


Volume II



CONFERENCE RECORD

Sessions Presented at
Northcon/88
Seattle, WA
October 4 - 6, 1988

Sponsored by: Seattle and Portland Sections, Institute of Electrical and Electronic Engineers, IEEE 

Cascade Chapter, Electronic Representatives Association, ERA 

Electronics Manufacturers Association, EMA 

GEOMETRICAL INTERPRETATION OF HOPFIELD'S CONTENT ADDRESSABLE MEMORY NEURAL NETWORK

Robert J. Marks II
Les E. Atlas
Interactive Systems Design Lab
Department of Electrical Engineering
University of Washington
Seattle, Washington 98195

INTRODUCTION

Although the artificial neural network has a long rich history, modern (1980's) exuberance about neural networks can largely be credited to Hopfield and, in particular, to his artificial neural network (ANN) for content addressable memories (CAM's) [1]. The network has been implemented electronically [2] and optically [3-5].

Unfortunately, Hopfield's ANN CAM, in its original form, will live only in history. This is because, simply, the algorithm contains too many negative attributes:

1. For different asynchronous operations, the network will converge to different steady states for the same initialization.
2. The network does not scale well. Doubling the number of neurons less than doubles the network's storage capacity [8].
3. When operated synchronously, the network can break into steady state oscillation. [15]
4. Without the use of time-consuming processes such as simulated annealing, [18,19], the network can get stuck in false minima.
5. As we will show in this paper, the Hopfield neural network is algorithmically equivalent to an iterative matched filter which can be implemented in fewer operations per iteration.

In this paper, Hopfield's CAM ANN is pedagogically developed from a geometrical point of view using classic matched filter theory concepts. Its performance is contrasted with those of iterative matched filters and, more generally, correlation based associative memories [16].

PRELIMINARIES

Given N binary vectors of length L , a neural net is formed according to Hopfield's recipe. The i th neuron is assumed to be connected to the j th neuron with a transmittance of T_{ij} . If the sum of the inputs to a neuron exceeds a threshold, the neuron fires. If this sum is less than the threshold, the neuron remains off. The net is designed to stabilize at states corresponding to the original library vectors.

Mathematically, the interconnection net can be described by an $L \times L$ symmetric matrix T whose elements are T_{ij} . The T_{ij} elements are specified by the binary vectors or "library elements." Given a portion of one of the library element vectors, the entire vector can sometimes be regenerated by iteratively multiplying by the matrix and clipping the result to generate a binary vector approximation of the desired result. Furthermore, the algorithm is highly tolerant of faults. Elements in Hopfield's net can be destroyed [3] or grossly quantized [5], and the processor will still work "well."

HOPFIELD'S MODEL

→ Let $\{\vec{f}_n \mid 1 \leq n \leq N\}$ denote a set of library elements, each length L . The i th element of \vec{f}_n is f_{ni} . Each f_{ni} is either 1 or -1. We form the $L \times L$ matrix T with elements

$$T_{ij} = \begin{cases} \sum_{n=1}^N f_{ni} f_{nj}; & i, j = 1, 2, \dots, L; i \neq j \\ 0 & ; i = j \end{cases}$$

Hopfield found that setting the diagonal elements of the T matrix to zero assured convergence of the iterative algorithm to a stable state [1,6-7]. In a more recent paper, this constraint has been relaxed [6]. Later, we will utilize the same recipe for the diagonal elements and define an alternate matrix, T^* , with elements

$$T_{ij}^* = \sum_{n=1}^N f_{ni} f_{nj}; \quad i, j = 1, 2, \dots, L$$

Clearly, T can be found from T^* by setting the diagonal elements to zero. Thus,

$$T = T^* - N I \quad (1)$$

where I is the identity matrix. Note that $T_{ij} = T^*_{ij}$ for $i \neq j$.

