In-Memory Noise Estimation using LDPC Codes for Reliable Edge Matrix-Vector Multiplication

Yotam Gershon Yuval Cassuto
The Viterbi Faculty of Electrical and Computer Engineering, Technion - Israel Institute of Technology
Email: {yotamgr@campus, ycassuto@ee}.technion.ac.il

***Abstract*—Reliable matrix-vector multiplication (MVM) in edge devices is a key enabler for modern computational tasks such as AI inference. Novel memory technologies, such as MRAMs, enable highly efficient analog computation of MVM. While significantly improving speed and power consumption, these architectures suffer from reduced reliability. We introduce a coding scheme in which the parity constraints of the code are employed toward in-memory noise estimation, in addition to their traditional role of error correction. Our scheme relies on objects we call analog syndromes, which on one hand can be efficiently computed in memory, and on the other hand are shown analytically to provide more accurate estimation than classical logical syndromes. The scheme utilizes a previously introduced bilayer LDPC design with sub-block locality, and describes how to design the degree distributions.**

I. Introduction

Matrix-vector multiplication (MVM) is one of the most widely used operations in modern computing systems, serving as the primary computationally intensive task in domains such as artificial intelligence (AI) and, in particular, machine learning (ML) [1]. A major bottleneck in MVM speed and power consumption is data transfer from memory to processing units (PU), a challenge known as the memory wall [2]. This provides a constant drive for the introduction of *accelerated computing* capabilities, with *in-memory computing* being one of the most promising candidates [3]. Specifically, by employing the unique properties of novel memory technologies, such as magneto-resistive random access memories (MRAM) [4], dot-product (DP) operations can be performed in the *analog domain*. One of the most efficient approaches is to store the desired matrix as a memory array, drive the rows simultaneously based on some input vector's values, and measure the accumulated conductance at each column (e.g. [5]). Thanks to the highly parallel nature of such mechanisms, they provide a significant reduction in MVM latency and power. This may be especially useful for edge devices performing AI tasks [6], [7]. However, these architectures suffer from challenging reliability, due to the combination of inherently less accurate analog computations and highly limited error control methods that can be embedded in-memory. A widely used technique is to encode the data being stored (in our case, the matrix) and transfer it to a PU for *error-correction decoding*. When performed frequently, this technique adversely affects the efficiency gain provided by the in-memory computing methodology.

Work supported in part by the US-Israel BSF under grant number 2023627.

Another promising technique is introduced by Roth in [8], and further generalized in [9], where redundancy columns are added to the desired matrix such that the resulted vector will follow parity constraints allowing error correction. This elegant methodology effectively utilizes the MVM already performed in-memory. The drawbacks are the expansion of the output vector and the non-trivial code design.

In this paper, we introduce a middle-ground approach: we propose a scheme that periodically estimates the noise level affecting the memory cells, and when too high, transfers the data to an external PU for error correction and re-write. The key ingredient in this scheme is accurate estimation of the noise standard deviation from in-memory DP-computable objects we call *analog syndromes*. The estimation of bit-flip probability using the classical (here called *logical*) syndromes of linear codes was studied in [10], and further generalized in [11] to enable in-memory error estimation in general-purpose (not MVM) memories. Nevertheless, we show analytically that the analog syndromes offer superior estimation performance over logical syndromes.

In Section II we formulate the model of in-memory DP computations and the problem of DP reliability. Then, in a similar vein to [10], [11], in Section III we perform a *maximum likelihood* (ML) estimation of the noise level ("channel parameter") from the Hamming weight of the analog syndrome. We analyze the proposed estimator using the Fisher information, and show its performance as a function of the noise parameter and the degree of the parity constraints. In Section IV we introduce a construction of low-density parity check (LDPC) codes [12] that enable such in-memory estimation, along with out-of-memory error correction of the entire matrix. This construction is based on bilayer LDPC codes [13], incorporating sub-block locality over columns to enable parity-based syndrome calculation in each column, along with additional global layer enabling strong error correction between columns. We describe how to design the check degree for the local sub-graphs for good estimation performance over different regions of noise levels, and how to design the other degree distributions for good error correction capability. Finally, in Section V we employ the tools from previous sections and suggest an architecture for reliable DP.

II. Problem Formulation

We assume an $n \times m$ binary memory array capable of performing in-memory MVM. $\phi(x), Q(x)$ denote the standard

normal distribution's density and tail functions, respectively. $[x_1,\ldots,x_m]$ mod 2 denotes the element-wise modulo 2 for any length-$m \geq 1$ integer vector. $\mathbf{0},\mathbf{1}$ denote the all-zeros and all-ones vectors with length that will be clear from the context. $w_H(\boldsymbol{x})$ denotes the usual Hamming weight. We treat logical XOR between logical elements in $\{0,1\}$ and 1-bit multiplication between elements in $\{1,-1\}$ as the same under the mapping $0 \leftrightarrow 1, 1 \leftrightarrow -1$. For an irregular LDPC code [12] with a parity check matrix H, Λ_i and Ω_i denote the fractions of variable and check nodes of degree i in the Tanner graph [14] corresponding to H, and $\boldsymbol{\Lambda} = [\Lambda_1,\ldots,\Lambda_{d_v}]$, $\boldsymbol{\Omega} = [\Omega_1,\ldots,\Omega_{d_c}]$, are the code's *degree distributions*. $a_{\mathrm{R}} \triangleq \sum_{i=1}^{d_c} i\Omega_i$ denotes the average check degree.

