ENGINEERING

In-memory analog solution of compressed sensing recovery in one step

Shiqing Wang, Yubiao Luo, Pushen Zuo, Lunshuai Pan, Yongxiang Li, Zhong Sun*

Modern analog computing, by gaining momentum from nonvolatile resistive memory devices, deals with matrix computations. In-memory analog computing has been demonstrated for solving some basic but ordinary matrix problems in one step. Among the more complicated matrix problems, compressed sensing (CS) is a prominent example, whose recovery algorithms feature high-order matrix operations and hardware-unfriendly nonlinear functions. In light of the local competitive algorithm (LCA), here, we present a closed-loop, continuous-time resistive memory circuit for solving CS recovery in one step. Recovery of one-dimensional (1D) sparse signal and 2D compressive images has been experimentally demonstrated, showing elapsed times around few microseconds and normalized mean squared errors of 10^{-2}. The LCA circuit is one or two orders of magnitude faster than conventional digital approaches. It also substantially outperforms other (electronic or exotically photonic) analog CS recovery methods in terms of speed, energy, and fidelity, thus representing a highly promising technology for real-time CS applications.

INTRODUCTION

Compressed sensing (CS) has been the cornerstone of modern signal and image processing (*1*). It finds applications in typical scenarios such as medical imaging (*2*), wireless communication (*3*), object tracking (*4*), and single-pixel camera (*5*). In CS, a sparse signal (in a given basis) is highly undersampled in the front-end sensor, which breaks through the Nyquist rate and thus improves remarkably the sampling efficiency. In the back-end processor, the original signal can be faithfully recovered by solving a sparse approximation problem, which, however, is intractable and has become the accepted bottleneck in the CS pipeline. CS recovery algorithms running on digital computers are doomed to be computationally intensive, comprising matrix-matrix operations and pointwise nonlinear functions in discrete time, where the former alone contribute a cubic computational complexity. To speed up the CS recovery processing, there have been two lines of efforts in the digital domain, using either advanced algorithms such as deep learning (*6*, *7*), or alternative computing hardwares (*8*, *9*). However, the computing efficiency is fundamentally bounded by the polynomial complexity of matrix operations, e.g., matrix multiplication, which is unlikely to collapse. Analog computing is promising to provide an enhanced acceleration for solving matrix problems because of the inherent computing parallelism, the continuous-time solution, and the high information capacity (*10*). However, again, because of the extraordinarily high complexity of CS recovery, previous demonstrations either rely on the precalculation of the Gram matrix that preserves the cubic complexity (*11*) or bare the discrete iterative process that requires expensive but frequent analog-digital conversions (*12*, *13*). Therefore, solving CS recovery in one step with high speed remains a grand challenge.

Recently, analog matrix computing (AMC) has been demonstrated for performing the prodigious feats of solving matrix equations in one step, offering orders of magnitude improvement of equivalent throughput and energy efficiency in applications including scientific computing, machine learning, PageRank, and wireless communications (*14*–*17*). Although traditional analog complementary metal-oxide semiconductor (CMOS) circuits might be adopted for AMC (*18*), emerging resistive memory devices [or memristors (*19*)] are advantageous because of their simple structure, high density integration, and high operation speed. In addition, in-memory AMC is very beneficial to alleviating the infamous von Neumann bottleneck in the conventional computers (*20*). There are a number of resistive memory concepts, whose underlying mechanisms range from ion migration to phase change, ferroelectric, or ferromagnetic polarity reversal (*21*). Nevertheless, they are used for in-memory AMC with the same principle, by exploiting the device resistance attribute for information encoding and the circuit physics laws such as Ohm's law and Kirchhoff's current law (KCL) for matrix/vector arithmetic (*22*). By connecting a crosspoint resistive memory array with operational amplifiers (OPAs) to form feedback loops, a matrix inversion problem (i.e., a system of linear equations) is solved in continuous time within a few nanoseconds (*14*). It is theoretically proven that the time complexity of AMC can be optimized to $O(1)$, which surpasses the logarithmic complexity of quantum algorithms for the same problems (*23*).

Contrary to the linear but smooth matrix computations, CS recovery with in-memory AMC is unintuitive, where the integration of complicated matrix operations and non-smooth nonlinear functions must be overcome, thus invoking innovative analog computing principle and ingenious circuit design. On the other hand, the nonlinear operation in CS recovery should help suppress the computation errors below the threshold and prevent their accumulation, thus achieving high accuracy in AMC, similar to the case of neural networks that have been extensively investigated. In this work, first, on the basis of a crosspoint resistive memory array and by adopting the conductance compensation (CC) strategy, we have designed a highly compact circuit for performing the in-memory matrix-matrix-vector multiplication (MMVM) in one step, which is pivotal to the elimination of Gram matrix precalculation and discrete iterations for efficient in-memory AMC circuit design.

School of Integrated Circuits, Institute for Artificial Intelligence and Beijing Advanced Innovation Center for Integrated Circuits, Peking University, Beijing 100871, China.
*Corresponding author. Email: zhong.sun@pku.edu.cn

Then, on the basis of this Gram module and the local competitive algorithm (LCA), we have designed a closed-loop circuit for solving the sparse approximation problem in one step. Nonlinear function modules for regularizing the output sparsity are included in the feedback loop, which interact with the Gram module in a fully analog manner, thus saving analog-digital conversions to maximize the speed and energy efficiency of CS recovery.

RESULTS

CS recovery and LCA

CS is a technique for efficiently measuring and reliably recovering a sparse signal, which are modeled as a feed-forward matrix multiplication and an inverse matrix problem solving, respectively (Fig. 1A). The two processes own distinctly different algorithmic complexities; consequently, the hardware implementation of CS recovery should be much more complicated than the one of CS measurement (*24*–*26*). Specifically, CS recovery is about solving an underdetermined linear system $\boldsymbol{\Psi x} = \boldsymbol{y}$ ($\boldsymbol{y} \in \mathbb{R}^N$, $\boldsymbol{x} \in \mathbb{R}^M$, $\boldsymbol{\Psi} \in \mathbb{R}^{N \times M}$, and $N < M$), but subject to solution sparsity constraint. Consequently, it becomes a nonlinear optimization problem whose objective function is the combination of the recovery mean squared error and a sparsity-inducing penalty term. Ideally, the ℓ_0-norm regularization should be imposed during the problem solving to count the nonzero elements, which, however, is non-convex and non-deterministic polynominal-time (NP)-hard (*27*). To this end, the ℓ_1-norm can be used as a convex surrogate, transforming the problem to Eq. 1, which is known as basis pursuit denoising (BPDN). It has been proven that, in many practical cases of interest, Eq. 1 has the same solution as the optimal sparse approximation problem (*28*)

$$\min_{\boldsymbol{x}} \left(\frac{1}{2} \|\boldsymbol{y} - \boldsymbol{\Psi x}\|_2^2 + \lambda \|\boldsymbol{x}\|_1 \right) \tag{1}$$

