SPLiT: Channel-Aware Learning for Quantum Error Mitigation

Anonymous authors
Paper under double-blind review

Abstract

Quantum error mitigation (QEM) improves observable estimates on noisy quantum devices. Circuit-specific estimation and explicit channel inversion incur additional execution, characterization, or sampling costs. Direct learned regressors enable fast inference on new circuits but leave noise accumulation implicit, which can impair accuracy under depth extrapolation and device drift. We propose **SPLiT** (Sparse Pauli–Lindblad Transformer), which reformulates learned mitigation around explicit layerwise noise channels. A causal Transformer predicts channel rates, from which we construct a survival prior that accumulates analytically across layers. Observable-conditioned residuals refine this shared prior into deterministic mitigation coefficients for individual Pauli terms. A differentiable qubit-transfer dynamic program evaluates the prior exactly for open-chain channels with per-layer cost linear in qubit count. Relative-position attention supports depth extrapolation, while calibration and measurement conditioning adapt predictions to noise drift without retraining. Across a simulation sweep of four circuit families spanning structured and random circuits under three noise models, SPLiT achieves the lowest mean absolute error (MAE) in 10 of 12 settings, with 1.3–2.3$\times$ lower MAE on multi-Pauli energy tasks than the strongest baseline in each setting. It also achieves the lowest reported MAE under temporal drift, beyond training depths, and on a 100-qubit benchmark.

1 Introduction

Quantum computing promises computational advantages in quantum chemistry and materials science [1; 2], combinatorial optimization [3; 4], machine learning [5], and cryptography [6]. Realizing these advantages at scale is expected to require fault-tolerant machines, in which logical qubits are encoded across many physical qubits, and logical error rates decrease with increasing code distance when physical error rates are below threshold [7; 8; 9]. Although surface codes have recently been operated below threshold [10], the physical-qubit overhead of large-scale logical encoding still exceeds what present hardware can afford, and the processors of today and the near future therefore execute algorithms directly on noisy physical qubits [11]. On such processors, gate errors, decoherence, and readout errors can bias observable estimates and reduce the precision attainable by quantum algorithms [12]. Quantum error mitigation (QEM) reduces noise-induced bias through classical post-processing, using additional circuit executions, calibration data, or reference data [13; 14; 12]. By improving observable estimates without the qubit overhead of logical encoding, QEM provides a practical approach to using noisy hardware for these applications.

Existing QEM methods can be grouped by their mitigation resources and the reuse of these resources across target circuits. **Circuit-specific estimation** derives a mitigated estimate from measurements or reference circuits tailored to each target. Zero-noise extrapolation (ZNE) estimates the zero-noise limit from measurements at amplified noise levels [13; 14], while Clifford data regression (CDR) fits an affine map on efficiently simulable near-Clifford references [15]. Accuracy depends on reliable noise scaling and the extrapolation model for ZNE, and on reference representativeness for CDR; both require additional circuit executions for each target [12]. **Explicit channel inversion** recovers ideal expectations through a characterized noise model. Probabilistic error cancellation (PEC) yields unbiased estimates through quasiprobability sampling when the channel is accurately characterized [13; 16]. Sparse Pauli–Lindblad (SPL) models compactly represent local correlated noise using Pauli-generator rates fitted to characterization data [17; 18]. PEC's sampling overhead grows exponentially with accumulated noise [19], and maintaining an accurate channel under device

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

drift requires updated characterization [20]. **Amortized regression** learns mappings from circuit features and noisy measurements to mitigated expectations. ML-QEM uses simulated or mitigation-generated targets and demonstrates hardware experiments up to 100 qubits [21]. DAEM trains on fiducial processes, without noise-free data from the target process [22]. These methods reuse training across targets. Graph and multi-range models encode circuit topology and long-range context [23; 24]. ML-QEM also shows noise-dependent depth extrapolation and drift adaptation through fine-tuning [21]. We focus on explicit layerwise noise accumulation to support depth generalization and calibration-conditioned adaptation.

