Better Trade Exits for Foreign Exchange Currency Trading using FXGP

Alexander Loginov[1], Garnett Wilson[1], and Malcolm Heywood[1]

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

Article originally appears at CEC'15 Companion under IEEE copyright 2015
`http://ieeexplore.ieee.org/document/7257197/`

Abstract

Retracement is the tendency of markets to move between upper 'resistance' and lower 'support' price levels. Human traders frequently make use of visual tools to help identify these resistance and support levels so that they can by used in their trading decisions. These decision can be put into trading strategies composed of rules designed to mitigate losses after a trade is started, often called 'stop loss' orders, or to take profit at a near optimal time, often called 'take profit' orders. However, identifying such resistance and support levels is notoriously difficult given market volatility. Indeed, the levels need recalculating on a continuous basis, and only hold to an approximate degree. In this work we describe an approach for evolving buy–stay–sell currency trading rules using genetic programming. These rules are explicitly linked to technical indicators that incorporate features characterizing retracement. Benchmarking is then performed using the most recent three years of data from the EURUSD foreign exchange market with three different methods of identifying retracement based on moving average, pivot points and Fibonacci ratios. Investment strategies employing Fibonacci ratios and found to provide superior performance among the strategies examined.

1 Introduction

Many strategies for intraday foreign exchange (FX) currency trading have been devised. One popular technique is to use well-established technical indicators (TI) that describe currency trends and then combine these trends into an appropriate trading strategy. The technical indicators can be used together in rules formed by a decision tree (DT), for instance. Various authors have proposed schemes for combining known TIs using rule structures identified by evolutionary computation or other forms of machine learning e.g., [4, 17, 3]. On the other hand, both TI and DT might be identified through some form of machine learning e.g., [8, 6, 7, 9]. In this work, we examine a recently proposed scheme called 'FXGP' [12, 11] for coevolving both TI and DT simultaneously through a symbiotic dual population framework.

The goal of this work is to extend FXGP to incorporate the use of 'retracement,' or the tendency for a currency pair's price to move between an upper and lower bound. The upper bound is known as the 'resistance' level, and the lower bound is known as 'support.' The identification of support and resistance levels characterize the direction of movement of a price and therefore form the basis for predictive sell or buy strategies. The support and resistance levels are used for the purpose of creating stop loss and take profit orders. Specifically, support levels define a price at which downward trends do not pass beyond, and hence they appear as if they bounce off the support level. That is, a downward price movement is expected to most often hit the support level, not pass through it, and then start a climbing price movement. However, should the price actually manage to break through the support level, then the drop is likely to continue until a new support level appears. Conversely, resistance levels define the price that an upward trend is observed to repeatedly bounce against. Likewise, should the price push through this resistance level, then it will continue until another resistance level is encountered. The challenge of achieving the greatest profits using these methods is to find support and resistance levels before they explicitly occur i.e., proactively determine support and resistance. A range of schemes have been proposed for this purpose, including comparisons of price against pivot points (e.g., [15]), moving average envelopes (e.g., [10, 16]), or Bollinger bands (e.g., [10, 1, 2, 3]).

In this work, we are interested in the case of introducing retracement levels for the purposes of dynamically characterizing the size of stop loss (SL) and take profit (TP) orders. Stop loss orders represent an *a priori* rule structured to stem further losses [14]. The converse, take profit orders, act in the predicted direction and result in closing a trading order at a profit. Previous instances of FXGP assumed that the SL orders were evolved based on the training partition of data [12, 11]. We continue with these assumptions in the extended versions of FXGP that we present in this work. We successfully evolve trading strategies that both minimize the number of times that a SL order is triggered and minimize the average size of SL orders. To accomplish these goals, we combine the SL orders with the trading rules evolved by FXGP such that they are more proactive and less sensitive to specific thresholds than was previously the case.

Section 2 provides the functionality for the three major schemes from which different families of SL and TP order are derived: pivot points, moving averages, and Fibonacci ratios. The proposed extension of FXGP is detailed in Section 3. A benchmarking study is performed using the EURUSD currency pair in order to empirically identify the most effective scheme for retracement (Section 5). The resulting ranking identifies Fibonacci and Moving average (cf., Bollinger band) definitions for retracement as being most effective. Section 6 concludes the paper.

2 Background

For the purpose of this research three commonly used technical indicators are selected for designing SL and TP orders: Fibonacci ratios, Pivot point and Moving average. Moving averages form the basis for the indicator known as 'Bollinger Bands,' as discussed below. In the following section, we summarize the basic scheme assumed for each.

2.1 Moving average

The Moving average (MA) over a period of n bars from time index i is calculated as (1), where C_j is a Close price (Figure 1).

$$MA_i = \frac{\sum_{j=0}^{n-1} C_{i+j}}{n} \tag{1}$$

SL and TP orders can then be defined relative to the MA, in particular MA$\pm\theta$ for some value θ to define a moving average envelope around the MA. The special case of θ set to some multiple k of the n-period standard deviation above and below the n-period MA of that series of prices gives rise to Bollinger bands. In such a case, the higher and lower band vary as a function of market volatility. In the context of genetic programming a stop loss order might take the form of MA $-\theta_l$ for a lower bound θ_l whereas a take profit order might take the form of MA $+\theta_p$ for a chosen upper bound of θ_p, where θ_p and θ_p are evolved thresholds. Leung *et al.* noted that the more general case of MA was preferable to Bollinger bands [10], whereas Butler and Kazakov recommended adapting the θ and n parameters on a continuous basis [1, 2].

