Analog error correcting codes for defect tolerant matrix multiplication in crossbars

Can Li[1,2], Ron M. Roth[1,3], Cat Graves[1], Xia Sheng[1], John Paul Strachan[1]
[1]Hewlett Packard Labs, HPE, Palo Alto, CA, USA; [2]Department of EEE, The University of Hong Kong, Hong Kong SAR, China; [3]Computer Science Department, Technion, Haifa, Israel. Email: canl@hku.hk, john-paul.strachan@hpe.com

***Abstract*—**Despite great promises shown in the laboratory environment, memristor crossbar, or non-volatile resistive analog memory, based matrix multiplication accelerators suffer from unexpected computing errors, limiting opportunities to replace main-stream digital systems. While many previously demonstrated applications, such as neural networks, are tolerant of small errors, they are challenged by any significant outliers, which must be detected and corrected. Herein, we experimentally demonstrate an analog Error Correcting Code (ECC) scheme that considerably reduces the chance of substantial errors, by detecting and correcting errors with minimum hardware overhead. Different from well-known digital ECC in communication and memory, this analog version can tolerate small errors while detecting and correcting those over a predefined threshold. With this scheme, we can recover the MNIST handwritten digit classification accuracy experimentally from 90.31% to 96.21% in the event an array builds up shorted devices and from 73.12% to 97.36% when current noise is injected. For applications where high reliability and compute precision are demanded, such as in high-performance and scientific computing, we expect the schemes shown here to make analog computing more feasible.

I. Introduction

Memristors, or non-volatile resistive analog memories, have demonstrated great potential in accelerating modern computing workloads, because of their ability to process information directly in its memory, and compute with physical laws. Such hardware, however, is prone to the emerging device imperfection and noises of various kind, and therefore yields inaccuracies in the computing result. While many algorithms make full use of the analog nature of memristor crossbar can tolerate those inaccuracies to some extent, significant error outliers are fatal and must be detected and corrected.

ECC has been widely used in scenarios where noise and errors are frequent, such as data communication over a noisy channel, or where data fidelity can degrade over time, such as in ECC for memories. The idea is that the sender encodes a message with additional redundancy, which allows the receiver to detect if an error has occurred in the message, and possibly correct the error without re-transmission (**Fig.** 1). Prior work has tried to apply this method to memristor-crossbar based computing [1–4], but most address the memristor crossbar memory itself, rather than the analog computing output which is the actual target for error-reduction. A novel theoretical scheme was recently proposed in [5] to address this problem, with mathematical support. However, no experimental demonstration of any ECC in crossbars, to the best of our knowledge, has been reported before. It is noted that the proposed scheme not only benefits memristor in-memory analog computing, but also those based on other analog, or even binary, non-volatile technologies (e.g., Phase change, Ferroelectric, STT-MRAM, and floating-gate memory devices).

II. Analog Error correcting Code

The idea of the proposed analog ECC (a-ECC) is described in **Fig.** 2 – **Fig.** 6. Distinct from digital ECC for data communication, where redundant bits are encoded based on the data to transmit, the proposed a-ECC involves expanding the in-memory computing area, by adding an encoder matrix, the contents of which are based on the original in-memory area for multiplication (**Fig.** 2). In this way, the redundancy output $\vec{r}$ is computed at the same time as the actual multiplication (for $\vec{y}$) takes place. Thus, the calculation of the redundancy output does not impose additional overhead in latency. In most scenarios, the matrix is not frequently updated; thus, the computing of the encoder matrix is low overhead. Another difference is the ability to tolerate small computing errors in normal operations due to the nature of analog computing, but to detect and correct significant outliers (**Fig.** 3), that may result from short/open junctions, memristor conductance state drift, external injected transient noise, environmentally induced corruption, etc.

Fig. 4 illustrates the a-ECC algorithm, and an example encoder matrix and a decoder matrix are shown in **Fig**. 5 to demonstrate how the scheme works. In this particular example, there are seven columns in the original matrix and four columns in the encoder matrix for redundancy outputs. The encoder matrix needs to be configured only once for a given matrix. Since the output vectors ($\vec{y}$ and $\vec{r}$) are weighted sums of each input row vector to the matrix, due to linearity, they shall preserve a specific relationship designed in the choice of the encoder matrix. In the event of a significant error outlier, which could result in the predefined constraint is not met anymore, and thus an error is detected. This can be checked with decoder matrix, involving another matrix multiplication in a crossbar.