A. System Model

As a basis for our model we will use a binary MVM architecture, like the one described in [5] using the magnetic tunnel junction (MTJ)-MRAM technology. The cells represent the logical bits of the matrix through parallel (P) and anti-parallel (AP) states, with low and high resistance, respectively, in a complementary bit-cell structure (each cell implements both its primary and complementary values). AP and P states represent -1 and 1 respectively. The 1-bit multiplication between an input element and a matrix element is given by the cell's conductance. We model the conductance of the cell in the i-th row and j-th column of a $n \times m$ array as

$$G_{ij} = g_{ij} + \Delta_{ij}\ ,\ g_{ij} \in \{G_{\mathrm{P}}, G_{\mathrm{AP}}\}\ ,\ \Delta_{ij} \sim \mathcal{N}\left(0,\sigma^2\right),$$

with $G_{\mathrm{P}} > G_{\mathrm{AP}}$ representing the conductance in P and AP states respectively, and Δ_{ij} describing a normally distributed conductance disturbance [15]. Different Δ_{ij} elements are modeled as independent and identically distributed. These disturbances are thought of as the result of accumulated, slowly drifting, physical error mechanisms having *spatial* random distribution. At each MVM instance, n rows are activated simultaneously by the input vector $\boldsymbol{v}$: $v_i = 1$ selects the primary and $v_i = -1$ the complementary value of the cell in the i-th row. The accumulated column conductance is

$$\bar{G}_j \triangleq \sum_{i=1}^{n} G_{ij}(v_i) = \sum_{i=1}^{n} g_{ij}(v_i) + \sum_{i=1}^{n} \Delta_{ij} \triangleq \bar{g}_j + \bar{\Delta}_j,$$
$$\bar{g}_j = n_{\mathrm{e}} G_{\mathrm{P}} + n_{\mathrm{o}} G_{\mathrm{AP}} = n_{\mathrm{e}}\Delta G + n G_{\mathrm{AP}}\ ,\ \bar{\Delta}_j \sim \mathcal{N}(0, n\sigma^2),$$

with n_{e} being the number of even products ($-1/1$ input with $-1/1$ cell) and $n_{\mathrm{o}} = n - n_{\mathrm{e}}$ the number of odd products ($-1/1$ input with $1/-1$ cell), and $\Delta G \triangleq G_{\mathrm{P}} - G_{\mathrm{AP}}$. We notice that by flipping a single matrix cell, $\bar{g}_j$ is shifted by ΔG. At the bottom of each column, a current sampler translates the conductance to a quantized value based on regions of $\bar{G}_j$. We assume a linear mid-tread quantizer [16] to n_{e} (equivalent to the result of the original binary DP result), of the form

$$y_j \triangleq \left\lfloor \frac{\bar{G}_j - nG_{\mathrm{AP}}}{\Delta G} + \frac{1}{2} \right\rfloor = \left\lfloor \frac{n_{\mathrm{e}}\Delta G + \bar{\Delta}_j}{\Delta G} + \frac{1}{2} \right\rfloor \in \mathbb{Z}, \quad (1)$$

which can be achieved (for example) by a flash ADC [17]. For the purpose of noise estimation, we disregard the fact that noise-free values of y_j are bounded to $\{0,1,\ldots,n\}$, while actual readout could truncate the result to this region. All the results can be generalized to any quantizer form with minor technical adjustments, which are not the main focus of this work. Under these considerations, y_j represents the result of a noisy binary DP between the input vector and the j-th column. From this point we take $\Delta G = 1$, without loss of generality, but keep in mind that σ should be replaced with $\sigma/\Delta G$ for the general case, providing a degree of freedom for the design.

B. Dot Product Reliability

We wish to evaluate the reliability of DP operations, compromised by the accumulation of disturbances Δ_{ij}.

Definition 1. *x_j denotes the desired DP result, obtained by replacing $\bar{G}_j$ with the noiseless $\bar{g}_j$ in Eq. (1). Then, $|y_j - x_j| \geq t$ with $t \geq 1$ is defined as a* ***DP error of magnitude*** *t.*

Lemma 2. *The probability of a DP error of magnitude t is*

$$\xi_{n,t}(\sigma) \triangleq 2Q\left(\frac{t-0.5}{\sqrt{n}\sigma}\right). \quad (2)$$

Proof: y_j differs from x_j by t or more when $\left|\bar{\Delta}_j\right| > (t-1)+0.5$, shifting the quantizer's output. ∎

It is clear from Eq. (2) that the probability of error grows (fast) with the number n of rows participating in the DP. This is a known challenge [5] restricting row parallelism, and therefore the efficiency gain from computing in-memory. Our goal is to perform an efficient *in-memory noise estimation* for σ. This allows to decide when to perform read, decode and re-write for noise reduction, so that a certain level of DP reliability, $\xi_{n,t}(\sigma) \leq \xi_{\max}$ is ensured.

