Solving matrix equations in one step with cross-point resistive arrays

Zhong Sun[a], Giacomo Pedretti[a], Elia Ambrosi[a], Alessandro Bricalli[a], Wei Wang[a], and Daniele Ielmini[a,1]

[a]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 20133 Milan, Italy

Edited by Eli Yablonovitch, University of California, Berkeley, CA, and approved January 17, 2019 (received for review September 11, 2018)

Conventional digital computers can execute advanced operations by a sequence of elementary Boolean functions of 2 or more bits. As a result, complicated tasks such as solving a linear system or solving a differential equation require a large number of computing steps and an extensive use of memory units to store individual bits. To accelerate the execution of such advanced tasks, in-memory computing with resistive memories provides a promising avenue, thanks to analog data storage and physical computation in the memory. Here, we show that a cross-point array of resistive memory devices can directly solve a system of linear equations, or find the matrix eigenvectors. These operations are completed in just one single step, thanks to the physical computing with Ohm's and Kirchhoff's laws, and thanks to the negative feedback connection in the cross-point circuit. Algebraic problems are demonstrated in hardware and applied to classical computing tasks, such as ranking webpages and solving the Schrödinger equation in one step.

in-memory computing | cross-point architecture | analog computing | linear algebra | resistive memory

Linear algebra problems, such as solving systems of linear equations and computing matrix eigenvectors, lie at the heart of modern scientific computing and data-intensive tasks. Traditionally, these problems in forms of matrix equations are solved by matrix factorizations or iterative matrix multiplications (1, 2), which are computationally expensive with polynomial time complexity, e.g., $O(N^3)$ where N is the size of the problem. As conventional computers are increasingly challenged by the scaling limits of the complementary metal-oxide-semiconductor (CMOS) technology (3), and by the energy and latency burdens of moving data between the memory and the computing units (4), improving the computing performance with increasing hardware resources becomes difficult and noneconomic. To get around these fundamental limits, in-memory computing has recently emerged as a promising technique to conduct computing in situ, i.e., within the memory unit (5). One example is computing within cross-point arrays, which can accelerate matrix-vector multiplication (MVM) by Ohm's law and Kirchhoff's law with analog and reconfigurable resistive memories (5–8). In-memory MVM has been adopted for several tasks, including image compression (5), sparse coding (6), and the training of deep neural networks (7, 8). However, solving matrix equations, such as a linear system $Ax = b$, in a single operation remains an open challenge. Here, we show that a feedback circuit including a reconfigurable cross-point resistive array can provide the solution to algebraic problems such as systems of linear equations, matrix eigenvectors, and differential equations in just one step.

Resistive memories are two-terminal elements that can change their conductance in response to applied voltage stimuli (9, 10). Owing to their nonvolatile and reconfigurable behavior, resistive memories have been widely investigated and developed for storage-class memory (11, 12), stateful logic (13–15), in-memory computing (5, 6, 16, 17), and neuromorphic computing applications (7, 8, 18, 19). Resistive memories include various device concepts, such as resistive switching memory (RRAM, refs. 9–12), phase-change memory (PCM, ref. 20), and spin-transfer torque magnetic memory (21). Implemented in the cross-point array architecture, resistive memories can naturally accelerate data-intensive operations with enhanced time/energy efficiencies compared with classical digital computing (5, 6, 17). It has also been shown recently that iterated MVM operations with resistive cross-point arrays can solve systems of linear equations, in combination with digital floating-point computers (22). The higher the desired accuracy of the solution, the more iterations are needed to complete the operation. However, iteration raises a fundamental limit toward achieving high computing performance in terms of energy and latency.

Results

Cross-Point Circuits for Solving a System of Linear Equations. Fig. 1*A* shows the proposed feedback circuit for solving a system of linear equations in one step, and the hardware circuit on a printed circuit board is shown in *SI Appendix*, Fig. S1. The circuit is a cross-point array of RRAM devices, each consisting of a metal-insulator–metal stack with a HfO_2 layer between a Ti top electrode and a C bottom electrode (15). The devices show a set transition from high resistance to low resistance when a positive voltage above the threshold V_{set} is applied to the Ti electrode, and a reset transition from low resistance to high resistance when a negative voltage above the threshold V_{reset} is applied to the Ti electrode. Multilevel operation is also possible by executing the set transition at variable maximum (compliance) current I_C, or executing the reset transition at variable maximum voltage V_{stop} (23), as shown in *SI Appendix*, Fig. S2. The 3×3 cross-point array

Significance

Linear algebra is involved in virtually all scientific and engineering disciplines, e.g., physics, statistics, machine learning, and signal processing. Solving matrix equations such as a linear system or an eigenvector equation is accomplished by matrix factorizations or iterative matrix multiplications in conventional computers, which is computationally expensive. In-memory computing with analog resistive memories has shown high efficiencies of time and energy, through realizing matrix-vector multiplication in one step with Ohm's law and Kirchhoff's law. However, solving matrix equations in a single operation remains an open challenge. Here, we show that a feedback circuit with cross-point resistive memories can solve algebraic problems such as systems of linear equations, matrix eigenvectors, and differential equations in just one step.

Author contributions: Z.S., G.P., and D.I. designed research; Z.S., G.P., E.A., A.B., W.W., and D.I. performed research; Z.S., G.P., and D.I. analyzed data; and Z.S. and D.I. wrote the paper.

The authors declare no conflict of interest.

This article is a PNAS Direct Submission.

[1]To whom correspondence should be addressed. Email: daniele.ielmini@polimi.it.

This article contains supporting information online at www.pnas.org/lookup/suppl/doi:10.1073/pnas.1815682116/-/DCSupplemental.

