CMU SCS

Indexing and Mining Time Sequences

Christos Faloutsos and Lei Li
CMU

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP (Digital Signal Processing)
- Linear Forecasting
- Kalman filters
- fractals and multifractals
- Non-linear forecasting
- Conclusions

CMU SCS

Problem definition

- Given: one or more sequences
  $x_1, x_2, \ldots, x_t, \ldots$
  $(y_1, y_2, \ldots, y_t, \ldots$
  $\ldots)$
- Find
  - similar sequences; forecasts
  - patterns; clusters; outliers

CMU SCS

Motivation - Applications

- Financial, sales, economic series
- Medical
  - reactions to new drugs
  - elderly care

CMU SCS

ECG - physionet.org

CMU SCS

EEG - epilepsy

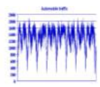
from wikipedia

Motivation - Applications (cont'd)

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

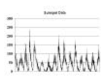
Motivation - Applications (cont'd)

- civil/automobile infrastructure
  - bridge vibrations [Oppenheim+02]
  - road conditions / traffic monitoring



Motivation - Applications (cont'd)

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

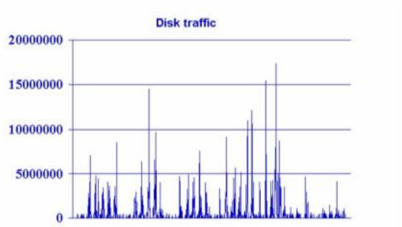


Motivation - Applications (cont'd)

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

Stream Data: Disk accesses

#bytes

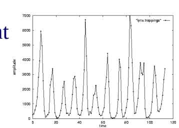


time

Problem #1:

Goal: given a signal (eg., #packets over time)
Find: patterns, periodicities, and/or compress

count



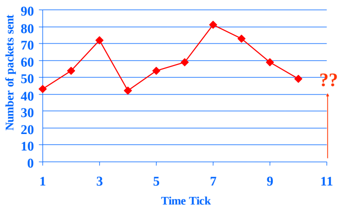
lynx caught per year
(packets per day;
temperature per day)

year

CMU SCS

Problem#2: Forecast

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

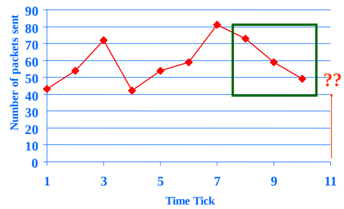


KDD 2010 (c) 2010, C. Faloutsos, Lei Li 13

CMU SCS

Problem#2': Similarity search

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

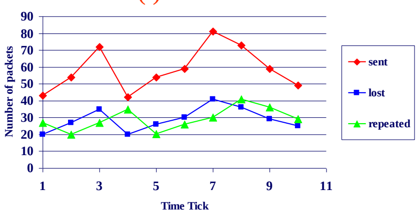


KDD 2010 (c) 2010, C. Faloutsos, Lei Li 14

CMU SCS

Problem #3:

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



KDD 2010 (c) 2010, C. Faloutsos, Lei Li 15

CMU SCS

Important observations

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

- To do forecasting, we need
  - to find patterns/rules
  - to find similar settings in the past
- to find outliers, we need to have forecasts
  - (outlier = too far away from our forecast)

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 16

CMU SCS

Important topics NOT in this tutorial:

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

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 17

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 18

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions: Euclidean;Time-warping
  - indexing
  - feature extraction
- DSP
- ...

CMU SCS

Importance of distance functions

Subtle, but **absolutely necessary**:

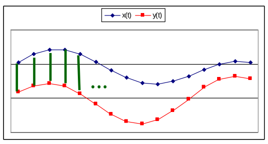
- A 'must' for similarity indexing (-> forecasting)
- A 'must' for clustering

Two major families

- Euclidean and Lp norms
- Time warping and variations

CMU SCS

Euclidean and Lp



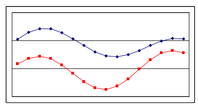
$$D(\vec{x},\vec{y}) = \sum_{i=1}^{n} (x_i - y_i)^2$$

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

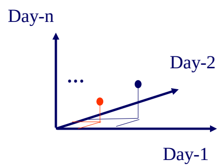
•L_1: city-block = Manhattan
•L_2 = Euclidean
•L_∞

CMU SCS

Observation #1



- **Time sequence -> n-d vector**

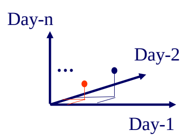


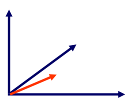
CMU SCS

Observation #2

Euclidean distance is closely related to

- cosine similarity
- dot product
- 'cross-correlation' function





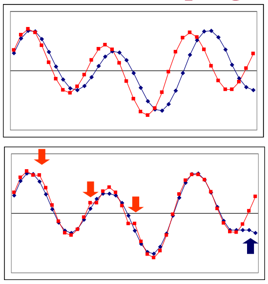
CMU SCS

Time Warping

- allow accelerations - decelerations
  - (with or w/o penalty)
- THEN compute the (Euclidean) distance (+ penalty)
- related to the string-editing distance

Time Warping

‘stutters’:



Skip

Time Warping

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*

Skip

Time Warping

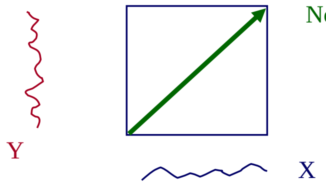
Thus, with no penalty for stutter, for sequences

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

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

Time Warping

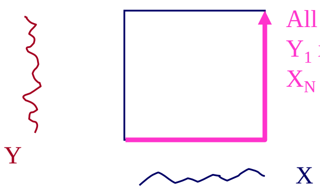
- Time warping matrix & optimal path:



No stutters

Time Warping

- Time warping matrix & optimal path:



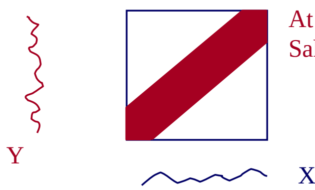
All stutters

Y_1 x N times;

X_N x M times

Time Warping - variations

- Time warping matrix & optimal path:

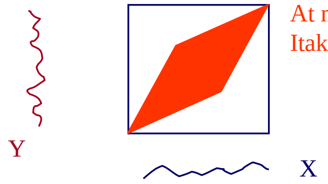


At most k stutters: Sakoe-Chiba band

CMU SCS

Time Warping - variations

- Time warping matrix & optimal path:



At most x% stutters:
Itakura parallelogram

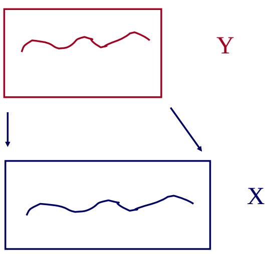
CMU SCS

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]
- Seems suitable for mo-cap

CMU SCS

A variation: Uniform axis scaling



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

CMU SCS

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]

CMU SCS

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

CMU SCS

Conclusions

Prevailing distances:
- Euclidean and
- time-warping

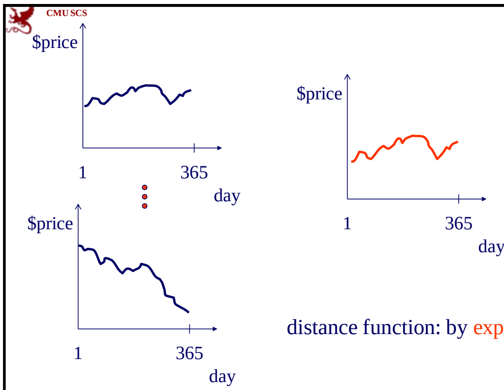
Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
- DSP
- ...