If we define the $L \times N$ library matrix by

$$F = [\vec{f}_1 : \vec{f}_2 : \dots : \vec{f}_N]$$

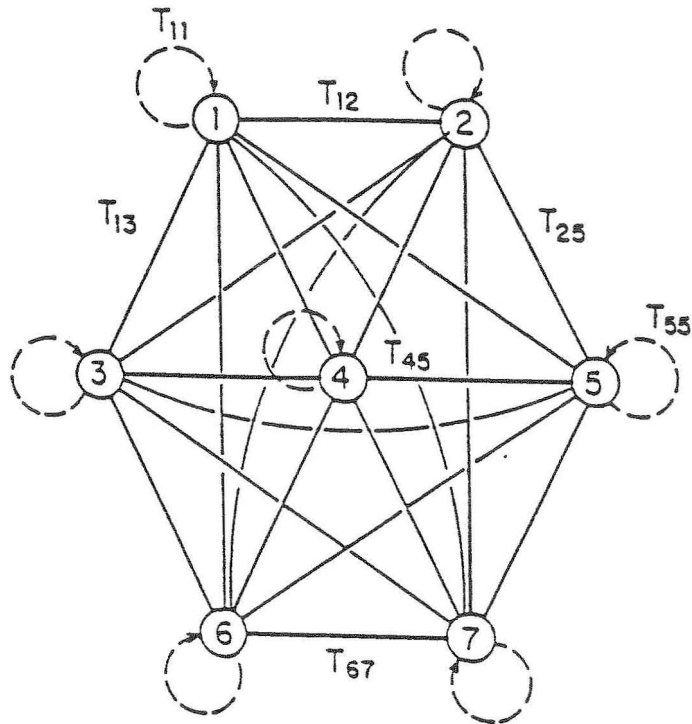


Fig. 1: A seven element neural net with symmetric interconnects. Autoconnects are shown with dashed lines.

we have

$$T^* = F F T \quad (2)$$

Hence, we refer to T^* as the outer product matrix.

The elements of both the T and the T^* matrices correspond to neural interconnects as illustrated in Fig. 1. Only when the sum of the inputs into a neuron exceeds a threshold, does the neuron fire (turns on). When a neuron fires, the resulting contribution to a second connected neuron might be sufficient to exceed its threshold and, as a result, the second neuron would fire, etc. Note that this model assumes symmetric interconnects ($T_{ij} = T_{ji}$). The zero diagonal in the T matrix corresponds to the use of no autointerconnects. (Autointerconnects are shown with dashed lines in Fig. 1.) With the autointerconnects, we have T^* . Without gives us T .

A firing neuron will be assigned a value of 1. Thus, if neuron i is on, then its contribution to neuron j is T_{ij} . If a neuron is off, it is assigned a -1. This model (bipolar) is in contrast to that of Hopfield where a non-firing neuron is equated to zero (binary). The justification for this alteration will later become clear. In either model, the neural firing threshold will be zero. (The effects of different thresholds for each neuron has been considered [8].)

Consider, then, a given initialization or input vector, \vec{g} , composed of ± 1 's. We form the iteration

$$\vec{g}_{M+1} = \text{sgn } T \vec{g}_M \quad (3)$$

where $\vec{g}_0 = \vec{g}$ and the vector operator, sgn , examines each element of $T\vec{g}_M$ and, if the element is positive, sets it to one. Otherwise, the element is set to -1. Ideally, (3) will converge to the library element \vec{f}_m that is closest to \vec{g} in the Hamming sense. Although this is commonly the case, some initializations result in oscillations or converge to elements not in the library [4, 15].

The iteration in (3) is implicitly synchronous and thus assumes every interconnect time delay is the same.

