Spectral Ranking in Complex Networks Using Memristor Crossbars

Spectral Ranking in Complex Networks using Memristor Crossbars

Anil Korkmaz, *Graduate Student Member, IEEE,* Gianluca Zoppo, Francesco Marrone, *Graduate Student Member, IEEE,* Su-in Yi, R. Stanley Williams, *Senior Member, IEEE* Fernando Corinto, *Senior Member, IEEE* and Samuel Palermo, *Senior Member, IEEE*

***Abstract*—Various centrality measures have been proposed to identify the influence of each node in a complex network. Among the most popular ranking metrics, spectral measures stand out from the crowd. They rely on the computation of the dominant eigenvector of suitable matrices related to the graph: EigenCentrality, PageRank, Hyperlink Induced Topic Search (HITS) and Stochastic Approach for Link-Structure Analysis (SALSA). The simplest algorithm used to solve this linear algebra computation is the Power Method. It consists of multiple Matrix-Vector Multiplications (MVMs) and a normalization step to avoid divergent behaviours. In this work, we present an analog circuit used to accelerate the Power Iteration algorithm including current-mode termination for the memristor crossbars and a normalization circuit. The normalization step together with the feedback loop of the complete circuit ensure stability and convergence of the dominant eigenvector. We implement a transistor level peripheral circuitry around the memristor crossbar and take non-idealities such as wire parasitics, source driver resistance and finite memristor precision into account. We compute the different spectral centralities to demonstrate the performance of the system. We compare our results to the ones coming from the conventional digital computers and observe significant energy savings while maintaining a competitive accuracy.**

***Index Terms*—Complex networks, Memristor crossbars, Normalization circuit, Power Method, Spectral Centrality.**

I. Introduction

In the past few decades, complex networks theory has attracted considerable attention in numerous real systems applications [1]. These models recast units as nodes of a graph and interactions as connecting edges. Depending on the topological structure of the system, each node has a distinct influence in the diffusion of the information within the graph. For instance, the identification of the most influential nodes within traffic networks can be used as an important reference for improving traffic flow.

Various centrality measures have been proposed to identify the influence of each node based on their number of neighbors' distributions. Among the most popular ranking metrics, spectral measures stand out from the crowd: EigenCentrality, PageRank, Hypertext Induced Topic Selection (HITS) and Stochastic Approach for Link-Structure Analysis (SALSA). These centrality measures evaluate nodes' importance by computing the dominant eigenvector of suitable matrices associated to the graph using the Power (Iteration) Method. This algorithms has a polynomial complexity that depends on the matrix size and relies on simple multiple Matrix-Vector Multiplications (MVMs) [2].

Current von-Neumann architectures shuttle back and forth massive amounts of data between memory and processing units requiring enormous computing hardware resources and large power consumption during the process. Analog circuits based on memristor crossbars are promising platforms capable of highly efficient parallel computations that can be used to accelerate linear algebra operations. The time complexity is independent of the matrix size [3] and computations are performed by exploiting Kirchhoff's and Ohm's circuit laws. Recently, circuits that solve the principal eigenvector's problem have been demonstrated to be more energy efficient compared to conventional computers [3]–[5]. In [3], the solution is restricted by the knowledge of the corresponding eigenvalue and suffers from high op-amp parameters. In [4], two different architectures are offered for the same purpose. The open-loop structure which can be used to perform the Power Iteration algorithm requires a round-trip between analog and digital domains for each iteration. The feedback structure suffers from high op-amp parameter requirements and dependency on matrix condition number. Moreover, this circuit can only be used when computing PageRanks or Markov Chains' principal eigenvector.

The aim of this work is to present and analyze a memristor-based computing system that accelerates the computation of spectral centrality measures by implementing a completely analog hardware version of the Power Method. This work takes [5] as a reference and removes extra circuitry that is limiting energy efficiency. In addition, it also provides a deep analysis of the circuit non-idealities associated with the system. The theoretical background of complex networks and centrality measures are outlined in Section II. Section III describes the current method used to find dominant eigenvectors. The circuit level implementation of the Power Method using memristor crossbars is presented in Section IV. The system performance is evaluated in Section V. Section VI concludes the paper.

This work was supported by the Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) under contract 2017LSCR4K-003 and by the X-Grants Program of the President's Excellence Fund at Texas A&M University.

F. Corinto, G. Zoppo and F. Marrone are with the Department of Electronics and Telecommunications, Politecnico di Torino, Torino, Italy, e–mails: (fernando.corinto@polito.it), (gianluca.zoppo@polito.it), (francesco.marrone@polito.it).

R. S. Williams, S. Palermo, S. Yi and A. Korkmaz are with the Electrical and Computer Engineering Department at Texas A&M, College Station, Texas, USA, e–mails: (rstanleywilliams@tamu.edu), (spalermo@tamu.edu), (yisuin@tamu.edu), (anil.korkmaz@tamu.edu).

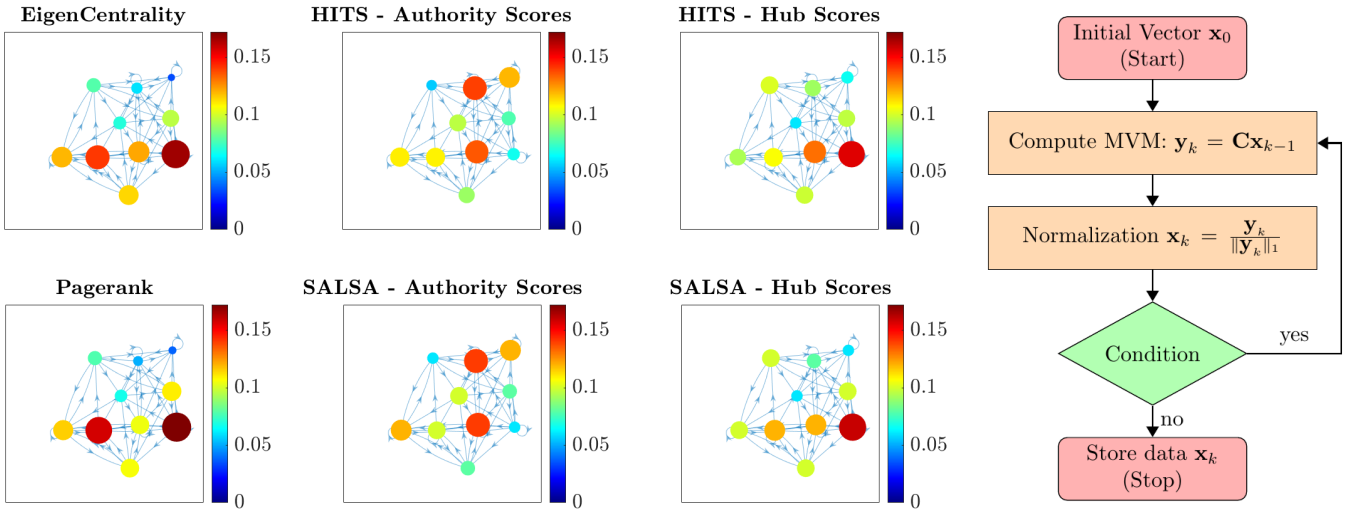


Fig. 1. (Left) Some most common centrality measures exemplified in a small network of 10 nodes. The size of each vertex is proportional to the ranking values provided by the different algorithms. Colors show the influence that each node has in the network. These values are found as components of the dominant eigenvector. The first row shows the non-stochastic spectral centrality measures (EigenCentrality and HITS) and the second row their stochastic counterpart (PageRank and SALSA) that rely on a discrete random walk on the graph. (Right) The block diagram shows the different steps required by the Power Method to compute the dominant eigenvector of a matrix. Starting from an initial condition, the system evaluates after each MVM and normalization a stopping criterion. If the difference between the output of two successive iterations is greater than a tolerance then the process continues, otherwise it stops and stores the data.

II. Spectral Centrality Measures

Traditionally, graph theory gives the mathematical foundation for topologically characterizing complex networks, i.e. mathematical structures used to model pairwise relations between entities. A graph consists in a set of vertices/nodes and a set of lines/edges that connect them. The graph may be undirected whenever there is no distinction between the two vertices associated with each edge, or directed if there is a specific direction between the two nodes. This introduces the necessity of discriminating between two types of links adjacent to a node: incoming and outgoing links.

Drawing a graph is certainly a good start to represent it. However, when the number of nodes and edges is large, the plot may become useless. An alternative representation of a complex network can be attained by using the adjacency matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ where N is the number of nodes. A_{ij} is equal to 1, if nodes i and j are connected, 0 if they are not. If the network is undirected, the adjacency matrix is symmetric. Otherwise, the matrix is asymmetric.

Finding important nodes is of great theoretical and practical interest for many applications. One common assumption about important nodes is that they are crucial for the quick diffusion of the information in the network. Thus, if one wants to spread information quickly, it is necessary to pass that information to a few important nodes. For example, influencers in social networks are important nodes used to diffuse information. Similarly, if one removes just a few important nodes from a network like an essential protein in a biological system, then the whole system may collapse.

The term centrality is a measure of a network that characterizes the importance of nodes and a brief overview is given in Table I. Among the most common measures, spectral centrality stands from the crowd [1]. The computation reduces to determining the dominant eigenvector of suitable matrices that describe the connectivity of the given complex network. Since the computation can be extremely demanding in real-world large scale networks, various approaches have been proposed to fast compute centrality values and approximate the solution. Some of them make use of aggregation techniques that exploit the inherent network's structure and the well-known Power Iteration method to break down the problem into subproblems that are related to the original one [6], [7].

A. Eigenvector Centrality

Eigenvector centrality (or EigenCentrality) states that the importance of a node should be determined by the number and importance of its neighbors (i.e., the degree of the node). Let us consider an undirected graph (i.e. $\mathbf{A} = \mathbf{A}^T$) and consider the vector $\mathbf{x}$ whose entries x_i represent the centrality of a node. This ranking metric assumes that each x_i is proportional to the sum of the neighbors' centralities x_j. This can be mathematically formalized as follows:

$$x_i = \frac{1}{\lambda_{max}} \sum_{j:j\to i} x_j = \frac{1}{\lambda_{max}} \sum_j A_{ij} x_j \tag{1}$$

or in matrix form:

$$\mathbf{x} = \frac{1}{\lambda_{max}} \mathbf{A}\mathbf{x} \tag{2}$$

where λ_{max} is the dominant eigenvalue of the matrix $\mathbf{A}$ and $\mathbf{x}$ is its associated column eigenvector. This measure can also be generalized to the case of a directed graph considering the dominant eigenvector of both matrices $\mathbf{A}$ and $\mathbf{A}^T$.

B. PageRank

PageRank considers a random walk on the graph and assigns a score proportional to the probability of being on each single

Table I: Spectral Measures and their applications.

