ADVANCED INTELLIGENT SYSTEMS
Open Access
www.advintellsyst.com

In-Memory Eigenvector Computation in Time *O*(1)

Zhong Sun, Giacomo Pedretti, Elia Ambrosi, Alessandro Bricalli, and Daniele Ielmini**

In-memory computing with cross-point resistive memory arrays has gained enormous attention to accelerate the matrix-vector multiplication in the computation of data-centric applications. By combining a cross-point array and feedback amplifiers, it is possible to compute matrix eigenvectors in one step without algorithmic iterations. Herein, time complexity of the eigenvector computation is investigated, based on the feedback analysis of the cross-point circuit. The results show that the computing time of the circuit is determined by the mismatch degree of the eigenvalues implemented in the circuit, which controls the rising speed of output voltages. For a dataset of random matrices, the time for computing the dominant eigenvector in the circuit is constant for various matrix sizes; namely, the time complexity is *O*(1). The *O*(1) time complexity is also supported by simulations of PageRank of real-world datasets. This work paves the way for fast, energy-efficient accelerators for eigenvector computation in a wide range of practical applications.

1. Introduction

Cross-point resistive memory arrays have been intensively utilized to accelerate the matrix-vector multiplication (MVM),[1] which is an elementary operation in several algebraic problems, for instance, the training and inference of neural networks,[2,3] signal and image processing,[4,5] and the iterative solution of linear systems[6] or differential equations.[7] In such implementations, the cross-point MVM is executed for several iteration cycles according to the algorithmic workflow, which might raise an issue in terms of processing time and energy efficiency of the computation. Recently, a cross-point memory circuit architecture has been proposed and demonstrated for solving matrix equations in one step, including solving linear systems and computing eigenvectors.[8] Although the one-step solution capability can solve the inefficiencies of the iterative approach, the underlying time complexity of the circuit needs to be rigorously evaluated to assess the computing performance.

Dr. Z. Sun, Dr. G. Pedretti, Dr. E. Ambrosi, Dr. A. Bricalli, Prof. D. Ielmini
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano and IU.NET
Piazza L. da Vinci 32, 20133 Milano, Italy
E-mail: zhong.sun@polimi.it; daniele.ielmini@polimi.it

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202000042.

DOI: 10.1002/aisy.202000042

Eigenvector calculation is a fundamental problem in a broad scope of computing scenarios, e.g., webpage ranking,[9] facial recognition,[10] and dynamic analysis and solving differential equations in fields such as physics and chemistry.[11] In the conventional computing paradigm, the dominant eigenvector (the eigenvector corresponding to the largest eigenvalue) of a matrix can be calculated using the power iteration method with a time complexity of $O(kN^2)$, where N is the matrix size and k is the number of iterations.[12] In this work, we show that the time complexity of the cross-point memory circuit for computing eigenvectors is $O(1)$; namely, the time to calculate a matrix eigenvector does not depend on the matrix size. Based on the feedback theory of operational amplifiers (OAs), we develop a numerical model to analyze the time response of the eigenvector circuit. Our time-dependent simulation results show that the time to calculate an eigenvector of an $N \times N$ matrix does not explicitly depend on N; i.e., the time complexity is $O(1)$ for our circuit. We find that the computational time is governed by the highest eigenvalue of an associated matrix, which, in turn, is controlled by the mismatch degree between the practical and the nominal conductance values, which implement the eigenvalue in the circuit. The $O(1)$ time complexity is further supported by circuit simulations of the calculation of dominant eigenvectors of random matrices and of PageRank evaluation for the Harvard 500 database and its subsets.

2. Results and Discussion

Computing an eigenvector means solving the matrix equation

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \tag{1}$$

where $\boldsymbol{A}$ is a square matrix, λ is an eigenvalue of $\boldsymbol{A}$, and $\boldsymbol{x}$ is the unknown eigenvector corresponding to λ. To solve Equation (1), matrix $\boldsymbol{A}$ is mapped by the conductance matrix $\boldsymbol{G_A}$ of a cross-point memory array, which plays the role of feedback network in the closed-loop circuit (**Figure 1**a). The feedback configuration is enabled by transimpedance amplifiers (TIAs) and analog inverters. The conductance G_λ of the feedback resistors in the TIAs represents the eigenvalue λ. As an analog eigenvalue cannot be exactly implemented, the practical eigenvalue in the circuit is termed λ_G. At the steady state, assuming that the output voltages of inverters constitute a column vector $\boldsymbol{v}$, the cross-point MVM currents are $\boldsymbol{i} = \boldsymbol{G_A}\boldsymbol{v}$ because of the virtual ground of the inverting-input node of TIAs. TIAs convert the cross-point MVM