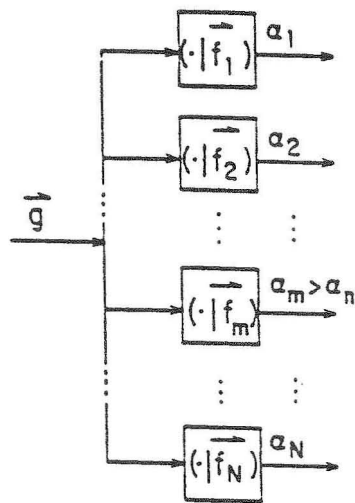


Fig. 2: A matched filter bank. If $\alpha_m > \alpha_n$ for all $m \neq n$, then \vec{f}_m is chosen as the correct library element.

MATCHED FILTER CAM'S

In this section, we consider CAM's based on the matched filter. The end result is an iterative matched filter based CAM that is identical to the Hopfield iteration in (3).

An Optimal Non Iterative CAM

Consider, again, a set of N bipolar library elements of length L and an arbitrary bipolar vector, \vec{g} . To find the library element, \vec{f}_m , that the closest to \vec{g} in the Hamming sense, we form the matched filter bank shown in Fig. 2. The output of the n th filter is the inner product.

$$\alpha_n = (\vec{g} | \vec{f}_n) = \vec{g}^T \vec{f}_n$$

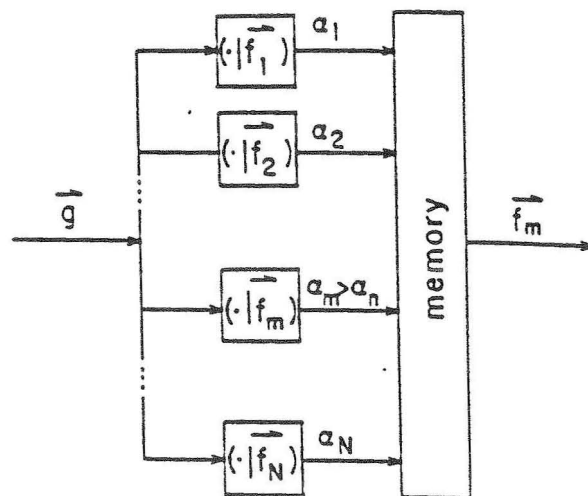


Fig. 3: Augmentation of the matched filter bank in Fig. 2 into noniterative optimal CAM. The maximum inner product, α_m , is used as a pointer to memory which outputs the corresponding library element: f_m .

One can easily show that

$$\alpha_n = L - 2h_n$$

where h_n is the Hamming distance between \vec{g} and \vec{f}_n . Thus, minimizing the Hamming distance is equivalent to maximizing the norm. Our task then is simply to search through the filter outputs for the maximum inner product, say, α_m . (If there is a tie for maximum, we can make no definitive decision.)

By adding a post processor, this matched filter can be made into a content addressable memory. The post processor consists of a linear memory which contains all of the library elements. The address, m , of the maximum matched filter output is the input to the memory, causing the corresponding library element, f_m , to be output (see Fig. 3). Assuming perfect processing, this CAM is clearly optimum in the Hamming sense. Classically, the matched filter is known for its optimality for signal detection in white Gaussian noise [9].

Another Non Iterative CAM

A second (sub-optimal) non iterative CAM based on the matched filter is shown in Fig. 4. Here, the matched filter outputs, $\{\alpha_n \mid 1 \leq n \leq N\}$, are used to weight their corresponding library elements. These weighted vectors are added to yield

$$\vec{r} = \sum_{n=1}^N \alpha_n \vec{f}_n \quad (4)$$

Each element in \vec{r} is clipped to give the bipolar output vector, \vec{f}^* . Clearly, we would like \vec{f}^* to be the library element, f_m , closest to \vec{g} in the Hamming sense.

