Proceedings Paper:
Ye, Cheng, Wilson, Richard C. orcid.org/0000-0001-7265-3033 and Hancock, Edwin R. orcid.org/0000-0003-4496-2028 (2016) Correlation Network Evolution Using Mean Reversion Autoregression. In: Structural, Syntactic, and Statistical Pattern Recognition - Joint IAPR International Workshop, S+SSPR 2016, Mérida, Mexico, November 29 - December 2, 2016, Proceedings. Lecture Notes in Computer Science . , pp. 163-173.

https://doi.org/10.1007/978-3-319-49055-7_15

Correlation Network Evolution Using Mean Reversion Autoregression

Cheng Ye, Richard C. Wilson and Edwin R. Hancock

Department of Computer Science,
University of York,
York, YO10 5GH, UK
Email: {cy666, richard.wilson, edwin.hancock}@york.ac.uk

Abstract. In this paper, we present a new method for modeling time-evolving correlation networks, using a Mean Reversion Autoregressive Model, and apply this to stock market data. The work is motivated by the assumption that the price and return of a stock eventually regresses back towards their mean or average. This allows us to model the stock correlation time-series as an autoregressive process with a mean reversion term. Traditionally, the mean is computed as the arithmetic average of the stock correlations. However, this approach does not generalize the data well. In our analysis we utilize a recently developed generative probabilistic model for network structure to summarize the underlying structure of the time-varying networks. In this way we obtain a more meaningful mean reversion term. We show experimentally that the dynamic network model can be used to recover detailed statistical properties of the original network data. More importantly, it also suggests that the model is effective in analyzing the predictability of stock correlation networks.

Keywords: time-evolving correlation network, Mean Reversion Autoregressive Model, generative probabilistic model

1 Introduction

Generally speaking, a correlation network is the diagrammatic representation of a complex system architecture, where the vertices in the network represent the system components and where the edges contain the connection and correlation information between components. It is for this reason that correlation networks play an increasingly critical role in observing, analyzing and predicting the structure, function and dynamics of realistic large-scale systems. For example, in a stock market there exist a large number of distinct relationships between economic components. By adopting appropriate filtration methods, the most influential correlations can be preserved for constructing the financial market correlation network, which is used for further statistical analyses [1].

Although the bulk of existing correlation network analysis is concerned with static networks, most networks are in reality dynamic in nature [2]. Motivated by

the need to understand more deeply the network dynamics, this paper presents a new method for modeling time-evolving correlation networks, and applies the resulting method to stock market data. Experimental results show the network model reflects detailed statistical properties of the original network data and more importantly, it can be used to analyze the predictability of stock correlation networks.

1.1 Related Literature

Until recently, one fundamental field of graph theory that has broad applications in network analysis, which has received only marginal attention, is evolutionary graph theory. In fact, many real-world complex systems such as citation networks, communications networks, neural networks and financial networks give rise to structures that change with time. For instance, networks grow and evolve with the addition of new components and connections, or the rewiring of connections from one component to another [3]. In order to analyze such time-evolving graphs, efficient tools for understanding and modeling their time-dependent structure and function are required.

In general, graph evolution can be approached from both macroscopic and microscopic perspectives [4]. On the one hand, the macroscopic approach aims at studying how the global parameters of a dynamic graph evolve from one epoch to another. This can be accomplished by directly employing a number of graph characterizations that have been developed for static graphs to each epoch, and then analyzing the time evolution of these characterizations. For instance, it has been demonstrated that the subgraph centrality can be interpreted as a partition function of a network [5], and as a result the entropy, internal energy and the Helmholtz free energy can be computed and shown to be intimately related to the network dynamics. On the other hand, at the microscopic level, it is the birth-death dynamics of an individual vertex or edge in the graph evolution that are under study. Based on this observation, Grindrod and Higham [4] have introduced a tractable framework for modeling evolving graphs. To do this, they propose a novel range-dependent birth-death mechanism, which allows a variety of evolutionary behaviours to be modeled. The resulting dynamic graph model is set up as a discrete-time Markov chain, and an analogous continuous-time framework can also be developed. This model has proved to be efficient in investigating the evolutionary processes that take place for evolving graphs.

