BRAID: Discovering Lag Correlations in Multiple Streams

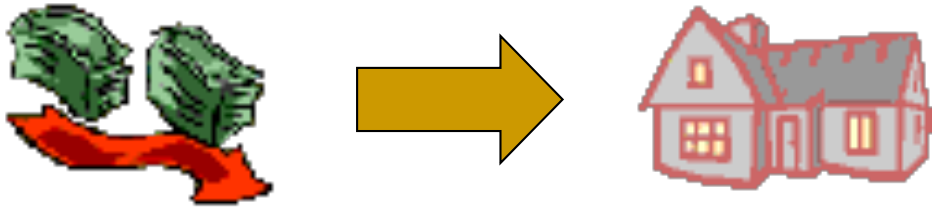
Yasushi Sakurai (NTT Cyber Space Labs)
Spiros Papadimitriou (Carnegie Mellon Univ.)
Christos Faloutsos (Carnegie Mellon Univ.)

Motivation

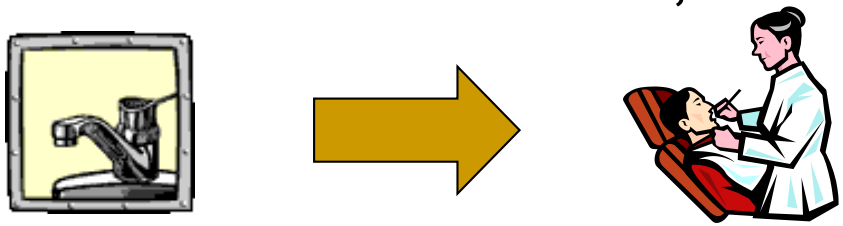
- Data-stream applications
  - Network analysis
  - Sensor monitoring
  - Financial data analysis
  - Moving object tracking
- Goal
  - Monitor multiple numerical streams
  - Determine which pairs are correlated with lags
  - Report the value of each such lag (if any)

Lag Correlations

- Examples
  - A decrease in interest rates typically precedes an increase in house sales by a few months



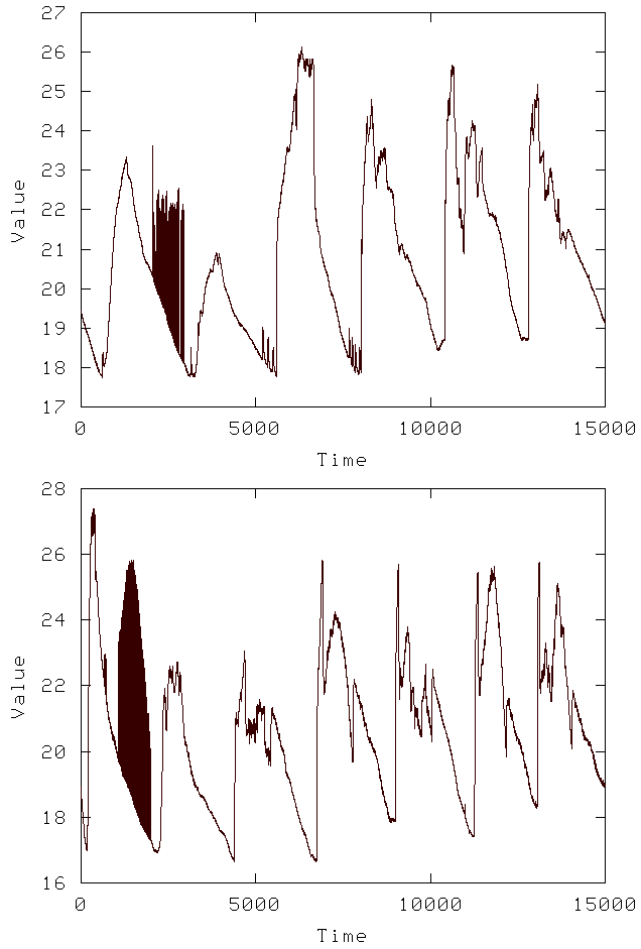
  - Higher amounts of fluoride in the drinking water leads to fewer dental cavities, some years later



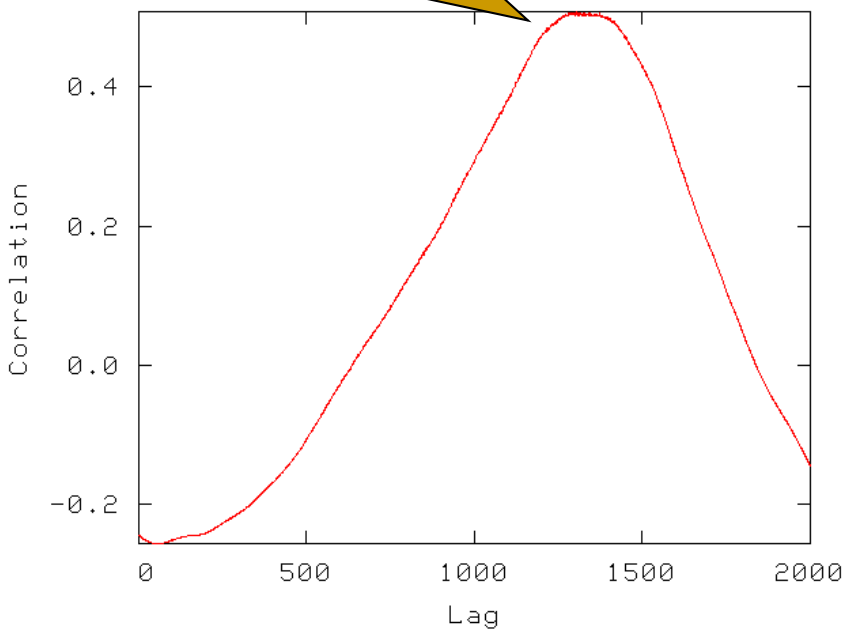
  - High CPU utilization on server 1 precedes high CPU utilization for server 2 by a few minutes

Lag Correlations

- Example of lag-correlated sequences



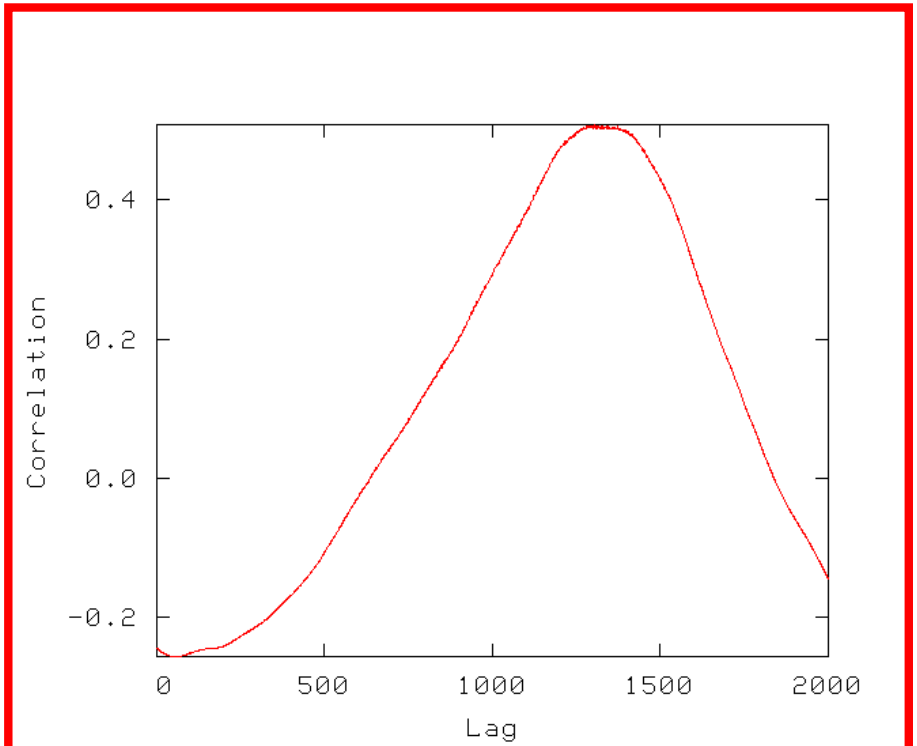
These sequences are correlated with lag l=1300 time-ticks



CCF (Cross-Correlation Function)

Lag Correlations

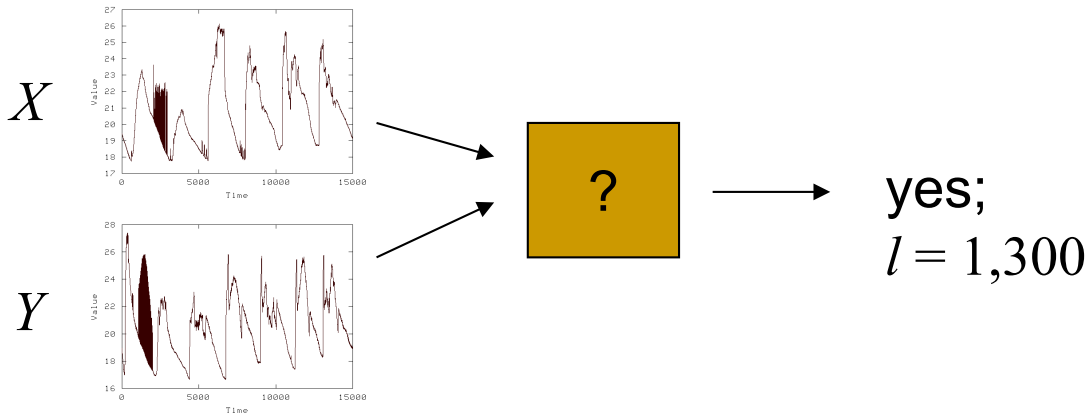
- Example of lag-correlated sequences
  - Fast
    (high performance)
  - Nimble
    (Low memory
    consumption)
  - Accurate
    (good approximation)



CCF (Cross-Correlation Function)

