A Learning Adaptive Bollinger Band System

Matthew Butler and Dimitar Kazakov

***Abstract*—This paper introduces a novel forecasting algorithm that is a blend of micro and macro modelling perspectives when using Artificial Intelligence (AI) techniques. The micro component concerns the fine-tuning of technical indicators with population based optimization algorithms. This entails learning a set of parameters that optimize some economically desirable fitness function as to create a dynamic signal processor which adapts to changing market environments. The macro component concerns combining the heterogeneous set of signals produced from a population of optimized technical indicators. The combined signal is derived from a Learning Classifier System (LCS) framework that combines population based optimization and reinforcement learning (RL). This research is motivated by two factors, that of non-stationarity and cyclical profitability (as implied by the adaptive market hypothesis [10]). These two properties are not necessarily in contradiction but they do highlight the need for adaptation and creation of new models, while synchronously being able to consult others which were previously effective. The results demonstrate that the proposed system is effective at combining the signals into a coherent profitable trading system but that the performance of the system is bounded by the quality of the solutions in the population.**

I. Introduction

Technical analysis is based on the premise that history tends to repeat itself and therefore past patterns can be used for predictive purposes [8]. This paper introduces a novel forecasting algorithm that is a blend of micro and macro modelling perspectives when using Artificial Intelligence (AI) techniques. The micro component concerns the fine-tuning of technical indicators with population based optimization algorithms. This entails learning a set of parameters that optimize some economically desirable fitness function as to create a dynamic signal processor which adapts to changing market environments. The macro component concerns combining the heterogeneous set of signals produced from a population of optimized technical indicators. The combined signal is derived from a Learning Classifier System (LCS) framework that combines population based optimization and reinforcement learning (RL). This research is motivated by two factors, that of non-stationarity and cyclical profitability (as implied by the adaptive market hypothesis [10]). The non-stationary nature of the stock market denotes a signal that has moments (such as the μ and σ^2) which are not constant in time. This implies that trading models will have to continually adapt to changing market conditions to remain profitable. However, cyclical profitability (CP), implies that the performance of trading models constructed from historical information will wax and wane with the current market conditions. Thus, experienced trading models will still contain useful information for forecasting. These two properties are not necessarily in contradiction but they do highlight the need for adaptation and creation of new models, while synchronously being able to consult others which were previously effective.

Matthew Butler and Dimitar Kazakov are with the Artificial Intelligence Group in the Department of Computer Science, The University of York, Deramore Lane, Heslington, York, UK (email: {mbutler, kazakov}@cs.york.ac.uk).

Using technical analysis with AI techniques has been a popular research topic and several methods have been shown to be effective. Some of the initial work that gained recognition was by Neely et al. [12], where the authors used genetic algorithms (GA) to find trading rules for the foreign exchange market. The results demonstrate that GAs were able to identify solutions which could outperform the market. Another notable contribution in this area is by Tsang et al. [9], where a forecasting tool called EDDIE (Evolutionary Dynamic Data Investment Evaluator) was developed which originally learned functions for forecasting the market based on technical indicators and tree-based genetic programming. EDDIE has been extensively tested and shown to produce trading strategies which can outperform a passive buy and hold approach.

The contributions of this paper are: (1) to provide a framework that extends the research of optimizing technical analysis with AI techniques to a combined signal approach, (2) extend an existing Particle Swarm Optimization algorithm to a multi-objective optimization, (3) to augment the LCS framework for online learning when the population of solutions are not classifiers, and (4) to extend the idea of cyclical profitability from the adaptive market hypothesis (AMH) into a white-box forecasting tool. The following sections will describe the various components of the Learning Adaptive Bollinger Band System (LABBS) and review its forecasting performance against some relevant benchmarks.

II. Learning Adaptive Bollinger Band System Overview

A major difference between LABBS and other LCS (even those that used technical indicators) is that the population does not contain classifiers. Many technical indicators do not have a "next tick" prediction of the asset price but provide signals for the future trend. As such, the canonical way of combining forecasts and updating the population must be changed to accommodate this new strategy.

LABBS is a blend of individual technical indicators which create a population [P] and a meta agent (MAG) that considers the behaviour and signals from the population to form a coherent investment strategy. The two components are only loosely coupled where the decisions made by the meta agent do not effect [P] directly. The meta agent uses the signals produced by [P] along with their meta parameters to infer its own investment decisions. The algorithm has three main

components: discovery, performance, and reinforcement. The discovery component concerns the optimization of ABBs based on an economically desirable fitness function(s). This component creates the initial population as well as new additions as the system evolves. The fine-tuning of the ABB parameters is performed with particle swarm optimization which will be discussed in section III-B. The performance component is responsible for indentifying ABBs in [P] that $match$ the current market environment and generating a prediction array for choosing an action. The actual matching criteria will be discussed in section II-C. The final component of reinforcement has been substantially reduced from its role in a traditional LCS and is only used to update the one meta parameter of [P] instrumental in forming the prediction array. Figure 1 depicts a typical iteration of the LABBS algorithm, which cycles through the following steps:

1) The environment at time t generates a signal S_t that is used as input to the population [P].
2) The parameters of the population [P] are updated and compared to the input to identify matches.
3) A match set $[M]_t$ is created.
4) If $|[M]_t| < MS_{min}$, then the covering algorithm attempts to create solutions to match S_t
5) The meta parameters of ABBs in the relative complement of $[M]_t$ in $[M]_{t-1}$ are updated ($[M]_{t-1} \setminus [M]_t$)
6) From $[M]_t$ a prediction array is formed, from which an action a_i is selected
7) The action a_i is performed in the environment.
8) A reward is received from the environment which is used to update the investment position of the MAG

In step 4, MS_{min} is the minimum number of ABBs required in the match set and in step 5, the relative complement of $[M]_t$ in $[M]_{t-1}$ are the ABBs that no longer meet the minimum criteria to be included in the match set but which did in the previous time step. In the following sections a more detailed description of the components is supplied.

