On the Utility of Trading Criteria based Retraining in Forex Markets

Alexander Loginov[1] and Malcolm I. Heywood[1]

[1]*Faculty of Computer Science, Dalhousie University, Halifax, NS. Canada*

Article originally appears at EvoAPPS'13 under Springer copyright 2013
`http://link.springer.com/chapter/10.1007\%2F978-3-642-37192-9_20`

Abstract

This research investigates the ability of genetic programming (GP) to build profitable trading strategies for the Foreign Exchange Market (FX) of three major currency pairs (EURUSD, USDCHF and EURCHF) using one hour prices from 2008 to 2011. We recognize that such environments are likely to be non-stationary. Thus, we do not require a single training partition to capture all likely future behaviours. We address this by detecting poor trading behaviours and use this to trigger retraining. In addition the task of evolving good technical indicators (TI) and the rules for deploying trading actions is explicitly separated. Thus, separate GP populations are used to coevolve TI and trading behaviours under a mutualistic symbiotic association. The results of 100 simulations demonstrate that an adaptive retraining algorithm significantly outperforms a single-strategy approach (population evolved once) and generates profitable solutions with a high probability.

1 Introduction

The Foreign Exchange (FX) Market is the world biggest financial market which produces 1/3 of all financial transactions in the world [12]. The average daily turnover of FX was almost $4 trillion in 2010 and it was 20% higher in April 2010 than in April 2007 [15]. An FX market consists of currency pairs which are weighted by economic conditions for that specific denomination versus any or all others in the marketplace. Thus, the perceived value of a currency is a reflection of the market's ranking for that denomination's economy on any given day. An FX market is technically 'purer' than a stock market i.e., a currency price action reacts more strongly to resistance and support levels than equity market does [13]. All the above factors make FX markets very attractive for traders and expands the demand for automated trading systems, albeit under demanding conditions.

However, the underlying data describing such trading environments is typically non-stationary. Thus, assuming the classical approach of training over fixed partitions of data (e.g., training, validation and test) results in brittle solutions that could be specific to the partition on which they are evolved [5], Chapter 7. Moreover, the use of validation data in financial data forecasting is not in itself a "silver bullet" [16]. One solution proposed to this problem is to train on a continuous basis, while explicitly maintaining population diversity [5], Chapter 8. Thus, following the initial evolution of a population of trading agents, taking G generations, the best solution trades for 'n' days. The same 'n' trading days are then employed to readapt the population for $g < G$ generations. However, the selection of appropriate g and n is in itself a function of the data (market dynamic). Another approach might be to coevolve a subset of the training partition [11] or to combine coevolution of the training partition with an explicitly streaming context [1].

In this work we investigate a somewhat different approach. As emphasized by the study of [5], we recognize that under non-stationary environments assuming a modular representation can be beneficial. There are several ways of potentially achieving this. In this work we adopt a symbiotic approach to genetic programming (e.g., [10, 6]). Specifically, trading indicators (TI) are coevolved with trading decision trees (DT). Assuming a symbiotic approach enables us to adopt unique representations for each while searching both representations in a co-ordinated way. Secondly, we define specific trading criteria that characterize when retraining should take place. Thus, relative to a fixed training–validation parameterization, training is only ever re-triggered by the trading criteria. Now, *relative to the point of re-triggering*, an entirely new training–validation sample is defined to evolve a new trading agent. Naturally, there is a trade off with regards to the cost of retraining versus maintaining a population of previously evolved models for redeployment e.g., [3]. Under this particular deployment, the retraining cost is significantly smaller than the interval between consecutive data pairs, thus the retraining cost does not detract from 'real-time' operation.

The proposed approach – hereafter FXGP – is demonstrated relative to the task of FX trading under three currencies for 3 year periods in each case. A base case employing the same symbiotic representation but without retraining provides a performance baseline. Conversely, adding the capability to dynamically identify retraining points through trading criteria results in a significant improvement.

1.1 Glossary

Hereafter the following terminology will be assumed [7]:

Ask: Price at which broker/dealer is willing to sell. Also referred to as "Offer".

Balance: The value of your account not including unrealized gains or losses on open positions.

Bid: Price at which broker/dealer is willing to buy.

Drawdown: The magnitude of a decline in account value, either in percentage or currency terms, as measured from peak to subsequent trough.