To combine channel structure with amortized inference, we propose SPLiT (Sparse Pauli–Lindblad Transformer), which combines explicit noise modeling with amortized mitigation (Fig. 1). We learn layerwise SPL channels and accumulate their survival prior for deterministic mitigation at deployment. To capture term-dependent responses and channel mismatch that the shared prior leaves unexplained, we design an observable-conditioned residual that refines individual Pauli-term mitigation coefficients while retaining shared layerwise channels. To address the depth extrapolation challenge of direct regression, we combine analytic noise accumulation with relative-position attention [25]. To adapt to device drift without retraining, we condition predictions on calibration inputs and deployment-time measurements. In summary, our contributions are as follows:

(1) We reformulate learning-based QEM around an explicit layerwise SPL representation, replacing direct observable regression with mitigation coefficients derived from an analytically accumulated survival prior shared across Pauli terms and observable-specific residuals.

(2) We design SPLiT, a causal Transformer that jointly predicts layerwise SPL rates and term-specific residuals, integrating relative-position attention, logarithmic gate-error conditioning, and measurement-driven feature-wise linear modulation (FiLM).

(3) We develop a differentiable qubit-transfer dynamic program that exactly evaluates the survival prior for the open-chain SPL model, with per-layer computational cost linear in the number of qubits.

(4) Experiments across four circuit families and three noise settings show that SPLiT achieves the lowest MAE in 10 of 12 standard settings, 1.3–2.3× lower MAE on multi-Pauli energy tasks than the strongest baseline in each setting, and the lowest reported MAE under drift, depth extrapolation, and at 100 qubits.

2 PRELIMINARIES

Circuits and states. For an n-qubit, L-layer circuit $\mathcal{C}$, index layers by $l \in \{1, \dots, L\}$ and qubits by $q \in \{1, \dots, n\}$. The ideal and noisy states after layer l, ρ_l and ρ_l^{noisy}, start from $\rho_0 = \rho_0^{\text{noisy}} = (|0\rangle\langle 0|)^{\otimes n}$ and evolve under the ideal unitary U_l and effective physical noise channel $\mathcal{N}_l$ as

$$\rho_l = U_l \rho_{l-1} U_l^\dagger, \qquad \rho_l^{\text{noisy}} = \mathcal{N}_l\left(U_l \rho_{l-1}^{\text{noisy}} U_l^\dagger\right). \tag{1}$$

Here $\mathcal{N}_l$ is a completely positive, trace-preserving map acting after the ideal layer [26]; ρ_l^{noisy} precedes readout error and finite-shot sampling.

Pauli observables and outputs. Write a Hermitian observable as a sum of N_{term} Pauli terms,

$$O = \sum_{i=1}^{N_{\text{term}}} a_i O_i, \qquad O_i = \bigotimes_{q=1}^{n} O_{i,q}, \qquad O_{i,q} \in \{I, X, Y, Z\}, \quad a_i \in \mathbb{R}, \tag{2}$$

where i indexes terms and I, X, Y, Z denote the single-qubit identity and Pauli operators. Subscripts in X_q, Y_q, Z_q specify the acted-on qubit, with identity elsewhere. The ideal expectation is

$$y_l(O) = \text{Tr}(O\rho_l) = \sum_{i=1}^{N_{\text{term}}} a_i y_l(O_i). \tag{3}$$

The noisy estimate $y_l^{\text{noisy}}(O)$ includes readout and sampling effects when present; $\hat{y}_l(O)$ is the mitigated estimate and $y_l(O)$ the ideal training label. This notation also applies to each O_i. A single Pauli observable has $N_{\text{term}} = 1$ and $a_1 = 1$.

Pauli commutation. For Pauli strings P, Q on the same qubits, let $\chi(P, Q) = \mathbf{1}[PQ = -QP]$ be their anticommutation indicator [27], where $\mathbf{1}[\cdot]$ is one when its condition holds and zero otherwise.

