Explorations in Evolutionary Visualisation

Lionel Barnett

Centre for Computational Neuroscience and Robotics (CCNR)
School of Cognitive and Computing Sciences, University of Sussex, Brighton, UK.
Email: lionelb@cogs.susx.ac.uk. Web: http://www.cogs.susx.ac.uk/users/lionelb/

Abstract

A real-valued evolutionary optimisation scenario affords a unique opportunity to render visually explicit the "landscape" metaphor in the concept of *fitness landscape*.

Introduction

This short paper presents some preliminary investigations into the visualisation of fitness landscapes and evolutionary processes for an optimisation problem involving real-valued parameters. The optimisation problem - evolution of an XOR logic gate by a continuous-time, recursive neural network (CTRNN) - is intended to represent a "generic" scenario where, from a *dynamical systems* perspective, a successful solution depends essentially on the *phase portrait* of the system; that is, a successful solution depends on the *attractors* - the long-term behaviour - of the system.

Experimental setup

We have a 3-node fully connected, unbiased, continuous-time, recursive neural network or *CTRNN* (see for example: (Beer and Gallagher, 1992)) described by the equations:

$$\dot{x_i}(t) = -D_i x_i(t) + \sum_{j=1}^{3} w_{ij}\sigma\left(x_j(t)\right) + I_i(t) \qquad (1)$$

where $x_i(t)$ is the activation of the i-th node at time t, w_{ij} is the weight connecting node j to node i, $I_i(t)$ is the input and D_i the decay constant for the i-th node. The input-output transfer function $\sigma(x)$ is the sigmoid:

$$\sigma(x) = \tanh(x) \qquad (2)$$

Note that the gain of the transfer function and decay (time-course) parameters are fixed: the only variable parameters are the 9 weights w_{ij} which mediate the interactions between the nodes; they are real numbers that may be positive (excitory), negative (inhibitory) or zero (no connection). In simulation, node activations are calculated forward through time by straight-forward time-slicing using Euler integration.

The CTRNN has to solve an XOR gate as follows: nodes 1 and 2 are designated as inputs, node 3 as output[1]. A network trial consists of four runs of the network, corresponding to XOR logic (Table 1). Inputs were

Input 1	Input 2	Output
high	*high*	*low*
high	*low*	*high*
low	*low*	*low*
low	*high*	*high*

Table 1: XOR logic

chosen as $low = -0.5$, $high = +0.5$ and target outputs were chosen as: $low = -\tau$, $high = +\tau$ with $\tau = 0.6$. Node activations are initialised for each of the four runs to[2] $(0, 0, -0.2)$. All decay parameters were set to 0.1 and the step size for Euler integration was 0.01.

During a run the appropriate (constant) inputs are fed continuously to the input nodes. Networks are run for a *stabilisation period* $T_s = 1000$ time steps to allow the network to fall into an *attractor* state, followed by an *evaluation period* of a further $T_e = 2000$ time steps; the idea is that a network be evaluated on its steady-state behaviour. Networks receive a fitness value f based on the mean distance between output and target (i.e. distance integrated over the evaluation period) averaged over the four runs; specifically:

$$f = \left[1 + \frac{k}{4(1+\tau)} \sum_{r=1}^{4} \Delta^{(r)}\right]^{-1} \qquad (3)$$

[1] We actually used the *sigmoided* output $\sigma\left(x_3\right)$ since it is guaranteed to lie between -1 and $+1$.

[2] The "off-centre" x_3 initial activation of -0.2 was found to aid finding solutions, apparently through some kind of *symmetry breaking* effect.

with mean distance between target and output for the r-th run defined by:

$$\Delta^{(r)} = \frac{1}{T_e} \int_{T_s}^{T_s+T_e} \left| \sigma\left(x_3^{(r)}(t)\right) - \tau^{(r)} \right| dt \qquad (4)$$

where $x_3^{(r)}(t)$ is the node 3 activation and $\tau^{(r)}$ the appropriate target (i.e. $\pm\tau$) for the r-th run. The constant "stretch factor" k may be deployed to control selection pressure. Note that maximum possible fitness is 1 and minimum possible fitness is $\frac{1}{1+k}$ since the maximum mean distance $\Delta^{(r)}$ for each of the 4 runs is $1+\tau$. We used $k = 10$ for our experiments.