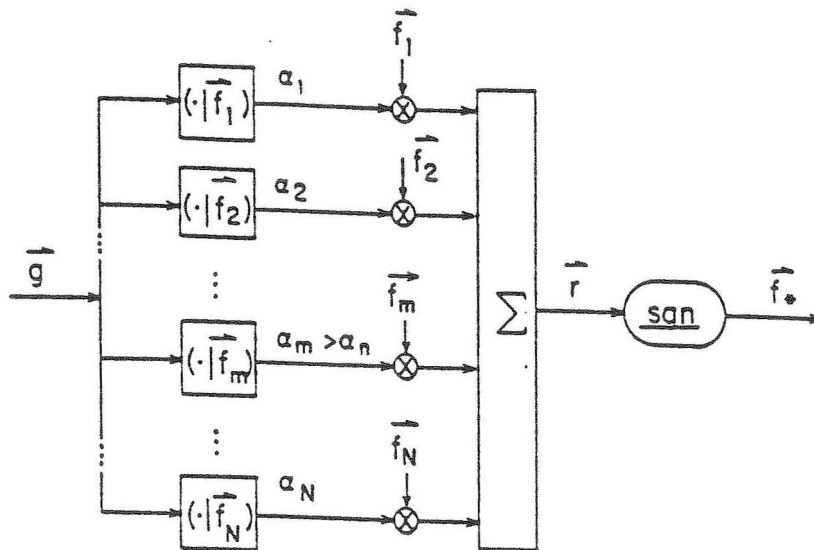


Fig. 4: A suboptimal non-iterative CAM. The output, \vec{f}^* , is the correct answer, \vec{f}_m , if $\alpha_m \gg \alpha_n$, $n \neq m$. Then $\alpha_m \vec{f}_m$ dominates in the sum of weighted library elements.

The rationale behind this CAM is as follows. If \vec{g} is sufficiently close to \vec{f}_m and sufficiently far from the other library elements, then $\alpha_m \gg \alpha_n$, ($n \neq m$), and the $\alpha_m \vec{f}_m$ term in 4) will dominate. Hence, clipping will result in $\vec{f}^* = \vec{f}_m$ which is our desired result.

The following is a specific example of how this processor will yield the correct answer: Let the library elements be orthogonal:

$$\vec{f}_p^T \vec{f}_q = L \delta_{p-q} \quad (5)$$

where δ_p is the Kronecker delta. If the ± 1 's in the library elements are chosen by a 50-50 coin flip and L is large, then (5) will hold to a good approximation.

Let \vec{f}_m denote a specific library element and let \vec{g} be a vector that is a Hamming distance k away from \vec{f}_m . The CAM in Fig. 4 will output $\vec{f}^* = \vec{f}_m$ if

$$k < \frac{L}{2N} \quad (6)$$

A proof is given in the Appendix. The authors have recently shown that if the α 's in Fig. 4 are run through a nonlinearity $N\alpha/2$ prior to weighting the library vectors, the output is always that library vector closest to the input in the Hamming sense [16].

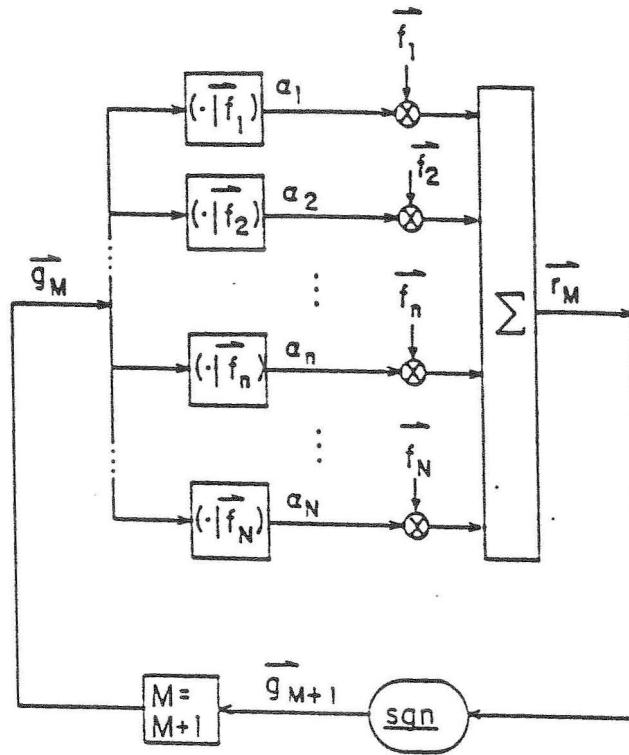


Fig. 5: An iterative matched filter (IMF) CAM. Its performance is equivalent to that of a Hopfield neural net with nonzero interconnects.

