Received ; | Revised ; | Accepted

DOI: xxx/xxxx

RESEARCH ARTICLE

Biclustering high-frequency financial time series based on information theory

Haitao Liu[1] | Jian Zou*[1,2] | Nalini Ravishanker[3]

[1]Data Science Program, Worcester Polytechnic Institute, MA, USA
[2]Department of Mathematical Sciences, Worcester Polytechnic Institute, MA, USA
[3]Department of Statistics, University of Connecticut, CT, USA

Correspondence
*Jian Zou, Department of Mathematical Sciences, Worcester Polytechnic Institute, MA, USA. Email: jzou@wpi.edu

Abstract

Clustering a large number of time series into relatively homogeneous groups is a well-studied unsupervised learning technique that has been widely used for grouping financial instruments (say, stocks) based on their stochastic properties across the entire time period under consideration. However, clustering algorithms ignore the notion of biclustering, i.e., grouping of stocks only within a subset of times rather than over the entire time period. Biclustering algorithms enable grouping of stocks and times simultaneously, and thus facilitate improved pattern extraction for informed trading strategies. While biclustering methods may be employed for grouping low-frequency (daily) financial data, their use with high-frequency financial time series of intra-day trading data is especially useful. This paper develops a biclustering algorithm based on pairwise or groupwise mutual information between one-minute averaged stock returns within a trading day, using jackknife estimation of mutual information (JMI). We construct a multiple day time series biclustering (MI-MDTSB) algorithm that can capture refined and local comovement patterns between groups of stocks over a subset of continuous time points. Extensive numerical studies based on high-frequency returns data reveal interesting intra-day patterns among different asset groups and sectors.

KEYWORDS:
biclustering, financial time series, high-frequency stock returns, mutual information

1 | INTRODUCTION

As computing power grows and massive storage technology develops, multiple high-frequency time series are available at an unprecedented rate in several domains. Such multi-dimensional, high-frequency data are usually large and noisy, and detecting patterns of underlying structure becomes a challenging task. In financial data analysis, analysts seek to detect patterns in daily or intra-day asset returns that exist over intervals of time. For example, *pair trading* is a statistical arbitrage strategy for forming a portfolio of two related stocks whose relative pricing is distant from an "equilibrium" market state [1,2,3]. The pattern detection usually relies on the correlation in the historical prices of the two stocks and various clustering algorithms enable grouping of the stocks [4].

Suppose the different stocks constitute the rows of a data matrix $\mathbf{S}$, whose columns are the times within a day. A clustering algorithm would group the rows (stocks) of $\mathbf{S}$ across all columns (times) by using a suitable distance metric. These algorithms may fail to extract coherent *local patterns* in stock prices that may manifest across only a certain subset of columns (time points). Biclustering approaches [5] address this issue by clustering the rows and columns of a data matrix simultaneously, and are

particularly attractive for large sets of long time series that occur in many domains including finance, bio-sciences, etc. Patterns obtained from biclustering often give valuable information in addition to (global across all times) clustering.

Biclustering algorithms can be broadly classified into four types: greedy, divide-and-conquer, exhaustive enumeration, and distribution parameter identification[6]. *Greedy approaches*[7,8,9,10,11,12] usually make the best local decision at each iteration, hoping that it leads to a globally optimal solution. Algorithms based on *divide-and-conquer* approaches[13] break a problem into smaller sub-problems and solve these sub-problems recursively, the solution of the original problem being a combination of the sub-problem solutions. *Exhaustive enumeration* algorithms[14,15] assume that the best submatrices can only be identified by generating all possible row and column combinations of the dataset. *Distribution parameter identification* algorithms[16,17,18] assume a statistical model suitable for the structure of the biclusters, and then apply some iterative procedure to adapt parameters of the assumed model.

We describe a biclustering algorithm that is based on a popular greedy biclustering approach proposed by Cheng and Church[7], which alternately eliminates rows (genes/stocks) and columns (time points) of **S** to minimize a mean squared residue score. Similar algorithms have been developed subsequent to their work. For example, the Conserved Gene Expression Motifs (xMOTIFs)[19] algorithm finds submatrices with simultaneously conserve genes in subsets of experimental conditions in a discrete data matrix. The Iterative Signature Algorithm (ISA)[9] starts with randomly selected genes and experimental conditions and then evaluates and updates them through iterative steps until convergence. Minimum Sum-Squared Residue Coclustering (MSSRCC)[10,20] simultaneously clusters the genes and experimental conditions of a dataset to find checkerboard patterns. QUalitative BIClustering (QUBIC)[11] discretizes the input dataset and builds a graph to search for biclusters corresponding to heavy subgraphs.

However, none of these algorithms preserve contiguity of the time points. In a financial setting, it is important to find biclusters across contiguous columns/times, corresponding to comovement patterns shared by a group of stocks over consecutive time points. Madeira et al.[21] proposed a biclustering algorithm that finds and reports all maximal contiguous column coherent biclusters (CCC-Biclusters) by processing a discretized version of the original (gene expression in their paper) matrix via efficient string processing techniques based on suffix trees; the computing time is linear in the size of the expression matrix. The CC-TSB algorithm of Zhang et al.[22] is an extension of the algorithm by Cheng and Church[7], and only allows the elimination of the start and end points in a time sequence, thus preserving contiguity. A resulting bicluster will therefore consist of a *continuous* interval of times, and not be the union of several disjoint (perhaps even singleton) times.

One limitation of these approaches is that they have only been applied on a single *static* data matrix with contiguous time intervals. However, in multi-dimensional high-frequency settings, the data can span many non-consecutive subintervals isolated over T time periods. The stock market provides one such example. Stocks are regularly traded from 9:30 am to 4:00 pm on *each* trading day, leading to multi-dimensional time series of asset prices daily. Simple concatenation of these (intrinsically non-contiguous) intra-day data matrices over multiple trading days ignores the overnight market dynamics and can possibly result in suboptimal biclusters. To address the issue in a high-frequency setting, Liu et al.[23] proposed a multiple day time series biclustering algorithm based on mean residue scores (CC-MDTSB) that extends the CC-TSB method[22] by imposing a threshold on the number of stocks within the selected biclusters in order to yield an adaptive stopping criterion for the multiple-day analysis.

A second issue is that the aforementioned biclustering algorithms are based on linear measures. Often, measuring the strength of a nonlinear relationship based on *information theory* can provide value in many applications. One such useful nonlinear measure is mutual information (MI), which is a special case of the class of distortion functions known as Bregman divergences[24] and is related to the joint entropy of two random variables and measures the dependence between them[25]. It is also closely related to cross-entropy and Kullback–Leibler divergence that are widely used in data science and machine learning. The literature on biclustering based on information theory is sparse. An algorithm based on information was studied in the context of word-document clustering[26], and a weighted MI based biclustering algorithm was developed for gene expression data[27]. However, these algorithms are not applicable to time series data, and to multiple day financial time series, in particular. There is a need to develop MI based biclustering methods using moving time windows on transaction-level data within a day, and to dynamically study patterns of comovement over multiple trading days. To this end, we propose a novel MI based multiple day time series biclustering algorithm (MI-MDTSB) to extract local comovement patterns within a trading day. We then link the bilcusters over the span of multiple trading days and explore periodic patterns of comovements.

In this paper, we focus on biclustering pre-averaged log returns[28]. The log returns are also referred to as continuously compounded returns, being closely related to the concept of compound interest rates. They have several desirable statistical properties including additivity which helps alleviate the aggregation of microstructure noise in the asset price data[29]. We develop a biclustering algorithm based on pairwise or groupwise mutual information (MI) between one-minute averaged stock returns within a

trading day, and construct a multiple day time series biclustering (MI-MDTSB) algorithm that can capture local comovement patterns between groups of stocks over a subset of continuous time points. This algorithm is based on the jackknife estimate of the mutual information (JMI) with less bias and improved efficiency over existing methods, and also preserves the contiguity of the time points, as we require.

The rest of the paper is organized as follows. Section 2 describe the high-frequency stock returns data. Section 3 provides a review of mutual information and its estimation. Section 4 describes the MI-based multiple day time series biclustering algorithm (MI-MDTSB). Section 5 estimates the comovement probability of m-tuples of stocks in biclusters generated by the MI-MDTSB algorithm. Section 6 discusses an empirical approach for approximating periodicity in comovement patterns within pairs or groups of selected stocks. Section 7 summarizes the paper and our contributions.

2 | DATA DESCRIPTION OF HIGH-FREQUENCY FINANCIAL TIME SERIES

High-frequency financial data refer to time-series data of traded financial assets collected at an extremely fine scale, usually at the tick-by-tick level. High-frequency data are widely used in financial analysis as a result of advanced computational power in recent years. Such intraday observations from high-frequency data are used to study market behaviors and micro-structures[30].

Studying high frequency data can be a challenging task due to diurnal effects[31,32] and micro-structure noise[30]. The intra-day stock market presents a diurnal pattern, which is caused by periodic trading patterns, the duration between trades being smallest at the open and close of the market due to the information transmitted overnight and traders' activities of re-balancing their positions at the closing time[23]. Microstructure noise is present in high-frequency data due to the imperfections of the trading process[33]. In the literature, researchers usually assume that the observed price P_t in high-frequency financial time series can be decomposed into the fundamental price X_t and market microstructure noise ϵ_t[34], i.e., $P_t = X_t + \epsilon_t$. Microstructure noise ϵ_t comes from various market frictions including bid-ask bounces, data recording errors, discreteness of price changes and rounding, and trades occurring on different markets or network, etc . By suitable statistical analysis of the observed price P_t, one can focus the attention on the fundamental price X_t[30,35,36]. Pre-averaging the observed data within selected time intervals (one-minute or five-minute, say) is an appealing way to reduce the effect of the noise, and can be regarded as a form of data cleaning[28].

In this paper, we use the S&P 100 data downloaded from the Trade and Quote (TAQ) database of Wharton Research Data Services (WRDS). The raw data are very large and noisy, so data preprocessing is necessary. We remove the transactions with the condition of special sales with any of the indicators including O, Z, B, T, L, G, W, J and K, since these special trading data are outside the scope of our analysis; any transactions with conditions other than 0, 1, or 2 are removed. We only keep transactions with positive price and volume. The pre-processing of the data is similar to Liu et al[23].

