PATTERN MODELLING IN TIME-SERIES FORECASTING

Sameer Singh

{s.singh@exeter.ac.uk}

University of Exeter
Department of Computer Science
Prince of Wales Road
Exeter EX4 4PT

S. Singh. Pattern Modelling in Time-Series Forecasting, Cybernetics and Systems-An International Journal, vol. 31, issue 1, 2000.

PATTERN MODELLING IN TIME-SERIES FORECASTING

ABSTRACT

Pattern modelling in time-series prediction refers to the process of identifying past relationships and trends in historical data for predicting future values. This paper describes the development of a new pattern matching technique for univariate time-series forecasting. The pattern modelling technique out-performs frequently used statistical methods such as Exponential Smoothing on different error measures and predicting the direction of change in time-series. The paper discusses the prediction results on popular benchmarks and the real US S&P index for financial markets.

PATTERN MODELLING IN TIME-SERIES FORECASTING

INTRODUCTION

Univariate time-series prediction is important in several scientific domains. A forecasting procedure may be needed for predicting a single time-dependent variable or predicting several independent variables individually to forecast a dependent variable as in multivariate analysis. Univariate series may be predicted using a range of available tools. Delurgio(1998) classifies forecasting procedures as of type quantitative or qualitative. In this paper we are primarily interested in quantitative methods, especially procedures with computational intelligence. In particular, we will develop a new methodology for encoding time-series and pattern modelling for generating accurate predictions on three different benchmarks from the Santa Fe competition (Weigend and Gershenfeld, 1994). The technique will be finally evaluated on real data from the US S&P index.

A large number of studies in the past have worked with intelligent techniques for forecasting (Kingdon, 1997): for example neural networks (Azoff, 1994; Aerrabotu et al., 1997; Mahfoud and Mani, 1997), genetic algorithms (Teran et al., 1997) statistical techniques optimised using neural networks and genetic algorithms (Bonnet et al., 1997; Rolf et al., 1997), and fuzzy techniques (Chorafas, 1994; Daijin, 1997; Motnikar et al., 1996; Muhammad and King, 1997; Pellizzari and Pizzi, 1997; Studer and Masulli, 1996). A range of dynamic econometric techniques have also been investigated for forecasting (see Hendry, 1995). In this paper an alternative method based on pattern modelling is discussed. Pattern modelling refers to the process of describing the time-series as a series of patterns. These patterns may be specified in terms of the gradient of the time-series at a given time, i.e. an upward of downward movement, and its size (number of segments in them). Figure 1 shows

the different types of patterns found in time-series. The pattern shape depends on the total number of segments in that pattern. A segment may be defined as the transition from state t to t+1 in a time series, e.g. y(t) - y(t-1) is a segment, and so is y(t-1) - y(t-2). We may formalise this mathematically. Consider the time series as a vector $\mathbf{y} = \{y_1, y_2, \ldots y_n\}$ where n is the total number of points in the series. Often, we also represent such a series as a function of time, e.g. $y_n = y_t$, $y_{n-1} = y_{t-1}$. A segment in the series may be defined as a difference vector $\boldsymbol{\delta} = (\delta_1, \delta_2, \ldots \delta_{n-1})$ where $\delta_i = y_{i+1} - y_i$, $\forall\ 1 \leq i \leq n-1$. A pattern contains one or more segments and may be visualised as a string of segments $\rho = (\delta_i, \delta_{i+1}, \ldots \delta_h)$ for given values of i and h, $1 \leq i,h \leq n-1$, provided that $h > i$. In order to define the pattern mathematically, we may choose to encode the time series y as a vector of change in direction. For this, a point y_i is encoded as 0 if $y_{i+1} < y_i$, as a 1 if $y_{i+1} > y_i$ and a 2 if $y_{i+1} = y_i$. Formally, a pattern in the time-series may be represented as $\rho = (b_i, b_{i+1}, \ldots b_h)$ where b is a binary value in most cases, either a 1 or a 0, expect in cases when the series doesn't change which is encoded as a 2. For example, a pattern $\rho = (0, 1)$ represents a valley shape as defined in Figure 1.

Figure 1 here

The complete time-series may be encoded as $(b_1, \ldots b_{n-1})$. For a total of k segments in a pattern, it is encoded with k *b* values. For a pattern of size k, the total number of pattern shapes possible is $2^k + 1$. In Figure 1, only a brief description of patterns of size two and three is shown. More complicated structural primitives are found with larger sizes. The technique of matching structural primitives is based on the premise that the past repeats itself. The basic procedure involving the use of structural patterns for forecasting is discussed next.

