# GP Age-layer and Crossover Effects in Bid-Offer Spread Prediction

Amy Willis
Dept. Computer Science
UCL, Gower Street, London,
UK, WC1E 6BT
amywillis@gmail.com

Suneer Patel
Dept. Computer Science
UCL, Gower Street, London,
UK, WC1E 6BT
Suneer.Patel@barclayscapital.com

Christopher D. Clack
Director, Financial Computing
Dept. Computer Science
UCL, Gower Street, London,
UK, WC1E 6BT
clack@cs.ucl.ac.uk

## ABSTRACT

The bid-offer spread on equity options is a key source of profits for market makers, and a key cost for those trading in the options. Spreads are influenced by dynamic market factors, but is there also a predictable element and can Genetic Programming be used for such prediction? We investigate a standard GP approach and two optimisations — age-layering and a novel crossover operator. If both are beneficial as independent optimisations, will they be mutually beneficial when applied simultaneously? Our experiments show a degree of success in predicting spreads, we demonstrate significant benefits for each optimisation technique used individually, and we show that when both are used together significant detrimental over-fitting can occur.

## Categories and Subject Descriptors

I.2.M [**Artificial Intelligence**]: Miscellaneous

## General Terms

Algorithms, Experimentation

## Keywords

GP, Finance, Options, Spreads, Age Layers, ALPS, Crossover

## 1. INTRODUCTION

Genetic Programming (GP) offers an interesting alternative options pricing technique [3, 4, 5], using directed exploration of a vast search space of equations to find an effective solution that makes no assumptions about the market. The problem of predicting bid-offer *spreads* is harder still, and it is not clear whether there exists a predictable element based purely on the basic parameters of the option being traded.

Encouraged by the existence of the commercial product "AutoQuote" (see below), we suspect that a learnable component of spreads does exist — though AutoQuote may derive its predictions from dynamic market information to which we do not have access. We therefore set out to investigate whether a GP system could detect any learnable gradient based purely on the available underlying inputs to standard options pricing equations.

Exploration of this search space is a very hard problem for GP, and we have investigated two optimisations to determine their efficacy in the context of such a large and complex search space:

1. age-layering (the Age Layered Population Structure — ALPS [9]) is a new GP optimisation that achieves fitter solutions by preventing premature convergence; and

2. a non-standard crossover operator, which attempts to achieve fitter solutions by more directed exploration the search space.

These two techniques both aim to improve the fitness of the final GP solution, but in opposite ways — ALPS delays convergence, whereas non-standard crossover operators attempt to speed convergence through more intelligent search. We are interested to explore the effects of implementing both optimisations simultaneously.

In this paper we present the real-world problem of predicting bid-offer spreads, describe the two optimisation techniques, and present empirical evidence of their effect on the fitness of the evolved solution. We investigate use of each optimisation individually, and in conjunction, and comment on their relative effects on fitness.

### 1.1 Options pricing

Pricing equity options is difficult. Standard mathematical techniques such as the Black Scholes model [2] make assumptions about the market (volatility, distributions, continuity of trading, etc.) that are unrealistic and unsatisfactory [14, 15, 18]. Yet accurate pricing is extremely important — especially to the seller (writer) of an option, who (unlike the buyer) cannot choose not to exercise the contract and so risks large losses.

The Euronext LIFFE web site provides data for a wide range of equity options. It also provides a system ("AutoQuote") that predicts bid-offer spreads for those options, though AutoQuote gives a rather inaccurate prediction of these spreads.

Figure 1 shows the format of a Euronext LIFFE screen (http://www.liffe-data.com/). A range of call and put options for Ladbrokes at various strike prices are shown. The underlying stock price of Ladbrokes is shown at the top to


*GECCO'08,* July 12–16, 2008, Atlanta, Georgia, USA.

**Ladbrokes** **398.5**

**Expiry: 16Mar07**

**Calls**

| Settle | OI | Total Daily Vol | Vol | Last Trade at | Last Trade | Bid | Offer | AQ Bid | AQ Offer | Strike |
|---|---|---|---|---|---|---|---|---|---|---|
| 101.25 | 6 | 0 | - | - | - | - | - | 100.8 | 107 | 300 |
| 72.75 | 11 | 0 | - | - | - | - | - | 72 | 78.25 | 330 |
| 46 | 3 | 0 | - | - | - | 46.8 | 49.25 | 45.25 | 51.5 | 360 |
| 24.25 | 246 | 4 | 4 | ##### | 24.5 | 25.3 | 27.25 | 22.5 | 28.75 | 390 |
| 10.25 | 59 | 10 | 10 | ##### | 10.5 | 10.3 | 12.25 | 8.75 | 13.75 | 420 |
| 2.25 | 101 | 0 | - | - | - | - | - | 1.25 | 3.75 | 460 |
| 0.5 | - | 0 | - | - | - | - | - | 0 | 1.75 | 500 |