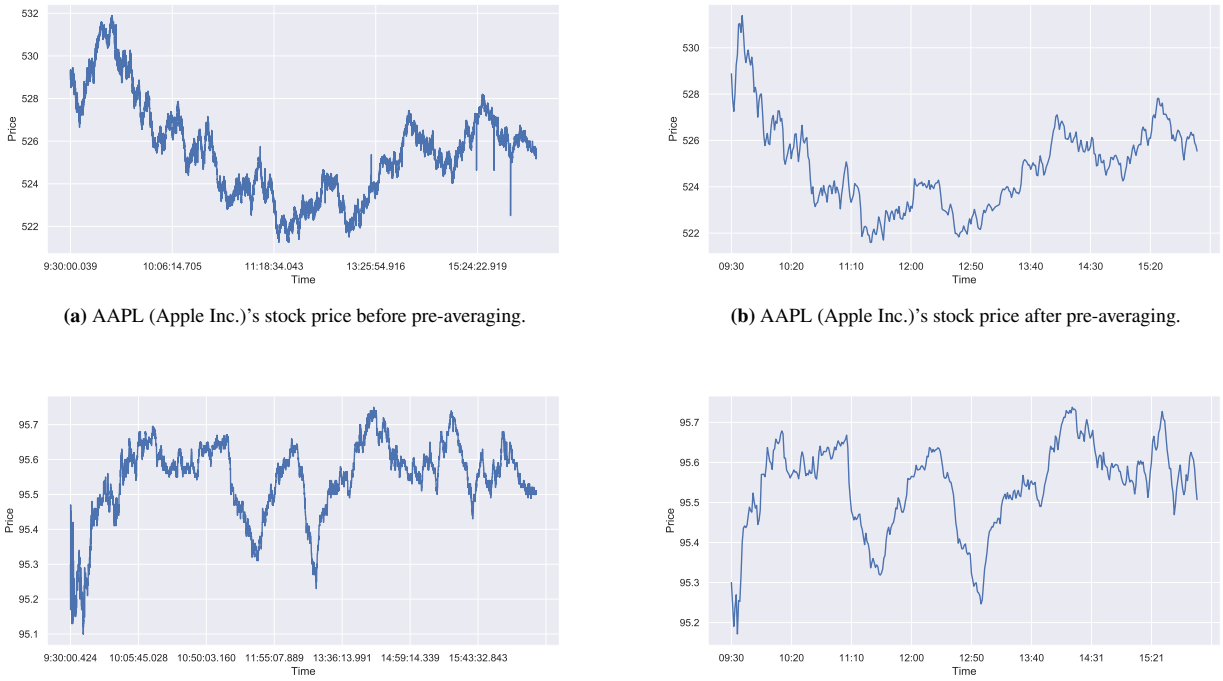
Table 1 shows the original raw stock prices at the transaction-by-transaction level. The trading prices are recorded from 9:30.00.001 am to 4:00.00.000 on a regular trading day. There are over a billion such prices for all the stocks in S&P 100 in a single year. For each trading day and each stock, we have one time series of prices. Figure 1 (a) and Figure 1 (c) show the intra-day stock price time series of AAPL (Apple Inc.) and MMM (3M Co) respectively. Such raw stock prices at the transaction level are very noisy. We pre-average the prices within one-minute intervals in order to address the microstructure noise ϵ_t. Figure 1 (b) and Figure 1 (d) show that the pre-averaged time series are smoother than the corresponding raw series. We then compute log-returns from the pre-averaged prices in the usual way, i.e., $r_t = (1 - B)\log P_t$, where B is the backshift operator. The data set consists of prices on 94 stocks from the S&P 100 index from 9:30 am to 4:00 pm for 249 trading days in 2013. We pre-average the stock prices to one-minute intervals within a day and then analyze the logarithms of the returns constructed from the one-minute averaged stock prices. Table 2 shows an example of data after pre-processing on January 2, 2013.

3 | MUTUAL INFORMATION AND ITS ESTIMATION BETWEEN STOCKS

The mutual information (MI) between two random variables is a generalization of the correlation between them and measures the dependence between the two variables[25]. Let $p(x)$ denote the probability mass function (pmf) of a discrete random variable X, let $f(x)$ be the probability density function (pdf) of a continuous random variable, and let $p(x, y)$ and $f(x, y)$ denote the joint pmf and pdf respectively of two random variables X and Y.

TABLE 1 Transaction by transaction level data of Citigroup Inc.

Date	Time	Stock	Price
20130102	9:30:00.005	C	40.91
20130102	9:30:00.005	C	40.91
20130102	9:30:00.007	C	40.91
20130102	9:30:00.007	C	40.91
20130102	9:30:00.013	C	40.94
...	...	...	...
20131231	15:59:59.433	C	52.09
20131231	15:59:59.802	C	52.10
20131231	15:59:59.804	C	52.10
20131231	15:59:59.812	C	52.10



(a) AAPL (Apple Inc.)'s stock price before pre-averaging.

(b) AAPL (Apple Inc.)'s stock price after pre-averaging.

(c) MMM (3M Co)'s stock price before pre-averaging.

(d) MMM (3M Co)'s stock price after pre-averaging.

FIGURE 1 Intra-day stock prices on January 8, 2013 for AAPL (top panel) and MMM (bottom panel).

The MI between X and Y measures the information shared by the two random variables and is defined as

$$\mathrm{MI}(X,Y) = = \begin{cases} \sum_x \sum_y p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right), & \text{discrete case} \\ \int\int f(x,y) \log\left(\frac{f(x,y)}{f(x)f(y)}\right) dxdy, & \text{continuous case.} \end{cases} \tag{1}$$

Consider two groups of random variables, $\mathbf{X} = (X_1, X_2, \cdots, X_P)'$ and $\mathbf{Y} = (Y_1, Y_2, \cdots, Y_Q)'$. The estimated MI is the largest value among these kernel estimates and its computation is summarized in the following steps described by Zeng et al[37].

$$\mathrm{MI}(\mathbf{X},\mathbf{Y}) = E\left\{\log\frac{f_{\mathbf{XY}}(\mathbf{X},\mathbf{Y})}{f_{\mathbf{X}}(\mathbf{X})f_{\mathbf{Y}}(\mathbf{Y})}\right\}. \tag{2}$$

Note that this definition is equivalent to the definition in equation (1), except that here, $\mathbf{X}$ and $\mathbf{Y}$ are respectively vectors of length P and Q. Since $\mathrm{MI}(X,Y)$ is a special case of $\mathrm{MI}(\mathbf{X},\mathbf{Y})$ when $P = 1$ and $Q = 1$, we use the notation $\mathrm{MI}(\mathbf{X},\mathbf{Y})$ below.

TABLE 2 Log returns on January 2, 2013

time	9:31	9:32	9:33	⋯	15:57	15:58	15:59
AAPL	-0.003232	-0.001185	0.000743	⋯	-0.000330	0.000168	-0.000766
ABBV	0.003227	-0.006762	-0.000002	⋯	0.002074	0.001167	0.001480
⋯	⋯	⋯	⋯	⋯	⋯	⋯	⋯
GOOG	0.005539	-0.001263	-0.000009	⋯	0.000237	0.000154	0.000462
WAG	0.001651	0.001112	0.004433	⋯	-0.000182	0.000260	0.000773

A large value of $\mathrm{MI}(\mathbf{X},\mathbf{Y})$ indicates that the two random variables are highly associated, while a low-value indicates low association. If and only if $\mathrm{MI}(\mathbf{X},\mathbf{Y})=0$, the two variables are independent. While the MI defined in equations (1) and (2) is not a distance metric, it can be converted to a distance metric and related to entropy. The joint entropy of two random variables $\mathbf{X}$ and $\mathbf{Y}$ is

$$H(X,Y)=\begin{cases}-\sum_x\sum_y p(x,y)\log\left(p(x,y)\right) & \text{if } \mathbf{X} \text{ and } \mathbf{Y} \text{ are discrete,}\\ -\int\int f(x,y)\log\left(f(x,y)\right)dxdy & \text{if } \mathbf{X} \text{ and } \mathbf{Y} \text{ are continuous.}\end{cases} \tag{3}$$

The mutual information between $\mathbf{X}$ and $\mathbf{Y}$ can be equivalently expressed in the terms of entropy[38],

$$MI(\mathbf{X},\mathbf{Y})=H(\mathbf{X})-H(\mathbf{X}|\mathbf{Y}). \tag{4}$$

Then,

$$MI(\mathbf{X},\mathbf{X})=H(\mathbf{X})-H(\mathbf{X}|\mathbf{X}). \tag{5}$$

Since the conditional entropy $H(\mathbf{X}|\mathbf{X})=0$, $\mathrm{MI}(\mathbf{X},\mathbf{X})=H(\mathbf{X})$, similarly $\mathrm{MI}(\mathbf{Y},\mathbf{Y})=H(\mathbf{Y})$.

The joint entropy of random variable $\mathbf{X}$ and $\mathbf{Y}$ is

$$\begin{aligned}H(\mathbf{X},\mathbf{Y})&=H(\mathbf{X})+H(\mathbf{Y})-\mathrm{MI}(\mathbf{X},\mathbf{Y})\\&=\mathrm{MI}(\mathbf{X},\mathbf{X})+\mathrm{MI}(\mathbf{Y},\mathbf{Y})-\mathrm{MI}(\mathbf{X},\mathbf{Y}).\end{aligned}$$

The MI based distance is defined as

$$d(\mathbf{X},\mathbf{Y})=H(\mathbf{X},\mathbf{Y})-\mathrm{MI}(\mathbf{X},\mathbf{Y}) \tag{6}$$

$$=\mathrm{MI}(\mathbf{X},\mathbf{X})+\mathrm{MI}(\mathbf{Y},\mathbf{Y})-2\mathrm{MI}(\mathbf{X},\mathbf{Y}), \tag{7}$$

which satisfies the non-negativity, symmetry and triangle inequality properties.

The normalized distance is then defined as

$$D(\mathbf{X},\mathbf{Y})=1-\frac{\mathrm{MI}(\mathbf{X},\mathbf{Y})}{H(\mathbf{X},\mathbf{Y})} \tag{8}$$

$$=1-\frac{\mathrm{MI}(\mathbf{X},\mathbf{Y})}{\mathrm{MI}(\mathbf{X},\mathbf{X})+\mathrm{MI}(\mathbf{Y},\mathbf{Y})-\mathrm{MI}(\mathbf{X},\mathbf{Y})}. \tag{9}$$

so that $0\le D(\mathbf{X},\mathbf{Y})\le 1$.

