Programming a Parallel Computer: The Ersatz Brain Project

James A. Anderson[1], Paul Allopenna[2], Gerald S. Guralnik[3], David Sheinberg[4], John A. Santini, Jr.[1], Socrates Dimitriadis[1], Benjamin B. Machta[3], and Brian T. Merritt[1]

[1] Department of Cognitive and Linguistic Sciences, Brown University, Providence, RI, 02912, USA
[2] Aptima, Inc., 12 Gill Road, Suite 1400, Woburn, MA, 01801, USA
[3] Department of Physics, Brown University, Providence, RI, 02912, USA
[4] Department of Neuroscience, Brown University, Providence, RI, 02912, USA

Abstract. There is a complex relationship between the architecture of a computer, the software it is to run, and the tasks it is designed to perform. The most difficult aspect of building a brain-like computer may not be in its construction, but in its use: How can it be programmed? What can it do well? What does it do poorly? In the history of computers, software development has proved far more difficult and far slower than straightforward hardware development. There is no reason to expect a brain like computer to be any different. This chapter speculates about programming techniques, potential applications, and intermediate level structures for hardware and software that might be useful for working with massively parallel, brain like computers.

1 Introduction

We want to design a suitable computer architecture – hardware and software – for the efficient execution of the applications now being developed that will display human-like cognitive abilities. We base our fundamental design on a few ideas taken from the neurobiology of the mammalian cerebral cortex; therefore we have named our project the Ersatz Brain Project. We gave it this name because, given our current state of knowledge, our goal can only be to build a shoddy second-rate brain. Even so, such a computer may be a starting point for projects that realize systems with the power, flexibility, and subtlety of the actual brain. We suggest that a "cortex-power" massively parallel computer is now technically feasible, requiring on the order of a million simple CPUs and a terabyte of memory for connections between CPUs.

We approach this problem from the point of view of cognitive computation. There is little attempt here to make a detailed "biologically realistic" neurocomputer. We are proposing instead a "cortically inspired" computing system specifically for cognitive applications. We know a good deal about the structure of human cognition and the operations it performs. Our belief is that these operations will be performed most efficiently by a computer system based on the design of cerebral cortex.

In particular, the brain has severe intrinsic limitations on connectivity, speed, and accuracy and many computational strategies may simply be forced on the system by hardware limitations. In many respects we have a dual computational system, one system old, highly evolved, highly optimized, basically associative, perceptually oriented, memory driven, and alogical, as well as a second system, recent, oddly contoured, unreliable, "symbolic" and "rule based". (See Sloman [1] for a cognitive science perspective.) We suggest a successful brain-like computer system should include aspects of both systems, since they are complementary and work effectively together, as the remarkable, very rapid success of our own species indicates.

2 Essentials of the Ersatz Approach

The human brain is composed of on the order of 10^{10} neurons, connected together with at least 10^{14} connections between neurons. These numbers are likely to be underestimates. Biological neurons and their connections are extremely complex electrochemical structures that require substantial computer power to model even in poor approximations. The more realistic the neuron approximation, the smaller is the network that can be modeled. Worse, there is very strong evidence from biology that **a bigger brain is a better brain,** thereby increasing greatly computational demands if biology is followed closely. **We need good approximations to build a useful brain-like computer**

2.1 The Ersatz Cortical Computing Module and the Network of Networks

Received wisdom has it that neurons are the basic computational units of the brain. However the Ersatz Brain Project is based on a different assumption. We will use the **Network of Networks [NofN]** approximation to structure the hardware and to reduce the number of connections required [2,3,4].

We assume that the basic neural computing units are not neurons, but small (perhaps 10^3 -10^4 neurons) attractor networks, that is, non-linear networks (modules) whose behavior is dominated by their attractor states that may be built in or acquired through learning [5,6,7]. Basing computation on module attractor states – that is, on intermediate level structure – and not directly on the activities of single neurons reduces the dimensionality of the system, allows a degree of intrinsic noise immunity, and allows interactions between networks to be approximated as interactions between attractor states. Interactions between modules are similar to the generic neural net unit except scalar connection strengths are replaced by state interaction matrices. (Figure 1) The state interaction matrix gives the effect of an attractor state in one module upon attractor states in a module connected to it. Because attractors are derived from neuron responses, it is potentially possible to merge neuron-based preprocessing with attractor dynamics. The basic Network of Networks system is composed of very many of these basic modules arranged in a two-dimensional array. (Figure 2).

2.2. Cortical Columns

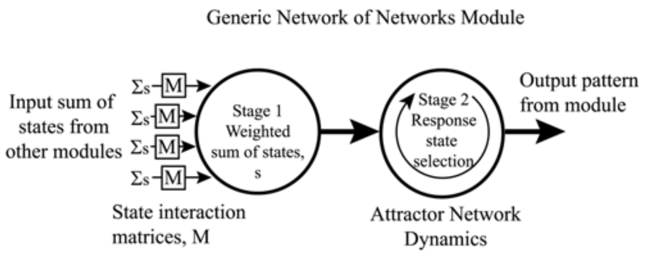
Generic Network of Networks Module
Input sum of states from other modules
Σs
M
Stage 1 Weighted sum of states, s
Stage 2 Response state selection
Output pattern from module
State interaction matrices, M
Attractor Network Dynamics

Fig. 1. Generic Network of Networks module structure.

The most likely physiological candidate for the basic component of a modular network is the cortical column. Cerebral cortex is a large two-dimensional layered sheet, with a repetitive structure. One of its most prominent anatomical features is the presence of what are called columns, local groups of cells oriented perpendicular to the cortical surface. There are several types of columns present at different spatial scales. A recent special issue of the journal *Cerebral Cortex* was devoted to cortical columns, their functions, and their connections. The introduction by Mountcastle [8] provides a useful summary of the two types of columns that will most concern us:

"The basic unit of cortical operation is the minicolumn … It contains of the order of 80-100 neurons, except in the primate striate cortex, where the number is more than doubled. The minicolumn measures of the order of 40-50 μm in transverse diameter, separated from adjacent minicolumns by vertical cell-sparse zones which vary in size in different cortical areas. Each minicolumn has all cortical phenotypes, and each has several output channels. … By the 26th gestational week the human neocortex is composed of a large number of minicolumns in parallel vertical arrays." ([8], p. 2)

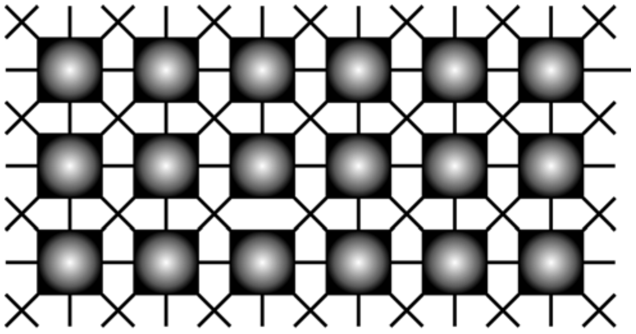

Fig. 2. Network of Networks 2-D modular architecture.

Minicolumns form a biologically determined structure of stable size, form and universal occurrence. What are often called "columns" in the literature are collections of minicolumns that seem to form functional units. Probably the best-known examples of functional columns are the orientation columns in V1, primary visual cortex. Vertical electrode penetrations in V1, that is, parallel to the axis of the column, found numerous cells that respond to oriented visual stimuli with the same orientation [9]. The cells in a column are not identical in their properties and, outside of orientation, may vary widely in their responses to contrast, spatial frequency, etc. Clusters of minicolumns make up functional columns:

Mountcastle continues, "Cortical columns are formed by the binding together of many minicolumns by common input and short range horizontal connections. The number of minicolumns per column varies probably because of variation in size of the cell sparse inter-minicolumnar zones; the number varies between 50 and 80. Long-range, intracortical projections link columns with similar functional properties. Columns vary between 300 and 500 μm in transverse diameter, and do not differ significantly in size between brains that may vary in size over three orders of magnitude … Cortical expansion in evolution is marked by increases in surface area with little change in thickness … ” ([8], p. 3)

If we assume there are 100 neurons per minicolumn, and roughly 80 minicolumns per functional column, this suggests there are roughly 8,000 neurons in a column.

2.3 Connectivity

Besides modular structure, an important observation about the brain in general that strongly influences how it works is its very sparse connectivity between neurons. Although a given neuron in cortex may have on the order of 100,000 synapses, there are more than 10^{10} neurons in the brain. Therefore, the fractional connectivity is very low; for the previous numbers it is 0.001 per cent, even if every synapse connects to a different cell. Connections are expensive biologically since they take up space, use energy, and are hard to wire up correctly. The connections that are there are precious and their pattern of connection must be under tight control. This observation puts severe constraints on the structure of large-scale brain models. One implication of expensive connections is that short local connections are relatively cheap compared to longer range ones. The cortical approximation we will discuss makes extensive use of local connections for computation in addition to the sparse, accurately targeted long-range connections.

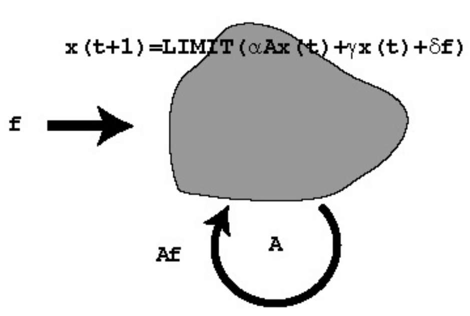

Fig. 3. Basic BSB module

2.4 Interactions between Modules

Let us discuss in a little more detail how to analyze interactions between small groups of modules. The attractor model we will use is the BSB network (Anderson, 1993) because it is simple to analyze using the eigenvectors and eigenvalues of its local connections.