Indexing

Problem:

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



distance function: by expert

Idea: ‘GEMINI’

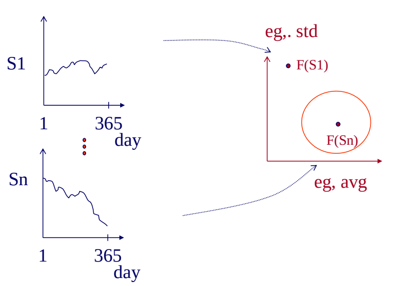
Eg., *‘find stocks similar to MSFT’*

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’)
- discard false alarms

CMU SCS

Examples of GEMINI

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

CMU SCS

Examples of GEMINI

Even on other-than-sequence data:

- Images (QBIC) [JIIS94]
- tumor-like shapes [VLDB96]
- video [Informedia + S-R-trees]
- automobile part shapes [Kriegel+97]

CMU SCS

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)

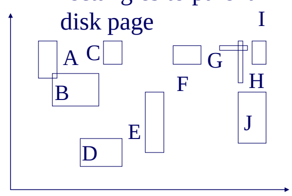
For example:

CMU SCS

Skip

R-trees

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

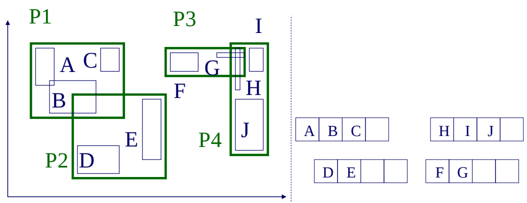


CMU SCS

Skip

R-trees

- eg., w/ fanout 4:

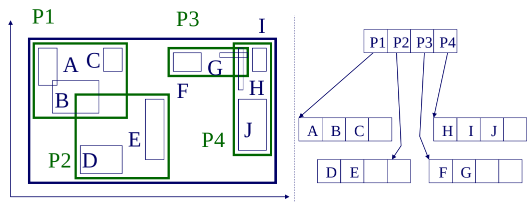


CMU SCS

Skip

R-trees

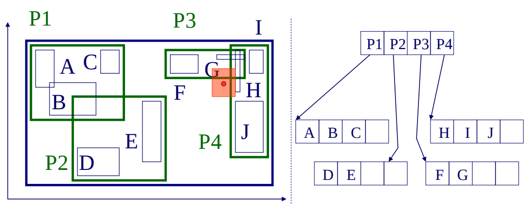
- eg., w/ fanout 4:



CMU SCS

Skip

R-trees - range search?

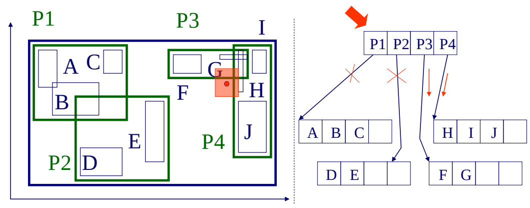


KDD 2010 (c) 2010, C. Faloutsos, Lei Li 49

CMU SCS

Skip

R-trees - range search?



KDD 2010 (c) 2010, C. Faloutsos, Lei Li 50

CMU SCS

Conclusions

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

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 51

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
- DSP
- ...

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 52

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD, etc (data dependent)
    - MDS, FastMap

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 53

CMU SCS

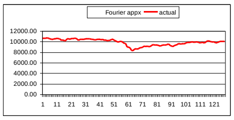
DFT and cousins

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

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 54

CMU SCS

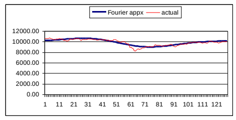
DFT and stocks

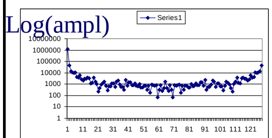


- Dow Jones Industrial index, 6/18/2001-12/21/2001

CMU SCS

DFT and stocks





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

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD etc (data dependent)
    - MDS, FastMap

CMU SCS

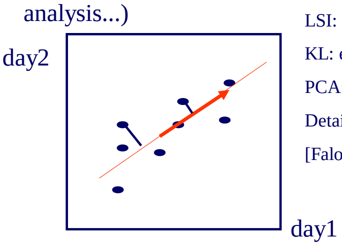
SVD

- <u>THE</u> optimal method for dimensionality reduction
  - (under the Euclidean metric)

CMU SCS

Singular Value Decomposition (SVD)

- SVD (~LSI ~ KL ~ PCA ~ spectral analysis...)



LSI: S. Dumais; M. Berry

KL: eg, Duda+Hart

PCA: eg., Jolliffe

Details: [Press+], [Faloutsos96]

CMU SCS

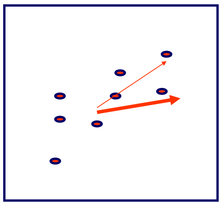
SVD

- **<u>Extremely</u>** 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?

CMU SCS

SVD shorcuts

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



CMU SCS

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])

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD etc (data dependent), **ICA**
    - MDS, FastMap

CMU SCS

Citation

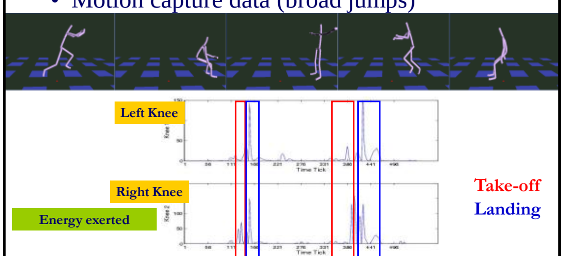
- *AutoSplit: Fast and Scalable Discovery of Hidden Variables in Stream and Multimedia Databases*, **Jia-Yu Pan**, Hiroyuki Kitagawa, Christos Faloutsos and Masafumi Hamamoto

PAKDD 2004, Sydney, Australia

CMU SCS

Motivation: (Q1) Find patterns in data

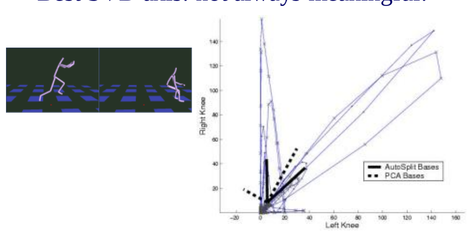
- Motion capture data (broad jumps)



CMU SCS

PCA sometimes misses essential features

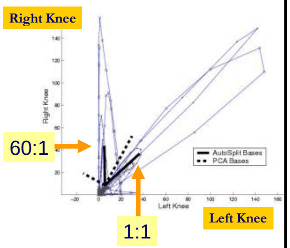
- Best SVD axis: not always meaningful!



CMU SCS

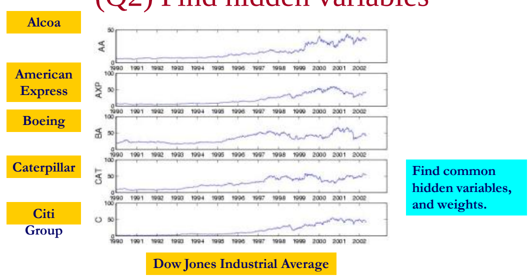
Motivation: (Q1) Find patterns in data

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



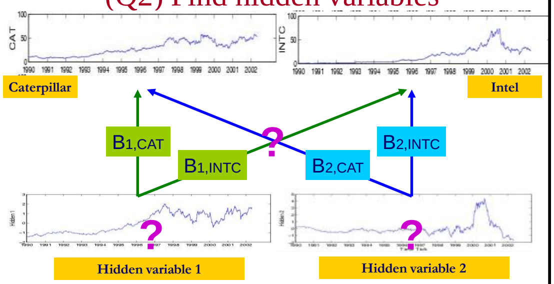
CMU SCS

Motivation: (Q2) Find hidden variables



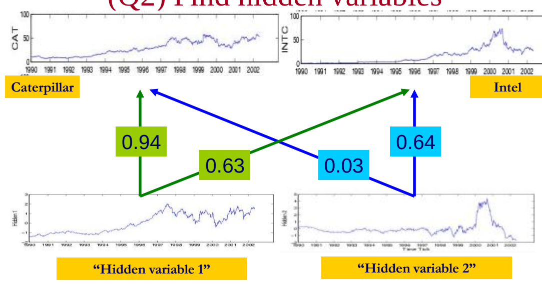
CMU SCS

Motivation: (Q2) Find hidden variables



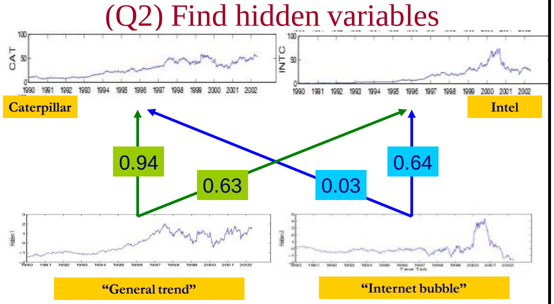
CMU SCS

Motivation: (Q2) Find hidden variables



CMU SCS

Motivation: (Q2) Find hidden variables



CMU SCS

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

Problem formulation

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

X11, X12, …, X1m | … | Xn1, Xn2, …, Xnm = H11, H12, …, H1m | … ? | Hn1, Hn2, …, Hnm B11, B12, …, B1m | … ? | Bm1, Bm2, …, Bmm