III. In-Memory Noise Estimation

In this section we develop the framework of in-memory noise estimation. The idea is to employ the MVM architecture for evaluating the *syndrome* of chosen matrix elements with respect to an LDPC code protecting the matrix. Throughout this section we assume that a set of d activated rows satisfy a parity constraint (as described ahead) in every column. Code design ensuring that property is discussed in Section IV.

A. Analog Syndromes

We define the notion of a syndrome in the analog DP domain. First, a parity constraint in that domain should be defined, analogous to the traditional XOR parity equations.

Definition 3. *We say that a set of d cells satisfies a* ***parity constraint*** *if it contains an even number of P-state cells.*

Following Definition 3, if we activate this set of d cells with 1 in the corresponding rows (without activating any more cells), the resultant $\bar{g}_j$ will have an even n_{e} (i.e., n_{e} mod 2 = 0). We assume, as discussed above, that some set of d rows satisfies a parity constraint on every column simultaneously. We denote this $d \times m$ memory sub-array by E.

Definition 4. *The* ***analog syndrome*** *of E is defined by $\boldsymbol{s}^{\mathrm{A}}(E) \triangleq [y_1,\ldots,y_m]$ mod 2, when y_j are the values from (1) obtained with $\boldsymbol{v} = \mathbf{1}$.*

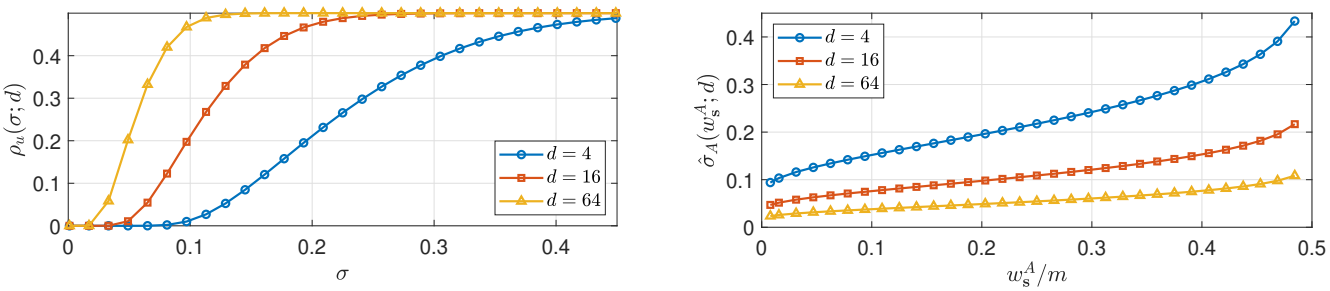

Fig. 1: **Left**: unsatisfied-analog-check probability ρ_u as a function of σ for $d = \{4, 16, 64\}$. **Right**: ML estimator $\hat{\sigma}_A$ as a function of w_s^A/m for the same d values. $m = 128$ in both plots.

Remark 5. *Notice that $s^A(E) \in \{0,1\}^m$. Based on the assumption above, in the noiseless case ($\Delta_{ij} = 0$) we have $s^A(E) = \mathbf{0}$. These properties make the analog syndrome naturally analogous to the traditional logical syndrome.*

B. Maximum Likelihood Noise Estimation

In close relation to the framework in [10], [11], we derive an ML estimator for σ based on the syndrome's Hamming weight, but here using the analog syndrome: $w_s^A \triangleq w_H(s^A)$.

Definition 6. *An **unsatisfied analog check** is defined as a syndrome element that satisfies $y_j \bmod 2 = 1$.*

Lemma 7. *The probability of an unsatisfied analog check is*

$$\rho_u(\sigma;d) \triangleq 2\sum_{\ell=0}^{\infty}(-1)^\ell Q\left(\frac{\ell+0.5}{\sqrt{d}\sigma}\right),\ 0<\sigma<\infty. \quad (3)$$

Proof: A syndrome element satisfies $y_j \bmod 2 = 1$ when $\bar{\Delta}_j$ shifts the quantizer an odd number of steps. That is, when $(2\ell+0.5) \le |\bar{\Delta}_j| < (2\ell+1.5)$, with $\ell \in \mathbb{N}$. ∎

For brevity, we use ρ_u or $\rho_u(\sigma)$ to denote $\rho_u(\sigma;d)$ when the inputs are clear from the context. Now, since $\bar{\Delta}_j$ over different columns are statistically independent, the syndrome's weight w_s^A follows a Binomial distribution of m trials and probability $\rho_u(\sigma)$. It can be readily seen that $\lim_{\sigma\to 0}\rho_u(\sigma) = 0$ since all terms vanish. Moreover, $\lim_{\sigma\to\infty}\rho_u(\sigma)$ is equivalent to the probability of randomly choosing an odd integer, which is clearly $1/2$. Numerical evaluation of $\rho_u(\sigma)$ shows that it monotonically increases between these limits, thus $\rho_u(\sigma)$ is an invertible function.