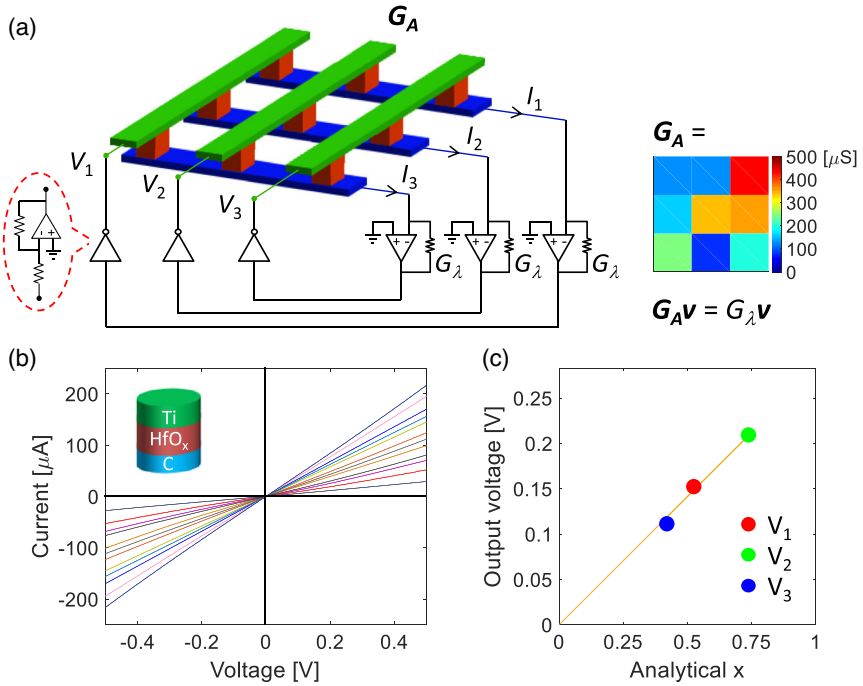

Figure 1. Eigenvector computation with a cross-point RRAM circuit. a) The cross-point RRAM circuit for computing the dominant eigenvector of a positive matrix. The circuit structure of the analog inverter is also shown. The output voltages of analog inverters form a vector $\boldsymbol{v} = [V_1; V_2; V_3]$ representing the eigenvector solution. The cross-point MVM currents form $\boldsymbol{i} = [I_1; I_2; I_3]$. A representative matrix is also shown, with a conductance unit of 100 μS. b) Twelve analog conductance levels of the Ti/HfO_x/C RRAM device, with values of 60, 90, 120, 150, 190, 210, 240, 290, 310, 340, 390, and 420 μS, respectively. The conductance shows a good linearity for all levels below 0.5 V. c) The dominant eigenvector of the matrix computed by the circuit, as a function of the analytical solution.

current into voltages, namely, $\boldsymbol{v}_{\mathrm{TIA}} = -\boldsymbol{G_A v}/G_\lambda$. Finally, $\boldsymbol{v}_{\mathrm{TIA}}$ is inverted by analog inverters, thus yielding resulting voltages $\boldsymbol{v}$, which satisfies the relationship $\boldsymbol{G_A v}/G_\lambda = \boldsymbol{v}$ or $\boldsymbol{G_A v} = G_\lambda \boldsymbol{v}$. The former translates Equation (1) into the voltage $\boldsymbol{v}$, which provides the eigenvector solution within the memory circuit. As a result, Equation (1) is physically solved in one step by the circuit where $\boldsymbol{v}$ represents the eigenvector. Note that λ is real and positive in Figure 1a, which is always the case for the largest eigenvalue of a positive matrix, according to the Perron–Frobenius theorem.[13] For the negative λ case, the inverters in the circuit should be removed, and the absolute value of λ is mapped by the feedback conductance G_λ.[8] In Figure 1a, we consider a positive matrix, because the conductance of a resistive memory device can only be positive. For matrices containing negative elements, two cross-point arrays are needed to split the matrix with two positive matrices.[8]

The in-memory calculation of eigenvectors was conducted in an array of resistive switching memory (RRAM) devices. In the RRAM device, the conductance can be changed by the formation and the dissolution of a conductive filament by local migration of ionized defects.[14] The RRAM conductance can be continuously tuned, thus enabling the analog storage in a cross-point array for in-memory matrix computation.[5,8] Figure 1b shows 12 conductance levels of the adopted Ti/HfO_x/C RRAM device by controlling the compliance current during the set transition. As an experimental demonstration, the dominant eigenvector of a 3×3 positive matrix was computed by the circuit, with the programmed conductance matrix of a cross-point array shown in the inset of Figure 1a. The experimental eigenvector results are shown in Figure 1c as a function of the normalized analytical eigenvector obtained by software calculations with floating-point precision. The linear relation between the two solutions demonstrates the circuit functionality of one-step eigenvector computation. By defining the solution error as $\varepsilon = |\boldsymbol{x} - \boldsymbol{x}^*|$, where $\boldsymbol{x}$ and $\boldsymbol{x}^*$ are the experimental and the ideal normalized eigenvector, respectively, and $\|\cdot\|$ is the Euclidean norm, an error $\varepsilon = 0.0303$ is found in Figure 1c. The 12 discrete conductance levels in Figure 1b were used to build matrices of various sizes for simulating the eigenvector circuit. For instance, the dominant eigenvectors of ten randomly constructed 10×10 matrices were computed in a simulation program with integrated circuit emphasis (SPICE) circuit. In all cases, the results are highly consistent with the analytical solutions, with an average error of $\varepsilon = 0.0056$ (Figure S1, Supporting Information).

