Global Advanced Research Journals

Global Advanced Research Journal of Engineering, Technology and Innovation (ISSN: 2315-5124) Vol. 3(9) pp. 248-255, December, 2014
Available online http://garj.org/garjeti/index.htm

Full Length Research Paper

A genetic algorithm for time-series optimization

Andrius Paukste

PhD Student at Vilnius University
Email: Andrius.Paukste@mif.stud.vu.lt

Accepted 10 September 2014

In this paper, we introduce novel genetic algorithm improvements applied to time-series optimization. We modify the classic genetic algorithm by introducing mutation and crossover operation modifications with the goal to solve premature convergence problem and achieve better optimization results. We propose few modification versions and develop simulation framework to select appropriate one. Evaluation results indicate that the modified genetic algorithm performs better than classic approach in empirical tests both on in-sample and out-of sample testing periods.

Keywords: genetic algorithms, optimization, time-series, decision support system

INTRODUCTION

Optimization in Investment Decision Support Systems

The domain of optimization in investment decision support systems has received an increasing attention from researchers. The investment rule selection process based on technical analysis creates an optimization problem with high-dimensional, large search space. Optimization space becomes even more complex in case of portfolio of decision support systems optimization. The goal of the optimization is to identify the most robust patterns and generate investment rules or trading strategies. In [2] authors use genetic algorithms (GAs) to discover profitable technical trading rules from technical indicators. Other researchers proposed combinations of few technical rules. [3] Offer to assign a weight to each technical rule and make decisions based on the weighted sum. [4] Propose the extension of GA by exhaustive local search to find profitable combinations of technical analysis.

Genetic Algorithm–Based Optimization and Premature Convergence Problem

Genetic algorithms (Goldberg, 1989) are nature-inspired search algorithms for solving complex, high-dimensional problems in many scientific and engineering domains (Holland and John, 1975). Despite that GA’s are effective at optimizing high-dimensional parameter space, they often suffer from premature convergence problem (Sivanandam and Deepa, 2008). The cause of the problem is decreasing genetic diversity of the GA's population during optimization steps. If the diversity of the population becomes too homogeneous GA will explore only limited part of the population and potentially will stick to local optimum. Few researchers tried to improve GA to avoid premature convergence (Andre et al., 2001), (McGinley and Brian, 2011).

Many engineering or scientific optimization problems solved with GA suffer from premature convergence problem. However, in case of time-series optimization, we are able to find a natural solution. Often the only measure of the population diversity is genotypic diversity. In (McGinley and Brian, 2011) genotypic diversity is calculated by summing the gene-wise Euclidean distances from the average individual to each other individuals. Applications of GA's to time-series optimization problems have one important difference from other engineering or scientific problems. We are able to measure population diversity not only based on gene values but also based on the correlation of the time-series besides the chromosomes. This solution first appears in (Yan and Clack, 2009). The authors proposed the following methods to increase population diversity. First, divide the population into segments by fitness value, and then measure the average performance output and compare the correlation of each individual with the average performance output. Individuals with high correlation will be deleted. The methods proposed in this paper are potentially more precise. However, they require more computation time.

In this paper, we propose modified mutation and crossover operations to get better optimization results by increasing population diversity.

This paper is structured as follows: Section 2 describes the optimization problem and GA. Section 3 proposes GA modifications to reduce population diversity and improve optimization results. Section 4 describes the simulation framework and experiments. Section 5 discusses the results of the performed experiments. Finally, Section 6 concludes the paper.

Genetic Algorithm and Fitness Function

Fitness Function

We define fitness function *f* representing a systematic trading strategy with Sharpe ratio [9], PnL, or other measures as a return:

$r = f(p1, p2, \dots , pN)$,

where *p*1, *p*2, … , *pN* are the strategies parameters in the predefined ranges.

The optimization goal is to find best parameters *p*1, *p*2, … , *pN* in order to get higher function *f* return.