A. Step 1 - The Environment

The environment can be any system represented by a time-series, for this study the environment is simulated or real-world financial time-series. A signal sent from the environment to the population [P] is a subsample of historical data for identifying ABBs that are effective in the current market environment.

B. Step 2 - The Population [P]

For a traditional LCS implementation an initial population can be randomly or selectively chosen. However, based on the implication of cyclical profitability, these experiments seed the initial population with optimized ABBs created from a training set that is supplied at the beginning of the experiment. The initial population contains N ABBs with representations from different training window sizes. Thus the population contains a spectrum of ABBs that were trained for very specific time-periods to much more general market environments. Each ABB in [P] is associated with three meta parameters: reputation, duration, and effectiveness, symbolized by rep, D and E, respectively. Reputation is an expectation of how long an ABB will remain effective (meet the match set criteria) after it becomes a member of [M]. Duration is the current count of how long the ABB has been present in [M] and is used, along with reputation, to construct the prediction array. Finally, effectiveness is a count of the total number of times an ABB has been in [M], this parameter is used for deleting ABBs from [P] that never prove to be useful. The results from Butler et al. [2] suggest that a proportion of ABBs from the seeding period will never again (or rarely) be effective in the market. The deletion probability of an ABB is inversely related to its effectiveness divided by the number of input signals received. When a new data point is added to the population, the ABBs are updated using the equations discussed in section III-A1.

C. Step 3 - The Match Set [M]

Generally in a LCS the match set is created from classifiers whose antecedents match the input from the environment. For LABBS the match set is constructed from ABBs that satisfy the performance criteria determined by the user. The performance criteria are based on percentage return, trading activity and accuracy. The accuracy of an ABB is calculated as the number of trades that were profitable divided by the total number of trades, where consecutive days that predict the same position are considered components of one trade. For example a trade history of {long, long, long, short, short, neutral} would be 2 trades over a 6-day time period.

D. Step 4 - Covering

The covering step is used when the match set contains less ABBs then the minimum required in [M]. When this step is called a set number of ABBs are created for a randomly selected subsample of historical data. The size of the data set is random but has a fixed end-point which is the current day.

E. Step 5 - Updating Reputation

An integral component of the process of choosing an action are the reputations of the ABBs in [M]. The reputation gives an estimate of how reliable the signal is. When an ABB is created and added to [P] during the seeding period a reputation is assigned that represents how long it would have remained in [M] after the time period is was optimized for. To allow for online learning this parameter is updated after each stint in [M] using the Widrow-Hoff delta rule with a learning rate parameter β:

$$rep_m \leftarrow rep_m + \beta(D_m - rep_m) \tag{1}$$

where, $0 < \beta \leq 1$, rep_m is the reputation of the m^{th} ABB and D_m is its most recent duration in [M]. Equation 1 allows for an ABBs reputation to increase at times when it is more effective and decrease when its signals are less reliable.

F. Step 6 - Action Selection

From [M] a $prediction\ array$ is constructed which determines an action (a_i) to be performed by the system. To construct the $prediction\ array$ the system forms a

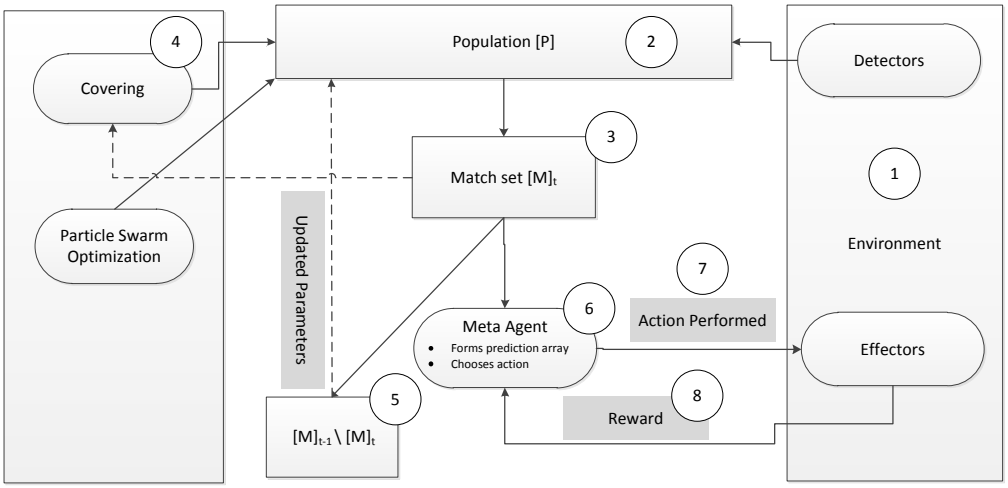

Fig. 1. LABBS framework - the values 1-7 indicate the typical steps included in a single learning iteration of the system. Dashed lines indicate steps that may not occur every iteration.

