Time Series Trend Detection and Forecasting Using Complex Network Topology Analysis

Leandro Anghinoni
Department of Computing and Mathematics
Faculty of Philosophy, Sciences and Literature
University of São Paulo
Ribeirão Preto, Brazil
anghinoni@usp.br

Liang Zhao
Department of Computing and Mathematics
Faculty of Philosophy, Sciences and Literature
University of São Paulo
Ribeirão Preto, Brazil
zhao@usp.br

***Abstract*—Extracting knowledge from time series analysis has been growing in importance and complexity over the last decade as the amount of stored data has increased exponentially. Considering this scenario, new data mining techniques have continuously developed to deal with such a situation. In this paper, we propose to study time series based on its topological characteristics, observed on a complex network generated from the time series data. Specifically, the aim of the proposed model is to create a trend detection algorithm for stochastic time series based on community detection and network metrics. It is expected that the proposed model presents some advantages over traditional time series analysis, such as adaptive number of classes with measurable strength and better noise absorption. Experimental results on the Bovespa index (Brazilian stock market) trend prediction shows that the proposed technique is promising.**

Keywords—time series, trend detection, complex networks, community detection.

I. Introduction

Time series analysis plays an important role in many different study fields, such as economics, medicine, biology and many others. A time series can be defined as a sequence of observations collected over regular time intervals. In a time series the order of the collected data is important, so that the observed process makes sense [1]. Its study has great relevance considering three aspects:

- Description: a time series describes the behavior of a process over time.
- Understanding: with the process description it is possible to observe some process individualities, like trends, seasonality, cycles and correlations.
- Forecasting: once the process is understood, one can propose a model to forecast its future behavior.

Data mining on time series can be approached on several different ways. In [1] many techniques are described in order to deal with time series composed of large datasets. The general model for time series analysis is composed of four steps: problem definition, data pre-processing, model building and, finally, model analyzing and prediction.

The main dataset used in this paper is composed of the closing price of the Brazil´s benchmark Bovespa index over the last 20 years. The Bovespa index has been the most important indicator to measure the perfomance of the brazilian open capital companies for over 50 years. Its metodology was first created in 1962 and, after some adjustments in 1966 and 1967, the index was adopted by the São Paulo Stock Exchange in 1968. Ever since, the index has gone through periods of stability, but also crashes and bubbles [2]. In the point of view of an investor or a market analyst, being able to forsee these market reversals is of great value.

However, the efficient market hypothesis states that markets are efficient in relation to information, i.e, an investor cannot outperform the market in the long term using only the public information available [3]. There are three versions of the hypothesis:

- Weak hypothesis considers that that the price reflects all the historical information available about the asset.
- Semi-strong hypothesis considers that the price reflects all the historical information available about the asset and that any new public information is reflected instantly at the current price.
- Strong hypothesis considers that the price reflects all the historical information available about the asset and that any new information, even privileged information, is reflected instantly at the current price.

Since the hypothesis was created, and specially after the crash of 2008, a lot of criticism has been held against this theory in the grounds that a lot of privileged information is not reflected in the price. A method that could predict the trend of the market based on historical prices would, in a way, go against the strong hypothesis.

The dataset can be approached as a stochastic process, with no deterministic function. That presents several challenges when creating a model to analyse the series, such as the concept of

short, mid and long terms and how to separate noise from real changes in the trend.

In this paper we propose a framework to classify and predict the time series trend based on a complex network generated from the time series characteristics. The network is generated from the original time series, respecting the data temporal order, and the data is classified using community detection techniques. In the proposed model this is done in an unsupervised manner, so that the number of classes and characteristics of each class are not known or set beforehand. Once the data is classified, it is analysed using complex network metrics, such as average degree and the centrality of a node. Finnaly, a prediction model based on these concepts is proposed.

The remainder of this paper is structured as follows: Sec.2 presents an overview of the related work; Sec.3 describes the datasets that will be used to test the proposed model; Sec.4 describes the motivations and objectives of the proposed model; Sec.5 describes the proposed model in details; Sec.6 shows some experimental results on the datasets and Sec.7 presents the final conclusions about this work.

II. Related Work

Stock time series prediction can be divided into two types: value forecasting and trend forecasting [4]. Value forecasting is a regression task and its most known melthod is the ARIMA (Auto Regressive Integrated Moving Average) model. This model outputs a value prediction within a confidence margin, based on its past observations. A lot of work has been dedicated to it with several variants of the original model, such as in [5],[6] and [7]. However, the ARIMA model does not capture nonliar patterns easily and, because of that, some recent works have proposed the combination of the ARIMA process with modern techniques, like suport vector machines (SVM) [8] and neural networks [9].

