# Local Correlation Tracking in Time Series

Spiros Papadimitriou[§] Jimeng Sun[‡] Philip S. Yu[§]

[§] IBM T.J. Watson Research Center
Hawthorne, NY, USA

[‡] Carnegie Mellon University
Pittsburgh, PA, USA

## Abstract

*We address the problem of capturing and tracking local correlations among time evolving time series. Our approach is based on comparing the local auto-covariance matrices (via their spectral decompositions) of each series and generalizes the notion of linear cross-correlation. In this way, it is possible to concisely capture a wide variety of local patterns or trends. Our method produces a general similarity score, which evolves over time, and accurately reflects the changing relationships. Finally, it can also be estimated incrementally, in a streaming setting. We demonstrate its usefulness, robustness and efficiency on a wide range of real datasets.*

## 1 Introduction

The notion of correlation (or, similarity) is important, since it allows us to discover groups of objects with similar behavior and, consequently, discover potential anomalies which may be revealed by a change in correlation. In this paper we consider correlation among time series which often exhibit two important properties.

First, their characteristics may change over time. In fact, this is a key property of semi-infinite streams, where data arrive continuously. The term *time-evolving* is often used in this context to imply the presence of non-stationarity. In this case, a single, static correlation score for the entire time series is less useful. Instead, it is desirable to have a notion of correlation that also evolves with time and tracks the changing relationships. On the other hand, a time-evolving correlation score should not be overly sensitive to transients; if the score changes wildly, then its usefulness is limited.

The second property is that many time series exhibit strong but fairly complex, non-linear correlations. Traditional measures, such as the widely used *cross-correlation coefficient* (or, *Pearson coefficient*), are less effective in capturing these complex relationships. From a general point of view, the estimation of a correlation score relies on an assumed joint model of the two sequences. For example, the cross-correlation coefficient assumes that pairs of values from each series follow a simple linear relationship. Consequently, we seek a concise but powerful model that can capture various trend or pattern types.

Data with such features arise in several application domains, such as:

- Monitoring of network traffic flows or of system performance metrics (e.g., CPU and memory utilization, I/O throughput, etc), where changing workload characteristics may introduce non-stationarity.
- Financial applications, where prices may exhibit linear or seasonal trends, as well as time-varying volatility.
- Medical applications, such as EEGs (electroencephalograms) [4].

Figure 1 shows the exchange rates for the French Franc (blue) and the Spanish Peseta (red) versus the US Dollar, over a period of about 10 years. An approximate timeline of major events in the European Monetary Union (EMU) is also included, which may help explain the behavior of each currency. The global cross-correlation coefficient of the two series is 0.30, which is statistically significant (exceeding the 95% confidence interval of ±0.04). The next local extremum of the cross-correlation function is 0.34, at a lag of 323 working days, meaning that the *overall* behavior of the Franc is similar to that of the Peseta 15 months ago, when compared over the entire decade of daily data.

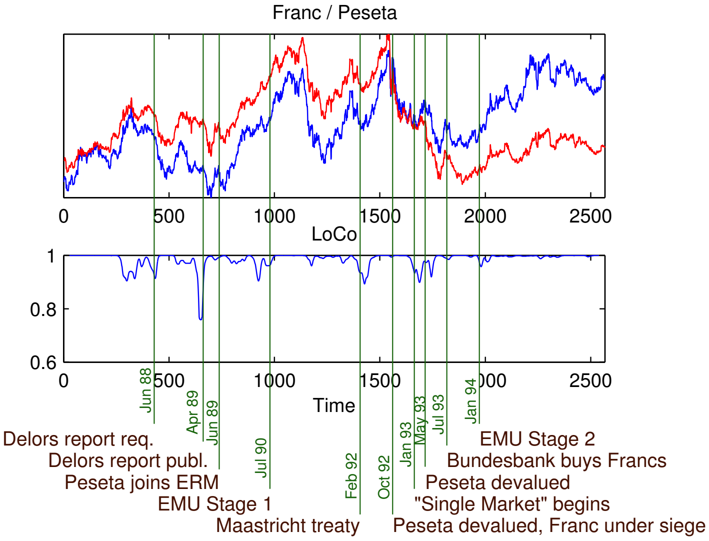


**Figure 1. Illustration of tracking time-evolving local correlations (see also Figure 6).**

This information makes sense and is useful in its own right. However, it is not particularly enlightening about the relationship of the two currencies as they evolve over time. Similar techniques can be employed to characterize correlations or similarities over a period of, say, a few years. But what if we wish to track the evolving relationships over shorter periods, say a few weeks? The bottom part of Figure 1 shows our local correlation score computed over a window of four weeks (or 20 values). It is worth noting that most major EMU events are closely accompanied by a correlation drop, and vice versa. Also, events related to anticipated regulatory changes are typically *preceded*, but not followed, by correlation breaks. Overall, our correlation score smoothly tracks the evolving correlations among the two currencies (cf. Figure 6).

To summarize, our goal is to define a powerful and concise model that can capture complex correlations between time series. Furthermore, the model should allow tracking the time-evolving nature of these correlations in a robust way, which is not susceptible to transients. In other words, the score should accurately reflect the time-varying relationships among the series.

**Contributions.** Our main contributions are the following:

- We introduce LoCo (LOcal COrrelation), a time-evolving, local similarity score for time series, by generalizing the notion of cross-correlation coefficient.
- The model upon which our score is based can capture fairly complex relationships and track their evolution. The linear cross-correlation coefficient is included as a special case.
- Our approach is also amenable to robust streaming estimation.

We illustrate our proposed method or real data, discussing its qualitative interpretation, comparing it against natural alternatives and demonstrating its robustness and efficiency.

The rest of the paper is organized as follows: In Section 2 we briefly describe some of the necessary background and notation. In Section 3 we define some basic notions. Section 4 describes our proposed approach and Section 5 presents our experimental evaluation on real data. Finally, in Section 6 we describe some of the related work and Section 7 concludes.

## 2 Background

In the following, we use lowercase bold letters for column vectors $(\mathbf{u}, \mathbf{v}, \ldots)$ and uppercase bold for matrices $(\mathbf{U}, \mathbf{V}, \ldots)$. The inner product of two vectors is denoted by $\mathbf{x}^\mathsf{T}\mathbf{y}$ and the outer product by $\mathbf{x} \otimes \mathbf{y} \equiv \mathbf{x}\mathbf{y}^\mathsf{T}$. The Euclidean norm of $\mathbf{x}$ is $\|\mathbf{x}\|$. We denote a time series process as an indexed collection $X$ of random variables $X_t, t \in \mathbb{N}$, i.e., $X = \{X_1, X_2, \ldots, X_t, \ldots\} \equiv \{X_t\}_{t \in \mathbb{N}}$. Without loss of generality, we will assume zero-mean time series, i.e., $\mathrm{E}[X_t] = 0$ for all $t \in \mathbb{N}$. The values of a particular realization of $X$ are denoted by lower-case letters, $x_t \in \mathbb{R}$, at time $t \in \mathbb{N}$.