In the conventional power iteration method, MVM is executed through element-wise multiply–accumulate operations and a number of iterations are required, resulting in a high computational complexity.[12,15,16] On the other hand, the MVM is instantaneously executed in the eigenvector circuit by physical laws in the cross-point array, whereas discrete iterations are eliminated in favor of a higher computational speed.

To analyze the time complexity, the eigenvector circuit is illustrated as a block diagram (**Figure 2**a), where $\boldsymbol{x}$ is the eigenvector

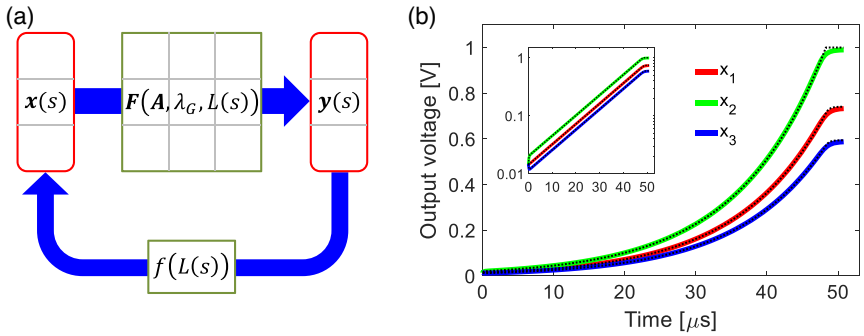

Figure 2. Time response of the eigenvector circuit. a) Block diagram of the eigenvector circuit, where $\boldsymbol{x}$ and $\boldsymbol{y}$ represent the output voltages of analog inverters and TIAs, respectively. The transfer function $\boldsymbol{F}$ from $\boldsymbol{x}$ to $\boldsymbol{y}$ is a matrix, whereas the transfer function f feeding $\boldsymbol{y}$ back to $\boldsymbol{x}$ is a scalar. b) Transient simulation results of computing the eigenvector in Figure 1c. The color full lines are SPICE simulation traces, whereas the dot lines are the iterative simulation result according to the FD algorithm. The inset shows the same results on a half-logarithmic plot to highlight the exponential increase in the output voltages.

solution mapped by the output voltages of the inverters and $\boldsymbol{y}$ represents the output voltages of TIAs. Due to the cross-point RRAM array acting as a feedback network, the transfer function linking $\boldsymbol{x}$ to $\boldsymbol{y}$ should be a matrix that involves the stored coefficient matrix $\boldsymbol{A}$, the mapped eigenvalue λ_G, and the open-loop gain $L(s)$ of the OAs, which is a function of the complex frequency s. The transfer function linking $\boldsymbol{y}$ backward to $\boldsymbol{x}$ is a scalar that is related solely to $L(s)$. In particular, according to Kirchhoff's voltage law and amplifier theory, $\boldsymbol{x}$ and $\boldsymbol{y}$ satisfy the following two equations.

$$-\boldsymbol{U}[\boldsymbol{A}\boldsymbol{x}(s)+\lambda_G\boldsymbol{y}(s)]L(s)=\boldsymbol{y}(s) \tag{2}$$

$$-\frac{\boldsymbol{x}(s)+\boldsymbol{y}(s)}{2}L(s)=\boldsymbol{x}(s) \tag{3}$$

where $\boldsymbol{A}$ is the $N\times N$ coefficient matrix and $\boldsymbol{U}$ is a diagonal matrix defined as $\boldsymbol{U}=\mathrm{diag}\left(\frac{1}{\lambda_G+\sum_i A_{1i}},\frac{1}{\lambda_G+\sum_i A_{2i}},\cdots,\frac{1}{\lambda_G+\sum_i A_{Ni}}\right)$. We assumed that the OAs in both the TIAs and the inverters are identical; thus, the same $L(s)$ applies to all the OAs. Combining Equation (2) and (3) leads to

$$\boldsymbol{U}(\boldsymbol{A}-\lambda_G\boldsymbol{I})L(s)\boldsymbol{x}(s)=(2\lambda_G\boldsymbol{U}+\boldsymbol{I})\boldsymbol{x}(s)+\frac{2\boldsymbol{x}(s)}{L(s)} \tag{4}$$

where $\boldsymbol{I}$ is the $N\times N$ identity matrix. Considering the single-pole OA model,[17] namely, $L(s)=\frac{L_0}{1+\frac{s}{\omega_0}}$, where L_0 is the direct current (DC) open-loop gain and ω_0 is the 3 dB bandwidth, Equation (4) becomes

$$\boldsymbol{U}(\boldsymbol{A}-\lambda_G\boldsymbol{I})\boldsymbol{x}(s)=\frac{2\lambda_G\boldsymbol{U}+\boldsymbol{I}}{L_0\omega_0}s\boldsymbol{x}(s)+\frac{2}{L_0^2\omega_0^2}s^2\boldsymbol{x}(s) \tag{5}$$