We compute the MI between stock returns similar to Liu et al.[39], by employing a jackknife version of the kernel estimate with equalized bandwidth varying over an interval[37]. The estimated MI is the largest value among these kernel estimates and its computation is summarized in the following steps.

JMI Step 1: Consider two random vectors $\mathbf{X}=(X_1,X_2,\cdots,X_P)'$ and $\mathbf{Y}=(Y_1,Y_2,\cdots,Y_Q)'$. Let $C=(\mathbf{x}_t,\mathbf{y}_t), t=1,\cdots,T$ with $\mathbf{x}_t=(x_{t1},x_{t2},\cdots,x_{tP})'$ and $\mathbf{y}_t=(y_{t1},y_{t2},\cdots,y_{tQ})'$ denoting independent samples from $(\mathbf{X},\mathbf{Y})$.

JMI Step 2: Consider the uniform transformation $\mathbf{U}=(U_1,U_2,\cdots,U_P)'=(F_{X_1}(X_1),F_{X_2}(X_2),\cdots,F_{X_P}(X_P))'$ and $\mathbf{V}=(V_1,V_2,\cdots,V_Q)'=\left(F_{Y_1}(Y_1),F_{Y_2}(Y_2),\cdots,F_{Y_Q}(Y_Q)\right)'$, where $F_{X_p}(.)$ and $F_{Y_q}(.)$, are the cumulative distribution functions of X_p and Y_q respectively for $p=1,2,\cdots,P$ and $q=1,2,\cdots,Q$. The MI between $\mathbf{X}$ and $\mathbf{Y}$ is invariant to such transformations, i.e., $\mathrm{MI}(\mathbf{U},\mathbf{V})=\mathrm{MI}(\mathbf{X},\mathbf{Y})$.

JMI Step 3: Let $c_{\mathbf{U}}(\mathbf{u})$, $c_{\mathbf{V}}(\mathbf{v})$ and $c_{\mathbf{UV}}(\mathbf{u},\mathbf{v})$ denote the copula density functions of $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{UV}$ respectively. The corresponding transformation of the observed data is $(\mathbf{u}_\mathbf{t}^\star,\mathbf{v}_\mathbf{t}^\star)'=(u_{t1}^\star,\cdots,u_{tP}^\star,v_{t1}^\star,\cdots,v_{tQ}^\star)'=\left(F_{X_1,T}(x_1),\cdots,F_{X_P,T}(x_P),F_{Y_1,T}(y_1),\cdots,F_{Y_Q,T}(y_Q)\right)'$ with $F_{X_P,T}, p=1,2,\cdots,P$ and $F_{Y_q,T}, q=1,2,\cdots,Q$ denoting the empirical cumulative distribution functions of $\{x_{tp},t=1,2,\ldots,T\}$ and $\{y_{tq},t=1,2,\ldots,T\}$, respectively.

JMI Step 4: Let $\mathbf{K}^P$ denote a P-dimensional symmetric density function with $\mathbf{K}^P(\mathbf{x}) = \Pi_{p=1}^{P} K(x_p)$. For a diagonal matrix $\mathbf{A}$, ${\mathbf{K}_\mathbf{A}}^P(\mathbf{x}) = |\mathbf{A}|^{-1/2}\mathbf{K}^P(\mathbf{A}^{-1/2}\mathbf{x})$. The MI estimator with one common bandwidth h is

$$\hat{I}(h) = \frac{1}{T}\sum_{t=1}^{T} \log \frac{\hat{c}^{\setminus t}_{\mathbf{UV},h^2\mathbf{I}_P,h^2\mathbf{I}_Q}(\mathbf{u}^\star{}_t, \mathbf{v}^\star{}_t)}{\hat{c}^{\setminus t}_{\mathbf{U},h^2\mathbf{I}_P}(\mathbf{u}^\star{}_t)\hat{c}^{\setminus t}_{\mathbf{V},h^2\mathbf{I}_Q}(\mathbf{v}^\star{}_t)}, \tag{10}$$

where $\mathbf{I}_P$ and $\mathbf{I}_Q$ are $P \times P$ and $Q \times Q$ identity matrices, and

$$\hat{c}^{\setminus t}_{\mathbf{U},h^2\mathbf{I}_P}(\mathbf{u}) = \frac{1}{T-1}\sum_{t'\neq t}^{T} \mathbf{K}^P_{h^2\mathbf{I}_P}(\mathbf{u}^\star{}_{t'} - \mathbf{u});$$

$$\hat{c}^{\setminus t}_{\mathbf{V},h^2\mathbf{I}_Q}(\mathbf{v}) = \frac{1}{T-1}\sum_{t'\neq t}^{T} \mathbf{K}^Q_{h^2\mathbf{I}_Q}(\mathbf{v}^\star{}_{t'} - \mathbf{v});$$

$$\hat{c}^{\setminus t}_{\mathbf{UV},h^2\mathbf{I}_P,h^2\mathbf{I}_Q}(\mathbf{u},\mathbf{v}) = \frac{1}{T-1}\sum_{t'\neq t}^{n} \mathbf{K}^P_{h^2\mathbf{I}_P}(\mathbf{u}^\star{}_{t'} - \mathbf{u})\mathbf{K}^Q_{h^2\mathbf{I}_Q}(\mathbf{v}^\star{}_{t'} - \mathbf{v}).$$

JMI Step 5: Estimate equation (10) repeatedly by leaving out each observation in $\mathbf{X}$ and $\mathbf{Y}$. The Jackknife estimator of MI is

$$\widehat{\text{JMI}}\,(\mathbf{X},\mathbf{Y}) = \max_{h>0}[\hat{I}(h)]. \tag{11}$$

These formulas apply directly to pairwise MI between two stocks when each group has one stock, i.e., $P = Q = 1$.

Let $\mathcal{S} = \{1, \ldots, S\}$ denote a set of S stocks that are observed over T time intervals (say, one-minute intervals) over N trading days. On the nth trading day (for $n = 1, \cdots, N$), we construct (pre-averaged) log returns $r_{i,t,n}$ in the tth time interval for the ith stock, for $t = 1, \ldots, T$ and $i = 1, \ldots, S$. We then estimate the MI to measure the similarity between the ith stock and the i'th stock within trading day n as $\widehat{\text{JMI}}\,(s_i, s_{i'})$ using (11), with $\mathbf{X} = \{r_{i,t,n}\}$ and $\mathbf{Y} = \{r_{i',t,n}\}$ as inputs. We use the R package *JMI*[40] with default settings for the choice of bandwidth h, using grid search to solve the maximization problem.[37] Let

$$\mathcal{H}_T = \{c_0 < h_1 < h_2 < \cdots < h_m < c_1\}, \tag{12}$$

where $c_0, c_1 > 0$ and $m \to \infty$ with T. Theoretically c_0 and c_1 can be proportional to $T^{-1/(P+Q+3)}$. In the calculation, Zeng et al.[37] set $m = 50$, and $\mathcal{H}_T = \{h_k = \frac{1}{50}k^2T^{-1/(P+Q+3)}\hat{\sigma}, k = 1, 2, \cdots, 50\}$ with $\hat{\sigma} = \sqrt{T/12(T+1)}$.

4 | MI BASED MULTIPLE DAY TIME SERIES BICLUSTERING ALGORITHM

Let $\mathcal{S} = \{1, \ldots, S\}$ denote a set of S stocks that are observed over T time intervals over N trading days. For each trading day, our MI-MDTSB algorithm first clusters the rows (stocks), see Step 1. Within each cluster obtained in Step 1, our algorithm operates on the log returns matrix $\mathbf{X}$ by (a) deleting columns (time points), see Step 2, (b) deleting rows (stocks) according to the absence of similarity between the relevant entries in terms of mutual information, see Step 3, followed by (c) adding back rows appropriately, see Step 4. The output of Step 4 is an identified bicluster. Let α, β and γ be user-defined thresholds. On trading day n, let K_n^* denote the optimum number of clusters, and $K_{\max}$ denote a pre-determined maximum number of possible clusters, and $C_{\kappa,n}$ denote cluster κ, for $\kappa = 1, \ldots, K_n^*$. Let $\mathcal{X}$ denote the corresponding set of seeds for the clusters, and let $|\mathcal{X}|$ denote its cardinality, i.e., the number of seeds in the set $\mathcal{X}$. The animation of the algorithm is on GitHub.

4.1 | Algorithm

For a single trading day, the algorithm consists of the following steps. Here we suppress all the n subscripts for brevity.

Step 1: Cluster rows (stocks). The row clustering algorithm consists of the following sub-steps.

Step 1-1: Select seeds for the first two clusters.

Step 1-1a: Set $g = 2$.

https://github.com/lht1949/Animation-for-mutual-information-based-bilustering

Step 1-1b: Compute the pairwise mutual information based distances $\hat{D}(s_i, s_i')$ for two different stocks $s_i, s_i' \in \mathcal{S}$, $i \neq i'$.

Step 1-1c: Find the pair of stocks with the largest distance $\bar{D}^2_{\max} = \max_{s_i, s_i' \in \mathcal{S}} \hat{D}(s_i, s_i')$, and denote these as x_1 and x_2.

Step 1-1d: Select x_1 as the seed for the first cluster C_1 and x_2 as the seed for the second cluster C_2. Remove these "seed" stocks x_1 and x_2 from the "stocks set" $\mathcal{S}$, and denote the current set of remaining stocks as $\mathcal{S}^g$ (with cardinality $S-2$). Set the current seed set as $\mathcal{X}^g = \{x_1, x_2\}$ (with cardinality 2).

Step 1-2: Determine the number of clusters and corresponding seeds for successive clusters.

Step 1-2a: Compute $\hat{D}(s_i, x_i)$ for all $s_i \in \mathcal{S}^g$ and $x_i \in \mathcal{X}^g$.

Step 1-2b: Compute the average distance between stock s_i and all the seeds in set $\mathcal{X}^g$ as

$$\bar{D}_{s_i} = \sum_{x_i \in \mathcal{X}^g} \frac{\hat{D}(s_i, x_i)}{|\mathcal{X}^g|}$$

where $|\mathcal{X}^g|$ denotes the number of seeds in the set $\mathcal{X}^g$.

Step 1-2c: Find the stock with the largest average distance from all current clusters, given by $\bar{D}^{g+1}_{\max} = \max_{s_i \in \mathcal{S}^g}(\bar{D}_{s_i})$.