Problem #1: PAIR of sequences

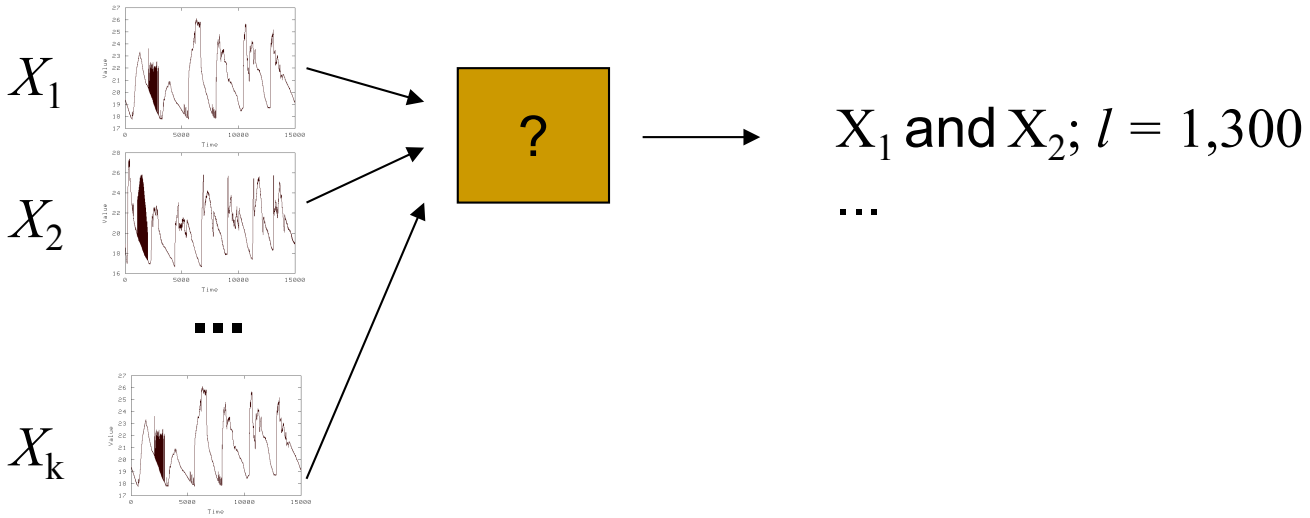
- For given two co-evolving sequences X and Y, determine
  - Whether there is a lag correlation
  - If yes, what is the lag length l



- Any time, on semi-infinite streams

Problem #2: k-way

- For given k numerical sequences, $X_1, \ldots, X_k$, report
  - Which pairs (if any) have a lag correlation
  - The corresponding lag for such pairs



- again, 'any time', streaming fashion

Our solution, BRAID

- characteristics:
  - ‘Any-time’ processing, and fast

    Computation time per time tick is **constant**
  - Nimble

    Memory space requirement is **sub-linear** of sequence length
  - Accurate

    Approximation introduces small error

Related Work

- Sequence indexing
  - Agrawal et al. (FODO 1993)
  - Faloutsos et al. (SIGMOD 1994)
  - Keogh et al. (SIGMOD 2001)
- Compression (wavelet and random projections)
  - Gilbert et al. (VLDB 2001)
  - Guha et al. (VLDB 2004)
  - Dobra et al.(SIGMOD 2002)
  - Ganguly et al.(SIGMOD 2003)

Related Work

- Data Stream Management
  - Abadi et al. (VLDB Journal 2003)
  - Motwani et al. (CIDR 2003)
  - Chandrasekaran et al. (CIDR 2003)
  - Cranor et al. (SIGMOD 2003)

Related Work

- Pattern discovery
  - Clustering for data streams
    Guha et al. (TKDE 2003)
  - Monitoring multiple streams
    Zhu et al. (VLDB 2002)
  - Forecasting
    Yi et al. (ICDE 2000)
    Papadimitriou et al. (VLDB 2003)

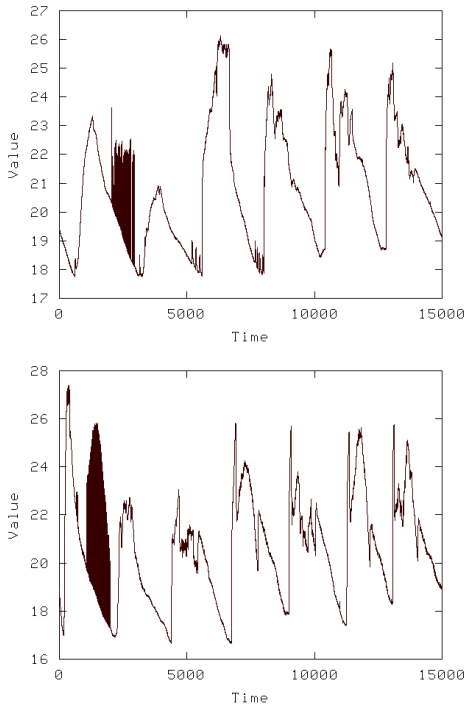
- None of previously published methods focuses on the problem

Overview

- Introduction / Related work
- Background
- Main ideas
- Theoretical analysis
- Experimental results

Background

- Lag correlation

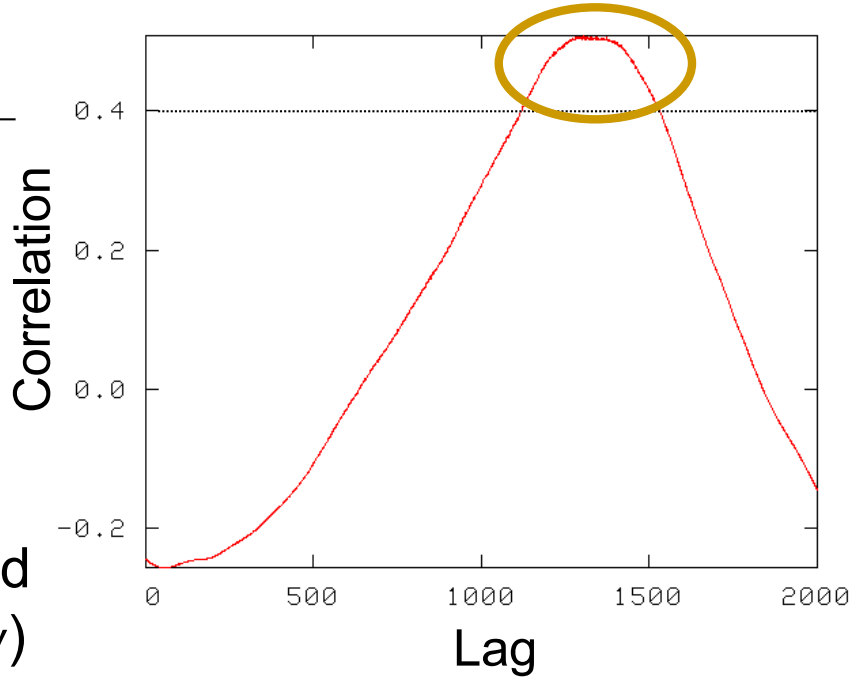


positively correlated

+γ

un-correlated

anti-correlated
(lower than -γ)



CCF (Cross-Correlation Function)

Background

details

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

$$score(l) = |R(l)|$$

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

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

Overview

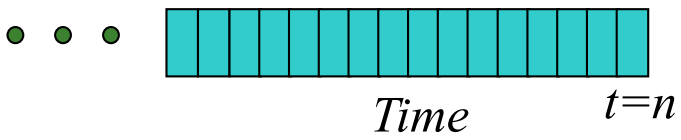
- Introduction / Related work
- Background
- Main ideas
- Theoretical analysis
- Experimental results

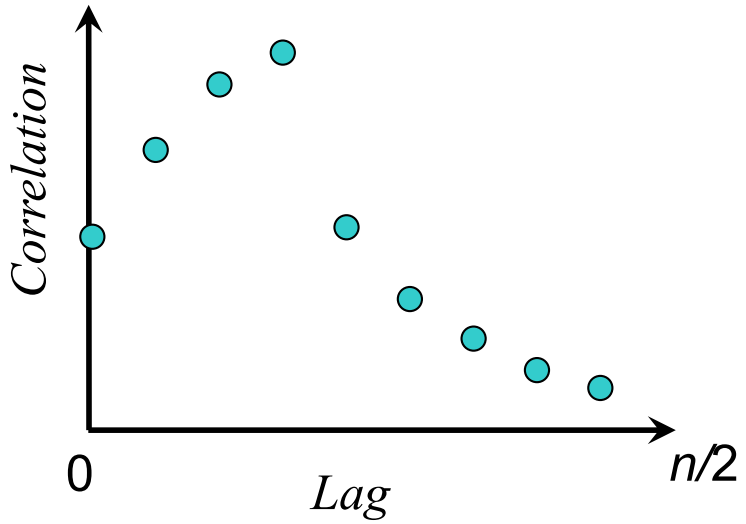
Why not 'naive'?

- Naive solution:
  - 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





Main Idea (1)

- Incremental computing:
  - the correlation coefficient of two sequences is 'algebraic' -> can be computed incrementally

- we need to maintain only 6 'sufficient statistics':
  - Sequence length n
  - Sum of X, Square sum of X
  - Sum of Y, Square sum of Y
  - Inner-product for X and the shifted Y

Main Idea (1)

- Incremental computing:
  - Sequence length n
  - Sum of X : $Sx(1,n) = \sum_{t=1}^{n} x_t$
  - Square sum of X : $Sxx(1,n) = \sum_{t=1}^{n} x_t^2$
  - Inner-product for X and the shifted Y : $Sxy(l) = \sum_{t=l+1}^{n} x_t y_{t-l}$
  - Compute $R(l)$ incrementally:

$$R(l) = \frac{C(l)}{\sqrt{Vx(l+1,n) \cdot Vy(1,n-l)}}$$

  - Covariance of X and Y:

$$C(l) = Sxy(l) - \frac{Sx(l+1,n) \cdot Sy(1,n-l)}{n-l}$$

  - Variance of X:

$$Vx(l+1,n) = Sxx(l+1,n) - \frac{(Sx(l+1,n))^2}{n-l}$$

Main Idea (1)

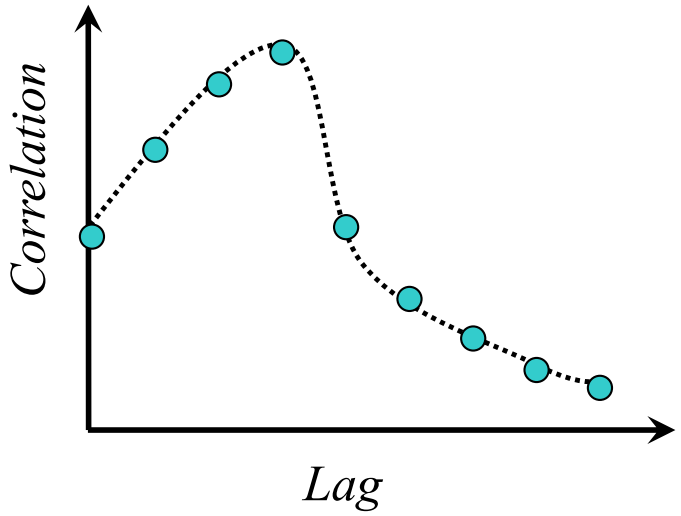
- Complexity

	Naive	Naive (incremental)	BRAID
Space	$O(n)$	$O(n)$	
Comp. time	$O(n \log n)$	$O(n)$	

Better, but not good enough!

