A HYPERBOLIC TANGENT REPLACEMENT BY THIRD ORDER POLYNOMIAL APPROXIMATION

Darío Baptista, Fernando Morgado-Dias

Madeira Interactive Technologies Institute and Centro de Competências de Ciências Exatas e da Engenharia, Universidade da Madeira
Campus da Penteada, 9000-039 Funchal, Madeira, Portugal.
Tel: +351 291-705150/1, Fax: +351 291-705199

Abstract: Artificial Neural Networks have been widely used in different fields ranging from social sciences to engineering. Many of these applications have reached a hardware implementation phase and have been reported in scientific papers. Unfortunately most of the implementations suffer from a low precision of the hyperbolic tangent replacement which has been the most common problem and the most resource consuming block in terms of hardware. This paper proposes a high resolution hyperbolic tangent substitute which is far more modest in consumed resources than most of the solutions proposed in the literature, by using the simplest solution in order to obtain the lowest error proposed so far with a set of 25 polynomials of third order, obtained with Chebyshev approximations. Copyright CONTROLO2012

Keywords: Artificial Neural Networks, Function Approximation, Hyperbolic Tangent.

1. INTRODUCTION

Artificial Neural Networks (ANN) have been actively studied since the 1980s. During these decades, in some of the implementations, a necessity of physical implementation of the ANN trained arose, instead of using it inside a computer. One of the first physical implementations was developed by Nestor and Intel in 1993 for an Optical Character Recognition (OCR), which uses Radial Basis Functions (Morgado-Dias, 2004). The OCR application could not afford a full computer. The need for physical implementations also arises in medical applications of ANN which work with the human body. The work referred in (Stieglitz, 2006) contains many examples of neural vision systems that cannot be used attached to computers. Most of these implementations use the hyperbolic tangent and this is where it becomes more difficult to build a physical implementations since the simplest solutions are highly resource consuming or take a long time to produce a result. Many solutions have been tested: piecewise linear functions, Taylor Series, Elliot functions, polynomial approximations, and the CORDIC algorithm to name but a few. The best solutions found in the review of the literature are: (Pinto, 2006) that with a set of five 5th order polynomial approximations with a maximum error of $8 \times 10^{-5}$ (using the sigmoid function) and (Ferreira, 2007) that with a piecewise linear approximation determined by an optimized algorithm which achieved $2.18 \times 10^{-5}$ error in the hyperbolic tangent. This work searches for the best replacement for the hyperbolic tangent as an activation function: the simplest physical implementation with the lowest error. This solution is meant to replace the hyperbolic tangent after the usual training process with a Personal Computer. To obtain such a solution, different polynomial approximations were tested: Lagrange, Chebyshev and Least Squares. As will be demonstrated further on, the Chebyshev approximation is the best solution.

2. THEORETICAL DESCRIPTION
This section describes the theoretical aspects of the most usual methods used to approximate a function through a polynomial, namely the Lagrange and Chebyshev interpolations and the Least Squares method.

1.1 Lagrange interpolation

Consider a function \( f \) defined in interval \([a, b] \subseteq \mathbb{R}\) and a set \( X \) where there are \( n+1 \) different nodes and a set \( Y \), where each element is a value of the function to interpolate for each element of \( X \):

\[
X = \{x_0, x_1, x_2, x_3, \ldots, x_i\} \quad (1)
\]

\[
Y = \{f(x_0), f(x_1), \ldots, f(x_i)\} \quad (2)
\]

The polynomials of Lagrange on the points of the set \( X \) can be built using the following equation (Rio, 2002):

\[
L_k = \prod_{i \neq k}^{n} \frac{x - x_i}{x_k - x_i} \quad (3)
\]

After building Lagrange polynomials, calculate the Lagrange’s polynomial interpolate is a simple task. Thus, the polynomial interpolates \( P(x) \), whose grade is equal to \( n \) and which interpolates the values of set \( Y \), is given by (Rio, 2002):

\[
P(x) = \sum_{k=0}^{n-1} L_k f(x_k) \quad (4)
\]

However, in some cases, this method will prove not to be the most convenient representation of the polynomial interpolation. On the one hand, it is possible to obtain this polynomial with fewer arithmetic operations. On the other hand, Lagrange polynomials (equation 4) depend on a certain set of points, and with a change of position or in the number of these points, the Lagrange’s polynomial is changed completely (Rio, 2002).

