8

Connectionism and the Dynamical Approach to Cognition

8.1 Are We on the Road to a Dynamical Revolution?

We saw in chapter 1 that connectionism has its roots in the 1940s, when ideas about the nervous system, computation, and intelligence came together synergistically. We briefly traced the path from *formal neurons* to *perceptrons* and noted that events in the 1960s led to a period in which these neural network models became overshadowed by symbolic models of intelligence. With the rise of connectionism in the 1980s neural networks regained prominence, and for two decades the two approaches have each sought to dominate cognitive science.

This is part of a larger story. The energetic collaborations of the 1940s had multiple, sometimes intertwined strands, of which the best known is an emphasis on feedback and control in the *cybernetics* of Norbert Wiener and others. This required attention to the dynamics of change in time, an aspect of the functioning of systems which has often been ignored by connectionists (though not by Stephen Grossberg and his group, who continued to advance network research during the period when symbolic models were dominant; see Grossberg, 1982). In an even broader context, the dynamics of complex physical systems have been mathematically describable at least since Newton (building on work by Galileo) formulated his three laws of motion and developed the calculus to gain the ability to predict specific planetary configurations. New geometrical approaches to dynamics by Poincaré in the late nineteenth century prepared the way for the rise of Dynamical Systems Theory (DST) in the twentieth century. Initially applied primarily to physical phenomena such as eddies in a stream (Landau, 1944), by the 1980s DST was being extended to motor coordination by Michael Turvey, Peter Kugler, and J. A. Scott Kelso (see Kelso, 1995) and by the 1990s to the development not only of coordinated activity but also more cognitive capacities (see the 1994 book by Esther Thelen and Linda Smith). Although the mathematics of nonlinear dynamical systems can be daunting, DST has spawned compelling graphics that help to provide an intuitive grasp of key concepts (among the suggested readings at the end of the chapter, see especially the copiously illustrated introduction by Abraham and Shaw, 1992).

The idea that cognition is dynamic and can best be understood using the tools of DST attracted increasing attention across the 1990s but is still somewhat of a frontier outpost in cognitive science. Much of the excitement was conveyed in *Mind as Motion*, a 1995 book originating in a 1991 conference which brought together many

of the pioneering modelers. The editors, Robert Port and Timothy van Gelder, wanted to convince a broad audience that DST has revolutionary implications for cognitive science. In an introductory chapter they characterized cognitive science as wedded to what they call the computational approach (i.e., symbolic modeling) and called for a change, contending that "dynamical and computational systems are fundamentally different *kinds* of systems, and hence the dynamical and computational approaches to cognition are fundamentally different in their deepest foundations" (van Gelder and Port, 1995, p. 10). Further, they portrayed the emergence of the dynamical approach in cognitive science as a Kuhnian revolution:

> The computational approach is nothing less than a research paradigm in Kuhn's classic sense. It defines a range of questions and the form of answers to those questions (i.e., computational models). It provides an array of exemplars – classic pieces of research which define how cognition is to be thought about and what counts as a successful model. . . . [T]he dynamical approach is more than just powerful tools; like the computational approach, it is a worldview. The cognitive system is not a computer, it is a dynamical system. It is not the brain, inner and encapsulated; rather, it is the whole system comprised of nervous system, body, and environment. The cognitive system is not a discrete sequential manipulator of static representational structures; rather, it is a structure of mutually and simultaneously influencing *change*. Its processes do not take place in the arbitrary, discrete time of computer steps; rather, they unfold in the *real* time of ongoing change. . . . The cognitive system does not interact with other aspects of the world by passing messages or commands; rather, it continuously coevolves with them. . . . [T]o see that there is a dynamical approach is to see a new way of conceptually reorganizing cognitive science as it is currently practiced. (van Gelder and Port, 1995, pp. 2–4)

If computational and dynamical worldviews are poles apart, connectionism occupies a somewhat more ambiguous territory in between. Highly interactive networks, such as Boltzmann machines, are dynamical systems of considerable interest in principle. In practice, they are hard to use and hard to analyze even with the availability of DST tools. Feedforward networks have made the greatest inroads into cognitive science, in part due to their tractability, but the only aspect of this architecture that is dynamical is the adaptation of weights during learning. Most connectionist models, even interactive ones, carry some symbolic/computational baggage and therefore are not the best poster children for van Gelder and Port's revolution. We see a somewhat different future, in which connectionist modeling can benefit from both computational and dynamical approaches and can sometimes even combine them within the analysis of a single network.

In what follows we will introduce some basic dynamical concepts and tools in section 8.2; describe how the simplest concepts have been utilized in four areas of network research in 8.3; and describe how the concept of chaos has been utilized in two network models in 8.4. Then in 8.5 we return to philosophical issues raised by van Gelder and Port. From their overall claim that classic connectionism occupies an untenable halfway position between the computational and dynamical approaches, we move to more specific arguments concerning explanation (countered by Bechtel) and representation (countered by Clark and Wheeler). The counter-arguments lead towards a more inclusive cognitive science, and in 8.6 we discuss a controversial version offered by philosophers Terrence Horgan and John Tienson. Let us start, then, by introducing some concepts and tools from the mathematical core of the dynamical approach: dynamical systems theory (DST).

8.2 Basic Concepts of DST: The Geometry of Change

8.2.1 Trajectories in state space: Predators and prey

If a picture is generally worth a thousand words, in the case of dynamical systems theory each picture is worth at least ten times that many: among DST's innovations is the adroit use of geometrical representations to help conceptualize how systems change. The simplest picture is a plot of the states traversed by a system through time, that is, the system's *trajectory* through *state space*. The trajectory is a continuous curve if the system is defined in real time, or a sequence of points in discrete time. Each dimension of state space corresponds to one variable of the system, and each point in the space corresponds to one of the possible states of the system. For systems with one continuous variable, the state space is a range of values on one dimension (e.g., the frequency of a tone; the height of a person; the firing rate of a neuron; the population size of a species in its habitat). The trajectory of a pure tone is a regular oscillation between two values (if a time dimension is added to the plot, a sine wave is obtained). A trajectory for height starts with an infant's height at birth and rises (irregularly) over time.

To move up to a two-dimensional state space, the most obvious way is to consider an additional variable; for example, you can visualize your child's growth by graphing height and weight on orthogonal axes and plotting a point each week; connecting the points approximates the child's trajectory through height-weight space in real time. Another way of moving up to a two-dimensional state space is to examine the same variable for two different individuals (or neurons, populations, network units, etc.). Keep adding more, and you end up with a high-dimensional system. For example, the activation values across a 90-unit output layer in a feedforward connectionist network can be represented as a single point in a 90-dimensional state space – each unit's activation is treated as a separate variable of the system. If the network is feedforward, its response to an input is one point in the space. If it is interactive, the changing activation values are represented as a sequence of points – that is, as a trajectory through activation state space – but the outcome may be similar since some trajectories simply converge on a point and remain there.

A two-dimensional state space is much easier to visualize than a 90-dimensional one, so we will use the case of two species in a predator–prey relationship to illustrate some key concepts (the case is described by Ralph Abraham and Christopher Shaw, 1992, pp. 82–5). The classic account was proposed independently by Alfred James Lotka (1925) and Vita Volterra (1926). Our first picture, figure 8.1(a), shows several variations on an idealized *cyclic trajectory* (also called a *periodic trajectory*) in the state space for number of prey (horizontal axis) and number of predators (vertical axis); it was inspired by periodicity in the population sizes of different species of fish in the Adriatic Sea. To understand the cyclic changes in population size, four parts of the outermost curve (labeled I–IV) can be considered separately. (See below for discussion of the whole family of curves.) When there are relatively few of both predators and prey, the number of predators declines for lack of food while the number of prey increases for lack of predators (region I). The increase in prey, though, provides a more ample food source for the predators, and beyond a transition point, the number of predators will begin to increase along with the number of prey (region II). But the increase in predators results in increased consumption of

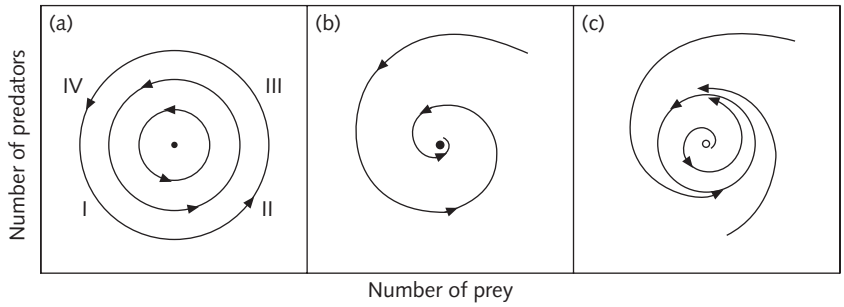

Figure 8.1 Possible trajectories through state space for interacting predator and prey populations: (a) cyclic trajectories (no attractor); (b) point attractor with spiraling transient; (c) cyclic attractor with spiraling transients beginning at points inside and outside of it.

prey, so after another transition point the number of prey declines while predators continue to increase (region III). But eventually the shrinking prey population leads to starvation for predators and both predators and prey decline in population (region IV). When the number of predators becomes sufficiently low, the number of prey will begin to increase again (region I). And so forth: in principle, each population's size is predicted to *oscillate* (move between two extremes) forever, with the same *period* (elapsed time) on each cycle. The prediction derives from the Lotka–Volterra equations, in which the rate of change of each population depends on the current number of prey (x) and predators (y) as well as the values of the *control parameters* (A, B, C, and D):

$$\frac{\partial x}{\partial t} = (\mathrm{A} - \mathrm{B}y)x \tag{1}$$

$$\frac{\partial y}{\partial t} = (\mathrm{C}x - \mathrm{D})y \tag{2}$$

To obtain a cyclic trajectory using these equations, an appropriate set of parameter values must be identified (not all values will produce such a trajectory).[1] As shown in figure 8.1(a) for one such set of values, the equations then yield a family of concentric closed curves around a central equilibrium point. Within that family, the particular curve – ranging from no oscillation at the center to the extreme population swings of the largest-diameter circle – depends upon the initial values of x and y. Once the system embarks on one of these trajectories, it will repeat it indefinitely – unless perturbed by some change outside the system. For example, unusually high temperatures may increase the predators' mortality rate, D, and also affect reproduction and predation rates, reflected in A–C.

Another way to get different trajectories is to change the equations. In fact, later researchers found that the Lotka–Volterra equations alone are unrealistic (e.g., they make no provision for competition among prey or predators for limited resources). One kind of revision to the system of equations adds "ecological friction" (by analogy to the physical friction that brings a pendulum asymptotically to rest by damping its oscillations). Figure 8.1(b) shows how this produces a very characteristic DST state space plot. The illustrative trajectory now has two parts. The *point attractor* at the center (also called the *limit point*) is stable – if exactly these predator and prey population sizes are attained, the system is in equilibrium and will remain

in that state. Thus, the attractor is the stable part of the trajectory, and can be viewed as describing the long-term behavior of the system. The curve that leads to the attractor is a *transient*, the part of the trajectory that the system traverses as it moves from its initial state towards equilibrium. The spiraling transients in this system reflect a story similar to the one we told for sections I–IV of the outer cycle in figure 8.1(a), except that here the system approaches the equilibrium point as the oscillations in population size diminish in amplitude. When the transient spirals towards a point like this, the system is a *damped oscillator* and the point attractor is also called a *focal point*.

Crucially, trajectories from many different initial states will converge on the same focal point: for various initial numbers of predators and prey, the populations will approach this point of equilibrium along an appropriate, spiraling transient. In fact, it is primarily the convergence of nearby trajectories to this equilibrium point that qualifies it as an attractor (cf. the equilibrium point in figure 8.1(a), which is not an attractor); and it is specifically a point attractor (limit point) because the subset of state space to which the trajectories converge (the *limit set*) consists of just one point. The set of all initial states whose trajectories converge on this attractor is its *basin of attraction* (generally a region of state space but possibly the entire space). The state space – also called *phase space* – filled with the possible trajectories of this system is its *phase portrait*. In figure 8.1(b) this was reduced, for display purposes, to a representation of the point attractor (by convention, a small solid circle) and one typical trajectory. For systems with multiple attractors (and perhaps other special features, such as *repellors*, *saddle nodes*, and the *separatrices* that may form boundaries between basins), a larger number of trajectories or special display conventions are needed to convey the essentials of the phase portrait.

This is a good place to pause and note that considerable idealization is involved in using even the modified system of equations to model changes in the population sizes of two species in a predator–prey relationship. First, the real-life populations are not a closed system; as already mentioned, external factors such as ocean temperature can affect parameter values. Second, here as in many other dynamical systems (those classified as *dissipative*), convergence is asymptotic – the state of the system approaches the attractor as time approaches infinity. Beyond the formal nature of this characterization, it also assumes a continuity that cannot be attained in population dynamics. Each birth or death brings a discrete change in the value of x or y. At best these values will jiggle around in the vicinity of the equilibrium point as individuals are born and die. This leads to another notion that is worth making explicit: even if the system could reach true equilibrium, what is stable is the value of two collective variables. Out in the Adriatic Sea, individual fish are still giving birth and dying, eating or being eaten. Trajectories of change in the lives of individual fish are not inconsistent with lack of change in the size of their two populations.

A phase portrait that is somewhat more realistic for the Adriatic case (though still idealized) can be obtained by making one more revision to the system of equations. Figure 8.1(c) shows a *cyclic attractor* (also called a *periodic attractor* or *limit cycle*) along with a few of the possible transients. This is a form of stable behavior that at least involves movement – the state of the system endlessly cycles around in state space rather than remaining at a single point as in (b). But the comparison to portrait (a) is even more informative. The circle in the middle of (c) looks the same as one of the circles in (a), but the dynamics producing it are quite different. As a cyclic attractor, it is the stable part of a variety of trajectories rather than a single trajectory.

That is, in (c) many different starting points all converge on the same circle (that is why it is called an attractor), whereas in (a) the starting point is already part of the circle and determines its diameter.[2] (Note that although both of these cycles are circles, in other systems they might be any sort of closed curve; we will see much odder-shaped limit cycles in figure 8.3(a).) Examining (c) more closely, it can be seen that a trajectory that begins with many predators and many prey and another that begins with very few predators but almost as many prey both have transients that converge on the limit cycle; that is, the long-term behavior of both trajectories is the same. Trajectories with transients inside the limit cycle also converge on it; in the one exemplar shown in (c), the trajectory begins near a *point repellor* (a point from which trajectories diverge – the opposite of an attractor – shown conventionally as a small hollow circle) and spirals out to converge on the limit cycle. Thus, the basin of attraction is extensive.

8.2.2 Bifurcation diagrams and chaos

Some systems exhibit behavior that is more complex than in figure 8.1. In addition to the possibility of multiple attractors and other special features, there exist unstable trajectories which appear random despite following a deterministic path (i.e., the trajectory never repeats itself, but from any given point in the state space there is an algorithm that determines the next point). Perhaps more dramatically than necessary, this is called a *chaotic trajectory*. When the (infinite number of) possible chaotic trajectories of a system exhibit the characteristics of an attractor, the system is said to have a *chaotic attractor*. In particular, trajectories that begin near each other in the (infinitely large) limit set of the attractor will tend to diverge (a characteristic known as *sensitive dependence on initial conditions*), while trajectories that begin in the limit set's basin of attraction will converge on it. Chaotic attractors tend to be topologically complex. One of the simpler examples starts with the doughnut-shaped torus, which can be thought of as offering an infinite number of cyclic trajectories. A system with a torus attractor in its phase portrait will exhibit quasi-periodic behavior (continuously circling the torus but not repeating any particular cycle). However, the behavior becomes unstable for a torus in a very high-dimensional space: the system samples various cyclic trajectories along the surface of the torus, jumping from one to the next at irregular intervals. This unstable sampling of cycles is a chaotic trajectory. And this glimpse of a complex field of mathematics will have to suffice here.

