Interactive Estimation of Agent-Based Financial Markets Models: Modularity and Learning

Ihsan Ecemis
CoalesiX
10 Fawcett Street
Cambridge, MA 02138, USA
ihsan@coalesix.com

Eric Bonabeau
Icosystem
10 Fawcett Street
Cambridge, MA 02138, USA
eric@icosystem.com

Trent Ashburn
Tiger
10 Fawcett Street
Cambridge, MA 02138, USA
thashburn@yahoo.com

ABSTRACT
Building upon the interactive inversion method introduced by Ashburn and Bonabeau (2004), we show how to dramatically improve the results by exploiting modularity and by letting the computer learn user preferences.

Categories and Subject Descriptors
I.2.6 [**Learning**], I.2.8 [**Problem Solving, Control Methods, and Search**]

General Terms
Algorithms, Economics.

Keywords
Agent-based modeling, interactive evolution.

1. INTRODUCTION
There have been a number of attempts over the last decade to model financial markets with agent-based modeling (ABM) [1,4,6,9,14,15,16,17,19]. In ABM, systems are modeled as collections of autonomous decision-making entities, called agents. Each agent individually assesses its situation and makes decisions based upon a set of rules. Agents may execute various behaviors appropriate for the system they represent –for example, buying or selling. Repetitive interactions between agents are a feature of ABM, which relies on the power of computers to explore dynamics out of the reach of pure mathematical methods [7]. The power of ABM lies in their ability (1) to let the modeler describe behavior in very natural terms and (2) to capture emergent phenomena. A financial market seems to be a natural fit for ABM. The dynamics of the stock market results from the behavior of many interacting agents, leading to emergent phenomena that can be understood using a bottom-up, ABM approach.

While ABM is useful in producing market-like aggregate-level patterns from individual-level rules, the main issue in financial markets ABM is calibration and validation: how can one evaluate the quality of a model, from both a structural perspective (how sound is the model?) and from an econometric perspective (how well does that model reproduce the data quantitatively?). In practice, calibration and validation of financial markets ABM are often neglected. One reason is that the use of purely numerical scoring methods to evaluate data fit and guide the search for explanatory models constrains the search path so dramatically that no good fitting model is found or the best fit is generated by a low-plausibility model. In many situations the fitness function for a model cannot be practically formulated mathematically. This problem can be overcome if one can allow more subjective factors to guide the search for "good" models, enabling ABM users to integrate financial economics expertise into their models.

Boschetti & Moresi [5,24] have proposed to replace the numerical evaluation of data fit by a subjective evaluation. A technique originally developed to generate "interesting" images and pieces of art [3,8,20,21,22] is used to perform model inversion by integrating subjective knowledge into the evaluation process. The technique (see [23] for a review) is a directed search evolutionary algorithm which requires human input to evaluate the fitness of a pattern (here, the fitness might be how well the model reproduces the data *qualitatively*) and uses common evolutionary operators such as mutation and crossover to breed the individual-level rules that produced the fittest collective-level patterns. Interactive evolutionary computation (IEC), as this technique is known, combines computational search with human evaluation [23].

In this paper we present an application of IEC to financial markets ABM inversion. One example illustrates how IEC can be used to discover the parameters of an agent-based model of a financial market from aggregate observations. The user operates a visualization tool to navigate a parameter space using selection and variation operators. Parameter values define traders and their trading strategies which, in turn, generate a synthetic price history. The user's goal is to find a combination of parameter values that can reproduce a target price history. The experiment utilizes a target price history from a market frenzy that occurred on the London Stock Exchange in September 2002. IEC is used in this case to obtain a qualitative fit.

2. AGENT-BASED MODEL
A simple model of a financial market is used in both experiments. Each model simulation includes its own order management and clearing mechanism (an order book), traders operating trading strategies, a market maker posting orders on the book which are

GECCO'05, June 25-29, 2005, Washington, DC, USA.

matched with traders' orders, and a price history. At the end of each time step, after all trading and market-making, the arithmetic mean of the best bid and best ask is appended to a simulation's price history.

2.1 Order book

The order book matches orders in a continuous double-sided auction. It holds and clears both limit orders and market orders, where clearing is the action of matching a bid with a same-priced ask and removing these from the book. Same-priced limit orders are matched with other limit orders or market orders. In the process of clearing, the size (number of shares) of each matched order is reduced by the amount of the smaller-size order. Orders with size zero are removed from the book. Upon order submission, the book immediately clears any and all clearable orders. Orders are characterized as follows: (i) Bid: an offer to buy; (ii) Ask: an offer to sell; (iii) Best bid: the highest bid; (iv) Best ask: the lowest ask; (v) Limit order: a bid or ask at a specific price or better; (vi) Market order: an offer to buy at the best ask or sell at the best bid.

2.2 Traders

At each time step, traders take turns trading according to their strategies. Traders trade once per time step, each trader submits only market orders, and traders' strategies (described below) are based upon a set of initial conditions and the accumulated price history of the stock. In each time step, after each trader trade, the market maker trades.