This paper centers on the modeling of time-varying correlation networks. In general, the modeling of the correlation time-series between a pair of components can be achieved using both stochastic and non-stochastic approaches. In particular, stochastic modelling has been widely considered as an essential tool in the analyses of finance, biology and other areas, too. A commonly used approach is to use correlated stochastic processes to map the relationships between components in the financial or biological systems. For example, in the finance literature it is well known that the stochastic process modeling plays a vital role in pricing and evaluation of financial derivatives [6]. On the other hand, the non-stochastic approach also provides a powerful tool for modeling time-series

of component correlations. One famous example is furnished by modeling the stock log-returns as autoregressive processes with random disturbances, such as the AR-GARCH model and EGARCH model [7].

1.2 Outline

The remainder of the paper is structured as follows. Section II details the development of the time-evolving correlation network modeling using Mean Reversion Autoregression. In this section we also show how the mean reversion term can be obtained from a recently developed generative probabilistic model for graph time-series analysis. In Sec. III, we show the effectiveness of our method by exploring its experimental performance on realistic stock market data. Finally, Sec. IV summarizes our contribution present in this paper and points out future research directions.

2 Time Evolving Correlation Network Model

In this section, we provide the detailed development of a model for time-evolving correlation networks. To commence, we introduce an autoregressive model that contains a mean reversion term and use this to simulate the pairwise correlation time-series. Applying such an analysis to each possible pair of vertices in the network, we have to hand a rule that governs the evolution of the edge weight change of the dynamic network. The idea behind the mean reversion model is based on the fact that the log-return of stock price always regresses back to its mean or average. Traditionally, the mean is computed as the arithmetic mean of the stock log-returns. However, this approach clearly does not capture the essential properties of the data. In order to overcome this problem, we show how a generative probabilistic model can be used to determine a more meaningful mean reversion term for the autoregressive model. In short, this generative model provides a structure called "supergraph" which can be used to best summarize the structural variations present in the set of network data.

2.1 Mean Reversion Autoregressive Model

In general, a stochastic process is a sequence of measurements representing the evolution of random variables over time. An autoregressive (AR) model represents a type of stochastic processes in which the value is linearly dependent on its previous value and on a stochastic term. Mathematically, let $Y_t = \{y_1, y_2, \cdots, y_t, \cdots\}$ represent a stochastic process of measurements y in time period $[1, 2, \cdots]$, the first-order AR model (AR(1)) implies that

$$y_t = \theta_0 + \theta_1 y_{t-1} + \epsilon_t,$$

where θ_1 is the parameter of the model, θ_0 is a constant and ϵ_t is the white noise. More generally, the p-th order AR model (AR(p)) gives that

$$y_t = \theta_0 + \sum_{i=1}^{p} \theta_i y_{t-i} + \epsilon_t$$

where θ_i represent the parameters of y_{t-i}. In our analysis, we consider the simple case, i.e., AR(1) process in order to reduce the number of parameters in the model.

In economics, the concept of mean reversion has proved to be a widely held belief, i.e., the stock return is likely to regress towards the mean value after a long period. Based on this observation, we add a mean reversion term to the standard AR(1) model in order to obtain the Mean Reversion Autoregressive Model (MRAM)

$$y_t - \bar{y} = \theta_1(y_{t-1} - \bar{y}) + \epsilon_t, \tag{1}$$

where $\bar{y}$ is the mean value of y and $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is the white noise. Clearly, the mean reversion term $\bar{y}$ plays a critical role in the MRAM. Broadly speaking, there are a number of different ways to define the mean reversion term. One example would be to simply use the arithmetic mean of y_t, but this approach cannot represent the full, underlying properties of the time-series.

2.2 Generative Model Learning

In the following we present a novel method for constructing a generative model to analyze the structure of labeled data and use this model to determine a more meaningful measure for representing $\bar{y}$. Let $\mathbf{G} = \{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_t, \cdots, \mathcal{G}_N\}$ represent the time-series graph dataset under study, and $\mathcal{G}_t$ is used to denote the t-th sample graph in the time-series. The generative model, or the supergraph, which we aim to learn from the sample data is denoted by $\tilde{\mathcal{G}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{E}})$, with vertex set $\tilde{\mathcal{V}}$ and edge set $\tilde{\mathcal{E}}$.

