Sensor Data Mining: Similarity search and pattern analysis

Christos Faloutsos
CMU

Thanks

Deepay Chakrabarti (CMU)

Prof. Dimitris Gunopulos (UCR)

Spiros Papadimitriou (CMU)

Mengzhi Wang (CMU)

Prof. Byoung-Kee Yi (Pohang U.)

Outline

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

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

Motivation - Applications

- Financial, sales, economic series
- Medical
  - ECGs +; blood pressure etc monitoring
  - reactions to new drugs
  - elderly care

Motivation - Applications (cont'd)

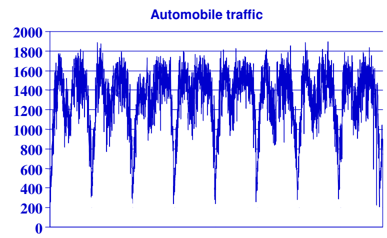
- 'Smart house'
  - sensors monitor temperature, humidity, air quality
- video surveillance

Motivation - Applications (cont'd)

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

Stream Data: automobile traffic

cars



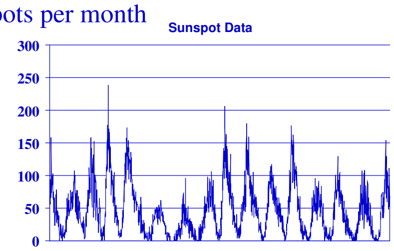
time

Motivation - Applications (cont'd)

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

Stream Data: Sunspots

#sunspots per month



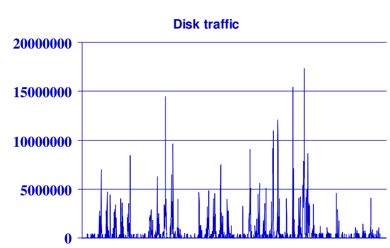
time

Motivation - Applications (cont'd)

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

Stream Data: Disk accesses

#bytes

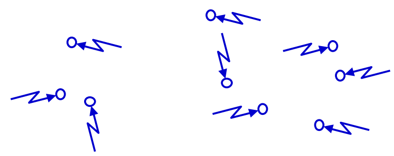


time

Settings & Applications

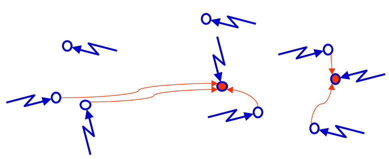
- One or more sensors, collecting time-series data

Settings & Applications



Each sensor collects data (x_1, x_2, …, x_t, …)

Settings & Applications

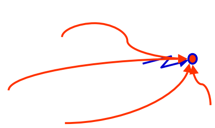


Some sensors ‘report’ to others or to the central site

Settings & Applications

Goal #1:
Finding patterns
in a single time sequence

Settings & Applications

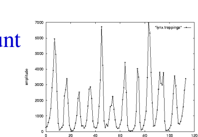


Goal #2:
Finding patterns
in many time
sequences

Problem #1:

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

count



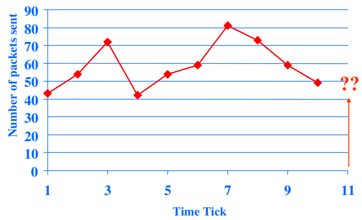
year

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

CMU SCS

Problem#2: Forecast

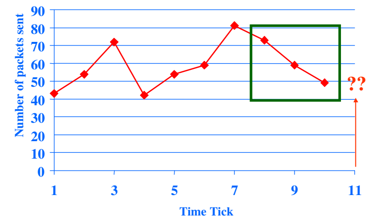
Given x_t, x_{t-1}, …, forecast x_{t+1}



CMU SCS

Problem#2': Similarity search

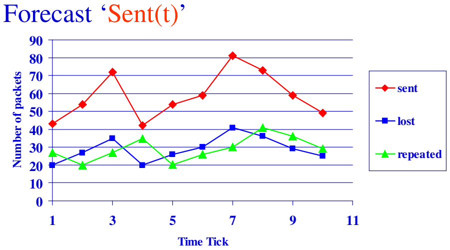
Eg., Find a 3-tick pattern, similar to the last one



CMU SCS

Problem #3:

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



CMU SCS

Differences from DSP/Stat

- Semi-infinite streams
  - we need on-line, 'any-time' algorithms
- Can not afford human intervention
  - need automatic methods
- sensors have limited memory / processing / transmitting power
  - need for (lossy) compression

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)

CMU SCS

Important topics NOT in this tutorial:

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

Outline

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

Outline

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

Importance of distance functions

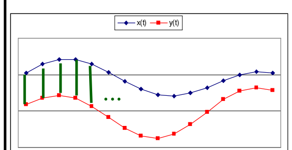
Subtle, but **absolutely necessary**:

- A ‘must’ for similarity indexing (-> forecasting)
- A ‘must’ for clustering

Two major families

- Euclidean and Lp norms
- Time warping and variations

Euclidean and Lp

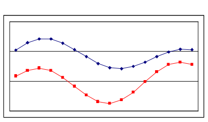


$$D(\vec{x},\vec{y}) = \sum_{i=1}^{n} (x_i - y_i)^2$$

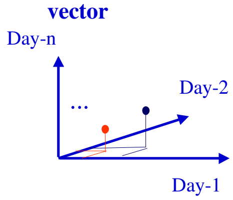
$$L_p(\vec{x},\vec{y}) = \sum_{i=1}^{n} | x_i - y_i |^p$$

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

Observation #1



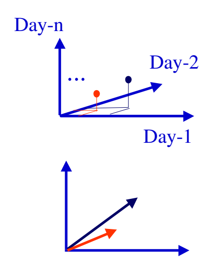
- **Time sequence -> n-d vector**



Observation #2

Euclidean distance is closely related to

- cosine similarity
- dot product
- ‘cross-correlation’ function



CMU SCS

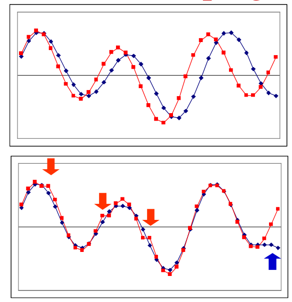
Time Warping

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

VLDB-02 © C. Faloutsos (2002) 31

CMU SCS

Time Warping



'stutters':