**Puts**

| Strike | AQ Bid | AQ Offer | Bid | Offer | Last Trade at | Last Trade | Vol | Total Daily Vol | OI | Settle |
|---|---|---|---|---|---|---|---|---|---|---|
| 300 | 0 | 1.75 | - | - | - | - | - | 0 | - | 0.5 |
| 330 | 0.5 | 3 | - | - | - | - | - | 0 | 130 | 2 |
| 360 | 3.75 | 7.5 | 4 | 6 | ##### | 5 | 10 | 10 | 45 | 6.25 |
| 390 | 11.5 | 16.5 | 13.3 | 15.25 | ##### | 13.75 | 3 | 3 | 298 | 15.25 |
| 420 | 28 | 33 | 29.3 | 31.75 | - | - | - | 0 | 26 | 32.25 |
| 460 | 59 | 65.25 | - | - | - | - | - | 0 | 30 | 64.75 |
| 500 | 98.3 | 104.5 | - | - | - | - | - | 0 | - | 104 |

**Figure 1: Example Euronext LIFFE screen, including AutoQuote data. The left side of the screen (Calls) is shown above: the right side (Puts) is below. The Strike column is shown twice for convenience.**

be 398.5 and the expiry date of the option is 16 March 2007. The four columns titled either AQ Bid or AQ Offer show the AutoQuote bid-offer spreads for both types of option (Calls and Puts). The columns titled Bid and Offer show the actual prices of the bid-offer spreads quoted by dealers. There are 3 cases for call and 3 for put where there is enough information to use as data, these are the ones corresponding to strike prices of 360, 390 and 420. They all illustrate the inaccuracy of AutoQuote pricing mechanisms.

The models that are used to predict the spreads are not divulged, but AutoQuote's inaccuracies may be due to it being based on standard parametric techniques. The instruments are American options, and pricing this type of option is more difficult than the simpler European options (for example, the Black-Scholes equation can, with its limitations, price European options but not American options).[1]

Genetic programming is a technique that requires minimal assumptions about the market. Used with data obtained from real-world trades at LIFFE, we set out to explore how a standard GP system and the two identified GP optimisations would respond to the challenge of predicting bid-offer spreads for American equity options.

# 2. RELATED WORK

## 2.1 Avoidance of premature convergence

The Age Layered Population Structure (ALPS) [9] is a new mechanism that aims to avoid premature convergence, and Hornby presents evidence of substantial success. By "premature convergence" we mean the tendency for a GP system to converge on a local optimum rather than continuing to search for a better (preferably global) optimum. A key contribution of ALPS is to enforce an age-layered structure on the population and to restrict breeding to individuals that occupy the same age layer. See Section 3.2.

Many techniques for the reduction of premature convergence and the preservation of diversity exist. The Hierarchical Fair Competition (HFC) model [10] is very similar to ALPS. It splits the population into layers and also introduces randomly generated individuals into the bottom layer. However HFC uses fitness rather than age to segregate individuals into layers. This results in the problem of individuals that have converged to a local optima near the top fitness layer preventing newer individuals in different basins of attraction from climbing through that fitness layer [9].

The multipopulation genetic programming [19] technique also uses a method based on the segregation of individuals. However, the system differs from ALPS and HFC as it involves an initial population split into subpopulations. The subpopulations evolve using differing mutation and crossover probabilities. Each subpopulation can communicate and transfer their best individuals every few generations. However, an individual can only move into a subpopulation with a lower fitness than its own. This would lead to a reduction in diversity given the newly migrated individual will have a higher fitness than the rest of the subpopulation and dominate the subpopulation during crossover and mutation. Therefore the rate at which individuals are allowed to move into new subpopulations must be carefully considered. There is also no introduction of randomly generated individuals and there is no guarantee that all subpopulations will not converge to the same local optimum.

## 2.2 Non-standard crossover

Many techniques have been proposed to improve the standard GP crossover operator. For example:

- Both homologous crossover [13] and size fair crossover [13] aim to reduce code bloat, the first by selecting nodes for crossover that are closest in position within their parent trees, and the second by selecting to cross over subtrees that are similar in size. These two techniques do not address fitness.
- Context preserving crossover [6] looks at only crossing over nodes from sections of the parents that are identical, as do one-point and uniform crossover [17]. These techniques are shown to speed convergence in problems with known solutions, but they can be overly restrictive, and prevent code from being moved into a position that may suit it better.
- Directed crossover [12] uses biases to choose crossover locations that avoid disrupting working code, crossover is biased in favour of changing code that appears to be performing poorly.

Novel crossover techniques that can modify GP search to improve the *fitness* of the final solution include:

- SSAC and SAMC [1] use a parameter tree for each individual of identical size and shape to determine the probability of crossover for each node. The parameter tree is updated with random noise each generation,

[1]Put simply, a European option can only be exercised on a stated date, whereas an American option can be exercised at any time up to a stated date and is therefore more difficult to price.

and if an update is beneficial to crossover this increases the fitness of the tree. However a detrimental change may inadvertently allow a good crossover point to go unnoticed. This technique doubles memory usage since two trees now have to be stored for each individual, which may be problematic for large problems. They measure fitness for a non-trivial problem (prediction of the number of sunspots, based upon historical data) and show that there is an increase in fitness of solutions using SSAC.