**Covariance and autocovariance.** The covariance of two random variables $X, Y$ is defined as $\mathrm{Cov}[X, Y] = \mathrm{E}[(X - \mathrm{E}[X])(Y - \mathrm{E}[Y])]$. If $X_1, X_2, \ldots, X_m$ is a group of $m$ random variables, their covariance matrix $\mathbf{C} \in \mathbb{R}^{m \times m}$ is the symmetric matrix defined by $c_{ij} := \mathrm{Cov}[X_i, X_j]$, for $1 \le i, j \le m$. If $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ is a collection of $n$ observations $\mathbf{x}_i \equiv [x_{i,1}, x_{i,2}, \ldots, x_{i,m}]^\mathsf{T}$ of all $m$ variables, the sample covariance estimate[1] is defined as

$$\hat{\mathbf{C}} := \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \otimes \mathbf{x}_i.$$

In the context of a time series process $\{X_t\}_{t \in \mathbb{N}}$, we are interested in the relationship between values at different times. To that end, the autocovariance is defined as $\gamma_{t,t'} := \mathrm{Cov}[X_t, X_{t'}] = \mathrm{E}[X_t X_{t'}]$, where the last equality follows from the zero-mean assumption. By definition, $\gamma_{t,t'} = \gamma_{t',t}$.

**Spectral decomposition.** Any real symmetric matrix is always equivalent to a diagonal matrix, in the following sense.

**Theorem 1.** *If $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a symmetric, real matrix, then it is always possible to find a column-orthonormal matrix $\mathbf{U} \in \mathbb{R}^{n \times n}$ and a diagonal matrix $\mathbf{\Lambda} \in \mathbb{R}^{n \times n}$, such that $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\mathsf{T}$.*

Thus, given any vector $\mathbf{x}$, we can write $\mathbf{U}^\mathsf{T}(\mathbf{A}\mathbf{x}) = \mathbf{\Lambda}(\mathbf{U}^\mathsf{T}\mathbf{x})$, where pre-multiplication by $\mathbf{U}^\mathsf{T}$ amounts to a change of coordinates. Intuitively, if we use the coordinate system defined by $\mathbf{U}$, then $\mathbf{A}\mathbf{x}$ can be calculated by simply scaling each coordinate independently of all the rest (i.e., multiplying by the diagonal matrix $\mathbf{\Lambda}$).

Given any symmetric matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, we will denote its eigenvectors by $\mathbf{u}_i(\mathbf{A})$ and the corresponding eigenvalues by $\lambda_i(\mathbf{A})$, in order of decreasing magnitude, where $1 \le i \le n$. The matrix $\mathbf{U}_k(\mathbf{A})$ has the first $k$ eigenvectors as its columns, where $1 \le k \le n$.

The covariance matrix $\mathbf{C}$ of $m$ variables is symmetric by definition. Its spectral decomposition provides the directions in $\mathbb{R}^m$ that "explain" the most of the variance. If we project $[X_1, X_2, \ldots, X_m]^\mathsf{T}$ onto the subspace spanned by $\mathbf{U}_k(\mathbf{C})$, we retain the largest fraction of variance among any other $k$-dimensional subspace [11]. Finally, the autocovariance matrix of a finite-length time series is also symmetric and its eigenvectors typically capture both the key

[1] The unbiased estimator uses $n-1$ instead of $n$, but this constant factor does not affect the eigen-decomposition.

oscillatory (e.g., sinusoidal) as well as aperiodic (e.g., increasing or decreasing) trends that are present [6, 7].

## 3 Localizing correlation estimates

Our goal is to derive a time-evolving correlation scores that tracks the similarity of time-evolving time series. Thus, our method should have the following properties:

(P1) Adapt to the time-varying nature of the data,

(P2) Employ a simple, yet powerful and expressive joint model to capture correlations,

(P3) The derived score should be robust, reflecting the evolving correlations accurately, and

(P4) It should be possible to estimate it efficiently.

We will address most of these issues in Section 4, which describes our proposed method. In this section, we introduce some basic definitions to facilitate our discussion. We also introduce localized versions of popular similarity measures for time series.

As a first step to deal with (P1), any correlation score at time instant $t \in \mathbb{N}$ should be based on observations in the "neighborhood" of that instant. Therefore, we introduce the notation $\mathbf{x}_{t,w} \in \mathbb{R}^w$ for the subsequence of the series, starting at $t$ and having length $w$,

$$\mathbf{x}_{t,w} := [x_t, x_{t+1}, \ldots, x_{t+w-1}]^\mathsf{T}.$$

Furthermore, any correlation score should satisfy two elementary and intuitive properties.

**Definition 1** (Local correlation score)**.** *Given a pair of time series $X$ and $Y$, a* local correlation score *is a sequence $c_t(X,Y)$ of real numbers that satisfy the following properties, for all $t \in \mathbb{N}$:*

$$0 \le c_t(X,Y) \le 1 \quad \textit{and} \quad c_t(X,Y) = c_t(Y,X).$$

### 3.1 Local Pearson

Before proceeding to describe our approach, we formally define a natural extension of a method that has been widely used for global correlation or similarity among "static" time series.

**Pearson coefficient.** A natural local adaptation of cross-correlation is the following:

**Definition 2** (Local Pearson correlation)**.** *The local Pearson correlation is the linear cross-correlation coefficient*

$$\rho_t(X,Y) := \frac{\left|\mathrm{Cov}[\mathbf{x}_{t,w}, \mathbf{y}_{t,w}]\right|}{\mathrm{Var}[\mathbf{x}_{t,w}]\,\mathrm{Var}[\mathbf{y}_{t,w}]} = \frac{|\mathbf{x}_{t,w}^\mathsf{T}\mathbf{y}_{t,w}|}{\|\mathbf{x}_{t,w}\| \cdot \|\mathbf{y}_{t,w}\|},$$

*where the last equality follows from* $\mathrm{E}[X_t] = \mathrm{E}[Y_t] = 0$.

It follows directly from the definition that $\rho_t$ satisfies the two requirements, $0 \le \rho_t(X,Y) \le 1$ and $\rho_t(X,Y) = \rho_t(Y,X)$.

| Symbol | Description |
|---|---|
| $\mathbf{U}, \mathbf{V}$ | Matrix (uppercase bold). |
| $\mathbf{u}, \mathbf{v}$ | Column vector (lowercase bold). |
| $x_t$ | Time series, $t \in \mathbb{N}$. |
| $w$ | Window size. |
| $\mathbf{x}_{t,w}$ | Window starting at $t$, $\mathbf{x}_{t,w} \in \mathbb{R}^w$. |
| $m$ | Number of windows (typically, $m = w$). |
| $\beta$ | Exponential decay weight, $0 \le \beta \le 1$. |
| $\hat{\Gamma}_t$ | Local autocorrelation matrix estimate. |
| $\mathbf{u}_i(\mathbf{A})$,<br>$\lambda_i(\mathbf{A})$ | Eigenvectors and corresponding<br>eigenvalues of $\mathbf{A}$. |
| $\mathbf{U}_k(\mathbf{A})$ | Matrix of $k$ largest eigenvectors of $\mathbf{A}$. |
| $\ell_t$ | LoCo score. |
| $\rho_t$ | Pearson local correlation score. |

**Table 1. Description of main symbols.**

## 4 Correlation tracking through local autocovariance

In this section we develop our proposed approach, the *Local Correlation (LoCo)* score. Returning to properties (P1)–(P4) listed in the beginning of Section 3, the next section addresses primarily (P1) and Section 4.2 continues to address (P2) and (P3). Next, Section 4.3 shows how (P4) can also be satisfied and, finally, Section 4.4 discusses the time and space complexity of the various alternatives.

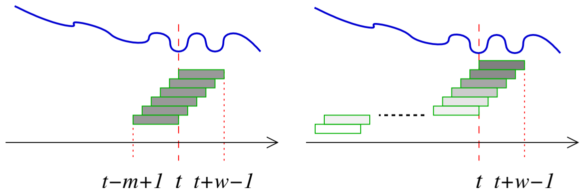
### 4.1 Local autocovariance

The first step towards tracking local correlations at time $t \in \mathbb{N}$ is restricting, in some way, the comparison to the "neighborhood" of $t$, which is the reason for introducing the notion of a window $\mathbf{x}_{t,w}$.