The BSB model (Figure 3.) was proposed several years ago as a simple feedback nonlinear neural network [6]. Its dynamics broke conveniently into a linear and a non-linear part. The analysis assumed it was a recurrent feedback network (See Figure 3). An input pattern, **f**, appears on an interconnected group of neurons, say from a sensory input. There is vector feedback through a connection matrix, **A,** weighted by a constant α and an inhibitory decay constant, γ, with amount of inhibition a function of the amplitude of the activity. The state of the system is $\mathbf{x}(t)$. The system is linear up to the point where the LIMIT operation starts to operate. LIMIT($\mathbf{x}(t)$) is a hard limiter with an upper and lower threshold. Sometimes it is useful to maintain the outside input $\mathbf{f}(0)$at some level; sometimes it is useful to remove the outside input. The constant δ performs this function.

The basic algorithm for BSB is:

$$\mathbf{x}(t+1) = \text{LIMIT}\,(\alpha\mathbf{A}\mathbf{x}(t) + \gamma\mathbf{x}(t) + \delta\mathbf{f}(0)\,)\,. \tag{1}$$

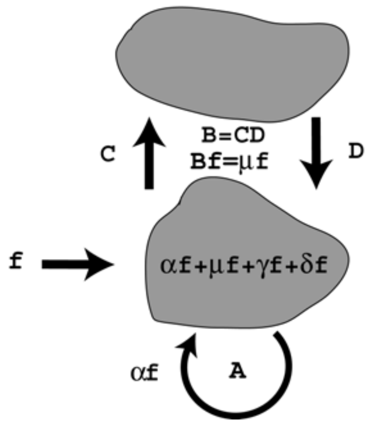

Fig. 4. Two modules linked by an associative pattern feedback loop.

In the nonlinear BSB network with growing activity, the state of the system will reach an attractor state based on the LIMIT function, usually the corner of a hypercube of limits. In practice, if **f** is an eigenvector the final BSB attractor state is close to the direction of **f**.

Activity can increase without bound or go to zero. The transition from increasing activity to decreasing activity is under control of α, γ and, the eigenvalues of **A**. These parameters provide a way of controlling network behavior.

Let us consider the implications of connections from other structures. In particular, we know two relevant facts about cortex: (1) one cortical region projects to others and, (2) there are back projections from the target regions to the first region that are at least of equal size as the upward projections. Therefore let us consider the anatomy shown in Figure 4. Note that the upward association can have any vector as an output association. The downward association has the eigenvector, **f**, as its output, perhaps mixed with other eigenvectors. Hebbian learning operating at the higher and lower ends of the loop will tend to construct **f** as an eigenvector because it is present at both input and output. These loops are reminiscent of the Bidirectional Associative Memory [BAM] of Kosko [10]. Analysis again is easy if **f** is an eigenvector. Let us assume δ is zero, as before. Analysis is the same as in the basic model, except that we now have a new term in the BSB equation corresponding to the contribution from the loop. (Figure 4.)

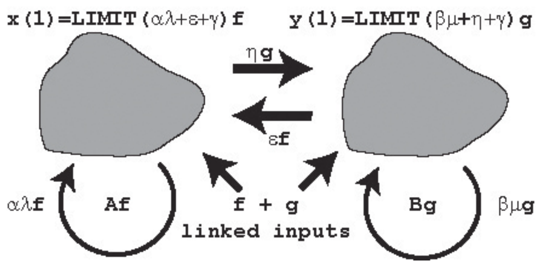

Fig. 5. Two coupled modules receiving a pair of linked input patterns, **f** and **g**.

We can also propose a computational mechanism for binding together a multi-module input pattern using local connections. If two modules are driven by two simultaneously presented patterns, **f** and **g**, associative links between **f** and **g** can be formed, increasing the gain of the module and therefore the likelihood that later simultaneous presentation of the patterns will lead to module activity reaching a limit. (Figure 5) Local pattern co-occurrence will form local pattern associative bonds, letting larger groupings act as a unit, that is, a unit that increases and decreases in activity together. Large-scale patterns will tend to bind many module activities together since they take place embedded in a larger informational structure.

2.5 Interference Patterns and Traveling Waves

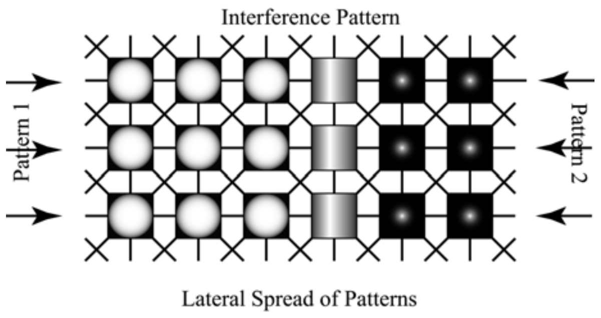

Fig. 6. Two patterns move laterally across an array and form an interference pattern, and can learn the resulting feature combination.

Because we have suggested many important connections are local, much information processing takes place by movement of information laterally from module to module as shown in Figure 6. This lateral information flow requires time and some important assumptions about the initial wiring of the modules. There is currently considerable experimental data supporting the idea of lateral information transfer in cerebral cortex over significant distances. The lateral information flow allows the potential for the formation of the feature combinations in the interference patterns, useful for pattern recognition. There is no particular reason to suggest that modules just passing information are in attractor states; for pattern transmission it is better if they re not.

Coincidences, where two patterns collide are of special interest. Since the individual modules are nonlinear learning networks, we have here the potential for forming new attractor states when an interference pattern forms, that is, when two patterns arrive simultaneously at a module over different pathways. (Figure 6.)

There have been a number of related computational models specifically designed for vision that have assumed that image processing involves lateral spread of information. An early example is Pitts and McCulloch [11] who suggested, “A square in the visual field, as it moved in and out in successive constrictions and dilations in Area 17, would trace out four spokes radiating from a common center upon the recipient mosaic. This four-spoked form, not at all like a square, would the be the size-invariant figure of a square (p. 55).” In the 1970’s Blum [12] proposed the “grassfire” model where visual contours ignited metaphorical “grassfires” and where the flame fronts intersected produced a somewhat size invariant representation of an object. The propagating waves are computing something like the **medial axis representation**, that is, the point on the axis lying halfway between contours [13].

There are many examples of traveling waves in cortex. Bringuier, Chavane, Glaeser, and Fregnac [14] observed long range interactions in V1 with an inferred conduction velocity of approximately 0.1 m/sec. Lee, Mumford, Romero, and Lamme [15] discuss units in visual cortex that seem to respond to the medial axis. Particularly pertinent in this context is Lee [16] who discuses medial axis representations in the light of the organization of V1. In psychophysics, Kovacs and Julesz [17] and Kovacs, Feher, and Julesz [18] demonstrated threshold enhancement at the center of circle and at the foci of ellipses composed of oriented Gabor patches forming a closed contour. These models assume that an unspecified form of "activation" is being spread whereas the Network of Networks assumes that pattern information (a vector) related to module attractor states is being propagated. We feel that the traveling wave mechanism and its generalizations may have more general applications that vision.

2.6 Ersatz Hardware: A Brief Sketch

How hard would it be to implement such a cortex-like system in hardware? This section is a "back of the envelope" estimate of the numbers. We feel that there is a size, connectivity, and computational power sweet spot about the level of the parameters of the network of network model. If we equate an elementary attractor network with 10^4 actual neurons, that network might display perhaps 50 well-defined attractor states. Each elementary network might connect to 50 others through 50x50 state connection matrices. Therefore a cortex-sized artificial system might consist of 10^6 elementary units with about 10^{11} to 10^{12} (0.1 to 1 terabyte) total strengths involved to specify connections. Assume each elementary unit has the processing power of a simple CPU. If we assume 100 to 1000 CPU's can be placed on a chip there would be perhaps 1000 to 10,000 chips in a brain sized system. These numbers are within the capability of current technology.

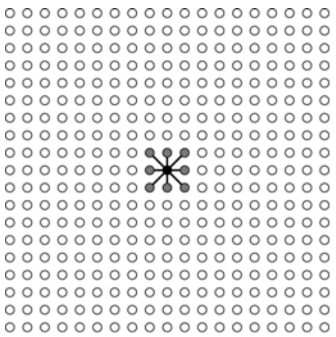
Fig. 7. A 2-D array of modules with simple local connectivity. The basic Ersatz architecture.

Therefore, our basic architecture consists of a large number of simple CPUs connected to each other and arranged in a two dimensional array. (Figure 7). A 2-D arrangement is simple, cheap to implement, and corresponds to the actual 2-D anatomy of cerebral cortex. An intrinsic 2-D topography can also make effective use of the spatial data representations used in cortex for vision, audition, skin senses and motor control.

2.7 Communications

The brain has extensive local and long-range communications. The brain is unlike a traditional computer in that its program and its computation are determined primarily by strengths of its connections. Details of these relatively sparse interconnections are critical to every aspect of brain function.
2.7.1. Short-range connections. There is extensive local connectivity in cortex. An artificial system has many options. The simplest is NEWS connectivity. Our experience has been that expanding NEWS to include at least the diagonal CPUs, and preferably even further, for example, to rings of modules two or three modules away, works significantly better.
2.7.2. Long-range connections. Many of the most important operations in brain computation involve pattern association where an input pattern is transformed to an associated output pattern that can be different from the input. One group of units connects to other groups with precisely targeted long-range connections.
2.7.3. CPU Functions. The CPUs must handle two quite different sets of operations. First is communications with other CPUs. Much of the time and effort in brain-based computation is in getting the data to where it needs to be. Second when the data arrives, it can then be used for numerical computation.

3 Topographic Computation

The cortex, and, in imitation, our computational model is a 2-D sheet of modules. Connections between modules, slow and expensive in the brain, as we have noted, perform the computation. Therefore topographic relationships between modules and their timing relationships become of critical computational significance. We suggest that it is possible to use the topographic structure of the array to perform some interesting computations. Indeed, **topographic computation** may be a major mechanism used to direct a computation in practice, and, therefore, a tool that can be used to program the array.