- Context-aware crossover [16] chooses a random subtree from the first parent and finds the best location in the second parent to place this subtree, with the intention of improving performance. No consideration is given to the strength of the subtree that is being inserted into the second parent, therefore it is possible the system merely finds the best location to insert a poor subtree. Also no second child is made, so a lot of genetic material is thrown away; most of the first parent, and a subtree from the second parent. The average fitness and fitness of the best individual is measured for several problems and an increase is shown.

- Selective crossover [8, 22] evaluates the contribution of each node to the overall fitness by removing the node and reevaluating the tree without it. The node that causes least change when removed is called the weakest node, and the one causing most change is called the strongest node. The weakest node in each parent is replaced by the strongest node from the corresponding parent, eliminating weak subtrees and combining strong ones. This method evaluates each tree $n$ times (where there are $n$ nodes), which causes a large increase in running time.

- Directed crossover [20] identifies the node whose contribution to the tree is maximal in each parent, and crossover is performed on these nodes, ensuring only highly operative segments of code are exchanged. It is not made clear how the contribution of a node to the tree is measured, just that fitness is measured and recorded at each node. Crossover biases [21] are used to select a node for crossover based upon fitness, and then find the best location to place it in the second parent. Again, there is no description of how to measure fitness, this paper just states that it is information that is freely available. Both of these papers measure the number of converging solutions for problems with known solutions and the classification accuracy for classification problems, and the average size of solutions is compared. However, details of implementation are not given, so it is difficult to reproduce their work.

## 2.3 Using GP to predict options prices

GP is increasingly being applied to problems in finance, and the prediction of market prices has been a particular area of interest. GP is an attractive *non-parametric* mechanism for deriving options prices from the underlying parameters that are thought to drive the price; it requires minimal assumptions, can adapt to changing market conditions, learns optima more flexibly and more accurately than SVM [23], and uses a directed search that is faster than random search (as used for example in Monte Carlo methods).

Chidambaran et al [4] propose a GP system to develop an adaptive evolutionary model of option pricing that is data-driven and non-parametric. They show that this method can operate on small data sets, thereby avoiding the large data requirement of the neural network approach (see for example Hutchinson et al [11]).

There have been several attempts to use GP to price options using simulated data; the Black-Scholes model is often taken to be the "true" model and is used to generate artificial data for training and testing. [5] is one such approach. Monte Carlo simulations are used to generate artificial stock price data; this is used to generate call options of varying strike prices and maturity. The call options are then priced using the Black-Scholes model. These sets of data are used to train and test the GP. The initial population is seeded with the Black-Scholes equation to see if this helps to develop a better model; it is discovered that this does help. Various parameter settings are tested, with the conclusion that smaller data sets that are stochastically changed, larger population sizes and higher mutation rates all lead to faster convergence of the problem. The paper concludes that their GP is suitable for use in the real world.

Though the study is an interesting exploration of the mechanisms of evolutionary search in a particular problem domain, and gives a good indication that evolutionary search is effective in such a solution landscape, using a GP to learn the Black-Scholes model has little real-world benefit — there would be nothing to gain in accuracy of price prediction from using the GP system over Black-Scholes model. Of course, if the evolved GP equation can price an option considerably faster than Black Scholes, then this could be beneficial when a very large number of options must be priced.

[3] is a study of option pricing using GP that does use real data. The data used is from S&P 500 index options. They are European options so the Black-Scholes model can be used to price them. Two implementations are used: "One-stage GP" simply trains on the data, and then validation occurs on out-of-sample data using a pricing formula obtained by averaging the prediction made by the top $k$ programs where $k$ = 10,50 and 100; by contrast "Two-stage GP" distinguishes between in-the-money (ITM) options and out-of-the-money (OTM) options. The data set is split into options which are ITM and options that are OTM. This is done to improve the performance of the GP, since ITM options are intrinsically more valueable than OTM options. Both techniques are compared to other methods of pricing options, and Two-stage GP outperforms all of the linear regression models and ANNs in the comparison. Two-stage GP also outperforms the Black-Scholes model in training, but not in testing. This experiment compares the errors from Black-Scholes calculations based upon the data (and compared to the actual trade prices) to the errors from the GP system.

Our problem is different to prior work in that we aim to predict bid-offer spreads, and our target is the actual quoted bid-offer spreads of the options dealers (as quoted on Euronext LIFFE). This is not as simple as predicting a mid price and then predicting the spread; indeed, there are elements to the setting of the spreads that could not be predicted either by us or by AutoQuote — for example, a dealer may modify the quoted spread depending on the dealer's exposure to risk.