Main Idea (2)

- Geometric lag probing

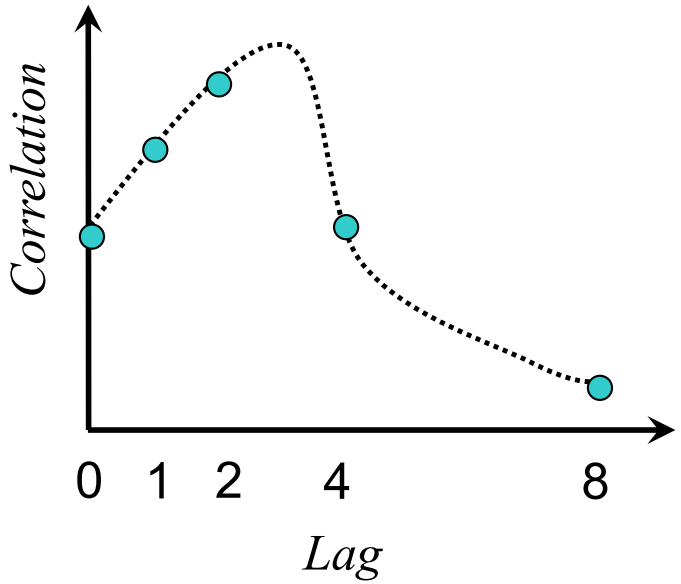


Main Idea (2)

- Geometric lag probing
- ie., compute the correlation coefficient for lag:

$$l = 0, 1, 2, 4, \ldots 2^h$$

$O(\log n)$ estimations



Main Idea (2)

- Geometric lag probing

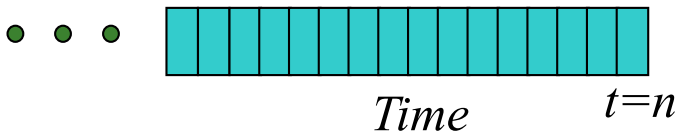
	Naive	Naive (incremental)	BRAID
Space	$O(n)$	$O(n)$	
Comp. time	$O(n \log n)$	$O(n)$	$O(\log n)$

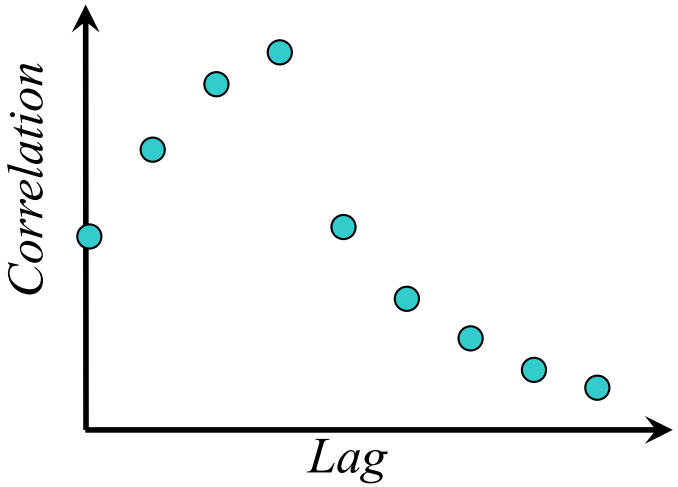
- But, so far, we still need $O(n)$ space because the longest lag is n/2

Main Idea (3)

- Sequence smoothing

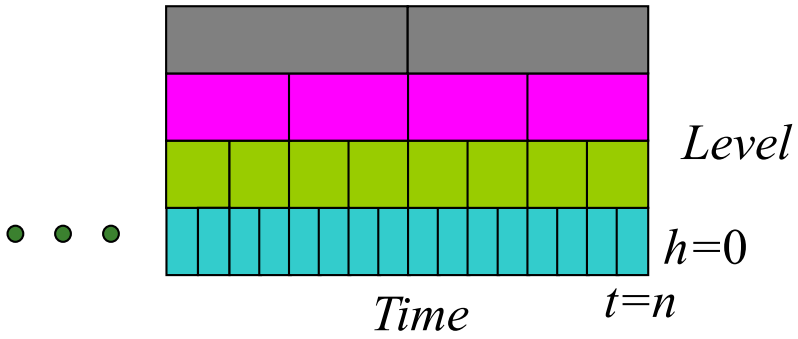
Reminder: Naïve:

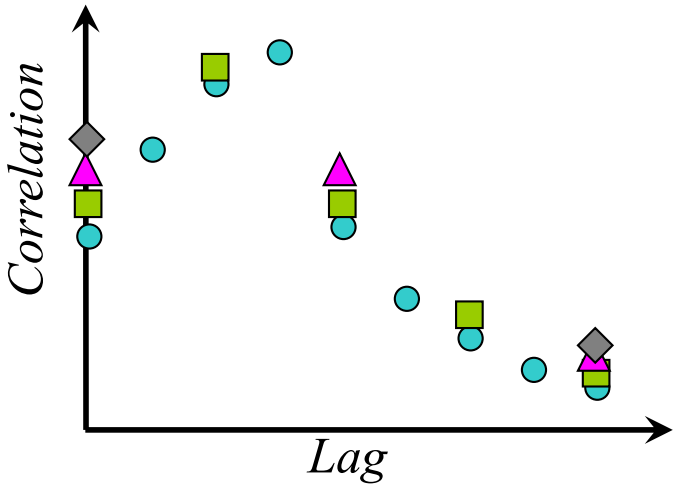




Main Idea (3)

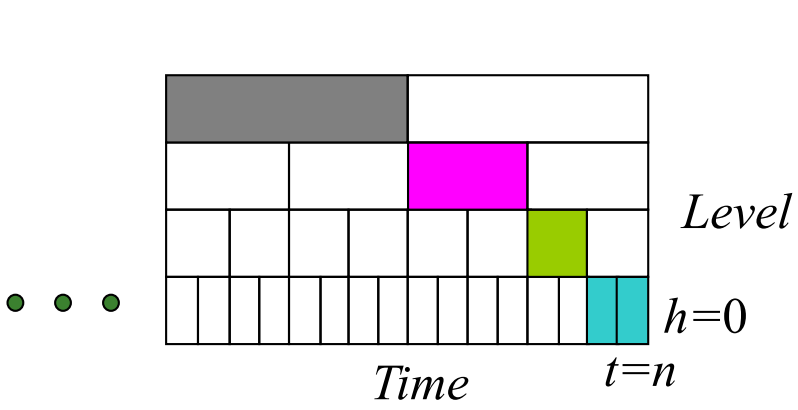
- Sequence smoothing
  - **Means of windows** for each level
  - Sufficient statistics computed from the means
  - CCF computed from the sufficient statistics
  - But, it allows a partial redundancy

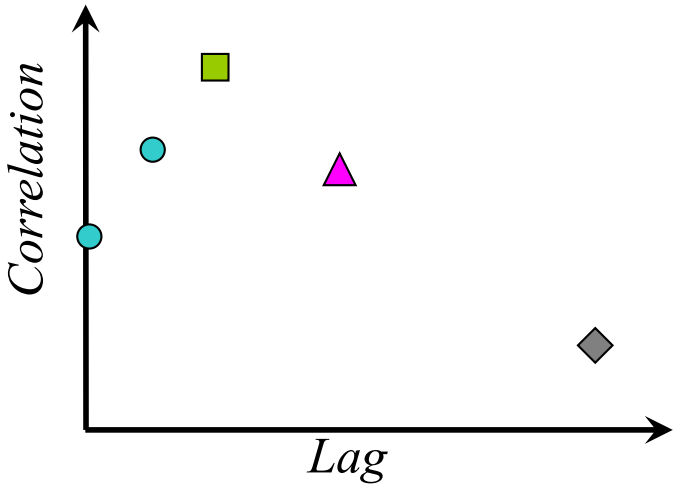




Putting it all together:

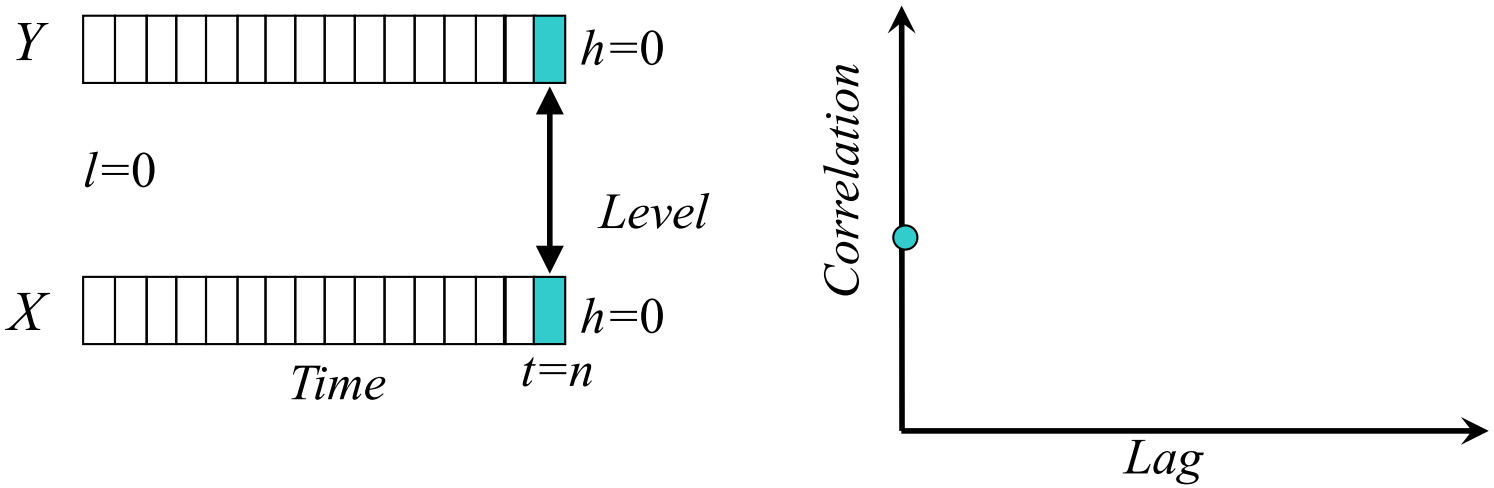
- Geometric lag probing + smoothing
  - Use colored windows
  - Keep track of only a geometric progression of the lag values: $l=\{0,1,2,4,8,\ldots,2^{h},\ldots\}$





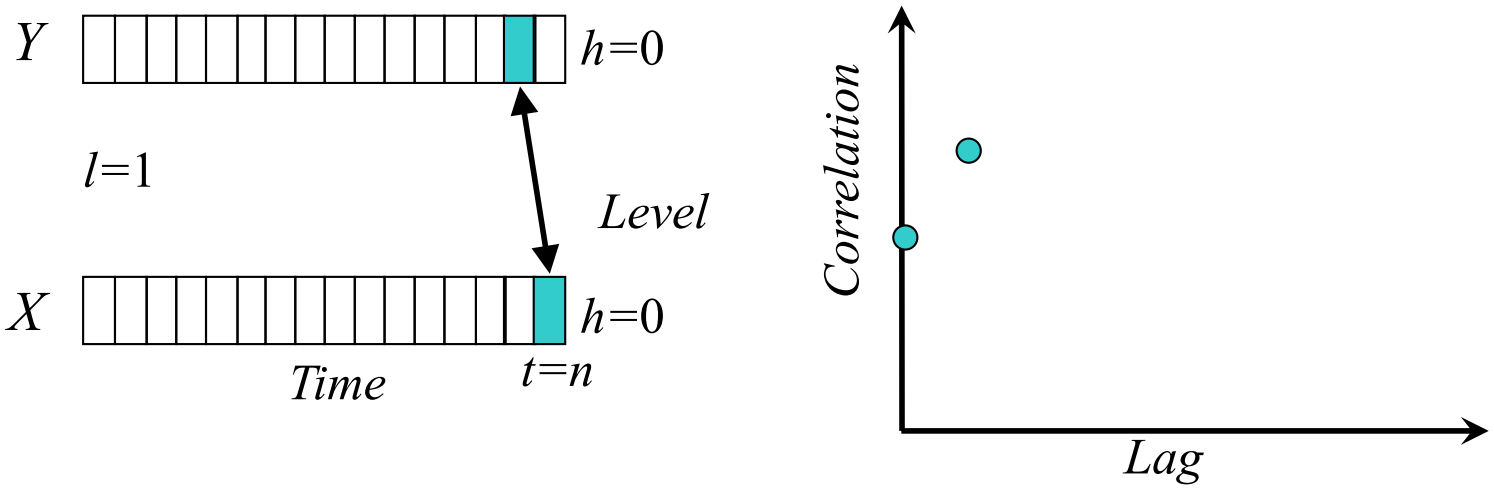
Putting it all together:

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



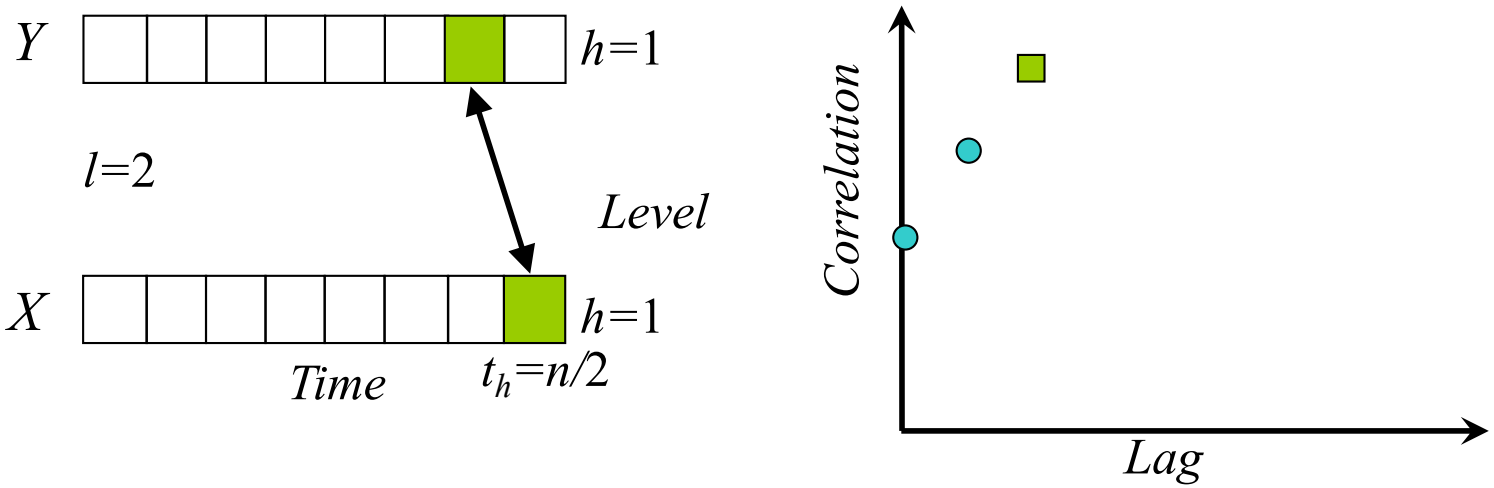
Putting it all together:

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



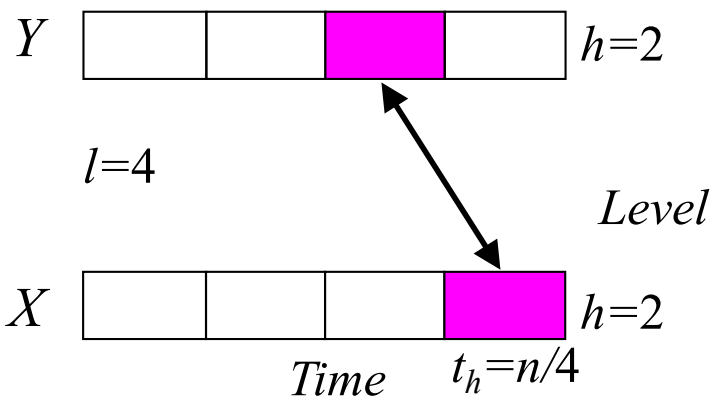
Putting it all together:

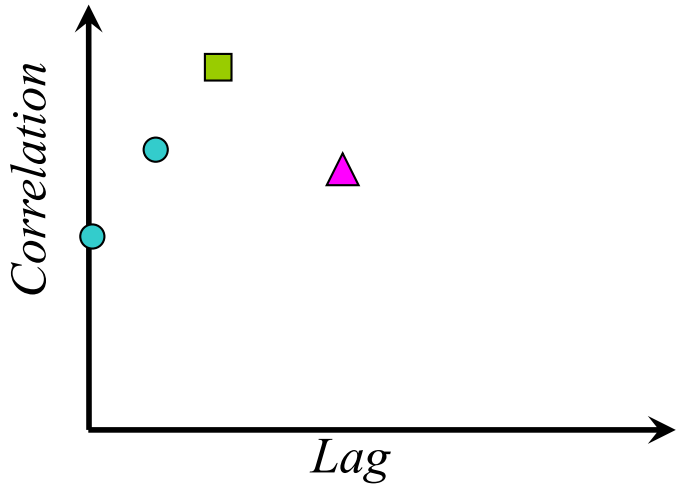
- Geometric lag probing + smoothing
  - Use colored windows
  - Keep track of only a geometric progression of the lag values: $l=\{0,1,2,4,8,\ldots,2^{h},\ldots\}$



Putting it all together:

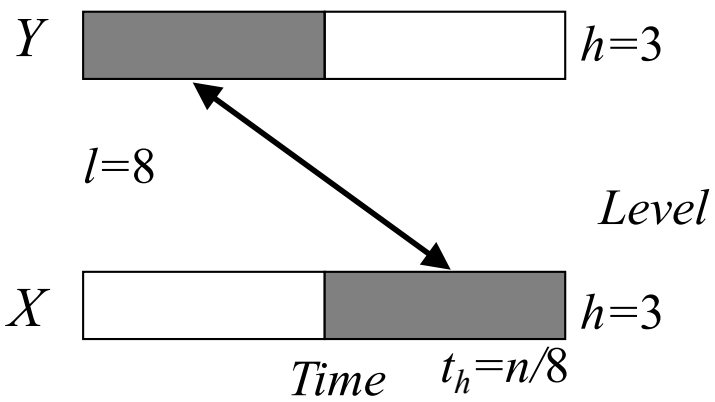
- Geometric lag probing + smoothing
  - Use colored windows
  - Keep track of only a geometric progression of the lag values: $l=\{0,1,2,4,8,\ldots,2^{h},\ldots\}$

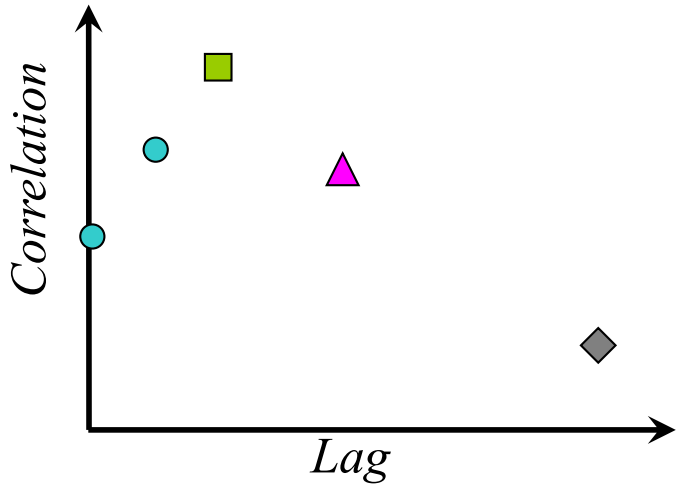




Putting it all together:

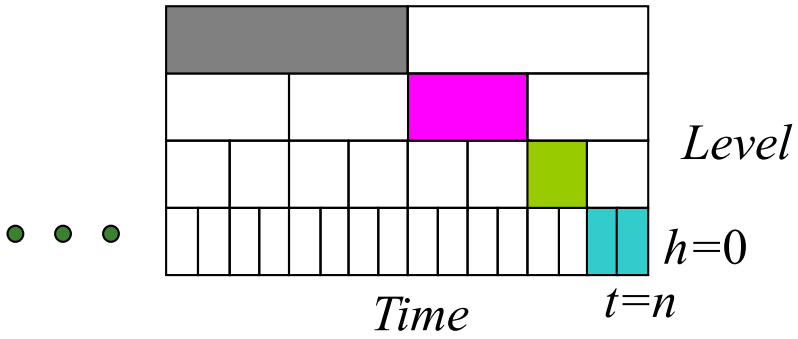
- Geometric lag probing + smoothing
  - Use colored windows
  - Keep track of only a geometric progression of the lag values: $l=\{0,1,2,4,8,\ldots,2^{h},\ldots\}$

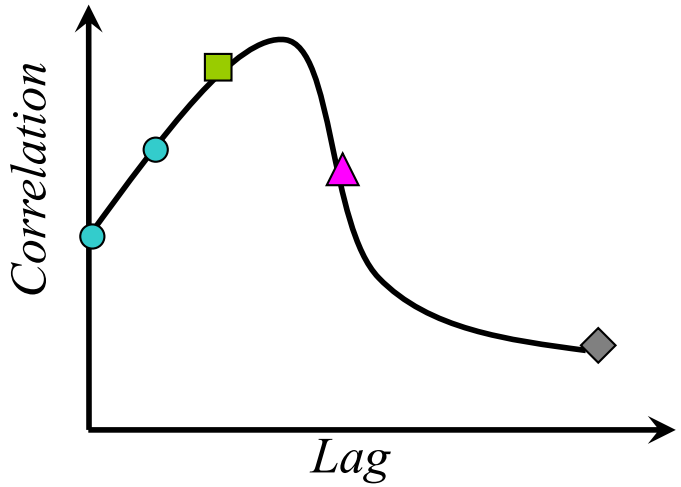




Putting it all together:

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





Thus:

- Complexity

	Naive	Naive (incremental)	BRAID
Space	$O(n)$	$O(n)$	$O(log\ n)$
Comp. time	$O(n\ log\ n)$	$O(n)$	$O(1)$ *