We are dealing with labeled graphs. Each vertex in a network has a unique label. In our application involving the New York Stock Exchange data, there are stocks trading in the New York Stock Exchange market. The vertex indices are denoted by lower-case letters including u, v, a, b, α and β, and will interchange these vertex indices with the vertex labels.

We represent the connectivity structure of the sample graph $\mathcal{G}_t$ using a weighted adjacency matrix W^t whose (u, v)-th entry W^t_{uv} indicates the connectivity between vertices u and v in the graph, and clearly, we have $W^t_{uv} \in [0, 1]$. Similarly, we use the matrix $\tilde{W}$ to represent the structure of the supergraph $\tilde{\mathcal{G}}$.

Having introduced the necessary formalism, we now proceed to develop the probabilistic framework for the generative model learning method. To commence, we require the posterior probability of the observed sample graphs given the structure of the generative model $p(\mathbf{G}|\hat{\mathcal{G}})$. Then, the problem of finding the optimal supergraph can be posed in terms of seeking the structure $\tilde{\mathcal{G}}$ that satisfies the condition

$$\tilde{\mathcal{G}} = \underset{\hat{\mathcal{G}}}{\operatorname{argmax}}\, p(\mathbf{G}|\hat{\mathcal{G}}).$$

We follow the standard approach to constructing the likelihood function, which has been previously used in [8][9]. This involves factorizing the likelihood function over the observed data graphs and making use of the assumption that each

individual edge in the sample graphs is conditionally independent of the remainder, given the structure of the supergraph. As a result, we have

$$p(\mathbf{G}|\tilde{\mathcal{G}}) = \prod_t p(\mathcal{G}_t|\tilde{\mathcal{G}}) = \prod_t \prod_u \prod_v p(W_{uv}^t|\tilde{W}_{uv}), \tag{2}$$

where $t = 1, 2, \cdots, N$. Moreover, $p(W_{uv}^t|\tilde{W}_{uv})$ is the probability that the connectivity between u and v in the sample graph $\mathcal{G}_t$ is equal to W_{uv}^t, given that the edge (u, v) in the supergraph $\tilde{\mathcal{G}}$ has connectivity $\tilde{W}_{uv}$. To proceed, we model the distribution $p(W_{uv}^t|\tilde{W}_{uv})$ by adopting a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ of the connection weights whose mean is the weight for the edge (u, v) in the supergraph, i.e., $\mu = \tilde{W}_{uv}$. With the observation density model to hand, we write

$$p(W_{uv}^t|\tilde{W}_{uv}) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(W_{uv}^t - \tilde{W}_{uv})^2/2\sigma^2}.$$

To locate the optimal supergraph, we adopt an information theoretic approach and use a two-part minimum description length (MDL) criterion. Underpinning MDL is the principle that the best hypothesis for a given set of data is the one that leads to the shortest code length of the observed data. To formalize this idea, we encode and transmit the data $\mathcal{G}_t$ together with the hypothesis $\tilde{\mathcal{G}}$, leading to a two-part message whose total length is given by

$$\mathcal{L}(\mathbf{G}, \tilde{\mathcal{G}}) = \mathcal{L}(\mathbf{G}|\tilde{\mathcal{G}}) + \mathcal{L}(\tilde{\mathcal{G}}),$$

where $\mathcal{L}(\mathbf{G}|\tilde{\mathcal{G}})$ is the code length of the data graphs given the supergraph and $\mathcal{L}(\tilde{\mathcal{G}})$ is the code length of the estimated supergraph. Determining the most likely supergraph structure can be viewed as seeking the one that minimizes the total code length of the likelihood function. To this end, we take into account the total code length and apply the MDL principle to the model, this allows us to construct a supergraph representation that trades off goodness-of-fit with the sample graphs against the complexity of the model.

To apply the two-part MDL principle, we commence by computing the code length of the data graphs given the supergraph. This can be achieved by simply using the average of the negative logarithm of the likelihood function, with the result that

$$\begin{aligned}\mathcal{L}(\mathbf{G}|\tilde{\mathcal{G}}) &= -\frac{1}{N} \ln p(\mathbf{G}|\tilde{\mathcal{G}}) \\ &= -\frac{1}{N} \sum_t \sum_u \sum_v \left\{ \ln \frac{1}{\sqrt{2\pi}\sigma} - \frac{(W_{uv}^t - \tilde{W}_{uv})^2}{2\sigma^2} \right\},\end{aligned} \tag{3}$$

where N is the length of the observed time-series data $\mathbf{G}$.

