

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

Searching for the Orthogonal States of a Neural Network

Heng Wang

Department of Computer and Information Science
New Jersey Institute of Technology
Newark, NJ 07102

Thesis Advisor: Dr. David T. Wang

ABSTRACT

Two approaches to find orthogonal states of neural network are presented in the paper. The first approach is a recursive one, it builds N orthogonal vectors based on $N/2$ orthogonal vectors. The second approach is a formula approach, in which orthogonal vectors can be obtained using a formula. Using these approaches, orthogonal states of neural network are found. Some properties of the neural network built on these orthogonal vectors are presented in Appendix A and some examples are given in Appendix B.

SEARCHING FOR ORTHOGONAL STATES
OF NEURAL NETWORKS

by:

Heng Wang

A Thesis

Submitted to

Department of Computer and Information Science

New Jersey Institute of Technology

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Computer and Information Science

January, 1992

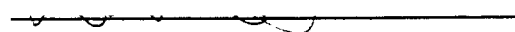
APPROVAL SHEET

Title of Thesis: Searching for Orthogonal States of Neural Networks

Name of Candidate: Heng Wang
Master of Science in Computer and Information Science
January, 1992

Thesis and Abstract

Approved by:


Dr. David T. Wang

Assistant Professor,

Department of Computer and Information Science.

Feb. 18, 1992

Date

VITA

Name : Heng Wang

Address : 172 6th Street Apt. 3. Harrison, NJ 07029

Degree and date to be conferred : MSCIS, Jan. 1992

Date of Birth :

Place of Birth :

Secondary Education : Shanghai Weiyu Middle School

Professional Institutions Attended	Dates	Degree	Date of Degree
Fudan University Shanghai, China.	1986- 1990	BSCS	July,1990
New Jersey Institute of Technology	1990- 1991	MSCIS	January,1992

ACKNOWLEDGEMENT

I am very grateful to my graduate advisor Dr. David T. Wang whose inspirations and guidance benefited me significantly; without his support this work could not have been finished.

I would also like to express my special thanks to my advisor Dr. Daniel Y. Chao who helped me a lot from the very beginning of this thesis. Dr. Chao provided me with some very helpful papers and was always willing to help me. Without his generous support, I cannot possibly complete the thesis.

And finally, a thank you to my friends and classmates at NJIT, including Sanjay K. Desai, Steve Lin, Richard Hong and Gery Lin for their encouragement and assistance.

Contents

1 Introduction	1
2 Methods to Find Orthogonal States of A N-neuron Network	4
2.1 Introduction	4
2.2 Recursive Procedure	5
2.3 Formula Approach	9
3 Conclusion	18
Bibliography	19
Appendix A: Some properties of the Associative Memory Built on Orthogonal Vectors	21
Appendix B: Examples	31

Chapter 1

Introduction

A neural network is a distributed information processing structure consisting of neurons interconnected by synaptic connections. The neurons can possess local memory and can carry out localized information processing operation. The function of neural networks is primarily determined by the connection topology between the neurons, the connection strengths and the type of processing formed at the computing elements. The complete system can conduct functions such as classification, optimization and associative memory.

Consider a interconnected network of n neurons, the state of the system is u , which is a set of binary states of each neuron. In the $M-P$ model, the new state of neuron i is

$$u_i = \text{sgn}(\sum_j w_{ij} u_j - L) = \begin{cases} 1, & \text{if } \sum_j w_{ij} u_j \geq L \\ 0, & \text{if } \sum_j w_{ij} u_j < L \end{cases}$$

where L is a threshold level, u is a n -tuple vector which stands for the state of the associative memory of the network, u_i is the state of neuron i and w_{ij} is the weight of the synaptic connection between neurons i and j .

In the *Hopfield-Little* model [9] of associative memory, the neurons have binary states with threshold value assumed to be zero. The weight matrix of the memory can

be made of the sum of the outer product of each desired vector. That is $W_{ij} = \sum s_i^{(k)} s_j^{(k)}$. The network is fully connected and symmetric, on which the *Hebb* rule applies. Once the weights are determined, the neural network can produce the desired output. When the input vector is corrupted, partially incorrect or incomplete, it still has the ability to converge the desired one. The neural states at these minima represent the memories of the system.

Therefore, it is crucial to determine what kind of prototypes to be selected and how to construct the associative memory.

The *Hebb's* law and the *projection* rule serve as the classical solutions to provide prototype vectors or fixed points of a neural network with synaptic matrix W . The former solution differs from the latter in that the prototype vectors (V 's) in the *Hebb's law* must be orthogonal to each other and the synaptic matrix can be constructed as the sum of outer products of these orthogonal vectors, i.e., the prototypes. The construction rule for the synaptic matrix is:

$$W = \frac{1}{n} (V_1 * V_1 + \dots + V_n * V_n) = \frac{1}{n} \sum_{k=1}^n V_k * V_k = \frac{1}{n} X X^T ,$$

where the superscript T stands for transpose, n is the total number of prototype vectors and '*' indicates the outer product. When they are not orthogonal to each other, the *Hebb's law* cannot guarantee the perfect retrieval of the prototypes.

It was mentioned in [10] that for orthogonal feature vectors, the capacity is 100% in comparison with the tradeoff between the memory capacity and the degree of fault-tolerance which has been estimated to be about 15% of B bits in *Hopfield-Little* Model [9].

In order to achieve good error recovery, it is desirable that these orthogonal prototype vectors are equally spaced apart in terms of the *Hamming Distance*. Further, this *Hamming Distance* should be as large as possible. For a N -neuron network, the largest *Hamming Distance* between any two orthogonal vectors is $N/2$, if they are equally spaced apart. A neural network constructed from such a set of prototype vectors should demonstrate a better retrieval capability which in turn results in a network with higher memory capacity. The *Hamming Distance* between any two of orthogonal vectors constructed by the proposed technique is $N/2$.

This paper presents two techniques to search for N orthogonal vectors of a N -neuron network. Some properties of the neural network built upon these orthogonal vectors are presented in the appendix. Due to the generality of these rules, some properties exhibited by low order neural networks can be extended to the networks with large number of neurons.

Chapter 2

Methods to Find Orthogonal States of A N-neuron Network

2.1 Introduction

There are various approaches to find out orthogonal vectors. One is to use those sinusoidal function and the other is to use those non-sinusoidal series. In the latter, there are some well-known functions: *Walsh* function, *Harr* function and *Ramemacher* function. Both *Walsh* function and *Harr* function can form a complete orthogonal set while *Ramemacher* function provided another set of two-level orthogonal functions which are incomplete but true subset of the *Walsh* function. In this paper, we present a unique recursive approach and a formula approach to find orthogonal vectors.