VLDB-02 © C. Faloutsos (2002) 32

CMU SCS

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*

VLDB-02 © C. Faloutsos (2002) 33

CMU SCS

Skip

Time warping

Thus, with no penalty for stutter, for sequences

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

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

VLDB-02 © C. Faloutsos (2002) 34

CMU SCS

Skip

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]

VLDB-02 © C. Faloutsos (2002) 35

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]

See tutorial by [Gunopulos Das, SIGMOD01]

VLDB-02 © C. Faloutsos (2002) 36

Conclusions

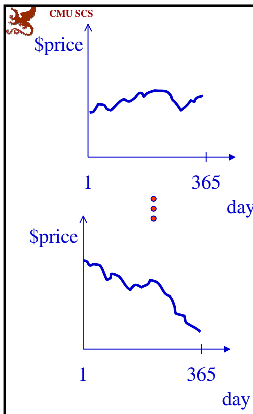
Prevailing distances:
- Euclidean and
- time-warping

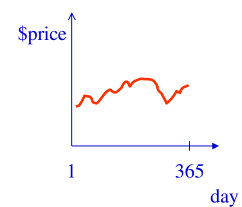
Outline

- 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'

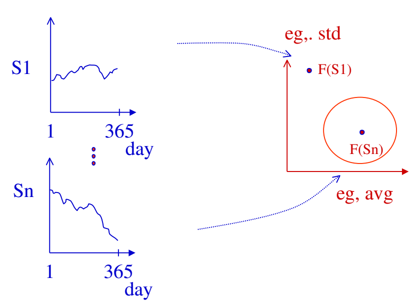
Eg., '*find stocks similar to MSFT*'

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

Examples of GEMINI

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

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]

Indexing - SAMs

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

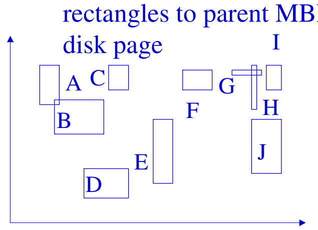
A: they group nearby points (or regions) together, on nearby disk pages, and answer spatial queries quickly ('range queries', 'nearest neighbor' queries etc)

For example:

Skip

R-trees

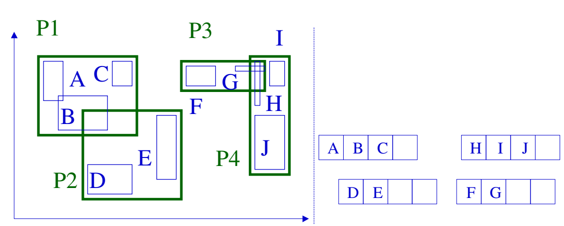
- [Guttman84] eg., w/ fanout 4: group nearby rectangles to parent MBRs; each group -> disk page



Skip

R-trees

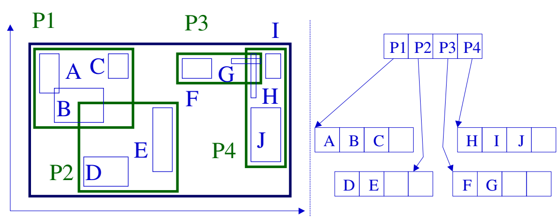
- eg., w/ fanout 4:



Skip

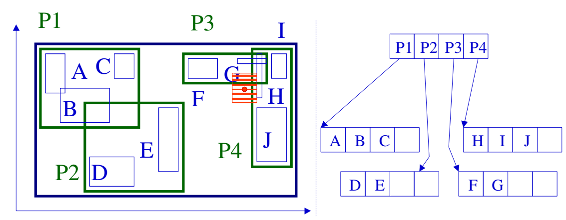
R-trees

- eg., w/ fanout 4:



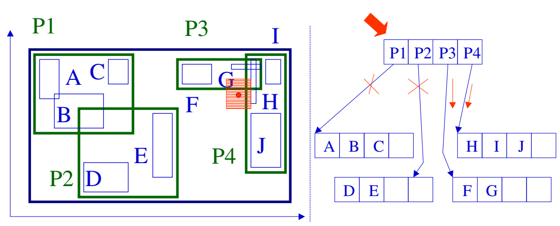
Skip

R-trees - range search?



Skip

R-trees - range search?



Conclusions

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

Outline

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

Outline

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

CMU SCS

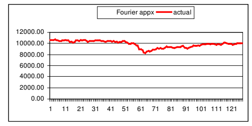
DFT and cousins

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

VLDB-02 © C. Faloutsos (2002) 55

CMU SCS

DFT and stocks

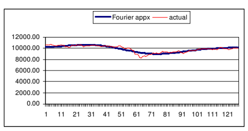


- Dow Jones Industrial index, 6/18/2001-12/21/2001

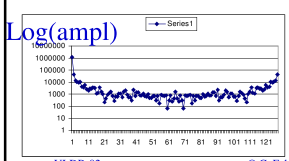
VLDB-02 © C. Faloutsos (2002) 56

CMU SCS

DFT and stocks



Log(ampl)



freq

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

VLDB-02 © C. Faloutsos (2002) 57

CMU SCS

Outline

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

VLDB-02 © C. Faloutsos (2002) 58

CMU SCS

SVD

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

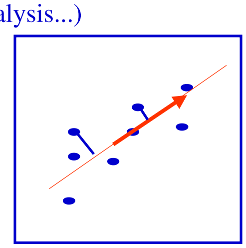
VLDB-02 © C. Faloutsos (2002) 59

CMU SCS

Singular Value Decomposition (SVD)

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

day2



day1

LSI: S. Dumais; M. Berry

KL: eg, Duda+Hart

PCA: eg., Jolliffe

Details: [Press+],

[Faloutsos96]

VLDB-02 © C. Faloutsos (2002) 60

CMU SCS

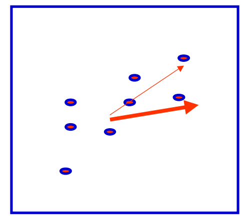
SVD

- **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?

