CONTENTS

ABSTRACT **1**
1. INTRODUCTION **1**
2. SYSTEM OVERVIEW **1**
2.2. DATA AND PARAMETERS **2**
2.3. ANALYSIS WORKFLOW **2**
2.4. INTERACTION AND RESULT VISUALIZATION **2**
3. DEMONSTRATION SCENARIOS **3**
4. CONCLUSIONS AND FUTURE WORK **4**
REFERENCES **4**

Correlation graph analytics for stock time series data

Tong Liu, Paolo Coletti, Anton Dignös, Johann Gamper and Maurizio Murgia
Free University of Bozen/Bolzano, Bozen/Bolzano, Italy
{name.surname}@unibz.it

ABSTRACT

Stock market events are hard to model. In recent years, one approach that has been receiving increasing attention is to analyze graphs induced by price correlations of different stock companies. By analyzing the structure of such graphs, it is possible to identify critical events, e.g., market crises. To the best of our knowledge, there are no tools available that offer comprehensive support for such analyses. This paper introduces a novel tool that offers in-depth analysis with the ability of fine tuning parameters with an intuitive user interface. With a proposed workflow to handle time series data, the tool becomes versatile and it can analyze correlation graphs of different semantics: minimum spanning tree, graphs with edge thresholds, and evolving graphs. It also provides a rich set of functions that enable users to explore easily, interactively and systematically the correlation graphs starting from a file of raw time series data. With real-world stock data, we demonstrate how straightforward yet effective it is to accomplish various analytical tasks with the proposed tool.

1 INTRODUCTION

Time series systems are pervasively used in many domains for different tasks such as monitoring and recording data gathered over time, with which analysts can discover meaningful and valuable information to understand the observed system and to avoid risks. The examples of applying time-series technologies to real-world problems are numerous, e.g., fault detection [18], seasonal trend analysis [21], financial data [11], etc.

For the analysis of financial data, Mantegna [14] proposed to compute all correlations between the time signals and to visualize them in a graph. This is illustrated in Fig. 1. First, the pairwise correlations between the five input signals are computed and stored in a matrix. Then, the correlation matrix is transformed into a graph, where nodes represent signals and edges are labeled with the correlation coefficient between the two nodes. Instead of visualizing a complete graph, the minimum spanning tree (MST) is often used as a compact overview of the signal correlations.

In recent years, a significant number of works have shown that by observing changes in the structure of a MST, it is possible to deduce important events in stock markets, such as market crises and volatility [5, 6, 16]. Many studies about stock market analysis adopted this approach and investigated the structure and topologies of the induced network or MST [4, 6, 9, 19]. The study of correlation graph analysis also raised the attention of the computer science community. Marti et al. [15] studied the best window length to calculate time series correlations. Azzalini et al. [2] proposed a technique to detect significant changes in financial time series clusters expressed with hierarchical correlation trees. Luo et al. [12] used correlation graphs to spot cheating behaviors with business data. In the database community, research focused on efficient methods for correlation graph

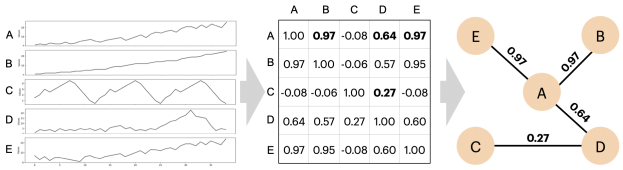


Figure 1: Correlation analysis of time series using MST.

analyses. Petrov et al. [17] proposed different approaches for exploring correlated time signals interactively. Aghasadeghi et al. [1] presented techniques to visualize evolving graphs at different resolutions. Likewise, Mamun et al. [13] studied efficient solutions for computing MSTs.

When it comes to correlation graph analytics for financial data, analysts have to tackle several challenges: identifying appropriate parameter values such that the computed correlation graph is meaningful and presents visible characteristics; investigating whether patterns exist for graphs under different representational semantics, such as MST, graphs with specific thresholds, graphs of signal classes; assessing the robustness of graph features (e.g., radius, degree distribution), and determining if they are reliable to capture special events. Existing tools in the finance literature [4, 6, 9, 19] suffer at least from the following two limitations: (i) they offer only a few of the above mentioned aspects and (ii) they were implemented ad hoc for specific case studies and are not designed to be easily integrated with other tools or analyses.

