Exploring the Value of Prediction in an Artificial Stock Market

Bruce Edmonds

Centre for Policy Modelling,
Manchester Metropolitan University
`http://cfpm.org/~bruce`

An action selection architecture is described which uses two learning modules: one to predict future events (the PPM) and one to decide upon appropriate actions (the IALM). These are only connected by the fact that they have access to each other's past results and the IALM can use the predictions of the PPM as inputs to its action selection. This is instantiated in a model which uses GP-based learning mechanisms for the two modules and is tested in an artificial stock market. The point of the exercise is to start exploring the conditions under which prediction might be helpful to successful action selection. Preliminary results indicate that prediction is not always an advantage. This set-up is briefly compared to that of the anticipatory classifier system. I speculate that prediction might have similar role to learning in the "Baldwin Effect" and that the "momentum" of the system might be a significant factor.

Introduction

There are two basic feedback mechanisms that are used in learning processes: *firstly*, from indicators such as pleasure, pain, or profit and, *secondly*, the (mis)match between what was anticipated to occur and what actually did occur, i.e. the error. These roughly correspond to the goals of *gaining utility* and *correctly predicting*. There are many ways of using these feedback mechanisms in different learning processes.

Planning systems attempt to *infer* what actions to take to achieve the former from models that accurately reflect their problem domain. Their use is dependent upon a raft of assumptions holding, including that:

1. A sufficiently correct model is known;

2. The domain is sufficiently static over the planning period;

3. And time, resources and the model structure make the inference feasible.

These conditions are onerous for many situations. Condition (1) means that either the model is knowable *a priori* or that some learning process using error-based feedback has had time to produce a sufficiently correct model. Condition (2) means that the domain must be pretty static and immune to unpredictable perturbation as a result of any actions that are taken. Condition (3) means that the model has to be pretty complete with respect to the domain, that the planning can be done without tight time constraint and that there are considerable computational resources available for the task. Taken together these rule out almost all situations of interest to animals or animats. In particular they rule out any dynamic or tightly-interactive situations; they rule out cases where any learnt model only captures restricted aspects of the domain; they rule out any entities without the luxury of expensive off-line computation.

On the other hand learning systems that *only* use indicator-based feedback can be at a disadvantage. They are unable to prevent or prepare for any event before it happens, which can be critical in life-threatening situations (e.g. escaping from a predictor, or being the first to acquire items of food). They may be 'locked-in' to local utility optima because they have no access to what *might* happen. Thus for some creatures it would seem that having some anticipatory ability would be highly advantageous (especially if it did not involve a high cost or force them to take 'time-out' to think).

However the exact conditions wherein an ability to anticipate future events might be advantageous are unclear. Many of the advantages of anticipation could be gained from learning over past sequences of events as to how to improve the appropriate indicator (e.g. avoidance of pain). Take an example: say event C tends to follow the event sequence A, B, and that it is in the entity's interest to take action X as (or just before) event C. Such an entity could approach this in two ways: *firstly*, it could learn to anticipate event C from the sequence A, B and thus learn to take action X, but *alternatively*, it could simply learn that taking action X after the sequence A, B was in its interest. In other words the intermediate anticipation C could be eliminated from the process.

In this paper I use a simulation to compare the performance of entities with and without the ability to predict future events. This is thus a preliminary exploration of the conditions under which such anticipation is advantageous. I have chosen the environment of an artificial stock market as the test bed for the following reasons:

- it is inherently unpredictable and dynamic;
- actions taken (buying and selling) affect the environment, sometimes in unpredictable ways;
- models of the environment (namely prices) have to be continually changed as part of a modelling "arms-race" with the other traders;
- accurate prediction (of future price of stocks) is cleanly distinguishable from utility (profit made by trading);
- is it possible, but unclear, that anticipation will be advantageous.

The agents employ learning algorithms based on Genetic Programming (GP) [7] – several learning process could have been used as long as they are sufficiently flexible and expressive. They do no planning and almost no inference – that is they do *no* formal inference and the action decision involves only fixed and limited steps that could be interpreted as inference (e.g. only attempting to buy what they have cash for).

The next section describes the model set-up, followed by a section on the results. I finish with a brief discussion.

The Model Set-up

The environment

The environment is a stock-market. That is to say there are a fixed number of stocks which traders can buy or sell at the current price. There is only a limited quantity of each stock available in the market. There is one market-maker who sets the prices depending upon past demand. There are two such rules that I used: *firstly* what I call the "inflationary pricing mechanism" where if there has been a lot of recent buying the price goes up, and if there has been selling it goes down; and *secondly*, the "reverse pricing mechanism", where recent net buying means a drop in price and net selling a price increase. These pricing mechanisms are *not* realistic but serve to allow the market dynamics (for an account of how this actually happens in markets see [8]). The extent of the trading affects the extent to which the price changes. There is also some random noise added to the price changes. The market-maker is not anticipatory but has a fixed behaviour. All buying and selling transactions are subject to a transaction fee.

Each trading cycle each trader can seek to buy or sell each stock, as far as this is possible for it. Traders can only buy if: the market-maker has stock to sell; and the trader has sufficient cash. Traders can only sell if they have the stock. Also each trading cycle traders receive a dividend dependent upon the current dividend rate for the stock and their current holding of that stock.