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

Skip

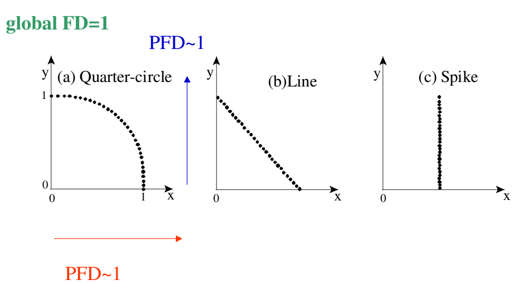
Feature extraction - w/ fractals

- Main idea: drop those attributes that don't affect the intrinsic ('fractal') dimensionality [Traina+, SBBD 2000]
- ie., drop attributes that depend on others (linearly or non-linearly!)

CMU SCS

Skip

Feature extraction - w/ fractals



CMU SCS

Outline

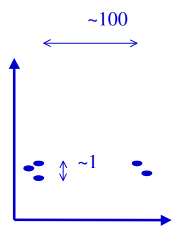
- Motivation
- Similarity Search and Indexing
  - distance functions
  - indexing
  - feature extraction
    - DFT, DWT, DCT (data independent)
    - SVD (data dependent)
    - MDS, FastMap

MDS / FastMap

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

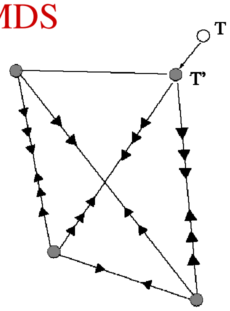
MDS/FastMap

	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

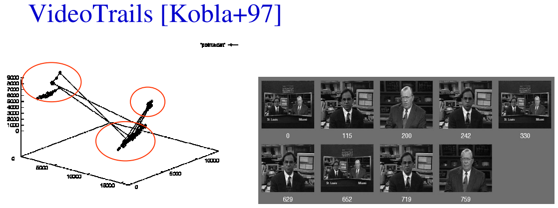


FastMap

- 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)

Outline

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

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.
- 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

Part 2: DSP (Digital Signal Processing)

CMU SCS

Outline

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

CMU SCS

Outline

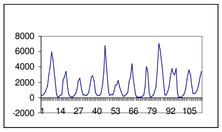
- DFT
  - Definition of DFT and properties
  - how to read the DFT spectrum
- DWT
  - Definition of DWT and properties
  - how to read the DWT scalogram

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

What does DFT do?

A: highlights the periodicities

Skip

DFT: definition

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

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

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

inverse DFT

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

DFT: definition

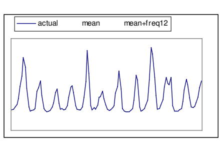
- **Good** news: Available in **all** symbolic math packages, eg., in ‘mathematica’

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

DFT: Amplitude spectrum

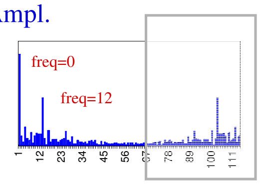
Amplitude: $A_f^{\,2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$

count



year

Ampl.



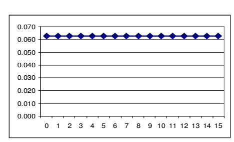
Freq.

Skip

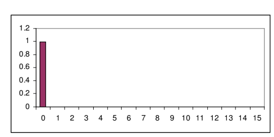
DFT: examples

flat

Amplitude



time



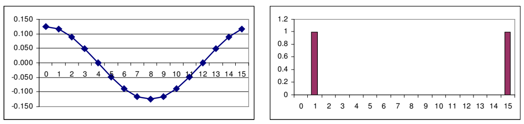
freq

CMU SCS

Skip

DFT: examples

Low frequency sinusoid



time freq

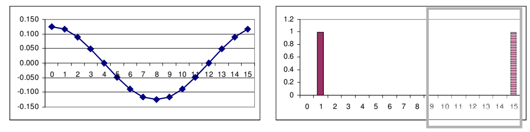
VLDB-02 © C. Faloutsos (2002) 91

CMU SCS

Skip

DFT: examples

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



time freq

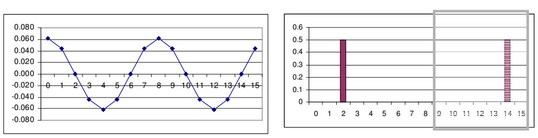
VLDB-02 © C. Faloutsos (2002) 92

CMU SCS

Skip

DFT: examples

- Higher freq. sinusoid



time freq

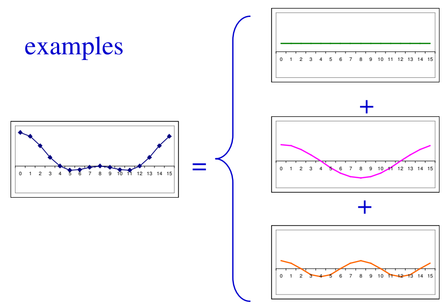
VLDB-02 © C. Faloutsos (2002) 93

CMU SCS

Skip

DFT: examples

examples



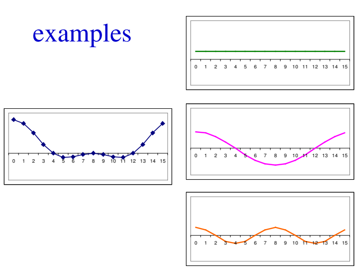
VLDB-02 © C. Faloutsos (2002) 94

CMU SCS

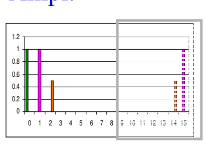
Skip

DFT: examples

examples



Ampl.



Freq.

VLDB-02 © C. Faloutsos (2002) 95

CMU SCS

Outline

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

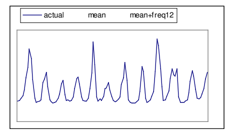
VLDB-02 © C. Faloutsos (2002) 96

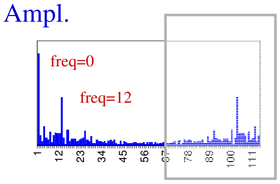
Outline

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

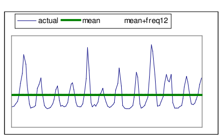
DFT: Amplitude spectrum

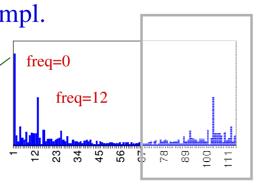
Amplitude: $A_f^{\,2} = \mathrm{Re}^2(X_f) + \mathrm{Im}^2(X_f)$



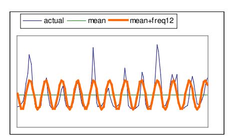


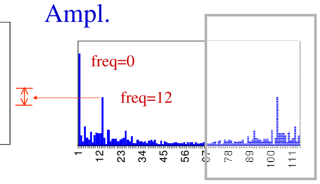
DFT: Amplitude spectrum





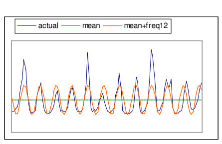
DFT: Amplitude spectrum

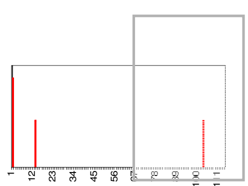




DFT: Amplitude spectrum

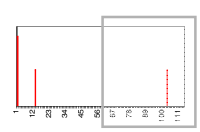
- excellent approximation, with only 2 frequencies!
- so what?





