Diogo Manuel Pires de Matos

Licenciatura em Engenharia Informática

State Detection in a Financial Portfolio

A Self-organizing maps approach for financial time series

Dissertação para obtenção do Grau de Mestre em Engenharia Informática

Orientador: Prof. Doutor Nuno Marques, Professor Auxiliar, FCT

Presidente: Prof. Doutor Carlos Augusto Isaac Piló Viegas Damásio

Arguente: Prof. Doutora Maria Margarida Guerreiro Martins dos Santos Cardoso

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Setembro, 2014

States Detection in a Financial Portfolio

Copyright © Diogo Manuel Pires de Matos, Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa.

A Faculdade de Ciências e Tecnologia e a Universidade Nova de Lisboa têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

Agradecimentos

Agradeço ao meu orientador e colegas de curso pelo acompanhamento e disponibilidade mostrada no apoio mais direto. Também queria agradecer à minha namorada, família e amigos pelo apoio extra. Finalizo com um agradecimento à FCT, por ter proporcionado esta etapa de formação académica e pessoal.

Abstract

This study analyses financial data using the result characterization of a self-organized neural network model. The goal was prototyping a tool that may help an economist or a market analyst to analyse stock market series. To reach this goal, the tool shows economic dependencies and statistics measures over stock market series.

The neural network SOM (self-organizing maps) model was used to extract behavioural patterns of the data analysed. Based on this model, it was developed an application to analyse financial data. This application uses a portfolio of correlated markets or inverse-correlated markets as input. After the analysis with SOM, the result is represented by micro clusters that are organized by its behaviour tendency.

During the study appeared the need of a better analysis for SOM algorithm results. This problem was solved with a cluster solution technique, which groups the micro clusters from SOM U-Matrix analyses.

The study showed that the correlation and inverse-correlation markets projects multiple clusters of data. These clusters represent multiple trend states that may be useful for technical professionals.

Keywords: Financial Markets, SOM, Correlated Markets, Clustering over U-Matrix

Resumo

Este estudo analisa a caracterização de dados financeiros que é feita através de um modelo de rede neural auto-organizada. O objetivo foi a criação de um protótipo que tem como funcionalidade ajudar um economista ou analista de mercado. Para alcançar este objetivo, a ferramenta mostra as dependências e estatísticas económicas que foram calculadas apartir de séries financeiras.

O modelo de rede neural SOM (mapas auto-organizáveis) foi usado para extrair padrões de comportamento dos dados analisados. Tendo por base esse modelo, foi desenvolvida uma aplicação que permite analisar dados financeiros. Esta aplicação utiliza um portfolio de mercados correlacionados ou mercados inversamente correlacionados como entrada. Após a análise com SOM, o resultado é representado por micro grupos que estão organizados pela sua tendência comportamental.

Durante o estudo surgiu da necessidade de melhorar a análise resultante do algoritmo SOM. Este problema foi resolvido com uma técnica de agrupamento, que agrupa os micro estados representados na U-Matrix do SOM.

O estudo mostrou que os mercados correlacionados e inversamente correlacionados projetam vários grupos de dados. Estes grupos organizam-se por tendencia podendo ser uma analise importante para os profissionais técnicos.

Palavras-chave: Mercados Financeiros, SOM, Mercados Correlacionados, Agrupamento sobre UMatrix

Content

Figure List

Tables List

1

Introduction

Financial markets may be subjected to extreme and complex events, which can easily go unnoticed by human perception. Many research groups search new algorithms and new technology that allows the extraction of knowledge from several conjoint data streams. In this thesis we study several methods that may be useful for developing new ways to analyse and visualize financial markets.

This study is organized in three major phases that can be titled as data characterization, state detection and validation. The data characterizations focus on choosing a set of time series and also prepare the data for the state detection. The state detection is based on the use of SOM (self-organizing map) [1] to find similarity patterns in the stream data. Finally, the validation phase consists of proposed tools to analyse the studied model.

SOM gives the possibility to use the properties of a neuronal network that self-organizes the characterized data. This technique presents data in a two dimensional map organized by its characteristics. We study how this method can be used to know how the financial markets are correlated, which is an added value to technical analysis studies. Possible applications of this method are the study of global inter-dependencies presented. Also, this study lead to develop a new method for discriminating clusters in SOM.

1.1 Motivation

The financial markets data may be one of the most complex fields of analysis. The speculation and the competition in stock markets may lead to high volatility and investment risk that could be mitigated if an early detection of relevant patterns is available. For example, this can be used to guide regulatory policies.

The problem addressed is the difficulty in mapping the state of the financial system. Based on this principle, mapping the huge number of states of a financial system seems a difficulty that stands out, as well the detection of interactions between the markets. Therefore the sub-problem of mapping interactions of correlated markets states is relevant. This study will take in consideration how the correlated markets interact when the global systems changes and if this sub-problem addresses any type of micro states that can be a marker of upcoming crises states or the return to normality after a crisis state.

The algorithm of self-organizing maps appears to be a good method for this problem. It is a self-adapting neuronal network that has the particularity of identifying clusters in data and also project them in a two dimensional map. This algorithm is already being used to study financial data but many approaches are still unexplored.

1.2 Objectives

The main objective for this thesis is to understand how correlation between markets may help to identify particular trend change states. As mentioned before the chosen algorithm was SOM (self-organizing maps). The objective is to cluster the data by a state of behaviour after training the net and finally define a path of states (clusters) which have been visited.

1.2.1 Data Selection

The first sub-goal of this thesis is the selection of meaningful data that will be used in the case studies. This data was selected based on economic history and possible correlations between it. Certain patterns will be tested to get the best

results and decide if correlation of stocks brings a useful analysis to stock market traders.

1.2.2 Characterizations and first usage of SOM

The second sub-goal for this thesis is characterize the data selected in the previous goal. This process uses SOM algorithm.

After the data was treated, it was classified with the SOM algorithm. This classification will be filtered by other clustering processes that uses the SOM distances matrix (UMAT). This process will allow the formation of big clusters that gather closer neurons in the map.

1.2.3 Behaviour and Time Series Generator

Based in this sub–goal several visual tools will be tested as possible new ways for analysing the states of the time series.

A particularly interesting visualization tool will be proposed by generating a time series based on the obtained clusters. The new time series will be analysed and compared with reality time series.

With this simulation it will be possible to see the stability of different time periods. Each state will present different mean and variance according to the considered cluster. It is expected that during a crises period the variance will be bigger than during a stable period.

1.3 Thesis Structure

Chapter 2 consists on the results of the literature studied where the support for the thesis is presented. There is a brief overview about financial studies, clustering algorithms and some methods to analyse the clustering.

In Chapter 3 we can find the proposed approach for this system. This part is divided in the Architecture of the system, the data source, the characterization method, the state identifier/clustering, the studies of the general states and the generator.

Chapter 4 analyses the clustering algorithm with well-known cluster datasets.

Chapter 5 tests the algorithm with the financial data. This chapter is divided in 3 sections. The first two corresponds to testing the case studies and the last on corresponds to presented some future work that may be done

Finally, the last chapter present some conclusions about the entire study including the methodology used and the clustering algorithm.

2

Related Work

2.1 Financial Markets Studies

Financial market studies can be divided in two areas namely technical and fundamental analyses. These studies have the objective of understanding the stock market movements, and ideally characterizing the economy. Here we aim to study possible tools to identify decision patterns that help in the generation of profitable processes in stock market exchange. The possibility of profitability attracts many people that are committed to improve the returns of their investments and this can be related with the efficient market approach of [2].

Fundamental analysis can be defined as a method of evaluating an intrinsic value of a security by examining related economic, financial and other qualitative and quantitative factors. This science also attempts to study everything that can affect the securities value, including macroeconomic factors (like the overall economy and industry conditions) and company-specific factors (like financial condition and management) [3].

Technical analysis is defined [3], as the academic study of historical chart patterns and trends of publicly traded stocks. Much of the fundamental analysis and market opinions may be reflected in the stock price that by getting the recurrent price patterns and fitting them to past movement, may be possible to predict the future of the price.

2.1.1 Portfolio selection

Portfolio investment is one form to do investment with low risk value when compared with single stock investment. One well known study [4] presented a statistic process to select some stocks evaluations from securities that are low correlated. Based in the rule which implies both that the investor should diversify and that he should maximize expected return. The rule states that the investor does (or should) diversify his funds among all those securities which give maximum expected return.

A study [5] explains the analyses to predict the future and understand the past. By risk measures it may be constructed a portfolio to maximize the returns of a found.

2.1.2 Correlation in markets

The effect of correlated markets in the global economy has been object of study. As showed in [6] reveals that crises states contagion between different countries is actually a fact. It has been used a correlation coefficient between r_i and r_j that can be written as:

$$\rho = \frac{\mathrm{Cov}(r_i, r_j)}{\sqrt{\mathrm{Var}(r_i) \times \mathrm{Var}(r_j)}} \quad \textbf{(1)}$$

This calculus is actually called Pearson's correlation coefficient. It oscillates between -1 and +1. Positive means that the two series are correlated and negative that they are not correlated. The value 1 means that is considered a perfect positive correlation but that value is very rare.

Other approach that appears in the study was the tracking of local correlations [7]. This Coefficient tries to respond to two Pearson coefficient problems that are: (i) the capturing of more complex relationships and (ii) it is too sensitive to transient changes, often leading to widely fluctuating scores.

As was showed in the study of local correlation Person coefficient presented the addressed problems, and the proposed approach seem to be a lot stable in an exchange rates dataset. This result may be seen in the figure 1.

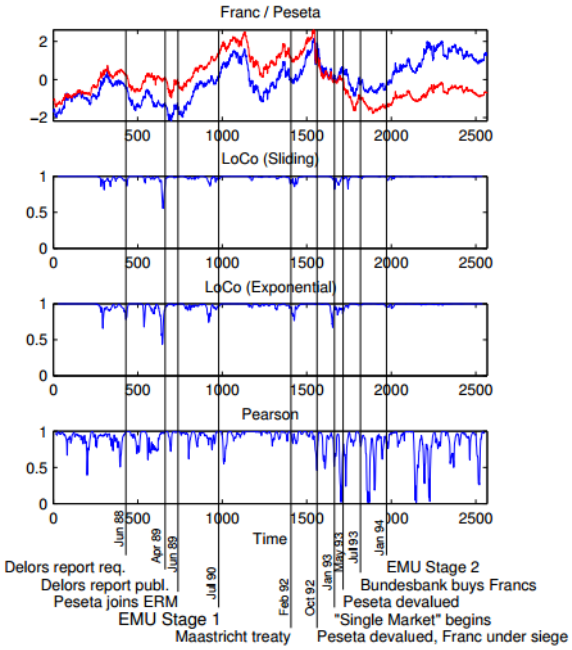

Figure 1 - Local correlation scores [7]

This study was made to select the datasets used by the SOM algorithm. The data selected for a set of stocks should be the most correlated. By doing this we can nominate the data as a set of correlated markets, which can be used for state detection process using a cluster technique.

In the next section it will be analysed how some technical indicators that were studied with the objective of characterizing a stock action time series. The ones that have been analysed were the moving average and the Hurst index.

2.1.3 Moving Average

The moving average is a technical indicator [8] that permits to calculate a trend of a time series. To calculate this index is really simple, it just has to be determined a period and then calculate the mean value for that period. The random variations of the time series are mitigated by this technique, making a lot easier to see if the markets are declining or rising.

The formula used to calculate the moving average uses t that represents the given time and the k represents the period of the moving average. With this we have:

$$\text{mavg(t)} = \frac{\sum_{t=t-k}^{t} x_t}{k} \quad (2)$$

In [8] is showed the use of moving average in constructing a profitable process. It is also explained some proprieties of using different time length with different moving averages. In the figure below can be seen two moving average calculations. One is up to 25 days period and the other is up to 40 days period. Analysing the figure 2 we can check that the index shows a more stable line compared to the real time series. This makes a lot easier to analyse the trend of the time series.

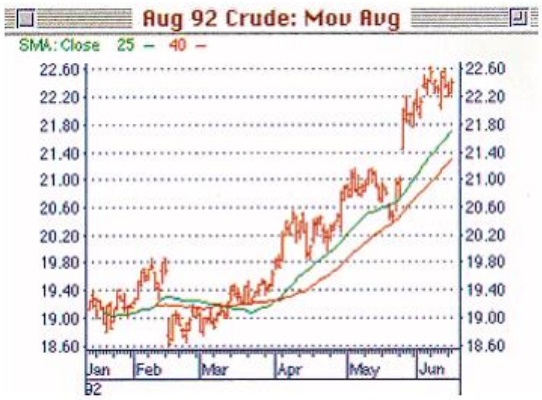

Figure 2 - Moving Average [8]

This technical indicator may be combined with machine learning technique to make a profitable process. In the study **[9]** is used a neuronal network to generate data and illustrate the potentiality as a trader method based on the Eurostoxx50 financial time series.

The index has a negative point regarding the choice of the analysis period. Meaning that when the chosen period is short, the line chart comes very close the actual series chart, confusing variations of real time series with a trend change. Choosing a long period makes the detection trend change more precise but also delayed when compared with a short period. The chosen period may be performed with some knowledge that professional's specialists have for each

problem in hand. Choosing the period with no specialist knowledge, may lead to a bad performance analyses.

2.1.4 Hurst

Hurst exponent [10] can be used as a long memory process for capital markets, making it one of the topics in finance.

The Hurst exponent referenced in [11], was used for fractal analysis but has been explored to use in many other research fields, including economics. This calculus has provided a measure for long term memory and the fractality of a time series. It is also known that Hurst exponent varies between 0 and 1, which can be considered in three subsets:

- $H = 0.5$ indicates a random series
- $0 < \mathrm{H} < 0.5$ indicates an anti-persistent series
- $0.5 < H < 1$ indicates a persistent series

Working with this index could be a good answer to measure time series predictability. It also can be used to measure the comparison of a generated time series with the real series that makes this index as an added value for the study.

Hurst exponent calculation

To calculate the Hurst exponent, the first action has to be the range estimation on the time span of n observations. A time series of full length N is divided into a number of shorter time series of length $n = N, N/2, N/4, \ldots$ The average re-scaled range is then calculated for each value of n.