Classically, the MA based form for a stop loss order might take the form of MA-θ_l whereas a take profit order might take the form of MA++θ_p, where θ_p and θ_p are evolved thresholds:

- Buy order: IF $price(t) > MA(t)$ THEN $SL = MA - \theta_l$ ELSE (DO NOT TRADE)
- Sell order: IF $price(t) < MA(t)$ THEN $SL = MA + \theta_l$ ELSE (DO NOT TRADE)

This will be detailed further in Section 3, and is shown in Figure 1. Figure 1, and all other price time series figures in this work, use what is called "candlestick" charting to represent price changes. Each candlestick is a box (or line if the box is absent) that can have an additional bar above and below that box/line (but it may not). The box is often called the "body," and the bars are called the "wicks." Each candlestick in this work represents a time period of one hour. If the price of a currency pair closed higher than it opened for that period, the body is white and the opening price is the bottom of the body and the close price is the top of the body. If the price of a currency pair closed lower than it opened, the body is black and the opening price is the top of the body and the closing price is the bottom. If a line is present instead of a body, then the opening price is the same as the close. The wick represents the highest and lowest prices traded during the time interval.

2.2 Fibonacci ratios

Fibonacci ratios used as a trading strategy that involves determining support and resistance levels based on the Fibonacci sequence. In particular, the ratios are derived by dividing a number in the Fibonacci sequence by some following number in the sequence. These ratios are then used as the divisor for the distance between two

Figure 1: Moving average example, where the red line indicates MA relative to the candlestick price statistic.

extreme points on the chart. Fibonacci ratios are frequently observed to be correlated with basic market maker strategies, where market makers are participants who hold an amount of a particular financial instrument in order to facilitate trading of that instrument. Fibonacci ratios should not be taken to be exact indicators of support / resistance levels. They appear to approximately coincide with points of retracement, but need to be tuned when reading the market given a trader's risk and investment goals [16]. In general, any system that attempts to construct rules of investment based on support and resistance levels need to be capable of adapting and revising the rules as the market conditions will undoubtedly change over time. Many applications for currency trading therefore include Fibonacci-derived levels for retracement, but leave the interpretation of where to make stop loss / take profit decisions to the trader.

Fibonacci ratios (Fibo) may be used to define the position against which a SL or TP order is expressed. Fibonacci ratios or levels for trading (Figure 2) most often involve the use of the following typical cases [16]:

1. Key levels: 0 (0%), 0.236 (23.6%), 0.382 (38.2%), 0.618 (61.8%) and 1 (100%).
2. Other levels: 0.5 (50%) is derived by dividing the number 1 (third number in the Fibonacci sequence) by 2 (forth number in the Fibonacci sequence), 0.764 (76.4%) is derived from the levels 1 and 0.236 as follows: $0.764 = 1 - 0.236$
3. Extension levels: 1.618 (161.8%), -0.618 (-61.8%)

The 0 and 100 levels are identified through recent historical low and high prices (Figure 2). New low or high prices result in a recalculation of the intervening Fibonacci levels in a high-to-low or low-to-high trend. Figure 2 shows Fibonacci levels between 0 and 100 drawn through recent significant low and high prices. Note how prices appear to drift down / up to these levels before 'pushing' through. One can also note that a strong retracement occurs after the first instance of the Fibonacci level of 100. A red line joins the lowest to the maximum price point that are used to establish the Fibonacci levels.

2.3 Pivot point

The Pivot point (P) is the average of the *High* (H), *Low* (L) and the *Close* (C) prices of the previous trading session. This information is used to provide candidate support and resistance levels that then can be utilized while setting the SL and TP orders (Figure 3). Typical definitions for pivot point and corresponding support and resistance levels are defined as follows [15]:

- Pivot point: $P = (H + L + C)/3$
- Resistance Level 1 (R1): $R1 = 2 \times P - L$
- Support Level 1 (S1): $S1 = 2 \times P - H$

Figure 2: Fibonacci retracement example.

- Resistance Level 2 (R2): $R2 = P + H - L$
- Support Level 2 (S2): $S2 = P - H + L$
- Resistance Level 3 (R3): $R3 = P - S1 + R2$
- Support Level 3 (S3): $S3 = P - R2 + S1$

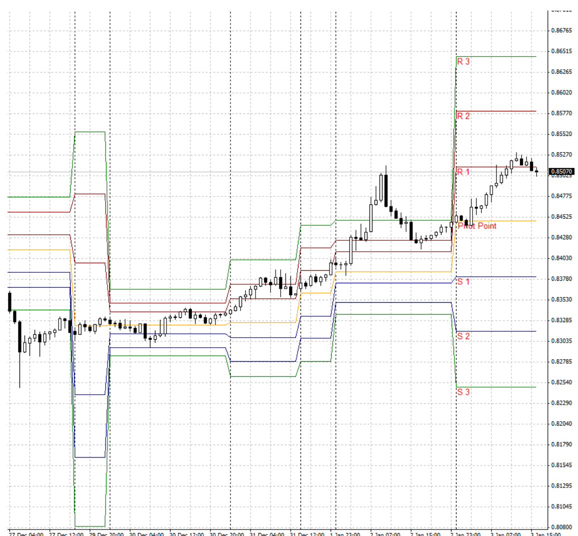

Figure 3: Example of support and resistance levels defined using Pivot Points.