Sparse Pauli–Lindblad channels. The sparse Pauli–Lindblad (SPL) model [18] uses K local Pauli generators $P_k \in \{I, X, Y, Z\}^{\otimes n}$, $k \in \{1, \ldots, K\}$, with few nonidentity factors, distinct from observable terms O_i. For any density matrix ρ, define the conjugation and identity channels $\mathcal{P}_k(\rho) = P_k \rho P_k$ and $\mathcal{I}(\rho) = \rho$. The layer-l model channel is

$$\Lambda_l = \exp\left[\sum_{k=1}^{K} \lambda_{l,k}(\mathcal{P}_k - \mathcal{I})\right], \qquad \boldsymbol{\lambda}_l = (\lambda_{l,1}, \ldots, \lambda_{l,K}) \in \mathbb{R}^K_{\geq 0}, \tag{4}$$

where the nonnegative rates include layer duration and Λ_l models local stochastic contributions to $\mathcal{N}_l$. Each generator applies Pauli conjugation with probability $\frac{1}{2}(1 - e^{-2\lambda_{l,k}})$ and identity otherwise; their composition is a mixture of Pauli errors [18]. Pauli strings are eigenoperators of this channel:

$$\Lambda_l(Q) = \exp\left[-2\sum_{k=1}^{K} \lambda_{l,k}\chi(Q, P_k)\right] Q. \tag{5}$$

Only generators anticommuting with Q attenuate its Pauli component. The identity coefficient $c_I(\boldsymbol{\lambda}_l)$ sums probabilities of net identity errors up to phase, including nonempty combinations that cancel. Its eigenvalue average (eq. (15)) underlies the mitigation survival prior.

3 METHOD

Overview. We propose SPLiT, a learning-based framework for channel-aware quantum error mitigation. A causal Transformer processes encoded circuit layers and noisy prefix measurements to predict nonnegative SPL rates. A differentiable decomposition of the predicted channels yields a cumulative survival prior, which is combined with an observable-conditioned residual to mitigate individual Pauli expectations. To support depth extrapolation, we adopt ALiBi relative-position biases [25]. To accommodate temporal noise drift, we introduce logarithmic gate-error conditioning and use FiLM [28] to incorporate auxiliary single-qubit measurements. The resulting model is trained end to end.

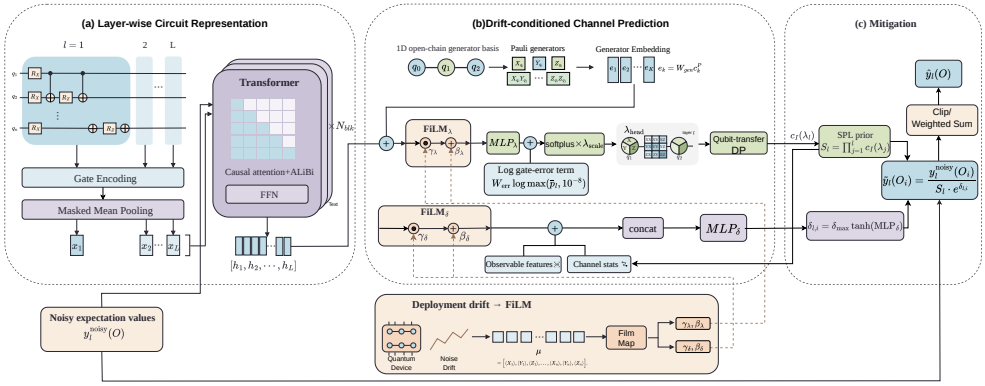


Fig. 1: Overview of SPLiT, combining causal SPL-rate prediction, a survival prior from qubit-transfer DP, and observable-conditioned residuals for mitigation. FiLM and gate-error conditioning support adaptation to noise drift.