Trend forecasting, however, is a classification task and, therefore, classification models, like Naive Bayes, MLP, Decision Trees and so on can be selected for the prediction. In order for the model to predict right, the dataset has to be labeled right, what imposes a great challenge by itself.

In [4] six classification models (DET, NB, SVM, MLP, RNN and LSTM) are tested in the stock data of a company listed in China. Results are presented and although a difference in performance can be noticed among the models, no model reaches a high level of accurary (all of them fall around 50%), leading the paper to conclude that the market is unpredictable. One of the reasons may be the fact that to test any of these models the data was labeled for training using a pre-define rule.

Labeling the trend on a stock time series is a very subjective task. In [4] three classes were created, UP, DOWN and FLAT and the time series was labeled according to the intra-day movemet of the following day, i.e, UP if the closing price was higher than the opening price, DOWN if the closing price was lower than the opening price and FLAT if it closed unchanged. The accuracy was then measured on a daily basis. This approach requires the model to be extremely precise during the prediction step, since it has be right on every daily movement.

III. Motivations

Complex networks are in the heart of complex systems due to its interdisciplinary, quantitative, mathematical and computational nature [10]. Complex networks, therefore, play an important role in the modern world, allowing the study of complex systems, where the correlation between data is sometimes hard to spot. One of the salient features of complex networks is the presence of community structure where groups of elements are densely connected, indicating that they share the same properties. Over the last years, some community detection models have been proposed. These algorithms can group nodes that are more connected to nodes in the same community than to nodes outside the community. Since all the nodes in the same community are similar to each other in some extent, one can use this information to reduce huge amounts of data to the number of communities on a complex network.

Community detection algorithms can be classified as agglomerative ("bottom-up") or divisive ("top-down"). In the agglomerative technique each node is considered to be a community and then the communities are merged taking into account the modularity measure. In the divisive technique the whole network is considered to be one community and the less connected nodes are disconnected until more dense communities are detached from the former community [11]. The algorithm proposed in [12] is a divisive one, that works by calculating the path between every node in the network and removing the most used links. The algorithm proposed in [13] is an agglomerative one, that joins nodes that cause the greater increase in the modularity Q of the network. The nodes are joined until the modularity stops increasing. This is the technique used in the proposed model.

More recently some new techniques have been developed, such as the one presented in [14], that uses particle competition to define the community of the node and the one presented in [15] that uses the degree of the node to define its participation on a certain community. Both methods consider the fact that a node can participate on more than one community in a lower or higher degree.

The model proposed in this paper tries to solve some of the issues presented in the previous sections. It sets out to create a classification and prediction method based on community detection on a complex network derived from the stock time series. The proposed model is similar to the one presented in [11], however no aplication to time series has been found in the literature. This approach has some advantages over traditional classification methods:

- Dataset labeling: the dataset does not have to be labeled manually based on an empirical time frame. The dataset is only labeled after the communities in the network are created and analysed, so a trend can have variable lenth and measurable strenght.

- Adaptive number of classes: the classes are defined in an unsupervised manner, according to patterns in the time series. So, the more diverse the time series, the more classes it will have. It is not tied to only three classes (UP, DOWN and FLAT).
- Length of the class and noise absorption: each class, represented in the network by a community, will have a certain length (in days). So, there is no need to define if the trend will be measured in days, weeks, months and so on. This also means that a daily DOWN can be interpreted by the model as a part of an UP trend, absorbing noises that do not affect the trend.
- Classes with measurable strengh: the classes are evaluated in terms of their average returns. This means that a trend can be ranked in terms of its historical behavior.

To study the topological properties of a time series, however, the time series must be converted to networks, consisting of a set of nodes and a set of links. In [16] some methods are described and divided in three big groups: i) proximity networks, ii) visibility graphs and iii) transition networks. The latter one, transition networks, builds a network where the nodes represent the transition of discrete states. The discretization of the linear series is commonly done by grouping similar observations (observations in the same pre-defined range) and converting them to symbols from the alphabet. For example, let *s* be the series *s = [0.3, 0.4, 0.6, 0.9, 0.1, 0.5, 0.2, 0.3]* with a discretization range of *0.2*, the converted series would be represented by the series *s' = [b, c, d, f, a, c, b, b]*. Then a linking rule can be applied to connect the nodes. The method proposed by this paper is very similar to the approach made by [11]. However, since in this case we are dealing with a stochastic time series, the data pre-processing step includes several transformations of the original time series, in order to obtain pseudo-cyclical processes. Most of the work made in this part stem from the concepts of the ARIMA process and the result produces several time series that are cyclical, detrended and related to the original series.

In addition to that, all the characterisitic series, produced by the original data are used together to define each node of the network. Thus, a node in the studied network is a multidimensional element in the topological space. All these aspects are covered in the proposed model section.