1.2 Least Square Method

Consider a function \( f \) defined into interval \([a, b] \subseteq \mathbb{R}\), consider a set \( Y \), where there are \( m \) equidistant nodes:

\[
Y = \{f(x_1), f(x_2), \ldots, f(x_m)\} \quad (5)
\]

The objective of the Least Squares method is to find a polynomial function \( P(x) \) of grade equal \( n \). Using method of least squares it is possible, to \( n < m \), find a polynomial function to adjust the data in order to minimize the total quadratic error (Valença, 1993),

\[
S = \sum_{i=1}^{n} (f(x_i) - P(x_i))^2 \quad (6)
\]

The Least Squares method minimizes the sum of squared residuals (it is also called the Sum of Squared Errors, SSE) (Valença, 1993). For the SSE to be null it is necessary to verify the following condition:

\[
\begin{bmatrix}
P(x_1) = f(x_1) \\
P(x_2) = f(x_2) \\
\vdots \\
P(x_m) = f(x_m)
\end{bmatrix}
\Rightarrow
\begin{bmatrix}
c_0 + c_1 x_1 + \cdots + c_n x_1^n = f(x_1) \\
c_0 + c_1 x_2 + \cdots + c_n x_2^n = f(x_2) \\
\vdots \\
c_0 + c_1 x_m + \cdots + c_n x_m^n = f(x_m)
\end{bmatrix}
\quad (7)
\]

Expression 7 can be represented by the following matrix equation (Valença, 1993):

\[
\begin{bmatrix}
1 & x_1 & x_1^2 & \cdots & x_1^n \\
1 & x_2 & x_2^2 & \cdots & x_2^n \\
1 & x_3 & x_3^2 & \cdots & x_3^n \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
1 & x_{m-1} & x_{m-1}^2 & \cdots & x_{m-1}^n \\
1 & x_m & x_m^2 & \cdots & x_m^n
\end{bmatrix}
\begin{bmatrix}
c_0 \\
c_1 \\
c_2 \\
\vdots \\
c_n
\end{bmatrix}
= \begin{bmatrix}
f(x_1) \\
f(x_2) \\
f(x_3) \\
\vdots \\
f(x_m)
\end{bmatrix}
\quad (8)
\]

However, minimizing the SSE is not simple because since \( m > n \) (a system with more equations than unknown variables) the system is over-determined. In some cases, this kind of systems cannot be solved, because \( y \) is not a linear combination of the values of the matrix \( V \)'s column. Therefore, it is necessary that the matrix \( V \) has \( m \) vectors of linearly independent columns (Valença, 1993).

Figure 1 illustrates the geometric description of Least Square’s method. The parallelogram represents the subspace \( \text{span}(V) \), generated by matrix \( V \), to which the vector \( y \) usually does not belong (Valença, 1993).

![Fig. 1: Geometric description of least square’s problem](image)

Considering a matrix \( V \subseteq \mathbb{R}^{m \times n} \) (where \( m > n \)) and a vector \( f(x) \), we look for a vector \( c \in \mathbb{R}^n \) such that residual value, \( r \), is smallest possible. Figure 1 shows that length of the residual is minimal when \( r \perp \text{span}(V) \) (Valença, 1993), i.e.:

\[
V^T r = 0
\]

\[
\Rightarrow V^T (y - Vc) = 0
\]

\[
\Rightarrow V^T Vc = V^T y
\]

Thus, to calculate the polynomials coefficients of \( P(x) \) using the Least Square’s method, it is necessary to solve the system of equation 9 because it is a
possible and determined system. The system solution is approximate of original system, $Vc \approx y$, which minimizes the residual value. However, the system has a disadvantage because when the degree of polynomial is increased, the columns of the matrix $V$ are approach to the linear dependence (Valença, 1993).

1.3 Chebyshev interpolation

A widely used class of orthogonal polynomials is the Chebyshev polynomial (Gil, 2007). The Chebyshev polynomials, $T_n(x)$, are polynomial in $x$ of degree $n$, generated by the following three-term recurrence relation (Gil, 2007) (Mason, 2003).

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x) \quad (10)$$

Where $T_0(x) = 1$ and $T_1(x) = x$. Note that these polynomials, of degree $n$, are orthogonal in the interval $[-1; 1]$. The Chebyshev polynomial $T_n(x)$ have $n$ different symmetric (Peixoto, 2008) and roots (Gil, 2007)(Mason, 2003). These values are called Chebyshev abscissas or Chebyshev nodes and they can be obtained by the following expression (Gil, 2007):

$$x_i = \cos \left( \frac{2i - 1}{2n + 2} \pi \right) \quad (11)$$

The distribution of these nodes, on $x$-axis, are not equidistant. The following scheme shows a sketch of the nodes’ distribution (Mason, 2003).

![Fig. 2: Scheme of distribution of 5, 10 and n Chebyshev nodes](image)

It is important to remember that expression 11, calculates the Chebyshev nodes in the interval $[-1:1]$. However, if one wishes to calculate $n$ nodes in any interval (ex.: $[a, b]$), it is necessary to use the following expression (Mason, 2003):

$$x_i = \frac{1}{a - b} \cos \left( \frac{2i - 1}{2n + 2} \pi \right) + \frac{1}{2}(a + b) \quad (12)$$

Where $i=0, 1, ..., n$. Nevertheless, the values found with expression 12 need to be translated to interval $[-1; 1]$, because in this interval the Chebyshev polynomials are orthogonal (Mason, 2003). At the end, a change of scale in the $x$-axis is necessary. Consider a function $f$, defined in interval $[a, b] \subseteq \mathbb{R}$ and a set $X$ with $n$ Chebyshev nodes, where:

$$X = \{x_1, x_2, ..., x_{n-1}, x_n\} \subseteq [a; b] \quad (13)$$

and consider the set $Y$, where each element is the value of the function to interpolate for each Chebyshev node.