FUZZY PATTERN MATCHING

Pattern matching in the context of time-series forecasting refers to the process of matching current state of the time series with its past states. Consider the encoded time series (b_1, b_i, ... b_{n-1}). Suppose that we are at time *n* (y_n) trying to predict y_{n+1}. A pattern of size k is first formulated from the last k values in the series, $\rho' = (b_{n-k}, \ldots b_{n-1})$. Smaller sized patterns may refer to easily identifiable shapes as in Fig. 1 whereas larger patterns may have more complex shapes. The size of the pattern used for matching has important impact on minimising the error and correctly predicting the direction of series change. The pattern size and matching procedure itself must be optimised for obtaining the best results. The aim of a pattern matching algorithm is to find the closest match of ρ' in the historical data (estimation period) and use this for predicting y_{n+1}. The match itself is never exact and may be loosely termed as 'fuzzy' in nature. The magnitude and direction of prediction depend on the match found. The success in correctly predicting series depends directly on the pattern matching algorithm. The overall algorithm is described below:

❶ Start with a pattern of minimal size k= 2, i.e. $\rho' = (b_{n-2}, b_{n-1})$.

❷ Search the time-series (b_1, ... b_{n-3}) to find the closest match for ρ'. Suppose that the closest match is found as $\rho'' = (b_{j-1}, b_j)$. Corresponding segment lengths for ρ' and ρ'' are (δ_{n-2}, δ_{n-1}) and (δ_{j-1}, δ_j) respectively. Here j is termed as the marker position.

❸ If $b_{j+1} = 1$ then predict *high*

$y_{n+1} = f(y_n, |b_{j-1} - b_{n-2}|, |b_j - b_{n-1}|)$

If $b_{j+1} = 0$ then predict *low*

$y_{n+1} = f(y_n, |b_{j-1} - b_{n-2}|, |b_j - b_{n-1}|)$

If $b_{j+1} = 2$ then

$y_{n+1} = y_n$

❹ Minimise the Mean Square Error (MSE) and Mean Absolute Percentage Error (MAPE) by optimising pattern size. Repeat steps 1-4 for patterns of size k = 3, 4, ... Choose the optimal model: a model which yields minimal error with the least complexity (law of parsimony).

In the above algorithm, several features of the structural pattern recognition have been simplified to aid the understanding of the basic procedure. Now we detail more sophisticated features of the procedure used for a more general definition of patterns $\rho' = (b_{n-k} \ldots b_{n-2}, b_{n-1})$ and $\rho'' = (b_{j-k} \ldots b_{j-1}, b_j)$, each of size k, where the value of j (also called as a marker) is known. The position of the marker lies between the [1, n-k-1] range.

❺ The matching algorithm tries to find a ρ'' which is most similar to ρ' by minimising offset ∇.

for J = 1 to n-k-1 do

{

$$\nabla = \sum_{i=1}^{k} w_i(\delta_{n-i} - \delta_{j-i}) \quad \ldots (1)$$

If ∇ < low then low = ∇ and j = J

}

Here low is a initial threshold value. The final value of j (representing the closest ρ'') at the end of the above loop is used in step 3 for generating a prediction. In our experimentation, w_i is equal to 1.

❻ The function f in step 3 may be further clarified.

If $b_{j+1} = 1$ then predict *high*

$$y_{n+1} = y_n + \beta * \delta_{j+1} \quad \ldots (2)$$

If $b_{j+1} = 0$ then predict *low*

$$y_{n+1} = y_n - \beta * \delta_{j+1} \quad \text{where } \beta = 1/k \sum_{i=1}^{k} \delta_{n-i}/\delta_{j-i} \qquad ...(3)$$

- The algorithm is optimised for minimal error in prediction. The error measures used in this paper include:

  Mean Square Error (MSE) $= 1/p \sum (y_n - \ddot{y}_n)^2$

  Mean Absolute Percentage Error (MAPE) $= 1/p \sum |y_n - \ddot{y}_n| / y_n$

  Direction of change error = error when $y_n - y_{n-1} > 0$ and $\ddot{y}_n - y_{n-1} \leq 0$

  or error when $y_n - y_{n-1} \leq 0$ and $\ddot{y}_n - y_{n-1} > 0$

where y_n is the actual forecast or the event that occurs, y_{n-1} is our most recent value before the forecast, $\ddot{y}_n$ is our prediction and p is the total number of points predicted (test size).