In the recursive approach, the $2N$ orthogonal vectors for a $2N$ -neuron network can be constructed based on the N orthogonal vectors of a N neural network with N neurons. The above N orthogonal vectors can be constructed in a similar fashion based on the $N/2$ orthogonal vectors of a neural network with $N/2$ neuron, ... and this alternative has the disadvantage of the necessity of constructing for all neural networks with $\frac{N}{2^k}$ neurons, $k = 0, 1, 2, \dots, \log N - 1$. The formula approach circumvents this drawback by calculating the N components of each of the N orthogonal vectors based on a formula. The resulting orthogonal vectors are identical to those constructed using

the recursive alternative.

2.2. Recursive Procedure

Consider a N-neuron network with $N = 2^k$ where k is a positive integer. The following lemma is obvious.

Lemma 1: The distance between any two orthogonal neuron vectors is $N/2$.

Proof: Let V_1 and V_2 be the two orthogonal vectors. Then $\sum_{j=1, \dots, N} v_{1j} v_{2j} = 0$, which indicates that half of the components in V_1 and V_2 have different signs. Hence the distance between them is $N/2$. \square

Let $V_1 = [1 \ 1 \ \dots \ 1]^T$. In order for V_2 to be orthogonal to V_1 , V_2 must have equal number ($\frac{N}{2}$) of 1's and -1's. Choose those V_2 's as

$$V_2 = [-1 \ -1 \ -1 \ \dots \ -1 \ 1 \ 1 \ 1 \ \dots \ 1]^T$$

or

$$V_2 = -[-1 \ -1 \ -1 \ \dots \ -1 \ 1 \ 1 \ 1 \ \dots \ 1]^T$$

It is easy to see that $V_1 V_2 = 0$. Notice that the two V_2 's are not orthogonal to each other and there are two blocks in V_2 : one with all 1's (denoted by **the + block**) and the other with all -1's (denoted by **the - block**).

To find V_3 , we divide each block of V_2 into one + *block* and one - *block*. Since there are two blocks in V_2 , there are four possible V_3 's,

$$V_3 = \left[1 \ 1 \ \dots \ 1 \ -1 \ -1 \ \dots \ -1 \ 1 \ 1 \ \dots \ 1 \ -1 \ -1 \ \dots \ -1 \right]^T = \left[+ \ - \ - \ - \right]^T,$$

or

$$V_3 = \left[1 \ 1 \ \dots \ 1 \ -1 \ -1 \ \dots \ -1 \ -1 \ -1 \ \dots \ -1 \ 1 \ 1 \ \dots \ 1 \right]^T = \left[+ \ - \ - \ + \right]^T,$$

or

$$V_3 = -\left[1 \ 1 \ \dots \ 1 \ -1 \ -1 \ \dots \ -1 \ 1 \ 1 \ \dots \ 1 \ -1 \ -1 \ \dots \ -1 \right]^T = \left[- \ + \ - \ + \right]^T,$$

or

$$V_3 = -\left[1 \ 1 \ \dots \ 1 \ -1 \ -1 \ \dots \ -1 \ -1 \ -1 \ \dots \ -1 \ 1 \ 1 \ \dots \ 1 \right]^T = \left[- \ + \ + \ - \right]^T$$

We call the above vectors with ‘+/-’ components +/- **equivalent vectors**. To evaluate the inner product between two +/- equivalent vectors, we substitute ‘1’ for ‘+’ and ‘-1’ for ‘-’. The following theorem states that one can test whether two vectors are orthogonal to each other using the +/- *equivalent vectors*.

Theorem 2.1: If V_i and V_j have identical number of blocks, each block has identical number of components, and $V_i V_j = 0$, then $V_i^e V_j^e = 0$, where V_i^e and V_j^e are the +/- equivalent vectors of V_i and V_j .

Proof: This theorem follows from that fact that the inner product between any two blocks has the same magnitude. \square

It is easy to see that all these four possible V_3 ’s are orthogonal to both V_1 and V_2 . Notice that these four V_3 ’s are not orthogonal to each other. Pick as many as possible so that they are orthogonal to each other. Theorem 2.1 helps because it is easier to test the orthogonality between +/- equivalent vectors whose dimensions are reduced by a factor of k. Thus, we search the four +/- vectors of V_3 for the largest set of orthogonal vectors. A further transformation is possible on these +/- vectors by

replacing each block of "+" with '1' and "-" with '-1'. The resulting vectors are: $\begin{bmatrix} 11 \end{bmatrix}^T$, $\begin{bmatrix} 1-1 \end{bmatrix}^T$, $\begin{bmatrix} -11 \end{bmatrix}^T$, $\begin{bmatrix} -1-1 \end{bmatrix}^T$, which is complete in the sense that it includes all vectors of dimension 2 whose components are 1 or -1. Thus there are two orthogonal vectors out of these four vectors.

Select $V_3^e = \begin{bmatrix} + & - & + & - \end{bmatrix}^T$ and $V_4^e = \begin{bmatrix} + & - & - & + \end{bmatrix}^T$ which are orthogonal to each other. Correspondingly $V_3 = \begin{bmatrix} 1 & 1 & .. & 1 & -1 & -1 & ... & -1 & 1 & 1 & .. & 1 & -1 & -1 & -1 \end{bmatrix}^T$ and $V_4 = \begin{bmatrix} 1 & 1 & ... & 1 & -1 & -1 & ... & -1 & -1 & -1 & ... & -1 & 1 & 1 & ... & 1 \end{bmatrix}^T$.

So far, we have generated four orthogonal vectors: one at each of the first two steps and two at the third step. In the next step, divide, as before, each of the four blocks of V_4 or V_3 into two subblocks which can be "+" or "-". Since each block has two possible divisions, there are 2^4 combinations. Although these vectors are orthogonal to the four vectors previously found, they are not orthogonal to each other. Again select as many orthogonal vectors as possible. These 2^4 vectors form a complete set of vectors of dimension 4. Hence there are 4 orthogonal vectors in this step. We can recursively perform this division to acquire new orthogonal vectors until the block consists of only one component. By induction, we have the following lemma.

Lemma: For all i , V_{i+1} has equal number of "+"s and "-"s,

Lemma: There are 2^n subblocks after the n -th step and there are $2^{2^{n-1}}$ possible vectors.

and we have $1 + \sum_{k=0}^{n-1} 2^k = 2^n$.

For each block, there are two possible divisions: $\begin{bmatrix} + \\ - \end{bmatrix}$ or $\begin{bmatrix} - \\ + \end{bmatrix}$. Denote $\begin{bmatrix} + \\ - \end{bmatrix}$ as 1 and $\begin{bmatrix} - \\ + \end{bmatrix}$ as -1. For n blocks, there are thus 2^n possible combinations of divisions.