Fundamental Analysis: Macro or strategic assessment of where a currency should be trading on any criteria but the price action itself. The criteria often includes the economic condition of the country that the currency represents e.g., monetary policy, and other "fundamental" elements.

Leverage: The amount, expressed as a multiple, by which the notional amount a trader exceeds the margin required to trade. The specific approach taken to this is described in Section 3.

Pip: The smallest price increment in a currency. Often referred to as "ticks" in the future markets. For example, in EURUSD, a move of 0.0001 is one pip.

Spread: The distance, usually in pips, between the Bid and Ask prices. Section 3 details the fixed spread assumed in this work.

Stop: Also called "Stop Loss" or "S/L". An order to buy or sell when the market moves to a specific price.

Technical Analysis: Analysis applied to the (price) action of the market to develop a trading decision, irrespective of fundamental factors.

2 Proposed Algorithm

2.1 The FXGP Algorithm Overview

As established in the introduction, a cycle of evolution is initially completed against the first 1,000 records (N_t) and validated against the next 500 (N_v); or $\approx$ 2 months and 1 month respectively. Trading then commences until some failure criteria is satisfied – e.g., an excessive drawdown is encountered – at which point the $N_t + N_v$ records leading up to the failure are used to evolve a new trading agent, and the process repeats. Note, however, that such a scheme is distinct from the purpose or configuration of N-fold cross validation. The first training–validation pair as initially employed to evolve the first FXGP individual, any later calls to evolve a new FXGP individual are relative to the failure criteria triggering retraining. Moreover, retraining results in all current population content being reset as the re-trigger event is taken to imply that the current content is inappropriate.

The trading agent takes the form of a decision tree and corresponding set of *coevolved* technical indicators or DT and TI populations respectively. Evolving the TI provides the opportunity to capture properties pertinent to specific nodes of a decision tree defining the overall trading rule; where the characterization of such temporal properties is known to be significant for a wide range of time series tasks [17]. Four prices – Open, High, Low and Close – are used as inputs to the TI population. Members of the DT population define the trading rule, and it is only with respect to the DT individuals that fitness is evaluated i.e., a symbiotic GP relationship [10, 6].

Table 1: TI functions. Note the three forms of division

Function	Definition	Function	Definition
Addition	$R[x] \leftarrow R[x] + R[y]$	Subtraction	$R[x] \leftarrow R[x] - R[y]$
Multiplication	$R[x] \leftarrow R[x] \times R[y]$	Square root	$R[x] \leftarrow \sqrt{R}[y]$
Division	$R[x] \leftarrow R[x] \div R[y]$	Invert	$R[x] \leftarrow 1 \div R[x]$
Div-by-2	$R[x] \leftarrow R[x] \div 2$	–	–

2.2 Training

2.2.1 Initialization

The *TI population* is randomly initialized with (initial) size defined by the user. TI individuals assume a linear GP representation (e.g., [2]) with instruction set summarized by Table 1. Moreover, each TI has a header defining the basic TI properties: type, scale, period, shift (Table 2).

Table 2: TI parameters. Estimated relative to the current hour t of trading.

Parameter	Description
TI type	Moving Average (MA), Weighted Moving Average (WMA) or Value
TI scale	TI that crosses 0 or TI that crosses price
Period n	Number n of hours in a price history to calculate MA or WMA
Shift m	Price m hours back

TI programs assume a register level transfer language (Table 1). Thus, $R[x]$ denotes the content of register x, $R[y]$ denotes either: register y content; a price, or; a price m hours back in (relative) time. Register $R[0]$ is assumed to contain a TI value after executing a TI program. The MA type of TI is calculated as (1) whereas the WMA type of TI is calculated as (2), where V_j is a TI value.

$$MA_i = \frac{\sum_{j=1}^{n} V_j}{n} \tag{1}$$

$$WMA_i = \frac{\sum_{j=1}^{n} \frac{V_j}{j+1}}{\sum_{j=1}^{n} \frac{1}{j+1}} \tag{2}$$

The *DT population* is initialized to a fixed size as defined the user. A DT header includes the following information: DT *score* in pips, number of trades and a size of a S/L order in pips. The score and number of trades are initialized with 0; whereas the S/L is assumes a fixed interval. A DT consists of a variable number of nodes. Each node consists of a conditional statement with either single or dual antecedent tests of the form:

- *if($X_i > Y_i$) then else*
- *if(($X_i > Y_i$) and ($X_{i+m} < Y_{i+m}$)) then else*

where X_i and Y_i can be 0, price or a TI. The *then* and *else* statements reference either: the next node or one of the trading signals: buy, sell or stay. Thus, a DT population is randomly generated with respect to X_i and Y_i scales; albeit under the following constraints:

1. if X_i is 0, then Y_i can be any TI which crosses 0 and can not be a price or a TI which crosses the price, or;
2. if X_i is price, then Y_i can be also a price or a TI which crosses the price, and can not be 0 or a TI which crosses 0.

