Learning

1. Simon on Learning
2. Artificial Neural Nets
3. Genetic Algorithms and Genetic Programming
4. Models from Psychology: Reinforcement Learning, etc.

Herbert Simon on Learning

“Any change in a system that allows it to perform better the second time on repetition of the same task or on another task drawn from the same population” — Simon (1983)

“Learning is any change in a system that produces a more or less permanent change in its capacity for adapting to its environment. ” — Simon (1996)

“Armchair speculation about expectations, rational or other, is not a satisfactory substitute for factual knowledge as to how human beings go about anticipating the future, what factors they take into account, and how these factors, rather than others, come within the range of their attention.” — Simon (1982)

Learning

Previous (e.g. Life, segregation) the agents are unchanging,

Now: change occurs in the agent's parameters (simplest) or in the form of the agent model (more complex).

Several sorts of models of learning:

1. **Artificial Neural Nets (ANNs): from machine learning, from biological simplifications of the brain's operation,**
2. **Evolutionary models, such as Genetic Algorithms (GAs) and Genetic Programming (GP), from natural evolution, and**
3. **Models from psychology experiments, such as Reinforcement Learning (RL).**

Artificial Neural Nets (ANNs)

An anecdote.

Instead of the ~100 million neurons in our brain, we use up to 50 artificial neurons, in three or more layers: Input, Hidden, and Output.

When presented with a stimulus, an ANN *learns* to output an appropriate signal.

Every unit (AN) in any layer is connected to every unit in the adjoining layer(s).

Every connection has a numerical weight.

One layer of units is the Input layer (by convention, on the left).

One layer of units is the Output layer (right, opposite).

In the middle are the Hidden layer or layers.

The crucial Hidden Layer ...

The Input layer gets stimulus from the environment; the Hidden layers take the signals received from the Input layer, process them, and pass them to the Output layer.

The strength of the Input layer's units are set by the environmental stimulus; the decoded strengths of the output layer is the ANN's response.

Activation of the non-Input-layer units depends on:

- **the strength of the inputs to the unit,**
- **the weights of each of its input connections,**
- **a mathematical function of these weights and inputs.**

Rescale the weighted sum of inputs to unity by a non-linear *activation* function bounded between 0 and 1: usually a sigmoid function ($y = \frac{1}{1+e^{-x}}$).

So ...

Need to encode any stimulus into numbers for the Input layer.

Need to train the ANN to give the correct output for a given input, usually by *back-propagation* of error: alter the weights to reduce the error (the difference between the output with random weights and the correct output response).

The amount of error adjustment of the weights depends on:

- **the derivative of the activation function,**
- **the size of the error,**
- **the ANN's exogenous learning rate, and**
- **a momentum proportional to all previous error adjustments.**

Applications of ANNs

ANNs used to:

- **recognise handwritten digits**
- **recognise human speech**
- **select good credit risks for bank loans**
- **track stock-market trends**
- **recognise objects in video images**
- **explore the development of a shared lexicon (= a language)**
- **dock Axelrod's (1987) evolution of cooperation with the IPD. (Marks & Schnabl 1999)**

Designing ANNs

More an art than a science.

“Good”: learn efficiently without needing too large a training set.

Design choices:

1. **encoding of the environmental stimuli**
2. **number of Hidden layers**
3. **number of units in each layer**
4. **the specific activation (or squashing) function (usually sigmoid)**
5. **the sizes of the learning rate and momentum constants (use the package defaults)**
6. **how the error is calculated.**

1. Encoding of Stimuli

Stimuli can be:

1. **Continuous or large integer (a measure or count)**
   Scaled $\in$ [0,1] and directly input, or categorised in bands.
2. **Categorical (e.g. male or female)**
   Assign one unit per category: 0 or 1 input.
3. **Qualitative (e.g. blue, heavy, with a sweet taste)**
   Code in binary, with a unit per binary position.

2. Number of Hidden Layers

Depends on the complexity of relationship between stimuli and responses.

Most phenomena only require a single Hidden layer (Masters 1992).

3. Number of Units

Input and Output layers:

Depends on the encoding: as many Input units as input categories.

e.g. An IPD with one period's memory?

with two week's memory?