We define historical performance vector *hv*:

$hv = (h_1, h_2 \dots h_{TN})$

where h_i is the performance of fitness function at time t_i and *i* is in ranges $[T_1 \dots T_{TN}]$. T_1 and T_{TN} represent the beginning and the end time of the trading period.

Genetic Algorithm

We use the following modification of simple GA (SGA) (Goldberg, 1989) as a starting point before introducing algorithm improvements:

1. Generate random initial population of K chromosomes. Each chromosome c_i represents the vector of function *f* parameters.
2. Evaluate fitness function. During the evaluation, we keep performance vector hv_k for every chromosome *k*.
3. Perform selection procedure using fitness-proportionate selection.
4. Perform crossover procedure. We use one-point crossover by selecting crossover point randomly.
5. Perform mutation procedure. We mutate each gene independently with a predefined probability.
6. Continue to the step 2 *L* times.

Genetic Algorithm Improvements

In this paper we propose a modification of SGA: Modified Genetic Algorithm (MGA). Following sections describe modifications of mutation and crossover operations of SGA with aim to solve premature convergence problem in time-series optimisation.

First, we define cr_{ij} as a correlation between two chromosomes c_i and c_j performance vectors hv_i and hv_j. *W*e compute correlation as a Pearson's linear correlation coefficient.

Mutation

We propose the following mutation operation improvements. We do additional mutation operations on chromosomes when the correlation between two chromosomes or the mean of the correlations between a chromosome and all other chromosomes is high enough. In addition, we keep in mind the fitness of the chromosome. From two chromosomes with high correlation, we choose one with lower fitness value. Finally, we introduce the measure to evaluate the correlation relative to the fitness value.

To ensure population diversity, we choose chromosomes to perform additional mutation operations based on one of the three measures:

1. Highest correlation with any other chromosome in the population
2. Average correlation with all other chromosomes
3. Highest correlation with any other chromosome in the population divided by the normalized fitness value

Based on each measure above we propose three different mutation procedure modifications. We start from calculation of the correlation cr_{ij} for each pair of the chromosomes c_i and c_j in the population, where *i* is not equal to *j.*

In the first mutation procedure modification we calculate the highest correlation hcr_i for each

Table 1. Data information

Stock number	10
Optimization period	2011-07-01 to 2012-06-31
Out-of-sample testing period	2012-07-01 to 2012-12-31
Data frequency	1 minute
Total number of minutes	210382
Every row consist from	Open, High, Low and Close

Table 2. In-sample results, 1st mutation procedure modification

α_1 values	0,65	0,7	0,75	0,8	0,85	0,9	0,95	0,97	0,99	1
Average sharpe ratio	2,5089	2,5327	2,5819	2,5554	2,5931	2,6546	2,6912	2,6602	2,7301	2,6849

Table 3. In-sample results, 2nd mutation procedure modification

α_2 values	0,65	0,7	0,75	0,8	0,85	0,9	0,95	0,97	1
Average sharpe ratio	2,4939	2,4798	2,6571	2,5994	2,5745	2,6235	2,4798	2,5546	2,4158

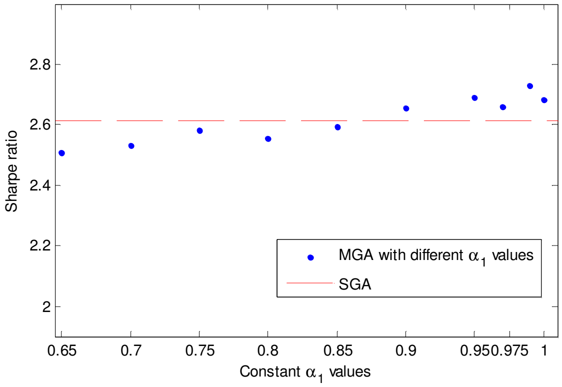

Figure 1. In-sample results, 1st mutation procedure modification

chromosome:

$hcr_i = max(cr_{i1}, cr_{i2}, \ldots, cr_{iN})$, where c_{ij}, where $i \neq j$

Then for each pair of the chromosomes where $cr_{ij} = hcr_{ij}$ and $hcr_{ij} > \alpha_1$, we perform an additional mutation procedure for a chromosome with lower fitness.

