15-826: Multimedia Databases and Data Mining

Lecture #25: Time series mining and forecasting
Christos Faloutsos

Must-Read Material
• Byong-Kee Yi, Nikolaos D. Sidiropoulos, Theodore Johnson, H.V. Jagadish, Christos Faloutsos and Alex Biliris, Online Data Mining for Co-Evolving Time Sequences, ICDE, Feb 2000.
• Chungmin Melvin Chen and Nick Roussopoulos, Adaptive Selectivity Estimation Using Query Feedbacks, SIGMOD 1994

Thanks
Deepay Chakrabarti (CMU)
Prof. Dimitris Gunopulos (UCR)
Spiros Papadimitriou (CMU)
Mengzhi Wang (CMU)
Prof. Byoung-Kee Yi (Pohang U.)
Outline

• Motivation
  • Similarity search – distance functions
  • Linear Forecasting
  • Bursty traffic - fractals and multifractals
  • Non-linear forecasting
  • Conclusions

Problem definition

• Given: one or more sequences
  \( x_1, x_2, \ldots, x_t, \ldots \)
  \( (y_1, y_2, \ldots, y_t, \ldots) \)

• 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

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

<table>
<thead>
<tr>
<th>#bytes</th>
<th>Disk traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000000</td>
<td></td>
</tr>
<tr>
<td>1500000</td>
<td></td>
</tr>
<tr>
<td>1000000</td>
<td></td>
</tr>
<tr>
<td>500000</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Problem #1:

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

Find: patterns, periodicities, and/or compress

- lynx caught per year
- (packets per day; temperature per day)
**Problem #2: Forecast**

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

**Problem #2’: Similarity search**

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

**Problem #3:**

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

Patterns, 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)

Outline

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

Outline

• Motivation
  ➤ Similarity search and distance functions
    – Euclidean
    – Time-warping
  • …
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

\[ D(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} (x_i - y_i)^2 \]

\[ L_p(\mathbf{x}, \mathbf{y}) = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p} \]

- \(L_1\): city-block = Manhattan
- \(L_2\) = Euclidean
- \(L_{\infty}\)

Observation #1

- Time sequence -> n-d vector

Day-n

... Day-2

Day-1
Observation #2

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

Time Warping

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

‘stutters’:
Time warping

Q: how to compute it?
A: dynamic programming

\[ D(i, j) = \text{cost to match} \]

prefix of length \( i \) of first sequence \( x \) with prefix of length \( j \) of second sequence \( y \)

Thus, with no penalty for stutter, for sequences \( x_1, x_2, \ldots, x_i; \ y_1, y_2, \ldots, y_j \):

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

VERY SIMILAR to the string-editing distance

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

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]

Other Distance functions

- In [Keogh+, KDD’04]: parameter-free, MDL based
Conclusions

Prevailing distances:
– Euclidean and
– time-warping

Outline

• Motivation
• Similarity search and distance functions
→ Linear Forecasting
• Bursty traffic - fractals and multifractals
• Non-linear forecasting
• Conclusions

Linear Forecasting
Forecasting

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

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

Outline

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

Reference

(Describes MUSCLES and Recursive Least Squares)
**Problem #2: Forecast**

- Example: give \(x_{t-1}, x_{t-2}, \ldots\), forecast \(x_t\)

**Forecasting: Preprocessing**

MANUALLY:
- remove trends
- spot periodicities

**Problem #2: Forecast**

- Solution: try to express \(x_t\) as a linear function of the past: \(x_{t-2}, x_{t-2}, \ldots\)
  (up to a window of \(w\))

Formally:

\[ x_t \approx a_1 x_{t-1} + \ldots + a_w x_{t-w} + \text{noise} \]
(Problem: Back-cast; interpolate)

- Solution - interpolate: try to express
  \[ x_t \]
  as a linear function of the past AND the future:
  \[ x_{t-1}, x_{t-2}, \ldots, x_{t-w_{\text{past}}}, x_{t+1}, x_{t+2}, \ldots, x_{t+w_{\text{future}}} \]
  (up to windows of \( w_{\text{past}}, w_{\text{future}} \))
- EXACTLY the same algo's