For a (partial) time series of length n, $X = X_1, X_2, \ldots, X_n$, the rescaled range is calculated as follows:

Calculate mean value m:

$$\mathrm{m} = \frac{1}{\mathrm{n}} \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{X_i} \tag{3}$$

Calculate mean adjusted series Y:

$$\mathrm{Yt} = \mathrm{Xt} - \mathrm{m}, \quad \mathrm{t} = 1, 2, \ldots, \mathrm{n} \tag{4}$$

Calculate cumulative deviate series Z:

$$Z_t = \sum_{i=1}^{t} Y_i , \qquad t = 1, 2, \dots, n \tag{5}$$

Calculate range series R:

$$Rt = \max(Z1, Z2, \dots, Zt) - \min(Z1, Z2, \dots, Zt) \qquad t = 1, 2, \dots, n \tag{6}$$

Calculate standard deviation series S:

$$S_t = \sqrt{\frac{1}{t}\sum_{i=1}^{t}(X_i - u)^2} , \qquad t = 1, 2, \dots, n \tag{7}$$

Calculate rescaled range series (R/S):

$$\left(\frac{R}{S}\right)t = \frac{Rt}{St} \qquad t = 1, 2, \dots, n \tag{8}$$

Hurst found that (R/S) scales by power-law as time increases, which indicates:

$$\left(\frac{R}{S}\right)t = c \times t^H \tag{9}$$

The calculation result of the Hurst index may be combined with the real time series to give a better perception of the index itself. In the following chart you can see the Hurst index and the actual time series. The Hurst index shows up at the top of the chart and the real time series appears below it. On the real time series is also represented the Moving Average.

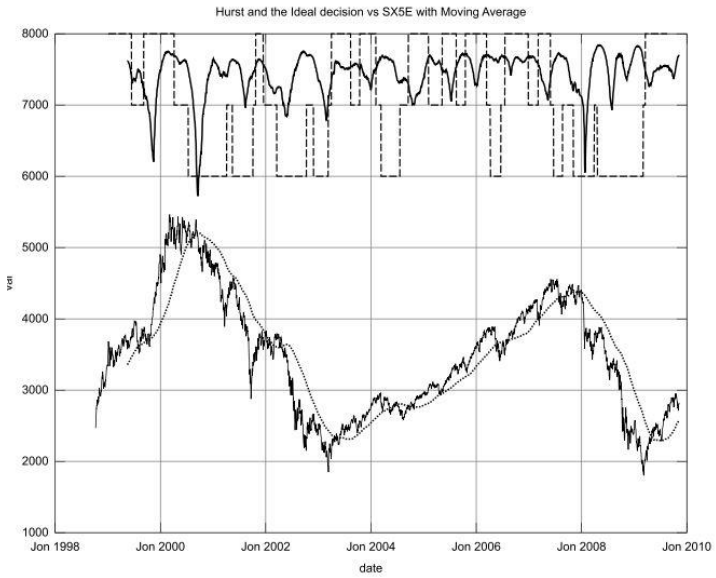

Figure 3 - Hurst Index [12]

With the graphic analysis, we can see that the Hurst is a good indicator of trend change in the time series. Changes in trends occur when it reaches a value below 0.5, which by analysis of the chart you can see in the beginning of 2001 and in June 2008.

2.2 Clustering

In this research appears the need of studying some methods that may be used to correlate and classify certain patterns of crisis state. As result some IA algorithms have been studied but by general characteristics self-organizing maps was chosen.

2.2.1 K-means

The k-means algorithm is a clustering algorithm widely known and used as presented in [13]. The principal objective of the algorithm is to distribute the data by a given number of clusters. By this we have the set of K that represent the number of clusters. The K setting may be considered as problem when the number of data classes/clusters is unknown. This problem has already some solutions for the k number auto-detection (a review of k-mean for k detection may be found in [14]).

After defining the number k, it also has to be defined the first k centroids that will be used in the learning process. The learning procedure is used to update the cords of each k centroids for a better cluster representation. This procedure uses a group of training patterns, which are sets of data to represent each cluster. The original algorithm [15] is depicted in figure 4:

```
Let         k   be the predefined number of centroids
            n   be the number of training patterns
            X   be the set of training patterns x1, x2,..xn
            P   be the set of k initial centroids μ1, μ2,… μk taken from X
            η   be the learning rate, initialized to a value in ]0,1[
1   Repeat
2           For i=1 to n
3               Find centroid μj∈P that is closer to xi
4               Update μj by adding to it Δμj = η(xi - μj)
5           Decrease η
6           Until η reaches 0
```

Figure 4 - K-means training process

After training, the data k-means uses an activation function for giving the respective centroid for each data object. If we have a data set $X = (x_1, x_2, \dots, x_n)$, k-means will classify which object x_i as a cluster c_i, where k is the number of clusters in the set of clusters $C = \{c_1, c_2, \dots, c_k\}$, that can be written as:

$$\underset{C}{arg\,min} \sum_{i=1}^{k} \sum_{x_i \in S_i} \|x_j - \mu_i\|^2 \quad (10)$$

, where μ_i is the centroid of cluster i.

2.2.2 KNN

The KNN algorithm [16] is a pattern recognition method used for classification and regression. This algorithm uses one simple way for its proposed objectives, where the objects are classified by majority vote of its neighbours. The majority class from the neighbours will be associated to the objects being classified.

The algorithm is executed based on labeled training samples, and also a configurable parameter *k* value. The value *k* may be defined by the user or automatically generated through other techniques, in order to optimize the process (see also [17] that compares some improves in KNN algorithm). During the classification stage, the algorithm searches for the *k* closest neighbours of the object in the training dataset and chooses the label of the majority class.

The usual distances metrics used to determine the distance of one testing entity and one trained entity differs from the type of the variables. Usually for continuous variables is used the Euclidian distance and for discrete variables may be used the Hamming distance. There are some studies [18] [19] to optimize the distance metrics with some statistical algorithms. This may improve the classification accuracy.

2.2.3 SOM

The Self-Organizing Map, developed by Teuvo Kohonen in the 80's [1], is a machine learning algorithm that helps to characterize high-dimensional data set. This method uses an unsupervised learning to produce a low-dimension data set, discretized representation of the input space of the training samples, called a map.

The map will reflect the preserved topological relations, i.e., patterns that are close in the input space will be mapped to units that are close in the output space, and vice-versa. So, to allow an easy visualization, the output space is

usually 1 or 2 dimensional. The basic SOM training algorithm can be described as follows, referenced from [20]:

Variables and formulas

• $\boldsymbol{s}$	current iteration
• $\boldsymbol{lim}$	iteration limit
• $\boldsymbol{t}$	index of the input data vector in the input data set **D**
• $\boldsymbol{D(t)}$	target input data vector
• $\boldsymbol{v}$	index id the node in the map
• $\boldsymbol{Wv}$	current weight vector of node v
• $\boldsymbol{u}$	index of the best matching unit (BMU) in the map
• $\boldsymbol{\Theta(u,v,s)}$	restraint due to distance from BMU, usually called the neighbourhood function
• $\boldsymbol{\alpha(s)}$	learning restraint due to iteration progress
• $\boldsymbol{d(p,q)} = \sqrt{\sum_{i=1}^{n}(q_i - p_i)}$	Euclidean distance
• $\boldsymbol{UpWv(s+1)} = \boldsymbol{Wv(s)} + \boldsymbol{\Theta(u,v,s)} \times \boldsymbol{\alpha(s)} \times (\boldsymbol{D(t)} - \boldsymbol{Wv(s)})$ Update formula for BMU (Best Matching Unit), new weight	

The SOM algorithm implementation:

This algorithm has the following steps, referenced in ref: [21]:

1) Randomize the map nodes weight vectors ($\boldsymbol{Wv}$)
2) Grab an input vector
3) Iterate each node in the map:

   - Use the Euclidean distance (in $\boldsymbol{d(p,q)}$) formula to find the similarity between the input vector and the map nodes weight vector

   - Track the node that produces the smallest distance (this node is the best matching unit, BMU)
4) Update the nodes in the neighbourhood of the BMU (including the BMU itself) by pulling them closer to the input vector. Use formula $\boldsymbol{UpWv(s+1)}$
5) Increase $\boldsymbol{s}$ and repeat from step 2, until reach iteration limit ($\boldsymbol{lim}$)

The steps considered before, allow the training of the self-organizing map based in the input samples. The grid made by the map will cover the data making possible the characterization over the map. This will make possible to differentiate the different classes of the processed data.

Figure 5 illustrates the training process evolution. It is possible to observe the selection of the best matching unit in the first step of the algorithm. The second step corresponds to update the best matching unit weight and also the surrounding neurons, as referenced in the formula $\boldsymbol{UpWv}$. Finally when the method achieves $\boldsymbol{lim}$, the map covers most of the input data.

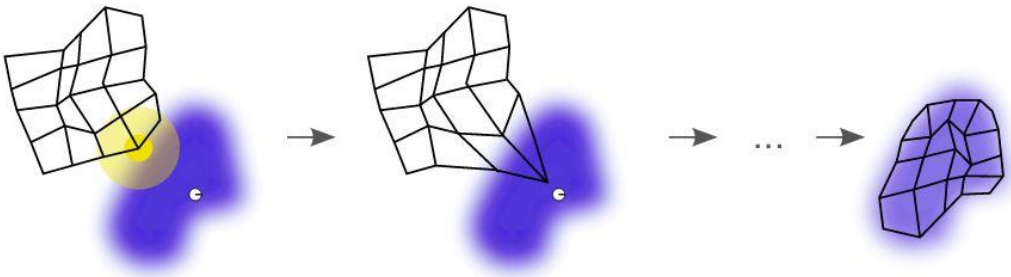

Figure 5 - Training Process of Self-Organizing Map [20]

After training it is possible to classify the testing data. When the new input is used, it will activate a neuron, the BMU (best matching unit), representing the map entity more correlated with the tested object. This neuron has a class associated (preferentially) that corresponds to the classification of the input data. An example of the SOM model is in figure 6. From one side we have the map neurons and from the other side we have the input node. Based in the input characteristics a map node (the BMU) will be activated.

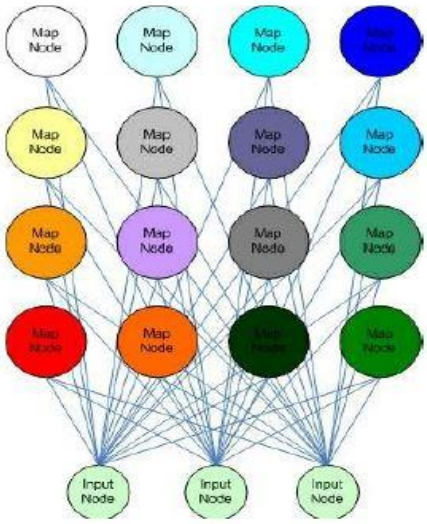

Figure 6 - Self-organizing map structure, referenced from [21]

There are some techniques for the interpretation of the map produced, a well-known is the UMatrix [22]. This technique provides us a simple way to visualize cluster boundaries in the map. This method is useful because it can find clusters in the input data without having any *a priori* information about the clusters. Figure 7 is an example of an UMatrix, in which darker colours represent a larger distance between neurons and lighter colours represent low distance between neurons.

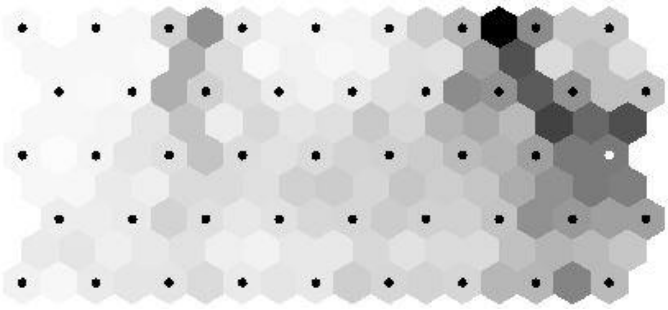

Figure 7 – UMatrix, referenced from [23]

The component planes consist in a view analyses which show a colour scale, which measures the importance of each map variable in some areas of the map. As showed in figure 8, is the UMatrix and the component planes from the iris dataset. On the left side of each variable is the colour scale, and by analysing the variable map is possible to see the zone where the variable is more important.

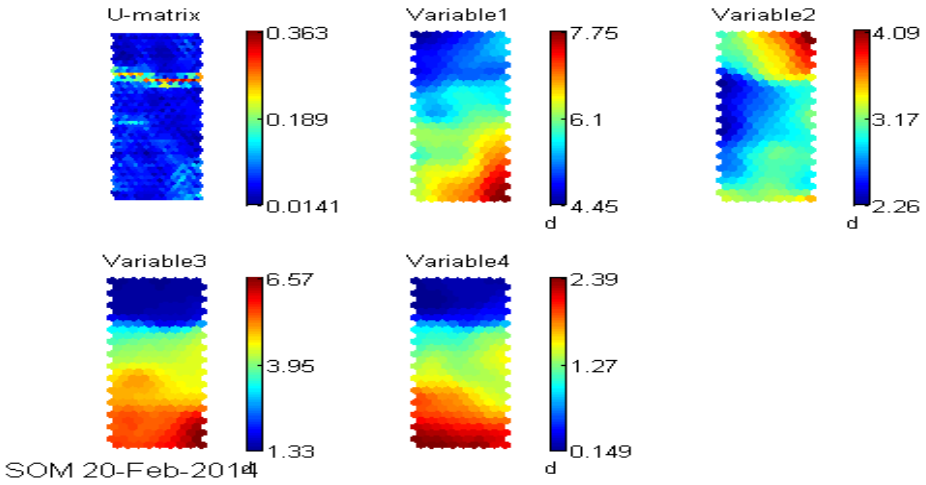

Figure 8 : Iris Dataset, generated in SOM toolbox

Figure 8 shows the importance of variable2 (corresponding to sepal width of iris dataset) on the identification of the upper cluster, which is classified as Iris-Setosa class. Also it can be seen the importance of the variable4 (Petal Width) in the identification of Versicolor class that can be easily mistaken with Iris-Virginica.

Some solutions were studied motivated by clustering on the UMAT. A two stage clustering approaches [24] seem to solve the clustering problem over the UMAT. The goal of a 2-stage clustering method is to overcome the major problems of the conventional methods as the sensibility to initial prototypes (proto-cluster) and the difficulty of determining the number of clusters expected. The aim of the SOM at the first stage is to identify the number of clusters and relative centroids, to overcome, the problems described above. In the second stage, a partitioned clustering method will assign each pattern to the definitive cluster.

