CMU SCS

15-826: Multimedia Databases and Data Mining

Lecture #27: Time series mining and forecasting

Christos Faloutsos

CMU SCS

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

CMU SCS

Thanks

Deepay Chakrabarti (UT-Austin)

Spiros Papadimitriou (Rutgers)

Prof. Byoung-Kee Yi (Samsung)

CMU SCS

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 , … , x_t , …
  (y_1, y_2, … , y_t, …
  …)
- Find
  - similar sequences; forecasts
  - patterns; clusters; outliers

Motivation - Applications

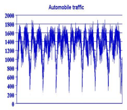
- Financial, sales, economic series
- Medical
  - ECGs +; blood pressure etc monitoring
  - reactions to new drugs
  - elderly care

Motivation - Applications (cont’ d)

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

Motivation - Applications (cont’ d)

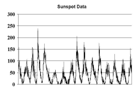
- civil/automobile infrastructure
  - bridge vibrations [Oppenheim+02]
  - road conditions / traffic monitoring



CMU SCS

Motivation - Applications (cont’d)

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



15-826 (c) C. Faloutsos, 2017 9

CMU SCS

Motivation - Applications (cont’d)

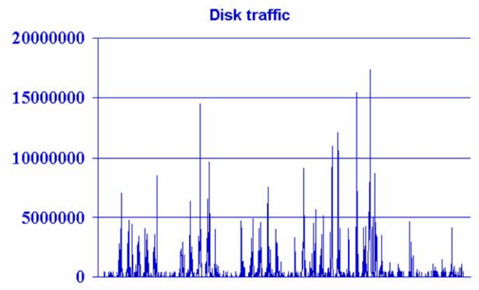
- Computer systems
  - ‘Active Disks’ (buffering, prefetching)
  - web servers (ditto)
  - network traffic monitoring
  - ...

15-826 (c) C. Faloutsos, 2017 10

CMU SCS

Stream Data: Disk accesses

#bytes



time

15-826 (c) C. Faloutsos, 2017 11

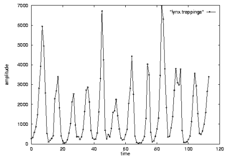
CMU SCS

Problem #1:

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

Find: patterns, periodicities, and/or compress

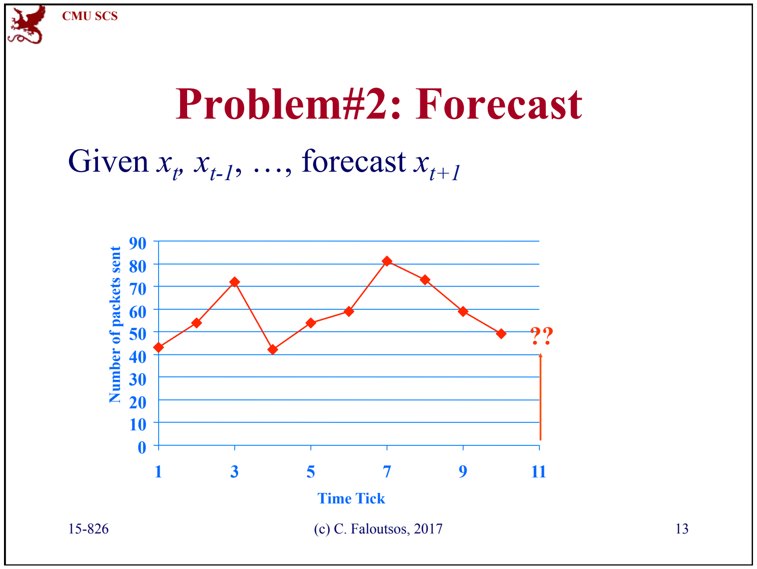
count



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

year

15-826 (c) C. Faloutsos, 2017 12

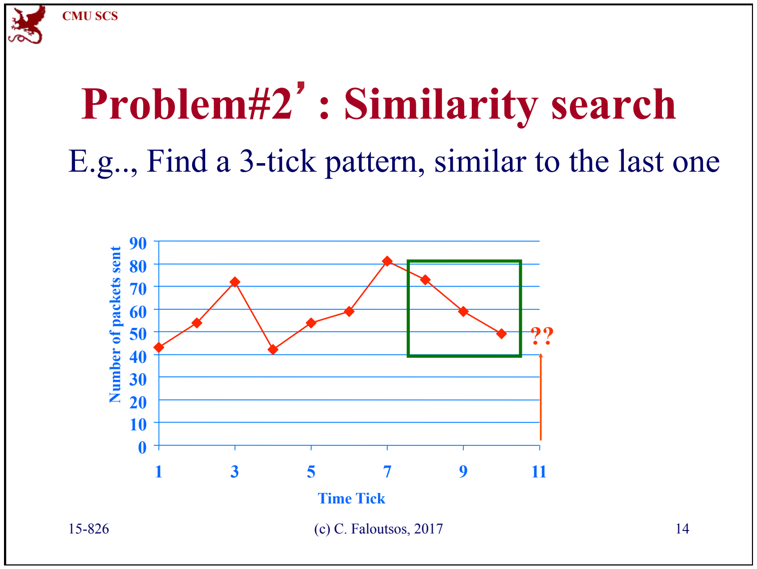


CMU SCS

Problem#2: Forecast

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

15-826 (c) C. Faloutsos, 2017 13

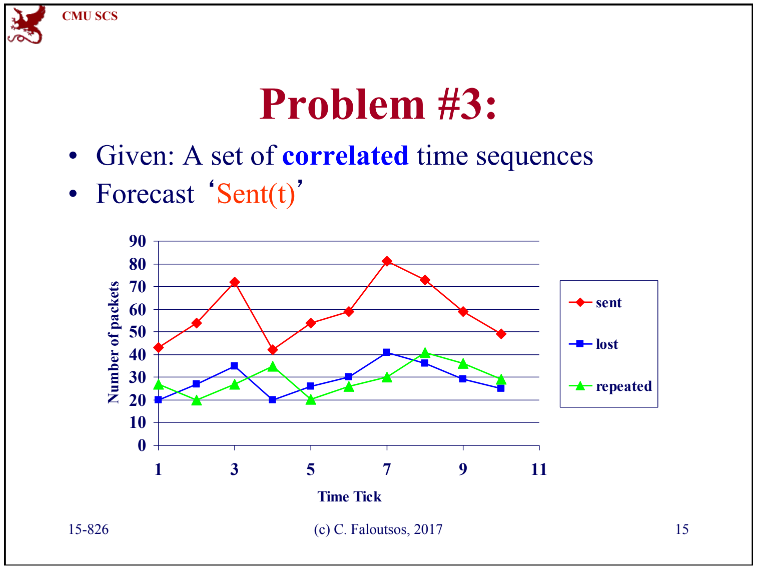


CMU SCS

Problem#2’: Similarity search

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

15-826 (c) C. Faloutsos, 2017 14

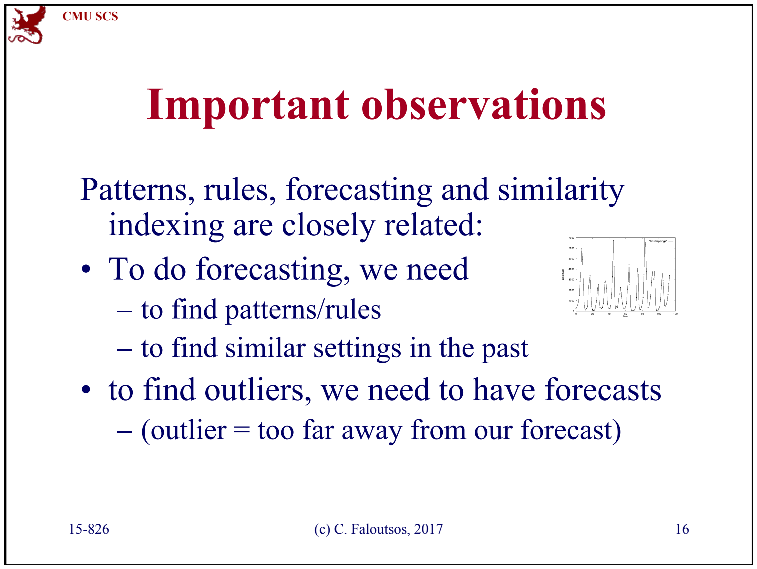


CMU SCS

Problem #3:

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

15-826 (c) C. Faloutsos, 2017 15



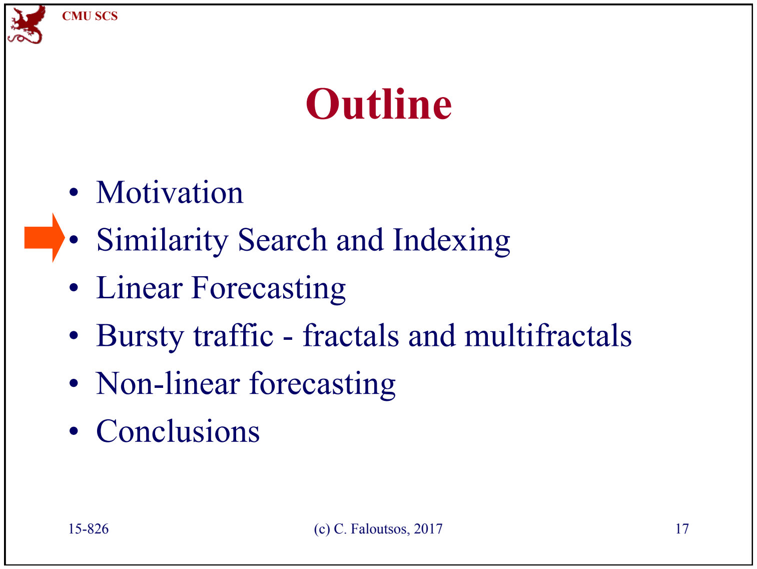
CMU SCS

Important observations

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

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

15-826 (c) C. Faloutsos, 2017 16



CMU SCS

Outline

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

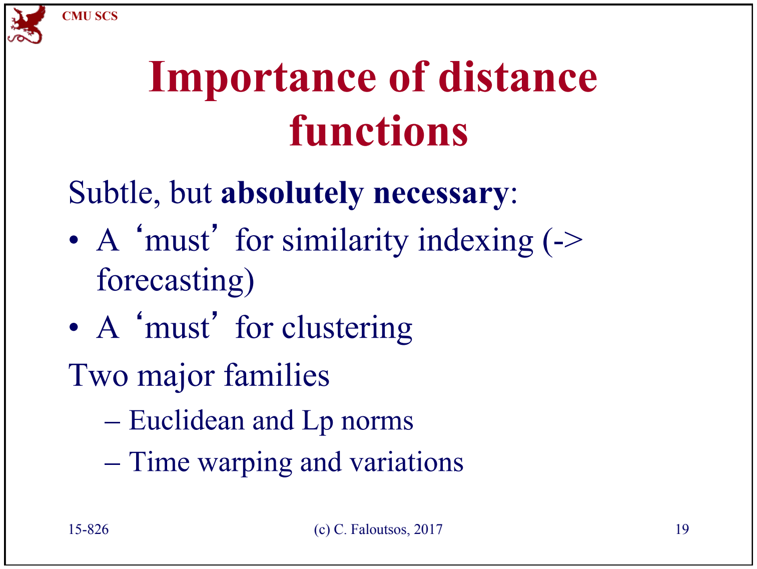
15-826 (c) C. Faloutsos, 2017 17

CMU SCS

Outline

- Motivation
- Similarity search and distance functions
  - Euclidean
  - Time-warping
- …

15-826 (c) C. Faloutsos, 2017 18



CMU SCS

Importance of distance functions

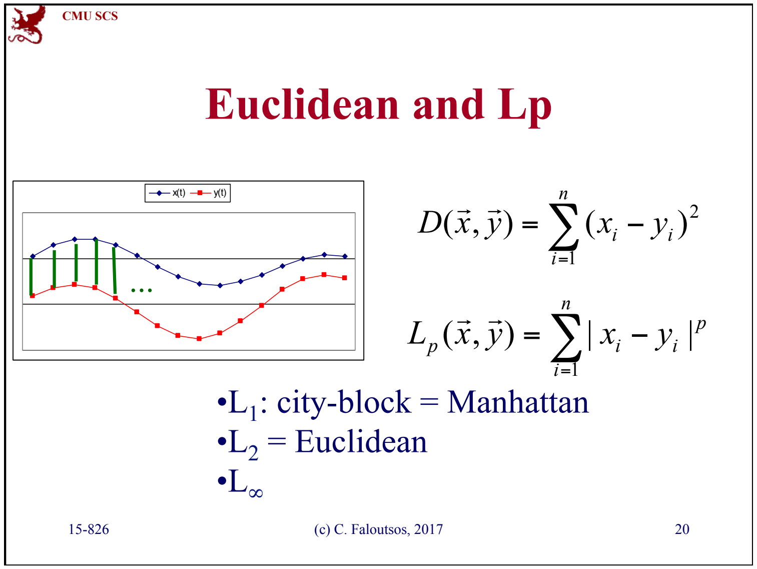
Subtle, but **absolutely necessary**:

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

Two major families

- Euclidean and Lp norms
- Time warping and variations

15-826 (c) C. Faloutsos, 2017 19



CMU SCS

Euclidean and Lp

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

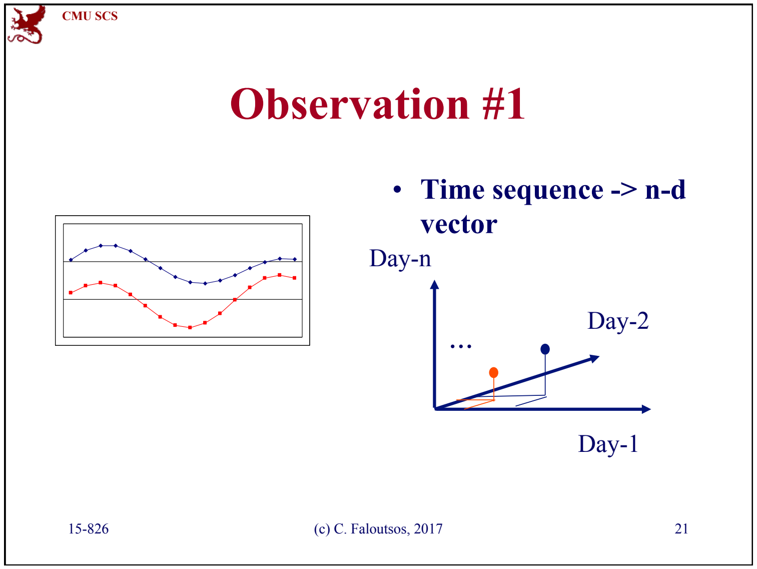
$$L_p(\vec{x}, \vec{y}) = \sum_{i=1}^{n} | x_i - y_i |^p$$

•L_1: city-block = Manhattan

•L_2 = Euclidean

•L_∞

15-826 (c) C. Faloutsos, 2017 20

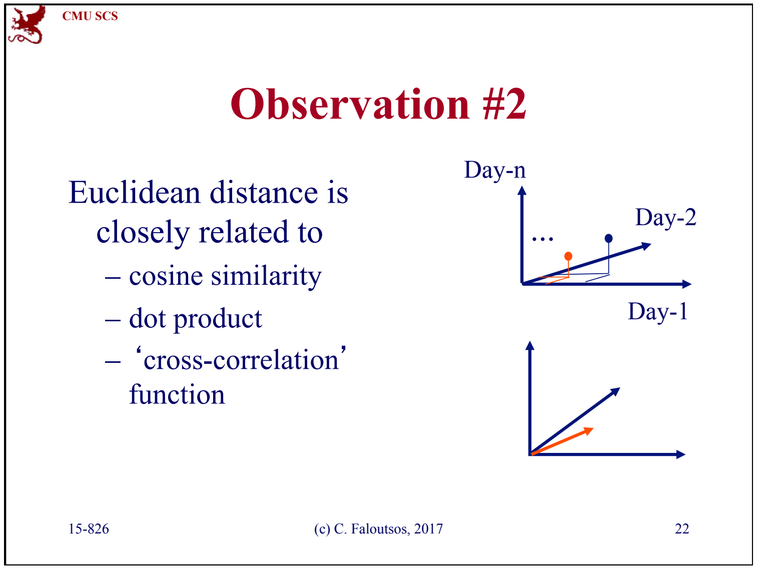


CMU SCS

Observation #1

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

15-826 (c) C. Faloutsos, 2017 21



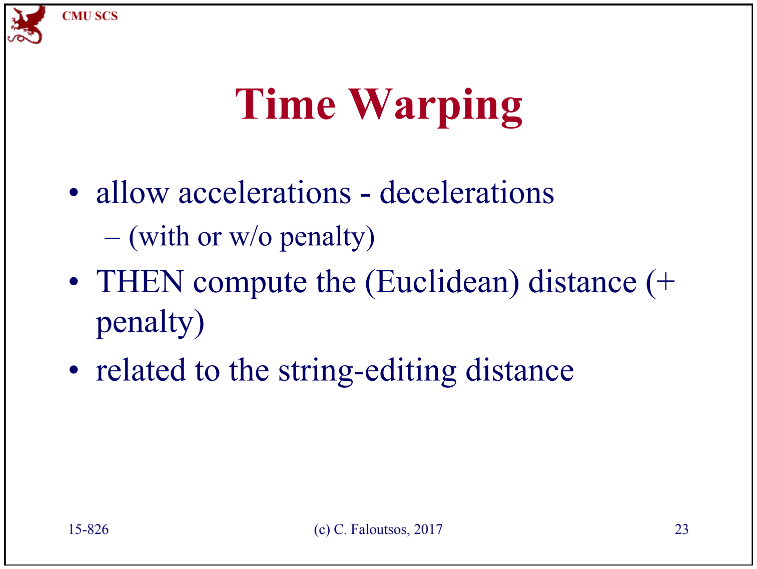
CMU SCS

Observation #2

Euclidean distance is closely related to

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

15-826 (c) C. Faloutsos, 2017 22

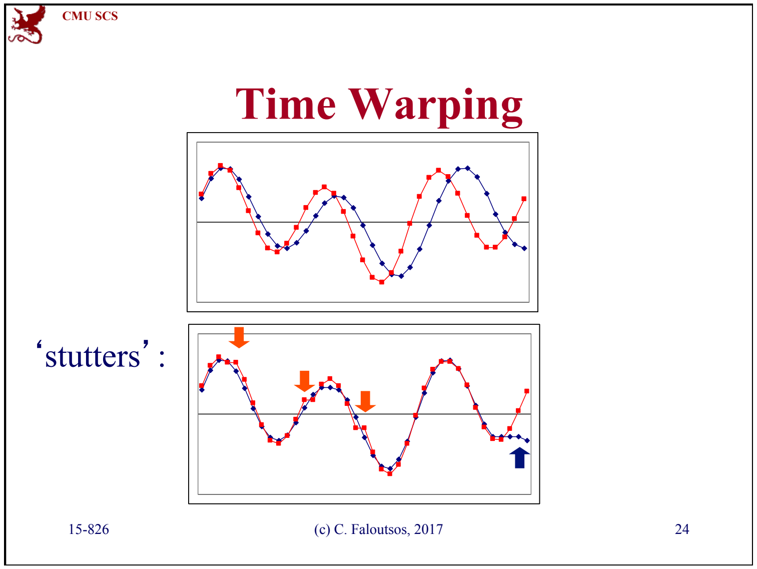


CMU SCS

Time Warping

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

15-826 (c) C. Faloutsos, 2017 23

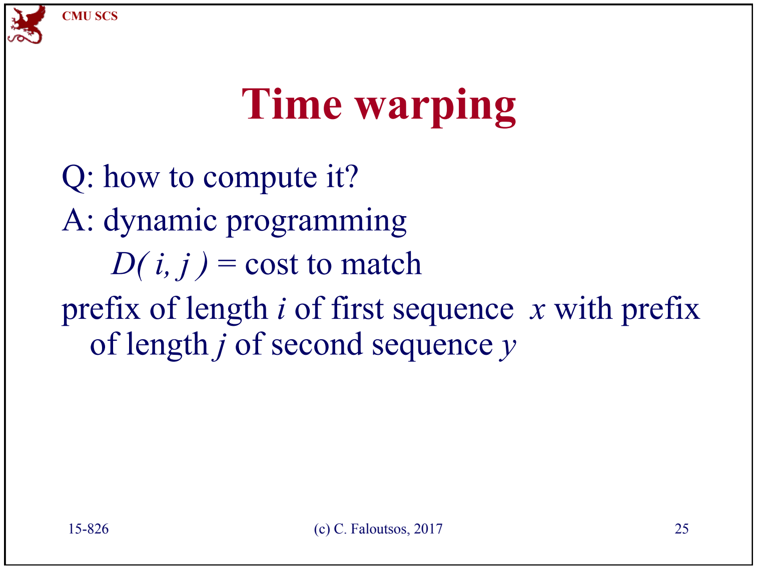


CMU SCS

Time Warping

‘stutters’ :