The next three sections will describe some uses of topographic organization to (1) Arithmetic fact learning, (2) Information integration, (3) Filtering based on spatial organization, (4) Formation of spatial assemblies of modules within and across layers. This discussion is obviously highly speculative, but suggests the kinds of intermediate level structures that may be useful in a practical, programmable, and flexible brain-like computer.

4 Example: Arithmetic in Humans Using Topographic Control

A few years ago, we suggested a model for elementary arithmetic fact learning [19] that showed that it was possible to perform what appeared to be "symbolic" elementary arithmetic operations by using topographically organized spatial weighting functions. Because this kind of mechanism is suggestive of how an array of modules might be programmed topographically, we review it here even though the array of modules in the arithmetic model was one-dimensional. We will expand the array to two dimensions in the next sections. In addition the problem itself illustrates a few of the peculiarities of real-world cognitive computation.

When a computer multiplies two single digit integers, it performs a sequence of formal operations based on logic, the definitions of integers and the rules of binary arithmetic. Humans do multiplication very differently, using a combination of estimation and memory. The human algorithm might be formalized as something like "Choose the **product number** (that is, a number that is the answer to *some* multiplication problem) that is about the right size." For example, the most common error for 9x6 is 56. More careful inspection of the experimental data indicates a complex associative structure working along with estimation, Errors for 9x6 are frequently "off by one" multiples of 6 or of 9, for example, 48 or 63. These effects do not seem to arise because of errors in the application of formal logic to the computation.

The error rate for elementary multiplication facts in college-educated adults is as high as 8% in some studies. Such a high rate is remarkable considering that several years are devoted to practicing arithmetic facts in elementary school, at a time when learning of other complex material – language, for example – is fast and accurate. However, there are also some positive aspects to such an error prone algorithm. For example, errors in elementary arithmetic are usually "close" to the correct answer. Sometimes being "close" is good enough to do the job and more useful than the gross magnitude errors that malfunctioning computers make.

Attempts by cognitive scientists to model human number performance have generally assumed the presence of an important "perceptual" part to the internal human number representation. In many experiments, numbers act more like "weights" or "light intensities" than abstract quantities. There is also evidence (Hadamard [20]) that higher mathematics as actually done by mathematicians or physicists is often more perceptual than abstract. The classic theorem-proof process is primarily used to check for errors and as a way of convincing others that results are correct. These ill-defined sensory and perceptual aspects of mathematics as it is practiced go by the name "mathematical intuition."

4.1 The Data Representation for Number

The most difficult problem in any application of neural networks is converting the input data into the state vectors that can be manipulated by network dynamics. Data representation often has more influence on system performance than the type of learning algorithm or the form of the network. There are few explicit rules that determine what the best data representations are for a particular problem. For example, there is constant intrinsic tension between accuracy and generalization since generalization to be useful requires an appropriate response to an unlearned pattern. Good generalization is critically dependent on the details of the specific application, system history, and the data representation. In practice, generalization needs to be controllable so the same network can generalize one way at one time and in a different way with different task requirements.

A combination of computer simulations and inference from experimental data has suggested one useful data representation for number. The starting point for our formal system will therefore be a suggestion for the representation of number in a neural network or in a one-dimensional array of modules.

Topographic representations of parameters are very common in the nervous system. For example, in vertebrates, the early stages of the visual cortical representation are topographic in that the image on the retina is roughly mapped onto the two dimensional cortex. A topographic map of the visual field is used in vision in animals as diverse as humans and the horseshoe crab, *Limulus*. There are many other topographic maps in vertebrate cortex, for example, somatosensory maps of the body surface and frequency of sound in audition.

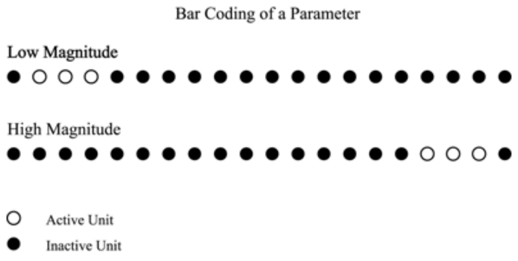

Fig 8. The location of the active units on a linear array of units codes the magnitude of a parameter.

In a neural network, one useful form of such a topographic representation is called a **bar code,** as shown in Fig. 8. The value of the parameter represented depends on the location of a group of active units on a linear array of units. The price paid for this useful data representation is low precision and inefficiency in use of units. If a single parameter is represented only as a location then many units are required to represent a value that a single unit could represent by an activity level. Such a physical mapping is also inherently low precision, with precision determined by the number of units devoted to representing the parameter. Such a representation technique, and its variants are most naturally implemented in a system that is composed of many relatively inexpensive units and that is performing a function that is only secondarily concerned with high accuracy. Some nanocomponent based computing devices will fall into this category, as does, of course, the nervous system.

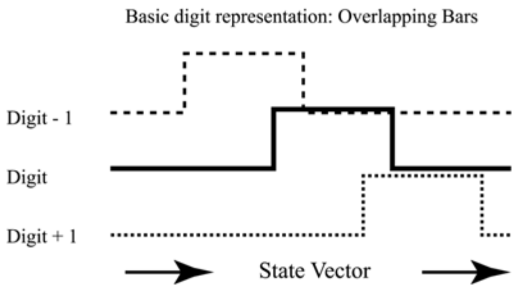

Fig 9. Bar coded data representation for number magnitude. Bars are arranged spatially in order of magnitude as in the number line.

Such a representation can convey an essentially continuous range of values. However, in cognition we often use discrete entities – words, concepts, or numbers – to represent many different but somehow related examples. Much of the computational power of human cognition arises from its ability to keep only enough information about the world to be useful, and no more. For example, every physical example of a table is different in detail from every other one, but the word "table" often captures the essence of the commonalities of the group and the individual details can be discarded. With our network model, we can couple a continuous underlying data representation with a nonlinear attractor neural network with discrete attractors.

4.2 Simple Arithmetic Operations

Learning numbers is only the beginning of arithmetic. If we want to build a useful computational system, we have to be able to direct the computation to give answers to specific problems without further learning. Therefore, our first job is to specify the operations that we would reasonably expect an "arithmetic" network to perform. Let us suggest several primitive candidate operations for our computing system.

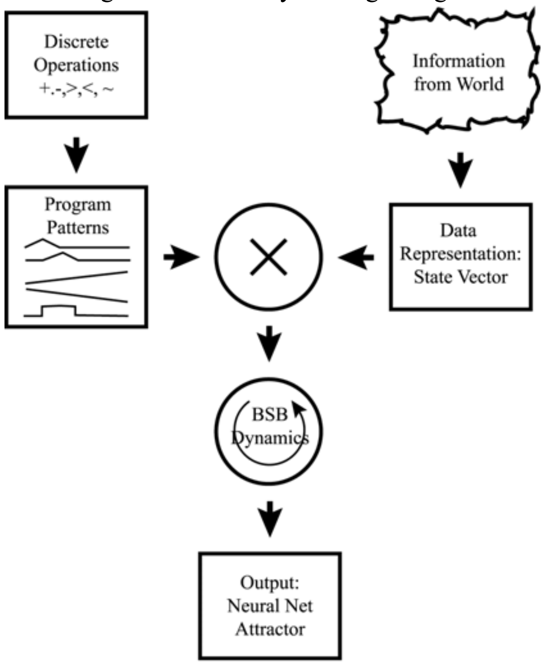
Discrete Operations +.-,>,<, ~
Information from World
Program Patterns
Data Representation: State Vector
BSB Dynamics
Output: Neural Net Attractor

Fig. 10. Numerical computation involves merging two components. The left branch contains the program patterns and the right branch contains input quantities. The attractor network generates the output.

Counting seems to be present in all human societies of which we have knowledge and evidence for it goes far back into prehistory. There is good evidence that nonhuman primates and many higher mammals can count up to small integers. Formally, we can represent the counting function as two related operations: starting with a digit, add one to it, that is, **increment**, and, the symmetrical operation, subtract one, that is, **decrement**. Another valuable arithmetic related function is comparison of two numbers, that is, the equivalent of the formal operations **greater than** or **lesser than**. Another useful operation is **round off**, that is, two and a bit more can be reported as "about two."

Therefore we will suggest that five operations form a useful starting point:

- **increment** (add 1)
- **decrement** (subtract 1)
- **greater than** (given two numbers, choose the larger)
- **lesser than** (given two numbers, choose the smaller)
- **round off** to the nearest integer

4.3 Programming Patterns for the Simple Arithmetic Operations

The digits in order are represented as adjacent locations in the array, that is, the spatial locations follow the order of magnitudes: 1 next to 2, 2 next to 3, 3 next to 4, and so forth. If we use the bar coded data representation we have discussed (Figure 9), it is surprisingly easy to find programming patterns that work. This robustness arises because we are dealing with qualitative properties of the geometry of representation, that is, **representational topology**.

In a previous paper [3] a way was suggested to control a network using a **vector programming pattern** that multiplied term by term the state vector derived from the input data. The number data representation we are using – the overlapping bar codes suggested earlier – contains enough information about the relations between digits to perform these operations.

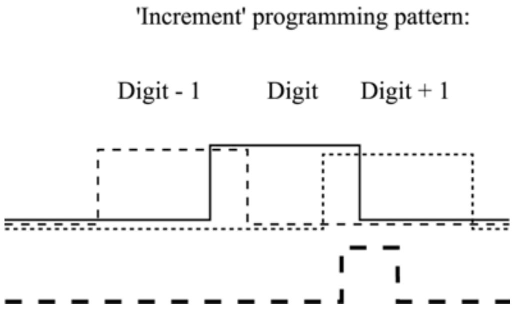

Fig. 11. The programming pattern for **increment** weights the input digit and shifts it to the next attractor to the right.