Spectral Centrality	Properties	Matrix Equation
EigenCentrality	It takes into consideration the number of connections of each vertex and the importance score of their neighbors.	$\mathbf{x} = \frac{1}{\lambda_{max}}\mathbf{A}\mathbf{x}$
PageRank	It can be interpreted as the probability that a random walker lands on each node after taking a large amount of random steps.	$\mathbf{z} = \mathbf{P}\mathbf{z}$
HITS	Given a specified topic, it ranks nodes in two different classes: hubs and authorities based on the incoming and outgoing links of each node.	$\mathbf{a} = c_1\mathbf{A}^T\mathbf{A}\mathbf{a}$ $\mathbf{h} = c_2\mathbf{A}\mathbf{A}^T\mathbf{h}$
SALSA	It is the combination of both PageRank and HITS algorithms.	$\mathbf{a}_s = d_1\mathbf{W}_c^T\mathbf{W}_r\mathbf{a}_s$ $\mathbf{h}_s = d_2\mathbf{W}_r\mathbf{W}_c^T\mathbf{h}_s$

node after a large amount of moves [8]. Let $k_i^{out} = \sum_{j=1}^{N} A_{ij}$ be the number of outgoing links of the i-th node. A random walker at page i can jump with probability

$$P_{ij} = \begin{cases} \alpha\frac{A_{ij}}{k_i^{out}} + (1-\alpha)\frac{1}{N} & \text{if } k_i^{out} \neq 0 \\ \frac{1}{N} & \text{if } k_i^{out} = 0 \end{cases} \tag{3}$$

to each single node j. The uniform probability $P_{ij} = \frac{1}{N}$ is used in the case of nodes with no outlinks. This adjustment enables the transition matrix $\mathbf{P}$ to be a stochastic matrix: all non-negative entries with columns sums equal to 1. The coefficient α is used to make nodes accessible from all other nodes. The PageRank vector $\mathbf{z}$ is the solution of the following linear system

$$z_i = \sum_j P_{ij} z_j \quad \forall i = 1,\ldots,N \tag{4}$$

or in matrix form:

$$\mathbf{z} = \mathbf{P}\mathbf{z}. \tag{5}$$

It can be proved that stochastic matrices have a dominant eigenvalue λ_{max} equal to 1 and the associated eigenvector corresponds with the stationary distribution of a regular Markov Chain [4], [8].

C. Hypertext Induced Topic Selection

The HITS algorithm was developed by Kleinberg [9], who conjectured that web pages have a dual role: they provide information on a certain subject and they are used to link to other pages that give information on that subject. If a web page provides reliable information on a given topic, then it is called an authority. Hub pages link to authorities of that topic. A page is considered to be more authoritative if it is referenced by many hub pages that are relevant to a search query.

Authority and hub score vectors $\mathbf{a}$ and $\mathbf{h}$ can be interpreted as the dominant eigenvectors of the authority matrix $\mathbf{A}^T\mathbf{A}$ and hub matrix $\mathbf{A}\mathbf{A}^T$ respectively:

$$\mathbf{a} = c_1\mathbf{A}^T\mathbf{A}\mathbf{a} \qquad \mathbf{h} = c_2\mathbf{A}\mathbf{A}^T\mathbf{h} \tag{6}$$

where c_1, c_2 are normalizing constants.

D. Stochastic Approach for Link-Structure Analysis

The SALSA algorithm was developed by Lempel and Moran, who combined the random walk idea of PageRank with the hub/authority classification of web pages [10].

Let $\mathbf{W}_r$ be the matrix generated from the adjacency matrix $\mathbf{A}$ by dividing each non-zero entry by its row sum. Let $\mathbf{W}_c$ be generated by dividing each non-zero entry of $\mathbf{A}$ by its column sum. By ignoring rows and columns whose sum is equal to zero we can build the stochastic hub matrix $\mathbf{W}_r\mathbf{W}_c^T$ and the stochastic authority matrix $\mathbf{W}_c^T\mathbf{W}_r$. The stochastic version of authority and hub scores $\mathbf{a}_s$ and $\mathbf{h}_s$ can be interpreted as the dominant eigenvectors of the stochastic authority and hub matrices respectively:

$$\mathbf{a}_s = d_1\mathbf{W}_c^T\mathbf{W}_r\mathbf{a}_s \qquad \mathbf{h}_s = d_2\mathbf{W}_r\mathbf{W}_c^T\mathbf{h}_s \tag{7}$$

where d_1, d_2 are normalizing constants.

III. Analog Implementation of the Power Method

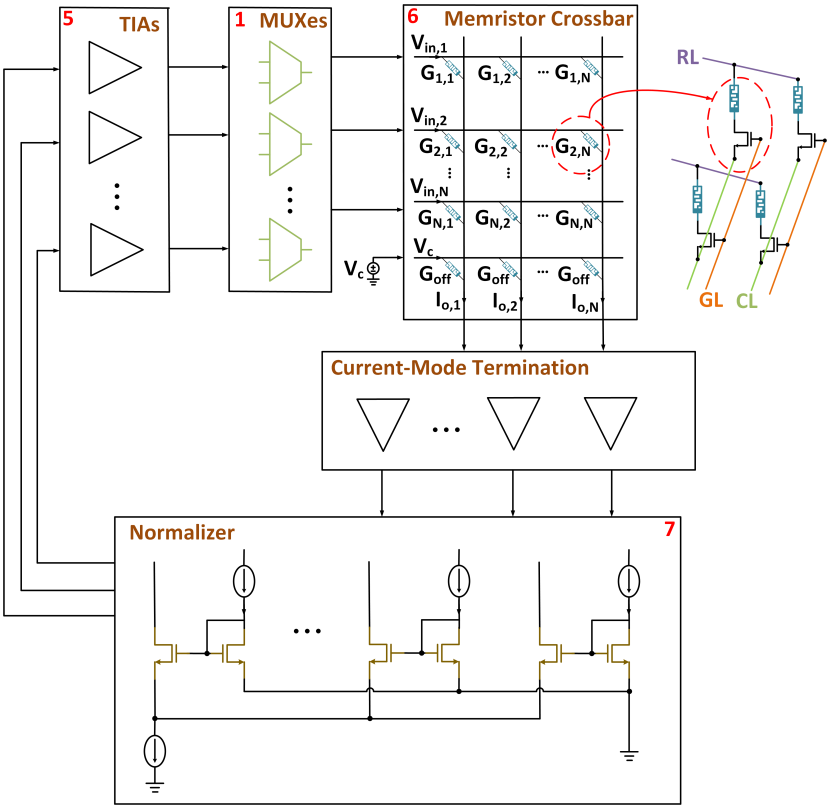
Conventionally, the Power Method has been the preferred approach for computing the dominant eigenvector of a generic matrix $\mathbf{C} \in \mathbb{R}^{N\times N}$. This iterative algorithm starts with an initial condition vector $\mathbf{x}_0 \in \mathbb{R}^N$ and recursively performs a MVM between the matrix $\mathbf{C}$ and the current vector $\mathbf{x}_k$ until a stopping criterion is satisfied:

$$\mathbf{x} = \frac{1}{\lambda_{max}}\mathbf{C}\mathbf{x} \quad \Rightarrow \quad \begin{cases} \mathbf{x}_{k+1} = \eta\mathbf{C}\mathbf{x}_k \\ \mathbf{x}_0 \end{cases} \tag{8}$$

where η is a normalizing constant and the matrix $\mathbf{C}$ is constant for each iteration. To prevent underflow/overflow, the Power Method normalizes the products after each MVM by using a suitable vector norm.

The convergence of the algorithm to the unique dominant eigenvector is guaranteed if $\mathbf{C}$ satisfies suitable conditions provided by the Perron-Frobenius theorem [11], [12]. Moreover, under these assumptions, it can be proved that the dominant eigenvector's components are all positive.

Memristor crossbars have proven their potential to solve MVMs efficiently using the fundamental laws of circuit theory [18]–[23]. For each column $k = 1,\ldots,N$, the system computes by means of the Ohm's Law the physical multiplication between the applied voltage v_j and each memristor's conductance $\mathbf{G}_{jk}$, $\forall j = 1,\ldots,N$. The resulting currents are then summed according to the Kirchhoff's Current Law, i.e. $i_k = \sum_j G_{jk}v_j$. For notation, the crossbar performs the product between $\mathbf{G}^T$



(a)

Algorithm 1 Power Method with normalization

Require: ε tolerance, k_{max} max number of steps

$\mathbf{x}_0 \in \mathbb{R}^N$ initial condition with $\|\mathbf{x}_0\|_1 = 1$

$\mathbf{B} \in \mathbb{R}^{N\times N}$

ERR output error

$k = 0$

while $ERR > \varepsilon$ and $k < k_{max}$ **do**

  $\mathbf{y}_{k+1} = B\mathbf{x}_k$

  $\mathbf{x}_{k+1} = \frac{\mathbf{y}_{k+1}}{\|\mathbf{y}_{k+1}\|_1}$

  $ERR = \|\mathbf{x}_{k+1} - \mathbf{x}_k\|_1$

  $k = k+1$

end while

(b)

CROSSBAR PARAMETERS

Parameter	Value
Conductance Range	1-10 μS
Memristor Precision	4− bits
Interconnect Resistance	0.9 Ω
Memristor Voltage	< 0.3 V
Crossbar Size	100×100
Supply Voltage	1 V
Process Technology	28 nm

(c)

Fig. 2. (a) Analog Power Iteration solver block diagram. Each cell consist of 1T1R. A 100×100 memristor crossbar was used in simulations. RL: Row Line, CL: Column Line, and GL: Gate Line. (b) Power Method iterative process algorithm. Numbers highlighted in red lines match with the different enumerated blocks in the diagram. (c) Crossbar and memristor parameters of this work. A high resistance/low conductance range was chosen for the memristors in this work which is a good compromise between medium resistance TaO_x, HfO_2 memristors (typically $1-10k\Omega$) [13]–[15] and very high resistance $Ag:a-Si$ memristors ($26-325M\Omega$) [16], [17] with the similar ON/OFF ratio. The memristor precison is chosen to be 4-bits to add more practicality to the simulations even though it is demonstrated that both medium and very high resistance memristors can be programmed to $>6-bits$ precision [14], [17]. Interconnect resistance value was obtained using the parasitic extraction tool of the given process technology.

and the column vector $\mathbf{v}$. In the current literature, this operation may also be referred to as Vector-Matrix Multiplication (VMM), i.e. $\mathbf{v}^T\mathbf{G}$. Unless otherwise stated, this work makes use of the MVM notation: $\mathbf{i} = \mathbf{G}^T\mathbf{v}$.