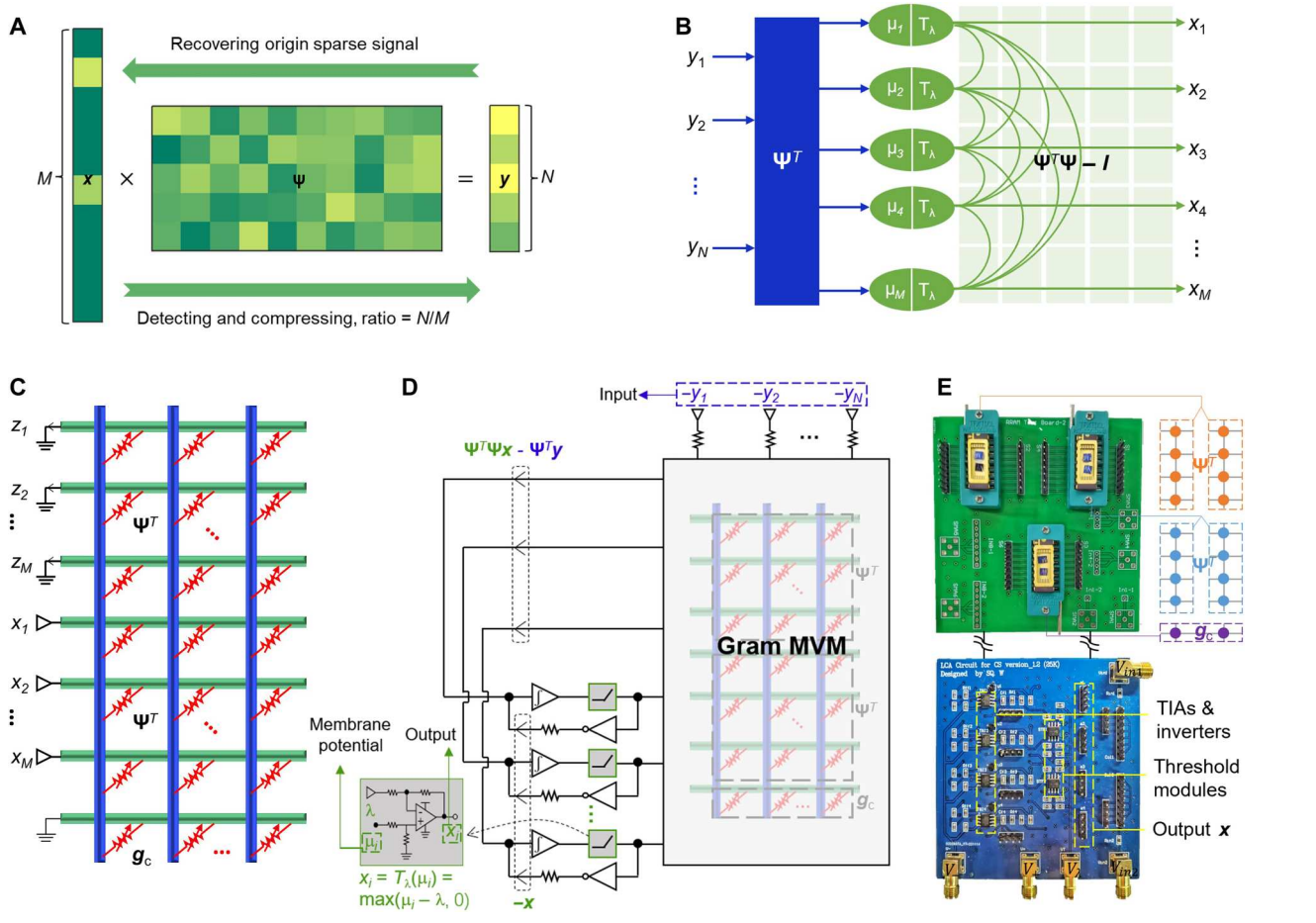

Fig. 1. In-memory LCA circuit for solving CS recovery in one step. (**A**) CS measurement and recovery, between a high-dimensional sparse signal vector $\boldsymbol{x}$ and the detected low-dimensional vector $\boldsymbol{y}$, through an overcomplete dictionary matrix $\boldsymbol{\Psi}$. (**B**) Schematic illustration of LCA. There are N inputs and M neurons. The inputs forwardly fed to the neurons through $\boldsymbol{\Psi}$ reflect how well the input signal matches each dictionary column. μ_i and x_i represent the internal state and output of the ith neuron, respectively, which are bridged by a soft thresholding function. The $\boldsymbol{\Psi}^T\boldsymbol{\Psi} - \boldsymbol{I}$ network represents the feedback connections between every pair of neurons. (**C**) In-memory Gram MVM circuit. Two copies of matrix $\boldsymbol{\Psi}^T$ and compensation values are mapped as device conductances in the crosspoint resistive memory array. To enable the exact mapping of Gram MVM, the matrix is magnified by a constant c, which is related to the row sums of $\boldsymbol{\Psi}$. (**D**) LCA circuit. It is composed of the Gram MVM module for implementing the $\boldsymbol{\Psi}^T\boldsymbol{\Psi x}$ term, analog inverters for inverting the sign of $\boldsymbol{x}$, TIAs for summing the two terms, soft thresholding modules for implementing the nonlinear function $T_\lambda(\cdot)$, and input terminals for submitting $\boldsymbol{y}$. The inset shows the composition of the soft threshold module based on an analog subtractor. All discrete resistors have a unit conductance g_0 for current-voltage conversion. (**E**) LCA circuit experimental setup.

In Eq. 1, λ is a parameter for weighting the signal sparsity during CS recovery, and $||\cdot||_2$ and $||\cdot||_1$ denote the ℓ_2- and ℓ_1-norm, respectively. Despite being convex, the objective function of BPDN contains a non-smooth nonlinearity, which induces a higher complexity than linear problems, say matrix inversion (*14*). There are several algorithms for solving the BPDN, such as interior-point methods (*29*), gradient projection (*30*), iterative shrinkage, and thresholding algorithm (*31*), which all show a high time complexity running on digital computers, in large part due to the time-consuming matrix operations with the Gram matrix $\boldsymbol{\Psi}^T\boldsymbol{\Psi}$. Such a burden should also be found in other CS recovery algorithms, such as orthogonal matching pursuit (*32*) and approximate message passing (*33*). By contrast, LCA, which is designed for parallel analog architecture of neural systems (*34*), is intriguing to provide massive computing parallelism in continuous time (Fig. 1B). It can be easily implemented in a parallel analog hardware, e.g., a crosspoint resistive memory array, to offer a fast computing speed. By using the soft thresholding function as the nonlinear operator, LCA also has the advantage of keeping coefficients identically zero during the computation unless they become active, thus achieving low power consumption. It is described by a first-order nonlinear matrix differential equation, i.e., Eq. 2, whose steady state provides the solution to Eq. 1. As a result, it is most straightforward to implement the algorithm in continuous time, to take full advantage of its computational parallelism, fast convergence, as well as the elimination of intermediate data storage and conversions.

$$\begin{aligned} \frac{d\mathbf{u}(\mathbf{t})}{dt} &= \frac{1}{\tau}[-\mathbf{u}(\mathbf{t}) + \boldsymbol{\Psi}^T\boldsymbol{y} - (\boldsymbol{\Psi}^T\boldsymbol{\Psi} - \boldsymbol{I})\boldsymbol{x}(\mathbf{t})] \\ \boldsymbol{x}(t) &= T_\lambda[\mathbf{u}(\mathbf{t})] = \max\,[\mathbf{u}(\mathbf{t}) - \boldsymbol{\lambda}, \mathbf{0}] \end{aligned} \tag{2}$$

where each element in vector $\mathbf{u}$ is the internal state variable resembling a neuron′s membrane potential, τ is a time constant, $\boldsymbol{I}$ is the $M \times M$ identity matrix, $\boldsymbol{\Psi}^T$ is the transpose of matrix $\boldsymbol{\Psi}$, and $T_\lambda(\cdot)$ is the soft thresholding function [also known as shrinkage function in the iterative shrinkage and thresholding algorithm (*35*)] with a threshold λ. The threshold λ determines the degree of penalty of the ℓ_1-norm, which results in a trade-off between recovery accuracy and solution sparsity. In LCA, each column of matrix $\boldsymbol{\Psi}$ has been normalized, and the term $\boldsymbol{\Psi}^T\boldsymbol{\Psi} - \boldsymbol{I}$ means that each neuron provides feedback to all other neurons except for itself.

In-memory Gram module and LCA circuit

To implement LCA with AMC, one major obstacle is related to the processing of the Gram matrix $\boldsymbol{\Psi}^T\boldsymbol{\Psi}$ in the computation dynamics. By calculating the $\boldsymbol{\Psi}^T\boldsymbol{\Psi}$ product in the first place and storing it in an analog floating gate transistor array, it is possible to perform LCA in continuous time by configuring feedback loops with analog thresholder circuits (*11*). However, the precalculation process should drag down the overall efficiency of this approach due to the high complexity of matrix-matrix multiplication and expense of data movement in conventional digital computer, as well as the large volume of data conversion for matrix mapping in the analog array. The product of a Gram matrix and a vector might be obtained by performing two successive in-memory matrix-vector multiplication (MVM) operations (*36*), which, however, suits only to the discretized iterations of LCA and requires frequent analog-digital conversions (*13*). To perform in-memory MMVM (or specifically Gram MVM), we have built a Gram module circuit based on the CC principle (Fig. 1C). It effectively solves the issues of expensive digital computations and/or data conversions in earlier approaches.