The main advantage of using a SOM as a first stage clustering algorithm is to make a clustering of nodes (proto-cluster) of Kohonen layer, which are generally less than input pattern. Therefore, this method led to an advantage from the computational point of view. However, in this case, the choice of a number of expected clusters may be required (partitioned clustering).

There are three post-processing methods that are known as a solution. The first one is the classic K-Means algorithm [25], which belongs to class of partitioned clustering methods. The second is the U-Matrix with connected components [26], a hierarchical algorithm with bottom-up approach. The third one is a model created by DAME group, inspirited by an agglomerative method based on dynamics SOM, hereinafter defined as Two Winners Linkage [27].

Just like a solution found in [28] that is based in [29] uses the property of high density region that is separated from another high density regions by regions of low density. Also this approach determines the number of clusters automatically.

There are several studies that apply the SOM algorithm to finance. The first example represents an application of SOM algorithm for crises states detection. This study [30] uses the self-organizing maps for mapping the state of the financial stability and as well for predicting systematic financial crises. The SOM algorithm was used to monitor the macro-financial vulnerabilities by locating a country in a financial stability cycle, which is represented by the pre-crises, crises, post-crises and tranquil state.

A recent work [31] explores this technique using the application of the SOM algorithm to characterise one stock market series based in some technical indicators. This approach gives some good results and there were some technical indicators highlighted from the study. The best technical indicators were the volume, mean volume and the price variation. In the end, the SOM was able to identify some crises and stable states.

One other study [32] uses the SOM algorithm to analyse the evolution of the companies and the same time track for important trajectory patterns. In this study is used a two stage procedure. In the first stage the SOM algorithm is used to cluster the data and the second stage analyses trajectory patterns over the map.

2.3 Characterizing the Clustering

2.3.1 Flooding

The Flooding is a concept that has been approached by some computer science algorithms. It extends from computer science network algorithms to graphics algorithms. For the network area we have the routing algorithm [33], which delivered packets are re-send using flooding concept (the packet received is re-sent for all output link except to the source link).

Since the distance matrix of self-organizing map may be seen as a topographic landscape, the flooding algorithm flood fill [34] was considered relevant to this study. This technique is used in the "bucket" fill tool of paint programs to fill connected, similarly coloured areas with a different colour. This approach served as an inspiration for our approach that starts flooding from the local mins detected from the previous method of the clustering algorithm used in this study. A similar approach was used to identify clusters in the UMAT.

2.3.2 Methods for evaluating clustering quality

For knowing which was the best flood parameter some clustering quality measuring approaches were covered by the study. The SOM Topographic error and Quantization error [24] were the typically measures used to evaluate a map quality and may not be the best response to measure a second stage clustering result.

The study [35] compares some of the well-known cluster criterion measures for determining the number of clusters based in the clustering quality. So this paper compares the stability measures of a recently proposed study [35], which also analyses the Calinski-Harabasz index [36] and the Krzanowski and Lai index [37].

As the Calinski-Harabasz index will be assumed as the variance ratio criterion (VRC_k) defined as:

$$VRC_k = \frac{SS_B}{SS_W} \times \frac{(N-k)}{k-1} \tag{11}$$

, where SSB is the overall between-cluster variance, SSW is the overall within-cluster variance, k is the number of clusters, and N is the number of observations.

The overall between-cluster variance SSB is defined as:

$$SS_B = \sum_{i=1}^{k} n_i \|m_i - m\|^2 \quad (12)$$

, where k is the number of clusters, n_i is the number of observations by cluster, m_i is the centroid of cluster i, and m is the overall mean of the sample data.

The overall within-cluster variance SSW is defined as:

$$SS_w = \sum_{i=1}^{k} \sum_{x \in c_i} \|x - m_i\|^2 \quad (13)$$

, where k is the number of clusters, x is a data point, c_i is the ith cluster, and m_i is the centroid of cluster i.

Well-defined clusters have a large between-cluster variance (SS_B) and a small within-cluster variance (SS_W). A larger VRC_k ratio results in a better data partition. To determine the optimal number of clusters, maximize VRC_k with respect to k. The optimal number of clusters is the solution with the highest Calinski-Harabasz index value.

As the Krzanowski -Lai index will be assumed as the variance ratio criterion (VRC_k) defined as:

$$VRC_k = \left| \frac{DIFF(k)}{DIFF(k+1)} \right| \quad (14)$$

, k represents the number of clusters and the $DIFF_k$ is represented by:

$$\mathrm{DIFF(k)} = (\mathrm{k}-1)^{\frac{2}{\mathrm{p}}} \times \mathrm{SS_w}(\mathrm{k}-1) - \mathrm{k}^{\frac{2}{\mathrm{p}}} \times \mathrm{SS_w}(\mathrm{k}) \quad (15)$$

, where the SS_w function represents the within-cluster variance (16).

Another well-known and used cluster index is the Davies-Bouldin [38]. This index measures the ratio of intra-cluster and extra-cluster distances, measured

from centroids. In analogy to Halkidi etal. [39] an instance of their index DB(K) may be defined as follows:

$$DB(K) = \frac{1}{K}\sum_{k=1}^{K} R_k \text{ for } K \in \mathbb{N} \tag{17}$$

, where

$$R_k = \max_{j=1,\dots,K,\, j \neq k} \left(\frac{S_k + S_j}{d_{k,j}}\right) \text{ for } k \in [1,\dots,K] \tag{18}$$

, and

$$S_k = \frac{1}{\sum_{i=1}^{N} w_{k,i}} \sum_{i=1}^{N} W_{k,i} \|x_i - \bar{x}_k\| \text{ for } k \in [1,\dots,K] \tag{19}$$

, as well as

$$d_{k,j} = \|\bar{x}_k - \bar{x}_i\| \tag{20}$$

For each cluster C_k an utmost similar cluster-regarding their intra-cluster error sum of squares-is searched, leading to R_k. Then the index calculates the average over these values. In contrast to the aforementioned cluster indexes, here, the minimal observed index indicates the best cluster solution.

2.3.3 Generator

The Monte Carlo method [40] relies on repeated random sampling to obtain numerical results. By running simulations many times in order to calculate those same probabilities heuristically just like actually playing and recording your results in a real casino situation: hence the name. They are often used in physical and mathematical problems and are most suited to be applied when it is impossible to obtain a closed-form expression or infeasible to apply a deterministic algorithm. Monte Carlo methods are mainly used in three distinct problems: optimization, numerical integration and generation of samples from a probability distribution.

In Monte Carlo simulation [41], a random value is selected for each of the tasks, based on the range of estimates. The model is calculated based on this

random value. The result of the model is recorded, and the process is repeated. A typical Monte Carlo simulation calculates the model hundreds or thousands of times, each time using different randomly-selected values.

When the simulation is complete, we have a large number of results from the model, each based on random input values. These results are used to describe the likelihood, or probability, of reaching various results in the model.

There are several approaches to generate more elaborated models of a given phenomenon. For example in financial markets, Quasi-Monte Carlo [42] is a new adaption that can get better performance than Monte Carlo. The conventional Monte Carlo uses pseudo random values to evaluate an expression. This method yields an error bound that is probabilistic creating a disadvantage. Other disadvantage is that many simulations require a high level of accuracy, and this is something that the conventional method can't promise.

Quasi-Monte Carlo is based in semi-deterministic instead of random sequences. The basic idea of quasi-Monte Carlo is to use a set of points that are carefully selected in a deterministic fashion.

We call these numbers quasi-random even though they are perfectly deterministic and have no random component.

In [43] QMC methods were used for estimating the value of a financial contract, such as an option, whose underlying assent fallow Black-Scholes model. The goal is to estimate:

$$\mu = E(g_p(S_1, \dots, S_t)) \tag{21}$$

, where $S_1, \dots, S_t$ are price. The function g_p is assumed to be square-integrable and it represents the discounted payoff the contract, and p is a vector of parameters (containing the risk-free rate r, the volatility σ, the strike price K, expiration time T, etc). The MC estimator would be:

$$Q_n = \frac{1}{n}\sum_{i=0}^{n-1} f_p(u_i) \tag{22}$$

To have the QMC estimator it is just needed to use a randomized QMC point set. So the u_i came from a randomized QMC point set, but being presented in the uniform distribution as MC. The only difference with MC is that with QMC, the observations $f_p(u_0), \ldots, f_p(u_{n-1})$ are correlated instead of being independent. With a carefully chosen QMC point set, the induced correlation should be such that the estimator has a smaller variance than the MC estimator.

Below is a figure of the results of comparing the Monte Carlo and the Quasi-Monte Carlo, where can be seen a faster convergence with the new method.

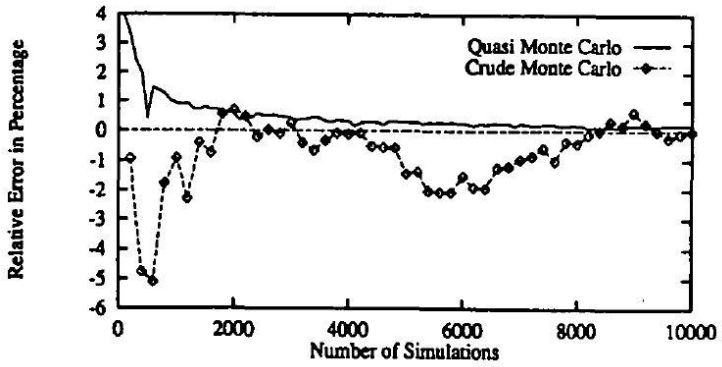

Figure 9 - Standard European Call Option **[42]**

3

Proposed Approach

The architecture and the procedures will be presented in this section. The work was developed on matlab, which is capable of producing useful statistical measures and visualisation tools. With the integration of SOM Toolbox [44], a library that represents an implementation of the SOM algorithm, it was possible to program with a graphic and calculus tool which uses the self-organizing maps model. This library is composed by tools to generate many SOM model statistics interpretation, as an example, the UMAT view.

It was also added a module to flooding analyse the UMAT result. This module produces some visual and text files results that bring a new way of showing the results of SOM algorithm.

3.1 Architecture

3.1.1 General Architecture

The proposed architecture, illustrated in figure 9, can be defined as a 3 stage system: the input layer, the main program and the output layer

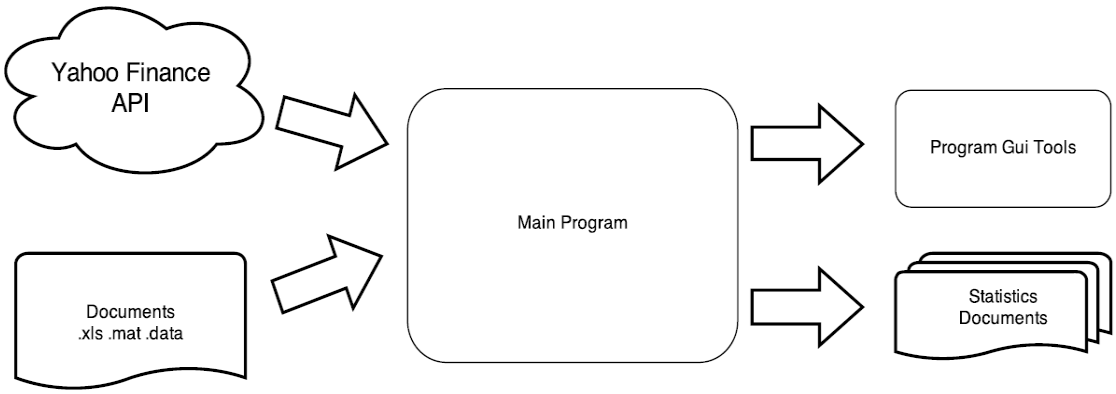

Figure 10 - Proposed Architecture

The input layer uses the Yahoo Finance API and also the Documents of type .xls, .mat and .data (these documents should be formatted by the program protocol). The main program layer corresponds to the Main Program component of the figure, which defines the algorithms and calculus researched in this thesis. Finally the output layer that is defined in the figure 9 as the Program GUI Tools and Statistics files.

This type of architecture gives the possibility to work with both local data and cloud systems. It is propitious to work as a modular system that adds or removes certain algorithms but it wasn't the focus of this research.

3.1.2 Work Flow Model

For a better understanding of the program concept, the figure 10 illustrates the work flow model. The work flow consists of various steps that the program has to process before achieving the final results, those were the statistic files produced or the plots that the visual tools generated.

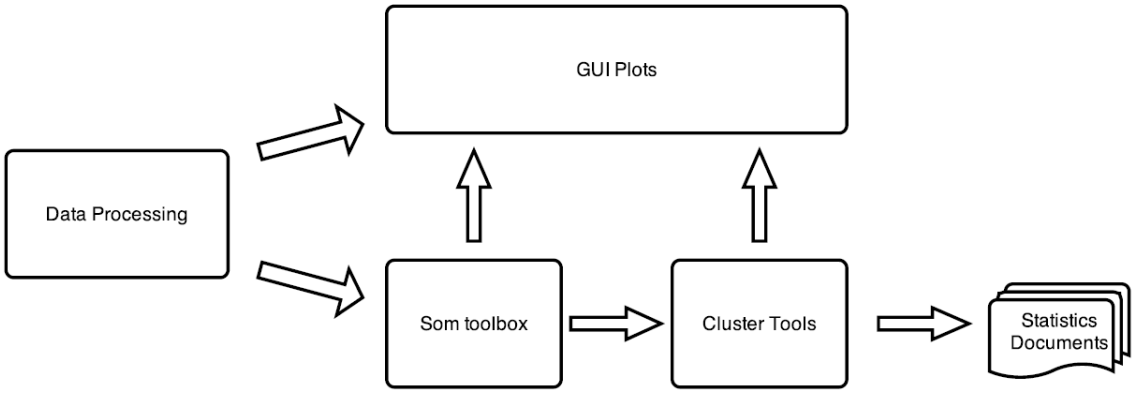

Figure 11 – Program Workflow

As it can be seen in the figure 9 the system starts by processing the input data. The input data comes from Yahoo services or from an input file. The data has to be distributed by the program global variables, to allow the GUI Plots tools be able to report statists graphics. If the next step is the usage of the SOM algorithm, the data has to be normalized before.