3.1 PROBLEM REFORMULATION

Direct regression. Let $\mathcal{C}_{1:l}$ denote the circuit prefix through layer l, including supplied gate metadata. Direct learned-QEM regressors predict ideal expectations from circuit features and noisy measurements [21]. Under our measurement-access model, we define a direct regressor using this prefix, the observable O, and the available noisy measurement history:

$$f_{\text{reg}} : \quad \left(\mathcal{C}_{1:l},\ O,\ \left(y_j^{\text{noisy}}(O)\right)_{j=1}^{l}\right) \longmapsto \hat{y}_l(O). \tag{6}$$

Channel-based reformulation. For the base model, SPLiT maps the same inputs to layerwise SPL rates and observable-conditioned log-residuals:

$$f_{\text{SPLiT}} : \quad \left(\mathcal{C}_{1:l},\ O,\ \left(y_j^{\text{noisy}}(O)\right)_{j=1}^{l}\right) \longmapsto \left(\{\boldsymbol{\lambda}_j\}_{j=1}^{l},\ \{\delta_{l,i}\}_{i=1}^{N_{\text{term}}}\right). \tag{7}$$

The rates $\boldsymbol{\lambda}_j$ parameterize the channels Λ_j in eq. (4), while $\delta_{l,i}$ models the term-dependent response and channel mismatch. Decomposing the predicted channels provides a shared survival prior; the readout combines this prior, the residuals, and the noisy term measurements $y_l^{\text{noisy}}(O_i)$ to obtain $\hat{y}_l(O)$, as detailed in Sec. 3.4. For Hamiltonians, the input history contains aggregate noisy energies $y_j^{\text{noisy}}(O)$, so the predicted channels are shared across terms within this measurement context. Both formulations use the ideal expectation $y_l(O)$ as the supervised target for $\hat{y}_l(O)$.

3.2 Features and Encoders

We construct one backbone token per circuit layer, with $g \in \{1, \dots, G\}$ indexing the G padded gate slots within each layer.

Features. *Circuit.* Gate inputs comprise categorical encodings $\mathbf{c}_{l,g}$, $\mathbf{q}_{l,g}$, and $\boldsymbol{\pi}_{l,g}$ for gate type, acting qubits, and ordered qubit pair, respectively. The numerical inputs are a four-dimensional Fourier encoding $\boldsymbol{\varphi}(\theta_{l,g})$ of the rotation angle $\theta_{l,g}$ and available rescaled gate metadata $\widetilde{\mathbf{s}}_{l,g}$ (duration, supplied gate-error probability $p_{l,g}$, and T_1, T_2). Each layer supplies the noisy measurement $y_l^{\text{noisy}}(O)$. With binary mask $m_{l,g}^G$ marking valid gate slots, the mean supplied gate-error probability over valid gates is

$$\bar{p}_l = \frac{\sum_{g=1}^{G} m_{l,g}^G p_{l,g}}{\max(1, \sum_{g=1}^{G} m_{l,g}^G)}. \tag{8}$$

Observables. We encode each factor $O_{i,q}$ and its position q by one-hot vectors $\mathbf{c}_{i,q}^O \in \mathbb{R}^4$ and $\mathbf{q}_q^O \in \mathbb{R}^n$.

Measurement context. We collect single-qubit Pauli expectations of the final noisy state into the measurement context

$$\boldsymbol{\mu} = \left[\mathrm{Tr}(X_q \rho_L^{\text{noisy}}), \mathrm{Tr}(Y_q \rho_L^{\text{noisy}}), \mathrm{Tr}(Z_q \rho_L^{\text{noisy}})\right]_{q=1}^{n} \in \mathbb{R}^{3n}, \tag{9}$$

These measurements reflect both the output state and noise.

