GENETIC ALGORITHMS FOR ROBUST OPTIMIZATION IN FINANCIAL APPLICATIONS

Li Lin, Longbing Cao, Chengqi Zhang
1. Faculty of Information Technology, University of Technology Sydney, NSW 2007
Australia
2. Capital Market CRC, Sydney, NSW 2000
Australia
linli@it.uts.edu.au

ABSTRACT

In stock market or other financial market systems, the technical trading rules are used widely to generate buy and sell alert signals. In each rule, there are many parameters. The users often want to get the best signal series from the in-sample sets, (Here, the best means they can get the most profit, return or Sharpe Ratio, etc), but the best one will not be the best in the out-of-sample sets. Sometimes, it does not work any more. In this paper, the authors set the parameters a sub-range value instead of a single value. In the sub-range, every value will give a better prediction in the out-of-sample sets. The improved result is robust and has a better performance in experience.

KEY WORDS
Stock Market Data Mining; Technical Trading Rules; Genetic Algorithms; Robust; Optimization

1. Introduction

Trading models are algorithms proposing trading recommendations for financial assets. In our approach we limit this definition to a set of trading rules based on the history financial data. The financial data, which are typically series of prices, bid, ask, volume, time, date, etc, enter the trading model in the form of indicators corresponding to various kinds of averages. Although the progress has been made in understanding financial markets, there is no definitive prescription on how to build a successful trading model and how to define the indicators. Automatic search and optimization techniques can be considered when addressing this problem. However, optimizing trading models for financial assets without over-fitting is a very difficult task because the scientific understanding of financial markets is still very limited. Over-fitting means building the indicators to fit a set of past data so well that they are no longer of general value: instead of modeling the principles underlying the price movements, they model the specific movements observed during a particular time period. Such a model usually exhibits a different behavior (or may fail to trade successfully at all) when tested out-of-sample. This difficulty is related to the fact that many financial time series do not show stability of their statistical behavior over time especially when they are analyzed intra-daily.

To minimize over-fitting during optimization, the optimization process must include the following important ingredients:

(1) A good measure of the trading model performance,
(2) Indicator evaluation for different time series,
(3) Large data samples,
(4) A robust optimization technique,
(5) Strict testing procedures.

On the stock market, SIRCA [18] offers the detailed real data order book since 1980s. The order book includes almost all the information of trading, such as: date, time, price, volume, ask price, bid price, order-id, etc., thus providing large data samples for developing trading models. It has also produced a good trading model technology that has found applications in successful real-time trading models.

The new element we want to present in this paper is a way to automatically search for improved trading models. Genetic Algorithm (GA) offers a promising approach for addressing such problems. It considers a population of possible solutions to a given problem and evolves it according to mechanisms borrowed from natural genetic evolution: reproduction, crossover, mutation and selection. The criterion for selecting an individual is based on its fitness to the environment, or more precisely, to the quality of the solution it bears. A possible solution is coded as a gene, which is formally the data structure containing the values of the quantities characterizing the solutions.

In the framework of the present application, a gene will contain the indicator parameters, for example time horizons and a weighting function for the past, and also the type of operations used to combine them. The fitness function will be based on the return obtained when following the recommendations of a given trading model.

2. Technical trading rule models

In this section, we review the different ingredients that constitute the basis of a trading model and reformulate them in terms of simple quantities that can be used in conjunction with a genetic algorithm. We first need to specify an appropriate universe of trading rules from which the current GA may have been applied to. Real trading models can be quite complicated and may require many different rules that also depend on the models own trading history. Here we limit ourselves to simple models that depend essentially on a set of indicators that are pure functions of the price history or of the current return.

E.g. Moving Average rules (MA). [10]

Define the following parameters for a simple Moving average rule:

Short run (term/day): *sr*

Long run (term/day): *lr*

Short run average: *s*, is the average price of the *sr* days trading price;

Long run average: *l*, is the average price of the *lr* days trading price;

Fix band: *x*.

