Fast solution of linear systems with analog resistive switching memory (RRAM)

Zhong Sun
DEIB, Politecnico di Milano and IU.NET
Milano, Italy
zhong.sun@polimi.it

Giacomo Pedretti
DEIB, Politecnico di Milano and IU.NET
Milano, Italy
giacomo.pedretti@polimi.it

Daniele Ielmini
DEIB, Politecnico di Milano and IU.NET
Milano, Italy
daniele.ielmini@polimi.it

***Abstract*—The in-memory solution of linear systems with analog resistive switching memory in one computational step has been recently reported. In this work, we investigate the time complexity of solving linear systems with the circuit, based on the feedback theory of amplifiers. The result shows that the computing time is explicitly independent on the problem size *N*, rather it is dominated by the minimal eigenvalue of an associated matrix. By addressing the Toeplitz matrix and the Wishart matrix, we show that the computing time increases with log(*N*) or $N^{1/2}$, respectively, thus indicating a significant speed-up of in-memory computing over classical digital computing for solving linear systems. For sparse positive-definite matrix that is targeted by a quantum computing algorithm, the in-memory computing circuit also shows a computing time superiority. These results support in-memory computing as a strong candidate for fast and energy-efficient accelerators of big data analytics and machine learning.**

Keywords—in-memory computing, linear system, resistive memory, time complexity

I. INTRODUCTION

In recent years, in-memory computing with crosspoint resistive memory array has gained an enormous attention, thanks to its potential in providing a high time/energy efficiency for a wide range of data-intensive computations [1]-[6]. By exploiting analog physical rules and massive parallelism in crosspoint architecture, arrays of resistive switching memory (RRAM) devices have been extensively used to accelerate matrix-vector multiplication (MVM) in various applications, such as training and inference of deep neural networks [3], [4], signal and image processing [5], [6], and iterative solution of linear systems [7]. Recently, a crosspoint RRAM circuit has also been shown to solve linear systems in one step, by combining feedback loops in the circuit with operational amplifiers [8]. A single-crosspoint-array circuit solves linear systems of positive matrix, while two crosspoint arrays can address general matrices containing positive, negative and null entries. The ability to solve linear problems in one step is valuable in numerous computing scenarios, such as solving differential equations and calculating eigenvectors, *e.g.*, for PageRank. To benchmark the circuit performance in practical implementations, the computing time complexity of the circuit should be assessed.

II. TIME COMPLEXITY OF SOLVING POSITIVE LINEAR SYSTEMS

A. Crosspoint resistive memory circuit

Solving a linear system means to find out the unknown vector x in the equation

$$Ax = b, \tag{1}$$

where A is an invertible square matrix, b is a known vector. Eq. (1) can be conveniently solved in one step by the circuit shown in Fig. 1a, where the crosspoint RRAM array stores matrix A, and the input voltages V_{in} represent the vector $-b$. The crosspoint rows and columns are connected by operational amplifiers (OAs), which provide a virtual ground at the rows through negative feedback. As a result, the output voltages V_{out} have to satisfy the equation by Kirchhoff's current law

$$AV_{out} + V_{in} = 0, \tag{2}$$

which implies that the output voltages constitute the solution to Eq. (1) with $V_{out} = -A^{-1}V_{in} = A^{-1}b$.

B. Time complexity analysis

The circuit in Fig. 1a is able to solve a linear system in one step, with the steady-state output voltages of OAs as solution. To study the circuit dynamic, it is illustrated as a feedback control system shown as a block diagram in Fig. 1b. The crosspoint matrix A plays the role of a feedback network, which weights the output vector x to interact with the input vector b, and the net result is transferred by OAs. According to Kirchhoff's voltage law, the system can be described by an equation in terms of Laplace transformation, namely