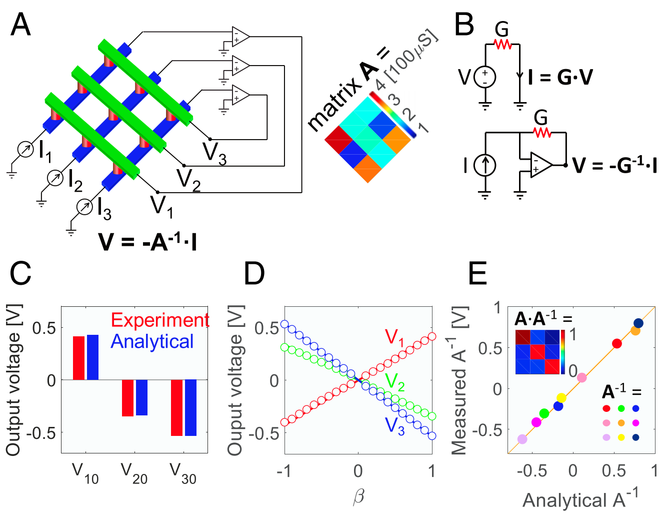

Fig. 1. Solving systems of linear equations with a cross-point array of resistive devices. (*A*) Cross-point circuit for solving a linear system or inverting a positive matrix. RRAM elements (red cylinders) are located at the cross-point positions between rows (blue bars) and columns (green bars). (*Inset*, *Right*) Experimental conductance values mapping the elements of matrix *A*. The transformation units between the real-valued matrices/vectors and the physical implementations were G_0 = 100 μS, V_0 = 1 V, and I_0 = 100 μA for RRAM conductance, input/output voltage, and output/input current, respectively. The other cases also follow this convention if not specified. (*B*) Circuits to calculate a scalar product $I = G{\cdot}V$ by Ohm's law, and to calculate a scalar division $V = -I/G$ by a TIA. (*C*) Measured solution to a linear system with an input current vector $I = [0.2;\ 1;\ 1]I_0$. The experimental output voltages give a solution very close to the analytical one. (*D*) Measured solution to the linear systems, namely output voltages, as functions of parameter β controlling the input current given by $I = \beta{\cdot}[0.2;\ 1;\ 1]I_0$ with $-1 \le \beta \le 1$. The experimental solutions (color circles) are compared with analytical solutions (color lines) of the system, supporting the accuracy of the physical calculation. (*E*) Experimental matrix inverse A^{-1}, namely measured output voltages in three subsequent experiments with input current $I = [1;\ 0;\ 0]I_0$, $[0;\ 1;\ 0]I_0$, and $[0;\ 0;\ 1]I_0$, respectively. The analytical solution is also shown. (*Inset*) Matrix product AA^{-1} is very close to the unit matrix *U*, thus supporting the experimental inversion.

in the figure can perform MVM with open loop, i.e., by applying a voltage vector *V* to the columns and measuring the current vector *I* at the rows without the row-column connections enabled by the operational amplifiers (OAs), which is shown in *SI Appendix*, Fig. S3. The measured currents yields the dot product $I = A{\cdot}V$ between the applied analog voltages and the matrix *A* of the RRAM conductance values in the cross-point array. The results evidence a small error generally below 8%, mostly arising from the nonlinearity of conductance in cross-point resistive devices. This is in line with previous results, where the MVM accuracy appeared satisfactory (5), although not aligned with single- and double-precision full digital operation.

The MVM operation is a consequence of physical Ohm's law $I = G{\cdot}V$, where *G* is the device conductance, *V* is the applied voltage, and *I* is the measured current (Fig. 1*B*, *Top*). On the other hand, the inverse operation $V = -I/G$ can be obtained for a given *I* and *G*, simply by forcing the current *I* at a grounded node of the resistive device and measuring the potential *V* at the second node. This physical division is accomplished by the transimpedance amplifier (TIA) in Fig. 1*B* (*Bottom*), where the current is injected at the inverting-input node of an OA, and the feedback conductance *G* connects input and output nodes of the OA. The differential input voltage $V^+ - V^-$ at the OA is minimized by the high gain of the OA, thus establishing a virtual ground ($V^- = 0$) at the inverting-input node (24, 25) and enabling the physical division. This provides the basis for the circuit in Fig. 1*A*, which solves a system of linear equations expressed by the matrix formula:

$$Ax = b, \qquad [1]$$

where *A* is a nonsingular square matrix mapped with conductance values of cross-point RRAM devices, *b* is a known vector, and *x* is the unknown vector. In this circuit, the input currents $I = -b$ are applied to cross-point rows connected to the virtual-ground nodes of the OAs. As a result, currents are forced to automatically distribute among the resistive elements in the cross-point array, to establish an output potential *V* satisfying

$$A{\cdot}V + I = 0, \qquad [2]$$

which implies $V = -A^{-1}{\cdot}I = x$. A circuit similar to the one in Fig. 1*A* was previously presented in the report of the International Roadmap for Devices and Systems (25) and suggested by ref. 26, although no demonstration was shown regarding the ability to solve a linear system by either experiments or simulations.