EXPERIMENTAL DETAILS

In this paper, the algorithm described in the previous section will be used to predict a total of four series. Three of the benchmark series (A, D and E) considered here come from the Santa Fe competition (Weigend and Gershenfeld, 1994). The fourth series is the real S&P index for US financial market (monthly data from August 1988 to August 1996). The details of these series are introduced below:

Series A : This is a univariate time series measured in a Physics laboratory experiment. This data set was selected because it is an example of complicated behaviour in a clean, stationary, low-dimensional non-trivial physical system for which the underlying dynamic equations are well understood. There are a total of 1000 observations. The correlation between y_t and y_{t-1} for the original series is .53 and for the difference series is .27.

Series D: This univariate time-series has been generated for the equation of motion of a dynamic particle. The series has been synthetically generated with relatively high-dimensional dynamics. There are a total of 4572 observations. The correlation between y_t and y_{t-1} for the original series is .95 and for the difference series is .72.

Series E: This univariate time-series is a set of astrophysical data (variation in light intensity of a star). The data set was selected because the information is very noisy, discontinuous and non-linear. There are a total of 3550 observations. The correlation between y_t and y_{t-1} for the original series is .81 and for the difference series is -.44.

Series S&P: This series represents the S&P index over a period of eight years. This data is noisy and exponentially increasing in nature. There are a total of 2110 observations. The correlation between y_t and y_{t-1} for the original series is .99 and for the difference series is .04.

The statistical characteristics of these series are summarised in Table 1.

Table 1 here

In our experimentation, we compare the results obtained using the fuzzy pattern matching algorithm and the well known statistical exponential smoothing method. The exponential smoothing method is a special class of ARIMA model (Delurgio, 1998) explained by the following equation:

$$y_{n+1} = \alpha(y_n + (1-\alpha)y_{n-1} + (1-\alpha)^2 y_{n-2} + (1-\alpha)^3 y_{n-3} + (1-\alpha)^4 y_{n-4}) \quad \ldots (4)$$

In our experiments, the optimal value of α for series A, D, E and S&P are .99, .99, .3 and .6 respectively. The time-series is initially divided into an estimation (training) period and forecast (test) period. The estimation period is used for finding structural patterns in past data (calculating statistical characteristics of the series for statistical techniques). The test period is used for making the forecasts on the basis of previously known values and these results are compared against actual events for computing the test error. Good models are characterised by low MSE and MAPE in test phase along with low direction error. Another important feature of good forecast models is that they yield errors with a random distribution. For forecast techniques that optimise parameters during estimation phase, for example neural networks, usually both train and test errors are quoted. However, for a pattern matching technique where such an optimisation is not performed, only test errors will be quoted in the next section.

RESULTS

In this paper we compare the performance of the proposed structural pattern recognition method with the Exponential smoothing method of forecasting which is a special case of ARIMA model. The results section is split into two sub-sections: the first section discusses the results on the three benchmark series A, D and E; the second section discusses the results on the US S&P index. In both sub-sections, we forecast the last 10% (estimation period of 90%) and the last 25% (estimation period of 75%) of the time-series to document results. The results are reported for the different error measures as described in section II.

Performance on Benchmark series

As mentioned before, the time series is divided into two parts: estimation or training period which contains either 90% or 75% of the total data, and the test part with 10% or 25% of the remaining data for forecasting. The forecast errors are averaged over the test data. Table 2

shows the results obtained on the benchmark series A. In Table 2, the fuzzy pattern recognition method performance is shown for patterns used for matching ρ' of varying size (k =2 ...5). This procedure of using patterns of varying size for deriving the optimal model is very much similar to the development of an optimal neural network architecture by optimising the number of hidden nodes. The best performance on different measures has been *italicised* in Table 2 (also Tables 3, 4 and 5; only the best performance for the Exponential Smoothing method are displayed for optimal α).

Table 2 here

It was observed experimentally that increasing pattern size beyond a certain threshold gives a poor model with higher error. In general we find that patterns of size k = 4 give reasonably good performance. The overall performance is much superior to that obtained using the exponential smoothing method. The MSE and MAPE error measures show acceptable values for the proposed system; the direction error quoted here at nearly 5-6% is very encouraging.

Table 3 shows the results on series D which is supposed to be more difficult to predict than the previous series. Here we observe the best performance of the pattern recognition system for pattern ρ' of size k = 3. The MSE and MAPE measurements are lower than those obtained using the Exponential smoothing method.

Table 3 here

Finally, Table 4 shows the results for series E. This is the most difficult of all benchmark series to predict. This may be directly observed as the success in predicting the direction of series change is much lower than Table 2 and Table 3.

Table 4 here

Here, the best results are obtained for small k, e.g. k = 2. The pattern recognition approach to prediction is superior on the direction of error and better on the MSE and MAPE measures.

We summarise the following points of observation from Tables 1, 2 and 3.