The SOM Toolbox corresponds to the SOM toolbox library. In this component we use the API of this library to generate the visual tools that will be showed in the GUI Plots component. We may stop the process right here or we may continue for the clustering tools.

Finally the GUI Plots components display the processing and results generation. The cluster processing technique generates the data for plot with the GUI Plots and also the documents reports with useful information for clustering analyses.

3.2 Source Data

Most of the data comes from the Yahoo Finance Cloud System. This system is commonly used by investors and offers a free use of their API [45]. With this, it was possible to get a stock evaluation with detailed information about the price.

The Yahoo Finance System is limited by a thousand queries per stock action, but it hasn't limitations with the size of the data. The connection that sends the data uses a stream channel that can be read to a variable in the code. This part of the code is done by the function *get_hist_stock_data.m* where the java.io library is used.

The query used asks for the high, low, open, close, volume of a stock daily value. It also has to be given the initial and final year date of the period to be analysed to process the query. In the study, the close value of the stock was used, that is the last value for the day in question.

Another way of adding data to the system is the use of default scripts created during the thesis elaboration or by using a file with the correct input format. The default scripts of the thesis use data from the Yahoo cloud services but data is uploaded to a file, it allows any other provider to be used but still has to be with the correct format to allow the program to read it.

So to use the data from a file, the data must be structured by the following specifications:

1) The top of the columns should have the name of the companies or the name of the calculus;

2) Each line of the file has to have the value for each variable

3) The first column is reserved to the date. The plots that show the date are programed to work with days by line and the date should be in the format: 'yyyy-mm-dd'.

3.3 Characterization Method - Min-Max Normalization

Given a time window of n days and an array of values arr we have $normalizeByMinMax(j)$:

$$normalizeByMinMax(j) = \frac{\text{arr(j)} - \min(arr(j-n,\dots,j))}{\max\big(arr(j-n,\dots,j)\big) - \min(arr(j-n,\dots,j))}$$

So it has to calculate the minimum and maximum value of the subset of the array with length n. With these results it is calculated the subtraction of the position j value by the minimum of the subset array and also the subtraction of the maximum value calculated by the minimum value. With the previous two results, we divided the first result by the second result.

This normalization gives a local perception of the prices variance. The SOM may be able to detect states with big and low variations in the price. With this state detection it will be possible to characterize the periods being analysed and obtain useful information to the economist or stock market traders.

3.4 State Identifier / Clustering

The chosen technique to classify the states of the stock market time series was the SOM algorithm. Based on this, we started to classify the first testing data set and it was noticed that the information by BMU didn't generate stable states with enough points. By analysing the trajectory over the UMAT it was noticed that some neurons should be considered as one. The result of these analyses leads the thesis to study a clustering algorithm that groups these sets of neurons.

To find the clusters in the UMAT it was used a two process procedure. The first procedure was to find interesting zones in the map that may represent a cluster, and the second was to explore those zones in order to decide which neurons may also be considered as the same cluster. For the first process we used the local min criterion and for the second process we used the flood fill algorithm.

3.4.1 Local Min

The local min is seen as a mathematic representation of a minimum for a certain set of a function. The local min can also be the minimum of the function but for that it has also to satisfy the global min restriction. The global restriction for a point x is:

$$if\ f(x*) \leq\ f(x) for\ all\ x\ in\ X$$

In the case we want the restriction for the local min value, it is needed to consider the X a sub set function.

Local min was important for this study because it is a way of studying the minimal points in a sub set. This type of technique is also used in some clustering algorithms referenced in [46]. This has a particular interest for the study when the finding of some points of minimal distance is required, meaning points of high proximity values. The proximity is one of the clusters formation criterions.

The local min criterion was the adequate approach to this process requisite. The local min in a U-Matrix corresponds to a local minimal distance. This local

minimal distance could be seen as the middle point of a cluster and by detecting all local min of a U-matrix we can have the maximum number of clusters considered by the distance between the entities. The density analysis has an important role in clustering but not for this study. The U-matrix represents only a distance matrix which doesn't take in consideration the information about the density of samples by neurons.

To get the local min of the U-matrix we have to consider all the positions of the UMAT. For this we will test each position of the U-matrix to see if it is a minimum. If it is a minimum we mark that position with true value and if not we mark it with false value. We called this procedure in *processMinimos* fuction:

```
function [ minimalMat ] = processMinimos( Umat )
     for  x : allPositions(Umat)
         if(isLocalMin(x, Umat)
             minimalMat[x] = true;
         else
             minimalMat[x] = false;
         end
     end
end
```

To see if a position in the U-matrix is a local min, the position has to be compared with all its neighbours. If the position value is minimal comparing to its neighbours, it is returned the true value for this position, otherwise, it will be returned the false value. So for this we have *Minimo* with the minimum position and *Mat* with the U-matrix.

```
function [ bool ] = isLocalMin(Minimo, Mat)
     for x : getNeighbours(Minimo, Mat)
         if( Mat[Minimo] < Mat[x])
             counter++;
         end
     end
     if(counter == nuberOfNeighbours) bool = true;
     else bool = false;
     end
end
```

The neighbouring function varies with the topology of the map. The study used the hexagonal topology. So for this the number of neighbours was six. This may vary but the *getNeighbours* fuction was only programed for this type of topology that was the only one used in this study.

3.4.2 Flood Algorithm

The flood algorithm is used to join the entities or clusters in one entity. This algorithm is executed with the result of the local min algorithm, so the starting points to make the flooding are *localMins* found before. The flood will be executed to a limit that corresponds to the boundaries defined by the user. In the pseudo code of the algorithm, showed in the box, corresponds to the mean distance of the U-matrix.

```
function [ finalClusters ] = flood( localMins )
    allClusters = LocalMins;
    finalClusters = [];
    meanValueUmat = mean(mean(Umat));
    for  cluster : allClusters
        cluster = floodACluster(cluster, meanValueUmat );
        addCluster(finalClusters,cluster);
    end
end
```

The result of this function represents a matrix with the same dimension of the U-matrix but instead of distance values each position as the number of the cluster that is associated to. With this matrix it is done a correspondence of the neuron with the position and the neuron is associated to each respective cluster.

The *floodACluster* function appears as an auxiliary function for the flood algorithm. This function verifies each position of the *cluster* and flood to the boundary established, which is represented by the *value*. The function *getNeighbours* is the same function used by *isLocalMin* and has the same properties.

```
function [cluster] = floodACluster(cluster, value)
    newCluster = [];
    for x : cluster
        neig = getNeighbours(x)
        for possb : negh
            if(possb < value)
                add(newCluster,possb)
            end
        end
    end
    add(cluster, newCluster)
end
```

When the process ends this function reaches to a new cluster or no change at all. In the project it is checked if any change was made, and if not, the flooding process stops because the result has been reached.

The result may then be analysed by some statistical and visual tools. These tools will be presented in further sections.

3.4.3 Missing Classification Procedure

After the result processing by the flooding algorithm, some neurons may be not yet classified, and based on the previous result a third procedure is executed to classify the missing neurons. To do this, we used the nearest neuron of the already classified neurons.

We have $C = \{c_1, c_2, \dots, c_n\}$ that represents the group of the classified neurons. The group of neurons $U = \{u_1, u_2, \dots, u_j\}$ represents an unclassified group. So $u_i$will assume the classification c_j where j represents the:

$$\min(\|C - u_i\|) \tag{23}$$

With this procedure all the neurons of the map will have a classification.

3.5 Study of the generated states

For this thesis the state/cluster studies were approached by three methods. It was difficult to understand what each state was representing and at the same time to establish a certain behaviour to the cluster. So, for this problem, it was proposed some ways of studying the generated clusters with simple calculus and projections of the obtained clusters.

3.5.1 State relevance of the data set.

To understand the state importance, it was measured the number of entities that each state has. For this to happen, the result cluster matrix has to be analysed with a histogram that has the number of entities per cluster.
The figure below is an example with the Iris dataset data. The cluster 6 in this test is really important, facing the fact that it has 90 entities. Through the analyse of this results, it is automatically possible to understand which clusters are the stable states, and which clusters represent micro states.

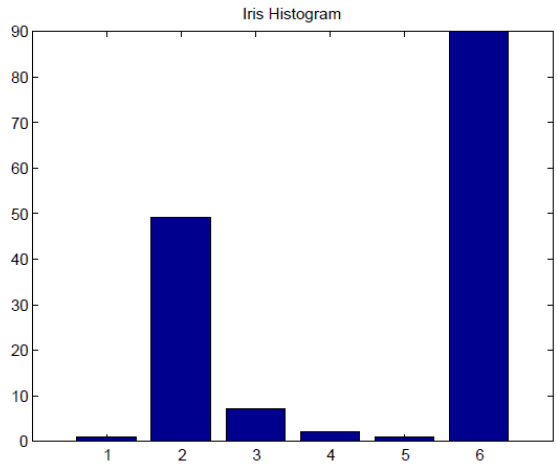

Figure 12 – Number of Entities by Clusters

3.5.2 Graph transition measures

This measure is useful to see the transitions between the states. Comparing it with the graphics of the markets mean value it is possible to understand each state or transition that is happening during a certain period. This projection turns out to be very useful in understanding the clustering result, because the clustering result is compared with the time series.

Figure 13 - Path between Clusters example

In the figure above, represented with blue colour are the states path, in which the vertical axis represents the cluster id, and the jumps over the y axis represent the transition of states. In the example the first cluster is represented by the id 33. It was only visited again during the middle of the year 2001.

3.5.3 Centroids distances measures

The centroids distance measures is used to measure the distance between the clusters, represented in cluster 13. This index is useful to compare transitions between far clusters with time series events.

This measure can also be seen as an instability measure. When compared with the time series we can see when there are big distances between clusters centroids where there is a big difference in the value variance between the two clusters. In the figure below there is a representation of this measure with the blue colour. The green line represents the mean value of the distances. The periods that don't have representation associated, it mean that the cluster hasn't changed, so there is no distance associated with a cluster change.

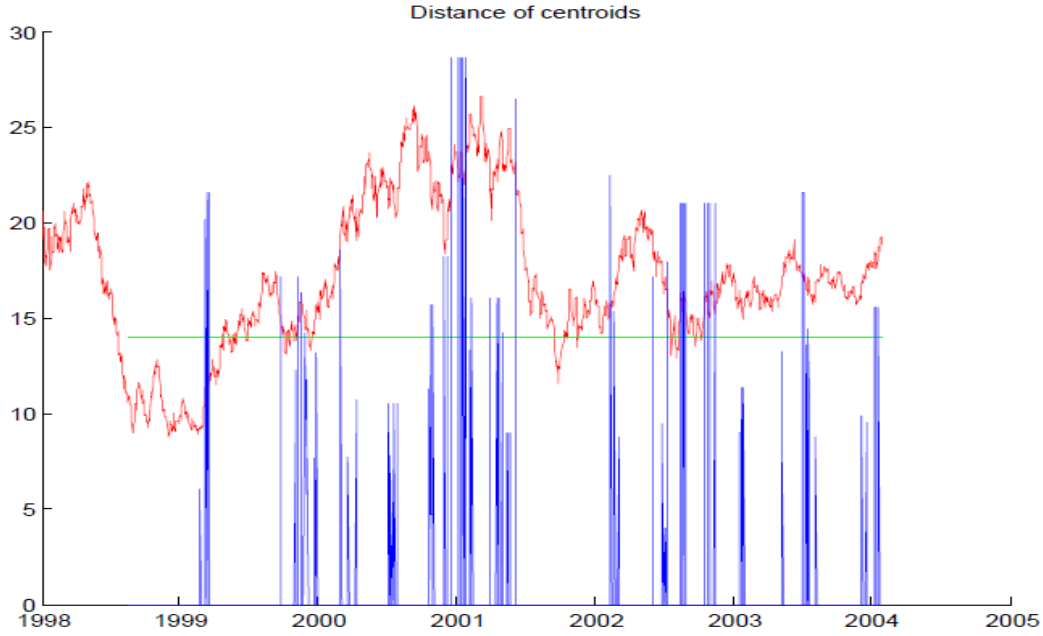

Figure 14 - Distance Between Clusters Example

3.6 Generator

The generator used for this study works based on a clustering result. The idea is to do a simulation with the states obtained by the SOM instead of using a total random generation like the Monte Carlo approach.

Based on the calculus of the current day variance to the previous day, each day will be grouped by the respective cluster and then the mean values and the standard deviation values (that is calculated based on the variance) from the cluster that were calculated.

With the previous result it is possible to make a simulation with the clusters information. Knowing the state order and state change, we only need to use the mean and standard deviation by state and generate a new value. To generate the new value, the next calculus was used:

$$day_{Varience} = normrnd(\overline{m}, \sigma)$$

, where the *normrnd* is a matlab function to generate a random value based in a normal distribution with the mean ($\overline{m}$) and the standard deviation (σ) from the respective cluster. After doing this we can calculate:

$$day_{Value} = dayBefore_{Value} * day_{Variance}$$

,where the $dayBefore_{Value}$ represents the stock simulated value of the day before and the result will be the current day stock simulated value. It has to be considered that the first iteration didn't have a $dayBefore_{Value}$ so the real value is used instead.

4

Clustering Technique Validation

This part of the thesis was important to validate the clustering algorithm being used. The objectives of these tests were to measure the precision of the clustering algorithm by clustering with some well-known problems of clustering.

Two well-known data sets for this purpose were used: the wine dataset [47] and the iris dataset [48]. It also was used a synthetic dataset based on generated values to validate the clusters results with a time series data. This validation gave promising results, and also confidence to use the algorithm in the next section, that was testing with the financial data.

To measure the clustering quality it was chosen the Calinski-Harabasz index. The presented results were the ones that maximize the result of the index. This estimation will decide the number of clusters that the algorithm will produce for the study.

4.1 Wine Data Set

The next image shows the clustering algorithm processes with the wine dataset. The first transition represents the formation of the Local Mins, and the second transition represents the clusters formation by the flooding technique. It can be seen four large blue zones that represent the four more important (populated) clusters.

