UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Posgrados

Design of CMOS Circuits for the Implementation of the Activation Function in Artificial Neural Networks

Proyecto de Investigación y Desarrollo

Jessica Tatiana Moposita Estrella

Felice Crupi Ph.D
Director de Trabajo de Titulación

Trabajo de titulación de posgrado presentado como requisito
para la obtención del título de Maestría en Nanoelectrónica

Quito, 11 de diciembre 2020

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

COLEGIO DE POSGRADOS

HOJA DE APROBACIÓN DE TRABAJO DE TITULACIÓN

Design of CMOS Circuits for the Implementation of the Activation Function in Artificial Neural Networks

Jessica Tatiana Moposita Estrella

Nombre del Director del Programa:	Omar Aguirre
Título académico:	Doctor of Philosophy
Director del programa de:	Maestría en Nanoelectrónica

Nombre del Decano del colegio Académico:	César Zambrano
Título académico:	Doctor of Philosophy
Decano del Colegio:	Colegio de Ciencias e Ingeniería

Nombre del Decano del Colegio de Posgrados:	Hugo Burgos
Título académico:	Doctor of Philosophy

Quito, diciembre 2020

© DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombre del estudiante:	Jessica Tatiana Moposita Estrella
Código de estudiante:	00207585
C.I.:	1725576860
Lugar y fecha:	Quito, 11 de diciembre del 2020

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following graduation project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

AGRADECIMIENTOS

Me gustaría agradecer a los miembros de mi comité de tesis por su ayuda en la preparación de este trabajo. A mis docentes de la Universidad San Francisco de Quito por transmitir todos sus conocimientos y experiencias. A la Universidad San Francisco de Quito por haberme otorgado una beca permitiendo que extienda mis estudios y me desarrolle más a nivel académico. Y un especial agradecimiento a mi familia que gracias a su apoyo incondicional permitieron que este logro sea posible.

Abstract

In recent years there has been a deep interest in artificial neural networks (ANNs), computing systems inspired by biological neural networks. An ANN involves a complex network interconnected by nodes called artificial neurons, it sums N weighted inputs and passes the result through a non-linear transfer function. Some approaches are used to develop a neural network based on CMOS devices.

This thesis presents the design and analysis of sigmoid and softmax function denominated as an activation function. Proposed circuits pretend to be calibrated in order to obtain low power consumption as possible. The performance of these simulations is developed by using 0.18 um CMOS technology parameters. The circuit operates at 0.5 V and 1 V supply voltage. For each configuration it is aimed to select the most optimal option in terms of energy.

Key words: ANN, artificial neurons, transfer function, activation function, Sigmoid, Softmax, CMOS technology.

RESUMEN

En los últimos años ha existido un profundo interés en las redes neuronales artificiales (ANN siglas en inglés), sistemas informáticos inspirados en las redes neuronales biológicas. Una red neuronal artificial involucra una red compleja interconectada por nodos llamados neuronas artificiales, suma N entradas ponderadas y pasa el resultado a través de una función de transferencia no lineal. Se utilizan algunos enfoques para desarrollar una red neuronal basada en dispositivos CMOS. Esta tesis presenta el diseño y análisis de la función Sigmoid y Softmax denominada función de activación. Los circuitos propuestos pretenden ser calibrados para obtener el menor consumo de energía posible. El rendimiento de estas simulaciones se desarrolla utilizando parámetros de tecnología CMOS de 0,18 µm. El circuito funciona con una tensión de alimentación de 0,5 V y 1 V. Para cada configuración se pretende seleccionar la opción con mejor optimización en términos de energía.

Palabras clave: ANN, neuronas artificiales, función de transferencia, función de activación, Sigmoid, Softmax, tecnología CMOS.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION AND BACKGROUND

Several neural network based computing models have been figured out in the last few years in order to develop hardware to be capable to perform human-like cognitive computing [1] reaching great success recently consequently, the hardware implementation of artificial intelligence is still essential in many applications[2]. Although the software version is essential where many GPUs are used, the hardware implementation is required for avoid bulky systems and high power consumption [2].

Artificial neural net models also known as connectionist models, or parallel distributed processing models, bring for to accomplish good performance via dense interconnection of simple computational elements, artificial neural structure is based on our knowledge of biological nervous system [3].

Many problems can be resolved by ANNs in the areas of signal processing, surface classification, object detection, electronic nose, pattern recognition, medical applications, navigation, and control. Nevertheless, in this field software simulations are not only developed but also analysis and studying of capabilities about ANN models and computational algorithms based on neural networks. In addition, hardware implementations are needing because of its advantage in inherent parallelism of neural networks [4].

1.1 Neural Networks

Exists some problems that cannot be formulated as an algorithm because depend in subtle factors in which our brain can calculate. The main goal in this situation is that computers do not have the capability to learn, even though computers have processing units and memory and perform the most complex numerical calculations in a very short time, they are not adaptive. Theoretically the computer should be more powerful than our brain, computers contains transistors with a switching time of 10^{-9} seconds. The brain contains 1011 neurons, but these only have a switching time of about 10^{-3} seconds.

The largest part of the brain is working continuously, while the largest part of the computer is only passive data storage. Meanwhile computer is unchanging, the brain as a biological neural network can reorganize itself during its ”lifespan” and therefore is able to learn, to compensate errors and so forth.

Study of artificial neural networks is motivated by the successfully working biological systems, which in comparison to the overall system consist of very simple but numerous nerve cells that work massively in parallel and have the capability to learn. Obtaining an explicitly program for a neural network is not necessary. For instance, it can learn from training samples or by means of encouragement [5].

1.2 Biological neural networks

The basic and primordial nerve cell, called a neuron, is the fundamental building block of the biological neural network [6]. The vertebrate nervous system is the burdened to process the entire information talking about central nervous system which consist of brain and spinal cord, focus on brain it has four areas, cerebrum, thalamus, cerebellum and truncus cerebri.

The cerebrum is responsible for abstract thinking processes. The cerebellum controls and coordinates motor functions. the thalamus decides which part of the information is transferred to the cerebrum also hypothalamus controls several processes within the body. All this topic is about the level of brain areas.

Now about the cellular level of the body there is the level of neurons, neuron is a switch with information input and output.

As the Figure 1.1 shows, in the neuron the electrical information way starts with dendrites which receive the information by special connections, the synapses. Synapses are incoming signals transferred from other neurons or cells to a neuron by special connections [5]. In other words, neurons are the fundamental computing units of the systems that connect to each other to external stimuli through programmable connections[3],[1], [7].

Soma are the dendrites branches and obtain electrical signals from numerous sources to later be transferred to the nucleus, when dendrites exceeds a certain value also known as threshold value activates an electrical pulse for later be transmitted to the neurons connected to the current one, the neuron will fire off an impulsive via its axon. The axon can be viewed as a connection wire [8].

When neuron accept inputs, that have been post-processed in the synaptic, they are condensed or accumulated to one single pulse. If the neuron is not stimulated enough, itself does not emit a pulse, consequently the output is nonlinear or not proportional to all inputs.

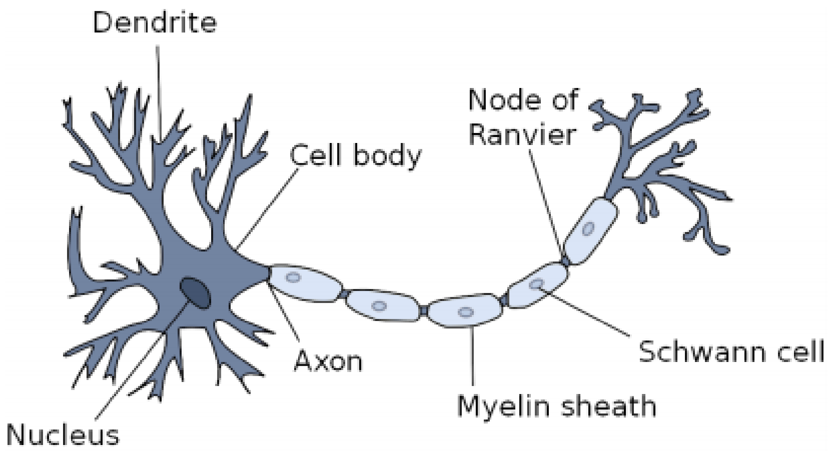


Figure 1.1: Biological neuron with their main components, weights of inputs are determined through dendritic biochemistry changes and synapse modification [8]. Image credit: [5]

This building blocks seem simple but the system is relatively complex, due to both the huge amount of neurons and the number of interconnection between them, obviously with the fact that the neurons function autonomously and in parallel. The great number of connections is essential because learning process depends on the growth of new connections that is why synapses role is important [7].

1.3 Artificial Neural Networks

“Artificial Neural Network is a computing system made up of a number of simple, highly interconnected processing elements which process information by their dynamic state response to external inputs.” - Robert Hecht-Nielsen.

1.3.1 Layers of ANN

Artificial Neural Networks have input layer, output layer and hidden layers, depending of the application can be more than one.

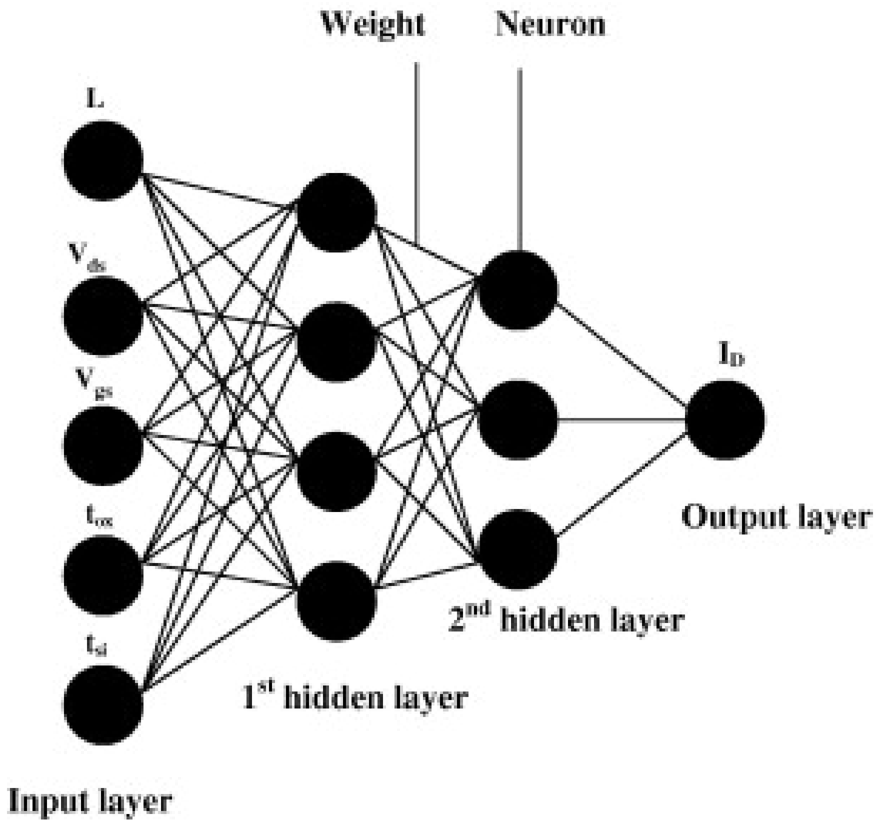


Figure 1.2: Multi-layered feedforward artificial neural

In Figure 1.2 is possible identify the layers above mentioned, the leftmost layer in this network is called the input layer, and the neurons within the layer are called input neurons. The rightmost or output layer holds the output neurons. The middle layer is called a hidden layer since the neurons in this layer are neither inputs nor outputs. The term “hidden” has a special meaning, nothing more than “not an input or an output” [9].

1.3.2 Properties of ANN

Some important properties are high level of parallelism, possibility of asynchronous processing, multidirectional execution, real-time adaptability, robustness with respect to damages and to missing data, ability to learn and automatic generalization, no need for additional software, no need for a fixed configuration, no difference between data and addresses, and, last but not least, well developed mathematical foundation [10].

Parallelism, a high degree of parallelism in ANN is affected by the fact that each neuron operates independently of other neurons.

1.3.3 Topology

Topology is one of the criteria in artificial neural networks, and can be used without layers, two-layered or multi-layered feedforward.

- ANN without layers, each neuron is connected to every other neuron in both directions.
- Two-layered feedforward ANN, as shows Figure 1.3 each input neuron is connected with each output neuron with a directed connection.
- Multi-layered feedforward, adding one or more additional (hidden) layers of neurons between the input and the output layer we get a multi-layered feedforward ANN

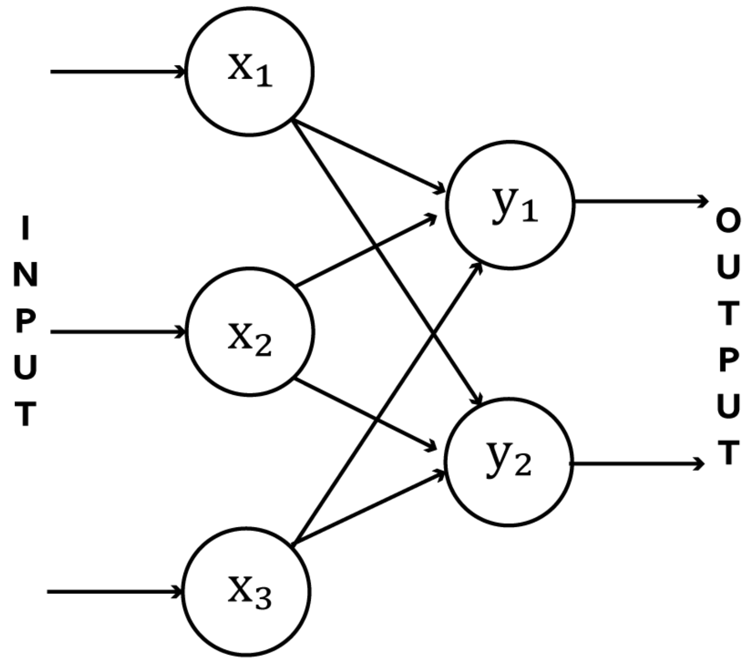


Figure 1.3: A two-layered feedforward ANN with three input and two output neurons

Multi-layer perceptron

The most common feed-forward network, it has three layers: an input layer, an output layer and a hidden layer [11]. Figure 1.2 shows multi-layered feedforward (MLF) artificial neural.

The multilayered perceptron uses the generalized delta learning rule (backpropagation of errors) and in principle is able to learn and calculate any (nonlinear) function [10].

Input and output models are connected by a set of neurons structured in hidden layers, the layers in these networks are linked by communication connections that are associated with weights which determine the information passing through them [12].

1.3.4 ANN Training

Single layer neural network (or perceptrons) can be trained using either the Perceptron training rule or the Adaline rule. The first rule, has an acceptable behaviour when training samples are linearly separable and updates weights based on the error in the threshold perceptron output. The second rule Works well even when the training samples are not linearly separable, updates weights based on the error in the non-threshold linear combination of outputs, affords a base for backpropagation algorithm, which can learn networks with many interconnected units.

Perceptron training rule

A perceptron is a computational unit that calculates the output based on weighted input parameters. The process start by initializing the weights from 0 or small random numbers, for each training sample $x(i)$ compute the output value and update the weights.

The perceptron receives the inputs of sample x and combines them with the weights w to compute the net input.

The net input is then passed on to the threshold function, which generates a binary output -1 or +1: the predicted class label of the sample. During the learning phase, this output is used to calculate the error of the prediction and update the weights.

Adaline Rule (Widrow-Hoff Rule)

In Adaline, the weights are updated based on a linear activation function. The linear activation function y(x) is the identity function of the net input, is used for learning the weights and a threshold function is used to make the final prediction, which is similar to the unit step function.

Gradient descent rule, often used as part of Adaline algorithm, it is used to find the weights that minimize the cost function. The main idea behind gradient descent is to go down the hill of cost function until a local or global minimum point is reached.

1.3.5 Learning Rule

In MLF networks the proper setting of the weights is adjusted by supervised training. The weights are optimized by means of several examples input models together with their associated required output pattern. During the training session the weights are modified according to the learning rule [13]. The most common learning algorithm is the back-propagation learning rule.

Back Propagation Algorithm

Back propagation networks are usually layered, with each layer fully connected to the layers below and above. Data input propagates forward through of each internal layer from the input layer toward the output layer of the processing units, delivering the network response. Correction mechanism starts from the output units and back propagates through each internal layer to the input, the process continue until RMS error becomes minimum and weights reaches their final state [14]. Back propagation algorithm consists of two propagation, namely forward and backward propagation.