$$-U[Ax(s) - b(s)]L(s) = x(s), \tag{3}$$

where $L(s)$ is the open loop gain of OA, and U is a diagonal matrix as $U = diag\left(\frac{1}{1+\sum_i A_{1i}}, \frac{1}{1+\sum_i A_{2i}}, \cdots, \frac{1}{1+\sum_i A_{Ni}}\right)$. For a single-pole OA, namely $L(s) = \frac{L_0}{1+\frac{s}{\omega_0}}$ [9], where L_0 is the DC open loop gain, ω_0 is the 3-dB bandwidth, Eq. (3) becomes

$$x(s) = -L_0\omega_0 U[Ax(s) - b(s)]. \tag{4}$$

The Laplace transformation of Eq. (4) results in a differential equation in time domain, which is

$$\frac{dx(t)}{dt} = -L_0\omega_0 U[Ax(t) - b(t)]. \tag{5}$$

Eq. (5) describes the circuit dynamic towards the steady-state solution. To analyze the computing time, Eq. (5) is converted into a finite difference (FD) algorithm, which reads

$$x(t + \Delta t) = \alpha Ub + (I - \alpha M)x(t), \tag{6}$$

where Δt is the incremental time, α is a small number equal to $L_0\omega_0\Delta t$, M is a matrix defined as $M = UA$, and I is the identity matrix. For the algorithm to be convergent, the spectral radius ρ of matrix $I - \alpha M$ has to be less than 1, which directly implies the minimal eigenvalue (or real part of eigenvalue) of the associated matrix M ($\lambda_{M,min}$) must be positive. Such a requirement is perfectly met by a positive definite (PD) matrix A, thanks to the positive diagonal matrix U.

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).

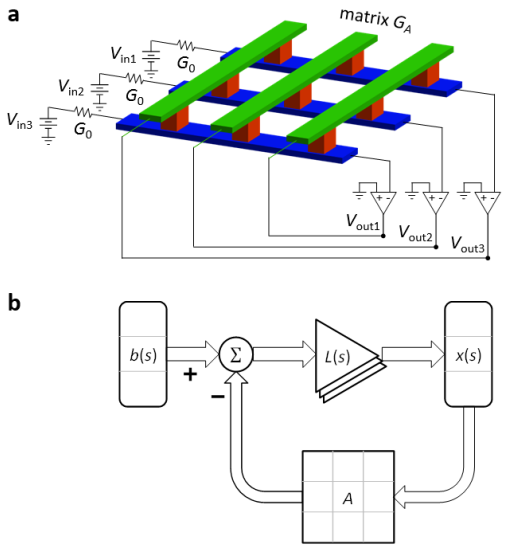

Fig. 1. (a) Crosspoint RRAM circuit for solving linear systems of positive matrix. Matrix A is stored in the crosspoint array with $A = G_A/G_0$. (b) Block diagram of the circuit as a feedback control system.

The FD algorithm is verified by comparing with transient analysis of the circuit in SPICE (Fig. 2). Specifically, 8 discrete conductance levels (120 μS, 80 μS, 60 μS, 50 μS, 30 μS, 20 μS, 15 μS, 10 μS) of RRAM devices from [8] were employed to construct a matrix randomly, under the $\lambda_{M,min}$ constraint. Solving a 3×3 linear system is shown in Fig. 2a, including the matrix normalized by the maximum conductance, an input vector b and the resulting evolvement of solution x. The FD algorithm and circuit simulation results of x are plotted in parallel, showing a high consistency. A 10×10 matrix has also been tested, with the conductance matrix, the input vector and the output dynamic shown in Fig. 2b. Again, the FD algorithm precisely traces the circuit simulation results, thus validating the capability of the algorithm for representing the circuit dynamic and hence for investigating the time complexity.