Samples of the m-th hidden variable

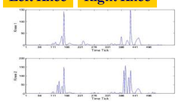
X=HB

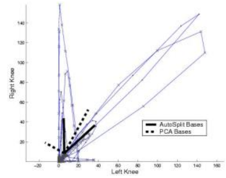
Formulation: (Q1) Find patterns in data

X11, X12 | … | Xn1, Xn2 = H11, H12 | … ? | Hn1, Hn2 B11, B12 ? B21, B22

Basis 1

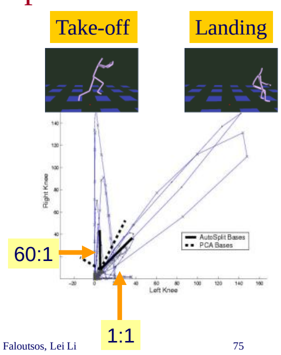
Left Knee Right Knee





Q1: Find patterns

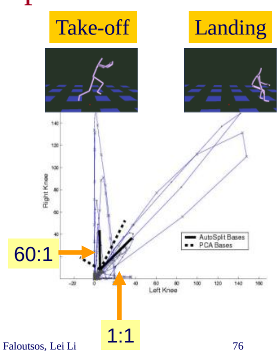
- Patterns found



m=2, n=550

Q1: Find patterns

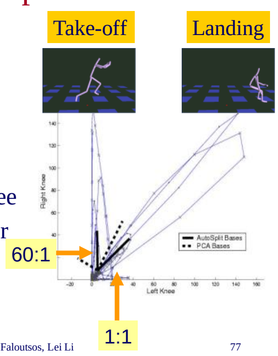
- Patterns found
  - Landing: both knees



m=2, n=550

Q1: Find patterns

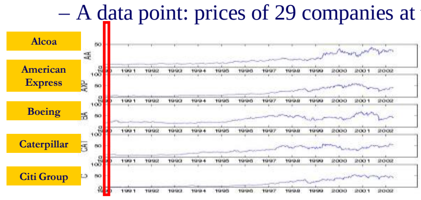
- Patterns found
  - Landing: both knees
  - Take-off: right knee
  - Right-handed actor



m=2, n=550

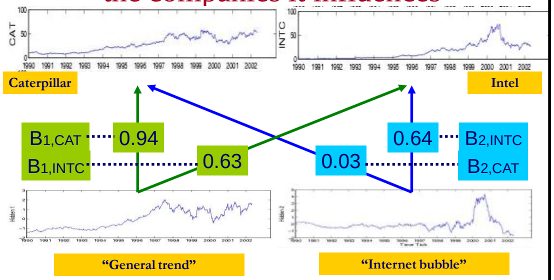
Q2: Find hidden variables (DJIA stocks)

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



CMU SCS

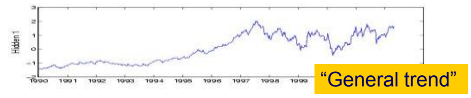
(Q2) Characterize hidden variable by the companies it influences



CMU SCS

Companies related to hidden variable 1

B1,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



CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD (data dependent)
    - MDS, FastMap

CMU SCS

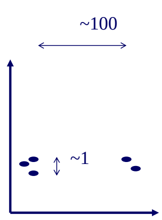
MDS / FastMap

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

CMU SCS

MDS/FastMap

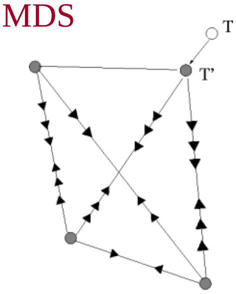
	O1	O2	O3	O4	O5
O1	0	1	1	100	100
O2	1	0	1	100	100
O3	1	1	0	100	100
O4	100	100	100	0	1
O5	100	100	100	1	0



CMU SCS

MDS

Multi Dimensional Scaling



CMU SCS

FastMap

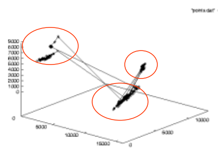
- Multi-dimensional scaling (MDS) can do that, but in O(N**2) time
- FastMap [Faloutsos+95] takes O(N) time

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 85

CMU SCS

FastMap: Application

VideoTrails [Kobla+97]



scene-cut detection (about 10% errors)

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 86

CMU SCS

Outline

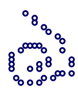
- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD (data dependent)
    - MDS, FastMap, **IsoMap** etc

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 87

CMU SCS

Variations

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

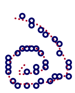


KDD 2010 (c) 2010, C. Faloutsos, Lei Li #88

CMU SCS

Variations

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



KDD 2010 (c) 2010, C. Faloutsos, Lei Li #89

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD (data dependent)
    - MDS, FastMap

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 90

CMU SCS

Conclusions - Practitioner's guide

Similarity search in time sequences

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

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

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

CMU SCS

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)

CMU SCS

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/

CMU SCS

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.

CMU SCS

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.

CMU SCS

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

CMU SCS

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.

CMU SCS

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.

CMU SCS

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.

CMU SCS

References

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

CMU SCS

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.

CMU SCS

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.

CMU SCS

References

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

CMU SCS

Part 2: DSP (Digital Signal Processing)

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP (DFT, DWT)
- Linear Forecasting
- Kalman filters
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

CMU SCS

Outline

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

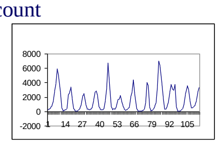
CMU SCS

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

CMU SCS

What does DFT do?

A: highlights the periodicities

CMU SCS

Skip

DFT: definition

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

$$X_f = 1/\sqrt{n} \sum_{t=0}^{n-1} x_t * \exp(-j 2\pi \, tf / n) \qquad f = 0, \dots, n-1$$

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

inverse DFT

$$x_t = 1/\sqrt{n} \sum_{t=0}^{n-1} X_f * \exp(+j 2\pi \, tf / n)$$

CMU SCS

DFT: definition

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

  x = [1,2,1,2];

  X = Fourier[x];

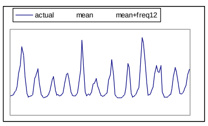
  Plot[Abs[X]];

CMU SCS

DFT: Amplitude spectrum

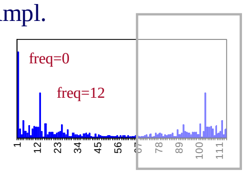
Amplitude: $A_f^2 = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



year

Ampl.



Freq.

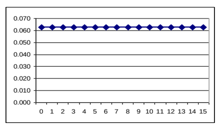
CMU SCS

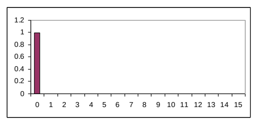
Skip

DFT: examples

flat

Amplitude





time

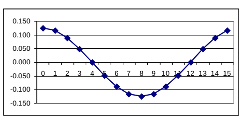
freq

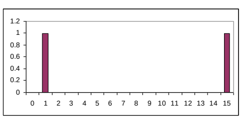
CMU SCS

Skip

DFT: examples

Low frequency sinusoid





time

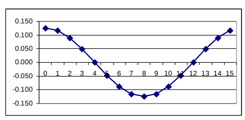
freq

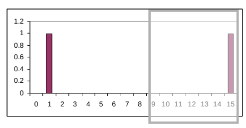
CMU SCS

Skip

DFT: examples

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





time

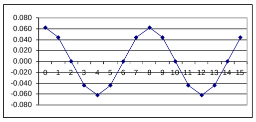
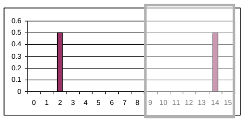
freq

CMU SCS

Skip

DFT: examples

- Higher freq. sinusoid

time

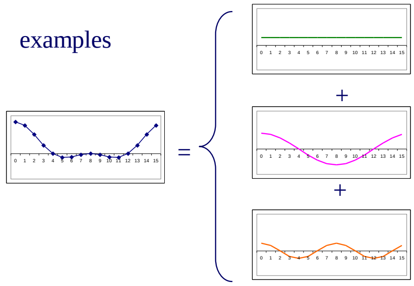
freq

CMU SCS

Skip

DFT: examples

examples



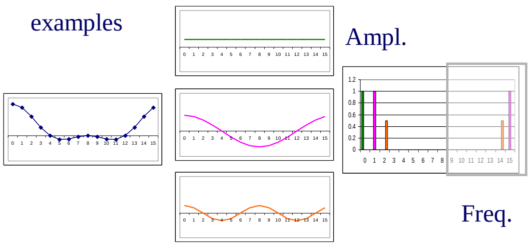
CMU SCS

Skip

DFT: examples

examples

Ampl.



