ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 181-186

# VLSI Complexity Reduction by Piece-Wise Approximation of the Sigmoid Function①

Valeriu Beiu[†,‡], Jan A. Peperstraete[†], Joos Vandewalle[†], and Rudy Lauwereins[†,②]

[†] Katholieke Universiteit Leuven, Department of Electrical Engineering
Division ESAT, Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium

[‡] on leave of absence from "Politehnica" University of Bucharest
Computer Science, Spl. Independentei 313, RO-77206 Bucharest, România

**Abstract.** The paper is devoted to show that there are *simple* and *accurate* ways to compute a sigmoid nonlinearity in digital hardware by piece-wise linearization. This is done as the computations involved are complex, but even more interesting, for the data compression performed by the sigmoid, which can half the number of Boolean functions to be further implemented. We detail such an approximation algorithm, and analyze its accuracy, showing improvements over the previous known algorithms. The algorithm uses only additions/subtractions and shifts (multiplications by powers of 2), supporting our claim of "simplicity".

## 1. Introduction

One of the difficult problems encountered when implementing artificial neural networks in digital hardware is the nonlinearity used after the weighted summation of the inputs. In this paper we shall consider only sigmoid activation functions like the family of sigmoids drawn in figure 1, and discuss on two related "complexity" aspects:

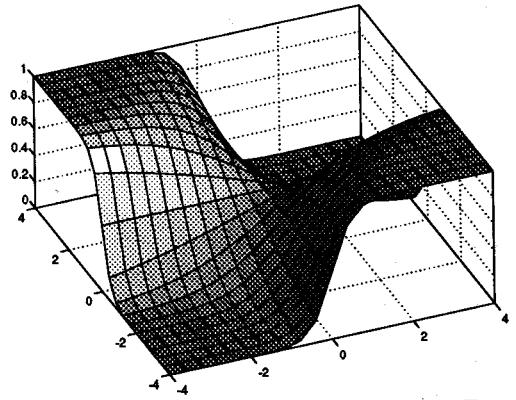



**Fig. 1.** 3D classical sigmoid function $f(x,T)$.

① This research work was partly carried out in the framework of a **Concerted Research Action** of the Flemish Community, entitled: ***"Applicable Neural Networks"***. The scientific responsibility is assumed by the authors.

② Senior Research Associate of the Belgian National Fund for Scientific Research.

ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 181-186

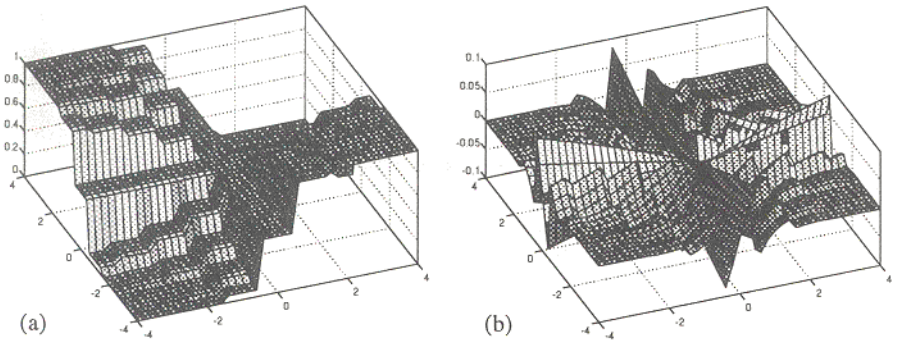



**Fig. 2.** Sum-of-steps approximation [5]: a) rough sigmoid; b) the errors.

- *first*, the computation of the classical sigmoid is quite complex

$$f(x,T) = \frac{1}{1+e^{-x/T}}, \tag{1}$$

  where $T$ is the "temperature" parameter (from thermodynamic-like functions);
- *second*, the input-output mapping done by the sigmoid is in fact a data compression which, for the required precision reported in the literature [7], roughly *halves the number of functions to be implemented.*

The classical digital solutions for implementing the sigmoid are either look-up tables [11], or truncation of Taylor series expansion. The second alternative can further be subdivided in: sum of steps [1,4,5], piece-wise linear [2,8,10,14,15], combination of the previous, or others [12,13].

