In-memory principal component analysis by analogue closed-loop eigendecomposition

Piergiulio Mannocci, *Member, IEEE,* Elisabetta Giannone, and Daniele Ielmini, *Fellow, IEEE*

***Abstract*—Machine learning (ML) techniques such as principal component analysis (PCA) have become pivotal in enabling efficient processing of big data in an increasing number of applications. However, the data-intensive computation in PCA causes large energy consumption in conventional von Neumann computers. In-memory computing (IMC) significantly improves throughput and energy efficiency by eliminating the physical separation between memory and processing units. Here, we present a novel closed-loop IMC circuit to compute real eigenvalues and eigenvectors of a target matrix allowing IMC-based acceleration of PCA. We benchmark its performance against a commercial GPU, achieving comparable accuracy and throughput while simultaneously securing $\times 10^4$ energy and $\times 10^{2\div 4}$ area efficiency improvements. These results support IMC as a leading candidate architecture for energy-efficient ML accelerators.**

***Index Terms*—In-memory computing, eigendecomposition, principal component analysis, machine learning, analog computing.**

I. Introduction

Among the machine learning (ML) algorithms for statistical data analysis, principal component analysis (PCA) [1] finds widespread application to improve the efficacy of classifiers. Particularly, PCA is a key algorithm in dimensionality reduction for datasets with highly correlated variables, enabling the identification of prominent features as opposed to non-sparsity inducing methods such as linear regression [2]. However, PCA heavily relies on data-intensive matrix operations [2] which makes it unsuited for conventional digital processors, where the physical separation between memory and computing units causes severe overhead in terms of energy and latency.

In-memory computing (IMC) has recently emerged as a promising computing paradigm to suppress data movement with considerable throughput and energy improvements, further enhanced by its inherent compatibility with highly-scalable crosspoint arrays of resistive memories [3]. IMC has been shown capable of accelerating matrix-vector multiplication (MVM) [4] in open-loop circuits exploiting Ohm's and Kirchhoff's laws to perform *in-situ* multiplication and accumulation. In-memory MVM-based acceleration of PCA was demonstrated using both neural network strategies [5] or covariance matrix-decomposition techniques [6], which however incur significant processing overheads for either weight update or iterated-MVM process, limiting their efficacy.

P. Mannocci, E. Giannone and D. Ielmini are with Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano and IU.NET, Piazza L. da Vinci 32, 20133, Milano, Italy.

This project has received funding from the European Research Council (ERC) under the European Union's Horizon Europe research and innovation programme (grant agreement n° 101054098).

Correspondence should be addressed to piergiulio.mannocci@polimi.it and daniele.ielmini@polimi.it.

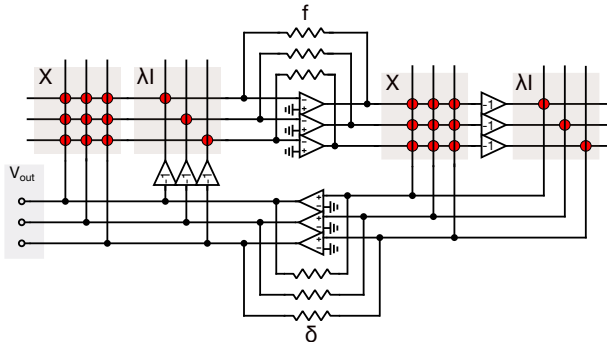

Fig. 1. Eigendecomposition circuit, consisting of four memory arrays mapping replicas of the target matrix $\mathbf{X}$ and target eigenvalue λ, two TIA sets with feedback conductances f, δ, and two sets of inverting buffers. The computed eigenvector appears on the δ-feeback TIAs outputs $\mathbf{v_{out}}$.

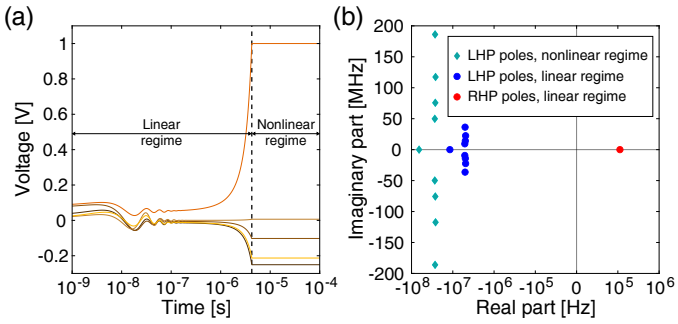

Fig. 2. (a) Example of output transient for a 5×5 random matrix $\mathbf{X}$. After a linear transient with divergent behavior, saturation of the larger output leads into a nonlinear regime where the remaining outputs stabilize to a steady-state value. The dashed line denotes the saturation time ($V_{sat} = 1\,\mathrm{V}$), corresponding to regime crossover. (b) Circuit poles in linear (circles) and nonlinear (diamonds) regimes. Before saturation, a single pole lies in the RHP, driving the divergent response. In the nonlinear regime, all poles lie in the stable LHP.

