THE ANALOG/DIGITAL DISTINCTION IN THE PHILOSOPHY OF MIND

by

Ellie Epp

B.A., Queen's University, 1968

THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF ARTS

in the Department
of
Philosophy

April 1993

Approval

Name: Ellie Epp

Degree: Master of Arts

Title of thesis: The Analog/Digital Distinction in the Philosophy of Mind

Examining Committee:

Chair: Dr. Martin Hahn

Dr. Philip Hanson
Associate Professor
Philosophy Department
Simon Fraser University
Senior Supervisor

Dr. Andrew Irvine
Assistant Professor
Philosophy Department
University of British Columbia

Dr. Robert Hadley
Associate Professor
Computing Science Department
Simon Fraser University
External Examiner

Title of Thesis/Project/Extended Essay

The Analog / Digital Distinction

in the Philosophy of Mind

Author: ________ ________

(signature)

Ellie Epp

(name)

29 April 1993

(date)

ABSTRACT

The computer metaphor of mind has been developed in an era when the serial digital computer is in ascendancy, and those classical cognitivists who support a notion of strong equivalence between mental and computational processes have had a von Neumann architecture in mind. Analog computers have offered an alternative picture of computation, and von Neumann's *The Computer and the Brain*, published in 1958, brought this sense of an alternative into the philosophy of mind by suggesting that human cognition is a function of processes some of which are analog not digital.

I have examined the analog/digital distinction in the light of this suggestion, looking first at the engineering community's uses of the contrast, and then at several sorts of philosophic construal of the terms. I conclude that, seen at the hardware level, there is no philosophically important difference between analog and digital computation, and that the contrast has its primary use in a dispute among language communities — those who offer explanations in formal/linguistic terms, and those who offer explanations in physical/mathematical terms.

Analog or connectionist systems are not easily interpreted as symbol-using systems, because they lack disjoint, finitely differentiated elements. Their intransigance in code terms, combined as it is with computational efficacy, suggests that we do not have to think of computation in terms of symbols. But those who offer a logical systems explanation have tended to think of the brain as code-using as well as code-describable. Those who say that some if not all intelligent processes do not use code, have tended to avoid logical systems explanation in favo$\overset{u}{\text{r}}$ of ren
explanation in dynamical systems terms. I argue that this separation of vocabularies is not necessary if we do not assume symbol-describable cognition is symbol-using cognition, and that any sort of formal modeling, whether logical or mathematical, implies symbol-describability at some level. The larger importance of connectionist processing does not lie in its resistance to description in symbol terms, but in the suggestions it offers about how cognitive states may have intrinsic content.

We wish to be able to say for some principled and not merely pragmatic reasons that some components of the cognitive system are more accurately described in biological (and ultimately physical) terms, while others demand a computational description. And we would like this distinction to coincide with the one between analog and digital. (Demopoulos, 1987, 83)

The issue is within what class of systems should a description of intelligent systems be sought. On one side were those who, following the lead of physical science and engineering, adopted sets of continuous variables as the underlying state descriptions. They adopted a range of devices for expressing the laws — differential equations, excitatory and inhibitory networks, statistical and probabilistic systems. Although there were important differences between these types of laws, they all shared the use of continuous variables. The other side adopted the programming system itself as the way to describe intelligent systems. This has come to be better described as the class of symbolic systems, that is, systems whose state is characterized by a set of symbols and their associated data structures. But initially, it was simply the acceptance of programs per se as the theoretical medium. (Newell, 1983, 198)

THE ANALOG/DIGITAL DISTINCTION IN THE PHILOSOPHY OF MIND

Introduction

Analog and digital systems are similar in some ways and different in more than one way, and I am going to assume there is no single right way to draw the analog/digital distinction. How we draw the contrast matters only because the notion of 'analog' has had, and in a new form continues to have, an important oppositional role in discussions of the functioning of human nervous systems. There is something about the way analog computers work — the way they *compute* — that seems to provide us with an alternative picture of what representation and computation might be and, indeed, with an alternative sense of what language we should be speaking when we attempt the discussion.

Notions of 'digital' are quite uniform and well understood, but 'analog' gives us trouble. It is vague, compendious, and not anchored to a particular, ubiquitous and ascendent technology the way 'digital' is. So I am going to look mainly at the ways 'analog' is used, first in the communications engineering community and then in the community of cognitive scholars including psychologists and philosophers.

I am looking first at the engineering community because I believe it is helpful to ground the term in the technologies that were its original context. Engineers are not concerned with providing definitions that will exclude bizarre counterexamples, and their definitions frankly accommodate the range and overlap which characterize our intuitive senses of the term. I am taking this as a virtue because it will allow me to lay out some of the several dimensions of what 'analog' means in practice. This in turn will allow me to show how and why philosophers can carve the concept in the several ways they do.

My larger intention is a defence of the cognitive alternative originally suggested by the existence of analog computers. Some of those who have defended the idea of analog cognition[1] have been ineffective because they were missing parts of the picture that have arrived since. I believe the development of parallel distributed processing supports, extends and refines the oppositional role played by earlier pictures of analog computation. This is not to say that parallel distributed processing is analog processing. (Sometimes it is and sometimes it isn't.) What I am going to suggest is that analog computing and parallel distributed processing can play something like the same oppositional role in the face of classical, digital cognitivism because they are both pictures of nonsymbolic cognition.

[1]Kosslyn's [1980] is an example.

This amounts to saying that what is important in the analog/digital debate is just what is important in the connectionist/language of thought debate — the question of whether or not we have to think of creature computation as employing a symbol system. And further: whether or not the modes of discourse we have developed to talk about operations in formal systems are suitable when we are talking about what brains do.

Analog computers are uncommon engines now, so in section I.1 I will offer a description of what they are, what they do and how they do it. In section I.2 I will use Nelson Goodman's specifications of the requirements for notational codes to show that they are symbol-describable but that they cannot be symbol-using, and to show what this has to do with mathematical continuity.

More than one kind and locus of continuity is cited in engineering definitions of 'analog'. Also important are notions of physicality, implicit representation, material implication and analogy. Analogy is tricky: just what is analogous about analog computation? This question will require a section of its own (section I.4), as will a discussion of transducers (section I.5).

Philosophers' use of 'analog' falls into two general camps. There is a rationalist line of argument — from Lewis through Fodor and Block to Demopoulos — that wishes to support the autonomy of psychology as a special science by emphasizing the language-like aspects of cognition. This group, discussed in section II.3, defines analog computation as law-governed rather than as rule-governed; in other words, as physical and not properly mind-like.

Another group, which includes Sloman and Boden and which is discussed in section II.2, defines analog computation as the operation of automated working models — computation by means of functional analogy. Analogy of function can of course be modeled formally, and most members of this group are not opposed to rationalist cognitivism.

Those who, in the empiricist style, wish to naturalize cognitive studies by emphasizing the evolutionary and developmental and perception-like aspects of intelligence, readily agree that analog computation should be described nomologically. The notion of functional analogy is not very useful to connectionists, however, since what makes nonsymbolic computation and representation possible in analog devices is not what makes it possible in creatures that construct their own representations in concert with a structured environment. Here another sense of representation is involved — representation which is intrinsic to the representer, not read into it by our uses of a device. The sort of representation ascribed to connectionist nets is, of course, still of the latter kind. What connectionist computation does have in common with analog computation and what it has in common with creature computation will be the subject of chapter IV.

A common argument from the rationalist camp says that anything an analog computer can do without explicit programs or symbols can also be done, although differently, by a digital computer. Any real numbers can after all be approximated, as closely as we like, by a rational number. This is correct and it can be granted at the outset. What is at issue however is not whether cognition can be described, or simulated, by digital computers: anything that can be modeled mathematically can be modeled in base$_2$ notation, even a gust of wind or a topological transformation. What is at issue is whether the mind-brain *is* a digital computer: whether representation and processing in the brain are what they are in digital computers.

The issue is important politically because if we think of human cognition as digital computation, then we will also think of those things a digital computer does easily as being central to human intelligence: we will think of the kinds of people who are good at what computers do easily, as being the most intelligent, and even the most human. And it is important in another way. If human brains are not digital computers, and if human cognitive virtue is of a different kind than digital competence, then we could misunderstand our own capability and fail to be as intelligent as we might.

I. The Engineering Distinction

I.1 Analog computers

> When computers were first developed in the 1940s they were divided into two large families. Analog computers represented quantities by continuous physical variables, such as current or voltage; they were fast, operated simultaneously, and had inherently limited accuracy. Digital computers represented quantities by discrete states; they were slow, operated serially, and had inherently unlimited accuracy. There was a certain amount of skirmishing about which type of computer was better for which type of job. But the technical opinion-leaders maintained a view of parity between the two families — each for its own proper niche. (Newell, 1983, 195)

When we talk about analog computation we have to talk about hardware, because there is no program as such. The machine consists of a moderate to large number of functional modules, based on operational amplifiers, that perform such operations as summation, inversion, integration and nonlinear function generation. In the early days these components would be strung together by hand. Later, patchboards were used. The set-up was generally thought of as implementing a differential equation or a set of simultaneous differential equations. Potentiometer settings would provide parameter values. Often an oscilloscope or pen plotter would graph the output.

Analog computers were good at modeling dynamical systems — physical, chemical, electrical. They could for instance model the response of the suspension system of a car to various levels and locations of impact. This kind of modeling presupposes a good understanding of the laws describing both the physical properties of the modeled system and the components of the analog model:

> To construct a mathematical analog, we take individual elements, which may have no similarity in form but have the same mathematical characteristics as the parts of the real system, and connect them together in the same manner as the parts of the real system are connected. When we say that the mathematical characteristics of the analogous device are the same as those of the original device, we of course refer to those mathematical characteristics describing that aspect of the original device in which we are interested. (Peterson, 1967, 2)

So an analog set-up is a physical implementation of a mathematical description of another physical system. It is thus an analog in the strict sense defined by Robert Rosen who says two natural (i.e.

physical) systems are analogous when they realize a common formalism (Rosen, 1991, 119). I will have more to say about this later.

Analog computation has found a new use recently in what has been called experimental mathematics. Linear time-invariant systems are the tractable part of dynamical systems studies. They can be modeled in the form of ordinary differential or difference equations. Those equations have modular virtue: any signal is a decomposable sum of its zero-input response (the response the system would evolve without input) and its zero-state response (the response the system would evolve from input excitation if its initial state were zero). The principle of superposition holds here — if x produces y, and z produces w, then $x + z$ produces $y + w$.

Nonlinear and time-varying systems are another matter. A system is nonlinear if even one system component changes its characteristics as a function of excitation applied, the way viscosity changes as a function of pressure, or friction as a function of velocity. A system is time-varying if even one system parameter changes over time, the way medium-frequency radio transmission changes over the course of a day due to changes in ionospheric reflectivity.

Neither nonlinear nor time-varying systems can be described by means of modular differential equations. Nonlinear systems, in which variables are mutually dependent, require the use of nonlinear differential equations; and time-varying systems will need partial differential equations. Both sorts of equation are generally insoluble by analytic means. The principle of superposition does not hold: the *form* of the solution will change depending on the values of inputs and initial conditions. (In system terms, zero-input response and zero-state response will not be summable because they are mutually dependent.) Examples of models which are both nonlinear and time-varying are the equations describing thermal equilibria, weather patterns, and neural signaling. So many modifications, compromises and approximations are needed when we try to digitize our descriptions of these systems that digital computers are sometimes said to *simulate* the equations involved rather than solve them — they will give us some kind of an idea of what's going on, but subject to so much error at intermediate stages that we must accept our results with caution.

Although global solutions can seldom be found for nonlinear and partial differential equations, there are ways of finding solutions for individual values of the variables; and analog computers, if correctly configured to model mutual dependencies among variables, can solve these more complex systems almost as easily as linear time-invariant systems. The new mathematics of nonlinear dynamics has found analog computers to be a direct, hands-on means of exploring the global behavior of complex equations by probing their behavior over wide ranges of values (Gleick, 1987, 244). At times this exploration is purely mathematical: we want to see how a particular equation responds over its range. At other times we are concerned to develop models of complex physical systems. In this context analog computers have unusual computational virtues:

> In the qualitative studies of finite dimensional dynamical systems, analog computers have several advantages over conventional digital machines ... Analog computers solve coupled nonlinear equations by mimicking a physical system. The error properties of a particular digital integration scheme need not ever be considered. Indeed, within the range of accuracy limited by the tolerances of the components, the *unsystematic* errors caused by the thermal fluctuations obliterate the detailed fine structure found in the mathematical description of chaos and thus effectively mimic the coarse-grained behavior that is observed in actual physical experiments in, for example, convecting fluids or nonlinear electronic circuits. (Campbell et al., 1985, 383)

I.2 Why analog systems are non-notational

> The significant distinction between the digital or notational and the non-notational, including the analog, turns not upon some loose notion of analogy or resemblance but upon the grounded technical requirements for a notational language. (Goodman, 1968, 171)

When we talk about analog computation or transmission we talk about hardware; when we talk about digital computation or transmission we talk about symbols, programs, languages. How do we account for this difference, which seems to make the analog/digital distinction a distinction across descriptional languages rather than a distinction within a common language?

Nelson Goodman says that digital technologies are able to employ a code or notational system, and analog technologies are not. Some form of representation is involved in analog signaling, but it does not qualify as notational representation.

What we want from a notational language is unambiguous identifiability of symbols and their compound expressions. Goodman prefers to talk about marks rather than tokens, and about characters rather than types, but I will adopt Pierce's token-type terminology and ground it in Allen Newell's notion of a physical symbol. For Newell a symbol is a "physical pattern in some natural or artifactual physical machine, which can be physically related to other such patterns in various ways to form compound symbol-structures" (Newell, 1980, 135).

Newell emphasizes the materiality of computational symbols, as Goodman does as well.[2] A character or type is an equivalence class of marks or tokens. 'Symbol' applies both to individual physical marks and to the type of which they are instances. Goodman himself uses 'symbol' in the loose sense by which any form of representation is said to be symbolic. In my use it will be synonymous with 'element of a code'. By 'code' I will mean what Goodman means by 'notational scheme'.

For Goodman a representational scheme may be notational only if it meets two kinds of syntactic requirement:

(1) Disjointness of types. No token can belong to two different types, whether at the same time or at different times.

(2) Finite differentiation. It must be theoretically possible to distinguish system states instantiating a 'one', say, from system states instantiating a 'zero'. "Determination of membership of a mark in a character will depend upon the acuteness of our perception, and the sensitivity of the instruments we can devise" (Goodman, 1967, 135), but not on processes requiring infinite (non)differentiation. In short, all characters of a symbol system must be disjoint classes of (theoretically if not actually) unambiguous inscriptions.

Practical code design relies on these properties of notational schemes and attempts to realize them in ways that make good use of channel properties. Disjointness of types and finite differentiation of tokens guarantee that, since a 6.38 will be read as a 6, signals will not degrade over stages, or else can be reformed at intervals. This robustness also makes possible storage and retrieval of the sort we assume in von Neumann architectures.

Here we have the relation between 'notational' and 'digital':

> Since the misleading traditional terms 'analog' and 'digital' are unlikely to be discarded, perhaps the best course is to try to dissociate them from analogy and digits and a good deal of loose talk, and distinguish them in terms of density and differentiation — though these are not opposites. (Goodman, 1967, 160)

So Goodman defines 'digital' in this way: a digital scheme is not merely discontinuous, but differentiated throughout. Since 'discontinuous' here means the same as 'disjoint', we have a digital scheme defined the same way as a notational scheme. 'Digital' and 'code' are synonymous in Goodman's definition. And the properties of notational systems are essential to the operation of digital systems:

[2]"I am speaking here of symbols only, not of numbers or anything else the symbols may stand for" (Goodman, 1967, 136).

> **digital computer** A computer that operates on discrete quantities. All computation is done within a finite number system and with limited precision, associated with the number of digits in the discrete numbers. The numerical information is most often represented by the use of two-state electrical phenomena (on-off, high voltage/low voltage, current/no current, etc.) to indicate whether the value of a binary variable is a 'zero' or a 'one'. Usually there is automatic control of sequencing (through a program) of operations so that they can be carried through to completion without intervention. (Illingworth, 1990, 131)

"'Zero' or a 'one'" gives us a paradigm set of disjoint types; and two-state pulsed signaling synced to a clocked bit-period gives us finite differentiation of tokens.

A representational scheme is analog for Goodman if it is syntactically dense, in other words, if there are indefinitely many characters so ordered that between each two there is always a third. The reals and the rationals are dense in this way. Density puts the disjointness requirement for notational representation out of reach because there are no gaps between types. We cannot say a token is a member of only one type, because every finite position will be representable by, and hence 'belong to', an indefinite number of characters.

Syntactic density is Goodman's only defining property for analog schemes, but he points out that syntactic density also makes finite differentiation impossible. Where every token is a member of indefinitely many types it will be impossible to say of any two tokens that they are members of the same or different types. Thus analog schemes are doubly disqualified from being notational.[3]

Analog computers are continuous-time systems and digital computers are discrete-time systems. This is the standard engineering taken on the analog/digital distinction. What is meant will become clearer in section I.3. For the moment I only want to point out the relation between mathematical continuity and Goodman's definition of 'analog'.

Analog computers compute continuous functions. The equations they implement are differential equations, thought of as taking values over the whole range of some real interval. Digital computers are discrete function computers; the equations they implement are difference equations, which take their values over a (rounded-off) subset of the rationals. Difference

[3]Any continuous signal may of course be discretized. We can set a ruled grid against the column of mercury in a thermometer and then disregard the intervals between marks, for instance. When we do so we are substituting a notational or digital scheme for a non-notational or analog scheme. The column of mercury is an analog signal only when the unmarked values between marked intervals are considered to have representational relevance even though they have not been discretized. Unthresholded electronic signals are analog in that all of the continuous values of certain magnitudes have causal and therefore computational relevance. Since none are disregarded by the system itself, we cannot think of the system as using a notational scheme. If physical reality is in fact granular at some ultimate level, then we could indeed consider every signal to be ultimately discrete and therefore potentially notational. But we do not have to think of a system as notational just because it has discrete elements. Discreteness of signal is a necessary but not a sufficient condition for notational systems. See also page 71 on code-using systems.

equations are discrete approximations of differential equations, adapted to handle the clocked nature and finite expression length of digital signals. In digital modeling we are not, and cannot be, interested in what happens between discrete time instants. Disjointness and differentiation depend on and follow from the minute divisions of the bit-period, and they also involve us in the approximation, the quantization error, that characterizes discrete mathematics.

So the non-notational nature of analog computers does not imply that analog computing is less exact than digital computing. The contrary may be true, within the limits of component manufactury. What mathematical continuity does imply is just that analog computation cannot be thought of as instantiating *symbols*. It can be thought of as implementing functions, and the functions being computed at any multiplier or integrator can be supplied with numerical descriptions if we choose (we could rig a device that measures current and gives us a numerical reading of an op amps's output, for instance), but it cannot be thought of as *using* symbols in its computation. Computation for an analog computer is plainly a matter of causal relations among physical materials, and the sort of causal relations involved are barred from implementing a code because they involve mathematical continuities.

It is also plain that the analog computer is implementing mathematical equations, which we think of as expressions in a formal language. If we see the equation-expression as a formal model of what the physical machine is doing, then we have a picture of something which is clearly computation, clearly describable in terms of a code, and also clearly not computation by means of a code.

I.3 Engineering definitions

'Analog' and 'digital' came into use in the 1940s and '50s to distinguish between kinds of computing technology. The terms received an extended use in other areas of electronic communications engineering when digital technologies were brought in. The old signal processing technologies — radio, telegraphy, telephone — had been around since the mid-1800s without needing a special term; now, however, these old technologies were called 'analog' to distinguish them from the digital technologies beginning to replace them. In communications engineering 'analog' also picked up senses less obviously related to analog computing.[4]

In electrical engineering, which includes computer engineering, 'analog' is applied to information sources and sinks, signals, entire communication systems, processing technologies and components, and channels.

Telephones are said to have analog *sources* and *sinks* because they receive continuous waveforms and emit reconstructed approximations of these waveforms. A wordprocessor would have digital source and sink. (See Couch 1983, 3.)

There is disagreement about how to define an analog *signal.* One writer (Kunt, 1986, 1) says an analog signal is any signal whose dependent variable is defined over a continuous range of time. It may have either continuous or discrete amplitude values. If they are discrete, it will be called a quantized analog signal. A discrete-time signal, on the other hand, is a digital signal only if its amplitude is also quantized. Other writers (Illingworth,1990, 14) say an analog signal varies continuously in both time and amplitude.

