Topic 1: Sources of nonidealities and compensating techniques in resistive-based in-memory computing neural network accelerators.

ELEC-L352002 - Postgraduate Course in Electronic Circuit Design II V D

Aalto University
School of Electrical
Engineering

Omar Numan

omar.numan@aalto.fi

10.05.2023

Outline

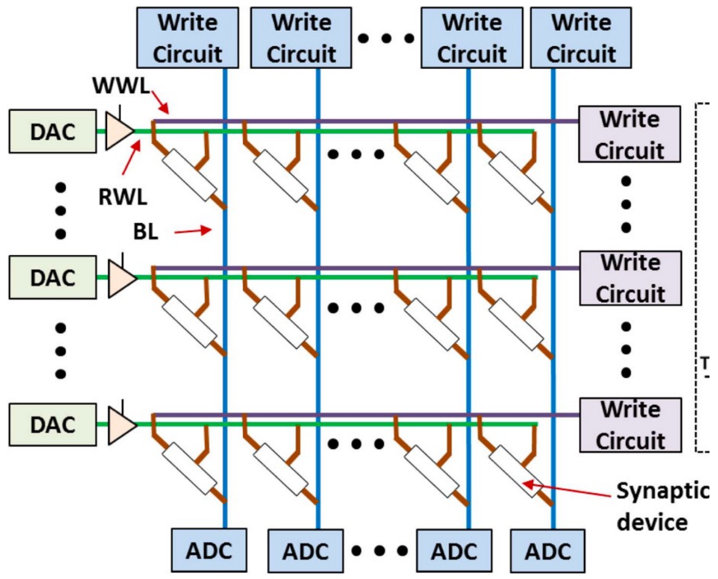
- Introduction
- Criteria for selecting the optimal compensation technique.
- Resistive memory crossbar nonidealities
- Different compensating techniques
- Conclusion
- Assignment question

Introduction

- We will examine the nonidealities associated with resistive memory crossbars and their effects on the performance of neural network (NN) accelerators.
- Our focus will be on resistive memory crossbar, which have gained significant interest in realizing NN accelerators due to their efficiency in carrying out vector-matrix multiplication (VMM).
- A crossbar array consists of a grid of horizontal and vertical wires, with resistive memory elements located at each intersection.
- Resistive crossbar may be designed using a range of emerging devices (e.g., memristors, ReRAM, or PCM devices).
- **Typical resistive crossbar operation**:

Digital inputs are first converted to voltages and applied to the rows of the crossbar (which is programmed with the weights as conductances), and the resulting column currents are converted back into digital form.

- Various device- and circuit-level nonidealities can lead to errors in the computed VMMs.



* **Resistive crossbar array**

* JAIN et al. “RxNN”

Compensation techniques

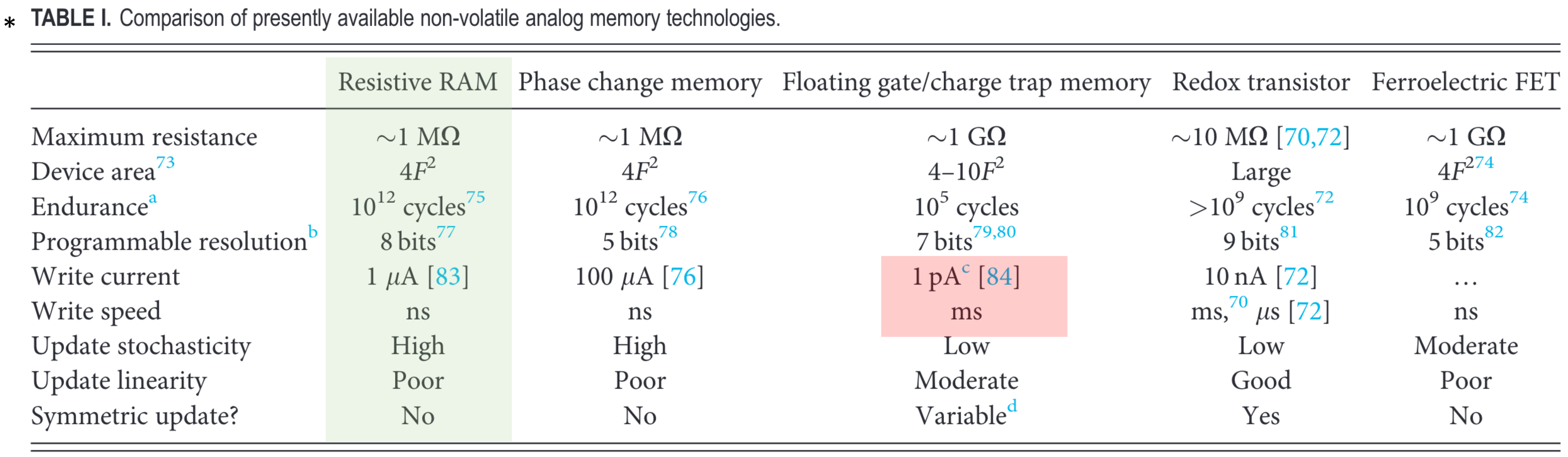
Types of compensation techniques:

- **Device-level:**
  - Crossbar memory devices technology.

- **Circuit-level:**
  -Accelerator's architecture, e.g., DNN, CNN.
  -Representation of the input signal.

- **Algorithm-level:**
  -Accelerator for inference or inference\training.

Crossbar memory devices technology



* **TABLE I.** Comparison of presently available non-volatile analog memory technologies.

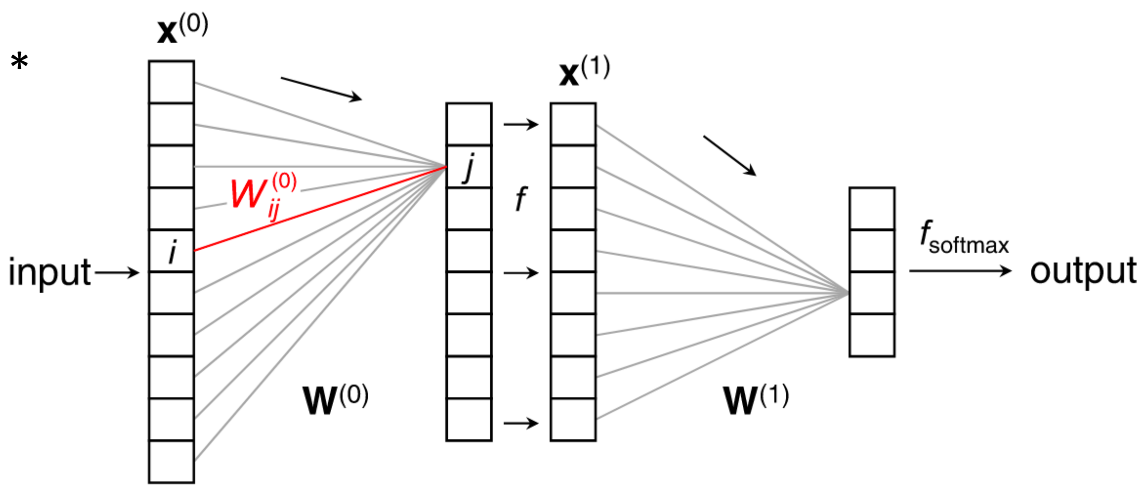
	Resistive RAM	Phase change memory	Floating gate/charge trap memory	Redox transistor	Ferroelectric FET
Maximum resistance	~1 MΩ	~1 MΩ	~1 GΩ	~10 MΩ [70,72]	~1 GΩ
Device area[73]	$4F^2$	$4F^2$	4–$10F^2$	Large	$4F^2$[74]
Endurance[a]	10^{12} cycles[75]	10^{12} cycles[76]	10^5 cycles	$>10^9$ cycles[72]	10^9 cycles[74]
Programmable resolution[b]	8 bits[77]	5 bits[78]	7 bits[79,80]	9 bits[81]	5 bits[82]
Write current	1 μA [83]	100 μA [76]	1 pA[c] [84]	10 nA [72]	…
Write speed	ns	ns	ms	ms,[70] μs [72]	ns
Update stochasticity	High	High	Low	Low	Moderate
Update linearity	Poor	Poor	Moderate	Good	Poor
Symmetric update?	No	No	Variable[d]	Yes	No

- ReRAM features: excellent scalability, BEOL compatibility, variable conductance, low write energy and latency, and high endurance.

