

To appear in Neurocomputing

**On the solution of the parity problem by a single hidden
layer feedforward neural network**

Rudy Setiono

Department of Information Systems and Computer Science

National University of Singapore

Kent Ridge, Singapore 119260

REPUBLIC OF SINGAPORE

email: rudys@iscs.nus.sg

On the solution of the parity problem by a single hidden layer feedforward neural network

Abstract

It is known that the N-bit parity problem is solvable by a standard feedforward neural network having a single hidden layer consisting of $(N/2) + 1$ hidden units if N is even and $(N+1)/2$ hidden units if N is odd. The network does not allow a direct connection between the input layer and the output layer and the transfer function used in all hidden units and the output unit is the usual sigmoidal function $\sigma(x) = 1/(1 + \exp(-x))$. We show that such a solution can be easily obtained by solving a system of linear equations.

Keywords: Parity problem, feedforward network, sigmoid function.

1. Introduction

The parity problem is one of the more widely used problems for testing neural network training algorithms. This problem is a mapping problem where the domain set consists of all distinct N-bit binary vectors and the result of the mapping is 0 if the number of ones in the vector is even, and 1 otherwise. The problem is considered to be very hard since the output changes whenever any single bit in the input changes.

Many researchers have proposed solutions for this problem using feedforward neural networks. Stork and Allen [12] showed that the problem can be solved by a network with just two hidden units. They made the assumptions that the network consists of three layers, that the transfer function used is strictly monotonically increasing function and that no direct connection between the input layer and the output layer is allowed. The key to their results is the use of an unusual transfer function

$$f(x) = \frac{1}{N} \left(x - \frac{\cos(\pi x)}{\alpha\pi} \right),$$

where α is any constant greater than one. When direct connections between the input units and the output unit are allowed, it is shown by Brown [3] that in fact just one hidden unit is needed.

Minor [7] showed that when the sigmoidal function is used as the transfer function and direct connections from the input layer and the output layer are allowed, the required number of hidden units is $\lceil N/2 \rceil$, where $\lceil \cdot \rceil$ indicates truncation to the nearest integer.

In practice however, the most commonly used network architecture is one where

- there is one hidden layer with connections only between the input layer and the hidden layer and between the hidden layer and the output layer, and
- the transfer function used in the hidden units and the output unit is the sigmoidal function $\sigma(x) = 1/(1 + \exp(-x))$.

With this kind of network architecture, it has been previously thought that N hidden units are required to solve the N bit parity problem [9]. Many experimentations with neural network construction and training algorithms were tested with the assumption that indeed N hidden units are necessary for solving the problem [1, 2, 5, 8]. A neural network construction algorithm that employs the quasi-Newton method for minimizing the error function was recently proposed by Setiono and Hui [10]. This algorithm successfully built networks having less than N hidden units that were capable to solve the N bit parity problem for small values of N ranging from 4 to 8.

Indeed, it follows from a result of Sontag [11] that a sufficient number of hidden units for the network is $(N/2) + 1$ if N is even and $(N + 1)/2$ if N is odd. In this paper, an alternative proof of this result is given by explicitly computing the weights of the neural network. We show that these weights can, in fact, be easily found by solving a system of linear equations.

2. Main results

Let us begin this section by defining all the notations used in this paper.

- H is the number of hidden units in the network.
- p_i^N : the i th N -bit pattern, where $i = 1, 2, \dots, 2^N$. For simplicity, it is assumed throughout that the patterns are arranged in increasing order.

- $p_{i,k}^N$: the k th component of pattern p_i^N , it is either 0 or 1, $k = 1, 2, \dots, N$.
- w_j^N is a real valued weight vector of outgoing arcs connecting the input layer to the j th hidden unit, $j = 1, 2, \dots, H$. The dimension of this vector is $N + 1$ with the value of $w_{j,N+1}^N$ reflecting the bias in the j th hidden unit.
- v_j is a real valued weight of arc connecting the j th hidden unit to the output unit.
- τ is the bias value of the output unit.