The overall architecture of the system we have been describing is presented in Fig. 10. The system has two branches. One is connected to the physical world thorough the sensory systems. The other forms the "abstract" branch and chooses the desired arithmetic manipulation. An operation is chosen. This operation is associated with a programming pattern. In the other branch of the computation, information from the world is represented as a bar code. The two vectors are multiplied term by term. BSB attractor dynamics are then applied. The state vector then evolves to an attractor that contains the answer to the problem.

We can present intuitive arguments to support the particular programming patterns used. Consider counting, that is, the **increment** operation. If we start from a particular location on the topographic map, we know that one direction on the map corresponds to larger numbers, the other to smaller. Therefore, if we differentially weight the map so that nearby larger numbers are weighted more strongly, the system state can be encouraged to move toward the attractor corresponding to the next largest digit. (The heavy dashed line in Figure 11.)

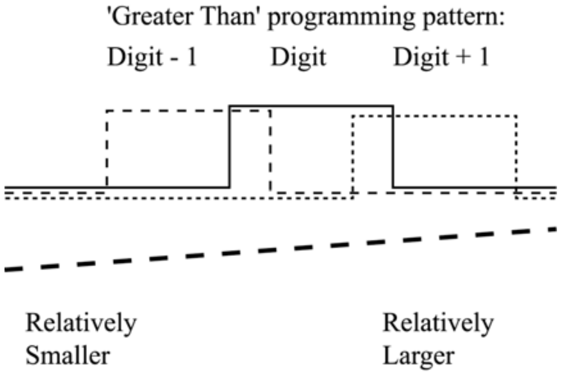

Fig. 12. The programming pattern for **greater than** weights the larger digit more heavily and drives the state vector to that attractor.

The **greater-than** operation

can also be done with differential weighting of the input data, so large digits reach attractors before small digits as shown in Figure 12.

Round off is performed slightly differently. A bar for an intermediate value is generated at a location between the two digits. The network then moves to the attractor with the largest overlap with the input data bar. The two other operations – **less than** and **decrement** – are simple reflections of the patterns for **greater than** and **increment**.

There are some important properties of the operation of the program that are worth mentioning. For example, when the 'greater-than' program runs, it displays what is called a **symbolic distance** effect in cognitive psychology. The time to an attractor is a function of the difference in magnitude between the two numbers being compared, that is, the larger of the pair (9,1) is reaches its termination state faster than the larger of (5,4). In our model, this effect is driven by the magnitude representation in the data representation. The intrusion of statistical and perceptual components into what appears at first to be a purely abstract computation is to be expected from our approach, though perhaps a result not welcomed by purists since it would seem to have no place in logic. Notice also that the topographic map is effectively a realization of the number line analogy for integers.

Although these programming techniques work reliably for the digits one through ten, they are appallingly slow, limited in precision and dynamic range, and are clearly of no value for constructing a practical computing system that works with precise numbers. But, then, humans are not very good at arithmetic either.

5 Example: Sensor Fusion using Topographic Arrays

One potential application of our Ersatz Brain architecture is to **sensor fusion.** Sensor fusion involves integration of information from different types of sensor into a unified interpretation.

Suppose we have a set of four numerical sensor readings. We can do quite a bit with these values alone, for example, using them as a set of features for object recognition. However, this is not really sensor fusion since the sensor data is not integrated but used as information to determine a previously learned classification. This may not be possible if there is no previous classification, that is, in unsupervised learning.

Spatializing the data, that is letting it find a natural topographic organization that reflects the relationships between multiple data values, is a technique of great potential power, though an unfamiliar one. It is, however, a natural and effective way to compute in a two dimensional cortex and our two dimensional Ersatz architecture.

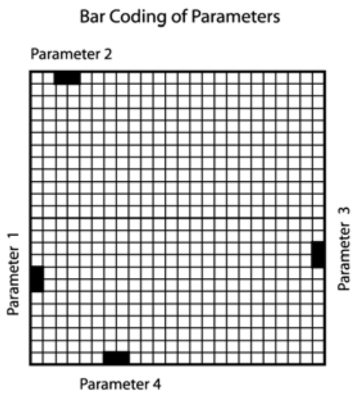

Fig. 13. Four bar codes.

Assume we have four parameters that we want to represent in the activity pattern that describes single entity (Figure 14).

Our architecture assumes local transmission of patterns from module to module according to the Network of Networks model assumptions. Modules have multiple stable attractor states. Patterns are transmitted laterally from module to module. When two different patterns arrive at a module simultaneously, there is the possibility for a new pattern to be generated in the module, now representing the feature coincidence as a new part of the module's repertoire of attractor states. Standard learning rules plus the non-linear network can perform this operation. There may be many two parameter interference patterns, the straight lines in Figure 14. Each pair of input patterns gives rise to an interference pattern, a line perpendicular to the midpoint of the line between the pair of input locations.

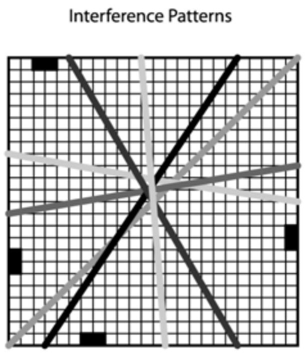

Fig. 14. Formation of simple interference patterns.

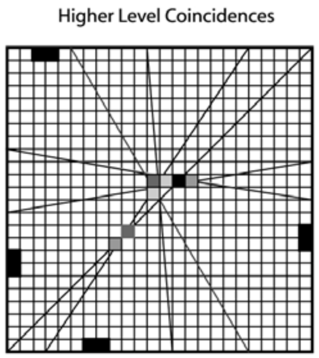

Fig. 15. Formation of higher-level feature combinations.

5.1 Higher Level Features

In addition to the interference patterns representing coincidences between pairs of patterns there are often places where three or even four features coincide at a module. (Figure 15) The higher-level combinations represent partial or whole aspects of the entire input pattern, that is, they respond to the Gestalt of the pattern. In this sense they have fused a number of aspects of the input pattern and represented it as a new activity pattern at a specific location.

5.2 Formation of Hierarchical "Concepts"

One intriguing aspect of this coding technique is the way it allows for the formation of what look a little like hierarchical concept representations. Suppose we have a set of "objects". For a simple demonstration, assume we have three parameter values that are fixed for each object and one value that varies widely from example to example. After a number of examples are seen, the system develops two different spatial codes. (Figure 16).

In the first, a number of high-order feature combinations are fixed since their three input **core** patterns never change. In the second, based on the additional spatial relations generated by the widely different examples, there is a varying set of feature combinations corresponding to the details of each specific **example** of the object. If the resulting spatial coding is looked at by an associative system, then two kinds of pattern can be learned.

The first learned pattern corresponds to an unchanging **core** and might correspond to an abstraction of the commonalities of many examples. The second learned set of patterns corresponds to the core, plus the examples -- patterns associated with each specific learned parameter set. All the specific examples are related by their common core pattern. This dichotomy has considerable similarity to the subordinate-superordinate relationships characterizing information stored in a hierarchical semantic network (Murphy [21]).

We have spatialized the logical structure of the hierarchy. Because the coincidences due to the "core" (three values) and to the "examples" (all four values) are spatially separated, we have the possibility of using the "core" as an abstraction of the examples and using it by itself as the descriptor of the entire set of examples. Many of the control mechanisms suggested for cortex, most notably attention, are held to work by exciting or inhibiting specific regions of the cortical array, a natural potential mechanism for this approach.

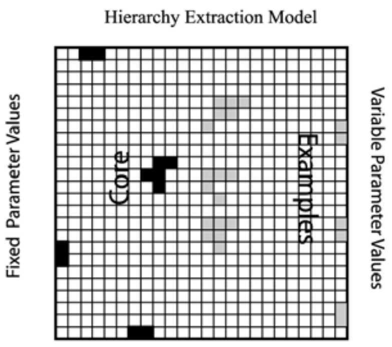

Fig. 16. Mechanism for formation of hierarchical structure.

Table 1. Average Formant Frequencies (Hz) and Ratios of Formant Frequencies for Men, Women and Children for Three Vowels. Data taken from Watrous [22] derived originally from Peterson and Barney [23]

		[i]		[æ]		[u]	
Men	f1	267	(0.86)	664	(0.77)	307	(0.81)
Women	f1	310	(1.00)	863	(1.00)	378	(1.00)
Children	f1	360	(1.16)	1017	(1.18)	432	(1.14)
Men	f2	2294	(0.82)	1727	(0.84)	876	(0.91)
Women	f2	2783	(1.00)	2049	(1.00)	961	(1.00)
Children	f2	3178	(1.14)	2334	(1.14)	1193	(1.24)
Men	f3	2937	(0.89)	2420	(0.85)	2239	(0.84)
Women	f3	3312	(1.00)	2832	(1.00)	2666	(1.00)
Children	f3	3763	(1.14)	3336	(1.18)	3250	(1.21)

6 Example: Selective Filtering of Vowels

Let us consider a test problem for some of our topographic based ideas on a real bit of biological signal processing. Let us consider how to build a geometric data representation inspired by one of the perceptual invariances seen in human speech.

6.1 Formant Frequencies in Speech Perception

The acoustic signals due to a vowel (and other phonemes) are dominated by the resonances of the vocal tract, called formants.

The resonant peaks are sometimes not very sharp and not very large. Vocal tracts come in different sizes, from men, to women, to children. Resonant peaks change their frequency as a function of vocal tract length.

Remarkably, this frequency shift – which can be substantial between a bass male voice and a small child --causes little problem for human speech perception. The important perceptual feature for phoneme recognition seems to be the based more on the ratios between the formant frequencies than on their absolute values. This relationship is what we would expect if a vocal tract simply increased in size without changing its shape. Then differences in vocal tract length approximately multiply the resonances by a constant.