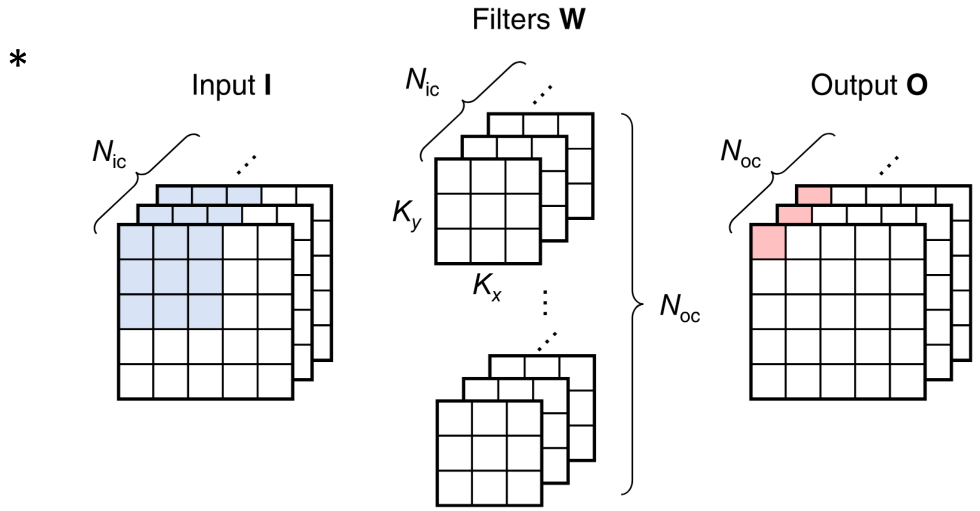
- In this presentation, we will assume the use of ReRAM to create crossbar arrays and to study the nonidealities associated with these devices.

*Xiao et al. "Analog arch."

Architectures of artificial neural network



Deep neural networks (DNN)



Convolutional neural networks (CNN)

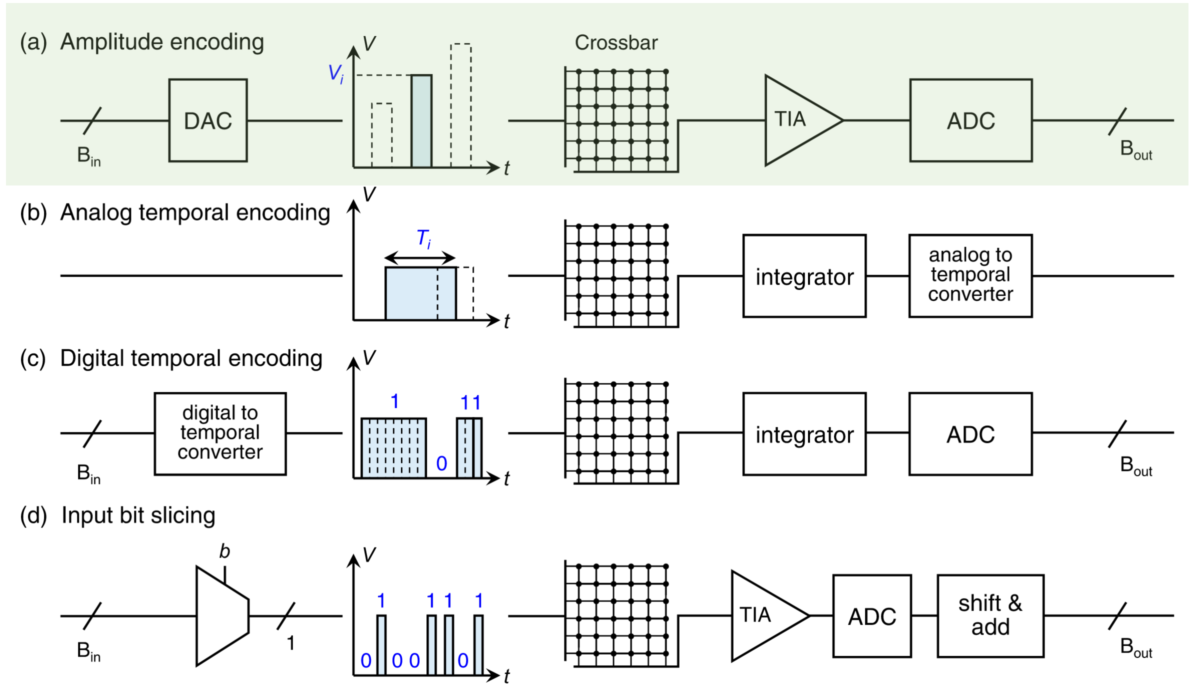
- Use fully-connected layers.
- Requires many trainable parameters.
- Suitable for problems where the input data doesn't have a specific spatial structure.
- Machine learning tasks: speech recognition and natural language processing.

- Use convolutional layers.
- Reduced number of trainable parameters.
- Suitable for for processing grid-like data, e.g., images.
- Machine learning tasks: image classification and object detection.

*Xiao et al. "Analog arch."

Representation of input signals

*



Four different schemes for representing the crossbar input signal and the associated peripheral circuitry in the signal path.

(a) In Amplitude encoding:

- The entire VMM can be completed in a single crossbar read operation.
- Latency does not scale with the input's precision.
- DAC's area and power consumption could scale exponentially with the DAC resolution.

(b) In Temporal encoding:

- Inputs are represented with a pulse having a fixed amplitude and variable duration.
- Latency will increase exponentially with the computation resolution.

* Xiao et al. "Analog arch."

Accelerators for inference and training

Although they share some standard features, the unique requirements of each task lead to several architectural differences:

- **On-chip vs. off-chip training**: Inference acc. -> off-chip training. Training acc. -> on-chip training.

- **Architecture complexity**: Inference acc. are less complex than training acc.
  (training acc. perform forward propagation, backpropagation, weight updates, and loss calculations.)

