

15-826: Multimedia (Databases) and Data Mining

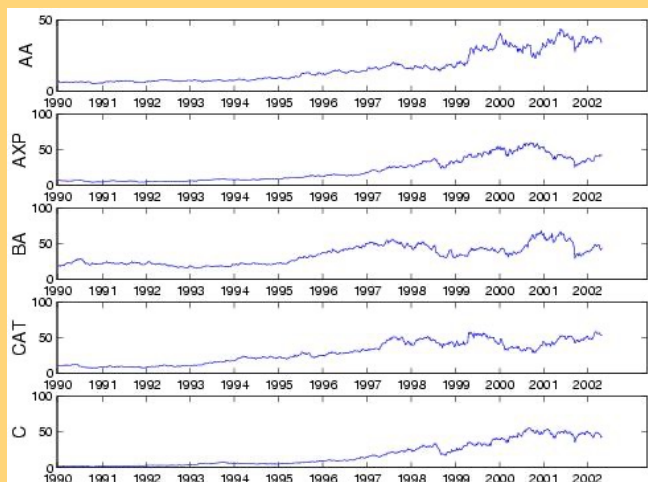
Lecture #25: Time series mining and
forecasting

Christos Faloutsos



Problem:

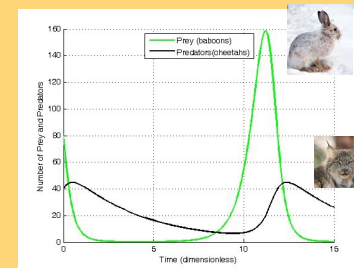
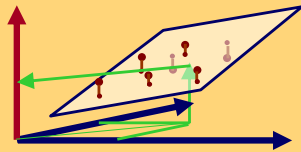
Q: mine/forecast (one, or more)
time sequences





Answers

- Similarity search: **Euclidean/time-warping; feature extraction and SAMs**
- Linear Forecasting: **AR (Box-Jenkins)**
- Non-linear forecasting: **lag-plots**
- Gray-box modeling: **Lotka-Volterra**



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 (UT-Austin)



Spiros Papadimitriou (Rutgers)



Prof. Byoung-Kee Yi (Samsung)



Outline

- ➔ • Motivation
- Similarity search – distance functions
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Gray box modeling – Lotka Volterra eq's
- Conclusions

Problem definition

- Given: one or more sequences

$x_1, x_2, \dots, x_t, \dots$

$(y_1, y_2, \dots, y_p, \dots$

$\dots)$

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

Motivation - Applications

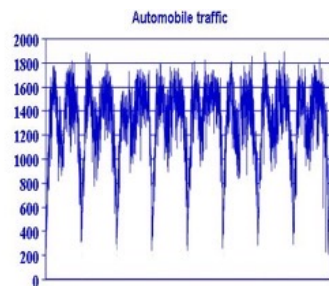
- Financial, sales, economic series
- Medical
 - ECGs +; blood pressure etc monitoring
 - reactions to new drugs
 - elderly care

Motivation - Applications (cont' d)

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

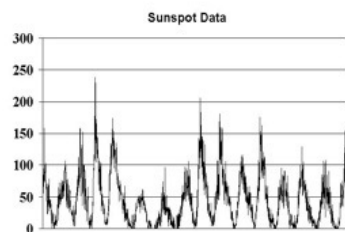
Motivation - Applications (cont' d)

- 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

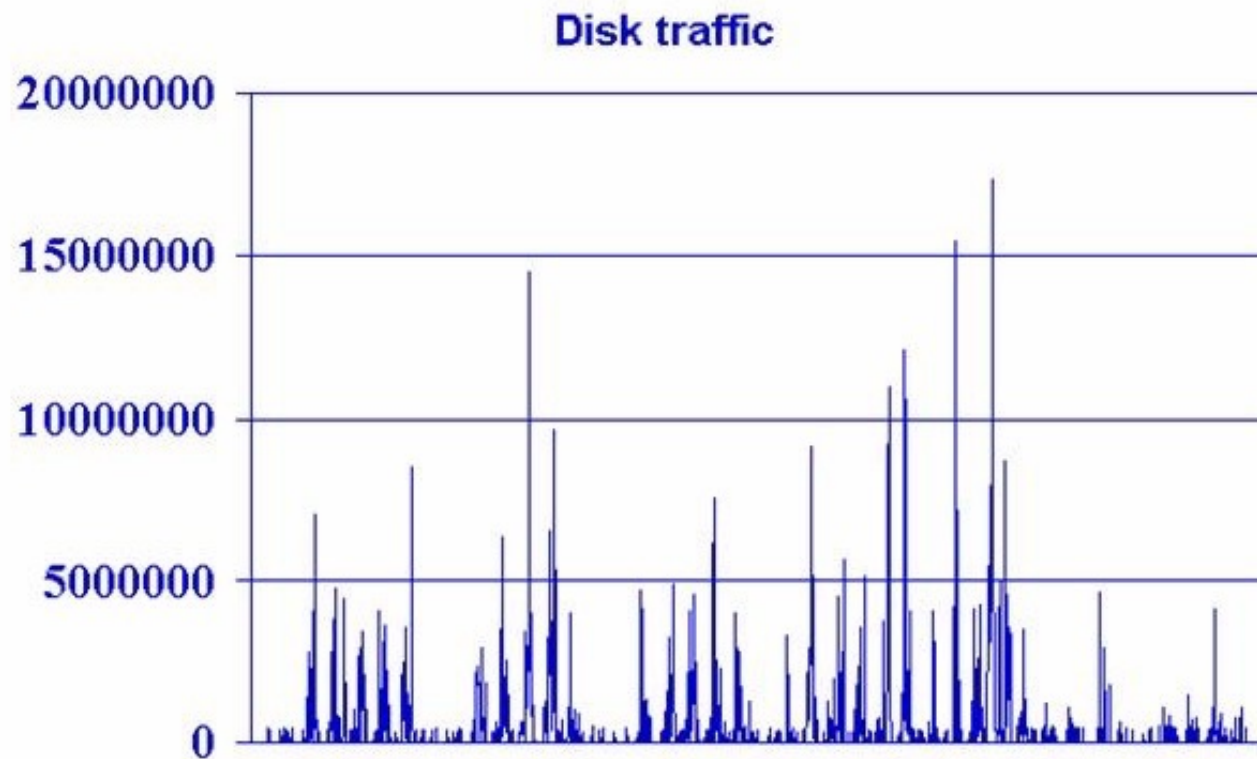


Motivation - Applications (cont' d)

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

Stream Data: Disk accesses

#bytes

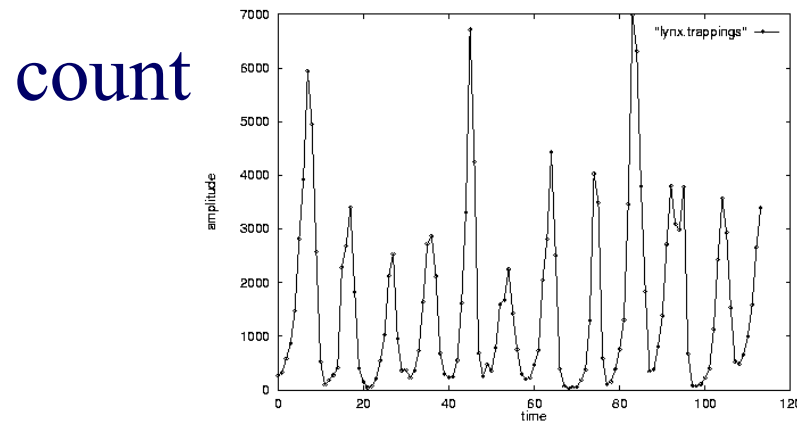


time

Problem #1:

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

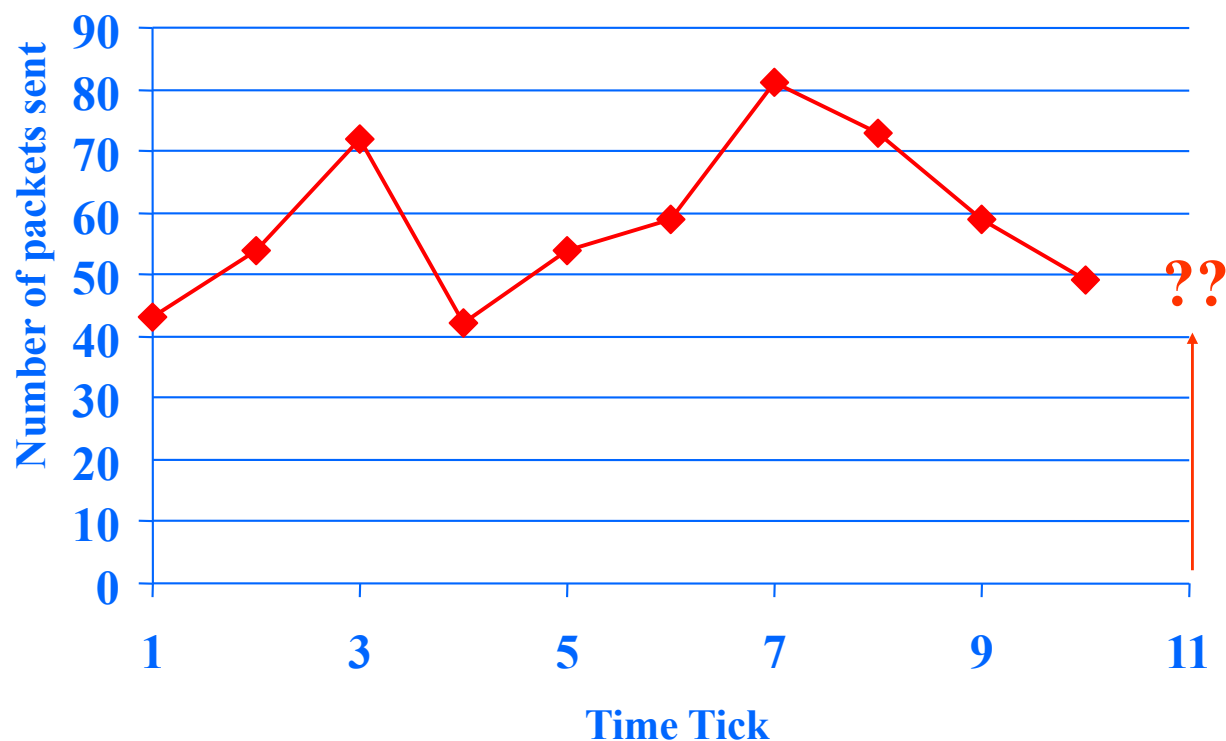
Find: patterns, periodicities, and/or compress



year

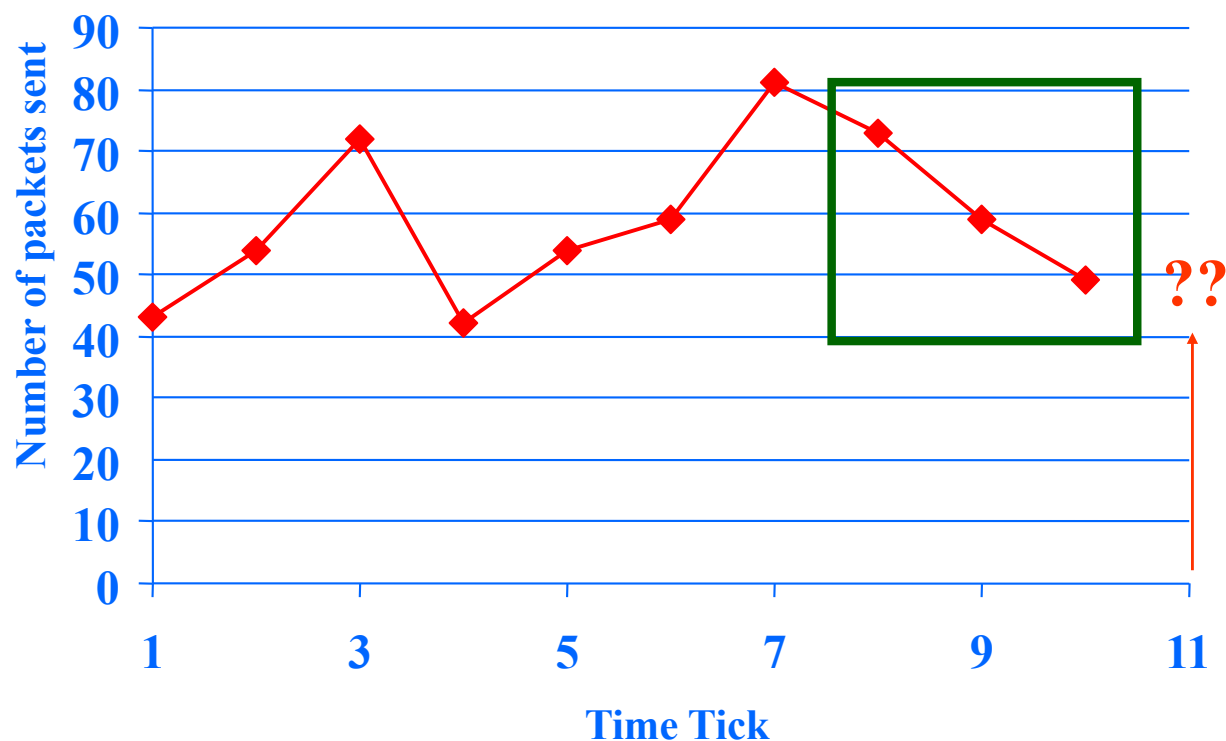
Problem#2: Forecast

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



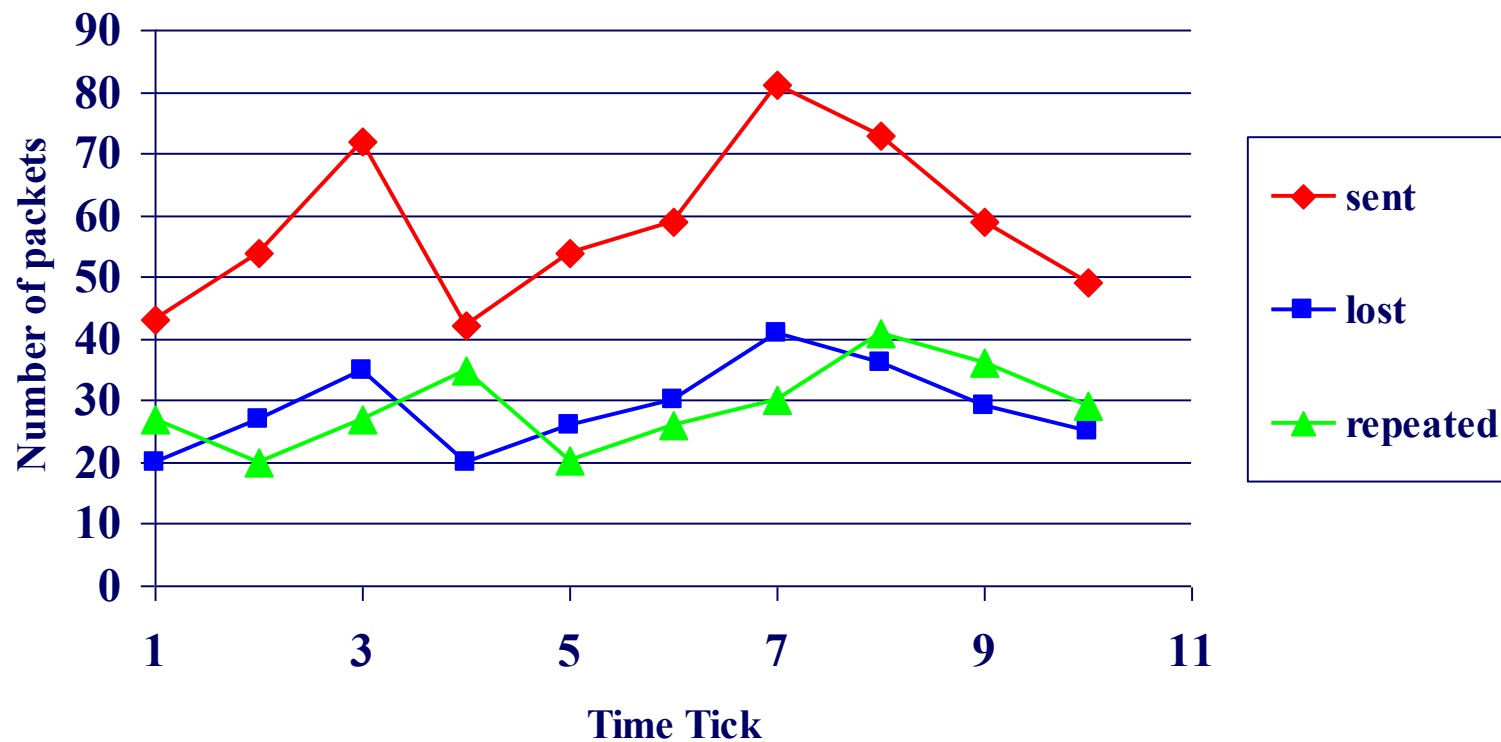
Problem#2' : Similarity search

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



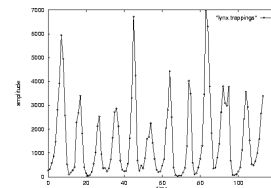
Problem #3:

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



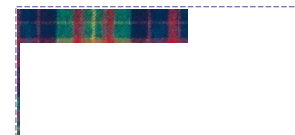


Important observations



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

- To do forecasting, we need
 - to find patterns/rules
 - **compress**
 - 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
- Gray box modeling – Lotka Volterra eq's
- Conclusions



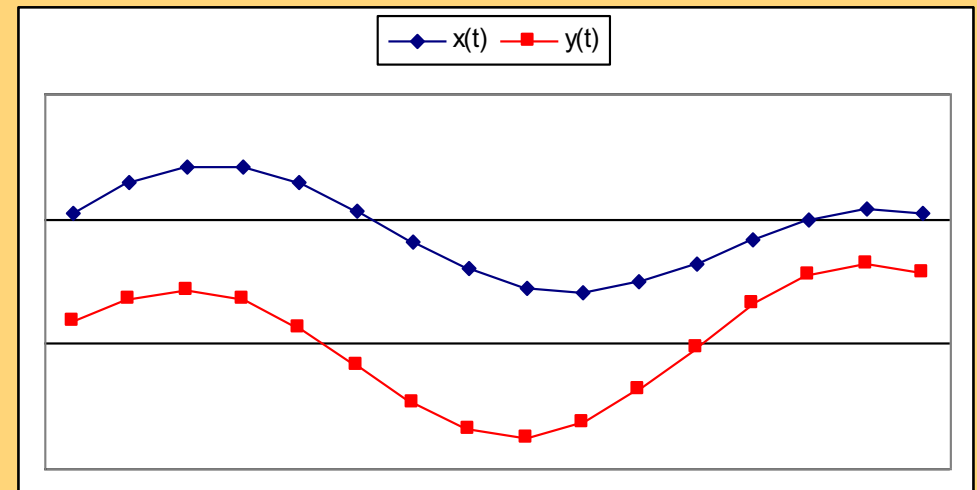
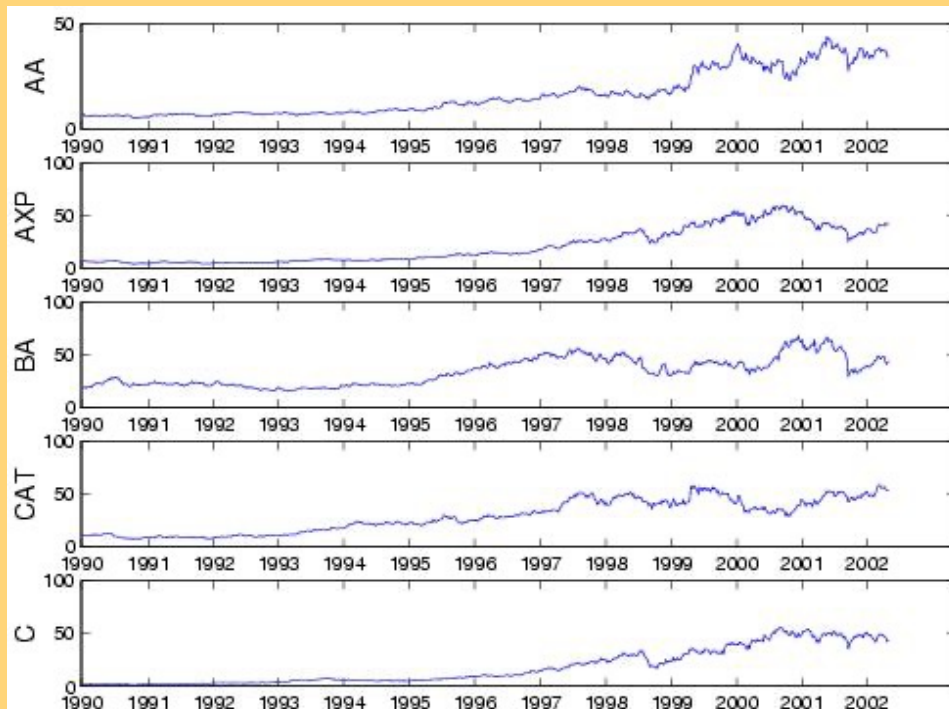
Detailed Outline

- Motivation
- ➔ • Similarity search and distance functions
 - Euclidean
 - Time-warping
- ...



Problem:

Q: How similar are two sequences?

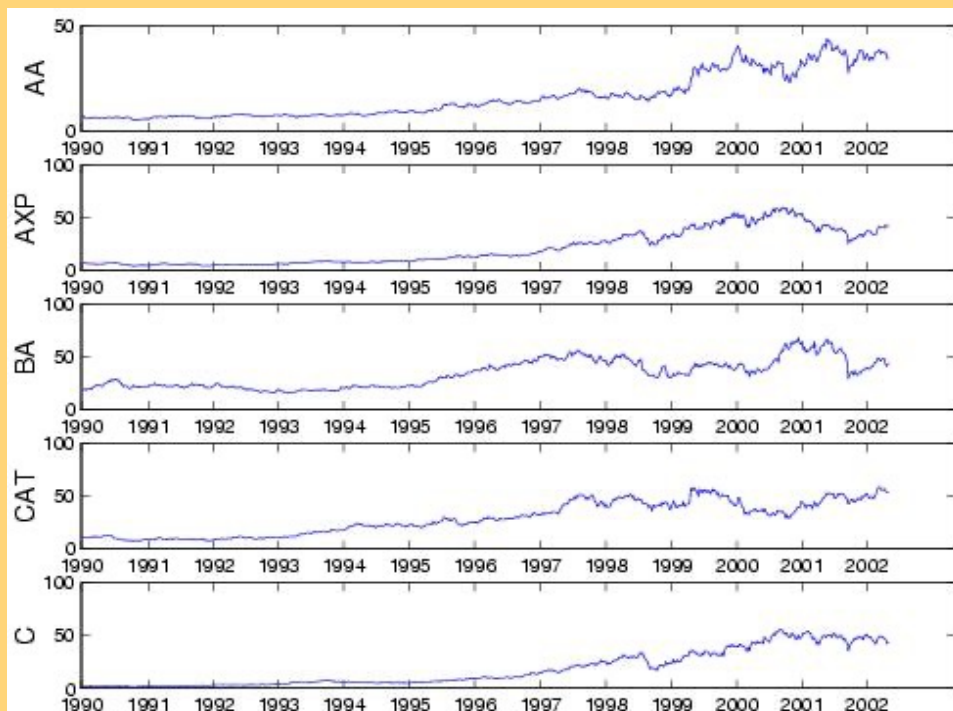




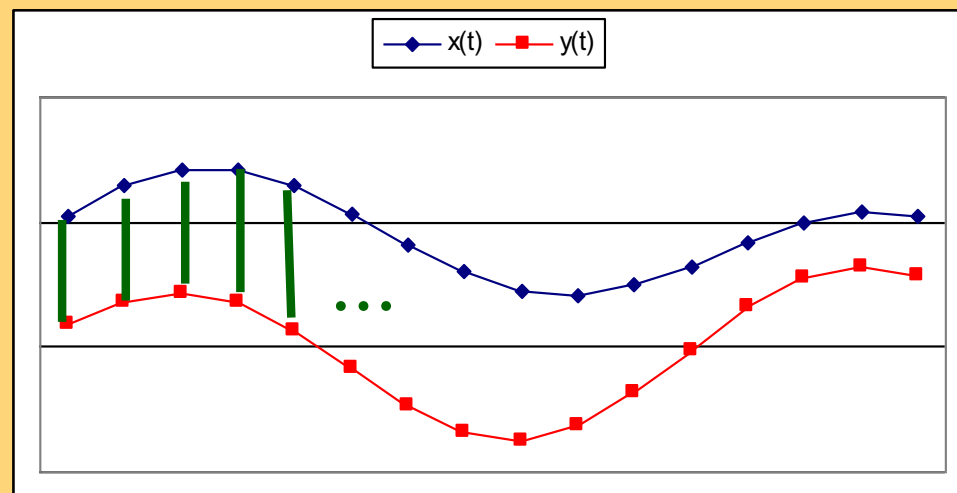
Answer:

Q: How similar are two sequences?

