# Pattern Discovery from Stock Time Series Using Self-Organizing Maps †

Tak-chung Fu, Fu-lai Chung‡, Vincent Ng and Robert Luk

Department of Computing
Hong Kong Polytechnic University
Hunghom, Kowloon, Hong Kong.
{cstcfu, cskchung, cstyng, csrluk}@comp.polyu.edu.hk

## ABSTRACT

Pattern discovery from time series is of fundamental importance. Particularly when the domain expert derived patterns do not exist or are not complete, an algorithm to discover specific patterns or shapes automatically from the time series data is necessary. Such an algorithm is noteworthy in that it does not assume prior knowledge of the number of interesting structures, nor does it require an exhaustive explanation of the patterns being described. In this paper, a clustering approach is proposed for pattern discovery from time series. In view of its popularity and superior clustering performance, the self-organizing map (SOM) was adopted for pattern discovery in temporal data sequences. It is a special type of clustering algorithm that imposes a topological structure on the data. To prepare for the SOM algorithm, data sequences are segmented from the numerical time series using a continuous sliding window. Similar temporal patterns are then grouped together using SOM into clusters, which may subsequently be used to represent different structures of the data or temporal patterns. Attempts have been made to tackle the problem of representing patterns in a multi-resolution manner. With the increase in the number of data points in the patterns (the length of patterns), the time needed for the discovery process increases exponentially. To address this problem, we propose to compress the input patterns by a perceptually important point (PIP) identification algorithm. The idea is to replace the original data segment by its PIP's so that the dimensionality of the input pattern can be reduced. Encouraging results are observed and reported for the application of the proposed methods to the time series collected from the Hong Kong stock market.

## KEY WORDS

Time Series Analysis, Pattern Discovery, Self-Organizing Maps

## 1. Introduction

Recently, the increasing use of temporal data has initiated various research and development attempts in the field of data mining. Time series are an important class of temporal data objects and they can be easily obtained from scientific and financial applications, e.g., daily temperatures, production outputs, weekly sale totals, and prices of mutual funds and stocks. They are in fact major sources of temporal databases and undoubtedly finding useful time series patterns are of primordial importance. Unlike the transactional databases with discrete/symbolic items, time series data are characterized by their numerical, continuous nature. Hence, time series data are difficult to manipulate. But when they can be focused as segments instead of each data point, interesting patterns can be discovered and it becomes an easy task to query, understand and mine them. So, interesting or frequently appearing pattern discovery is one of the important tasks in time series analysis.

In the field of pattern discovery in continuous data, there exists quite a number of techniques, ranging from the classical works, see, e.g. [1,2], to the more recent idea of localized radial basis functions [3]. In general, these approaches seek a nonparametric description of the data's structure by a weighted summation of locally tuned basis functions. Radial basis functions have been shown to possess the universal approximation ability [4,5]. Theoretically, given unlimited numbers of hidden units and infinite samples, any pattern governed by a functional relationship can be detected and learned by such networks. However, in the absence of large amount of samples, nonparametric methods cannot simultaneously uncover both local and global structure in the data [6]. That is also why the radial basis function networks suffer from the dimensionality problem when the intrinsic dimensionality of the data is less than the apparent one [7].

Finding periodic pattern is concerned by pattern discovery in temporal data. In [8], two types of problems are presented. One is to find periodic patterns of a time series with a given period, while the other is to find a periodic pattern with arbitrary length. On the other hand, exploratory data analysis embodies a diverse collection of methods. Graphical

† *This work was supported by the RGC CERG project PolyU 5065/98E and the Departmental Grant H-ZJ84*

‡ *Corresponding author*

methods attempt to pictorially illuminate the inherent structure in the data by the use of multiple scatter plots, glyphs, color, and parallel axes plots [6]. These visual discovery techniques rely entirely on human's interpretations, which are undoubtedly powerful, but not immune to fatigue, error, and information overload. Projection methods such as principal component analysis (PCA) and projection pursuit (PP) are effective discovery tools when the relationships in the data are linear. When the structure lies on a nonlinear manifold, both methods have difficulty in detecting the structure of data [9,10]. In [11], the application of principal component analysis and independent component analysis (ICA) for blind source separation of univariate financial time series has been proposed. Its main objective is to find out if these techniques are able to perform feature extraction, signal-noise-decomposition and dimensionality reduction because that would enable a further inside look into the behavior and mechanics of the financial markets.

In [12], methods for the automatic derivation of qualitative descriptions of complex objects are presented. The ultimate goals of this investigation are the development of a methodology for the qualitative representation of complex objects, and the discovery of knowledge based on the study of collections of such qualitative descriptions. In [13], the reconstructed state space allows temporal pattern extraction and local model development. Using the a priori data mining objective, an optimal local model is chosen for short-term forecasting. For the same sampling period, multiple time series embedding produces better temporal patterns than single time series embedding. As another way of pattern discovery [14], the residual analysis and recursive partitioning were proposed to identify statistically significant events in a data set. Moreover, a probabilistic description of the data is automatically furnished by the discovery process.