Encoders. We use $\mathbf{W}$ for learnable matrices, W for learnable scalar weights, and m for masks, with indices and superscripts identifying their roles. We denote concatenation by $[\cdot;\cdot]$ and layer normalization by LN [29]. All multilayer perceptrons (MLPs) use two affine layers with hidden width d and SiLU between them; MLP_λ and MLP_δ additionally apply LN to their input. The gate embedding $\mathbf{z}_{l,g} \in \mathbb{R}^d$ is

$$\mathbf{z}_{l,g} = \mathrm{MLP}_{\text{gate}}\left(\mathrm{LN}\begin{pmatrix}\mathbf{W}_{\text{type}}\mathbf{c}_{l,g} + \mathbf{W}_{\text{qubit}}\mathbf{q}_{l,g} + \mathbf{W}_{\text{pair}}\boldsymbol{\pi}_{l,g} \\ + \mathrm{MLP}_{\text{num}}([\boldsymbol{\varphi}(\theta_{l,g}); \widetilde{\mathbf{s}}_{l,g}])\end{pmatrix}\right), \tag{10}$$

We pool the gate embeddings and add the projected noisy measurement to form the layer token:

$$\mathbf{x}_l = \frac{\sum_{g=1}^{G} m_{l,g}^G \mathbf{z}_{l,g}}{\max(1, \sum_{g=1}^{G} m_{l,g}^G)} + \mathrm{MLP}_y\left(y_l^{\text{noisy}}(O)\right), \mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_L]^\top \in \mathbb{R}^{L \times d}, \tag{11}$$

The observable encoder uses projections $\mathbf{W}_{\text{Pauli}} \in \mathbb{R}^{d \times 4}$ and $\mathbf{W}_{\text{qubit}}^O \in \mathbb{R}^{d \times n}$ for Pauli factors and positions. Its soft mask $m_{i,q}^O$ is 1 for $O_{i,q} \neq I$ and 0.1 otherwise. The encoding $\mathbf{o}_i \in \mathbb{R}^d$ includes the nonidentity support fraction:

$$\mathbf{o}_i = \mathrm{MLP}_O\left(\mathrm{LN}\left(\begin{array}{c}\dfrac{\sum_{q=1}^{n} m_{i,q}^O [\mathbf{W}_{\text{Pauli}}\mathbf{c}_{i,q}^O + \mathbf{W}_{\text{qubit}}^O \mathbf{q}_q^O]}{\max(1, \sum_{q=1}^{n} m_{i,q}^O)} \\ + \mathrm{MLP}_{\text{supp}}\left(\frac{1}{n}\sum_{q=1}^{n} \mathbf{1}[O_{i,q} \neq I]\right)\end{array}\right)\right). \tag{12}$$

The Transformer receives $\mathbf{X}$; the residual head also receives $\mathbf{o}_i$.

3.3 Causal Transformer and Rate Prediction

Causal backbone. A pre-normalized causal Transformer [30] processes $\mathbf{X}$ through N_{blk} blocks, indexed by b. The binary layer mask m_l^L marks valid circuit layers in padded batches. Initialize

$\mathbf{h}_l^{(0)} = \mathbf{x}_l$. With LN, multi-head attention (MHA), and a feed-forward network (FFN), the unmodulated block recursion is

$$\begin{aligned} \widetilde{\mathbf{h}}_l^{(b)} &= \mathbf{h}_l^{(b-1)} + \mathrm{MHA}_b\Big(\mathrm{LN}(\mathbf{h}_l^{(b-1)}); \{\mathrm{LN}(\mathbf{h}_{l'}^{(b-1)})\}_{l' \le l}\Big), \\ \mathbf{h}_l^{(b)} &= \widetilde{\mathbf{h}}_l^{(b)} + \mathrm{FFN}_b\Big(\mathrm{LN}(\widetilde{\mathbf{h}}_l^{(b)})\Big), \qquad b = 1, \dots, N_{\mathrm{blk}}. \end{aligned} \tag{13}$$

The FFN uses GELU activation, and each LN instance has its own parameters. Both heads consume the final representation $\mathbf{h}_l = \mathrm{LN}(\mathbf{h}_l^{(N_{\mathrm{blk}})})$. Attention includes only valid preceding layers ($l' \le l$, $m_{l'}^L = 1$); relative positional bias is specified in Sec. 3.5. The backbone computes all depth outputs in one forward pass.