Crosspoint array-based, closed-loop feedback circuits [7] further allow acceleration of inverse problems such as inverse-matrix-vector multiplication (IMVM), linear, generalized [8], and regularized [9] regression. Positive-feedback-based, closed-loop IMC circuits [10] have been shown capable of computing eigenvectors, although they are currently limited to extraction of leading and trailing eigenvectors only.

Here, we present a novel closed-loop IMC circuit for matrix eigendecomposition, namely extraction of the full set of eigenvectors and eigenvalues. We study the circuit performance in terms of accuracy and speed, highlighting the design trade-offs and deriving tuning rules for the circuit parameters. We

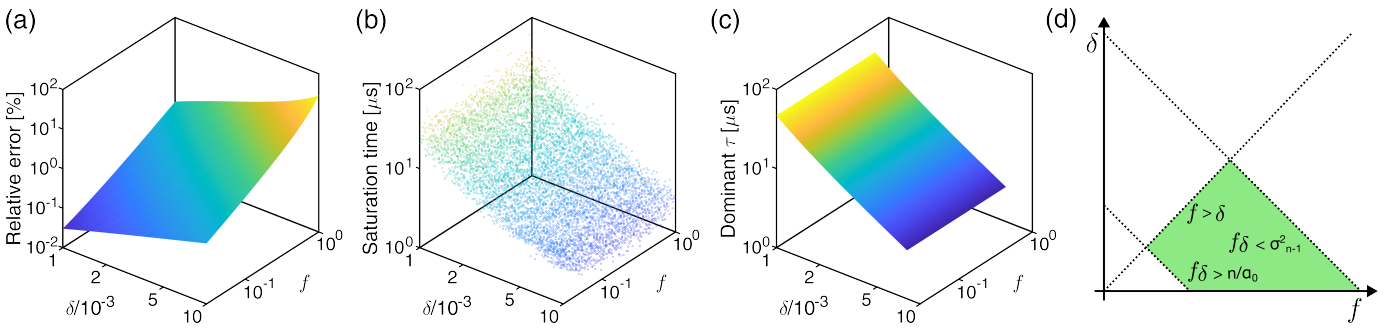

Fig. 3. (a) Relative error of the measured outputs with respect to the expected eigenvector, showing a bilinear dependence on f and δ. (b) Saturation time for different values of f and δ, showing a linear dependence on δ only. (c) Time constant of the positive pole, showing a similar linear dependence on δ and mostly independent on f. (d) Design space for the circuit parameters. Highlighted equations must be satisfied to locate only one pole in the RHP.

benchmark its performance on the PCA of large datasets and compare it with a commercial graphic processing unit (GPU). Simulation results show performance improvements for all considered metrics while guaranteeing the same accuracy of a 64-bit floating-point (FP64) precision computer.

II. In-memory eigenvector circuit

Fig. 1 shows the eigenvector IMC circuit, consisting of four crosspoint memory arrays, storing the data matrix $\mathbf{X}$ and the eigenvalue λ whose eigenvector is to be computed. Two sets of transimpedance amplifiers (TIAs) with feedback conductances f and δ in inverting and non-inverting configuration respectively, and two sets of inverting buffers with -1 gain are used to close the feedback loops. Thanks to the double array structure [8], the circuit is capable of computing all eigenvectors of a matrix, thus representing an improvement of previous in-memory topologies [10], which were limited to the computation of the leading and trailing eigenvectors only. Assuming an infinite DC gain of the operational amplifiers (OAs), the nodal equation at the output $\mathbf{v_{out}}$ yields:

$$\mathbf{v_{out}} = \delta^{-1}(\mathbf{X} - \lambda\mathbf{I})^T f^{-1}(\mathbf{X} - \lambda\mathbf{I})\mathbf{v_{out}}, \tag{1}$$

which can be rewritten as:

$$((\mathbf{X} - \lambda\mathbf{I})^T(\mathbf{X} - \lambda\mathbf{I}) - f\delta\mathbf{I})\mathbf{v_{out}} = \mathbf{0}. \tag{2}$$

When $f\delta$ is sufficiently small, *i.e.* $f\delta \simeq 0$, Eq. (2) reduces to:

$$(\mathbf{X} - \lambda\mathbf{I})\mathbf{v_{out}} = \mathbf{0}, \tag{3}$$

which is equivalent to the secular equation $\mathbf{Xv} = \lambda\mathbf{v}$ for matrix $\mathbf{X}$. The output voltages $\mathbf{v_{out}}$ thus correspond to the eigenvector $\mathbf{v}$ associated to eigenvalue λ for matrix $\mathbf{X}$.