For our proof on the sufficient number of hidden units for the N bit parity problem, we rely heavily on the function

$$y^N(x) = x_1 - x_2 + \dots + x_{N-1} - x_N + \frac{1}{2},$$

if N is even, and the function

$$y^N(x) = -x_1 + x_2 - \dots + x_{N-1} - x_N + \frac{1}{2},$$

if N is odd. These functions are defined to allow us to divide the input set of the problem into subsets where all patterns in each subset have the same function value. This is done in the following lemma which is trivial to prove.

Lemma 1 *Let S be the set of all 2^N distinct input patterns of the N -bit parity problem, then S can be divided into $N + 1$ subsets S_1, S_2, \dots, S_{N+1} such that the cardinality of S_i is $\binom{N}{i-1}$, and*

1. *for each element $p \in S_i, y^N(p) = i - 1 - \frac{N}{2}$ if N is odd*
2. *for each element $p \in S_i, y^N(p) = i - \frac{1}{2} - \frac{N}{2}$ if N is even,*

for all $i = 1, 2, \dots, N + 1$.

It follows from the above lemma that all patterns that belong to the same subset will have the same parity.

As an illustration on how the above lemma will be used, let us consider the simplest parity problem, which is the 2-bit parity problem, better known as the XOR problem.

It has been established that two hidden units are needed to solve this problem. Let us define the vector z^2 as the coefficients of the function $y^2(x)$:

$$z^2 = \begin{pmatrix} 1 \\ -1 \\ \frac{1}{2} \end{pmatrix}.$$

Let us also define the weight vectors from the input layer to each of the two hidden units in the network as follows

$$\begin{aligned} w_1^2 &= \alpha_1 z^2 \\ w_2^2 &= \alpha_2 z^2, \end{aligned}$$

where $|\alpha_1| \neq |\alpha_2| \neq 0$. A bias in each of the hidden unit is incorporated by appending a one to each of the input pattern. The bias in the output unit is denoted by τ . Since

$$\begin{aligned} \lim_{x \rightarrow -\infty} \sigma(x) &= 0 \quad \text{and} \\ \lim_{x \rightarrow \infty} \sigma(x) &= 1, \end{aligned}$$

we would like to find the values of v_1, v_2 and τ such that the following set of equations are satisfied

i	Pattern p_i^2	Equation
1	00 1	$\sigma(w_1^2 p_1^2) v_1 + \sigma(w_2^2 p_1^2) v_2 + \tau = -\beta$
2	01 1	$\sigma(w_1^2 p_2^2) v_1 + \sigma(w_2^2 p_2^2) v_2 + \tau = \beta$
3	10 1	$\sigma(w_1^2 p_3^2) v_1 + \sigma(w_2^2 p_3^2) v_2 + \tau = \beta$
4	11 1	$\sigma(w_1^2 p_4^2) v_1 + \sigma(w_2^2 p_4^2) v_2 + \tau = -\beta$

for some $\beta > 0$. Note that the product between the weight vector w and the pattern p_i is a scalar product, that is

$$w_j^N p_i^N = \sum_{k=1}^{N+1} w_{j,k}^N p_{i,k}^N = \alpha_j \sum_{k=1}^{N+1} z_k^N p_{i,k}^N.$$

At first glance, it seems that we have a set of overdetermined system of linear equations in hand. Lemma 1 however tells us that there should actually be no more than $N + 1$ equations. Indeed, the first equation and the fourth equation in the table are identical since $w_1^2 p_1^2 = w_1^2 p_4^2 = \alpha_1/2$ and $w_2^2 p_1^2 = w_2^2 p_4^2 = \alpha_2/2$. Rather than using the first three equations to solve for v_1, v_2 and τ , we replace equation 2 by

$$v_1 + v_2 + 2\tau = 0. \tag{1}$$

Since $w_1^2 p_2^2 = -w_1^2 p_1^2 = -\alpha_1/2$ and $w_2^2 p_2^2 = -w_2^2 p_1^2 = -\alpha_2/2$, we have

$$\begin{aligned} \sigma(w_1^2 p_2^2) v_1 + \sigma(w_2^2 p_2^2) v_2 + \tau &= (1 - \sigma(w_1^2 p_1^2)) v_1 + (1 - \sigma(w_2^2 p_1^2)) v_2 + \tau \\ &= -(\sigma(w_1^2 p_1^2) v_1 + \sigma(w_2^2 p_1^2) v_2 + \tau) + v_1 + v_2 + 2\tau. \end{aligned}$$

We see that the second equation will automatically be satisfied as long as the first equation and Eq. (1) are satisfied.