- The pattern recognition approach is superior to the statistical exponential smoothing approach on all three benchmark time-series when compared on the MSE, MAPE and direction of series change error measure.
- The proposed approach may be vulnerable to wrong matches in highly noisy series as in Series E. This should require further detailed investigation in future studies.
- The number of segments used to formulate ρ' has an important bearing on the prediction success.

Performance on the US Financial Index - S&P series

One of the advantages of evaluating new forecasting systems on artificial benchmarks is the ability to reliably describe the operational characteristics of the prediction algorithm since the underlying nature of the problem domain is well specified. In other words, since the mathematical nature of the time-series is well defined, the evaluation of forecasting systems is easier. It may be reasonably claimed that what works on a given time series, will continue to work well on another time-series which has a similar mathematical framework.. However,

it is equally important that forecasting systems perform reasonably well on real data. A large amount of real data in time-series research mostly comes from financial domains, e.g. predicting sales, profit, labour costs, financial index, etc. A large number of studies have tried to predict the financial indices usually taking a multivariate approach with neural networks, e.g. (Delurgio, 1998). One of the most popular series often predicted is the US financial index, the S&P series whose characteristics have been detailed in section III of this paper.

The prediction of most financial indices is not straightforward. The S&P series is not *stationary*. A stationary series is one whose mean, variance and other statistical indices are uniform with time (as in Series A, D and E, Fig. 2-7). The S&P series shows an exponential upward growth. For this reason, one of the standard models used in the markets is the random geometric walk model which is defined as:

$$y_{n+1} = y_n (1+ \varepsilon) \quad \ldots (5)$$

where $0 \leq \varepsilon \leq 1$. The optimal value of the constant ε used for S&P series is .0056 [Delurgio, 1998, p. 297].

Another simpler method of working with non-stationary time-series is to predict the difference of a time-series or a difference of the difference time-series. These series are usually stationary in nature and statistical methods perform better on these resultant series. If our predictions on the difference time-series are better than the original series, we can translate our predictions to original series predictions with better results. A word of caution before moving on to the results. The aim of our experimentation is not to develop a market strategy for making profit but to demonstrate how well the proposed system predicts compared to a well established statistical method on the same data. However, a more

detailed analysis would allow us to plot profit graphs and outline a market strategy advising the trader when to stay in the market and how to invest. A more general discussion on the nature of international markets may be found in (Levich, 1987).

Table 5 here

Table 5 shows the performance of the pattern recognition system on the difference of difference S&P series. Pattern size of k = 3 yields the best performance. The pattern recognition system has an impressive performance on all measures. Specially, the important result in favour of the pattern recognition method is the high accuracy on predicting the direction of series change. This measure is of considerable importance to traders in the financial markets. At a give time *t* with the market in state y_n, it is important for the traders to know whether the market will go up $y_{n+1} > y_n$ or down $y_{n+1} < y_n$. In case the market goes up and the trader predicts correctly that the market will go up, i.e. ($y_n - y_{n-1} > 0$ and $\ddot{y}_n - y_{n-1} > 0$), the trader makes a profit; similarly if the market goes down and the trader correctly predicts that the market will go down, i.e. ($y_n - y_{n-1} < 0$ and $\ddot{y}_n - y_{n-1} < 0$), the trader minimises any risks taken. However, if the trader makes an error on predicting market change, they mostly end up making losses. Hence, a good forecast system should correctly predict the market position in relation to its current index with better than chance (50%) accuracy. Table 5 shows that the proposed system performs well roughly three out of four times. This is a very encouraging performance.

It was mentioned earlier in this paper that a good forecast model can be evaluated by observing its error statistics. Ideally, a good forecast model should generate random errors (there should be no relationship between error e_n and past errors e_{n-1}, e_{n-2}, ... e_{n-r} where *r* is the error lag). One method of evaluating a model is therefore to calculate the correlation between its test error at time *t* and time *t-r*. This is known as auto-correlation at lag *r*. The

auto-correlation will lie within the [-1, +1] range: an auto-correlation coefficient of -1 represents a very strong inverse relationship, whereas an auto-correlation coefficient of +1 represents a very strong positive relationship. Ideally, this coefficient should be as close to zero as possible showing no relationship between error values as different lags (random errors).

Table 6 here

Table 6 shows the auto-correlation coefficients between error at a given time t and at lags *t*-1, *t*-2, *t*-3 and *t*-4. The first value in each table box shows the result for the test size of 10% and the second value in *italics* shows the result for 25% test size. The results are very impressive. In all benchmark series and S&P series, the coefficients are very low as desired. This confirms that the proposed model does not produce errors that are related or biased in any way.