The back-propagation algorithm involves a forward-propagating step followed by back-propagation. Both the forward and the back propagation steps are carried out for each pattern presentation during training. In each successive layer, every processing unit sums its inputs and then applies a sigmoid function to compute its output. The output layer of units then produces the output of the network [14].

Steps for back-propagation algorithm;

1. Initialize weights to random values in a range around 0
2. Apply the k-th input training pattern
3. Forward propagate the signal till the output of the output layer,
4. Compute errors for the output layer.
5. Back propagate the error to update weights and computer errors for previous layers.
6. loop to step 2 till all training data has been cycled once (one epoch)
7. check if the total error is acceptable then terminate if not initiate another training epoch by going to step 2.

In the back-propagation approach the weight adaption is made in the path that minimizes the error. The back-propagation algorithm is thus necessarily a gradient based optimization method [13]. Therefore, the gradient of the error as a function of the weights must be calculated how it is shown in figure Figure 1.4.

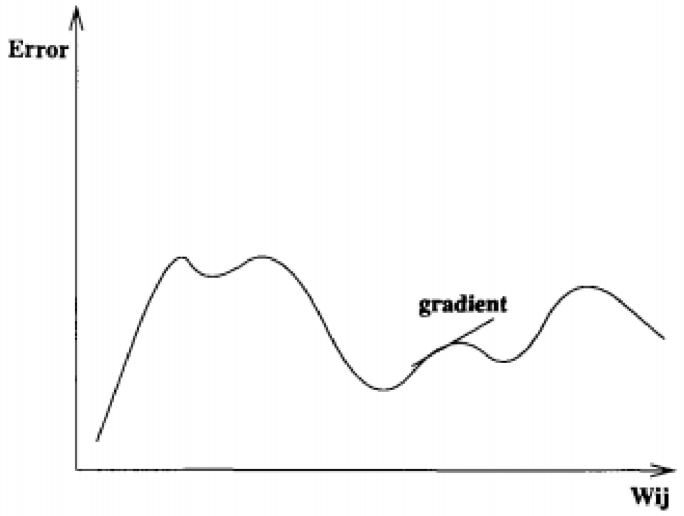


Figure 1.4: An example of an error surface of an MLF as function of one weight value. The gradient determines the change of the weight in the next iterations [13]

1.4 Structure of a Neural Network

A neuron model receives a vectorial input with components x_i for later be multiplied by the appropriate weights w_i that are accumulated and passing the result through a non-linear transfer function [1], [5].

Perceptron which is historically possibly the earliest artificial neuron that was proposed by Roseblatt in 1985 is also the basic block of nearly all Artifitial Neural Networks [8], Figure 1.5 seems like a perceptron which takes several binary inputs x_1, x_2, ... and produces a single binary output [9], further above mention that perceptron can be two-layered or multi-layered depending on the topology used.

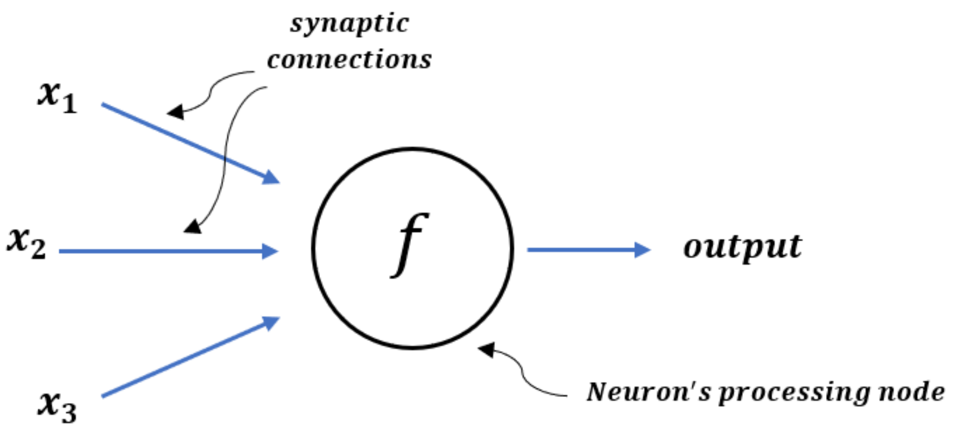


Figure 1.5: Basic model of an structure artificial neuron

Weights denoted as w_1, w_2, ... in the artificial model are related to the synaptic connections in biological neurons. Managing elements are typically developed by the equation which characterize the model of an artificial neuron as follows [1], [15],[16]:

$$y = f\left(\sum_{i=1}^{N} w_i x_i\right) \tag{1.1}$$

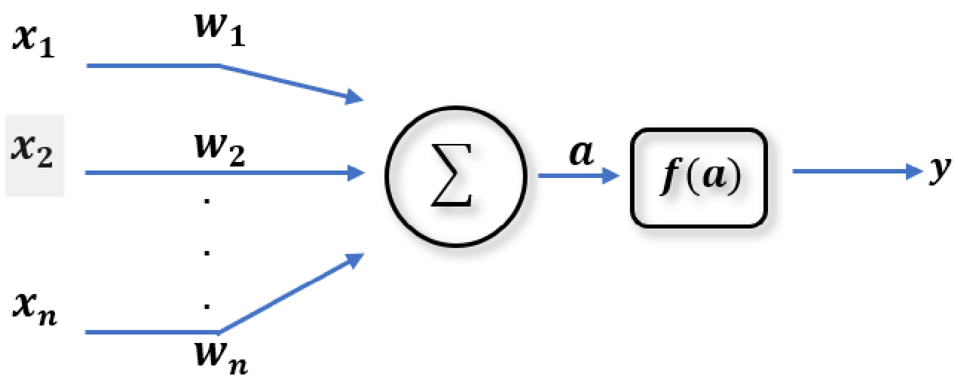


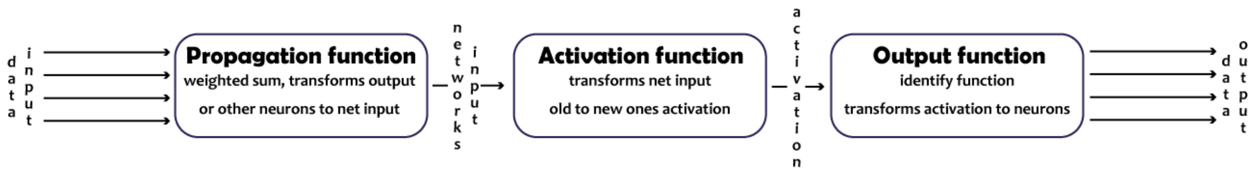
Figure 1.6: Model of an artificial neuron

The weighted sum is called the activation function, and the threshold function f (also can be a sigmoidal, hyperbolic tangent or radial basis function [11]) is called the output function. An output function can return more than two different values. An activation and an output function together are called the combination function [10], Figure 1.6 shows the model of an artificial neuron, the synapse is excitatory when the weight is positive, but if it is negative the synapses is inhibitory, this happens because artificial neurons can get either excitatory or inhibitory inputs. Excitatory inputs cause summing process by adding the next neuron while the inhibitory inputs cause it to subtract [17].

Finally, is a fact that neural network consist in processing units called neurons multiplied with weighted connections between those neurons.

1.4.1 Propagation function

Figure 1.7 shows data processing of a neuron, propagation functions receive output signals of other neurons and transform them in order to be processed by the activation function that is why the network input is the result of the propagation function.The calculation starts with the first hidden layer and consequently through the other layers towards the output layer [10].



data input
Propagation function
weighted sum, transforms output or other neurons to net input
networks input
Activation function
transforms net input old to new ones activation
activation
Output function
identify function
transforms activation to neurons
data output

Figure 1.7: Data processing of a neuron,image credit:[5]

1.4.2 Activation Function

A combination function involves two parts: an activation function and output function.

Activation functions are functions used in neural networks to calculates the weighted sum of input and biases, it is used to determine if a neuron can be fired or not [18], it correlates the output of a neuron to its input based on the neuron's input activity level [12]. It uses and manipulates the presented data through some gradient processing usually gradient descent and subsequently generate an output for the neural network, that holds the parameters in the data [18]. Activation function can be either linear or non-linear depending on the function it represents, and are used to control the outputs of out neural networks[18], the most frequently used activation function is the linear activation function, implemented as a weighted sum.

Non-linear functions are needed in neural networks. It is required of a neural network to learn, characterize and process any data and any arbitrary complex function which maps the inputs to the outputs. Neural Networks are also known as Universal Function Approximators which, this functions have been developed to calculate and learn any function given to them. Any possible process can be represented as a functional computation in Neural Networks [19].

Some of the frequently non-linear functions used consist of: the threshold, piecewise linear, sigmoid, tangent hyperbolic, and the Gaussian function [12], [20],[15], [21], [22] which also are used in analogue neural networks.

Exist another important functions like step and saturated, the step function is called hard-limiting transfer function because of the binary output states. The saturated linear, logistic sigmoid and hyperbolic tangent functions are softlimiting transfer functions because of the constant neuron output states. [1].

Observe from Equation 1.1 that the neuron as a processing node develop into the operation of summation of its weighted inputs [6]. Subsequently, because of its activation function, it develops nonlinear operation $f(x)$;

$$y = \frac{1}{1 + e^{-x}} \tag{1.2}$$

Sigmoid Activation Function

The most common activation function is the sigmoid function which is continuously differentiable [8]. This function is used for predicting probability based output and has been applied successfully in binary classification problems, it is also known as squashing function in some literature [18]. .

Taking the Equation 1.2 as a reference, the sigmoid function is developed as;

$$y = \frac{1}{1 + e^{-ax}} \tag{1.3}$$

From Equation 1.3 such that for;

$$x \mapsto \begin{cases} x \to -\infty & \Leftrightarrow \quad y \to 0 \\ x \to 0 & \Leftrightarrow \quad y = 0.5 \\ x \to \infty & \Leftrightarrow \quad y \to 1 \end{cases} \tag{1.4}$$

.

Figure 1.8 shows sigmoid function, observe that follows the conditions refers in Equation 3.1, observe that the slope of function is given by exponential a.

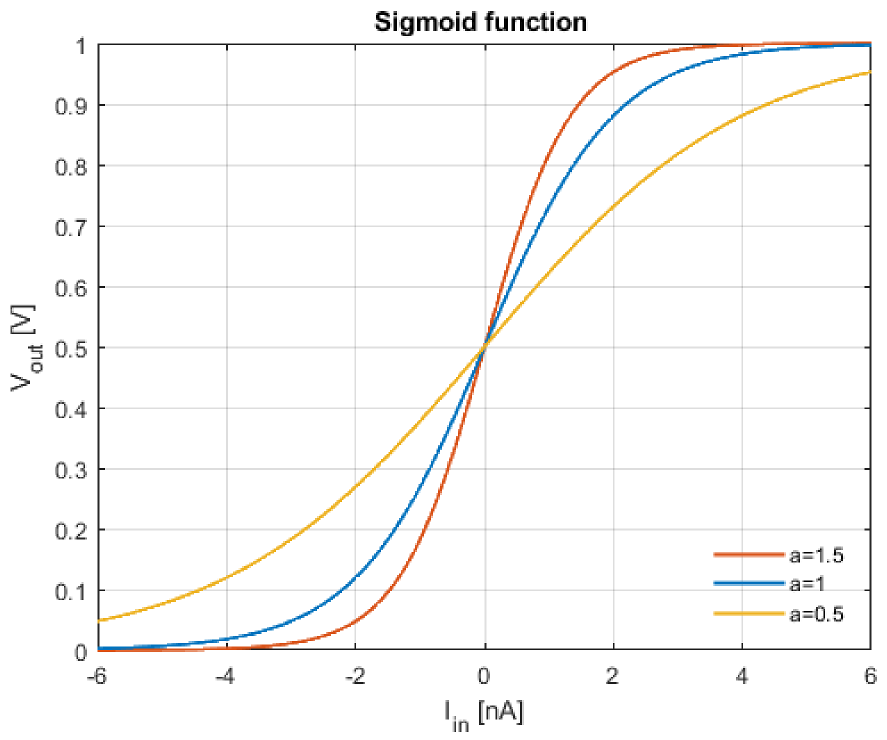


Figure 1.8: Sigmoid activation function

The sigmoid function performs a sort o f “soft” threshold that is rounded (and differentiable) compared to step function [14].

Hyperbolic Tangent Function

There is another popular active function whose shape is rather similar to Sigmoid function expressed like;

$$y = \tanh(ax) \tag{1.5}$$

Like sigmoid in this case the considerations are the following and Figure 1.9 shows this conditions ;

$$x \mapsto \begin{cases} x \to -\infty & \Leftrightarrow \quad y \to -1 \\ x \to 0 & \Leftrightarrow \quad y = 0 \\ x \to \infty & \Leftrightarrow \quad y \to 1 \end{cases} \tag{1.6}$$

.

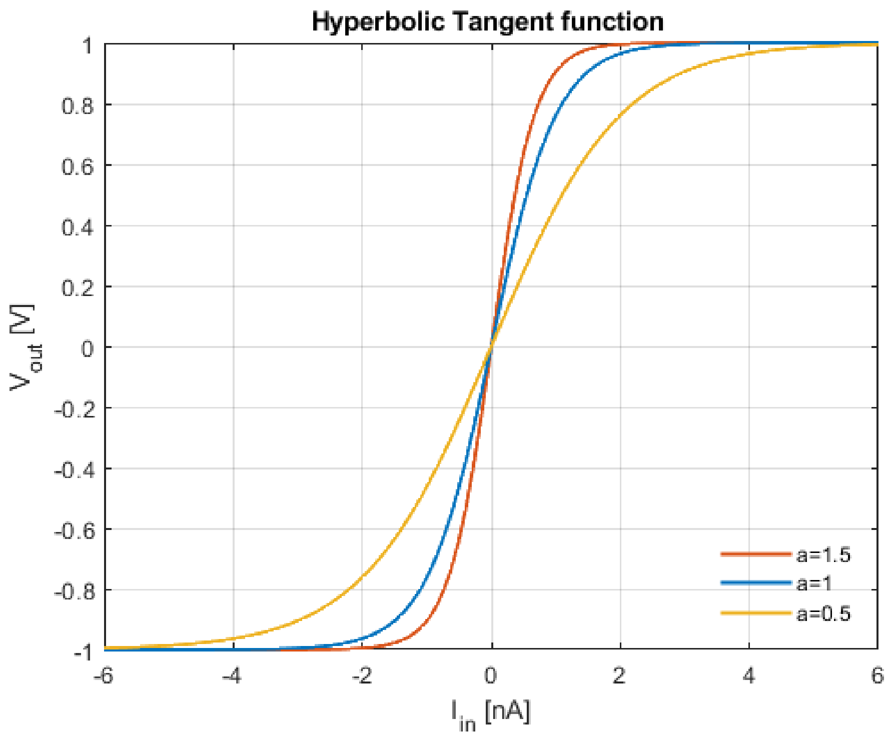


Figure 1.9: Hyperbolic Tangent activation function

S shaped curves are considering Sigmoid and Tanh, where sigmoid is limited between 0 and 1 and Tanh is limited between -1 and 1. Here, all functions with S shaped curve between 0 and 1 are called sigmoid-like functions and with S shaped curved between -1 and 1 are called Tanh-like functions [23].

Threshold Function

The simplest active function is hard-switch limits threshold element [8] represented as Equation 1.7 and shown in Figure 1.10;

$$y = \begin{cases} 1 \quad for \quad x \geq 0 \\ 0 \quad for \quad x < 0 \end{cases} \tag{1.7}$$

.