system prediction $P(a_i)$ for each a_i represented in [M]. The calculation of $P(a_i)$ is based on reputation, return and accuracy of the ABBs advocating a_i, this is similar to the fitness-weighted method described in Wilson [14]. Determining the action to be selected can be based on a variety of methods. For this study, deterministic action selection was used, which selects the a_i with the largest $P(a_i)$. The three possible actions the system can perform are: Long, Short, and R_f, where R_f is a risk-free rate that the system can opt to invest in if a position in the market cannot be determined. The equation for calculating $P(a_i)$ is:

$$P(a_i) = \underbrace{\left[\sum_{j=0}^{m} r_j \left(\frac{Rem_j}{\sum_{j=0}^{m} Rem_j}\right)\right]}_{reputation-weighted\ return} \times \underbrace{\left[(\kappa_j/\kappa_{avg})\right]}_{relative\ accuracy} \quad (2)$$

The first component of the equation is a reputation weighted return, where m is the number of ABBs advocating action a_i in [M], the return (r_j) is the percentage return of j^{th} ABB for the match period. Rem_j is the difference between the ABBs reputation and its current duration (D_j) in the match set, so:

$$Rem_j \leftarrow \begin{cases} rep_j - D_j & \text{if } rep_j > D_j \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

reducing the reputation by the current duration takes into account the confidence the system has for a particular ABB given that it will only be effective for a given duration based on cyclical profitability.

The second component is an accuracy multiplier, where κ_j is the directional accuracy of the j^{th} ABB and κ_{avg} is the average accuracy in [M]. The accuracy multiplier allows for an extra dimension to the system prediction that considers accuracy as well as return.

G. Steps 7 & 8 - action and reward

The action (a_i) is performed in the environment and a reward is received. The reward is a percentage return over the investment time-horizon (set to a 1-day return for this study).

III. Creating Adaptive Bollinger Bands

One of the most important components of LABBS is the population of solutions that is used to derive the investment signals. The quality of these solutions will dictate the effectiveness of the overall strategy. Based on the implication of cyclical profitability as described by Lo in the adaptive market hypothesis [10] and the results from Butler et al. [2] which demonstrate this property, this study generates an initial population of ABBs that in theory should be profitable, on average, in 20% of the out-of-sample data[1]. Therefore this approach is dependent on the ability of a meta agent to identify ABBs that will generate the best estimate of future market behaviour. It is worth noting that in this paper we are using BBs as the underlying technical analysis tool but any technical indicator that has parameters which can be tuned to current market conditions could have been used.

A. Adaptive Bollinger Bands

The ABBs were initially developed because, despite their popularity, the recent academic literature had shown Bollinger Bands (BB) to be ineffective [7]. However, through PSO-based parameter fine tuning the indicator could be improved and outperform the market index under certain market conditions. The three main components of BBs are:

1) An N-day moving average (MA), which creates the middle band,

[1]The results from Butler et al. [2] only tested out-of-sample profitability for a period of 5 years after the ABBs were optimized and therefore does not demonstrate cyclical profitability beyond this time horizon.

2) an upper band, which is the MA plus k times the standard deviation of the middle band, and
3) a lower band, which is the MA minus k times the standard deviation of the middle band.

The default settings for using BBs are a MA window of 20 days and a value of k equal to 2 for both the upper and lower bands. When the price of the stock is trading above the upper band, it is considered to be overbought, and conversely, an asset which is trading under the lower band is oversold. The trading rules generated from using this indicator are given by equations 4–5:

$$Buy : P_i(t-1) < BB_n^{low}(t-1) \& P_i(t) > BB_n^{low}(t) \quad (4)$$

$$Sell : P_i(t-1) > BB_n^{up}(t-1) \& P_i(t) < BB_n^{up}(t) \quad (5)$$

Essentially, the above rules state that a buy signal is initialized when the price (P_i) crosses the lower band from below, and a sell signal when the price crosses the upper band from above. In both cases the trade can be closed out when the price crosses the middle band.

To allow for efficient online optimization of the BBs we define two new forms of the traditional indicator, running and exponential BBs, that make use of estimates of the 1^{st} and 2^{nd} moments of the time series.

1) Running and Exponential Bollinger Bands: We define a BB as:

$$BB = MA_n \pm k \times \sigma(n_{period}) \quad (6)$$

where MA_n is an n-day moving average and σ is the standard deviation. Then a Running Bollinger Band that makes use of estimates of the 1^{st} and 2^{nd} moments is:

$$BB = A_n \pm k \times J_n(B_n - A_n^2)^{1/2} \quad (7)$$

where,

$$A_n = \frac{1}{n}\sum_{i=1}^{n} Y_i \quad , \quad B_n = \frac{1}{n}\sum_{i=1}^{n} Y_i^2 \quad (8)$$

$$J_n = \frac{n}{n-1}^{1/2} \quad (9)$$

where the normalization factor J_n allows for an unbiased estimate of the σ and Y_i is i^{th} data point. From this, recursive updates of the BBs can be performed as follows:

$$A_n = \frac{1}{n}Y_n + \frac{n-1}{n}A_{n-1} \quad (10)$$

$$B_n = \frac{1}{n}Y_n^2 + \frac{n-1}{n}B_{n-1} \quad (11)$$

For the exponential form we define the BB on a time scale η^{-1}. Where incremental updates of the estimates are:

$$A_n = \eta Y_i + (1-\eta)A_{n-1} \quad (12)$$

$$B_n = \eta Y_i^2 + (1-\eta)B_{n-1} \quad (13)$$

and the normalization factor becomes:

$$J_n = \frac{1-\eta/2}{1-\eta} \quad (14)$$

TABLE I
THE PARAMETERS FOR PSO FINE-TUNING. MA STANDS FOR MOVING AVERAGE. PARTICELS IS THE NUMBER OF PARTICLES FROM AN INDIVIDUAL IN THE SWARM ALLOCATED FOR THAT PARAMETER.

