# Process, Bias and Temperature Scalable CMOS Analog Computing Circuits for Machine Learning

Pratik Kumar[†], Ankita Nandi[†], Shantanu Chakrabartty*, Chetan Singh Thakur[†]
{pratikkumar, ankitanandi, csthakur}@iisc.ac.in, {shantanu}@wustl.edu
[†]Department of Electronic Systems Engineering, Indian Institute of Science, Bangalore, India, 560012
*Department of Electrical and Systems Engineering, Washington University in St. Louis,USA, 63130

***Abstract*—Analog computing is attractive compared to digital computing due to its potential for achieving higher computational density and higher energy efficiency. However, unlike digital circuits, conventional analog computing circuits cannot be easily mapped across different process nodes due to differences in transistor biasing regimes, temperature variations and limited dynamic range. In this work, we generalize the previously reported margin-propagation-based analog computing framework for designing novel *shape-based analog computing* (S-AC) circuits that can be easily cross-mapped across different process nodes. Similar to digital designs S-AC designs can also be scaled for precision, speed, and power. As a proof-of-concept, we show several examples of S-AC circuits implementing mathematical functions that are commonly used in machine learning architectures. Using circuit simulations we demonstrate that the circuit input/output characteristics remain robust when mapped from a planar CMOS 180nm process to a FinFET 7nm process. Also, using benchmark datasets we demonstrate that the classification accuracy of a S-AC based neural network remains robust when mapped across the two processes and to changes in temperature.**

***Index Terms*—Machine Learning, Process Scalability, Approximate Computing, Margin Propagation, Shape-based Computing.**

## I. Introduction

ANALOG computing techniques are attractive for implementing machine learning (ML) architecture because of the potential to achieve high computational density and high energy-efficiency when compared to an equivalent digital implementation. ML training also allows for offline and online calibration which can compensate for analog artifacts due to device mismatch and non-linearity [1]. Examples of previous analog ML implementations include [2], [3], [4], [5]. However, one of the key advantages of digital implementation is its *process scalability* where a digital circuit module designed in one process node (typically with a larger feature size) can be mapped to a more advanced process node (with a smaller feature size) with minimal to no circuit modification [6], [7]. Due to *process scalability*, digital implementations (both ML and non-ML architectures) can benefit from sub-10nm technology scaling in terms of improved speed, power and compute density. On the other hand, scaling analog computing circuits across process nodes has been difficult due to several reasons [8], [9] and can be highlighted using Fig. 1. The

*This work has been accepted in IEEE for publication and is available with journal DOI: 10.1109/TCSI.2022.3216287. Copyright may be transferred without notice, after which this version may no longer be accessible.

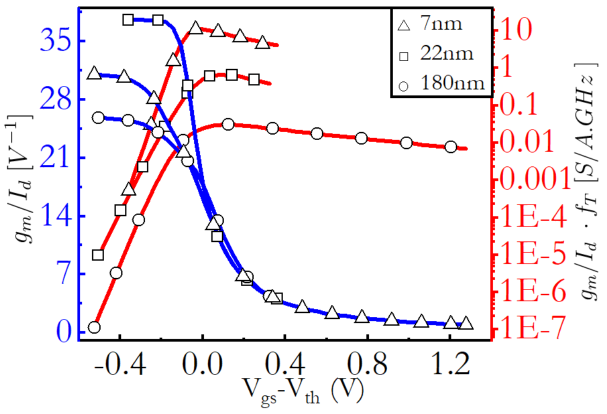

Fig. 1. Plot of transconductance efficiency ($g_m/I_d$) as a function of $V_{gs}-V_{th}$ at different process nodes [11]. The plot also shows the product of $g_m/I_d$ and speed ($f_T$), denoting the efficiency peak obtained in moderate inversion. Plots are shown for n-type planar CMOS and FinFET at different process nodes. Here the maximum supply voltages of each process node can be noted as $1.8V$, $0.8V$ and $0.7V$ for 180nm, 22nm and 7nm, respectively.

figure compares transistor performance using the product of transconductance efficiency and speed as a figure-of-merit (FOM) [10] for a 180nm planar CMOS process and for a 7nm FinFET process [11], [12]. The FOM is shown in Fig. 1 for three different biasing regimes, weak-inversion (WI), moderate-inversion (MI), and strong-inversion (SI). Note that in Fig. 1, biasing regimes can be differentiated from each other by their respective transconductance efficiencies ($g_m/I_d$). For $V_{gs} - V_{th} < 0$, transistors are operating in WI while the transition slope denotes MI regime. It can also be seen from Fig. 1 that for 7nm FinFET process, the highest dynamic range and the best FOM is achieved in the moderate inversion regime. As the feature size is scaled to a more advanced process node (7nm feature size), the moderate inversion region dominates the transistor dynamic range. Whereas, most analog computing circuits that exploit the large-signal transistor I-V characteristics operate either in the weak-inversion regime [13] or in the strong-inversion regime [14]. These circuits cannot be directly scaled to the sub-10nm process nodes without significant performance degradation.

In this paper, we propose analog computing circuits that are process scalable and hence similar to digital designs, can be used as synthesizable analog standard cells. At the core of the proposed circuits is a generalization of the previously reported bias scalable analog computing framework called margin prop-

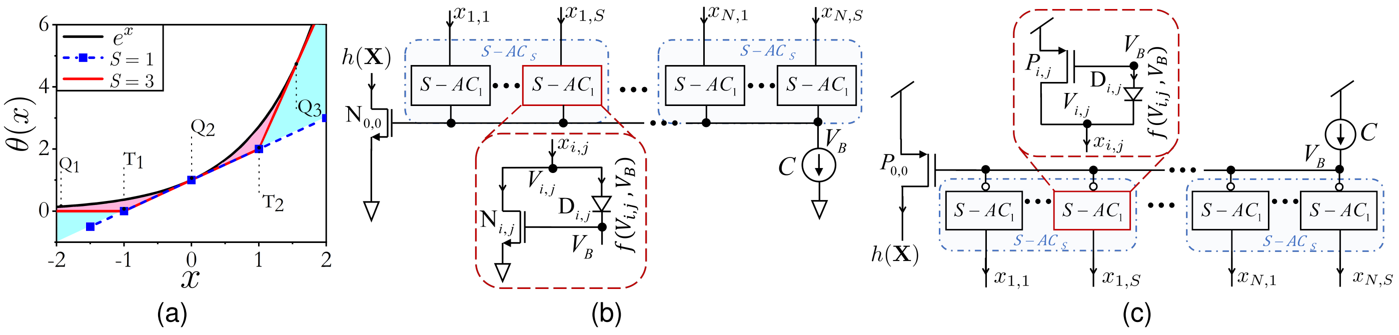

Fig. 2. (a) Plot showing the approximation of a non-linear function $\theta(x) \simeq e^x$ using linear splines ($S$). Here, the approximations are shown for different spline ($S$) counts i.e, $S = 1, 3$, where $Q_1, Q_2, Q_3$ are the tangential points and $T_1, T_2, T_3$ are the tuning points; (b) Implementation of N-type S-AC circuit for $N$ inputs and $S$ splines, the inset shows the circuit implementation of a single S-AC unit using n-type FET and a diode; (c) Implementation of P-type S-AC circuit for $N$ inputs and $S$ splines, the inset shows the circuit implementation of a single S-AC unit using p-type FET and a diode. The circle on the P-type S-AC unit is used to differentiate between an N-type S-AC unit and a P-type S-AC unit.

agation [15]. In this work, we show that generalized margin propagation leads to a novel shape-based analog computing (S-AC) framework where circuit accuracy can be traded off with speed and power, like digital designs. Furthermore, S-AC circuits can be scaled across processes and can work across different biasing regimes and across different operating temperatures. As a proof-of-concept, this paper presents the design of several key analog computing circuits commonly used in ML architectures. The key contributions of this work in relation to previous approaches are as follows:

- Generalization of Margin Propagation (MP) design framework and introducing a multi-spline approach that allows trading off computational accuracy with speed and power. Using this multi-spline approach, we design a basic prototype function and show that its characteristic is robust with respect to biasing, process nodes, and temperature variations.
- Using the basic prototype function, we synthesize S-AC circuits that approximate different functions commonly used in ML architecture.
- Using the basic S-AC circuits, we present a complete design of a 3-layer neural network that is process and bias scalable with respect to classification accuracy on benchmark datasets.

The rest of the sections are organized as follows. Section II presents the mathematical framework for generalized margin propagation (GMP) and the S-AC framework. Section III shows the MOS circuit implementation of the basic S-AC circuit and demonstrates its scalability at different biases and process nodes. In Section IV we show the design of widely used machine learning functions implemented using a basic S-AC unit and its performance trade-off analysis. Section V presents a case study of S-AC based 3-layer neural network. Section VI concludes the papers with discussions and final remarks.

## II. Generalization of Margin Propagation and Shape-based Analog Computing

In this section, we extend our previous work in the area of bias-scalable analog computing circuits [15] using a multi-spline approach which is then generalized to shape-based computing. These shapes or prototype functions are then shown to be implemented using physical operating principles of MOSFETs and diodes.

### A. Multi-Spline Approximation of log-sum-exp function

Similar to the previous Margin Propagation framework [15], the starting point of our framework is an approximation of the log-sum-exp function [15] $h_{\log} : \mathbb{R}^N \to \mathbb{R}$ given by

$$h_{\log}(\mathbf{x}) = C \cdot \log \left( \sum_{i=1}^{N} e^{\frac{x_i}{C}} \right) \quad (1)$$

where $C$ is a hyper-parameter and $\mathbf{x} \in \mathbb{R}^N$ is a vector with elements $x_i \in \mathbb{R}$. Equation (1) can be written as

$$\sum_{i=1}^{N} e^{\frac{x_i - h_{\log}}{C}} = 1 \quad (2)$$

which is an equivalent non-linear constraint satisfaction problem. In our previous Margin Propagation related work [15] we had approximated the $e^{(\cdot)}$ using a single spline as

$$e^x \cong [x]_+ \quad (3)$$

where $[.]_+$ denotes a rectifying linear unit (ReLU) function. However, the rectification function can be substituted based upon some key properties to formulate the GMP formulation explained in detail in Section II-B.