Note in the case of a dual antecedent clause, X_{i+m} and Y_{i+m} represent the value of X_i or Y_i respectively, albeit m samples back in (relative) time. FXGP can generate additional TI during DT population initialization if the TI population does not have a TI capable of satisfying the DT initialization constraints.

2.2.2 Fitness and Selection

The DT fitness is defined as a DT score in pips over the training records. When TI and DT populations are initialized, FXGP simulates the trading activity for each DT over the training records and stores the score and the number of trades in the DT header. The number of trades is used to penalize the DT score for too high or low a frequency of trading. Moreover, if a DT generates only *buy* or *sell* signals its score is discarded and the DT targeted for replacement. The subset of DT individuals with lowest scores are explicitly identified for replacement cf., a breeder model of selection / replacement. Thus, a fixed percentage or *gap* size of the DT population is replaced per generation. All variation is asexual, thus following parent selection, cloning and variation takes place where either the DT or TI component of a clone can be varied. Following DT selection, any TI individual that no longer receive a DT index are considered ineffective and are also deleted (resulting in a variable size TI population).

2.2.3 Mutation

FXGP uses mutation to produce offspring. Only one DT node or one linked TI can be mutated in each cloned TI–DT pair. FXGP randomly selects the target for mutation (TI or node) according to the probability of mutation (Table 4). A *mutated TI* is first cloned to avoid interfering with other DT employing the same TI. The following parameters and functions of a TI can be mutated: TI type, period (n), shift (m), generate a new function, delete a function, or insert a function. Likewise, a *DT Mutation* also begins by cloning the parent, and then applies one of the following to define an offspring:

1. Generate a new conditional function;
2. Increment / decrement shift parameter m;
3. Generate new X_i and Y_i;
4. Switch X_i and Y_i;
5. Switch content of *then* and *else* clauses;
6. Insert new *then* clause content.
7. Insert new *else* clause content.

Training stops when the specified number of generations was reached or when the best score in the DT population plateaus for a fixed number of generations, τ (Table 4). Thereafter a validation cycle is initiated.

2.3 Validation

During validation we require a proportion of the population α_v to demonstrate feasible trading behaviour before trading may actually commence. FXGP checks the score of every tree in a DT population and if the DT score is greater than $\alpha_v \times$ *best score* it tests the DT and increments the tested tree's counter. When all DT are evaluated, FXGP checks the number of the tested DT and if it is greater than $\alpha_v \times$ *DT population size* then the TI–DT pair with best score on validation is selected as the 'champion' for trading, otherwise the entire training procedure is restarted i.e., rather than invest more generations in a population which fails under validation we choose to reinitialize and restart the training cycle.

2.4 Trading and Retraining criteria

During FX trading, the following three trading quality criteria are monitored and used to trigger retraining from an entirely new population pair of TI–DT populations: 1) Drawdown i.e., decline in account value – see Glossary; 2) The number of consecutive losses permitted; and 3) The number of consecutive hours without trading activity. This approach lets FXGP retrain the DT population only when the market situation is deemed to differ from that of the last training period. In doing so we explicitly recognize the non-stationary nature of the task. Specific values for the quality criteria are established by the user. When the quality criteria are exceeded, FXGP stops trading, reinitializes the TI and DT populations and restarts the training-and-validation cycle. Content of the training and validation partitions is taken relative to the point 't' at which the trading quality criteria interrupted trading. Once a new TI–DT champion is identified trading resumes at the point trading was interrupted.