15-826 (c) C. Faloutsos, 2017 24



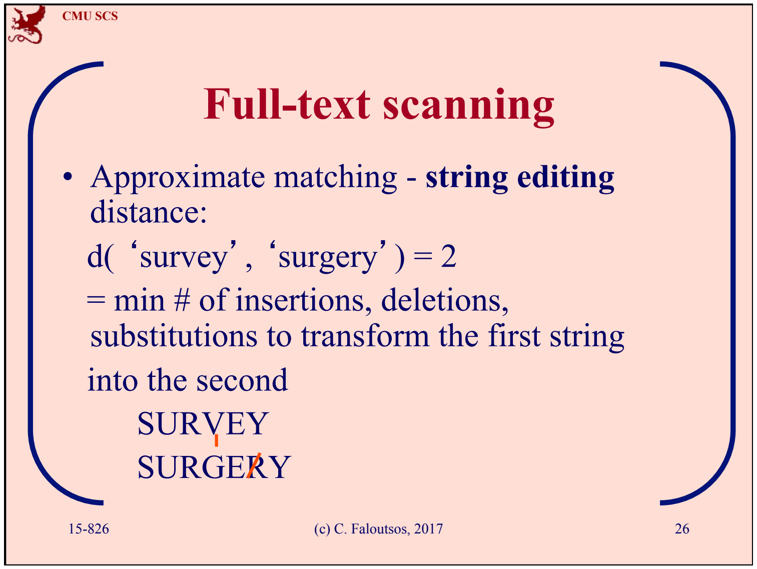
Time warping

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*



Full-text scanning

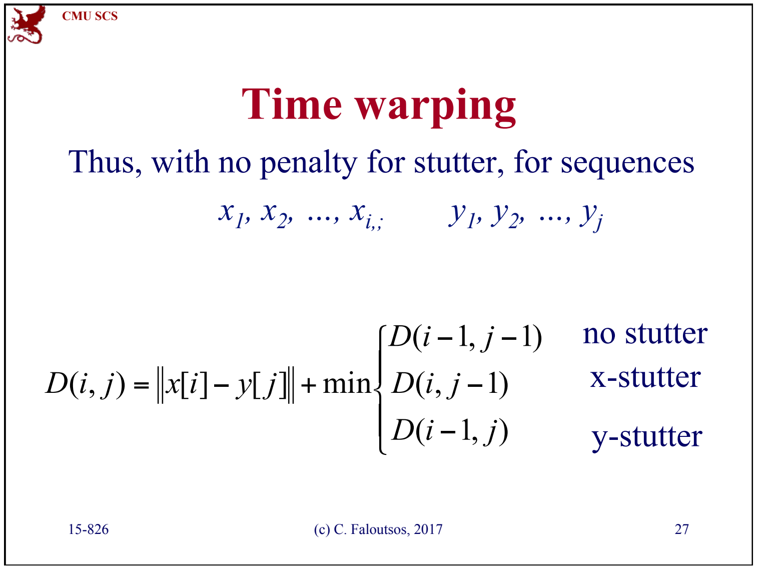
- Approximate matching - **string editing** distance:

d(‘survey’, ‘surgery’) = 2

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

SURVEY

SURGERY

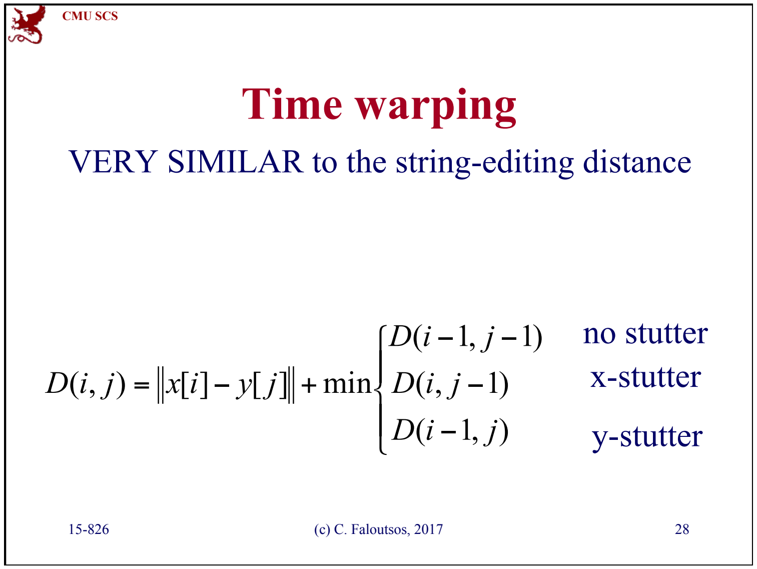


Time warping

Thus, with no penalty for stutter, for sequences

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

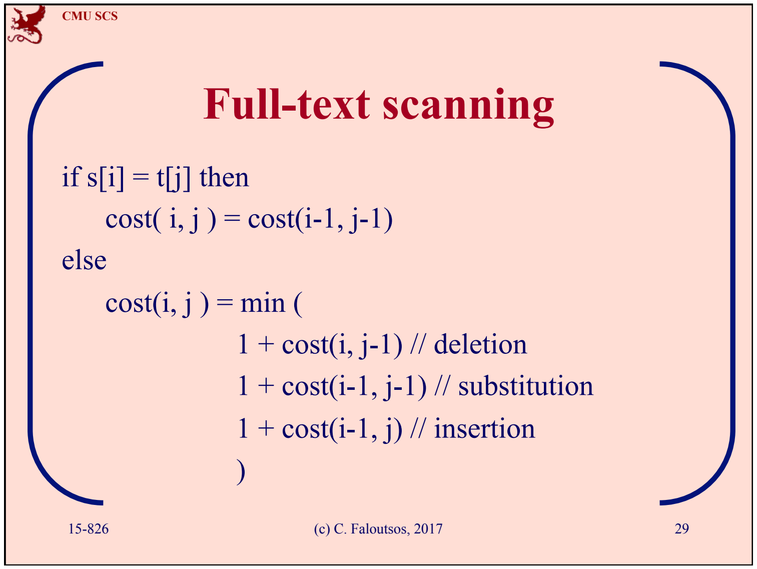
$$D(i,j) = \|x[i]-y[j]\| + \min\begin{cases} D(i-1,j-1) & \text{no stutter} \\ D(i,j-1) & \text{x-stutter} \\ D(i-1,j) & \text{y-stutter} \end{cases}$$



Time warping

VERY SIMILAR to the string-editing distance

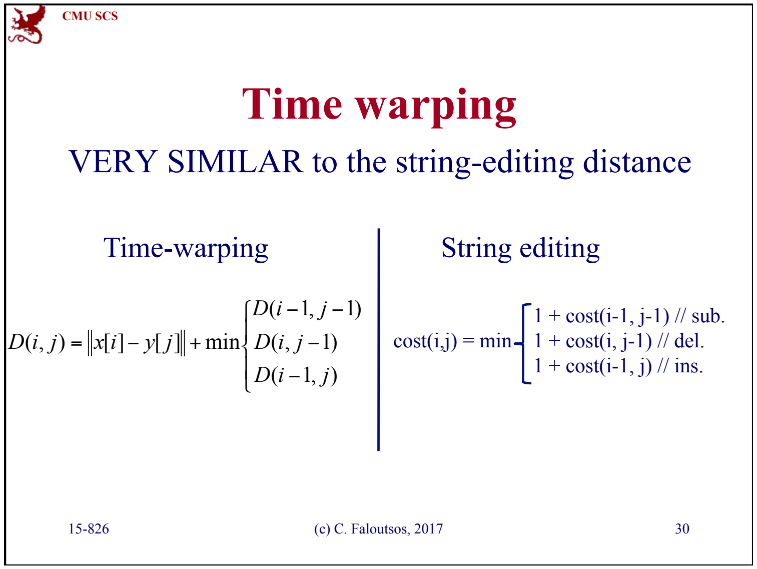
$$D(i,j) = \|x[i]-y[j]\| + \min\begin{cases} D(i-1,j-1) & \text{no stutter} \\ D(i,j-1) & \text{x-stutter} \\ D(i-1,j) & \text{y-stutter} \end{cases}$$



CMU SCS

Full-text scanning

```
if s[i] = t[j] then
    cost( i, j ) = cost(i-1, j-1)
else
    cost(i, j ) = min (
            1 + cost(i, j-1) // deletion
            1 + cost(i-1, j-1) // substitution
            1 + cost(i-1, j) // insertion
            )
```



CMU SCS

Time warping

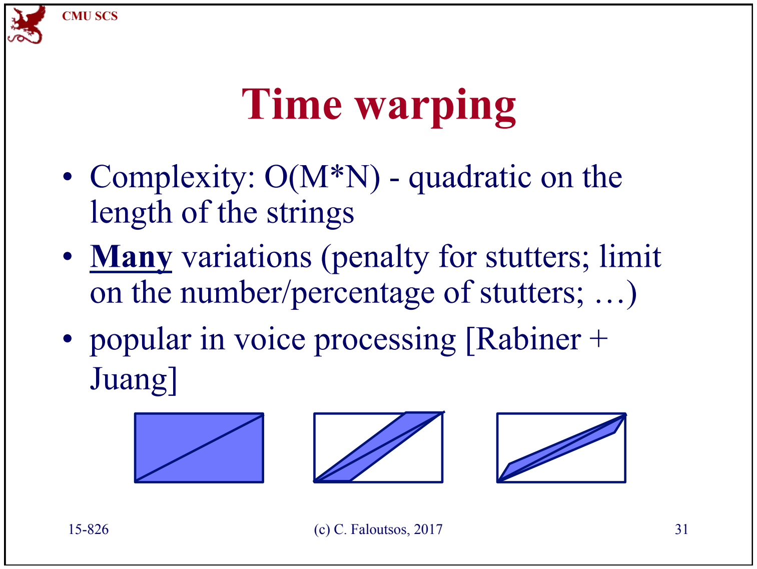
VERY SIMILAR to the string-editing distance

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

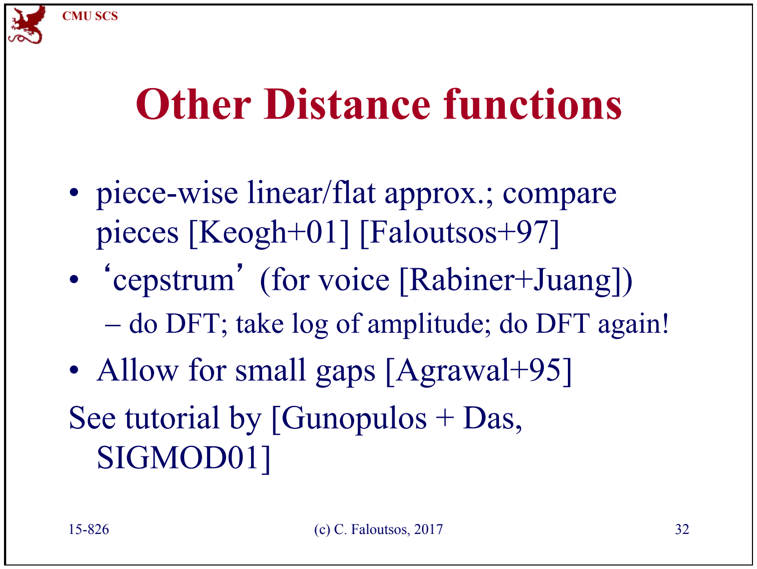
$$\text{cost(i,j)} = \min \begin{cases} \text{1 + cost(i-1, j-1) // sub.} \\ \text{1 + cost(i, j-1) // del.} \\ \text{1 + cost(i-1, j) // ins.} \end{cases}$$



CMU SCS

Time warping

- Complexity: O(M*N) - quadratic on the length of the strings
- **Many** variations (penalty for stutters; limit on the number/percentage of stutters; …)
- popular in voice processing [Rabiner + Juang]

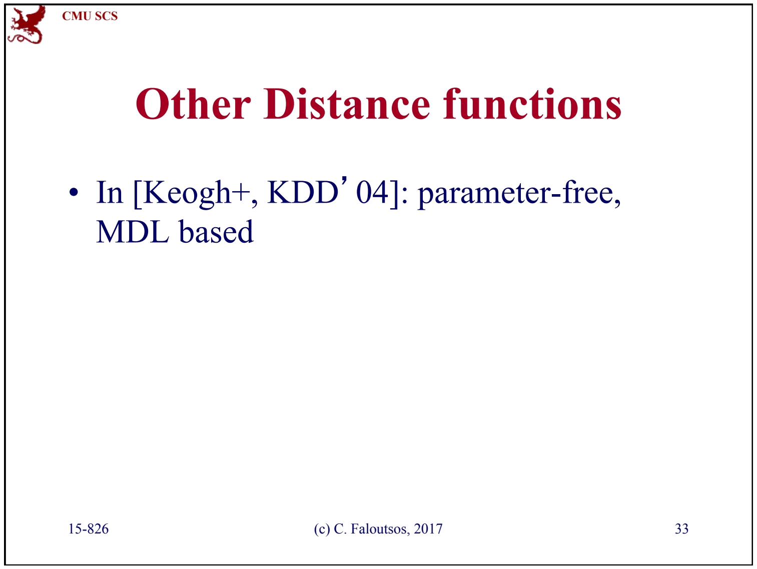


CMU SCS

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]

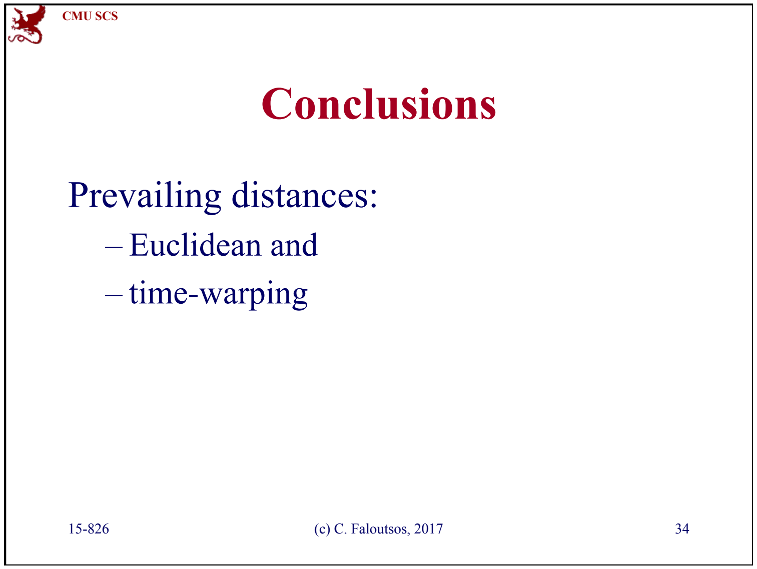
See tutorial by [Gunopulos + Das, SIGMOD01]



CMU SCS

Other Distance functions

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

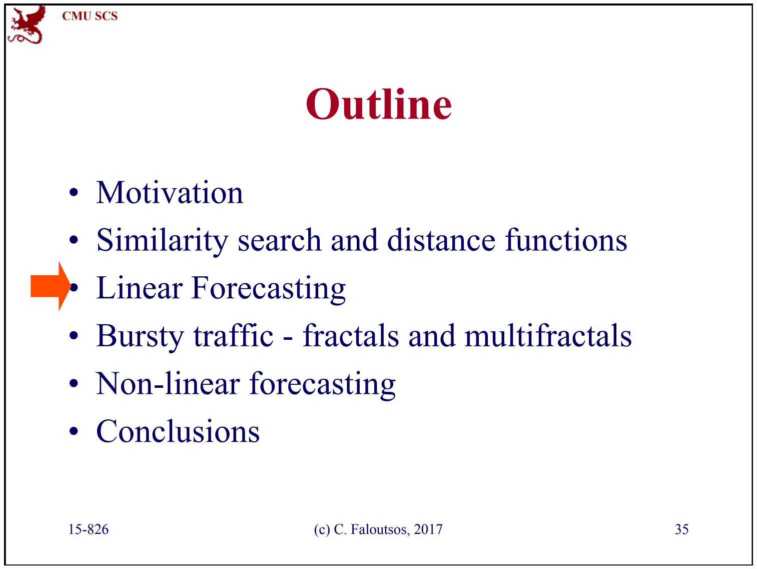


CMU SCS

Conclusions

Prevailing distances:

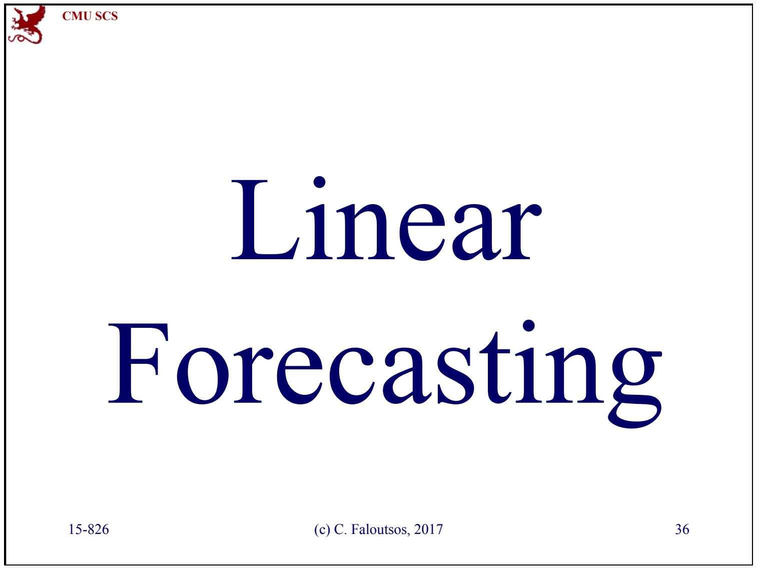
- Euclidean and
- time-warping



CMU SCS

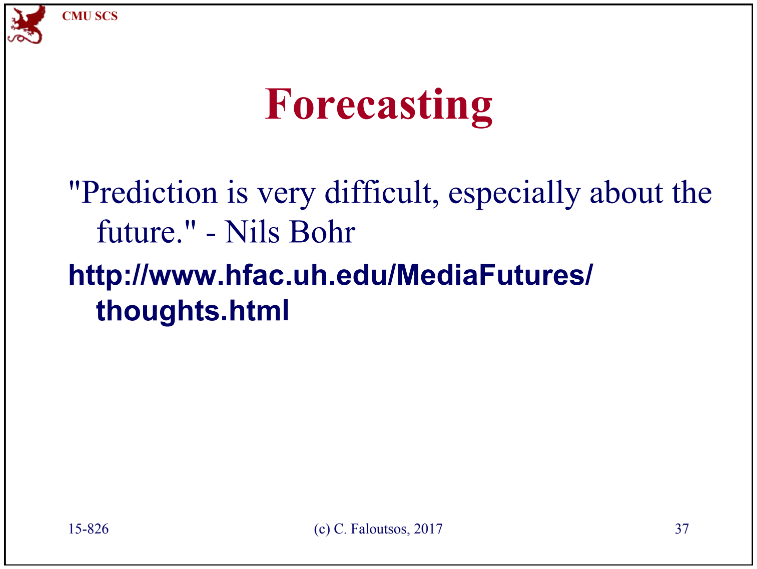
Outline

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



CMU SCS

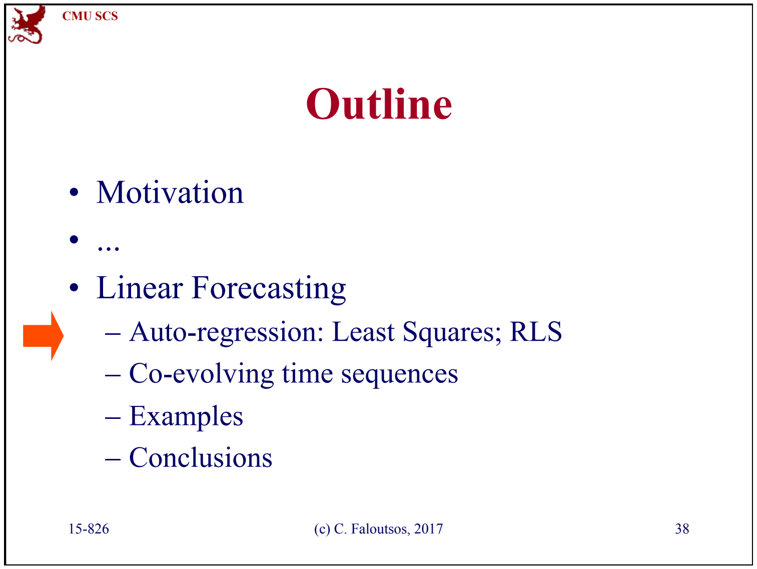
Linear Forecasting



Forecasting

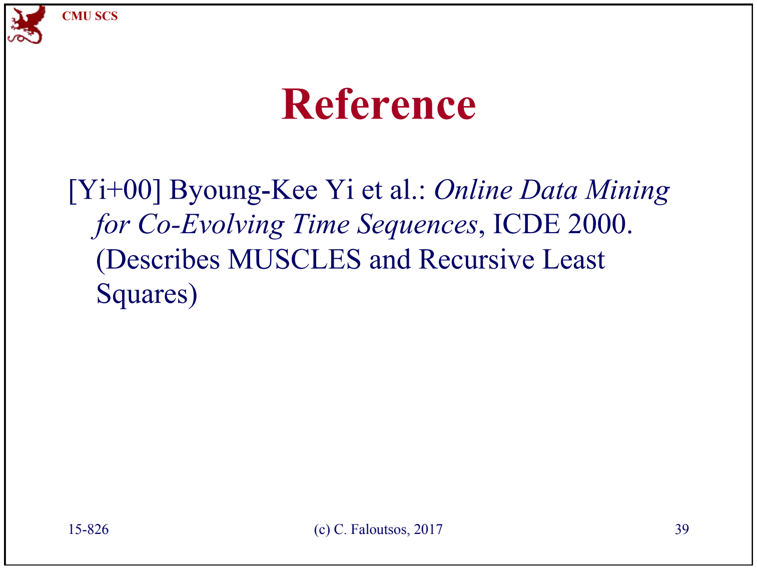
"Prediction is very difficult, especially about the future." - Nils Bohr