CONCLUSION

In this paper, a new pattern recognition method for time-series forecasting has been proposed. The method uses structural primitives (patterns) for forecasting. Present patterns are matched with past patterns to identify the direction and magnitude of the next prediction. The proposed method was evaluated against the Exponential smoothing method used for forecasting. The results on three benchmark series and the real S&P index prediction are very encouraging. The proposed method is simple to use and works in real-time (especially used for tick-type data in financial markets where the system can be re-trained in real-time). Further work should now be prompted at investigating the ability of the proposed method to work with series contaminated with noise of different underlying distribution and comparing its performance against other artificially intelligent methods such as neural networks.

References

Aerrabotu, N., Tagliarini, G. A. and Page, E. W. 1997. Ensemble encoding for time series forecasting with MLP networks, In *Proceedings of the SPIE-The International Society for Optical Engineering*, 3077:84-89.

Azoff, M. E. 1994. *Neural Network Time Series Forecasting of Financial Markets*, John Wiley and Sons.

Bonnet, D., Perrault, V. and Grumbach, A. 1997. Delta -NARMA neural networks: a connectionist extension of ARARMA models, *In Proceedings of the 5th European Symposium on Artificial Neural Networks*(ESANN '97), 127-132.

Chorafas, D. N. 1994. *Chaos Theory in the Financial Markets: Applying Fractals, Fuzzy Logic, Genetic Algorithms, Swarn Simulation & the Monte Carlo Method to Manage Markets*, Probus Publishing Company.

Daijin, K. 1997. Forecasting time series with a genetic fuzzy predictor, *Journal of KISS[B] (Software and Applications)*, 24(2):193-206.

Delurgio, S. 1998. *Forecasting: Principles and Applications*, McGraw-Hill.

Hendry, D. F. 1995. *Dynamic Econometrics*, Oxford University Press.

Kingdon, J. 1997. *Intelligent Systems and Financial Forecasting*, Springer-Verlag.

Wesley.

Levich, R. M. 1987. *International Financial Markets: Prices and Policies*, McGraw-Hill.

Mahfoud, S and G. Mani, 1997. Financial forecasting using genetic algorithms, *Applied Artificial Intelligence*, 10(6):543-565.

Motnikar, B. S., Pisanski, T. and Cepar, D. 1996. Time-series forecasting by pattern imitation, *OR Spektrum*, 18(1):43-49.

Muhammad, A. and King, G. A. 1997. Foreign exchange market forecasting using e evolutionary fuzzy networks, *In Proceedings of IEEE/IAFE 1997 Computational Intelligence for Financial Engineering* (CIFEr), 213-219.

Pellizzari P. and Pizzi, C. 1997. Fuzzy weighted local approximation for financial time series modelling and forecasting, *In Proceedings of IEEE/IAFE 1997 Computational Intelligence for Financial Engineering* (CIFEr), 137-143.

Rolf, S., Sprave, J. and Urfer, W. 1997. Model identification and parameter estimation of ARMA models by means of evolutionary algorithms, *In Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering* (CIFEr), 237-243.

Studer, L. and Masulli, F. 1996 On the structure of a neuro-fuzzy system to forecast chaotic time series, *In Proceeding of the 1st International Symposium on Neuro-Fuzzy Systems* (AT'96), 103-110.

Teran, G., Draye, T. P., Pavistic, D., Calderon, G. and Libert, G. 1997. Predicting a chaotic time series using a dynamical recurrent neural network, *In Proceedings of the 3rd International Workshop on Image and Signal Processing on the Theme of Advances in Computational Intelligence* (IWISPO '96), 115-118.

Weigend, A. S. and Gershenfeld, N. A. eds. 1994. *Time Series Prediction: Forecasting the Future and Understanding the Past*. Reading, MA: Addison-Wesley.