3 FXGP

FXGP represents a framework based on two central innovations [12, 11]. The first of these innovations is that two populations are simultaneously co-evolved under a symbiotic relationship in order to independently evolve representations appropriate for technical indicator, TI, and decision tree, DT. The DT is composed of antecedent—consequent clauses in a manner to be described in this section. Symbiosis implies that individuals from each population cannot exist without the other. Thus, TI on their own are not capable of expressing an investment action (buy, stay or sell) without a DT, and vice versa. Hence, each DT evolves indexes to some subset **of** the TI population and fitness is only evaluated at the level of the DT (Figure 4).

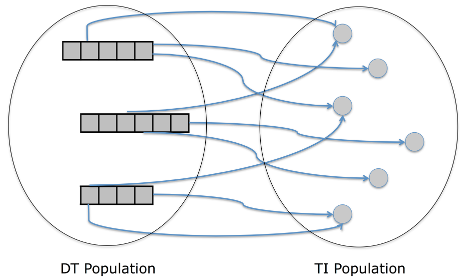

Figure 4: DT—TI interaction. DT population make use of a subset of the available TI. The TI are independently represented in the TI population. The same TI may appear in multiple DT. Fitness is only (directly) expressed for each DT.

The second innovation is that trading criteria are used to define when performance of the current best (champion) policy has became unacceptable. In this case any of three criteria are employed: maximum single drawdown, maximum number of consecutive loss making trades, maximum number of consecutive hours without a sell or buy action. Should the trading criteria flag a champion policy as unacceptable, then the entire population is re-evolved with respect to a new initialization and a three month window defined relative to the re-triggering event. Such a scheme was previously observed to perform better than assuming continuous evolution relative to a sliding window of historical trading data [11]. Moreover, as well as employing historical data to train each new population, a validation partition is used to identify the champion TI—DT for trading (Figure 5).

The FXGP algorithm includes three major parts: Training, Validation and Trading. The Training begins from initialization of the TI population. The number of TI in the population is not fixed and can vary for different generation but the initial size is fixed and is defined by user. As we described in [12, 11] FXGP adopts the linear GP to build unique TI using the set of traditional arithmetic operations (addition, subtraction, multiplication and division). Each TI consists of the header with with the TI's properties (including the reference counter) and the TI's program with assumes the register level transfer language with *N* registers. The original FXGP support TIs of three types: Value, Moving Average or MA and Weighted Moving Average or WMA.

When the TI population is ready, the DT population is initialized. The size of the DT population is fixed. The DT parameters are stored in the DT header and include information about the number of nodes in the DT, size of the SL and TP orders and DT score measured in pips (percentage in point). Pips are one unit of the fourth decimal point of a currency pair, or 1/100th of a cent for dollar currencies. The DT node is represented as a conditional statement where *then* and *else* consequents can be a pointer to the next node or one of the following actions: buy, sell or stay. Each conditional statement can include pointers to TIs in the TI population. Every time a TI is referenced in the DT population its reference counter is incremented.

DT fitness is defined as a score in pips over the training period of time. The subset of *DTgap* DTs with the lowest score is targeted for replacement. The reference counters of linked TIs are decremented and the TIs with 0 value in the reference counters are also discarded.

The *DTgap* parent DTs are randomly selected from the rest of the DT population, cloned and mutated. A DT mutation include one of the following six actions: New conditional function; New *shift* value; New parts of a conditional function; Switch of the part of the conditional function; Switch content of the *then* and *else* clauses; New content of *then* or *else* clause.

TIs associated with a parent DT can also be mutated. In the case of such TI, they are first cloned resulting in the size of the TI population varying, whereas the DT population is of a fixed size.

The Validation step is used (in case of the original FXGP) to check the quality of the DT—TI population and to pick the best DT—TI agent for trading [12]. The trading step is performed until one of the mentioned above retrain criteria is met and then the Train—Validate—Trade cycle (Figure 5) is repeated.

3.1 Representation

In this work we make use of a simplified version of the original FXGP framework (hereafter *s*FXGP), where this reflects a desire to reduce some of the functionality of the original system, particularly with respect to developing the real-time functionality of the approach [13]. With this in mind, *s*FXGP has the following form. Individuals of the TI population utilize the three types of instructions:

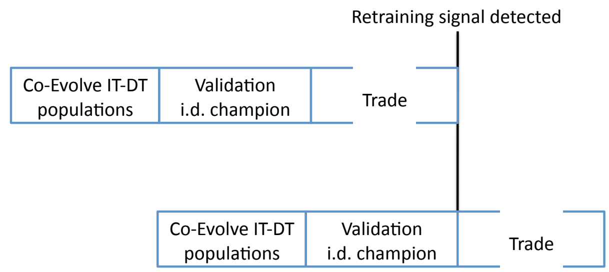

Figure 5: FXGP Train—Validate—Trade cycle. TI-DP populations are first symbolically co-evolved over a fixed duration partition of historical data. A single DT (and supporting TI) is identified from an independent fixed duration partition of validation data. The resulting champion trading agent is then deployed until trading criteria flag a poor trading event. The training and validation partitions are then realigned at the point of failure and the process of evolution repeated.

