International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:1, No:10, 2007

An Engineering Approach to Forecast Volatility of Financial Indices

Irwin Ma, Tony Wong, and Thiagas Sankar

***Abstract*—**By systematically applying different engineering methods, difficult financial problems become approachable. Using a combination of theory and techniques such as wavelet transform, time series data mining, Markov chain based discrete stochastic optimization, and evolutionary algorithms, this work formulated a strategy to characterize and forecast non-linear time series. It attempted to extract typical features from the volatility data sets of S&P100 and S&P500 indices that include abrupt drops, jumps and other non-linearity. As a result, accuracy of forecasting has reached an average of over 75% surpassing any other publicly available results on the forecast of any financial index.

***Keywords*—** Discrete stochastic optimization, genetic algorithms, genetic programming, volatility forecast

I. The Problem and Approach

VOLATILITY is the standard deviation of the rate of return distributions and is a commonly accepted measure of *risk* in the investment field. Integrated Volatility (IV) is calculated from the cumulative squared intraday returns of the underlying securities at high frequencies as defined by Anderson *et al* [1]. The daily volatility is a crucial variable in evaluation of option prices and in conducting different hedging strategies. The importance of volatility estimation/forecast is further exemplified *by the Nobel Prize awarded in 2003 to Professor Robert F. Engle for his pioneering work on modeling volatility dynamics*.

The research outlined in this paper intends to establish a systematic approach by using the existing engineering methods to forecast effectively the volatility of a selected financial market i.e., to improve the accuracy of the forecast. The approach will revolve around evolutionary algorithms (EA) within a Time Series Data Mining (TSDM) framework, which is supported by a Markov chain based random search method specifically for discrete stochastic optimization. A key concept borrowed from engineering practices is data transformation prior to any in depth analysis. As such, a time series of an equity index is first converted into IV's and then wavelet coefficients. By converting the latent variance of an index into an effectively observable IV time series, we arrive at a typical time series forecasting problem. Instead of going the classical econometric route, a wavelet transform is used to preprocess the IV data, and then a combined genetic algorithm (GA) and genetic programming (GP) approach is used to explore the hidden repetitive patterns for the forecasting purpose. By treating the wavelet coefficients as 4-lag recursive data, we apply simple "`IF/THEN`" rules with GA's and similarly with GP's in order to capture the typical patterns most frequently found in the data set including the nonlinear cases. Since wavelet coefficients are far smaller in size compared with the original time series, calculation efficiency could be achieved. The research covered in this paper intends to forecast both the direction and range of the volatility movement with GA's and the corresponding values with GP's. Two indices of S&P500 and S&P100 are obtained from different sources and analyzed with a similar approach. The results are compared with each other for validation purpose.

This paper could be divided into three sections, Section I provides background information regarding volatility forecasting and the general plan and methodology of the current research. Section II lays the theoretical foundation for the practical approach adopted in this work, *i.e*. the rationale of converting the IV series into a 4-lag recursive data set. And the third part describes the procedure to implement the proposed IV-wavelet-EA method including the analysis on real data and the corresponding results.

A. Literature Review

As mentioned in Ma *et al* [2], traditional financial engineering methods based on parametric models such as the GARCH model family, seem to have difficulty to improve the accuracy in volatility forecasting due to their rigid as well as linear structure. The requirement of distribution assumption further hinders the forecasting accuracy [3]. Except those based on proprietary methodology, there is still a lack of publicly available and effective method to deal with the non-linearity inherent in the volatility series of financial indices [4, 5]. In this regard, recent development in financial time series analysis could be beneficial to these forecasting problems.

Work conducted in [6] has established that by using a GA method, the one day ahead moving direction and range of the volatility of selected underlying securities could be forecasted at an average accuracy of over 75%. Table I summarizes the important characteristics of the reviewed papers, which

Manuscript received July 16, 2005. This work was supported by research grants awarded to Professor Wong and Professor Sankar by The Natural Sciences and Engineering Research Council of Canada.

Irwin Ma, Tony Wong, Thiagas Sankar are with the Department of Automated Manufacturing Engineering, École de technologie supérieure, University of Québec, Montréal (Québec), Canada (email: acron@aei.ca, {tony.wong|thiagas.sankar}@etsmtl.ca})University of Québec, Montréal (Québec), Canada (email: acron@aei.ca, {tony.wong|thiagas.sankar} @etsmtl.ca})

provides us with guidelines to further our discussion.

TABLE I.
SUMMARY OF THE PAPERS REVIEWED IN DETAILS

AUTHOR(S)	APPROACH	GOAL	COMMENTS
Pictet *et al*., [7]	GP	Discover new FX volatility models using "typed" GP trees.	Does not take into account non-linearity. Use FX symmetry to reduce considerably the search space.
Zumbach *et al*., [8]	GP+LS	Use hybrid GP to forecast FX volatility.	Does not take into account non-linearity. Use FX symmetry to reduce considerably the search space.
Chen & Yeh, [9]	GP	Use a recursive GP to detect and adapt to structural changes of market volatility.	Explicit recognition of non-linearity but does not attempt to forecast. Integrated Volatility was not used.
Neely & Weller, [4]	GARCH, GP, RiskMetrics	Compare three approaches: Parametric, generalized parametric and non-parametric in FX forecasting.	In many instances, GP out-performed the other approaches. They were tested on FX volatility only. Non-linearity is not accounted for. No IV.
Kaboudan, [10]	GP, wavelet, NN	Apply an integrated approach to forecast one-step as well as 16-step-ahead exchange rate forecasting.	Does not deal with volatility. Other type of wavelet might improve the effectiveness.
Lee, [11]	ANN + GP, GARCH	Compare the computation intelligence method with GARCH models.	Better results are achieved at questionable calculation efficiency for medium to long forecasting horizons.
Lawrenz & Westerhoff, [12]	GA	Explore how trading rules can explain market volatility. Use GA to combine six simple trading rules using the chartists – fundamentalists point of view.	In real world there are more than two players (i.e. chartists and fundamentalists) and trading rules are much more complex.
Kinlay *et al*., [13]	GA, SM	Asset allocation and optimization system based on a weighted sum technique. The weights are determined by statistical inference and aided by a GA.	Proprietary techniques with many undisclosed details. Best published results with 72% – 75% prediction accuracy.
Fong & Szeto, [14]	GA	Use GA to determine simple if – then – else rules in order to predict the behavior of artificially generated time series.	Obtained 50% - 60% accuracy using only 100 simple if – then – else rules. Demonstrated the search power of GA applied to stochastic series.
Maheu, [15]	SM	Explores the nonlinear features of FX integrated volatility.	Found that stochastic jumps (structural changes) are a very determinant feature in IV.
Gaunersdorfer, [16]	SM	Attempt to define a nonlinear model that explains the volatility clustering phenomenon.	It concludes that the rate of return have non predictable behavior while the variance do show trend that is close to the index measured. Thus confirming the usefulness of the integrated volatility approach.
Dunis & Huang, [17]	NN	Applied a non-linear non-parametric approach to forecast and trade FX	Achieved slightly above 50% of forecast accuracy. But elaborate models produced poor results.
Wang *et al.* [18]	CEV	Account for the non-linearity in volatility with a stochastic jump-decay process	Provides further theoretical foundation for the current research to deal with non-linearity.
Hovspian, *et al.* [19]	SVC, periodogram, GARCH	Detect and predict periods of relatively increased volatility by a synthesizing method.	It is still a GARCH based approach, *i.e.* parametric models and lacks verification with real data sets.
Tino *et al.* [20]	Sparse Bayesian Kernel	By quantizing real value time series, forecast the one-day-ahead volatility to generate profit.	Only forecasts the directions of volatility.
Gavrishchaka, [21]	Boosting framework	Make optimal investment decisions by forecasting the directions of volatility.	Other calculation methods could help improve efficiency. Only forecasts the directions of volatility.

GA – Genetic Algorithm, GP(+LS) – Genetic Programming (with local search), SM – Statistical Methods, FX – Foreign Exchange.

Based on the literature survey, one can conclude that the contemporary research of volatility forecast has started to venture into the non traditional domain particularly in the computational intelligence area such as GA/GP in a backdrop of active IV research; each has its own advantages. IV provides a good starting estimation of the current volatility, and as indicated in previous sections, researchers could apply a variety of techniques including stochastic analysis to forecast future volatility more accurately. GA/GP on the other hand, could effectively and progressively efficiently deal with non-linearity, which opens up an alternative avenue besides the effort made by the traditional academics.