If we stop there, we can compare the two windows $\mathbf{x}_{t,w}$ and $\mathbf{y}_{t,w}$ *directly*. If, in addition, the comparison involves capturing any *linear* relationships between localized values of $X$ and $Y$, this leads to the local Pearson correlation score $\rho_t$. However, this joint model of the series it is too simple, leading to two problems: (i) it cannot capture more complex relationships, and (ii) it is too sensitive to transient changes, often leading to widely fluctuating scores.

Intuitively, we address the first issue by estimating the *full* autocovariance matrix of values "near" $t$, and avoid making *any* assumptions about stationarity (as will be explained later). Any estimate of the local autocovariance at time $t$ needs to be based on a "localized" sample set of windows with length $w$. We will consider two possibilities:

- Sliding (a.k.a. boxcar) window (see Figure 2a): We use a exactly $m$ windows around $t$, specifically $\mathbf{x}_{\tau,w}$ for $t-m+1 \le \tau \le t$, and we weigh them equally. This takes into account $w + m - 1$ values in total, around time $t$.



(a) Sliding window (b) Exponential window

**Figure 2. Local auto-covariance; shading corresponds to weight.**

- Exponential window (see Figure 2b): We use all windows $\mathbf{x}_{\tau,w}$ for $1 \leq \tau \leq t$, but we weigh those close to $t$ more, by multiplying each window by a factor of $\beta^{t-\tau}$.

These two alternatives are illustrated in Figure 2, where the shading corresponds to the weight. We will explain how to "compare" the local autocovariance matrices of two series in Section 4.2. Next, we formally define these estimators.

**Definition 3** (Local autocovariance, sliding window)**.** *Given a time series $X$, the local autocovariance matrix estimator $\hat{\Gamma}_t$ using a sliding window is defined at time $t \in \mathbb{N}$ as*

$$\hat{\Gamma}_t(X, w, m) := \sum_{\tau=t-m+1}^{t} \mathbf{x}_{\tau,w} \otimes \mathbf{x}_{\tau,w}.$$

The sample set of $m$ windows is "centered" around time $t$. We typically fix the number of windows to $m = w$, so that $\hat{\Gamma}_t(X, w, m) = \sum_{\tau=t-w+1}^{t} \mathbf{x}_{\tau,w} \otimes \mathbf{x}_{\tau,w}$. A normalization factor of $1/m$ is ignored, since it is irrelevant for the eigenvectors of $\hat{\Gamma}_t$.

**Definition 4** (Local autocovariance, exponential window)**.** *Given a time series $X$, the local autocovariance matrix estimator $\hat{\Gamma}_t$ at time $t \in \mathbb{N}$ using an exponential window is*

$$\hat{\Gamma}_t(X, w, \beta) := \sum_{\tau=1}^{t} \beta^{t-\tau} \mathbf{x}_{\tau,w} \otimes \mathbf{x}_{\tau,w}.$$

Similar to the previous definition, we ignore the normalization factor $(1-\beta)/(1-\beta^{t+1})$. In both cases, we may omit some or all of the arguments $X$, $w$, $m$, $\beta$, when they are clear from the context.

Under certain assumptions, the *equivalent window* corresponding to an exponential decay factor $\beta$ is given by $m = (1-\beta)^{-1}$ [22]. However, one of the main benefits of the exponential window is based on the following simple observation.

**Property 1.** *The sliding window local autocovariance follows the equation*

$$\hat{\Gamma}_t = \hat{\Gamma}_{t-1} - \mathbf{x}_{t-w,w} \otimes \mathbf{x}_{t-w,w} + \mathbf{x}_{t,w} \otimes \mathbf{x}_{t,w},$$

*whereas for the exponential window it follows the equation*

$$\hat{\Gamma}_t = \beta \hat{\Gamma}_{t-1} + \mathbf{x}_{t,w} \otimes \mathbf{x}_{t,w}.$$

An incremental update to the sliding window estimator has rank 2, whereas an update to the exponential window estimator has rank 1, which can be handled more efficiently. Also, updating the sliding window estimator requires subtraction of $\mathbf{x}_{t-w+1,w} \otimes \mathbf{x}_{t-w+1,w}$, which means that by necessity, the past $w$ values of $X$ need to be stored (or, in general, the past $m$ values), in addition to the "future" $w$ values of $\mathbf{x}_{t,w}$ that need to be buffered. Since, as we will see, the local correlation scores derived from these estimators are very close, using an exponential window is more desirable.

The next simple lemma will be useful later, to show that $\rho_t$ is included as a special case of the LoCo score. Intuitively, if we use an *instantaneous* estimate of the local autocovariance $\hat{\Gamma}_t$, which is based on just the latest sample window $\mathbf{x}_{t,w}$, its eigenvector is the window itself.

**Lemma 1.** *If $m = 1$ or, equivalently, $\beta = 0$, then*

$$\mathbf{u}_1(\hat{\Gamma}_t) = \frac{\mathbf{x}_{t,w}}{\|\mathbf{x}_{t,w}\|} \quad \text{and} \quad \lambda_1(\hat{\Gamma}_t) = \|\mathbf{x}_{t,w}\|^2.$$

*Proof.* In this case, $\hat{\Gamma}_t = \mathbf{x}_{t,w} \otimes \mathbf{x}_{t,w}$ with rank 1. Its row and column space are span $\mathbf{x}_{t,w}$, whose orthonormal basis is, trivially, $\mathbf{x}_{t,w}/\|\mathbf{x}_{t,w}\| \equiv \mathbf{u}_1(\hat{\Gamma}_t)$. The fact that $\lambda_1(\hat{\Gamma}_t) = \|\mathbf{x}_{t,w}\|^2$ then follows by straightforward computation, since $\mathbf{u}_1 \otimes \mathbf{u}_1 = \mathbf{x}_{t,w} \otimes \mathbf{x}_{t,w}/\|\mathbf{x}_{t,w}\|^2$, thus $(\mathbf{x}_{t,w} \otimes \mathbf{x}_{t,w})\mathbf{u}_1 = \|\mathbf{x}_{t,w}\|^2 \mathbf{u}_1$. □

## 4.2 Pattern similarity

Given the estimates $\hat{\Gamma}_t(X)$ and $\hat{\Gamma}_t(Y)$ for the two series, the next step is how to "compare" them and extract a correlation score. Intuitively, we want to extract the "key information" contained in the autocovariance matrices and measure how close they are. This is precisely where the spectral decomposition helps. The eigenvectors capture the key aperiodic and oscillatory trends, even in short, non-stationary series [6, 7]. These trends explain the largest fraction of the variance. Thus, we will use the subspaces spanned by the first few ($k$) eigenvectors of each local autocovariance matrix to locally characterize the behavior of each series. The following definition formalizes this notion.