The decoder matrix is used to check if the output meets the predefined constraints, i.e., whether an error has occurred. As is shown in **Fig.** 6a, 6b, if all inaccuracies are within a defined tolerance (δ=0.5), the decoder output will be close to zero within a threshold. In the case of a significant outlier, the corresponding column pattern in the decoder matrix will be added to the result, leading to considerable deviation from zero output (**Fig.** 6c–6f). The design of the decoder matrix guarantees that the pattern for any substantial deviations is

unique so that the location of the error can be located by looking up this unique pattern in the decoder matrix. The error can be corrected accordingly (correction in this setting means locating the affected column and computing an estimate for the correct value at that column) based on the output magnitude that exceeds the threshold [5]. The proposed hardware implementation for this algorithm is shown in **Fig.** 7. It should be noted that all the signals remain in the analog domain, and one only needs to convert the multiplication result to the digital domain when desired.

III. Integrated Nanoscale Memristor Arrays

The idea is experimentally demonstrated in our integrated memristor crossbar system. **Fig.** 8 shows the experiment setup, including an integrated memristor crossbar arrays in a chip operated in a PCB board that communicates externally through a micro-controller [6]. The circuits peripheral to the crossbar array include signal routing, amplifiers, analog-digital conversion, sampling, etc., and were taped-out in TSMC's 180 nm technology node. The Ta/TaO_x/Pt memristor devices in the arrays were monolithically integrated with a back-end-of-line process fabricating devices of lateral dimension 50 nm×50 nm (**Fig.** 9). With the platform, we programmed conductance patterns for a convolutional neural network (CNN) that classifies MNIST handwritten digits, with readout conductance map after programming shown in **Fig.** 10.

The accuracy and precision of analog multiplications performed in the integrated crossbar platform are shown in **Fig.** 11. The outputs are normalized from the output current based on the conductance-weight-value ratio. The results show accurate results around zero error, but notable imprecision within ± 0.5, which indicates nearly 5-bit output precision as the range of the output is 0-26, consistent with previous results with off-chip peripherals [7,8]. On the other hand, there remain large but less frequent errors that are outside a tolerable range and need detection and correction.

IV. Analog Error correction Experiments

We first consider a case where devices are programmed between 0-100 μS for MNIST classification with a CNN, and a device in the crossbar is disturbed to a very high conductance (200 μS) state. Prior reports, including ours [9], suggest the network can be reasonably tolerant of device defects, but the result in **Fig.** 12 shows device disturbance at some locations can precipitously drop MNIST classification accuracy from 98.20% to as low as 44.03%, and therefore must be detected and corrected. The effectiveness of our a-ECC is validated by intentionally setting a device located on the upper left corner to 200 μS (**Fig.** 13) and analyzing the output accuracy before and after the error correction (**Fig.** 14). The cumulative probability is shown in **Fig.** 15, from which one sees that the a-ECC successfully and sharply eliminates significant outliers. With this a-ECC and single shorted devices, the experimental classification accuracy on the entire MNIST handwritten digit test set improves from 90.31% to 96.21% (software baseline is 98.20%) (**Fig**. 16). It should be noted that the particular scheme and level of redundancy implemented here detects and corrects one defective device at a time (at any location), allowing the corresponding array to be scheduled for re-programmed to avoid more defective devices.

Some other significant errors may be transient, due to noise induced from the environment, device state fluctuations, etc., and re-programming is not required for such errors. To experimentally emulate such effects, we activated extra rows in the array and programmed some of those devices to a high conductance state (**Fig.** 17) to artificially inject additional current noise to columns (**Fig.** 18). **Fig.** 19 shows that the a-ECC succeeds in correcting most of the injected error, and **Fig.** 20 indicates that the MNIST accuracy rose from 73.12% to 97.36% after the a-ECC corrects the injected noise current. Since this type of error is transient, they can also be fixed by performing the multiplication operation again after a significant inaccuracy is detected without additional programming steps. An interesting phenomenon is that, after analog error correction, the accuracy can be higher than even without injected noise (96.92%), because the a-ECC can also correct intrinsic outlier errors caused by device-wire resistance interactions, IV nonlinearity, etc.