Another advantage of forcing Eq. (1) to be satisfied by v_1, v_2 and τ is that the solution can also be used for the 3-bit parity problem. Consider now a network with 3 input units and 2 hidden units for this problem and let us redefine a new vector z^3 to be the coefficient of the function $y^3(x)$

$$z^3 = \begin{pmatrix} -1 \\ 1 \\ -1 \\ \frac{1}{2} \end{pmatrix}$$

and the weight vectors

$$\begin{aligned} w_1^3 &= \alpha_1 z^3 \\ w_2^3 &= \alpha_2 z^3. \end{aligned}$$

Since

$$w_j^2 p_i^2 = w_j^3 p_i^3,$$

for all $i = 1, 2, 3, 4$ and $j = 1, 2$, the first four patterns will be correctly classified. Now we consider the remaining four patterns p_{9-i}^3 . Since pattern p_{9-i}^3 is a bit-wise complement of pattern p_i^3 and the number of bit is odd, the parity of these two patterns must be different. The following equations show that as long as pattern p_i is correctly classified, pattern p_{9-i} will also be correctly classified. First note that

$$\begin{aligned} y^3(p_{9-i}^3) &= -p_{9-i,1}^3 + p_{9-i,2}^3 - p_{9-i,3}^3 + \frac{1}{2} \\ &= -1 + p_{9-i,2}^3 - p_{9-i,3}^3 + \frac{1}{2} \\ &= (1 - p_{i,2}^3) - (1 - p_{i,3}^3) - \frac{1}{2} \\ &= -\left(p_{i,1}^2 - p_{i,2}^2 + \frac{1}{2}\right) \\ &= -y^2(p_i^2) \end{aligned}$$

Since $w_j^3 = \alpha_j z^3$, we have

$$\begin{aligned} w_j^3 p_{9-i}^3 &= \alpha_j z^3 p_{9-i}^3 \\ &= \alpha_j y^3(p_{9-i}^3) \\ &= -\alpha_j y^2(p_i^2) \\ &= -\alpha_j y^3(p_i^3) \\ &= -\alpha_j z^3 p_i^3 \\ &= -w_j^3 p_i^3. \end{aligned}$$

The fourth equality above follows from the fact that the first digit of the pattern p_i^3 is zero. Hence

$$\begin{aligned} \sigma(w_1^3 p_{9-i}^3) v_1 + \sigma(w_2^3 p_{9-i}^3) v_2 + \tau &= \sigma(-w_1^3 p_i^3) v_1 + \sigma(-w_2^3 p_i^3) v_2 + \tau \\ &= \left(1 - \sigma(w_1^3 p_i^3)\right) v_1 + \left(1 - \sigma(w_2^3 p_i^3)\right) v_2 + \tau \\ &= -\left(\sigma(w_1^3 p_i^3) + \sigma(w_2^3 p_i^3) + \tau\right) + v_1 + v_2 + 2\tau \\ &= -\left(\sigma(w_1^3 p_i^3) + \sigma(w_2^3 p_i^3) + \tau\right). \end{aligned}$$

Consequently, if pattern p_i^3 is correctly classified, pattern p_{9-i}^3 will also be correctly classified.

If we define the error in prediction to be the absolute difference between the network output value and the actual target value

$$\begin{aligned} \text{error} &= |\text{Network output} - \text{target value}| \\ &= \left| \sigma \left(\sum_{j=1}^H \sigma(w_j^N p_i^N) v_j + \tau \right) - \text{target value} \right|, \end{aligned}$$

then the value of error is equal to $1/(1 + e^\beta)$ for all patterns. This implies that the output can be made to within any accuracy requirement by selecting a value of β that is sufficiently large.

The output values of two neural networks each with two hidden units for the 3-bit parity problem are plotted in Figure 1. The weights of the first network have been obtained by setting the values of $\alpha_1 = 0.5, \alpha_2 = 1$ and $\beta = 1$. For the second network, the same α values are used but the value of β is set to 10.