Numbers of Units in the Hidden layer(s):

ANNs have a degree of generalisation: they can recognise a variant never seen before.

But we need to retain some specificity (to avoid misinterpreting new inputs).

Permit a degree of generalisation, but not too much.

As the number of Hidden layer units increases, the accuracy of input recognition increases, but generalisation ability falls — when the number of Hidden units = the number of distinct input examples, then there is 100% recognition, but no generalisation.

4. Measuring Recognition Error

Difference, or root square error.

But need to avoid local optima (local hills), by repeating the training on a new random set of initial connection weights.

How are ANNs Agents?

What are ANNs?

- **Hornik et al. (1989) have shown that ANNs are "universal approximators:"**
- **Can be used as agents when used in parallel, but don't require other ANNs.**
- **Used singly, don't lead to emergence, but can learn.**
- **Could use a population of ANNS, playing against each other, as agents.**

See Beltratti et al. (1996) for early applications of ANNs to financial and economics models.

Modelling Learning in ACE Models

Two sorts of (deductive) learning have dominated ACE models: GA (genetic algorithms) & RL (reinforcement learning)

- **GA, with implicit learning as the population "learns" from generation to generation: either**
  - **— a population of players (the single-population GA model), or**
  - **— a population of routines, ideas, heuristics, with each player modeled as a population (the multi-population GA model).**
- **RL, where the probability of choosing an action that was effective last round increases.**
- **Anticipatory (inductive), Belief-Based Learning — the future?**

Evolutionary Computation

Based on evolution with natural selection.

Fitter individuals have more offspring to pass their genes to;
less fit individuals have fewer offspring.

Genes occur in chromosomes; each gene codes for one (or more) functions.

The *genotype* = the structure of the individual's chromosomes.

The *phenotype* = the expressed characteristics (behaviours) of the individual, coded by the genotype.

Many phenotypes emerge from the individual genotypes: difficult to predict.

Evolution ...

1. **Populations evolve, not individuals.**
2. **Evolutionary change requires diversity at the genotype level in the population.**
   Clones are identical, and so are their offspring.
3. **While a species changes and adapts to its environment, the environment itself might change, because of the species' actions (example?) or because of other species' actions.**
4. **Acquired skills die with the parent: only inherent characteristics are passed on.**
   But: For *Homo sapiens*, language means culture, which can be passed on through deliberate learning.

Wollemia nobilis

Holland's (1975) Genetic Algorithm mimics natural evolution.

1. A population of "individuals," each having a fitness, which is measured.
2. The fittest individuals are chosen to breed a new population of offspring, inheriting fitter traits and genotypes.
3. Return to 1.

In breeding, the GA uses the processes of:

- *Selection* of parents to breed new offspring,
- *Crossover* of parents' chromosomes to pass on a mixture of their two genotypes, and
- Random *mutation* of some genes.

Or: selection, exploitation or imitation (of fit phenotypes), and exploration (of the genotype space) which reduces the risk of local optima.

Four GA Design Choices:

1. **Measures of Fitness.**
   Depends on what's being modelled.
   e.g. utility, wealth, survival time, profit,
   The average fitness of successive populations is always monotonically increasing: local optima.

2. **Selection mechanisms.**
   In choosing parents, need to retain some diversity in genotypes: don't only choose the fittest individuals to mate.

   Tournament selection: choose pairs of individuals at random; take the fitter of the pair as a parent.
   Typically, retain 40% of new population from the old, to protect better traits (embodied in the genotypes).

3. **Genetic Operators: crossover & mutation.**
   Single-point crossover: take two chromosomes (one from each parent), cut at the same randomly chosen position, swap the cuts to create two offspring (use at least one).
   Possible to have more than one cut point, but not important.

   Crossover preserves traits (from combinations of adjoining genes on the chromosome).

   Mutation: probability that a gene changes.
   Creates novelty at the genome level: diversity.

4. **Population size.**
   Should considerable exceed the number of genes in each chromosome.
   If the population is too small, then increased risk of convergence to a local optimum.
   But 20 to 50 has been used successfully.

Docking of a GA Model

In G&T 2nd edition, they dock Axelrod's (1987) model of the IPD:

Axelrod (1987) wrote in Pascal VS,

Marks (1988) wrote in C,

G&T (2005, pp. 239–247) write in NetLogo !