Fig. 2a shows a SPICE simulation of a typical circuit transient for a random 5×5 matrix. The circuit is operated by first pre-charging the outputs to random voltages, *e.g.* by randomly injecting currents on virtual ground nodes, after which the circuit is left free to evolve. When f and δ are properly tuned, the circuit poles are divided into the stable left-half plane (LHP) and the unstable right-half plane (RHP) as shown in the root locus in Fig. 2b. The presence of an RHP pole results in at least one OA being in positive feedback and thus to divergent behavior of the output voltages $\mathbf{v_{out}}$ in the initial *linear regime*. As soon as one OA output reaches the supply voltage, it enters a saturation regime causing a loss of feedback action, *i.e.* the saturated OA now operates as a constant-voltage source, and the corresponding RHP pole vanishes. Unsaturated OAs still work in a feedback configuration, their poles being all located in the LHP (Fig. 2b, diamonds). In this *nonlinear regime*, the circuit thus operates in negative feedback, allowing for output voltages $\mathbf{v_{out}}$ to stabilize at steady-state values matching the matrix eigenvector according to Eq. (3).

Conductances f and δ impact both the overall circuit speed and accuracy by controlling (i) the position of circuit poles and (b) the magnitude of the $f\delta\mathbf{I}$ term in Eq. (2), which represents a perturbation of the target secular equation given in Eq. (3). Figs. 3a-b show the relative error and saturation time as a function of conductances f and δ, highlighting a speed-accuracy tradeoff. Particularly, for small values of δ, the circuit is more precise but suffers from increased saturation time owing to the smaller displacement of the positive pole in the RHP, which is directly controlled by δ and represents the dominant term in defining the circuit solution time. The latter can be modeled by a weighted sum of exponential terms, namely [9]:

$$\mathbf{v_{out}}(t) = \sum_{i=1}^{n} \beta_{i,0}\mathbf{v}_i e^{\frac{t}{\tau_i}}, \tag{4}$$

where n is the size of matrix $\mathbf{X}$, $\mathbf{v}_i$ is the i-th matrix eigenvector, $\beta_{i,0}$ is the initial amplitude of the i-th component as a result of the random precharge. The term e^{t/τ_i} in Eq. (4) models the exponential transient of the i-th component which vanishes if τ_i lies in the LHP, thus leaving only the RHP-related component $\mathbf{v}$ with a positive time constant τ_+. By evaluating the time-dependent ℓ_∞-norm of the output vector:

$$\|\mathbf{v_{out}}\|_\infty(t) \simeq \beta_{+,0}\|\mathbf{v}\|_\infty e^{\frac{t}{\tau_+}}, \tag{5}$$

imposing $\|\mathbf{v_{out}}\|_\infty(t_{sat}) = V_{sat}$ yields the saturation time:

$$t_{sat} = \tau_+ \log\left(V_{sat}/(\beta_{+,0}\|\mathbf{v}\|_\infty)\right), \tag{6}$$

showing a weak logarithmic dependence on the initial precharge process through $\beta_{+,0}$ and a strong linear dependence on the RHP-pole time constant τ_+. Fig. 3c shows τ_+ as a function of f and δ, highlighting the correlation with the saturation time. As δ increases, Eq.(3) no longer well-approximates Eq. (2), resulting in increased error with respect to the target eigenvector, but also in reduced saturation time