Let us assume we want to compute the MVM defined by $\mathbf{i} = \mathbf{C}\mathbf{v}$. The matrix $\mathbf{C}$ is converted into the finite conductance range of memristors using the linear transformation described in [4], [18], [20]:

$$\begin{aligned} \gamma &= \frac{G_{on} - G_{off}}{A_{max} - A_{min}} \\ \delta &= G_{on} - \gamma A_{max} \\ \mathbf{G} &= \gamma\mathbf{C}^T + \delta\mathbf{1}_N\mathbf{1}_N^T \end{aligned} \tag{9}$$

where G_{on} and G_{off} are the maximum and the minimum memristor conductances, respectively and $\mathbf{1}_N$ is the vector of all ones. This linear transformation is performed before programming each memristor's conductance. Once the output vector from the crossbar $\tilde{\mathbf{i}} = \mathbf{G}^T\mathbf{v}$ is measured, the result of the original MVM can be computed by means of the following inverse operations:

$$\mathbf{i} = \frac{1}{\gamma}\left[\tilde{\mathbf{i}} - \left(\delta\sum_{j=1}^{N} v_j\right)\mathbf{1}_N\right] \tag{10}$$

Thus, after the analog to digital conversion, the output of the MVM can be recovered using Eq. (10) in a digital system.

When dealing with an iterative process such as the one defined by the the Power Method, each iteration requires a recovering step and a subsequent normalization. Performing Eq. (10) after each iteration is a time consuming procedure that requires the knowledge of the sum of the input vector $\sum_{j=1}^{N} v_j$. However, this work considers matrices that obey Perron-Frobenius conditions guaranteeing that the resulting output vector $\mathbf{x}_{k+1} = \mathbf{i}$ of the MVM yields $i_k > 0\ \forall k = 1,\ldots,N$ whenever the input vector $\mathbf{v} = \mathbf{x}_k$ has all positive components. Since the output of the system is then fed back as input to the crossbar due to the iterative process, we have that:

$$\sum_{j=1}^{N} v_j = \sum_{j=1}^{N} |v_j| = \|\mathbf{v}\|_1 \tag{11}$$

for each input vector. If a $1-$norm is used for the normalization step, it follows that

$$\sum_{j=1}^{N} v_j = \|\mathbf{v}\|_1 = 1. \tag{12}$$

As a consequence, Eq. (10) can be simplified in

$$\mathbf{i} = \frac{1}{\gamma}\left(\tilde{\mathbf{i}} - \delta\mathbf{1}_N\right) \tag{13}$$

Since after each iteration, the output defined by the iterative process in system (8) is normalized, it follows that:

$$\frac{\mathbf{i}}{\|\mathbf{i}\|_1} = \frac{\frac{1}{\gamma}\left(\tilde{\mathbf{i}} - \delta \mathbf{1}_N\right)}{\|\frac{1}{\gamma}\left(\tilde{\mathbf{i}} - \delta \mathbf{1}_N\right)\|_1} = \\ = \frac{\frac{1}{\gamma}\left(\tilde{\mathbf{i}} - \delta \mathbf{1}_N\right)}{\frac{1}{\gamma}\|\left(\tilde{\mathbf{i}} - \delta \mathbf{1}_N\right)\|_1} = \\ = \frac{\left(\tilde{\mathbf{i}} - \delta \mathbf{1}_N\right)}{\|\left(\tilde{\mathbf{i}} - \delta \mathbf{1}_N\right)\|_1}$$

thus, the final recovering step after each MVM simply results in the systematic simplified shift

$$\mathbf{i} = \tilde{\mathbf{i}} - \delta \mathbf{1}_N. \tag{14}$$

As can be inferred from Eq. (14), if δ is relatively small, then the eigenvectors of system (8) and

$$\begin{cases} \mathbf{i}_{k+1} = \eta \mathbf{G}^T \mathbf{v}_k = \eta(\mathbf{C} + \delta \mathbf{1}_N \mathbf{1}_N^T)\mathbf{v}_k \\ \mathbf{v}_{k+1} = \frac{\mathbf{i}_{k+1}}{\|\mathbf{i}_{k+1}\|} \\ \mathbf{v}_0 = \mathbf{x}_0 \end{cases} \tag{15}$$

are almost identical. However the correct matching can be found if the recovering step defined in Eq. (14) is performed after each iteration. To avoid the use of a digital system to perform this operation, we added an extra row to the matrix $\mathbf{G}$ and perform the augmented MVM:

$$\mathbf{i} = (\mathbf{G}^T \mathbf{v} - \delta \mathbf{1}_N) = \left[\mathbf{G}^T, \delta \mathbf{1}_N\right] \begin{bmatrix} \mathbf{v} \\ -1 \end{bmatrix} = \tilde{\mathbf{G}}^T \tilde{\mathbf{v}} \tag{16}$$

where $\tilde{\mathbf{v}} \in \mathbb{R}^{N+1}$, $\tilde{\mathbf{G}} \in \mathbb{R}^{N+1,N}$. Here, the multiplicative factor $\frac{1}{\gamma}$ is omitted as previously highlighted in Eq. (14). Thus, by considering the augmented MVM, the output of the system after normalization does not need a recovering step.

This work presents a circuit architecture used to accelerate the Power Method algorithm for solving the MVM operation defined in Eq. (16) and the subsequent normalization of the results. In a recent contribution [5], a sample and hold was used to be able to read and observe each iterated output of the MVM after the normalization step. Here, the circuit depicted in Fig. 2(a) removes the sample and hold block and the clocking circuitry, resulting in a lower circuit complexity, smaller area and less power consumption. The two circuits serve different purposes and can be used for different applications. The former circuit can be used whenever the evolution of the output vector is of particular interest for the user. For example, this system can be exploited to either predict the next state distribution vector of a discrete Markov Chain or to iteratively compute its stationary distribution [4]. Alternatively, Power Iteration Clustering requires knowing the evolution of the intermediate vectors obtained by the successive MVMs of the Power Method for solving problems of community detection and clustering [24], [25]. Whenever the evolution of the system is not necessary, one can compute the dominant eigenvector of a given matrix that satisfies the Perron-Frobenius conditions using the system depicted in Fig. 2(a). This work focuses on the computation of spectral centrality measures using the present architecture, which provides a faster and power efficient solution to the eigenvector problem.

Some of the real-world problems have extensively large number of pages/nodes to rank. To deal with this problem, in [26] a large real-world dataset [27] is divided into sub-networks/sub-blocks to reduce the computation complexity. Then, individual local PageRanks for each sub-block are computed. In a second step, these associated local scores are weighted by the importance of the corresponding blocks. The new local and weighted sub-block rankings are used as a starting vector for the standard PageRank algorithm decreasing the number of iterations significantly. For very large datasets, the proposed circuit can be used to accelerate local and block PageRank steps of the overall algorithm given in [26]. Subsequently, the standard PageRank algorithm can still run in the digital domain after collecting the results from sub-blocks and use them as the initial condition of the iterative process.

Whenever the complex network changes its properties, the proposed system can handle this in two ways: 1) if there are new links appearing or disappearing, we can re-program the memristors to reflect the new topology 2) if a new node is appearing (or disappearing) we can either turn ON (or OFF) column transistor gates in 1T1R cells, or re-perform the BlockRank algorithm to re-compute the new sub-blocks to map into the crossbar.

IV. Circuit Description

In this section, the overall analog Power Method solver circuit architecture will be discussed. The memristor crossbar performs the core vector matrix multiplication operation. The main "acceleration" and thus energy efficiency advantage of this work mainly comes from the crossbar block. A complete system working together with this crossbar is required to implement the Power Method algorithm. In our previous work [4], we have already defined a system which is a hybrid version of the analog and digital systems that can implement the required iterative process. However, the data transfer and conversion between the two systems can decrease the energy efficiency, especially if the number of iterations is high. The accuracy and precision of the system is also affected and reduced due to quantization process of the ADCs which is also the main energy bottleneck for many in-memory computing systems [28]–[31]. To prevent that round trip between the analog and digital systems, we present a completely analog circuit implementation of the Power Method algorithm. This circuit has an overall feedback loop to calculate successive iterations until the convergence is reached.

A. Multiplexers and Crossbar

The first block of the system is composed by Multiplexers (MUXes). The control signal of the MUXes, *INIT*, is responsible for the initialization of the system. When *INIT* is high, the initial voltages are applied to the crossbar. The applications that are going to be presented in the next section use the same amplitude voltage inputs for all crossbar rows. Thus, $V_{initial,1}, \ldots, V_{initial,N}$ can be all tied together to the same DC input bias voltage. This also reduces the circuit complexity

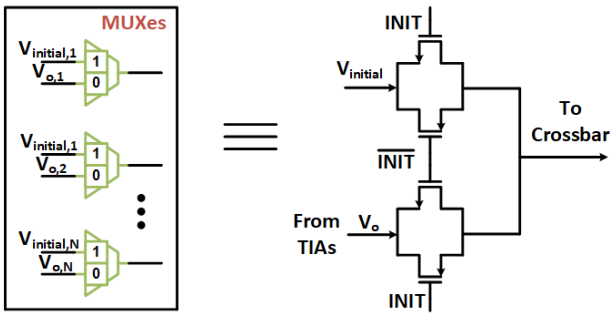


Fig. 3. Multiplexer block diagram on the left and a single multiplexer with the corresponding signal flow. Transmission gates are used as switches. When *INIT* signal goes low, iterations start until the convergence is reached.

significantly and saves extra power due to the very simple input peripheral circuitry requirement. After the *INIT* signal goes low, which can easily be controlled by an external signal provided to the chip pins, the circuit in Fig. 2(a) operates within the feedback loop and self iteration proceeds until the convergence is reached. The $k-$th output of the previous iteration is connected to the $k-$th input row of the crossbar for the next iteration. The transmission gates are used as switches in the analog multiplexer.

The second block of the architecture is the memristor crossbar. The details on the memristors and device programming can be found in our previous and other relevant works [4], [14], [32], [33]. For the inference phase, the selection of the input voltages is important and needs to be done according to the theory provided in Section II. The input vector should be a stochastic vector, i.e. all non-negative components that sum to 1. Let us consider a 100×100 crossbar. If a $1V$ is selected to represent the summation of all inputs, then $10mV$ is required to be provided to each input. However, $10mV$ is a low amplitude input voltage and this signal level is not able to provide a good signal-to-noise ratio compared to noise generated in the system. The other disadvantage of using a very low amplitude input signal is that the system becomes more susceptible to offset voltages arising from op-amps in the current-mode termination (next) block. Because of these reasons, we have selected a $100mV$ input voltage for each row. This introduces a scaling factor of 10 for all the elements related to the system, including the correction scheme and the TIA feedback resistors. From Eq. (16), δ is required to be mapped into the last row memristors' conductance. However, to compensate the larger currents resulting from the $10\times$ input voltage scaling, $10\times$ the conductance value is mapped for the last row, which also results in larger currents. The bias voltage is required to be applied as a correction voltage equal to the sum of the input vector components, which sum to $10V$ after the up-scaling (i.e. $10 \times 1V$). It is not possible to apply $-10V$ input voltage for the last row due to the restrictions from the $1V$ process technology considered here. Moreover, the generation of a negative voltage is also a hard task in chip design and is not desired. For these reasons, a correction voltage is scaled down to the $[0, V_{ref}]$ range. As in the previous case of scaling of input voltages, correction bias voltage scaling also needs to be compensated using memristors in the last row. The summation of the input voltages equals $10V (= 100mV \times 100)$ in our case, but instead of $V_c = -10V$, $V_c = V_{ref} - 250mV = 350mV$ in this work. To compensate that $-10V$ to $-0.25V$ change, $40\times$ higher conductance value needs to be mapped for the last row's memristor.

