Analyzing Multilevel Stochastic Circuits using Correlation Matrices

Owen Hoffend & John P. Hayes
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI, 48109, USA
{ohoffend, jhayes}@umich.edu

***Abstract*—Stochastic computing (SC) is a digital design paradigm that foregoes the conventional binary encoding in favor of pseudo-random bitstreams. Stochastic circuits operate on the probability values of bitstreams, and often achieve low power, low area, and fault-tolerant computation. Most SC designs rely on the input bitstreams being independent or uncorrelated to obtain the best results. However, circuits have also been proposed that exploit deliberately correlated bitstreams to improve area or accuracy. In such cases, different sub-circuits may have different correlation requirements. A major barrier to multi-layer or hierarchical stochastic circuit design has been understanding how correlation propagates from a circuit's inputs to its outputs while meeting the correlation requirements for all its sub-circuits. In this paper, we introduce correlation matrices and extensions to probability transfer matrix (PTM) algebra to analyze complex correlation behavior, thereby alleviating the need for computationally intensive bit-wise simulation. We apply our new correlation analysis to two multi-layer SC image processing and neural network circuits and show that it helps designers to systematically reduce correlation error.**

Keywords - Stochastic computing; signal correlation; correlation matrix; error analysis; stochastic circuit design

I. INTRODUCTION

Stochastic computing (SC) is a digital logic design paradigm that performs computation using pseudo-random sequences of bits [1]. SC was originally proposed in the 1960s, but has seen a recent resurgence of interest due to its ability to implement computationally expensive functions such as digital filters [2] and low-density parity-check (LDPC) decoders [3] with lower area and higher soft-error tolerance than conventional binary approaches [1]. In SC, a stochastic number (SN) $X = (x_1, x_2, \dots, x_N)$ is a sequence, or bitstream, of N successive 0s or 1s sampled from a binary random variable. In the basic (unipolar) case, X takes on a value P_X in the interval [0,1] defined by the probability that any arbitrary bit x is 1, i.e., $P_X = P(x = 1)$. Figure 1a shows a classic example of how SC achieves multiplication using a single AND gate, saving considerable area relative to a conventional adder-based binary multiplier. Figure 2a-b show the circuits used to convert binary values to and from SNs, respectively. One inherent practical limitation of SC is that all operations are approximate and are susceptible to quantization errors and random fluctuation errors (RFEs) [4]. In general, these errors decrease as the bitstream length N is increased, creating a smooth tradeoff between accuracy and latency.

Stochastic circuits are also susceptible to correlation error, which occurs when the correlation(s) between one or more pairs of inputs are different from the optimal amount of input correlation.

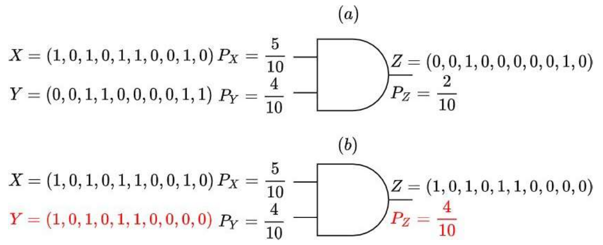

(a)
$X = (1,0,1,0,1,1,0,0,1,0)$ $P_X = \frac{5}{10}$
$Y = (0,0,1,1,0,0,0,0,1,1)$ $P_Y = \frac{4}{10}$
$Z = (0,0,1,0,0,0,0,0,1,0)$
$P_Z = \frac{2}{10}$
(b)
$X = (1,0,1,0,1,1,0,0,1,0)$ $P_X = \frac{5}{10}$
$Y = (1,0,1,0,1,1,0,0,0,0)$ $P_Y = \frac{4}{10}$
$Z = (1,0,1,0,1,1,0,0,0,0)$
$P_Z = \frac{4}{10}$

Fig. 1: Example of stochastic behavior of an AND gate: (a) With uncorrelated inputs of P_X and P_Y, computing the product $P_X \times P_Y$. (b) With correlated inputs, computing $min(P_X, P_Y)$. Here, the bitstream length is $N = 10$.

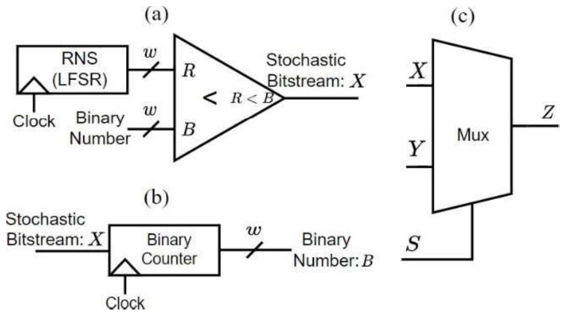


Fig. 2: Other fundamental SC components: (a) Stochastic number generator (SNG), consisting of a random number source (RNS) and a comparator; (b) Stochastic-to-binary converter using a binary counter; (c) Multiplexer (MUX) computing scaled addition $P_X = (1 - P_S)P_X + P_S P_Y$.

For example, the AND gate circuit in Fig. 1 requires independent (uncorrelated) inputs to perform accurate multiplication, and Fig. 1b shows that the circuit's behavior changes from $P_Z = P_X P_Y$ to $P_Z = min(P_X, P_Y)$ when the bitstream X is suitably correlated with Y.

Observe that the primary difference between the two cases in Fig. 1a-b is the fact that the input bitstreams for the bottom case have the maximum possible overlap between the 1s. In prior SC research, correlation between pairs of bitstreams is most frequently measured using the stochastic cross correlation (SCC) metric, which assigns $SCC(X, Y) = 1$ to this maximum overlap (maximally correlated) case, $SCC(X, Y) = -1$ to the case with minimum overlap (sometimes called anti-correlated), and $SCC(X, Y) = 0$ to the uncorrelated case [5]. SCC, which is discussed further in Sec. II, is a piecewise-linear function that connects these three extrema, allowing bitstreams with partial correlation to be represented as well.