The following two lemmas generalize our results for all values of N

Lemma 2 Solution of the N-bit parity problem where N is even *Let N be even and positive, $H = (N/2) + 1$ and $\beta > 0$. Let z^N be an $N + 1$ dimensional array*

$$z^N = \begin{pmatrix} 1 \\ -1 \\ \vdots \\ 1 \\ -1 \\ \frac{1}{2} \end{pmatrix}.$$

Let S_1, S_2, \dots, S_{N+1} be subsets of the input patterns as defined in Lemma 1 and let $p_{kj}^N, j = 1, 2, \dots, H$ be any pattern that belongs to the subset $S_{j+(N/2)}$. A neural network with H hidden units with connection weights from the input units to the hidden units defined as $w_j^N = \alpha_j z^N$ for $j = 1, 2, \dots, H$ with $|\alpha_j| \neq |\alpha_{\bar{j}}|$ if $j \neq \bar{j}$, $\alpha_j \neq 0$ for all j and with connection weights from the hidden units to the output unit v_1, v_2, \dots, v_H and the output

unit bias τ defined as the solution of the $H + 1$ by $H + 1$ system of linear equations

$$\begin{aligned}
\sum_{j=1}^H \left(\sigma(w_j^N p_{k_1}^N) v_j \right) + \tau &= -\beta \\
\sum_{j=1}^H \left(\sigma(w_j^N p_{k_2}^N) v_j \right) + \tau &= \beta \\
&\dots = \dots \\
\sum_{j=1}^H \left(\sigma(w_j^N p_{k_H}^N) v_j \right) + \tau &= \begin{cases} -\beta & \text{if } H \text{ is odd} \\ \beta & \text{otherwise,} \end{cases} \\
\sum_{j=1}^H v_j + 2\tau &= 0
\end{aligned} \tag{2}$$

correctly classifies all 2^N input patterns of the problem.

Proof First note that in view of Lemma 1 and the definition of the weights w_j^N , correct classification of a pattern $p_\ell^N \in S_i$ implies correct classification of all $\binom{N}{i-1}$ patterns in that set. If N is even, then patterns in the set $S_{1+\frac{N}{2}}$ will have even parity, patterns in the set $S_{2+\frac{N}{2}}$ will have odd parity etc. The first H equations in Eq. (2) ensure that all these patterns are correctly classified. The last equation will be used to ensure that the remaining patterns in set $S_1 \cup S_2 \dots \cup S_{\frac{N}{2}}$ will also be classified correctly.

Let us now consider a pattern $p_\ell^N \in S_i$ for any $i = 1, 2, \dots, \frac{N}{2}$. Let p_m^N be any element of the set S_{N+1-i} , then from Lemma 1 we have that

$$y^N(p_\ell^N) = -y^N(p_m^N).$$

Hence for the pattern p_ℓ^N we have that

$$\begin{aligned}
\sum_{j=1}^H \left(\sigma(w_j^N p_\ell^N) v_j \right) + \tau &= \sum_{j=1}^H \left(\sigma(-w_j^N p_m^N) \right) v_j + \tau \\
&= \sum_{j=1}^H \left(1 - \sigma(w_j^N p_m^N) \right) v_j + \tau \\
&= - \left(\sum_{j=1}^H \left(\sigma(w_j^N p_m^N) v_j \right) + \tau \right) + \sum_{j=1}^H v_j + 2\tau \\
&= \begin{cases} -\beta & \text{if } H - i \text{ is even} \\ \beta & \text{otherwise.} \end{cases}
\end{aligned}$$

Hence we have shown that correct classification of all 2^N patterns is guaranteed by the solution of the system of linear equations (2). \square

We note that the values of α_j must be chosen to ensure that the linear system of equations (2) has a solution. By choosing α_j as described in Lemma 2, we prove in the Appendix that the columns of the coefficient matrix in Eq. (2) are linearly independent.

The solution for the even N-bit parity problem can be used to solve the problem with one more input bit as shown by Lemma 3 below. The proof of this lemma is similar to the proof of Lemma 2 and is omitted.