λ-head. The matrix $\mathbf{W}_{\mathrm{gen}} \in \mathbb{R}^{d \times K}$ embeds the one-hot generator code $\mathbf{c}_k^P \in \mathbb{R}^K$. With optional FiLM and gate-error conditioning (Sec. 3.6), the rate head computes

$$\lambda_{l,k} = \lambda_{\mathrm{scale}}\ \mathrm{softplus}\left(\begin{array}{c} \mathrm{MLP}_\lambda\big(\mathrm{FiLM}_\lambda(\mathbf{h}_l + \mathbf{W}_{\mathrm{gen}}\mathbf{c}_k^P; \boldsymbol{\mu})\big) \\ + W_{\mathrm{err}} \log\max(\bar{p}_l, \varepsilon_p) \end{array}\right), \tag{14}$$

where $\lambda_{\mathrm{scale}} > 0$ is fixed and $\varepsilon_p > 0$ is a fixed gate-error probability floor.

3.4 Channel-aware Mitigation

Qubit-transfer DP. For each predicted layer channel, Pauli orthogonality expresses the identity coefficient as the average of the eigenvalues in eq. (5) [18]:

$$c_I(\boldsymbol{\lambda}_l) = 4^{-n} \sum_{Q \in \{I,X,Y,Z\}^{\otimes n}} \exp\left[-2 \sum_{k=1}^{K} \lambda_{l,k} \chi(Q, P_k)\right]. \tag{15}$$

Direct evaluation involves 4^n Pauli configurations and costs $\mathcal{O}(K4^n)$ per layer.

We parameterize the predicted SPL channel with single-qubit and nearest-neighbor two-qubit generators on an open chain, giving $K = 3n + 9(n-1)$. This local structure enables exact identity-coefficient evaluation by qubit-transfer DP. Writing $Q = \bigotimes_{q=1}^n s_q$, with $s_q \in \{I, X, Y, Z\}$, each summand separates into site factors $u_{l,q}$ and nearest-neighbor factors $B_{l,q}$:

$$c_I(\boldsymbol{\lambda}_l) = \sum_{s_1,\dots,s_n \in \{I,X,Y,Z\}} \prod_{q=1}^{n} u_{l,q}(s_q) \prod_{q=1}^{n-1} B_{l,q}(s_q, s_{q+1}). \tag{16}$$

Write $(P_k)_q$ for the factor of P_k at qubit q and $\mathrm{supp}(P_k)$ for its nonidentity qubits. For $s, s' \in \{I, X, Y, Z\}$, the site and edge factors are

$$\begin{aligned} u_{l,q}(s) &= \frac{1}{4} \exp\left[-2 \sum_{k:\mathrm{supp}(P_k)=\{q\}} \lambda_{l,k} \chi(s, (P_k)_q)\right], \\ B_{l,q}(s, s') &= \exp\left[-2 \sum_{k:\mathrm{supp}(P_k)=\{q,q+1\}} \lambda_{l,k} \chi(s \otimes s', (P_k)_q \otimes (P_k)_{q+1})\right]. \end{aligned} \tag{17}$$

Let $v_{l,q}(s)$ denote the partial sum of the product of site factors $u_{l,1}, \dots, u_{l,q}$ and bond factors $B_{l,1}, \dots, B_{l,q-1}$, over $s_1, \dots, s_{q-1}$ with $s_q = s$ fixed. For $q < n$, the only factor crossing the cut after site q is $B_{l,q}(s_q, s_{q+1})$, so these four boundary weights suffice for exact sum-product elimination on the open chain [31]. Algorithm 1 evaluates eq. (16) by this recursion.

