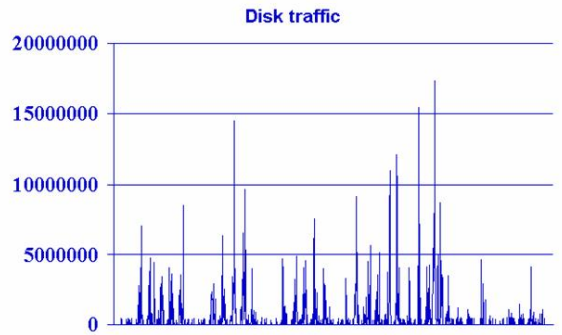
Motivation - Applications (cont'd)

- Weather, environment/anti-pollution
  - volcano monitoring
  - air/water pollutant monitoring



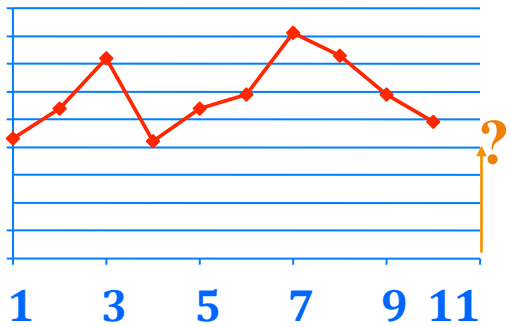
Motivation - Applications (cont'd)

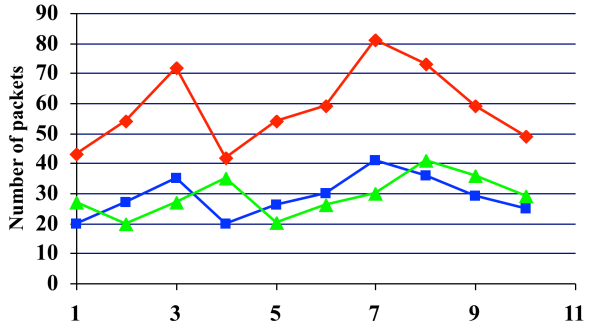
- Computer systems
  - 'Active Disks' (buffering, prefetching)
  - web servers (ditto)
  - network traffic monitoring
  - ...



Wish list

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



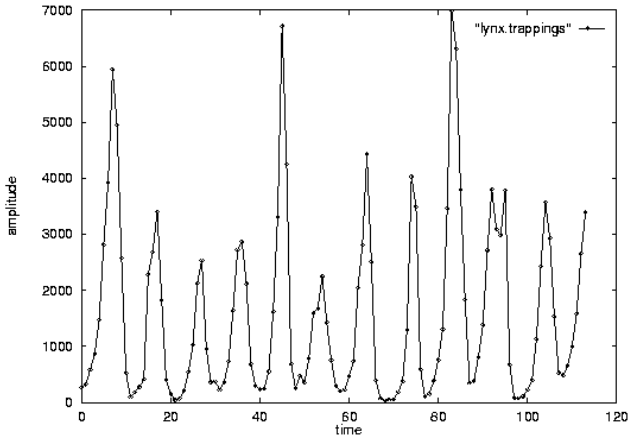


Problem #1:

Goal: given a signal (eg., #packets over time)

Find: patterns, periodicities, and/or compress

count

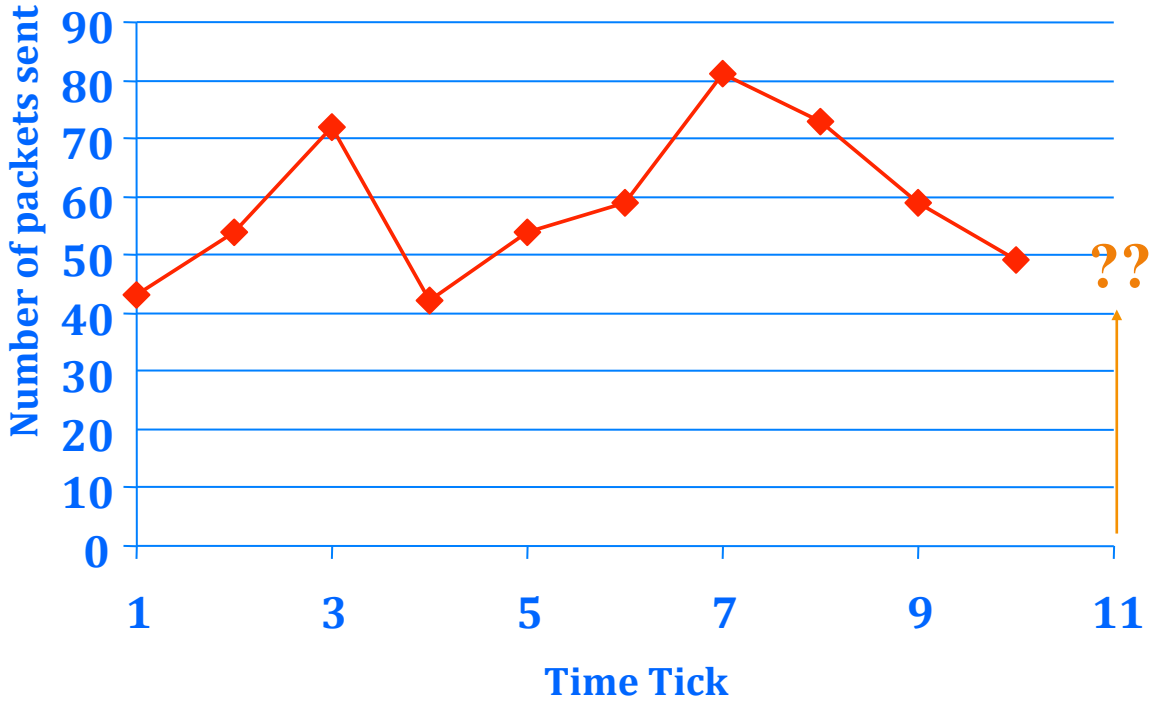


year

lynx caught per year
(packets per day;
temperature per day)

Problem#2: Forecast

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

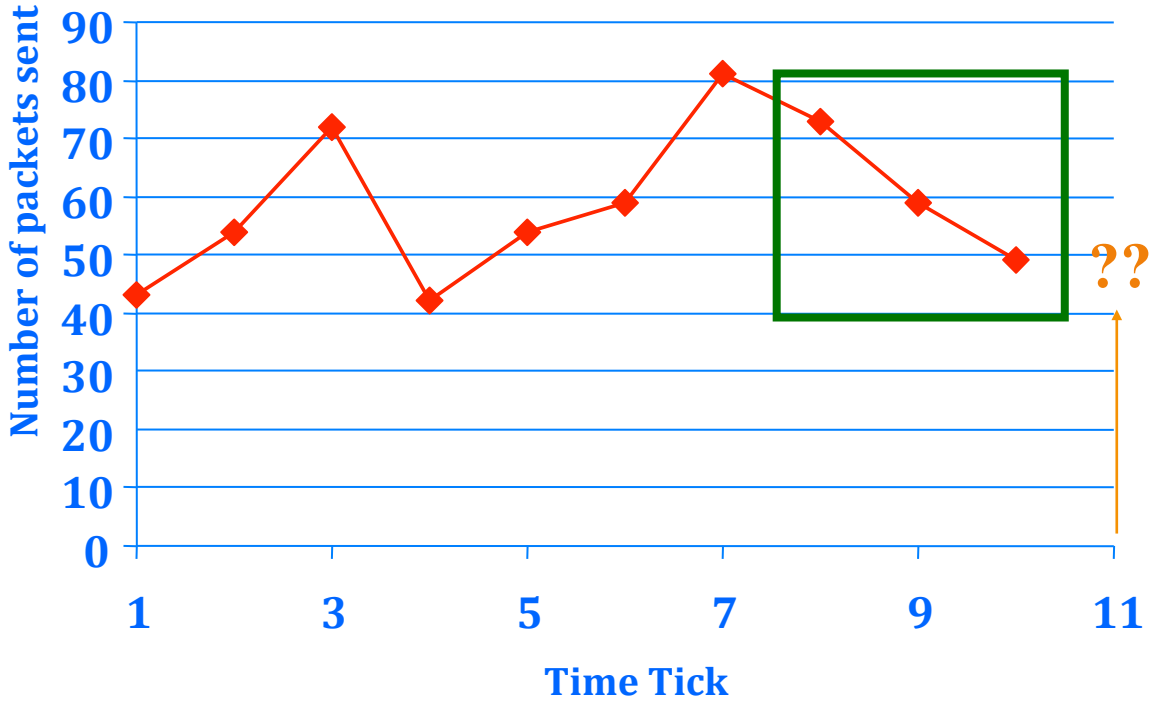


Kumamoto U

CMU CS

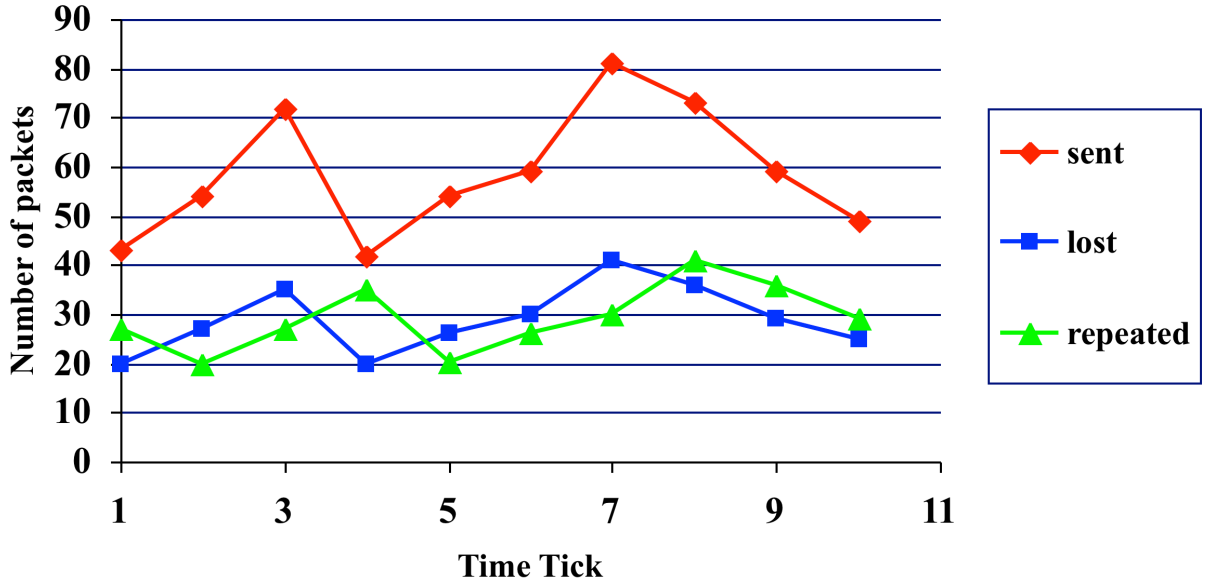
Problem#2' : Similarity search

Eg., Find a 3-tick pattern, similar to the last one



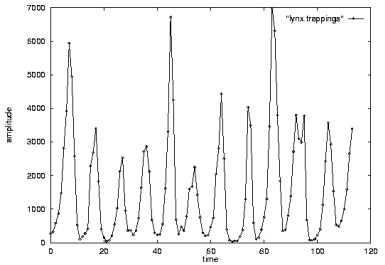
Problem #3:

- Given: A set of **correlated** time sequences
- Forecast ‘Sent(t)’



Important observations

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



- For forecasting, we need
  - patterns/rules
  - 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

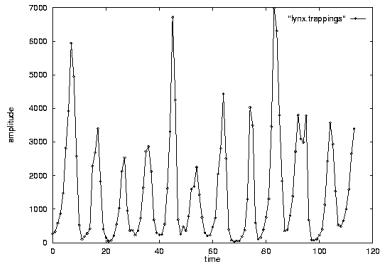
Importance of distance functions

Subtle, but **absolutely necessary**:

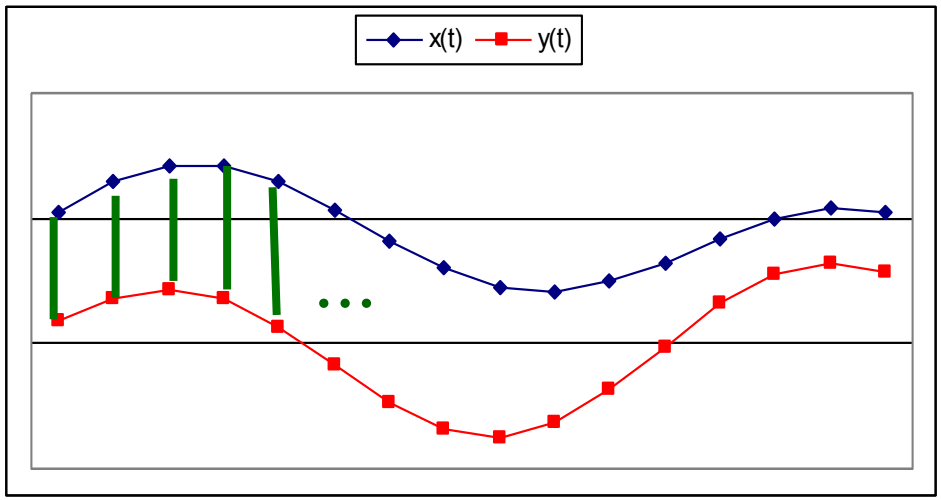
- A 'must' for similarity indexing
  - (-> forecasting)
- A 'must' for clustering

Two major families

- Euclidean and Lp norms
- time warping and variations



Euclidean and Lp

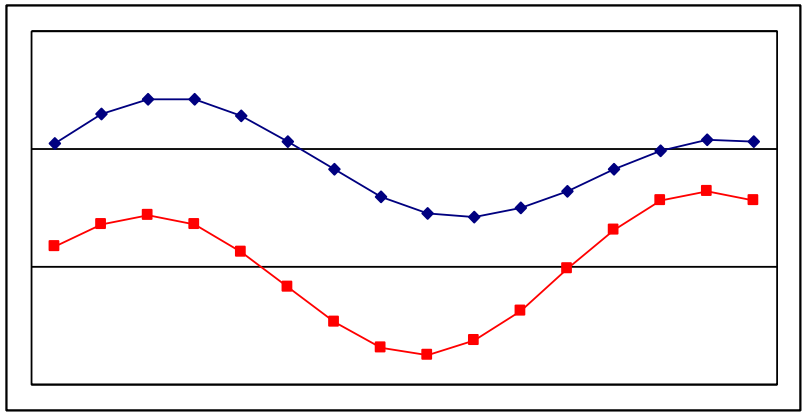


$$D(\vec{x},\vec{y}) = \sum_{i=1}^{n} (x_i - y_i)^2$$

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

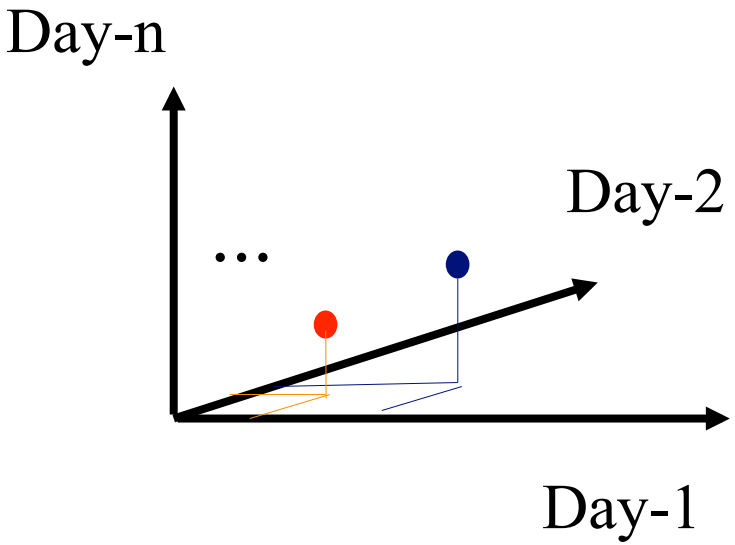
- L_1: city-block = Manhattan
- L_2 = Euclidean
- L_∞

Observation #1



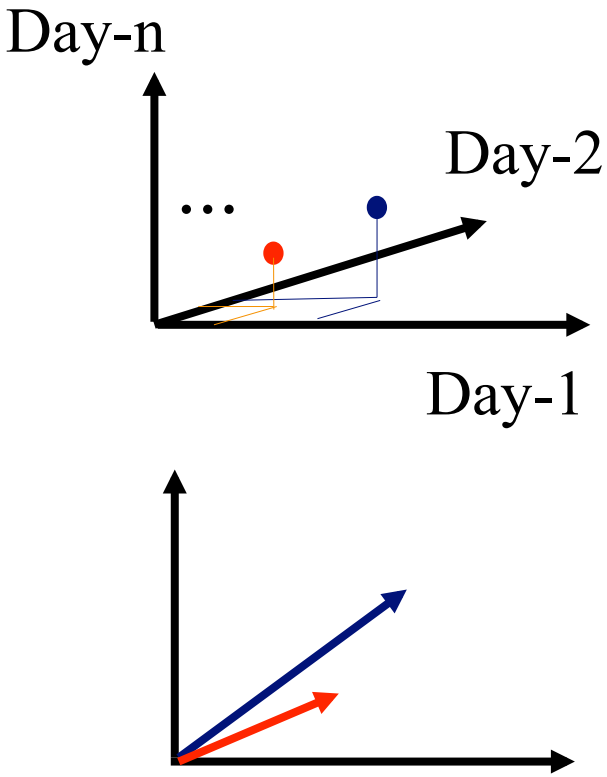
- **Time sequence**

  -> n-d vector



Observation #2

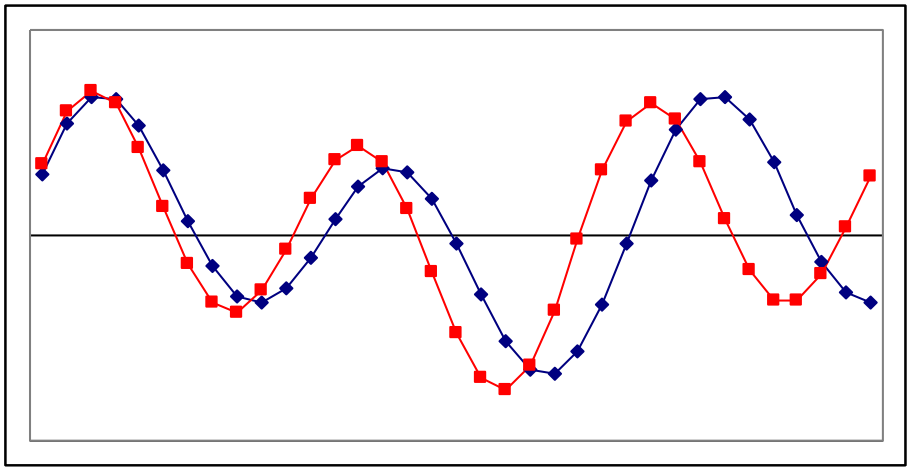
- Euclidean distance is closely related to
  - cosine similarity
  - dot product
  - 'cross-correlation' function



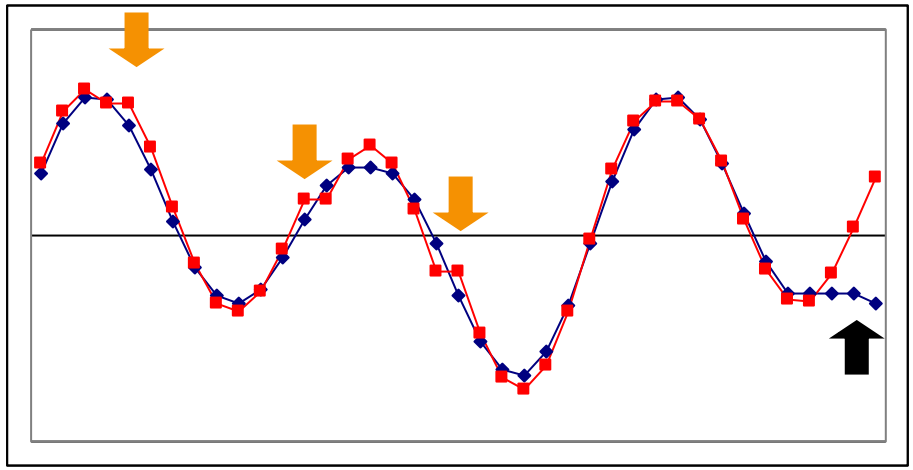
Time Warping

- 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



‘stutters’:



Time Warping DETAILS

Q: how to compute it?

A: dynamic programming

$D(i, j)$ = cost to match prefix of length i of first sequence x with prefix of length j of second sequence y

Time Warping DETAILS

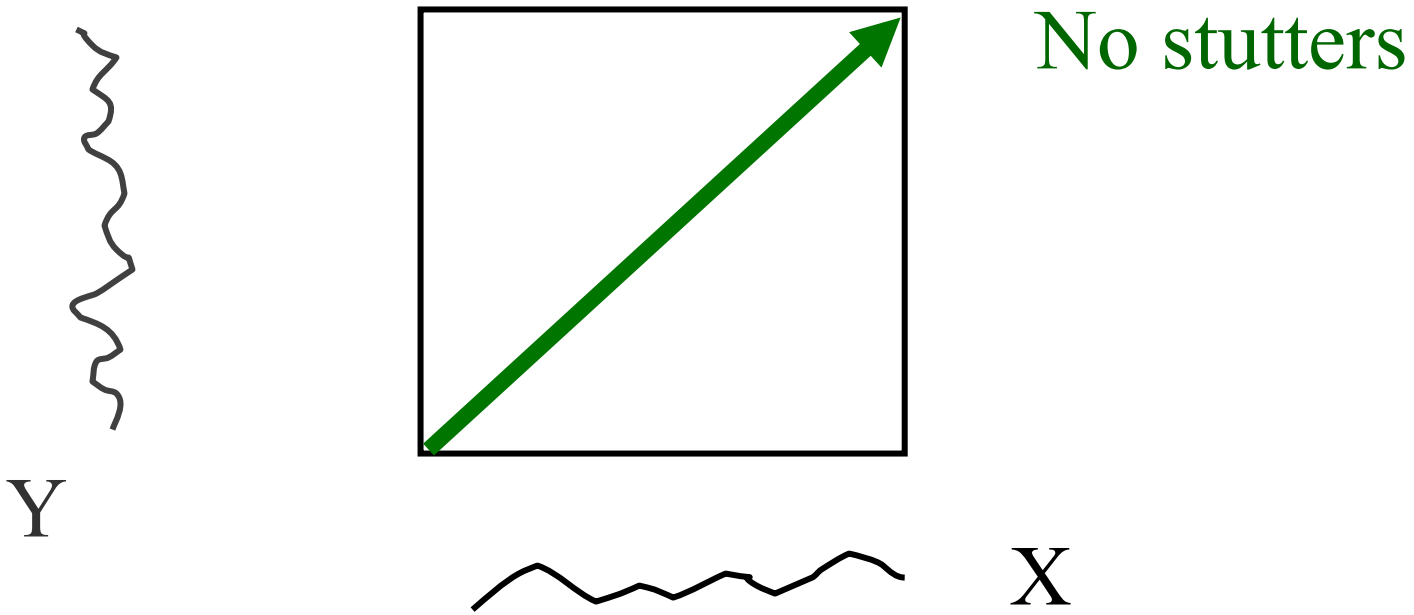
Thus, with no penalty for stutter, for sequences

$$x_1, x_2, \ldots, x_{i,;} \qquad y_1, y_2, \ldots, y_j$$

$$D(i,j) = \|x[i] - y[j]\| + \min \begin{cases} D(i-1, j-1) & \text{no stutter} \\ D(i, j-1) & \text{x-stutter} \\ D(i-1, j) & \text{y-stutter} \end{cases}$$

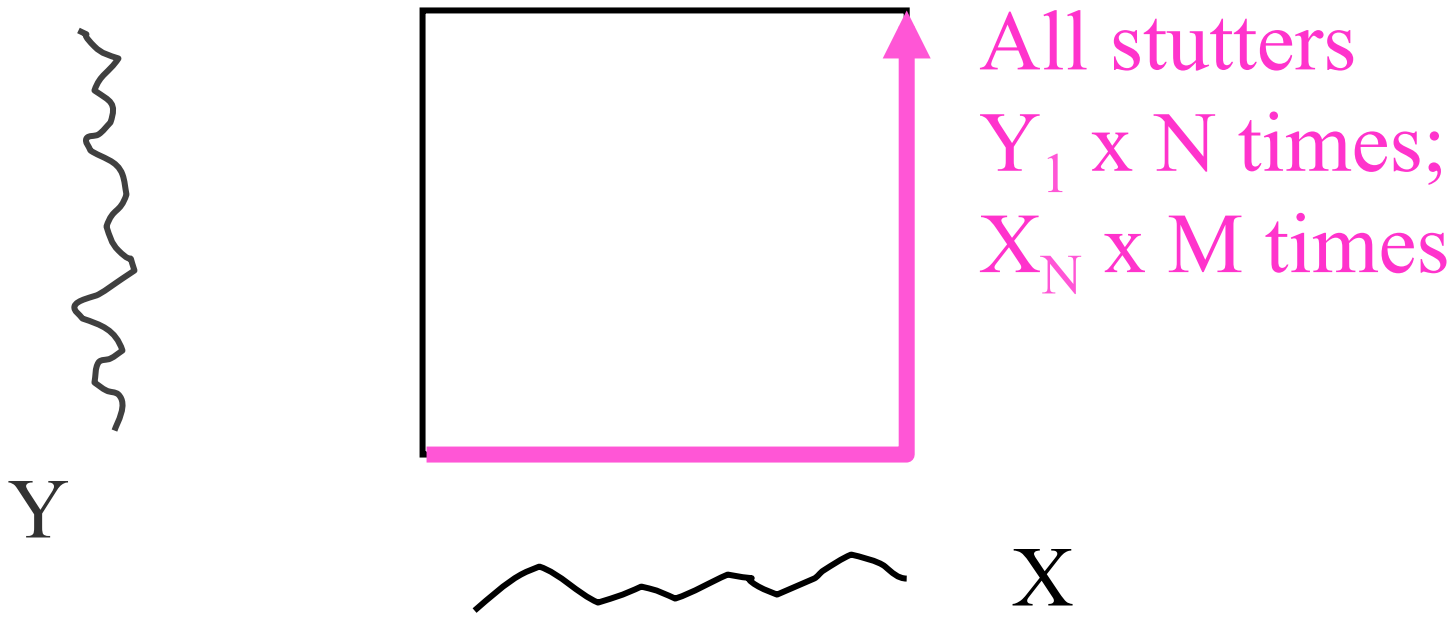
Time Warping

- Time warping matrix & optimal path:



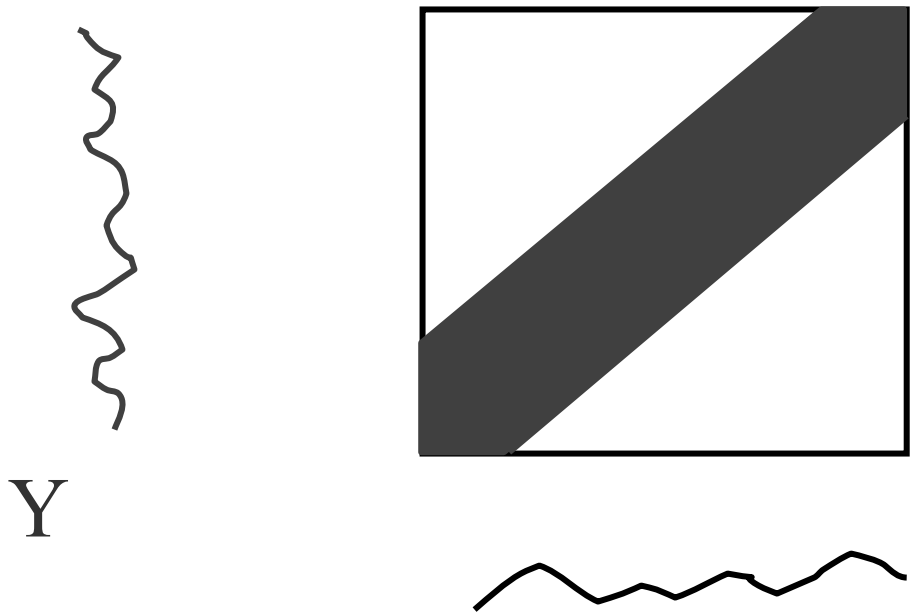
Time Warping

- Time warping matrix & optimal path:



Time Warping - variations

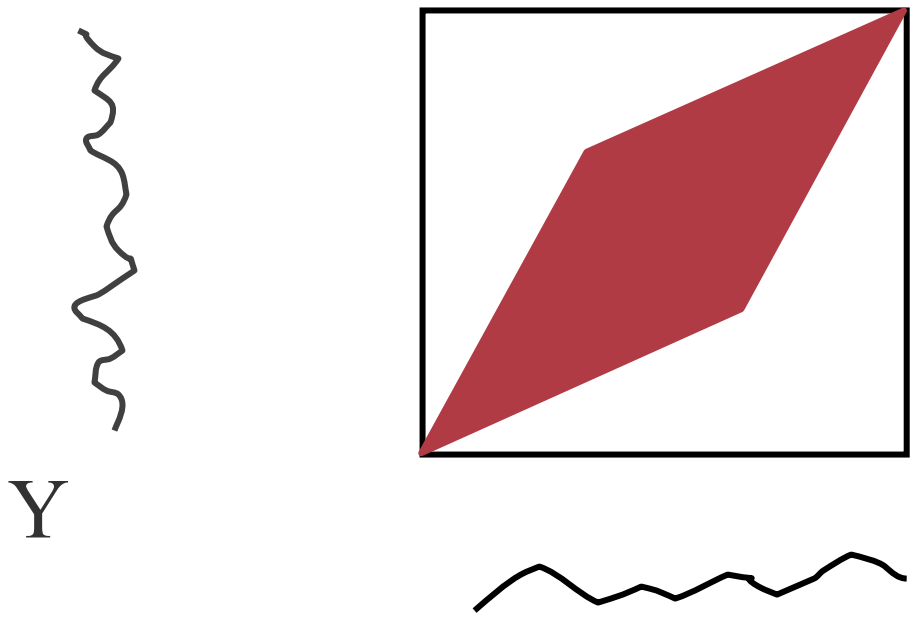
- Time warping matrix & optimal path:



At most k stutters: Sakoe-Chiba band

Time Warping - variations

- Time warping matrix & optimal path:

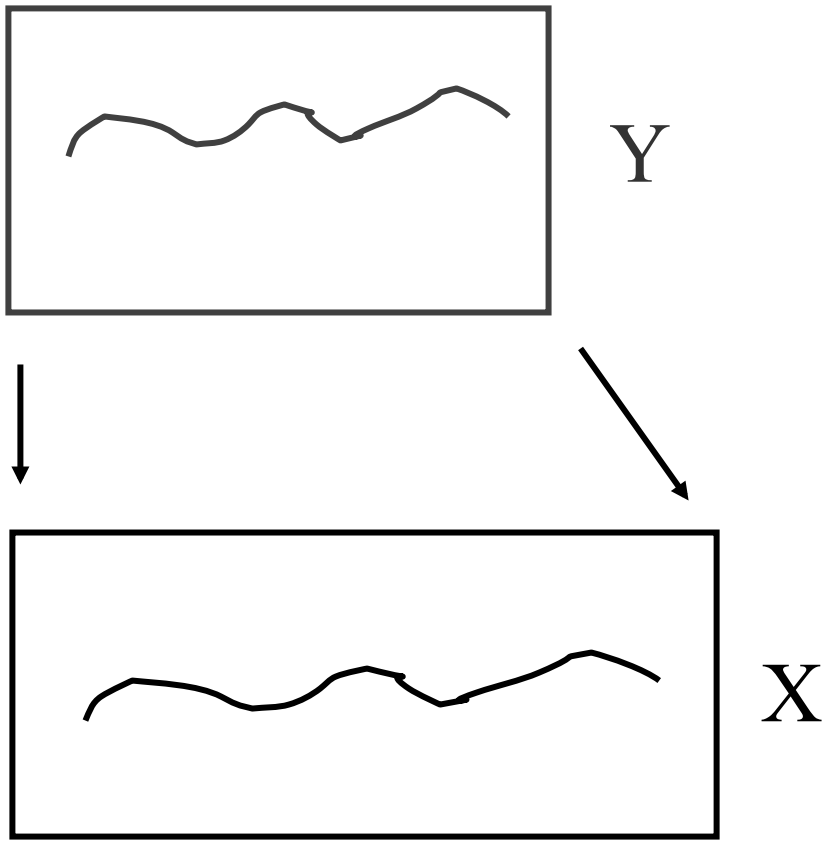


At most x% stutters: Itakura parallelogram

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]

More distance functions.