Lemma 3 Solution of the N-bit parity problem where N is odd *Let N be odd and greater than 1 and $H = (N + 1)/2$. Let z^N be an $N + 1$ dimensional array*

$$z^N = \begin{pmatrix} -1 \\ 1 \\ \vdots \\ 1 \\ -1 \\ \frac{1}{2} \end{pmatrix}.$$

Let $\alpha_j, j = 1, 2, \dots, H$ have the same values as in Lemma 2 and let v_1, v_2, \dots, v_H and τ be the solution of the linear system (2) for the (N-1)-bit parity problem as defined in Lemma 2. Define the weight vectors $w_j^N = \alpha_j z^N$ for $j = 1, 2, \dots, H$. Then the neural network with weights w_j^N for the connections between the input units and the hidden units, weights v_j^N for the connections between the hidden units and the output unit and an output unit bias of τ correctly classifies all input patterns of the problem.

Combining the results of Lemmas 2 and 3, we state below our main result.

Theorem 1 *The N-bit parity problem is solvable by a feedforward neural network with a single hidden layer. Only connections between input layer and hidden layer and between hidden layer and output layer are allowed in the network. When the sigmoidal transfer function is used in the hidden units and the output units, a sufficient number of hidden units is $N/2 + 1$ if N is even and $(N + 1)/2$ if N is odd.*

3. Conclusion

We have shown how the N -bit parity problem can be solved by a feedforward neural network having $N/2 + 1$ hidden units if N is even and by a network having $(N + 1)/2$ hidden units if N is odd. The weights for the arcs connecting the input units and the hidden units can be determined trivially, while the weights for the arcs connecting the hidden units and the output unit can be easily obtained by solving a system of linear equations.