For the problem of pattern discovery in time series, one of the largest groups of technique being employed is clustering. Cluster analysis is applied to the sliding window of the time series and it is a useful method for grouping related temporal patterns that are dispersed along the time series. Clustering methods seek out a special type of structure, namely, grouping tendencies in the data. In this regard, they are not as general as the other approaches but can provide valuable information when local aggregation of the data is suspected. In [15], the pattern templates for matching are not predefined. Instead, the templates are generated automatically by clustering technique and they will then be used for further matching in the discretization process to produce meaning symbols. References [16,17] describe adaptive methods for finding rules of the above type from time series data. The methods are based on discretizing the sequence by methods resembling vector quantization. They first form subsequences by sliding a window through the time series, and then cluster these subsequences by using a suitable measure of time series similarity.

In this paper, a neural clustering based pattern discovery scheme that cooperates with a new pattern matching scheme is proposed. Its application to time series of stock price is demonstrated. It adopts the self-organizing map (SOM) [18] for pattern discovery in temporal patterns. SOM is a special type of clustering algorithm [19] with attractive discovery power. Although popular for its visual interpretability, the SOM imposes a topological structure on the data. Frequently appearing and potentially interesting patterns of different length can be discovered from time series in an efficient manner. The paper is organized into seven sections. The statement of problem is presented in section 2. Applying self-organizing map to pattern discovery in time series is introduced in section 3. Section 4 briefly describes a perceptually important point based pattern matching scheme. In section 5, the integration of the pattern discovery approach with the pattern matching scheme is described. The simulation results are reported in section 6 and the final section concludes the paper.

## 2. Statement of the Problem

Patterns occur in all types of real-life data. The analysis of economic, biomedical, demographic, diagnostic, financial data usually reveals some type of organization that is different from uniform randomness. Many authors refer to this organization as the structure of the data, or equivalently, the patterns in the data. We will use these intuitive terms interchangeably. With an understanding of the data's structure we can perform useful operations in the space of interest. These operations include the prediction of variable values, classification of previously unobserved samples and assessing the likelihood of a specific event. Therefore, the discovery of patterns is an indispensable step in the understanding of a given time series.

A more important usage of pattern discovery from time series is numeric-to-symbolic (N/S) conversion [20]. Unlike the transactional databases with discrete/symbolic items, time series data are characterized by their numerical, continuous nature. It is suggested in [21] to break down the sequences into meaningful subsequences and represent them using real-valued functions. Furthermore, there is a need to discretize the continuous time series into meaningful labels/symbols [15, 22]. We term this process as numeric-to-symbolic (N/S) conversion and consider it as one of the most important components in time series data mining systems. Most of the pattern matching researches take prior domain information into account in a principled manner. However, we do not want to define beforehand which patterns are to be used; rather, we want the patterns to be formed from the time series data itself. So, an algorithm to discover specific patterns or shapes automatically from the time series data is necessary. The algorithm is noteworthy in that it does not assume prior knowledge of the number of interesting structures, nor does it requires an exhaustive explanation of the patterns being described.

Without given user-defined pattern templates, a time series can still be converted into a symbolic representation by first forming subsequences (using sliding window) and then clustering these subsequences using suitable measure of pattern similarity. The symbolic version of the time series is obtained by using the cluster identifiers corresponding to the subsequence, in a manner which is similar to the data compression technique of vector quantization.

Applying cluster analysis to sliding windows of time series is useful for grouping related temporal patterns that are dispersed along the time series. Since the input patterns are time series, a similar series of data points that lead to a similar result would be clustered together. The switches from one structure of data to another, which are usually vague and not focused on any particular time point, are naturally treated by means of clustering.

The general scheme of the proposed pattern discovery algorithm for time series using unsupervised clustering includes rearrangement of time series into a set of continuously sliding windows for the clustering procedure and, optionally, applying feature extraction for each window. Then, similar temporal patterns are grouped together into clusters, which may represent the different structure of data or temporal patterns, by an unsupervised clustering procedure while self-organizing map is chosen as the clustering tool in this paper.

Consequently, two problems must be considered, the first one is the efficiency of the discovery process. With the increase of the number of data points in the patterns (the length of patterns), the time needed for the discovery process increases exponentially. So, a mechanism to reduce the execution time of the pattern discovery process is necessary. The second problem is the multi-resolution problem. Interesting patterns may appear in different length of subsequences instead of a fixed length. However, a unified set of pattern templates is always preferred for representing frequently appearing patterns from different length. These two problems are closely related to the pattern matching scheme involved in the discovery process.