The best known results for sum-of-steps approximations reported in literature have errors of ± 8.1% [4,5] (see figure 2), and ± 13.1% [1]. Previous known piece-wise linear approximation have ± 5.07% [8], or ± 4.89% [10], while the best known approximations are ± 2.45% [12] and ± 1.14% [6]. Deville's solution [6], while achieving the lowest error, is also the only one to use floating-point multiplications, which makes it far too complicated for a practical VLSI implementation.

## 2. Improved Precision by Piece-Wise Linearization

The starting point is the algorithm proposed by Pesulima [12]. We show how an exponential can be computed by a piece-wise approximating algorithm. By linking these two methods, a piece-wise approximation algorithm for the sigmoid which uses only very simple operations (additions/subtractions and shifts) can be built.

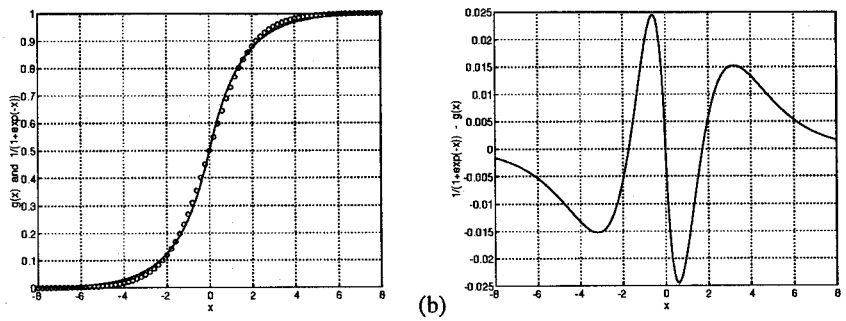
### 2.1 Fitting the sigmoid by two exponential

A good approximation of the classical sigmoid [12] is based on a two-exponential--fitting: one for the negative part of the real axis, and one for the positive part (the same result has been later thoroughly mathematically detailed in [2]):

$$g(x) = \begin{cases} 2^{x/T-1} = \dfrac{2^{x/T}}{2} = \dfrac{1}{2 \cdot 2^{|x|/T}} & \text{for } x \le 0 \\ 1 - \dfrac{1}{2 \cdot 2^{x/T}} & \text{for } x > 0 \end{cases}. \tag{2}$$

For $T = 1$, $g(x)$ can be seen in figure 3, where the classical sigmoid function has

ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 181-186




(a) (b)

**Fig. 3.** (a) classical sigmoid (circlrs) and $g(x)$; and (b) the difference between them $\delta(x)$.

been plotted for comparison. The differences are only $\delta \leq \pm 0.0245$ for this continuous approximation. If we now quantize $g(x)$ (i.e. sum of steps approximation of $g(x)$), the errors increase to more than $\pm$ 22%! Instead we can approximate $2^x$ by piece-wise linearization (all the other operations are subtractions or divisions by 2).

### 2.2 Piece-wise linear approximation of an exponential

A "log-like" algorithm which does a piece-wise linearization of the logarithmic function has been proposed [16] and also used [15]. If $x$ is represented in base 2 as:

$$x = x_{m-1} x_{m-2} \ldots x_0 \bullet x_{-1} x_{-2} \ldots x_{-k} = \sum_{i=m-1}^{i=-k} x_i \cdot 2^i, \tag{3}$$

then:

$$\log_2 x \cong \mathit{LOG\text{–}LIKE}(x) = (m-1) + 0 \bullet x_{m-2} \ldots x_0 x_{-1} x_{-2} \ldots x_{-k}, \tag{4}$$

where $m-1$ is the position of the MSB (most significant bit). The addition in the above equation is just a concatenation, as what we add is the fractional part of $x$ to an integer.