A: Euclidean distance (<-> cosine similarity)



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



Importance of distance functions

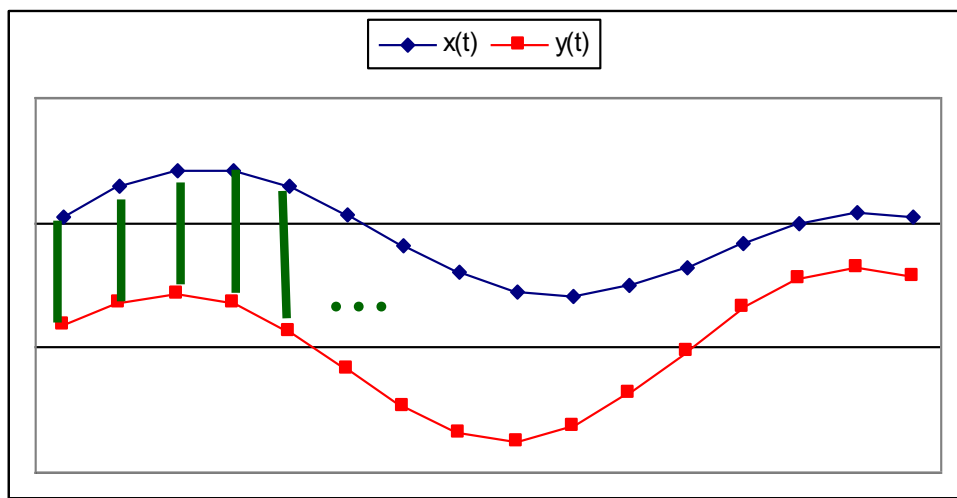
Subtle, but **absolutely necessary**:

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

Two major families

- Euclidean and L_p norms
- Time warping and variations

Euclidean and Lp

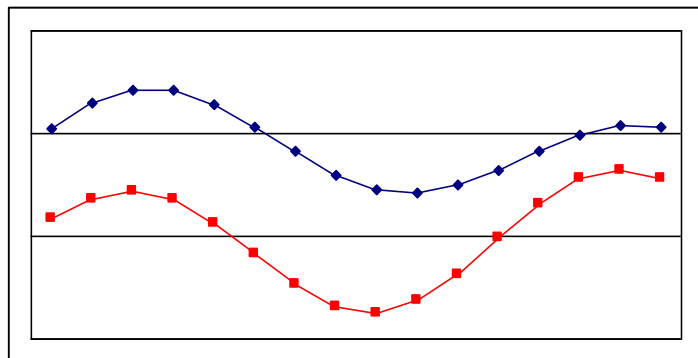


$$D(\vec{x}, \vec{y}) = \sum_{i=1}^n (x_i - y_i)^2$$

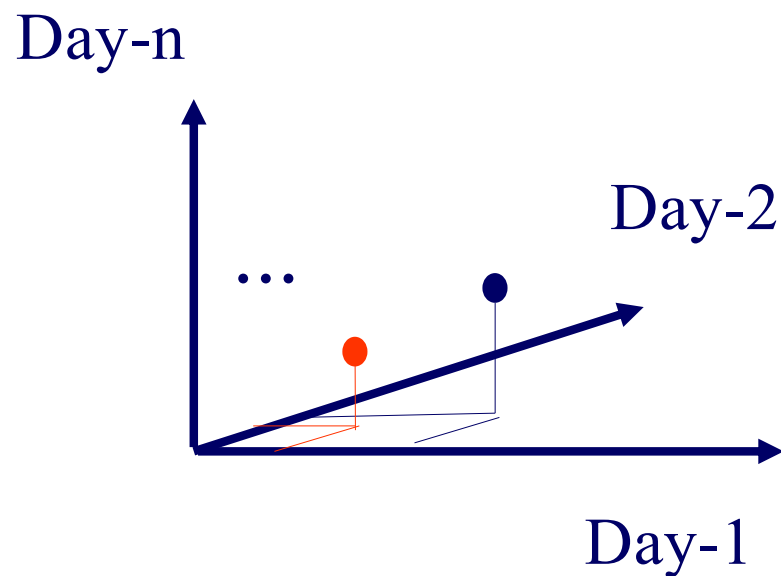
$$L_p(\vec{x}, \vec{y}) = \sum_{i=1}^n |x_i - y_i|^p$$

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

Observation #1



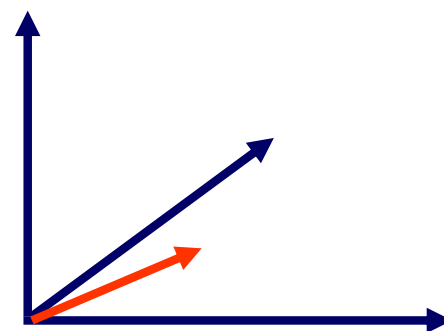
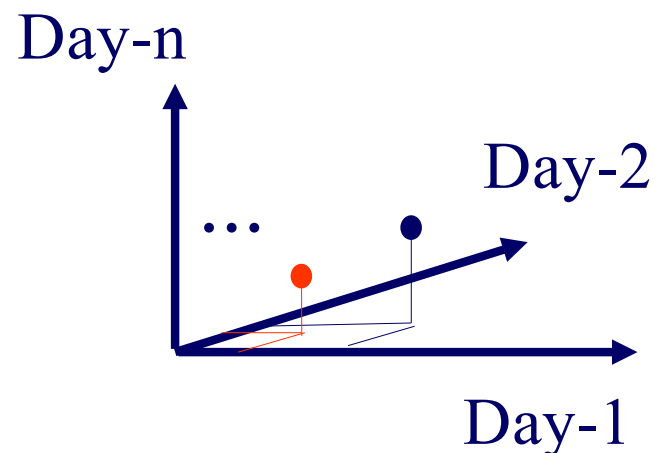
- Time sequence $\rightarrow$ n-d vector



Observation #2

Euclidean distance is closely related to

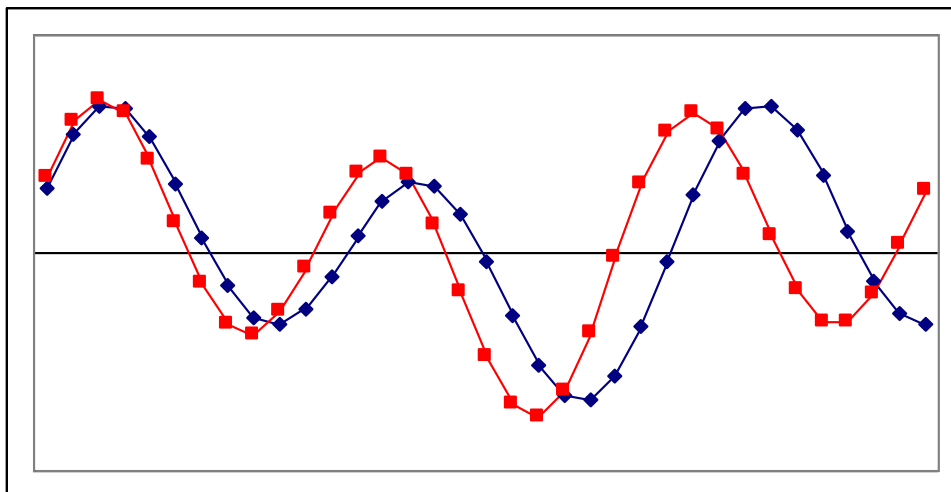
- cosine similarity
- dot product
- ‘cross-correlation’ function



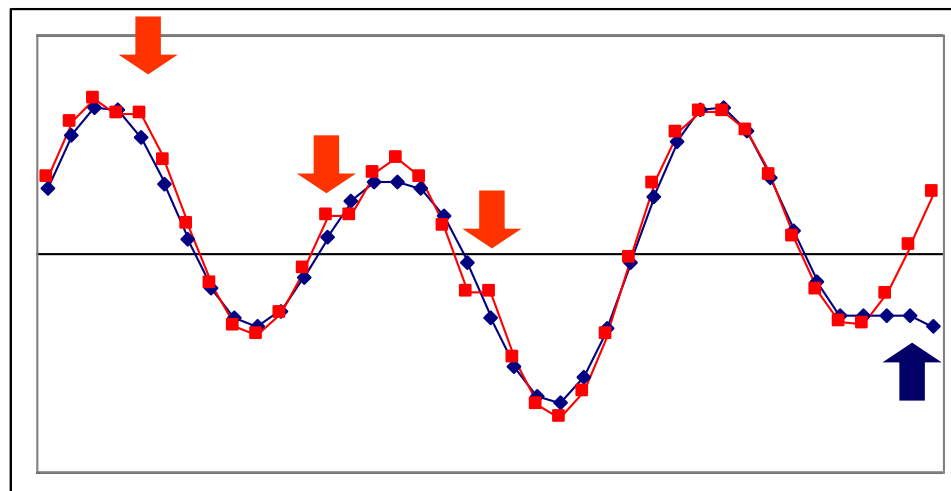
Time Warping

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

Time Warping



‘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



Full-text scanning

- Approximate matching - **string editing distance**:

$$d(\text{'survey'}, \text{'surgery'}) = 2$$

= min # of insertions, deletions,
substitutions to transform the first string
into the second

SURVEY
SURGERY

Time warping

Thus, with no penalty for stutter, for sequences

$$x_1, x_2, \dots, x_i, \quad y_1, y_2, \dots, y_j$$

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

Time warping

VERY SIMILAR to the string-editing distance

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



Full-text scanning

if $s[i] = t[j]$ then

$\text{cost}(i, j) = \text{cost}(i-1, j-1)$

else

$\text{cost}(i, j) = \min ($

$1 + \text{cost}(i, j-1)$ // deletion

$1 + \text{cost}(i-1, j-1)$ // substitution

$1 + \text{cost}(i-1, j)$ // insertion

)

Time warping

VERY SIMILAR to the string-editing distance

Time-warping

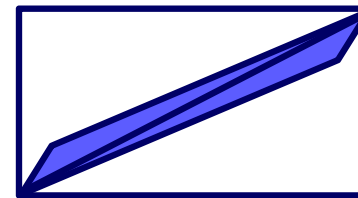
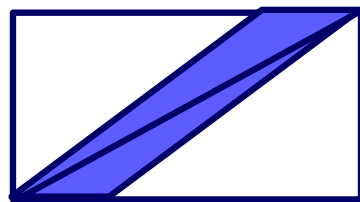
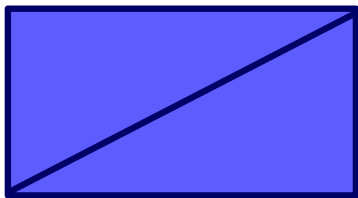
$$D(i, j) = \|x[i] - y[j]\| + \min \begin{cases} D(i-1, j-1) \\ D(i, j-1) \\ D(i-1, j) \end{cases}$$

String editing

$$\text{cost}(i, j) = \min \begin{cases} 1 + \text{cost}(i-1, j-1) // \text{ sub.} \\ 1 + \text{cost}(i, j-1) // \text{ del.} \\ 1 + \text{cost}(i-1, j) // \text{ ins.} \end{cases}$$

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:

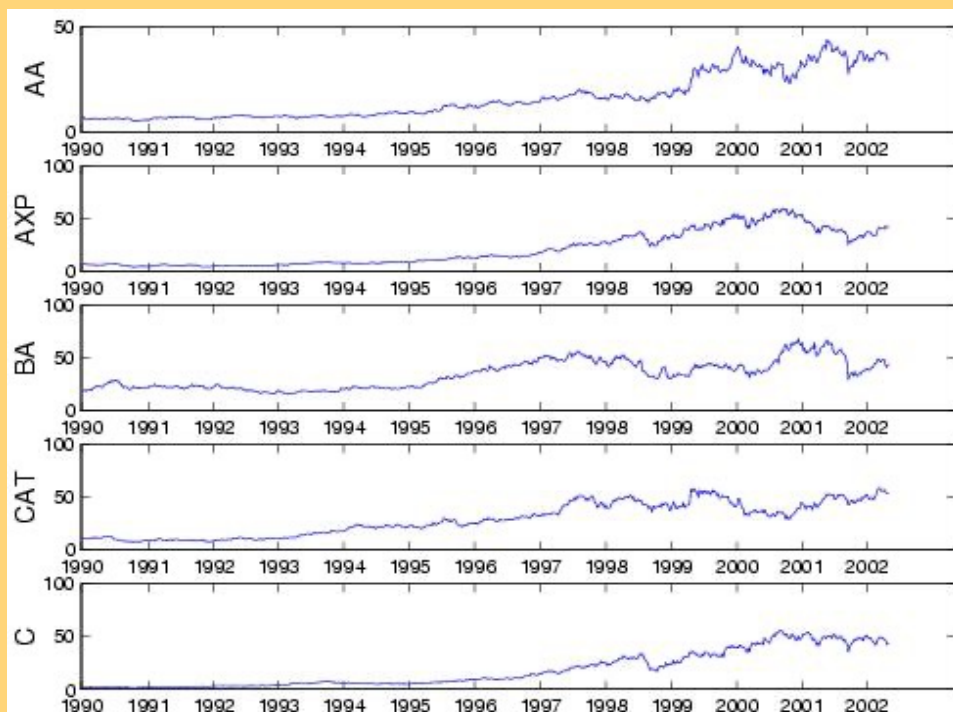
- Euclidean and
- time-warping



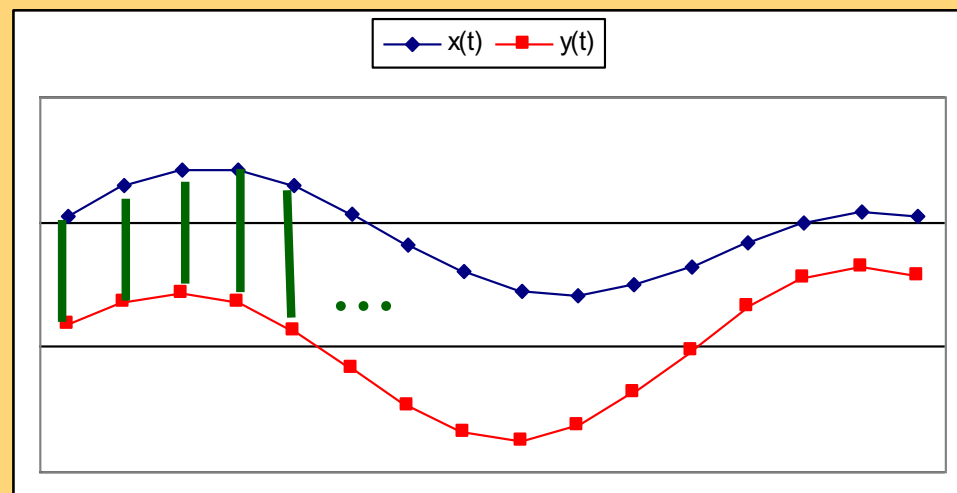
Answer:

Q: How similar are two sequences?

A: Euclidean distance (<-> cosine similarity)



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





Outline

- Motivation
- Similarity search and distance functions
- ➔ • Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Gray box modeling – Lotka Volterra eq's
- Conclusions

Linear Forecasting

Forecasting

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

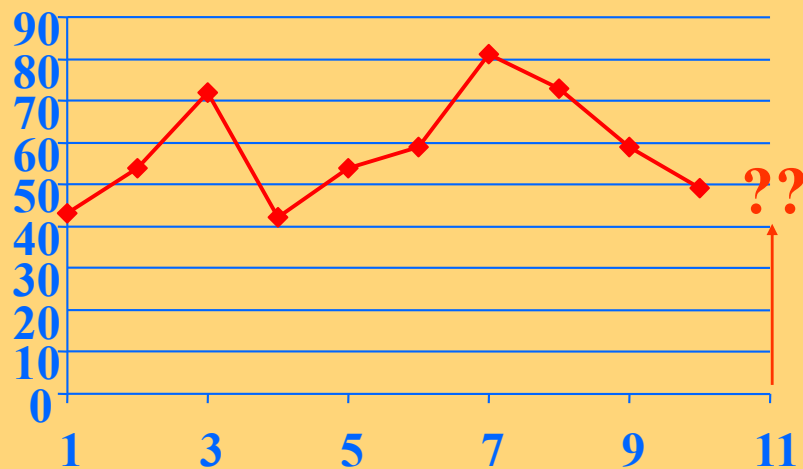
<http://www.hfac.uh.edu/MediaFutures/thoughts.html>





Problem#2: Forecast

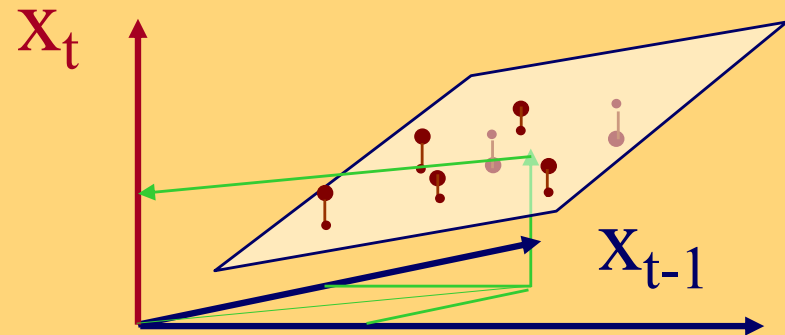
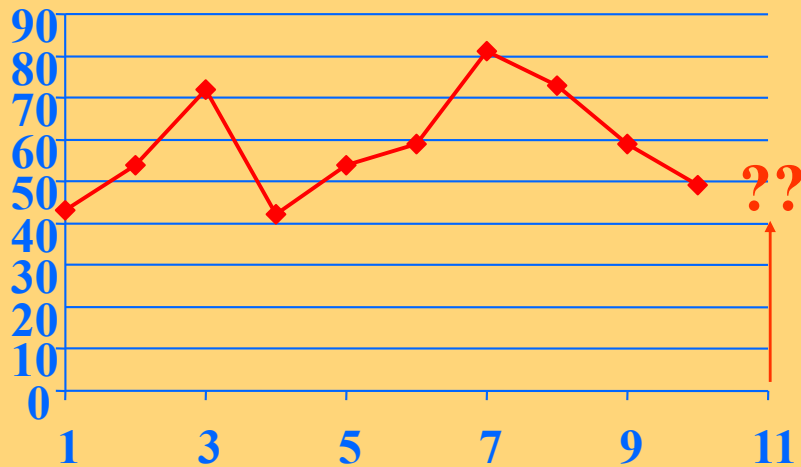
- given $x_{t-1}, x_{t-2}, \dots$,
- Q: forecast x_t





Solution: AR(IMA)

- given $x_{t-1}, x_{t-2}, \dots$,
- Q: forecast x_t
- A: AR(IMA) = Box-Jenkins (< Holt-Winters, Kalman)





Detailed Outline

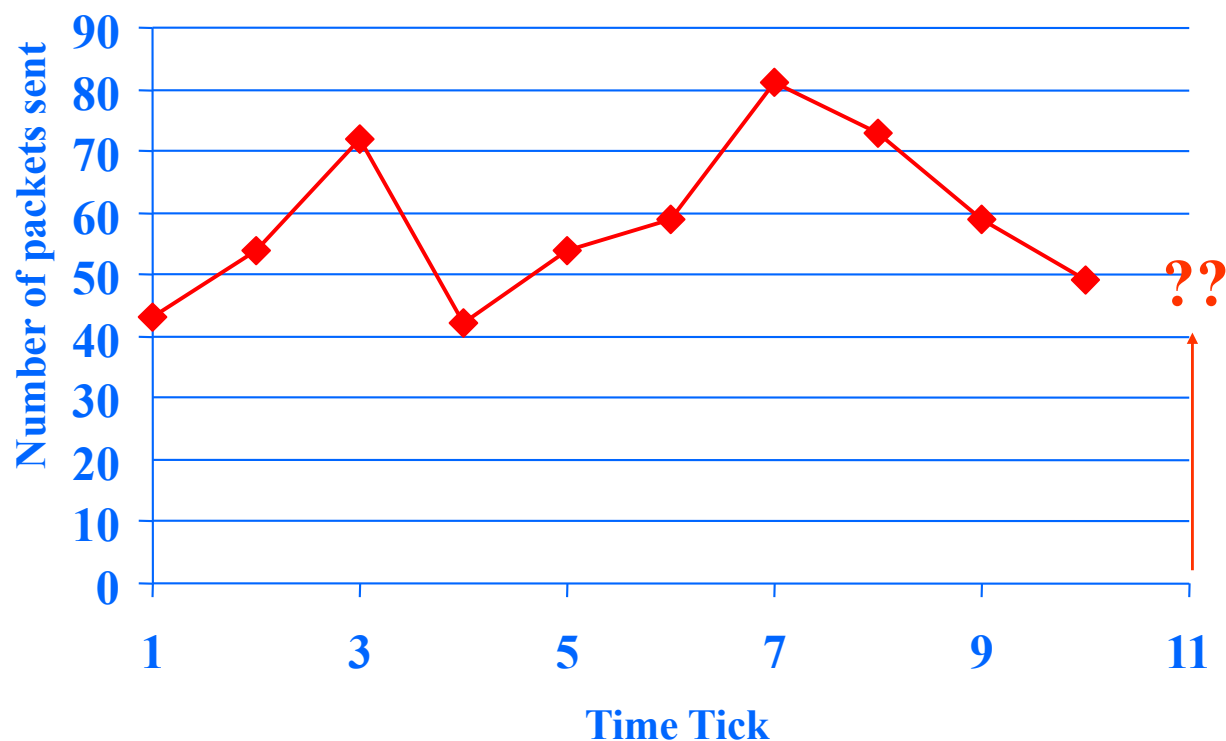
- Motivation
- ...
- Linear Forecasting
 - ➔ – Auto-regression: Least Squares; RLS
 - Co-evolving time sequences
 - Examples
 - Conclusions

Reference

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

Problem#2: Forecast

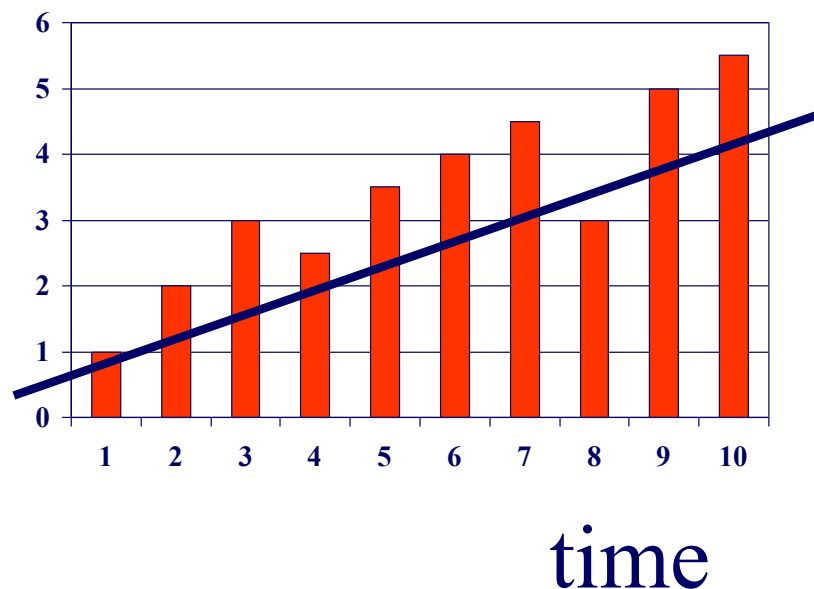
- Example: give $x_{t-1}, x_{t-2}, \dots$, forecast x_t



Forecasting: Preprocessing

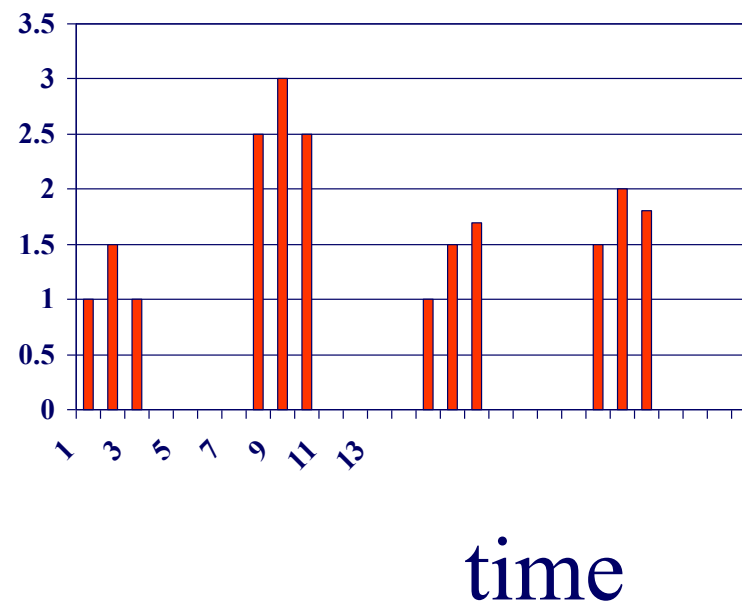
MANUALLY:

remove trends



spot periodicities

7 days



Problem#2: Forecast

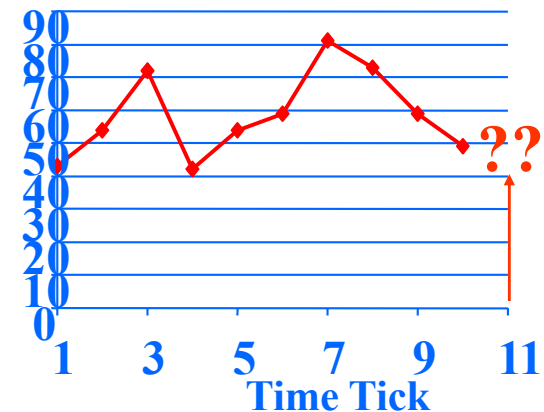
- Solution: try to express

x_t

as a linear function of the past: $x_{t-2}, x_{t-2}, \dots$,
(up to a window of w)

Formally:

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



(Problem: Back-cast; interpolate)

- Solution - interpolate: try to express

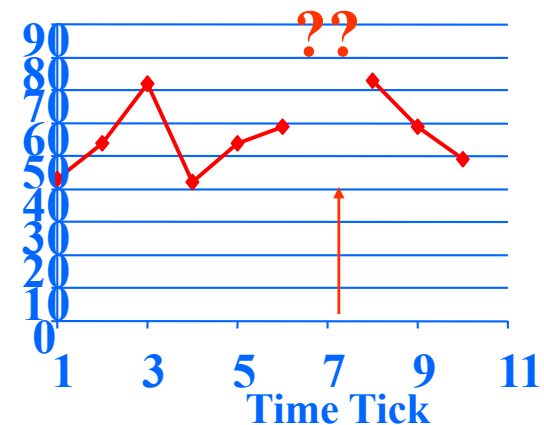
$$x_t$$

as a linear function of the past AND the future:

$$x_{t+1}, x_{t+2}, \dots x_{t+w_{future}}; x_{t-1}, \dots x_{t-w_{past}}$$

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

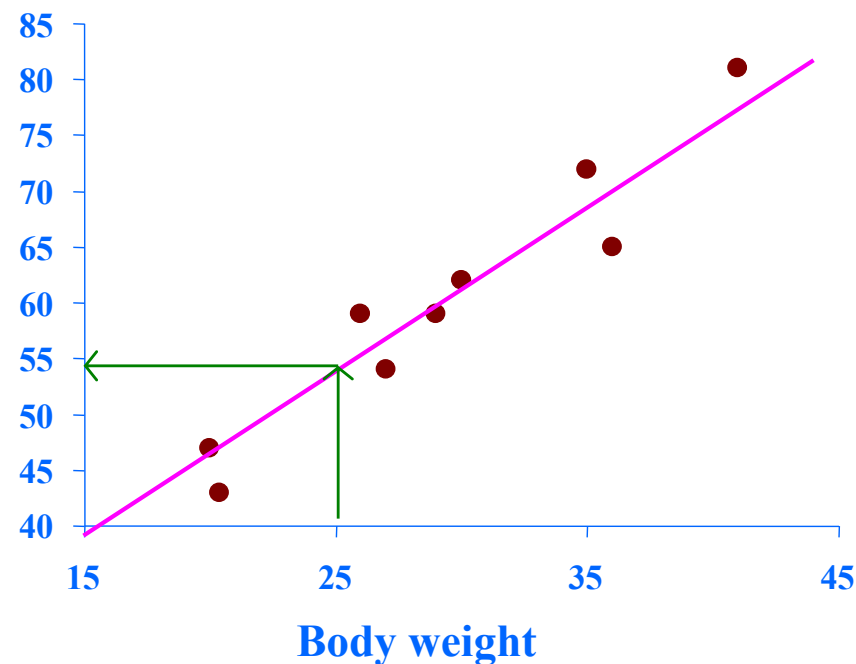
- EXACTLY the same algo's



Linear Regression: idea

<i>patient</i>	<i>weight</i>	<i>height</i>
1	27	43
2	43	54
3	54	72
...	...	...
N	25	??