A communications *system* may be said to be analog if it transmits information from a continuous source by either continuous or discrete means (Couch, 1983, 3); or, on the other hand, if it describes a continuous source by means of continuous signals (Proakis, 1983, 60).

Lafrance (1990, 7) uses 'analog' only in relation to *representation schemes.* Analog registration (of acoustic waveforms as physical engravings on vinyl, for instance) does not employ code, while digital representation (of acoustic information as numbers on compact disk) does.

Analog *filters* are old-technology processing elements like resistors, constructed to modify a physical waveform directly. A digital filter is an algorithm, part of a program, an instruction for altering a signal by changing the symbols that represent it.

Every *channel* or medium — whether coaxial cable or seawater — is, as a physical substance, inherently continuous and therefore analog.[5] But some writers will call any channel used to carry the pulsed signals instantiating binary symbols a digital channel, because it is used to transmit a digital code (Couch, 1983, 4).

[4]The term has had other even more general applications to just about any kind of working system, mechanical or not, representational or not. I will not concern myself with these uses because they derive by metaphor from one or another of the primary engineering uses I will outline here. The inconsistencies in these wide-ranging applications are attributable to the inconsistencies in engineering usage.

[5]"The question of whether the signal (or its source) is intrinsically continuous is unresolvable: any experiment to determine this would require infinite bandwidth (or infinite time) and infinite signal-to-noise ratio, and so would be impossible in practice. All that is in question is whether a discrete or continuous representation is more convenient, or useful, or appealing" (Illingworth, 1990, 134).

To summarize, some of the dimensions of contrast picked out by the analog/digital distinction in engineering practice are these:

(1) A source is analog if the information we are picking up arrives continuously.

(2) A signal is analog if it is a continuous function of space or time parameters,

OR

it is analog if it is either a continuous or a discrete function of the continuous independent variable.

(3) Systems are analog if they describe a continuous source by means of continuous signals,

OR

they are analog if they transmit signals from an analog source, regardless of how transmitted.

(4) Analog processing, analog filters, are realized in hardware directly, while digital processing is implemented in a program which is then realized in hardware.

(5) Channels may be called analog simply in virtue of their physical continuity,

OR

they may be called digital when they carry pulsed signals

OR

they may be called digital whenever they are carrying signals from a digital source, even when the carrier is continuous.

I.4 "An analogous electrical network"

Engineers often remark that in analog technologies there is some sort of analogy between representing signal and represented source, while in digital technologies there is not. A standard example would be analog and digital acoustic storage media: in an analog tape medium, variation in an acoustic waveform is represented as a proportional variation in degree of magnetization of a ferrous coating; the degree of magnetization of a digital tape is not proportional to amplitude of the source signal because, on a DAT tape, magnetization represents binary numerals. Hyndman calls an analog computer "a direct analogy between the behavior of variables in a mechanical system and those in an electrical system" — it is an "analogous electrical network" (Hyndman, 1970, 6).

Proportional relations between representing and represented values can of course be exploited in filtering and other sorts of processing. Where the original value had representational significance, the transformed value will also have representational significance. This is the principle behind computation of any sort — the structure of relations in the representing system is set up to preserve the structure of relations in the represented, regardless of transformations, and in such a way that systematic transformation in the representation models some transformation, often causal, in the represented. Where our representing relations are logical relations in a code, transformations will be truth-preserving. In digital systems representational relations are of this logical kind: they are nonproportional, but otherwise systematic, rule-governed relations among expressions of a code.

Where representational relations are among values of current at various components of an analog computer, systematic physical transformation is what preserves predictive relevance. An analog computer "mimics a physical system" by being a physical system whose materials preserve representational relevance through systematic causal transformation.

There is a long tradition of working models in engineering, and the analog computer is seen as a superior sort of working model — superior because of its flexibility and general applicability. Analog computation is the operation of a working model. This is just what makes analog computers relevant to cognitive questions — they offer a picture of nonsymbolic inference, systematic transformation without code. This is what Campbell means when he says analog computers solve nonlinear equations by mimicking physical systems (Campbell et al., 1985, 383).

It is sometimes said (see Peterson, 1967, 2) that an analog computer is analogous not to some other physical system, but to a differential equation, a set of mathematical operations. Its input and output are, after all, numbers, at least from our point of view. Its components are called 'summers', 'scalers', 'integrators', 'nonlinear function generators'. General purpose versions of the machine are often called differential analyzers. The device does combine different values of some electrical magnitude in the same manner as numbers are combined in a mathematical operation.

What is wrong with saying analog computers are analogs of equations is that it conflates analogy with modeling; It does not preserve the distinction between physical things and their descriptions. An equation is a description in a mathematical formalism. It is a formal expression. It can, therefore, be *implemented* or *realized* in an analog computer's system states. And the computer's system states can be *modeled* by the equation. This allows us to reserve 'analogy' for the relation between two physical systems whose system states realize a common description. #

There are more and less informational ways of writing system equations. I will have more to say about equation forms in section II.2. For now I will just mention that engineers find it valuable to implement equations given in the form that allows the most direct and detailed modeling of components and their relations. A system equation giving the global input-output function of the system will be some n-th order differential equation. If this equation is put into its *normal form* of n simultaneous first-order equations, and implemented in this form by simultaneous parallel circuits, then the engineer watching the computation can 'see' more of the internal dynamics of the system.

Some of the ways physical systems can be analogous are these:

(1) They may be analogous with respect to time. An analog computer which is continuously computing the evolution of some system may be operating in real time, that is, in parallel with the represented system, or it may be time-scaled, with proportionality preserved.

(2) They may be analogous with respect to spatial relation. Analogous computers may, if we like, be set up so that components are in the same spatial orientation to each other as working parts in the represented system. This would have no mathematical purpose, but it would make the computing system easier to 'read'.

(3) They may be analogous with respect to complex, simultaneous, causal interrelations among system components independently identifiable. The analogous systems might be thought of as realizing a weighted directed graph of causal connections.

(4) They may be analogous in some higher dimensional way, where their mutual description is realized in relations among relations of first-order magnitudes in the two physical systems.

(5) All of these dimensions of possible analogy can also be analogies with respect to part-whole relations.

I.5 Transduction and codes

Once information-bearing physical parameters have been expressed as a suitable set of signals by the intermediacy of transducers such as keyboards, analog-to-digital converters, etc., we need no longer be concerned with the physical parameters themselves ... The physical details of an information source are irrelevant to the communications engineer. Only its unique representation by mathematical symbols matters. (Lafrance, 1990, 7)

"We need no longer be concerned with the physical parameters themselves": the most interesting aspect of the analog/digital distinction is this leap in category from physical to linguistic. A code, it is admitted, is always 'realized' by physical markers in some physical medium; but it is assumed that realization may be dropped from the discussion. Why this is so, and what we import with this assumption, is the question at the heart of this thesis. Answering it would help us see through old disagreements about analog and digital representation, and now disagreements about the import of connectionism.

I will make a start at answering it by looking more closely at transducers, and in particular at A/D converters. A/D converters are the technical bridge between analog and digital systems. Are they also a bridge — is such a thing possible — between a domain of physical description and a domain of linguistic description?

A transducer, in its most general sense, is any device by which energy is transmitted from one system to another, whether of the same or a different kind. The term is usually used, however, for devices which convert one form of energy — sound, light, pressure — into another, usually electrical energy. An example is a photomultiplier tube which converts light energy into electrical energy.

In any old-style communications system, transducers would accept non-electronic signals and emit electronic signals which could then be filtered or otherwise processed in this electronic form. Transduction here was called 'source coding', because a form of representation was involved: light waves were being represented by electrical impulses, for example. If yet another sort of channel — airwaves maybe — was to be involved in signal propagation, there would be a second transduction called channel coding, which often involved modulation of carrier waves.

When we describe analog information systems we do not easily lose sight of the fact that physical energy is being propagated through a physical medium. And we don't have to think of the message as being in existence when it is in transit. It — the analog form of it that human senses can recognize — is reconstructed by channel decoders and source decoders at the other end. There is no music in the grooves of the record. And, oddly, no speech in the telephone lines. Nothing but electrical currents. Potential messages, we might want to say.

With digital media something changes. We are not tempted to say there is music on the DAT tape or compact disk, but we *are* tempted to say there are numbers. We might say we have our texts on floppy disk but we more seriously believe our binary 1's and 0's are stored there. These phenomena are a consequence of digitization, which is the task of a transducer called an A/D converter. Interestingly, an A/D converter is not usually called a transducer, although it is thought to require a transducer as its first stage.

Digitization involves representing a continuous-time signal in the form of an *n*-bit binary word. Two steps are usually recognized: sampling and quantization. Sampling measures the amplitude of a continuous-time signal at discrete intervals. Formally, it is the mapping of a continuous-time signal onto a finite subset of its coordinates in signal space. Nyquist's sampling theorem assures us that a (band-limited, i.e. prefiltered) signal waveform can be uniquely represented by samples taken at a rate at least twice as fast as the highest period contained in the signal. Sampling, also called time-quantization, leaves us with sampled real signals — signals whose amplitude values still range over the real numbers.

Amplitude quantization is sometimes called *coding*, because it matches an absolute amplitude value to a digital word representing some amplitude closest to the actual sampled value. Amplitude quantization and coding are conceptually separable but physically identical. A *sampled* signal is said to be pulse amplitude modulated (PAM), and still analog (half-analog maybe, i.e. analog in the sense that signal amplitude still "denotes analogic information"). A sampled *quantized* signal is said to be pulse code modulated (PCM) and digital (see Couch, 1983, 82-88).

> The PCM signal may be viewed as an encoded instantaneously sampled PAM signal ... The PCM signal is obtained from the quantized PAM signal by encoding each quantized sample value into a digital word. It is up to the system designer to specify the exact code word that will represent a particular quantized level. (Couch, 1983, 87)

There are half a dozen different serial-bit signaling formats. In unipolar signaling a binary 1 is realized as a high level and a binary 0 as a zero level of current. In differential encoding a binary 0 is realized by a change in level and a binary 1 as no change in level. This format has the advantage that polarity may be inverted without affecting recoverability of data.

Physically, transduction is a process that takes us from

(1) a continuous waveform in some physical channel

to

(2) a continuous waveform carried intermittently in an electronic channel,

to

(3) a patterned sequence of pulses and absences of pulse carried on a similar electronic channel.

Conceptually, what we have is a process that takes us from

(1) a real-valued and linguistically inexplicit representation of quantities by the measurable but unmeasured magnitudes of some physical parameter,

to

(2) an intermittent but still real-valued representation of magnitudes in such a way that the order of representing signals is time-correlated with the order of represented signals, and the amplitude of representing signals is proportional to the amplitude of represented signals,

to

(3) intermittent pulses whose order of amplitudes is not at all related to the order of amplitudes of represented signals.

The overall sequence of *code words* does retain some temporal relation to the sequence of waveform values coded, but the sequence of pulses within a code word is unrelated to the values coded. Transducer output is determined rather by the designer of the code, who has decided that a measured value of +7 (or +6.85, or +7.16) will be coded as the binary sequence 110, and a measured value of -3 as 001.

The transduction process loses two, correlated dimensions of analogy — amplitude and temporal order — and it gives us a temporal mapping between code expressions and sampled regions of our original signal: code expressions naming measured quantized amplitudes flow from the transducer in the same order as the signals they describe. But there is no analogy between the *elements* of the code words — the pulses and no pulses — and the pulses or no pulses of the original signal. The dependence of the operation of the code on some form of temporal correspondence is worth noticing but the 'analogy' now is between the ordering of physical entities, waveforms, and the ordering of linguistic entities, code words. What has happened is that we are only able to identify the chunks of the transduced signal relevant to *representational* order by knowing they are code words.

There is no longer an immediately systematic relation — analogy — between the physical/causal properties of the source signal and the physical/causal properties of the transduced signal. It does not follow that transformations within the post-transducer medium will be unsystematic with respect to the physical/causal properties of the original signal. The contrary is true: any transformation will be effectivelyreversible; we will always be able to approximate the original signal's physical properties. But the systematicity which allows for computation — which preserves relevance to the original signal — now is the systematicity of the code.

There have to be two sorts of systematicity to give us this reversibility and computational relevance between signals that are not physically analogous. One is the systematicity of the formal system: the discrete nature of the elements, the syntactic specificity of the rules for combining them into expressions, and the total explicitness of the procedures which determine how transformations will take place. The formal system is our inferential engine.

The other sort of systematicity is the systematicity of our encoding schemes and practices. The systematicity of the formal system is typically axiomatic, but the systematicity of the modeling relation — and that is what we are talking about — involves empirical decisions, trial-and-error, curve-fitting, testing. The transducer's designer first has to decide that a measured signal amplitude of +6.85 will be encoded as if it were +7, and that both values will emerge from the transducer named '110'. The designer then has to be able to implement the decision in circuitry.

If quantization is to be nonlinear, for instance, with more importance given to some areas of dynamic range than others, the nonlinearity of the relation between source signal and quantized signal must also be encoded into system equations, or inference will be lost.

The systematicity of computation by means of formal system and encoding practices replaces the analogous systematicity of computation by means of proportional transformations. It does this precisely by encoding the proportionality itself. Any number system and all measurement practices are designed to do this. So the fact that computation over symbols preserves inference is not mysterious.

What is mysterious, what continues to have a feeling of legerdemain about it, is the way physical computation seems to turn into nonphysical computation by passing through an electronic transducer. One way to handle the transubstantiation is to talk about dual description. We might say something like this: as soon as we have a code we have the option of talking about what is happening in terms of linguistic expressions or, as often happens, in terms of their referents. We also, if we happen to be engineers or neurologists, still have the option of talking about energy propagated in a physical medium. Both descriptions are valid. Which we choose depends on our interest. If we are interested in the *representational* function of the signals we must talk about code words and their referents because, as we saw earlier, there is now no other way to identify the units doing computational work. So the story goes.

This story does work for digital computers. When we are talking about human cognition, though, the problem with dual description is that it perpetuates a form of mind-body split. We do not know how to translate from the terms of one description into the terms of the other. We have (to anticipate chapters III and IV) top-down functionalist-intentionalist description working its way downward toward finer grained functional description, and bottom-up structural-biological description working its way upward toward more comprehensive structural organization, and a curious blank where we would want there to be a join.

For the transducer, though, there is no problem. So I would like to look again at how the transducer does what it does. Input: physical waveforms. Output: physical waveforms. I have said there is no longer a relation of *analogy* between the form of the input waveforms and the form of the output waveforms. But there is some other systematic relation between the two physical signals. If there were not, machines could not compute or represent. And it is a physical relation too, but it is a relation that cannot be seen if we look just at the output of the transducer.

We also have to look at how the transducer output signal is processed in the digital machine. Incoming pulsed signals — one by one, or in batches corresponding to register capacity — simply reset switches. It is said that all a computer can do is add and compare. But even this is anthropomorphizing. All a computer can do is reset switches. It resets switches not because it can 'read' but because the materials it is made of have response properties. It can't read; but it does not have our problem recognizing what the representational units in a stream of pulses and no pulses are. It does not have to move into intentional or linguistic description to do so. And this is because the materiality of the machine — the physical organization of the machine — is doing the work.

Even the Turing machine, that icon of virtual computation, is a description and not a computer as long as it does not have a minimal physicality — a power source, a bit clock, a mechanical tape mover, a real tape with right and left sides, a magnetize/demagnetize head, and, yes, a tiny transducer setting up pulses or no pulses in response to magnetization magnitudes. All of these mechanical and electronic functions would have to be enabled by the causal properties and spatial organization of a whole slew of little switches that either open or don't open when each pulse or no pulse is evoked. It is this whole contraption that instantiates the code.

We have seen that physical computation, either of the analog or the digital kind, requires a systematic relation of some physical sort between represented and representing signals. Now we are in a position to see that the physical form of the input waveform has a systematic relation not (immediately) to the form of the output waveform, but to the electrical state of the entire machine. An incoming pulse-pulse-no pulse will only function as a binary 110 if certain switches are in certain states; and an incoming 110 will only function to represent a source magnitude of +7 if many other switches are in certain states. This is what compilers and assemblers are are about, making sure the entire electrical configuration of a machine is such that it will conform to the program we think we are running. Or we could say it the other way: making sure the program we are running is able to make inferential use of the causal properties of circuit materials. We *can* say it either way because there is reciprocity here. We have no philosophical problem getting top-down functional description (what we say the computer is doing in task domain language) and bottom-up electrical engineering description (what we say current is doing in the machine) into register at any scale we like. All that is needed is massive technical labour.

With brains, massive technical labour has not yet been enough. The important difference is that we make computers and don't make brains. We know how hardware realizes software because we have built it to do just what it does.

One question remains. Is the relation between the physical form of the source signal and the electrical configuration of the entire machine a relation of analogy? There will certainly be a mathematical mapping possible between a signal space description of signal waveform and a state space description of the electrical configuration of the machine, and this mapping will be subject to systematic transformation. In other words, seen in the right way there *is* an analogy between source signal and representing electrical configuration.

This does not give us logical parity between ‘analog’ and ‘digital’, however. Why not? Because ‘digital’ never is seen in the ‘right’ way. It is always seen in terms of the referents of the instantiations of the code — in terms of 0’s and 1’s, or in terms of higher level program entities. ‘Digital’ and ‘analog’ belong to different language-games.

II. Philosophers' Distinctions

II.1 Von Neumann

> The nervous system is based on two types of communications: those which do not involve arithmetical formalisms, and those which do, i.e. communications of orders (logical ones) and communications of numbers (arithmetical ones). The former may be described as language proper, the latter as mathematics. (von Neumann, 1958, 80)

Von Neumann's *The Computer and the Brain*, written in 1956 and published in 1958, made the analog/digital distinction relevant to philosophy by claiming that the logics and mathematics of the central nervous system, viewed as representational systems, must "structurally be essentially different from those languages to which our common experience refers" (1958, 82). He has in mind here both natural language and binary mathematics.

Von Neumann did not claim, as is often said, that the brain must be an analog computer. He thought it was a system using both analog and digital signal types, organized so there is frequent interaction throughout the system. His sense of 'analog' comes, of course, from analog computers: "in an analog machine each number is represented by a suitable physical quantity" (1958, 3). He is thinking of analog computers in their computational and not their analogical aspect, here, and so he is thinking of voltage or current magnitudes as representing (continuous) *numbers* by non-coded means. His sense of 'digital' emphasizes code: "in a decimal digital machine each number is represented in the same way as in conventional writing or printing, i.e. as a sequence of decimal digits" which are, in turn, instantiated as physical magnitudes (1958, 6).

His sense of computation is in accord with the kind of signal being used: analog computation is systematic transformation of signal magnitudes by physical means. Digital processes are "patterns of alternative actions, organized in highly repetitive sequences, and governed by strict and logical rules" (1958, 10).

Von Neumann thinks of "nerve pulses", spiking frequencies in axons or dendrites, as having a digital character because he thinks of them as discrete and notational. Given the possibility of dual description, von Neumann is giving spiking frequencies a linguistic description: they are "communicating orders", telling other physical systems what to do. Chemical changes in the nervous system are analog for von Neumann because they are a form of communication which is directly causal and not coded: they have representational significance but do not involve symbols. Because they have computational effect von Neumann thinks of them as arithmetical, quantitative. These continuous quantities are then transduced:

> Intensities of quantitative stimuli are rendered by periodic or nearly periodic pulse trains, the frequency always being a function of the intensity of the stimulus ... That it is a *monotone* function ... permits the introduction of all kinds of scale effects and expressions of precision in terms that are conveniently and favorably dependent on the scales that arise. (von Neumann, 1958, 77)

Notice here that analogy is still present, though von Neumann is talking about digital signals. To his mind we have gone from physical/implicit magnitudes to signals encoding numbers, but the code employs proportional relation between magnitudes and pulse frequency. Of course this is not true of codes in a digital computer, but in the nervous system it would give neural coding an essentially statistical character:

> What matters are not the precise positions of definite markers, digits, but the statistical characteristics of their occurrence, i.e. frequencies ... Thus the nervous system appears to be using a radically different system of notation from the ones we are familiar with in ordinary arithmetics and mathematics. (von Neumann, 1958, 79)

Von Neumann set a challenge to cognitive studies because he said that if the brain is a computer it is not necessarily or wholly a *digital* computer. Even the aspects of neural function that seem to be using discrete signals seem to be using them in a different form of code than we use in digital programs. That the CNS might be, or be significantly like, an analog computer was the obvious alternative, but most writers who followed von Neumann had technological or philosophical or political reasons for wanting to show that the brain is digital after all.

