Part 1

Similarity search, pattern discovery and summarization

Yasushi Sakurai (Kumamoto University)
Yasuko Matsubara (Kumamoto University)
Christos Faloutsos (Carnegie Mellon University)

Outline

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Stream mining

- Applications
  - Sensor monitoring
  - Network analysis
  - Financial and/or business transaction data
  - Web access and media service logs
  - Moving object tracking
  - Industrial manufacturing

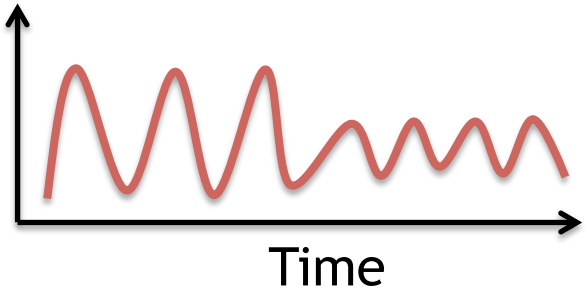
Stream mining

- Requirements
  - **Fast**

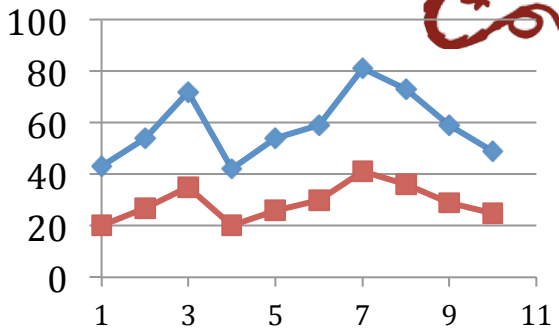
    high performance and quick response
  - **Nimble**

    low memory consumption, single scan
  - **Accurate**

    good approximation for pattern discovery and feature extraction



Monitoring data streams



- Correlation coefficient

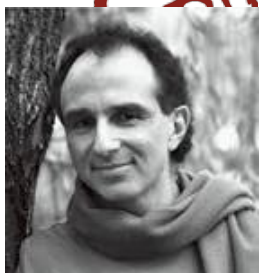
$$\rho = \frac{\sum_{t=1}^{n} (x_t - \bar{x}) \cdot (y_t - \bar{y})}{\sigma(x) \cdot \sigma(y)}$$

$$\sigma(x) = \sqrt{\sum_{t=1}^{n} (x_t - \bar{x})^2}$$

- Correlation coefficient and the (Euclidean) distance

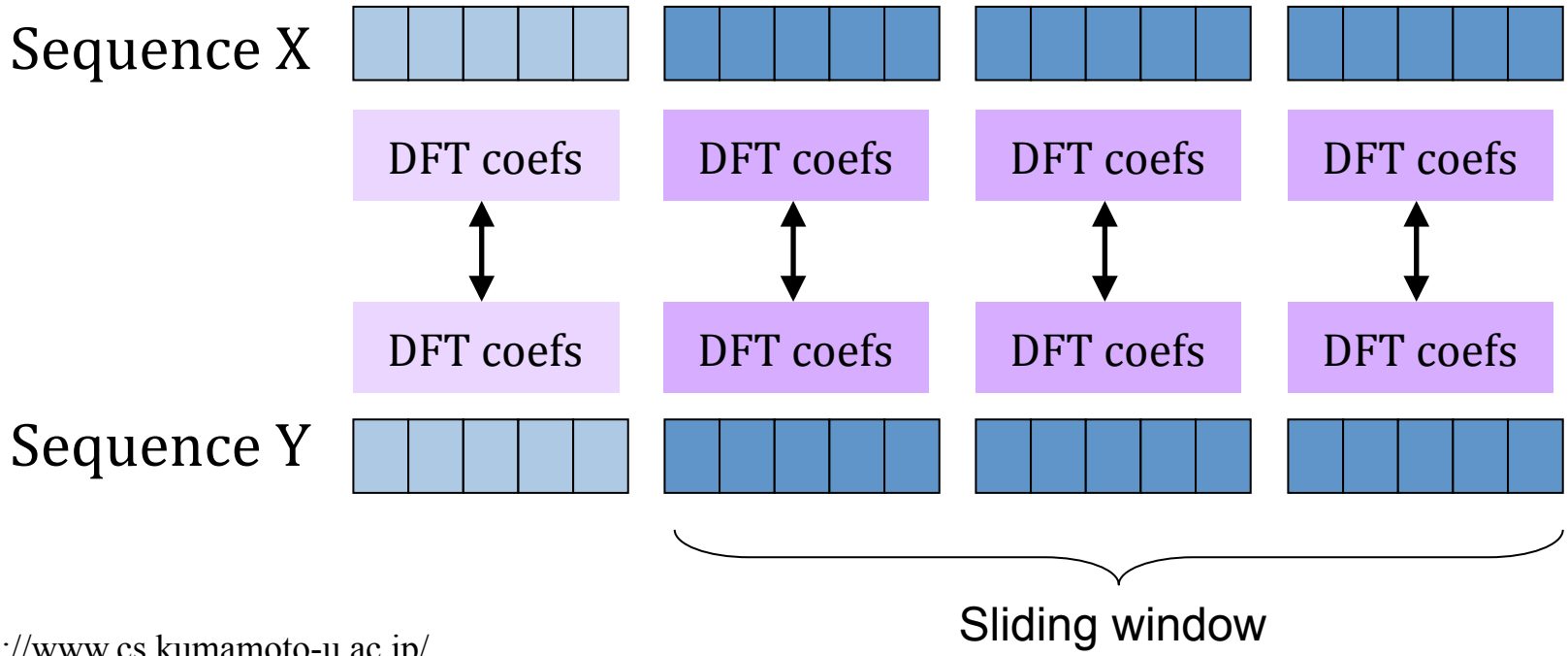
$$\rho = 1 - \frac{1}{2} \sum_{t=1}^{n} (\hat{x}_t - \hat{y}_t)^2 \qquad \hat{x}_t = (x_t - \bar{x}) / \sigma(x)$$

Monitoring data streams



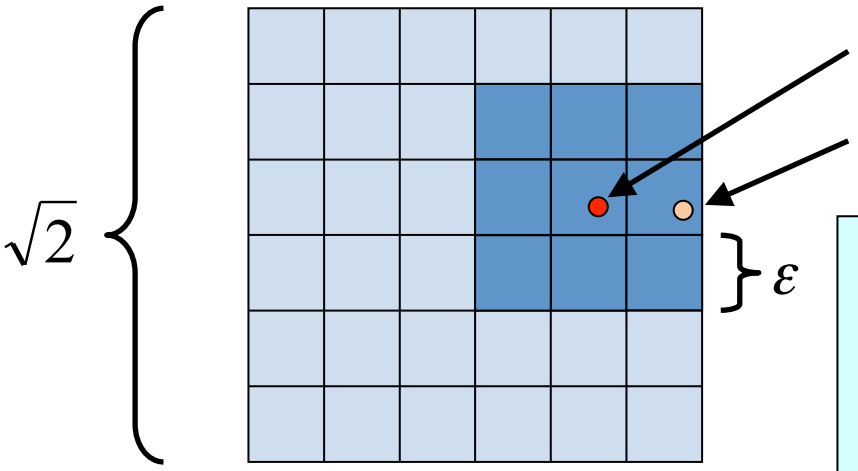
Dennis Shasha

- Correlation monitoring [Zhu+, vldb02]
  - DFT coefficients for each basic window
  - Correlation coefficient of each sliding window computed from the `sketch' (DFT coefs)



Monitoring data streams

- Grid structure (to avoid checking all pairs)
  - DFT coefficients yields a vector
  - High correlation -> closeness in the vector space

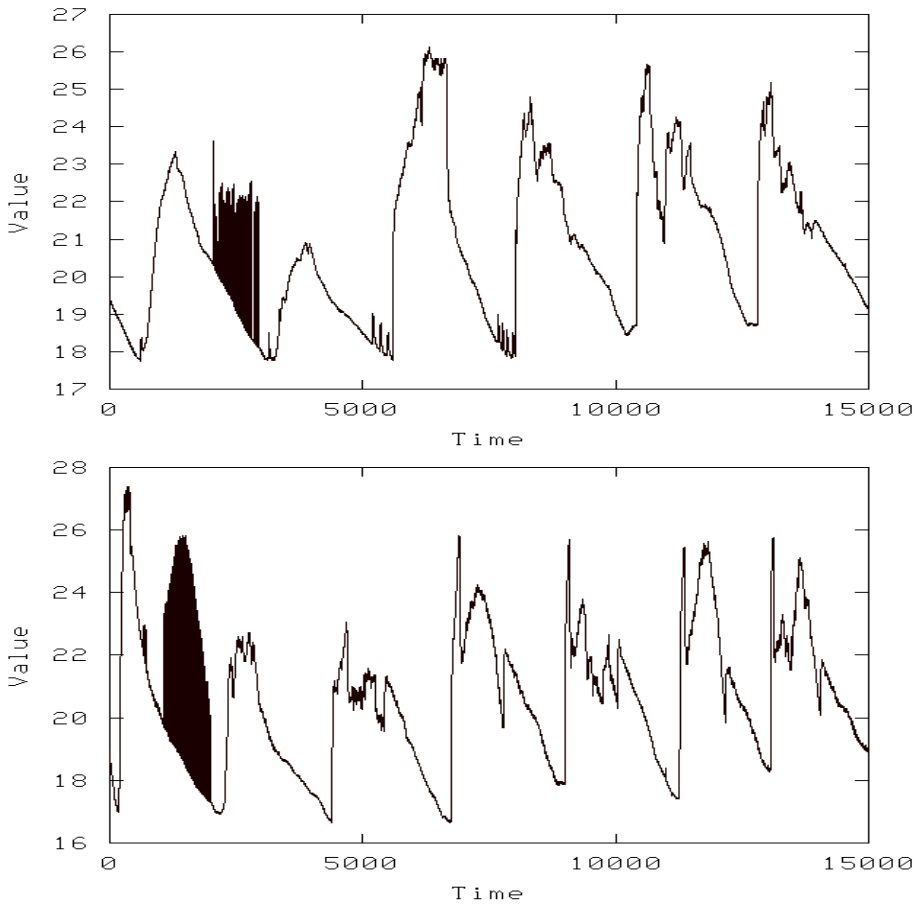


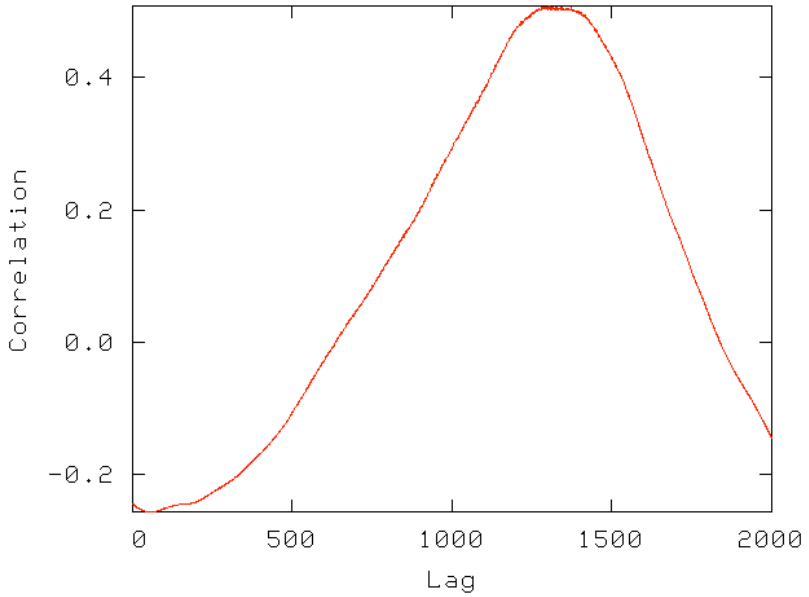
Correlation coefficients and the Euclidean distance

$$\rho = 1 - \frac{1}{2}\sum_{t=1}^{n}(\hat{x}_t - \hat{y}_t)^2$$

Monitoring data streams

- Lag correlation [Sakurai+, sigmod05]



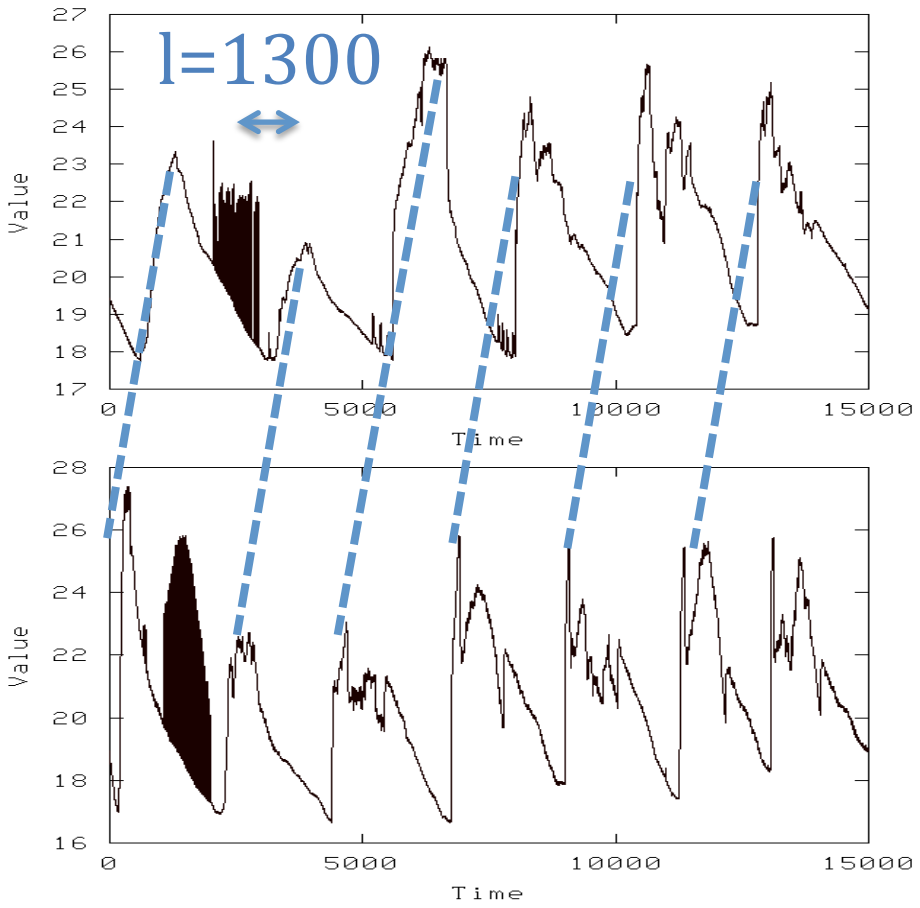


CCF (Cross-Correlation Function)

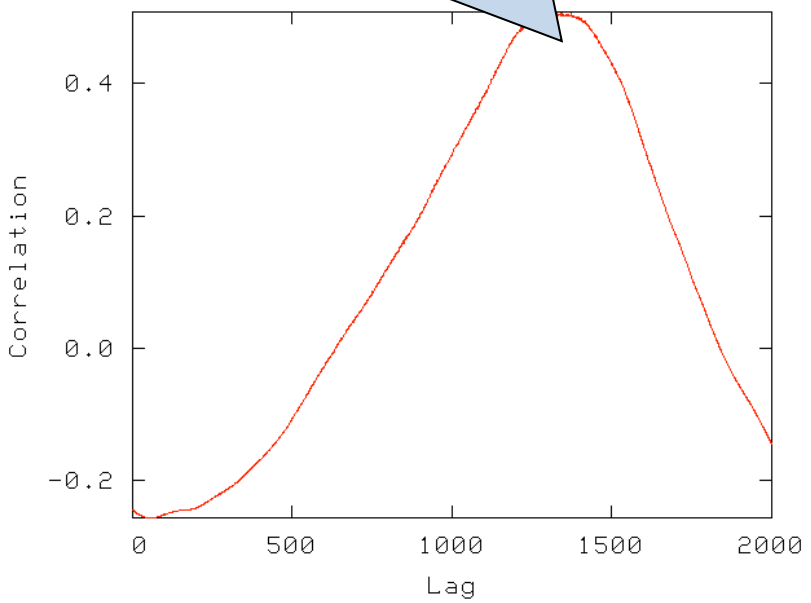
http://www.cs.kumamoto-u.ac.jp/~yasuko/TALKS/15-SIGMOD-tut/

Monitoring data streams

- Lag correlation [Sakurai+, sigmod05]



correlated with lag l=1300



CCF (Cross-Correlation Function)

Lag correlation

- Definition of 'score', absolute value of $R(l)$

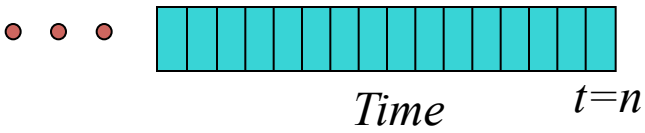
$$score(l) = |R(l)| \qquad R(l) = \frac{\sum_{t=l+1}^{n}(x_t - \bar{x})(y_{t-l} - \bar{y})}{\sqrt{\sum_{t=l+1}^{n}(x_t - \bar{x})^2}\sqrt{\sum_{t=1}^{n-l}(y_t - \bar{y})^2}}$$

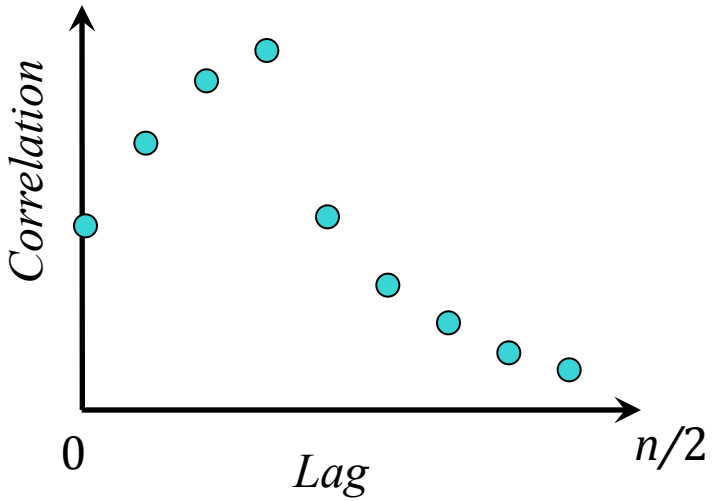
- Lag correlation
  - Given a threshold γ, $score(l) > \gamma$
  - A local maximum
  - The earliest such maximum, if more maxima exist

Lag correlation

- Why not naïve?
  - Compute correlation coefficient for each lag

    $l = \{0, 1, 2, 3, \ldots, n/2\}$
- But
  - $O(n)$ space
  - $O(n^2)$ time
  - or $O(n \log n)$ time w/ FFT

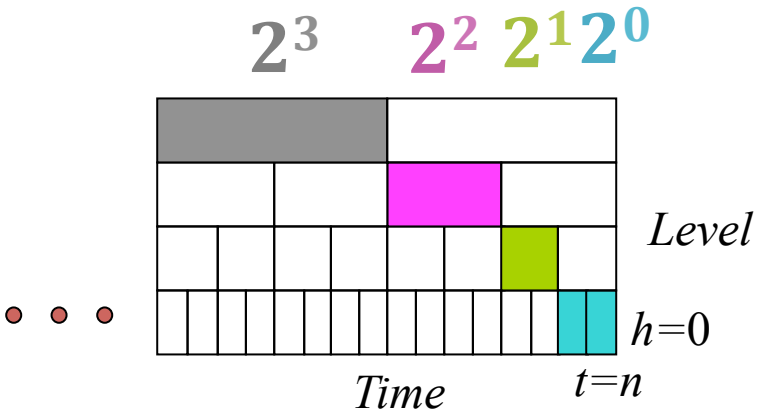


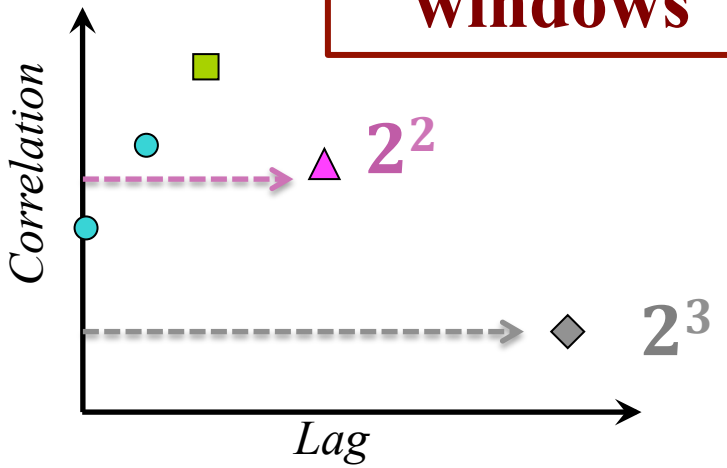


Lag correlation

- BRAID
  - Geometric lag probing + smoothing
  - Use colored windows
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

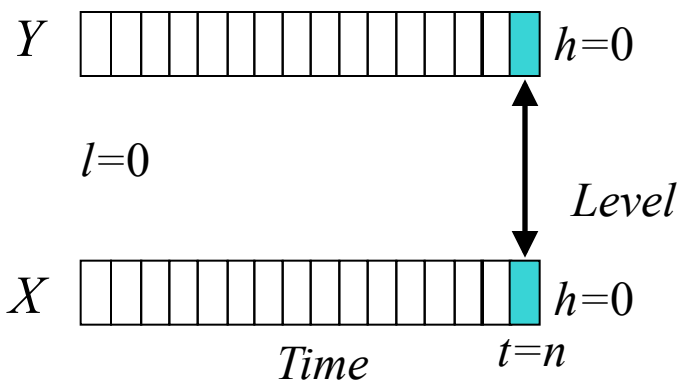
Multi-scale windows

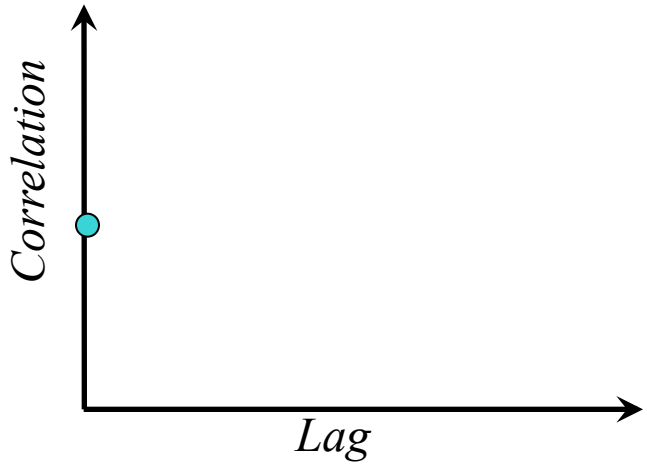




Lag correlation

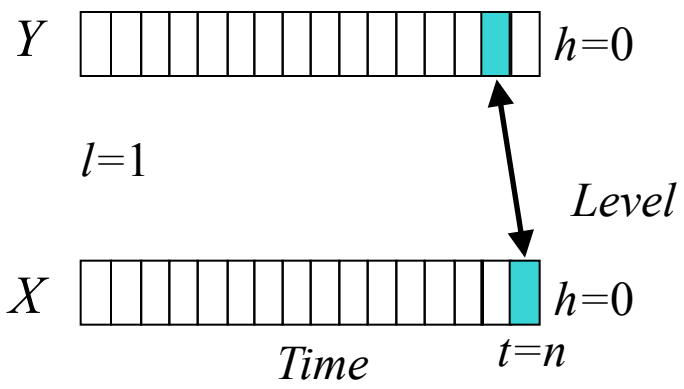
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