In SC, bitstreams may become correlated for several reasons, such as circuit fan-in/fan-out or random number source (RNS) sharing among inputs. Unlike quantization error and RFEs, correlation error does not generally decrease with bitstream length N, rather it is an inherent property of the circuit's structure and its input sources.

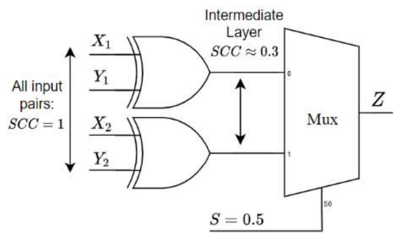


Fig. 3: Roberts Cross edge detection (RCED) circuit, exploiting input correlation SCC= +1 to compute a scaled sum of absolute differences.

To design accurate SC circuits despite undesired correlation, prior research has relied on adding extra hardware in the form of sequential correlators and decorrelators for (re-)generating input bitstreams [6][7][12]. Circuits that deliberately exploit correlation have shown promising results for some important SC applications, since they often lead to compact designs without expensive correlation manipulation hardware [5].

For instance, in [8], Alaghi et al. propose the small edge detection circuit shown in Fig. 3, which computes the function $P_Z = 1/2|P_{X_1} - P_{Y_1}| + 1/2|P_{X_2} - P_{Y_2}|$ when the data inputs all have maximum correlation of $SCC = 1$ with respect to each other. The first-layer XOR gates perform absolute-valued subtraction; the second layer consists of a MUX scaled-adder. This RCED design saves considerable area over conventional SC implementations because the first layer is simple and employs maximally correlated input bitstreams that can share a single RNS. However, like the AND gate in Fig. 1, it exhibits correlation error if its inputs do not meet its input correlation specification. The AND gate multiplier requires $SCC = 0$, while the RCED circuit requires $SCC = 1$. Fig. 3 also shows how the intermediate signals from the first layer of XOR gates are *partially correlated* at some in-between value $SCC \approx 0.3$. With the notable exception of the MUX scaled-adder, which is insensitive to correlation, nearly all known useful stochastic circuits require that the correlation between pairs of inputs be maintained at $SCC(X,Y) = 0$, 1, or -1 for the best accuracy, so using the pair of XOR-gate absolute-subtractors as an input to any other design would likely result in correlation error. In general, such correlation degradation across circuit layers poses a major and poorly understood problem for modular, hierarchical stochastic circuit design.

One notable example of prior work done in designing multi-layer circuits that exploit correlation is [12]. Here, Abdellatef et al. design a multi-layer SC image processing pipeline consisting of median filtering, smoothing, edge detection, and thresholding. However, their design strategy is restricted to finding sub-circuits that either leave correlation completely unmodified or are entirely insensitive to correlation, otherwise re-correlators are added to correct partial correlation. Such an all-or-nothing strategy is often unnecessarily complicated and pessimistic, as designs receiving input correlations that are *close* to the ideal requirements (e.g., $SCC = 0.9$ instead of 1) may still achieve sufficiently low error. To guide SC design, a mathematical model for quantifying correlation propagation and its error is highly desirable.

Early attempts to perform mathematical SC correlation analysis utilized probabilistic transfer matrix (PTM) theory, first proposed in [9], partly because of its ability to model some aspects of correlation [5]. PTMs, intuitively, are an extension of Boolean truth tables that permit probabilistic inputs/outputs, and thus naturally bridge the gap between conventional Boolean analysis and SC. Existing PTM theory, however, has no general answer for how to specify the input correlation requirements of SCs, or how to extract the output correlation structure after applying PTMs. Therefore, in the literature the accuracy of stochastic circuits is determined almost exclusively by means of bit-level simulation, which is computationally expensive and provides little insight into the factors underlying correlation [4]. In this paper, we address this issue by introducing correlation matrices,which are matrices specifying all of the pair-wise correlations between a set of bitstreams. Our method is unique in its ability to handle circuits with any number of inputs and outputs, and with any input/output correlation structure.

The main contributions of this work are:

- The adoption of correlation matrices as explicit and concise representations of the correlations between stochastic bitstreams in multi-layer circuits.
- An extension of PTM algebra that employs correlation matrices as a tool for correlated circuit analysis and synthesis, which reduces reliance on bit-level simulation.
- An application of our correlation analysis to the design of multi-layer image processing and neural network circuits in order to reduce correlation error.

II. BACKGROUND

First, we briefly discuss methods of measuring correlation within stochastic circuits, as well as the relevant theory behind probability transfer matrices (PTMs).

A. Correlation Measurement in SC

Early attempts to understand the impact of correlation on SC primarily used conventional Pearson correlation from statistics [10]. However, in [5] Pearson correlation was shown to be suboptimal for SC due to its value dependence. By far the most widely used correlation measure in the SC context is the stochastic cross correlation (SCC) [5] defined by Eq. 1:

$$SCC(X,Y) = \begin{cases} \frac{P_{X \wedge Y} - P_X P_Y}{min(P_X, P_Y) - P_X P_Y} & if\ P_{X \wedge Y} > P_X P_Y \\ \frac{P_{X \wedge Y} - P_X P_Y}{P_X P_Y - max(P_X + P_Y - 1, 0)} & otherwise \end{cases} \quad (1)$$

where $P_{X \wedge Y}$ measures the likelihood of finding a pair of overlapping 1s, as in $P(xy = 11)$. For example, if the SNs are $X = \mathbf{101001}$ ($P_X = 3/6$) and $Y = \mathbf{111001}$ ($P_Y = 4/6$), then $P_{X \wedge Y} = 3/6$. Consequently, $SCC(X,Y) = 1$, because these bitstreams have maximum overlap of 1s. To gain some intuition about SCC, note that the numerators of both piecewise cases are the same. We can rewrite Eq. 1 as:

$$SCC(X,Y) = \frac{P_{X \wedge Y} - P_X P_Y}{Norm(P_X, P_Y)} \quad (2)$$

Here, $Norm(P_X, P_Y)$ is a piecewise function that scales the SCC value to keep it in the interval $[-1,1]$.