Table 3: Trading conditions with initial trading balance of 100,000 USD

Condition	Value	Condition	Value
Spread USDCHF	0.0003	Min. S/L level	0.0005
Spread EURUSD	0.0002	Pip	0.0001
Spread EURCHF	0.0005	Leverage	1:100

Table 4: FXGP parameterization

Parameter	Value	Parameter	Value
DT pop. size	100	DT gap	25
Training period, hours (N_t)	1,000	Validation period, hours (N_v)	500
Max. generation	1,000	Max. DT size, nodes	6
Max. TI program size, steps	8	Number of TI registers	2
Probability of TI mutation	0.5	S/L size, pips	100
Training plateau length (τ)	200	TI–DT validation fraction (α_v)	0.7
Trading Quality Criteria			
Num. consecutive losses	5	Drawdown, pips	400
–	–	Max. time without trading activity, hrs	72

2.5 Source Data

The historical rates for EURUSD, USDCHF and EURUSD (1 hour resolution) were downloaded with MetaTrader 4 FX terminal from the MetaQuotes Software Corp. history centre [7, 4]. Each 24 hour period therefore typically consists of 24 samples. Sampling at the rate of one hour intervals is conducted to reduce the impact of "trading noise" [8]. Three datasets are employed each of which includes 2008-2011 historical rates and consists of following fields: Date, Time, bid price Open (Open), bid price High (High), bid price Low (Low), bid price Close (Close) and Volume.

3 Experimental setup

Performance is evaluated for the case of TI–DT trading agents as evolved from a single initial cycle of evolution versus the proposed scheme for retraining TI–DT trading agents interactively as established by the trading quality criterion (100 runs in each case); hereafter static and retrain respectively. Each run simulates trading activity for three major currency pairs (EURUSD, USDCHF and EURCHF) over the trading period from January 2, 2009 to December 30, 2011 (approx. 18,500 hours). These pairs are popular during all trading sessions i.e., a trading day typically consists of 24 samples. The trading conditions [4] are described in the Table 3.

An initial balance of 100,000 USD is assumed and the leverage is 1:100. The account balance was then recalculated in USD based on a "2% Rule" [9] i.e., only two percent of the current balance may be reinvested at the next round of trading. Moreover, in addition to quoting the resulting pips and USD at the end of 3 years, we also consider the 'overall' trading outcome when conducting trading across a portfolio of currencies. This represents the case of a trader conducting independent runs for all three currency pairs and expressing performance as the combined income from the portfolio of three currency pairs (again at the end of the 3 year period).

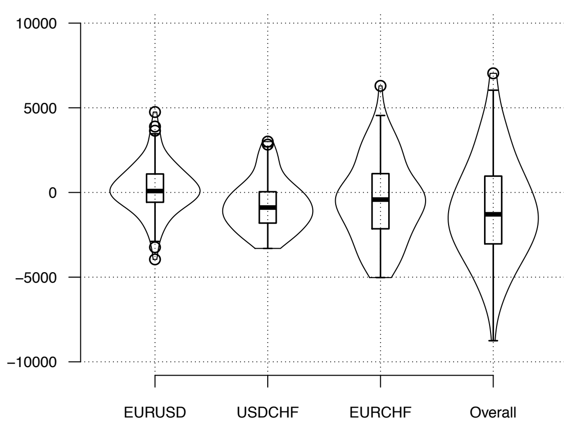
4 Results

The resulting (pips) distribution for one hundred runs for TI–DT under 'static' and 'retrain' deployments are shown in the Fig. 1(a) and Fig. 1(b) respectively. Table 5 provides summary statistics. Not only is the profitability of TI–DT traders with retraining significantly higher, but the number of profitable runs are discovered between 1.3 to 3.5 times more frequently (depending on the currency pair). For completeness the account balance *during trading* of the best, typical and worse runs are shown in Fig 2(a) (pips) and Fig 2(b) (USD).

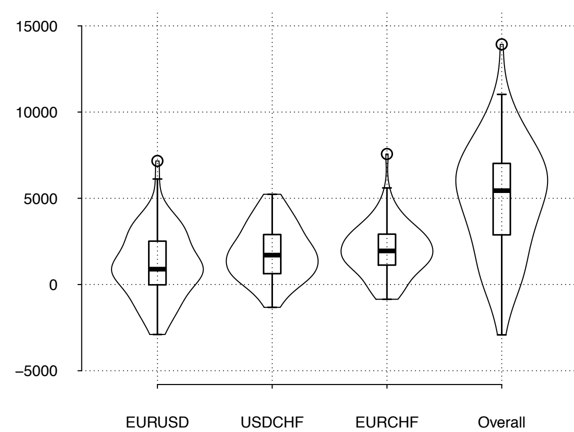
Finally, we can also consider the typical interval between retraining episodes, Figure 3. Retraining appears to be called at a median rate of 200 hours, implying that over the total (three year) trading interval, there are 92 calls for retraining. From an end user perspective this corresponds to retraining once every 8 days (with trading performed on a 5 day trading week). Retraining the population under the current parameterization takes 30

Table 5: TI–DT Static versus Retraining summary. For consistency with Figure 2(b) 'Overall balance' include the initial trading balance of 100,000 USD.

