Forecasting Euro - United States Dollar Exchange Rate with Gene Expression Programming

Maria A. Antoniou, Efstratios F. Georgopoulos, Konstantinos A. Theofilatos, Spiridon D. Likothanassis

Forecasting Euro–United States Dollar exchange rate with Gene Expression Programming

Maria A. Antoniou[1], Efstratios F. Georgopoulos[1,2], Konstantinos A. Theofilatos[1] and Spiridon D. Likothanassis[1]

[1] Pattern Recognition Laboratory, Dept. of Computer Engineering & Informatics, University of Patras, 26500, Patras, Greece
[2] Technological Educational Institute of Kalamata, 24100, Kalamata, Greece
{Maria Antoniou: antonium@ceid.upatras.gr, Efstratios F. Georgopoulos: sfg@teikal.gr, Konstantinos A. Theofilatos: theofilk@ceid.upatras.gr, Spiridon D. Likothanassis: likothan@cti.gr}

Abstract. In the current paper we present the application of our Gene Expression Programming Environment in forecasting Euro-United States Dollar exchange rate. Specifically, using the GEP Environment we tried to forecast the value of the exchange rate using its previous values. The data for the EURO-USD exchange rate are online available from the European Central Bank (ECB). The environment was developed using the JAVA programming language, and is an implementation of a variation of Gene Expression Programming. Gene Expression Programming (GEP) is a new evolutionary algorithm that evolves computer programs (they can take many forms: mathematical expressions, neural networks, decision trees, polynomial constructs, logical expressions, and so on). The computer programs of GEP, irrespective of their complexity, are all encoded in linear chromosomes. Then the linear chromosomes are expressed or translated into expression trees (branched structures). Thus, in GEP, the genotype (the linear chromosomes) and the phenotype (the expression trees) are different entities (both structurally and functionally). This is the main difference between GEP and classical tree based Genetic Programming techniques.

Keywords: Gene Expression Programming, Genetic Programming, Evolutionary Algorithms, System Modeling, time series, Euro-Dollar Exchange rate.

1 Introduction

The problem of discovering a mathematical expression that describes the operation of a physical or artificial system using empirically observed variables or measurements is a very common and important problem in many scientific areas. Usually, the observed data are noisy and sometimes missing. Also, it is very common, that there is

no known mathematical way to express the relation using a formal mathematical way. These kinds of problems are called modeling problems, symbolic system identification problems, black box problems, or data mining problems [2].
Most data-driven system modeling or system identification techniques assume an a-priori known model structure and focus mainly to the calculation of the model parameters' values. But what can be done when there is no a-priori knowledge about the model's structure?
Gene Expression Programming (GEP) is a domain-independent problem-solving technique in which computer programs are evolved to solve, or approximately solve, problems [3],[9]. GEP is a member of a broad family of techniques called Evolutionary Algorithms. All these techniques are based on the Darwinian principle of reproduction and survival of the fittest and are similar to the biological genetic operations such as crossover and mutation [9]. GEP addresses one of the central goals of computer science, namely automatic programming; which is to create, in an automated way, a computer program that enables a computer to solve a problem [1].
In GEP the evolution operates on a population of computer programs of varying sizes and shapes. GEP starts with an initial population of thousands or millions of randomly generated computer programs, composed of the available programmatic ingredients and then applies the principles of biological evolution to create a new (and often improved) population of programs. The generation of this new population is done in a domain-independent way using the Darwinian principle of survival of the fittest, an analogue of the naturally-occurring genetic operation of sexual recombination (crossover), and mutation. The fundamental difference between other Evolutionary Algorithms and GEP is that, in GEP there is a distinct discrimination between the genotype and the phenotype of an individual. So, in GEP the individuals are symbolic strings of fixed length representing an organism's genome (chromosomes/genotype), but these simple entities are encoded as non-linear entities of different sizes and shapes, determining an organism's fitness (expression trees/phenotype). GEP is a new evolutionary technique and its applications so far are quiet limited. However, it has been successfully applied in some real life problems [4], [5], [6].
In the current paper we present an integrated GEP environment with a graphical user interface (GUI), called jGEPModeling. The jGEPModeling environment was developed using the JAVA programming language, and is an implementation of the steady-state gene expression programming algorithm. In order to evaluate the performance of the jGEPModeling we tested it in the task of forecasting the next trading period of EUR/USD exchange rate based on the exchange rates of the past N periods, where N is a user-defined variable.