I will look first at a group who attempted to understand computation by means of functional analogy, and then at a line of argument committed to digital description.

II.2 Working analogy

> We recall that two different natural systems N_1, N_2 are analogous when they realize a common formalism. F. As we saw, analogy is like a modeling relation except that it relates two natural systems, rather than a natural system and a formal one.
>
> The relation of analogy between natural systems is in fact independent of their material constitution. The most materially disparate systems can still be analogous. All that is required is that each natural system, individually and separately, be describable by the same formalism F. (Rosen, 1990, 119)

Margaret Boden uses 'analog' in a way that applies to representation in general and is far removed from its roots in computing technology. Consequently she ignores the continuity/discreteness aspect of the contrast and settles on this definition: "since significant similarity contributes to all senses of analogical representation, it may be regarded as the basic, minimalist definition of the term" (Boden, 1988, 29).

Because she is disregarding the question of continuity, her definition can cut across the symbolic/nonsymbolic line. Consequently her "significant similarity" can be either a modeling relation or an analogical relation, in Rosen's sense of those terms — it can be a relation between two physical processes that both realize the same formalism, or it can be a relation between a physical thing and a formalism. Boden is a classical cognitivist; her definition allows for mental representation that is both analog and symbolic.

For her the relevant contrast is between representation that has *some* and representation that has *no* "interpretive mapping, or significant isomorphism, between the structure of the representation and the structure of the thing represented" (Boden, 1988, 29). Her contrasting term is not 'digital' but (Sloman's term) 'Fregean'. Normal word order in a sentence would count as Fregean representation when it does not reflect the order of events described; it is Fregean because it involves the application of logical functions to arguments by a process like filling in slots in a pre-built sentence structure. On those occasions where word order does reflect some significant order of events ("Jehane drank her cocoa and went to bed" means something different than "Jehane went to bed and drank her cocoa") the sentence is said to be both analog and Fregean.

This is a definition that could be fun. It would give us 'tall' as analog because of its preponderance of (relatively) tall consonants, 'bite' as analog because of its snap of the teeth, 'look' as analog because it shows two eyes, 'howl' as analog for its sound, and 'concatenated' as analog because it is what it means. But none of this is relevant to whether human brains are analog or digital computers. We need a reading of 'significant similarity' that is systematic enough to give us inferential structure.

Johnson-Laird (1988) offers a contrast between mental models and propositional representation which does not mention analogicity but which could stand as a gloss for the similarly ambiguous notion of an analog medium of representation found in Kosslyn (1980). Johnson-Laird cites as his forbear Craik (1943) who, with a sense of computation not yet informed by digital computers, thought human brains might construct some form of

> physical working model which works in the same way as the processes it parallels, in the aspects under consideration at any moment ... By a model we thus mean any physical or chemical system which has a similar relation-structure to that of the processes it imitates. (Craik, 1943, 51)

What we are talking about here is analogy between two physical processes whose causal structure is 'the same' — i.e. whose respective causal relations can be modeled in the inferential relations of some common formalism. Johnson-Laird does not commit himself to a non-*symbolic* understanding of what I will call working analogs. For him a 'mental model' is "not propositional", and by 'propositional' he seems to mean something like what Boden and Sloman mean by Fregean representation — he gives the predicate calculus, semantic networks and natural language sentences as examples. But does 'not propositional' imply 'not notational', not coded? He does speak of a "set of tokens that corresponds to the set of men, a set of tokens that corresponds to the set of cars, and a mapping relation between these two sets of entities". On the face of it these "tokens" and "mappings' could be just a way of speaking about something actually to be understood in hardware terms directly, as weighted neural connections passing a flow of patterned activation, perhaps.

The sorts of mental model or analog Johnson-Laird postulates involve some of the dimensions of analogy I have discussed. A "simple relational" model/analog provides analogy of elements and their properties and their relations. A spatial model/analog provides element-relation analogy in which relations are spatial. A dynamic model or analog provides causal analogy along with temporal analogy. (See Johnson-Laird, 1983, 422-3.) Any of these may also be part-whole analogies.

It is difficult to know how to take this proposal. A mental model has a "similar relation-structure to the process it models", and this differentiates it from "a simulation which merely mimics the phenomenon without relying on a similar underlying relation-structure" (Johnson-Laird, 1983, 4). What is ambiguous is whether the "underlying relation-structure" is implemented as a physical/causal structure or as a data structure like an array. This is the ambiguity that also troubles Kosslyn's story about an analog medium of image representation.

What is troublesome is that both Kosslyn and Johnson-Laird want cognition to be entirely modelable in formalisms suited to digital computers. Both are aware that a code-modeled procedure is not necessarily a code-using procedure; both want the determinacy and compositionality of code structures; and yet both want to talk about representational structures that have inference built into them in ways that are importantly unlike the ways inference is built into, for instance, a number system, or other systems normally used in digital computation. The question is, how do we understand an inferential relation over *symbols*, which is semantics-driven and not syntax-driven? There will be more about semantics in chapter IV. For now I would like just to pursue what might be meant by a physical relation-structure.

Any relation-structure, temporal succession for instance, can be modeled (in Rosen's sense) as a data structure in a code. Another physical event, modelable by the same formalism, may provide an analog of its temporal relation. If our cognitive creature is using a code — if the mental model Johnson-Laird proposes is a relation-structure implemented in a data structure — then that data structure must itself be physically realized. What is the nature of the mapping (through intermediate states if necessary) between data structure and the physical states realizing it? Does the data structure model its realization? Yes: modeling and realization are symmetrical with respect to each other. So, technically speaking, even when the creature is using a code there will be a relation of analogy between the world-event and its brain state representation. In other words, if we have a coded model physically realized, we will always have an analogy — of some kind. Now we need to ask whether there is an important difference in kinds of analogy. ¶

Let's say we have some physical process we want to model — a bridge subjected to a flood for example. We want to know whether the bridge will hold. We write an equation that describes the response properties of the parts of the bridge and the stressing properties of the flood at various points and times. We tie these descriptions together with mathematical operators. Our equation can take different forms which nonetheless are mathematically equivalent.

We write the equations in a form readily interpretable into the bridge and storm scenario. We have a set of simultaneous equations where parts of the equation model particular parts of the causal story, and where mathematical operations are ordered the way the causal sequence is ordered. Then we set up an analog computer to realize this equation — to be a *working analog* of the bridge and storm. It will give us a result that tells us whether the bridge will hold.

Alternatively we can write the equations in a parsimonious form in which operations are differently ordered and values are differently lumped. This equation will give us the same result. When we realize this form of the equation on our analog computer, the analog computer and the bridge-with-flood are still analogs, because they realize a common formalism. But the computer is no longer a working analog of the bridge system. It is a *functional analog*: if we consider it a black box we will say that it implements the same global input-output function.

Yet another alternative is that we write the equations in an extremely prolix form, in which recursive functions are defined over binary digits. If we realize this equation in an analog computer our analog computer is a digital computer. Once again, the physical machine will be the analog of the bridge with flood; they realize the same formalism. Once again it will be a functional or black-box analog but not a working analog.

So now our question is this: Are brain states working analogs or merely functional analogs of world states? The answer, it seems, might be a matter of detail.

Let's go back to Craik's naive formulation in which human brains may be thought to construct a "physical working model which works in the same way as the processes it parallels." (Recall that I want to preserve Rosen's distinction between a model and an analogy, and so am speaking of working analogs rather than working models.) What is involved in a physical analog "working the same way" as something else? We want to disqualify any sort of global input-output functional correspondence because we have in mind some of the ways the details of representing structure can be more rather than less representationally relevant.

Sloman (1978) (in a chapter called "Intuition and Analogical Reasoning") puts it this way:

> Analogical representations have parts which denote parts of what they represent. Moreover, some properties of, and relations between, the parts of the representation represent properties of, and relations between, parts of the things denoted. ... An analogical representation has a structure which gives information about the structure of the thing denoted, depicted or represented. (Sloman, 1978, 165)

Sloman is principally interested in non-formal inference and he wants to establish the plausibility of forms of representation that would make it possible. He uses 'analogical' and not 'analog' and, like Boden, he has an understanding of the term dissociated from analog computation with its necessary continuity. So his contrast is not between discrete and continuous, or between notational or non-notational, or between model and analogy. His use of 'symbol' also does not discriminate between explicit/notational code and representation generally. This should be kept in mind throughout the following passage:

> The contrast between Fregean and analogical symbolisms is concerned with the ways in which complex symbols work. In both cases complex symbols have parts which are significant, and significant relations between parts ... The essential feature of Fregean symbolism. is that all complex symbols are interpreted as representing the applications of functions to arguments ... It will suffice to notice that although a complex Fregean symbol, "the brother of the wife of Tom," has

"Tom" as a part, the thing it denotes (Tom's brother-in-law) does not have Tom as a part. The structure of a complex symbol bears no relation to the structure of what it denotes, though it can be interpreted as representing the *structure of a procedure for identifying* what it denotes. (Sloman, 1978, 164)

An arithmetic expression like

$$\frac{3 \times 5 \ + \ 4 \times 3}{11 - 2}$$

which Sloman takes as Fregean, can certainly be seen as representing a procedure for finding what the expression denotes. But notice that if this equation were realized on an analog computer, and if individual numerical values were encoded magnitudes from our bridge-in-flood system, and if addition, subtraction and multiplication encode causal relations, then the analog computer realization of the equation will (intuitively speaking, since we still lack a definition) be a *working analog* of the bridge in flood.

It may be that if we think of 'procedure' in brain terms, a procedure being what the brain does, in which order, then "procedure for identifying what is denoted" would be just whatever the brain does when something is being understood. If there has to be an activation of neural patterns instantiating the many things known about Tom, in order to activate the neural patterns instantiating "Tom's wife", in order to activate the neural patterns instantiating "her brother", on the way to activating the neural patterns instantiating "Jerry Rigg", then it will also be true of the structure of a brain procedure (the sequence of structurally-related system states) that "some properties of, and relations between, the parts of the representation, represent properties of, and relations between, parts of the things denoted".

I will emphasize again that the Fregean expression is a model not an analogy. The way the expression is realized in the brain hardware of a person understanding it is an analog if there is some world state which also realizes that formal expression. This analogy may be a functional analogy, or what I have called a working analogy. At this point our Fregean model does not give us enough information to be able to tell which, but there could be another formal expression, more complex, relationally more specific, detailing more entailment structure. (See Rosen, 1991, 98-103.) Two physical configurations which were analogs in relation to the above formalism may also be analogs in relation to this much more detailed formalism. We could thus define 'working analogy' as a relation mediated by models with more detailed, more informative, entailment structure. This would put working analogy and functional analogy on two ends of a continuum, and I would think this is correct.

Where does this leave us with analog and digital, symbolic and non-symbolic? We have agreed that analog computers are continuous function computers and that they are therefore not code-using devices. We have seen that an analog computer configured to realize some differential equation is a functional analog of every other physical process describable by that equation — and that this property is one they share with digital computers. I have proposed that they will be working analogs of another physical process when the formal description that models them both is an expression in a formalism capable of noting more complex and detailed entailment structure. This property is the one that can distinguish them from digital computers, whose causal states and relations will not be mappable onto the more detailed inferential structure of the expanded formalism.

II.3 The rationalist line

David Lewis published a paper in 1973 attempting a definition of analog and digital representation of numbers. Fodor and Block in an often cited unpublished paper written the same year attempted to improve on Lewis's definition. Both papers inspired replies, the most important of which was Demopoulos (1987). Although the spirit of Pylyshyn's (1984) definition would make him part of this group, I will look at his distinction separately because the detail of his treatment makes it a useful foil for the discussion of connectionism which will follow.

Lewis draws the analog/digital distinction in relation to the fundamental versus derived nature of the physical magnitudes that characterize the relevant physical realizations of computational states. "Analog representation of numbers is representation of numbers by physical magnitudes that are either primitive or almost primitive" in some "good reconstruction of the language of physics" (1973, 324-5). Digital representation of numbers is "representation of numbers by differentiated multidigital magnitudes" (1973, 327).

A physical magnitude as Lewis is defining it is the measuring practice which assigns numbers to physical systems. A primitive magnitude is one that is straightforward in its operation — something like measuring electrical resistance or weighing a volume of fluid. These measurement functions — voltage, current, resistance, weight, velocity, and the rest — are "definable in the language of physics more or less as we know it" (1973, 323).

Digital representation of numbers in a computer is also representation by means of physical magnitudes. Why then isn't it analog representation? Lewis imagines a 36-bit register where each bit position indicates a binary 0 or 1 by means of negative or positive voltage. This is digital representation of a number, and it is representation by means of physical magnitudes. At each bit position the magnitude is also primitive in his terms. But the number is not represented by the individual magnitudes. It is represented by a particular pattern of magnitudes, and *pattern* of magnitudes is not a primitive magnitude in the language of physics as we know it. Lewis would call it a derived magnitude. Describing a pattern of magnitudes is no straightforward matter, and this complexity of description is what differentiates digital from analog representation.

The pattern of voltages in Lewis's 36-bit register resembles what in section I.5 I was calling the state space description of the electrical configuration of the machine. It too could be described in the form of a vector, an ordered set of primitive magnitudes representing voltage at the 36 bit positions. The thing about both Lewis's bit pattern and the state space of the machine is that the causal properties of the relevant physical magnitudes are emergent. They are *network properties* that appear exactly when the elements of some substrate are suitably organized, when they stand in certain relations to each other. This sense of an emergent causal property, the sense in which it is understood as consisting exactly of an organizational feature of a substrate, implies reducibility.[6] So we are not dealing with a representational property that is not reducible to primitives of physics. What the difference between analog and digital seems to come to for Lewis, then, is a question of whether reduction is necessary or not.

In this paper Lewis also rejects Goodman's definition of 'analog'. He produces two counterexamples to show that mathematical continuity — denseness of representational types — is not essential to forms of computation accepted as analog. The first is an analog computer whose input is controled by a potentiometer with differentiated click-positions. The rest of its operation is the usual analog computer setup. He says this computer is analog even though input signal magnitudes are quantized, because numbers are being represented by electrical resistances. His second counterexample is a complicated device for multiplying by means of pellets draining from buckets through spring-loaded valves. Here the signal is disjoint and differentiated, and yet we would call this analog multiplication because, again, numbers are being realized as physical magnitudes, weight perhaps, simply describable in the language of physics.

It is true that engineers would have no trouble calling either of these devices analog, although they propagate amplitude-quantized signals. They are both still continuous function

[6]See P. M. Churchland (1989, 51) for a discussion of this sense of emergence along with the other sense that specifies an emergent property as one that does *not* consist in an organizational property of a substrate and hence is not reducible. It should be noted however that Churchland's understanding of property reduction does not involve deducibility. "Formal considerations alone guarantee that, for any predicate F not already in the proprietory lexicon of the aspirant reducing theory T_N, no statements whatever involving F ... will be deducible from T_N. The deducibility requirement would thus trivialize the notion of reduction by making it impossible for any conceptual framework to reduce any other, distinct conceptual framework." (1989, 51).

computers. The implementations of mathematical operators in both systems, op amps in the first and spring-loaded valves in the second, are still continuous-time processors.

But what if they were not, what if processing elements *could* operate only with quantized time-sampled signals? Then we would call the computers either analog or digital I think, depending on whether we thought of the discretized signals and processes as implementing a code. Mechanical digital computers must have had something of this character — gear wheels with teeth, some large and some small. We would have lost mathematical continuity but we would still have processing that depends on causal proportionalities among angles of rotation and so on, computational elements whose output was proportional to their input. Here the gear teeth would also be thought of as realizing number symbols. The point about Goodman's criteria for notational systems is that we *can't* think of continuous signals as instantiating symbols. Analog computation *has* to depend on causal proportionalities because there are no elements in an analog signal[7] for syntactic rules to get a grip on.

Fodor and Block's (1973) reply to Lewis's paper also rejects the continuous/discrete characterization of the analog/digital distinction, because they think that in this form the distinction will not carry the weight they want it to carry — i.e. the full weight of the distinction between cognitive and noncognitive processes:

> From the point of view of the fundamental organization of the machine, the distinction between continuous and discrete systems is a very trivial one. Any continuous system can be rendered discrete simply by reducing its sensitivity (e.g., by incorporating threshold elements). Thus the theorist who wants to rest the weight of the cognitivism issue on the continuous/discrete distinction is likely to be embarrassed by the existence of pairs of machines which obviously operate fundamentally in the same way, even though one is digital and the other analog on the present way of drawing the distinction. (Fodor and Block, 1973, 8-9)

But the continuous/discrete distinction really is not so trivial. A discretized analog machine will be like the geared digital machine I spoke of earlier: It will be possible to see it as a symbol-*using* device, whose inferential rules are implemented in angles of rotation. With an analog machine whose representational processes are not digitized, we do not have that option.

Fodor and Block, and Demopoulos after them,

> wish to be able to say for some principled and not merely pragmatical reason that some components of the cognitive system are more accurately described in biological (and ultimately physical) terms, while others demand a computational

[7]Again, it is always possible to introduce thresholding devices into analog circuits but analog computation occurs precisely without the computational use of such thresholds.

> description. And we would like this distinction to coincide with the one between analog and digital. (Demopoulos, 1987, 83)

But *why* do we want "this distinction to coincide with the one between analog and digital"? For reasons, as Demopoulos admits, having to do with disciplinary territory. It is being assumed that cognitive processes, representation and inference, must be language-like processes — must be like language thought of as a formal system. Representation must be symbolic, and inference must be directed by rules. And 'rule', here, is not just another name for a formal description of a nonformal process. In a cognitive system, in Fodor's sense of it, rules are *deployed* to control the behavior of the organism (Fodor and Block, 1973, 4); cognitive processing relies on the analysis of signals into language-like constituents recognized by these rules. (Recall how in section I.5 the order of signals was representationally relevant only in relation to divisions of the bit-stream into code words.)

Given the assumption that cognition depends on code, Fodor and Block do not want it to be possible for analog and digital machines to "operate in the same way". A discretized analog machine, if it were seen as digital, would have to be thought of as a symbol-using machine — a representing machine — whose inferential processes nonetheless were not rule-governed but physical/causal in an immediately transparent way. If inferential processing of representations can be accomplished without the intermediacy of rules themselves implemented as symbol-expressions in code, then cognitive psychology loses its special science autonomy. Human psychology becomes a division of biophysics. Machine psychology becomes a division of engineering (which it is anyway). Cognitive science would lose its founding assurance that logic-using systems constitute a natural kind. So 'analog' and 'digital' must be defined in a way that divides logic-users from non-logic-users. [illegible]

Lewis's way of drawing the distinction won't do because, as we saw, the causal properties of an organization of primitive magnitudes are, in the end, reducible to physical properties. So Fodor and Block say it this way: a computing device is analog if its input-output behavior instantiates a physical law, and digital otherwise. An example of the instantiation of a law is an analog component set up in such a way that relations of resistance, voltage and current — described by Ohm's Law — do the computational work. That is, those of the machine's state transitions that are representationally relevant fall under Ohm's Law. The state transition map just is Ohm's Law. [illegible]

The same computation done in a digital machine will also have a physical description: the state space description of its electrical configuration. Will the machine's transitions in state space instantiate a law or laws? Fodor and Block say no. Its description will be a true empirical generalization not a law. The difference they admit seems to be a matter of degree.

Demopoulos argues that none of this will work, because transitions in the electrical state space of any individual digital machine will always be subsumable under some combination of physical laws. To get to a principled distinction between cognitive and noncognitive computational systems, we have to talk about classes of machines:

> A class of machines is analog if its state-transition functions are subsumable under a physical law. A machine-class is digital if its computational behavior can be captured only in computational terms — if there is no single set of physical principles which captures the generalizations governing the behavior of all the devices in the class. (Demopoulos, 1987, 84)

By "computational behavior" Demopoulos means behavior described in terms of its task domain. We may want to say different models of digital computers are multiplying 23 x 47. The state transition functions of all these machines will not have anything in common that can be described by means of a physical principle. "Multiplying 23 x 47" will not be a projectible predicate.

This is a version of cognitive psychology's argument from multiple instantiation. Psychology, it is argued, wants explanation over such functional entities as 'believing it will snow' or 'knowing it is time to go home'. Functionalist kinds — psychological or computational — are physically realized in such diverse ways that it would be impossible to find a common physical description. So functionalist kinds cannot be reduced to physical kinds. If we want to talk about what a class of computers is doing, or can do — its functions — then we have to give it a computational description. For the class of digital computers, Demopoulos says, this task-domain description will be the only *sort* of computational description we can give.