Next, we compute the code length of the supergraph structure. Traditionally, the complexity of a model is measured by counting the number of parameters in the model. However, this does not generalize well for graphs since the true graph complexity cannot be accurately reflected by information such as the numbers of vertices or edges in the graph. To overcome this problem, we adopt a more

meaningful measure of graph complexity, namely the von Neumann entropy, to encode the complexity of the supergraph structure (see [10] and [11] for detailed information of this entropy). Then, we have the supergraph complexity code length as follows,

$$\mathcal{L}(\tilde{\mathcal{G}}) = 1 - \frac{1}{|\tilde{\mathcal{V}}|} - \frac{1}{|\tilde{\mathcal{V}}|^2} \sum_{(u,v)\in\tilde{\mathcal{E}}} \frac{\tilde{W}_{uv}}{w_u w_v}, \tag{4}$$

where $w_u = \sum_{(u,v)\in\tilde{\mathcal{E}}} \tilde{W}_{uv}$ is the weighted degree of vertex u, which is defined as the sum of the connectivity weights of the edges connected to u and w_v is similarly defined. In effect, the complexity of the supergraph depends on two factors. The first is the order of the supergraph, i.e., the number of the vertices while the second is based on the degree statistics of the vertices in the supergraph.

To recover the supergraph we must optimize the total code length criterion, which can be computed by adding together the two contributions to the total code length, with respect to the weighted adjacency matrix $\tilde{W}$. This can be done in a number of ways. These include gradient descent and soft assign [12]. However here we use a simple fixed-point iteration scheme. We compute the partial derivative of the code length criterion $\mathcal{L}(\mathbf{G}|\tilde{\mathcal{G}})$ given in Eq. (3) with respect to the elements of the weighted adjacency matrix $\tilde{W}_{ab}$. After some analysis the required derivative is

$$\begin{aligned}\frac{\partial\mathcal{L}(\mathbf{G},\tilde{\mathcal{G}})}{\partial\tilde{W}_{ab}} = \frac{1}{N\sigma^2}\sum_t(\tilde{W}_{ab} - W^t_{ab}) - \frac{1}{|\tilde{\mathcal{V}}|^2}\Big\{\frac{1}{w_a w_b} \\ -\frac{1}{w_a^2}\sum_{(a,\beta)\in\tilde{\mathcal{E}}}\frac{\tilde{W}_{a\beta}}{w_\beta} - \frac{1}{w_b^2}\sum_{(\alpha,b)\in\tilde{\mathcal{E}}}\frac{\tilde{W}_{\alpha b}}{w_\alpha}\Big\}.\end{aligned} \tag{5}$$

where β denote the neighbour vertices of a and α are the neighbours of b.

To set up our fixed-point iteration scheme, we set the above derivative to zero, and re-organize the resulting equation to obtain an update process of the form $\tilde{W}_{ab} \rightarrow g(\tilde{W}_{ab})$, where $g(\cdots)$ is the iteration function. There is of course no unique way of doing this, and for convergence the iteration function $g(\tilde{W}_{ab})$ must have a derivative of magnitude less then unity at the fixed point corresponding to the required solution. One such scheme is

$$\begin{aligned}\tilde{W}_{ab} \rightarrow \frac{1}{N\sigma^2}\sum_t W^t_{ab} + \frac{1}{|\tilde{\mathcal{V}}|^2}\Big\{\frac{1}{w_a w_b} - \frac{1}{w_a^2} \\ \sum_{(a,\beta)\in\tilde{\mathcal{E}}}\frac{\tilde{W}_{a\beta}}{w^{(n)}_\beta} - \frac{1}{w_b^2}\sum_{(\alpha,b)\in\tilde{\mathcal{E}}}\frac{\tilde{W}_{\alpha b}}{w_\alpha}\Big\}.\end{aligned} \tag{6}$$

The update process is governed by two terms. The first is computed from the local windowed mean of the time-series $\frac{1}{N\sigma^2}\sum_t W^t_{ab}$, while the second term is a step away from the local time-series mean determined by the partial derivative of the von Neumann entropy. This latter update term depends on the local pattern of vertex degrees.