Body height



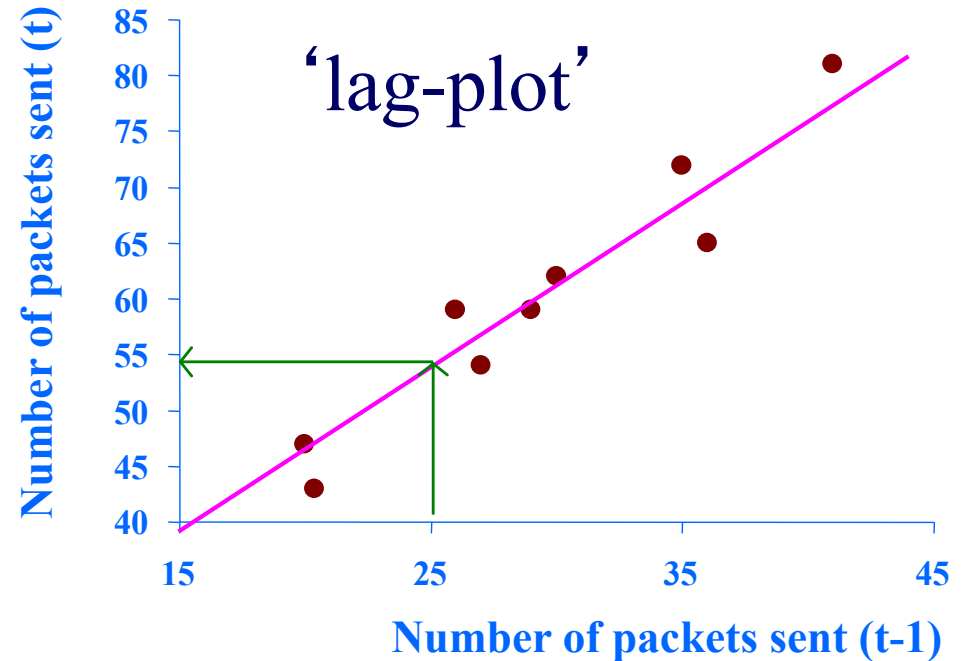
- express what we don't know (= 'dependent variable')
- as a linear function of what we know (= 'indep. variable(s)')

Linear Auto Regression:

<i>Time</i>	<i>Packets Sent(t)</i>
1	43
2	54
3	72
...	...
N	??

Linear Auto Regression:

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



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

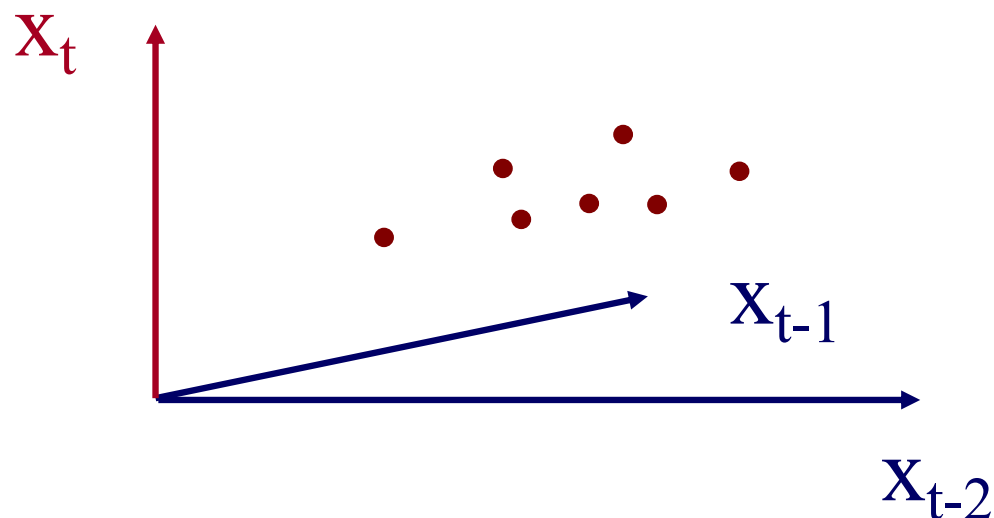


Detailed Outline

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

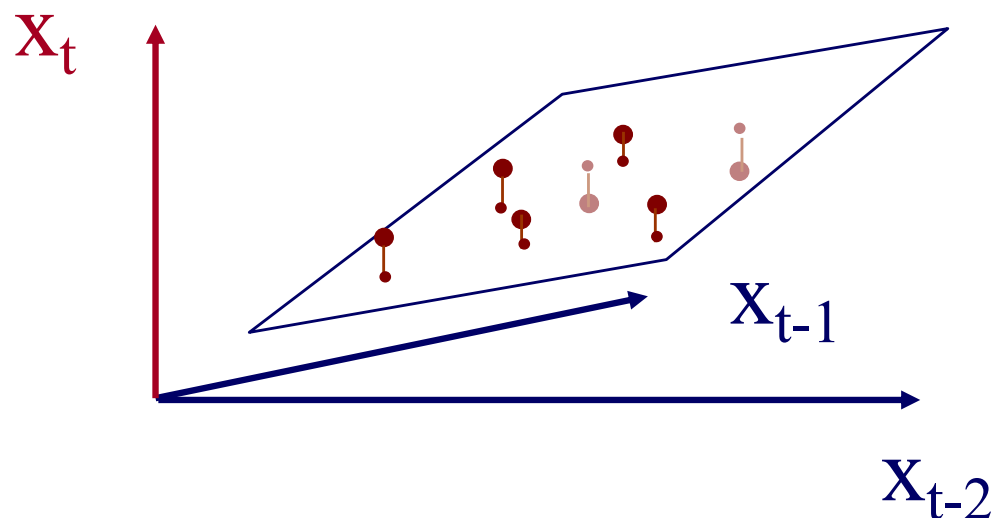
More details:

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



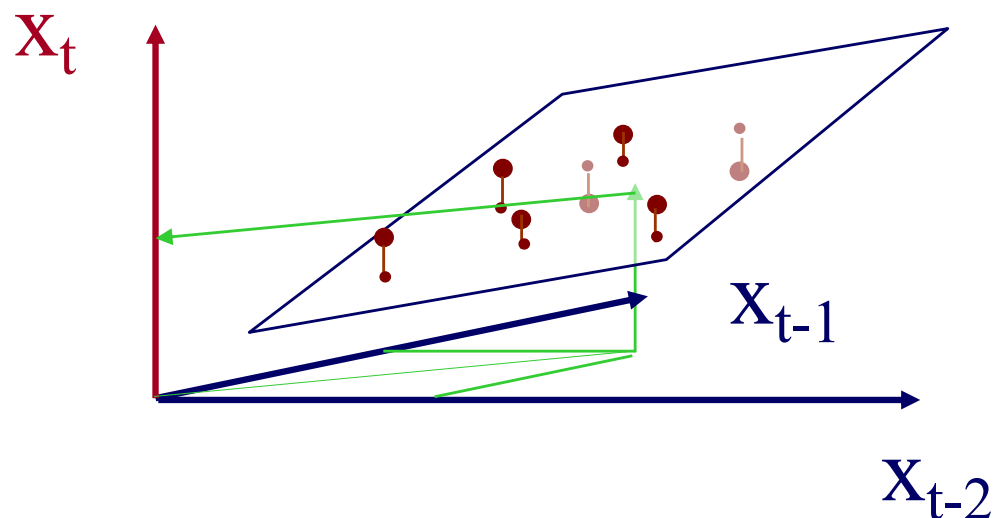
More details:

- 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!)



More details:

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

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

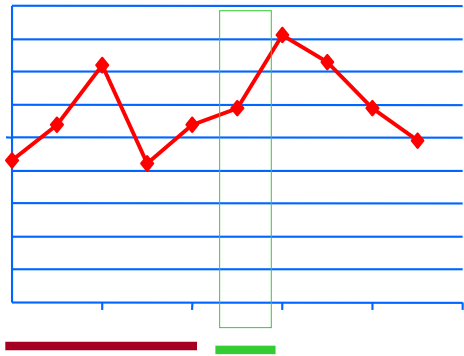
- OVER-CONSTRAINED
 - $\mathbf{a}$ is the vector of the regression coefficients
 - $\mathbf{X}$ has the N values of the w indep. variables
 - $\mathbf{y}$ has the N values of the dependent variable

More details:

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

Ind-var1

Ind-var-w



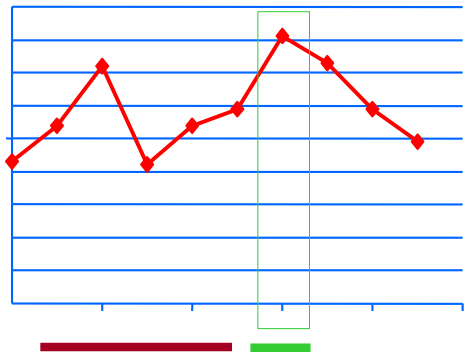
time

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

More details:

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

Ind-var1 Ind-var-w



time

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

More details

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

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

- (Moore-Penrose pseudo-inverse)
- $\mathbf{a}$ is the vector that minimizes the RMSE from $\mathbf{y}$
- <identical math with ‘query feedbacks’>



More details

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

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

Identical to earlier formula (proof?)

$$\mathbf{a} = \mathbf{V} \times \mathbf{\Lambda}^{(-1)} \times \mathbf{U}^T \times \mathbf{y}$$

Where

$$\mathbf{X} = \mathbf{U} \times \mathbf{\Lambda} \times \mathbf{V}^T$$

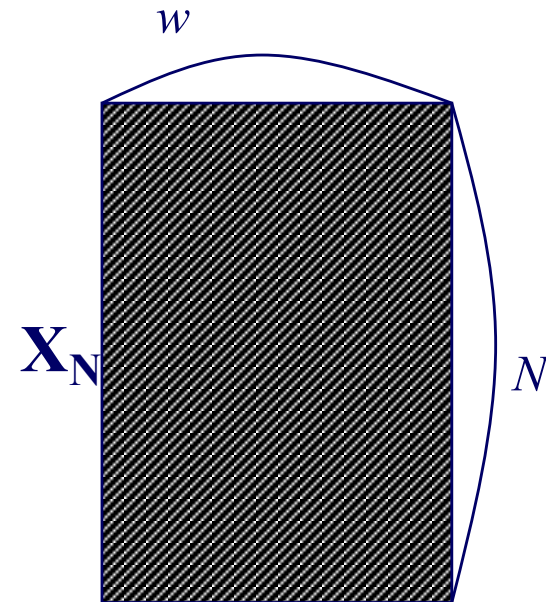
More details

- Straightforward solution:

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

$\mathbf{a}$: Regression Coeff. Vector

$\mathbf{X}$: Sample Matrix



- Observations:
 - Sample matrix $\mathbf{X}$ grows over time
 - needs matrix inversion
 - $\mathbf{O}(N \times w^2)$ computation
 - $\mathbf{O}(N \times w)$ storage

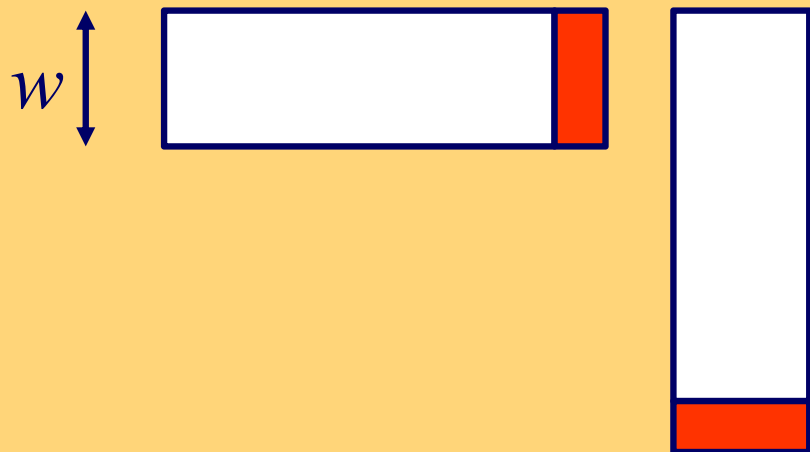
Even more details

- 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?!)



Sub-Problem: matrix inversion

- How to invert: $(X^T X)^{-1}$
- Incrementally
- WITHOUT inverting (!)

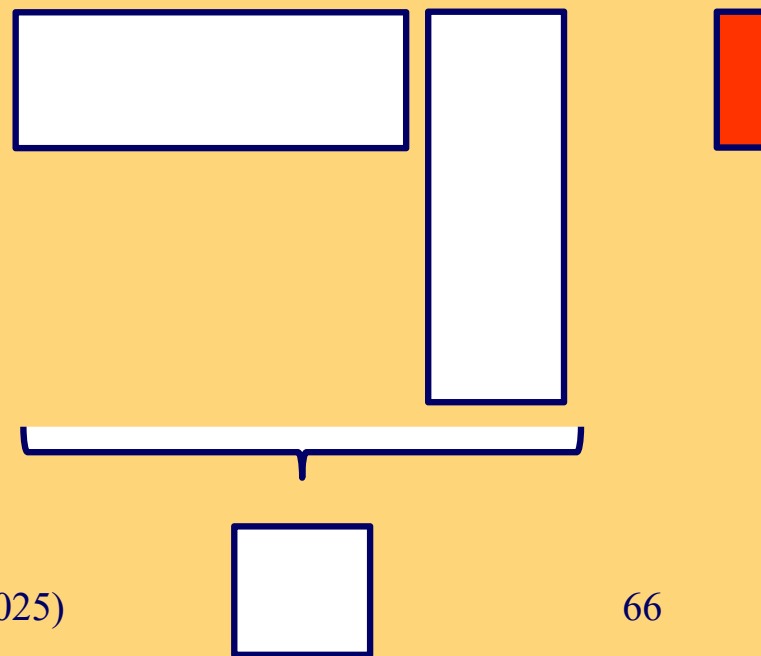




Sub-Problem: matrix inversion

- How to invert: $(\mathbf{X}^T \mathbf{X})^{-1}$
- Incrementally
- WITHOUT inverting (!)

A: keep a $w \times w$ matrix,
& update it



Even more details

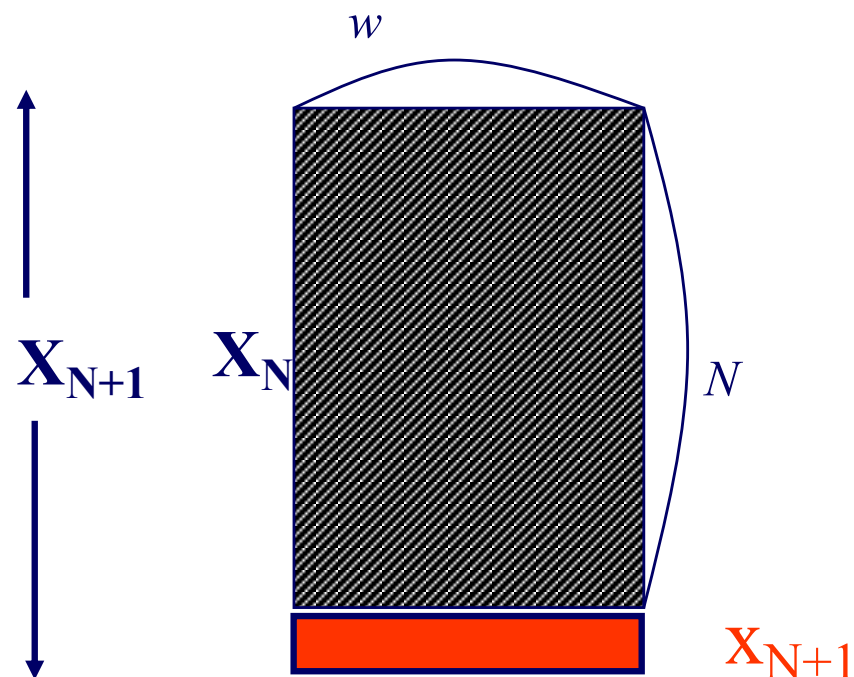
- Q3: Can we estimate $\mathbf{a}$ incrementally?
- A3: Yes, with the brilliant, classic method of ‘Recursive Least Squares’ (RLS) (see, e.g., [Yi+00], for details).
- We can do the matrix inversion, WITHOUT inversion! (How is that possible?!)
- A: our matrix has special form: $(\mathbf{X}^T \mathbf{X})$

Intuition:

- How to compute the average of $x_1, x_2, \dots, x_n$
 - Incrementally
 - Solution: ‘sufficient statistics’
 - Count ‘ k ’ so far : $k += 1$
 - Sum ‘ s_k ’ so far : $s_{k+1} \leftarrow s_k + x_{k+1}$
- Return $s_k \div k$

More details

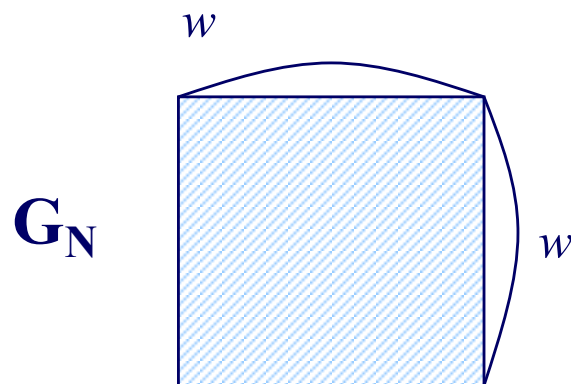
At the $N+1$ time tick:



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

More details

- Let $\mathbf{G}_N = (\mathbf{X}_N^T \times \mathbf{X}_N)^{-1}$ ('gain matrix')
- $\mathbf{G}_{N+1}$ can be computed recursively from $\mathbf{G}_N$



EVEN more details:

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

 1 x w row vector

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

Let's elaborate
(VERY IMPORTANT, VERY VALUABLE!)

EVEN more details:

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

EVEN more details:

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

$[w \times 1]$ $[(N+1) \times w]$ $[(N+1) \times 1]$
 $[w \times (N+1)]$ $[w \times (N+1)]$

EVEN more details:

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

$[(N+1) \times w]$

$[w \times (N+1)]$

EVEN more details:

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

‘gain
matrix’

$$G_{N+1} \equiv [X_{N+1}^T \times X_{N+1}]^{-1}$$

1 x w row vector

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


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

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:

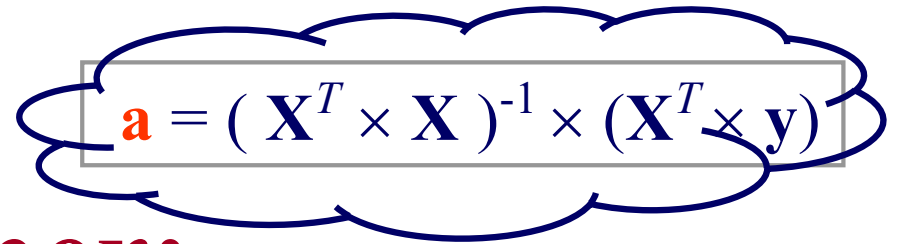
Diagram illustrating a 1D CNN architecture with 3 layers:

- Layer 1 (Input): 3 nodes, each labeled $w \times w$.
- Layer 2: 2 nodes, labeled $w \times w$ and $w \times 1$.
- Layer 3 (Output): 1 node, labeled 1×1 .

Arrows indicate the flow of information from Layer 1 to Layer 2, and from Layer 2 to Layer 3.

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

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



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

Altogether:

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

$$G_{N+1} \equiv [X_{N+1}^T \times X_{N+1}]^{-1}$$

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

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

Altogether:

$$G_0 \equiv \delta I \quad \text{IMPORTANT!}$$

where

I : $w \times w$ identity matrix

δ : a large positive number (say, 10^4)

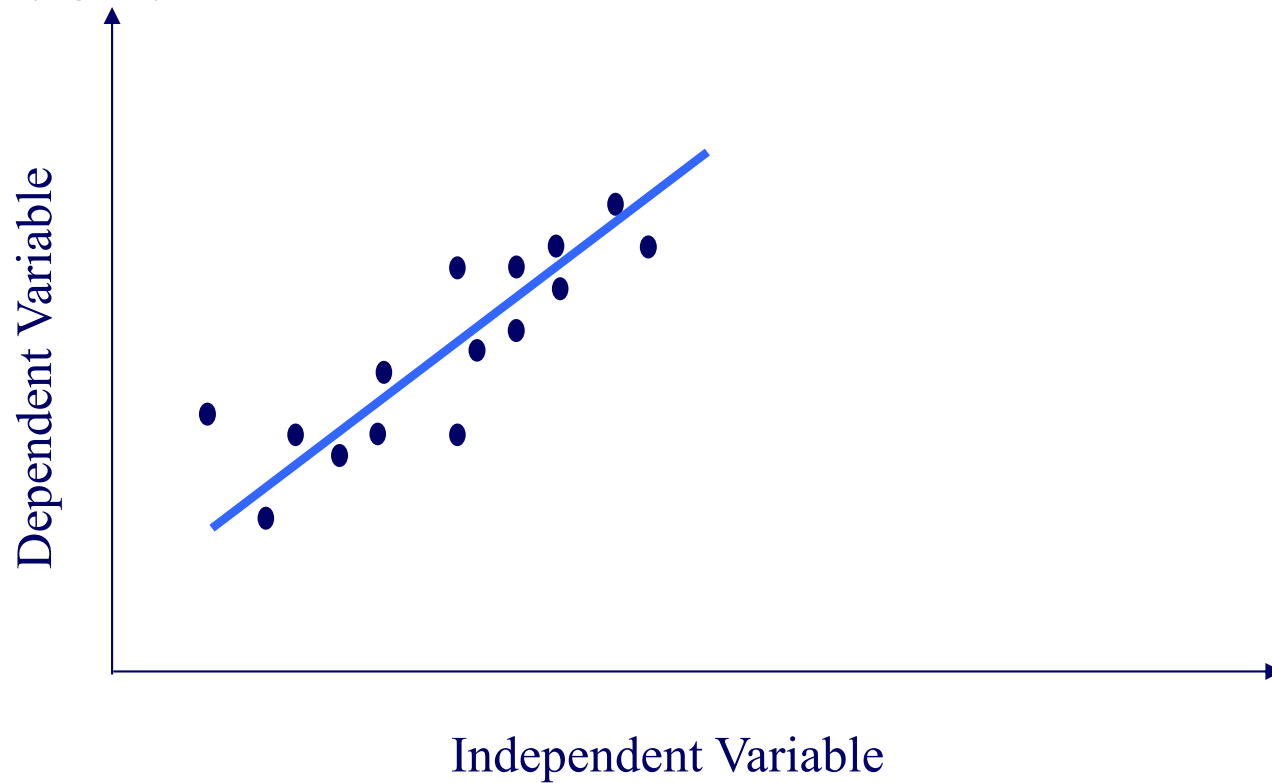
Comparison:

- **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 $O(1 \times w^2)$
 - **no matrix inversion**