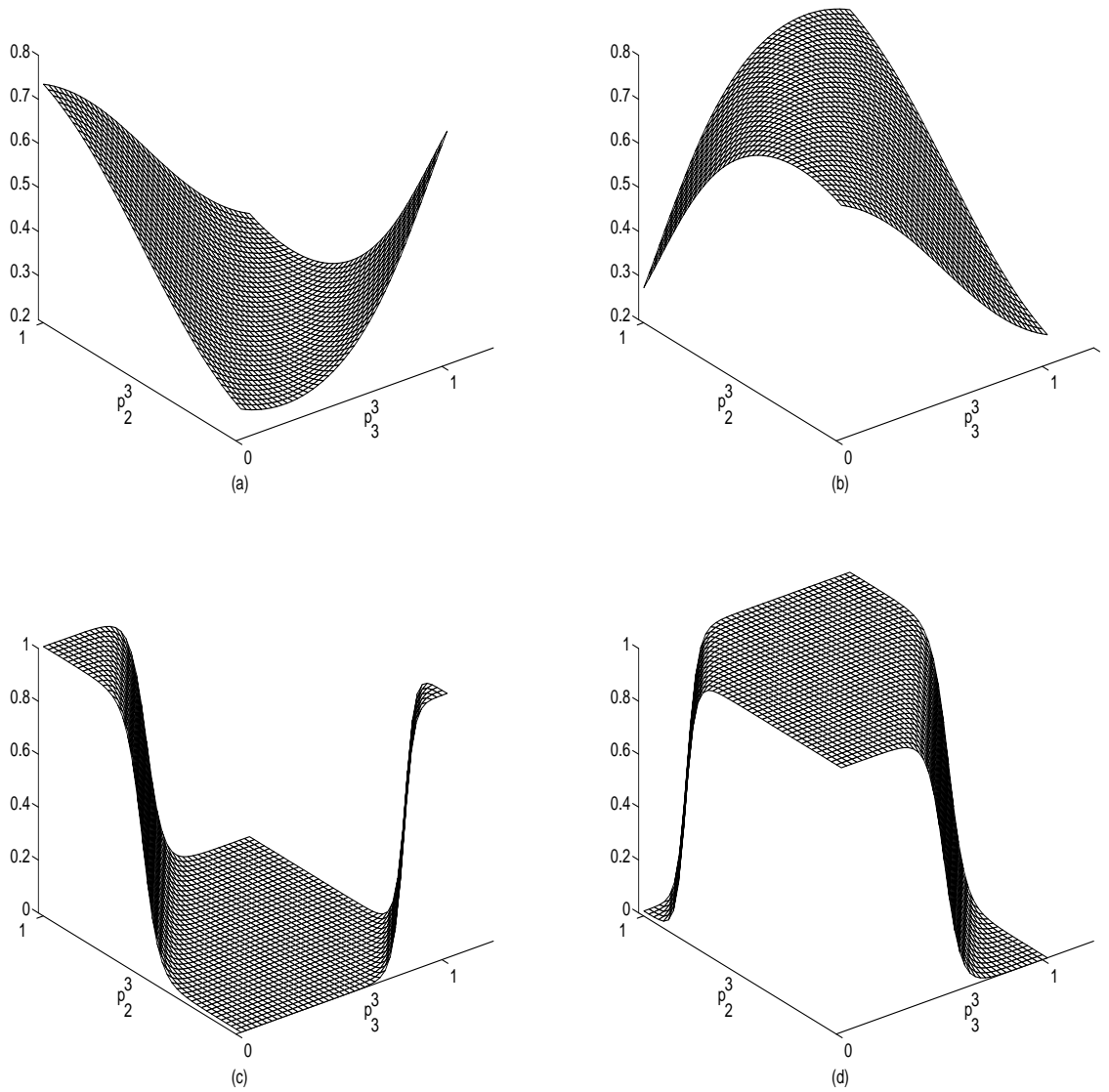


Figure 1: Output values of two neural networks with two hidden units for the 3-bit parity problem plotted as a function of p_2^3 and p_3^3 . Plots (a) and (b) are from the first network and (c) and (d) are from the second network. Weights for both networks are obtained by solving a 3×3 system of linear equations. The values of the parameters are $\alpha_1 = 0.5, \alpha_2 = 1, \beta = 1$ for (a) and (b) and $\beta = 10$ for (c) and (d). For (a) and (c) the value of p_1^3 is set to 0 (hence these plots also depict the predicted values for the XOR problem) and for (b) and (d) the value of p_1^3 is set to 1.

Appendix

Consider the system of linear equations

$$Mx = 0$$

where

$$M_{ij} = \begin{cases} 1 & \text{if } i = 1, 2, \dots, H; j = H + 1 \\ 1 & \text{if } i = H + 1; j = 1, 2, \dots, H \\ 2 & \text{if } i = H + 1; j = H + 1 \\ \sigma(h(2i - 1)\alpha_j/2) & \text{otherwise} \end{cases}$$

$$x = (v_1, v_2, \dots, v_H, \tau)^T,$$

where $h > 0$, $|h(2H - 1)\alpha_j/2| < \pi$, $|\alpha_i| \neq |\alpha_j|$ if $i \neq j$ and $\alpha_i \neq 0, \forall i = 1, 2, \dots, H$. We shall prove that zero is the only solution of this linear system.

Upon removing τ from the system, we get a new system of H linear equations

$$Cv = 0$$

where

$$\begin{aligned} C_{ij} &= \sigma(h(2i - 1)\alpha_j/2) - \frac{1}{2} \\ &= \frac{1}{2} \tanh[h(2i - 1)\alpha_j/4] \\ &= \frac{1}{2} \sum_{k=1}^{\infty} \left(2^{2k}(2^{2k} - 1)/(2k)!\right) B_{2k} [h(2i - 1)\alpha_j/4]^{2k-1}, \end{aligned}$$

where $B_{2k}, k = 1, 2, \dots$ denote Bernoulli numbers [4]. For all $i = 1, 2 \dots H$, we have

$$\begin{aligned} 0 &= \sum_{j=1}^H C_{ij} v_j \\ &= \frac{1}{2} \sum_{j=1}^H \left(\sum_{k=0}^{\infty} \left(2^{2k}(2^{2k} - 1)/(2k)!\right) B_{2k} [h(2i - 1)\alpha_j/4]^{2k-1} \right) v_j \\ &= \frac{1}{2} \sum_{k=0}^{\infty} \left(2^{2k}(2^{2k} - 1)/(2k)!\right) B_{2k} \left(\sum_{j=1}^H [h(2i - 1)\alpha_j/4]^{2k-1} v_j \right) \\ &= \frac{1}{2} \sum_{k=0}^{\infty} \left(2^{2k}(2^{2k} - 1)/(2k)!\right) B_{2k} [h(2i - 1)/4]^{2k-1} \sum_{j=1}^H \alpha_j^{2k-1} v_j \\ &= \frac{1}{2} \sum_{k=0}^{\infty} a_{i,k} h^{2k-1}, \end{aligned}$$