In this demo paper, we present a versatile and user-friendly tool for analyzing time series data based on the pairwise correlations that are visualized in a graph. We propose a workflow in which the parameters are considered logically, and it allows a high level of flexibility to visualize and analyze the graphs: the comparison of two graphs representing subsequences of the data selected by two different windows; the evolution and changes of a graph over time by applying a moving window; and a compact heatmap representation of crucial graph parameters for all possible windows over the data. Throughout the paper, we use the financial domain to illustrate the key features of the system. However, the analysis tool can also be applied in other domains, as we will illustrate in one of the demonstration scenarios. The tool is online at https://dbs.inf.unibz.it/projects/ismard/, and people can use it with their data at hand. In four demo scenarios we discuss how to use the tool and what can be learned by a data analyst.

2 SYSTEM OVERVIEW

2.1 Architecture

The system is implemented as a web App based on a client-server architecture (cf. Fig. 2). On the client side, a user can upload time series data, set parameters for data pre-processing and graph computations, and inspect the resulting graphs. By tuning the parameters in the client side, the web page will perform AJAX calls and trigger the processing of the data in the server. The computed graph data is then returned to the client and interpreted by visualization components. On the client side, two Javascript

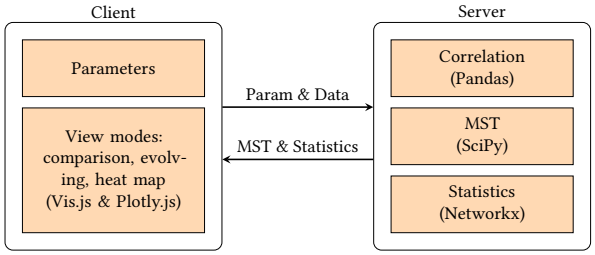


Figure 2: System Architecture

libraries are used for the visualization: VIS.js for graph views and Plotly.js for data reports. On the server side, three Python libraries are employed: Pandas for correlation computation, SciPy for MST computation, and Networkx for graph statistics.

2.2 Data and Parameters

The system accepts time series data in CSV format. The first row specifies the signal names. Each subsequent row stores a time point followed by the signal values for that time point. Missing values are marked with `NaN`. Optionally, a second CSV file containing ⟨signal name, class name⟩-pairs can be uploaded in order to group the time signals into classes.

There are a number of parameters that can be controlled by the user and make the tool flexible for data analytics. For the preparation of the input data, the user can, among others, specify a time granularity (e.g., daily, weekly, or monthly) for the analysis and whether to use the raw data values or the variation of the value w.r.t. the previous time point. The computation of the correlations and the construction of the MST/graph can be controlled by several parameters, including a time range for the analysis, the required minimum number of values for each signal for a meaningful analysis, a minimum overlap between two signals, and a correlation threshold for an edge to be included in the graph visualization. Finally, for the visualization of the analysis results three different views are offered.

2.3 Analysis Workflow

The first step in the analysis workflow is to load the raw data file, store it on the server, and then to optionally apply two *preprocessing* operations. First, the user can specify a time granularity for the analysis which is different from the granularity of the raw data. For instance, in the financial sector the raw data typically contain the daily closing prices, but financial analysts often prefer to use weekly or monthly price values in order to reduce noise. Instead of computing the average, the closing price of Thursday is used as the weekly price, and the price of the last trading day of each month as the monthly price. To support the analysis of data from other domains, the tool also offers the use of the average value in combination with different granularities. Second, the analysis of stock data usually investigates the variation of the price rather than the price itself. Two different variations are frequently used: the return rate, which is the relative price difference to the previous time point, i.e., $r_i = (p_i - p_{i-1})/p_{i-1}$ with p_i being the closing price at time point t_i; and the log return $r_i = (\log p_i - \log p_{i-1})$. The result of the pre-processing phase is stored in the so-called working file on the server.