Description	retrain	static	retrain vs static, %
EURUSD profitable runs, %	75	58	129.3
USDCHF profitable runs, %	88	25	352.0
EURCHF profitable runs, %	89	38	234.2
Overall profitable runs, %	92	34	270.6
Best overall score, pips	13929	7036	n/a
Typical overall: pips	7500	-2000	n/a
Worse overall: pips	-2922	-8753	n/a
Best overall balance: USD	1,242,920.54	414,005.96	n/a
Typical overall balance: USD	356,825.19	61,528.97	n/a
Worse overall balance: USD	44,816.85	13,513.47	n/a

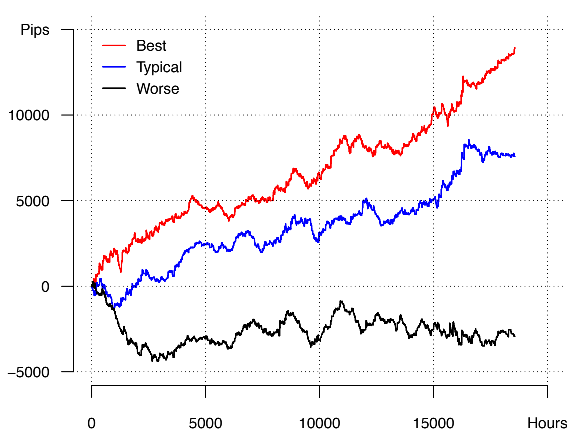


(a) Pips

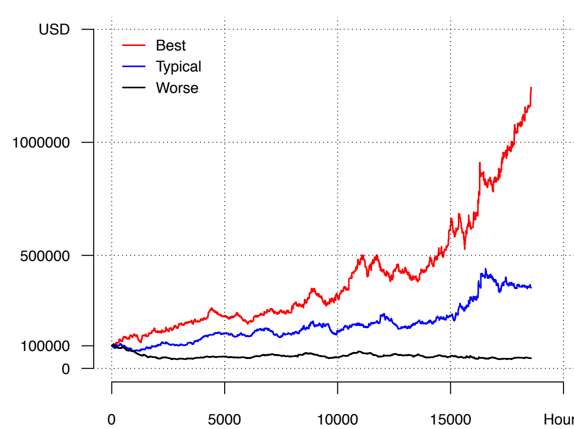


(b) USD

Figure 1: Performance distributions for TI–DT in **pips**: (a) static TI–DP and (b) TI–DP with retraining criteria. Internal box-plot provides quartile statistics. Violin profile characterizes the distribution.



(a) Pips



(b) USD

Figure 2: Adaptive TD results best, typical and worse runs for (a) Pips metric and (b) USD (including initial balance of 100,000 USD).

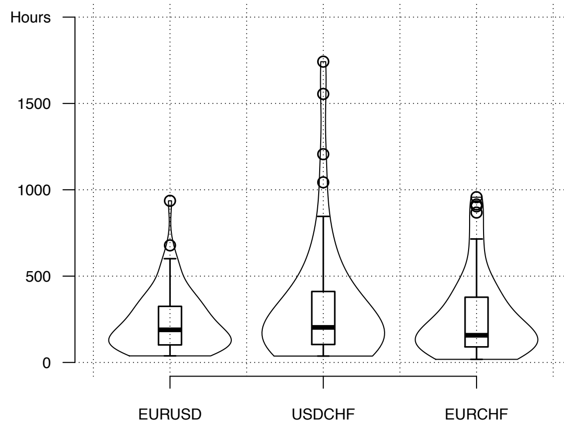


Figure 3: Distribution of retraining per currency over trading period.

seconds on an iMac desktop.[1] This implies that the approach is able to complete the retraining cycle well before the next (1 hour) sample is received, effectively rendering the approach 'real-time' from a deployment perspective.

Finally, by way of comparison, we note that investing $100,000 in a commercial mutual fund over the same period (January 2, 2009 to December 30, 2011) would result in a best case rate of return of 5.58% [14] i.e., an account balance of $117,687 at the end of the period. This is clearly better than the case of static TI–DP, but is typically bettered by TI–DP with retraining enabled. Moreover, post 2011 hindsight would be necessary to select such a profitable mutual fund.