IV. Datasets

Two datasets are used in this paper. The first dataset, as already mentioned, is the Bovespa index daily close time series for the last 20 years. The second dataset is an artificial sinusoid curve in the format of (1).

$$y = \beta \pm \varphi, \quad t \in T \tag{1}$$

Where $$\beta = 100 + 50 * \sin\left(\frac{t}{50}\right) \tag{2}$$

And $$\varphi = \beta * \alpha, \quad \alpha \in [0,1] \tag{3}$$

The artificial dataset will be used to measure the classification accuracy and noise absorption of the model on dateset where the trend is known, since it is defined by a deterministic function with a variable noise.

V. Proposed Model

In this section, the proposed model will be described so that all the goals described in Sec.3 are addressed. First, the dataset is formalized and three parameters used in the model are defined. Then the model is divided into four steps: (a) characteristics extraction; (b) network generation; (c) community detection and trend classification and (d) trend forecasting.

The stock price series is smoothed with a five day moving average and assumes the form:

$$cp = [cp_1, cp_2, cp_3, \ \dots\ , cp_t] \tag{4}$$

Where cp_t is the moving average of order three of the Bovespa index in time T. The whole model depends on three parameters that are used in the pre-processing step. A definition for the short-term period ($srt \in \mathbb{N}$), a definition for the long-term period ($lng \in \mathbb{N}$) and a bin size for the discretization of the series ($N_b \in \mathbb{N}$).

In this paper, $srt = 25, lng = 250$ and $N_b = 0.28$.

Notice that setting a short term and long term period is used to extract short term and long term characteristics from the dataset, but it does not define the classes, i.e, does not set the duration of the trends.

A. Characteristics extraction

At this step, characteristics are extracted from the original series. The selection of the characteristics to be extracted is an heuristic process and although six characteristics were selected in this paper, others can be used. Six new series (C1,C2,…,C6), called characteristic time series, are created as follows:

- Characteristics C1 and C2 are the short-term and long-term noise (noise): for each point of the original series the noise is calculated as the percentage deviation of the price (cp_t) and the short-term (C1) or long-term (C2) moving average of the series cp. A $noise_t = 0$ means that the price on day t was equal to the moving average price of the period.

$$noise_t = \frac{cp_t - MA(q)_t}{MA(q)_t} \tag{5}$$

Where $MA(q)_t$ it the moving average of the price at a given t and q is the order of the moving average (number of days).

- Characteristics C3 and C4 are the short-term and long-term gradient (grad): for each point of the original series the short-term (C3) and long-term (C4) gradient are calculated as the percentage change in the moving average.

$$grad_t = \frac{MA(q)_t - MA(q)_{t-1}}{MA(q)_{t-1}} \quad (6)$$

- Characteristics C5 and C6 are the short-term and long term relative high-low position (rhl): for each point of the original series the relative short-term (C5) and long-term (C6) high-low position is calculated as the position of the price (cp_t) in relation to its moving minimum and maximum.

$$rhl_t = \frac{cp_t - \text{MIN}(cp_{t-q}:cp_t)}{\text{MAX}(cp_{t-q}:cp_t) - \text{MIN}(cp_{t-q}:cp_t)} \quad (7)$$

Where q is the order of the minimum and maximum period (number of days).

The six series are then normalized and discretized so that $-1 \leq C(n)_t \leq 1$ and $C(n)_t$ assumes the value of the closest bin defined by N_b. Finally, a time series composed of six dimensional vectors (V) is formed by joinning the characteristics at each time $t \in T$ (8).

$$V = [[C1_t, C2_t, C3_t, C4_t, C5_t, C6_t], [C1_{t+1}, C2_{t+1}, C3_{t+1}, C4_{t+1}, C5_{t+1}, C6_{t+1}], \ldots, [C1_T, C2_T, C3_T, C4_T, C5_T, C6_T] \quad (8)$$

By doing so, similar states of the process can be identified along time, i.e., days where the characteristic vector is exactly the same. For example, if $V_{t+a} = V_{t+b} = V_{t+c}$, these three days (represented by vectors) have similar short-term noise, long-term noise, short-term gradient, long-term gradient, relative short high-low position and relative long high-low position. Therefore, this process pre-groups these days into similar states of the process.

To ilustrate this part of the process, Fig. 1 shows the characetristics series C1, C3 and C5 (short-term characteristics) for a period of 50 days. Fig. 1.a shows the dataset (cp) for this period. Fig. 1.b shows the normalized short-term characteristics series (C1 in red, C3 in green and C5 in blue) for the same period, while Fig 1.c shows the discretized short-term characteristics series (C1 in red, C3 in green and C5 in blue) for this period.