Description	Particles
The value for calculating the upper/lower band.	2/2
Window size for the upper/lower band MA.	5/5
The type of ABB to use for upper/lower band.	1/1
The stop loss for short-sells/buys.	2/2

This implementation of the ABBs optimizes eight parameters displayed in table I. A result from [1] concluded that BBs are ineffective at generating profits when the market is trending, due to the exiting of profitable trades prematurely. To counteract this consequence of using the middle band (the N day MA) to initiate the exiting of a trade, this implementation uses trailing stop-losses.

An advantage to using BBs as the underlying technical analysis tool is that we are able to tap into a common heuristic used by traders of identifying turning points in stock prices. The identification of an overbought/oversold security signals a correction and therefore a change in directional movement.

B. Particle Swarm Optimization

PSO [5] is a population based algorithm inspired from swarm intelligence commonly used in optimization tasks. PSO was chosen for the original study as it had been shown to be as effective as genetic algorithms when modelling technical trading rules, as in Lee et al. [6], yet it had a much simpler implementation and arrived at a global optimum with fewer iterations. Each particle in the swarm represented an n-dimensional position vector that maps to the various parameters displayed in table I. In its canonical form the swarm is governed by the following:

$$\upsilon_{i,j} = \omega \times \upsilon_{i,j} + c_1 r_1 \times (local_{best\,i,j} - x_{i,j}) + c_2 r_2 \times (global_{best\,j} - x_{i,j}) \quad (15)$$

Here $\upsilon_{i,j}$ is the velocity of j^{th} dimension of the i^{th} particle, c_1 and c_2 determine the influence on a particular particle by its optimal position previously visited and the optimal position obtained by the swarm as a whole, r_1 and r_2 are uniform random numbers between 0 and 1, and ω is an inertia term (see [13]) chosen between 0 and 1.

$$x_{i,j} = x_{i,j} + \upsilon_{i,j} \quad (16)$$

Here $x_{i,j}$ is the position of the j^{th} dimension of the i^{th} particle in the swarm. To encourage exploration and limit the speed with which the swarm would converge, a maximum velocity was chosen for each dimension dependent on its range of feasible mappings. In table II the range and maximum velocity for each parameter is displayed. The type of ABB to use was mapped using a wrapper function which evaluated to a running BB if the particle had a value greater than or equal to 0.5 and mapped to a exponential BB if the particle had a value less than 0.5.

TABLE II
THE RANGE OF PARAMETER FEASIBLE VALUES AND CORRESPONDING MAXIMUM VELOCITY. WHERE $\oplus$ SIGNIFIES THAT THE MA TYPE CAN ONLY TAKE ON VALUES OF 0 OR 1.

Parameter	Range	Max Velocity
Upper Band	{-4,4}	0.10
Lower Band	{-4,4}	0.10
MA Type	$\{0 \oplus 1\}$	0.10
Stop Loss	{-0.99,0}	0.10
Window Size	{5,500}	20

1) Heterogeneous Particle Swarm Optimization: This study used a more sophisticated version of PSO called Dynamic Heterogeneous Particle Swarm Optimization (dHPSO) [3] which has been shown to outperform the canonical from of PSO on a variety of optimization problems. With dHPSO the position update remains the same but the calculation of the velocity update is expanded to allow for alternatives. The swarm becomes heterogeneous as each particle in the swarm will have one of five possible velocity update profiles and the swarm is dynamic as the velocity update profile will change if a particle becomes stagnant. The additional velocity updates are as follows:

$$v_{i,j} = \omega \times v_{i,j} + c_1 r_1 \times (local_{best\,i,j} - x_{i,j}) \quad (17)$$

$$v_{i,j} = \omega \times v_{i,j} + c_2 r_2 \times (global_{best\,j} - x_{i,j}) \quad (18)$$

$$v_{i,j} \sim N\left(\frac{local_{best\,i,j} + global_{best\,j}}{2}, \sigma\right) \quad (19)$$

$$v_{i,j} = \begin{cases} local_{best\,i,j} & \text{if U(0,1)} < 0.5 \\ N\left(\frac{local_{best\,i,j} + global_{best\,j}}{2}, \sigma\right) & \text{otherwise} \end{cases} \quad (20)$$

where, N and U are normal and uniform distributions respectively. Equation 17 is the cognitive only profile where the social component has been removed. This promotes exploration as each particle becomes a hill-climber. Equation 18 is the social only profile where the cognitive component has been removed. In effect the entire swarm becomes one large hill-climber. Equation 19 is the Barebones PSO where the position update is the velocity update, so:

$$x_{i,j} = v_{i,j}, \quad and \quad (21)$$

$$\sigma = |local_{best\,i,j} - global_{best\,j}|. \quad (22)$$

Finally, equation 20 is the modified Barebones profile. One additional improvement has been made to the dHPSO algorithm where particles that continue to be stagnant after velocity profile changes will be re-initialized randomly in the solution space. This modification was shown to improve the algorithms ability to find solutions that outperform the market index.