Szpiro (1997) used a GA to demonstrate the emergence of risk aversion.

GA Developments

Coding of the Genotype:
Nature uses the four nucleotide amino acids.
Holland's classical GA uses bits in a binary string.

But real numbers are possible: easier to deal with many real-valued problems.
Mutation: added a randomly chosen small number ($N(0, \sigma)$) to a small proportion of genes.

Or: use programs as genes $\rightarrow$ Genetic Programming (Koza 1992).
Need to ensure that crossover and mutation preserve a syntactically correct program, even if it performs very poorly.

Classifier Systems **(Holland et al. 1986): Production Sets which can alter their rules by learning from feedback after it has acted.**

Best Individual or Whole Population?

GA largely used as an optimiser: asking what is the best value (of fitness) in the population?

But focussing on the value of the best individual throws away the population's emerging characteristics as a population.

It ignores the aggregate level of emerging phenomena.

Axelrod (1987) not only sought the best-performing strategy in the IPD,

but also asked questions at the aggregate level of the population, such as its stability against invasion by a different strategy: e.g. Tit for Tat's stability against Always Defect. (See Marks 1989.)

How many GA populations?

Vriend (2000): with a GA and a single population, distinguish *social learning* from *individual learning*.

Social learning occurs at the genotypic level: sexual reproduction means that parents can communicate (share information) with their offspring via crossover:

$\therefore$ over generations, fitter genes or traits can spread through a single population covertly, by inheritance of genetic material.

How many parents, grandparents, great grandparents, ... do you have?

Multiple-population GA learning

With one population per player: Individual learning occurs only through arm's-length competition, and the selection of fitter individuals as future parents, not through inheritance of genetic material. (Illegal communication?)

When all members of a population are identical, then genetic inheritance is not a problem, since the aim is in general only to seek the fittest individual.

GAs with Co-evolution: Many Populations

When the environment in which the GA operates changes, and when such change is due to the behaviour of the species' competitors — co-evolution — then sharing of genetic material blurs the distinction between species.

Example: If the GA is being used to explore the behaviour of sellers in an oligopolistic market, genetic sharing can only model sub-rosa communication across brands.

This is illegal under most antitrust regimes, and therefore in general should not occur in the model, lest the results rely on it.

... so: How many populations?

The answer to the question: how many populations? is then: as many as there are distinct players, or distinct species coevolving.

Example: When each seller in an oligopoly has distinct costs, faces distinct demand, perhaps with a distinct actions set, then it should be modeled using a distinct population.

Perhaps because each GA has an internal population of individuals, a tendency to think of the GA as modeling heterogenous players.

But a single population assumes homogeneity.

Individuals in the GA

Each string in a GA population could be:

- **an individual brand (say), which I have argued above is unrealistic in general, or**
- **one possible decision, of a population that the agent could make — makes sense with a population per distinct player.**

So each new generation could be:

- **new individual decision makers (brands), or**
- **new ideas or heuristics belonging to long-lived players.**

Disputes about GAs in Economics:

Chattoe (1998) argues (correctly) that there has been confusion over the role of the GA:

- **an instrument to search a rugged solution space, or**
- **a model of firms' decision making and individual behaviour.**

Dawid (1999) argues that the GA is good at modelling the learning of populations of agents.

Curzon Price (1997): the GA provides a stream of hypothetical actions or strategies, which may or may not be used.

Duffy (2006) concludes that empirical evidence exists that GAs are reasonable "models of adaptive learning by populations of heterogeneous agents."

Marks & Schnabl (1999) compare a GA and an ANN playing an IPD

The logical structure of this ANN is a kind of dual to the normal ANN: normally the net gets inputs from the environment of data and forecasts future behaviour, here it makes the data by creating behaviour of the actual move (i.e. Cooperate or Defect). The data input of the ANN then is the history of one's own and one's opponent's moves.

- **The ANN did not really transform itself to a structure to best play the IPD.**
- **The ANN had to approximate 0 and 1 using real-valued functions.**

$\therefore$ **The ANN not as close to the IPD as is the GA: ANN seen to be less stable in solution.**

- **But with real-valued real-world variables, ANN might do better than a GA.**
- **GAs sometimes used to search for better weights in an ANN.**

Explicit Agent-Based Learning

With populations in a GA, learning is implicit: it occurs at the population level, not at the individual level — it emerges.