Notice that two important things are achieved with this data pre-processing: (a) 20 years of data can be represented with characteristics series that oscilate from -1 to 1 and (b), as already mentioned, similar days are pre-grouped at this step. By looking at Fig 1.c, it is possible to observe that, in terms of short-term characteristics, days 5019, 5020 and 5021 are similar, which cannot be concluded from Fig 1.a and Fig 1.b.

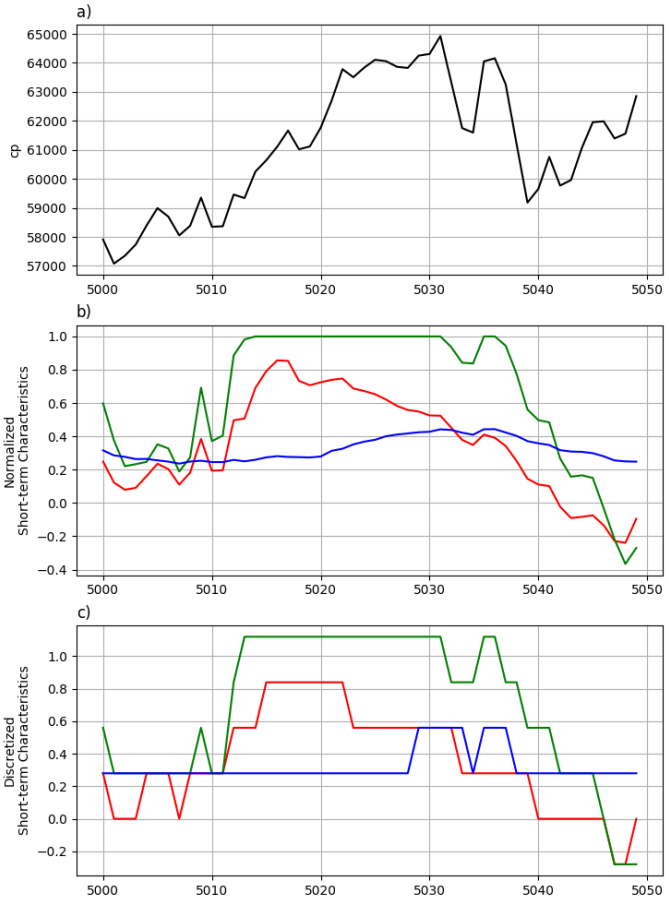

Fig 1 - Pre-processing of the series cp into the short-term characteristics series. (a) Series cp; (b) Normalized short-term characteristics and (c) Discretized short-term characteristics. C1 in red, C3 in green and C5 in blue.

B. *Network generation*

In order to generate the network, two rules have to be applied, one that defines the nodes and another that defines the linking procedure. The proposed model considers that each distinct vector in V is a node of the network and uses a simple temporal rule to link the nodes, i.e, V_t is connected to V_{t+1}.

Let W be the set of nodes in the network and W' an ordered list of nodes as they appear in V, the network is formed as indicated in Algorithm 1 and is stored in the format of and adjacency matrix M.

Algorithm 1 – Network Generation

1: **procedure** Network Generation
2: $new\_node \leftarrow 1$

```
3:    for t in T do:
4:        if V_t is a vector not read yet then:
5:            add new_node to W
6:            W'_t = new_node
7:            new_node ← new_node + 1
8:        else:
9:            search W for the node related to V_t
10:           W'_t = node
11:   create empty matrix M_xy  (x = y = number of nodes)
12:   for t in T − 1 do:
13:       x = W'_t,  y = W'_{t+1}
13:       M_xy = M_xy + 1
14:   return M
15: end procedure
```

C. Community detection and trend classification

The previous step ended up with a network, represented in the form of the adjacency matrix M. The proposed method assumes that similar observations of the time series (in regards to the characteristics chosen) can be grouped in communities and since the links are generated considering the temporal order of the nodes, it is expected that trends will be represented by the different communities.

Therefore, the community detection algorithm proposed in [11] is applied to the network to group the nodes considering their position in time. This method is based on a modularity measure (Q) that measures how the given network is structured in comparison to a random network. A modularity of over 0.30 indicates that the network is not random and the higher the modularity, the more the communities can be used to classify its elements.

However, the community detection algorithm simply groups the nodes in an unsupervised manner and thus, the communities generated have no meaning whatsoever.

To classify each community detected, the ordered node list (W') is read from $t_0 \rightarrow T$ and every time a different community is read, the percentage difference (%Δ) in the original series (cp) is recorded to the community. Notice that both series are referenced in the same time space. In the end the average %Δcp is calculated for each community and a label 'UP' or DW' can be atributed acording to the average %Δcp. This process is described in Algorithm 2.