Finally, with the generative structure to hand, we have the MRAM for the time-evolving correlation network. Mathematically, the edge weights of edge (u, v) in the networks follow the process

$$W_{uv}^{t} - \tilde{W}_{uv} = \theta_1^{uv}(W_{uv}^{t-1} - \tilde{W}_{uv}) + \epsilon_t^{uv} \tag{7}$$

where θ_1^{uv} and ϵ_t^{uv} are the parameter and white noise of edge (u, v).

3 Experiments

In this section, we evaluate the proposed time-evolving correlation network model by applying the model to the stock market data. We confine our attention to two main tasks, the first is to explore whether the MRAM can be used to reflect similar statistical properties of the original correlation time-series; the second is to analyze how the predictability of the stock network changes between different time periods. To commence, we give a brief introduction of the dataset used in the experiments.

NYSE Stock Market Network Dataset. Is extracted from a database consisting of the daily prices of 3799 stocks traded on the New York Stock Exchange (NYSE). This data has been well analyzed in [13], which has provided an empirical investigation studying the role of communities in the structure of the inferred NYSE stock market. The authors have also defined a community-based model to represent the topological variations of the market during financial crises. Here we make use of a similar representation of the financial database. Specifically, we employ the correlation-based network to represent the structure of the stock market since many meaningful economic insights can be extracted from the stock correlation matrices [14]. To construct the dynamic network, 347 stocks that have historical data from January 1986 to February 2011 are selected. Then, we use a time window of 20 days and move this window along time to obtain a sequence (from day 20 to day 6004) in which each temporal window contains a time-series of the daily return stock values over a 20-day period. We represent trades between different stocks as a network. For each time window, we compute the cross-correlation coefficients between the time-series for each pair of stocks. This yields a time-varying stock market network with a fixed number of 347 vertices and varying edge structure for each of 5976 trading days.

In the first experiment, we randomly select two pairs of stocks from the *NYSE Stock Market Network Dataset* and apply the MRAM to their cross-correlation time-series for the entire period, in order to explore whether the model simulation is effective in recovering the statistical properties of the real data. Specifically, we use Eq. (7) to model the correlation time-series of stocks u and v with the mean reversion term determined by the supergraph structure $\tilde{W}_{uv}$. Then, we estimate the model parameters θ_1^{uv} and σ_t^{uv}, which is used to compute the noise term ϵ_t^{uv}. This allows us to obtain a simulation process whose start value is the same as the real correlation time-series, using the parameters we have estimated from the real time-series in the *NYSE Stock Market Network Dataset.*