The next step is to compute the *pairwise correlations* matrix using the Pearson Correlation coefficient. For this, the subsequences of the signals that correspond to the specified window are loaded from the working file into a so-called DataFrame — an efficient two-dimensional main memory data structure of the Pandas library, which natively supports the computation of the Pearson correlation. To obtain robust and reliable results, time series that contain an excessive number of missing values are omitted from the computation of the correlation. Similarly, we do not compute the correlation between two time series if they do not sufficiently overlap in time. Both parameters can be controlled by the user. For the time series that pass these filters, the Pearson Correlation coefficient is computed, which for two time signals x and y with average value $\bar{x}$ and $\bar{y}$, respectively, is defined as $r = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}$ with $-1 \leq r \leq 1$. A value of 1 indicates total positive linear correlation, a value of -1 total negative linear correlation, and a value of 0 no linear correlation. Instead of analyzing the correlation between individual stock signals, the user can decide to investigate the stock sectors (classes), which allows them to gain insights into entire sectors rather than individual stocks. The signal of a class is computed as the average of all stock signals belonging to that class. More advanced aggregation methods are known in the literature, e.g., the free-float adjusted market-capitalization weighting method [10], which computes a weighted average based on the stock shares.

Next, we compute the *minimum spanning tree (MST)* from the correlation matrix using the classical algorithm by Kruskal [7]. Each node represents a stock signal, and the edges are labeled with the correlation coefficient between the connected nodes. As an alternative to the MST, a correlation graph can be used for the visualization. Since a complete graph would not be very useful for analysis purposes, the user can prune weak edges by introducing a threshold for the correlation.

The last step is to compute several *statistics* for the MST/graph. A very important measure for the analysis is the degree distribution of the graph nodes. This measure reflects structural information of a graph and can be used to detect stock market events [3, 6]. Other useful statistics include the range of the actual correlation values and the radius of the MST/graph. Additionally, we also compute the modularity score [8, 20], which allows us to measure whether the stock connections in a MST correlate with the stock sectors. This score is defined as $\sum_{i=1}^{k}(e_{ii} - a_i^2)$, where k is the number of sectors, e_{ii} is the percentage of edges connecting two nodes of the same sector i, and a_i is the percentage of edges for which at least one node belongs to sector i.

2.4 Interaction and Result Visualization

For the visualization of the analysis results, our tool offers three principal viewing modes: comparison mode, evolving graphs mode, and heatmap mode.

The goal of the *comparison mode* is to analyze the data over two different time periods in order to spot similarities and differences, e.g., comparing a normal period with a crisis period. Figure 3 shows a screenshot of this mode, which contains two panels for the two graphs. A scroll bar allows users to select a time window for the analysis. The graphs report the node names (either signal name or class name), and edges are labeled with the correlation coefficients. By clicking on a node, additional information is shown such as the node's eccentricity and class. A color coding is used for the graphs. The node color indicates the class of the represented stock. For the edges, the colors distinguish different levels of correlation: red for high correlations above 0.7, orange for medium correlations between 0.4 and 0.7, and gray for low correlations below 0.4. At the bottom of the screen, the computed statistics are summarized.