The only 'fundamental' for this market is the dividend rate for each stock which wanders in a slow random walk. Almost all of the market dynamics result endogenously from the trading that occurs. These dynamics never 'quieten down' to anything like an equilibrium, because each agent is continually learning about the market, so if it settles down to a pattern this is quickly learnt; exploited and hence disrupted. The interplay of agents can be seen as a series of learning "arms-races" where by competing behaviours are being co-developed[1]. This can be characterised (albeit rather inadequately) as a version of the "minority game" [1, 3].

At the start each agent is endowed with 100 units of cash and a small amount of each stock. The initial price and dividend is random for each stock. The market maker has 100 units of each stock. For the first 5 trading cycles, the traders make small random trades.

The agents

The agents perceive the current state of their environment and take a trading action each trading cycle. They do not have to trade. The actions they can take are:

- Buy or sell an amount of each stock;
- Do nothing.

[1] Since the learning is implemented by a GP algorithm the behaviours are literally co-*evolved*.

Their ‘inputs’ are:

- the current price of each stock;
- the current holding of each stock;
- the current holding of cash;
- the current stock index (an average of all prices);
- the last actions they took;
- the actions last trading cycle of other traders;
- any of the above a trading cycle ago (which can be iterated a number of times).

Each agent has two learning modules: the instrumental action learning module (IALM) and the price prediction module (PPM). Both are GP algorithms with very small populations (i.e. 10). The IALM attempts to learn the action strategy that will produce the most profit and the PPM attempts to accurately predict the prices of stocks. Each is given a rich vocabulary of nodes and terminals to build their tree-structures out of. The initial populations are generated at random out of these to a given depth.

Each module does the following steps once each trading cycle:

1. they build a new generation of the population (using propagation, tree-crossover and by generating new random trees);
2. they evaluate their population over the past 5 trading cycles – the IALM according to the profit it would have resulted in if the strategy was used (presuming prices to be unchanged by any actions the agent took), and the PPM as to how well it would have predicted the prices;
3. they choose the best price prediction in PPM and interpret it to predict the prices;
4. they choose the best action strategy and interpret it to decide their attempted action.

It is important to note that the action strategies in IALM can include terminals that refer to the current predictions made by the PPM’s best strategy. Thus the actions determined in stage (4) above can depend on the predictions in stage (3). The use of the predictions is *not* pre-programmed (e.g. by buying if it predicts the price will rise etc.) but is only provided as a possible input for the strategies learnt by the basic action learning module, the IALM. Thus it is quite possible that the IALM could learn to use the results of the PPM in strange ways, or even ignore it altogether. In fact, in the long run, one would *only* expect the IALM to preferentially evolve strategies that referred to the predictions *if* these provide it with some advantage in terms of profit.

Both the action strategies and the price prediction models being evolved can refer to *past* actions and predictions actually made as the result of the past selected best strategies and models. The current selected action strategy can utilise the present predictions about price. In order to test whether anticipation is advantageous to the agents we will set two versions of these entities in competition with each other. That is, the market will be composed of the market-maker plus two equally sized sets of traders – each set of traders made of the corresponding version of the entity. The only difference between the versions is that the first, *predictive*, version will predict the next price of each stock whilst the second, *non-predictive*, version will ‘predict’ only the present price of each stock. The alternative is to not allow the action strategies to utilise the predictions as inputs in the non-anticipatory version, but this would not be a fair comparison because the entity could use the price predictors as feature extractors and so act to simply increase the computational capacity of the entity and so denying the action strategies access to this might impoverish them. The predictive process the PPM feedback mechanism is adapted so that in the anticipatory version the predictors are evaluated against past predictions of the *next* price (at those times) and in the non-predictive version the predictors are evaluated against past predictions of the *current* price (at those times). The two versions of the entity are illustrated in figure 1.

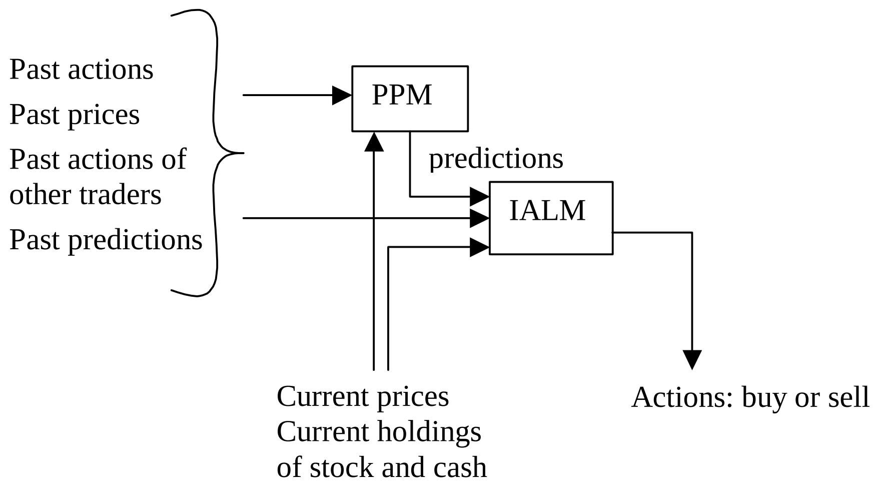

Fig. 1. An illustration of the learning structures of the traders

Thus the approach to anticipation used here is that the instrumental and predictive learning is done separately, but are loosely coupled by their ability to utilise (and hence adapt to exploit) the (best) results of each other. Each population co-evolves with the *results* of the other. This is in contrast to other approaches where the anticipation is directly associated with the action – in these approaches there is an anticipation of the effects of the possible action rather than, as here, an anticipation of environment. It is possible that in this model the PPM does evolve trees so that its predictions are dependent upon the previous actual actions if this was advantageous, but this is not necessarily so. Thus this algorithm is fundamentally different from the condition-action-anticipation of Tolman [11], Drescher [4] or Stolzmann [10] to a more modular structure where the anticipation and decision are done separately. In the final section I will briefly compare and discuss this approaches.