## 3. Applying Self-organizing Maps for Pattern Discovery in Time Series

Clustering is a common method for finding structure in given data [19,23], in particular for finding structure related to time. In Fig.1, the general clustering procedure is listed. In each iteration, the winner cluster is found and its center is updated accordingly. The initial cluster centers can be chosen in various ways, e.g., chosen arbitrarily or by some sequences. Also, the number of clusters is a critical parameter to be determined. It can be fixed beforehand or can vary during the clustering process. The clustering procedure finally terminates when the number of iterations exceeds the maximum allowed number of iterations.

```
Begin
  Define number of cluster centers
  Set the initial cluster centers
  Repeat
    For each input data
      For each cluster, compute:
        Distance between the input data and
        its center
      Choose the closest cluster center as
      the winner
      Update the cluster centers
    Until (Convergence or maximum number of
     iterations is reached)
End
```

**Figure 1 Typical Clustering Process**

There are many popular clustering techniques developed, such as the hierarchical, nearest-neighbor, and *c*-mean algorithms. The most common one perhaps is the *c*-mean algorithm. However, it requires predefining the number of classes (clusters) which is not easy to achieve in many applications. The self-organizing map [18,24], which is a well-known neural network based unsupervised clustering algorithm, on the other hand offers a partial solution to this problem by introducing redundancy. Only a rough estimation of the number of nodes in the output layer is required. As such, we have adopted it for finding frequently appearing pattern in the time series.

Learning in SOM is based on competitive learning where the output nodes of the network compete among themselves to be activated or fired. Only one output node, or one node per group, is on at any one time. The output nodes that win the competition are called winner-take-all nodes. There are usually $m$ cluster nodes, arranged in a one- or two-dimensional array as shown in Fig.2. The input patterns are $n$-tuples. During the self-organization process, the cluster unit whose weight matches the input data most closely (typically, in terms of the minimum squared Euclidean distance) is chosen as the winner. The weights of the winning node and its neighbors will then be updated proportionally. The weight vectors of neighboring unit are not necessarily close to the input pattern. The overall clustering process is shown in Fig.3. The learning rate *eta* is a slowly decreasing function of time (or training epochs). Typically, a linearly decreasing function is satisfactory for practical computations. The radius $R$ of the neighborhood around a cluster unit also decreases as the clustering process progresses.

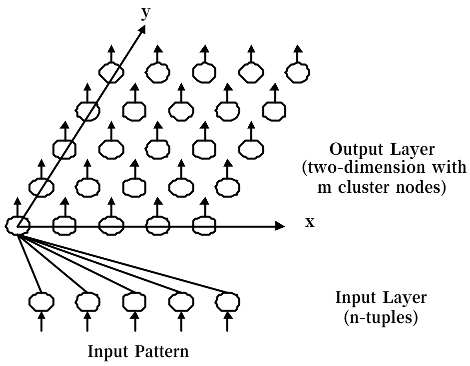



**Figure 2 The architecture of self-organizing map**

The formation of a (clustering) map occurs in two phases: the initial formation of the correct order and the final convergence. The second phase takes much longer than the first and requires a small value for the learning rate. Many iterations through the training set may be necessary. Random values may be assigned for the initial weights. If some information is available concerning the distribution of clusters that might be appropriate for a particular problem, the initial weights can be taken to reflect that prior knowledge. The inference part of the algorithm simply calculates each output nodes' value and finds the winner to determine which cluster the input pattern has fallen into.

In order to extract frequently appearing patterns from the original time series for further mining operations, the self-organizing map is applied to the set of continuously sliding windows of a time series for grouping similar temporal patterns which are dispersed along the time series. Given a time series $s = (x_1,..., x_n)$ and fixing the sliding window's width at $w$, a set of time series segments $W(s) = \{s_i = (s_i,..., s_{i+w-1}) \mid i = 1,..., n - w + 1\}$ can be formed. With respect to the availability of multiple time series, a collection of $W(s)$'s will be generated and they will be the input patterns to the SOM for training. The trained network is expected to group a set of patterns $p_1,..., p_T$ where $T$ is the size of output layer plane. During the training process, the data or temporal patterns represented by each node of the output layer alter in each iteration according to the input pattern. Finally, a pattern structure that can represent most of the winning candidates and is distinct from its neighbors is formed. The set of patterns obtained from the output layer then represent the most frequently appearing patterns according to the given time series. Fig.4 exemplifies the formation of frequently appearing patterns from two output nodes. Such an evolutionary illustration is captured from different iteration of the whole training process.

```
Begin
    Set neighborhood parameters
    Set learning rate parameters
    Initialize weights
    Repeat
        For each input pattern
            For each node, compute the distance:
```

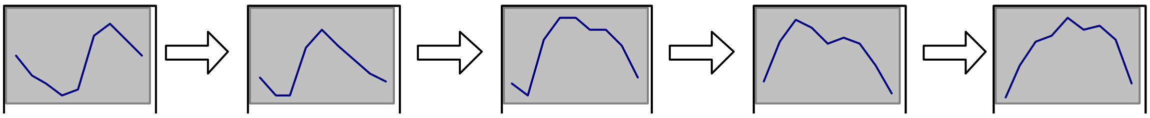
$$D(j) = \sum_i (w_{ij} - x_i)^2$$

```
            Find index j such that D(j) is a minimum
            For all units j, and for all i:
                Factor=exp(-(distance_with_winner/R));
```