For instance,

$$V_a = \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix} \rightarrow V'_a = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

and

$$V_b = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \rightarrow V'_b = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

It is easy to see that $V_a \cdot V_b = V_a \cdot V'_b = 0$. This is true for the general case as stated in the following lemma:

Lemma: Let V_a, V_b be two generated vectors and V'_a, V'_b the corresponding vectors by the substitution of $\begin{bmatrix} + \\ - \end{bmatrix} \rightarrow \begin{bmatrix} 1 \end{bmatrix}$, and $\begin{bmatrix} - \\ + \end{bmatrix} \rightarrow \begin{bmatrix} -1 \end{bmatrix}$, then $V_a \cdot V_b = 0$, iff $V'_a \cdot V'_b = 0$.

Theorem 2.2: Let V_i be any one orthogonal vector generated earlier, V_j a vector among all the $2^{2^{k-1}}$ at the k -th step, then $V_i \cdot V_j = 0$.

Proof: Let $V_i = [v_{i1} \ v_{i2} \ \dots \ v_{ik}]^T$ where v_{i1} is the first block where components are all "1"s or all "-1"s, v_{i2} is the second such component, and so on. Let $V_j = [v_{j1} \ v_{j2} \ \dots \ v_{jk}]^T$ where v_{j1} corresponds to v_{i1} , v_{j2} to v_{i2} , and so on.

By the nature of the procedure, there are equal number of "1"s and "-1"s in each v_{je} , $e=1, \dots, k$. Thus, for all $e \in [1, 2, \dots, k]$, $v_{ie}v_{je} = 0$, which in turn leads to $V_i V_j = 0$. \square

Theorem 2.3: The number of division steps to obtain N orthogonal vectors is $P = \log N$.

Note that after the k -th step, the total number of vectors generated from step 0 to step k is 2^k and the corresponding total number of orthogonal vectors is k . At the $(k+1)$ th step, there are 2^k blocks. With the substitutions of 1 for $\begin{bmatrix} + \\ - \end{bmatrix}$ and -1 for $\begin{bmatrix} - \\ + \end{bmatrix}$, these 2^k blocks forms a complete k -dimension vector space, denoted as S_1 . Note the total vectors generated from step 0 to k also forms a k -dimension space, denoted as S_2 . Thus, the k orthogonal vectors for S_1 can be found in a similar way to those in S_2 .

2.3 Formula Approach

Another recursive procedure is to construct the N orthogonal vectors based on the $N/2$ orthogonal vectors for a $N/2$ -neuron network. The first $N/2$ orthogonal vectors of a N -neuron network are constructed in the same manner to that for a $N/2$ -neuron network except that the N components of each of these $N/2$ orthogonal vectors can be

decomposed into $N/2$ blocks of $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ or $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$. With the substitutions of 1 for $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and -1 for $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$, these first $N/2$ orthogonal vectors become the $N/2$ orthogonal vectors for

a $N/2$ -neuron network. Applying the division procedure to $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$, they become $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ respectively. Consider $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ as 1 and $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ as -1, the last $N/2$ orthogonal vectors can be constructed in exactly the same manner as that for a $N/2$ -neuron network and it is obvious that the resulting vectors are orthogonal to each other.

Therefore, the above algorithm of constructing N orthogonal vectors based on $N/2$ orthogonal vectors is concluded as follows:

- 1) For the first $N/2$ vectors of N -neuron network, substitute $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ for 1 and $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ for -1 in the $N/2$ vectors of $N/2$ -neuron network.
- 2) For the second $N/2$ vectors of N -neuron network, substitute $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ for 1 and $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ for -1 in the $N/2$ vectors of $N/2$ -neuron network.

The following theorem presents a formula approach and the resulting vectors are proved to be orthogonal each other by using the above algorithm.

Theorem 2.4: Suppose $N = 2^k$, k is an integer constant, vectors V_i and component v_{ij} , where $i, j \in \{0, 1, \dots, N-1\}$. Suppose $i = \sum_{m=0}^{k-1} i_m * 2^m$, $i_m \in \{0, 1\}$, that is binary representation of i in binary digits i_m . Then the j th component of vector V_i is

$$v_{ij} = \begin{cases} +1, & \text{if } \sum_{m=0}^{k-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) \text{ is even} \\ -1, & \text{if } \sum_{m=0}^{k-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) \text{ is odd} \end{cases} \quad (2.1)$$

In expression

$$\sum_{m=0}^{k-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) \quad (2.2)$$

the notation $\lfloor x \rfloor$ stands for the integer part of the number x .

Proof :

It can be proved by induction. — It is easy to see that the theorem holds for $N = 2$.

When $k = 1, N = 2 = 2^1$, we have $i, j \in \{0,1\}$.

When $i = 0, i_0 = i_1 = 0$, from (2.1)

we have $v_{00} = 1, v_{01} = 1$.

When $i = 1, i_0 = 1$ and $i_1 = 0$, from (2.1)

we have $v_{10} = 1, v_{11} = -1$.

Therefore, $V_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $V_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and they are orthogonal each other.

Assuming the theorem holds for $N = 2^p$, that is, from the formula (2.1) and

$\sum_{m=0}^{p-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right)$, we can obtain those N orthogonal vectors, which are the

same as those generated from the above algorithm.

Now we need to prove that the theorem also holds for $N' = 2^{(p+1)} = 2N$ and generates $2N$ orthogonal vectors which have $2N$ components each. Let

$$X = \sum_{m=0}^{p-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) \quad (2.3)$$

Let's consider $\sum_{m=0}^p i_m \left(\lfloor j' \frac{2^{m+1}}{N'} \rfloor \bmod 2 \right)$, $j' \in \{0, 1, \dots, 2N-1\}$, the dimension

of the vectors is expanded by 2, also the total number of orthogonal vectors generated will doubled. We denote the first half of the $2^{(p+1)}$ orthogonal vectors S_1 and the second half S_2 . The most significant bit of the binary representation of S_1 is 0. Since in (2.2), the m is ranging from 0 to p instead of ranging from 0 to $p-1$. The binary representation of i has one more bit i_p and for those $i < N$, its i_p is 0 while for those $N \leq i < 2N$, i_p is 1.

