Overview

- Introduction
- Biologically-inspired Algorithms
- A Tour of Some Financial Applications ...
  - Optimisation
  - Model-induction
  - Agent-based Modelling

Research Group

- Natural Computing Research and Applications Group
  - Interdisciplinary research group based in UCD's CASL Institute
  - Development of NC algorithms / theory
- Application areas
  - Finance, bioinformatics, architecture, engineering, sound synthesis, game AI etc.
- Staffing
  - Five faculty
  - 15 PhDs
  - Five Post docs

Natural Computing

Inspire Design of *
Natural Processes
- Natural processes as computational machines
- Understanding Natural Processes

- Working definition for this presentation …
  - "The development of computational algorithms using metaphorical inspiration from systems and phenomena that occur in the natural world"
- Many of the best known algorithms in have been derived from a biological or social inspiration
- What makes the biological metaphor interesting when designing artificial problem solvers?

Biologically-inspired Algorithms

- Biological organisms earn a living in ‘difficult’ environments
  - Typically "high-dimensional" and dynamic
- Mechanisms have arisen which assist the ‘survivability’/adaptability of populations of biological creatures in these environments
- These are potentially useful in helping inspire us when designing algorithms to attack interesting real-world problems in the finance (and other) domain(s)
Dynamic Environments

- Biological responses to dynamic environments include...
  - Populations rather than individuals
  - Multiple 'probes' (learning trials) of the environment
  - Multiple mechanisms for learning
- Mechanisms for maintaining a memory of good past solutions
- Mechanisms for generating diverse new individuals/capabilities
  - Fitness-based (de) selection
- Focus is on robust not optimisation (i.e. robustness)

Earning a Living ...

- Let's take a simple example of the problem of 'earning a living' in a dynamic environment where the future actions of other agents are unknown ...

Earning a Living ...

- At each time step in the program one of nine rules (in decreasing priority) is fired

  Rule 1:
  IF (distance(nearest_power_pill) \( \leq 5(3\times) \) AND (4 \( \leq \) distance(nearest_ghost) \( \leq 8 \)) AND (distance(ghost_nearest_to_the_nearest_power_pill) \( \leq 6(4\times) \)),
  THEN stop moving and ambush (enter the ambush state) at the corner or cross point near the nearest power pill waiting for a ghost to come closer, where distance(nearest_power_pill) is the distance from Ms. Pac-Man to the nearest power pill, distance(nearest_ghost) the distance from Ms. Pac-Man to the nearest ghost, and distance(ghost_nearest_to_the_nearest_power_pill) the distance from Ms. Pac-Man to the ghost nearest to the power pill nearest to Ms. Pac-Man, and the numbers with * in the parentheses are those for the second stage of the game.

- How can we find good 'rules' for surviving in this (or any other...) environment?
Evolutionary Computation

Canonical evolutionary algorithm

\[ x[t + 1] = r(v(x[x[t]])) \]

Initiate population
WHILE (Termination condition False)
Calculate fitness of each individual
Select parents
Create offspring
Update population
ENDWHILE

UCD Natural Computing Research & Applications Group (NCRA)

A Simple Optimisation Example

- Sometimes designing the genotype to phenotype mapping is simple
  - For example, suppose we want to design a genotype to encode three coefficients for a linear regression model of the form ...
  - The genotype could be a real-valued string, encoding the three model coefficients

\[ \beta_0, \beta_1, \beta_2 \]

Fitness (MSE)

<table>
<thead>
<tr>
<th></th>
<th>( \beta_0 )</th>
<th>( \beta_1 )</th>
<th>( \beta_2 )</th>
<th>Fitness (MSE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-3.1245</td>
<td>5.6219</td>
<td>11.3411</td>
<td>0.3245</td>
</tr>
<tr>
<td>2</td>
<td>-4.5612</td>
<td>-0.2317</td>
<td>6.1311</td>
<td>0.7436</td>
</tr>
<tr>
<td>3</td>
<td>2.3412</td>
<td>1.6432</td>
<td>2.7811</td>
<td>0.6718</td>
</tr>
<tr>
<td>( n )</td>
<td>-3.6245</td>
<td>4.8219</td>
<td>13.3411</td>
<td>0.3015</td>
</tr>
</tbody>
</table>

UCD Natural Computing Research & Applications Group (NCRA)

A Simple Optimisation Example

We generate a 'child' solution by applying a pseudo-crossover operation to the two parents
Crossover uses information from both parent (recombines their good information – here using a simple averaging process)