where the insignificant terms have been omitted, due to the fact that L_0 is usually much larger than 1. The inverse Laplace transform of Equation (5) implies the second-order differential equation in the time domain, that is

$$\frac{d^2\boldsymbol{x}(t)}{\mathrm{d}t^2}=\frac{1}{2}L_0^2\omega_0^2\boldsymbol{U}(\boldsymbol{A}-\lambda_G\boldsymbol{I})\boldsymbol{x}(t)-\left(\lambda_G\boldsymbol{U}+\frac{1}{2}\boldsymbol{I}\right)L_0\omega_0\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} \tag{6}$$

which describes the time response of the eigenvector circuit. To study the computing time of the circuit, Equation (6) is converted into the first-order differential equation[18] by defining

$$\boldsymbol{z}(t)=\frac{2}{L_0\omega_0}\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} \tag{7}$$

which leads to

$$\frac{\mathrm{d}\boldsymbol{z}(t)}{\mathrm{d}t}=L_0\omega_0\boldsymbol{U}(\boldsymbol{A}-\lambda_G\boldsymbol{I})\boldsymbol{x}(t)-L_0\omega_0\left(\lambda_G\boldsymbol{U}+\frac{1}{2}\boldsymbol{I}\right)\boldsymbol{z}(t) \tag{8}$$

Equation (7) and (8) are merged as one, which reads

$$\frac{d}{\mathrm{d}t}\begin{bmatrix}\boldsymbol{x}(t)\\ \boldsymbol{z}(t)\end{bmatrix}=L_0\omega_0\begin{bmatrix}\boldsymbol{0} & \frac{1}{2}\boldsymbol{I}\\ \boldsymbol{U}(\boldsymbol{A}-\lambda_G\boldsymbol{I}) & -\left(\lambda_G\boldsymbol{U}+\frac{1}{2}\boldsymbol{I}\right)\end{bmatrix}\begin{bmatrix}\boldsymbol{x}(t)\\ \boldsymbol{z}(t)\end{bmatrix} \tag{9}$$

where $\boldsymbol{0}$ is the $N\times N$ zero matrix. By defining the $2N\times 2N$ matrix $\boldsymbol{M}$ according to

$$\boldsymbol{M}=\begin{bmatrix}\boldsymbol{0} & \frac{1}{2}\boldsymbol{I}\\ \boldsymbol{U}(\boldsymbol{A}-\lambda_G\boldsymbol{I}) & -\left(\lambda_G\boldsymbol{U}+\frac{1}{2}\boldsymbol{I}\right)\end{bmatrix} \tag{10}$$

which is associated with matrix $\boldsymbol{A}$, and defining a $2N\times 1$ vector as $\boldsymbol{w}(t)=\begin{bmatrix}\boldsymbol{x}(t)\\ \boldsymbol{z}(t)\end{bmatrix}$, Equation (9) becomes

$$\frac{\mathrm{d}\boldsymbol{w}(t)}{\mathrm{d}t}=L_0\omega_0\boldsymbol{M}\boldsymbol{w}(t) \tag{11}$$

According to the finite difference (FD) method, Equation (11) can be expressed as

$$\boldsymbol{w}(t+\Delta t)=(\boldsymbol{I}_{2N}+\alpha\boldsymbol{M})\boldsymbol{w}(t) \tag{12}$$

where $\boldsymbol{I}_{2N}$ is the $2N\times 2N$ identity matrix, Δt is the incremental time, and α is a dimensionless constant defined as $\alpha=L_0\omega_0\Delta t$.

For Equation (11) to have a nontrivial solution, the spectral radius of matrix $\boldsymbol{I}_{2N}+\alpha\boldsymbol{M}$ has to be larger than 1, which implies that the highest eigenvalue (or real part of eigenvalue) λ_h of matrix $\boldsymbol{M}$ must be positive, assuming that the eigenvalues of $\boldsymbol{M}$ are ranked in a descending order according to their real parts. This condition on λ_h is satisfied if the implemented

λ_G is slightly smaller than the largest eigenvalue of $\boldsymbol{A}$, namely, $\lambda_G = (1-\delta)\lambda_{\max}$, where δ is a small positive number and $\lambda_{\max}$ is the largest eigenvalue; thus, the dominant eigenvector of $\lambda_{\max}$ can be computed by the circuit. According to the FD algorithm of Equation (12), the eigenvector solution is boosted as long as the circuit is powered, until the boundary condition is encountered, i.e., upon reaching the supply voltage V_{supp} of the OAs. The parasitic noise in the circuit provides the initial solution $\boldsymbol{x}(0) \neq 0$ at $t = 0$, which initiates the transient evolution of the circuit in Equation (12). At the steady state, the first N elements of $\boldsymbol{w}$ constitute the dominant eigenvector, whereas the last N elements that represent the time derivative of $\boldsymbol{x}(t)$ are zero.