The single-spline approximation ($S = 1$) of the exponential function is highlighted in Fig. 2a. Using a single spline, (2) can be expressed as a piece-wise approximation $h \approx h_{\log}$ where $h$ is computed as a solution to the non-linear equation

$$\sum_{i=1}^{N} [x_i - h]_+ = C \quad (4)$$

Fig. 2a also provides the insight that the exponential function can be better approximated using multiple splines ($S > 1$) defined by different parameters $Q_j, T_j$, where $j = 1, .., S$. It also shows that the single spline approximation has a wider margin due to one intersecting solution, while the three spline approximation shows a narrower margin. Thus introducing multiple splines in the approximation provides control over the

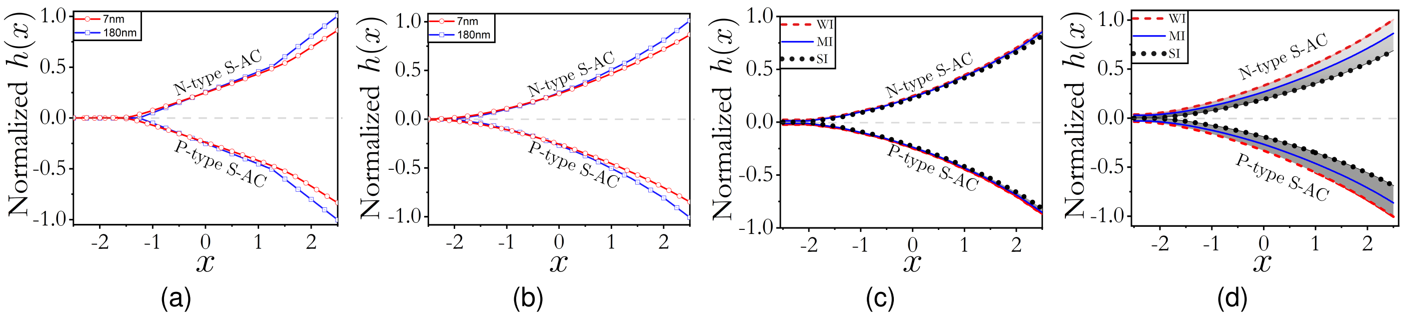

Fig. 3. Basic S-AC functions implemented by the N-type S-AC circuit and P-type S-AC circuit in different process technology nodes when a single input $x$ is varied: (a) for spline-count $S = 1$;(b) for spline-count $S = 3$; (c) for different operating regimes in a 180nm process node; (d) for different operating regimes in a 7nm process node. The output current $h(x)$ shown in the plots have been normalized with respect to $I_{max}$, where $I_{max}$ is the maximum current for each biasing regime.

precision and accuracy trade-off. In this work, we, therefore, generalize the MP framework in (4) by approximating $e^{(\cdot)}$ using multiple splines as

$$e^x \cong \sum_{j=1}^{S} \left( e^{Q_j} - \sum_{k=1}^{j-1} e^{Q_k} \right) [x - T_j]_+ \tag{5}$$

The rationale for choosing the values of $Q_j, T_j$, where $j = 1,..,S$ in (5) is provided in Appendix A. Appendix A also shows that, for specific values of $Q_j, T_j$, where $j = 1,..,S$, (5) leads to a simplified multi-spline approximation like (4) and given by

$$\sum_{i=1}^{N} \sum_{j=1}^{S} [x_{i,j} - h]_+ = C \tag{6}$$

Thus, $h(.)$ is a function of a matrix whose elements are $x_{i,j}, i = 1,..,N; j = 1,..,S$. Equation (6) therefore serves as a generalization of margin-propagation using multiple splines.

## B. Generalized Margin Propagation and S-AC

Both the log-sum-exp function $h_{\log}(.)$ and its multi-spline approximation $h(.)$ satisfy properties (7) and (8) given by

$$1 \geq \frac{\partial h_{\log}}{\partial x_i}, \frac{\partial h}{\partial x_i} \geq 0, \forall i \tag{7}$$

$$\begin{aligned} \lim_{x_i \to \infty} \frac{\partial h_{\log}}{\partial x_i}, \frac{\partial h}{\partial x_i} &= 1 \\ \lim_{x_i \to -\infty} \frac{\partial h_{\log}}{\partial x_i}, \frac{\partial h}{\partial x_i} &= 0 \end{aligned} \tag{8}$$

These properties indicate some common features of the shape of both $h_{\log}(.)$ and $h(.)$. It also indicates a mechanism to further generalize MP shapes $g(\cdot)$ instead of splines (5). If a function $g : \mathbb{R} \to \mathbb{R}$ satisfies the following property

- $g(0) = 0$ and $g(\cdot)$ is always positive or $g(\cdot) \geq 0$
- $g'(\cdot) \geq 0$ and $g(-\infty) = 0$
- $g(\cdot)$ is monotonic function

then we propose a function $h$, which is computed as the solution to the following constraint

$$\sum_{i=1}^{N} \sum_{j=1}^{S} g(x_{i,j} - h) = C, \forall i = 1,..,N, \forall j = 1,..,S \tag{9}$$

Since the choice of $g(.)$ is arbitrary, we refer to the function $h(.)$ as a shape-based function and any computation that exploits the constraint (9) as shape-based analog computing (S-AC). In the next section, we show how $g(.)$ can be implemented using transistor and diode characteristics in a bias and process scalable manner.

## III. S-AC Circuit Implementation and Analysis

### A. FET device characteristics and the basic S-AC Circuit

In its most general form, the drain-to-source current ($I_{ds}$) flowing through an n-type MOSFET can be expressed as the difference between the forward and reverse currents [16], [17] as

$$I_{ds} = I_s[f(V_g, V_s) - f(V_g, V_d)] \tag{10}$$

where $I_s$ is the specific current and $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a function that models the forward and reverse currents with respect to the gate ($V_g$), drain ($V_d$) and source ($V_s$) voltages respectively. A similar expression as (10) also holds for a p-type MOSFET, except that the signs of the respective variables are reversed. Without any lack of generality, our analysis in this section will be based on the n-type MOSFET model; however, the formulation is applicable to p-type MOSFETs as well. It should also be noted that, as long as the source and the drain terminals are symmetric to each other, the expression in (10) holds irrespective of the choice of transistor models such as EKV (Enz, Krummenacher, and Vittoz) [18], ACM (Advanced Compact MOSFET) [19], etc. or operating regimes, i.e. weak-inversion, moderate-inversion or strong-inversion, or process nodes viz. MOSFET, FinFET, etc. The function $f(\cdot, \cdot)$ always satisfies the properties similar to $g(\cdot)$, and can be listed as:

- $f(0, 0) = 0$ and $f(\cdot, \cdot)$ is always positive or $f(\cdot, \cdot) \geq 0$, by construction.
- $f'(\cdot, \cdot) \geq 0$
- $f(\cdot, \cdot)$ is monotonic. For $V_{g1} > V_{g2}$, $f(V_{g1}, V_s) > f(V_{g2}, V_s)$ and for $V_{s1} > V_{s2}$, $f(V_g, V_{s1}) < f(V_g, V_{s2})$.

Thus, $f(\cdot, \cdot)$ and hence the FETs could be used to implement S-AC function as follows:

Given an input matrix $X \in \mathbb{R}^{N \times S}$ where $\mathbf{x}_i \in \mathbb{R}^N$, $\forall i = 1,..,N$ is the input vector and $\mathbf{x}_j \in \mathbb{R}^S$, $\forall j = 1,..,S$ is the

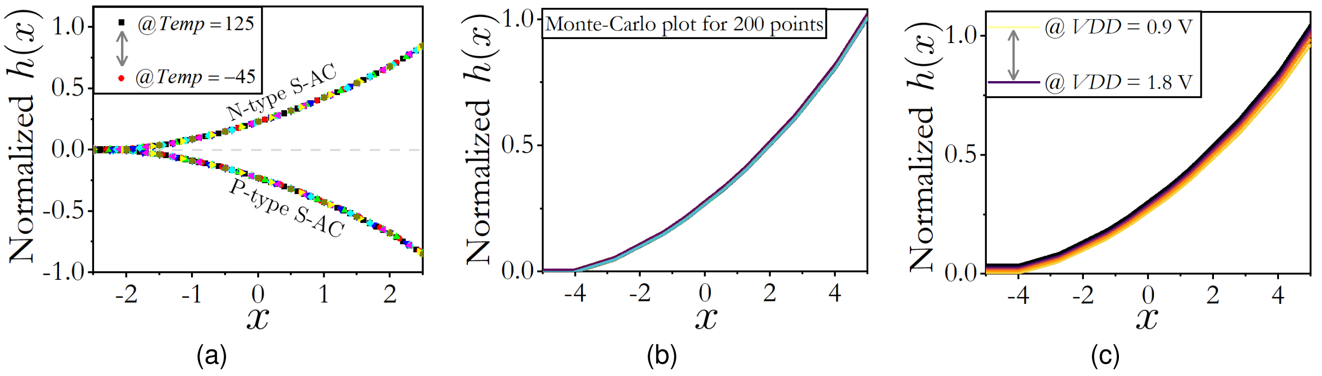

Fig. 4. Basic S-AC functions implemented by the S-AC circuit in a 180nm process node when a single input $x$ is varied: (a) when the temperature is varied from $-45°C$ to $125°C$ in 180nm; (b) in the presence of device mismatch (upto 5% mismatch) - the plot shown here is for N-type S-AC circuit; (c) in the presence of power supply voltage variation from $0.9V$ to $1.8V$. The output current $h(x)$ shown in the plots have been normalized with respect to $I_{max}$, where $I_{max}$ is the maximum current for each of the biasing regime.

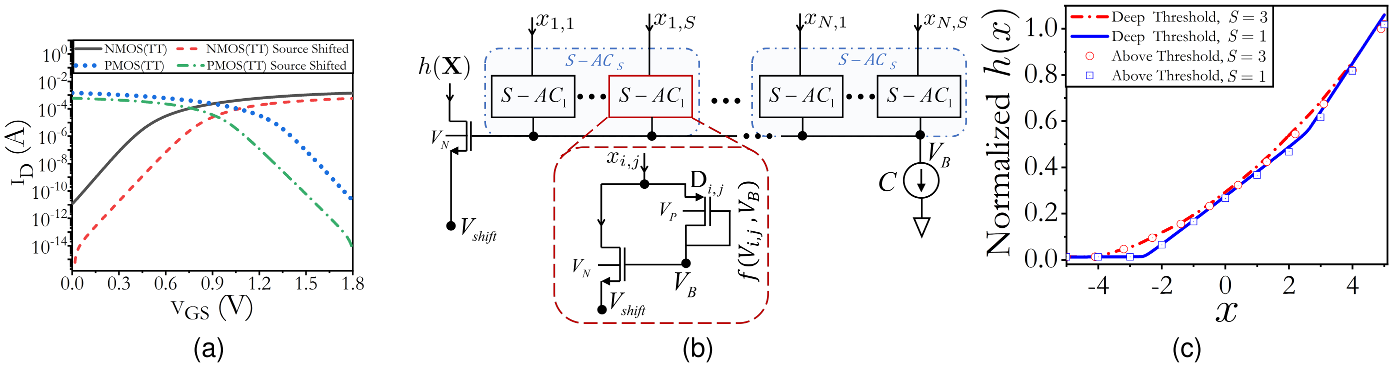

Fig. 5. Results for S-AC circuit when operating in deep-threshold regime in a 180nm process: (a) MOSFET I-V characteristics showing the effect of source shifting to lower the operating current into the diode leakage regime; (b) MOSFET implementation of the N-type S-AC circuit capable of operating in deep-threshold regime; (c) Normalized output current response of the source-shifted S-AC circuit for $S = 1, 3$.