B. Research Objectives

This paper intends to establish a systematic approach and eventually a software tool for analysts to forecast more precisely the direction, range or even the value of future volatility of financial indices as well as different equities. In other words, this volatility forecasting method should be:

- free of strong assumptions, *e.g.* no need to assume normal or any other statistical distributions associated with the time series and its estimation errors;
- more flexible, *i.e.* not limited by the parametric structure;
- more accurate on the current and hence the forecasted volatility, *i.e.* IV transforms volatility from a latent variable into an observable variable;
- more efficient because of the data preprocessing by means of wavelet transform.

C. Research Procedure

It is apparent that most of the contemporary researchers use EA in time series analysis simply as a tool without building up rigorous theoretical substantiation for its application in the process.

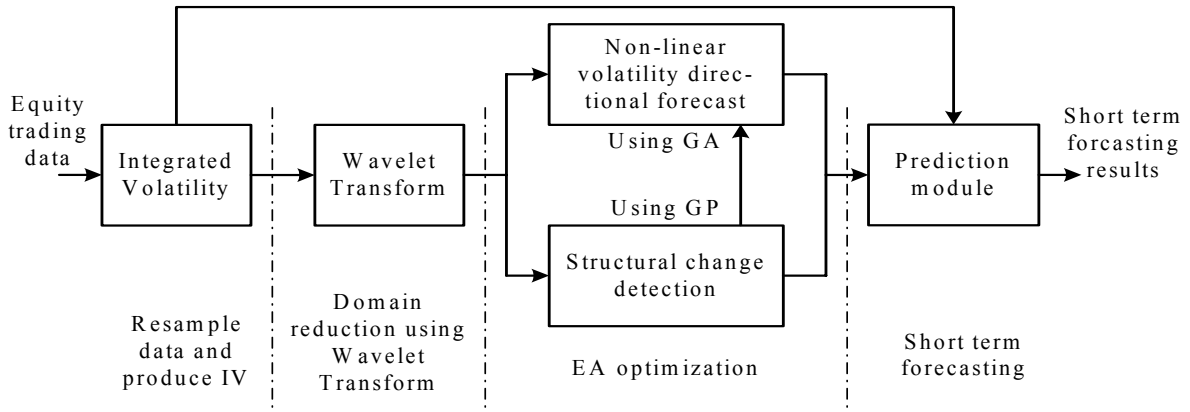

Fig. 1 Proposed volatility directional forecasting system architecture

EA be it GA or GP is typically lack of rigorous mathematical proof even though it is a powerful tool for optimization. In order to lay the theoretical foundation for the current approach, the IV time series is first converted into a four-lag recursive series in the TSDM framework as shown in Section II.*A* and *B*. In Section II.*C* we found that the resultant series is in fact a Markov chain and thus demonstrated that a Markov chain based discrete stochastic optimization method could provide the theoretical support for applying GA's for the forecast purpose. The following figure summarizes the architecture of the proposed IV-wavelet-EA time series forecasting approach. Figure 1 illustrates the operational procedure to implement the proposed method in which the application of a data reduction and analysis technique converts a difficult volatility forecasting problem into a classical signal analysis one. And such a principle is one of the main contributions made by this research.

II. Theoretical Foundation

A. Data Mining of the IV Time Series

Data mining is the analysis of data with the goal of uncovering hidden patterns especially those complex relationships in large data sets. Predictive data mining is a search for very strong patterns in those data that can be generalized to make accurate future decisions [22].

Povinelli [22] introduced the Time Series Data Mining (TSDM) framework, which differs fundamentally from most of other contemporary approaches. The TSDM framework helps reveal hidden temporal patterns that are characteristic and predictive of time series events. This contrasts with other time series analysis techniques, which attempt to characterize and predict all observations. There are inherently many different patterns in financial time series, linear and nonlinear. A financial series is a dynamic entity which is affected by many variables, be it economical, financial, political, psychological, legal, *etc*. It is philosophically unwise to use one fix model, linear or nonlinear to estimate such a process, let alone forecasting. We believe that instead of using one single formula to explain the entire time series, a better idea would be to use multiple GA's and/or GP's exploring sequentially in the search space to obtain an overall estimation represented by a set of formula rules. For example, the GP's try to find the best fitting rules based on the input time series. The best formula rules are then combined and used to forecast the future IV values. The same applies to rules in the GA approach.

B. Rule-based EA Forecasting Method

In this section and in the Appendix, background information regarding Povinelli's TSDM methods is introduced. The TSDM methods create a new structure for analyzing time series by adapting concepts from data mining, time series analysis, EA's, and nonlinear dynamics system. They are designed to predict non-stationary, non-periodic and irregular time series, and not restricted by the use of predefined templates. More specifically, they help discover hidden temporal structures predictive of sharp movements in time series, using a time-delay embedding process that reconstructs the time series into a phase space that is topologically equivalent to the original system under certain assumptions [22]. The TSDM methods are developed and applicable to make one-step predictions for time series data sets. In order to extract non-stationary temporal patterns, a specific TSDM method could be used to address quasi-stationary temporal patterns, *i.e.,* temporal patterns that are characteristic and predictive of events for a limited time window *Q*. It is called the Time Series Data Mining evolving temporal pattern (TSDMe2) method, which uses a fixed training window and a single period prediction window. The TSDMe2 method differs from the other TSDM methods in how the observed and testing time series are formed. The TSDMe2 method creates the overlapping observed time series:

$$X_j = \{\theta_j, t = j, \ldots, j+N\}. \quad (1)$$

The testing time series is formed from a single observation:

$$Y_j = \{\theta_j, t = j+N+1\}, \quad (2)$$

where θ_j is the time series value at time $t=j$, and N is the size of the window.

In characterizing different patterns hidden in the time series, there are two key factors to consider, number of pattern types and size of the patterns (or window Q). By parsimony, the simplest characterization of events possible is desired *i.e.*, as small a dimensional phase space Q as possible and as few characterization patterns as necessary. However, the following modifications have been made to the typical TSDM in order to implement the proposed data mining procedure,

i) to increase the pattern characterizations by involving as many as 100 different arithmetic expressions to describe a windowed time series;
ii) to use a 4-lag recursive memory as the size of the patterns Q. For definitions of some related concepts, refer to Appendix I.

By using 100 different formula/rules to match the frequently appearing events and to extract different patterns buried in noise, there is a high probability to extract the patterns and further to forecast the one step-ahead activity. Note that in each of the 100 rules used to characterize different patterns, the value of δ could be considered as the margin of accuracy the rules match the points in the window. The 4-lag recursive system is used due to the economy of time and memory and is particularly useful in dealing with volatility forecasting because of past research showing that most of the information is contained in the most recent lags, resulting the popularity of GARCH(1,1) or other short memory models. Moreover, the application of volatility estimation in option trading deem necessary to extract also the non-event, so that one could capitalize on the time value of the option. The potentially wide variety of the 100 rules could also help extracting those non events.

To find the different temporal patterns the time series is embedded into a reconstructed phase space with a time delay of one and a dimension of four [22]. The patterns are determined by the first four points in the windowed data while making a prediction for the value at the fifth time interval. Once the data is embedded, temporal structures are located using a GA/GP search. Pattern clusters are made of points within a fixed distance of the temporal structures δ. In case of using GP, the event characterization function $g(t) = \theta_{t+1}$, determines the value given to the prediction made from the clustering using the temporal structures. This value is the IV value for the next time interval. The temporal structures are next ordered by how well each predicts the IV movements. A ranking function is defined as the average value within a temporal structure, and it is used to order the structures for optimization. The optimization is a search to find the best set of temporal structures and is done with GP that finds fitness value parameters that maximize the ranking function $f(P)$ – the frequency of the correct guessed patterns. The GP uses a combination of Monte Carlo search for population initialization with a fixed percentage selection, crossover and mutation to find the optimal P^*, and a limited number of generations to halt the GP [22, 23].

When GA is used, the IV window $X_j = \{\theta_j, t=j, \ldots, j+3\}$ will be converted into a set of numbers {1, 2, 3, 4, *} by classifying the range as (-∞, -a], (-a, b], (b, c], (d, ∞), where '*' means "`don't care`". Therefore, all data will become a sequence of numbers. The rules will take the form of <`IF` [(($\theta_j = I$) `AND/OR` ($\theta_{j+1} = J$) `AND/OR` (($\theta_{j+2} = K$)) `AND/OR` ($\theta_{j+3} = L$))], `THEN` ($\theta_{j+4} = M$)>, where the event characterization function $g(t) = \theta_{j+4}$ will be a number that predicts the range of the subsequent IV value. And δ will become obsolete. The key difference between using GP and GA is the form of the rule. More details could be found in Section III.*B*.