**Definition 5** (LoCo score)**.** *Given two series $X$ and $Y$, their LoCo score is defined by*

$$\ell_t(X, Y) := \tfrac{1}{2}\left(\|\mathbf{U}_X^T \mathbf{u}_Y\| + \|\mathbf{U}_Y^T \mathbf{u}_X\|\right),$$

*where $\mathbf{U}_X \equiv \mathbf{U}_k(\hat{\Gamma}_t(X))$ and $\mathbf{U}_Y \equiv \mathbf{U}_k(\hat{\Gamma}_t(Y))$ are the eigenvector matrices of the local autocovariance matrices of $X$ and $Y$, respectively, and $\mathbf{u}_X \equiv \mathbf{u}_1(\hat{\Gamma}_t(X))$ and $\mathbf{u}_Y \equiv \mathbf{u}_1(\hat{\Gamma}_t(Y))$ are the corresponding eigenvectors with the largest eigenvalue.*

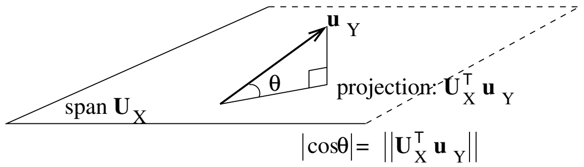
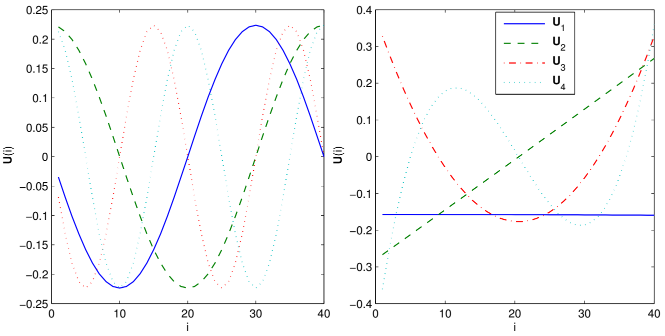

u Y
θ
projection: $\mathbf{U}_X^\mathsf{T}\mathbf{u}_Y$
span $\mathbf{U}_X$
$|\cos\theta| = \|\mathbf{U}_X^\mathsf{T}\mathbf{u}_Y\|$

**Figure 3. Illustration of LoCo definition.**



(a) Periodic

(b) Polynomial trend

**Figure 4. First four eigenvectors ($w = 40$) for (a) periodic series, $x_t = 2sin(2\pi t/40) + sin(2\pi t/20)$ and, (b) polynomial trend, $x_t = t^3$.**

In the above equation, $\mathbf{U}_X^\mathsf{T}\mathbf{u}_Y$ is the projection of $\mathbf{u}_Y$ onto the subspace spanned by the columns of the orthonormal matrix $\mathbf{U}_X$. The absolute cosine of the angle $\theta \equiv \angle(\mathbf{u}_Y, \operatorname{span}\mathbf{U}_X) = \angle(\mathbf{u}_Y, \mathbf{U}_X^\mathsf{T}\mathbf{u}_Y)$ is $|\cos\theta| = \|\mathbf{U}_X^\mathsf{T}\mathbf{u}_Y\|/\|\mathbf{u}_Y\| = \|\mathbf{U}_X^\mathsf{T}\mathbf{u}_Y\|$, since $\|\mathbf{u}_Y\| = 1$ (see Figure 3). Thus, $\ell_t$ is the average of the cosines $|\cos\angle(\mathbf{u}_Y, \operatorname{span}\mathbf{U}_X)|$ and $|\cos\angle(\mathbf{u}_X, \operatorname{span}\mathbf{U}_Y)|$. From this definition, it follows that $0 \leq \ell_t(X,Y) \leq 1$ and $\ell_t(X,Y) = \ell_t(Y,X)$. Furthermore, $\ell_t(X,Y) = \ell_t(-X,Y) = \ell_t(Y,-X) = \ell_t(-X,-Y)$—as is also the case with $\rho_t(X,Y)$.

Intuitively, if the two series $X, Y$ are locally similar, then the principal eigenvector of each series should lie within the subspace spanned by the principal eigenvectors of the other series. Hence, the angles will be close to zero and the cosines will be close to one.

The next simple lemma reveals the relationship between $\rho_t$ and $\ell_t$.

**Lemma 2.** *If $m = 1$ (whence, $k = 1$ necessarily), then $\ell_t = \rho_t$.*

*Proof.* From Lemma 1 it follows that $\mathbf{U}_X = \mathbf{u}_X = \mathbf{x}_{t,w}/\|\mathbf{x}_{t,w}\|$ and $\mathbf{U}_Y = \mathbf{u}_Y = \mathbf{y}_{t,w}/\|\mathbf{y}_{t,w}\|$. From the definitions of $\ell_t$ and $\rho_t$, we have $\ell_t = \frac{1}{2}\left(\frac{|\mathbf{x}_{t,w}^\mathsf{T}\mathbf{y}_{t,w}|}{\|\mathbf{x}_{t,w}\|\cdot\|\mathbf{y}_{t,w}\|} + \frac{|\mathbf{y}_{t,w}^\mathsf{T}\mathbf{x}_{t,w}|}{\|\mathbf{y}_{t,w}\|\cdot\|\mathbf{x}_{t,w}\|}\right) = \frac{|\mathbf{x}_{t,w}^\mathsf{T}\mathbf{y}_{t,w}|}{\|\mathbf{x}_{t,w}\|\cdot\|\mathbf{y}_{t,w}\|} = \rho_t$. □

**Choosing $k$.** As we shall see also see in Section 5, the directions of $\mathbf{x}_{t,w}$ and $\mathbf{y}_{t,w}$ may vary significantly, even at neighboring time instants. As a consequence, the Pearson score $\rho_t$ (which is essentially based on the instantaneous estimate of the local autocovariance) is overly sensitive. However, if we consider the low-dimensional subspace which is (mostly) occupied by the windows during a short period of time (as LoCo does), this is much more stable and less susceptible to transients, while still able to track changes in local correlation.

One approach is to set $k$ based on the fraction of variance to retain (similar to criteria used in PCA [11], as well as in spectral estimation [19]). A simpler practical choice is to fix $k$ to a small value; we use $k = 4$ throughout all experiments. From another point of view, key aperiodic trends are captured by one eigenvector, whereas key oscillatory trends manifest themselves in a pair of eigenvectors with similar eigenvalues [6, 7]. The former (aperiodic trends) are mostly present during "unstable" periods of time, while the latter (periodic, or oscillatory trends) are mostly present during "stable" periods. The eigen-decomposition can capture both and fixing $k$ amounts to selecting a number of trends for our comparison. The fraction of variance captured in the real series of our experiments with $k = 4$ is typically between 90–95%.

**Choosing $w$.** Windows are commonly used in stream and signal processing applications. The size $w$ of each window $\mathbf{x}_{t,w}$ (and, consequently, the size $w \times w$ of the autocovariance matrix $\hat{\Gamma}_t$) essentially corresponds to the time scale we are interested in.

As we shall also see in Section 5, the LoCo score $\ell_t$ derived from the local autocovariances changes gradually and smoothly with respect to $w$. Thus, if we set the window size to any of, say, 55, 60 or 65 seconds, we will qualitatively get the same results, corresponding approximately to patterns in the minute scale. Of course, at widely different time scales, the correlation scores will be different. If desirable, it is possible to track the correlation score at multiple scales, e.g., hour, day, month and year. If buffer space and processing time are a concern, either a simple decimated moving average filtering scheme or a more elaborate hierarchical SVD scheme (such as in [16]) can be employed—these considerations are beyond the scope of this paper.

**Types of patterns.** We next consider two characteristic special cases, which illustrate how the eigenvectors of the autocovariance matrix capture both aperiodic and oscillatory trends [7].

We first consider the case of a weakly stationary series. In this case, it follows from the definition of stationarity that the autocorrelation depends only on the time distance, i.e., $\gamma_{t,t'} \equiv \gamma_{|t-t'|}$. Consequently, its local autocovariance matrix is *circulant*, i.e., it it symmetric with constant diagonals. Its estimate $\hat{\Gamma}_t$ will have the same property, provided that the sample size $m$ (i.e., number of windows used by the estimator) is sufficiently large. However, the eigenvectors of a circulant matrix are the Fourier basis vectors. If we additionally consider real-valued series, these observations lead to the following lemma.