Step 1-2d: Compare $\bar{D}^g_{\max}$ and $\bar{D}^{g+1}_{\max}$.

- If $\bar{D}^g_{\max} \geq \bar{D}^{g+1}_{\max}$, set x_{g+1} as the seed for a new $(g+1)$th cluster. Move x_{g+1} from the set $\mathcal{S}^g$ to the current seed set $\mathcal{X}^g$. Denote the updated seed set and stocks set as $\mathcal{X}^{g+1}$ and $\mathcal{S}^{g+1}$ respectively. Set $g = g+1$, and repeat from Step 1-2b to Step 1-2d until $\bar{D}^g_{\max} < \bar{D}^{g+1}_{\max}$.
- If $\bar{D}^g_{\max} < \bar{D}^{g+1}_{\max}$, conclude there are $K^* = g$ clusters.

Step 1-3: Assign stocks to the K^* clusters.

Step 1-3a: For each stock $s_i \in \mathcal{S}^g$, calculate the distance $\hat{D}(s_i, s_f)$ for all $s_f \in C_\kappa$, $\kappa = 1, \cdots, K^*$.

Step 1-3b: Record the value $D_{s_i, C_\kappa} = \max_{s_f \in C_\kappa}(\hat{D}(s_i, s_f))$ as the distance between stock s_i and cluster C_κ.

Step 1-3c: Find the cluster that has smallest distance with stock s_i, i.e., $\min_{\kappa=1,\cdots,K^*}(D_{s_i, C_\kappa})$, denote this cluster as C_{κ_i}.

Step 1-3d: Find the stock that has the smallest distance with the cluster, i.e, s_j has the smallest distance, $\min_{i=1,\cdots,S}(C_{\kappa_i})$. Add stock s_j to cluster C_{κ_j}, remove stock s_j from $\mathcal{S}^g$, and denote the updated stocks set as $\mathcal{S}^{g+1}$. Set g=g+1.

Step 1-3e: Repeat from Step 1-3a to Step 1-3d until $\mathcal{S}_g$ is empty.

The output is K^* clusters, and K^* is possibly different for different trading days. Let $\mathbf{X}_\kappa$ denote the log return matrix for the κ cluster and $\kappa = 1, 2, \ldots, K^*$. For each cluster κ, with the input matrix $\mathbf{X}_\kappa$, we repeat Steps 2 to 4.

Step 2: Cluster columns. To cluster columns and preserve contiguity of the time sequence, we divide all the columns into groups (each group contains d contiguous columns (for a user specified d), measure the distance between a group (of length d) and its following group or neighbor to its right (of length d) in terms of mutual information by using equation (11), and merge the groups if the groups are close to each other in terms of mutual information.

Step 2a: Let the vector $\mathbf{x}_{\cdot t}$ denote the column (time) t in the matrix $\mathbf{X}_\kappa$, for $t = 1, \ldots, T$. Let $\mathbf{X}_{\cdot j} = (\mathbf{x}_{\cdot t}, \ldots, \mathbf{x}_{\cdot t+d})$ denote d columns (say $d = 10$ minutes) in the matrix $\mathbf{X}_\kappa$, for $t = 1, (1+d), \ldots, (\frac{T}{d}+d)$ and $j = 1, \ldots, J$, where $J = \frac{T}{d}$. For $j = 1, \ldots, J-1$, compute $m_j = \widehat{\text{JMI}}\,(\mathbf{X}_{\cdot j}, \mathbf{X}_{\cdot j+1})$, the groupwise MI for the group $\mathbf{X}_{\cdot j}$ and its succeeding group $\mathbf{X}_{\cdot j+1}$.

Step 2b: Let $\mathcal{J}$ be the placeholder for any j such that $m_j > \alpha$ (threshold). Let e denote the user specified minimum number of consecutive time intervals (each of length d) that constitute a bicluster. For example, $e = 3$ means that a bicluster must minimally be 30 minutes long, given that $d = 10$. Having selected a value e, let $\mathcal{J}'$ define the set with at least e consecutive intervals of length d in the set $\mathcal{J}$, and F denote the cardinality of the set $\mathcal{J}'$, with $F \leq \frac{[T/d]}{e}$. Let $\mathbf{X}^{1,f}_\kappa$ denote the log returns matrix containing all the columns in the fth set, where $f = 1, \cdots, F$. For simplicity, we suppress the superscript f and denote $\mathbf{X}^{1,f}_\kappa$ by $\mathbf{X}^1_\kappa$, which forms the input for next step. For $f = 1, \ldots, F$, repeat Step 3 to 4 shown below.

Step 3: Row Deletion. After clustering the columns, the rows of cluster κ need to be adjusted under this updated time interval. To do so, for each row (stock) in $\mathbf{X}^1_\kappa$, we measure the distance between this row and the remaining rows in $\mathbf{X}^1_\kappa$ by computing the groupwise MI; we do this for each $f = 1, \cdots, F$. We keep the rows (stocks) if the row is close in MI to the remaining rows.

Step 3a: Let vector $\mathbf{x}_s^1$ denote row s in matrix $\mathbf{X}_\kappa^1$. Let $\mathbf{x}_{s'.}^{\mathbf{1}}$ denote all the rows (stocks) in the matrix $\mathbf{X}_\kappa^1$, except row s. Compute the MI between x_s^1 and $\mathbf{x}_{s'.}^{\mathbf{1}}$, $m_{s_d} = \widehat{\text{JMI}}\,(x_s^1, \mathbf{x}_{s'.}^{\mathbf{1}})$.
Step 3b: If $m_{s_d} > \beta$, add s to the set $\mathcal{I}$. Denote $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$ as the log returns matrix containing all $s \in \mathcal{I}$; $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$ is an input into Step 4.

Step 4: Row Insertion. After row deletion, for each stock not in $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$, we measure the distance between this stock and $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$. We insert the stock to the rows of $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$, if a stock is close to $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$.
Step 4a: Let the vector $\mathbf{x}_{\mathbf{s}}^{\mathbf{2}}$ denote row s in the matrix $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{2}}$. Compute $m_{s_i} = \widehat{\text{JMI}}\,(x_s^2, \mathbf{x}_{\boldsymbol{\kappa}}^{\mathbf{2}})$ for each stock $s \in \mathcal{S}$ and $s \notin \mathcal{I}$.
Step 4b: If $m_{s_i} > \gamma$, add s to the set $\mathcal{I}$. Denote $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{3}}$ as the log returns matrix containing all $s \in \mathcal{I}$. Then, $\mathbf{X}_{\boldsymbol{\kappa}}^{\mathbf{3}}$ is the identified bicluster.

Parameter Selection The choice of the time length d depends on the applications and domains. Generally, a smaller d is preferred to identify a bicluster with shorter time length. By grouping fewer time points together, the algorithm may fail in detecting a bicluster with longer period. By contrast, a larger d is preferred for a bicluster with longer time length but may fail in detecting a bicluster with shorter period. The user-defined threshold α is the tolerance for column deletion. A large α tends to eliminate more columns and result in biclusters within a relatively short time interval. In our analysis, we use a data-driven approach to dynamically determine the value of α. For each trading day, and each cluster from Step 1, we repeat Step 2a for all the previous h trading days (here $h = 10$), and set α equal to the 30th percentile of the distribution of m_j for the previous h trading days. Similarly, β is the tolerance for row deletion. A large β tends to eliminate more rows and result in small biclusters. We repeat Step 3a for all the previous h trading days, and set β equal to the 70th percentile of the distribution of m_{s_d} for the previous h trading days. The parameter γ is the tolerance for row insertion. We repeat Step 4a for all the previous h trading days, and set γ equal to the 95th percentile of the distribution of m_{s_i} for the previous h trading days.

Algorithm Complexity and Computation Time The computation time of the algorithm mainly dependents on the number of times computing JMI and the size of the matrices. Table 3 shows the computation time of JMI of different matrix sizes, and that increases exponentially as the dimension of a matrix becomes bigger. The time cost of Step 1 is negligible since the pairwise mutual information based distance can be calculated once and reused those distance for all the trading days. The most time consuming part of the algorithm is from Step 2 to Step 4, which requires intensive JMI computation to delete rows and columns and insert rows. The time complexity of the algorithm depends on the number of clusters κ from Step 1, the number of preserved time intervals F in Step 2b. For example, the output of Step 1 is 13 stocks, Step 2b preserves $F = 3$ time intervals (40, 90, and 80 minutes), the computation time from Step 2b to Step 4 for these three sub-matrices are 325, 1566, and 1290 seconds. If fixed parameters are used, the time is reduced to 39, 176, 120 seconds without parameter selection. The experiment was conducted on Google Colab.

TABLE 3 JMI computation time of different matrix sizes.

Matrix dimension (row x column)	JMI Computation time (seconds)	Matrix dimension (row x column)	JMI Computation time (seconds)
100x1	0.009	100x10	0.101
1000x1	1.454	1000x10	8.627
5000x1	31.215	5000x10	312.265

4.2 | Algorithm validation via simulation study

To validate the algorithm, we designed a synthetic data set with five stocks over $T = 300$ time points. Assume that stocks s_1, s_2, and s_3 co-move from time t_{101} to time t_{200} (in terms of having high MI), which is indicated by a bicluster in the shaded green rectangle in Figure 2. Let $d = 10$, $e = 1$, and $J = \frac{T}{d}$ (see Step 2 of the algorithm). For $j = 1, \ldots, J$, let $\mathbf{X}_{.j} = (\mathbf{x}_{.t}, \ldots, \mathbf{x}_{.t+d})$ denote the data matrix corresponding to stocks s_1, s_2, and s_3 (only the first three rows of the matrix), where $t = 1, (1+d), \ldots, (\frac{T}{d} + d)$. This example has $L = 3$ groups. To design a bicluster shown in Figure 2, we need high groupwise MI between $\mathbf{X}_{.j}$ and $\mathbf{X}_{.(j+1)}$ for $j = 11, \cdots, 20$, and low groupwise MI for $j = 1, \cdots, 10$ and $j = 21, \cdots, 30$. That is, we wish to simulate a data set with high groupwise MI in the time intervals within the bicluster, and low groupwise MI in the remaining time intervals.

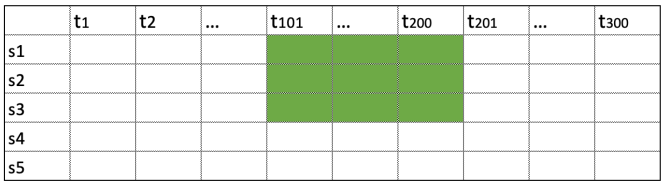


FIGURE 2 Structure of a designed bicluster (green shaded area) for simulated data.