where

$$a_{i,k} = \left(2^{2k}(2^{2k} - 1)/(2k)!\right) B_{2k} [(2i - 1)/4]^{2k-1} \sum_{j=1}^H \alpha_j^{2k-1} v_j.$$

The equation $\sum_{k=0}^{\infty} a_{i,k} h^{2k-1} = 0$ holds for infinitely many different values of $h > 0$, hence we must have $a_{i,k} = 0$ or equivalently

$$\sum_{j=1}^H \alpha_j^{2k-1} v_j = 0$$

for all $k = 1, 2, \dots$. The first H of these equations can be written in the matrix form

$$\begin{pmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_H \\ \alpha_1^3 & \alpha_2^3 & \dots & \alpha_H^3 \\ \vdots & \vdots & & \vdots \\ \alpha_1^{(2H-1)} & \alpha_2^{(2H-1)} & \dots & \alpha_H^{(2H-1)} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_H \end{pmatrix} = 0$$

or equivalently

$$VDv = 0$$

where V is the Vandermonde matrix [6]

$$V = \begin{pmatrix} 1 & 1 & \dots & 1 \\ \alpha_1^2 & \alpha_2^2 & \dots & \alpha_H^2 \\ \vdots & \vdots & & \vdots \\ \alpha_1^{(2H-2)} & \alpha_2^{(2H-2)} & \dots & \alpha_H^{(2H-2)} \end{pmatrix}$$

and D is a diagonal matrix with $\alpha_1, \alpha_2, \dots, \alpha_H$ on the diagonal. The condition $|\alpha_i| \neq |\alpha_j| \forall i \neq j$ ensures that the matrix V is invertible, while the condition $\alpha_i \neq 0 \forall i$ ensures that D is invertible. We conclude that $v = 0$ is the only solution to the linear system of equations $VDv = 0$ and consequently $x = 0$ is the only solution to the system of linear equations $Mx = 0$.

Acknowledgements

The author wishes to thank two anonymous reviewers for their comments and suggestions.

References

- [1] T. Ash, Dynamic node creation, *Connection Science* 1 (4)(1989) 365–375.
- [2] K.P. Bennett and O.L. Mangasarian, Neural network training via linear programming. in: P.M. Pardalos, ed., *Advances in Optimization and Parallel Computing* (Elsevier Science Publishers B.V., Amsterdam, 1992) 56–67.
- [3] D.A. Brown, Solving the N-bit parity problem with only one hidden unit, *Neural Networks* 6 (1993) 607–608.
- [4] CRC Standard Mathematical Tables and Formulae, 29th Edition (CRC Press, Boca Raton, FL) page 280.
- [5] O. Fujita, Optimization of the hidden unit function in feedforward neural networks, *Neural Networks* 5 (1992) 755–764.
- [6] G.H. Golub and C. F. van Loan, *Matrix Computation*, (The Johns Hopkins University Press, Baltimore) (1983) page 119.
- [7] J.M. Minor, Parity with two layer feedforward nets, *Neural Networks* 6 (1993) 705–707.
- [8] M.F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, *Neural Networks* 6 (1993) 525–533.
- [9] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing*, (MIT Press, Cambridge, MA, 1986).
- [10] R. Setiono and L.C.K. Hui, Use of quasi-Newton method in a feedforward neural network construction algorithm, *IEEE Transactions on Neural Networks* 6 (1) (1995) 273–277.
- [11] E.D. Sontag, Feedforward nets for interpolation and classification, *Journal of Computer and System Sciences* 45 (1992) 20–48.
- [12] D.G. Stork and J.D. Allen, How to solve the N-bit parity problem with two hidden units, *Neural Networks* 5 (1992) 923–926.