5 Conclusion

It is increasingly being acknowledged that the non-stationary aspect of trading environments places additional requirements on the model building process for constructing trading agents. There are at least three different factors: providing an appropriate representation, detecting change, and maintaining diversity in the models proposed. This work assumes two specific properties: 1) a highly modular representation care of coevolving TI and DT populations, and 2) dynamically re-triggering training relative to a set of trading criteria or change detection. Moreover, the approach taken to change detection is to adopt criteria that a trader might well assume in practice. Such a scheme appears to be feasible with both profitable trading strategies typically discovered and, given the hourly rate of receiving new data, sufficient time to complete retraining before a new sample is received. As a final verification of the approach, randomly selected TI–DT 'traders' were implemented in the MetaTrader 4 trading terminal [4] as Expert Advisors and tested on the FxPro demo account. The obtained result were similar to the results of the FXGP simulation.

Acknowledgements

The authors gratefully acknowledge support from MITACS and NSERC (Canada).

References

[1] A. Atwater, M. I. Heywood, and A. N. Zincir-Heywood. GP under streaming data constraints: A case for Pareto archiving? In *ACM Genetic and Evolutionary Computation Conference*, pages 703–710, 2012.

[2] M. Brameier and W. Banzhaf. *Linear Genetic Programming*. Springer, 2007.

[3] I. Contraras, J. I. Hidalgo, and L. Nú nez Letamendia. A GA combining technical and fundamental analysis for trading the stock market. In *EvoApplications*, volume 7248 of *LNCS*, pages 174–183, 2012.

[4] MetaQuotes Software Corp, accessed Sept, 2012. `http://www.fxpro.com/trading/cfd/mt4/forex`.

[1] Intel Core i7, 2.8 GHz, 16 Gb RAM 1333 MHz DDR3, OS X 10.7.5

[5] I. Dempsey, M. O'Neill, and Brabazon A. *Foundations in Grammatical Evolution for Dynamic Environments*, volume 194 of *Studies in Computational Intelligence*. Springer, 2009.

[6] J. A. Doucette, A. R. McIntyre, P. Lichodzijewski, and M. I. Heywood. Symbiotic coevolutionary genetic programming. *Genetic Programming and Evolvable Machines*, 13(1):71–101, 2012.

[7] ICM Trade Capital Markets Ltd. Guide to online forex trading. 19 pages.

[8] Investopedia, accessed Sept, 2012. `http://www.investopedia.com/terms/n/noise.asp#axzz27d0d2rid`.

[9] Investopedia, accessed Sept, 2012. `http://www.investopedia.com/terms/t/two-percent-rule.asp#axzz2710QU8jR`.

[10] P. Lichodzijewski and M. I. Heywood. Symbiosis, complexification and simplicity under GP. In *ACM Genetic and Evolutionary Computation Conference*, pages 853–860, 2010.

[11] M. Mayo. Evolutionary data selection for enhancing models of intraday forex time series. In *EvoApplications*, volume 7248 of *LNCS*, pages 184–193, 2012.

[12] I. V. Morozov and R. R. Fatkhullin. Forex: from simple to complex, 2004. Teletrade Ltd.

[13] A. Passamonte. Six facts that give forex traders an edge. *Forex Journal*, 2011. `http://www.forexjournal.com/fx-education/forex-trading/12125-six-facts-that-give-forex-traders-an-edge.html`.

[14] RBC Global Asset Management. Investment portfolio tools, Jan 2013. `https://services.rbcgam.com/portfolio-tools/public/investment-performance/`.

[15] Bank For International Settlement. Triennial central bank survey of foreign exchange and otc derivatives market activity - preliminary global results, April 2010. `http://www.bis.org/press/p100901.htm`.

[16] C. Tuite, A. Agapitos, M. O'Neill, and A. Brabazon. A preliminary investigation of overfitting in evolutionary driven model induction. In *EvoApplications*, volume 6625 of *LNCS*, pages 120–130, 2011.

[17] N. Wagner, Z. Michalewicz, M. Khouja, and R. R. McGregor. Time series forecasting for dynamic environments: The DyFor genetic program model. *IEEE Transactions on Evolutionary Computation*, 11(4):433–452, 2007.