2) Multiobjective Optimization: The goal is to have an optimal agent for a particular market segment. However, what makes an agent optimal for this study is not simply a profit maximization but also trading activity, where agents generating a greater number of trades are preferred. This is because LABBS requires signals to make decisons. This optimization (of profitability and trade activity) is achieved by extending the dHPSO algorithm to multi-objectives using the dynamic neighbourhood approach of Hu and Eberhart [4]. This multi-objective optimization (MOO) approach is simple and straight forward but has been proven effective when the number of objectives is 3 or less. The algorithm works by optimizing objectives one at a time and executing the following steps:

1) Calculate the distances between each particle in the swarm in the fitness value space of the first objective (profit).
2) Secondly, locate the m closest particles for each particle in the swarm, these m particles form the local neighbourhood.
3) Find the local optima among the neighbours in terms of the fitness-value of the second objective (number of trades). This local optima is used as the $local_{best}$ position in the above equations.

In this study the neighbourhood size was set to 3, so each particle was influenced by 2 neighbours, this creates a ring structure for the neighbourhood topology. To calculate the profitability of an ABB the following equation was used:

$$fitness_i = \sum_{t=1}^{T} capital_t \times \frac{(P_{1,t} - P_{0,t})}{P_{0,t}} - (\tau \times capital_t) \quad (23)$$

where $fitness_i$ is the fitness of the i^{th} particle in the swarm, τ represents the transaction costs, T is the total number of trades, and P_0 and P_1 are the entering and exiting price for the underlying asset. The profit for each trade is the rate of return multiplied by the capital invested minus the transaction cost which is also a function of the amount of capital invested.

IV. EMPIRICAL RESULTS

A. Trader Simulation

1) Data: The simulated data is generated to be a random walk with autoregressive trend processes, as discussed in Moody et al. [11]. The two-parameter model is:

$$p_t = p_{t-1} + \beta_{t-1} + \kappa\epsilon_t \quad (24)$$

$$\beta_t = \alpha\beta_{t-1} + \nu_t \quad (25)$$

where α and κ are constants, and ϵ_t and ν_t are normal random variables with zero mean and unit variance. The simulated price process is:

$$z_t = exp\left(\frac{p_t}{R}\right) \quad (26)$$

where R is the range of p_t: $max(p_t) - min(p_t)$ over a simulation of 10,000 samples.

2) Long/Short Trader of Single Security: In this section we present results from two simulated time-series where κ = 3 and α = 0.9. This configuration creates a series with a significant amount of noise and is trending on short time-scales. Example plots of the time-series can be seen in the top panels of figures 2 and 3. The results from the LABBS trading system under different [M] criteria is compared to a buy-and-hold approach, the optimized agents in [P] and a Bollinger Band with the canonical settings. In each time step

TABLE III
THE MATCH SET CRITERIA FOR EACH EXPERIMENT RUN FOR LABBS. TRADES, ACCURACY AND RETURN REPRESENT THE MINIMUM LEVEL REQUIRED. THE ONLY PARAMETERS THAT CHANGE ARE THE MINIMUM RETURN AND THE SIZE OF THE WINDOW FOR EVALUATING RECENT PERFORMANCE.

Case	Window Size (days)	Trades	Accuracy	Return	$[M]_{min}$
1	100	1	0.70	0.0500	1
2	50	1	0.70	0.0250	1
3	25	1	0.70	0.0125	1
4	10	1	0.70	0.0100	1

TABLE IV
THE VARIOUS PARAMETER SETTINGS FOR THE PSO ALGORITHM.

Parameter	Value	Parameter	Value
population size	30	iterations	100
ω	0.99	neighbours	2
c_1	2	c_2	2

LABBS can either be in a short/long or risk-free position. Table III displays the match set [M] criteria for each of the experiment runs. The value for β in equation 1 was set to 0.10, the population size was set to 500 and the window sizes for training the ABBs were $\{50, 100, 250, 500$ and $750\}$ days. The parameter settings for the PSO algorithm are in table IV. Additionally, a transaction cost of 0.25% is applied to all trades exiting and entering the market.

3) Simulated Trading Results: From figures 2 and 3 we can observe that LABBS was able to outperform each of the benchmarks in terms of cumulative return. The first simulated series (figure 2) experiences a strong upward trend in the beginning and during this time the buy and hold approach is able to outperform some of the LABBS models. However, when the trend changes and the simulated market begins to contract each of the LABBS models are able to increase profits by taking short or risk-free positions. The second series experiences a negative trend mid way, where once again the LABBS models were able to capitalize. The disparity between LABBS and the benchmarks is more prevalent in this series as it had more major trend changes.

The frequency histograms in figure 4 reveal that LABBS was able to outperform the majority of the population from which it derived its signals from. The plots represent the top performing LABBS model for each simulated time-series. In the first series LABBS outperformed 432 of the 500 individuals in [P] and in the second series LABBS outperformed 483 of the 500 in [P].