---

Linear Regression: idea

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

---

Linear Auto Regression:
**Linear Auto Regression:**

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


**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!)

\[ x_{t-2} \quad x_{t-1} \quad x_t \]

More details:

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

[Diagram]

More details:

• Q1: Can it work with window \( w > 1? \)
  
  • A1: YES! The problem becomes:

\[ X_{[N \times w]} \times a_{[w \times 1]} = y_{[N \times 1]} \]

• OVER-CONSTRAINED
  
  – \( a \) is the vector of the regression coefficients
  
  – \( X \) has the \( N \) values of the \( w \) indep. variables
  
  – \( y \) has the \( N \) values of the dependent variable
More details:

1. \( X_{[N \times w]} \times a_{[w \times 1]} = y_{[N \times 1]} \)

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

More details:

Q2: How to estimate \( a_1, a_2, \ldots, a_w = a \)?

A2: with Least Squares fit

\[
\hat{a} = (X^T \times X)^{-1} \times (X^T \times y)
\]

(Moore-Penrose pseudo-inverse)

\( a \) is the vector that minimizes the RMSE from \( y \)

<identical math with ‘query feedbacks’>
More details

- Straightforward solution:
  \[ \mathbf{a} = (\mathbf{X}^T \times \mathbf{X})^{-1} \times (\mathbf{X}^T \times \mathbf{y}) \]

- Observations:
  - Sample matrix \( \mathbf{X} \) grows over time
  - Needs matrix inversion
  - \( O(Nw^2) \) computation
  - \( O(Nw) \) storage

---

Even more details

- Q3: Can we estimate \( \mathbf{a} \) incrementally?
- A3: Yes, with the brilliant, classic method of ‘Recursive Least Squares’ (RLS) (see, e.g., [Yi+00], for details).
  
- We can do the matrix inversion, WITHOUT inversion! (How is that possible?!)

---

Even more details

- Q3: Can we estimate \( \mathbf{a} \) incrementally?
- A3: Yes, with the brilliant, classic method of ‘Recursive Least Squares’ (RLS) (see, e.g., [Yi+00], for details).
- We can do the matrix inversion, WITHOUT inversion! (How is that possible?!)
- A: our matrix has special form: \( (\mathbf{X}^T \mathbf{X}) \)
At the $N+1$ time tick:

\[ \begin{bmatrix} x_{N}^T \\ x_{N+1} \end{bmatrix} \]

More details

- Let $G_N = (X_N^T \times X_N)^{-1}$ ("gain matrix")
- $G_{N+1}$ can be computed recursively from $G_N$

\[ G_{N+1} = G_N - [c]^{-1} \times [G_N \times x_{N+1}^T] \times x_{N+1} \times G_N \]

\[ c = [1 + x_{N+1}^T \times G_N \times x_{N+1}] \]

EVEN more details:

Let’s elaborate

(VERY IMPORTANT, VERY VALUABLE!)
EVEN more details:

$$a = (X_{N+1}^T \times X_{N+1})^{-1} \times (X_{N+1}^T \times y_{N+1})$$

\[\begin{array}{ccc}
[w \times 1] & [(N+1) \times w] & [(N+1) \times 1] \\
\end{array}\]

\[\begin{array}{ccc}
[w \times (N+1)] & [w \times (N+1)] \\
\end{array}\]
EVEN more details:

\[ a = [X_{N+1}^T \times X_{N+1}]^{-1} \times [X_{N+1}^T \times y_{N+1}] \]

'gain matrix' \[ G_{N+1} = [X_{N+1}^T \times X_{N+1}]^{-1} \]
\[ G_{N+1} = G_n - [c]^{-1} \times [G_n \times x_{N+1}^T \times x_{N+1} \times G_n] \]
\[ c = [1 + x_{N+1} \times G_n \times x_{N+1}^T] \]

EVEN more details:

\[ G_{N+1} = G_n - [c]^{-1} \times [G_n \times x_{N+1}^T \times x_{N+1} \times G_n] \]
\[ c = [1 + x_{N+1} \times G_n \times x_{N+1}^T] \]