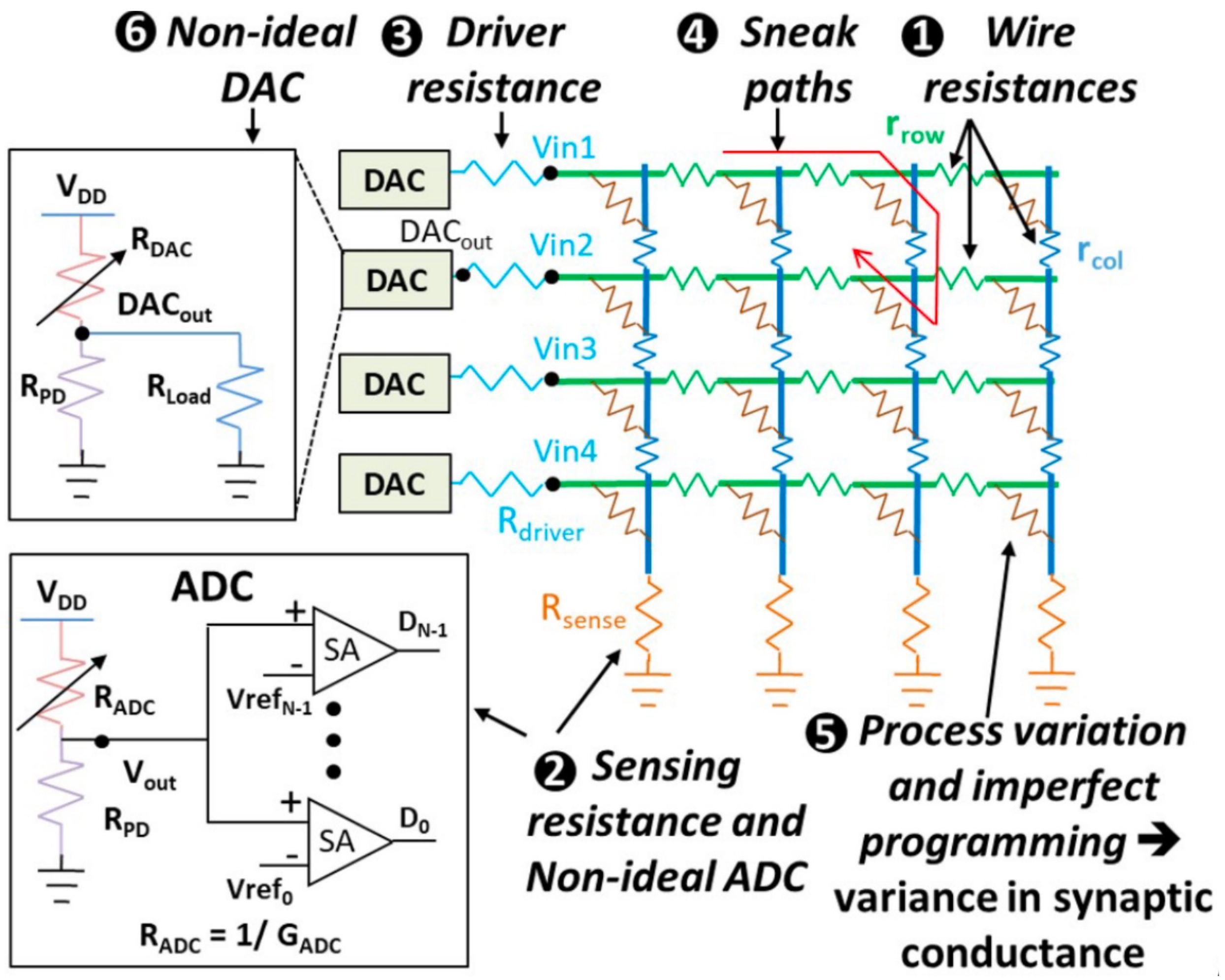
- **Hardware requirement**: Training acc. require additional hardware components compared to inference acc.
  (e.g., additional memory to store intermediate results, specialized circuitry for gradient calculations, and mechanisms for weight updates).

- **Precision requirement**: Training acc. demand higher precision to ensure the convergence
  and stability of the learning process.

- **Power consumption and energy efficiency**: training acc. consume more power and have lower energy
  efficiency compared to inference acc.

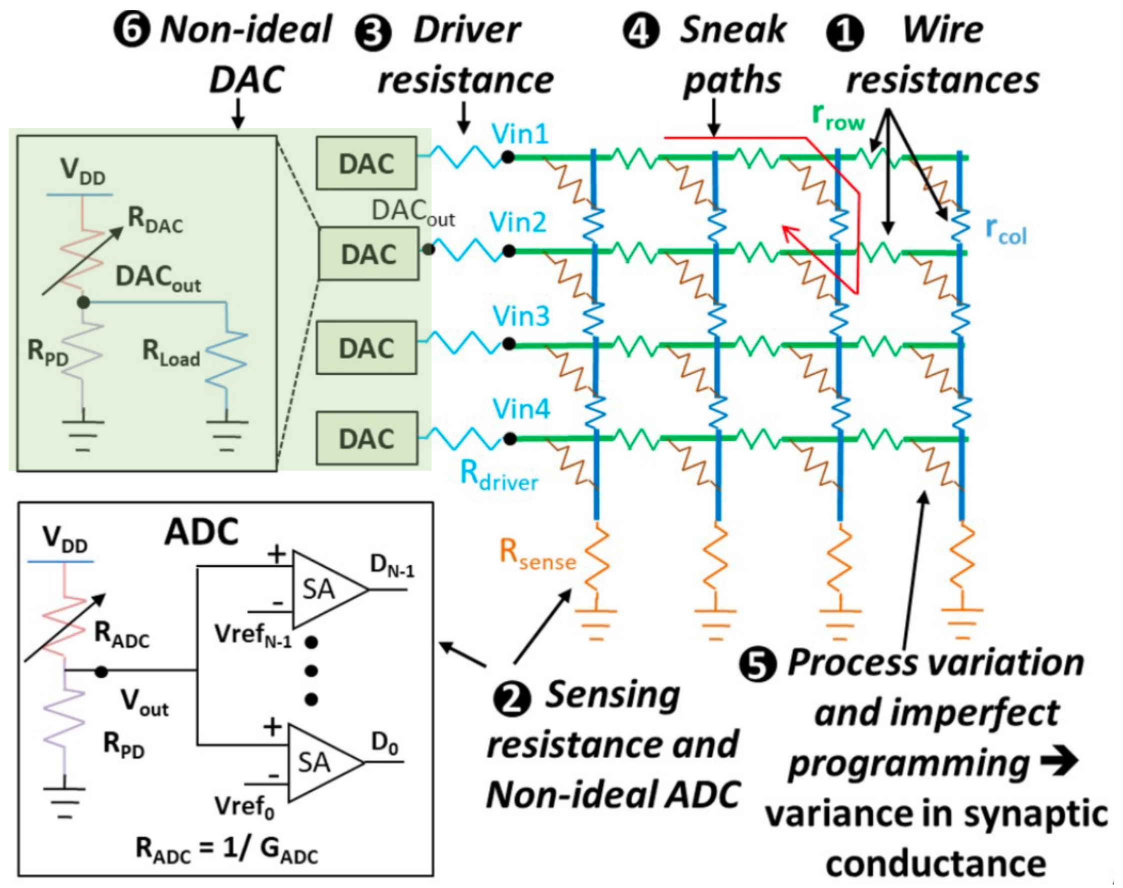
Resistive Crossbar Nonidealities

*



* JAIN et al. "RxNN"