We can see these properties using a classic set of data [22,23] for the formant frequencies of vowels. Table 1 gives the average frequencies (in Hz) of the formants F1, F2, and F3 for three vowels as spoken by men, women and children. The value of the formant for women is given the value 1.00. It can be seen the absolute values of

Table 2. Ratios Between Formant Frequencies (Hz) for Men, Women and Children. Data taken from Watrous [22] derived originally from Peterson and Barney [23]

		[i]	**[æ]**	**[u]**
Men	f1/f2	0.12	0.38	0.35
Women	f1/f2	0.11	0.42	0.39
Children	f1/f2	0.11	0.43	0.36
Men	f2/f3	0.78	0.71	0.39
Women	f2/f3	0.84	0.72	0.36
Children	f2/f3	0.84	0.70	0.37

the frequency of the formants for the same vowel vary about 30% between men women and children. If we look instead at the ratios of the formants the picture changes. Table 2 presents the values for the ratios of f1 and f2 and f2 and f3 for the same three vowels. Note the ratios vary only by a few percent over different vocal tract sizes. The ratio of formants is more constant than the absolute values of frequency.

6.2 Data Representation

We know from many sources there is a roughly logarithmic spatial mapping of frequency onto the surface of auditory cortex, what is sometimes called a tonotopic representation. A logarithmic spatial coding has the effect of translating the all the parameters multiplied by the constant by the same distance.

It might be easy to simply compute ratios of frequencies and use those values as inputs to a speech processor. However, there might be advantages to using a more brain-like approach. The transformations are not quite as simple as simple ratio invariance, the system is intrinsically noisy, the stimuli show substantial variability both within and between speakers (for example, there are many different accents), and the system is adaptive.

6.3 Representational Filtering

Our specific goal is to **enhance** the representation of ratios between "formant" frequencies, and to **de-emphasize** the exact values of those frequencies. That is, we wish to make a kind of filter using the data representation that responds to one aspect of the input data.

Let us start by assuming our usual information integration square array of modules with parameters fed in from the edges.

We start by duplicating the frequency representation three times along the edges of a square. Multiple frequency maps are common in the auditory system (See Talavage et al. [24] for an fMRI study of tonotopic maps in humans.).

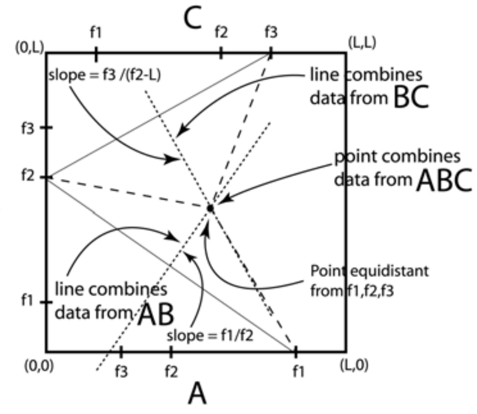
C
f1
f2
f3
(0,L)
(L,L)
slope = f3 /(f2-L)
line combines data from BC
f3
f2
point combines data from ABC
B
Point equidistant from f1,f2,f3
f1
line combines data from AB
slope = f1/f2
(0,0)
f3
f2
f1
(L,0)
A

Fig. 17. The location of the point equidistant from f1, f2 and f3 contains information about the ratios between them.

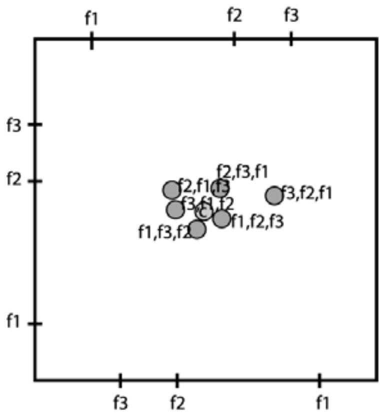

Fig. 18. Coincidence locations for various combinations of f1. f2. and f3.

We will start by assuming that we are interested in locations where three frequency components come together at a single module at the same time. Figure 17 shows the point equidistant from f1 on the bottom edge, f2 on the side, and f3 on the top along with a few geometrical relations between the frequencies. We conjecture that this means the module may form a new internal representation corresponding to this triple coincidence. If we assume initially pattern transmission between modules is isotropic we then look for module locations equidistant from the locations of initial triple sets of frequencies. These points will be the modules that lie at the center of a circle whose circumferences contains the three points containing the three different frequency components, one from each side.

There are multiple possible combinations of the three formant frequencies. With three frequencies, the three possibilities are (f1,f2,f3), (f1,f3,f2), (f2,f1,f3), (f2,f3,f1), (f3,f1,f2), and (f3,f2,f1). Each triple combination produces a slightly different geometry.

Depending on what triple (location of the frequency combination) is used, different points are activated. Therefore a three "formant" system has six locations corresponding to possible triples. (We discuss in Section 8 the possibility of forming linked assemblies of active modules to function as a higher-level data representation.)

Figure 18 shows the points (marked with dots) that are equidistant from various combinations of frequencies. There are actually a lot more possible combinations possible involving both lower and higher order conjunctions, but we will start by only looking at triples.

The positions of the coincidences shift slightly if the frequency marks are translated by a uniform amount, corresponding in our model system to a change in vocal tract length giving rise to a shift in frequencies.

Because the geometry of the triple coincidence points varies with the location of the inputs along the edges, a different set of frequencies will give rise to a different set of triple coincidences. Figure 19 adds to Figure 18 the set of triple coincidences for another set of frequencies. Note that one triple for the second set of frequencies lies outside the array.

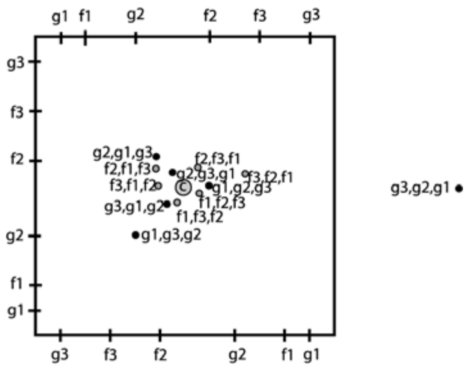

Fig. 19. (Top) Modules with triple coincidences for two sets of frequency patterns, f1, f2, and f3 and g1, g2, and g3.

7 Sparse Connectivity and Sparse Coding

7.1 Full Connectivity

Let us make some general, and highly speculative comments about the development of possible useful intermediate level structures for brain like computation.

Most neural network learning models assume full or nearly full connectivity between layers of units. Units are most often arranged in layers because it is an arrangement that pays homage to the neuroanatomy of mammalian cerebral cortex where cortical regions project to other cortical regions over long-range projection pathways through the white matter. Full connectivity means that a unit in a layer projects to, and it projected to, by all the units in layers above and below it.

Fully connected systems can be analyzed with standard mathematical techniques. They can perform a number of powerful information processing operations, and, combined with simple local learning algorithms such as the Hebb rule, they can be used to build adaptive systems with a number of useful applications in both engineering and science. The number of connections in fully connected systems grows very rapidly, order n^2, where n is the number of units.

7.2 Sparse Connectivity

Although many neural net models assume full or high connectivity, the actual cerebral cortex is **sparsely connected**, that is, each neuron projects to relatively few others given the potential number they could connect to, even in projections from one cortical region to another. (See Section 2.3).

7.3 Sparse Coding for Data Representation

Besides sparse connectivity, there is reasonably strong experimental evidence that **sparse coding** is used for data representation in the cortex, that is, information is represented by the activities of relatively few units. A review by Olshausen and Field [25] comments, "In recent years a combination of experimental, computational, and theoretical studies have pointed to the existence of a common underlying principle involved in sensory information processing, namely that information is represented by a relatively small number of simultaneously active neurons out of a large population, commonly referred to as 'sparse coding.'" ([25], p. 481). Many of these ideas first emerged in a classic paper by Barlow [26].

There are numerous advantages to sparse coding. Olshausen and Field mention that sparse coding provides increased storage capacity in associative memories and is easy to work with computationally. Among other virtues, sparse coding also "makes structure in natural signals explicit" ([25], p. 481) and is energy efficient (see [27])

"Higher" regions (for example, inferotemporal cortex) seem to show a greater degree of sparse coding than lower ones (for example, V1). Cells in the higher levels of the visual system also have less spontaneous activity than lower regions, for example, cells in inferotemporal cortex are silent much of the time until they find a specific stimulus that piques their interest.

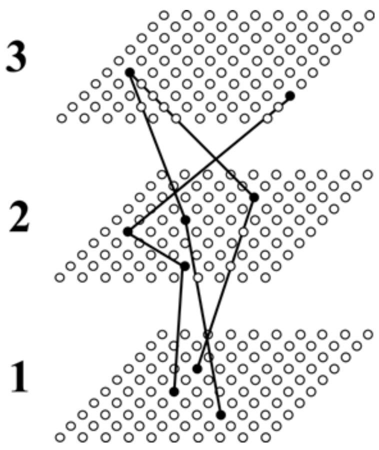

Fig. 20. A sparsely connected, sparsely coded three-layer svstem.

7.4 Sparse Coding combined with Sparse Representation

Figure 20 shows a cartoon version of a system that shows both **sparse coding** (three active units in input layer 1, four in layer 2, and two in output layer 3) and **sparse connectivity**.

Instead of trying to derive very general pattern association systems like back propagation, using high connectivity, let us see if we can make a learning system that starts from the assumption of both **sparse connectivity** and **sparse coding**.

Such extreme restrictions on connectivity and representation do not seem at first to form a very promising information processing system. We suspect this is why it has not been looked at seriously as a nervous system model, even though it seems to fit the qualitative structure of the cortex better than the high connectivity assumptions that underlie common neural network models.