It is well known that if $(X_1, X_2)'$ follows a bivariate normal distribution, i.e.,

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}\right),$$

the MI between X_1 and X_2 is $-\frac{1}{2}\log(1-\rho^2)$. Thus, a larger ρ is associated with a higher MI. Let the partitioned form of a multivariate distribution be denoted by

$$\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \sim N\left(\begin{pmatrix} \boldsymbol{\mu}_\mathbf{x} \\ \boldsymbol{\mu}_\mathbf{y} \end{pmatrix}, \boldsymbol{\Sigma}\right), \text{ where } \boldsymbol{\Sigma} = \left(\begin{array}{c|c} \boldsymbol{\Sigma}_{\mathbf{11}} & \boldsymbol{\Sigma}_{\mathbf{12}} \\ \hline \boldsymbol{\Sigma}_{\mathbf{21}} & \boldsymbol{\Sigma}_{\mathbf{22}} \end{array}\right), \tag{13}$$

where $\boldsymbol{\Sigma}_{\mathbf{11}} = \text{Var}(\mathbf{x})$, $\boldsymbol{\Sigma}_{\mathbf{22}} = \text{Var}(\mathbf{y})$, and $\boldsymbol{\Sigma}_{\mathbf{12}} = \text{Cov}(\mathbf{x}, \mathbf{y})$. We assume that

$$\boldsymbol{\Sigma}_{\mathbf{11}} = \begin{bmatrix} 1 & 0.9 & \dots & 0.9 \\ 0.9 & 1 & \ddots & \vdots \\ \vdots & \ddots & 1 & 0.9 \\ 0.9 & \dots & 0.9 & 1 \end{bmatrix}, \quad \boldsymbol{\Sigma}_{\mathbf{12}} = \begin{bmatrix} 0.01 & 0.01 & \dots & 0.01 \\ 0.01 & 0.01 & \ddots & \vdots \\ \vdots & \ddots & 0.01 & 0.01 \\ 0.01 & \dots & 0.01 & 0.01 \end{bmatrix}, \quad \boldsymbol{\Sigma}_{\mathbf{22}} = \begin{bmatrix} 1 & 0.01 & \dots & 0.01 \\ 0.01 & 1 & \ddots & \vdots \\ \vdots & \ddots & 1 & 0.01 \\ 0.01 & \dots & 0.01 & 1 \end{bmatrix}. \tag{14}$$

We have designed three groups of data submatrices $\mathbf{G_1}$, $\mathbf{G_2}$, and $\mathbf{G_3}$ (see Figure 3) each having three rows (corresponding to stocks s_1, s_2, and s_3) and $d = 10$ columns. We drew a random sample from a multivariate normal distribution with covariance matrix $\boldsymbol{\Sigma}$ with dimension $3Ld \times 3Ld$ (or, 90×90 in this example), as shown in (13) and (14). We assign the first $3d = 30$ elements of the generated sample as $\mathbf{G_1}$, the next $3d = 30$ elements as $\mathbf{G_2}$, and the last $3d = 30$ elements as $\mathbf{G_3}$. In this way, the data matrix $\mathbf{G_1}$ has high MI with $\mathbf{G_2}$, and low mutual information with $\mathbf{G_3}$. The MI between $\mathbf{G_2}$ and $\mathbf{G_3}$ is low as well.

We perturbed these data submatrices by adding a different random noise generated from a $N(0, \delta)$ distribution to each of their elements, for small δ (we used 0.001). For simplicity, we continue to use the notation $\mathbf{G}_1, \mathbf{G}_2$, and $\mathbf{G}_3$ for the perturbed data matrices. We then concatenated $\mathbf{G_1}$ and $\mathbf{G_2}$ repeatedly and denote this concatenated matrix as $\mathbf{D}_{middle}$; it has three rows (stocks) and 100 columns (time points). Similarly, we concatenated $\mathbf{G_1}$ and $\mathbf{G_3}$ repeatedly and denoted it as $\mathbf{D}_{start}$, and concatenated $\mathbf{G_3}$ and $\mathbf{G_2}$ repeatedly and denoted as $\mathbf{D}_{end}$. Figure 4 shows the final simulated time series, which are the concatenation of $\mathbf{D}_{start}$, $\mathbf{D}_{middle}$ and $\mathbf{D}_{end}$. Figure 5 shows plots of one set of generated realizations for the stocks s_1, s_2, and s_3. Stocks s_4 and s_5 were drawn from a normal distribution $N(0, 0.01)$, together generating the full data matrix shown in Figure 2. We created 100 replicated samples of the data matrix with five stocks and $T = 300$ columns, and employed our biclustering algorithm. The algorithm detected the bicluster correctly in 89 of these replicated samples, corresponding to an error rate of only 11%. We also compared with several existing benchmark approaches, such as Cheng and Church[7] and Zhang et al.[22]. Neither of them can detect the designed biclusters successfully.

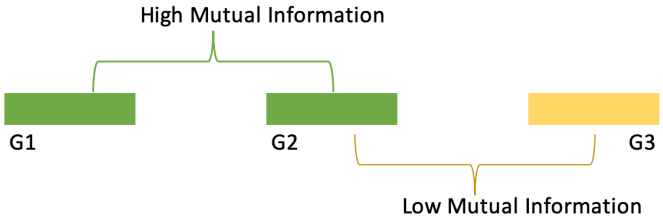


FIGURE 3 Three groups of the simulated data matrix.

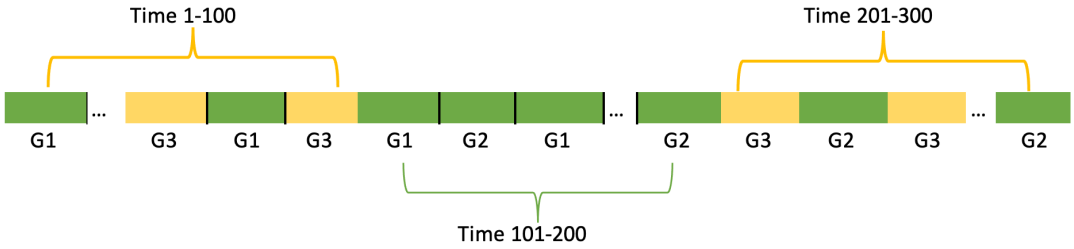


FIGURE 4 A representation of the simulated data structure.

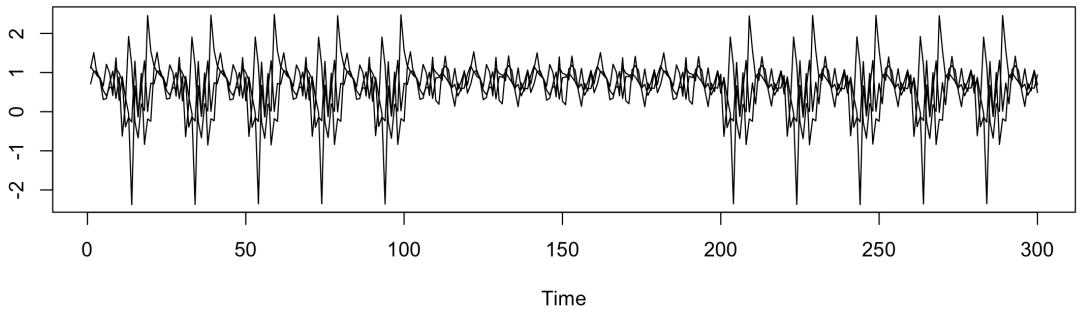


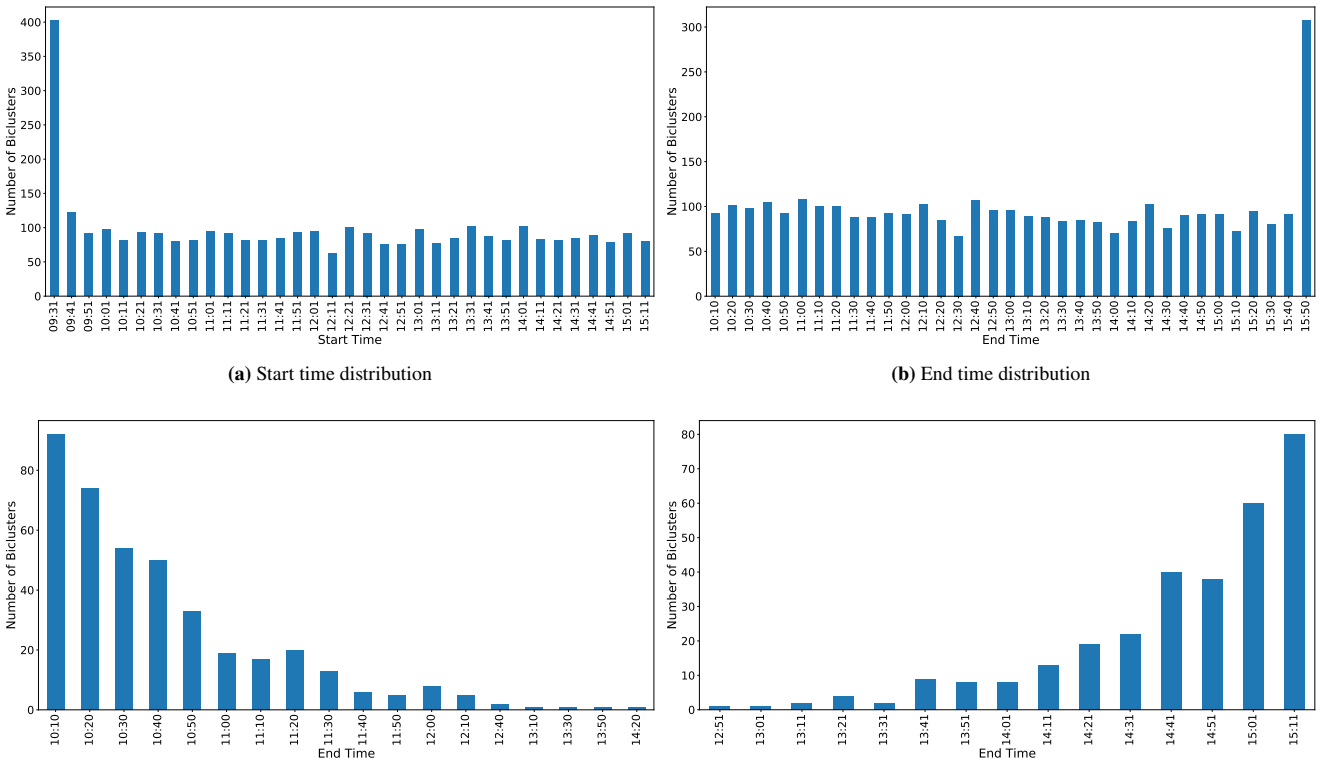
FIGURE 5 Plots of the simulated time series for stocks s_1, s_2, and s_3.