To assess the circuit dynamics, we simulated the time evolution of $\boldsymbol{x}(t)$ from the FD model in Equation (12) for the experimental matrix in Figure 1c. Figure 2b shows the simulation results, where we assumed $\delta = 0.01$ and $V_{\text{supp}} = \pm 1$ V for generality. A lower V_{supp} or a diode connected to the output of the OAs might be used to protect the cross-point memory devices against a possible voltage disturb if necessary. The results of the FD simulations are fully consistent with the SPICE circuit simulation results, demonstrating a good description of the circuit dynamics by Equation (12). The half-logarithmic plot in the inset evidences the exponential increase in the output voltages, which can be explained by a power iteration where the output voltage is regenerated at each cycle in the closed-loop feedback circuit. The physical power iteration process stops at the saturation of the largest output voltage, after which point all other output voltages quickly stabilize and provide the final eigenvector solution.

According to Equation (12), the speed of the eigenvector circuit is controlled by λ_h of matrix $\boldsymbol{M}$, namely, the larger the value of λ_h, the faster the computation. To study the computing time of the circuit, we conducted a series of simulations by varying the eigenvalue difference δ for the eigenvector computation in Figure 1c. **Figure 3** shows the computed saturation time as a function of δ in the range from 0.003 to 0.06. In the simulations, the initial solution of each output was assumed as $x_i(0) = 0.001$.

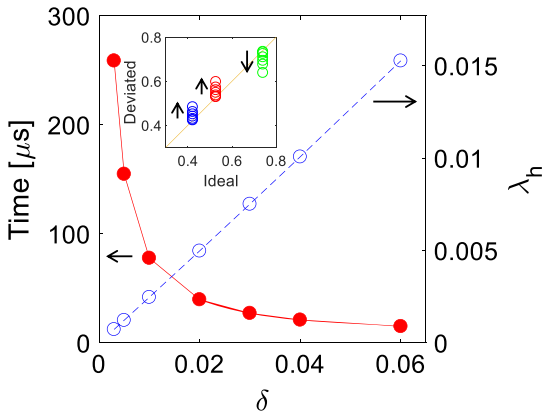

Figure 3. Analysis of λ_h of the associated matrix $\boldsymbol{M}$ and computing time of the circuit. Various mismatch degrees δ were introduced in the eigenvalue implementation in circuit simulations. The computing time is proportional to $1/\delta$, whereas the value of λ_h is proportional to δ. The inset shows the correlation plot between the simulated eigenvectors and the ideal eigenvector, with the arrows indicating the increase in δ.

The computing time is defined as the time at which point the dynamic solution approaches the steady-state solution with an error of <0.1%. It decreases with the inverse of δ, with the time being 15.2 μs for the case of $\delta = 0.06$. The value of λ_h for each case is also summarized in the figure, which shows a linear dependence on δ, thus supporting the dominant role of λ_h in controlling the computing time of the circuit. Though the circuit gets faster for a larger δ, the resulting eigenvector deviates more from the ideal solution, as shown in the inset of Figure 3. The solution error ε increasing with δ is shown in Figure S2, Supporting Information. These results indicate a tradeoff between the computing speed and the solution accuracy, which should be addressed according to the specific application scenario.

To study the dependence of computing time on matrix size N, we constructed a series of random matrices using the 12 conductance levels in Figure 1b. The size ranges from 3×3 to 30×30, as shown in **Figure 4**a, with three representative matrices. For each size, 100 matrices were randomly generated and simulated in the circuit. The dominant eigenvector of every matrix was computed in simulation by assuming different values of δ, namely, $\delta = 0.003$, 0.01, 0.02, and 0.04. Figure 4b shows the computing time as a function of N: the computing time is independent of N, thus demonstrating the $O(1)$ time complexity of the circuit for eigenvector computation. The computing time decreases for increasing δ, in agreement with the results in Figure 3. For the 100 different matrices with the identical N and δ, the computing time distribution is very tight, which further supports the dominant role of δ in controlling the computing time. Note that λ_h increases with δ, which is also consistent with the results in Figure 3. The solution errors are also independent of N (see Figure S3, Supporting Information), thus supporting the scalability of time/energy efficiency gain combined with no accuracy loss.

One concern about the computing time analysis is the parasitic wire resistance in the cross-point array.[19] To investigate the impact of wire resistance on the time complexity of the circuit, we considered the interconnect parameters at 65 nm adopted

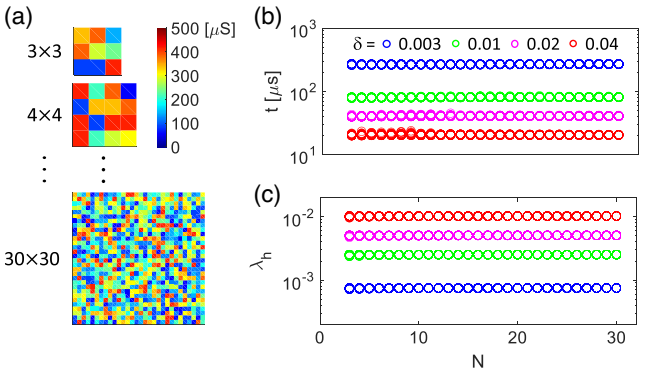

Figure 4. Time complexity for computing dominant eigenvectors of random matrices. a) Illustrations of random matrices with sizes from 3×3 to 30×30; 100 matrices were constructed with the 12 discrete conductance levels in Figure 1b for each size. b) Computing time of the dominant eigenvectors of the matrices with different sizes, with four different δ values introduced in the eigenvalue implementation. c) Values of λ_h of all simulation cases. Both the computing time and λ_h show a clear independence on the matrix size N, indicating the $O(1)$ time complexity.