Briefly, it is built with a $(2M + 1) \times N$ crosspoint resistive memory array, where the first $2M$ rows are used to map two copies of matrix $\boldsymbol{\Psi}^T$. The input vector $\boldsymbol{x}$ is represented by the voltages applied to the bottom M rows, while the results are collected at the top M grounded rows. The last row of devices in the array is for CC to let the conductance sum of each column equal to a constant, say c. The CC values are determined on the basis of the prior knowledge of the row sums of $\boldsymbol{\Psi}$, which might be calculated in advance with little computational overhead. Then, according to KCL in the array, the potentials on the column lines constitute a vector $\boldsymbol{\Psi x}/c$, and the collected currents on the top M rows represent the Gram MVM result $\boldsymbol{\Psi}^T\boldsymbol{\Psi x}/c$. Instead, by mapping the matrix $c\boldsymbol{\Psi}^T$ in the resistive memory array, the exact result $\boldsymbol{\Psi}^T\boldsymbol{\Psi x}$ should be obtained. In addition to the Gram MVM, this module could be used to accelerate general MMVM by mapping two different matrices in the array. The Gram MVM circuit can also be extended to matrices that contain negative values (fig. S1). The detailed explanation of the individual Gram modules is presented in text S1.

According to the LCA in Eq. 2, the Gram module is combined with other traditional analog components to construct a closed-loop in-memory AMC circuit (termed LCA circuit for simplicity) for solving CS recovery (Fig. 1D). In the circuit, the analog inverters are used to cancel out the diagonal contribution of the Gram product $\boldsymbol{\Psi}^T\boldsymbol{\Psi}$, i.e., delivering the current vector $-\boldsymbol{x}(t)$. Although the crosspoint array is dedicated for Gram MVM, it can also be directly used for incepting the input voltage vector $-\boldsymbol{y}$, thus forming a current vector $\boldsymbol{\Psi}^T\boldsymbol{\Psi x}(t) - \boldsymbol{\Psi}^T\boldsymbol{y}$, again because of the identical column sums guaranteed by CC in the array. Note that the matrix to be mapped is $c_1\boldsymbol{\Psi}^T$ ($c_1 = c + 1$, text S1), where the number 1 is the unit conductance in the linear mapping. The two current vectors are summed up by a set of transimpedance amplifiers (TIAs) to produce the vector $\mathbf{u}(t)$. The single-pole low-pass filter characteristics of TIAs also account for the differential (or equivalently integral) operation in Eq. 2, generating dynamical output voltages toward the final solution (*23*). Following the TIAs, the outputs are delivered to the soft thresholding module to perform the pointwise nonlinear operation $T_\lambda(\cdot)$, resulting in a closed loop to approach the solution vector $\boldsymbol{x}(t)$ at the equilibrium. The soft thresholding module is a modified analog subtractor with a defined threshold value for subtraction but supplied with a single positive voltage source to effectively suppress the negative outputs. The TIA and the nonlinear module could be merged to deliver a more compact design (fig. S2), but the circuit in Fig. 1D is more suitable for explanation. In fig. S3, the experimental nonlinear transfer characteristics of the soft thresholding module are shown, with the threshold λ ranging from 10 to 90 mV, demonstrating that the signals below λ are transferred to a zero output. In text S2, we have performed sophisticated circuit analysis based on Laplace transform; it turns out that the circuit maps faithfully the LCA with slight approximations. On the basis of the Gram MVM circuit for matrices that contain negative values, the LCA circuit for arbitrary real-valued CS recovery is obtained (fig. S4). Since LCA is used for solving general sparse approximation problems, the circuit can be applied to other tasks, e.g., sparse coding (*13*).

Experimental results

To perform experimental demonstrations of in-memory Gram MVM and LCA circuits, we have fabricated crosspoint resistive memory arrays using a high-performance and industry-ready resistive random-access memory (RRAM) technology, which comprises a HfO_2 dielectric layer and a Ta metal layer (see Materials and Methods) (*37*). The resistive switching behavior, current-voltage (*I-V*) linearity, and retention of eight typical states are shown in fig. S5. It demonstrates good analog conductance capability, high retention (tested for 10^4 s), and high endurance (tested for 1.5×10^5 cycles), which are beneficial to the LCA circuit applications. In the experiment, the matrix $\boldsymbol{\Psi}$, e.g., some canonical CS measurement matrices, is mapped in a crosspoint RRAM array with reference to a unit conductance g_0, which is 40 μS in this work. The mapping to conductance is achieved by using a write-verify process, under the constraint of a predefined verify window (*14*). After mapping, the RRAM device conductance ranges from 100 to 350 μS, except for some large compensation conductance, which might be distributed in several devices to adapt to the RRAM conductance range. To map matrices with large values, e.g., $c\boldsymbol{\Psi}^T$ resulting from a large sum constant, a small unit conductance g_0 for matrix mapping should be used, which should be achievable by setting a predefined scaling factor. Zeros in the matrix are mapped represented by the deep high resistance state of RRAM devices, whose *I-V* curve is shown in fig. S5. All of the matrices used in Gram MVM and LCA circuit experiments are reported in table S1. Figure 1E and fig. S1 present the experimental setups for LCA and Gram MVM circuits, respectively. For demonstration purposes, the crosspoint array is composed by connecting several 16 × 1 RRAM columns in a printed circuit board (PCB), which are obtained from cutting off several 16 × 16 arrays and help solve the sneak path issue during programming and reading of RRAM devices (fig. S5).

We have performed the Gram MVM experiment for the case of $N = 2$ and $M = 4$ (Fig. 2A). It is translated to use 9 × 2 or 8 × 2 devices for the case with or without CC devices. Figure 2B shows the experimental results of a sparse checkerboard-like binary matrix. To adapt to the conductance range of RRAM devices in matrix mapping, it is linearly scaled by multiplying a constant 1.4. Consequently, the nonzero elements in the matrix become 1.4 instead, and the sum constant c is 5.6. Note that, for this matrix, since the row sums are equal, the CC strategy is unnecessary. In the experiment, two copies of the matrix $c\boldsymbol{\Psi}^T$ were mapped in the crosspoint RRAM array with a moderate verify window of ±5%, namely, if the resulting device conductance is within ±5% error of the desired value, the iterative write process is stopped. The programming results are also included in the figure. On the basis of the programmed memory array, 12 input voltage vectors were provided to the circuit sequentially, and the Gram MVM output currents were measured. The experimental output results are compared with the ideal ones, showing a good consistency (Fig. 2B). The same experiments have been performed for another matrix, namely, the sensing matrix that will be used in LCA circuit for CS recovery (Fig. 2C). The results of matrix mapping and Gram MVM experiments are also included in the figure. In particular, as the row sums of matrix $\boldsymbol{\Psi}$ are not identical, the CC devices were programmed with the same verify window. The detailed input voltages and output currents monitored in the Gram MVM circuit experiments are shown in Fig. 2D. The computing accuracy is quantitatively measured by calculating the normalized mean squared error (NMSE) that is defined as $\|\boldsymbol{v} - \boldsymbol{v}^*\|_2^2 / \|\boldsymbol{v}^*\|_2^2$ for a given vector, where $\boldsymbol{v}$ and $\boldsymbol{v}^*$ represent the experimental and ideal vector, respectively. The NMSEs of Gram MVM are 1.7×10^{-3} and 3.9×10^{-3} for the two cases, respectively. The larger computing error of the second matrix should be related to the presence of CC in the case. We have performed more Gram MVM experiments for the matrices by programming the RRAM array with a different verify window (±20%) for the two matrices. The results are shown in fig. S6. We have also calculated the NMSE of each matrix mapping (i.e., the Frobenius norm). It turns out that the NMSE of Gram MVM is proportional to it for both matrices, although with different ratios (fig. S7). The results suggest that the precision of Gram MVM can be boosted by optimizing the NMSE of array programming but at the expense of large overhead of the verify process (*38*).