- 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

On The Marriage of Lp-norms and Edit Distance, Lei Chen, Raymond T. Ng:, VLDB'04

Towards Parameter-Free Data Mining, E. Keogh, S. Lonardi, C.A. Ratanamahatana, KDD'04

Conclusions

- Prevailing distances:
  - Euclidean and
  - time-warping

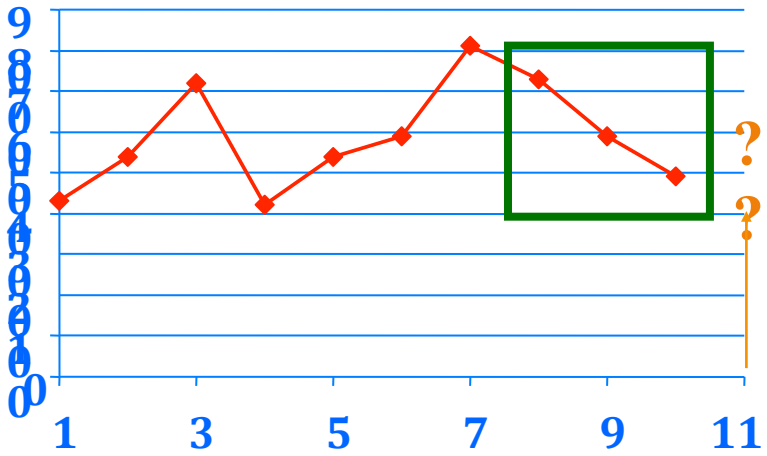
Roadmap

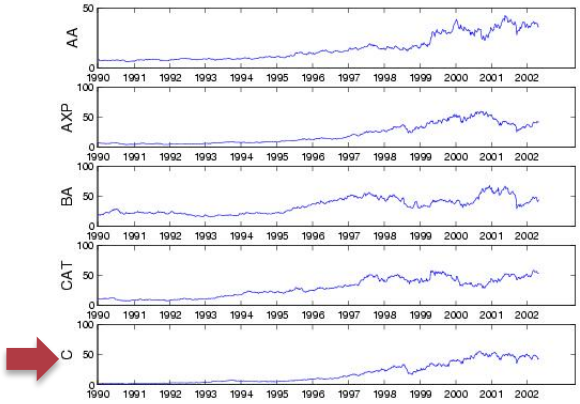
- Motivation
- Similarity Search and Indexing
  - distance functions: Euclidean;Time-warping
  - indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Indexing

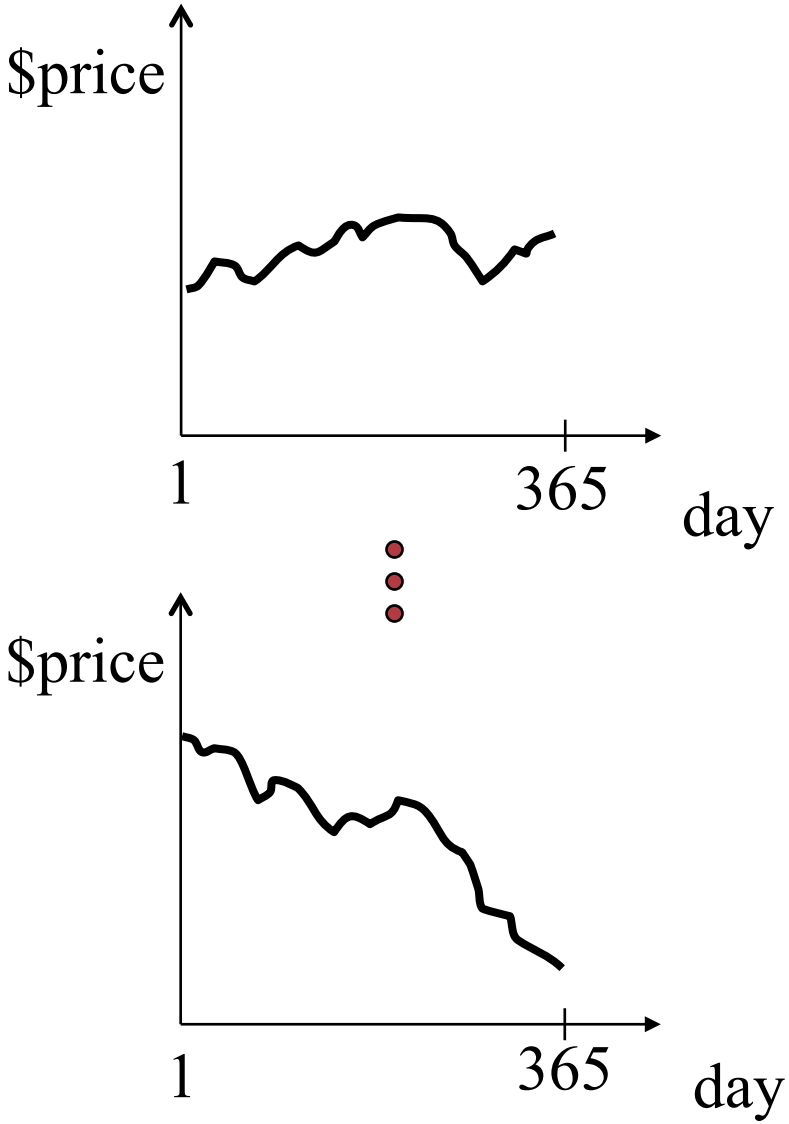
Problem 2':

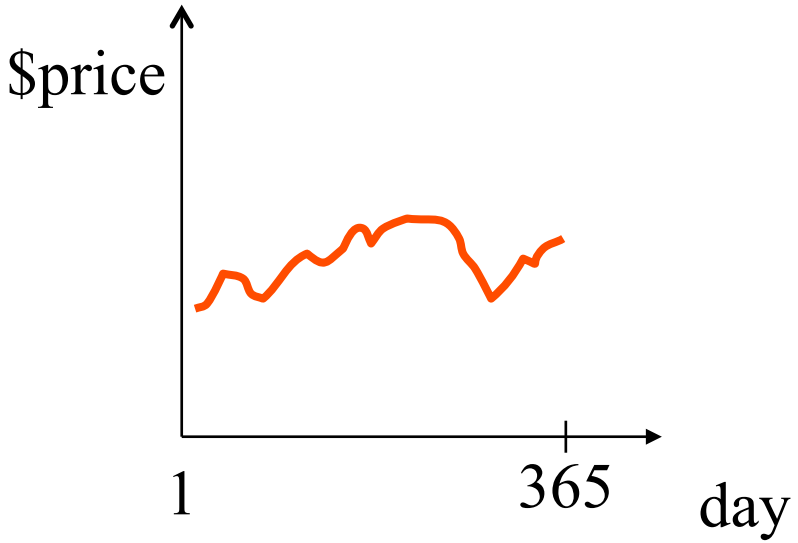
- given a set of time sequences,
- find the ones similar to a desirable query sequence





Indexing





distance function: by expert

(Euclidean; DTW; …)

Idea: ‘GEMINI’

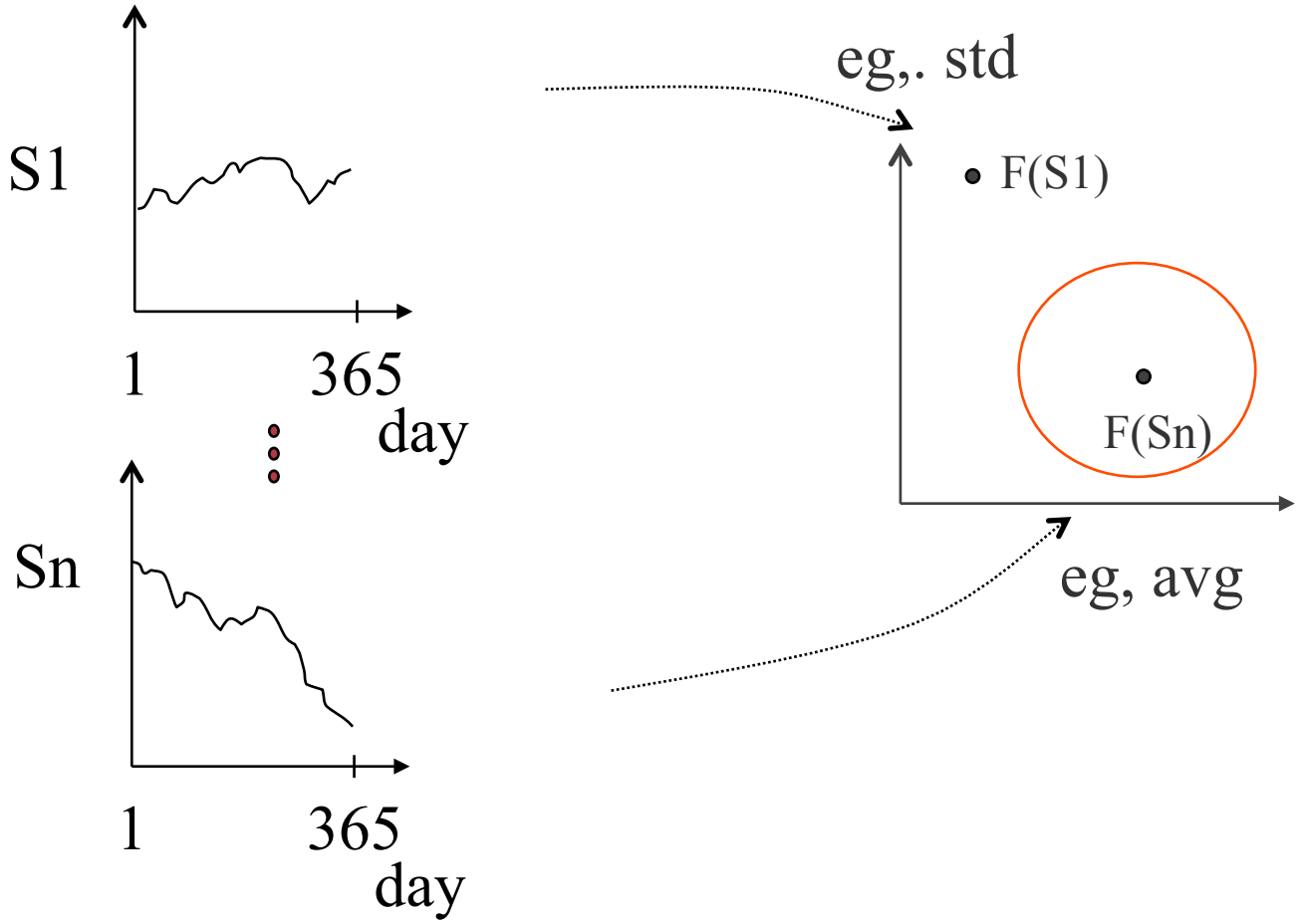
Eg., ‘*find stocks similar to MSFT*’

Seq. scanning: too slow

How to accelerate the search?

[Faloutsos96]

'GEMINI' - Pictorially



GEMINI

Solution: Quick-and-dirty' filter:

- extract *n* features (numbers, eg., avg., etc.)
- map into a point in *n*-d 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

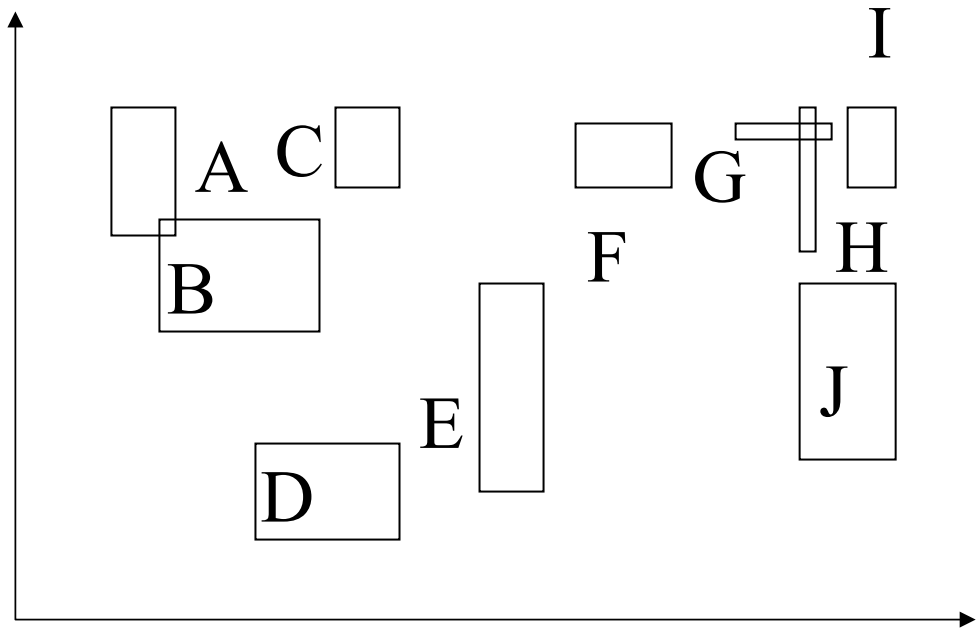
Q: How do Spatial Access Methods (SAMs) work?

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:

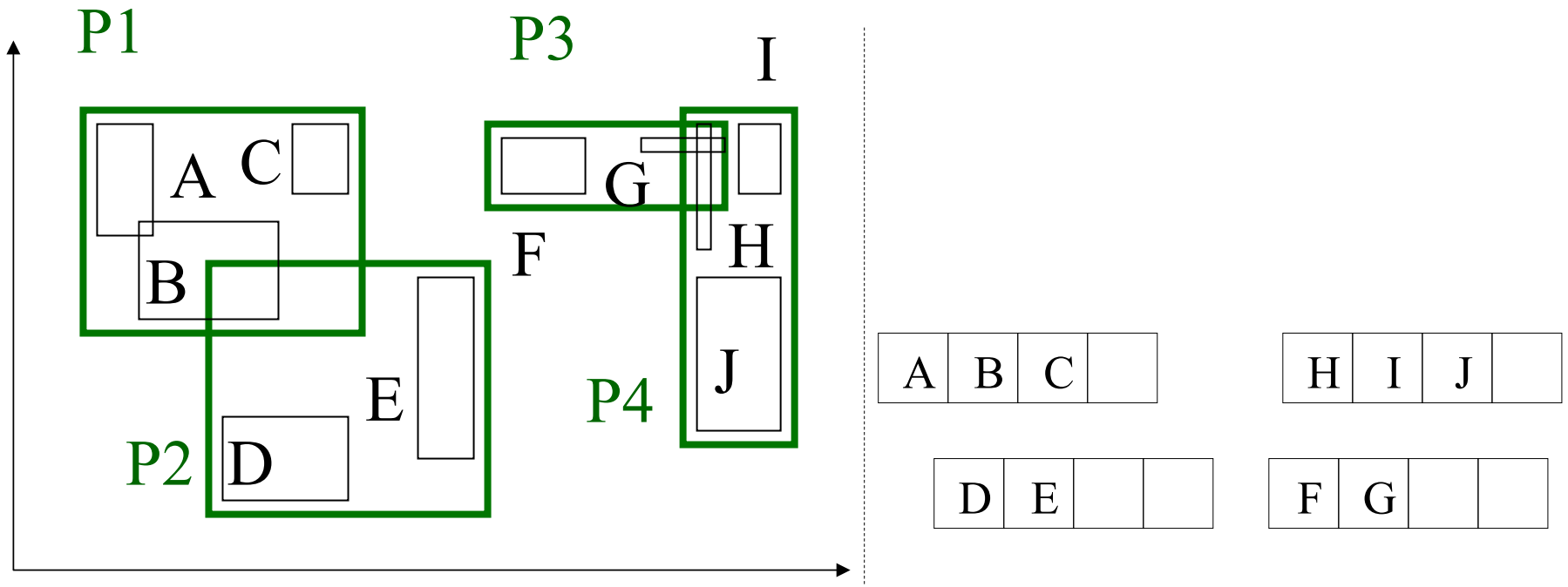
R-trees

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



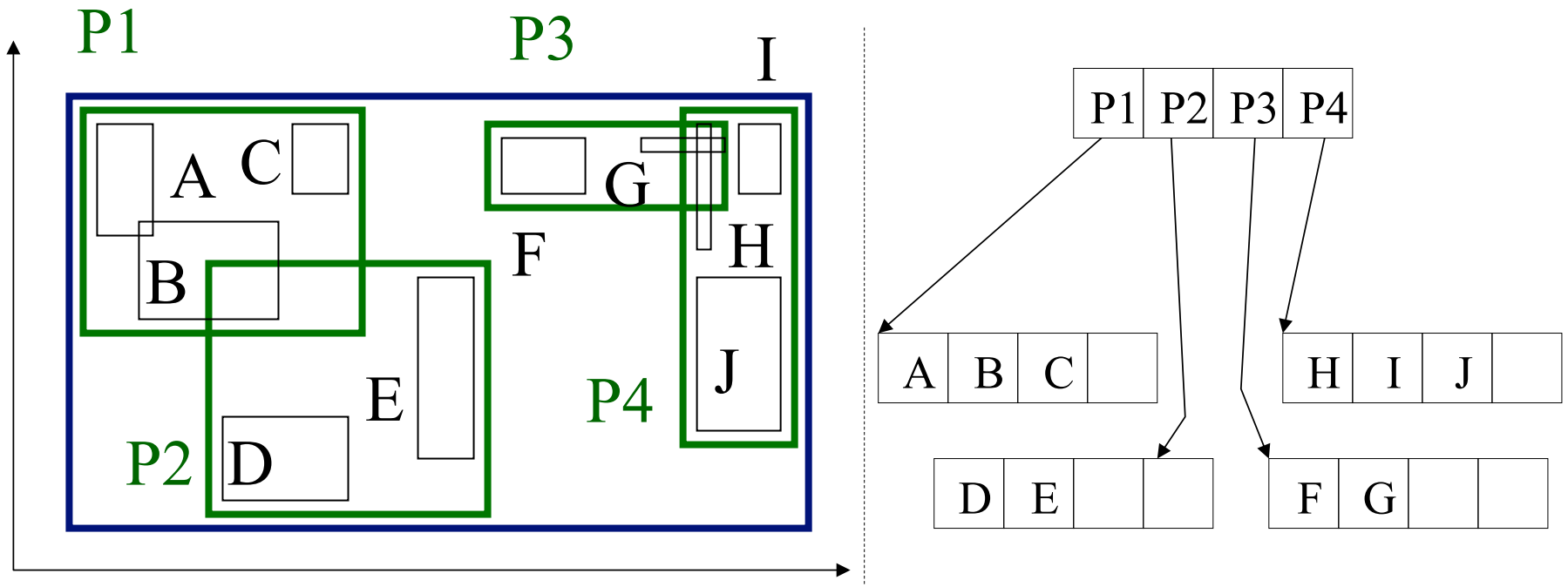
R-trees

- eg., w/ fanout 4:

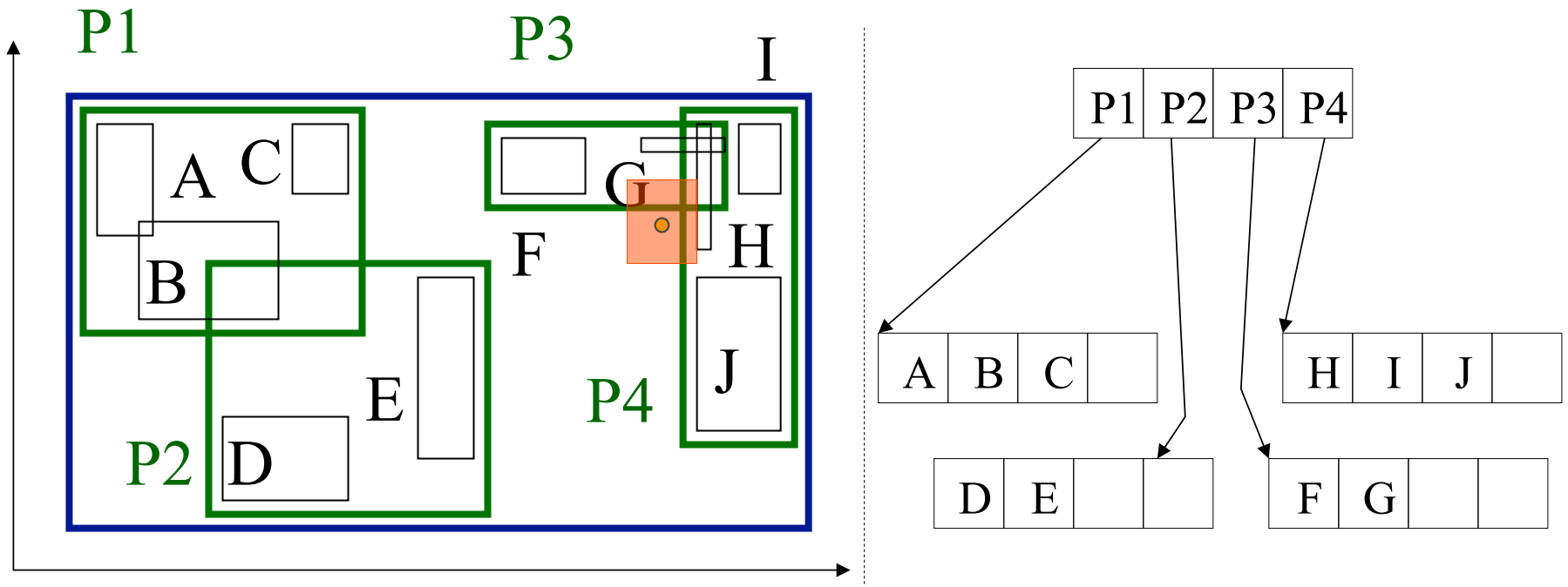


R-trees

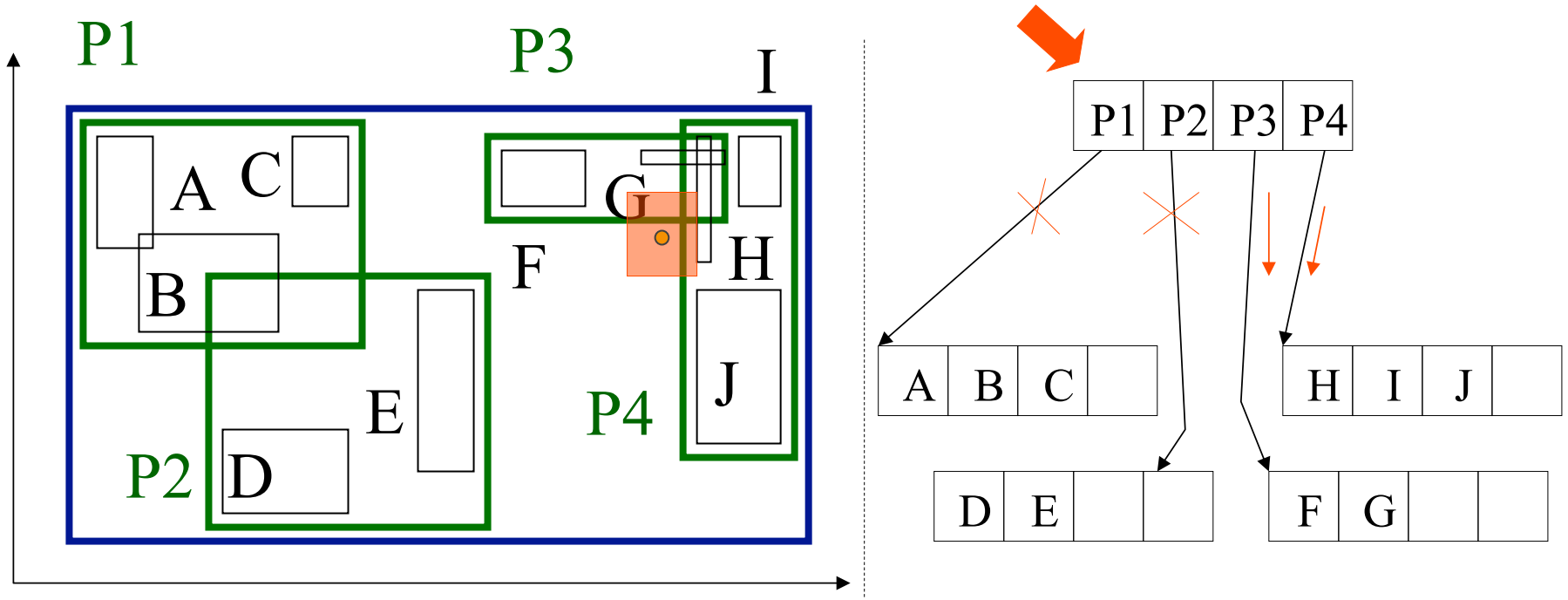
- eg., w/ fanout 4:



R-trees - range search?



R-trees - range search?



Kumamoto U

CMU CS

Recent work

- Rakthanmanon+ [kdd' 12]: EXCELLENT Software, the UCR Suite for ultrafast subsequence search
- Zoumpatianos+ [sigmod'14]: ADS+, exploratory analysis
- Camerra+ [KAIS' 14]: iSAX2+, indexing for bulk loading

Eamonn Keogh (UCR)

Searching and Mining Trillions of Time Series Subsequences under Dynamic Time Warping, T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, E. Keogh:,KDD' 12
Indexing for Interactive Exploration of Big Data Series, K. Zoumpatianos, S. Idreos, T. Palpanas:, SIGMOD' 14
Beyond One Billion Time Series: Indexing and Mining Very Large Time Series Collections with iSAX2+, A. Camerra, J. Shieh, T. Palpanas, T. Rakthanmanon, E. Keogh:, KAIS 2014

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

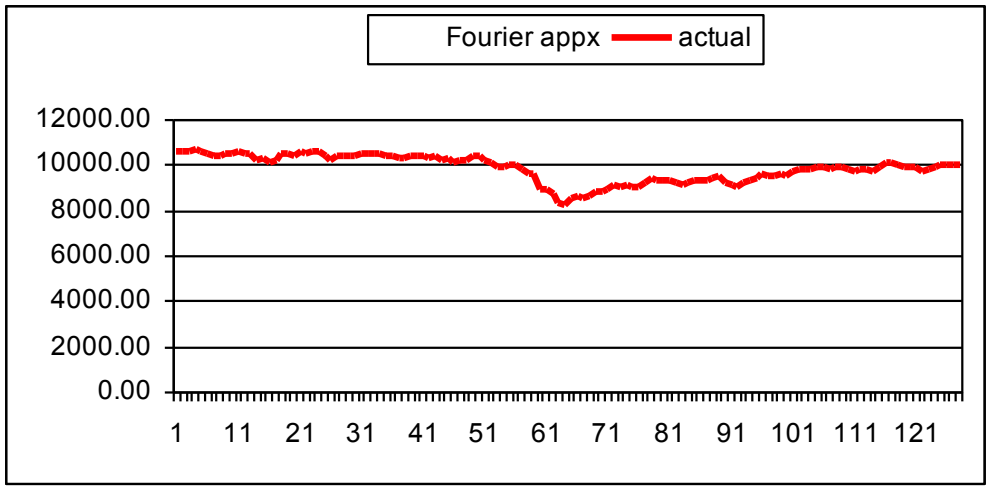
Roadmap

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

DFT and cousins

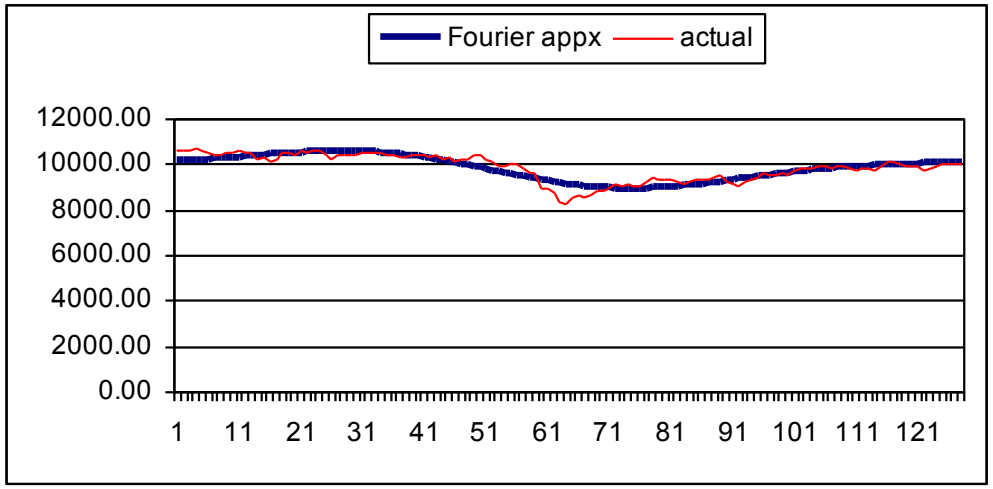
- very good for compressing real signals
- more details on DFT/DCT/DWT: later

DFT and stocks

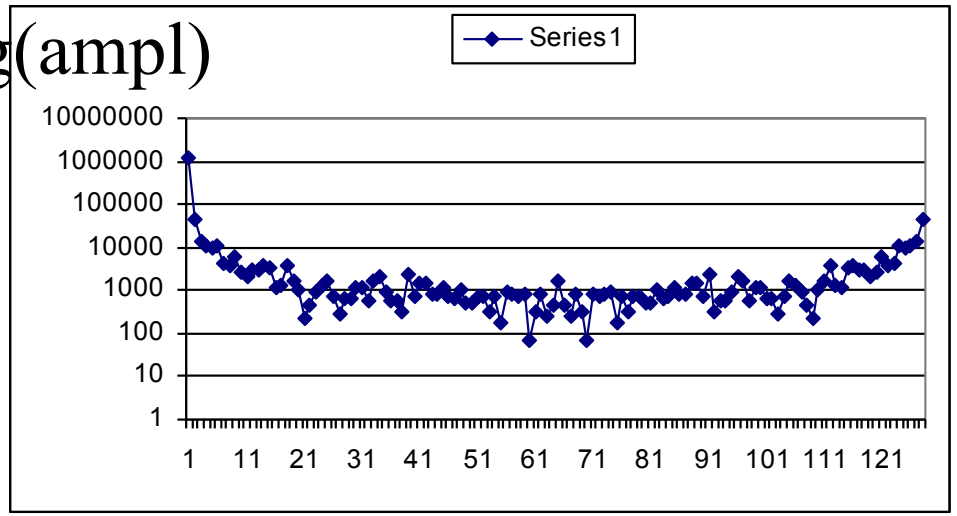


- Dow Jones Industrial index, 6/18/2001-12/21/2001

DFT and stocks



Log(ampl)

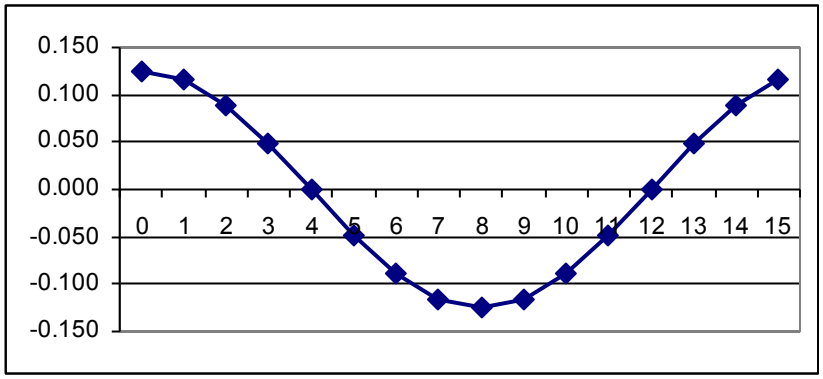


freq

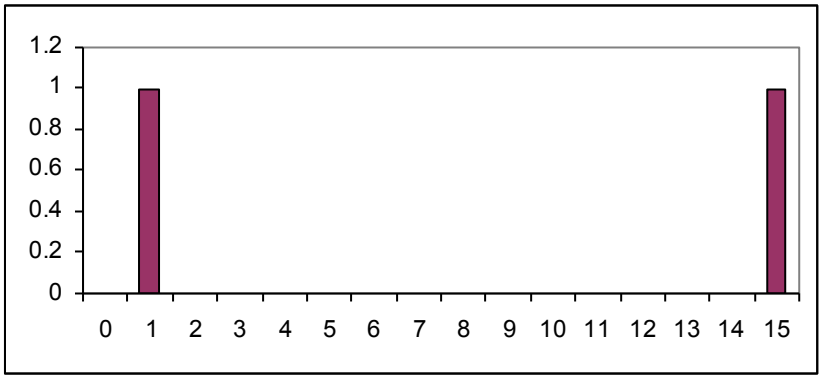
- Dow Jones Industrial index, 6/18/2001-12/21/2001
- just 3 DFT coefficients give very good approximation

DFT (and DWT)

- Many more details, soon



time



freq

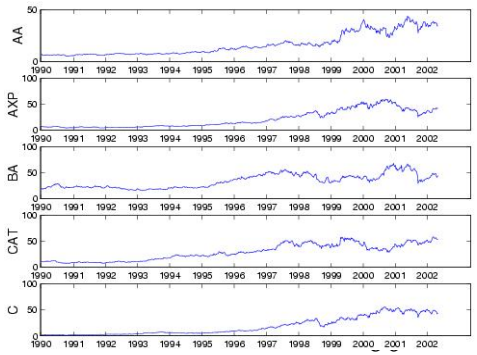
Roadmap

- 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

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

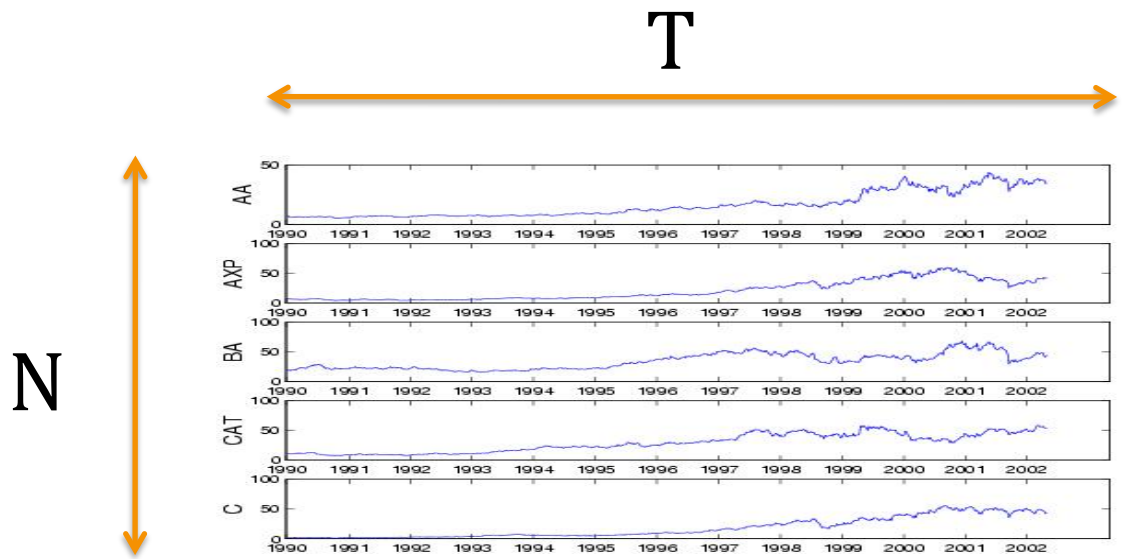
- Given: many time sequences
- Find: the latent ('hidden') variables



SVD

Two (equivalent) interpretations:

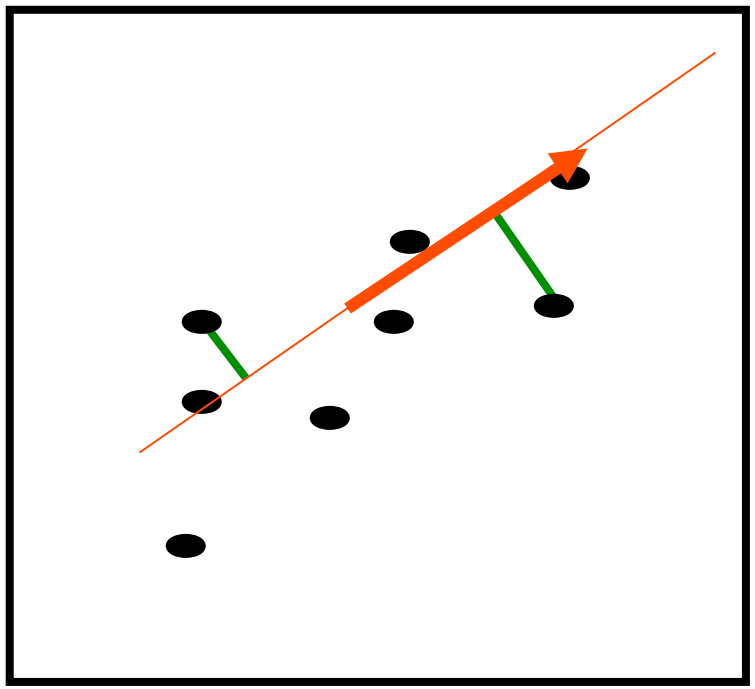
- Geometric (each sequence -> point in T-d space)
- Matrix algebra (N x T matrix)



Singular Value Decomposition (SVD)

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

day2



day1

LSI: S. Dumais; M. Berry

KL: eg, Duda+Hart

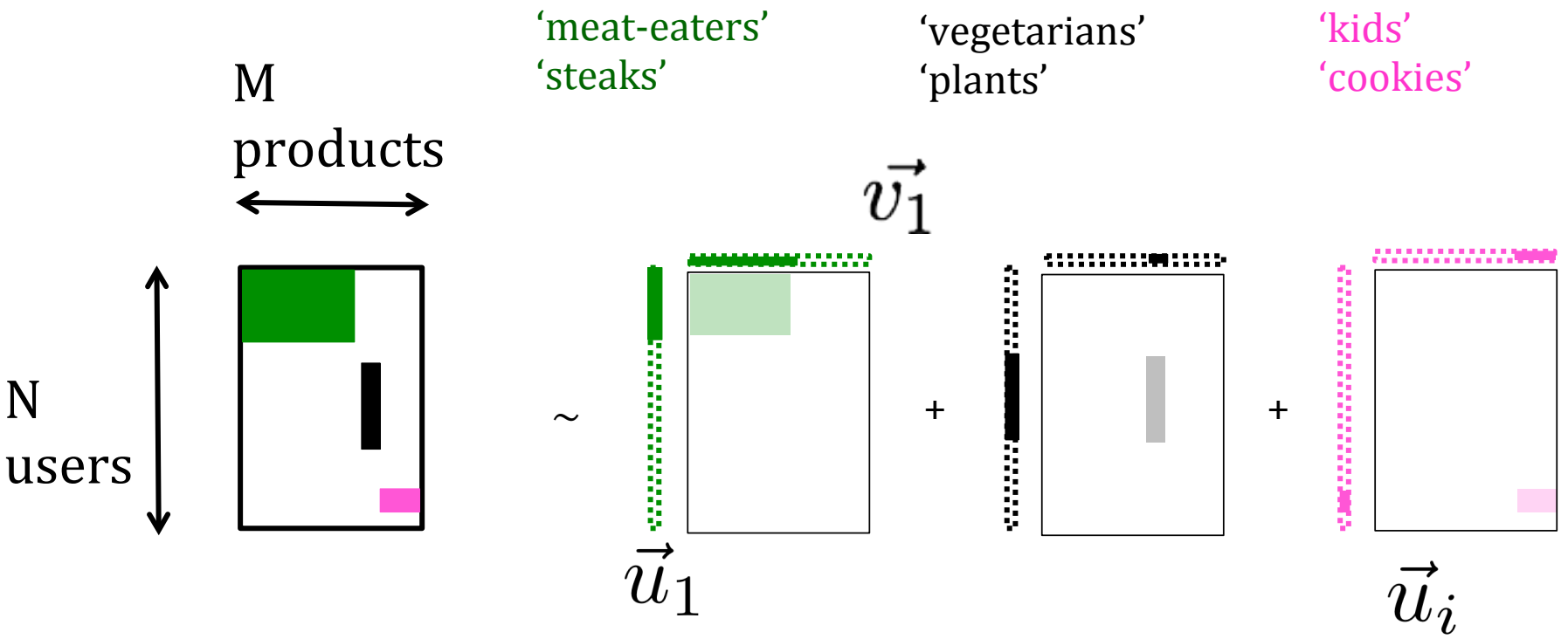
PCA: eg., Jolliffe

Details: [Press+],

[Faloutsos96]