5 | BICLUSTERING INTRA-DAY STOCK RETURNS

As described in Section 2, the real data for our analysis are from S&P 100 companies for which the entire 2013 data are available. For each trading day, we have a data matrix where each column corresponds to a time point, while the rows are the log-returns of the one-minute average stock prices for the stocks. The biclustering algorithm in Section 4 operates on these data matrices. The average computing time (the computation was performed on Google Colab) for each trading day is around three hours, the JMI estimations in each step and parameter selection being the most computationally expensive parts. To accelerate the run time and reduce the computing time significantly, parallel computing can be implemented to run the algorithm for different trading days simultaneously on different cores.

Figure 6 (a) shows the distributions of start time of the biclusters for all trading days in 2013. The number of biclusters that start at different time points is uniformly distributed except those starting at 9:31, the opening time of the stock market when trading is very active to reflect the information transmitted overnight. For the distributions of the end time, Figure 6 (b) indicates a similar pattern except for those biclusters ending at 15:51, the closing time, as traders adjust their positions. We also examine the end time distributions for all biclusters starting at 9:31, and the start time distributions for all biclusters ending at 15:51. Figure 6 (c) indicates that most of the biclusters starting at 9:31 end before 11:00, which perhaps reflects the time taken for the market to adjust for the overnight information. Figure 6 (d) indicates that most of the biclusters ending at 15:51 start after 14:41, which possibly means that the traders start to balance their positions before market closing. Such trading patterns (reflecting diurnal effect) provides empirical evidence that a clustering (rather than a biclustering) algorithm may be inadequate for high-frequency financial trading, since the comovement pattern is disrupted by overnight information digestion and traders' activities for re-balancing their positions.

Bicluster duration distributions. Figure 7 shows the distributions of durations of all the biclusters over the trading days in 2013. The distribution is right-skewed, with a long tail. For most of the biclusters, the duration ranges from 40 to 100 minutes. The number of biclusters with a duration of more than 180 minutes is almost negligible.



(a) Start time distribution

(b) End time distribution

(c) End time distribution for biclusters starting at 9:31

(d) Start time distribution for biclusters ending at 15:50

FIGURE 6 (a) and (b) show the distribution of start and end time over trading days in 2013. (c) and (d) show the distributions of for bicluster starting at 9:31 and ending at 15:50 over trading days in 2013.

6 | ANALYSIS OF COMOVEMENT OF STOCKS

Post biclustering, we analyze the comovement of stocks in two ways. One way is to explore comovement probability (see equations (15) and (16)) of selected m-tuples of stocks. A second is to study the periodicity of such comovement patterns.

6.1 | Comovement Probability

We explore interesting patterns in selected m-tuples of stocks which fall within the same bicluster in a calendar week w in 2013, for $w = 1, \ldots, W$. To do this, we employ the cumulative empirical comovement probability[23]. Let $T(w)$ denote the number of trading days in week w, and $\tau_{\mathbf{F}}(w)$ denote the number of trading days for which the m-tuple falls within the same biclusters in week w. The comovement probability for an m-tuple of stocks in a given week w is computed as the proportion

$$\mathscr{P}_{\mathbf{F}}(w) = \frac{\tau_{\mathbf{F}}(w)}{T(w)}. \tag{15}$$

The cumulative comovement probability for an m-tuple *up to* a given week w_c, is then computed as the proportion

$$P_{\mathbf{F}}(w_c) = \frac{\sum_{w=1}^{w_c} \tau_{\mathbf{F}}(w)}{\sum_{w=1}^{w_c} T(w)}. \tag{16}$$

Post biclustering, we calculate the cumulative empirical comovement probability up to 53 weeks for all pairs of 94 stocks. While the above formulas may be applied to any m-tuple of assets, we report results for the $m = 2$ case since it has an interesting connection with pair trading in financial applications. Table 4 shows the top $m = 2$ comovement pairs in each industry sector as defined by the Global Industry Classification Standard (GICS). Energy, Consumer Discretionary and Consumer Staples sectors are associated with relatively higher comovement probability, while in contrast, the Finance sector has a lower comovement probability.

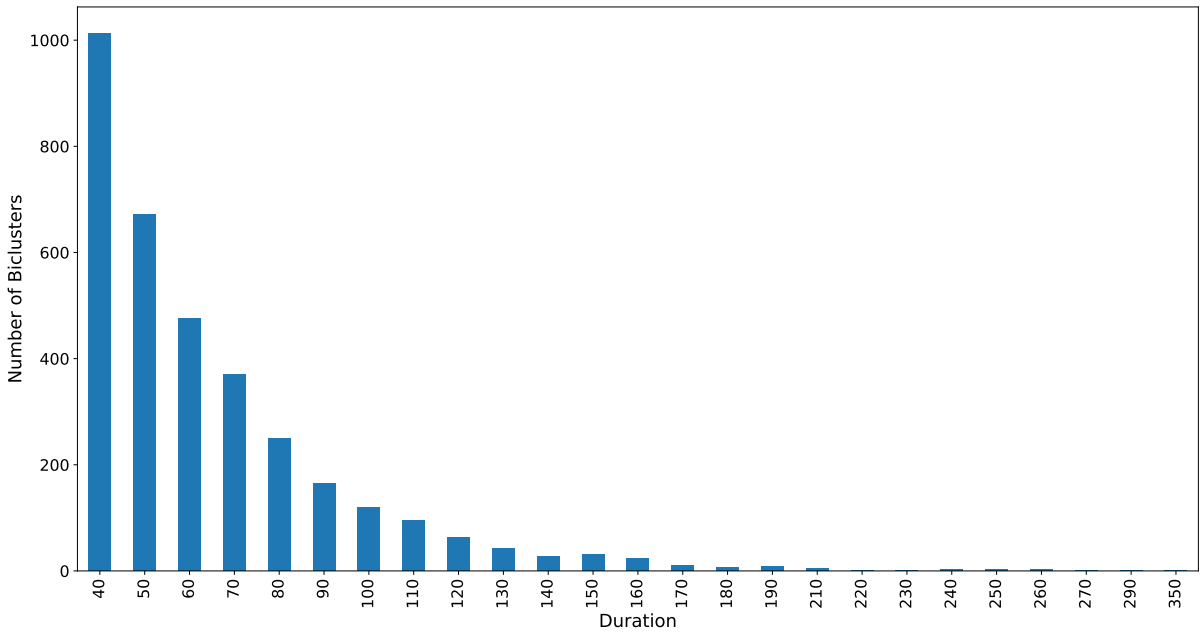


FIGURE 7 Distribution of bicluster durations over trading days in 2013.

TABLE 4 Top $m = 2$ comovement pairs in each GICS sector.

Sectors	S1	S2	P(%)
Consumer	DIS	TWX	40.67
Discretionary	DIS	HD	35.20
Consumer	PM	MO	38.01
Staples	CL	PG	36.01
Energy	XOM	CVX	49.07
	CVX	COP	48.18
Financials	SPG	AXP	20.25
	AXP	BK	15.72
Health	MDT	JNJ	36.09
	JNJ	LLY	34.10
Information	QCOM	TXN	32.51
Technology	QCOM	ORCL	31.16
Industrials	MMM	HON	42.67
	EMR	HON	42.51
Materials	DD	DOW	32.92
	DOW	MON	30.47
Telecom	VZ	T	29.65
Utilities	SO	EXC	32.08

P denotes the cumulative comovement proportion (up to week 53) defined in equation (16). For Telecom and Utilities sectors, there are only two stocks in the S&P 100. Accordingly, only one pair of co-moving stocks is shown in this table for these sectors.

6.2 | Periodicity in comovement of stocks

An interesting question to ask is which comoving stock pairs will repeat their comovement pattern periodically. To investigate the existence and nature of such patterns, we form binary time series $Y_{i,t} = (Y_{i,1}, \cdots, Y_{N,T})$ for all $N = 4372$ two-pair stocks for $T = 247$ weekdays from January 21, 2013, to December 31, 2013, with 1 indicating that the two-pair stocks comove together on this day, and otherwise 0. A frequency domain analysis on each binary comovement time series to detect possible periodicity will be interesting.

For Gaussian time series Y_t with second-order periodogram

$$I(\omega) = \frac{1}{N} \mid \sum_{t=1}^{N} Y_t \exp(-i\omega t) \mid^2, \quad \omega \in [0, \pi], \tag{17}$$

TABLE 5 Stock pairs that comove once every 2 days, with their GICS sector

Stock one	(Industry)	Stock two	(Industry)
CVX	(Energy)	MS	(Financials)
PEP	(Comsumer Staples)	MET	(Financials)
EBAY	(IT)	JPM	(Financials)
MON	(Materials)	JPM	(Financials)
HPQ	(IT)	VZ	(Telecom)
UPS	(Industrials)	GS	(Financials)
BAX	(Health)	GS	(Financials)
MMM	(Industrials)	MS	(Financials)
WFC	(Financials)	F	(Consumer Discretionary)
LOW	(Consumer Discretionary)	ABBV	(Health)
CAT	(Industrials)	VZ	(Telecom)

the well-known Fisher's maximum periodogram ordinate procedure[41,42] detects periodicity by testing whether the maximum sample periodogram ordinate is significant. The g-statistic given by

$$g = \frac{\max_k I(\omega_k)}{\sum_{k=1}^{N/2} I(\omega_k)}, \tag{18}$$

and p-value of the test is obtained through the distribution of g given by

$$P(g > x) = n(1-x)^{n-1} - \frac{n(n-1)}{2}(1-2x)^{n-1} + \cdots + (-1)^p \frac{n!}{p!(n-p)!}(1-px)^{n-1} \tag{19}$$

where $n = [N/2]$ and p is the largest integer less than $1/x$.