An Iterative Matched Filter

Even if \vec{f}^* in Fig. 4 is not the correct library element, it may be closer to \vec{f}_m than to \vec{g} . Thus, another iteration could be in order. Generalizing, we obtain the iterative matched filter (IMF) CAM in Fig. 5. Initializing with $\vec{g}_0 = \vec{g}$, the M th iterate, \vec{g}_M , is matched with each library element to form the vector of inner products:

$$\vec{\alpha}_M = F^T \vec{g}_M \quad (7)$$

As before, the library elements are weighted by these inner products to obtain the updated iterate:

$$\vec{g}_{M+1} = \text{sgn } F \vec{\alpha}_M \quad (8)$$

From the previous section, if the library elements are orthogonal, and \vec{g}_M is less than $L/2N$ away (in the Hamming sense) from a library element, then the iteration will converge in the next cycle.

	Hopfield	Iterative Matched Filter	Matched Filter with Table Lookup
Fault Tolerance	Excellent	None in Design	None in Design
Type of Algorithm	Iterative	Iterative	Noniterative
Operations per Iteration	L^2	$2NL$	NL^*
Final Result	Good	Same as Hopfield	Can be Optimum
Clocking	Synchronous or Asynchronous	Synchronous	Synchronous
Required Memory	$L^2/2$	$2NL$	$2NL$

*plus memory access

Table 1: Performance Comparison Among Three CAM's

outer product matrix. A total of NL elements in the library matrix, F , are mapped into a total of L^2 numbers in the T^* matrix. Recalling that we require $L \gg N$ reveals the outer product matrix as highly redundant. In contrast, the iterative matched filter and noniterative CAM's have no fault tolerance in their design, but could possibly be redundantly augmented.

In the absence of processing errors, the IMF and Hopfield CAM's are algorithmically identical. Thus, their convergence properties will be the same for identical inputs.

When comparing operations per iteration, one must take care. Every multiply in the three CAM's has a multiplicand of ± 1 . The multiplications thus are less important than the additions. We therefore chose a performance criteria of additions (or subtractions) per iteration in Table 1.

Lastly, we consider the required CAM memory. The Hopfield net requires memory on the order of $L^2/2$, equal to the number of interconnects. The IMF, on the other hand, requires storage only of the NL bits of the library matrix plus, depending on implementation, the same number of bits for the inner product stage.

SIGNAL SPACE INTERPRETATION

In the absence of processor errors, the Hopfield and IMF CAM's perform identically. In this section, we offer a signal space interpretation of the unified algorithm from an IMF viewpoint. Signal space interpretations can allow significant intuitive insight into algorithm performance.

Consider, again, Fig. 5. Our initial input vector $\vec{g}_0 = \vec{g}$, can be considered as a point in an L dimensional Hilbert space. In general, the vector