**Lemma 3** (Stationary series)**.** *If $X$ is weakly stationary, then the eigenvectors of the local autocovariance matrix (as $m \to \infty$) are sinusoids. The number of non-zero eigenvalues is twice the number of frequencies present in $X$.*

Figure 4a illustrates the four eigenvectors of the autocovariance matrix for a series consisting of two frequencies. The eigenvectors are pairs of sinusoids with the same frequencies and phases different by $\pi/2$. In practice, the estimates derived using the singular value decomposition (SVD) on a finite sample size of $m = w$ windows have similar properties [19].

Next, we consider simple polynomial trends, $x_t = t^k$ for a fixed $k \in \mathbb{N}$. In this case, the window vectors are always polynomials of degree $k$, $\mathbf{x}_{t,w} = [t^k, (t+1)^k, \ldots, (t+w-1)^k]^\mathsf{T}$. In other words, they belong to the span of $\{1, t, t^2, \ldots, t^k\}$, leading to the next simple lemma.

**Lemma 4** (Trends)**.** *If $X$ is a polynomial of degree $k$, then the eigenvectors of $\hat{\Gamma}_t$ are polynomials of the same degree. The number of non-zero eigenvalues is $k+1$.*

Figure 4b illustrates the four eigenvectors of the autocovariance matrix for a cubic monomial. The eigenvectors are polynomials of degrees zero to three, which are similar to Chebyshev polynomials [3].

In practice, if a series consists locally of a mix of oscillatory and aperiodic patterns, then the eigenvectors of the local autocovariance matrix will be linear combinations of the above types of functions (sinusoids at a few frequencies and low-degree polynomials). By construction, these mixtures locally capture the maximum variance.

## 4.3 Online estimation

In this section we show how $\ell_t$ can be incrementally updated in a streaming setting. We also briefly discuss how to update $\rho_t$.

**LoCo score.** The eigenvector estimates of the exponential window local autocovariance matrix can be updated incrementally, by employing eigenspace tracking algorithms. For completeness, we show above one such algorithm [22] which, among several alternatives, has very good accuracy with limited resource requirements.

**Procedure 1** EIGENUPDATE ($\mathbf{V}_{t-1}$, $\mathbf{C}_{t-1}$, $\mathbf{x}_{t,w}$, $\beta$)

```
$\mathbf{V}_t \in \mathbb{R}^{w \times k}$: basis for $k$-dim. principal eigenspace of $\hat{\Gamma}_t$
$\mathbf{C}_t \in \mathbb{R}^{k \times k}$: covariance w.r.t. columns of $\mathbf{V}_t$
$\mathbf{x}_{t,w} \in \mathbb{R}^w$: new window with arriving value $x_{t+w}$
$0 < \beta \leq 1$: exponential decay factor

$\mathbf{y} := \mathbf{V}_{t-1}^\mathsf{T} \mathbf{x}_{t,w}$
$\mathbf{h} := \mathbf{C}_{t-1} \mathbf{y}$
$\mathbf{g} := \mathbf{h} / (\beta + \mathbf{y}^\mathsf{T} \mathbf{h})$
$\epsilon := \mathbf{x}_{t+1,w} - \mathbf{V}_{t-1} \mathbf{y}$
$\mathbf{V}_t \leftarrow \mathbf{V}_{t-1} + \epsilon \otimes \mathbf{g}$
$\mathbf{C}_t \leftarrow (\mathbf{C}_{t-1} - \mathbf{g} \otimes \mathbf{h}) / \beta$
return $\mathbf{V}_t$, $\mathbf{C}_t$
```

This simple procedure will track the $k$-dimensional eigenspace of $\hat{\Gamma}_t(X, w, \beta)$. More specifically, the matrix $\mathbf{V}_t \in \mathbb{R}^{w \times k}$ will span the same $k$-dimensional subspace as $\mathbf{U}_k(\hat{\Gamma}_t)$. Its columns may not be orthonormal, but that can be easily addressed by performing an orthonormalization step. The matrix $\mathbf{C}_t$ is the covariance in the coordinate system defined by $\mathbf{V}_t$, which is not necessarily diagonal since the columns of $\mathbf{V}_t$ do not have to be the individual eigenvectors. The first eigenvector is simply the one-dimensional eigenspace and can also be estimated using EIGENUPDATE. The detailed pseudocode is shown below.

**Algorithm 1** STREAMLOCO

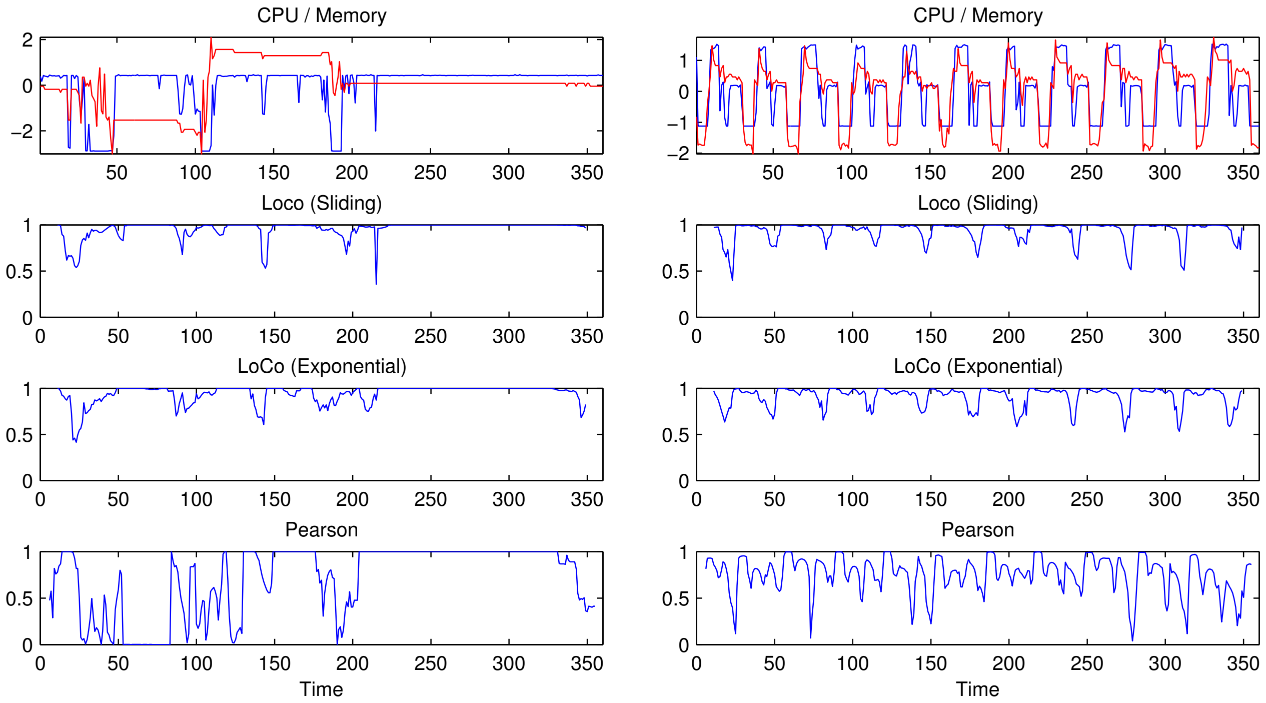
```
Eigenvector estimates $\tilde{\mathbf{u}}_X, \tilde{\mathbf{u}}_Y \in \mathbb{R}^w$
Eigenvalue estimates $\tilde{\lambda}_X, \tilde{\lambda}_Y \in \mathbb{R}$
Eigenspace estimates $\tilde{\mathbf{U}}_x, \tilde{\mathbf{U}}_Y \in \mathbb{R}^{w \times k}$
Covariance (eigen-)estimates $\tilde{\mathbf{C}}_X, \tilde{\mathbf{C}}_Y \in \mathbb{R}^{k \times k}$

Initialize $\tilde{\mathbf{U}}_x, \tilde{\mathbf{U}}_Y, \tilde{\mathbf{C}}_X, \tilde{\mathbf{C}}_Y$ to unit matrices
Initialize $\tilde{\mathbf{u}}_X, \tilde{\mathbf{u}}_Y, \tilde{\lambda}_X, \tilde{\lambda}_Y$ to unit matrices
for each arriving pair $x_{t+w}, y_{t+w}$ do
  $\mathbf{x}_{t,w} := [x_t \cdots x_{t+w}]^\mathsf{T}$
  $\mathbf{y}_{t,w} := [y_t \cdots y_{t+w}]^\mathsf{T}$
  $\tilde{\mathbf{U}}_X, \tilde{\mathbf{C}}_X \leftarrow$ EIGENUPDATE$(\tilde{\mathbf{U}}_X, \tilde{\mathbf{C}}_X, \mathbf{x}_{t,w}, \beta)$
  $\tilde{\mathbf{U}}_Y, \tilde{\mathbf{C}}_X \leftarrow$ EIGENUPDATE$(\tilde{\mathbf{U}}_Y, \tilde{\mathbf{C}}_Y, \mathbf{y}_{t,w}, \beta)$
  $\tilde{\mathbf{u}}_X, \tilde{\lambda}_X \leftarrow$ EIGENUPDATE$(\tilde{\mathbf{u}}_X, \tilde{\lambda}_X, \mathbf{x}_{t,w}, \beta)$
  $\tilde{\mathbf{u}}_Y, \tilde{\lambda}_Y \leftarrow$ EIGENUPDATE$(\tilde{\mathbf{u}}_Y, \tilde{\lambda}_Y, \mathbf{y}_{t,w}, \beta)$
  $\ell_t := \frac{1}{2}\big(\| \operatorname{orth}(\tilde{\mathbf{U}}_X)^\mathsf{T} \tilde{\mathbf{u}}_Y \| + \| \operatorname{orth}(\tilde{\mathbf{U}}_Y)^\mathsf{T} \tilde{\mathbf{u}}_X \|\big)$
end for
```