A consequence of this way of drawing the analog/digital distinction is that analog computers will also be classed as digital to the extent that we view them as representing and computing. There is more than one way for analog computers, also, to multiply 23 x 47; and any variation in their component materials will require that they be described by different physical laws. So all classes of computers will be digital, inasmuch as they realize task-domain functions; and all singleton computers will be analog, inasmuch as their computational state-transitions are law-describable as well as rule-describable. An odd consequence.

"It would be worthwhile if the analog/digital distinction were drawn in a way which illuminated the computationalist's interest in the computer analogy", Demopoulos says (1987, 80). He himself draws it in a way that certainly does not make the appropriateness of the digital computer metaphor less plausible; indeed he designs a distinction which merely brings the analog/digital distinction into line under the physicalist/functionalist distinction. This is a pragmatic rather than a principled move because he has disregarded the features of analog computers that suggest an alternative to logic-using computation over symbols.

III. Pylyshyn and Symbolic Systems

> It was all over by 1970. The field of computers came to mean exclusively digital computers. Analog systems faded to become a small sub-part of electrical engineering. The finish was spelled not just by the increased speed and cost-efficiency of digital signal processing systems, but by the discovery of the Fast Fourier Transform, which created the field of digital signal processing and thus penetrated the major bastion of analog computation. The transformation of the field is so complete that many young computer scientists hardly know what analog computers are.
>
> The main significance of this issue, with its resolution, was to help create the discipline of computer science and separate it from electrical engineering. Its effect on AI lies mostly in the loss of an analytic point of view, in which the contrast between analog and digital computation is taken as the starting point for asking what sort of information-processing the nervous system does. (Newell, 1983, 195)

Allen Newell's 1983 account of intellectual issues in the history of artificial intelligence makes the interesting claim that the issue of continuous versus symbolic systems was the issue that, from about 1955, resulted in the institutional separation of artificial intelligence from engineering and cybernetics. Taking a position on this issue would result in coordinated choices on four other issues: (1) pattern-recognition versus problem-solving as research areas; (2) learning versus performance; (3) parallel versus serial processing; and (4) neurophysiology versus psychology as background from which problems are drawn.

> Continuous system folk ended up in electrical engineering departments; the AI folk ended up in computer science departments. (It must be remembered that initially computer science departments were almost exclusively focused on software systems and almost all concern with hardware systems was in electrical engineering departments.)
>
> Adopting a class of systems has a profound influence on the course of a science. Alternative theories that are expressed within the same class are comparable in many ways, but theories expressed in different classes of systems are almost totally incomparable. Even more, the scientist's intuitions are tied strongly to the class of systems he or she adopts — what is important, what problems can be solved, what possibilities exist for theoretical extension, and so forth. Thus, the major historical effect of this issue in the 1960s was the rather complete separation of those who thought in terms of continuous systems from those who thought in terms of programming systems. The former were the cyberneticians and engineers concerned with pattern recognition; the latter became the AI community. (Newell, 1983, 198)

Classical cognitive science takes its computational bearings from computer science, which evolved with AI in the symbol-system camp. Looking at what I have called the rationalist line on the analog/digital distinction, it is helpful to keep these institutional alliances in mind.

Pylyshyn is a computational psychologist, a cognitive scientist — one of those who adopt "the programming system itself as the way to describe intelligent systems" (Newell, 1983, 198). His 1984 monograph *Computation and Cognition* offers a definition of 'analog' that is like Fodor and Block's, and which serves to drive a wedge between what are considered fixed, stimulus-bound, biological components of cognition and those that may participate in rationality. Before I outline Pylyshyn's notion of analog processing I will look briefly at the more general approach that informs it.

III.1 Logical functionalism

Functionalism is an approach to explanation which defines its explanatory kinds over causal relations rather than physical structures. Freudian psychology is an instance of a functionalist theory, in which 'ego', 'id' and 'superego' are defined in relation to their causal effects. They are hypothetical *functions* from some input to some output, which have not been localized to any organ in the brain although it is assumed they are neurally implemented in some way. [illegible]

Pylyshyn's computationalism is a logical functionalism because his explanatory entities are identified with the elements of formal languages or, more precisely, with the elements of those formal languages which are also programming languages. Any formal language has these characteristics:

(1) a finite lexicon of logical primitives — discrete, context-independent elementary *symbols;*

(2) well-formation *rules* and inferential operators which compose and re-order strings of these atomic symbols in purely syntactic ways that are nonetheless guaranteed to be reference- and truth-preserving;

(3) a *semantic function* from symbols to referents which is combinatorial — which guarantees that the meaning or reference of any string is a function of the meaning or reference of its parts.

Pylyshyn's logical functionalism, then, is a psychological theory which attributes certain kinds of behavioral regularities to the causal interaction of symbols and rules. An explanation of a psychological event would take the form of a program specifying a series of procedures performed on symbols. In this context 'program' has the same explanatory status as 'symbol' or 'rule'. It too is a functionally defined entity. Like id, ego and superego, *symbol*, *rule* and *program* are not localized to any identifiable structure or process in the brain: they are said to be implemented or instantiated or realized in neural structures and procedures, but the details of this instantiation need not be considered. Indeed they cannot be considered, because they are not known. Cognitive psychologists are in the Kantian position of trying to extrapolate from experimental data to the rules and representational structures which must be thought to be responsible for that data.

There are many psychologies, and functionalism is an approach that lends itself to many systems of functional kinds — we can think of Gestalt theory, Reichian orgone theory and even well-developed theologies in this light. Cognitive psychologists are motivated to the choice of *logical* functionalism by the class of behaviors that most interest them. They develop a logical or linguistic functionalism because they are interested in rational behaviors mediated by language. Rational behavior is thought to be goal-directed behavior arrived at through the intermediacy of inferential operations on beliefs. Goals or beliefs are themselves thought to require internal representation in the form of propositions, because, as is the case with those of our beliefs and goals articulated in natural language, they are thought to refer, to abstract, to have truth conditions and to be in logical relation to each other. They are also thought to have constituent structure: to have parts which may be connected and disconnected and moved around. Consequently they are thought to be productive and systematic. Linguistic productivity — the indefinite variability of what may be said in any language — is given by recursive processes operating on syntactic/semantic units. Our ability to believe just about anything is thought to be given by similar recursive operations over units of belief. Linguistic systematicity — the fact that we can say that eggs lay ducks as easily as we can say that ducks lay eggs — is also thought to be a feature of the internal representations of beliefs.

A psychology which looks for explanations in terms of beliefs and goals which themselves name some external features of a task domain is called propositional attitude psychology or intentional psychology or belief-desire psychology. An intentional psychology could theoretically be neutral as to the form taken by internal representations of beliefs and goals, but intentional psychologies of Pylyshyn's sort invariably posit sentence-like strings, both for the reasons given above and because their explanatory medium and construct — the program — comes in the form of linguistic strings. The internal language posited by such psychologies is called mentalese; and the hypothesis itself is called LOT (language of thought) theory. So Pylyshyn is a belief-desire psychologist and a LOT theorist, and logical functionalism is the methodology that makes these theories computationally workable. "Sentential predicate-argument structures," Pylyshyn says, "are the only known means of encoding truth-valuable assertions for which there exist a partial calculus and at least certain combinatorial aspects of a semantic theory" (1984, 196). We have well-developed theories of deductive inference defined over formal elements of a language. We know how to implement formal languages in lower-level languages and ultimately in hardware processes. Digital computation yields an existence proof for the physical instantiability of formal languages. But what exactly are symbols and rules and programs, functionally defined?

III.2 Functionalist symbols

We know the work 'symbol' is doing in Pylyshyn's theory: it is the theoretical atom: the logical primitive, the epistemological and computational foundation. (Pylyshyn cites Goodman on this.) A symbol must have disjoint types and finitely differentiated tokens. It must be discrete, non-fractionable, and context-independent. But what does it mean to say a functionally defined entity is discrete?

A symbol in a digital computer is something we can know top-down, bottom-up, both coming and going. Seen at the hardware level, as just this temporary setting of flip-flops in a register, an instantiation of a symbol is a structure not a function. It is some particular inflection of a physical material. This identifiability of token symbols with a material state gives the notion of a digital symbol a clarity it does not have when we are talking about human cognition. We know what it means for digital symbols to be discrete: it means a digital machine's switches are bistable devices with no intermediate states. But where in the massively interconnected structure of the central nervous system should we imagine a similar discreteness? It seems very unlikely that the firing decisions of individual neural synapses should be identified with instantiations of logical atoms, since there are more than 10^{14} distinct synapses in the human brain (see P.M. Churchland, 1989, 209-10). If symbols are thought to be instantiated by cell assemblies (Hebb, 1949), do we say an activation pattern *a* 'is discrete' if it results in decision *A* rather than decision *B* somewhere downstream? That would allow connectionist patterns of activation to be thought of as realizing symbols. But all patterns of activation that result in decision *A* will be called symbol *a* in a functionalist identification. And if the activation patterns which instantiated *a* on one occasion instantiate *b* on another occasion when surrounding cells are differently activated, we have the same local state instantiating different symbol types at different times. This violates Goodman's disjointness criterion.

My point is that the status of a functionalist symbol is unclear as long as neural realization is unknown. There is no problem with external description in symbol terms. We identify symbol *a* with whatever results in decision *A*, whatever the local circumstance. This is perfectly good functionalist description. We can use it to write programs in which decision *A* is always followed by decision *E* given the copresence of decisions *C* and *D*. Our program is a formal description of the causal relations in a sequence of events — it is a model, in Rosen's terms. Like any model it can be run on a digital computer. The problem with functionalist programs whose neural realization is unknown comes when we want to say the nervous system is a computer executing — literally running — the program we have written. There is nothing in a functionalist definition of symbols or rules or programs that licences us to do this. And there is nothing in a functionalist description that requires it. We have seen that analog computation can be given a formal description without having to be seen as itself employing symbols. A claim that the human brain is using the symbols named in a program describing it seems so manifestly to overstep the functionalist mandate that one wonders if Pylyshyn really wishes to make this claim.

The difference between weak and strong equivalence of program and cognitive process, Pylyshyn says, is a matter of detail. A program is weakly equivalent if it realizes the same global input-output function as the process modeled:

> Clearly, if the computational system is to be viewed as a model of the cognitive process rather than a simulation of cognitive behavior, it must correspond to the mental process in greater detail than is implied by weak equivalence. On the other hand, it is equally clear that, because computers not only are made of quite different material than brains but, through details of realizing particular operations (for example, certain register-transfer paths or binary mechanisms and bit-shifting operations), differ from the way the brain works. The correspondence between computational models and cognitive processes appears to fall somewhere between these two extremes. (Pylyshyn, 1984, 89)

Pylyshyn's suggestion for the appropriate level at which to define strong equivalence is the level of the algorithm — a level which specifies the steps a system is to follow. Elementary steps in the algorithm are to be formally isomorphic to primitive operations of the cognitive architecture — and primitive operations in the architecture are those defined over the most elementary cognitive units, which are identified with symbols. Strong equivalence of an algorithm written for a digital machine and the procedure followed by a computing brain would imply that brain and machine have the same functional architecture. Whether equivalence of functional architecture would rule out connectionist realization of elementary symbols of the language of thought remains unclear. I will come back to the notion of functional architecture, but first I would like to look more closely at what a functionalist version of a *rule* is like.

III.3 Functionalist rules

An algorithm is a sequence of well-defined rules or procedures; strong equivalence of digital algorithms and biological cognition will imply that cognitive creatures perform computational operations by means of the same steps taken in the same order. Again, these steps are functionally not physically defined:

> Computational operators deal only with computational events. Although there are physical-event tokens which underlie instances of computational events, the regularities among the computational events are not captured by physical laws, but by computational rules instead ... Operators map computational states onto computational states, or symbols onto symbols, but not physical descriptions onto physical descriptions. That is why physical *descriptions* are irrelevant in characterizing computational operators, though, of course, they are relevant to showing how primitive operators are realized in a particular system. (Pylyshyn, 1984, 170)

That 'rule', 'operator', and 'procedure' are often used synonymously follows from the functionalist nature of their description. 'Rule' is a term that is at home in logical calculi, where rules of inference are prescriptive rather than descriptive. It is also at home in grammars, where its status is mixed or ambiguous. The rules of grammar are taught, and thus prescriptive; but when grammars are studied as an empirical domain, as is the case with Chomsky's generative grammar of natural languages, grammatical rules are descriptions of regularities assumed to lie behind the construction or comprehension of sentences having constituent structure. It may or may not be thought that these grammatical rules are also inscribed as conditional sentences in mentalese, somewhere in the human language system. A functionalist characterization is compatible with either a descriptive or a prescriptive reading of 'computational rule'.

High-level programs for digital computers are written in the form of procedures, simple commands and repetition statements. These *can* be seen as rules in the prescriptive sense: they are orders telling the machine what to do. Instantiated at the hardware level they are, of course, causal sequences, a series of events which reset switches and thus reorganize the physical machine. Here the relation of the written program to the machine in operation *can* also be seen as descriptive — a modeling relation. This interesting overlay of the prescriptive and descriptive senses of 'rule' at different conceptual levels in a digital computer is made possible by the fact that digital machines are string processors some of whose instructions are programmed and so enter the machine in the same form as the sentences to be processed. Programmed rules (as opposed to those implicit in hardware) are sentences designed to have logical effects on other sentences. We see them as causes because we are thinking of them as task initiators in an intentional domain.

They are commands, orders which are also causal determinants. A parallel in cognitive terms would be what happens when someone says to us "Give me the sum of your birth year and your telephone number", and we do. Their sentence, somehow input and implemented in personal mentalese, i.e. realized in personal wetware, programs us and results in a computation.

When we are dealing with digital machines, then, we seem to have a process that can be seen as simultaneously rule-programmed or *rule-using* — at the intentional or functional level — and *rule-describable* — at the structural or physical level.We have seen that this is not the case with analog machines, which are physically configured — 'programmed' — without the intermediacy of coded rules. Analog machines 'obey' mathematical rules inasmuch as their materials have been chosen for their proportion-preserving causal properties, but they cannot be seen as responding to internalized sentences. They are rule-describable but not rule-using. What we noticed above, however, is that digital machines, described at the hardware level, are rule-describable rather than rule-using as well. The description of a computational machine as rule-using, then, presupposes two things: (1) the description must be a description at a level higher than the hardware level; and (2) the machine must be programmed at least partially by the intermediacy of coded sentences.

As we will see in chapter IV, a connectionist criticism of LOT theories such as Pylyshyn's is that a description of human cognition as rule-using is implausible and unnecessary. Pylyshyn's reply would emphasize the sorts of cognitive instance where humans very evidently compute by means of sentences. Rationalist tradition thinks of intelligence as rationality, and rationality as inference. So paradigm instances of cognition, for a rationalist, are those where an issue is debated internally — evidence marshalled, implications explored, and conclusions reached. Our experience of this sort of episode is an experience of 'hearing ourselves think'. Pylyshyn would not be so naive as to think the phenomenal experience of cognition is a window onto the cognitive substrate, but, like Fodor and other LOT theorists, he does wish to support the possibility that creatures with speech are able to use sentences to *program themselves* .

With digital machines, we are the programmers, we input the rules, which are 'obeyed' when the machine does what we tell it to do. We have this masterful relation to our machines because we design them. Generally we speak of the relation of the descriptive levels available with digital machines in a sequential way that supports this sense of mastery: we write a program in source code, i.e. in a task domain language; we then implement this upper-level language in an assembly language of some kind; and we then compile this intermediate language into absolute code suitable for direct execution by the central processor. The chain of command flows top-down from program to hardware realization — the linguistic seen as commanding the material in a fashion that's downright theocratic. But rules in the head cannot, of course, be implemented in this order: they are not linguistic before they are physical. As rules in the head they are immediately physical; their linguistic description will be a functional description of some structure which is already a physical structure. Without temporal priority, intention-level description does not so easily seem description of a command structure. When we have both top-down and bottom-up descriptions, and they are descriptions of the same event seen simultaneously as material and linguistic, there is no special reason to think of the linguistic (or voluntary, perhaps) as commanding the material. There is as much reason to think of the material as self-organizing, and the linguistic as self-organized along with it.

III.4 Analog as functional architecture

The functional architecture of a digital computer is a description of the machine at its lowest functional level. Because these functions are thought of as directly realized in the physical machine, they can be given physical descriptions. We can explain their workings either in terms of physical laws describing material properties of components, or in terms of the ways components are configured. This is the level where a bistable device is called a switch and a certain configuration of bistable devices is called a register — both descriptions being descriptions of computational function at its lowest level. More general aspects of a computer's functioning are also part of its functional architecture — the way memory and addressing are organized, the operation of control functions and user interfaces, and so on. A description of the functional architecture of a computing system is an engineer's description of parts and their functions and configurations. It can be given a mathematical description — and although there is still room for a variety of hardware realizations — these mathematical descriptions can be seen as descriptions of constraints on the computational behavior of the system. While these constraints must be thought of as having computational effect, they are not themselves modifiable by computational (read 'linguistic') means. They are computationally impenetrable. [illegible]

Pylyshyn uses a similar notion of cognitive impenetrability in relation to human computation. The notion of cognitive impenetrability is Pylyshyn's device for attempting an empirical boundary between (what amounts to) the physical and the mental. A cognitive process [illegible] will belong to functional architecture if it is fixed and stimulus-bound, or modifiable only by non-cognitive means such as maturation or blood chemistry. It will be called a cognitive or symbolic or representational process if it is modifiable by linguistic or rational or logical means. In the psychology lab we can apply tests to discover whether instructions or information change the outcome of a process: if so, it is not part of the functional architecture, Pylyshyn says. The boundary between functional architecture and rational processes obviously is not a boundary that can be drawn within a hardware description of a computational process, though: the modification of a synapse as a response to recency of use looks exactly like the modification of a synapse due to the input of a sentence *S* of experimenter instruction--when it is in fact be the same event. For Pylyshyn this is irrelevant. What he (like Demopoulos) wants is a separation of descriptive domains, descriptional formalisms:

> The distinction between functional architecture and representation-governed processes ... marks the boundary in the theory between two highly different kinds of principles — the kind stated in terms of semantics, in terms of properties of the *intentional objects* or *contents* of thoughts or other representations — and those that can be explained in more traditional, functional terms. (Pylyshyn, 1984, 262)

This is, in fact, just Demopoulos' distinction between "components of the cognitive system ... more accurately described in biological (and ultimately physical) terms", and components which "demand a computational description". And Pylyshyn, too, wants his distinction to "coincide with the one between analog and digital". His definition of 'analog' is wider than Fodor and Block's, however. For Pylyshyn analog processes are not only those whose regularities are consequences of direct instantiations of physical laws, but also any processes for which a system "behaves as it does because of the kind of *system* it is, because of the way its nervous system or circuits are structured to encode information and to process these codes" (1984, 212).

So the boundary between 'digital' and 'analog', for Pylyshyn, is exactly the boundary between processes which are thought to require a cognitive or task-domain or representational description and those that can be given a physical or configural description. Pylyshyn does not exactly stint the computational nature of analog processes: he grants they are "a form of complex process that avoids both the logical step-by-step character of reasoning and the discrete language-like character" of the representation typical of cognitivist theories. They deliver their output in one bold step and "can be achieved within material causal systems, just as the syntactic alternative can" (1984, 199). But he considers there is nothing to be gained from considering that human representation might employ analog forms:

> We have no idea what explanatory advantage the proposal is supposed to gain us, because (a) there is nothing equivalent to a calculus, or even principles of composition, for what are called "dense symbol systems"; so we do not know what can be done with them, how elementary symbols can be combined to form complex representations, or the kinds of regularities that such symbols can enter into; and (b) there is no idea of a physical mechanism that could instantiate and process such symbols. The "analog computing" view ... does not do this; it is not a symbolic system at all, inasmuch as none of its regularities require

or Chapter

> explanation at a separate symbol level, as is true of digital computing. (Pylyshyn, 1984, 199)

We want human cognition to have a "sentential-representation format" because formalist ideas of the sort that are native to twentieth-century logic and mathematics are "the only scheme we have for capturing normative patterns of reasoning" (Pylyshyn, 1984, 198).

Where "normative patterns of reasoning" are the paradigm of intelligence in cognitive creatures, other discriminations tend to follow. Pylyshyn's logic-based view of knowledge leads him to think of any process as unintelligent just insofar as it is non-digital. The operations of functional architecture, which most likely will turn out to include learning and development, emotion, mood, psychopathology, motor skills, early stages of perception, and sensory store, will — when they can be said to involve representation at all — involve non-articulated, continuous, or "wholistic" representation that disqualifies them from those combinatorial possibilities of systematicity and productivity that mark anything we want to call a thought or belief. It follows that perception will be seen as intelligent just to the extent that it involves inference, and it will be suspected of involving inference just to the extent that it is seen as intelligent.