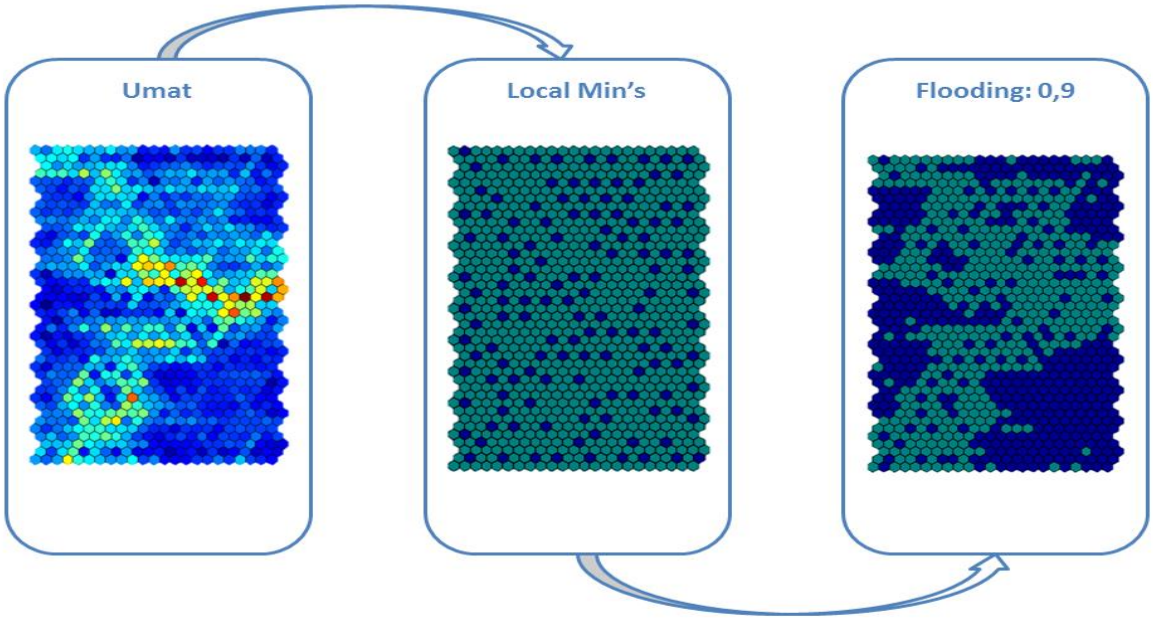

Figure 15 - Flooding Evolution Wine

The image above used a flooding parameter (deep value) of 0.9. That represents the best flooding value based on the Calinski-Harabasz index. With this flooding parameter, the number of clusters calculated was 27 and only 4 clusters presented at least 14 entities. The other clusters represent single neurons located in a micro clustering zone. These zones may also have the presence of outliers and may be distant from big and high density clustering zones. The next image shows the calculation of Calinsky-Harabasz index for the different deep values, in which the deep with the best index value was extracted.

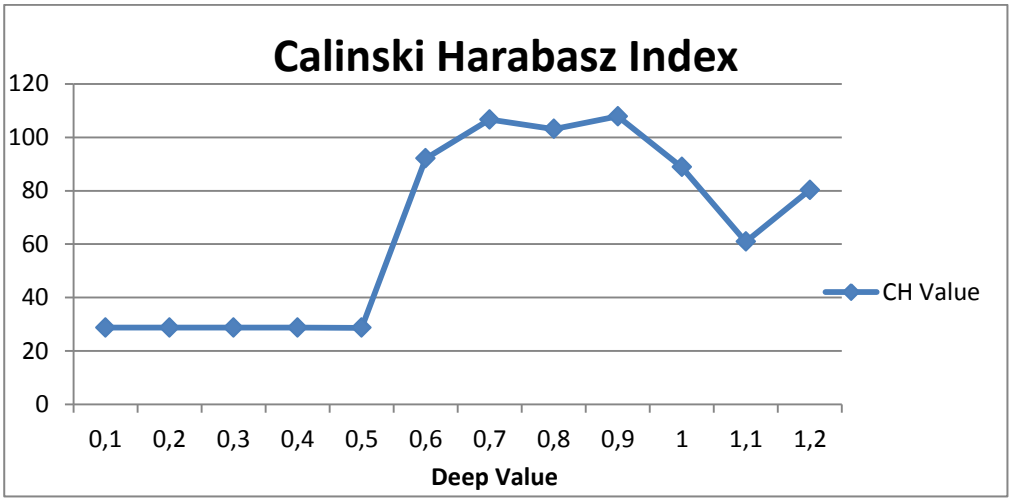

Figure 16 - CH Graphic Wine

To analyse the precision of the method, a confusion matrix, represented in the next table, was used. With this analyse it is possible to see the predicted class entities and the actual class entity. We can see that the predicted class presents a reduced number of wrong predicted entities. By a simple calculus of sum of the wrong predicted entities and dividing it by the total number of entities, we have a percentage of error of 2.8 %.

Confusion Matrix: 0,9 Deep		Predicted Class		
		Class 1	Class 2	Class 3
Actual Class	Class 1	57	2	0
	Class 2	0	67	3
	Class 3	0	0	47

Table 1- Confusion Matrix Wine

It can be concluded that the clustering algorithm tested with the Wine Dataset resulted in a very good solution. The 5 units that had a wrong prediction were already bad classified by the SOM algorithm because within some BMUs that the SOM presented, entities of different classes were found. This resulted in error propagation for the clustering algorithm.

4.2 Iris Dataset

In the next figure it is possible to see the evolution of the clustering algorithm in the Iris Dataset. We can see U-matrix, the local min's and finishing with the flooding algorithm result. It is possible to see the evolution from the Umat to the local min's where the zones with closer neurons form more local min's. On the transition from the local min's to the result of flooding it is possible to see of the joining the zones with more density of local min's.

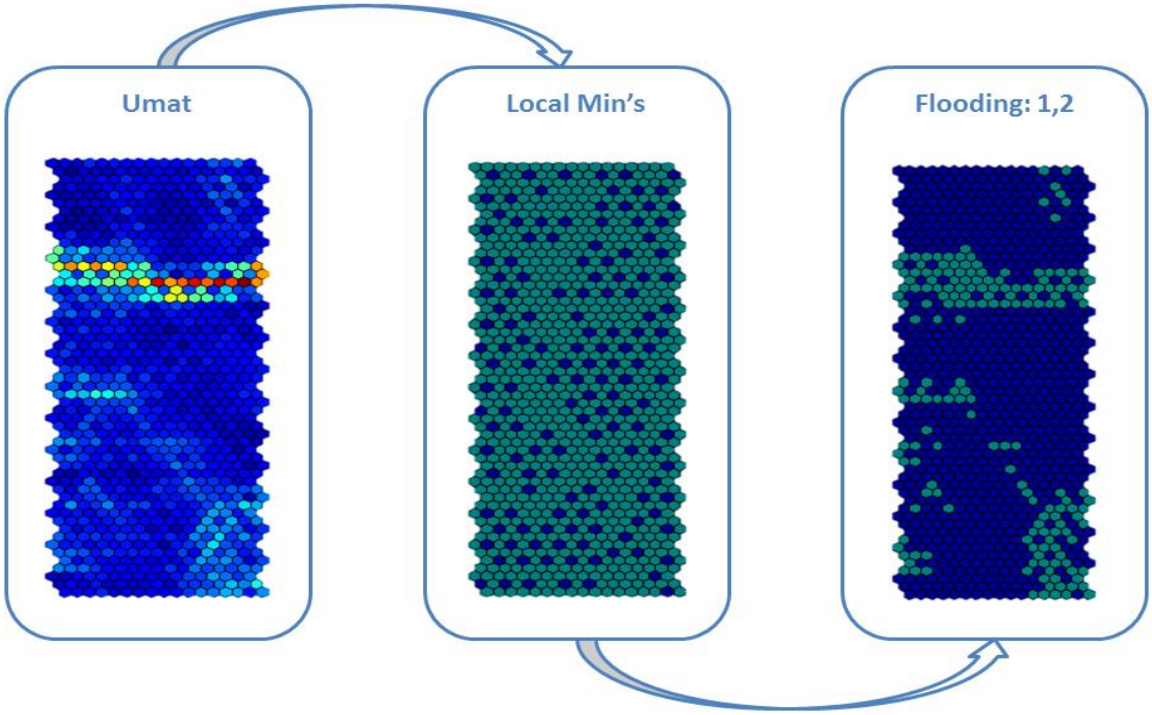

Figure 17 - Flooding Evolution Iris

The value 1.2 maximizes the value of Calinski Harabasz index and generates 6 clusters. This cluster measure has produced two important clusters that correspond to the cluster two and six. The cluster 2 has 49 entities and the cluster 6 has 90 entities. This deep measure wasn't the best because it joined 2 classes in one cluster that were represented by the class Iris Versicolour and Iris Virginica. This two classes were easily mistaken by its similarities proving that the cluster measure should have a higher value with a less deep measure.

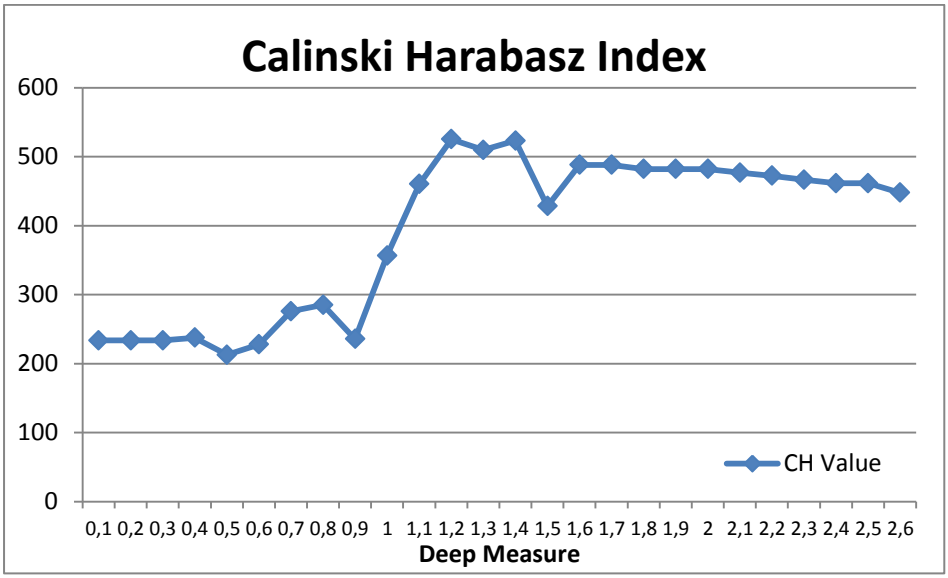

Figure 18 - CH Graphic Iris

The next table presents the true positives and false positives of the Iris Dataset. The error percentage was 26.7%. This results compared with the Wine Dataset is less precise. It just reflects the difficulty of clustering the Iris Dataset but also the need of a self-adapting clustering algorithm for clustering certain zones. With a less deep measure when clustering the Iris Versicolour or Iris Virginica classes, the separation of the two classes would be more expressed, reducing the error of the algorithm.

Confusion Matrix: 0,9 Deep		Actual Class		
		Iris Setosa	Iris Versicolour	Iris Virginica
Actual Class	Iris Setosa	50	0	0
	Iris Versicolour	0	50	0
	Iris Virginica	0	40	10

Table 2 - Confusion Matrix Iris

4.3 Synthetic Time Series

It was generated a synthetic dataset to test the proposed approach when using the cluster technique in a time series dataset. The construction of the dataset used 4 different normal distributions, which generate 4 different trends in a time series. Based on this, 11 data series were generated that would serve as an input to the clustering algorithm.

The 4 different normal distributions had the flooding parameters:

$$f(x) = norm(\overline{m}, \sigma) \begin{cases} \overline{m} = 1.4 \ \sigma = 20 \ if(1 \leq x \leq 500) \\ \overline{m} = -1.4 \ \sigma = 20 \ if(501 \leq x \leq 1000) \\ \overline{m} = 1.4 \ \sigma = 20 \ if(1001 \leq x \leq 1500) \\ \overline{m} = -1.3 \ \sigma = 20 \ if(1501 \leq x \leq 2000) \end{cases}$$

, where $\overline{m}$ assumes the mean value and σ assumes the standard deviation parameter. By analysing the function we will have a positive variation during two sliding windows that will be the first 500 values and between the 1000 and 1500 values. The other two sliding windows generate numbers with negative variation.

The resulting time series dataset is presented in the next figure. As it can be seen it is impossible to detect any pattern by human observation.

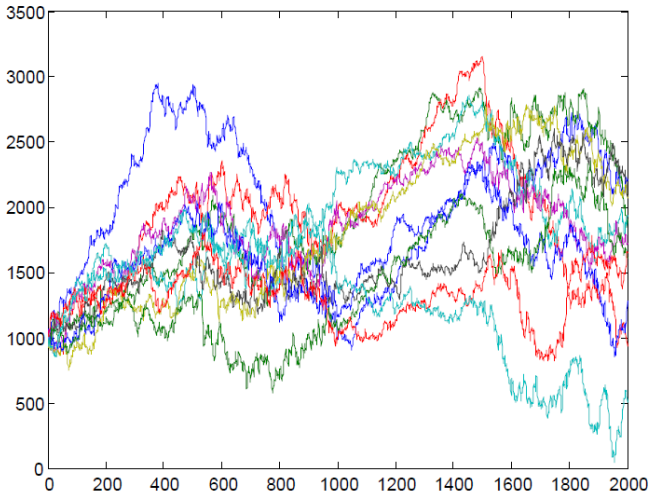

Figure 19 – Synthetic Time Series

The result of clustering with the SOM algorithm is presented in the Figure 20. To generate this image, the mean of the previous time series was used, representing the single time series in the figure. By this it is better understood the trends of the time series and it is possible to project the clusters in the time series. The clustering result is represented over the time series, where different colors represent different clusters.