Obviously a major reason why I opted for the approach here is that it makes it easy to compare and test different adaptive mechanisms and analyse the results. When the action is bound tightly with the anticipation the feedback from the utility and the accuracy interact in ways that are sometimes difficult to detangle. I am not claiming here the superiority of my approach in any way, merely describing and investigating a series of approaches based on a loosely connected modular structure of learning processes.

Results

The results I describe here are preliminary. The time I had available and the slow speed of the simulations mean that I have not performed as many runs of the simulation exploring many different parameterisations as I would have liked.

Sets of runs were done for 8 simulation set-ups, each with an equal number of each kind of agent so that the relative success of each kind could be compared. The simulations came in two sizes: 3 agents of each kind and 7 agents of each kind. There were two numbers of stocks (4 stocks and 2 stocks); two pricing mechanisms (inflationary and reverse).

Inflationary Pricing Mechanism

Rising prices mean that agents learn to buy into stocks rather than keep their assets in cash. On the other hand, the fact that agents are always wishing to buy means that prices rise (under the inflationary pricing mechanism). Thus a pattern of rising prices is self-confirming. A typical pattern of exponential price rise from a typical run of the simulation is shown in figure 2 (note logarithmic scale).

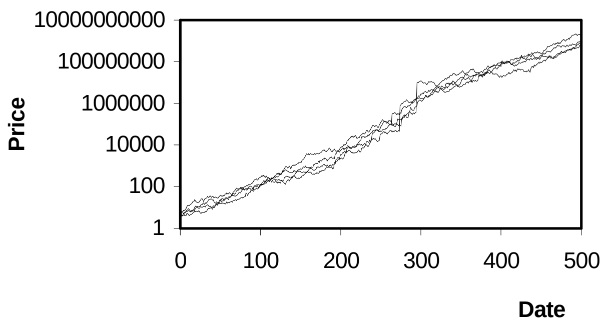

Fig. 2. Prices of stocks with inflationary pricing mechanism in simulation 1.

However, within this pattern of rising prices there is a lot of swapping between stocks, resulting in different stocks being the best buy at different times. The assets of the traders depend upon which stocks they choose to buy into at which stages.

Reverse Pricing Mechanism

In this version of the simulation the reverse pricing mechanism is used. This results in very different market dynamics – prices decline in the long run but with peaks caused by speculative bubbles in between. The prices of a typical run are shown in figure 3. Despite the appearance of less uniformity than in the simulations above with the inflationary pricing mechanism, the relative differences between stocks are more predictable being more stable. Also the price movements of the stocks are more obviously correlated.

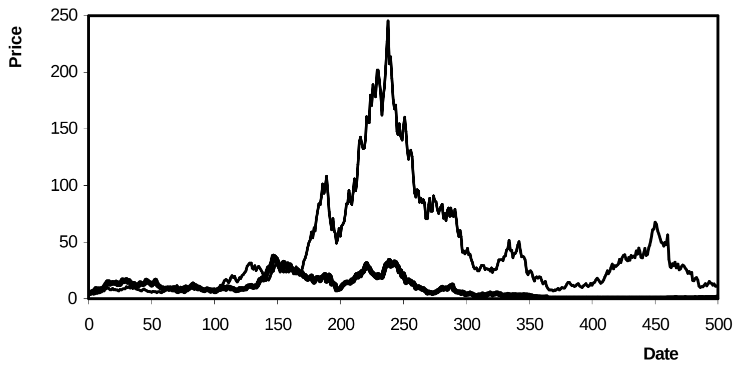

Fig. 3. Prices of stocks with reverse pricing mechanism (*non*-logarithmic price scale)

Table 1. Basic simulation set-ups

Number of agents of each kind	Number of stocks that can be traded	Pricing mechanism	Simulation label
3	2	Inflationary	3x2Agents 2Stocks, Inflationary
3	2	Reverse	3x2Agents 2Stocks, Reverse
3	4	Inflationary	3x2Agents 4Stocks, Inflationary
3	4	Reverse	3x2Agents 4Stocks, Reverse
7	2	Inflationary	7x2Agents 2Stocks, Inflationary
7	2	Reverse	7x2Agents 2Stocks, Reverse
7	4	Inflationary	7x2Agents 4Stocks, Inflationary
7	4	Reverse	7x2Agents 4Stocks, Reverse

The raw measure of the success of an agent is the total value of its assets, but since the market is roughly a zero-sum situation and the amount of money that it is possible to make depends upon the extent of the price turbulence. Thus it is more meaningful to compare the average success of each kind of agent against the average success of the other kind scaled by the standard deviation of the prices in that run. Figures 4 to 11 show the scaled difference of average assets for each of the 10 runs and the average of these. Lines above zero indicate that (on average) the predictive traders are doing better and below it the non-predictive agents.