The conductance range of the memristors can be selected to cover this $40\times$ higher conductance for the last row. For example, if the systematic shift given by the δ parameter is equal to G_{off}, then G_{on} is required to be at least $40 \times G_{off}$, which implies an ON/OFF ratio of at least 40. If the ON/OFF ratio of the memristors is not able to satisfy this criteria, then different material memristors with higher conductance range can be used for the last row. As an alternative way, resistors can be used for the correction row. However, in this case the bias voltage of the last row is required to be programmable to be able to map different δ values and to compensate process variability of resistors, which generally implies the usage of a DAC for programmability with increased circuit complexity. Regardless of which matrix is mapped or which application is in use, the summation of the all input voltages will always be equal to the same voltage because of the normalizer, see Eq. (12). Thus, the bias voltage applied to the last row is not going to change for different applications if the δ parameter is the same:

$$I_{tot} R_F = \sum_i V_{in,i} = \delta V_c. \tag{17}$$

B. Current-Mode Termination

In many papers related to memristor crossbars, the conventional method to terminate the crossbar is the use of transimpedance (TIA) or sense amplifiers [19], [34], [35]. This method is convenient for applications that involve both analog and digital systems because output currents are converted directly to voltages after crossbars and can be read by ADCs afterwards. The TIAs are also able to set crossbar output nodes to the reference voltage for accurate computation and ensure that the voltage across each memristor is $V_{in} - V_{ref}$ between its top and bottom electrodes. However, our system does not have an ADC connection right after the crossbar and most importantly, the next block normalizer requires currents to operate rather than voltages. To pass the crossbar output currents to the next stage and to set the crossbar output nodes to the reference voltage, a current-mode termination circuitry employing a "gain boosted" topology is added. As depicted in Fig. 4(a), a two-stage amplifier with an output PMOS transistor M_0 stage is used. The M_0 transistor shows the characteristics of a common drain amplifier and is non-inverting. It forms a negative feedback with the negative terminal of the op-amp connected to its source node to ensure stability, see Fig. 4(b). The output current from the crossbar sets the V_{GS} value of M_0. If the crossbar output current is excessively high, that can result in a large V_{GS} value, which decreases the V_{DS} voltage of the op-amp's output stage NMOS M_7. This leads to a gain loss for the op-amp and thus introduces gain errors for the reference voltage at the output of the crossbar. To prevent this, a large PMOS transistor $30u/100n$ to produce low V_{GS} voltages even with high currents is used. The other advantage of using a PMOS transistor instead of an NMOS is that crossbar sees

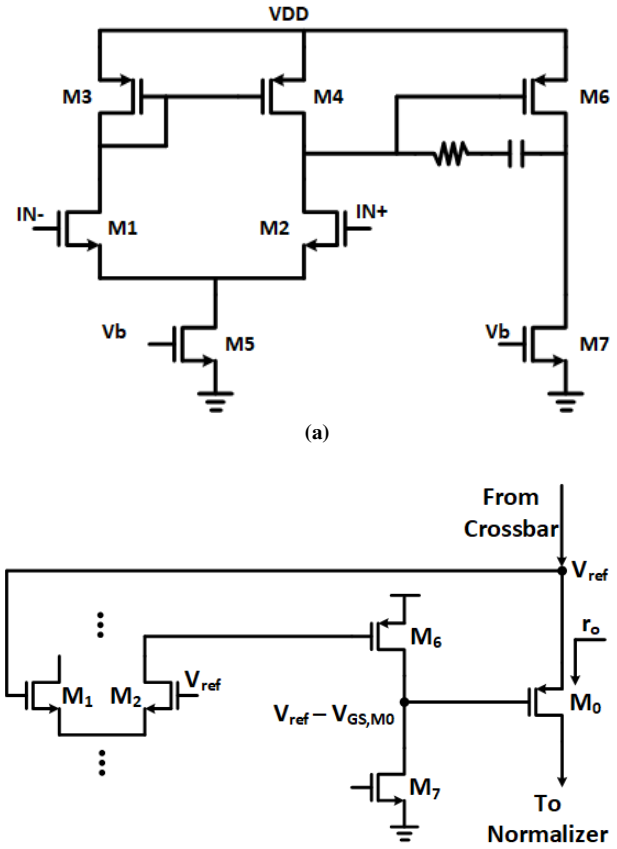


Fig. 4. a) The schematic of the op-amp used in this work. A $62dB$ gain with $1.1GHz$ gain-bandwidth product is obtained while consuming $149\mu W$ power. RC compensation was added to increase the phase margin to 80°. b) The current-mode termination block schematic. The r_o is the low output resistance seen from the crossbar and is approximately equal to $1/g_m$. The V_{DS} voltage of the M_7 transistor is equal to $V_{ref} - V_{GS,M0}$

a very low impedance output via the source terminal of M_0, which approximately has $1/g_m$ output resistance of r_o. The use of a large W/L ratio transistor also increases the g_m value, thus, providing even lower output resistance. A two-stage amplifier employing Miller OTA topology is implemented. A $62dB$ gain with $1.1GHz$ gain-bandwidth product is obtained while consuming $149\mu W$ power.

C. Normalizer

The current-mode termination block passes the current to the next block, normalizer. The normalization is an important step in the Power Method algorithm. It ensures that the algorithm converges to the correct eigenvector of the matrix and prevents divergence. To be able to implement the normalization step, the original Gilbert normalizer [36] and its modified CMOS version [37] are referenced. The Gilbert normalizer makes use of the logarithmic V_{BE} relationship between the BJT transistors Q_1 and Q_2. The voltage difference at the emitter node for each $Q_1 - Q_2$ pair can be calculated from:

$$V_{BE2} - V_{BE1} = V = \frac{kT}{q} ln \frac{I_{xN}}{I_s} - \frac{kT}{q} ln \frac{I_{wN}}{I_s} = \frac{kT}{q} ln \frac{I_{xN}}{I_{wN}} \quad (18)$$

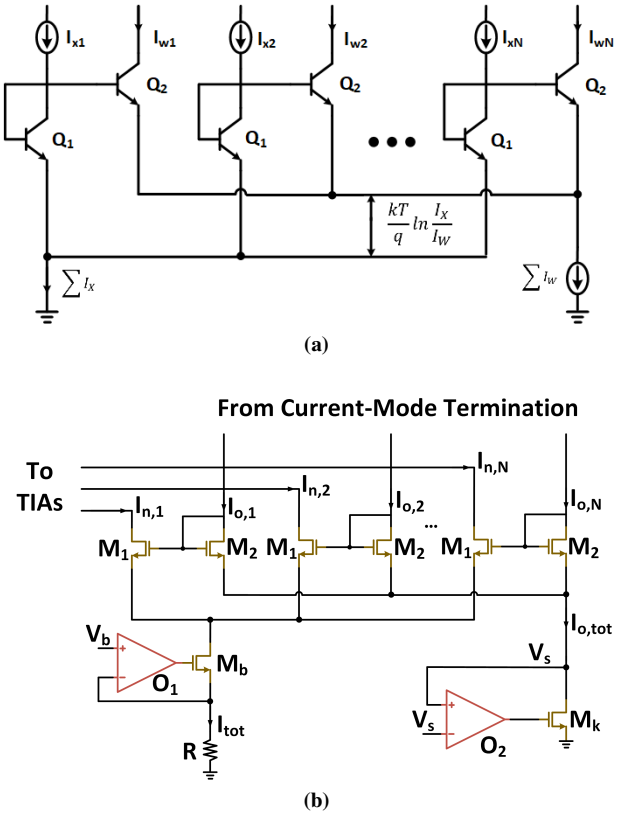


Fig. 5. a) The original Gilbert normalizer cell. It makes use of the exponential relationship between the BJT transistor pairs. b) The CMOS version of the Gilbert normalizer cell. The transistor pairs operate in the sub-threshold region. The summation of the normalized currents is controlled by the V_b and R and is equal to $I_{tot} = V_b/R$.

This voltage difference at the emitter is the same for all pairs, thus forcing the relationship (ratio) between the currents I_1 and I_2 to be the same for all pairs:

$$\frac{I_{x1}}{I_{w1}} = \frac{I_{x2}}{I_{w2}} = \frac{I_{xN}}{I_{wN}}. \quad (19)$$

Since the emitter nodes of all pairs Q_1s and Q_2s are connected together, we can find the following relationship

$$\frac{I_{xN}}{\sum I_x} = \frac{I_{wN}}{\sum I_w}. \quad (20)$$

Even though the sub-threshold region current equations of the MOSFET transistors become more complicated with advanced technology nodes, the fundamental equations still hold. The CMOS transistors operating in the sub-threshold region are used to mimic the Gilbert normalizer. The sub-threshold equations of BJT and CMOS transistors are

$$I_C = I_S \cdot \exp\left(\frac{V_{BE}}{V_T}\right), \quad (21)$$

$$I_D = I_0 \cdot \exp\left(\frac{V_{GS}}{V_T}\right). \quad (22)$$

Thus, the normalization relationship given in Eq. (20) can be reproduced using CMOS transistors as shown in Fig. 5(b):

$$\frac{I_{o,N}}{I_{o,tot}} = \frac{I_{n,N}}{I_{tot}}. \quad (23)$$

Any change in Eq. (22) resulting from the use of advanced nodes will be introduced as an error at the output due to the inaccurate normalization. The transistor M_b together with the op-amp O_1 form a current reference circuit to set the summation of all normalized currents. This total current is equal to $I_{tot} = V_b/R$. In this work, we set the I_{tot} to be $100\mu A$. Correction bias voltage is selected based on the summation of the crossbar input voltages ($100mV \times 100 = 10V$). When I_{tot} is selected to be $100\mu A$ together with the $100k\Omega$ TIA feedback resistors, then the summation of all crossbar input voltages stays at $10V$ even after the normalization step and assures an accurate correction with V_c. It is also a good practice to make V_b programmable to compensate any unforeseen non-idealities together with process variations that may come with the fabrication.