# 3. DESCRIPTION OF THE ALGORITHMS

## 3.1 The GP system

Traditionally, the value of an option is held to depend upon five factors; the strike price at which the option can be exercised, the stock price of the underlying instrument, the time to expiry, the risk-free interest rate and the volatility of the stock price. A data set consisting of these five pieces of information along with the AutoQuote predictions for the bid-offer prices, and the bid-offer prices of actual trades has been collected. The GP system creates two sets of equations based on the five variables discussed. One set of equations will be for bid prices and the other for offer prices. The fitness function aims to minimize the mean squared error of the predictions from the actual values; thus, the fitness of each equation will be calculated as follows:

$$f = \frac{(\sum_{i=1}^{n}(x_i - a_i)^2)}{n} \quad (1)$$

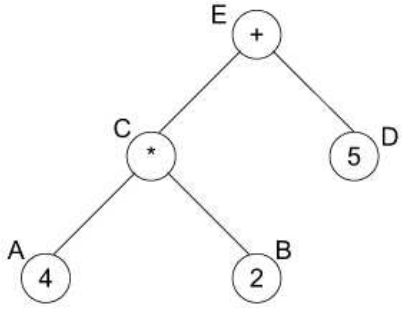
where $x_i$ represents the value calculated for the $i^{th}$ item in the data set, and $a_i$ represents the actual (bid or offer) price for the $i^{th}$ item in the data set. System parameters are given in Table 1.

## 3.2 ALPS

A common problem in GP is premature convergence, caused by individuals in the population being strongly drawn towards a high fitness individual that is close to a local (not global) optimum in the search space. New individuals need to be introduced to explore other parts of the search space allowing us to move away from local optimum and hopefully towards a global optimum. The problem with this in standard GP is the fitness of a new individual will be much lower than that of the old individual that has reached a local optimum, so the search will still be directed towards the old individual. ALPS is suggested as a method to overcome this problem [9]. It allows new randomly generated individuals to be integrated into the population and provides a mechanism to prevent these new individuals from being directed away from the new area of search space they are exploring. ALPS introduces age as a measure of how long an individual has been in the population. Age begins at 0, and is increased by 1 each time an individual is used to produce offspring, and the age of the offspring produced via mutation and crossover is the age of its oldest parent +1. Age is used as a restriction on breeding, hence competition between individuals is reduced. Age-layers are used to implement the restriction. The population is separated into multiple age-layers where each age-layer has a maximum age limit. Selection, breeding and replacement are restricted to adjacent age layers, and new individuals are regularly introduced into the initial age layer. Structuring the population this way allows individuals to explore newly discovered optimum in the search space, without being directed towards local optimum discovered by older individuals.

## 3.3 Novel crossover

The second optimisation aims to modify crossover so that the overall fitness of the population is increased. Standard crossover in genetic programming randomly selects a node from each parent and exchanges their subtrees. Doing this may destroy a strong subtree, if a point within that subtree is chosen for crossover. An alternative method is to determine the strongest subtree from each parent and perform crossover on a node above this subtree. Doing so would preserve the structure of the subtree. It is possible to determine which subtrees are strong by analysing the fitness of each node.




| Node | Value | Fitness |
|---|---|---|
| A | 4 | $(10-4)^2 = 36$ |
| B | 2 | $(10-2)^2 = 64$ |
| C | 8 | $(10-8)^2 = 4$ |
| D | 5 | $(10-5)^2 = 25$ |
| E | 13 | $(10-13)^2 = 9$ |

**Figure 2: Tree and fitness calculation for each node**

We present a novel crossover operator that has some similarity to Terrio's work [20] (though Terrio does not give sufficient details for a full comparison).

As each tree is evaluated the fitness of each node within the tree is calculated, and this is stored in the node for future use. The fitness of a node is calculated in the same way as the fitness of a tree; a mean squared error between the values of the node when evaluated, and the actual prices from the data set. A simplified example of this is shown in Figure 2, where fitness values of nodes are calculated for a single data item, with a target value of 10. The fitness value of each node depends upon how close the node was when evaluated to 10. The diagram shows node C as the node with the minimum error.

Parents are chosen using a tournament selection, and when crossover is performed the fitness values of the nodes are considered. For each parent, a pair of nodes is chosen for crossover. To choose this pair of nodes, the fitness values of all the nodes in the tree are compared allowing us to find the best one (in this case the best one is the one which minimises the error). This best node is called $n$. The second node, $c$ is a randomly selected node on the path from the root to $n$. Choosing a random $c$ on the path to $n$ allows the context in which $n$ is used to survive. We know that $n$ works well in this context because the overall fitness of the parent is high, otherwise the parent would not have been selected for crossover. If $n$ is the root node of the parent, then $n$ and $c$ are the same — see Figure 3.