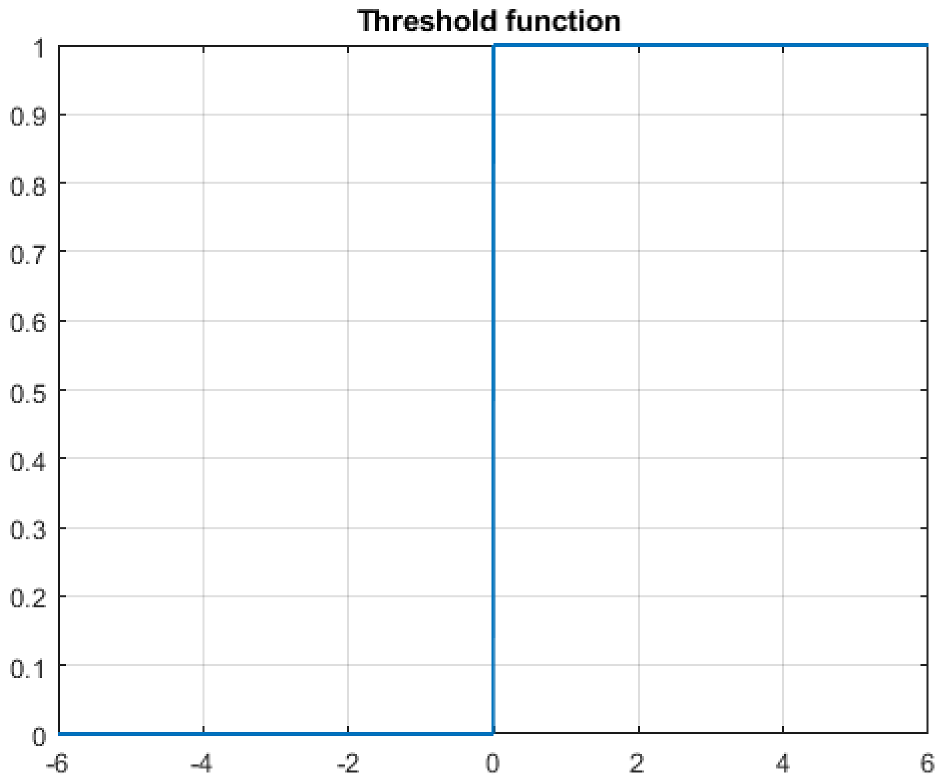


Figure 1.10: Threshold activation function

Softmax Function

The Softmax function is another type of activation function used in neural computing [18]. The softmax function is used in machine learning to give a probabilistic interpretation to outputs of classification models [24]. As the softmax function plays an important role in the machine learning models the several approximations have been proposed. Softmax function, also called softargmax or normalized exponential function, considering a mathematical function that allows an input vector of K real numbers and regulates it into a probability distribution comprising of K probabilities proportional to the exponentials of the input numbers [25], [26]. The Softmax function produces an output which is a range of values between 0 and 1 [19], with the sum of the probabilities been equal to 1 [18]. The main difference between the Sigmoid and Softmax AF is that the Sigmoid is used in binary classification while the Softmax is used for multivariate classification tasks [18],[19].

The softmax function is a sigmoid function normalized respect to all the input of the output level, the softmax function is defined by the Equation 1.8;

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{M} e^{x_j}} \qquad for \quad i = 1, ..., M. \tag{1.8}$$

.

After applying Softmax, each element will be in the range of 0 to 1, and the elements will add up to 1 how it is shown in Figure 1.11.

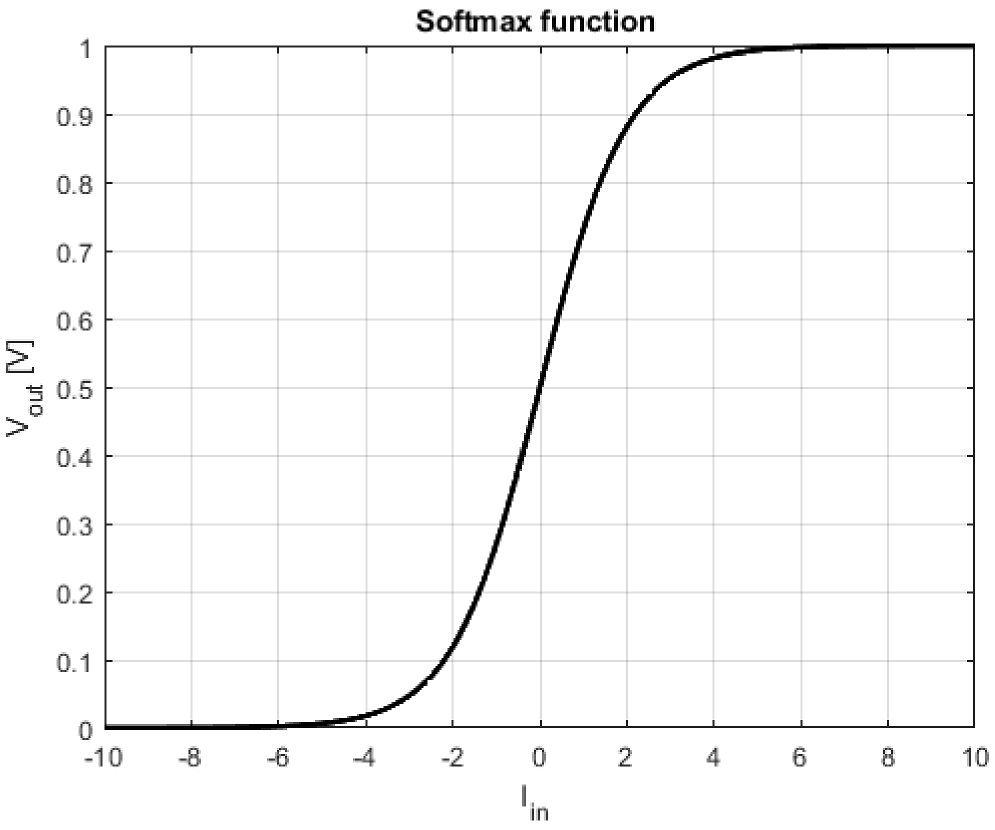


Figure 1.11: Softmax activation function

1.5 State of Art

Research in ANNs has resulted in a variety of models and learning algorithms. There are several ANN models considering paradigms. The differences between ANN models are their topology and their way of learning and recalling information. The most commonly used and widely applied model is the multi-layer perceptron (MLP) [27]. There are two main phases in the operation of any

ANN, learning and recall phase.

Learning phase, the network is qualified so that application of a set of inputs produces the preferred set of outputs. Learning is accomplished by sequentially applying input vectors while adjusting network weights according to a predetermined process.

Recall phase: Deals with how the trained network processes a stimulus presented to its input layer and produces response at its output layer. Depending on the network connections, the recalling process can be a feed-forward or a feedback. A commonly used supervised learning algorithm is the Back Propagation algorithm [14], [27]

1.5.1 Recent Advances

An interesting development with a low-power analogue complementary metal oxide semiconductor (CMOS) artificial neuron circuit is presented in [15] called Ghomi Solution, the analogue neuron is composed of multiplier and programmable activation function circuits. It is a four-quadrant multiplier and has a single ended current, its programmability of activation function improves the neural network design process while enables the proposed neuron circuit to be used properly in learning algorithms such as back propagation.

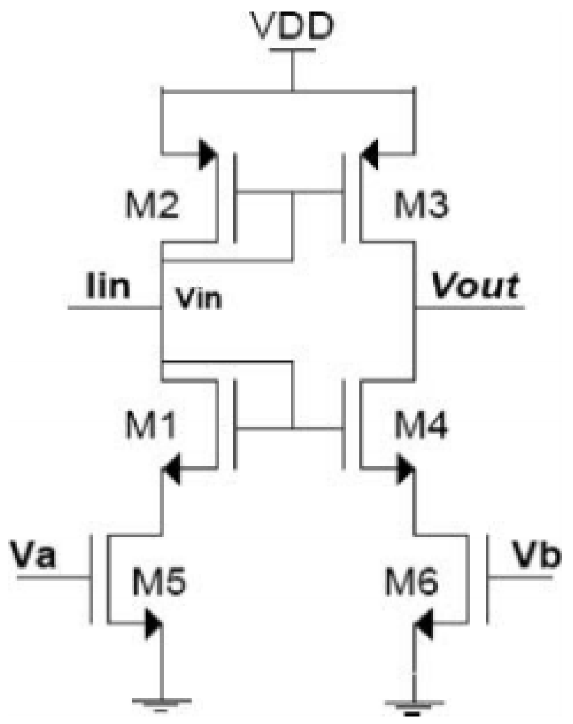


Figure 1.12: Modified programmable sigmoid activation function generator circuit

Figure 1.12 implementation lead to a significant reduction in power consumption and area. So, the power consumption for neuron circuit is only 15 μW at a 1V supply voltage. The performance uses 0.18 mm CMOS technology parameters[15].

Another recently and similiar work is proposed by [28] it is a new NMOS/PMOS design proposed for realizing the sigmoid function as the activation function. Transistors are biased using only one biasing voltage, it operates in triode and saturation regions providing an accurate approximation of the sigmoid function. In the neuron presented in Figure 1.14, the input to the neuron is the summation current from the synapses. Depending on the value of the input current, a voltage is generated at the output node. This neuron, named a resistive-type neuron, has a resistive-like nature. [28]

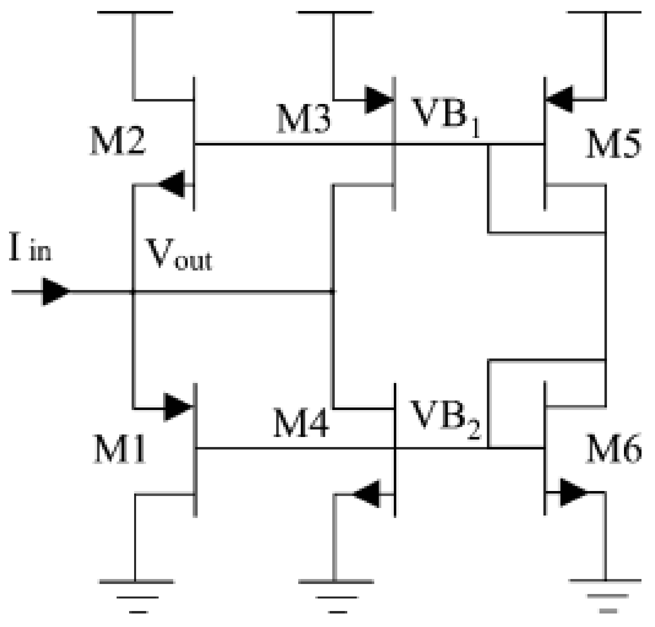


Figure 1.13: Schematic of the proposed resistive-type sigmoidal neuron

As the paper mention, this solution use only one biasing voltage, it uses transistors in both triode and saturation region to get the sigmoid function. The proposed is more accurately compared to the previous design specially in linear region, while it consumes less area. It can be used in analog implementation of activation functions for both pure analog neural networks and HNNs.

A Design of a passive resistive-type neuron is proposed in [23] to generate the hyperbolic tangent function as the activation function. The proposed resistive-type neuron has the advantage of not needing any biasing voltage and therefore its power consumption is low. It proposed neuron is applied in a large neural network.

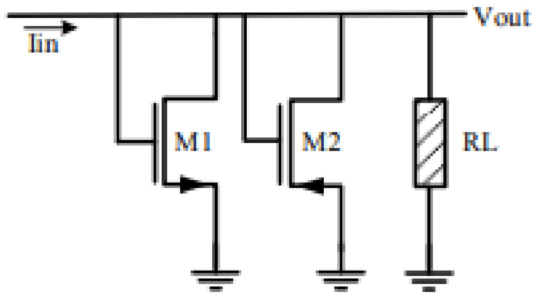


Figure 1.14: Schematic proposed passive resistive-type neuron

The proposed neuron is a passive circuit which consists of two PMOS/NMOS transistors and a small resistor.The standby power is zero and the operation power consumption between -200 μA and 200 μA is 62.5 μW. The area estimation is based on the full costumed layout design of the proposed neuron in 180 nm which it is about 39 μm^2. [23]

A proposed solution is also used as a reference for this work, the proposal serves as a guide to find a better behavior, this solution is detailed in the third chapter for a better understanding of the applied neural network.

CHAPTER 2
SIMULATION ENVIROMENTAL

All the simulations are done in LTspice®and confirmed in Cadence®–Virtuoso®. We will show characteristics, parameters and considerations in order to simulate the best circuit considering the main approach about the active function that is related.

SPICE (Simulation Program with Integrated Circuits Emphasis, often written as Spice) environment turn into an analysis platform of many design software digital and analog alike.

SPICE is the most popular software code because of its elements can be used automatically without user interference. The program is managed from the design platform GUI while the user operates the graphical elements. Circuit simulation programs, of which SPICE and derivatives are the most renowned, take a text net list describing transistors, resistors, capacitors, inductors, diodes, and others. The general equations produced are nonlinear, differential, algebraic equations which are solved using implicit integration methods, Newton's method and sparse matrix techniques.

The LTspice platform is very user-friendly and allows for extensive circuit analysis. Each of the elements and functions added can be configured, as well as the schematic preferences and waveforms. Finally we can run and visualize the behavior of the circuit at any point of operation.

There are different types of analysis and simulations that can be configured to obtain the desired results.

2.1 Transient Analysis

A time domain transient analysis allows to plot parameters such as voltage or current against time. If you are looking at an output you can see the behavior over a specified length of time.

2.2 AC Analysis

Ac analysis provides the frequency response of your circuit. The output waveform is the bode plot that shows the amplitude and phase across a specified frequency range, and the options for this AC analysis are sufficient. It is possible to plot the frequency response as a bode plot, in Cartesian coordinate plane with the real and imaginary axis and even as a Nyquist graph.

2.3 DC Sweep and Transfer Analysis

A DC Sweep is a type of simulation that lets to vary voltage or current of a specified device while DC Transfer function calculates the low frequency gain and the input and output resistances of the circuit.

Finally there are several steps to create your own model in LTspice. A model consists of a subcircuit and a symbol.

There are two ways to examine a circuit in LTspice by changing the value for a particular parameter: that means change manually by entering each value and then resimulate the circuit to view the response, or use the .step command to sweep across a range of values in a single simulation run. This command is helpful since in particular this thesis pretends to perform simulations in search of the best behavior, so a data sweep allows a better selection and processing of information. Particularly for the development of this thesis we used a particular model, 0.18 nm technology for the CMOS.

CHAPTER 3

PROPOSED SOLUTIONS

As mentioned in the previous chapter, Artificial Neural Networks (ANNs) have been develop successfully and reached great success recently, the merits of the hardware-based neural networks over those of the software-based ones are processing speed and power consumption [2].

This chapter shows the method and simulation methodologies for the analysis of the active functions both sigmoid and softmax. It is mentioned the approach used for knowing the model based on artificial neural network behavior, which is a requirement to simulate active functions circuits. Furthermore, a brief description of the simulation structure is presented. Finally, is shown comparisons between both functions.

3.0.1 Sigmoid Function

In the previous chapter had been mentioned that activation function produces the output of each layer in the feed forward neural networks according to the value of its input. Activations functions such as tangent hyperbolic and sigmoid are nonlinear functions and generate an S shaped curve. But Sigmoid is used more due to the simplest hardware implementation of his derivate tangent hyperbolic.

It has been repeatedly mentioned that sigmoid is considered as a nonlinear function and the output range is from 0 to 1, sigmoid function is defined as Equation 1.3. Considering that analog activation functions are generally less in area and power consumption than digital implementations[28], [29], [30]. Analog can be susceptible to mismatch and circuit variations.

From an analytical point of view the Sigmoid function can be divided into 5 main regions as shown below.

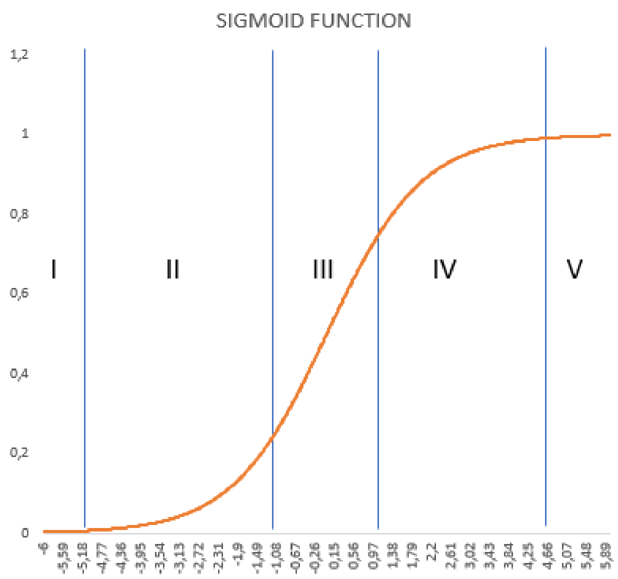


Figure 3.1: Approximation sigmoid function

$$y \mapsto \begin{cases} y = 0 & for \quad x < -6 \\ y = \frac{e^x}{1+e^x} & for \quad -6 < x < -0.6 \\ y = \frac{1}{2} + \frac{x}{4} & for \quad -0.6 < x < 0.6 \\ y = \frac{e^x}{1+e^x} & for \quad 0.6 < x < 6 \\ y = 1 & for \quad x > 6 \end{cases} \tag{3.1}$$

.