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



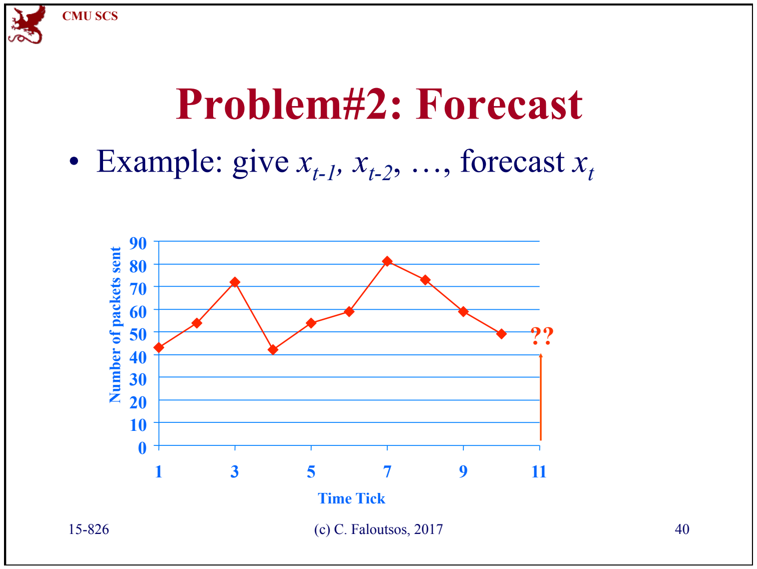
Outline

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



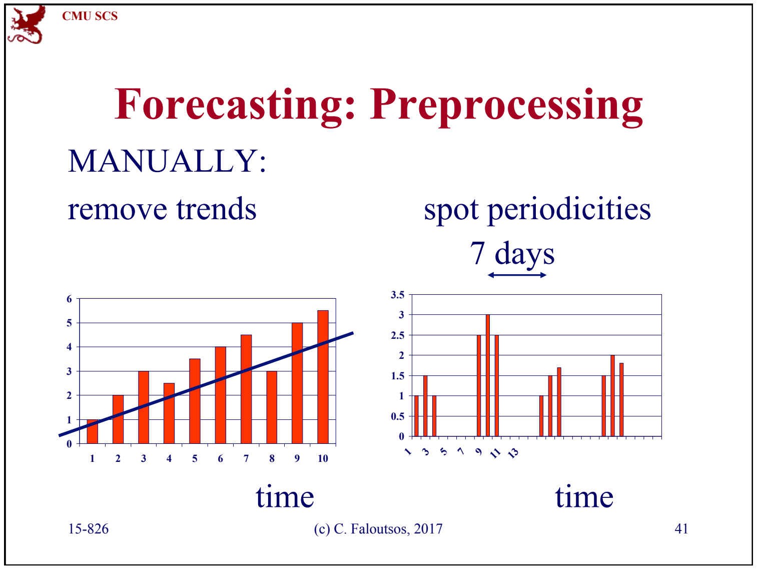
Reference

[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}, …, forecast x_t



CMU SCS

Forecasting: Preprocessing

MANUALLY:

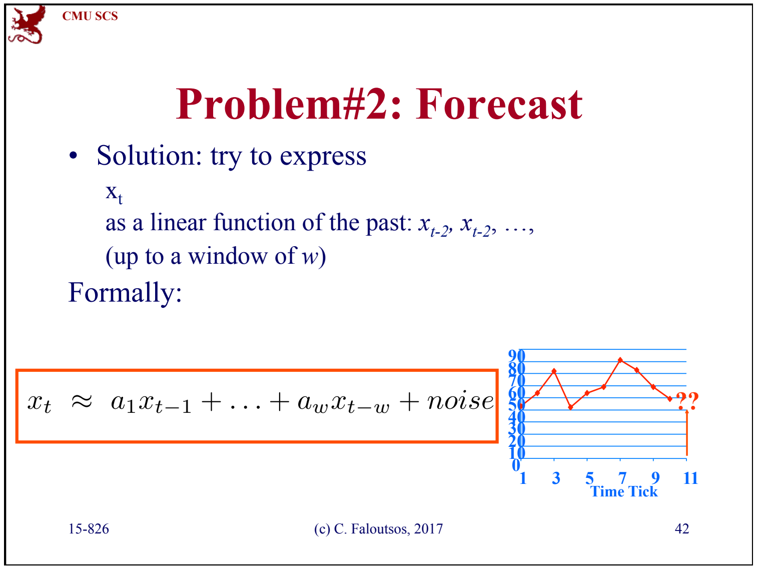
remove trends

spot periodicities

7 days

time

time



CMU SCS

Problem#2: Forecast

- Solution: try to express

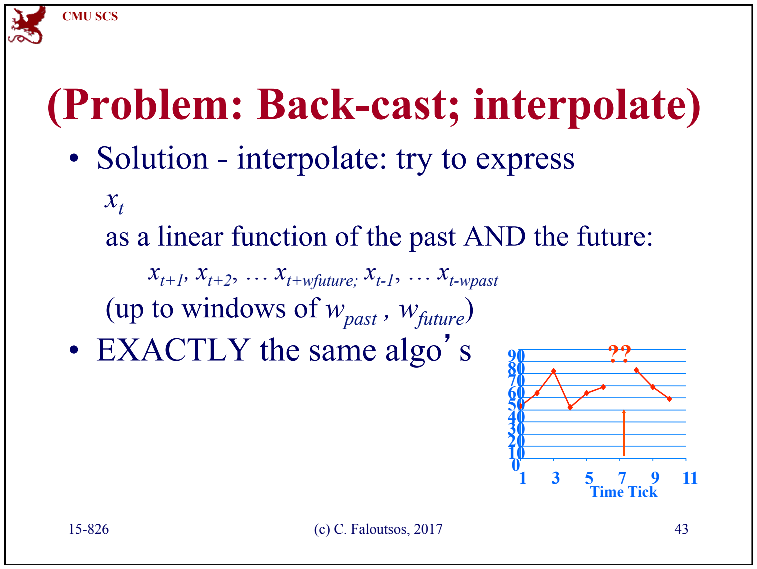
  x_t

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

  (up to a window of w)

Formally:

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



CMU SCS

(Problem: Back-cast; interpolate)

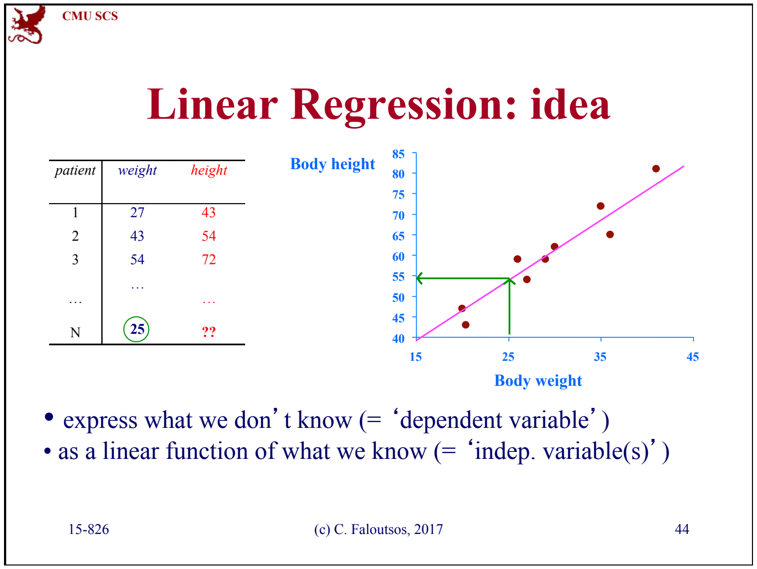
- Solution - interpolate: try to express

  x_t

  as a linear function of the past AND the future:

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

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

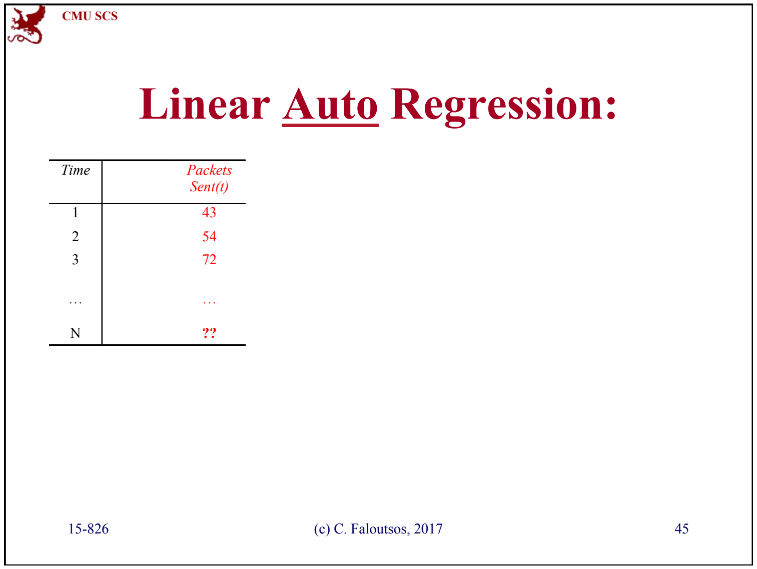


CMU SCS

Linear Regression: idea

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

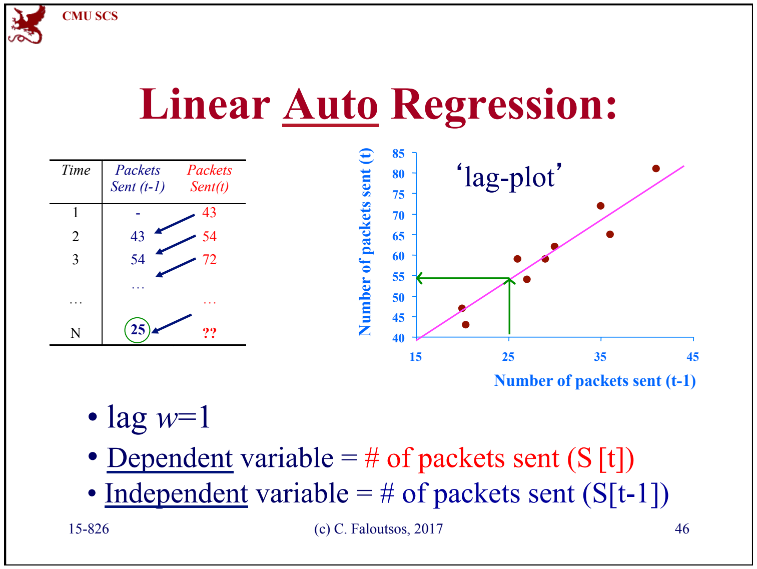
- express what we don’ t know (= ‘dependent variable’)
- as a linear function of what we know (= ‘indep. variable(s)’)



CMU SCS

Linear Auto Regression:

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

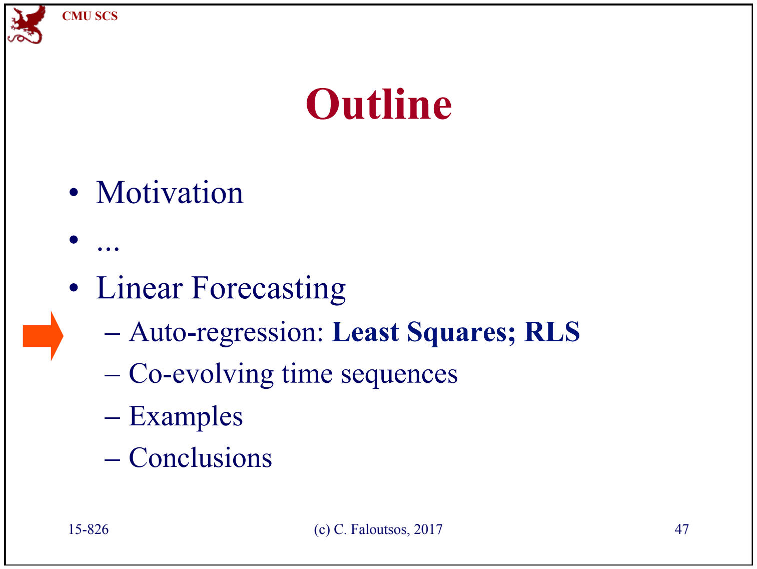


CMU SCS

Linear Auto Regression:

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

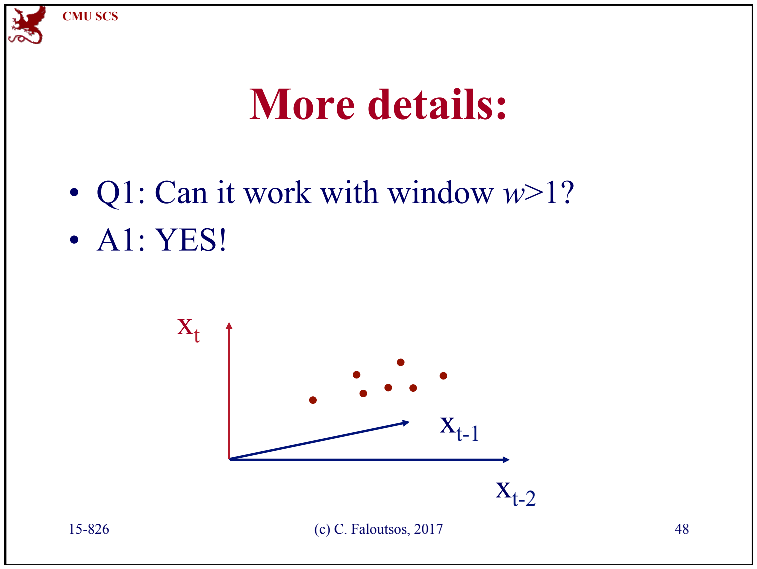
- lag *w*=1
- Dependent variable = # of packets sent (S [t])
- Independent variable = # of packets sent (S[t-1])



CMU SCS

Outline

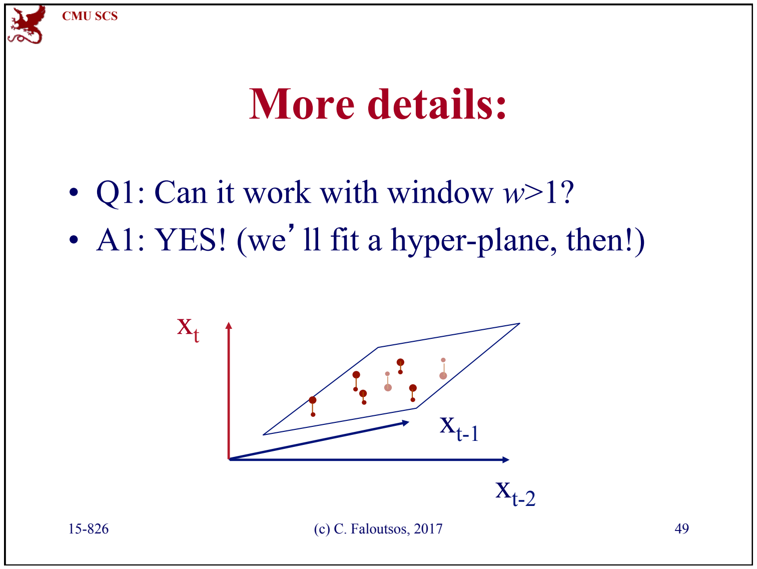
- Motivation
- ...
- Linear Forecasting
  - Auto-regression: **Least Squares; RLS**
  - Co-evolving time sequences
  - Examples
  - Conclusions



CMU SCS

More details:

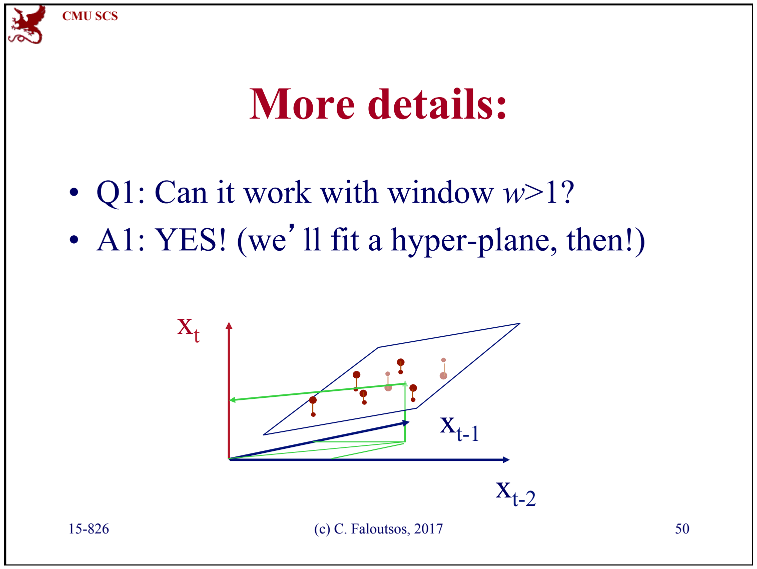
- Q1: Can it work with window *w*>1?
- A1: YES!



CMU SCS

More details:

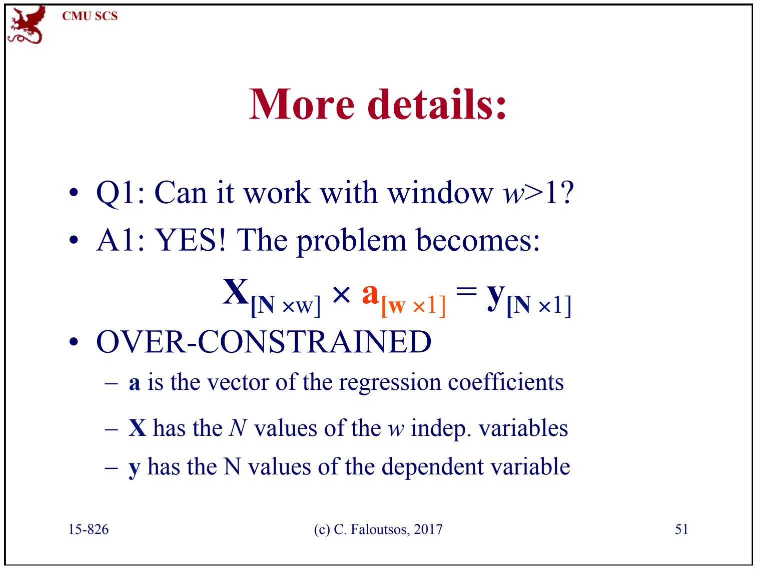
- Q1: Can it work with window *w*>1?
- A1: YES! (we' ll fit a hyper-plane, then!)