Generating a buy alert signal, when *sr-lr*>*x*;

Generating a sell alert signal, when *lr-sr*>*x*; (see Figure 1)

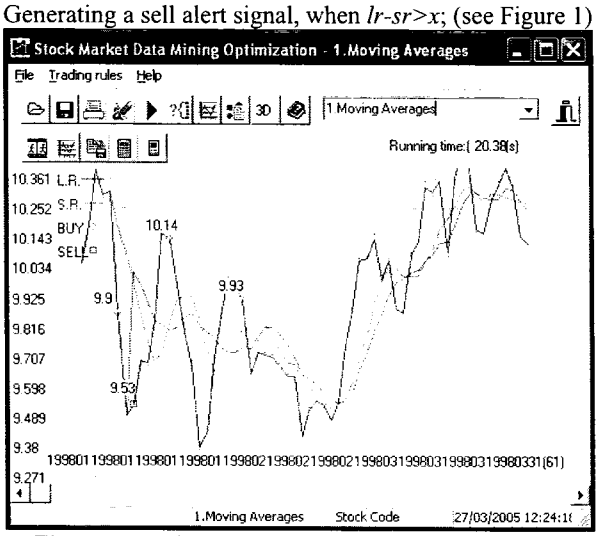

Figure 1 Moving Average trading rule.

About the data set, we divide them into two parts: in-sample set (training set) and out-of-sample set (testing set). In training set, we find the robust parameters and we use the same parameters in the testing set to evaluate the result. In this paper, we use a one-year-long data as training set and continued one-year-long data as testing set.

We have tested four trading rules: Filter Rules, Moving Average (MA), Support and Resistance and Channel Break-outs. [10]

The evaluation used in this paper is Sharpe Ratio [16], which is defined by $(R_p - R_f)/\sigma_p$, where R_p is Expected portfolio return, R_f is Risk free rate and σ_p is portfolio standard deviation.

The reason we use Sharpe Ratio is because it considers both return and risk at the same time, and more and more researchers and traders are considering it.

3. Genetic Algorithms

Genetic Algorithm [3-9] is a heuristic function for optimization, where the extreme of the function (i.e., minimal or maximal) cannot be established analytically. A population of potential solutions is refined iteratively by employing a strategy inspired by Darwinist evolution or natural selection. Genetic Algorithms promote "survival of the fittest". This type of heuristic has been applied in many different fields, including construction of neural networks and finance.

We represented the parameters of a trading rule with a one-dimension vector that is called "chromosome", each element is called a "gene", and all of the "chromosomes" are called "population". Here, each gene stands for a parameter value; each chromosome is the set of parameters of one trading rule.

Generally, genetic operations include: "crossover", "mutation" and "selection".

"Crossover" operator. Suppose $S_1=\{s_{11}, s_{12}, \ldots, s_{1n}\}$, $S_2=\{s_{21}, s_{22}, \ldots, s_{2n}\}$, are two chromosomes, select a random integer number $0 \le r \le n$, S_3, S_4 are offspring of crossover(S_1, S_2),

$S_3=\{s_i \mid \text{if } i \le r, s_i \in S_1, \text{else } s_i \in S_2\}$,

$S_4=\{s_i \mid \text{if } i \le r, s_i \in S_2, \text{else } s_i \in S_1\}$.

"Mutation" operator. Suppose a chromosome $S_1=\{s_{11}, s_{12}, \ldots, s_{1n}\}$, select a random integer number $0 \le r \le n$, S_3 is a mutation of S_1,

$S_3=\{s_i \mid \text{if } i \ne r, s_i=s_{1i}, \text{else } s_i=\text{random}(s_{1i})\}$.

"Selection" operator. Suppose there are *m* individuals, we select [*m*/2] individuals but erase the others, the ones we selected are "more fitness" that means their profits are greater.

Algorithm 1. Genetic Algorithm.

Step 1. InitializePopulation: Producing a number of individuals randomly, each individual is a chromosome which is an *n*-length array, *n* is the number of parameters.

Step 2. Test if one of the stopping criteria (running time, fitness, generations, etc) holds. If yes, stop the genetic procedure.

Step 3. Selection: Select the better chromosomes. It means the profit under these parameters is greater.

Step 4. Applying the genetic operators: such as "crossover" and "mutation" to the selected parents to generate an offspring.

Step 5. Recombine the offspring and current population to form a new population with "selection" operator.

Step 6. Repeat steps 2-5.

GA is also shown as following:

P ← InitializePopulation();

While (not stop (P)) do
    Parents[1..2] ← SelectParents(P);
    Offspring[1] ← Crossover(Parents[1]);
    Offspring[2] ← Mutation(Parents[2]);
        P ← Selection(P, Parents[1..2], Offspring [1..2]);
Endwhile.