In general, combination can potentially eliminate the erroneous predictions which might be generated due to the noise in the data. In the case of the rule learning process, independent trials of the GA/GP can be considered to explore different parts of the search space, thereby learning different types of patterns for prediction. As a result, at a given time some rules generate better predictors than others, thus making them ideal candidates as base predictors to achieve increased predictive accuracy.

C. Markov Chain based Discrete Stochastic Optimization

The GA approach outlined above and employed in Ma *et al* [6] satisfied some stringent criteria and yielded forecasting accuracy that is higher than those derived from other publicly available research. GA methods have, however, certain drawbacks, *e.g.* GA's are not guaranteed to give an optimal solution and they lack convergence proof. Compared with other stochastic optimization techniques such as simulated annealing, it is less rigorous.

The recent advancement in discrete stochastic optimization methods provides the theoretical foundation to solidify the GA approach. For example, Andradottir [24] demonstrated the feasibility of applying the Markov chain method when the transitional matrix is initially non-time homogeneous and asymptotically approaches time homogeneous, unlike Duan [25] and most other work in the field, which are confined to time-homogeneous cases. However, the main difficulty while applying Markov chain theory to solve time series problem is that data in time series problems are typically correlated, while Markov chain by definition does not concern about the historical states prior to the current point. This is exemplified by the application of Markov chain method on the non-linear asymmetric GARCH(1,1) process, as done in Duan *et al*'s [25] research. Therefore, one needs to transform a time series into a Markov chain while maintaining the necessary characteristic of the original data, in order to make use of the rigorous mathematical theory to substantiate the stochastic optimization such as GA's.

The Markov chain approach allows one to decouple the partitioning of time and state. In other words, one can use time steps suitable for a particular contingent claim without being unduly constrained to have a particular set of state values, unlike other option valuation methods such as

binomial tree or lattice and finite difference method. Such a characteristic motivates the current IV data conversion into the overlapping four lag data groups thus making the optimization feasible when we use both Markov chain and GA together.

In the following sections we attempt to apply a Markov chain based discrete stochastic optimization method (DSOM) to substantiate the use of GA's in Ma *et al.* [6], which typically lacks rigorous mathematical proofs. In Section II.*C.1* Andradottir's [24] global search method for discrete stochastic optimization is introduced. In Section II.*C.3*, the method is applied to substantiate the use of GA's for volatility forecast. In Section II.*E*, we generally discuss the approach with respect to its limitations and potential applications.

C.1 Typical Markov Chain Method

The following is the general form of a discrete stochastic optimization problem that needs to determine global optimal solutions:

$$\max_{\theta\in\Theta} f(\theta), \quad \text{where } f(\theta) = E\{X(\theta)\}, \forall\theta\in\Theta. \tag{3}$$

- Here, $f: \Theta \to \Re$ is the objective function.
- Θ is the discrete feasible region containing at least two states; in the current case, for a finite feasible set, $\Theta^* \neq 0$, where $\Theta^* = \{\theta \in \Theta : f(\theta) \geq f(\theta') \text{ for all } \theta \in \Theta\}$ ($\theta' \in \Theta\backslash\{\theta\}$) is the set of global optimal solutions to the optimization problem; since $f: \Theta \to \Re$, the optimal value $f^* = \max_{\theta\in\Theta} f(\theta)$ is finite and can be achieved.
- $\{X(\theta):\theta \in \Theta\}$ is a collection of random variables having the property that $E\{X(\theta)\}$ cannot be evaluated analytically but estimated or measured.
- θ is a random variable in a stochastic process.

Rather than sequentially using either the current point or the most frequently visited point to estimate the optimal solution, Andradottir [24] proposed using *all the observed objective function values* generated as the random search method moves around the feasible region to obtain increasingly more accurate estimates of the objective function values at different points. At any given time, the feasible solution that has the best estimated objective function value, *e.g.* the largest one for maximization problems, is used as the estimate of the optimal solution. Numerical evidence presented by Andradottir [24] suggests that the use of this approach for estimating the optimal solution appears to yield improved performance relative to other approaches for estimating the optimal solution

C.2 More Contemporary Approach

Andradottir's [24] Lemma 3.1 assumes that P_m, $m = 0, 1, 2, \ldots$ and P are Markov matrices on the state space Θ such that P is irreducible and aperiodic and $P_m \to P$ as $m \to \infty$. If $q: \Theta \to \Re$, then as $M \to \infty$

$$\frac{1}{M}\sum_{m=1}^{M} q(\theta_m) \to \sum_{d=1}^{J} \pi_d q(d), \quad as\ M \to \infty, \tag{4}$$

where $\pi^T = (\pi_1, \ldots, \pi_J)$ is the steady-state distribution corresponding to P, while $\{X_m\}$ is a non-homogeneous Markov chain with transition probabilities

$$P\{\theta_{m+1} = d \mid \theta_0, \ldots \theta_m\} = P_m(\theta_m, d) \quad \forall d \in \Theta \wedge m = 0,1,2,\ldots \tag{5}$$

In other words at iteration $m+1$, θ_{m+1} has $d=J$ possible states. Here the number of states d is countable and limited. θ_t could be, in the case of Ma *et al* [6] the successively overlapped 4-lag recursive data set that has been converted from the original IV time series. And it takes the states in the form of numerical values $\theta_t \in \{1, 2, 3, 4, *\}$, and $j = 1, \ldots, J$. Consequently, θ_t*'s* will form a typical rule θ with numerical values joined by operators "AND" and "OR". Details about the structure of the rules will be given in Section III.*B*. At the limit, the transitional matrix becomes time-homogeneous. By using Theorem 1 shown in Section II.*C*.5 with our GA operation, we look for rules that most frequently match with the actual overlapped 4-lag IV data. Those patterns that appear more often tend to be caught by rules derived from crossover and/or mutation, and will gradually lead to more successful estimates.

Andradottir' method [24] needs to maintain two variables for each point $\theta \in \Theta$, *i.e.* $K_m(\theta)$ would count how many estimates of $f(\theta)$ have been generated in the first m iterations, while $\sum_m(\theta)$ would contain the sum of all $K_m(\theta)$ estimates of $f(\theta)$ that have been generated in the first m iterations. The specific procedure is outlined as follows:

Algorithm 1 – Modified Global Search Method

Step 0: Select a starting point $\theta_0 \in \Theta$. Let $K_{-1}(\theta) = \Sigma_{-1}(\theta) = 0$ $\forall\ \theta \in \Theta$. Let $m = 0$ and $\theta^*_m = \theta_0$ and go to Step 1.

Step 1: Given the value of θ_m, generate a uniform random variable θ'_m on $N(\theta_m)$ independently of the past (so that $\forall\ \theta\in\Theta$, $\theta \neq \theta_m$, we have that $\theta'_m = \theta$ with probability $1 / (|\Theta| - 1)$). Go to Step 2.

Step 2: Given the value of θ_m and θ'_m, generate observations $X_{m,l}(\theta)$ of $X(\theta)$, for $l = 1, \ldots, L$ and $\theta = \theta_m$, θ'_m independently of the past. Let $R_m = \Sigma^L_{l=1} (X_{m,l}(\theta_m) - X_{m,l}(\theta'_m)/L$. if $R_m > 0$, then let $\theta_{m+1} = \theta_m$. Otherwise let $\theta_{m+1} = \theta'_m$. Go to Step 3.

Step 3: Let $K_m(\theta) = K_{m-1}(\theta) + L$ for $\theta = \theta_m$, θ'_m, and $K_m(\theta) = K_{m-1}(\theta)\ \forall\theta \in \Theta\backslash\{\theta_m, \theta'_m\}$. Moreover, let $\sum_m(\theta) = \sum_{m-1}(\theta) + \sum^L_{l=1} X_{m,l}(\theta)$ for $\theta = \theta_m$, θ'_m, and $\sum_m(\theta) = \sum_{m-1}(\theta)\ \forall\theta \in \Theta\backslash\{\theta_m, \theta'_m\}$. Let $\theta^*_m \in \arg\max_{\theta\in\Upsilon_m} \sum_m(\theta)/K_m(\theta)$, where $\Upsilon_m = \{\theta \in \Theta : K_m(\theta) > 0\}$. Let $m = m + 1$ and go to Step 1.