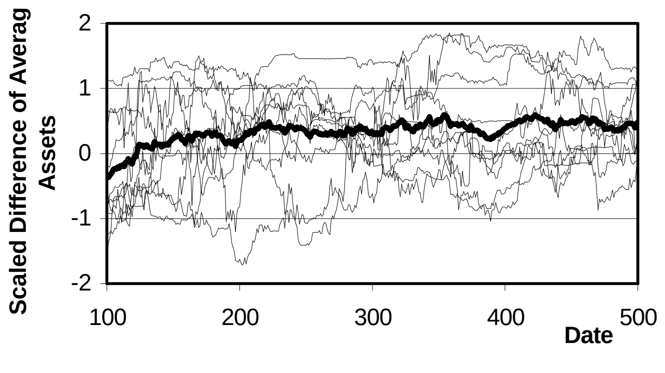

Fig. 4. Difference of average assets of predictive traders and average assets of non-predictive traders scaled by the standard deviation of each run in the simulation with label **3x2Agents 2Stocks, Inflationary** (the thin lines is the scaled difference for each of the runs, the average of these is the thick line).

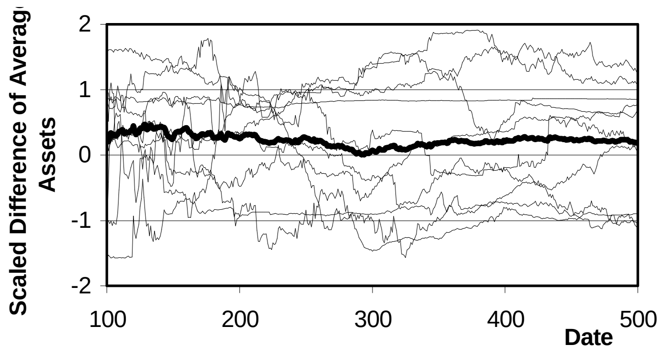

Fig. 5. Difference of average assets, **3x2Agents 2Stocks, Reverse**.

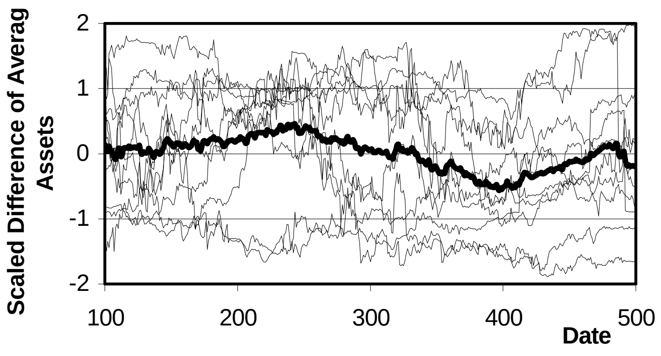

Fig. 6. Difference of average assets, **3x4Agents 4Stocks, Inflationary** .

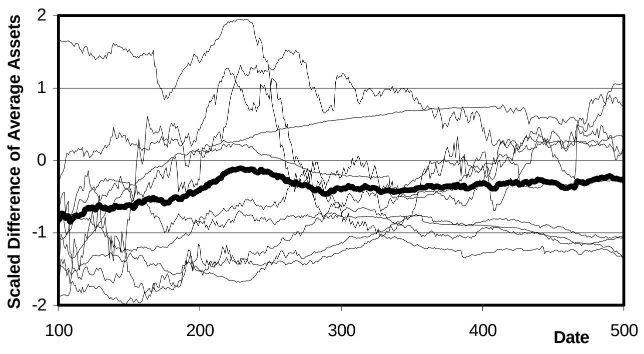

Fig. 7. Difference of average assets, **3x4Agents 4Stocks, Reverse**.

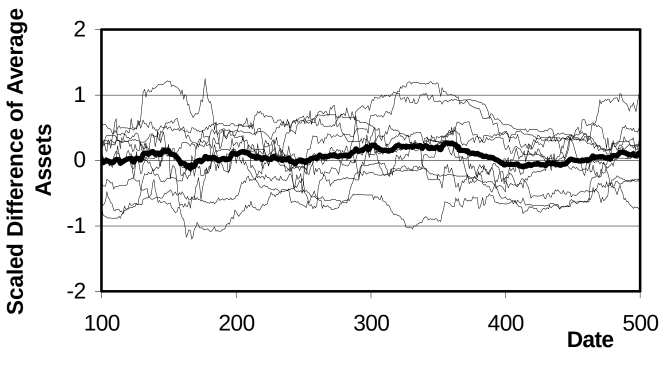

Fig. 8. Difference of average assets, **7x2Agents 2Stocks, Inflationary** .

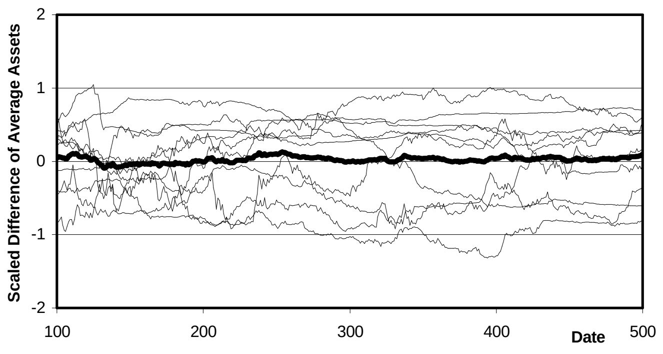

Fig. 9. Difference of average assets, **7x2Agents 2Stocks, Reverse**.