from the International Technology Roadmap for Semiconductors (ITRS) table.[20] For the same dataset in Figure 4, the circuit simulations including wire resistances were conducted. The computing time for matrices with different sizes and values of δ is shown in Figure S4, Supporting Information. Compared with the ideal circuit, the real circuit including parasitic resistances generally shows a longer computing time, which may show a legible N-dependence. On the other hand, the solution error becomes lower for the real circuit, suggesting that wire resistances equivalently reduce the mismatch degree δ of eigenvalue implementation in the circuit. These results indicate an additional constraint on the tradeoff between the computing time and solution accuracy imposed by the wire resistance issue.

As a practical case study, we addressed the PageRank of a real-world dataset. The PageRank algorithm is widely used for ranking webpages in search engines,[9] and link prediction and recommendation in social media.[21] PageRank aims at calculating the dominant eigenvector,[22] which can be naturally accelerated by the cross-point eigenvector circuit. We adopted the Harvard500 database,[23] which contains 500 relevant webpages of the Harvard University to be ranked according to their connections. In the PageRank of a webpage network, the citations among webpages give a citation matrix $\boldsymbol{C}$, which is defined as follows: if page j contains a link to page i, the citation element C_{ij} is set to 1, otherwise $C_{ij} = 0$. More pages citing the same page indicate that the latter is more important. Also, citation by important pages gives rise to the importance of the page. **Figure 5**a shows the citation matrix of Harvard500, which is a sparse logical matrix. To rank the webpages by their importance, a transition matrix $\boldsymbol{T}$ is defined according to

$$T_{ij} = \begin{cases} \frac{pC_{ij}}{\sum_i C_{ij}} + \sigma, & \text{if} \quad \sum_i C_{ij} \neq 0, \\ 1/N, & \text{if} \quad \sum_i C_{ij} = 0 \end{cases} \tag{13}$$

where $N = 500$ is the number of pages, $p = 0.85$ is the random walk probability, and $\sigma = \frac{1-p}{N}$ is the probability for randomly picking a page. A uniform probability $1/N$ is assigned if a page gets no link.[23] The transition matrix is basically a stochastic matrix with the largest eigenvalue always being 1 and the dominant eigenvector giving the importance scores of webpages.[22] The resulting transition matrix for Harvard500 is shown in Figure S5, Supporting Information.

The transition matrix was stored in the cross-point array, and the largest eigenvalue was mapped in the feedback conductance with a mismatch degree δ to compute the eigenvector in the circuit. Figure 5b shows the circuit dynamics of the eigenvector computation with $\delta = 0.01$, indicating a computing time of around 135 µs. The simulated eigenvector values were normalized, so that the sum of all elements is equal to one, thus giving the importance scores of the webpages. The results are shown in Figure 5c, where page #1 (the home page of Harvard university) ranks in the first place in the search results.

To study the time complexity of PageRank for webpage networks with different sizes, we selected the first N pages in the Harvard500 database to form a new network, for which a new