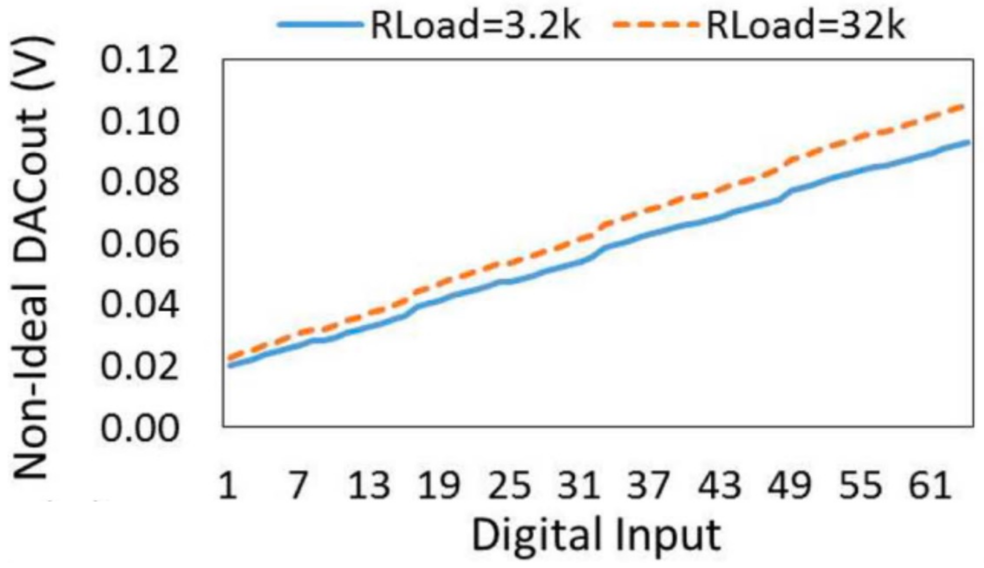
Non-ideal DACs



*

$$Error\ (DAC_{out}) = \frac{V_{DD}\,R_{DAC}}{R_{DAC}+R_{PD}} - \frac{V_{DD}\,R_{DAC}}{R_{DAC}+(R_{PD}\,||\,R_{Load})}$$

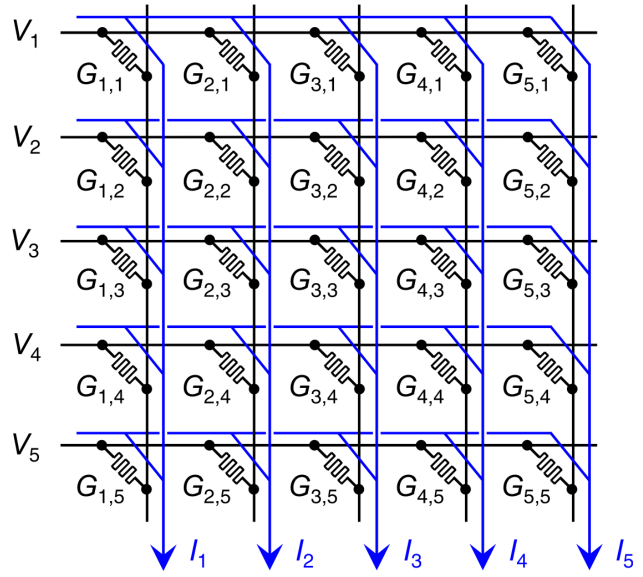
Ideal DAC_{out} Non-Ideal DAC_{out}



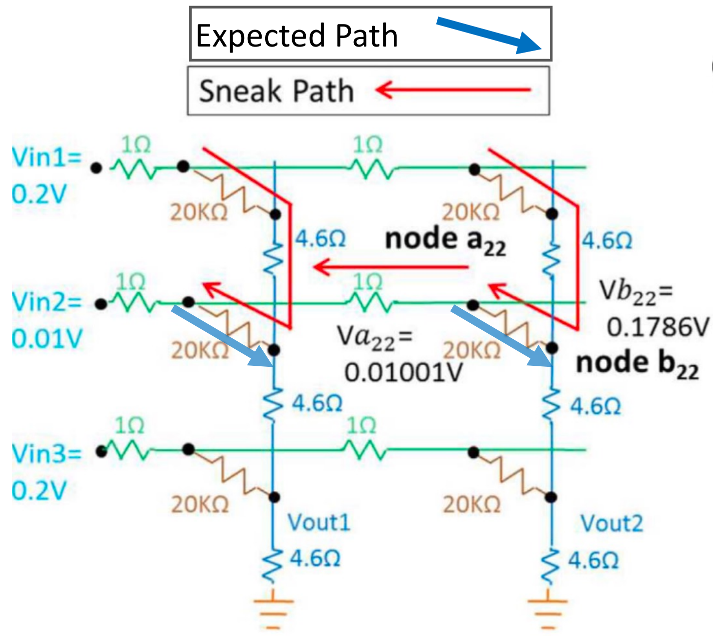
* JAIN et al. “RxNN”

Sneak Paths During VMM

* Ideal Current Paths



**



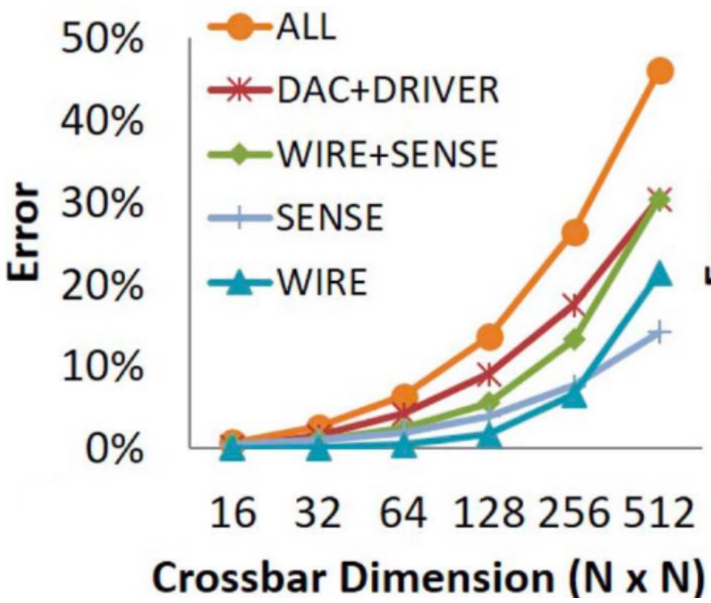
Example for a given crossbar state

- Due to parasitic wire resistances, the internal node voltages may vary, resulting in additional current paths, known as sneak paths.