There is another reference circuit described in [28],the goal is to design a circuit that replicates as faithfully as possible the trend described above. The starting circuit is the one described in [28], the solution proposed is referenced in this paper as follows;

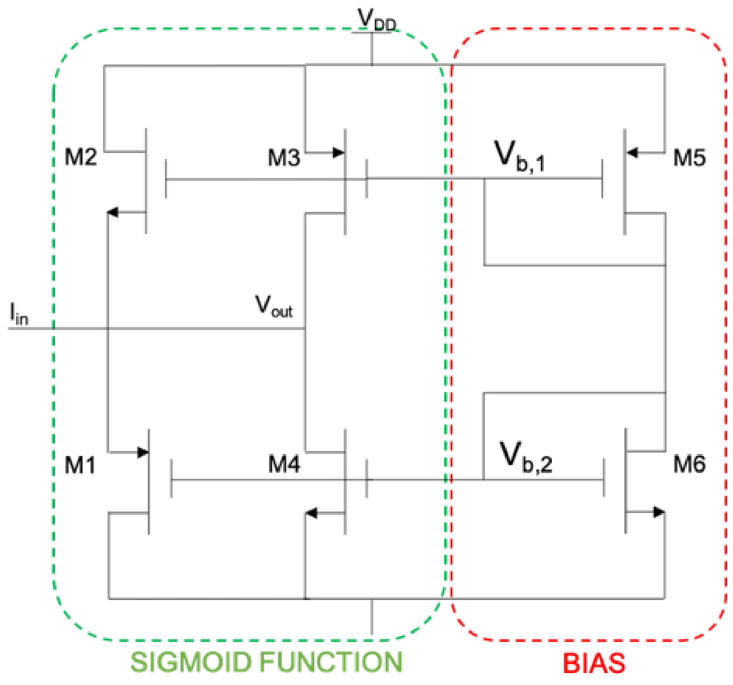


Figure 3.2: Schematic of Golnar proposed sigmoid solution

Where the right side of the circuit is the one intended for polarization, more precisely it must polarize the gate of all other transistors at a voltage equal to $V_{DD}/2$. The left circuit instead is the one that takes care of the sigmoidal activation function. Also in this case you can analyze the branches of the left circuit separately. The first branch dominates the phases where the input current is much lower or much higher than zero, where the output voltage must be stable at 0 or V_{DD} respectively. While the second branch dominates the transition phase where the input current is close to 0 and the output voltage is close to $V_{DD}/2$.

In this way the circuit can be analyzed in three distinct regions how it is shown in table :

Table 3.1: Operating regions

Region	I_{in}	V_{out}	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**
I	$<< 0$	$0 \leq V_{out} \leq V_B - V_{tn}$	CUT-OFF	SAT	SAT	LIN	SAT	SAT
II	< 0 $= 0$ > 0	$V_B - V_{tn} \leq V_{out} \leq V_B$ $V_{out} = V_B$ $V_B \leq V_{out} \leq V_B + V_{tP}$	CUT-OFF	CUT-OFF	SAT	SAT	SAT	SAT
III	$>> 0$	$V_B + V_{tp} \leq V_{out} \leq V_{DD}$	SAT	CUT-OFF	LIN	SAT	SAT	SAT

With $V_B = V_{DD}/2$. The transfer characteristic of the circuit is shown in Figure 3.3.

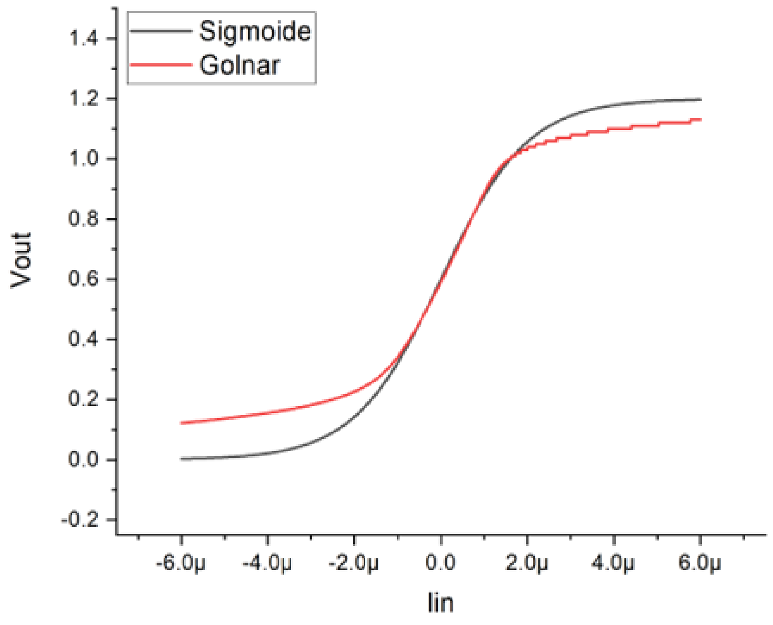


Figure 3.3: Schematic of Golnar proposed vs sigmoid solution

As can be seen from the image above, the circuit faithfully replicates the sigmoid behavior especially in the linear region, showing a larger error in the two outer regions.

3.0.2 Proposed Sigmoid Function

The objective of the proposed solution is to improve the response of the circuit in regions where $V_{DD} << \frac{V_{DD}}{2}$ and $V_{DD} >> \frac{V_{DD}}{2}$. Compared to the starting circuit, the proposed solution has the following architecture.

Because the transistors are changing their state from the saturation region to the cut-off region, it can be intrinsically presumed that there are spikes in the transitions, unfortunately, these peaks considered as noise influence directly to the total circuit, also affects to currents that are close to zero which not allow following the function. Figure 3.4 shows the proposed circuit to approximate the sigmoid function.

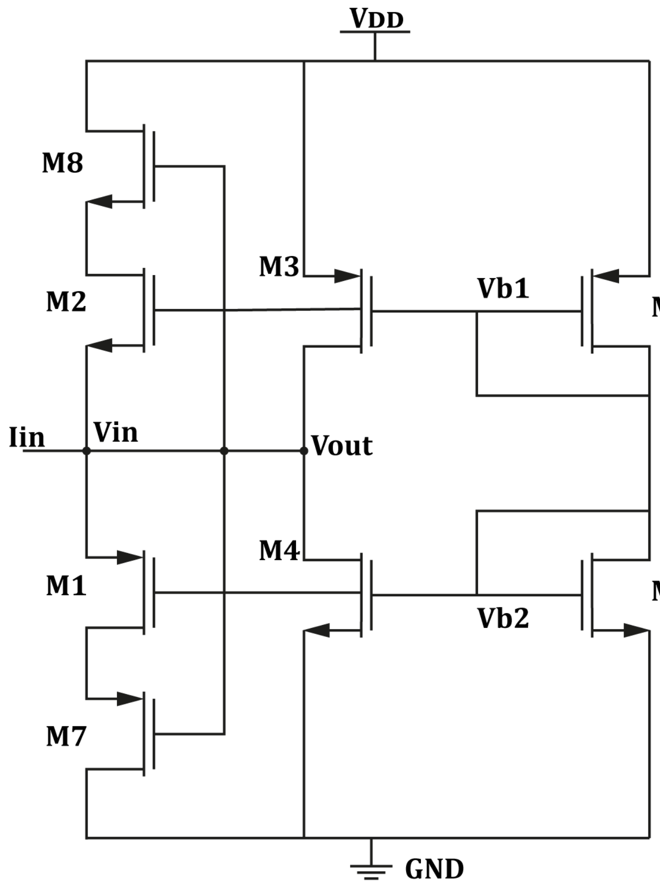


Figure 3.4: Schematic of the proposed sigmoid neuron

Qualitative Analysis

From a qualitative point of view the structure is very similar to the reference one, where the right circuit is concerned to provide the correct polarization of the transistors, while, again, the left branches dominate in two distinct regions. The first dominates when the current is much lower or much higher than zero, while the second when the current is close to zero.

The circuit shown is divided into two sections on the left side is the sigmoid function and on the right side the bias as show in Figure 3.5.

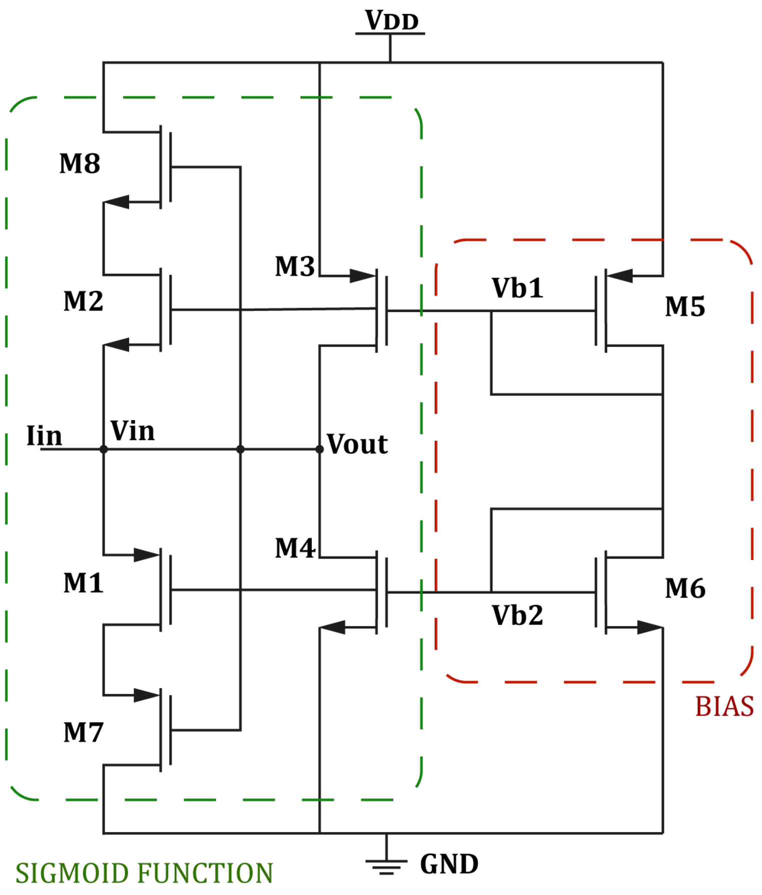


Figure 3.5: Schematic of the proposed sigmoidal neuron, Bias and Sigmoid function sections.

From circuit, transistors M5 and M6 are sized to develop bias voltage at Vdd/2, while transistor M1, M2, M3, M4, M7 and M8 produce the sigmoid function. V_{b1} is equal to V_{b2} and are considered as reference voltage, the aim is to bias the transistors. The length of all transistor is the same, particularly for this proposal lengh is 1 μ m. The following Table 3.2, shows the aspect radio of each transistor.

Table 3.2: Transistors Aspect Radio

Transistor	M1=M7	M2=M8	M3=M5	M4=M6
W/L	5.3	2.7	8	1.7

In this way the circuit can be analyzed in three distinct regions, Table 3.3 below summarizes the operation of each transistor, it operates in all regions, cut-off, saturation and lineal and input current range is from -6nA to 6nA.

Table 3.3: Operating regions

Region	I_{in}	V_{out}	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**	**M7**	**M8**
I	$<< 0$	$0 \leq V_{out} \leq 4V_T$	CUT-OFF	LIN	SAT	LIN	SAT	SAT	CUT-OFF	CUT-OFF
II	< 0	$4V_T \leq V_{out} \leq V_{DD}/2$	LIN	LIN	SAT	SAT	SAT	SAT	CUT-OFF	CUT-OFF
	$= 0$	$V_{out} = V_{DD}/2$								
	> 0	$V_{DD}/2 \leq V_{out} \leq V_{DD} - 4V_T$								
III	$>> 0$	$V_{DD} - 4V_T \leq V_{out} \leq V_{DD}$	LIN	CUT-OFF	LIN	SAT	SAT	SAT	CUT-OFF	CUT-OFF

In order to obtain Vout and for ease of calculation, threshold voltage of NMOS and PMOS are considered to be the same in all the equations, the constant K is an experimental value of 5mV. Table 3.4 shows initial assumptions to obtain this analysis.

Table 3.4: Initial Assumptions

Parameter	Value
$I_{0,n} = I_{0,p} = I_0$	564 nA
$V_{th,n} = V_{th,p} = V_{th}$	400 mV
$\lambda_{D,n} = \lambda_{D,p} = \lambda_D$	0.0205
$n_n = n_p = n$	1.3

Figure 3.6 shows the transfer characteristic with sizing proposed in Table 3.2

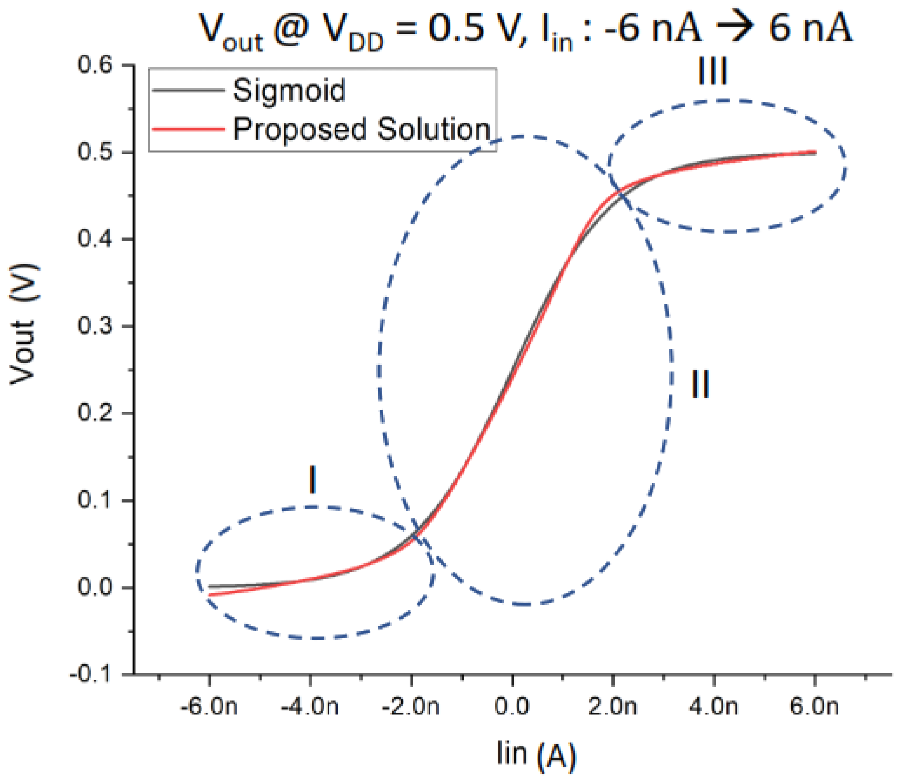


Figure 3.6: Comparison sigmoid function Taylor Series vs Propose

As can be seen from the image above, compared to the previous solution, the circuitry more faithfully replicates the sigmoid in all three regions. Moreover, compared to the previous solution, the circuit is designed to work with a supply voltage of 500 mV compared to 1.2 V of the reference solution. From an analytical point of view the response of the circuit has been divided into three main regions.

Quantitative Large Signal Analysis

For a deep analysis, each region is developed to obtain V_{out}, In region I, considering $I_{in} << 0, V_{out} << \frac{V_D D}{2}$, output voltage is shown in Equation 3.2. The input negative currents causes a small voltage in the output. Therefore, transistor M1 turns off and M4 enters in linear operation, while M5 and M6 are in saturation. The current which enter from input/output node is smaller for M2 and M3. Transistors M7 and M8 are off.

$$V_{out} = 0 + \frac{V_{out} - V_{th}}{n} + V_T \ln \left[\frac{I_{0,2} + I_{0,1}\left(1 - e^{-k}\right)}{I_{in}} \right] \tag{3.2}$$

.