All the source nodes of the M_2 transistors are tied together and connected to the drain of the M_k transistor. The op-amp O_2 is connected using the same topology given in current-mode termination, but this time with an NFET M_k. The op-amp sets the M_k drain node to V_s and assures a sufficient V_{DS} voltage over the M_b transistor due to the relationship given in Eq. (18). The main reason to select an NFET for M_k instead of a PFET is to overcome voltage headroom issues. Some of the crossbar columns can output a high current, which generates a high V_{GS} voltage for the PFET and reduces the overall voltage headroom. The use of an NFET for M_k only adds a V_{DS} voltage instead of V_{GS} and thus can be lower amplitude. The positive terminal of the op-amp O_2 connected to the M_k drain ensures negative feedback, since the NFET output stage is an inverting common source amplifier.

One of the possible issues with this approach is limited voltage headroom, see Fig. 6, due to the use of an advanced node with a low power supply for power efficiency. If the headroom is calculated starting from the V_s node, the total voltage drop between the normalizer and current-mode termination is equal to:

$$0 < V_{DS,M_k} + V_{GS,M_2} + V_{DS,M_0} < V_{ref}. \tag{24}$$

The reference voltage V_{ref} is set to be $0.6V$ in this work, which is a good compromise, almost in the middle of the supply range. In that way, TIA outputs and crossbar inputs can use a wide range from $0.6-1V$. To satisfy the constraint given in Eq. (24), low threshold CMOS transistors were used. If the path with normalizer bias transistor V_b is used, then the overall voltage headroom becomes:

$$0 < V_b + V_{DS,M_b} + V_{GS,M_1} + V_{DS,M_0} < V_{ref}. \tag{25}$$

Using low threshold devices solved headroom issues for this work. If issues arise from the limited headroom, a different and older process technology with high supply voltage or using I/O transistors can relax the headroom issues at the cost of increased power consumption.

It is also a good practice to use high resistance/low conductance memristors in the crossbar, since normalizer transistors operate in the sub-threshold region and smaller currents provide lower V_{GS} voltages. The output current of each crossbar column is matrix dependent, and for applications that have distinct outputs (very low and very high amplitudes), some of the outputs are expected to flow high current. In this case, M_2 transistors can get out of the sub-threshold region and not work as intended, which results in an extra error for the final results. In this work, very large W/L ratio transistors $(30u/100n)$ were used in the normalizer block to assure sub-threshold region operation. The downside of having large transistors is that large parasitic capacitances decrease the stability of the overall feedback loop.

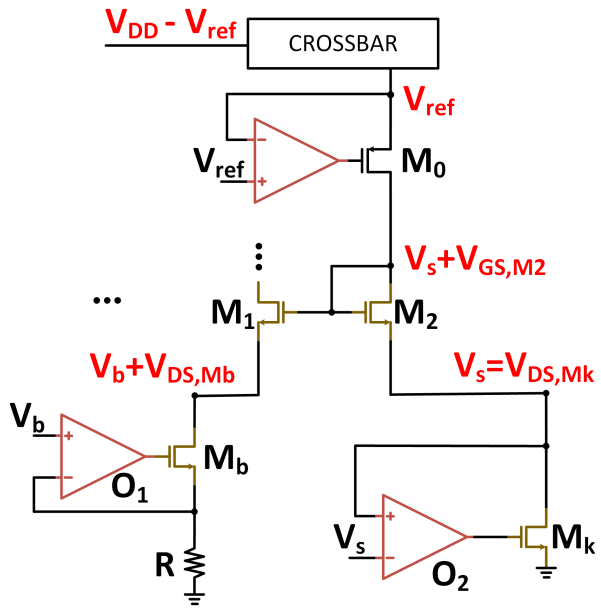


Fig. 6. The demonstration of the headroom situation arising from the combination of all blocks. The two paths emerging from normalizer block were evaluated separately. The V_{DS} voltage over M_0 transistor is equal to V_{ref} - $V_{DS,M_k} + V_{GS,M_2}$. The W/L ratio of the M_0, M_1 and M_2 transistors are selected to be large to decrease the V_{GS} to satify the headroom requirement and to keep them in the sub-threshold region.

The drains of the M_1 transistors in the normalizer block are connected to TIAs. Transimpedance amplifiers convert the normalized currents back to voltage, since the memristor crossbar requires voltage inputs. The same op-amp in Fig. 4(a) is used. As previously mentioned, TIA feedback resistor is selected in accordance with total normalized current flowing through M_b, I_{tot}. The multiplication between I_{tot} and R_f sets the summation of all input voltages for the correction step. The reference voltage of the TIAs are the same with the current-mode termination block, that is $0.6V$. Since the direction of the normalized currents are from TIA to normalizer block, this generates a positive polarity voltage at the output of the TIA that is between V_{ref} and V_{DD}. This sets the input range of the memristor crossbar as well. The output swing of the TIA is practically $0.6-0.9V$, due the V_{DS} voltage drop of the PFET M_6 in Fig. 4(a). This is an advantageous property for the memristors in the crossbar because the linear operation range of the memristors is generally less than $0.3V$ [13], [38], [39]. It is a good practice to keep voltage difference between memristor electrodes less than the switching threshold voltage to ensure linearity. The TIA block completes the feedback loop. Each TIA is then connected to the input "0" of the each multiplexers in Fig. 3.

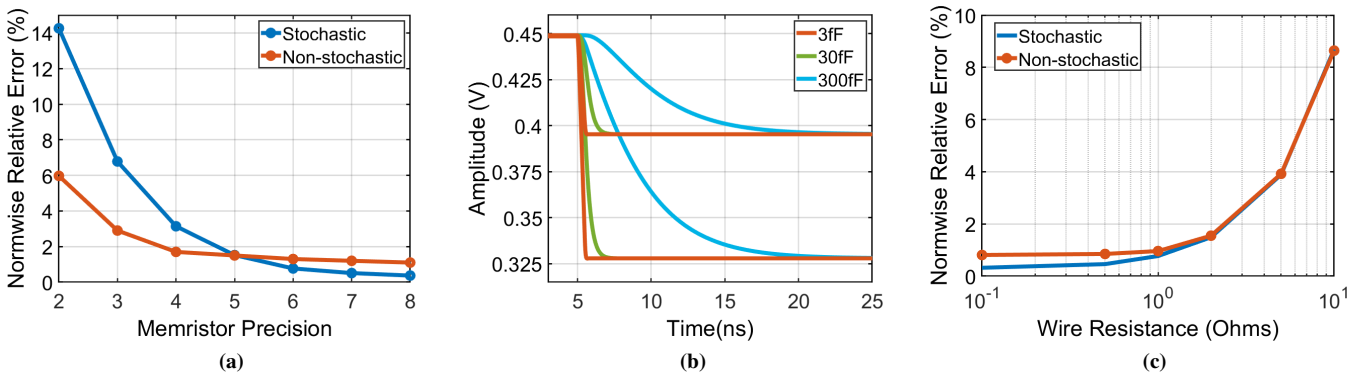


Fig. 7. (a) The normwise relative error at the output was evaluated using different memristor precision for the different matrices, stochastic and non-stochastic. For the memristor precision five bits and more, accuracy does not change significantly. For four bits and less, memristor precision becomes the dominant error source. (b) Different wire capacitances for each cell were simulated for a single crossbar. Two randomly selected output waveforms were used (29th and 98th outputs). If the accuracy is important for an application, wide wires with higher capacitance can be used to decrease the effects of wire resistances but with lower bandwidth. However, if above GHz operation is desired, wire capacitance must be less than a few femtofarads. (c) The effect of the wire resistance on output accuracy. The output accuracy exponentially decreases with higher wire resistance after 1 Ω for both type of matrices.

D. Accuracy Analysis and Non-Idealities

There are circuit non-idealities that affect the overall accuracy and performance of the system such as interconnect resistances and capacitances, source driver resistances, finite gain and bandwidth op-amps, transistor parasitic capacitances, finite memristor precision/states (programming noise), memristor linearity and noise. The parasitic capacitances arising from the circuit implementation mainly affect the stability and settling time of the system by generating undesired poles. In this subsection, the effects of non-idealities and systematic errors are analyzed using a 100×100 crossbar. Each error was considered separately and tested with the absence of other errors.

One of the systematic errors in the system is finite gain op-amps, which introduces a gain error to the architecture and reduces the accuracy. The system has been tested using op-amps with different open-loop gains ranging from $40dB$ to $90dB$ and without any other systematic error, a slight accuracy improvement has been observed, 1.1% output error to 0.5%, respectively. The source driver resistance that drives initial voltages is equal for each crossbar row and the parallel memristor resistance seen by the source driver will also be similar for each row because of the large crossbar size. Thus, the errors arising from the source driver resistance become insignificant. Completely removing the source driver resistance errors requires a buffer between multiplexers and the crossbar. However, our simulation data showed very little improvement resulted from buffers, which were not implemented. The system was simulated with and without any buffer, using a 100Ω and $1k\Omega$ source driver resistance and accuracies of 0.5, 0.52 and 0.53 were obtained, respectively. The source driver resistance is only affected by the initial voltages and thus, is irrelevant within the iterations after the feedback loop is established with the *INIT* signal.

The precision levels of the memristors directly affect the output accuracy [4]. Memristors with higher precision provide more accurate results. Here we tested our system using memristors with different numbers of available states (precision). Let N_b be the number of preferred bit precision, then the programming error is calculated as

$$\sigma = \frac{G_{on} - G_{off}}{6(2^{N_b} - 1)} \tag{26}$$

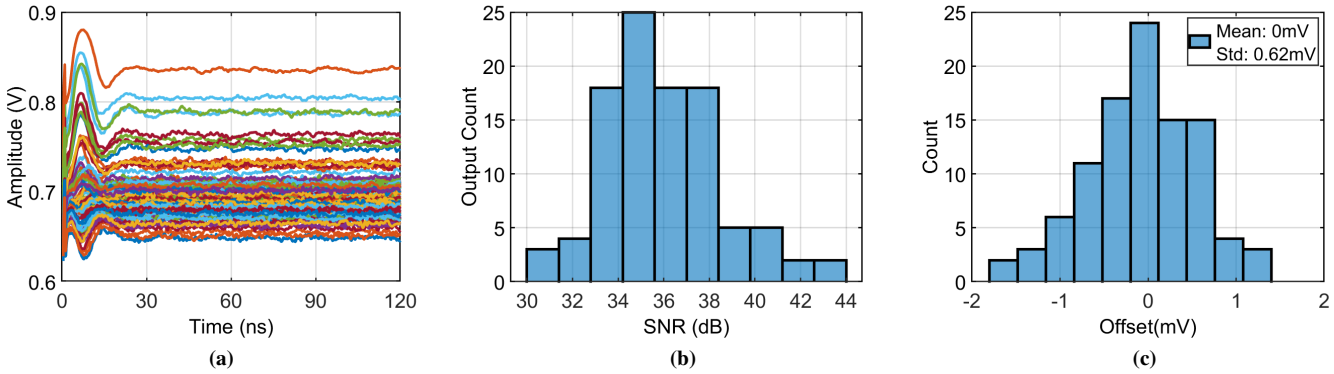
and added to the ideal conductance values of the memristors. The coefficient 6 ensures the separation between the distinct conductance levels ($\pm 3\sigma$).