Chaotic behavior is one innovative concept of DST; another is the importance of parameter values in whether a system exhibits that behavior. Even simple nonlinear systems may exhibit *phase transitions* (rapid shifts from one phase portrait to another) when the value of one or more control parameters changes slightly. For example, a single difference equation may produce a single point attractor, a cyclic attractor (limit cycle), or a chaotic attractor for x depending on the value of one control parameter A. The rapid transition in dynamics, brought about by a small but critical change in the parameter, is referred to mathematically as a *bifurcation*. The simplest example (from which the general term derives) is a *pitchfork bifurcation*, wherein a single point attractor splits in two. To explain and illustrate this concept, we will use a *bifurcation diagram* to display the varied behavior of a well-studied type of system defined in discrete time. It is specified by the logistic equa-

tion (which should not be confused with the logistic activation function introduced in chapter 2):

$$x_{t+1} = \mathrm{A}x_t\,(1 - x_t) \tag{3}$$

where x is a variable with range $0<x<1$ and A is a control parameter from the range $0<\mathrm{A}<4$. The subscripts t and $t + 1$ index successive time-steps t_0, t_1, t_2, t_3, and so forth; any value x_{t+1} depends in part on the value x_t on the previous time-step. We can begin by simply examining the sequence of values taken by x (its trajectory) when A is fixed, as it is generally assumed to be for a given system. Here is the calculation of the first five points of the trajectory when $\mathrm{A} = 3$ and the initial state $x_0 = 0.5$:

$x_1 = 3 \times 0.5 \times 0.5 = 0.75$
$x_2 = 3 \times 0.75 \times 0.25 = 0.5625$
$x_3 = 3 \times 0.5625 \times 0.4375 = 0.7383$
$x_4 = 3 \times 0.7383 \times 0.2617 = 0.5796$
$x_5 = 3 \times 0.5796 \times 0.4204 = 0.7310$

It can be seen even from these few points that the system is behaving as a damped oscillator (its behavior through much but not all of the stated range). It is converging on a point attractor at 0.6667 and the transient is a discrete version of the continuous spiral in figure 8.1(b). That is, it alternates between high and low values rather than spiraling between them. (In both cases, since the oscillation is damped, the high and low values themselves keep changing as the system converges on the attractor.) If a different initial value of x is used, the high and low values may begin further apart or closer together but will converge on the same attractor value of 0.6667.

From all this detail about the trajectories of x when $\mathrm{A} = 3$, the only information needed for the bifurcation diagram is the value of the point attractor, 0.6667. In general the bifurcation diagram for the logistic equation shows, for values of A within some range, the stable (long-term) values that x may attain. The transients one would obtain from different initial values of x are ignored; only the stable value or values to which they converge are plotted. A single stable value is the simplest outcome, and that is how this equation behaves when A is between 0 and 3.

Figure 8.2 shows the bifurcation diagram for values of A between 2.6 and 4.0. This catches some of the simple part of the range. The first bifurcation (a pitchfork) appears just beyond $\mathrm{A} = 3$ where x's stable long-term behavior suddenly switches to an alternation between two values (that is, the point attractor is replaced by a periodic attractor with periodicity of 2). These values (the prongs of the fork) drift further apart as A increases, but what matters most is that the dynamics are qualitatively different before vs. after the point of transition, which is called a *bifurcation point*. Just beyond 3.4 another bifurcation point is reached, and the system's periodicity increases to 4.

At a value of A beyond 3.6 a different kind of bifurcation begins to develop, one in which the attractor ceases to be periodic and becomes chaotic. The darkened region is a rough representation of the fact that x is taking a nonrepeating sequence of values. The value at each time step is deterministic since it is generated by equation (3), but each value is a new one. The new phase portrait contains a *chaotic attractor*, which changes form as the value of A increases. Interestingly, there are even values

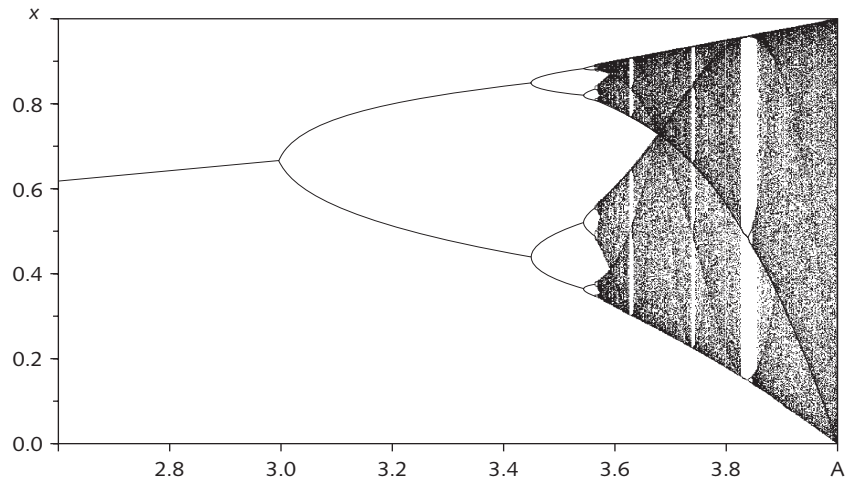

Figure 8.2 Bifurcation plot of the logistic function (1) for values of A between 2.6 and 4.0. For values of A less than 3.0 the function settles into a point attractor. Above 3.0 it bifurcates into a two-point attractor, than a four-point attractor, and so forth. Beyond 3.6 it enters a chaotic regime punctuated by periodicity within narrow ranges.

of A beyond the initial bifurcation to chaos for which x will once again exhibit simple periodicity. The values of x comprising each sequence show up in figure 8.2 as little lines within otherwise white bands near A = 3.6, 3.7, and 3.8.

8.2.3 Embodied networks as coupled dynamical systems

Now we can return to the assumption that A is a constant for a given system. This actually applies only if the system is an *autonomous dynamical system* – one that is unaffected by any other system. If so, trajectories through the state space can be specified in terms of equations (frequently nonlinear) which simply relate the variables of the system. Often, however, a system will be influenced by factors outside its boundaries (e.g., if we construe planet Earth as a system, variation in the sun's radiant energy is an external factor influencing this system). A *nonautonomous dynamical system* is one in which the values of one or more parameters vary due to external influences.

The dynamics get especially interesting if two systems are nonautonomous because they are *coupled*, with the states of each system influencing the values of parameters or variables in the other system across time. Thus, when the cognitive system is construed as a dynamical system, it may be further construed as coupled with other dynamical systems involving the organism's body and environment. In carrying out even a simple activity such as tapping a pencil, there may be reciprocal relations among three systems: the firing of neurons (brain), the movements of the fingers (body), and the tapping of the pencil against a surface (environment). This is the sort of interaction among brain, body, and environment that might best be modeled by construing them as coupled dynamical systems. (One can also construe

coupled dynamical systems as a single, autonomous dynamical system, but at a cost of complexity. Considerations of tractability often dictate treating the systems as separate, with each affecting the other by determining the values of some of its parameters.)

Dynamicists often celebrate coupling as a much more powerful and useful way of thinking about the interactions between the cognitive system and its environment than is offered by traditional perspectives. Both symbolic and connectionist modelers have found it difficult to advance beyond the "boxes in the head" models which began to replace stimulus–response models in the 1960s. This legacy includes a static characterization of the environment (e.g., the fixed corpus that provides input patterns to a network) and a cognitive model that acts with no further reference to the environment once an input is received. The input is operated upon (with interim results passed from one processor to another) and the result is sent out of the system as an output. For example, the DISCERN system (chapter 7) has a sequence of processing modules, each of which performs its own transformation on the input it receives and sends output to the next module (and, when the last module has completed its work, back out to the world). Most of DISCERN's modules are simple recurrent networks, but this allows only a very limited type of interactive processing (no settling dynamics, and connections remain within the bounds of the module). None of the modules has loops out to the environment and back or a means of changing its own operations in sync with ongoing changes in the environment. Interactive network designs offer some potential as tools to implement such loops, but in the absence of correspondingly complex characterizations of the environment (e.g., in terms of coupled dynamical systems) they do not overcome the limitations of traditional input–processing–output architectures.

Dynamicists are much more inclined to focus on the multiple ongoing interactions between the cognitive system and its environment, and some have already begun adapting the notion of coupling and other tools of DST to this ambitious project. In so doing, they have become natural allies of a diverse community of researchers who emphasize situated and embodied cognition. Rejecting the idea that one can study cognition solely as a set of processes occurring within an agent, these theorists focus on the interactions between cognitive operations and such external structures as the instruments in an airplane cockpit (Hutchins, 1995). Cognitive science has become increasingly receptive to this view and to the use of DST as a source of sophisticated tools for modeling more fine-grained transactions amongst the brain, body, and world – a trend celebrated by Andy Clark in the subtitle of his 1997 book, *Being There: Putting Brain, Body, and World Together Again*.

8.3 Using Dynamical Systems Tools to Analyze Networks

In applying dynamical systems tools to connectionist networks, one must decide which variables of the system to represent. In the case of connectionist networks, the two most plausible choices are the weights on connections (viewing learning as a trajectory through weight space) or the activation values of units (viewing processing as a trajectory through activation space). We begin with some examples in which state-space plots have been used to analyze trajectories through activation space. Of particular interest are the displays of attractors in phase portraits.

8.3.1 Discovering limit cycles in network controllers for robotic insects

Randall Beer (1995; see also Beer, 1997) has made effective use of state space plots to analyze connectionist networks designed to be coupled to a model insect in order to control its leg movements. The model insect has six legs, each with a foot at one end and a joint connecting it to the body at the other end. Each leg has three effectors: one raises or lowers the foot, and the other two apply opposing torques at the joint which combine to move the leg forward or backward. Each leg also has a single sensor that tracks the angle of the joint. The insect lives as a computer simulation, though it could also be embodied in a robot. Its walking is controlled by a 30-unit recurrent network composed of 6 subnetworks. The subnetwork controlling each leg consists of five units that are fully interconnected. Three are output units (motor neurons) which send instructions to the leg's three effectors, and two are connected only to the output units and to each other (interneurons). Each unit also receives weighted input from the leg's sensor, completing a loop between the leg and its controller network. That is, the dynamical systems of the body and the control network are coupled. Additional connections between subnetworks assure that the sensory information and motor activity of the six legs are coordinated.

With this basic architecture as a starting point, several different controller networks with task-adapted weights on their connections were created using a genetic algorithm procedure (see section 9.1). The fact that the weights were obtained by simulated evolution rather than learning is not crucial here; Beer's focus was on the dynamics of the systems once they had those weights. To contrast autonomous with nonautonomous dynamical systems, he manipulated access to sensory feedback. *Coupled networks* evolved with full access to input from the joint angle sensors (the network and the body were nonautonomous dynamical systems because each received input from the other); *autonomous networks* evolved with the sensors turned off (the body, still controlled by the network, was nonautonomous; but the network, lacking feedback, generated its own states autonomously); and *mixed networks* evolved with the sensors sometimes on and sometimes off.

All networks eventually could make the model insect walk employing the "tripod gait" characteristic of fast-moving six-legged insects: the front and back legs on one side would move in synchrony with the middle leg on the opposite side. While one set of three feet was swinging, the other set would remain on the ground, providing support. However, the three sets of networks fared differently when challenged in further testing. The coupled networks exhibited fine-tuned control of walking, but performed poorly if the sensors were turned off; they had evolved circuits that were dependent on a constant flow of sensory input and could not generate an appropriate sequence of states when forced to function autonomously. The autonomous networks produced walking in a stereotyped tripod gait regardless of whether sensory input was now made available; they autonomously cycled through the same sequence of states and had no means of incorporating a new, external variable to generate a modified sequence. The mixed networks worked as well as the coupled networks with sensory input, but could also function autonomously when input was removed.

To more closely examine these differing dynamics, Beer began with the relative simplicity of the autonomous networks and narrowed his focus to a single five-unit subnetwork controlling what he posited was a single-legged insect. Beer sought to

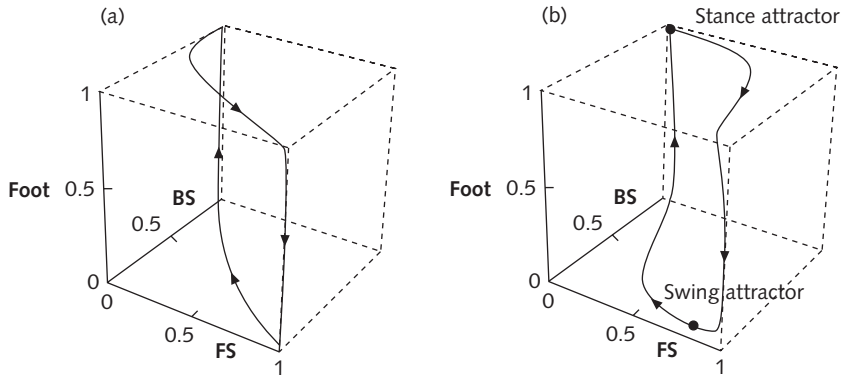

Figure 8.3 Trajectories in motor space for two of Beer's (1995) network controllers for a model insect. Successive activation values are plotted for the foot, backward swing (BS), and forward swing (FS) motor neurons in the subnetwork controlling one leg. (a) The autonomous network (sensor off) produces a limit cycle. When the activation values are in the upper back corner the foot is down and the network is in a stance phase; when activation values are in the lower front corner, the network is in a swing phase. (b) The coupled network (sensor on) exhibits a roughly similar trajectory, but it is produced by moving between two point attractors.

understand how the network controlled the leg's movement. He found that the five-dimensional state space for this simplified control system exhibited a limit cycle, which is projected into a three-dimensional motor space in figure 8.3(a). When sent to the effectors, this repetitive sequence of motor neuron activation patterns propels the leg repeatedly through a one-legged version of the tripod gait. For example, when the control network is in the state at the lower right front corner (the middle of the swing phase), the insect's foot is raised and leg is swinging forward while the torso remains still. In the state at the upper left rear corner (the middle of the stance phase), the foot is on the ground and the leg is still; the backward-swing torque is transmitted to the torso, propelling it forward. The shift from one phase to another depends upon reversing the relative dominance of the forward swing neuron (**FS**) and backward swing neuron (**BS**) as well as the activity of the foot neuron (values above 0.5 instruct the foot to be down); on each cycle there is one such shift into a swing phase and another into a stance phase. Following the trajectory from the rear to front corner, the relative dominance is shifting from the backward swing neuron (**BS**) to the forward swing neuron (**FS**); from the front to rear corner, the opposite shift occurs.

When the same kind of analysis is applied to a coupled network (one that evolved with the leg-angle sensor turned on), the results are superficially similar. Figure 8.3(b) shows that a limit cycle fairly similar to the one in figure 8.3(a) is obtained, and it produces essentially the same gait in the insect. The underlying dynamics are quite different, however. The leg angle is now variable, rather than taking a constant value of zero. Because the sensor supplies a stream of leg-angle values to the network, each point on the limit cycle has its own instantaneous trajectory. Most of these trajectories terminate in one of two point attractors, which are superimposed on the state space plot as solid circles. For example, each point on the leftmost portion of the limit cycle has a trajectory that terminates in the stance attractor (top circle). For the lowest of these points the trajectory to that attractor is relatively long, but as the network's states advance up the limit cycle the instantaneous trajectories get shorter.

Eventually the stance attractor is reached and the network remains in that state for a time. However, the state is one that produces forward motion of the body, which gradually changes the leg angle; under the leg-angle sensor's influence, the stance attractor disappears and the swing attractor appears. (Additional, unstable attractors make a brief appearance during the transition.) Now the state of the network progresses along the rightmost portion of the limit cycle in accord with a sequence of instantaneous trajectories terminating in the swing attractor.

While more interesting than the dynamics of the autonomous network, the reliance on sensory feedback to make the appropriate attractor appear has a price. If the coupling is broken by turning the sensor off, the network will get stuck on one trajectory leading to the stance attractor and, once it reaches that state, will have no way to move out of it. The insect will stand forever. The mixed network avoids this fate because its instantaneous trajectories are based on limit cycle attractors rather than point attractors. Not only does it not get stuck if input is removed; it exhibits an adaptive "functional plasticity" as its trajectories dynamically adjust to the flow of sensory feedback. For example, when Beer intervened by making the sensory input change more slowly than normal (as would happen if the insect's legs were longer), the mixed controller network became entrained by the sensory signal, slowing its own cycle to remain in phase. Beer (1995, p. 203) remarked: "The likelihood of anyone designing such a flexible and compact leg controller by hand is probably rather low." To continuously adjust to a changing environment, he finds the "messy" design of intermittently coupled dynamic systems more promising than the modular designs of engineers.

If Beer's study had been run purely as a network simulation, without the DST analysis, there would have been no plotting of limit cycles and instantaneous trajectories, no understanding of the role of attractor dynamics, and no explanation of why the constantly coupled network fails to function properly when the sensor is turned off but the intermittently coupled (mixed) network does. In the next section we show that DST is equally important for attaining a deeper understanding of a quite different class of models: layered interactive networks trained to simulate reading aloud. This more differentiated, cognitive task elicits the development of multiple point attractors.

8.3.2 Discovering multiple attractors in network models of reading

We saw in chapters 2 through 5 that feedforward networks can be used for a variety of tasks involving pattern recognition (e.g., assignment to semantic categories) and pattern transformation (e.g., past-tense formation). A simple way to add interactivity to such networks is to add recurrent connections between pairs of output units. Lateral inhibition produces dynamical interaction, in which even a small advantage in initial activation value can become a large difference as the system settles. The disparity in activation values between two units at time t is one determinant of the disparity in their inhibitory effects on each other at time $t + 1$, with the more active unit suppressing the less active unit via its stronger inhibitory signal. We saw lateral inhibition at work in the Jets and Sharks simulation in section 2.1, McClelland and Rumelhart's 1981 word-recognition model in section 4.1.2, and (along with lateral excitation) in the lexical and episodic memory modules of DISCERN in sections 7.5 and 7.7.