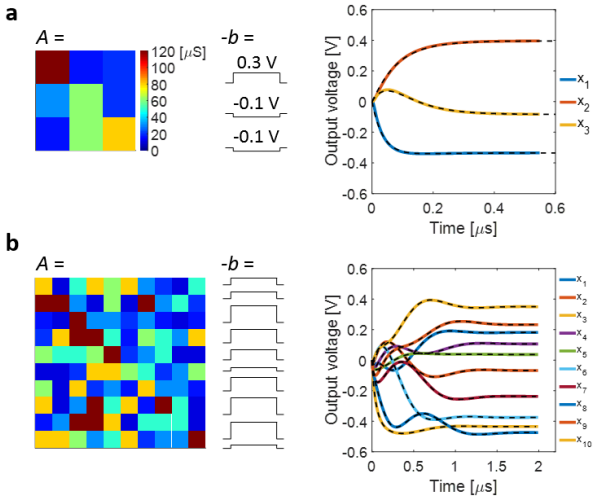

Fig. 2. Simulated solution dynamics of linear systems. (a) A 3×3 linear system case, including the implemented matrix A, the input vector b and the resulting dynamic of solution x. Color lines: transient output voltages in SPICE simulation. Black dash lines: FD algorithm. (b) Same as (a), but for a 10×10 linear system.

The time complexity can be deduced from the convergence analysis of the FD algorithm. The convergence is measured in the A-norm, which is defined as $\|x\|_A = \sqrt{x^T A x}$ [10]. If a linear system is solved with an accuracy ϵ at time t, the A-norm of solution error has to be satisfy

$$\|x(t) - x^*\|_A \le \epsilon, \tag{7}$$

where $x^* = A^{-1}b$ is the precise solution. $\|x(t) - x^*\|_A$ follows

$$\begin{aligned} \|x(t) - x^*\|_A^2 &= \|\alpha U b + (I - \alpha M)x(t - \Delta t) - x^*\|_A^2 \\ &= \|(I - \alpha M)x(t - \Delta t) - (I - \alpha M)x^*\|_A^2 \\ &\le \|(I - \alpha M)\|^2 \|x(t - \Delta t) - x^*\|_A^2 \\ &= \left(1 - \alpha\lambda_{M,min}\right)^2 \|x(t - \Delta t) - x^*\|_A^2 \end{aligned} \tag{8}$$

Repeating the above process iteratively leads to

$$\begin{aligned} \|x(t) - x^*\|_A^2 &\le \left(1 - \alpha\lambda_{M,min}\right)^{2t/\Delta t} \|x(0) - x^*\|_A^2 \\ &< e^{-2\lambda_{M,min} L_0 w_0 t} \|x^*\|_A^2 \\ &= e^{-2\lambda_{M,min} L_0 w_0 t} x^{*T} b \end{aligned} \tag{9}$$

The upper bound of $\|x(t) - x^*\|_A$ satisfying inequation (7) finally reveals the computing time as

$$t \ge \frac{1}{\lambda_{M,min} L_0 w_0} ln \frac{\sqrt{x^{*T} b}}{\epsilon}, \tag{10}$$

which tells the time complexity of solving linear systems with the crosspoint circuit is explicitly independent on matrix size N, instead it is dominated by $\lambda_{M,min}$ of the associated matrix M.

Since U is a diagonal matrix that reflects row sums of matrix A, $\lambda_{M,min}$ is approximately proportional to λ_{min} for matrices with the same size. Therefore, the computing time will scale as $t \propto \frac{1}{\lambda_{min}}$ for a specific N. A series of 3×3 PD matrices with different λ_{min} were generated using the 8 conductance levels. For each matrix, input vectors b were randomly generated and hence the corresponding linear systems were solved by the circuit, with the computing times recorded. The summarized results in Fig. 3a show that the time of solving a linear system increases inversely with the λ_{min} of the matrix. In Fig. 3b, the computing time shows a more precise linear dependence on