Arthur (1991, 1993) was the first economist to model explicit agent learning, and to calibrate his models using data from human-subject experiments.

In his Reinforcement Learning (RL) model, how an agent chooses to act later is a function of the outcomes it experienced as a result of earlier choices — the Thorndike effect.

At first he calibrated individual learning, but with the artificial stock market (Arthur et al. 1997), he became interested in data at the aggregate level.

Arthur's RL Model, the earliest

His model: In round t, player i has a propensity $q_{ij}(t)$ to choose pure strategy j, and q_{ij} is updated:

$$q_{ij}(t+1) = q_{ij}(t) + (x - x_{\min}).$$

where x was the payoff for choosing strategy j previously, and

$x_{\min}$ is the lowest possible payoff.

$\therefore$ Propensity to choose a strategy is *reinforced* if it has provided higher payoffs in the past, and vice versa.

$p_{ij}(t) = \frac{q_{ij}(t)}{\sum_{k=1}^{N} q_{ik}(t)}$ is the probability that agent i plays strategy j in period t.

Roth and Erev's generalisation

Roth & Erev (1995), Erev & Roth (1998) generalised Arthur's RL model to get a better fit with experimental data from multi-player games.

Initial propensities $q_{ij}(1)$ are equal across all strategies.

$\Sigma_j q_{ij}(1) = S_i(1) = S(1)$, an initial propensity parameter, equal across all players and strategies.

The rate of learning is proportional to $S(1)$.

Again, $p_{ij}(t) = \frac{q_{ij}(t)}{\Sigma_{k=1}^{N} q_{ik}(t)}$ is the probability that agent i plays strategy j in period t.

The Roth-Erev Model ...

Player i updates his propensity to play strategy j according to the rule:

$$q_{ij}(t+1) = (1-\phi)q_{ij}(t) + E_k(j, R(x)),$$

where $E_k(j, R(x)) = (1-\epsilon)R(x)$ **if** $j = k$, **or**

$$= \frac{\epsilon}{N-1} R(x) \text{ otherwise.}$$

where $R(x) = x - x_{\min}$.

Three parameters:

- **initial-propensity parameter $S(1)$**
- **recency parameter ϕ: reduces the power of past experiences**
- **experimentation parameter ϵ**

When $\phi = \epsilon = 0$, Roth-Erev is Arthur.

Five Types of RL Models

Five types of RL models (Duffy 2006, Brenner 2006):

1. **the Arthur-Roth-Erev model above**
2. ***Q*-learning, which optimises long-term payoffs rather than immediate returns (Watkins & Dayan 1992)**
3. **multi-agent *Q* learning (Hu & Wellman 1998), and**
4. **Adaptive Play (Young 1998)**
5. **Another modification of RL: suppose that agents have certain "aspiration levels" in payoff terms that they are trying to achieve. This idea has a long history in economics dating back to Simon's (1955) notion of *satisficing.***

   Could use x_{asp} instead of x_{min} above.

Selten's Directed Learning

Ex-post rationality determines adaptive behaviour.

Requires an ordering over the set of possible actions.

Players probabilistically move towards actions that would have been profitable had they been chosen earlier; and never move to lower their payoffs.

Hailu & Schilizzi (2004): use a mixed (i.e. probabilistic) strategy

- **if won last auction, use $P(\frac{1}{2})$ the same and $P(\frac{1}{2})$ +10% as next bid**
- **if not win, use $P(\frac{1}{2})$ the same and $P(\frac{1}{2})$ –10% as next bid, both bounded.**

Anticipatory, Belief-Based Learning

RL and GA-based learning models are inductive: respond to past actions and payoffs.
No attempt to anticipate and reason back, deductively.

***Belief-based learning*: agents form beliefs about other players' likely actions, and so respond to their beliefs.**

Gjerstad & Dickhaut (1998): "heuristic belief learning": agents use heuristics to update their beliefs about others' actions (expressed as probabilities) — good convergence to competitive equilibrium and good fit with aggregate behaviour.