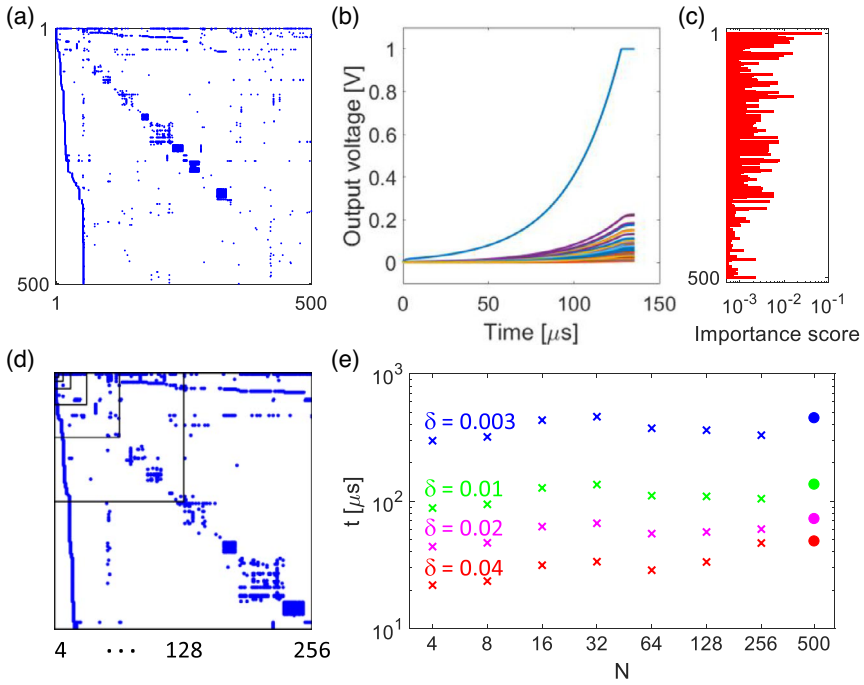

Figure 5. Time complexity of PageRank with the Harvard500 database and its subsets. a) The citation matrix of Harvard500, which is a sparse matrix containing 2636 connections among webpages in the entire database. b) Transient simulation results of PageRank for Harvard500 assuming $\delta = 0.01$. c) Importance scores of the 500 webpages obtained from the simulated eigenvector. d) Citation matrices of subsets of the Harvard500 database, with different sizes from 4×4 to 256×256. e) Computing time of PageRank of Harvard500 and its subsets, with four different δ values assumed. For each δ, the computing time of PageRank of various datasets is on the same level, thus supporting the $O(1)$ of the cross-point circuit for eigenvector computation.

set of importance scores is computed. $N = 4, 8, 16, 32, 64, 128$, and 256 were assumed, and the $N \times N$ submatrix was extracted from the entire citation matrix, as shown in Figure 5d. For each subset of webpages, $\delta = 0.003, 0.01, 0.02$, and 0.04 were assumed for the eigenvalue implementation in the circuit. The computing time of all simulation cases is reported in Figure 5e, which also includes the computing time for the original Harvard500 with the four values of δ. For a specific δ, the computing time for all matrices is approximately at the same level, whereas a different δ causes a significant change in the computing time. The scattering of the computing time over the N range is due to the variation of λ_h with N (Figure S6, Supporting Information), which, in turn, is due to the specific structure of the transition matrices. These results support the $O(1)$ time complexity of the eigenvector circuit for practical application of PageRank.

Regarding the solution accuracy of PageRank, we show the comparison between the simulated importance scores and the ideal ones for the Harvard500 database in Figure S7, Supporting Information, indicating a good consistency between the two solutions. In particular, we ranked the top ten pages for the ideal case and the four simulated cases, showing that all the ideal top ten pages are preserved in the top ten places in the simulations, except for the case with $\delta = 0.04$, where one page was missed out. We also studied the wire resistance issue for the PageRank of Harvard500 subsets, with the results shown in Figure S8, Supporting Information. The parasitic wire resistance causes a small increase in the computing time for relatively large δ, thus leading to an N-dependence to the time complexity of eigenvector computation. These results suggest a careful choice of δ for circuit implementation to achieve the best performance regarding both the computing time and the accuracy of the results. A strategy of dynamic tuning of δ might be adopted to achieve both high speed and accuracy. In this algorithm, a large δ can be used in the initial phases to accelerate the transition of the output voltages; then, δ can be reduced in the later stages for fine-tuning of the final solution.

As the mismatch degree δ is generally considered to be small to maintain the eigenvector accuracy, it may suffer from the conductance variation, i.e., the feedback conductance values of the TIAs being slightly different. In this case, the associated matrix of Equation (10) becomes

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{0} & \frac{1}{2}\boldsymbol{I} \\ \boldsymbol{U}(\boldsymbol{A} - \boldsymbol{\Lambda}) & -(\boldsymbol{\Lambda}\boldsymbol{U} + \frac{1}{2}\boldsymbol{I}) \end{bmatrix} \tag{14}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix that is defined as $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_G^{(1)}, \lambda_G^{(2)}, \cdots, \lambda_G^{(N)})$, assuming $\lambda_G^{(i)}$ is the ith practical implementation of the nominal eigenvalue λ. We simulated the PageRank of Harvard500 in the circuit by considering the eigenvalue variations. Instead of a uniform $\delta = 0.01$ in Figure 5b, the value of δ for each eigenvalue conductance was assumed lying randomly in the range of (0, 0.02). Ten trials were tested. All the results of computing time and solution error show a tight distribution around the ones of the uniform case (Figure S9, Supporting Information), thus confirming the robustness of the circuit against feedback conductance variations in a practical implementation.

3. Conclusion

We have studied the time response of the cross-point RRAM circuit for eigenvector computation, based on the feedback analysis of the self-sustained system. The circuit shows a computing time that relies solely on the mismatch degree of eigenvalue implementation in the circuit, which governs the convergence rate of output voltages toward the steady-state solution. The computing time shows no dependence on the matrix size N, which supports the $O(1)$ complexity of the cross-point eigenvector circuit. The PageRank of the Harvard500 database and its subsets also supports the $O(1)$ time complexity of the circuit. With such a low time complexity, this work supports the significant time/energy efficiency gains of in-memory computing for big data analytics in a wide range of real-world applications.