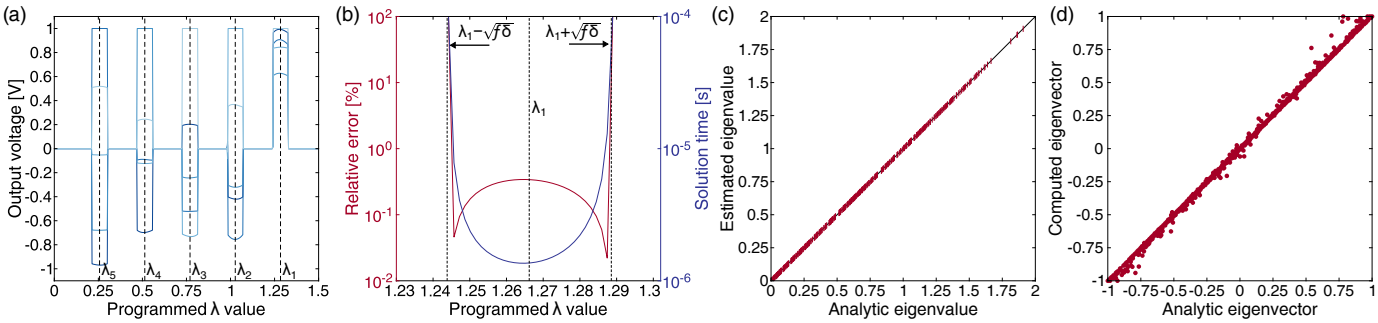

Fig. 4. Sweep-based circuit operation. (a) Output voltages, extracted after 100 µs, for different equivalent values of the eigenvalue conductance λ. Dashed lines mark the location of the target eigenvalues of the considered matrix, around which the circuit exhibits a non-zero response. (b) Relative error and solution time inside the $\pm\sqrt{f\delta}$ activity window for λ_1 in (a), highlighting a speed-accuracy trade-off. (c,d) Estimated eigenvalues and eigenvectors as a function of analytic eigenvalues and eigenvectors for 100 eigendecomposition simulations on 5×5, symmetric positive-definite matrices.

and thus faster convergence. Fig. 3d illustrates the overall design space of variables f and δ, which can be summarized in three conditions. First, condition (i) holds for $f > \delta$, which ensures a faster decay of LHP components with respect to the growth of the RHP component in Eq. (5). Condition (ii) $f\delta < \sigma_{n-1}$, where σ_{n-1} is the smallest nonzero singular value of $(\mathbf{X} - \lambda\mathbf{I})$, ensures that at most one pole is located in the RHP, thus preventing the simultaneous recall of multiple eigenvectors. Finally, condition (iii) $f\delta > n/\alpha_0$ sets a lower boundary to overcome finite gain non-ideality.

By iteratively reprogramming the eigenvalue conductance λ, the full set of eigenvectors of $\mathbf{X}$ can be extracted. Fig. 4a shows the circuit output extracted after 100 µs for a random 5×5 positive definite matrix $\mathbf{X}$ and different values of conductance λ, denoting a nonzero output only around $\lambda \simeq \lambda_{1\ldots5}$, corresponding to the eigenvalues of matrix $\mathbf{X}$. Each activity window extends symmetrically around the target eigenvalues by $\pm\sqrt{f\delta}$, which thus defines the minimum resolution for eigendecomposition. Fig. 4b highlights the speed-accuracy tradeoff, with speed and accuracy maximized at the center and extrema respectively. Figs. 4c-d shows simulation results of eigenvalue and eigenvector computation for 100 5×5 symmetric matrices $\mathbf{X}$, for 80 dB-gain OAs, $f = 0.05$, and $\delta = 0.01$, corresponding to a resolution $\sqrt{f\delta} \simeq 0.02$, demonstrating almost perfect correlation with analytical results for both computation targets.

III. In-memory principal component analysis

Computation of eigenvalues and eigenvectors is crucial in PCA, whose aim is the estimation of principal components (PCs), namely the vector directions that maximize the dataset variance [1]. The i-th PC is defined as the i-th eigenvector of the dataset covariance matrix $\mathbf{C}$, namely:

$$\mathbf{C} = m^{-1}\mathbf{D}^T\mathbf{D}, \tag{7}$$

where $\mathbf{D} \in \mathbb{R}^{m\times n}$ is the standardized dataset matrix and m is the number of observations, corresponding to rows of $\mathbf{D}$. Dimensionality reduction is attained by retaining only those PCs for which $\lambda > 1$ [11]. The obtained PC subset is used to form a new reference basis $\mathbf{P}$, obtaining the projected dataset $\mathbf{Y} = \mathbf{DP}$, thus allowing extraction of relevant features, clustering and classification [1].

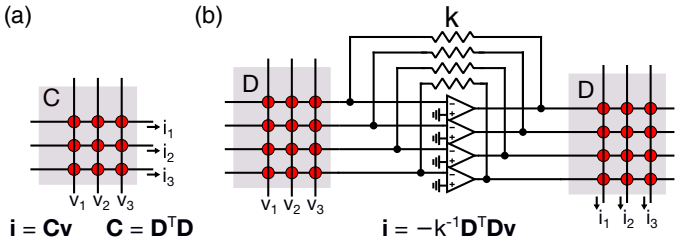

Fig. 5. (a) Covariance matrix $\mathbf{C}$ can be computed offline and mapped into a single memory array, which then corresponds to the computation target matrix $\mathbf{X}$ in the circuit in Fig. 1. (b) Alternatively, the covariance matrix can be computed online using two memory arrays mapping the standardized dataset $\mathbf{D}$ and a set of TIAs with feedback conductance k. The resulting voltage-to-current transfer between voltages $\mathbf{v}$ applied to the left array and currents $\mathbf{i}$ probed at the right array matches Eq. (7) analogously to (a).