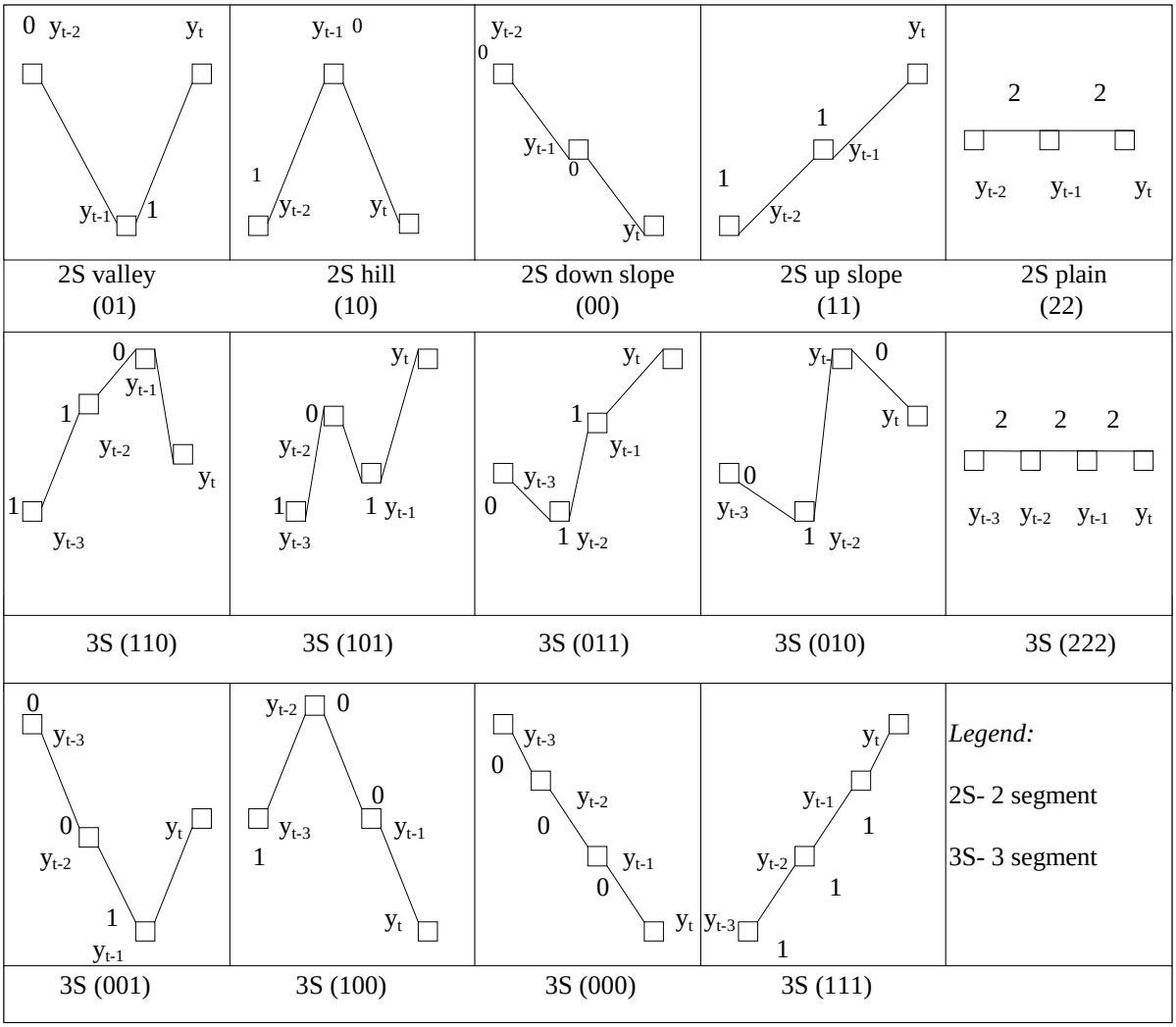

Figure 1. Pattern description for size two and three in any univariate series y(t).

Table 1. Series A, D, E and S&P statistics

Series	Min	Max	Mean	SD	Size
A	2	255	59.90	46.87	1000
D	.05	1.15	.58	.23	4572
E	-.31	.34	0	.10	3550
S&P	257.10	678.50	422.74	100.32	2110

Table 2. Series A performance of the pattern recognition system with varying segment size k compared with the Exponential Smoothing method

No. of segments k	MSE	MAPE	% direction success	% size predicted
2	*169.7*	14.4	*95*	*10*
	201.1	12.1	*94*	*25*
3	216.9	13.0	94	*10*
	115.5	9.1	94	*25*
4	202.4	*11.8*	95	*10*
	126.7	*8.7*	94	*25*
5	184.0	13.0	94	*10*
	115.6	9.0	94	*25*
ES method	*3158.8*	*83.9*	*70*	*10*
	1930.2	*61.6*	*70*	*25*

Table 3. Series D performance of the pattern recognition system with varying segment size k compared with the Exponential Smoothing method

No. of segments k	MSE	MAPE	% direction success	% size predicted
2	.003	9.6	82	*10*
	.003	9.1	79	*25*
3	*.003*	*8.6*	*84*	*10*
	.003	*8.8*	*81*	*25*
4	.003	9.4	82	*10*
	.004	9.7	79	*25*
5	.003	9.2	81	*10*
	.004	9.6	80	*25*
ES method	*.005*	*9.4*	*80*	*10*
	.003	*8.8*	*79*	*25*