V. Discussion

Although the encoding step of the a-ECC does not impose additional overhead in latency, it requires an additional chip area and thus energy consumption due to extra cells and periphery for encoder and decoder. Analog ECC imposes an inherent trade-off between more encoder column overhead, and the acceptable level of error outliers. This overhead is reduced if only error detection is needed, rather than full correction, enabling a tunable knob for varying applications (**Fig**. 21). In detecting the permanent errors such as device state disturbance, the decoder matrix can be shared among the duplicated convolutional kernels, and only enabled to identify which array needs to be re-programmed, based on the statistical error rate.

In conclusion, we have experimentally demonstrated, for the first time, an a-ECC scheme that detects and corrects computational errors of various kinds for in-memory analog computing. Applying the method to a CNN for MNIST classification, we achieved 97.36% accuracy with significant injected errors that would otherwise lead to 73.12% accuracy, close to the software baseline of 98.20%. Such analog ECC is a critical enabler for future deployment of analog computing using emerging memory devices in applications that demand high reliability.

Acknowledgment

This research was based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract number 2017-17013000002.

References

[1] Niu, D., *et al.*, Y. ASP-DAC (2012); [2] Liu, M., *et al*., Proc. Int. Test Conf. (2019); [3] Wang, M., *et al*., NVMTS (2015); [4] Liu, M., *et al*., ACM Trans. Des. Autom. Electron. Syst. 25, (2020); [5] Roth, R., IEEE ISIT (2019) [6] Li, C., *et al.*, IWM, (2020). [7] Li, C., *et al.*, Nat. Electron., 1, 52-59 (2018). [8] Hu, M., *et al.*, Advanced Materials, 30, 1705914 (2018). [9] Li, C., et al, Nat. Commun., 9, 2385 (2018).

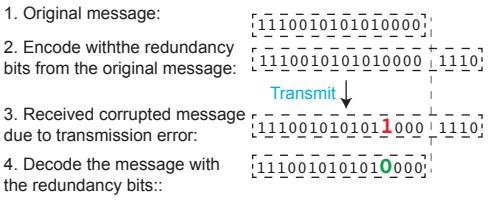

Fig. 1: Conventional digital Error Correcting Codes (ECC) corrects the corrupted data during a noisy transmission.

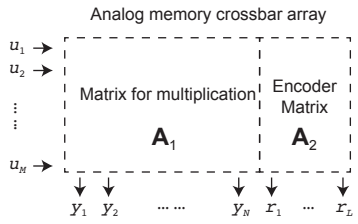

Fig. 2: The proposed analog ECC (a-ECC) that computes the output $\vec{y}$ and the encodes the redundancy $\vec{r}$ in parallel.

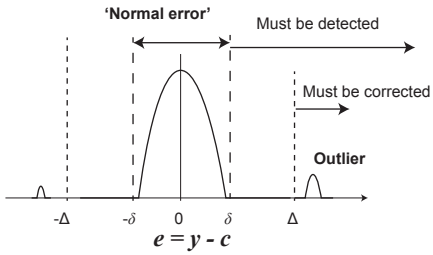

Fig. 3: Schematic of analog computing error, i.e., the difference between expected output and the hardware output, that needs to be detected and corrected.