Commonly used PCA algorithms generally rely on iterative methods [2], [12] rather than on straightforward eigendecomposition of $\mathbf{C}$, owing to the high computational and memory overhead incurred by the explicit calculation of the covariance matrix. In our IMC topology, the equivalent transfer function of memory array $\mathbf{C}$ in Fig. 5a can be achieved by the circuit of Fig. 5b, where two memory arrays are used to store the standardized dataset $\mathbf{D}$, while a set of TIAs with feedback conductance k allows the in-situ matrix-matrix multiplication $\mathbf{D}^T\mathbf{D}$. The transfer function between the input voltages $\mathbf{v}$ applied to the columns of the left array, and the output currents $\mathbf{i}$ probed at the grounded columns of the right array reads:

$$\mathbf{i_{out}} = -k^{-1}\mathbf{D}^T\mathbf{D}\mathbf{v_{in}} = -mk^{-1}\mathbf{C}\mathbf{v_{in}}, \tag{8}$$

which provides the covariance matrix in Eq. (7). By replacing arrays $\mathbf{X}$ in Fig. 1 with the equivalent block in Fig. 5b, eigendecomposition of $\mathbf{C}$ can be performed following the sweep-based procedure of Sec. II without explicitly computing and storing the covariance matrix $\mathbf{C}$.

To assess the PCA acceleration capability of our circuit, we considered the Wine dataset, a collection of 6497 observations of 11 chemical constituents for *red* and *white* variants of the Portuguese Vinho Verde wine [13]. As one of the main limitations of IMC is the reduced precision of memory elements with respect to FP64 digital computers, we evaluated the impact of memory precision on the PC computation accuracy by measuring the mean absolute cosine similarity between the

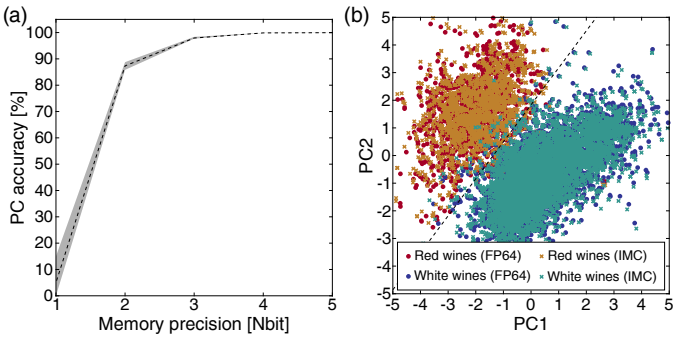

Fig. 6. (a) PC computation accuracy for the Wine dataset as a function of the memory cell precision. The dashed line shows the mean absolute cosine similarity of the computed PCs. Conversely, shading extends between minimum and maximum accuracy for each precision value. (b) Wine dataset projections on the PC1-PC2 subspace, for FP64 PCA (red, blue) and 4-bit IMC PCA (brown, teal), allowing identification of *red* and *white* wine clusters. Dashed lines highlight projections of the classification hyperplane.

IMC-computed and FP64-computed PCs, namely:

$$|\text{cosim}(\mathbf{x},\mathbf{y})| = |\mathbf{x}^T\mathbf{y}|/(\|\mathbf{x}\|_2\|\mathbf{y}\|_2). \tag{9}$$

Fig. 6a shows the resulting mean PC computation accuracy as a function of the memory precision highlighting a requirement of at least 4-bit memory cells to provide acceptable similarity above 99%. Fig. 6b shows the corresponding dataset projection onto the PC1-PC2 subspace for an FP64 computer and the IMC circuit with 4-bit memory cells. Logistic regression was used to evaluate the classification accuracy using a 500-sample training subset, yielding comparable 98.32% and 98.08% accuracies for FP64 and IMC respectively.