For finding the robust result, we add a soft filter onto the GA algorithm to remove the single peak points. (See Algorithm 2) For each point, we compute the values of its neighborhood points, if the values are far from the central point, or negative, we discard it. While we finding the most fitness one, we also consider its neighborhood points. If there is a small range, in which all the value can make more profit, this small range is "robust".

Algorithm 2. Finding the best sub-range algorithm.

Step 1. For every parameter, set an initial size s_1, and step t for the first parameter sub-range;

Step 2. Computing the Sharpe Ratio with GA in every sub-range combination;

Step 3. If there is the best robust sub-range, in which all the values are positive and better than in the others, then output the sub-range and finish the algorithm; else running step 4.

Step 4. Reset another sub-range size $s_2 = \frac{1}{2} s_1$, repeat steps 2 and 3.

If $s_2 = t$, (in every sub-range, there is the least size, generally only one value.) the algorithm becomes the ordinary algorithm.

We can use the Algorithm 1 to find the robust optimized point for the trading rules.

4. Experiments

We show the simple GA and Robust GA results in the following figures. The data is the real stock price from ASX (Australian Stock Exchange) and the in-sample data is one-year from 01/01/2000 to 31/12/2000, out-of-sample data is one year continuously after in-sample data.

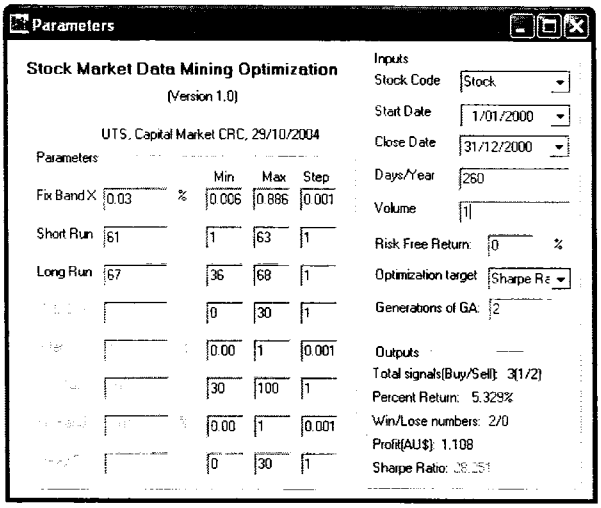

Figure 2 Before optimization, the algorithm gets the best Sharpe Ratio, the result is 28.251. The normal value is between -4 to +4, obviously, the value 28.251 is a noisy, but it is really the "best" one from mathematical definition.

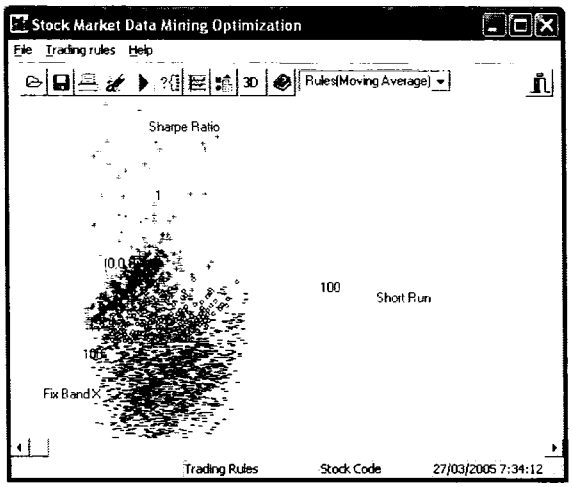

Figure 3 The result before optimization. The best points are "discrete" so we can not find a better range. ("+": the positive value; "o": zeroed value; "-" : negative value)

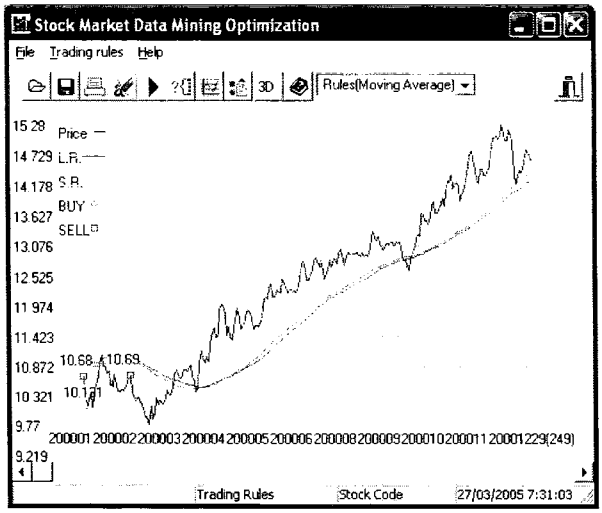

Figure 4 Before optimization results, the system generated three signals only although its Sharpe Ratio is the best, it is not reasonable.