We now have two parents, $p_1$ and $p_2$, and the pairs of nodes chosen from each parent, $n_1$ and $c_1$ from $p_1$, and $n_2$ and $c_2$ from $p_2$

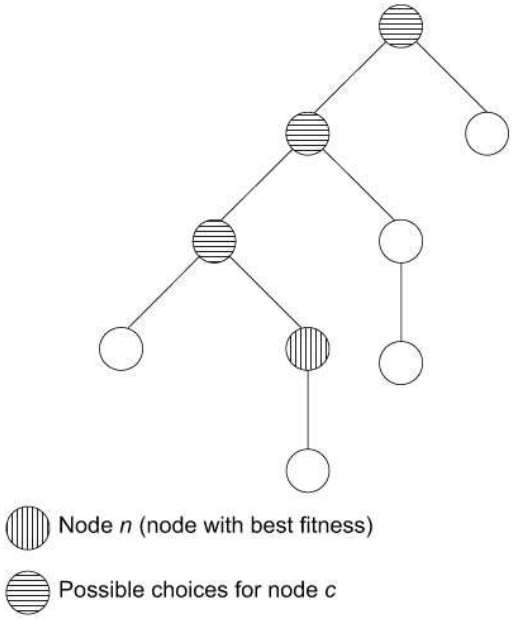



**Figure 3: Choosing the best node $n$ and a path node $c$.**

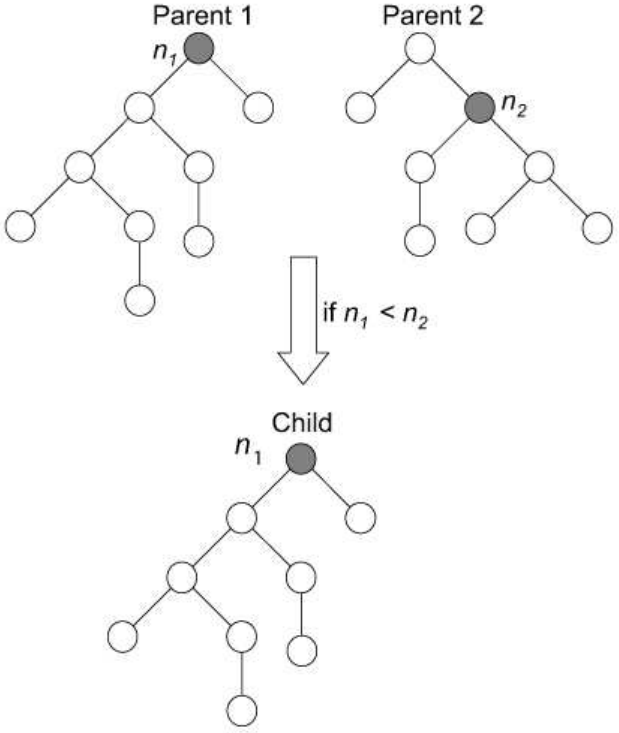



**Figure 5: Single child: replication of parent.**

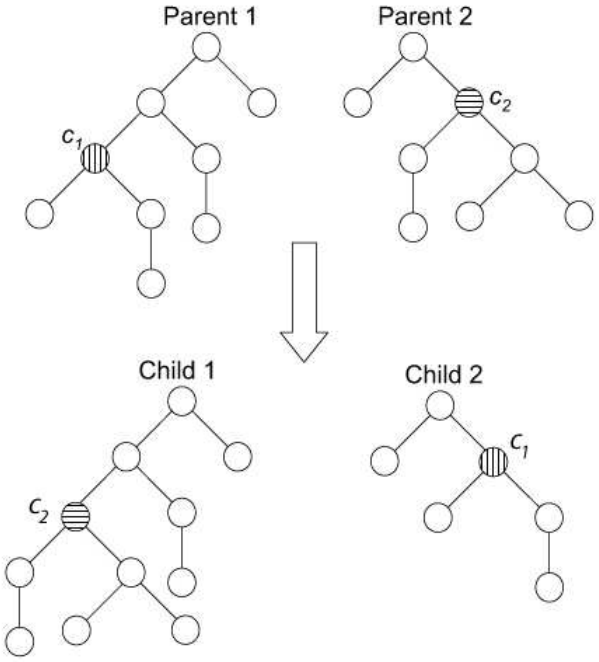



**Figure 4: Crossover of path nodes.**

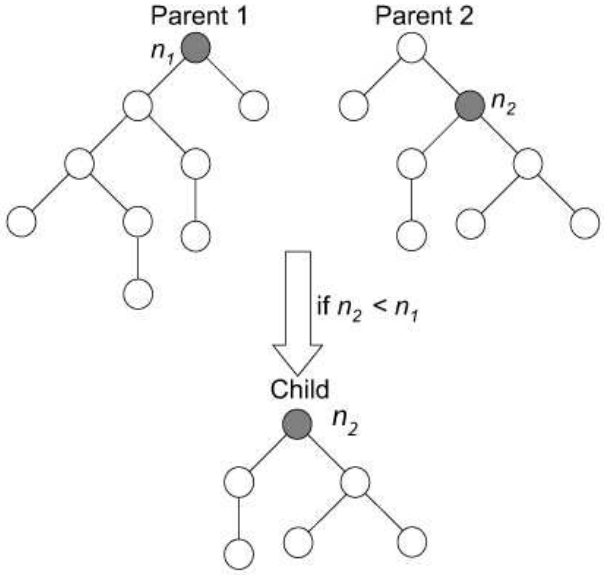



**Figure 6: Single child: subtree.**

Crossover can now be performed; a threshold fitness level has been set; if the fitness of $n_1$ or $n_2$ does not beat this threshold, standard crossover is performed, with new nodes being randomly selected. This allows random exploration — if a subtree is not particularly strong we do not want to preserve it. If the threshold has been reached the pair of selected nodes is used: a biased coin is flipped to decide which of the nodes should be used for crossover. An 80% chance is given to the nodes $c_1$ and $c_2$ being chosen. In this case a strong subtree is tested in a different position in the next generation, and it may perform better — see Figure 4. A 20% chance is given to the better out of $n_1$ and $n_2$ being chosen as one crossover point with the second crossover point being chosen as the root node of the corresponding tree.

In this case, if the better out of $n_1$ and $n_2$ was the root of its respective parent, this will simply replicate the parent — see Figure 5. Otherwise the better one will become the root of a child, as shown in Figure 6. This will throw away a lot of genetic material, hence it is only used a small amount of times.

## 4. EXPERIMENT

Four systems have been evaluated: Standard GP (SGP), ALPS, SGP with modified crossover (X), and ALPS with modified crossover (X+ALPS) (see Figure 7). In all cases, a generational GP is used, running for 500 generations with a population size of 500. Tournament selection is used with elitism — system parameters were empirically tuned by hand for SGP and then kept constant for ALPS, X and X+ALPS. See Table 1.

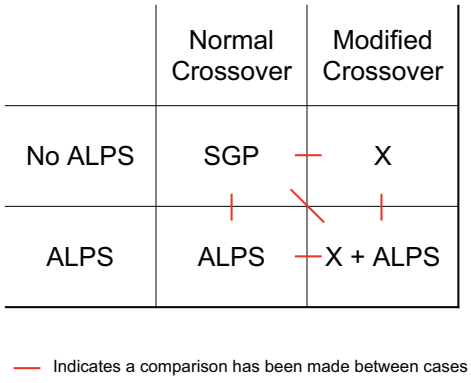



**Figure 7: Cross-comparison of ALPS and novel crossover**

The training data is a set of 500 real bid and ask prices for equity options taken from Euronext LIFFE. The data was collected over a period of 3 days for 100 different equity options. It consists of the five factors that affect the prices of options, predicted bid-offer spreads and actual bid-offer spreads. All data collected was randomly sorted to prevent the training and validation sets from containing large amounts of data that may be biased.