**Local Pearson score.** Updating the Pearson score $\rho_t$ requires an update of the inner product and norms. For the former, this can be done using the simple relationship $\mathbf{x}_{t,w}^\mathsf{T} \mathbf{y}_{t,w} = \mathbf{x}_{t-1,w}^\mathsf{T} \mathbf{y}_{t-1,w} - x_{t-1} y_{t-1} + x_{t+w-1} y_{t+w-1}$. Similar simple relationships hold for $\|\mathbf{x}_{t,w}\|$ and $\|\mathbf{y}_{t,w}\|$.

## 4.4 Complexity

The time and space complexity of each method is summarized in Table 2. Updating $\rho_t$ which requires $O(1)$ time (adding $x_{t+w-1} y_{t+w-1}$ and subtracting $x_{t-1} y_{t-1}$) and also buffering $w$ values. Estimating the LoCo score $\ell_t$ using a



(a) `MemCPU1`

(b) `MemCPU2`

**Figure 5. Local correlation scores, machine cluster.**

sliding window requires $O(wmk) = O(w^2k)$ time (since we set $m = w$) to compute the largest $k$ eigenvectors of the covariance matrix for $m$ windows of size $w$. We also need $O(wk)$ space for these $k$ eigenvectors and $O(w + m)$ space for the series values, for a total of $O(wk + m) = O(wk)$. Using an exponential window still requires storing the $w \times k$ matrix $\mathbf{V}$, so the space is again $O(wk)$. However, the eigenspace estimate $\mathbf{V}$ can be updated in $O(wk)$ time (the most expensive operation in EIGENUPDATE is $\mathbf{V}_{t-1}^{\mathsf{T}}\mathbf{x}_{t,w}$), instead of $O(w^2k)$ for sliding window.

| Method | Time<br>(per point) | Space<br>(total) |
|---|---|---|
| Pearson | $O(1)$ | $O(w)$ |
| LoCo sliding | $O(wmk)$ | $O(wk + m)$ |
| LoCo exponential | $O(wk)$ | $O(wk)$ |

**Table 2. Time and space complexity.**

## 5 Experimental evaluation

This section presents our experimental evaluation, with the following main goals:

1. Illustration of LoCo on real time series.
2. Comparison to local Pearson.
3. Demonstration of LoCo's robustness.
4. Comparison of exponential and sliding windows for LoCo score estimation.
5. Evaluation of LoCo's efficiency in a streaming setting.

**Datasets.** The first two datasets, `MemCPU1` and `MemCPU2` were collected from a set of Linux machines. They measure total free memory and idle CPU percentages, at 16 second intervals. Each pair comes from different machines, running different applications, but the series within each pair are from the same machine. The last dataset, `ExRates`, was obtained from the UCR TSDMA [13]. and consists of daily foreign currency exchange rates, measured on working days (5 measurements per week) for a total period of about 10 years. Although the order is irrelevant for the scores since they are symmetric, the first series is always in blue and the second in red. For LoCo with sliding window we use exact, batch SVD on the sample set of windows—we do not explicitly construct $\hat{\Gamma}_t$. For exponential window LoCo, we use the incremental eigenspace tracking procedure. The raw scores are shown, without any smoothing, scaling or post-processing of any kind.

**1. Qualitative interpretation.** We should first point out that, although each score has one value per time instant $t \in \mathbb{N}$, these values should be interpreted as the similarity of a "neighborhood" or window around $t$ (Figures 5 and 6). All scores are plotted so that each neighborhood is centered around $t$. The window size for `MemCPU1` and `MemCPU2` is $w = 11$ (about 3 minutes) and for `ExRates` it is $w = 20$ (4 weeks). Next, we discuss the LoCo scores for each dataset.

**Machine data.** Figure 5a shows the first set of machine measurements, `MemCPU1`. At time $t \approx 20$–$50$ one series fluctuates (oscillatory patterns for CPU), while the other remains constant after a sharp linear drop (aperiodic patterns for memory). This discrepancy is captured by $\ell_t$, which gradually returns to one as both series approach constant-valued intervals. The situation at $t \approx 185$–$195$ is similar. At $t \approx 100$–$110$, both resources exhibit large changes (aperiodic trends) that are not perfectly synchronized. This is reflected by $\ell_t$, which exhibits three dips, corresponding to the first drop in CPU, followed by a jump in memory and then a jump in CPU. Toward the end of the series, both resources are fairly constant (but, at times, CPU utilization fluctuates slightly, which affects $\rho_t$). In summary, $\ell_t$ behaves well across a wide range of joint patterns.

The second set of machine measurements, `MemCPU2`, is shown in Figure 5b. Unlike `MemCPU1`, memory and CPU utilization follow each other, exhibiting a very similar periodic pattern, with a period of about 30 values or 8 minutes. This is reflected by the LoCo score, which is mostly one. However, about in the middle of each period, CPU utilization drops for about 45 seconds, without a corresponding change in memory. At precisely those instants, the LoCo score also drops (in proportion to the discrepancy), clearly indicating the break of the otherwise strong correlation.