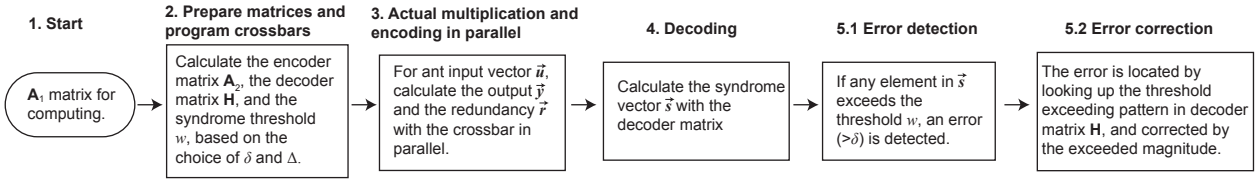

Fig. 4: Diagram for the proposed analog ECC algorithm. Step 2 calculates the encoder and decoder matrices shown in Fig. 5, which is a one-time overhead. Step 3 and Step 4 are performed in the crossbar, which are indicated by the arrows in Fig. 5. The error detection and correction step is illustrated in Fig. 6.

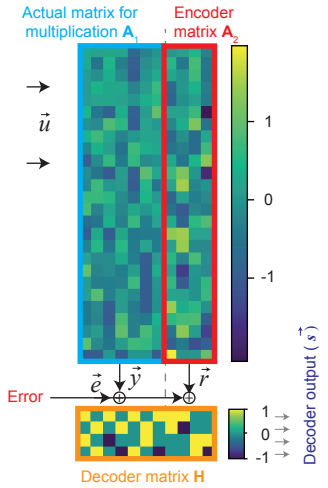

Fig. 5: An example encoder matrix and decoder matrix for analog ECC demonstration. For each input vector, the matrix multiplicatoin generated 11 real numbers with seven intended outputs and four redundancy outputs for error correction.

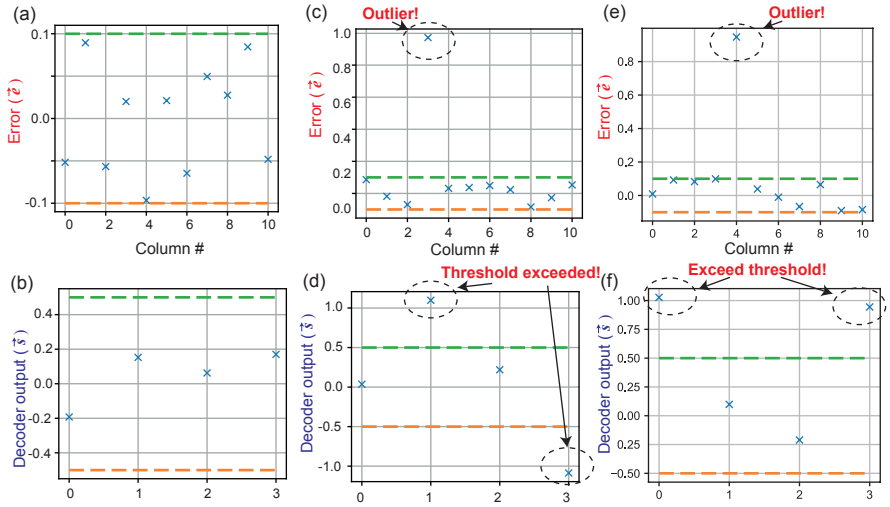

Fig. 6: Examples show how the analog ECC works. **(a,b)** An example of errors within the tolerable limit. **(c,d)** One significant outlier in the 4th column (col #3) leads to the 2nd and 4th decoder output exceeding the upper and lower threshold. The location of the error can be identified by searching the pattern [+1, 0, 0, -1] in the decoder matrix (4th column). **(e,f)** Another example with outlier error in the 5th column, which is located by looking up the pattern [+1, 0, 0, +1] in the decoder matrix.

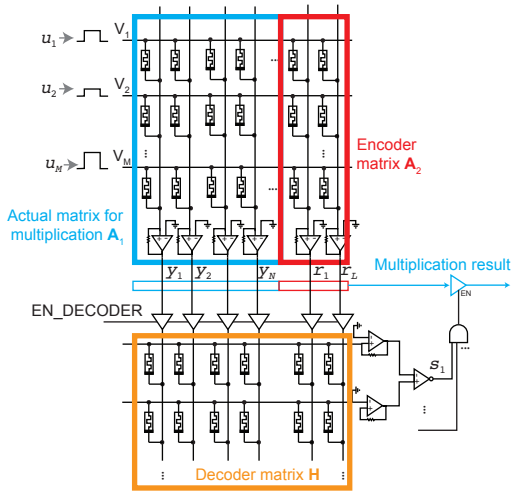

Fig. 7: The proposed hardware implementation of the a-ECC scheme with memristor crossbars for matrix multiplication.