In S_1 , the set of the first half, $i_p = 0$. So we have

$$\sum_{m=0}^p i_m \left(\lfloor j' \frac{2^{m+1}}{N'} \rfloor \bmod 2 \right) = \sum_{m=0}^{p-1} i_m \left(\lfloor j' \frac{2^{m+1}}{2N} \rfloor \bmod 2 \right) \quad (2.4)$$

where $0 \leq j' < 2^{p+1}$. And for all components of V_i' , we can use $2j$ and $2j+1$ ($j \in \{0, 1, \dots, N-1\}$) instead of j' in the space of $j' \in \{0, 1, \dots, 2N-1\}$. The equation of (2.4) can be separated into the following (2.4.1) and (2.4.2).

$$\sum_{m=0}^{p-1} i_m \left(\lfloor 2j \frac{2^{m+1}}{2N} \rfloor \bmod 2 \right) = \sum_{m=0}^{p-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) = X \quad (2.4.1)$$

and

$$\sum_{m=0}^{p-1} i_m \left(\lfloor (2j+1) \frac{2^{m+1}}{2N} \rfloor \bmod 2 \right) = \sum_{m=0}^{p-1} i_m \left(\lfloor (j + \frac{1}{2}) \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) \quad (2.4.2)$$

Since $0 \leq m < p$, we have $2 \leq 2^{m+1} \leq 2^p$, dividing all the the three expression in this inequality, we obtain

$$\frac{1}{2^{p-1}} \leq \frac{2^{m+1}}{2^p} \leq 1.$$

and the value of $\frac{2^{m+1}}{2^p}$ can be $\{1, \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{2^{p-1}}\}$. Therefore the value of

$$\frac{1}{2} \frac{2^{m+1}}{2^p} \text{ can be } \left\{ \frac{1}{2}, \frac{1}{4}, \dots, \frac{1}{2^p} \right\}.$$

Then we have $\lfloor (j + \frac{1}{2}) \frac{2^{m+1}}{2^p} \rfloor = \lfloor j \frac{2^{m+1}}{2^p} + \frac{1}{2} \frac{2^{m+1}}{2^p} \rfloor = \lfloor j \frac{2^{m+1}}{2^p} \rfloor$, and j is integer.

$$\lfloor (j + \frac{1}{2}) \frac{2^{m+1}}{2^p} \rfloor \bmod 2 = \lfloor j \frac{2^{m+1}}{2^p} \rfloor \bmod 2.$$

From (2.4.2), we have

$$\sum_{m=0}^{p-1} i_m \left(\lfloor (2j+1) \frac{2^{m+1}}{2N} \rfloor \bmod 2 \right) = \sum_{m=0}^{p-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right) \quad (2.4.3)$$

Therefore, in the first half of the vectors, when $j' = 2j$ or $j' = 2j+1$, from equation (2.4.1) and (2.4.3),

$$\sum_{m=0}^p i_m \left(\lfloor j' \frac{2^{m+1}}{N'} \rfloor \bmod 2 \right) = \sum_{m=0}^{p-1} i_m \left(\lfloor j \frac{2^{m+1}}{N} \rfloor \bmod 2 \right)$$

From (2.1), we have $V_{i(2j)'} = V_{i(2j+1)'} = V_{ij}$, that is the substitutions of $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ for 1 and

$\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ for -1 in the assumed N orthogonal vectors.

In S_2 , the set of the second half, $i_p = 1$. One more item is added to S_1 .

$$\begin{aligned} & \sum_{m=0}^p i_m \left(\lfloor j' \frac{2^{m+1}}{N'} \rfloor \bmod 2 \right) \\ &= \sum_{m=0}^{p-1} i_m \left(\lfloor j' \frac{2^{m+1}}{2N} \rfloor \bmod 2 \right) + \lfloor j' \frac{2^{p+1}}{2^{p+1}} \rfloor \bmod 2 = X + j' \bmod 2. \end{aligned}$$

Again this can be separated into the following (2.5.1) and (2.5.2).

$$\begin{aligned} & \sum_{m=0}^{p-1} i_m \left(\left\lfloor 2j \frac{2^{m+1}}{2N} \right\rfloor \bmod 2 \right) + (2j) \bmod 2 \\ &= \sum_{m=0}^{p-1} i_m \left(\left\lfloor j \frac{2^{m+1}}{N} \right\rfloor \bmod 2 \right) = X \end{aligned} \quad (2.5.1)$$

and

$$\begin{aligned} & \sum_{m=0}^{p-1} i_m \left(\left\lfloor (2j+1) \frac{2^{m+1}}{2N} \right\rfloor \bmod 2 \right) + (2j+1) \bmod 2 \\ &= \sum_{m=0}^{p-1} i_m \left(\left\lfloor j \frac{2^{m+1}}{N} \right\rfloor \bmod 2 \right) + 1 = X + 1 \end{aligned} \quad (2.5.2)$$

In the last situation (2.5.2), those *odd* in formula (2.1) will change into *even* and *even* will change into *odd*. $V_{i(2j)'} = V_{ij}$ while $V_{i(2j+1)'} = -V_{ij}$, that is the substitutions of $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ for 1 and $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$ for -1 in the assumed N orthogonal vectors.

We can see that the way we proof the result above is exactly the procedure of producing the orthogonal vectors in the algorithm described at the beginning the section. $2N$ orthogonal vectors can be constructed based on N orthogonal vectors with the dimension expanded by 2. \square

From this procedure, we can see that the set of resulting $2N$ orthogonal vectors is constructed from the N orthogonal vectors and the number of components is doubled.

For each V_1 in S_1 there is a V_2 in S_2 such that $i(s_2) = i(s_1) + 2^p$ and $V_2 = T[V_1]$,

where T stands for the substitutions of $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ by $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ by $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$. And V_1 is

constructed by substitution of $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ for 1 and by substitution of $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ for -1 in the N

components' vector. Note that V_1 and V_2 differ in the even components.

The following helps to explain the above theorem. From the following table, we can understand the formula constructing rule.

For example:

Index	Orthogonal Vector	Binary Vector
0 0	1 1 1 1	0 0 0 0
0 1	1 1 -1 -1	0 0 1 1
1 0	1 -1 1 -1	0 1 0 1
1 1	1 -1 -1 1	0 1 1 0

Denote *Base Vectors* as those vectors (orthogonal or index or binary) whose *Index Vector* has one and only one component of 1. With the substitutions of -1 by 1 and 1 by 0, an *Orthogonal Vector* in the second column becomes a *Binary Vector* in the third column, linear combinations of base vectors in the above table generate non-base vectors.

For instance, the *Index Vector* 0011 is composed of *Base Vectors* 0001 and 0010. That is, $0011 = 0001 + 0010$. And the corresponding *Binary Vectors* of *Index Vectors* 0001 and 0010 are 0011 and 0101. And addition rules are:

$$1 + 1 = 0, \quad 1 + 0 = 0 + 1 = 1, \quad \text{and} \quad 0 + 0 = 0.$$

Note that there are no carries in the above additions.

Add the two *Binary Vectors*, we have $0011 + 0101 = 0110$ whose *Orthogonal Vector* is $[1 -1 -1 1]$.

For the orthogonal vectors obtained from those two approaches, we have the following theorem.

Theorem 2.5: Let V_1, V_2, \dots, V_N be the generated N orthogonal vectors, and v_{ij} the j -th component of v_i ,

(1) For all $i, v_{i1} = 1$, and

(2) $\sum_{i=1}^N v_{ij} = 0, j \neq 1$

Proof : It is easy to see (1).

(2) It can be proved by induction.

Let $N=2$. The two orthogonal vectors are $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and (2) is satisfied.

Assume (2) holds for $N=2^{m-1}$. The theorem can be proved by showing that (2) also holds for $N=2^m$, there are m division steps.