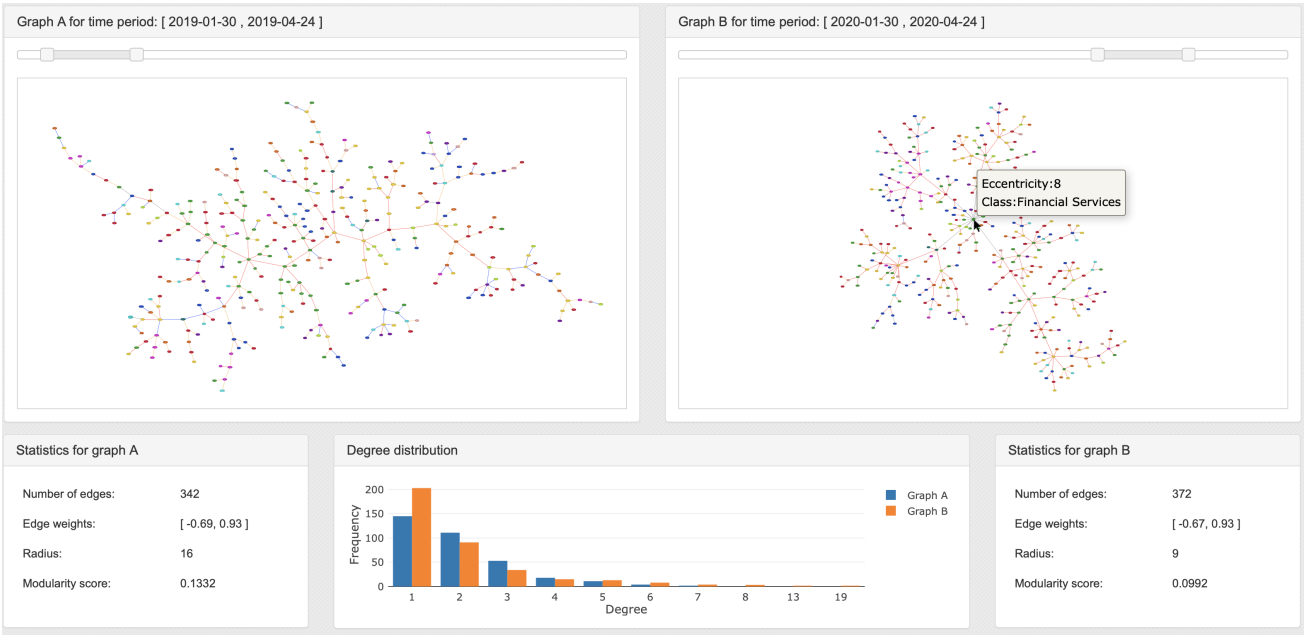


Figure 3: Compare MSTs in ordinary period with COVID crisis period.

The *evolving graph* mode shows the evolution of a graph along the time axis, which helps to understand the subtle changes of graph components over time. For this, a sliding window is used, where the user can specify the window length and the step size. For each window a graph is computed. To highlight the changes between consecutive windows, the new edges w.r.t. the previous graph are shown by dashed lines (cf. Fig. 4).

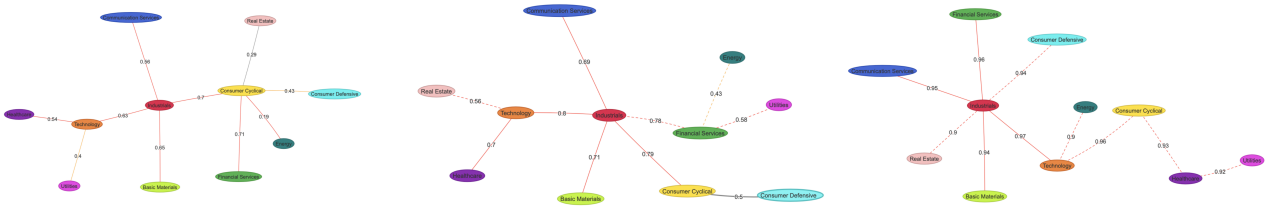
The *heatmap* mode helps to understand the stability and reliability of a MST metric w.r.t. the values of the selected time window. For this, we compute a heatmap for three important graph metrics: first degree score defined as the number of nodes with degree one divided by the total number of nodes; the modularity score; and the radius. Each heatmap shows the metric for all possible windows over the data (cf. Fig. 5). The x-axis represents all possible starting points of time windows, and the y-axis all possible window lengths. Heatmaps allow users to investigate whether the metric value changes drastically with different time windows, as well as discern whether a metric's values are similar with time windows that cover special events in the timeline. The heatmaps are interactive. By clicking on the heatmap, the system computes the MST over the corresponding window. This allows us to inspect the MST in detail. Since the computation of the heatmaps in real time is a computational bottleneck, this feature is currently only activated for the demo dataset. An efficient computation of the heatmaps for large datasets is part of our future work.

3 DEMONSTRATION SCENARIOS

The following four scenarios provide a glimpse of how our tool can be used to systematically investigate behaviors and properties of time series data. We use the Italian stock market data collected from Yahoo Finance. This dataset contains the daily stock prices of 407 companies which traded from January 2019 to June 2020 and includes the COVID stock crash. The companies are classified into 12 market sectors, such as financial service, real estate, energy, etc. In addition, we use a dataset that contains signals from industrial devices provided by a local company.