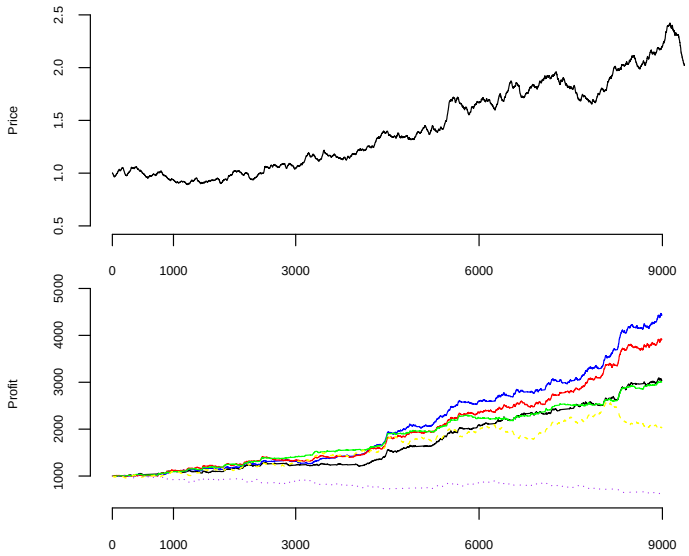

Fig. 2. An artificial price series (top) and cumulative profits (bottom) for case 1 (black), case 2 (blue), case 3 (red), case 4 (green), buy-and-hold (yellow-dashed) and the canonical form of the BB (purple-dotted). The system is able to capture short-term trend changes in the price series for long and short trading, which can be seen clearly when cumulative profits increase as the price-series is contracting.

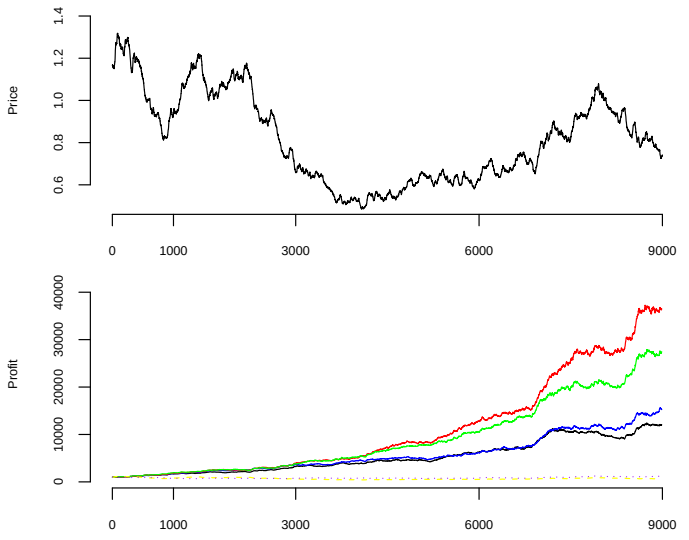

Fig. 3. An artificial price series (top) and cumulative profits (bottom) for case 1 (black), case 2 (blue), case 3 (red), case 4 (green), buy-and-hold (yellow-dashed) and the canonical form of the BB (purple-dotted). Each of the LABBS models well outperforms the benchmarks capitalizing on the series contractions and the major trend changes.

B. Real World Application

1) Data: To test the effectiveness of LABBS on real-world data, experiments were performed on two data sets comprised of daily closing prices. The first is BAE systems (ticker: BAE-L) from the London Stock Exchange for an 8-year time period from Jan 2003 - Nov2011. The second is Exxon Mobile (ticker: XOM) for a 10-year time period from 2001-2010 (Both data sets were collected from Yahoo! Finance). Plots of both series can be seen in the top panels of figures 5 and 7. These time-series were chosen as BAE represents a strong negative trend and XOM exhibits several trend changes and produces a profit over the test period.

2) Experiment Setup: The experiments were performed using the same parameter settings from the simulated data. With the exceptions that the match set window size is set to 200 days (as it was found that larger windows were more effective with the actual stock data) and the introduction

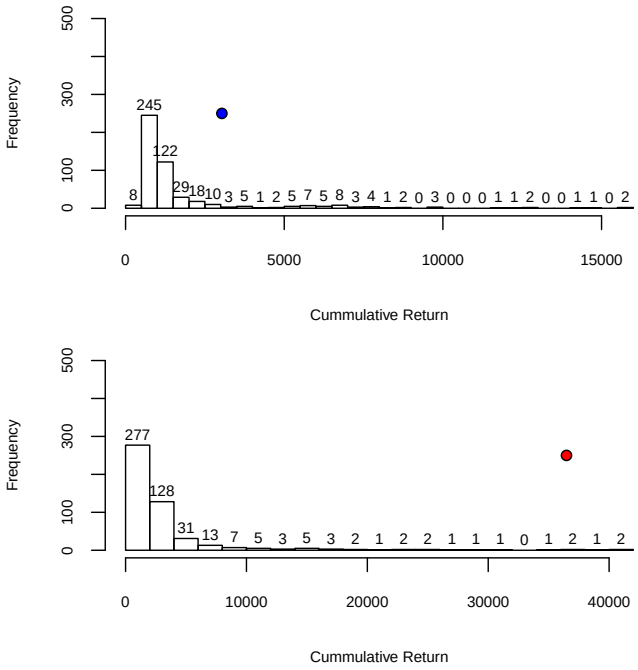

Fig. 4. Histogram plots of the cumulative profits from LABBS population for first simulated series (top) and second simulated series (bottom), for the top performing LABBS models. The blue and red dots represent the cumulative profits for the LABBS model. In both cases it is clear that LABBS was able to outperform the majority of the population, which shows that it was effective at combining the signals.