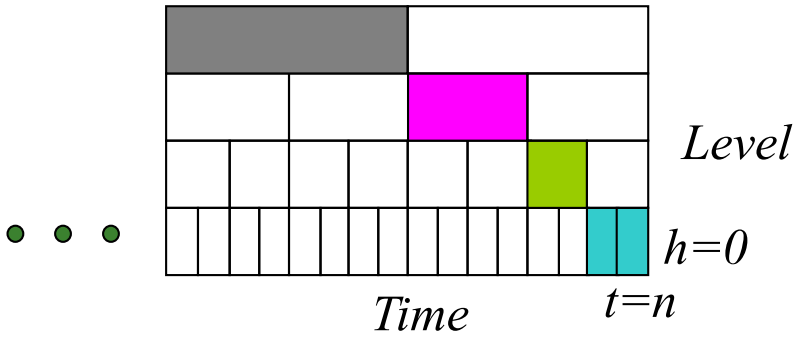
(*) Computation time: O(logn)
And actually, amortized time: O(1)

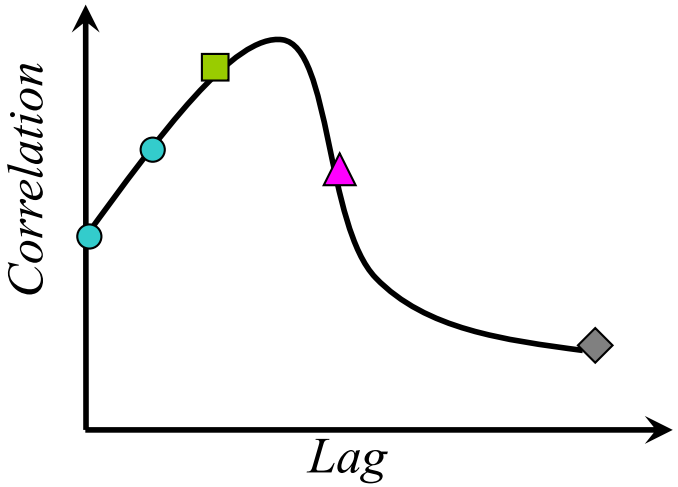
Overview

- Introduction / Related work
- Background
- Main ideas
  - enhancing the accuracy
- Theoretical analysis
- Experimental results

Enhanced Probing Scheme

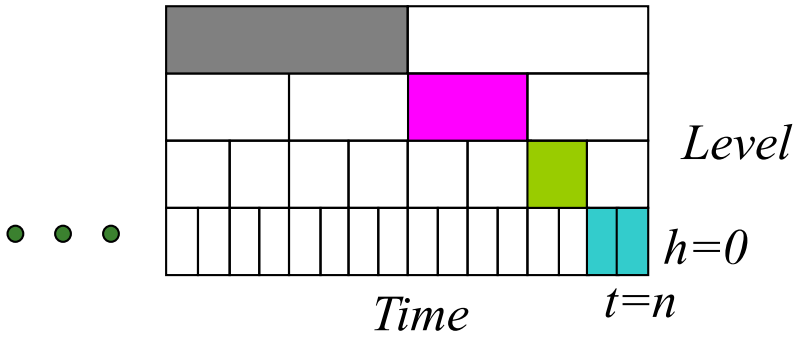
- Q: How to probe more densely than 2^h ?

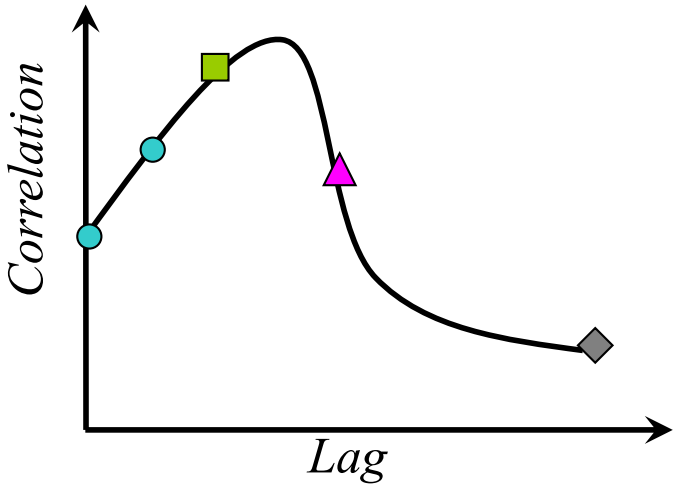




Enhanced Probing Scheme

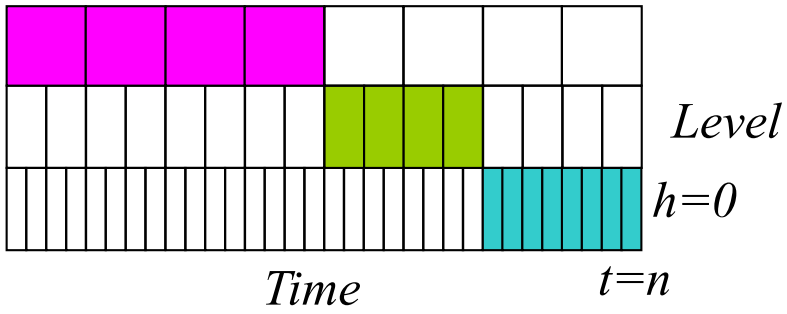
- Q: How to probe more densely than 2^h ?
- A: probe in a mixture of geometric and arithmetic progressions

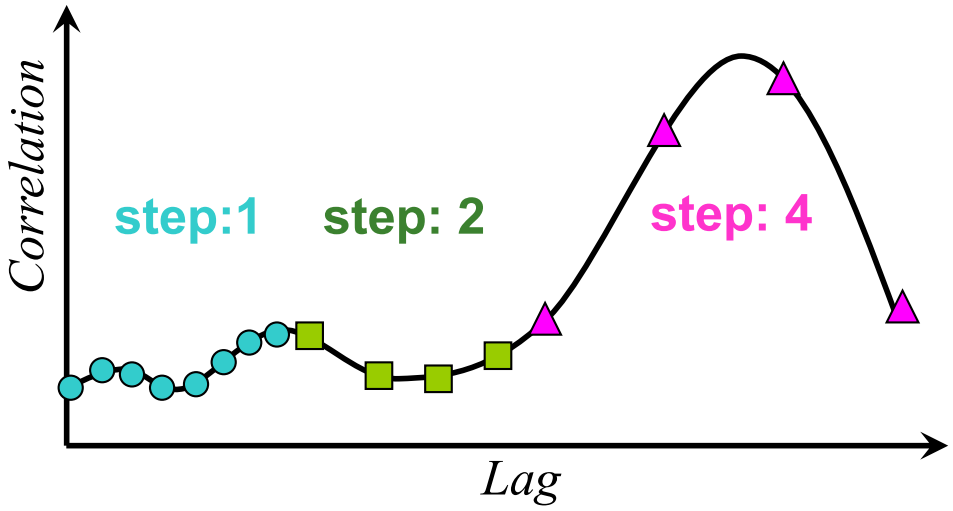




Enhanced Probing Scheme

- Basic scheme: $b=1$ (one number for each level)
- Enhanced scheme: $b>1$
  - Example of $b=4$
  - Probing the CCF in a mixture of geometric and arithmetic progressions: $l=\{0,1,\ldots,7;8,10,12,14;16,20,24,28;32,40,\ldots\}$





Overview

- Introduction / Related work
- Background
- Main ideas
- Theoretical analysis
- Experimental results

Theoretical Analysis - Accuracy

- Effect of smoothing

  For sequences with low frequencies, smoothing introduces only small error

- Effect of geometric lag probing

  BRAIDS will provide no error, if lag probing satisfies the sampling theorem (Nyquist's)

Theoretical Analysis - Accuracy

details

- Effect of geometric lag probing
  - Informally, BRAIDS will provide no error, if lag probing satisfies the sampling theorem (Nyquist's)
  - Formally: Theorem 2

BRAID will find the lag correlations perfectly, if

$$0 \le l \le \frac{2b}{f_R}$$

f_R: the Nyquist frequency of CCF, f_R=min(f_x, f_y)

f_x, f_y: the Nyquist frequencies of X and Y

Theoretical Analysis - Complexi

details

Naive solution

- $O(n)$ space
- $O(n)$ time per time tick

BRAID

- $O(log\ n)$ space
- $O(1)$ time for updating sufficient statistics
- $O(log\ n)$ time for interpolating (when output is required)

Overview

- Introduction / Related work
- Background
- Main ideas
- Theoretical analysis
- Experimental results

Experimental results

- Setup
  - Intel Xeon 2.8GHz, 1GB memory, Linux
  - Datasets:

    Synthetic: *Sines*, *SpikeTrains*,

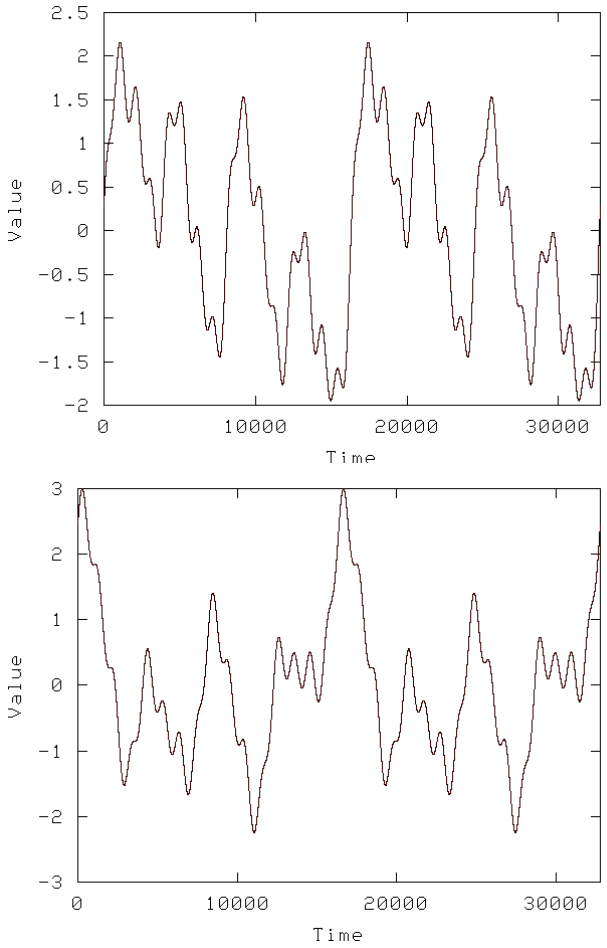
    Real: *Humidity*, *Light*, *Temperature*, *Kursk*, *Sunspots*
  - Enhanced BRAID, b=16