***Example:*** $\log_2 56 = \log_2 111000_2 \cong 5_{10} + 0.11000_2 = 101_2 + 0.11000_2 = 101.11000_2 = 5.75 \cong 5.803.$

The algorithm can be reversed to compute an "exp-like" function [3]. For the particular case when the base is 2:

$$2^x \cong \mathit{EXP\text{–}LIKE}(x) = 2^{\lfloor x \rfloor} + (x - \lfloor x \rfloor) \cdot 2^{\lfloor x \rfloor}, \tag{5}$$

where $\lfloor x \rfloor$ is the integer part of $x$, and $x - \lfloor x \rfloor$ is the fractional part of $x$, or:

$$2^x \cong \mathit{EXP\text{–}LIKE}(x) = 2^{x_{m-1} x_{m-2} \cdots x_0} + 0 \bullet x_{-1} x_{-2} \ldots x_{-k} \times 2^{x_{m-1} x_{m-2} \cdots x_0}, \tag{6}$$

from which it can be seen that the exponential is divided in:

- compute the exponent for the integer part (only shifts are needed), and
- add the fractional part properly shifted (the multiplicand being a power of 2) as a correction term.

***Example:*** $2^{5.25} = 2^{101.01_2} = 2^{101_2 + 0.01_2} \cong 2^{101_2}(1 + 0.01_2) = 100000_2 \times 1.01_2 = 101000_2 = 40 \cong 38.054.$

We will not go into details, but it is obvious that what this algorithm does is a linear approximation between integers (the correction terms varying linearly). The hardware requirements are minimal (shift register, subtracter, few additional logic gates [9]).

ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 181-186

## 3. Precision Analysis

A thorough analysis of the range where the differences between the classical sigmoid function and the proposed piece-wise linear approximation of the two-exponential--fitting lay, has been pursued. The results can be seen in figure 4. We have considered $k$, the number of bits from the fractional part of $x$ use for computing the correction term in eq. 3, as parameter. The value of $k$ has been varied between 0 and 5. If $k = 0$, the algorithm is doing no linearization at all as the correction term will always be 0 (it is in fact a sum-of-steps approximation). If $k$ grows, the set of curves come more and more closer to the smooth shape of the classical sigmoid. For $k = 1,2,3$ the staircase effect is still visible, but, already for $k = 4$ it starts disappearing, and the piece-wise linear approximation takes shape. Meanwhile the differences