Freq.

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
  - DFT
    - Definition of DFT and properties
    - how to read the DFT spectrum
  - DWT

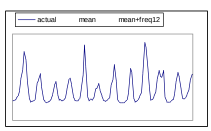
CMU SCS

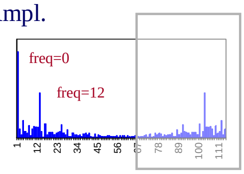
DFT: Amplitude spectrum

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

count

Ampl.





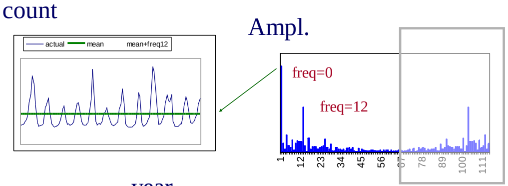
year

Freq.

DFT: Amplitude spectrum

count

Ampl.



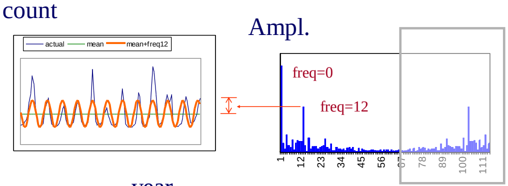
year

Freq.

DFT: Amplitude spectrum

count

Ampl.

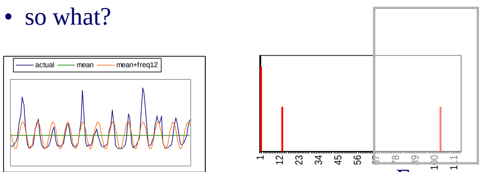


year

Freq.

DFT: Amplitude spectrum

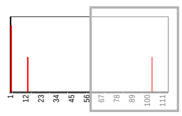
- excellent approximation, with only 2 frequencies!
- so what?



Freq.

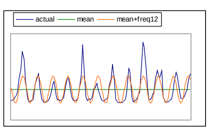
DFT: Amplitude spectrum

- 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)

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
  - DFT
  - DWT
    - Definition of DWT and properties
    - how to read the DWT scalogram

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 127

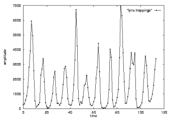
CMU SCS

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)

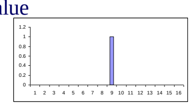
KDD 2010 (c) 2010, C. Faloutsos, Lei Li 128

CMU SCS

Wavelets - DWT

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

value



time

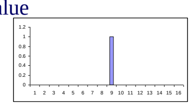
KDD 2010 (c) 2010, C. Faloutsos, Lei Li 129

CMU SCS

Wavelets - DWT

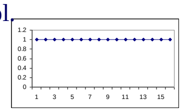
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



time

Ampl.



Freq.

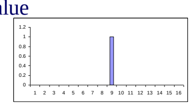
KDD 2010 (c) 2010, C. Faloutsos, Lei Li 130

CMU SCS

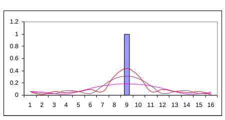
Wavelets - DWT

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

value



time



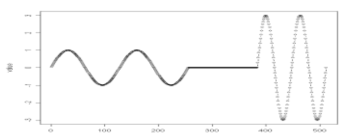
KDD 2010 (c) 2010, C. Faloutsos, Lei Li 131

CMU SCS

Wavelets - DWT

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

value



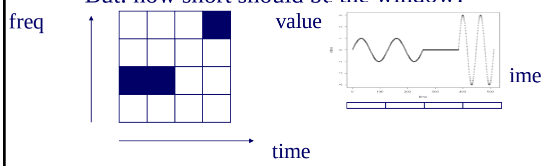
time

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 132

CMU SCS

Wavelets - DWT

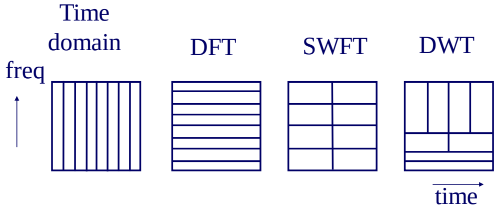
- Solution#1: Short window Fourier transform (SWFT)
- But: how short should be the window?



CMU SCS

Wavelets - DWT

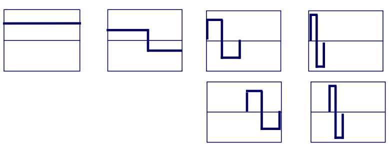
- Answer: **multiple** window sizes! -> DWT



CMU SCS

Haar Wavelets

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



CMU SCS

Skip

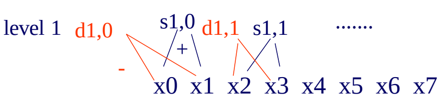
Wavelets - construction

x0 x1 x2 x3 x4 x5 x6 x7

CMU SCS

Skip

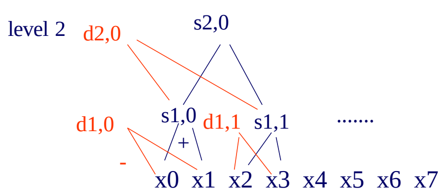
Wavelets - construction



CMU SCS

Skip

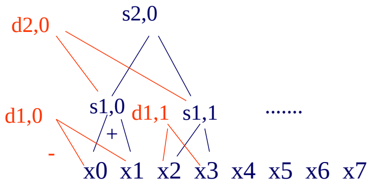
Wavelets - construction



Skip

Wavelets - construction

etc ...

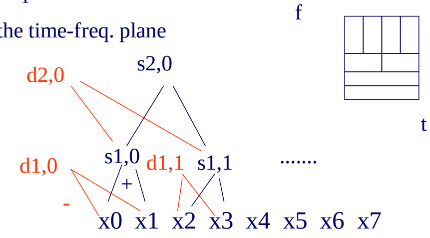


Skip

Wavelets - construction

Q: map each coefficient

on the time-freq. plane

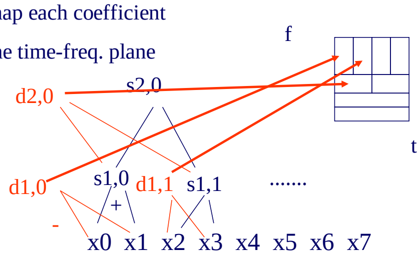


Skip

Wavelets - construction

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 - construction

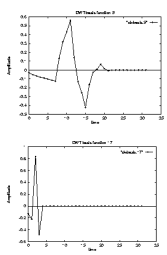
Observation1:

- '+' 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, ...

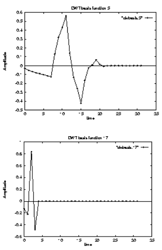
Wavelets - how do they look like?



- E.g., Daubechies-4

CMU SCS

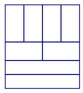
Wavelets - how do they look like?



- E.g., Daubechies-4

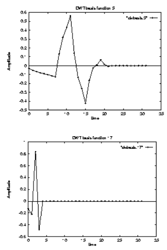
?

?

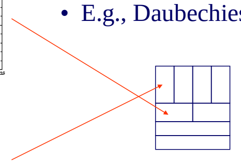


CMU SCS

Wavelets - how do they look like?



- E.g., Daubechies-4



CMU SCS

Outline

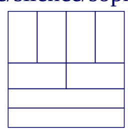
- Motivation
- Similarity Search and Indexing
- DSP
  - DFT
  - DWT
    - Definition of DWT and properties
    - how to read the DWT scalogram

CMU SCS

Wavelets - Drill#1:

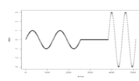
- Q: baritone/silence/soprano - DWT?

f



t

value



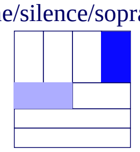
time

CMU SCS

Wavelets - Drill#1:

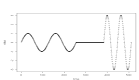
- Q: baritone/silence/soprano - DWT?

f



t

value



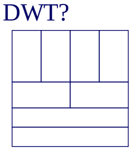
time

CMU SCS

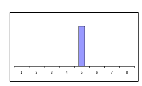
Wavelets - Drill#2:

- Q: spike - DWT?

f

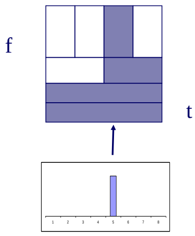


t



Wavelets - Drill#2:

- Q: spike - DWT?