Experimental results

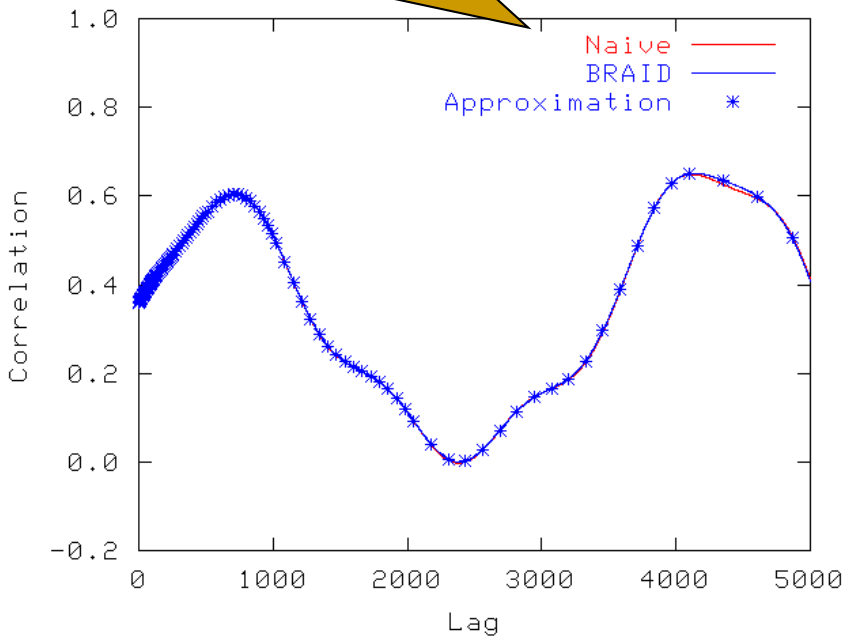
- Evaluation
  - Accuracy for CCF
  - Accuracy for the lag estimation
  - Computation time
  - k-way lag correlations

Accuracy for CCF (1)

- *Sines*



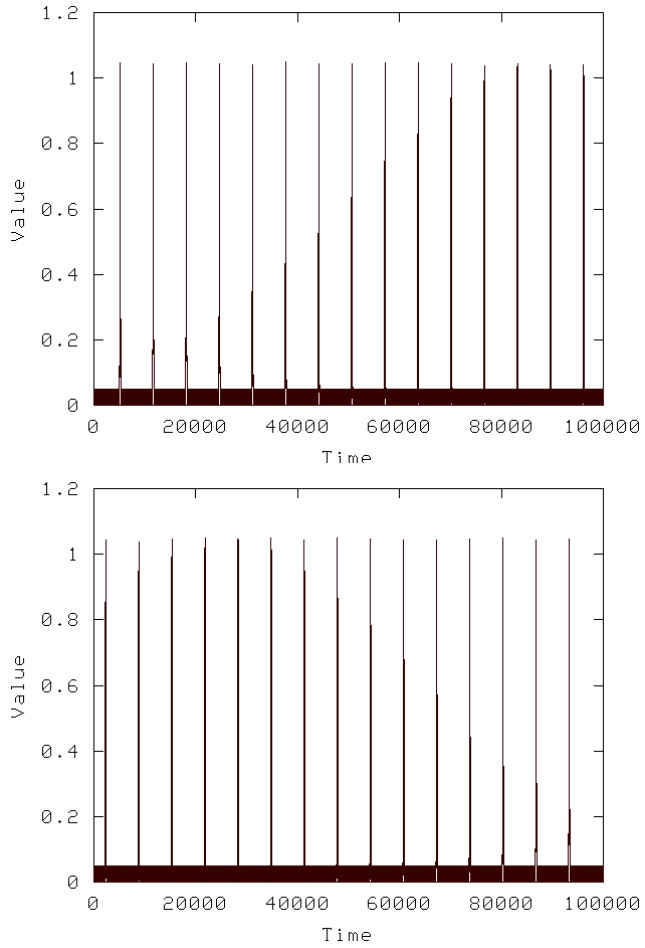
BRAID perfectly estimates the correlation coefficients of the sinusoidal wave



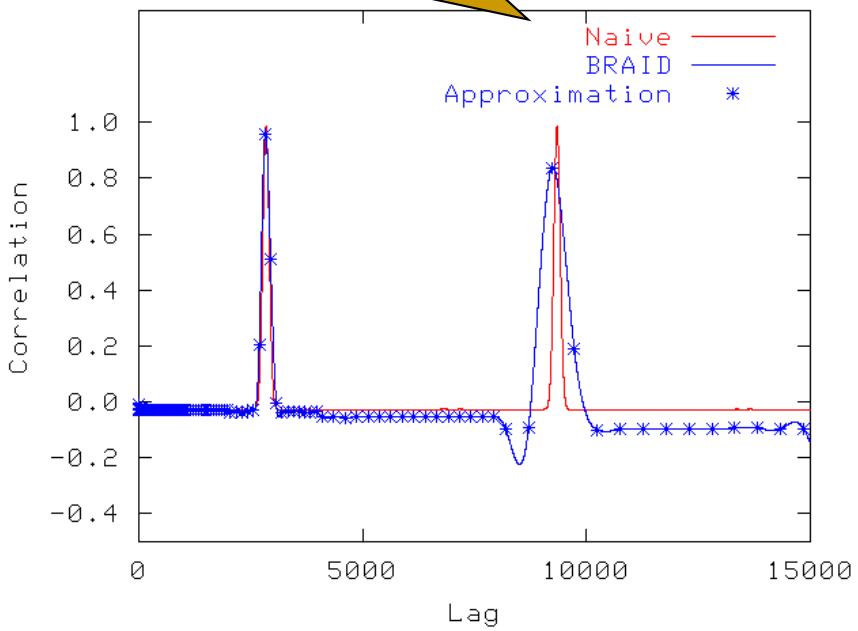
CCF (Cross-Correlation Function)

Accuracy for CCF (2)

- *SpikeTrains*



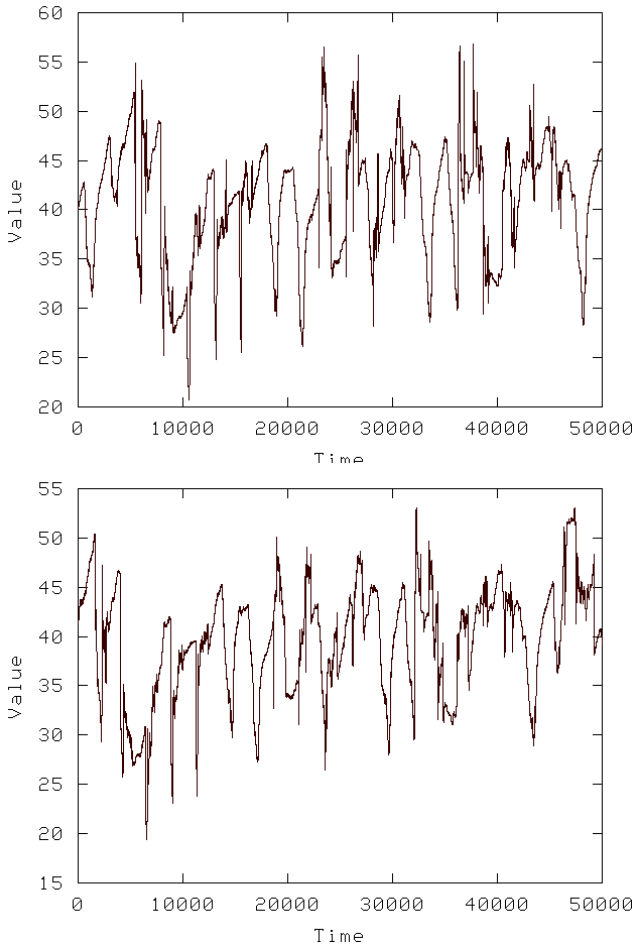
BRAID closely estimates the correlation coefficients



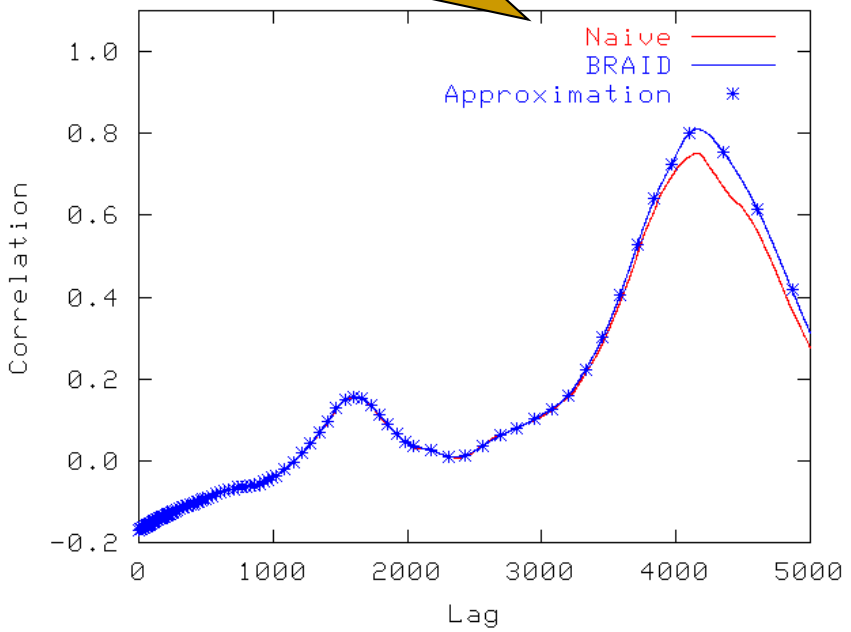
CCF (Cross-Correlation Function)

Accuracy for CCF (3)

- *Humidity* (Real data)



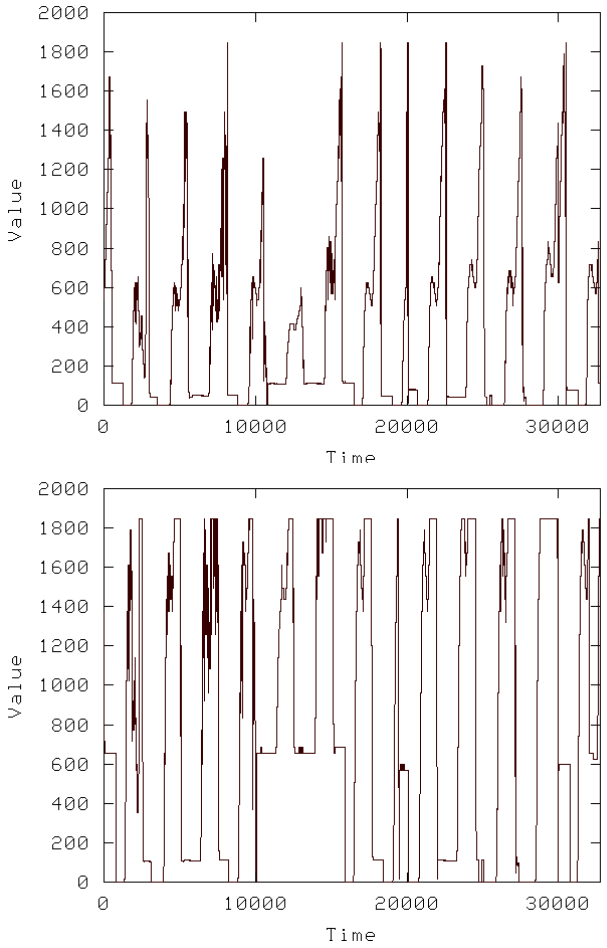
BRAID closely estimates the correlation coefficients