Network behaviour visualisation[3]

Successful solutions of the XOR problem by a CTRNN may be characterised by a phase portrait for which the initial values fall into the basin of attraction of an attractor with node 3 (output) activation *high* (resp. *low*) when the inputs are *high*, *low* or *low*, *high* (resp. *high*, *high* or *low*, *low*) - see Figure 1.

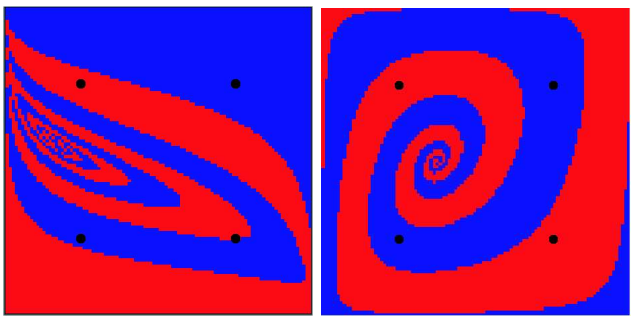

Figure 1: Node 3 (output) attractors of two successful networks: x, y axes (varying from -1 to $+1$) represent inputs to nodes 1 and 2, while red (resp. blue) represents an attractor with *high* (resp. *low*) node 3 activation. To solve the XOR-gate the points (0.5, 0.5), (0.5, -0.5), (-0.5, -0.5), (-0.5, 0.5) - marked by black dots - must go: low, high, low, high; i.e. blue, red, blue, red.

Figure 2 illustrates the approach to attractors. The setup is similar to Figure 1, except that now height represents node 3 activation after a limited number of time steps, so that the system has not quite settled into an attractor. As the number of time steps increases, these figures become increasingly "sharp-edged", approaching (in vertical projection) the appearance of Figure 1. In Figure 3 surfaces for three nearby weight configurations are plotted together (one weight has been varied slightly), illustrating the changing phase portrait as we move through weight-space.

[3]Technical note. The visualisations in this paper were made using the Geomview program (`http://www.geomview.org/`).

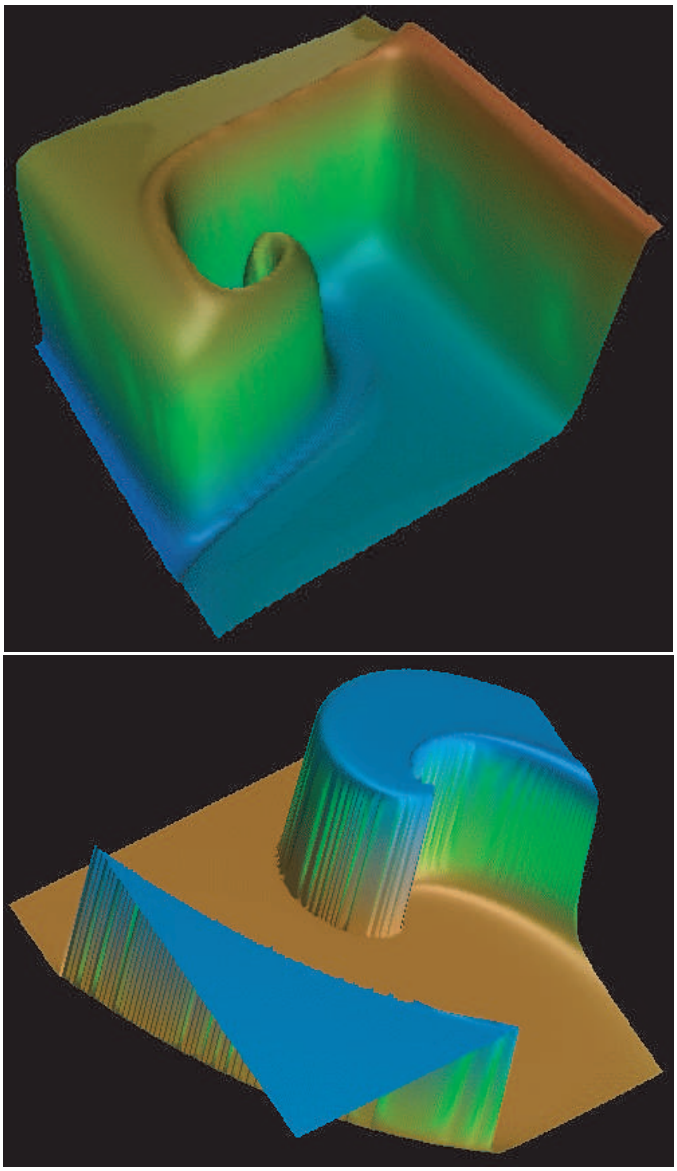

Figure 2: Approach to attractors: x, y axes again represent inputs, while height is node 3 activation after a limited number (500) of time steps.