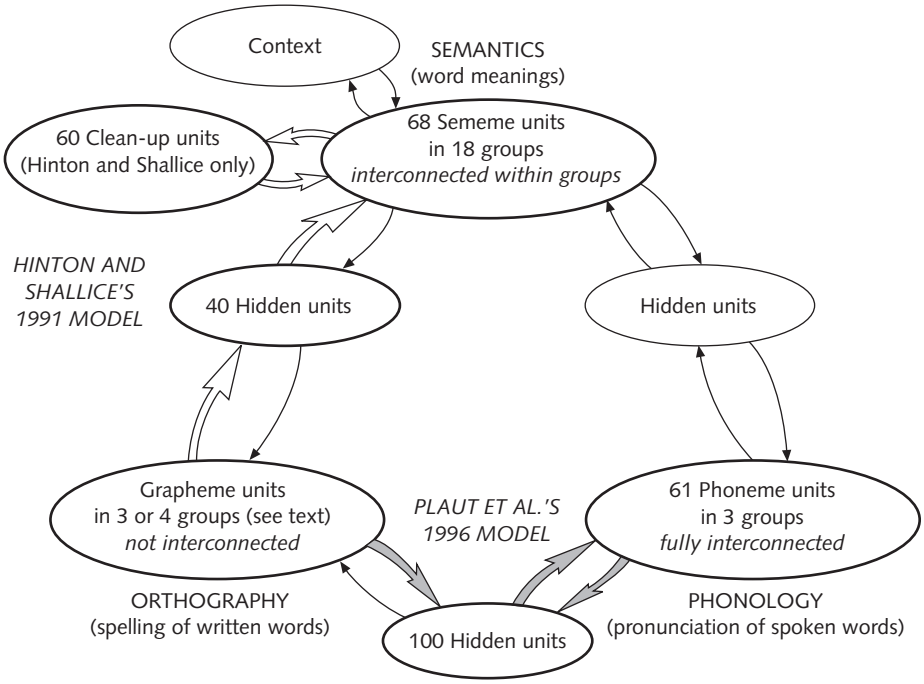

Figure 8.4 An overall framework for lexical processing adapted from Seidenberg and McClelland (1989). Those pathways that had been implemented by 1996 are in shown in boldface: the pathway from orthography to phonology (Plaut et al., 1996: filled arrows) and the pathway from orthography to semantics (Hinton and Shallice, 1991: hollow arrows). Neither model included the full set of feedforward and feedback connections of the original framework. Plaut et al. left out the hidden-to-grapheme feedback connections and Hinton and Shallice obtained interactivity by adding clean-up units rather than feedback connections. The models also differed in how they specified their grapheme units (see text).

In dynamical terms, we can say that such networks have multiple attractors and, when provided with an input, follow a trajectory in activation space that converges on the most appropriate one. In a network that learns to sort input patterns into four categories, for example, each of the four categories may develop its own point attractor in the activation state space for the output units. As long as the initial response to an exemplar is a pattern that falls into the basin of attraction for the appropriate category, the repeated revisions of each output unit's activation will gradually bring the pattern arbitrarily close to the desired pattern for that category (i.e., the system follows a trajectory that converges on the point attractor).

Interactive networks that develop multiple basins of attraction (*attractor networks*) have played a key role in a research area with a long and contentious history: accounting for how humans read. Figure 8.4 shows an overall framework for lexical processing adapted from Seidenberg and McClelland (1989), in which specialized groups of units (including groups of hidden units) interact with each other. To model reading, input is provided to the orthographic units. If the goal is to read aloud, then the system must generate a phonological output. If the goal is to understand the word, then it must retrieve a semantic interpretation. (Usually humans do both while reading aloud – they interpret the text as they pronounce it.) Since all of

the interactions between groups of units are bidirectional in this overall framework, semantics can influence a phonological output and phonology can influence a semantic interpretation. In practice, it has been difficult to implement the full framework (but see section 10.2.3.3 for a rough approximation to the interaction between semantics and phonology that was advantageous in a model of surface dyslexia). Here we focus on more limited models in which only one of the pathways involved in reading was extracted and examined in isolation. Specifically, Hinton and Shallice (1991) modeled the pathway from orthography to meaning (large unfilled arrows) and Plaut, McClelland, Seidenberg, and Patterson (1996) modeled the pathway from orthography to phonology (large solid arrows). The number of units and other details in figure 8.4 refer to these two simulations. The units at the end of each pathway were completely or partly interconnected as indicated, with the result that multiple point attractors developed in those groups of units.

8.3.2.1 Modeling the semantic pathway Hinton and Shallice's (1991) network runs vertically through figure 8.4. As we will discuss in chapter 10, they planned to lesion the network to simulate errors made by individuals with deep dyslexia when reading aloud (most frequently semantics-based errors such as *PEACH* → "apricot"). Because the pathway from orthography to semantics appears to play a prominent role in this disorder, Hinton and Shallice isolated it for study. (They assumed that the pathway from semantics to phonology, which is needed to complete the reading-aloud task, functioned with little error and was not crucial in simulating this particular disorder.) In their intact model of this pathway, 28 grapheme units encode a word's orthography (i.e., its spelling or visual form) and send activation through a hidden layer to 68 sememe units. As described below, there are interactive connections within this last layer and between it and a layer of *clean-up units*; aided by attractor dynamics, the semantic layer settles into the pattern corresponding to the word's meaning. There is no further interactivity, though: the connections specified in figure 8.4 from orthography to hidden units and from hidden units to meaning were unidirectional rather than bidirectional in this particular model.

Hinton and Shallice trained the network on a corpus of 40 three- and four-letter words across 1,000 epochs of backpropagation. Each written word (indicated by upper case; e.g., *MUD*) could be given a localist, position-specific encoding using binary grapheme units. Position 1 (linguists call it the *onset*) had 11 consonant units, position 2 (*vowel*) had 5 vowel units, position 3 had 10 consonant units, and position 4 had optional *E* or *K* (positions 3 and 4 are both part of the *coda* which follows the vowel in some but not all syllables of English). The word meanings (indicated by lower case; e.g., *mud*) were limited to five semantic categories (animals, foods, body parts, indoor and outdoor objects) and were represented using semantic roles appropriate to those categories. For each role a group of sememe units was designated (sometimes called an *ensemble* or *assembly* of units), and activation of one unit provided a localist encoding of how that role was filled in a particular word meaning. For example, units 9–15 formed an ensemble for encoding color, with one unit each for white, brown, green, transparent, etc. Representing the meaning of one word involved activating approximately 15 units, which could include multiple units from a given role ensemble (e.g., *lime* has two fillers for the taste role: **sweet** and **strong**).

The challenge for the model is that visually similar words (e.g., *MUD* and *MUG*) need to be mapped on to dissimilar meanings (e.g., role:filler sememes for the meaning *mud* include **hardness:soft** and **location:on-ground** and those for *mug*

include **hardness:hard** and **location:indoors**), whereas visually different words (*MUD* and *BOG*) need to be mapped on to similar meanings. But their network does not treat the spelling of a word as an arbitrary pointer to its meaning, Hinton and Shallice noted, because networks most naturally develop weights that map similar inputs to similar outputs (that is why networks are good at generalization). Extra work is needed to overcome this tendency. Hinton and Shallice therefore employed interactive connections to move the initial semantic patterns towards the more distal desired patterns and to form point attractors at these locations. During learning, changes in the weights on the interactive connections created appropriate basins of attraction in semantic space. Similar-looking words initially produce similar semantic patterns in the resulting network, but if the points in the corresponding semantic space are in different attractor basins, the system should follow a trajectory to the correct meaning in each case. As shown in figure 8.5 for *MUD* and *MUG* (using points in spaces of reduced dimensionality to partly represent patterns in the network), the similarity in spelling tends to result in representations that maintain some proximity through the transformations from orthographic to hidden to semantic layers of the network. But then the interactive dynamics among sememes put the two words on to diverging trajectories (less direct than shown) towards distant attractors. The additional words shown (without their basins of attraction) indicate that attractors for other words in the same semantic category tend to be nearby.

The interactive connections that implemented this dynamic were of two kinds. First, Hinton and Shallice inserted pairwise connections between sememe units within the same semantic role ensemble, which tended to produce varying degrees of lateral inhibition within these groups. Second, they added a set of 60 *clean-up units* which received input from the sememe units and sent activation back to them. These units learned which combinations of sememes were characteristic in the corpus and guided the sememe patterns towards these combinations. For example, when the units for **size:small** and **hardness:hard** and **shape:3D** all are active, **location:indoors** also should be active. The clean-up units will notice that and boost the net input to the **location:indoors** unit over time until it crosses its activation threshold. That is, the co-occurrence of these four units is an attractor, and its dynamics enhance processing of such words as *mug*, *cup*, *can*, *gem*, and *bone*. Moreover, the entire sememe pattern comprising the meaning *mug* is an attractor in part due to this subregularity and in part due to idiosyncratic factors and other subregularities it shares with other words. For example, *mug* shares the cluster **use:for-eating-drinking/location:on-surface/location:indoors/shape:3D** with *can*, *rum*, *pop*, and *lime* (*cup* differs because it is placed on a saucer rather than on a surface and *gem* and *bone* are not used for eating or drinking). In effect, attractor basins for subregularities like these are assembled into word-sized attractor basins, each of which occupies a distinctive region in the full 68-dimensional semantic space. These 40 basins guide the network into the appropriate meaning for each input despite the initial tendency to keep visually similar words too close together.

8.3.2.2 Modeling the phonological pathway Whereas Hinton and Shallice introduced interactivity because they anticipated that attractors in semantic space would help solve a problem, Plaut, McClelland, Seidenberg, and Patterson (1996; hereafter, PMSP) included an interactive network as part of a long-standing commitment to interactive architectures on the part of their research team. They compared it to an otherwise identical feedforward network in part to find out whether interactivity

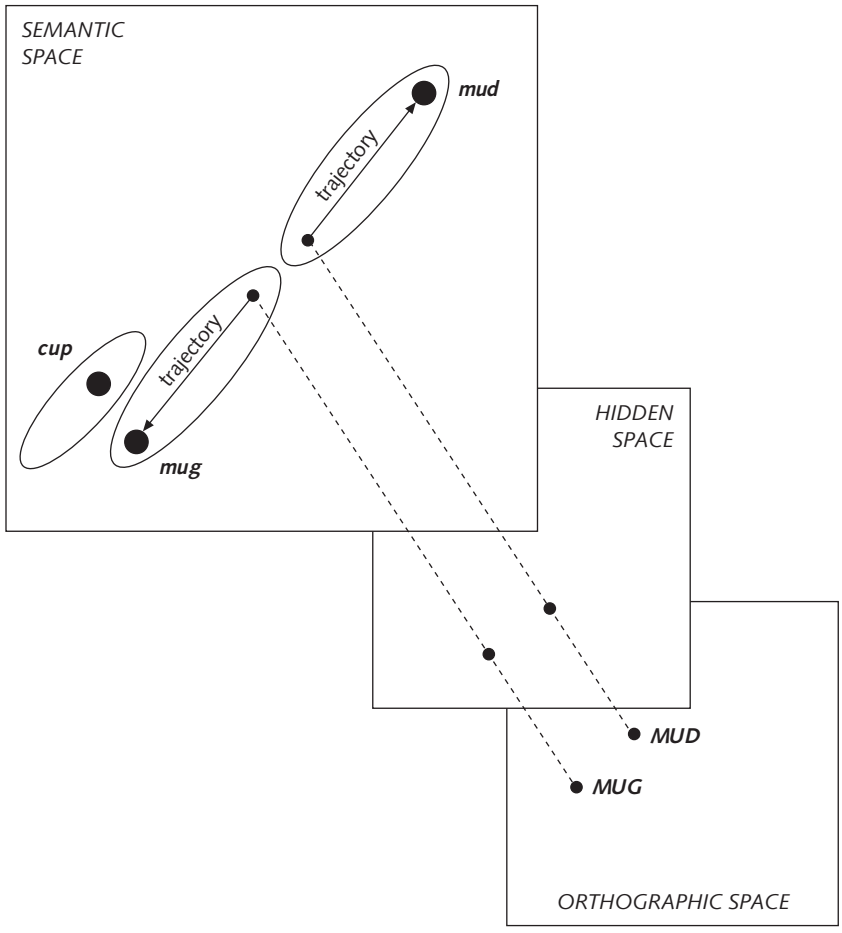

Figure 8.5 A schematic illustration in just two dimensions of how attractor dynamics overcome the tendency for visually similar words like and *MUD* and *MUG* to land near each other in semantic space (Hinton and Shallice, 1991). Each word's meaning corresponds to a point attractor (large dots, far apart for these two words) whose large basin of attraction includes the initial point of entry to the space (small dots, nearby for these two words). Via interactive processing each word follows a trajectory (less direct than shown) to the appropriate point attractor, *mud* or *mug*. Note that *mug* is close to *cup*, not to *mud*.

would be disadvantageous for this particular task, which modeled the pathway leading from orthography directly to phonology. Though ultimately they expected the entire network in figure 8.4 to be involved in the reading-aloud task, this pathway seemed most crucial for reading in general (especially somewhat mechanical or absent-minded reading) and for a form of surface dyslexia in which low-frequency exception words are especially prone to disruption. Like Hinton and Shallice, they planned to simulate one form of dyslexia by lesioning their network (this part of their project is discussed in section 10.2.3).

Reading aloud is a task that is *quasi-regular*; that is, largely systematic, but with exceptions. For example, *MINT*, *HINT*, and *TINT* are regular words but *PINT* is an exception word because its vowel has an atypical pronunciation. The regularities

are sometimes described using rules of grapheme–phoneme correspondence, but one of PMSP's goals was to show that both regular and exceptional pronunciations can be successfully modeled using a network rather than explicitly stated rules. PMSP trained their network to map graphemic into phonemic encodings using a corpus of 2,998 regular and exception words (restricted to onset–vowel–coda monosyllables, e.g., *TH–I–NK*). Like Hinton and Shallice they used a localist, position-specific encoding scheme, but it had to be more elaborate because their words were more complex (they could include multi-letter graphemes like *TH* and also multiple graphemes in each position). There were 105 grapheme units divided into three ensembles: 30 onset units such as *G*, *W*, *WH*, and *TH*; 27 vowel units such as *A*, *I*, *AI*, *OO*, and *OY*; and 48 coda units such as *G*, *X*, *KS*, *TH*, *SS*, and *TCH*. The hidden and phoneme layers were as shown in figure 8.4. The three groups of phoneme units were position-specific for onset, vowel, or coda. More than one unit could be active for a position; for example, to present *THINK* to the network requires activating three onset units (the primary grapheme *TH* and by convention also *T* and *H*), one vowel unit (*I*) and two coda units (*N* and *K*, which are separate graphemes). In addition to the feedforward connections, each phoneme unit sent a lateral connection to every other phoneme unit (including itself) and a feedback connection to each hidden unit; it was interactive processing across these two sets of connections that produced attractors in the network. (Note that the use of position-specific ensembles of units in the input and output layers contrasts with Seidenberg and McClelland's use of Wickelfeature representations in a 1989 model; reasons for changing their encoding system are noted in section 10.2.3.3.)

The network was trained with the *backpropagation through time* procedure (an adaptation of backpropagation for recurrent networks). After 1,900 epochs of training it had learned to pronounce all but 25 of the 2,998 regular and exception words in the corpus. The comparison network, which had the same feedforward connections but neither type of recurrent (interactive) connection, learned much more easily; it made 0 errors after just 300 epochs of training. However, given that interactive networks are more neurally plausible than purely feedforward networks, PMSP thought it important to find out whether an attractor network was able to generalize its training on words so as to attain human-like performance in pronouncing nonwords. It was by no means obvious that this would be the case, since attractor dynamics are supposed to help ensure that a network's response will be one of those already learned (e.g., the pronunciation "think," which is /θink/ in phonemic notation), not only to the inputs on which it was trained (e.g., the written word *THINK*) but also to similar inputs on which it was not trained. (e.g., the nonword *BINK*). As long as the input activates a point within some word's basin of attraction, the interactions between the units during settling should result in the activation of that word. This would seem to preclude the network correctly reading aloud nonword test items; for example, if the initial response to *BINK* fell into the attractor basin for the pronunciation of *THINK* it would be incorrectly pronounced /θink/.