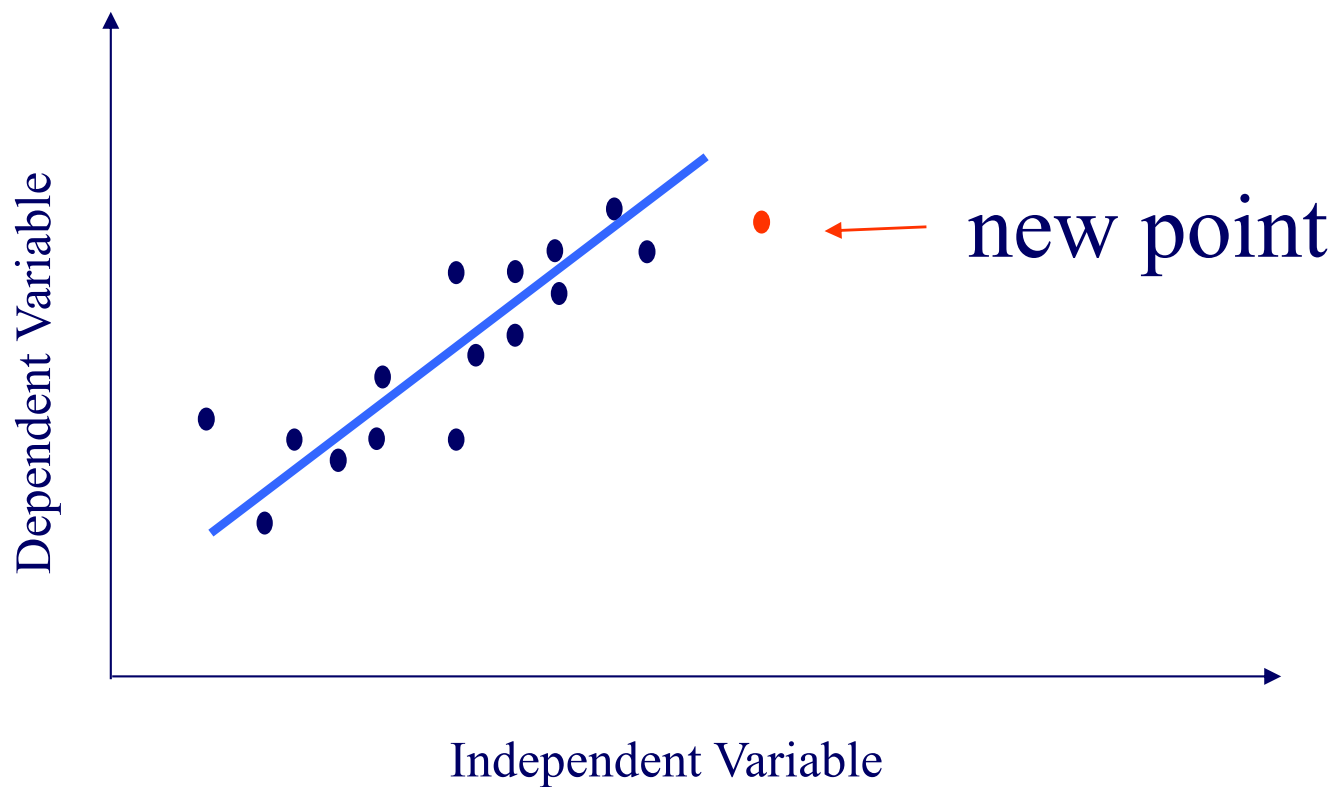
$$N = 10^6, \quad w = 1-100$$

Pictorially:

- Given:

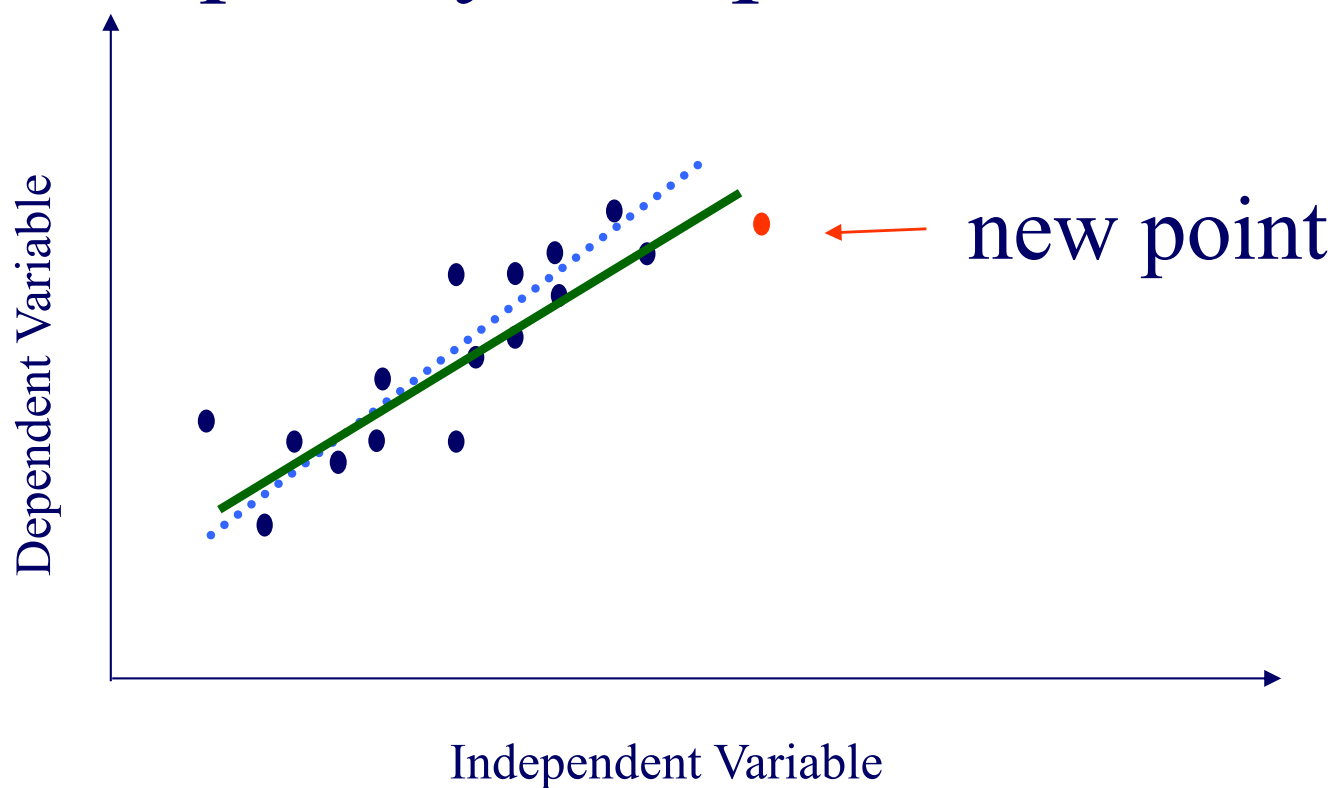


Pictorially:



Pictorially:

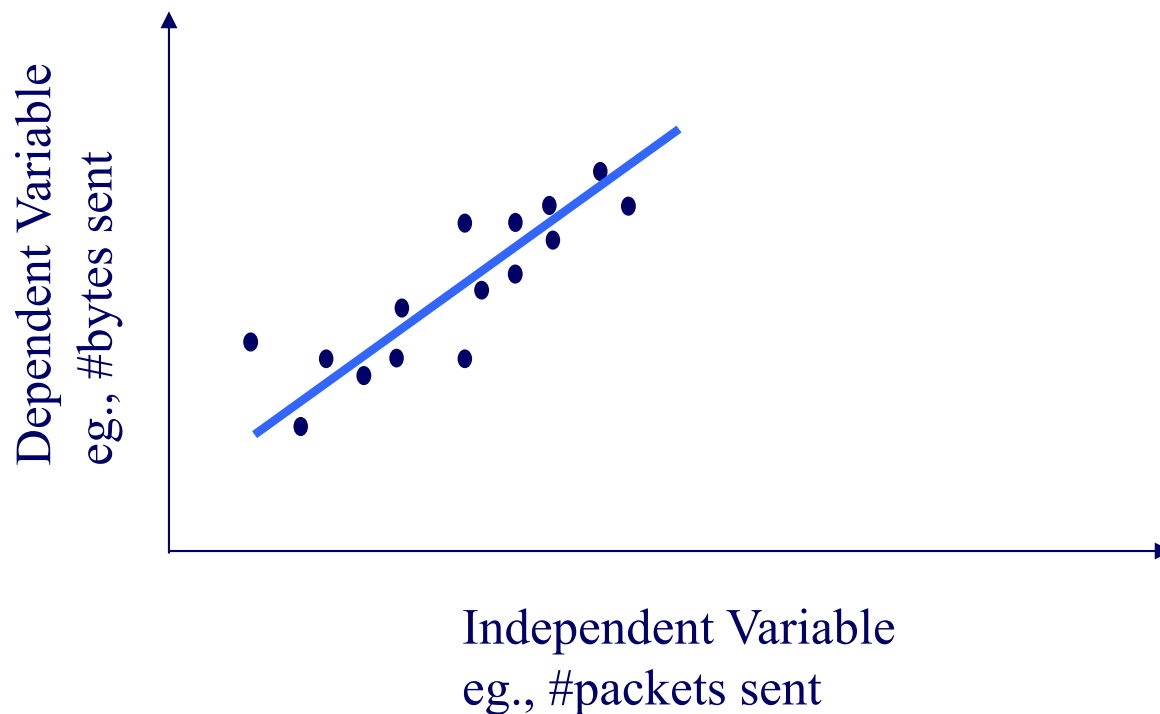
RLS: quickly compute new best fit



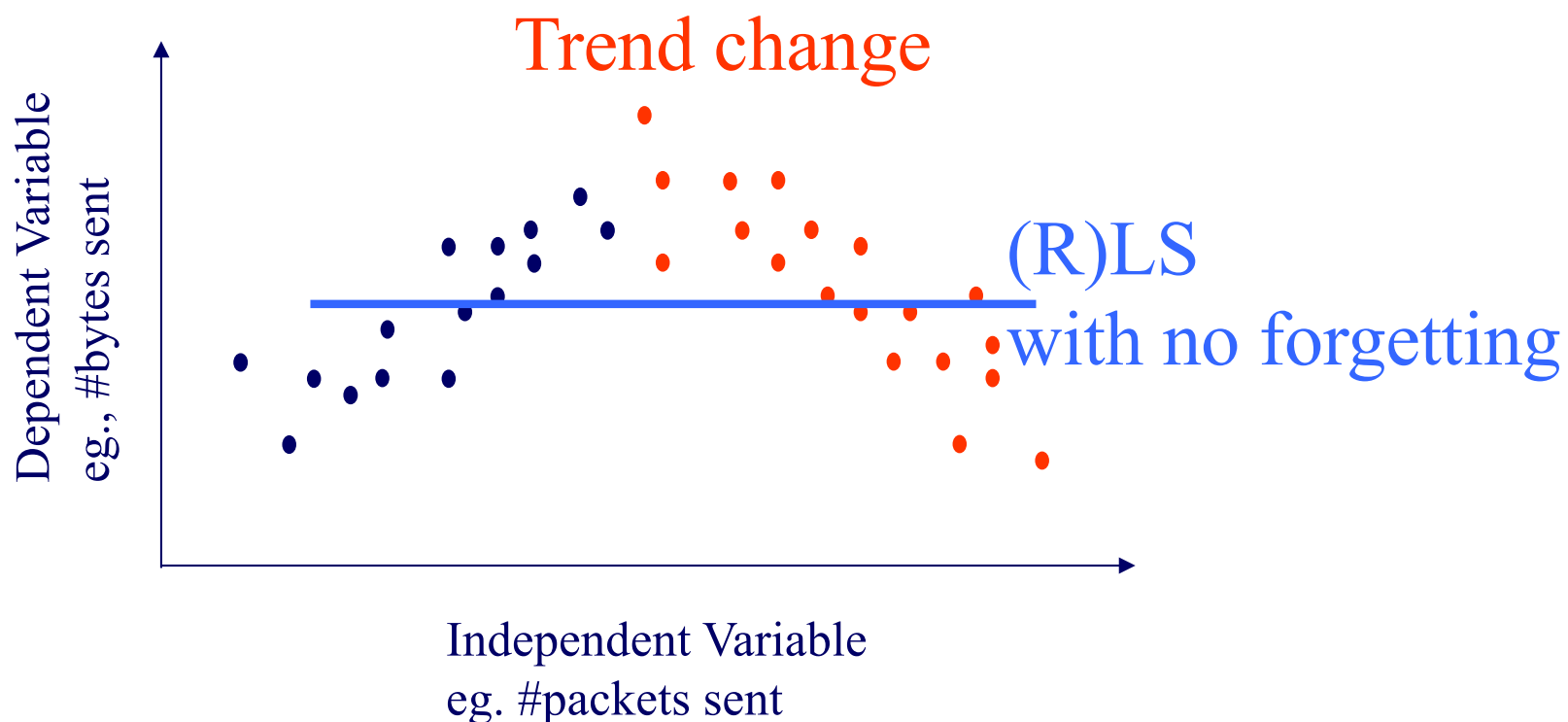
Even more details

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

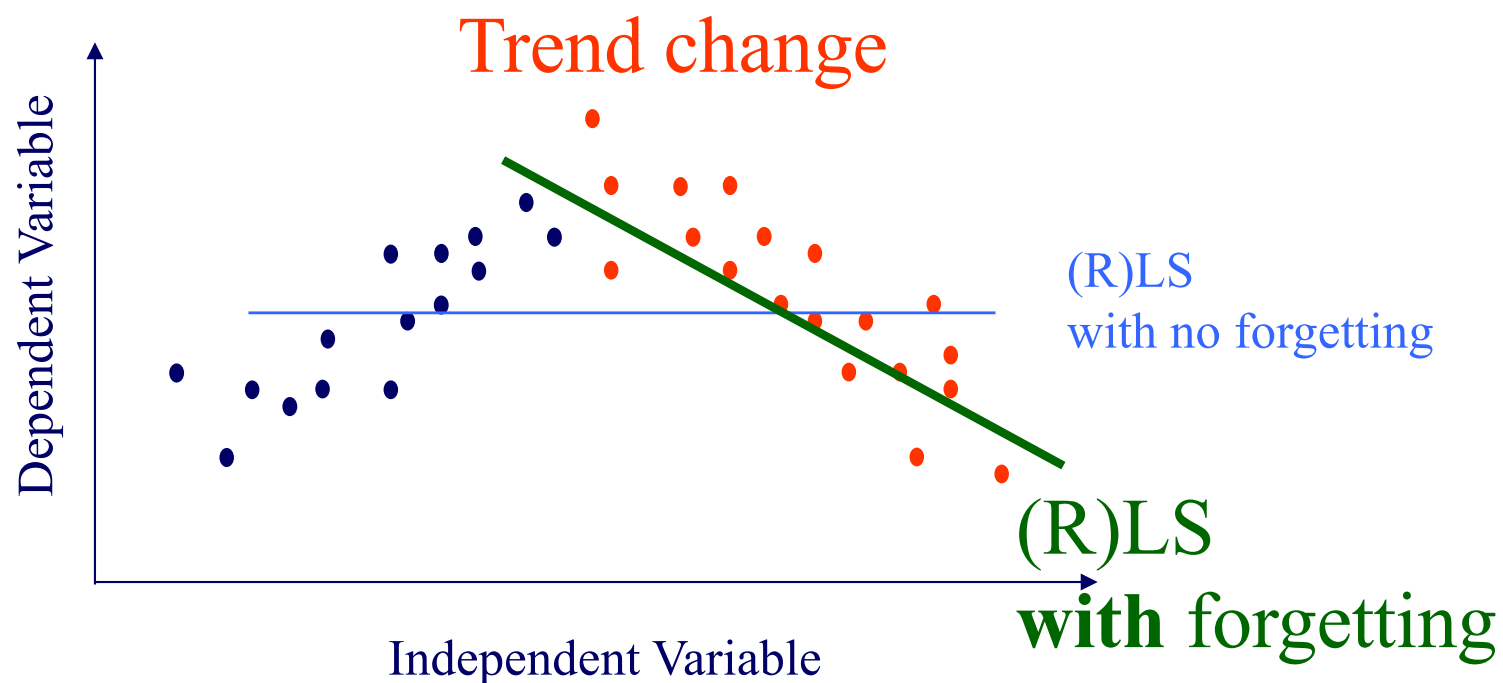
Adaptability - ‘forgetting’



Adaptability - 'forgetting'



Adaptability - 'forgetting'



- RLS: can *trivially* handle 'forgetting' (see [Yi+,2000])

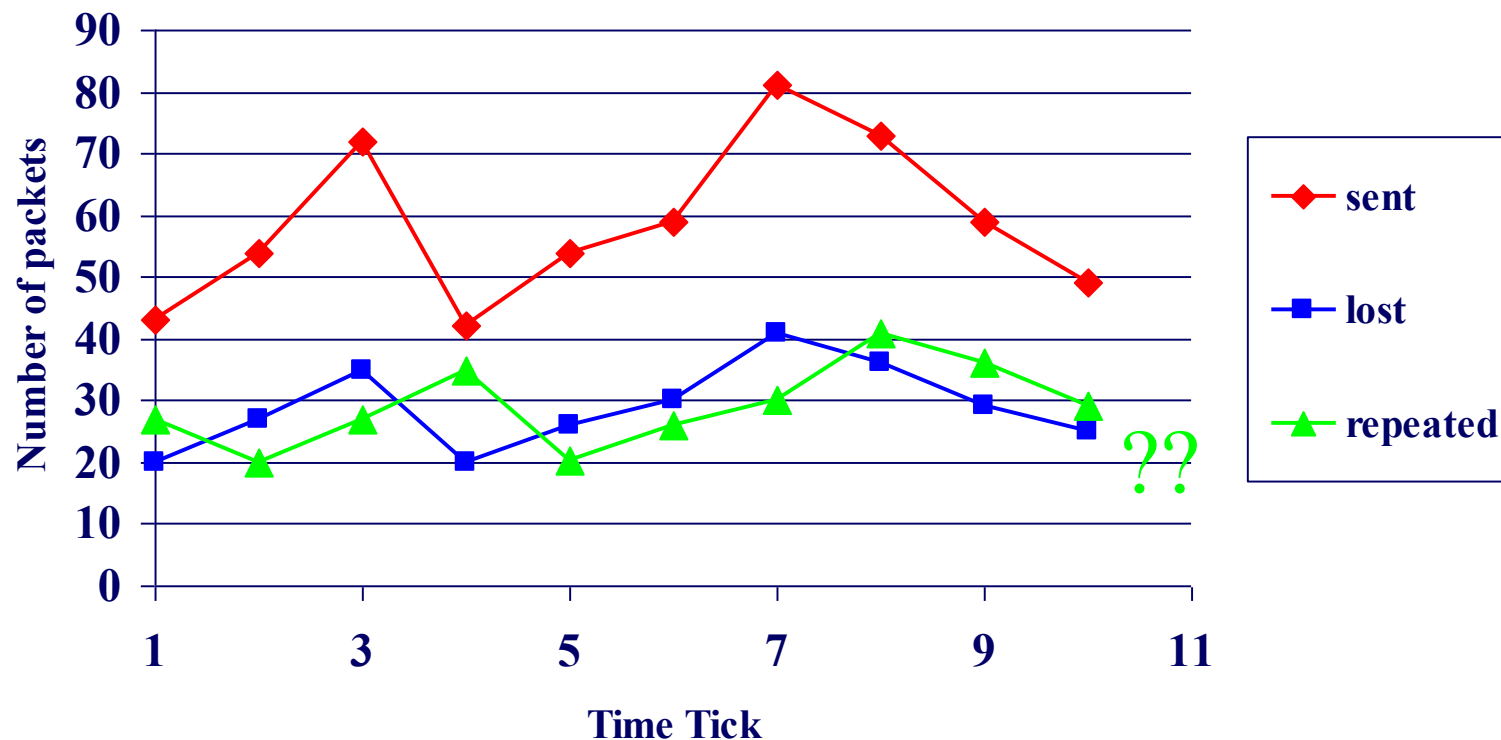


Detailed Outline

- 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])

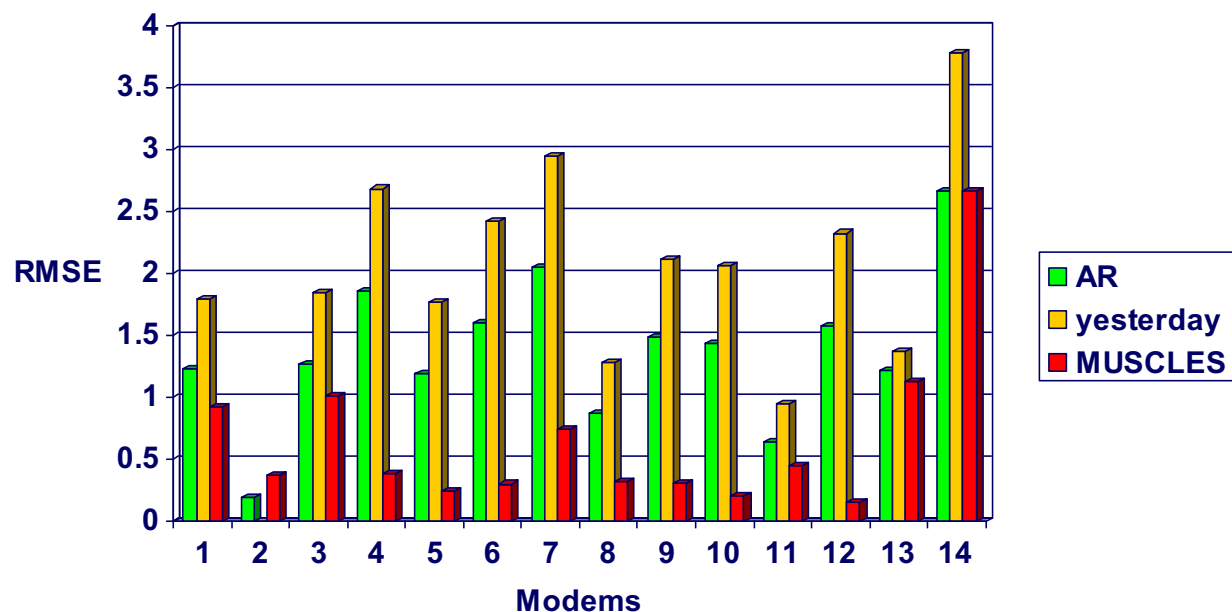
Forecasting - Outline

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

Examples - Experiments

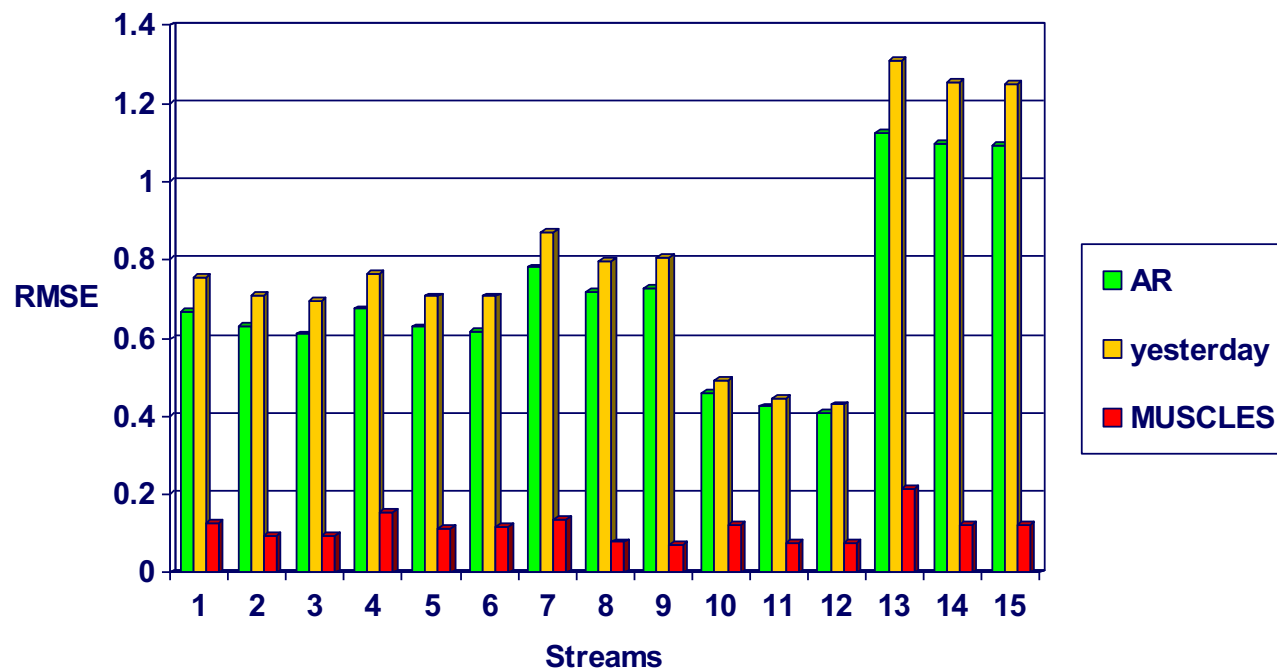
- Datasets
 - Modem pool traffic (14 modems, 1500 time-ticks; #packets per time unit)
 - AT&T WorldNet internet usage (several data streams; 980 time-ticks)
- Measures of success
 - Accuracy : Root Mean Square Error (RMSE)

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

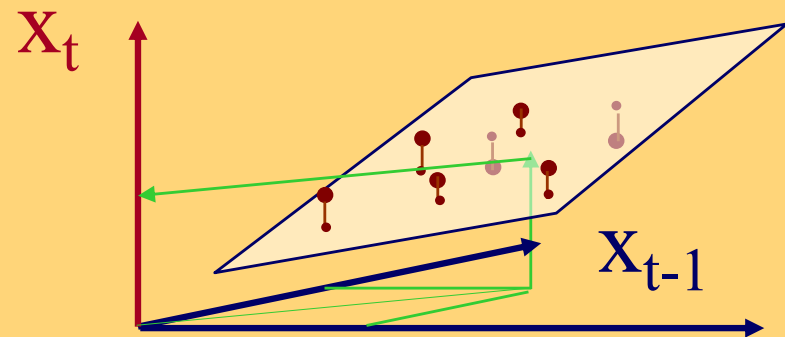
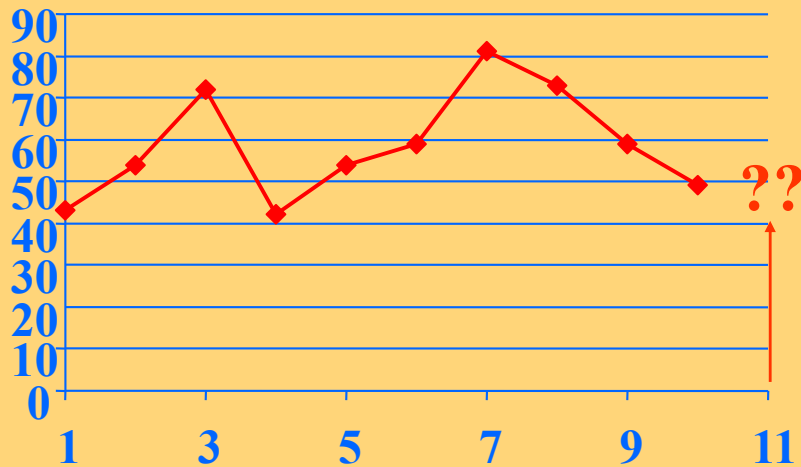
Conclusions - Practitioner's guide

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



Solution: AR(IMA)

- given $x_{t-1}, x_{t-2}, \dots$,
- Q: forecast x_t
- A: AR(IMA) = Box-Jenkins (< Holt-Winters, Kalman)



Resources: software and urls

- free-ware: 'R' for stat. analysis
(clone of Splus)
<http://cran.r-project.org/>
- python script for RLS
<http://www.cs.cmu.edu/~christos/SRC/rls-all.tar>

Books

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

Additional Reading

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



Outline

- Motivation
- Similarity search and distance functions
- Linear Forecasting
- ➔ • Bursty traffic - fractals
- Non-linear forecasting
- Gray box modeling – Lotka Volterra eq's
- Conclusions

Bursty Traffic & fractals



Detailed Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals



- Problem
- Main idea (80/20, Hurst exponent)
- Results

Reference:

[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.