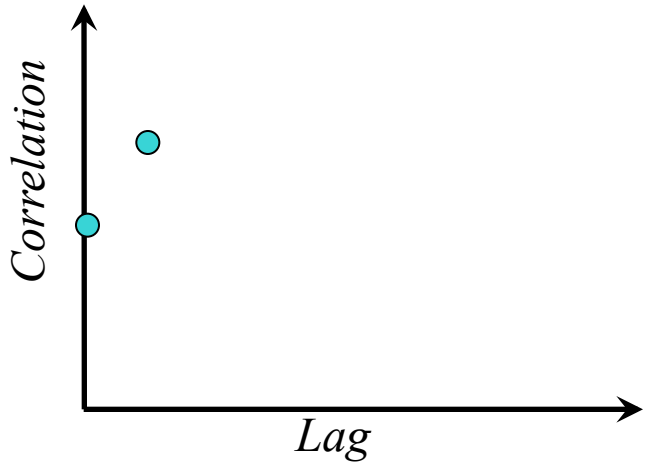




Lag correlation

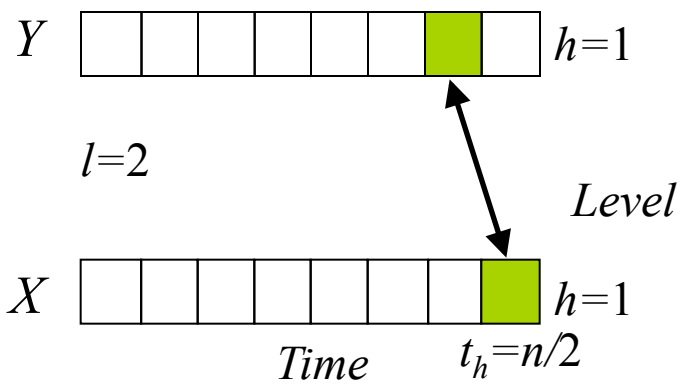
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

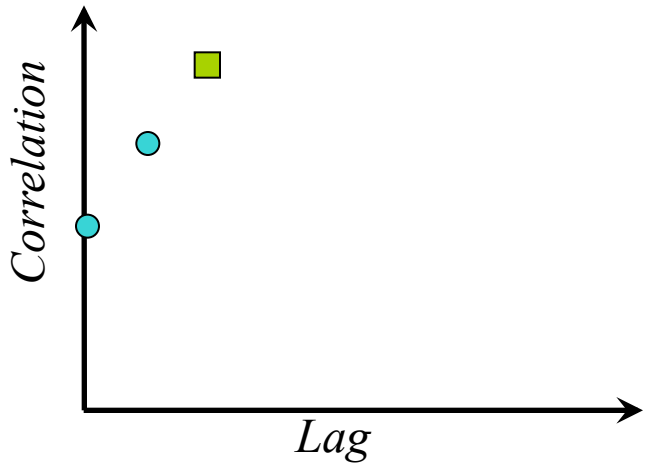




Lag correlation

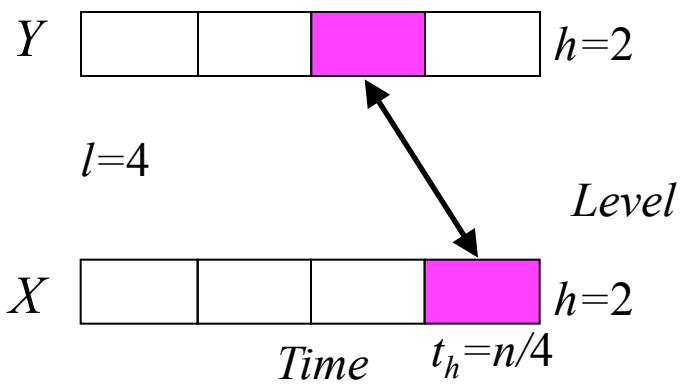
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

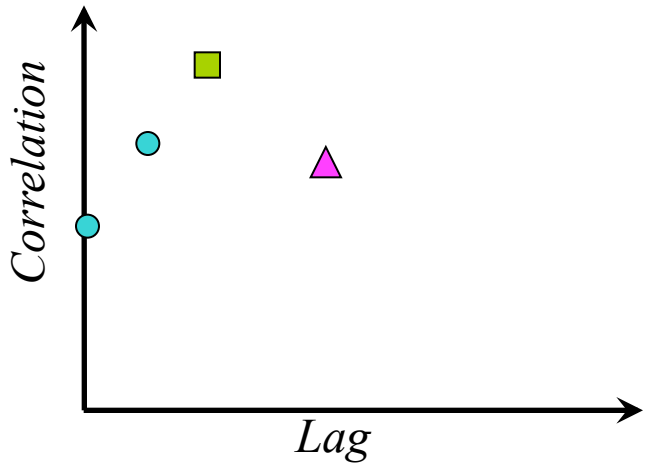




Lag correlation

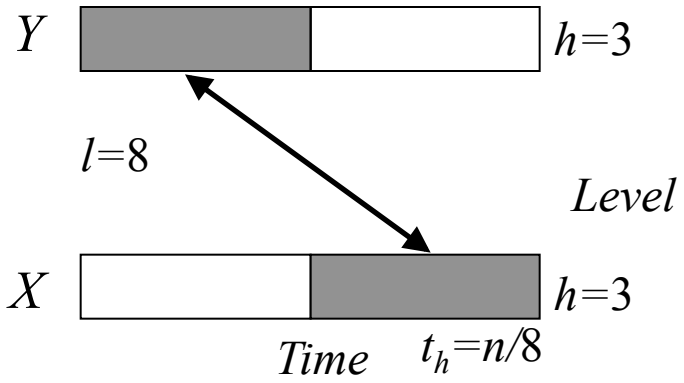
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

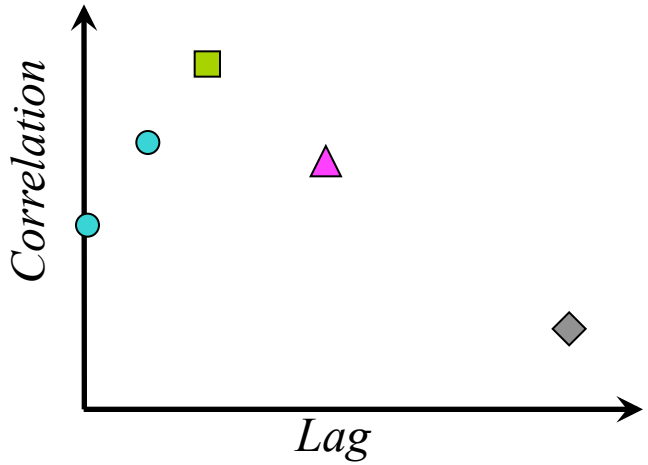




Lag correlation

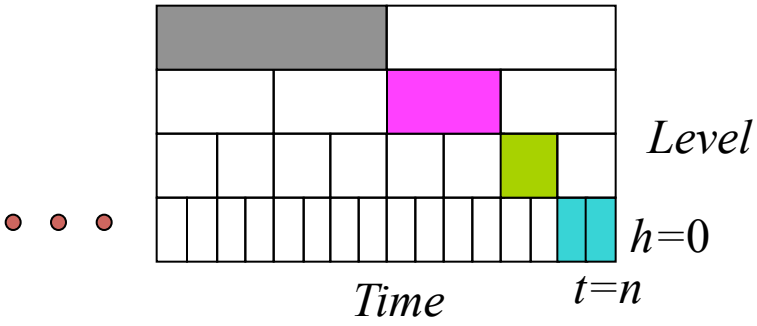
- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$

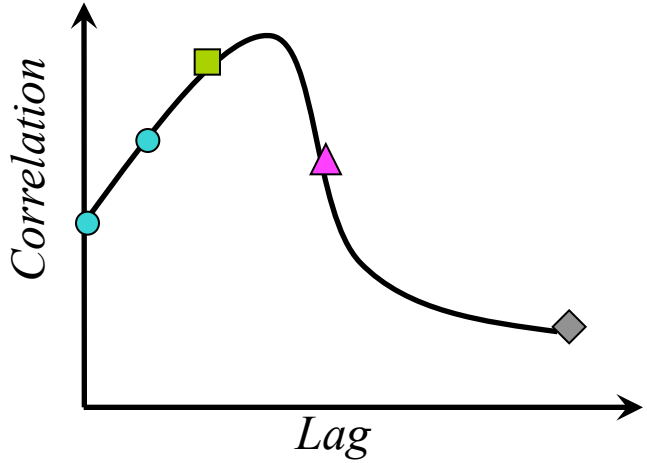




Lag correlation

- BRAID
  - Geometric lag probing + smoothing
  - Keep track of only a geometric progression of the lag values: $l = \{0, 1, 2, 4, 8, \ldots, 2^h, \ldots\}$
  - Use a cubic spline to interpolate

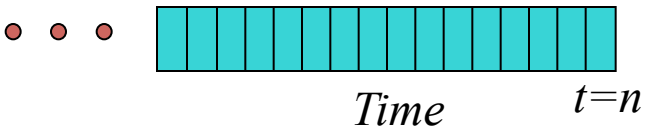




Lag correlation

- Why not naïve?
  - Compute correlation coefficient for each lag

    $l = \{0, 1, 2, 3, \ldots, n/2\}$
- But
  - $O(n)$ space
  - $O(n^2)$ time
  - or $O(n \log n)$ time w/]



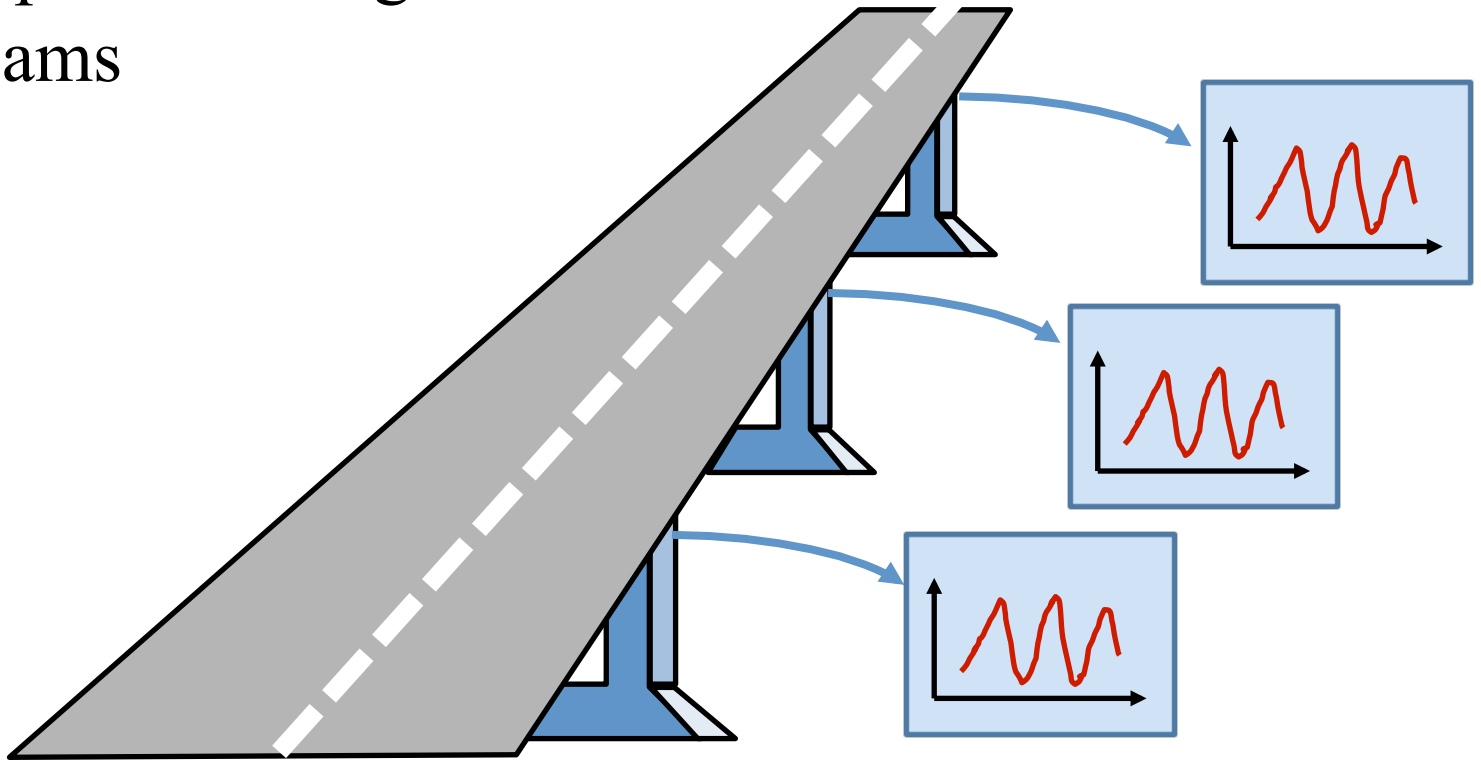
BRAID

- $O(log\ n)$ space
- $O(1)$ time

Multi-scale windows

BRAID in the real world

- Bridge structural health monitoring
  - Structural monitoring using vibration/shock sensors
  - Keep track of lag correlations for sensor data streams



BRAID in the real world

- Bridge structural health monitoring
  - Goal: real-time anomaly detection for disaster prevention
  - Several thousands readings (per sec) from several hundreds sensor nodes

Structural health monitoring

Vibration/shock sensor

- Uses BRAID
- Metropolitan Expressway (Tokyo, Japan)

BRAID in the real world

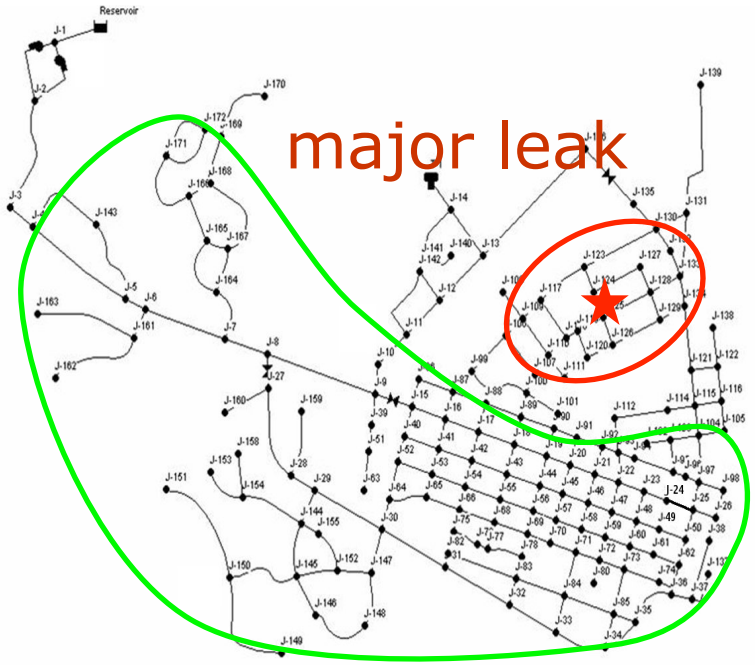
- Bridge structural health monitoring with BRAID

Metropolitan Expressway (Tokyo, Japan)

Tokyo Gate Bridge (Tokyo, Japan)

Can Tho Bridge (Vietnam)

Feature extraction from streams



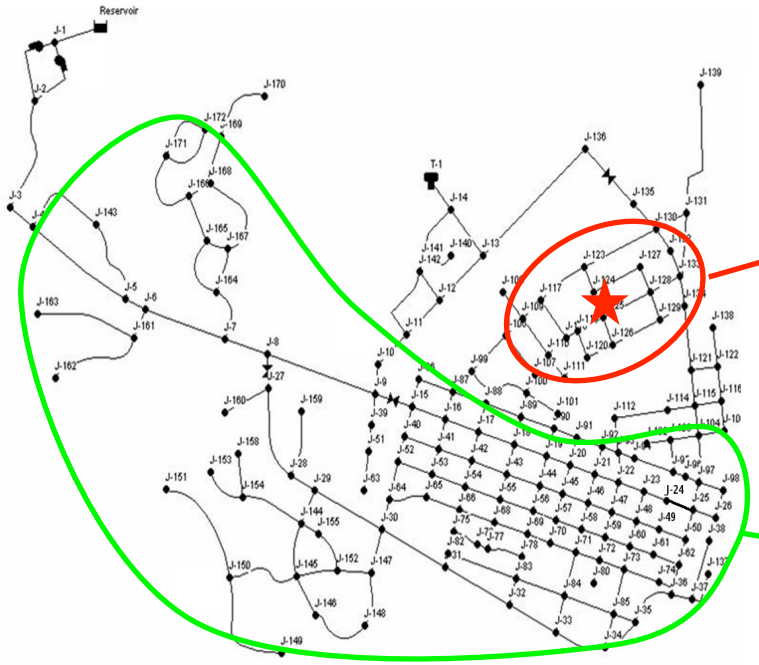
water distribution network

- Find hidden variables from streams [Papadimitriou+, vldb2005]

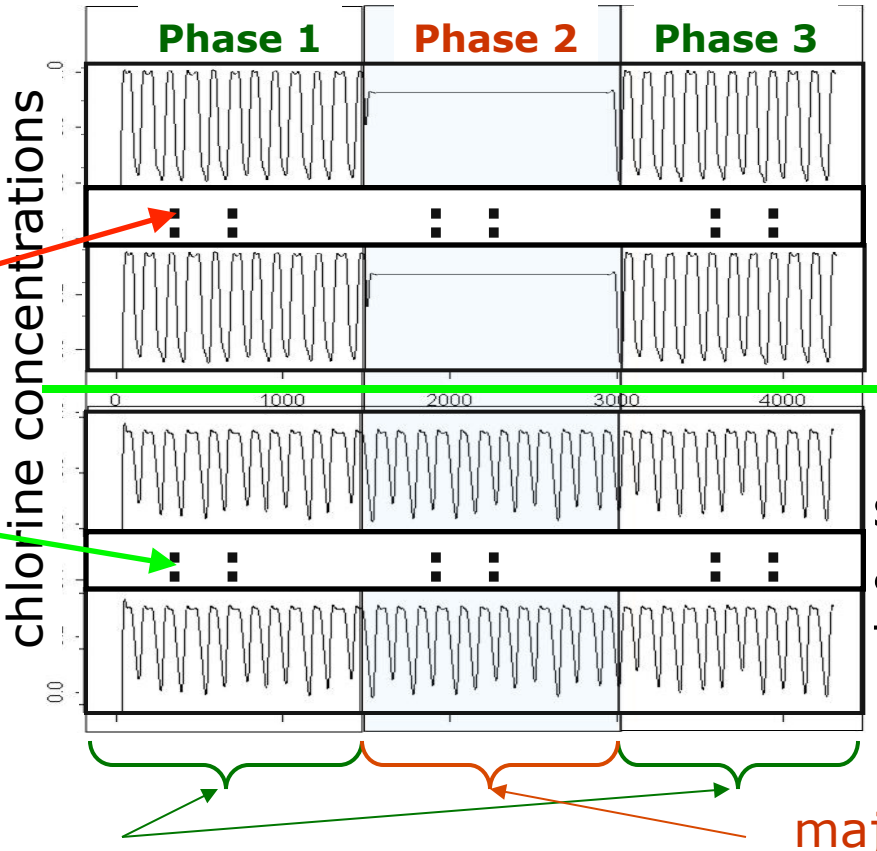
May have hundreds of measurements, but it is **unlikely they are completely unrelated**!

Feature extraction from streams

hidden variables

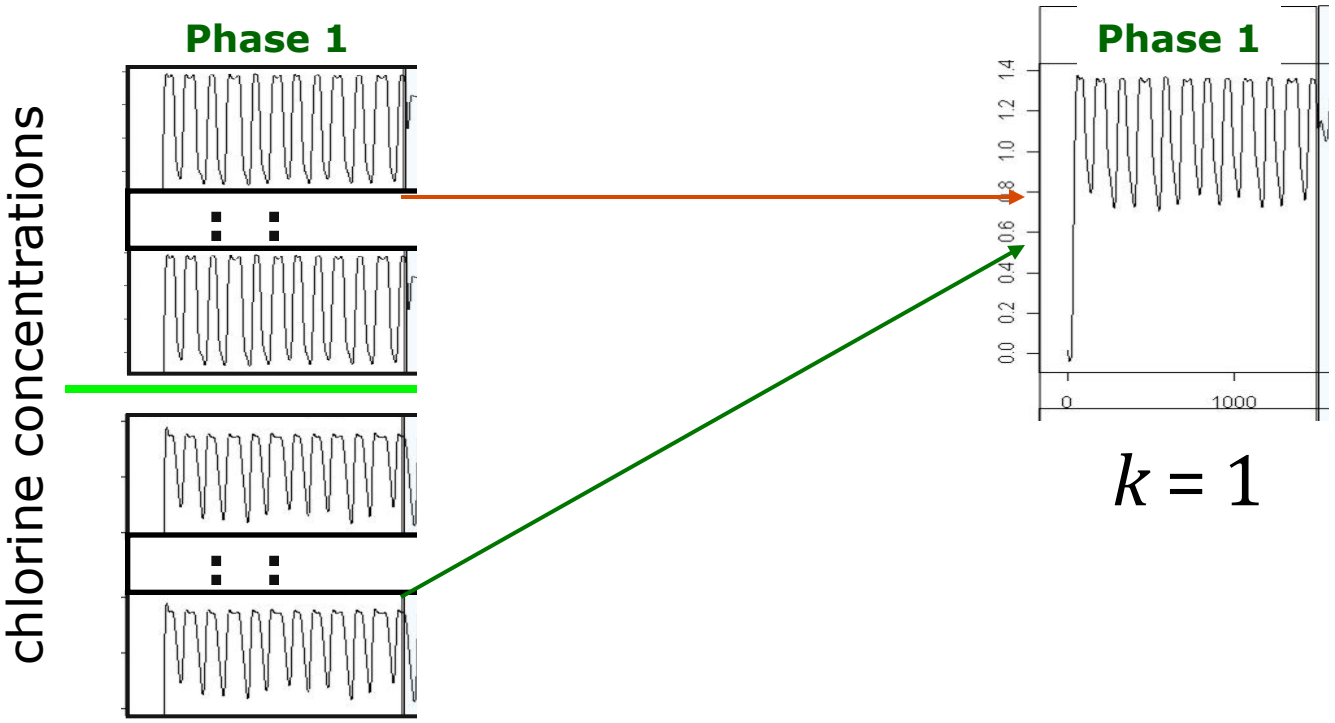


water distribution network



May have hundreds of measurements, but it is **unlikely they are completely unrelated**!