In fact, though, the network performed very well. It was tested on a list of 86 nonwords created by Glushko (1979), in which half were derived from regular words and half from irregular words, and a list of 80 nonwords used by McCann and Besner (1987) for a control condition. Table 8.1 compares the performance of human participants in their studies with that of the interactive (attractor) network as well as the comparable feedforward network. The similarities are impressive, with the same pattern of difficulty in each row and the absolute percentages very close in

Table 8.1 Percentages of regular pronunciations in tests of three sets of nonwords

	Glushko (1979)		*McCann and Besner (1987)*
Reader	*Regular nonwords*	*Exception nonwords*	*Control nonwords*
Humans	93.8	78.3	88.6
Interactive network	93.0	62.8	86.3
Feedforward network	97.7	67.4	82.5

Note: Adapted from Plant et al. (1996), table 3.

two of the three columns. Moreover, error in the networks, used as an index of difficulty, showed the same regularity by frequency interaction as human naming latencies (i.e., infrequent exception words are slower than frequent ones, whereas regular words show little or no frequency effect). A closer analog to naming latency was available for the interactive networks (average time to a criterion of stable responding); it showed the same interaction. (However, Spieler and Balota, 1997, argued that the model should yield good predictions of relative performance by humans on individual items, not just two-way interactions involving broad categories such as high-frequency items. On their analysis, it did not.)

How was an attractor network able to respond appropriately to nonwords? PMSP proposed that the network's primary strategy was to develop, not whole-word attractors, but rather *componential attractors* – one each for the various onset, vowel, and coda clusters that make up words (a cluster contains one or more graphemes or phonemes, such as the *N* and *K* in the coda of *THINK*). That is, the network did not always treat written words as unanalyzed wholes but rather learned the usual pronunciation of each onset, vowel, and coda that recurred across the words of the corpus. It encoded them as "soft" activation-based correspondences between orthographic feature patterns and phonemes rather than "hard" grapheme–phoneme correspondence rules. Learning the regularities in this way produced attractors for particular phoneme clusters in phonemic space that were associated with the appropriate orthographic clusters via additional attractors in hidden-unit space. To pronounce a regular word, in effect, the network found the intersection in each of these spaces of the attractors for its onset, vowel, and coda. The same dynamic could work just as well for pronouncing nonwords composed of novel combinations of familiar, regular phoneme clusters. For example, the intersection of attractors for /b/ and /i/ and /nk/ would yield the correct pronunciation of *BINK* even though the entire pattern *BINK* had never been experienced.

Exception words are more complex, since they present a mixture of regular and irregular correspondences. PMSP suggested that the system takes advantage of whatever regularities do exist within the word but goes part-way to a whole-word approach to handle the more idiosyncratic aspects. In order to support this claim, they created several innovative analyses of how the onsets, vowels, and codas of various types of words were handled by the network. For example, if *MINT* were in the corpus, they could show that the orthographic encoding of the onset *M* would be responsible for the inclusion of /m/ in the network's output, the vowel *I* would be responsible for /i/, and so forth. But for the exception word *PINT*, the onset and

coda in the orthographic representation actually would be more influential than the vowel. These analyses provided a window on what was happening at the hidden layer in the absence of a good way to directly examine the attractors that developed there. Even the indirect analyses were too complex to describe here, but the need for them underscores that part of the unfinished business of connectionism is to attain better tools for understanding activity in large networks. In the next section we return to a simulation by Jeffrey Elman that was discussed at some length in section 6.4. We can now view it as a discrete dynamical system and examine an additional analysis he performed which combined principal components analysis with state space plots to get a direct (though partial) look at hidden layer activity.

8.3.3 Discovering trajectories in SRNs for sentence processing

One way of examining the activity of units in a network is to plot trajectories in a state space. Each dimension represents the activation value of one unit, and a trajectory in this space displays the changing state of the network across time. As we saw above, Beer (1995) made good use of this method to unearth the reasons why his autonomous, coupled, and mixed networks behaved so differently in certain tests. By limiting his analysis to three motor neurons controlling a single leg, he was able to provide three-dimensional displays of the limit cycles that evolved and, for the networks with variable input, characterize the dynamics in terms of instantaneous trajectories and attractors.

In this section we show how Jeffrey Elman (1990, 1991) used trajectories in state space to explore the activity of simple recurrent networks (SRNs). His project is otherwise so different from that of Beer that it provides some sense of how broadly useful the tools of DST can be. One difference is that Beer had an unusually small number of units to examine. For most network models, including that of Elman, some method of collapsing the activity of numerous units into a low-dimensional plot is needed in order to visualize the state space. Another difference is that SRNs have a distinctive design that lies somewhere between interactive networks (whose recurrent connections can exhibit attractor dynamics) and feedforward networks (which have no recurrent connections and cannot develop attractors). They have recurrent connections, but they are used in a special way that enables the network to retain and re-use a (compressed) history of its own sequence of states (it recursively copies states; no attractor dynamics are involved or possible). Changes of state are discrete, in response to input, but the possible changes are constrained by the state history that forms part of the input. In DST terms, SRNs are nonautonomous systems (because they receive input) which change state at discrete time-steps (once per input) and might be viewed as composed of subsystems, two of which are coupled (because the units that store the state history and the hidden units provide input to each other).

As we discussed in section 6.4, Elman trained simple recurrent networks to predict successive words in a corpus of sentences. He was interested not merely in the network's success, but also in understanding how it accomplishes its task. In his 1990 paper he used cluster analysis to show that hidden unit patterns were similar for words with similar privileges of occurrence in sentences; that is, patterns for words in the same syntactic/semantic class, such as *human* or *transitive verb*, were clustered together in the hierarchy extracted by the analysis. This way of examining

hidden-unit activity was necessarily rather coarse-grained: since patterns had to be averaged across contexts to obtain a single pattern for each word, the cluster analysis provided no insight into how the hidden-unit patterns for a given word varied according to current grammatical context. Such variations play a key role in enabling an SRN to predict the next word.

To examine the time dimension rather than averaging it away, a DST framework would suggest plotting the trajectories of the activation patterns on the hidden layer as the network moves from one word to the next in a corpus. For a network with 70 hidden units this would require a trajectory in an activation space of 70 dimensions, which can easily be computed but cannot be displayed on a page nor grasped by mere mortals. Elman (1991) turned to principal components analysis as a method for taming the surfeit of detail. This statistical procedure extracts a set of orthogonal dimensions (principal components) that captures much of the structure in the data and projects the high-dimensional vectors on to this reduced space. If the number of derived dimensions is still larger than desired, a state space plot can be limited to two or three of them (selected because they account for the most variance or are most relevant for a specific analysis). Elman views his networks as having learned constraints on possible trajectories. When a word is presented along with a record of the sentence's trajectory so far (on the input and context units, respectively), the hidden units integrate this information and propagate it to the output units so as to activate one (or more) words that meet the constraints. The plots generated using dimensions from the principal components analysis display the word-by-word trajectory of a sentence through a subspace of the overall hidden-unit activation space.

Elman (1991) used this technique to view some of the dynamics involved in processing sentences generated by a phrase structure grammar (the grammar and the "starting small" technique of training simple sentences first are the same as in Elman, 1993; see section 6.4 for details). Figure 8.6 displays plots in the two-dimensional state space obtained from principal components 1 and 11. This particular hidden-unit subspace happened to be particularly informative about the processing of relative clauses. A separate plot is provided for each of the following sentences. (To minimize other influences on the trajectories, *boy* is the only noun used throughout.)

(a) boy chases boy.
(b) boy chases boy who chases boy.
(c) boy chases boy who chases boy who chases boy.
(d) boy who chases boy chases boy.

Consider sentence (a). The first word (*boy*) is presented on the 26 input units and becomes re-encoded on the 70 hidden units. Weights on the connections leading out from the hidden layer support the network's prediction on the 26 output units (which is not a single word, but rather all words that satisfy the distributional constraints of the corpus: all of the singular verbs, including *chases*, plus the word *who*). When the 70-dimensional activation vector on the hidden layer is projected on to the subspace to give us a partial look at it, in figure 8.6(a), we see it has landed in the lower right corner (at a point high on component 1 and somewhat low on component 11). Now the network is (in effect) wiped clean except that the hidden-layer vector is copied on to the context units and the next word (*chases*) is presented on the input units. This combination produces a hidden-unit vector that projects to a middle left point in the state space. After it has been used to predict the next word

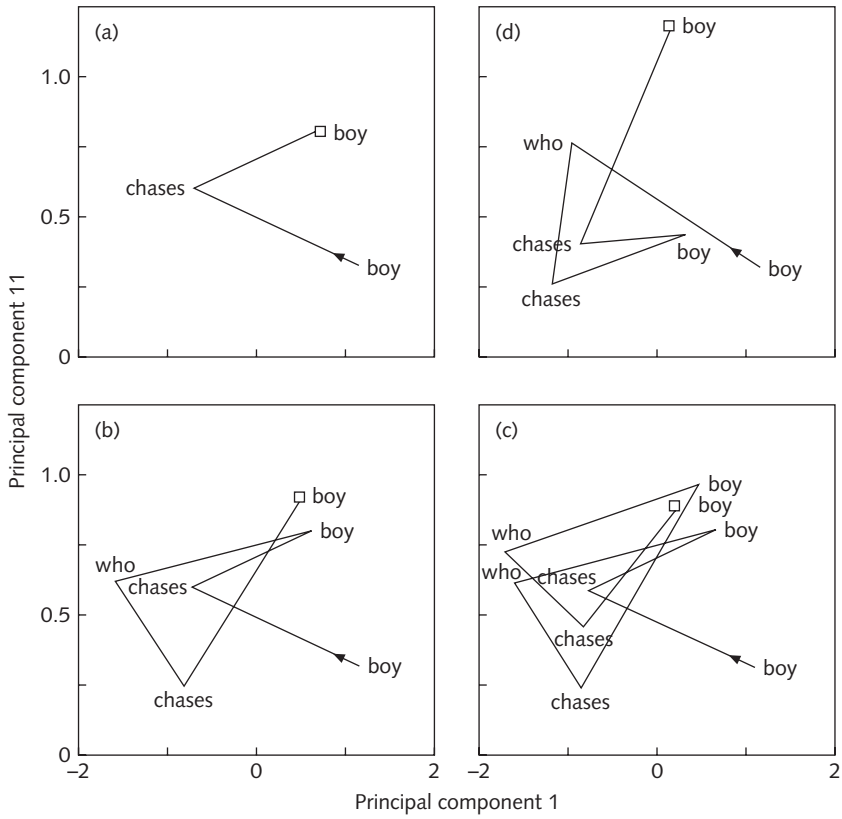

Figure 8.6 Trajectories through activation state space as Elman's (1991) simple recurrent network (SRN) predicts successive words in a corpus generated by a phrase structure grammar. Only the first and eleventh principal components of the hidden unit activations are shown. Sentences (a) through (d) are displayed counter-clockwise to facilitate comparison of (a) to the sentences obtained by adding an object relative clause (b) or subject relative clause (d). Recursion of the object relative clause is shown in (c).

on the output layer, this new hidden-layer vector (which now encodes the history of hidden-layer responses to *boy* and then *chase*) is copied on to the context units and the next word (another token of *boy*) is presented on the input units (it is another token of *boy*, and would have been among the predicted words on the previous step if the network had learned its task well). This combination (*boy* as the next word in a sequence beginning with *boy chases*) produces a hidden-unit vector different enough from the first one that it projects to a different part of the state space (it is similar on component 1 but much higher on component 11 than the first *boy*).

To make this short sentence's trajectory easy to see in figure 8.6(a), the three points in the state space are labeled and joined with lines, with an arrowhead added near the beginning and a square at the end. It is a genuine trajectory, in that each state is constrained by the previous state (e.g., the positioning of *boy* is context-dependent). However, because the state changes are discrete and input-driven, intermediate points are not actually traversed by the network. The lines merely indicate the temporal order in which the system jumps from one point to the next. Nonetheless, it will become evident as we discuss plots (b)–(d) that state-space plots combined

with principal components analysis provide an extremely useful window on the sequential activity of simple recurrent networks.

Figure 8.6(b) shows what happens when an embedded clause is added to the simple sentence (a). The trajectory begins as in 8.6(a) but then reveals (for just components 1 and 11) how the network deals with a relative clause modifying the object. Though *who* is a subject, like the initial *boy*, the network takes note of its relative pronoun category and context by producing a hidden-layer encoding that yields a negative value on component 1 (i.e., the trajectory jumps to the far left of the state space). The embedded *chases boy* subtrajectory is more like that of the main clause: though displaced somewhat, it still has the object *boy* higher and further right than the verb *chases*.

When yet another relative clause is added, figure 8.6(c) shows that a triangular subtrajectory much like that of the first relative clause is produced – but displaced slightly. This small difference in states is not accidental; it reflects the fact that although each clause is linguistically identical (a relative clause modifying an object) their contexts are different. One is preceded by another relative clause; the other is followed by another relative clause. Elman states that the failure of the network to informationally encapsulate each clause contrasts with the way recursion is handled in a formal grammar or computational push-down stack. He calls the network's solution "leaky recursion" (p. 218) and argues that it is actually advantageous to encode the same kind of clause in different contexts a bit differently on the hidden layer, even though they result in the same output behavior (the network produces the same sequence of word predictions twice).

The remaining plot, figure 8.6(d), shows what happens when those same words must be predicted, but in a context with very different sequential dependencies (modifying the subject rather than the object). The fact that the words make up a relative clause is reflected in their now-familiar triangular subtrajectory; the fact that this clause modifies a subject rather than an object is reflected in its very distinctive placement in the space compared to 8.6(b).[3]

What if additional relative clauses are added? Since an increasingly long history must be compressed on to a fixed number of hidden units as they recursively track progress through a sentence, eventually the network's performance degrades. In sentence (d), for example, the first *boy* and the final *chases* must agree in number. The second principal component (not shown in figure 8.6) is especially sensitive to the subject noun's number; here, it captures that *boy* is singular, not plural.[4] The weights leading out from the hidden units know how to use this information to predict a verb that agrees in number (here, *chases* rather than *chase*), and also know how to delay exercising this knowledge when an intervening embedded clause is encountered. With additional embeddings, the hidden-layer encoding becomes too compressed and sends less usable information to the outgoing weights. In a nutshell, networks (like humans) become less dependable at dealing with long-distance dependencies as distance increases.

8.3.4 Dynamical analyses of learning in networks

The state space plots in this chapter have displayed activation spaces for networks, some (such as Elman's) focusing on individual trajectories and others (phase portraits) showing at least some of the attractors in a particular system. State space plots can

also be used to display a network's weight space and the trajectory of weight changes it traverses during learning. Because error reduction plays such a key role in connectionist learning algorithms, generally a dimension is added to the weight dimensions which displays the amount of error associated with each weight state. In chapter 3 we introduced two plots of this type. Figure 3.1 was the simplest possible plot: error on the vertical axis and the range of possible weights for one unit on the horizontal axis. Its curvilinear function indicated how much error was associated with each weight within a range for a hypothetical system. The lowest point on the line is especially important; we referred to it there as a global minimum, but in dynamical terms it is an attractor. (The single local minimum in that figure is also an attractor.)

Learning rules such as the delta rule and backpropagation are gradient descent rules, which means they change weights in a network so as to follow a trajectory to a lower error. The various points in weight space from which a network will settle into a particular minimum constitutes its basin of attraction. When there are multiple attractor basins, as in figure 3.1, they are separated by repellors – points in the weight space from which the system will move away. (A successful learning procedure can escape local minima by perturbing the weights enough to get beyond repellors.) In figure 3.3 we showed how such a weight space representation can be generalized to two weights.[5] Again, the low points are attractors (now on a two-dimensional error surface rather than a line), and gradient descent will lead the network to follow a (frequently meandering) downwards trajectory. The networks used to model human performance generally have high-dimensional weight spaces, but the same general concepts apply.

8.4 Putting Chaos to Work in Networks

8.4.1 Skarda and Freeman's model of the olfactory bulb

Most of the researchers reviewed so far in this chapter are card-carrying connectionists who design and test network models in the usual way and then add DST tools to obtain a better than usual understanding of how the models work. How does the research differ when DST is the starting point, and networks are simply one of the possible mediums in which to explore the potential of DST tools and concepts? Beer's work on network controllers for model insects provides a partial answer, and in this section we will consider work from two additional groups of investigators. One characteristic of DST-driven research that quickly becomes apparent in the original papers is the extent to which mathematical considerations and analyses are front and center. For example, the paper by Beer cannot be meaningfully summarized without talking about limit cycles and attractors. In this section we go further by considering, at a very schematic level, systems that exhibit *chaos* in some phases of their behavior. Another characteristic is that DST researchers enthusiastically put genuinely novel findings on display. These are not easily assimilated by the uninitiated, and in section 8.5 we will discuss arguments for (and against) regarding dynamical approaches to cognition as a new paradigm that supersedes, rather than augments, existing paradigms.