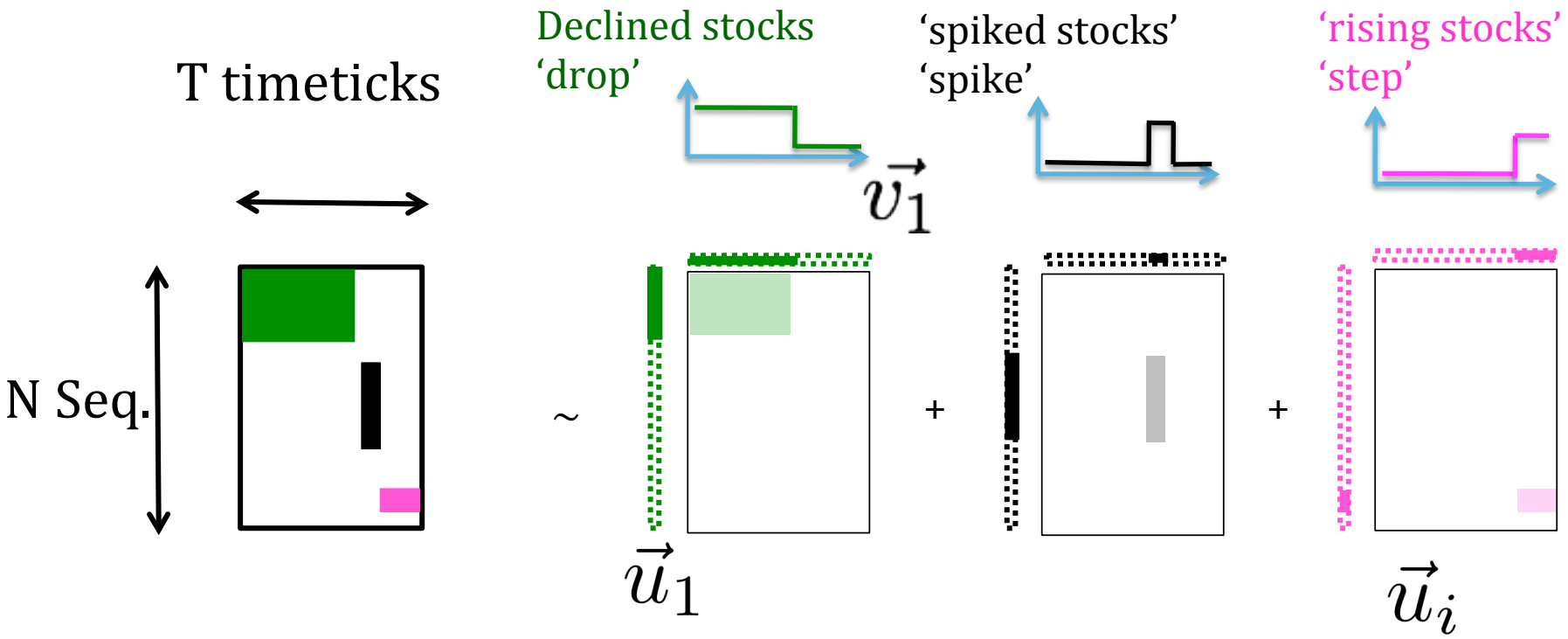
Reminder:

- SVD -> matrix factorization: finds blocks



SVD – matrix interpretation

- SVD -> matrix factorization: finds blocks



SVD

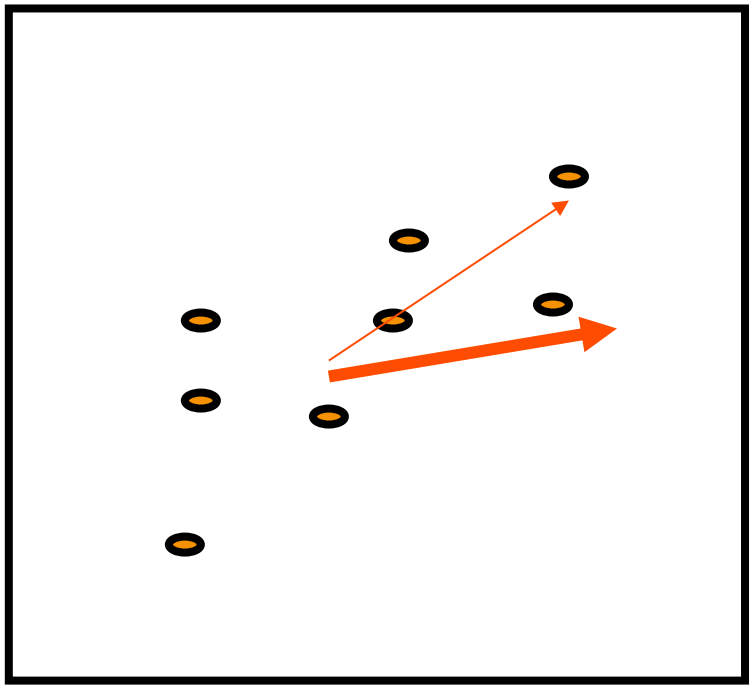
- **Extremely** useful tool
  - (also behind PageRank/google and Kleinberg's algorithm for hubs and authorities)

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])

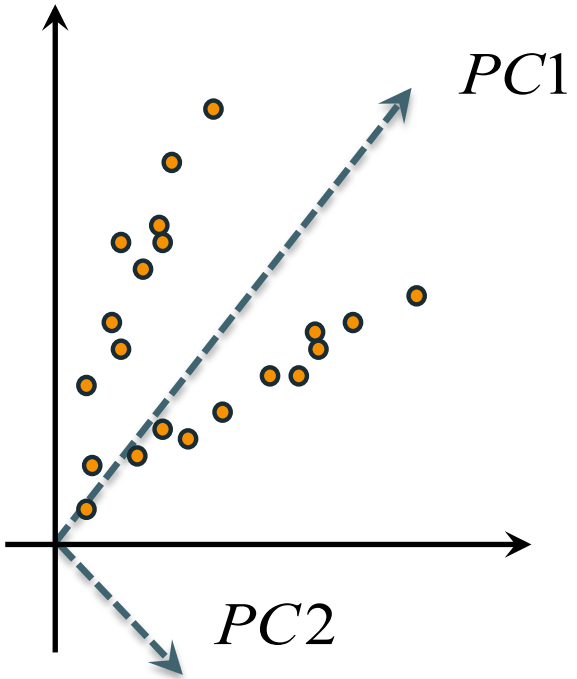


Random projections

- 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])

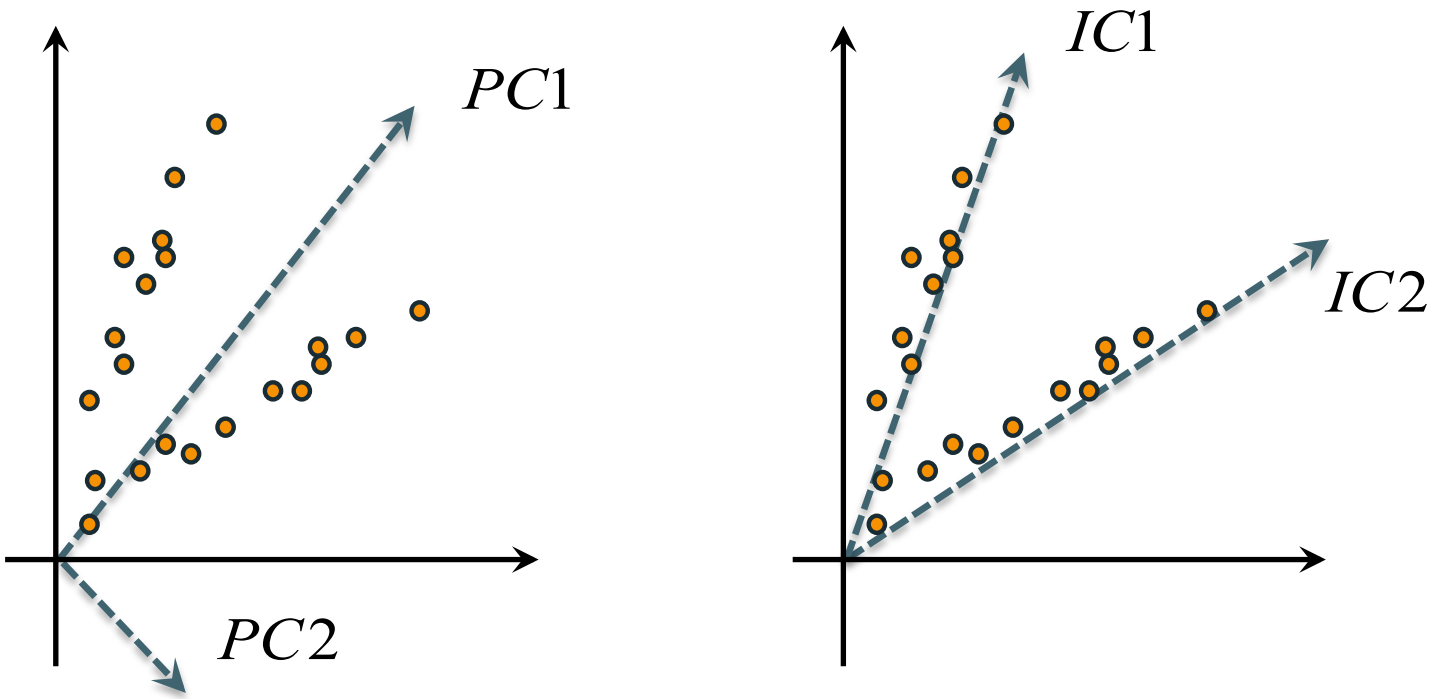
SVD & improvement

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



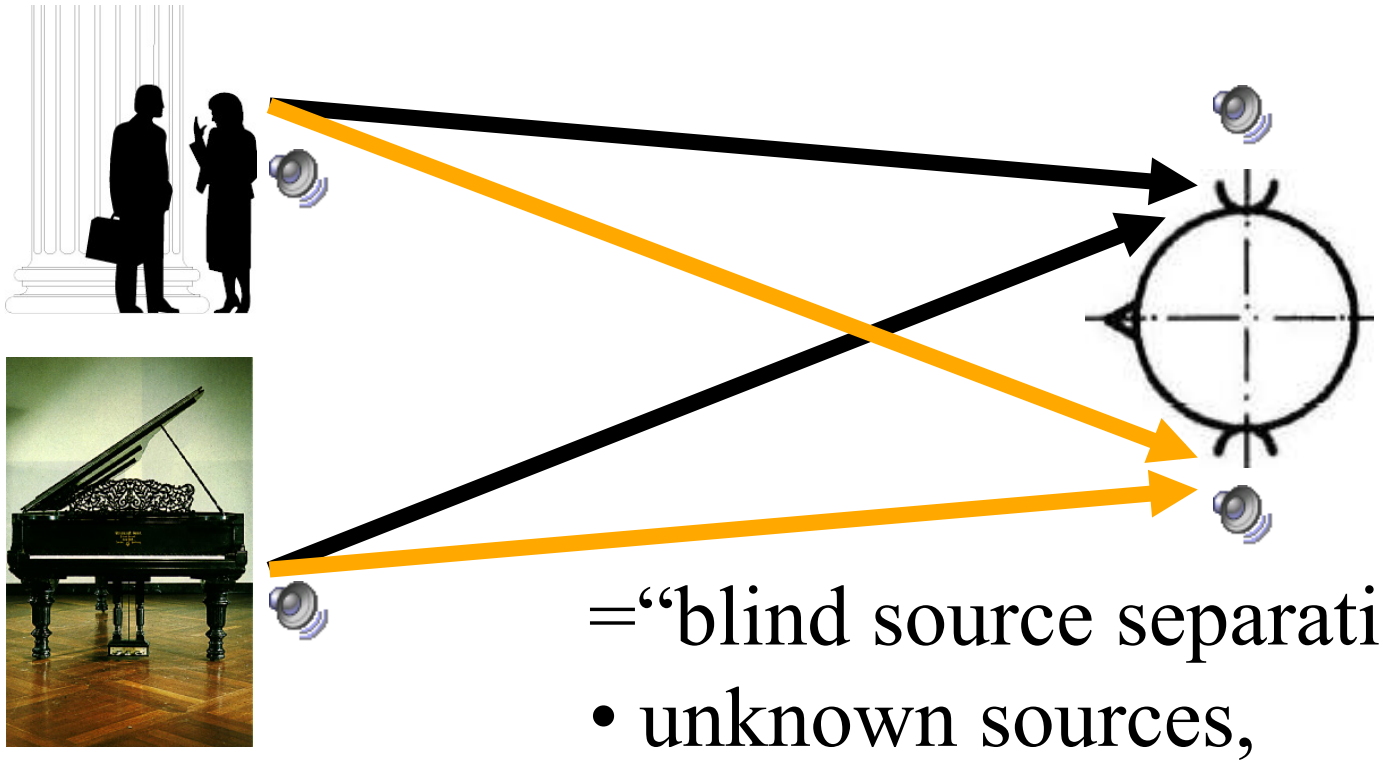
Independent Component Analysis (ICA)

- PCA sometimes misses essential features
  - PCA vs. ICA



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

- Untangle two sound sources



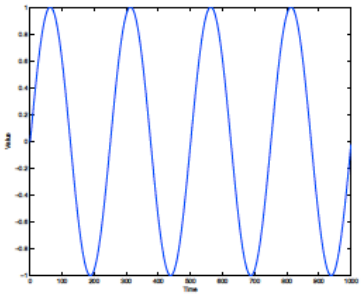
="blind source separation"

- unknown sources,
- unknown mixing

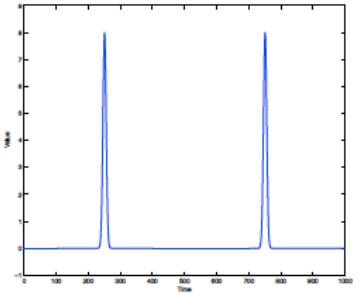
ICA

- Why not PCA

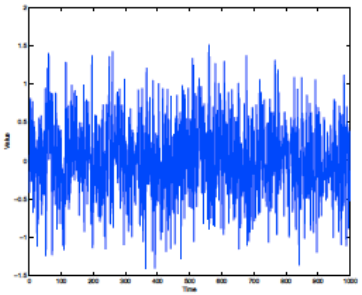
Source



Source #1

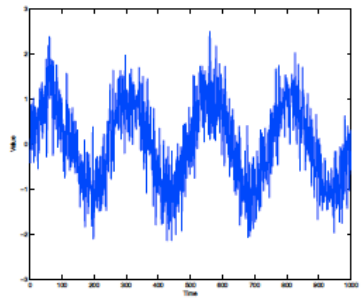


Source #2

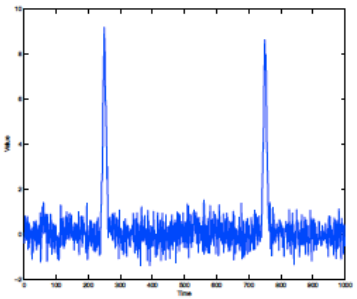


Source #3

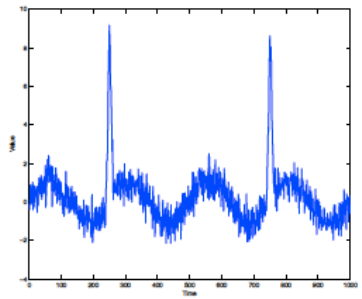
Mix



Sequence #1
(Sources #1 & #3)



Sequence #2
(Sources #2 & #3)

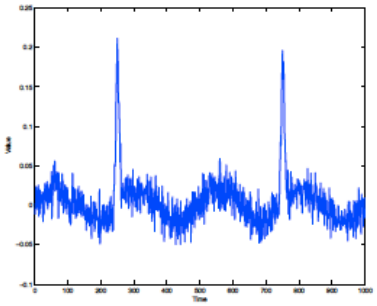


Sequence #3
(Mix of all 3 sources)

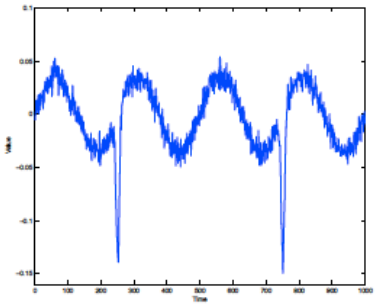
ICA

- Why not PCA

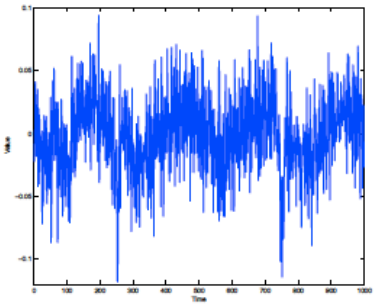
PCA



PC1

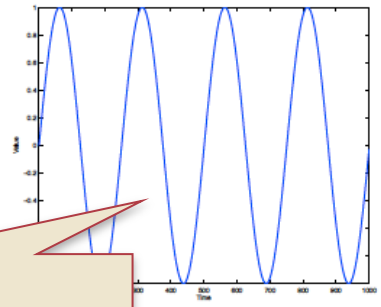


PC2

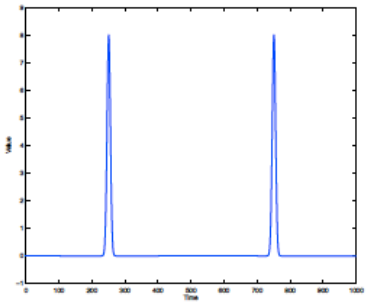


PC3

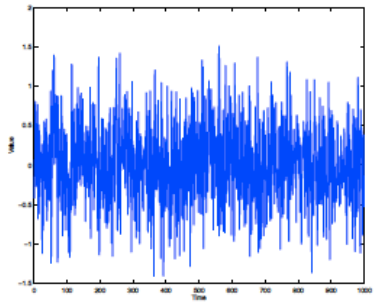
ICA



IC1



IC2



IC3

ICA recognizes the components successfully and separately

Hidden variables

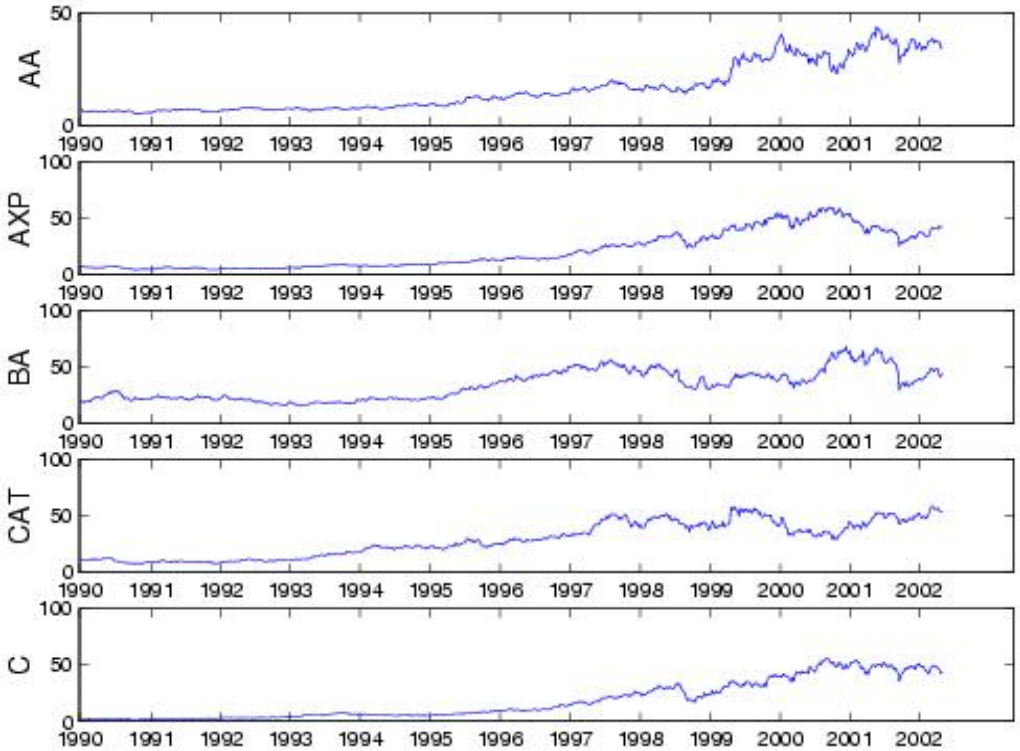
Alcoa

American Express

Boeing

Caterpillar

Citi Group

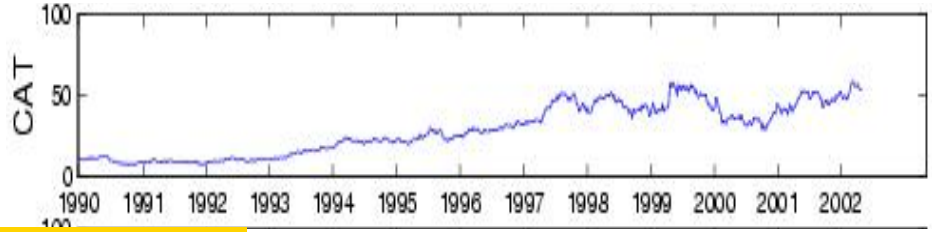


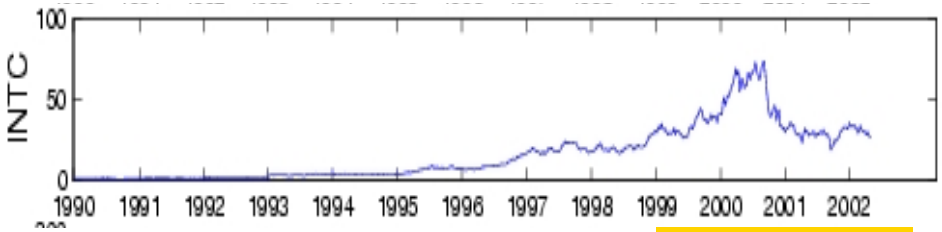
Dow Jones Industrial Average

Find common hidden variables, and weights.

Hidden variables

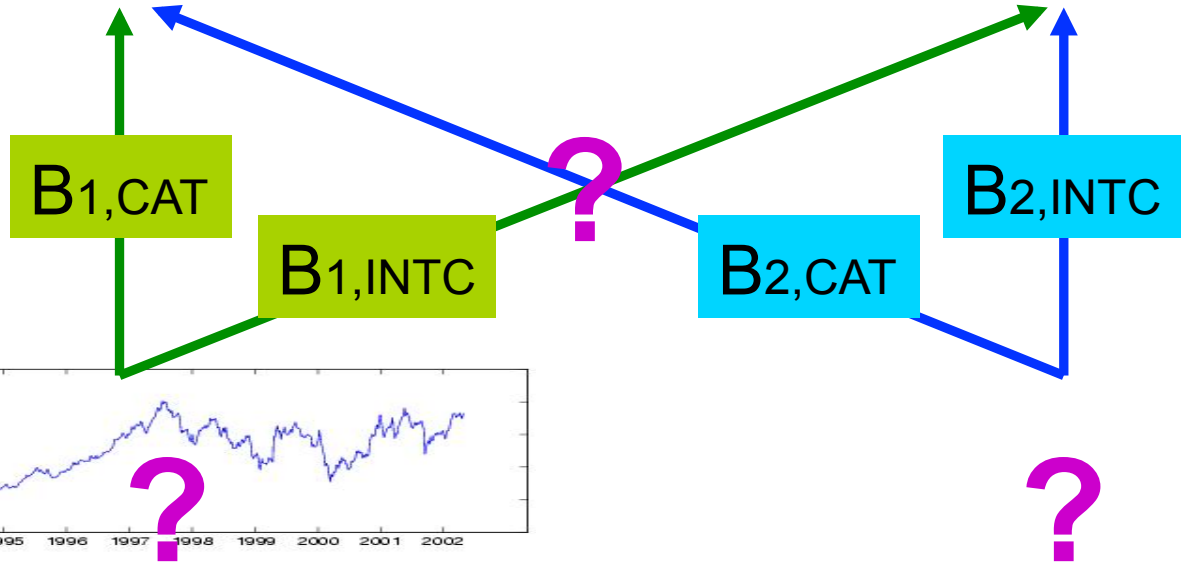
- Find hidden variables [Pan+04]





Caterpillar

Intel

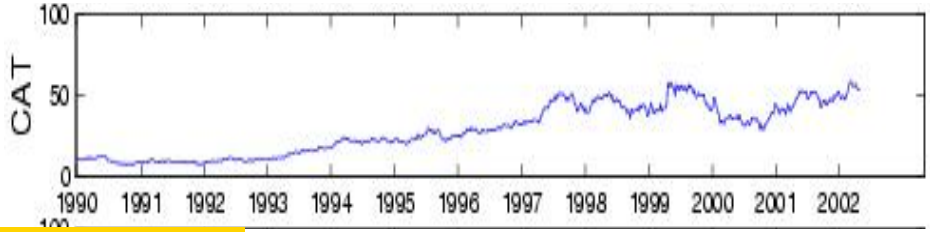


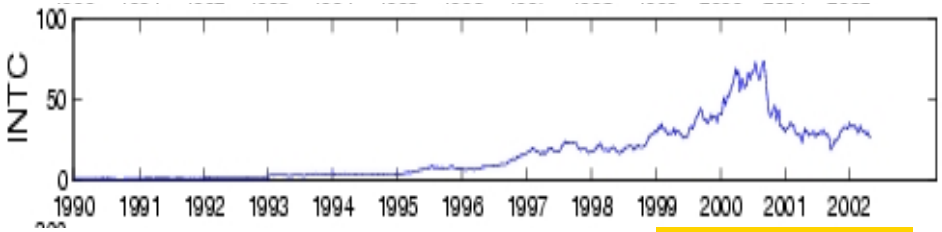
Hidden variable 1

Hidden variable 2

Hidden variables

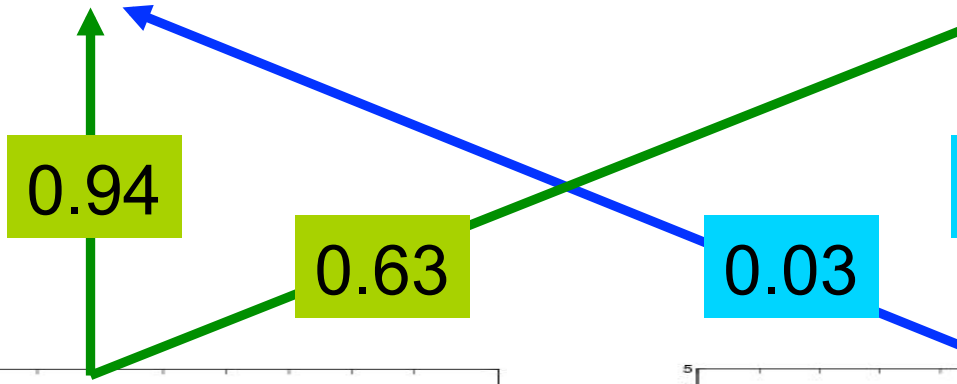
- Find hidden variables [Pan+04]

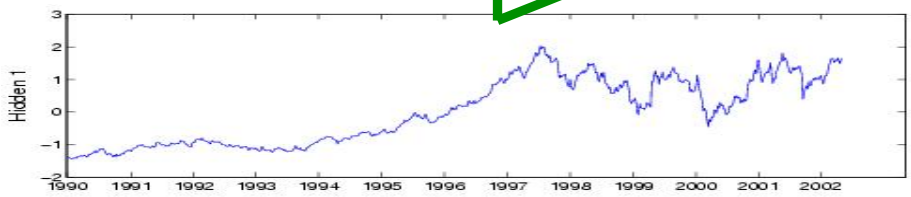


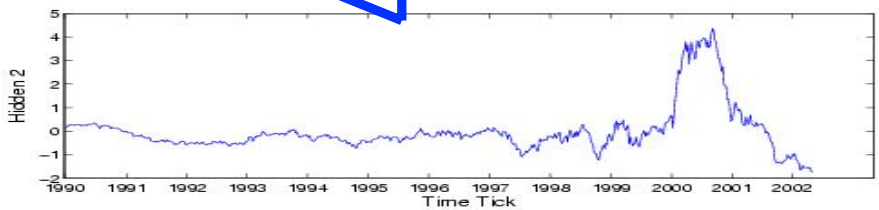


Caterpillar

Intel





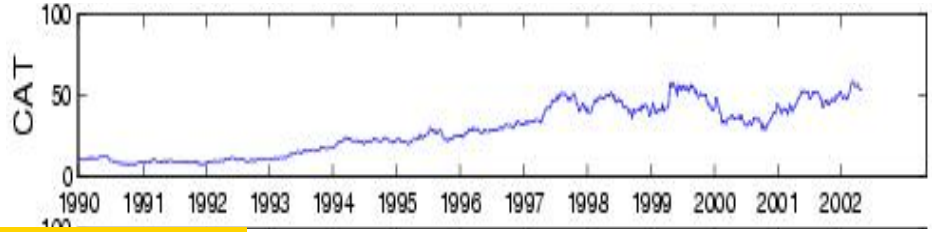


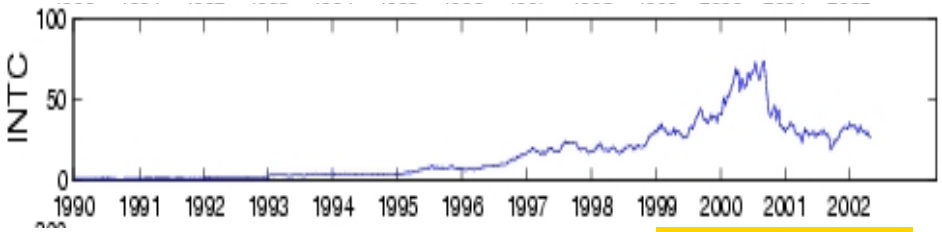
"Hidden variable 1"

"Hidden variable 2"

Hidden variables

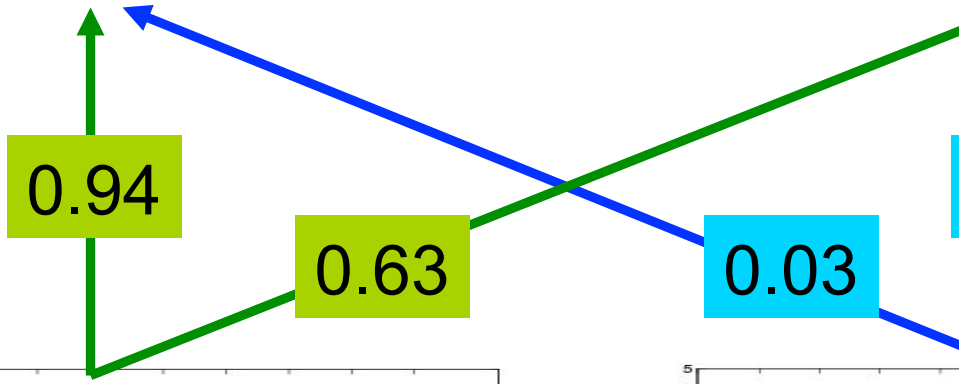
- Find hidden variables [Pan+04]

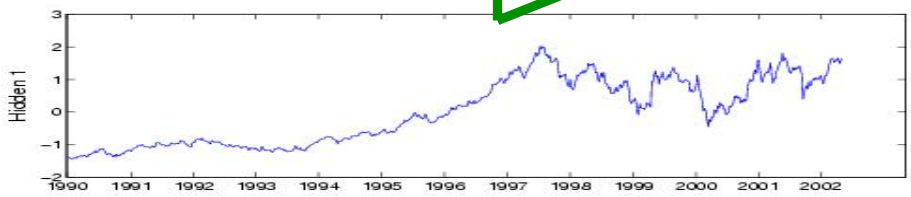


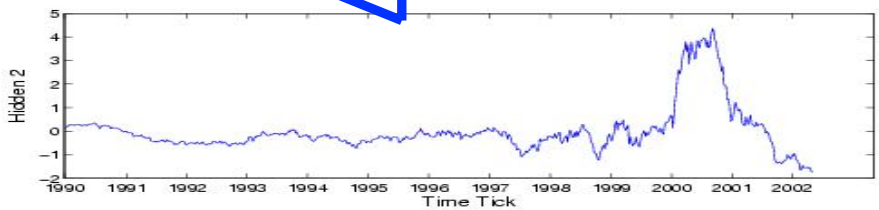


Caterpillar

Intel





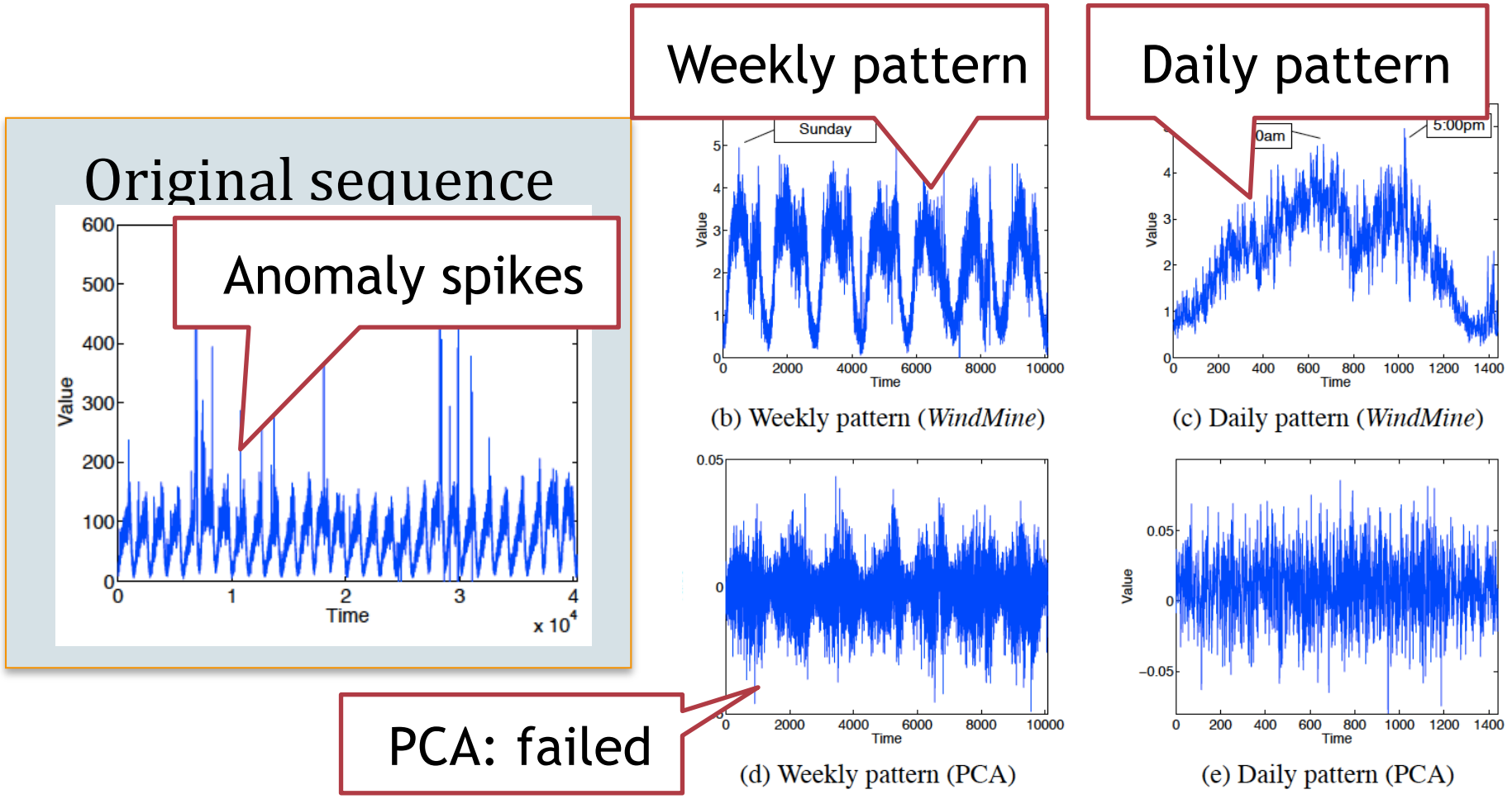


"General trend"