Figure 4 gives two views of the changing phase portrait as we move through weight-space[4]. The colour dimension (*cold..warm*) represents variation of a single weight. The other two dimensions again represent inputs. Points are plotted for each colour (weight) along the *separatrices* on the input plane for which the node 3 activation attractors change from *high* to *low* (the seperatrices correspond to the boundaries between red and blue regions

[4]This would look better if the separatrices were joined-up, or plotted as surfaces. Geomview doesn't seem to be able to do this, perhaps because parametrising the separatrices would seem to be non-trivial.

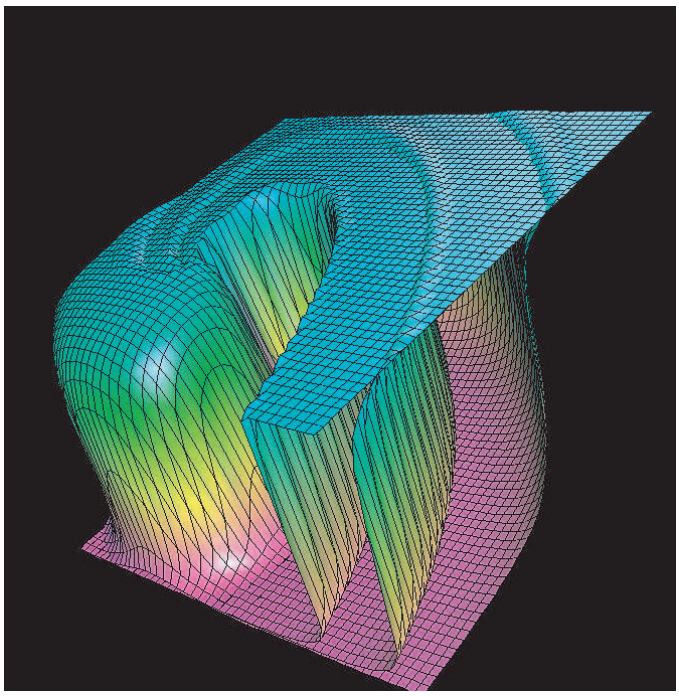

Figure 3: Approach to attractors: surfaces are plotted (as in Figure 2) for three nearby weight configurations - one weight has been varied slightly - illustrating the changing phase portrait as we move through weight-space.

in Figure 1). Bifurcations can be seen where "pinching" occurs.

Landscape visualisation

Figure 5 illustrates some 2-dimensional slices through the (9-dimensional) CTRNN XOR fitness landscape: x, y axes represent some two out of the nine weights; height is fitness as given by Equation 3. Landscapes are plotted in the vicinity of (near) optimal solutions - the red-tipped "peaks" in the figures.

Evolutionary visualisation

We deployed a simple $1 + 1$ Evolution Strategy with Gaussian-creep mutation - i.e. a (real-valued) random-mutation hill-climber, or *netcrawler* (Barnett, 2001) - on the CTRNN XOR landscape. The hill-climber (i.e. the current weight configuration) is initially dropped in weight space according to an (uncorrelated) multivariate Gaussian distribution with standard deviation of 2.0 centred on the zero-weight network, and its fitness evaluated. At each generation a new weight configuration is generated randomly, again according to a multivariate Gaussian with (fixed) standard deviation of 0.02, this time centred on the current configuration; fitness of the new configuration is then evaluated. If the fitness of the new configuration is greater than or equal to that