DST researchers also exhibit a bias towards systems with *nonstationary* dynamics – those with an intrinsic ability to keep moving between states rather than getting stuck in an attractor. This contrasts with the typical connectionist view of networks

as input–output devices; from that perspective, using an input to push a system's state into the appropriate point attractor is a pretty interesting way to get an output. The fact that nothing will happen next, unless an external agent zeros the activation values and supplies another input, has not been a high-priority concern. Christine A. Skarda and Walter J. Freeman (1987, p. 172) tried to raise consciousness on this issue by noting that "the neural system does not exhibit behavior that can be modeled with point attractors, except under deep anesthesia or death. Convergence to a point attractor amounts to 'death' for the system." Instead, they view the nervous system as a dynamical system that is constantly in motion, finding different opportunities not only on trajectories within a single phase portrait but also as changes in parameters reshape the phase portraits themselves.

Even the humblest aspect of nervous system functioning is re-construed by these neuroscientists: its background activity is claimed to emerge from deterministic chaos rather than random noise. Despite the drama of the term "chaos," this simply means that the system continuously changes state along a trajectory that appears random but is determined by the equations governing the system and its initial conditions (the values of its variables at t_0). Skarda and Freeman viewed chaos as a way of keeping the overall state space active and ready for more targeted action, in contrast to the usual assumption that background activity is noise that is unrelated to signals and obscures them. (Chaotic systems are famous for their sensitivity to initial conditions – that is, small differences in initial values tend to produce quite dissimilar trajectories – but the particular trajectories are unimportant for Skarda and Freeman's purposes.)

Skarda and Freeman sought to entice their readers towards this perspective by discussing Freeman's (1987) model of the olfactory bulb. The model is a network, but is connectionist only in the broadest sense of that term. Its design was motivated by considerations from DST and neuroscience: each component of the olfactory system (with subsets of excitatory and inhibitory neurons of different cell types treated as separate components) is represented by a second-order nonlinear differential equation, and these components are coupled via excitatory and inhibitory connections into an interactive network. In a painstaking series of studies, Freeman and his earlier collaborators had conditioned animals (typically rabbits) to respond to particular odors. In tracking concomitant electrical activity using EEG recordings, they had found that the olfactory bulb exhibits a pattern of disorderly firing during exhalation and more orderly firing during inhalation. The model exhibits similar alternations. During late exhalation it receives no input and behaves chaotically – engaging in "restless, but bounded, activity" (p. 165). During inhalation an odor is supplied, which usually sends the system from chaos into the basin of one of several limit cycle attractors that rather suddenly make an appearance. Each attractor is a previously learned response to a particular odor (except that one corresponds to a no-odor control condition); hence, the system can be said to have recognized an odor when the system lands in the appropriate attractor.

Note that the recognition response is not static. First, when the trajectory is pulled into a limit cycle attractor it cycles through multiple states (vs. a point attractor's single state). Second, once that cyclic attractor has done its job, other aspects of the system's dynamics (referred to as nonstationary) provide routes into other activity. One way out is that the relatively organized phase portrait for inhalation includes the low-energy *chaotic well* to which the system will retreat if a novel

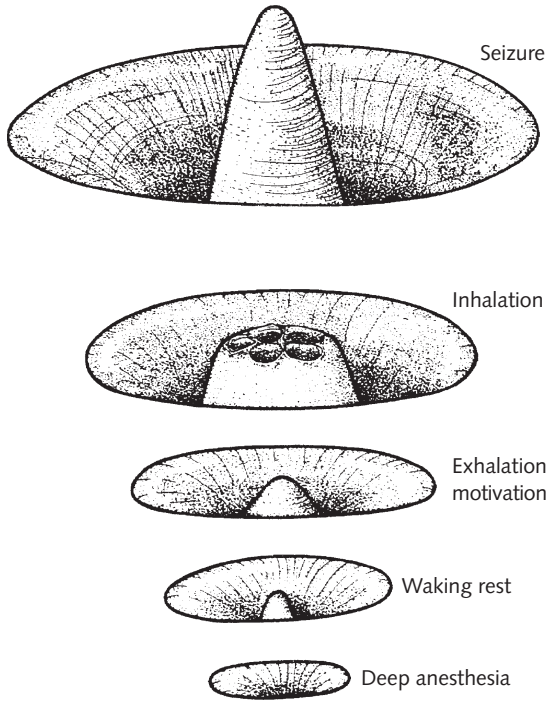

Figure 8.7 Hypothetical phase portraits for the olfactory system Reprinted with permission from Freeman (1987, p. 146), who emphasized the inhalation and exhalation phases in rabbits when motivated by presentation of previously-conditioned odors.

odor is supplied (and from which a new limit cycle can form across repeated presentations). The more usual way out is that the phase portrait itself is continuously changing. During exhalation the limit cycle attractors disappear and the system finds new opportunities in chaos. Skarda and Freeman (p. 168) "conjecture that chaotic activity provides a way of exercising neurons." On the next inhalation, chaos plays a more task-relevant role, allowing "rapid and unbiased access to every limit cycle attractor . . . so that the entire repertoire of learned discriminanda is available to the animal at all times for instantaneous access. There is no search through a memory store."

For Skarda and Freeman, then, odor recognition is achieved when the olfactory system alternates between relatively free-ranging chaotic behavior (exhalation) and odor-specific cyclic behavior (inhalation). The same system is capable of reaching extremes of anesthesia and seizure, as shown by the hypothetical "snapshots" of some of its possible phases in figure 8.7. In each phase portrait the two primary dimensions represent the overall activity of two subsets of neurons (excitatory and inhibitory). The vertical dimension represents the amount of energy when a point is active. During anesthesia a point attractor produces a temporary "death" (very low-energy state). A point repellor replaces it as the system moves to a waking rest. A chaotic well (the circular trench, whose base is a chaotic attractor) develops and deepens as the system becomes more motivated and alternates between exhalation and inhalation. The limit cycles are represented in the center of the inhalation

portrait, and become latent as the system relaxes into exhalation or (exceptionally) gets repelled into the degenerate, low-dimensional chaos of seizure.

The system's ability to temporarily lose and regain its limit cycles via its own nonstationary dynamics is an intriguing solution to the problem of how to stop responding to one input and begin responding to another. To understand what moves Freeman's model between inhalation and exhalation, recall the logistic equation (equation (1)). In figure 8.2 it was seen to exhibit chaotic dynamics in a region with values of A beyond 3.6. But within this region there existed values of A for which the dynamics again became periodic. This suggests the possibility of a system moving from chaotic regimes to temporarily stable ones (and back to chaotic ones) through small changes in parameter values – an ability that would be extremely useful for a nervous system. The equations describing the functioning of the olfactory bulb are more complex, but they show this same characteristic. Importantly, changes in parameter values are not arbitrary (e.g., some reflect the influence of systems to which the olfactory bulb is connected). As Skarda and Freeman (p. 167) note in discussing the overall states captured in figure 8.7: "[T]he olfactory system and its corresponding model have a hierarchy of states. The basic neural dynamics and the equations are the same in all states but, depending on various neural conditions and model parameters, the systems behave differently. . . . Both systems display the capacity for abrupt, dramatic global jumps from one state to another. These are the bifurcations."

8.4.2 Shifting interpretations of ambiguous displays

In Freeman's model, changes in parameter values (usually due to the activities of related systems) are responsible for the system's transitions; the role of chaos is affected by, but does not effect, those changes. Chaos has been argued to play a much more prominent role in the spontaneous shifts of attention that people report when they look at such well-known ambiguous figures as the duck/rabbit, young/old woman, and the Necker cube. For example, Cees van Leeuwen and his collaborators (van Leeuwen, Steyvers, and Nooter, 1997) proposed a DST-based network model of people's shifting perceptions of the ambiguous display at the center of figure 8.8. To the left and right of it are unambiguous displays that produce relatively stable percepts. The same network model that can simulate ordinary percepts like these becomes destabilized in the presence of the ambiguous display, repeatedly switching between column and row interpretations of its organization. In their words (p. 321): "The noisy processes which help construct the pattern will revolt against it, once it becomes established." In achieving switching behavior, they made an important advance beyond the first network model of perceiving ambiguous figures, in which the network settled to one of two point attractors (chose one of the possible interpretations of a Necker cube) but then stopped (Rumelhart, Smolensky, McClelland, and Hinton, 1986).

The work of van Leeuwen et al. expanded upon three related strands of research. First, Skarda and Freeman (see section 8.4.1) had the insight that chaos may be fundamental to perception and constructed the first network model in which chaotic and stable behavior alternate.

Second, J. A. Scott Kelso showed that coupled systems with nonlinear dynamics could switch between *metastable* (not quite stable) states at irregular intervals, mim-

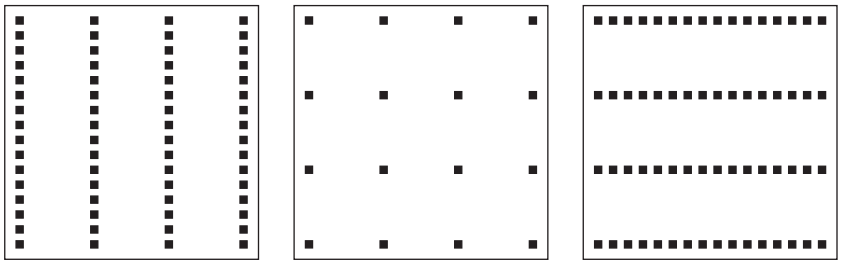

Figure 8.8 Stimuli used in van Leeuwen, Steyvers, and Nooter (1997). If the gestalt principle of symmetry is used to group items, the black squares in the left display will be grouped vertically and those in the right display horizontally. Those in the center display, however, will be ambiguous and subjects may alternate between grouping them vertically and grouping them horizontally.

icking the switching intervals of people asked to press a button each time their interpretation of a Necker cube reversed. On Kelso's account (see Kelso, 1995), each interpretation is attractive to the system but is not quite an attractor. The system therefore exhibits *intermittency*, alternating between metastable states and chaotic bursts in which the system breaks free and moves erratically through state space. In the words of Kelso (1995, p. 200): "[T]he system dwells for varying times near attractive states where it can switch flexibly and quickly. Faced with ambiguous stimuli, the brain will be revealed as a twinkling metastable system living on the brink of instability."[6] Kelso also emphasized that the metastable states (and the flow of patterns in the brain more generally) are an outcome of *self-organization*. Patterns are generated by a large number of components interacting nonlinearly, with no supervisors or agents needed. Although Kelso gave considerable attention to the dynamics resulting when two or more self-organized systems become coupled, he characterized the systems themselves in terms of equations with a small number of variables and parameters (see section 8.5.2).

This leads to the third strand contributing to van Leeuwen et al.'s work. One way to understand how the systems became self-organized in the first place is to build a network model whose units are low-level components of the perceptual system. Kunihiko Kaneko (1990) explored the stability characteristics of a type of network called a *coupled map lattice* (CML). A *lattice* is a sparsely connected network in which the couplings (connections) can be viewed as topologically arranged such that *neighbors* are coupled and other units are not; for example, the Kohonen feature maps used for DISCERN's lexical and episodic memories in chapter 7 are lattices. A *map* is a type of function in which values are iteratively determined in discrete time; for example, the logistic equation (equation (3) in section 8.2.2) is a map and was used by Kaneko to obtain the value of each unit in a lattice at each time-step. This choice yields coupled nonlinear units which move between values within a range (*oscillate*) in discrete time either periodically (e.g., alternating between the two most extreme values) or chaotically (yielding a quasi-random sequence of values within the range). Such a network can exhibit different kinds of behavior depending on what values have been assigned to certain control parameters; among the possibilities are synchrony[7] across periodic or chaotic units (i.e., all units in a cluster have the same sequence of activation values, even if that sequence is chaotic) and chaotic behavior across chaotic units (*chaoto-chaotic emergence*).

Van Leeuwen et al. proposed that CML networks could be harnessed as lower-level, self-organizing mechanisms for achieving intermittency in models of perception. Given the ambiguous display in the center of figure 8.8 as input, an appropriate CML quickly comes to exhibit one pattern of synchronized activity for the columnar interpretation (as on the left) and a different one for the row interpretation (as on the right). These patterns are metastable states of the network, so they can suddenly reorganize (shifting the synchronization from rows to columns or vice versa, or under some conditions from global to more localized synchronization or vice versa).

Preliminary to studying a full-scale CML model of perceptual organization, van Leeuwen et al. first examined the simplest network in which synchronization can be achieved – a network of just two units. Since individual neurons probably are linear, each unit is best thought of as a micro-ensemble of excitatory (pyramidal) and inhibitory (stellate) neurons. The net input to each unit is calculated according to the following equation, in which a_x represents the value of unit x (our notation, reflecting that it is roughly comparable to an activation value in a traditional connectionist network) and C represents a coupling parameter (comparable to a connection weight) that determines how much each unit is affected by its own value versus that of the other unit:

$$netinput_x = Ca_y + (1 - C)a_x \tag{4}$$

To obtain the value of unit x at each discrete time, they incorporated the net input calculated by equation (4) within the logistic equation (see equation (3)):

$$a_{x,t+1} = A\ netinput_{x,t}\ (1 - netinput_{x,t}) \tag{5}$$

The net input and value of unit y were obtained in the same way. As shown earlier (in figure 8.2 for equation (3)), the values of a unit approach a point attractor at lower values of A and periodic attractors at intermediate values, but behave chaotically for most values above 3.6. This is how each unit on its own would behave. Because the units are coupled, however, the additional parameter (C) can alter these outcomes. The overall behavior of the two-unit network depends on the values of both A and C.

In this miniature network it is easy to measure synchronization: the two units are synchronized when the difference between their activation values at each time-step is zero. Generally they do not start out synchronized, but van Leeuwen et al. demonstrated that the size of the difference will decrease monotonically to zero when

$$^1/_2(1 - 1/A) < C < {}^1/_2(1 + 1/A).$$

That is, for appropriate values of C relative to A, after a transition period the two units will exhibit the same sequence of values – a sequence which itself (depending on A) may be chaotic. It is outside this range of guaranteed synchrony that things get interesting. The size of the difference may be a constant or may vary periodically, quasi-periodically, or even chaotically. Most relevant for a psychological model of perception, the size of the difference may vary intermittently: alternating between zero (a semi-stable state of synchronization) and a chaotic sequence of values (wandering through state space until the difference rests temporarily at zero again).

Van Leeuwen et al. then extended their analysis to the larger CML networks appropriate for perceptual tasks. For example, each unit in an array of 50×50 units may be sparsely coupled to just four other units – its neighbors in the array. (Note that van Leeuwen et al. usually coupled each unit to its corresponding unit in

additional arrays as well, but one array is enough to get the key results.) They began by simply generalizing equations (4) and (5) to apply to more than two units, and found that small values of C relative to A tended to generate relatively small clusters within which units may synchronize their activity. What was needed to simulate shifting perceptions of the grid in figure 8.8, however, was a very specific synchronization in which the clusters were specialized to its rows and columns; for more stable perception of a large "X" pattern, two diagonal clusters (at different orientations) were needed. To obtain networks that could adapt to the input patterns of interest, they modified the way in which C and A were used. First, A became a variable controlled by input rather than a fixed parameter. Relatively high activity in the receptive field of a unit was realized by lowering the value of A for that unit. (This seems backwards, but lower values of A would tend to drive the unit to a level of chaotic activity at which it is more likely to synchronize with other units: *weak chaos*.) Second, the coupling parameter C was replaced by adaptive weights on each local connection plus a global parameter C_{max}, which scales those weights so as to produce a bias towards stability or instability (depending on the value of A). When the activation sequences of two units begin to synchronize the weight between them is increased; this favors greater synchronization in the succeeding time-steps. Thus, synchronization that initially just happened to occur between two chaotic sequences gets grabbed and used by the system to move towards more structured activity. In a sense, the weights serve as a short-term memory of recent synchronization that helps to reinstate that synchronization. With this occurring across multiple pairs of units simultaneously, the system can advance towards larger clusters within which all units are synchronized (e.g., a cluster specialized to the third column) and leave behind its chaotic behavior in favor of one of the metastable states (e.g., seeing the grid as organized in columns).