Theorem 8. *The analog ML estimator for σ is*

$$\hat{\sigma}_A(w_s^A;d) = \rho_u^{-1}\left(w_s^A/m\,;d\right),\ 0<w_s^A<m/2, \quad (4)$$

where $\rho_u^{-1}(x;d)$ is the inverse function of $\rho_u(\sigma;d)$.

Proof: We have $\mathrm{P}\left(w_s^A = w|\sigma\right) = \binom{m}{w}\rho_u(\sigma)^w(1-\rho_u(\sigma))^{m-w}$. Setting the derivative to 0 gives $w = m\rho_u(\sigma)$. ∎

Fig. 1 depicts $\rho_u(\sigma;d)$ and $\hat{\sigma}_A(w_s^A;d)$ for different values of d. For ρ_u, the vanishing and saturation can be observed with regions that significantly depend on d. For $\hat{\sigma}_A$, it can be seen that different effective ranges of σ can be estimated by different values of d. This will be further discussed in Section III-C. Moreover, based on the limiting behavior of $\rho_u(\sigma)$, it is natural to set $\hat{\sigma}_A(0;d) = 0$ and $\hat{\sigma}_A(\ge m/2;d) = \sigma_{\max}(d)$ for some

$$\sigma_{\max}(d) \ge \hat{\sigma}_A(m/2-1;d). \quad (5)$$

C. Estimation Performance

We now wish to study the estimation performance of (4). Like in [10], we rely on the fact that ML estimators are asymptotically efficient [18], that is, they are *asymptotically unbiased* and their asymptotic variance approaches the *Cramer-Rao lower bound* (CRLB) [19]. The CRLB is obtained from the reciprocal of the *Fisher information* of σ, which is derived in the following lemma (proof in Appendix A).

Lemma 9. *The Fisher information of σ with respect to w_s^A is*

$$\mathcal{I}_A(\sigma;d) = \frac{m}{\rho_u(1-\rho_u)}\left[2\sum_{\ell=0}^{\infty}(-1)^\ell\frac{c_\ell}{\sqrt{d}\sigma^2}\phi\left(\frac{c_\ell}{\sqrt{d}\sigma}\right)\right]^2,$$

where $c_\ell \triangleq (\ell+0.5)$.

We notice that $\mathcal{I}_A(\sigma;d)$ includes the well-known $m/[\rho_u(1-\rho_u)]$ term that stems from the Binomial nature of w_s^A. The additional term is induced by the need to further estimate σ from the Binomial probability ρ_u. This term introduces interesting behavior that diverges significantly from the standard Binomial estimation, shown in the following proposition (proof in Appendix B).

Proposition 10. *The Fisher information $\mathcal{I}_A(\sigma;d)$ satisfies the following limiting behavior.*

$$\mathcal{I}_A(\sigma;d) \propto \begin{cases} m(d\sigma^2)^{-2.5}\exp\left\{-\frac{1}{8d\sigma^2}\right\} & ,\sqrt{d}\sigma \ll 1 \\ m/(d\sigma^4) & ,\sqrt{d}\sigma \gg 1. \end{cases}$$

Based on Proposition 10, it can be seen that $\mathcal{I}_A(\sigma;d)$ decays exponentially to zero as $\sigma \to 0$, and decays polynomially to zero as $\sigma \to \infty$. Taking the reciprocal and using CRLB, we conclude that the estimator becomes ill-conditioned (variance diverges) for very small or large values of σ. This happens due to the vanishing or saturation of ρ_u at these regimes, causing a loss of observability on the value of σ. It can also be seen that as d increases, the decay of Fisher information for $\sigma \to 0$ gets slower, whereas the decay for $\sigma \to \infty$ gets faster. This suggests that *as d increases, the estimator becomes*

better suited for smaller values of σ. These observations are apparent in Fig. 1 (right) showing the different ranges of σ "covered" by each d; see further discussion in the next section.

D. Criterion for Good Estimation

Based on the CRLB, $1/\sqrt{\mathcal{I}_A(\sigma;d)}$ is a lower bound on the estimation standard deviation of σ. We can therefore introduce the following criterion of an estimator.

Definition 11. *Let $\alpha \in (0,1)$. An estimator $\hat{\sigma}_A(\sigma;d)$ is said to be α-**accurate** for a value σ_0 if*

$$1/\sqrt{\mathcal{I}_A(\sigma_0;d)} \leq \alpha\sigma_0.$$

Definition 11 is illustrated in Fig. 2, where $1/\sqrt{\mathcal{I}_A(\sigma;d)}$ normalized by σ is shown as a function of σ for different values of d.

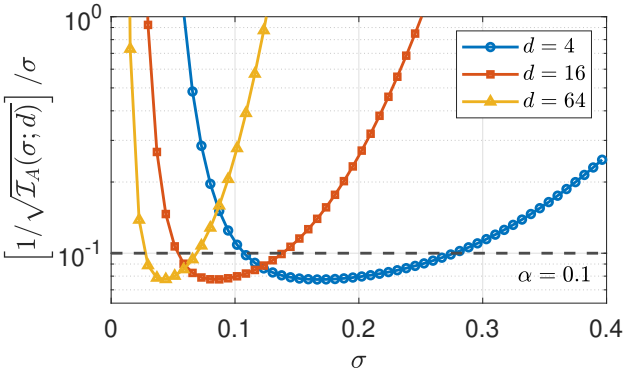

Fig. 2: $1/\sqrt{\mathcal{I}_A(\sigma;d)}$ normalized by σ for different d values, for $m=128$. $\alpha=0.1$ is shown as vertical dashed line.