The main issue in using Algorithm 1 will be the way to use the state data generated by a random search method in order to obtain an estimate of the optimal solution. There is no particular requirement how θ *,* the rule should behave. On the other hand, $\{X(\theta):\theta \in \Theta\}$ should be a collection of random variables having the feature that $E\{X(\theta)\}$ is the unbiased and consistent estimation of $f(\theta)$, *i.e.* the prediction accuracy recorded at the current generation among all best rules. Details

regarding the rationale of unbiased and consistent estimation of $f(\theta)$ are given in Section II.*C.5*. The GA operation in Ma *et al* [6] repetitively loops through Step 1 through 3 until the last generation. Its main differences include a) evaluating far more rules per iteration, b) ranking instead of tournament as the selection method and c) limited number of rules instead of all historical rules involved in fitness value optimization. In Algorithm 1, rules in each group are first compared against each other pair by pair. Those rules that have better prediction rate will be retained for the next generation, *i.e*. the selection of θ_{m+1} based on the value of R_m. The approach here requires the search of optimized solution to be identified in Step 3, where $K_m(\theta)$ and $\sum_m(\theta)$ for each $\theta \in \Theta$ are stored, accumulated and compared for maximization. This is the key difference between Andradottir's method and others including the one used in Ma *et al* [6], *i.e*. all values of θ_{m+1} are kept in memory while GA's are ongoing. In every generation, new rules in the groups that have been derived from crossover and mutation in the previous generation will be put back into the pool to compare with those retained from the last generation. Only those new ones that have higher prediction rates will replace the respectively selected peers for the next generation. Either accepted or rejected they are recorded in memory together with other existing rules. At the last generation, among thousands of rules in the memory the optimization is performed with $\theta_m^* \in \text{arg max}_{\theta \in \tau_m} \sum_m(\theta)/K_m(\theta)$. The top 100 non-identical rules will be used for validation on another set of IV data. The detailed procedure in applying Andradottir's method with GA will be given in Section III.*B*.

C.3 Proposed Methodology

Algorithm 1 is quite a general form of optimization without explicit assumption on the variables. To apply Algorithm 1, we classify the IV time series into four ranges and randomly generate 100 groups of 100 rules, as shown in Section II.*B*. The first four elements in the "IF" part of the rule are used as the qualifying criteria and the "THEN" part is for predicting the subsequent IV value. We define a set $\theta = \{\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}, \theta_{t+4}\}$, where $\theta \in \{1, 2, 3, 4, *\}$. As a result, the collection of random variable

$$X(\theta) = \begin{cases} 1, & \text{if } \{\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}, \theta_{t+4}\} \text{ matches the data sequence;} \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

where $\{\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}, \theta_{t+4}\}$ represents the complete rule including the qualifying part $\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}$ and the prediction part θ_{t+4}. The nature of $X(\theta)$ makes it IID as required in equation (3). The problem is therefore, converted into a search of rules that best fit the four-point patterns in the IV data set, so that the immediate fifth IV value could be forecasted upon knowing the previous four points. Each rule with five recursive points in θ*'s* will be independent of each other or at least treated as independent in the eye of GA's, thus satisfies the requirement of Markov chain operation. A time series problem is thus converted into a set of random data that could be approached with the Markov chain method. Here, L is the smaller number of the possible matches derived by comparing θ_m and θ_m' and is at maximum equals the number of data points in the IV time series minus four, whereas m is the number of generations to perform GA. One important feature GA's incorporate in Step 2 is the way of generating $X_{m,l}(\theta)$ of $X(\theta)$, for l =1, …, L and $\theta = \theta_m, \theta_m'$ independently of the past. By applying GA's, θ'_m are generated through crossover or mutation, while $X(\theta)$ depends on whether the qualified rule predicts correctly. With the value of R_m we could choose either θ_m or θ_m' to go through further GA manipulation, *i.e*. crossover or mutation. At the last generation, we could retain θ_m^* as the optimal solution for the m^{th} generation by carrying out the optimization process. Note that the calculation of $K_m(\theta)$ could be modified as

$$K_m(\theta) = \begin{cases} K_{m-1}(\theta) + L, & \text{if } \{\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}\} \\ & \text{matches the data sequence;} \\ K_{m-1}(\theta), & \text{otherwise.} \end{cases} \quad (7)$$

where $\{\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}\}$ is again the qualifying part of the rule.

C.4 Procedure to Apply DSOM with GA

When apply Algorithm 1 to solve the current discrete stochastic optimization problem, we have Algorithm 2:

Step 0: Randomly assign any one value of {1, 2, 3, 4, *} to the first four fields in $\theta = (\theta_t, \theta_{t+1}, \theta_{t+2}, \theta_{t+3}, \theta_{t+4})$, randomly assign operators "AND" and "OR" to join these four fields and then assign θ_{t+4} = 1 for the first 25 rules. Repeat the same process with θ_{t+4} = 2, 3 and 4 respectively to form a total of 100 rules. Repeat the operation to generate another 99 such groups. Then randomly select 50 rules in each group as $\theta_0's$. Set all counters to zeros

Step 1: The rest of 50 rules in each group that have been generated in Step 0 will become θ'_m's. Or when $m > 0$ θ'_m are derived by applying crossover or mutation on the first four points and the three joining operators of rules in those ones rejected in Step 2 during the previous generation.

Step 2: Generate the random variable $X_{m,l}(\theta)$ by running the pair of rules respectively selected from θ_m and θ'_m sequentially through the entire IV data set. L would be the smaller of the two corresponding total matches for each θ_m and θ'_m. $X_{m,l}(\theta) = 1$ when predict correctly, 0 otherwise. Let $R_m = \sum_{l=1}^{L}(X_{m,l}(\theta_m) - X_{m,l}(\theta'_m))/L$. If $R_m > 0$, then let $\theta_{m+1} = \theta_m$. Otherwise let $\theta_{m+1} = \theta'_m$. Select another pair rules from θ_m and θ'_m and repeat the comparison procedure until obtaining 50 θ_{m+1} rules. 25 of the rejected rules will be used for crossover and the other 25 mutation at Step 1 in the next generation. Repeat the entire process for the rest of the 99 groups.

Step 3: $K_m(\theta)$ would be the total number of matches in the qualifying part of rules θ_m and θ'_m up to generation m, while $\sum_m(\theta)$ is the number of correct predictions for the

corresponding rules. Increase the counter by 1 until reaching the preset limit. At the last generation, optimize among all rules stored in the memory based on the given criteria and retain the top 100 θ^*_m that could best forecast in the given data set, i.e. maximize the percentage of correct forecast by letting $\theta^*_m \in \arg\max_{\theta\in\Upsilon_m} \sum_m(\theta)/K_m(\theta)$, where $\Upsilon_m = \{\theta \in \Theta : K_m(\theta) > 0\}$. In ranking all stored rules, among those rules that are numerically identical, qualified and predicting correctly only the one has minimum "`don't care`" fields and "`OR`" operators will be retained.

At Step 0 generation 0, first rule is generated to take a value of θ_0 and the success rate of prediction to be zero. For whatever value of θ_0 we generate a different rule based on criteria given in Step 1. At Step 1 we would make use of the GA technique such as the tournament/elitist selection criterion to improve the chance of reaching the optimal objective function. Tournament selection is a mechanism for choosing individuals from a population. A group (typically between 2 and 7 individuals) are selected at random from the population and the best (normally only one, but possibly more) is chosen. An elitist GA is one that always retains in the population the best individual found so far. Tournament selection is naturally elitist.

At Step 2, we generate the expected outcome $X_m(\theta)$ for both rules by comparing each rule with all data points in the IV series. In carrying on the same process to the next point in the data set till completion, we find the respective L. For generation m>1, we only need to go through this process for θ'_m while values of $X_m(\theta)$ and L for θ_m have been derived in the previous generation. If θ'_m have higher rates of success, replace the current rules with the more successful ones and keep them in memory as θ_{m+1}. In such an operation, the same θ_{m+1} from different groups could appear more than once as indicated in Step 1, and it will yield the same $X(\theta)$ as before due to the nature of the data set. But only one of them should be registered when they predict better than the current best θ_m. In order to comply with Algorithm 1, we could incorporate a screening mechanism firstly to reject rules that are the same as those currently exist in the memory and secondly to reject rules that are identically qualified and correctly predicting in the current generation. This is necessary because in Andratottir's algorithm, θ_m' which is the same as previous θ's will be rejected in Step 2. This process is repeated in parallel for all 100 groups. At step 3, at the last generation we calculate for the optimal solutions θ^*_{m+1} based on the corresponding number of correct predictions, *i.e.* to determine the rules that maximize the prediction among all retained rules. Once the top 100 rules are derived, we could use them to predict another set of IV data especially those at a subsequent time period in order to confirm the validity of the approach.