"Internet bubble"

Hidden variables

- Local component analysis [Sakurai+11]

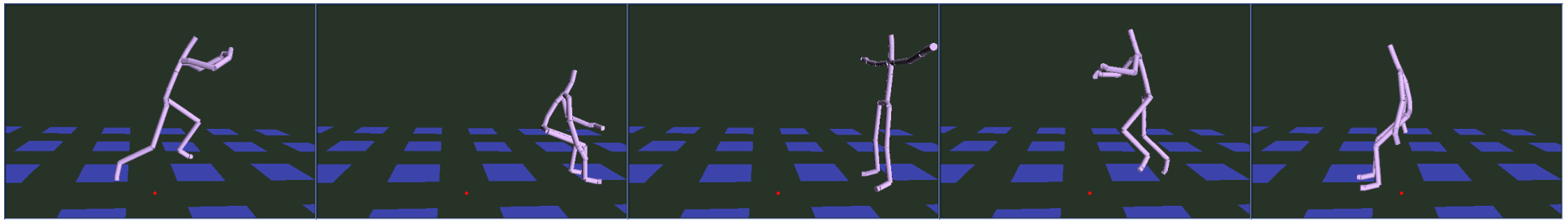


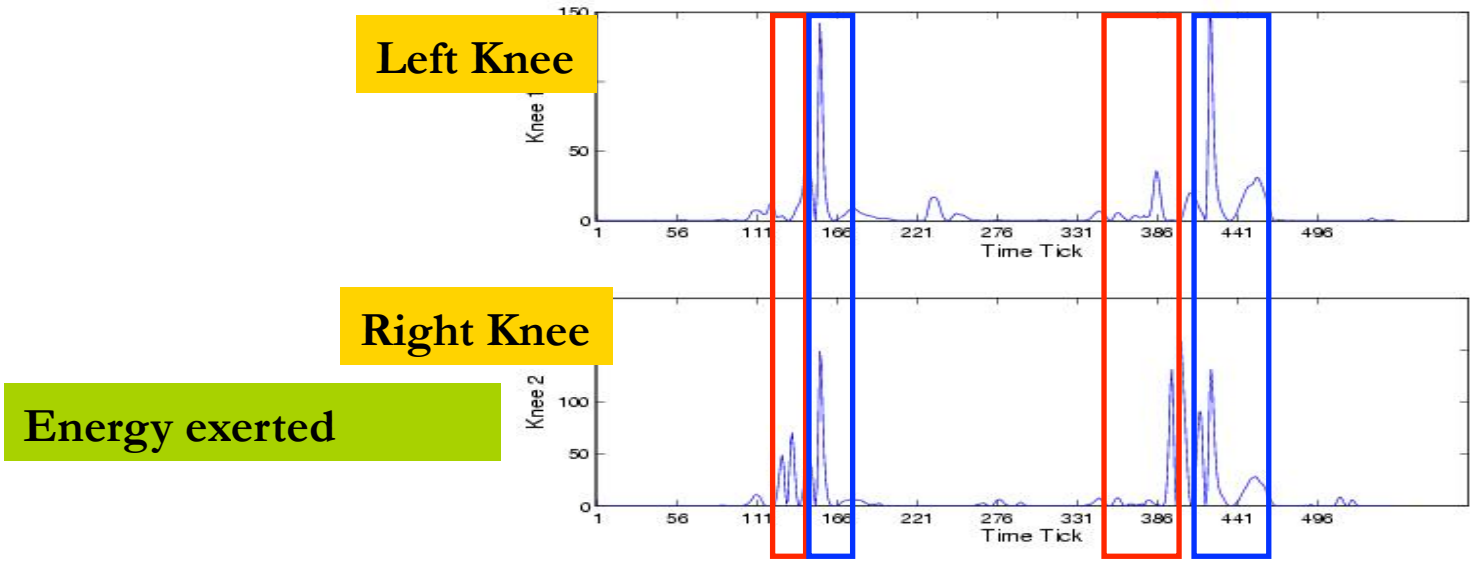
Motivation: Find hidden variables

- 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

Motivation: Find patterns in data

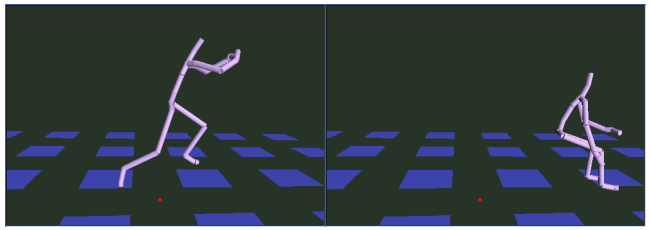
- Motion capture data (broad jumps)

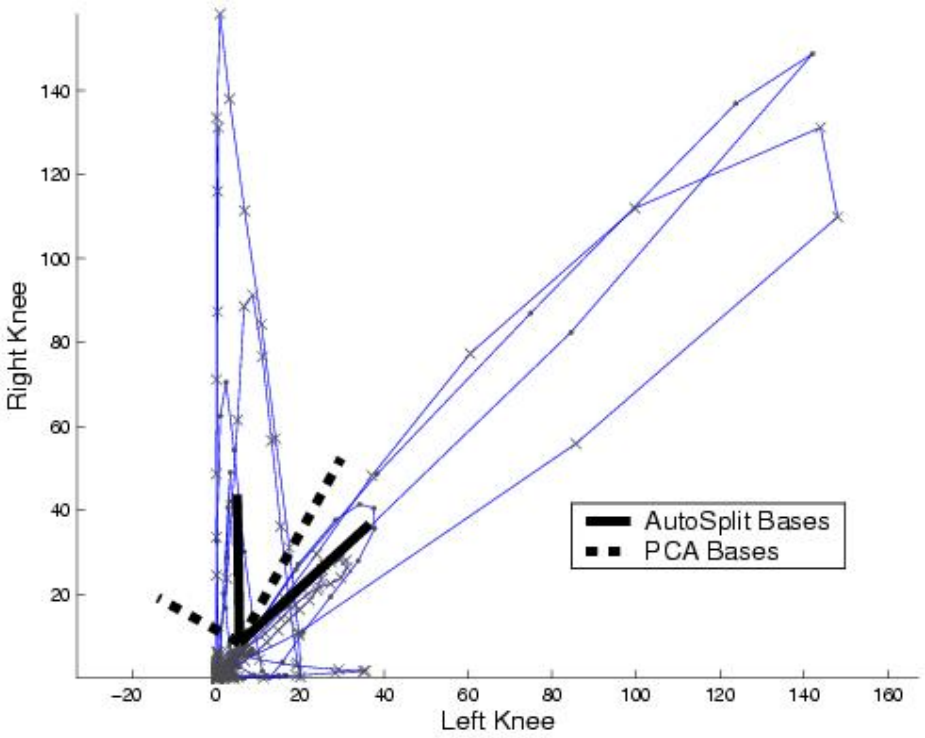




Motivation: Find patterns in data

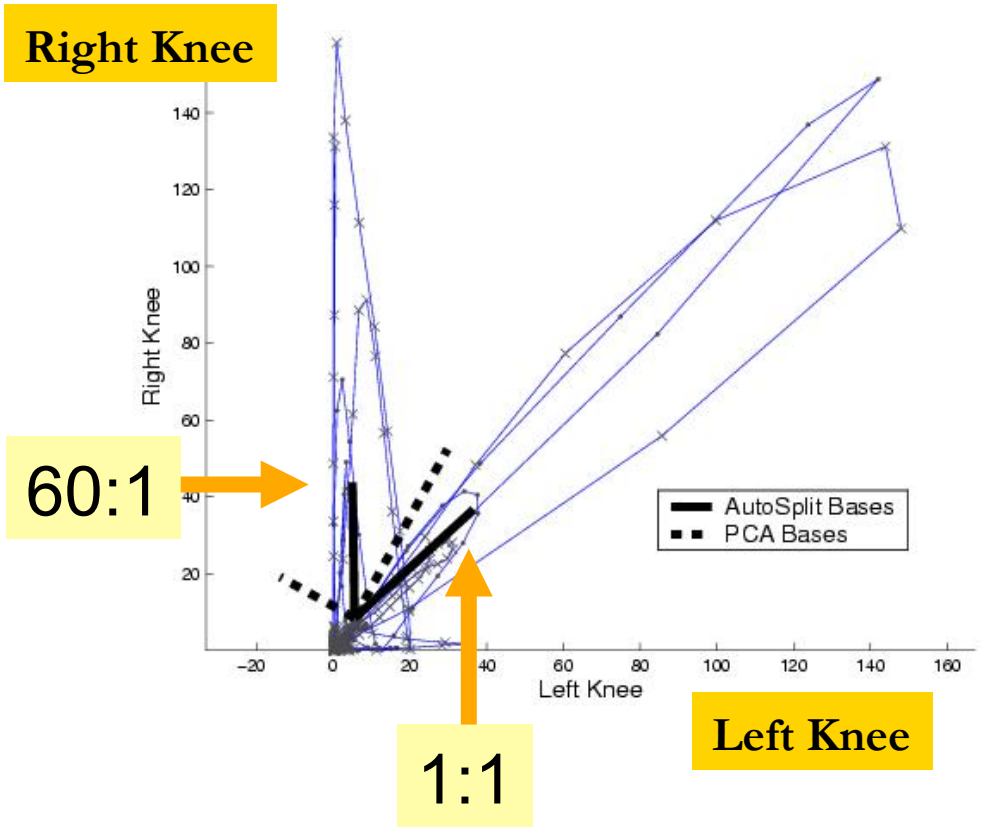
- Best SVD axis: not always meaningful!





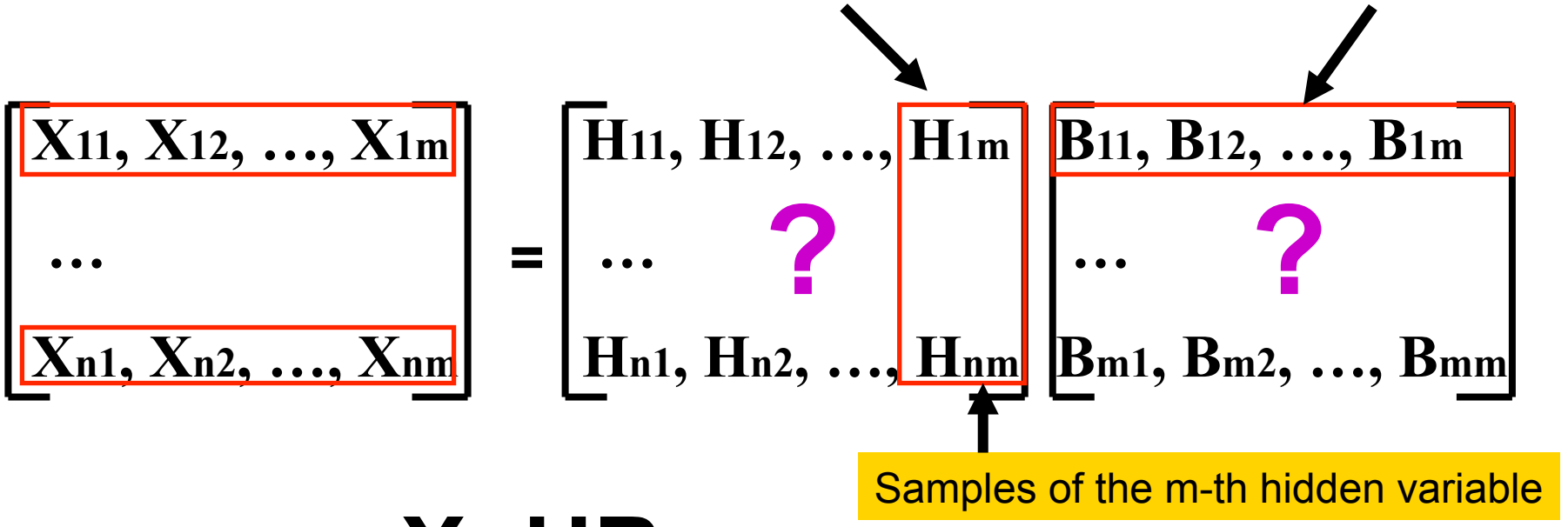
Motivation: Find patterns in data

- Human would say
  - Pattern 1: along diagonal
  - Pattern 2: along vertical axis
- How to find these automatically?



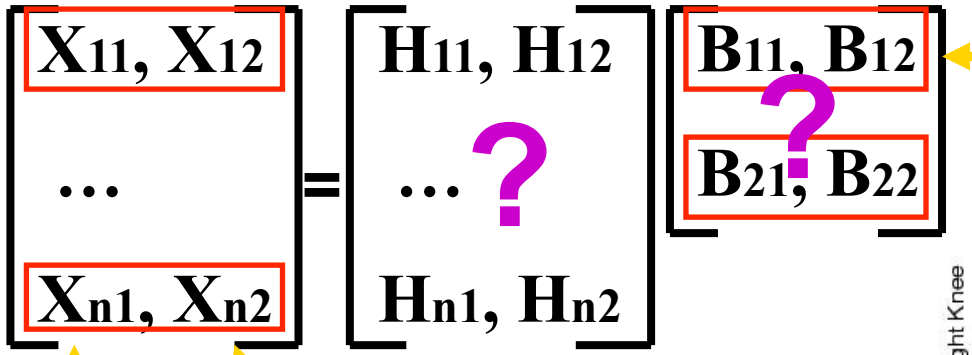
Problem formulation

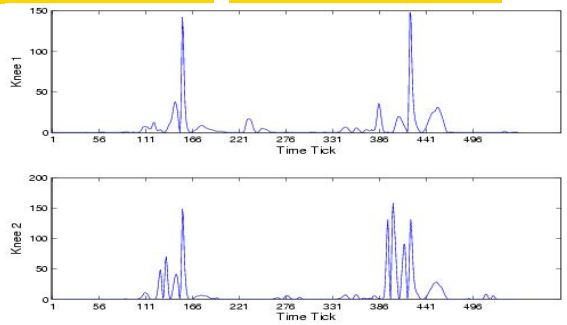
- Given n data items, each has m attributes
- Find the m hidden variables and the m bases

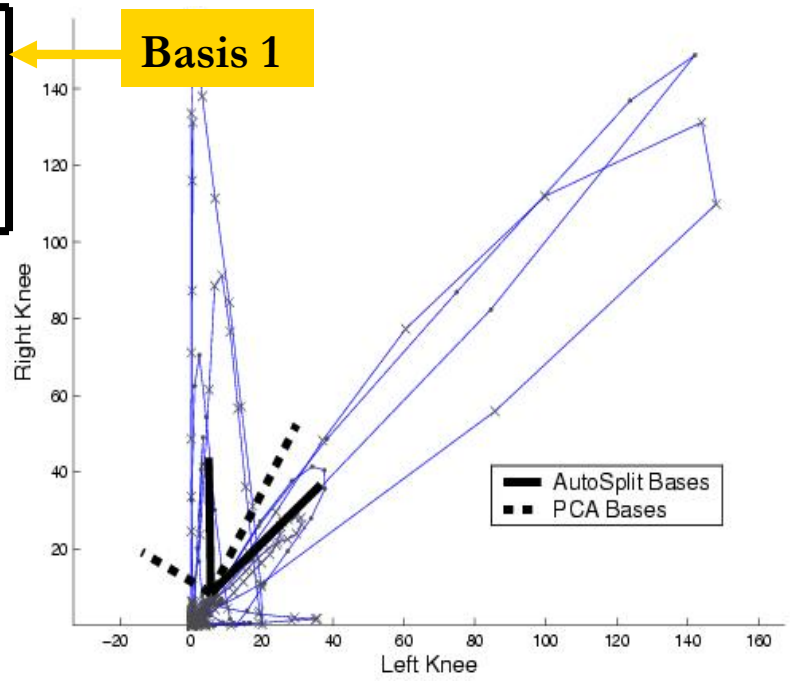


X=HB

Formulation: (Q1) Find patterns in data



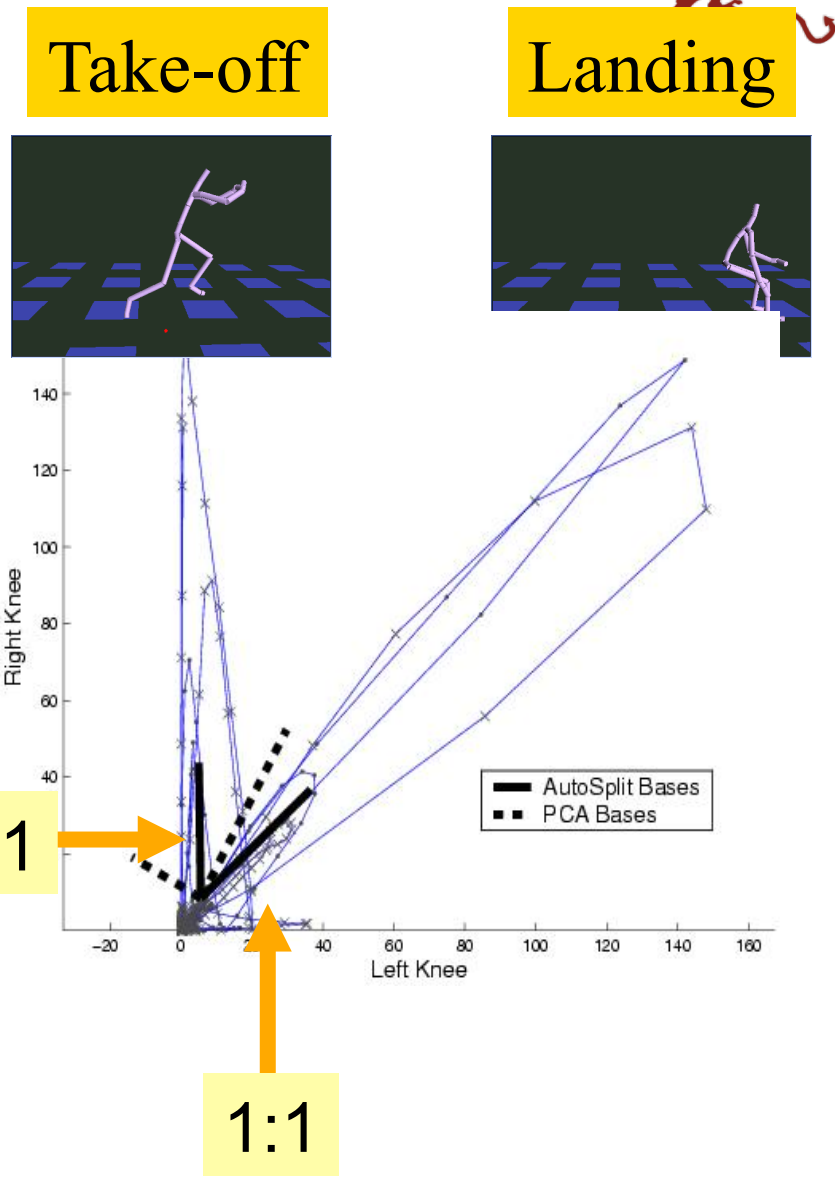




http://www.cs.kumamoto-u.ac.jp/~yasuko/TALKS/15-SIGMOD-tut/

Q1: Find patterns

- Patterns found

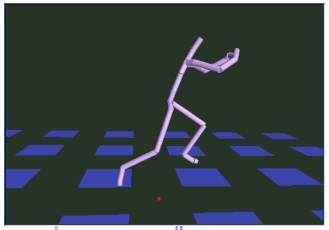


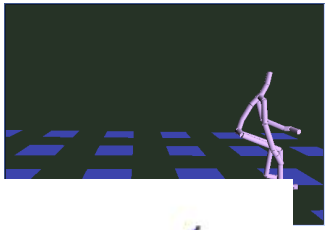
Q1: Find patterns

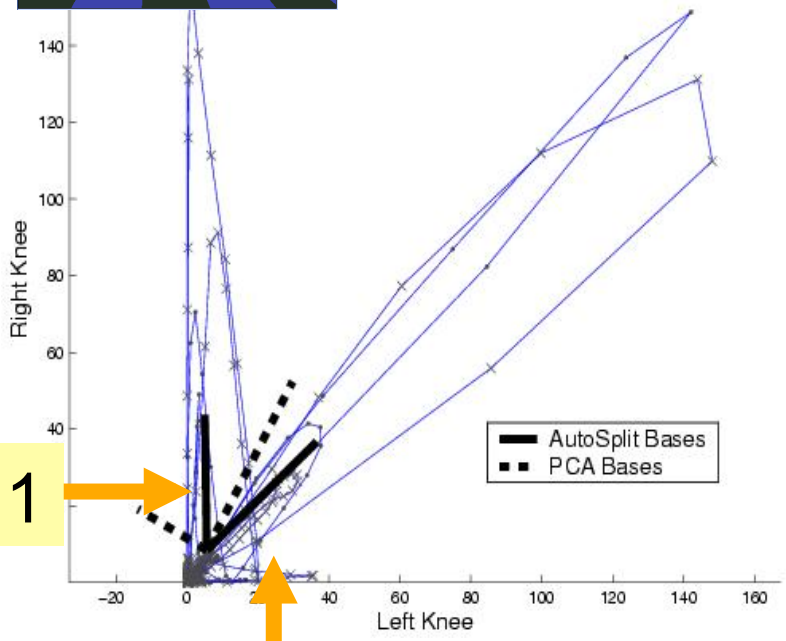
- Patterns found
  - Landing: both knees

Take-off

Landing





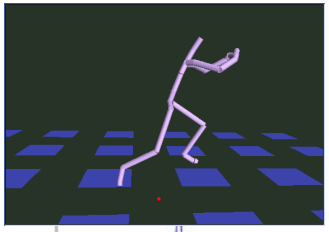


m=2, n=550

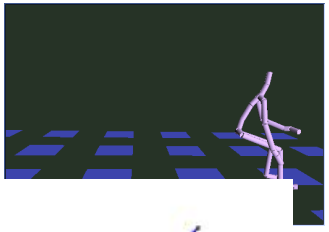
Q1: Find patterns

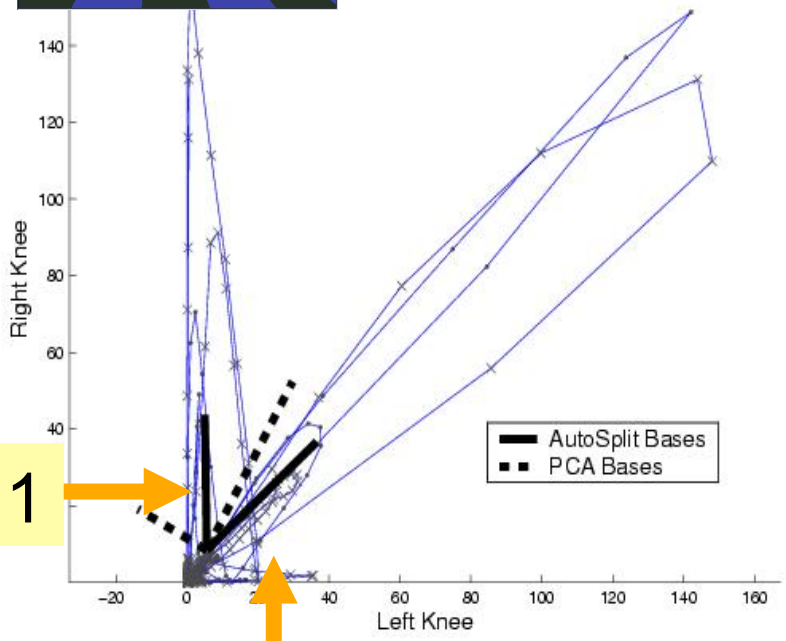
- Patterns found
  - Landing: both knees
  - Take-off: right knee
  - Right-handed actor

Take-off



Landing





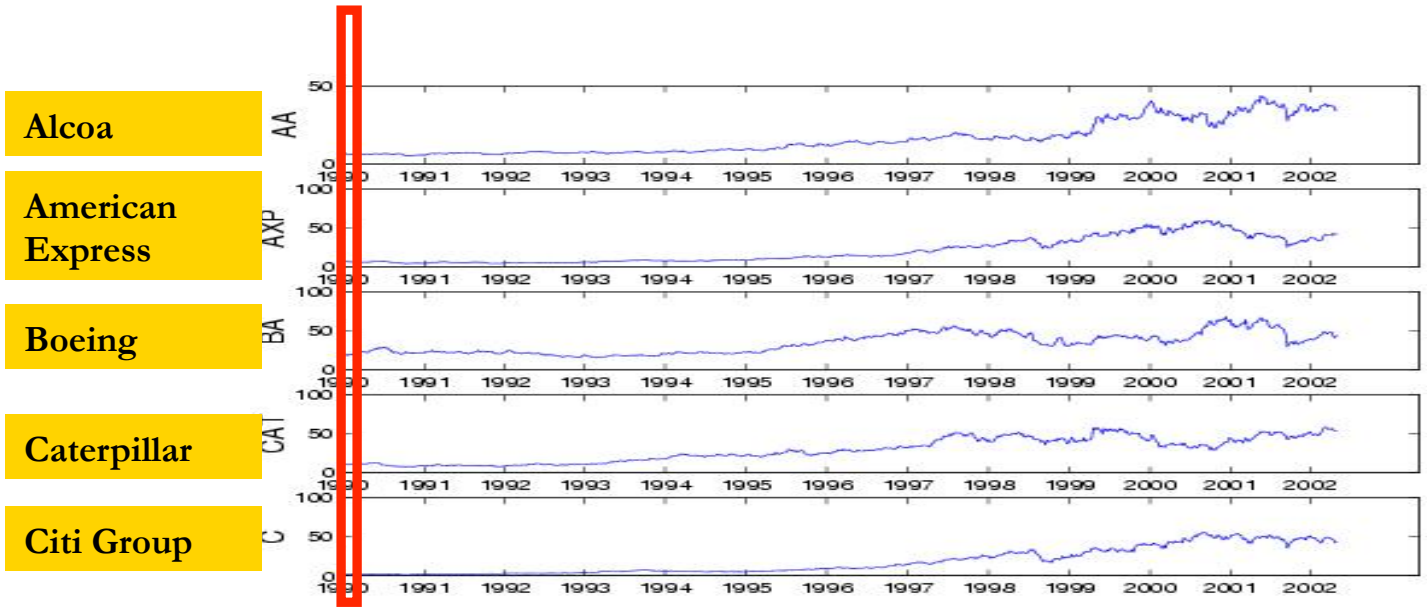
60:1

1:1

m=2, n=550

Q2: Find hidden variables (DJIA stocks)

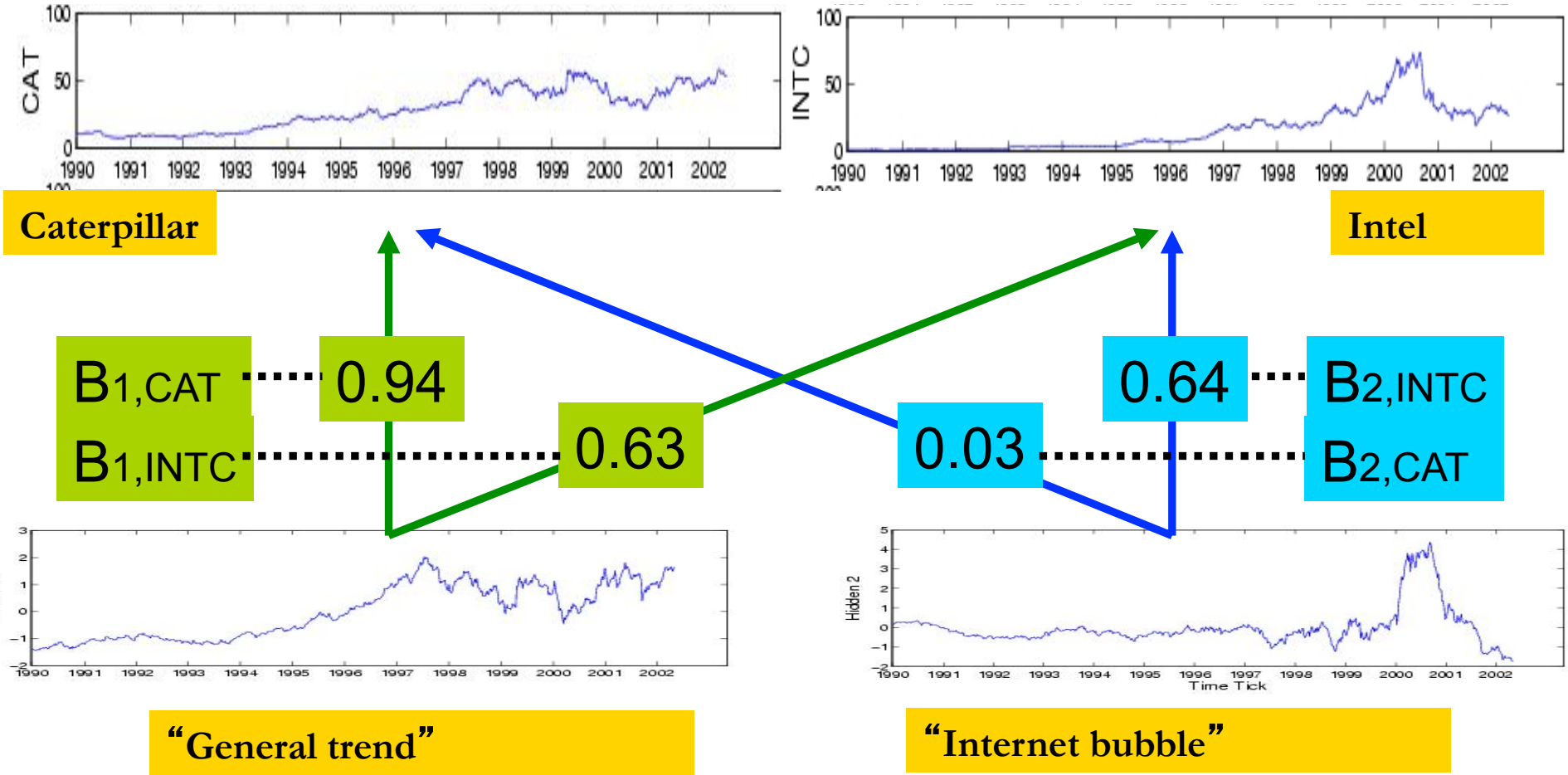
- Weekly DJIA closing prices
  - 01/02/1990-08/05/2002, n=660 data points
  - A data point: prices of 29 companies at the time



Kumamoto U

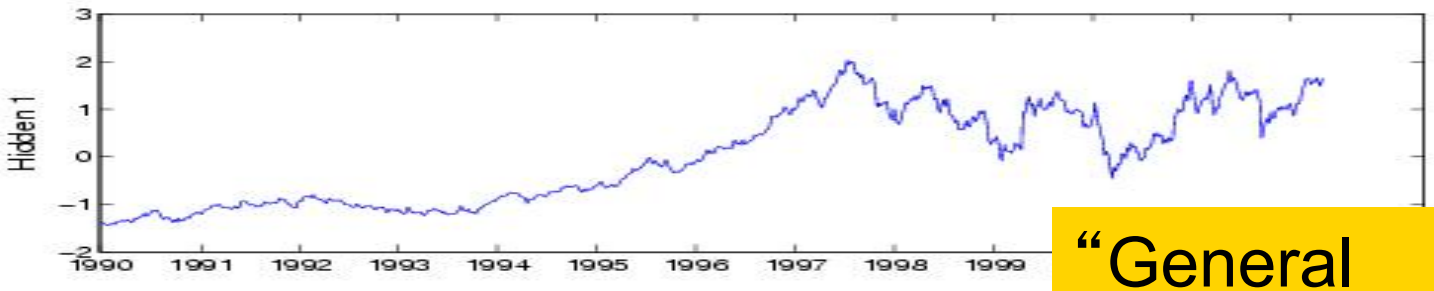
CMU CS

(Q2) Characterize hidden variable by the companies it influences



Companies related to hidden variable 1

$B_{1,j}$			
Highest		Lowest	
Caterpillar	0.938512	AT&T	0.021885
Boeing	0.911120	WalMart	0.624570
MMM	0.906542	Intel	0.638010
Coca Cola	0.903858	Home Depot	0.647774
Du Pont	0.900317	Hewlett-Packard	0.658768



"General trend"

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

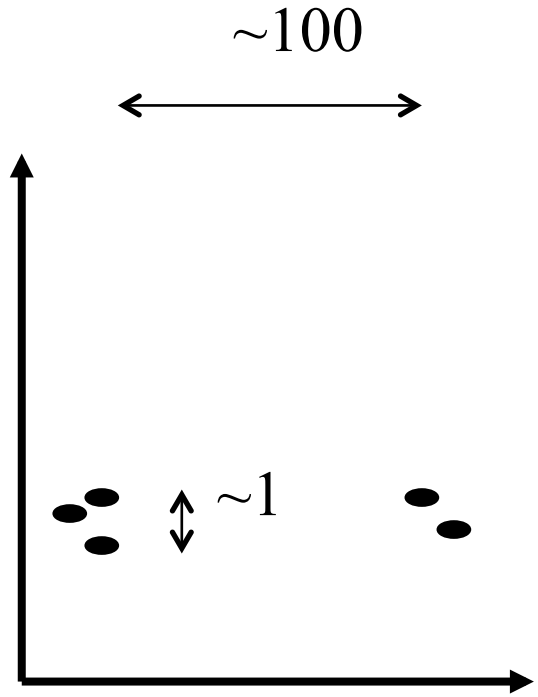
- Motivation
- Similarity Search and Indexing
- Feature extraction
  - DFT, DWT, DCT (data independent)
  - SVD, ICA (data independent)
  - MDS, FastMap
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

MDS / FastMap

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

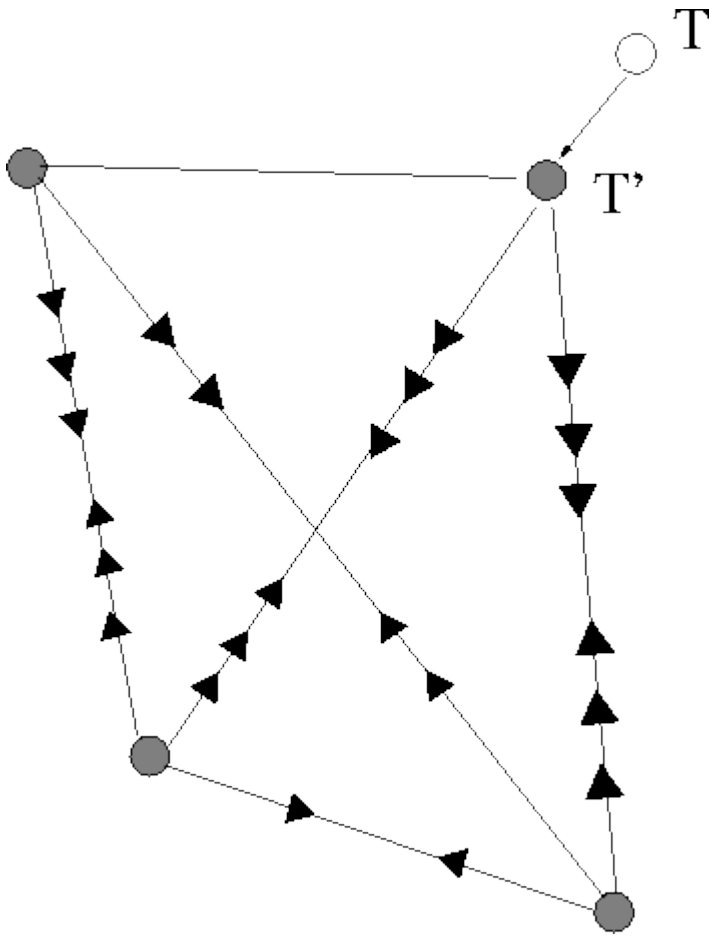
MDS/FastMap

	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

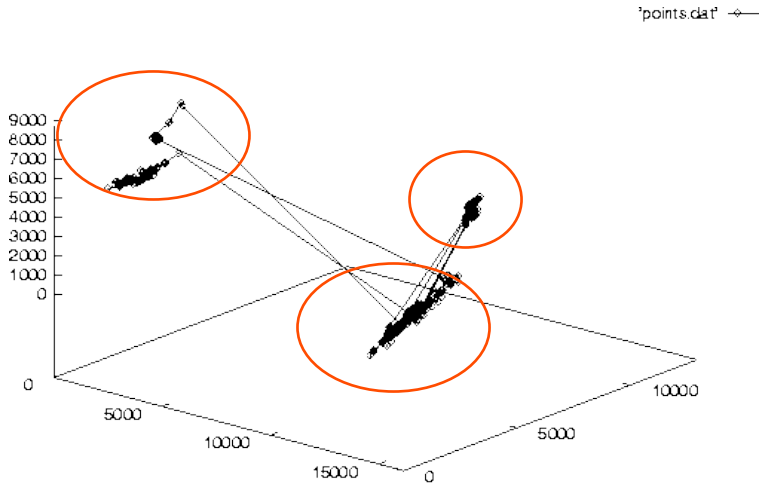


FastMap

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

FastMap: Application

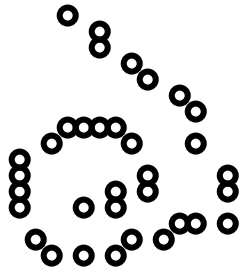
VideoTrails [Kobla+97]



scene-cut detection (about 10% errors)

Variations

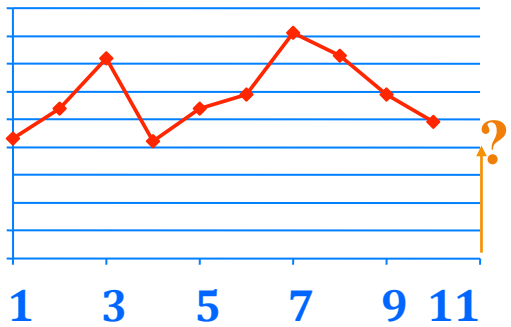
- Isomap [Tenenbaum, de Silva, Langford, 2000]
- LLE (Local Linear Embedding) [Roweis, Saul, 2000]
- MVE (Minimum Volume Embedding) [Shaw & Jebara, 2007]

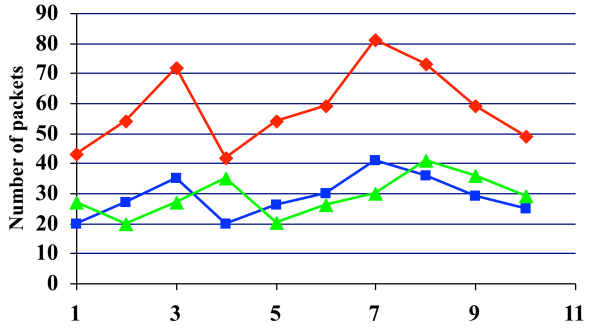


Wish list

- Problem 1: find patterns/**rules**
- Problem 2: **forecast**

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





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

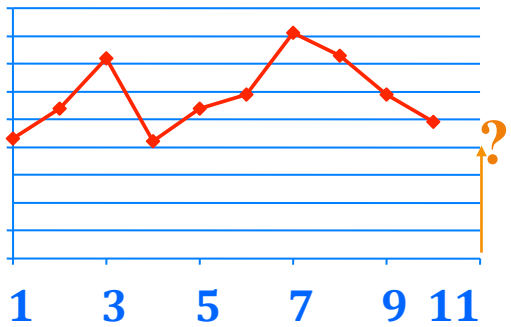
- Josh Tenenbaum, Vin de Silva and John Langford. *A Global Geometric Framework for Nonlinear dimensionality Reduction*. Science 290, pp. 2319-2323, 2000.