number of splines, the basic shape of S-AC $h : \mathbb{R}^{N\times S} \to \mathbb{R}$ can be computed as a solution to the equation $h(\mathrm{X}) = f(V_B, 0)$ where the variable $V_B$ is the solution to:

$$\sum_{i=1}^{N}\sum_{j=1}^{S} f(V_{i,j}, V_B) = C, \forall i = 1, .., N, \forall j = 1, .., S \quad (11)$$

$$f(V_B, 0) - f(V_B, V_{i,j}) + f(V_{i,j}, V_B) = x_{i,j} \quad (12)$$

Here, $C$ is a hyper-parameter and $V_{i,j}$ is an internal variable. Equation (12) can also be written as

$$f(V_{i,j}, V_B) - f(-V_{i,j}, V_B) = x_{i,j} - f(V_B, 0) \quad (13)$$

A closer look at the derivatives of $f'(V_{i,j} - V_B)$ and $f'(-V_{i,j} + V_B)$ show that these are intersecting curves with respect to $x_{i,j}$. The point of intersection defines a unique solution. Further, the left-hand side of (14) is monotonic by definition of $f'(\cdot, \cdot)$, thus when $\frac{dx_{i,j}}{dV_{i,j}} \geq 0$, $f(V_B)$ is the solution.

$$\frac{dx_{i,j}}{dV_{i,j}} - f'(V_B, 0) = f'(V_{i,j} - V_B) + f'(-V_{i,j} + V_B) \quad (14)$$

Equations (11) and (12) can be implemented using CMOS circuits as shown in Fig. 2b. Here, $x_{i,j}$ is the input current for the $i^{th}$ input and the $j^{th}$ spline and $h(\mathrm{X})$ is the output current. $V_{i,j}$ and $V_B$ are the voltages across the $N_{i,j}^{th}$ transistor, $C$ is a constant current and $D_{i,j}$ denotes diode elements (Schottky, MOS diode or any other). In addition, the S-AC circuit implementation with the GMP algorithm satisfies additional constraints such that the inputs must add up to a constant $C$. Thus, based on Fig. 2b, the current through the diode $D_{i,j}$ as a function of the voltages can be given by $f(V_{i,j} - V_B)$ where all the currents sum up to $C$. Then using KCL at node $V_B$, (11) can be obtained while the current across diode $D_{i,j}$ gives (12). A similar operation can be obtained in other quadrant using the PMOS variant shown in Fig. 2c.

## B. Process and Temperature Scalability of Basic S-AC Circuit

We first show that the basic S-AC shape-function implemented by circuits in Fig. 2b and Fig. 2c are robust to changes in biasing conditions and operating temperature. Fig. 3a shows the proto-shape $h(x)$ obtained using the circuits shown in Fig. 2b (N-type S-AC) and Fig. 2c (P-type S-AC) for spline count $S = 1$. Similar results are shown for spline count $S = 4$ in Fig. 3b. It can be noted that with the increase in the number of splines, the approximation accuracy increases while the basic S-AC function remains scalable across process-technology nodes. Fig. 3c shows the similar shape-function (for $N = 1$ and $S = 3$ ) obtained for different MOSFET

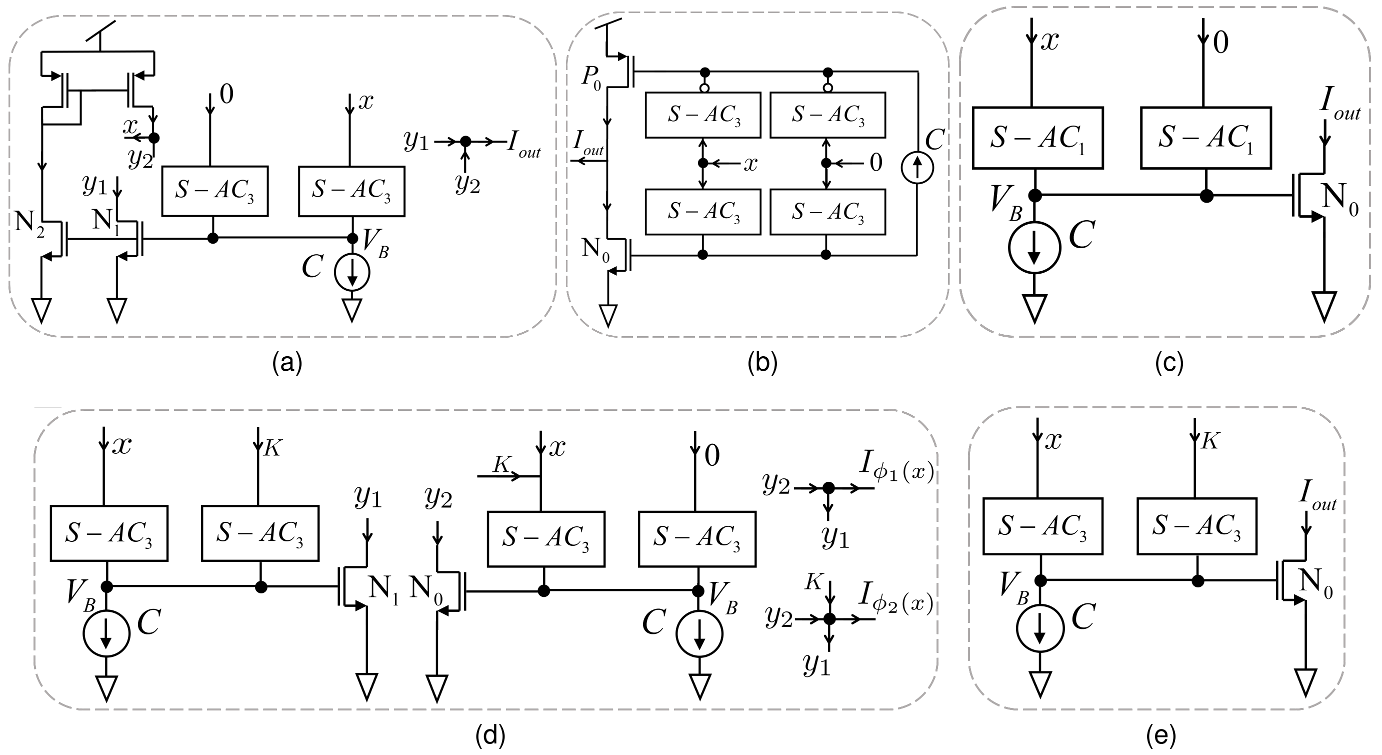

Fig. 6. Implementation of S-AC based analog activation standard cells for machine learning applications: (a) $\cosh(\cdot)$; (b) $\sinh(\cdot)$; (c) ReLU; (d) Compressive non-linearity; $\phi_1(\cdot)$ similar to $\tanh(\cdot)$ and $\phi_2(\cdot)$ similar to Sigmoid functions respectively; (e) Soft-Plus. Here $K$ represents a constant current.

biasing regimes, i.e., WI, MI, and SI biasing regimes which correspond to different functions $f(\cdot,\cdot)$ in (11)-(12) in 180nm node. The plots show that the basic shape of S-AC remains robust to the biasing condition and is constrained within a well-defined ”margin”. This margin is determined by the design parameter $S$ and the inherent feedback from the hyper-parameter $C$. The number of splines $S$ provides control over the precision and accuracy while $C$ is the constraint that is satisfied by (6) or its circuit equivalent given by (11). Fig. 3d shows similar response plots for 7nm process node. Fig. 4a shows the effect of temperature on the shape function. It can be observed that the shape-function is almost immune to changes in temperature. Fig. 4b shows the effect of Monte Carlo analysis on the basic shape of N-type S-AC, while Fig. 4c shows the effect of power supply variation in WI regime. It can be observed that the shape remains preserved despite variations in analysis.

### C. Deep-threshold Operation of Basic S-AC Circuit

To exploit the complete available current range of transistors in planar CMOS, two approaches can be followed. First, the $V_{GS}$ could be made negative; second, the threshold voltage, $V_{T0}$, could be increased in a way that the channel inversion occurs at higher $V_{GS}$ voltages [20], [21], [22]. Therefore, to bias the transistors at smaller currents, $V_{GS}$ should be biased to the reverse direction ($V_{GS} < 0V$ for NMOS and $V_{GS} > 0V$ for PMOS), which is defined as the deep sub-threshold region. For this approach, the source voltage shifting technique can be used. By shifting the source voltage slightly higher than the lowest potential ($V_{SS}$), the gate voltage can reach down to the lowest potential ($V_{SS}$). Fig. 5a shows the $I_D$ vs. $V_{GS}$ characteristic plot in log scale for source shifted MOS transistors. The lowest value of current found with the source shifting technique was 1.97fA for NMOS and 3.19fA for PMOS in CMOS 180nm process node. On analyzing Fig. 5a one can see that the source voltage shifting can exploit the complete available current range (down to the diffusion diodes reverse leakage currents).

For the second approach (so-called channel conduction manipulating technique), the body terminals of transistors are connected to the highest potential ($V_{DD}$). This will prevent the channel inversion from taking place at low $V_{GS}$ voltages which, along with source shifting, further lowers the lowest level of the operating current of the circuit. We used a technique that combines a fixed source shifted voltage with the channel conduction manipulation technique. Fig. 5b shows the circuit implementation of equations (11)-(12) using the above two approaches combined, where source voltage modulation along with channel conduction manipulation techniques were used to shift the operation in femto-ampere (fA). Fig. 5c shows the response of the basic S-AC circuit operating in the deep-threshold at current levels down to femto-amperes (fA). The results show that if biased properly, S-AC can operate at ultra-low current levels and yet its characteristics are maintained.

## IV. Synthesis of Analog Computing Modules using Basic Proto-shapes

S-AC circuits shown in Fig. 2b and Fig. 2c can be used to synthesize mathematical functions required to perform com-