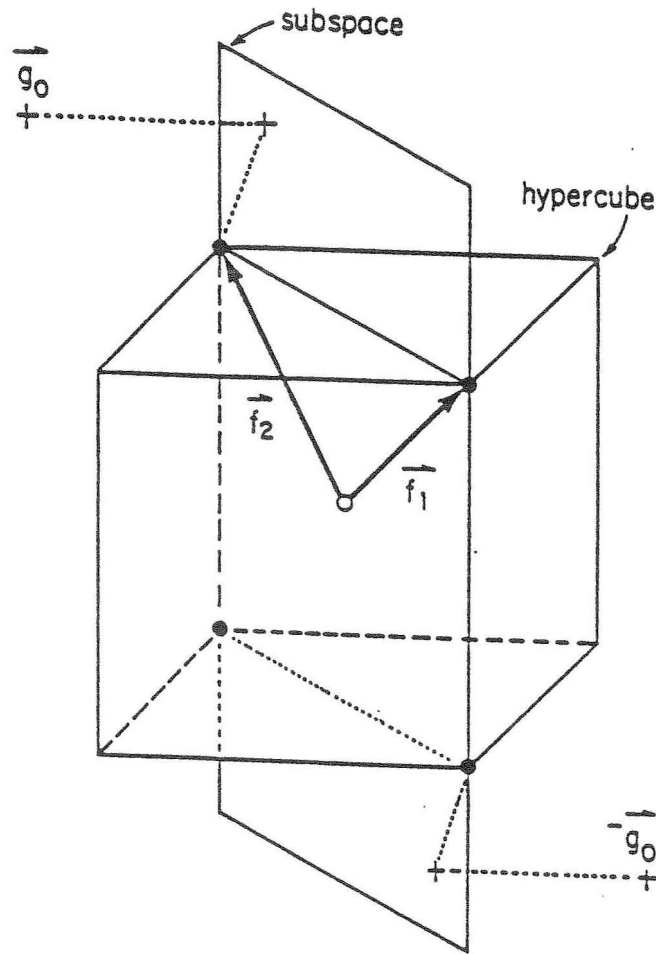


Fig. 7: Signal space interpretation of the IMF CAM. For orthogonal library elements, the iteration first corresponds to projection onto the subspace and then onto a hypercube vertex.

$$\vec{r}_M = \sum_{n=1}^N \alpha_n \vec{f}_n$$

must lie in the N dimensional subspace spanned by the N library elements. Indeed, if the library elements are orthogonal, then \vec{r}_M is the projection of \vec{g}_M onto that subspace.

To form the next iterate, we threshold each element of \vec{r}_M to obtain \vec{g}_{M+1} . Geometrically, an L vector of ± 1 's can be considered as a vertex of an L dimensional hypercube. Thresholding each element of \vec{r}_M results in the vertex, \vec{g}_{M+1} , that is closest to \vec{r}_M in the mean square sense. Thus, for orthogonal library elements, the algorithm, as it iterates, corresponds to alternately projecting onto a subspace and then to a vertex of the hypercube. This behavior suggests that the IMF CAM is algorithmically close to alternating convex set projections techniques for signal recovery and synthesis [10-13]. The problems differ only in the nonconvexity of the hypercube vertex set.

An illustration of the IMF CAM is in Fig. 7. Two library vectors, \vec{f}_1 and \vec{f}_2 , correspond

to two hypercube vertices. Both vectors have their tails at the origin which is at the cube's center. The closure of these two elements is the planar subspace shown. Beginning with some element, $\vec{g}_0 = \vec{g}$, we project onto the subspace as shown.* Next, we threshold and project to the nearest vertex as shown which, in this case, is the desired library element, \vec{f}_2 .

A fixed point is a vector that is unaltered by further iteration. Clearly, we desire that each of our library elements be fixed points. In general, any point that is both on a hypercube vertex and the subspace is a fixed point. Unfortunately, not only do our library elements lie on both sets, so do their negatives. Indeed, as illustrated in Fig. 7, if an initial vector \vec{g} converges to \vec{f}_m , then $-\vec{g}$ converges to $-\vec{f}_m$. The negative of a library element will normally not be a library element. One possible remedy is to have varying (nonzero) neural thresholds which has the effect of translating the subspace into a linear variety.

Note also, that vertices other than library elements and their negatives can lie on the subspace. For example, if

$$[\vec{f}_1 : \vec{f}_2 : \vec{f}_3] = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 1 \\ 1 & 1 & -1 \end{bmatrix}$$

then $\vec{f}_1 + \vec{f}_2 + \vec{f}_3 = [1 \ 1 \ 1 \ 1]^T$. Thus $\vec{f}_1 + \vec{f}_2 + \vec{f}_3$ is also a vertex which lies on the subspace and, for this library set, is an undesired fixed point.