Table 4. Series E performance of the pattern recognition system with varying segment size k compared with the Exponential Smoothing method

No. of segments k	MSE	MAPE	% direction success	% size predicted
2	.015	8.6	66	*10*
	.014	8.9	68	*25*
3	.020	10.2	69	*10*
	.023	11.1	69	*25*
4	.086	12.2	68	*10*
	.557	14.5	70	*25*
5	.481	16.3	65	*10*
	.213	13.9	68	*25*
ES method	*.033*	*17.55*	*56*	*10*
	.034	*17.37*	*57*	*25*

Table 5. Series S&P performance of the pattern recognition system with varying segment size k compared with the Exponential smoothing method.

No. of segments k	MSE	MAPE	% direction success	% size predicted
2	82.9	1.1	73	*10*
	54.7	0.9	71	*25*
3	82.3	1.1	76	*10*
	52.9	0.9	72	*25*
4	83.0	1.1	72	*10*
	57.0	1.0	67	*25*
5	92.4	1.1	69	*10*
	59.3	1.0	69	*25*
ES method	*4379.2*	*9.6*	*48*	*10*
	2786.2	*8.2*	*49*	*25*

Table 6. Performance validation of the pattern recognition model: Auto-correlations between lagged error values for $1 \leq r \leq 4$.

Time lag → *Series* ↓	1	2	3	4
A	.17	-.03	-.03	.01
	.11	*-.01*	*-.03*	*-.01*
D	.10	.005	.005	.04
	.08	*.02*	*.04*	*.04*
E	.04	.02	-.08	.01
	-.04	*-.01*	*-.07*	.009
S&P	-.50	.13	-.10	.08
	-.44	*.08*	*-.09*	*.05*