DFT: Amplitude spectrum

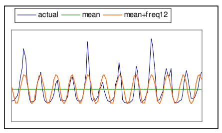
- excellent approximation, with only 2 frequencies!
- so what?
- A1: **(lossy) compression**
- A2: pattern discovery



CMU SCS

DFT: Amplitude spectrum

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



CMU SCS

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

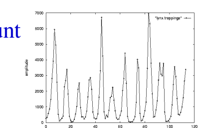
CMU SCS

Problem #1:

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

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

count



year

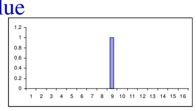
lynx caught per year (packets per day; virus infections per month)

CMU SCS

Wavelets - DWT

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

value



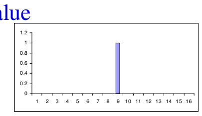
time

CMU SCS

Wavelets - DWT

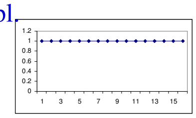
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!

value



time

Ampl.

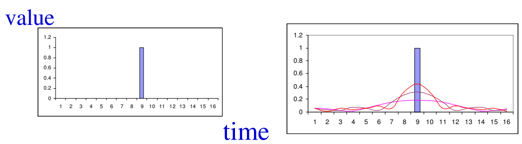


Freq.

CMU SCS

Wavelets - DWT

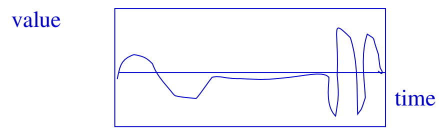
- DFT is great - but, how about compressing a spike?
- A: Terrible - all DFT coefficients needed!



CMU SCS

Wavelets - DWT

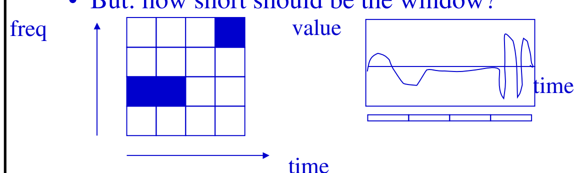
- Similarly, DFT suffers on short-duration waves (eg., baritone, silence, soprano)



CMU SCS

Wavelets - DWT

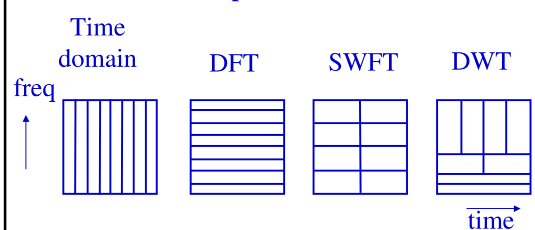
- Solution#1: Short window Fourier transform (SWFT)
- But: how short should be the window?



CMU SCS

Wavelets - DWT

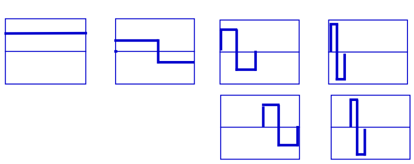
- Answer: **multiple** window sizes! -> DWT



CMU SCS

Haar Wavelets

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



CMU SCS

Skip

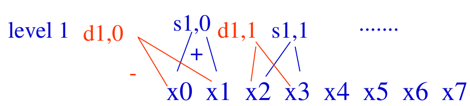
Wavelets - construction

x0 x1 x2 x3 x4 x5 x6 x7

CMU SCS

Skip

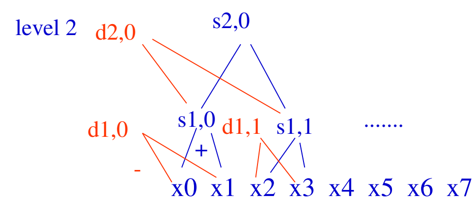
Wavelets - construction



CMU SCS

Skip

Wavelets - construction

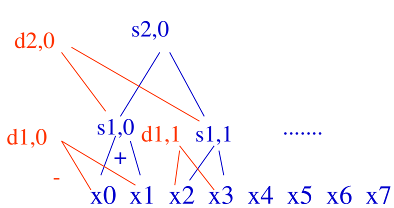


CMU SCS

Skip

Wavelets - construction

etc ...



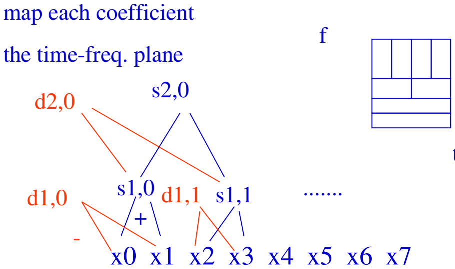
CMU SCS

Skip

Wavelets - construction

Q: map each coefficient

on the time-freq. plane



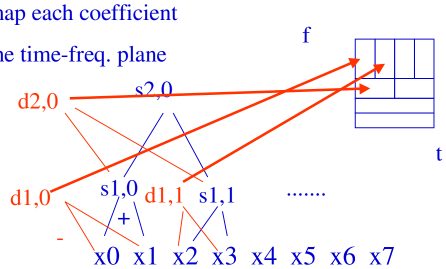
CMU SCS

Skip

Wavelets - construction

Q: map each coefficient

on the time-freq. plane



CMU SCS

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:

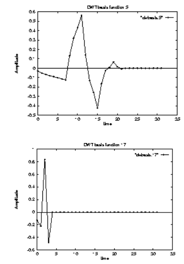
‘+’ can be some weighted addition

‘-’ 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, ...