1. $R[x] \leftarrow R[x] + R[y]$;
2. $R[x] \leftarrow R[x] - R[y]$;
3. $R[x] \leftarrow R[x] \div 2$.

where x and y are integers indexing one of two general purpose registers R. In addition, a mode bit selects the source of $R[y]$ between general purpose register 'y' or an input from the FX trading data. Such data input can take the form of either the (open) price at the current time step 't' or the outcome from a technical indicator, in this case a moving average (Equation 1). In addition, the TI individual can parameterize the MA for its length n and offset relative the current open price, i.e., $t - m$. TIs evolved by sFXGP are therefore distinct from the discussion of Section 2. That is, the three metrics discussed in Section 2 are used to define additional constraints on what constitutes a valid buy or sell order (detailed in Section 3.2), but they have no impact on how TI are evolved by the TI population.

Members of the DT population can only have the form:

- IF $(X_i > Y_i)$ THEN a_j ELSE a_k
- IF $(X_i > Y_i) AND (X_{i+m} > Y_{i+m})$ THEN a_j ELSE a_k

where X_i and Y_i are either 0, the (open) price or a TI selected from the TI population when $X_i \neq Y_i$; and X_{i+m} and Y_{i+m} represent values for X_i and Y_i m previous samples. Consequents a_j and a_k are either a trading action (buy, stay, or sell) or reference a node one level down in the decision tree.

Strongly typed constraints are enforced for the DT conditional statements. For the rule "IF (X_i is 0) THEN Y_i," the values between X_i and Y_i must not include 0. Also, neither X_i nor Y_i can be a price (variable) or describe a TI crossing over the price. For the rule "IF X_i is type 'price' THEN..." cannot be zero or describe a TI that involves crossing 0.

Stop loss orders take the form of a comparison against a threshold, p_{sl}. Thus, if an agent buys at 100 dollars, but the price drops to 90 dollars, then a $p_{sl} = 10$ would trigger selling before any further loss. However, as pointed out in Section 2, much more sophisticated mechanisms exist for constructing SL (or TP) orders. Section 3.2 introduces how we incorporate such mechanisms into the context of FXGP.

3.2 FXGP with sell / buy validation

FXGP as defined above coevolves the technical indicators and decision tree for generating sell, buy or stay actions, plus a limited form of stop loss order. However, whenever a sell or buy action is generated we now include verification against one of four forms of SL or TP order, as follows:

- TP orders. When a trading agent generates a *buy* or *sell* action, FXGP can set a TP order along with the *buy* or *sell* and SL orders. The minimal size of the TP order (s_{min}) is defined by the user. Please see Table 1.
- Fibonacci based SL orders verification mode. In this case the SL order is verified by the Fibo levels. The *buy* and *sell* rules are described in the Table 2.

- Pivot-based SL orders verification mode. In this case the SL order is verified by the Pivot levels. The *buy* and *sell* rules are described in the Table 2.
- MA-based SL orders verification mode. In this case the SL order is verified by the MA. The MA's period is defined by user and the *buy* and *sell* rules are described in the Table 3.

Table 1: Fibonacci and Pivot based TP orders verification modes. $price(t)$ is the current 'open' price; $TP(t)$ is the evolved size of a TP order from the GP individual; $TP(t).level^{low}$ is the nearest Fibo (Pivot) level below $TP(t)$; $TP(t).level^{high}$ is the nearest Fibo (Pivot) level above $TP(t)$; θ is the TP order threshold.

Signal	SL order rule
buy	IF $([price(t).level^{high} - \theta] > [price(t) + tpMin])$ THEN $(TP = price(t).level^{high} - \theta)$ ELSE (!valid trade)
sell	IF $([price(t).level^{low} + \theta] < [price(t) - tpMin])$ THEN $(TP = price(t).level^{low} + \theta)$ ELSE (!valid trade)

Table 2: Fibo and Pivot based SL orders verification modes. $price(t)$ is the current 'open' price; SL is the evolved size of a SL order from the GP individual; $price(t).level^{low}$ is the nearest Fibo (Pivot) level below $price(t)$; $price(t).level^{high}$ is the nearest Fibo (Pivot) level above $price(t)$; θ is the SL order threshold.

Signal	SL order rule
buy	IF $([price(t) - SL] < [price(t).level^{low} - \theta])$ THEN $(SL = price(t).level^{low} - \theta)$ ELSE (!valid trade)
sell	IF $([price(t) + SL] > [price(t).level^{high} + \theta])$ THEN $(SL = price(t).level^{high} + \theta)$ ELSE (!valid trade)

Table 3: MA based SL orders verification mode. An additional test is inserted to check that the price is on the relevant side of the moving average. $MA(t)$ is the scalar moving average at time step 't' as estimated by equation (1); SL is the evolved size of a SL order from the GP individual; θ is the SL order threshold.