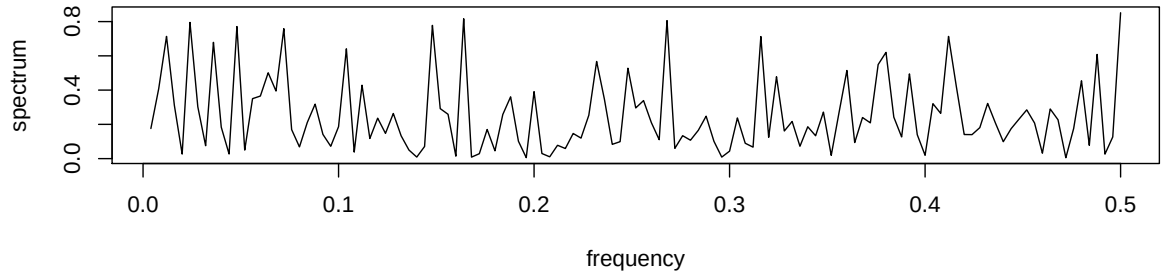
Although Fisher's g-test assumes the time series is Gaussian, we expect this should work approximately well as a preliminary analysis for detecting periodicity in the binary comovement time series data $Y_1, \cdots, Y_N$, where $N = 4372$ two-pair stocks. To perform this periodic analysis, we use the R package GeneCycle[43]. Figure 8 (b) shows a plot of the sample spectrum of the binary (comovement) time series for the stock pair MON and JPM, and indicates that the maximum periodogram occurs at frequency 0.42 with p-value 0.018 for the g-test. This corresponds to these two stocks comoving roughly once every two days. The other plots show comovements with periods of three and four days respectively. In contrast, the ordinates of the sample periodogram of the binary time series corresponding to the stock pair MSFT and CVS in Figure 8 (a) are uniformly distributed with no significant peak, suggesting that these two stocks do not comove periodically.

Table 5 shows that all the two-pair stocks comove once every 2 days, and are from different industries. In Figures 9 and 10 show the visualization of stocks with 1 (colored cell) indicating the pair of stocks comove and otherwise 0. Dark blue shaded cells indicate that the stock pairs comoving together are in different industries, while the light blue shaded cells indicate that they are in the same industry.

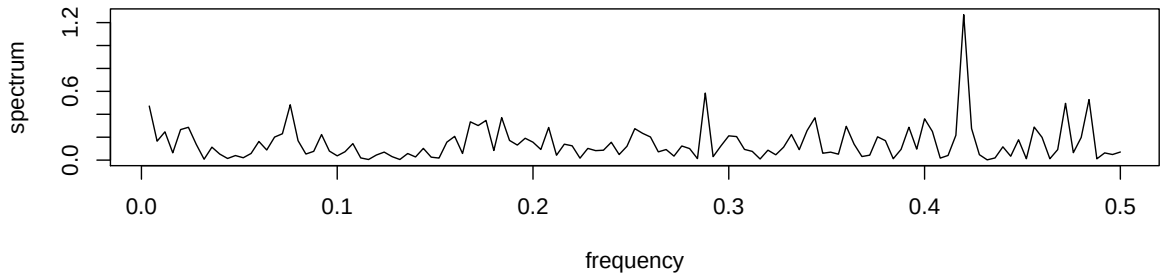
7 | SUMMARY AND DISCUSSION

Biclustering in financial settings exploits the grouping of stocks and times simultaneously, and thus facilitates improved pattern extraction for designing trading strategies. While biclustering methods may be employed for grouping low-frequency (say, daily) financial data such as log returns of assets, their use with high-frequency time series of intra-day trading data is especially useful.

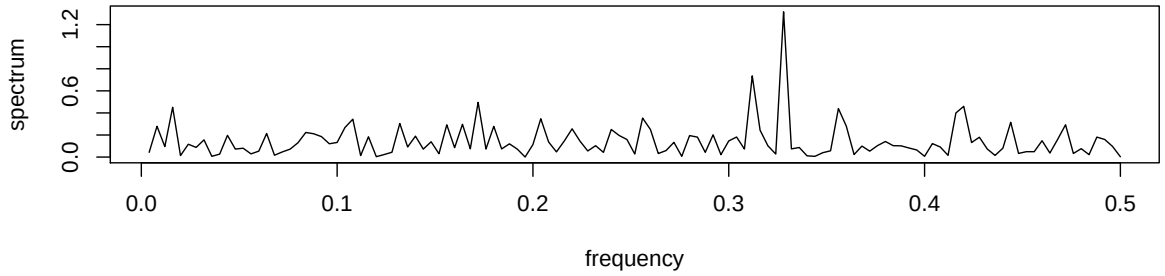
In this paper, we have described a novel mutual information based multiple day time series biclustering algorithm (MI-MDTSB) to capture the non-linear variability of asset returns on transaction-level intraday data and to dynamically study patterns of comovement over multiple trading days. To preserve the contiguity of the sequence of times, we divide the time interval into multiple groups with equal length, and measure the similarity between every group and its neighbor to the right based on the groupwise JMI. Moreover, the proposed algorithm dynamically determines biclusters by preserving the contiguity of time series data. Our numerical results reveal interesting trading patterns including diurnal effects, duration spreads and comovement periodicity and persistence. Interesting future directions include the design and analysis of different trading strategies that incorporate these patterns and a study of their performance.



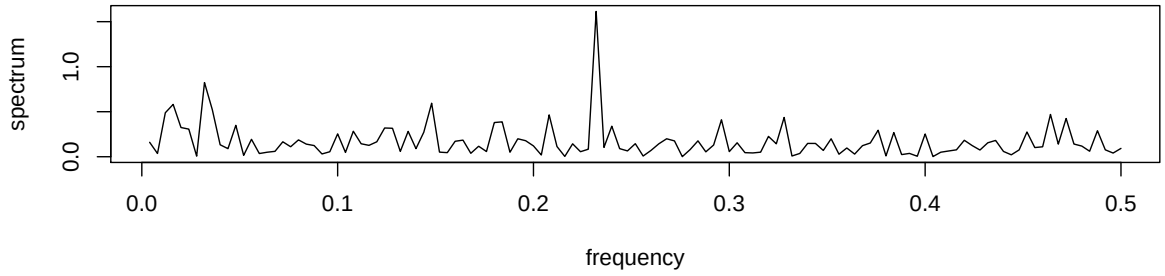
(a) No comovement of the stocks MSFT and CVS



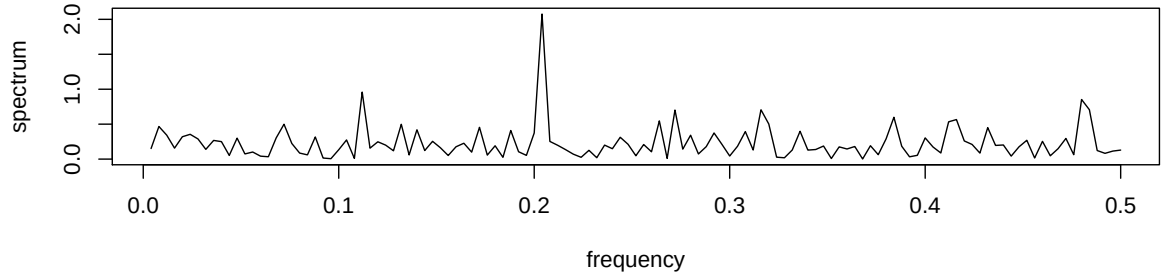
(b) Stocks MON and JPM comove once every 2 days.



(c) Stocks ORCL and BAC comove once every 3 days.



(d) Stocks ALL and C comove once every 4 days



(e) Stocks WAG and BK comove once every 5 days

FIGURE 8 Sample spectrum plots of the binary (comovement) time series.

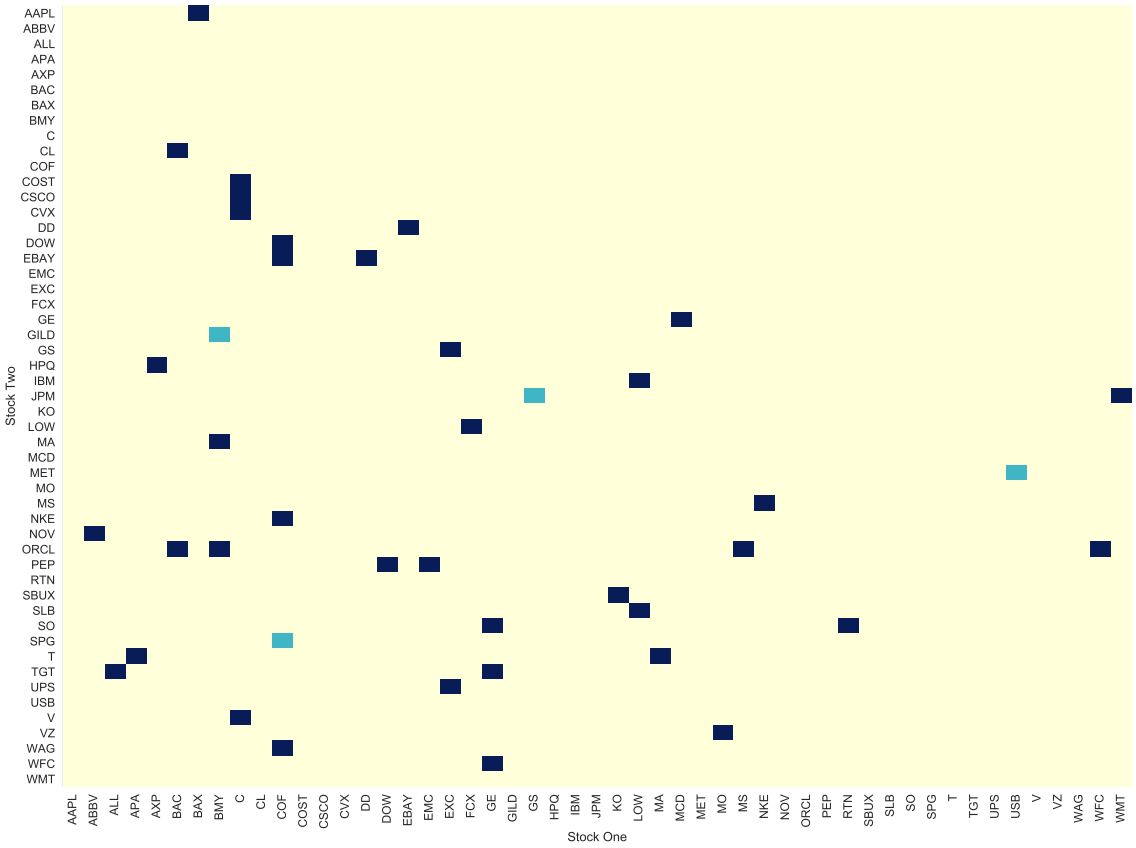
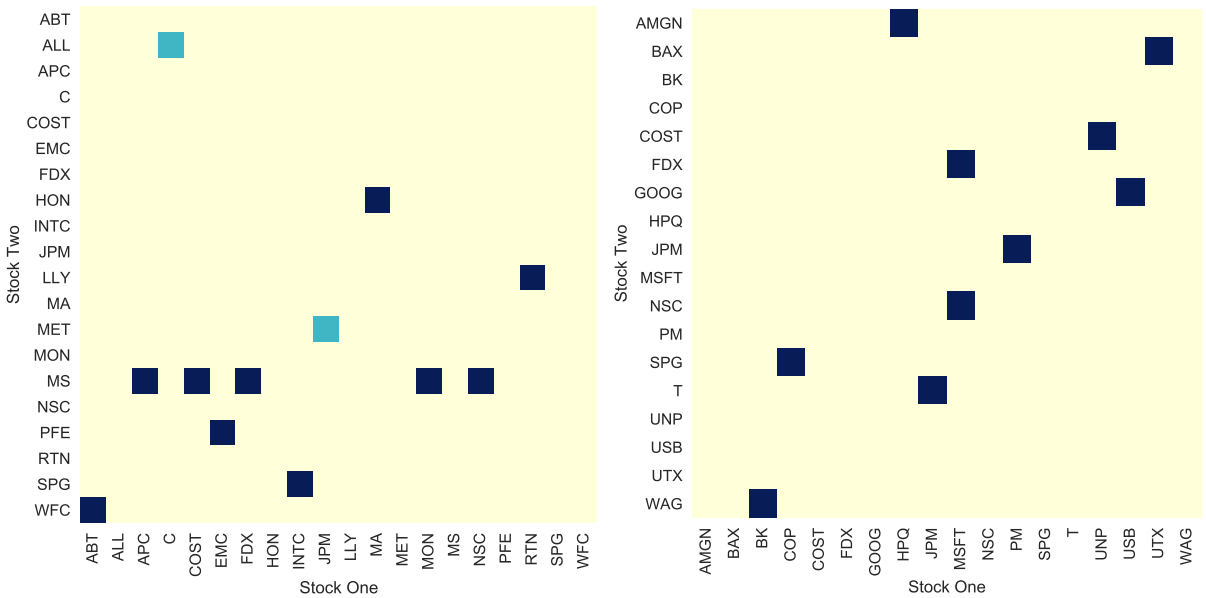