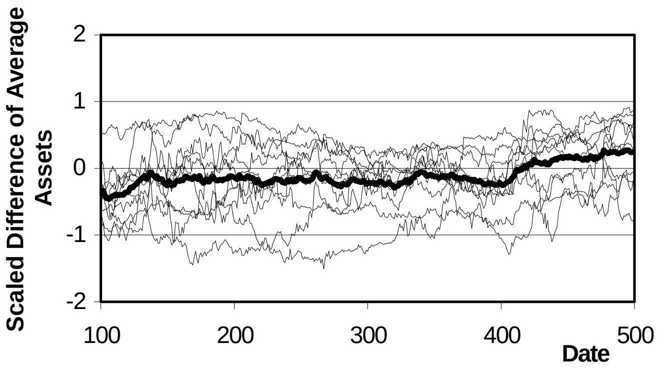

Fig. 10. Difference of average assets, **7x2Agents 4Stocks, Inflationary** .

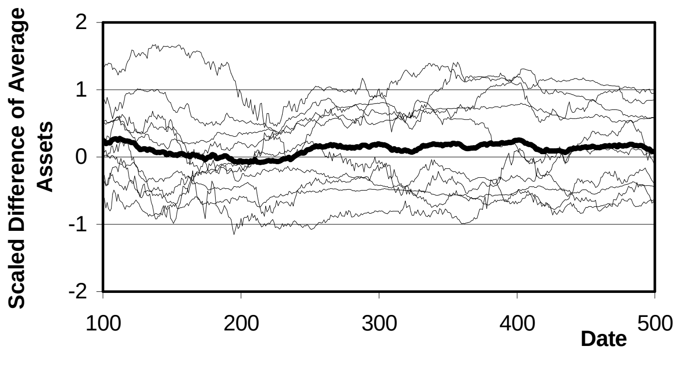

Fig. 11. Difference of average assets, **7x2Agents 4Stocks, Reverse**.

On the whole the above results do not indicate any solid evidence that prediction imparts any significant advantage over traders who do not predict. If any general trends are discernable the predictive traders did better when there where only 3 traders of each kind and only two stocks. It also did slightly better with the inflationary pricing mechanism than the reverse pricing mechanism. The table below shows the average of the average difference over the 500 time periods.

Table 2. A summary of the overall advantage (as the average over time of the average scaled difference of the average asset values of the two types of traders).

Number of agents of each kind	**Number of stocks that can be traded**	**Pricing mechanism**	**overall advantage to predictors**
3	2	Inflationary	0.25
3	2	Reverse	0.27
3	4	Inflationary	0.067
3	4	Reverse	-0.46
7	2	Inflationary	0.070
7	2	Reverse	-0.026
7	4	Inflationary	-0.098
7	4	Reverse	0.098

Momentum

I speculated that the reason that prediction did not seem to aid the agents was that there was no ‘momentum’ in the actions or their environment – in other words that everything can change direction instantly. It might be that prediction is more useful in circumstances where a significant level of momentum exists because then one has to anticipate the results – simple reaction will not be so effective because the effect of actions are delayed. In the next four sets of runs the set-up was the same except that the intentions of actions (that is the determined actions before they are moderated by what is possible for the agent) were delayed so that half the effect occurred immediately and half next cycle. That is each intended action is a 50/50 mix of those actions determined in this cycle and the last.

Four sets of runs were done with 3 agents of each type, with each pricing mechanism and with 2 or 4 stocks. The scaled difference of average asset values are shown in figures 12 to 15.

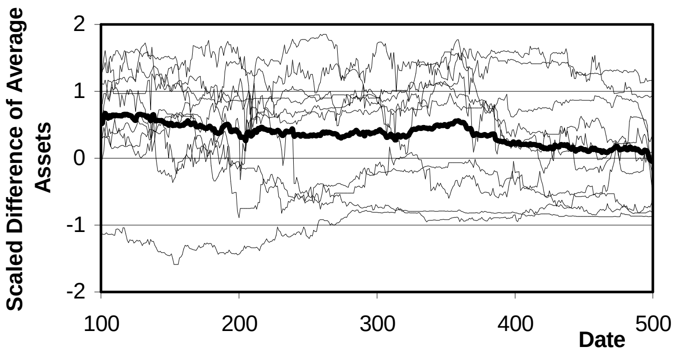

Fig. 12. Average assets of predictive traders minus average assets of non-predictive traders scaled in the simulation with action delay **3x2Agents 2Stocks, Inflationary**.

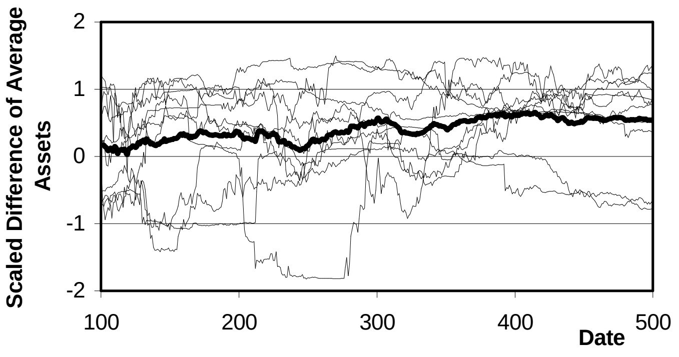

Fig. 13. Difference of average assets, **3x2Agents 2Stocks, Reverse**.