In the second mutation procedure modification, we calculate the average correlation acr_i for each chromosome c_i with all other chromosomes in the population:

$acr_i = mean(c_{i1}, c_{i2}, \ldots, c_{iN})$, where $i \neq j$

Then for each chromosome with $acr_i > \alpha_2$, we perform an additional mutation procedure.

In the third mutation procedure modification, we calculate a measure of correlation and fitness value fc_i for each chromosome:

$fc_i = acr_i/rn_i$,

where rn_i is the normalized value of the fitness function. We compute rn_i by rescaling fitness values to ranges from 0 to 1. Then for each chromosome with $fc_i > \alpha_3$, we perform an additional mutation procedure.

Constants α_1, α_2, and α_3 are predefined and will be determined by empirical tests later.

Crossover

We propose the following crossover operation improvements. After the selection procedure, we compute correlation cr_{ij} between every chromosome pair selected for crossover. If correlation $cr_{ij} > \beta$, we omit selected chromosomes and repeat the selection procedure to obtain a new pair of chromosomes.

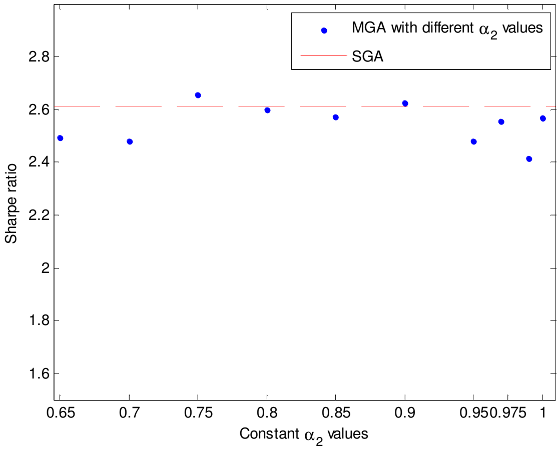

Fig. 2. In-sample results, 2nd mutation procedure modification

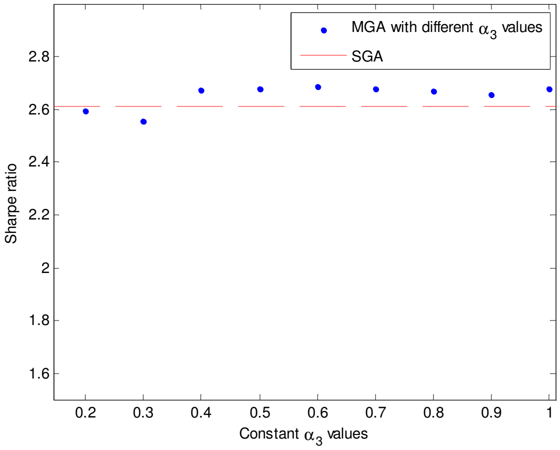

Figure 3. In-sample results, 3rd mutation procedure modification

Table 4. In-sample results, 3rd mutation procedure modification

α_3 values	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Average sharpe ratio	2,5937	2,5541	2,6746	2,6798	2,6871	2,6789	2,6682	2,6537	2,6768

Table 5. In-sample results, crossover procedure modification

β values	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1
Average sharpe ratio	2,6481	2,6915	2,6974	2,6698	2,6972	2,6928	2,6869	2,6847	2,6799

Constant β is predefined and will be determined by empirical tests later.