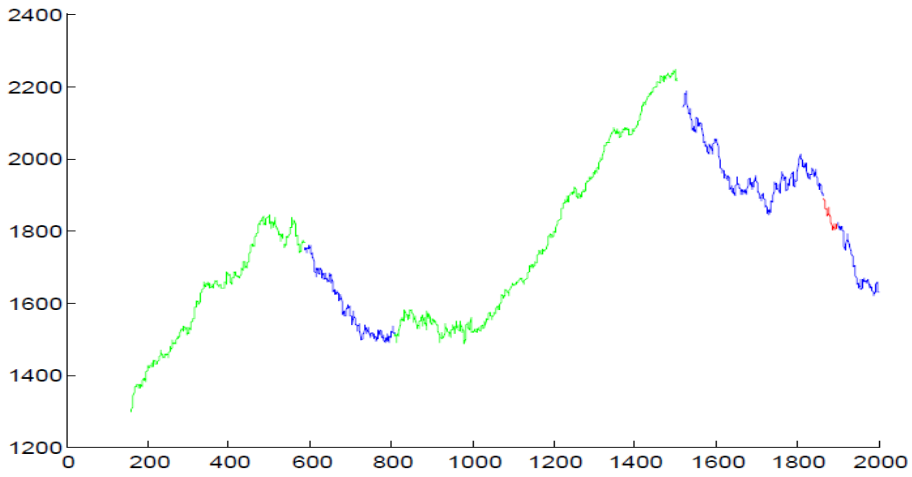

Figure 20 - Cluster over synthetic time series

The clustering showed uses a deep measure of 1.3, and is presented with 7 clusters. Four of the 7 clusters have less than 12 entities, so they weren't represented in this graphic.

The Calinski-Harabasz index is used to choose the best deep measure, but the best deep value wasn't the result of the CH index. This type of analyses makes the measure precision to be questioned, yet it is not enough to put this measure apart without testing it with a real data set.

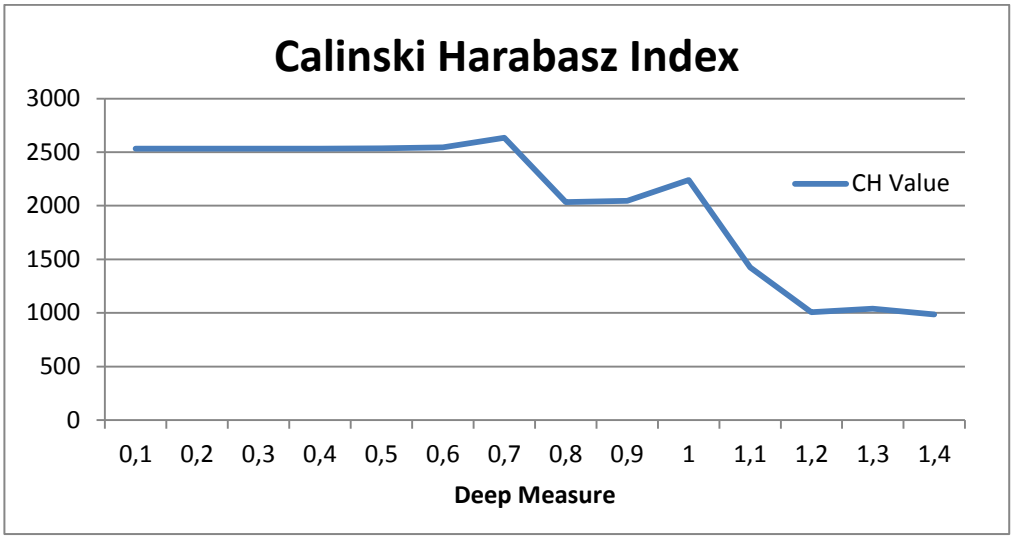

Figure 21 - CH Synthetic Time Series

5

Application to Financial Data

This chapter will present the results of clustering with financial data. The data used in this paper is based on the sp500 index companies. The companies of this index correspond to the 500 biggest companies in US, making the usual investors picks for analyses and trading. The period being studied starts at 2004 and ends in 2014. In this period it is possible to see the impact of the most recent crisis which started in 2007 and started recovering at 2009.

For this datasets it will be processed a selection of two sets with correlated and inverse-correlated stock values. This will allow seeing the impact of the states produced during the pre-crises, crises and pos-crises. It was also analysed studies of states importance and analyses of distances transitions.

To finish it will be analysed the result of generating a price time series based on the previous analyses of the clustering algorithm.

5.1 Correlation Method

The correlation method is a calculus to measure the relation of two time series. For this purpose it was used a simple form that looks at the two time series trends to see if they are similar to each other.

The calculus measures the correlation based in a time interval. The higher the value the more correlated the time series were, using the sum of the scores available. The $min\,T$ represents the minimal length for both time series and the time is the length of the time interval that will be measured in the score function. So by definition as much as it is scored the better the correlation is.

Based in:

$$\text{score}(\text{x},\text{y}) = \begin{cases} 1, & \text{trend}(\text{x}) = \text{trend}(\text{y}) \\ 0, & \text{trend}(\text{x}) \neq \text{trend}(\text{y}) \end{cases} \quad \textbf{(24)}$$

We have:

$$\varphi = \sum_{t=time}^{\min T} score(x_t, y_t) \quad \textbf{(25)}$$

This method was designed to look for similar trends behaviour and not to the variance of the data set. The impact of big and small variations in the time series will be analysed by the SOM algorithm, and based in that, track particular states.

5.2 Case Study One

The result of the previous method is used to choose the company that was more correlated with the rest of the data set, resulted in the Simon Property Group Inc. From this company it was selected the top 10 companies that presented more correlation with it, giving: Macerich, Boston Properties, Public Storage, Simon Property Group Inc., AvalonBay Communities, Inc., Prologis, Equity Residential, Apartment Investment & Mgmt, Ventas Inc., HCP Inc.. The next graphic illustrates the sock value time series of each company.

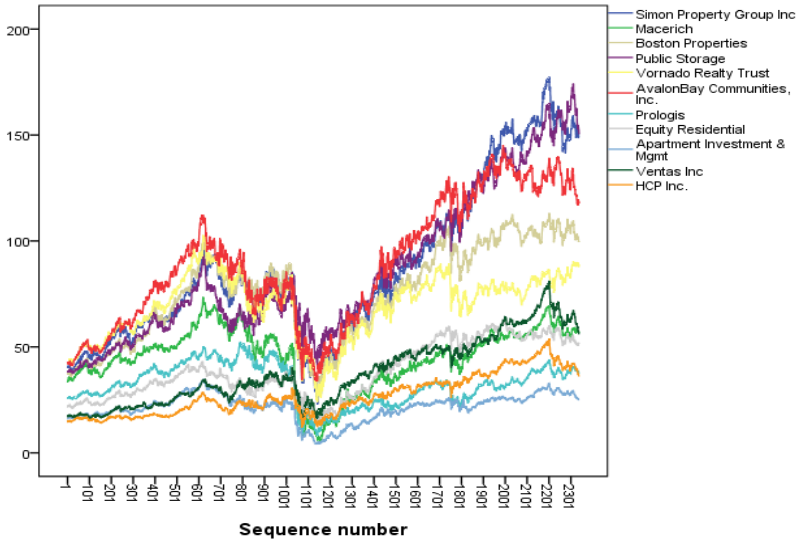

Figure 22 - Correlated Companies

In the previous image it is possible to see that the trends of each stock are very similar with each other. Each company represents or is connected (only two are just connected) with the financial sector of real estate investment industry. This shows how the companies of real estate represent the same behaviour and where affected by the same factors.

This companies still survived to the crises of 2007 but it is evident the huge crises effect over these group. With this result it was interesting to see the states produced before and after the crises, and see if there is a presence of a state that may be associated with a pre-crises or pos-crises pattern.

So with this dataset we proceeded to the SOM classification with the flooding parameter adjusted to the max Calinski-Harabasz index. After that, it was possible to analyse the results with the state study measures (the three approaches presented in the 3.5 section).

The next image shows the SOM result with the U-matrix and the component planes. With a brief analyse it is possible to see the similarities between certain component planes. It is also possible to see some well-defined zones in the U-matrix.

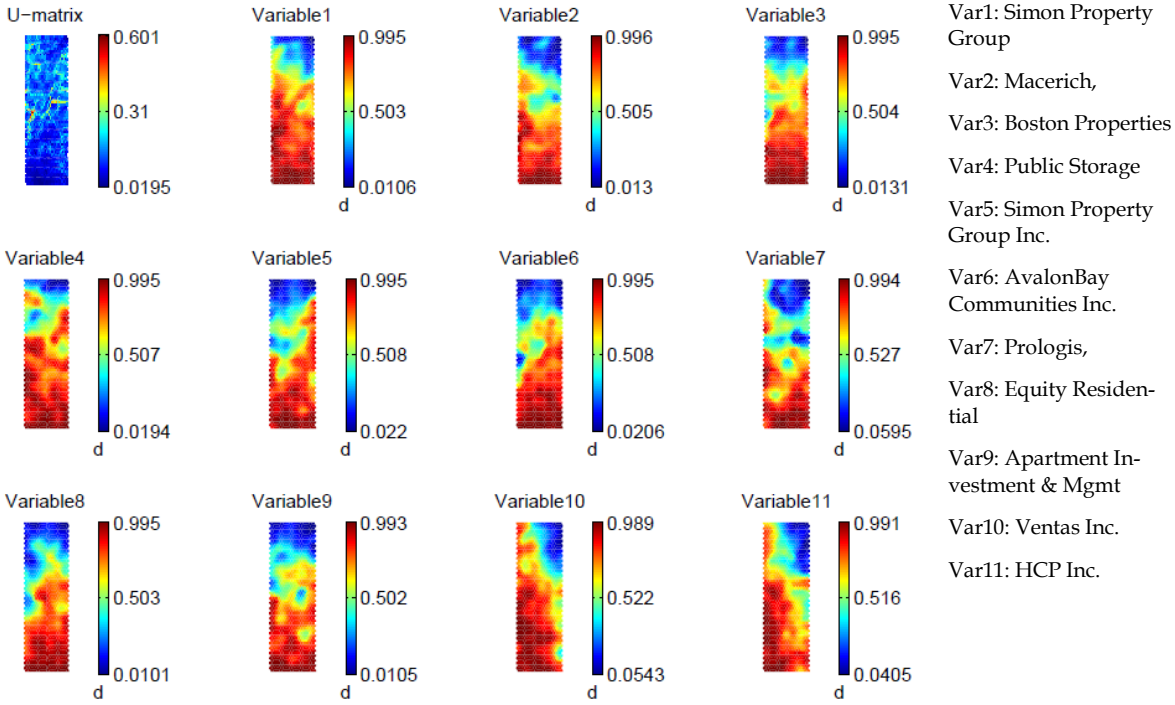

Figure 23 - UMAT and Component Planes Correlated Markets

In the image bellow it is possible to see the impact that certain cluster had in the time series. Cluster number 33 has more entities than all the other clusters together. In the representation of the time series it is possible to see that the cluster 33 appears during a long period and this period corresponds to positive variance of the stock market value.

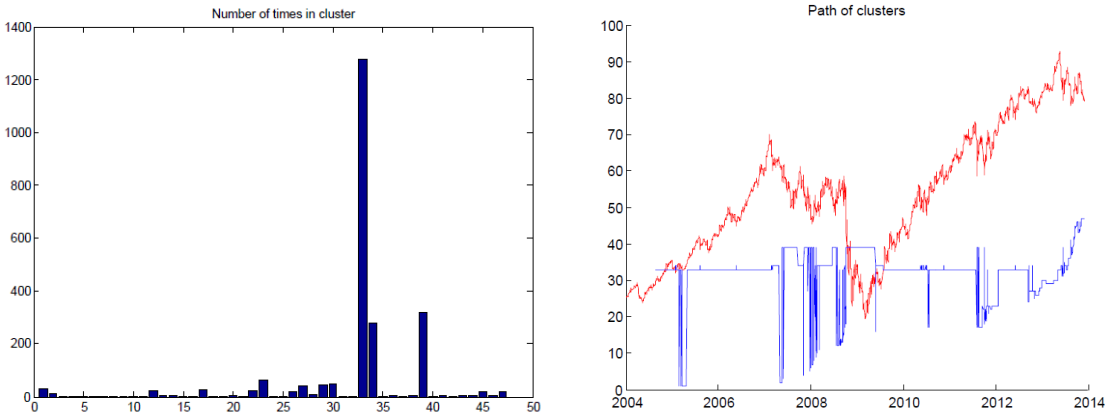

Figure 24 - Clusters Size (left side) and Clusters Transitions (right side)

In the next image it is possible to see the distance between the clusters jumps and also a colour graphic that represents the clusters by colour in the

time series. In the distance graphic in the next figure it is possible to see a great variance of the stock value between crises states. The clusters in crises periods change more often and to longer distances comparing to long periods of no change or of small distant clusters. Between the year of 2008 and 2009 (crises period) the distance measure assumes very unusual values, showing the instability of the states.

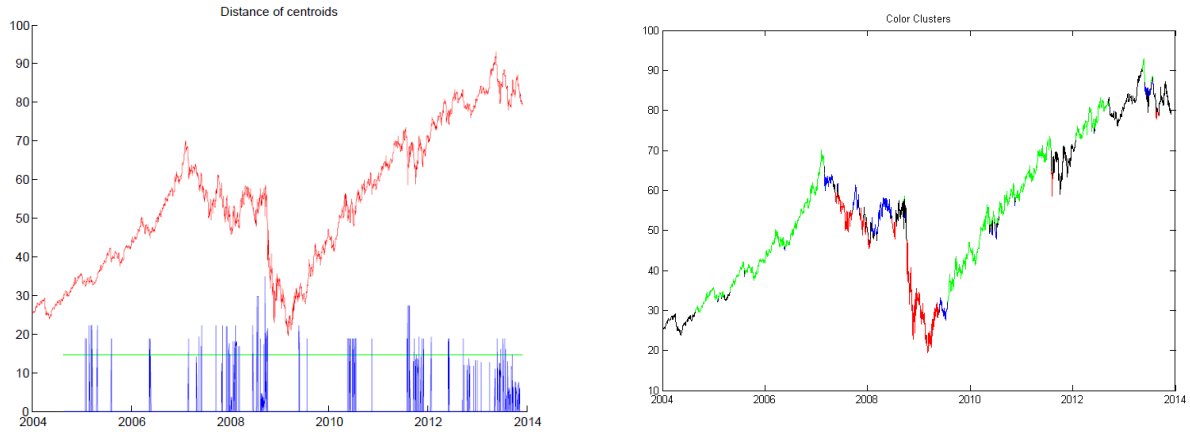

Figure 25 - Distance Clusters (Left Side) and Colour Clusters (right side)

In the colour clusters graphic in Figure 25 - Distance Clusters (Left Side) and Colour Clusters (right side) it is possible to see the representation of the clusters that have more entities directly in the time series. In the figure is represented the cluster 33 (green coloured), the cluster 39 (red coloured), the cluster 34 (blue coloured) and the rest of the clusters black coloured. It is possible to see that a great part of the time series is represented by the cluster 33 that represents most of the period where the time series had a positive variation.

