AutoLag: Automatic Discovery of Lag Correlations in Stream Data

Yasushi Sakurai *
NTT Cyber Space Laboratories
sakurai.yasushi@lab.ntt.co.jp

Spiros Papadimitriou
Carnegie Mellon University
spapadim@cs.cmu.edu

Christos Faloutsos
Carnegie Mellon University
christos@cs.cmu.edu

1 Introduction

Data streams appear in multiple settings, such as environmental, medical and socioeconomic systems. There are many, fascinating research problems in such settings, like clustering [1], summarization [3], and forecasting [2].

We focus on a less-studied problem, namely to automatically determine whether a sequence X follows another sequence Y, with an unknown lag l, which we need to determine on the fly. The criterion is based on the correlation coefficient $\rho(l)$. Recall that the correlation coefficient $\rho(l)$ for lag l between two time sequences $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$ of equal length n, is defined as follows:

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

$$\bar{x} = \frac{1}{n-l}\sum_{t=l+1}^{n} x_t, \;\; \bar{y} = \frac{1}{n-l}\sum_{t=1}^{n-l} y_t$$

where $\bar{x}$, $\bar{y}$ denote the mean of X and Y, respectively. We are interested in high absolute values of $\rho(l)$.

Definition 1 (Lag Correlation) *We say that two sequences X and Y have a* lag correlation *of l, and specifically that X lags Y by lag l, if the absolute value $|\rho(l)|$ of the correlation coefficient between x_t and y_{t-l} is above a threshold γ, say $\gamma = 0.4$, and this is the global maximum, or the earliest local maximum, if more maxima exist.*

We choose the earliest local maximum, to account for two periodic signals lagging each other.

If X and Y were static, the problem would be trivial: compute the CCF (cross-correlation function) $\rho(l)$ ($l = 1, 2, \ldots$) and report the smallest lag l for which the CCF is maximized. When X and Y continuously increase in length, the problem is challenging. We want a method which will monitor X and Y, and, whenever the user wants, the method should determine (a) whether there is a lag correlation, and (b) if yes, the value l of the lag. We propose AutoLag, a lag capture method for two data streams. AutoLag has dramatically better performance (in terms of speed and memory), while it maintains excellent accuracy.

*This work was done while this author was visiting Carnegie Mellon University.

2 AutoLag

As just mentioned, if we had infinite space and time, the problem would have the following straightforward solution:

Naive Solution At time n, access all the values of X and Y, estimate $\rho(l)$ for all values of the lag l ($= 0, 1, \ldots$), and choose the maximum absolute value above γ, or report that there is no lag correlation.

Our solution is based on three major steps, each described next. The first step towards a streaming solution is the observation that, for a fixed lag l, we can update the correlation coefficient $\rho(l)$ incrementally. Specifically, if we want to estimate the correlation coefficient $\rho(0)$ of two time sequences x_t and y_t ($t = 1, \ldots, n$) at time n, we can do that incrementally, by keeping track of the count (n), the sum of x_t values, the sum of squares x_t^2, the sum of products $x_t y_t$, and the sum and sum of squares of the y_t values. These five numbers are enough to estimate the correlation coefficient $\rho(0)$ for lag $l = 0$. For any desirable value of the lag l, we just need to keep track of the corresponding 5 numbers, which we shall refer to as "sufficient statistics."

For the second step towards a streaming solution, we propose an approximation: Instead of computing $\rho(l)$ for every possible value of the lag l, we propose to keep track of only a geometric progression of lag values: l= 0,1, 2, 4, ..., 2^i, ..., and do interpolation for the rest. We obtained good results with cubic splines, but the choice of interpolation method is orthogonal to our approach, and, in fact, other interpolation methods may give even better results. The justification for our geometric progression idea is that it is necessary to achieve sub-linear space and time requirements, while we expect that the loss in accuracy will be small, if we are interested in the relative error of the lag. The intuition is that our method will give good accuracy for small l, exactly because for small values of the lag l we have many points to interpolate; it may give a larger error for large lag l, but the relative error will probably be small.