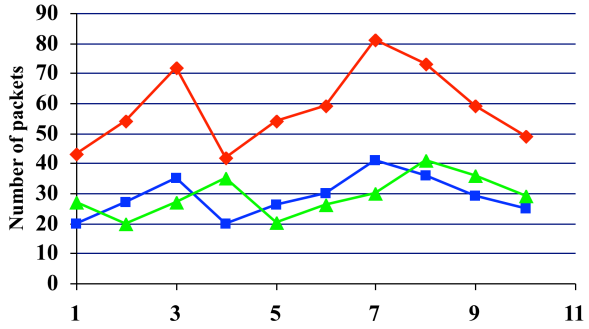
Roadmap

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





Forecasting

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

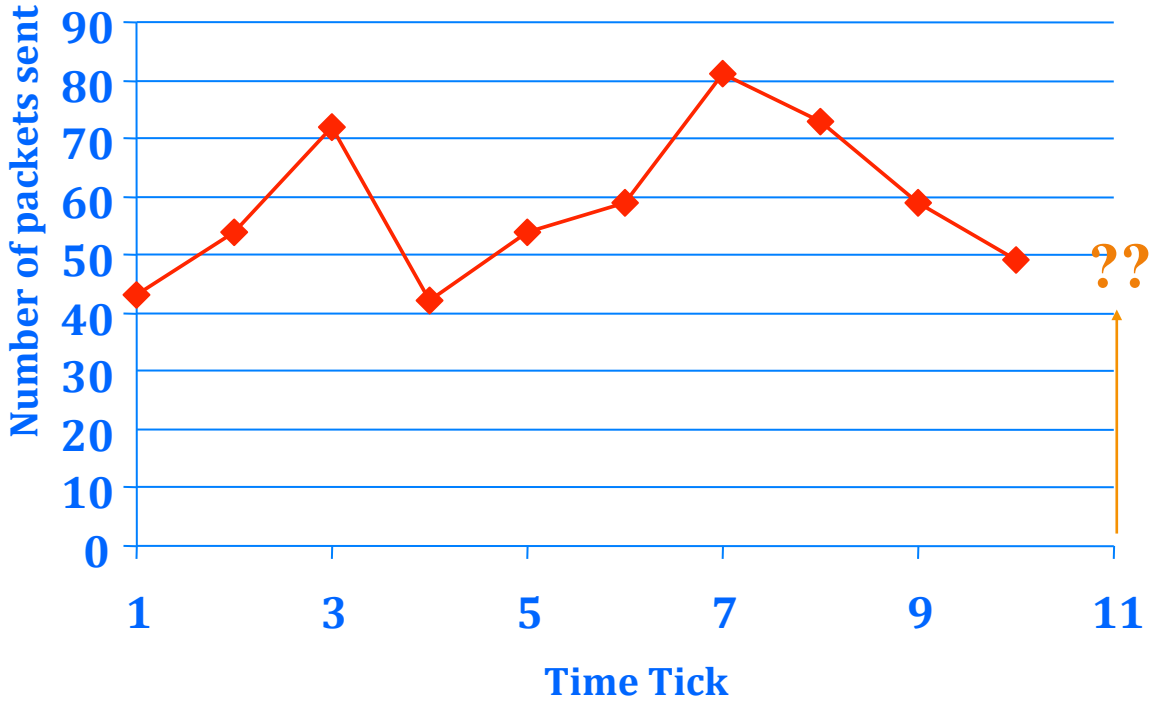
http://www.hfac.uh.edu/MediaFutures/thoughts.html

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

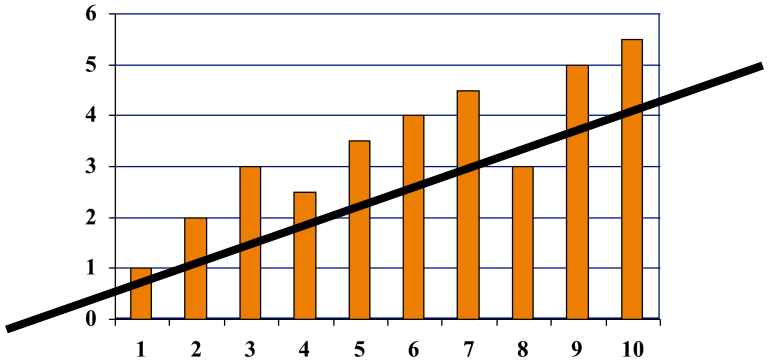
MANUALLY:

remove trends

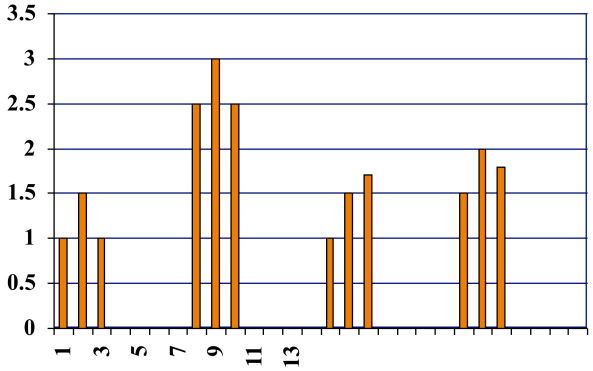
periodicities

spot

7 days



time



time

Problem: Forecast

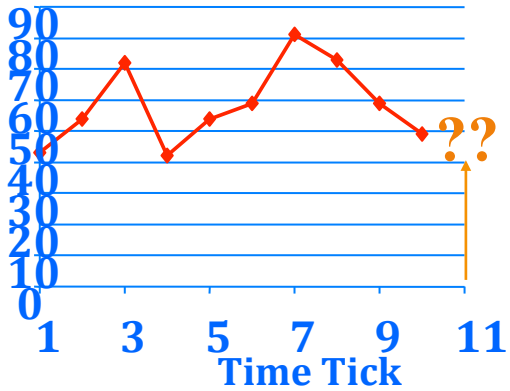
- Solution: try to express

  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)

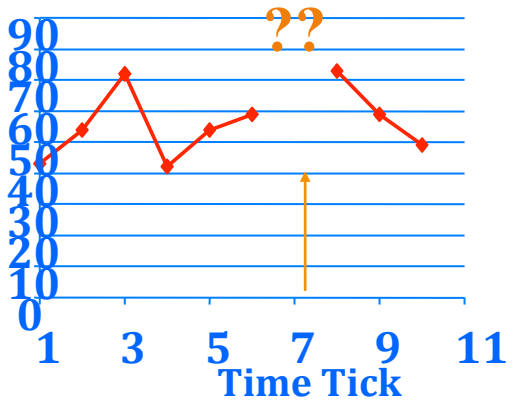
- Solution - interpolate: try to express

  x_t

  as a linear function of the past AND the future:

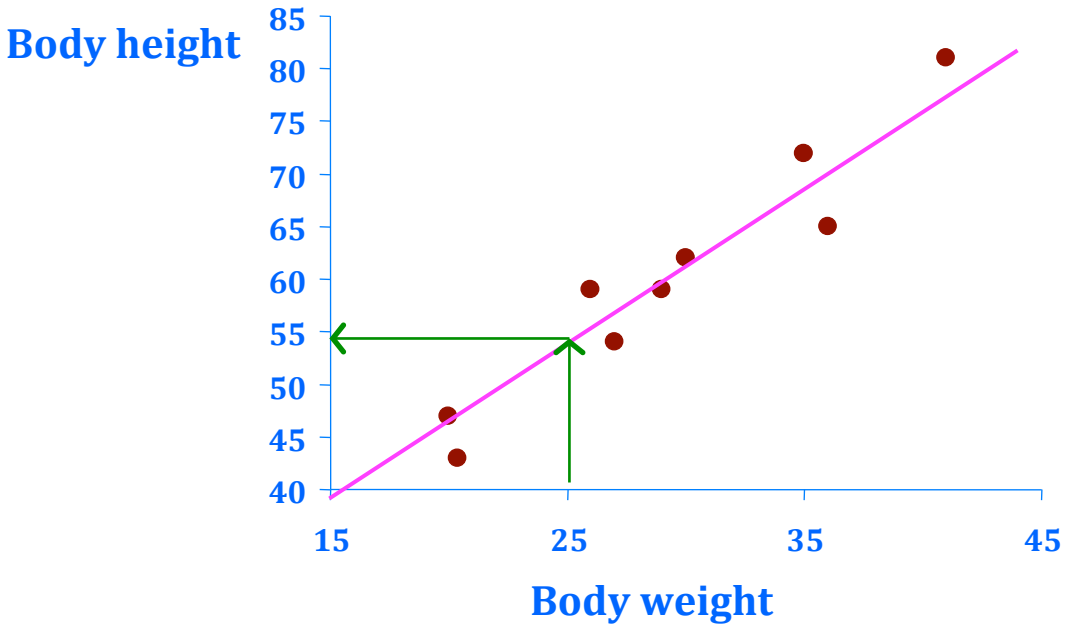
  $x_{t+1}, x_{t+2}, \ldots x_{t+wfuture}; x_{t-1}, \ldots x_{t-wpast}$

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



Linear Regression: idea

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



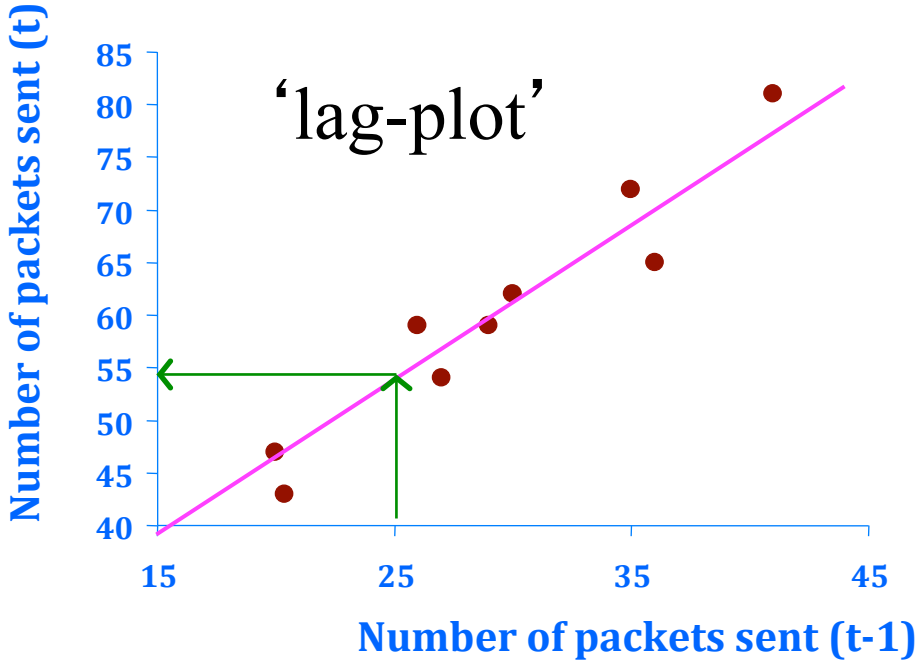
- express what we don't know (= 'dependent variable')
- as a linear function of what we know (= 'indep. variable(s)')

Linear Auto Regression:

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

Linear Auto Regression:

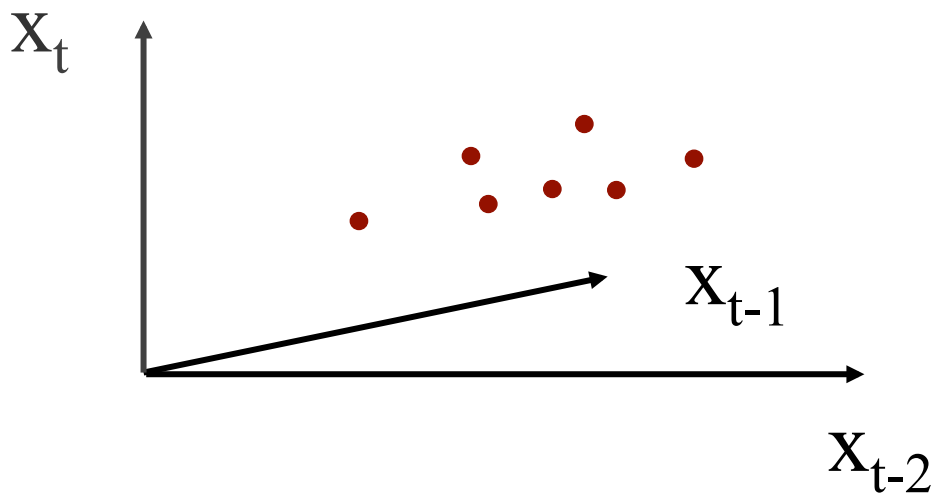
Time	*Packets Sent (t-1)*	*Packets Sent(t)*
1	-	43
2	43	54
3	54	72
	…	
…		…
N	25	??



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

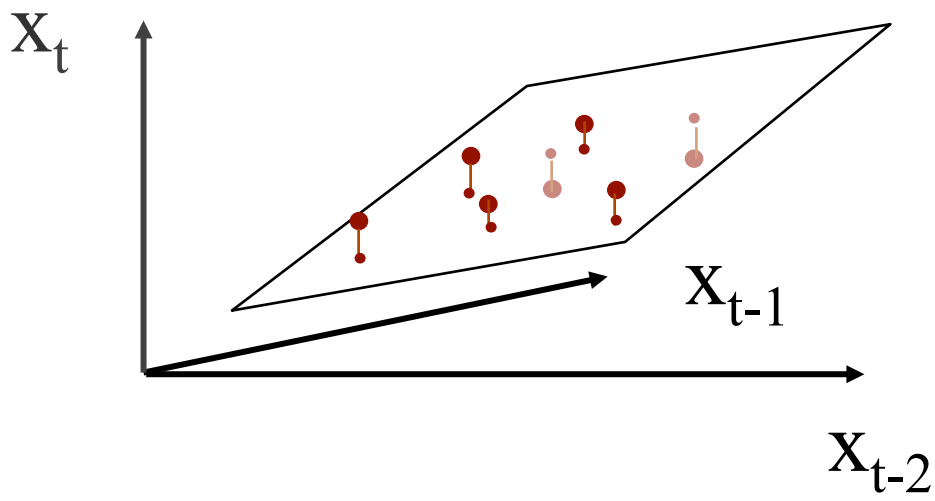
More details:

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



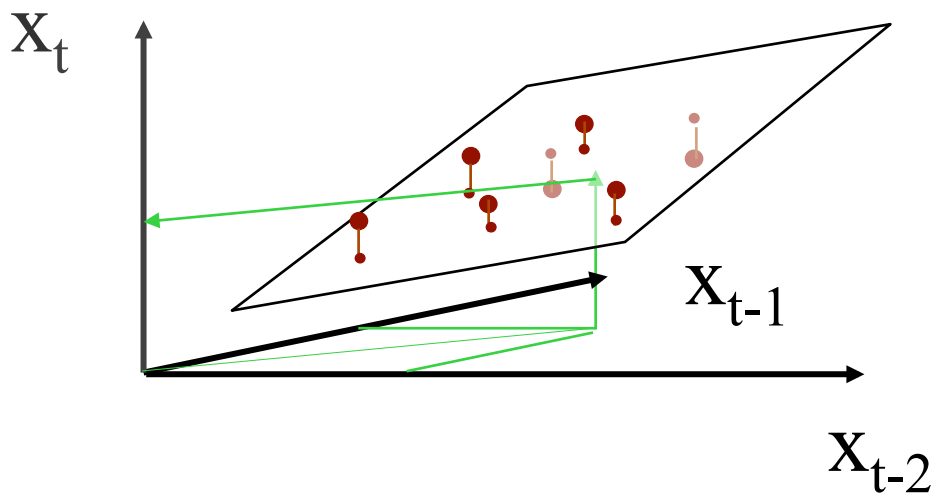
More details:

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



More details:

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



More details: DETAILS

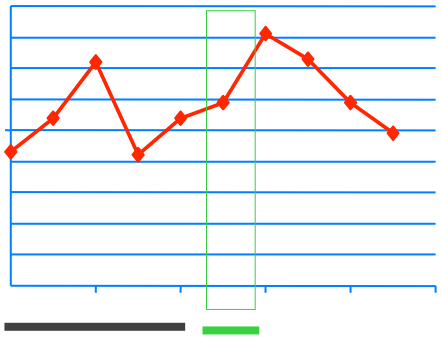
- Q1: Can it work with window w>1?
- A1: YES! The problem becomes:

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

- 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



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

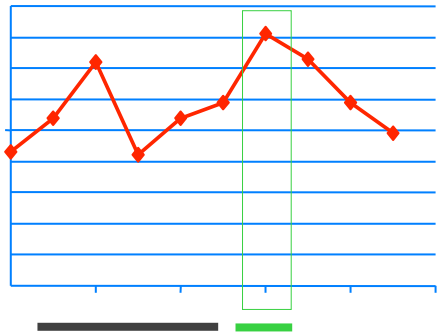
Ind-var1 Ind-var-w

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**

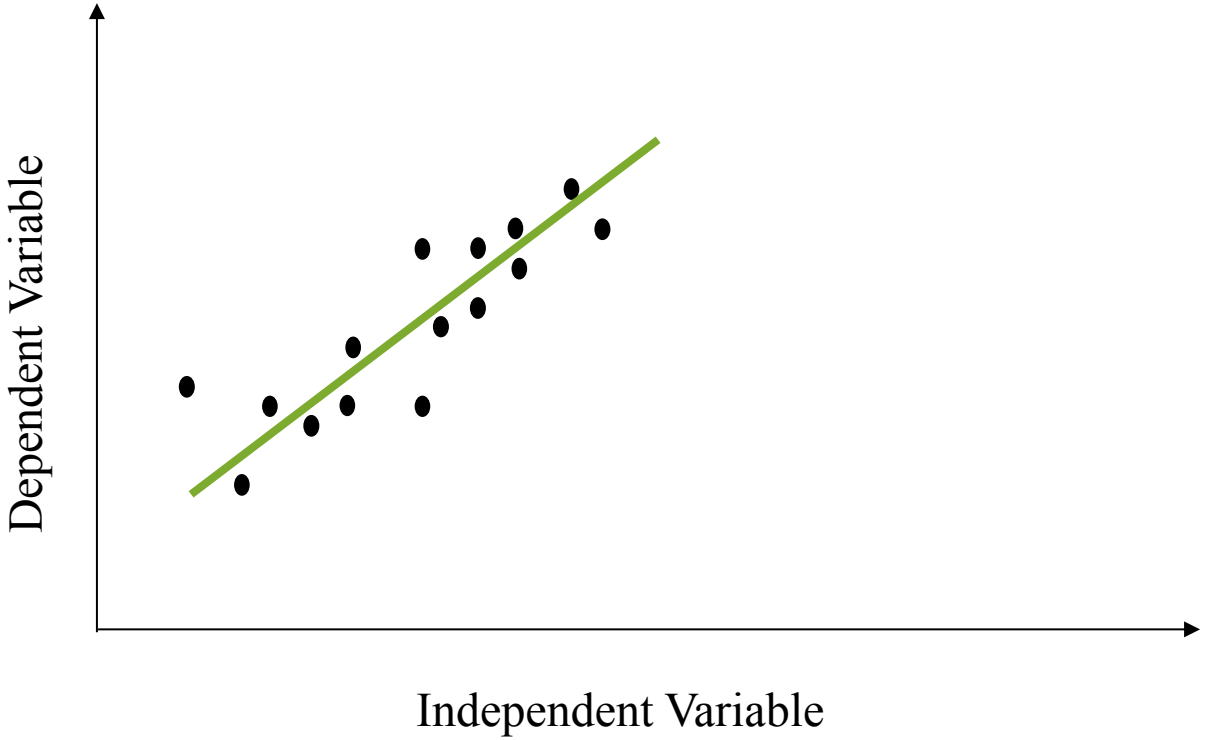
Even more details DETAILS

- 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.

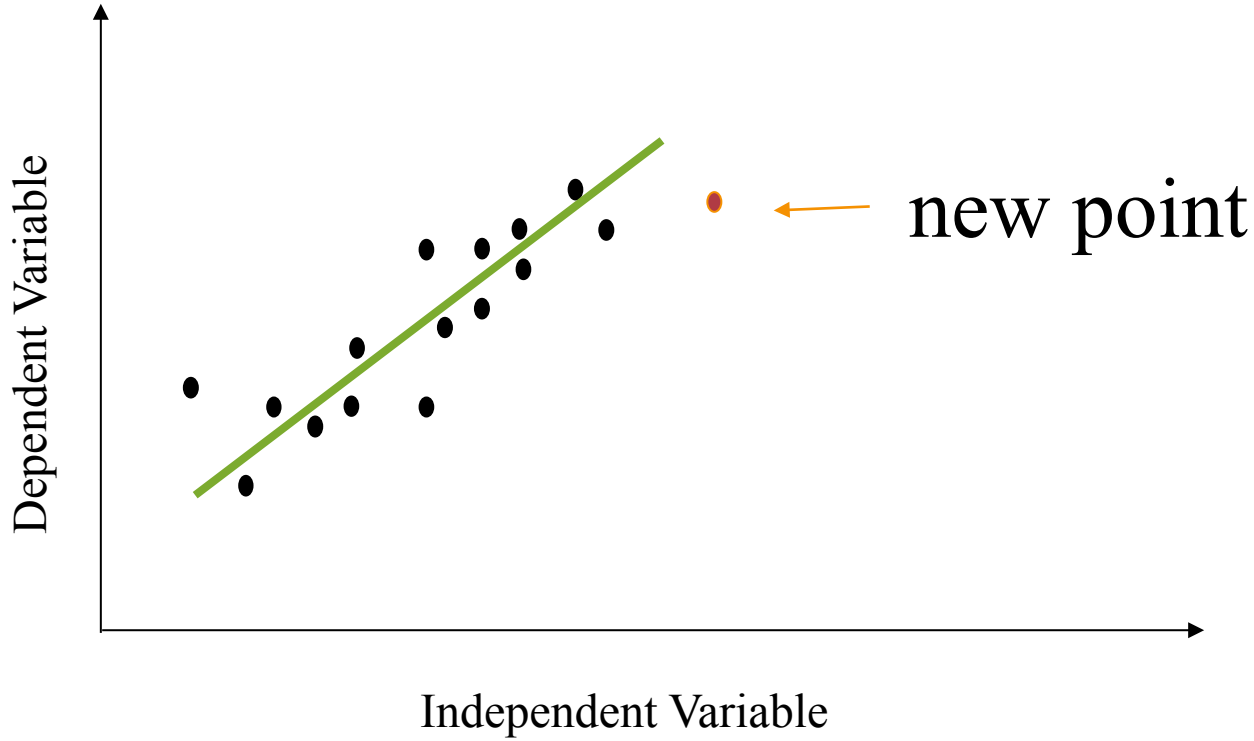
Even more details

- Given:



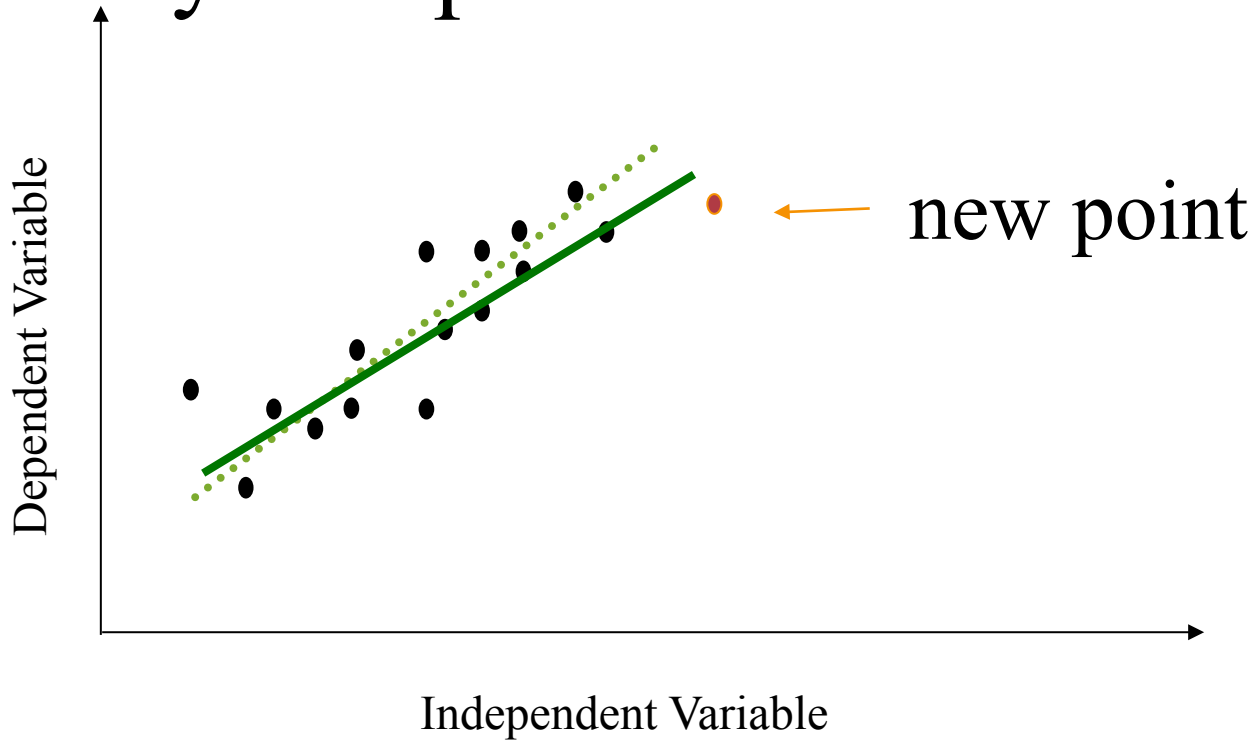
Even more details

- 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 operat… $O(N \times w^2)$
- Recursi…
  - …naller, fixed …x $O(w \times w)$
  - …, incremental computation $O(1 \times w^2)$

49,0… → 49

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

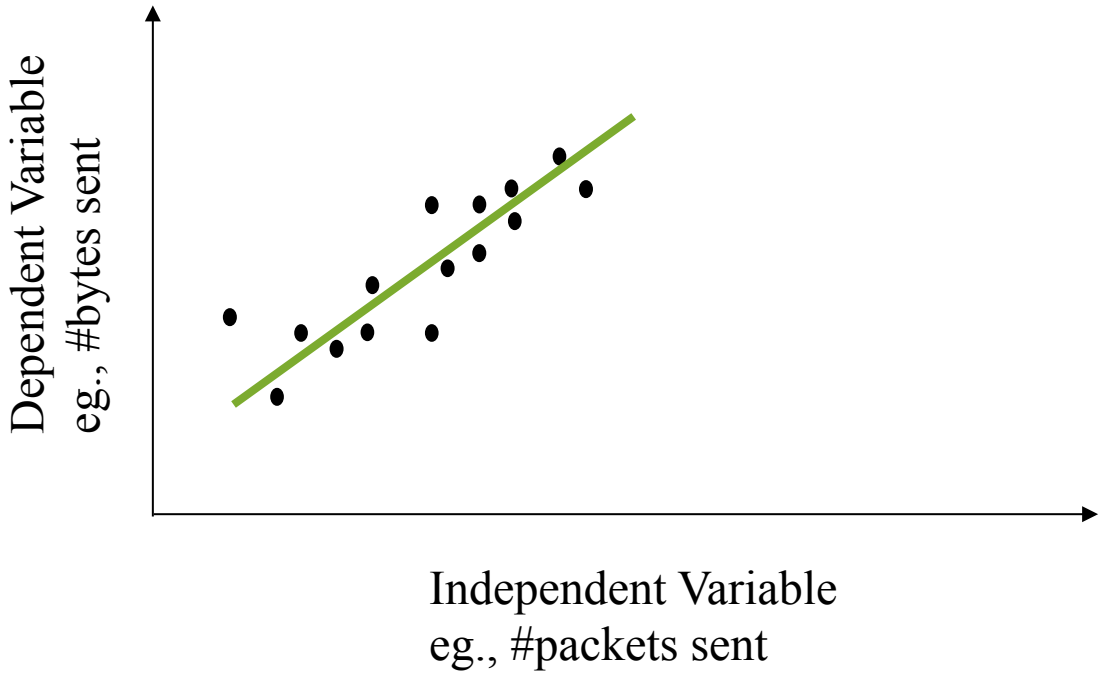
RLS: GREAT for streams

Even more detail DETAILS

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

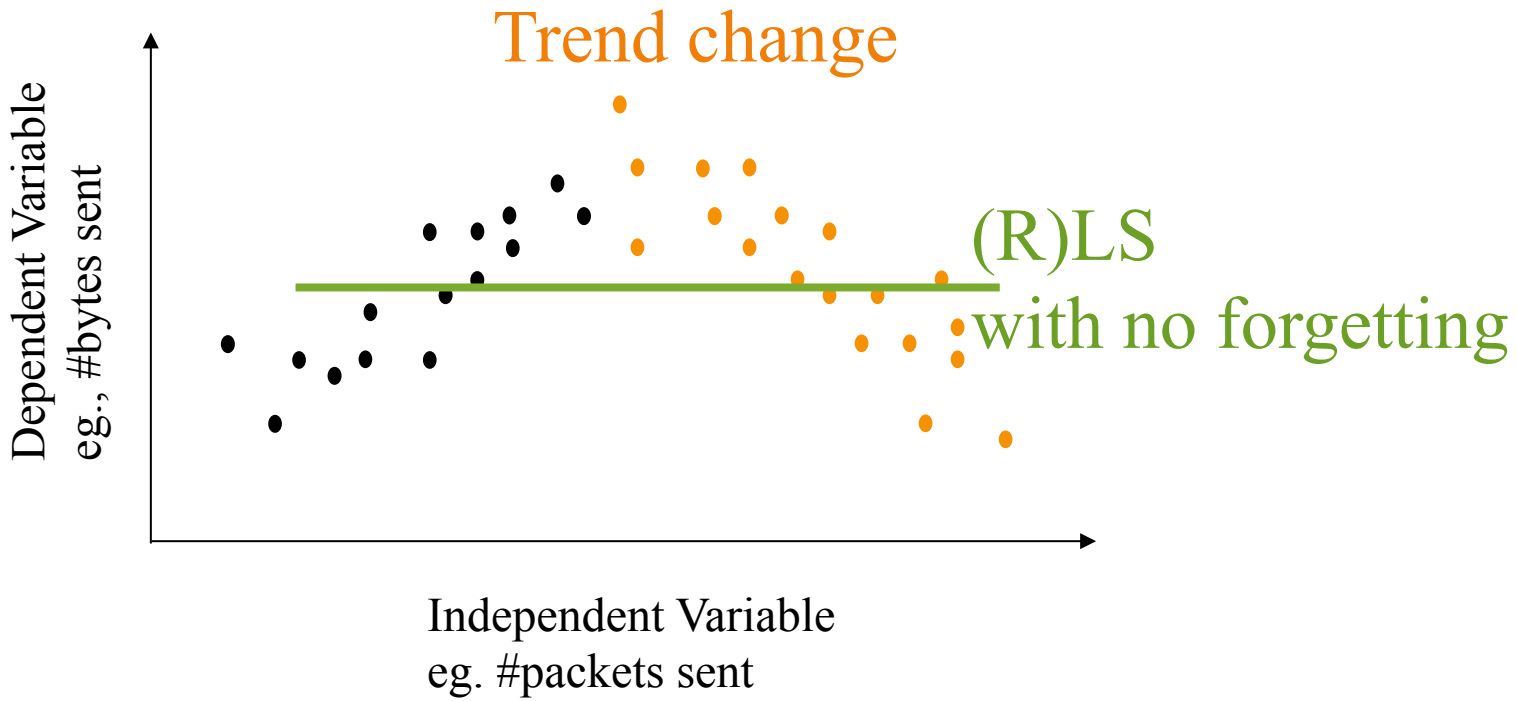
Adaptability - 'forgetting'