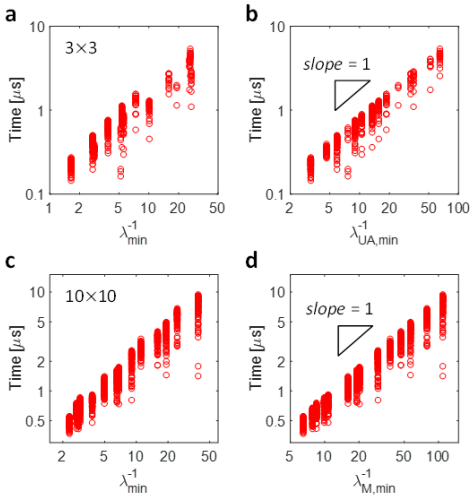

Fig. 3. (a) Computing time for solving linear systems of 3×3 PD matrices with various λ_{min}. (b) Computing time plotted against $\lambda_{M,min}$ of the associated matrix M. (c) Same as (a), but for a series of 10×10 PD matrices. (d) Same as (b), but for the 10×10 matrices.

$1/\lambda_{M,min}$, thus verifying the time complexity in Eq. (10). A series of 10×10 PD matrices have also been tested. Again, while the time of solving a linear system increases roughly with $1/\lambda_{min}$ (Fig. 3c), it shows a more precise linear dependence on $1/\lambda_{M,min}$ (Fig. 3d).

To study the dependence of computing time on the matrix size N, we solved linear systems of a Toeplitz matrix [11], which is defined according to $A_{ij} = \frac{1}{|i-j|+1}, i,j = 1,2,\dots,N$. As N increases, the minimal eigenvalue λ_{min} of the Toeplitz matrix is asymptotically constant, while $\lambda_{M,min}$ decreases linearly with log(N) (Fig. 4a). As a result, the computing time of solving linear systems increases linearly with log(N), namely the time complexity is O(log$N$) (Fig. 4b). In conventional computers, the Toeplitz systems can be solved with the Levinson algorithm, whose time complexity is $O(N^2)$ [11]. Therefore, our approach provides an exponential speed-up for solving such linear systems compared to conventional digital computing.

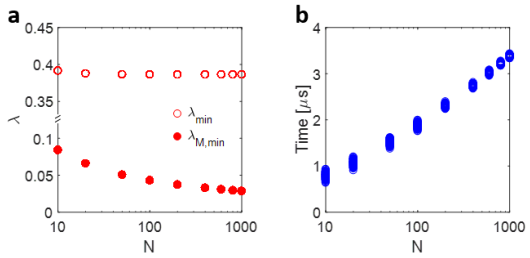

Fig. 4. (a) λ_{min} of the Toeplitz matrix with various sizes, and $\lambda_{M,min}$ of the associated matrices. (b) Computing time of solving linear systems of the Toeplitz matrices. 100 random input vectors b were generated for each size, forming linear systems to be solved. The resulting computing time indicates an O(log$N$) time complexity.

The robustness of the circuit against device variation and analog noise is related to the condition number (κ) of the stored matrix [8]. A well-conditioned matrix whose κ is small can tolerate the circuit imperfections, as the matrix property is stable. Only if the $\lambda_{M,min}$ is positive for a deviated matrix, the solution should be convergent. In Fig. 5, a 100×100 Toeplitz system is solved with or without device deviations in the crosspoint array. Both sets of solution, *i.e.* the 100 output voltages converge with a slight difference of computing time, and the steady-state solution shows relatively small errors. Specifically, the computing time for solving the ideal linear system is 2.126 μs, while it is 2.174 μs for solving the deviated one, which is explained by the reduced $\lambda_{M,min}$ for the latter case.