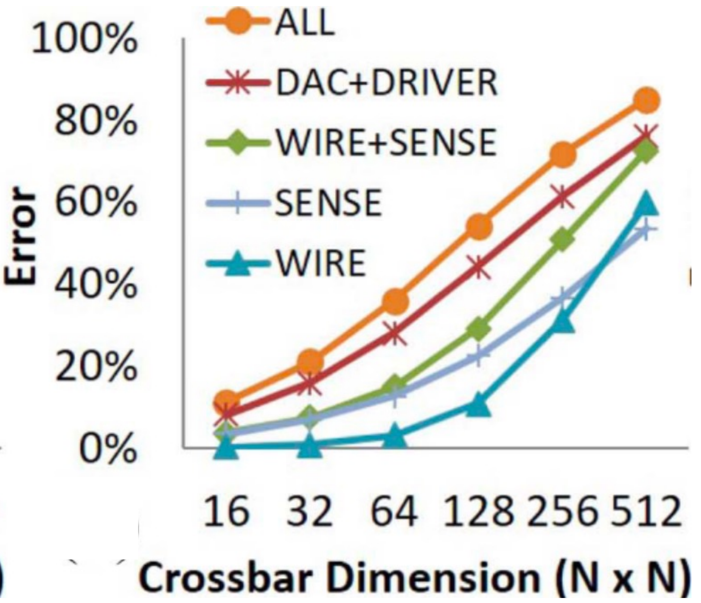
*Xiao et al. "Analog arch."
**JAIN et al. "RxNN"

Crossbar dimension

[*] Sensitivity to Crossbar Size for Various Crossbar Dimensions



R_{MAX}



R_{MIN}

- The errors due to a combination of nonidealities, or individual nonidealities, increase with crossbar dimension.

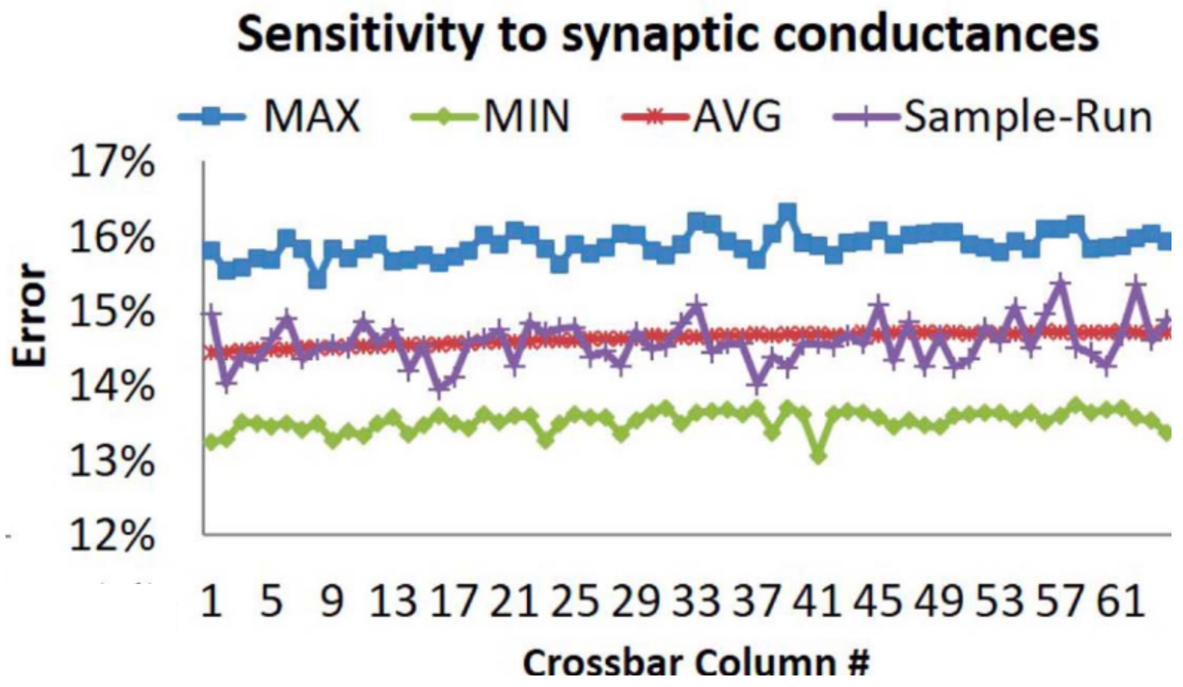
Error percentage ∝ Crossbar dimension:

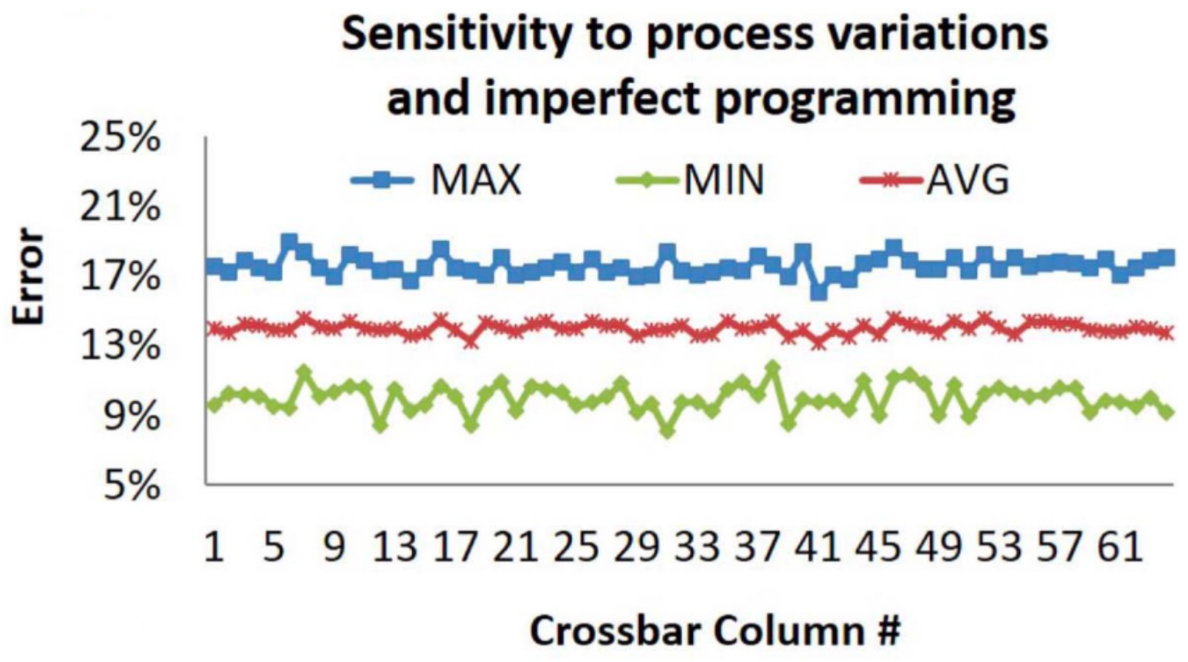
1) Wire resistances increase.
2) Variability of DAC load resistance increases.
3) Variability of BLs resistance increases.

[*] JAIN et al. “RxNN”

Other Errors

*





- Errors due to the variation of the resistive memory device conductance.

- Errors during the right operation of the resistive memory devices.

* JAIN et al. “RxNN”

Sneak paths solutions

- **Multistage reading by HP Labs team, the reading procedure is given as:**[*]

(1) Perform current measurement in the path for the target memory cell.
(2) Put the target memory cell in the OFF state, and perform current measurement in the path for the target memory cell.
(3) put the target memory cell in the ON state, and perform current measurement in the path for the target memory cell.
(4) Based on these three current measurements, the memory cell's initial state is decided.
(5) Return the memory cell to its (assumed) original state.

This adaptive measurement algorithm:

(1) Requires a large amount of time.
(2) Requires large sensing circuits (three sample-and-hold circuits, a voltage comparator, voltage divider, and control circuits).
(3) This technique will also be inefficient for the narrow noise margins at large array sizes, since the effect of sneak paths will dominate, and the resistance value of the target cell will be negligible.