DETAILS



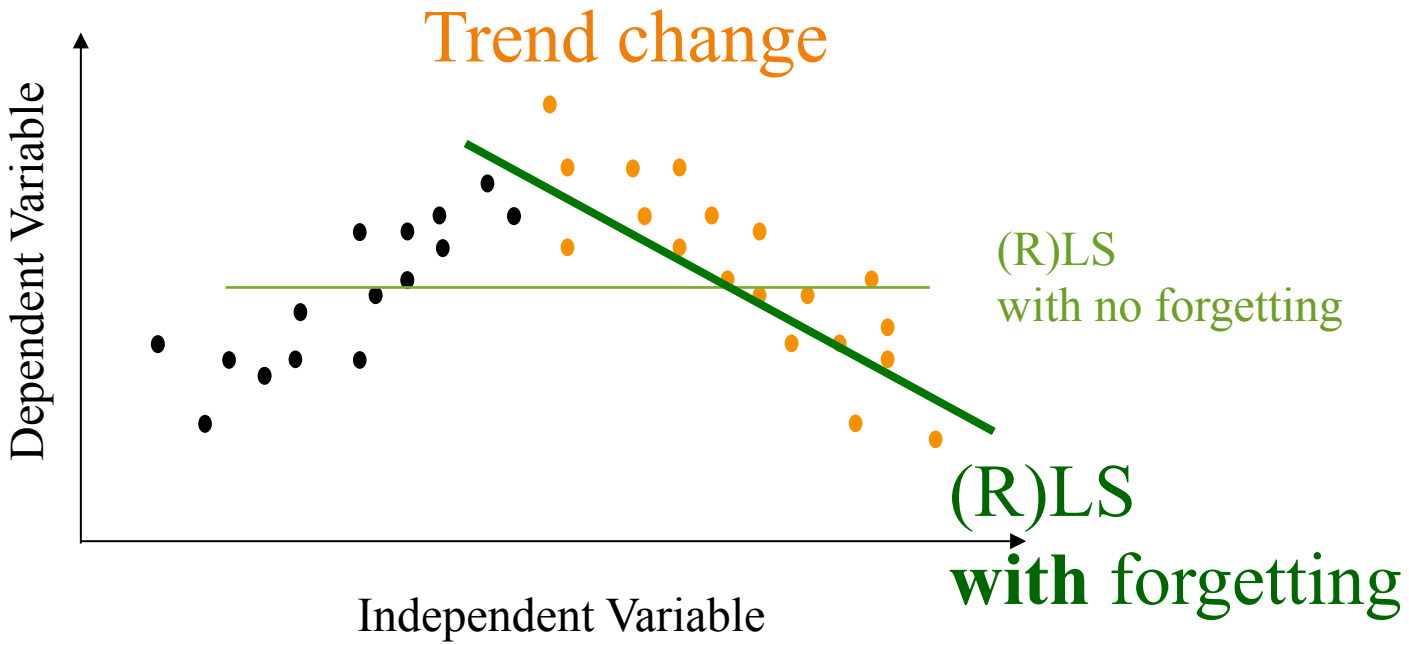
Adaptability - 'forgetting'

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

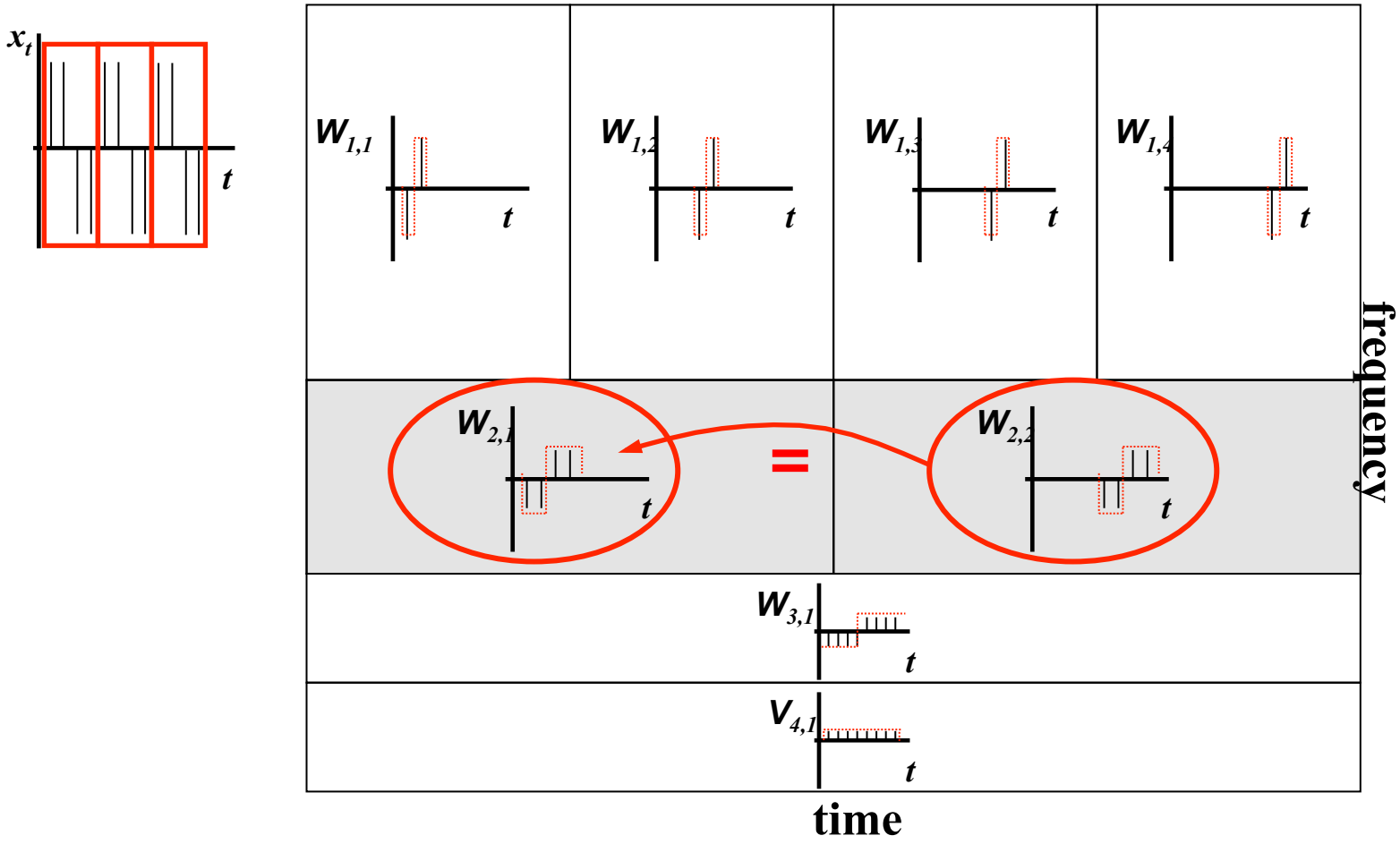
How to choose '*w*'?

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

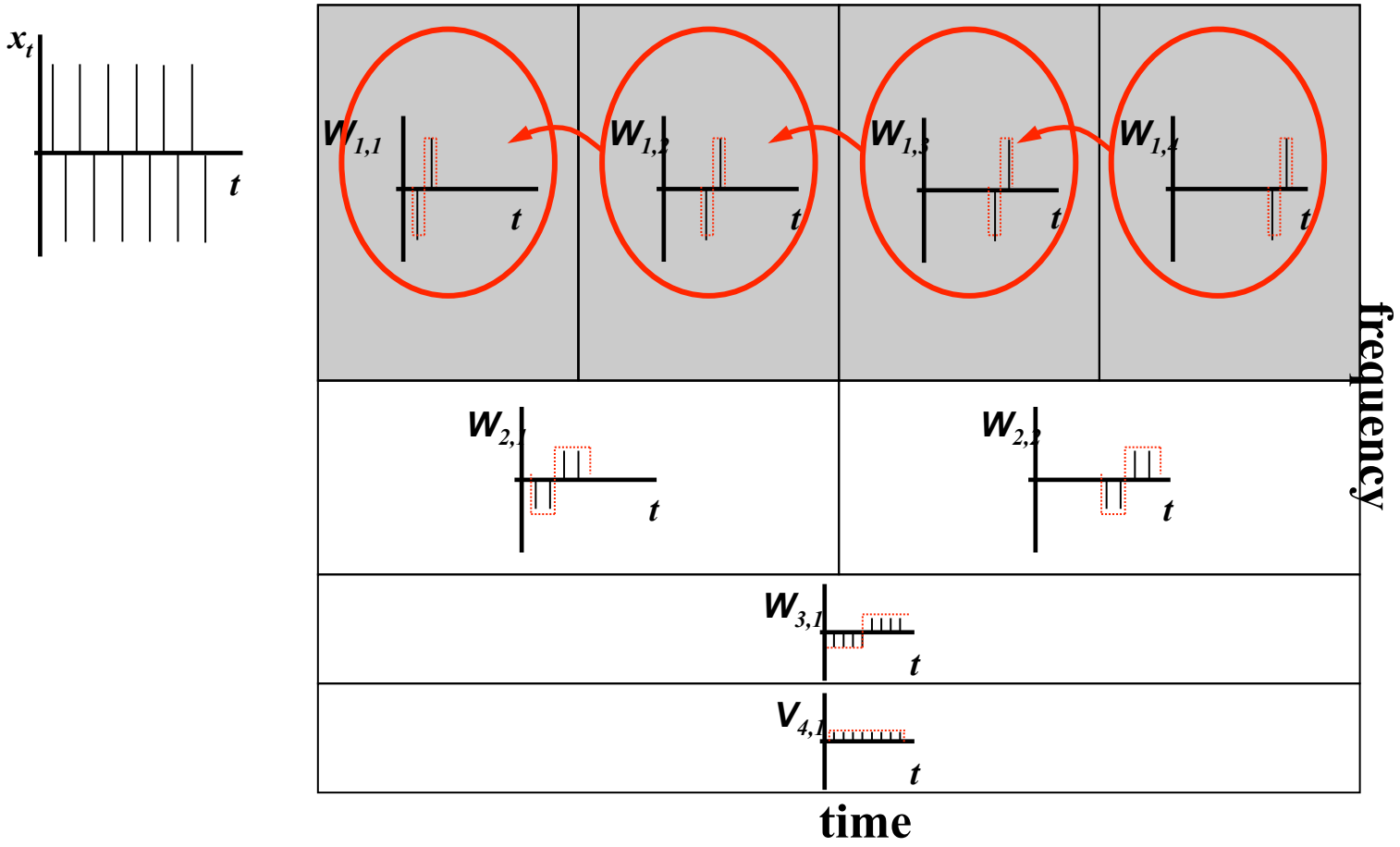
Answer:

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

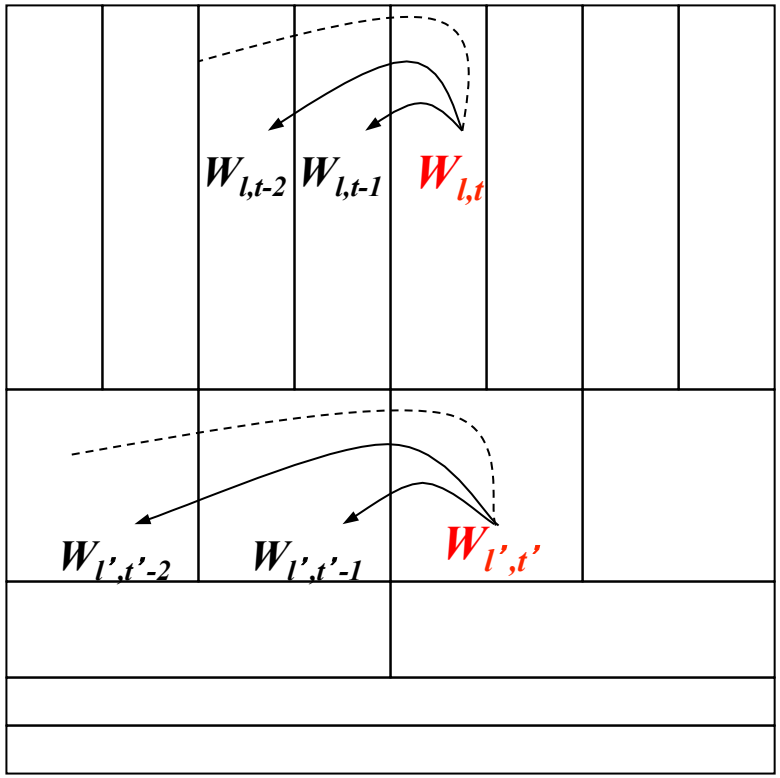
AWSOM



AWSOM



AWSOM - idea



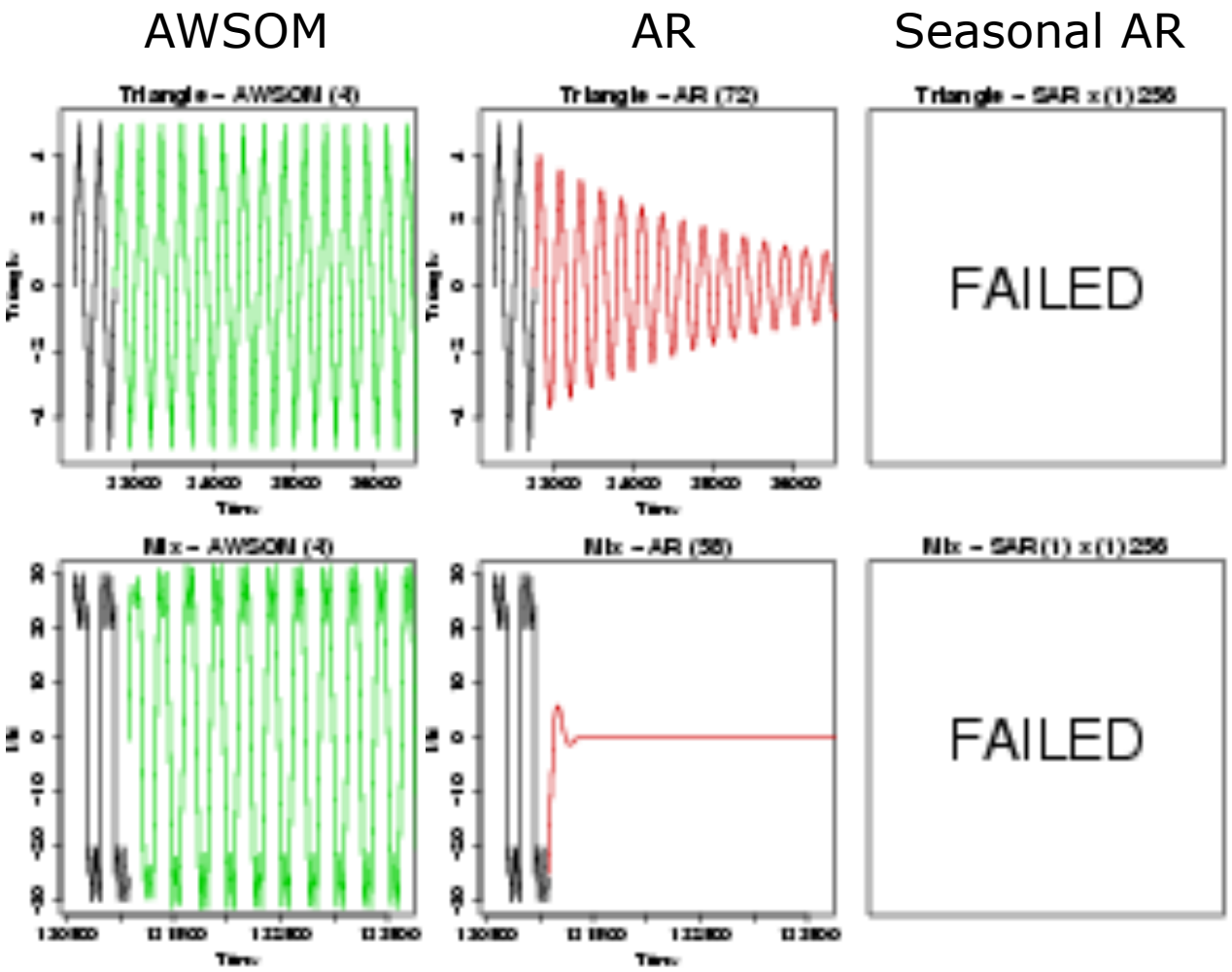
$$W_{l,t} = \beta_{l,1} W_{l,t-1} + \beta_{l,2} W_{l,t-2} + \dots$$

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

More details…

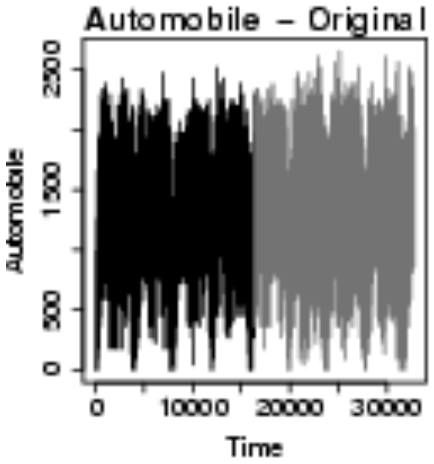
- Update of wavelet coefficients (incremental)
- Update of linear models (incremental; RLS)
- Feature selection (single-pass)
  - Not all correlations are significant
  - Throw away the insignificant ones ("noise")

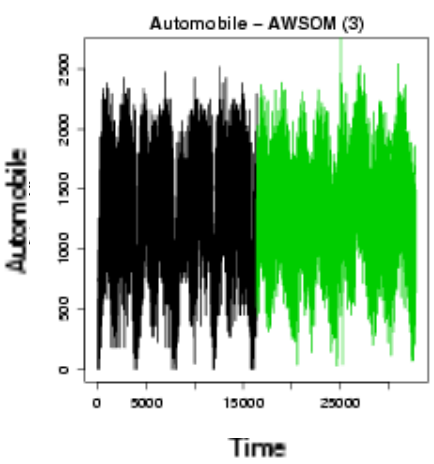
Results - Synthetic data

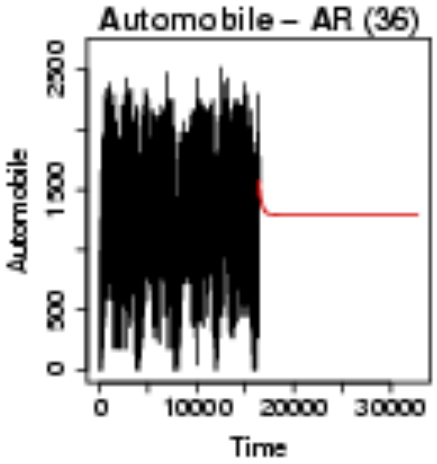


- Triangle pulse
- Mix (sine + square)
- AR captures wrong trend (or none)
- Seasonal AR estimation fails

Results - Real data

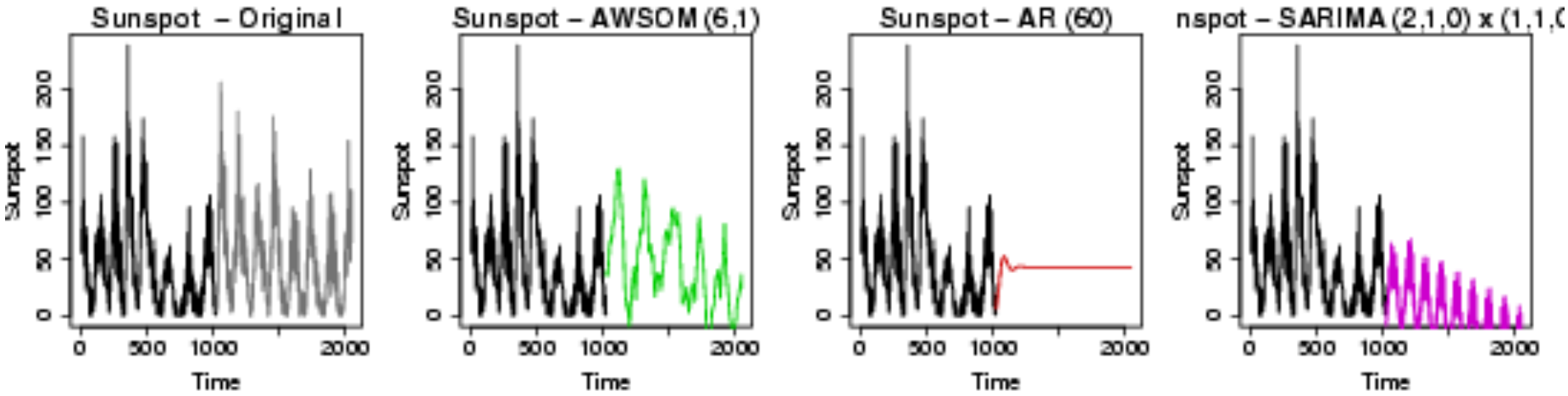






- Automobile traffic
  - Daily periodicity
  - Bursty "noise" at smaller scales
- AR fails to capture any trend
- Seasonal AR estimation fails

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

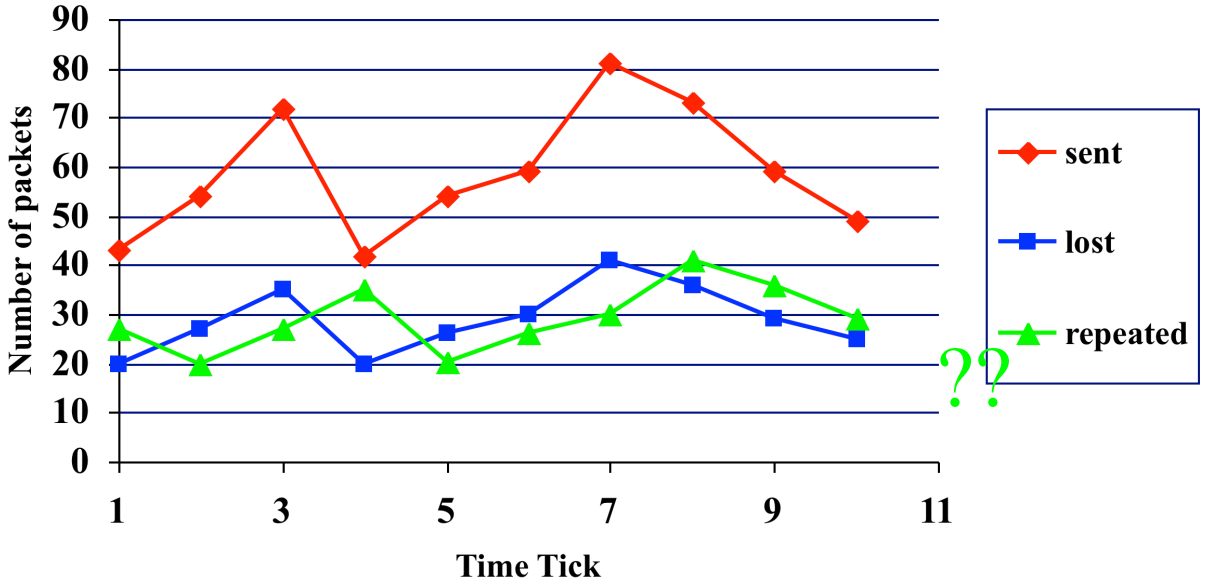
- Where
  - N: number of points (so far)
  - k: number of regression coefficients; fixed
  - m: number of linear models; $O(lgN)$

Roadmap

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

Co-Evolving Time Sequences

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



Solution:

Q: what should we do?

Solution:

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)

Part 1

Similarity search, pattern discovery and summarization

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

Roadmap

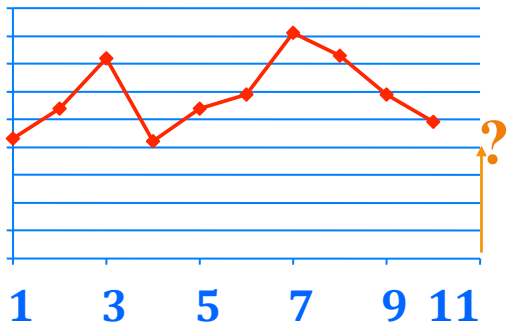
- Motivation
- Similarity Search and Indexing
- Feature extraction
  - DFT, DWT, DCT (data independent)
  - SVD, ICA (data independent)
  - (MDS, FastMap)
- Streaming pattern discovery

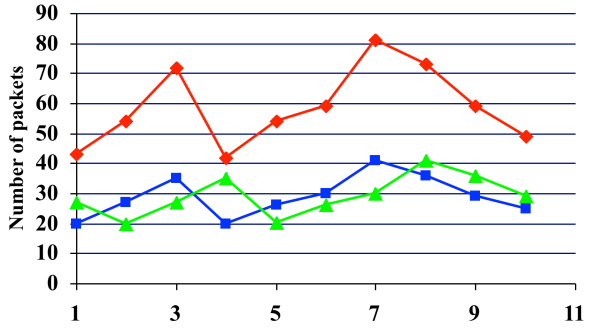
Roadmap

- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
  - Definition of DWT and properties
  - how to read the DWT scalogram

Wish list

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



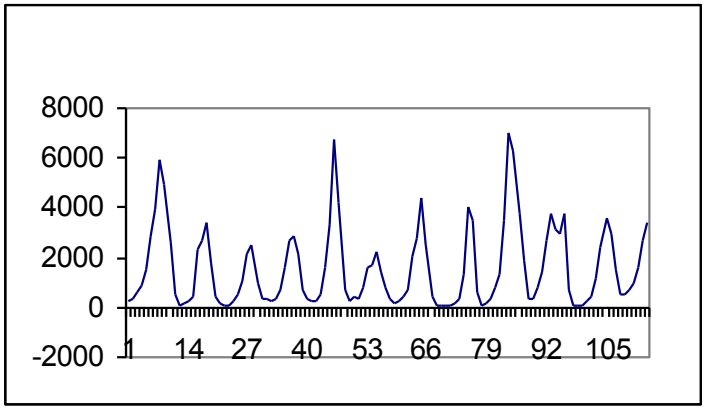


Introduction - Problem#1

Goal: given a signal (eg., packets over time)

Find: patterns and/or compress

count



lynx caught per year
(packets per day;
automobiles per hour)

year

DFT: definition DETAILS

- 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: definition

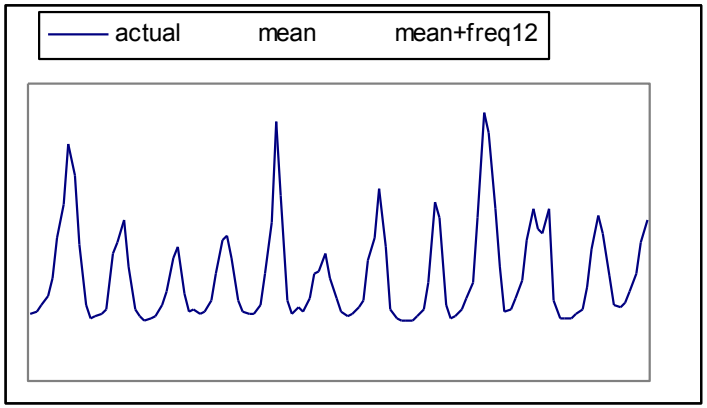
- **Good** news: Available in **all** symbolic math packages, eg., in ‘mathematica’

```
x = [1,2,1,2];
X = Fourier[x];
Plot[ Abs[X] ];
```

DFT: Amplitude spectrum

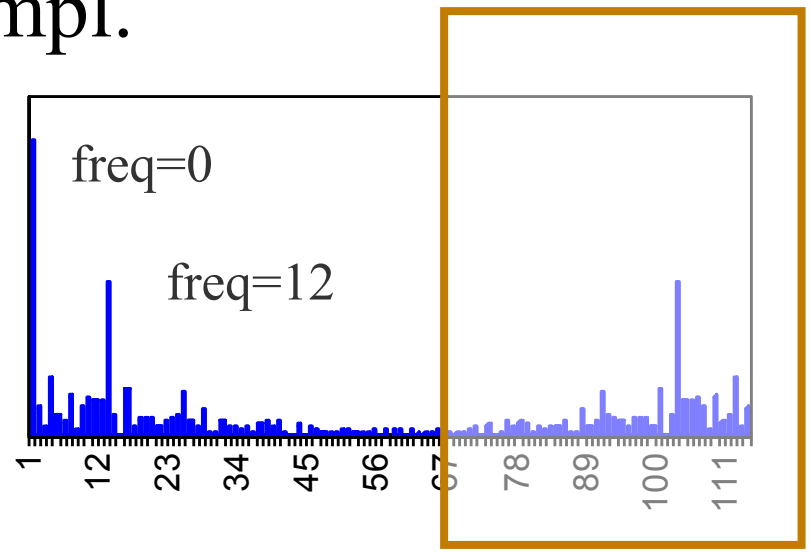
Amplitude: $A_f^{\ 2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



year

Ampl.

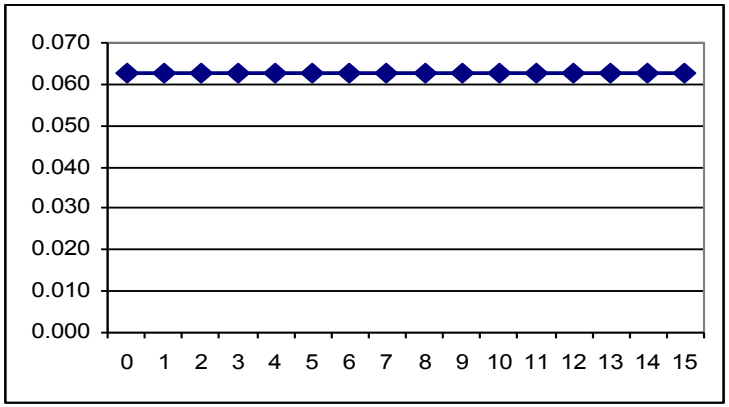


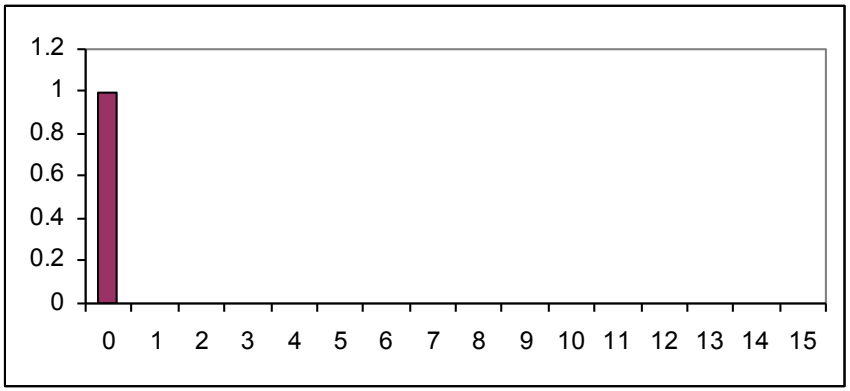
Freq.