**Table 1: GP Parameter Settings**

| | |
|---|---|
| Population size ($N$) | 500 |
| Method of generation | Ramped half and haif |
| Function set | {+, -, *, /, Exp, min, max, power, SQRT, ln} |
| Terminal set | {stock price, strike price, time to expiry, volatility, risk-free interest rate,random numeric value} |
| Selection scheme | Tournament selection |
| Tournament size | 7 |
| Criterion of fitness | Mean squared error |
| Number of trees generated by elitism | 1 (0.2%) |
| Number of trees generated by crossover | 500 (100%) |
| Number of trees modified by mutation | 250 (50%)) |
| Termination criterion | 500-generation evolution |
| Maximum depth of initial generation | 7 |

We make multiple runs of all four systems and choose 25 evolved equations from each system. Distributions containing 25 data points each are large enough for statistical comparison; however, these distributions cannot be assumed to be Gaussian and so we use a non-parametric statistical test — the Ranked T-Test.[2]

The out-of-sample validation data consists of 200 data points.

## 5. MAIN RESULTS

Figure 8 shows the effect of the two optimisations on evolutionary convergence. It plots the mean performance of the best individual per generation, averaged across all runs, for each of the four experiments.

The first observation to be made from Figure 8 is that all four systems *are learning* — evolutionary pressure is causing the mean squared errors to improve. This supports the notion that there is some learnable component to bid-offer spreads, even from the basic data used in this experiment.

ALPS is shown to initially learn faster, although it converges to a similar level as novel crossover (X), and novel crossover with ALPS (X+ALPS). SGP is shown to learn more slowly initially and to converge at a higher (less fit) level. This is counter to our expectation that ALPS would converge more slowly.

Table 2 shows the minimum and average errors for the performance on the training data of the 25 individuals for each of the 4 experiments. Table 3 shows this same information, but for the performance on validation data. Tables 4 and 5 show the results of applying a ranked T-test to compare each of the five comparisons shown in Figure 7, on training and validation errors.

A (non-parametric) Ranked T-test is used to determine whether the means of the error distribution from the experiments are statistically different from each other. The p-value

[2]A valid alternative would be the Mann-Whitney-Wilcoxon test.