Market maker. To ensure liquidity in the market, there is one market maker in each simulation, tightening the spread and maintaining order depth. The *spread* is the difference between the best bid and the best ask. When a trader trades, bids or asks are often cleared from the book, which widens the spread in a particular direction. *Spread tightening* is the action of the market maker adding bids if the trader had bought, or adding asks if the trader had sold, until the spread is a specific dollar amount. In our experiments, the market maker separates all its orders by $1 increments, and attempts to maintain a spread of $1. *Order depth* refers to the number of uniquely-priced bids and asks that the market maintains on the book. This buffers trading and must be of a sufficient depth to handle all trading; several hundred orders suffice in our experiments.

Traders in this model come in four flavors: fundamentalists, chartists, noise traders, and a whoops trader. Each has unlimited buying power, and unlimited shorting power (in the real world, shorting is betting that the stock is going down in value, using borrowed stock; covering is buying back the stock and returning it to the lender. For our purposes, shorting and covering are simply a different kind of selling and buying, respectively). Also, only one trade can be made per time step, per trader.

Fundamentalists. Fundamentalists calculate a "true value" of the underlying stock by averaging the price history a certain number of minutes back – a moving average of the form $T = \sum_{x=r}^{r-l} p_x$ where T is the true price, p is a historical price, r is the time of the most recently recorded historical price, and l is the number of time steps used in the computation. The trader then computes an upper threshold, U, and a lower threshold, L. Each is a percentage t away (50%=0.50) from the true price: $U = T(1+t)$ and $L = T(1-t)$. Once the price action has passed through one of the thresholds, a number of time steps must pass (greater than the fundamentalist's reaction time) before the fundamentalist is "awake" and able to trade. Once this reaction time threshold has been passed, the fundamentalist makes one trade per time step according to the rules:

- *Buy* if $A \le L$ and a long position is not held
- *Sell* if $B \ge T$ and a long position is held
- *Short* if $B \ge U$ and a short position is not held
- *Cover* if $A \le T$ and a short position is held

where B is the best bid, and A is the best ask. The trader holds a long position if he has bought, and he no long holds it if he sells; likewise for the short side. Characteristically, a fundamentalist appears to anchor the price history and dampen volatility.

Chartists. Chartists' trading is triggered by momentum – they trade in the direction of a trend when the trend is steep enough. The momentum M is computed: $M = (p_r / p_{r-l})/l$, where l is the number of time steps used in the calculation, p_r is the most recent historical price, and p_{r-l} is the price l time steps before p_r. A chartist's trading rules are:

If the number of my previous trades is less than the maximum allowed, then:

- *Buy* if $M \ge p_r(1+T)$ and the trader does not hold a long position
- *Sell* if $M \le 0$ and the trader holds a short position *Short* if $B \ge U$ and a short position is not held
- *Short* if $M \le p_r(1-T)$ and the trader does not hold a short position
- *Cover* if $M \ge 0$ and the trader holds a short position

where p_r is the most recent historical price and T is the threshold (5%=0.05). Buying, selling, shorting, and covering each count as one trade. Characteristically, chartists appear to reinforce trends and enhance both volatility and waves.

Noise Traders. Noise traders either buy or sell with equal probability in each time step. Noise traders are meant to reflect the apparent randomness in the real markets and act to move the price action around, in effect "tripping" the strategies of the other traders.

Whoops Trader. The whoops trader places one 10,000-stock order to buy at the market at 10:15am which represents the alleged trading mistake that happened on September 20, 2002 at 10:10am. While the trading mistake is reported to have occurred at 10:10am [18], orders on the London Stock Exchange may take many minutes before they are observed by other traders who then may take minutes more to act on that information. Because our synthetic traders immediately see information and react just as quickly, the simulation time of the mistaken order is adjusted to 10:15am.

Each simulation begins with the market maker quoting a best bid at $3820, best ask at $3821. All the market maker's orders are in lots of 100, and each simulation is run for the duration of the real dataset – from 9:00am to 11:30am in 1-minute increments.

3. INTERACTIVE EVOLUTION

The IEC search method employs a genetic algorithm and a graphical user interface to facilitate the user acting as the fitness function. A small initial population of agent based models is generated with random parameter values. The resulting price histories are generated by running the models, and then shown to the human observer. The observer selects interesting patterns according to whatever objective and subjective criteria the observer may be using to visually compare candidates with the target. These selections are considered the fittest individuals of the generation. The user configures a set of operators - elitism, mutation, and crossover – which are used to produce a new generation of models from the user-selected "fittest" individuals in the previous generation. The new generation is then simulated and the resulting price histories are again presented to the observer. This procedure is iterated until interesting patterns emerge from the search that more closely match the target. Over multiple generations the population may converge toward the target.