Fitness (MSE)

<table>
<thead>
<tr>
<th></th>
<th>( \beta_0 )</th>
<th>( \beta_1 )</th>
<th>( \beta_2 )</th>
<th>Fitness (MSE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-3.2346</td>
<td>5.2218</td>
<td>12.3911</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

UCD Natural Computing Research & Applications Group (NCRA)

A Simple Optimisation Example

Next, apply a mutation operator to the child 'solution' and determine its fitness
Mutation allows for the discovery of information not contained in either parent

Fitness (MSE)

<table>
<thead>
<tr>
<th></th>
<th>( \beta_0 )</th>
<th>( \beta_1 )</th>
<th>( \beta_2 )</th>
<th>Fitness (MSE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-3.2346</td>
<td>5.2218</td>
<td>12.3911</td>
<td>0.2918</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

UCD Natural Computing Research & Applications Group (NCRA)
A Simple Optimisation Example

<table>
<thead>
<tr>
<th>$\beta_0$</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
<th>Fitness</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crossover 1</td>
<td>-3.1245</td>
<td>5.6219</td>
<td>11.3411</td>
</tr>
<tr>
<td>Crossover 2</td>
<td>-3.6245</td>
<td>4.8219</td>
<td>13.3411</td>
</tr>
<tr>
<td>Child 1</td>
<td>-3.3745</td>
<td>5.2219</td>
<td>12.3411</td>
</tr>
<tr>
<td>Child n</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- Place is selected from the next generation of the population and the algorithm continues.
- Gratification, cross-over, the quality of members of the population improves and converges on the optimum values of $\beta_0$, $\beta_1$, $\beta_2$.

UCD Natural Computing Research & Applications Group (NCRA)

Crossover

- Crossover
  - Aims to use information from better parents....
  - Could implement ‘intermediate crossover’ (a simple average)

Parent 1

<table>
<thead>
<tr>
<th>$\beta_0$</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3.1245</td>
<td>5.6219</td>
<td>11.3411</td>
</tr>
</tbody>
</table>

Parent 2

<table>
<thead>
<tr>
<th>$\beta_0$</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3.6245</td>
<td>4.8219</td>
<td>13.3411</td>
</tr>
</tbody>
</table>

Child

<table>
<thead>
<tr>
<th>$\beta_0$</th>
<th>$\beta_1$</th>
<th>$\beta_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3.3745</td>
<td>5.2219</td>
<td>12.3411</td>
</tr>
</tbody>
</table>

- More generally, could use $P_1\cdot\text{max}(P_1, P_2)$, where $P_1$ and $P_2$ are the
  real-values in that locus of each parent and $\alpha$ is a scaling factor
  (perhaps randomly drawn from [-2, 2])
- Defines a hypercube based on the current location of the parents.

UCD Natural Computing Research & Applications Group (NCRA)

Mutation

- Mutation
  - Allows the uncovering of new information that was not present in either parent
  - Could add a random draw from $N(0, \alpha)$ to each element of each child solution
  - Hence, most mutations are small with occasional larger mutation steps
  - Value of $\alpha$ is user-defined (scaled as appropriate)

UCD Natural Computing Research & Applications Group (NCRA)

Model Calibration

- Often in finance, we may have a ‘theory’/model which explains returns/prices etc.
- Model may be complex/non-linear
- In order to apply the model, its parameters need to be determined or ‘calibrated’
  - Select model parameter values so that ‘model output’ matches actual market output
  - Calibrated model can then be used to (for example) price financial instruments.

UCD Natural Computing Research & Applications Group (NCRA)

Model Calibration

\[
F(S_t; \Delta S_t) = \text{Black Scholes Option Pricing Model} \\
F(S_t; \Delta S_t) = \text{Calibration (implied volatility)}
\]

\[
\beta_0 = \beta_1 = \beta_2 = a + \sigma \cdot (x_t - y_t) \\
\alpha = \frac{\beta_0}{\beta_1 + \beta_2}
\]

Calibration (implied volatility)

\[
\alpha = \frac{\beta_0}{\beta_1 + \beta_2}
\]

\[
C(S_t; K, \sigma) = \text{Black Scholes Option Pricing Model} \\
C(S_t; K, \sigma) = \text{Calibration (implied volatility)}
\]

UCD Natural Computing Research & Applications Group (NCRA)

Parameterising a Simple Rule

- How might you represent a simple technical trading rule of the following form as a string?
  - IF $x$ day MA of price $> y$ day MA of price
  - THEN Go Long ELSE Go Short