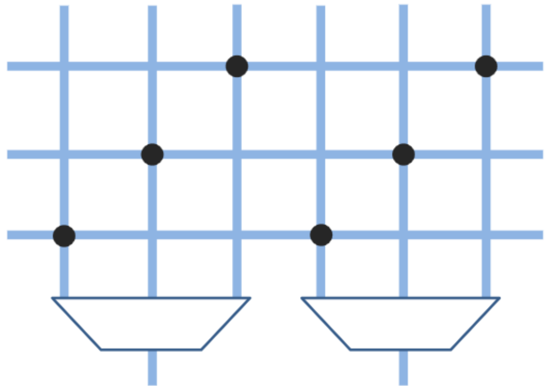
* DOI: 10.1088/0957-4484/20/42/425204

Sneak paths solutions

- **Unfolded architecture solution:**

(1) This method is is based on having a separate column for each memristor.
(2) This method eliminates the sneak paths problem entirely.
(3) It enormously reduces the crossbar density.

$$D_{uf} = \frac{D_o}{\text{no. of rows}}$$



* Unfolded architecture

- **Complimentary memristors solution:**

(1) In this technique two complimentary memristors are used as the memory cell, so that their total resistance is always (R_{on} + R_{off}).
(2) Having always a high resistance cell reduces the sneak-path current significantly.
(3) This method requires a complex reading technique.

* DOI: 10.1145/1785481.1785548

Sneak paths solutions

- **Transistor gating (1T1R) architecture:**

(1) One commonly used solutions for the sneak paths is adding a transistor in series to each memory cell.

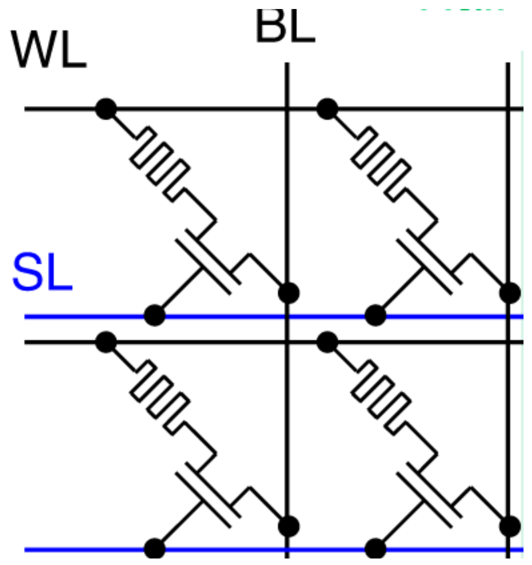
(2) Producing a new cell of one transistor and one resistor/memristor (1T1R/1T1M), which is a three-terminal device.

(3) The transistor gate makes the third terminal to control the access to the memory device during writes.

(4) This method will reduce the high memristor-memory crossbar density, since the gating transistor's size is much larger than the memristor's.

(5) Using small transistors will reduce the sneak paths but it will not eliminate it, since small-size transistors introduce higher leakage currents.

(6) Additional wires are required (Source-Lines) to control the transistors' gates.



* **1T1R crossbar**

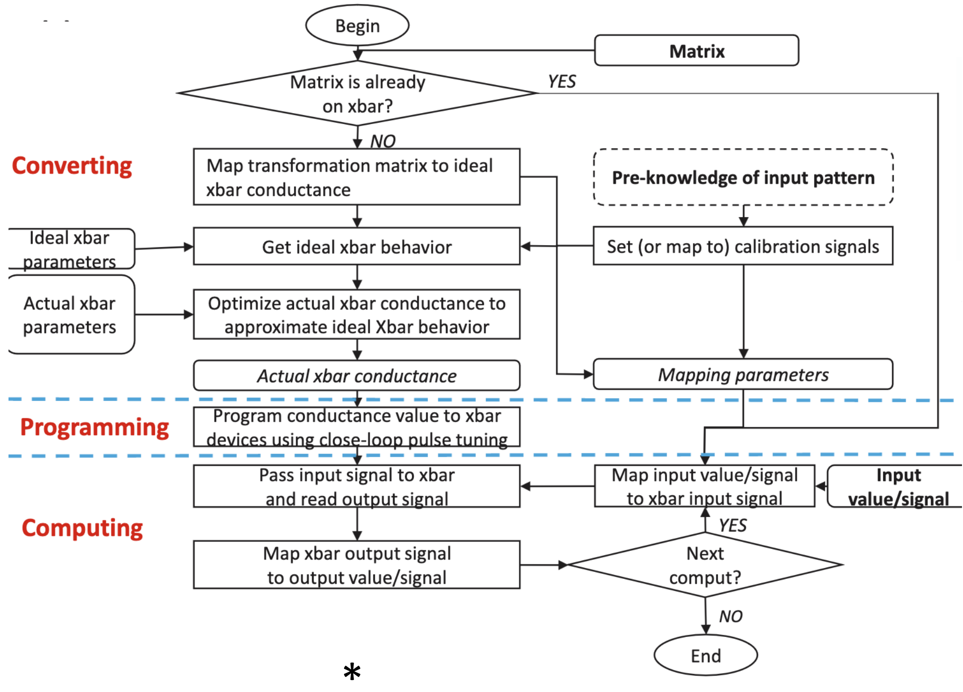
*Xiao et al. "Analog arch."

Mitigating parasitic resistances

(1) The parasitic wire resistances include parasitic voltage drops that cause errors in the output currents.
(2) This effect grows linearly with the crossbar array size.

Conversion algorithm:

(1) A matrix is first linearly mapped to an ideal memristor crossbar to get the ideal crossbar behavior.
(2) It assumes the ideal crossbar has zero wire resistance, a perfectly linear I-V relationship in the memory devices, and zero noise.
(3) The conversion algorithm then simulates the actual (non-ideal) current and voltages across the realistic crossbar array and tunes the device conductances to match the current that should pass through each memory device in an ideal crossbar.
(4) After the conversion, a close-loop tuning scheme is used to program memristors to the desired conductance value.



Begin
Matrix
Matrix is already on xbar?
YES
NO
Converting
Map transformation matrix to ideal xbar conductance
Pre-knowledge of input pattern
Ideal xbar parameters
Get ideal xbar behavior
Set (or map to) calibration signals
Actual xbar parameters
Optimize actual xbar conductance to approximate ideal Xbar behavior
Actual xbar conductance
Mapping parameters
Programming
Program conductance value to xbar devices using close-loop pulse tuning
Computing
Pass input signal to xbar and read output signal
Map input value/signal to xbar input signal
Input value/signal
YES
Map xbar output signal to output value/signal
Next comput?
NO
End