Figure 2. Plot of Series A

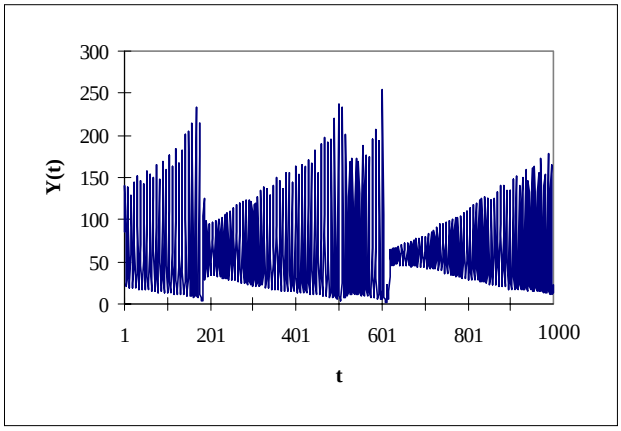

Figure 3. Plot of the difference of series A

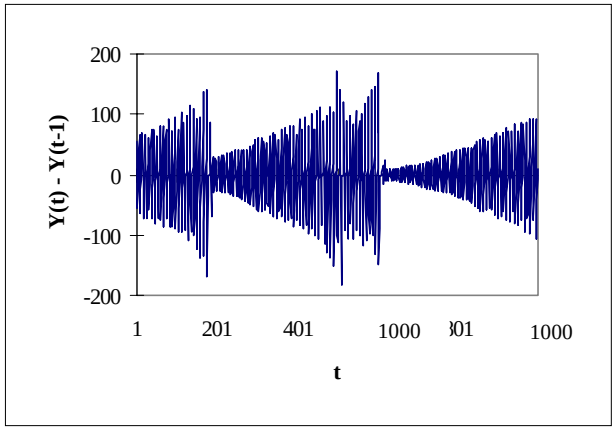

Figure 4. Plot of Series D

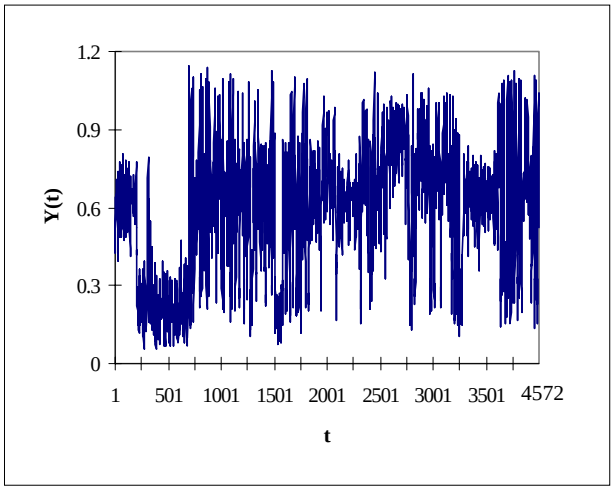

Figure 5. Plot of the difference of series D

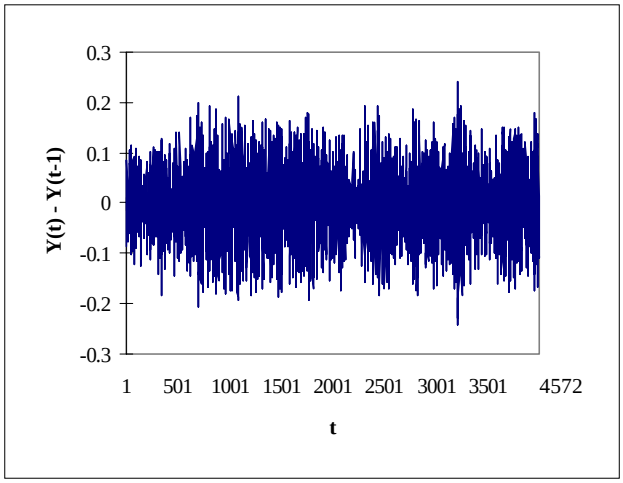

Figure 6. Plot of Series E

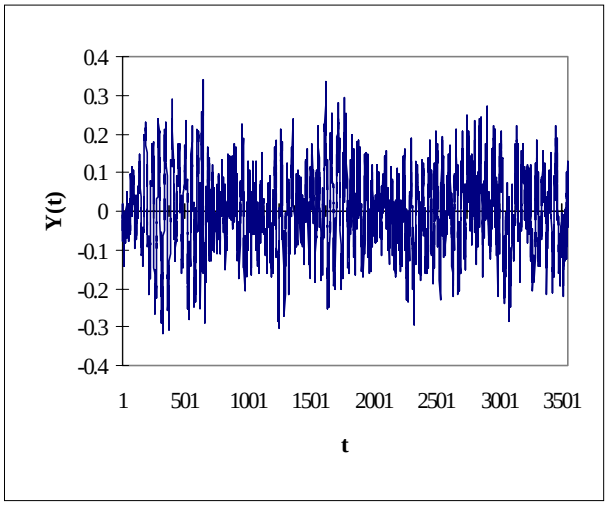

Figure 7. Plot of the difference of series E

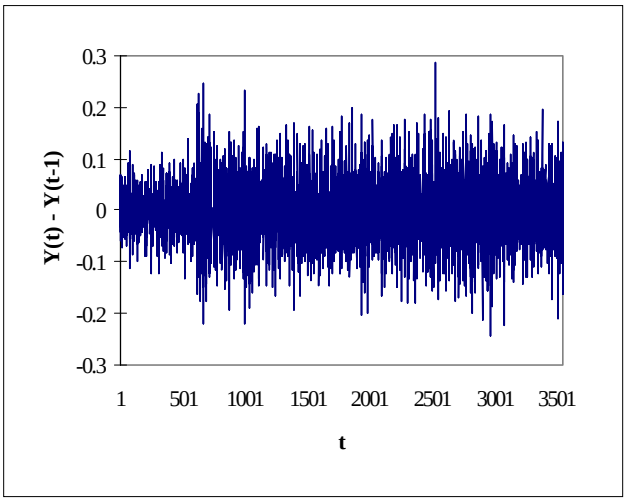

Figure 8. Plot of S&P index

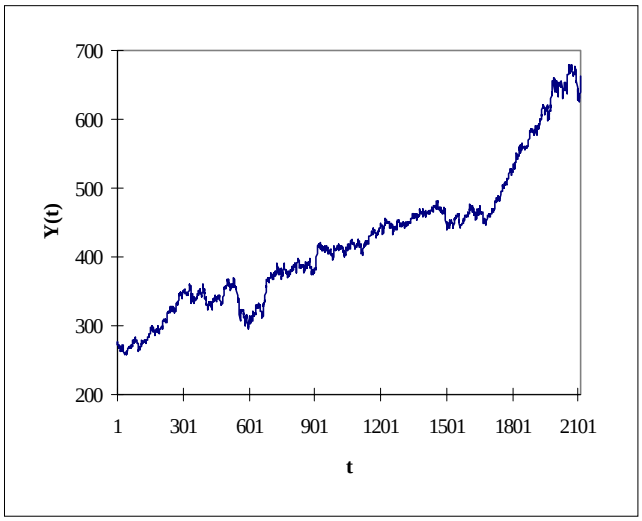

Figure 9. Plot of the difference of S&P index