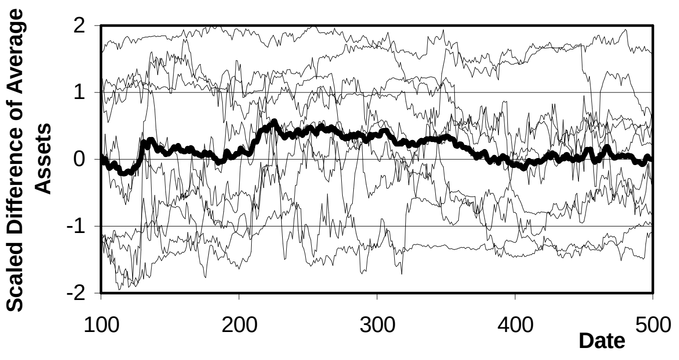

Fig. 14. Difference of average assets, **3x2Agents 4Stocks, Inflationary**.

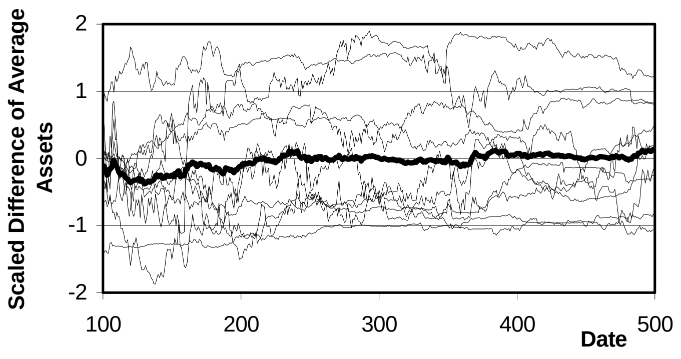

Fig. 15. Difference of average assets, **3x2Agents 4Stocks, Reverse**.

In each case the addition of momentum into the set-up did improve the overall performance of the predictive traders, but not by much. Table 3 below is the equivalent of table 2 for the set-ups with momentum.

Table 3. A summary of the overall advantage in the presence of momentum (as the average over time of the average scaled difference of the average asset values of the two types).

Number of agents of each kind	**Number of stocks that can be traded**	**Pricing mechanism**	**overall advantage to predictors**
3	2	Inflationary	0.36
3	2	Reverse	0.31
3	4	Inflationary	0.11
3	4	Reverse	-0.055

Four Longer Runs

To see what happened after a longer period of time I ran the **3x2Agents 2Stocks, Reverse** set-up four times for 5000 time periods. I chose this set-up because this seemed to indicate that the predictive traders had the most advantage in this set-up. The results are shown in figure 16. Here one sees that the initial advantage slowly dissipates until there is none by time period 3000. One also observes that the agents gradually lock into their relative positions – this is beacause there is limited stock and hence, once this is all owned and nobody wants to trade the market becomes quietscent. These runs do not indicate any permanent advantage of the predictive traders over the non-predictive traders, but it may be that what we are observing here is a sort of Baldwin Effect (see discussion below).

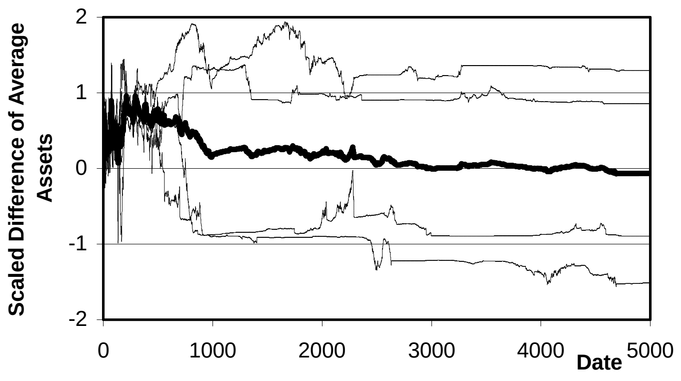

Fig. 16. Difference of average assets, **3x2Agents 2Stocks, Reverse**

Discussion

The simulations here do not indicate that prediction (in the form used here) always improves performance. Indeed this would be a surprising, even counter-intuitive result, if it had turned out to be the case. Rather the preliminary explorations highlight the importance of discovering the conditions under which prediction can aid adaptive action selection. Intuitively one would expect prediction to contribute to more successful decisions only if: (1) such prediction were feasible; and (2) that there was some advantage in knowing ahead of time what might occur compared to simply reacting to present events in the context of recent past events.

There are several reasons why prediction might not be significantly helpful here. It may be that prediction is simply not feasible in such markets. After all, if there is any pattern to the prices then a trader will learn to exploit this and the pattern will disappear. This is suggested by the fact that the predictive traders did better in the small markets with few stocks and more predicable inflationary prices.

It may be that we are observing something akin to the "Baldwin Effect" [2, 12], in which the advantage gained by learning is slowly incorporated into the genome over evolution. Here it might be that prediction is slowly learnt by the IALM module in a non-predictive fashion, cutting out the intermediate prediction and substituting a simple reaction to an observed pattern of events.

The results give weak support for the possibility that one of the factors is the amount of momentum in the system, for this would provide a clear need for prediction.

An area that I have not explored in this paper is a comparison with anticipatory systems where the prediction is tightly bound to a possible action. In anticipatory classifier systems the learning of how to exploit the predictions occurs in the 'chaining' of classifiers as they are successively applied. Thus the anticipation may well aid the development of such chaining. In the system I use here the *use* of the predictions is learnt separately to the predictions itself. As I had it the predictions came before the action selection so that the predictions could not make use of the intended action to anticipate its effect. In future explorations I hope to compare different ways of organising this, including a set-up where the predicted prices of various actions can be compared as part of the action selection.