But cognitive units — 'ideas' in the old-fashioned parlance — must come from somewhere, at least for Pylyshyn, who does not wish to be a methodological solipsist. So Pylyshyn's theory provides transducers, those magical devices for inputting the physical and outputting the mental.

III.5 Pylyshyn and transducers

Transducers are part of the functional architecture, that is to say, their operation is thought of as cognitively impenetrable, stimulus-bound and fixed. A perceptual transducer is thought of as *encoding* exactly those (proximal) characteristics of an input signal which, inferentially combined, will yield a description that can be reliably correlated with distal properties of an object. Its biological fixity at this first level of perception would guarantee the reliability of perception, its independence of cognitive influences. A transducer is thought to provide a point of contact between world and mind, and between physical and computational vocabularies, because the cognitive state resulting from the operation of a transducer "happens to be type-equivalent to a physiological state" (Pylyshyn, 1984, 142). It might help to remember, here, the operation of an A/D converter, which accepts a continuous waveform and emits a pulse train taken as instantiating a binary '1'. Any pulse train emerging from the transducer with just that sequence of pulses and no pulses will also instantiate a '1'. There is type equivalence between pulse-pulse-no pulse and '1'. Transduction for Pylyshyn is strictly local: there are neural sites the fixed, biologically-given function of which is to emit coded signals in response to specific physical magnitudes. Transducers may also be internal to cognitive processes. Any change in a belief must be arrived at by rational means, so any alteration of functional architecture which results in cognitive change must do so by means of transducers. Emotional changes, if thought of as chemical changes, cannot affect beliefs directly, since syntactic compositionality and inferential integrity depend on uninflectable logical atoms. The only way mood or emotion could moderate a belief would be if an emotion-transducer were to produce token propositions able to interact logically with belief-tokens already present (Pylyshyn, 1984, 270). The presence of a specific magnitude of some chemical concentration would have to be *coded* to have cognitive effect. An extraordinary tale, which Pylyshyn apparently does not find counterintuitive.

There is, incidentally, what I take to be a major hedge in Pylyshyn's account of topological property transduction. If a perceptual transducer is to qualify as an analog or functional architecture process, then it must produce *atomic* symbols and not symbolic expressions. It is difficult to know how such atomic transductions will not lose topological information. Pylyshyn's solution is to say that transducers must signal their identity or their relative location in such a way that global properties may be reconstructed by combinatorial means. "The location (or other unique identifier) of the source of a local code must be available ... This information might even be signaled topographically, by the actual location of the neural connections from the other transducers" (Pylyshyn, 1984, 163). It would certainly make sense to have topological information signaled topographically, but as soon as we have the physical configuration of the machine given computational importance in this way, 'analog' and 'digital' seem to drift into register: a feature of the functional architecture is at the same time a feature of the code. This, to the connectionist, is just as it should be.

IV. Thermodynamic Functionalism

> The new wave of neuroscience-inspired AI contains, of course, a commitment to highly parallel network structures. The degree of parallelism is in the millions, and computing elements in the network have modest powers; in particular, they are not computers with their own local symbols. In the new structures, computation must be shared right down to the roots, so to speak. The interaction cannot be limited to communicating results of significant computations. Furthermore, the communication media between elements are continuous signals, and not just bits. However, unlike the earlier work, these new computational systems are not to be viewed as neural nets; that is, the nodes of the network are not to be put in one-to-one correspondence with neurons, but, rather, with physiological subsystems of mostly unspecified character. (Newell, 1983, 220)

Connectionist computation has historical and technological links with both analog and digital computers. There are connectionist technologies which are called analog by their designers, and others called digital by their designers. But I will argue that connectionist computation is a new, third, kind of computation, one whose links with neuroscience make it more than a synthesis of its two forebears. Its most important carryover from analog computation is the form taken by its cultures of description, which are not logical/linguistic but mathematical/physical. This is so even where the connectionist processes described are discrete; and this interesting turn of events may in the end allow us to look at digital computers, and their codes, in new ways.

In what follows I will give connectionist technologies a brief introduction, describe one neuroscientist's vision of the brain as a connection machine, and outline some of the epistemological possibilities of this sort of vision. I will argue that the representational states of brains (and of connection machines to a lesser degree) may be thought of as having a pre- or sub- or non-linguistic semantics; that the mathematical/physical languages of description allow this sort of content to be conceived of in psychologically interesting ways; and that non-linguistic content becomes plausible when representational states have intrinsic content.

IV.1 Intrinsic content

Analog computers, like digital computers, are devices whose computational states represent because we have designed them to do so. Just what is taken as being represented at any time will depend on an interpretation function we are applying to the terms of an uninterpreted formalism, often by means of automated output interface devices. It may be anything we like, as long as the machine's inferential systematicities and our encoding/decoding systematicities preserve representational relevance over physical changes in the machine. In other words, the representational states of analog and digital machines have extrinsic content--content which is supplied by the machine's users, in the same way it is supplied to pencil marks or morse clicks. That it is extrinsic is obvious to the extent that it is arbitrary.

Creature representation, on the other hand, is neither arbitrary nor extrinsic. The computational states of a cat watching a bird are what they are precisely because they are the computational states of a cat watching a bird. They are the states of a cat's brain; a seal's brain will not have just this sort of connectivity between hypothalamus and visual cortex and motor cortex. There could be other sorts of brains than there are, but they would have to be part of other sorts of bodies than there are. Brains co-evolve with bodies, they are the brains of precisely the kind of body they organize: they are, in some encompassing way, the brains of bodies evolved on precisely this kind of planet, with this gravitational force and this rotational period; and they are, individually, in some entirely specific way, the brain of some individual creature with a particular genetic make-up, a particular developmental and experiential history, and a particular momentary situation. They are organs which provide their creature with flexibility of behavior, but they are not general-purpose machines.

If we were perverse enough, we could, of course, use them as general purpose machines. Given a detailed enough map of the causal interrelations of parts of a cat's brain, say, and given microelectrodes accurately enough implanted, we could use the causal systematicities of the animal brain to compute mathematical functions or to provide a coarse-grained functional analog of some other physical system. Like any other causally systematic physical processes, brain processes could be supplied with extrinsic representational content — content read into physical processes.

Intrinsic content would be the kind of content a cat's representational states have as a consequence of functioning in and for the cat. Both conscious and nonconscious cat-states can have this sort of content, and they will have it in virtue both of the materials and of the organization of materials of the physical cat-brain. I am assuming an evolution-based causal semantics of some sort here — cat functional architecture will be something like truth-discovering and truth-preserving for evolutionary reasons. This will apply both to hard-wired representational capacities and to flexible capacities modifiable with experience.

I have said that the representational states of neither analog nor digital machines have intrinsic content of this sort. Our ability to see them as representing at all is derived from our own primary, fully intrinsic representational ability, which allows us to design interpretation functions for formal systems or to set up analog components in such a way that their computational results will interest us. So both analog and digital computers have a crucial inadequacy as pictures of creature computation. They are both pictures of physical systems whose states are subject to systematic transformations that may be read as having representational relevance, but neither of them — if we rule out programmer-gods — provide any sense of what makes a brain-state a representational state in the first place. The most important thing about connectionist computational architectures is that they provide, not yet a picture, but a hint of a possibility of a picture, of how the states of a creature's central nervous system might be able to have intrinsic content.

IV.2 Connection machines

> There is a reasonable chance that connectionist models will lead to the development of new somewhat-general-purpose self-programming, massively parallel analog computers, and a new theory of analog parallel computation. They may possibly even challenge the strong construal of Church's Thesis as the claim that the class of well-defined computations is exhausted by those of Turing machines. (Smolensky, 1988, 3)

[illegible]

Sometimes called synthetic neural systems or neuromorphic computers, connection machines, like ordinary analog or digital machines, are given mathematical descriptions which can then be implemented in different ways. Hardware development of physically parallel machines is complicated, and a number of the more famous connectionist designs have been simulated on von Neumann architectures. Parallel machines also may resemble digital machines in having bistable switching elements whose operation is modeled in discrete mathematics. Or they may resemble analog machines in having continuously varying computational nodes. Both sorts of circuit have been implemented in very large scale integrated circuits on chips or wafers.

Connectionist nets with bistable switches at the nodes may, like McCulloch and Pitts' 1943 design, execute boolean functions. Their operation may be deterministic, as in certain kinds of Hopfield nets, or it may be stochastic, as in Boltzmann machines. Processing operations may be synchronous or asynchronous. There are designs in which each processing element functions as a miniature digital computer, with router, memory, control unit, accumulator and synchronization by an external clock. In chip implementations, the bistable elements are called LTFs, linear threshold functions, which are step functions, and which are identified with Turing machines.

Hinton and Sejnowski's 1983 Boltzmann machine is one of the connectionist designs simulated in digital hardware while a chip realization is being developed:

> A Boltzmann machine is a system of stochastic binary units in which the computational processes are *iterative* and *local* (that is, they concern the *repeated* mutual adjustments of *neighbouring* units). The behavior of an individual unit is

partly stochastic and partly determined by its weighted connections with its neighbours ... in the basic Boltzmann machine, the excitatory/inhibitory weights on the various connections are *fixed*. An optimal global equilibrium-state (more accurately: an optimal probability-distribution, made up of the probabilities of co-excitation of many pairs of neighbouring units) is obtained by a process comparable to that of minimizing the global energy of a physical system. (Boden, 1988, 218)

("life game")

The settling process Boden describes takes place in response to some input across an array of input nodes which are designed to respond to the presence or absence of specific input properties. The computational task of the machine is to arrive at an output decision which from our point of view is a correct response to the input. It can only do so by responding to systematicities among input features, and these systematicities can only take causal effect if there is some mechanism which responds differentially to various distributions of feature values. The mechanism provided by a Boltzmann machine is a system of weighted connections among middle-layer nodes: the machine will be wired in such a way that the presence or absence of a feature *along with* the presence or absence of the other features will determine processing output. No feature can determine an outcome by itself; its computational effect will always depend on the simultaneous copresence of other features. Excitatory and inhibitory weightings of connections will ensure that the copresence of certain features has more computational effect than that of others. Since each node is a threshold device, and since it is connected to a number of others, there will be a period of uncertainty as excitation propagates through the net. The node will respond when incoming excitation values surpass its threshold value, and fail to respond when they do not. Eventually the whole net will sort itself out, with some of the bistable switches remaining on, and some remaining off. This 'settling' of the net will be the machine's computational result. The process of settling into a stable state is likened to a thermodynamic process because Hinton and Sejnowski were able to describe it by means of the Boltzmann equations of thermodynamics:

> These represent the temperature of a large volume of gas as an overall energy-equilibrium obtained by (binary) energy-transfer between pairs of neighbouring molecules. Thermodynamics deals with stochastic systems: the energy-states of an individual molecule are not predictable, but the energy-distributions of large collections of molecules are ... Accordingly, Hinton and Sejnowski use thermodynamic equations in defining Boltzmann machines and also in defining and justifying a learning algorithm that could be used by such machines. (Boden, 1988, 218)

A volume of gas is a self-equilibrating system, and the net's arrival at a stable configuration of hidden unit switches is similarly seen as a self-equilibration of a physical system by means of energy transfer. As long as the Boltzmann machine is being simulated on a serial digital architecture, though, thermodynamic description is merely a mathematical structure implemented in a code which is implemented in programs running on a digital computer. We have seen that digital computers do simulate physical dynamical systems, and in this case it is simulating a dynamical system which is thought of as itself computing the result also arrived at by the digital computation.

Connection machines implemented using continuous dynamics have processing elements that can be considered operational amplifiers, and their connection weights are implemented by means of resistors. In chip implementations they will be SAFs, sigmoidal semilinear activation functions. Processing elements will demonstrate certain gain ratios, and they may be stochastic or deterministic, synchronous or asynchronous in operation.

Hopfield is credited with inspiring the first analog chip implementations of Hopfield nets around 1982 (Akers, et al., 1989, 142). An analog chip version of a Hopfield net has continuous and deterministic response functions, and its circuit equations are thus coupled differential equations. The processing operations of the chip may also be given a further, dynamical, description. Its connection matrix is symmetrical, and "this symmetry, together with the asynchronous execution protocol, allows the computational dynamics to be described as a relaxation process in which Lyapunov (or 'energy') function E is minimized" (Akers, et al.,1989, 153).

Analog chips have certain computational advantages over chips with bistable transistors. Exploiting the functions, like summation, which are one-step processes in analog circuits, reduces die area requirements and power consumption, because there are fewer computational steps overall. Analog circuits offer lower resolution, but with greater speed and density. Where hardware is used to implement learning networks, that is, where connection weights are modified as a result of error feedback, 'analog depth' accommodates incremental adjustments, which makes the system more sensitive and allows it to settle into a more global, stable solution. This latter characteristic has made analog chips efficient at generating good, if not perfect, solutions to computationally complex problems such as the traveling salesman problem. Commercial uses have included voice recognition and vision systems. Like analog computers, analog connection machines have no trouble with nonlinearities, and they can be used to model the dynamics of nonlinear systems such as fluid flow. The connection machine is itself thought to be a nonlinear dynamical system, which evolves toward solution states that may be called attractors (P.M. Churchland, 1989, 206).

Connection machines, then, have a historical dependency on the digital technologies we use to explore their capabilities. Their overall complexity is more like the complexity of digital hardware than it is like the relative simplicity of analog computers. Some connection machines incorporate digital systems of memory and control; some connection machines involve binary states satisfying discrete forms of mathematical equations. And connection machines are like analog computers in being parallel processors which evolve toward solutions, and sometimes in having processing components which implement continuous functions. The more important similarity, as I have said, is the sort of description given processing states of the machine — descriptions given not in terms of syntactic elements and rules, but in terms of system state space vectors, thermodynamic equilibria and the like.

How are connection machines also unlike both analog and digital machines? What could justify Smolensky's guess that they may lead to "a new theory" of computation?

(1) 'Settling', 'relaxation', 'cooperation' — the connectionist style of computation as typified by the Boltzmann machine — allows weighted consideration of many features at once. Where multiple layers are present, the cascading of activation from one layer to the next provides for something like a registration of higher-order relations among features as well: nodes of the first hidden unit layer can register relations among relations of first-order features. After five or six such mappings from layer to layer, activations arriving at output nodes may reflect responses to input features which are very abstract indeed. This logical depth of relativized complexity is given in what can be seen as a single step, with no need for an external control function to schedule the individual transitions.

(2) The scale of parallelism of connection machines is envisaged by ultra large scale integrated chip designers as heading toward a billion connections (Akers et al., 1989, 149).

(3) The processing effects of any component of analog and digital machines are understood in principle — we are able to say what contribution it is making to a computational result. With connection machines we are not certain what contribution is being made by any node. There are empirical ways to try to discover its effect after the event, but we do not design every detail of the computational process. And the failure of a component is not critical in connectionist computing, whereas failures of single components in analog or digital machines may result in a failure of the whole process.

(4) Thresholds in connectionist circuits are thought of differently than they are in digital machines. This is true especially where overall processing is stochastic — where the effect of the individual node's particular binary state determines nothing by itself, and decisions downstream are reached by a kind of overall averaging process. Thresholds, in other words, have statistical not logical effect, and thus we are not as tempted to think of them assigning truth values to strings of symbols in a calculus. Quantized signal-packets are discrete, but their effect is not differentiated with respect to the computational outcome.

(5) The physical organization of a connection machine is not effected by means of input strings of instantiated code which reset switches in order to guarantee the computational relevance of machine states to a represented domain. Instead we have several sorts of hardware organization. The first sort is the design of connections, which may be local or global, which may be one- or many-layered, which may allow for symmetrical or asymmetrical activation, which may be uniformly or differentially weighted, which may allow for feedback as well as feed-forward transmission, and which may have fixed or modifiable gain ratios. Where gain ratios or weights between nodes are modifiable a second sort of hardware organization is effected: the organization of the machine, over time, by means of sample input-and-feedback pairings. The fabric of the machine itself changes with what we're tempted to call experience — a result which is not self-programming, but a sort of self-structuring in concert with given environmental structures. We have a hardware which becomes more dedicated over time — customized by, and hence more attuned to, its conditions.

(6) Connection machines have, from their beginnings (which coincided with the beginnings of analog and digital computers in the 1940s and 50s), been built with some eye to neural plausibility. David Rumelhart of the PDP Research Group makes this motivation explicit:

> Is it possible to somehow come up with a replacement for the language of the computer, and if so can we develop a kind of language that is appropriate, that is inspired, by our understanding of the nature of the brain? Our goal, in short, was to try to replace the computer metaphor of mind with the brain metaphor of mind. (Rumelhart, 1989, 111)

A

IV.3 Holonomic brain theory

> An analog parallel model of computation is especially interesting from the point of view of the present understanding of the biophysics of neurones, membranes and synapses. Increasing evidence shows that electrotonic potentials play a primary role in many neurones. Mechanisms as diverse as dendrodendritic synapses, gap junctions, neurotransmitters acting over different times and distances, voltage-dependent channels that can be modulated by neuropeptides and interactions between synaptic conductance changes provide neurons with various different circuit elements. Patches of neural membrane are equivalent to resistances, capacitances and phenomenological inductances. Synapses on dendritic spines mimic voltage sources, whereas synapses on thick dendrites or the soma act as current sources. Thus, single neurons or small networks of neurons could implement analog solutions. (Poggio, Torre and Koch, 1985, 317; cited in Pribram, 1991, 19)

Can a new theory of connectionist processing lead us to a "brain metaphor of mind"? In von Neumann's day it was thought that the computationally relevant behavior of neurons was their thresholded propagation of spike trains. More recent theories, such as that of Poggio et al., give an analog account of neurons as circuit elements. In any case, the brain does consist of massively interconnected nodes in multilayered arrays. If certain key features of connectionist processing operations are also present — features such as cooperative effect, abstractive cascading, and modifiable connection weights — then connectionist theory may be a good base for the new disciplines of computational neuroscience.

Karl Pribram, the bold and eminent neuroscientist who published his controversial theory of holographic memory in the early 70s, has placed his latest bets on connectionism. In his (1991) he supports what he calls holonomic brain theory by means of notions drawn from connectionist sources. His account claims to be an account of figural perception in the first instance, but wider applications are obviously intended. I am going to recount his story of the computational brain in some detail, both because it is a very energized vision of connectionist possibilities, and because his forms of description — which are entirely nonsymbolic, nonlogical — support my thesis that connectionist computation refines the sense of a cognitive alternative formerly offered by analog computation.

In the preface of his (1991) Pribram says explicitly that one of his motivations for the present work was a desire to update his holographic brain thesis in the light of the emergence in AI of parallel distributed processing architectures.

> These 'neural networks' or 'connectionist' models are similar to OCCAM, a content-addressable computational model that we (Pribram, 1971; Spinelli, 1970) developed in the late 1960s, and stem directly from the content-addressable procedures that characterize optical information processing such as holography (see e.g., Hinton, 1979; Willshaw, 1981). (Pribram,1991, xvi)

That connectionist models "stem directly" from the content-addressable procedures characterizing holographic optical processing is a claim based on mathematical grounds. The representing structures employed by optical versions of connectionist architectures are, in fact, holograms. But the larger vision underlying Pribram's equation of connectionist and holographic processing is a vision of the two processes as forms of pattern matching in which superposed waveforms result in interference patterns that are both computational results and representational structures.

Pribram sees neural events as involving a complex interplay of discrete and continuous processes. The axon hillock is a thresholding device which provides all-or-none propagation of action potentials down axonal fibres which are a nerve cell's output device. Axon fibres tend to be relatively long and they may communicate with fairly remote regions of the nervous system. These impulse discharges or spike trains spatially and temporally segment the results of the dendritic microprocess into discrete packets for communication and control of other levels of processing. Packets are more resistant to degradation and interference than the graded microprocess. They are said to constitute the channels of communication of the processing element.