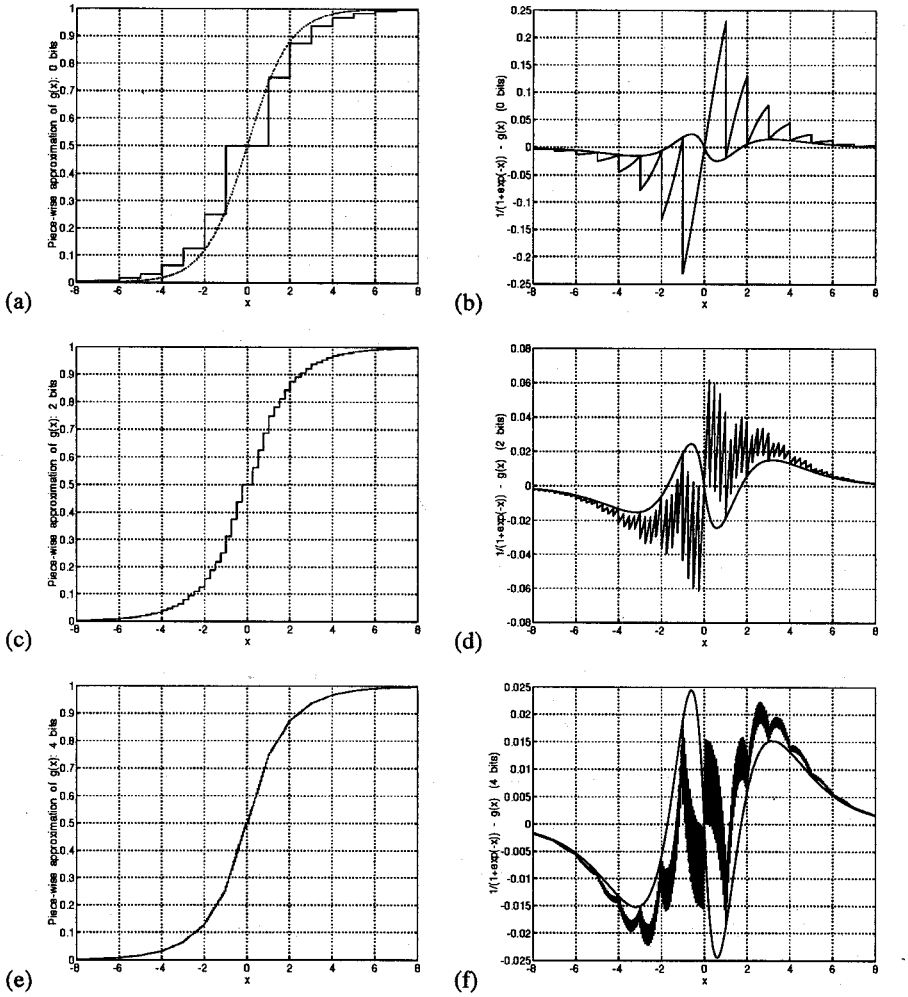



**Fig. 4.** Succesive approximations of the classical sigmoid by piece-wise linearization: (a) with no bits for the fractional part; (c) with two bits for the fractional part; (e) with four bits for the fractional part, and the differences $\delta(x)$ for these approximations: (b) with no bits for the fractional part; (d) with two bits for the fractional part; (f) with four bits for the fractional part.

ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 181-186

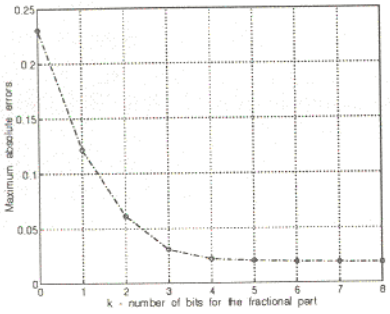



**Fig. 5.** The maximum absolute value of the differences between the classical sigmoid and the piece-wise linearization of the two-exponential-fitting, plotted versus $k$.

$\delta(x) = f(x) - g(x)$ between the classical sigmoid function and the piece-wise linearization of the two-exponential-fitting are more than tenfold reduced (from $\pm 22\%$ in figure 4b, to $\pm 2\%$ in figure 4f). The maximum absolute value of these differences remains almost constant for $k \geq 5$ (the minimum being 0.0184...), such that the error interval settles to $\delta \leq \pm 0.019$. Figure 5 presents this decrease of the maximum absolute differences versus $k$. Interesting to note is that the errors ($\pm 1.84\%$) have dropped under the ones for the continuous approximation ($\pm 2.45\%$), the explanation being that we are in a lucky case where the piece-wise approximation of $g(x)$ falls between $g(x)$ and $f(x)$.

This support the claim that 3 bits are needed in the fractional part to obtain a very good approximation for the classical sigmoid (see also figure 6, and compare it with figure 2).

## 4. Conclusions

Piece-wise linearization can be used for approximating sigmoid functions as increasing the accuracy over sum-of-steps approximations with very little hardware overhead. The data compression done by the sigmoid can thus be used to reduce the needed number of Boolean functions to be implemented, as compared to a direct digitization.

Further research should be concentrated to find similar efficient approximations for the derivative of the sigmoid (without using multiplication), needed for "learning".

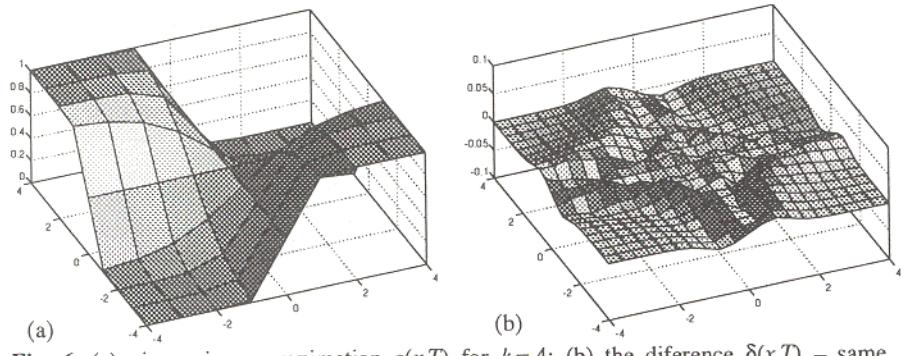



**Fig. 6.** (a) piece-wise approximation $g(x,T)$ for $k = 4$; (b) the diference $\delta(x,T)$ – same scale like in figure 1.