0.00 0.00 **0.71** 0.00

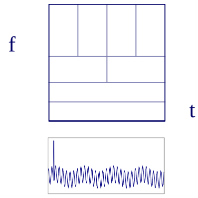
0.00 **0.50**

-0.35

0.35

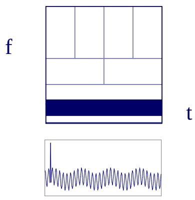
Wavelets - Drill#3:

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



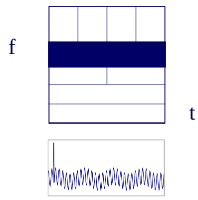
Wavelets - Drill#3:

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



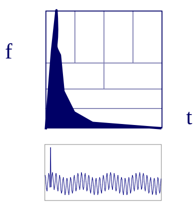
Wavelets - Drill#3:

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



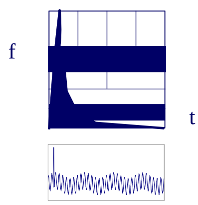
Wavelets - Drill#3:

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



Wavelets - Drill#3:

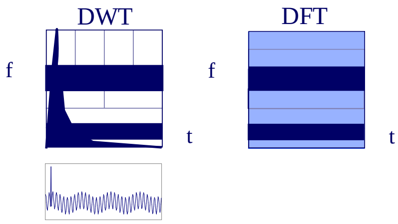
- Q: weekly + daily periodicity, + spike - DWT?



CMU SCS

Wavelets - Drill#3:

- Q: DFT?



CMU SCS

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

CMU SCS

Overall 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!)

CMU SCS

Overall Conclusions

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

CMU SCS

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)

CMU SCS

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

CMU SCS

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)

CMU SCS

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

CMU SCS

Part 3: Linear Forecasting

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
- Linear Forecasting
- Kalman filters
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

CMU SCS

Forecasting

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

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

CMU SCS

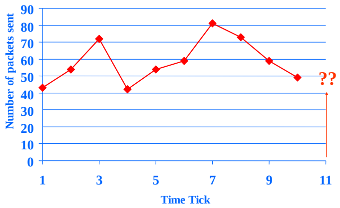
Outline

- Motivation
- ...
- Linear Forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
  - Examples
  - Conclusions

CMU SCS

Problem#2: Forecast

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



CMU SCS

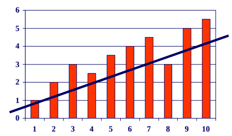
Forecasting: Preprocessing

MANUALLY:

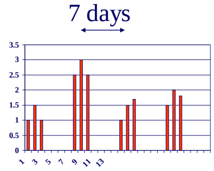
remove trends

spot periodicities

7 days



time



time

CMU SCS

Problem#2: Forecast

- Solution: try to express

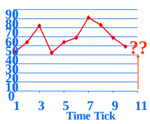
  x_t

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

  (up to a window of w)

Formally:

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



CMU SCS

(Problem: Back-cast; interpolate)

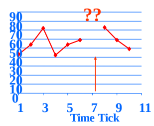
- Solution - interpolate: try to express

  x_t

  as a linear function of the past AND the future:

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

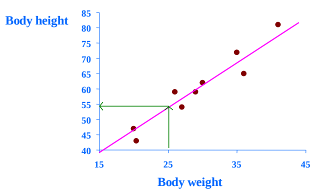
  (up to windows of w_{past} , w_{future})
- EXACTLY the same algo's



CMU SCS

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)')

CMU SCS

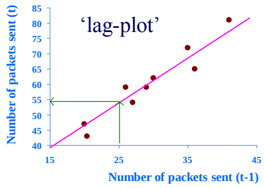
Linear Auto Regression:

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

CMU SCS

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])

CMU SCS

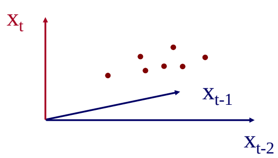
Outline

- Motivation
- …
- Linear Forecasting
  - Auto-regression: **Least Squares; RLS**
  - Co-evolving time sequences
  - Examples
  - Conclusions

CMU SCS

More details:

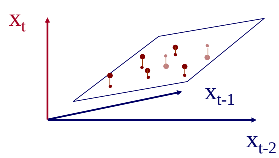
- Q1: Can it work with window w>1?
- A1: YES!



CMU SCS

More details:

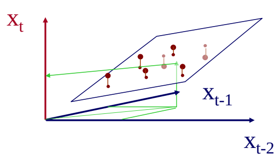
- Q1: Can it work with window w>1?
- A1: YES! (we'll fit a hyper-plane, then!)



CMU SCS

More details:

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



CMU SCS

Skip

More details:

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

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

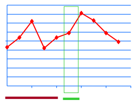
- OVER-CONSTRAINED
  - **a** is the vector of the regression coefficients
  - **X** has the N values of the w indep. variables
  - **y** has the N values of the dependent variable

CMU SCS

Skip

More details:

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



Ind-var1 Ind-var-w

time

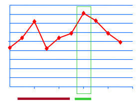
$$\begin{bmatrix} \underline{X_{11}, X_{12}, \cdots, X_{1w}} \\ X_{21}, X_{22}, \ldots, X_{2w} \\ \vdots \\ \vdots \\ \vdots \\ X_{N1}, X_{N2}, \ldots, X_{Nw} \end{bmatrix} \times \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_w \end{bmatrix} = \begin{bmatrix} \underline{y_1} \\ y_2 \\ \vdots \\ \vdots \\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

CMU SCS

Skip

More details:

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



Ind-var1 Ind-var-w

time

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

CMU SCS

Skip

More details

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

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

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

CMU SCS

Skip

Even more details

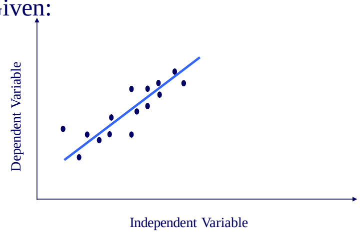
- Q3: Can we estimate **a** incrementally?
- A3: Yes, with the brilliant, classic method of 'Recursive Least Squares' (RLS) (see, e.g., [Yi+00], for details) - pictorially:

CMU SCS

Skip

Even more details

- Given:

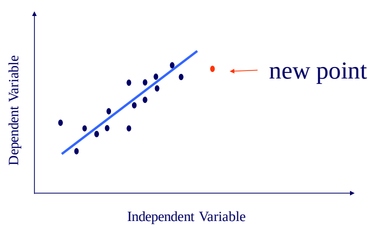


CMU SCS

Skip

Even more details

-

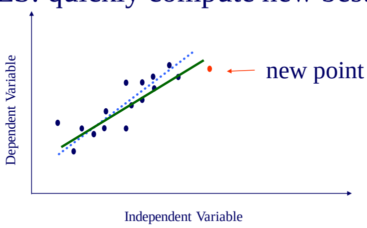


CMU SCS

Skip

Even more details

RLS: quickly compute new best fit



CMU SCS

Skip

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 $O(1 \times w^2)$

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

CMU SCS

Skip

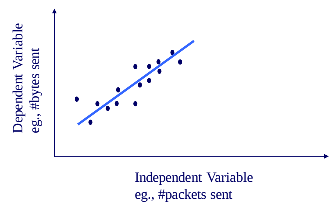
Even more details

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

CMU SCS

Skip

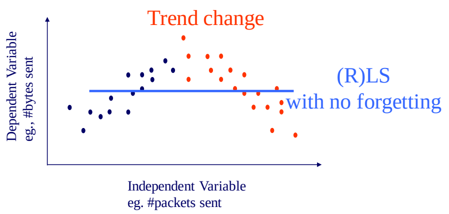
Adaptability - 'forgetting'



CMU SCS

Skip

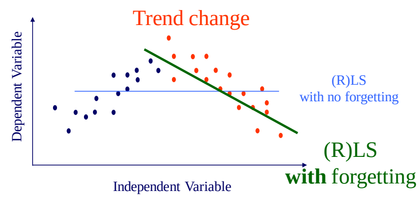
Adaptability - 'forgetting'



CMU SCS

Skip

Adaptability - 'forgetting'



- RLS: can *trivially* handle 'forgetting'

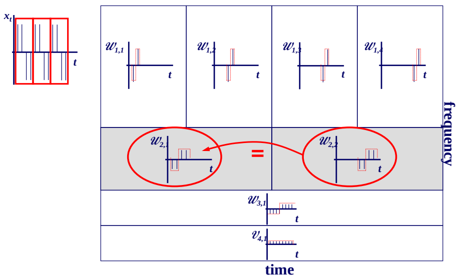
How to choose '*w*'?