To demonstrate the concept of Fig. 1*A*, we measured the output voltages in the 3 × 3 RRAM cross-point array of Fig. 1*A*, where the conductance matrix is also shown. All of the matrices adopted in the experiments of this work are reported in *SI Appendix*, Table S1. A current vector $[I_{10};\ I_{20};\ I_{30}]$ with I_{10} = 20 μA, I_{20} = 100 μA, and I_{30} = 100 μA, was applied to the array rows, and the resulting potential at the array columns, i.e., $[V_{10};\ V_{20};\ V_{30}]$, was measured, as shown in Fig. 1*C*. The good agreement (with relative errors within 3%) with the analytical solution supports the functionality of the feedback circuit of Fig. 1*A* for solving the matrix equation in Eq. **1**. The circuit was further demonstrated by linearly changing the input currents according to $I_i = \beta\, I_{i0}$, where *i* = 1, 2, or 3, and β was changed uniformly in the range from −1 to 1. Results are reported in Fig. 1*D*, showing the measured output voltages compared with the analytical solutions $x = A^{-1}b$. The error remains below 10% for $|\beta| > 0.5$ (*SI Appendix*, Fig. S4). Notably, Eq. **1** is physically solved in just one step thanks to the physical MVM in the cross-point array and to the feedback connection forcing the virtual ground at cross-point rows.

The same concept can be extended to compute the inversion of a matrix *A* satisfying $AA^{-1} = U$, where *U* is the unit matrix. The *i*th column of A^{-1} can be measured as the output voltage when the *i*th column of *U* is applied as an input, thus realizing matrix inversion in *N* steps. Fig. 1*E* shows the measured elements of A^{-1} compared with the analytically solved inverse-matrix elements, and the relative errors are calculated in *SI Appendix*, Fig. S5. Fig. 1*E* (*Inset*) shows that the experimental product AA^{-1} well approximates *U*, which further supports the computed matrix inversion.

The circuit of Fig. 1*A* is essentially a matrix-inversion operator, which can be utilized to solve linear systems and matrix inversions, while a cross-point array without feedback is a matrix operator, which can be naturally used to perform MVM. Since the matrix-inversion circuit is a negative feedback system, the stability of the output voltage requires that the loop gain (G_{loop}) of every feedback loop is negative (27). The analysis reveals that the condition $G_{loop} < 0$ is satisfied when the signs of the diagonal elements of A^{-1} are all positive (*SI Appendix*, Fig. S6). Following this guideline, a system of linear equations and the inversion of a 5 × 5 matrix have been solved, with the matrix implemented in a cross-point array of discrete resistors. The small relative error around few percent in this ideal case with discrete resistors evidences that a high accuracy might be achieved with accurate and linear resistive memory devices (*SI Appendix*, Fig. S7).

Solving a Linear System with Positive and Negative Coefficients. Since conductance can only be positive in a resistive element, the scheme of Fig. 1 can only solve linear systems with a positive matrix of coefficients. To solve linear systems with nonpositive coefficients, the mixed-matrix circuit in Fig. 2 shall be adopted. Here, the matrix A is split into two cross-point arrays according to $A = B - C$, where B and C are both positive. Fig. 2*A* shows the two cross-point-array implementation, where the input current I is split by the circuit into two components I_B and $I_C = I - I_B$, being submitted to virtual-ground rows of B and C, respectively. Analog inverters enable voltage inversion between the columns of B and C. Based on Ohm's law and Kirchhoff's current law, the output voltage V of the OAs is given by

$$B \cdot V + C(-V) + I = 0, \qquad [3]$$

or $A \cdot V + I = 0$, which solves the linear system of Eq. **1** with $I = -b$.

We experimentally demonstrated the inversion of a 3×3 mixed matrix A with the two matrices B and C implemented in an RRAM array and a resistor array, respectively. The values of A, B, and C are shown in Fig. 2*B*, while Fig. 2*C* shows the measured elements of A^{-1} as a function of the analytical results, demonstrating good accuracy. To further support physical matrix inversion, we inverted A^{-1}, which is a positive matrix, with a single cross-point array. For that purpose, the elements of A^{-1} were first mapped as conductance values in an RRAM array by employing a program-and-verify algorithm with error below 5% (*SI Appendix*, Fig. S8). Although the program-and-verify algorithm was applied to an individual RRAM device at a time, a cross-point array is suitable for parallel programming to significantly reduce the array initialization time (28, 29). Fig. 2*D* shows the measured RRAM conductance values as a function of the target values obtained from the experimental A^{-1} in Fig. 2*C*. The inversion of A^{-1}, i.e., $(A^{-1})^{-1}$, was computed by the matrix-inversion circuit of Fig. 1*A*, yielding the results in Fig. 2*E*. The computed $(A^{-1})^{-1}$ is compared with the original matrix A in Fig. 2*F*, which supports the good accuracy of the double inversions $(A^{-1})^{-1} = A$. The relative errors of the above operations are reported in *SI Appendix*, Fig. S9.

Similar to the single cross-point-array circuit of Fig. 1*A*, the condition for negative feedback applies to the mixed matrix A. Besides, since cross-point array B is directly involved in the closed-loop feedback with the OAs, matrix B also has to satisfy the condition $G_{loop} < 0$. As a proposal for practical applications, a reference matrix B satisfying the G_{loop} condition can be adopted in the mixed-matrix circuit, while matrix C can be freely arranged with an RRAM cross-point array with the condition $C = B - A$. To demonstrate the generality of this concept, the one-dimensional steady-state Fourier equation for heat diffusion was solved with a cross-point-array circuit (*SI Appendix*, Figs. S10 and S11). By using the finite difference method, the differential equation is first converted into a system of linear equations, where the characteristic matrix A is a mixed tridiagonal matrix. Input currents correspond to the known term, namely the dissipated power in the one-dimensional structure. The solution yields the temperature profile along the reference structure, which solves the numerical Fourier equation.