Impact of α_1, α_2, α_3, and β Constants

In case α_1 and α_2 are equal to 1, we mutate only chromosomes with completely identical behavior (when correlation is equal to 1). When $\alpha_1 < 1$ or $\alpha_2 < 1$, we also mutate chromosomes with similar behavior. Smaller α_1 and α_2 means mutation of less similar chromosomes.

Constant α_3 has a more complicated impact because of division by normalized fitness function value.

Constant β has a similar behavior with α_1 and α_2 constants. When β is equal to 1, we skip only chromosomes with a completely identical behavior. When

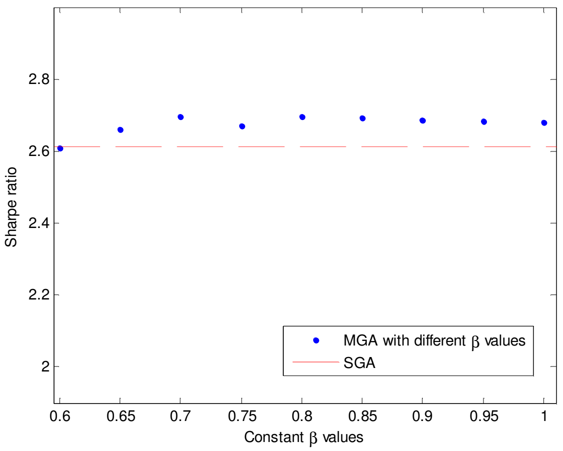

Figure 4. In-sample results, crossover procedure modification

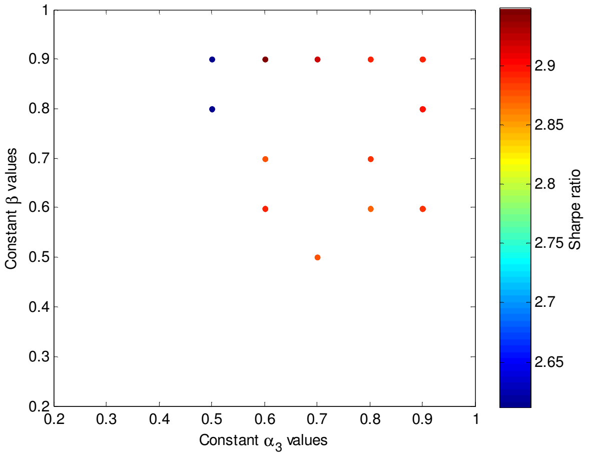

Figure 5. In-sample results, 3rd mutation and crossover procedures modification

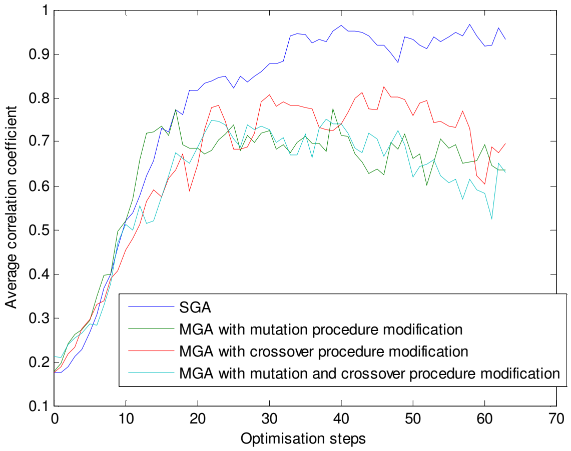

Figure 6. Average correlation coefficients during optimization steps.

Table 6. Combination of α_3 and β values with best fitness results.