After the $(m-1)$ -th division, there are 2^{m-1} orthogonal vectors; the rest 2^{m-1} orthogonal vectors are to be generated in the m -th division. Each of the first 2^{m-1}

orthogonal vectors has $(m-1)$ subblocks; each subblock is of the form $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ or

$\begin{bmatrix} -1 \\ -1 \end{bmatrix}$. By the assumption, it is easy to see that $\sum_{i=1}^{2^{m-1}} v_{ij} = 0, j \neq 1$. It remains to be

seen that $\sum_{i=2^{m-1}+1}^{2^m} v_{ij} = 0, j \neq 1$. At the m -th division $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $\begin{bmatrix} -1 \\ -1 \end{bmatrix}$ both become

$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$ or $\begin{bmatrix} -1 \\ 1 \end{bmatrix}$. By making the same substitution as mentioned earlier

$\begin{bmatrix} 1 \\ -1 \end{bmatrix} \rightarrow 1$ and $\begin{bmatrix} -1 \\ 1 \end{bmatrix} \rightarrow -1$, the number of components of each vector generated in the m -th division reduces from 2^m to 2^{m-1} . Let the corresponding vectors

be $V'_{2^{m-1}+1}, \dots, V'_{2^m}$. By the assumption that (2) holds for $N=2^{m-1}$, $\sum_{i=2^{m-1}+1}^{2^m} v'_{ij} = 0$.

This implies that $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} -1 \\ 1 \end{pmatrix}$ appear equally often for each subblock $j \neq 1$

among all V 's. Thus $\sum_{i=2^{m-1}+1}^{2^m} v_{ij} = 0, j=2, 3, \dots, 2^m$. \square

Property: Consider a matrix $M, M = (V_1 V_2 \cdots V_N)$, V_i is the orthogonal vectors constructed using the above approaches, then M is a symmetric matrix. Notice that the order of the vectors store in the matrix M .

Let $N = 8$, we have a symmetric matrix

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix}$$

The columns of the matrix are those orthogonal vectors and $V_1 = 1$, and

$$\sum_{i=1}^8 v_{ij} = 0, j \neq 1.$$

Chapter 3

Conclusion

Two approaches to find out orthogonal vectors were presented. The formula approach is an efficient one in compare with those recursive approaches. A neural network constructed using those orthogonal vectors will demonstrate some special abilities and properties such as non-increasing energy and performance of convergence. And some properties of the neural network are presented in Appendix.

BIBLIOGRAPHY

- [1] Kamp, Yves and Hasler, Martin. "Recursive Neural Networks for Associative Memory", John Wiley & Sons.
- [2] Abu-Mostafa, Y.S. and St. Jacques, J-M, "Information Capacity of the Hopfield Model", IEEE Tran Infa Th., Vol. 31, No. 4, July 1985, pp. 461-464.
- [3] Baldi. P. Neural networks, orientations of the hypercube and algebraic threshold functions. IEEE Transactions on In formation Theory, 34(3), 1988, pp. 523-530.
- [4] Chao, D. Y. and Wang, D. T., "Proof of some Higher order Neural Network Properties", to appear in Proc. Artificial Neural Networks in Engng, St. Louis, Missouri, Nov. 10-12, 1991.
- [5] Philip D. Wasserman, "Neural computing: Theory & Practice" (1989) Van Nos-trand Reinhold.
- [6] Psaltis D. & Venkatesh S. S., "Information Storage in Fully Connected Net-works". In: Lee, Y. C., " Evolution, Learning and Cognition " (1988). World Scientific Publishing Co. Pte. Ltd.
- [7] Hopfield, J. J. , "Neural Networks and Physical Systems with Emergent Collec-tive Computational Abilities", Proc. Nat'l Academy Sci., USA, Vol. 79, 1982, pp. 2554-2558.
- [8] Bruck, J., "On the Convergence Properties of the Hopfield Model", Proc. IEEE, Vol. 78, No. 10, Oct. 1990, pp. 1579

- [9] Hopfield, J. J. "Neurons with graded response have Collective Computational Properties like those of two-state Neurons." Proc. Nat'l Academy of Sci. 81, 1984, pp, 3088-3092.
- [10] Szu, H. H. "What is the significance of Neural Networks for AI". In: Lee, Y.C., "Evolution, Learning and Cognition" (1988). World Scientific Publishing Co. Pte. Ltd. pp. 373-381.
- [11] Beauchamp, K. & Yuen, C., "Digital Method for Signal Analysis". George Allen & Unwin Ltd. 1979 pp. 85-129
- [12] A. D. Bruce, A. Canning, B. Forrest, E. Gardner and D. Wallace, "Learning and Memory Properties in fully Connected Network".

APPENDIX A:

Some Properties of the Associative Memory built on Orthogonal Vectors.

Base on these orthogonal vectors, the synaptic matrices for autoassociative and heteroassociative neural networks can be constructed.

Considering the autoassociative memory, let the synaptic matrix W be the sum of outer product of V_i as follow,

$$W = V_1 * V_1 + V_2 * V_2 + \dots + V_k * V_k = \sum_{i=1}^k V_i * V_i,$$

where $0 < k \leq N$ and $N = 2^p$, p is a integer integer. And we call this method as "*C-Method*". If a normalization is required for the W , then it is

$$W = \frac{1}{k} \sum_{i=1}^k V_i * V_i = \frac{1}{k} X \cdot X^T$$

The memory is made of outer-product of prescribed state vector, with some ability for recall and error correction and we will see that the neural network constructed using this method has the ability of fault tolerance. For the synaptic matrix W , we have the following theorem and lemma.

Theorem a.1: Let V_1, V_2, \dots, V_N be the constructed orthogonal vectors, then

$$W = \frac{1}{N}(V_1 * V_1 + V_2 * V_2 + \dots + V_N * V_N) = I,$$

where I is an $N \times N$ identity matrix and $N = 2^p$, p is a integer constant.

Proof : Let w_{ij} be a element of the matrix W ,

$$w_{ij} = \frac{1}{N} \sum_{k=1}^N v_{ki} \cdot v_{kj},$$

When $i = j$,

$$w_{ii} = \frac{1}{N} \sum_{k=1}^N v_{ki} \cdot v_{ki} = \frac{N}{N} = 1.$$

for all $i \neq j$, w_{ij} can be proven by induction to be zero.

It also can be proved from the *property* in section II. Since the matrix M whose elements are made of orthogonal vectors is symmetric, the expression

$$w_{ij} = \frac{1}{N} \sum_{k=1}^N v_{ki} \cdot v_{kj} w_{ij} = \frac{1}{N} \sum_{k=1}^N v_{ik} \cdot v_{jk}$$

that is,

$$w_{ij} = \frac{1}{N} (V_i \cdot V_j) = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}$$

□

Lemma: Let W be a synaptic matrix generated from the sum of outer products of a set of $(N-1)$ orthogonal neuron vectors,

$$W = \sum_{N-1 \text{ vectors}} V_i * V_i$$

then we have $w_{ij} = 1$ or $-1, \forall i, j, i \neq j$.