While current increase, output voltage consequently increase, and this causes changes in transistor to operate in other region. How it is observed in Table 3.3, region II consider three stages, when currents is less than zero that means negative, when the current is zero and when it is grater than zero therefore positive. When current is negative the current from M3 is greater than current in M4, while output voltage increase current in M3 decrease and M4 increase until currents get cancelling each other. Finally output voltage keep increasing and current of transistor M4 is grater than the current in M3, being in the stage when input current is positive. In region II, considering $I_{in} \approx 0, V_{out} \approx \frac{V_{DD}}{2}$, output voltage is shown in Equation 3.3.

$$V_{out} = I_{in} \frac{nV_T}{2I_{0,2}\lambda_D} e^{-\frac{2V_{const} + \lambda_D V_{DD} - 2V_{th}}{2nV_T}} + \frac{V_{DD}}{2} \tag{3.3}$$

.

In region III the input current in completely positive and it generates transistor M1 and M3 operates in linear region. In region III, considering $I_{in} >> 0, V_{out} >> \frac{V_{DD}}{2}$, output voltage is shown in Equation 3.4.

$$V_{out} = V_{DD} - \frac{V_{out} - V_{th}}{n} + V_T \ln \left[\frac{I_{0,2} + I_{0,1} \left(1 - e^{-k}\right) e^{-\frac{V_{DD}}{V_T}}}{I_{in}} \right] \tag{3.4}$$

.

Taking into account the initial assumptions from Table 3.4 and the last analysis to obtain V_{out} the following Figure 3.7 shows the relation between ideal and proposed.

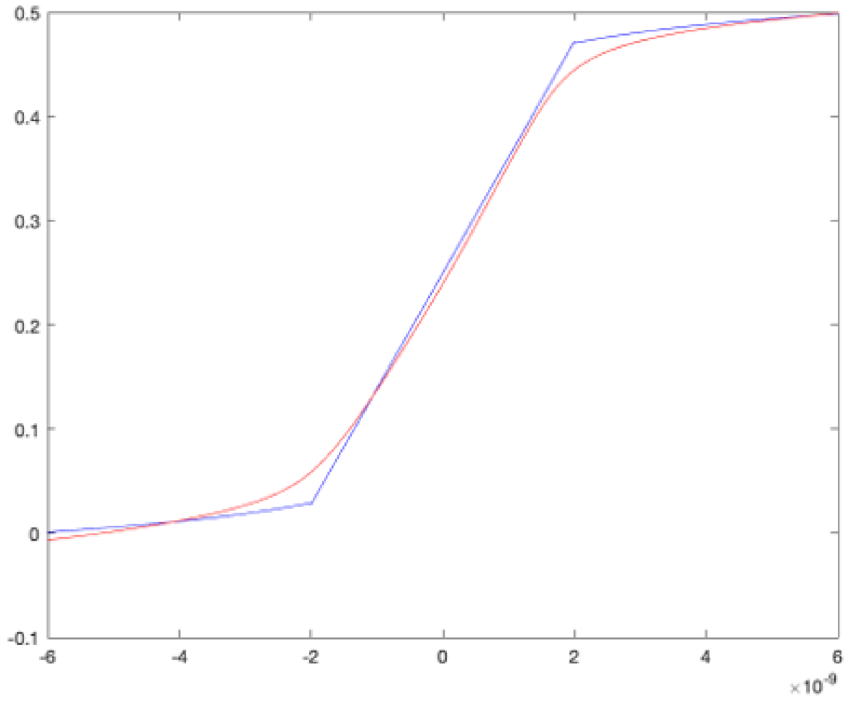


Figure 3.7: Ideal and the proposed (from the equations) sigmoid Function

Properties

An essential property that characterize this circuit focuses on transistors M3 and M4, by changing its aspect radio it is possible to modify the slope of the characteristic function in the linear region as it is shown in Figure 3.8, maintaining $I_{o,n} = I_{o,p}$.

$$W_p = \frac{\mu_u}{\mu_p} W_n \tag{3.5}$$

.

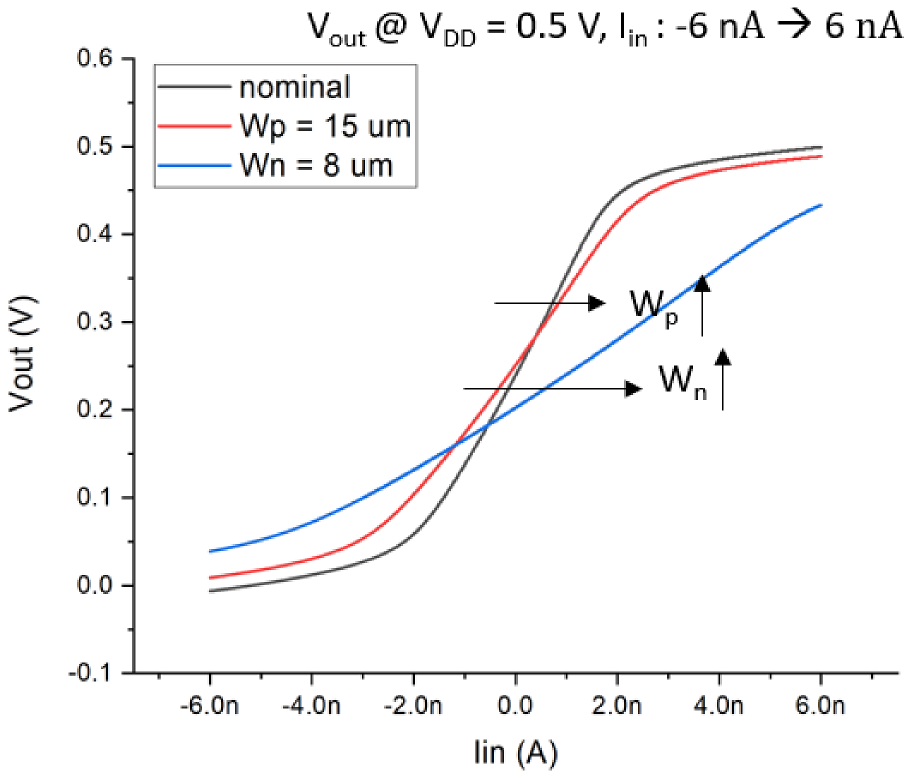


Figure 3.8: Comparison sigmoid function at different weight

Error Analysis

The following Figure 3.9 shows the perceptual error of the sigmoid function taking into account Equation 1.3 as ideal function and the response of the circuit propose as real data. As can be observed the highest peak identified by the green circle is the maximum error obtained in the linear region.

$$ERROR(\%) = \frac{|V_{out,real} - V_{out,ideal}|}{V_{DD}} \cdot 100 \tag{3.6}$$

.

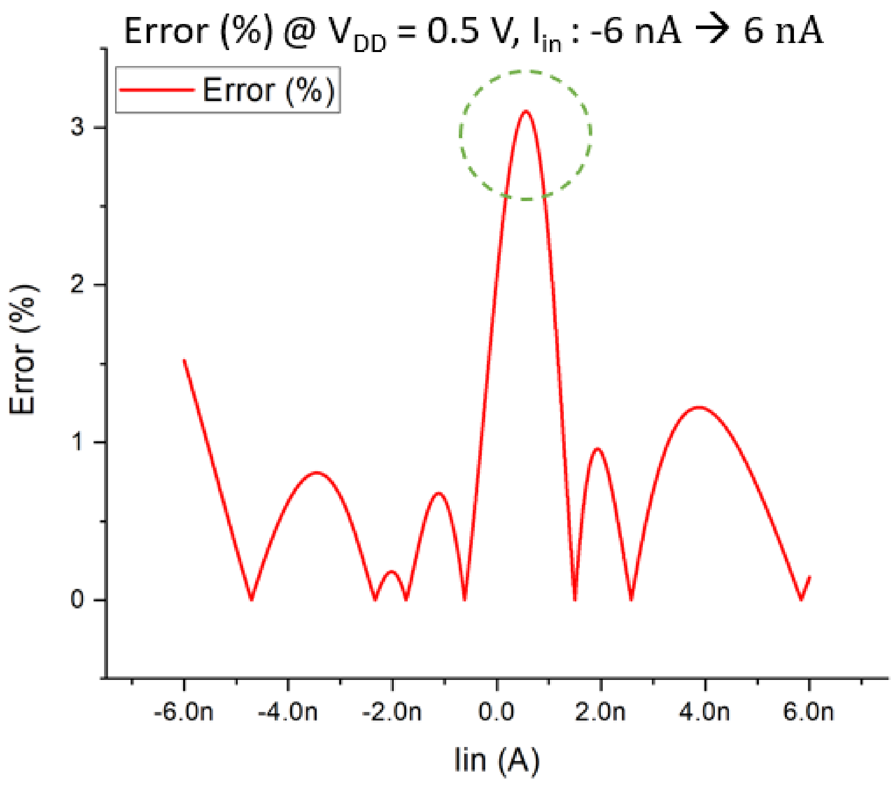


Figure 3.9: Error graph between real and ideal sigmoid function

Power Consumption

About power consumption, for its calculation, the voltage V_{DD} and the current passing through the source line are related, see Equation 3.7 .

$$Power = V_{DD} \cdot |I_{V_{DD}}| \tag{3.7}$$

.

For proposal sigmoid solution, Figure 3.10 allows observing power consumption and identify specific regions such as minimum and maximum power consumption and standby power.

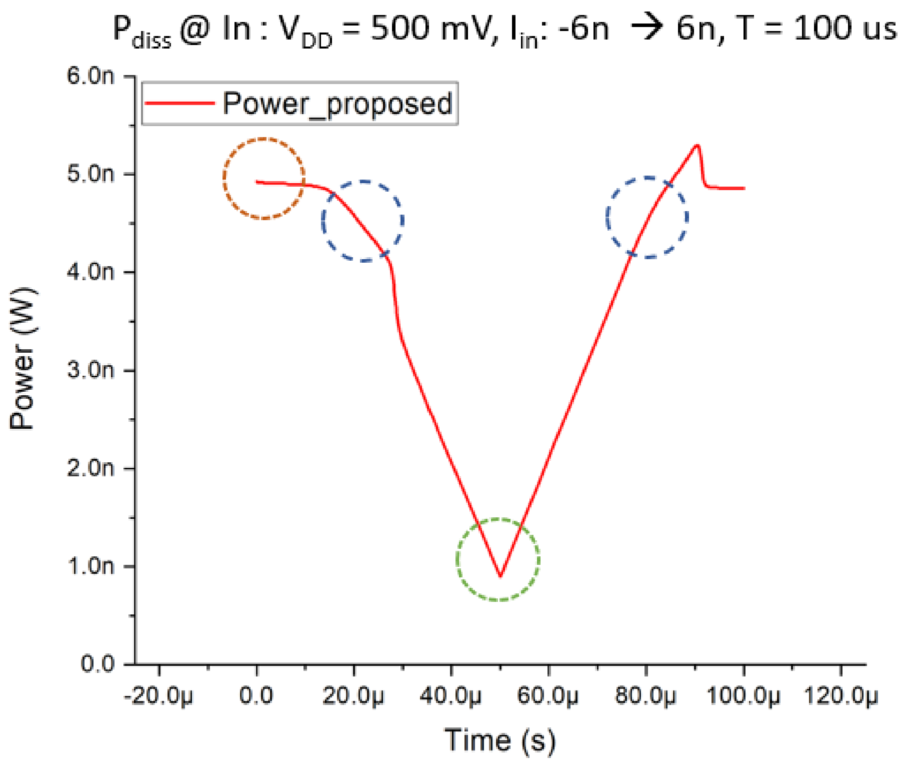


Figure 3.10: Power consumption proposed sigmoid function, red circle identifies the maximum power consumption when $I_{in} = -6nA$ and in the same way green circle represents the minimum power consumption when $I_{in} = 6nA$, moreover blue circles identify standby power when $I_{in} = 0$

This proposed is considered as reference for the following analysis. The error average is 3.1 %. About average power consumption the obtained result is 4.31 nW, maximum consumption 5.30 nW, minimum 900pW and standby power consumption 4.65nW.

3.0.3 Softmax Function

As mentioned in previous chapter, the softmax function is a sigmoid function normalized respect to all the input of the output level. Applying a softmax relates all the elements, which implies that the distinct probabilities produced by the softmax function are interrelated.

The Softmax approximation is given by Equation 1.8, and the proposed functions is defined in Equation 3.8.

$$y = \frac{e^{x_1}}{e^{x_1} + \sum_{i=2}^{M} e^{xi}} \tag{3.8}$$

.

Figure 3.11shows the approximation of a Softmax function applied in analog implementation.

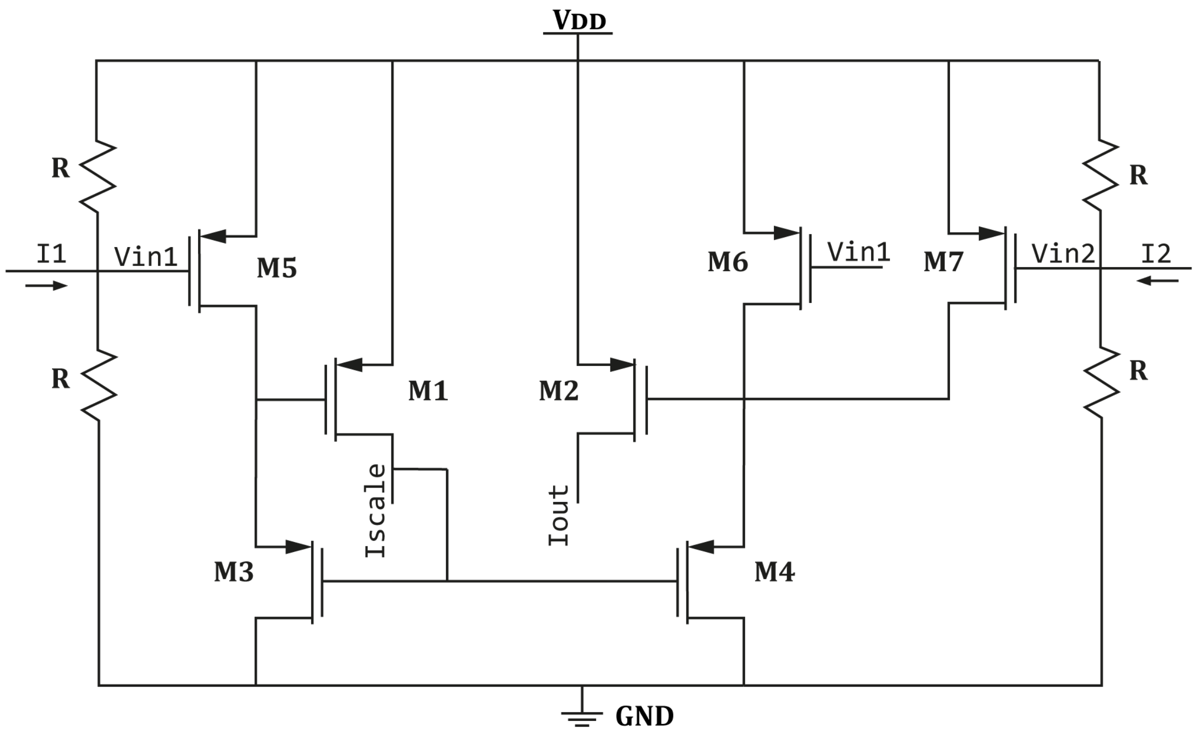


Figure 3.11: Schematic of a Softmax propose

The following Figure 3.12 shows three important sections to take into account in order to understand Sigmoid function.