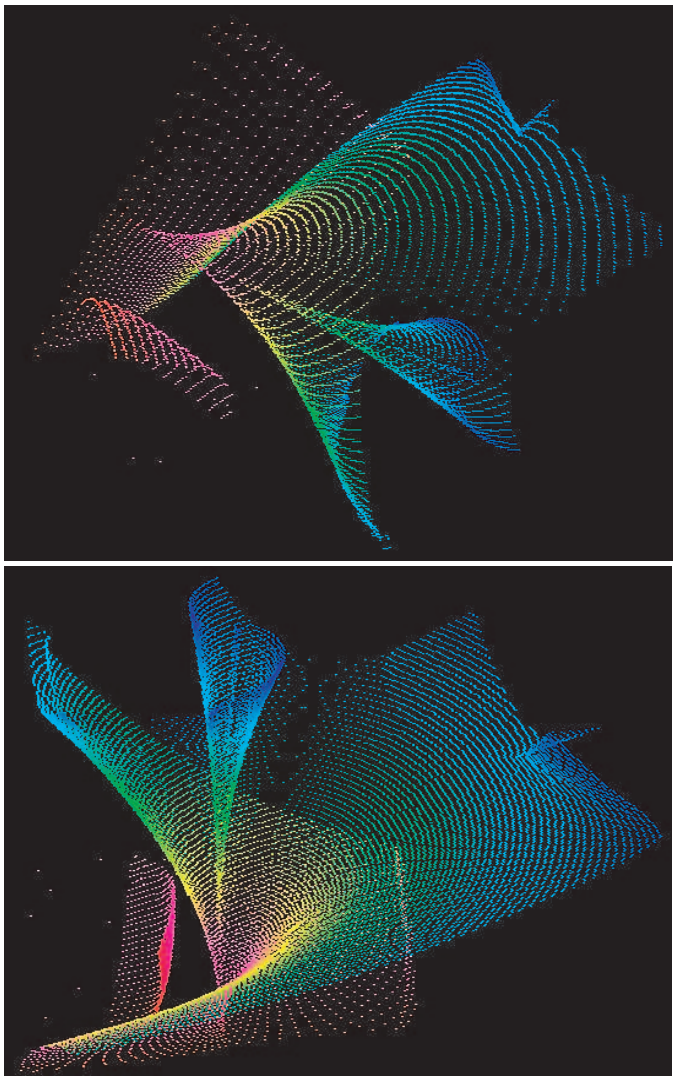

Figure 4: Varying phase portrait as we move through weight space: colour (*cold..warm*) represents the value of a single varied weight. The other two dimensions again represent inputs. For each weight value points are plotted along the separatrices on the input plane between regions corresponding to *high* and *low* attractors (see text).

of the current configuration, then the new configuration replaces the current; otherwise the current configuration is retained.

It was found that near-optimum solutions could be obtained quite easily within a few thousands of generations. We then "clamped" some seven of the nine weights in the network to their values at a fitness (near-)optimum and repeated the evolution experiment, now with variation constrained to the two unclamped weights - i.e. to 2 dimensions in weight space. Some paths followed by the hill-climber on the fitness landscape are plotted in Figure 6; in these figures a 2-dimensional "slice" of the landscape is drawn (as in Figure 5) with the same