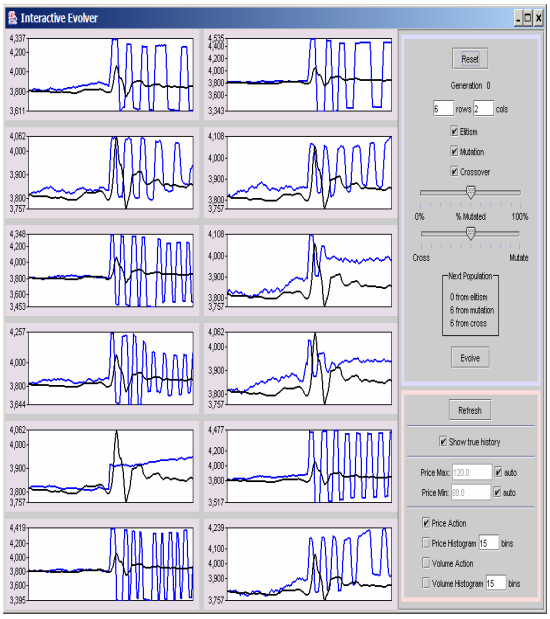

Figure 1. User Interface.

3.1 User Interface

The user interface, shown in Figure 1, is a critical component of the method, which depends crucially upon the user's ability to evaluate visualizations of the candidate solutions [23] –obviously this method can only work if the population size is kept small and if interesting patterns emerge after a reasonably small number of generations. The user interface shows the price histories of each candidate model in the left window, each overlaid on top of the target price history. On the right are two controls panels: (1) the main IEC control panel configures the size of the left-hand visualization grid, toggles operators, controls the chance that a gene is mutated, and controls the proportion of crossover versus mutation; the evolve button produces the next generation; (2) the second control panel controls model drawing.

3.2 Evolutionary Algorithm

Genotype representation. Each model simulation and its price history can be considered a phenotype, generated from the set of trading strategies and their parameters used in the simulation (the genotype). The parameters of these strategies vary across genotype, and it is the composition of these genotypes that we are interactively evolving. The genotype of a model has genes that are numbers – initially random numbers. Each number codes for a specific aspect of the set of traders (how many of each kind of trader, parameters of each strategy) and has bounds specific to the aspect it codes for. For example, a gene that codes for the number of traders might be bounded between 10 and 100. The trading strategies of each trader are designed to make possible a wide variety of behaviors that can potentially produce the target pattern. The genotype of each simulation is composed of genes that are numbers that code for what are considered random variables – either how many of one kind of trader (the random variable is a constant in this case) or the α 's and β 's for a beta distribution. Random values drawn from the Beta distribution serve as specific parameters to traders. In this way, only α and β describe a distribution of values used to parameterize any number of traders, compressing the genotype. The Beta probability distribution has non-zero values and takes the form:

$$p(x) = \left(\Gamma(\alpha+\beta)/\Gamma(\alpha)\Gamma(\beta)\right)x^{\alpha-1}(1-x)^{\beta-1}$$

where x is real number in the interval **[**0,1**],** $p(x)$ is the probability at x, and $\Gamma(n)$ is Euler's gamma function. We chose the beta distribution for our purposes because it is a bounded distribution that can take many shapes, it is nicely parameterized by just α and β, and it is easily scaled to cover each variable's legal bounds. Note also that a beta distribution with $\alpha = 1$ and $\beta = 1$ is equivalent to a uniform distribution. Alphas and betas are bounded between 1 and 100, and each parameter has an upper and lower bound for values drawn from this distribution which scales the corresponding beta distribution. Constants, α 's, and β 's comprise a genotype, and the parameter values drawn from these random variables and their bounds define the search space. The bounds and distributions for the parameters are defined as follows:

Fundamentalists: # of traders is a constant between 10 and 50; Trade size, Reaction time, Moving average length and threshold percentage are all drawn from Beta distributions within bounds [100,500], [1,50], [20,100] and [0%,5%], respectively.

Chartists: # of traders is a constant between 10 and 100; Trade size, Maximum # of trades, Momentum look back and Momentum threshold percentage are all drawn from Beta distributions within bounds [100,1000], [1,200], [1,5] and [0.1%,1%], respectively.

Noise traders: # of traders is a constant between 1 and 10; Trade size is drawn from a Beta distribution within bounds [100,200].

Selection. Candidates are either selected or not, and non-selected candidates are discarded. The user may turn on and off each of three operators to produce next generation members - elitism, mutation, and crossover. Elitism is applied first; then crossover and mutation are performed on randomly chosen selected members to produce remaining members. The number of new members produced by crossover versus the number produced by mutation is a proportion chosen by the user.

Elitism. Selected candidates are copied to the new generation.

Mutation. The user controls the chance that each gene is mutated - between 0% and 100%. As the mutation algorithm iterates through each gene, a random number between 0.0 and 1.0 is chosen. If that number is less than the mutation percentage (expressed as a decimal, 50% is 0.50), that gene is mutated to a new value within the bounds of that variable. Mutation occurs on each gene if rand<chance, where rand is a random number in the interval [0,1] and chance is the user-selected chance that a gene is mutated. New values for a given gene are chosen as follows: v_n=*min*+*rand**(max-min)*, where v_n is the new value, *min* is the lower bound for this gene, *max* is the upper bound for this gene, and *rand* is a random number in the interval [0,1].