It can be seen that for $\alpha=0.1$, $d=4,16,64$ provide α-accurate estimators for $\sigma \in (0.03,0.06)$, $\sigma \in (0.055,0.14)$ and $\sigma \in (0.1,0.28)$, respectively.

E. Comparison to Syndromes from Cell-Wise Hard Decision

As a baseline for estimation performance, we examine the case in which we take a hard decision on each cell's state and XOR the results. Recall we use $\Delta G=1$.

Definition 12. *A cell's hard decision is defined by*

$$z_{ij} \triangleq \begin{cases} 0, & G_{ij}-G_{\mathrm{AP}} \leq 0.5 \\ 1, & G_{ij}-G_{\mathrm{AP}} > 0.5 \end{cases}.$$

It is clear from Definition 12 that the probability of a state flip (bit-flip) is $p(\sigma)=Q\left(\frac{0.5}{\sigma}\right)$. We assume $p(\sigma)<0.5$.

Definition 13. *The **logical syndrome** $\boldsymbol{s}^{\mathrm{L}}$ is defined as usual by $s_j^L=\bigoplus_{i=1}^{d} z_{ij}$, where $\oplus$ denotes the logical XOR.*

A set that satisfies the parity constraint from Definition 3 results in $s_j^L=0$ in the noiseless case. As shown in [10], the probability of unsatisfied logical parity constraint is

$$p_u(\sigma)=\frac{1}{2}\left(1-(1-2p(\sigma))^d\right). \tag{6}$$

It is further shown in [10] that the ML estimator for $p(\sigma)$ is $\hat{p}=p_u^{-1}\left(\min\{2w_{\boldsymbol{s}}^L/m,1\}\right)$, where we defined $w_{\boldsymbol{s}}^L \triangleq w_H(\boldsymbol{s}^{\mathrm{L}})$. Since $p(\sigma)$ is monotone increasing, the ML estimator for σ is extracted from $p(\sigma)$.

Lemma 14. *The logical ML estimator for σ is $\hat{\sigma}_L=\frac{0.5}{Q^{-1}(\hat{p})}$, and the Fisher information of σ with respect to $w_{\boldsymbol{s}}^L$ is*

$$\mathcal{I}_L(\sigma;d)=\frac{m}{p_u\left(1-p_u\right)}\left[d(1-2p)^{d-1}\frac{0.5}{\sigma^2}\phi\left(\frac{0.5}{\sigma}\right)\right]^2.$$

Proof: For the ML derivation, see [10] (discussing a similar case). The Fisher information is derived in a similar way to the proof of Lemma 9. ∎

Fig. 3 shows a comparison between inverse square-root Fisher information of analog and logical syndromes, for $d=4,64$. It can be seen that the logical syndrome does not provide observability of low σ values. This is a problem for MVM because a DP error of significant magnitude ($t=3$) occurs with non-negligible probability at low σ values (see vertical line marking the maximal σ for which $\xi_{n,t=3}(\sigma)<10^{-3}$).

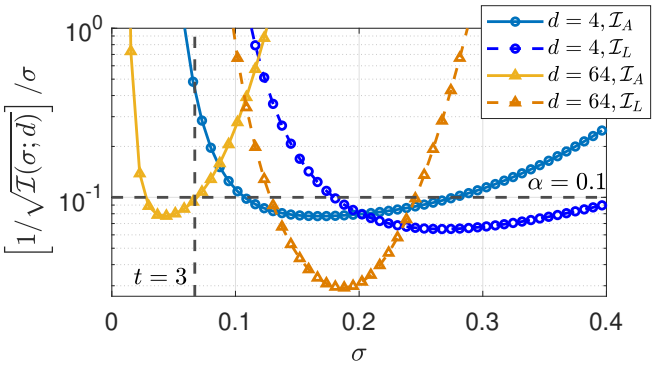

Fig. 3: Comparison between $1/\sqrt{\mathcal{I}_A(\sigma;d)}$ and $1/\sqrt{\mathcal{I}_L(\sigma;d)}$ normalized by σ for $d=4,64$. The maximal σ for which $\xi_{n,3}(\sigma)<10^{-3}$ is shown in the vertical dashed line.

IV. Code Construction and Design

We aim to design a code that will enable both in-memory estimation and out-of-memory error correction. The key idea is to incorporate additional parity rows that are used as follows: (1) During standard MVM operation, the parity rows are not being activated and are transparent, (2) during in-memory noise estimation, a set of information and parity rows, satisfying a parity constraint (Definition 3) in all columns, are activated together, and (3) during error-correction, the entire matrix is read (information and parity rows) and decoding is performed. This is achieved through a bilayer LDPC code with a local-global structure, as introduced in [13], [20].