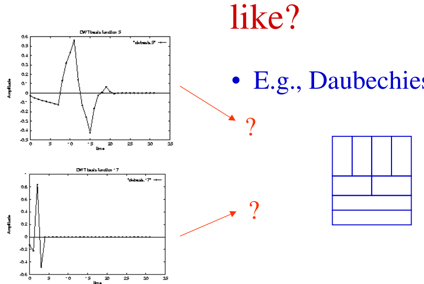
Wavelets - how do they look like?



- E.g., Daubechies-4

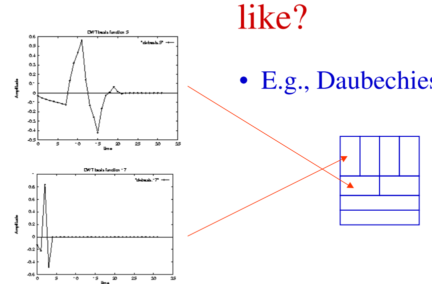
Wavelets - how do they look like?

- E.g., Daubechies-4



Wavelets - how do they look like?

- E.g., Daubechies-4

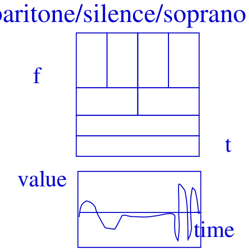


Outline

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

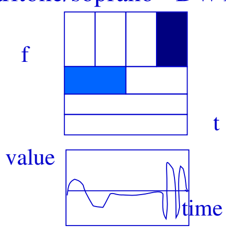
Wavelets - Drill#1:

- Q: baritone/silence/soprano - DWT?



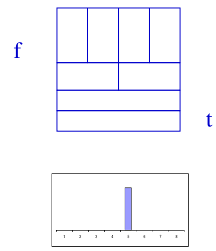
Wavelets - Drill#1:

- Q: baritone/soprano - DWT?



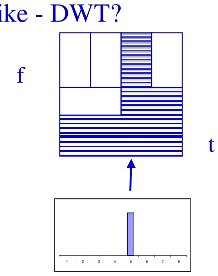
Wavelets - Drill#2:

- Q: spike - DWT?



Wavelets - Drill#2:

- Q: spike - DWT?



0.00 0.00 **0.71** 0.00

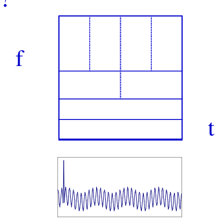
0.00 **0.50**

-0.35

0.35

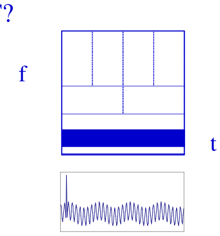
Wavelets - Drill#3:

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



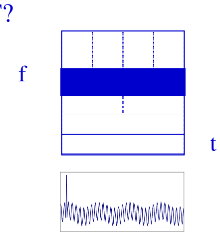
Wavelets - Drill#3:

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



Wavelets - Drill#3:

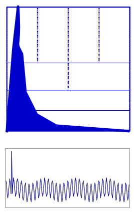
- Q: weekly + **daily** periodicity, + spike - DWT?



Wavelets - Drill#3:

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

f

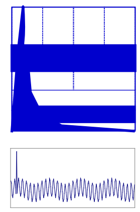


t

Wavelets - Drill#3:

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

f



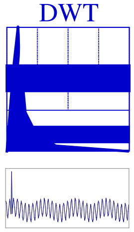
t

Wavelets - Drill#3:

- Q: DFT?

DWT

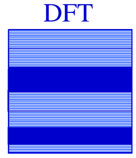
f



t

DFT

f



t

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

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!)

Overall Conclusions

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

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

BREAK!

Sensor Data Mining: Similarity search and pattern analysis

Christos Faloutsos
CMU

CMU SCS

Part 3: Linear Forecasting

VLDB-02 © C. Faloutsos (2002) 145

CMU SCS

Outline

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

VLDB-02 © C. Faloutsos (2002) 146

CMU SCS

Forecasting

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

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

VLDB-02 © C. Faloutsos (2002) 147

CMU SCS

Outline

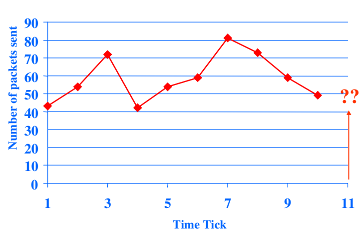
- Motivation
- ...
- Linear Forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
  - Examples
  - Conclusions

VLDB-02 © C. Faloutsos (2002) 148

CMU SCS

Problem#2: Forecast

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



VLDB-02 © C. Faloutsos (2002) 149

CMU SCS

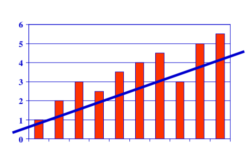
Forecasting: Preprocessing

MANUALLY:

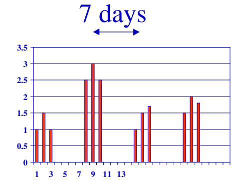
remove trends

spot periodicities

7 days



time



time

VLDB-02 © C. Faloutsos (2002) 150

CMU SCS

Problem#2: Forecast

- Solution: try to express

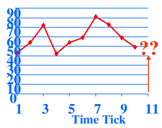
  x_t

  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

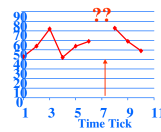
  x_t

  as a linear function of the past AND the future:

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

  (up to windows of w_{past} , w_{future})

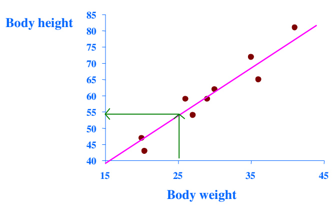
- EXACTLY the same algo's



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

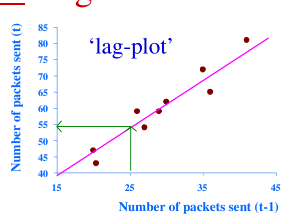
Linear Auto Regression:

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])

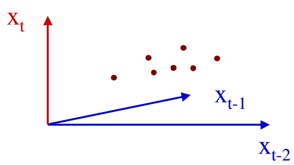
CMU SCS

Outline

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

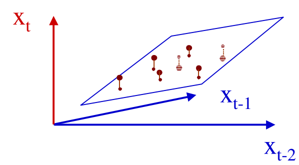
More details:

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



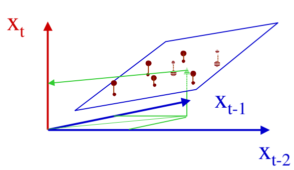
More details:

- 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!)