Crossover. Double-point crossover chooses crossover points in a uniform random fashion and produces a new candidate by recombining two parent genotypes from among the selected candidates.

4. EXPERIMENTS AND RESULTS

4.1 Experiments

A market frenzy is a large deviation from the efficient market hypothesis for a relatively short period time, usually triggered by an anomalous event such as a trading error, a glitch in the trade processing information system, a significant piece of news, etc. Although such events are unusual, they do happen. One example is the market frenzy that occurred on September 20, 2002 at the London Stock Exchange and lasted for about 20 minutes and generated losses on the order of £100M for some of the players [2]. The trade volume (£3.2B) of this 20-minute event was larger than the volume of an average trading day. According to [18], "the trade activity (trades/second) was so high that some computer systems failed to cope and prices of shares were delayed" – probably leading to an amplification effect. The event began at 10:10 am and within 5 minutes the FTSE100 index rose from 3,860 to 4,060. Within another few minutes, the index fell to 3,755, before returning to a value slightly above its original level at the end of the 20 minutes (Figure 2).

Although Muchnik and Solomon [18] tell us that "it is believed that the event was triggered by a huge order (£1.2B) mistakenly submitted twice (or even three times) causing all trade participants to try to exploit the opportunity," the exact course of events is not precisely known; the players don't like to publicize their mistakes. The hypothesis that the event was triggered by a glitch is likely, given the return to a stable state, but according to various private sources the glitch is actually a combination of a trading error and an information system problem. In that context, our formulation of the problem we wish to address is: what is the nature of the initial trigger and what are the likely behaviors of the different players that can generate such a deviation from the market's stationary behavior? Our model will assume that one large order is being submitted, producing the necessary trigger to push the market out of its stationary state "without affecting its fundamental dynamical parameters", following Muchnik and Solomon's [18] plausible assumption.

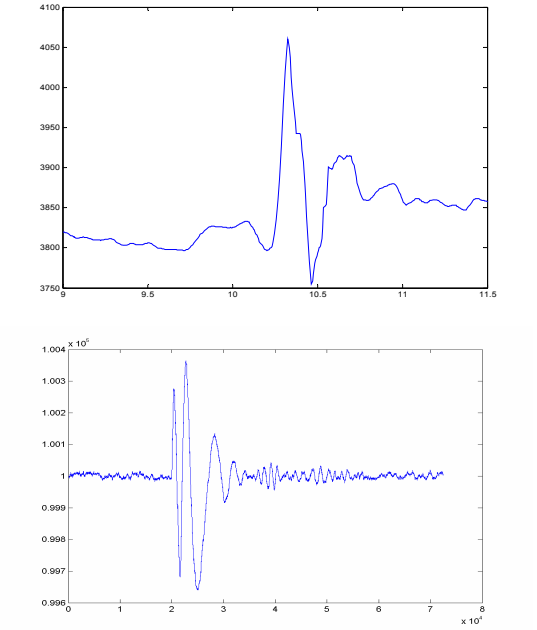

Figure 2. Top: FTSE 100 dynamics following a likely trading mistake on September 20, 2002. Bottom: From Muchnik and Solomon's model [18] with a trend-dependent panic factor. Modified from [18].

The goal of IEC here is to determine the nature of the herding behavior that could produce the event. While Muchnik and Solomon [18] offer an elegant first-order explanation of the phenomenon (an oscillator model akin to that of a guitar string), their model produces several oscillations around the stationary level before stabilization (Figure 2), while in the real data there is only one oscillation before stabilization. The IEC approach will be used to design a model that produces a better qualitative fit with the real data.

4.2 Results

It was possible to match the target price history qualitatively in an average of 10 generations, from a variety of starting populations. While "qualitative match" is an imprecise term, the characteristics we attempted to match are the amplitude, period, phase, and damping rate of the approximate wave – and of course the size, shape, and location of the price history. We matched reasonably well on each of these dimensions (Figure 3). The parameter values of the best solution are:

Noise traders

Number of traders (constant) = 8

Trade size (beta distrib.): α=97.02, β=49.75

Chartists

Number of traders (constant) = 27

Trade size (beta): α=12.68, β=40.16

Threshold % (beta): α=92.52, β=25.52

Look back (beta): α=93.64, β=30.77

Max trades (beta): α=41.28, β=73.28

Fundamentalists

Number of traders (constant) = 45

Trade size (beta): α=55.01, β=59.92

True price look back (beta): α=65.76, β=28.95

Percent away (beta): α=27.3, β=28.24

Reaction time (beta): α=99.35, β=83.8

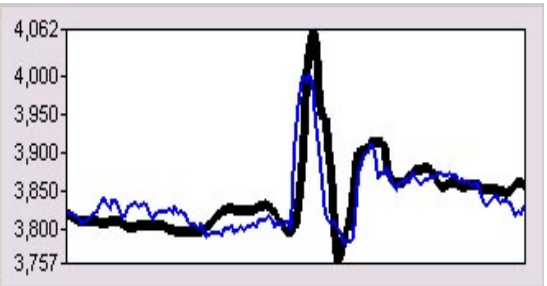

Figure 3. Best solution with data to be fitted (thick curve).