EVEN more details:

\[ \text{SCALAR!} \]
\[ c = [1 + x_{N+1} \times G_n \times x_{N+1}^T] \]
Altogether:

\[ a = \left[ X_{N+1} \right]^T \times X_{N+1}^{-1} \times \left[ X_{N+1} \right]^T \times y_{N+1} \]

\[ G_{N+1} = \left[ X_{N+1} \right]^T \times X_{N+1}^{-1} \]

\[ G_{N+1} = G_N - [c]^{-1} \times [G_N \times x_{N+1} ] \times x_{N+1} \times G_N \]

\[ c = \left[ 1 + x_{N+1} \times G_N \times x_{N+1} \right]^T \]

Comparison:

- **Straightforward Least Squares**
  - Needs huge matrix (growing in size) \( O(N^2w) \)
  - Costly matrix operation \( O(N^2w^2) \)

- **Recursive LS**
  - Need much smaller, fixed size matrix \( O(w \times w) \)
  - Fast, incremental computation \( O(1 \times w^2) \)
  - no matrix inversion

\[ N = 10^6, \quad w = 1-100 \]
Pictorially:

- Given:

Independent Variable

Dependent Variable

Pictorially:

new point

Pictorially:

RLS: quickly compute new best fit

new point
**Even more details**

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

**Adaptability - ‘forgetting’**

- Independent Variable: eg., packets sent
- Dependent Variable: eg., bytes sent

**Trend change**

- (R)LS with no forgetting
Adaptability - ‘forgetting’

- RLS: can *trivially* handle ‘forgetting’

How to choose ‘w’?

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

Reference

Answer:

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

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

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

More details...

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

Results - Synthetic data

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

• Automobile traffic
  – Daily periodicity
  – Bursty “noise” at smaller scales
• AR fails to capture any trend
• Seasonal AR estimation fails

Results - Real data

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

Complexity

• Model update
  Space: $O(\lg N + mk^2) \sim O(\lg N)$
  Time: $O(k^2) \sim O(1)$
• Where
  – $N$: number of points (so far)
  – $k$: number of regression coefficients; fixed
  – $m$: number of linear models; $O(\lg N)$
Outline

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

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])

---

**Forecasting - Outline**

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

---

**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)
Accuracy - “Modem”

MUSCLES outperforms AR & “yesterday”

Accuracy - “Internet”

MUSCLES consistently outperforms AR & “yesterday”

Linear forecasting - 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]
- (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
  Verlag.
Additional Reading

- [Papadimitriou vldh2003] Spiros Papadimitriou, Anthony Brockwell and Christos Faloutsos
- [Yi+00] Byoung-Kee Yi et al.: Online Data Mining for Co-Evolving Time Sequences, ICDE 2000. (Describes MUSCLES and Recursive Least Squares)

Outline

- Motivation
- Similarity search and distance functions
- Linear Forecasting
- Bursty traffic - fractals and multifractals
  - Non-linear forecasting
  - Conclusions

Bursty Traffic & Multifractals
Outline

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

Reference:


Full thesis: CMU-CS-05-185

Recall: Problem #1:

Goal: given a signal (e.g., #bytes over time)
Find: patterns, periodicities, and/or compress

<table>
<thead>
<tr>
<th>#bytes</th>
<th>Bytes per 30'</th>
</tr>
</thead>
<tbody>
<tr>
<td>(packets per day; earthquakes per year)</td>
<td>time</td>
</tr>
</tbody>
</table>
Problem #1

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

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?
Solution: self-similarity

But:

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

Outline

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

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

Approach

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

Binary multifractals

\[ 20 \land 80 \]
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
  - $n=1$: $E(1) = -p_1 \log_2(p_1) - p_2 \log_2(p_2)$
  - $n=2$: $E(2) = -\sum \pi_{2,i} \log_2(\pi_{2,i})$
Real traffic

Entropy $E(n)$

- Has linear entropy plot ($\Rightarrow$ self-similar)

Observation - intuition:

Entropy $E(n)$

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

Observation - intuition:

Entropy $E(n)$

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

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

Is next MSB = 0?
Entropy plot

• Slope \sim intrinsic dimensionality (in fact, ‘Information fractal dimension’)
• = info bit per coordinate bit - eg

Dim = 1

Info value = 1 bit
= \text{E}(2) - \text{E}(1) = \\
\text{Is MSB 0?}
\text{Is next MSB =0?}

\text{Dim=0}

Entropy plot

• Repeat, for all points at same position:

\text{Dim=0}

Entropy plot

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

\text{Dim=0}
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 |

(Fractals, again)

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

Cantor dust

• Eliminate the middle third
• Recursively!
Cantor dust

Cantor dust

Cantor dust
Cantor dust

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

Some more entropy plots:

- Poisson vs real

Poisson: slope = -1 -> uniformly distributed
**b-model**

- b-model traffic gives perfectly linear plot
- Lemma: its slope is
  \[ \text{slope} = -b \log_2 b - (1-b) \log_2 (1-b) \]
- 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
  - [Repository](http://repository.cs.vt.edu/)
  - *lbl-conn-7.tar.Z*
Model validation

- Linear entropy plots

(a) Disk Traces  
(b) Web Traces

Bias factors $b$: 0.6-0.8  
smallest $b$ / smoothest: mntp traffic

Web traffic - results

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

Prob($l$)

(a) lbl-all  
(b) lbl-nntp  
(c) lbl-smtp  
(d) lbl-ftp

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

Books


Further reading:

Further reading


Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

Chaos and non-linear forecasting
Reference:

[ Deepay Chakrabarti and Christos Faloutsos
F4: Large-Scale Automated Forecasting using Fractals CIKM 2002, Washington DC, Nov. 2002.]

Detailed Outline

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

Recall: Problem #1

Given a time series \{x_t\}, predict its future course, that is, \(x_{t+1}, x_{t+2}, \ldots\)
Datasets

Logistic Parabola:
\[ x_t = ax_t(1-x_t) + \text{noise} \]
Models population of flies [R. May/1976]

How to forecast?

• ARIMA - but: linearity assumption

• ANSWER: ‘Delayed Coordinate Embedding’ = Lag Plots [Sauer92]
  ~ nearest-neighbor search, for past incidents
General Intuition (Lag Plot)

- Interpolate these…
- To get the final prediction

Lag = 1, k = 4 NN

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?

Q1: Choosing lag $L$

- Manually (16, in award winning system by [Sauer94])
Q2: Choosing number of neighbors $k$

- Manually (typically ~ 1-10)

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?)

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)

Example: Lotka-Volterra equations

\[
\begin{align*}
\frac{dH}{dt} &= r H - a H P \\
\frac{dP}{dt} &= b H P - m P
\end{align*}
\]

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, …).
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

\[ P(t) \]

Detailed Outline

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

Datasets

Logistic Parabola:

\[ x_t = ax_t(1-x_t) + \text{noise} \]

Models population of flies [R. May/1976]
Datasets

Logistic Parabola:

\[ x_t = ax_{t-1}(1-x_{t-1}) + \text{noise} \]

Models population of flies [R. May/1976]

Lag-plot
ARIMA: fails

Logistic Parabola

Value

Comparison of prediction to correct values

Timesteps

Our Prediction from here

Timesteps

Value
Datasets

LORENZ: Models convection currents in the air
\[ \frac{dx}{dt} = a (y - x) \]
\[ \frac{dy}{dt} = x (b - z) - y \]
\[ \frac{dz}{dt} = xy - cz \]
Conclusions

• Lag plots for non-linear forecasting (Takens’ theorem)
• suitable for ‘chaotic’ signals

References

References


Overall conclusions

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

- Signal processing: DWT is a powerful tool
Overall conclusions

- Similarity search: Euclidean/time-warping; feature extraction and SAMs
- Signal processing: DWT is a powerful tool
- Linear Forecasting: AR (Box-Jenkins) methodology; AWSOM
- Bursty traffic: multifractals (80-20 ‘law’)
- Non-linear forecasting: lag-plots (Takens)