Proof: The lemma is a extension of the theorem above.

Suppose the vector not picked up is V_m , then $V_m * V_m$ is a $N \times N$ matrix having

the elements -1 or $+1$. $w_{ij} = \frac{1}{N} \sum_{k=1}^N v_{ki} v_{kj}$

When $i \neq j$, $Nw_{ij} = \sum_{k=1, k \neq m}^N v_{ki} v_{kj} + v_{mi} v_{mj} = 0$, $\sum_{k=1, k \neq m}^N v_{ki} v_{kj} = -v_{mi} v_{mj}$

$$w_{lj} = -v_{mj}v_{mj} = \pm 1$$

□

Theorem a.2: Let the initial state be V'_1 which differs from V_1 only in the j -th component, i.e., $v'_{1j} = -v_{1j}$.

$$(1) \quad V_1 \cdot V'_1 = N - 2,$$

$$(2) \quad V_k \cdot V'_1 = -2v_{kj}v_{1j}, \text{ and}$$

$$(3) \quad (W \cdot V'_1)_k = N v_{1k} - 2w_{jk} v_{1j}, \quad k=1,2,\dots,N, \quad W = \sum_{i=1}^p V_i * V_i.$$

$$(4) \quad \text{In order to converge to } V_1, P < \frac{N}{2}.$$

Proof :

$$(1) \quad V_1 \cdot V_1 = \sum_{i=1}^N v_{1i}v_{1i} = \sum_{i=1, i \neq j}^N v_{1i}v_{1i} + v_{1j}v_{1j} = N$$

$$V_1 \cdot V'_1 = \sum_{i=1, i \neq j}^N v_{1i}v_{1i} + v_{1j}(-v_{1j})$$

$$V_1 \cdot V_1 - V_1 \cdot V'_1 = 2v_{1j}v_{1j} = 2.$$

Hence,

$$V_1 \cdot V'_1 = N - 2.$$

$$(2) \quad V_k \cdot V_1 = 0 = \sum_{i=1}^N v_{ki}v_{1i} = \sum_{i=1, i \neq j}^N v_{ki}v_{1i} + v_{kj}v_{1j}$$

$$V_k \cdot V'_1 = \sum_{i=1, i \neq j}^N v_{ki}v_{1i} + v_{kj}(-v_{1j})$$

$$V_k \cdot V'_1 - V_k \cdot V_1 = -2v_{kj}v_{1j}.$$

Hence,

$$V_k \cdot V'_1 = -2v_{kj}v_{1j}.$$

$$(3) \quad (W \cdot V'_1) = \left(\sum_{i=1}^p V_i * V_i \right) \cdot V'_1$$

$$(W \cdot V'_1) = V_1 \cdot (V_1 \cdot V'_1) + \sum_{i=2}^P V_i \cdot (V_i \cdot V'_1)$$

$$(W \cdot V'_1) = (N-2)V_1 + \sum_{i=2}^P (-2v_{ij}v_{1j})V_i$$

$$(W \cdot V'_1)_k = (N-2)v_{1k} + \sum_{i=2}^P (-2v_{ij}v_{1j})v_{ik}$$

$$(W \cdot V'_1)_k = N v_{1k} - 2 v_{1k} v_{1j} v_{1j} + \sum_{i=2}^P (-2v_{ij}v_{ik})v_{1j}$$

$$(W \cdot V'_1)_k = N v_{1k} + \sum_{i=1}^P (-2v_{ij}v_{ik})v_{1j} = N v_{1k} - 2 w_{jk} v_{1j}$$

$$(4) (W \cdot V'_1)_j = (N-2)v_{1j} + \sum_{i=2}^P (-2v_{ij}v_{1j})v_{ij}$$

$$(W \cdot V'_1)_j = (N-2)v_{1j} + \sum_{i=2}^P (-2)v_{1j}$$

$$= [N - 2P] v_{1j}$$

In order that $\text{sgn}(W \cdot V'_1) = V_1$, $k=1, \dots, N$, $N-2P > 0$. That is,

$$\text{sgn}((N-2)v_{1j}) = -v_{1j}.$$

$$\text{Hence } P < \frac{N}{2}. \quad \square$$

Lemma: $(WV')_j = Nv_j - 2v_k w_{kj}$

Proof: This lemma is a straight forward extension of *Theorem a.2*.

Theorem a.3: Let W be a synaptic matrix generated from the sum of outer products of a set of orthogonal neuron vectors. W is not normalized and N is the number of neurons. $w_{\max} = \max \{w_{ij}, i, j = 1, \dots, N\}$. A corrupted pattern with one bit error can be recovered (i.e., distance of 1) if $N/2 > w_{\max}$.

Proof : Let V be a pattern neural vector and V' its corrupted version with k -th bit in error. From Lemma above, $(WV')_j = Nv_j - 2v_k w_{kj}$, which must have the same sign as v_j in order for the network to converge to V in one step. Thus $(WV')_j v_j$ must be greater than 0. The corresponding sufficient condition is $N/2 > w_{\max}$. \square

In general, a corrupted pattern with m bits of error can be recovered if $N/2m > w_{\max}$.

Corollary: Let M be the set of indexes of erroneous bits, still V' is the corrupted vector of V . We have

$$(WV')_j = Nv_j - 2v_k \sum_{k \in M} w_{kj}$$

Proof : This lemma is a straight forward extension of *Theorem a.3*.

Theorem a.4: If $N/2m > w_{\max}$, then the neural network can be recovered from m bits of errors.

Proof : From corollary above, $(WV')_j = Nv_j - 2v_k \sum_{k \in M} w_{kj}$, which must have the same sign as v_j in order for the network to converge to V in one step. Thus $(WV')_j v_j$ must be greater than 0. The corresponding sufficient condition is $N/2m > w_{\max}$. \square

One can see that the smaller the entry values of the synaptic matrix, the better the error recovery capability.

It was mentioned in [10] that for orthogonal feature vectors, the capacity is 100% in comparing with the tradeoff between the memory capacity and the degree of fault-tolerance which has been estimated to be about 15% of B bits in *Hopfield-Little*

Model [9].

There are some proposition for the symmetric matrix which can be applied for the scheme base on the sum of outer product.

Theorem a.5 [6]: The trajectories in the state space follow contours of non-increasing energy in the above associative memory.

Proof :

W is symmetric and has non-negative diagonal elements when the mode of operation is asynchronous. And the detail of prove is referred to [6]. \square

Consider $sgn (W \cdot U)_i = U_i, i=1, \dots, k. k \leq n.$, It is obvious that the memory U_i are true eigenvectors of the weight matrix W .