Motivation

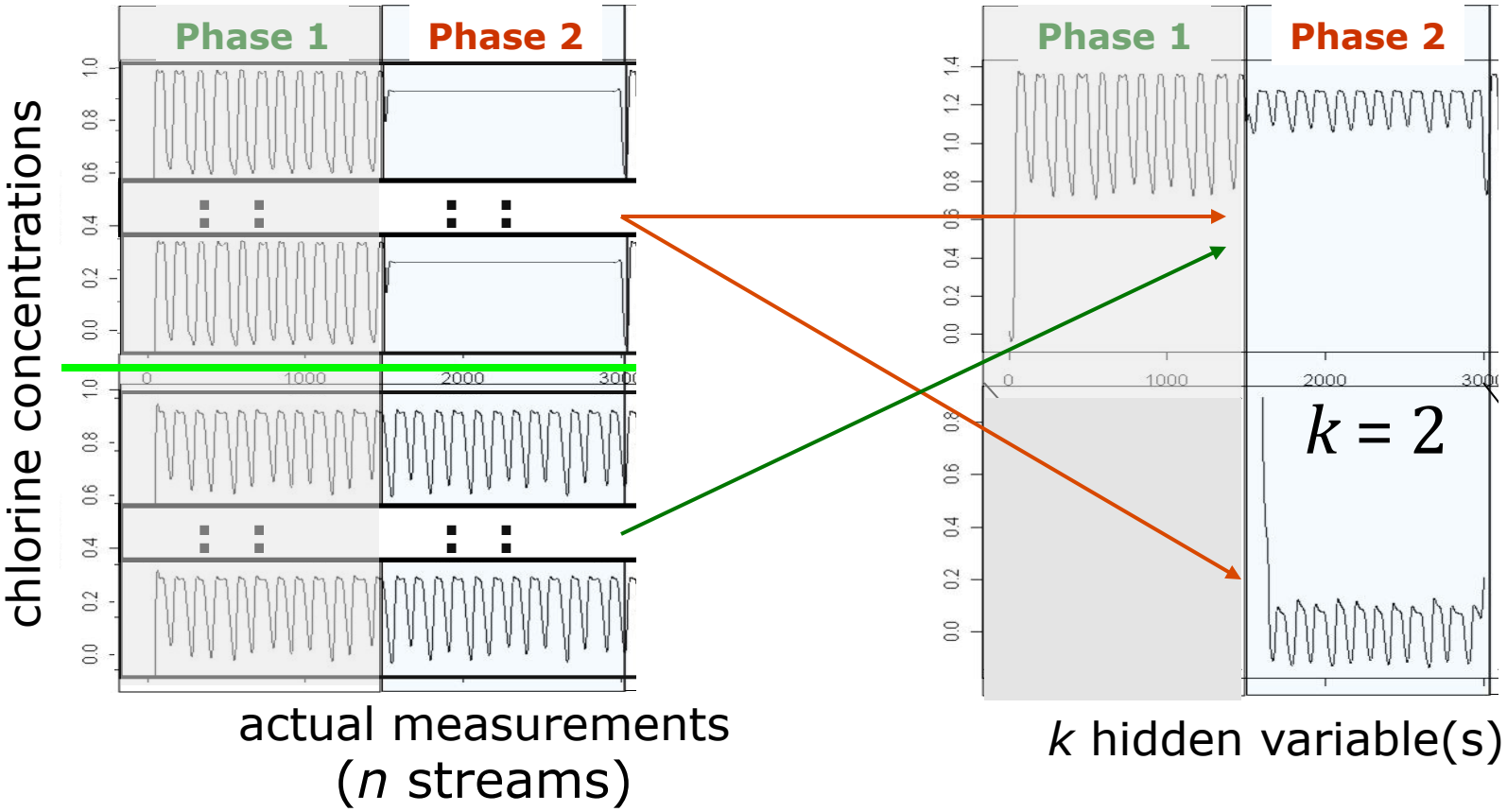


actual measurements
(*n* streams)

k hidden variable(s)

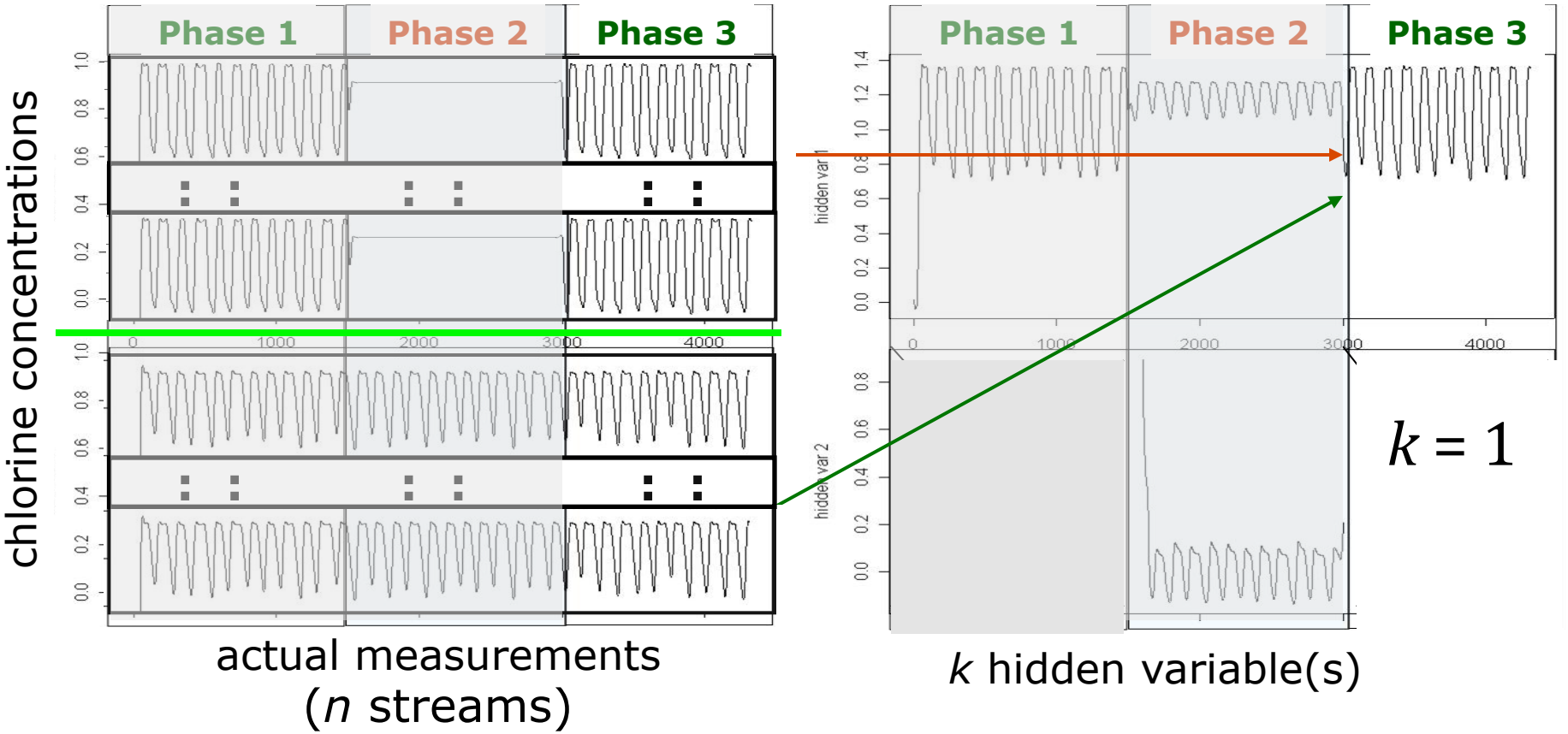
We would like to discover a few "hidden (latent) variables" that summarize the key trends

Motivation



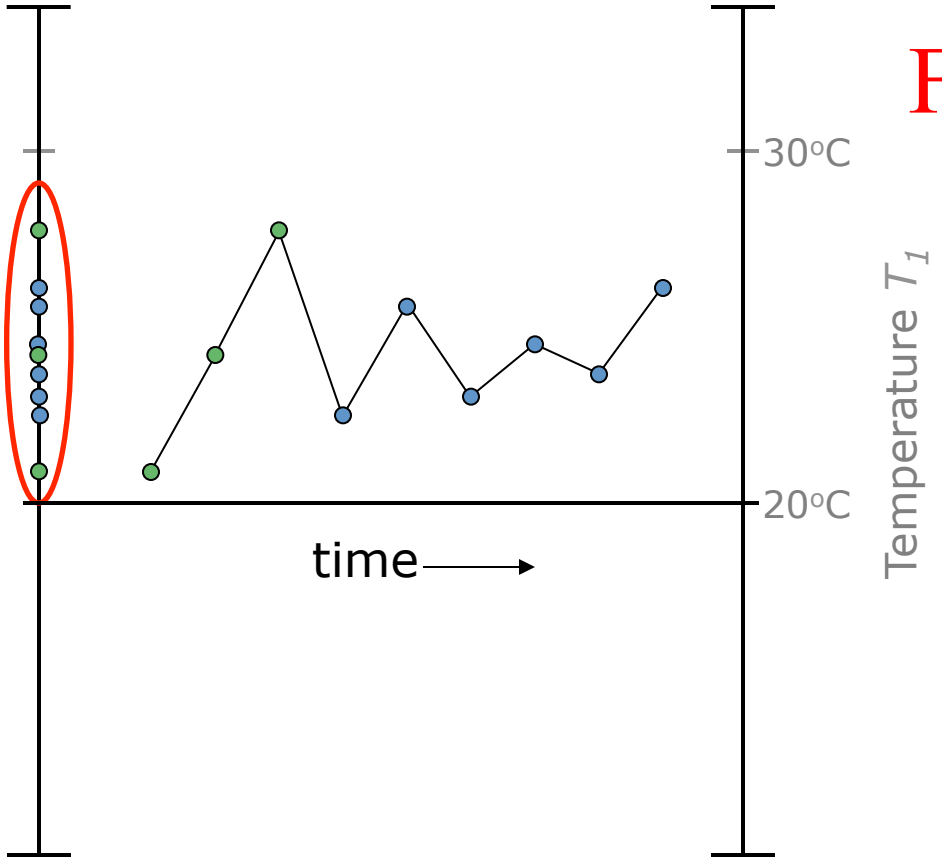
We would like to discover a few "hidden (latent) variables" that summarize the key trends

Motivation



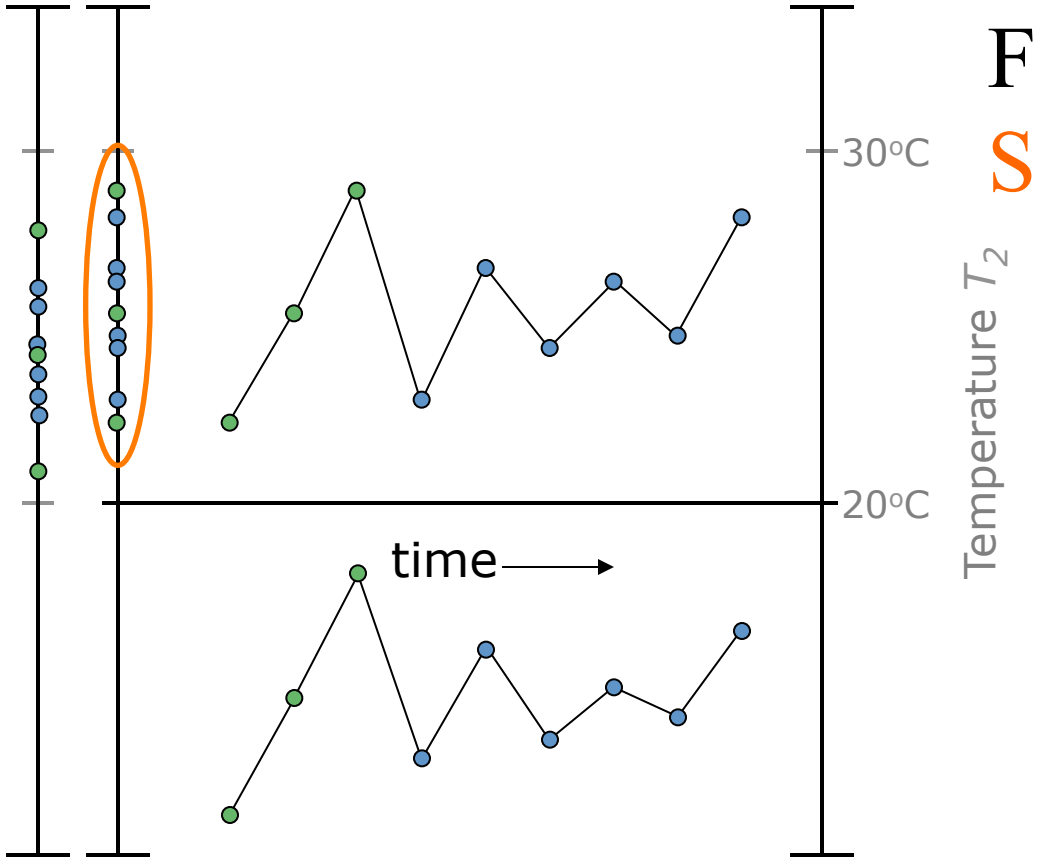
We would like to discover a few "hidden (latent) variables" that summarize the key trends

How to capture correlations?



First sensor

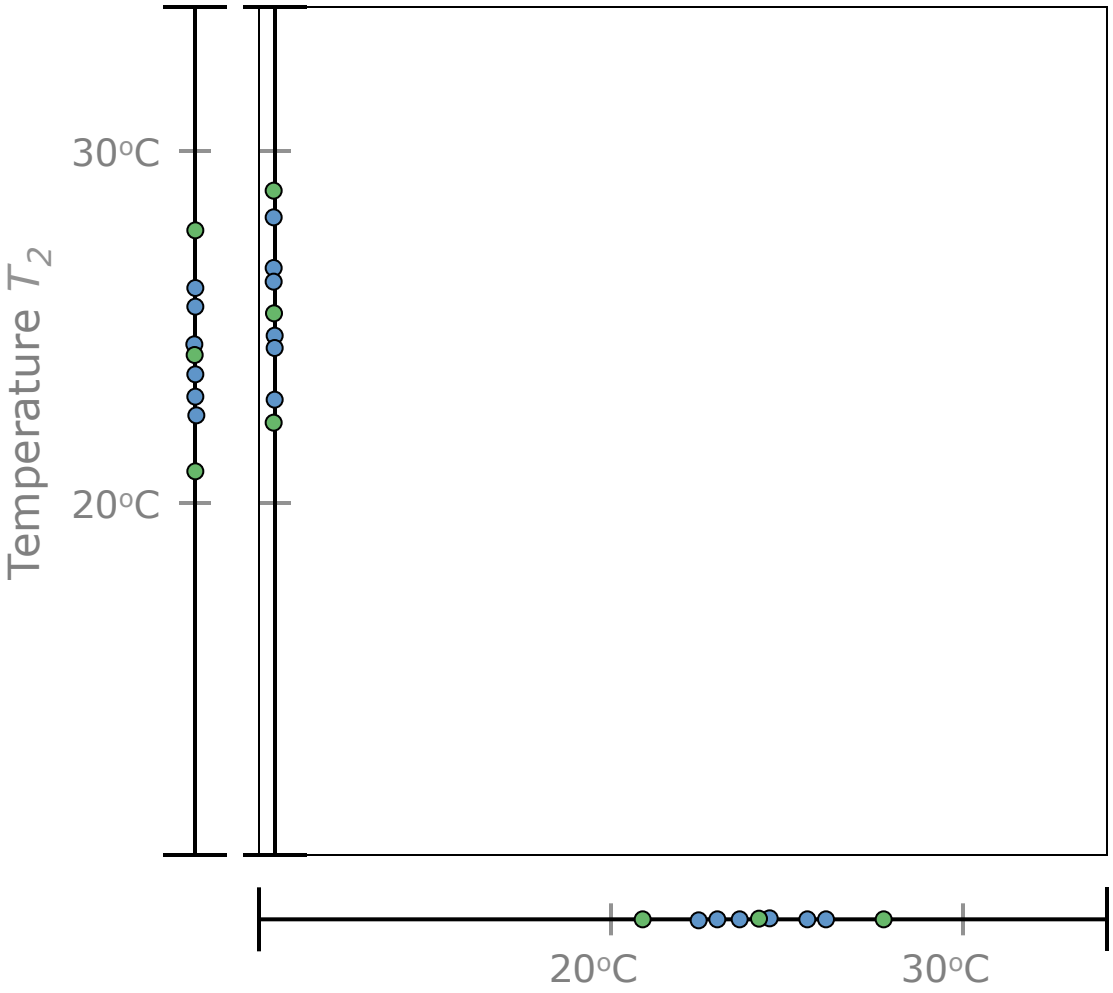
How to capture correlations?



First sensor

Second sensor

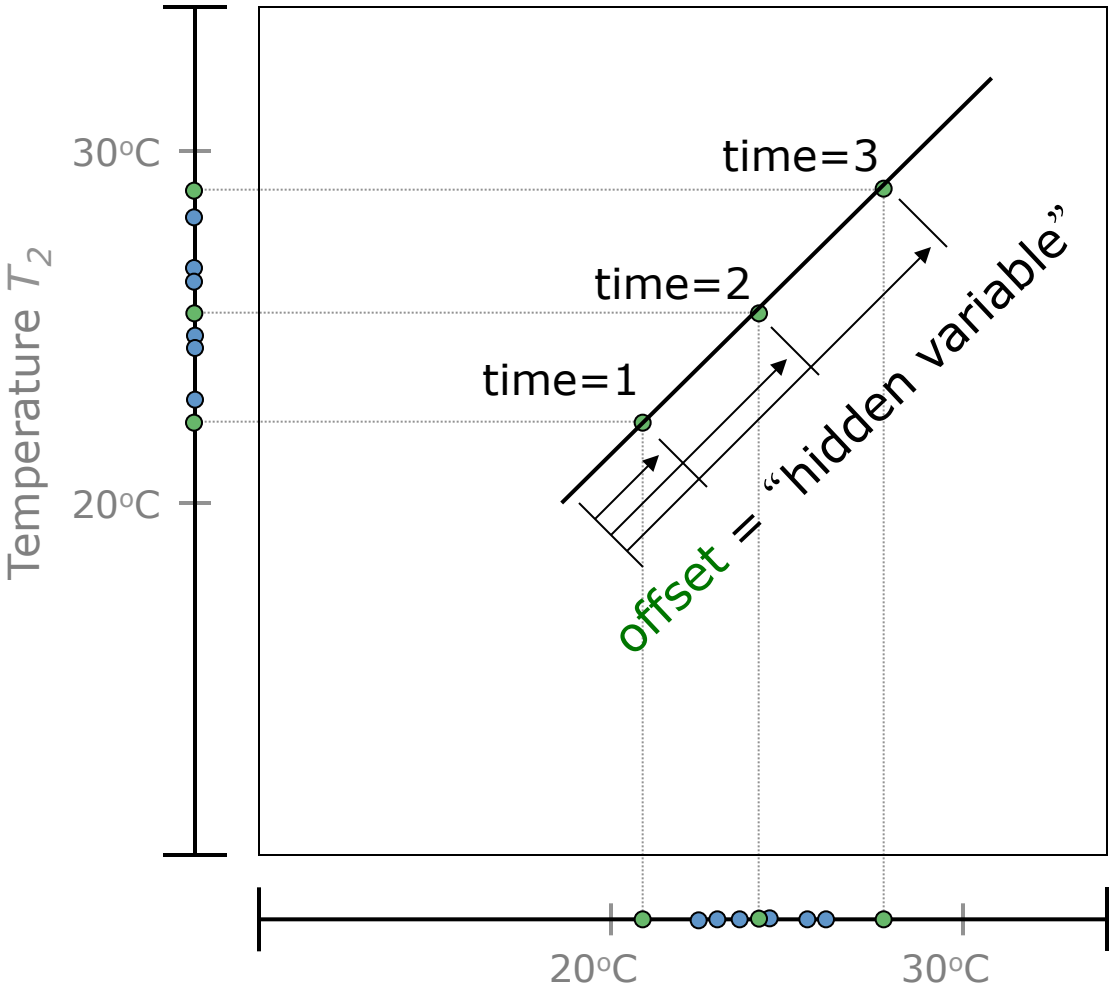
How to capture correlations?



Correlations:

Let's take a closer look at the first three value-pairs…

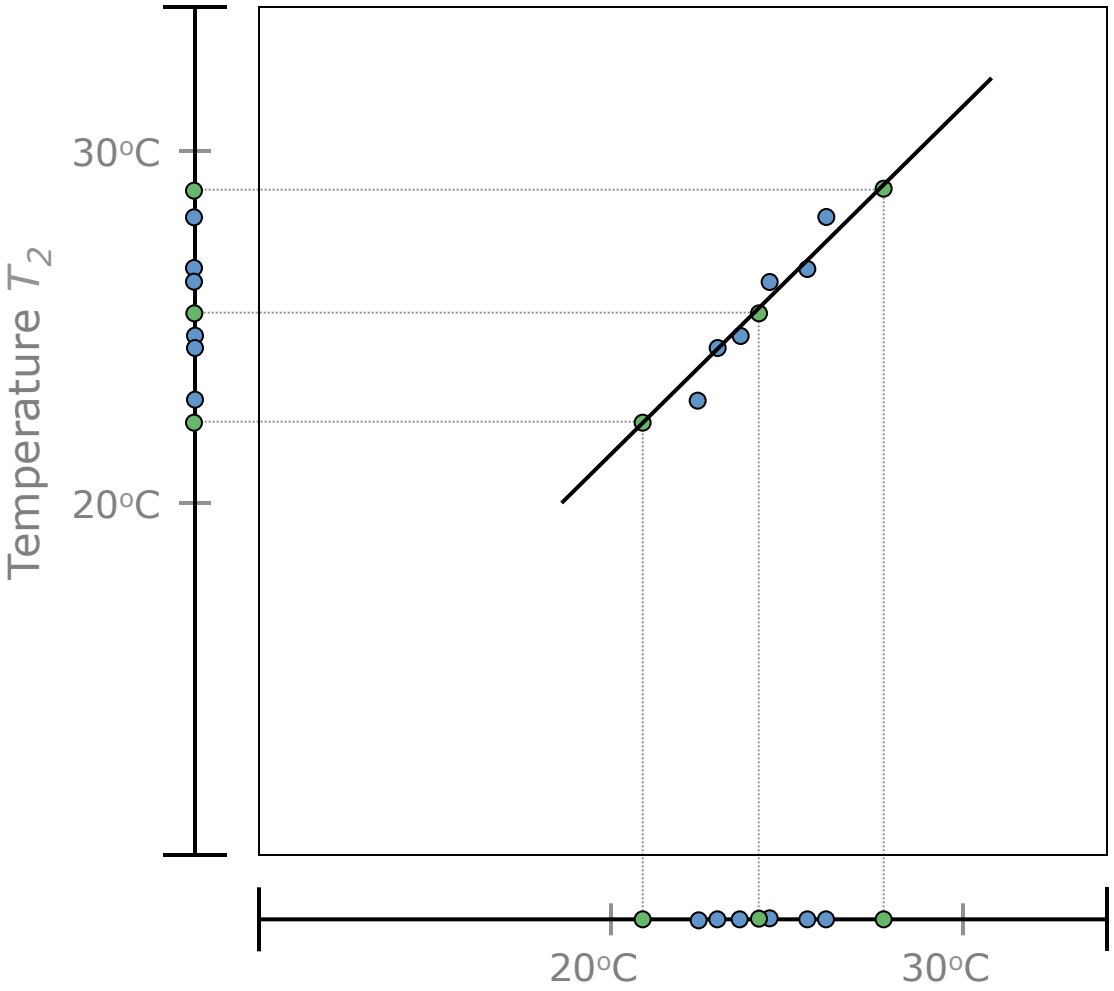
How to capture correlations?



First three lie (almost) on a line in the space of value-pairs…

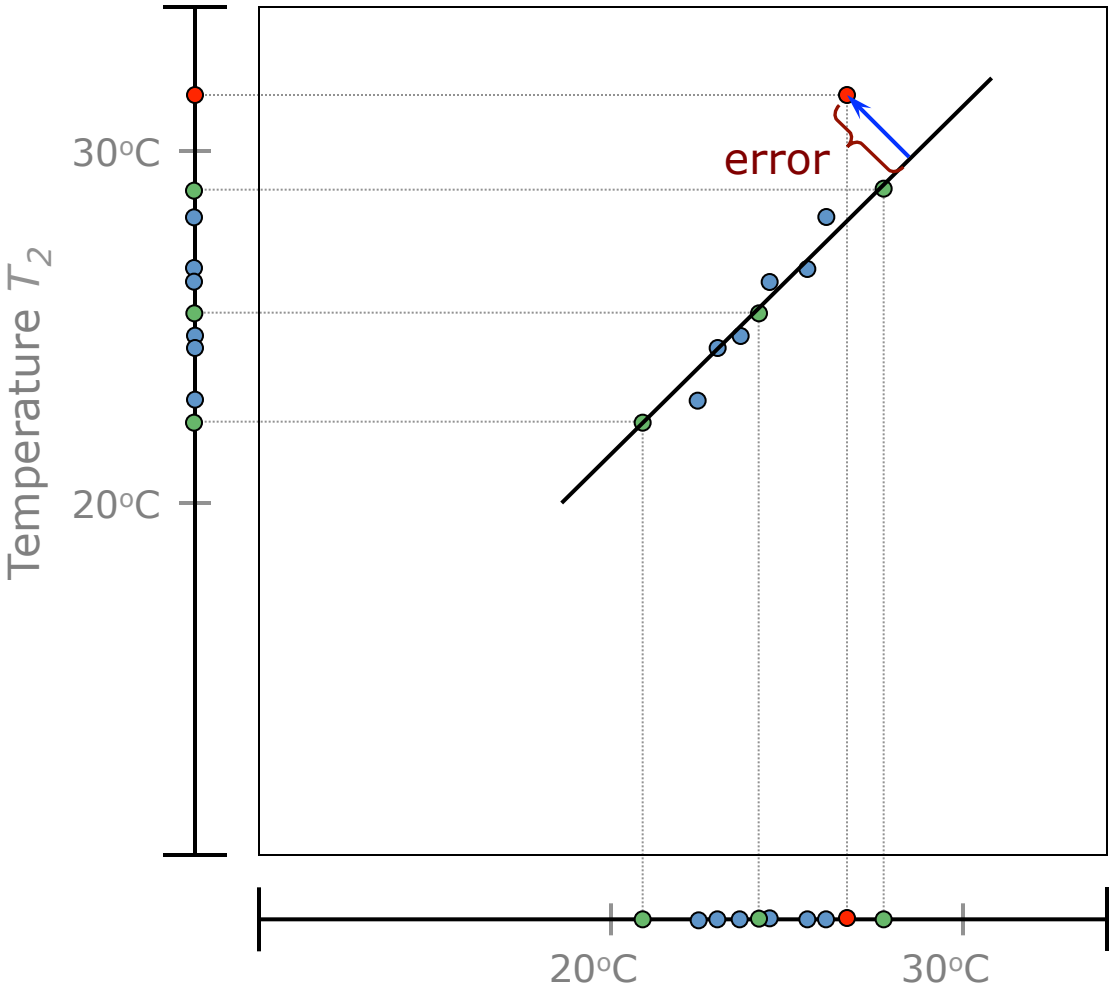
- O(*n*) numbers for the slope, and
- *One* number for each value-pair (offset on line)

How to capture correlations?