A key parameter to describe the stability of the solution of a linear system is the condition number κ of matrix (30). The condition number reflects the stability of the solution x upon small variations of the known term b in Eq. **1**, where the sensitivity to perturbations increases with the condition number. To study the impact of the condition number on the solution of linear systems in resistive memory arrays, we simulated the circuit-based inversion of three 10×10 matrices with increasing condition number. To test the stability of the solution, a random variation of 0.1 or −0.1 was added to each element in the term b of the equation $Ax = b$, where b is the ith column of the unit matrix U, x is the ith column of A^{-1}, and i was swept from 1 to 10 to calculate the whole inverse matrix. The results are reported in *SI Appendix*, Fig. S12, indicating that the computing error increases with the condition number of the matrix.

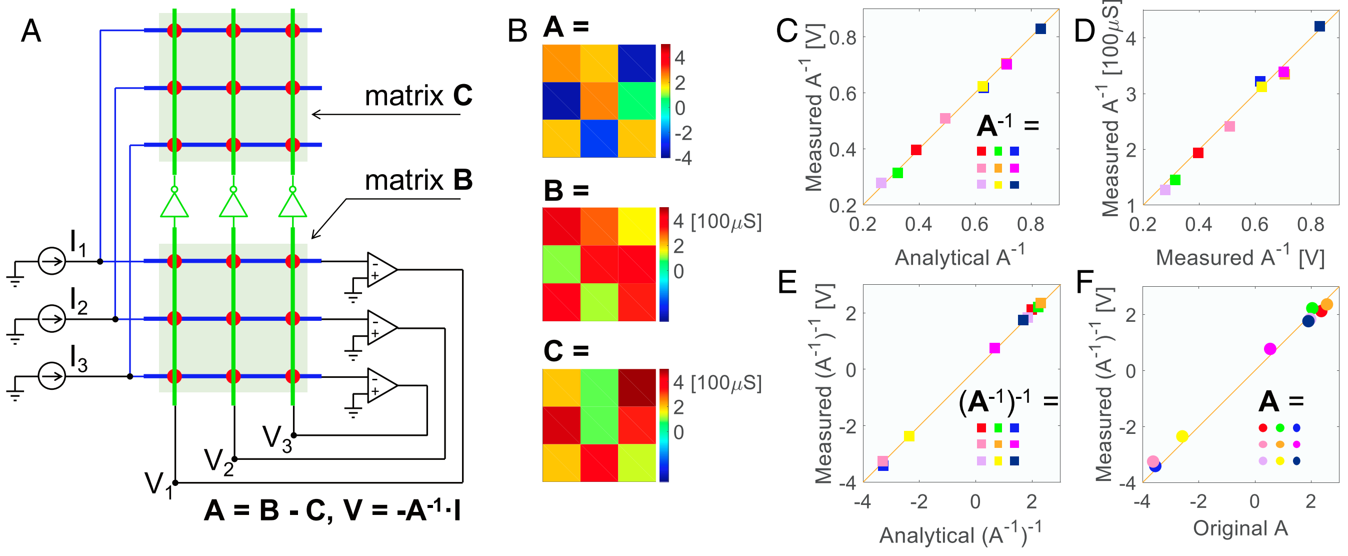

Fig. 2. Inversion of a mixed matrix. (*A*) Schematic of two cross-point-array circuit for matrix inversion, where two cross-point arrays contain the elements of matrices B (*Bottom*) and C (*Top*) with $A = B - C$. The voltage in matrix C is inverted in the other by analog inverters, while the input current is injected in virtual-ground lines and split in the two matrices. (*B*) Measured values of the matrices A, B, and C, with $A = B - C$. In the experiment, matrix B was implemented by a cross-point array of RRAM, while matrix C was implemented by a cross-point array of discrete resistors. (*C*) Measured values of the inverse matrix A^{-1} as a function of the analytically calculated elements of A^{-1}. As A^{-1} is a positive matrix, it can be inverted by a single cross-point array as in Fig. 1. (*D*) Conductance values for matrix A^{-1} implemented in RRAM elements, as a function of the experimental values of A^{-1} in *C*. To make the devices work in the high-conductance region, the matrix A^{-1} was implemented with $G_0 = 500$ μS for RRAM conductance. (*E*) Measured elements of matrix $(A^{-1})^{-1}$ as a function of analytical calculations. $I_0 = 500$ μA and $V_0 = 1$ V were used for input current and output voltage, respectively. (*F*) Measured elements of matrix $(A^{-1})^{-1}$ as a function with the original matrix A, showing a remarkable accuracy despite the accumulated errors over the two sequential inversion processes and the device-programming process.

The impact of the condition number was also tested in experiments, by performing a double inversion of a matrix with a larger condition number ($\kappa = 16.9$), compared with the one with $\kappa = 9.5$ in Fig. 2. Condition numbers for all of the matrices in experiment are summarized in *SI Appendix*, Table S1. As shown in *SI Appendix*, Fig. S13, the matrix with a larger κ is successfully inverted twice, although the computing errors are larger than the case in Fig. 2 (*SI Appendix*, Fig. S14). It should be noted that the matrices addressed in this work are well conditioned. For an ill-conditioned matrix with extremely high condition number, additional schemes should be required, possibly including iterative refinement algorithms that can be either supported by a conventional digital computer (22) or implemented in a resistive memory array (26). The error caused by the thermal noise and shot noise of the components in the cross-point circuit also increases with the condition number, although representing a much less significant concern (*SI Appendix*, Fig. S15).