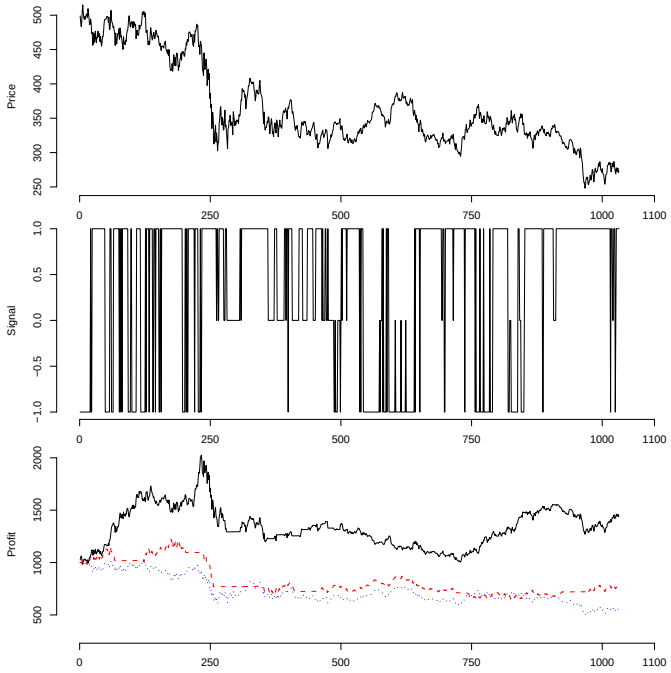

Fig. 5. BAE price series (top), trading signals (middle) and cumulative profits (bottom). Plotted are LABBS (black), buy-and-hold (blue-dotted), and canonical BB (red-dashed). The system performs well when the market is trending and increases profits by taking short positions during down trends. The system is negatively impacted by the large drop around the 250 pt. mark but then picks up and continues to be profitable afterwards. LABBS was active in the market 86% of the time.

of current trade criteria. The current trade criterion simply evaluates any current trades ABBs are in when assessing suitability for [M]. This criterion was introduced as a result of extending the match set window. A performance measure called *trade accuracy* was the number of positive trades divided by the total number of trades, where a trade is defined as it was in section II-C.

3) BAE Results: The results in figure 5 display the effectiveness of LABBS. Over the 4 year test period BAE experienced a significant downward trend. LABBS was able to model this property and produce a positive return. From the signal plot we can observe that LABBS executed short and long positions during this period. The trade accuracy for this data set was 65.03%. The histogram from figure 6 shows that LABBS was effective at combining the signals produced by [P], where LABBS outperformed 486 of the 500 agents.

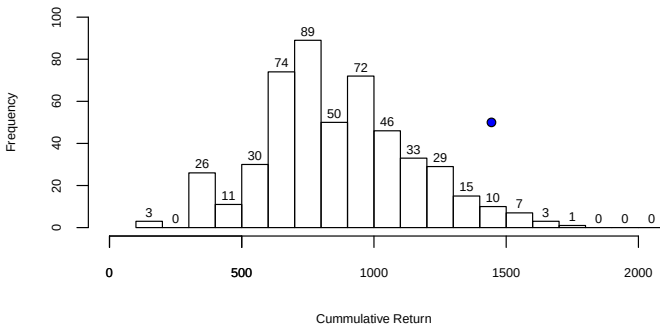

Fig. 6. A histogram of the BAE cumulative returns of the population [P], the blue dot is the return for LABBS. LABBS was able to outperform 486 of the 500 agents in [P].

4) XOM Results: From figure 7 we can observe that LABBS was able to outperform both benchmarks (the buy and hold and the canonical BB), however the separation between the models is less significant. For the first three years of the test period LABBS was underperforming the buy and hold approach and at times the canonical use of the BBs. However, after the mid-way point of the second year LABBS recovers and experiences a steady increase in profitabiltiy. After the market as a whole suffered a sharp decline, LABBS was able to insulate itself using short positions. The first two years of this example highlights a situation when LABBS is not as effective. During this period the population [P] was also underperforming and since LABBS is dependent on the signals generated from [P] it suffered from the lack of reliable information being generated. A good indicator of how well the system was working is the system performance in relation to the individual agents in [P]. From figure 8 we can observe that LABBS outperformed 460 of the 500 agents which implies that LABBS was still able to improve on the agents' individual signals using the action selection criteria described in section II-F. Additionally LABBS was also 67.45% accurate over the entire test period.