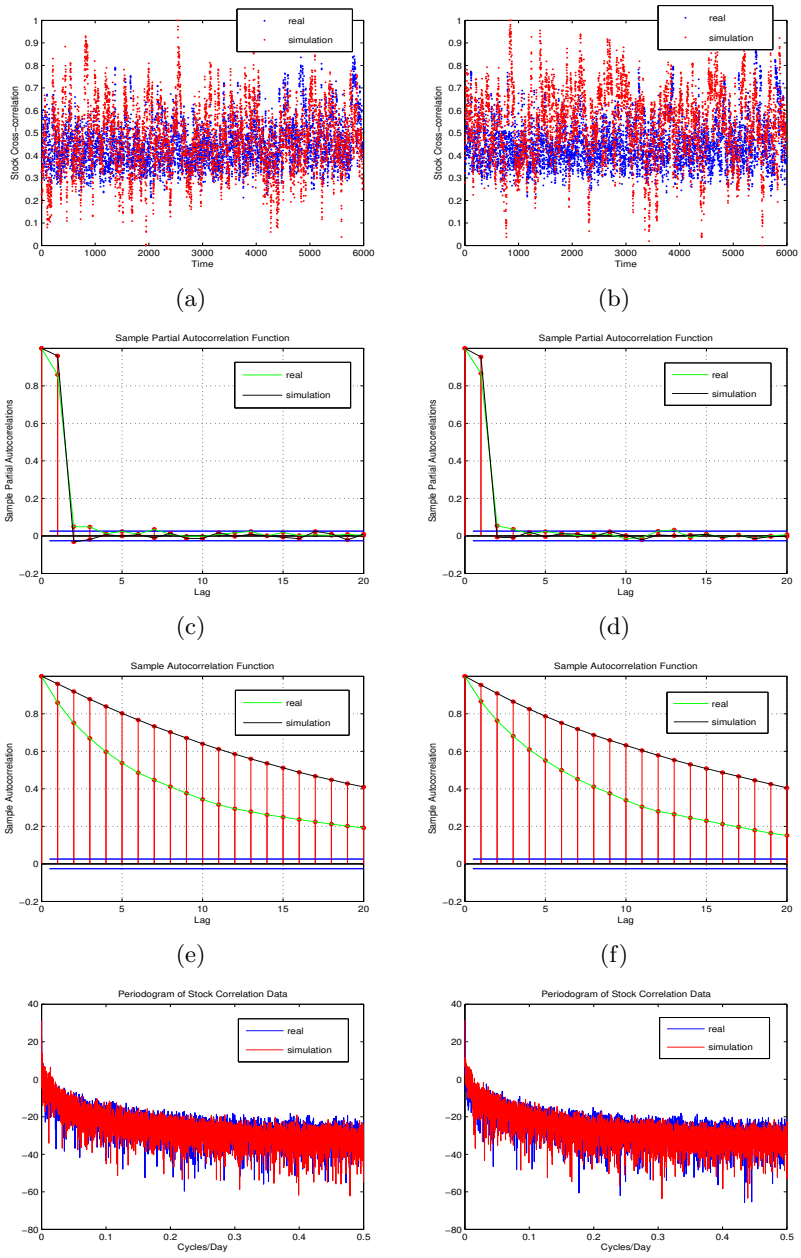

Fig. 1. Comparison of cross-correlation time-series of different pairs of stocks of real data and simulation data.

Figure 1 shows a comparison of the real data and the model simulation for the stock correlation time-series of two pairs of stocks. The statistical properties under study include the sample partial autocorrelation function, cross-autocorrelation function and the periodicity. Clearly, it is difficult to evaluate the performance of the proposed model from the top plots in Fig. 1, as there is no clear correlation between the real data and the simulation data. However, from the rest of the plots, we observe that the model is able to follow the statistical properties of the real data, especially the partial autocorrelation function and the periodicity. Also from the partial autocorrelation function plots, the values of both real data and simulation data significantly decrease after the first lag, which implies that the choice of using AR(1) process to model correlation time-series is plausible.

The second experimental goal is to analyze the predictability of the financial network, i.e., to explore whether the MRAM can be used to help determine whether the structure of a network is predictable or not. In Fig. 2 we plot the von Neumann entropy for both real data graphs (blue solid line) and model graphs (magenta dot-line) for the time period from year 2000 to 2011. Before year 2007, the von Neumann entropy for realistic graphs is relatively stable, implying that the stock network structure does not experience significant structural changes. After 2007, however, the entropy curve witnesses a number of dramatic fluctuations, which means that the network structure is extremely unstable. Turning attention to the simulation data, we note its von Neumann entropy curve clearly exhibits the similar behaviour, following the trend of that of the real data. This again illustrates that the proposed autoregressive model provides an effective method for modeling the time-evolving correlation network in terms of reflecting the structural properties of real data. Moreover, although our model does not offer a way to predict the future network structure, it is indeed useful in understanding and determining the predictability of time-evolving networks.

4 Conclusion

To summarize, in this paper we present a new method for modeling the time-evolving correlation networks, using a Mean Reversion Autoregressive Model. The idea is motivated by the fact that in the finance literature, autoregressive processes are widely employed to model the pricing and evaluation of financial derivatives. Moreover, based on the assumption that stock log-returns always regress back towards the mean or average, we introduce a mean reversion term to the autoregressive model. The mean reversion terms clearly plays a key role in the effectiveness of the model, so it is imperative to have to hand an efficient measure for this term. To this end, we turn our attention to a recently developed generative model, which has been used to analyze the underlying structure of graph time-series, and use this structure to determine a meaningful mean reversion term. In the experiments, we apply the method to realistic stock market data and evaluate the properties of the proposed model.