Cross-Point Circuits for Computing Eigenvectors. The solution of a linear system in Eq. **1** can be further extended to the calculation of eigenvectors via physical computing in a cross-point array. The eigenvector equation reads

$$Ax = \lambda x, \qquad [4]$$

where A is a real square matrix, λ is its eigenvalue, and x is the corresponding eigenvector. Fig. 3*A* shows the eigenvector circuit, consisting of a self-operated feedback circuit where the voltage vector V developed at the cross-point columns develops a current vector $I = A{\cdot}V$, with the conductance of a cross-point array mapping the matrix A. The output currents are converted into voltages by TIAs with the feedback resistors G_λ mapping the known eigenvalue λ. The outputs of the TIAs are then inverted and fed back to cross-point columns. Combining Ohm's law and Kirchhoff's law, one gets $-A{\cdot}V/G_\lambda = -V$, hence $A{\cdot}V = G_\lambda V$, which satisfies Eq. **4**. As physical voltages and currents can only have real values, the eigenvector circuit only applies to real eigenvalues and eigenvectors. For a positive matrix, according to the Perron–Frobenius theorem (31), the highest eigenvalue must be a positive real number, and its eigenvector also consists of positive real numbers. As a result, the eigenvector of the highest eigenvalue of a positive matrix can always be solved with a cross-point circuit. If the eigenvector of the lowest negative eigenvalue is real, it can also be measured by removing the analog inverters in the feedback circuit (*SI Appendix*, Fig. S16*A*). Note that the eigenvector circuit in Fig. 3*A* works in a self-sustained manner, similar to a positive-feedback oscillator, thanks to the active TIAs establishing the voltage vector V.

The eigenvector circuit in Fig. 3*A* was experimentally demonstrated for an RRAM cross-point array with conductance values G mapping the matrix A (Fig. 3*A*, *Inset*), by computing the eigenvectors for the highest positive eigenvalue ($\lambda_+ = 9.41$) and the lowest negative eigenvalue ($\lambda_- = -3.31$). Fig. 3*B* shows the measured values of the eigenvectors as functions of the normalized eigenvectors obtained by the analytical solutions. The proportionalities between experimental and calculated eigenvectors in the figure indicate the correct physical computation of the eigenvectors.

While the restriction of the solution to the highest/lowest eigenvalues might seem inconvenient, it turns out that for many applications only the highest positive or the lowest negative eigenvalues are concerned. For instance, in the PageRank algorithm (32, 33), which yields the importance scores of webpages for their ranking, the eigenvector of a link matrix is calculated for the highest positive eigenvalue. The latter is always equal to 1, since the link matrix is a stochastic matrix (33). Fig. 3*C* shows an example of four pages with their respective links while Fig. 3*D* shows the corresponding link matrix which was implemented as the conductance values of a 4 × 4 RRAM cross-point array. Using the eigenvector circuit of Fig. 3*A*, the eigenvector of the link matrix was solved to compute the importance scores of the pages. Fig. 3*E* shows the experimental scores compared with the analytical scores, demonstrating good accuracy of physically computing the eigenvector. A real-world case of PageRank is reported in *SI Appendix*, Fig. S17.

The analysis of the eigenvector circuit of Fig. 3*A* shows that G_{loop} should be ideally equal to 1 (*SI Appendix*, Fig. S18), which however can never be exactly satisfied in practical circuits. In practice, G_λ can be experimentally chosen such that G_{loop} is slightly larger than 1, which enables the correct solution of the eigenvector with an acceptable error. In fact, although the output initially increases due to $G_{loop} > 1$, the circuit nonlinearity arising from the saturation of the TIA output reduces the G_{loop} to 1. On the other hand, setting G_{loop} smaller than 1 results in vanishing

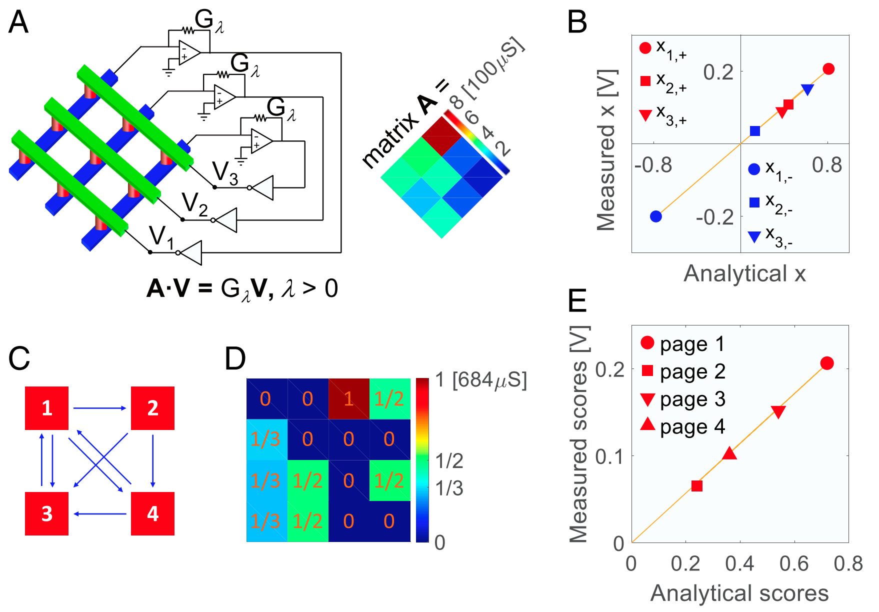

Fig. 3. Eigenvector and PageRank calculations. (*A*) Cross-point circuit for the solution of the eigenvector equation $Ax = \lambda x$, where x is the eigenvector and λ is the highest positive eigenvalue of a positive matrix A reported in the inset. To prevent set/reset disturb to the RRAM conductance, the output voltages of the OAs were limited to ±0.2 V. (*B*) Measured eigenvectors corresponding to the highest positive eigenvalue and the lowest negative eigenvalue, as a function of the normalized eigenvectors obtained by analytical solutions. The highest positive eigenvalue and the lowest negative eigenvalue were stored as the feedback conductance G_λ of the TIAs with conductance of 940 and 331 μS, respectively. (*C*) A system of four webpages with their respective links. An arrow pointing from page *i* to page *j* indicates a citation of *j* in page *i*, thus the importance of a webpage can be stated from the number of arrows pointing at that page. (*D*) The link matrix for the system in *C*. The elements in each column sum to 1, while diagonal elements are all null as pages do not cite themselves. The transformation unit was $G_0 = 684$ μS for RRAM conductance, to minimize RRAM nonlinearity. The highest positive eigenvalue is 1, corresponding to feedback resistors with conductance G_0. (*E*) Measured eigenvector, representing the importance scores of four pages, as a function of the analytically solved normalized eigenvector.

output voltages, which should thus be avoided. Similar to Fig. 2*A*, the eigenvector solution can be extended to a mixed matrix A by the splitting technique with two cross-point arrays connected by analog inverters (*SI Appendix*, Fig. S16*B*).