β values	α₃ values	Sharpe ratio
0,8	0,5	2,612
0,9	0,5	2,615184
0,6	0,6	2,892905
0,7	0,6	2,878487
0,9	0,6	2,948175
0,5	0,7	2,875483
0,9	0,7	2,917236
0,6	0,8	2,871998
0,7	0,8	2,889901
0,9	0,8	2,895398
0,6	0,9	2,885696
0,8	0,9	2,89711
0,9	0,9	2,893806

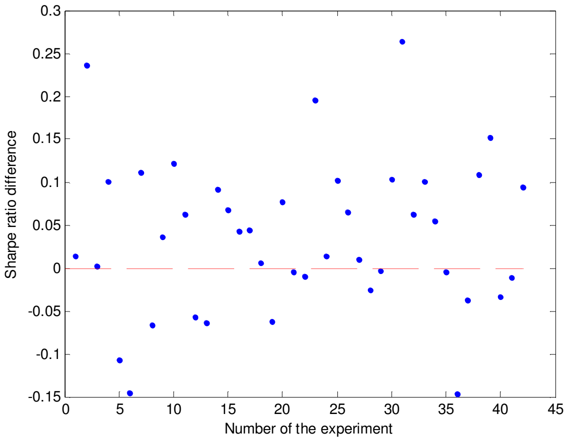

Figure 7. Out-of-sample results, 3rd mutation and crossover procedures modification

β is smaller than 1, we also skip chromosomes with a similar but not identical behavior.

Simulation Framework

Genetic algorithm optimization is based on random numbers, and every optimization result may be different from each other. To better understand the influence of the proposed algorithms, we repeat every optimization 100 times. To reduce computation time, we use the OpenMP (Open Multi-Processing) Application Program Interface (Chandra et al., 2001).

For simulation, we empirically choose 10 stocks from most traded stocks at the New York Stock Exchange: AAPL, INTC, BP, C, F, BAC, GOOG, IBM, GE, PFE.

Tested Strategies

We use the following algorithmic trading strategies based on classic technical analysis (Lo et al., 2000) to test the proposed GA improvements.

Stochastic Oscillator and Two Moving Averages.

We use a combination of two technical analysis indicators: stochastic oscillator and simple moving average. Stochastic oscillator compares the stock's closing price to its price range over a certain period of time. Simple moving average is unweighted mean of previous *k* price changes.

Number of the fitness function parameters: 7

Number of unique parameter vectors: $100^3 + 1000^4$

Two Exponential Moving Averages

A classic example of usage of the technical indicators is strategy with two moving averages with Stop Loss and Take Profit orders.

Number of the fitness function parameters: 4

Number of unique parameter vectors: 200^4

Trend Following Strategy

We use a strategy that takes an opposite direction in trades than price movement.
Number of the fitness function parameters: 4
Number of unique parameter vectors: 400^4

Data

The data used in this study were collected from TradeStation (www.tradestation.com). We use one-minute data in the period from July 1, 2011, to December 31, 2012. The data consists of open, high, low, and close prices for every minute with positive volume. Before optimization, we preprocess the data to exclude non-trading hours.

Data preparation

Noise

Experiment Results

The aim of the experiments is to find answers to the following questions:

1. What are the optimal values of the α_1, $\alpha_{2,}$ α_3, and β constants?
2. How do proposed GA improvements are able to solve premature convergence problem and increase population diversity in latest optimization steps?
3. Do proposed algorithms provide better in sample optimization results?
4. How do optimization results reflect out-of-sample simulation results?

Mutation Operator Impact

Figures 1-3 and Tables 2-4 show impact of constants α_1, α_2, α_3 on in-sample optimization performance. Dashed red line show SGA without modification performance. As we may see GA with proposed mutation procedure improvement achieves better results of in-sample simulation than SGA with α_1 >0.85, $\alpha_{2,}$ equal to 0.75 and 0.9 and α_3 > 0.3.

Crossover Operator

As we may see from Figure 4, GA with the proposed crossover procedure improvement also achieve better results of in-sample optimisation than SGA with constant β > 0.6.

Combination of Crossover and Mutation Operators

Figure 5 and Table 6 show the combination of 3rd mutation procedure modification with crossover procedure modification. Only combinations with better than SGA performance are shown. As we may see MGA with the proposed modified crossover and mutation procedure achieves better results of in-sample optimisation than MGA with modified only mutation or crossover procedures.