Comparing Normal and Crisis Periods. In the first demo scenario, an analyst is interested in investigating how stock correlations were affected by the COVID crash. This can be observed by graph shapes and indicators in the comparison mode shown in Fig. 3. The left-hand side shows the MST over a normal period (Jan 2019 – Apr 2019), while the right-hand side shows the MST during the COVID crisis (Jan 2020 – Apr 2020). From the graph shapes, analysts can observe that the crisis led to the formation of many star-like hubs and red-color edges, indicating that a large number of companies were strongly correlated. From the indicators, analysts find that during the COVID crisis the MST had a much smaller radius and a steeper degree distribution compared to the normal period. In particular, significantly more nodes had a degree of 1 during the crisis period. Similar analyses can be performed for other market events. More details on shapes and indicators are described in [6]. Moreover, different parameter values and time windows can be chosen to examine the stability of the above features.

Changes over Time. In this scenario, an analyst wants to investigate whether the correlation between different sectors has changed in proximity to the COVID crash. This type of analysis is supported in the evolution mode. Figure 4 shows the MST computed over three consecutive windows over the aggregated stock signals. The length of the sliding window is three months, and the step size one month. A dashed line indicates a new edge compared to the MST over the previous window. It can be observed that the Industrials sector (red node) tends to be at the center of the tree. Presumably, sectors in Fig. 4c that are connected with Industrials are the most negatively affected sectors by the COVID crash. In contrast, Healthcare (purple) switched its connection from Technology to Consumer Cyclical and Utilities. They could be the positively influenced sectors, since with the medical research and "stay at home" order, these sectors either received more investments or increased consumption.



(a) 10/2019–01/2020 (b) 11/2019–02/2020 (c) 12/2019–03/2020

Figure 4: Evolving graphs.

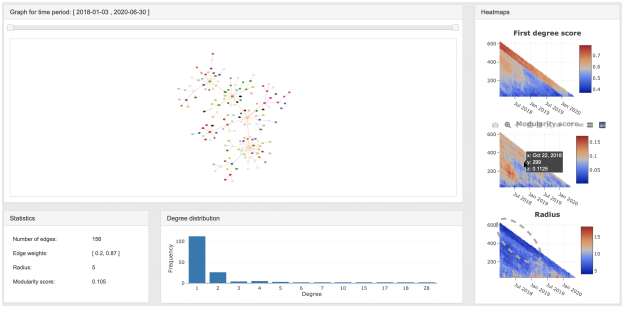


Figure 5: Heatmap view.

Concise Overview of All Windows. A critical aspect for time series analysis is to choose an appropriate window length, owing to the fact that real-world data are often noisy and contain missing values. The heatmap mode helps in this regard as it provides a concise overview of important MST metrics over all possible windows. An example is shown in Fig. 5 with three heatmaps for the first degree score, the modularity score, and the radius of the MST, respectively. It can be observed that for long time windows (i.e., more than 300 time points) the radius is more reliable to signal the crisis period. This is explained by a dark strip at the top border of the heatmap (gray circle in Fig. 5). It appears that for all long time windows that cover the crisis period, the radius of the corresponding MST is significantly lower than the MSTs in normal periods. A similar pattern can be observed for the heatmap of the first degree score. In this scenario, a user learns which are sound indicators for detecting crises and the conditions to use them, e.g., the appropriate window length.

Beyond Stock Data. To show the generality of our tool, we also provide a demonstration scenario for the analysis of data from the industrial sector. The time series come from sensors monitoring components of large industrial printers, such as the temperature of internal CPUs, ink speed, or belt velocity. In this scenario, we show how our tool can be used to understand the interaction of different components, similar to stock sectors, based on their correlation over different time periods.

4 CONCLUSIONS AND FUTURE WORK

In this demo paper, we presented a new web App to investigate stock time series data. The key idea is to compute the pairwise correlations between the data and – in order to facilitate the analysis – to visualize them in a minimum spanning tree, where nodes represent the trading companies and edges show their correlation. With Italian stock market data we demonstrated the effectiveness and versatility of our tool.

Future work points in two directions. First, we will investigate more efficient algorithms for the computation of heatmaps in order to make the tool scalable for larger datasets. The other direction concerns the use of machine learning techniques to detect useful parameter configurations for the analysis automatically.

ACKNOWLEDGMENTS

This work is supported by the projects ISMarD and TASMA, which are funded by the Free University of Bozen-Bolzano.