It appears that chartists are responsible for the herding behavior (wave-form of the price action) and that nearly twice as many fundamentalists, submitting trades more than twice as big as chartists, are needed to dampen the effect of chartists. Fundamentalists use a roughly 60-minute moving average, have a ±2.5% response envelope, and might be considered patient in waiting a half hour after a threshold cross before awakening. The chartists seem to be conservative, waiting an average of 4 minutes before assessing a trend and acting only when that trend is relatively steep – that is, an average of 0.75% growth per time step (-0.75% in the case of a down-trend). Chartists having a maximum number of trades around 65 may or may not make a difference; the number of chartists who reached their maximum was not recorded. It might be the case that because none of the beta distributions were very close to the edges of their possible ranges, that reasonable bounds were chosen for the variables. The beta distributions also seem to be quite "tight," and we are tempted to conclude that narrow ranges tend to be most appropriate, but we are reminded that it is only when both α and β are very small that a beta distribution is not tight. It must be noted that while this best candidate appears to match that target exceptionally well, consider that some of the match is a result of random outcomes resulting from the noise trader and from parameters values being drawn randomly from beta distributions. Seeded with different random numbers, model simulations with the same parameters produce variety although they all exhibit similar qualitative features.

5. GA vs IEC

One of the first questions one may ask about the example described in the previous sections is, "why can't a straight genetic algorithm come up with similar results in a short amount of time?" The answer is not trivial. In fact, it is worth testing. In order to apply a GA with an automated rather than user-defined fitness function, one has to define one such fitness function. The easiest and most natural one here is simply the inverse of a fitting error function $E=\left(N^{-1}\sum_{t=0}^{N}\left(p_t-a_t\right)^2\right)^{1/2}$, where p_t and a_t represent the model-predicted and actual price at time t, respectively. We used a simple GA, where the probability to be selected for each mutation or crossover operation is proportional to $1/\left(E+0.0001\right)$. Figure 4 shows the fittest phenotype the GA found after 50 generations (top) together with a good match found with IEC. The error value for the fittest GA-discovered phenotype is 31.68, which is lower than the value obtained with IEC, 38.85. However, the phenotype discovered by the GA does not satisfy the requirements of the experiment, namely it does not match the amplitude and shape of the first wave and dampens so fast as to almost not exhibit the second wave. This result is not entirely surprising since these qualitative features were not built into the error function E and that is why GA could not "see" them but a human could. We repeated the experiment with GA more than 10 times with different initial conditions but could not get a good match to the true price curve. E is not a good measure of performance in this experiment.

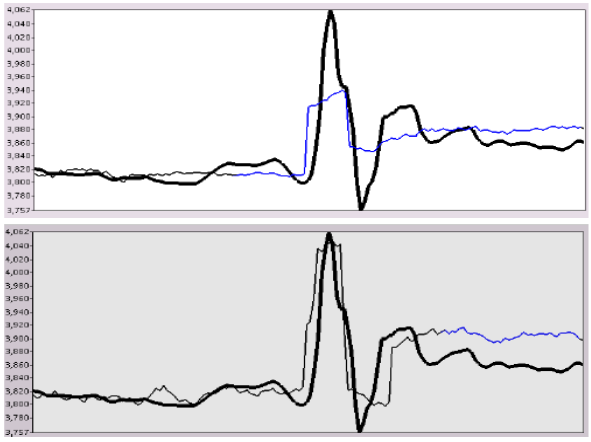

Figure 4. Best solution with data to be fitted (thick curve). Top: GA best solution. Bottom: IE best solution.

6. MODULARITY

Another interesting aspect of IEC is the fact that users using IEC often find parts of a solution they like and end up selecting solutions that have such interesting parts. Because they are not given the opportunity to select only part of a solution, they have to select an entire solution. We tested the effectiveness of a modularity-based selection strategy. However, in the present example, the mapping from genotype to phenotype is "indirect", in the sense that modules observed in the phenotype have no easy-to-identify counterparts at the genotype level. To introduce modularity in a way that is relevant to the user, we divided each

phenotype (basically a time series) into 5 temporal modules of equal size. The user can select any number of modules from as many phenotypes as he wants. Each phenotype of the next generation is then evaluated by computing an error function E_m with respect to each module m as:

$$E_m = \left(L^{-1}\sum_{i=0}^{L}\left(p_{o+i} - m_i\right)^2\right)^{1/2},$$

where L is the length of module m, o is the offset that sets the location of the chosen module on the phenotype, p_{o+i} is the model-predicted price at time $o+i$, and m_i is the model-predicted price that the user sets through the module he picks. This error function computes how much this phenotype conforms to this particular module. We discard a phenotype if E_m is above a certain threshold T for all the modules at a certain location. The user can adjust T as another way of interacting with the system. While a too small T would make the system discard all phenotypes generated through evolutionary operators, a big threshold would not act as a good filter and would create phenotypes that do not match user's module selections. We found that T=75 worked well for this example. We tested the system with 10 random initial conditions. Figure 5 shows the result of this experiment in terms of the number of cases which could find a good match after a given number of generations. It is clear that MIE outperformed IE in this example. IE found a good match at generations 4, 9 (twice), 10, 11, 13, 14, 15, 18, and 49 (15 +/- 12 generations) while MIE found a good match at generations 3 (3 times), 7 (3 times), 9, 10, 15, and 16 (8 +/- 4.4 generations).