Other pairs also follow the same pattern: they lie (approximately) on this line

Incremental update

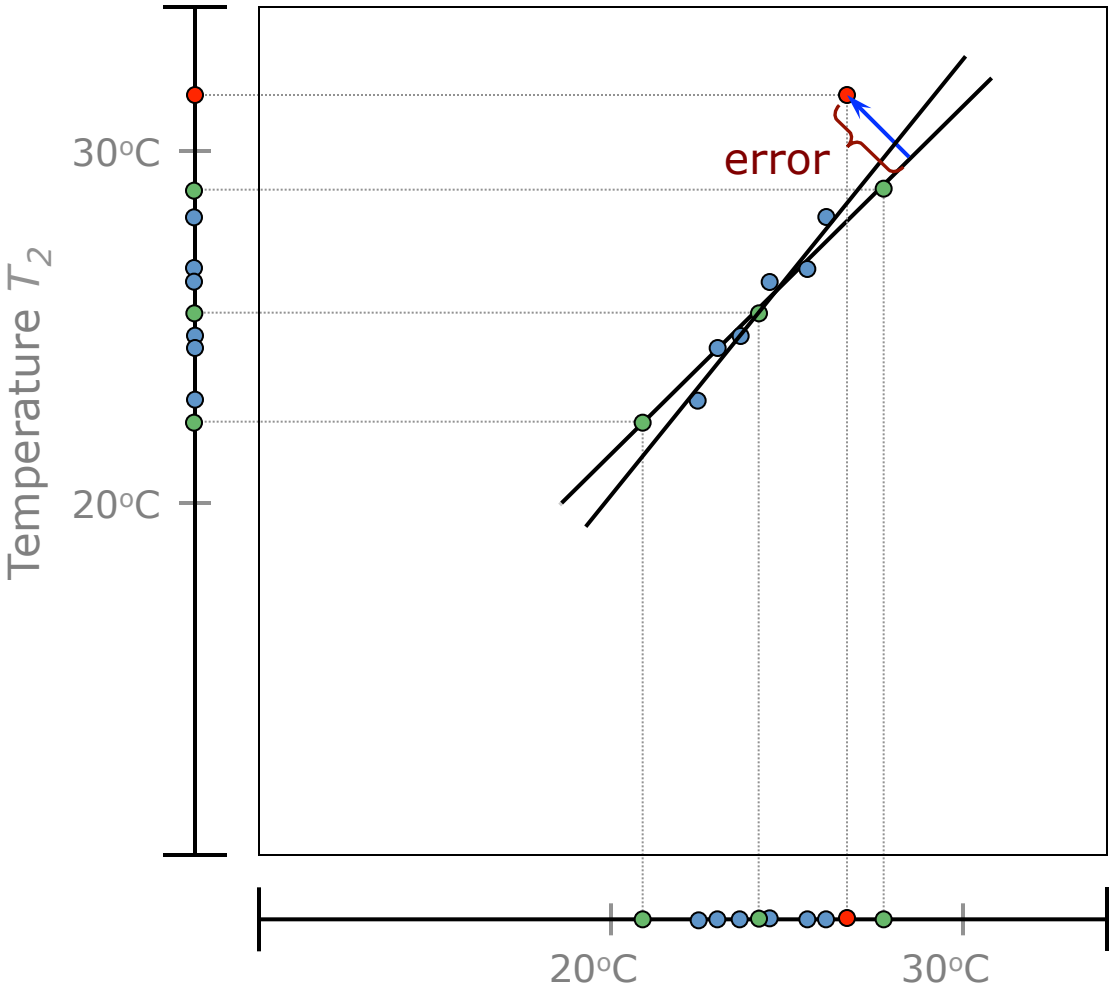


For each new point

- Project onto current line
- Estimate error

New value

Incremental update

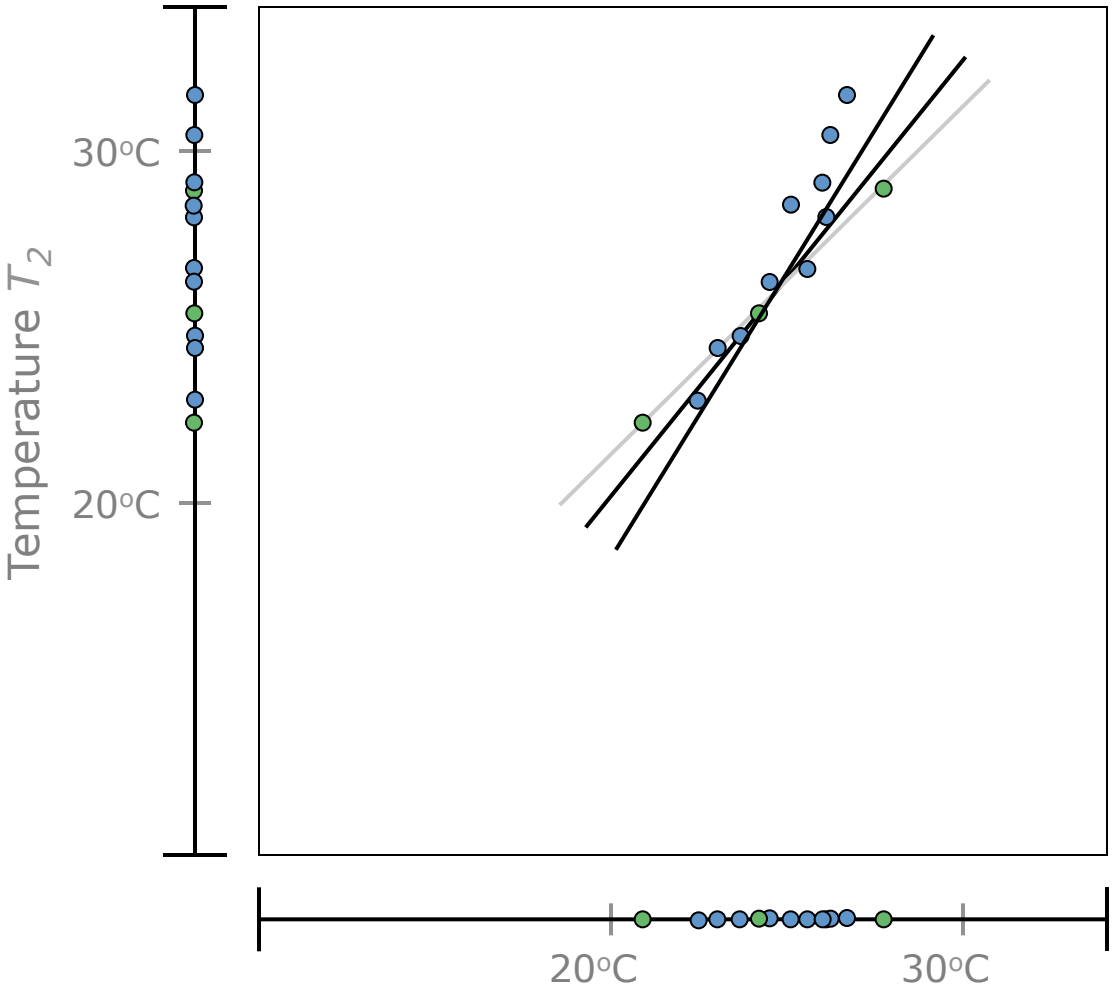


For each new point

- Project onto current line
- Estimate error
- Rotate line in the direction of the error and in proportion to its magnitude

➔O(n) time

Incremental update



For each new point

- Project onto current line
- Estimate error
- Rotate line in the direction of the error and in proportion to its magnitude

Related work

- Wavelet over streams [Gilbert+,vldb01] [Guha +,vldb04]
- Fourier representations [Gilbert+, stoc02]
- KNN [Koudas+, 04] [Korn+, vldb02]
- Histograms [Guha+, stoc01]
- Clustering [Guha+, focs00] [Aggarwal+, vldb03]
- Sketches [Indyk+, vldb00] [Cormode+, J. Algorithms 05]
- …
- …

Related work

- Heavy hitters [Cormode+, vldb03]
- Data embedding [Indyk+, focs00]
- Burst detection [Zhu+, kdd03]
- Segmentation [Keogh+, icdm01]
- Multiple scale analysis [Papadimitriou+, sigmod06]
- Fractal [Korn+, sigmod06]
- Time warping [Sakurai+, icde07]…
- …

Tutorial@PODS'15

Graham Cormode

Tutorial@SIGMOD'15

Divesh Srivastava

Outline

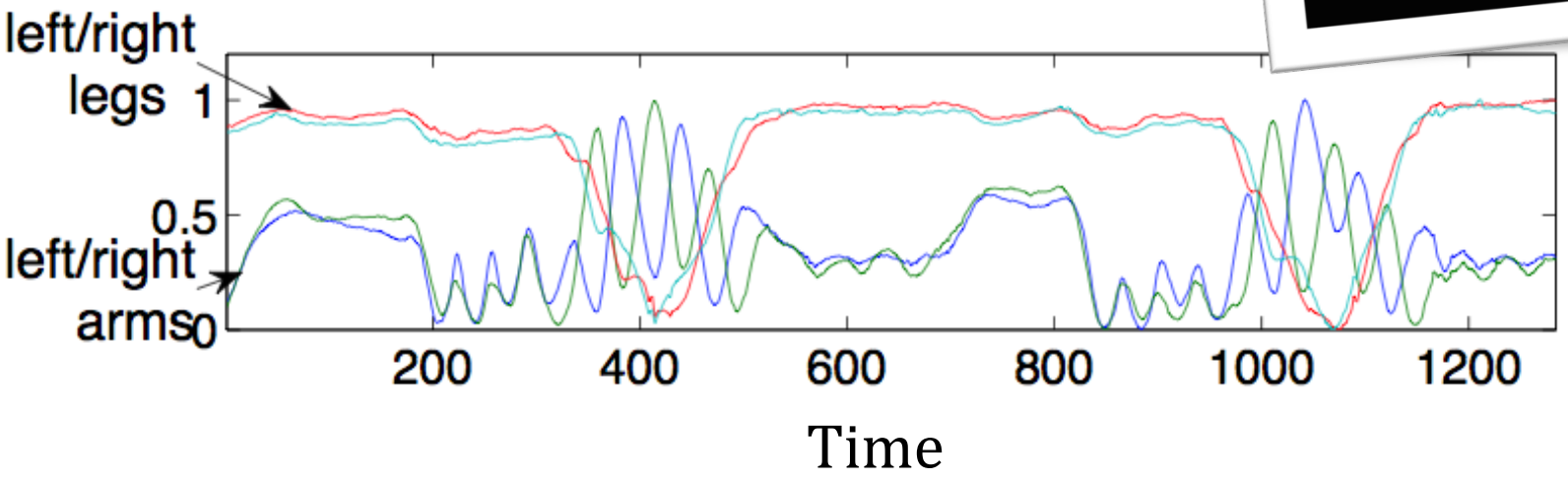
- Motivation
- Similarity Search and Indexing
- Feature extraction
- Streaming pattern discovery
- Linear forecasting
- Automatic mining

Motivation

Given: co-evolving time-series

– e.g., MoCap (leg/arm sensors)

"Chicken dance"

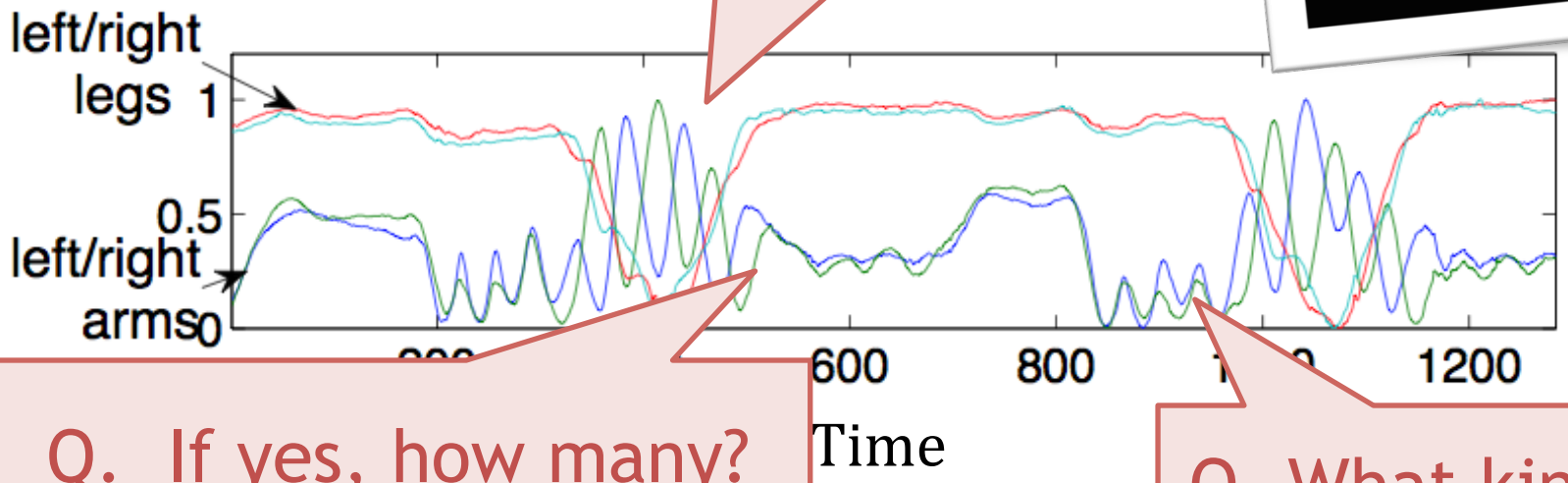


Motivation

Given: co-evolving time-series

– e.g., MoCap (leg/arm sensors)

"Chicken dance"



Q. Any distinct patterns?

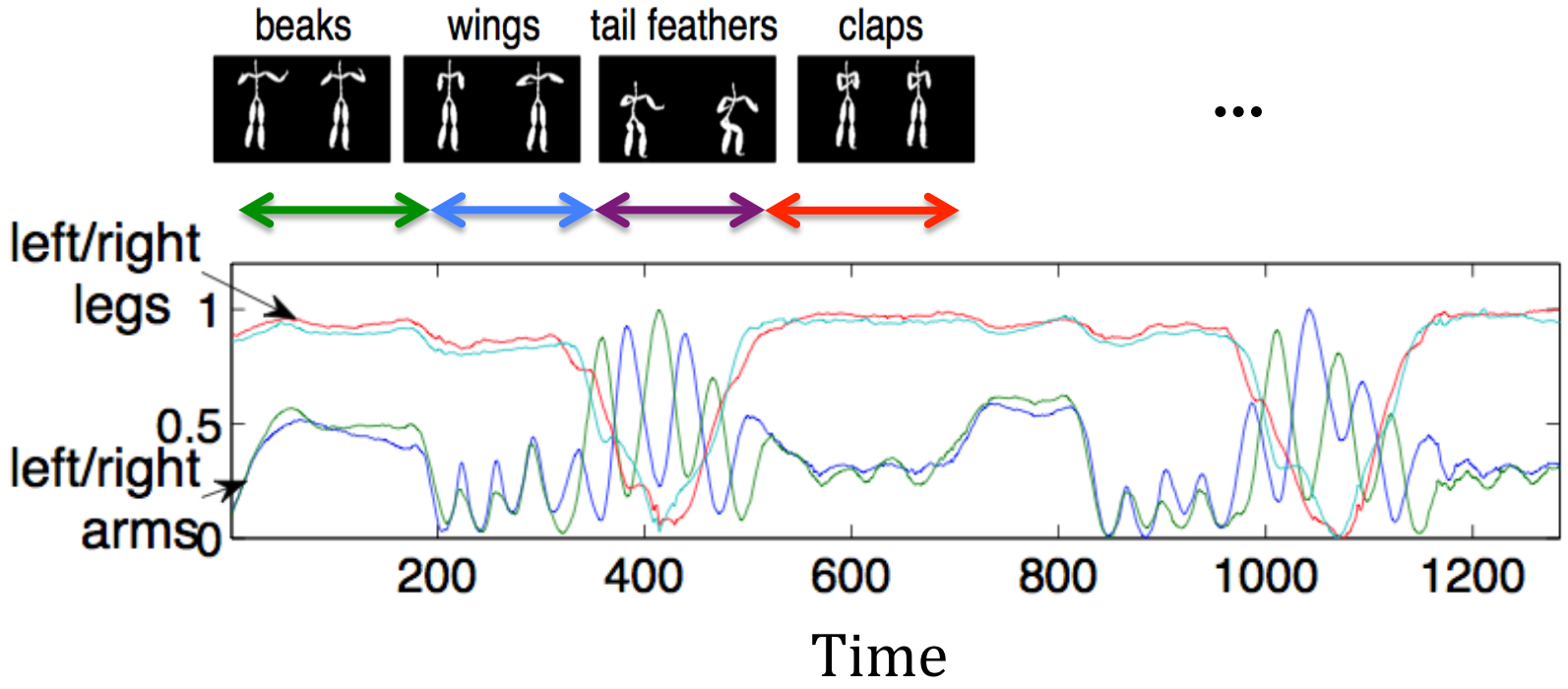
Q. If yes, how many?

Q. What kind?

Motivation

Challenges: co-evolving sequences

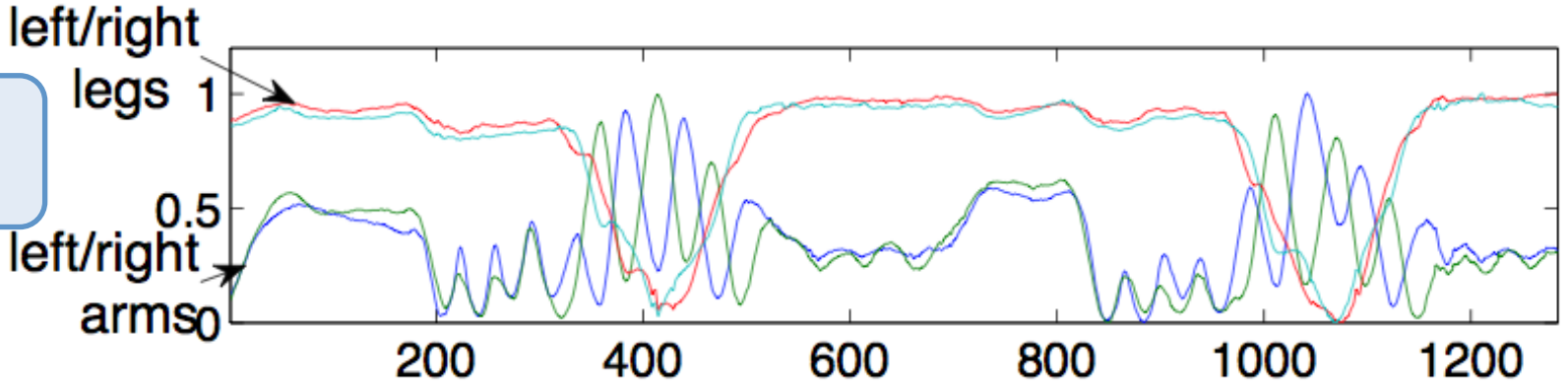
- Unknown # of patterns (e.g., beaks)
- Different durations



Motivation

Goal: find patterns that agree with human intuition

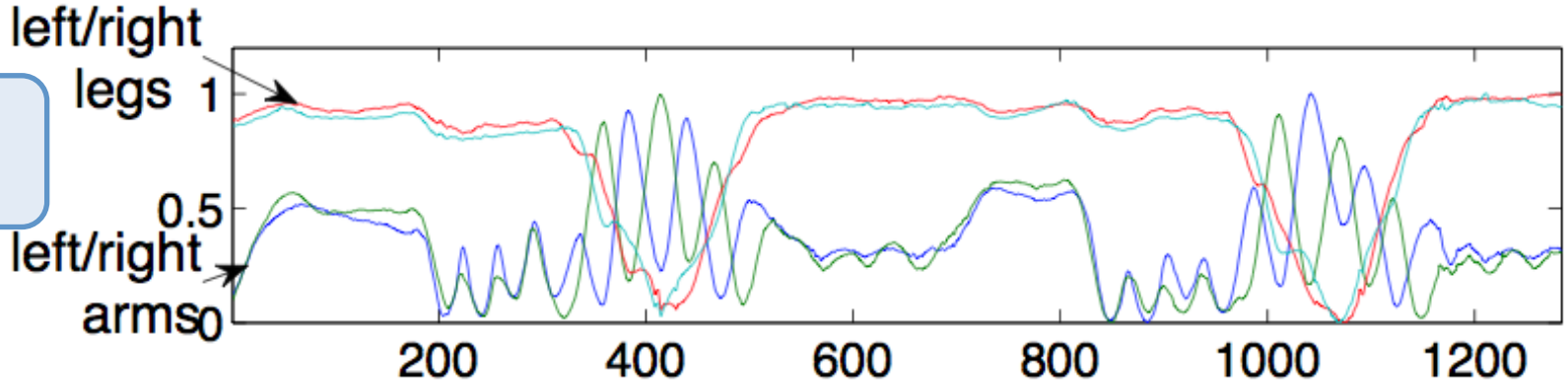
Input



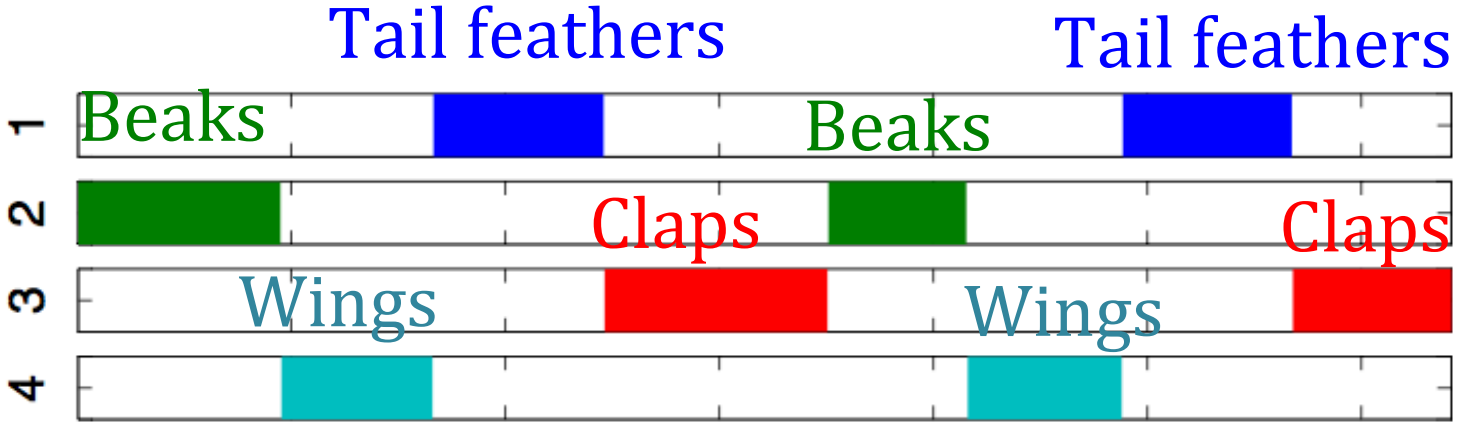
Motivation

Goal: find patterns that agree with human intuition

Input



Output



Motivation

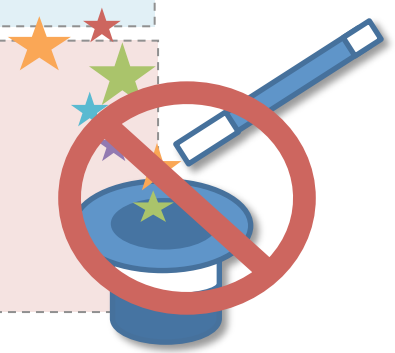
Goal: find patterns that agree with human intuition

Why: Automatic mining

No magic numbers! ... because,

Manual (use magic)

- sensitive to the parameter tuning
- long tuning steps (hours, days, …)



Automatic (no magic numbers)

- no expert tuning required

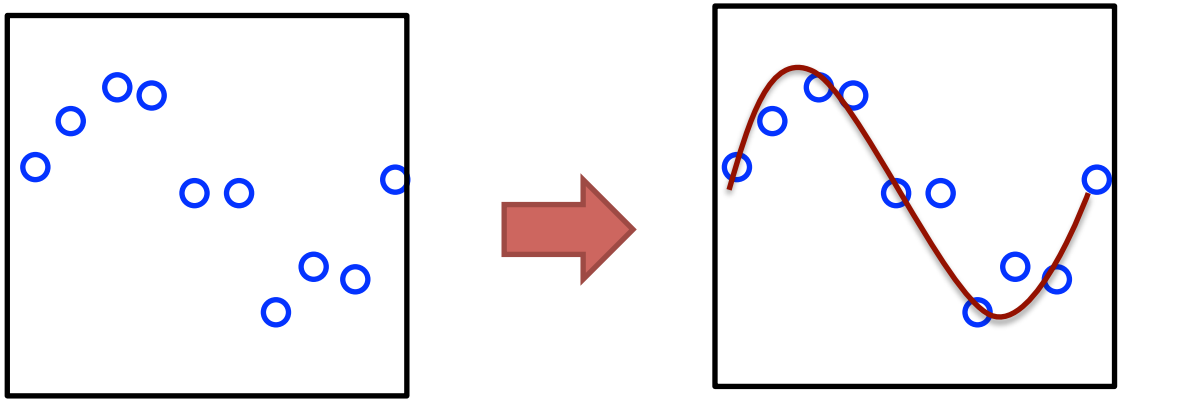
Big data mining:

-> we cannot afford human intervention!!