Theorem a.6 : Global energy minima are formed at the memories for the k -fold degenerate spectral scheme.

proof :

The synaptic matrix constructed above is by summing of outer product as

$$W = \sum_{k \text{ vectors}} X X^T$$

and the energy function is

$$E(x) = -\frac{1}{2} X^T W X$$

It is positive definite on the set of $\{ -1, 0, 1 \}^n$ and the eigenvalues are the same, so it is k -fold degenerated. The detail proof can be found in [6].

Theorem a.7 [1]: The recursive network is free of cycles for synchronous updating.

Hebb's law as well as the project rule guarantee the absence of cycles for syn-

chronous operation.

Theorem a.8: Parasitic points cannot be formed by linear combinations of prototype vectors generated by the C -method.

Proof: This theorem can be proved by induction.

For a 2-neuron network, the two prototype vectors are $[1 \ 1]^T$ and $[1 \ -1]^T$, whose linear combination is

$$\alpha [1 \ 1]^T + \beta [1 \ -1]^T = [-1 \ -1] \text{ or } [-1 \ 1]$$

where α and β are nonzero integers. Thus, $\alpha + \beta = -1$ and $\alpha - \beta = -1$ or 1 . No solutions exist such that both α and β are nonzero integers.

Now assume the theorem holds for $N_1 = 2^P$, we want to prove that it also holds for $N_2 = 2^{P+1} = 2N_1$. The first N_1 vectors of the $2N_1$ prototype vectors can be transformed into the N_1 prototype vectors of the N_1 -neuron network by the $[1 \ 1] \rightarrow 1$ and $[-1 \ -1] \rightarrow -1$ substitutions. Hence linear combinations of any of these first N_1 vectors would not produce any parasitic points. By the same token, the last N_1 components of the $2N_1$ prototype vectors can be transformed into the N_1 prototype vector of the N_1 -neuron network by the $[1 \ -1] \rightarrow 1$ and $[-1 \ 1] \rightarrow -1$ substitutions. Hence linear combinations of any of these last N_1 vectors would not produce any parasitic points. We have checked the first two cases of linear combinations of vectors from the $2N_1$ prototype vectors. The only case remained to be checked is that part of vectors in the combination come from the first N_1 vectors and the rest come from the last N_1 vectors.

Then $V = V_i + \sum_{i \in Q} \dots$, Let $Q = i_1, \dots, i_q$

$$\begin{aligned}
E(V) &= -1/2 VWV \\
&= -1/2 (V_i + \sum_{\mu \in Q} Y_\mu)W(V_i + \sum_{\mu \in Q} Y_\mu) \\
&= -1/2 V_i WV_i - 1/2 \sum_{s \in Q} \sum_{t \in Q} Y_s W Y_t - 1/2 V_i W \sum_{\mu \in Q} Y_\mu - 1/2 \sum_{\mu \in Q} Y_\mu W V_i \\
&= -1/2 + 1/2(4q) - 1/2 \sum_{s \in Q} \sum_{t \in Q} 4V_{st} V_{tt} W_{st}
\end{aligned}$$

Hence if $h(V, V_i) < 4, (\delta E)_{\min} > 0$ and V always converge to V_i . \square

Lemma: $V_i W \sum_{\mu \in Q} Y_\mu$

$$\begin{aligned}
&= (\sum_{\mu \in Q} Y_\mu) W V_i \\
&= -2q
\end{aligned}$$

proof:

$$\begin{aligned}
&V_i W Y_\mu \\
&= V_i (W_{1\mu} i, \dots, W_{N\mu})^T (-2V_{i\mu}) \\
&= V_i (\sum_{e \in M} V_{e\mu} V_e) (-2V_{i\mu}) \\
&= -2V_{i\mu} V_{i\mu} \\
&= -2
\end{aligned}$$

$$\begin{aligned}
Y_\mu W V_i &= Y_\mu V_i = -2V_{i\mu} V_{i\mu} \\
&= -2
\end{aligned}$$

Hence $V_i W (\sum_{\mu \in Q} Y_\mu) = (\sum_{\mu \in Q} Y_\mu) W V_i = -2q \quad \square$

Lemma: $Y_s W Y_t = 4W_{st} V_{ts} V_{tt}$

proof :

$$Y_s W Y_t = Y_s (W_{1t}, \dots, W_{Nt})^T (-2V_{it}) = 4W_{st} V_{is} V_{it} \quad \square$$

For the heteroassociative memory, we present a procedure to synthesize a neural network which has limit cycles with length less than the number of neurons and the states of the limit cycle are orthogonal to each other.

Let V_1, V_2, \dots, V_n be a set of orthogonal states with N components. The following theorem describes a synaptic matrix which results in a limit cycle whose states are V_1, V_2, \dots, V_n .

Theorem a.9: The neural network with the synaptic matrix W' which is

$$W' = V_2 * V_1 + V_3 * V_2 + V_4 * V_3 + \dots + V_n * V_{n-1} + V_1 * V_n$$

(the sum of outer products) has a limit cycle of $V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow \dots \rightarrow V_n \rightarrow V_1$.

Proof : It is easy to see that $\text{sgn}(W' V_1) = V_2$, $\text{sgn}(W' V_2) = V_3$, ... $\text{sgn}(W' V_{n-1}) = V_n$, and $\text{sgn}(W' V_n) = V_1 \quad \square$

Remark: $V_i * V_j$ is an $N \times N$ matrix with entry value of $a_{lm} = v_{il} v_{jm}$.

Using the orthogonal vectors to synthesize a neural network with a limit cycle of length m (less than n), just pick any m vectors from V_1, V_2, \dots, V_n . Let these m vectors be $V_{i_1}, V_{i_2}, \dots, V_{i_m}$ and the synaptic matrix be $W = V_{i_2} * V_{i_1} + V_{i_3} * V_{i_2} + \dots + V_{i_1} * V_{i_m}$.

Then $\text{sgn}(W V_{i_1}) = V_{i_2}$, $\text{sgn}(W V_{i_2}) = V_{i_3}$, ... $\text{sgn}(W V_{i_m}) = V_{i_1}$. Thus the limit cycle is $V_{i_1} \rightarrow V_{i_2} \rightarrow \dots \rightarrow V_{i_m} \rightarrow V_{i_1}$ with length m .

The above are some properties and theorem found in those recursive network with orthogonal states. Examples and application of the associate memory constructing on the orthogonal vectors are presented in the next section.