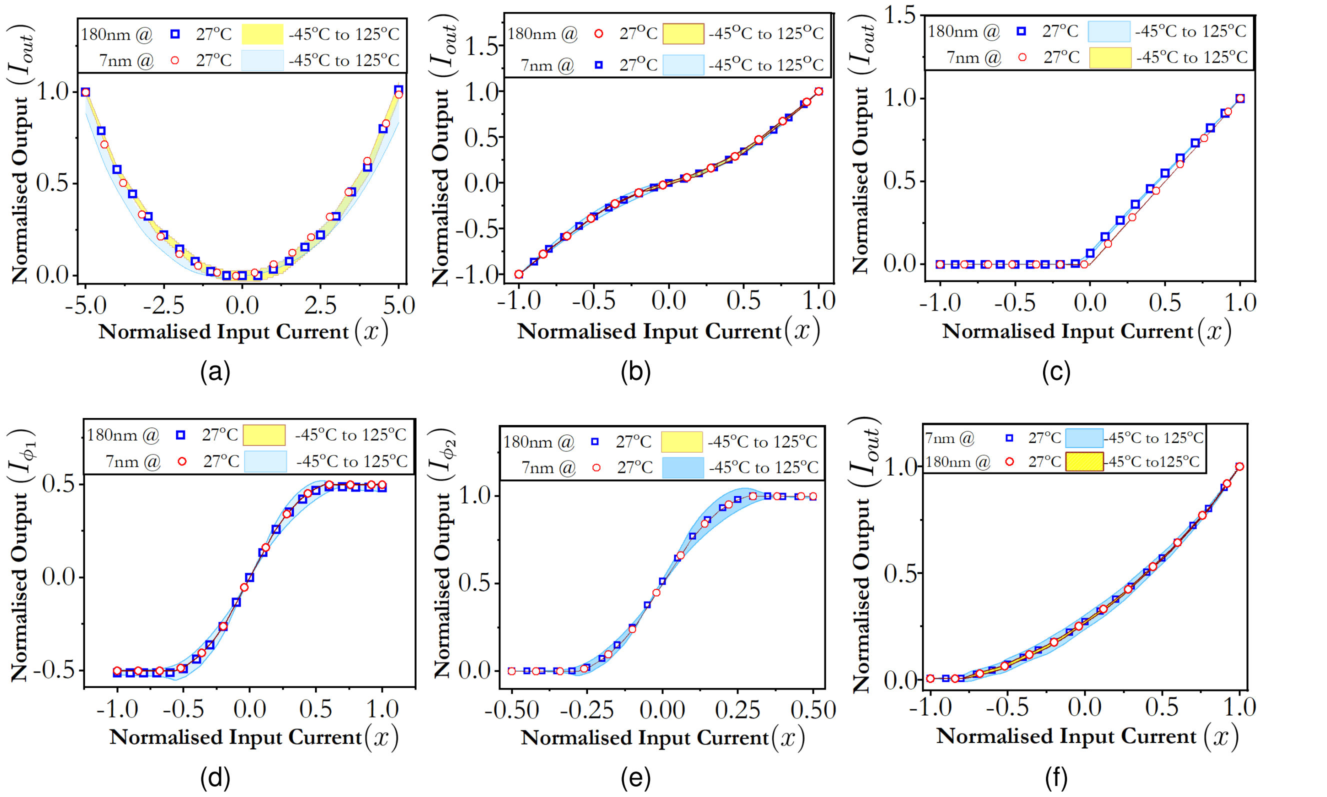

Fig. 7. Simulated output corresponding to the S-AC standard cells shown in Fig. 6: (a) $\cosh(\cdot)$; (b) $sinh(\cdot)$; (c) ReLU; (d) Compressive non-linearity $\phi_1(\cdot)$ equivalent to $\tanh(\cdot)$; (e) Compressive non-linearity $\phi_2(\cdot)$ equivalent to Sigmoid; (f) Soft-Plus at FinFET (7nm) and CMOS (180nm) process nodes.

plex machine learning tasks which can then be used as analog standard cells. As a proof of concept, we show the S-AC-based implementation of various machine learning computations such as activation functions, energy-efficient multiplication, Winner-Takes-all (WTA), N-of-M encoder, SoftArgMax, and Max circuits. It can be noted that the proto-shape $h(x)$ can be mapped to *exponential* for $e^x$ ranging from $[-\infty, 1)$. Within this defined range i.e. $\left|\frac{\partial h}{\partial x}\right| < 1$, if (7) and (8) are satisfied, then complex functions such as hyperbolic functions (*Cosine, Sine*) and other such functions can also be constructed. The following text has a detailed formulation and S-AC based implementation of each of these circuits in both 180nm and 7nm process technology nodes.

## A. Cosine-Hyperbolic

The $\cosh(\cdot)$ function can be constructed using N-type S-AC standard cells for $S = 3$ as shown in Fig. 6a. The $\cosh(\cdot)$ function is given by:

$$\cosh(x) = \frac{e^x + e^{-x}}{2} \tag{15}$$

Now, if in Fig. 2a, the response of one S-AC unit is $h(x) \cong \frac{e^x}{2}$, then by tuning the offsets $O_1, \cdots, O_3$, we can get $\cosh(\cdot)$. In terms of S-AC computation, (15) can be written as

$$\cosh(x) = h(x) + h(-x) \tag{16}$$

Thus the $\cosh(\cdot)$ function is the addition (KCL) of the response of S-AC ($y_1$ in Fig. 6a) and its vertically flipped response (shown by $y_2$ in Fig. 6a). Fig. 7a shows characteristics response $I_{out}$ obtained using Fig. 6a across different process nodes and different temperatures for the same values of offset.

## B. Sine-Hyperbolic

The $\sinh(\cdot)$ function can be constructed using both P- and N-type S-AC standard cells for $S = 3$ as shown in Fig. 6b. The $\sinh(\cdot)$ function is given by:

$$\sinh(x) = \frac{e^x - e^{-x}}{2} \tag{17}$$

Similar to *Cosine* if the response of S-AC unit is $h(x) \cong \frac{e^x}{2}$, then (17) can be written in terms of S-AC computation as

$$\sinh(x) = h(x) - h(-x) \tag{18}$$

Fig. 7b shows characteristics response $I_{out}$ obtained using Fig. 6b across different process nodes and temperatures.

## C. ReLU

ReLU can be implemented using two S-AC units as shown in Fig. 6c. Fig. 6c implements the function given by

$$I_{out} = \begin{cases} \max(0, x) & , C \to 0 \\ \max(0, C - x) & , else \end{cases} \tag{19}$$

Fig. 7c shows the characteristic response of ReLU obtained from the circuit in Fig. 6c at different process nodes and temperatures. Fig. 8a and Fig. 8d show the Monte Carlo plot of S-AC *ReLU* in 180nm and 7nm process technology nodes.

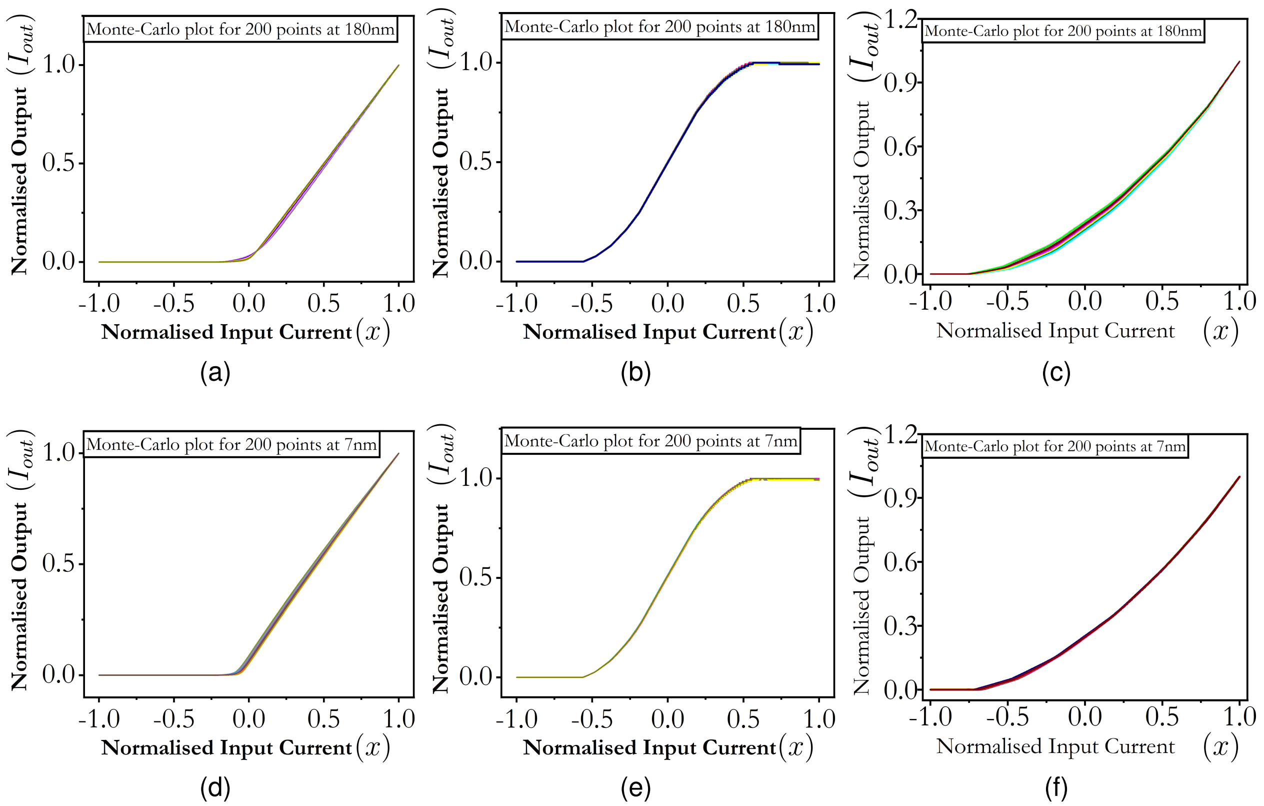

Fig. 8. **Monte Carlo plots along with Maximum % Deviation for S-AC based analog activation functions shown in Fig. 6 when operated in Weak Inversion regimes: (a) ReLU in 180nm,** $(3.11\%)$**; (c) Soft-plus in 180nm,** $(2.44\%)$**; (b) Sigmoid in 180nm,** $(7.31\%)$**; (d) ReLU in 7nm,** $(4.14\%)$**; (e) Sigmoid in 7nm,** $(0.91\%)$**; (f) Soft-plus in 7nm,** $(3.86\%)$**.**

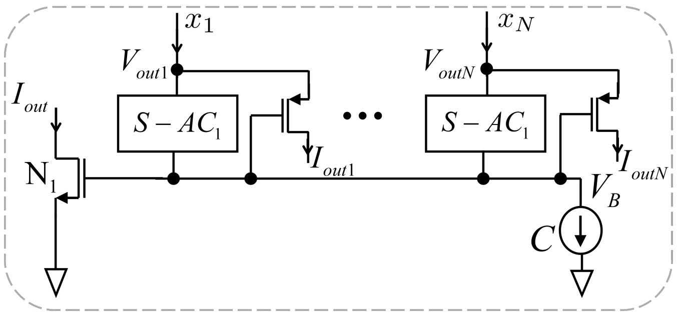

Fig. 9. Implementation of $N$-input winner takes all standard cells using S-AC units. This circuit can be tuned to function as a soft-WTA and Max circuit.

## D. Compressive non-linearity

We formulate the compressive non-linearity function $\phi_1(\cdot)$ given by

$$\phi_1(x) = \log \frac{1+e^{x+K}}{e^x + e^K} \tag{20}$$

where $K$ is a constant. This compressive function in (20) can then be used to emulate $\tanh(\cdot)$ and sigmoid functions. Equation (20) when mapped to S-AC computation can be written as:

$$\phi_1(\cdot) = h(0, x+K) - h(x, K) \tag{21}$$

The circuit for emulating $\phi_1(\cdot)$ is shown in Fig. 6d. We show that the characteristics response $I_{\phi 1}$ shown in Fig. 7d can be used to emulate response equivalent to $\tanh(\cdot)$ function.

## E. Sigmoid

The sigmoid equivalent characteristic plot can be obtained from a shifted version of the $\phi_1(\cdot)$ function. Thus adding a constant current $K$ to $I_{\phi 1}$ gives a sigmoid equivalent response and is shown in Fig. 6d. We show that the characteristics response $I_{\phi 2}$ shown in Fig. 7e can be used to emulate a response equivalent to the $sigmoid$ function. The Monte Carlo plot of S-AC *sigmoid* in both 180nnm and 7nm process node is shown in Fig. 8b and Fig. 8e respectively.

## F. Soft-Plus

The soft plus activation function can be obtained using two $S = 3$, S-AC units. The circuit for emulating the $Soft\text{-}Plus$ response is shown in Fig. 6e. We show that the characteristics response $I_{out}$ shown in Fig. 7f are robust across different process nodes and different temperatures. Fig. 8c and Fig. 8f show the Monte Carlo plot of S-AC *soft-plus* in 180nm and 7nm process technology node.