B. Probabilistic Transfer Matrices

Next, we explain how PTMs may be used to model the behavior of stochastic circuits [9][5]. Suppose we have an n-input combinational stochastic circuit that receives a vector of SNs, $\boldsymbol{X}$, and computes a set of k Boolean functions: $f: \{0,1\}^n \rightarrow \{0,1\}^k$. The resulting output $\boldsymbol{Z}$ is a new k-element vector of SNs. We treat $\boldsymbol{P_X}$ and $\boldsymbol{P_Z}$ (boldface) as vectors of the circuit's input and SN probabilities, respectively, i.e. $\boldsymbol{P_X} = [P_{X_1}, P_{X_2}, \dots, P_{X_n}]^T$.

This circuit's PTM is a $2^n \times 2^k$ matrix $\boldsymbol{M}_f$ whose rows represent all possible circuit input patterns: 00…0 through 111…1, and whose columns represent all possible output patterns. The (i,j)th entry of the PTM is the probability that the input pattern given by row i will produce the output pattern given by column j. A PTM is therefore summarizes the circuit's stochastic behavior. As an example, consider the MUX adder shown in Fig. 2c, which has a select input probability of P_S. If P_S is constant and input S is independent of both MUX data inputs X and Y (such as in RCED), then the circuit's usual 8-row Boolean truth table may be replaced by a 4-row PTM:

$$\mathrm{M}_{MUX} = \begin{array}{c} \\ xy=00 \\ xy=01 \\ xy=10 \\ xy=11 \end{array} \begin{array}{c} \begin{array}{cc} P(z=0) & P(z=1) \end{array} \\ \begin{pmatrix} 1 & 0 \\ 1-P_S & P_S \\ P_S & 1-P_S \\ 0 & 1 \end{pmatrix} \end{array} \tag{3}$$

PTMs generalize the concept of truth tables to circuits with probabilistic input-output values, including those that are correlated. If the MUX circuit were fed correlated inputs, then the input pattern $xy = 11$ will occur more frequently than the pattern $xy = 01$, so these rows of the PTM would have a higher impact on the output probability distribution.

On their own, PTMs, like truth tables, make no assumption about the circuit's input signal distribution. To model this distribution, we must assign a probability value to each of the PTM's 2^n rows. In the existing PTM literature, a separate $2^n \times 1$ PTM containing all the joint input probabilities is used [9][5]. For example, a circuit with 2 inputs has the following length-4 input vector:

$$\boldsymbol{v}_{in} = \begin{pmatrix} P(xy=00) \\ P(xy=01) \\ P(xy=10) \\ P(xy=11) \end{pmatrix} \tag{4}$$

As noted in [5], $\boldsymbol{v}_{in}$ implicitly holds information about correlation between circuit inputs. For instance, the vectors $\boldsymbol{v}_{+1} = [0.5,0,0,0.5]^\top$ and $\boldsymbol{v}_{-1} = [0,0.5,0.5,0]^\top$, transposed here for brevity, both refer to a pair random variables with marginal probabilities of 0.5, but they differ significantly in terms of SCC: The first has $SCC(X,Y) = 1$, while the second has $SCC(X,Y) = -1$. Since $\boldsymbol{v}_{in}$ is a vector carrying external information about the input SNs, we distinguish it from ordinary circuit PTMs by designating it as a *probability transfer vector* (PTV).

When combined with PTMs, PTVs are a powerful tool for analyzing the behavior of stochastic circuits. Consider an n-input circuit with output function f and a $2^n \times 2^k$ PTM $\boldsymbol{M}_f$. The circuit's $2^k \times 1$ output PTV $\boldsymbol{v}_{out}$ can be expressed as a matrix-vector product between a PTM and PTV, thus:

$$\boldsymbol{v}_{out} = \boldsymbol{M}_f^\top \boldsymbol{v}_{in} \tag{5}$$

Intuitively, this product computes a weighted sum of the rows of the PTM, analogous to indexing into a Boolean truth table. An important consequence of Eq. 5 is that the PTMs of subsequent layers within a multi-layer circuit may be combined using matrix multiplication, as shown in [9]. Consider a two-layer circuit where $\boldsymbol{M}_{f_1}$ and $\boldsymbol{M}_{f_2}$ are the PTMs for the first and second layers, respectively. Then the overall circuit behavior is given by $\boldsymbol{v}_{out} = \boldsymbol{M}_{f_2}^\top \boldsymbol{M}_{f_1}^\top \boldsymbol{v}_{in}$. This composition can be thought of as two iterations of Eq.5, where we first compute $\boldsymbol{v}_{temp} = \boldsymbol{M}_{f_1}^\top \boldsymbol{v}_{in}$ followed by $\boldsymbol{v}_{out} = \boldsymbol{M}_{f_2}^\top \boldsymbol{v}_{temp}$. Therefore, if the input PTV $\boldsymbol{v}_{in}$ is known, the output PTV of any m-layer combinational circuit may be found via m successive matrix multiplications.

III. EXTENSIONS TO PTM ALGEBRA

A. Finding the Output Probabilities

In Sec. II, we showed how to find an output PTV for a given input PTV. Now, we present a new method for transforming any PTV back into a vector of probabilities. This allows us to evaluate any circuit's stochastic function using exact PTM algebra instead of approximating it with bit-level simulation. Suppose we define a $2^n \times n$ matrix $\boldsymbol{B}(n)$ such that the ith row contains the n-bit binary representation of the integer value $i - 1$. We call such a matrix a *binary integer matrix* (BIM). For example, the BIM for 2-bit binary integers is:

$$\boldsymbol{B}(2) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix} \tag{6}$$

BIMs are useful for PTM analysis because each row i of a PTV corresponds to the probability of seeing a specific binary bit pattern such as $P(xy = 10)$, and this pattern is given by the matching ith row of a BIM. Formally, we write: $\boldsymbol{v}_i = P(x_1x_2 \ldots x_n = \boldsymbol{B}(n)_i)$. Given this relationship, we can use BIMs to reduce (marginalize) PTVs back into probability vectors via matrix-vector multiplication.