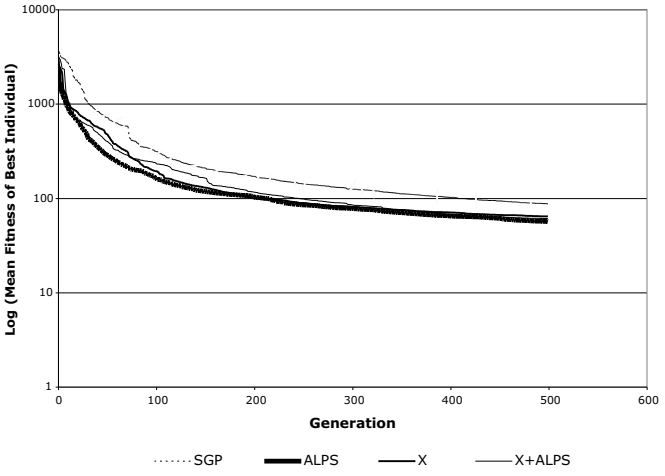



**Figure 8: Evolutionary convergence of the systems.**

**Table 2: Training Results**

| | Standard Crossover | | Modified Crossover | |
|---|---|---|---|---|
| | Min. error | Mean error | Min. error | Mean error |
| Standard GP | 55.06 | 87.87 | 41.25 | 64.31 |
| GP with ALPS | 49.49 | 56.83 | 57.52 | 61.502 |

represents the confidence that they are drawn from the same distribution; a small p-value suggests that we can be highly confident that there is a statistically significant difference. We consider anything under 0.05 to be significant.

The novel crossover operator (X) shows a significant improvement when compared against SGP on both training and validation results.

The results show that while ALPS makes no significant improvement on training, it does improve validation results significantly when compared to SGP. This suggests that the mechanisms used in ALPS have reduced over-fitting during training.

The situation with X+ALPS is more complex:

- *X+ALPS compared with ALPS*: ALPS performs significantly better on both training and validation than X+ALPS.
- *X+ALPS compared with SGP*: X+ALPS produces significant improvements during training, but no significant difference in validation results, when compared to SGP. We infer that the improvement in training results was the result of over-fitting.
- *X+ALPS compared with X*: X+ALPS produces significantly better results on training than X alone, but in validation, this result is reversed: using X alone produces better results.

To summarize: either technique used on its own gives significant benefit during validation, but the combination of the two techniques does not perform well.

**Table 3: Validation Results**

| | Standard Crossover | | Modified Crossover | |
|---|---|---|---|---|
| | Min. error | Mean error | Min. error | Mean error |
| Standard GP | 58.01 | 107.14 | 42.0998 | 71.27 |
| GP with ALPS | 39.81 | 66.93 | 71.1775 | 79.34 |

**Table 4: Training Significances**

| Test | P-Value | Winner |
|---|---|---|
| SGP vs ALPS | $2.58\times10^{-1}$ | |
| SGP vs X | $6.20\times10^{-3}$ | X |
| SGP vs X + ALPS | $1.05\times10^{-4}$ | X + ALPS |
| X vs X + ALPS | $1.15\times10^{-2}$ | X + ALPS |
| ALPS vs X + ALPS | $6.58\times10^{-4}$ | ALPS |

# 6. SUMMARY AND CONCLUSION

We have investigated the possibility that there is a GP-learnable component to predicting American options bid-offer spreads in the Euronext LIFFE market, coupled with an investigation of the effects of using two optimisation methods and how they interact when used simultaneously on this problem.

The first result from our experiments is that the progressive reduction in errors from generation to generation indicates that a GP-learnable component to bid-offer spreads does indeed exist. Of course, the errors are still high even after many generations and this raises interesting questions for further work — is this the best that we can achieve using the described basic input data? what further (public) input data could we present to the GP system in the expectation that errors might be reduced further? is it possible to conjecture what additional data is used by the AutoQuote system? and is it possible that a GP system could, with access to the same data, provide more accurate predictions than AutoQuote?

Of the two optimisation techniques investigated, one is a novel crossover operator and the other is ALPS. We have compared how the two techniques perform alone, and alongside each other, against a standard GP system. We have demonstrated that when either optimisation is used individually it significantly improves validation results (e.g. when ALPS is used alone, over-fitting during training is reduced), but when used in combination they increase over-fitting to the training data. We conjecture that using the two tech-

**Table 5: Validation Significances**

| Test | P-Value | Winner |
|---|---|---|
| SGP vs ALPS | $2.74\times10^{-2}$ | ALPS |
| SGP vs X | $2.74\times10^{-2}$ | X |
| SGP vs X + ALPS | $8.11\times10^{-1}$ | |
| X vs X + ALPS | $3.924\times10^{-2}$ | X |
| ALPS vs X + ALPS | $4.52\times10^{-8}$ | ALPS |

niques together encourages more exploration of local optima by the novel crossover technique, and this may prevent the exploration of a better optimum. Reducing the amount of novel crossover occurring in the younger age-layers might provide a solution to this problem.