How: Automatic mining

Goal: fully-automatic modeling

- Given: **data X**
- Find: a compact description **(model M)** of X



Data (X)

Ideal model (M)

Q. How can we find the best model M?

How: Automatic mining

Goal: fully-automatic modeling

- Given: **data X**
- Find: a compact description **(model M)** of X

Answer: MDL!

Data (X) Ideal model (M)

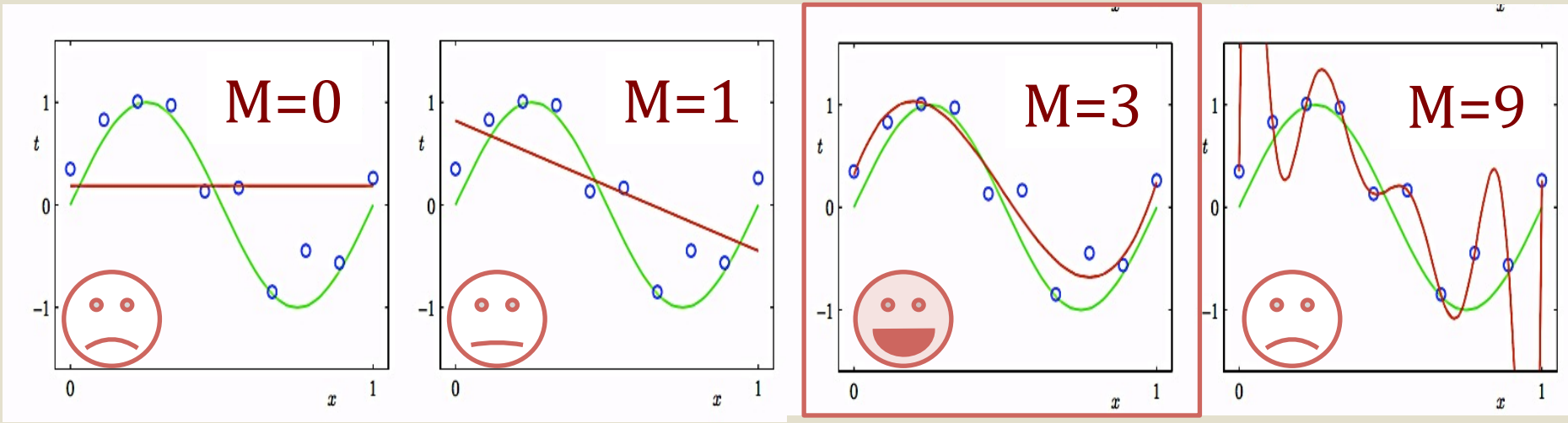
Q. How can we find the best model M?

Solution: MDL (Minimum description length)

Solution: Minimize total encoding cost $!

- Occam's razor (i.e., law of parsimony)
- **Fully automatic** parameter optimization
- No over-fitting

Ideal model



[Bishop: PR&ML]

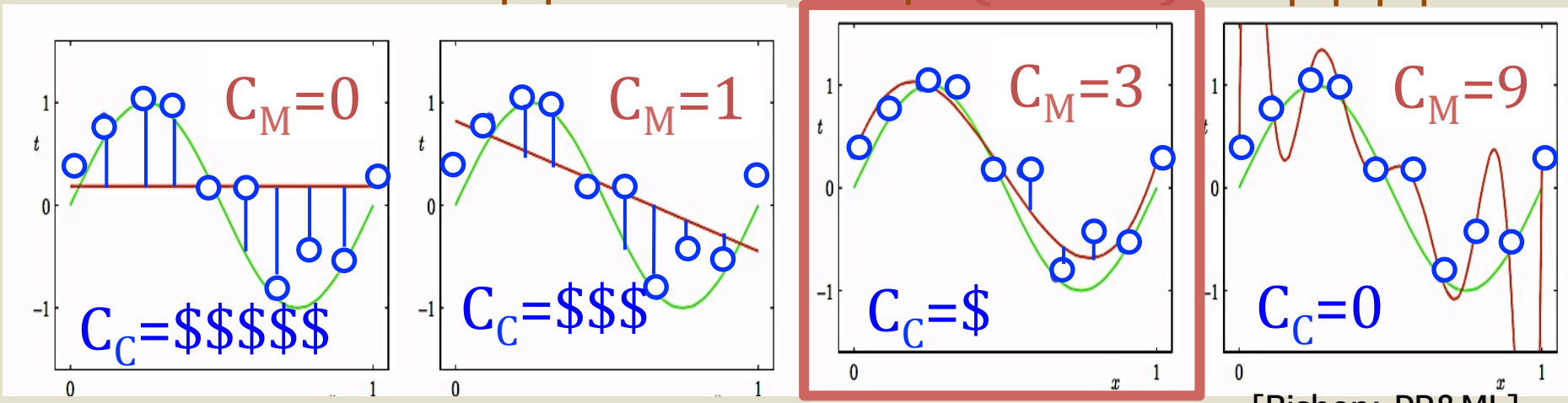
Solution: MDL (Minimum description length)

Solution: Minimize total encoding cost $!

$$\text{Cost}_T(X;M) = \min\left(\boxed{\text{Cost}_M(M)} + \boxed{\text{Cost}_c(X|M)} \right)$$

Total cost | Model cost | Coding cost (error)

$$$** **$$** **$ (Ideal!) **$$$$**



[Bishop: PR&ML]



[Matsubara+ SIGMOD'14]

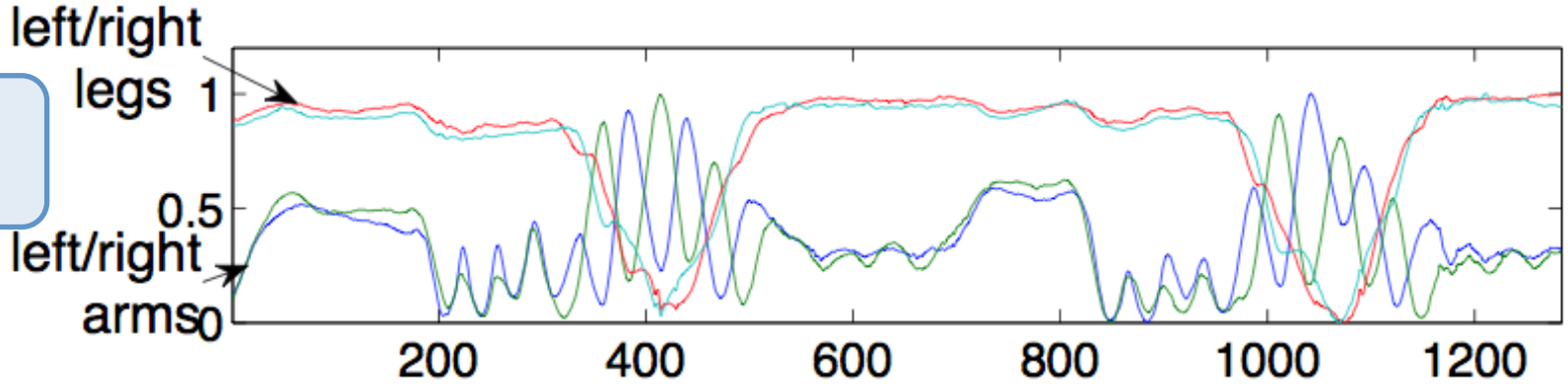
AutoPlait: Automatic Mining of Co-evolving Time Sequences

Yasuko Matsubara (Kumamoto University)
Yasushi Sakurai (Kumamoto University),
Christos Faloutsos (CMU)

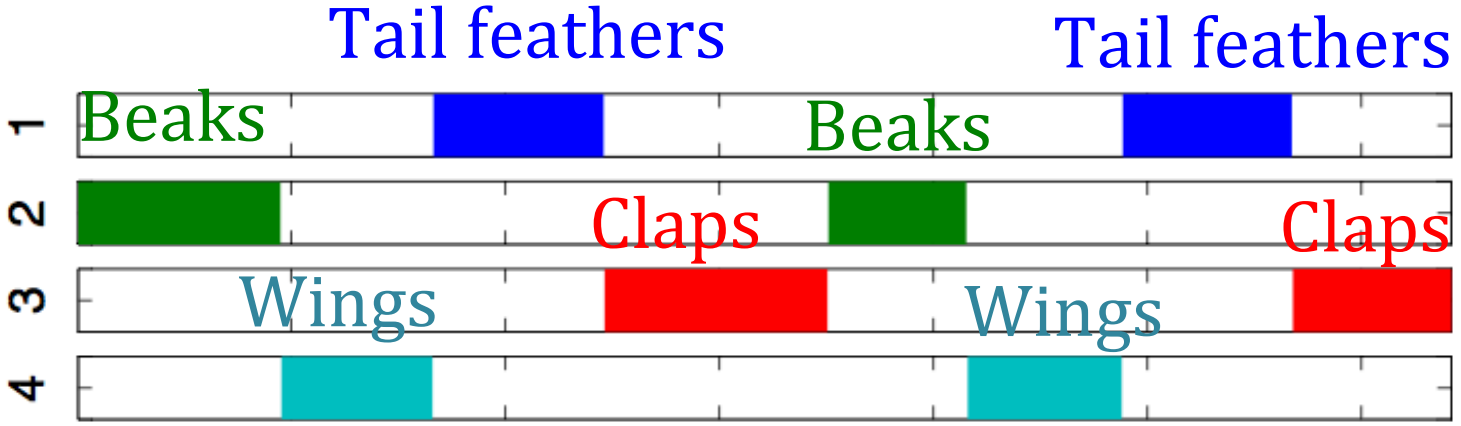
Problem definition

Goal: find patterns that agree with human intuition

Input



Output



http://www.cs.kumamoto-u.ac.jp/~yasuko/TALKS/15-SIGMOD-tut/

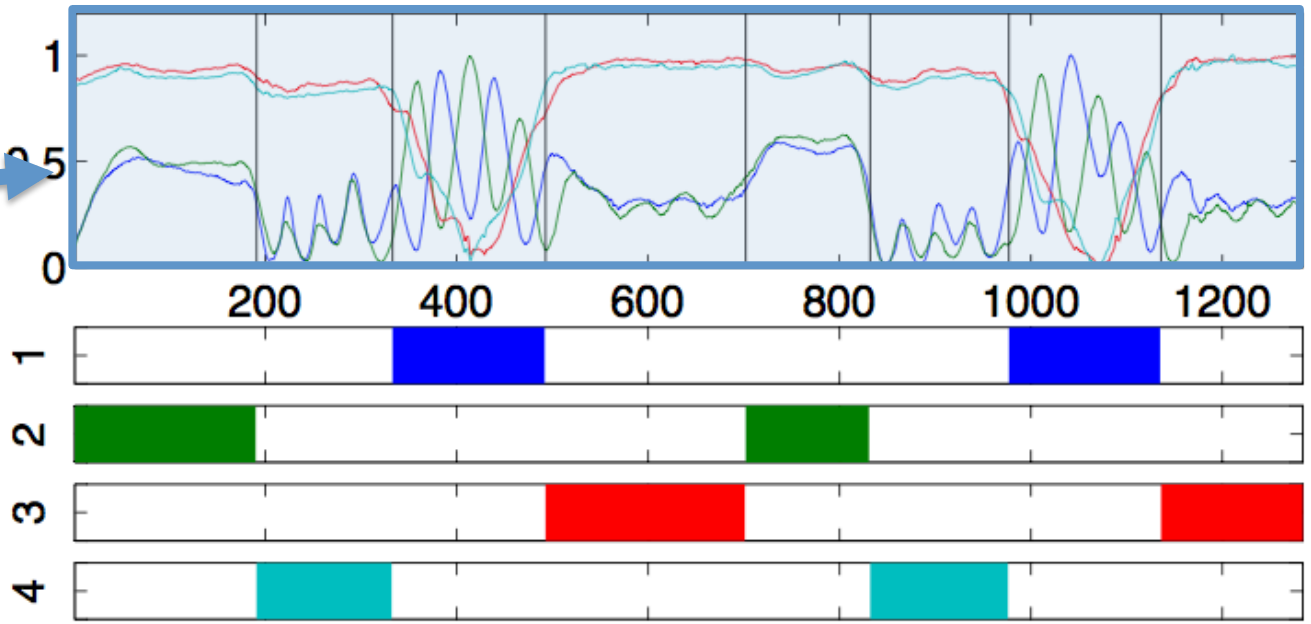
Problem definition

- Bundle : set of d co-evolving sequences

given

$$\underset{d \times n}{X} = \{x_1, \ldots, x_n\}$$

Bundle X (d=4)



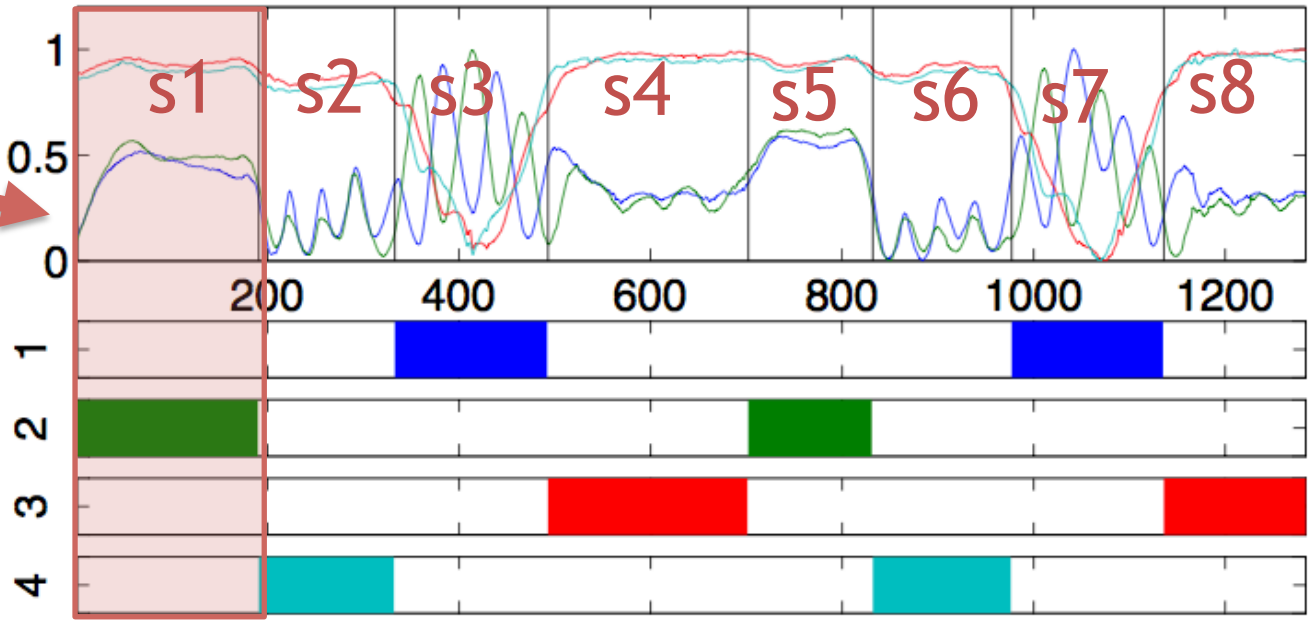
Problem definition

- Segment: convert X -> m segments, S

hidden

$$S = \{s_1, \ldots, s_m\}$$

Segment (m=8)



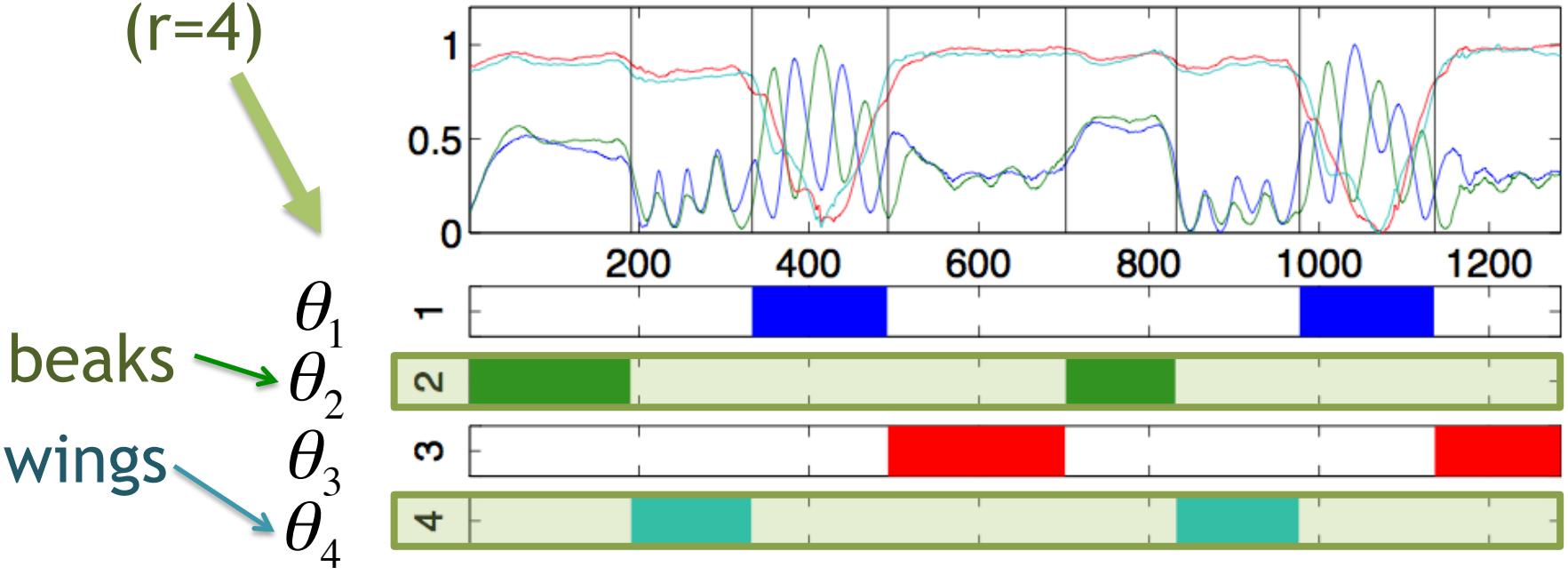
Problem definition

- Regime: segment groups: $\Theta = \{\theta_1, \theta_2, \ldots \theta_r, \Delta_{r \times r}\}$

hidden

Regimes (r=4)