IV. Benchmark Simulations

For benchmark purposes, we considered the IMC circuit to be implemented in its block-form shown in Fig. 7 [9], requiring a single $(2n+2m)\times(2n+2m)$ memory array. Since the block-form mapping is equivalent to recasting the original problem as a single-matrix inversion, blockwise inversion [19] and matrix partitioning algorithms [4], [14] can be used to alleviate size-dependent effects associated with large memory arrays, such as IR drop [20], by dividing the computation in smaller-size closed-loop cores. To evaluate the circuit performance, we performed SPICE simulations considering a $14\,\text{nm}$ technological node. For the front-end section, we considered $500\,\text{MHz}$, $80\,\text{dB}$ gain OAs with $0.8\,\text{V}$ supply voltage and $12\,\mu\text{W}$ leakage power [14]. OAs corresponding to the eigenvector output $\mathbf{v_{out}}$ in Fig. 7 were considered to be limited between $0.3\,\text{V}$ and $0.5\,\text{V}$. For output readout, we considered 10-bit ADCs with $0.9\,\text{V}$ supply voltage and $5.5\,\text{pJ}$ conversion energy [15]. Two different memory cell implementations were benchmarked by exploiting the memory-agnosticism of the proposed circuit, *i.e.* correct operation can be achieved with any memory exhibiting ohmic behavior between a terminal pair. We considered memory arrays in either a 1R-RRAM technology (Fig. 8a, $10\,\text{k}\Omega$ LRS, $4F^2$ cell area, $5\,\text{ns}$ write time, $0.5\,\text{pJ}$ programming energy [16], 10% programming variability [17]) or a 4T4R-CMOS technology (Fig. 8b, $5\,\text{k}\Omega$ LRS, $10\,\mu\text{m}^2$ cell area, $33\,\text{ps}$ write time, $4\,\text{fJ}$ programming energy [18], negligible variability).

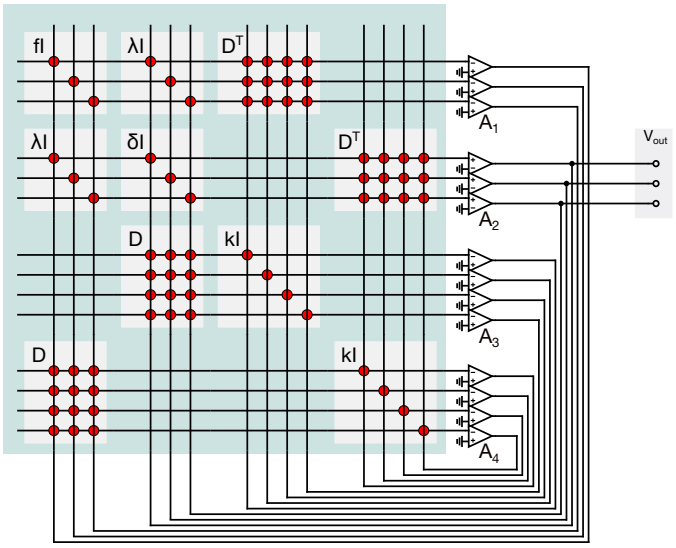

Fig. 7. Block-mapping of the eigendecomposition circuit in Fig. 1 when the covariance block of Fig. 5b is used in place of matrix $\mathbf{X}$. The block-circuit is fully equivalent to the original circuit from an electrical standpoint, while at the same time reducing implementation complexity.

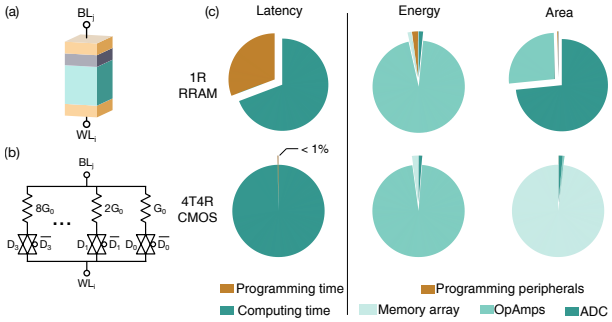

Fig. 8. (a) 1R-RRAM cell. (b) 4T4R-CMOS cell. Selection bits D_i are stored in dedicated SRAM latches. (c) Latency, energy, and area breakdowns for 1R-RRAM and 4T4R-CMOS implementations of IMC PCA.

Fig. 8 shows the two proposed implementations' corresponding energy and area breakdown. For 1R-RRAM, peripheral circuitries such as ADCs and OAs dominate the area occupation, with the array representing a negligible contribution owing to the high density of integration of resistive switching devices. Conversely for the 4T4R-CMOS implementation, the array shows a dominating $50\times$ area occupation with respect to 1R-RRAM owing to the larger memory cell size. From the energy standpoint, the dominant contribution for both implementations is given by the OAs. Particularly, the RRAM solution further shows an increased programming energy contribution, counterbalanced by the increased feedback array dissipation in the CMOS implementation owing to the higher conductance of 4T4R-CMOS cells.