> Communication via neurons often consists of dividing a message into chunks, labelling the chunks so that they are identifiable, transmitting the chunked message, and reassembling it at its destination. Neurons are labelled by their location in the network. This form of labelling is highly efficient because of the essentially parallel nature of neuronal connectivities. (Pribram, 1991, 7)

So Pribram makes a distinction between neural processing and a communication of the results of processing which also acts to organize computational activity over various levels and locations in the brain. Parallelism, convergence and divergence of axon fibres result in the propagation of spatially and temporally patterned trains of discrete bursts of activation, and Pribram, like von Neumann, thinks of these packets as digital, and as code elements which "communicate orders." Still, the system as a whole cannot be seen as digital because there is no through transmission of digital signals. At every synapse the arriving pulse will decrement into the local slow potential. The digital message is received by being convolved with analog processes. On the far side of the receiving neuron other discrete impulses will be propagated. Incoming pulses will influence the frequency of outgoing pulses, but it cannot be said that incoming pulses are reconstituted beyond the soma — only that their message has been taken under consideration.

When Pribram speaks of 'code' here, what he has in mind is not something primarily language-like. In his (1971) he mentions a number of different characteristics of 'impulse coding' which may have computational effect. They involve both spatial and temporal variations, and may include duration of bursts, overall probability of firing, variations in this probability, incrementing or decrementing of this probability, rate of change of this probability, shifts in latency, the spatial distribution of trains across arrays of fibres, and differences of timing among neighbouring ensembles. 'Codes' can be read into the computational effectiveness of such bursts of activation, just because their elements are discrete. We can say "That pattern of bursts is telling these neruons that *X*", or it is telling them to do *X*. But speaking of it as code is metaphoric and redundant: it is not required in an explanation of its computational effect, which follows directly from the physical organization of the machine. We have no reason to postulate the presence of coded rules operating over these discrete bursts.

Discreteness of computational elements can be computationally important in ways that have nothing to do with symbolization. Pribram sees the packeted nature of axonal discharge as providing a necessary linearity to processes taking place at the synapses. Wave to pulse conversion at trigger zones in the axon is thought to be nonlinear, that is, it is thought to be the result of nonlinear threshold functions. In constrast, pulse to wave conversion — the incoming effect of pulses arriving at junctions — is thought to be a linear function, multiplication by a constant. And the linearity provided to junctional microprocesses by discrete incoming activation packets allows them to be described by means of Huygen's principles of wave propagation, the linear principles which underlie holographic theory. Then operations of filtering, integration and transmission can be descibed by linear differential equations and the "locus of entry of nonlinearities can be identified without jeopardizing the advantages that accrue to the overall linearity of the operation of the brain systems involved in configuring percepts" (Pribram, 1991,19).

Dendrites are the nerve cell's input devices, and Pribram takes the electrical and chemical microprocesses surrounding dendritic junctures with axons or other dendrites as constituting the local circuits which effect actual neuronal processing. These microprocesses are the analog processes described by Poggio et al., processes which involve the release and absorption of thirty-odd sorts of neurotransmitters with different sorts and rates and ranges of chemical effect, the mitigating influence of various enzymes, and various degrees of unmyelinated dendrodendritic contact within a feltwork of dendritic fibres. The overall electrical effect of these processes is the creation of patterns of charge density distribution in the tissue matrix within and between postsynaptic dendritic membranes. These charge densities are temporary microstructures, steady states of polarization amplitude which do not propagate and which Pribram calls slow potentials. Slow potential distributions across ensembles of synapses are, for Pribram, the actual neural locus of computational processing.

> If we focus our attention not on the membranes of single neurons, but upon charge density distributions in the tissue matrix of neurons, glial cells and mucopolysaccharide processes, we can envisage a complex, three-dimensional volume of isopotential contours, topologically comprised of portions of cellular membranes and extracellular binding sites and constantly changing over time. Let us call this volume of isopotential contours or convoluted surfaces a *hyperneuron* (Thatcher and John, 1977, 305; cited in Pribram, 1991, 10-11)

Pribram calls these "volumes of isopotential contours" holoscapes, which may be thought of as standing waves of activation, and which constitute the nervous system's representational medium. So representation, for Pribram, is inherently 4-dimensional it consists of topological, configural, temporarily self-maintaining electrical structures, which are induced by a combination of genetic and experiential factors. Synaptic characteristics, like connectionist weights, are thought to be modifiable as a result of practice, but they may also be modified as a result of non-informational events such as vitamin deficiency or a full moon. There is no hint here of Pylyshyn's wedge between biological and informational causes.

Pribram's guess is that holoscapes — isopotential contours of slow potential microstructure — are the "physical correlates of a percept", and that cortical interference patterns are "coordinate with awareness". His guess has received interesting support from Freeman's connectionist-inspired work on olofaction, which has found that the "identity of an odorant is reliably discernible only in the bulbwide spatial pattern of the carrier-wave amplitude", and that "as long as we do not alter the animal's training, the same map emerges each time an animal sniffs a particular odorant, even though the carrier wave differs with each sniff" (Freeman, 1989, 80).[8]

Pribram thinks of all neural processing as pattern matching, because it is the outcome of a superposition of two patterns:

> Nerve impulses arriving at junctions generate dendritic microprocesses. The design of these microprocesses interacts with that which is already present by virtue of the spontaneous activity of the nervous system and its previous experience. (Pribram, 1991,9)

When dendritic microprocesses are generated there will be horizontal cooperativity in

> ensembles of mutually interacting pre- and post-synaptic events distributed across limited extents of the dendritic network. The limits of reciprocal interaction vary as a function of input (sensory and central) to the network — limits are not restricted to the dendritic tree of a single neuron. In fact, reciprocal interaction among pre- and post-synaptic events often occurs at a distance from one another, that is, in a saltatory fashion. (Pribram, 1991, 16)

These reciprocal interactions will be pattern matching because they convolve incoming activation patterns with resident microstructure. Microstructure generated by nerve impulse arrival interacts with what is present in virtue of pacing and previous configuration; the resulting interference patterns "act as analog cross-correlation devices to produce new figures from which the patterns of departure of nerve impulses are generated" (1971, 105). In other words, slow potential structure computes "both the spatial neighbourhood interactions among neural elements,

[8]I will add that, where perceptual processing is involved, Pribram thinks of neuronal states as *presentations* rather than representations. The nervous system is presenting a perceptual situation to the organism. Processing hierarchies may then *re-present* processing results to other computational stages, and this re-presentation will involve some form of systematic transformation. Pribram's notion of perceptual presentation is compatible with Gibson's theory of direct perception, and Pribram's description of neuronal processing as frequency filtering can serve to fill out Gibson's notion that brain structures "resonate with" the higher-order properties of perceptual stimuli. (It is also interesting to notice a historical relation between connectionist and direct theories: Dreyfus tells us (1988, 330) that Gibson and Rosenblatt coauthored a paper in 1955.)

and, to some extent, the temporal interactions over a range of sites" (1971, 18). What is passed on with axonal firing patterns will thus be the effects of reinforcement and occlusion at intersections of resident and newly evoked wavefronts. Mathematically, this sort of transformation is — like holographic processes — a filtering operation, implemented by means of parallel connections among cooperating analog circuit elements.

Along with a wish to develop a connectionist version of holographic neural processing Pribram expresses a

> desire to portray a neural systems analysis of brain organization in figural perception by developing sets of quasi-quantitative models, that is, to describe processing in terms of formalisms found useful in ordering data in 20th century physical and engineering science. It is my conviction that it is only through the use of these formalisms that the psychological and neurological levels of inquiry regarding perception can become related. The relationship entails sets of transformations which, unless they are described precisely and formally, are apt to be misunderstood by both psychologists and neuroscientists. (Pribram, 1991, xvi)

As a consequence, holonomic brain theory offers a set of mathematical models of various aspects of neural function. Pribram says his theory is a form of *probabilistic functionalism,* and by this he means what, in relation to connection machines, I have called thermodynamic functionalism. Pribram makes it clear that thermodynamic minimization is a metaphor for another sort of global self-stabilization which he calls entropy, rather than energy, minimization. Hamiltonians, principles of least action, define paths in a Hilbert space. Applied to statistical mechanics, Hamiltonians become operations which achieve minimization of uncertainty. The system settles into a state in which energy is maximally *ordered* — redundancies and correlations extracted, structure distinguished from noise. The resulting state will embody a maximum number of informational constraints; it will be maximally coherent. I will say more about constraint satisfaction and representational coherence in the next section. Here I only want to add that the phase space Pribram has adopted to describe the overall configuration of neural microprocesses is a 4-dimensional space providing for two coordinates in the spectral domain (this involves Fourier transforms of wave properties) and two spacetime coordinates required by the inherent spatiality of brain processing.

In concluding this description of holonomic brain theory I want to point out that computational neuroscience, as outlined by Pribram, does not envisage an incommensurability between physical and cognitive languages of description.

> A computational neural theory of perception must specify the relationship between operations of the subneuronal to the neuronal level; those at the neuronal level to

> those at the neural systems level; and, as well, those at the neural systems level to those at the perceptual level. (Pribram, 1991, 18)

Because the computations envisaged are mathematical functions over physical states, and not, in the first instance, logical computations over task domain concepts, it is thought that physical reduction of cognitive terms may be effected bottom-up. This belief is based on a naturalist assumption that all cognitive states have intrinsic content as a function of the physical microprocesses that construct and re-construct them.

IV.4 Epistemological suggestions

> It is very likely that connectionist models will turn out to offer contributions to the modeling of human cognitive performance on higher-level tasks that are at least as significant as those offered by traditional, symbolic, models.
>
> It is likely that the view of the competence/performance distinction that arises from the connectionist approach will successfully heal a deep and ancient rift in the science and philosophy of mind. (Smolensky, 1988, 3)

Logical functionalism thinks of high-level computations over internal representations as being like the sorts of formal operations we perform on external representations — sequential reasoning by means of deductive chains, categorization by means of definitions which are sets of necessary and sufficient conditions. Any cognitive performance — any actual cognitive achievement — is thought to be effected by competence-structures which are procedures or rules having the same logical/linguistic form as instructions or rules articulated by people communicating to each other or to computers. These ways of conceiving of cognitive behavior perpetuate a form of mind/body division: languages of description based on the objects and procedures of our task domains seem to be largely incommensurable with languages developed to describe the physical behaviors of biological bodies. Classical cognitive science, inasmuch as it wants to consider cognitive behavior as symbol-using behavior, is motivated to try to drive a wedge between cognitive and non-cognitive functions in the brain. It isn't possible to keep this sort of wedge in place unless we want to think of human bodies as the residences or vehicles of angels. If we are biological beings, then cognition just is a physical function. In this section I will outline briefly some of the epistemological suggestions a thermodynamic functionalism can offer. In chapter V I will go on to discuss the relations of connectionism to codes and to languages of description.

We have seen how, in a connection machine, a representational state is a stabilized pattern of node values, and how, in holonomic brain theory, a representational state is a temporary stability in the holoscape of dendritic microprocesses. We have also seen that both sorts of pattern can be given a geometrical representation as positions in some sort of phase space whose (orthogonal or nonorthogonal) coordinates are the dimensions along which node values may vary. Paul Churchland takes the additional step of simply identifying creature representation with its geometrical description. This identification allows him to make connectionist suggestions about the nature of categorization, generalization, and abstraction — about the inheritance by classes of properties of their superclasses, and the generalization to superclasses of the properties of classes.

Where a connectionist net is being trained to respond differentially to sample inputs, classification is automatic. A certain range of combinations of features will result in one output decision, and another range of combinations will result in another output decision. It can be said that training has partitioned the net's weight space. Within any partition, every combination of features will be represented as a single point and, moreover, the geometrical relation between point locations in state space will embody a similarity metric (P.M. Churchland, 1989, 102) A similarity space of this kind may eventually be identified with some phenomenal domain: olofactory space, color space, tonal space, motor control space. Using this geometrical formalism allows us to think of categorization as multi-dimensional and scalar.

Categorization is a form of generalization: somewhat different stimuli are given a similar response, whether this response is a name or an action or an internal processing decision. If we have layered nets we can think of partitions as being subsumed within larger partitions: where the net is trained to respond differentially to the more inclusive hypervolume, the subvolume will inherit the properties — i.e. the downstream decisions — of the superclass. Connectionist geometry also suggests how successive identifications might work. A blindfolded gardener is offered something to smell. First sniff: "It's a rose". Second sniff: "It's an old rugosa of some kind". Third sniff: "I think it's Blanc Double de Coubert". Here the net would be fine-tuning, involving a finer sub-partition with each trial. And a prototype of a class can be seen as a central subvolume in a partitioned space, the volume that is most similar — on the largest number of axes — to the largest number of training instances. Donald Hebb speaks about categorization in something like these terms: he is guessing that the activity of a trained cell-assembly is "representative of that combination of events that individually organized the first-order assemblies" (1980, 108) and that abstraction has to do with the practice-based organization of downstream cell-assemblies responding selectively to commonalities of first-order assemblies.

So the construction of weight distributions in a net is already a cognitive activity: it contributes to present experience an *order* achieved over previous experience. A trained net rapidly configures itself into the simplest, most coherent pattern consistent with input features. This response can be seen as a form of abduction — rapid inference to a best explanation in the light of species interests and individual experience. This sort of inference can be seen as semantics-driven, where calculus-plus-proof-procedural schemes of the symbolic paradigm are syntax-driven. We do not need to posit symbols because we have representational structures with intrinsic semantics — structures that satisfy logical constraints in virtue of what and where they are. The construction of a representation of an individual or a class will automatically construct a simultaneous configuration of relations to other individuals or classes. If we think of node values as embodying representational microfeatures, net-training will result in feature-representation that gives us ready-made the interdependencies among input features. These interdependencies are the systems of constraints a representation will have to satisfy. Units that are *on* together in a trained response to input will define consistency for that input. Units that are on together over larger regions, or that activate one another in trained cascades, will define consistency for larger cognitive territories. Trained sequences can be seen as giving us a form of induction: if *X*, then *Y*, for most or all of the instances encountered. Inference by activation cascades through a structured net gives us a sense of reasoning as a skill like other sorts of practical skill — a sensitive, multidimensional equilibration in the midst of complex inner and outer conditions. A matter not of flawless formal sequencing, but of considering many factors at once, hanging out in the centre of a possibility space until a solution achieves itself. George Lakoff puts it this way:

> In Connectionism, reasoning is not deducing one thing from other things, but rather putting things together well. That changes the whole idea of what it is to think. And that means philosophy will change, economics will change, sociology will change, and anthropology will change. (Lakoff in Allman, 1989, 171)

Classical cognitivists will reply that if philosophy does change it will not be for the better: the intuitive arts may do for basketball or needlework, noncognitive stuff having to do with bodies, but the banner capacities of human rationality — the ones that make us better than women and children and animals — must be explained in terms of deductions from axioms, recursive operations on symbols.

Well, it is often observed that axiomatization comes late in a game. We know how to know things long before we know how to say how we know how to know things. But it is true that connectionism is better at explaining simultaneous capacities like perception than it is at explaining recursive serial capacities like sentence generation. And connectionism does owe us an explanation of language. We may not be very good at handling the sort of recursion that occurs in the second sentence of this paragraph[9], but we do often 'think in sentences' and hear ourselves doing so.

If we think of creature brains as representing and computing in virtue of the organization of their materials, and if we think of this organization as having semantic content intrinsically, in virtue of its causal relations with a larger world, then we have no reason to think that only linguistic states have semantic content. We can imagine language functions as requiring an output transducer which maps certain intrinsically representational states onto sets of output decisions which result in the utterances of words and sentences. This is not a transduction from a physical domain to a nonphysical domain, of course — it is a transduction from brain behavior to muscle behavior. Patricia Churchland suggests that we can think of output transduction into language as the convenient provision of a precis of idiosyncratic neural complexity. She quotes C.A. Hooker to the effect that "Language will surely be seen as a surface abstraction of much richer, more generalized processes in the cortex, a convenient condensation fed to the tongue and hand for social purposes" (Hooker, cited in P.S. Churchland, 1986, 396). The implication is that our sentences need not be seen as expressing the whole of our thought — there is 'more behind them'. I will raise this possibility again when I reopen the discussion of codes in chapter V.

If language is a convenient abstraction from "richer, more generalized processes in the cortex", what neural regularities can account for the regularities observed in sentences? Piaget speaks of repertoires of sensory-motor schemes, schemas that bind object-schemes to action-schemes. The noun-verb-noun form ubiquitous in human sentences does resemble ubiquitous forms of practical action: *X* does *Y* to *Z*. It is also not unusual for the order of elements in a sentence to reflect the order of some event, or the order in which it has come to be known. Piaget further posits schemes of assimilation by which typical cognitive sequences will be applied to new items or domains. Hebb's physiological suggestion here is that organized activity in fields and cascades of cell-assemblies will recruit other cell-assemblies active at the same time, and that this sort of recruitment results in a progressive binding-together of consistent representational content (1980, --). So linguistic output can reflect the structures of characteristic non-linguistic sequences.

[9]Classical cognitivists argue for a linguistic description of cognition as symbol-using on the basis that the systematicity and productivity evinced in cognitive processes can only be explained if brains are code-using in the ways digital computers may be seen as code-using. A connectionist can reply that the systematicity and productivity displayed by human cognition is seen to be very limited when we compare it to the recursive abilities of digital computers, and that this is why we seek their help for tasks requiring unlimited numbers of recursive steps. Secondly, elementary systematicity and productivity are normal features of social and sensory-motor behavior, and are not special to linguistic skills. And thirdly, the more advanced sorts of recursion we do display seem to have to be supported by internalization of external codes and other cognitive prostheses such as diagrams.

Still, given that our own and other people's language also arrives as input, linguistic structure will also organize cognitive sequences. There will be a two-way interaction. Vygotsky (1962) offers a suggestive account of internalized speech which gradually, as we become more adept, becomes less social and more idiosyncratic, so that in the end the subjects of sentences are often dropped in favor of some sort of non-linguistic wave in the direction of an 'image' or other perceptual reactivation. When we 'catch ourselves thinking' this is the sort of thing we are not surprised to observe. In any case, the idea is that the convenient precis offered by language can be made available to one's own processes as well as those of other people, by being internalized as simulated inputs. These precis may also include internalizations of our uses of other cognitive prostheses such as diagrams, maps, written arithmetical calculations, or spoken instructions.

I will assume here a theory of imagining that takes it to be "a percept occurring in the absence of the thing that seems to be perceived" (Hebb, 1980, 107). There will be important cognitive advantages to be gained by being able to evoke an activation pattern in the absence of its usual cause. Pribram (1971, 370) speaks of a "repeated return to the configural form of a representation"as serving a rehearsal function and allowing the occurrence of additional distributions in memory — that is, of restimulating a representational configuration in order to make use of new connections with other representations. Hebb suggests that centrally activated simulations will allow us to hold or rework inputs, to set up new interactions among inputs, and to activate outputs in the absence of external stimuli — to give us cognitive flexibility, in short. Some restimulations may involve cell-assemblies quite close to the sensory periphery: "Ordinarily the cognitive operations operate back onto those that are sensory driven" (Pribram, 1991, xviii). Hebb suggests that some perceptual restimulation may be less inclusive "of lower-order representations" and then "the image of the whole, as a higher -order cell assembly activity, lacks detail but is still an image of the whole" (1980, 127). In other words, there may be different degrees of inclusion of the perceptual base in cognitive activity.

Piaget (in Piattelli-Palmarini, 1980, 165-67) relates simulation abilities to naming abilities by means of the development of what he calls the semiotic function. He posits a cognitive sequence which begins with imitation: a child imitates the sound of a name pronounced in the presence of the object named. The next step is deferred imitation: the child speaks the name when no one else is pronouncing it, but still in the presence of the object. Then comes symbolizing play: the child speaks the name in the absence of the object. And then full internalization of both the speech act and the perception of the object: the child mentally rehearses the name of the object together with a perceptual simulation of the object. Piaget stops there but we could go on to a short-circuited 'abstract' version where the physical instantiation of the mental rehearsal of the name cascades directly to the usual cognitive consequences of the activation of the object-assemblies. ✓

So linguistic input will sometimes evoke and sequence — organize — non-linguistic representation and computation. And distinctive linguistic regularities — as well as distinctive cultural and social and practical regularities — can become the cascading habits of neural activation. wisdom It may be that regularities of these sorts, if centrally instantiated, can operate top-down by means of feedback which in effect assigns subroutines, or pre-tunes perceptual capacities, by providing a 'set' that tells the system how input should be taken. These centrally-imposed constraints need not be seen as linguistically imposed, but they may be.

This discussion of connectionism and language capabilities has sketched some of the ways connectionists can begin to try to account for abilities taken as central by classical cognitivists. They are hints, not theories, and all that needs to be drawn from them is the suggestion that we are not forced to explain sentential capabilities by means of internal representations that are themselves sentences.