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

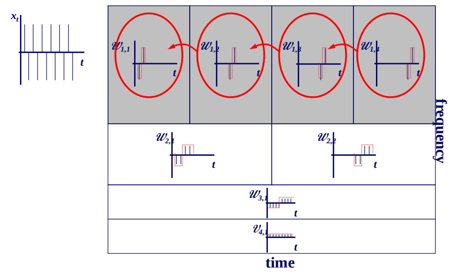
Answer:

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

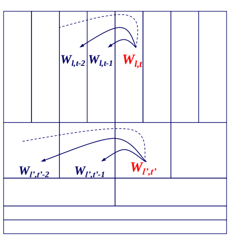
AWSOM



AWSOM



AWSOM - idea



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

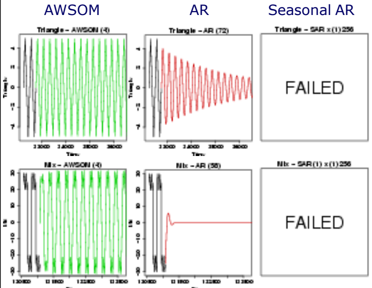
$W_{l',t'} = \beta_{l',1} W_{l',t'-1} + \beta_{l',2} W_{l',t'-2} + \ldots$

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")

CMU SCS

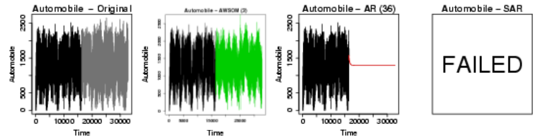
Results - Synthetic data



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

CMU SCS

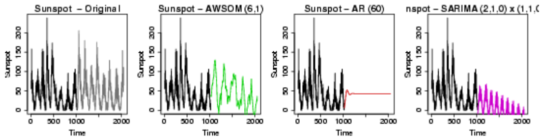
Results - Real data



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

CMU SCS

Results - real data



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

CMU SCS

Skip

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; $O(lgN)$

CMU SCS

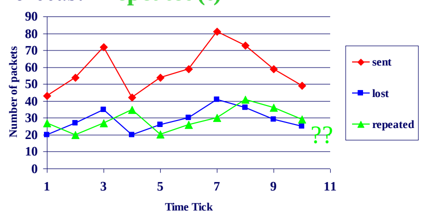
Outline

- Motivation
- ...
- Linear Forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
  - Examples
  - Conclusions

CMU SCS

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])

Time Series Analysis - Outline

- Auto-regression
- Least Squares; recursive least squares
- Co-evolving time sequences
- Examples
- Conclusions

Conclusions - Practitioner's guide

- AR(IMA) methodology: prevailing method for linear forecasting
- Brilliant method of Recursive Least Squares for fast, incremental estimation.
- See [Box-Jenkins]
- very recently: AWSOM (no human intervention)

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.

CMU SCS

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)

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 211

CMU SCS

BREAK!

Next: Kalman filters

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 212

CMU SCS

Part 5: Bursty traffic and multifractals

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 213

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
- Linear Forecasting
- Kalman filters
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 214

CMU SCS

Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
  - Problem
  - Main idea (80/20, Hurst exponent)
  - Results

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 215

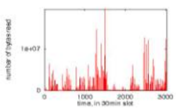
CMU SCS

Recall: Problem #1:

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

Find: patterns, periodicities, and/or compress

#bytes



Bytes per 30'
(packets per day;
earthquakes per year)

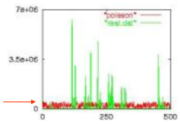
time

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 216

Problem #1

- model bursty traffic
- generate realistic traces
- (Poisson does not work)

bytes



Poisson

time

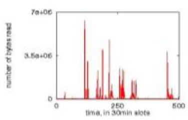
Motivation

- predict queue length distributions (e.g., to give probabilistic guarantees)
- “learn” traffic, for buffering, prefetching, ‘active disks’, web servers

Q: any ‘pattern’?

- Not Poisson
- spike; silence; more spikes; more silence…
- any rules?

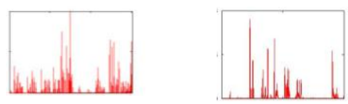
bytes



time

solution: self-similarity

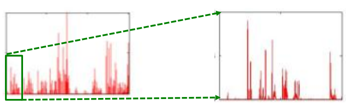
bytes # bytes



time time

solution: self-similarity

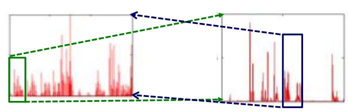
bytes # bytes



time time

solution: self-similarity

bytes # bytes

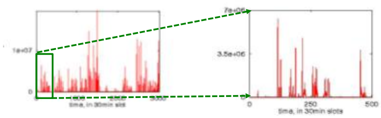


time time

CMU SCS

solution: self-similarity

 # bytes **# bytes**



time **time**

CMU SCS

But:

- Q1: How to generate realistic traces; extrapolate; give guarantees?
- Q2: How to estimate the model parameters?

CMU SCS

Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
  - Problem
  - Main idea (80/20, Hurst exponent)
  - Results

CMU SCS

Approach

- Q1: How to generate a sequence, that is
  - bursty
  - self-similar
  - and has similar queue length distributions

CMU SCS

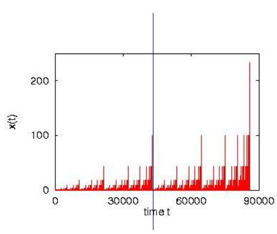
Approach

- A: ‘binomial multifractal’ [Wang+02]
- ~ 80-20 ‘law’:
  - 80% of bytes/queries etc on first half
  - repeat recursively
- *b*: bias factor (eg., 80%)

CMU SCS

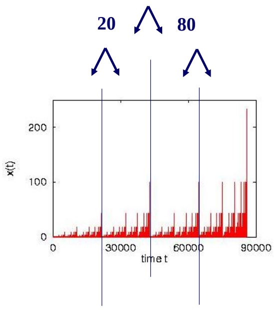
binary multifractals

20 **80**



CMU SCS

binary multifractals



CMU SCS

Parameter estimation

- Q2: How to estimate the bias factor *b*?

 (c) 2010, C. Faloutsos, Lei Li

CMU SCS

Parameter estimation

- Q2: How to estimate the bias factor *b*?
- A: MANY ways [Crovella+96]
  - Hurst exponent
  - variance plot
  - even DFT amplitude spectrum! (‘periodogram’)
  - More robust: ‘entropy plot’ [Wang+02]

 (c) 2010, C. Faloutsos, Lei Li

CMU SCS

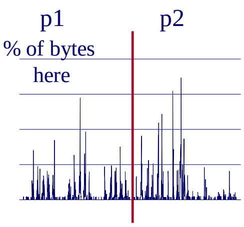
Entropy plot

- Rationale:
  - burstiness: inverse of uniformity
  - entropy measures uniformity of a distribution
  - find entropy at several granularities, to see whether/how our distribution is close to uniform.