Let $\tilde{\mathbf{x}}$ be the measured output vector and $\mathbf{x}$ the output provided by a digital software, Fig. 7(a) depicts the total normwise relative error

$$E_r = \frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|}{\|\mathbf{x}\|} \tag{27}$$

as a function of the different memristor precision. As can be seen, the error approximately halves with the increment of the memristor precision by one bit. More interestingly, the normwise relative error at the output is matrix dependent. The use of a non-stochastic matrix with two-bit precision and a stochastic matrix with the same memristor precision resulted in different normwise relative errors, 5.97% and 14.26% in Fig. 7(a). We also compared the importance score rankings for each node in the system using two-bit memristors with a non-stochastic matrix and impressively only the ranking of a single node changed by six places in a 100 node network.

The only remaining errors are related to the interconnects, which the IR drop effect will be less for the first row and column and severe for the last row and column. A comprehensive study is performed for the interconnect resistance and capacitance and their effect on accuracy and settling time in this and previous related works [34], [40]–[42]. To see the wire capacitance settling time effects resulting from the RC delay, the crossbar was separated from the rest of the circuit and tested with ideal peripheral circuitry. The different capacitance values used ranged between $C_w = 3, 30$ and $300fF$, where C_w is the wire capacitance between two adjacent rows or columns. Even with the unrealistically high $C_w = 300fF$, two randomly selected outputs of a 100×100 crossbar settled within $15ns$, see Fig. 7(b). Thus, the wire capacitance has no



(a) (b) (c)

Fig. 8. (a) Noisy output waveforms obtained from the transient noise analysis. σ_{noise} was measured to be $1.5mV$.(b) The signal-to-noise ratio was calculated using signal power and noise power for each output. The mean SNR is calculated to be $36dB$ using ENOB equation. (c) The random offset distribution of the op-amp used in this work. The mean of $\approx 0mV$ and variance of $\sigma = 0.62mV$ are obtained.

significant effect on sub-GHz operation. Then, the memristor crossbar together with the rest of the Power Method solver was simulated. Up to $C_w = 3fF$, no stability issue was observed. However, for the wire capacitances equal or higher than $10fF$, an excessive ringing was observed at the output. To stabilize the system under high C_w conditions, a TIA compensation capacitor value was increased, thus slowing down the overall circuit with a longer settling time.

The IR drop resulting from wire resistances affects the output accuracy. One of the reasons why this work selected a $1-10\mu S$ conductance range (or $100k\Omega - 1M\Omega$ resistance range) was to minimize the effects of the wire resistances [43]. As can be inferred from Fig. 7(c), with 10Ω of wire resistance, the total error at the output exceeds 8%. If a $10-100\mu S$ conductance range was used, then the error arising from a 1Ω wire resistance would have increased by an order of magnitude. For these reasons, it is beneficial to use high resistance memristors or decrease the wire resistance as much as possible by using wider wires. Having wider wires will result in a larger parasitic capacitance for each cell and RC time constant of the crossbar. In addition, the stability of the system should be analyzed. Based on the cell sizes from [13], [19], a parasitic extraction resulted in $R_w = 0.9\Omega$ and $C_w = 3.2fF$ per cell using lower level metal layers. The wire resistance can be decreased more than an order of magnitude by using higher level metal layers at the cost of higher layout complexity.

The last non-ideality arising from the circuit implementation is the thermal noise associated with memristors, resistors and CMOS transistors in the system. A detailed noise analysis can be found in [4]. The signal-to-noise was calculated for a stochastic matrix mapped onto a 100×100 crossbar using transient noise analysis. Noisy waveforms can be found in Fig. 8(a) for 100 outputs. The signal-to-noise ratio for each output was calculated from the signal power divided by the noise power, and results in a mean value of $36dB$. The SNR distribution for the outputs is in the histogram given in Fig. 8(b). The SNR can also be calculated by using the full-scale range of the output swing $(0.9-0.6 = 0.3V)$ divided by the noise variance similar to ADCs [44] which results in $46dB$ SNR. However, the first notation was used in this work.

The current for each memristor is obtained by the multiplication between the voltage across its electrodes and the mapped conductance value. For an accurate calculation of the MVM, this voltage needs to be precisely equal to $V_{in} - V_{ref}$. However, there are several factors that cause V_{ref} to deviate from its ideal value, especially related to the fabrication processes. One of them is the random input offset for the op-amps in the current-mode termination block. The offset mainly arises from the mismatch and process variations of the transistors. There are several ways to reduce the effects of the random offset: 1) offline offset calibration, 2) implementation of bigger devices, 3) use of an advanced technology node, 4) lower overdrive voltage, $V_{GS} - V_{thr}$, for the input transistors, and 5) larger crossbar input voltage amplitude. The offsets for each op-amp in the system can be modelled as random samples from a gaussian distribution and will be different for each column, which reduces the output accuracy.

To reduce the offset, large devices with low overdrive voltage are used for input pairs. The use of $28nm$ technology also helped to obtain a lower offset. The random offset distribution can be found in Fig. 8(c) for the op-amp given in Fig. 4(a). After 100 runs in Cadence, the mean of $\approx 0mV$ and the standard deviation of $0.6mV$ were measured and the resulting distribution was (approximately) Gaussian. The offset is one of the consequences of the mismatch and process variations that can only been seen after chip fabrication or through Monte Carlo simulations. So far, we have only discussed the offset from the input pairs of the op-amps. However, mismatch and process variations will degrade the system accuracy through the mismatch in normalizer transistor pairs and through the variation in op-amp gains. To analyze the effects of these non-idealities, an analysis was performed in Cadence using Monte Carlo simulations to determine the contribution to each block from the mismatch and process variations. The variations arising in each individual block were enabled separately, and the output errors were 1.51% from the current-mode termination, 0.32% from the normalizer, and 1.14% from the TIAs block. Based on these results, the normalizer block mismatch had no significant contribution to the output errors and the biggest contributor was the current-mode termination op-amps. Then every mismatch and process variation was combined and

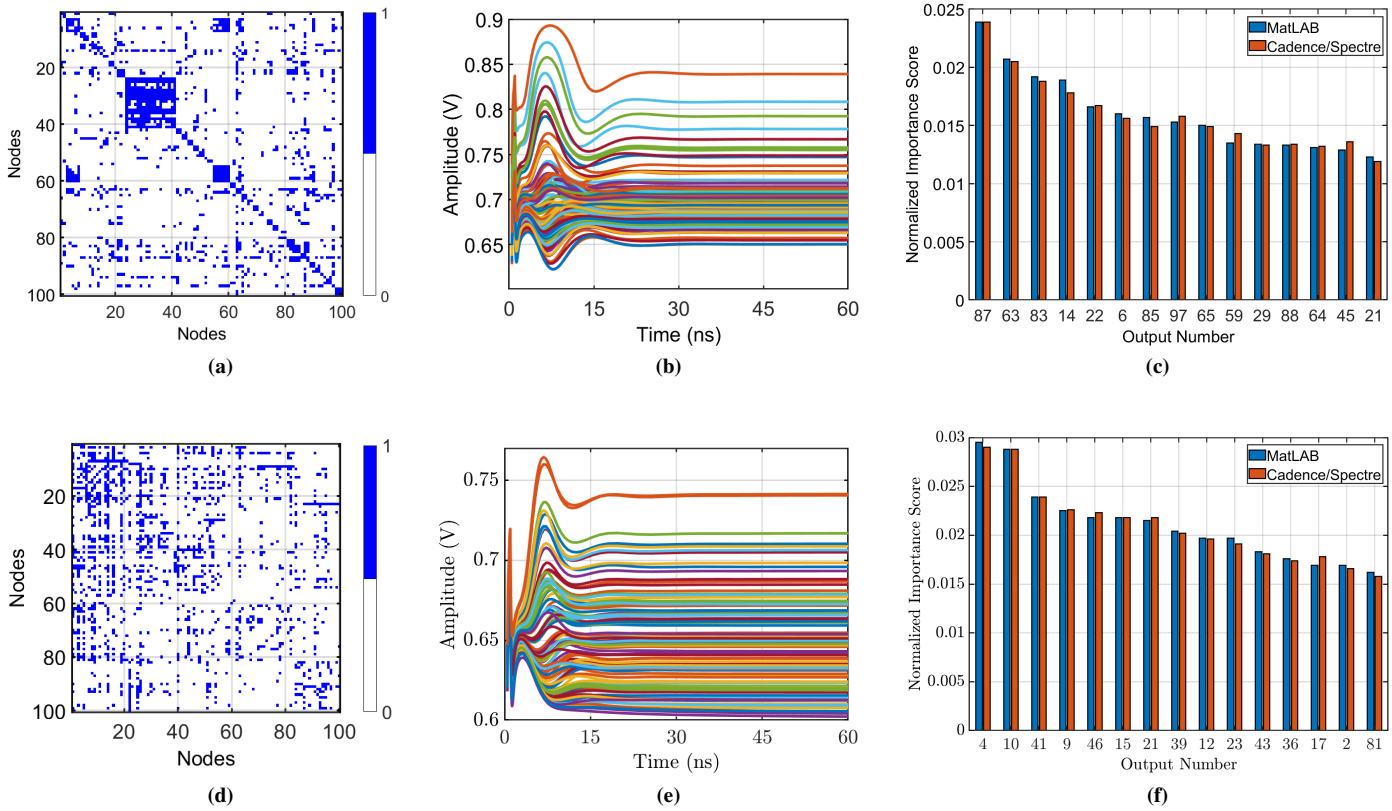


Fig. 9. (a) and (d) Adjacency matrices representing the links between the 100 pages of the Email-EU-core and Higgs Twitter networks, respectively. Each entry A_{ij} is blue if there is a connection between the two nodes i and j, white otherwise. (b) and (e) Transient behaviors of the feedback system converging to the dominant eigenvector of the matrices. The former computes the PageRank of the first network whereas the latter the authorities of the second network. (c) and (f) Comparison between the simulated importance scores provided by the system (in orange) and the correct output provided by `MatLAB` (in blue) ranked from the most important to least important node.

enabled. The total normwise relative error at the output was measured with a mean of 2.1% and standard deviation of 0.2% for 100 runs in Cadence. It was also observed that the error at the output linearly increased with a higher random offset in the system.