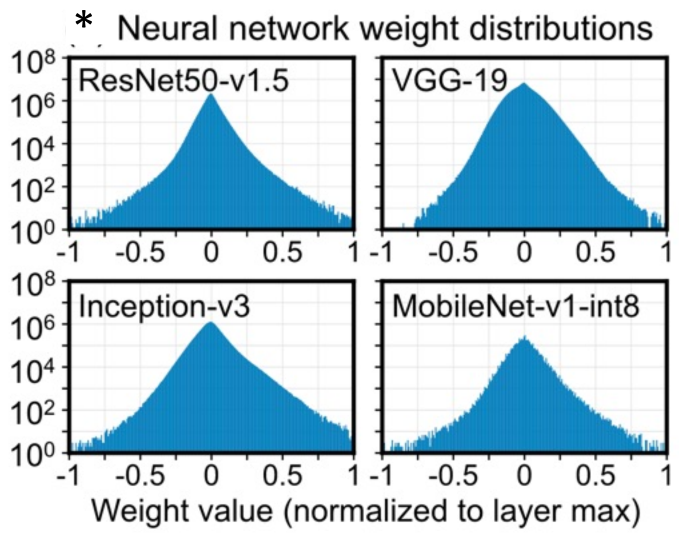
* **DPE workflow**

* DOI: 10.1145/2897937.2898010

Mitigating parasitic resistance

Proportional Mapping:

(1) A very common property of neural networks is the abundance of low-valued or zero-valued weights.
(2) Make use of zero and small-valued weights in analog accelerators by using proportional mapping.
(3) **Proportional mapping**: a linear relationship between numerical values in the algorithm and the physical quantities in the analog hardware.
(4) Weight values are mapped to conductances in proportion to their magnitude.
(5) This method requires using cells with a high On/Off ratio (G_{max}/G_{min}).
(6) The method reduces the average cell conductance by orders of magnitude.
(7) Reduction of the average conductance proportionally decreases the memory device programming errors and parasitic voltage across the crossbar columns and rows.



*Xiao et al. “Analog arch.”

Mitigating parasitic resistance

Other methods to mitigate errors due to parasitic voltage drops:

(1) Adding series resistances to the output periphery of the array to equalize the parasitic voltage drops seen by all parts of the crossbar.

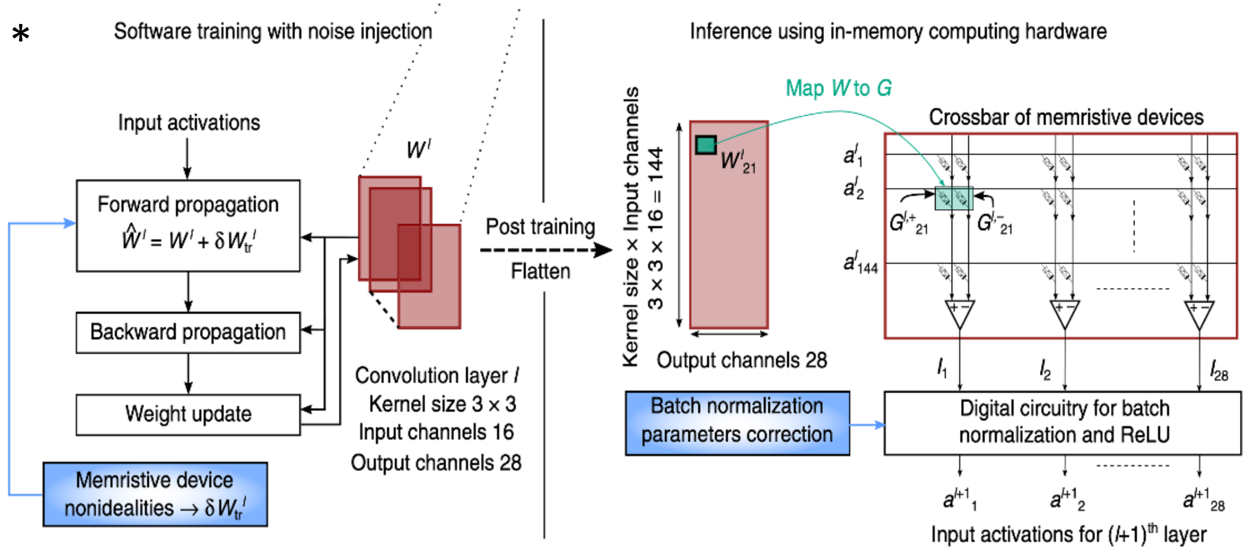
(2) Another method shows that a neural network can learn around the parasitic voltage drops by modeling their effect as an injection of Gaussian noise on the VMM results during training.

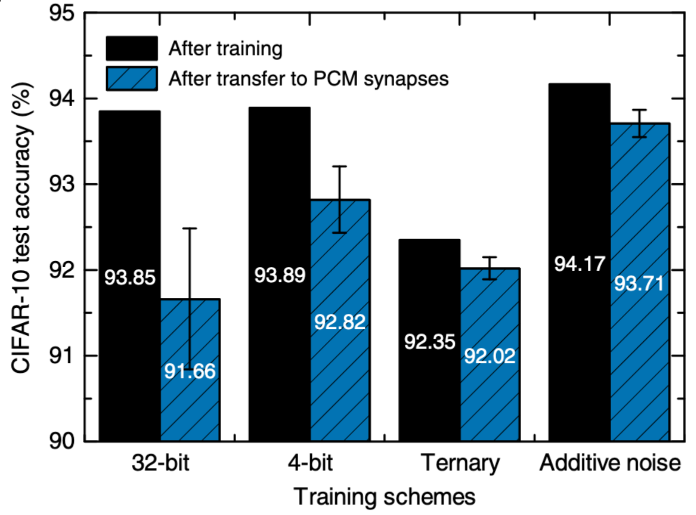
[1] S. Agarwal, R. L. Schiek and M. J. Marinella, "Compensating for Parasitic Voltage Drops in Resistive Memory Arrays," *2017 IEEE International Memory Workshop (IMW)*, Monterey, CA, USA, 2017, pp. 1-4, doi: 10.1109/IMW.2017.7939075.

[2] Z. He, J. Lin, R. Ewetz, J. -S. Yuan and D. Fan, "Noise Injection Adaption: End-to-End ReRAM Crossbar Non-ideal Effect Adaption for Neural Network Mapping," *2019 56th ACM/IEEE Design Automation Conference (DAC)*, Las Vegas, NV, USA, 2019, pp. 1-6.

Compensating for device nonidealities

(1) This method focuses on phase-change memory devices.
(2) Injection of Gaussian noise into the matrix weights during training.
(3) Errors due to read and write noises are injected as a Gaussian distribution, which is usually the case for analog memory devices.
(4) Adding noise to the weights only in the forward propagation is sufficient to achieve close to baseline accuracy.
(5) The tests are performed for ResNet-32 neural network model on the CIFAR-10 dataset.
(6) The results show that performing training by injecting Gaussian noise has the best overall performance.