C.5 Rate of Convergence

The rate of convergence of the algorithm is the rate at which the distribution of θ_m in Algorithm 1 converges to an optimal distribution, *i.e.* only puts a positive mass on elements of Θ^*. In other words, rate of convergence of a random search method for discrete stochastic optimization is the rate at which the estimated value of the objective function at the estimated optimal solution converges to the optimal values of the objective function. Theorem 1: Rate of Convergence of Random Search Methods [24]. Assume that

- $\Theta^* \neq 0$ and is finite;
- The estimate of the optimal solution $\theta^*_{m+1} \in \arg\max_{\theta\in\Upsilon_m} \sum_m(\theta)/K_m(\theta)$ in Algorithm 2 converges almost surely to the set Θ^*as $m \to \infty$. Since $\sum_m(\theta)$ is the number of correct predictions while $K_m(\theta)$ is the number of hits, *i.e.* the number of matches between the first four points of the rule and the 4-lag recursive points in the IV data set, as $m \to \infty$, $K_m(\theta) \to \infty$. From the Strong Law of Large Numbers, consistent and unbiased solutions exist [24];
- For all $\theta \in \Theta^*$, the estimate of $f(\theta)$ (obtained from single trials, i.e. at a certain value of m) are independent and identically distributed with mean $-\infty < f(\theta) < \infty$ and variance $0<\sigma^2<\infty$; If Θ is finite and for all $\theta \in \Theta$ we have $|f(\theta)| < \infty$, the estimates of $f(\theta)$ here are IID. Since the rules are initially randomly generated, and each rule is independent of each other; rules after randomly crossover and mutated are also independent. Moreover, they are generated in a similarly random fashion therefore, it is understandable that the rate of correct prediction for all rules at each iteration is IID.
- The estimates $f(\theta)$ are independent of the estimates of $f(\theta')$ for all $\theta' \in \Theta^*\backslash\{\theta\}$ (when each estimate is obtained from a single trial); and
- there exists a constant $0<c(\theta)<\infty$ and a sequence $\{a_m\}$ of constants such that as $m \to \infty$, $a_m \to \infty$ and $K_m(\theta)/a_m \to^P c(\theta)$. (i.e. $K_m(\theta)$ can be tracked so that it is feasible for each θ to have a distinguishable value of $K_m(\theta)$.) We then have

$$\sqrt{a_m}\left(\frac{\sum_m\left(\theta^*_{m+1}\right)}{K_m\left(\theta^*_{m+1}\right)} - \min_{\theta\in\Theta} f(\theta)\right) \Rightarrow \min_{\theta\in\Theta^*} Z(\theta) \quad (8)$$

as $m \to \infty$

where $\forall\theta \in \Theta^*$, the random variables $Z(\theta)$ are independent and

$$Z(\theta) \sim N\left(0, \frac{\sigma^2(\theta)}{c(\theta)}\right). \quad (9)$$

D. Limitations of the Proposed Method

Indeed, Andradottir's local and global search methods are based on the assumptions of initially non-homogeneous but asymptotically time homogeneous Markov transition matrix as $m \to \infty$, while other assumptions are easy to satisfy, *i.e.*

irreducible, aperiodic, *etc*. [24]. Such a principle of time averages for non-homogeneous Markov chains may be applicable to our case, because the 100 rules found by GA are derived from matching the 100 most popular patterns in the IV data set through an evolutionary process. The 100 popular patterns may not necessary be at the steady-state because the limited number of generations and size of the available data set. In other words, the required conditions for applying Andradottir's method are stronger than what we actually possess. However, Andradottir's approach provides at least a basic mathematical foundation for further development.

The second limitation is closely related with the first one. Andradottir assumed that unbiased estimates of the objective function values are available. In particular, if $X_1(\theta),\ldots,X_L(\theta)$ are IID observations of $X(\theta)$ for all $\theta \in \Theta$, then $\sum_{l=1}^{L} X_l(\theta)/L$ is an unbiased estimate of $f(\theta)$ for all $L \in \mathrm{N}$ and $\theta \in \Theta$. And at current time we will need to accept such an assumption prior to the GA operation with a limited data set, which leads to limited m.

E. Potential Applications of volatility Forecasting

As detailed in [6], the current GA approach provides the flexibility and guideline in determining the forecasting horizon based on the values of entropy of the respective wavelet coefficients by incorporating the wavelet transform. More details could be found in Section III.*A*. This will be more reasonable than arbitrarily select the forecasting horizon based on users' experience or requirement in other prevalent approaches. The method's concentration on abrupt changes and the use of 4-lags does not limit it from the analysis of longer term volatility activities. The current 4-lag could be configured to deal with hourly, weekly or even monthly data. Moreover, 5-lag, 6-lag or more could be used to account for data in longer time horizons. With an improved prediction of volatility, we could proceed to trade volatility itself. One way to do so would be to use the newly formed VIX Futures in the CFE (Chicago Futures Exchange), namely VXB.

III. Implementing the Approach

A. Data Pre-processing

The original time series such as the S&P100 index is first converted into the IV data and then transformed into wavelet coefficients. One advantage of combining wavelet analysis with EA is the flexibility that they bring in. By selecting the corresponding wavelet coefficient series, the current 4-lag configuration could help focus on different time ranges depending on nodes on the wavelet tree as given by the following expression:

$$\frac{f_{j,n}}{\Delta t} = \left[\frac{n}{2^{j+1}\Delta t}; \frac{n+1}{2^{j+1}\Delta t} \right], \tag{10}$$

where j is the level on the tree, n is the location on the tree, Δt is the sampling period; in the current case, it is one day and $f_{j,n}$ is the nominal frequency band. For example, wavelet packet node (1, 0) gives information up to four days ahead; while (5, 5) is between 11 and 13 days.

For a data set of N samples of IV's, level 2 of the wavelet packets has $N/2$ number of coefficients, representing a saving of 50% calculation for the subsequent GA processing. A maximum of level 5 has been selected as the analysis scale in this paper, because as mentioned earlier reliability of forecasting accuracy drops as time horizon expands. Filters such as the db2 wavelet, which has two vanishing moments, were used for the current analysis in order to maximize the match of the reconstructed data with the original data while retain the minimal amount of data. Note that a wavelet of the Daubechies family with fewer vanishing moments may fail to suppress the higher order polynomial signal. This has been confirmed in analyzing the current S&P100 series when db1, db2 and db3 occasionally fail to generate the wavelet coefficients based on the best tree that is created from the wavelet packet. On the other hand, higher order wavelet tends to generate smoother decomposed plots, which may loose some desirable details from the original series. Different combinations of orders and levels of the db wavelets could be tried to obtain the best tree. Analysis could focus on the node with the highest entropy. For example, db3 with level 5 in the best tree, the highest entropy occurs in packet (4, 0), where j = 4 and n = 0. Packet (4, 0) corresponds to the frequency of $(n + 1)/(2j\,\Delta t) = 1/(32) = 32$ days (in case of j = 1, n = 2, $(2 + 1)/(4 \times 1) \geq 1.3$ days). In general, there are five parameters to be determined before conducting the analysis, *i.e*. number of levels of the wavelet packet tree, order of the filter, number of generations, number of groups of rules and the training period. In this research, the effect of each variable is investigated by holding others constant. Since the main difficulty that contemporary researchers are faced is the forecast of abrupt changes, short term patterns in the volatility are the focus of this paper.

B. Volatility Forecasting with GA

The premise of the GA approach adopted here is originated from Fong and Szeto's method [14]. First, the transformed IV time series in form of wavelet coefficients is classified into four ranges, and generate rules randomly in the form as shown in Section II.*B*. More details about wavelet transform of the data are given in the following section. For the '`THEN`' part of the rule, there are four different classes, 1, 2, 3 and 4. Randomly generate 25 rules for each class to have a total 100 rules. Repeat the process to generate 100 groups of such rules. As a result, the rules would be like $<((\theta_t = 2)\,\mathtt{OR}\,(\theta_{t+1} = 3)\,\mathtt{AND}\,(\theta_{t+2} = 4)\,\mathtt{AND}\,(\theta_{t+3} = 2)) = (\theta_{t+4} = 1)>$.

Fitness value of each rule is calculated as described in the following. In each training step, the rules for class k are trained by comparing the patterns of the randomly generated rules with the S&P100 IV historical data. Three possible cases can arise:

Case 1: the '`IF`' part of the rule does not match the data point pattern. So, no prediction can be made.

Case 2: the '`IF`' part of the rule matches the data points in the training set. Prediction can be made.

When the '`THEN`' part of the rule also matches the class of the corresponding data point, it is counted as a correct guess otherwise a wrong guess. The fitness value of rule i will be

$$F_i = N_c / N_g = N_c / (N_c + N_w). \quad (11)$$

Here N_c is the number of correct guess and N_w is the number of wrong guess, so that

$$N_g = N_c + N_w. \quad (12)$$

Apply each rule to all training data and find the accumulated N_c.