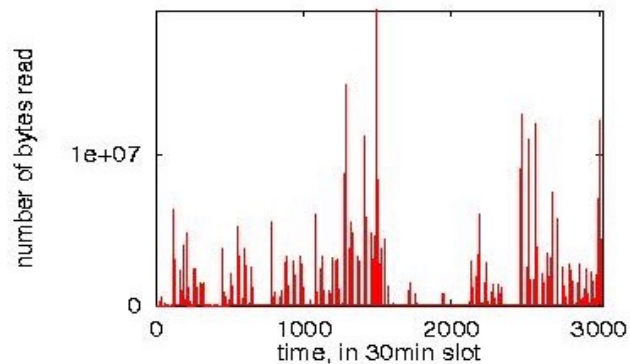
Full thesis: CMU-CS-05-185

Performance Modeling of Storage Devices using Machine Learning Mengzhi Wang, Ph.D. Thesis
[Abstract](#), [.ps.gz](#), [.pdf](#)



Bursty traffic

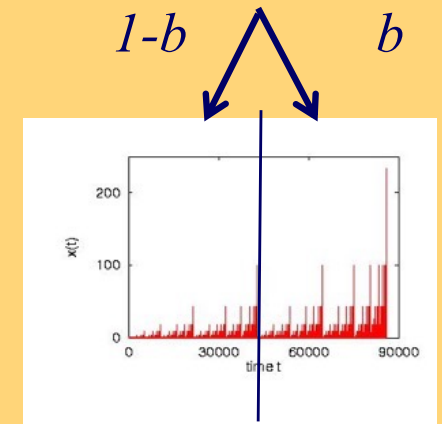
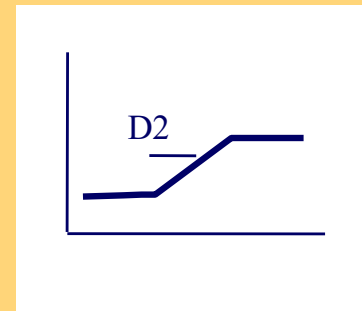
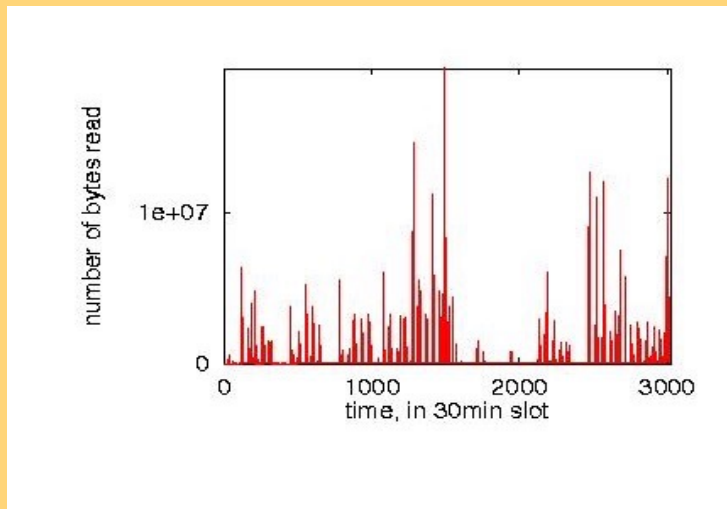
- Q: Any pattern?





Bursty traffic

- Q: Any pattern?
- A: fractal dimension (' b ' model e.g. 80/20)

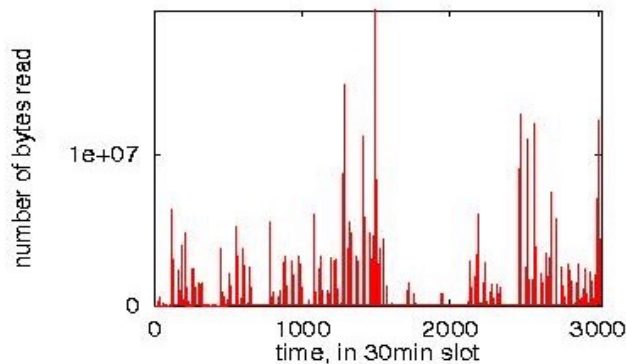


Recall: Problem #1:

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

Find: patterns, periodicities, and/or compress

#bytes



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

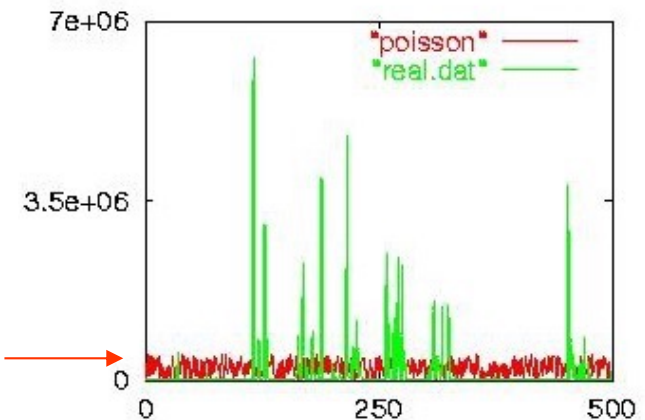
time

Problem #1

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

Poisson →

bytes



time

Motivation

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

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



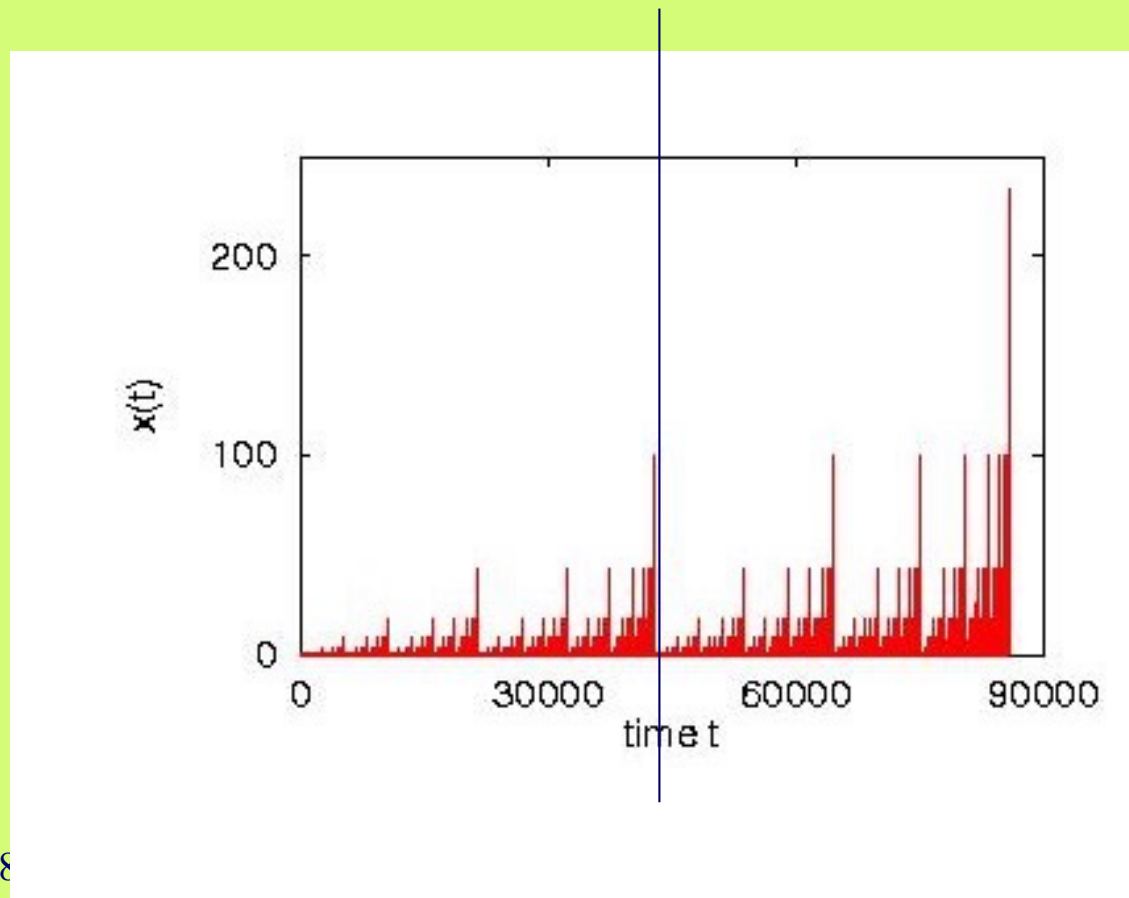
Approach

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



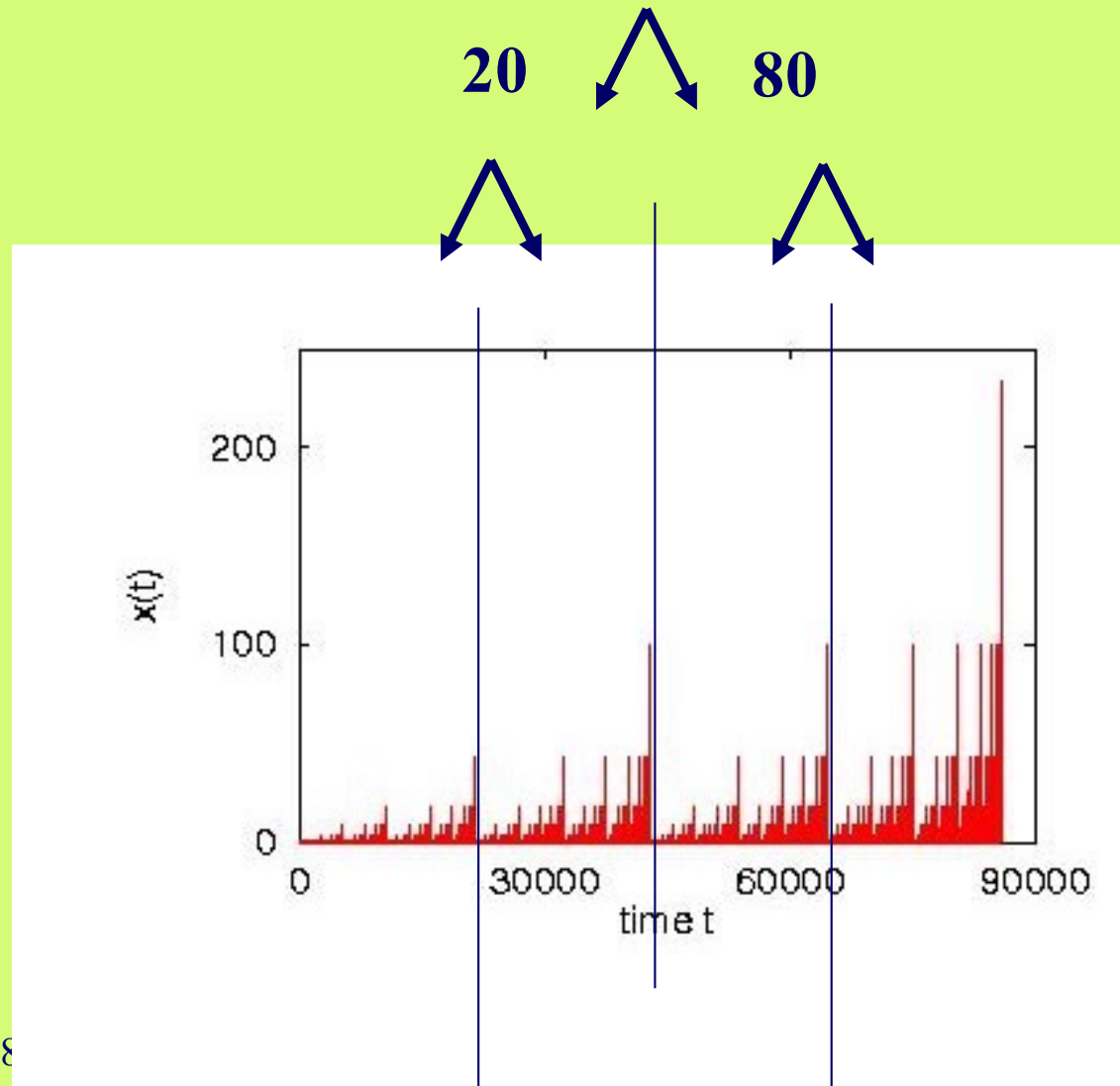
Binary multifractals

20  80





Binary multifractals



Could you use IFS?

To generate such traffic?

Could you use IFS?

To generate such traffic?

A: Yes – which transformations?

Could you use IFS?

To generate such traffic?

A: Yes – which transformations?

A:

$$x' = x / 2 \quad (p = 0.2)$$

$$x' = x / 2 + 0.5 \quad (p = 0.8)$$

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’)
 - Fractal dimension (D2)
 - Or D1 (‘entropy plot’ [Wang+02])

Fractal dimension

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

Dim = 1



Dim=0



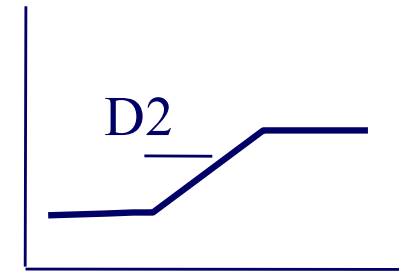
$0 < \text{Dim} < 1$



Estimating 'b'

- **Exercise:** Show that

Log (#pairs(<r))

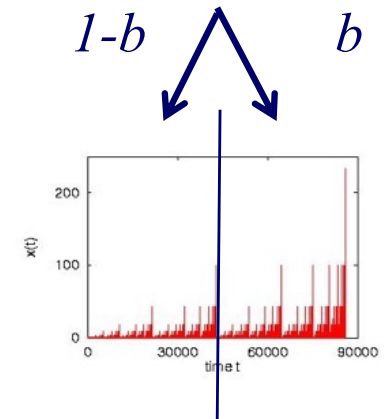


log (r)

$$D_2 = -\log_2 (b^2 + (1-b)^2)$$

Sanity checks:

- $b = 1.0$ $D_2 = ??$
- $b = 0.5$ $D_2 = ??$



(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



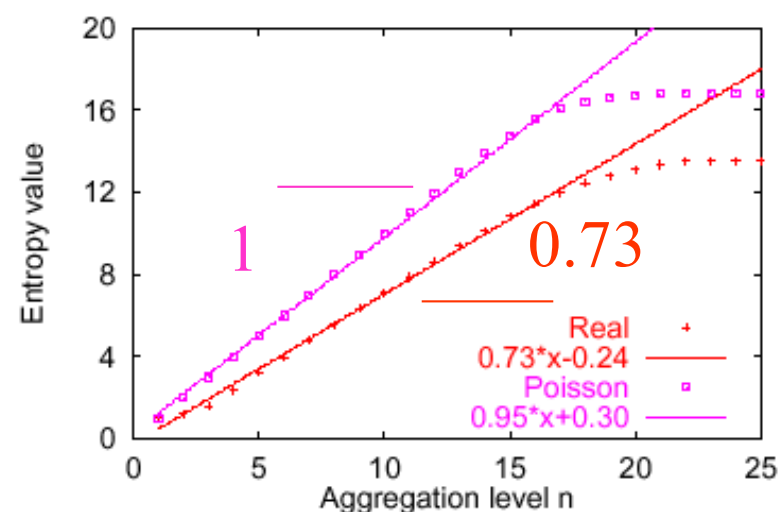
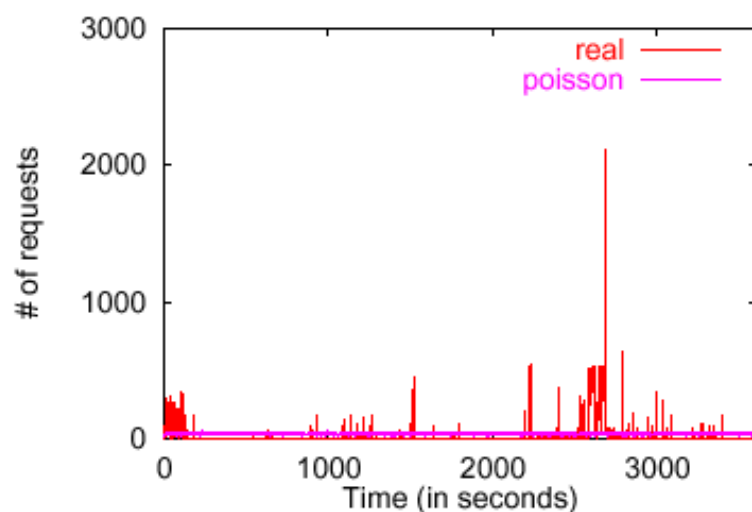
Cantor dust



Dimensionality?
(no length; infinite # points!)
Answer: $\log 2 / \log 3 = 0.6$

Some plots:

- Poisson vs real



Poisson: slope = ~ 1 $\rightarrow$ uniformly distributed

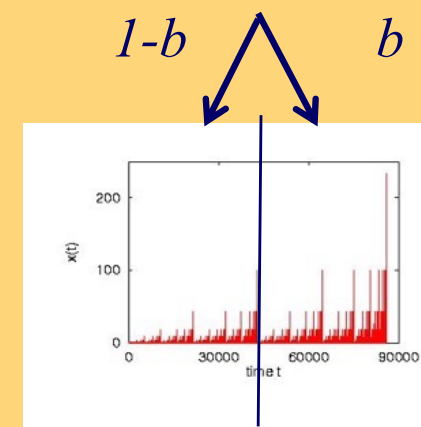
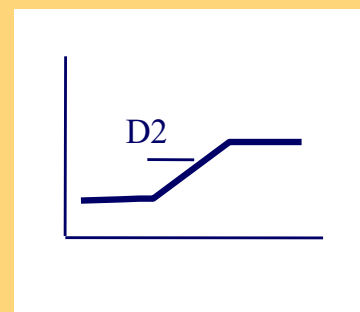
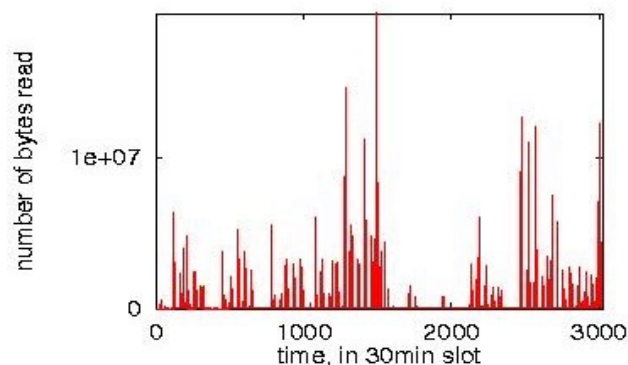
Conclusions

- 80/20, '*b*-model', Multiplicative Wavelet Model (MWM), for analysis and synthesis of bursty traffic



Bursty traffic

- Q: Any pattern?
- A: fractal dimension (' b ' model e.g. 80/20)



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.

Entropy plots



Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- ➔ • Non-linear forecasting
- Gray box modeling – Lotka Volterra eq's
- Conclusions



Detailed Outline

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

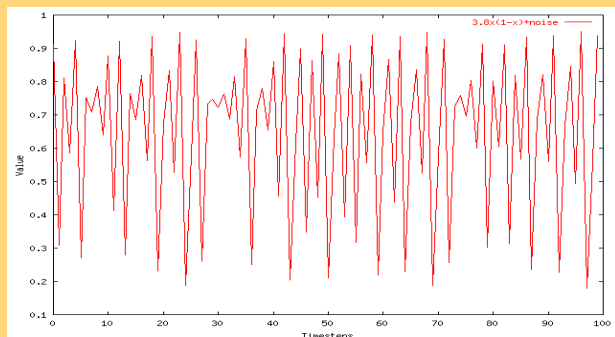
Reference:

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



Problem: Forecast

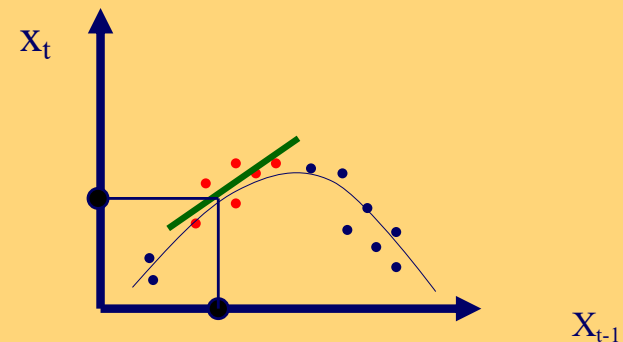
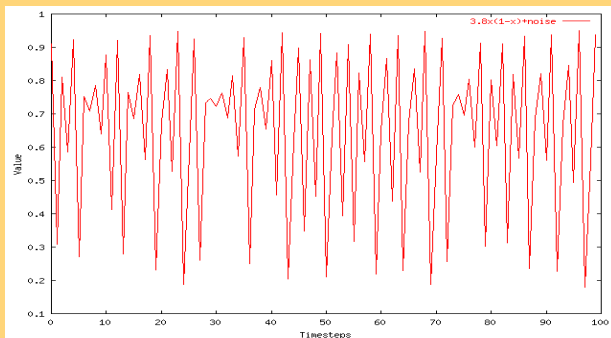
- given $x_{t-1}, x_{t-2}, \dots$, ('chaotic'/non-linear)
- Q: forecast x_t





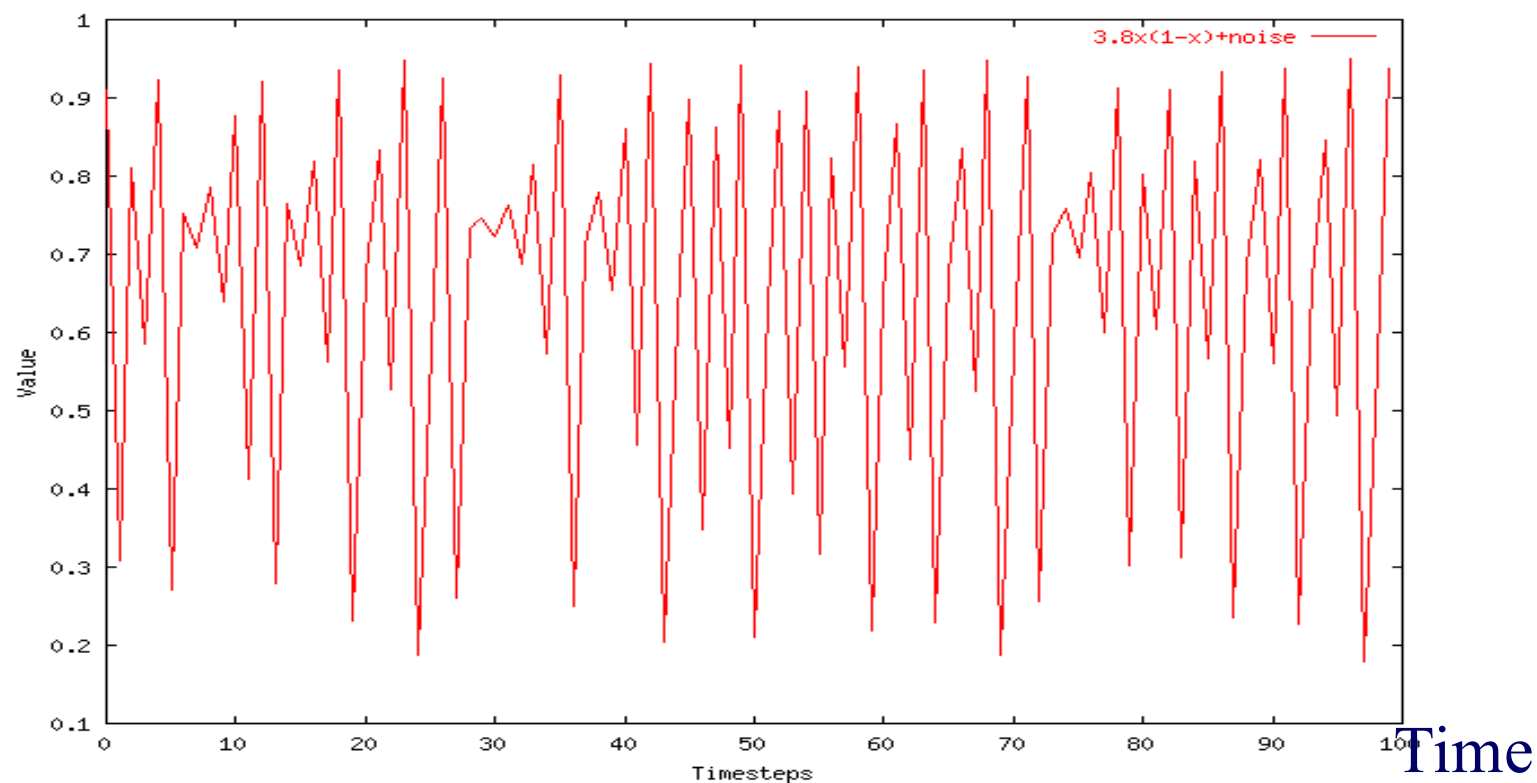
Solution

- given $x_{t-1}, x_{t-2}, \dots$, ('chaotic'/non-linear)
- Q: forecast x_t
- A: lag-plots + sim. search (= 'Delayed Coordinate Embedding')



Recall: Problem #1

Value



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

How to forecast?