*

Software training with noise injection
Inference using in-memory computing hardware
Input activations
Forward propagation
$\hat{W}^l = W^l + \delta W^l_{tr}$
Backward propagation
Weight update
Memristive device nonidealities → δW^l_{tr}
W^l
Convolution layer l
Kernel size 3 × 3
Input channels 16
Output channels 28
Post training
Flatten
Kernel size × Input channels
3 × 3 × 16 = 144
Output channels 28
Map W to G
W^l_{21}
Crossbar of memristive devices
a^l_1
a^l_2
a^l_{144}
G^{l+}_{21}
G^{l-}_{21}
I_1
I_2
I_{28}
Batch normalization parameters correction
Digital circuitry for batch normalization and ReLU
a^{l+1}_1
a^{l+1}_2
a^{l+1}_{28}
Input activations for (l+1)th layer

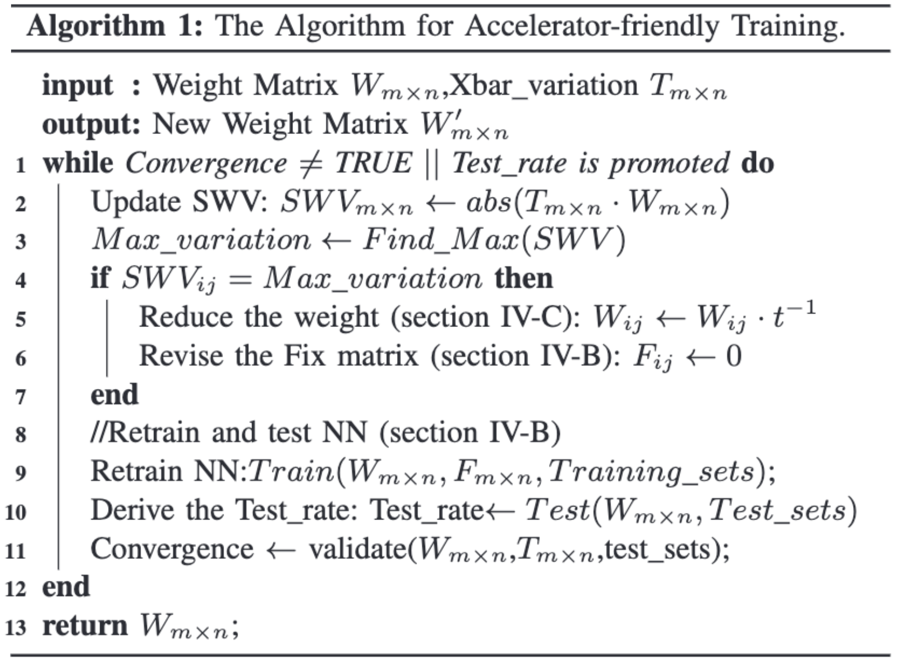
*


* DOI: 10.1038/s41467-020-16108-9

Compensating for process variation

(1) Process variations or defects can be suppressed by an appropriate re-mapping of the neural network weights to the hardware.
(2) * Proposes a scheme to assign the rows of the weight matrix to the crossbar rows in a way that minimizes the expected deviations on the column outputs due to cycle-to cycle variability.
(3) Inject resistance-variation matrix T_{mxn} with the weight matric W_{mxn}.
(4) Find one weight-memristor mapping with the largest summed weighted variations.
(5) The neural-network is re-trained with a small learning rate.
(6) Validate the training process by testing the derived neural-network and by checking the convergence of the training process.
(7) The iterative procedure goes on as long as the training process is not convergent, or the test rate can still be promoted.
(8) Otherwise, the algorithm terminates and returns the new weight matrix.

*



Algorithm 1: The Algorithm for Accelerator-friendly Training.

```
input  : Weight Matrix $W_{m\times n}$,Xbar_variation $T_{m\times n}$
output: New Weight Matrix $W'_{m\times n}$
while $Convergence \neq TRUE \;||\; Test\_rate\ is\ promoted$ do
    Update SWV: $SWV_{m\times n} \leftarrow abs(T_{m\times n} \cdot W_{m\times n})$
    $Max\_variation \leftarrow Find\_Max(SWV)$
    if $SWV_{ij} = Max\_variation$ then
        Reduce the weight (section IV-C): $W_{ij} \leftarrow W_{ij} \cdot t^{-1}$
        Revise the Fix matrix (section IV-B): $F_{ij} \leftarrow 0$
    end
    //Retrain and test NN (section IV-B)
    Retrain NN:$Train(W_{m\times n}, F_{m\times n}, Training\_sets)$;
    Derive the Test_rate: Test_rate$\leftarrow Test(W_{m\times n}, Test\_sets)$
    Convergence $\leftarrow$ validate($W_{m\times n}$,$T_{m\times n}$,test_sets);
end
return $W_{m\times n}$;
```

* DOI: 10.5555/3130379.3130384

Suppressing error propagation

(1) High accuracy can be obtained with sufficiently precise cells and proportional errors, even if analog errors propagate from layer to layer in a DNN.
(2) With less precise cells, some works have relied on ADC quantization to cut off the propagation of device errors.

Advantages of using ADC:

(1) They are necessary to integrate a memristor-based analog VMM module into the digital environment, as functions like pooling, ReLU, and normalization still need digital implementation.
(2) The calibration step can be performed at the ADC stage to improve crossbar results further.
(3) ADC provides quantization, which is helpful in filtering out the output error of memristor-based analog VMM modules, to prevent the error accumulation of inter- and intra-layers in DNNs.

Conclusion

- Analog accelerators face unique challenges that arise from the crossbar geometry and from the individual memory devices that constitute the crossbar.
- Nonideal physical properties of the devices can be compensated using circuit-level techniques.
- Another general approach that is gaining popularity is to learn around these non-idealities by designing the neural network training algorithm with the specific devices or array/circuit effects in mind.

References

- S. Jain, A. Sengupta, K. Roy, and A. Raghunathan, “RxNN: A Framework for Evaluating Deep Neural Networks on Resistive Crossbars,” *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 40, no. 2, pp. 326–338, Feb. 2021.
- T. P. Xiao, C. H. Bennett, B. Feinberg, S. Agarwal, and M. J. Marinella, “Analog architectures for neural network acceleration based on non-volatile memory,” *Applied Physics Reviews*, vol. 7, no. 3, p. 031301, Sep. 2020.
- Y. Jeong, M. A. Zidan, and W. D. Lu, “Parasitic Effect Analysis in Memristor-Array-Based Neuromorphic Systems,” *IEEE Trans. Nanotechnology*, vol. 17, no. 1, pp. 184–193, Jan. 2018.
- S. Agarwal, R. L. Schiek, and M. J. Marinella, “Compensating for Parasitic Voltage Drops in Resistive Memory Arrays,” in *2017 IEEE International Memory Workshop (IMW)*, Monterey, CA, USA: IEEE, May 2017.
- Z. He, J. Lin, R. Ewetz, J.-S. Yuan, and D. Fan, “Noise Injection Adaption: End-to-End ReRAM Crossbar Non-ideal Effect Adaption for Neural Network Mapping,” in *Proceedings of the 56th Annual Design Automation Conference 2019*, Las Vegas NV USA: ACM, Jun. 2019.
- B. Feinberg, S. Wang, and E. Ipek, “Making Memristive Neural Network Accelerators Reliable,” in *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Vienna: IEEE, Feb. 2018.

Assignment

- Summarize in one paragraph the method used in [1] or [2] to mitigate the errors due to parasitic voltage drops generated from wires parasitic resistances.

- For [1] read section II.A (Parasitic compensation)
- For [2] read section 4 (4.2 & 4.3).

[1] S. Agarwal, R. L. Schiek and M. J. Marinella, "Compensating for Parasitic Voltage Drops in Resistive Memory Arrays," *2017 IEEE International Memory Workshop (IMW)*, Monterey, CA, USA, 2017, pp. 1-4, doi: 10.1109/IMW.2017.7939075.

[2] Z. He, J. Lin, R. Ewetz, J. -S. Yuan and D. Fan, "Noise Injection Adaption: End-to-End ReRAM Crossbar Non-ideal Effect Adaption for Neural Network Mapping," *2019 56th ACM/IEEE Design Automation Conference (DAC)*, Las Vegas, NV, USA, 2019, pp. 1-6.