ESANN'1994 proceedings - European Symposium on Artificial Neural Networks
Brussels (Belgium), 20-21-22 April 1994, D-Facto public., ISBN 2-9600049-1-4, pp. 181-186

## References

1. C. Alippi: Weight Representation and Network Complexity Reductions in the Digital VLSI Implementation of Neural Nets. Res. Note RN/91/22, Dept. CS, Univ. College London, 1-20 (1991).
2. C. Alippi, G. Storti-Gajani: Simple Approximation of Sigmoidal Functions: Realistic Design of Digital Neural Networks Capable of Learning. Proc. IEEE Intl. Symp. on Circuits and Systems ISCAS'91, Singapore, 1505-1508 (1991).
3. V. Beiu, M. Ionescu, E. Pasol, L. Zuzu: Adaptive Multiplexing Algorithm and Its Possible VLSI Implementation. Proc. MELECON'87, Rome, 780-785 (1987).
4. V. Beiu, J.A. Peperstraete, R. Lauwereins: Using Threshold Gates to Implement Sigmoid Nonlinearity. Proc. ICANN'92, Brighton, Vol. **2**, 1447-1450 (1992).
5. V. Beiu, J.A. Peperstraete, J. Vandewalle, R. Lauwereins: Close Approximations of Sigmoid Functions by Sum of Steps for VLSI Implementation of Neural Networks. Scientific Annals ("Al.I. Cuza" Univ., Iassy), accepted for publication (1994).
6. Y. Deville: A Neural Implementation of Complex Activation Functions for Digital VLSI Neural Networks. Microelectronic J., **24**(3), 259-262 (1993).
7. J.L. Holt, J.-N. Hwang: Finite Precision Error Analysis of Neural Network Hardware Implementations. IEEE Trans. on Comp., **C-42**(3), 281-290 (1993).
8. A. Krikelis. A Novel Massively Parallel Associative Processing Architecture for the Implementation of Artificial Neural Networks. Proc. Intl. Conf. ASSP'91, Toronto, Vol. **2**, 1057-1060 (1991).
9. P. Murtagh, A.C. Tsoi: Implementation Issues of Sigmoid Function and Its Derivative for Digital Neural Networks. IEE Proceedings-E, **139**(3), 207-214 (1992).
10. D.J. Myers, R.A. Hutchinson: Efficient Implementation of Piecewise Linear Activation Functions for Digital VLSI Neural Networks. Electronics Letters, **25**(24), 1662-1663 (1989).
11. M.E. Nigri, P. Treleaven, M. Vellasco: Silicon Compilation of Neural Networks. Proc. CompEuro'91, Bologna, 541-546, (1991).
12. E.E. Pesulima, A.S. Pandya, R. Shankar: Digital Implementation Issues of Stochastic Neural Networks. Proc. IJCNN'90, Washington, Vol. **2**, 187-190 (1990).
13. G. Saucier, J. Ouali: Silicon Compiler for Neuron ASICs. Proc. IJCNN'90, Washington, Vol. **2**, 557-561 (1990).
14. A. Siggelkow, J.A.G. Nijhuis, S. Neußer, B. Spaanenburg: Influence of Hardware Characteristics on the Performance of a Neural System. Proc. ICANN'91, Helsinki, 697-702 (1991).
15. B. Spaanenburg, B. Hoefflinger, S. Neußer, J.A.G. Nijhuis, A. Siggelkow: A Multiplier-Less Digital Neural Network. Proc. MicroNeuro'91, Münich, 281-289 (1991).
16. N. Weste, D.J. Burr, B.D. Ackland: Dynamic Time Warp Pattern Matching Using an Integrated Multiprocessing Array. IEEE Trans. on Comp., **C-32**(8), 731-744 (1983).