DFT: examples

flat

Amplitude



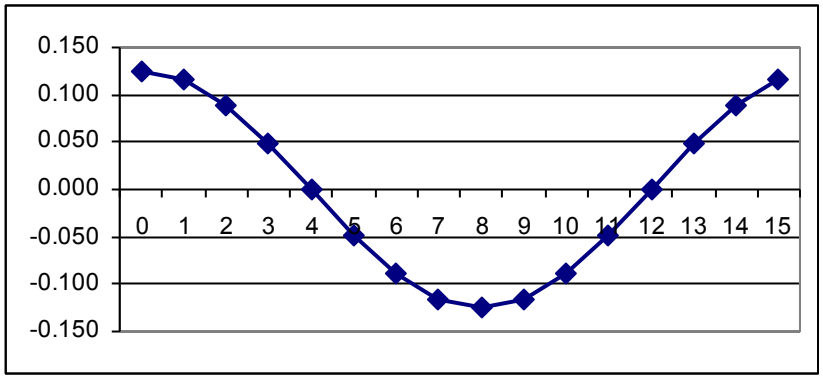


time

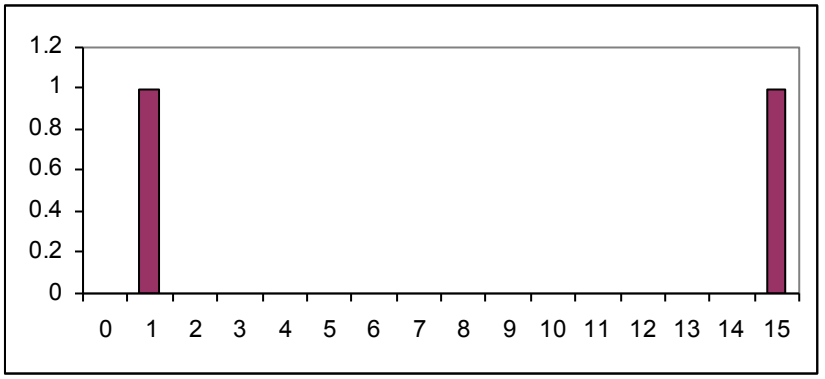
freq

DFT: examples

Low frequency sinusoid



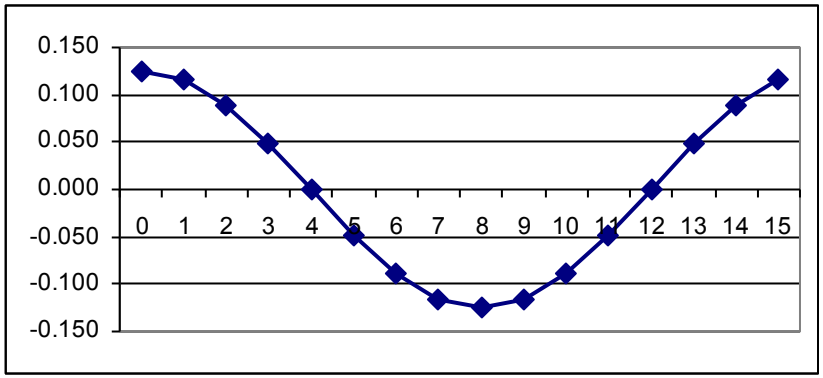
time



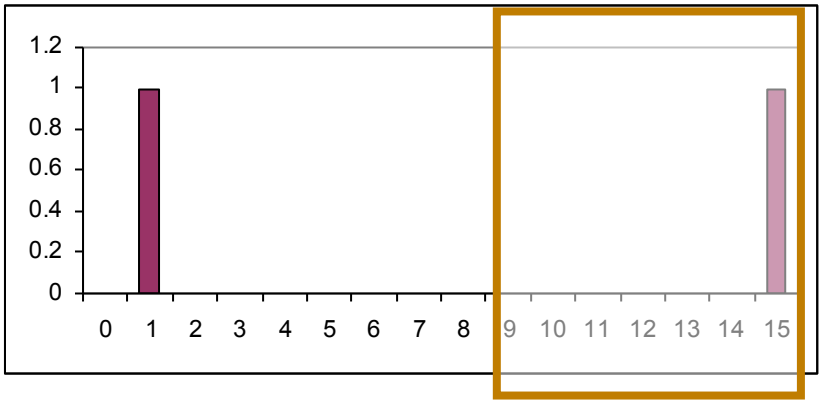
freq

DFT: examples

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



time

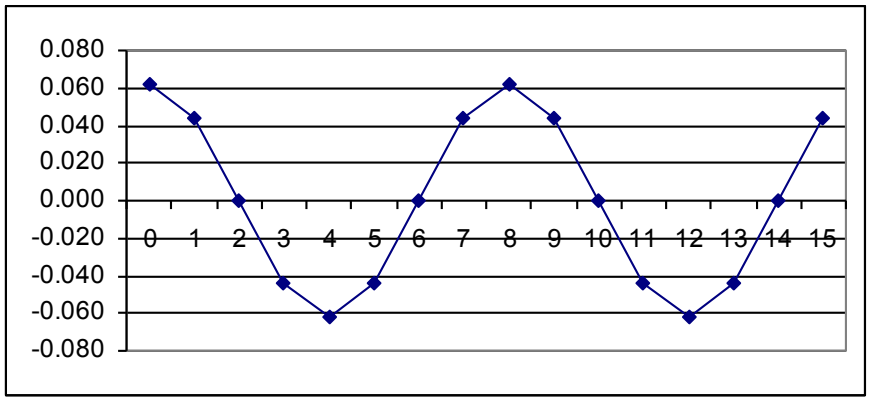


freq

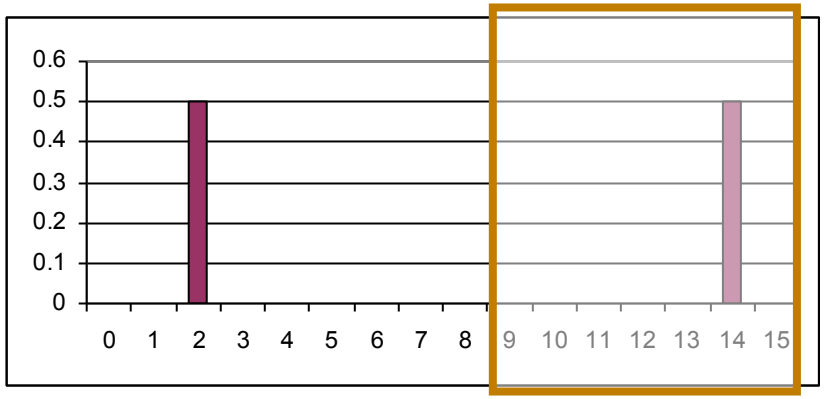
Skip

DFT: examples

- Higher freq. sinusoid



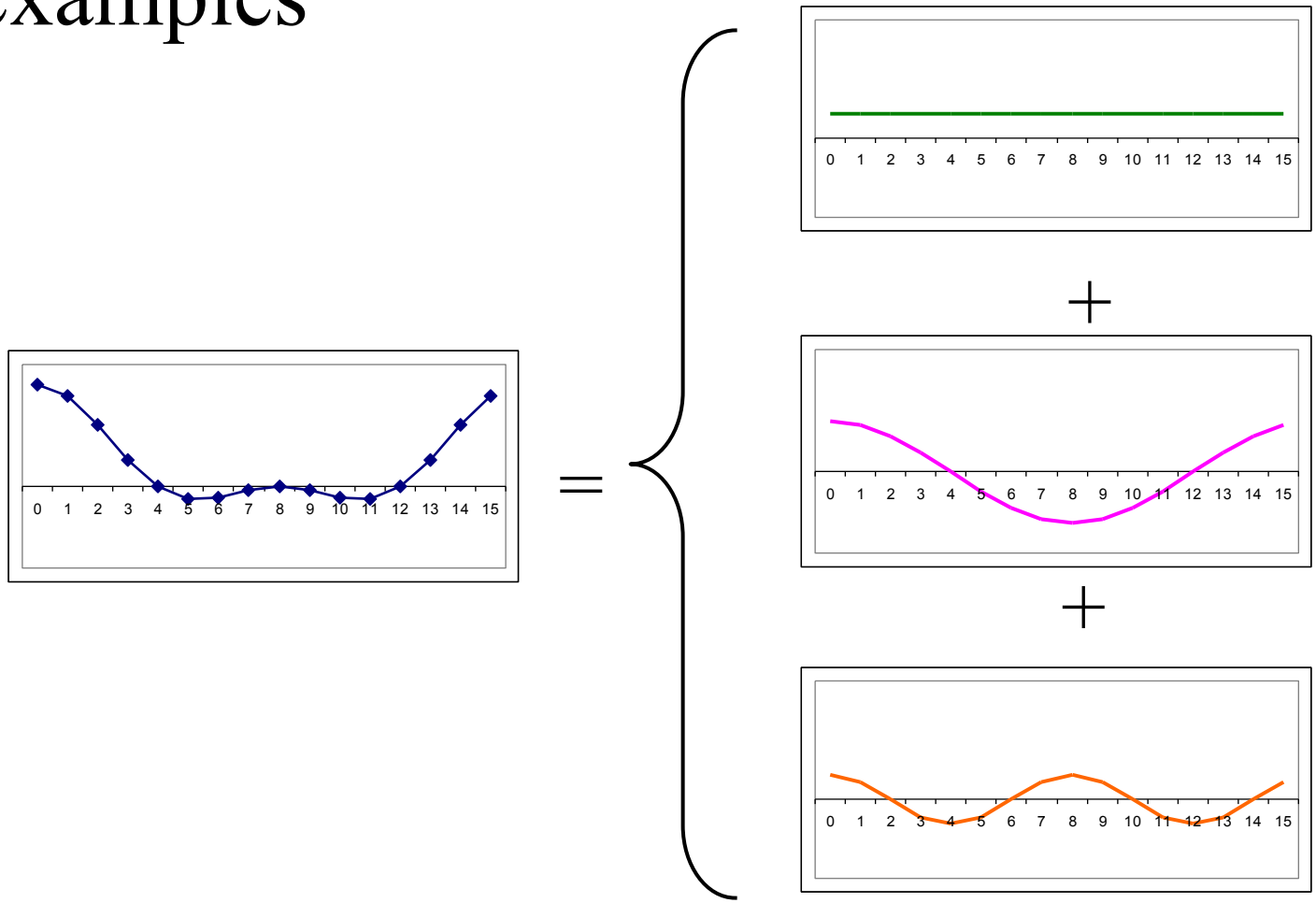
time



freq

DFT: examples

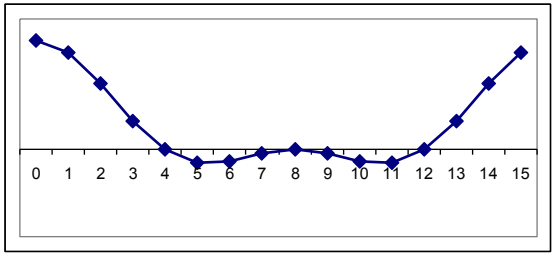
examples

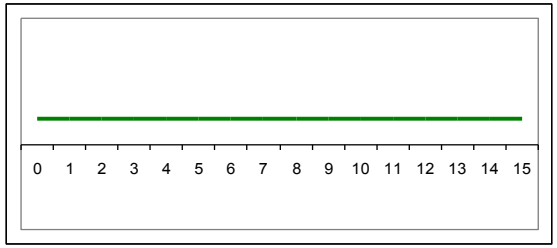


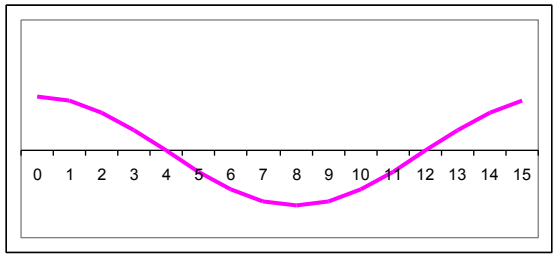
DFT: examples

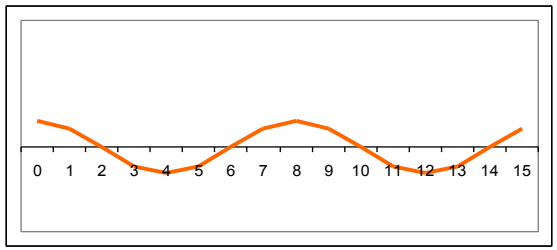
Skip

examples

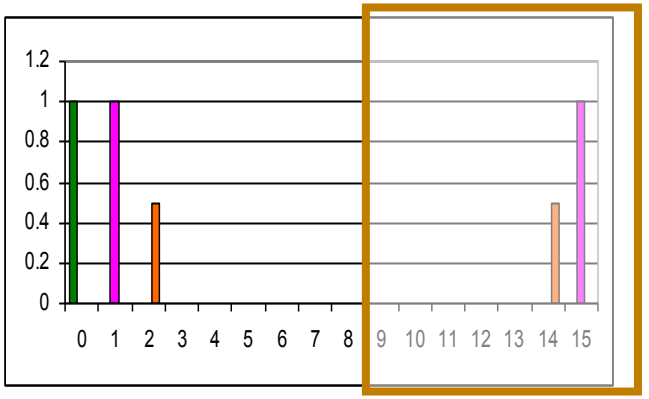








Ampl.



Freq.

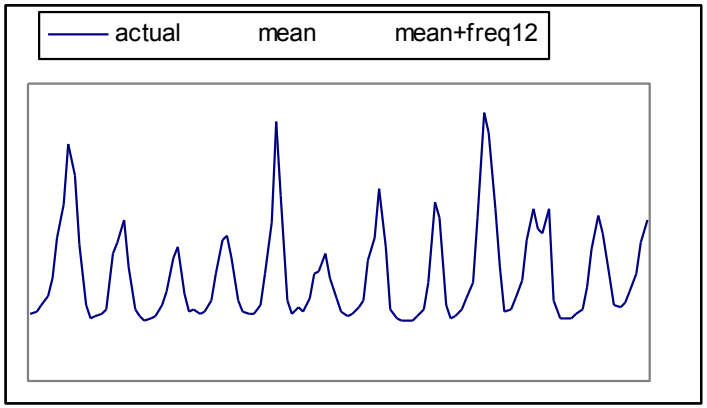
Roadmap

- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
  - Definition of DWT and properties
  - how to read the DWT scalogram

DFT: Amplitude spectrum

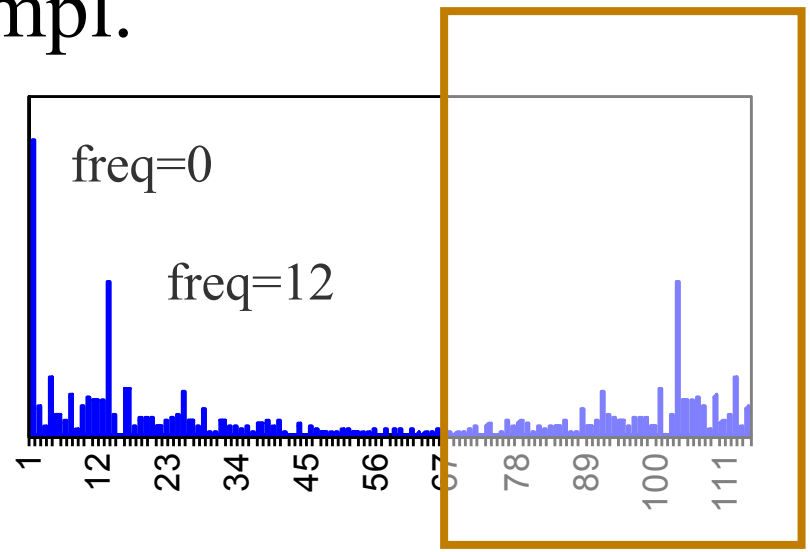
Amplitude: $A_f^{\ 2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



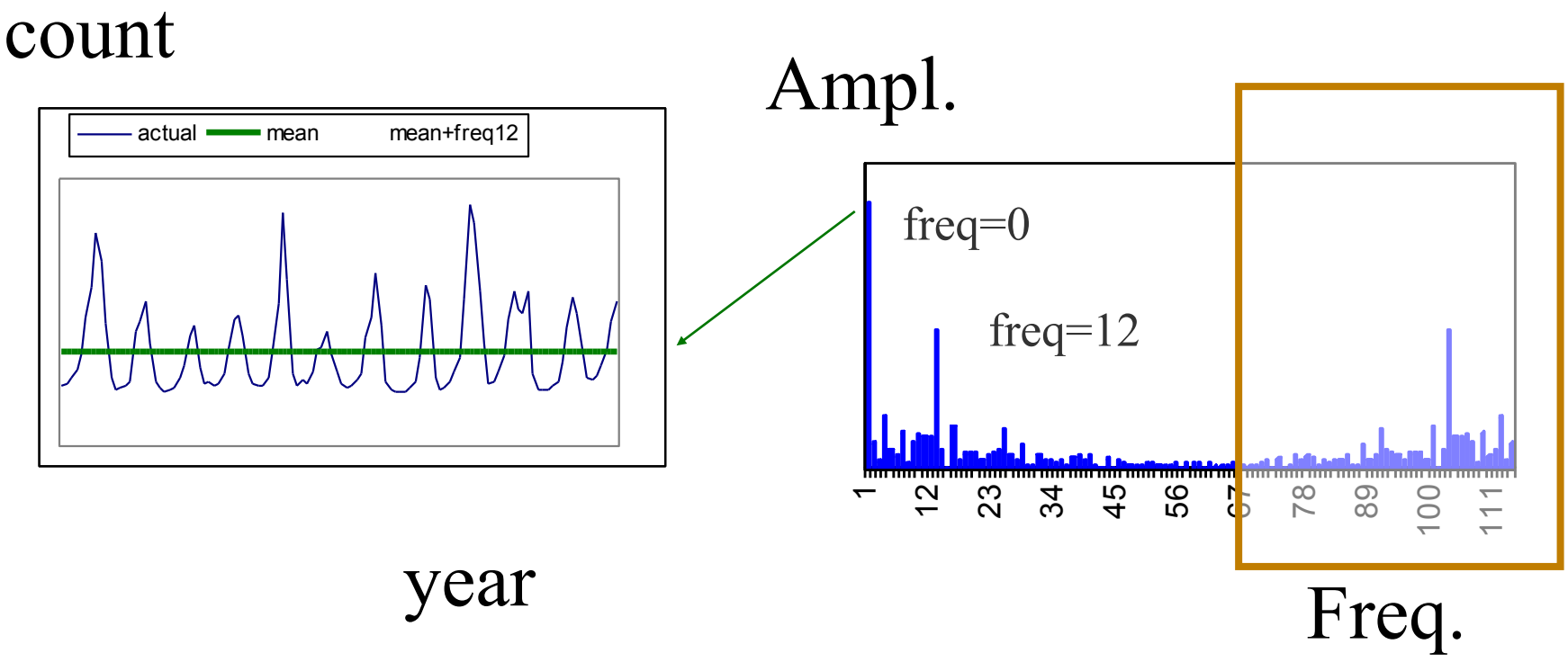
year

Ampl.

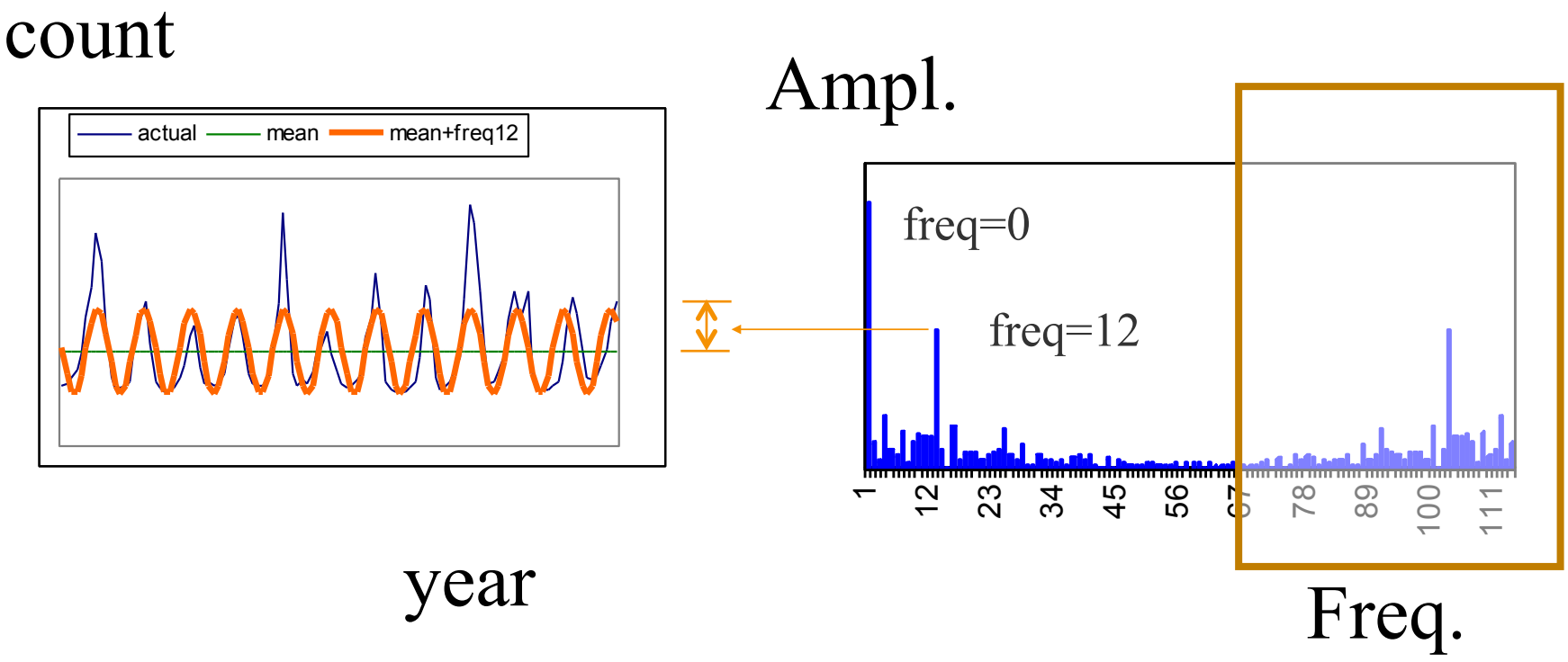


Freq.

DFT: Amplitude spectrum

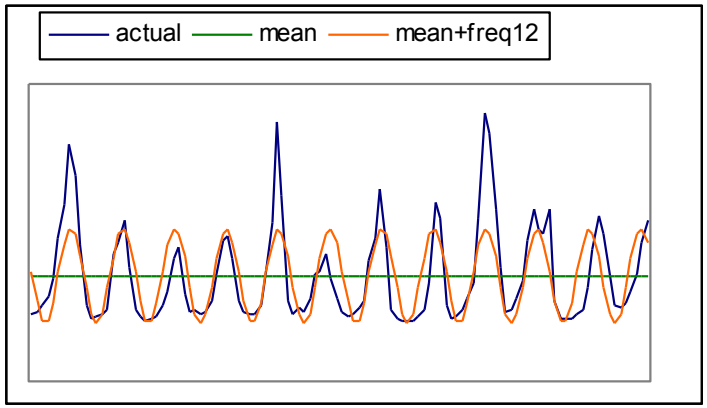


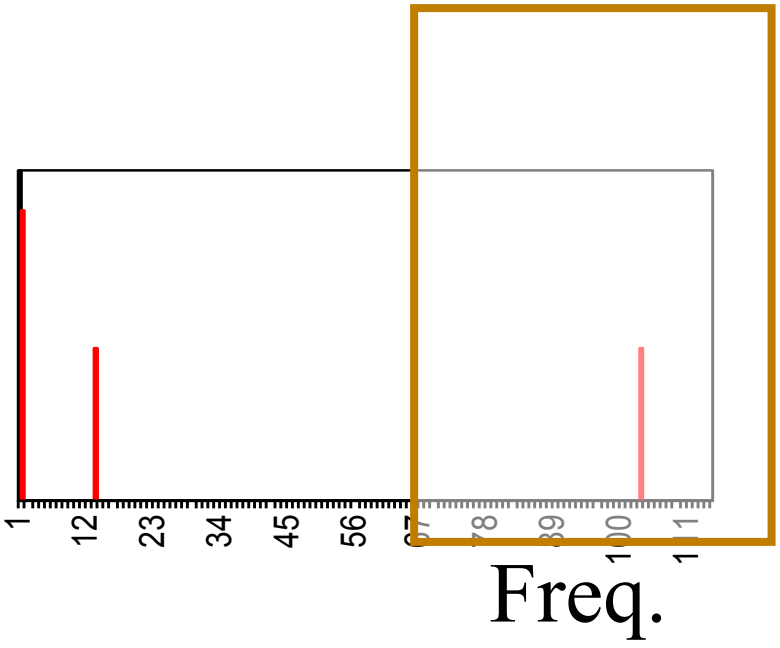
DFT: Amplitude spectrum



DFT: Amplitude spectrum

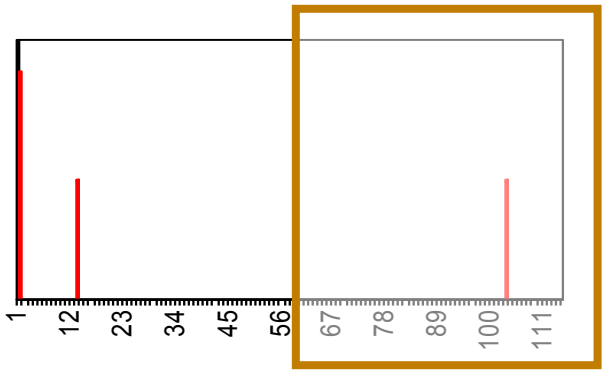
- excellent approximation, with only 2 frequencies!
- so what?





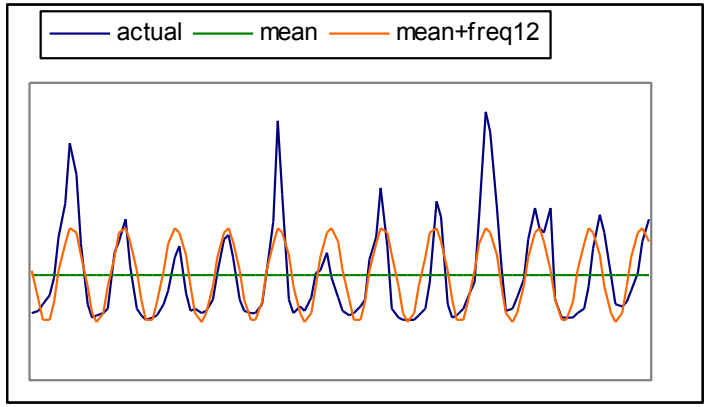
DFT: Amplitude spectrum

- 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**



DFT - Conclusions

- It 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)

Roadmap

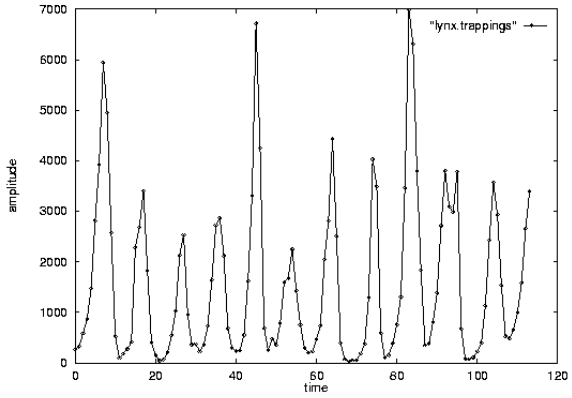
- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
  - Definition of DWT and properties
  - how to read the DWT scalogram

Problem #1:

Goal: given a signal (eg., #packets over time)

Find: patterns, periodicities, and/or **compress**

count



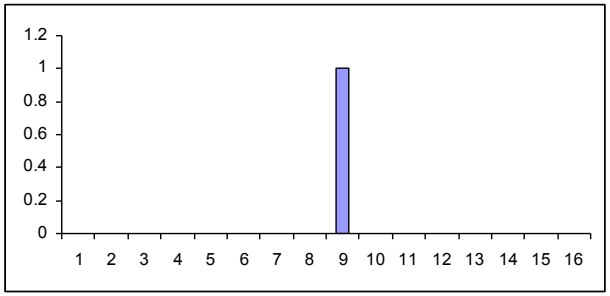
year

lynx caught per year
(packets per day;
virus infections per month)

Wavelets - DWT

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

value

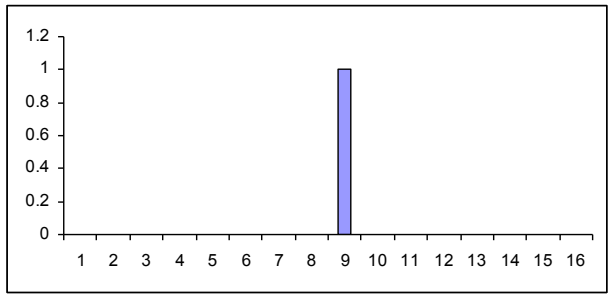


time

Wavelets - DWT

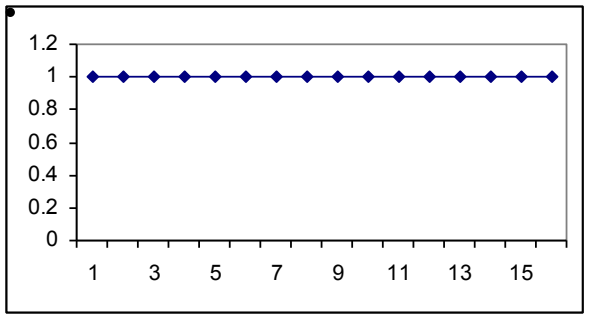
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



time

Ampl.

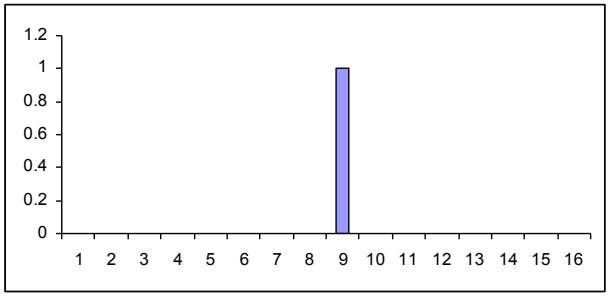


Freq.

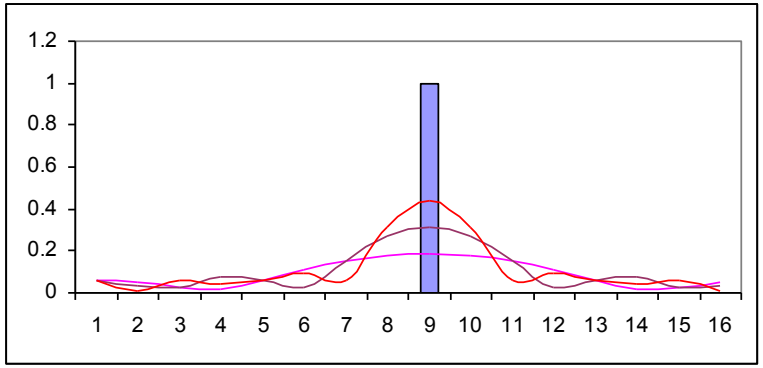
Wavelets - DWT

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

value



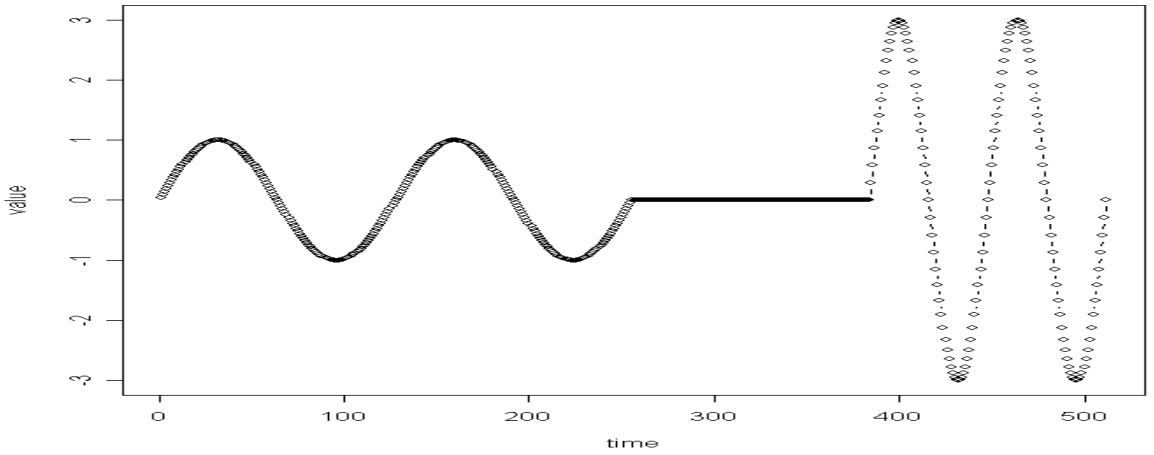
time



Wavelets - DWT

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

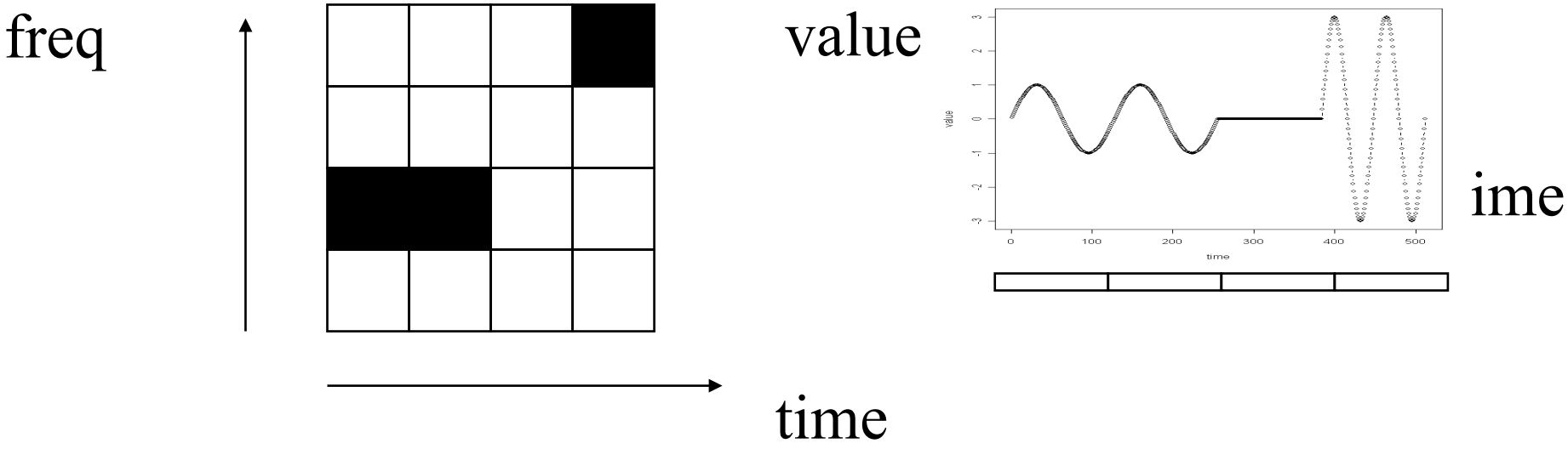
value



time

Wavelets - DWT

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



Wavelets - DWT

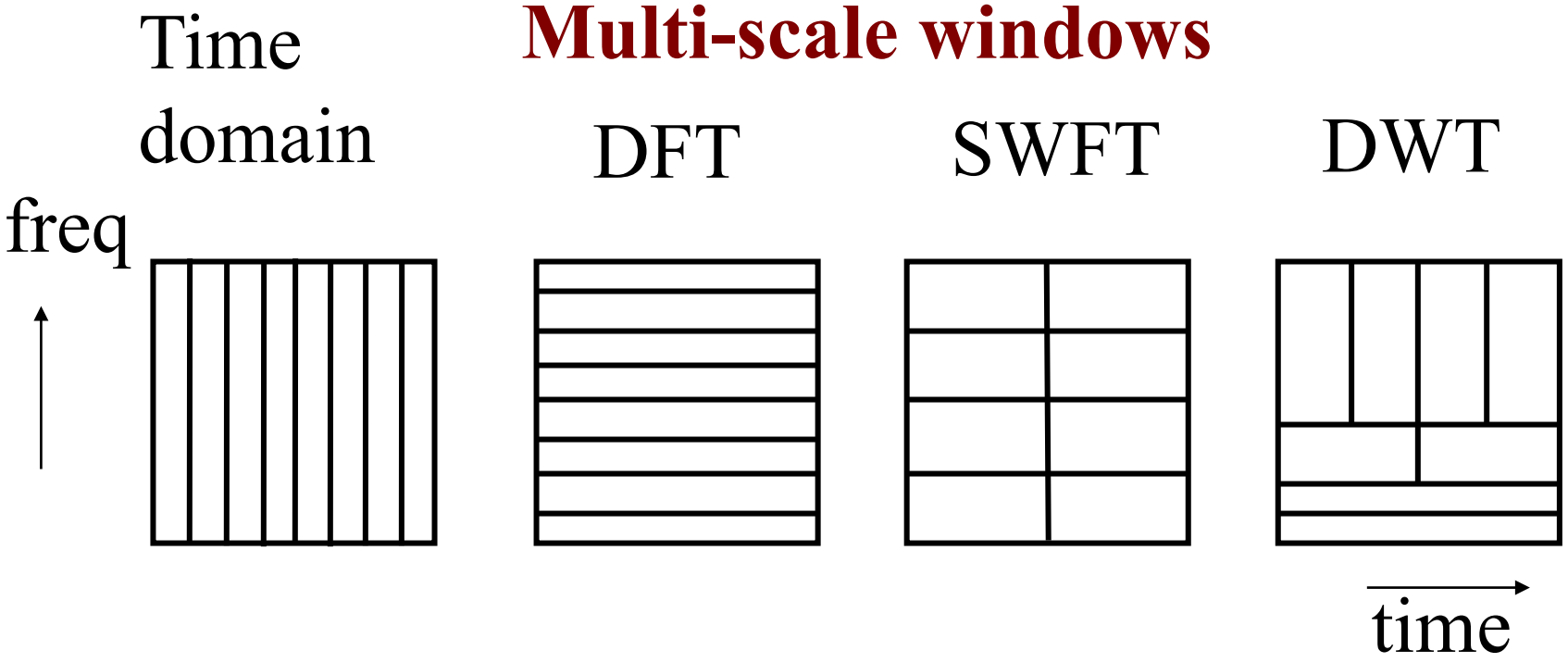
- Answer: **multiple** window sizes! -> DWT

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

Wavelets - DWT

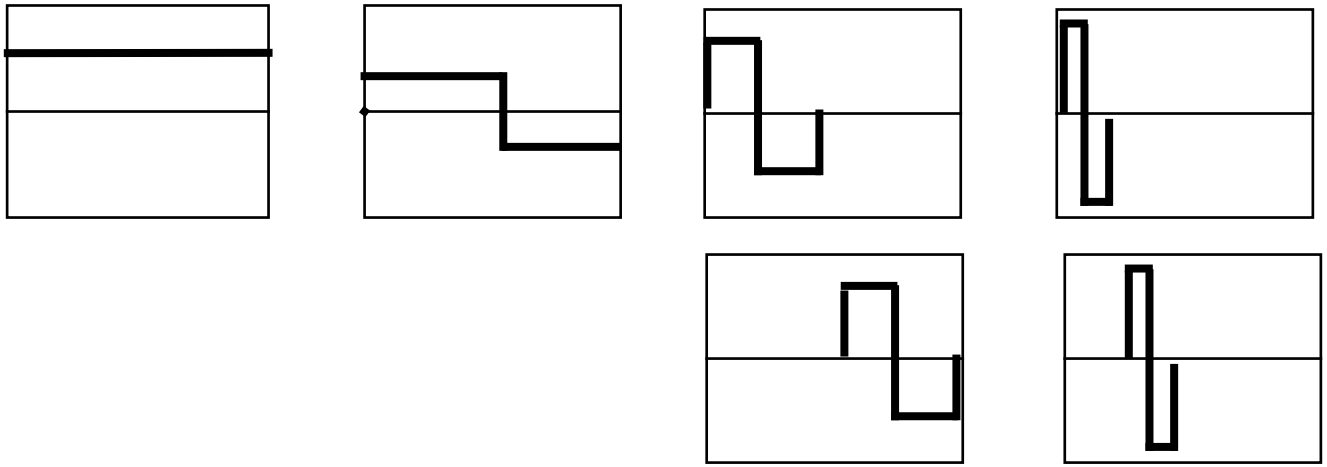
- Answer: **multiple** window sizes! -> DWT

Multi-scale windows



Haar Wavelets

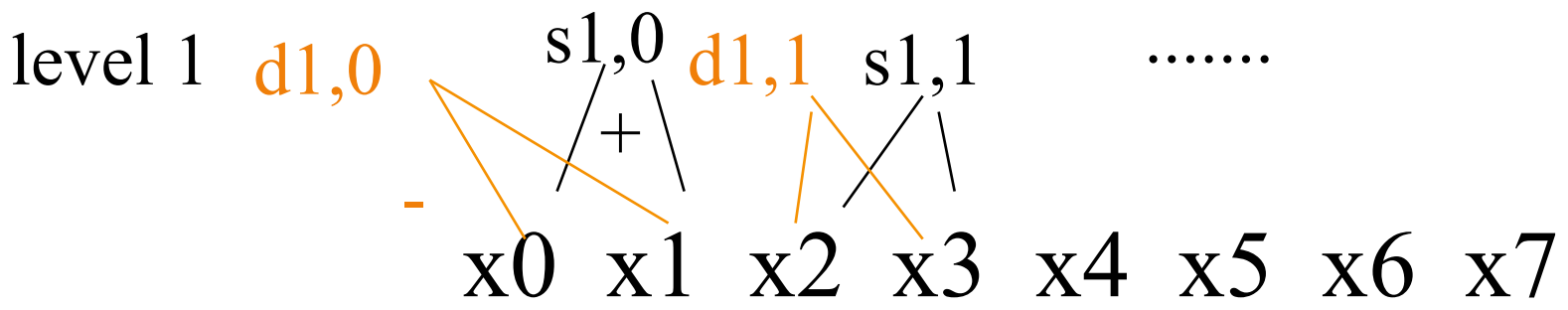
- subtract sum of left half from right half
- repeat recursively for quarters, eight-ths, ...