Population diversity

Figure 6 show the average population correlation with different GA modifications during genetic optimization steps. As we may see proposed methods significantly decrease correlation in latest optimization steps and solve premature convergence problem.

Out-of-Sample Results

To understand the impact of the proposed methods on out-of-sample testing period we repeat the experiments we did in section 5.3 with out-of-sample data. We measure effectiveness of MGA as average Sharpe ratio of MGA minus average Sharpe ratio of SGA. Figure 7 show that more than half of performed experiments produce results better that SGA on out of sample data.

CONCLUSIONS

An SGA modifications applied to time-series optimization has been presented. We proposed two main improvements with aim to solve premature convergence problem. First, we introduced additional mutation of highly correlated and lower-fitness individuals. Second, we proposed an advanced crossover operation based on correlation between two selected individuals.

We demonstrated the efficiency of the proposed methods based on experimental results. Performed simulations show significant improvements of in-sample optimization results compared with SGA.

The proposed methods have been successfully applied to reduce average correlation in the population and increase population diversity. We conclude that the most dominant factor improving optimization accuracy is modified crossover operator, decreasing the selection of highly correlated individuals.

Finally, the achieved experimental results show that the proposed approach outperforms SGA-based created trading rules in more than half out-of-sample simulations.

The proposed methodology may be extended and applied to various time-series optimization problems.

REFERENCES

Goldberg D (1989). Genetic Algorithms in Search, Optimisation, and Machine Learning. Addison-Wesley, 1989.

Allen, Franklin, and Risto K (1999). "Using genetic algorithms to find technical trading rules." Journal of financial Economics 51.2 (1999): 245-271.

Subramanian, Harish, (2006). "Designing safe, profitable automated stock trading agents using evolutionary algorithms." Proceedings of the 8th annual conference on Genetic and evolutionary computation. ACM, 2006.

Lipinski P (2012). Parallel Evolutionary Algorithms for Stock Market Trading Rule Selection on Many-Core Graphics Processors. In: Brabazon, A., O'Neill, M., Maringer, D. (eds.) Natural Computing in Computational Finance. SCI, vol. 380, pp. 79–92. Springer, Heidelberg (2012)

Yan W, and Clack CD (2009). ”Behavioural GP diversity for adaptive stock selection”, In Proc. of the 11th annual conference on Genetic and evolutionary computation, pp. 1641 - 1648, 2009.

Lo, Andrew W, Harry M, Jiang W (2000). "Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation." The Journal of Finance 55.4 (2000): 1705-1770.

R. Sivaraj and Ravichandran T (2011). A review of selection methods in genetic algorithm, Int. J. Eng. Sci. Tech. 3,3792 (2011)

Biais, Bruno, and Paul W (2011). "High frequency trading." Manuscript, Toulouse University, IDEI (2011).

Sharpe (1994). The sharpe ratio. J. Portfolio Management, 21:49-58, 1994.

Chandra R, Dagum L, Kohr D, Maydan D, McDonald J, and Menon R (2001). Parallel Programming in OpenMP. Morgan Kaufmann, San Francisco, CA, 2001.

Sivanandam SN, and Deepa SN (2008). Genetic Algorithm Optimization Problems. Springer Berlin Heidelberg, 2008.

Andre J, Siarry P, and Dognon T (2001). "An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization." Advances in engineering software 32.1 (2001): 49-60.

Holland, John H (1975). Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. U Michigan Press, 1975.

McGinley, Brian, (2011). "Maintaining healthy population diversity using adaptive crossover, mutation, and selection." Evolutionary Computation, IEEE Transactions on 15.5 (2011): 692-714.

Miller, Brad L, and David E. Goldberg (1995). "Genetic algorithms, tournament selection, and the effects of noise." *Complex Systems* 9.3 (1995): 193-212.