- ARIMA - but: linearity assumption
- ANSWER: ‘Delayed Coordinate Embedding’ = Lag Plots [Sauer94]

ARIMA pitfall

Example: logistic parabola

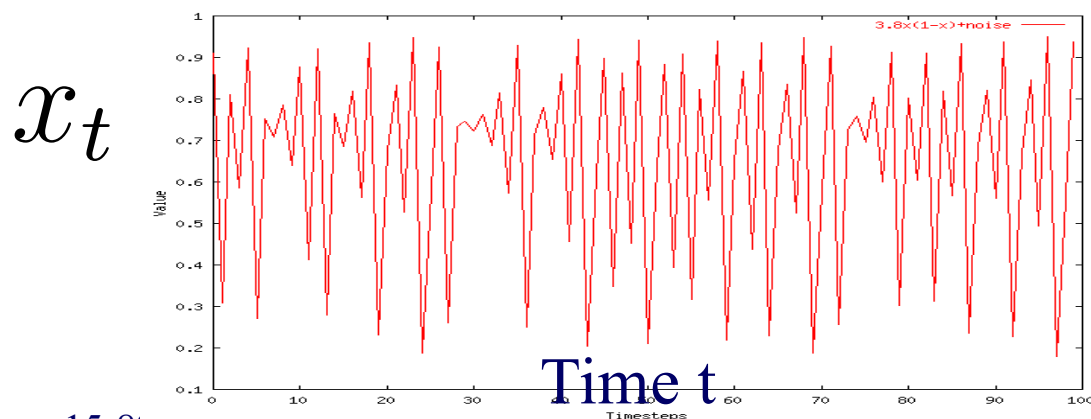
Models population of flies [R. May/1976]



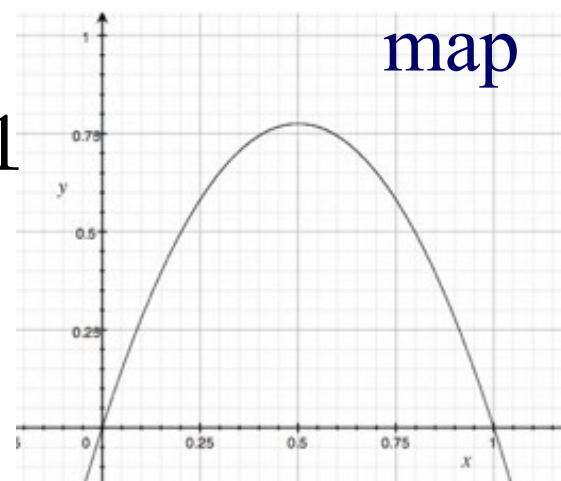
$$x_{t+1} = ax_t \cdot (1 - x_t)$$

Logistic
map

Time-series plot



x_{t+1}



x_t

ARIMA pitfall

Example: logistic parabola

Models population of flies [R. May/1976]



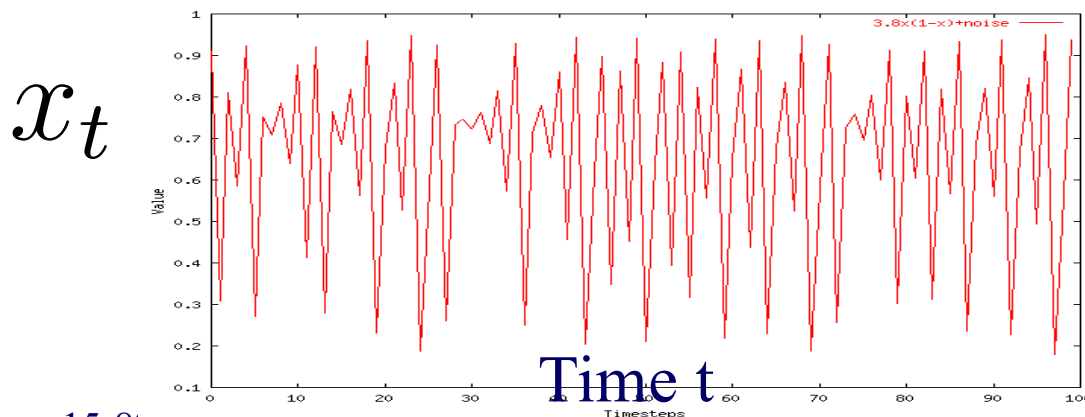
#flies(t+1)

Reproductive
rate

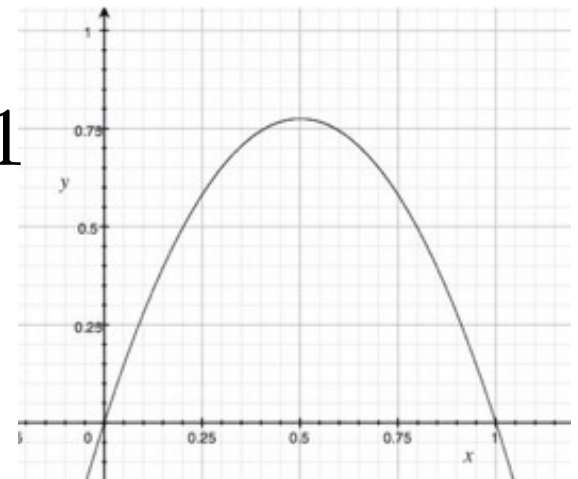
Max# flies

$$x_{t+1} = ax_t \cdot (M - x_t)$$

Time-series plot



x_{t+1}



x_t

ARIMA pitfall

Example: logistic parabola

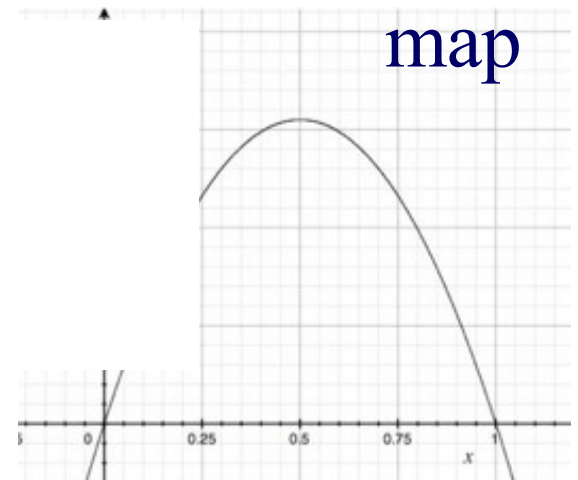
Models population of flies [R. May/1976]



$$x_{t+1} = ax_t \cdot (1 - x_t)$$

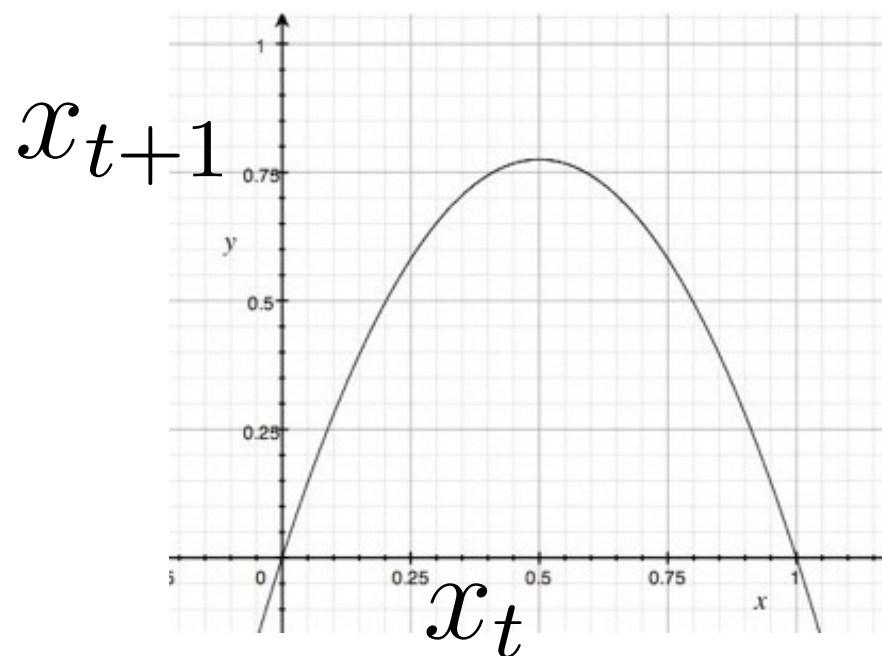
Logistic
map

- = SI virus prop. model
- ~ Bass equation (market penetration)
- Special case of Lotka-Volterra



ARIMA pitfall

Linear equations, e.g., AR, ARIMA, ...

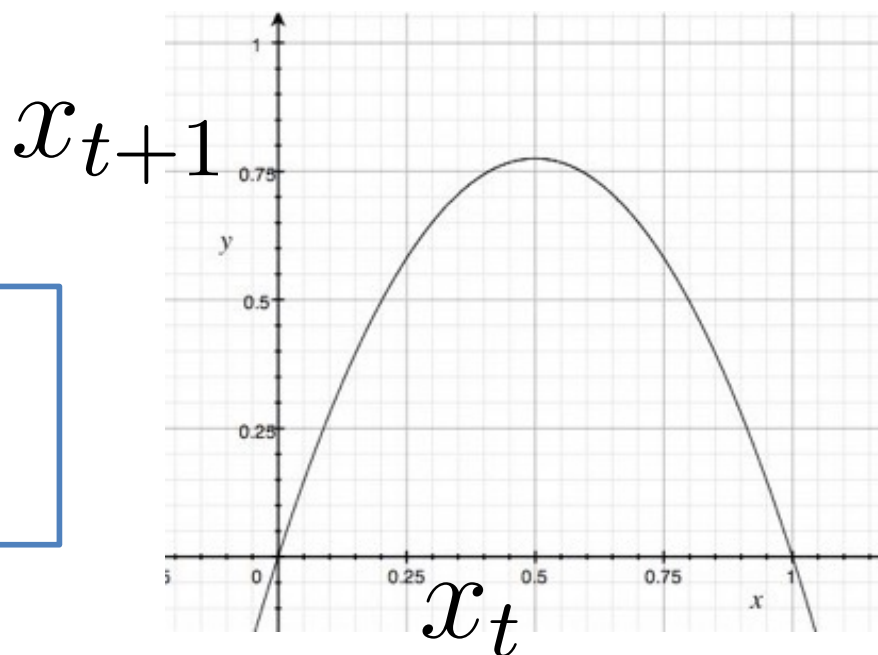


ARIMA pitfall

Linear equations, e.g., AR, ARIMA, ...

e.g., AR(1)

$$x_{t+1} = ax_t + \epsilon$$

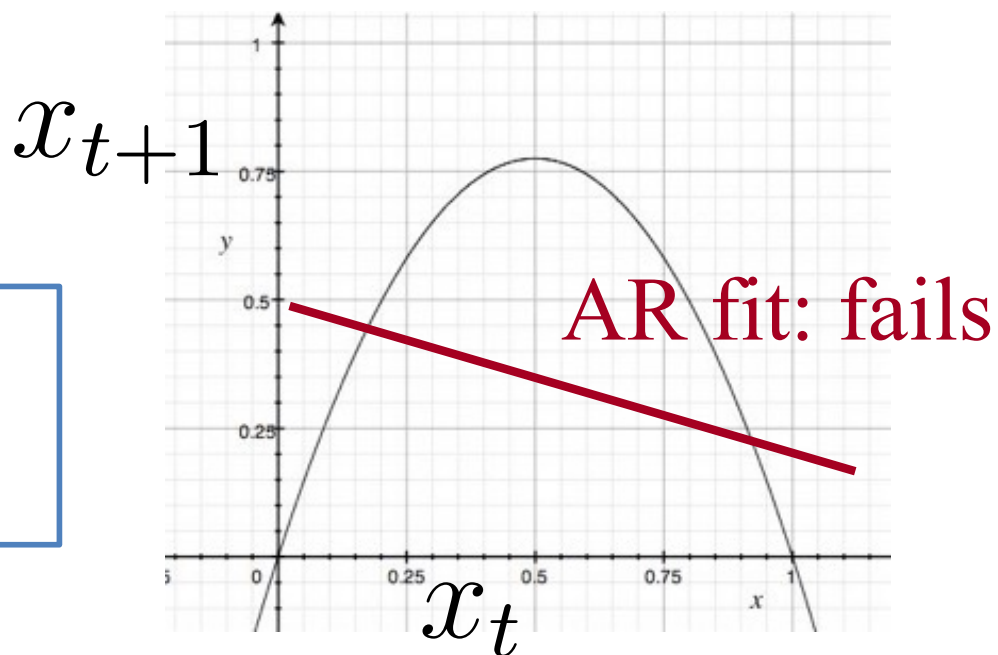


ARIMA pitfall

Linear equations, e.g., AR, ARIMA, ...

e.g., AR(1)

$$x_{t+1} = ax_t + \epsilon$$



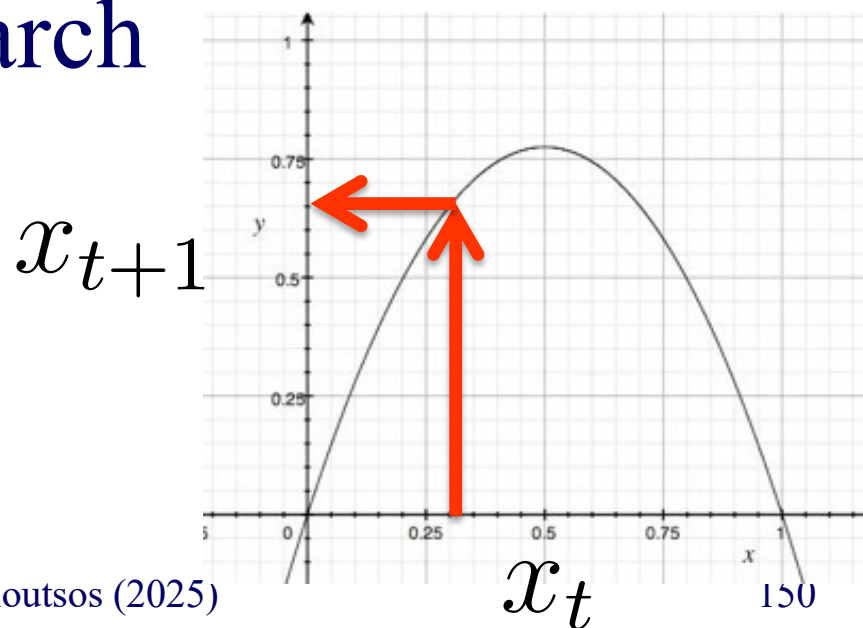
Solution?

“Delayed Coordinate Embedding”

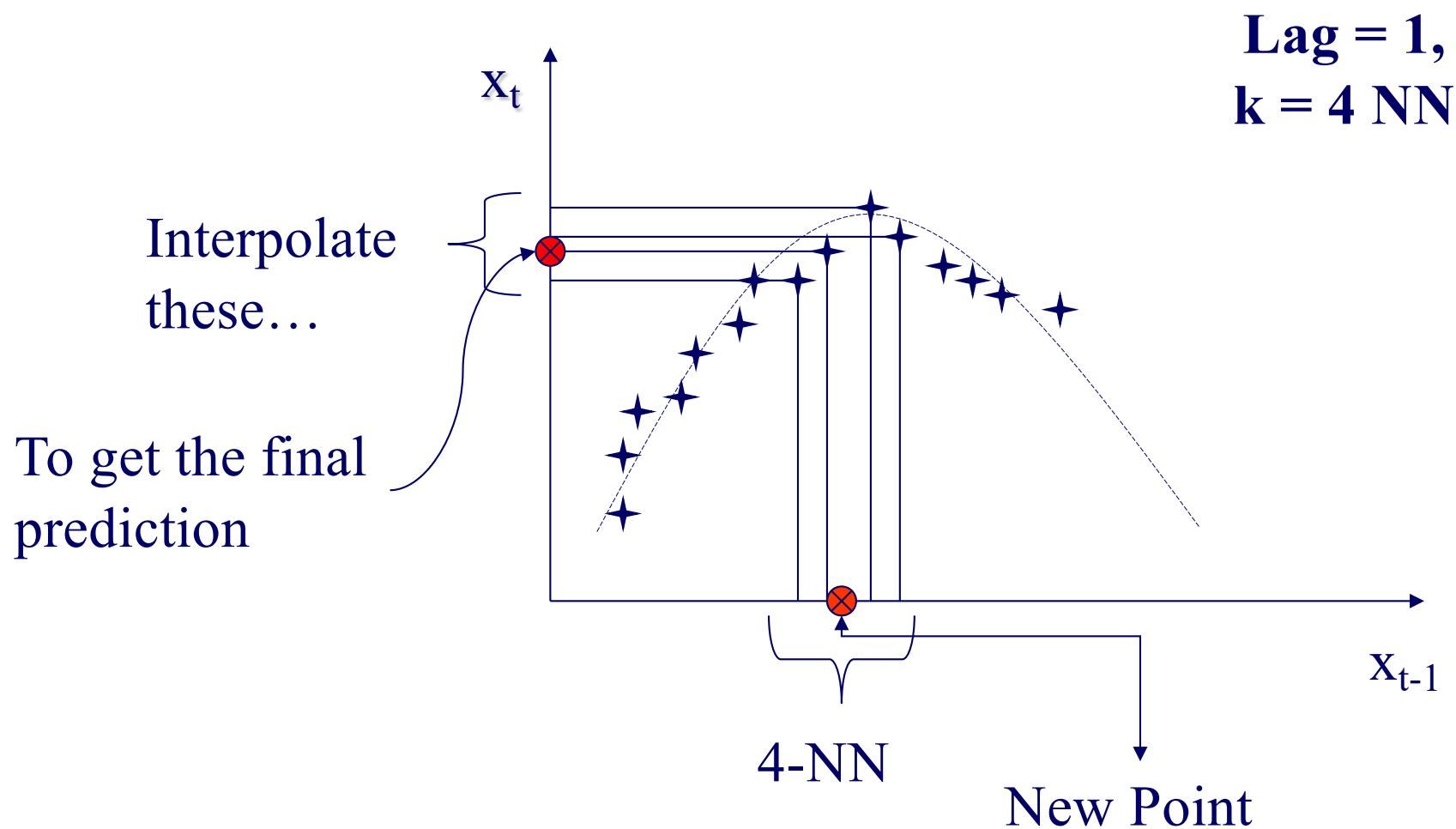
= Lag Plots

[Sauer94]

k-nearest neighbor search



General Intuition (Lag Plot)



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 $\sim 1-10$)

Q3: How to interpolate?

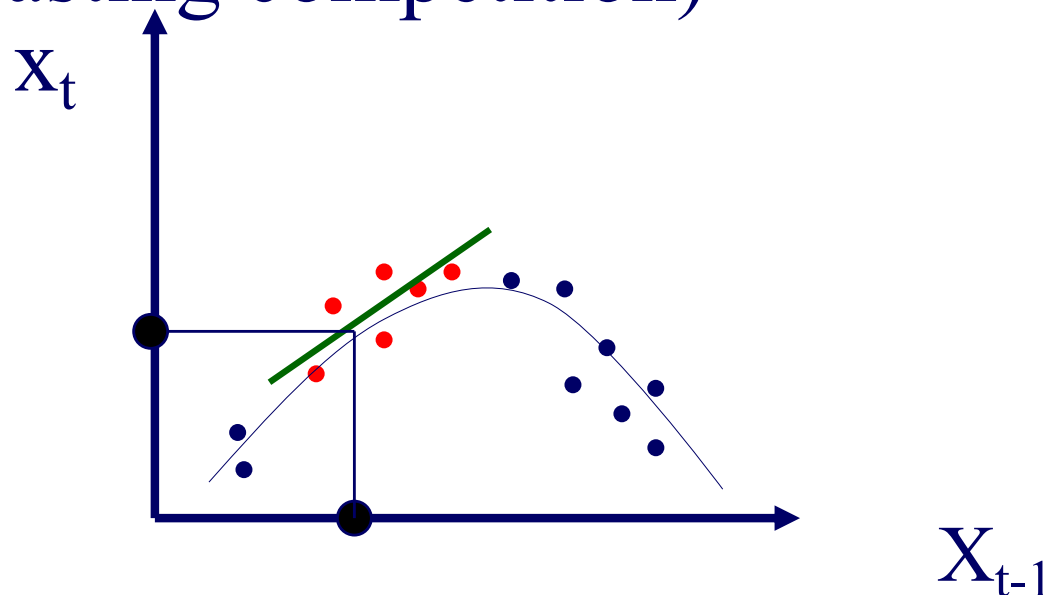
How do we interpolate between the k nearest neighbors?

A1: Average

A2: Weighted average (weights drop with distance - how?)

Q3: How to interpolate?

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

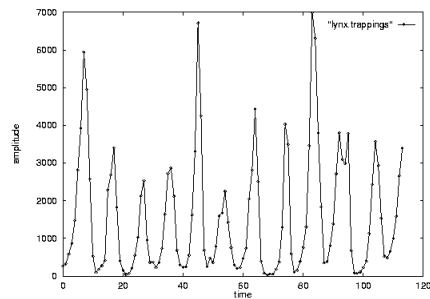


Q4: Any theory behind it?

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)



Theoretical foundation

Example: Lotka-Volterra equations

$$dH/dt = r H - a H * P$$

$$dP/dt = b H * P - m P$$

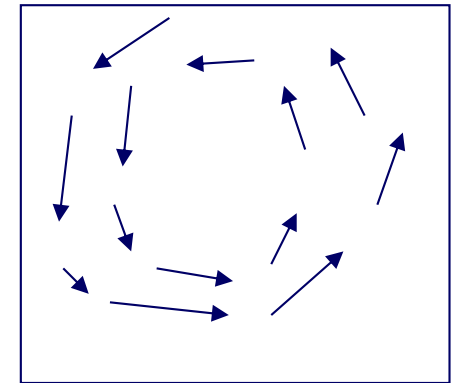
H is count of prey (e.g., hare)

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

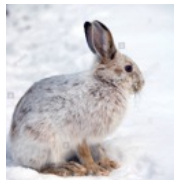
Suppose only $P(t)$ is observed ($t=1, 2, \dots$).