Theorem 1: Given an n-input, k-output stochastic circuit represented by the PTM $\boldsymbol{M}_f$, and a PTV $\boldsymbol{v}_{in}$ defining the circuit input, the input and output probability vectors are, respectively:

$$\boldsymbol{P_X} = \boldsymbol{B}(n)^\top \boldsymbol{v}_{in} \tag{7}$$

$$\boldsymbol{P_Z} = \boldsymbol{B}(k)^\top \boldsymbol{v}_{out} = \boldsymbol{B}(k)^\top \boldsymbol{M}_f^\top \boldsymbol{v}_{in} \tag{8}$$

As an example, by applying Eq. 7 to the two-input PTV in Eq. 4, we see that the result is exactly equal to the vector of marginal probability distributions (by the law of total probability):

$$\begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}^T \begin{pmatrix} P(xy=00) \\ P(xy=01) \\ P(xy=10) \\ P(xy=11) \end{pmatrix} = \begin{pmatrix} P(xy=10) + P(xy=11) \\ P(xy=01) + P(xy=11) \end{pmatrix} = \begin{pmatrix} P_X \\ P_Y \end{pmatrix}$$

If the input PTV is known, Theorem 1 is an exact alternative to circuit simulation for finding the output probabilities, even for circuits with many more than 2 inputs/outputs. However, in practical circuit analysis, often only $\boldsymbol{P_X}$ is known, not $\boldsymbol{v}_{in}$. Unfortunately, we cannot generally compute the inverse of Eq. 7 or Eq. 8. Input PTVs cannot be found with only knowledge of $\boldsymbol{P_X}$, since $\boldsymbol{P_X}$ on its own does not include correlation information. In Sec. III-B we explain how to find this information, then in Sec. IV we combine it with $\boldsymbol{P_X}$ to form the input PTV.

B. Correlation Matrices & Mutual Correlation

For a SC with n inputs, we must consider each of the $\binom{n}{2}$ possible input signal pairs individually for correlation analysis. We employ correlation matrices as a concise method of representing all such pairs. Thus, we define a *correlation matrix* $\boldsymbol{C}$ such that the (i,j)th entry is $\boldsymbol{C}_{ij} = SCC(X_i, X_j)$. Correlation matrices are always symmetric, and their diagonal entries are always 1 because bitstreams are always fully correlated with themselves. For SC circuits requiring independent inputs, $\boldsymbol{C}$ is just the $n \times n$ identity matrix $\boldsymbol{I}$. A more complex example is the RCED edge detector in Fig. 3. It requires that all data inputs X_1,

Y_1, X_2, and Y_2 be fully correlated (SCC = 1) with each other, however the select input S must remain uncorrelated ($SCC = 0$) with all data inputs. We can express this input correlation requirement with the following correlation matrix:

$$\mathbf{C}_{RC} = \begin{array}{c|ccccc} & X_1 & Y_1 & X_2 & Y_2 & S \\ \hline X_1 & 1 & 1 & 1 & 1 & 0 \\ Y_1 & 1 & 1 & 1 & 1 & 0 \\ X_2 & 1 & 1 & 1 & 1 & 0 \\ Y_2 & 1 & 1 & 1 & 1 & 0 \\ S & 0 & 0 & 0 & 0 & 1 \end{array} \quad (9)$$

Typical stochastic circuit designs including RCED require all pairs of input bitstreams, or a subset of pairs, to maintain the same correlation value with respect to each other. We call this desirable property *mutual correlation*. For example, the set of data inputs to the RCED $\{X_1, Y_1, X_2, Y_2\}$ is *mutually correlated* with $SCC = 1$, since the respective entries in the correlation matrix $\boldsymbol{C}_{RC}$ (Eq. 9) are 1 for all pairs of bitstreams in this group.

Just as we were able to reduce a PTV to probabilities (Theorem 1), we can also extract the correlation information from a PTV, yielding a correlation matrix without having to perform bit-wise simulation:

Theorem 2: Given a PTV $\boldsymbol{v_{in}}$ of size 2^n specifying a vector of n stochastic bitstreams $\boldsymbol{X}$, and their corresponding probability values $\boldsymbol{P_X}$, the correlation matrix is:

$$\boldsymbol{C_{ij}} = \frac{\sum_{q=1}^{2^n} \boldsymbol{B}(n)_{qi}\boldsymbol{B}(n)_{qj}\boldsymbol{v}_{\boldsymbol{in}_q} - P_{X_i}P_{X_j}}{Norm\left(P_{X_i}, P_{X_j}\right)} \quad (10)$$

Observe that P_{X_i} and P_{X_j} can be found from $\boldsymbol{v_{in}}$ using Theorem 1, thus Eq. 10 in Theorem 2 depends only on $\boldsymbol{v_{in}}$. Eq. 10 extends the definition of SCC given in Eq. 2. The summation term computes $P_{X_i \wedge X_j}$. It first identifies all possible ways bitstream i can overlap with bitstream j using the relevant columns of a BIM. If bitstreams i and j overlap on the qth pattern, the probability of this pattern occurring, $\boldsymbol{v}_{\boldsymbol{in}_q}$, is added to the total for $P_{X_i \wedge X_j}$. Theorem 2 can be used to find the exact output correlation matrix of a circuit with an arbitrary PTM $\boldsymbol{M_f}$ using the output PTV $\boldsymbol{v}_{out} = \boldsymbol{M}_f^{\top}\boldsymbol{v}_{in}$. This knowledge can guide multi-level circuit design by allowing one to check if the output correlation matrix of a given layer matches (or is close to) the correlation requirements for the next layer.