 (c) 2010, C. Faloutsos, Lei Li

CMU SCS

Entropy plot

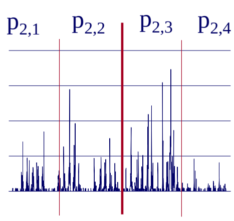


- Entropy *E(n)* after *n* levels of splits
- n=1: E(1)= - p1 $\log_2$(p1)- p2 $\log_2$(p2)

 (c) 2010, C. Faloutsos, Lei Li

CMU SCS

Entropy plot

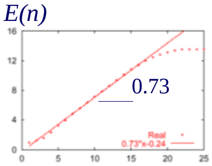


- Entropy *E(n)* after *n* levels of splits
- n=1: E(1)= - p1 log(p1)- p2 log(p2)
- n=2: E(2) = - Σ_ι $p_{2,i}$ * $\log_2$ ($p_{2,i}$)

 (c) 2010, C. Faloutsos, Lei Li

Real traffic

Entropy
E(n)



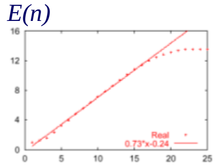
# of levels (*n*)

- Has linear entropy plot (-> self-similar)

Skip

Observation - intuition:

Entropy
E(n)



# of levels (*n*)

intuition: slope =
intrinsic dimensionality =
info-bits per coordinate-bit
- unif. Dataset: slope =1
- multi-point: slope = 0

Skip

Entropy plot - Intuition

- Slope ~ intrinsic dimensionality (in fact, 'Information fractal dimension')
- = info bit per coordinate bit - eg

Dim = 1

Pick a point;
reveal its coordinate bit-by-bit -
how much info is each bit worth to me?

Skip

Entropy plot

- Slope ~ intrinsic dimensionality (in fact, 'Information fractal dimension')
- = info bit per coordinate bit - eg

Dim = 1

Is MSB 0?

'info' value = E(1): 1 bit

Skip

Entropy plot

- Slope ~ intrinsic dimensionality (in fact, 'Information fractal dimension')
- = info bit per coordinate bit - eg

Dim = 1

Is MSB 0?

Is next MSB =0?

Skip

Entropy plot

- Slope ~ intrinsic dimensionality (in fact, 'Information fractal dimension')
- = info bit per coordinate bit - eg

Dim = 1

Is MSB 0?

Info value =1 bit
= E(2) - E(1) =
slope!

Is next MSB =0?

CMU SCS

Skip

Entropy plot

- Repeat, for all points at same position:

Dim=0

CMU SCS

Skip

Entropy plot

- Repeat, for all points at same position:
- we need 0 bits of info, to determine position
- -> slope = 0 = intrinsic dimensionality

Dim=0

CMU SCS

Skip

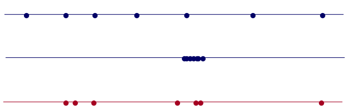
Entropy plot

- Real (and 80-20) datasets can be in-between: bursts, gaps, smaller bursts, smaller gaps, at every scale

Dim = 1

Dim=0

0<Dim<1



CMU SCS

(Fractals)

- What set of points could have behavior between point and line?

CMU SCS

Cantor dust

- Eliminate the middle third
- Recursively!

CMU SCS

Cantor dust

CMU SCS

Cantor dust

CMU SCS

Cantor dust

CMU SCS

Cantor dust

CMU SCS

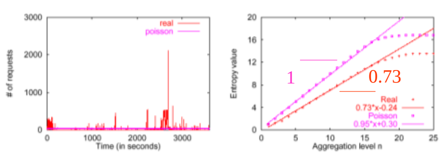
Cantor dust

Dimensionality?
(no length; infinite # points!)
Answer: log2 / log3 = 0.6

CMU SCS

Some more entropy plots:

- Poisson vs real

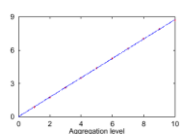


Poisson: slope = ~1 -> uniformly distributed

CMU SCS

B-model

E(n)



n

- b-model traffic gives perfectly linear plot
- Lemma: its slope is
  $slope = -b \log_2 b - (1-b) \log_2 (1-b)$
- Fitting: do entropy plot; get slope; solve for *b*

CMU SCS

Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
  - Problem
  - Main idea (80/20, Hurst exponent)
  - Experiments - Results

CMU SCS

Experimental setup

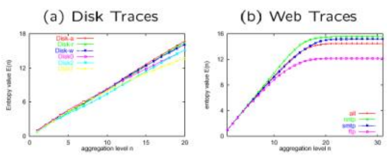
- Disk traces (from HP [Wilkes 93])
- web traces from LBL

  http://repository.cs.vt.edu/lbl-conn-7.tar.Z

CMU SCS

Model validation

- Linear entropy plots



Bias factors b: 0.6-0.8

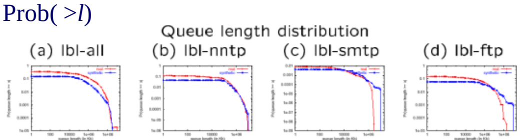
smallest b / smoothest: nntp traffic

CMU SCS

Web traffic - results

- LBL, NCDF of queue lengths (log-log scales)

Prob(>l)



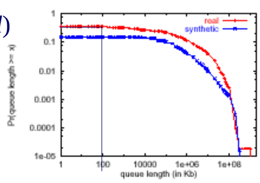
How to give guarantees? (queue length l)

CMU SCS

Web traffic - results

- LBL, NCDF of queue lengths (log-log scales)

Prob(>l)



20% of the requests will see queue lengths <100

(queue length l)

CMU SCS

Conclusions

- Multifractals (80/20, 'b-model', Multiplicative Wavelet Model (MWM)) for analysis and synthesis of bursty traffic
- can give (probabilistic) guarantees

CMU SCS

Books

- Fractals: Manfred Schroeder: *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise* W.H. Freeman and Company, 1991 (Probably the BEST book on fractals!)

CMU SCS

Further reading:

- Crovella, M. and A. Bestavros (1996). Self-Similarity in World Wide Web Traffic, Evidence and Possible Causes. Sigmetrics.
- [ieeeTN94] W. E. Leland, M.S. Taqqu, W. Willinger, D.V. Wilson, *On the Self-Similar Nature of Ethernet Traffic,* IEEE Transactions on Networking, 2, 1, pp 1-15, Feb. 1994.

CMU SCS

Further reading

- [Riedi+99] R. H. Riedi, M. S. Crouse, V. J. Ribeiro, and R. G. Baraniuk, *A Multifractal Wavelet Model with Application to Network Traffic*, IEEE Special Issue on Information Theory, 45. (April 1999), 992-1018.
- [Wang+02] Mengzhi Wang, Tara Madhyastha, Ngai Hang Chang, Spiros Papadimitriou and Christos Faloutsos, *Data Mining Meets Performance Evaluation: Fast Algorithms for Modeling Bursty Traffic*, ICDE 2002, San Jose, CA, 2/26/2002 - 3/1/2002.

CMU SCS

Part 6: chaos and non-linear forecasting

CMU SCS

Outline

- Motivation
- Similarity Search and Indexing
- DSP
- Linear Forecasting
- Kalman filters
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

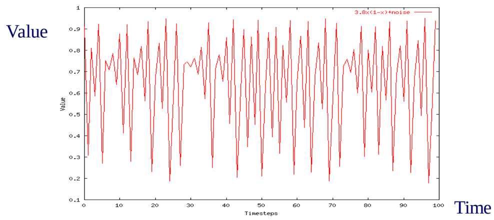
CMU SCS

Detailed Outline

- Non-linear forecasting
  - Problem
  - Idea
  - How-to
  - Experiments
  - Conclusions

CMU SCS

Recall: Problem #1



Given a time series $\{x_t\}$, predict its future course, that is, x_{t+1}, x_{t+2}, ...

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 265

CMU SCS

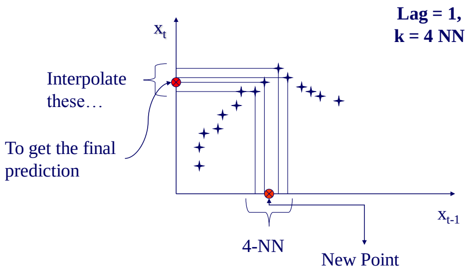
How to forecast?

- ARIMA - but: linearity assumption
- ANSWER: 'Delayed Coordinate Embedding' = Lag Plots [Sauer92]

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 266

CMU SCS

General Intuition (Lag Plot)



KDD 2010 (c) 2010, C. Faloutsos, Lei Li 267

CMU SCS

Questions:

- Q1: How to choose lag *L*?
- Q2: How to choose *k* (the # of NN)?
- Q3: How to interpolate?
- Q4: why should this work at all?

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 268

CMU SCS

Q1: Choosing lag *L*

- Manually (16, in award winning system by [Sauer94])

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 269

CMU SCS

Q2: Choosing number of neighbors *k*

- Manually (typically ~ 1-10)

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 270

Q3: How to interpolate?