For proof-of-concept demonstration, the LCA circuit is built upon a PCB using off-the-shelf OPAs (see Materials and Methods). To perform the recovery computation, the CS acquisition process is modeled by an MVM operation with the measurement matrix. The measurement matrix is considered with a 2 × 4 size to be accommodated in the crosspoint RRAM array, which implies that the compression rate is 50%. In the case of one-dimensional (1D) sparse signal recovery, the measurement matrix is shown in Fig. 3A, and the matrix mapping results from the verify window of ±5% are shown in Fig. 3B. Note that no CC is needed in this case because of the same sums of two columns in $\boldsymbol{\Psi}^T$. The sampled signal is segmented as 2 × 1 input vectors (14 in total) and provided as voltages to the LCA circuit. Consequently, a 4 × 1 output vector is obtained each time as the recovered partial signal. According to the numerical algorithm results, the threshold λ was empirically set as 10 mV. The complete signal recovery is plotted in Fig. 3C, in comparison with the original sparse signal, showing a great agreement between both, with an NMSE of merely 5.1×10^{-3}.

Image compression is another important application of CS. In this case, as images are usually not sparse in their own, they should be transformed with a sparsity basis $\boldsymbol{A}$. If $\boldsymbol{\Psi}$ (the measurement matrix) and $\boldsymbol{A}$ are incoherent, the original signal can be recovered through the matrix product $\boldsymbol{\Psi A}$, which is termed sensing matrix. In the experiment, the random sparsity matrix is used as the measurement matrix, and the sparsity basis is obtained by training. For simplicity, the symbol $\boldsymbol{\Psi}$ is retained for representing the sensing matrix for CS recovery in the following. Eight natural red-green-blue (RGB) images with size 200 × 112 are used for the training of the sparsity basis $\boldsymbol{A}$ (fig. S8). Each image is divided into 5600 2 × 2 patches in each channel, which are then converted into 4 × 1 vectors for training (fig. S9). A 4 × 4 sparsity basis matrix is obtained that all the 16,800 vectors can be sparsely represented under the same linear transform. The training algorithm is described in Materials and Methods, and the corresponding sparsity basis is shown in fig. S10. By using a 2 × 4 random sparsity matrix as the measurement matrix, the images are compressed by 50% (fig. S9). To recover the original image with the LCA circuit, the sensing matrix is mapped in the crosspoint RRAM array (Fig. 3, D and E). The compressed image vectors are normalized and provided as the input voltages to the circuit. The width of each input pulse is 50 ms, which is always sufficient for the LCA circuit to reach the equilibrium. Since the original large-scale images have been

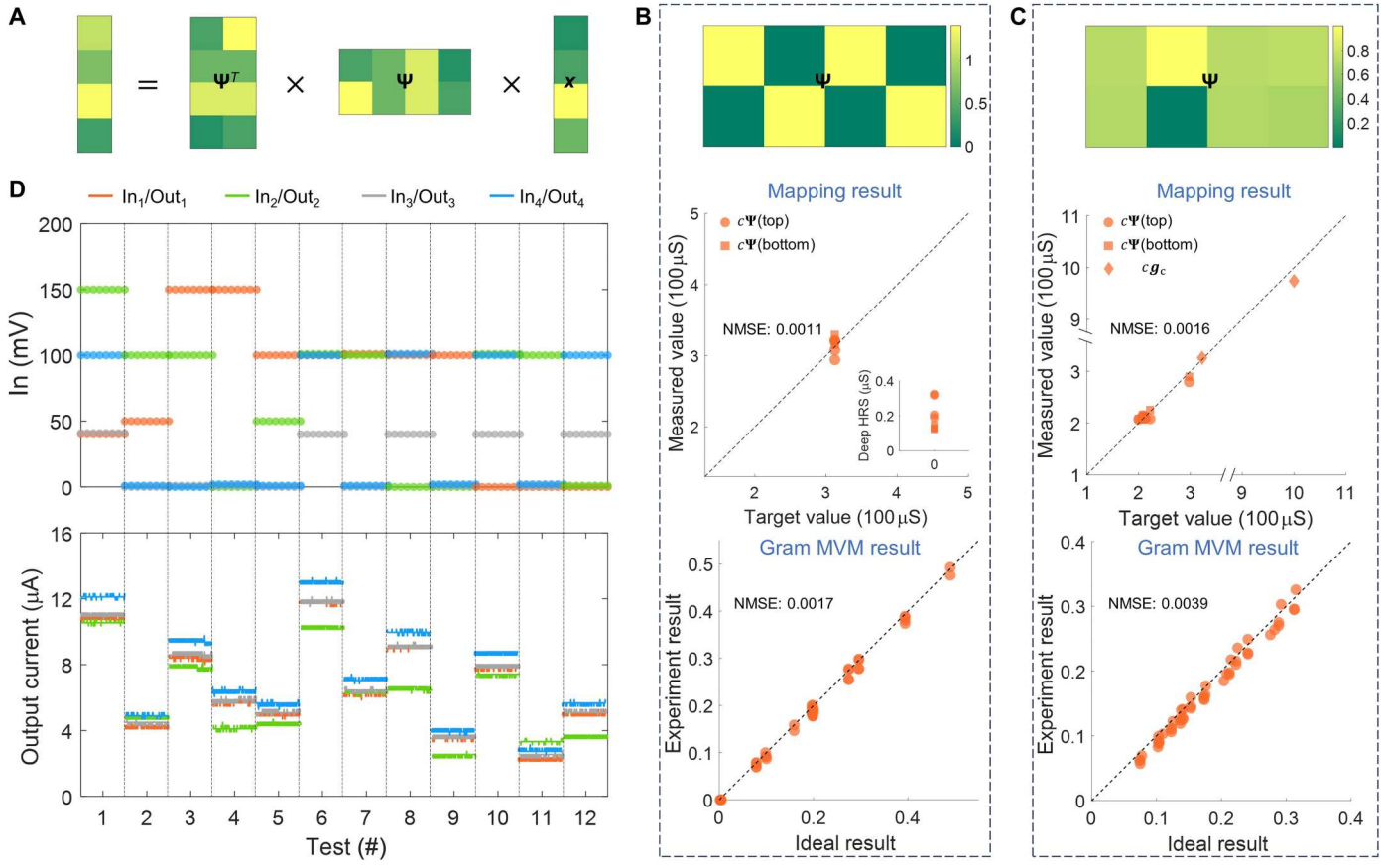

Fig. 2. Experimental results of in-memory Gram MVM. (**A**) Illustration of the Gram MVM operation $\boldsymbol{\Psi}^T\boldsymbol{\Psi}\boldsymbol{x}$. (**B**) A sparse checkerboard-like binary matrix (top), conductance mapping results of two copies of matrix $\boldsymbol{\Psi}^T$ in the crosspoint array (middle), and experimental results of 12 Gram MVM operations with randomly generated input vectors (bottom). Both the mapping results and Gram MVM results are plotted in comparison with the ideal ones. The NMSE of Gram MVM is calculated on the basis of the 12 operations. (**C**) Same as (B), but for a different matrix. In this case, the conductance should be compensated, and the CC values are also included in the mapping results (middle). The input vectors are the same as the ones in (B). (**D**) The 12 input voltage vectors and output current vectors in the Gram MVM experiment of (C).

segmented as small patches, 16,800 successive operations have been performed in the experiment. The original Mona Lisa picture is recovered as shown in Fig. 3F, based on the experimental sparse representations from the LCA circuit (Fig. 3G). The experimental recovery results are compared with the original ground-truth pixels, showing an NMSE of 1.18×10^{-2} (Fig. 3G). Accompanied with the experimental results are the algorithmic results from high-precision digital computer, clearly testifying to the adequacy of the analog LCA circuit for reliable CS recovery of image processing. Another indicator of recovery quality is the peak signal-to-noise ratio (PSNR), which is defined as $10\log_{10}\left(\frac{255^2}{\|\boldsymbol{x}-\boldsymbol{x}^*\|_2^2/M}\right)$ here for natural images. It is 26.86 dB for the experimental recovery, showing a loss of 3.26 dB compared to the algorithmic result. Recovery of two other images is shown in fig. S11.