Skip

More details:

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

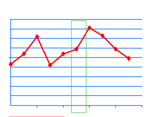
$$\mathbf{X}_{[N \times w]} \times \mathbf{a}_{[w \times 1]} = \mathbf{y}_{[N \times 1]}$$

- 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

Skip

More details:

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



Ind-var1 Ind-var-w

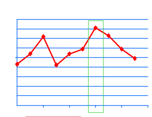
time

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

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

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

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**

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:

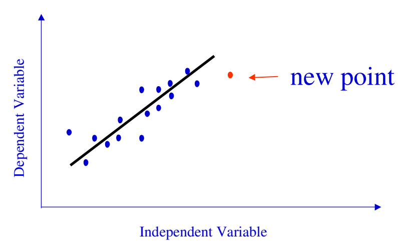
Skip

Even more details

- Given:

Skip

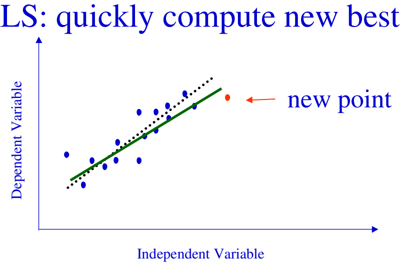
Even more details



Skip

Even more details

RLS: quickly compute new best fit



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

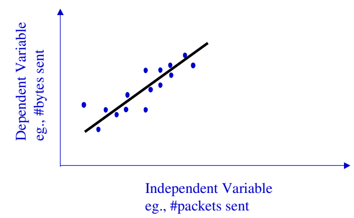
Even more details

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

CMU SCS

Skip

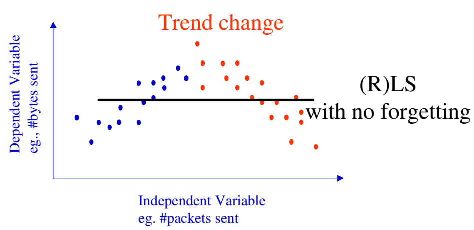
Adaptability - 'forgetting'



CMU SCS

Skip

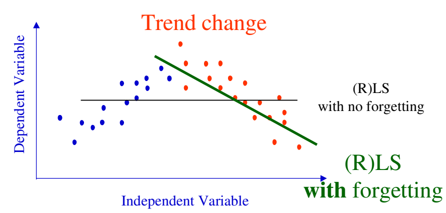
Adaptability - 'forgetting'



CMU SCS

Skip

Adaptability - 'forgetting'



- RLS: can *trivially* handle 'forgetting'

CMU SCS

Outline

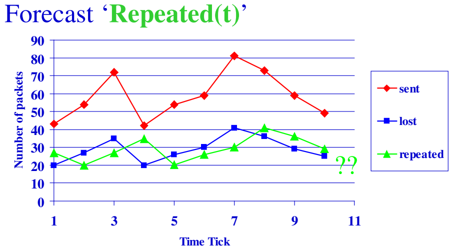
- Motivation
- ...
- Linear Forecasting
  - Auto-regression: Least Squares; RLS
  - Co-evolving time sequences
  - Examples
  - Conclusions

CMU SCS

Skip

Co-Evolving Time Sequences

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



CMU SCS

Skip

Solution:

Q: what should we do?

VLDB-02 © C. Faloutsos (2002) 175

CMU SCS

Skip

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])

VLDB-02 © C. Faloutsos (2002) 176

CMU SCS

Skip

B.II - Time Series Analysis Outline

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

VLDB-02 © C. Faloutsos (2002) 177

CMU SCS

Skip

Examples - Experiments

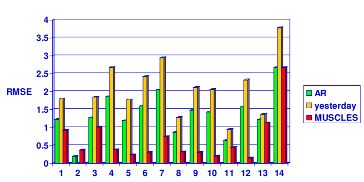
- Datasets
  - Modem pool traffic (14 modems, 1500 time-ticks; #packets per time unit)
  - AT&T WorldNet internet usage (several data streams; 980 time-ticks)
- Measures of success
  - Accuracy : Root Mean Square Error (RMSE)

VLDB-02 © C. Faloutsos (2002) 178

CMU SCS

Skip

Accuracy - “Modem”



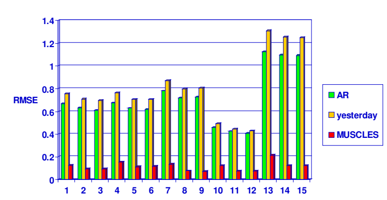
MUSCLES outperforms AR & “yesterday”

VLDB-02 © C. Faloutsos (2002) 179

CMU SCS

Skip

Accuracy - “Internet”



MUSCLES consistently outperforms AR & “yesterday”

VLDB-02 © C. Faloutsos (2002) 180

B.II - Time Series Analysis Outline

Skip

- 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]
- Research issue: how to avoid the manual preprocessing (de-trending; periodicities)

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

- [Yi+00] Byoung-Kee Yi et al.: *Online Data Mining for Co-Evolving Time Sequences*, ICDE 2000. (Describes MUSCLES and Recursive Least Squares)

Part 4: Bursty traffic and multifractals

CMU SCS

Outline

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

VLDB-02 © C. Faloutsos (2002) 187

CMU SCS

Outline

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

VLDB-02 © C. Faloutsos (2002) 188

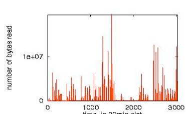
CMU SCS

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

VLDB-02 © C. Faloutsos (2002) 189

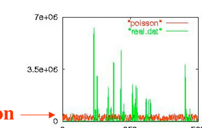
CMU SCS

Problem #1

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

 # bytes

Poisson



time

VLDB-02 © C. Faloutsos (2002) 190

CMU SCS

Motivation

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

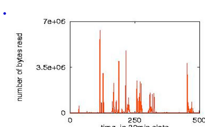
VLDB-02 © C. Faloutsos (2002) 191

CMU SCS

Q: any 'pattern'?