A classical cognitivist may, at this point, say that it is *those* activation cascades that are organized into language-like sequences and centrally evoked that we are calling cognition proper; the rest is functional architecture. When we supply a story about connectionist language, do we open the door to cooption by logical functionalists who want to assign us to everlasting labour as implementers? Not if we can show that brain connectionism is cognitive from the bottom up — that even its microprocesses are semantic — and that language itself is a function that presupposes a deep reservoir of intrinsic creature content.

IV.5 Intrinsic content again

> The existence of obvious, causal connections between semantic principles and physical properties suggests the possibility of 'mixed vocabulary' principles involving both semantic and physical terms. If true in the general case, this might undermine the thesis of autonomy of the semantic level, hence, the basis of the cognitive-penetrability criterion ... The distinction between functional architecture and symbolic processes would disappear. (Pylyshyn, 1986, 137)

We have come a long way from 'analog' and 'digital'. I said in the introduction to chapter I that what makes nonsymbolic computation and representation possible in analog computers is not what makes it possible in creatures that construct their own representations in concert with a structured environment. Analog computers may operate as functional analogs to other physical systems if the causal systematicities present in their materials and organization can be used to realize a formalism which also models that other system. If the formalism concerned provides a detailed model of the causal interrelations of the working parts of the two physical systems, then the analog computer will be what I have called a working analog. In either case its nonsymbolic transformations will preserve representational relevance because causal systematicities are mapped onto inferential systematicities. So we can read the states of one physical system into the states of the other. But these representing states have only extrinsic content, content that depends on *our* provision of the semantic function that maps physical states onto our mediating formalism.

What is it that makes nonsymbolic computation possible in connection machines? Physically, connection machines can be seen as very complicated quasi-analog machines, whose computational systematicity stems directly from the physical organization of the machine, even when their component switches are bistable. So what makes representation and computation possible in connection machines is basically what makes it possible in ordinary analog machines. Connection machines are a step closer to intrinsic content in just one way — their physical causal systematicity is in some small degree provided by their self-organization in response to input samples. To this small extent they behave like something that is calibrated, not just to our purposes, but to their environment. This is also the key to nonsymbolic computation in brains, which are environment calibrated from top to bottom — or, I should say, from bottom to top.

Environment-calibration is not analogy: it is not a relation between physical systems both of which are modeled in some formal structure. Environment-calibration could be defined as computational efficacy constructed, phylogenetically and ontogenetically, by means of the structured nature of interactions with a world. Creature calibration is the cumulative self-organization which supplies and is built by interactive competence. It results in a structured creature, an environmentally configured animal. A creature structured in such a way may be seen as a global input-output transducer, because the whole nervous system can be seen as one very complicated circuit for delivering viable responses. Pylyshyn accuses both Gibson and the connectionists of concentrating on mere transduction. But we do not need to see the animal as either passive or non-cognitive in relation to the computation of its overall response. Intermediate stages in response computation may provide great stimulus-independence, great sensitivity to small variations, and they may well involve language areas of the brain. They can thus be seen as fully cognitive by those who care to highlight that distinction.

A

Direct theorists (see Michaels and Carello, 1981) point out that the concept of representation is redundant in an account that gives us representations as the changed structure of a creature. Physically seen, a representation is *that* only inasmuch as it changes the flow of selections of alternatives in a brain. We do not really have to talk about the representation *of* features by nodes, but only about the response *to* features of nodes. This allows us to leave our psychological predicates to what I would think is their proper application as descriptions of whole persons — who do know, remember, compute, infer, classify, solve problems, test hypotheses, and represent. But we do not have a separate vocabulary for central processes and so we continue to apply a person-metaphor to brain processing. The largely unexamined presence of this metaphor may set up the classical cognitivist's greased slide from outer symbols to inner symbols.

Calibrational content is Patricia Churchland's term (1980, 8) for what I have been calling intrinsic content. She contrasts calibrational content with translational content, which is the content *we* assign to the signals of another organism on the strength of some systematic fit between their terms and ours. Translational content is made possible by the prior existence of calibrational content. Words and symbols can trigger structured responses because those structures are already there and are inherently semantic — whether by this we mean that they are certain sorts of conscious experience, or only that they have systematic causal relations with other structures. Pylyshyn claims (1986, 215) that "structural or topological constraints" do not "show through" in "representation-governed regularities' important to such activities as "solving cross-word puzzles, doing science, appreciating a novel". But this cannot be so, since structural and topological constraints are what give us the ability to respond to words and sentences with activation reconstructions that bind those words to perceptual experience and thus to environmental events — events which supply the pages of a novel with content, in other words.

There is a last question I should ask here. If ordinary analog and digital computers were able to modify their hardware as a result of regularities in input, would we want to ascribe calibrational content to their representational states? (I will keep the term, with due caution.) It is hard to draw a bead on the answers to this question, because the computational efficacy of both types of computers has depended on the fact that we can count on them *not* to change their hardware in response to regularities in the data. If they spontaneously modified their connections or binary states we would think they were malfunctioning. If an analog computer modified its resistances or capacitances as a result of a series of inputs (inputs we feed it), then it would no longer be an analog of whatever domain we want to investigate. It would be describable by some other formalism — by some equation we would have to try to extrapolate from our data. What if the analog computer gets its input settings via sensors that link it directly to the domain we want to investigate? What if its input comes from measuring devices at various points of our bridge and flood system? The answer remains the same: unless the physical laws by which we have described the dynamics of our physical system are changing, input-induced hardware alteration will constitute failure of the machine. If we want states of the machine to remain representationally relevant, our hardware must not flex.

What about digital machines? Chips are being developed which have some limited measure of ability to modify themselves. (This involves many-layered wiring with silicon interlayers that allow burn-through in selected locations; when current is present in both upper and lower wires, a short-circuit results, and this short serves to wire in an item of read-only memory store.) Will we want to grant their states intrinsic content? This sort of structural change is permanent, and to the extent that it occurs, it will create a dedicated rather than a general purpose machine. But this is not the point. The point here is that this digital machine is still running on top-down semantics, with certain slots left open for 'environmental' input. The machine is not importantly reconfigured by its input.

What if there are more general rewirings? We have several possibilities. They may result in what the compiler takes to be syntactic errors — errors in the combining and sequencing of what it is taking as syntactic elements. In this case the compiler will report the error and stop the machine, or it will enter some default state that works around the error and continues. The other possibility is that the 'error' is not a syntactic error but a semantic error — an error that does not violate the rules of a programming language, but that does enter as data some value that throws our deductions out of whack. Again, we will say the machine is not working properly.

What if our digital machine reorders itself comprehensively, in a globally systematic way? In this case we would say it has been recompiled. It is now implementing some other language than we had originally compiled it to run. And we will have no way to interpret this language. A digital computer with intrinsic content would be a maverick, a runaway, absolutely unuseable.

There are two suggestions to be extracted from these speculations about environment-calibrated content in analog and digital machines. One is that creatures must be environment-calibrated in a thorough-going way if they are environment-calibrated at all. Their whole machine, with its combination of rigidities and flexibilities, must work as an ensemble. The other suggestion is that, where machine computation is code-mediated, content *must* be assigned in a top-down fashion and then compiled right down to the floor — otherwise we are left without the key that allows us to decode our results.

V. Languages of Description

The issue is within what class of systems should a description of intelligent systems be sought. (Newell, 1983, 198)

A digital computer can be described as a physical system and as a logical system. Two modes of discourse are involved: the modes of discourse we have developed to speak about the behaviors of dynamical systems and the modes of discourse we have developed to speak about operations in formal systems. The computer metaphor of intelligent function brings with it the possibility of both kinds of description. Classical cognitivism chooses the languages of programs with their rules and symbols. Connectionism chooses dynamical description. There are two kinds of questions we can ask about these choices. One is this: does it matter what class of system *we* use to describe intelligent behavior? The other is this: does it matter what class of description we say the intelligent system *itself* is using? These two questions are obviously quite different but they are not always discriminated in polemical practice.

Our answer to the first question can hinge on several kinds of motivation. We may say it matters for methodological reasons — one class of system has greater explanatory adequacy to the data, stimulates more interesting research. Or we may say it matters for reasons of disciplinary alliance — it is always pleasant if research funds and ideas do not migrate away from the formalisms in which we are already competent.

What are the alliances and competencies associated with both sorts of descriptive frame? Logical functionalism has historical links with the movements to axiomatize mathematics and logic: its style and explanatory kinds are those of set theory, formal language theory, predicate calculus. Programming languages like LISP, which are procedural languages, are still thought of as strings of conditionals naming conditions and operations. Connectionists and connectionist neuroscientists are looking for explanatory possibilities in many language systems — electrical engineering, general systems theory, nonlinear systems theory, thermodynamics, high-dimensional geometry, topology, Fourier analysis, holographic theory. There is an evident desire to recast the terms of computational psychology so they will be more plausibly relevant to biological and phenomenological conversations, and to do it without losing the possibilities of modeling precision offered by mathematical languages.

We have seen some of the ways mathematical description has been useful to connectionists and neuroscientists. When Pribram says "It is my conviction that it is only through the use of these formalisms that the psychological and neurological levels of inquiry regarding perception can become related" (1991, xvi), what he has in mind may be something like Marr's sense that figural perception — a psychological-level task — is constrained by optical facts — facts for which we have a developed mathematical theory. If we manage a mathematical theory of neural function, we may be able to link our theoretical levels.

Marr, Pribram, and connectionists like Paul Churchland tend to speak as if the nervous system is 'doing math' — computing a difference-of-Gaussians, arriving at Fourier coefficients, performing matrix multiplication. Are they implying that brains are *using* equations, and is this akin to Pylyshyn's saying brains use sentences? Not necessarily.

> There is no more contradiction between a functional description of an ensemble of cortical cells as performing a spatial frequency analysis and their having receptive fields with certain physiological properties than there is between a functional description of some electronic component as being a multiplier and its being made up of transistors and resistors wired in a certain fashion. The one level functionally describes the process, the other states the mechanism. (DeValois and DeValois, 1988, 288; cited in Pribram, 1991, 269)

"Functionally describes the process" seems to me to be the right way to say it: we have a physical process; we are describing it; and our description is given in the mathematical form of a function. If, instead, we were talking about 'implementing' or 'instantiating' a function, we would inherit some of the ambiguity attendant on terms that do not discriminate between a relation between two linguistic terms and a relation between a thing and a term. 'Implementation', for instance, covers many sorts of situation. We can implement an idea, a design, an algorithm, an equation, a language, a program, or a computer type; and we can implement it *into* a more detailed design, a class of physical devices, a methodology, a programming language, an algorithm, an equation. We 'instantiate' types into tokens, but we do not distinguish between the physical token and its linguistic function. So, if we were to talk about cortical cells 'implementing' or 'instantiating' equation *E*, we could mean either that *E* is a good mathematical description of their physical behavior, or that they physically embody the syntactic elements of equation *E*, the way bistable switches in a digital computer can be seen as physically embodying O's and l's.

There is a further difficulty with 'implementation'. The term is at home in hardware and software design contexts, and there it is used when we have a design allowing more than one alternative in how a function is to be performed. We implement a multiplier either by successive additions and shifts, or by table look-up, for instance. One of the inevitable connotations of the terms is top-down designation: we implement one programming language into a lower level one, or we implement a design specification into a hardware design. We do not have an equivalent term for the relation of lower- to higher-level descriptions where computational systems organize themselves from the bottom up. 'Implementation' won't do, because it imports a suggestion of top-down organization which may additionally be seen as the syntactic organization of the formalism expressing that organization.

If we hold on to our sense of the way a mathematical description "functionally describes a process", we can look again at theories that describe intelligent behavior in terms of logical systems. Pylyshyn and theorists like him have drawn on Chomskian notions for their sense of intelligence as rule-using. Boden makes the point that Chomsky himself is not making their sort of claim.

> Chomsky's formal rules were not programs, nor were they intended as specifications of actual psychological processes. Rather they were abstract descriptions of structural relationships. They were 'generative' in the timeless mathematical sense, whereby a formula is said to generate a series of numbers, not in the sense of being descriptions of a temporal process of generation. Similarly, his 'transformations' were abstract mappings from one structure to another as a square-root function transforms 9 into 3, not actual psychological changes or mental events. Likewise, his 'finite-state and non-finite *machines*' were mathematical definitions (as are Turing machines), not descriptions of any actual manufactured systems that might conform to those definitions. (Boden, 1988, 4)

"Abstract descriptions of structural relationships" in observed behavior are like descriptions of a neural cell ensemble as computing a difference-of-Gaussians. Both are functional descriptions of a properly functionalist kind:

> The programmer attempts a general proof that results of this class can be computed by computational systems of this form, given certain specific constraints (which may apply to naturally evolved psychological systems). Indeed, there may not even be a program, but only an abstract analysis of the information-processing task in question. Such theorists agree with Chomsky in stressing the *what* of computation, rather than the *how*. Accordingly, they may use the term 'computational' to refer not (as is more usual) to computational *processes*, but to the *abstract analysis of the task* facing the psychological system — irrespective of how it is actually performed. (Boden, 1988, 7)

If we take logical functionalists as speaking of a what rather than a how of intelligence, then we lose the need to object to rules and symbols. A *rule* is just what we call a processing regularity if we are speaking the language of logic. A *symbol*, if we are speaking the language of logic, is just a causally-significant processing nexus. Thought of this way, we have no trouble giving connectionist computation a logical description. Symbols will be distributed excitation patterns. Rules will be information processing interdependencies deriving from connective topology and weighting. Algorithms will be descriptions of the progressive self-organization of a network of computational units. Computation will be non-sequential cooperative equilibrium-seeking alterations of patterns of activity.

Two characteristics of connectionist computation help us make the transition to logical description that does not imply the use of digital-style symbols. One is the lack of storage in connectionist machines. Digital machines store copies of digital symbols in storage locations; but connectionist representation is re-evoked, not copied, stored, shunted, or retrieved. The second is a view of data-intake that is not limited to one superficial array. A cascade through layers of a net can be seen as continuing to extract higher-level input features all the way along its progress. This gives us Gibson's sense of the richness of environmental information — a richness it has in conjunction with the complexity of the structure that responds to it. If we are not positing impoverished input, then we do not need superficial transducers supplying elementary symbols from which a description will be deduced.

Connectionists have been more nervous than they need to be about the possibility that connectionist processing might be describable in logical language. They are wary of the likelihood that, if connectionist computation can be described in a logical language, classical cognitivists who equate rule-describable systems with rule-using systems will describe connectionist processes as implementing symbols and procedures of the sort implemented in digital machines. As a result they have put quite a lot of effort into arguing that connectionist systems have no elements that can be supplied with a semantic interpretation , or that they have no rules, only "simultaneous soft constraints". Dreyfus provides an argument of the former kind:

> Most nodes, however, cannot be interpreted semantically at all. A feature used in a symbolic representation is either present or not. In a net, however, although certain nodes are more active when a certain feature is present in the domain, the amount of activity not only varies with the presence or absence of this feature but is affected by the presence or absence of other features as well. (Dreyfus, 1988, 328)

And Tienson and Horgan of the latter kind:

> Models capable of solving these problems will not even be describable by cognitive level rules. For if the system underlying, say, basketball skills could be described by such rules, then a classical rules and representations model of the same system could be produced in which these rules (or purely formal isomorphic ones) are explicitly represented as a stored program. (Tienson and Horgan, 1987, 104)

What is curious about these arguments is that they seem implicitly to accede to the strange assumption that if a system is rule-describable then it must be rule-using. If they try to defeat the classical cognitivists by demonstrating that connectionist computation is not rule-describable — or not simulable in some fashion on a digital computer, which amounts to the same thing — then they are also commiting themselves to saying it can have no systematic description at all. And this would have to include mathematical description.

They may want to say, as Robert Rosen (1991) does, that our present forms of dynamical description are not adequate to the functional complexities of systems that not only organize themselves but build themselves. Rosen argues that *both* logical and Newtonian dynamical descriptions are fit only to model closed systems, systems in which static elements are pushed around by unchanging forces. Biological systems, though, are high-level open systems, less dependent on substantive or energy linkages with the world and more dependent on their own energy organization, which provides large stores of potential energy ready to be triggered by input that in energy terms is insignificant.[10]

I have hinted throughout this essay that 'analog' and 'digital' are not logically symmetrical, that digital processes are a subclass of analog processes. There is a sense in which this is not true. Digital computers make up one of three classes (the other two are analog computers and connection machines) of contemporary computational technologies: digital processes may be a subclass of analog processes, but digital technologies are not a subclass of analog technologies — as technologies they have stood as equal alternatives.

But there are other senses in which a hierarchical relation is plain. The elements of discrete mathematics are subsets of the elements of continuous mathematics. Linguistic behaviors of organisms are a subclass of intelligent behaviors. Rule-governed behaviors of computational machines are a subset of law-governed behaviors of computational machines. Logical systems are a subclass of dynamical systems. Cultural behaviors are a subclass of natural behaviors. Wilden makes the point in the following way:

> In considering further the communicational and socioeconomic categories of relationship often obscured by the term 'opposition', one can discern a developmental and dialectical sequence of possibilities, beginning in (analog) DIFFERENCE, moving to digital DISTINCTION (which may or may not involve levels), and thence to the BILATERAL OPPOSITION in which each item is either actually of the same logical type as the other, or treated as if it were. (Wilden, 1980, 509)

[10]There is a distinction in logical type, Rosen says, between mechanistic, energetic-logical, closed system explanation, and open-system explanation in terms of self-organization and self-construction. Biological organisms are more complex than mechanical systems and, although there can always be simple models of complex systems, the category of all functional models of organisms is larger than, and includes as a subcategory, the category of all models of mechanisms, including machines. Human cognition, as a biological function, will also be more complex than any sort of machine function. Rosen's position, which emerges from a radical reenvisioning of the epistemology of science, and which would see Newtonian physics subsumed within a more comprehensive physics of organisms, supplies an illuminating broadly-based objection to the computational hypothesis as it has been imagined up to now.

What are the analog differences that make possible digital distinctions? How do they become digital distinctions? I have described codes — which are subclasses of languages — as being made possible by an output transduction of linguistic decisions over a depth of non-linguistic organization. In the connectionist picture, it is the partitioning of cognitive phase space — the significant organization of the net — that provides the semantic reservoir upon which digital/symbolic code elements are floated: *

> A specific image or belief is just an arbitrary projection or slice of a deeper set of data structures, and the collective coherence of such sample slices is a simple consequence of the manner in which the global information is stored at the deeper level. (P.M. Churchland, 1989, 109)

If 'digital' is, in general, a subclass of 'analog', how do we come to symmetrize what in fact is a hierarchical relation? Political and social motivations have something to do with it, but how does it escape notice? Wilden says dyadic contrasts which result in paradox or puzzlement do so because a third term remains unseen. **

> Dyadic oppositions of the same logical type are products of mediated — triadic — relations in which the third term commonly remains unrecognized. This is all the more true when the purported 'opposition' is between terms of distinct logical types. (Wilden, 1980, 510)

If this is true for the analog/digital distinction, what third term have we been neglecting? My guess is that we have been misdescribing transducers. It is as if the value of connectionist models is that they demonstrate a way to think about transduction as a passage from physical pattern to physical pattern. There can be cognitive behavior — we can be intelligent — not because transducers supply us with an exit from the physical, but because the physical itself, in virtue of its origin and structure, its complex function in complex creatures, organizes us into intelligence.

Conclusion

Computational psychologies which base themselves on some form of the computer metaphor of cognitive function have had access to two classes of explanatory system: a linguistic functionalism associated with digital computation and based on the modes of discourse we use to describe operations in formal systems, and various mathematics-based functionalisms originally associated with analog computation and based on the modes of discourse we have developed to describe the behaviors of dynamical systems.

I have argued that what I have called linguistic functionalism is compatible with two possible understandings of what we might mean by linguistic kinds as realized in brains. One is a construal which sees the physical states of the computer as realizing a function, just in the sense that their input-output transformations may be accurately described by means of that function. In a generous construal of this sort we may, if our explanatory entities are linguistic entities, identify causally significant computational substates with symbols, and processing regularities with rules. This is a construal that would see brain events as rule- and symbol-describable without carrying its functionalist metaphor into speculation about neural realization. A psychologist offering a computer simulation of some sequence of psychological causation would, on this construal, take the program as offering a linguistic description of a physical event and not a linguistic description of a linguistic event.

The other sort of understanding we might have of the relation of linguistic functionalism to brain function is what I could call a hard construal, which is based on a similar construal of code-processing in digital computers. This construal takes the brain to be realizing the formula as well as the function given in the program's model of psychological causation. In other words, the hard construal takes brains as rule-using as well as rule-describable; processing would depend on rules being present in the brain as program strings — as prescriptive inscriptions of some sort.