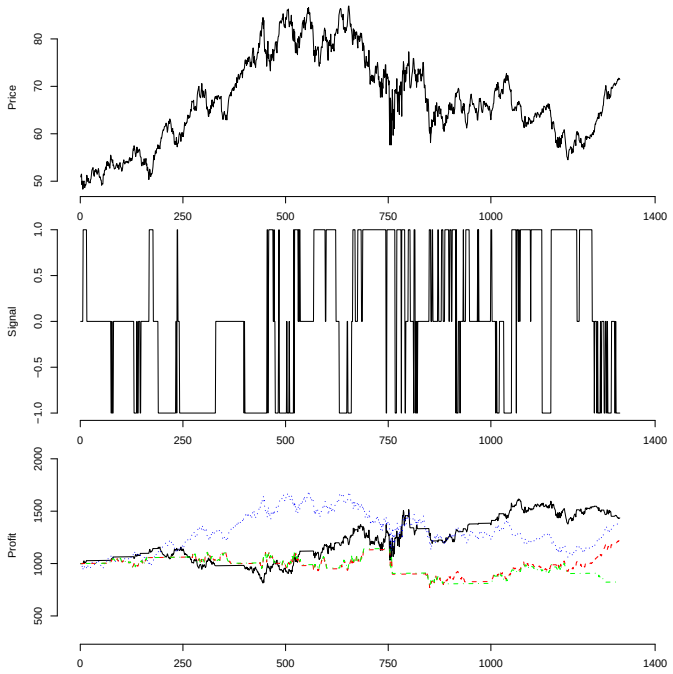

Fig. 7. XOM price series (top), trading signals (middle) and cumulative profits (bottom). Plotted are LABBS (black), buy-and-hold (blue-dotted), and canonical BB (red-dashed, green-dash-dot). The two BB models only differ by a 1% difference in the allowable stop-loss, these two plots highlight the sensitivity of the indicator. LABBS outperforms the benchmarks in the test-period but only after the large structural break in stock price. LABBS was able to capitalize on the strong downward trend commencing around 700 pt. marker, and was active in the market 62% of the time.

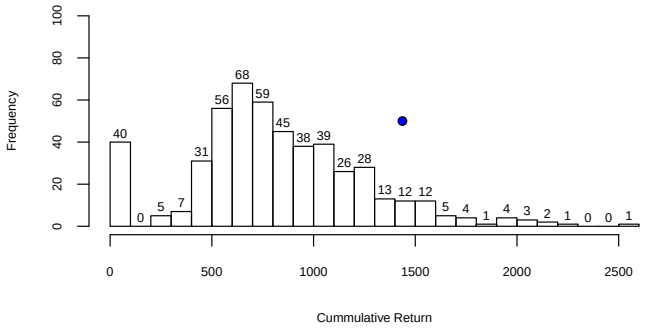

Fig. 8. A histogram of the XOM returns of [P], the blue dot is the return for LABBS. LABBS was able to outperform 92% of the agents in [P].

V. Conclusions

We have proposed and tested a methodology for optimizing a population of technical indicators and combining their heterogeneous set of signals into a coherent investment strategy. We have also extended a PSO algorithm to multi-objective optimization and allowing re-initializing to counteract local maxima. The overall framework of the proposed system is inspired from a Learning Classifier System but has been significantly changed to accommodate a population that does not contain classifiers. Though the study investigates the particular situation where BBs are utilized, the framework can be generalized to any technical indicator or function that can be fitted using population based optimization.

The results from the simulated data clearly demonstrate that LABBS is able to identify short-term trending properties in a financial time-series and exploit them for superior investment returns. This was observed during market expansion and contractions, though it is more obvious in market down trends. The real-world results highlight the strengths and weaknesses of the LABBS approach. The results from BAE display that LABBS can insulate an investor from downward trends and produce a profit in market declines. However, in XOM the results highlight that the performance of LABBS is bounded by the quality of the solutions in [P], and when XOM experienced a strong positive trend the LABBS system only produced modest gains.

Future work concerns widening the influence and control of the meta agent. Currently the only parameter concerned for reinforcement learning was the reputation of the ABBs in [P] but this list could be expanded to include other parameters the system is sensitive to, such as, the match window size, number of agents in [P], and performance criteria.

References

[1] Matthew Butler and Dimitar Kazakov. Particle swarm optimization of Bollinger bands. In *ANTS Conference*, pages 504–511, 2010.

[2] Matthew Butler and Dimitar Kazakov. Testing implications of the adaptive market hypothesis via computational intelligence. In *Proceedings of IEEE Computational Intelligence for Financial Engineering and Economics*, page TBD, 2012.

[3] Andries Engelbrecht. Heterogeneous particle swarm optimization. In *Swarm Intelligence*, volume 6234 of *Lecture Notes in Computer Science*, pages 191–202. Springer Berlin / Heidelberg, 2010.

[4] Xiaohui Hu and R. Eberhart. Multiobjective optimization using dynamic neighborhood particle swarm optimization. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, pages 1677 –1681, 2002.

[5] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942 –1948 vol.4, Nov/Dec 1995.

[6] Ju-Sang Lee, Sangook Lee, Seokcheol Chang, and Byung-Ha Ahn. A comparison of GA and PSO for excess return evaluation in stock markets. In *IWINAC (2)*, pages 221–230, 2005.

[7] C. Lento, N. Gradojevic, and C.S. Wright. Investment information content in Bollinger Bands? *Applied Financial Economics Letters*, 3(4):263–267, 2007.

[8] Robert A. Levy. Conceptual foundations of technical analysis. *Financial Analysts Journal*, 22(4):83–89, 1966.

[9] Jin Li and Edward P. K. Tsang. Improving technical analysis predictions: An application of genetic programming. In *Proceedings of the Twelfth International Florida Artificial Intelligence Research Society Conference*, pages 108–112. AAAI Press, 1999.

[10] Andrew W. Lo. The adaptive markets hypothesis. *The Journal of Portfolio Management*, 30(5):15–29, 2005.

[11] John Moody, Lizhong Wu, Yuansong Liao, and Matthew Saffell. Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting*, 17(5-6):441–470, 1998.

[12] Christopher Neely, Paul Weller, Robert Dittmar, Chris Neely, Paul Weller, and Rob Dittmar. Is technical analysis in the foreign exchange market profitable? A genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32:405–426, 1997.

[13] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pages 69–73, 1998.

[14] Stewart W. Wilson. Classifier fitness based on accuracy. *Evolutionary Computation*, 2(3):149–175, 1995.