Note that even though we used an analytical formula to filter phenotypes according to module settings, we always assumed that the true price history is unknown to the system and we have neither computed nor used the error function with respect to the true prices.

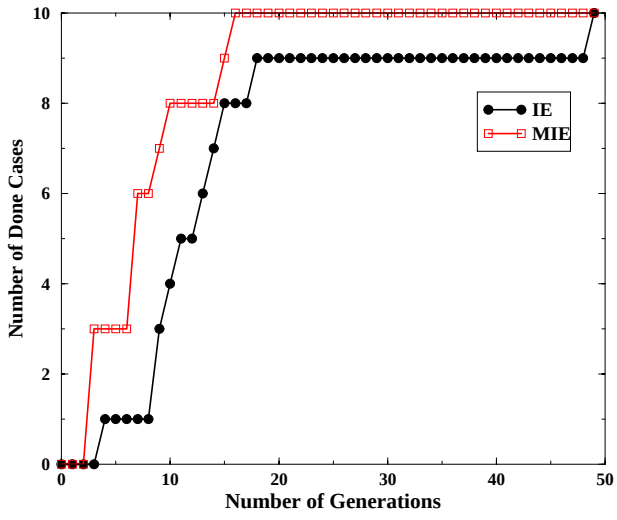

Figure 5. Comparison of IEC and modular IEC.

7. LEARNING USER PREFERENCES

In this section we use machine learning techniques to learn the underlying characteristics of human evaluation in IEC. The user evaluates the phenotypes for a number of generations (e.g., until he/she is tired) while a supervised neural network is trained. Then the user lets the neural network evaluate the phenotypes on his behalf and the evolutionary algorithm evolves solutions based on the fitness assigned by the neural network. The supervised and predictive phases can be repeated until the user is satisfied with a solution.

Though the idea sounds simple, there are some challenges inherent in the approach:

- The selection and implementation details of the machine learning technique depend on the particular problem.
- Neural networks usually require large training sets. Since human fatigue is the main motivation for the approach, we should typically expect less than 100 training patterns. That is not adequate for most networks.
- In IEC, the user evaluates a given number of phenotypes and selects some of them based on his/her comparison with other phenotypes presented *within the same generation*. The same phenotype he selects as the best of the first generation may be the worst of generation 10. To deal with this relative evaluation problem, we let the user rate each phenotype on a scale from 0 to 10, 10 being the best solution. We also assume that the user is more or less consistent on his ratings, which we know is not always true as humans may change their evaluation criteria over time.
- We assume that the user does not reach the optimal solution during the initial supervised phase, otherwise he does not need to proceed with the neural network. Since the optimal solution is not in the training set, it is unlikely that the neural network will recognize it in the predictive phase.

In our implementation, applied to the problem described in the previous sections, each generation consists of 12 genotypes with 2 elites surviving to the next generation. Five genotypes are created through mutation and 5 genotypes through crossover. The probability of a genotype to be selected for each mutation or crossover operation is proportional to R^2, where R is the user rating between 0 and 10. A simplified version of the Radial Basis Function network [12,13] is used. The output of the network is given by

$$y(x) = \sum_{j=0}^{K}\omega_k g_k(x),$$

where ω_k's are the elements of a weight vector and x is the K-dimensional input vector. We defined the radial basis functions as Gaussians:

$$g_k(x) = \exp\left[-\left(x_k - \mu_k\right)^2 / 2\sigma_k^2\right],$$

where μ_k and σ_k are the center and variance of the Gaussian. In this implementation we use an iterative learning rule to find the centers μ_k that minimize the difference between the user ratings and the network predictions. Since we know that each dimension is equally important for the user we fixed the weights ω_k at 10/K and σ_k at 0.5.

In order to measure the performance of the system it is useful to define an automatic error function $E=\left(N^{-1}\sum_{t=0}^{N}\left(p_t-a_t\right)^2\right)^{1/2}$, where p_t and a_t represent the model-predicted and actual price at time t, respectively. Figure 6 shows user evaluations and automatic error values for a large set of phenotypes. It is clear that the user rating is not a simple function of E, the user assigns the same rating for phenotypes with very different E values.