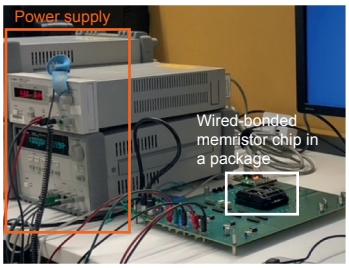

Fig. 8: The a-ECC experimental setup with an integrated memristor crossbar arrays in a packaged chip, the PCB boards tthat controlls the memristor chip and communicates externally.

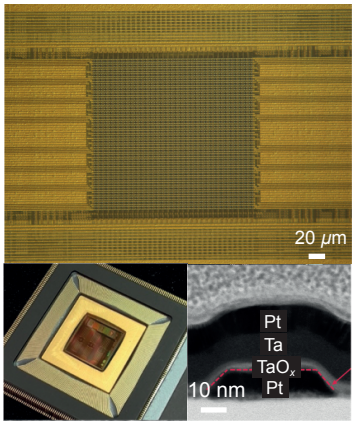

Fig. 9: An integrated chip with nanoscale Ta/TaO_x/Pt memristors and peripheral circuit, e.g. signal routing, amplifiers, sampling, ADC, etc., for a-ECC demonstration.

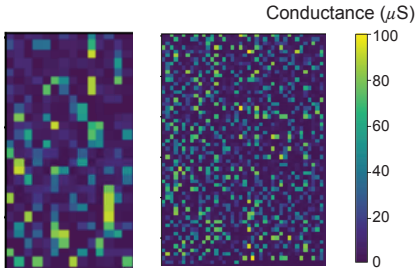

Fig. 10: The readout conductances after experimentally programming a convolutional layer and a fully connected layer for MNIST handwritten digit classification.

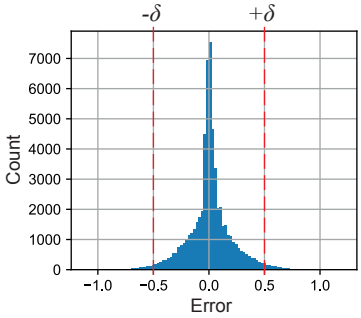

Fig. 11: The statistics of the output error of the analog matrix multiplications in the integrated memristor crossbar array.

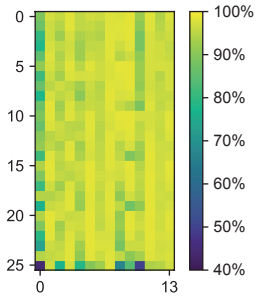

Fig. 12: The simulated MNIST classification accuracy map after one corresponding device is stuck ON.

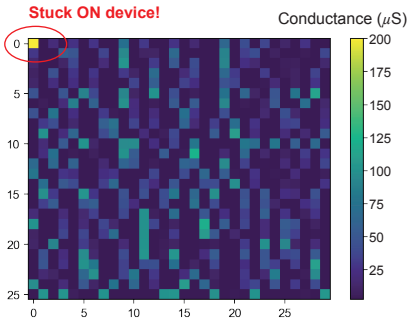

Fig. 13: The readout conductance map after intentionally set one device to a very high conductance state.

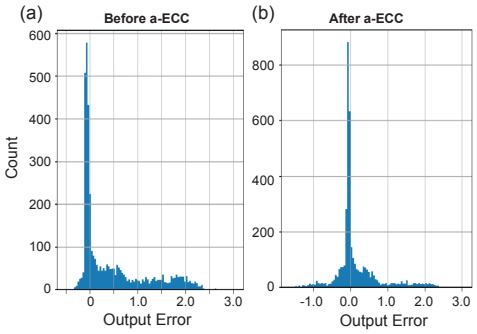

Fig. 14: (a) The distribution of the experimental output error introduced by one stuck-ON device. (b) After a-ECC, most signficant outliers were corrected.