Algorithm 2 – Community Classification

```
1: procedure Community Classification
2:     create empty %Δcp list for each detected community
2:     for t in T do:
3:         if W'_t ≠ W'_{t−1} then:
4:             record cp_t as the initial cp of the current community
5:         else if W'_t ≠ W'_{t+1} then:
6:             record cp_t as the final cp of the current community
7:         calculate %Δcp and update community %Δcp list
8:     for each community do:
9:         calculate the average %Δcp
10:            if %Δcp > 0 then:
11:                label community as 'UP'
12:            else:
13:                label community as 'DW'
14: end procedure
```

Another approach to this part of the proposed method is to use particle competition for community detection [14]. This community detection method has the advantage of identifying nodes that participate in more than one community, however the number of communities, or at least a sense of this number, has to be known a priori. Therefore, this method can be performed to identify overlapping nodes, after the number of communities is known.

D. Trend forecasting

Forecasting the trend was approached in this paper as the similarity between the characteristics of the days to be forecasted and the communities of a network constructed with the time series. In order for this concept to work, the communities have to be reduced to one representative node. This way, the distance between a new node (or a test node) and the representative node can be measured.

Reducing the communities to a single representative node can be achieved by using a centrality measure combined with the spatial position of the nodes, not of the whole network, but of each one of the communities.

Several methods are available for this task, such as the degree, betweenness, eigenvector, katz, pagerank, among others. For tha sake of simplicity, the proposed method uses the degree centrality as a weight measure to calculate the representative node of each community.

In order to test the forecasting capability of the model, the dataset was divided into a training set and a test set. Since the temporal order of the observations is vital to the network formation, the training set must contain one block of sequential data prior to the test set. Once this is observed, several strategies can be applied to run the test.

In this paper the forecasting was done as following:

1. The last 1000 observations were removed from the orginal dataset and saved as a test set;
2. Steps A to C of the proposed model were applied to the training set composed of the remaining original dataset. A network, namely the historical network, was generated;
3. The representative node of each community, or community center (C_c), is defined using the degree measure and the spatial position of each node of the community. This step reduces every community to one characteristic node, which is the gravity center of each community (9).

$$C_c = \frac{\sum_1^n k_n * V_n}{n} \quad (9)$$

Where n is the number of nodes in the community, k is the degree of the node and V is the characteristic vector.

4. The test set is divided in 10 groups os 100 observations.

5. To predict the first 100 days of the test set, the days are added to the trainning set, and only step A of the proposed model is run, in order to generate the characteristic vector of theses 100 days.
6. The distance between each 100 vector and the community centers is measured using euclidean distance. Each day is forecasted according to the closest community center.
7. The 100 observations are included in the trainning set and a new historical network is generated.
8. Steps 2-6 are repeated until the test set is empty.

As described above, the forecast was done for the last 1000 observations, that were divided into 10 tests containing 100 days to be forecasted. One could also update the network for every day being tested, however the computational cost of such a strategy is not worth considering that one day has little impact in the previous network. Updating the network with bigger blocks of observations yelds similar results at a much lower computational time.

VI. Experimental results

A. *Trend classification*

As the original series (cp) is not determined by a known function, the algorithm was applied to an artificial deterministic curve with an adjustable noise, so that the classification error and modularity of the network could be measured.

The sinusoid function described in Sec.4 was used and the noise φ was tested with α ranging from 0 to 1. This was performed to test how sensible the model is to noise, both in terms of classification error as well as the modularity measure of the network (since a modularity lower than 0.3 indicates that the structure is random). Fig. 2 shows, respectively, the classified artificial curve with no noise ($\alpha = 0.0$), with a noise of up to half the β value ($\alpha \in [0.0,0.5]$) and with a noise of up to β value ($\alpha \in [0.0,1.0]$). The noise is generated randomly inside the specified range.

As the noise is increased from 0.0 to 1.0, the classification error increases linearly until it reaches 0.4 (Fig. 3), where the model has very little advantage over a blind guess, considering only two classes. The modularity behaves as shown in Fig. 4. In this case the modularity stayed above the 0.3 threshold, showing that the network presented structured communities throughout the whole noise range used. These two figures show a good noise tolerance of the algorithm and makes it very useful in processes with noise ranging below 0.2 of the function value, where the algorithm has an accuracy greatter than 0.9. This is the case of the original series (cp). If we consider the short term moving average as the function value and the drift from the average as the noise, most of the short term noise of cp falls in the range $-0.1 \leq \alpha \leq 0.1$. At these levels of noise the algorithm should have an accuracy ranging from 0.90 to 0.95. This assumes, however, that the process is determined by a deterministic function (wheter it is known or not).