Analysing the crisis period in the colour clusters graphic its notable the importance of the cluster 39 and cluster 34. During this period these clusters were active by a longer and repeated period comparing with the rest of the time series.

The next step was a simulation with the clusters. To do this it was used the technique presented in the proposed approach. The time series is represented by the mean value of the stocks used, and then by the clusters it was obtained a mean and the standard deviation values. The generated a time series that is presented in the next image.

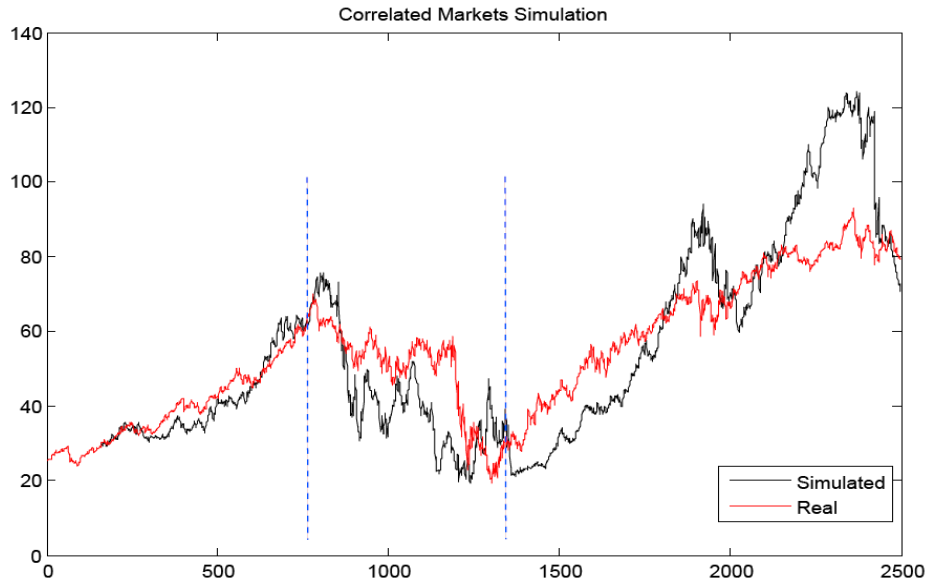

Figure 26 - Correlated Markets Simulation

In the previous image it is possible to see how close the two time series are and also how equal both presented trends are. Another interesting factor is the behaviour of the simulated time series during the crisis period (it is represented between the two blue dotted lines), where it is possible to see a high value of instability comparing to the period before crises.

In the next image it's shown an experiment that was generating 15 time series. Analysing the result it can be seen an instability behaviour during the crisis period (between the 750 and 1400). On other hand the time series experiments a stable behaviour when it is not in the crisis state.

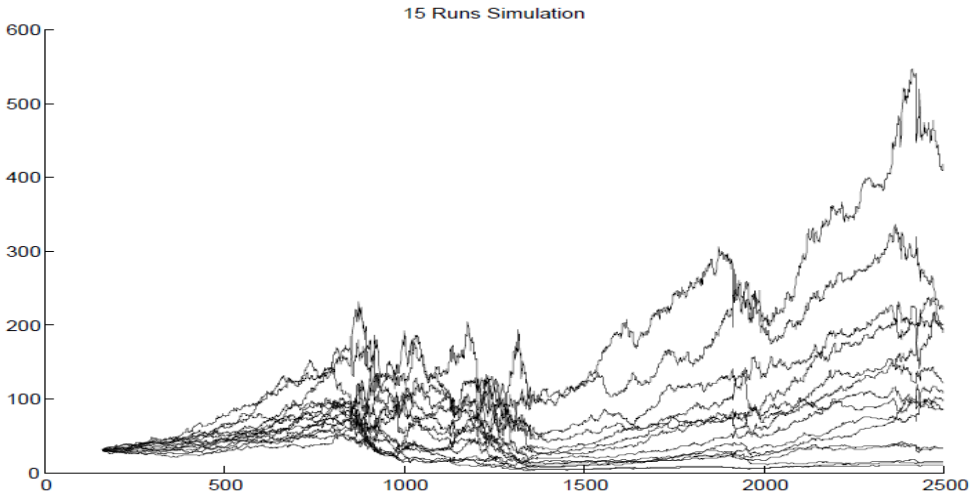

Figure 27 - 15 Runs Simulation Correlated Markets

5.3 Inverse-Correlation Method

The Inverse-Correlated method uses the same principle of the correlated method but the score function changes. So for this method we have:

$$\text{score}(x,y) = \begin{cases} 0, & \text{trend}(x) = \text{trend}(y) \\ 1, & \text{trend}(x) \neq \text{trend}(y) \end{cases} \quad (26)$$

So the score function sets the value to 1 when two trends have a different behaviour. With this it will be possible to track series that are inverse-correlated with a specific sock value series.

This type of analyses is also important because of its potentiality to project a prediction of a crisis state. It will be possible to detect through the analyses of behaviour of inverse-correlated markets before reaching a crisis state.

5.4 Case Study Two

Based on the inverse-correlated method it was selected the most inverse-correlated company from the sp500 index. The company selected was Dell Inc. that represents the Information Technology sector from the Computer Hardware industry. The top 10 correlated companies were Mylan Inc., Brown-Forman Corporation, Lilly (Eli) & Co., United Health Group Inc., Jabil Circuit, People's United Bank, Gilead Sciences, SCANA Corp, The Hershey Company, Nucor Corp. Most of the companies were from the health care sector.

In the figure it is showed the referenced companies.

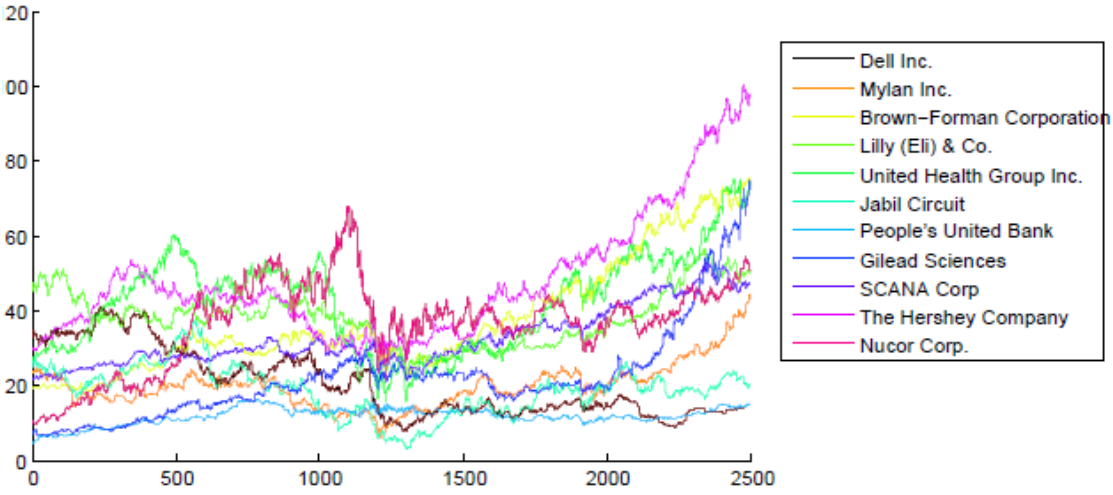

Figure 28 - Inverse-Correlated Time Series

Comparing the correlated data set with the inverse-correlated dataset it is possible to see a lot of differences. The Dell Inc. that was the selected stock,

doesn't seem to have any type of similarity with the other stocks. With a closer analyse it is possible to see, during the 2007 crisis, most of the stock's values went in a negative trend.

With this data set, the SOM algorithm and the cluster algorithm were executed. The result of the SOM with the component planes is shown in the next image. It is possible to analyse how different the component planes are from each other. Also, by analysing the U-matrix it is possible to see how the clusters are well defined.

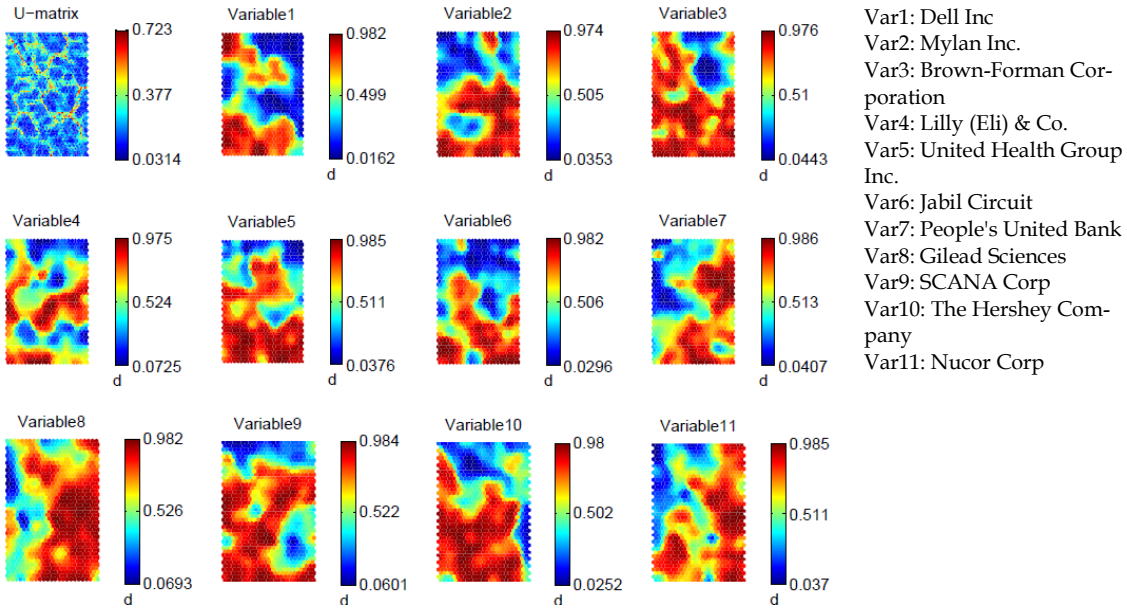

Figure 29 - UMAT and Component Planes Correlated Markets

The flood parameter that presented the best Calinski-Harabasz index was 0.7 deep. This value generated 176 clusters using the clustering method. Based on this, it was performed a state's study for better understanding of the flooding results.

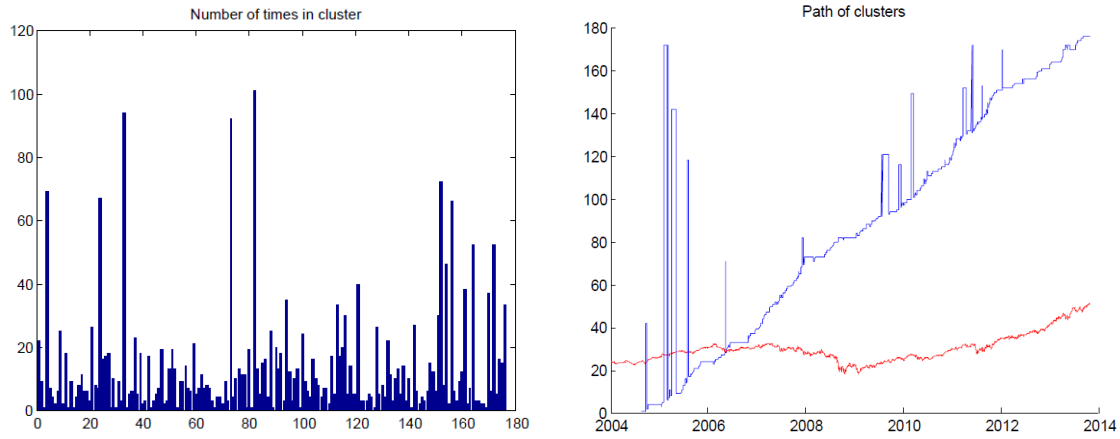

Figure 30 - Clusters Size (left side) and Clusters Transitions (right side)

In these results it's quite notable that there are more clusters but less data by cluster. In the previous image it is possible to see that there are some clusters with almost 100 entities of data, but there are many with less data. In this test Calinski-Harabasz didn't give the intended deep value and pointed to a low deep measure. The right graphic from Figure 31 shows that the clusters aren't really stable and that the data is always changing from cluster to cluster.

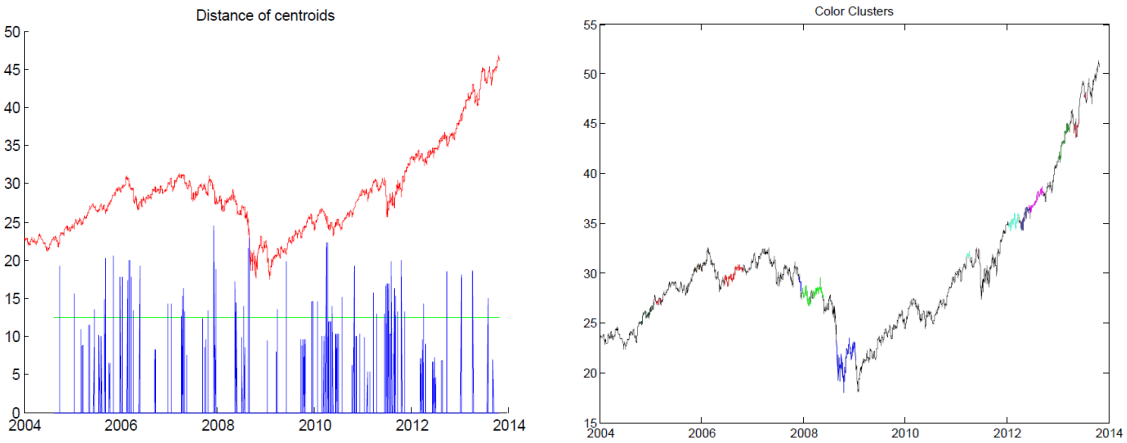

Figure 31 - Distance Clusters (Left Side) and Colour Clusters (right side)

In the Figure 31 it was showed the results of the distance measures and top 10 clusters combined with the time series. A low deep measure originated by clusters with few entities that didn't express too much in the stock series. The distance measure is still a good measure to detect crises periods. In the stock series it is possible to see the high distance transition during the crisis.