The application of the LCA circuit is extended to magnetic resonance imaging (MRI), where CS is very beneficial to saving energy dissipation of data acquisition and latency of data transmission (*2*). The subsampled Fourier operator $\boldsymbol{F}_u$ was used to simulate the measurement process in *k*-space, and again, a sparsity basis matrix is trained to sparsely represent the vectors, resulting in the matrix product in Fig. 3H. To recover the MRI result, which is a 178 × 256 grayscale image, dividing as 2 × 2 patches has been performed as usual. On the basis of the programmed RRAM array (Fig. 3I), 11,392 operations have been performed with the circuit, and the final recovered image in shown in Fig. 3J. The NMSE and PSNR are calculated to be 4.42×10^{-2} and 26.83 dB, respectively. Compared to the algorithmic solution, there is an accuracy loss of 3.34 dB. More experiments for the same signal/image recovery tasks, but with verify window of ±20% for device programming, have been performed, showing that the PSNR degradation of CS recovery with the LCA circuit is proportional to the NMSE of array programming (fig. S12).

To study the scalability of both the individual Gram MVM and the LCA circuits, especially the impact of RRAM device and circuit nonideal factors on the computing accuracy, we have performed circuit simulations with a large (N = 32 and M = 64) sensing matrix $\boldsymbol{\Psi}$ (fig. S13). A relatively mature analog RRAM technology (*39*) is applied in the large-scale circuit simulation, where the conductance range of device is 1 to 40 μS. To map the matrices in the large RRAM array with the given conductance range, the unit conductance g_0 is assumed as 2 μS in all large-scale circuit simulations. In Gram MVM circuit, one concern lies in the inaccurate and nonidentical programming of two copies of matrix $\boldsymbol{\Psi}^T$ in the RRAM

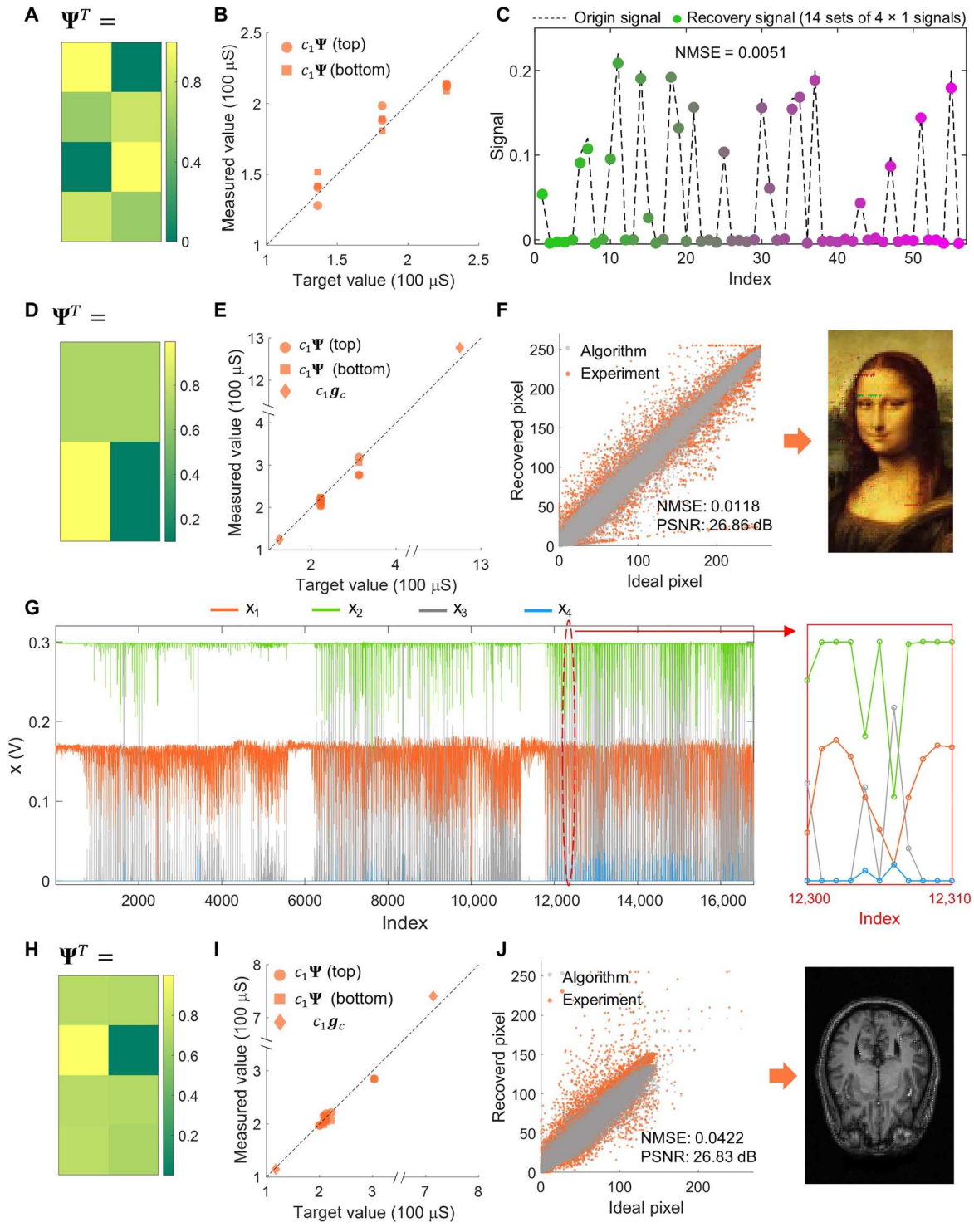

Fig. 3. CS recovery experiments for 1D sparse signal, 2D natural image, and MRI. (**A**) Measurement matrix and (**B**) its mapping results for 1D sparse signal recovery. (**C**) Original sparse signal and the recovered one by the LCA circuit. (**D**) Sensing matrix and (**E**) its mapping results for 2D natural image recovery. (**F**) Experimentally recovered pixels in comparison with the algorithmic solution and the recovered Mona Lisa image. (**G**) The 16,800 experimental 4 × 1 sparse output voltages of the LCA circuit. The rightmost inset shows 10 output vectors in detail. (**H**) Sensing matrix and (**I**) its mapping results for MRI recovery. (**J**) Experimentally recovered pixels and image of MRI. For all the recovery results, the NMSE and PSNR have been calculated and noted.

array. To investigate this issue, Gram MVM circuits with ideal precise programming and nonideal programming with conductance variations (naturally nonidentical two matrices or hypothetically identical) have been considered (see fig. S14). The same verify window of ±5% (or equivalently ±2 μS) as in the experiments is assumed. For each situation, 100 simulations have been conducted with different random input voltage. The results show that the NMSEs of all circuits with inaccurate (identical or nonidentical) device programming are comparable and only slightly larger than the NMSEs of the ideally programmed circuit. The latter is due to the limited conductance range of devices, particularly the absence of efficient mapping of near-zero elements in the assumed RRAM technology.

For LCA circuit, more nonidealities have been considered, including device programming variations, conductance relaxations, parasitic wire resistors in the array, wire resistors in the interfaces between the array and other analog components, and yield of RRAM arrays. The parasitic resistor model is shown in fig. S15, including a row resistor and a column resistor for each crosspoint RRAM device. The correspondence between the wire resistance and CMOS technology node is calculated according to the International Technology Roadmap for Semiconductors 2013 (fig. S15). In addition, the interface wire resistors are assumed as 100 ohms. The simulation results of image recovery from LCA circuit with RRAM programming variations and wire resistors are shown in fig. S16, together with the calculated NMSEs and PSNRs. It turns out that the recovery results at technology nodes of 45 and 32 nm are visually good, featuring low NMSEs. The reconstructed images at 22- and 16-nm nodes show lower qualities, which suggests that the mapping methods need to be optimized to eliminate the effect of wire resistances. The effect of conductance relaxations is studied on the basis of a representative device model (*40*). The relaxation behaviors of the mapped conductance under 85°C are shown in fig. S17. By combining all the nonidealities, LCA circuits are simulated and the results are shown in fig. S17, revealing that the temporal evolution of device variations has a limited impact on the accuracy of image recovery. Last, to explore the impact of array yield, some RRAM devices are randomly set to be stuck-on (40 μS) or stuck-off (1 μS). The PSNR degradation of CS recovery with the LCA circuit is less than 1.7 dB, and the reconstructed images are visually good when yield is ≥99% (fig. S18), which should be easily met in state-of-the-art RRAM fabrication technology (*38*, *41*, *42*).

