Dynamic High Frequency Trading: A Neuro-Evolutionary Approach

Robert Bradley, Anthony Brabazon, and Michael O'Neill

Natural Computing Research and Applications Group
University College Dublin, Ireland
robert.bradley@ucdconnect.ie, anthony.brabazon@ucd.ie, m.oneill@ucd.ie

Abstract. Neuro-evolution of augmenting topologies (NEAT) is a recently developed neuro-evolutionary algorithm. This study uses NEAT to evolve dynamic trading agents for the German Bond Futures Market. High frequency data for three German Bond Futures is used to train and test the agents. Four fitness functions are tested and their out of sample performance is presented. The results suggest the methodology can outperform a random agent. However, while some structure was found in the data, the agents fail to yield positive returns when realistic transaction costs are included. A number of avenues of future work are indicated.

1 Introduction

Bond markets play a significant role in capital allocation in developed economies. As an illustration of the scale of these markets, the total value of outstanding marketable bond debt in the US was approximately \$29.2 trillion [1] as of the 30th of September 2007. In comparison, the total amount of marketable equity of all US companies on the same date was approximately \$22.3 trillion [1]. Multiple issuers use bond markets to raise funding including companies, financial institutions, government agencies and central government. Given the scale and liquidity of bond markets they attract substantial trading interest.

The construction of an intraday trading system for bond futures is a difficult task as the time series of prices from this market is quite volatile and we do not have strong theory to describe the exact nature of the price-generation process. This suggests that a model-induction methodology such as a multi-layer perceptron may have utility in uncovering this process from the underlying data. A practical difficulty in developing a multi-layer perceptron is that the creation of a quality network for a specific application can be time-consuming. Hence, there has been significant interest in the possibility of automating some or all of this development process by means of evolving neural net structures. A key issue in doing this efficiently is the matching of the design of the diversity-generating operator(s) to the choice of representation for the neural network. This is not a trivial task and a diverse range of approaches have been suggested. In this study we adopt the NEAT methodology [14,13] which adopts a principled approach to this issue.

Evolutionary algorithms have been used to evolve trading rules for a number of markets with varying degrees of success. Previous studies which apply evolutionary algorithms to the problem of trading try a number of fitness functions which vary from a simple total return value, to a risk adjusted return based ratio. This study employs a neuro-evolutionary methodology to model a dateset of high frequency bond futures data in an attempt to automatically trade the market. In addition, we investigate whether or not better out of sample performance can be gained by using different fitness functions. Four such functions were tested and their performance is compared against a zero intelligence random agent.

1.1 Structure of Paper

The rest of this paper is organised as follows. The next section provides an overview of the bond futures market along with a short literature review of EC applied to trading. We then outline our experimental approach. This is followed by a results section. The paper finishes with a conclusion and future work section.

2 Background

2.1 Trading Bond Futures

A futures contract is a standardised agreement between two counter-parties where the buyer (seller) is agreeing to take delivery of (deliver) a specific quantity of the *underlying* (for example, a financial security, a foreign currency or a commodity etc.) on a specific date in the future (called the *maturity date*), for a price agreed upon now. Futures can, for example, be used by producers and consumers in order to hedge their respective future income and exposures by providing assurance as to the future price they will receive/pay for some item. This is essential for businesses when trying to manage projected cashflows and associated risks. Traders in financial markets can also use futures to hedge certain exposures in their portfolio. For example, a Fixed Income dealer might want to protect a portfolio of government bonds from adverse changes in interest rates by purchasing, or selling, bond futures. Futures can also be used by speculators who want to express a view on the direction of the market.

In the case of fixed income futures the underlying is a fixed income product such as a government bond. The future's market price is quoted in the same way as the underlying bond, i.e. as a percentage of the face value of the bond.

In this study we look at three particular fixed income futures which derive their value from German government bonds and which are traded on Eurex; the *Euro-Schatz* (FGBS), *Euro-Bobl* (FGBM), and *Euro-Bund* (FGBL) Bond Futures. All three futures trade the next 4 maturities in a quarterly (March, June, September, December) delivery cycle and expire on the 10^{th} day of the delivery month.