We validated the physical computation of eigenvectors for the solution of the one-dimensional time-independent Schrödinger equation:

$$H\Psi = E\Psi, \qquad [5]$$

where H is the Hamiltonian operator, E is an energy eigenvalue, and Ψ is the corresponding eigenfunction. Eq. **5** can be numerically solved by the finite difference method, yielding an eigenvector problem given by Eq. **4**, where A is a tridiagonal matrix of coefficients, x is the vector of Ψ values at discrete positions, and λ is the highest/lowest eigenvalue. The Schrödinger equation was solved for a square potential well shown in Fig. 4*A*, which was divided equally into 32 segments (*SI Appendix*, Figs. S19 and S20). Fig. 4*B* shows the 33 × 33 tridiagonal mixed matrix A describing the eigenvector equations. The matrix A is split into two positive tridiagonal matrices B and C, which are mapped into the conductance values of two cross-point arrays, respectively. The eigenvector was calculated for the ground state with energy $E = -4.929$ eV, corresponding to the lowest negative eigenvalue of the problem. The eigenvalues and eigenvectors obtained by numerical solution in a digital computer are also reported in *SI Appendix*, Fig. S19. Fig. 4*C* shows the eigenvector obtained by a simulated eigenvector circuit, compared with the analytically computed eigenvector. The physically computed wave function agrees well with the numerical solution, which further supports physical computing in cross-point circuits for real-world applications.

Discussion

Cross-point arrays allow solutions of a broad set of algebra problems, from linear systems to eigenvector problems, thus enabling the physical solution of differential equations describing

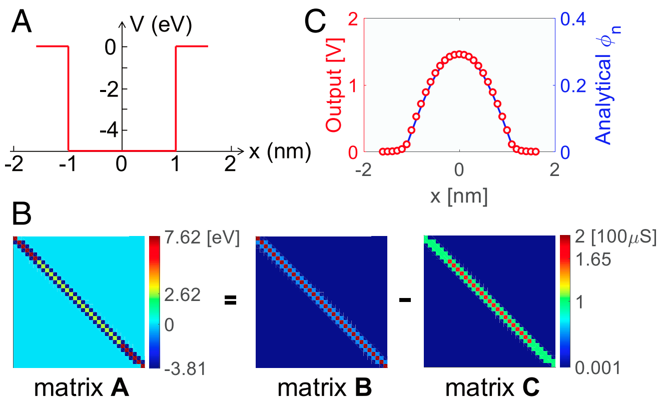

Fig. 4. Solution of the Schrödinger equation in a cross-point circuit. (*A*) Rectangular well of potential $V(x)$ adopted in the Schrödinger equation. The potential well has a depth of −5 eV and a width of 2 nm, while the solution is conducted on an overall width of 3.2 nm, discretized in 32 equal intervals. (*B*) Matrix A with size 33 × 33 obtained from the space discretization of the Schrödinger equation, and the two positive matrices B and C implemented in the cross-point arrays, with $A = B - C$. A conversion unit of 100 μS for 7.6195 eV was adopted in matrices B and C. The two conductance matrices share the same color bar. The ground-state eigenvalue is −4.929 eV, which was mapped into the conductance (65 μS) of the TIA feedback resistors. (*C*) Discrete ground-state eigenfunction obtained as the simulated output voltage in the cross-point circuit compared with the analytical solutions. Note that the peak voltage is around the supply voltage 1.5 V of the OA due to saturation.

real-world problems in industry, economy, and health. The solution relies on extremely simple circuit elements, such as commercially available OAs and state-of-the-art resistive memories such as RRAM and PCM. In comparison, previous solutions of linear systems by using a quantum-computing approach (34, 35) are less attractive, since quantum circuits generally operate at cryogenic temperature and require dedicated instrumentation and noncommercial technology. Other proposed solutions with neural network architectures (36) or CMOS-based analog accelerators (37) rely on iterative operations, resulting in polynomial computation time and cost. In contrast, the cross-point array allows for fast solution in just one step without iteration. The computing time is limited by the settling time of the OA, which can reach the few-nanosecond range in advanced CMOS technology (38).

To fulfill the expectations of practical applications, the cross-point circuit should be scaled up to demonstrate the circuit feasibility. To demonstrate the scalability of the cross-point circuit, solving a system of linear equations for a 100 × 100 model coefficient matrix in simulation is shown in *SI Appendix*, Fig. S21. The results show that the linear system is precisely solved by the circuit, which supports the suitability of the cross-point circuit to address real-world problems. Since the matrix coefficients are stored in real nanoscale devices with inherent stochastic variations, the cross-point circuit only provides an approximate solution to the linear problem. To evaluate the impact of device variations, we included a random deviation to the conductance of each cross-point device for the 100 × 100 matrix and calculated the relative errors of the output voltages (*SI Appendix*, Fig. S22). The simulation results show relatively low errors (around 10%), even with a variation of 10%. High-precision storage of conductance values by program-and-verify techniques is thus essential to improving the accuracy of the solution, depending on the specific applications. Nonlinear conduction in the resistive element, physically arising from hopping conduction and local Joule heating, also affects the accuracy of the solution. Conduction linearity can be maximized by increasing the device conductance (5), which however leads to a higher energy demand for reconfiguring and operating the cross-point circuit. Advancing the technology of resistive memories, aiming at a higher accuracy of multilevel placement and better linearity of conduction, can boost the cross-point circuit for in-memory computing of linear algebra.