Signal	SL order rule
buy	IF $((price(t) > MA(t))$ AND $([price(t) - SL] < [MA(t) - \theta]))$ THEN $(SL = MA(t) - \theta)$ ELSE (!valid trade)
sell	IF $((price(t) < MA(t))$ AND $([price(t) + SL] > [MA(t) + \theta]))$ THEN $(SL = MA(t) + \theta)$ ELSE (!valid trade)

3.3 Parallel populations of FXGP

We also consider the case of evolving an *ensemble* of FXGP champions [13]. Specifically, in order to evolve k sets of TI–DT populations in parallel, all k TI–DT populations are first trained on the designated training partition, Na_t (see Figure 6). An independent validation partition, Na_v, identifies the subset of TI–DT individuals, p^*, from each population that satisfy validation criteria. A new training partition, Nt_t, is now identified (potentially overlapping with both Na_t and Na_v) and used to perform a final cycle of evolution on the subset, p^*. Throughout this process all individuals from p^* are treated independently. This process is repeated in parallel for the k independent TI–DT populations. A single champion is then identified relative to Nt_t from each of the k independent TI–DT populations at the last generation. A team, T, is thus defined consisting of k TI–DT champions. The following rule is now assumed for combining the outcome from each team member [13]:

Table 4: FXGP parameters

Parameter	Value
Training period (N_t)	1,000
Validation period (N_v)	500
Probability of TI mutation (p_{ti})	0.5
Max. training generations (T_{max})	1,000
DT population size (P_{size})	100
# DT replaced per generation (P_{gap})	25
Max. nodes (instr.) per DT (TI)	6
Min (Max) Stop/Loss (s_{min}, s_{max})	$(5, 100)$
Criteria for champion identification	
Proportion with feasible trade (α_v)	0.95
Plateau detection threshold (t_{flt})	200
Retraining criteria	
Max (Min) Drawdown in pips	$(200, 400)$
Max. consecutive losses (n_{loss})	3
Max. consecutive inactive bars (n_{nt})	72
Parameters specific to multi-agent mode	
Tolerance for buy–stay–sell intervals (γ)	2

$$\begin{array}{c}\text{IF } (a \geq \gamma) \text{ THEN } (buy) \\ \text{ELSE IF } (a \leq -\gamma) \text{ THEN } (sell) \text{ ELSE } (stay)\end{array} \tag{2}$$

where the original actions from the k independent TI–DT champions are mapped as either buy = 1, stay = 0, or sell = -1.

We define $a = \sum_{i \in T} a_i$ where $a_i \in \{-1, 0, 1\}$ are the aforementioned mapping of team member actions to scalars. Threshold γ is a static tolerance parameter chosen *a priori* by the user. The generic form assumed for equation (2) has previously been assumed for discretizing the output of a neural network into short and long positions [5] and 'risk management' under boosted decision trees [3].

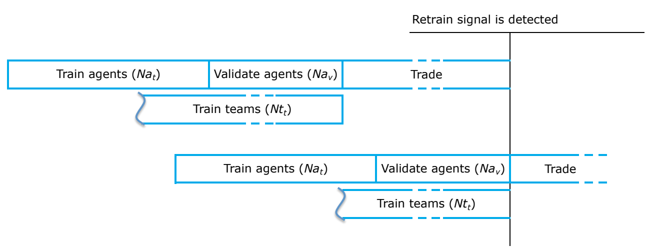

Figure 6: Train–Validate–Train–Trade cycle. Only during the 'trade' cycle are k champions deployed as a team. Trading continues until re-training criteria are triggered by poor trading outcome.

4 Experimental setup

The experimental setup is described below. In all cases *s*FXGP employs the single trading agent mode and uses the same data set, currency pair (EURUSD), and the same parametrization. All parameters were optimized for the 2009 historical rates, as in previous work [12, 11]. Table 4 summarizes parameter settings assumed. To establish the bases for comparison, we simulated[1] the trading activity for the period Jan. 2010 — Nov. 2012.

[1] All runs were performed on a 2.8 GHz iMac computer with Intel Core i7 CPU, 16GB RAM and Mac OS X 10.7.2.

A total of five configurations are considered. The first configuration is the original simplified FXGP in which SL orders are evolved from training data. The second and third configurations use Fibonacci retracement in which two forms for the min–max definition are considered. The motivation behind these two configurations are that the close for a given candle statistic is more robust than the high / low price swing. The remaining configurations assume MA and Pivot TIs respectively (Section 3.2). The configurations involve particular thresholds and associated rules:

1. *s***FXGP mode:** Unmodified version of FXGP, hence SL orders are limited to a simplistic threshold comparison.

2. *s***FXGPF***hl* **mode:** The SL orders are verified by the Fibo levels (Table 2). The 0 and 100 Fibo levels are set to the recent significant Swing **Low** and Swing **High** prices (Figure 2). The TP order is placed 15 pips below the 161.8% level ("buy" signal) or 15 pips above the -61.8% level ("sell" signal). If the difference between the trading order ("buy" or "sell") price and the TP order is less than *tpMin*, then the trading signal is ignored and trading order is not opened.