2.2 Literature Review

There have been numerous previous studies which employ EC methodologies such as GA and GP to evolve trading rules. Most of these works have either

Table 1. An overview of the fitness functions employed by previous work

Reference	Market	Frequency	Fitness Function
Dittmar & Neely et al 97 [8]	FX	Daily	Log Returns
Allen & Karjalainen 99 [9]	Stock Index	Daily	Log Returns
Neely 99 [11]	Stock Index	Daily	Log Returns & Sharpe Ratio & X*
Dempster & Jones 00 [6]	FX	Intraday	Sterling Ratio
Dempster et al 01 [7]	FX	Intraday	Log Returns
Bhattacharyya & Pictet et al 02 [3]	FX	Intraday	Sharpe Ratio & UtilityFunc
Brabazon & O'Neill 04 [4]	FX	Daily	Return - MaxDrawdown
Neely & Weller 03 [10]	FX	Intraday	Log Returns
Dempsey & O'Neill et al 04 [5]	Stock Index	Daily	Sharpe Ratio
Azzini & Tettamanzi 08 [2]	Stocks	Daily	Sortino Ratio

evolved trading rules for spot foreign exchange or stock markets. There has been little or no investigation into applying these algorithms to the problem of trading derivatives such as futures.

A key decision when setting up an EC experiment is choosing a suitable fitness function. This function acts as a selection criterion which biases the stochastic search process. A number of different fitness functions have been utilised in previous work. These can be broadly categorised as being risk, or non risk adjusted measures of fit.

Table 1 lists some of the more recent applications of EC to trading. Most of the literature referenced in table 1 does not compare the out of sample performance under different fitness functions, with the exception of [3] where it is found that risk adjusted metrics yield more stable out of sample performance. This study aims to investigate whether or not different fitness functions do in fact yield different out of sample behavior. In addition, we are applying our chosen methodology to a dataset of high frequency Bond Futures data to see if profit can be captured from the futures market.

3 Experimental Approach

3.1 Methodology

In this study we employ the neuro-evolution of augmenting topologies (NEAT) methodology to evolve multi-layer perceptrons (MLP) for the purposes of developing a trading system for the German bond market. NEAT was developed in 2002 by Stanley and Miikkulainen [14,13] and attempts to overcome the problems of evolving MLPs using a direct encoding of an MLP structure. NEAT simultaneously evolves both MLP topology and weights. Proponents of NEAT claim that it:

1. applies a principled method of crossover (i.e. attempts to overcome the permutation problem),
2. protects structural innovations (i.e. attempts to overcome noisy fitness problem), and
3. grows MLPs from a minimal structure (i.e. attempts to ensure that final MLP is no more structurally complex than necessary).

In order to achieve this NEAT uses three mechanisms, a novel MLP encoding which allows for 'sensible crossover', a speciation concept which offers some protection for structural innovations, and a seeding process whereby all initial MLP structures are of miminal size.

3.2 Data Review

The dataset used in this study consists of 5,000 5 minute bars of intraday data for the three German bond futures mentioned in Sect. 2; the Euro-Schatz, Euro-Bobl, and Euro-Bund. A bar contains a value for the open, high, low, and close prices for the interval, and also the number of contracts traded in the 5 minute period (volume). The series exhibit a variety of price behaviors, including bullish, bearish, and choppy periods. This varied behavior poses a difficult learning environment for NEAT.

The bond futures market is one of the most active fixed income markets. Table 2 shows a number of basic statistics on the volume of activity (volume of contracts traded) on the market for the period 13/06/2008 to 25/07/2008. The Bund tends to be more volatile, and more heavily traded than the Schatz and Bobl. Total volume for the given period of trading in the Schatz is over 15 million contracts, compared to the Bund where over 25 million contracts were traded. The average number of contracts traded per 5 minute block ranges from 2,887 (Bobl) to 4,812 (Bund). Each of these corresponds to a bond with a face value of Euro 100,000 and hence represents a substantial 'gross' position in the underlying (bond) instrument.

Table 2. Volume statistics (measured in number of contracts traded per 5 min block)

	Schatz	**Bobl**	**Bund**
Total	15,186,286	14,432,952	24,057,734
Mean	3,037	2,887	4,812
Std	3,817	3,146	5,367
Max	59,738	40,240	57,818
Min	1	1	1

A move in a bond future price from 99.31 to 99.32 corresponds to a full 'tick'. As already noted, the Bund moves in full ticks, but the Schatz and Bobl contracts move in half ticks. Table 3 describes the behavior of the first differences of the closing price data. The volatility varies across contracts, with the Bund being the most volatile with an average move of 0.274 of a tick between each 5 minute block. As can be seen, the standard deviation of the number of tick changes between five minute blocks is quite high relative to the mean, illustrating the volatile nature of price changes in even short time periods for these futures.