θ_r : model of regime r

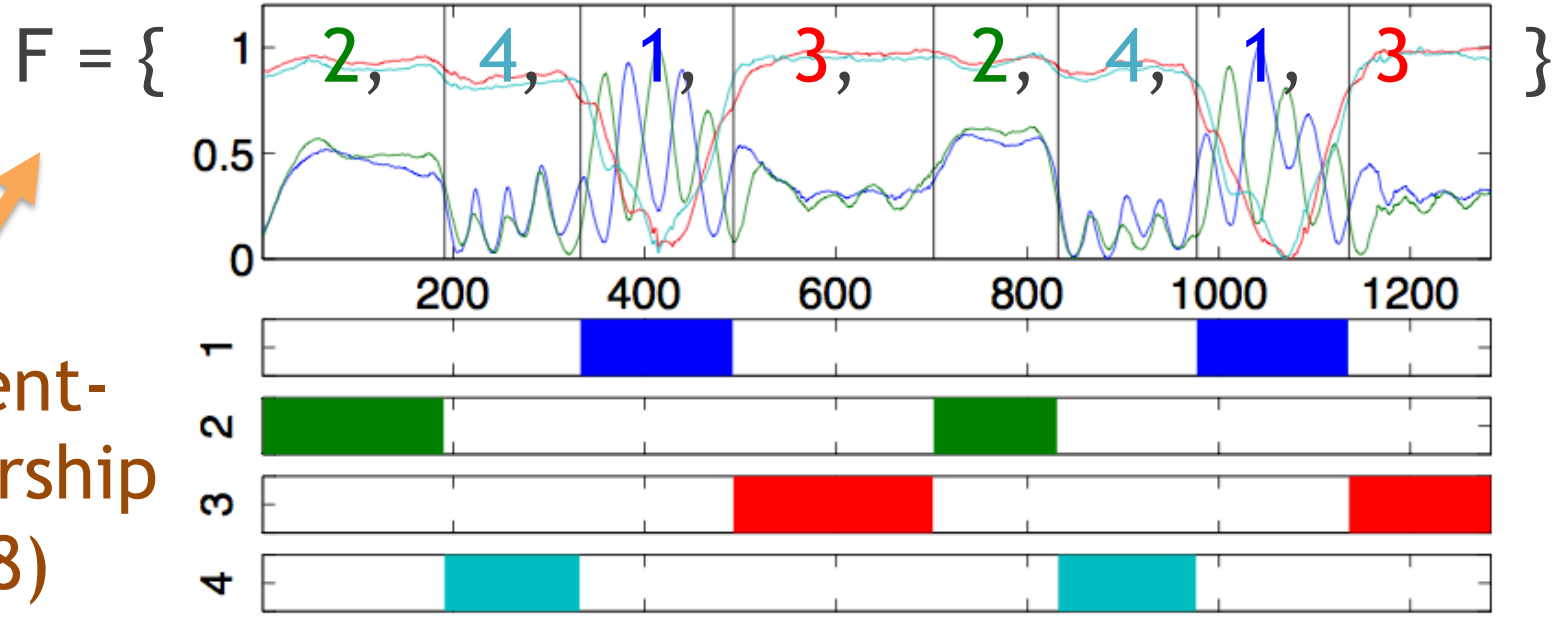


Problem definition

- Segment-membership: assignment

hidden

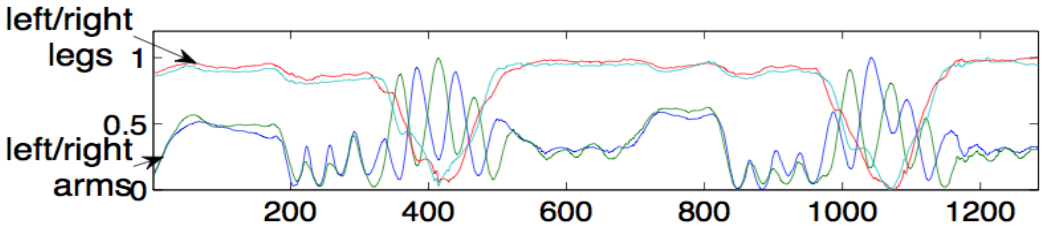
$$F = \{f_1, \ldots, f_m\}$$



Problem definition

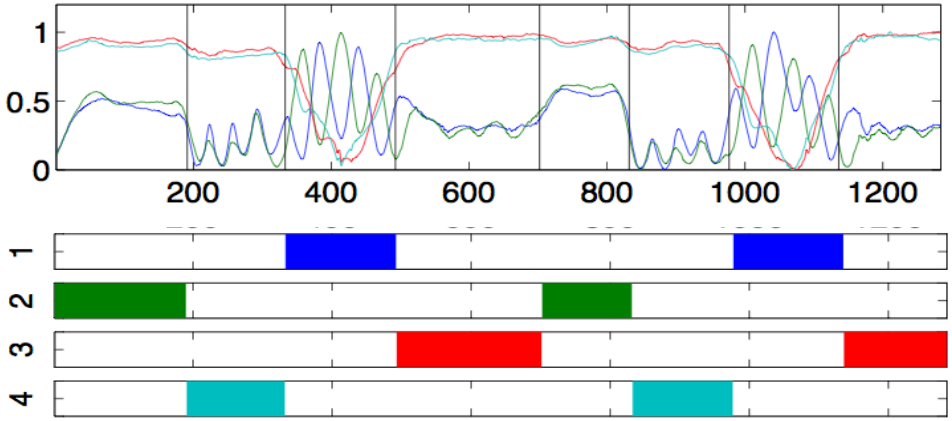
- Given: bundle X

$$X = \{x_1, \ldots, x_n\}$$



- Find: compact description C of X

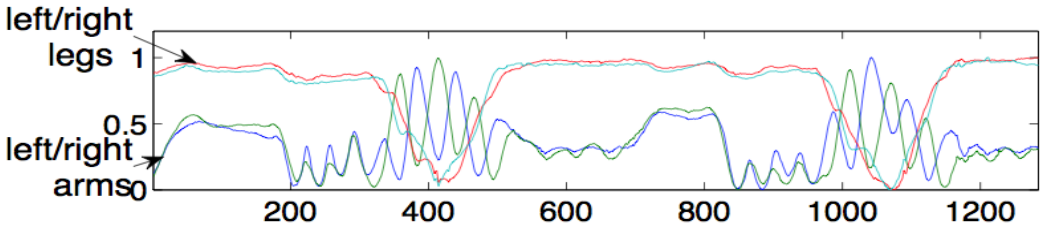
$$C = \{m, r, S, \Theta, F\}$$



Problem definition

- Given: bundle X

$$X = \{x_1, \ldots, x_n\}$$



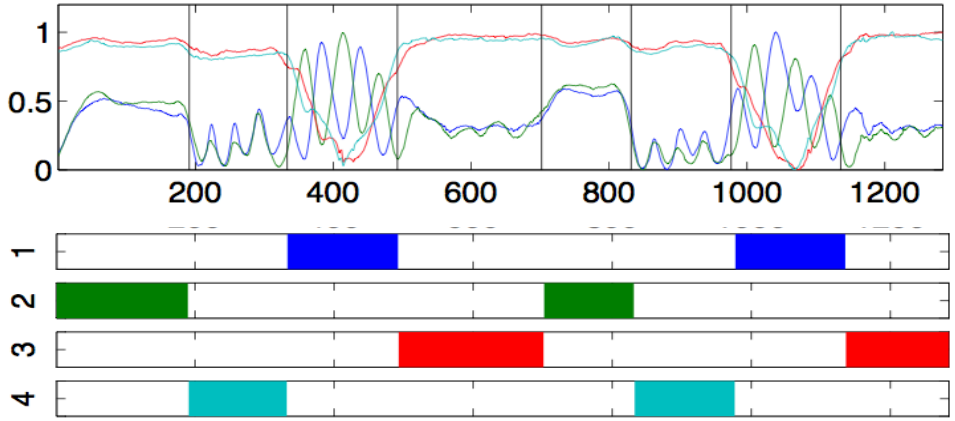
- Find: compact description C of X

$$C = \{m, r, S, \Theta, F\}$$

m segments

r regimes

Segment-membership



Main ideas

Goal: compact description of X

$$C = \{m, r, S, \Theta, F\}$$

without user intervention!!

Challenges:

Q1. How to generate 'informative' regimes ?

Q2. How to decide # of regimes/segments ?

Main ideas

Goal: compact description of X

$$C = \{m, r, S, \Theta, F\}$$

without user intervention!!

Challenges:

Q1. How to generate 'informative' regimes ?

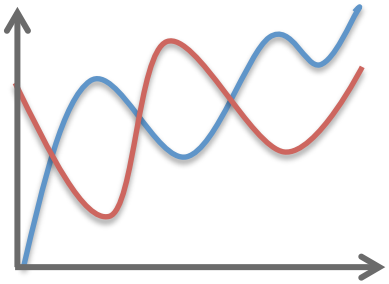
Idea (1): Multi-level chain model

Q2. How to decide # of regimes/segments ?

Idea (2): Model description cost

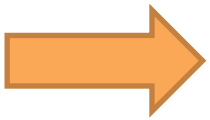
Idea (1): MLCM: multi-level chain model

Q1. How to generate 'informative' regimes ?



Sequences

Model



beaks

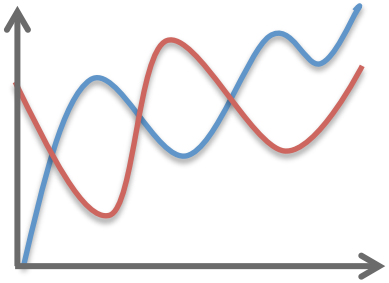
claps

wings

Regimes

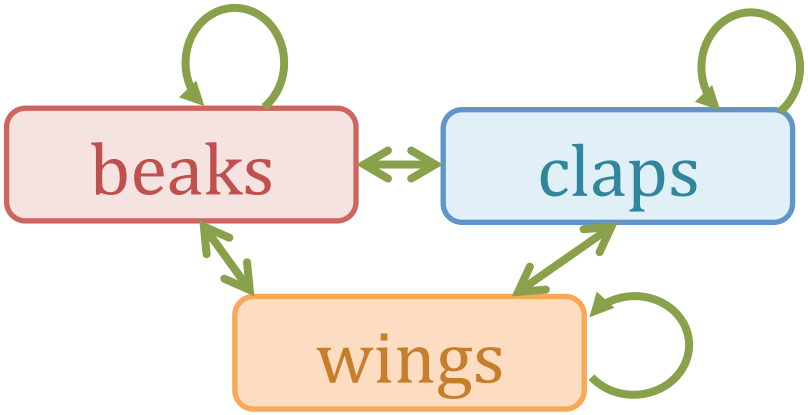
Idea (1): MLCM: multi-level chain model

Q1. How to generate 'informative' regimes ?



Sequences

Model



Regimes

Idea (1): Multi-level chain model

– HMM-based probabilistic model

– with "across-regime" transitions

Details

Idea (1): MLCM: multi-level chain model

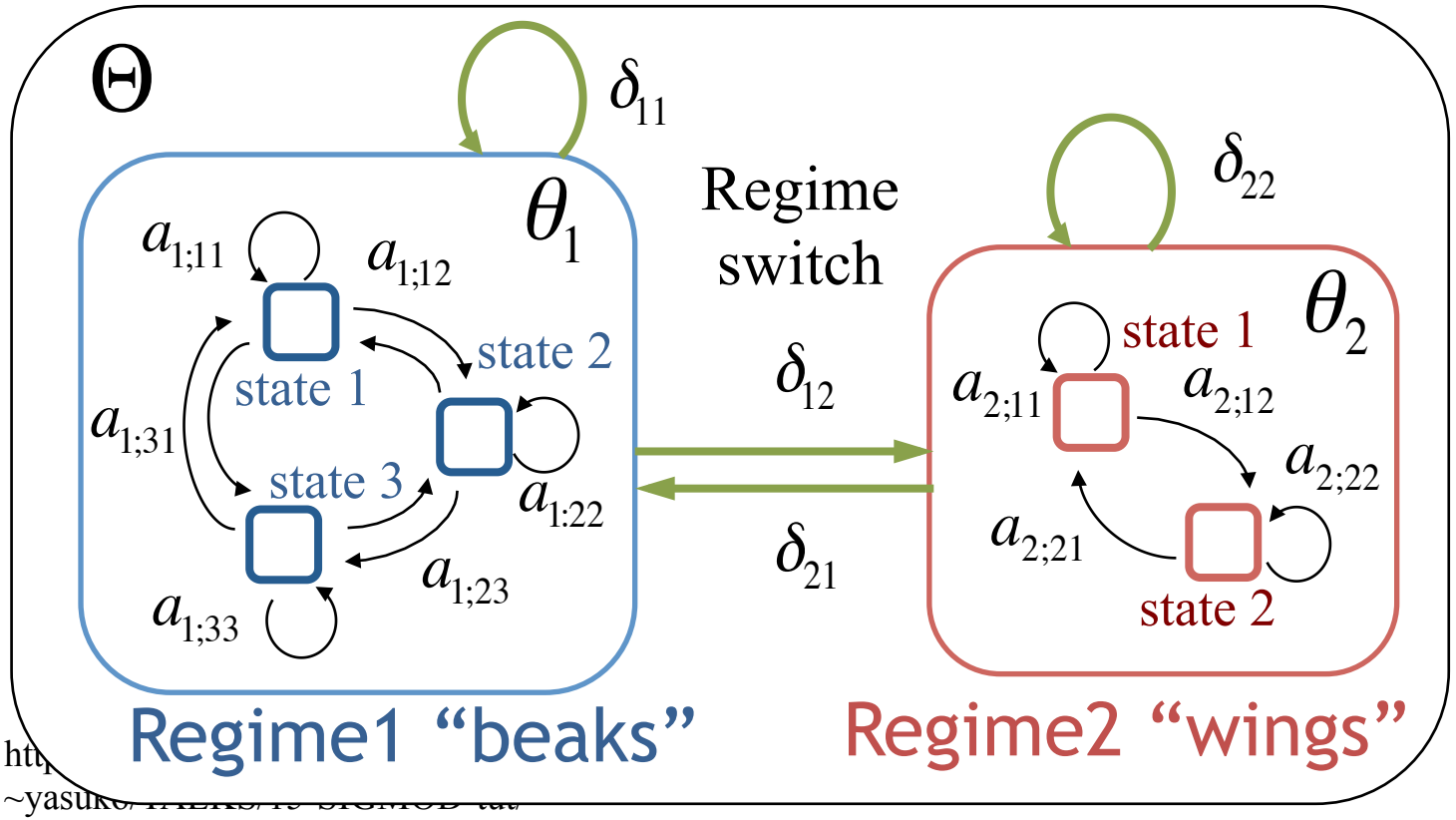
CMU CS

$$\Theta = \{\theta_1, \theta_2, \ldots \theta_r, \Delta_{r\times r}\} \qquad (\theta_i = \{\pi, A, B\})$$

r regimes (HMMs)

across-regime transition prob.

Single HMM parameters



Regimes r=2

Regime 1 (k=3)

Regime 2 (k=2)

Idea (2): model description cost

Q2. How to decide # of regimes/segments ?

Idea (2): Model description cost

- Minimize encoding cost
- find "optimal" # of segments/regimes

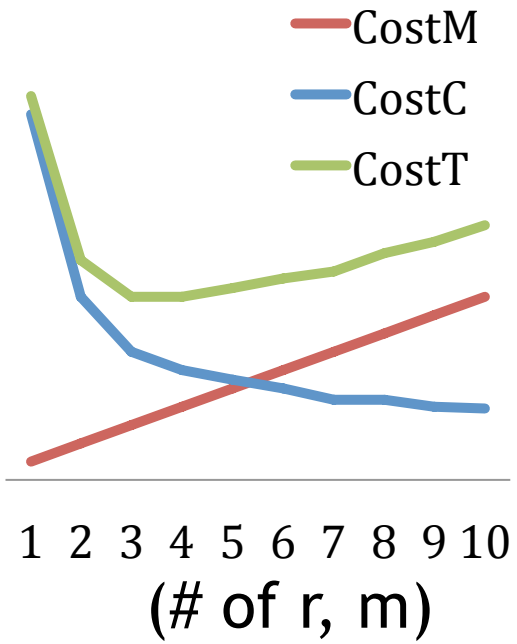
Idea (2): model description cost

Idea: Minimize encoding cost!

$$\min \left(\text{Cost}_M(M) + \text{Cost}_c(X \mid M) \right)$$

Model cost

Coding cost



Good compression

Good description

Details

Idea (2): model description cost

Total cost of bundle X, given C

$$C = \{m, r, S, \Theta, F\}$$

$$Cost_T(\boldsymbol{X}; \mathcal{C}) = Cost_T(\boldsymbol{X}; m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F})$$
$$= \log^*(n) + \log^*(d) + \log^*(m) + \log^*(r) + m\log(r)$$
$$+ \sum_{i=1}^{m-1} \log^* |s_i| + Cost_M(\boldsymbol{\Theta}) + Cost_C(\boldsymbol{X}|\boldsymbol{\Theta}) \qquad (6)$$

Details

Idea (2): model description cost

CMU CS

Total cost of bundle X, given C

$$C = \{m, r, S, \Theta, F\}$$

duration/ dimensions

# of segments/ regimes

segment-membership F

$$Cost_T(\boldsymbol{X}; \mathcal{C}) = Cost_T(\boldsymbol{X}; m, r, \mathcal{S}, \boldsymbol{\Theta}, \mathcal{F})$$

$$= \log^*(n) + \log^*(d) + \log^*(m) + \log^*(r) + m \log(r)$$

$$+ \sum_{i=1}^{m-1} \log^* |s_i| + Cost_M(\boldsymbol{\Theta}) + Cost_C(\boldsymbol{X}|\boldsymbol{\Theta}) \quad (6)$$

segment lengths

Model description cost of Θ

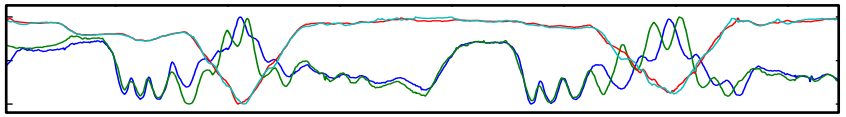
Coding cost of X given Θ

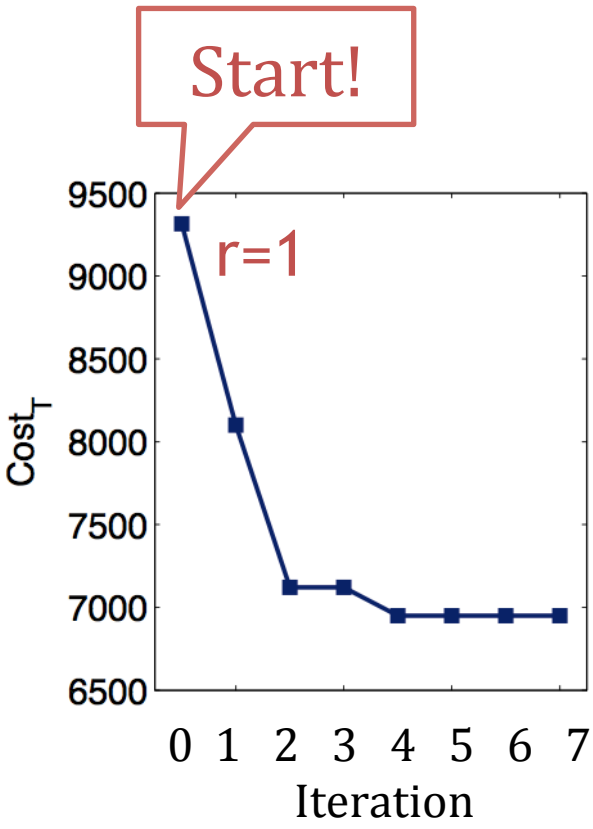
AutoPlait

Overview

Iteration 0
r=1, m=1

X

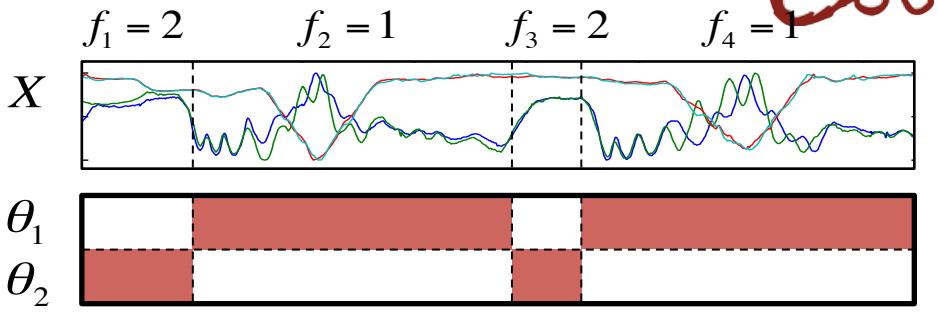


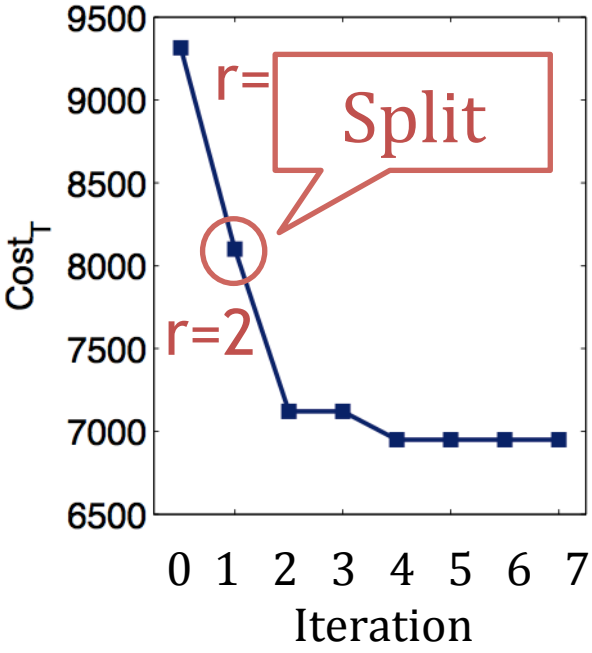


AutoPlait

Overview

Iteration 1
r=2, m=4

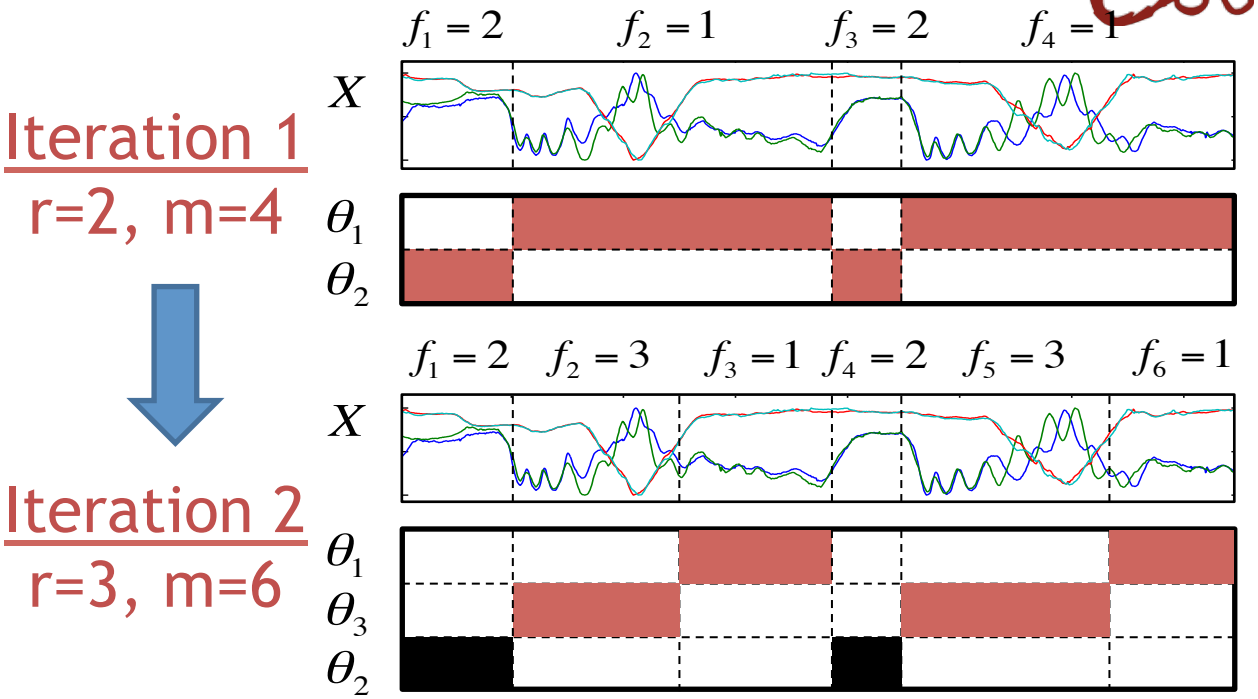


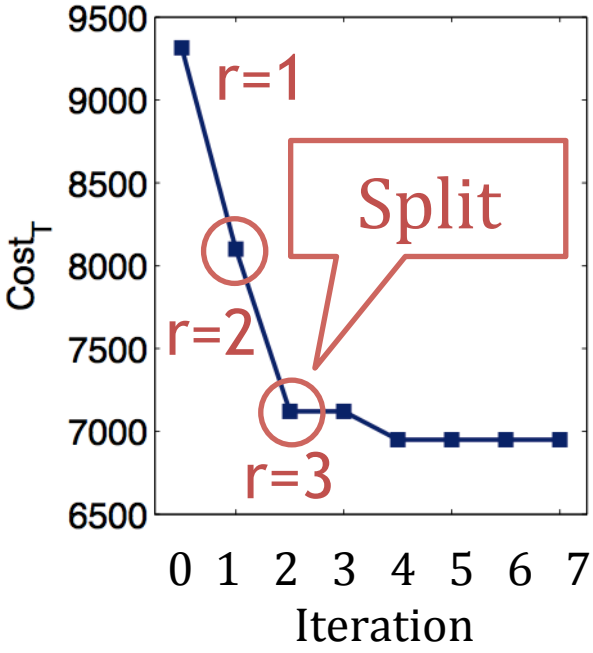


http://www.cs.kumamoto-u.ac.jp/~yasuko/TALKS/15-SIGMOD-tut/

AutoPlait

Overview

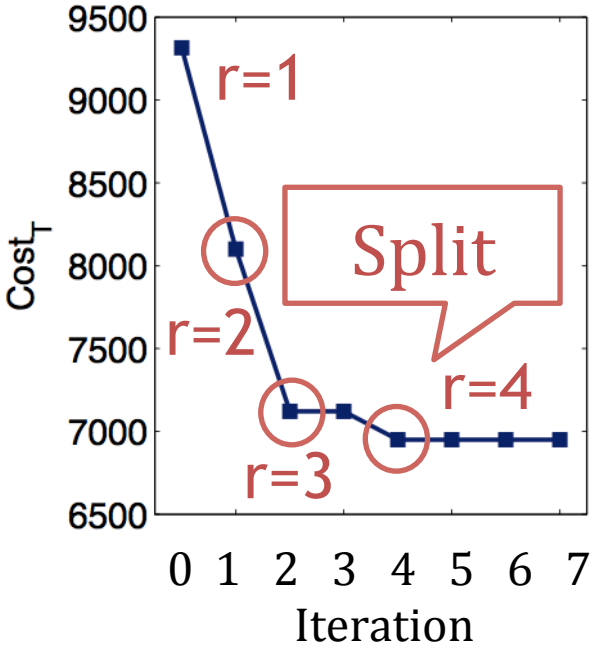




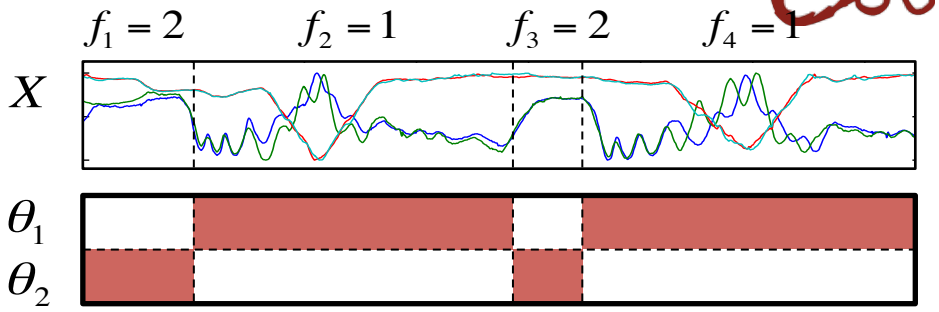
http://www.cs.kumamoto-u.ac.jp/~yasuko/TALKS/15-SIGMOD-tut/