2 The Gene Expression Programming Algorithm

Gene Expression Programming is a new Evolutionary Algorithm proposed by Ferreira (2001) as an alternative method to overcome the drawbacks of Genetic Algorithms (GAs) and Genetic Programming (GP) [1], [3], [7], [9] . Similar to GA and GP, GEP follows the Darwinian principles of natural selection and survival of the fittest individual [8]. The fundamental difference between the three algorithms is that, in

GEP there is a distinct discrimination between the genotype and the phenotype of an individual. This difference resides in the nature of the individuals, namely in the way the individuals are represented: in GAs the individuals are symbolic strings of fixed length (chromosomes); in GP the individuals are non-linear entities of different sizes and shapes (parse trees); and in GEP the individuals are also symbolic strings of fixed length representing an organism's genome (chromosomes/genotype), but these simple entities are encoded as non-linear entities of different sizes and shapes, determining an organism's fitness (expression trees/phenotype) [3].

GEP chromosomes are usually composed of more than one gene of equal length. Each gene is composed of a head and a tail. The head contains symbols that represent both functions and terminals, whereas the tail contains only terminals. The set of functions usually includes any mathematical or Boolean function that the user believes is appropriate to solve the problem. The set of terminals is composed of the constants and the independent variables of the problem. The head length (denoted h) is chosen by the user, whereas the tail length (denoted t) is evaluated by:

$$t = (n-1)h+1 \quad (1)$$

, where n is the number of arguments of the function with most arguments. Despite its fixed length, each gene has the potential to code for Expression Trees (ETs) of different sizes and shapes, being the simplest composed of only one node (when the first element of a gene is a terminal) and the largest composed of as many nodes as the length of the gene (when all the elements of the head are functions with maximum arity). One of the advantages of GEP is that the chromosomes will always produce valid expression trees, regardless of modification, and this means that no time needs to be spent on rejecting invalid organisms, as in case of GP [9].

In GEP, each gene encodes an ET. In the case of multigenic chromosomes, each gene codes for a sub-ET and the sub-ETs interact with one another using a linking function (any mathematical or Boolean function with more than one argument) in order to fully express the individual. Every gene has a coding region known as an Open Reading Frame (ORF) that, after being decoded, is expressed as an ET, representing a candidate solution for the problem. While the start point of the ORF is always the first position of the gene, the termination point does not always coincide with the last position of a gene.

The flowchart of the GEP algorithm is shown in Figure 1. The process begins with the random generation of the linear chromosomes (or individuals) of the initial population. Then the chromosomes are expressed as ETs and the fitness of each individual is evaluated. After that, the individuals are selected according to their fitness in order to be modified by genetic operators and reproduce the new population. The individuals of this new population are, in their turn, subjected to the same developmental process: expression of the chromosomes, evaluation, selection according to fitness and reproduction with modification. The process is repeated for a certain number of generations or until a good solution has been found.

Next, follows a full description of the algorithm's steps:

1. Creation of initial population: The initialization in GEP is a very trivial task· in fact is the random creation of the chromosomal structure of the individuals. According to the nature of the problem, we must choose the symbols used to create the chromosomes, that is, the set of functions and terminals we believe to be appropriate to solve the problem. We must also choose the length of each gene, the number of

genes per chromosome and how the products of their expression interact with one another.

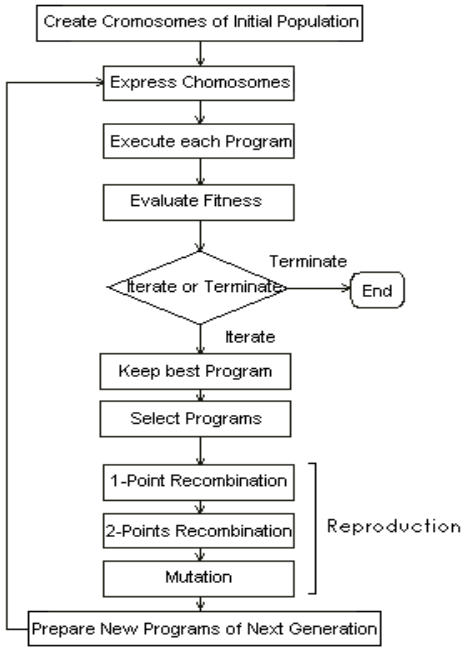

Fig. 1: Flowchart of gene expression algorithm.