Table 3. Ticks in five minute block

	Schatz	**Bobl**	**Bund**
Mean	0.0099	0.0208	0.0274
Std. Dev	1.1825	2.6142	3.5879
Max Up	11.5000	24.5000	22.0000
Max Down	-8.5000	-24.0000	-34.0000

3.3 Experimental Parameters

In setting the parameters for the NEAT system used to generate the MLPs, we considered parameter settings reported in prior applications of NEAT and supplemented this with some trial and error experimentation. We have not attempted to optimise the parameter settings but experimentation indicated that the results obtained were not hypersensitive to small changes in these settings. Table 4 lists the most important parameter settings we employed. The elitism proportion parameter was set to 20% which means that the top 20% of the population are passed on to the next generation without being altered by genetic operators. This ensures that the population does not lose high performing individuals to destructive mutation and crossover. The “min/max species threshold” settings allow the user to keep the number of species within a certain range. The upper limit was set to 10 and the lower limit was set to 6. As the population evolves, the number of neurons and connections in the average network increases according to the mutation probabilities in Table 4. Over time the level of complexity can increase to a level which brings with it major computational overhead. To combat this problem the average complexity is tracked, and once it breaks the complexity threshold the search switches from complexification to pruning mode. Pruning reduces the population’s average complexity until it falls below the specified threshold (100 in our case). This is achieved by randomly removing nodes and edges from more complex individuals.

Table 4. Experimental Parameters

Parameter	Value
Pop size	500
Mutate weights	.005
Add neuron	.01
Add connection	.1
Crossover	.9
Elitism Proportion	.2
Complexity threshold	150

3.4 Trading Methodology

In this paper we concentrate on inputs which are developed from the price time series of the futures. A total of 75 inputs are available for use by the MLPs being evolved. The inputs were calculated by firstly taking the first differences (between two succeeding five minute blocks) of the open, high, low, close, and volume time series for each of the three futures, and then dividing each observation in each time series by their respective minimum price change, which is .01 in the Bund, and .005 in the Schatz and Bobl. This results in a series of tick changes for each future. Five lags of each series is then input to the network.

The networks are all initialised with 5% of the inputs chosen at random and a single bias node, connected to a single output node. Thus, the population of MLPs are initialised with minimal complexity. This leaves evolution to decide which of the other inputs should be switched on and the number of hidden nodes that should be added. The evolved MLPs output an integer between 0 and 1, which is post-processed using:

$$y = \begin{cases} 1 & \text{if } a \geq 0.6 \\ 0 & \text{if } 0.3 < a < 0.6 \\ -1 & \text{if } a \leq 0.3 \end{cases}$$

where y is the final network output and a is the MLP output before postprocessing. The resulting value of y is then input to the trading simulator. The trading simulator is used to evaluate the performance of a network over a given dataset. The simulator accepts three signals; buy (1), sell (-1), and do nothing (0). At any point in time the system is either long or short 1 future contract, or flat. If the trade position is long and we get a sell signal, the long position is closed out at the current market price. Conversely, if we are short and get a buy signal we close out the short position and go flat. This behavior results in a simple trading strategy where the network is allowed to stay out of the market. The maximum position is limited to a single future to make post trade analysis of results easier.

The trading simulator records the state of the system at the end of each five minute interval. A number of state variables are recorded including the realised profit and loss, the position (long/short), and any trade executed. Upon reaching the end of the dataset the simulator prints the state vector to a csv file for further analysis.

Fitness functions. The moving window approach yields an out-of-sample realised equity curve for agent A

$$R(A) = (r(A,t) : 0 <= t < N) \tag{1}$$

where t are evenly spaced 5 minute intervals, N is the size of the equity curve, and $r(A,t)$ is the realised profit and loss (PnL) including cost deductions at interval t.

Total return. The first fitness function tested was the finishing PnL of the system $r(A, N-1)$. This is a naive measure of fit as it ignores the equity curve up to and including N-2.

$$FF_1 = r(A, N-1) \tag{2}$$

Total return minus max drawdown. A better approach is to subtract the maximum drawdown in R(A) from the finishing PnL $r(A, N-1)$. Subtracting the max drawdown will put a selection pressure on the search which favours networks that avoid devastating losses.

$$FF_2 = r(A, N-1) - MDD(A, R(A)) \tag{3}$$

Although this measure is an improvement over total return, it ignores information from the equity curve.