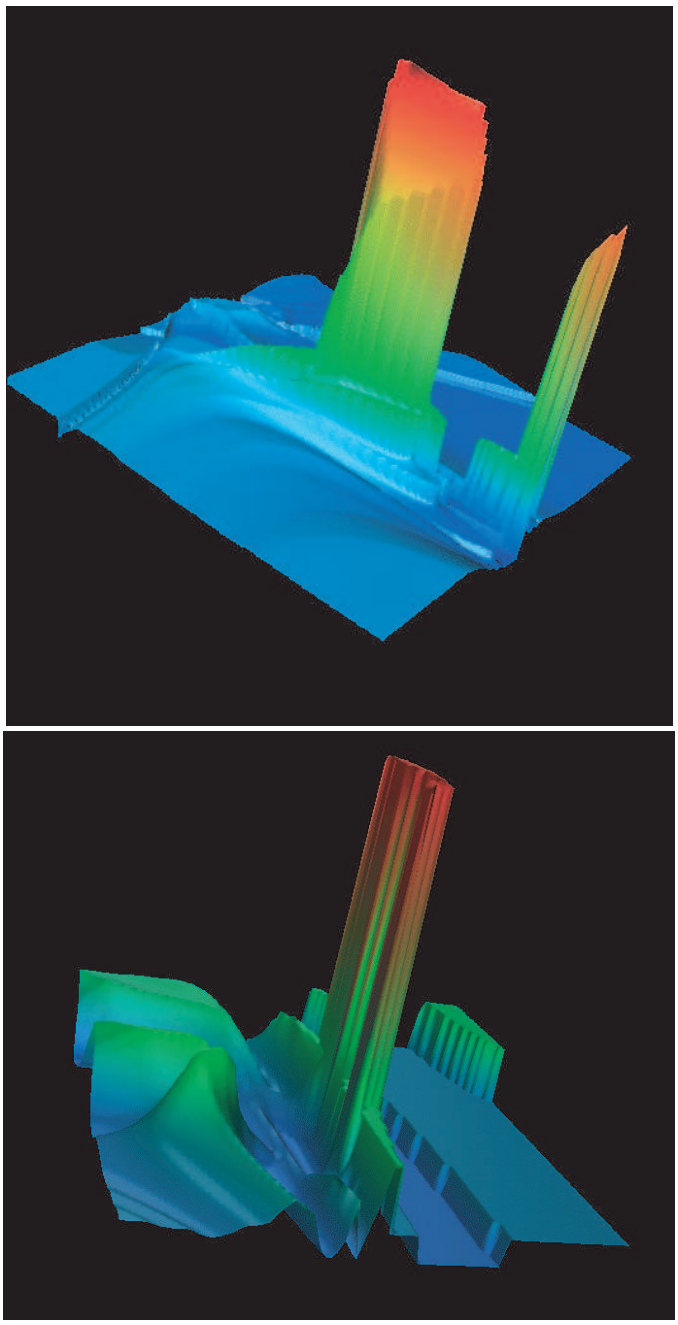

Figure 5: Two 2-dimensional slices through the (9-dimensional) fitness landscape for the XOR problem.

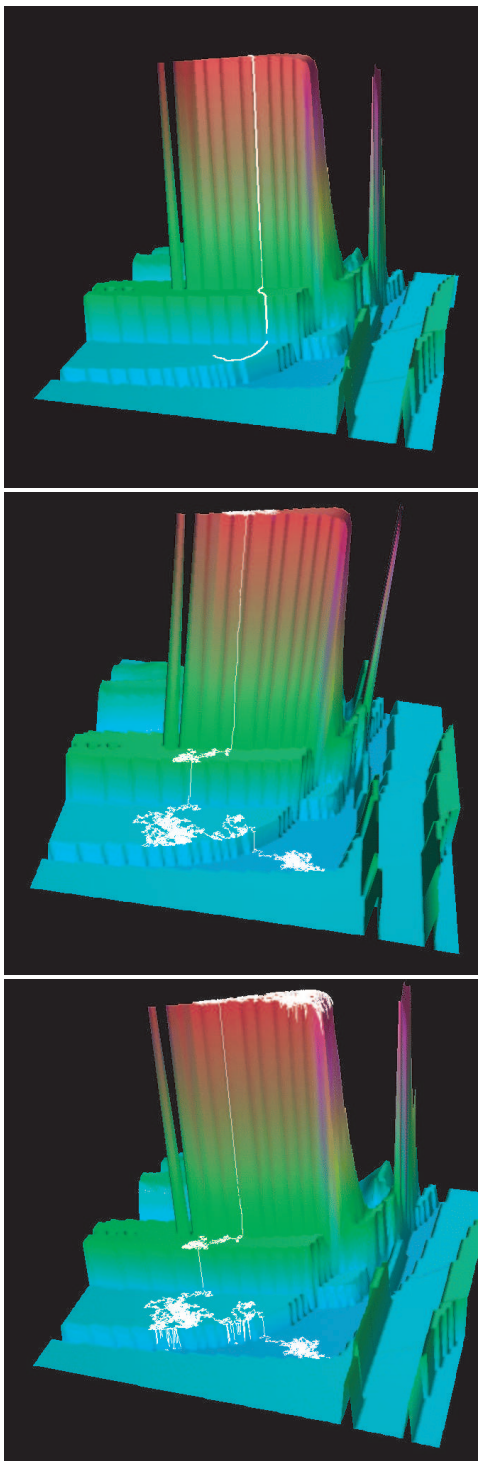

Figure 6: The path of a hill-climber (constrained to 2 dimensions) on the XOR fitness landscape. Noise on fitness evaluation increases from the top to bottom figure.

weights clamped as for the constrained evolutionary run - the hill-climber thus appears to "stick to the surface" of the landscape.