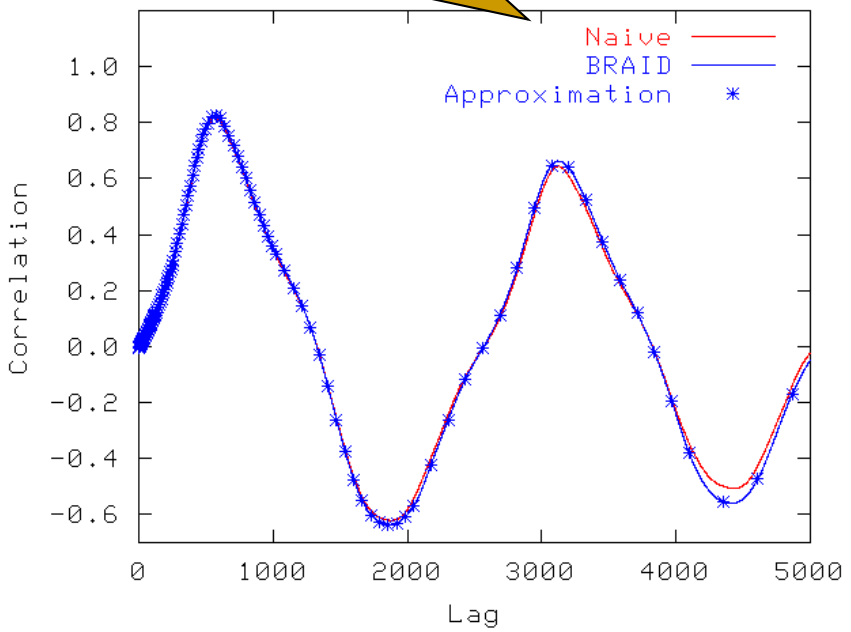
CCF (Cross-Correlation Function)

Accuracy for CCF (4)

- *Light* (Real data)



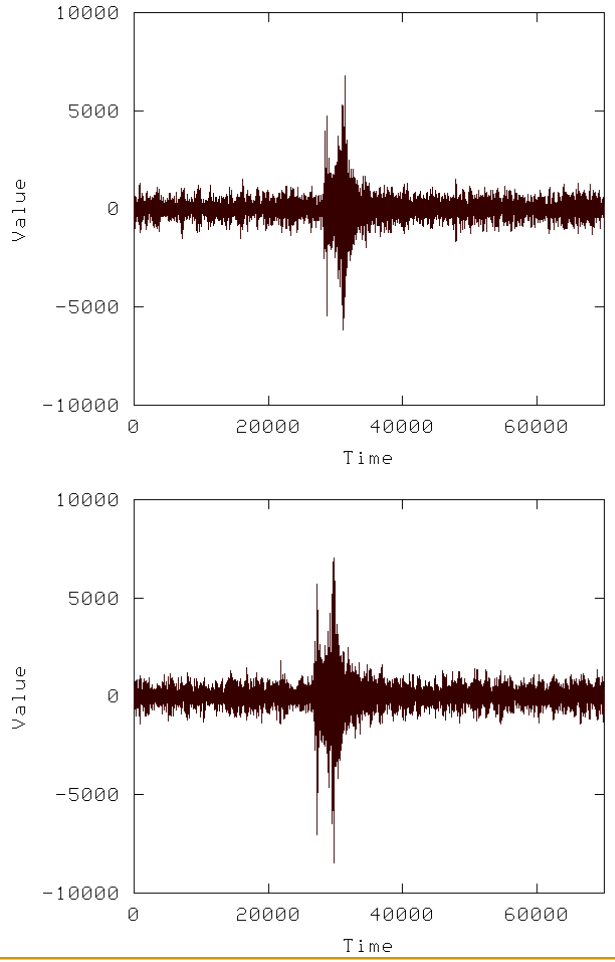
BRAID closely estimates the correlation coefficients



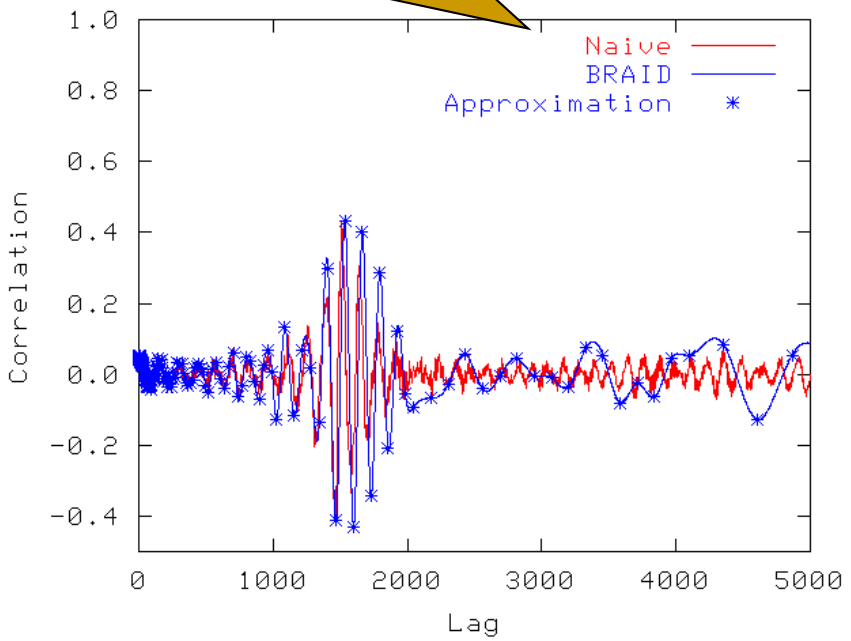
CCF (Cross-Correlation Function)

Accuracy for CCF (5)

- *Kursk* (Real data)



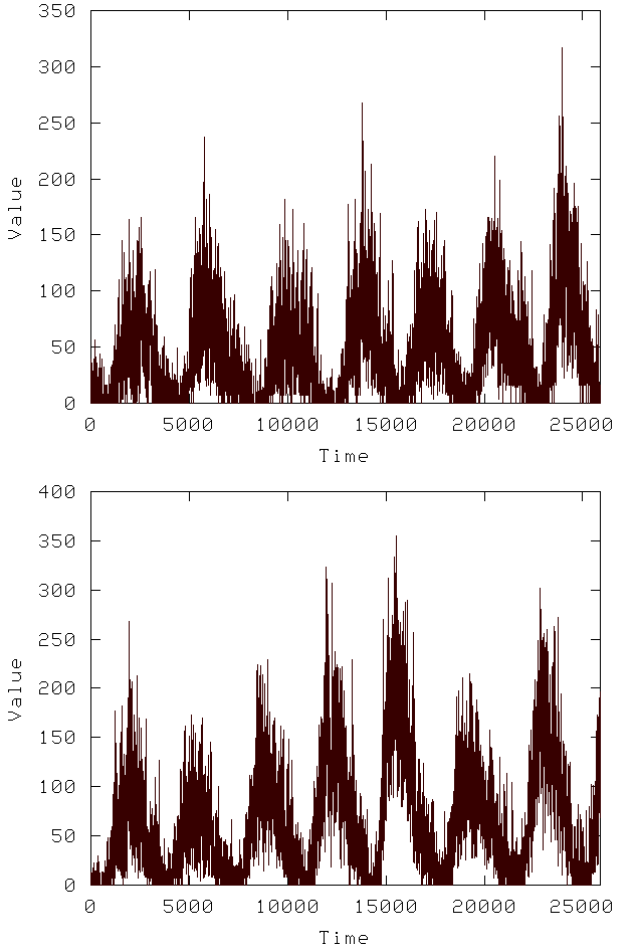
BRAID closely estimates the correlation coefficients



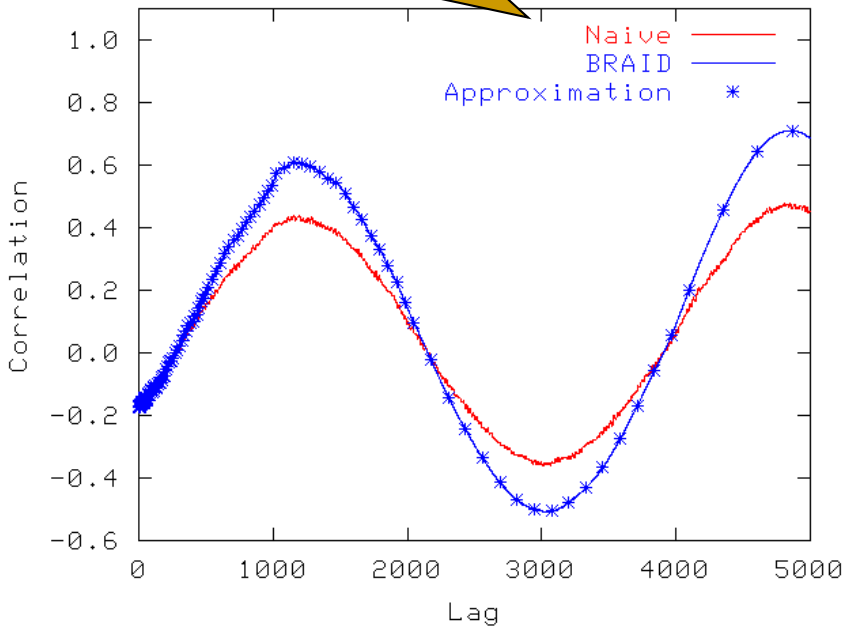
CCF (Cross-Correlation Function)

Accuracy for CCF (6)

- *Sunspots* (Real data)



BRAID closely estimates the correlation coefficients



CCF (Cross-Correlation Function)

Experimental results

- Evaluation
  - Accuracy for CCF
  - Accuracy for the lag estimation
  - Computation time
  - k-way lag correlations