V. Case Study

In this section, two examples are illustrated in order to show the system's performance on different applications: PageRank and HITS. The Cadence/Spectre software was used to simulate the circuit and the measured output vector is denoted as $\tilde{\mathbf{x}}$. The overall output accuracy is computed in terms of the normwise relative error between $\tilde{\mathbf{x}}$ and $\mathbf{x}$, and the 'true' output, $\mathbf{x}$, provided by a digital software (i.e. `MatLAB`).

A. Email-EU-core network

The network was generated using anonymous email data from a large European research institution [27]. A member i is connected to another member j if i sent at least one email to j. A high in-degree means the member receives a large amount of e-mails and a high out-degree corresponds to a member who sends plenty of e-mails. We are not only interested in the number of people who are connected to the different members but also on the importance of the people to whom they are connected. As described in Section II, the importance of a member can be specified as proportional to the summation of importance of the immediate members connected to him/her. The goal is to rank the members using the PageRank algorithm. We used a subset of the Email-EU-core network that contains 100 nodes and 1315 edges of the biggest connected component of the original data.

We used a subset of the Higgs Twitter Dataset that contains 100 nodes and 1423 edges of the biggest connected component of the original data. As described in Section II, HITS identifies authorities and hubs for a given selected topic by assigning two numbers to a page: an authority and a hub weight. The goal is to extract users of high influence during this information spreading process. Each output node corresponds to a different Twitter user and the highest amplitude output signal is the most influential account.

Performance Evaluation: The adjacency matrix of the Email-Eu-core network, see Fig. 9(a), was mapped to the conductance range given in Fig. 2(c) using four-bit precision memristors. All the non-idealities and systematic errors discussed in Section IV are included in the circuit, including interconnect resistance and capacitance, source driver resistance, finite gain and bandwidth op-amps, random offset, mismatch and process variations, and finite memristor precision. The resulting output waveforms for the 100 node network can be found in Figure 9(b). To quantify the accuracy, a normwise relative error of

TABLE II
SUMMARY OF RESULTS AND PERFORMANCE COMPARISON

Solver for a 100×100 Matrix	Email-Eu-core Network (This Work)	Higgs Twitter Dataset (This Work)	Iterative Power Method Solver [5]	Haswell CPU [45]	Google TPU [45]
Time-to-solution*	25 ns	45 ns	k** $\times$ 15 ns	$k \times 7.69$ ns	$k \times 0.22$ ns
Power	31.25 mW	31.25 mW	40 mW	145 W	40 W
Energy	0.78 nJ	1.41 nJ	$k \times 0.6$ nJ	$k \times 1.11$ μJ	$k \times 8.8$ nJ
Solutions/s/W	1.28×10^9	0.71×10^9	1.67×10^9 / k	0.89×10^6 / k	0.11×10^9 / k
Output Accuracy (% Error)	2.54 %	3.27 %	2.08 %	0.21 %	0.21 %
Output Bit Precision	$\approx$ 6 bits	$\approx$ 6 bits	$\approx$ 6 bits	8 bits	8 bits

* Time-to-solution metric for digital processors calculated from the number of operations required divided by TOPS/s
** k is the number of iterations to be performed using digital processors until the convergence is reached. The k parameter is equal to 7 for the Higgs Twitter Dataset and 4 for the Email-EU-core Dataset

2.54% was obtained. The accuracy parameter is tightly tied to individual memristor precision in the crossbar. The outputs settled within 0.1% error of their final value in $25ns$.

The use of high resistance memristors reduces the power consumption of the crossbar. The 200 op-amps (Current-mode termination and TIA blocks) consumed 96% of the power in the system, and the remaining 4% was linked to the crossbar and normalizer. The total power consumption was:

$$\sum_{i=1}^{200}(P_{opamp,i}) + P_{crossbar} + P_{normalizer} \approx 31.25mW \qquad (28)$$

Since the power consumption is directly tied to the op-amps in the system, it is possible to reduce it by more power efficient op-amps at the cost of the system accuracy. The energy consumption of the interconnect capacitance ($0.5C_wV^2$) was measured to be negligible. If the $10\times$ lower resistance range memristors are chosen for the application, for example $10-100\mu S$, then the power consumption of the crossbar, $P_{crossbar}$, increases by $10\times$ and brings the total power consumption to $42.5mW$. The SNR analysis was performed for this application and a mean value of $37dB$ was obtained, which corresponds to six-bit precision using the effective number of bits (ENOB) calculation [44].

To determine the accuracy, ideal results obtained from `MatLAB` software were compared to transistor level simulation results. A comparison of the importance scores for the first 15 nodes can be found in Fig. 9(c). The ranking of the first 5 nodes was correct and the highest shift in importance ranking was only 3 ranks (the 45th output was shifted three ranks up).

B. Higgs Twitter Dataset

A social network, such as Twitter, can be defined as a graph for which nodes represent users and edges their relationships. The identification of influential users has always been of particular interest. Several influence measures can be considered. This analysis focuses on the application of the HITS algorithm. Rather than analysing the entire web structure, we are only interested in a single topic: the spread of information after the discovery of a new particle that showed similar characteristics as the Higgs boson [27]. We aim to monitor social relationships among users involved in re-tweets, replies to existing tweets and mentions of other users. In this setting, a user i is connected to another user j if one of the previous actions mentioned took place.

Performance Evaluation: The adjacency matrix of the Higgs boson Twitter Dataset, see Figure 9(d), was mapped to the memristor crossbar using the mapping algorithm given in Section III. The major difference between the two adjacency matrices discussed here can be observed in Figs. 9(a) and (d). The EU mail dataset has self connections for each node i. However, this feature cannot be found in the Twitter dataset, since users are less likely to re-tweet themselves. If any user does re-tweet themselves, then it creates a connection within the same node i. The output waveforms representing the influence scores can be found in Figure 9(e). The outputs have settled within 0.1% of their final value in $45ns$. No significant difference was observed between the two algorithms (HITS for Twitter dataset and PageRank for EU-mail dataset) in terms of settling time. The same power consumption was obtained for the two applications, because op-amps consume the highest percentage of the power and it is constant for different matrices. The output precision analysis also provided very similar results with a mean of $34dB$ signal-to-noise ratio. To determine the accuracy of the system, a normwise relative error of 3.27% was obtained with four-bit memristor precision. Similar comparison between the ideal versus simulation importance rankings were also performed for the Twitter dataset in Figure 9(f). The first 11 nodes were ranked correctly, and the highest shift in importance ranking was only three ranks (the 81st output was shifted three ranks down).

C. Discussion

After obtaining the results using both algorithms and different dataset, a performance comparison between the crossbar architecture and its digital counterparts was performed in Table 2. In contrast with our previous architecture that can also solve the Power Method [4], the present circuit does not require analog-to-digital conversion between iterations, thus offers further time and energy savings. In many systems that utilize memristor crossbars, the main energy efficiency bottleneck is the ADCs. The present architecture makes use of the same amplitude voltage for all its inputs, and thus it does not require digital-to-analog conversion even if it is used in a hybrid system as an analog processing core. The performance of the digital systems, TPU and CPU [45], considered for different final accuracies, depended on the different number of iterations k performed. The energy efficiency is given in terms of Solutions/s/W. For the present architecture, a total

number of iterations is not meaningful because it directly converges to the final eigenvector. In Table 2, the analog Power Method solver provides 45× and 5750× energy efficiency compared to the TPU and 18 core Haswell CPU, respectively. The circuit in this work, also provides 3× energy efficiency compared to the iterative Power Method solver circuit [5]. The digital processors achieve higher output accuracy due to their eight-bit digital numbers compared to the four-bit precision of the memristors used here. A correlation between the settling time in the analog Power Method solver and the number of iterations using digital processors was also observed. The HITS algorithm with the Higgs Twitter dataset converged 1.8× slower compared to the PageRank algorithm with the Email-EU-core dataset in this work. Analogously, convergence required 7 iterations for the Higgs Twitter dataset compared to 4 iterations for the Email-EU-core dataset, which corresponds to 1.75× more iterations. The overall optimization of the system for Power Iteration algorithm provides significant area savings by only occupying $0.0053mm^2$ for the peripheral circuitry. The area distribution is 75% for current-mode termination and TIA blocks, 21% for normalization, and 4% for multiplexers. The area of the memristor crossbar depends on the material and process technology and examples can be found in [18], [19], [46]–[48].

VI. Conclusion

An analog Power Iteration solver to compute the dominant eigenvector of suitable matrices was presented. Four different applications were described that can be implemented on the solver, namely EigenCentrality, PageRank, HITS and SALSA. The architecture consists of purely analog elements, does not require successive iterations and directly calculates the dominant eigenvector, which provides significant energy savings. A detailed analysis of the non-idealities arising from experimental demonstrations such as interconnect resistances and capacitances, source driver resistances, finite gain and bandwidth op-amps, transistor parasitic capacitances, finite memristor precision/states (programming noise), memristor linearity and noise was performed. Their effects on output accuracy and settling time were demonstrated. The two case studies, the Email-EU-core network and Higgs boson Twitter dataset, were used to illustrate the system performance and the effectiveness of the analog solver. The results show that the one-shot solver can be significantly more energy efficient than their digital counterparts. The analog solver ranked the complex network nodes with an acceptable accuracy using low precision memristors.

References

[1] R. Albert and A.-L. Barabási, "Statistical mechanics of complex networks," *Reviews of modern physics*, vol. 74, no. 1, p. 47, 2002.

[2] A. Quarteroni, R. Sacco, and F. Saleri, *Numerical mathematics*. Springer Science & Business Media, 2010, vol. 37.

[3] Z. Sun *et al.*, "Solving matrix equations in one step with cross-point resistive arrays," *Proceedings of the National Academy of Sciences*, vol. 116, no. 10, pp. 4123–4128, 2019.

[4] G. Zoppo, A. Korkmaz, F. Marrone, S. Palermo, F. Corinto, and R. S. Williams, "Analog solutions of discrete markov chains via memristor crossbars," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 12, pp. 4910–4923, 2021.

[5] A. Korkmaz, G. Zoppo, F. Marrone, , R. S. Williams, F. Corinto, and S. Palermo, "Analog acceleration of the power method using memristor crossbars," *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2022.

[6] A. N. Langville and C. D. Meyer, "Deeper inside pagerank," *Internet Mathematics*, vol. 1, no. 3, pp. 335–380, 2004.

[7] A. Z. Broder, R. Lempel, F. Maghoul, and J. Pedersen, "Efficient pagerank approximation via graph aggregation," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, 2004, pp. 484–485.

[8] A. N. Langville and C. D. Meyer, *Google's PageRank and beyond: The science of search engine rankings*. Princeton university press, 2011.

[9] J. M. Kleinberg, "Authoritative sources in a hyperlinked environment," *Journal of the ACM (JACM)*, vol. 46, no. 5, pp. 604–632, 1999.