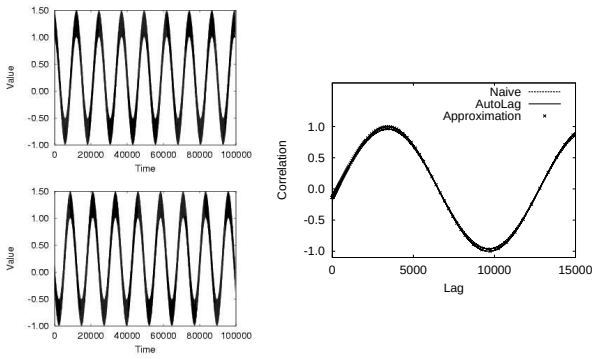

Figure 1. Estimation of the correlation coefficients (CCF) for *Sines*.

However, there is still a subtle point: we still need to have a window of values that is as large as the largest lag l_{max} that we want to be able to detect. This is prohibitive, since l_{max} grows with time. We propose to solve this problem with an approximation: Instead of operating on the original time sequences, we also compute their smoothed version, by computing the means of non-overlapping windows. The window widths will be powers of 2, although any other number would be acceptable, too. Let X be the original time sequence, and X_h be its smoothed version with windows of length 2^h. We refer to h as the "level" of the approximation. That is, $X_0(t)$ is the original sequence; X_1 consists of $n/2$ ticks, with the pair means; X_2 has $n/4$ ticks, with the quadruplet means, and so on. At time n, we need $O(\log n)$ levels; for each level, we compute the sufficient statistics. Formally we have $\rho(l) \approx \rho_1(l/2)$, and in general

$$\rho(l) \approx \rho_h(l/2^h) \tag{2}$$

where $\rho_h()$ is the correlation coefficient of the h-level smoothed sequences, and $\rho_0() \equiv \rho()$. It turns out that, for smooth input sequences, the error is small. In retrospect, it makes sense: if the input sequences are smooth, they won't be affected too much by our smoothing through window-means.

3 Experiments

To evaluate the effectiveness of AutoLag, we performed experiments on real and synthetic datasets. Each synthetic sequence has additive white noise. We compared AutoLag with the naive implementation. AutoLag can keep more than one coefficient for each level to improve the accuracy. We used 16 coefficients for each level. We performed our experiments on an Intel Xeon 2.8GHz with 1GB of memory, running Linux.

Figure 1 shows the estimation of AutoLag for *Sines*. In this figure, "Naive" denotes the exact correlation coefficients computed by the naive implementation. "Approximation" means the correlation coefficients computed from the smoothed version. AutoLag interpolates the missing values between these correlation coefficients. This figure shows that AutoLag closely approximates the correlation coefficients. Table 1 shows that the estimation error of captured lag correlations. The experiments demonstrate clearly that AutoLag detects the correct lag perfectly most of the time. The largest relative error was about 1 %.

Datasets	Lag correlation		Estimation error (%)
	Naive	AutoLag	
Sines	3410	3412	0.059
SpikeTrains	2841	2829	0.422
TwoSpikes	10408	10436	0.269
Temperature	1632	1622	0.613
Kursk	2698	2688	0.371
Sunspots	1156	1168	1.038

Table 1. Estimation error of lag correlations.

Sequence Length	Wall clock time (ms)		Speed-up
	Naive	AutoLag	
1e+04	0.0256	0.00038	67:1
1e+05	0.2896	0.00053	546:1
1e+06	3.0520	0.00061	5003:1
1e+07	29.8844	0.00071	42090:1

Table 2. Wall clock time as a function of sequence length.

Table 2 compares AutoLag and the naive implementation in terms of the wall clock time under varying sequence length n. The wall clock time is the average of processing time to update sufficient statistics and detect lag correlations for each time tick. AutoLag achieves a dramatic reduction in computation time. Specifically, AutoLag is up to about 42,000 times faster than the naive implementation.

4 Conclusions

We have introduced the problem of automatic lag correlation detection on streaming data and proposed AutoLag to address this problem by using careful approximations and smoothing. Our experiments on real and realistic data show that AutoLag works as expected, estimating the unknown lags with excellent accuracy and significant speed-up. In our experiments on real and realistic data, AutoLag was up to about 42,000 times faster than the naive implementation, with at most 1% relative error.

References

[1] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: Theory and practice. *IEEE TKDE*, 15(3):515–528, 2003.

[2] S. Papadimitriou, A. Brockwell, and C. Faloutsos. Adaptive, hands-off stream mining. *VLDB*, pages 560–571, Sept. 2003.

[3] Y. Zhu and D. Shasha. Statistical monitoring of thousands of data streams in real time. *VLDB*, pages 358–369, Aug. 2002.