A. Bilayer Code Construction

To describe the code construction, we consider the $n\times m$ stored matrix C as composed from k *information rows*, that is, the original user-defined binary matrix B, and $r \triangleq n-k$ *parity*

rows. We further consider $r = r_1 + r_2$ such that r_1 is defined as the number of *column-local* parity rows, and r_2 as the number of *matrix-global* parity rows. We use the column-stack vector of length nm of the matrix to define the parity-check matrix structure. We use the following useful terminology for the construction ahead.

Definition 15. *Denote v_B, v_C as the column-stack $1 \times km$ and $1 \times nm$ vectors of the matrices B, C, respectively. A generator matrix U of size $km \times nm$ is said to be in* ***column-systematic form*** *if given $v_C = v_B U$, the first k rows of C equal B.*

Construction 1. Let H_{L} be a parity check matrix of size $r_1 \times (k + r_1)$. Let $\mathbf{0}_{r_1,r_2}$ be the all-zeros matrix of size $r_1 \times r_2$. Let H_{J} be a parity matrix of size $r_2 \times nm$. Let the parity check matrix be defined by

$$H = \begin{bmatrix} [H_{\mathrm{L}}, \mathbf{0}_{r_1,r_2}] & & \\ & \ddots & \\ & & [H_{\mathrm{L}}, \mathbf{0}_{r_1,r_2}] \\ \hline & H_{\mathrm{J}} & \end{bmatrix}. \tag{7}$$

Let U be the generator matrix corresponding to H, in a column-systematic form. Encode an input matrix B to the matrix form of $v_C = v_B U$.

Discussion. *By analyzing the Tanner graph corresponding to H in Eq. (7), we can see that H_{L} provides a sub-graph that connects each column of length k in B to r_1 column-local check nodes. Note that H_{L} is identical for all columns. The additional r_2 parity nodes are connected to variable nodes within different columns through H_{J}, to provide strong correction capability when decoding the full array. Note that the check equations of H_{L} also participate in the full-array decoding, in addition to their role in noise estimation.*

The use of Construction 1 to achieve the estimation and correction mechanism will be further described in Section V.

B. Check Degrees for Good Estimation

The check equations in H_{L} are used for noise estimation, as will be discussed in details in Section V. We therefore wish the check degrees in H_{L} to provide good estimation performance over interesting values of σ.

Definition 16. *Given a check-degree distribution $\mathbf{\Omega}$ for H_{L} define $D_\Omega = \{d : \lfloor r_1 \cdot \Omega_d \rfloor \geq 1\}$ as the* ***set of active degrees****. We assume D_Ω is ordered with increasing degrees.*

Definition 17. *Let $\Sigma = \{\sigma_\nu\}_{\nu=1}^{N}$ with $0 < \sigma_1 < \cdots < \sigma_N$. A set of active degrees D_Ω is said to be α-**accurate for** Σ if*

$$\forall \nu \in [1, \ldots, N] : \exists d \in D_\Omega \ \ s.t \ \ 1/\sqrt{\mathcal{I}_A(\sigma_\nu; d)} \leq \alpha \sigma_\nu.$$

Definition 17 follows directly from Definition 11, while specifically ensuring that the set of check degrees in $\mathbf{\Omega}$ provides estimators that "cover" together the set of σ values in Σ in terms of α-accuracy. For example, in Fig. 2, if $D_\Omega = \{4, 64\}$, any value $\sigma \in (0.07, 0.105)$ would not be covered by $\mathbf{\Omega}$ in the sense that none of the estimators for $d = 4, 64$ provide α-accuracy in this range (with $\alpha = 0.1$). However, for $D_\Omega = \{4, 16, 64\}$, the entire range $\sigma \in (0.03, 0.28)$ is covered. We can now design the check distribution of H_{L}, using Algorithm 1. Since the average check degree a_{R} plays a vital role in coding rate, the design is set to provide a_{R} as close as possible to a desired value $a_{\mathrm{R}}{}^*$.

Algorithm 1: Design of Local Check Degrees

Input : $r_1, a_{\mathrm{R}}{}^*, \Sigma, \alpha$

Do :

- Find $D_\Omega = \{d_1, ..., d_\ell\}$ which is α-accurate for Σ.
- Solve the quadratic integer programming problem:

$$\text{minimize} \quad (\textstyle\sum_{i=1}^{\ell} \beta_i / r_1 \cdot d_i - a_{\mathrm{R}}{}^*)^2$$

$$\text{subject to} \quad 0 < \beta_i \in \mathbb{Z}, \ \sum_{i=1}^{\ell} \beta_i = r_1.$$

Output: Check deg. dist. $\left\{\Omega_{d_i} = \beta_i / r_i\right\}_{i=1}^{\ell}$

C. Degree-Distribution Design

We now describe in high level the stages of designing degree distributions for H_{L} and H_{J}, to provide strong correction capability in decoding. We employ the efficient, low complexity and capacity approaching design technique introduced in [13]. As will be discussed in Section V, the decoding will be performed over the AWGN channel. Therefore, the design will be done for that channel, using EXIT charts [21].