Case 3: there are more than one rule with the '`IF`' part, which matches the data points in the training set. The most specific rule, which does not have "`don't care`" and all logical operators are '`AND`', is chosen.

These rules will be ranked based on their fitness. Repeat these steps sequentially throughout the training data set for other 99 groups. Out of the 50 groups with F_i's below its medium, randomly choose 25 groups for crossover, in which each group goes through the following process:

1) From the pool of 100 rules, randomly select 2 rules to conduct crossover;
2) Register the rules in a memory;
3) From the second round of selection onward, compare selected the rules with those stored in the memory;
4) If both rules have been selected as a pair before, then repeat the selection process till picking a different pair. Repeat the process until forming 100 new rules.

The other 25 groups undergo mutation at a rate of 4%, which means 1% overall in each generation. The same process is repeated for a preset number of generations, *e.g.* 1000. In each generation, only 50% of rules need evaluation of F_i's, thus a 50% of CPU time saving. At the end, the best group of rules is selected for testing of their forecasting accuracy with the subsequent part of the data.

B.1 Data Testing and Results

The intraday data of S&P100 between 1987 and Aug. 2003 is acquired from TickData Inc. Part of the data set θ_t, the 15-minute high-low prices between 1998 and 2002 is taken for training purpose. The second part, *e.g.* between Jan. 2 and Aug. 29, 2003 will be used to test the validity of the rules. The data are imported into a MATLAB environment to calculate the corresponding normalized IV's. The IV values are then wavelet transformed to find the best tree. The GA programs are then applied to forecast the IV values at the selected time ranges ahead.

In applying the GA programs, all rules are initially assigned to have zero fitness. The data range. $(-\infty,-a]$, $(-a, b]$, $(b, c]$, (c, ∞) are preset at $a = -0.3$, $b = 0$ and $c = 0.3$ based on observation of the time series plots as well as analyst's risk requirement. The data is then processed with the GA programs in the Matlab environment and the best group of rules is found. Upon completion of the training process, the best group of rules will be used to test the subsequent part of the S&P100 IV data to assess the forecasting accuracy. To achieve calculation economy, the programs that involve GA are written in Java while the wavelet transformation is done with MATLAB.

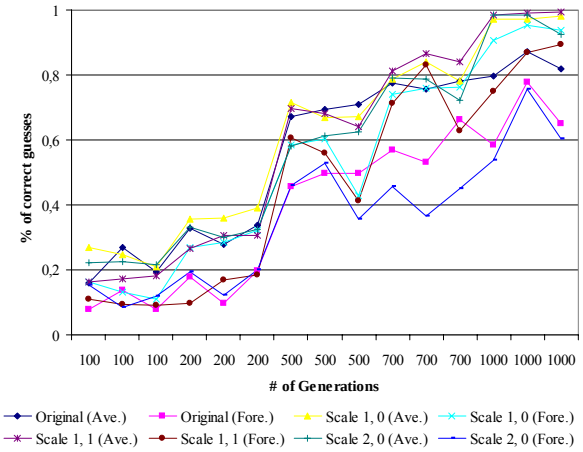

Fig. 2 Daily 2003 S&P100 forecasting accuracy based on 2002 S&P100 data

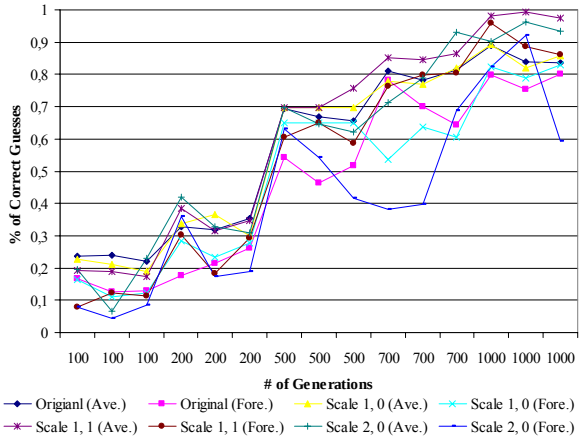

Fig. 3 Daily 2003 S&P100 forecasting accuracy based on 2001 and 2002 S&P100 data

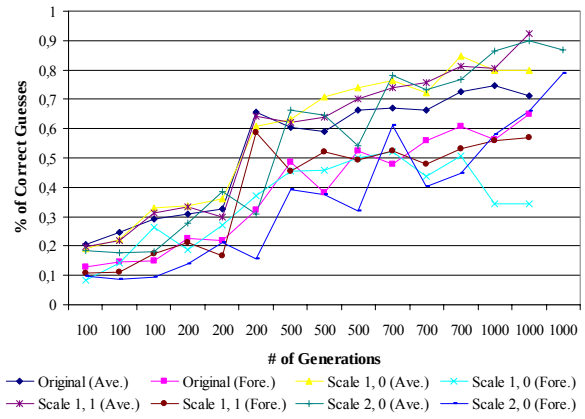

Fig. 4 Hourly 2003 S&P100 forecasting accuracy based on 2002 S&P100 data

It could be observed from Figure 2 and 3, while less obvious in Figure 4 that a) the forecasting accuracy is generally above 60%, which is better than the traditional methods and matches those derived from the proprietary methods [4]; b) the forecasting accuracy is higher for the wavelet transformed series with higher scales compared with those derived on the original series, *i.e.* the non-transformed ones. This may be attributed to the fact that variance of the original series is the sum of variances of its spectral components. The same data in the same one-year, two-year and five-year time horizons as in the current research are processed with the GARCH(1,1) model. The forecasting values from GARCH(1,1) are first normalized to the respective logarithmic means and are then converted into values of 1, 2, 3 and 4 according to their amplitudes, similar to the preprocessing described in the previous sections where GA is used. The accuracy of forecasting is calculated based on the comparison between the converted data and the realized volatility in the validity-testing period, *i.e.* from Jan. 03 to Aug. 29, 2003. The one day ahead forecasting accuracy for the 2003 S&P100 daily data based on training sets at the selected periods are as shown:

TABLE II
ONE DAY AHEAD FORECASTING ACCURACY

Period	% Accuracy
2002	0.485
2001—2002	0.491
1998—2002	0.503

They agree well with the results derived in many contemporary GARCH as well as IV studies [1, 3], but are markedly lower than those achieved by using the GA method proposed in this research. The GA method is superior to the GARCH approach simply because it takes more historical patterns linear or nonlinear, into consideration for forecasting purpose.

C. Volatility Forecast with GP

In this section, we attempt to forecast the numerical values of the volatility by formulating a nonlinear and non parametric approach based on GP in the TSDM framework. Different patterns, linear or non-linear including the stylized clustering effect of volatility may repeat in different time intervals. This is true when dealing with different types of financial securities or dealing with different historical periods for the same underlying security. By making use of the stylized characteristics of financial volatility, we extend the TSDM method with GP to forecast as many events/non-events as practically feasible in the IV time series in order to guide option trading.

C.1 Data Testing and Results

The same intraday data of S&P100 index (OEX) as those used earlier is again applied here. Part of the data set θ_t, the 15-minute high-low prices between Dec. 3, 2001 and Dec. 31, 2002 are taken for training purpose. The second part, *e.g.* between Jan. 2 and Aug. 29, 2003 will be used to test the validity of the rules. The first 21 days of both sets of data are used to prepare for the 21-day moving average, in order to take the monthly effect into consideration, to de-trend and to improve the forecasting accuracy.

The corresponding normalized *IV*'s were then calculated and fed to the GA programs as described in Section III.*A* and *B* in order to find the best 100 rules by maximizing the value *f*. These rules are further processed by GP to find the arithmetic formulae to forecast the *IV* values at the selected time ranges ahead such as one-day-ahead [6]. The execution cycle of the generational GP algorithm includes the following steps:

1. Initialize the population. An initial population of 100 is created randomly from the basic building blocks.
2. Evaluate the individual programs in the existing population. A value for fitness, *e.g.* the absolute difference between the individual and the desired one is assigned to each solution depending on how close it actually is to solving the problem (thus arriving at the answer of the desired problem).
3. Until the new population is fully populated, repeat:
   a. Select an individual or individuals in the population using the selection algorithm
   b. Perform genetic operations (crossover & mutation) on the selected individuals
   c. Insert the result of the genetic operations into the new population.
4. If the termination criterion is fulfilled, then continue. Otherwise, replace the existing population with the new population and repeat steps 2-4
5. Present the best individuals in the population as the output from the algorithm.