IF $x$ day MA of price $> y$ day MA of price

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y$</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>10</td>
<td>Short</td>
</tr>
</tbody>
</table>

IF $x$ day MA $> y$ day MA THEN Go Long

UCD Natural Computing Research & Applications Group (NCRA)
Back-testing the Quality of a Genotype …

Genotype

Trading Model

Historical Data

Fitness Feedback

A Slightly More Complex Example …

IF \[\text{Indicator}_{i}(t) (<,>) \text{value}_{j}\] THEN (Buy, Sell, Do nothing)

- The above are simple illustrative examples, much more complex, compound, trading rules, which would defy any attempt at discovery via enumerative methods, could also be generated using AND, NOT, OR etc. operators

Uncovering an Stock Selection Rule

- In a simple case, we may be trying to uncover a good subset from an array of plausible filter rules (possible rules depend on your investment style)
  - Each indicator could be coded as a 0 (no) or 1 (yes), with an evolutionary process being applied to uncover the best subset of filter rule components

Model Induction

- Can we ‘recover’ a suitable model (+ associated parameters) from the data?
  - ANNs (universal approximators … but readability?)
  - GP / GE
  - Likely to be useful when we have data but weak theory (perhaps some idea of the likely relevant variables but little idea how they might link together)
- Applications include:
  - Financial time-series forecasting, credit risk modeling, pricing model discovery, forecasting takeover targets, prediction of earnings, IPO underpricing, trading system etc. etc.

Genetic Programming

- An evolutionary model-induction methodology
- Idea dates from at least the 1950s, popularised by John Koza
  - In his 1992 book ‘Genetic Programming: on the programming of computers by means of natural selection’
- GP adopts an evolutionary metaphor
  - Generate a population of trial solutions, assess worth of each, select, crossover, mutate, replace
Genetic Programming

- Individual is or represents a program

```c
#include<stdio.h>
#include<stdlib.h>
...
int main(int argc, char* argv){
float x=0.0, y=0.0, z=0.0;
x=atof(argv[1]); y=atof(argv[1]); z=atof(argv[1]);
z1 = 2.0*sin(y) + exp(z);
printf("The answer is: %f\n",z1);
return (0);
}
```

Of course, a 'program' (or the equivalent tree representation) can be viewed as a list of rules – and many financial problems can be viewed as a search for a ‘good’ list of rules – lending decisions, investment decisions, ...

UCD Natural Computing Research & Applications Group (NCRA)

### Diversity Generation

- Identifying appropriate Representations
- Fitness Landscapes & Problem Difficulty
- Static vs. Dynamic Problems
- The Influence of Biology
- Open-ended Evolution
- Generalization
- Benchmarks
- Modularity
- Complexity of GP
- Misc...

UCD Natural Computing Research & Applications Group (NCRA)
Grammatical Evolution

- Grammar-based GP ... model specification / choice of explanatory variables not assumed ex-ante
- Genotype → phenotype mapping
  - Each program is generated from a variable length linear binary (or integer) string
  - Key item is that the evolutionary process is effectively applied to the 'production rules' - i.e., the developmental rules governing the production of the phenotype, and not directly to the phenotype itself

UCD Natural Computing Research & Applications Group (NCRA)

Grammatical Evolution

Binary String

Integer String

Search Algorithm

Production Rules / Grammar

Program

Output (Fitness)

UCD Natural Computing Research & Applications Group (NCRA)

BNF Grammar

T = [cos, sin, tan, log, x, y, a, b, c, d] x = (expr)
S = (expr, expr, expr, const)
expr =
  (expr) => (expr) (expr) (expr)
  (expr) => cos(expr) (expr) (expr)
  (expr) => sin(expr) (expr) (expr)
  (expr) => tan(expr) (expr)
  (expr) => log(expr) (expr) (expr)

 expr = a + b - c
 expr = tan(x) + sin(y)

UCD Natural Computing Research & Applications Group (NCRA)

Example Mapping

<table>
<thead>
<tr>
<th>Expression</th>
<th>Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>sin(x) + cos(y)</td>
<td>2220131</td>
</tr>
<tr>
<td>tan(x) * sin(y)</td>
<td>877895</td>
</tr>
<tr>
<td>log(x) - cos(y)</td>
<td>987987</td>
</tr>
<tr>
<td>(x + y) / z</td>
<td>897897</td>
</tr>
</tbody>
</table>

UCD Natural Computing Research & Applications Group (NCRA)

Sample Grammars

- Canonical GA and GA search engine

UCD Natural Computing Research & Applications Group (NCRA)
Want to learn more?