Using this adaptive CML, van Leeuwen et al. were able to simulate the behavior of a perceiver switching between metastable synchronizations when the input represents an ambiguous figure, but also attaining stably synchronized clusters when the input is an unambiguous figure. What is important is that the system has the intrinsic capacity to achieve percepts via synchronization but also the flexibility to change to a different percept via desynchronization. Ambiguous figures are useful for researchers because they can be counted upon to put the system into irregular swings between synchronization and desynchronization. This case would be only a curiosity, though, if it did not point the way to the system's overall design and capacities. That the same system can handle unambiguous figures is an initial demonstration of the generality of the design. Recently this research group has provided further demonstrations. Within perception, they have shown that CMLs can provide an especially efficient solution to Grossberg's boundary contour problem (van Leeuwen, Verver, and Brinkers, 2000). In a much bolder move, they have proposed to extend the timescale at which coupled maps are considered to operate downwards as far as iconic memory and upwards to long-term memory (van Leeuwen and Raffone, 2001). In this unified view of perception and memory, representations at a variety of timescales may be realized and maintained by the chaotic dynamics of coupled oscillators. A bold claim elicits tough questioning. Much work would be needed to show that this mechanism is adequate to account for a broad variety of perceptual and memory phenomena; and even if it works as a base, additional mechanisms may be needed as well. There is also the question of whether this is the way the brain actually does the job.

The questions raised by chaos-inspired dynamical models, exemplified here by those of Skarda and Freeman and van Leeuwen's group, will not be answered quickly. The results to date suggest that incorporating a DST perspective in the very design of networks yields distinctive properties which may be used to advantage in modeling and may also change the way cognitive scientists think about perception and cognition. However, the approach is too new to have moved much beyond individual models; for example, Freeman and van Leeuwen make very different uses of chaos. As researchers gain more experience with DST-driven network design, principles and practices will emerge and the approach will have its best chance of gaining increased visibility and impact within cognitive science. Will enough researchers be sufficiently enticed to re-situate their own work within an unfamiliar theoretical territory, bringing about a Kuhnian paradigm shift? If so, would the impact of connectionism be seen retrospectively as merely transitional? We now leave specific models behind and return to philosophical inquiry into implications.

8.5 Is Dynamicism a Competitor to Connectionism?

8.5.1 Van Gelder and Port's critique of classic connectionism

Connectionist networks are clearly complex systems and, as we have seen, certain connectionists have found DST tools to be extremely useful in analyzing the behavior of their networks and developing new kinds of networks. As exemplified in the quotation from van Gelder and Port at the beginning of the chapter, though, for some theorists the emergence of dynamical approaches offers not just a set of tools to be utilized within existing paradigms but an actual Kuhnian revolution in cognitive science. On this view, connectionism was not the real revolution:

> [C]onnectionism should not be thought of as constituting an alternative to the computational research paradigm in cognitive science. The reason is that there is a much deeper fault line running between the computational approach and the dynamical approach. In our opinion, connectionists have often been attempting, unwittingly and unsuccessfully, to straddle this line: to use dynamical machinery to implement ideas about the nature of cognitive processes which owe more to computationalism. From the perspective of a genuinely dynamical conception of cognition, classic PDP-style connectionism (as contained in, for example, the well-known volumes [of] Rumelhart and McClelland, 1986, and McClelland and Rumelhart, 1986) is little more than an ill-fated attempt to find a halfway house between the two worldviews. (van Gelder and Port, 1995, pp. 33–4)

In support of this claim, they asserted that the classic connectionism that used networks (especially feedforward networks) as "sophisticated devices for mapping static inputs into static outputs" (p. 32) is disappearing as it splits into two distinct streams. Researchers in the relatively computational stream design networks that straightforwardly implement computational mechanisms or have hybrid architectures. Researchers in the relatively dynamical stream design networks like those discussed in the current chapter and give at least some attention to their dynamics. Van Gelder and Port allow that (p. 34): "Connectionist researchers who take the latter path are, of course, welcome participants in the dynamical approach" but also point to ways they differ from nonconnectionist dynamicists – especially those

dynamicists taking the quantitative approach that van Gelder and Port regard as a standard or prototype.

One difference that tends to keep connectionists at the periphery of dynamical modeling is the type of formal model employed: massive networks targeting the "microstructure of cognition" versus equations with collective variables targeting its macrostructure. Moreover, van Gelder and Port seem concerned that connectionists are still carrying baggage from the classical computational approach that slows their progress along the road from the halfway house. This concern is elucidated in several papers in which van Gelder laid out his view of the differences between the computational and dynamical worldviews (van Gelder, 1995, 1998, 1999). In what follows we will first briefly contrast the two styles of modeling and then grapple with whether the computational baggage is a help or a hindrance by discussing issues of explanation and representation.[8]

8.5.2 Two styles of modeling

Dynamicists generally strive for compact models in which one or more (preferably differential) equations capture the overall behavior of a system in terms of a very small number of variables and parameters. Connectionists (even those taking a dynamical approach) produce models which have about the same number of equations but apply them repeatedly across the ranges of variables. Iteration of this kind yields as many activation values as there are units and as many weights as there are connections at each time-step. These differences in the type of formal model employed reflect differences in goals and desired grain-size of one's account.

For example, we have already seen that in van Leeuwen et al.'s CML account of the perception of ambiguous figures (a network-like dynamical model) a large number of coupled oscillators are governed by the same equations but do not behave identically. They interact to produce conflicting, metastable interpretations of a stimulus. Kelso's (1995) standard dynamical account also achieves metastability (pp. 218–23), but with respect to the value of a single collective variable (ϕ) that measures the synchrony of just two coupled oscillators. The two primary interpretations of the ambiguous figure are indexed by these values rather than simulated. It is a new application of Kelso's signature model of a surprisingly salient task: finger twiddling. People are asked to move both index fingers up and down either together (in-phase) or in opposition to one another (antiphase) in synchrony with a metronome. As the speed of the metronome increases it becomes impossible to maintain the antiphase movement, and subjects involuntarily switch to in-phase twiddling. This can be dynamically understood as a transition from a landscape with stable attractors for both types of movement to one with a stable attractor only for in-phase movement. Intermediate values of *relative phase* (φ) may appear during the transition (e.g., one finger just a bit ahead of the other in their up–down cycles). The attractor landscapes (V) within which in-phase, antiphase or intermediate relative phases occur can be obtained from equation (6) by providing appropriate values of the parameters:

$$V = -\varphi\delta\omega - a\cos\varphi - b\cos 2\varphi \tag{6}$$

To genuinely understand this equation and its ramifications, you must read Kelso's book. The main point here is that an equation with just a few parameters can give an account of the behavior of two coupled oscillators (here, fingers). The difference

between the fingers' spontaneous frequency and the metronome's frequency is reflected in $\delta\omega$. The a and b parameters reflect (indirectly) the oscillation frequency (how fast the fingers are moving), and a/b is a crucial coupling ratio (when it is small, oscillation is fast and in-phase movement is the only attractor). Certain combinations of parameter values produce intermittency (the system fluctuates chaotically between the two kinds of movement, which now are semi-stable rather than stable). Change the realization of the equation from finger phases to competing interpretations of an ambiguous figure (by changing the interpretation of each collective variable), and *voilà* – equation (6) models a perceptual phenomenon (the distribution of switching times) at an abstract level.

Van Leeuwen et al.'s CML model is different in part because it does two jobs. Unlike Kelso's model, it simulates the percepts themselves – its units actually organize themselves into synchronized columns or rows to simulate the two interpretations of the ambiguous display. Like Kelso's model, though, it also models the more global perceptual phenomenon of semi-stable interpretation by repeatedly but irregularly switching between these percepts.

This difference in style of modeling has other consequences. Certain concepts that are part of the "computational baggage" (our metaphor) apply much more naturally to dynamical network models than to standard dynamical models. In the ambiguous figures task, van Leeuwen et al.'s explanation comes in the form of a *mechanistic model*, within which the metastable patterns can reasonably be regarded as two alternative *representations* of the stimulus (albeit distributed rather than classical). Kelso's compact system was not designed to do these jobs; it models the fluctuation between interpretations of the stimulus array but not the interpretations themselves. In taking a closer look at these computational concepts and at the dynamical alternatives, we will find that each can play a different but useful role in exploring dynamical network models.

8.5.3 Mechanistic versus covering-law explanations

The notions of mechanistic model and representation that we find useful in thinking about dynamical network models are rooted in stronger, classical notions: homuncularity and symbolic representation. Van Gelder (1995, p. 351) had the classical versions in mind when he characterized the computational approach in terms of "a mutually interdependent cluster" of properties: "representation, computation, sequential and cyclic operation, and homuncularity." A computer program with subroutines is a prototype that gives a good, quick sense of what he means. Computation involves discrete operations that manipulate representations; they apply sequentially (not in parallel); and sometimes a particular sequence of such operations will apply iteratively or recursively (cyclic operation – here a discrete notion that is not to be confused with oscillation or limit cycles in a dynamical model). When combined with these other properties, representations are sequences of manipulable elements that usually also have meaning (are symbolic) – this special case is the classical notion of representation. Before discussing representation further, we will take a look at computational versus dynamical approaches to explanation and the special case of homuncularity.

The homuncularity property derives from Daniel Dennett's (1977) characterization of the components in a mechanistic model of the mind as homunculi. By this

metaphor – whimsical but making a serious claim – Dennett saw the mind as a committee of little agents, each with its own specialized subtask (e.g., discrimination, memory, evaluation), who pass messages (representations) to one another to perform the overall task. Each little agent itself can be analyzed as a committee of somewhat more specialized, less clever agents; and so forth until the lowest-level agents perform primitive calculations such as picking the larger of two numbers. Put less colorfully, such a system has a hierarchy of components, each of which performs its subtask by taking representations as input, operating on them, and sending the outputs to other components.

A mechanistic model relaxes this characterization, such that in some cases the components may function and interact continuously rather than discretely and their interactions may be better characterized as causal or information-bearing than as classically representational. One way in which scientists construct such a model is to first decompose a task into subtasks and then try to recompose it by specifying the component performing each subtask and its interactions with other components (see Bechtel and Richardson, 1993). The model is taken to provide an explanation of the system's performance of the task. In principle the decomposition could continue down to primitives (cognitive models are often assumed to "bottom out" in neurobiology), but in practice modelers usually limit themselves to going down one or two levels. When the components of a model like this can be localized in the system being modeled (e.g., identifying edge-detectors in the primary visual cortex), Bechtel and Richardson (1993) call them *complex systems* to distinguish them from *simple systems* (for which components have not been identified) and from *integrated systems* (in which the components interact by feedback loops or other reciprocal connections – the most difficult kind of model and usually not achieved until a research program reaches maturity).

Who needs mechanistic explanation? Most explanations in the mainstream of biology tend to be of this genre. Symbolic models in cognitive science generally qualify – they served as Dennett's paradigm case of homuncularity and van Gelder's prime target. (However, competence models such as Chomskian grammars have a componential structure that may not map cleanly on to the processing system modeled). Standard dynamical models are generally claimed to bypass the mechanistic style of explanation. It is network models for which the question gets most interesting, but to show why, we first need to ask: what are dynamical modelers doing if not providing mechanistic explanations?

One possibility is that dynamicists merely *describe* a system by identifying regularities in its behavior. But van Gelder (1998, p. 625) rejected this suggestion, noting that dynamical accounts of cognition are no different in form than dynamical accounts of physical phenomena such as planetary motion. Since the latter count as explanatory, so should the former. Indeed, his point is well taken; the contrast is not between explanation and description, but rather between two forms of explanation. The logical positivists identified a form of explanation, deductive-nomological or covering-law explanation, which fits the dynamicist case. In a covering-law explanation, a phenomenon is explained by showing how a description of it can be derived from a set of laws and initial conditions. The dynamical equations provide the laws for such covering-law explanations, and by supplying initial conditions (values of variables and parameters), one can predict and explain subsequent states of the system. That is the appropriate kind of explanation for van Gelder, since he proposes to regard cognition solely as a dynamical system – one that changes states in time as expressed in equations.

Now we can consider networks. The simplest thing to say is that the type of explanation attributed to them can vary with the design of the network and with the designer's leanings between classical connectionism and dynamicism. Much more interesting is the prospect that they may lead us to new kinds of explanation that combine and extend current options. Van Gelder (1995, p. 374) projected that combining ingredients of the two worldviews in connectionism "may well turn out to be an unstable mixture" and refrained from doing so; we prefer to focus on the constructive synthesis that may emerge from the energy put into such a project. (Clark, 1997b, was similarly optimistic about combining these types of explanation and provided a thoughtful discussion of explanation more generally.)

Here are some cases. Sejnowski and Rosenberg's (1986) NETtalk network can serve as an exemplar of a feedforward connectionist design, which in this case provides a mechanistic explanation of reading aloud (see section 3.2.2.3 for more details). At the first level down, it is easy to identify components (the input, hidden, and output layers) and to state they are connected by two full sets of feedforward connections. Below that there are two more levels of decomposition on the input layer (seven sets of letter units) and one more level on the output layer (one set of articulatory feature units). It is distinctive to connectionist networks that the hidden layer affords only a functional decomposition: cluster analysis reveals functional components for various grapheme–phoneme regularities, vowels vs. consonants, etc. The connectivity of the network is also highly distributed, unlike classical systems. The designers built the network as a mechanistic model, but cannot give a complete mechanistic analysis of the microfeatures and microactivities that result from its adaptive weight changes during learning. However, the incomplete mechanistic explanation is the best available; dynamical analysis has little to offer towards understanding feedforward networks.

When interactivity is added to layered networks of this type via recurrent connections, complex activity extended across time and connections becomes very important. This aspect of a network model benefits from dynamical analysis. We suggest that a connectionist dynamical approach offers the opportunity to embrace both types of explanation and use them to serve complementary purposes (see Bechtel, 1998). Most simply, the dynamical analysis can offer covering laws that characterize overall patterns of change in a system (generally in terms of aggregate or external variables), and the mechanistic one can show how those changes are effected. The connectionist researcher built the network, knows its components and how they are connected, and can use this knowledge to "go behind the scenes" and provide a (partial) mechanistic account of the phenomena captured in the dynamical covering-law explanation of the system. Dynamical tools may then be used not only to characterize the complete system, but also the interaction of the components. For example, in building the CML model of ambiguous figure perception, van Leeuwen et al. (1997) knew from Kelso's work that two coupled oscillators could produce the right kind of semi-stable behavior. They wanted to build a system at a much finer grain to model the microstructure and microactivity underlying the percepts themselves in addition to the overall semi-stable behavior. Their novel solution (using a large number of coupled oscillators as components) was informed by the prior dynamical model but extended it significantly by treating component parts as dynamically interacting oscillators. The resulting CML model is best described using a combination of dynamical and mechanistic analysis.

In another example from this chapter, a mechanistic analysis gets even further with Skarda and Freeman's (1987) model of the olfactory bulb. Built at an intermediate

grain-size, the "units" were just 16 large nervous system components that were less homogeneous than the units of a connectionist network; for example, all of the excitatory (pyramidal) cells of the anterior olfactory nucleus were represented collectively in one component. However, their interactive connections produced complex behavior in the network as a whole. Dynamical concepts and analysis were crucial for understanding the chaotic and limit cycle behavior that emerged. Another example: Beer's autonomous controller network for insects' tripod gait has six replications of the same five units (a componential organization at this level). Three of the units controlling each leg have well-defined subtasks (each clearly controls its own specialized motor effector) but are dynamical in their quantitative states and interconnected activity (including influence from the other two units – the interneurons). Each leg's activity is best understood in terms of a limit cycle, but the state space is defined with respect to components of the network.

The ability of these two perspectives to complement each other becomes even more evident when dynamical analysis is applied not only to the whole network but also to some of its components. This can work even in a network with thoroughly homogeneous units within each layer. An interesting case covered above is that of Elman (1991), who knew the gross architecture of his simple recurrent network (SRN) for predicting words in sentences and could give a rough mechanistic account based on his own design work. However, he also used principal components analysis (a tool from yet another tradition, multivariate statistical analysis) to identify functional components of the system on the hidden layer and used DST tools to discover exactly what task those components were accomplishing. For example, plotting trajectories through the state space of components 1 and 11 revealed the phenomenon that relative clauses were being wrapped corkscrew-like into the space.