The above figures 2, 3 and 4 show the results before optimization. They are the best result in Mathematical computation. But they do not make sense from financial definitions and applications. (The average value is far from the original data, it can not be used as prediction) And some users often want to change some value a little according to their experience. But these algorithms do not give them any suggestions and ranges. So they have to test randomly by themselves.

The following algorithm after optimization will give the best parameter combination and meanwhile the sub-range, in which any value can make a positive profit. Moreover, the experienced users can choose some value from the sub-range.

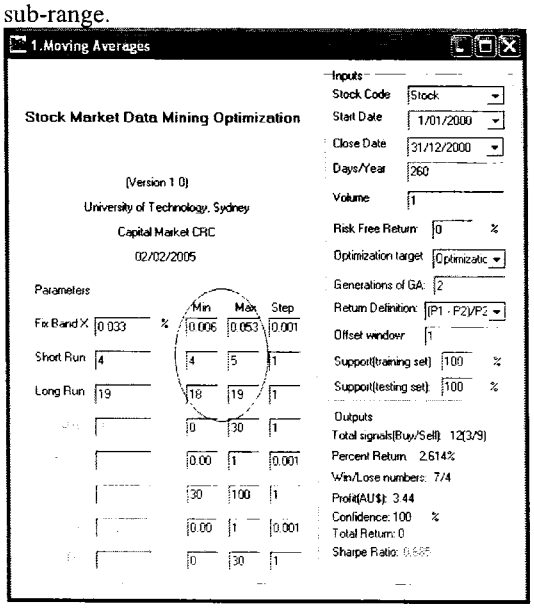

Figure 5 The robust parameters combination and range. (Fix Band range is [0.006, 0.053], Short run is [4, 5] and Long run is [18, 19].)

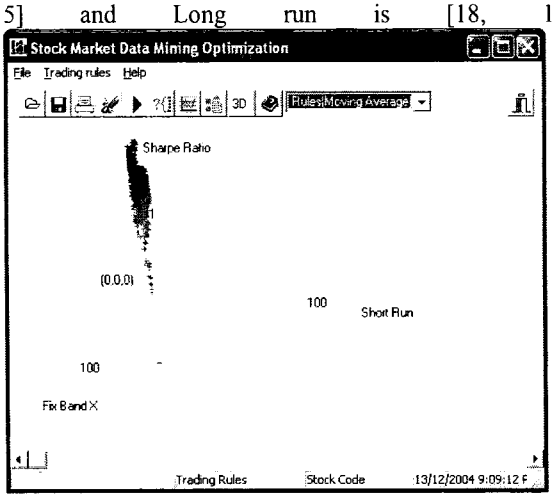

Figure 6 The optimization sub-range, in which, all parameter combination can give a positive Sharpe Ratio.

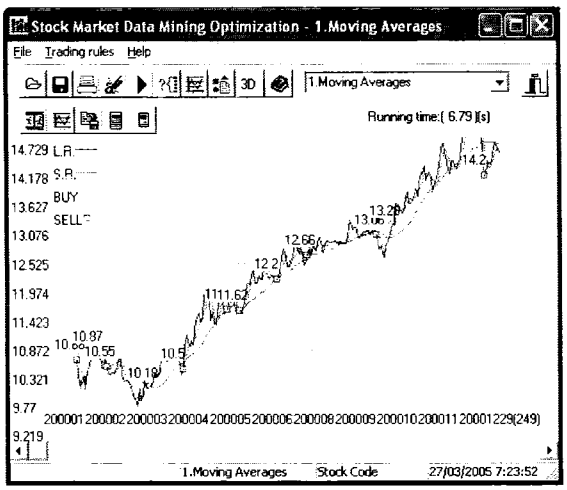

Figure 7 After optimization, the system generated the alert signals.