Estimation Error of Lag Correlations

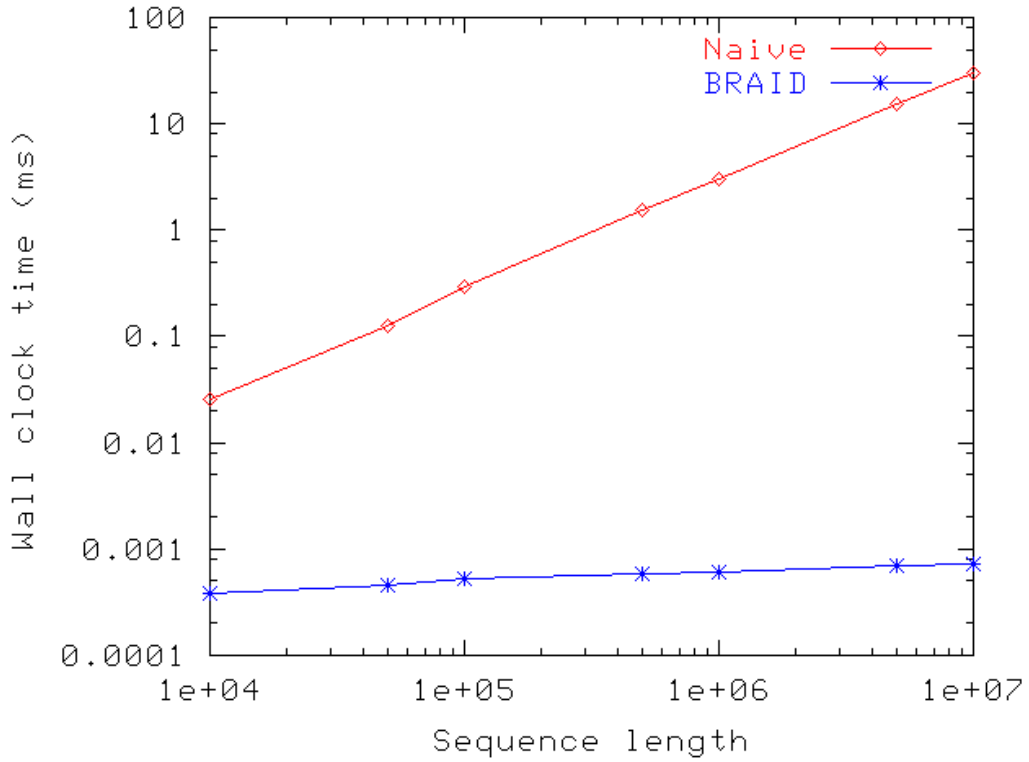
Datasets	Lag correlation		Estimation error (%)
	Naive	BRAID	
Sines	716	716	0.000
SpikeTrains	2841	2830	0.387
Humidity	3842	3855	0.338
Light	567	570	0.529
Kursk	1463	1472	0.615
Sunspots	1156	1168	1.038

- Largest relative error is about 1%

Experimental results

- Evaluation
  - Accuracy for CCF
  - Accuracy for the lag estimation
  - Computation time
  - k-way lag correlations

Computation time



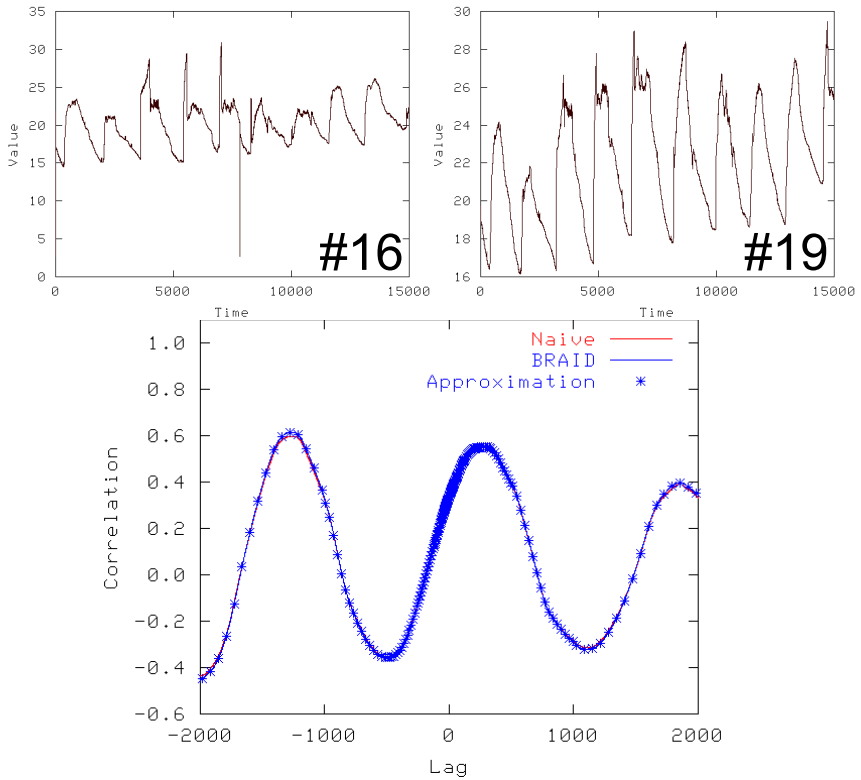
- Reduce computation time dramatically
- Up to 40,000 times faster

Experimental results

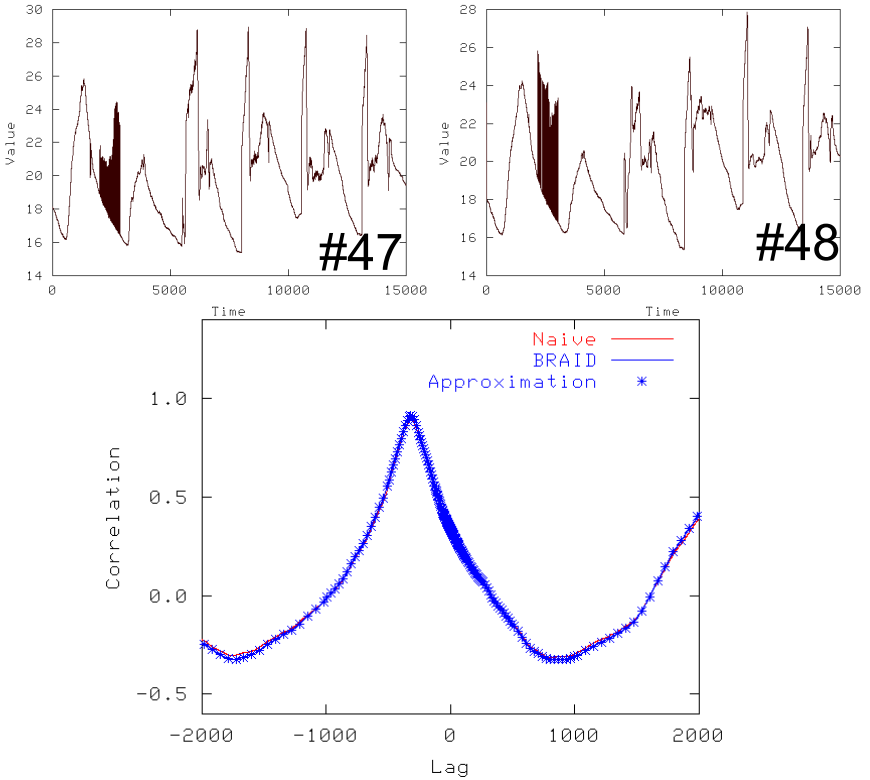
- Evaluation
  - Accuracy for CCF
  - Accuracy for the lag estimation
  - Computation time
  - k-way lag correlations

Group Lag Correlations

- 55 *Temperature* sequences
- Two correlated pairs



Estimation of CCF of #16 and #19



Estimation of CCF of #47 and #48

Conclusions

- Automatic lag correlation detection on data stream

1. 'Any-time'
2. Nimble
   - $O(log\ n)$ space, $O(1)$ time to update the statistics
3. Fast
   - Up to 40,000 times faster than the naive implementation
4. Accurate
   - within 1% relative error or less

Theoretical Analysis - Accuracy

- Effect of geometric lag probing
  - Informally, BRAIDS will provide no error, if lag probing satisfies the sampling theorem (Nyquist's)
  - Formally: Theorem 2

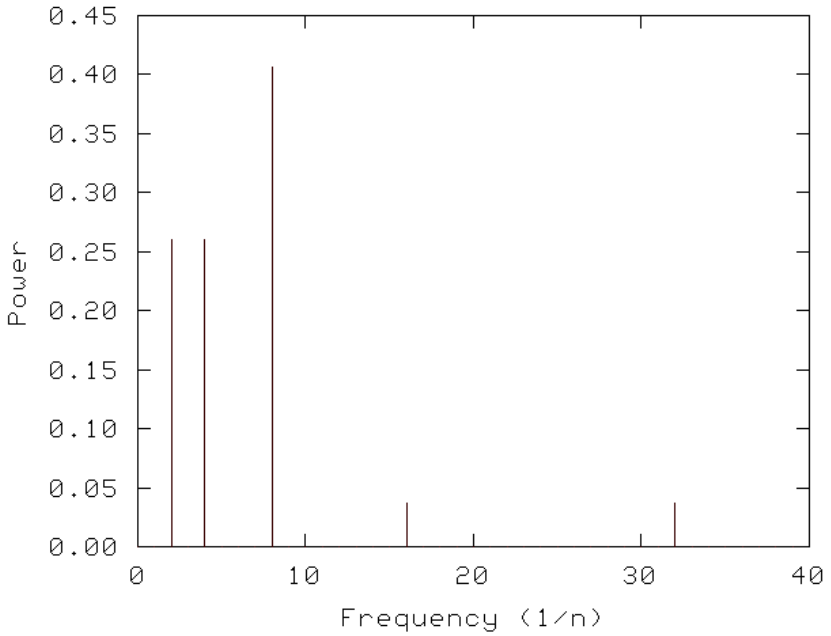
BRAID will find the lag correlations perfectly, if

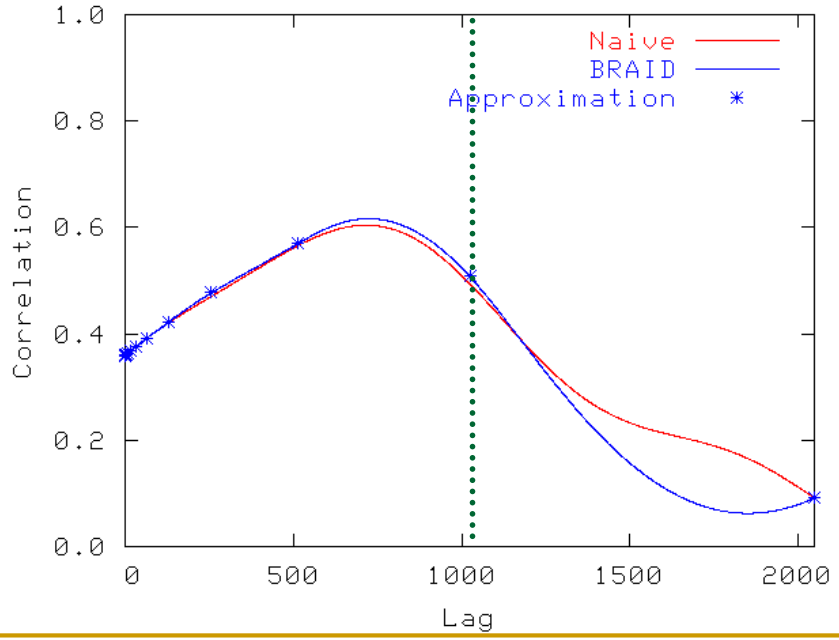
$$0 \le l \le \frac{2b}{f_R}$$

f_R: the Nyquist frequency of CCF, f_R=min(f_x, f_y)
f_x, f_y: the Nyquist frequencies of X and Y

Effect of Probing

- Dataset: *Sines*
- Lag correlation with $b=1$
- $l_R=1024$





Effect of Probing

- Dataset: *Light*
- Lag correlation with $b=1$
- $l_R=630$