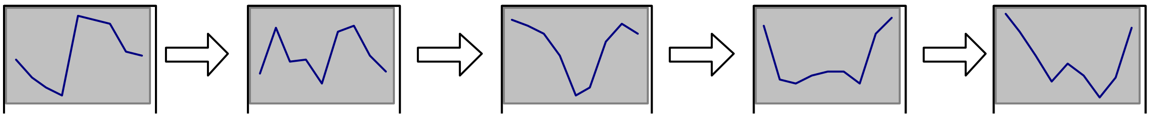
$$w_{ij}(new) = w_{ij}(old) + eta \times Factor \times [x_i - w_{ij}(old)]$$

```
            Reduce learning rate eta and radius R of neighborhood
            by specified ratio eta_rate and R_rate respectively
            If eta < eta_min
                eta=eta_min
            If R < R_min
                R=R_min
      Until (Convergence or maximum no. of iterations is exceeded)
End
```

**Figure 3 Training process of self-organizing map**



a) sample 1 of the iterative process of a discovery pattern in a node of the output plan of SOM



b) sample 2 of the iterative process of a discovery pattern in a node of the output plan of SOM

**Figure 4 Two pattern examples during pattern discovery process**

Begin

Set parameter λ which is the maximum similarity for the grouping of discovered patterns

Repeat

Calculate all the distance between every pattern:

$$Dist(i,j)=\sqrt{\frac{\sum_{k=1}^{w}(a_{i,k}-a_{j,k})^2}{w}} \quad (\text{for } i = i \text{ to } T \text{ and } j = 1 \text{ to } T \text{ where } i \neq j)$$

Find *min*(*Dist*(*i*, *j*))

If *min*(*Dist*(*i*, *j*)) $\leq l$ ,then group pattern templates i and j into one and represent the new template by:

For *k = 1 to w*, compute

$$a_{new,k} = (a_{i,k} \times \frac{occur_i}{occur_i + occur_j}) + (a_{j,k} \times \frac{occur_j}{occur_i + occur_j})$$

where *occur* is the no. of testing patterns refer to such pattern template in the recall stage

Number of pattern templates $T = T - 1$

Until ( $min(\,Dist(i,j)) > l$ or *T* = *1*)

End

**Figure 5 Redundancy removal for time series pattern discovery**

To fine-tune the proposed method, two issues were considered. First is about the effectiveness of the patterns discovered. We propose to filter out those nodes (patterns) in the output layer that did not participle in the recall process. In addition, redundant or similar pattern templates may be found especially in low input dimension (i.e., short sliding window width). This will cause undesirable effects in the data mining process afterwards because redundant or similar patterns are referring to different patterns, which will affect the support of actually interesting patterns. Thus, a redundancy removal process is introduced to group similar patterns together and represent them with a relatively more general pattern. Such a step is depicted in Fig.5. It will reduce the number of pattern templates being formed and will generate relatively more representative patterns. The distances between them are guaranteed to be larger than λ. With such a process, both the number and the structure of discovered patterns are optimized to a reasonable state.

## 4. A Flexible Pattern Matching Scheme Based on Locating Perceptually Important Points

A pattern matching scheme is then essential for similarity measure in the clustering process. In this section, a new time series pattern matching scheme is introduced. Interesting and frequently appearing patterns are typically characterized by a few critical points. For example, the head and shoulder pattern should consist of a head point, two shoulder points and a pair of neck points. These points are perceptually important in the human identification process and should also be taken into accounts in the pattern matching process. The proposed scheme follows this idea by locating those perceptually important points in the data sequence *P* in accordance with the query sequence *Q* [25]. The location process works as follows.

With sequences $P$ and $Q$ being normalized, the perceptually important points (PIP's) are located in order according to Fig.6. Currently, the first two PIP's will be the first and last points of $P$. The next PIP will be the point in $P$ with maximum distance to the first two PIP's. The fourth PIP will then be the point in $P$ with maximum distance to its two adjacent PIP's, either in between the first and second PIP's or in between the second and the last PIP's. The PIP location process continues until number of points located is equal to the length of query sequence $Q$. To determine the maximum distance to the two adjacent PIP's, the vertical distance between the test point $p_3$ and the line connecting the two adjacent PIP's is calculated (Fig.7), i.e.,

$$VD(p_3, p_c) = |y_c - y_3| = \left| \left( y_1 + (y_2 - y_1) \cdot \frac{x_c - x_1}{x_2 - x_1} \right) - y_3 \right|$$

where $x_c = x_3$. It is intended to capture the fluctuation of the sequence and the highly fluctuated points would be considered as PIP's.

```
Function Pointlocate (P,Q)
  Input: sequence p[1..m], length of
          Q[1..n]
  Output: pattern sp[1..n]
  Begin
    Set sp[1]=p[1], sp[n]=p[m]
    Repeat until sp[1..n] all filled
    Begin
      Select point p[j] with maximum
       distance to the adjacent points
       in sp (sp[1] and sp[n] initially)
      Add p[j] TO sp
    End
    Return x
End
```