In these figures, we find the "best" individual is not really the reasonable value. But, the robust algorithms can get a better result than before optimization. We know, the trading rule features, the reasonable value in training sets will be reasonable in testing set, too.[10] And we also do some experiments to prove it. [9]

5. Conclusions

In every stock market analyzing systems, the most important task is giving some buy/sell alert signals to the users. The traditional methods to do this are technical analysis methods, which do the prediction according to the historical data and generate one "best" result. But, in the historical data, there maybe some noisy, random, occasional trends and signals during the trading period. So the best one may mislead the analyzing system to get some wrong information or alert sometimes. (See Figure 4)

In this paper, we have presented an optimization method to get a robust result, which is a sub-range instead of only one value for every parameter.(See Figure 5) In this sub-range, there maybe no noisy disturbance although it is not the best point sometimes. In this sub-range, any point can give a positive profit. (See Figure 6)

The sub-range can be gotten in a very short time with GA, only 10 to 20 seconds, which is acceptable in a real time trading system.

Another advantage of the new algorithm is the users can also change a little of the parameter value by their own experience in the sub-range. They can still get a positive profit in the out-of-sample data. This idea is being accepted by more and more field experts and trading systems.

In the next step, we want to improve the algorithms to get the result more quickly and exactly when the data is huge or the range is more than one. And we will also consider optimizing the size of training set and testing set. The new algorithm can find the best size automatically for different data and different stocks.

Acknowledgments

The authors would like thank Dr. SC Zhang, JQ Wang, WL Chen and JR Ni for their helpful discussion, and Capital Market CRC[17], SIRCA[18] and AC3[19] for funding this project and providing large realistic data of ASX for this research. All supports and feedbacks from finance Prof. Mike Aitken in UNSW, Prof. Alex Frino in Sydney University, Australia and others are much appreciated.

References:

[1] M. Dacorogna, A. Muller, et al, A geographical model for the daily and weekly seasonal volatility in the FX market, *Journal of International Money and Finance*, *12*(4), 1993, 413-438.

[2] R. GenÁay, G. Ballocchi, M. Dacorogna, et al, Real-time trading models for foreign exchange rates, International Economic Review, May 2002, (43) no. 2, pp. 463-492.

[3] A. Franklin, K. Risto Using genetic algorithms to find technical trading rules, *Working Paper*. The Rodney L. White Centre for Financial Research, The Wharton School, University of Pennsylvania, 1993. 20-93, 1-17.

[4] X. Yin., N. Germay, A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization, *Proceedings of International Conference on Artificial Neural Nets and Genetic Algorithms*, Innsbruck, Austria, 1993. 450-457.

[5] S.H. Chen, Genetic algorithms and genetic programming in computational finance (Boston, MA: Kluwer. 2002.)

[6] D. Robert, N. Christopher J., W. Paul, Dittmar, R., Is technical analysis in the foreign exchange market profitable? A genetic programming approach. *Journal of Financial and Quantitative Analysis, 32*, 1997, 405-26.

[7] J. Thomas, K. Sycara, The importance of simplicity and validation in genetic programming for data mining in financial data. *Proceedings of the Joint AAAI-1999 and GECCO-1999 Workshop on Data Mining with Evolutionary Algorithms*. 1999.

[8] A. Franklin, K. Risto, Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*. 1999. 51:245-271. Studies 27:221-334.

[9] L. Lin, L.B. Cao, et al, The applications of genetic algorithms in stock market data mining optimization. *The 5th International Conference on Data Mining, Text Mining and their Business Applications*, Malaga, Spain. 2004. 273-280.

[10] R. Sullivan, T. Allan, H. White, Data-snooping, technical trading rule performance, and the bootstrap. *The Journal of Financial*, 1999. 54, (5):1647-1692.

[11] M. Mitchell, *An introduction to genetic algorithms* (MIT press, Cambridge, USA. 1996.)

[12] D. Whitley, An overview of evolutionary algorithm: practical issues and common pitfalls. *Information and Software Technology*. 2001.43: 817-831.

[13] M.K.P. So, K. Lam, W.K. Li, Forecasting exchange rate volatility using autoregressive random variance model. *Applied Financial Economics*, 1999. 9: 583-591.

[14] K. Lam, K.C. Lam, Forecasting for the generation of trading signals in financial markets. *Journal of Forecasting*, 2000, 19: 39-52.

[15] R. Edwards, J. Magee, *Technical analysis of stock trends* (Seventh edition, New York, 1997. page 4.)

[16] http://www.investopedia.com, 2005

[17] http://www.cmcrc.com, 2005

[18] http://www.sicra.org.au, 2005

[19] http://www.ac3.com, 2005