We first set the check-degree distribution of H_{L} using Algorithm 1. We then design the variable-degree distribution of H_{L}, following the steps in [22, Chapter 4.10.1], with a fixed local threshold $\sigma_T^{(L)}$. Then, we design H_{J} as described in [13, Section IV] to meet a global threshold $\sigma_T^{(G)}$. A detailed exploration of this technique is outside the scope of this work.

V. Reliable Dot Product Architecture

In this section we employ the tools from previous sections to design an architecture for reliable in-memory DP operations. The key ingredient in the proposed architecture is to periodically estimate the noise level. If the estimated noise suggests a compromised DP reliability, the matrix is re-written to memory after read and decoding operations. The mechanism is implemented using the following:

1) **Encode:** given a desired matrix $B_{k \times m}$, encode it using the code from Construction 1 to get $C_{n \times m}$. Write C to the memory array using the mapping $0 \to 1, 1 \to -1$.
2) **Compute and Estimate:** perform MVM operations. Once in a predefined number of MVM cycles, perform noise estimation for $\hat{\sigma}_A$ using Algorithm 2. If $\xi_{n,t}(\hat{\sigma}_A) > \xi_{\max}$, for some predefined t, perform 3).
3) **Read, decode and re-write:** read the matrix from memory. Use the belief propagation (BP) algorithm [23] over the log-likelihood ratio values $2G_{ij}/\hat{\sigma}_A$, using the global parity-check matrix H. Use the result $\hat{B}$ and return to 1).

Algorithm 2: In-Memory Noise Estimation

Input : C from memory, parameters H_{L}, D_Ω

Do :

- Set $\hat{\sigma}_{\mathrm{tmp}} = \sigma_{\max}(d_1)$ // Eq. (5)
- For $i \in 1, \ldots, \ell$: // $\ell \triangleq |D_\Omega|$
  - Choose a row $\underline{h}$ in H_{L} with check degree d_i.
  - Set E as the sub-array containing the rows from C for which $\underline{h}$ is 1.
  - Calculate $s_A(E)$ and its weight w_s^A. // Def. 4
  - If $w_s^A \geq m/2$, break
  - If $w_s^A > 0$, update $\hat{\sigma}_{\mathrm{tmp}} \leftarrow \hat{\sigma}_A(w_s^A; d_i)$ // Eq. (4)
- If $i = \ell$ and $w_s^A = 0$, update $\hat{\sigma}_{\mathrm{tmp}} \leftarrow 0$
- Update $\hat{\sigma} \leftarrow \hat{\sigma}_{\mathrm{tmp}}$

Output: Noise estimation $\hat{\sigma}$

Discussion. *Algorithm 2 considers the different ranges of σ that can be accurately estimated for each check degree. It goes through different degrees in D_Ω of H_{L}, and tries to estimate σ in each. If d is too large and σ is in the right divergent region of the estimator, $w_s^A \geq m/2$ is obtained with high probability, and there is no need to further increase d. If d is too small and σ is in the left divergent region of the estimator, $w_s^A = 0$ is obtained with high probability, and d is increased, or if $i = \ell$, $\hat{\sigma} = 0$ is returned. Anytime $0 < w_s^A < m/2$ is obtained, $\hat{\sigma}$ is updated.*

Appendix A
Proof of Lemma 9 - Fisher Information

Proof: Since w_s^A is Binomial with probability $\rho_u(\sigma)$,

$$\frac{\partial \log \mathrm{P}\left(w_s^A = w|\sigma\right)}{\partial \sigma} = \left[\frac{w}{\rho_u} - \frac{m-w}{1-\rho_u}\right]\frac{\partial \rho_u}{\partial \sigma}$$
$$= \frac{w - m\rho_u}{\rho_u(1-\rho_u)} \cdot 2\sum_{\ell=0}^{\infty}(-1)^\ell \frac{\partial}{\partial \sigma} Q\left(\frac{\ell + 0.5}{\sqrt{d}\sigma}\right).$$

We use $\frac{\partial Q(x/\sigma)}{\partial \sigma} = \frac{x}{\sigma^2}\phi\left(\frac{x}{\sigma}\right)$, and by substituting $\ell + 0.5 = c_\ell$, squaring and taking the expectation we get

$$\mathcal{I}_{\mathrm{A}}(\sigma; d) = \frac{\mathbb{E}\left[(w_s^A - m\rho_u)^2|\sigma\right]}{\rho_u^2(1-\rho_u)^2}\left[2\sum_{\ell=0}^{\infty}\frac{(-1)^\ell c_\ell}{\sqrt{d}\sigma^2}\phi\left(\frac{c_\ell}{\sqrt{d}\sigma}\right)\right]^2.$$

Noticing that $\mathbb{E}\left[(w_s^A - m\rho_u)^2|\sigma\right] = \mathbf{var}(w_s^A) = m\rho_u(1-\rho_u)$, we complete the proof. ∎

Appendix B
Proof of Proposition 10 - Limiting Behavior

Proof: We begin with the case where $\sqrt{d}\sigma \ll 1$, where the argument of the $Q(\cdot)$ terms in $\rho_u(\sigma)$ are very large. In this regime $Q(x)$ is well approximated by $\phi(x)/x$. Moreover, the decay of terms in the sum with ℓ is very sharp. Thus, we have

$$\rho_u(\sigma) \approx 2Q\left(\frac{0.5}{\sqrt{d}\sigma}\right) \approx 4\sqrt{d}\sigma \cdot \phi\left(\frac{0.5}{\sqrt{d}\sigma}\right).$$