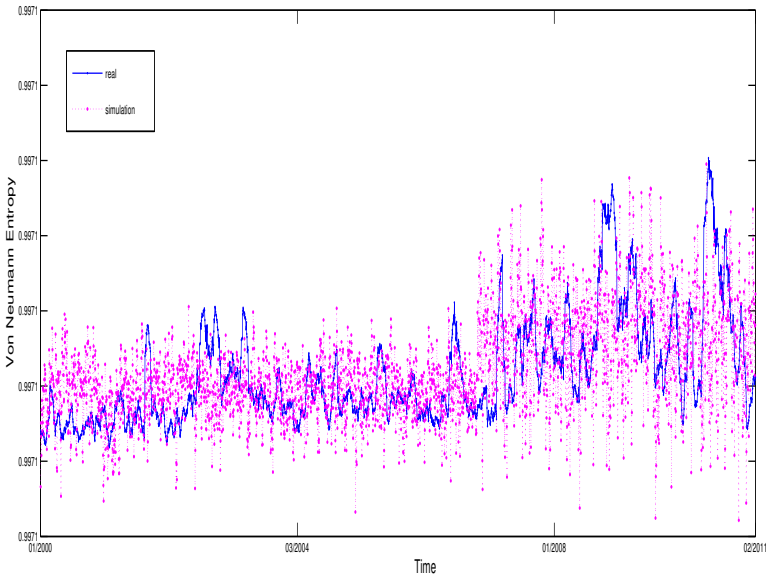

Fig. 2. Comparison of von Neumann entropy for real data and simulation data graphs for time period from 2000 to 2011.

In the future, the work reported in this paper can be extended in a number of ways. For instance, it would be interesting to explore how the stochastic processes can be used to model the correlation dynamics, which can help develop more efficient time-evolving network models. Moreover, we would be interested in investigating the relationship between the model parameters we have optimized and the network topological characteristics. With this knowledge to hand, we would be able to understand how the network topology influences its structural evolution.

References

1. Tumminello, M., Aste, T., Di Matteo, T., Mantegna, R.N.: A tool for filtering information in complex systems. Proceedings of the National Academy of Sciences **102** (2005) 10421–10426
2. Albert, R., Barabási, A.L.: Statistical mechanics of complex networks. Reviews of Modern Physics **74**(1) (2002) 47–97
3. Albert, R., Barabási, A.L.: Topology of evolving networks: Local events and universality. Physical Review Letters **85**(24) (2000) 5234–5237
4. Grindrod, P., Higham, D.J.: Evolving graphs: dynamical models, inverse problems and propagation. Proceedings of the Royal Society A **466** (2009) 753–770

5. Estrada, E., Hatano, N.: Statistical-mechanical approach to subgraph centrality in complex networks. Chemical Physics Letters **439** (2007) 247–251
6. Christie, A.A.: The stochastic behavior of common stock variances: Value, leverage and interest rate effects. Journal of Financial Economics **10** (1982) 407–432
7. Ferenstein, E., Gasowski, M.: Modelling stock returns with ar-garch processes. Statistics and Operations Research Transactions **28**(1) (2004)
8. Luo, B., Hancock, E.R.: Structural graph matching using the em algorithm and singular value decomposition. Pattern Analysis and Machine Intelligence, IEEE Transactions on **23** (2001) 1120–1136
9. Han, L., Wilson, R., Hancock, E.: Generative graph prototypes from information theory. IEEE Transactions on Pattern Analysis and Machine Intelligence **37**(10) (2015)
10. Han, L., Escolano, F., Hancock, E., Wilson, R.: Graph characterizations from von neumann entropy. Pattern Recognition Letters **33** (2012) 1958–1967
11. Ye, C., Wilson, R.C., Comin, C.H., Costa, L.d.F., Hancock, E.R.: Approximate von neumann entropy for directed graphs. Physical Review E **89**(052804) (2014)
12. Gold, S., Rangarajan, A.: A graduated assignment algorithm for graph matching. IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(4) (1996) 377388
13. Silva, F.N., Comin, C.H., Peron, T.K.D., Rodrigues, F.A., Ye, C., Wilson, R.C., Hancock, E.R., Costa, L.d.F.: On the modular dynamics of financial market networks. ArXiv e-prints (1501.05040) (2015)
14. Battiston, S., Caldarelli, G.: Systemic risk in financial networks. Journal of Financial Managements Markets and Institutions **1** (2013) 129–154