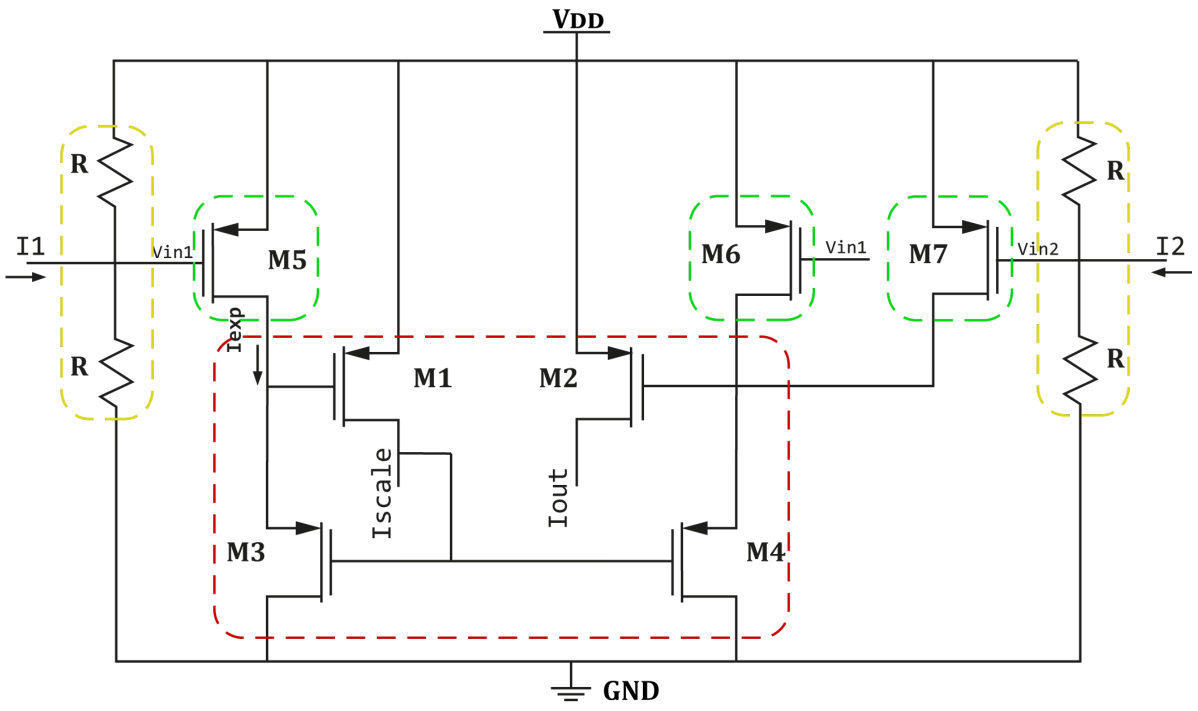


Figure 3.12: Schematic of a Softmax propose

The yellow section identified with resistors let to perform the linear transformation from current to voltage. Voltage V_{in1} or V_{in2} can be calculated by the following equation;

$$V_{in1} = \frac{V_{DD} + R \cdot I_1}{2} \tag{3.9}$$

.

The three blocks in green section formed by transistors M5, M6 and M7 perform the transformation from voltage to current in order to obtain an exponential relationship how is shown in Equation 3.10

$$I_{exp} \approx I_0 \cdot e^{\frac{V_{in}-V_{th}}{nV_T}} \tag{3.10}$$

.

Red section in other hand is a block that performs the analog division between input of the current node and the sum of the other input of the output level.

$$I_{out} \approx I_{scale} \cdot \frac{I_{exp,1}}{I_{exp,1} + I_{exp,2}} \tag{3.11}$$

.

Taking the scale current into account, the following Figure 3.13 shows the behaviour of the output current (I_{out}) in relation to the current I_1. It can be seen how the function varies depending on the value of this parameter, while the maximum output current value increases to this scale current value.

From Equation 3.11, it can be observed that there is a direct dependence of the output current I_{out} and the scale current I_{scale}, because the expressions have a direct proportional relationship.

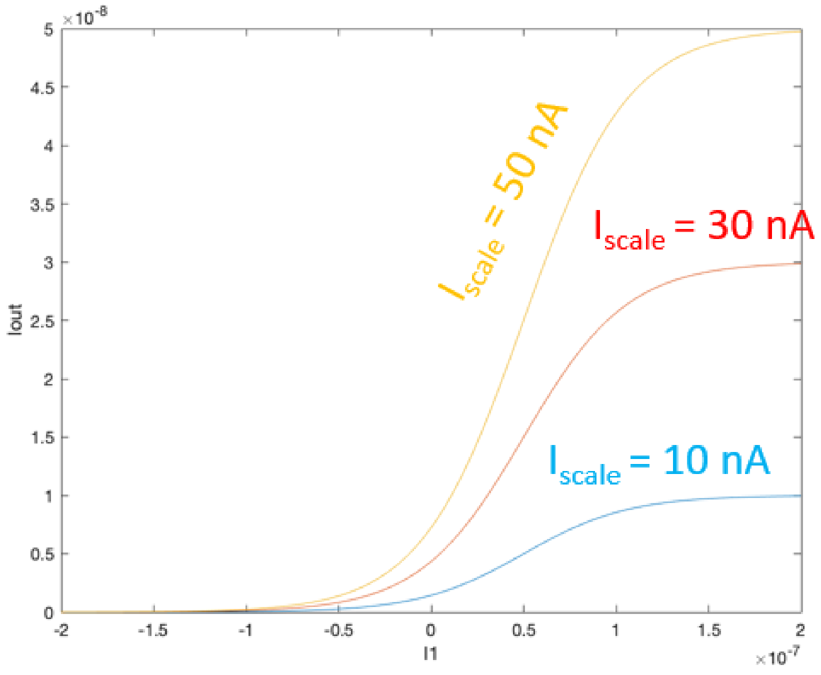


Figure 3.13: Softmax function behavior varyingI_{scale}

On the other hand there are also modifications in the function when the current I_2 is changed, which depending of the type of data, positive or negative, it is in charge of moving the function to the right or left, as shown in Figure 3.14

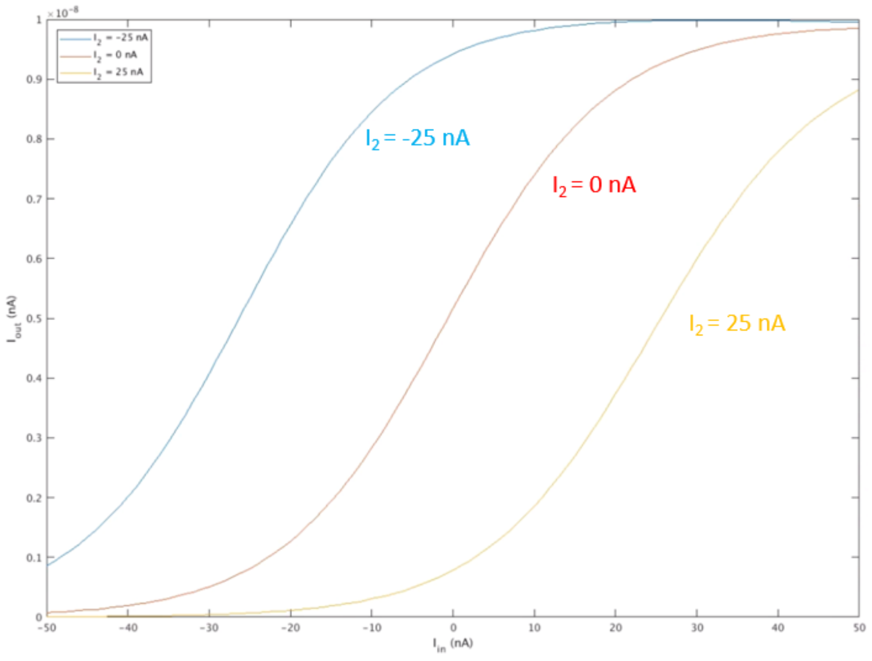


Figure 3.14: Softmax function behavior varying I_2

Finally there is a dependence by varying resistors, it allows the function vary its slope. Figure 3.15 shows the above mentioned variation.

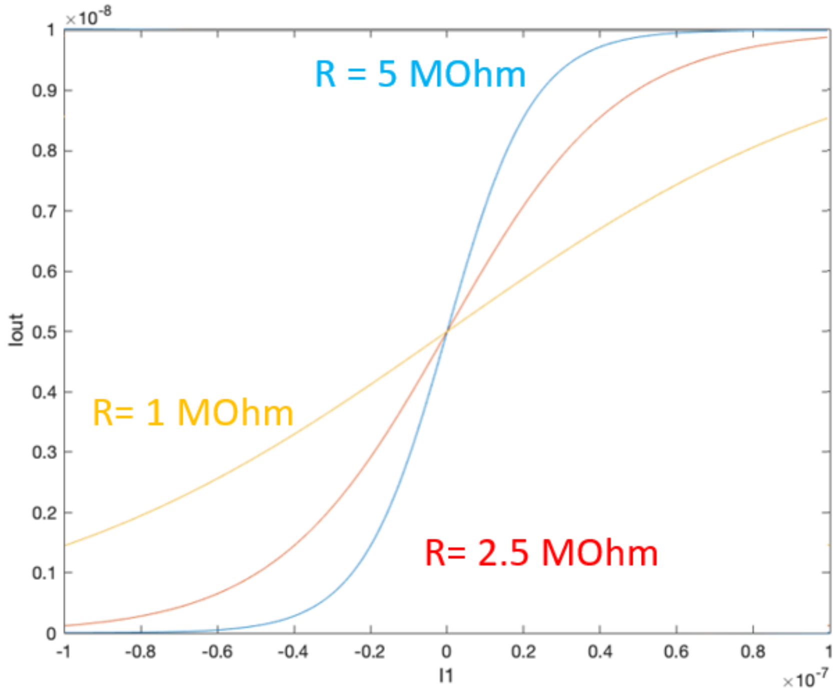


Figure 3.15: Softmax function behavior varying Resistance

Figure 3.16 shows the first result of this proposed that is considered as reference for the following analysis. The error average is 0.61 % and the maximum power consumption 286.03 nW

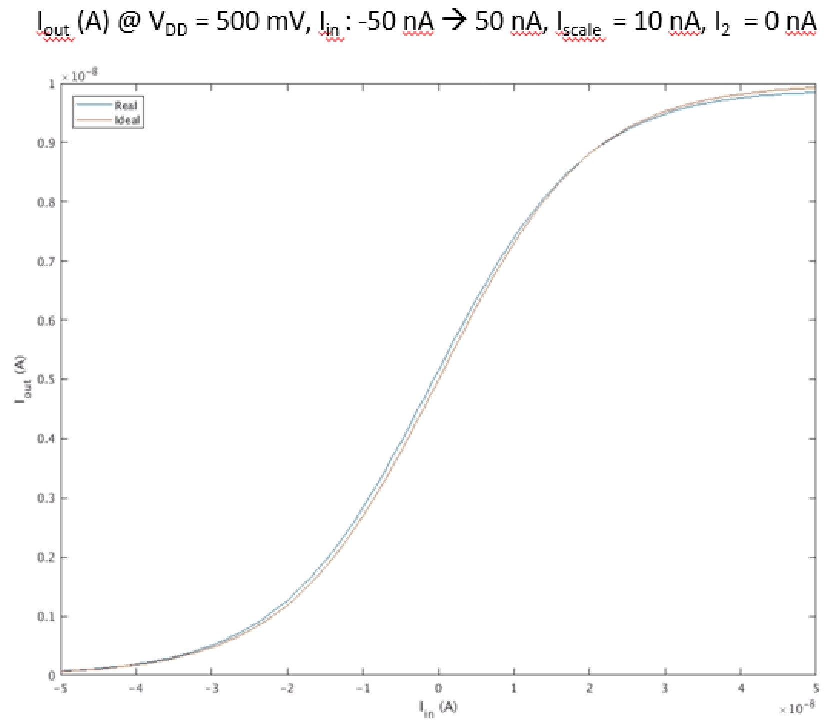


Figure 3.16: Softmax function proposed results

CHAPTER 4

RESULTS

This chapter provides the simulations and main results of the thesis. As it was mentioned in the previous chapter, the proposed sigmoid circuit is compared with the approximation of the sigmoid function that uses Taylor series. It offers a smoother and more precision response and this design is simulated at 180nm CMOS technology. About Softmax neuron similar results are obtained but due to the software used and internal conditions the region closer to V_{dd} presents an abrupt drop in data.

4.1 Sigmoid analysis

Starting from circuit proposed in Figure 3.4 and applying the dimensions established in Table 3.2, the initial result obtained is shown in Figure 4.1, input data is current and it has a range from -6nA to 6nA. Figure 4.1 shows the real response applied in Ltspice, the aim is to change the aspect radio in order to obtain a more approximate solution to the real function taking into account the power consumption and the minimum possible error.

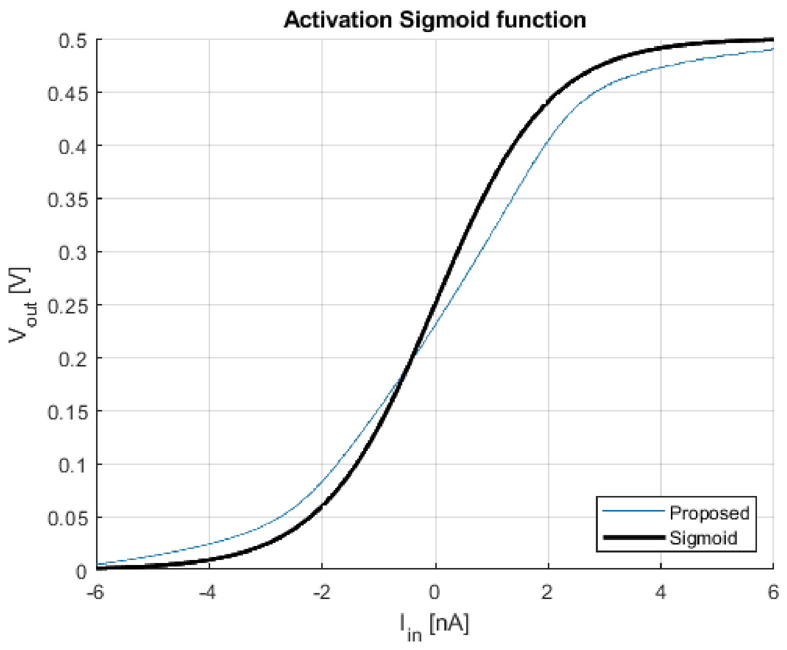


Figure 4.1: Sigmoid proposed solution applied in Ltspice

The previous Figure 4.1 mentioned shows the proposed solution very close to real sigmoid function, the following figure allows to identify the percentage error of the relation of both figures, it can be noticed that exist a high percentage of error in positive current input zone, but the average eventually is within the range less than 5%.

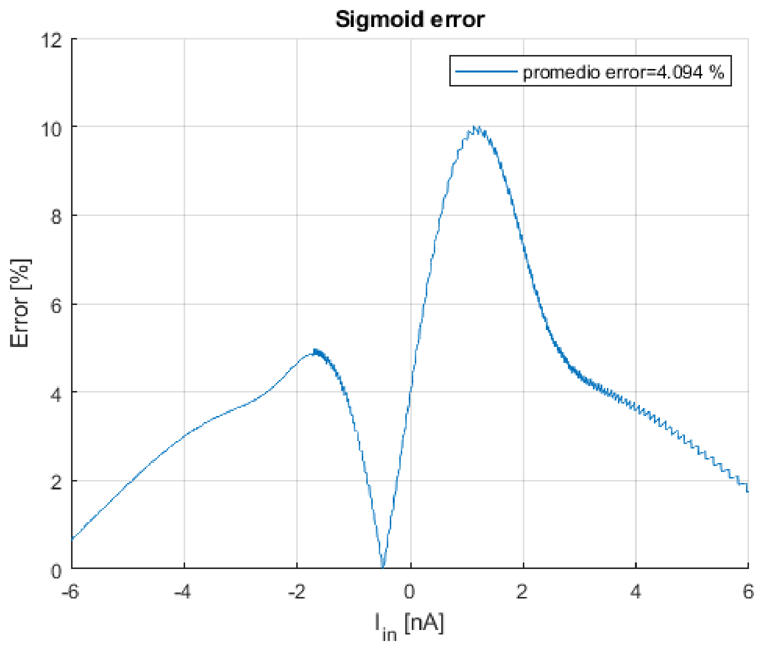


Figure 4.2: Sigmoid percentage error between real and proposed function

First we start by varying the NMOS transistors localized at the top of the circuit M3 and M5 from $5\mu m$ to $10\mu m$ in steps of $2\mu m$. Figure 4.3 represent the variation previously mentioned and consider the parameters shown in Table 4.1.