AutoPlait

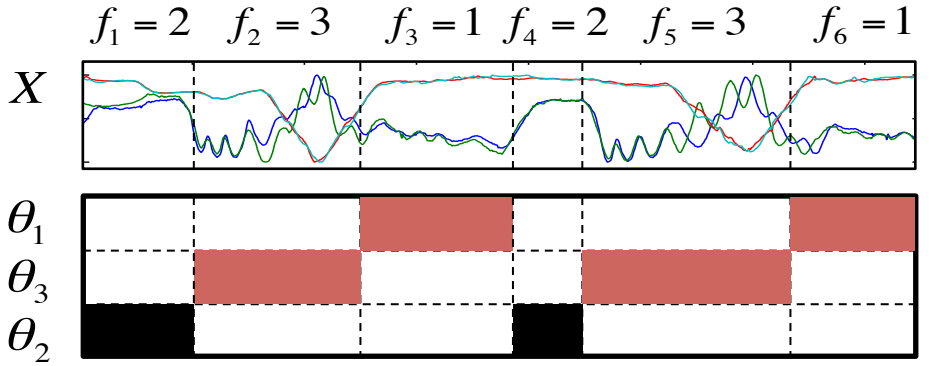
Overview



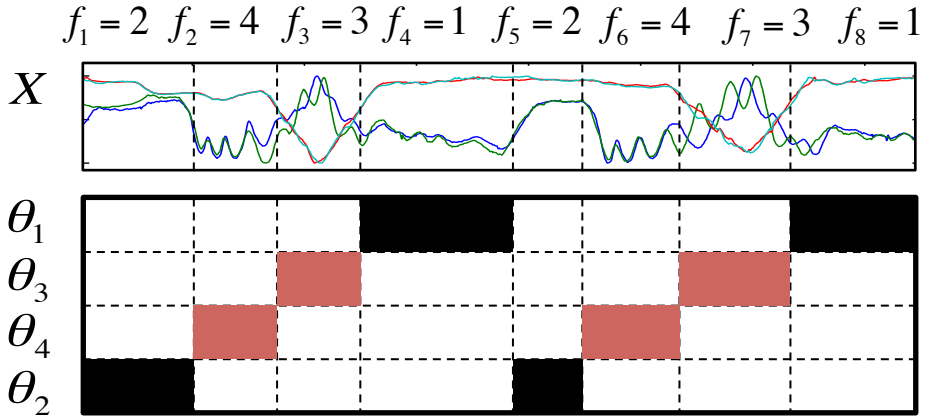
Iteration 1
r=2, m=4



Iteration 2
r=3, m=6



Iteration 4
r=4, m=8



http://www.cs.kumamoto-u.ac.jp/~yasuko/TALKS/15-SIGMOD-tut/

AutoPlait

Algorithms

1. CutPointSearch — Inner-most loop

   Find good cut-points/segments

2. RegimeSplit — Inner loop

   Estimate good regime parameters Θ

3. AutoPlait — Outer loop

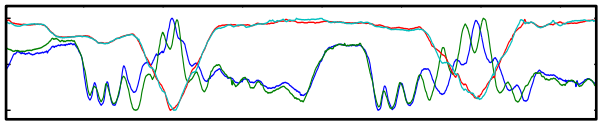
   Search for the best number of regimes (r=2,3,4…)

1. CutPointSearch

Inner-most loop

Given:

- bundle X



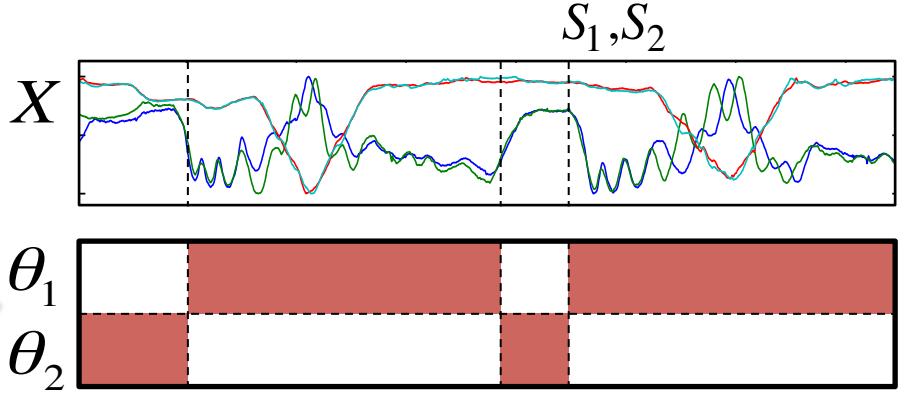
- parameters of two regimes $\Theta = \{\theta_1, \theta_2, \Delta\}$

Find: cut-points of segment sets S_1, S_2,

$$\{S_1, S_2\} = \operatorname{argmax} P(X \mid S_1, S_2, \Theta)$$

X

$\{\theta_1, \theta_2, \Delta\}$

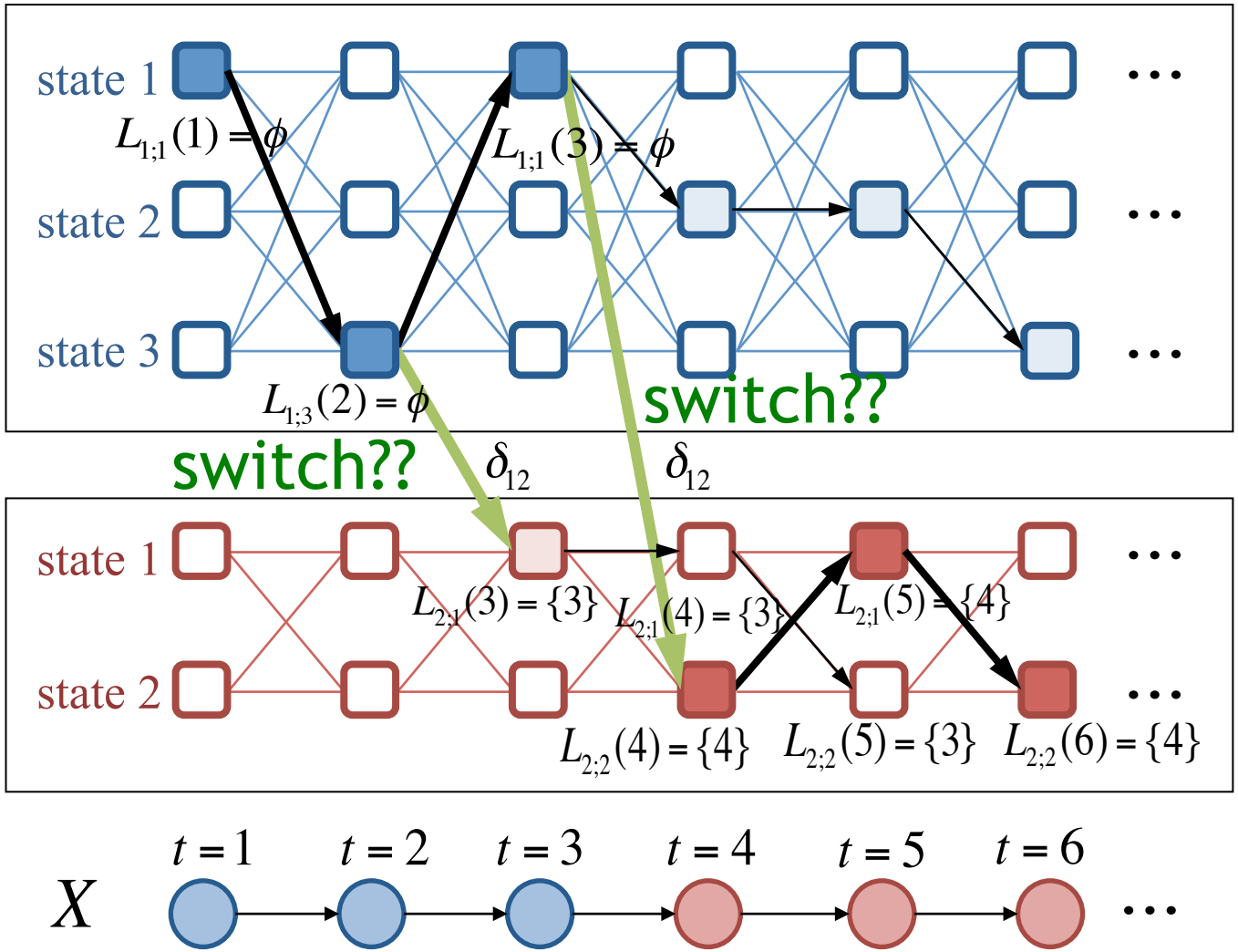


$S_1 = \{s_2, s_4\}$
$S_2 = \{s_1, s_3\}$

1. CutPointSearch

DP algorithm to compute likelihood:

$P(X \mid \Theta)$



1. CutPointSearch

Theoretical analysis

Scalability

- It takes $O(ndk^2)$ time (only single scan)
  - n: length of X
  - d: dimension of X
  - k: # of hidden states in regime

Accuracy

It guarantees the optimal cut points

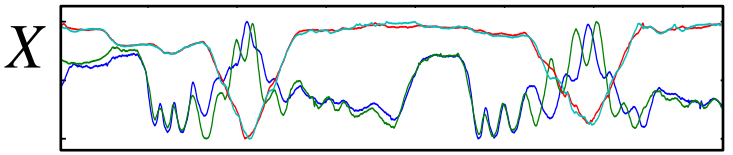
- (Details in paper)

2. RegimeSplit

Inner loop

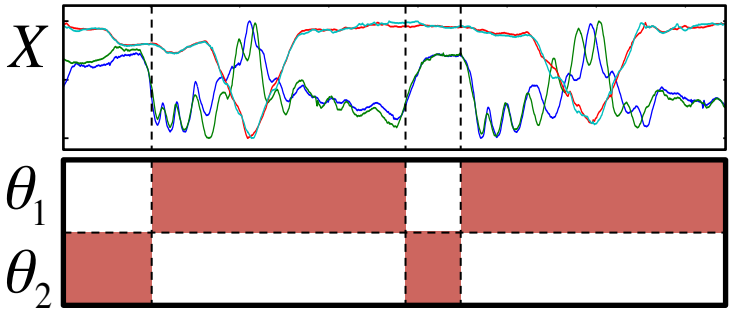
Given:

- bundle X



Find: two regimes

1. find cut-points of segment sets: S_1, S_2
2. estimate parameters of two regimes:

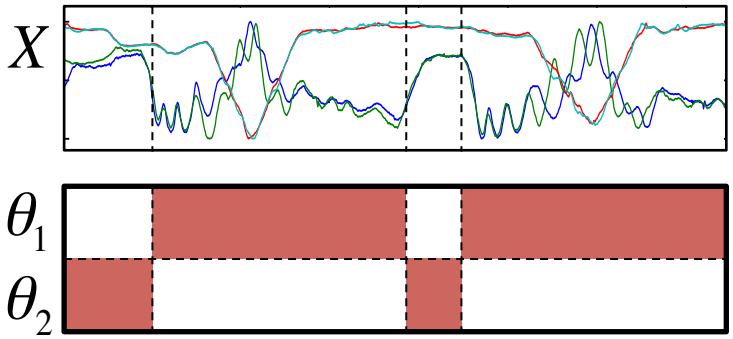


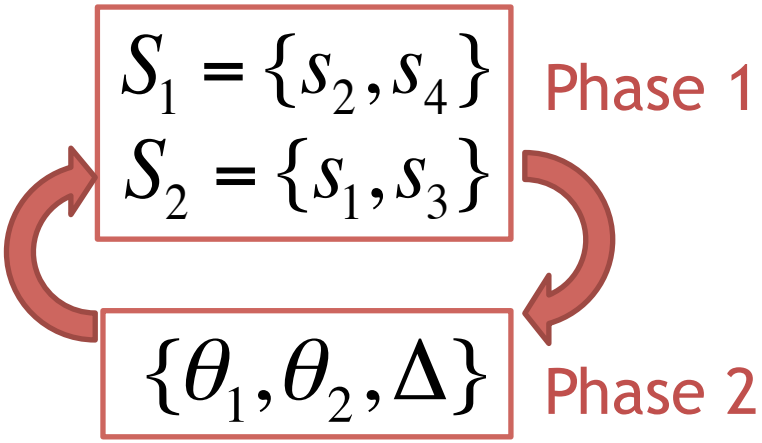
$$\Theta = \{\theta_1, \theta_2, \Delta\}$$

2. RegimeSplit

Two-phase iterative approach

- Phase 1: (CutPointSearch)
  - Split segments into two groups : S_1, S_2
- Phase 2: (BaumWelch)
  - Update model parameters: $\Theta = \{\theta_1, \theta_2, \Delta\}$



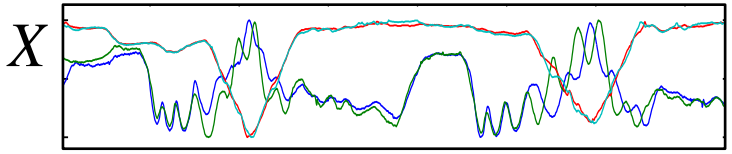


3. AutoPlait

Outer loop

Given:

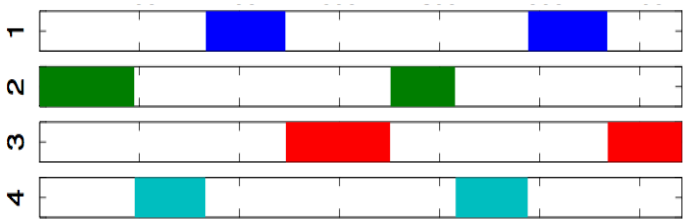
- bundle X



Find: r regimes (r=2, 3, 4, …)

- i.e., find full parameter set

$$C = \{m, r, S, \Theta, F\}$$



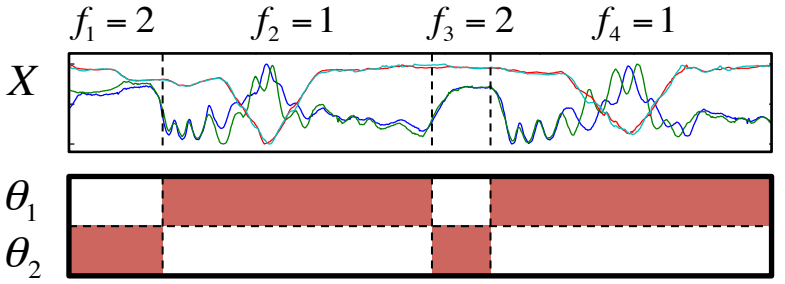
3. AutoPlait

Split regimes r=2,3,…, as long as cost keeps decreasing

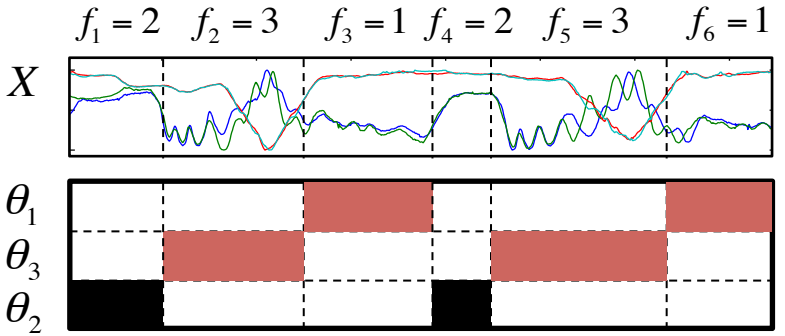
- Find appropriate # of regimes

$$r = \min_{r} Cost_T(X; m, r, S, \Theta, F)$$

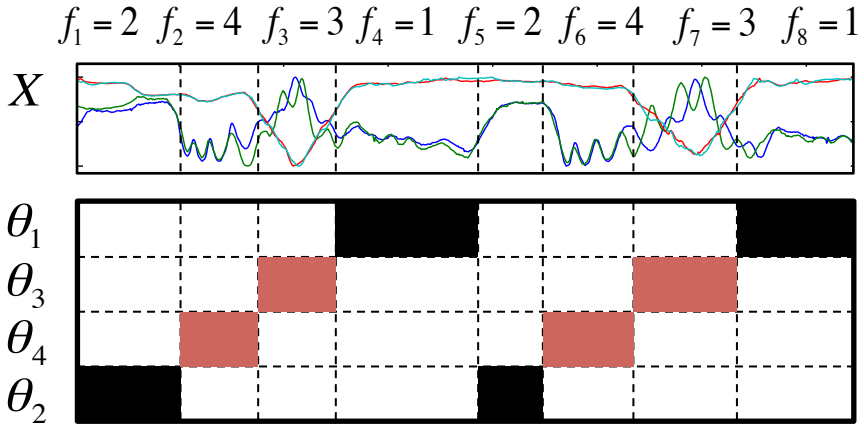
r=2, m=4



r=3, m=6



r=4, m=8



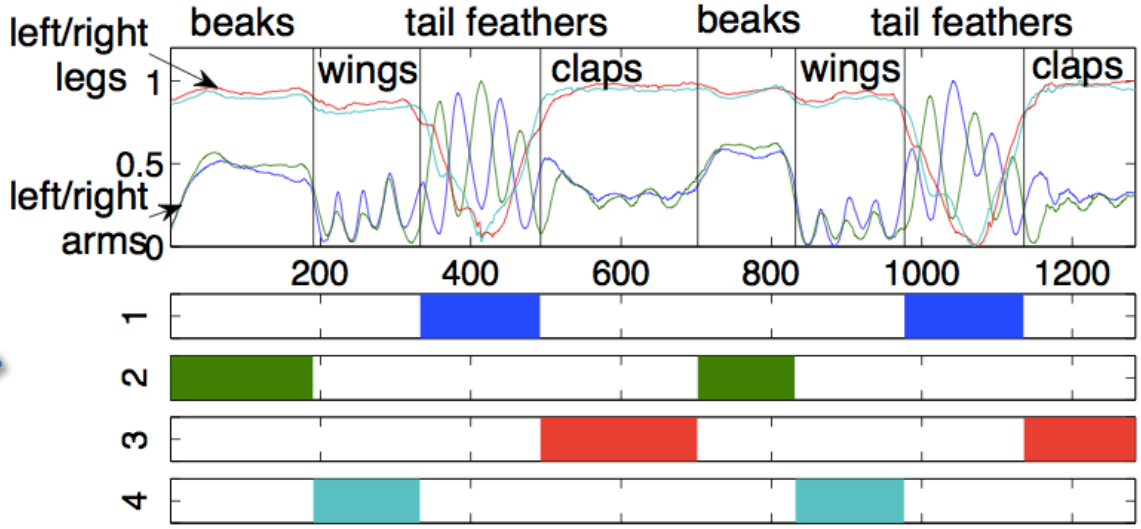
Results

- Mocap data
- WebClick data
- Google Trends

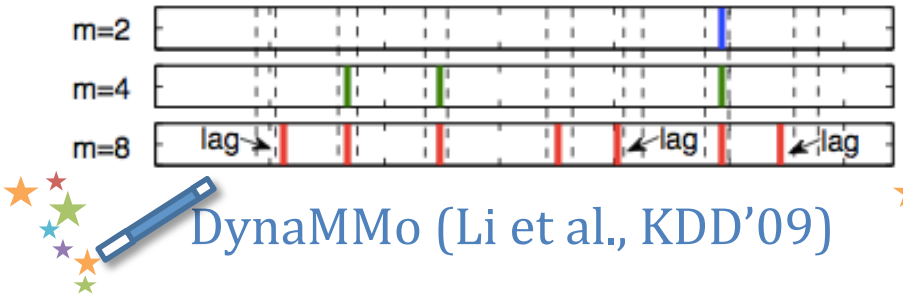
Q1. Sense-making

MoCap data

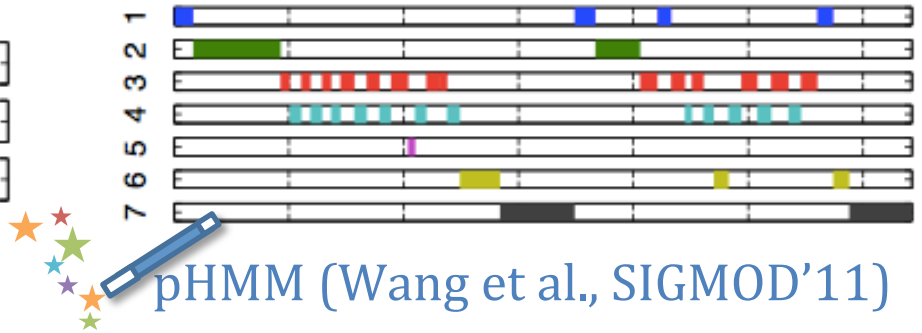
AutoPlait (NO magic numbers)



(a) AUTOPLAIT (**no** user defined parameters)



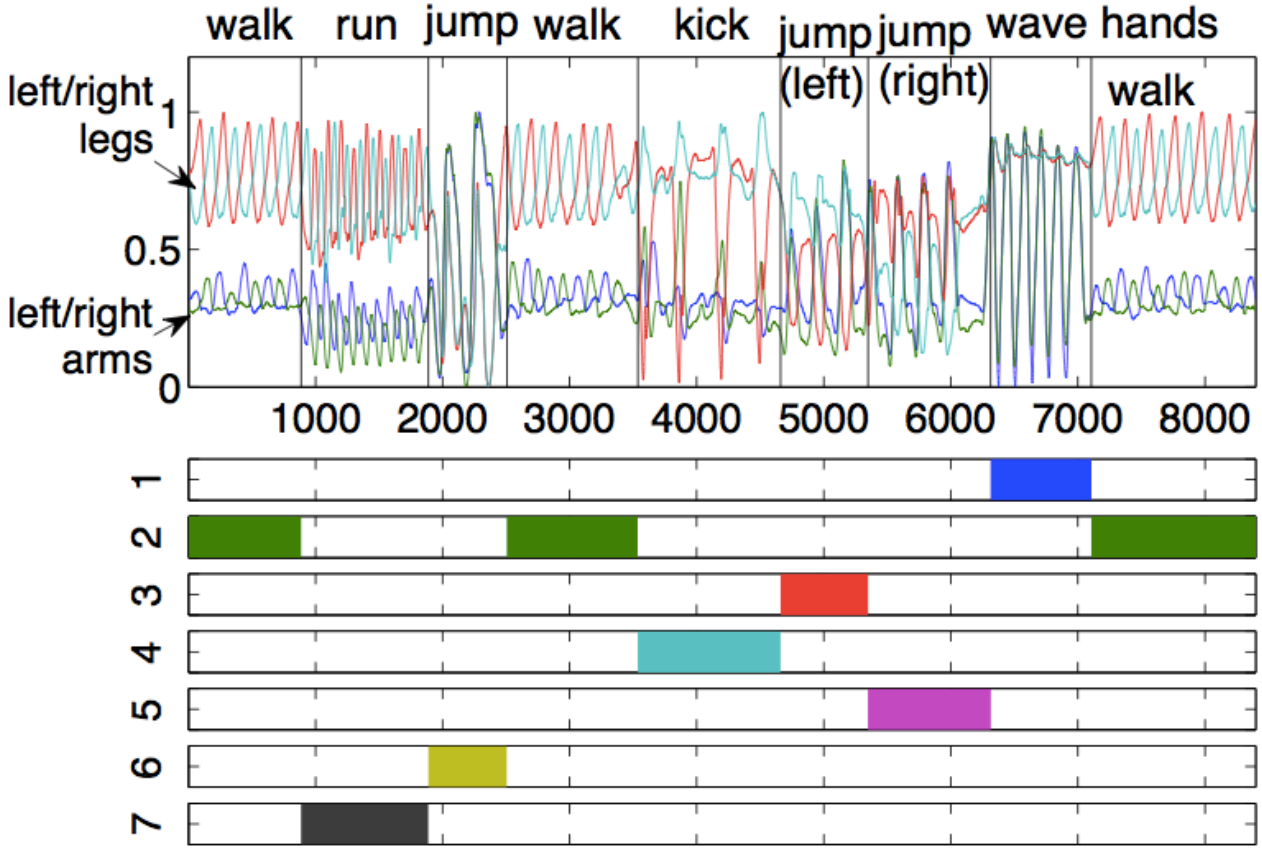
DynaMMo (Li et al., KDD'09)



pHMM (Wang et al., SIGMOD'11)