As the cross-point circuit scales up, the parasitic resistance due to the dense interconnect wiring in the memory array can become an additional concern. To assess the impact of the parasitic resistance, we simulated the same 100 × 100 linear system of *SI Appendix*, Fig. S21 with an additional parasitic wire resistance (*SI Appendix*, Fig. S23). As a reference, the interconnect parameters were taken from the International Technology Roadmap for Semiconductors at the 65- and the 22-nm technology nodes (39). The relative errors are found to be within ∼10 and 30% for 65- and 22-nm nodes, respectively. These results suggest that there is a tradeoff between scaling and accuracy of circuit-based solutions of algebra problems. It should also be noted that the computing errors are essentially dictated by the resistance ratio between the device resistance and the parasitic resistance. As a result, the computing accuracy might be improved by increasing the resistances of memory devices, which in turn might raise an issue about the conduction nonlinearity that also affects the computing accuracy. We conclude that there is a complex tradeoff between scaling, parasitic resistance, and device nonlinearity, to optimize the operations (40, 41). In this scenario, 3D integration of the cross-point memory, where density does not necessarily cause an increase of interconnect resistance, may improve the immunity of computing accuracy to the parasitic resistance (42).

While the lack of iteration is a highly attractive feature for fast computation, the time needed to program individual matrix coefficients in the memory should also be considered for a comprehensive assessment of the technology. Although the write time in our devices was relatively long for the purpose of a tight tuning of conductance values (see, e.g., *SI Appendix*, Fig. S8), the programming time in a real application might be strongly accelerated thanks to the parallel programming (28, 29), analog programming schemes (43), in addition to the subnanosecond switching of RRAM devices (44) and PCM devices (45). Also, according to the concept of in-memory computing, the same data can be frequently reused for computation (42), thus the programming time can play a negligible role in the overall computing time.

Although the accuracy of our scheme cannot be compared with that of a floating-point solution in a high-precision digital computer, it is important to note that the required accuracy might not be high for all applications. There are in fact many cases where a linear algebra problem must be solved in a short time, with a low energy budget, and with sufficient tolerance to errors. For instance, in machine-learning algorithms, the coefficients for classification/recognition can settle with a certain amount of inaccuracy. The network coefficients can be obtained by a pseudoinverse matrix (46), the calculation of which can be accelerated by our approach. Another example is webpage ranking, where the computed website scores should appear in the correct order, although some amount of inaccuracy might still be tolerated for the individual scores. For similar types of application, our circuits can provide a solution with an excellent tradeoff between accuracy, speed, and energy consumption.

In conclusion, solutions of linear algebra problems in cross-point resistive arrays have been presented. Problems such as systems of linear equations, matrix eigenvectors, and differential equations are solved (*i*) in one step (and matrix inversion in N steps), (*ii*) in situ within the cross-point memory array, and (*iii*) via physical laws such as Ohm's law, Kirchhoff's law, and feedback mechanisms in closed-loop circuits. The proposed in-memory computing paves the way for future in-memory approximate computing systems to solve practical big-data problems with huge savings of time and energy for a wide range of real-world applications.

Methods

Details of device fabrication and characterization, circuit design and measurement techniques are reported in *SI Appendix*.

ACKNOWLEDGMENTS. This work has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program (Grant Agreement 648635). This work was partially performed at Polifab, the micro- and nanofabrication facility of Politecnico di Milano.