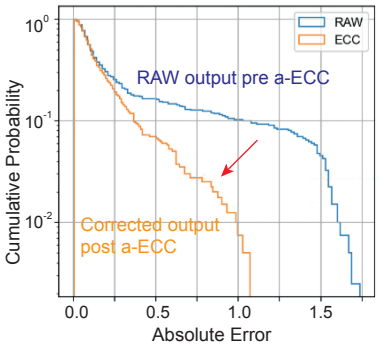

Fig. 15: The cumulative probability plot shows the effectiveness of the analog ECC.

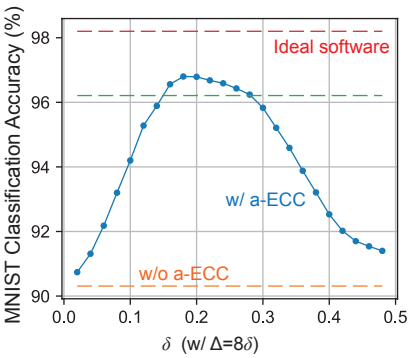

Fig. 16: The a-ECC corrects the misclassifications resulted from the shorted device, and recovers the MNIST accuracy from 90.31% to 96.21% with δ=0.18.

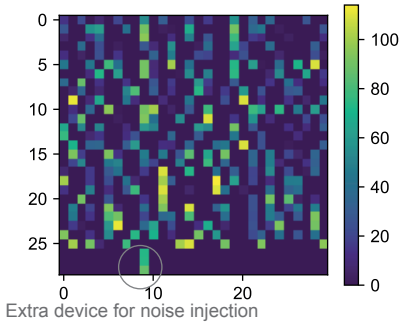

Fig. 17: The readout conductance map after programing several devices to LRS for artificial noise injection.

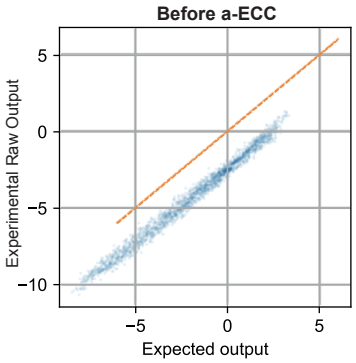

Fig. 18: The experiment output after current noise injection. The dashed line indicates the location where the experiment agrees with the expected output.

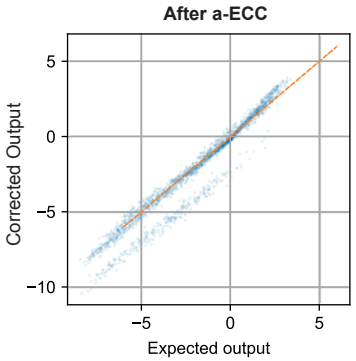

Fig. 29: Most inaccuracies caused by the injected noise are corrected by the a-ECC.

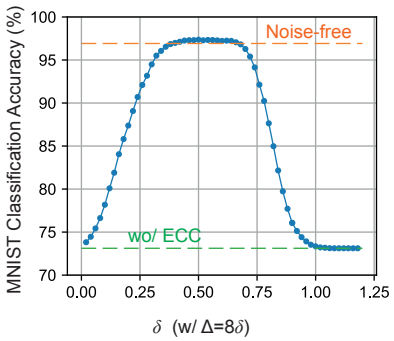

Fig. 20: The MNIST classification accuracy with artificially injected noise is about 73.12%, which was recovered to 97.36% by a-ECC with δ=0.50.

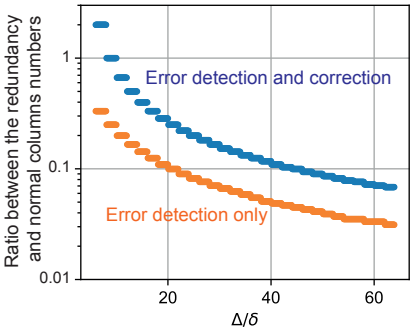

Fig. 21: The trade-off between the acceptable level of error outliers and the number of redundancy columns, with the the number of normal columns for computing set to 512 for this plot.