IV. Generating Input PTVs

So far, we have shown how to extract the probability vector and correlation matrix information from a PTV using Theorems 1 and 2. However, we also noted that these theorems have limited applicability unless the circuit's input PTV is known ahead of time. To solve this problem, we now explain how to construct PTVs for several common types of correlation matrices. In each case, the PTV for correlation matrix $\boldsymbol{C}$ is treated as a function of the input probabilities, $\boldsymbol{v_C}(\boldsymbol{P_X})$.

A. PTVs for Mutually Independent Inputs

In SC design, the most common correlation matrix is the identity matrix $\boldsymbol{I}$, which occurs when all bitstreams are mutually correlated at $SCC = 0$, i.e., are independent. For this case, we write the PTV as $\boldsymbol{v}_I(\boldsymbol{P_X})$. Now consider such a PTV for a pair of inputs X and Y. When sampling from bitstream X, there are two possible outcomes: $x = 0$ and $x = 1$, with probabilities $(1 - P_X)$ and P_X, respectively (Bernoulli trials). Likewise, the same is true for Y. Then, with $\otimes$ denoting the tensor product operator used in PTM theory for modelling parallel circuit elements (see [9]), we can write:

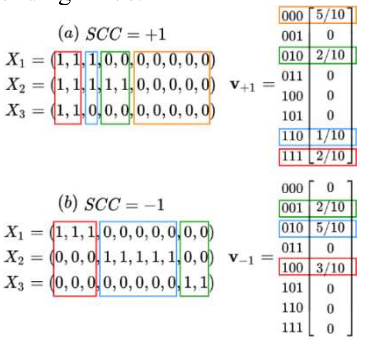
$$\boldsymbol{v}_I(P_X, P_Y) = \begin{pmatrix} 1 - P_X \\ P_X \end{pmatrix} \otimes \begin{pmatrix} 1 - P_Y \\ P_Y \end{pmatrix} = \begin{pmatrix} (1 - P_X)(1 - P_Y) \\ P_X(1 - P_Y) \\ (1 - P_X)P_Y \\ P_X P_Y \end{pmatrix} \quad (11)$$

Here, we utilize the well-known identity from probability theory that $P(xy = 11) = P(x = 1)P(y = 1)$ if X and Y are independent. The tensor product in Eq. 11 has the effect of applying this rule to all four possible joint x, y samples. For instance, the probability of sampling $x = 0$ and $y = 1$ is $(1 - P_X)P_Y$. This observation generalizes to the following equation for the PTV of any number of independent inputs:

$$\boldsymbol{v}_I(\boldsymbol{P_X}) = \bigotimes_{i=1}^{n} \begin{pmatrix} 1 - P_{X_i} \\ P_{X_i} \end{pmatrix} \quad (12)$$

B. PTVs for Mutually +1/-1 Correlated Inputs

Next, we show how to derive a PTV for a set of inputs that are pairwise mutually correlated with $SCC = \pm 1$, which we write as $\boldsymbol{v_{+1}}(\boldsymbol{P_X})$ and $\boldsymbol{v_{-1}}(\boldsymbol{P_X})$, respectively. After independent inputs, these are the most frequently used correlation matrices. To illustrate, Fig. 4 shows a group of three bitstreams of length $N = 10$ correlated at $SCC = \pm 1$, as well as their corresponding PTVs:



(a) SCC = +1
X1 = (1, 1, 1, 0, 0, 0, 0, 0, 0, 0)
X2 = (1, 1, 1, 1, 1, 0, 0, 0, 0, 0)
X3 = (1, 1, 0, 0, 0, 0, 0, 0, 0, 0)
v+1 =
000 5/10
001 0
010 2/10
011 0
100 0
101 0
110 1/10
111 2/10
(b) SCC = −1
X1 = (1, 1, 1, 0, 0, 0, 0, 0, 0, 0)
X2 = (0, 0, 0, 1, 1, 1, 1, 1, 0, 0)
X3 = (0, 0, 0, 0, 0, 0, 0, 0, 1, 1)
v−1 =
000 0
001 2/10
010 5/10
011 0
100 3/10
101 0
110 0
111 0

Fig. 4: Example of three correlated bitstreams and their PTVs. (a) $SCC = 1$. (b) $SCC = -1$. Here, $P_{X_1} = 3/10$, $P_{X_2} = 5/10$, $P_{X_3} = 2/10$.

From Fig. 4, we observe that both correlated PTVs are sparse vectors, and the locations of the nonzero entries depend on the relative magnitudes of P_{X_1}, P_{X_2}, and P_{X_3}. To construct such PTVs, first recall that the ith row of a PTV corresponds to the probability of seeing the bit pattern represented by the ith row of a BIM, i.e., $\boldsymbol{v}_i = P(x_1 x_2 \dots x_n = \boldsymbol{B}(n)_i)$. For the ith pattern, suppose we collected the probability values of the inputs that are required be 0 and those that are required be 1 into two sets, $\boldsymbol{r}_{1i} = \{P_{X_j} | \boldsymbol{B}(n)_{ij} = 1\}$ and $\boldsymbol{r}_{0i} = \{P_{X_j} | \boldsymbol{B}(n)_{ij} = 0\}$, respectively. For example, the index $i = 7$ in the PTV of a 3-input circuit has $\boldsymbol{v_{+1}}(\boldsymbol{P_X})_7 = P(x_1 x_2 x_3 = \boldsymbol{B}(3)_7) = P(x_1 x_2 x_3 = 110)$. Therefore, $\boldsymbol{r}_{1i} = \{P_{X_1}, P_{X_2}\}$ and $\boldsymbol{r}_{0i} = \{P_{X_3}\}$. We utilize these two sets to state the following theorem:

Theorem 3: Given an input vector $\boldsymbol{P_X}$, of length n, the PTVs for $SCC = \pm 1$ mutual correlation are:

$$\boldsymbol{v_{+1}}(\boldsymbol{P_X})_i = max\left(0, min(1 \cup \boldsymbol{r}_{1i}) - max(0 \cup \boldsymbol{r}_{0i})\right) \quad (13)$$

$$\boldsymbol{v_{-1}}(\boldsymbol{P_X})_i = \begin{cases} max(0, 1 - \sum \boldsymbol{P_X}) & if\ i = 1 \\ \boldsymbol{r}_{1i} - max(0, \sum \boldsymbol{P_X} - 1) & if\ |\boldsymbol{r}_{1i}| = 1 \\ max(0, \sum \boldsymbol{P_X} - 1) & otherwise \end{cases} \quad (14)$$

In the case of +1 mutual correlation, we can see from Fig. 4a that a 1 on the smaller of X_1 and X_2 will imply a 1 on the larger.