4. Experimental Section

Experimental Devices: The RRAM devices characterized in this work used a HfO_2 thin film as the switching layer, whose thickness was 5 nm. The HfO_2 dielectric layer was deposited by e-beam evaporation on a confined graphitic carbon bottom electrode (BE); then, a Ti thin layer was deposited on top of the dielectric layer as top electrode (TE) without breaking the vacuum during evaporation. The forming process of RRAM was operated by applying a DC voltage sweep from 0 to 5 V to TE with BE being grounded. The forming process induces a soft breakdown of the dielectric HfO_2 layer, initiating the conductive filament formation and the resistive switching behavior. The set and reset transitions take place under positive and negative voltages applied to the TE, respectively. The DC conduction and switching characteristics of the RRAM were collected by a Keysight B1500A Semiconductor Parameter Analyzer, which was connected to the RRAM device in a conventional probe station for electrical characterization.

SPICE Simulations: Simulations of the cross-point circuit were carried out using LTSPICE (https://www.linear.com/solutions/1066). Linear resistors were used to dictate the conductance values of the programmed RRAM levels, thus mapping a matrix in the cross-point array. AD823 from Analog Devices was used as the OA in the circuit, and parameters can be found here (https://www.analog.com/en/products/ad823.html).

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

This article has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 648635).

Conflict of Interest

The authors declare no conflict of interest.

Keywords

eigenvector, in-memory computing, $O(1)$, resistive memory, time complexity

Received: March 16, 2020
Revised: April 12, 2020
Published online:

[1] D. Ielmini, H.-S. P. Wong, *Nat. Electron.* **2018**, *1*, 333.
[2] M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, D. B. Strukov, *Nature* **2015**, *521*, 61.
[3] T. Gokmen, Y. Vlasov, *Front. Neurosci.* **2016**, *10*, 333.
[4] P. M. Sheridan, F. Cai, C. Du, W. Ma, Z. Zhang, W. D. Lu, *Nat. Nanotechnol.* **2017**, *12*, 784.
[5] C. Li, M. Hu, Y. Li, H. Jiang, N. Ge, E. Montgomery, J. Zhang, W. Song, N. Dávila, C. E. Graves, Z. Li, J. P. Strachan, P. Lin, Z. Wang, M. Barnell, Q. Wu, R. S. Williams, J. J. Yang, Q. Xia, *Nat. Electron.* **2018**, *1*, 52.
[6] M. Le Gallo, A. Sebastian, R. Mathis, M. Manica, H. Giefers, T. Tuma, C. Bekas, A. Curioni, E. Eleftheriou, *Nat. Electron.* **2018**, *1*, 246.
[7] M. A. Zidan, Y. Jeong, J. Lee, B. Chen, S. Huang, M. J. Kushner, W. D. Lu, *Nat. Electron.* **2018**, *1*, 411.
[8] Z. Sun, G. Pedretti, E. Ambrosi, A. Bricalli, W. Wang, D. Ielmini, *Proc. Natl Acad. Sci. USA* **2019**, *116*, 4123.
[9] L. Page, S. Brin, R. Motvani, T. Winograd, *Technical Report of Stanford InfoLab*, Tech. Rep. SIDL-WP-1999-0120, Stanford InfoLab, Stanford, CA **1998**.
[10] M. Turk, A. Pentland, *J. Cogn. Neurosci.* **1991**, *3*, 71.
[11] D. C. Lay, S. R. Lay, J. J. McDonald, *Linear Algebra and Its Applications*, 5th ed., Pearson, Boston, NY **2015**.
[12] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, NJ **1980**.
[13] A. Berman, R. J. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*, Society for Industrial and Applied Mathematics, Philadelphia, PA **1994**.
[14] D. Ielmini, *Semicond. Sci. Technol.* **2016**, *31*, 063002.
[15] D. Tufts, C. Melissinos, *IEEE Trans. Acoust. Speech Signal Process.* **1986**, *34*, 1046.
[16] D. Garber, E. Hazan, C. Jin, S. M. Kakade, C. Musco, P. Netrapalli, A. Sidford, presented at *Proc. of the 33nd Int. Conf. on Machine Learning (ICML)*, New York, NY, USA, June **2016**.
[17] B. Razavi, *Design of Analog CMOS Integrated Circuits*, McGraw-Hill, New York, NY **2001**.
[18] F. Tisseur, K. Meerbergen, *SIAM Rev.* **2001**, *43*, 235.
[19] S. Yu, P.-Y. Chen, Y. Cao, L. Xia, Y. Wang, H. Wu, present at *IEEE Int. Electron Devices Meeting*, Washington, WA, USA, December **2015**.
[20] International Technology Roadmap for Semiconductors (ITRS), http://www.itrs.net/reports.html (accessed: October 2019).
[21] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, R. Zadeh, presented at *Proc. of the 22nd Int. Conf. on World Wide Web*, Rio de Janeiro, Brazil, May **2013**.
[22] K. Bryan, T. Leise, *SIAM Rev.* **2006**, *48*, 569.
[23] C. B. Moler, *Numerical Computing with MATLAB*, Society for Industrial and Applied Mathematics, Philadelphia, PA **2004**.