From the same considerations of decaying terms, we have

$$\left[2\sum_{\ell=0}^{\infty}\frac{(-1)^\ell c_\ell}{\sqrt{d}\sigma^2}\phi\left(\frac{c_\ell}{\sqrt{d}\sigma}\right)\right]^2 \approx \left[\frac{1}{\sqrt{d}\sigma^2}\phi\left(\frac{0.5}{\sqrt{d}\sigma}\right)\right]^2.$$

Plugging into $\mathcal{I}_A(\sigma; d)$ provides the desired expression.

Now, for $\sqrt{d}\sigma \gg 1$, we have $\rho_u(\sigma) \approx 1/2$ (as discussed after Lemma 7), and the sum can be expressed as $(c/\sqrt{d}\sigma^2)^2$, with $c = 2\sum_{\ell=0}^{\infty}(-1)^\ell c_\ell \phi(c_\ell/\sqrt{d}\sigma) < \infty$ (due to the decaying exponential terms in summation). ∎

References

[1] Y. Chen, Y. Xie, L. Song, F. Chen, and T. Tang, "A survey of accelerator architectures for deep neural networks," *Engineering, vol. 6, no. 3*, 2020.

[2] S. A. McKee, "Reflections on the memory wall," *Conference on Computing Frontiers*, 2004.

[3] A. Sebastian, M. Le-Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature Nanotechnology, Vol. 15*, 2020.

[4] Y. Li, T. Bai, X. Xu, Y. Zhang, B. Wu, H. Cai, B. Pan, and W. Zhao, "A survey of MRAM-centric computing: From near memory to in memory," *IEEE Trans. on Emerging Topics in Computing, vol. 11, no. 2*, 2023.

[5] P. Deaville, B. Zhang, and N. Verma, "A fully row/column-parallel MRAM in-memory computing macro with memory-resistance boosting and weighted multi-column ADC readout," *IEEE Journal of Solid-State Circuits*, 2024.

[6] M. Suri, A. Gupta, V. Parmar, and K. H. Lee, "Performance enhancement of edge-ai-inference using commodity MRAM: IoT case study," in *2019 IEEE 11th International Memory Workshop (IMW)*, 2019.

[7] X. Yang, Y. Mao, L. Chang, H. Wei, Y. Wang, J. Wang, C. Fan, Z. Wu, S. Peng, and J. Zhou, "Edge-optimized ai architecture: MRAM-based near memory computing macro balancing between memory capacity and computation," *Integrated Circuits and Systems*, 2025.

[8] R. M. Roth, "Fault-tolerant dot-product engines," *IEEE Trans. on Information Theory, vol. 65, no. 4*, 2019.

[9] ——, "Analog error-correcting codes," *IEEE Trans. on Information Theory, vol. 66, no. 7*, 2020.

[10] G. Lechner and C. Pacher, "Estimating channel parameters from the syndrome of a linear code," *IEEE Communications Letters, Vol. 17, No. 17*, 2013.

[11] Y. Gershon and Y. Cassuto, "In-memory BER estimation using syndromes of LDPC codes," *2025 IEEE Int. Symp. on Information Theory (ISIT), to Appear*, 2025.

[12] R. Gallager, "Low-density parity-check codes," *IRE Trans. on Information Theory, Vol. 8, No. 1*, 1962.

[13] E. Ram and Y. Cassuto, "Design of bilayer and multi-layer LDPC ensembles from individual degree distributions, vol. 67, no. 11," *IEEE Trans. on Information Theory*, 2021.

[14] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. on Information Theory, vol. 27, no. 5*, 1981.

[15] E. Dupraz, F. Leduc-Primeau, K. Cai, and L. Dolecek, "Turning to information theory to bring in-memory computing into practice," *IEEE BITS the Information Theory Magazine, Vol.3, No. 3*, 2023.

[16] L. Tan and J. Jiang, *Digital signal processing: fundamentals and applications*. Academic press, 2018.

[17] R. J. van de Plassche, *Integrated Analog-to-Digital and Digital-to-Analog Converters*. Springer Science & Business Media, 2012.

[18] G. Casella and R. L. Berger, *Statistical Inference*. Thomson Learning, 2002.

[19] F. Nielsen, *Cramér-Rao Lower Bound and Information Geometry*. Hindustan Book Agency, Gurgaon, 2013.

[20] E. Ram and Y. Cassuto, "LDPC codes with local and global decoding," in *2018 IEEE Int. Symp. on Information Theory (ISIT)*, 2018.

[21] S. ten Brink, G. Kramer, and A. Ashikhmin, "Design of low-density parity-check codes for modulation and detection," *IEEE Trans. on Communications, vol. 52, no. 4*, 2004.

[22] T. Richardson and R. Urbanke, *Modern coding theory*. Cambridge university press, 2008.

[23] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. on Information Theory, vol. 47, no. 2*, 2001.