Parameters used in the current study are listed in Table III. Note that based on the findings in Neely and Weller's research [10], the fitness of the GP operation in the current investigation is derived from the Mean Absolute Error between the generated individual and the actual IV value. In two separate tests, the 2002 training data set were pre-processed by following the same steps as those in the GA programs, one for 500 generations and the other 1000. The intermediate results are then passed through GP programs and

the final results of percentage accuracy are shown in Table IV.

TABLE III
GP CONFIGURATION

PARAMETER VALUES
Generations: 25/50/100
Populations: 100
Function set: +, –, %, ×, sin, exp, sqrt, ln
Terminal set : {$x(t-1)$, …, $x(t-4)$}
Fitness: difference between obtained values and desired values
Max. depth of new individual: 6
Max. depth of new subtrees for mutation: 6
Max. depth of individuals after crossover: 17
Mutation rate: 0.05
Generation method: 50%

TABLE IV THE FORECASTING ACCURACY FOR 2003 IV DATA BASED ON BOTH 2002 TRAINING DATA SETS.

GP PARAMETERS	2002 DATA SET (500 GENERATIONS OF GA)	2002 DATA SET (1000 GENERATIONS OF GA)
[25, 100, 6]	72.65, 73.21, 74.40	74.23, 66.00, 68.77
[50, 100, 6]	71.46, 75.36, 74.46	76.77, 69.13, 67.54
[100, 100, 6]	71.44, 69.60, 68.42	68.09, 67.49, 67.17

The 2001-2002 training data set was then pre-processed using GA's [6] and 1000 generation GP was implemented to obtain the results as shown in Table V.

TABLE V THE FORECASTING ACCURACY FOR 2003 IV DATA BASED ON 2001/2002 TRAINING DATA SETS.

GP PARAMETERS	2001/2002 DATA (BASED ON 1000 GENERATIONS GA)
[25, 100, 6]	78.25, 77.66, 78.25
[50, 100, 6]	80.20, 79.51, 79.54
[100, 100, 6]	79.10, 78.86, 78.98

For example, an initial population of 100 rules is generated and 25 generations of GP are performed with a maximum depth of six of new individuals.

An interesting phenomenon could be observed that the forecasting accuracy in the current tests is not positively related to the number of generations used in either GA or the subsequent GP operations. This may be caused by the early convergence to the local minima in the search process. Further investigation and appropriate search strategy may be necessary to resolve the issue.

D. Data Analysis with DSOM

In order to verify the GA principles outlined earlier, we analyzed the volatility of S&P500 index based on the algorithms shown in Section II.*C.4*. The data in 2003 is first employed to train the algorithms, while the selected rules are tested with the data in 2004. The same procedure is repeated with the data between 2002 and 2003. Refer to Figure 5 for results.

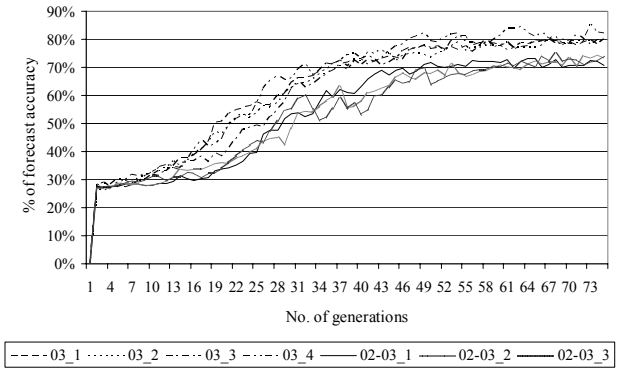

Fig. 5 Daily 2004 S&P500 forecasting accuracy with both 2002/2003 and 2002 data as training data sets

One could observe that the forecasting accuracy ranged between 70 and 80%. Other tests of up to 500 generations have shown that results tend to converge after 75 generations. One advantage of using the *DSOM* algorithms is the fast convergent due mainly to the usage of larger amount of memory instead of looping while implementing GA's. It is interesting however, forecasting accuracy for the 2003 data set is generally higher than those derived based on 2002-2003 data sets. Further tests would be necessary to shed more light into this issue.

IV. Conclusions and Recommendations

The methodology described in this research illustrates a systematic approach to address specifically non linearity problems in the forecast of financial indices. There is no need to have much *a priori* knowledge regarding the nature of the phenomenon, neither parametrically nor stochastically. Yet, highly accurate estimation and even forecasts could be achieved.

The rationale of the forecasting approach is based on the stylized fact of volatility clustering that has been the foundation of the GARCH approach. The key characteristics of the IV–wavelet-EA approach are the use of the EA four-lag recursive data conversion. Within the TSDM framework the four-lag data set acts as the sliding window. And at the same time, such data set is a Markov chain thus, enabling the subsequent use of the Markov-chain-based discrete stochastic optimization process for validation purpose. The current approach helps analysts identify different patterns in the time domain even if those patterns are abrupt jumps or drops. Since it is non parametric and free of any strong pre-assumption, it is more flexible and robust to deal with non-linearity. The wavelet transform enables analysts to study the volatility patterns in different frequencies (time horizons). The combination of these techniques opens up a broader field for analysts to explore different properties hidden in the volatility series. In general, the GA part of the approach is proved to forecast at an accuracy of 75% matching the level achieved by other proprietary methods. These results are further validated by the tests of S&P500 with the discrete stochastic optimization algorithms, in which the forecasting accuracy ranged between 70 and 80%.

The 100 rules found by GA's are however, derived from matching the 100 most popular patterns in the IV data set through an evolutionary process. As such, the 100 popular patterns may not necessarily be at the steady-state because the limited number of generations and size of the available data set. Further proof is needed to generalize Andradottir's method.

Appendix

A group of TSDM definitions are given in this section, *e.g.* events (important occurrences), temporal patterns (vector of length Q), event characterization function ($g(t)$), temporal pattern cluster (P), time-delay embedding, phase space, augmented phase space, and objective function [23].

An event may be defined as the sharp rise or fall of an IV value. Let $\Theta = \{\theta_j, t = 1, \ldots, M\}$ be the daily IV series for the S&P100 index between 2001 and 2002. A temporal pattern is a hidden structure in a time series that is characteristic and predictive of events. The temporal pattern P is a real vector of length Q. And it best characterizes the desired events, *e.g.* P is used to predict events in a testing time series. The temporal pattern is represented as a point in a Q dimensional real metric space, $P \in \Re^Q$. Because a temporal pattern may not perfectly match the time series observations that precede events, a temporal pattern cluster is defined as the set of all points within δ of the temporal pattern.

Let $\tau > 0$ be a positive integer. If t represents the present time index, then $t-\tau$ is a time index in the past, and $t+\tau$ is a time index in the future. A phase space is a Q dimensional metric space into which a time series is embedded. In our case, Q is chosen to be 4, representing 4-lag recursive memory. To link a temporal pattern (past and present) with an event (future) the event characterization function $g(t)$ is introduced. The event characterization function represents the value of future "eventness" for the current time index. One possible event characterization function to address this goal is $g(t) = \theta_{j+1}$, which captures the goal of characterizing IV values one-step in the future. The concept of an augmented phase space follows from the definitions of the event characterization function and the phase space. The augmented phase space is a $Q+1$ dimensional space formed by extending the phase space with $g(.)$ as the extra dimension. Every augmented phase space point is a vector $< \theta_j, g(t) > \in \Re^{Q+1}$.

As shown in Fig. A-1, the height of the leaf represents the significance of $g(.)$ for that time index. From this plot, the required temporal pattern and temporal pattern cluster are easily identified. The TSDM objective function represents the efficacy of a temporal pattern cluster to characterize events. The objective function f maps a temporal pattern cluster P onto the real line, which provides an ordering to temporal pattern clusters according to their ability to characterize events. The objective function is constructed in such a manner that its optimizer P^* meets the TSDM goal. The objective function must capture the accuracy with which a temporal pattern cluster predicts all events. Since it may be impossible for a single temporal pattern cluster to perfectly predict all events, a collection C of temporal pattern clusters is used for this objective function. The objective function $f(C)$ is the ratio of correct predictions to all predictions, *i.e.*

$$f(\mathrm{C}) = \frac{t_p + t_r}{t_p + t_r + f_p + f_r}. \qquad \text{(A-1)}$$

This objective function would be used to achieve maximum event characterization and prediction accuracy for binary $g(t)$. The key concept of the TSDM framework is to find optimal temporal pattern clusters that characterize and predict events. Thus, an optimization algorithm represented by $\max_{P,\delta} f(P)$.