## G. S-AC based Winner-Take-All

A winner-take-all (WTA) circuit is designed to emulate the $max(\cdot)$ function. The proposed S-AC-based WTA circuit

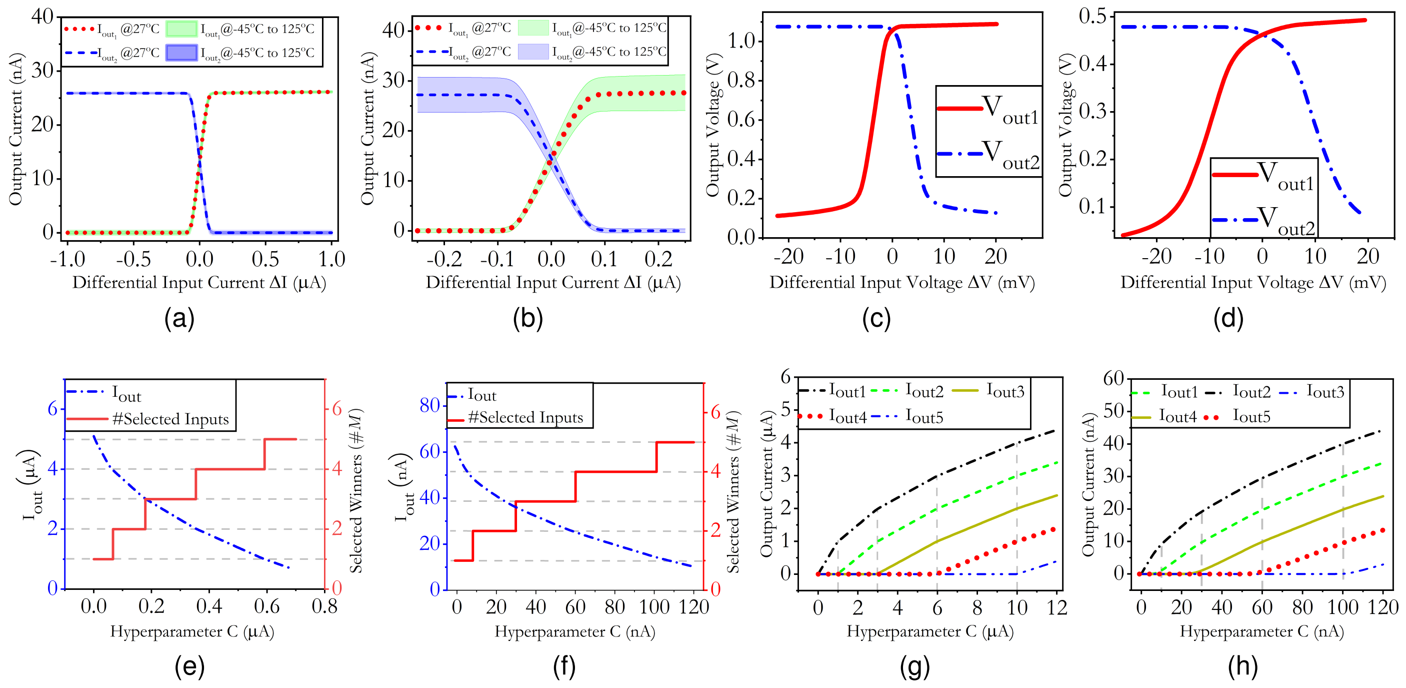

Fig. 10. Simulated output of a two input S-AC based WTA standard cell shown in Fig. 9 showing: (a) current output ($I_{out1}$ and $I_{out2}$) versus the differential input current for 180nm process node and (b) for 7nm process node; (c) Voltage output ($V_{out1}$ and $V_{out2}$) versus the differential input voltage for 180nm process node and (d) for 7nm process node. The simulated output of a five-input S-AC WTA standard cell showing (e) $M$ selected winners as a function of hyperparameter $C$ for 180nm process node and (f) for 7nm process node where $I_{out}$ is the contribution of $M$ selected winners in the output. Simulated output of five-input S-AC WTA standard cell showing outputs $[I_{out1}, \cdots, I_{out5}]$ as a function of hyperparameter $C$ at (g) 180nm and (h) 7nm process node where responses of Fig. 10 are obtained for the inputs $[x_1, x_2, x_3, x_4, x_5] = [\alpha, 2\alpha, 3\alpha, 4\alpha, 5\alpha]$ where $\alpha = 1\mu A$ for 180nm and $10nA$ for 7nm.

is shown in Fig. 9. It is modular, i.e., it can be extended for N inputs like the original circuit proposed by Lazzaro et al. (1989) [23]. Fig. 10a and Fig. 10b show the current characteristics plot of two-input S-AC based WTA at 180nm and 7nm for an input differential current. It can be observed that when the differential current is 0, the output currents are equal. The corresponding voltage outputs for both the process nodes are shown in Fig. 10c and Fig. 10d respectively.

## H. S-AC based N-of-M Encoder

The N-of-M encoder extends the computational capabilities of the standard WTA circuit and has found profound importance in sparsely distributed memory [24]. N-of-M encoder allows the user to obtain the max current that takes into account the influence due to the contribution of top M winners out of N inputs ($M/N$). For the specific case of $M = 1$, the N-of-M encoder behaves like a simple WTA circuit. Fig. 10e shows the response of the five-input S-AC-based WTA circuit shown in Fig. 9 as a function of hyper-parameter $C$. It can be noted that with the increase in hyper-parameter value, the output current $I_{out}$ decreases and is the result of more than one winner. Using (11) for $S = 1$, $I_{out}$ is given by

$$I_{out} = \frac{\sum_{i=1}^{M} x_i - C}{M} \tag{22}$$

where $M$ is the number of winners. Similar results for 7nm are shown in Fig. 10f. It can be analyzed that depending on the hyper-parameter $C$, the circuit can select the top $M$ winners.

## I. S-AC based SoftArgmax

In machine learning, SoftArgmax offers an improvement over Argmax to support backpropagation and gradient operation. The S-AC-based WTA circuit can be configured to implement SoftArgmax. Fig. 10g and Fig. 10h show the response curve of outputs $I_{out1}, \cdots, I_{out5}$ for the variation in hyper-parameter $C$ for 180nm and 7nm respectively. It can be observed that with the increase in hyper-parameter, outputs corresponding to the maximum input along with other inputs are activated and can be given by

$$I_{out_i} = x_i - C, \qquad \forall x_i > C \tag{23}$$

## J. S-AC based Max Circuit

The S-AC based *winner-take-all* circuit can also be configured to select the maximum input among the given set of $N$ inputs. For hyper-parameter, $C \to 0$, the circuit starts behaving as a max input selector.

## K. S-AC based Four Quadrant Multiplier

The four-quadrant multiplier is proposed in [25]. Owing to the Lipchitz behavior of the S-AC function, the design is formulated such that the multiplication satisfies the Lipchitz condition. Fig. 11 shows the S-AC cell-based implementation of a multiplier circuit. Consider the following equation, where $y$ is given by

$$\begin{aligned} y = h(C + w + C + x) - h(C + w + C - x) + \dots \\ h(C - w + C - x) - h(C - w + C + x) \end{aligned} \tag{24}$$

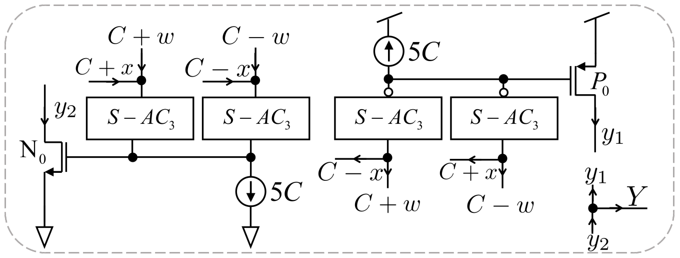

Fig. 11. Implementation of a four-quadrant multiplier using P-type and N-type S-AC circuits.

The goal is to implement scalar multiplication between two variables $x$ and $w$. Here $x, w, y \in \mathbb{R}$, $h$ is a non-linear monotonic function and $C$ is a hyperparameter. If we write the Taylor expansion of $h(x)$ around $w$, we get

$$h(C+w+C+x) = h(0) + \frac{h'(0)}{1!}(C+w+C+x)\dots + \frac{h''(0)}{2!}(C+w+C+x)^2 + \dots \quad (25)$$

$$h(C+w+C-x) = h(0) + \frac{h'(0)}{1!}(C+w+C-x)\dots + \frac{h''(0)}{2!}(C+w+C-x)^2 + \dots \quad (26)$$

$$h(C-w+C-x) = h(0) + \frac{h'(0)}{1!}(C-w+C-x)\dots + \frac{h''(0)}{2!}(C-w+C-x)^2 + \dots \quad (27)$$

$$h(C-w+C+x) = h(0) + \frac{h'(0)}{1!}(C-w+C+x)\dots + \frac{h''(0)}{2!}(C-w+C+x)^2 + \dots \quad (28)$$

Substituting (25) - (28) in (24) and ignoring the higher order terms, we get,

$$y \cong 4h''(0)x \times w \quad (29)$$

where $4h''(0)$ is a scaling factor. For $h(\cdot)$ assumed to be a non-linear function such that $h''(0) \neq 0$, (29) simplifies to

$$y \cong x \times w \quad (30)$$

(24) shows that the multiplication is then reduced to addition and subtraction operations, thus altering the use of a bulky multiplier.

In Fig. 12a, we show that the behavior of the four-quadrant S-AC multiplier is scalable across the process technology nodes. Fig. 12b shows the response of the four-quadrant S-AC multiplier at different operating regimes for the 7nm process node. A similar response can be observed in Fig. 12c at the 180nm process node. It can be analyzed from Fig. 12b and Fig. 12c that the shape of the multiplier characteristic curve remains preserved when the circuit operation moves from WI to SI.

TABLE I
OPERATION PERFORMANCE PARAMETER FOR S-AC SYSTEM

| **Parameter** (@ $S = 1$) | **Technology and Operating Regimes** | | | | | |
|---|---|---|---|---|---|---|
| | **CMOS 180nm** | | | **FinFET 7nm** | | |
| | **SI** | **MI** | **WI** | **SI** | **MI** | **WI** |
| **Computational Efficiency (TOPS/mm2)** | 5 | 0.082 | 1.83e-4 | 5100 | 3460 | 50.28 |
| **Power Efficiency (TOPS/W)** | 13.49 | 27.74 | 73.36 | 69.4 | 2.9e4 | 3.6e5 |
| **System Efficiency (pJ/MAC)** | 0.19 | 0.24 | 0.67 | 0.03 | 0.14e-3 | 6.12e-3 |

## L. *Performance and Trade-off Analysis*

This section shows the analysis of the S-AC computational blocks at different process technology nodes, viz. 180nm and 7nm and at different operating conditions. The inter-dependence of power, throughput, accuracy and area trade-off for designing a performance scalable system is explained.

*1) Power & Operation Performance Analysis:* Fig. 13a shows the plot of the average power consumption plot with the increase in the number of S-AC units for different technology nodes and at different biasing regimes. It can be observed that with the increase in S-AC units in parallel, power consumption increases for a fixed value of hyperparameter $C$.