P

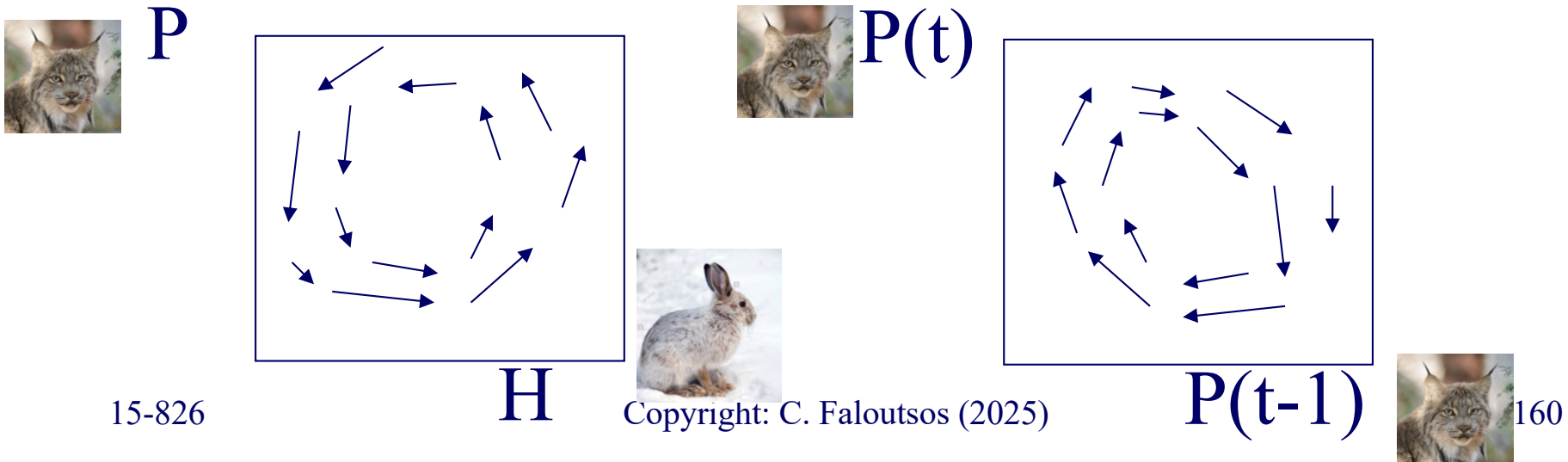


H



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**





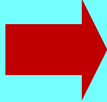
Detailed Outline

- Non-linear forecasting

- Problem

- Idea

- How-to

-  – Experiments

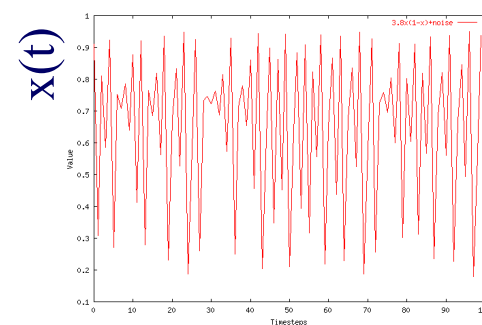
- Conclusions

Datasets

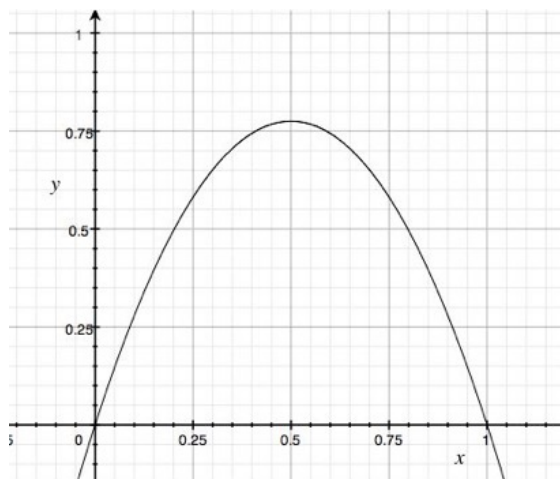
Logistic Parabola:

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

Models population of flies [R. May/1976]



time

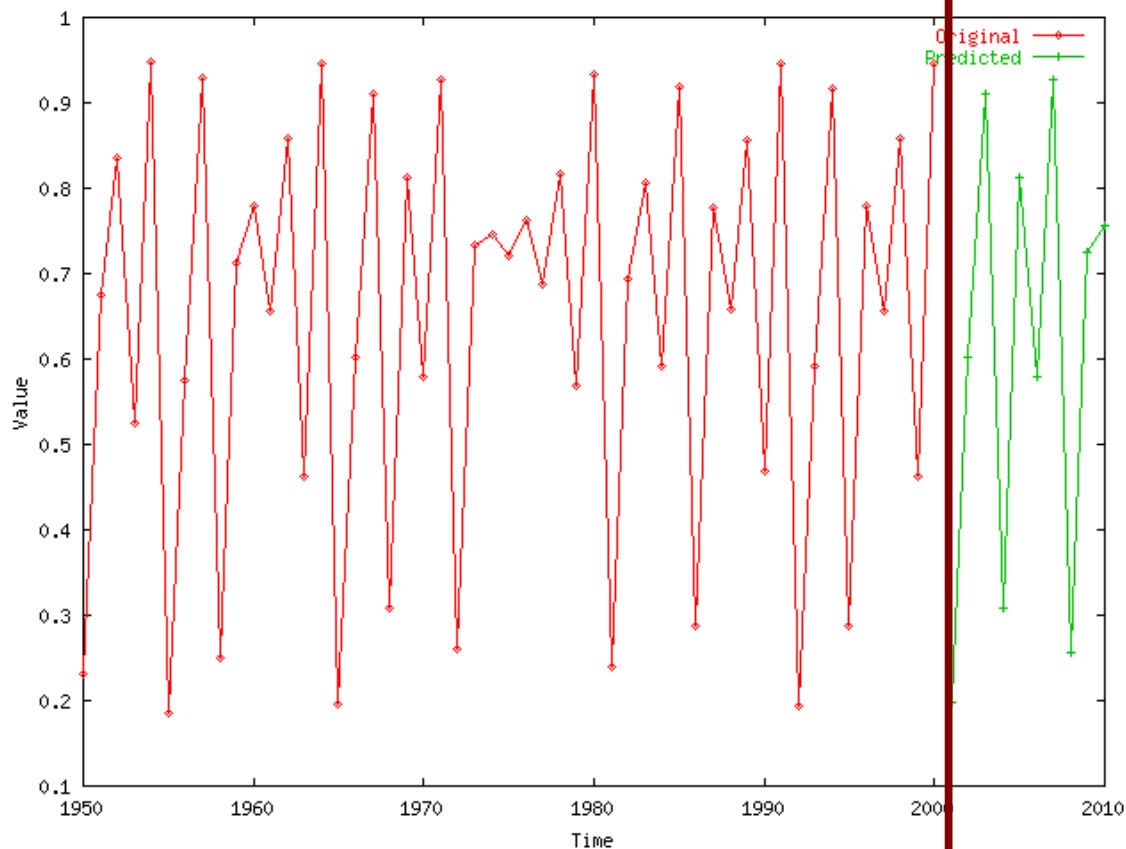


Lag-plot

Logistic Parabola

Our Prediction from
here

Value

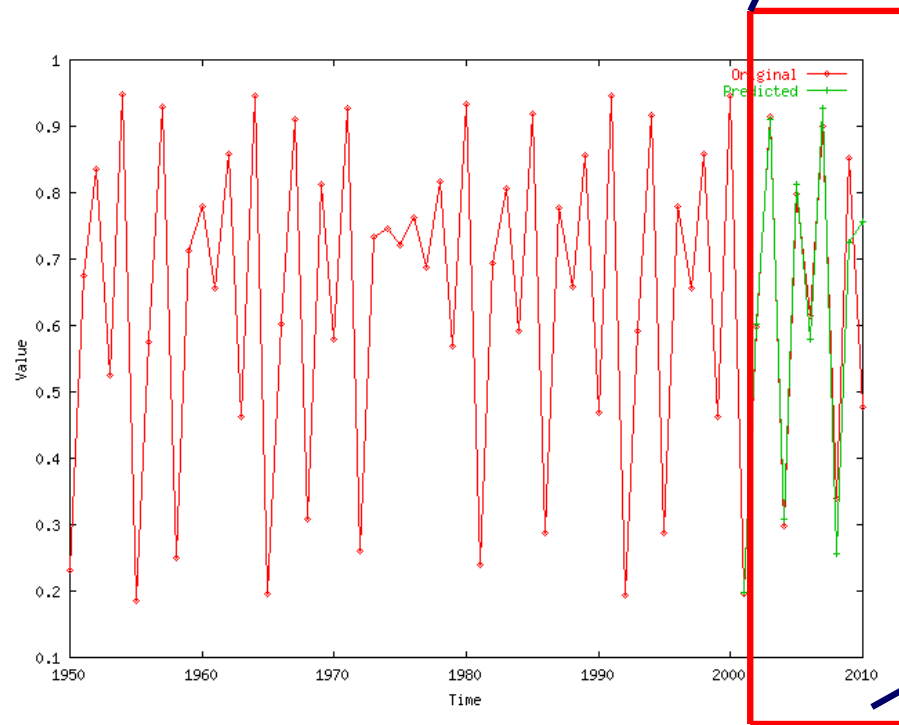
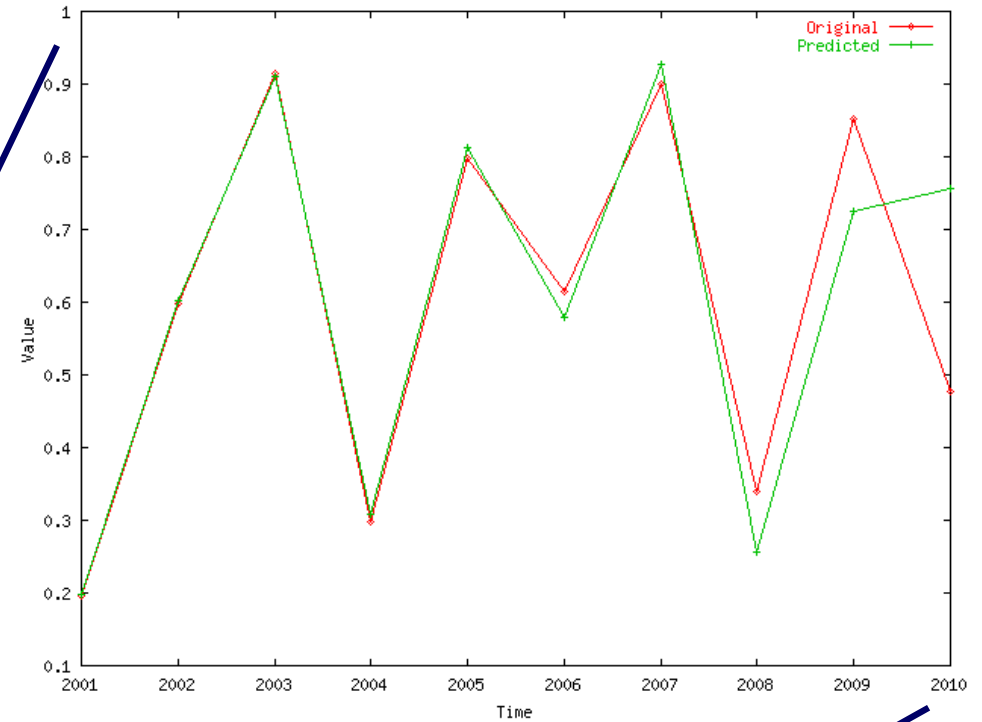


Timesteps

Value

Logistic Parabola

Comparison of prediction
to correct values



Timesteps

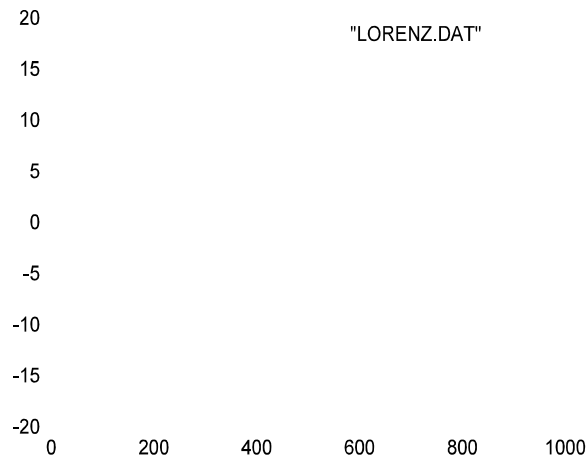
Datasets

LORENZ: Models convection currents in the air

$$dx / dt = a (y - x)$$

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

$$dz / dt = xy - c z$$





Edward Lorenz

Datasets

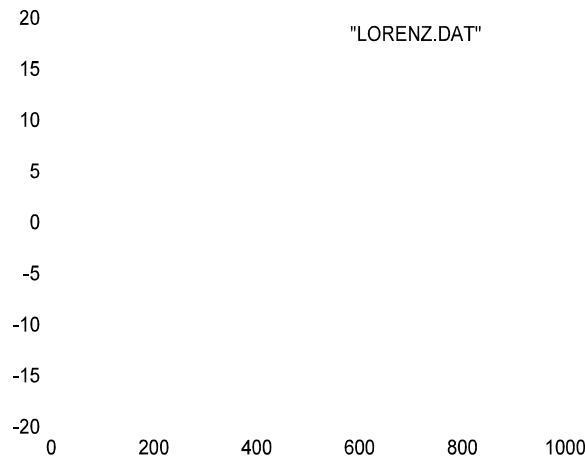
LORENZ: Models convection currents in the air

$$dx / dt = a (y - x)$$

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

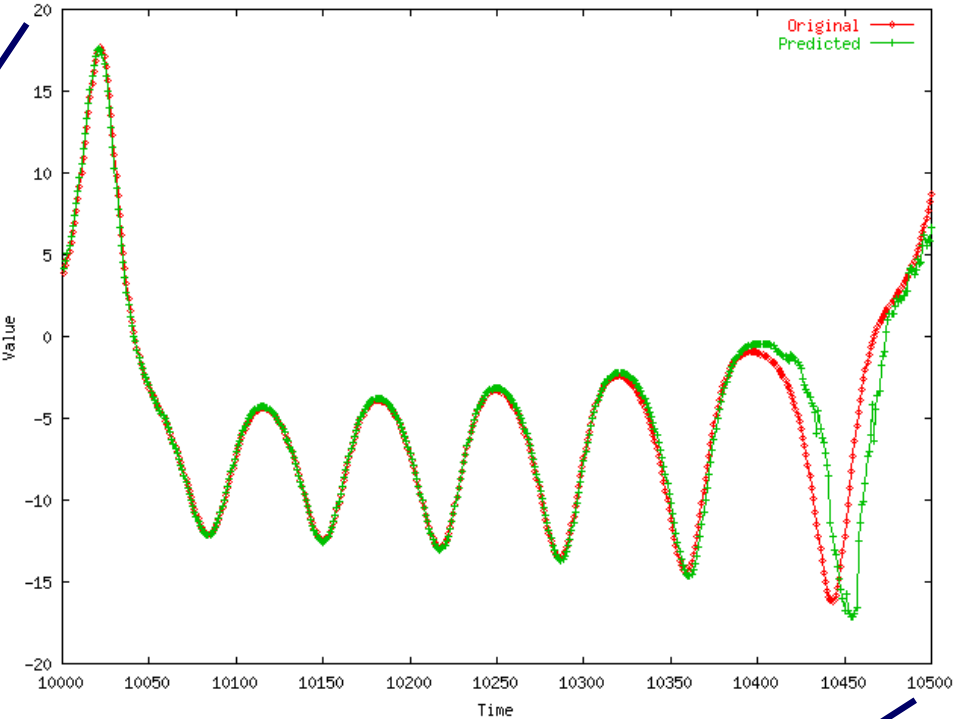
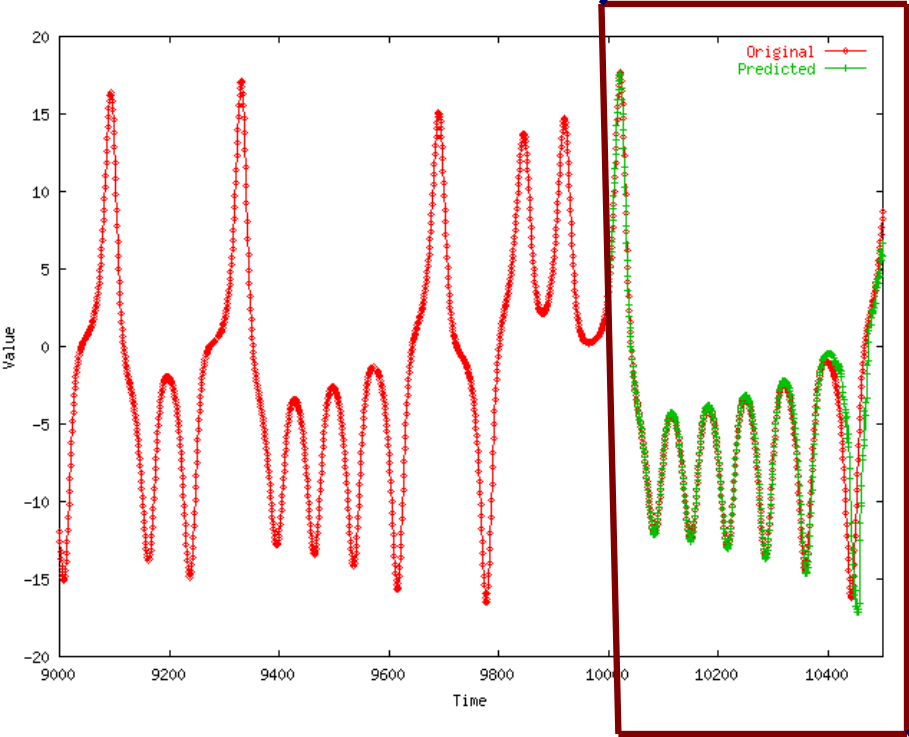
$$dz / dt = xy - c z$$

- **Deterministic chaos**
- **‘butterfly effect’**



LORENZ

Comparison of prediction
to correct values

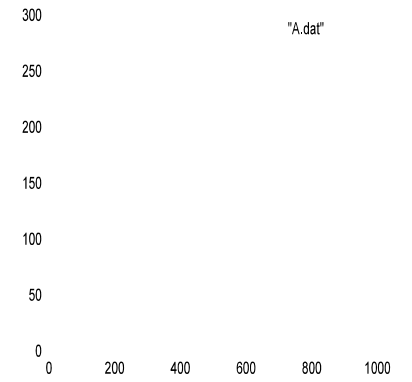


Timesteps

Value

Datasets

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

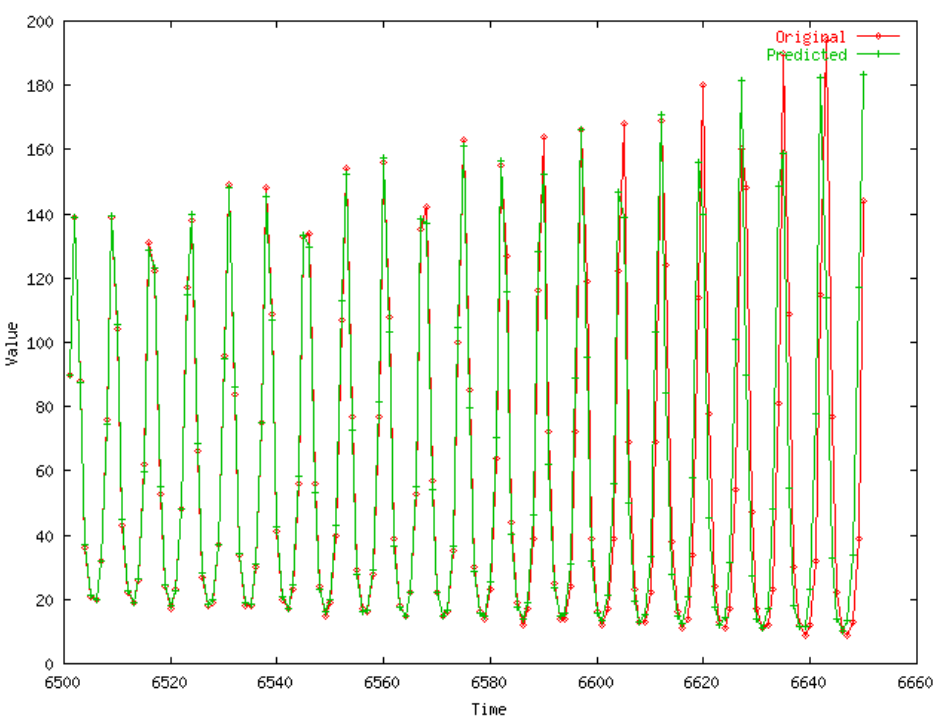


Time

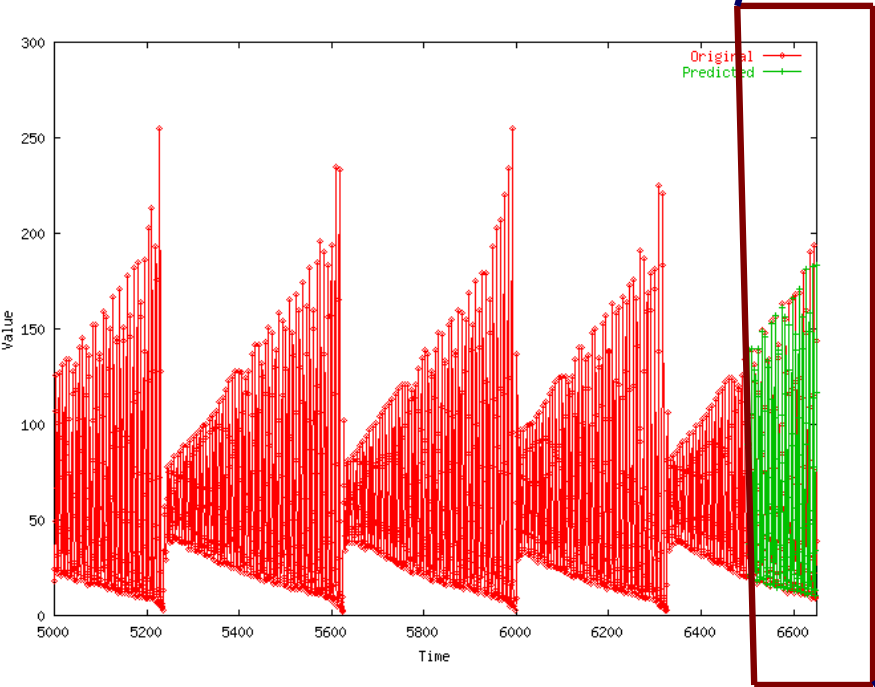
Laser

Comparison of prediction
to correct values

Value



Timesteps



Conclusions

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



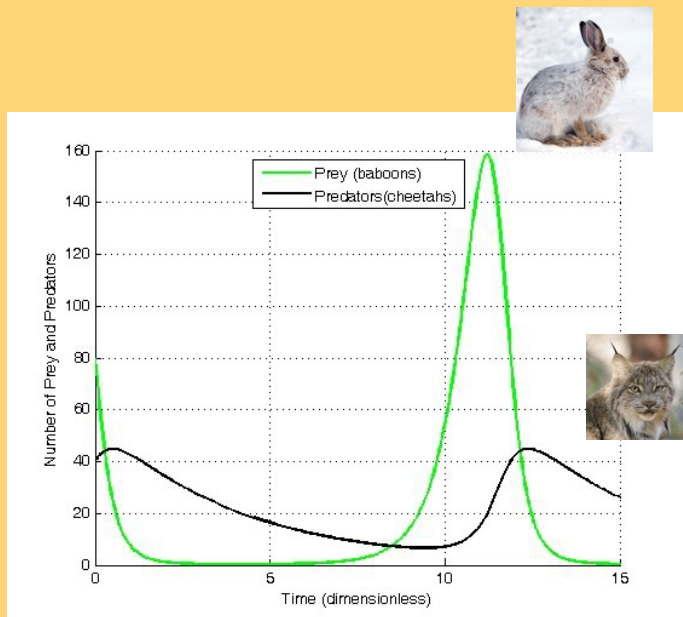
Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- ➔ • Gray box modeling – Lotka Volterra eq's
- Conclusions



Problem: Gray-box forecast

Q: How to model (competing)
species/products/ideas?

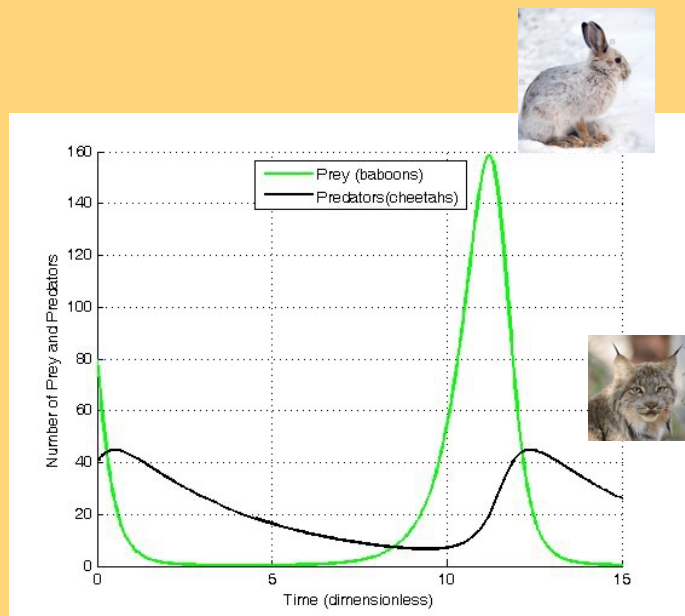




Answer

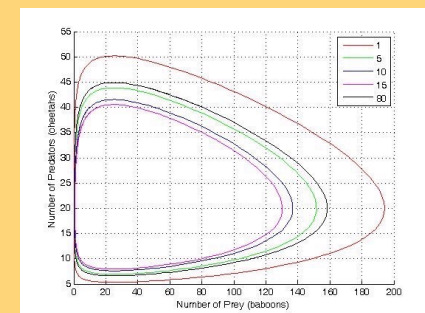
Q: How to model (competing) species/products/ideas?

A: Lotka-Volterra



$$P_i(t+1) = P_i(t) \left[1 + r_i \left(1 - \frac{\sum_{j=1}^d a_{ij} P_j(t)}{K_i} \right) \right],$$

$(i = 1, \dots, d), \quad (3)$



Theoretical foundation

Example: Lotka-Volterra equations

$$dH/dt = r H - a H * P$$

$$dP/dt = b H * P - m P$$

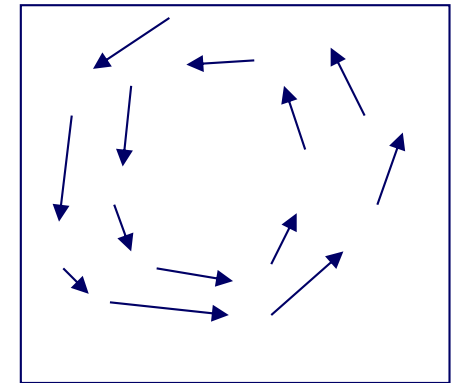
H is count of prey (e.g., hare)

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

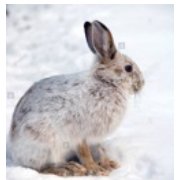
Suppose only $P(t)$ is observed ($t=1, 2, \dots$).