2. Express chromosomes: The second step is the expression of the chromosome of each individual as an ET. This process is also very simple and straightforward. For the complete expression, the rules governing the spatial distribution of functions and terminals must be followed. First, the start of a gene corresponds to the root of the ET, forming this node the first line of the ET. Second, depending on the number of arguments of each element (functions may have a different number of arguments, whereas terminals have an arity of zero), in the next line are placed as many nodes as there are arguments to the elements in the previous line. Third, from left to right, the new nodes are filled, in the same order, with the elements of the gene. This process is repeated until a line containing only terminals is formed.
3. Execute each program: Having expressed each individual to an ET is now easy to find and compute the mathematical (or Boolean) expression it codes. We implement this by a post order traversal on the ET.
4. Evaluate fitness: One crucial step in GEP is finding a function that performs well for all fitness cases within a certain error of the correct value. In the design of the fitness function, the goal must be clearly and correctly defined in order to make the system evolve in the intended direction.
5. Keep best Program: A feature that plays a significant role in GEP is the elitism. Elitism is the cloning of the best chromosome(s)/individual(s) to the next population

(also called generation). By elitism, we guarantee that at least one descendant will be viable in the next generation, keeping at the same time the best trait during the process of adaptation.
6. Selection: In GEP, individuals are selected according to fitness by the tournament selection method to reproduce with modification. Tournament selection involves running several "tournaments" among a few individuals randomly chosen from the population. The winner of each tournament (the one with the best fitness) is selected for genetic modification. Selection pressure is easily adjusted by changing the tournament size. If the tournament size is larger, weak individuals have a smaller chance to be selected.
7. Reproduction: At this step of the GEP we apply the genetic operators of mutation and recombination on the winners of the tournaments.

a. Mutation: one new chromosome is created from an existing one, by mutating a single symbol. The mutation point is randomly chosen within the chromosome and then the symbol at this point is changing according to a mutation probability. In GEP, mutations can occur anywhere in the chromosome. However, the structural organization of chromosomes must be preserved.
b. Recombination: the parent chromosomes are paired and split up at exactly the same point(s). The material downstream of the recombination point(s) is afterwards exchanged between the two chromosomes according to a recombination probability. In GEP, an event of recombination always involves two parent chromosomes and always results in two new individuals.

Note that, during reproduction it is the chromosomes of the individuals, not the ETs that are reproduced with modification and transmitted to the next generation.
8. Prepare new programs of the next generation: At this step, we replace the tournament losers with the new individuals creating by reproduction in the population.
9. Termination criterion: We check if the termination criterion is fulfilled, if it is not we return to step 2. As a termination criterion it was used the maximum number of 500.000 generations that GEP was left to run.
10. Results: As a result we return the best individual ever found during the evolution process.

3 Modeling Experiments

In this section we present the performance of GEP environment in a real series forecasting problem. Particularly, we apply the GEP environment to an one- day-ahead forecasting and trading task of the Euro/Dollar (EUR/USD) exchange using the Euro Central fixing series.

The European Central Bank (ECB) publishes a daily fixing for selected EUR exchange rates: these reference mid-rates are based on a daily concertation procedure between central banks within and outside the European System of Central Banks, which normally takes place at 2.15 p.m. ECB time. The reference exchange rates are published both by electronic market information providers and on the ECB's website shortly after the concertation procedure has been completed. Although only a reference rate, many financial institutions are ready to trade at the EUR fixing and it

is therefore possible to leave orders with a bank for business to be transacted at this level.
The ECB daily fixing of the EUR/USD is therefore a tradable level which makes using it a realistic approach.

Table 1. Dataset description

Name of period	Trading days	Beginning	End
Total dataset	2721	6 January 1999	23 October 2009
Training dataset	2178	6 January 1999	30 August 2007
Out-of-sample dataset [Test set]	543	31 August 2007	23 October 2009

Based on the exchange rates of the past N periods, where N is a user-defined variable, we forecast the next trading period EUR/USD exchange rate. The difficulty in this modeling problem is to identifυ and the capturε all the discontinuous, the nonlinearities and the high frequency multipolynomial components which characterize the financial series today.
After a certain number of experiments, we are in position to define standard values for some running parameters of GEP. Table 2 defines all these defaults parameters we use for our experiments.

Table 2. Default parameters derived by experimentation.

Parameter	Value
Number of Generations	300.000
Function Set	{+, -, *, /, ^, $\sqrt{}$,abs, cos, sin, ln, exp, tan, min, max}
Constants Range	[-3, 3]
Mutation Probability	0.9
Population Size	1500
Tournament Size	4
Head Size	30
Type of Recombination	two points recombination

Next, we present our experiments with various values of past exchange rates used for our prediction.

Table 3. Results taken by experimenting with models from GEP which use various past exchange rates for prediction.