Table I shows the operational performance parameters [26], [27] of the S-AC analog cells at different operating regimes and at different process nodes. We here computed the peak capabilities of the S-AC architectures for some of the widely used performance metrics such as *Computational Efficiency-Power Efficiency, System Efficiency*. It can be observed that the *Computational Efficiency* is highest in SI for planar CMOS and Finfet node. It also shows that the best *System Efficiency* and *Power Efficiency* and lowest *pJ/MAC* operation can be obtained in weak inversion.

*2) Mismatch and Process Variation Analysis:* In the sub-micron technology nodes, the threshold voltage ($V_{T0}$) and current factor ($\beta$) differences are the dominant sources underlying the drain-source current or gate-source voltage mismatch for a matched pair of MOS transistors [9], [28]. These mismatches affects the speed, accuracy, and other performance parameters of analog circuits. Fig. 13b shows the variation of output current due to variation in the *Fin* count and overdrive voltage in the 7nm FinFET node. It can be observed that the variations are within 5%. Similar variations can be observed in Fig. 13c for planar CMOS for change in area and over-drive voltage.

*3) SNR Analysis:* In a standard analog system (generally amplifiers) the total noise (input noise $n_{in}$, and the noise contributed by the circuit itself $n_{ckt}$) gets multiplied by the system gain and appears at the output. Thus resulting in no improvement in the signal-to-noise ratio (SNR). S-AC architectures exploit the inherent parallelism in the structure to overcome this limitation. In S-AC parallel current-mode configuration, because we are summing two uncorrelated noise sources, the overall noise increases as $\sqrt{2}$, while the correlated input signal amplitude increases by 2. Mathematically, for a single S-AC block having input signal $x_1$ and gain $G_1$, the total combined input signal is given by $x_1 + n_{in_1}$ . The output

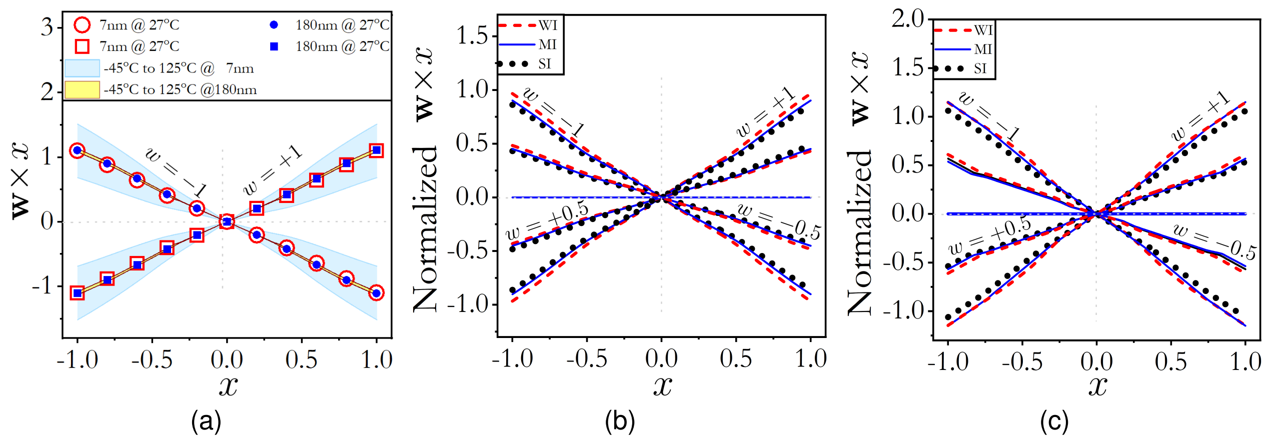
Fig. 12. Simulated multiplier characteristics obtained for $S = 3$, (a) at different process nodes and across temperature; (b) at 7nm process node for different operating regimes; (c) at 180nm process node for different operating regimes.

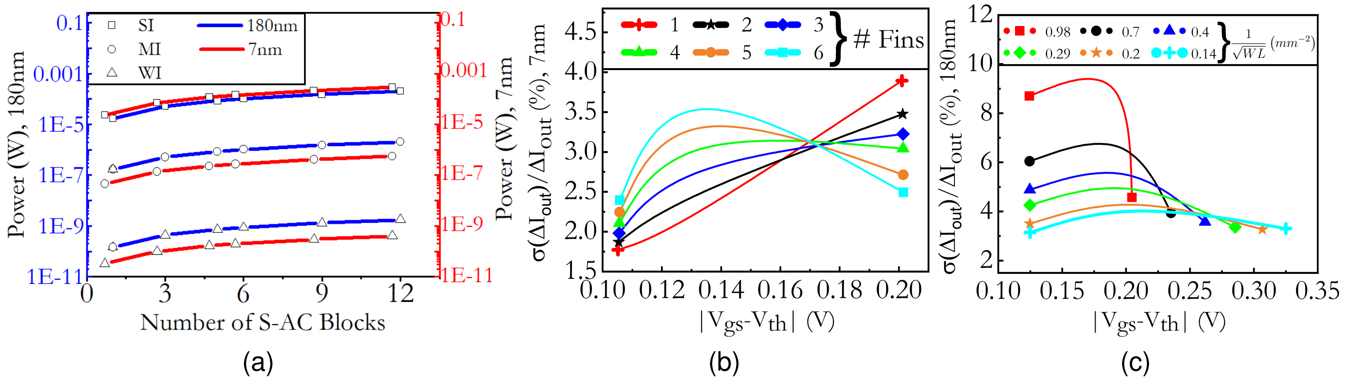
Fig. 13. Simulations showing (a) average power consumption at different process technology nodes as the function of $S$ (number of inputs $N = 1$); (b) Standard deviation of output current $h(\mathbf{x})$ as a function of change in Fins count and gate-source voltage for 7nm; (c) change in aspect ratio and overdrive voltage for 180nm.

includes the output signal $Z$ plus the total output noise $n_{out_1}$.

$$n_{out_1} = n_{in_1} \times G_1 + n_{ckt_1} \tag{31}$$
$$Z = x_{in_1} \times G_1 \tag{32}$$

The SNR is calculated by dividing the output RMS signal power by the output RMS noise power. This comes out to be

$$SNR_1 = \frac{Z}{n_{out_1}} = \frac{(x_{in_1} \times G_1)^2}{(n_{in_1} \times G_1)^2 + n_{ckt1}^2} \tag{33}$$

Assuming the external noise input power is minimal (for simplicity), then (32) reduces to

$$SNR_1 = \frac{(x_{in_1} \times G_1)^2}{n_{ckt1}^2} \tag{34}$$

Now, adding a second S-AC block in parallel increases the RMS signal power by 2×. However, it only increases the RMS circuit noise by $\sqrt{2}$ because the circuit adds uncorrelated noise. So instead of the noise doubling, we obtain a noise of $\sqrt{2} \times n_{ckt}$. The SNR equation for two interconnected S-AC blocks (assuming the composite gain remains almost the same) becomes

$$SNR_2 = \frac{(2x_{in} \times G)^2}{(2n_{in} \times G)^2 + \left[\sqrt{2}\left((n_{ckt_1})^2 + (n_{ckt_1})^2\right)\right]} \tag{35}$$

For $n_{ckt_1} = n_{ckt_2} = n_{ckt}$ and assuming the external input noise power is minimal (35) changes to

$$SNR_2 = \frac{(x_{in} \times G)^2}{\left(0.5(n_{ckt})^2\right)} \tag{36}$$

On comparing (34) and (36) one can analyze that for each increase in the number of connected S-AC blocks in parallel, the circuit SNR increases by twice.

*4) S-AC Area and Power Saving Analysis:* S-AC design offers a range of trade-offs between accuracy, area, and power benefits by changing the design parameter ($S$). The choice of $S$ is determined based on the application requirements. Theoretically, the number of splines selected can vary from 1 to $S$, therefore offering a wide range of trade-offs to choose from. We here evaluate the performance of the S-AC multiplier for input dimension $N = 2$ as a function of $S$. The results summarized in Table II show the maximum error, average absolute

TABLE II
S-AC MULTIPLIER PARAMETERS FOR DIFFERENT $S$

| Error (%) | S=1 | S=2 | S=3 |
|---|---|---|---|
| Max Error | 50 | 33.333 | 11.111 |
| Average abs error | 22.29 | 9.31 | 3.66 |
| Error bias | 22.29 | -9.31 | -2.57 |
| Standard Deviation | 11.78% | 5.89% | 1.68% |
| Approx. Area Savings* | 68.7% | 49.9% | 31.3% |
| Approx. Power Savings* | 68.4% | 52.7% | 37.2% |

* Compared to full-precision multiplier [29].

TABLE III
ENERGY/OPERATION AND MEAN ABSOLUTE DEVIATION

| Operation | Err* | Process Node | Energy per Operation (fJ) | | |
|---|---|---|---|---|---|
| | | | WI | MI | SI |
| Cosh | 0.0599 | 180nm | 40.86 | 108.12 | 222 |
| | | 7nm | 0.0196 | 0.612 | 23.1 |
| Sinh | 0.0098 | 180nm | 81.72 | 216.24 | 444 |
| | | 7nm | 0.0392 | 1.224 | 46.2 |
| ReLU | 0.0337 | 180nm | 11 | 24.42 | 75.94 |
| | | 7nm | 0.0035 | 0.101 | 2.97 |
| Compressive Non-Linearity | 0.0054 | 180nm | 81.72 | 216.24 | 444 |
| | | 7nm | 0.0392 | 1.224 | 46.2 |
| Soft-plus | 0.0321 | 180nm | 40.86 | 108.12 | 222 |
| | | 7nm | 0.0196 | 0.612 | 23.1 |
| WTA (N-Input) | 0.1760 | 180nm | N × 6.81 | N × 18.02 | N × 37 |
| | | 7nm | N × 0.0033 | N ×0.102 | N × 3.85 |
| Multiply/ (Divide) | 0.0139 | 180nm | 190 | 240 | 670 |
| | | 7nm | 0.018 | 0.42 | 90 |

$^*Err = MAX_{\forall x}\,|Mean\;Absolute\;Deviation|$ between 180nm and 7nm process node at room temperature.

error, error bias, and standard deviation. It can be analyzed that with the increase in the value of $S$, all error metrics decrease. Furthermore, the errors reduce to roughly half for each increase in the value of $S$. Table II shows the design savings offered by S-AC methodology when compared with a similar full-precision state-of-the-art multiplier implemented in [30]. As expected, the design area and average power consumption increase as a factor of $S$. As analyzed, significant savings can be achieved while introducing insignificant amounts of error. As an example, with an average absolute error of 3.66%, $S = 3$ offers up to 31.3% in area savings and up to 37.2% in power savings.

*5) Task-Energy Efficiency Analysis:* Table III summarizes the energy requirement and Mean-Deviation for the S-AC block for implementing basic computational operations at different operating regimes and at different process nodes. We reported the maximum mean absolute deviation obtained from the resultant functional shapes at 180nm and 7nm when similar architecture was used in both process nodes. Table III also summarizes the Energy/Operation obtained at different operating regimes and at different process nodes for different S-AC computations. The least energy consumption is seen in the WI regime and the worst in the SI regime.

## V. CASE STUDY: S-AC BASED NEURAL NETWORK

In this section we show the design flow optimization of a S-AC based neural network synthesized using S-AC standard cells. Note that the approach presented here can also be generalized to implement other machine learning architectures.