Analog computation is computation which clearly realizes functions, but which as clearly cannot be seen as doing so by realizing formulae, just because it lacks the syntactic requirements for notational systems. Because computational results are achieved without any involvement of the sorts of time-sampling, quantization or thresholding that allow digital computers to be seen as code-using systems, the existence of analog computers allows us to separate the notion of computation from the notion of code-using computation. Non-code-using computation is transparently computation achieved by means of the causal systematicities of the physical machine. Code is involved in analog processing, but in a way that is easily seen to be external to the operation of the machine itself — it is involved just in the sense that an analog computer set-up cannot be used either as a functional analogy or as a working analogy of some other physical system, except by means of a description which is common to both systems, which we provide, and which is usually expressed by means of some code.

A computational system cannot be seen as a code-using system unless it provides discrete signals that may be identified with the discrete, nonfractionable, context-independent syntactic elements necessary to notational systems. But the existence of discrete signal elements also does not imply that a computational system *must* be a code-using system. As Lewis observes, there may be discretized systems that operate in ways which are not significantly different from the operation of analog computers — in both, the computational work may be done straightforwardly and transparently by means of the given physical configuration of the machine, so that positing the operation of an internal code would be redundant. Pribram suggests that discrete signal elements may also have kinds of computational relevance other than suitability to the assignment of code values; they may for instance be used to linearize a dynamical system. Discrete signal elements, then, are a necessary but not a sufficient condition for code-using systems.

Connection machines may have either continuous or discrete processing nodes, but because the thresholds of two-state nodes have statistical rather than logical effect, connection machines, like analog computers, are usually given non-linguistic descriptions. Since the non-linguistic functionalisms do not mention code at all, we avoid both the generous and the hard construal of linguistic functionalism. But if we do wish to invoke linguistic functions — and we are certainly free to do so, since we can give functionalist components any sort of name we like — we can call connectionist activation patterns 'symbols' and processing regularities 'rules'. But doing so does not license us to assume that these functionalist 'symbols' are processed with the intermediacy of the sort of structure-sensitive 'rules' that in a digital computer are entered as data strings that instruct a central processor. Because there is no central processor in a connectionist system, we can see connectionist 'symbols' as *processes* but not as *processed*: they are active patterns with computational significance. Their activity embodies the 'rules', but they are ~~not~~ submitted to rules conceptually separable from their activity. Since this is so, we have no particular reason to want to describe the system as code-using. As is true of analog computers, the computational work of the system is accomplished transparently and directly by means of the organization of the materials of the physical machine.

We do not know whether or to what extent brain processing is controlled by something like a central processor or processors, nor do we know very much about how linguistic input can function to organize brain events in a top-down manner. We do know that we can respond to verbal instructions given by other people, and that we sometimes remember verbal instructions ("Now I'm supposed to add two beaten eggs") in the course of performing some task. Like digital computers, we *can* be programmed by linguistic means. So it does sometimes make sense to say that we are rule-using systems. It can be argued that this sort of rule following behavior indicates a general capacity for storing rules and for using them to guide centrally-initiated behaviors. But we are also able to 'store' non-linguistic sequences and use them to guide complicated non-linguistic behaviors. A trained dancer, for instance, can take in a long sequence of complex movement demonstrated once, and reproduce it either on the spot or sometime later. This argues for a general capacity not so much to follow rules as to register a perceived circumstance and to re-evoke it at will. Linguistic instructions are, after all, perceived circumstances like any other, and if we can re-evoke the sight of the Downs at sunrise it is not surprizing that we can also re-evoke a sentence in a recipe.

A digital computer is described as rule-using because the physical instantiations of strings of code reset switches in the central processor and in memory registers. Because digital machines are general-purpose machines, these programmed switch-settings are essential to the computational viability of the machine. They in effect provide the machine with the physical organization that gives it the causal systematicity necessary to computational relevance. The computational organization of the machine is thus code-mediated in an obvious and thorough-going way. I have argued that the computational states of the digital computer have extrinsic content supplied by our provision of this interface with a code whose interpretation is arbitrary with respect to the machine itself.

Brains are devices whose computational organization is created not by externally-supplied strings of instantiated code, but by their intrinsic phylogenetic and ontogenetic calibration to a structured environment. Where the primary computational organization of a device is intrinsic to the device and is not externally supplied, there is no explanatory role for an internal code. Explanations in terms of code would be generally redundant, except perhaps in the specific instances where code is involved as perceptual data.

I will emphasize this point: we *require* explanations in terms of code when we are dealing with general-purpose digital computers whose physical organization lacks the systematicity required for computational relevance until we input certain program strings of instantiated code. We do not require explanations in terms of code when we are speaking of analog computers or connection machines whose computational organization is either hard-wired or practice-related. We also will not require explanations in terms of code when we are speaking of brain processing, just to the extent that the brain's basic computational organization is either hard-wired or practice-based.

We can of course still posit code if we intend what I have called the generous construal, by which 'symbols' are equated with representationally relevant activation patterns, 'rules' with typical computational sequences, and 'algorithms' or 'programs' with high-level descriptions of computational events. But there are, I think, good reasons to be cautious in our use of even this softer construal. I will list them briefly.

(1) General talk of an inner code tempts us to conflate a description with the dynamical process it describes — a category error leading to puzzlement and forms of mind-body split.

(2) Describing intelligence in terms of a formal language imports a political bias in favor of the sorts of intelligence which are skills with formal language activities. It leads us to leave out of the account large areas of important and interesting cognitive ability.

(3) Describing cognition as rule-governed supports a hierarchical picture of personal and social order, wheras the connectionist paradigm supplies an image of autonomous, cummulative self-organization.

(4) A logic-based description of cognition tends to misdescribe human cognition as organized primarily in relation to the principle of non-contradiction. A dynamical picture of intelligence makes it easier to notice that cognition is variable, context-dependent, multidimensional, multifunctional, possibly contradictory and inherently biological.

Bibliography

Akers, L., Ferry, D. and Grondin, R., 1989: 'VLSI Implementation of Neural Systems'. In J. Brink, C. Burawa and C. Haden (eds), *The Computer and the Brain.* New York, North-Holland, 1989.

Allman, W., 1989: *Apprentices of Wonder: Inside the Neural Network Revolution.* New York, Bantam Books.

Arbib, M., 1989: *The Metaphorical Brain 2.* New York, John Wiley and Sons.

Arbib, M., 1990: 'A Piagetian Perspective on Mathematical Construction'. *Synthese* 84, pp. 43-58.

Bechtel, W., 1987: 'Connectionism and the Philosophy of Mind'. *Southern Review of Philosophy* 26, Supplement, pp. 17-40.

Benacerraf, P. and Putnam, H. (eds), 1983: *Philosophy of Mathematics*, 2nd edn. New York, Cambridge University Press.

Beth, E. and Piaget, J., 1966: *Mathematical Epistemology and Psychology.* Dordrecht, D. Reidel.

Bickhard, M. and Richie, D., 1983: *On the Nature of Representation.* New York, Praeger.

Bobrow, D., 1975: 'Dimensions of Representation'. In D. Bobrow (ed.), *Representation and Understanding*, New York, Academic Press, 1975.

Boden, M., 1988: *Computer Models of Mind.* New York, Cambridge University Press.

Boden, M. (ed.), 1991: *The Philosophy of Artificial Intelligence.* New York, Oxford University Press.

Boolos, G., 1989: *Computability and Logic.* New York, Cambridge University Press.

Block, N. (ed.), 1981: *Imagery.* Cambridge, MIT/Bradford.

Campbell, D., Crutchfield, J., Farmer, D. and Jen, E., 1985: "Experimental Mathematics: the Role of Computation in Nonlinear Science'. *Communications of the ACM* 28, pp. 374-384.

Castleman, K., 1979: *Digital Image Processing.* Englewood Cliffs, Prentice-Hall.

Churchland, P.M., 1989: *A Neurocomputational Perspective.* Cambridge, MIT/Bradford.

Churchland, P.S., 1980: 'Language, Thought and Information Processing'. *Nous* 14, pp. 147-168.

Churchland, P.S., 1986: *Neurophilosophy.* Cambridge, MIT/Bradford.

Churchland, P.S. and Churchland, P.M., 1983: 'Stalking the Wild Epistemic Engine'. *Nous* 17, pp. 5-18.

Couch, L., 1983: *Digital and Analog Communication Systems.* New York, Macmillan.

Craik, K., 1952: *The Nature of Explanation.* London, Cambridge University Press.

Cummins, R., 1989: *Meaning and Mental Representation.* Cambridge, MIT/Bradford.

Cussins, A., 1990: 'The Connectionist Construction of Concepts'. In M. Boden (ed.), *The Philosophy of Artificial Intelligence.* New York, Oxford University Press, 1990.

Delong, H., 1970: *A Profile of Mathematical Logic.* Reading, Addison-Wesley.

Demopoulos, W., 1987: 'On Some Fundamental Distinctions of Computationalism'. *Synthese* 70, pp. 79-96.

Dennett, D., 1983: 'Styles of Mental Representation'. *Proceedings of the Aristotelian Society* 83, pp. 213-226.

Dennett, D., 1984: 'Cognitive Wheels'. In *The Robot's Dilemma*, Z. Pylyshyn (ed), New York, Ablex, 1987.

DeValois, D. and DeValois, R., 1988, *Spatial Vision.* New York, Oxford University Press.

Dreyfus, H. and Dreyfus, S., 1988: 'Making a Mind Versus Modelling the Brain'. In M. Boden (ed.), *The Philosophy of Artifical Intelligence.* New York, Oxford University Press, 1988.

Field, H., 1980: *Science Without Numbers.* Princeton, Princeton University Press.

Fodor, J., 1981: *Representations.* Cambridge, MIT/Bradford.

Fodor, F. and Block, N., 1973: 'Cognitivism and the Analog/Digital Distinction'. Unpublished manuscript.

Fodor, J. and Pylyshyn, Z., 1988: 'Connectionism and Cognitive Architecture: A Critical Analysis'. *Cognition* 28, pp. 3-71.

Freeman, W., 1991: 'The Physiology of Perception'. *Scientific American* Feb. 1991, pp. 78-85.

Frege, G., 1891: 'Function and Concept'. In G. Frege, *Collected Papers on Mathematics, Logic and Philosophy*, B. McGuinness (ed.), New York, Basil Blackwell, 1984.

1904: 'What is a Function?' In G. Frege, *Collected Papers on Mathematics, Logic and Philosophy*, B. McGuinness, (ed.), New York, Basil Blackwell, 1984.

Gibson, J., 1979: *The Ecological Approach to Visual Perception.* Boston, Houghton Mifflin.

Gleick, J., 1987: *Chaos.* New York, Penguin.

Gluck, M. and Rumelhart, D. (eds) 1990: *Neuroscience and Connectionist Theory.* Hillsdale, Lawrence Erlbaum Associates.

Goodman, N., 1968: *Languages of Art.* Indianapolis, Bobbs-Merrill.

Hallett, M., 1984: *Cantorian Set Theory and Limitation of Size.* Oxford, Clarendon.

Halliday, M., 1990: *An Introduction to Functional Grammar.* London, Edward Arnold.

Halmos, G., 1960: *Naive Set Theory.* Princeton, Van Nostrand.

Hatfield, G., 1988: 'Representation and Content in Some (Actual) Theories of Perception'. *Studies in the History of the Philosophy of Science* 19, pp. 175-214.

Hatfield, G. and Kosslyn, S., 1984: 'Representation Without Symbol Systems'. *Social Research* 51, pp. 1019-1045.

Haugeland, J., 1981: 'Analog and Analog'. In J. Biro and R. Shahan (eds), *Mind, Brain and Function.* New York, The Harvester Press, 1981.

Haugeland, J., 1985: *Artificial Intelligence: the Very Idea.* Cambridge, MIT/Bradford.

Hebb, D., 1949: *The Organization of Behavior.* New York, John Wiley and Sons.

Hebb, D., 1980: *Essay on Mind.* Hillsdale, Lawrence Erlbaum Associates.

Hebb, D., 1982: *The Conceptual Nervous System: Selected Papers,* H. Buchtel (ed.). New York, Pergamon Press.

Horgan, T. and Tienson, J., 1987: 'Settling into a New Paradigm'. *Southern Journal of Philosophy* 26, Supplement, pp. 97-113.

Hurford, J., 1987: *Language and Number: The Emergence of a Cognitive System.* New York, Basil Blackwell.

Hyndman, D., 1970: *Analog and Hybrid Computing.* New York, Pergamon.

Illingworth, V. (ed.), 1990: *Dictionary of Computing*, 3rd edn. New York, Oxford University Press.

Irvine, A., 1989: 'Epistemic Logicism and Russell's Regressive Method'. *Philosophical Studies* 55, pp. 303-327.

Irvine, A. (ed.), 1990: *Physicalism in Mathematics.* Dordrecht, Kluwer.

Jackel, L., Howard, R., Graf, H., Hubbard, W., Denker, J. and Henderson, D., 1989: 'Electronic Neural Network Chips'. In J. Brink, C. Burawa and C. Haden (eds), *The Computer and the Brain.* New York, North-Holland, 1989.

Jackson, L., 1991: *Signals, Systems and Transforms.* Reading, Addison-Wesley.

Johnson-Laird, D., 1983: *Mental Models.* New York, Cambridge University Press.

Kampsis, G., 1991: *Self-Modifying Systems in Biology and Cognitive Science.* New York, Pergamon.

Keller, E., 1985: *Reflections on Gender and Science.* New Haven, Yale University Press.

Killeen, P., 1989: 'Behavior as a Trajectory through a Field of Attractors'. In J. Brink, C. Burawa and C. Haden (eds), *The Computer and the Brain.* New York, North-Holland, 1989.

Kitcher, Philip, 1983: *The Nature of Mathematical Knowledge.* Oxford, Oxford University Press.

Kitcher, Patricia, 1988: 'Marr's Computational Theory of Vision'. *Philosophy of Science* 55, pp. 1-24.

Korzybski, A., 1933: *Science and Sanity.* New York, Institute of General Semantics.

Kuc, R., 1988: *Introduction to Digital Signal Processing.* New York, McGraw-Hill.

Kunt, M., 1986: *Digital Signal Processing.* Norwood, Artech House.

Lafrance, P., 1990: *Fundamental Concepts in Communication.* Englewood Cliffs, Prentice Hall.

Langer, S., 1962: *Philosophical Sketches.* Baltimore, Johns Hopkins.

Lewis, D., 1971: 'Analog and Digital'. *Nous,* pp. 321-327.

Lindsey, R., 1988: 'Images and Inference'. *Cognition* 29, pp. 229-250.

McClelland, J., Rumelhart, D. and the PDP Research Group, 1986: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vols I and II.* Cambridge, MIT/Bradford.

Mcgillem, C. and Cooper, G., 1984: *Continuous and Discrete Signal and System Analysis.* New York, Holt, Rinehart and Winson.

Maddy, P., 1980: 'Perception and Mathematical Intuition'. *Philosophical Review* 89, pp. 163-196.

Maddy, P., 1984: 'Mathematical Epistemology: What is the Question?' *The Monist* 67, pp. 46-75.

Maddy, P., 1988: 'Believing the Axioms: I and II'. *Journal of Symbolic Logic* 53, pp. 481-511, 736-64.

Maddy, P. 1990: Realism in Mathematics. New York, Oxford University Press.

Marr, D., 1982: *Vision.* New York, Freeman.

Michaels, C. and Carello, C., 1981: *Direct Perception.* Englewood Cliffs, Prentice-Hall.

Millikan, R., 984: *Language, Thought and Other Biological Categories.* Cambridge, MIT/Bradford.

Nadel, L., 1989: 'Computation in the Brain: the Organization of the Hippocampal Formation in Space and Time'. In J. Brink, C. Burawa and C. Haden (eds), *The Computer and the Brain.* New York, North-Holland, 1989.

Newell, A., 1980: 'Physical Symbol Systems'. *Cognitive Science* 4, pp. 135-83.

Newell, A., 1983: 'Intellectual Issues in the History of Artifical Intelligence'. In F. Machlup and U. Mansfield (eds), *The Study of Information: Interdisciplinary Messages*. New York, John Wiley and Sons, 1983.

Nussbaum, M., 1986: *The Fragility of Goodness.* New York, Cambridge University Press.

Peterson, G., 1967: *Basic Analog Computation.* New York, Macmillan.

Penrose, R., 1989: *The Emperor's New Mind.* New York, Oxford University Press.

Piattelli-Palmarini, M. (ed), 1980: *Language and Learning: A Debate between Jean Piaget and Noam Chomsky*. Cambridge, Harvard University Press.

Poularikas, A. and Seely, S., 1988: *Elements of Signals and Systems.* Boston, PWS-Kent.

Powers, W., 1980: 'Pylyshyn and Perception'. *The Behavioral and Brain Sciences* 3, pp. 148-164.

Premack, D., 1983: *The Mind of an Ape.* New York, Norton.

Pribram, K., 1971: *Languages of the Brain.* Englewood Cliffs, Prentice Hall.

Pribram, K., 1991: *Brain and Perception.* Hillsdale, Lawrence Erlbaum Associates.

Proakis, J., 1983: *Digital Communications.* New York, McGraw Hill.

Putnam, H., 1988: *Representation and Reality.* Cambridge, MIT.

Pylyshyn, Z., 1984: *Computation and Cognition.* Cambridge, MIT.

Quine, W., 1960: *Word and Object.* Cambridge, MIT.

Quine, W., 1969: 'Epistemology Naturalized'. In H. Kornblith (ed) (1985), *Naturalizing Epistemology*. Cambridge, MIT, 1985.

Resnick, M., 1975: 'Mathematical Knowledge and Pattern Cognition'. *Canadian Journal of Philosophy* 5, pp. 25-39.

Rosch, E. and Lloyd, B. (eds), 1978: *Cognition and Categorization.* Hillsdale, Lawrence Erlbaum Associates.

Rosen, R. (ed.), 1985: *Theoretical Biology and Complexity.* Orlando, Academic Press.

Rosen, R., 1987: 'On the Scope of Syntactics in Mathematics and Science: the Machine Metaphor'. In J. Casti and A. Karlquist (eds), *Real Brains, Artificial Minds*. New York, North-Holland, 1989.

Rosen, R., 1991: *Life Itself.* New York, Columbia University Press.

Rucker, R., 1989: *Mind Tools.* Boston, Houghton Mifflin.

Rumelhart, D., 1989: 'Brain-Style Computations: Mental Processes Emerge from Interactions among Neuron-Like Elements'. In J. Brink, C. Burawa and C. Haden (eds), *The Computer and the Brain.* New York, North-Holland, 1989.

Sedivy, S., 1989: 'Conventional Naturalism: An Account of Pictorial Representation'. Unpublished manuscript.

Sejnowski, T., 1989: '"The Computer and the Brain" Revisited'. In J. Brink, C. Burawa and C. Haden (eds), *The Computer and the Brain.* New York, North-Holland, 1989.

Smolensky, P., 1988: 'On the Proper Treatment of Connectionism'. *Behavioral and Brain Sciences* ll, pp. 1-74.

Sterelney, K., 1991: *The Representational Theory of Mind: An Introduction.* Cambridge, Basil Blackwell.

Stokes, M., 1971: *One and Many in Presocratic Philosophy.* Washington, Centre for Hellenic Studies.

Strawson, P., 1966: *The Bounds of Sense.* London, Methuen.

Tienson, J., 1987: 'An Introduction to Connectionism'. *Southern Journal of Philosophy* 26, Supplement, pp. 1-15.

Tiles, M., 1984: *Bachelard, Science and Objectivity.* New York, Cambridge University Press.

Tiles, M., 1990: *Philosophy of Set Theory.* New York, Basil Blackwell.

Turkle, S., 1984: *The Second Self.* New York, Simon and Schuster.

Varela, F., 1984: 'Living Ways of Sense-Making: A Middle Path for Neuroscience'. In P. Livingston (ed), *Disorder and Order.* Saratoga, Anma Libri, 1984.

Von Neumann, J., 1958: *The Computer and the Brain.* New Haven, Yale University Press.

Vygotsky, L., 1962: *Thought and Language.* Cambridge, MIT.

Warnock, M., 1976: *Imagination.* London, Faber and Faber.

Whitford, M., 1991: *Luce Irigaray: Philosophy in the Feminine.* New York, Routledge.

Wilden, A., 1980: *System and Structure,* 2nd edn. London, Tavistock.

Wittgenstein, L., 1953: *Philosophical Investigations.* Oxford, Basil Blackwell.