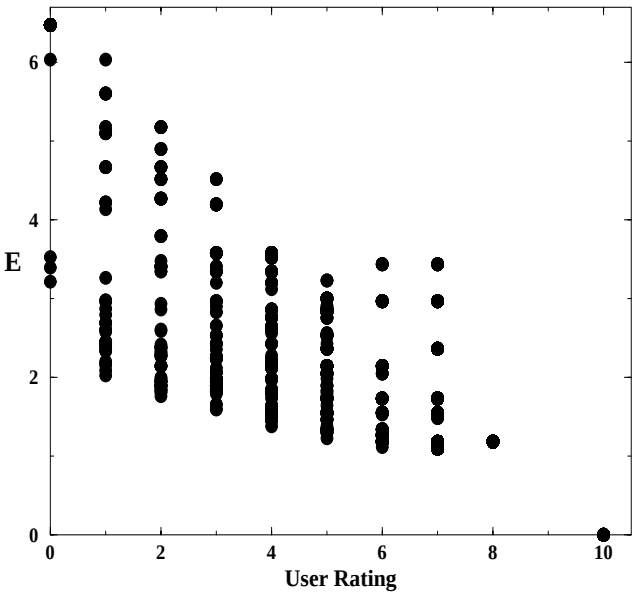

Figure 6. *E* vs user rating..

The system was tested with 30 random initial conditions. In each run, the user evaluates the phenotypes for 5 generations. If the user reaches the optimal solution during this training phase, that run is discarded and the system reset. If the optimal solution is not found after 5 generations the user lets the neural network predict his/her evaluations and evolve the genotypes.

Figure 7 shows the average of best user rating and E as a function of the number of generations. The number of distinct training patterns ranged from 6 to 25 with a mean of 16.7. In order to produce Figure 7a, we had the user rate the phenotypes after the 5th generation, but we did not use those ratings neither to train the neural network nor to use in evolutionary operations. It is clear that the quality of the phenotypes continues to increase even after the neural network replaced the user.

It is interesting to analyze the generalization performance of the neural network. Figure 8 shows the neural network predictions as a function of the user ratings. The blue curve shows the histogram of the training patterns. The network is trained on input patterns with user ratings between 0 and 8, and it could successfully predict the optimal solution's score around 9.5.

8. CONCLUSION

We have shown with a simple financial markets example how IEC can be used to perform agent-model inversion. When applied to simple models and reasonably small parameter spaces that have human-interpretable visualizations, IEC as a technique nicely combines human expertise with evolutionary computation and may improve the speed and accuracy of search in the fields of model inversion and model design, particularly when goal evaluation is multi-variate, complex, or qualitative. Although a straight optimization algorithm such as a genetic algorithm (GA) with a least-mean squares objective function does find a solution, IEC enables the user to estimate models without having to define a problem-specific, *ad hoc* objective function. One useful extension to this work consists of using IEC to fit two or more aggregate-level datasets simultaneously, thereby increasing the plausibility of the discovered model. It was also shown that adding modularity to the user selection process greatly enhances the user's ability to find a good solution. A machine learning technique that learns the user's selection preferences also helps mitigate one of the biggest issues with IEC, user fatigue.

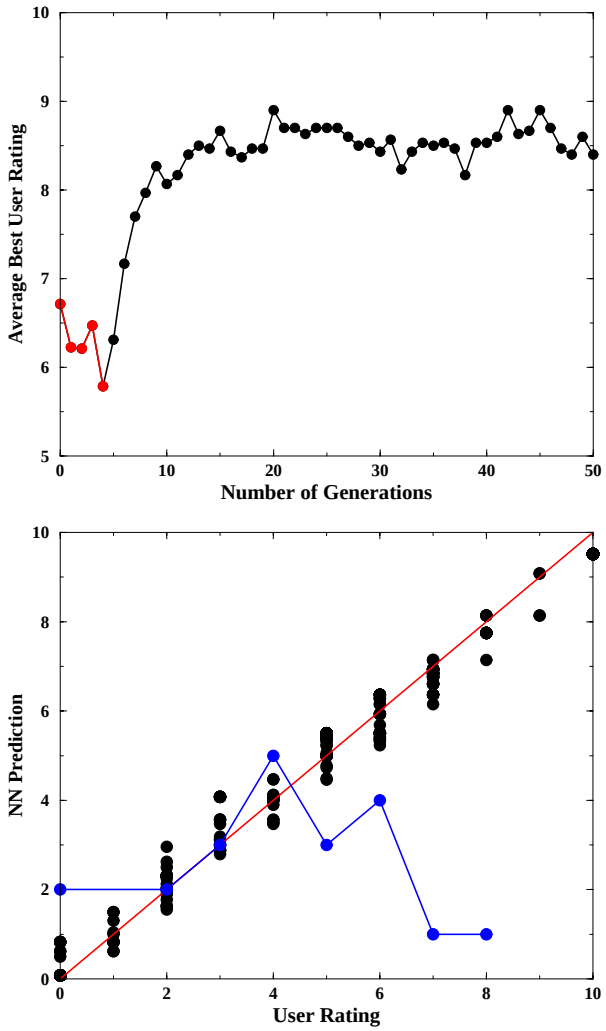

Figure 7. Top: Average best user rating as a function of generation number. Bottom: *E* as a function of generation number.