7.5 Paths

In sparse systems, selectivity can come from other sources than a precise pattern of connection strengths. A useful notion in sparse systems is the idea of a **path**. A **path** connects a sparsely coded input unit with a sparsely coded output unit. Paths have **strengths** just as individual connections do, but the strengths are based on the entire path, from beginning to end, which may involve several intermediate connections. A path may have initial strength zero or start with a nonzero strength due to initial connectivity. The concept of a path is related to useful ways to structure initial wiring of the nervous system. Connections that are likely to be valuable are sketched in initially. Learning can then tune and sharpen the pre-existing connections. This strategy seems to be present in sensory systems. For an example, see a recent discussion of the development of different aspects of the visual system [28].

Consider a path with several intermediate units. If there is no initial transmission of activity through the path, there will be no learning. If, however, when the input unit becomes active and gives rise to even low activity at the output unit, learning can take place and the path can become stronger. Continued learning will further strengthen that particular path. If the path is strictly feed forward, connections earlier in the path cannot be changed properly without additional assumptions, for example, some form of backward information flow. However cortical regions generally project both forward and backwards.

7.6 Initial Biases in Connectivity

For a simple system with scalar weights by far the best, though unrealistic, strategy would be to somehow construct independent paths for each single unit association.

If many independent paths were desirable, then a useful initial construction bias for such a sparsely connected system would be to make available as **many** potential paths as possible. This bias differs significantly from back propagation, where there are almost always fewer units in the hidden layers than in the input and output layers, therefore **fewer** potential paths. (Many of the most important statistical properties of back propagation arise from the fact that there are a relatively small number of hidden layer units [7].) In a fully connected system, adding more units than contained in the input and output layers would be redundant. This is not so in sparse systems. But we know there is a huge expansion in number of units going from retina to thalamus to cortex. A million input fibers drive 200 million V1 neurons.

7.7 Sparseness in Systems Using the Network of Networks

Suppose we assume that modules, not single units, are connected. Simple scalar activity is not, by nature, selective. Activity arising from different sources cannot be told apart; there is no “tag” giving its source. But if we use the Network of Networks approximations, activity along a path is not a scalar. Activity is now not simple unit activity but **module activity.** Patterned activity in a single module connected to another single module, can give rise to a selective pattern associator.

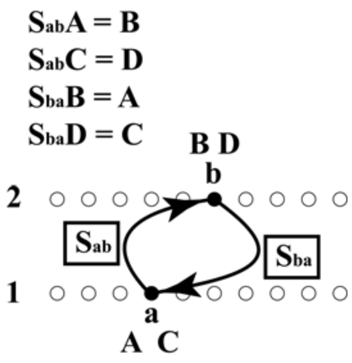

Fig. 21. About the simplest arrangment of interconnected modules. The **S**’s are matrices and **A,B**, etc. are patterns.

For ease in analysis, let us assume the pattern associator is the simplest kind, the linear associator. Because we aiming for the development of correct associations over entire paths in a system using non-linear modules, we suspect the exact details of the network associators are not important.

If a pattern **A** on module **a** is input to Layer 1, (see Figure 21) the activity, **B**, on **b** is given by the matrix-vector product, $\mathbf{S_{ab}A}$. We assume as part of the Network of Networks architecture that there are reciprocal connections between modules. If pattern **B** is present on module **b** of layer 2, the activity on **A** is given by $\mathbf{S_{ba}B}$. This loop between **a** and **b** is similar to a Bidirectional Associative Memory [10].

If pattern **A** on **a** and pattern **B** on **b** are simultaneously present, the increment in strengths are given by standard Hebb learning, that is,

$$\Delta \mathbf{S_{ab}} \sim \mathbf{BA}^{T}. \text{ and } \Delta \mathbf{S_{ba}} \sim \mathbf{AB}^{T}. \qquad (2)$$

If no other pattern associations are stored in the connection matrices, after learning, if pattern **A** is present at module **a**, something like pattern **B** will appear at module **b**. The sparsely represented association (the only activity is on **a** and **b**) has been learned.

7.8 Multiple Patterns

Unlike simple scalar connections between units, a single connection between modules can learn multiple associations.

In the simplest situation, suppose we look at modules **a** and **b** as before, but assume that **a** and **b** can display two different patterns, **A** and **C** on **a** and **B** and **D** on **b**. Subject to the usual limitations of simple pattern associators, (that is, it works best if **A** and **B,** and **C** and **D,** are orthogonal) such a network can easily learn to associate **A** with **B** and **C** with **D** using the same connections. We can use pattern selectivity to overcome some of the limitations of scalar systems.

We can also work with more complex systems. Consider a multilayer path with a common segment as in Figure 22. We want to associate patterns on two paths, **a-b-c-d** and **e-b-c-f** with link **b-c** in common.

First, even though parts of the path are physically common they can be functionally separated if the paths use different module patterns.

Second, because the paths can be functionally separated by their pattern selectivity, pattern information propagating backwards can now be used to sharpen and strengthen one path without interfering with the strengths of another path. The next section gives a very simple example.

7.9 Backwards Projections

There is no reason to assume that the backward projections and the forward projections must have the same strength. All that may be required initially for sparse systems is that modules produce activity along all points in the path, backwards and forwards. We can analyze a simple arrangement of active, sparsely connected modules, part of a sparse representation, using elementary outer-product Hebb learning.

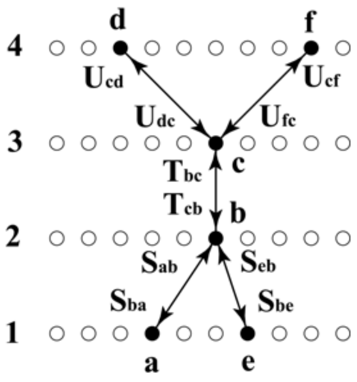

Fig 22. Two network of networks paths with a common segment.

First consider the two associative chains, up and down, **a>b>c>d** and **d>c>b>a**.

We assume supervised learning, that is, the desired patterns on **a** and **d** are present at input and output.

We will first consider outer-product learning in the weak signal linear range of operation of the system. As the path gets strong, we conjecture that modules in the path will develop internal states corresponding to the association activation patterns, that is, attractors.

Consider the two intermediate modules **b** and **c** on the path. Patterns on **b** and **c** are the result of information moving upwards from the input layer and downwards from the output layer, that is, initially with clamped patterns on **a** and **d**, with patterns **e** and **f** zero.

$$\text{Activity on } \mathbf{c} = \mathbf{T_{bc}b} + \mathbf{U_{dc}d} \tag{3}$$

$$\text{Activity on } \mathbf{b} = \mathbf{S_{ab}a} + \mathbf{T_{cb}c} \tag{4}$$

If the activities on **b, c** and **d** are non-zero patterns, we have the ability to use simple Hebb learning to change the strengths of coupling between modules using their connection matrices so as to change the associative path strength connecting the pattern on **a** with the pattern on **d**.

There are two active modules connected to **b**, up from **a** and down from **c.** The change in upward coupling between **a** and **b**, through $\mathbf{S_{ab}}$ is given by

$$\mathbf{\Delta(coupling\ S_{ab}) \sim ba^T} \tag{5}$$

The change in upward coupling between **b** and **c** is

$$\mathbf{\Delta(coupling\ T_{bc}) \sim cb^T} \qquad \textbf{(6)}$$

and between **c** and **d** is

$$\mathbf{\Delta(coupling\ U_{cd}) \sim dc^T}\ . \qquad \textbf{(6)}$$

We assume initial coupling is small, but sufficient to allow some patterns to emerge at **b** and **c**, that is, get through the path, weakly. After multiple learning events, the weak initial connections are small relative to later learning, and we can compute overall coupling between modules **a** and **d**.

If pattern **a** is presented at layer 1.and the pattern on **d** is zero, then the

$$\text{generated pattern on module } \mathbf{d} \sim \mathbf{(U_{cd})(T_{bc})(S_{ab})a} \qquad \textbf{(8)}$$

$$\sim (\mathbf{dc^T})(\mathbf{cb^T})(\mathbf{ba^T}\ )\mathbf{a} \qquad \textbf{(9)}$$

$$\sim \mathbf{d} \qquad \textbf{(10)}$$

The downward associations learn in the same way so that if the supervised pattern **d** occurs it will produce a constant times pattern **a**. Note the potential for a BAM-like associative loop. These results are well known from the properties of simple associators. The properties of the system are subject to the capacity limitations of simple associators and the size of the vectors and matrices being associated.

The other leg of the association (Figure 22), the pattern on **e** linked with the one on **f,** potentially can learn independently. It uses a different set of intermodule connections. One final state after extensive path learning might consist of a different set of activities on **b** and **c,** and, ultimately, different attractor states developing on **b** and **c** from learning on the other path.

Note that the somewhat arbitrary activities developing on **b** and **c** – the pattern sum of initial upward and downward connections – serves as a random 'hidden layer' link between input and output patterns. The argument holds for more intermediate layers and more associations, as long as sparse initial paths both up and down exist.

The system gets more complex as more patterns are learned. We know there are many regular topographic mappings in the cortex, particularly in the sensory systems. Our hunch is that some of the statistical abstraction and "concept forming" properties of associators may develop in the intermediate layers based on representational topography combined with learning in sparse systems. For example, it is possible, even likely, that sharing paths would be expected and computationally valuable in associations having a degree of sensory, perceptual or conceptual similarity. This issue remains to be explored.

7.10 Other Mechanisms

There are numerous other mechanisms that will be required to make this speculative, sparse system work in practice. For example, we have not considered here the important gain control mechanisms common in cortex that could limit the total number of active columns in a region. Besides generalized regional gain control, there is also some evidence that active columns corresponding to a complex object in monkey inferotemporal cortex can inhibit other columns [29].

In conclusion, with sparse data representations and sparse connectivities, Hebb learning has the potential to form selective paths that can be used for general associative learning in Network of Networks systems. Both upward and downward paths can be used together. Because the modules are non-linear, with limits on activities, stability of learning and dynamics is not likely to be a problem.