In Fig. 4, we present the transient behaviors of a dozen of LCA circuit experiments for each CS recovery case, where each time a vector of four output voltages was monitored by the oscilloscope. Although addressing different problems, 1D sparse signal recovery or 2D image reconstructions, the output voltages stabilize within only few microseconds in all cases (Fig. 4, A to C), demonstrating a high speed of the LCA circuit for fast, real-time CS recovery. For each LCA circuit operation, the computing time can be evaluated by tagging the time when the NMSE is smaller than 2.5×10^{-3}. As shown in Fig. 4D, for the three cases, most of the computing times are within 3 to 6 μs. We have also calculated the dynamical NMSE resulting from all 12 vectors (48 curves) in a figure to give an averaged computing time. According to the same criterion, the computing times of Fig. 4A to Fig. 4C are determined to be 5.8, 4.3, and 5.0 μs, respectively, although some among the 48 curves in each figure remain to be fully stable. The results are several tens of times faster than the floating gate transistor–based approach for solving problems with comparable sizes (*11*).

To explore the scaling behavior of computing time of the circuit, a series of simulations have been performed with different matrix sizes, ranging from 8 × 16 to 64 × 128. Given that the computing time of the circuit should be proportional to the gain bandwidth product (GBWP) of the OPAs in use (*23*), an OPA model with a high GBWP has been designed (fig. S19) to maintain the fast response of the circuit. Simulation results show that the convergence time is comparable for different matrix sizes (fig. S20), which suggests a speed improvement by scaling up the size of sensing matrix (also RRAM array) for recovering a given image. To study the convergence time affected by resistance-capacitance (RC) delay, the wire parasitic capacitors were considered on the basis of the model in fig. S21 (*43*). Fifteen different 64 × 128 sensing matrices have been tested, where $C_{\text{wirecol}} = 30\,f\text{F}$ and $C_{\text{wirerow}} = 7.5\,f\text{F}$ (*44*) are assumed for the 257 × 64 RRAM array. The simulation results for comparison are summarized in fig. S22, showing that only a slight delay is induced by the wire parasitic capacitors (6.1 μs becomes 6.3 μs). Another issue of this approach is related to the CC devices in the array. To evaluate the number of extra rows for CC, different sizes of typical sensing matrix $\boldsymbol{\Psi}$ have been studied and the maximum CC of each case is extracted. The calculated results of extra fraction of rows are summarized in fig. S23, showing that the averages of extra rows are all less than 20%. The extra CC rows should not incur a power consumption issue, as the power consumption of this circuit is dominated by OPAs, as will be disclosed later in text S3.

DISCUSSION

To benchmark the performance of the LCA circuit against other approaches, we have performed a comprehensive evaluation of our approach, by including the costs of digital-to-analog converters (DACs), analog-to-digital converters (ADCs), and sample-and-hold circuits (S/H circuits). The architecture schematic is shown in fig. S24, and the corresponding power consumption and latency of each component are summarized in table S2. On the basis of the 64 × 128 sensing matrix and the high-GBWP OPA model, the overall computing time for recovering a 200 × 112 natural image is estimated to be 3.34 ms, which should be more than one order of magnitude faster than the conventional digital approaches (*9*) and the Ising machine (*45*), according to the estimations summarized in text S3 (*46*, *47*) and table S3 (*48*, *49*). Compared to the resistive memory-based iterative MVM approach (*12*, *13*), the LCA circuit solution time is even less than the one of merely MVM operations, excluding the large fraction contributed by the frequent digital-analog conversions and the digital computations therein. It also outperforms the subthreshold floating gate transistor circuit despite the latter making the most favorable assumptions and neglecting the Gram matrix precalculation. Because of the fast response of LCA circuit, its energy dissipation is estimated to be 1.0 mJ (text S3), which also shows a considerable advantage over other approaches. Note that we observed that the output sparsity is around 25%; hence, only $M/4$ S/H circuits and ADCs were assumed activated during each operation. Further energy efficiency improvements would be achievable by multiplexing the use of less ADCs. In addition, the latency and power consumption contributed by each part are calculated (fig. S25), indicating that the DACs and

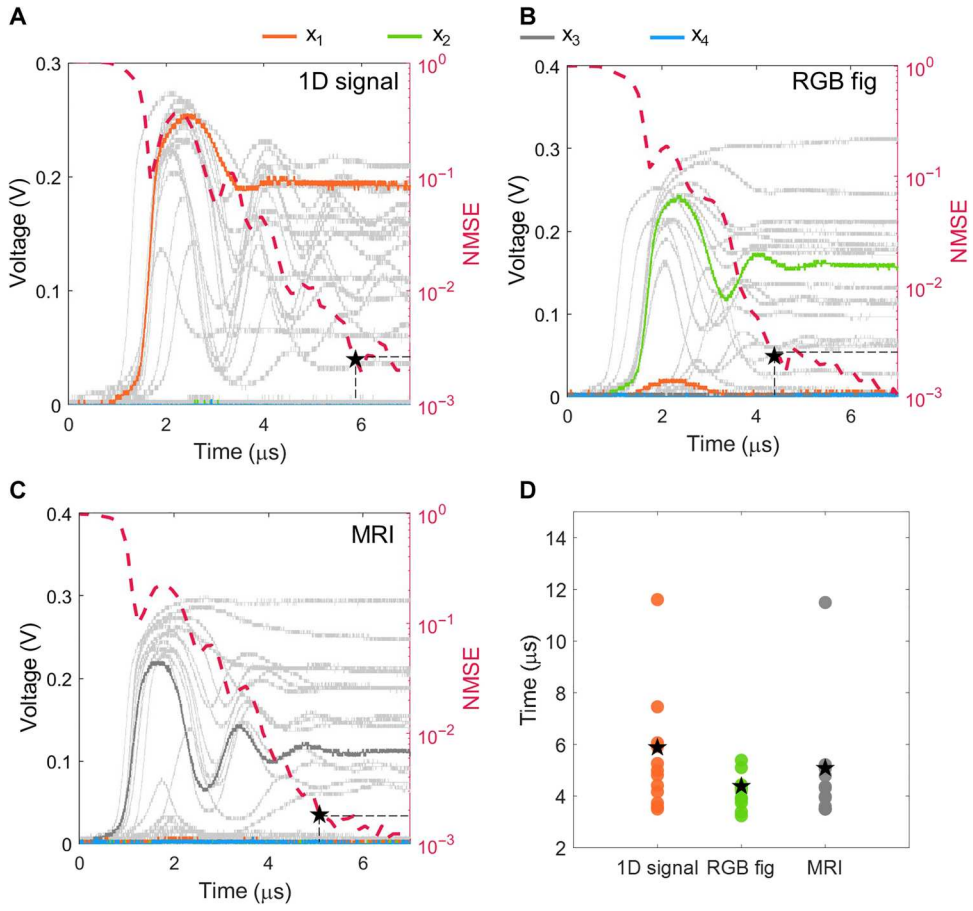

Fig. 4. Transient behaviors and computing times of LCA circuit. Dynamical output voltages in LCA circuit experiments for (**A**) 1D sparse signal recovery, (**B**) 2D natural image recovery, and (**C**) MRI recovery. In each case, 12 sets of 4 × 1 sparse vectors are included, but only one vector is indicated by color lines. The dynamical NMSE evolution based on the combination of 12 LCA circuit operations is also shown in each plot. (**D**) Summary of computing times of LCA circuit for the three cases, including the averaged ones calculated according to the dynamical NMSE curves.

ADCs occupy 68% of the energy dissipation. By optimizing the DACs and ADCs, the energy efficiency should be significantly improved. The high speed and energy efficiency of LCA circuit are attributed to the immense parallelism in the circuit topology and the closed-loop network working in continuous time. Notably, such benefits do not compromise other performance, particularly the accuracy of CS recovery. Since different images with the same size usually show different NMSEs (PSNRs) of CS recovery with the ideal algorithm per se, we compared the PSNR degradation of the hardware solution with reference to the algorithmic result. It turns out that the continuous-time circuit shows a lower decibel loss than the discrete iterative MVM approach, in large part due to the elimination of repeated analog-digital conversions that always introduce additional errors.