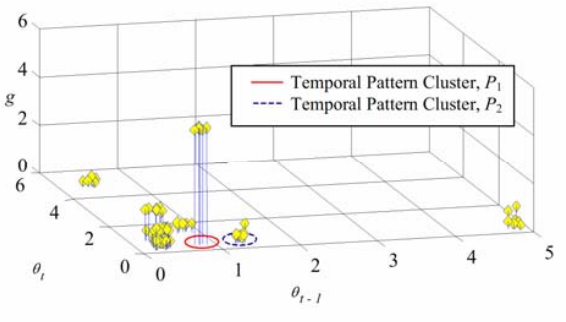

Fig. A-1 Stem-and-leaf plot showing the augmented phase space for a time series

REFERENCES

[1] Andersen, T., Bollerslev, T., Diebol, F. and Labys, P., "Modeling and Forecasting realized volatility ", First Draft: January 1999, This Version: January 2001
[2] Ma, I., Wong, T., Sankar, T., & Siu, R., "Volatility Forecasts of the S&P100 by Evolutionary Programming in a Modified Time Series Data Mining Framework", *WAC* 2004 conference, Seville, Jun. 2004a
[3] Andersen, T. G., T. Bollerslev, F. X. Diebold. and P. Labys. 2001, "The distribution of realized exchange rate volatility," *Journal of the American Statistical Association*, no. 96, pp. 42-55
[4] Kinlay, H. R., J. Neftci and P. S. Wilmott, 2001, "Investment analytics volatility report," *Investment Analytics*, N.Y.
[5] Christoffersen, P. and F. Diebold, 2000, "How relevant is volatility forecasting for financial risk management?," *Review of Economics and Statistics*, vol. 82, pp. 12-22
[6] Ma, I., Wong, T., Sankar, T., & Siu, R., "Forecasting the Volatility of a Financial Index by Wavelet Transform and Evolutionary Algorithm", *IEEE SMC* 2004 Conference, Hague, Oct. 2004
[7] Pictet, O. V., Masutti, O. and Zumbach, G., "Volatility forecasting using genetic programming", *Dynamic Asset Management*, Geneva, Switzerland and Olsen & Associates, Zurich, Switzerland, 2000
[8] Zumbach, G., Pictet, O. V. and Masutti O., "Genetic Programming with Syntactic Restrictions applied to Financial Volatility Forecasting", Olsen & Associates Research Institute for Applied Economics Seefeldstrasse 233, 8008 Zurich, Switzerland, Dynamic Asset Management Chemin des Tulipiers 9, 1208, Geneva, Switzerland, 2001
[9] Chen, Shu-Heng and Yeh, Chia-Hsuan, "Using Genetic Programming to Model Volatility in Financial Time Series: The Cases of Nikkei 225 and S&P 500", AI-Econ Research Group, Dept. of Economics, National Chengchi University, Taipei, Taiwan 1623, *The 4th JAFEE International Conference on Investments and Derivatives (JIC'97)*, Royal Park Hotel, Tokyo, Japan, Jul. 31-Aug. 1, 1997
[10] Neely, C. J. and P. Weller, "Predicting Exchange Rate Volatility: Genetic Programming vs. GARCH and RiskMetrics", Research Department, Federal Reserve Bank of St. Louis and Department of Finance, Henry B. Tippie College of Business, Sept. 2001
[11] Kaboudan, M, "Extended Daily Exchange Rates Forecasts Using Wavelet Temporal Resolutions", *New Mathematics and Natural Computation*, Vol. 1, No. 1, by World Scientific Publishing Co., 2005, p. 79- 107
[12] Lee, W. C., "Forecasting High-Frequency Financial Data Volatility Via Nonparametric Algorithms -Evidence from Taiwan Financial Market", *4th International Conference on Computational Intelligence in Economics and Finance*, Salt Lake City, Utah, Jul. 2005, p. 1031 - 1034
[13] Lawrenz, C. and Westerhoff F., "Explaining Exchange Rate Volatility with a Genetic Algorithm", *6th International conference of the Society for computational Economics on Computing in Economics and Finance*, Barcelona, paper 325, Jul. 2001
[14] Fong, L. Y. and Szeto, K. Y., "Rules Extraction in Short Memory Time Series Using Genetic Algorithms", *The European Physical Journal B,* 20, pp. 569-572, 2001
[15] Maheu, J. and McCurdy, T., "Nonlinear Features of Realized FX Volatility", Department of Economics, University of Alberta and Rotman School of Management, University of Toronto
[16] Gaunersdorfer C. and Hommes Cars, "A Nonlinear Structural Model for Volatility Clustering", Center for Nonlinear Dynamics in Economics and Finance (CeNDEF) Faculteit Economie, Universiteit van Amsterdam, CeNDEF Working Papers, number 00-02, Sept. 2000
[17] Dunis, C. and X. Huang , "Forecasting and Trading Currency Volatility: An Application of Recurrent Neural Regression and Model Combination", *The Journal of Forecasting*, 21, 317–354., 2002
[18] Wang, N. J., K. Wang and J. Jeng, "An Empirical Approach toward Realistic Modeling of Capital Market Volatility", *4th International Conference on Computational Intelligence in Economics and Finance*, Salt Lake City, Utah, Jul. 2005, p. 1052 – 1058
[19] Hovsepian, K, Anselmo P. C., and Mazumdar, S., "Detection and Prediction of Relative Clustered Volatility in Financial Markets", *4th International Conference on Computational Intelligence in Economics and Finance*, Salt Lake City, Utah, Jul. 2005, p. 1052 – 1058
[20] Tino, P., Nikolaev, N. and Yao, X., "Volatility Forecasting with Sparse Bayesian Kernel Models", *4th International Conference on Computational Intelligence in Economics and Finance*, Salt Lake City, Utah, Jul. 2005, p. 1052 - 1058
[21] Gavrishchaka1, G. G, "Boosting Frameworks in Financial Applications: From Volatility Forecasting to Portfolio Strategy Optimization", *4th International Conference on Computational Intelligence in Economics and Finance*, Salt Lake City, Utah, Jul. 2005, p. 1052 - 1058
[22] Povinelli, R. J., 1999, "Time Series Data Mining: Identifying Temporal Patterns for Characterization and Prediction of Time Series Events", Marquette University, Ph. D. dissertation
[23] Povinelli, R. J., "Identifying Temporal Patterns for Characterization and Prediction of Financial Time Series Events," Temporal, Spatial and Spatio-Temporal Data Mining: First International Workshop; revised papers / TSDM2000, 46-61, 2000
[24] Andradottir, S., "Accelerating the Convergence of Random Search Methods for Discrete Stochastic Optimization", *ACM Transactions on Modeling and Computer Simulation*, Vol. 9, No. 4, Oct. 1999
[25] Duan, J. C., Gautier, G. and Simonato, J. G, "A Markov Chain Method for Pricing Contingent Claims", *Stochastic Modeling and Optimization – with applications in queues, finance, and supply chains*, Springer-Verlag, New York, Inc., 2003

Irwin Ma holds a B. Eng. And M. Eng. Degrees from Concordia University and is currently a Ph. D. candidate in École de technologie supérieure of Montréal. His research interest focuses on the financial analysis in the derivative field using different new mathematical and engineering approaches.

Tony K. N. Wong holds a B.Eng. and M.Eng. degrees from École de technologie supérieure in electrical engineering. He received his Ph.D. degree in computer engineering from École Polytechnique de Montréal. Dr. Wong is a professional engineer and chair of the automated manufacturing engineering department, École de technologie supérieure. His current research interests are multi-objective optimization using evolutionary algorithms and its parallel implementations.

Thiagas Sankar Ph.D., P.Eng., F.ASME., F.CSME., F.EIC., F.IEAust. received his Ph.D in Solid Mechanics from University of Waterloo, Canada in 1967 and is a long standing member and Fellow of several engineering societies around the world. He is currently the Distinguished Research Professor at ETS and he is also the Senior Advisor for Corporate R&D programs at KPMG. Professor Sankar's academic career spans over 38 years of contribution to engineering education, administration, advanced research and industrial consulting. He has published over 350 scientific papers in scholarly journals and conferences, and has guided over 50 Masters and 20 Doctoral research projects in wide areas covering mechanical systems, robotics, machine dynamics, manufacturing and production, industrial engineering, aerospace, signal processing, intelligent sensing and MEMS. He is active in professional and learned societies, and was the National President of the Canadian Society for Mechanical Engineering and the International Chair of the Design Engineering Division of the American Society of Mechanical Engineers. He also served on the Accreditation Board of the Canadian Council of Professional Engineers and is a licensed professional engineer in North America and Australia.