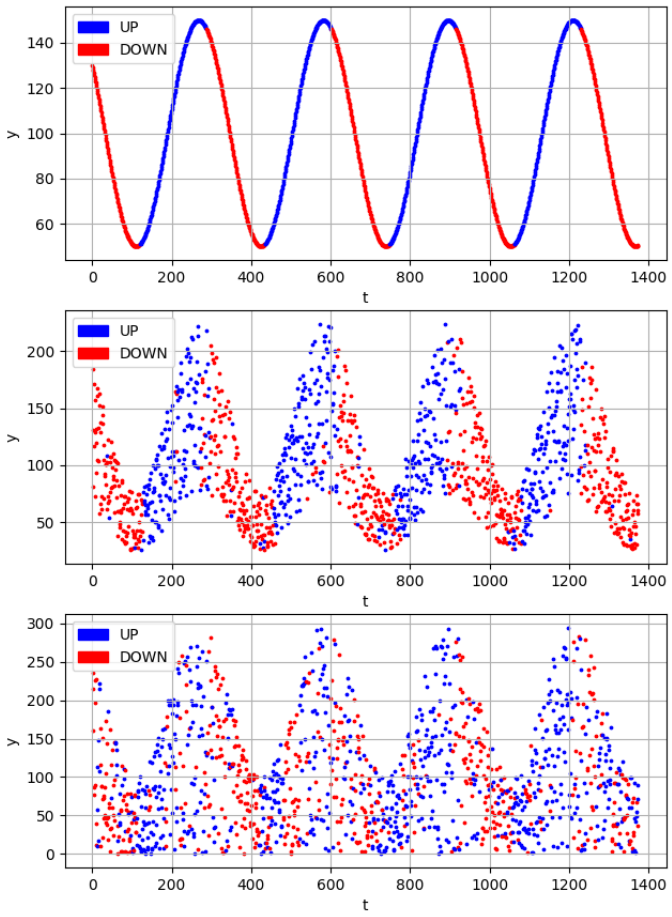

Fig 2 – Classified artificial curve. (a) $\alpha = 0.0$, (b) $\alpha = 0.5$ and (c) $\alpha = 1.0$

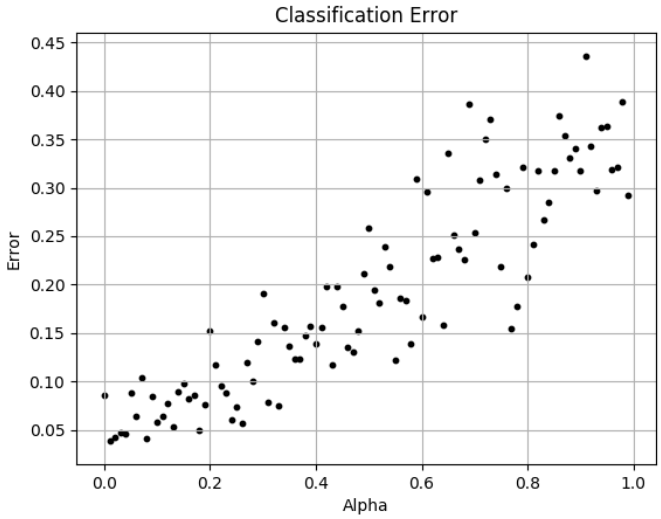

Fig 3 - Classification error as alpha is increased

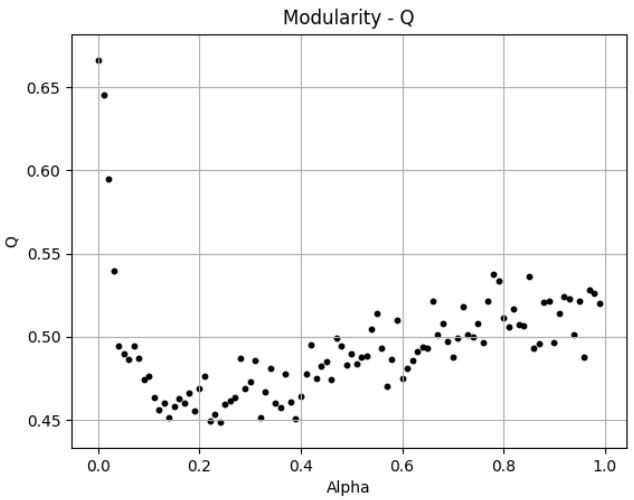

Fig 4 - Modularity variation as alpha is increased

The series cp, on the other hand, is more likely a stochstic process. So, in this case, the proposed method can not be verified in terms of its accuracy and has to be used as a classification rule, given its good results in the artificial curve. In the forecasting section this rule will be used to validate the predictions.