Summary

Hopfield's neural net CAM has been shown to be equivalent to an iterative matched filter (IMF) CAM. The advantages of a Hopfield neural net are high fault tolerance and asynchronicity. The high fault tolerance, however, is bought with the price of more operations per iteration. Both procedures can be viewed in a signal space as alternate projections between the subspace spanned by the library elements and the hypercube's vertices.

As was mentioned, the projection argument holds exactly only for orthogonal elements. For $L=3$, no two vertices are orthogonal. Hence, our illustration in Fig. 7 is pedagogical but nonetheless instructive. For non-orthogonal library elements, the matrix $\hat{T} = F(F^T F)^{-1} F^T$ (in lieu of T^) provides an orthogonal projection onto the subspace spanned by the library elements. Unlike T and T^* , the corresponding neural net has, in general, noninteger interconnects. Note, however, that $\alpha \hat{T}$ ($\alpha > 0$) will perform the same as \hat{T} due to the sgn. For an appropriate α , $\alpha \hat{T}$ could contain all integers to a good approximation. Other projection based neural networks have been recently proposed [14,17].

APPENDIX A

Here, we prove (6). Clearly

$$\alpha_m = \vec{g}^T \vec{f}_m = L - 2k$$

and

$$|\alpha_n| \leq 2k; n \neq m$$

Thus

$$\vec{r} = (L - 2k) \vec{f}_m + \sum_{\substack{n=1 \\ n \neq m}}^N \alpha_n \vec{f}_n$$

or, in terms of components

$$r_j = (L - 2k) f_{mj} + \sum_{\substack{n=1 \\ n \neq m}}^N \alpha_n f_{nj}; 1 \leq j \leq L \quad (A1)$$

If $f_{mj} = -1$ clipping results in $r_j < 0$, then $f_{mj}^* = -1$. Similarly, for $f_{mj} = 1$, we desire $r_j > 0$ and $f_{mj}^* = 1$. Let's impose this latter case on (A1). From (A1), for $f_{mj} = 1$, using the worst case condition, we obtain

$$r_j \geq (L - 2k) - 2k(N - 1)$$

Thus, to insure that $r_j > 0$ for $f_{mj} = 1$, it is sufficient to require (5). By a similar treatment, one can demonstrate that this constraint assures that $r_j < 0$ when $f_{mj} = -1$.

APPENDIX B

Let $\{\vec{v}_q | 1 \leq q \leq 2^L\}$ denote the set of all distinct vectors of length \sqrt{L} with ± 1 elements. Each \vec{v}_q corresponds to a vertex of an L dimensional hypercube. For an arbitrary vector \vec{g} , we will show that $\text{sgn} \vec{g}$ is that vertex closest to \vec{g} in the mean square sense.

For each vertex, define the error:

$$\epsilon_q = \|\vec{g} - \vec{v}_q\|^2; 1 \leq q \leq 2^L$$

where $\|\vec{h}\|^2 = \vec{h}^T \vec{h}$. Clearly

$$\epsilon_q = \|\vec{g}\|^2 + \|\vec{v}_q\|^2 - 2\vec{g}^T \vec{v}_q$$

Since $\|v_q\|^2 = L$ for all q , minimizing ϵ_q is equivalent to maximizing the inner product $\vec{g} \cdot \vec{v}_q$. (The same principle on which the matched filter is founded.) We write

$$\vec{g} \cdot \vec{v}_q = \sum_{p=1}^L g_p v_{pq}$$

For a given g_p with every $v_{pq} = \pm 1$, this is clearly maximum when $v_{pq} = \text{sgn } g_p$ and our proof is complete.