Released 1st July....

http://ncra.ucd.ie/geva/

Financial Applications ...

- Recent Financial Applications of GE include:
  - Asset selection
  - Options price modeling
  - Credit risk assessment
  - Algorithmic trading
  - Portfolio rebalancing
  - .......

Bond Rating Example

- Companies can be financed by share capital or debt capital
- Debt capital
  - Bond issue
- If bonds are to be publicly traded, they require (in US) a credit rating from a ‘recognized’ rating agency (S&P, Moody’s, Fitch’s) ...
- S&P covers 99.2% of traded debt in the US
- Credit rating represents an agency’s opinion of the creditworthiness of a borrower

Background

Initial Rating | Default (%) |
--- | --- |
AAA | 0.52 |
AA | 1.31 |
A | 2.32 |
BBB | 6.44 |
BB | 19.52 |
B | 35.76 |
CCC | 54.38 |

Results

Score

Determinantal Grade

Initial Rating

Bankruptcy Example

File: bank_data.txt

UCD Natural Computing Research & Applications Group (NCRA)

Data

600 US firms (420 training: 180 out of sample)

S&P bond ratings + data drawn from their financial statements

Grammar

<lc> ::= if( <expr> <relop> <expr> ) class='Junk';
else class='Investment Grade';
<expr> ::= ( <expr> ) + ( <expr> ) | <coeff> * <var>
<var> ::= var1[index] | ... | var8[index]
<coeff> ::= ( <coeff> ) <op> ( <coeff> ) | <float>
<op> ::= + | - | *
<float> ::= 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | -1 | .1
<relop> ::= <=

UCD Natural Computing Research & Applications Group (NCRA)
Other Applications / Other Methods


Model Induction

Prediction of Going Concern Audit Qualification

Data:
- Liquidity, Debt, Profitability, Activity / Efficiency, Cash Generation
- Rate of change of above
- Loss in current year, Loss in past three years, Big-six auditor

<table>
<thead>
<tr>
<th>Data</th>
<th>MLP Train</th>
<th>Out of sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loss</td>
<td>72.79%</td>
<td>70.61%</td>
</tr>
<tr>
<td>Loss</td>
<td>68.87%</td>
<td>63.79%</td>
</tr>
<tr>
<td>Loss</td>
<td>65.96%</td>
<td>57.02%</td>
</tr>
</tbody>
</table>


The Trading Environment

- Growth of Algorithmic Trading
  - Can encompass any aspect of the trading process
  - Trade execution only, or...
  - Entire process of opportunity identification and trading e.g. arbitrage trading

- Fragmentation of Liquidity

Algorithmic Trading

- High-frequency trading (HFT)
  - Algorithm may place 000s of orders per second (with up to 90% being cancelled)

Algorithmic Gaming ....
Algorithmic Trading

- High-frequency trading (HFT)
  - Concurrent need for low latency both in terms of algorithmic decision-making (opportunity identification and execution), data link speed (co-location with exchange matching engine is now common ... NYSE moved to new data centre in NJ at end-AUG), and speed of processing of exchange platform to incoming orders.
  - At these speeds, even internal machine processes can become bottlenecks e.g. system processes which clean house and reassign memory can slow down market response ... (hence, HF Traders may look to ‘control’ when these processes occur ...!)

UCD Natural Computing Research & Applications Group (NCRA)

---

Exchange Time to execute a trade (milliseconds)

<table>
<thead>
<tr>
<th>Exchange</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wall Street</td>
<td>0.677</td>
</tr>
<tr>
<td>London / Sydney</td>
<td>3</td>
</tr>
<tr>
<td>Tokyo / Bombay</td>
<td>6</td>
</tr>
<tr>
<td>Hong Kong</td>
<td>9</td>
</tr>
<tr>
<td>Brazil</td>
<td>20</td>
</tr>
<tr>
<td>Singapore</td>
<td>36</td>
</tr>
</tbody>
</table>

Source: Nicola Vladimiro (2010)

---

Algorithmic Trading

- Algorithmic gaming
  - ‘Quote stuffing’
    - Send heavy order traffic to exchange which are almost immediately cancelled in order to slow down data feeds from the exchange e.g. a large CQS quote is 102 bytes. Suppose an algorithm generates 1,000,000 CQS quotes per second ... message traffic from the exchange
    - As a T1 line has a capacity of 1.544 MBps, (2.048 MBps for OC-3), you must send a CQS quote every 1.544 MBps ...
    - This opens up microsecond arbitrage opportunities between different venues as the ‘quote stuffer’ can differentiate the true liquidity from the fake liquidity generated by the algorithm and create a potential profit opportunity