The algorithm was then applied to cp. Table I shows some metrics of the network generated by the method when it uses the modularity measure for the community detection step (only three communities are displayed for the sake of space). The following indicators are displayed:

- Community: community label.
- Node count: number of nodes gouped in the community.
- Day count: number of days from the series cp grouped in the community.
- % Part: participation of the community in the total number of days.
- Occurrences: number of occurrences of the community in the series cp. The more occurrences, the more common is the community pattern.
- % Δcp: average return measured as indicated in the proposed model and used to define the trend.
- Trend: community trend. All average returns grater than zero are grouped as 'UP' communities.
- % Com_ch: percentege of trend change over the communities occurrences. A %com_ch of zero means that the community has no link to a community of a different trend.

Finally, Fig. 5 shows the last 1000 points of the dataset. Each observation is represented in a different colour, depending on the community it participates. This segment of the dataset contains 23 of the 34 communities, however it is possible to see the ability of each community to capture a diferent pattern in the series. Then, Fig. 6 shows the same period colored by its trend. Here it is possible to see how several communities can form a longer 'UP' or 'DOWN' trend.

TABLE I. ACCURACY BY MODEL

Total Observations (Series length): 5405
Network Modularity (Q): 0.543
Total node count: 910
Total community count: 34

Community ID	**A**	**B**	**…**	**AI**
Node Count	21	22	…	33
Day Count	193	460	…	147
% Part	3.6	8.2	…	2.7
Occurrences	5	18	…	7
%Δcp	16.94	7.93	…	-13.58
Trend	UP	UP	…	DW
%Com_Change	60.0	38.9	...	28.6

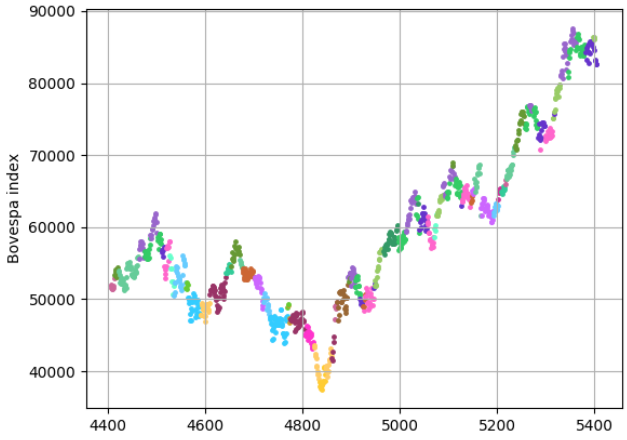

Fig 5 - Last 1000 points of the cp series, colored by the community they participate

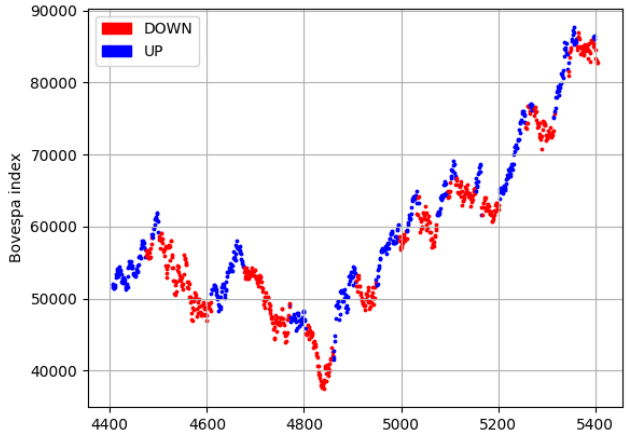

Fig 6 - Last 1000 points of the cp series, colored by its trend

B. *Trend forecasting*

To start the forecasting test, the full *cp* series had to be labeled according to the proposed method, so steps A to C of the proposed method were applied to the full dataset. This approach assumes that the classification process successfully labels the full dataset, considering that the communities are classified according to their average $\%\Delta cp$.

Then, the forecasting method was tested as described in D of the proposed method. The method was applied to the test set (composed of 1000 data points and divided into 10 blocks of 100 days). Results are shown in Table II.

Finally, the proposed method was compared to three popular classification methods: Naive Bayes (NB), Decision Tree (DET) and Multilayer Perceptron (MLP). This time the data set was not divided into 10 blocks of 100 days. Instead, the network data was passed to the classifiers and a 10 fold strategy was used. Therefore, each classifier received a list of nodes, the six characteristics for the node and the label for the node. This strategy was used so that there was no repetition of element in the trainning set and the test set. The comparison is also demonstrated in Table II.