Looking at the top figure, we note that the apparently flat (*selectively neutral*) plateaux are actually not quite flat, with slight gradients that (happily for the hill-climber) tend to lead to the "cliff bases" - at least in this region/projection of the landscape. In the middle figure we've added some (Gaussian) *noise* to fitness evaluation, so that the plateaux are now flat as far as the hill-climber is concerned - the noise effectively obliterates the slight

gradient. The hill-climber thus drifts randomly on the plateaux until it chances upon a cliff base, which it then climbs. Sometimes (not in the run illustrated) it will climb the "wrong" cliff on the initial (lowest) plateau; it still tends to find a path to a near-optimum eventually, but only after many more fitness evaluations. In the bottom figure even more fitness noise has been added. The hill-climber now sometimes "falls off" its current plateau, although selection pressure is generally strong enough to drag it back up.

Discussion

Our CTRNN XOR landscape was designed as a simple model for an evolutionary optimisation scenario where the effectiveness of an evolved artefact depends on the long-term behaviour of some *dynamical sytem*; several classes of highly non-trivial real-world design-by-evolution problems (including some evolutionary electronics and evolutionary robotics problems) might be viewed as systems of this type. Our perspective is that we are, essentially, evolving *phase portraits* - the sets of attractors and their basins of attraction (Figures 1, 2) - of dynamical systems. The topography of the resulting fitness landscape thus depends on the manner in which the phase portrait varies (Figures 3, 4) as we move about within the space of configurations - i.e. the parameter space - of our dynamical system. This highly *geometric* view of evolution not only allows full rein to our spatial intuition, but also enables application of the well-developed machinery of dynamical systems theory to our understanding of fitness landscape structure and evolutionary processes.

Perhaps the most striking aspect of the CTRNN XOR landscape is the shear richness of the topography; even in 2-dimensional slices of the full 9-dimensional surface (Figures 5, 6) a bewildering assortment of spikes, ridges, saddles, twisting gullies, shelves, crenelations and undulating plains are in evidence. We might, perhaps, contrast this complexity with the (arguably simplistic) "test suites" of landscapes that permeate the evolutionary optimisation literature. These test landscapes, often built on simple mathematical functions, tend to be designed to illustrate some particular element - eg. multi-modality - deemed "difficult" or "deceptive" for evolutionary optimisation. For our landscape it seems likely that any evolutionary algorithm proven effective in dealing with one "difficult" landscape feature - an algorithm that is adept, for instance, at escaping local sub-optima - may well founder on some other exotic landscape feature. To be effective on our landscape an evolutionary search procedure has to overcome a formidable array of challenges; and our landscape is itself almost certainly a "toy" - of trivial complexity - in comparison with fitness landscapes for many real-world optimisation scenarios (eg. evolutionary design of a neural network controller for a robot navigation task). Thus we would argue that simple test landscapes may be less than useful when it comes to the design of effective search algorithms for real-world problems. We remark that preliminary research suggests that some form of annealed mutation-based hill-climber may be more effective on the CTRNN XOR landscape than, say, a population-based genetic algorithm with recombination.

Another interesting factor that emerges from our simple experiments is the role that *noise* may have to play in optimisation. It might be remarked that while fitness evaluation for real-world problems will frequently be inherently noisy, here we could argue a case for the deliberate *addition* of noise to (effectively) obliterate deceptive gradients - gradients, that is, which may lead to evolutionary cul-de-sacs.

Finally, we remark that the parameters in our model - the network weights - are inherently real-valued (rather than binary) and in our evolutionary experiments we operated directly on the real parameters; thus eg. we use Gaussian creep mutation. In the evolutionary optimisation literature there is frequently an emphasis on evolving some binary encoding of the "phenotype" (an exception being *evolution strategies*). We might, thus, have transformed our real-valued network weights via a binary or Gray code, say, into a binary bitstring and deployed bitwise mutation. There does not, however, seem to be any convincing reason to use such a binary representation; indeed, it might be argued that we retain more control over evolutionary variation by working with the real parameters themselves.

References

Barnett, L. (2001). Netcrawling - optimal evolutionary search with neutral networks. In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC'2001)*. IEEE Press, Piscataway, New Jersey.

Beer, R. D. and Gallagher, J. C. (1992). Evolving dynamical neural networks for adaptive behaviour. *Adaptive Behaviour*, 1:94–110.