In both multilayer and single layer systems the potential exists for active feedback loops. Self-exciting loops may be useful intermediate level computational building blocks, as we suggest next.

8 Module Assemblies

The brain shows large differences in scale. Understanding how neurons work together in groupings of larger and larger size is perhaps the key to understanding brain function. We have already suggested one intermediate level of structure in the modules comprising the Network of Networks. We can take this idea a bit further where we speculate about the possibility of the formation of stable **module assemblies** as the next higher step in intermediate level representation.

In Section 6, we showed that the information integration model as discussed in the context of vowel filtering, produced a stimulus dependent series of nearby excited modules. (See Figure 23).

Suppose we bind together those nearby excited modules through associative linkages so they can become excited as a group. An interesting intermediate-level representation arises we call a **module assembly**.

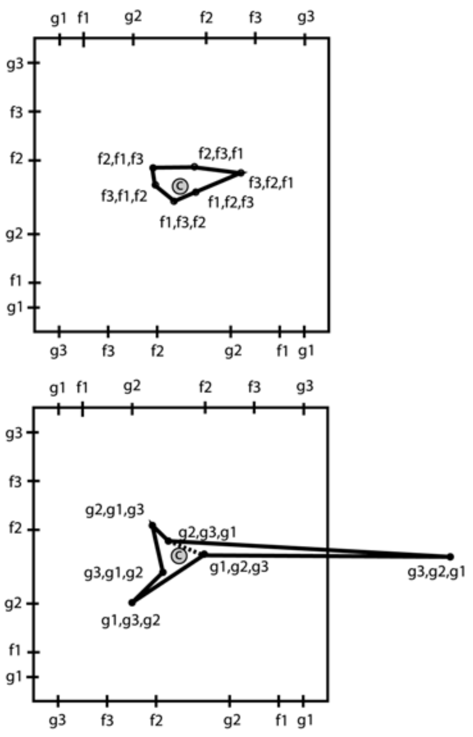

Fig. 23. Active modules for two different frequency patterns. (See Section 6.) These distinct frequency patterns have the potential to form distinct module assemblies.

8.1 Physiological Data from Cortex

We reviewed some of the properties of cortical columns in Section 2. It is hard to study detailed functional properties of columns. However, optical imaging of intrinsic cortical signals has now allowed visualization of structures of the size of cortical columns. The spatial resolution of this difficult technique can be a factor of 50 or more better than fMRI and gets into the size region where perhaps we can see some of the kinds of specialization for cortical computation that we need to see. A small body of intrinsic imaging work has been done on inferotemporal cortex [IT] in primates. As the highest visual area, IT contains the results of previous processing presumably in a form that is congenial for further use. Several groups have studied the organizational properties of infereotemporal cortex (area TE) from a computational point of view (See Tsunoda et al., [29]; Tanaka, [30,31]). They proposed a model for inferotemporal cortex function that is in rough harmony with the basic architecture we assume for the Ersatz Brain

Tanaka's early work on inferotemporal cortex used (1) computer simplified stimuli and (2) microelectrode recordings. Cells in area TE respond to few stimuli and have relatively little spontaneous activity. Once Tanaka found an image that drove a cell, the next step was to perform a series of abstractions of it until an optimal simplified image – a "critical feature" -- was found that adequately drove the cell that but any further simplifications of it did not. Cells in a small region of TE tended to have similar critical features. For example, a number of cells in a small region might respond to various "T-junctions." The T-junctions would be different from each other in detail, but seemed to be examples of the same 'T' structure. Regions of similar responses seemed to have roughly the size (300 µm) and character of the functional columns found in cortex. Nearby "columns" had quite different critical features. There was no sign of the continuous gradation of angle of best response found, for example, in orientation columns in V1.

From four to about a dozen columns were excited by presentation of complex objects and seemed to represent a complex percept. Such sets of observations led both Tanaka and Tsunoda et al. to propose a sparsely distributed, column based model of image recognition. From [29]:

"… objects are represented by combination of multiple columns in a sparsely distributed manner. The region activated by a single object image is only 3.3 ± 2.5% of the entire recording area (number of examined object images, 37). … These results are consistent with feature-based representation, in which an object is represented by the combined activation of columns each responding to a specific visual feature of the object." ([29], p. 835)

8.2 "Cell Assemblies" and the Formation of Module Assemblies

If two modules are reciprocally associatively linked, we have a situation similar to the Bidirectional Associative Memory. If there are multiple interacting modules, we have the potential to form other interesting and complex associatively linked structures through Hebb learning, what we will call **module assemblies,** in harmony with what is seen in IT.

The source for this idea is the **cell assembly**, first proposed by Donald Hebb in his 1949 book *Organization of Behavior* [32]. What has become known as the **Hebb Rule** for synaptic coupling modification was originally proposed specifically to allow for the formation of cell assemblies.

A cell assembly is a closed chain of mutually self-exciting neurons. Hebb viewed the assembly as the link between cognition and neuroscience. When an assembly was active, it corresponded to a cognitive entity, for example, a concept or a word. Although the idea is an appealing one, it is hard to make it work in practice because it is difficult to form stable cell assemblies. Two common pathological situations are (a) no activity in the network after a period due to inadequate gain around the loop and, (b) spread of activity over the entire network since neurons in a realistic model system will participate in multiple assemblies and activity spread will widely. It is possible to control this behavior to some degree by making strong assumptions about inhibition, but the resulting systems are not robust.

As we mentioned, a key assumption of the Network of Networks model is that the basic computing elements are interacting groups of neurons. Module activity is not a scalar but a pattern of activity, that is, a high dimensional vector. Connections between modules are in the form of interactions between patterns. There is an intrinsic degree of selectivity. Patterns are less likely to spread promiscuously.

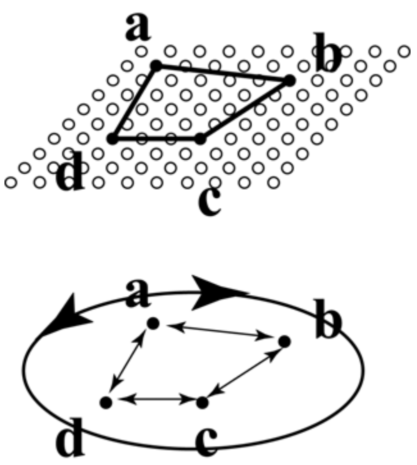

Fig. 24. Scheme for formation of a **module assembly**. Patterns flow around the loop based on local pairwise associations.

Because of this increased selectivity it might be possible that several nearby modules can become linked together to form loops through Hebb learning and can remain stable structures. We showed in Section 2 that associatively connecting modules together can increase the feedback coefficients in both modules (Figures 4, 5.).

We can make speculations about the local density of connections in neocortex. Data from Lund et al. [33] suggests substantial connectivity over a region of one or two millimeters. Recurrent collaterals of cortical pyramidal cells form relatively dense projections around a pyramidal cell. The extent of lateral spread of recurrent collaterals in cortex seems to be over a circle of roughly 3 mm diameter [34]. If we assume that a column is roughly a third of a mm, there are perhaps 10 columns per mm^2. A 3 mm diameter circle has an area of roughly 10 mm^2, suggesting that a column could project locally to perhaps 100 nearby columns.

Let us assume that the intralayer connections are sufficiently dense so that active modules a little distance apart can become associatively linked.

Consider the set of four active modules, **a,b,c,d**, in Figure 24. Assume they are densely connected locally. That will imply that **a** is connected to **b**, **c** to **d**, etc. Suppose that **a,b,c,d** are forced to hold a particular pattern by some outside mechanism, say another brain region, an outside supervising input, or perhaps frequent co-occurance. Then the analysis we just performed on sparse paths with simple associators suggests that associative links between modules will develop.

However, the path closes on itself. If the modules **a,b,c,d** are simultaneously active and are associatively linked, then if **a** is present, after traversing the linked path **a>b>c>d>a**, the pattern arriving at **a** will be a constant times the pattern on **a**. If the constant is large enough and the starting pattern is still present, there is the potential for feedback loop gain greater than one. The same analysis holds true for traversing the loop in the opposite direction, that is, **a>d>c>b>a.** Limitations on maximum activity in each module will stabilize activity as it becomes large so activity in the loop will not increase without bounds. All the modules in the loop are likely to reach attractors in synchrony after extensive learning.

Timing considerations become of critical importance. Adjusting travel time through the strength of associative connections is a potentially promising way to control loop dynamics. Note that critical timing relationships also appeared earlier when we discussed for formation of high order feature coincidences to form selective filters. The parameters that control module dynamics are also a potential source of control.

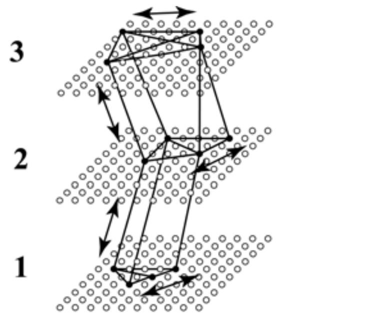

Fig. 25. Linkages between modules in a layer and from layer to layer may become bound into a more complex structure.

We should note that the capacity of these seemingly impoverished sparse systems can be very large. For example, suppose columnar sized regions are about one third mm in diameter. Then one mm^2 contains roughly 10 columns and a cm^2 contains roughly 1,000 columns. If something like 10 columns become active to specify an object in a one cm^2 piece of cortex, there are roughly $(1000)^{10}$ or 10^{30} possible different combinations of 10 active columns. If we assume in addition that each column has 10 attractors, we increase the number of potential combinations of states shown by those 10 active columns by 10^{10} for a total number of states of a one cm^2 chunk of cortex to as much as 10^{40} distinct states, a large enough number to satisfy almost any conceivable need, especially when we consider that there may be on the order of a million columns in a human cortex.