It is not known to what extent an orchestration of methods like this might yield insight into large networks with more complex interactions among units. Michael Wheeler (1998) considered the difficulty of explaining the activity of networks exhibiting what Clark (1997a, pp. 163–6) called "continuous reciprocal causation." An example is Beer's nonautonomous control network, in which sensor input completes a loop between the controller and the environment, but Wheeler was most concerned about larger, more homogeneous networks and especially those in evolutionary robotics (see sections 9.4 and 9.5) whose organization is minimally constrained by human preconceptions of design. He thought the explanatory stance must become more holistic as the amount of continuous reciprocal causation increases, shifting away from modular (mechanistic) explanation and towards system dynamics. "The justification for this claim is that the sheer number and complexity of the causal interactions in operation in such systems force the grain at which useful explanations are found to become coarser" (Wheeler, in press, p. 16). Clark (1997a, p. 175) had a similar concern but argued for proceeding more optimistically, "adding new tools to cognitive science's tool kit, refining and reconfiguring but not abandoning those we already possess. After all, if *the brain* were so simple that a single approach could unlock its secrets, *we* would be so simple that we couldn't do the job!"[9]

In accord with the more optimistic view of Clark, we look forward to seeing how much headway can be made, even with highly interactive networks, when mechanistic and dynamical explanation are combined and extended. However, it is so early in dynamical network research that this suggestion of complementary, even integrated, use of two kinds of explanation must be tentative. Those who wish to use an

exclusively dynamical approach to cognition may find their own way to go behind the scenes, in some way parsing the system and building a hierarchy of dynamical models that cover a variety of grain-sizes from the overall phenomenon of interest on down. In his 1995 book, Kelso sketched such a vision in chapter 2 and considered modeling at neuronal levels in chapter 8. He emphasized, though, that his equations give a different parsing than does a mechanistic analysis of the biological systems that realize them. For now, networks of homogeneous units and connections, each with its own application of the equations governing the system, do a modeling job that is hard to replace by purely dynamical analyses. Can we do better than simply performing each kind of analysis separately? It would be most interesting if attempts to grapple with these problems led to new notions of explanation within the philosophy of science in addition to better models.

8.5.4 Representations: Who needs them?

We turn now to a second area of discrepancy between connectionist and dynamical approaches that places those who seek to combine them at the periphery of dynamicism (but perhaps at the leading edge of the future of cognitive modeling): the role of representations. Disagreements about representation abound. It is least troublesome within the symbolic approach, where the notion that cognition involves operations upon representations (construed as structured sequences of symbols) is central. At the other end of the road, among those dynamicists who attend to the concept of representation at all, it tends to be either denied or radically redefined. Between these extremes are the connectionists, who see part of their mission as (less radically) redefining representation. Activation patterns across the layers of a network are commonly (though not universally) regarded as "subsymbolic" (Smolensky, 1988) – a departure from symbols in their fine grain, their status as numerical vectors, and the kinds of activity that generate them (parallel processing, and for interactive networks, settling into attractors or other kinds of change across a nontrivial temporal dimension). However, van Gelder and Port (1995) worried that connectionists taking this view (especially those working with feedforward networks such as NETtalk) have insufficiently shed notions of representation rooted in the symbolic approach. On this view, an unchanging composition of subsymbols is in danger of being treated as a static symbol.

Dynamical reappraisals of representation were considered as part of an argument for the *dynamical hypothesis* by van Gelder (1998; quotes from p. 622). His starting point was to unequivocally dispense with "static configurations of symbol tokens" – a core commitment of what he calls the computational view (some of the commentators on this BBS paper, seeing computation as broader, would prefer a different term). The main alternative he noted is that dynamicists "find their representations among the kinds of entities that figure in DST, including parameter settings, system states, attractors, trajectories, or even aspects of bifurcation structures" and eventually "even more exotic forms." (See also van Gelder and Port, 1995, p. 12.) That he felt no urgency to pare down this mixed bag of possibilities reflects the fact that few if any dynamicists view representation as a core concern. Indeed, van Gelder also noted (refraining from endorsement or disapproval) the more radical view that dynamicists can develop models of cognition that "sidestep representation altogether."[10] He cited the work by Beer and by Skarda and Freeman (discussed above in sections 8.3.1 and

8.4.1) as exemplifying the ability "to imagine how any nonrepresentational system could possibly exhibit cognitive performances" and also to model such a system.

One could quibble about whether models of sensorimotor function are a sufficient basis to argue that cognition more generally can be modeled nonrepresentationally; yet, they give a toehold. Freeman and Skarda (1990) clearly endorsed the position that representation can be dispensed with (at least in dynamical models of perception grounded in brain function) in a commentary with the title "Representations: Who needs them?" They answered: "Functionalist philosophers, computer scientists, and cognitive psychologists need them, often desperately, but physiologists do not, and those who wish to find and use biological brain algorithms should also avoid them" (p. 379). Why should they be avoided? "[T]he idea of representation is seductive," giving "the illusion that we understand something that we do not" but in fact "is unnecessary to describe brain dynamics" and even "impedes progress" (pp. 375–6). In illustration they gave this remarkable reprise of their olfactory bulb work (note that the "burst" they refer to is a spatial pattern of activity across the entire bulb under the control of one of the cyclic attractors sketched in figure 8.7):

> For more than 10 years we tried to say that . . . each burst served to represent the odorant with which we correlated it. . . . This was a mistake. After years of sifting through our data, we identified the problem: it was the concept of representation. . . . [They explain that the pattern for a given odor occurs only under conditioning and changes if the reinforcement contingency is altered or a new odor is added.] Our findings indicate that patterned neural activity correlates best with reliable forms of interaction in a context that is behaviorally and environmentally co-defined by what Steven Rose (1976) calls a dialectic. There is nothing intrinsically representational about this dynamic process until the observer intrudes. It is the experimenter who infers what the observed activity patterns represent to or in a subject, in order to explain his results to himself (Werner, 1988a, 1988b). (Freeman and Skarda, 1990, p. 376)

They further stated that this insight led them to ask new questions of their data and that their dynamical network model, with its emphasis on the role of chaos, was one of the novel answers that resulted.

Skarda and Freeman's network was intended to model an actual biological system. Researchers in the field of artificial life (the topic of chapter 9) attempt a more abstract characterization of such biological constructs as evolution, sensation, and motor control, and many of them share the skepticism about representation. For example, speaking from his experience with autonomous agents (including his insect controller networks), Beer (1995, p. 144) concluded that generally "there need be no clean decomposition of an agent's dynamics into distinct functional modules and no aspect of the agent's state need be interpretable as a representation." Philosophers focusing on evolutionary robotics, including Beer's work, have launched into a major re-examination of the notion of representation. Wheeler (1998), writing about systems that exhibit high degrees of continuous reciprocal causation, assumed that his arguments against their homuncularity (and more generally, their modularity) counted as well against their having representations. (He argued elsewhere the point that makes this plausible: the claim that these properties are mutually supportive.) This was a soft rather than hard conclusion; for example, he left open the possibility that more sophisticated evolved robotic control systems of the future would be more decomposable. Clark (1997a; all quotes from pp. 174–5) agreed that such systems present the "most potent challenge" in finding a role for internal representation, but

again preferred a more optimistic and inclusive stance. First, in coming to grips with such complex dynamics, "the notion of internal representation itself may be subtly transformed," losing some of its classical connotations while co-opting dynamical notions of inner events (chaotic attractors, trajectories in state space, and so forth; a list similar to van Gelder's). But Clark equally threw down the gauntlet to dynamicists:

> The recent skepticism concerning the role of computations and representation in cognitive science is, I believe, overblown. . . . The minimal conditions under which internal representation talk will be useful, I have argued, obtain whenever we can successfully unpack the complex causal web of influences so as to reveal the information-processing adaptive role of some system of states or of processes. . . . [C]ontinuous reciprocal causation between internal and external factors . . . appears unlikely to characterize the range of cases for which the representational approach is in any case most compelling–viz., cases involving reasoning about the distant, the nonexistent, or the highly abstract. In such cases, the focus shifts to the internal dynamics of the system under study. The crucial and still-unresolved question is whether these internal dynamics will themselves reward a somewhat more liberalized but still recognizably representation-based understanding. . . . no alternative understanding of genuinely representation-hungry problem solving yet exists, and . . . it is hard to see how to give crisp, general, and perspicuous explanations of much of our adaptive success without somehow reinventing the ideas of complex information processing and of content-bearing inner states." (Clark, 1997a, pp. 174–5)

So far we have looked at how the notion of representation has fared when confronted with biological systems and with their simulated counterparts, evolved robot controllers. Van Gelder added industrial machinery to the array of test cases in a 1995 article, "What might cognition be, if not computation?" His general goal was to argue that dynamical systems provide a plausible alternative to computational ones, and that such systems need not have representations. Although at some points in the paper he focused on a classical computational definition of representation, he also said that his arguments should go through using "pretty much any reasonable characterization, based around a core idea of some state of a system which, by virtue of some general representational scheme, stands in for some further state of affairs, thereby enabling the system to behave appropriately with respect to that state of affairs" (van Gelder, 1995, p. 351; he adapted this characterization from Haugeland, 1991). This definition of representation is broad enough to cover both classical symbolic and connectionist subsymbolic approaches.

The notion of "stands in for" requires some explication. One way to characterize it is in terms of carrying information: one state or event can stand in for another if it carries information about that other state or event. The notion of carrying information about something is usually explicated in terms of causal relations (Dretske, 1988). But this notion of information is both too general and too narrow for explicating representation. It is too general because any effect carries information about its cause, but not every effect constitutes a representation. It is too narrow because it fails to account for misrepresentation – the possibility that some state might falsely represent something else when it was not caused by it. Based on this and other arguments, Millikan (1993) proposed that we need to look in a different direction, specifically, at the agent or device that uses (consumes) the information. If a consumer Z is designed (e.g., by evolution or by an engineer) to use Y to carry information about X, then Y

might serve as a representation of X even if Y never actually carries information about X. For example, a radiation detector (Y) may have been designed for a plant supervisor who wants to be informed if there is ever a radiation leak (X). If no leaks occur, it will in fact produce only false alarms; nonetheless, the radiation detector serves to represent radiation leaks.

In practice, we generally need to look in both directions: some state is a representation only if it is used to gain information about something else, but we identify what it represents by determining what is capable of bringing it about. For example, in Lettvin et al.'s (1959) classic identification of ganglion cells in the frog's retina as bug detectors, two kinds of findings jointly were needed to determine what the firing of these cells represented: (a) increased firing of these cells generated bug-eating behaviors in the frog; (b) bug-like shapes generated increased firing. There are, therefore, three interrelated components in a representational story: what is represented (bugs), the representation (increased firing of ganglion cells), and the user of the representation (the frog, or the frog's action system).

While this is not a complete characterization of representation, it provides a sufficient foundation to begin considering van Gelder's contention that dynamical systems need not have representations. To make his case, van Gelder reached back over two centuries to James Watt's groundbreaking design for using a steam engine to power machinery via a flywheel, and suggested that its centrifugal governor would be "preferable to the Turing machine as a landmark for models of cognition" (van Gelder, 1995, p. 381). The governor was the second of two major innovations necessary to the invention's success. Watt's first innovation was a gearing system that allowed an oscillating piston to drive a rotating flywheel. But the solution to the problem of translating one kind of motion into another raised a second problem: how to maintain a constant flywheel speed in the face of constantly fluctuating steam pressure as well as resistance due to workload on the machinery being driven by the flywheel. (For many kinds of machinery, such as industrial weaving machines, it is important that a constant speed of operation be maintained, despite fluctuations in resistance, via constant flywheel speed.) The speed at which the flywheel turns can be reduced, when necessary, by partly closing a valve to reduce the amount of steam coming through the pipe leading from the boiler to the piston. Similarly, partly opening the valve increases the amount of steam. But who or what would keep adjusting the valve?

Watt's solution borrowed a technology already employed in windmills; it is shown pictorially in figure 8.9(a) and schematically in figure 8.9(b). (The pistons and gearing system between the valve and flywheel are not shown, but complete the loop.) To create a governor, he attached a vertical spindle to the flywheel which would rotate at a speed proportionate to that of the flywheel, and attached to the spindle two arms with metal balls on their ends. The arms were free to rise and fall as a result of centrifugal force. Through a mechanical linkage, the angle of the arms would change the opening of the valve, thereby controlling the amount of steam driving the piston and hence the rotational speed of the flywheel itself.

As a first step towards establishing the plausibility of the idea that cognitive systems, construed as dynamical systems, lack representations, van Gelder argued that the Watt governor operates without representations. He called "misleading" "a common and initially quite attractive intuition to the effect that the angle at which the arms are swinging is a representation of the current speed of the engine, and it is because the arms are related in this way to engine speed that the governor is able

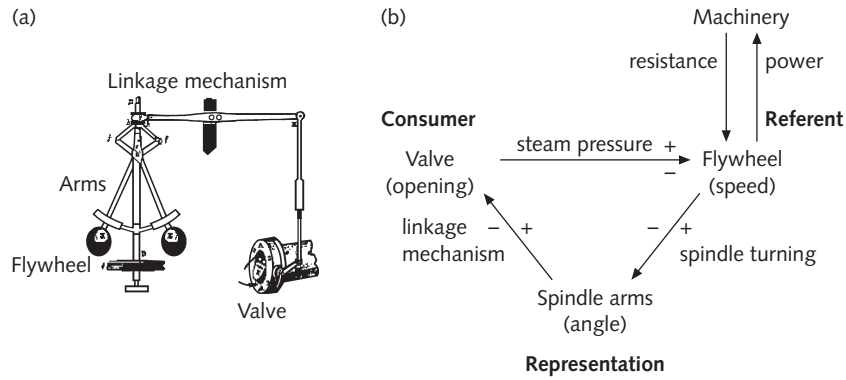

Figure 8.9 Watt's centrifugal governor for a steam engine. (a) Drawing from J. Farley, *A Treatise on the Steam Engine: Historical, Practical, and Descriptive* (London: Longman, Rees, Orme, Brown, and Green, 1927). (b) A schematic representation showing that the angle of the spindle arms carries information about the speed of the flywheel for the valve, which uses the angle to determine the opening, thereby regulating the speed of the flywheel.

to control that speed" (van Gelder, 1995, p. 351). He offered several arguments for not construing the angle of the arms as representations; here we suggest counter-arguments to two of them (for a fuller discussion of these and the other arguments, see Bechtel, 1998). Van Gelder began by contending that there is no explanatory utility in describing the angle of the arms in representational terms (that is, the dynamical analysis is sufficient). To establish explanatory utility, we must argue that (a) a mechanistic analysis is informative, and (b) that analysis includes a particular representational story about the arm angles: they *stand in for* the speed of the flywheel and can regulate the valve opening *because* they carry this information.

First, then, here is a brief mechanistic analysis of the Watt governor (see Bechtel, 1999). It has several different parts, including the flywheel, the spindle and arms, and a linkage mechanism connected to a valve. As figure 8.9(b) makes clear, each component operates on a different engineering principle and hence performs a specific subtask; each subtask contributes to the overall task of the system via the component's connection with the next component in the loop. That is, the opening of the valve gets transformed (via the piston) into the rotation of the flywheel, which gets transformed into the angle of the spindle arms, which gets transformed into the opening of the valve. In this way, we have shifted vocabularies from one describing the overall behavior of the Watt governor to one describing what its parts do. Then there is an extra step back up to the system level by connecting the task of each component to the needs of the whole system. Here it becomes clear why Watt inserted the spindle arms. It is *because* the spindle arms rise and fall in response to the speed of the flywheel that their angle can be used by the linkage mechanism to open and shut the valve. Without the spindle arms and their linkage mechanism, the valve has no access to information about the flywheel speed. They were inserted in order to encode that information in a format that could be used by the valve-opening mechanism.

This makes the spindle arm angle an instance of a more general point about representation: typically someone (a designer, or evolution, or the particular consumer produced by design or evolution) has gone to the trouble of representing a state of affairs in another medium because the representational medium is more suitable for

use by the consumer. This can be due to its format, its accessibility (e.g., words are generally more accessible to our cognitive system than are their referents), the efficiency with which it can be manipulated (e.g., computer-aided design), economy (e.g., industrial prototypes), and so forth. The representation is not just one vertex of the triangle in figure 8.9(b), a part like any other part of the dynamic loop; it was inserted to play a particular (representational) role and the system functions because it was designed appropriately for that role.

Another of van Gelder's arguments "for supposing that the centrifugal governor is not representational is that, when we fully understand the relationship between engine [flywheel] speed and arm angle, we see that the notion of representation is just the wrong sort of conceptual tool to apply" (van Gelder, 1995, p. 353). Because "arm angle and engine speed are at all times both determined by, and determining, each other's behavior" the relationship is "much more subtle and complex . . . than the standard concept of representation can handle." Here is Clark's continuous reciprocal causation again, but the complexity resides only in the dynamical analysis. It is quite possible, and desirable, to undertake a complementary classical analysis. As just noted, it (a) identifies the system's components, the subtasks they perform, and their connectivity, and (b) picks out the spindle arm angle as *representing* the flywheel's speed for use by the valve. It happens in this case that something is standing in for something else by being coupled to it in a dynamical manner. This opens the way to a dynamical analysis that makes use of the identified components, but emphasizes their coupling and provides equations that provide an elegant and specific account of their state changes in time. Within the confines of this dynamical analysis the components form a loop in which no one of them is viewed as providing a starting point, let alone a differentiated role such as referent or representation. If the equations can be uncovered, this analysis yields an elegant covering-law explanation of the dynamics of that loop–no more and no less.