Timing of bids is crucial.

Learning: How to optimise (LHTO), or how to predict (LHOP)?

LHTO: GA searches for actions or strategies that are best, lead to highest fitness (profits etc.)

LHTP: use the GA strings to encode how prices will change from period to period.

Used to calibrate GA output with human-subject experimental data.

For us, seems predicting prices is easier than predicting how to respond to changing prices.

Perhaps this suggest how markets help us solve difficult problems (see the *EJ* June 2005 feature discussed in Class 2).

References

[1] Axelrod R., The evolution of strategies in the iterated Prisoner's Dilemma, in L. Davis (ed.), *Genetic Algorithms and Simulated Annealing,* London, Pittman, 1987.

[2] A. Beltratti, S. Margarita and P. Terna, *Neural Networks for Economic and Financial Modelling*, London: International Thomson Computer Press, 1996,

[3] Brenner T., 2006, Agent Learning Representation: Advice on Modelling Economic Learning, Ch 3 in the *Handbook of Computational Economics, Volume 2: Agent-Based Modeling*, edited by Leigh Tesfatsion and Kenneth L. Judd, (Amsterdam: Elsevier Science).

[4] Duffy J., 2006, Agent-Based Models and Human Subject Experiments, Ch 4 in the *Handbook of Computational Economics, Volume 2: Agent-Based Modeling*, ed. by L. Tesfatsion and K.L. Judd, (Amsterdam: Elsevier Science).

[5] A. Hailu & S. Schilizzi (2004), Are Auctions More Efficient Than Fixed Price Schemes When Bidders Learn? *Australian Journal of Management*, 29(2): 147–168.

[6] Holland J.H. (1975), *Adaptation in Natural and Artifical Systems,* Univ. of Michigan Press.

[7] Holland J.H. et al., *Induction: Processes of Inference, Learning and Discovery,* Bradford, Cambridge, Mass., 1986.

[8] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, *Neural Networks*, 2(5), 360–365, 1989.

[9] Koza, J.R. (1992). *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, Cambridge.

[10] Marks, R.E. (1988) Niche strategies: the Prisoner's Dilemma computer tournaments revisited, http://www.agsm.edu.au/~bobm/papers/niche.pdf

[11] Marks, R.E. (1989) Breeding optimal strategies: optimal behavior for oligopolists, *Proc. of the Third International Conference on Genetic Algorithms,* George Mason

Publishers, pp.198–207.

[12] Marks R.E., 2006, Market Design Using Agent-Based Models, Ch 12 in the *Handbook of Computational Economics, Volume 2: Agent-Based Modeling*, edited by Leigh Tesfatsion and Kenneth L. Judd, (Amsterdam: Elsevier Science).

[13] Marks R.E., and Schnabl H. (1999) Genetic Algorithms and Neural Networks: a comparison based on the Repeated Prisoner's Dilemma, Thomas Brenner (ed.), *Computational Techniques for Modelling Learning in Economics*, (Dordrecht: Kluwer Academic Publishers), pp. 197–219.

[14] T. Masters, *Practical Neural Network Recipes in C++*, San Diego: Academic Press, 1992.

[15] N.N. Schraudolf & J.J. Grefenstette, A User's Guide to GAucsd 1.4, Technical Report CS92-249, CSE Department, UC San Diego, 1992.

[16] Simon, H.A. (1955), A Behavioral Model of Rational Choice, *Quarterly Journal of Economics*, 69, 99–118.

[17] H. A. Simon. *Models of bounded rationality,* MIT Press, 1982.

[18] H. A. Simon (1983) Why should machines learn?, in *Machine Learning: An Artificial Intelligence Approach,* R. S. Michalski, J. G. Carbonell and T. M. Mitchell, editors 1983, Springer-Verlag. Berlin. pp. 25–37.

[19] H. A. Simon , *The Sciences of the Artificial,* The MIT Press, 2nd ed., 1996.

[20] Szpiro G., The emergence of risk aversion, *Complexity*, 2(4), 31–39, 1997.

[21] N. J. Vriend, An illustration of the essential difference between individual and social learning, and its consequences for computational analyses, *Journal of Economic Dynamics & Control,* 24: 1–19, 2000.