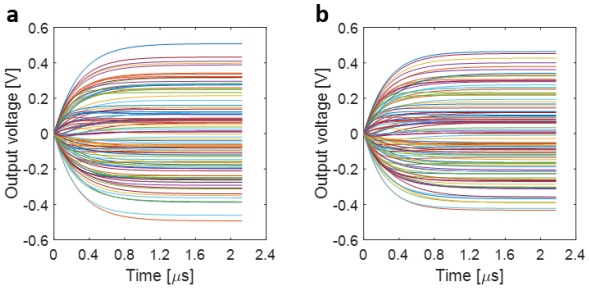

Fig. 5. (a) Transient behavior of solving a 100×100 Toeplitz system without device variations, whose condition number is 19.6. (b) Transient behavior of solving the same linear system, but each crosspoint device is of a variation within ±5%. The $\lambda_{M,min}$ is 0.0429 and 0.0408 for the ideal and the deviated matrix, respectively.

III. TIMECOMPLEXITY OF SOLVING MIXED LINEAR SYSTEMS

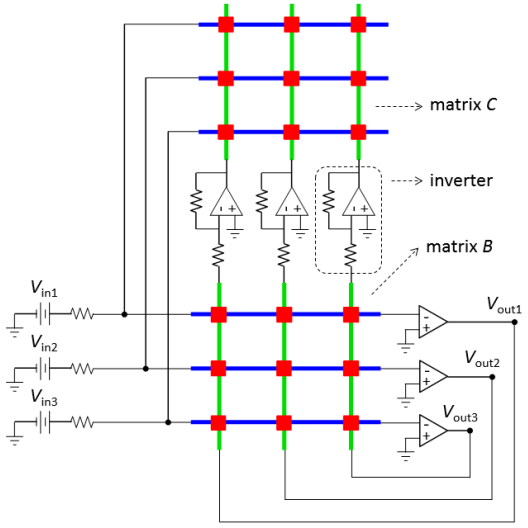

Fig. 6. Crosspoint resistive memory circuit for solving linear systems of general real matrix. Two crosspoint arrays are used to implement the original matrix A, with $A = B - C$.

A. Dual-crosspoint-array circuit

To address mixed matrices containing negative elements, a general circuit was also proposed [8], as shown in Fig. 6. A mixed matrix A can always be split by two positive matrices B and C, namely $A = B - C$. In the circuit, matrices B and C are mapped in two crosspoint RRAM arrays, respectively, and the minus operation is realized by analog inverters between columns of the two arrays. By considering the virtual ground at row lines and applying Kirchhoff's current law, namely $BV_{out} - CV_{out} + V_{in} = 0$, one can easily find out that the output voltages of OAs constitute the solution to a linear system of matrix A.

B. Time complexity analysis

To investigate time complexity of the circuit in Fig. 6, we also analyze the circuit dynamics based on feedback theory of amplifiers. Term the output voltages of OAs and inverters x and y, respectively, and $-V_{in}$ is represented by b, then the following two equations are obtained

$$-U[Bx(s) + Cy(s) - b(s)]L(s) = x(s), \qquad (11\text{-}1)$$

$$-\frac{x(s)+y(s)}{2}L(s) = y(s), \qquad (11\text{-}2)$$

where $U = diag\left(\frac{1}{1+\sum_i B_{1i}+\sum_i C_{1i}}, \frac{1}{1+\sum_i B_{2i}+\sum_i C_{2i}}, \cdots, \frac{1}{1+\sum_i B_{Ni}+\sum_i C_{Ni}}\right)$, and the identical open loop gain $L(s)$ is assumed for all OAs in the circuit. Combining the two equations in Eq. (11) leads to

$$-U\left[Bx(s) - \frac{L(s)}{L(s)+2}Cx(s) - b(s)\right]L(s) = x(s). \qquad (12)$$

By considering a single-pole OA model [9], and then applying Laplace transformation, a second-order differential equation describing the circuit dynamic is obtained,

$$\frac{d^2x(t)}{dt^2} = -\frac{L_0\omega_0}{2}\left[(2UB + I)\frac{dx(t)}{dt} + L_0\omega_0 UAx(t) - L_0\omega_0 Ub\right]. \qquad (13)$$