**Figure 6 Pseudo code of the PIP identification process**

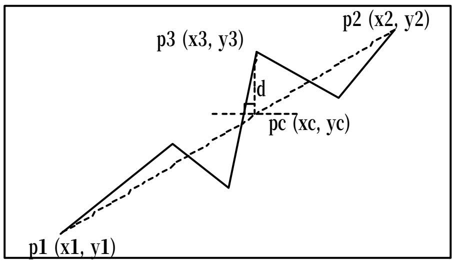



**Figure 7 Distance metric by calculate the vertical distance**

To illustrate the identification process, the 'head and shoulder' pattern is used and Fig.8 shows the step-by-step result using the Euclidean distance metric. Here, the number of data points in the input sequence $P$ and query sequence $Q$ are 29 and 7 respectively, i.e., $m$=29 and $n$=7.

After identifying the PIP's in the data sequence, the similarity between $P$ and $Q$ can be computed and direct point-to-point comparison can be applied, i.e., to compute

$$Dist(SP, Q) = \frac{1}{n} \sum_{k=1}^{n} (sp_k - q_k)^2$$

for all query pattern $Q$ in the predefined set. Here, $SP$ and $(sp_k)$ denote the PIP's found in $P$. However, the measure in eq.(2) has not yet taken the horizontal scale (time dimension) into considerations. To do so, the distance (or similarity) measure could be modified as:

$$Dist'(SP, Q) = \frac{1}{n} \sum_{k=1}^{n} (sp_k - q_k)^2 + \frac{1}{n-1} \sum_{k=1}^{n-1} (sp_k^t - q_k^t)^2$$

where $sp_k^t$ and $q_k^t$ denote the time coordinate of the sequence points $sp_k$ and $q_k$ respectively.

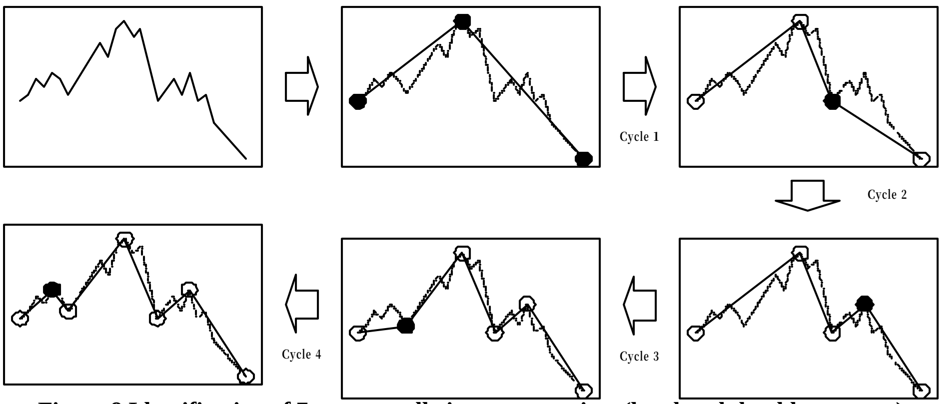



**Figure 8 Identification of 7 perceptually important points (head-and-shoulder pattern)**

## 5. Efficient Pattern Discovery Based on Perceptually Important Point Compression

Although the clustering procedure can be applied directly to the sliding window of the sampled raw data, it will become quite time consuming when a large number of data points (high dimension) are considered. To address this problem, we propose to compress the input patterns by the perceptually important point identification algorithm described in previous section. The idea is to replace the original data segment by its PIP's so that the dimensionality of the input pattern can be reduced. In Fig.9, the original patterns and the compressed ones are compared. One may easily find that the compressed ones are perceptually very similar to their original patterns. By applying this process, the speed for pattern discovery in time series data can be improved significantly without affecting too much of the structure of the patterns discovered.

As we know, interesting patterns may not always appear in a fixed width of subsequences. Hence, the ability to discover patterns in different resolutions becomes important but yet it is still a nontrivial task. The proposed method can be easily adapted to process multi-resolution patterns. This can be done by adjusting the sliding window's length *w*, through which different scales of the frequent appearing patterns can be discovered. Thus, the most direct way to discover multi-resolution patterns is to conduct the clustering process a few times for different resolutions (using different *w*). The results are then integrated to prepare a final set of pattern templates. Although such an approach may be effective for multi-resolution pattern discovery, it is not intuitive in a sense that it needs to apply the clustering process to each resolution and duplicated/redundant patterns may exist. A more systematic approach perhaps is to simplify the above process by first grouping the training patterns from different resolutions into a set of training samples to which the SOM clustering process is applied for only one time. We call this approach as combined multi-resolution pattern discovery as shown in Fig.10 and it is adopted in the proposed pattern discovery algorithm. With the setup described above, pattern templates in different resolutions (different length) can be easily obtained for the subsequent data mining process.