Q1. Sense-making

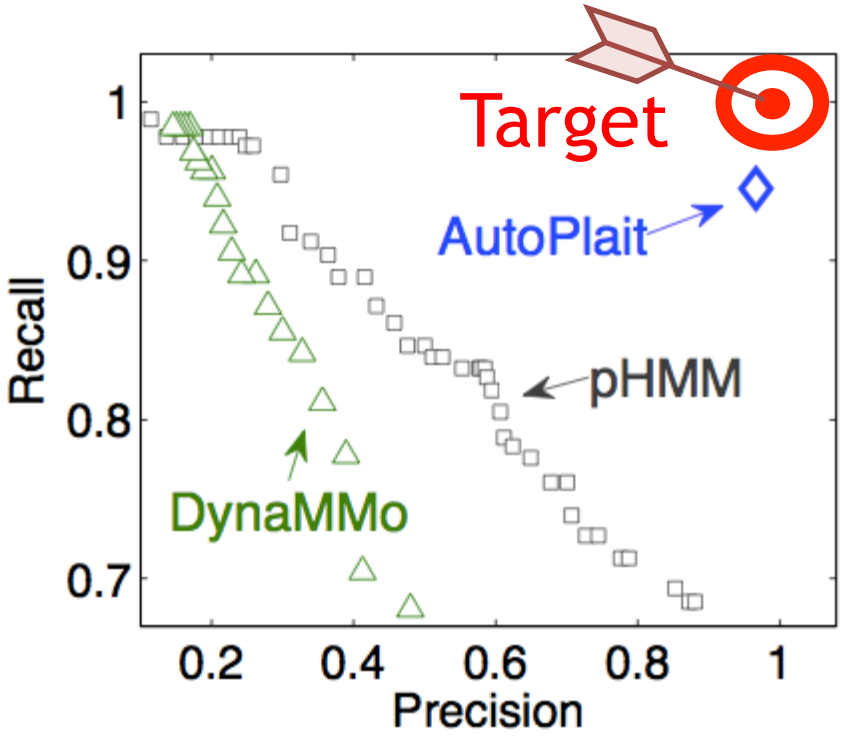
MoCap data



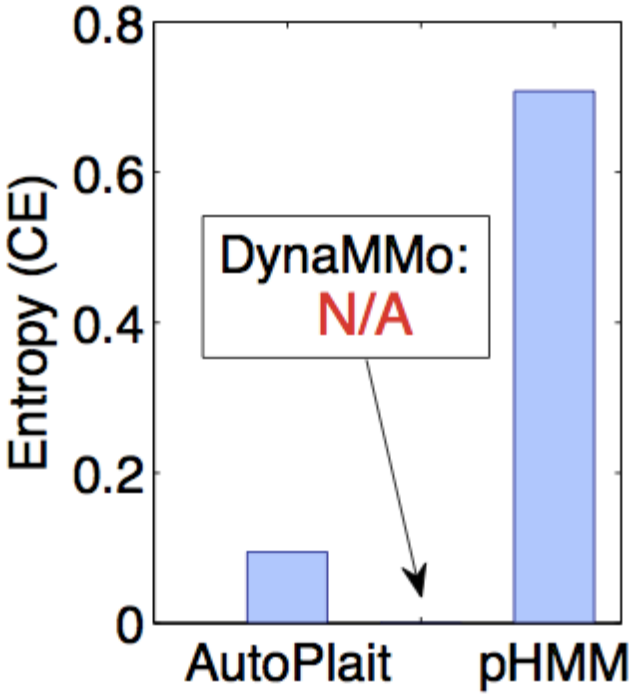
AutoPlait (NO magic numbers)

Q2. Accuracy

(a) Segmentation



(b) Clustering



(a) Precision and recall (higher is better)

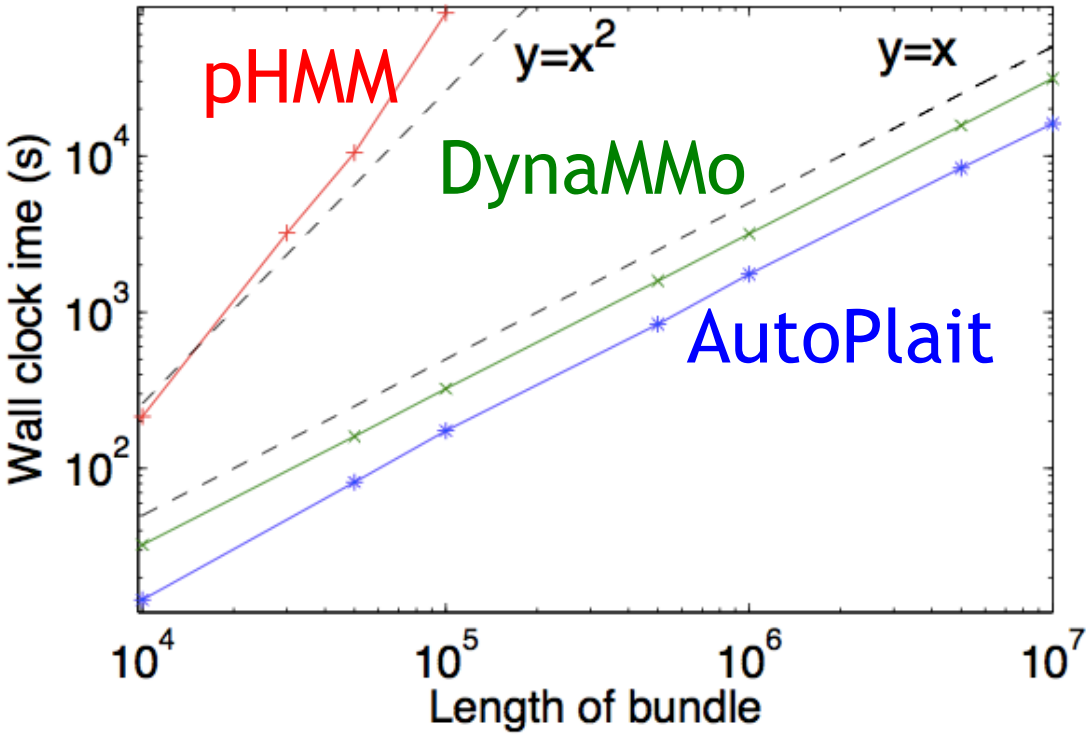
(b) CE score (lower is better)

AutoPlait needs "no magic numbers"

Q3. Scalability

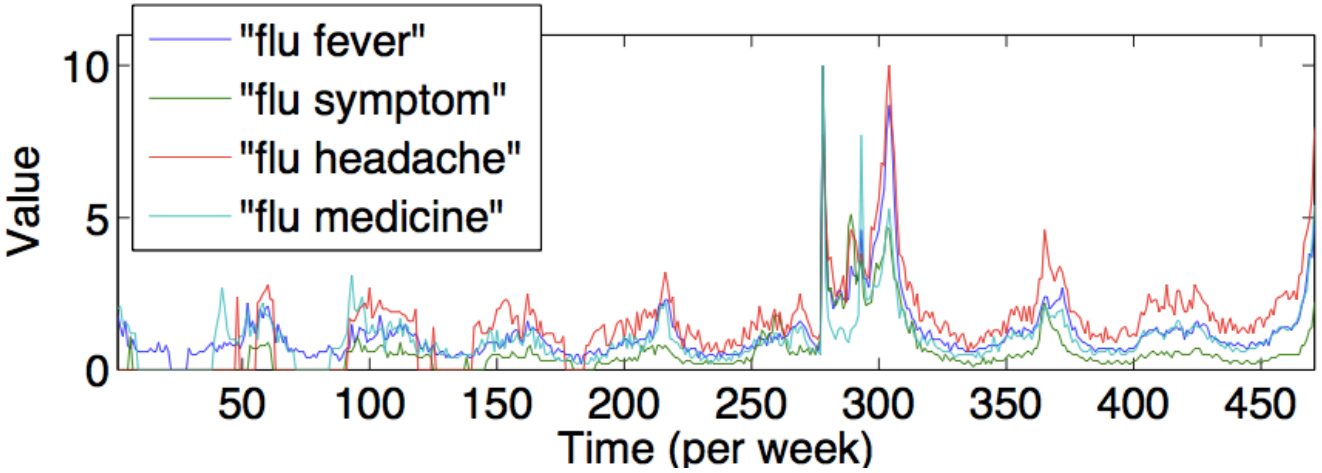
Wall clock time vs. data size (length) : n

AutoPlait scales linearly, i.e., O(n)



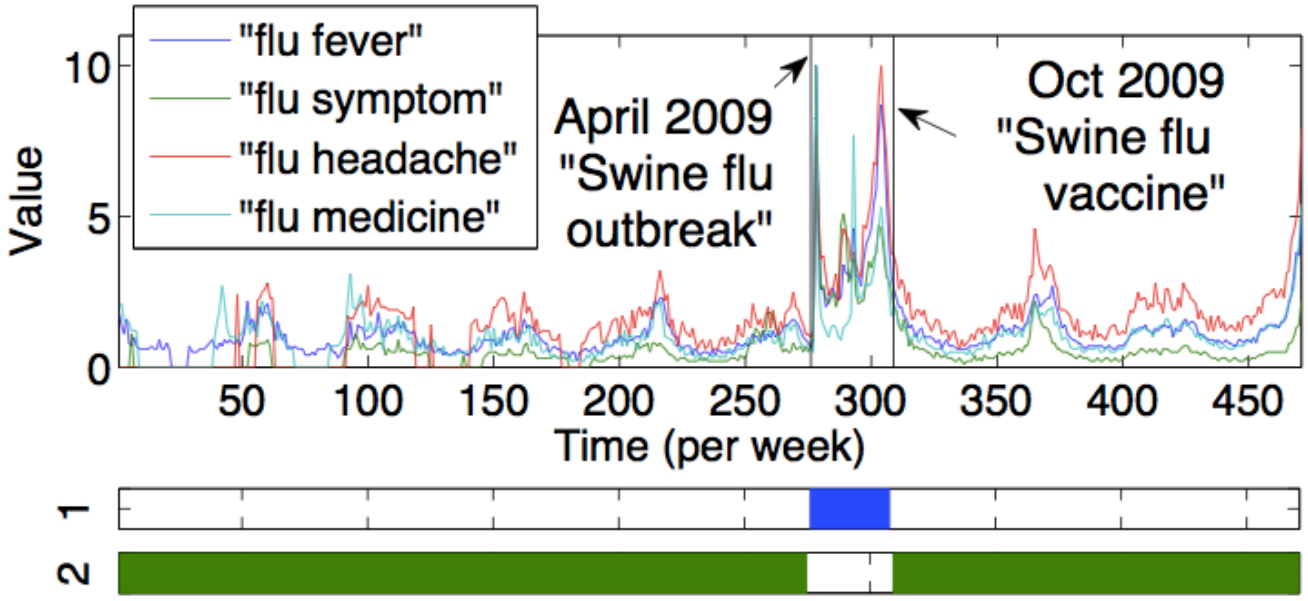
App. Event discovery (GoogleTrend)

Anomaly detection (flu-related topics,10 years)



App. Event discovery (GoogleTrend)

Anomaly detection (flu-related topics,10 years)

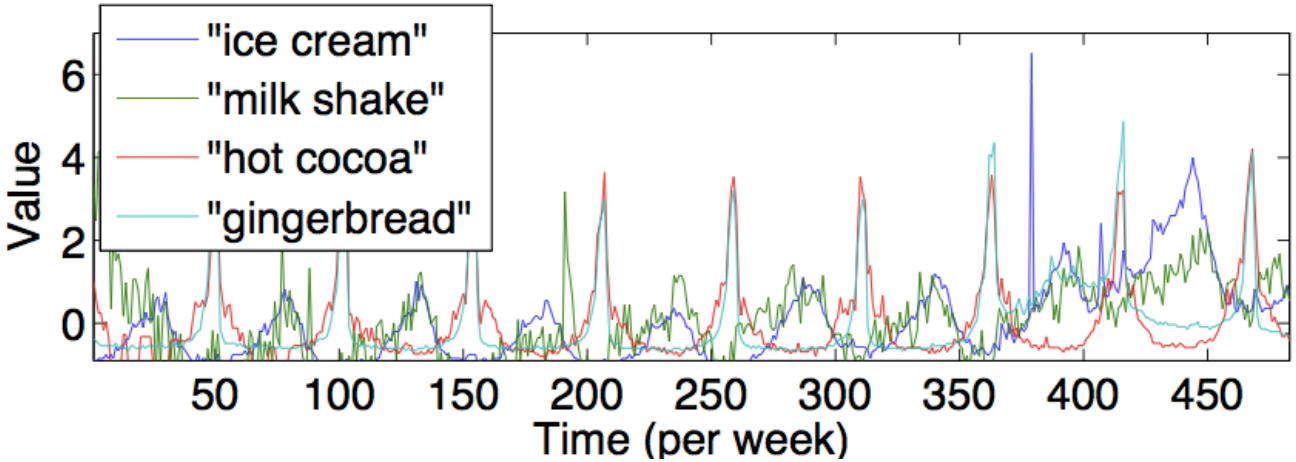


(a) Flu-related topics (regimes $r = 2$)

AutoPlait detects 1 unusual spike in 2009 (i.e., swine flu)

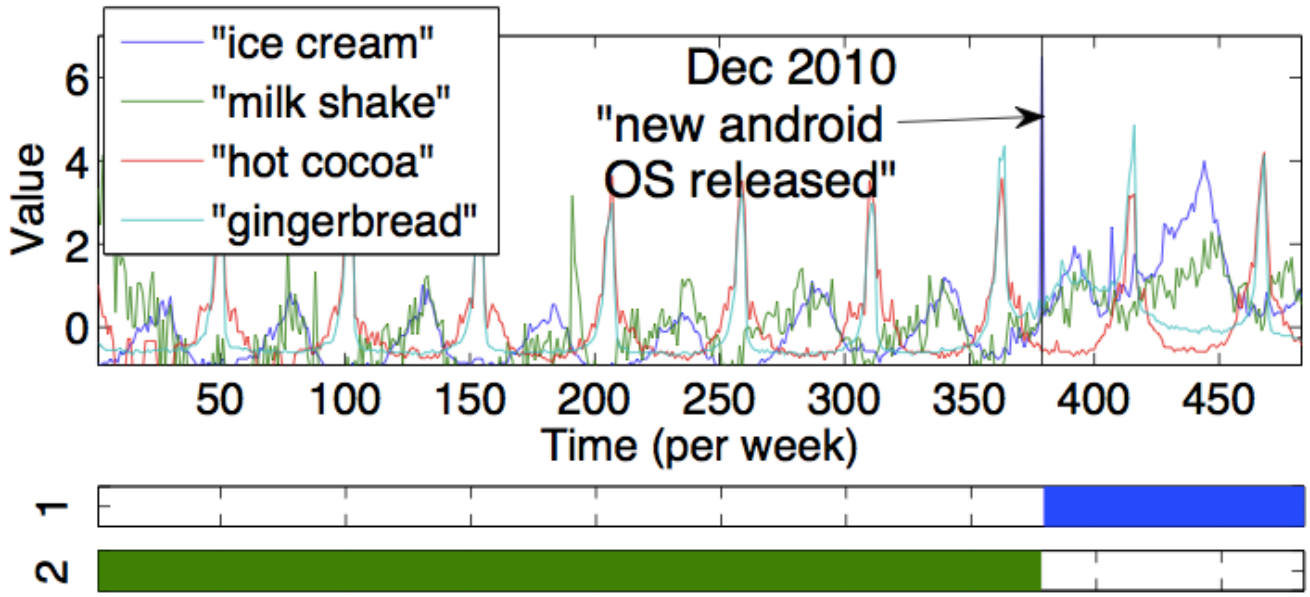
App. Event discovery (GoogleTrend)

Turning point detection (seasonal sweets topics)



App. Event discovery (GoogleTrend)

Turning point detection (seasonal sweets topics)

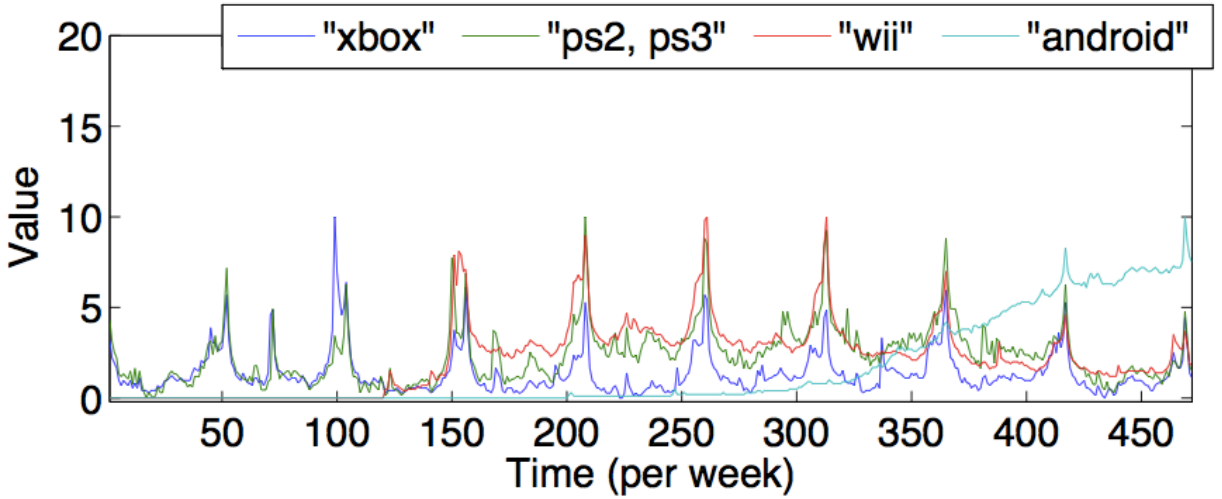


(b) Seasonal sweets topics (regimes $r = 2$)

Trend suddenly changed in 2010 (release of android OS "Ginger bread", "Ice Cream Sandwich")

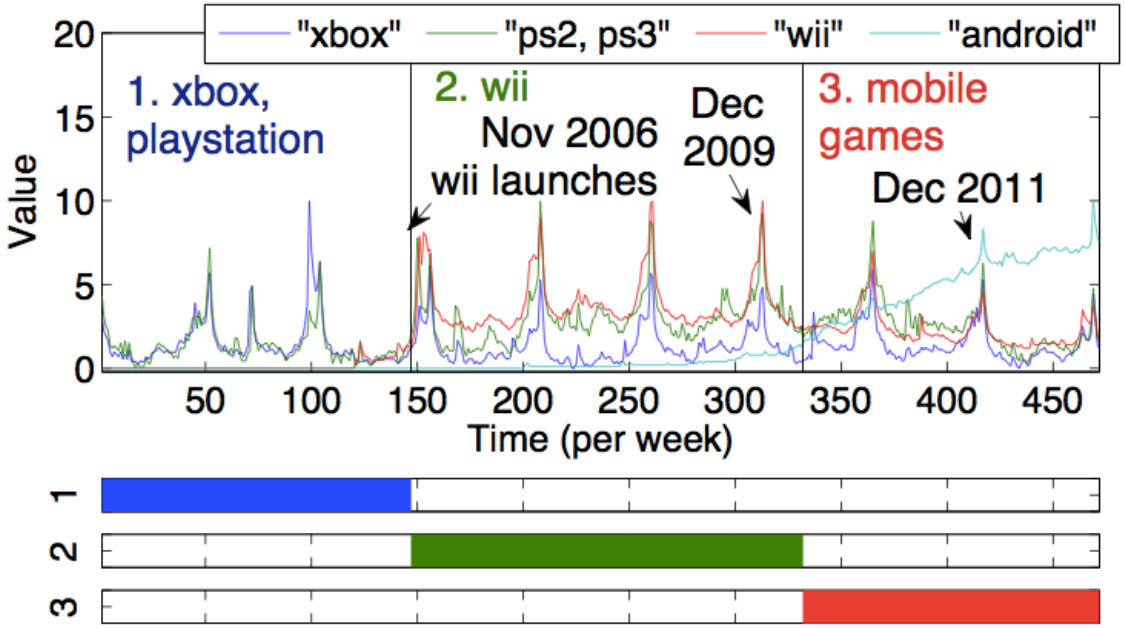
App. Event discovery (GoogleTrend)

Trend discovery (game-related topics)



App. Event discovery (GoogleTrend)

Trend discovery (game-related topics)



(c) Game-related topics (regimes $r = 3$)

It discovers 3 phases of "game console war" (Xbox&PlayStation/Wii/Mobile social games)

Industrial contribution

- Automobile sensor data
  - location, velocity, longitudinal/lateral acceleration

Code at

- http://www.cs.kumamoto-u.ac.jp/~yasuko/software.html

Part 1 – Conclusions

- Motivation
- Similarity Search and Indexing
- Feature extraction
- Linear forecasting
- Streaming pattern discovery
- Automatic mining

Part 1 – Conclusions

- Motivation
- Similarity Search and Indexing
  - Euclidean/time-warping
  - extract features
  - index (SAM, R-tree)
- Feature extraction
  - SVD, ICA, DFT, DWT (multi-scale windows)

Part 1 – Conclusions

- Linear forecasting
  - AR, RLS
- Streaming pattern discovery
  - RLS, "incremental" wavelet transform
  - Multi-scale windows
- Automatic mining
  - MDL

References

- Yunyue Zhu, Dennis Shasha ``*StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time*'' VLDB, August, 2002. pp. 358-369.
- Spiros Papadimitriou, Jimeng Sun, Christos Faloutsos. *Streaming Pattern Discovery in Multiple Time-Series*. VLDB 2005.
- Yasushi Sakurai, Spiros Papadimitriou, Christos Faloutsos. *BRAID: Stream Mining through Group Lag Correlations*. SIGMOD 2005.
- Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, Martin Strauss. *Surfing Wavelets on Streams: One-Pass Summaries for Approximate Aggregate Queries*. VLDB 2001.
- Sudipto Guha, Nick Koudas, Amit Marathe, Divesh Srivastava. *Merging the Results of Approximate Match Operations*. VLDB 2004.
- Anna C. Gilbert, Sudipto Guha, Piotr Indyk, S. Muthukrishnan, Martin Strauss. *Near-optimal sparse fourier representations via sampling*. STOC 2002.

References

- Nick Koudas, Beng Chin Ooi, Kian-Lee Tan, Rui Zhang. *Approximate NN queries on Streams with Guaranteed Error/ performance Bounds*. VLDB 2004.
- Flip Korn, S. Muthukrishnan, Divesh Srivastava. *Reverse Nearest Neighbor Aggregates Over Data Streams*. VLDB 2002.
- Sudipto Guha, Nick Koudas, Kyuseok Shim. *Data-streams and histograms*. STOC 2001.
- Sudipto Guha, Nina Mishra, Rajeev Motwani, Liadan O'Callaghan. *Clustering Data Streams*. FOCS 2000.
- Charu C. Aggarwal, Jiawei Han, Jianyong Wang, Philip S. Yu. *A Framework for Clustering Evolving Data Streams*. VLDB 2003.

References

- Piotr Indyk, Nick Koudas, S. Muthukrishnan. *Identifying Representative Trends in Massive Time Series Data Sets Using Sketches*. VLDB 2000.
- Graham Cormode, S. Muthukrishnan. *An improved data stream summary: the count-min sketch and its applications*. J. Algorithms 55 (1), 2005.
- Graham Cormode, Flip Korn, S. Muthukrishnan, Divesh Srivastava. *Finding Hierarchical Heavy Hitters in Data Streams*. VLDB 2003.
- Piotr Indyk. *Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation*. FOCS 2000.
- Yunyue Zhu, Dennis Shasha. *Efficient elastic burst detection in data streams*. KDD 2003.

References

- Eamonn J. Keogh, Selina Chu, David M. Hart, Michael J. Pazzani. *An Online Algorithm for Segmenting Time Series*. ICDM 2001.
- Spiros Papadimitriou, Philip S. Yu. *Optimal multi-scale patterns in time series streams*. SIGMOD 2006.
- Flip Korn, S. Muthukrishnan, Yihua Wu. *Modeling skew in data streams*. SIGMOD 2006.
- Yasushi Sakurai, Christos Faloutsos, Masashi Yamamuro. *Stream Monitoring under the Time Warping Distance*. ICDE 2007.

Part 1

Similarity search, pattern discovery and summarization

Yasushi Sakurai (Kumamoto University)
Yasuko Matsubara (Kumamoto University)
Christos Faloutsos (Carnegie Mellon University)