CMU SCS

More details:

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



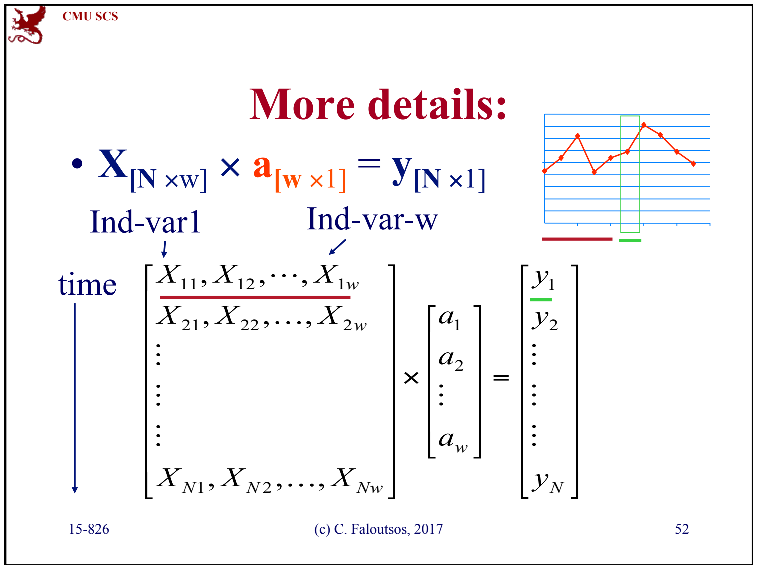
CMU SCS

More details:

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

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

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

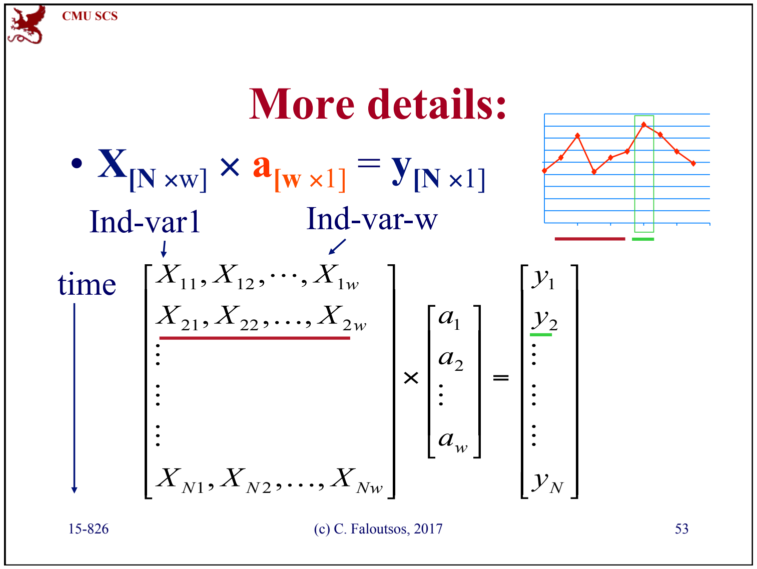


CMU SCS

More details:

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

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



CMU SCS

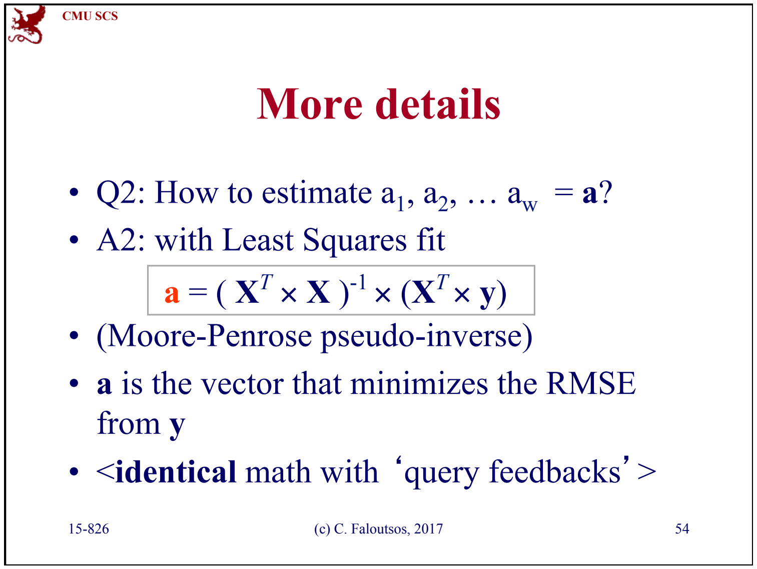
More details:

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

Ind-var1 Ind-var-w

time

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



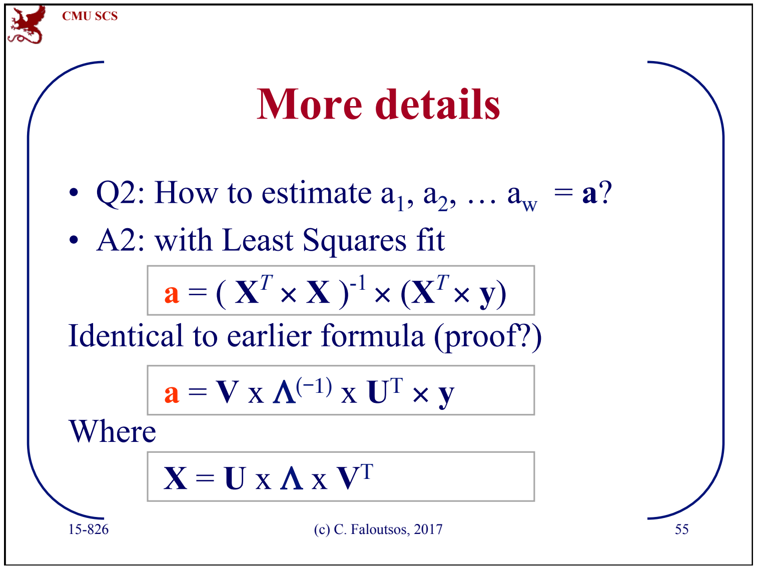
CMU SCS

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**
- <**identical** math with 'query feedbacks'>



CMU SCS

More details

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

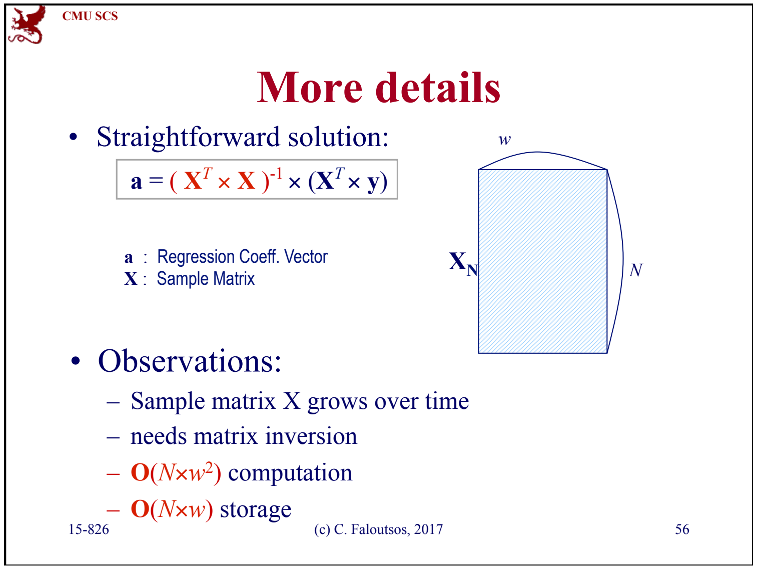
$$\boxed{\mathbf{a} = (\mathbf{X}^T \times \mathbf{X})^{-1} \times (\mathbf{X}^T \times \mathbf{y})}$$

Identical to earlier formula (proof?)

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

Where

$$\boxed{\mathbf{X} = \mathbf{U} \text{ x } \mathbf{\Lambda} \text{ x } \mathbf{V}^T}$$



CMU SCS

More details

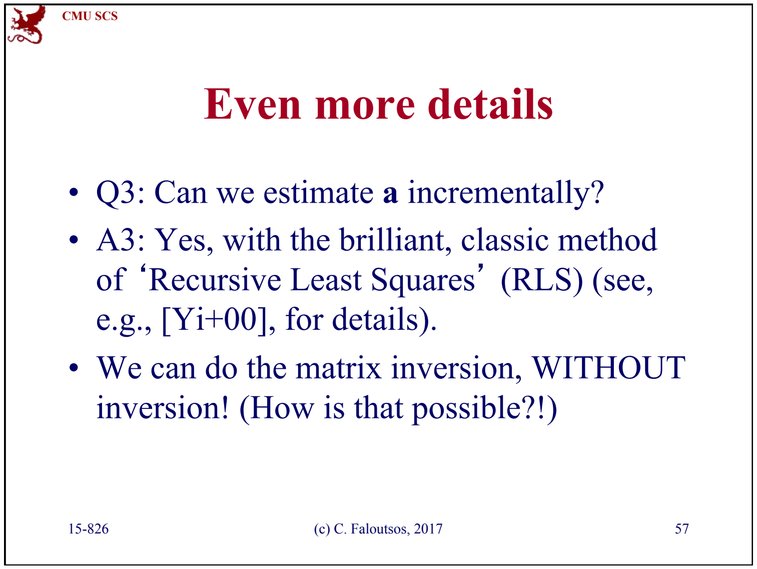
- Straightforward solution:

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

a : Regression Coeff. Vector
X : Sample Matrix

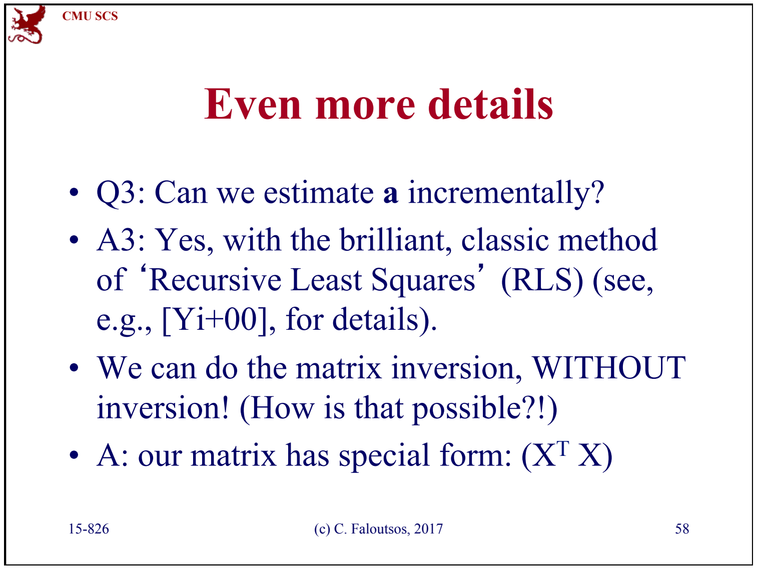
w
$\mathbf{X}_N$
N

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


CMU SCS

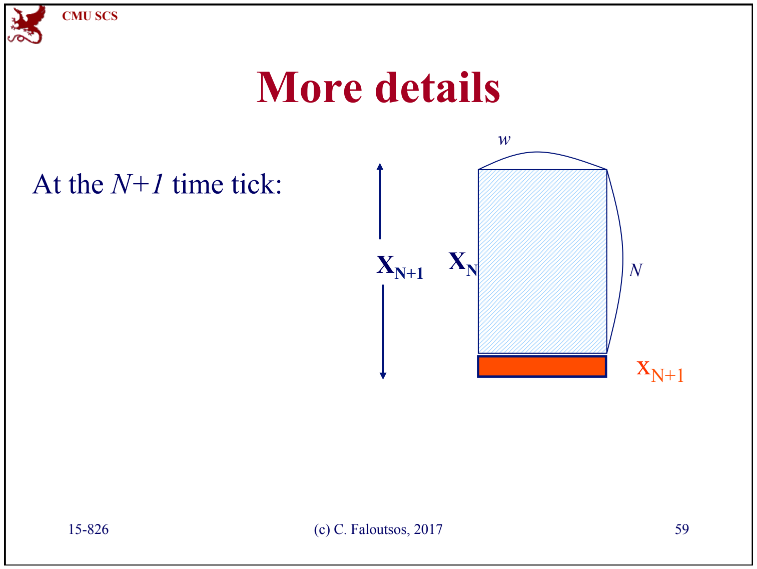
Even more details

- 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).
- We can do the matrix inversion, WITHOUT inversion! (How is that possible?!)


CMU SCS

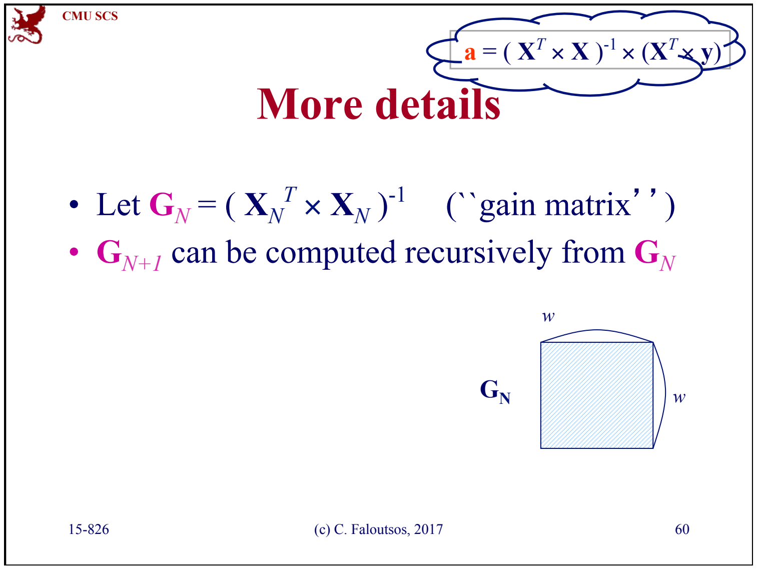
Even more details

- 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).
- We can do the matrix inversion, WITHOUT inversion! (How is that possible?!)
- A: our matrix has special form: $(X^T X)$


CMU SCS

More details

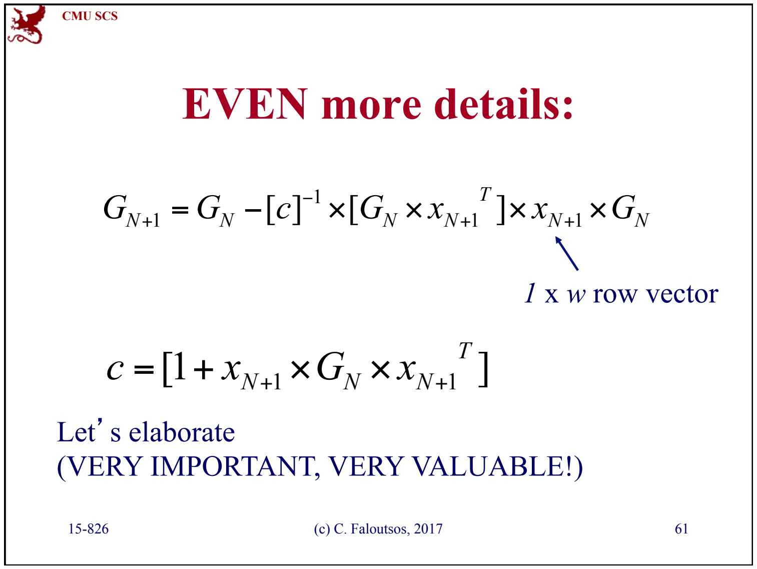
At the *N+1* time tick:


CMU SCS

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

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 $\mathbf{G}_N$



CMU SCS

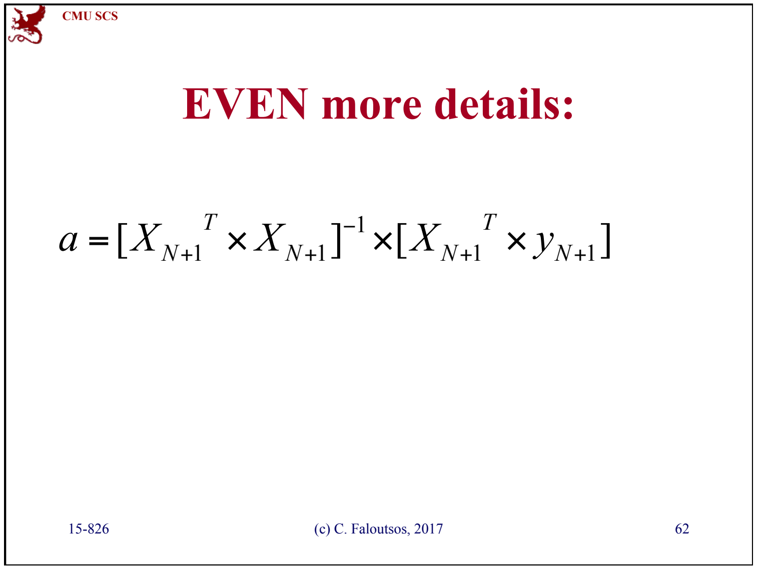
EVEN more details:

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

1 x *w* row vector

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

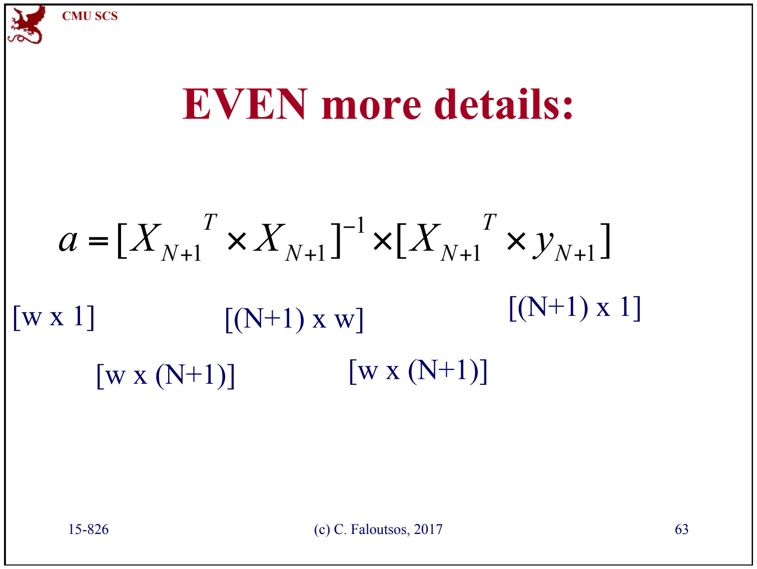
Let’s elaborate
(VERY IMPORTANT, VERY VALUABLE!)