Table 4.1: Dimensions and parameters for sigmoid proposed active function Varying M3 and M5

Parameter	Value
I_{in}	-6 nA to 6 nA
V_{DD}	500 mV
$M1 = M7$	5.3μm
$M2 = M8$	2.7μm
M3=M5	5μm to 10 μm
$M4 = M6$	1.7μm

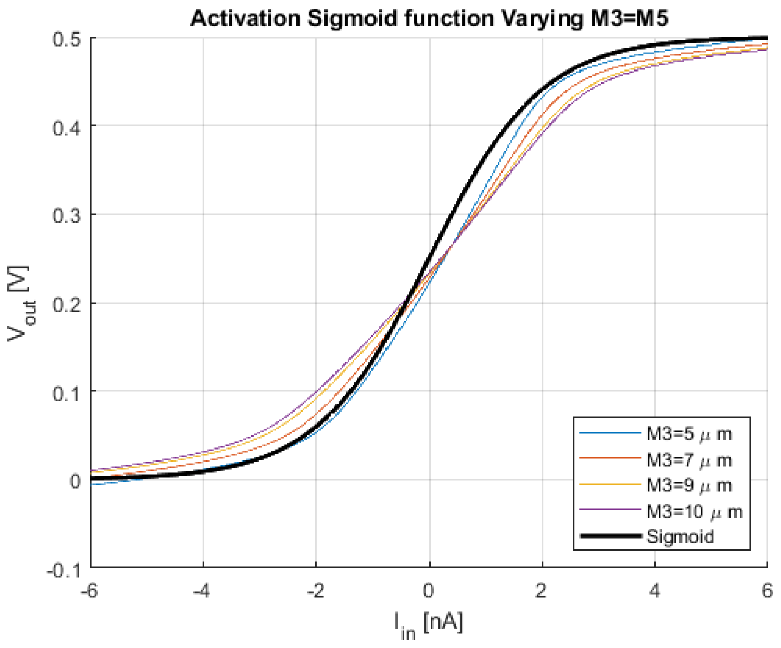


Figure 4.3: Sigmoid result by varying width of M3 and M5

How we can observe the best behaviour for M3 and M5 is at 5μm of weight which is very close to the real function. By varying transistors M4 and M5 significant changes are not sensed, transistors vary from $1.4\mu m$ to $2\mu m$ in steps of $0.2\mu m$ as shown in Figure 4.4, the rest of parameters are detail in Table 4.2.

In this case, the changes are not noticeable, basically they are all far from the real function, so it must be identified that the voltages V_{b1} and V_{b2} are at $V_{DD}/2$.

Table 4.2: Dimensions and parameters for sigmoid proposed active function Varying M4 and M6

Parameter	Value
I_{in}	-6 nA to 6 nA
V_{DD}	500 mV
$M1 = M7$	5.3μm
$M2 = M8$	2.7μm
$M3 = M5$	8 μm
M4=M6	1.4μm to 2 μm

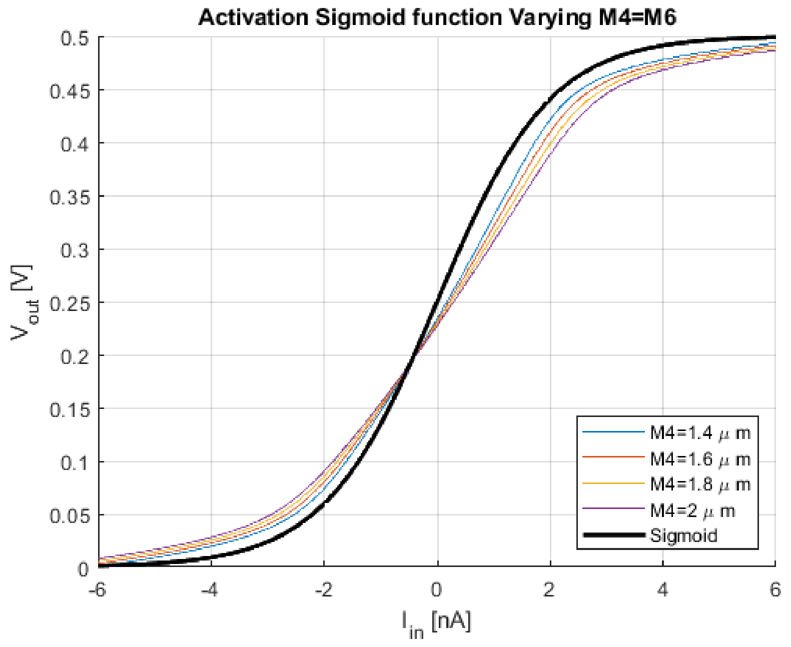


Figure 4.4: Sigmoid result by varying width of M4 and M6

I decide to modify just transistor M4 until obtain the closest value to $V_{DD}/2$ for this purpose the best response is at W=1.78 μm for M4. From this modification and again varying the NMOS M3 and M5 transistors, the results obtained are shown in Figure 4.5, and the parameters can be identified in Table 4.3

Table 4.3: Dimensions and parameters for sigmoid proposed active function Varying M3 and M5 at M6 fixed

Parameter	Value
I_{in}	-6 nA to 6 nA
V_{DD}	500 mV
$M1 = M7$	5.3μm
$M2 = M8$	2.7μm
M3=M5	5μm to 10 μm
$M4$	1.7μm
M6	1.78μm

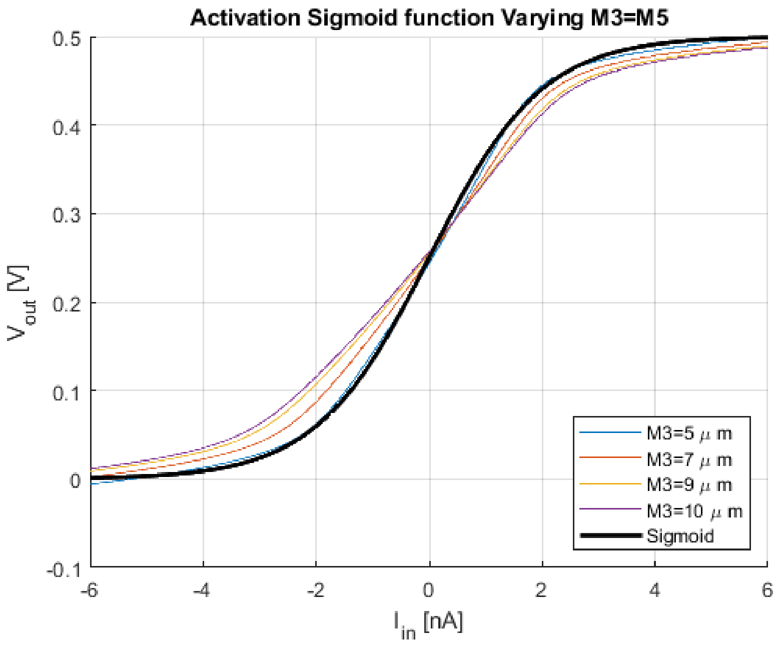


Figure 4.5: Sigmoid result by varying width of M3 and M5

Now the best response is in M5 at 5μm of weight, the following Figure 4.6 and Figure 4.7 shows error and power consumption respectively in order to verify its feasibility. In this case the highest percentage of error fluctuated in 2.5%, but the goal is the average at 0.90365% what determines the accuracy of the answer. The next Figure 4.7 shows power consumption behaviour and maximum, minimum and standby points. The average consumption is 4.37e-09 W.

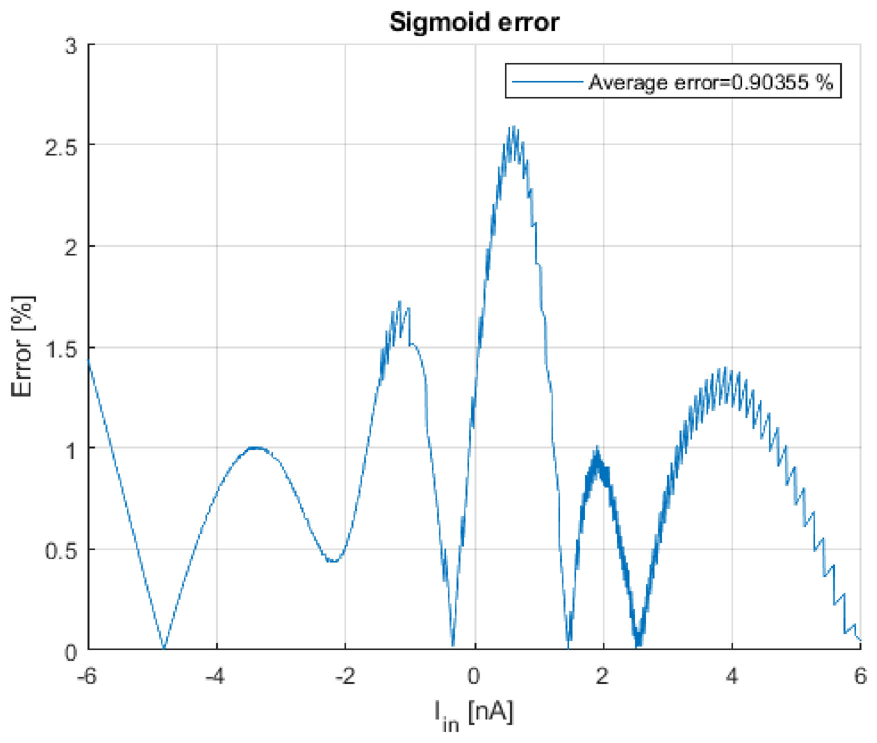


Figure 4.6: Sigmoid percentage error between real and modified proposed function

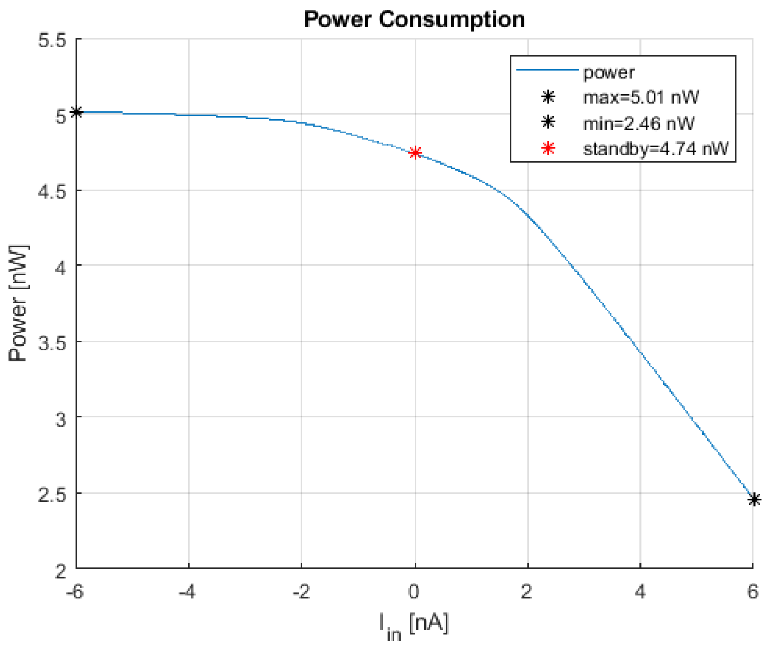


Figure 4.7: Sigmoid power consumption

4.2 Softmax analysis

An analysis similar to the sigmoid function must be done in this section, in the same way applying the schematic from Figure 4.8 the initial graph obtained is shown in figure, the initial dimensions are shown in Table 4.4, for this study the current input goes from -100 to 100nA, and particularly for this analysis the output data is the current variable compared with sigmoid where output data is voltage. But it is possible convert this variable by adding a resistor.

Table 4.4: Dimensions and parameters for Softmax proposed active function Initial conditions

Parameter	Value
I_1	-100 nA to 100 nA
I_2	0 nA
I_{scale}	10 nA
V_{DD}	500 mV
$M5 = M6 = M7$	4μm
$M1 = M2$	5μm
$M3 = M4$	5μm
R	5$M\Omega$

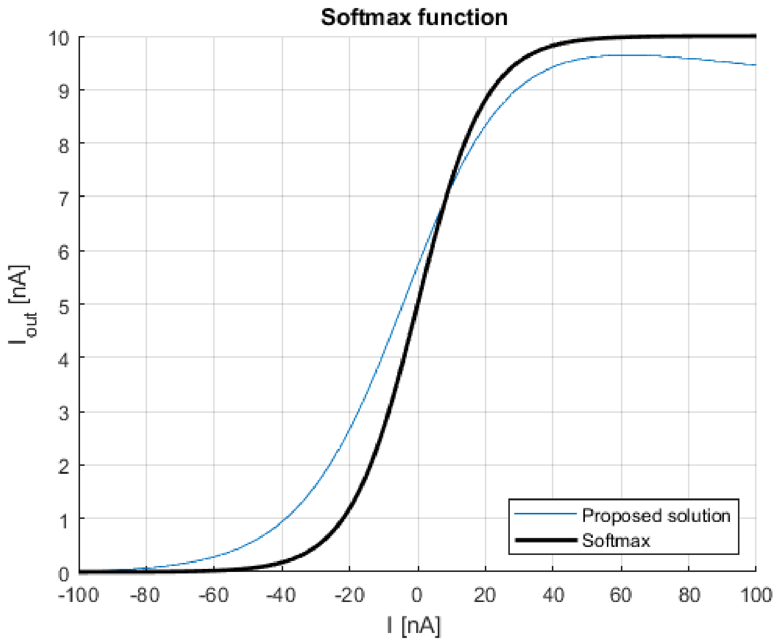


Figure 4.8: Softmax proposed solution with initial parameters

The first consideration is vary resistor R, in this way is possible to find the correct slope how it is shown in Figure 4.9. R varies from 5 $M\Omega$ to 15 $M\Omega$. The best response is between 8 $M\Omega$ and

9 $M\Omega$.

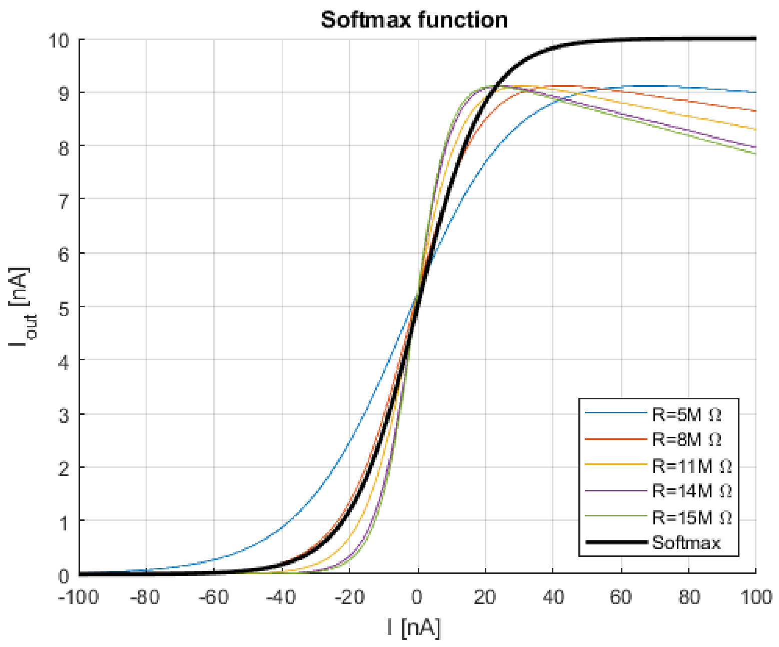


Figure 4.9: Softmax proposed solution varying R

Modifying the upper and lower transistors, is not so relevant since the curve is maintained in spite of any change, it is possible to change the curve in the positive section by varying weight because as you can see it does not reach the level of 10nA and it has a sharp drop, which would cause in this area a high percentage of error. Another direct solution is to vary I_{scale} current until it reaches 10nA, since this parameter controls the upper level of the curve.

The next analysis is shown in Figure 4.11 taking as reference resistor of 8 $M\Omega$, and varying the transistors M1, M2, M3 and M4.

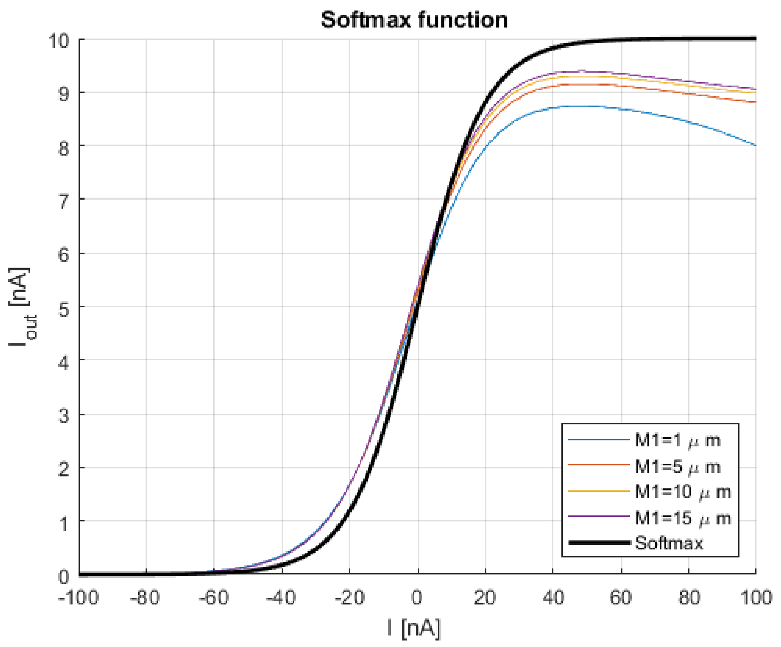


Figure 4.10: Softmax proposed solution varying M1, M2, M3 and M4.