In conclusion, inspired by the LCA, we have developed an in-memory AMC circuit for solving CS recovery problem in one step. It is only possible with the efficient design of a crosspoint resistive memory array–based module for in-memory Gram MVM operation, which in turn is supported by the CC principle. While neural networks have been widely targeted by in-memory AMC in pursuit of tremendous computing performance improvements, CS recovery (or generally sparse approximation) should also benefit from such a radical paradigm shift, given that both problems work with nonlinear functions that could depress unwanted signals in the solution and thus protect the computing accuracy. Our results show that with the LCA circuit, the non-smooth optimization problem of CS recovery can be reliably solved within only few microseconds while maintaining a reasonably small accuracy loss compared to the full-precision digital solution. In addition, the circuit shows an outstanding scalability toward large-scale problem solving, where the accuracy degradation would be mitigated. Therefore, we believe that modern in-memory AMC is highly promising for the back-end CS processor implementation that delivers real-time processing capability in the microsecond regime, which might enable advanced medical, visual, and communication techniques.

MATERIALS AND METHODS

Device fabrication

The RRAM devices were fabricated on SiO_2/Si substrate. First, the bottom electrode composed of 5-nm Ti adhesion layer and 30-nm Pt was deposited using magnetron sputtering and patterned by photolithography and lift-off process. The dielectric layer is 5-nm HfO_2,

which was prepared by atomic layer deposition. The top electrode consisting of 40-nm Ta and 30-nm Pt was finally deposited and patterned.

Experimental measurements

The switching characteristics, *I-V* linearity, endurance, and retention time of RRAM devices were collected with the Keysight B1500A Semiconductor Parameter Analyzer. In the experiment, the LCA circuit was formed by simply mounting the packaged RRAM array on a custom PCB. During the image recovery experiments, the supply and threshold voltages were powered by the Tabor Electronics Model WW5064 Four Channel Waveform Generator, and the tens of thousands voltage pulses with 50-ms width were generated serially by the Keysight 4200A Semiconductor Parameter Analyzer. The write-verify method for programming RRAM into different conductance levels was also implemented using the Keysight 4200A Semiconductor Parameter Analyzer. An Arduino Mega 2560 was used to measure the output voltages in the Gram MVM and CS recovery experiments. In the transient behavior experiment, we used the RIGOL MSO8104 Four Channel Digital Oscilloscope to capture the signals. In addition, four pairs of diodes were used in LCA circuit to limit the output voltage below 0.3 V, which should protect the programmed RRAM devices from conductance change during tests.

Sparsity basis training

First, elements of sparsity basis $\boldsymbol{A}$ were initialized to random values. Then, we ran LCA to sparsely represent the original images, in the context of sparse coding. The stochastic gradient descent update rule was applied in the training process

$$\Delta \boldsymbol{A}^T = \beta(\boldsymbol{p} - \boldsymbol{A}^T\boldsymbol{x}) \otimes \boldsymbol{x}$$

where β is the learning rate, $\boldsymbol{p}$ is the original pixel of image, $\boldsymbol{x}$ is the sparse representation during the training process, and ⊗ is the outer product. In our case, $\beta = 5 \times 10^{-4}$ was chosen, and 15 iterations were sufficient to reach the convergence.

Simulations

The simulations of large-scale LCA circuit in the Supplementary Materials were performed in Simulation Program with Integrated Circuit Emphasis (SPICE). We considered a general RRAM device conductance range of 1 to 40 μS, and the verify window of ±2 μS was preserved. The AD823 and AD8572 models from Analog Devices Inc. were used for OPAs in SPICE simulations.

Supplementary Materials

This PDF file includes:

Supplementary Text
Figs. S1 to S25
Tables S1 to S3
References

REFERENCES AND NOTES

1. E. T. Candès, M. B. Wakin, An introduction to compressive sampling. *IEEE Signal Process. Mag.* **25**, 21–30 (2008).
2. M. Lustig, D. L. Donoho, J. M. Santos, J. M. Pauly, Compressed sensing MRI. *IEEE Signal Process. Mag.* **25**, 72–82 (2008).
3. W. U. Bajwa, J. Haupt, A. M. Sayeed, R. Nowak, Compressed channel sensing: A new approach to estimating sparse multipath channels. *Proc. IEEE* **98**, 1058–1076 (2010).
4. H. Li, C. Shen, Q. Shi, Real-time visual tracking using compressive sensing, in *Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2011), pp. 1305–1312.
5. M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, R. G. Baraniuk, Single-pixel imaging via compressive sampling. *IEEE Signal Process. Mag.* **25**, 83–91 (2008).
6. V. Antun, F. Renna, C. Poon, B. Adcock, A. C. Hansen, On instabilities of deep learning in image reconstruction and the potential costs of AI. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 30088–30095 (2020).
7. H. Gu, B. Yaman, S. Moeller, J. Ellermann, K. Ugurbil, M. Akçakaya, Revisiting ℓ1-wavelet compressed-sensing MRI in the era of deep learning. *Proc. Natl. Acad. Sci. U.S.A.* **119**, e2201062119 (2022).
8. H. Rabah, A. Amira, B. K. Mohanty, S. Almaadeed, P. K. Meher, FPGA implementation of orthogonal matching pursuit for compressive sensing reconstruction. *IEEE Trans. Very Large Scale Integr. VLSI Syst.* **23**, 2209–2220 (2015).
9. A. Kulkarni, T. Mohsenin, Accelerating compressive sensing reconstruction OMP algorithm with CPU, GPU, FPGA and domain specific many-core, in *2015 IEEE International Symposium on Circuits and Systems (ISCAS)* (IEEE, 2015).
10. Y. Tsividis, Not your father's analog computer. *IEEE Spectr.* **55**, 38–43 (2018).
11. S. Shapero, A. S. Charles, C. J. Rozell, P. Hasler, Low power sparse approximation on reconfigurable analog hardware. *IEEE J. Emerg.* **2**, 530–541 (2012).
12. M. Le Gallo, A. Sebastian, G. Cherubini, H. Giefers, E. Eleftheriou, Compressed sensing recovery using computational memory, in *International Electron Devices Meeting (IEDM)* (IEEE, 2017).
13. P. M. Sheridan, F. Cai, C. Du, W. Ma, Z. Zhang, W. D. Lu, Sparse coding with memristor networks. *Nat. Nanotechnol.* **12**, 784–789 (2017).
14. Z. Sun, G. Pedretti, E. Ambrosi, A. Bricalli, W. Wang, D. Ielmini, Solving matrix equations in one step with cross-point resistive arrays. *Proc. Natl. Acad. Sci. U.S.A.* **116**, 4123–4128 (2019).
15. Z. Sun, G. Pedretti, A. Bricalli, D. Ielmini, One-step regression and classification with cross-point resistive memory arrays. *Sci. Adv.* **6**, eaay2378 (2020).
16. Z. Sun, D. Ielmini, Invited tutorial: Analog matrix computing with crosspoint resistive memory arrays. *IEEE Trans. Circuits Syst. II Express Briefs* **69**, 3024–3029 (2022).
17. P. Zuo, Z. Sun, R. Huang, Extremely-fast, energy-efficient massive MIMO precoding with analog RRAM matrix computing. arXiv:2211.03624 [eess.SP] (7 November 2022).
18. Y. Huang, N. Guo, M. Seok, Y. Tsividis, S. Sethumadhavan, Evaluation of an analog accelerator for linear algebra. *ACM SIGARCH Comput. Archit. News* **44**, 570–582 (2016).
19. D. B. Strukov, G. S. Snider, D. R. Stewart, R. S. Williams, The missing memristor found. *Nature* **453**, 80–83 (2008).
20. P. Mannocci, M. Farronato, N. Lepri, L. Cattaneo, A. Glukhov, Z. Sun, D. Ielmini, In-memory computing with emerging memory devices: Status and outlook. *APL Mach. Learn.* **1**, 010902 (2023).
21. Z. Wang, H. Wu, G. W. Burr, C. S. Hwang, K. L. Wang, Q. Xia, J. J. Yang, Resistive switching materials for information processing. *Nat. Rev. Mater.* **5**, 173–195 (2020).
22. G. W. Burr, A. Sebastian, T. Ando, W. Haensch, Ohm's law + Kirchhoff's current law = better AI: Neural-network processing done in memory with analog circuits will save energy. *IEEE Spectr.* **58**, 44–49 (2021).
23. Z. Sun, G. Pedretti, P. Mannocci, E. Ambrosi, A. Bricalli, D. Ielmini, Time complexity of in-memory solution of linear systems. *IEEE Trans. Electron Devices* **67**, 2945–2951 (2020).
24. L. E. Aygun, P. Kumar, Z. Zheng, T. S. Chen, S. Wagner, J. C. Sturm, N. Verma, Hybrid system for efficient LAE-CMOS interfacing in large-scale tactile-sensing skins via TFT-based compressed sensing, in *IEEE International Solid-State Circuits Conference (ISSCC)* (IEEE, 2019), pp. 280–282.
25. P. V. Rajesh, J. M. Valero-Sarmiento, L. Yan, A. Bozkurt, C. V. Hoof, N. V. Helleputte, R. F. Yazicioglu, M. Verhelst, A 172μW compressive sampling photoplethysmographic readout with embedded direct heart-rate and variability extraction from compressively sampled data, in *IEEE International Solid-State Circuits Conference (ISSCC)* (IEEE, 2016), pp. 487–496.
26. Y, Oike, A. El Gamal, A 256×256 CMOS image sensor with ΔΣ-based single-shot compressed sensing, in *IEEE International Solid-State Circuits Conference (ISSCC)* (IEEE, 2012).
27. B. K. Natarajan, Sparse approximate solutions to linear systems. *SIAM J. Comput.* **24**, 227–234 (1995).
28. D. Donoho, M. Elad, Optimally sparse representation in general (nonorthogonal) dictionaries via ℓ^1 minimization. *Proc. Natl. Acad. Sci. U.S.A.* **100**, 2197–2202 (2003).
29. S. J. Kim, K. Koh, M. Lustig, S. Boyd, D. Gorinevsky, An interior-point method for large-scale ℓ_1-regularized least squares. *IEEE J. Sel. Top. Signal Process* **1**, 606–617 (2007).
30. M. A. Figueiredo, R. D. Nowak, S. J. Wright, Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Sel. Top. Signal Process* **1**, 586–597 (2007).