APPENDIX B: Examples

Example 1:

For $N = 4$, let

$$V_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

There are two possible choices of V_2 :

$$\begin{bmatrix} -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \text{ or } \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}.$$

Only one of them can be selected because they are not orthogonal to each other. Pick

$$V_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} = \begin{bmatrix} V_{2a} \\ V_{2b} \end{bmatrix},$$

where $V_{2a} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ and $V_{2b} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$ and $V_1 V_2 = 0$. Now divide V_{2a} and V_{2b} into two subblocks:

$$\begin{bmatrix} 1 \end{bmatrix} \text{ and } \begin{bmatrix} -1 \end{bmatrix}.$$

That is $V_{2a}, V_{2b} \rightarrow \begin{bmatrix} +1 \\ -1 \end{bmatrix}$ or $\begin{bmatrix} -1 \\ +1 \end{bmatrix}$. Thus there are four possible choices of V_3 :

$$\begin{bmatrix} +1 \\ -1 \\ +1 \\ -1 \end{bmatrix}, \begin{bmatrix} +1 \\ -1 \\ -1 \\ +1 \end{bmatrix}, \begin{bmatrix} -1 \\ +1 \\ +1 \\ -1 \end{bmatrix} \text{ and } \begin{bmatrix} -1 \\ +1 \\ -1 \\ +1 \end{bmatrix}.$$

They are all orthogonal to V_1 and V_2 . \square

Example 2:

For $N = 8$, using the formula approach, the following orthogonal vectors can be obtained.

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

Picking up the vectors V_0 , V_1 and V_6 from the the above orthogonal vectors, we can construct a weight-matrix by applying *Strategy 1*:

$$W = (V_0 * V_0 + V_1 * V_1 + V_6 * V_6),$$

or using the normalization factor $\frac{1}{p}$, ensures that $-1 \leq w_{ij} \leq +1$.

$$W = \frac{1}{3} * (V_0 * V_0 + V_1 * V_1 + V_6 * V_6)$$

where

$$V_0 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad V_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \quad V_6 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}.$$

the matrix:

$$\begin{pmatrix} 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \end{pmatrix}$$

or the normalized one.

$$\begin{pmatrix} 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \end{pmatrix} * \frac{1}{3}$$

The synaptic matrix is symmetric and the positive definite. By computing the energy $E(v) = -\frac{1}{2}V^T \cdot W \cdot V$, following the slope of the energy function, it may go to

the local minima. The local minima can be found in the following states,

$$\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \\ -1 \\ 1 \\ -1 \\ -1 \\ -1 \\ 1 \end{pmatrix}$$

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

and global minima are

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

which are $V_0, V_1, V_6, -V_6, -V_1$ and $-V_0$. Note that there are no spurious states.

If there is 1 bit corrupted in vector V_6 , and new vector is named as V'_6 ,

$$V'_6 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ -1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \quad \text{and} \quad V_6 = \begin{bmatrix} 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix}$$

Applying the weight matrix, V'_6 can go back to V_6 through one step.

$$\text{sgn}(W * V'_6) = V_6$$

That is the result from *Theorem a.3*, and one bit error can be recovered.

Example 3:

When $N = 16$, using those approaches, we can get 16 orthogonal vectors. Still let

$W = V_1 * V_1 + \dots + V_k * V_k$, let $k = 3$, pick the vectors V_0, V_{13} and V_2 ,

$$V_0^T = [+1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1 +1],$$

$$V_2^T = [+1 +1 +1 +1 -1 -1 -1 -1 +1 +1 +1 +1 -1 -1 -1 -1],$$

$$V_{13}^T = [+1 -1 -1 +1 +1 -1 -1 +1 -1 +1 +1 -1 -1 +1 +1 -1].$$

constructing the synaptic matrix using the outer product scheme, the weight matrix is as following:

$$\begin{bmatrix} 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 & 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 & 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 & 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 & 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 & -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 & 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 & 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 & -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 & 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 & 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 & 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 \\ 1 & 3 & 3 & 1 & -1 & 1 & 1 & -1 & 3 & 1 & 1 & 3 & 1 & -1 & -1 & 1 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 & 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 & -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 & -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 \\ -1 & 1 & 1 & -1 & 1 & 3 & 3 & 1 & 1 & -1 & -1 & 1 & 3 & 1 & 1 & 3 \end{bmatrix}$$

V_0 , V_2 and V_{13} are the fix point of the memory. Suppose there two bits in V_{13} are corrupted, or suppose another state V_{13}' near V_{13} whose *Hamming Distance* is 2.

$V_{13}' = [1 -1 -1 1 1 -1 -1 1 -1 1 -1 -1 1 1 1 -1]$ and it is obvious that the different bits are the #11 and #13. By applying $sgn(W X)$, the state V_{13}' will converge to the fixed point V_{13} through one step.

$$sgn (W \cdot V_{13}') = V_{13}$$

That is what stated in the *Theorem a.4*, where $w_{\max} = 3$, $N = 16$ and number of the corrupted bit is $m = 2$, $\frac{N}{2m} > w_{\max}$.

Example 4:

We have four orthogonal vectors in example 1,

$$V_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad V_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \end{bmatrix} \quad V_3 = \begin{bmatrix} +1 \\ -1 \\ +1 \\ -1 \end{bmatrix} \quad \text{and} \quad V_4 = \begin{bmatrix} +1 \\ -1 \\ -1 \\ +1 \end{bmatrix}$$

To construct a heteroassociative memory described in *Section III*, the corresponding synaptic matrix is

$$W = V_2 * V_1 + V_3 * V_2 + V_4 * V_3 + V_1 * V_4 = \begin{bmatrix} 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 \\ 0 & 0 & -4 & 0 \\ 0 & -4 & 0 & 0 \end{bmatrix}$$

It is straightforward to see that

$$\text{sgn}(WV_1) = V_2,$$

$$\text{sgn}(WV_2) = V_3,$$

$$\text{sgn}(WV_3) = V_4,$$

and

$$\text{sgn}(WV_4) = V_1.$$

Hence the limit cycle is $V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow V_4 \rightarrow V_1$. It is easy to see that $-V_1 \rightarrow -V_2 \rightarrow -V_3 \rightarrow -V_4 \rightarrow -V_1$ is another limit cycle. It is interesting to see whether the rest of 8 vectors are also on certain limit cycles.

Let

$$V_5 = [1 \ 1 \ 1 \ -1]^T$$

then

$$\text{sgn}(WV_5) = [1 \ -1 \ -1 \ -1]^T = V_6,$$

$$\text{sgn}(WV_6) = [1 \ -1 \ 1 \ 1]^T = V_7,$$

$$\text{sgn}(WV_7) = [1 \ 1 \ -1 \ 1]^T = V_8,$$

and

$$\text{sgn}(WV_8) = [1 \ 1 \ 1 \ -1]^T = V_5,$$

Thus there is a limit cycle: $V_5 \rightarrow V_6 \rightarrow V_7 \rightarrow V_8 \rightarrow V_5$. $-V_5 \rightarrow -V_6 \rightarrow -V_7 \rightarrow -V_8 \rightarrow -V_5$ is also a limit cycle. Thus every state is in some limit cycle and the memory capacity is greatly increased.

A neural network constructed using those orthogonal vectors is introduced and we demonstrated the ability and properties of the neural network and some advantages comparing to those other models, the advantages are especially as no spurious states, energy non-increasing and performance of convergence to fix point.