As a comparison, we benchmarked the performance of a commercial GPU [21] in $8\,\text{nm}$ technology. Notably, our analog circuit operates by exploiting electrical equivalents of algebraic relations, *i.e.* results are computed in a continuous domain rather than bit-by-bit. In a digital processor, the same operation is performed in a series of floating-point operations (FLOPs), which can be estimated as 9.5 MOPs under a complexity of $\mathcal{O}(mn^2+n^3)$ [2]. We considered both the

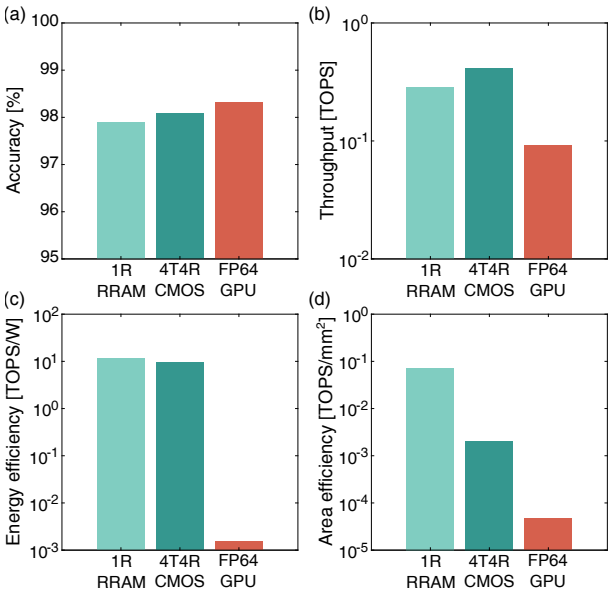

Fig. 9. 1R-RRAM IMC, 4T4R-CMOS IMC, and FP64 GPU comparisons for (a) classification accuracy, (b) throughput, (c) energy efficiency, and (d) area efficiency. IMC implementations show 2 to 4 orders of magnitude improvements from energy and area standpoint while ensuring the same accuracy and throughput of a digital system.

computational throughput at 64-bit precision (129 GFLOPS) and the memory bandwidth (192 GB/s), assuming data to be transferred to consist of matrix $\mathbf{D}$, the eigenvalues λ, and the computed PCs $\mathbf{P}$. For energy and area estimation, we refer to the GPU's thermal design power ($450\,\mathrm{W}$) and die size ($200\,\mathrm{mm}^2$) respectively.

Fig. 8 shows the resulting estimates for accuracy (a), throughput (b), energy efficiency (c), and area efficiency (d) for the IMC 1R-RRAM, IMC-4T4R CMOS, and GPU-based PCAs on the Wine dataset. Both IMC implementations achieve a GPU-comparable throughput and accuracy, with 4T4R-CMOS slightly outperforming 1R-RRAM thanks to the reduced programming time of 4T4R-CMOS cells. On the other hand, 1R-RRAM achieves a superior area efficiency with respect to 4T4R-CMOS thanks to the vastly reduced feedback array size, resulting in an $\times 10^{\gamma}$, with γ between 2 and 4, improvement with respect to GPU. Finally, both IMC solutions achieve similar energy efficiencies in the tens of TOPS/W, with a significant $\times 10^4$ increase with respect to GPU.

V. Conclusions

We present a novel IMC-based circuit capable of performing matrix eigendecomposition by closed-loop analog feedback and study its accuracy and speed, identifying the main trade-offs and providing calibration rules for available parameters. We demonstrate PCA acceleration by sweep-based circuit operation operating the circuit under a sweep-based procedure on the Wine dataset, achieving floating-point equivalent accuracy with 4-bit memory precision. Benchmark simulations show the same throughput as a commercial GPU, with improvements in energy and area efficiency by 2 to 4 orders of magnitude. These results strengthen the position of IMC as the leading next-generation candidate architecture for compact, energy-efficient ML accelerators.

References

[1] I. Jolliffe, "Principal Component Analysis," in *Encyclopedia of Statistics in Behavioral Science*, B. S. Everitt and D. C. Howell, Eds. Chichester, UK: John Wiley & Sons, Ltd, Oct. 2005, p. bsa501. https://onlinelibrary.wiley.com/doi/10.1002/0470013192.bsa501

[2] G. H. Golub and C. F. Van Loan, *Matrix computations*. Baltimore: The Johns Hopkins University Press, 2013, oCLC: ocn824733531.