1. Golub GH, van Loan CF (2013) *Matrix Computations* (Johns Hopkins Univ Press, Baltimore), 4th Ed.
2. Tan L, et al. (2014) A survey of power and energy efficient techniques for high performance numerical linear algebra operations. *Parallel Comput* 40:550–573.
3. Waldrop MM (2016) The chips are down for Moore's law. *Nature* 530:144–147.
4. Horowitz M (2014) Computing's energy problem (and what we can do about it). *International Solid-State Circuits Conference Digest of Technical Papers* (IEEE, Piscataway, NJ), pp 10–14.
5. Li C, et al. (2018) Analogue signal and image processing with large memristor crossbars. *Nat Electron* 1:52–59.
6. Sheridan PM, et al. (2017) Sparse coding with memristor networks. *Nat Nanotechnol* 12:784–789.
7. Burr GW, et al. (2015) Experimental demonstration and tolerancing of a large-scale neural network (165 000 synapses) using phase-change memory as the synaptic weight element. *IEEE Trans Electron Dev* 62:3498–3507.
8. Prezioso M, et al. (2015) Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* 521:61–64.
9. Waser R, Dittmann R, Staikov G, Szot K (2009) Redox-based resistive switching memories–Nanoionic mechanisms, prospects, and challenges. *Adv Mater* 21:2632–2663.
10. Ielmini D, Waser R (2015) *Resistive Switching: From Fundamentals of Nanoionic Redox Processes to Memristive Device Applications* (Wiley-VCH, Weinheim, Germany).
11. Wong H-SP, et al. (2012) Metal-oxide RRAM. *Proc IEEE* 100:1951–1970.
12. Ielmini D (2016) Resistive switching memories based on metal oxides: Mechanisms, reliability and scaling. *Semicond Sci Technol* 31:063002.
13. Borghetti J, et al. (2010) 'Memristive' switches enable 'stateful' logic operations via material implication. *Nature* 464:873–876.
14. Balatti S, Ambrogio S, Ielmini D (2015) Normally-off logic based on resistive switches–Part I: Logic gates. *IEEE Trans Electron Dev* 62:1831–1838.
15. Sun Z, Ambrosi E, Bricalli A, Ielmini D (2018) Logic computing with stateful neural networks of resistive switches. *Adv Mater* 30:e1802554.
16. Traversa FL, Ramella C, Bonani F, Di Ventra M (2015) Memcomputing NP-complete problems in polynomial time using polynomial resources and collective states. *Sci Adv* 1:e1500031.
17. Hu M, et al. (2016) Dot-product engine for neuromorphic computing: Programming 1T1M crossbar to accelerate matrix-vector multiplication. *Proceedings of the 53rd Annual Design Automation Conference* (ACM, New York), pp 1–6.
18. Pedretti G, et al. (2017) Memristive neural network for on-line learning and tracking with brain-inspired spike timing dependent plasticity. *Sci Rep* 7:5288.
19. Ielmini D (2018) Brain-inspired computing with resistive switching memory (RRAM): Devices, synapses and neural networks. *Microelectron Eng* 190:44–53.
20. Raoux S, Wełnic W, Ielmini D (2010) Phase change materials and their application to nonvolatile memories. *Chem Rev* 110:240–267.
21. Chappert C, Fert A, Van Dau FN (2007) The emergence of spin electronics in data storage. *Nat Mater* 6:813–823.
22. Le Gallo M, et al. (2018) Mixed-precision in-memory computing. *Nat Electron* 1: 246–253.
23. Yu S, Wu Y, Wong H-SP (2011) Investigating the switching dynamics and multilevel capability of bipolar metal oxide resistive switching memory. *Appl Phys Lett* 98:103514.
24. Sedra AS, Smith KC (2003) *Microelectronic Circuits* (Oxford Univ Press, Oxford).
25. International Roadmap for Devices and Systems (IRDS) (2017) 2017 Ed. Available at https://irds.ieee.org/roadmap-2017. Accessed November 20, 2018.
26. Richter I, et al. (2015) Memristive accelerator for extreme scale linear solvers. *Government Microcircuit Applications & Critical Technology Conference (GOMACTech)*. Available at www2.ece.rochester.edu/~xiguo/gomac15.pdf. Accessed February 9, 2019.
27. Horowitz P, Hill W (1989) *The Art of Electronics* (Cambridge Univ Press, New York).
28. Gao L, et al. (2015) Fully parallel write/read in resistive synaptic array for accelerating on-chip learning. *Nanotechnology* 26:455204.
29. Gokmen T, Vlasov Y (2016) Acceleration of deep neural network training with resistive cross-point devices: Design considerations. *Front Neurosci* 10:333.
30. Higham NJ (2002) *Accuracy and Stability of Numerical Algorithms* (SIAM, Philadelphia).
31. Berman A, Plemmons RJ (1994) *Nonnegative Matrices in the Mathematical Sciences* (SIAM, Philadelphia).
32. Page L, Brin S, Motvani R, Winograd T (1998) The PageRank citation ranking: Bringing order to the web (Stanford InfoLab, Stanford, CA), Technical Report.
33. Bryan K, Leise T (2006) The $25,000,000,000 eigenvector: The linear algebra behind google. *SIAM Rev* 48:569–581.
34. Harrow AW, Hassidim A, Lloyd S (2009) Quantum algorithm for linear systems of equations. *Phys Rev Lett* 103:150502.
35. Zheng Y, et al. (2017) Solving systems of linear equations with a superconducting quantum processor. *Phys Rev Lett* 118:210504.
36. Cichocki A, Unbehauen R (1992) Neural network for solving systems of linear equations and related problems. *IEEE Trans Circ Syst* 39:124–138.
37. Huang Y, Guo N, Seok M, Tsividis Y, Sethumadhavan S (2016) Evaluation of an analog accelerator for linear algebra. *Proceedings of the ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)* (IEEE, Piscataway, NJ), pp 570–582.
38. Seth S, Murmann B (2013) Settling time and noise optimization of a three-stage operational transconductance amplifier. *IEEE Trans Circuits Syst I Regul Pap* 60: 1168–1174.
39. International Technology Roadmap for Semiconductors (ITRS), Available at www.itrs2.net/2013-itrs.html. Accessed November 20, 2018.
40. Chen P-Y, et al. (2015) Technology-design co-optimization of resistive cross-point array for accelerating learning algorithms on chip. *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE)* (IEEE, Piscataway, NJ), pp 854–859.
41. Xia L, et al. (2016) Technological exploration of RRAM crossbar array for matrix-vector multiplication. *J Comput Sci Technol* 31:3–19.
42. Ielmini D, Wong H-SP (2018) In-memory computing with resistive switching devices. *Nat Electron* 1:333–343.
43. Liu J-C, Wu T-Y, Hou T-H (2018) Optimizing incremental step pulse programming for RRAM through device–circuit co-design. *IEEE Trans Circuits Syst II* 65:617–621.
44. Lee HY, et al. (2010) Evidence and solution of over-RESET problem for HfOx based resistive memory with sub-ns switching speed and high endurance. *IEEE Electron Devices Meeting* (IEEE, Piscataway, NJ), pp 19.7.1–19.7.4.
45. Bruns G, et al. (2009) Nanosecond switching in GeTe phase change memory cells. *Appl Phys Lett* 95:043108.
46. Bishop CM (2006) *Pattern Recognition and Machine Learning* (Springer, New York).