CMU SCS

EVEN more details:

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



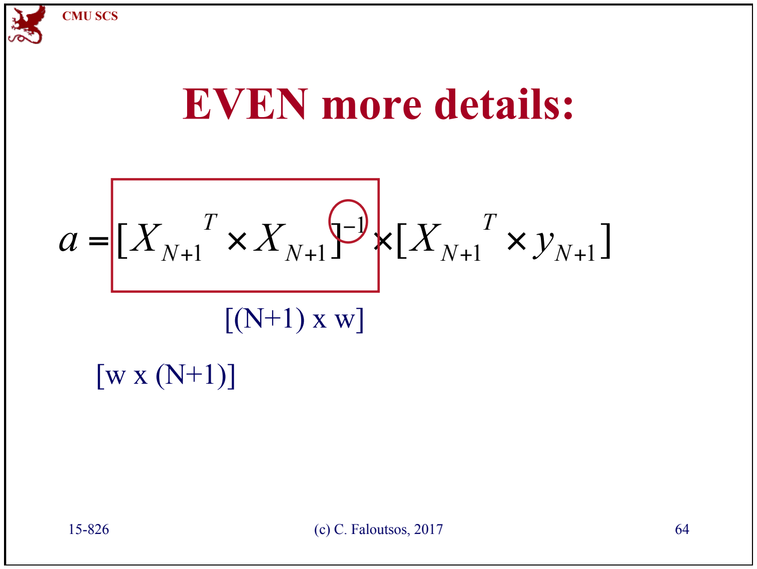
CMU SCS

EVEN more details:

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

[w x 1] [(N+1) x w] [(N+1) x 1]

[w x (N+1)] [w x (N+1)]



CMU SCS

EVEN more details:

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

[(N+1) x w]

[w x (N+1)]

CMU SCS

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

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

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

15-826 (c) C. Faloutsos, 2017 65

CMU SCS

EVEN more details:

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

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

15-826 (c) C. Faloutsos, 2017 66

CMU SCS

EVEN more details:

$$\overset{wxw}{G_{N+1}} = \overset{wxw}{G_N} - \overset{1x1}{[c]^{-1}} \times [\overset{wxw}{G_N} \times \overset{wx1}{x_{N+1}^T}] \times \overset{1xw}{x_{N+1}} \times \overset{wxw}{G_N}$$

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

15-826 (c) C. Faloutsos, 2017 67

CMU SCS

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

Altogether:

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

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

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

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

15-826 (c) C. Faloutsos, 2017 68

Altogether:

$G_0 \equiv \delta I$ **IMPORTANT!**

where
I: w x w identity matrix
δ: a large positive number (say, 10^4)

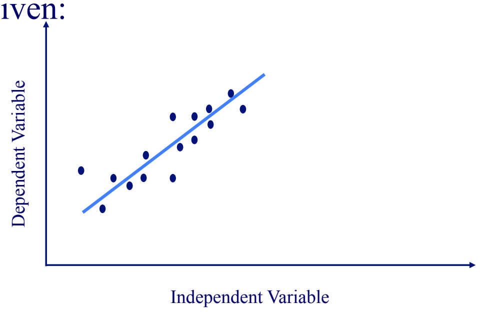
Comparison:

- 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**

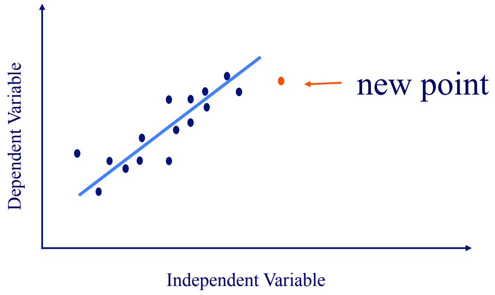
$\boxed{N = 10^6, \quad w = 1\text{-}100}$

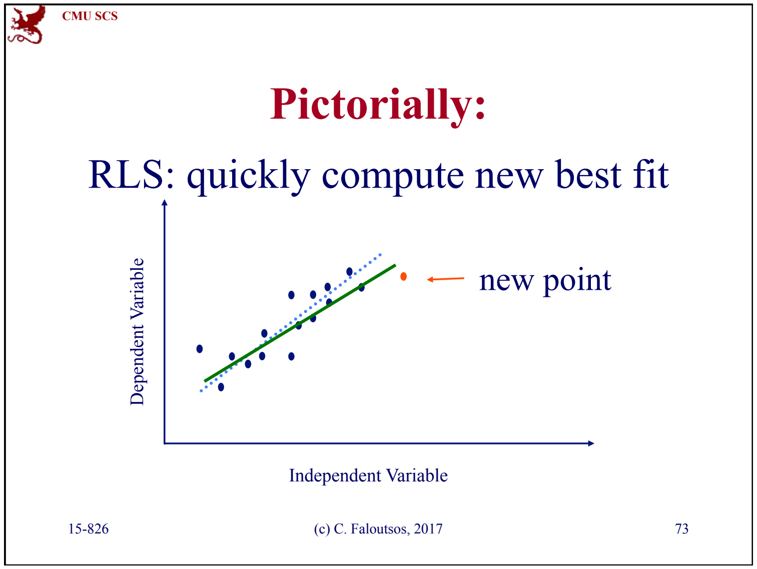
Pictorially:

- Given:



Pictorially:



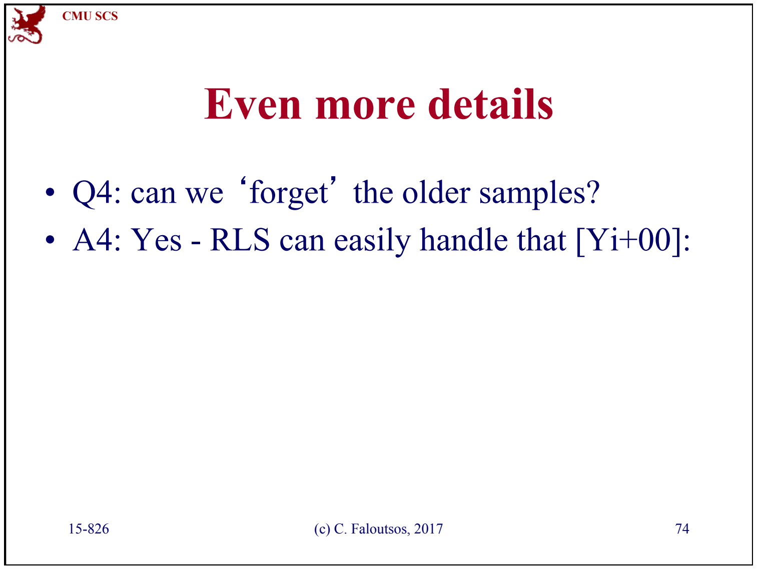


CMU SCS

Pictorially:

RLS: quickly compute new best fit

15-826 (c) C. Faloutsos, 2017 73

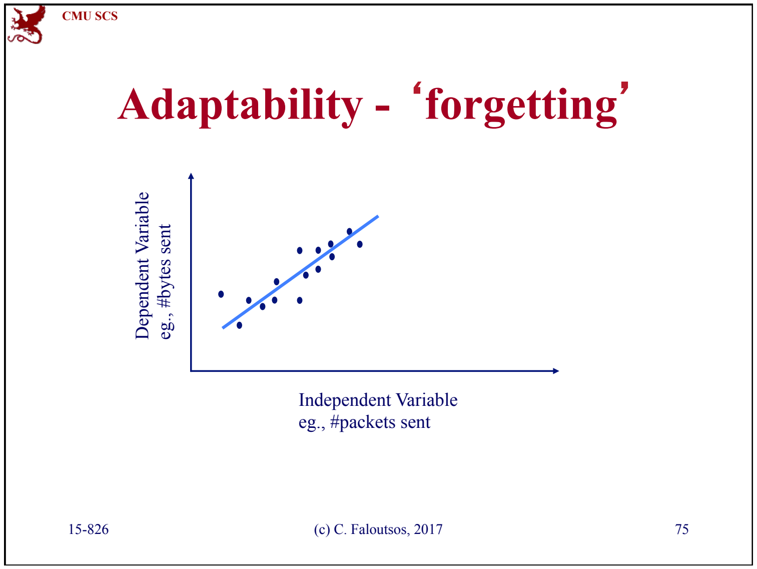


CMU SCS

Even more details

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

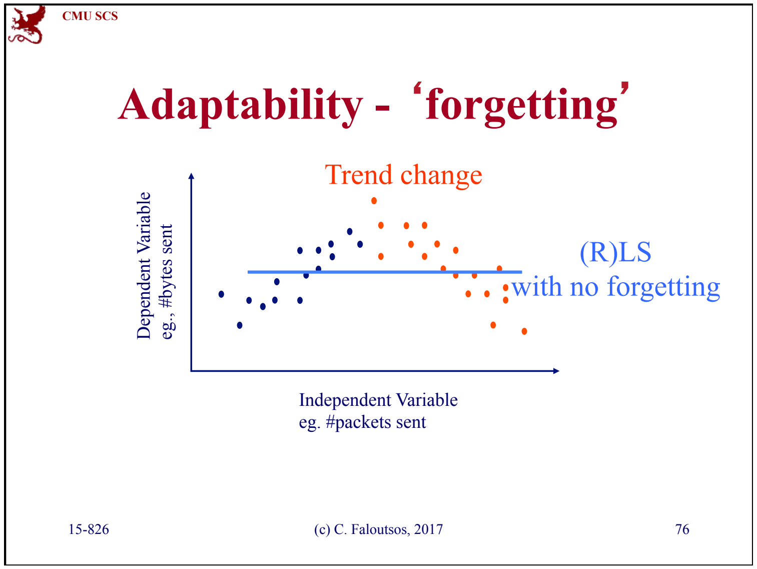
15-826 (c) C. Faloutsos, 2017 74



CMU SCS

Adaptability - ‘forgetting’

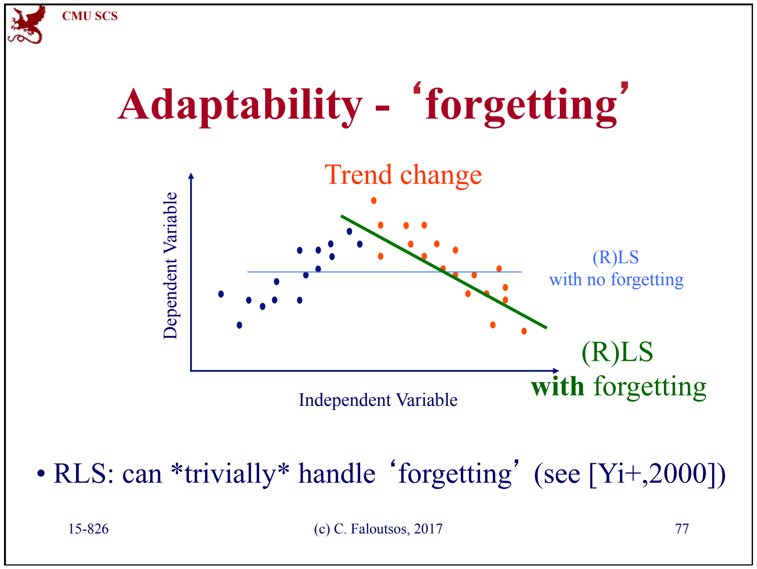
15-826 (c) C. Faloutsos, 2017 75



CMU SCS

Adaptability - ‘forgetting’

15-826 (c) C. Faloutsos, 2017 76

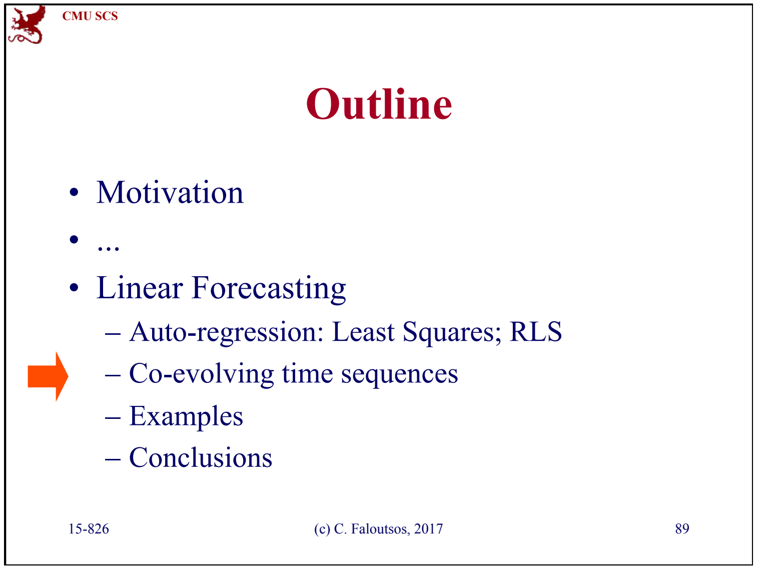


CMU SCS

Adaptability - ‘forgetting’

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

15-826 (c) C. Faloutsos, 2017 77

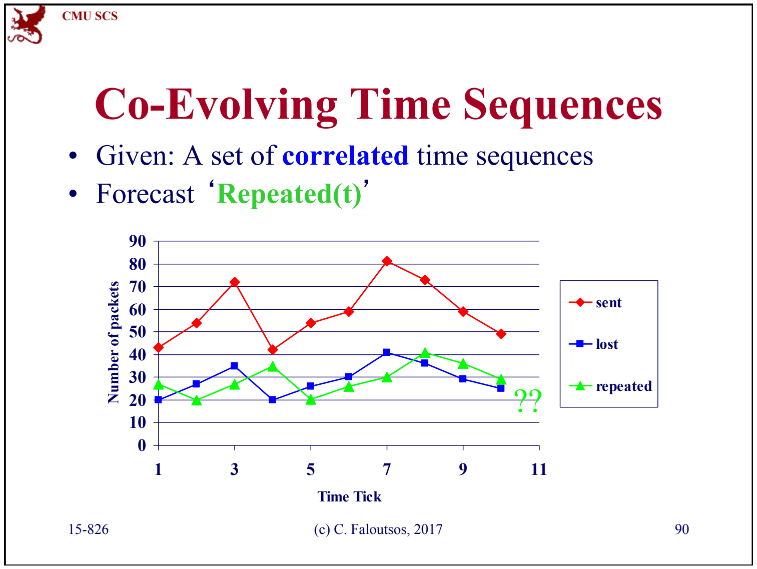


CMU SCS

Outline

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

15-826 (c) C. Faloutsos, 2017 89

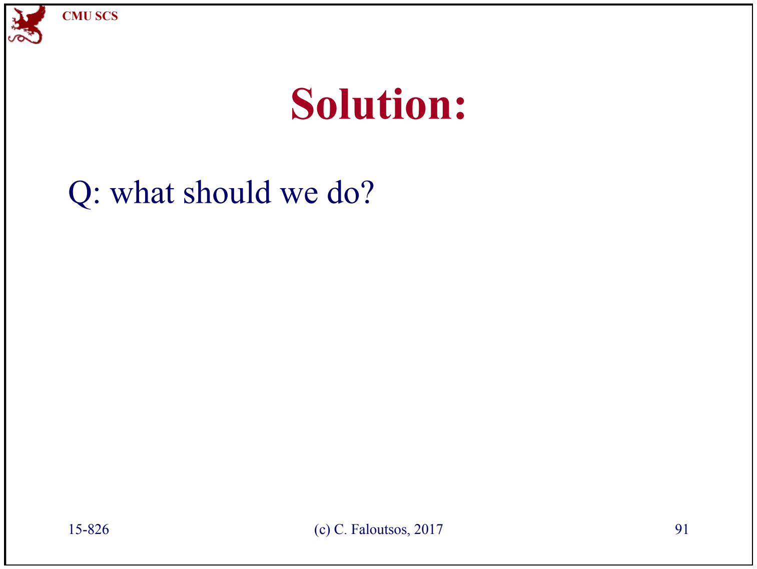


CMU SCS

Co-Evolving Time Sequences

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

15-826 (c) C. Faloutsos, 2017 90



CMU SCS

Solution:

Q: what should we do?

15-826 (c) C. Faloutsos, 2017 91

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])

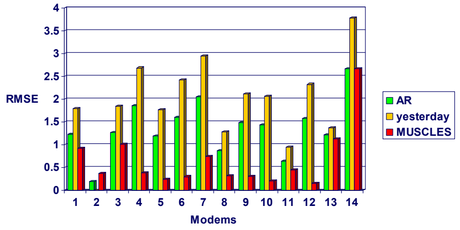
Forecasting - Outline

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

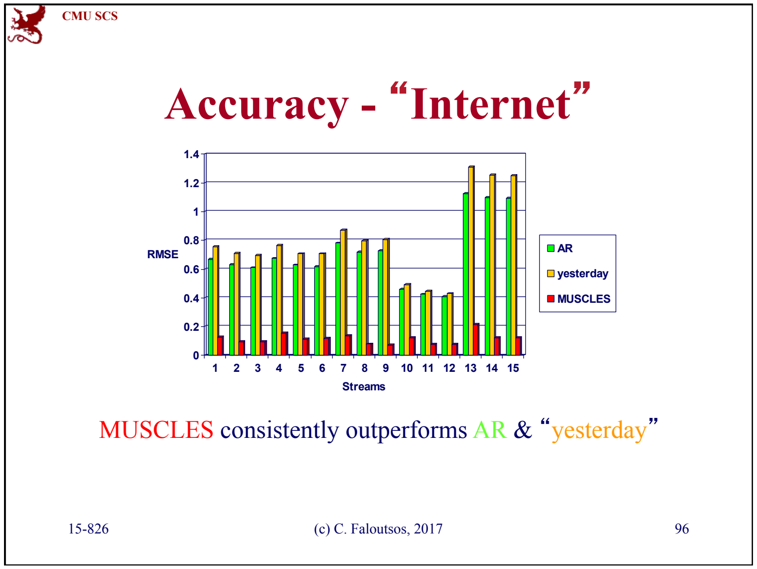
Examples - Experiments

- 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”

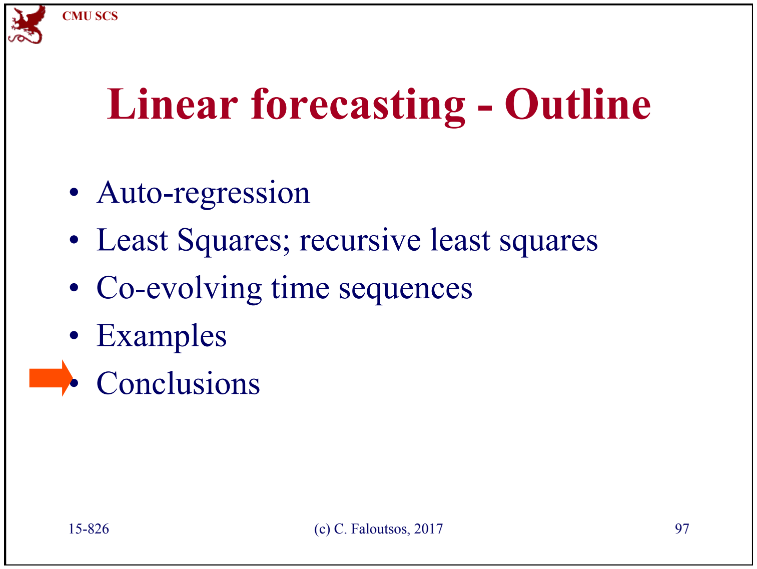


MUSCLES outperforms AR & “yesterday”



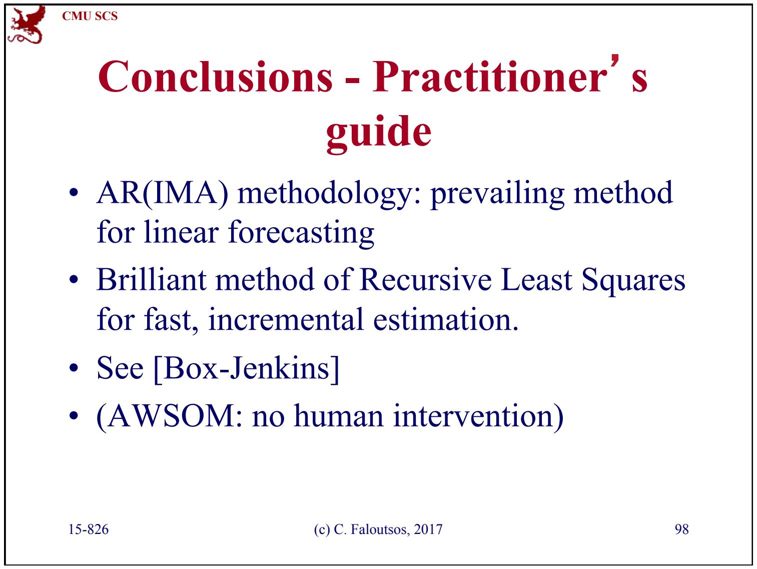
Accuracy - “Internet”

MUSCLES consistently outperforms AR & “yesterday”