Acknowledgements

Thanks to David Hales for helpful comments on this work and Scott Moss for many discussions which have informed my thinking in ways I could not possibly articulate. Thanks also to Steve Wallis for writing SDML and to Mark Mills for discussions on how real markets actually work.

References

1. Arthur, B., Inductive Reasoning and Bounded Rationality. American Economic Association Papers, 1994. 84: p. 406-411.
2. Baldwin, J.M., A new factor in evolution. American Naturalist, 1996. 30: p. 441-451. http://www.santafe.edu/sfi/publications/Bookinforev/baldwin.html.
3. Challet, D. and Y.-C. Zhang, Emergence of Cooperation and Organization in an Evolutionary Game. Physica A, 1997. 246: p. 407. http://xxx.lanl.gov/abs/adap-org/9708006.
4. Drescher, G.L., Made-up Minds - A Constructivist Approach to Artificial Intelligence. 1991, Cambridge, MA: MIT Press.
5. Edmonds, B., Modelling Bounded Rationality In Agent-Based Simulations using the Evolution of Mental Models, in Computational Techniques for Modelling Learning in Economics, T. Brenner, Editor. 1999, Kluwer. p. 305-332. http://cfpm.org/cpmrep33.html.
6. Edmonds, B., Developing Agents Who Can Relate To Us - putting agents in our loop via situated self-creation, in Socially Intelligent Agents, K. Dautenhahn, et al., Editors. 2002, Kluwer. http://cfpm.org/cpmrep58.html
7. Koza, J.R., Genetic Programming: On the Programming of Computers by Means of Natural Selection. 1992, Cambridge, MA: MIT Press.
8. Mills, M., Price fluctuations from the order book perspective: empirical facts and simple model. 1998. http://www.htcomp.net/markmills/order book and price fluctuation.pdf.
9. Palmer, R.G.e.a., Artificial Economic Life - A simple model of a stockmarket. Physica D, 1994. 75: p. 264-274.
10. Stolzmann, W. Anticipatory Classifier Systems. in Genetic Programming. 1998. University of Wisconsin, Madison, Wisconsin: Morgan Kaufmann.
11. Tolman, E.C., Purposive behavior in animals and men. 1932, New York: Appleton.
12. Turney, P., D. Whitley, and R.W. Anderson, Evolution, learning, and instinct: 100 years of the Baldwin effect. Evolutionary Computation, 1996. 4(3): p. iv-viii.

Appendix – Specification of the simulation

Original Sources

The original agent-based artificial stock market was [9]. The rationale for the style of learning used in the learning modules may be found in [5]. The dual instrumental-predictive module structure was suggested in [6] but for a different purpose.

Static Structure

There is one market, where a fixed number of stocks are traded. In this market there is one market-maker who sets global prices and with whom all trades are made. There are a number of traders who buy and sell stocks with a view to increasing (the total book value of) their assets. Each of the traders is one of two types: predictive and non-predictive . There are equal numbers of each type of agent.

Each agent holds an amount of each stock and some cash. They have two learning modules: the IALM and the PPM. Each of these modules is a Genetic Programming algorithm with its own small population of tree structures. The population is of fixed size.

Each member of the PPM population is a composite model with one tree for each stock. When the corresponding tree is interpreted it outputs the predicted price for that stock.

Each member of the IALM population is a similar composite tree with a couple of trees as 'feature extractors' and then one tree for each stock which outputs how much the trader would like to buy and sell of that stock.

Temporal Structure

The model progresses via a sequence of discrete trading cycles. All the traders trade (or not) each cycle in parallel. In each trader there are a fixed number of stages that the cognition works through in order to make a trading decision.

Important Parameters

- Number of trading cycles
- Number of anticipatory traders
- Number of non-anticipatory traders
- Size of the GP populations
- Initial depth of the GP trees
- Proportion of new GP population created by propagation
- Proportion of new GP population created by crossover
- Proportion of new GP population newly generated
- Nodes and terminals of the populations (see below)
- Number of trading cycles over which the GP trees are evaluated
- Initial cash of the traders
- Initial amount of each stock held by the traders
- Initial amount of each stock held by the market-maker

- Level of random noise in the price setting process
- Number of initial trading cycles where traders trade randomly

The nodes and terminals of the trees are the following:

- *PPM Nodes*: AND, averageDoneByLast, averageIndexOverLast, averageOccuredToStockLast, boundedByMaxPrice, divide, dividendOf, doneByLast, F, greaterThan, IBoughtLastTime, IDidLastTime, indexLag, indexLastTime, indexNow, indexTrendOverLast, ISoldLastTime, lagBoolean, lagNumeric, lastBoolean, lastNumeric, lessThan, maxHistoricalPrice, minus, myMoney, NOT, onAvIBoughtLastTime, onAvISoldLastTime, OR, plus, presentStockOf, priceDownOf, priceLastWeek, priceNow, priceUpOf, randomBoolean, randomIntegerUpTo, randomPrediction, T, times, totalStockValue, volumeLastTime;

- PPM Terminals: F, indexLastTime, indexNow, maxHistoricalPrice, myMoney, onAvIBoughtLastTime, onAvISoldLastTime, randomBoolean, randomPrediction, T, totalStockValue, volumeLastTime *plus*: the names of the traders, stocks and a random selection of constants;