Information ratio. The third fitness function we tested is a variation the mean change in return in a 5 minute period divided by the volatility of these changes. First, we calculate the first differences of the equity curve

$$RD(A) = (rd(A, t) : 1 <= t < N) \tag{4}$$

where $rd(A, t) = r(A, t) - r(A, t-1)$. We then divide the mean of the first differences by the standard deviation of the same series.

$$FF_3 = \frac{R\bar{D}(A)}{\sigma_{RD(A)}} \tag{5}$$

This ratio translates to the average change in the realised PnL in a 5 minute interval divided by the average deviation from this level. This fitness function favours networks with a good risk to reward relationship, thus deselecting agents with volatile performance. Unlike FF_1 and FF_2, this metric considers the entire equity curve in its calculation. The denominator in equation 5 is the standard deviation statistic, which gives an equal weighting to positive and negative deviations from the mean.

Sortino ratio. Another risk adjusted ratio is the Sortino Ratio [12] which does not include positive deviations in its measurement of risk. This makes sense as positive volatility results in profit and should not be penalised. For the purpose of this study we define a slight variation of the original ratio

$$FF_4 = \frac{R\bar{D}(A)}{DSR(A, RD(A))} \tag{6}$$

where

$$DSR(A, RD(A)) = \sqrt{\frac{1}{N}\sum_{i=1}^{n}[rd(A, i) < 0][rd(A, i)]^2} \tag{7}$$

4 Results

A total of 30 runs were carried out for each of the four fitness functions discussed in Section 3.4 using the experimental parameters listed in Section 3.3. Each

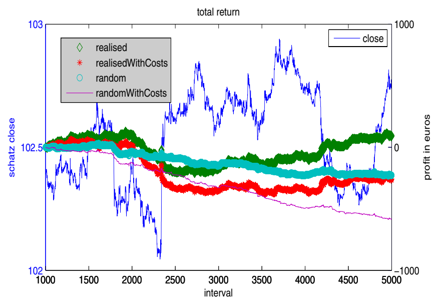

Fig. 1. Out of sample performance for fitness function 1: Total Return

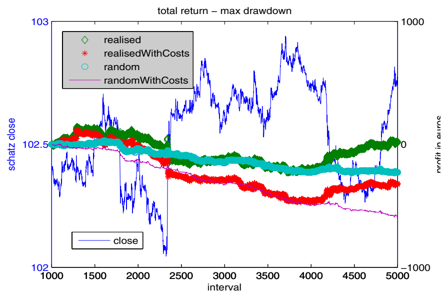

Fig. 2. Out of sample performance for fitness function 2: Total Return - Max Drawdown

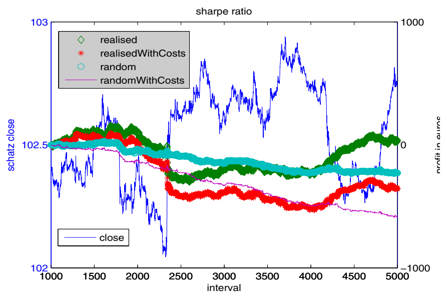

Fig. 3. Out of sample performance for fitness function 3: Info Ratio

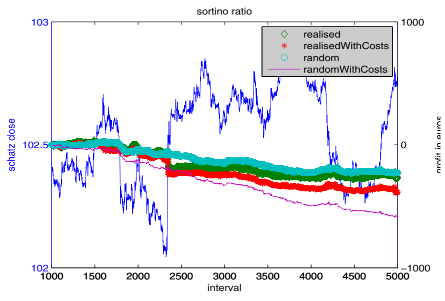

Fig. 4. Out of sample performance for fitness function 4: Sortino Ratio

run results in an out of sample equity curve for the best network, which may change between moving window increments. Figs. 1 to 4 show the equity curves for the aforementioned fitness functions averaged over the 30 runs, which are plotted against a random agent. The random agent buys, sells, or does nothing at each interval with equal probability. An average of 30 simulations is used as a benchmark.