In analyzing the Watt governor in this way, we have taken the position that representations should be construed broadly rather than restrictively. They can be dynamic rather than static; vary continuously in time rather than discretely; and involve quantitative operations rather than sequential manipulations of symbols. The important thing is that something is standing in for something else. Generalizing the lessons learned from the Watt governor to biological or artificial agents, it would seem that they can coordinate their behavior with an environment because components of these agents vary their states in response to the environment so as to stand in for it. Without such representations, it seems difficult to explain how the system is able to take into account specific features of the environment. We should emphasize that this does not require that the system build up a complete representation of its environment. Theorists such as Ballard (1991) and Churchland, Ramachandran, and Sejnowski (1994) have argued that we only selectively, and actively, sample the environment. Whatever information we do sample, however, must ultimately be represented within the system in order to be employed in coordinating behavior.

De-emphasizing the importance of the quantitative status of a system turns out to be helpful in characterizing dynamical analysis as well. Rick Grush pointed out in a review of Port and van Gelder's 1995 book that "A large portion of the models of 'higher' cognitive processes articulated in the book have exactly the same processing-step character as the vilified computational alternatives, even though the language, mathematics, and illustrations used to present the models obscure this fact" (Grush,

1997b, p. 235). Where equations were supplied, that is, they tended to be difference equations by which the value of a variable at time $t + 1$ depends on that variable and others at time t. This is no closer to real time than the processing steps of a classical AI model. In responding to this argument, van Gelder (1999, p. 8) arrived at a more realistic way of distinguishing types of models. "In dynamical models, there are distances in state, and distances in time, and both are systematically related to the behavior of the system." It is the geometry of state spaces in relation to time (whether discrete or continuous) that best characterizes dynamical analysis. This also suggests that the suitability of dynamical analysis for a particular network has more to do with whether trajectories in state space capture something important than with the fact that their representations are quantitative vectors rather than symbol strings. That sounds just right.

8.6 Is Dynamicism Complementary to Connectionism?

Terence Horgan and John Tienson (1996) presented a very specific vision of how connectionism and DST can collaborate in providing an alternative to classicism. They unfolded it by first characterizing the classical approach in terms of David Marr's (1982) well-known three levels of description and then adding five specific assumptions of classicists regarding these levels. Essentially, this is the same starting point as van Gelder and Port's computational approach, although in the end Horgan and Tienson rejected less of it. In summarizing their framework here, we primarily use their terminology and their characterization of classicism but also (in parentheses) show Marr's way of referring to each level.

Level 1: Cognitive-state transitions (Marr: an abstract theory of the computation) A *cognitive-transition function* (CTF) maps one *total cognitive state* (TCS) to the next; that is, it specifies input–output mappings of intentional states. The choice of function depends in part on the goal of the computation. The CTF is regarded as *tractably computable* because, classically, general psychological laws reduce what would otherwise be a brute list of mappings (*see assumption 5*).

Level 2: Mathematical-state transitions (Marr: an algorithmic specification of the computation) An *algorithm* is chosen to realize the level-1 input–output mapping (CTF) and *representations* are chosen for the input and output. Classically, formal rules (*see assumptions 2 and 3*) manipulate syntactically structured sequences of symbols (*see assumptions 1 and 4*).

Level 3: Physical implementation (Marr: implementation) The level-2 computations are realized (implemented) in a physical system. A particular machine language program run on a particular digital computer is the best exemplar, both generally and for classicists.

Horgan and Tienson then identified the five key assumptions of classicism. Assumptions 1–3 give the basic layout of level-2 rule-governed symbol manipulation (3 is a stronger version of 2 and implies it; both imply 1). Assumption 4 makes language or language-like processing a special case (it is a stronger version of 1 and implies it, but leaves room for imagistic processing, for example, to satisfy 1 but not 4). Assumption 5 asserts the fundamentally computational worldview, which the other

assumptions elaborate, and also makes a specific claim that it is realizable within available resources. Retaining the numbering and wording of the assumptions (pp. 24–5) but labeling and rearranging them for easy reference:

Fundamental computational assumption (level 1):

(5) Human cognitive transitions conform to a *tractably computable* cognitive-transition function.

Representation (level 2):

Weak (1) Intelligent cognition employs structurally complex mental representations.

Strong (4) Many mental representations have *syntactic* structure.

Processing (level 2):

Weak (2) Cognitive processing is sensitive to the structure of these representations (and thereby is sensitive to their content).

Strong (3) Cognitive processing conforms to precise, exceptionless rules, statable over the representations themselves and articulable in the format of a computer program.

In evaluating these five assumptions for their own (nonclassicist) purposes, Horgan and Tienson made a provocative cut: in addition to retaining the weakest assumptions about representation and processing, they also argued for retaining the strong assumption that representations can be syntactically structured. They picked out "hard" rules (algorithms) and computational tractability as classicism's points of vulnerability. Hence, in arguing for an alternative to classicism, they retained assumptions 1, 2, and 4 and denied assumptions 3 and 5. Their favored alternative was a dynamically oriented connectionism, which they viewed not as a half-hearted halfway house but rather as the kind of dynamicism that is needed for the job of modeling cognition.

These choices put Horgan and Tienson into several different fights, not just with classicists but also with most connectionists and most dynamicists. Horgan and Tienson took issue with classicists over assumptions 3 and 5 (by rejecting hard rules and expressing extreme skepticism that cognition could be computationally tractable). They stood with their fellow connectionists in rejecting hard rules (the general arguments are covered at length in chapter 5 and need not be reviewed here). But their insistence that there is indeed a language of thought, and that its syntactic structure must in some way be represented, placed them in opposition to many connectionists (see chapter 6). Finally, dynamicists should applaud Horgan and Tienson's rejection of assumption 5 but would tend to agree with van Gelder and Port that connectionist networks are a rather marginal medium for dynamical modeling (especially if they are viewed as realizing syntactically structured representations).

Horgan and Tienson dealt with the nay-sayers in two ways: argumentatively by dissecting and debating each assumption and positively by trying to entice them into the alternative framework that they called *noncomputational dynamical cognition* (we will simply call it dynamicism). To make the contrast with classicism clear, they constructed a noncomputational dynamical reconstrual of Marr's three levels. Their claims at each level can be summarized as follows (see their pp. 63–4 for more detail):

Level 1: Cognitive-state transitions In dynamicism the cognitive system is viewed as having general dispositions to move from one total cognitive state to another, and these dispositions are captured in psychological laws that are soft rather than general. There is no need for a tractably computable cognitive-transition function.

Level 2: Mathematical-state transitions Total cognitive states are realized as points in the state space of a dynamical system, and transitions between states are realized in trajectories through that state space. Each representation is a point (or region), but not every point is a representation. Relations between syntactic structures (e.g., sentences with vs. without a direct object) are captured not in the presence or absence of certain steps in the algorithm generating a tree but rather in the relative positions of points. The discrete mathematics of algorithms is replaced by the continuous mathematics of dynamical systems theory. More specifically, given how the dynamical system is implemented at level 3, it is a high-dimensional activation landscape in which each dimension corresponds to the range of possible activation values of one unit in a network.

Level 3: Physical implementation The dynamical system is implemented in a neural network of some sort (working hypothesis: a connectionist network). "Points in the state space of a dynamical system are realized by total activation patterns in the associated network" (p. 64).

Horgan and Tienson's denial of assumption 5 (the need for a tractably computable function) involves a distinction between *computable* and *tractably computable*. One could ask whether or not the cognitive transition functions that classicists attempt to realize in algorithms are actually computable (e.g., by a universal Turing machine). But to Horgan and Tienson, this was not the real issue. A cognitive system whose transitions are computable but intractable is as nonrealizable as one whose transitions are noncomputable. Any actual cognitive system must be realizable in a physical system that can implement mappings efficiently and quickly enough to be usable. They noted that tractable computation of this kind is far from guaranteed:

> [T]here are infinitely many functions [even] with *finite* domain and range that are not tractably computable. Consider, for example, any googolplex of unrelated pairings. (A googolplex is 1 followed by 10^{100} zeros.) The difference between infinite and huge-but-finite is not important for cognitive science! (Horgan and Tienson, 1996, p. 26)

Classicists purport to get into the small-and-finite range by means of general psychological laws which reduce what would otherwise be a brute list of mappings " too gargantuan" to specify. Horgan and Tienson presented arguments that laws of the right sort for a computational system (hard laws) are not what the cognitive system consists in, nor are they implemented by hard rules at the algorithmic level. On their view, the way the cognitive system actually is built does not involve computation (the discrete mathematics of algorithms), so the tractability question does not even arise.

Instead of general laws, Horgan and Tienson claimed that cognition has an isotropic and Quinean nature that makes computation untractable if not impossible. The problem of *isotropy* is that of gaining access to the right information for solving a given task from all of the information in the system, while the *Quinean* problem

concerns measuring the appropriate characteristics of the whole network of beliefs (e.g., its coherence) so as to determine how to revise beliefs. They credited Fodor with recognizing that these problems, originally identified in the context of confirming scientific hypotheses, extend to the general task of belief fixation faced regularly by cognitive systems. They did not see Fodor, or anyone else, as having shown how a classical system can overcome these challenges so as to handle a new input appropriately and efficiently: "Not only do we have no computational formalisms that show us how to do this; it is a highly credible hypothesis that a tractable computational system with these features is not possible for belief systems on the scale possessed by human beings" (p. 42). One avenue of response would be to deny that these are problems that need to be solved. Waskan and Bechtel (1997) contended that cognition is far less isotropic and Quinean than suggested by Fodor or Horgan and Tienson. A system with some degree of modularity (in the weaker sense exemplified in chapter 7, not Fodorian modularity) can concentrate its resources within a small part of the overall system. Rather than exhibiting an overall property of isotropy, for example, in such a system finding the right information quickly will depend on whether the right module is active (and in fact, humans not infrequently do fail to access relevant information that is somewhere in the system). A different avenue of response was preferred by Horgan and Tienson: they claimed that noncomputational dynamical cognition could easily be isotropic and Quinean. The problem of how to compute in such an environment (a version of the tractability problem) does not arise if the system does not compute.

In characterizing their noncomputational dynamical conception of cognition, Horgan and Tienson retained the commitment to syntactic structure that is a legacy of the symbolic approach (as stated in assumption 4). They construed as mistaken the common view that one of connectionism's contributions is to repudiate the use of syntactically structured representations in cognitive models. As Horgan pointed out, instead of improving on classical symbolic accounts this move produces "a seriously crippled cousin of classicism." He asked:

> What exactly are we supposed to be gaining, in terms of our abilities to model cognitive processes, by adopting an approach which (i) retains the assumption that cognitive processing is representation-level computation, but (ii) eschews one extremely powerful way to introduce semantic coherence into representation-level computation: viz., via the syntactic encoding of propositional content? *Qua* representation-level computation, it looks as though this amounts to trying to model semantically coherent thought processes with one hand – the *good* hand – tied behind one's back. (Horgan, 1997, p. 17)

It quickly becomes apparent, however, that their construal of syntax is one that many connectionists – but no classicists – would find comfortable. They pointed to Pollack's RAAM networks as exemplifying how to get syntactically structured representations in a noncomputational cognitive system. These are the same representations that we characterized in section 6.3 as exhibiting only *functional* compositionality, in contrast to the *explicit* compositionality of symbolic representations. What makes RAAM representations functionally compositional is that there is usable information in them about the constituent structure of the tree from which they were generated. However, neither Pollack nor Horgan and Tienson have probed to discover how much syntactic information is represented nor for what range of linguistic performances it is adequate (beyond the passive transformation). They also

have not pursued in any detail how the dynamical level of analysis contributes (though work by Elman and Christiansen suggests such a pursuit would be rewarding).

8.7 Conclusion

Cognitive scientists, including connectionist modelers, increasingly are employing dynamical concepts and the tools of DST. In this chapter we have seen how some researchers have plotted trajectories through state space and identified different types of attractors to better understand how their interactive networks did their job, and others have incorporated concepts such as chaos even at the design stage of their research. But some advocates of a dynamical approach to cognition claim that much more is at stake than the introduction of new tools. Van Gelder and Port (1995) contended that dynamics offers a paradigm for cognitive science that promises to replace both the symbolic and connectionist approaches. We examined critiques of mechanistic explanation and representation and concluded that these concepts need not be discarded (e.g., mechanistic explanation is complementary to the covering law explanations of dynamicism). Finally, we provided an overview of Horgan and Tienson's version of a dynamical approach to cognition, which was friendlier than van Gelder and Port's version with respect to both connectionist networks and the symbolic notion that representations have syntactic structure. We did not agree with all their claims, but their placement of dynamical and connectionist approaches at two different levels of a Marr-style analytic framework is a good starting point for working out how these approaches can complement each other.

Notes

1 A simulator for running Lotka–Volterra equations developed by Hendrik J. Blok is available free at http://www.physics.ubc.ca/~blok/files.html.

2 Of course, (c) also has starting points that are already on the cycle, and it is one of the defining characteristics of an attractor that trajectories beginning at those points will remain within it. These special trajectories have no transients. Although the cycle in (a) exhibits this characteristic, it lacks another defining characteristic of attractors, the tendency to attract nearby trajectories. Thus, *none* of its trajectories have transients.

3 Interestingly, it does not much disturb encoding of the matrix (outer) clause; the three relevant points are similar to those in figure 8.6(a). This may be a leaky network analog to "popping" the push-down stack in a standard symbolic parser.

4 This sensitivity is important, because pairs of words such as *boy* and *boys* were given unrelated encodings on the input layer. The hidden layer seems to have extracted the systematic contrast between singular and plural (as well as the essentials of subject–verb agreement) purely from distributional information in the corpus. The principal components analysis not only provides evidence of the network's systematization of number but also localizes identification of subject noun number in component 2 – an impressive contribution to understanding how the network performs its task.

5 A weight space with an error dimension yields a very detailed display of its attractors – not just their locations but also their relative depths along the error surface. Occasionally activation spaces are plotted with an energy dimension included to obtain the same effect, as in figure 8.7 in section 8.4.1 (reprinted from Freeman, 1987).

6 Understanding why the system behaves in this way requires considerably more depth in DST than our brief introduction can provide. To pursue this elsewhere, a reader should pay particular attention to the tangency of stable (attracting) and unstable (repelling) directions at saddle nodes; to obtain intermittency, the system must be nudged to near-tangency. "When the saddle nodes vanish, indicating loss of entrainment, the coordination system tends to stay near the previously stable fixed point. It's as though the fixed point leaves behind a remnant or a phantom of itself that still affects the overall dynamical behavior. . . . Motion hovers around the ghost of the previously stable fixed point most of the time, but occasionally escapes along the repelling direction (phase wandering)" (Kelso, 1995, p. 109).

7 In section 6.2 we discussed simulations of Shastri and Ajjanagadde (1993) that made use of synchrony to effect variable binding. In their simulation synchrony did not emerge spontaneously, but was created by the way in which connections were engineered. An important feature of the van Leeuwen et al. simulation is that synchrony emerges from the local activities of the components.

8 As the commentaries to van Gelder's (1998) paper make clear, there is considerable disagreement about the scope and definition of such terms as *computational*, *dynamical*, and *representation*. To keep the discussion manageable, we will focus on van Gelder's version.

9 Clark included a footnote to this last sentence beginning: "The phrase is memorable, but its authorship rather elusive."

10 In the words of van Gelder and Port (footnote 8, p. 40): "A more radical possibility is that dynamical systems can behave in a way that depends on knowledge without actually *representing* that knowledge by means of any particular, identifiable aspect of the system."

Sources and Suggested Readings

Abraham, R. H. and Shaw, C. D. (1992) *Dynamics: The Geometry of Behavior*. New York: Addison-Wesley.

Beer, R. D. (2000) Dynamical approaches to cognitive science. *Trends in Cognitive Sciences*, 4, 91–9.

Devaney, R. (1986) *An Introduction to Chaotic Dynamical Systems*. New York: Addison-Wesley.

Gleick, J. (1987) *Chaos: Making a New Science*. New York: Viking Penguin.

Horgan, T. and Tienson, J. (1996) *Connectionism and the Philosophy of Psychology*. Cambridge, MA: MIT Press.

Kellert, S. H. (1993) *In the Wake of Chaos*. Chicago: University of Chicago Press.

Kelso, J. A. S. (1995) *Dynamic Patterns: The Self-organization of Brain and Behavior*. Cambridge, MA: MIT Press.

Port, R. and van Gelder, T. (eds) (1995) *Mind as Motion*. Cambridge, MA: MIT Press. (See especially chapter 1 by van Gelder and Port, and chapter 2 by Norton.)

Waldrop, M. M. (1992) *Complexity: The Emerging Science at the Edge of Order and Chaos*. New York: Simon and Schuster.