REFERENCES

[1] Amir Aghasadeghi, Vera Zaychik Moffitt, Sebastian Schelter, and Julia Stoyanovich. 2020. Zooming Out on an Evolving Graph. In *EDBT 2020*. 25–36.
[2] Davide Azzalini, Fabio Azzalini, Mirjana Mazuran, and Letizia Tanca. 2019. Tracking the Evolution of Financial Time Series Clusters. In *DSMM@SIGMOD 2019*. 4:1–4:5.
[3] AQ Barbi and GA Prataviera. 2019. Nonlinear dependencies on Brazilian equity network from mutual information minimum spanning trees. *Physica A: Statistical Mechanics and its Applications* 523 (2019), 876–885.
[4] Xuewei Cao, Yongbin Shi, Penghao Wang, Liujun Chen, and Yougui Wang. 2018. The evolution of network topology structure of Chinese stock market. In *ICBDA 2018*. IEEE, 329–333.
[5] Ricardo Coelho, Claire G Gilmore, Brian Lucey, Peter Richmond, and Stefan Hutzler. 2007. The evolution of interdependence in world equity markets-Evidence from minimum spanning trees. *Physica A: Statistical Mechanics and its Applications* 376 (2007), 455–466.
[6] Paolo Coletti and Maurizio Murgia. 2016. The network of the Italian stock market during the 2008–2011 financial crises. *Algorithmic Finance* 5, 3-4 (2016), 111–137.
[7] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.
[8] Santo Fortunato. 2010. Community detection in graphs. *Physics reports* 486, 3-5 (2010), 75–174.
[9] Raphael H Heiberger. 2014. Stock network stability in times of crisis. *Physica A: Statistical Mechanics and its Applications* 393 (2014), 376–381.
[10] S&P Dow Jones. 2014. Index mathematics methodology.
[11] Kyoung-jae Kim. 2003. Financial time series forecasting using support vector machines. *Neurocomputing* 55, 1-2 (2003), 307–319.
[12] Ping Luo, Kai Shu, Junjie Wu, Li Wan, and Yong Tan. 2020. Exploring Correlation Network for Cheating Detection. *ACM Trans. Intell. Syst. Technol.* 11, 1 (2020), 12:1–12:23.
[13] Abdullah-Al Mamun and Sanguthevar Rajasekaran. 2016. An efficient Minimum Spanning Tree algorithm. In *ISCC 2016*. 1047–1052.
[14] Rosario N Mantegna. 1999. Hierarchical structure in financial markets. *Eur. Phys. J. B* 11, 1 (1999), 193–197.
[15] Gautier Marti, Sébastien Andler, Frank Nielsen, and Philippe Donnat. 2016. Clustering Financial Time Series: How Long Is Enough?. In *IJCAI 2016*. 2583–2589.
[16] Salvatore Micciché, Giovanni Bonanno, Fabrizio Lillo, and Rosario N Mantegna. 2003. Degree stability of a minimum spanning tree of price return and volatility. *Physica A: Statistical Mechanics and its Applications* 324, 1-2 (2003), 66–73.
[17] Daniel Petrov, Rakan Alseghayer, Mohamed A. Sharaf, Panos K. Chrysanthis, and Alexandros Labrinidis. 2017. Interactive Exploration of Correlated Time Series. In *ExploreDB 2017*. 2:1–2:6.
[18] Francisco Serdio, Edwin Lughofer, Kurt Pichler, Thomas Buchegger, Markus Pichler, and Hajrudin Efendic. 2014. Fault detection in multi-sensor networks based on multivariate time-series models and orthogonal transformations. *Information Fusion* 20 (2014), 272–291.
[19] M Wiliński, A Sienkiewicz, Tomasz Gubiec, R Kutner, and ZR Struzik. 2013. Structural and topological phase transitions on the German Stock Exchange. *Physica A: Statistical Mechanics and its Applications* 392, 23 (2013), 5963–5973.
[20] Lisi Xia, Daming You, Xin Jiang, and Quantong Guo. 2018. Comparison between global financial crisis and local stock disaster on top of Chinese stock network. *Physica A: Statistical Mechanics and its Applications* 490 (2018), 222–230.
[21] G Peter Zhang and Min Qi. 2005. Neural network forecasting for seasonal and trend time series. *Eur. J. Oper. Res.* 160, 2 (2005), 501–514.