The networks outperform the random agent both with and without transaction costs. However, the networks fail to finish in positive territory once transaction costs are included. Realistic transaction costs were used as specified by Eurex: a single contract round trip costs 40 cents (20 cents each way). As is evident from Figs. 1 to 4 the four fitness functions did not yield significantly different results. However, the trading strategy was limited to a single contract position, and it would be interesting to let the system buy/sell a variable number of contracts depending on the strength of the signal and see if the different fitness functions result in different behavior. It is possible that agents evolved with risk adjusted metrics of return as their fitness function would yield more conservative trading behavior relative to those agents evolved using fitness functions based on non risk adjusted measure of return and this will be investigated in future work.

5 Conclusion and Future Work

This study presents a novel application of a neuro-evolutionary methodology for the purposes of the intraday trading of German government bond futures. To date, few studies have examined the potential utility of computational intelligence methodologies for the purpose of trading on this market and none have adopted a neuro-evolutionary approach. In spite of restricting attention to a very limited set of potential inputs, the results suggest that the resulting model is capable of uncovering structure in the time series of bond futures prices and is capable of using this information to trade with some success. The out of sample results for the four fitness functions were not significantly different. However, due to the fact that the trading strategy was limited to a 1 lot position there was not much room for evolution to induce different levels of trading aggressiveness. An interesting piece of future work might be to allow the agents to buy/sell a variable number of contracts depending on the strength of the signal.

Of course, no conclusive assessment of the utility of this methodology can be made on the basis of the initial experiments we have undertaken in this study and we intend to pursue multiple avenues to further extend this work. For example, the current trading simulator only uses a limited range of inputs and its power could be further enhanced by use of established filter rules for price data such as technical indicators. We also note that we adopt a simple trading strategy in this study, whereby the simulator can only go long or short one contract in response to a buy/sell signal. We intend to refine this in order to allow the system to build up a larger long / short position in response to successive buy/sell signals and also to allow the system to buy/sell varying numbers of contracts depending on the strength of the signal generated by the system. We also intend to investigate the application of grammar-based GP for trading this market.

References

1. Securities industry and financial markets association - research quarterly. Technical report (November 2007)
2. Azzini, A., Tettamanzi, A.G.B.: Evolving neural networks for static single-position automated trading. J. Artif. Evol. App. 8(2), 1–17 (2008)
3. Bhattacharyya, S., Pictet, O.V., Zumbach, G.: Knowledge-intensive genetic discovery in foreign exchange markets. IEEE Transactions on Evolutionary Computation 6(2) (April 2002)
4. Brabazon, A., O'Neill, M.: Evolving technical trading rules for spot foreign-exchange markets using grammatical evolution. Computational Management Science 1(3), 311–327 (2004)
5. Dempsey, I., O'Neill, M., Brabazon, A.: Live trading with grammatical evolution. In: GECCO 2004 Workshop Proceedings, June 26-30 (2004)
6. Dempster, M.A.H., Jones, C.M.: A real-time adaptive trading system using genetic programming. Quantitative Finance 1, 397–413 (2000)
7. Dempster, M.A.H., Payne, T.W., Romahi, Y., Thompson, G.W.P.: Computational learning techniques for intraday fx trading using popular technical indicators. IEEE Transactions on Neural Networks 12(4), 744–754 (2001)

8. Dittmar, R., Neely, C.J., Weller, P.: Is technical analysis in the foreign exchange market profitable? a genetic programming approach. CEPR Discussion Papers 1480, C.E.P.R. Discussion Papers (September 1996)
9. Allen, F., Karjalainen, R.: Using genetic algorithms to find technical trading rules. Journal of Financial Economics 51, 245–271(27) (1999)
10. Neely, C.J., Weller, P.A.: Intraday technical trading in the foreign exchange market. Journal of International Money and Finance 22(2), 223–237 (2003)
11. Neely, C.J.: Using genetic algorithms to find technical trading rules: A comment on risk adjustment. SSRN eLibrary (1999)
12. Sortino, F.A., van der Meer, R.: Downside risk. Journal of Portfolio Management 17, 27–31 (1991)
13. Stanley, K., Miikkulainen, R.: Efficient evolution of neural network topologies. In: Proceedings of the 2002 Congress on Evolutionary Computation (CEC 2002), pp. 1757–1762. IEEE Press, Los Alamitos (2002)
14. Stanley, K.O., Miikkulainen, R.: Evolving neural networks through augmenting topologies. Evolutionary Computation 10, 99–127 (2002)