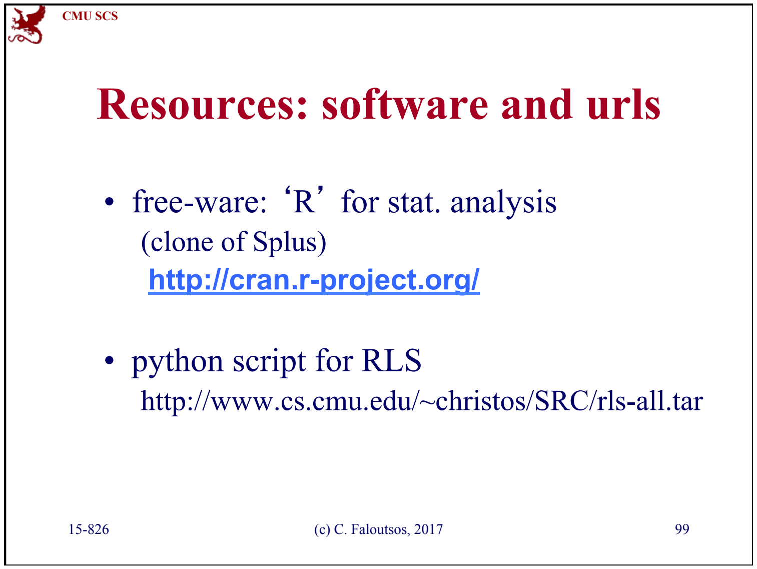
Linear forecasting - Outline

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



Conclusions - Practitioner’ s guide

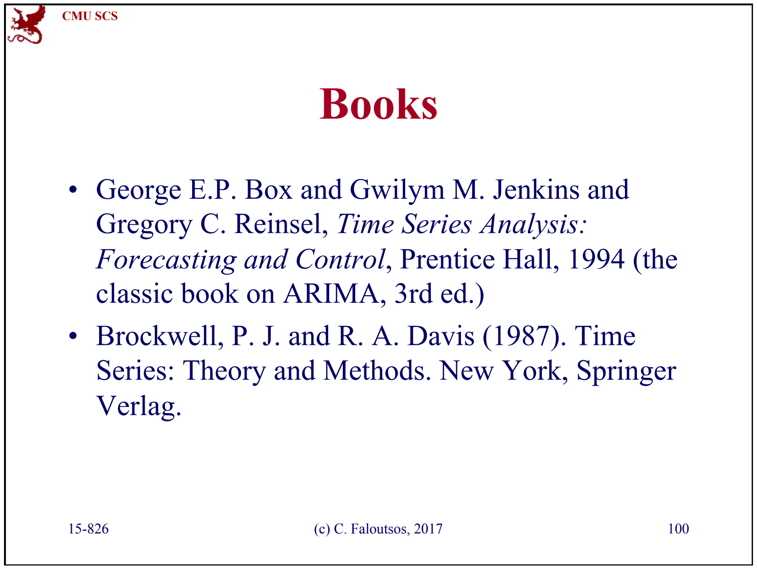
- AR(IMA) methodology: prevailing method for linear forecasting
- Brilliant method of Recursive Least Squares for fast, incremental estimation.
- See [Box-Jenkins]
- (AWSOM: no human intervention)



Resources: software and urls

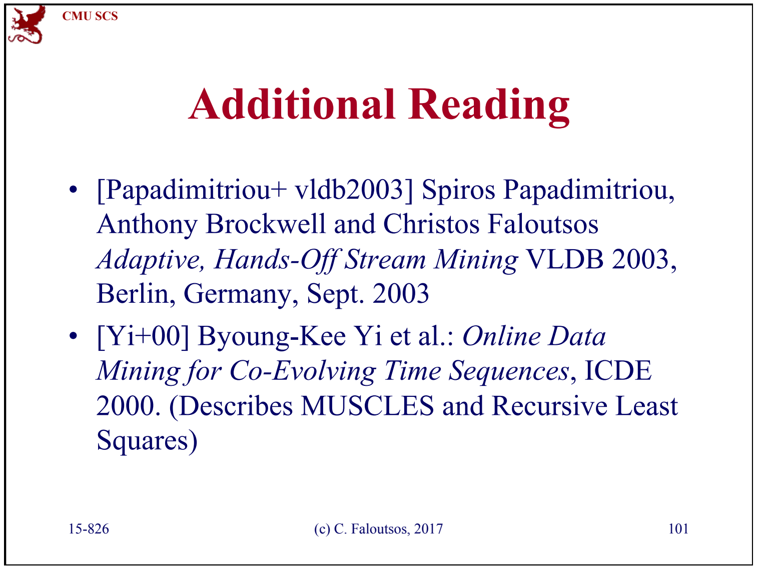
- free-ware: ‘R’ for stat. analysis
  (clone of Splus)
  http://cran.r-project.org/

- python script for RLS
  http://www.cs.cmu.edu/~christos/SRC/rls-all.tar



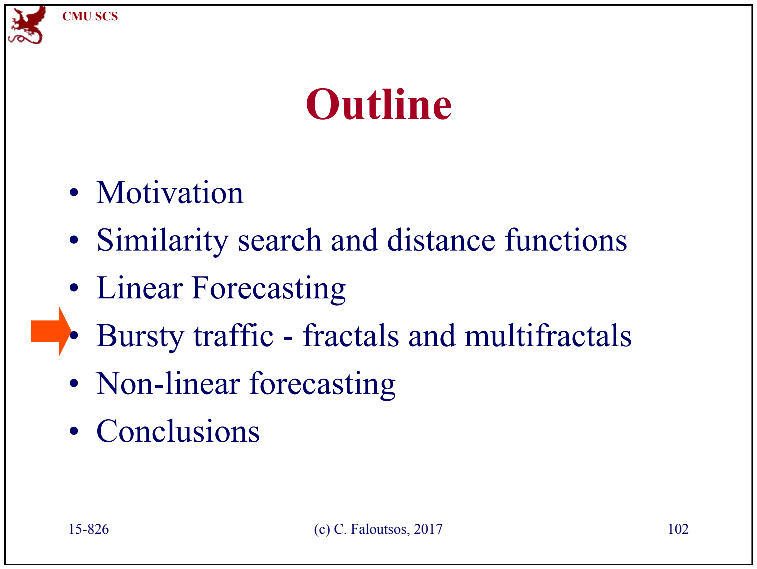
Books

- 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

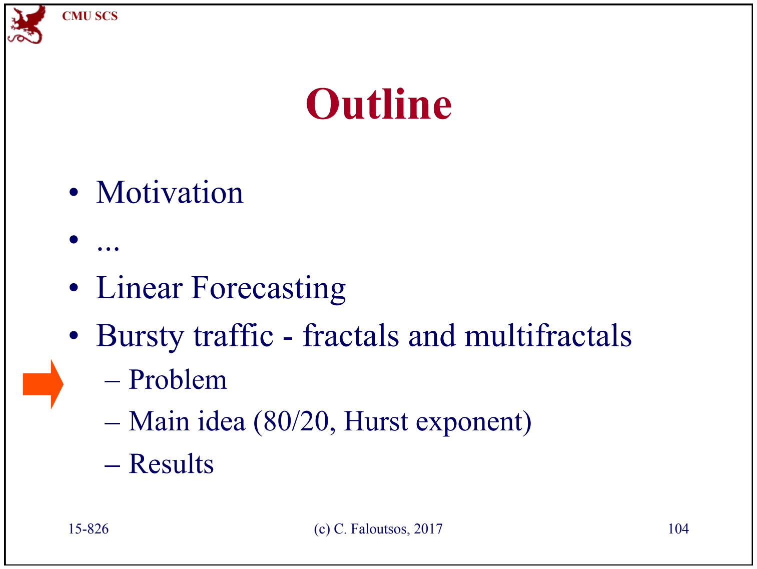
- [Papadimitriou+ vldb2003] Spiros Papadimitriou, Anthony Brockwell and Christos Faloutsos *Adaptive, Hands-Off Stream Mining* VLDB 2003, Berlin, Germany, Sept. 2003
- [Yi+00] Byoung-Kee Yi et al.: *Online Data Mining for Co-Evolving Time Sequences*, ICDE 2000. (Describes MUSCLES and Recursive Least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



Outline

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

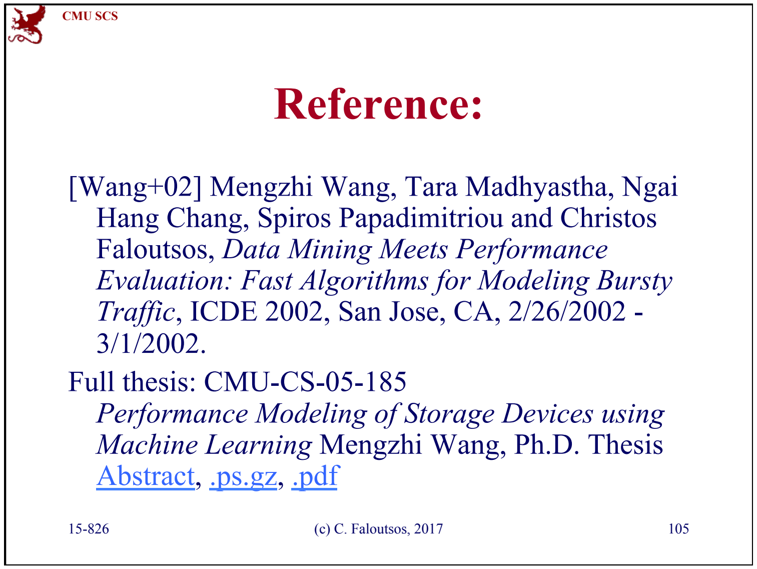
Bursty Traffic & Multifractals



CMU SCS

Outline

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

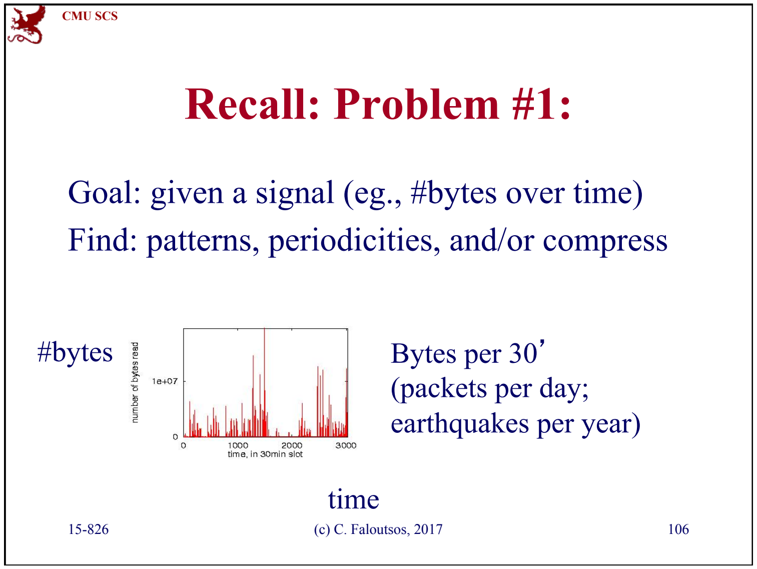


CMU SCS

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



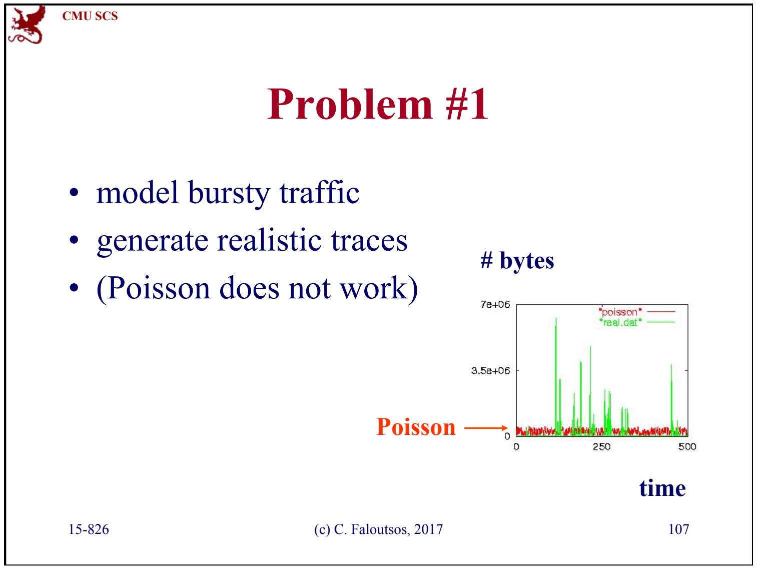
CMU SCS

Recall: Problem #1:

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

Find: patterns, periodicities, and/or compress

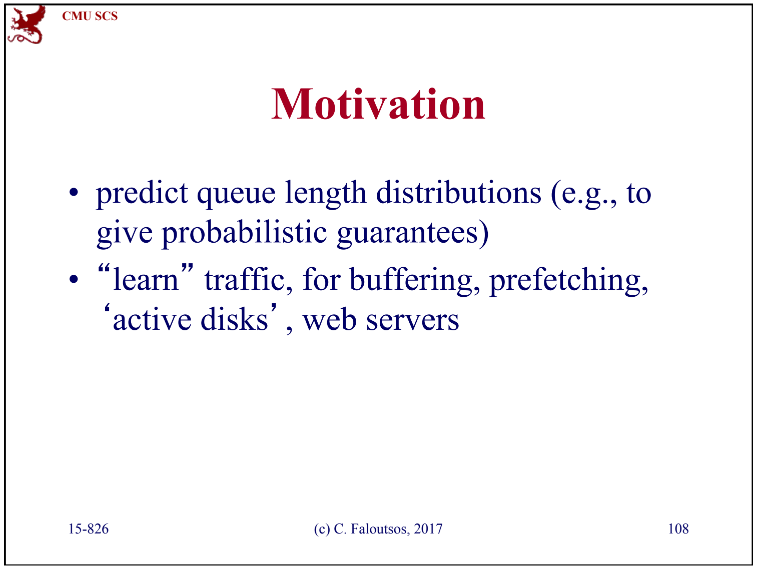
Bytes per 30’ (packets per day; earthquakes per year)



CMU SCS

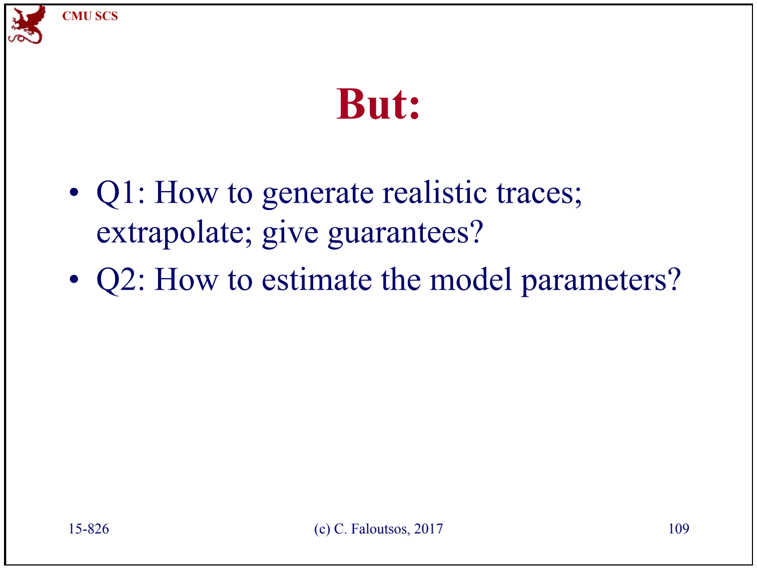
Problem #1

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



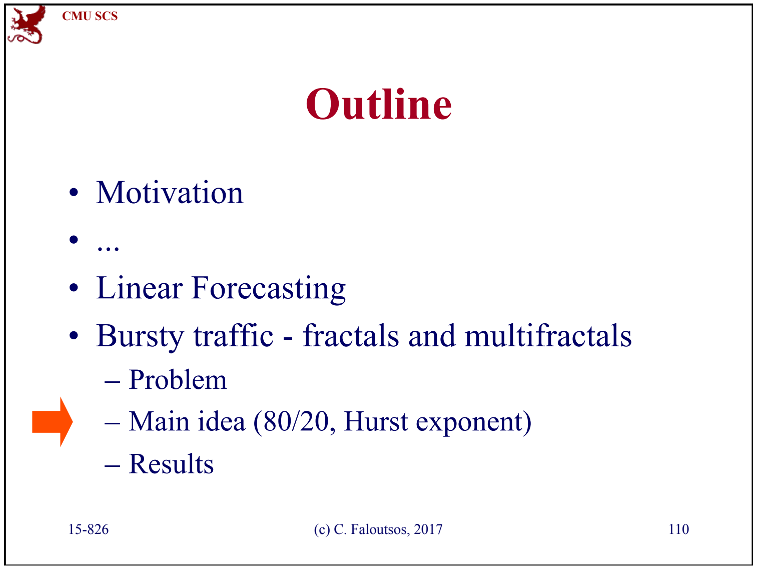
Motivation

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



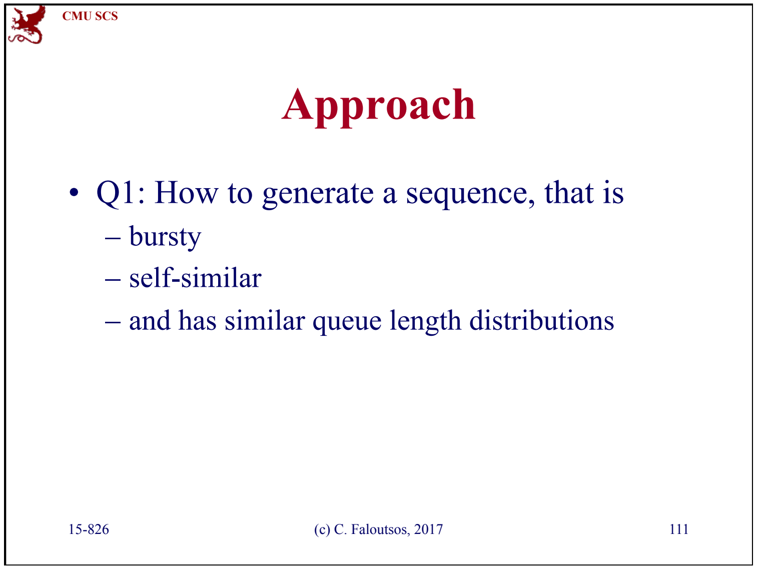
But:

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



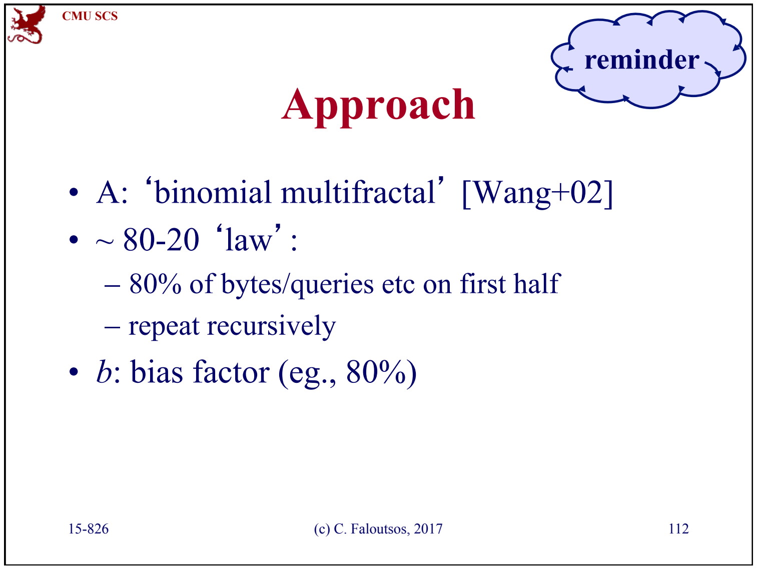
Outline

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



Approach

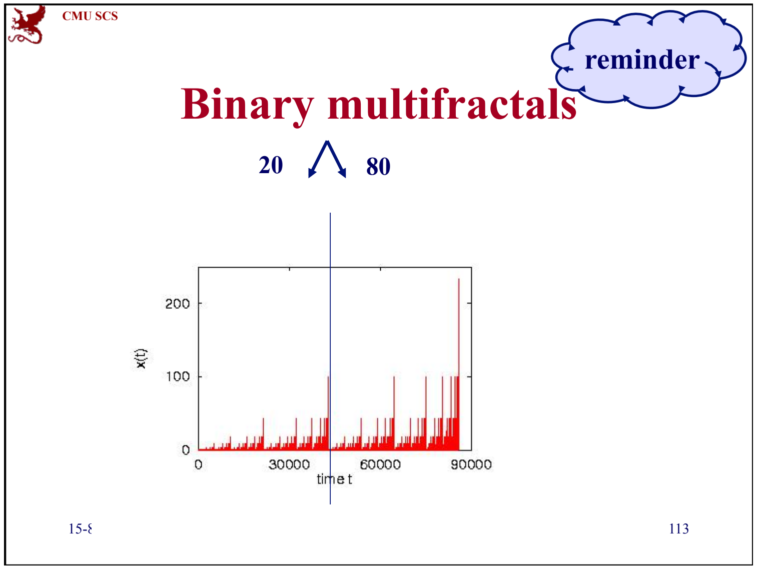
- Q1: How to generate a sequence, that is
  - bursty
  - self-similar
  - and has similar queue length distributions



reminder

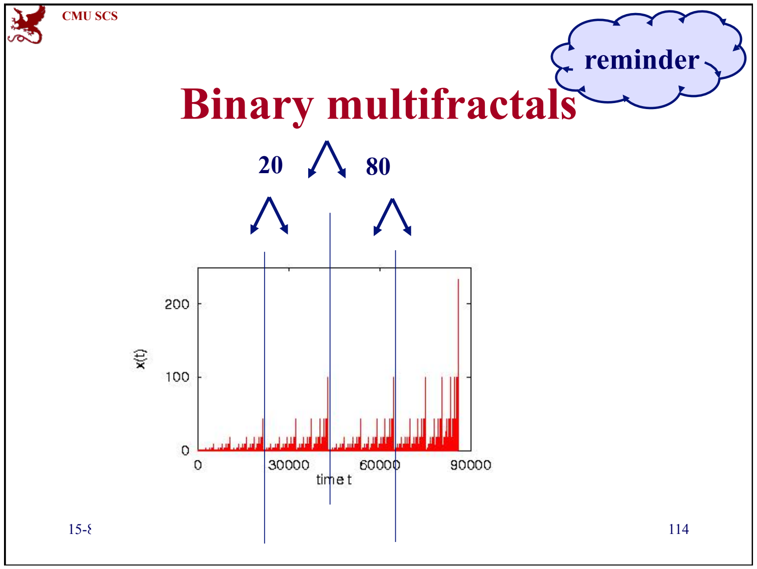
Approach

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



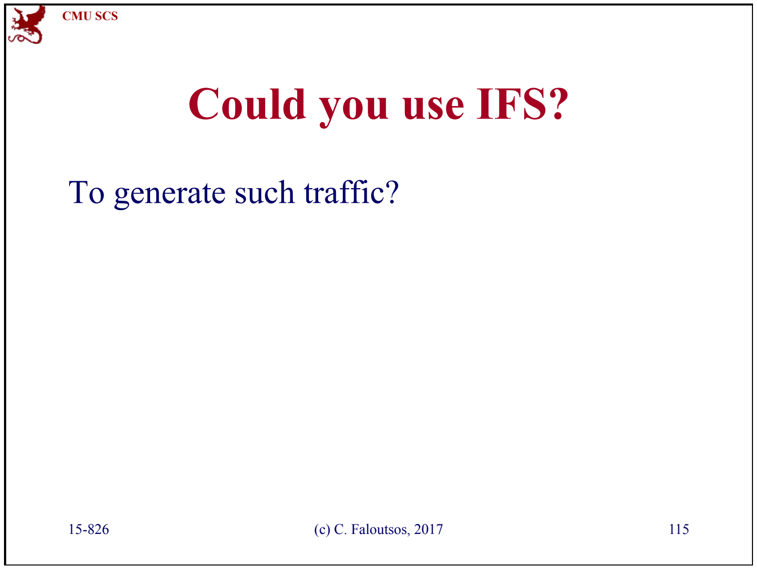
reminder

Binary multifractals



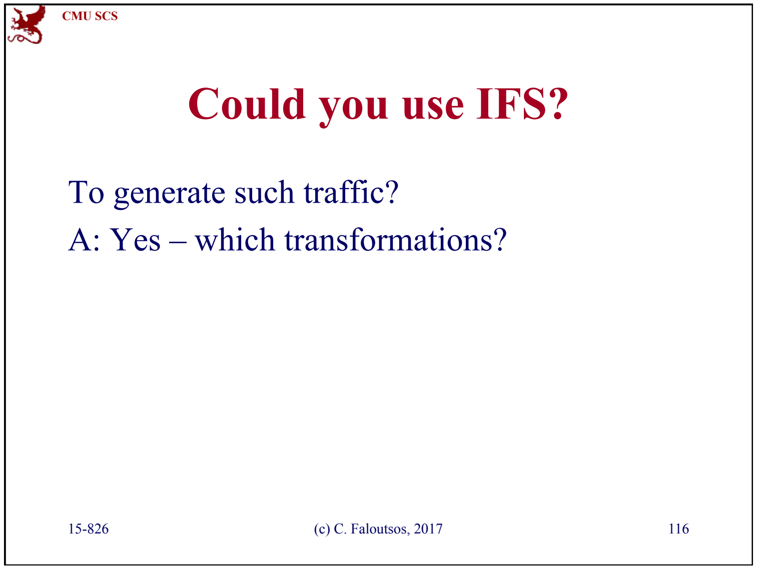
reminder

Binary multifractals



Could you use IFS?

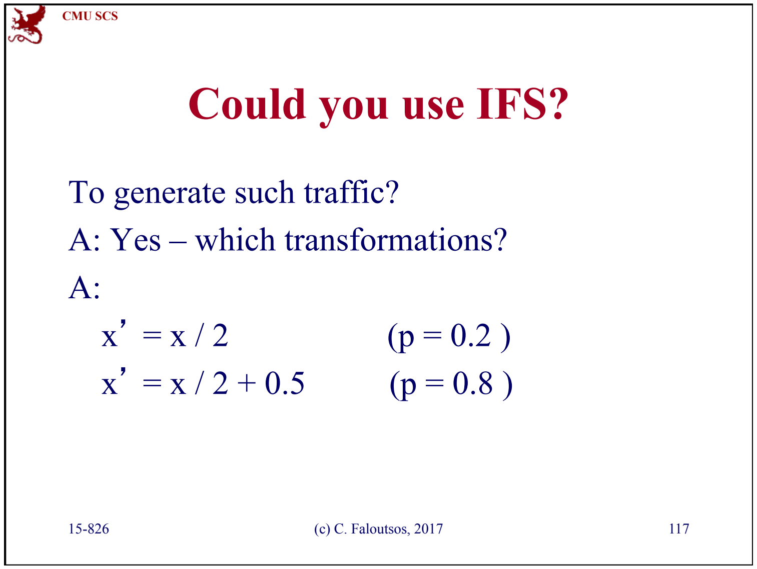
To generate such traffic?



Could you use IFS?

To generate such traffic?

A: Yes – which transformations?



Could you use IFS?

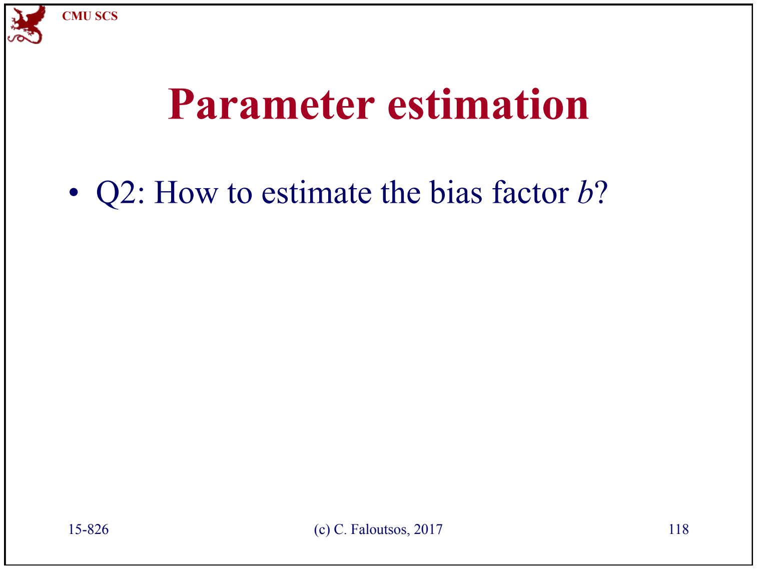
To generate such traffic?

A: Yes – which transformations?

A:

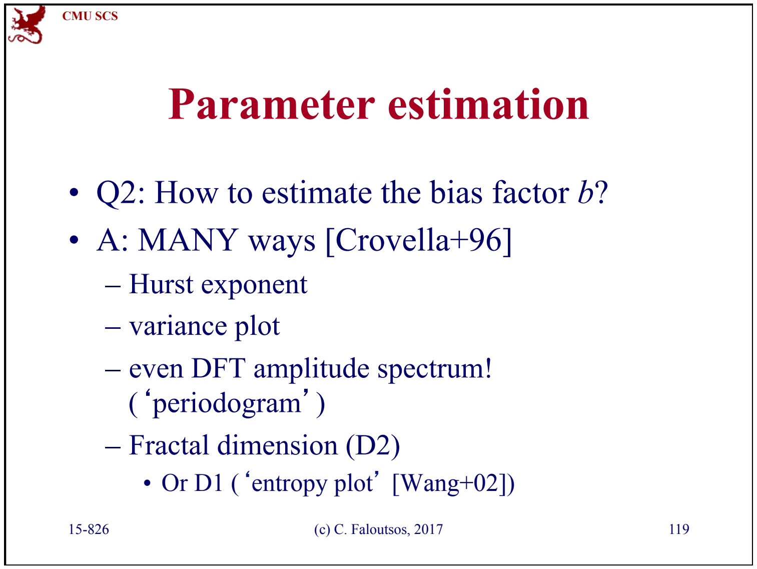
$x' = x / 2$ $(p = 0.2)$

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



Parameter estimation

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

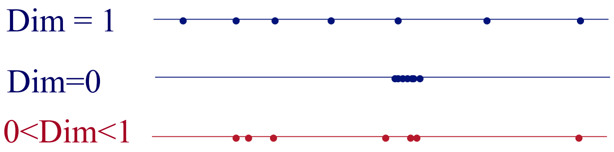


Parameter estimation

- 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])

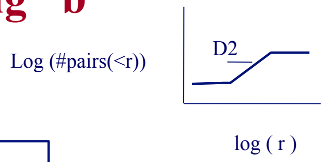
Fractal dimension

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

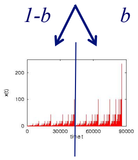


Estimating ‘b’

- **Exercise**: Show that



$$\boxed{D_2 = - \log_2 (\; b^2 + (1-b)^2 \,)}$$



Sanity checks:

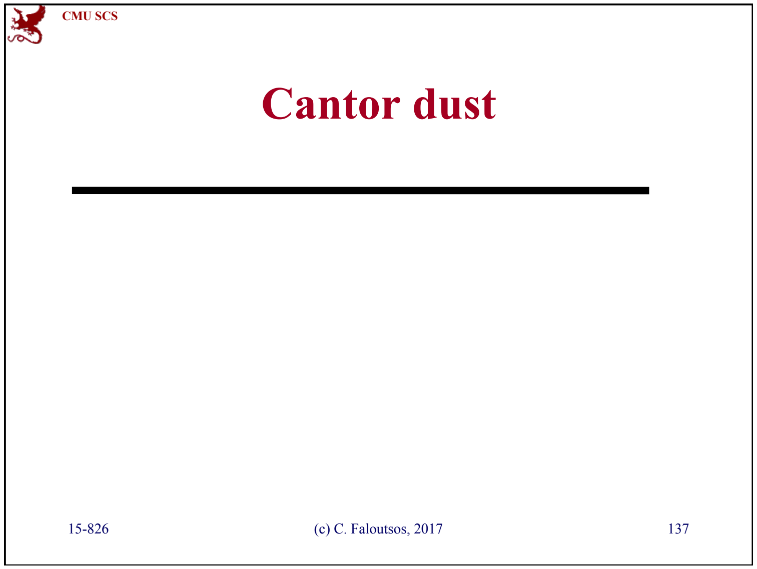
- b =1.0 D_2 = ??
- b =0.5 D_2 = ??

(Fractals, again)

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

Cantor dust

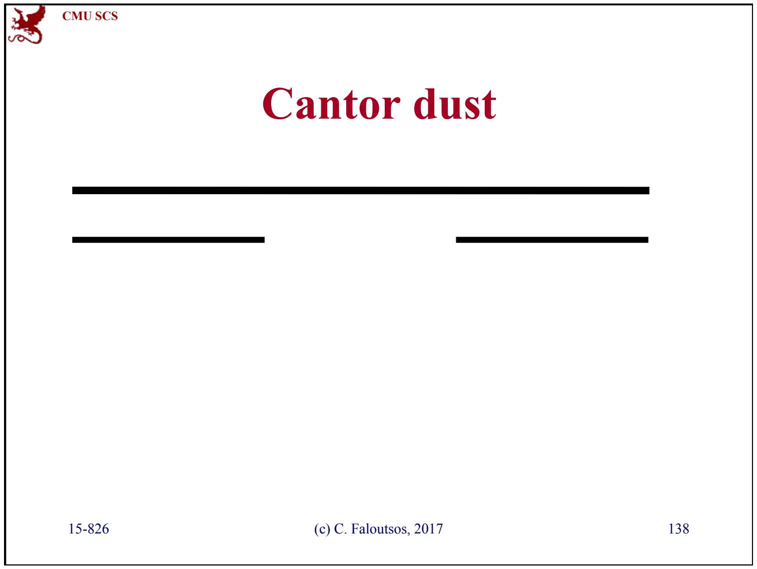
- Eliminate the middle third
- Recursively!



CMU SCS

Cantor dust

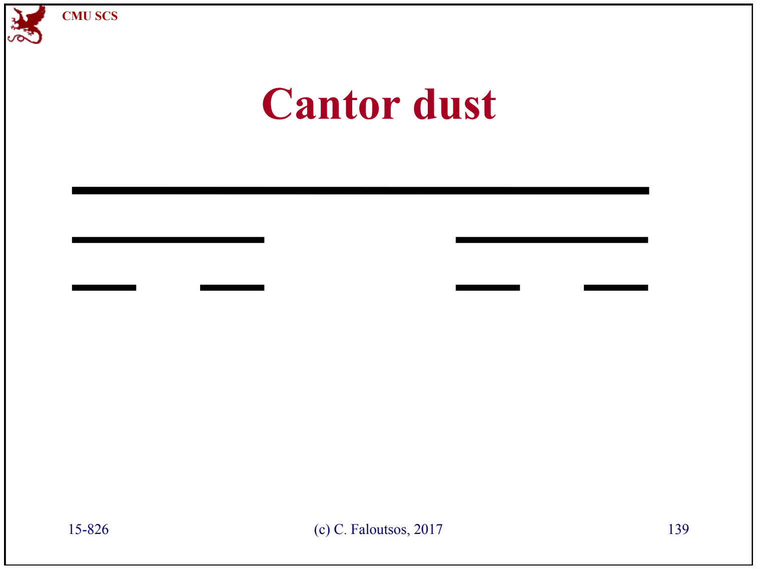
15-826 (c) C. Faloutsos, 2017 137



CMU SCS

Cantor dust

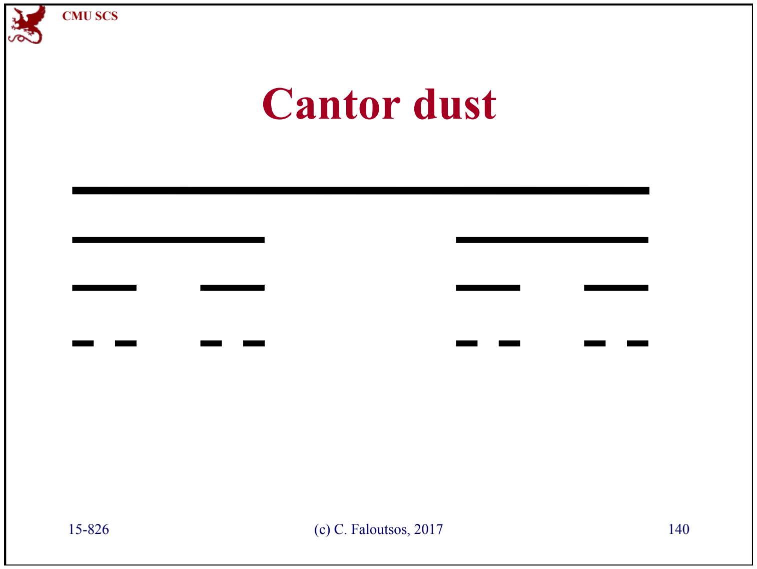
15-826 (c) C. Faloutsos, 2017 138



CMU SCS

Cantor dust

15-826 (c) C. Faloutsos, 2017 139

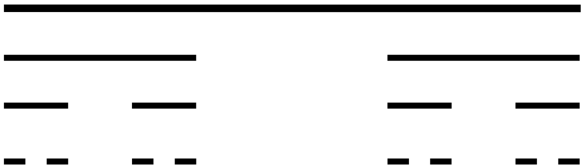


CMU SCS

Cantor dust

15-826 (c) C. Faloutsos, 2017 140

Cantor dust



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

Conclusions

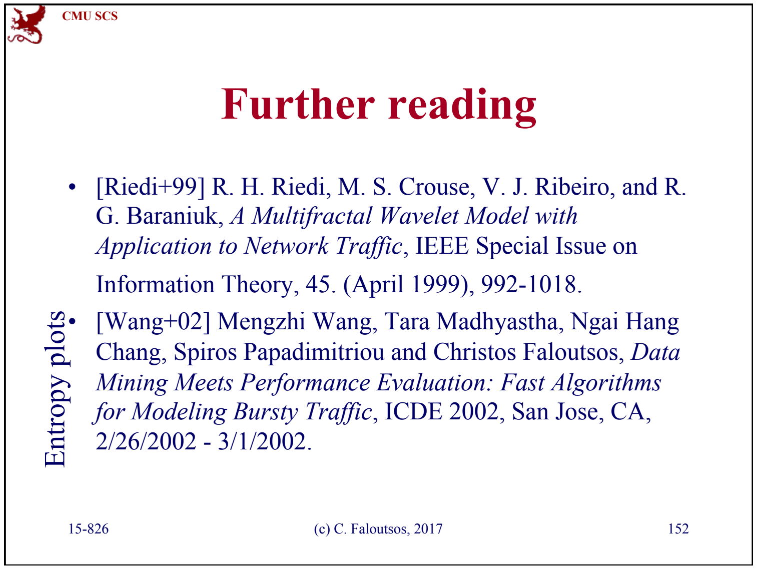
- Multifractals (80/20, ‘b-model’, Multiplicative Wavelet Model (MWM)) for analysis and synthesis of bursty traffic

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.



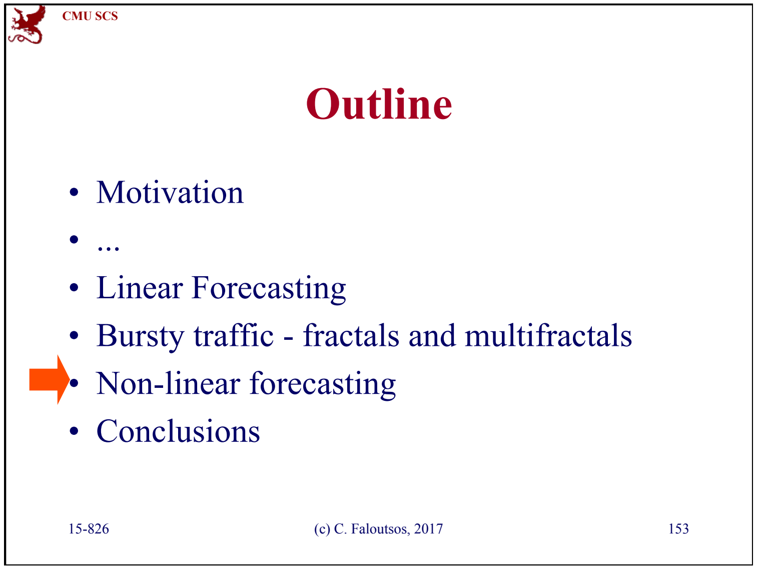
CMU SCS

Further reading

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

Entropy plots

15-826 (c) C. Faloutsos, 2017 152



CMU SCS

Outline

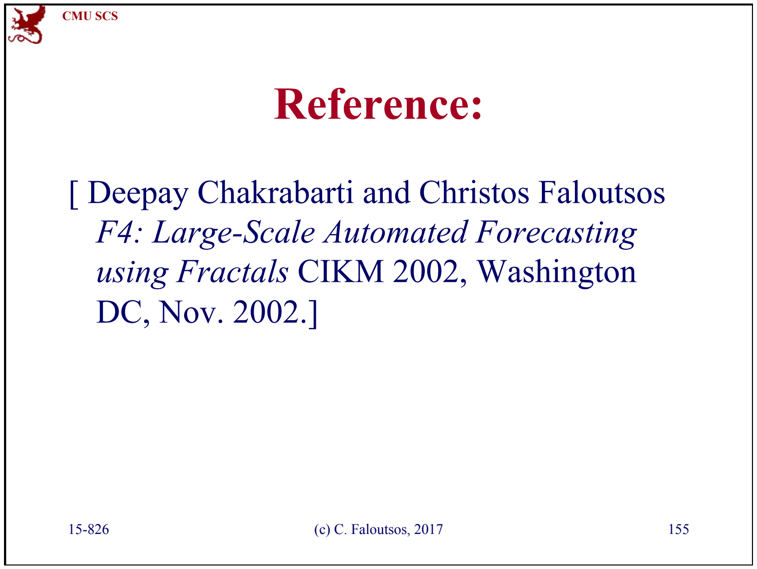
- Motivation
- ...
- Linear Forecasting
- Bursty traffic - fractals and multifractals
- Non-linear forecasting
- Conclusions

15-826 (c) C. Faloutsos, 2017 153

CMU SCS

Chaos and non-linear forecasting

15-826 (c) C. Faloutsos, 2017 154



CMU SCS

Reference:

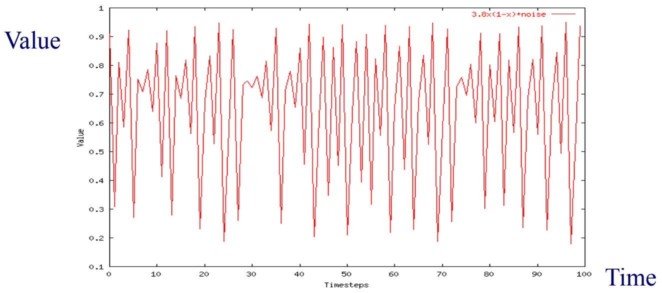
[Deepay Chakrabarti and Christos Faloutsos *F4: Large-Scale Automated Forecasting using Fractals* CIKM 2002, Washington DC, Nov. 2002.]