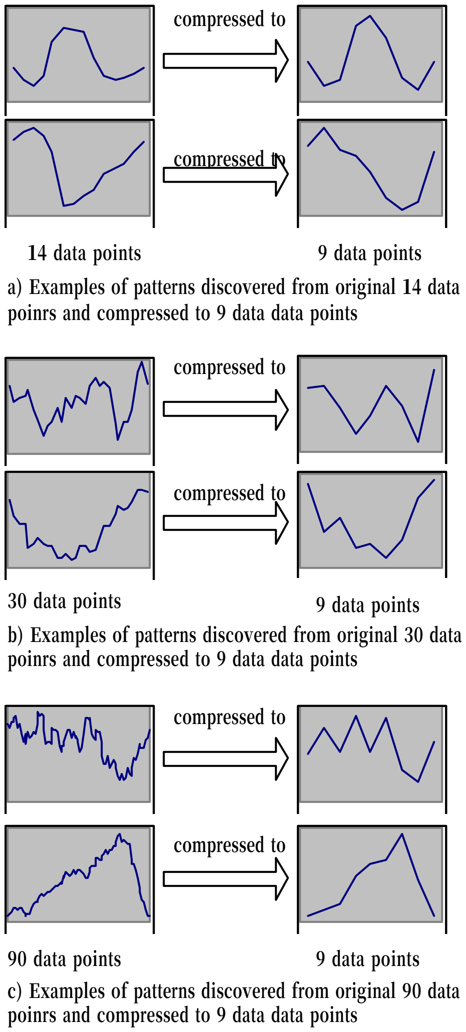



**Figure 9 Examples of original patterns and compressed patterns (by PIP's) discovered from the clustering process**

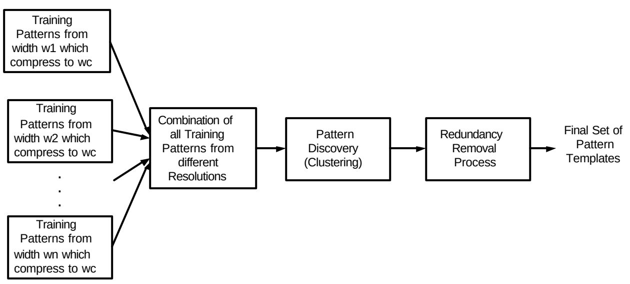



**Figure 10 Combined multi-resolution pattern discovery process**

## 6. Simulation Results

In this section, the results of two sets of experiments are reported. First, the effectiveness of the proposed pattern discovery algorithm using different λ (maximum grouping similarity) values is presented. Second, the efficiency of applying the PIP-based compression step to the input patterns for clustering is demonstrated, with sample discovered patterns reported and comp ared. All the parameters of the SOM clustering process except the one being varied were set to the default values shown in Table 1.

**Table 1 Parameters for the SOM Clustering Process**

| Parameter | | Value |
|---|---|---|
| *Ncycle* | Number of Training Iterations | 1,000 |
| *Ntrain* | Number of Training Patterns | 2542 |
| *eta* | Current Learning Rate | 0.5 |
| *eta_rate* | Rate of decreasing in learning rate | 0.975 |
| *eta_min* | Minimum Learning Rate | 0.1 |
| *R* | Current Radius of Neighborhood | 5.0 |
| *R_rate* | Rate of decreasing in radius | 0.95 |
| *R_min* | Minimum Radius | 0.01 |

In the first experiment, we have tested the proposed pattern discovery algorithm on the daily open price of Hong Kong Hang Seng Index (HSI) from 3/89 to 3/99. In order to test on the effectiveness of the redundancy removal step and to determine the necessary resource for such a task, we have simulated different SOM sizes. In Table 2, the number of patterns discovered is recorded.

**Table 2 Number of patterns discovered using different number of output nodes and different l**

| No. of Output Nodes | Execution Time | l | | | | |
|---|---|---|---|---|---|---|
| | | **0.15** | **0.175** | **0.2** | **0.225** | **0.25** |
| 10×10 | 2:50:11 | 98 | 91 | 73 | 52 | 27 |
| 8×8 | 1:36:24 | 64 | 59 | 46 | 35 | 25 |
| 6×6 | 1:08:42 | 36 | 36 | 29 | 23 | 20 |
| 5×5 | 0:57:20 | 25 | 25 | 24 | 18 | 13 |

We noticed that the pattern templates obtained by 8×8 output nodes are still reasonable with redundancy removal using λ=0.225. The merging of patterns from the output layer is acceptable, in terms of our subjective evaluation, such as those shown in Fig.11. However, when we further increase λ to 0.25, unreasonable merging as shown in Fig.12 is observed. In terms of the number of pattern templates remained, i.e., 35, it matches with the 6×6 output layer where the resulted patterns are reasonable without duplication. Hence, the 6×6 output layer size is generally suggested, with respect to both efficiency and effectiveness concerns, for our stock time series applications.