**Exchange rate data.** Figure 6 shows the exchange rate (`ExRates`) data. The blue line is the French Franc and the red line is the Spanish Peseta. The plot is annotated with an approximate timeline of major events in the European Monetary Union (EMU). Even though one should always be very careful in suggesting any causality, it is still remarkable that most major EMU events are closely accompanied by a break in the correlation as measured by LoCo, and vice versa. Even in the cases when an accompanying break is absent, it often turns out that at those events both currencies received similar pressures (thus leading to similar trends, such as, e.g., in the October 1992 events). It is also interesting to point out that events related to anticipated regulatory changes are typically *preceded* by correlation breaks. After regulations are in effect, $\ell_t$ returns to one. Furthermore, after the second stage of the EMU, both currencies proceed in lockstep, with negligible discrepancies.

In summary, the LoCo score successfully and accurately tracks evolving local correlations, even when the series are widely different in nature.

**2. LoCo versus Pearson.** Figures 5 and 6 also show the local Pearson score (fourth row), along with the LoCo score. It is clear that it either fails to capture changes in the joint patterns among the two series, or exhibit high sensitivity to small transients. We also tried using a window size of $2w - 1$ instead of $w$ for $\rho_t$ (so as to include the same number of points as $\ell_t$ in the "comparison" of the two series). The results thus obtained where slightly different but similar, especially in terms of sensitivity and lack of accurate tracking of the evolving relationships among the series.

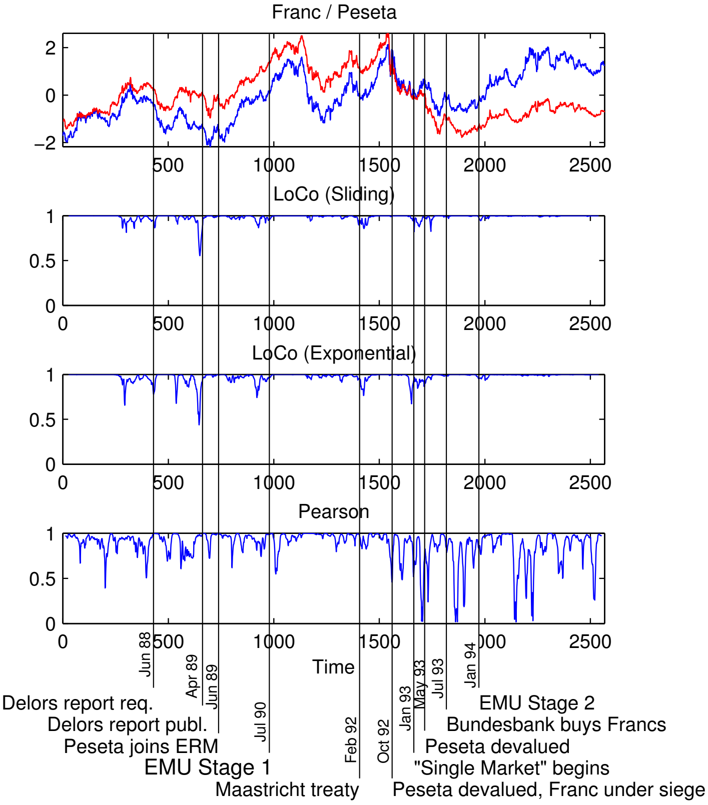
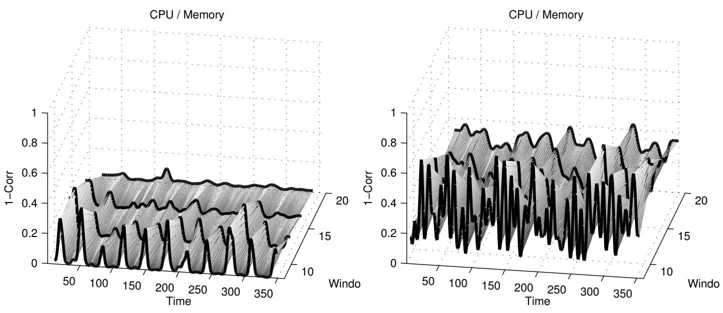


**Figure 6. Local correlation scores, `ExRates`.**

**3. Robustness.** This brings us to the next point in our discussion, the robustness of LoCo. We measure the "stability" of any score $c_t, t \in \mathbb{N}$ by its smoothness. We employ a common measure of smoothness, the (discrete) total variation $V$ of $c_t$, defined as $V(c_t) := \sum_\tau |c_{\tau+1} - c_\tau|$, which is the total "vertical length" of the score curve. Table 3 (top) shows the relative total variation, with respect to the baseline of the LoCo score, $V(\rho_t)/V(\ell_t)$. If we scale the total variations with respect to the range (i.e., use $V(c_t)/R(c_t)$ instead of just $V(c_t)$—which reflects how many times the vertical length "wraps around" its full vertical range), then Pearson's variation is consistently about 3 times larger, over all data sets.

| | Dataset | | |
|---|---|---|---|
| Method | `MemCPU1` | `MemCPU2` | `ExRates` |
| Pearson | 4.16× | 3.36× | 6.21× |
| LoCo | 5.71 | 10.53 | 6.37 |

**Table 3. Relative stability (total variation).**



(a) LoCo (b) Pearson

**Figure 7. Score vs. window size; LoCo is robust with respect to both time and scale, accurately tracking correlations at any scale, while Pearson performs poorly at all scales.**

**Window size.** Figure 7a shows the LoCo scores of `MemCPU2` (see Figure 5b) for various windows $w$, in the range of 8–20 values (2–5 minutes). We chose the dataset with the highest total score variation and, for visual clarity, Figure 7 shows $1 - \ell_t$ instead of $\ell_t$. As expected, $\ell_t$ varies smoothly with respect to $w$. Furthermore, it is worth pointing out that at about a 35-value (10-minute) resolution (or coarser), both series the exhibit clearly the same behavior (a periodic increase then decrease, with a period of about 10 minutes—see Figure 5b), hence they are perfectly correlated and their LoCo score is almost constantly one (but not their Pearson score, which gets closer to one while still fluctuating noticeably). Only at much coarser resolutions (e.g., an hour or more) do both scores become one. This convergence to one is not generally the case and some time series may exhibit interesting relationships at all time scales. However, the LoCo score is robust and changes gracefully also with respect to resolution/scale, while accurately capturing any interesting relationship changes that may be present at any scale.

| Dataset | `MemCPU1` | `MemCPU2` | `ExRates` |
|---|---|---|---|
| Avg. var. | 0.051 | 0.071 | 0.013 |
| Rel. var. | 5.6% | 7.8% | 1.6% |

**Table 4. Sliding vs. exponential score.**

**4. Exponential vs. sliding window.** Figures 5 and 6 show the LoCo scores based upon both sliding (second row) and exponential (third row) windows, computed using appropriately chosen equivalent window sizes. Upon inspection, it is clear that both LoCo score estimates are remarkably close. In order to further quantify this similarity, we show the average variation $\hat{V}$ of the two scores, which is defined as $\hat{V}(\ell_t, \ell'_t) := \frac{1}{t}\sum_{\tau=1}^{t} |\ell_\tau - \ell'_\tau|$, where $\ell_t$ uses exact, batch SVD on sliding windows and $\ell'_t$ uses eigenspace tracking on exponential windows. Table 4 shows the average score variations for each dataset, which are remarkably small, even when compared to the mean score $\hat{\ell} := \frac{1}{t}\sum_{\tau=1}^{t} \ell_\tau$ (the bottom line in the table is $\hat{V}/\hat{\ell}$).