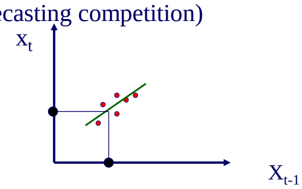
How do we interpolate between the *k* nearest neighbors?

A3.1: Average

A3.2: Weighted average (weights drop with distance - how?)

Q3: How to interpolate?

A3.3: Using SVD - seems to perform best ([Sauer94] - first place in the Santa Fe forecasting competition)



Q4: Any theory behind it?

A4: YES!

Theoretical foundation

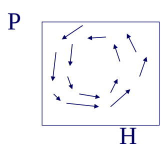
- Based on the "Takens' Theorem" [Takens81]
- which says that <u>long enough</u> delay vectors can do prediction, even if there are unobserved variables in the dynamical system (= diff. equations)

Skip

Theoretical foundation

Example: Lotka-Volterra equations

dH/dt = r H – a H*P
dP/dt = b H*P – m P



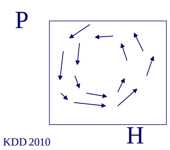
H is count of prey (e.g., hare)
P is count of predators (e.g., lynx)

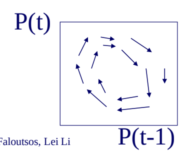
Suppose only P(t) is observed (t=1, 2, …).

Skip

Theoretical foundation

- But the delay vector space is a faithful reconstruction of the internal system state
- So prediction in **delay vector space** is as good as prediction in **state space**

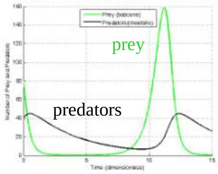


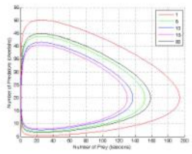


CMU SCS

Solution to Volterra-Lotka eq.

predators





time

prey

from wikipedia

KDD 2010 277

CMU SCS

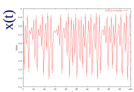
Detailed Outline

- Non-linear forecasting
  - Problem
  - Idea
  - How-to
  - Experiments
  - Conclusions

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 278

CMU SCS

Datasets



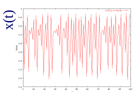
Logistic Parabola:
$x_t = ax_{t-1}(1-x_{t-1})$ + noise
Models population of flies [R. May/1976]

Lag-plot

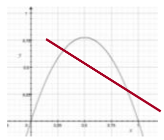
KDD 2010 (c) 2010, C. Faloutsos, Lei Li 279

CMU SCS

Datasets



Logistic Parabola:
$x_t = ax_{t-1}(1-x_{t-1})$ + noise
Models population of flies [R. May/1976]



Lag-plot
ARIMA: fails

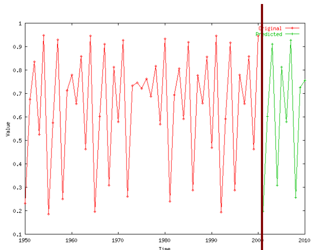
KDD 2010 (c) 2010, C. Faloutsos, Lei Li 280

CMU SCS

Logistic Parabola

Our Prediction from here

Value



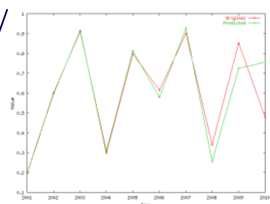
Timesteps

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 281

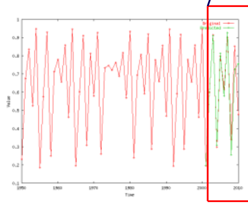
CMU SCS

Logistic Parabola

Value



Comparison of prediction to correct values



Timesteps

KDD 2010 (c) 2010, C. Faloutsos, Lei Li 282

CMU SCS

Skip

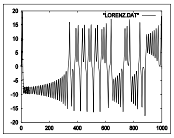
Datasets

LORENZ: Models convection currents in the air

dx / dt = a (y - x)

dy / dt = x (b - z) - y

dz / dt = xy - c z

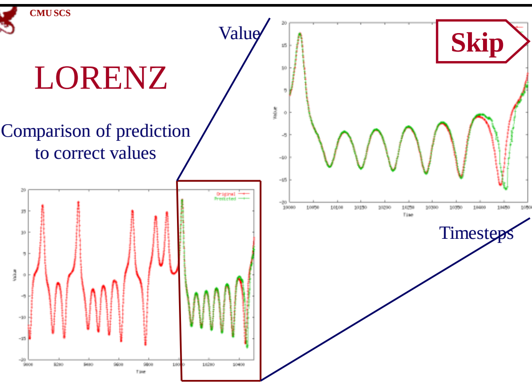


CMU SCS

Skip

LORENZ

Comparison of prediction to correct values

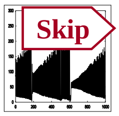


CMU SCS

Skip

Datasets

- LASER: fluctuations in a Laser over time (used in Santa Fe competition)

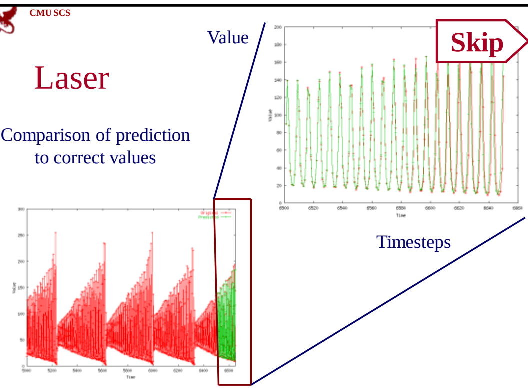


CMU SCS

Skip

Laser

Comparison of prediction to correct values



CMU SCS

Conclusions

- Lag plots for non-linear forecasting (Takens' theorem)
- suitable for 'chaotic' signals

CMU SCS

References

- Deepay Chakrabarti and Christos Faloutsos *F4: Large-Scale Automated Forecasting using Fractals* CIKM 2002, Washington DC, Nov. 2002.
- Sauer, T. (1994). *Time series prediction using delay coordinate embedding*. (in book by Weigend and Gershenfeld, below) Addison-Wesley.
- Takens, F. (1981). *Detecting strange attractors in fluid turbulence*. Dynamical Systems and Turbulence. Berlin: Springer-Verlag.

CMU SCS

References

- Weigend, A. S. and N. A. Gerschenfeld (1994). *Time Series Prediction: Forecasting the Future and Understanding the Past*, Addison Wesley. (Excellent collection of papers on chaotic/non-linear forecasting, describing the algorithms behind the winners of the Santa Fe competition.)

CMU SCS

Overall conclusions

- Similarity search: **Euclidean**/time-warping; **feature extraction** and **SAMs**

CMU SCS

Overall conclusions

- Similarity search: **Euclidean**/time-warping; **feature extraction** and **SAMs**
- Signal processing: **DWT** is a powerful tool

CMU SCS

Overall conclusions

- Similarity search: **Euclidean**/time-warping; **feature extraction** and **SAMs**
- Signal processing: **DWT** is a powerful tool
- Linear Forecasting: **AR** (Box-Jenkins)

CMU SCS

Overall conclusions

- Similarity search: **Euclidean**/time-warping; **feature extraction** and **SAMs**
- Signal processing: **DWT** is a powerful tool
- Linear Forecasting: **AR** (Box-Jenkins)
- **Kalman** filters & extensions: forecasting, pattern discovery, segmentation

CMU SCS

Overall conclusions

- Similarity search: **Euclidean**/time-warping; **feature extraction** and **SAMs**
- Signal processing: **DWT** is a powerful tool
- Linear Forecasting: **AR** (Box-Jenkins)
- **Kalman** filters & extensions: forecasting, pattern discovery, segmentation
- Bursty traffic: **multifractals** (80-20 ‘law’)

Overall conclusions

- Similarity search: **Euclidean**/time-warping; **feature extraction** and **SAMs**
- Signal processing: **DWT** is a powerful tool
- Linear Forecasting: **AR** (Box-Jenkins)
- **Kalman** filters & extensions: forecasting, pattern discovery, segmentation
- Bursty traffic: **multifractals** (80-20 'law')
- Non-linear forecasting: **lag-plots** (Takens)

THANK YOU!

christos AT cs.cmu.edu
www.cs.cmu.edu/~christos

leili AT cs.cmu.edu
www.cs.cmu.edu/~leili
www.cs.cmu.edu/~leili/pubs/dynammo.2.1.2.zip