Thus, $P(x_1x_2 = 11) = min(\boldsymbol{r}_{1i}) = 3/10$ in this case. Similarly, a 0 on the larger bitstream implies a 0 on the smaller one, so we subtract $max(\boldsymbol{r}_{0i})$, yielding: $P(x_1x_2x_3 = 110) = 3/10 - 1/10 = 2/10$. Eq. 13 from Theorem 3 generalizes this min-minus-max procedure, and accounts for the edge cases $P(x_1x_2\ldots x_n = 00\ldots0) = 1 - max(\boldsymbol{r}_{0i})$ and $P(x_1x_2\ldots x_n = 11\ldots1) = min(\boldsymbol{r}_{1i})$ by unioning the $\boldsymbol{r}_{1i}$ and $\boldsymbol{r}_{0i}$ sets with 1 and 0, respectively, preventing the sets from being empty.

For $SCC = -1$ mutual correlation, note that the bitstreams in Fig. 4b contain no overlapping 1s, which is true if $\sum \boldsymbol{P_X} \leq 1$. We find that, with very few exceptions, inputs mutually correlated at -1 can only have $\sum \boldsymbol{P_X} > 1$ if $n = 2$. Under such input restrictions, the only nonzero rows of the PTV are those with $|\boldsymbol{r}_{1i}| = 1$, representing the patterns 001, 010, 100, etc. These rows contain all of the respective input's probability mass. For example, $P(x_1x_2x_3 = 010) = P_{X_2} = 5/10$. Eq. 14 implements this, with special cases for $P(x_1x_2\ldots x_n = 00\ldots0)$ and $n = 2$.

C. PTVs for more Complex Correlation Matrices

Next, we introduce two identities that allow us to express a broader class of correlation matrices as PTVs. In practical correlation analysis, the circuit inputs are not necessarily all mutually correlated at $SCC = 0$, 1, or -1. For instance, the matrix $\boldsymbol{C}_{RC}$ from our RCED example (Eq. 9) specifies that the data inputs are mutually correlated with $SCC = 1$, while the select input has $SCC = \boldsymbol{0}$ with all other inputs. $\boldsymbol{C}_{RC}$ is an example of a broad class of *block diagonal* correlation matrices, a structure that appears commonly in SC design. A general block diagonal correlation matrix has the form

$$\boldsymbol{C} = \begin{pmatrix} \boldsymbol{C}_1 & 0 & \ldots & 0 \\ 0 & \boldsymbol{C}_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \boldsymbol{C}_s \end{pmatrix} \tag{15}$$

where $\boldsymbol{C}_1, \ldots, \boldsymbol{C}_s$ are square correlation matrices (which may have different sizes) and s is the number of sub-matrices. We can construct PTVs for such matrices by again employing the tensor product of standard PTM theory. If $\boldsymbol{v}_{\boldsymbol{C}_1}, \boldsymbol{v}_{\boldsymbol{C}_2}, \ldots, \boldsymbol{v}_{\boldsymbol{C}_s}$ are the PTVs of the component correlation matrices, then the PTV of the full block matrix is:

$$\boldsymbol{v}_{\boldsymbol{C}} = \bigotimes_{i=1}^{s} \boldsymbol{v}_{\boldsymbol{C}_i} \tag{16}$$

In addition to block matrices, fractional SCC values often appear in practical circuit analysis. They occur naturally as signals propagate through multi-layer circuits. To handle such cases, we can use the fact that the SCC equation is (piecewise) linear. Thus, if $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ are correlation matrices containing values that all have the same sign, and $\alpha \in [0,1]$ is a scalar, then the following equation holds:

$$\boldsymbol{v}_{\alpha\boldsymbol{C}_1+(1-\alpha)\boldsymbol{C}_2} = \alpha\boldsymbol{v}_{\boldsymbol{C}_1} + (1-\alpha)\boldsymbol{v}_{\boldsymbol{C}_2} \tag{17}$$

Equation 17 generalizes a result in [5] that circuits whose inputs have fractional SCC values compute functions that are linear combinations of the functions at $SCC = 0$ and $SCC = \pm1$. We now illustrate the utility of Eqs. 16 and 17 with an example: Suppose we wish to model the behavior of the RCED circuit when the data inputs are mutually correlated at $SCC = 0.7$ instead of 1. This could occur if the RCED circuit is fed from a prior layer. The desired correlation matrix is:

$$\boldsymbol{C} = \begin{array}{c} \\ X_1 \\ Y_1 \\ X_2 \\ Y_2 \\ S \end{array} \begin{array}{c} \begin{array}{ccccc} X_1 & Y_1 & X_2 & Y_2 & S \end{array} \\ \begin{pmatrix} 1 & 0.7 & 0.7 & 0.7 & 0 \\ 0.7 & 1 & 0.7 & 0.7 & 0 \\ 0.7 & 0.7 & 1 & 0.7 & 0 \\ 0.7 & 0.7 & 0.7 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} = 0.7\begin{pmatrix} \mathbf{1}_{4\times4} & \mathbf{0} \\ \mathbf{0} & 1 \end{pmatrix} + 0.3\mathbf{I}_{5\times5} \tag{18}$$