3. *s***FXGPF mode:** The SL orders are verified by the Fibo levels (Table 2). The 0 and 100 Fibo levels are set to the recent significant Swing **Close** prices (Figure 7). The TP order is placed 15 pips below the 161.8% level ("buy" signal) or 15 pips above the -61.8% level ("sell" signal). If the difference between the trading order ("buy" or "sell") price and the TP order is less than the threshold *tpMin*, then the trading signal is ignored and trading order is not opened.

4. **Pivot mode:** The SL orders are verified by the Pivot levels (Table 2). The TP order is placed 15 pips below the R3 resistance level ("buy" signal) or 15 pips above the S3 support level ("sell" signal). If the difference between the trading order ("buy" or "sell") price and the TP order is less than *tpMin*, trading signal is ignored and trading order is not opened.

5. **MA mode:** The SL orders are verified by the MA (Table 3). The MA periods are set to 48 bars (MA48), 72 bars (MA72) or 96 bars (MA96). TP orders performed significantly worse, so for clarity they are not reported here.

Figure 7: Fibonacci retracement. The 0 and 100 levels are drawn through recent significant Swing Close prices.

5 Results

The results of all experiments are summarized in Table 5. Each experiment includes 100 simulations. Results are ranked in terms of the median number of pips[2] accumulated at the conclusion of the trading period. A Student's T-test (Table 6) confirms the independence of the distributions relative to the top ranked configuration. It is clear

[2]Under the EURUSD market, a move of 0.0001 is equivalent to one pip.

that validation of buy / sell orders using specific configurations of MA and Fibonacci derived levels was more effective than any other scenario. Moreover, MA is more sensitive to the specific parameterization assumed for the length of the moving average calculation. Conversely, the only design decision playing a role in the definition for the Fibonacci levels is the statistic, i.e., either close or high/low) used to configure the 0 and 100 percent levels (compare Figure 2 with Figure 7). Employing the 'swing close' prices from the summary statistic of the candle is typically more robust than swing high or swing low prices. It is also clear that basing the TI on levels identified by pivot points is universally ineffective, albeit with respect to this currency pair and period.

The average number of trades per run and SL orders' statistics are summarized in the Table 7 for each mode in Section sec:setup. Both of the top two TI two configurations – *s*FXGPF and MA72 – perform more valid trades than any other configuration other than Pivot based TI. However, the number of *triggered* stop losses (as a fraction of the number of trades) under *s*FGPF is significantly lower than any other configuration. This implies that the rules generated from this parameterization relied less on the 'corrective' effect of SL orders. Moreover, we believe that the Fibonacci levels based on the close prices were able to filter more of the noise effects than Fibonacci levels initialized under high–low prices. All adaptive schemes for validating SL orders resulted in a smaller SL order than the original SL thresholding scheme (*s*FXGP) and therefore lost less through SL orders.

The champion configuration *s*FXGPF was deployed as an ensemble of 3 trading agents (FXGPFT 3) and the results of 100 simulations were compared with the unmodified version of FXGPT 3 (Section 3.3). The results are summarized in the Table 8. Comparison with Table 5 demonstrates that the ensemble version pushes the tail of the distribution up in each case. This makes the resulting trading agents less sensitive to initial conditions (more dependable) as, for example, 98 percent of the FXGPFT 3 runs now avoid recording a loss over the trading period. Figure 8 illustrates this phenomena in terms of a combined violin / box plot summarizing the distribution of cumulative pips for all 100 runs for each of the pairs of trading agent. Figure 9 provides an illustration of the cost of training / retraining under single and ensemble frameworks. Given that trading information characterizes 1 hour intervals, we note that both forms of the algorithm operate in 'real-time'.[3]

Table 5: Single trading agent comparison. Results are sorted with respect to the median scores. † indicates the runs that are illustrated by the distribution of Figures 8 and 9.

Algorithm's mode	Profitable runs (%)	Score (pips)				
		min	1st quartile	median	3rd quartile	max
*s*FXGPF†	96	-992.6	1145.4	1846.9	2632.8	4473.9
MA72	90	-1192.6	766.3	1419.5	2121.4	3491.5
*s*FXGPF*hl*	80	-2217.2	130.7	1189.9	1843.1	4146.1
*s*FXGP†	74	-3153.8	-94.8	988.5	1917.8	4054.6
MA96	94	-1790.6	328.6	827.8	1647.3	3528.6
MA48	71	-2540.7	-208.1	533.5	1334.4	2827.3
Pivot	49	-3280.8	-715.2	-66.1	395.5	2262.8

Table 6: $p - values$ for pairwise Student T-test.

*s*FXGPF vs MA72	*s*FXGPF vs *s*FXGPF*hl*	*s*FXGPF vs *s*FXGP	*s*FXGPF vs MA96	*s*FXGPF vs MA48	*s*FXGPF vs Pivot
5.56×10^{-3}	2.54×10^{-7}	9.00×10^{-6}	5.78×10^{-9}	8.29×10^{-16}	9.00×10^{-25}

Table 7: SL algorithms comparison. Average number of trades and SL statistics.