To analyze time complexity of the circuit, Eq. (13) is converted as a first-order differential equation by assuming

$$z(t) = -\frac{2}{L_0\omega_0}\frac{dx(t)}{dt}. \qquad (14\text{-}1)$$

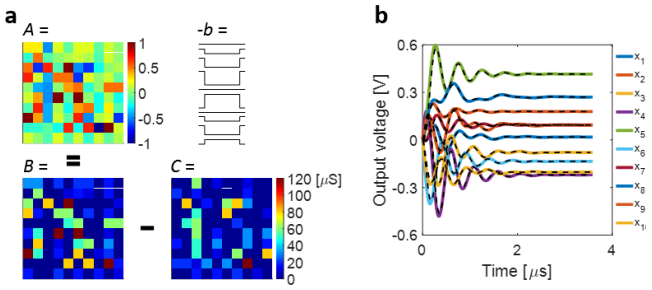

Fig. 7. (a) A 10×10 mixed matrix A and two positive matrices B and C splitting it. Matrices B and C are stored in the two crosspoint arrays in the circuit of Fig. 6, respectively. An input vector b is also provided to form a linear system to be solved by the circuit. (b) Solution dynamic of the constructed 10×10 linear system. Color lines: transient output voltages in SPICE. Black lines: FD algorithm.

As a result, Eq. (13) becomes

$$\frac{dz(t)}{dt} = -\frac{L_0\omega_0}{2}[(2UB + I)z(t) - Ax(t) + Ub]. \qquad (14\text{-}2)$$

The two equations in Eq. (14) can be merged as

$$\frac{d}{dt}\begin{bmatrix}x(t)\\z(t)\end{bmatrix} = -L_0\omega_0\left\{\begin{bmatrix}0 & I/2\\-UA & UB + I/2\end{bmatrix}\begin{bmatrix}x(t)\\z(t)\end{bmatrix} + \begin{bmatrix}0\\Ub\end{bmatrix}\right\}, \qquad (15)$$

Defining $M = \begin{bmatrix}0 & I/2\\-UA & UB + I/2\end{bmatrix}$ as the associated matrix with a size of $2N$×$2N$, $X(t) = \begin{bmatrix}x(t)\\z(t)\end{bmatrix}$ and $J = \begin{bmatrix}0\\Ub\end{bmatrix}$, whose sizes are $2N$×1, Eq. (15) finally becomes

$$X(t + \Delta t) = -\alpha J + (I - \alpha M)X(t), \qquad (16)$$

where I is the $2N$×$2N$ identity matrix, α is equal to $L_0\omega_0\Delta t$.

Eq. (16) is same as Eq. (6), hence the discussions for solving positive linear systems can be applied to solving mixed ones. Consequently, the $\lambda_{M,min}$ of matrix M must also be positive. The FD algorithm in Eq. (16) is verified by comparing with SPICE transient simulation for solving a 10×10 mixed linear system. In Fig. 7a, a mixed matrix is randomly generated under the $\lambda_{M,min}$ constraint, then it is split by two positive matrices, which are implemented by the 8 conductance levels. An input vector is also provided to form a linear system to be solved. In Fig. 7b, both SPICE simulation and iterative algorithm results of the solution are plotted in parallel, indicating a precise description of the circuit dynamic by Eq. (16).

C. Solving linear systems of Wishart matrix

The time complexity of solving linear systems of mixed matrix should also be solely determined by $\lambda_{M,min}$. As a case study, solving linear systems of a sample covariance matrix named Wishart matrix is addressed. Wishart matrix is a random matrix defined as $W = \frac{RR^T}{N}$, where R is a S×N matrix whose entries are IID random variables with zero mean ($\mu = 1$) and one variance ($\sigma = 1$) [12]. Apparently W is a mixed matrix containing both positive and negative entries, as shown in Fig. 8a for a 100×100 Wishart matrix. When solving a linear system of Wishart matrix, W is split by two positive matrices B and C, to be mapped by the two crosspoint arrays in the circuit, respectively. Matrix B is element-wise composed of the positive entries in W, and the other entries are assigned a small number, *e.g.* 10^{-4}, which can be represented by the high resistance state of RRAM device. Matrix C is then calculated according to $W = B - C$.