[10] R. Lempel and S. Moran, "The stochastic approach for link-structure analysis (salsa) and the tkc effect," *Computer Networks*, vol. 33, no. 1-6, pp. 387–401, 2000.

[11] A. Berman and R. J. Plemmons, *Nonnegative matrices in the mathematical sciences*. SIAM, 1994.

[12] A. Farahat, T. LoFaro, J. C. Miller, G. Rae, and L. A. Ward, "Authority rankings from hits, pagerank, and salsa: Existence, uniqueness, and effect of initialization," *SIAM Journal on Scientific Computing*, vol. 27, no. 4, pp. 1181–1201, 2006.

[13] J. P. Strachan, A. C. Torrezan, F. Miao, M. D. Pickett, J. J. Yang, W. Yi, G. Medeiros-Ribeiro, and R. S. Williams, "State dynamics and modeling of tantalum oxide memristors," *IEEE Transactions on Electron Devices*, vol. 60, no. 7, pp. 2194–2202, 2013.

[14] E. J. Merced-Grafals, N. Davila, N. Ge, J. P. Strachan, and R. S. Williams, "Repeatable, accurate, and high speed multi-level programming of memristor 1t1r arrays for power efficient analog computing applications," *Nanotechnology*, vol. 27, no. 36, 2016.

[15] A. Hardtdegen, C. La Torre, F. Cüppers, S. Menzel, R. Waser, and S. Hoffmann-Eifert, "Improved switching stability and the effect of an internal series resistor in hfo¡sub¿2¡/sub¿/tio¡sub¿¡italic¿x¡/italic¿¡/sub¿ bilayer reram cells," *IEEE Transactions on Electron Devices*, vol. 65, no. 8, pp. 3229–3236, 2018.

[16] S. H. Jo, T. Chang, I. Ebong, B. B. Bhadviya, P. Mazumder, and W. Lu, "Nanoscale memristor device as synapse in neuromorphic systems," *Nano Letters*, vol. 10, no. 4, pp. 1297–1301, 2010.

[17] B. Wu, D. Feng, W. Tong, J. Liu, C. Wang, W. Zhao, and M. Peng, "Reram crossbar-based analog computing architecture for naive bayesian engine," in *2019 IEEE 37th International Conference on Computer Design (ICCD)*, 2019, pp. 147–155.

[18] C. Li *et al.*, "Analogue signal and image processing with large memristor crossbars," *Nature Electronics*, vol. 1, no. 1, p. 52, 2018.

[19] M. Hu *et al.*, "Memristor-based analog computation and neural network classification with a dot product engine," *Advanced Materials*, vol. 30, no. 9, p. 1705914, 2018.

[20] M. Hu *et al.*, "Dot-product engine for neuromorphic computing: Programming 1t1m crossbar to accelerate matrix-vector multiplication," in *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2016, pp. 1–6.

[21] M. Hu, J. P. Strachan, Zhiyong Li, R. Stanley, and Williams, "Dot-product engine as computing memory to accelerate machine learning algorithms," in *2016 17th International Symposium on Quality Electronic Design (ISQED)*, 2016, pp. 374–379.

[22] A. Shafiee *et al.*, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 14–26.

[23] A. Ankit *et al.*, "PUMA: A programmable ultra-efficient memristor-based accelerator for machine learning inference," *CoRR*, vol. abs/1901.10351, 2019.

[24] F. Lin and W. W. Cohen, "Power iteration clustering," in *ICML*, 2010.

[25] W. Yan, U. Brahmakshatriya, Y. Xue, M. Gilder, and B. Wise, "p-pic: Parallel power iteration clustering for big data," *Journal of Parallel and Distributed computing*, vol. 73, no. 3, pp. 352–359, 2013.

[26] S. Kamvar, T. Haveliwala, C. Manning, and G. Golub, "Exploiting the block structure of the web for computing pagerank," 04 2003.

[27] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[28] Y. Kim, Y. Zhang, and P. Li, "A digital neuromorphic vlsi architecture with memristor crossbar synaptic array for machine learning," in *2012 IEEE International SOC Conference*, 2012, pp. 328–333.

[29] Xiaoxiao Liu *et al.*, "A heterogeneous computing system with memristor-based neuromorphic accelerators," in *2014 IEEE High Performance Extreme Computing Conference (HPEC)*, 2014, pp. 1–6.

[30] S. Ambrogio, P. Narayanan, H. Tsai *et al.*, "Equivalent-accuracy accelerated neural-network training using analogue memory," *Nature*, vol. 558, pp. 60–67, 2018.
[31] A. Ankit *et al.*, "Panther: A programmable architecture for neural network training harnessing energy-efficient reram," *IEEE Transactions on Computers*, vol. 69, no. 8, pp. 1128–1142, 2020.
[32] F. Alibart, L. Gao, B. D. Hoskins, and D. B. Strukov, "High precision tuning of state for memristive devices by adaptable variation-tolerant algorithm," *Nanotechnology*, vol. 23, no. 7, 2012.
[33] L. Gao, I.-T. Wang, P.-Y. Chen, S. Vrudhula, J. sun Seo, Y. Cao, T.-H. Hou, and S. Yu, "Fully parallel write/read in resistive synaptic array for accelerating on-chip learning," *Nanotechnology*, vol. 26, no. 45, p. 455204, oct 2015.
[34] C. Xu *et al.*, "Overcoming the challenges of crossbar resistive memory architectures," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, 2015, pp. 476–488.
[35] X. Liu *et al.*, "Reno: A high-efficient reconfigurable neuromorphic computing accelerator design," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6.
[36] B. Gilbert, "A monolithic 16-channels analog array normalizer," *IEEE Journal of Solid-State Circuits*, vol. 19, no. 6, pp. 956–963, 1984.
[37] M. V. Nair, L. K. Muller, and G. Indiveri, "A differential memristive synapse circuit for on-line learning in neuromorphic computing systems," *Nano Futures*, vol. 1, no. 3, p. 035003, Dec. 2017.
[38] Y. V. Pershin and M. Di Ventra, "Practical approach to programmable analog circuits with memristors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 8, pp. 1857–1864, 2010.
[39] K. Fleck, C. La Torre, N. Aslam, S. Hoffmann-Eifert, U. Böttger, and S. Menzel, "Uniting gradual and abrupt set processes in resistive switching oxides," *Phys. Rev. Applied*, vol. 6, p. 064015, Dec 2016.
[40] A. Dozortsev, I. Goldshtein, and S. Kvatinsky, "Analysis of the row grounding technique in a memristor-based crossbar array," *International Journal of Circuit Theory and Applications*, vol. 46, no. 1, pp. 122–137, 2018.
[41] B. Liu, H. Li, Y. Chen, X. Li, T. Huang, Q. Wu, and M. Barnell, "Reduction and ir-drop compensations techniques for reliable neuromorphic computing systems," in *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 63–70.
[42] S. Agarwal, R. L. Schiek, and M. J. Marinella, "Compensating for parasitic voltage drops in resistive memory arrays," in *2017 IEEE International Memory Workshop (IMW)*, 2017, pp. 1–4.
[43] S.-i. Yi, S. Kumar, and R. S. Williams, "Improved hopfield network optimization using manufacturable three-terminal electronic synapses," 2021. [Online]. Available: https://arxiv.org/abs/2104.12288
[44] B. Razavi, *Principles of data conversion system design*. IEEE Press, 1995.
[45] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *2017 ACM/IEEE 44th Annual International Symposium on Computer Architecture (ISCA)*, 2017, pp. 1–12.
[46] C. Li, D. Belkin, and Y. Li, "Efficient and self-adaptive in-situ learning in multilayer memristor neural networks," *Nat Commun*, vol. 9, no. 2385, 2018.
[47] M.-F. Chang *et al.*, "19.4 embedded 1mb reram in 28nm cmos with 0.27-to-1v read using swing-sample-and-couple sense amplifier and self-boost-write-termination scheme," in *2014 IEEE International Solid-State Circuits Conference (ISSCC)*, 2014, pp. 332–333.
[48] Q. Liu, B. Gao, P. Yao, D. Wu, J. Chen, Y. Pang, W. Zhang, Y. Liao, C.-X. Xue, W.-H. Chen, J. Tang, Y. Wang, M.-F. Chang, H. Qian, and H. Wu, "33.2 a fully integrated analog reram based 78.4tops/w compute-in-memory chip with fully parallel mac computing," in *2020 IEEE International Solid- State Circuits Conference (ISSCC)*, 2020, pp. 500–502.

Anil Korkmaz (S'09) received the B.Sc. degree in Electronics Engineering from Istanbul Technical University, Istanbul, Turkey, in 2014, and the M.Sc. degree in Electrical Engineering from Southern Illinois University Edwardsville, IL, USA, in 2019. He is currently working toward the Ph.D. degree at Texas A&M University, College Station, TX, USA. His research interests include energy efficient analog & mixed-signal systems, in-memory computing, neuromorphic computing systems.

Gianluca Zoppo received the B.Sc. degree in Mathematics and the M.Sc. degree in Stochastics and Data Science from Università di Torino, Italy, in 2016 and 2018, respectively. He received the Ph.D. degree in the Electronics and Telecommunication Department at Politecnico di Torino in 2022. His research interests include nonlinear dynamics, neuromorphic computing and MCMC optimization.

Francesco Marrone (S'10) received the B.Sc. and M.Sc. degrees in Electronics Engineering from Politecnico di Torino, Italy, in 2015 and 2018, respectively. He is currently pursuing the Ph.D. degree in the Electronics and Telecommunication Department at Politecnico di Torino. His research interests include nonlinear dynamics, neuromorphic computing and memristive technologies.

Su-in Yi received the Ph.D. degree in mechanical engineering from Texas A&M University. He is a Post-Doctoral Research Fellow with the Electrical and Computer Engineering Department, Texas A&M University. He has been a Staff Engineer with the Semiconductor Research and Development Center, Samsung Electronics, South Korea, since 2018. His research interests are neuromorphic device compact modeling, including charge trap flash memories and thermoelectric devices.

R. Stanley Williams is the Hewlett Packard Enterprise Chair Professor in the Electrical and Computer Engineering Department at Texas A&M University. His research includes materials with nonlinear dynamical properties that can provide new types of device and circuit properties for computation.

Fernando Corinto (M'96-SM'10) is Professor of Electrical Engineering with the Department of Electronics and Telecommunications at the Politecnico di Torino. His research activities are mainly on nonlinear circuits and systems, locally coupled nonlinear/nanoscale networks and memristor nanotechnology.

Samuel Palermo (S'98-M'07-SM'17) received the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA in 2007. In 2009, he joined the Electrical and Computer Engineering Department of Texas A&M University where he is currently an associate professor. His research interests include high-speed electrical and optical interconnect architectures and neuromorphic computing systems.