ACKNOWLEDGEMENTS

The authors express their gratitude to Ms. Cindy Chernoff for her assistance in the preparation of the manuscript. They also gratefully acknowledge the support of the Boeing High Technology Center in Bellevue Washington for their support of their work in neural networks. Marks and Atlas' work in neural networks is supported by the Washington Technology Center and Physio-Control. Les Atlas is also supported in part by an NSF PYI award.

REFERENCES

1. J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," Proc. Natl. Acad. Sci., USA, 79 pp, 2554-2558 (1982).
2. C. Brown, "Hopfield's Neural Nets Realize Biocomputing" Electronic Engineering Times (April 7, 1986).
3. "Optoelectronics Builds Viable Neural-Net Memory" in Electronics (June 16, 1986).
4. D. Psaltis and N. Farhat "Optical Information Processing Based on an Associative-Memory Model of Neural Nets with Thresholding and Feedback," Optics Letters 10, pp 98-100 (1985).
5. N. Farhat, D. Psaltis, A. Prata and E. Paek, "Optical Implementation of the Hopfield Model," Applied Optics 24, pp 1469-1475 (1985).
6. D.W. Tank and J.J. Hopfield, "Simple Neural" Optimization Networks: An A/D converter, Signal Decision Circuit and a Linear Programming Circuit," IEEE Trans Circuits and Systems, CAS-33, pp 533-541 (1986).
7. J.J. Hopfield, "Neurons with Graded Response have Collective Computational Properties Like that of Two-State Neurons," Proc. Natl. Acad. Sci. 86, pp 3088-3092 (1984).
8. Y.S. Abu-Mostafa and J.M. St. Jacques, "Information Capacity of the Hopfield Model," IEEE Trans. on Information Theory, IT-31, pp 461-464 (1985).
9. H.L. Van Trees, Detection, Estimation and Modulation Theory, Part I, (Wiley, New York, 1968), pp 257-271.

10. J.R. Fienup, "Reconstruction and Synthesis Applications of an Iterative Algorithm," in Transformation in Optical Signal Processing, Rhodes, Fienup and Saleh, Editors, Proc. SPIE 373, pp 147-160 (1984).
11. D.C. Youla and H. Webb, "Image Restoration by the Method of Convex Projections: Part 1-Theory," IEEE Trans. Med. Imaging, Mi-1, pp 81-94 (1982).
12. M. Goldberg and R.J. Marks II, "Signal Synthesis in the Presence of an Inconsistent Set of Constraints," IEEE Trans. Circuits and Systems, CAS-32, pp 647-663 (1985).
13. D.C. Youla and V. Velasco, "Extensions of a Result on the Synthesis of Signals in the Presence of Inconsistent Constraints," IEEE Trans. Circuits and Systems, CAS-33, pp 465-468 (1986).
14. R.J. Marks II, "A class of continuous level associative memory neural nets," Applied Optics 26, pp 2005-2009 (1987).
15. K.F. Cheung, L.E. Atlas and R.J. Marks II, "Synchronous versus asynchronous behavior of Hopfield's content addressable memory," Applied Optics 26, pp 4808-4813 (1987).
16. R.J. Marks II, L.E. Atlas, J.J. Choi, S. Oh, K.F. Cheung and D.C. Park, "A performance analysis of associative memories with nonlinearities in the correlation domain," Applied Optics (in press).
17. R.J. Marks II, S. Oh, L.E. Atlas and J.A. Ritcey, "Alternating projection neural networks," ISDL Report 11587 (submitted for publication to IEEE Trans. CAS).
18. S. Kirkpatrick, C. Gelatt, Jr. and M. Vecchi, "Optimization by simulated annealing," Science Magazine, vol. 220, #4598, pp 671 (1983).
19. S. Geman and D. Geman, "Stochastic relaxation, Gibbs distribution, and the Bayesian Restoration of Images," IEEE Tr. Pattern Recognition and Machine Intelligence, vol. PAMI-6, pp 721 (1984).