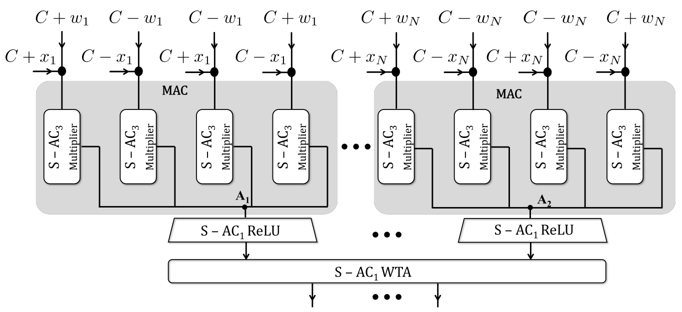

Fig. 14. System-level architecture of a 3-layer neural network using S-AC standard cells.

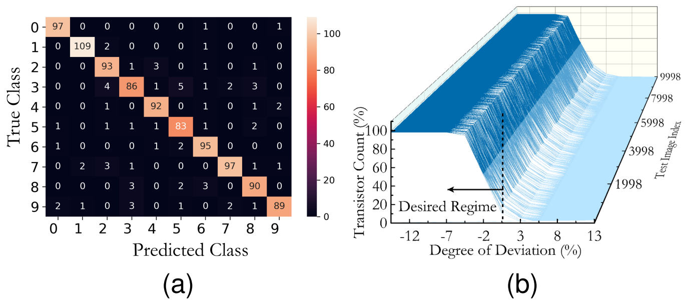

Fig. 15. (a) Confusion plot showing the distribution of randomly chosen 1000 test images from MNIST dataset; (b) Distribution showing the percentage of transistors that have deviated from the desired operating condition for the MNIST test-set.

### A. Algorithm to S-AC Hardware Mapping

Consider a vector $\mathbf{x} \in \mathbb{R}^N$ where the output $y$ for a standard MLP [31] is given as,

$$y = \varphi\left(\eta\left(\mathbf{x}\right)\right) \tag{37}$$

where $\varphi\left(\cdot\right)$ is any non-linear function be it tanh($\cdot$), sigmoid, ReLU, etc and $\eta\left(\mathbf{x}\right)$ be the decision function given by

$$\eta(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b \tag{38}$$

where $\mathbf{w} \in \mathbb{R}^N$ is the trained weight vector, $\mathbf{x} \in \mathbb{R}^N$ is the input vector, $b \in \mathbb{R}$ is the bias and the function $\eta : \mathbb{R}^N \rightarrow \mathbb{R}$ is the decision function. For $\{x, y\} \in \mathbb{R}$, the decision function $\eta(x)$ can then be rewritten as

$$\eta(x) = w \cdot x + \mathrm{b} \tag{39}$$

Using equation (24) and (30) in (39) the decision function gets mapped to S-AC based form as

$$\begin{aligned} \eta(x) = h(2C + w + x) - h(2C + w - x) + \cdots \\ \cdots h(2C - w - x) - h(2C - w + x) + b \end{aligned} \tag{40}$$

Equation (40) can be viewed as generic decision function $\eta(\cdot)$ mapped into shape domain. This shape equation can then be easily synthesized using only S-AC analog cells. It can be noted that variable $b$ can be assumed as a constant current added to the dot-product $w \cdot x$ and implemented using KCL without additional circuits. Furthermore, to add non-linearity to this output, function $\varphi\left(\cdot\right)$ can then be mapped to the shape domain using the formulation demonstrated in Section IV.

TABLE IV
Classification Accuracy at Different Operating Regimes & Different Process Nodes

| Dataset | Operating Regime | Classification Accuracy (S/W) | Classification Accuracy (H/W) | |
|---|---|---|---|---|
| | | | 180nm | 7nm |
| XOR | SI | 95 | 95 | 95 |
| | MI | | 94 | 93 |
| | WI | | 93 | 93 |
| AReM | SI | 94 | 93.5 | 93 |
| | MI | | 93 | 94 |
| | WI | | 93 | 93 |
| MNIST | SI | 93 | 92.5 | 92.1 |
| | MI | | 92 | 92 |
| | WI | | 92.2 | 92.1 |

## B. S-AC based Neural Network

Fig. 14 shows the system-level implementation of a S-AC based neural network designed using S-AC based analog standard cells. The network was mapped using the algorithm mapping approach described in Section V-A and designed using S-AC based analog cells described in Section IV. We trained the network using the margin propagation algorithm mentioned in [32] with variation aware training [33].

Table IV summarizes the classification accuracy of synthesized neural networks using S-AC blocks at different process technology nodes and operating regimes. We tested out system classification capabilities on standard Activity Recognition dataset (AReM) [34] dataset where we chose two of the activities as positive cases, i.e., bending and lying activities and utilized One versus all approach for binary classification. We further verified the functionality on the benchmark MNIST dataset [35] of handwritten digits. It can be analyzed from Table IV that the classification accuracy of implemented hardware nearly matches that of software at both 180nm and 7nm implementation and at different operating regimes. This clearly signifies that the design is both process technology scalable and bias scalable. It can be noted that due to time-complexity in running analog SPICE simulations on the EDA tools(where each full-precision simulation run extended for nearly 6 hours), we randomly selected 1000 test images out of the given 10000 test set.

Fig. 15a shows the confusion plot showing the distribution for randomly selected 1000 test images. Furthermore, the implemented MNIST network consisted of 256 input nodes (where each input $28 \times 28$ image dimension was scaled down to $16 \times 16$), 15 hidden nodes, and 10 output nodes for which the baseline accuracy from both vanilla network and margin propagation network with variation aware training in PyTorch was obtained as 93%. As the hidden node count increases, both vanilla and margin propagation network match the state-of-the-art accuracy.Table V shows a comparison of different analog ANN implementations. It can be observed that the Energy/pixel obtained from S-AC network varies as the operating regime shifts from WI to SI, signifying that as per the application, the user can tune the performance specification. The energy/pixel of the S-AC ANN at 180nm for WI is 3$\times$ lesser than [36] while in the SI, it is approximately 14$\times$ more. Fig. 15b shows the percentage deviation of transistors from

TABLE V
Comparison with State-of-the-art Analog ANNs

| | [37] | [38] | [36] | This Work | | | |
|---|---|---|---|---|---|---|---|
| Process (nm) | 130 | 130 | 65 | 7 | | 180 | |
| Supply (V) | 1.2 | - | 1.2 | 0.7 | | 1.2 | |
| Classifier | binary | binary | ANN | ANN | | ANN | |
| Speed (MHz) | 1.3 | 50 | 5 | 6 | | 3.37 | |
| Feature Size | 48 | 81 | 25 | 256 | | 256 | |
| Biasing Regime | - | - | - | WI | SI | WI | SI |
| Accuracy (%) | 90 | 90 | 82 | 92.2 | 92.5 | 92.1 | 92.1 |
| Energy/pixel (pJ) | 11.1$^*$ | 7.8$^*$ | 6.9 | 0.05$^\dagger$ | 3.7$^\dagger$ | 2.3$^\dagger$ | 97.6$^\dagger$ |

$^*$ Only for first layer implemented on-chip;
$^\dagger$ Based on SPICE simulations of hidden and output layers.

the desired biasing regime for the MNIST dataset. It can be found that on average around 8% of the total transistors in the designed system, shifted their operation from their expected operating regime. For instance, when the designed system was biased to operate in weak inversion regime, near about 8% of total transistors shifted their operation to moderate inversion regime while for the system biased in moderate inversion regime, a similar percentage of transistors shifted their operation to strong inversion regime. However, despite this change in operating condition, there is no significant drop in accuracy as the classification network leverages the benefits of bias scalability of the S-AC circuit.

## VI. Conclusions

In this work, we proposed a framework for designing analog computing circuits that are process, bias, and temperature scalable. The key differentiator from the previous work [15] is the multi-spline approach that allows the framework to trade-off power/area with accuracy (Fig. 2a). The framework leads to the design of S-AC standard cells whose responses were found to be robust to variations in biasing and temperature and scalable across process nodes similar to digital standard cells. As a result, a S-AC standard cell designed in a 180nm process could be easily used for design in a 7nm process. Thus, the modules can be potentially used for the automated synthesis of large-scale analog processors. While the focus of this work was to design S-AC circuits for machine learning processors, the approach can be generalized to other analog processors as well. As a proof-of-concept, we demonstrated the approach for a 3-layer neural network whose test accuracy remains minimally affected by changes in biasing and process nodes. Even though the results in the paper show excellent agreement between the accuracy of the software simulation and circuit simulation for small-scale neural networks, future work will leverage this framework to synthesize large-scale analog deep neural networks and reconfigurable machine learning processors.

## Appendix A
## Proof: Generalized Margin Propagation Formulation using Multi-Spline Approach

Let us consider a log-sum-exp function [15] $h_{\log} : \mathbb{R}^N \rightarrow \mathbb{R}$ given by

$$h_{\log}(\mathbf{x}) = C \cdot \log\left(\sum_{i=1}^{N} e^{\frac{x_i}{C}}\right) \tag{41}$$

where $C$ is a hyper-parameter and $\mathbf{x} \in \mathbb{R}^N$ is a vector with elements $x_i \in \mathbb{R}$. Equation (1) can be written as

$$\sum\nolimits_{i=1}^{N} e^{\frac{x_i - h_{\log}}{C}} = 1 \tag{42}$$

which is an equivalent non-linear constraint satisfaction problem. Let us approximate this non-linear function using linear splines. It can be noted that the same methodology can also be extended to other non-linear functions. As a proof-of-concept, we show for log-sum-exponential. In Fig. 2a, we have shown the plot of the *exponential* function and its approximation using one-spline ($S = 1$) and three-splines ($S = 3$). Here, $Q_1, Q_2, \cdots, Q_S$ are the tangential points and $T_1, T_2, \cdots, T_S$ are the tuning points. The generic line equation for the $j^{th}$ spline where $j \in (1, \cdots, S)$ when approximated using piece-wise-linear lines can be written using point-slope form as

$$\theta_j(x) = e^{Q_j} \cdot x + e^{Q_j}(1 - Q_j) \quad \forall x \geq 0 \tag{43}$$

where $e^{Q_j}$ is the slope and $e^{Q_j}(1 - Q_j)$ is its intercept on the line on the vertical axis. Similarly, for the $(j+1)^{th}$ spline, we can write its line equation as,

$$\theta_{j+1}(x) = e^{Q_{j+1}} \cdot x + e^{Q_{j+1}}(1 - Q_{j+1}) \forall x \geq 0 \tag{44}$$

The tuning points $T_j$ (intercept between $j^{th}$ and $(j+1)^{th}$ spline) can be obtained by equating the line equations of $j^{th}$ and $(j+1)^{th}$ spline at $x_i = T_j$ and can be written as,

$$e^{Q_j} \cdot T_j + e^{Q_j}(1 - Q_j) = e^{Q_{j+1}} \cdot T_j + e^{Q_{j+1}}(1 - Q_{j+1}) \tag{45}$$

Equation (45) can be re-written as

$$T_j = \frac{Q_{j+1} \cdot e^{Q_{j+1}} - Q_j \cdot e^{Q_j}}{e^{Q_{j+1}} - e^{Q_j}} - 1 \tag{46}$$