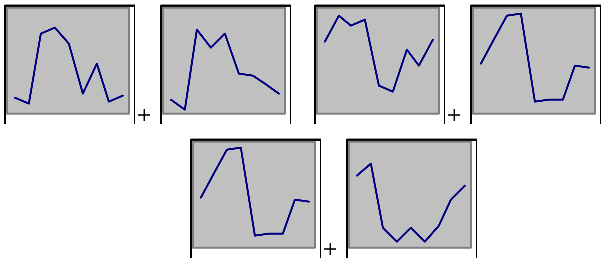


**Figure 11 Reasonable pattern merging using l=0.225 during the redundancy removal process**

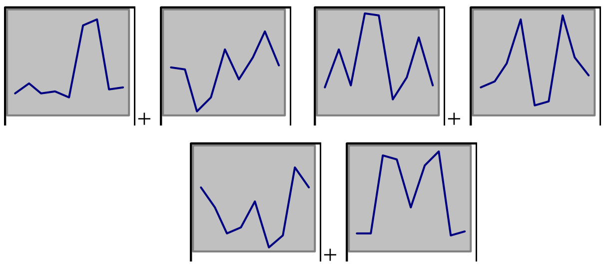


**Figure 12 Unreasonable pattern merging using l =0.25 during the redundancy removal process**

In the second experiment, we have tested our algorithms on a few Hong Kong Stock Exchange listed stocks' time series. The time series are the daily close price collected from 1/1996 to 10/2000. By using the proposed PIP identification technique to compress the input data sequences, we can reduce the dimension of the clustering process. The execution time of the pattern discovery task using different sliding window width is plotted in Fig.13 where one curve corresponds to process the data in original dimensions without compression and the other one depicts the results of compressing to the minimum dimension (i.e., 9 data points). As expected, the execution time when compression is used is invariant to the window width being used while the execution time for the original pattern discovery process (without compression) increases exponentially with the dimension of the input data sequences. Furthermore, by applying the PIP-based compression step, Fig.14(b)&(d) show that the set of patterns discovered is quite similar in shape with the uncompressed counterpart, see particularly Fig.14(a) vs. Fig.14(b).

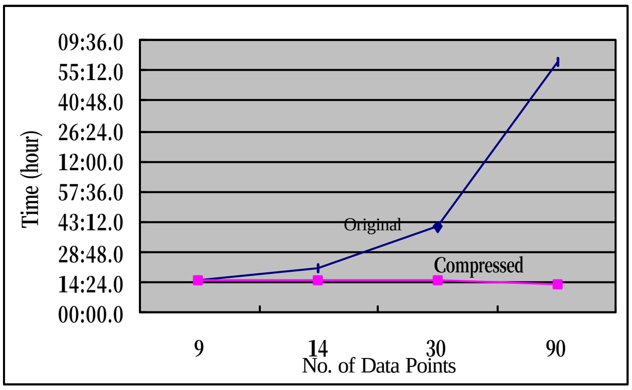



**Figure 13 Execution time used by the pattern discovery process**

## 7. Conclusions

In this paper, we have described a method for automatic pattern discovery in time series data. It is based on the self-organizing maps, which group similar temporal patterns dispersed along the time series. The proposed method is further improved by a redundancy removal step to consolidate the patterns discovered and the introduction of the perceptually important point identification method to reduce the dimension of the input data sequences for the SOM clustering process. Enhancement for handling multi-resolution patterns has also been discussed. Encouraging simulation results on the Hong Kong Hang Seng Index series and a few stock time series are reported. Future work includes applying the discovered patterns for various time series analysis domains such as:

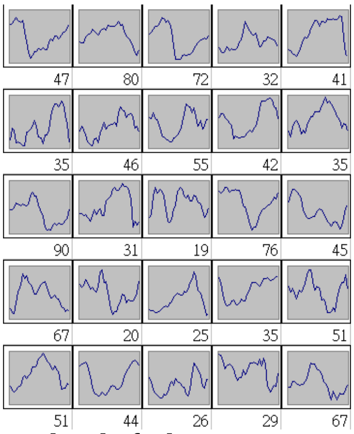
- prediction
- serves as the pattern templates for numeric-to-symbolic (N/S) conversion
- summarization of the time series