In Eq. 18, the fractional correlation matrix is broken into a linear combination of a block correlation matrix and an identity matrix. We then use Eqs. 16 and 17 to find the PTV:

$$\boldsymbol{v}_{\boldsymbol{C}} = 0.7\left(\boldsymbol{v}_{+1}(P_{X_1}, P_{Y_1}, P_{X_2}, P_{Y_2}) \otimes \boldsymbol{v}_{+1}(P_S)\right) + 0.3\boldsymbol{v}_I(\boldsymbol{P_X}) \tag{19}$$

V. Application to Circuit Design

In this section, we use our PTM and correlation matrix results to show that apparently equivalent [11] designs can have very different correlation properties, and that this fact is useful for reducing correlation error in multi-layer circuits. Consider the MAJ gate shown in Fig. 5b, which outputs a 1 if a majority of its inputs are 1. It is known that the stochastic functions computed by MUX and MAJ are the same when the select input S (an arbitrary choice for MAJ) is set to 0.5 [11]—both gates compute the weighted sum: $Z = 0.5(X_1 + X_2)$. Despite this, the PTMs of these two circuits differ, as shown in Fig. 5c-d:

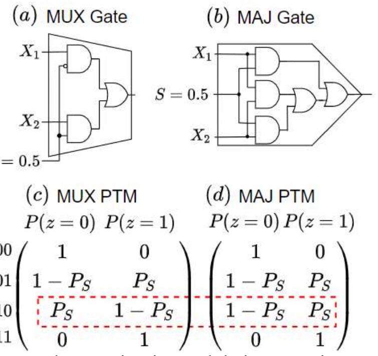


Fig. 5: MUX and MAJ circuits, and their respective PTMs.

Observe that when the select bit $S = 1$, a MAJ gate outputs 1 if either data input is also 1, whereas the output of a MUX under the same input conditions is sensitive to only one of the two data inputs. This hints that the two gates might affect correlation differently. To analyze this, we evaluated the impact of substituting MUX gates for MAJ gates within two different three-layer digital filtering circuits, shown in Fig. 6. The first circuit, Fig. 6a-b, is a set of four RCEDs followed by an OR-gate max-pooling tree [13]. The second circuit, Fig. 6c, is a new design of a multiply-accumulate (MAC) circuit, followed by a rectified linear activation unit (ReLU) layer, which is a ubiquitous operation in modern neural networks [14]. Under ideal correlation conditions, this MAC-ReLU neural network circuit will compute $max(0, \boldsymbol{w}_p^T\boldsymbol{X} - \boldsymbol{w}_n^T\boldsymbol{X}) = \boldsymbol{ReLU}(Z_{pos} - Z_{neg})$. The subtraction comes from the $SCC = 1$ behavior of an OR gate with one inverted input [5], which we exploit to add the positive and negative weight terms together.

For these experiments, we derived a set of three PTMs (one for each layer) for each circuit in Fig. 6, then a second set for the same two circuits but with the layer-2 MUX gates replaced with MAJ. We then sampled 100,000 random input vectors ($\boldsymbol{P_X}$) for each circuit. In the case of RCED, these samples were randomly selected 3×3 pixel patches from the popular image processing datasets MNIST (handwritten digits), and gray-scaled CIFAR-10 (thousands of images in ten classes). For MAC-ReLU, we simply used data and weights selected from the uniform random distribution. Note that any simulation-based approach would still require doing this sampling, followed by simulating N bits on every sample.

For each sample, we generated the input PTV according to the input correlation matrices for layer 1. These were block diagonal for both circuits, so we utilized Eq. 16. Specifically,

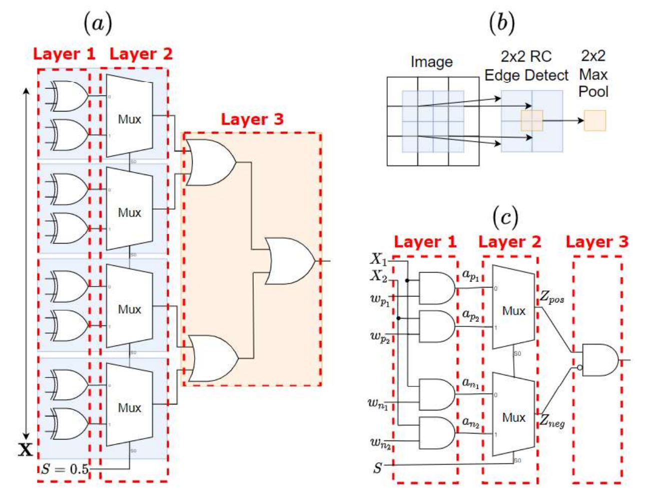


Fig. 6: (a) Three-layer image processing circuit: Layers 1 and 2 together are 4 parallel instances of RCED; Layer 3 is a 2x2 max pooling layer. (b) Image pixels contributing to (a)'s output. (c) Three-layer neural network circuit: Layers 1 and 2 together compute inner products: $Z_{\text{pos}} = \boldsymbol{w}_p^T\boldsymbol{X}$ and $Z_{neg} = \boldsymbol{w}_n^T\boldsymbol{X}$; Layer 3 computes $max(0, Z_{\text{pos}} - Z_{neg})$.

RCED requires a larger equivalent of C_{RC} (Eq. 9), where the data inputs $\boldsymbol{X}$ are mutually correlated with $SCC = 1$ and the select input S is independent. On the other hand, the MAC circuit requires the following correlation matrix:

$$C_{MAC} = \begin{array}{c|ccc} & \boldsymbol{X} & w & S \\ \hline \boldsymbol{X} & 1_{2\times2} & 0 & 0 \\ w & 0 & 1_{4\times4} & 0 \\ S & 0 & 0 & 1 \end{array} \qquad (20)$$

The block structure of $\boldsymbol{C}_{MAC}$ enables AND-gate multiplication between independent data and weight signals, yet keeps the data as correlated as possible for the subsequent ReLU unit. Given each input PTV, we computed the overall output PTV (Eq. 5) as well as the PTVs for all intermediate layers. This allowed us to examine both the output probability values with Theorem 1, and the intermediate correlation matrices via Theorem 2. We computed the output correlation error using the usual mean squared error formula:

$$Avg\ Corr\ Error \approx \frac{1}{m}\sum_{i=1}^{m}\left|\boldsymbol{P}_{\boldsymbol{Z}_{ideal}} - \boldsymbol{P}_{\boldsymbol{Z}}\right| \qquad (21)$$

where $\boldsymbol{P}_{\boldsymbol{Z}_{ideal}}$ is the circuit output supposing that the input to every layer is ideally (re)correlated according to the design's requirements. For the intermediate correlation matrices, we measured the average of all off-diagonal entries across all samples as an estimation for how well $SCC = 1$ correlation is preserved. The results are summarized in Table 1.