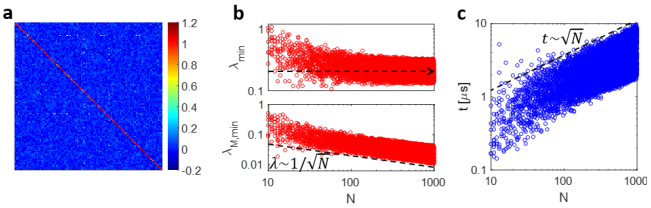

Fig. 8. Computing time of solving linear systems of Wishart matrices with various sizes. (a) A representative 100×100 Wishart matrix. (b) Top panel: λ_{min} of 10,000 randomly-generated Wishart matrices, with sizes from 10×10 to 1,000×1,000. The black dash line indicates that λ_{min} converges to 0.2. Bottom panel: $\lambda_{M,min}$ of the associated matrices, plotted as a function of N. The black dash line evidences that $\lambda_{M,min}$ decreases linearly with $1/\sqrt{N}$. (c) Computing time of solving linear systems of the 10,000 Wishart matrices. The black dash line reveals the computing time increases linearly with $\sqrt{N}$.

It is stated that the λ_{min} of W converges almost surely to $\left(1 - \sqrt{y}\right)^2$ as $N \to \infty$ with $s/N \to y \in (0,1)$ [12]. Such a property is illustrated in the top panel of Fig. 8b, where $y = 0.3$ is assumed and hence λ_{min} is supposed to converge to 0.2. $\lambda_{M,min}$ is also calculated in parallel for each matrix, showing a scaling of $\lambda_{M,min}$~$1/\sqrt{N}$ (bottom panel of Fig. 8b). Together with the generation of a Wishart matrix, a random input vector is also created for solving a linear system by the circuit. The computing time was recorded for each trial, and all the results are summarized in Fig. 8c. As can be observed, computing time increases with $\sqrt{N}$, thus demonstrating the validity of Eq. (10) as time complexity of solving linear system of mixed matrix by the dual-crosspoint-array circuit. Note that the time complexity of classical algorithms, such as Gaussian elimination for solving linear systems in conventional digital computers is $O(N^3)$. Therefore, the $O(N^{1/2})$ time complexity of the crosspoint circuit represents a significant speed-up for solving linear systems in memory.

IV. Comparison with Digital and Quantum Computing

For the matrix product $M = UA$, there is an inequality describing the minimal eigenvalues, namely $\lambda_{M,min} \geq U_{min}\lambda_{min}$, where U_{min} is the minimal eigenvalue, *i.e.*, the minimal element of matrix U. As U_{min} is determined by the maximum row sum of U, the lower bound of $\lambda_{M,min}$ is expected to decrease with N for dense matrices, and thus imposing an N-dependence to time complexity. However, this is not the situation for sparse matrix. Here we concern s-sparse positive-definite matrix, which contains at most s non-zero elements per row. In this case, $\lambda_{M,min} \geq U_{min}\lambda_{min}$~$\frac{\lambda_{min}}{s}$ does not rely on matrix size N, and hence the time complexity of solving linear systems will be $O\left(\frac{s}{\lambda_{min}} ln\frac{1}{\epsilon}\right)$. In conventional computers, the most efficient algorithm for solving linear systems of positive-definite matrix is the conjugate gradient (CG) method that owns a linear time complexity for sparse matrix [13]. A quantum computing algorithm has also been proposed for solving linear systems with a logarithmic time complexity [14], as summarized in details in Table I. Therefore, compared to both digital and quantum computing, the crosspoint RRAM circuit shows a computing time superiority to extremely speed up solving linear systems of sparse positive-definite matrix, which is encountered in a wide range of practical applications [15].