Past Exchange Rates	Mean Square Train Error	Mean Square Test Error
1	3.528493651301604e-005	7.038129152437989e-005
2	3.525297196338534e-005	7.002301886682884e-005
3	3.529949324406443e-005	6.967481960607426e-005
4	3.510071391816327e-005	6.979235421769149e-005

We observed that using only three past values of past exchange rates for the prediction models, gives as the best performance. Increasing the number of previous

values to be used as inputs did not enhanced our models and this reveals short term phenomena in the time series of EURO/USD exchange rate. The best model found has mean square test error 6.959490069577218E-5.

For comparison reasons, we experimented using our dataset with 3 other methods that have been previously used in this forecasting problem [11], [12]. These methods are the naïve method, the moving average predictor and Artificial Neural Networks. Specifically:

- Naïve strategy assumes that the most recent period change is the best predictor of the future [11], [12].
- Moving Average method uses a weighted average of past observations to smooth short-term fluctuations. In this approach 35 previous values were used as in [11], [12].
- Neural Network models are universal approximators capable of approximating any continuous function [13]. In this approach we used a feed-forward Multi-Layer Perceptron(MLP) using 4 previous values as inputs in order to make a fair comparison with our method. For the neural network training we used back propagation algorithm.

The performance of each model in the out of sample period is presented in Table 4. Because of the stochastic nature of the MLP method we experimented 100 times with it and we show the mean results. We observe that gene expression programming outperforms the three classic methods in our forecasting problem.

Table 4. Experimental results of three benchmark methods in the out of sample period.

Method	Mean Square Test Error
Naïve strategy	1.3475e-004
Moving Average	7.5000e-005
MLP	7.4551e-005

4 Conclusions

The reported results taken by the application of our GEP implementation in the task of forecasting the next trading period of EUR/USD exchange rate based on the exchange rates of the past N periods, confirm our intuition that Gene Expression Programming is a very effective technique in system modeling and timeseries prediction. So, we conclude that our GEP environment can be used in a variety of problems in different scientific areas. Also, it is important to note that the rapid development of very fast computer hardware encourages the use of techniques like GEP, and is expected that these kinds of techniques will be used more and more for solving difficult problems in the future.

Now, concerning some future directions of the presented GEP environment these could be:

- The application of our GEP implementation in other system modeling tasks in various scientific areas, such as system identification, timeseries prediction, e.t.c..

- The further improvement of the tool in terms of speed, memory management and parallel implementation.
- The implementation of more sophisticated genetic operators, initialization techniques and selection techniques in the basic Gene Expression Programming technique.

Thus, our environment has been proved to be a powerful GEP tool with great expansion capabilities that we intend to perform in the near future.

5 References

1. Koza J.R.: Genetic programming: on the programming of computers by means of natural selection, MIT Press, Cambridge, MA, (1992)
2. S. Winkler, M. Affenzeller, S. Wagner: Identifying Nonlinear Model Structures Using Genetic Programming Techniques. Cybernetics and Systems 2004, pp. 689-694. Austrian Society for Cybernetic Studies, (2004)
3. Ferreira C: Gene Expression Programming: A New Adaptive Algorithm for Solving Problems. Complex Systems, Vol. 13, issue 2:87-129, 2001
4. Lopez H. S., Weinert W. R.: EGIPSYS: An Enhanced Gene Expression Programming Approach for Symbolic Regression Problems, International Journal of Applied Mathematics in Computer Science, Vol. 14, No. 3, p.p. 375-384, 2004
5. Margny M. H., El-Semman I. E.: Extracting Logical Classification Rules with Gene Expression Programming: Micro array case study, AIML 05 Conference, 19-21 December 2005, Cairo, Egypt
6. Dehuri S., Cho S. B.: Multi-Objective Classification Rule Mining Using Gene Expression Programming, Third International Conference on Convergence and Hybrid Information Technology,2008
7. Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, 1989.
8. Darwin C.: On the Origin of Species, 1859.
9. Ferreira C.: Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence, 2nd edition, Springer, 2006.
10. Mitchell M.: An Introduction to Genetic Algorithms (MIT Press, 1996).
11. Dunis C., Williams M., Modelling and Trading the EURO/USD Exchange Rate: Do Neural Network Models Perform Better?, Derivatives Use, Trading and Regulation, 2002.
12. Dunis C., Laws J., Sermpinis G., Modelling and trading the EUR/USD exchange rate at the ECB fixing, The European Journal of Finance, 2009.
13. Haykin, S., Neural Networks: A comprehensive foundation, 2nd edn. Prentice Hall, Upper Saddle River (1999)