P

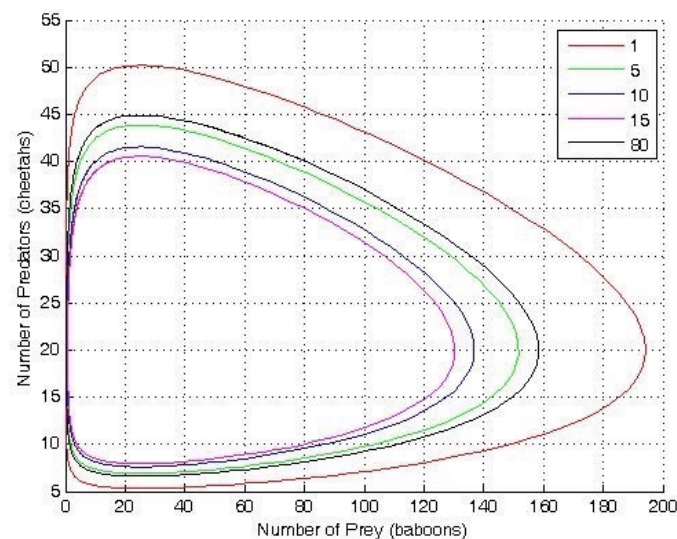
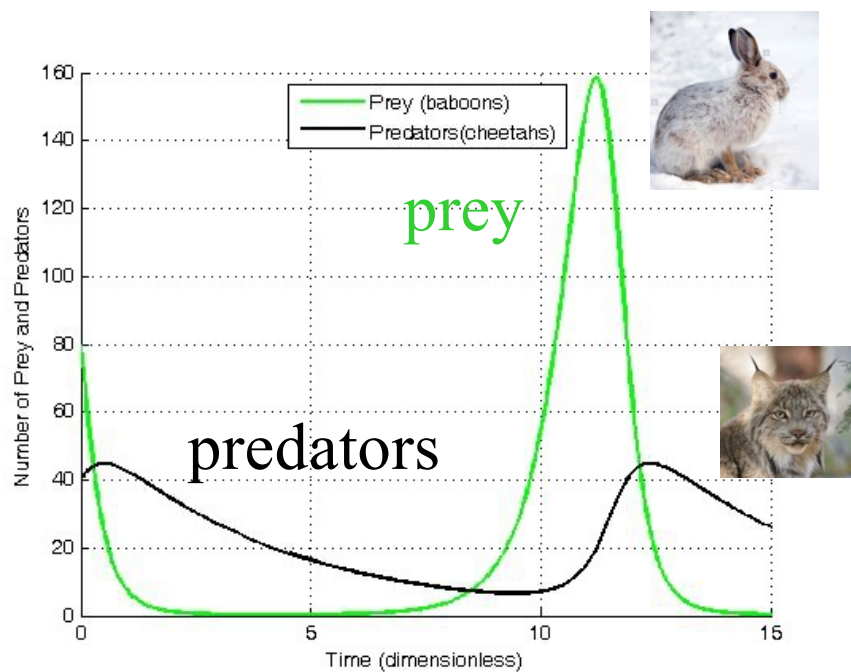


H



Solution to Lotka-Volterra eq.

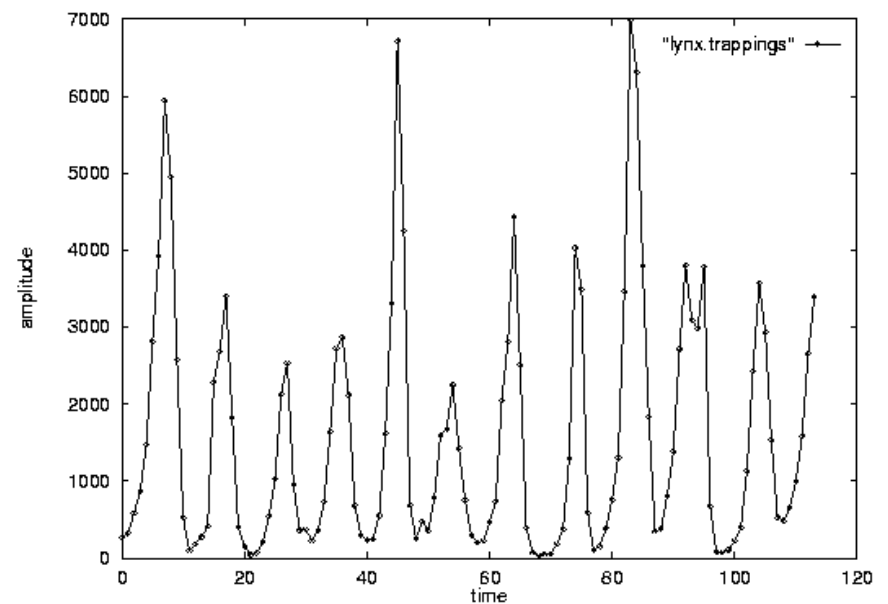
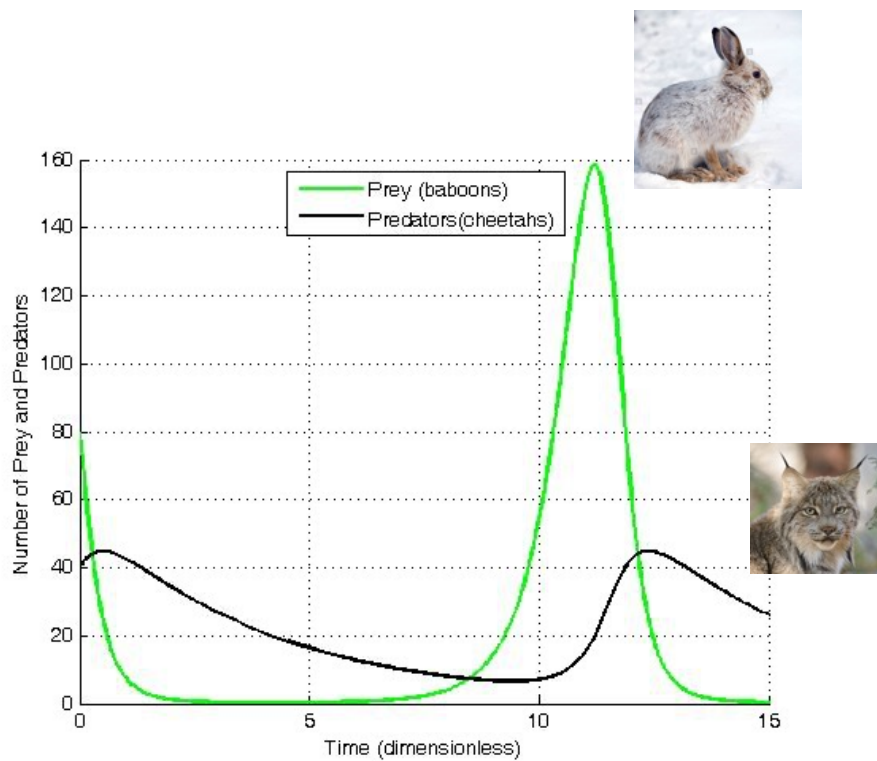
predators



prey

time

Compare to reality:



Notice: LV are vital!

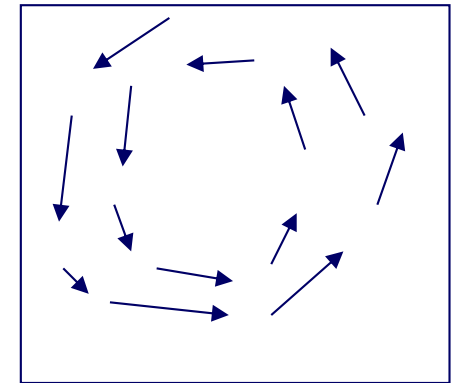
Example: Lotka-Volterra equations

$$dH/dt = r H - a H * P$$

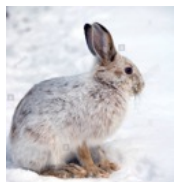
$$dP/dt = b H * P - m P$$



P



H



- Prey-predator
- Competing animals (rabbits/goats)
- Self-competition (Bass model)
- Competing products (stocks/bonds)

The Web as a Jungle: Non-Linear Dynamical Systems for Co-evolving Online Activities



Yasuko Matsubara (Kumamoto University)



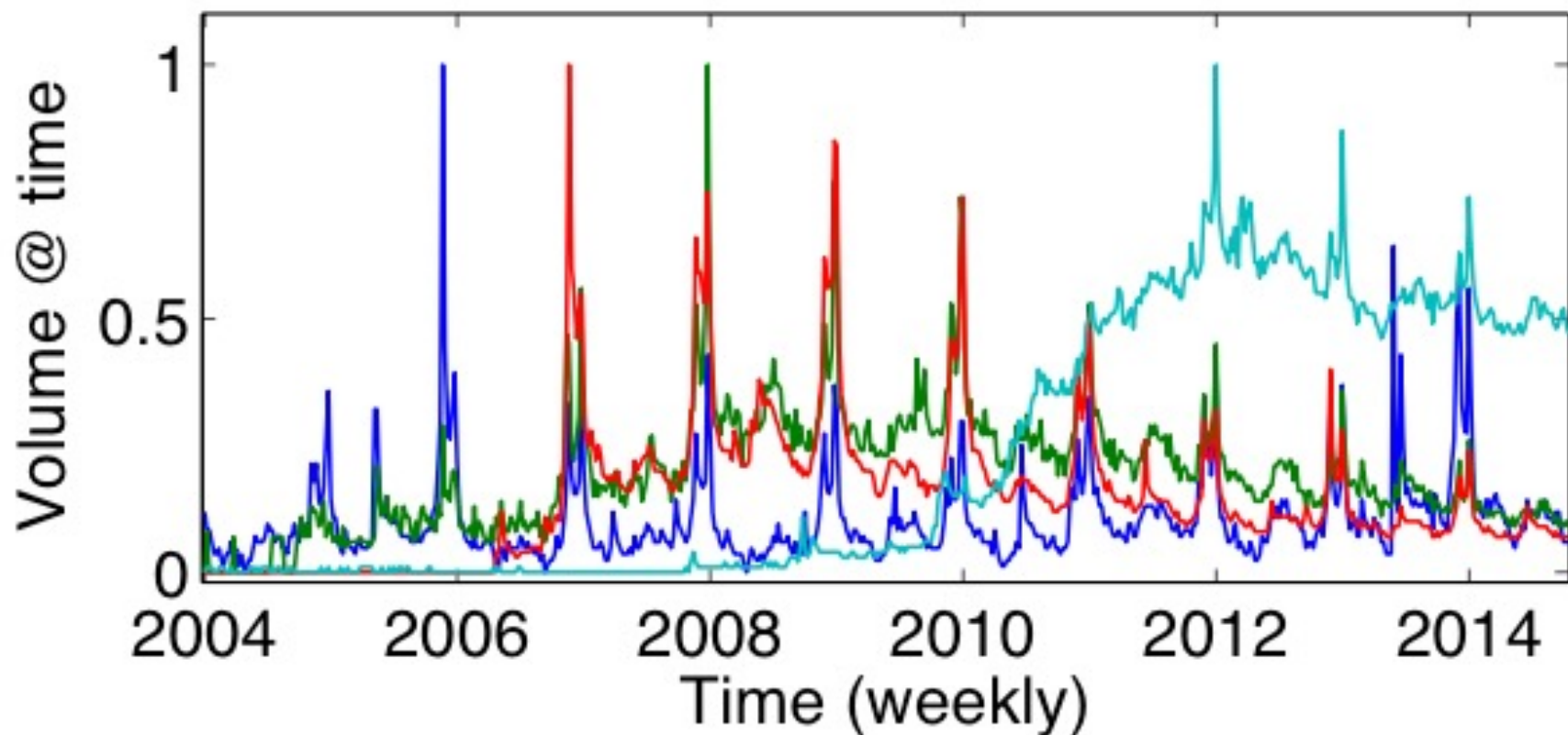
Yasushi Sakurai (Kumamoto University)

Christos Faloutsos (CMU)

Open source code: [here](#)

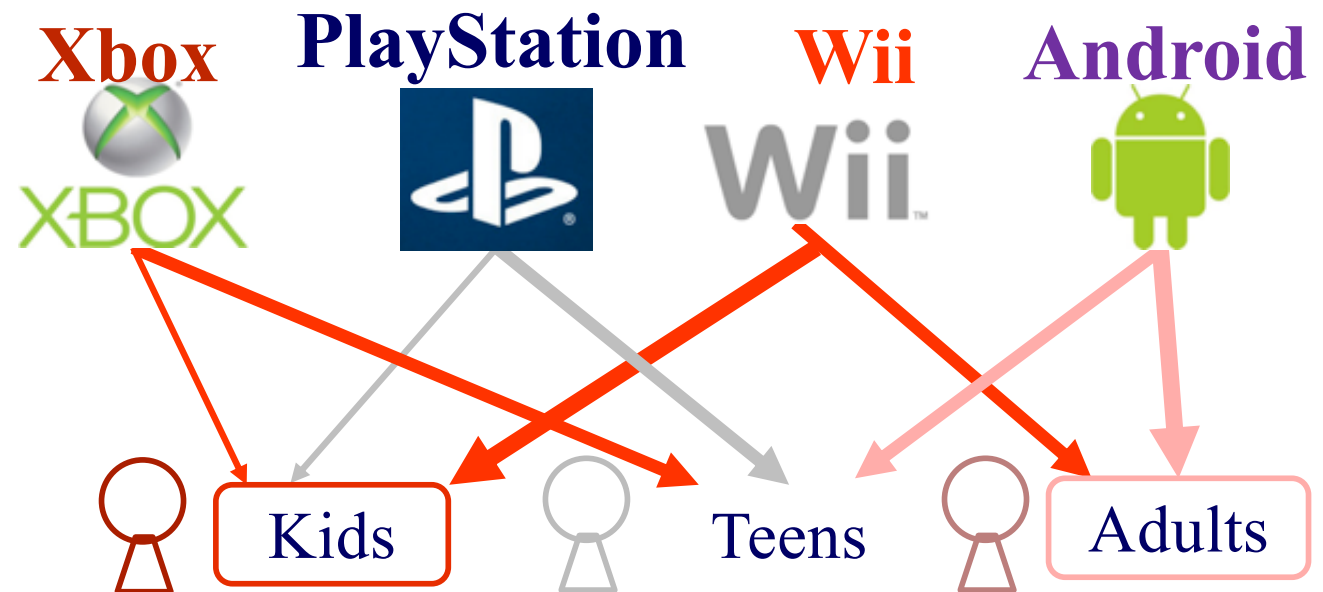
Given: online user activities

Xbox, **PlayStation**, **Wii**, **Android**



The Web as a jungle

Ecosystem on the Web



The Web as a jungle

Squirrel monkeys



Spider monkeys



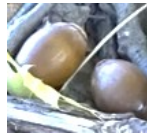
Macaws



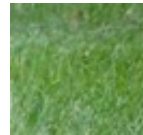
Capybaras



Fruits



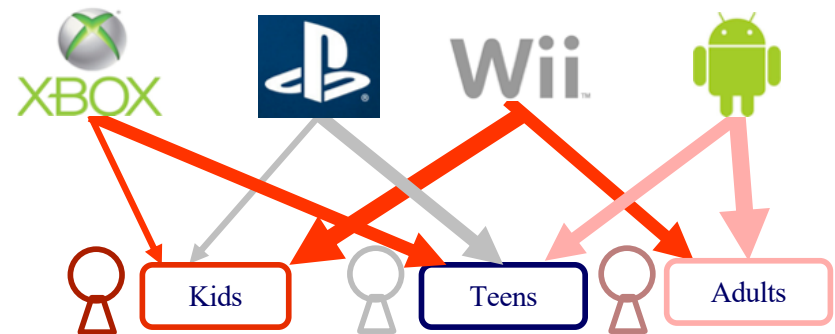
Nuts



Grass



Ecosystem in the Jungle



LV equations

Interaction between multiple (' d ')
species/products/viruses



...



$$\boxed{P_i(t+1)} = \boxed{P_i(t)} \left[1 + r_i \left(1 - \frac{\sum_{j=1}^d a_{ij} \boxed{P_j(t)}}{K_i} \right) \right],$$

$$(i = 1, \dots, d),$$

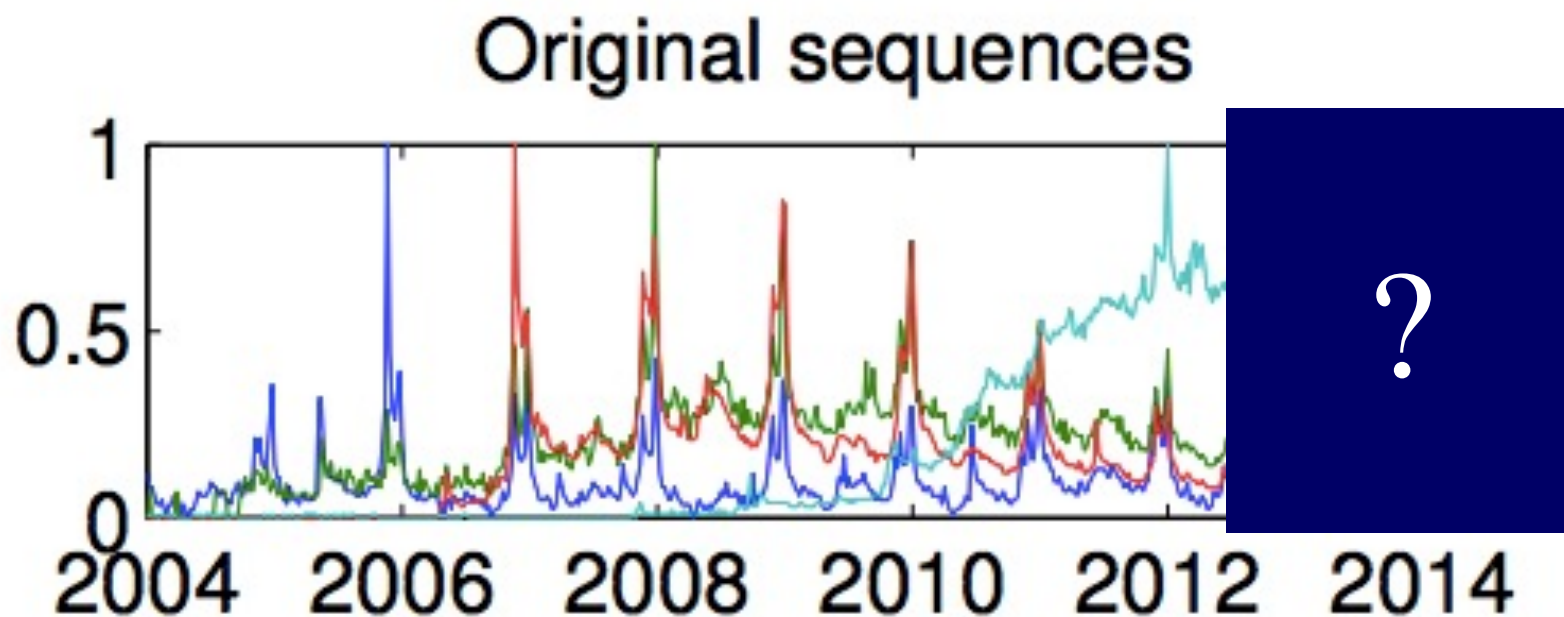
a_{ij} - effect of species j on species i

- (positive: hurts)

EcoWeb at work - forecasting

Train:
2/3 sequences

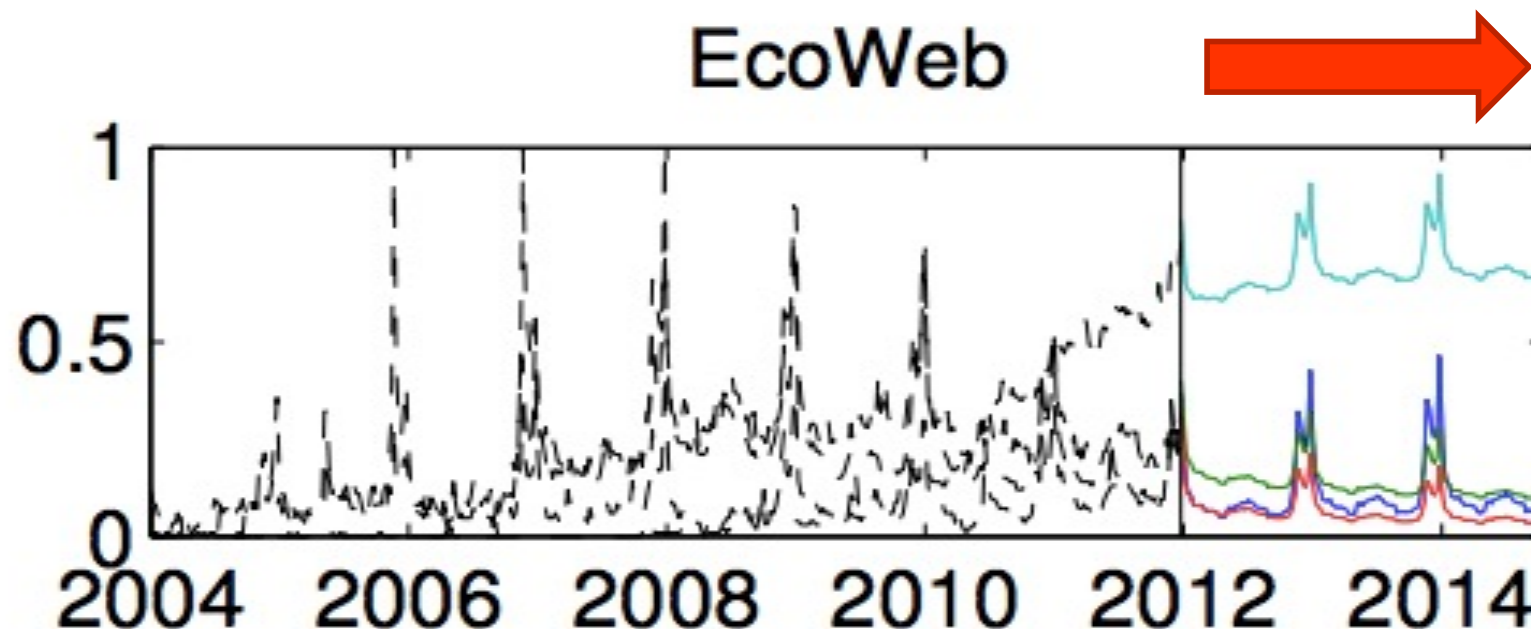
Forecast:
1/3 following years



EcoWeb at work - forecasting

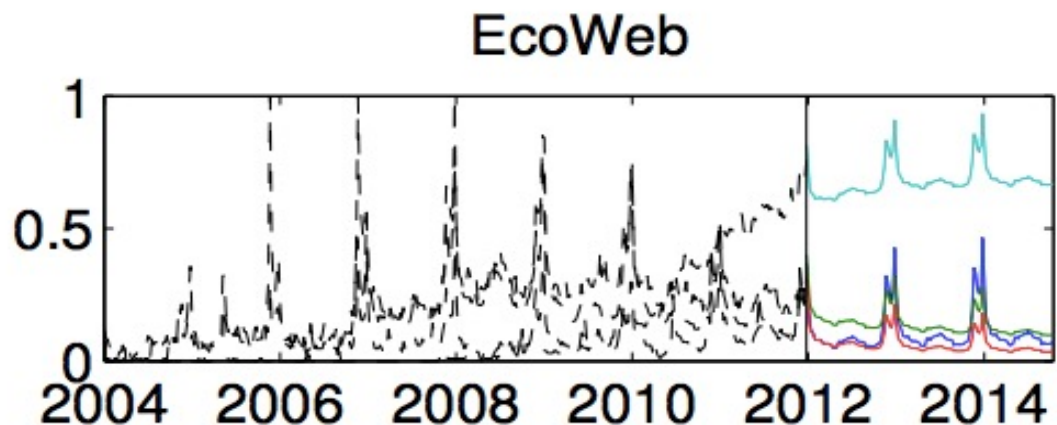
Train:
2/3 sequences

Forecast:
1/3 following years



EcoWeb can capture future patterns

EcoWeb at work - forecasting



Open source code: [here](#)



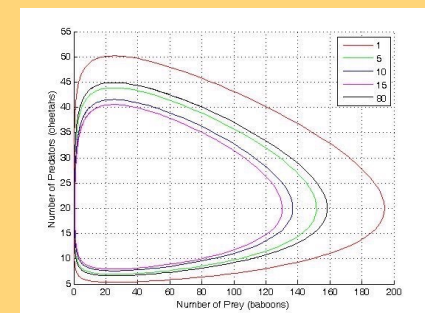
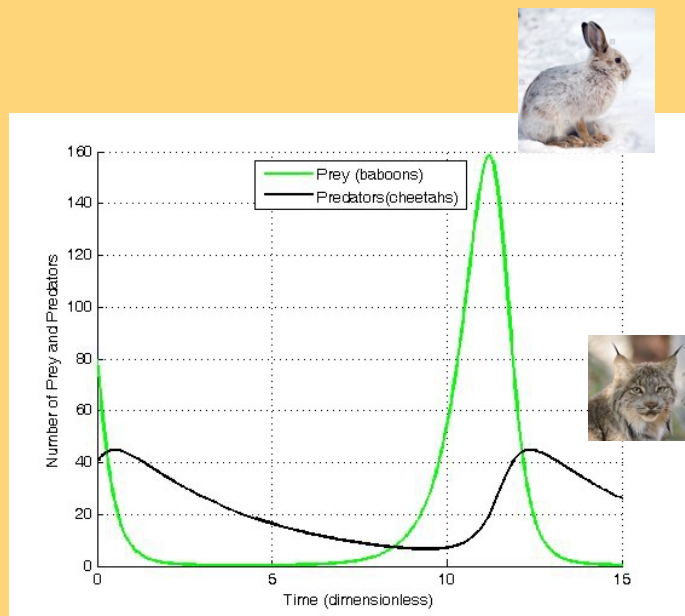
Answer

Q: How to model (competing) species/products/ideas?

A: Lotka-Volterra

$$P_i(t+1) = P_i(t) \left[1 + r_i \left(1 - \frac{\sum_{j=1}^d a_{ij} P_j(t)}{K_i} \right) \right],$$

$(i = 1, \dots, d), \quad (3)$





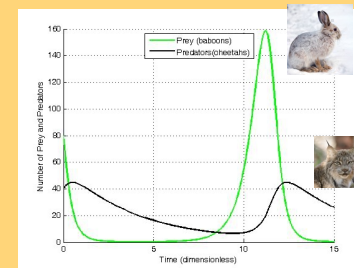
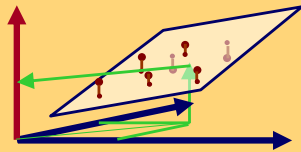
Outline

- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Gray box modeling – Lotka Volterra eq's
- ➔ • Conclusions



Overall conclusions

- Similarity search: **Euclidean/time-warping; feature extraction and SAMs**
- Linear Forecasting: **AR (Box-Jenkins)**
- Non-linear forecasting: **lag-plots**
- Gray-box modeling: **Lotka-Volterra**



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.

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.)