FIGURE 9 Visualization of stock pairs that comove once every 3 days. Dark blue cells indicate that the comoving stocks are in different industries, and light blue cells that they are in the same industry.



(a) Comove once every 4 days

(b) Comove once every 5 days

FIGURE 10 Visualization of stock pairs that comove once every 4 or 5 days. Dark blue cells indicate that the comoving stocks are in different industries, and light blue cells that they are in the same industry.

8 | ACKNOWLEDGEMENT

We thank the referees whose insightful comments helped us enhance the paper. This paper was based upon work partially supported by the National Science Foundation under Grant DMS-1638521 to the Statistical and Applied Mathematical Sciences Institute. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

1. Elliott RJ, Van Der Hoek J, Malcolm WP. Pairs trading. *Quantitative Finance* 2005; 5(3): 271–276.
2. Vidyamurthy G. *Pairs trading: quantitative methods and analysis*. 217. John Wiley & Sons . 2004.
3. Do B, Faff R, Hamza K. A new approach to modeling and estimation for pairs trading. In: Proceedings of 2006 Financial Management Association European Conference. ; 2006: 87–99.
4. Aghabozorgi S, Shirkhorshidi AS, Wah TY. Time-series clustering–a decade review. *Information Systems* 2015; 53: 16–38.
5. Hartigan JA. Direct clustering of a data matrix. *Journal of the American Statistical Association* 1972; 67(337): 123–129.
6. Padilha VA, Campello RJ. A systematic comparative evaluation of biclustering techniques. *BMC Bioinformatics* 2017; 18(1): 55.
7. Cheng Y, Church GM. Biclustering of expression data. In: ISMB Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology. AAAI Press; 2000: 93–103.
8. Dhillon IS. Co-clustering documents and words using bipartite spectral graph partitioning. In: ACM Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ; 2001: 269–274.
9. Bergmann S, Ihmels J, Barkai N. Iterative signature algorithm for the analysis of large-scale gene expression data. *Physical Review E* 2003; 67(3): 031902.
10. Cho H, Dhillon IS, Guan Y, Sra S. Minimum sum-squared residue co-clustering of gene expression data. In: SIAM Proceedings of the 2004 SIAM International Conference on Data Mining. ; 2004: 114–125.
11. Li G, Ma Q, Tang H, Paterson AH, Xu Y. QUBIC: a qualitative biclustering algorithm for analyses of gene expression data. *Nucleic Acids Research* 2009; 37(15): e101–e101.
12. Bozdağ D, Parvin JD, Catalyurek UV. A biclustering method to discover co-regulated genes using diverse gene expression datasets. In: Springer. ; 2009: 151–163.
13. Prelić A, Bleuler S, Zimmermann P, et al. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics* 2006; 22(9): 1122–1129.
14. Rodriguez-Baena DS, Perez-Pulido AJ, Aguilar-Ruiz JS. A biclustering algorithm for extracting bit-patterns from binary datasets. *Bioinformatics* 2011; 27(19): 2738–2745.
15. Serin A, Vingron M. Debi: Discovering differentially expressed biclusters using a frequent itemset approach. *Algorithms for Molecular Biology* 2011; 6(1): 1–12.
16. Kluger Y, Basri R, Chang JT, Gerstein M. Spectral biclustering of microarray data: coclustering genes and conditions. *Genome Research* 2003; 13(4): 703–716.
17. Izenman AJ, Harris PW, Mennis J, Jupin J, Obradovic Z. Local spatial biclustering and prediction of urban juvenile delinquency and recidivism. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 2011; 4(3): 259–275.
18. Hochreiter S, Bodenhofer U, Heusel M, et al. FABIA: factor analysis for bicluster acquisition. *Bioinformatics* 2010; 26(12): 1520–1527.

19. Murali T, Kasif S. Extracting conserved gene expression motifs from gene expression data. In: World Scientific. 2002 (pp. 77–88).

20. Cho H, Dhillon IS. Coclustering of human cancer microarrays using minimum sum-squared residue coclustering. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2008; 5(3): 385–400.

21. Madeira SC, Teixeira MC, Sa-Correia I, Oliveira AL. Identification of regulatory modules in time series gene expression data using a linear time biclustering algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 2008; 7(1): 153–165.

22. Zhang Y, Zha H, Chu CH. A time-series biclustering algorithm for revealing co-regulated genes. In: IEEE Information Technology: Coding and Computing, 2005. ITCC 2005. International Conference on. ; 2005: 32–37.

23. Liu H, Zou J, Ravishanker N. Multiple day biclustering of high-frequency financial time series. *Stat* 2018; 7(1): e176.

24. Banerjee A, Merugu S, Dhillon IS, Ghosh J, Lafferty J. Clustering with Bregman divergences.. *Journal of Machine Learning Research* 2005; 6(10).

25. Kraskov A, Stögbauer H, Grassberger P. Estimating mutual information. *Physical Review E* 2004; 69(6): 066138.

26. Li Y, Liu W, Jia Y, Dong H. A weighted mutual information biclustering algorithm for gene expression data. *Computer Science and Information Systems* 2017; 14(3): 643–660.

27. Dhillon IS, Mallela S, Modha DS. Information-theoretic co-clustering. In: ACM Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ; 2003: 89–98.

28. Zou J, An Y, Yan H. Volatility matrix inference in high-frequency finance with regularization and efficient computations. In: IEEE International Conference on Big Data. ; 2015: 2437–2444.

29. Hansen PR, Lunde A. Realized variance and market microstructure noise. *Journal of Business & Economic Statistics* 2006; 24(2): 127–161.

30. Aït-Sahalia Y, Mykland PA, Zhang L. Ultra high frequency volatility estimation with dependent microstructure noise. *Journal of Econometrics* 2011; 160(1): 160–175.

31. Engle RF, Russell JR. Autoregressive conditional duration: a new model for irregularly spaced transaction data. *Econometrica* 1998: 1127–1162.

32. Andersen TG, Bollerslev T. Intraday periodicity and volatility persistence in financial markets. *Journal of Empirical Finance* 1997; 4(2-3): 115–158.

33. Ait-Sahalia Y, Yu J. High frequency market microstructure noise estimates and liquidity measures. *Annals of Applied Statistics* 2009; 3(1): 422–457.

34. Hasbrouck J. Assessing the quality of a security market: a new approach to transaction-cost measurement. *The Review of Financial Studies* 1993; 6(1): 191–212.

35. Roll R. A simple implicit measure of the effective bid-ask spread in an efficient market. *The Journal of Finance* 1984; 39(4): 1127–1139.

36. Harris L. Statistical properties of the Roll serial covariance bid/ask spread estimator. *The Journal of Finance* 1990; 45(2): 579–590.

37. Zeng X, Xia Y, Tong H. Jackknife approach to the estimation of mutual information. *Proceedings of the National Academy of Sciences* 2018; 115(40): 9956–9961.

38. Cover TM, Thomas JA. *Entropy, Relative Entropy, and Mutual Information*ch. 2: 13-55; John Wiley Sons, Ltd . 2005

39. Liu H, Zou J, Ravishanker N. Clustering High-Frequency Financial Time Series Based on Information Theory. *Applied Stochastic Models in Business and Industry* 2021.

40. Xianli Z, Weiqiang H. *JMI: Jackknife Mutual Information*. 2018. R package version 0.1.0.

41. Fisher RA. Tests of significance in harmonic analysis. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 1929; 125(796): 54–59.

42. Priestley MB. *Spectral Analysis and Time Series: Probability and Mathematical Statistics*. No. 04; QA280, P7. . 1981.

43. Ahdesmaki M, Fokianos K, Strimmer. K. *GeneCycle: Identification of Periodically Expressed Genes*. 2019. R package version 1.1.4.