8.3 Multi-layer Assemblies

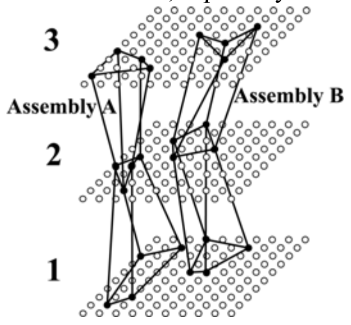

Fig. 26. Suppose we have two previously formed assemblies.

We can speculate on the extension of idea of module assemblies to multiple layers. Now we can perhaps see the outlines of a dynamic, adaptive computational architecture that becomes both feasible and interesting. The basic notion is that local module assemblies form, perhaps at each layer. Sparse connections between each layer learn to link the assemblies through associative learning. The individual sparse paths now can work together to form complex multilevel entities. We might end up with something like that shown in Figure 25, tightly connected module assemblies within a layer, sparsely linked together. This rings (loops in layers) and strings (sparse connections between layers) sparse architecture becomes a way to get multi-area activation patterns bound together through common dynamics.

Computation and learning from this point of view is based on the formation of sparsely interconnected between layers, less sparsely interconnected at a layer, module assemblies working with sparsely represented data.

8.4 Combinations

A natural way for learning to progress with this approach is to build combinations of module assemblies. Early learning, we conjecture, might form small module assemblies. Figure 26 shows two such module assemblies that occur together. They can become bound together by learning through co-occurrence (Figure 27).

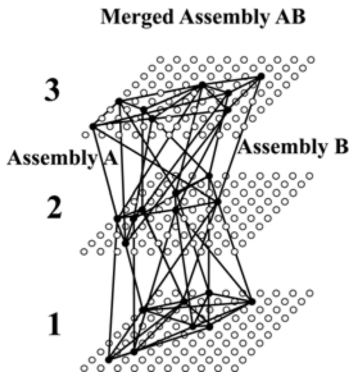

Fig. 27. Co-occurrence between assemblies can cause the structures to become associatively bound, forming a new more complex linked structure.

As learning progresses, groups of module assemblies will bind together through the linkage mechanisms we have discussed. The small initial assemblies perhaps can act as the "sub-symbolic" substrate of cognition and the larger assemblies, symbols and concepts. Because of sparseness smaller components might be largely independent of each other and not interfere with each other. Cognitive learning would then come to have something of the aspect of an erector set or Legos, where the parts start by being independent and then get bound together to form a sturdy higher-level spatially extended assembly. Note that there are more associative connections possible in the bound system than in the parts because there are many more possible paths. An interesting conjecture is that larger assemblies (words? concepts?) can be stable or more so than their component parts because of extensive cross-linkages.

This process looks somewhat like what is called compositionality (Geman [35]). The virtues of compositionality are well known. It is a powerful and flexible way to build information processing systems because complex mental and cognitive objects can be built from previously constructed, statistically well-designed pieces.

What we are suggesting in this discussion is a highly speculative possible model for the dynamics, intermediate level structure, and learning in a potentially compositional system. Note however, that this system is built based on constraints derived from connectivity, learning, and dynamics and not as a way to do optimal information processing.

Perhaps compositionality as we see it manifested in cognitive systems is more like a splendid bug fix than a carefully chosen computational strategy.

References

1. Sloman, S.: Causal Models: How People Think About the World and Its Alternatives. Oxford University Press, New York (2005)
2. Anderson, J.A., Sutton, J.P.: If We Compute Faster, Do We Understand Better? Behavior Research Methods, Instruments, and Computers. **29** (1997) 67-77
3. Anderson, J.A.: Arithmetic on a Parallel Computer: Perception Versus Logic. Brain and Mind. **4** (2003) 169-188
4. Strangman, G., Sutton, J.P.: The Behaving Human Neocortex as a Network of Networks. In: Hecht-Nielsen, R., McKenna, T. (eds.): Theories of the Cerebral Cortex. Springer, New York (2003) 205-219
5. Hopfield, J.: Neural Networks and Physical Systems with Emergent Collective Computational Abilities. Proceedings of the National Academy of Sciences. **79** (1982) 2554-2558
6. Anderson, J.A.: The BSB Network. In: Hassoun, M.H. (ed.): Associative Neural Networks. Oxford University Press, New York (1993) 77-103
7. Anderson, J.A.: An Introduction to Neural Networks. MIT Press, Cambridge, MA (1995)
8. Mountcastle, V.B.: Introduction. Cerebral Cortex. **13** (2003) 2-4
9. Hubel, D.H.,Wiesel, T.N.: Receptive Fields, Binocular Interactions, and Functional Architecture in the Cat's Visual Cortex. Journal of Physiology. **148** (1962) 106-154
10. Kosko, B.: Bidirectional Associative Memory. IEEE Transactions on Systems, Man, and Cybernetics. **18** (1988) 49-60
11. Pitts, W., McCulloch, W.S.: How We Know Universals: The Perception of Auditory and Visual Forms. In: McCulloch, W.S. (ed., 1965): Embodiments of Mind. MIT Press, Cambridge, MA (1947/1965) 46-66
12. Blum, H.J.: Biological Shape and Visual Science (Part I*).* Journal of Theoretical Biology. **38** (1973) 205-287
13. Kimia, B., Tannenbaum A., Zucker, S.W.: Shapes, Shocks and Deformations {I}: The Components of Shape and the Reaction-Diffusion Space. International Journal of Computer Vision. **15** (1995) 189-224
14. Bringuier, V., Chavane, F., Glaeser, L., Fregnac, Y.: Horizontal Propagation of Visual Activity in the Synaptic Integration Field of Area 17 Neurons. Science **283** (1999) 695-699
15. Lee, T.-S., Mumford, D., Romero, R., Lamme, V.A.F.: The Role of Primary Visual Cortex in Higher Level Vision. Vision Research. **38** (1998) 2429-2454
16. Lee, T.-S.: Analysis and Synthesis of Visual Images in the Brain. In: Olver, P., Tannenbaum, A. (eds): Image Analysis and the Brain. Springer, Berlin (2002) 87-106
17. Kovacs, I. ,Julesz, B.: Perceptual Sensitivity Maps Within Globally Defined Shapes**.** Nature. **370** (1994) 644-6
18. Kovacs, I., Feher, A., Julesz, B.: Medial Point Description of Shape: A Representation for Action Coding and its Psychophysical Correlates. Vision Research. **38** (1998) 2323-2333
19. Anderson, J.A.: Seven times seven is About Fifty. In: Scarborough, D., Sternberg, S. (eds.): Invitation to Cognitive Science, Volume 4. MIT Press, Cambridge, MA (1998) 255-299
20. Hadamard, J.: The Psychology of Invention in the Mathematical Field. Dover, New York (1949)
21. Murphy, G.L.: The Big Book of Concepts. MIT Press, Cambridge, MA (2002)
22. Watrous, R.L.: Current Status of Peterson-Barney Vowel Formant Data. Journal of the Acoustical Society of America. **89** (1991) 2459-2460
23. Peterson, G.E., Barney, H.L.: Control Methods Used in a Study of the Vowels. Journal of the Acoustical Society of America. **24** (1952) 175-184

24. Talavage, T.M., Sereno, M.I., Melcher, J.R., Ledden, P.J., Rosen, B.R., Dale, A.M.: Tonotopic Organization in Human Auditory Cortex Revealed by Progressions of Frequency Sensitivity. Journal of Neurophysiology. **91** (2004) 1292-1296
25. Olshausen, B.A., Field, D.J.: Sparse Coding of Sensor Inputs. Current Opinions in Neurobiology. **14** (2004) 481-487
26. Barlow, H.B.: Single Units and Sensation: A Neuron Doctrine for Perceptual Psychology? Perception **1** (1972) 371-394
27. Vincent, B.T., Baddeley, R.J., Troscianko, T., Gilchrist, I.D.: Is the Early Visual System Optimized to be Energy Efficient? Network: Computational neural systems. (In press)
28. Cooper, L.N., Intrator, N., Blais, B.S., Shoval, H.Z.: Theory of Cortical Plasticity. World Scientific Publishing, Singapore (2004)
29. Tsunoda, K., Yamane, Y., Nishizaki, M., Tanifuji, M.: Complex Objects are Represented in Macaque Inferotemporal Cortex by the Combination of Feature Columns. Nature Neuroscience. **4** (2003) 832-838
30. Tanaka, K.: Inferotemporal Cortex and Object Vision. In: Cowan, W.M., Shooter, E.M., Stevens, C.F., Thompson, R.F. (eds.): Annual Review of Neuroscience. **19** (1996) 109-139
31. Tanaka, K.: Columns for Complex Visual Object Features in Inferotemporal Cortex: Clustering of cells with similar but slightly different stimulus selectivities. Cerebral Cortex. **13** (2003) 90-99
32. Hebb, D.O.: The Organization of Behavior. Wiley, New York (1949)
33. Lund, J.S., Angelucci, A., Bressloff, P.C.: Anatomical Substrates for Functional Columns in Macaque Monkey Primary Visual Cortex. Cerebral Cortex. **13** (2003) 15-24
34. Szentagothai, J.: Specificity Versus (Quasi-) Randomness in Cortical Connectivity. In: Brazier, M.A.B., Petsche, H. (eds): Architectonics of the Cerebral Cortex. Raven, New York (1978) 77-97
35. Geman, S.: Invariance and Selectivity in the Ventral Visual Pathway. Division of Applied Mathematics, Brown University, Providence, RI. (in preparation)

Acknowledgements

This work was supported by a seed-funding grant from the Office of the Vice President for Research, Brown University and by an SBIR contract (FA8750-05-C-0122) from the Air Force Research Laboratory, Rome, NY to Aptima, Inc.