## References

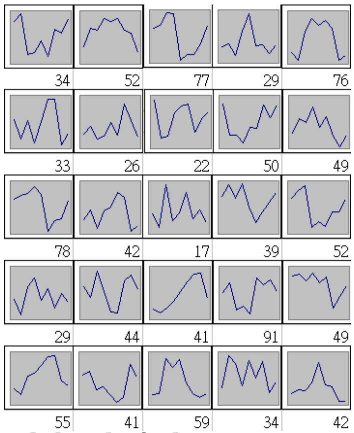
1. M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *Annals of Math. Statistics*, Vol. 27, pp.435-525, 1985.
2. E. Parzen, "On estimation of probability density function and mode," *Annals of Math. Statistics*, Vol. 33, pp.1065-1076, 1962.
3. J. Moddy and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computation*, Vol. 1, no. 2, pp.281-294, 1989.
4. E. Hartman, J. Keeler and J. Kowalski, "Layered neural networks with Gaussian hidden units as universal approximations", *Neural Computation*, Vol. 2, No. 2, pp.210-215, 1990.
5. J. Park and I. W. Sandberg, "Approximation and radial basis function networks," *Neural Computation*, Vol. 5, No. 2, pp. 305-316, 1993.
6. D. W. Scott, *Multivariate Density Estimation*. John Wiley and Sons, 1992.
7. C. Bishop, *Neural Networks for Pattern Recognition*. Oxford Univ. Press, 1995.
8. W. Gong, *Periodic Pattern Search on Time-Related Data Sets*. M.Sc. thesis, Computing Science, Simon Fraser University, December 1997.
9. M. Dolson, "Discriminative nonlinear dimensionality reduction for improved classification," *Int. J. Neural Systems*, Vol. 5, No. 4, pp. 313-333, 1994.
10. P. J. Huber, "Projection pursuit (with discussion)," *Annals of Statistics*, Vol. 13, pp. 435-525, 1985.
11. R. H. Lesch, Y. Caille and D. Lowe, "Component analysis in financial time series," *Proc. of the IEEE/IAFE 1999* Conference on *Computational Intelligence for Financial Engineering (CIFEr)*, , pp. 183-190, 1999.
12. E. H. Ruspini and I. S. Zwir, "Automated qualitative description of measurements," *Proc. of the 16th IEEE Instrumentation and Measurement Technology Conference IMTC/99*, Vol. 2, pp. 1086–1091, 1999.
13. R. J. Povinelli and X. Feng, "Data Mining of Multiple Nonstationary Time Series," *Proc. Artificial Neural Networks in Engineering*, pp. 511-516, 1999.
14. T. Chau and A. K. C. Wong, "Pattern discovery by residual analysis and recursive partitioning," *IEEE Transactions* on *Knowledge and Data Engineering*, Vol.11, No.6, pp. 833–852, Nov.-Dec. 1999.
15. G. Das, K. I. Lin and H. Mannila, "Rule discovery from time series", *Proc. KDD-98*, pp. 16-22, 1998.
16. S. Policker and A. B. Geva, "Non-stationary signal analysis using temporal clustering," *Proc. of the 1998 IEEE Signal Processing Society Workshop on Neural Networks for Signal Processing VIII*, pp. 304 –312, 1998.
17. A. B. Geva, "Non-stationary time series prediction using fuzzy clustering," *Fuzzy Information Processing Society, NAFIPS. 18$^{th}$ International Conference of the North American*, pp. 413-417, 1999.
18. T. Kohonen, *Self-Oranizing Maps*, Springer, Berlin, 1995.
19. B. D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge Univ. Press, 1996.
20. T.C. Fu, F.L. Chung, V. Ng and R. Luk, "Evolutionary segmentation of financial time series into subsequences," *Proc. 2001 Congress on Evolutionary Computation (CEC2001)*, pp. 426-430, Seoul, Korea, 27-30 May, 2001.
21. H. Shatkay and S. B. Zdonik, "Approximate queries and representations for large data sequences," *Proc. of the Int. Conf. on Data Engineering* (*ICDE96*), 1996.
22. O. Kai, W. Jia, P. Zhou and X. Meng, "A new approach to transforming time series into symbolic sequences," *Proc. of the First Joint BMES/ EMBS Conference*, October 1999.
23. A. K. Jain and R. C. Dubes, *Algorithms for Clustering*

*Data*, Prentice Hall, 1988.

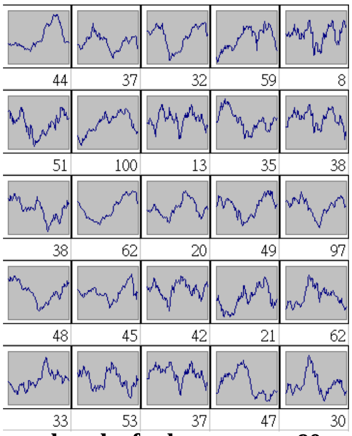
24. T. Kohonen, *Self-organization and Association Memory (3rd ed.)*, Berlin: Springer-Verlag, 1989.
25. F.L. Chung, T.C. Fu, R. Luk and V. Ng, "Flexible Time Series Pattern Matching Based on Perceptually Important Points," accepted by *International Joint Conference on Artificial Intelligence (IJCAI'01) Workshop on Learning from Temporal and Spatial Data*, Seattle, Washington, 4-10 August, 2001.



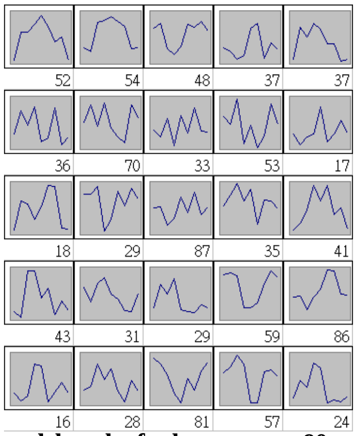

**a. length of subsequence = 30**




**b. length of subsequence = 30 (compressed to 9)**




**c. length of subsequence = 90**




**d. length of subsequence = 90 (compressed to 9)**

**Fig.14 Pattern templates discovered from different resolutions**