TABLE I. TIME COMPLEXITY COMPARISON

CG method	Quantum computing	In-memory computing
$O\left(Ns\sqrt{\frac{\lambda_{max}}{\lambda_{min}}}ln\frac{1}{\epsilon}\right)$	$O\left(\frac{s^2{\lambda_{max}}^2}{\epsilon{\lambda_{min}}^2}lnN\right)$	$O\left(\frac{s}{\lambda_{min}}ln\frac{1}{\epsilon}\right)$

CONCLUSION

In this work, we study the circuit dynamics based on the feedback theory of amplifier circuits. The results show that, for both single- and dual-crosspoint circuits, the time complexity is solely dictated by the minimal eigenvalue of an associated matrix. For a representative Toeplitz matrix, the time complexity of solving linear systems is shown to be $O(\log N)$, while for the Wishart matrix dataset, the time complexity is $O(N^{1/2})$. These results demonstrate a significant speed-up over the polynomial time complexity of classical digital computers. For sparse linear systems, the time complexity is reasoned as N-independent, evidencing a higher time efficiency than the quantum computing algorithm, thus strongly supporting in-memory computing as a promising approach for data analysis and machine learning.

REFERENCES

[1] D. Ielmini and H. -S. P. Wong, "In-memory computing with resistive switching devices," *Nature Electronics*, vol. 1, no. pp. 333-343, 2018.

[2] Z. Sun, E. Ambrosi, A. Bricalli, and D. Ielmini, "Logic computing with stateful neural networks of resistive switches," *Advanced Materials*, vol. 30, no. 38, p. 1802554, 2018.

[3] M. Prezioso et al., "Training and operation of an integrated neuromorphic network based on metal-oxide memristors," *Nature*, vol. 521, pp. 61-64, 2015.

[4] P. Yao et al., "Face classification using electronic synapses," *Nature Communications*, vol. 8, p. 15199, 2017.

[5] P. M. Sheridan, F. Cai, C. Du, W. Ma, Z. Zhang, and W. D. Lu, "Sparse coding with memristor networks," *Nature Nanotechnology*, vol. 12, pp. 784-789, 2017.

[6] C. Li et al., "Analogue signal and image processing with large memristor crossbars," *Nature Electronics*, vol. 1, pp. 52-59, 2018.

[7] M. Le Gallo et al., "Mixed-precision in-memory computing," *Nature Electronics*, vol. 1, pp. 246-253, 2018.

[8] Z. Sun et al., "Solving matrix equations in one step with cross-point resistive arrays," *PNAS*, vol. 116, no. 10, pp. 4123-4128, 2019.

[9] B. Razavi, Design of Analog CMOS Integrated Circuits. McGraw-Hill, New York, 2001.

[10] G. H. Golub & C. F. van Loan, Matrix Computations, 4th ed., Johns Hopkins Univ. Press, Baltimore, 2013.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, Numerical Recipes: The Art of Scientific Computing, 3rd ed. Cambridge University Press, New York, NY, 2007.

[12] J. W. Silverstein, "The Smallest Eigenvalue of a Large Dimensional Wishart Matrix," *The Annals of Probability*, vol. 13, pp. 1364-1368, 1985.

[13] J. R. Shewchuk, "An Introduction to the Conjugate Gradient Method without the Agonizing Pain," *Report CMU-CS-94-125, School of Computer Science*, Carnegie Mellon University, Pittsburgh, 1994.

[14] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum Algorithm for Linear Systems of Equations," *Physical Review Letters*, vol. 103, p. 150502, 2009.

[15] R. Bhatia, Positive definite matrices. Princeton Univ. Press, Princeton, NJ, 2007.