UCD Natural Computing Research & Applications Group (NCRA)

---

Algorithmic Trading

Flash Crash 6 May, 2010 - DJIA drops 1000 points in 20 minutes mid session:


---

Algorithmic Trading

Estimated that this cost retail investors >$2bn as stop-loss orders were hit

UCD Natural Computing Research & Applications Group (NCRA)

---

Algorithmic Trading

The Bandsaw ..... Unusual trading activity on BATs (6 May 2010), (AGG – ETF based on US investment grade bonds)

Source: [http://www.nanex.net/20100506/FlashCrashAnalysis_Part4-1.html](http://www.nanex.net/20100506/FlashCrashAnalysis_Part4-1.html)

UCD Natural Computing Research & Applications Group (NCRA)
Agent-based Modelling of Markets (ASM)

- Actual market history is a single-sample path
  - Observation does not allow 'what-if' analyses
- ABM adopts a 'bottom up' approach to 'simulate' financial markets (markets 'in silico')
  - Autonomous agents whose interactions produce complex, emergent, system dynamics
  - Can embed heterogeneous agents with differing risk attitudes, with differing expectations to future outcomes, and with learning/adaptive capabilities
  - How do agents learn? (application of NC … GE … NN … etc.)
  - The 'Red Queen'

Agent-based Modelling of Markets (ASM)

- Plausibility testing of ABM / Model validation (?)
  - Can we replicate observed real world characteristics of markets?
  - What are the critical assumptions?
- Applications of ABM
  - Attempt to explain market behaviour (theory building)
  - Provide insights for policy makers and regulators
  - Provide test-bed for simulation of trading strategies / execution strategies

Agent-based Modelling of Markets (ASM)

- How can investors buy/sell large orders efficiently?
  - The traders dilemma ....
- We need to balance Market Impact (embeds cost of 'liquidity demand' and 'information leakage') and Opportunity cost

Agent-based Modelling of Markets (ASM)

- Trade execution requires the 'design' of an appropriate trading strategy
  - Online scheduling (number of sub-orders, their size, when submitted to market)
  - Order aggressiveness (what 'style' to adopt?)
  - Split of order across markets
- A practical problem then is how do we assess the utility of a strategy without actually implementing it?
- Scope to 'test' in an ASM

ASM as a Test Bed

- How can investors buy/sell large orders efficiently?
- The traders dilemma ....
  - Trade quickly
  - Trade slowly
- We need to balance Market Impact (embeds cost of 'liquidity demand' and 'information leakage') and Opportunity cost

Trade Execution Strategy

- Trade execution requires the 'design' of an appropriate trading strategy
  - Online scheduling (number of sub-orders, their size, when submitted to market)
  - Order aggressiveness (what 'style' to adopt?)
  - Split of order across markets
- A practical problem then is how do we assess the utility of a strategy without actually implementing it?
- Scope to 'test' in an ASM
Conference proceedings are published as part of Springer's LNCS series on foraging algorithms, artificial immune systems, hybrid systems and agent-based systems, evolutionary strategies, as well as other related natural computing methodologies such as particle swarm, systems. Other applications of interest include artificial stock markets.

Applications of interest include (but are not limited to) forecasting financial time series, portfolio selection, and management, estimating econometric parameters, pricing options, and developing risk management strategies. EvoFIN 2011 focuses primarily on the use of EC and related Natural Computing techniques in Computational Finance and Economics.

References

CALL FOR PAPERS

EvoFIN 2011
5th European on Evolutary and Natural Computation in Finance and Economics
27-29 April 2011
Torino, Italy

EvoFIN 2011 focuses primarily on the use of EC and related Natural Computing techniques in Computational Finance and Economics.

Applications of interest include (but are not limited to) forecasting financial time series, portfolio selection, estimating econometric parameters, pricing options, and developing risk management systems. Other applications of interest include artificial stock markets.

Submissions should include EC methods such as genetic programming, genetic algorithms, and evolutionary strategies, as well as other natural computing methodologies such as particle swarm, artificial immune systems, hybrid systems, and agent-based systems.

Conference proceedings are published as part of Springer's LNCS series.

Submission deadline: 22nd November

http://evofin.dei.u.ac.pt/