[3] P. Mannocci, M. Farronato *et al.*, "In-memory computing with emerging memory devices: Status and outlook," *APL Machine Learning*, vol. 1, no. 1, p. 010902, Mar. 2023. doi: 10.1063/5.0136403

[4] M. Le Gallo, A. Sebastian *et al.*, "Mixed-Precision In-Memory Computing," *Nature Electronics*, vol. 1, no. 4, pp. 246–253, Apr. 2018. doi: 10.1038/s41928-018-0054-8

[5] S. Choi, P. Sheridan *et al.*, "Data Clustering using Memristor Networks," *Scientific Reports*, vol. 5, no. 1, p. 10492, May 2015. doi: 10.1038/srep10492

[6] P. Mannocci, A. Baroni *et al.*, "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 Nanotechnology Magazine*, vol. 16, no. 2, pp. 4–13, Apr. 2022. doi: 10.1109/MNANO.2022.3141515

[7] Z. Sun, G. Pedretti *et al.*, "Solving matrix equations in one step with cross-point resistive arrays," *Proceedings of the National Academy of Sciences*, vol. 116, no. 10, pp. 4123–4128, Mar. 2019. doi: 10.1073/pnas.1815682116

[8] P. Mannocci, G. Pedretti *et al.*, "A Universal, Analog, In-Memory Computing Primitive for Linear Algebra Using Memristors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–11, 2021. doi: 10.1109/TCSI.2021.3122278

[9] P. Mannocci and D. Ielmini, "A generalized block-matrix circuit for closed-loop analogue in-memory computing," *IEEE Journal on Exploratory Solid-State Computational Devices and Circuits*, pp. 1–1, 2023. doi: 10.1109/JXCDC.2023.3265803

[10] Z. Sun, G. Pedretti *et al.*, "In-Memory Eigenvector Computation in Time $O(1)$," *Advanced Intelligent Systems*, vol. 2, no. 8, p. 2000042, Aug. 2020. doi: 10.1002/aisy.202000042

[11] K. A. Yeomans and P. A. Golder, "The Guttman-Kaiser Criterion as a Predictor of the Number of Common Factors," *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 31, no. 3, p. 221, Sep. 1982. doi: 10.2307/2987988

[12] A. Björck, *Numerical Methods in Matrix Computations*, ser. Texts in Applied Mathematics. Cham: Springer International Publishing, 2015, vol. 59. https://link.springer.com/10.1007/978-3-319-05089-8

[13] P. Cortez, A. Cerdeira *et al.*, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, Nov. 2009. doi: 10.1016/j.dss.2009.05.016

[14] B. Feinberg, R. Wong *et al.*, in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. Seoul, Korea (South): IEEE, Feb. 2021, pp. 761–774. doi: 10.1109/HPCA51647.2021.00069

[15] F. Tang, Q. Ma *et al.*, "A 28 nm CMOS 10 bit 100 MS/s Asynchronous SAR ADC with Low-Power Switching Procedure and Timing-Protection Scheme," *Electronics*, vol. 10, no. 22, p. 2856, Nov. 2021. doi: 10.3390/electronics10222856

[16] F. Zahoor, T. Z. Azni Zulkifli *et al.*, "Resistive Random Access Memory (RRAM): an Overview of Materials, Switching Mechanism, Performance, Multilevel Cell (mlc) Storage, Modeling, and Applications," *Nanoscale Research Letters*, vol. 15, no. 1, p. 90, Dec. 2020. doi: 10.1186/s11671-020-03299-9

[17] Z. Wang, H. Wu *et al.*, "Resistive switching materials for information processing," *Nature Reviews Materials*, vol. 5, no. 3, pp. 173–195, Jan. 2020. doi: 10.1038/s41578-019-0159-3

[18] W. Gul, M. Shams *et al.*, "SRAM Cell Design Challenges in Modern Deep Sub-Micron Technologies: An Overview," *Micromachines*, vol. 13, no. 8, p. 1332, Aug. 2022. doi: 10.3390/mi13081332

[19] F. Tisseur and J. Dongarra, "A Parallel Divide and Conquer Algorithm for the Symmetric Eigenvalue Problem on Distributed Memory Architectures," *SIAM Journal on Scientific Computing*, vol. 20, no. 6, pp. 2223–2236, Jan. 1999. doi: 10.1137/S1064827598336951

[20] N. Lepri, M. Baldo *et al.*, "Modeling and Compensation of IR Drop in Crosspoint Accelerators of Neural Networks," *IEEE Transactions on Electron Devices*, vol. 69, no. 3, pp. 1575–1581, Mar. 2022. doi: 10.1109/TED.2022.3141987

[21] "NVIDIA RTX A2000 Embedded," 2022. www.nvidia.com/en-us/design-visualization/resources/rtx-embedded/