- IALM Nodes: AND, averageDoneByLast, averageIndexOverLast, averageOccuredToStockLast, avTraderPredictedIndex, boundedByMaxPrice, divide, dividendOf, doneByLast, F, greaterThan, IBoughtLastTime, IDidLastTime, indexLag, indexLastTime, indexNow, indexTrendOverLast, ISoldLastTime, lagBoolean, lagNumeric, lastBoolean, lastNumeric, lessThan, maxHistoricalPrice, minus, myMoney, NOT, onAvIBoughtLastTime, onAvISoldLastTime, OR, plus, predictionFor, presentStockOf, priceDownOf, priceLastWeek, priceNow, priceUpOf, randomBoolean, randomIntegerUpTo, randomPrediction, statResult, T, talkResult, times, totalStockValue, volumeLastTime;

- IALM Terminals: avTraderPredictedIndex, F, indexLastTime, indexNow, maxHistoricalPrice, myMoney, onAvIBoughtLastTime, onAvISoldLastTime, randomBoolean, randomPrediction, T, totalStockValue, volumeLastTime *plus*: the names of the traders, stocks and a random selection of constants;

There is not room to explain every single primitive, so I will give some indicative examples and summarise others.

- AND, OR, NOT, T, F are the obvious Boolean operators;
- times, divide, plus and minus are the normal arithmetic operators (except that divide by 0 results in 0);

- `greaterThan` and `lessThan`, output a Boolean depending on the numeric comparison;
- those of form `random*` produce a randomised output;
- those of form `last*` shift the evaluation one time period back;
- those of form `lag*` shift the evaluation back an indexed number of time periods;
- `priceNow` outputs the present price of a stock;
- `indexNow` outputs the present price index (the average of all prices);
- `donByLast` outputs the action of another trader for a given stock;
- `myMoney` outputs the agents amount of cash;
- `averageDoneByLast` outputs the average action over stocks done by another agent;
- `boundedByMaxPrice` takes the minimum of the number and the maximum price;
- `dividendOf` outputs the current dividend rate of a stock;
- `IboughtLastTime` outputs a Boolean to represent whether the agent bought the given stock last time;
- `IdidLastTime` outputs the action for the given stock last time period;
- `predictionFor` outputs the prediction made for the current stock (by the PPM);
- `indexTrendOverLast` projects the next value given a linear trend over the last number of time periods;
- `presentStockOf` outputs the present level of a given stock owned by the agent;
- `totalStockValue` outputs the total value of the agent's stock.

The idea here is to provide agent with a rich set of primitives that include most of the operations they will need. If some turn out to be useless they will quickly be discarded by the internal evolutionary process in the learning modules. Below are a couple of examples to show what could be built from these.

- `[minus [priceNow 'stock-1'] [maxHistoricalPrice 'stock-1']]` – sell if price is greater than the maximum historical price otherwise buy;
- `[lagNumeric [2] [divide [doneByLast 'trader-2' 'stock-3'] [indexNow]]]` – do action of the action done by trader-2 for stock-3 divided by the price index 3 time periods ago.

Initialisation

The GP trees were initially randomly generated to a depth of 4 from the nodes and terminals available. For the first 5 cycles each trader makes small random trades, this has the effect of giving the GP process something to go on when it starts at cycle 6 and has the effect of damping down initial transient dynamics. The initial price of each stock and the initial dividend rate is randomly determined within a limited range.

Dynamics
Each trading cycle the following occurs in the market:

- Prices are set by the market maker, dividend rates are updated
- Each trader does the following (in parallel to other traders):
    1. Constructs new populations for its learning modules using propagation, crossover and new generation of trees (based on the fitness of trees last cycle).
    2. IALM evaluates its trees by how much its assets would be worth if the trading strategy of the tree was followed over a fixed number of past trading cycles (assuming historical prices).
    3. PPM evaluates its trees by their accuracy in predicting the next stock prices over a fixed number of past trading cycles (assuming historical prices). In the non-anticipatory version this is all done shifted one cycle back so that it is evaluated against predicting current stock prices.
    4. The best tree in PPM is picked and evaluated to make predictions of next stock prices.
    5. The best tree in IALM is picked and evaluate to decide intended trading decisions.
    6. Trading decisions are moderated depending on what is possible for the trader (e.g. only buying what it can afford).
- The market maker deals with the bids to buy and sell it stock. And updates the accounts (how much stock and cash traders have).

Results claimed as significant
That an ability to predict future prices is not always an advantage in this market.

Intended interpretation
The simulation is intended to be interpreted in terms of how an animat might learn in a dynamic environment in which it was embedded.

Implementation details necessary to get the simulation to run
The price setting mechanism of the market maker is basic and somewhat arbitrary. At the present it simply changes prices depending upon the net buying of the stock concerned during the previous trading period. A more sophisticated mechanism would be if the market maker tried to predict the net demand for each stock and guess the price necessary in order to fulfil the demand.

Implementation Language
The simulation was implemented in the social simulation language: SDML, version 4.1. This is freely downloadable for academic use. See `http://sdml.cfpm.org` for details.

Source Code
The source code is available as an SDML module at URL: `http://cfpm.org/~bruce/etvop/code`. It requires SDML version 4.1 or later to run.

Example Simulation Output
Example output from this simulation can be found at URL: `http://cfpm.org/~bruce/etvop/output`.