Circuit	Dataset	Avg. Corr at Layer 2	Avg. Corr. at Layer 3	*Avg Corr. Error* $m = 100{,}000$
RCED-MUX	MNIST	0.8607	0.8572	0.03077
RCED-MAJ	MNIST	0.8607	**0.8779**	**0.02670**
RCED-MUX	CIFAR-10	0.2986	0.3257	0.05492
RCED-MAJ	CIFAR-10	0.2986	**0.4792**	**0.04342**
MAC-MUX	Uniform	0.8474	0.8458	0.05161
MAC-MAJ	Uniform	0.8474	**0.9371**	**0.00875**

Table 1: Experimental results for Fig. 6 comparing the average SCC values of all bitstream pairs entering Layers 2 and 3, and the resulting absolute error relative to the design with ideal correlation, for the RCED/MAC-ReLU circuits using either MUX or MAJ gates for scaled addition.

Table 1 shows that switching from MUX to MAJ gates increases the average SCC entering layer 3 for all cases. For the RCED circuit, the improvement is more dramatic for the CIFAR-10 dataset, likely because the distribution of MNIST pixel values is highly skewed toward 1 and 0, relative to CIFAR-10 (see [4] for a detailed comparison). We see that the value distribution influences correlation after the first RCED layer; layer 2 sees higher incoming SCC for MNIST than for CIFAR-10. Switching to MAJ therefore yields a higher relative performance improvement for CIFAR-10 than for MNIST (1.26x reduction vs 1.15x). Our MAC-ReLU circuit shows that switching to MAJ achieves almost 6x lower correlation error, down to less than 0.9%, suggesting that our new MAC-ReLU design with MAJ gates does not require re-correlation hardware. Overall, these experiments show that stochastic equivalent circuits can yield different correlation behaviors.

VI. Conclusion

Correlation propagation through multi-layered stochastic circuits has been poorly understood in the past, especially for circuits with more than 2 inputs. In this paper, we introduced the use of correlation matrices to model the correlation requirements of relatively complex stochastic circuits with an arbitrary number of inputs and outputs. We then described how to translate these input correlation requirements into PTM form, and how use them to compute the output correlation matrix and output probabilities given known input probabilities. Finally, we applied our correlation analysis technique to two digital filtering circuits, including a novel neural network layer design. We showed that replacing the MUX-adder sub-circuit in these designs with an equivalent MAJ-adder reduced the overall circuit's correlation error. This is an example of how matrix-based correlation propagation analysis can be used to guide design decisions for practical circuits with multiple layers and many inputs/outputs.

Acknowledgement

This research was supported by the U.S. National Science Foundation under grant no. CCF-2006704.

References

[1] A. Alaghi and J.P. Hayes, "Survey of stochastic computing.", *ACM Trans. Embed. Comput. Syst.*, vol.12, issue 2s, pp.1-19, 2013.

[2] H. Ichihara et al., "Compact and accurate digital filters based on stochastic computing." *IEEE Trans. ETC*, vol.7, pp.31-43, 2019.

[3] W. J. Gross, V.C. Gaudet and A. Milner, "Stochastic implementation of LDPC decoders." *in proc. 39th Asilomar Conf. Signals, Systems and Computers*, pp. 713–717, 2005.

[4] T. Baker and J.P. Hayes, "Bayesian accuracy analysis of stochastic circuits." in proc. *IEEE ICCAD*, pp.1-9, 2020.

[5] A. Alaghi and J.P. Hayes, "Exploiting correlation in stochastic circuit design." *in proc. 31st IEEE ICCD*, pp.39–46, 2013.

[6] P.S. Ting and J.P. Hayes, "Isolation-based decorrelation of stochastic circuits." *in proc. 34th IEEE ICCD*, pp.88-95, 2016.

[7] V. Lee, A. Alaghi, and L. Ceze, "Correlation manipulating circuits for stochastic computing." *in proc. 2018 DATE*, pp.1417-1422, 2018.

[8] A. Alaghi, C. Li, and J.P. Hayes, "Stochastic circuits for real-time image-processing applications." in *proc. 50th IEEE DAC*, pp.1-6, 2013.

[9] S. Krishnaswamy et al., "Probabilistic transfer matrices in symbolic reliability analysis of logic circuits." *ACM Trans. Design Automation of Elec. Sys.*, vol.13, issue 1, pp.1-35, 2008.

[10] M. Parhi, M. Riedel and K. Parhi, "Effect of bit-level correlation in stochastic computing." *in proc. 2015 IEEE DSP*, pp.463-467, 2015.

[11] T.H. Chen and J.P. Hayes. "Equivalence among stochastic logic circuits and its application to synthesis." *IEEE Trans. ETC*, vol.7, pp.67–79, 2019.

[12] H. Abdellatef, M. Khalil-Hani, and N. Shaikh Husin, "Accurate and compact stochastic computations by exploiting correlation." *Turkish Journal of Elec. Engin. & Comp. Sci.*, vol.27, pp.547-564, 2019.

[13] H. Abdellatef, M. Khalil-Hani, N. Shaikh Husin and S Ayat, "Stochastic computing correlation utilization in convolutional neural network basic functions." *Telkomnika,* vol.16, no.6, pp.2835-2843, 2018.

[14] A. Apicella, et al., "A survey on modern trainable activation functions", *J. Neural Networks*, vol.138, pp.14-32, 2021.