TABLE II. RESULTS COMPARISON AMONG REFERENCE METHODS

Method	Precision	Recall	F-measure	Accuracy
Proposed	0.83 ± 0.27	0.92 ± 0.11	0.89 ± 0.13	0.93 ± 0.07
DET	0.78 ± 0.14	0.80 ± 0.11	0.78 ± 0.08	0.79 ± 0.07
NB	0.76 ± 0.16	0.77 ± 0.13	0.75 ± 0.10	0.76 ± 0.09
MLP	0.72 ± 0.16	0.74 ± 0.15	0.71 ± 0.11	0.73 ± 0.10

VII. CONCLUSIONS

The work presented here proposed a new approach to time series trend classification and forecasting. Although the framework was set to deal with stock time series (more specifically the Bovespa index), many other applications can be imagined using the same algorithm. We used six characteristics series derived from the original series to create a network composed of multidimensional nodes. One could, for example, use any number of related series to study a process that is correlated to all these series. In the case of the Bovespa index, one could, for instance, collect the historical data for the Brazilian interest rate, the dollar/real rate, Brazilian GDP, Brazilian inflation, US interest rate, US GDP, etc., and use the same algorithm to predict how the trend in one of the series is affected by the others combined (different from a linear correlation, which is done in pairs). It is important to notice that although the proposed model does not keep track or predict the values of the time series, it gives a trend classification that can be used as a trigger for a buy/sell action. Attaching other methods, like risk management algorithms, could yeld a successful trading strategy.

Furthermore, the method can possibly be extended to other domains, with a few adaptations, such as the characteristics to be extracted and the time frame for the moving averages. One could, for example, use the presented framework to study the production yeld of a certain crop using enviromental data, like temperature, precipitation and wind as the characteristic series.

As far as the experimental results are concerned, the proposed method was able to present a new way to classify the trend of a stochastic series in an unsupervised manner. The model was able to cluster similar patterns in different communities of the network, what solved most of the problems detailed in Sec. 3. Looking at the forecasting results, we can conclude that the proposed method could beat the selected methods in terms of accuracy,indicating that the community center, as described in the proposed method, is a good reference to predict the trend of the current observation.

These results indicate that looking at the network structure can yeld rich data both to create new prediction frameworks, as well as to use in other applications.

REFERENCES

[1] Rani, S., 2014, September. Review on time series databases and recent research trends in Time Series Mining. In Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference- (pp. 109-115). IEEE.

[2] Leite, Helio de Paula and Sanvincente, Antonio Zoratto, 1995. Índice Bovespa: Um padrão para os invetimentos brasileiros. Ed. Atlas.

[3] Fama, Eugene, 1970. Efficient Capital Markets: A Review of Theory and Empirical Work. The Journal of Finance.

[4] Li, Wei and Liao, Jian, 2017. A comparative study on trend forecasting approach for stock price time series.

[5] Tseng, F.M., Tzeng, G.H., Yu, H.C. and Yuan, B.J., 2001. Fuzzy ARIMA model for forecasting the foreign exchange market. Fuzzy sets and systems, 118(1), pp.9-19.

[6] Chen, P., Pedersen, T., Bak-Jensen, B. and Chen, Z., 2010. ARIMA-based time series model of stochastic wind power generation. IEEE Transactions on Power Systems, 25(2), pp.667-676.

[7] Hillmer, S.C. and Tiao, G.C., 1982. An ARIMA-model-based approach to seasonal adjustment. Journal of the American Statistical Association, 77(377), pp.63-70.

[8] Pai, P.F. and Lin, C.S., 2005. A hybrid ARIMA and support vector machines model in stock price forecasting. Omega, 33(6), pp.497-505.

[9] Zhang, G.P., 2003. Time series forecasting using a hybrid ARIMA and neural network model. Neurocomputing, 50, pp.159-175.

[10] Barabási, A.L., 2016. Network science. Cambridge university press.

[11] Ferreira, L.N. and Zhao, L., 2014, October. Detecting Time Series Periodicity Using Complex Networks. In Intelligent Systems (BRACIS), 2014 Brazilian Conference on (pp. 402-407). IEEE.

[12] Newman, M.E. and Girvan, M., 2004. Finding and evaluating community structure in networks. Physical review E, 69(2), p.026113.

[13] Clauset, A., Newman, M.E. and Moore, C., 2004. Finding community structure in very large networks. Physical review E, 70(6), p.066111.

[14] Quiles, M.G., Zhao, L., Alonso, R.L. and Romero, R.A., 2008. Particle competition for complex network community detection. Chaos: An Interdisciplinary Journal of Nonlinear Science, 18(3), p.033107.

[15] Palla, G., Derényi, I., Farkas, I. and Vicsek, T., 2005. Uncovering the overlapping community structure of complex networks in nature and society. Nature, 435(7043), pp.814-818.

[16] Donner, R.V., Small, M., Donges, J.F., Marwan, N., Zou, Y., Xiang, R. and Kurths, J., 2011. Recurrence-based time series analysis by means of complex network methods. International Journal of Bifurcation and Chaos, 21(04), pp.1019-1046.