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

 # bytes



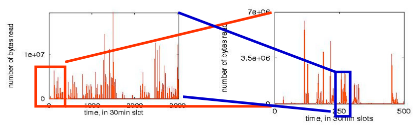
time

VLDB-02 © C. Faloutsos (2002) 192

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

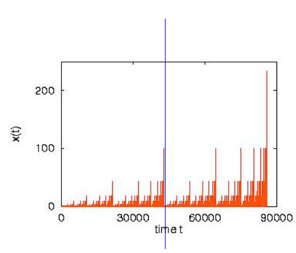
Approach

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

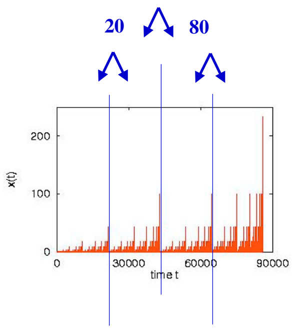
CMU SCS

binary multifractals

20 **80**



binary multifractals



Parameter estimation

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

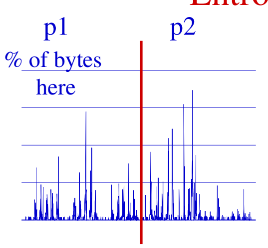
Parameter estimation

- 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]

Entropy plot

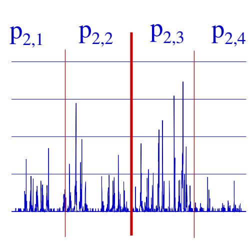
- Rationale:
  - burstiness: inverse of uniformity
  - entropy measures uniformity of a distribution
  - find entropy at several granularities, to see whether/how our distribution is close to uniform.

Entropy plot



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

Entropy plot

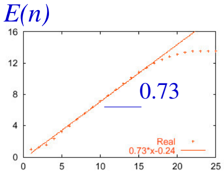


- Entropy *E(n)* after *n* levels of splits
- n=1: E(1)= - p1 log(p1)- p2 log(p2)
- n=2: $E(2) = - \Sigma_i \, p_{2,i} * \log_2 (p_{2,i})$

CMU SCS

Real traffic

Entropy
E(n)



# of levels (*n*)

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

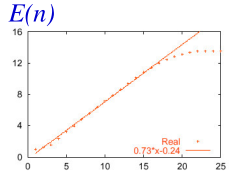
VLDB-02 © C. Faloutsos (2002) 205

CMU SCS

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

VLDB-02 © C. Faloutsos (2002) 206

CMU SCS

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?

VLDB-02 © C. Faloutsos (2002) 207

CMU SCS

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

VLDB-02 © C. Faloutsos (2002) 208

CMU SCS

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?

VLDB-02 © C. Faloutsos (2002) 209

CMU SCS

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?

VLDB-02 © C. Faloutsos (2002) 210

Skip

Entropy plot

- Repeat, for all points at same position:

Dim=0

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

Skip

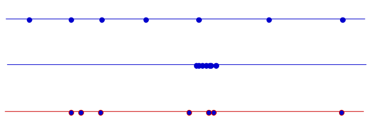
Entropy plot

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

Dim = 1

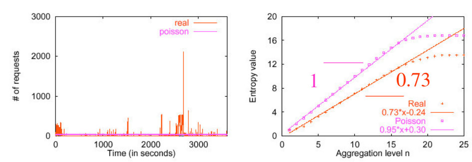
Dim=0

0<Dim<1



Some more entropy plots:

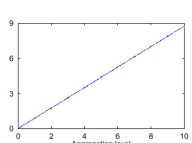
- Poisson vs real



Poisson: slope = ~1 -> uniformly distributed

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*

Outline

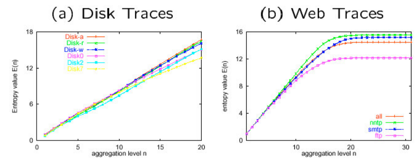
- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
  - Problem
  - Main idea (80/20, Hurst exponent)
  - Experiments - Results

Experimental setup

- Disk traces (from HP [Wilkes 93])
- web traces from LBL
  `http://repository.cs.vt.edu/lbl-conn-7.tar.Z`

Model validation

- Linear entropy plots

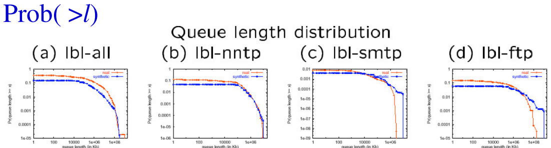


Bias factors *b*: 0.6-0.8
smallest *b* / smoothest: nntp traffic

Web traffic - results

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

Prob(>*l*)

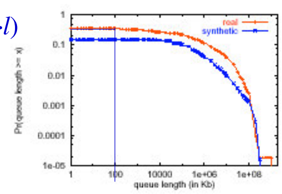


How to give guarantees? (queue length *l*)

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*)

Conclusions

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

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!)

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.

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.

Part 5: chaos and non-linear forecasting

Outline

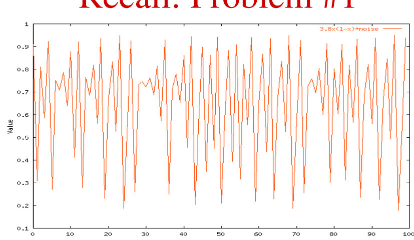
- Motivation
- Similarity Search and Indexing
- DSP
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

Detailed Outline

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

Recall: Problem #1

Value



Time

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

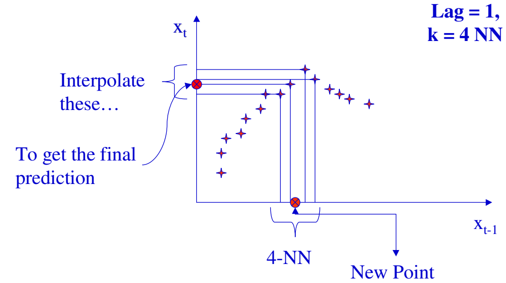
CMU SCS

How to forecast?

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

CMU SCS

General Intuition (Lag Plot)



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?

CMU SCS

Q1: Choosing lag *L*

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

CMU SCS

Q2: Choosing number of neighbors *k*

- Manually (typically ~ 1-10)

CMU SCS

Q3: How to interpolate?