Description	*s*FXGPF	MA72	*s*FXGPF*hl*	*s*FXGP	MA96	MA48	Pivot
Average # of trades, per run	558.30	440.08	421.80	428.40	412.04	438.04	484.80
Average # of triggered SL, per run	204.04	208.20	199.48	193.70	195.12	242.12	251.64
Average % of triggered SL, per run	36.55	47.31	47.29	45.21	47.35	55.27	51.91
Average SL order size, pips	44.13	39.38	43.62	73.39	39.59	36.41	38.42

[3]The General Central Dispatch utility available as part of the Apple OS is used to schedule the execution of multiple threads during the ensemble experiments.

Table 8: Ensemble trading agent comparison for $k = 3$. Results are sorted with respect to the median scores. † indicates the runs that are illustrated by the distribution of Figures 8 and 9.

SL Algorithm	Profitable runs (%)	Score (pips)				
		min	1st quartile	median	3rd quartile	max
FXGPFT 3†	98	-511.6	1383.4	2066.8	2647.5	3960.9
FXGPT 3†	81	-1876.6	288.5	1489.2	2462.8	4362.4

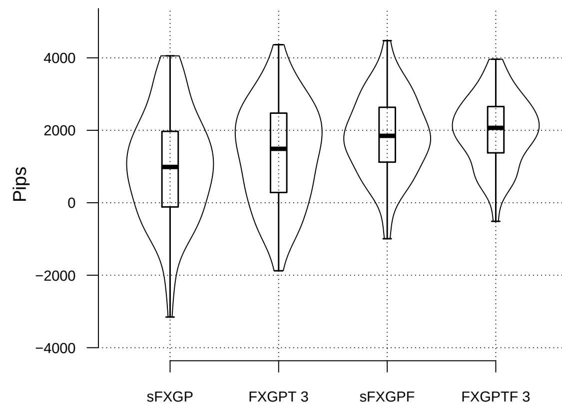

Figure 8: Distribution of cumulative scores in pips over trading simulation period of time for single (*s*FXGP and *s*FXGPF) and ensemble of 3 champion trading agents.

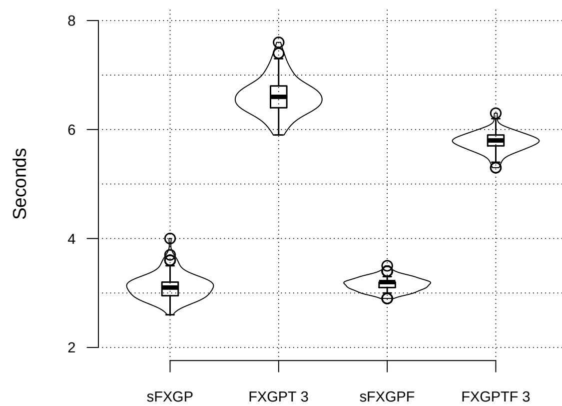

Figure 9: Distribution of average training times (over a run) per population (*s*FXGP and *s*FXGPF) and per team of three populations (FXGPT 3 and FXGPTF 3).

Table 9: *s*FXGPF annual results. 2013 vanilla and 2013 optimized† lines include the first 10 months of the year 2013. 2013 optimized† line shows result after optimization of the algorithm parameters based on the 2012 year historical rates.

Year	Profitable runs (%)	Score (pips)				
		min	1st quartile	median	3rd quartile	max
2010	100	222.9	1165.6	1671.6	2167.9	3570.2
2011	84	-979.4	177.7	675.2	1200.8	3127.2
2012	0	-1842.4	-1204.2	-946.5	-680.2	-58.7
2013	25	-1048.2	-429.9	-180.3	-4.4	481.3
2013†	81	-628.2	83.7	463.5	754.6	1757.8

A final experiment is performed with the winning *s*FXGPF configuration. Previous practice in the academic literature has been to describe performance of an investment strategy as the cumulative profit / loss collected over

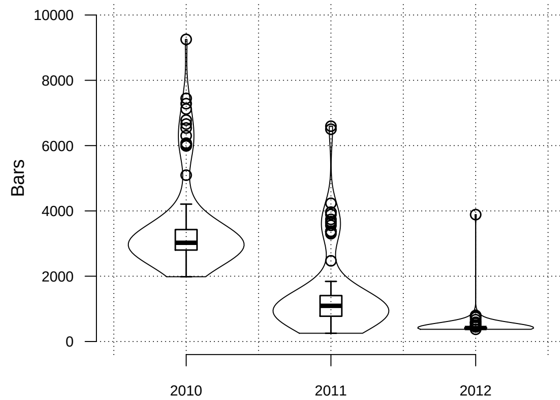

Figure 10: Distribution of the first time occurrences of the drawdown over trading simulation periods of time starting from January 2010 (2010), January 2011 (2011) and January 2012 (2012). 500 bars period is approximately equivalent to a month.

a total investment period. This means that early successes can potentially mask latter losses. See for example, the first four rows of Table 9. With this in mind a trader actually using FXGP in practice might periodically revisit the parameterization using historical data. Thus, the final row of Table 9 illustrates the effectiveness of re-parameterizing FXGP using the data from 2012 and deploying this during 2013.