The next figure consider the width of M1 $10\mu m$ and resistor 8 $M\Omega$ because illustrates a better behaviour. Table 4.5 shows all the parameters

Table 4.5: Softmax parameters

Parameter	Value
I_1	-100 nA to 100 nA
I_2	0 nA
I_{scale}	10 nA
V_{DD}	500 mV
$M5 = M6 = M7$	3μm
$M1 = M2$	10μm
$M3 = M4$	5μm
R	8$M\Omega$

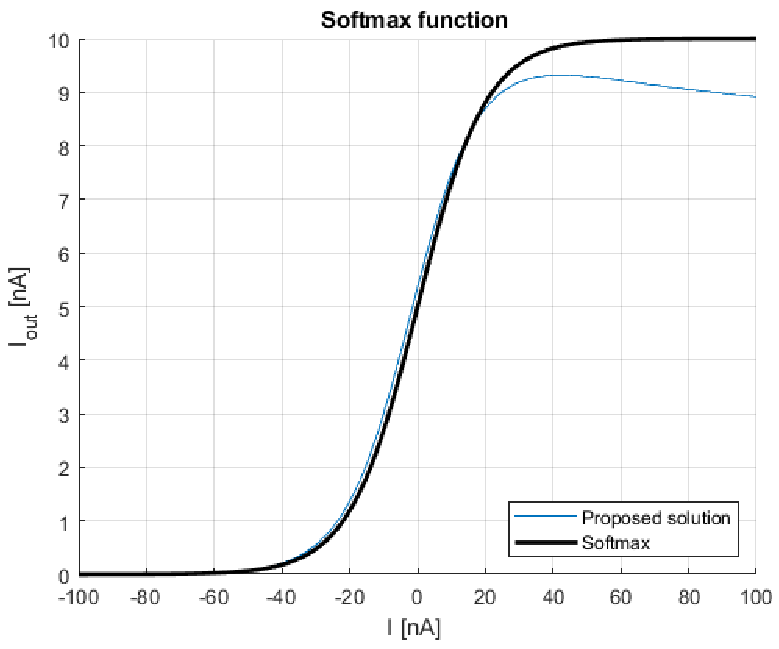


Figure 4.11: Softmax proposed solution with modified parameters

To solve the falling behavior of the curve we consider modifying a little the I_{scale}, with the objective that it can reach the maximum value of the curve and the response is shown in the following figure.

The error and power consumption are shown in the following figures respectively;

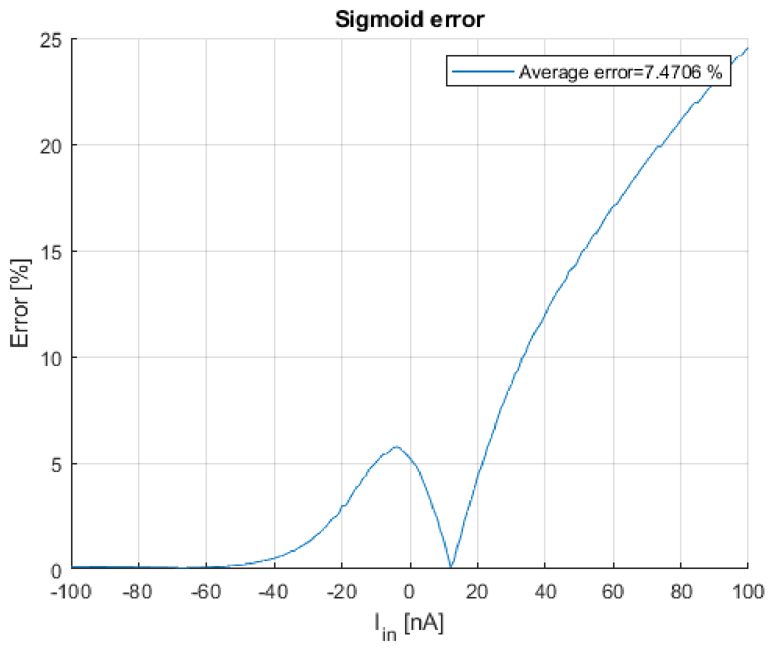


Figure 4.12: Softmax percentage error

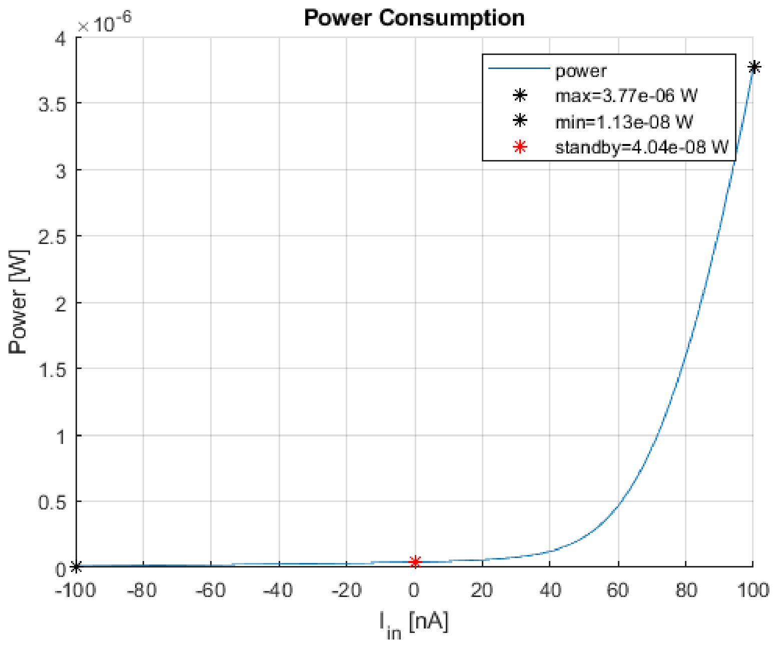


Figure 4.13: Softmax solution power consumption input current from-100 nA to 100 nA

An important aspect that can be highlighted is that the average error is high in the range of -100 nA to 100 nA but if the range is reduced from -50 nA to 50 nA, the average error also decreases, reaching a value of 2.87%.

Figure 4.13 shows Power comsuption which average is 412 nW, the maximum consumption is 3.77e-6 W. But if we reduce the input range, power consumption decrease drastically how it is shown in figure Figure 4.14 the average power consumption decrease at 81.79 nW, Maximum power consumption diminish at 230 nW

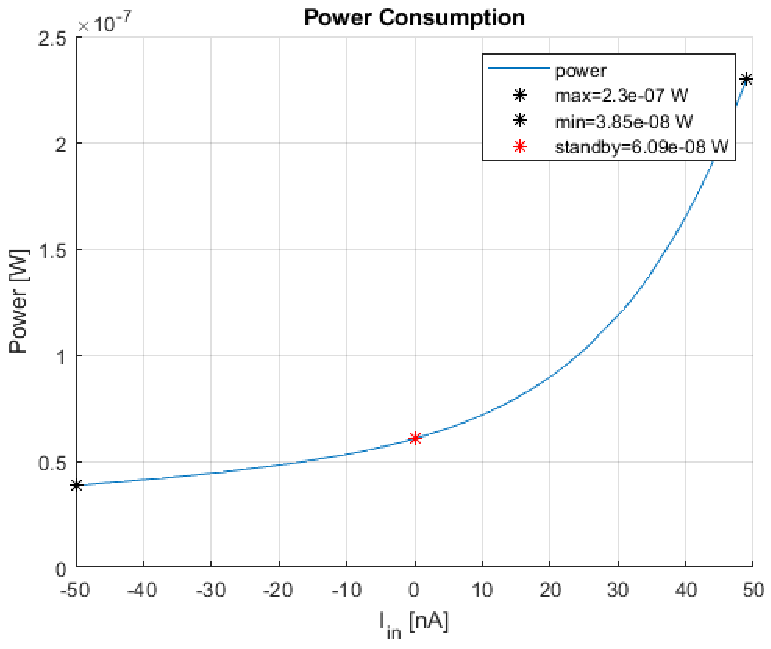


Figure 4.14: Softmax solution power consumption input current from-50 nA to 50 nA

The results obtained are represented in the following table to make a comparative analysis between all the proposed solutions. The following table relates sigmoid results;

Table 4.6: Power Consumption comparative solutions

Power Consumption	AVERAGE$[nW]$	MAX$[nW]$	MIN$[nW]$	STANDBY$[nW]$
GHOMI	4.28	5.81	2.37	4.65
GOLNAR	4.98	7.04	3.44	4.78
REFERENCE	4.31	5.30	900pW	4.65
SHAMI	369	1.56uW	0	0
PROPOSED	4.37	5.01	2.46	4.74

Finally about softmax activation function between reference and proposed considering input range from -50nA to 50nA because shows the minimum error and using $I_{scale} = 10nA$; the maximum power consumption in reference solution is 286.03 nW meanwhile in proposed solution is 230 nW.

These results as mentioned above are executed in the software Ltspice, while the reference circuit is simulated in Cadence, which is why the results are different and with a great difference, for example in Figure 4.11, in the positive zone of the softmax function, there is an abrupt fall of data, causing a high percentage of error in that section.

CHAPTER 5
CONCLUSION

This thesis describe brief description of activation functions specifically sigmoid and softmax function that are used in the field of deep learning and also about the importance of activation functions in developing an effective and efficient deep learning model and improving the performance of artificial neural networks. Activation Functions can improve the learning rate. Firstly, we have given a description of activation functions, then we have given a brief analysis about previous proposed solutions.

CMOS analogue neurons as activation function circuits are proposed, these circuits have its particularity for each analysis but the requirement to obtain the minimum power consumption is developed. The performance of the proposed circuit is simulated in Ltspice using 0.18 mm CMOS technology parameters.

The behavior of sigmoid and softmax as active functions have a basic structure composed by NMOS and PMOS transistor which depending on the analysis, their operates in cut-off, triode and saturation regions. In order to characterize its behavior, Ltspice design environment was used.

Since simple sigmoid activation function circuit is proposed and modifying some parameters, we finally lead to a significant reduction in power consumption compared with similar solution as reference. So, the maximum power consumption for neuron circuit is 5.01 nW at a 0.5V supply voltage, showing a less power consumption respect to the previous circuits.

In the same way for the softmax function, since input range from -50nA to 50nA and supply voltage is 0.5V the maximum power consumption in reference solution is 286.03 nW meanwhile in proposed solution is 230 nW.

Finally, as a summary, the results the results obtained are equal or even better in terms of power consumption with respect to the reference solutions, but considering certain aspects and certain ranges where the behavior of the function is closer to the real one. But the results achieved have been satisfactory.

REFERENCES

[1] F. Deliang, S. Yong, and Raghunathan, “Stt-snn: A spin-transfer-torque based soft-limiting non-linear neuron for low-power artificial neural networks,” *IEEE Transactions on Nanotechnology*, vol. 14, no. 6, 2015.

[2] C. Yang, H. Kim, S. P. Adhikari, and L. O. Chua, “A circuit-based neural network with hybrid learning of backpropagation and random weight change algorithms,” *Sensors*, vol. 17, 2017.

[3] R. P. Lippmann, “An introduction to computing with neural nets,” *IEEE Assp Magazine*, vol. 4, no. 2, pp. 4–22, 1987.

[4] G. Andisheh and D. Mehdi, “Design of a new cmos low-power analogue neuron,” *IETE JOURNAL OF RESEARCH*, vol. 64, no. 1, pp. 67–75, 2018.

[5] D. Kriesel, *A Brief Introduction to Neural Networks*. 2007.

[6] J. M. Zurada, *Introduction to artificial neural systems*. West Publishing, 1992, pp. 25–52.

[7] P. J. Braspenning, F. Thuijsman, and A. Weijters, *Artificial Neural Networks: An Introduction to ANN Theory and Practice*. 1995, vol. 931.

[8] D. Graupe, “Principles of artificial neural networks,” *World Scientific*, vol. 3, 1999.

[9] M. A. Nielsen, *Neural Networks and Deep Learning*. Determination Press, 2015.

[10] I. Kononenko and M. Kukar, *Machine Learning and Data Mining*. Horwood Publising Limited, 2007.

[11] M. Behrang, E. Assareh, A. Ghanbarzadeh, and A. Noghrehabadit, “The potential of different artificial neural network (ann) techniques in daily global solar radiation modeling based on meteorological data,” *Solar Energy*, 2010.

[12] F. Djeffal, M. Chahdi, A. Benhaya, and M. Hafiane, “An approach based on neural computation to simulate the nanoscale cmos circuits: Application to the simulation of cmos inverter,” *Solid-State Electronics*, vol. 65, no. 1, pp. 48–56, 2007.

[13] B.G.M.Vandeginste, D.L.MassartL, M.C.Buydens, S. Jong, P.J.Lewi, and J.Smeyers-Verbeke, *Handbook of Chemometrics and Qualimetrics Part B*. 1998, vol. 20.

[14] Z. Yilbas, *Optical Method and Neural Network for Surface Roughness Measurement and Surface Pattern Classification*. 1998.

[15] A. Ghomi and M. Dolatshahi, "Design of a new cmos low-power analogue neuron," *IETE JOURNAL OF RESEARCH,*, vol. 64, no. 1, pp. 67–75, 2018.

[16] A. Joubert, B. Belhadj, O. Temam, and R. Heliot, "Hardware spiking neurons design: Analog or digital," *IEEE World Congress on Computational Intelligence*, 2012.

[17] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research," *Journal of Pharmaceutical and Biomedical Analysis*, vol. 22, pp. 717–727, 2000.

[18] C. E. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *ArXiv*, 2018.

[19] S. Siddharth Sharma and A. Athaiya, "Activation functions in neural networks," *International Journal of Engineering Applied Sciences and Technology*, vol. 4, pp. 310–316, 2020.

[20] K. Sammut and S. Jones, "Implementing nonlinear activation functions in neural network emulators," *IEEE, Electronics Letters*, vol. 27, no. 12, pp. 1037–1038, 1992.

[21] M. C. Robles and L. Serrano, "A novel minimum-size activation function and its derivative," *IEEE Trans. Circuits Syst. II 56 (4)*, vol. 56, no. 4, pp. 280–284, 2009.

[22] F. Pie¸kniewski and L. Rybicki, "Visual comparison of performance for different activation functions in mlp networks," *IEEE IJCNN*, vol. 4, pp. 2947–2952, 2004.

[23] J. Shamsi, A. Amirsoleimani, S. Mirzakuchaki, A. Ahmadi, S. Alirezaee, and M. Ahmadi, "Hyperbolic tangent passive resistive-type neuron," *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015.

[24] M. Dukhan and A. Ablavatski, "The two-pass softmax algorithm," *arXiv*, 2001.

[25] D. Sharma, Z. A. Jaffery, and N. Ahmad, "Categorical vehicle classification using deep neural networks," *IEEE Xplore*, 2019.

[26] I. Kouretas and V. Paliouras, "Simplified hardware implementation of the softmax activation function," *Modern Circuits and Systems Technologies (MOCAST)*, 2019.

[27] M. S. Kame, "Neural networks:the state of the art," *Proceedings. Eleventh International Conference on Microelectronics*, 1999.

[28] G. Khodabandehloo, M. Mirhassani, and Majid, "Analog implementation of a novel resistive-type sigmoidal," *IEEE transactions on very large-scale integration (VLSI) systems*, vol. 20, no. 4, 2012.

[29] S. M. Gowda, B. J. Sheu, J. Choi, C. G. Hwang, and J. S. Cable, "Design and characterization of analog vlsi neural network modules," *Solid-State Circuits*, vol. 28, no. 3, pp. 301–313, 1993.

[30] L. Gatet, H. T. B. adn, and F. Bony, "Comparison between analog and digital neural network implementations for range-finding applications," *EEE Trans. Neural Netw*, vol. 20, no. 3, pp. 460–470, 2009.