$$Y = \{f(x_0), f(x_1), ..., f(x_{n-1}), f(x_n)\} \quad (14)$$

However, it is necessary that Chebyshev nodes are within the interval $[-1, 1]$, because this is where the Chebyshev polynomials are orthogonal (Gil, 2007) (Mason, 2003). Therefore, one must ensure that the length of the interval $[a, b]$ will be the same in the interval $[-1, 1]$, i.e., the length between $a$ and $b$ must be 2.

$$k(b - a) = 2 \Leftrightarrow k = \frac{2}{b - a}, \quad k \in \mathbb{R} \quad (15)$$

Therefore, the extremes of the interval $[a, b]$ must be multiplied by factor $k$ (eq. 15) so that the length between them is equal to 2. It is also necessary to multiply each of the values of set $X$ to correspond to the new interval.

![Fig. 3: Scheme of distribution of Chebyshev nodes in the values of set $X$](image)

Now, it is necessary that the values of set $X$ move into positions between -1 and 1. Hence, simply subtract each set $X$’s values by $(ka+1)$.

$$k = \frac{2}{b - a} \quad (16)$$

The values of the set $Y$, do not suffer any change. Thus, after scaling and translation, the polynomial approach can be calculated according to equation (Gil, 2007)(Mason, 2003)

$$p(x) \approx \sum_{i=0}^{n} c_i T_i(x) - \frac{1}{2} c_0 \quad (17)$$

where $c_i$ is the coefficient of order $n$ and can be calculate through the following expression (Mason, 2003):
\[ c_k = \frac{2}{n} \sum_{z=0}^{n-1} f(x_z) T_k(x_z) \quad (19) \]

Obviously, the ideal would be for \( n \) to equal infinity. However, this is not possible. For fixed \( n \), expression 19 is a polynomial in \( x \) which approximates the function \( f(x) \) in the interval \([-1, 1]\). It is important to mention that, the higher the value of \( n \), the better approximation which will be achieved (Gil, 2007) (Mason, 2003).

3. RESULTS

In order to determine the quality of the approximation obtained by the proposed solutions, they are compared with the real hyperbolic tangent. Considering that the hyperbolic tangent function is symmetric about the origin, i.e., \( f(-x) = -f(x) \), it is possible to reduce the part of the hyperbolic tangent that is effectively implemented and based in this part implement the rest of the function. Thus, in this function, the domain was divided (only for \( x > 0 \)) into segments and each one was approximated by a polynomial function. For the present implementation, the domain between \( 0 < x < 6 \) was divided to obtain, in each sub-interval, a polynomial with a Mean Square Error (MSE) equal to \( 1 \times 10^{-14} \). For \( x > 6 \), the function is interpolated by a constant value, whereas the remaining intervals used third order polynomials.

The number and the division of the intervals is dependent on the approximation. Once the MSE reaches the predefined value of \( 1 \times 10^{-14} \) a new interval must be inserted. As a result the number of intervals and maximum errors will be different for each approximation although the MSE will be the same.

Figures 6, 7 and 8 show the error and the division of hyperbolic tangent to different kinds of interpolations.
Table 1 shows the different maximum and minimum errors for each approximation of the hyperbolic tangent.

Table 1 Maximum positive and negative errors and number of intervals for each kind of polynomial approximation.

<table>
<thead>
<tr>
<th>Kind of polynomial approximation</th>
<th>Maximum Positive error</th>
<th>Maximum Negative error</th>
<th>Number of intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lagrange interpolation</td>
<td>1.832x10^{-7}</td>
<td>-1.935x10^{-7}</td>
<td>27</td>
</tr>
<tr>
<td>Least Square Method</td>
<td>1.664x10^{-7}</td>
<td>-1.955x10^{-7}</td>
<td>25</td>
</tr>
<tr>
<td>Chebyshev interpolation</td>
<td>1.614x10^{-7}</td>
<td>-1.929x10^{-7}</td>
<td>25</td>
</tr>
</tbody>
</table>

CONCLUSION

In this paper a high resolution low resources alternative to the hyperbolic tangent is presented. It is based in polynomial approximations of 3rd order, selected to avoid the use of many multiplications. This solution shows a maximum error of 1.929x10^{-7} if the Chebyshev interpolation is selected and MSE of 1x10^{-14}.

From the results presented it is possible to conclude that the Chebyshev's approximation is the best solution. These results can be explained because of the way in which the nodes are distributed. In the Chebyshev's approximation, the nodes are more concentrated in the extremity of each sub-interval while the m nodes used, in each sub-interval in the method of Least Squares are equidistantly separated.

ACKNOWLEDGMENTS

The authors would like to acknowledge the Portuguese Foundation for Science and Technology for their support for this work through project PEst-OE/EEI/LA0009/2011.

REFERENCES