## 7. ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their helpful comments.

## 8. REFERENCES

[1] P. J. Angeline. Two Self-Adaptive Crossover Operators for Genetic Programming. *Advances in Genetic Programming 2*, pp. 89–110, MIT Press, 1996.

[2] F. Black and M. Scholes. The Pricing of Options and Corporate Liabilities. *Journal of Political Economy*, 81(3): 637–654, 1973.

[3] S-H. Chen, C-H. Yeh and W-C. Lee. Option Pricing with Genetic Programming. *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pp. 32–37, Morgan Kaufmann, 1998.

[4] N. K. Chidambaran, C. H. J. Lee and J. R. Trigueros. An Adaptive Evolutionary Approach to Option Pricing via Genetic Programming. *Computational Finance — Proceedings of the Sixth International Conference*, editors: Y. S. Abu-Mostafa, B.LeBaron, A. W. Lo, and A. S. Weigend, Cambridge, MA, MIT Press 1999.

[5] N. K. Chidambaran. Genetic programming with Monte Carlo simulation for option pricing. *Genetic Programming 1998: Proceedings of the Third Annual Conference*, Vol. 1, pp. 285–292, IEEE, 2003.

[6] P. D'haeseleer. Context preserving crossover in genetic programming. *IEEE World Congress on Computational Intelligence*, Vol. 1, pp. 256–261, IEEE Press, 1994.

[7] A. Ekart and S. Z. Nemeth. Maintaining the Diversity of Genetic Programs. *Proceedings of EuroGP 2002*, LNCS 2278, pp.162–171, Springer-Verlag, 2002.

[8] S. Hengproprohm and P. Chongstitvatana. Selective Crossover in Genetic Programming. *ISCIT International Symposium on Communications and Information Technologies*, 2001.

[9] G. S. Hornby. ALPS: the age-layered population structure for reducing the problem of premature convergence. *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, Vol. 1, pp. 815–822, ACM Press, 2006.

[10] J. J. Hu and E. D. Goodman. The Hierarchical Fair Competition (HFC) Model for Parallel Evolutionary Algorithms. *Proceedings of the Congress on Evolutionary Computation 2002*, pp. 49–54, IEEE Press, 2002.

[11] J. Hutchinson, A. Lo and T. Poggio. A Nonparametric approach to the Pricing and Hedging of Derivative Securities Via Learning Networks. *Journal ofFinance*, Vol. 49, 1994.

[12] W. B. Langdon. Directed Crossover within Genetic Programming. *Advances in Genetic Programming 2*, Number RN/95/71, 1995.

[13] W. B. Langdon. Size Fair and Homologous Tree Genetic Programming Crossovers. *Genetic Programming and Evolvable Machines*, Vol. 1, No.1/2, pp. 95–119, 2000.

[14] J. D. Macbeth and L. J. Merville. An empirical estimation of the Black-Scholes call option pricing model. *Journal of Finance*, Vol. 34, 1980.

[15] J. D. Macbeth and L. J. Merville. Tests of the Black-Scholes and Cox call option valuation model. *Journal of Finance*, Vol. 35, 1980.

[16] H. Majeed and C. Ryan. Using context-aware crossover to improve the performance of GP. *GECCO 2006: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, Vol. 1, pp. 847–854, ACM Press, 2006.

[17] R. Poli and W. B. Langdon. On the Search Properties of Different Crossover Operators in Genetic Programming. *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pp. 293–301, Morgan Kaufmann, 1998.

[18] M. Rubinstein. Implied Binomial Trees. *Journal of Finance*, Vol. 49, 1977.

[19] L. Tang, M. Li and J. Zhang. Multipopulation Genetic Programming For Forecasting Crop Pests. *IEEE Int. Conf. Neural Networks and Signal Processings 2003*, pp.554–557, 2003.

[20] M. D. Terrio and M. I. Heywood. Directing Crossover for Reduction of Bloat in GP. *IEEE CCECE 2003: IEEE Canadian Conference on Electrical and Computer Engineering*, pp. 1111-1115, IEEE Press, 2002.

[21] M. D. Terrio and M. I. Heywood. On Naive Crossover Biases with Reproduction for Simple Solutions to Classification Problems. *Genetic and Evolutionary Computation, GECCO-2004*, Vol. 3103 of *Lecture Notes in Computer Science*, pp. 678–689, Springer-Verlag, 2004.

[22] K. Vekaria and C. D. Clack. Genetic Programming with Gene Dominance. *in J.Koza (ed) Late breaking papers at the Genetic Programming 1997 Conference*, pp 300, Stanford University, ISBN 0-18-206995-8, 1997.

[23] W. Yan, M. Sewell and C. D. Clack. Learning to Optimize Profits Beats Predicting Returns — Comparing Techniques for Financial Portfolio Optimisation. *Proc. Genetic and Evolutionary Computation, GECCO-2008*, ISBN 978-1-60558-131-6, ACM, 2008.