6 Conclusion

This work examined the usage of three different techniques for identifying retracement opportunities in the FXGP framework, namely moving average, pivot points, and Fibonacci ratios. The strategies based on the Fibonacci series clearly had the best performance of the three techniques. Moreover, Fibonacci retracement with 0% and 100% levels drawn through the Swing Close prices (*s*FXGPF) is more efficient than Fibonacci retracement with 0% and 100% levels drawn through the Swing Low and Swing High prices for FXGP. The use of Fibonacci retracement to define the SL order position increased the median score by 86.7%, increased the percent of profitable trades by 29.7% (Table 5), and reduced the average size of a SL order by 51% (Table 7) compared to the previous FXGP version. The use of Fibonacci retracement to define the SL order position reduced the retraining time of the team of three trading agents (Figure 9) by 12%. This reduction likely occurs because the use of Fibonacci levels to define the SL order position increases chances of a TI–DT population passing the validation process. Of the different modes of FXGP, *s*FXGPF had the biggest average number of trades per simulated period of time and the lowest percent of triggered SL orders (Table 7). Teams of trading agents were confirmed to be more efficient than single trading agents (Table 8). In future work, we plan to combine drawdown criteria (Figure 10) that triggers FXGP retraining with parameter optimization based on the newest real trading activity to further improve performance. Self adaptation of parameters would also be of interest, for it appears that the variation in year-to-year drawdown values might be particularly promising for triggering re-parameterization (Figure 10).

Acknowledgment

The suggestions of reviewers in improving this work are gratefully acknowledged. The authors are also thankful for the continued support from the NSERC CRD program (Canada) and RUAG Schweiz AG (Switzerland) while conducting this research.

References

[1] Matthew Butler and Dimitar Kazakov. Particle swarm optimization of Bollinger Bands. In *International Conference on Swarm Intelligence*, volume 6234 of *LNCS*, pages 504–511, 2010.

[2] Matthew Butler and Dimitar Kazakov. A learning adaptive Bollinger Band system. In *IEEE Conference on Computational Intelligence on Financial Engineering & Economics*, pages 1–8, 2012.

[3] G. Creamer. Model calibration and automated trading agent for Euro futures. *Quantitative Finance*, 12(4):531–545, 2012.

[4] M. A. Dempster and C. M. Jones. A real-time adaptive trading system using genetic programming. *Quantitative Finance*, 1:397–413, 2001.

[5] C. L. Dunis, J. Laws, and G. Sermpinis. Higher order and recurrent neural architectures for trading the EUR / USD exchange rate. *Quantitative Finance*, 11(4):615–629, 2011.

[6] P. Fernandez-Blanco, D. Bodas-Sagi, F. Soltero, and J. Hidalgo. Technical market indicators optimization using evolutionary algorithms. In *ACM Conference Companion on Genetic and Evolutionary Computation*, pages 1851–1858, 2008.

[7] A. Hirabayashi, C. Aranha, and H. Iba. Optimization of the trading rule in foreign exchange using genetic algorithm. In *ACM Conference on Genetic and Evolutionary Computation*, pages 1529–1536, 2009.

[8] A. Hryshko and T. Downs. An implementation of genetic algorithms as a basis for a trading system on the foreign exchange market. In *IEEE Congress on Evolutionary Computation*, pages 1695–1701, 2003.

[9] H. Iba and C. C. Aranha. *Practical applications of evolutionary computation to financial engineering*, volume 11 of *Adaptation, Learning, and Optimization*. Springer, 2012.

[10] J. Leung and T. Chong. An empirical comparison of moving average envelopes and Bollinger Bands. *Applied Economics Letters*, 10(6):339–341, 2003.

[11] Alexander Loginov and Malcolm I. Heywood. On the impact of streaming interface heuristics on GP trading agents: an FX benchmarking study. In *Proceeding of the ACM Genetic and Evolutionary Computation Conference*, pages 1341–1348, 2013.

[12] Alexander Loginov and Malcolm I. Heywood. On the utility of trading criteria based retraining in Forex markets. In *Applications of Evolutionary Computation (EvoFIN)*, volume 7835 of *LNCS*, pages 192–202, 2013.

[13] Alexander Loginov and Malcolm I. Heywood. On evolving multi-agent FX traders. In *Applications of Evolutionary Computation (EvoFIN)*, volume 8602 of *LNCS*, pages 203–214, 2014.

[14] Tony Loton. *Stop orders: a practical guide to using stop orders for traders and investors*. Harriman House, 2009.

[15] J. L. Person. *Candlestick and Pivot Point Trading Triggers: Setups for stock, Forex, and futures markets*. John Wiley & Sons, 2007.

[16] Boris Schlossberg. *Technical analysis of the currency market*. Wiley trading. John Wiley & Sons, Inc., 2006.

[17] G. Wilson and W. Banzhaf. Interday and intraday stock trading using PAM GP and linear GP. In A. Brabazon, O'Neill, and D. G. Maringer, editors, *Natural Computing in Computational Finance 3*, volume 293 of *SCI*, pages 191–212. Springer, 2010.