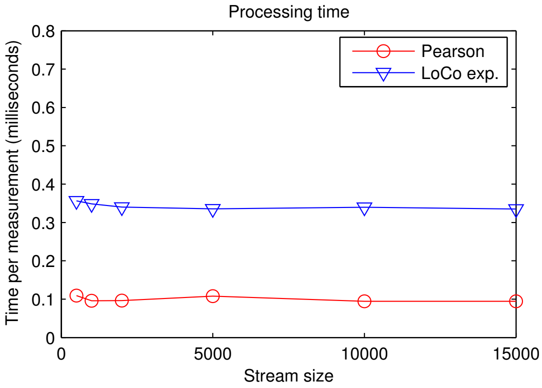


**Figure 8. Processing wall-clock time.**

**5. Performance.** Figure 8 shows wall clock times per incoming measurement for our prototype implementations in Matlab 7, running on a Pentium M 2GHz. Using $k = 4$ and $w = 10$, LoCo is in practice slightly less than 4× slower than the simplest alternative, i.e., the Pearson correlation. The additional processing time spent on updating the eigenvector estimates using an exponential window is small, while providing much more meaningful and robust scores. Finally, it is worth pointing out that, even using an interpreted language, the processing time required per pair of incoming measurements is merely 0.33 milliseconds or, equivalently, about $2 \times 3000$ values per second.

## 6 Related work

Even though, to the best of our knowledge, the problem of local correlation tracking has not been explicitly addressed, time series and streams have received much attention and more broadly related previous work addresses other aspects of either "global" similarity among a collection of streams (e.g., [5]) or mining on time evolving streams (e.g., CluStream [1], StreamCube [8], and [2]). Change detection in discrete-valued streams has also been addressed [10, 23].

BRAID [18] addresses the problem of finding *lag correlations* on streams, i.e., of finding the first local maximum of the *global* cross-correlation (Pearson) coefficient with respect to an arbitrary lag. StatStream [24] addresses the problem of efficiently finding the largest cross-correlation coefficients (at zero lag) among all pairs from a collection of time series streams. EDS [12] address the problem of separating out the noise from the covariance matrix of a stream collection (or, equivalently, a multidimensional stream), but does not explicitly consider trends across time. Quantized representations have also been employed for dimensionality reduction, indexing and similarity search on static time series, such as the Multiresolution Vector Quantized (MVQ) representation [15], and the Symbolic Aggregate approXimation (SAX) [14, 17].

The work in [20] addresses the problem of finding specifically burst correlations, by preprocessing the time series to extract a list of burst intervals, which are subsequently indexed using an interval tree. This is used to find all intersections of bursty intervals of a given query time series versus another collection of time series. The work in [21] proposes a similarity metric for time series that is based on comparison of the Fourier coefficient magnitudes, but allows for phase shifts in each frequency independently.

In the field of signal processing, the eigen-decomposition of the autocovariance matrix is employed in the widely used *MUSIC (MUltiple SIgnal Classification)* algorithm for spectrum estimation [19], as well as in *Singular Spectrum Analysis (SSA)* [6, 7]. Applications and extensions of SSA have recently appeared in the field of data mining. The work in [9] employs similar ideas but for a different problem. In particular, it estimates a changepoint score which can subsequently be used to visualize relationships with respect to changepoints via multi-dimensional scaling (MDS). Finally, the work in [16] proposes a way to efficiently estimate a family of optimal orthonormal transforms for a single series at multiple scales (similar to wavelets). These transforms can capture arbitrary periodic patterns that may be present.

## 7 Conclusion

Time series correlation or similarity scores are useful in several applications. Beyond global scores, in the context of time-evolving time series it is desirable to track a time-evolving correlation score that captures their changing similarity. We propose such a measure, the Local Correlation (LoCo) score. It is based on a joint model of the series which, naturally, does not make any assumptions about stationarity. The model may be viewed as a generalization of simple linear cross-correlation (which it includes as a special case), as well as of traditional frequency analysis [7, 6, 19]. The score is robust to transients, while accurately tracking the time-varying relationships among the series. Furthermore, it lends itself to efficient estimation in a streaming setting. We demonstrate its qualitative interpretation on real datasets, as well as its robustness and efficiency.

## References

[1] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. In *VLDB*, 2003.

[2] E. Bingham, A. Gionis, N. Haiminen, H. Hiisilä, H. Mannila, and E. Terzi. Segmentation and dimensionality reduction. In *SDM*, 2006.

[3] Y. Cai and R. Ng. Indexing spatio-temporal trajectories with Chebyshev polynomials. In *SIGMOD*, 2004.

[4] P. Celka and P. Colditz. A computer-aided detection of eeg seizures in infants: A singular-spectrum approach and performance comparison. *IEEE TBE*, 49(5), 2002.

[5] G. Cormode, M. Datar, P. Indyk, and S. Muthukrishnan. Comparing data streams using Hamming norms (how to zero in). In *VLDB*, 2002.

[6] M. Ghil, M. Allen, M. Dettinger, K. Ide, D. Kondrashov, M. Mann, A. Robertson, A. Saunders, Y. Tian, F. Varadi, and P. Yiou. Advanced spectral methods for climatic time series. *Rev. Geophys.*, 40(1), 2002.

[7] N. Golyandina, V. Nekrutkin, and A. Zhigljavsky. *Analysis of Time Series Structure: SSA and Related Techniques*. CRC Press, 2001.

[8] J. Han, Y. Chen, G. Dong, J. Pei, B. W. Wah, J. Wang, and Y. D. Cai. StreamCube: An architecture for multi-dimensional analysis of data streams. *Dist. Par. Databases*, 18(2):173–197, 2005.

[9] T. Idé and K. Inoue. Knowledge discovery from heterogeneous dynamic systems using change-point correlations. In *SDM*, 2005.

[10] D. C. in Data Streams. Daniel kifer and shai ben-david and johannes gehrke. In *VLDB*, 2004.

[11] I. T. Jolliffe. *Principal Component Analysis*. Springer, 2nd edition, 2002.

[12] H. Kargupta, K. Sivakumar, and S. Ghosh. Dependency detection in MobiMine and random matrices. In *PKDD*, 2002.

[13] E. Keogh and T. Folias. Ucr time series data mining archive. `http://www.cs.ucr.edu/~eamonn/TSDMA/`.

[14] J. Lin, E. J. Keogh, S. Lonardi, and B. Y.-C. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *DMKD*, 2003.

[15] V. Megalooikonomou, Q. Wang, G. Li, and C. Faloutsos. A multiresolution symbolic representation of time series. In *ICDE*, 2005.

[16] S. Papadimitriou and P. S. Yu. Optimal multi-scale patterns in time series streams. In *SIGMOD*, 2006.

[17] C. A. Ratanamahatana, E. J. Keogh, A. J. Bagnall, and S. Lonardi. A novel bit level time series representation with implication of similarity search and clustering. In *PAKDD*, 2005.

[18] Y. Sakurai, S. Papadimitriou, and C. Faloutsos. BRAID: Stream mining through group lag correlations. In *SIGMOD*, 2005.

[19] R. O. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Trans. Ant. Prop.*, 34(3), 1986.

[20] M. Vlachos, K.-L. Wu, S.-K. Chen, and P. S. Yu. Fast burst correlation of financial data. In *PKDD*, 2005.

[21] M. Vlachos, P. S. Yu, and V. Castelli. On periodicity detection and structural periodic similarity. In *SDM*, 2005.

[22] B. Yang. Projection approximation subspace tracking. *IEEE Trans. Sig. Proc.*, 43(1), 1995.

[23] J. Yang and W. Wang. AGILE: A general approach to detect transitions in evolving data streams. In *ICDM*, 2004.

[24] Y. Zhu and D. Shasha. StatStream: Statistical monitoring of thousands of data streams in real time. In *VLDB*, 2002.