15-826 (c) C. Faloutsos, 2017 155

Detailed Outline

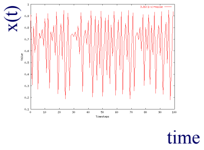
- Non-linear forecasting
  - Problem
  - Idea
  - How-to
  - Experiments
  - Conclusions

Recall: Problem #1

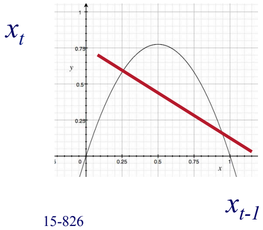


Given a time series $\{x_t\}$, predict its future course, that is, x_{t+1}, x_{t+2}, ...

Datasets



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

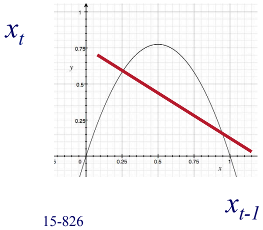


Lag-plot

ARIMA: fails

How to forecast?

- ARIMA - but: linearity assumption



Lag-plot

ARIMA: fails

CMU SCS

How to forecast?

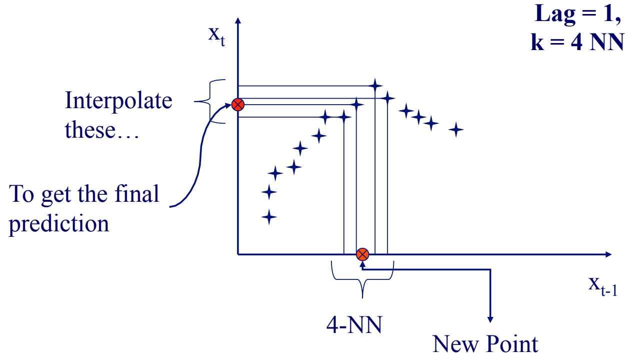
- ARIMA - but: linearity assumption
- ANSWER: 'Delayed Coordinate Embedding' = Lag Plots [Sauer92]

  ~ nearest-neighbor search, for past incidents

CMU SCS

General Intuition (Lag Plot)

Lag = 1,
k = 4 NN



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])

Q2: Choosing number of neighbors *k*

- Manually (typically ~ 1-10)

Q3: How to interpolate?

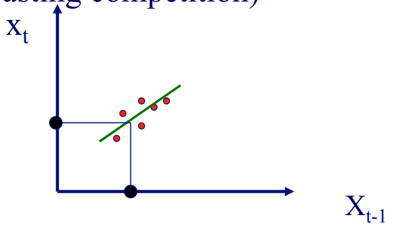
How do we interpolate between the *k* nearest neighbors?

A3.1: Average

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

Q3: How to interpolate?

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



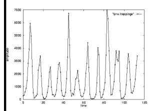
Q4: Any theory behind it?

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)

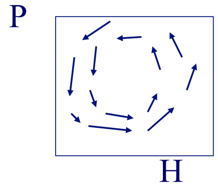
Skip



Theoretical foundation

Example: Lotka-Volterra equations

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



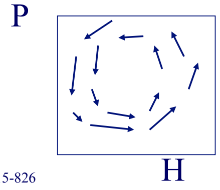
H is count of prey (e.g., hare)
P is count of predators (e.g., lynx)

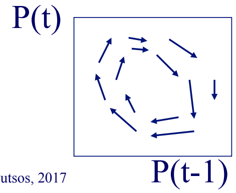
Suppose only P(t) is observed (t=1, 2, …).

Skip

Theoretical foundation

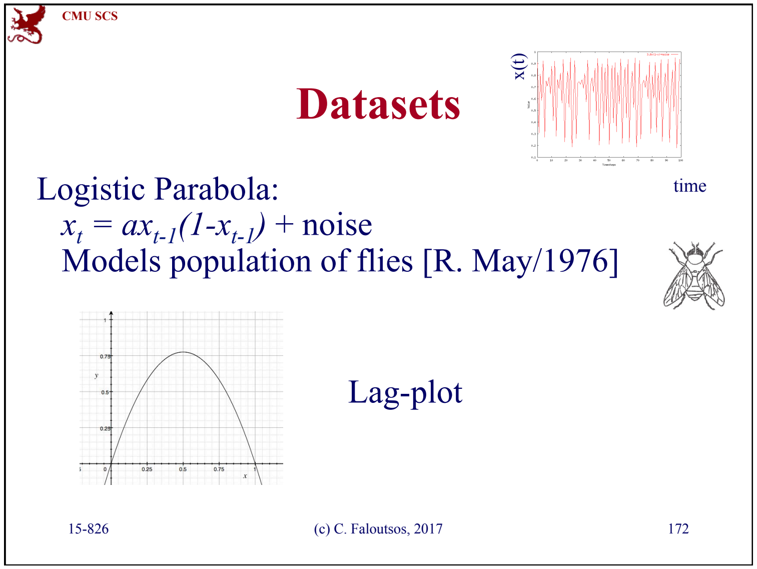
- But the delay vector space is a faithful reconstruction of the internal system state
- So prediction in **delay vector space** is as good as prediction in **state space**





Detailed Outline

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



CMU SCS

Datasets

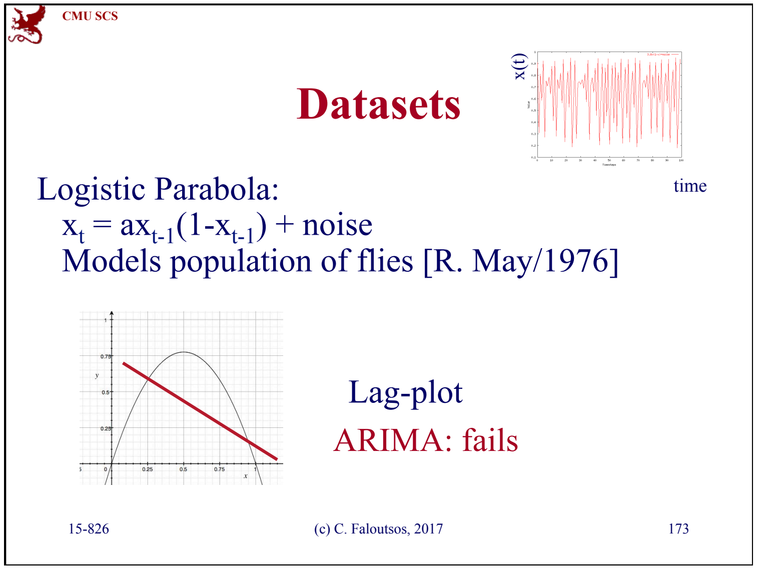
Logistic Parabola:

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

Models population of flies [R. May/1976]

Lag-plot

15-826 (c) C. Faloutsos, 2017 172



CMU SCS

Datasets

Logistic Parabola:

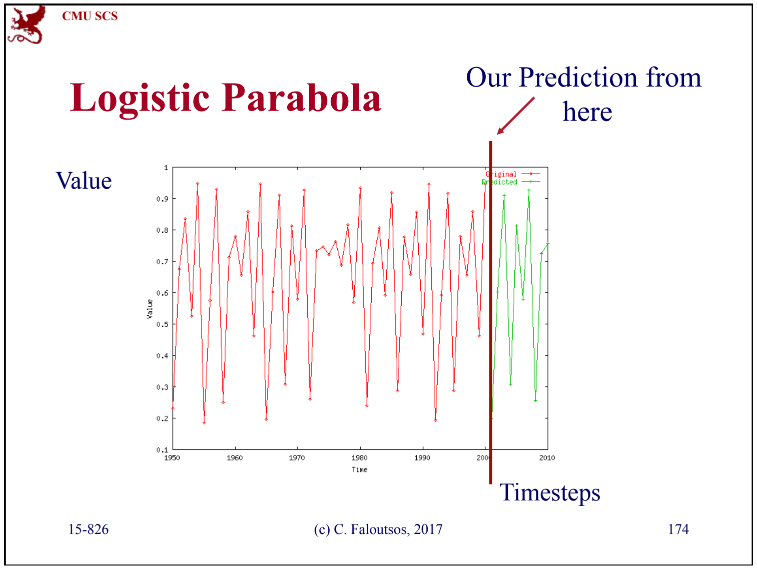
$x_t = ax_{t-1}(1-x_{t-1})$ + noise

Models population of flies [R. May/1976]

Lag-plot

ARIMA: fails

15-826 (c) C. Faloutsos, 2017 173

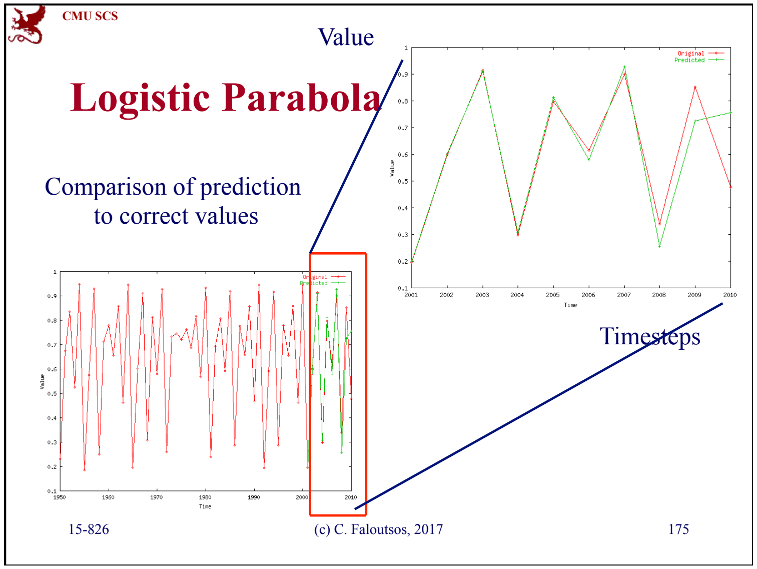


CMU SCS

Logistic Parabola

Our Prediction from here

15-826 (c) C. Faloutsos, 2017 174



CMU SCS

Logistic Parabola

Comparison of prediction to correct values

15-826 (c) C. Faloutsos, 2017 175

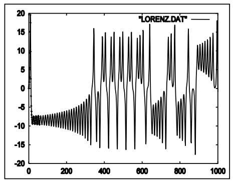
Datasets

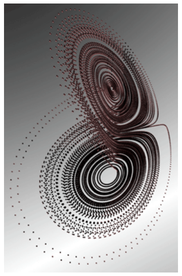
LORENZ: Models convection currents in the air

dx / dt = a (y - x)

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

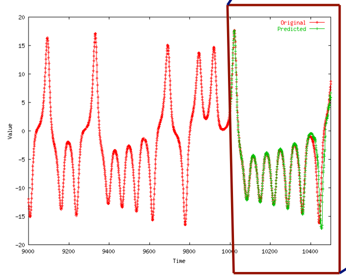
dz / dt = xy - c z

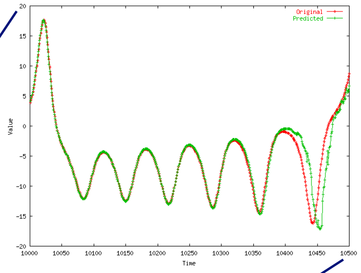




LORENZ

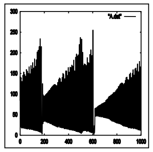
Comparison of prediction to correct values





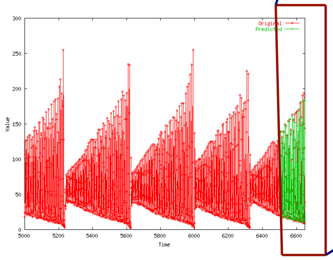
Datasets

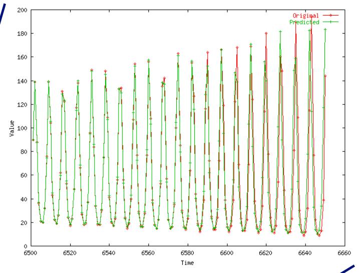
- LASER: fluctuations in a Laser over time (used in Santa Fe competition)



Laser

Comparison of prediction to correct values





Conclusions

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

References

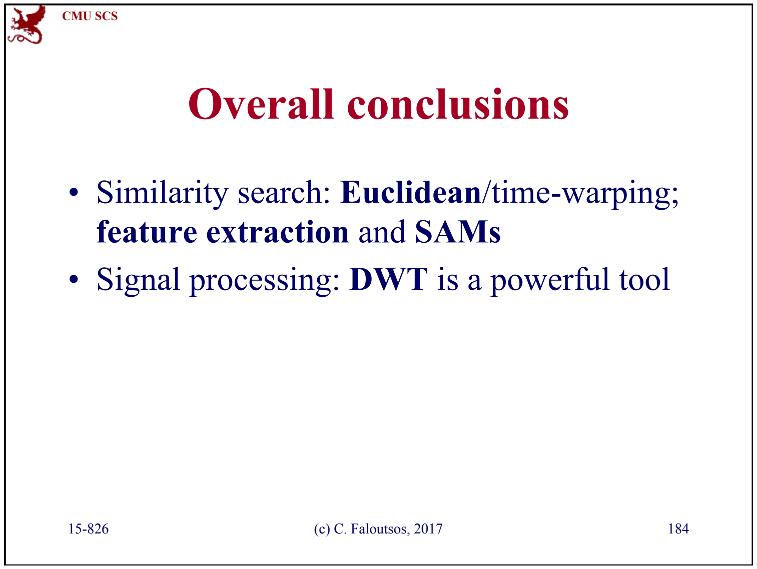
- Deepay Chakrabarti and Christos Faloutsos *F4: Large-Scale Automated Forecasting using Fractals* CIKM 2002, Washington DC, Nov. 2002.
- Sauer, T. (1994). *Time series prediction using delay coordinate embedding*. (in book by Weigend and Gershenfeld, below) Addison-Wesley.
- Takens, F. (1981). *Detecting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

Overall conclusions

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

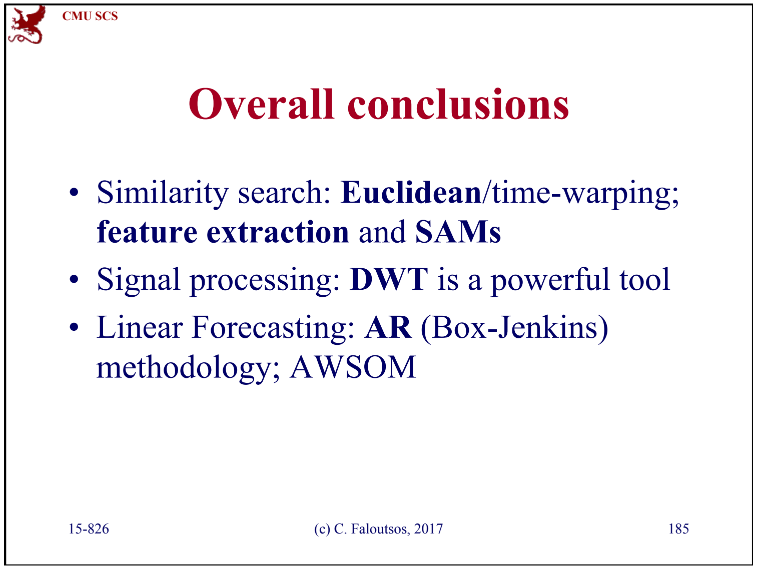


CMU SCS

Overall conclusions

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

15-826 (c) C. Faloutsos, 2017 184

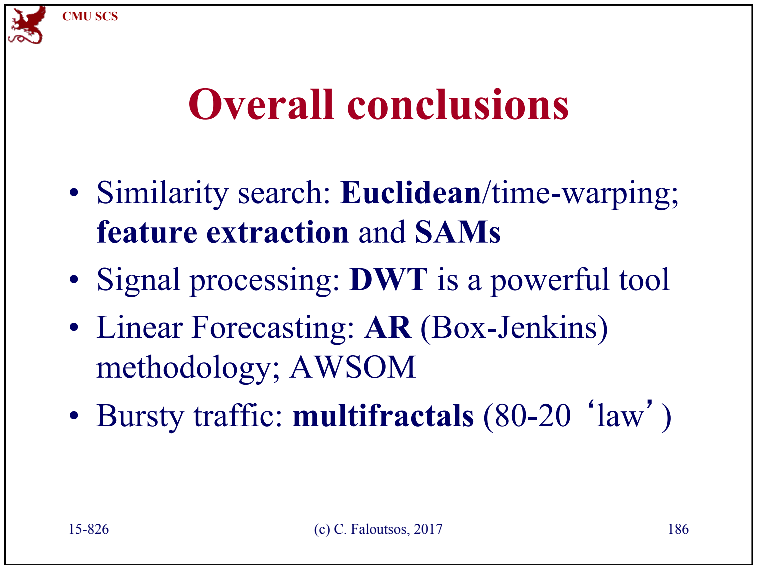


CMU SCS

Overall conclusions

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

15-826 (c) C. Faloutsos, 2017 185

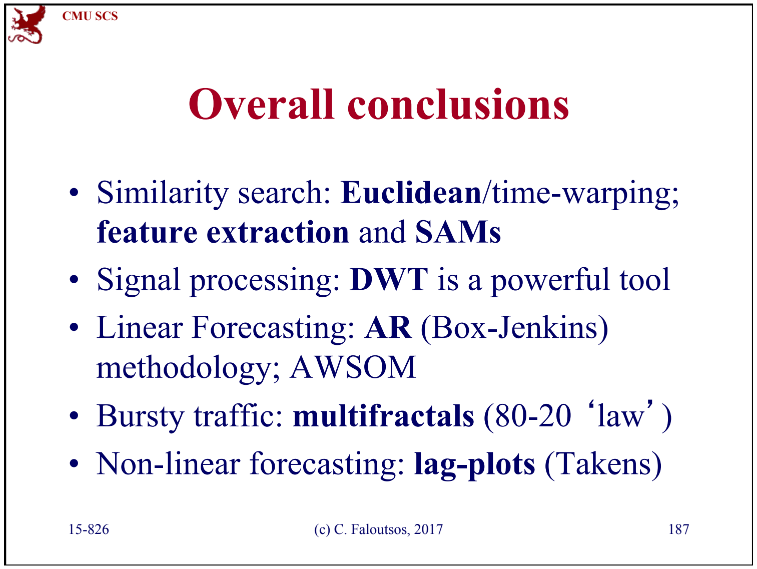


CMU SCS

Overall conclusions

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

15-826 (c) C. Faloutsos, 2017 186



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; AWSOM
- Bursty traffic: **multifractals** (80-20 ‘law’)
- Non-linear forecasting: **lag-plots** (Takens)

15-826 (c) C. Faloutsos, 2017 187