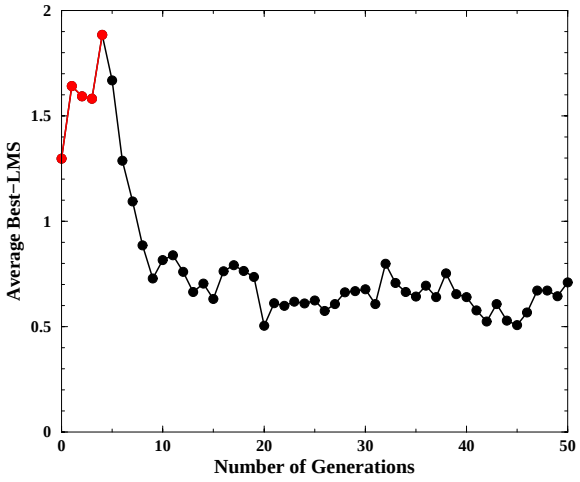

Figure 8. Neural net predicted rating vs user rating (black dots). The line shows a histogram of training patterns.

9. REFERENCES

[1] Arthur, W. B., Holland, J. H., LeBaron, B., Palmer, R. G. & Taylor, P. (1997) Asset pricing under endogenous expectations in an artificial stock market, in *The Economy as a Complex Evolving System II*, eds. Arthur, W. B., Durlauf, S. & Lane, D. (Santa Fe Institute Studies in the Sciences of Complexity, Proceedings Volume XXVII, Addison-Wesley, Reading, MA).

[2] Ball, P. (2002) Stock market shock explained. Physicists model recent trading frenzy. Nature Science Update, http://www.nature.com/nsu/020923/020923-18.html

[3] Banzhaf, W. (1997) Interactive evoution. In: Handbook of Evolutionary Computation (T. Baeck, D. Fogel, and Z. Michalewicz, eds), Ch. C2.10, pp. 1-5, Oxford University Press.

[4] Bonabeau, E. (2002) Agent-based modeling: methods and techniques for simulating human systems, *Proc. Nat. Acad. Sci. USA* **99**, 7280-7287

[5] Boschetti, F., Moresi, L. 2001. Interactive inversion in geosciences. *Geophysics* 66, 1226-1234.

[6] Caldarelli, G., Marsili, M. & Zhang, Y.-C. (1997) A prototype model of stock exchange. *Europhys. Lett*. **40**, 479-484

[7] Epstein J. M., Axtell R. L. (1996) *Growing artificial societies: social science from the bottom up* (MIT Press, Cambridge, MA).

[8] Dawkins. R. 1987. *The Blind Watchmaker*. W. W. Norton, New York.

[9] Farmer, J.D. (1999) Physicists attempt to scale the ivory towers of finance. *Computing in Science and Engineering* (IEEE), November-December 1999, 26-39.

[10] Forrest, S. 1993. Genetic algorithms: Principles of adaptation applied to computation. *Science* **261**: 872-878.

[11] Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Longman Publishing.

[12] Hassoun, M.H. (1995), Fundamentals of Artificial Neural Networks, The MIT Press, Cambridge, MA

[13] Haykin, S. (1999), Neural Networks: A Comprehensive Foundation, Second Edition, Macmillan, New York.

[14] LeBaron, B. (2001) A builder's guide to agent-based financial markets. *Quantitative Finance* **1**, 254-261

[15] Levy, M., Levy, H. and Solomon, S. (2000). Microscopic Simulation of Financial Markets: From Investor Behavior to Market Phenomena. (Academic Press, San Diego, CA).

[16] Lux, T. & Marchesi, M. (1999). Scaling and criticality in a stochastic multi-agent model of a financial market. *Nature* **397**, 498-500

[17] Mantegna, R. & Stanley, H. E. (1999). An Introduction to Econophysics, Cambridge University Press

[18] Muchnik, L. & Solomon, S. (2003). Statistical mechanics of conventional traders may lead to non-conventional market behavior. *Physica Scripta* **T106**, 41-47.

[19] Palmer, R. G., Arthur, W. B., Holland, J. H., Le Baron, B. & Tayler, P. (1994) Artificial economic life: a simple model of a stock market, *Physica D* **75**, 264-274.

[20] Sims, K. 1991. Artificial evolution for computer graphics. *Computer Graphics* **25**: 319-328.

[21] Sims, K. 1992. Interactive evolution of dynamical systems. Pages 171-178 in: *Towards a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life* (F. J. Varela & P. Bourgine, eds.), MIT Press, Cambridge, MA.

[22] Sims, K. 1993. Interactive evolution of equations for procedural models. *Vis. Comput*. **9**: 446-476.

[23] Takagi, H. 2001. Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. *Proc. IEEE* **89**: 1275-1296.

[24] Wijns, C., Boschetti, F. & Moresi, L. 2003. Inversion in geology by interactive evolutionary computation. *Journal of Structural Geology,* in print.