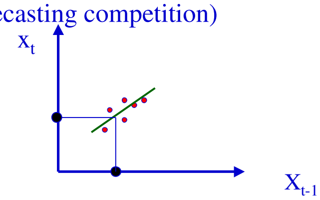
How do we interpolate between the *k* nearest neighbors?

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

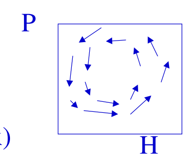
- Based on the "Takens' Theorem" [Takens81]
- which says that long enough delay vectors can do prediction, even if there are unobserved variables in the dynamical system (= diff. equations)

Skip

Theoretical foundation

Example: Lotka-Volterra equations

$$dH/dt = r\,H - a\,H{*}P$$
$$dP/dt = b\,H{*}P - m\,P$$



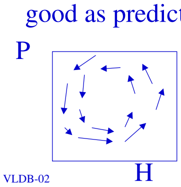
H is count of prey (e.g., hare)
P is count of predators (e.g., lynx)

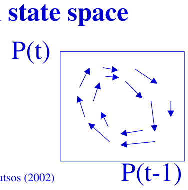
Suppose only P(t) is observed (t=1, 2, …).

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**

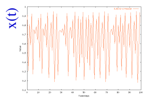




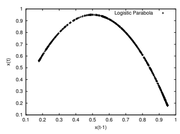
Detailed Outline

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

Datasets

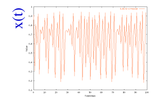


Logistic Parabola:
$x_t = a x_{t-1}(1-x_{t-1})$ + noise
Models population of flies [R. May/1976]

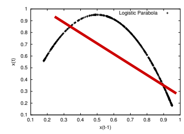


Lag-plot

Datasets

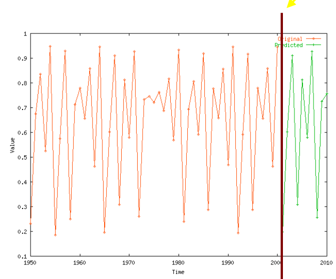


Logistic Parabola:
$x_t = a x_{t-1}(1-x_{t-1})$ + noise
Models population of flies [R. May/1976]



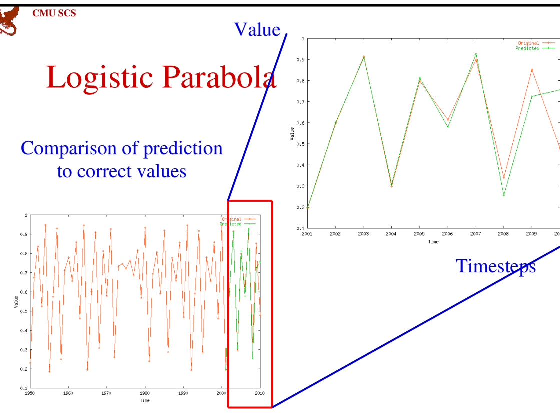
Lag-plot
ARIMA: fails

Logistic Parabola



Logistic Parabola

Comparison of prediction to correct values

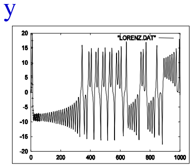


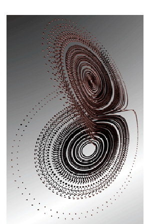
Skip

Datasets

Value

LORENZ: Models convection currents in the air
dx / dt = a (y - x)
dy / dt = x (b - z) - y
dz / dt = xy - c z

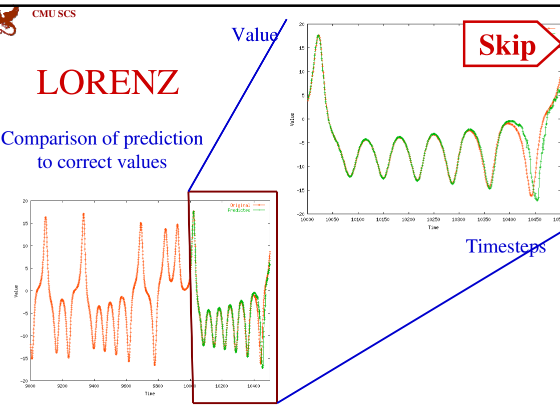




Skip

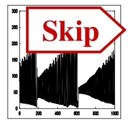
LORENZ

Comparison of prediction to correct values



Value

Datasets



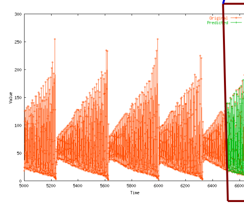
Skip

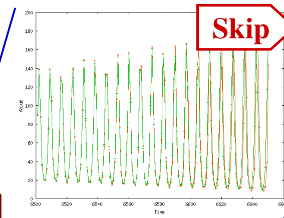
- LASER: fluctuations in a Laser over time (used in Santa Fe competition)

Time

Laser

Comparison of prediction to correct values





Value

Timesteps

Skip

Conclusions

- Lag plots for non-linear forecasting (Takens' theorem)
- Determining *L*-optimal, *k*-optimal and interpolation method: need research
- suitable for 'chaotic' signals

References

- 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.

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.)

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

VLDB-02 © C. Faloutsos (2002) 253

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) methodology

VLDB-02 © C. Faloutsos (2002) 254

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) methodology
- Bursty traffic: **multifractals** (80-20 'law')

VLDB-02 © C. Faloutsos (2002) 255

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) methodology
- Bursty traffic: **multifractals** (80-20 'law')
- Non-linear forecasting: **lag-plots** (Takens)

VLDB-02 © C. Faloutsos (2002) 256

CMU SCS

Overall conclusions

- Solved problems and open questions

	Similarity search – non-linear forecasting	DWT	ARIMA	Multifractals
Semi-infinite		yes [Gilbert+01]	yes	yes
Incr. Upd	yes			
Mem. Req.	Large			
Automatic?	NO (Lag? K-opt?)	yes	NO (lag? Trends? Periodicities?)	yes
Forecasting?	yes	NO	yes	NO

VLDB-02 © C. Faloutsos (2002) 257

CMU SCS

'Take home' messages

- Hard, but desirable query for sensor data: '*find patterns / outliers*'
- We need **fast**, **automated** such tools
  - Many great tools exist (DWT, ARIMA, …)
  - some are readily usable; others need to be made scalable / single pass/ automatic

VLDB-02 © C. Faloutsos (2002) 258

THANK YOU!

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