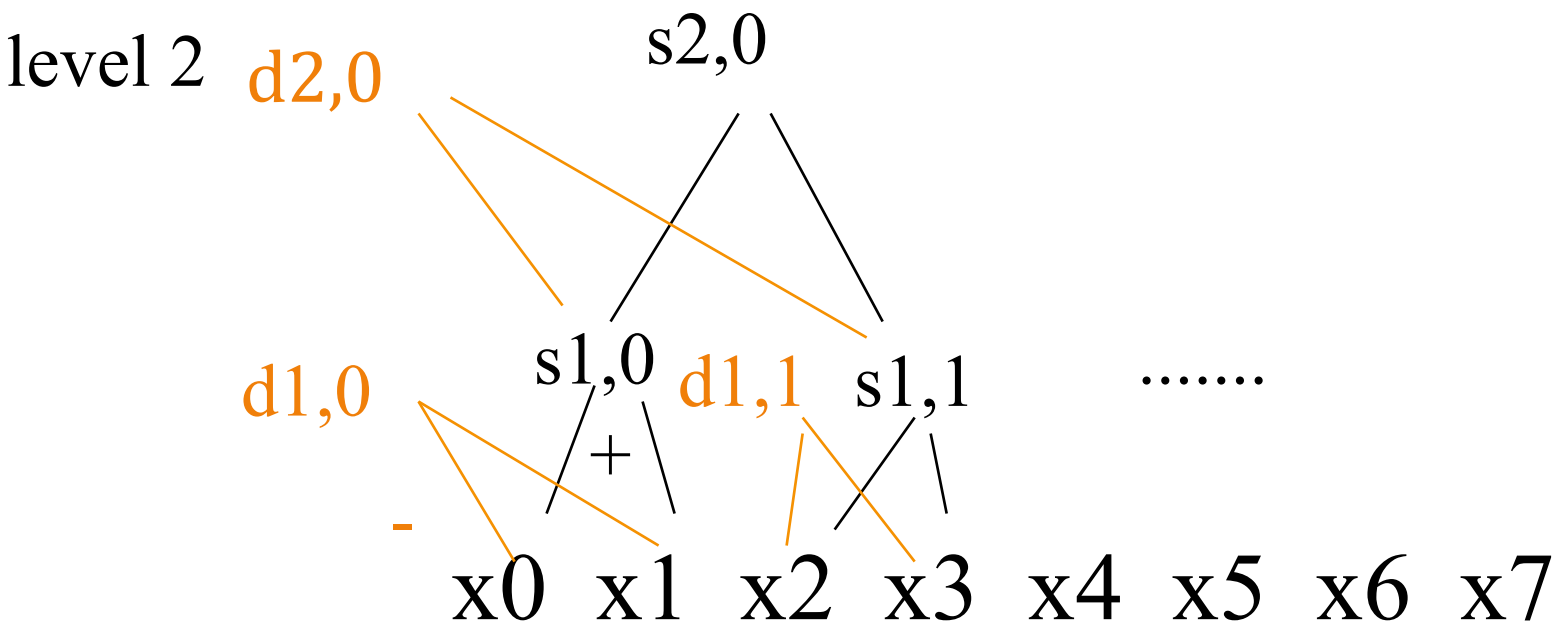
Wavelets - construc DETAILS

x0 x1 x2 x3 x4 x5 x6 x7

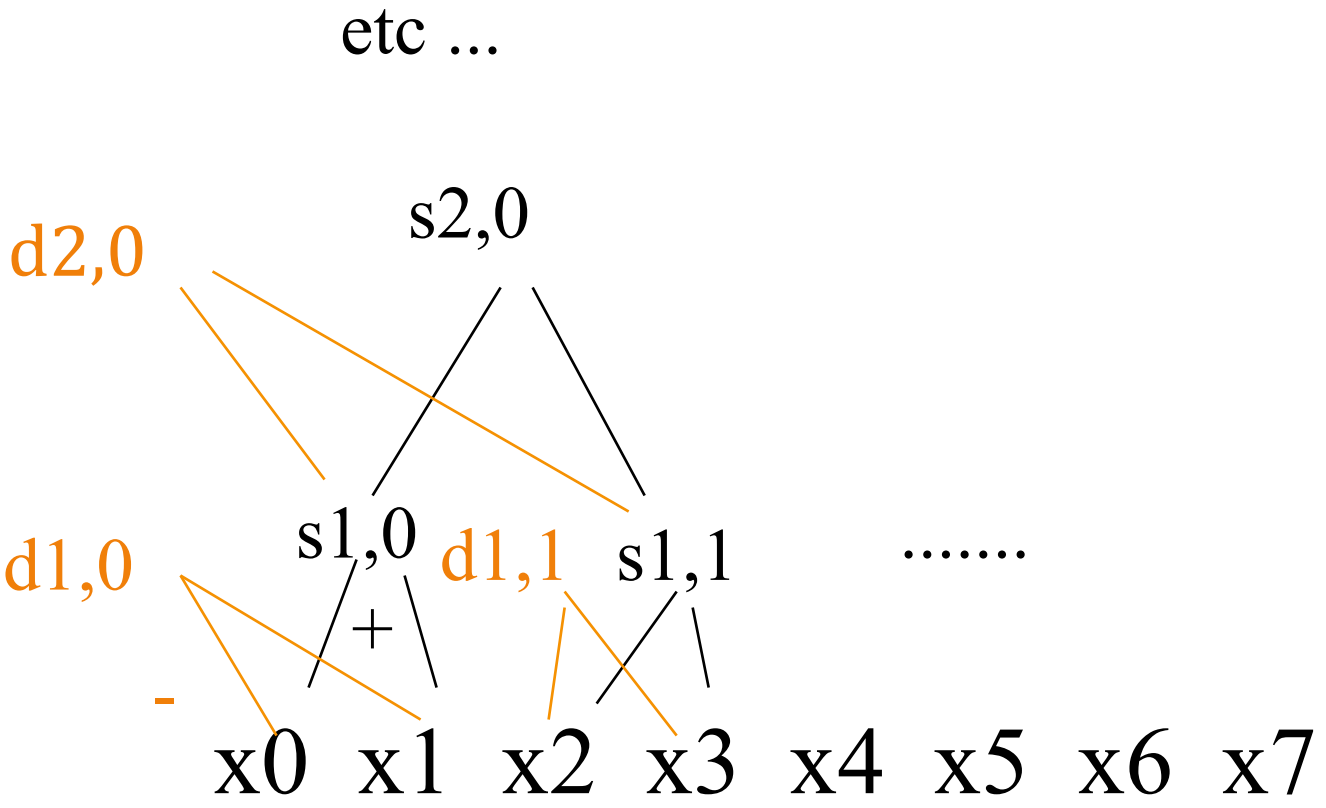
Wavelets - construc DETAILS



Wavelets - construc DETAILS



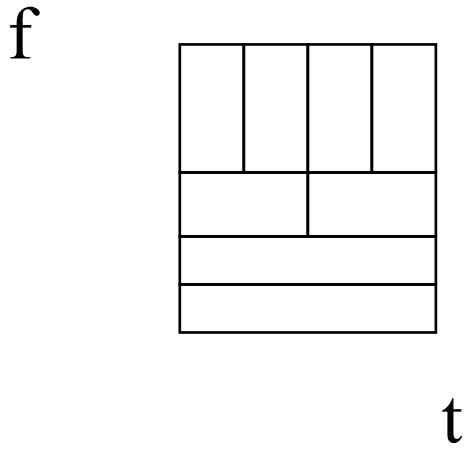
Wavelets - construc DETAILS

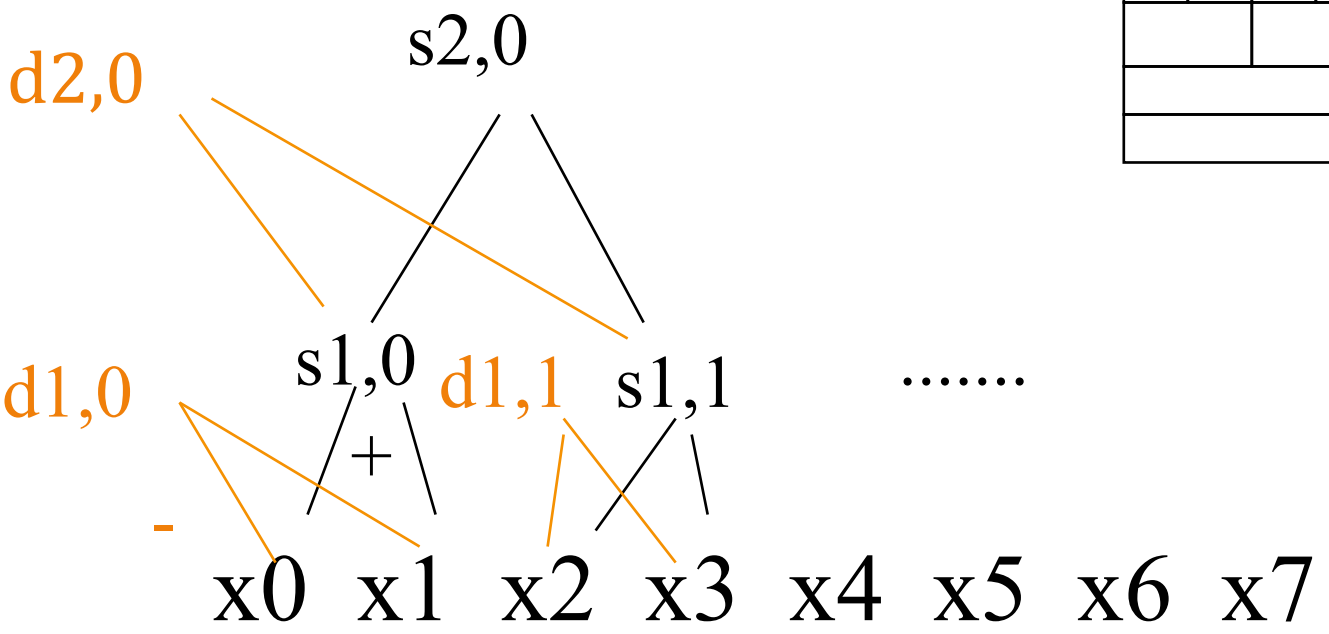


Wavelets - construc DETAILS

Q: map each coefficient

on the time-freq. plane

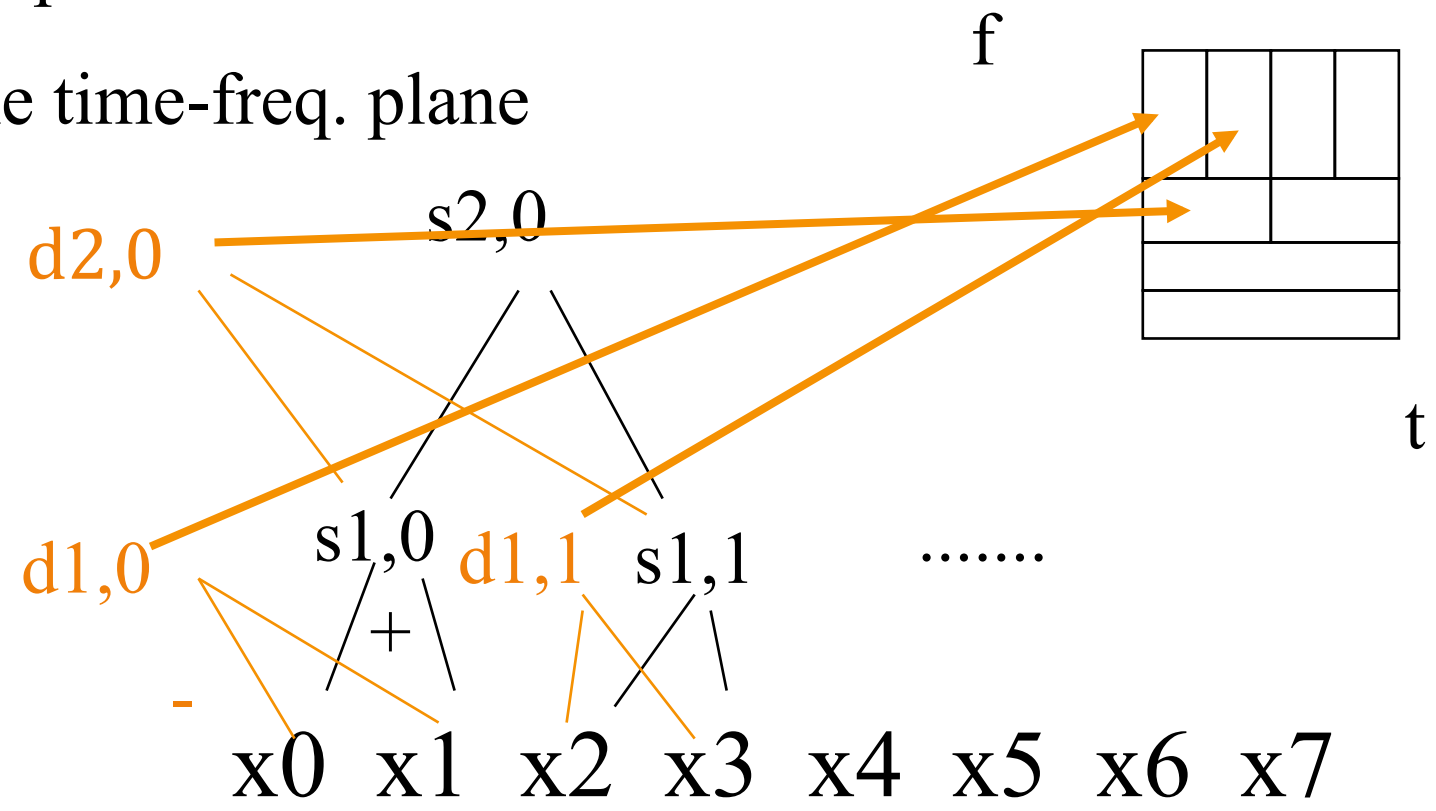




Wavelets - construc DETAILS

Q: map each coefficient

on the time-freq. plane



Haar wavelets - code

```
#!/usr/bin/perl5
# expects a file with numbers
# and prints the dwt transform
# The number of time-ticks should be a power of 2
# USAGE
#    haar.pl <fname>

my @vals=();
my @smooth; # the smooth component of the signal
my @diff;   # the high-freq. component

# collect the values into the array @val
while(<>){
        @vals = ( @vals ,  split );
}
```

```
my $len = scalar(@vals);
my $half = int($len/2);
while($half >= 1 ){
  for(my $i=0; $i< $half; $i++){
        $diff [$i] = ($vals[2*$i] - $vals[2*$i + 1] )/ sqrt(2);
        print "\t", $diff[$i];
        $smooth [$i] = ($vals[2*$i] + $vals[2*$i + 1] )/ sqrt(2);
  }
  print "\n";
  @vals = @smooth;
  $half = int($half/2);
}
print "\t", $vals[0], "\n" ;      # the final, smooth component
```

Wavelets - construc DETAILS

Observation1:

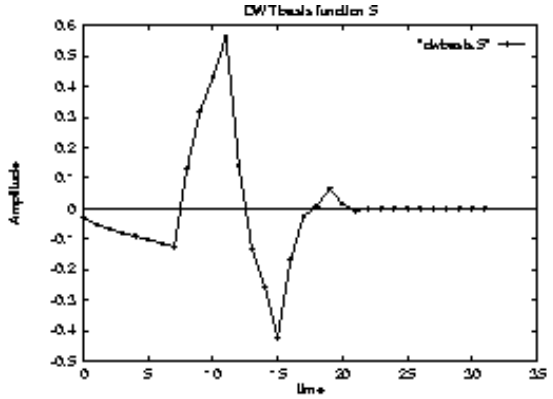
‘+’ can be some weighted addition

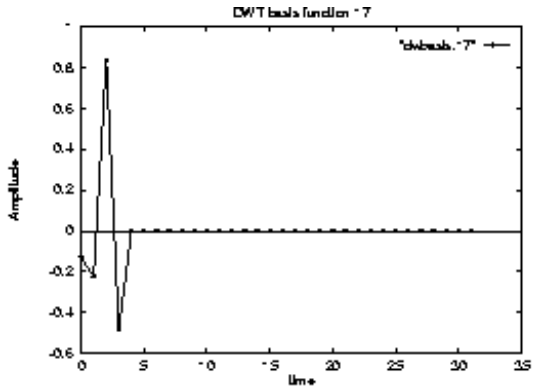
‘-’ is the corresponding weighted difference (‘Quadrature mirror filters’)

Observation2: unlike DFT/DCT,

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

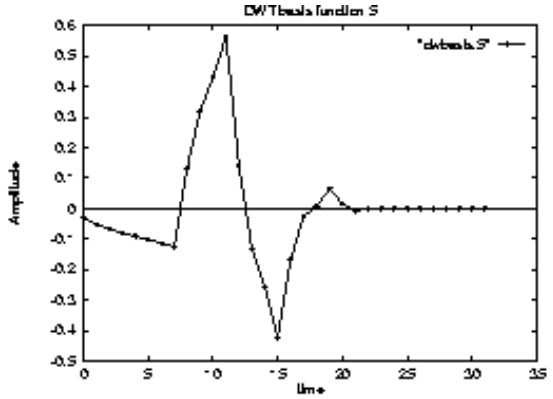
Wavelets - how do they look like?

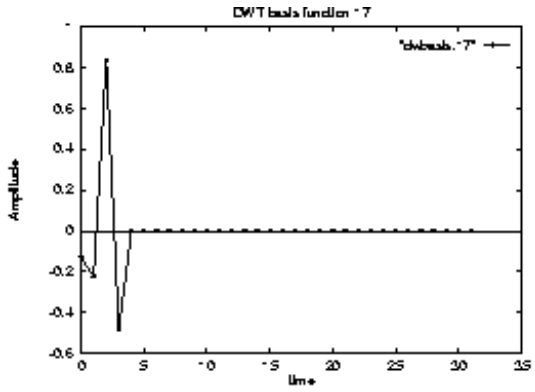




- E.g., Daubechies-4

Wavelets - how do they look like?

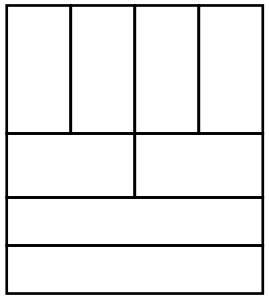




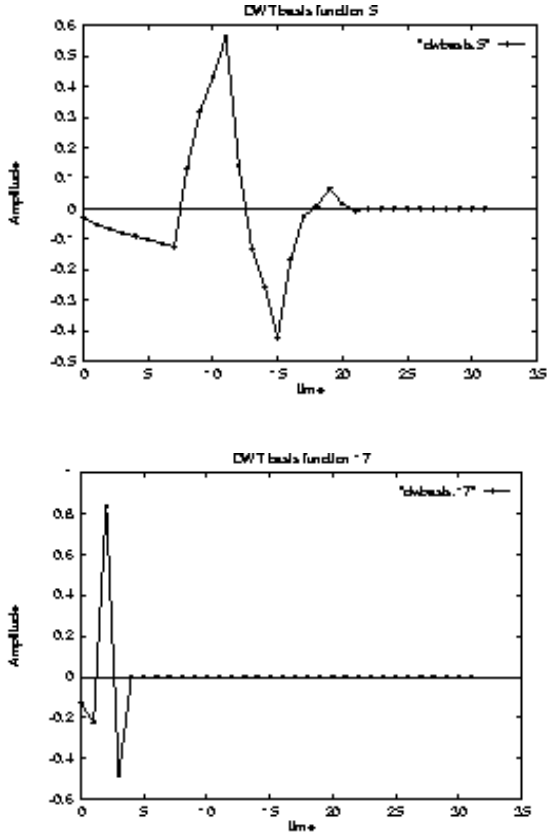
- E.g., Daubechies-4

?

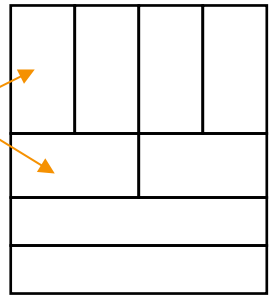
?



Wavelets - how do they look like?



- E.g., Daubechies-4



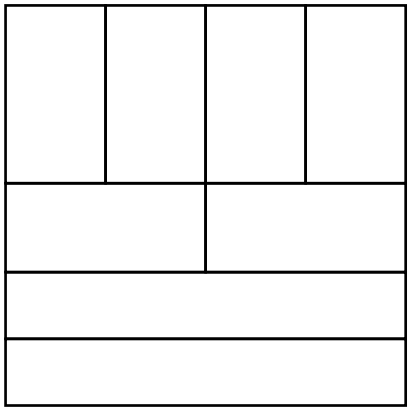
Roadmap

- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
  - Definition of DWT and properties
  - how to read the DWT scalogram

Wavelets - Drill#1:

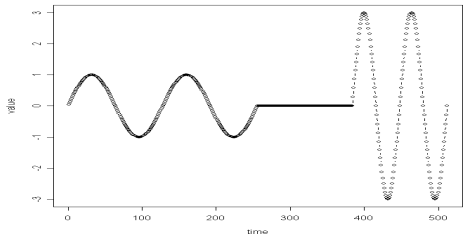
- Q: baritone/silence/soprano - DWT?

f



t

value

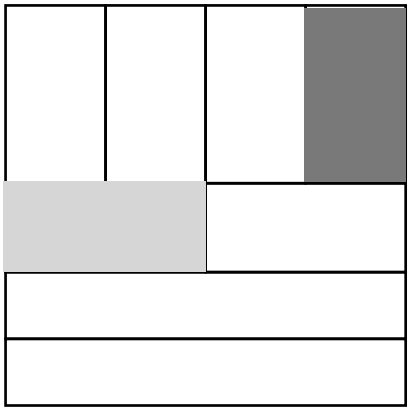


time

Wavelets - Drill#1:

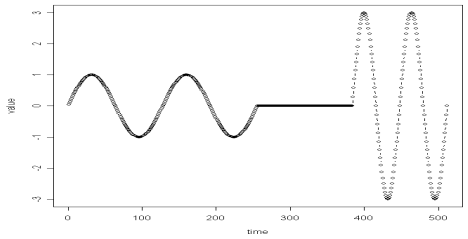
- Q: baritone/silence/soprano - DWT?

f



t

value

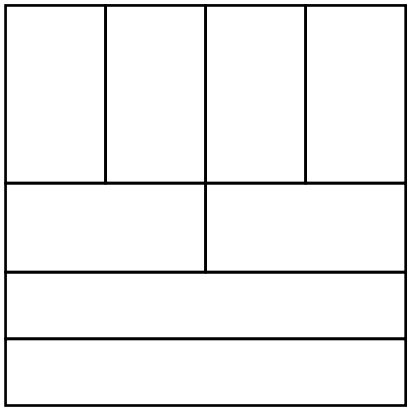


time

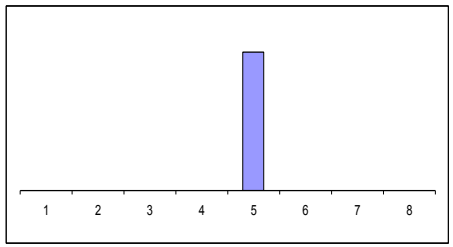
Wavelets - Drill#2:

- Q: spike - DWT?

f



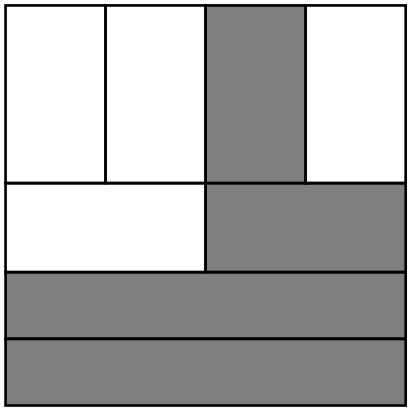
t



Wavelets - Drill#2:

- Q: spike - DWT?

f



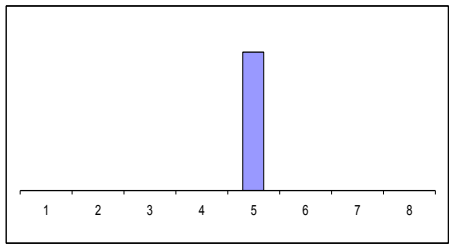
t

0.00 0.00 **0.71** 0.00

0.00 **0.50**

-0.35

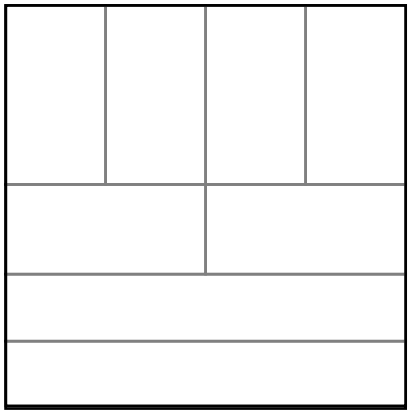
0.35



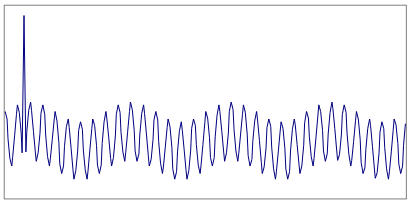
Wavelets - Drill#3:

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

f

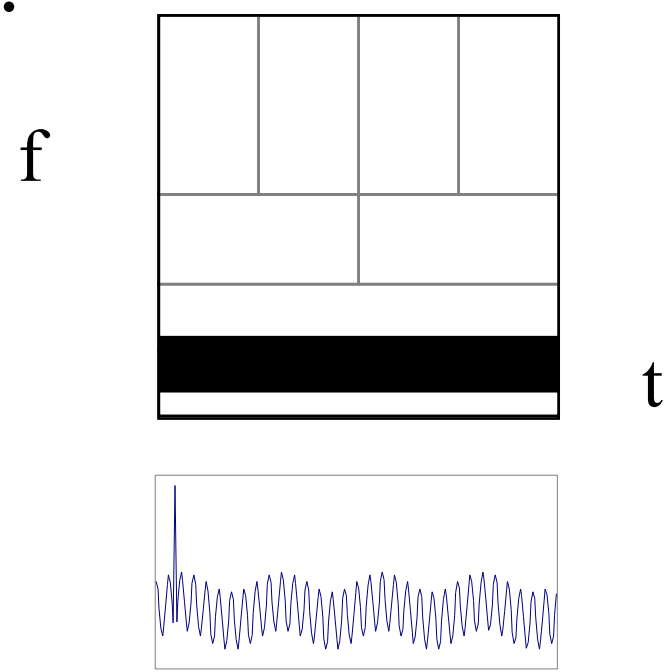


t



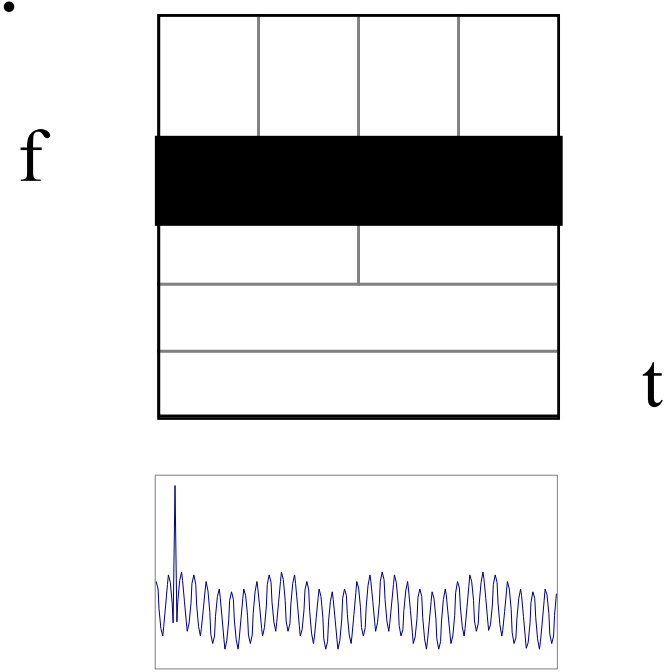
Wavelets - Drill#3:

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



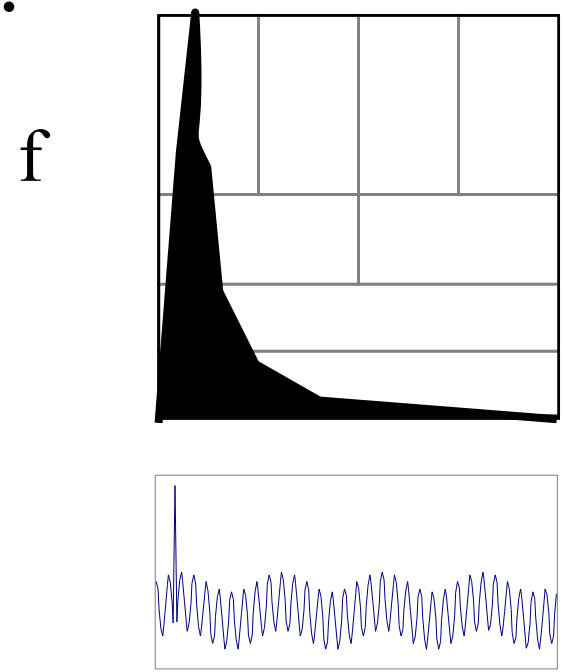
Wavelets - Drill#3:

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



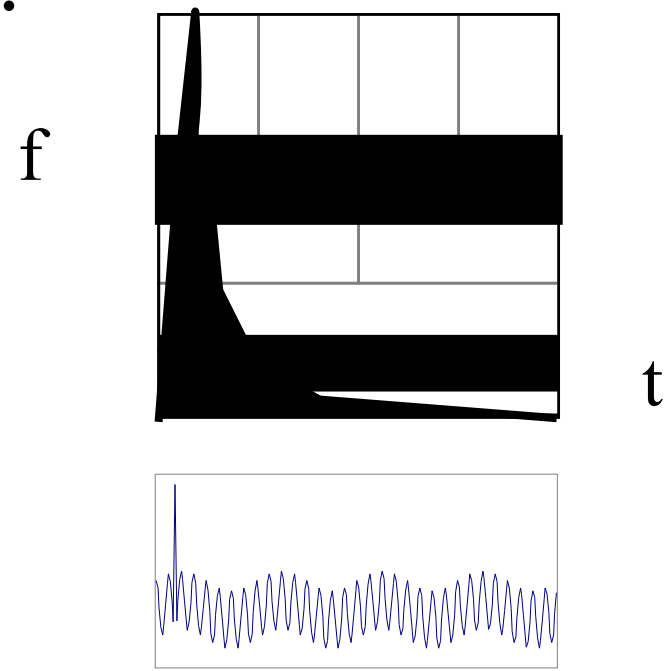
Wavelets - Drill#3:

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



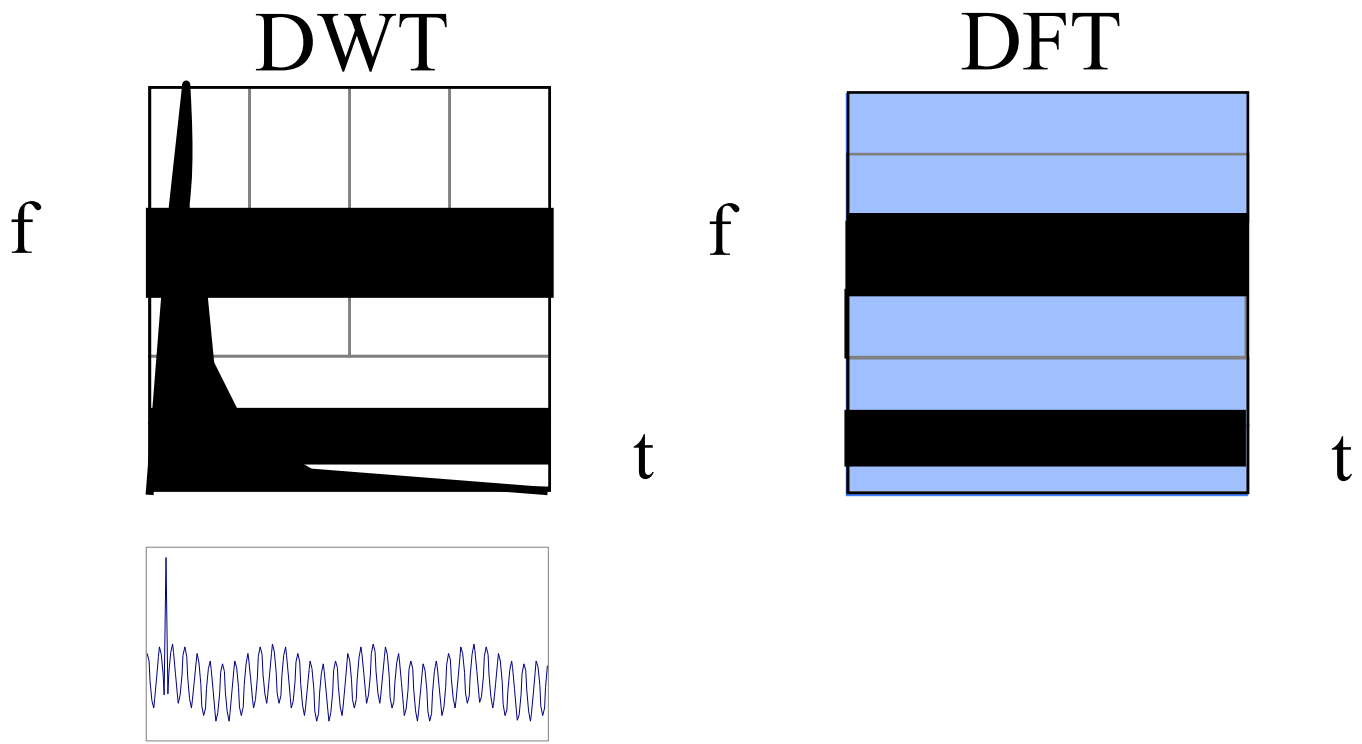
Wavelets - Drill#3:

- Q: 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’
- mammalian eye and ear: Gabor wavelets

Conclusions

- DFT, DCT spot periodicities
- **DWT** : multi-resolution - matches processing of mammalian ear/eye better
- All three: powerful tools for **compression**, **pattern detection** in real signals
- All three: included in math packages
  - (matlab, 'R', mathematica, … - often in spreadsheets!)

Conclusions

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

Part 1 - Roadmap

- Motivation
- Sim. Search and Indexing — Euclidean/DTW + Feature extraction + R-trees
- Feature extraction — DFT, DWT (SVD, ICA)
- Linear forecasting — AR, RLS

- Streaming pattern discovery
- Automatic mining

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