Then, the approximation of the function $e^x$ using $S$-splines (a case of 3-spline approximation is shown in Fig. 2a) can be written using point-slope form and using (43) and (46) as

$$\begin{aligned} e^x \cong & e^{Q_1}[x - T_1]_+ + \left(e^{Q_2} - e^{Q_1}\right)[x - T_2]_+ + \cdots \\ & \cdots + \left(e^{Q_S} - \cdots - e^{Q_2} - e^{Q_1}\right)[x - T_S]_+ \end{aligned} \tag{47}$$

Equation (47) can then be generalized as

$$e^x \cong \sum\nolimits_{j=1}^{S} \left(e^{Q_j} - \sum\nolimits_{k=1}^{j-1} e^{Q_k}\right)[x - T_j]_+ \tag{48}$$

The above equation (48) shows the approximation of $e^x$ using S-splines. For ease of understanding let us choose a specific case of 3-splines, viz. $S = 3$. Let the tangential points for this case be $Q_1 = \log_e(0.5)$,$Q_2 = \log_e(1)$, $Q_3 = \log_e(2)$. Then by using (46) we get,

$$T_1 = \log_e(0.5) - 1 = -\log_e 2 - 1 \tag{49}$$

$$T_2 = \frac{-\log_e(0.5) \times 0.5}{0.5} - 1 = \log_e 2 - 1 \tag{50}$$

$$T_3 = \frac{2\log_e 2}{2 - 1} - 1 = 2\log_e 2 - 1 \tag{51}$$

Using equation (49) - (51) in (48), we have

$$\begin{aligned} e^x \cong & \frac{1}{2}[x + \log_e 2 + 1]_+ + \frac{1}{2}[x - \log_e 2 + 1]_+ + \\ & \cdots \frac{1}{2}[x - 2 \cdot \log_e 2 + 1]_+ \end{aligned} \tag{52}$$

Substituting equation (52) in (42), for $i \in (1, \cdots, N)$, we get

$$\sum\nolimits_{i=1}^{N} [x_i + O_1 - h] + [x_i + O_2 - h] + [x_i + O_3 - h] = C' \tag{53}$$

Here, $O_1 = C(1 + log_e 2), O_2 = C(1 - log_e 2)$ and $O_3 = C(1 - 2log_e 2)$ are the offsets and $C' = 2C$ is a tunable parameter. Equation (53) shows the approximation of exponential function with 3-splines. The same can then be generalized to $S$-splines, $N$-inputs as

$$\begin{aligned} \sum\nolimits_{i=1}^{N} \sum\nolimits_{j=1}^{S} [x_i + O_j - h]_+ = C \\ \sum\nolimits_{i=1}^{N} \sum\nolimits_{j=1}^{S} [x_{i,j} - h]_+ = C \end{aligned} \tag{54}$$

where $O_j$ is the offset due to $j^{th}$ spline, $x_{i,j}$ is the $i^{th}$ input corresponding to $j^{th}$ spline, $C$ is a hyperparameter and $S$ is a design parameter also called splines count. It can be noted from Fig. 2a that with the increase in the number of splines ($S$), the computational precision increases, while the input dimension increases along $N$. Equation (54) is called Generalized Margin Propagation function (GMP).

## A. Acknowledgments

The authors would like to acknowledge the joint IISc-WashU MoU to facilitate the collaboration between the two institutions. This work is also supported by the Department of Science and Technology of India (SERB CRG/2021/005478, DST/IMP/2018/000550).

## References

[1] G. Cauwenberghs and M. Bayoumi, *Learning on silicon: Adaptive VLSI neural systems*. Springer Science & Business Media, 1999, vol. 512.
[2] S. Chakrabartty and G. Cauwenberghs, "Sub-microwatt analog vlsi trainable pattern classifier," *IEEE Journal of Solid-State Circuits*, vol. 42, no. 5, pp. 1169–1179, 2007.
[3] K. Kang and T. Shibata, "An on-chip-trainable gaussian-kernel analog support vector machine," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 7, pp. 1513–1524, 2009.
[4] T. Yamasaki and T. Shibata, "Analog soft-pattern-matching classifier using floating-gate mos technology," *IEEE Transactions on Neural Networks*, vol. 14, no. 5, pp. 1257–1265, 2003.
[5] P. Kumar, K. Zhu, X. Gao, S.-D. Wang, M. Lanza, and C. S. Thakur, "Hybrid architecture based on two-dimensional memristor crossbar array and cmos integrated circuit for edge computing," *npj 2D Materials and Applications*, vol. 6, no. 1, pp. 1–10, 2022.
[6] G. Huang, J. Hu, Y. He, J. Liu, M. Ma, Z. Shen, J. Wu, Y. Xu, H. Zhang, K. Zhong *et al.*, "Machine learning for electronic design automation: A survey," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 26, no. 5, pp. 1–46, 2021.
[7] B. Vigoda, "Analog Logic: Continuous-Time Analog Circuits for Statistical Signal Processing," Ph.D. dissertation, Program in Media Arts and Sciences, School of Architecture and Planning, Massachusetts Institute of Technology, 2003.
[8] R. S. Pal, S. Sharma, and S. Dasgupta, "Recent trend of FinFET devices and its challenges: A review," in *2017 Conference on Emerging Devices and Smart Systems (ICEDSS)*, 2017, pp. 150–154.
[9] P. Kinget and M. Steyaert, *Implications of Transistor Mismatch on Analog Circuit Design and System Performance", bookTitle="Analog VLSI Integration of Massive Parallel Signal Processing Systems*. Boston, MA: Springer US, 1997, pp. 21–81. [Online]. Available: https://doi.org/10.1007/978-1-4757-2580-3_2
[10] P. Jespers and B. Murmann, *Systematic Design of Analog CMOS Circuits: Using Pre-Computed Lookup Tables*. Cambridge University Press, 2017.

[11] "Predictive technology models, arizona state university, nanoscale integration and modeling group," 2022. [Online]. Available: https://ptm.asu.edu/

[12] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "ASAP7: A 7-nm finFET predictive process design kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002626921630026X

[13] E. Vittoz and J. Fellrath, "CMOS Analog Integrated Circuits Based on Weak Inversion Operations," *IEEE journal of solid-state circuits*, vol. 12, no. 3, pp. 224–231, 1977.

[14] E. Seevinck and R. J. Wiegerink, "Generalized translinear circuit principle," *IEEE journal of solid-state circuits*, vol. 26, no. 8, pp. 1098–1102, 1991.

[15] M. Gu and S. Chakrabartty, "Synthesis of Bias-Scalable CMOS Analog Computational Circuits Using Margin Propagation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 2, pp. 243–254, 2012.

[16] Y. Tsividis, *The MOS Transistor*. New York: Oxford University Press, 2013.

[17] J.-P. Colinge *et al.*, *FinFETs and other multi-gate transistors*. Springer, 2008, vol. 73.

[18] C. C. Enz, F. Krummenacher, and E. A. Vittoz, "An Analytical MOS Transistor Model Valid in All Regions of Operation and Dedicated to Low-Voltage and Low-Current Applications," *Analog Integr. Circuits Signal Process.*, vol. 8, no. 1, p. 83–114, jul 1995. [Online]. Available: https://doi.org/10.1007/BF01239381

[19] C. Galup-Montoro, M. C. Schneider, A. I. A. Cunha, F. R. de Sousa, H. Klimach, and O. F. Siebel, "The Advanced Compact MOSFET (ACM) Model for Circuit Analysis and Design," in *2007 IEEE Custom Integrated Circuits Conference*, 2007, pp. 519–526.

[20] B. Linares-Barranco and T. Serrano-Gotarredona, "On the design and characterization of femtoampere current-mode circuits," *IEEE Journal of Solid-State Circuits*, vol. 38, no. 8, pp. 1353–1363, 2003.

[21] H. F. Baghtash, K. Monfaredi, and S. J. Azhari, "A Novel High Performance Atto-Ampere Current Mirror," in *2010 International Conference on Signal Acquisition and Processing*, 2010, pp. 27–30.

[22] S. J. Azhari and G. Nickhah, "A Novel Ultra-Low-Power, Low-Voltage, Ultra-High Output Resistance and Uniquely High Bandwidth Femto-Ampere Current Mirror," *Circuits, Systems, and Signal Processing*, vol. 36, pp. 3527–3548, 2017.

[23] J. Lazzaro, S. Ryckebusch, M. A. Mahowald, and C. A. Mead, "Winner-take-all networks of O (n) complexity," *Advances in neural information processing systems*, vol. 1, 1988.

[24] S. B. Furber, W. John Bainbridge, J. Mike Cumpstey, and S. Temple, "Sparse distributed memory using n-of-m codes," *Neural Networks*, vol. 17, no. 10, pp. 1437–1451, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608004001443

[25] P. Kumar, A. Nandi, S. Chakrabartty, and C. S. Thakur, "CMOS Circuits for Shape-Based Analog Machine Learning," 2022. [Online]. Available: https://arxiv.org/abs/2202.05022

[26] S. Narayanan, A. Shafiee, and R. Balasubramonian, "INXS: Bridging the throughput and energy gap for spiking neural networks," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 2451–2459.

[27] A. Shafiee, A. Nag, N. Muralimanohar, R. Balasubramonian, J. P. Strachan, M. Hu, R. S. Williams, and V. Srikumar, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 14–26, 2016.

[28] M. Pelgrom, A. Duinmaijer, and A. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 5, pp. 1433–1439, 1989.

[29] C. Popa, "Improved Accuracy Current-Mode Multiplier Circuits With Applications in Analog Signal Processing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 2, pp. 443–447, 2014.

[30] N. Saxena and J. Clark, "A four-quadrant CMOS analog multiplier for analog neural networks," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 6, pp. 746–749, 1994.

[31] N. Cristianini, J. Shawe-Taylor *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[32] A. R. Nair, P. K. Nath, S. Chakrabartty, and C. S. Thakur, "Multiplierless MP-Kernel Machine For Energy-efficient Edge Devices," *arxiv*, vol. abs/2106.01958, 2021. [Online]. Available: https://arxiv.org/abs/2106.01958

[33] V. Joshi, M. Le Gallo, S. Haefeli, I. Boybat, S. R. Nandakumar, C. Piveteau, M. Dazzi, B. Rajendran, A. Sebastian, and E. Eleftheriou, "Accurate deep neural network inference using computational phase-change memory," *Nature communications*, vol. 11, no. 1, pp. 1–13, 2020.

[34] F. Palumbo, C. Gallicchio, R. Pucci, and A. Micheli, "Activity Recognition system based on Multisensor data fusion (AReM)," 2016. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/Activity+Recognition+system+based+on+Multisensor+data+fusion+(AReM)

[35] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[36] S. Tannirkulam Chandrasekaran, A. Jayaraj, V. Elkoori Ghantala Karnam, I. Banerjee, and A. Sanyal, "Fully integrated analog machine learning classifier using custom activation function for low resolution image classification," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 3, pp. 1023–1033, 2021.

[37] Z. Wang and N. Verma, "A low-energy machine-learning classifier based on clocked comparators for direct inference on analog sensors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 11, pp. 2954–2965, 2017.

[38] J. Zhang, Z. Wang, and N. Verma, "A machine-learning classifier implemented in a standard 6t sram array," in *2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, 2016, pp. 1–2.