As in the previous study, it was generated a price time series based on the clustering obtained by the SOM algorithm. The result is represented in the next image.

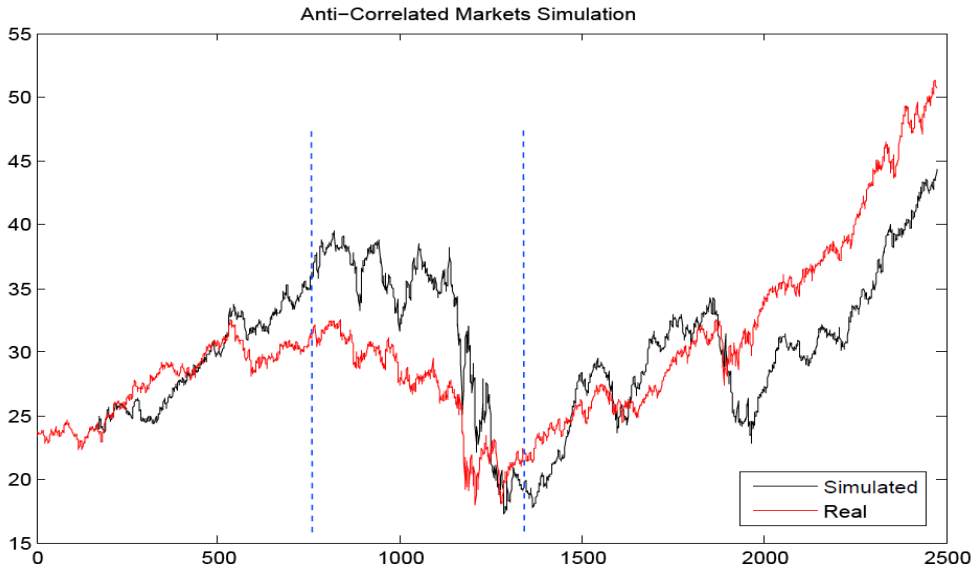

Figure 32 – Inverse-Correlated Markets Simulation

The 2007 crisis is represented between the blue dotted lines. The same pattern of instability that was seen in the previous result can be seen again. Time period before crisis is very stable and seem to increase with less variance compared with the period when the time series is decreasing.

As the previous case study, 15 random time series were also generated. This result is presented in the next figure. By analysing the figure we can see the similarities in the behaviour of the 15 presented trends. It may also be seen the crisis period in the time series where it is marked by a very strong negative variation.

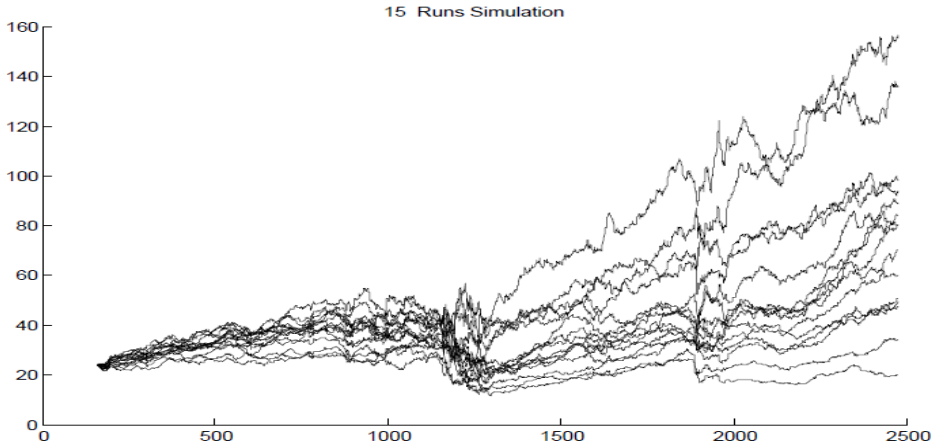

Figure 33 - 15 Runs Simulation Inverse-correlated Markets

6

Conclusions

The achieved results by this work were positive and show the possibility of using a SOM model to extract interesting clusters from a set of stock market correlated time series. In a general way, the states extracted by SOM were able to identify the crisis state and at the same time to project the stable time periods over the time series. This tool can be useful to an investor which pretends to measure the investment risk and to analyse the clusters that this tool produce.

In begin of the study most of the time was spent to create a data set. For this it was analysed some of the sources for data extraction but in the end the chosen system to study with was the Yahoo finance Api. With this decision it was possible to start projecting the first part of the system designed in this study that is the information extractor.

With this information it was possible to start creating the analyser. During this process many script functions were tested to check if all the correct procedures were taken in consideration. The first projections of the SOM algorithm started to appear and by the end of this process it was concluded the need of better clustering analysing. The classification that SOM's presented was based on a micro patterns perspective, which is complicated to understand the meaning of each pattern over the time series by human observation.

The need of a tool that groups the micro clusters based in the UMAT view appeared. Based on that, this study brings a new form of clustering over the UMAT. The result was a new technique that projects a good clustering not only for the case studies of this thesis, but also for some well-known clustering prob-

lems. This technique may still need some optimizations to improve its results, as will be mentioned in the future work.

The clusters obtained from the simulation with the generator method were very well structured and organized. The groups corresponded to similar behaviour and this result may be verified by the generated time series. The variation per cluster corresponded with the real time series, and the repetition of behaviour patterns shows that the model was well constructed.

This work contributed in many different ways. As a first contribution, we have the cluster technique, which already is referenced by an article that was presented in JOCLAD 2014 Lisbon. This technique brings a new way of clustering and also a new way of visualising data over the UMAT. The second is a new approach to financial data analysing. The correlated markets project good results that in my opinion may be subject of further study.

6.1 Future Work

This work generated some possibilities for further study. The clustering method has some points that may be explored. Better analyses of the financial time series with the SOM also may be done. Based on this, some points that can be researched in future works will be presented.

For the clustering methods we have:

- ➔ A local flood exploration: A local flooding exploration would be interesting to explore because the flood variable would auto-adapt to the distance environment for a certain zone. It has to be used some statistical analysis on the min local distance to decide which will be the flooding measure. After this calculus it could project a flooding based on new well established boundary.
- ➔ Another possible combination with the method could be the usage of the density by zone. This approach is already being investigated by some second stage SOM methods (examples are in the related work) but none of them are combined with the UMAT. So this may also be a good test to improve the algorithm.

For the states detected with SOM:

- ➔ The States detected weren't yet totally understood. The main objective of this research was the study if there were interested states generated, leaving most of the meaning of each apart. Based on this assumption, there is more work that can be done in the understanding of the clus-

ters, which may even be more interesting from an economic perspective for study.

- ➔ Another research that could be done is the pattern extraction based on the clusters observed. By human observation some patterns were interesting, like the repetition of certain states and the time presence in certain states. So these patterns could be more investigated.
- ➔ Finally but not the least it would be interesting to define a rule system that works with the result states of this application. With this it would be possible to implement an automatic trading system, which may be validated based on the return generated.

7

Bibliography

[1] T. Kohonen, J. Hynninen, J. Kangas and J. Laaksonen, "The Self-Organizing Map Program Package," 1995.

[2] E. F. Fama, "Market Efficiency, Long-Term Returns, and Behavioral Finance," *Journal of financial economics,* vol. 49, no. Finance, pp. 283-306, 1998.

[3] "Investopedia," 2013. [Online]. Available: http://www.investopedia.com.

[4] H. Markowitz, "Portfolio selection," *he journal of finance,* vol. 7.1, pp. 77-91, 1952.

[5] Madhavan, A. N. and J. Yang, "Practical risk analysis," *The Journal of Portfolio Management* , vol. 30.1, pp. 73-85, 2003.

[6] G. Corsetti, M. Pericoli and M. Sbracia, "CORRELATION ANALYSIS OF FINANCIAL CONTAGION: WHAT ONE SHOULD KNOW BEFORE RUNNING A TEST," *ECONOMIC GROWTH CENTER,* April 2001.

[7] S. Papadimitriou, J. Sun and P. S. Yu, "Local Correlation Tracking in Time Series," *Data Mining, 2006. ICDM '06. Sixth International Conference*

on, Dec 2006.

[8] C. Alexande, Stocks & Commodities V, 1998.

[9] N. C. Marques and C. Gomes, "Implementing an Intelligent Moving Average with a Neural Network," in *ECAI*, 2010.

[10] M. S. Graneroa, J. T. Segovia and J. G. Pérez, "Some comments on Hurst exponent and the long memory processes on capital markets," in *Physica A: Statistical Mechanics and its Applications*, 2008.

[11] H. E. Hurst, "Long-term storage of reservoirs: an experimental study," *Transactions of the American society,* 1951.

[12] A. M. D. G. C. M. N. C. Raposo, "A stock market crash alarm system based on a Hurst Index Neural Network," *Portuguese Conference on Artificial Intelligence,* 2011.

[13] V. L. M. P. Fernando Bação, "Self-organizing Maps as Substitutes for K-Means Clustering," in *Computational Science - ICCS 2005*, Springer, 2005.

[14] S. S. D. a. C. D. N. D T Pham, "Selection of K in K -means clustering," *Proceedings of the Institution of Mechanical Engineers Part C Journal of Mechanical Engineering Science ,* 2004.

[15] J. MacQueen, "Some methods for classification and analysis of multivariate observation.," *5th Berkeley Symposium on Mathematical Statistics and Probability,* 1967.

[16] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *The American Statistician 46.3,* 1992.

[17] S. Dhanabal and S. Chandramathi, "Review of various k-Nearest Neighbor Query Processing Techniques," *International Journal of Computer Applications 31 ,* 2011.

[18] S. R. G. H. R. S. J. Goldberger, "Neighbourhood Component Analysis," *Advances in Ne ural Information Processing Systems 17,* 2005.

[19] K. Q. Weinberger, B. J. C. and S. L. K., "Distance Metric Learning for

Large Margin Nearest Neighbor Classification," *Advances in Neural Information Processing Systems 18,* 2006.

[20] wikipedia, "Self-organizing map," 2013.

[21] S. M. Guthikonda, "Kohonen Self-Organizing Maps," December 2005.

[22] A. Ultsch and H. P. Siemon, "Kohonen's SeIf Organizing Feature Maps for Exploratory Data Analysis," *Proceedings of the International Neural Network Conference (INNC-90),* 1990.

[23] http://users.ics.aalto.fi/jhollmen/dippa/node24.html, "UMatrix".

[24] C. S-C and Y. C-C, "A Two-stage Clustering Method Combining Ant Colony SOM," *Journal of Information Science and Engineering,* vol. 24, 2008.

[25] J. A. Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Applied statistics,* pp. 100-108, 1979.

[26] L. Hamel and C. W. Brown, "Improved interpretability of the unified distance matrix with," in *7th International Conference on Data Mining*, 2011.

[27] D. Deng and N. Kasabov, "On-line pattern analysis by evolving self-organizing maps," *Neurocomputing,* vol. 51, pp. 87-103.

[28] G. Cabanes and Y. Bennani, "Learning the number of clusters in Self Organizing Map," in *Self-Organizing Maps*, Intech, 2010.

[29] A. Ultsch, "Clustering with SOM: U*C," in *Proc. Workshop on Self-Organizing Maps (WSOM 2005)*, Paris, 2005.

[30] P. a. T. A. P. Sarlin, "Mapping the state of financial stability," *Journal of International Financial Markets,* 2013.

[31] G. C. Panosso, "Análise do mercado financeiro baseada em análise," ISEGI, Lisboa, 2012.

[32] N. C. B. R. A. V. A. Chen, "Clustering and visualization of bankruptcy

trajectory using self-organizing map," *Expert Systems with Applications,* vol. 40, 2013.

[33] D. W. A. Tanenbaum, Computer Networks, 5th Edition, 2010.

[34] C. Bond, 2011. [Online]. Available: http://www.crbond.com/papers/fldfill_v2.pdf. [Accessed Jan 2014].

[35] G. B. P. B. a. E. L. M. Mufti, "Determining the number of groups from measures of cluster stability," *Proceedings of International Symposium on Applied Stochastic Models and Data Analysis,* 2005.

[36] Caliński and Harabasz, "A Dendrite Method for Cluster Analysis," in *Communications in Statistics - Theory and Methods*, 1974.

[37] W. J. Krzanowski and Y. T. Lai, "A criterion for determining the number of groups in a data set using sum-of-squares clustering," *Biometrics,* vol. 44.1, 1985.

[38] Davies, D. L. and D. W. Bouldin, "A cluster separation measure," *Pattern Analysis and Machine Intelligence, IEEE Transactions on 2,* 1979.

[39] Y. B. a. M. V. M. Halkidi, "Cluster validity methods," in *part I & II SIGMOD*, 2002.

[40] wikipedia, "Monte Carlo Method," 2013.

[41] riskamp, "thumbstacks," [Online]. Available: http://www.thumbstacks.com/files/RiskAMP%20-%20Monte%20Carlo%20Simulation.pdf.

[42] C. Joy, P. P. Boyle and K. S. Tan, "Quasi-Monte Carlo Methods in Numerical Finance," *Management Science,* June 1996.

[43] C. Lemieux and P. L'Ecuyer, "On the Use of Quasi-Monte Carlo Methods in Computational Finance," *Springer-Verlag,* 2001.

[44] J. H. J. A. E. &. P. J. Vesanto, "Self-organizing map in Matlab: the SOM Toolbox," in *Proceedings of the Matlab DSP conference*, 1999.

[45] yahoo, "yahoo," yahoo inc., 2013. [Online]. Available: https://code.google.com/p/yahoo-finance-managed/wiki/YahooFinanceAPIs. [Accessed 2013].

[46] I. E. L. Ioannis G. Tsoulos, "MinFinder: Locating all the local minima of a function," 2006.

[47] R. Fisher, "The use of multiple measurements in taxonomic problems," in *Annals of Eugenics*, 1936, pp. 179-188.

[48] S. Aeberhard, D. Coomans and O. de Vel, "Comparison of Classifiers in High Dimensional Settings," Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland, Townsville, 1992.

[49] D. Holtom and E. Fisher, Enjoy Writing Your Science Thesis or Dissertation!: A step by step guide to planning and writing dissertations and theses for undergraduate and graduate science students, London: Imperial College Press, 1999.