31. I. Daubechies, M. Defrise, C. De Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Commun. Pure Appl. Math.* **57**, 1413–1457 (2004).
32. Y. C. Pati, R. Rezaiifar, P. S. Krishnaprasad, Orthogonal matching pursuit: Recursive function approximation with application to wavelet decomposition, in *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers* (IEEE, 1993).
33. D. L. Donoho, A. Maleki, A. Montanari, Message-passing algorithms for compressed sensing. *Proc. Natl. Acad. Sci. U.S.A.* **106**, 18914–18919 (2009).
34. C. J. Rozell, D. H. Johnson, R. G. Baraniuk, B. A. Olshausen, Sparse coding via thresholding and local competition in neural circuits. *Neural Comput.* **20**, 2526–2563 (2008).
35. K. Gregor, Y. LeCun, Learning fast approximations of sparse coding, in *Proceedings of 27th International Conference on International Conference on Machine Learning* (Association for Computing Machinery, 2010), pp. 399–406.
36. P. Mannocci, A. Baroni, E. Melacarne, C. Zambelli, P. Olivo, E. Pérez, C. Wenger, D. Ielmini, In-memory principal component analysis by crosspoint array of resistive switching memory: A new hardware approach for energy-efficient data analysis in edge computing. *IEEE Nanotechnol. Mag.* **16**, 4–13 (2022).
37. S. Brivio, S. Spiga, D. Ielmini, HfO_2-based resistive switching memory devices for neuromorphic computing. *Neuromorph. Comput. Eng.* **2**, 042001 (2022).
38. P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, H. Qian, Fully hardware-implemented memristor convolutional neural network. *Nature* **577**, 641–646 (2020).
39. W. Wan, R. Kubendran, C. Schaefer, S. B. Eryilmaz, W. Zhang, D. Wu, S. Deiss, P. Raina, H. Qian, B. Gao, S. Joshi, H. Wu, H. S. P. Wong, G. Cauwenberghs, A compute-in-memory chip based on resistive random-access memory. *Nature* **608**, 504–512 (2022).
40. M. Zhao, H. Wu, B. Gao, Q. Zhang, W. Wu, S. Wang, Y. Xi, D. Wu, N. Deng, S. Yu, H. Chen, H. Qian, Investigation of statistical retention of filamentary analog RRAM for neuromorphic computing, in *IEEE International Electron Devices Meeting (IEDM)* (IEEE, 2017).
41. S. Yu, Z. Li, P. Y. Chen, H. Wu, B. Gao, D. Wang, W. Wu and H. Qian, Binary neural network with 16 Mb RRAM macro chip for classification and online training, *in IEEE International Electron Devices Meeting (IEDM)* (IEEE, 2016), pp. 16.2.1–16.2.4.
42. W. Zhang, P. Yao, B. Gao, Q. Liu, D. Wu, Q. Zhang, Y. Li, Q. Qin, J. Li, Z. Zhu, Y. Cai, D. Wu, J. Tang, H. Qian, Y. Wang, H. Wu, Edge learning using a fully integrated neuro-inspired memristor chip. *Science* **381**, 1205–1211 (2023).
43. Y. Zhou, P. Huang, L. Cai, Y. Feng, L. Liu, X. Liu, J. Kang, Optimized programming scheme enabling symmetric conductance modulation in HfO_2 resistive random-access memory (RRAM) for neuromorphic systems. *IEEE Electron Device Lett.* **43**, 1203–1206 (2022).
44. Y. Feng, P. Huang, Y. Zhang, W. Shen, W. Xu, Y. Xiang, X. Ding, Y. Zhao, B. Gao, H. Wu, H. Qian, L. Liu, X. Liu, J. Kang, A self-terminated operation scheme for high-parallel and energy-effcient forming of RRAM array. *Adv. Electron. Mater.* **6**, 1–6 (2020).
45. M. D. S. H. Gunathilaka, S. Kako, Y. Inui, K. Mimura, M. Okada, Y. Yamamoto, T. Aonishi, Effective implementation of l_0-regularised compressed sensing with chaotic-amplitude-controlled coherent Ising machines. arXiv:2302.12523 [quant-ph] (24 February 2023).
46. B. Feinberg, R. Wong, T. P. Xiao, C. H. Bennett, J. N. Rohan, E. G. Boman, M. J. Marinella, S. Agarwal, E. Ipek, An analog preconditioner for solving linear systems, *in IEEE International Symposium on High-Performance Computer Architecture (HPCA)* (IEEE, 2021).
47. L. Kull, T. Toifl, M. Schmatz, P. A. Francese, C. Menolfi, M. Brandli, M. Kossel, T. Morf, T. M. Andersen, Y. Leblebici, A 3.1 mW 8b 1.2 GS/s single-channel asynchronous SAR ADC with alternate comparators for enhanced speed in 32 nm digital SOI CMOS. *IEEE J. Solid-State Circuits* **48**, 3049–3058 (2013).
48. L. Bai, P. Maechler, M. Muehlberghuber, H. Kaeslin. High-speed compressed sensing ee-construction on FPGA using OMP and AMP, *in 2012 19th IEEE International Conference on Electronics, Circuits and Systems (ICECS)* (IEEE, 2012).
49. T. S. Chen, H. C. Kuo, A. Y. Wu, A 232-to-1996KS/s robust compressive-sensing reconstruction engine for real-time physiological signals monitoring, *in IEEE International Solid-State Circuits Conference (ISSCC)* (IEEE, 2018).

Acknowledgments: We thank W. Sun (Tsinghua University) for help with the wire boding. **Funding:** This work has received funding from the National Key R&D Program of China (2020YFB2206001), the National Natural Science Foundation of China (62004002, 92064004, and 61927901), and the 111 Project (B18001). **Author contributions:** Z.S. conceived the idea of solving CS recovery in one step with an in-memory AMC circuit. S.W. and Z.S. originated the Gram MVM and LCA circuits. Y.Luo, P.Z., L.P., and Y. Li helped improve the circuits. S.W. and Z.S. designed the experiments. S.W. fabricated and characterized the devices, designed the printed circuit boards, conducted the experiments, analyzed the circuits in theory, and simulated the large-scale circuits. S.W. and Z.S. wrote the manuscript with input from all authors. Z.S. supervised the research. **Competing interests:** The authors declare that they have no competing interests. **Data and materials availability:** All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Materials.

Submitted 17 June 2023
Accepted 13 November 2023
Published 13 December 2023
10.1126/sciadv.adj2908