Algorithm 1 Qubit-transfer DP for one circuit layer

Require: Rates $\boldsymbol{\lambda}_l$ for the complete open-chain basis
Ensure: Identity coefficient $c_I(\boldsymbol{\lambda}_l)$
1: Construct $u_{l,q}$ and $B_{l,q}$ using eq. (17).
2: $v_{l,1}(s) \leftarrow u_{l,1}(s)$ for all $s \in \{I, X, Y, Z\}$.
3: **for** $q = 1, \ldots, n-1$ **do**
4: $\quad v_{l,q+1}(s') \leftarrow u_{l,q+1}(s') \sum_{s\in\{I,X,Y,Z\}} v_{l,q}(s) B_{l,q}(s, s')$ for all $s' \in \{I, X, Y, Z\}$.
5: **end for**
6: **return** $c_I(\boldsymbol{\lambda}_l) = \sum_{s\in\{I,X,Y,Z\}} v_{l,n}(s)$.

Each DP step uses a 4×4 update. Including local factor construction, the per-layer cost is $\mathcal{O}(n)$, and the computation is differentiable in the predicted rates.

The layer coefficients computed by the DP give the cumulative survival prior:

$$S_l = \prod_{j=1}^{l} c_I(\boldsymbol{\lambda}_j). \tag{18}$$

Under the predicted SPL channels, S_l is the weight of the branch with net identity error at every layer through depth l. For a fixed target O, it provides a shared reference scale for mitigation across its terms O_i. Evaluating all layer coefficients and their prefix products costs $\mathcal{O}(Ln)$.

Observable-conditioned residual. The survival prior S_l supplies a shared reference scale, while nonidentity error branches contribute differently to each observable. The residual captures these term-dependent contributions and channel mismatch using $\mathbf{h}_l$, the observable embedding $\mathbf{o}_i$, and four physical summary statistics:

$$\bar{\lambda}_l = \frac{1}{K}\sum_{k=1}^{K} \lambda_{l,k}, \boldsymbol{\sigma}_l = \left[\bar{\lambda}_l,\ \max_k \lambda_{l,k},\ \frac{\sum_{j=1}^{l} m_j^L \bar{\lambda}_j}{\max(1, \sum_{j=1}^{l} m_j^L)},\ S_l\right]. \tag{19}$$

For the base and head-only variants, the residual input is

$$\mathbf{u}_{l,i} = [\mathrm{FiLM}_\delta(\mathbf{h}_l; \boldsymbol{\mu}); \mathbf{o}_i; \mathrm{MLP}_{\mathrm{stats}}(\boldsymbol{\sigma}_l)] \in \mathbb{R}^{3d}. \tag{20}$$

The dense variant appends a context embedding as specified in Sec. A.2. With fixed bound $\delta_{\max} > 0$, the head predicts

$$\delta_{l,i} = \delta_{\max} \tanh(\mathrm{MLP}_\delta(\mathbf{u}_{l,i})). \tag{21}$$

Mitigation readout. For a fixed numerical floor $\varepsilon > 0$, the stabilized denominator and unclipped mitigated term are

$$D^{\varepsilon}_{l,i} = \max(S_l, \varepsilon) e^{\delta_{l,i}}, \qquad \widetilde{y}_l(O_i) = \frac{y_l^{\mathrm{noisy}}(O_i)}{\max(D^{\varepsilon}_{l,i}, \varepsilon)}. \tag{22}$$

The final estimate clips a single-Pauli output to its physical range and uses unclipped terms for Hamiltonian aggregation:

$$\hat{y}_l(O) = \begin{cases} \mathrm{clip}_{[-1,1]}(\widetilde{y}_l(O_1)), & \text{single-Pauli mode,} \\ \sum_{i=1}^{N_{\mathrm{term}}} a_i \widetilde{y}_l(O_i), & \text{Hamiltonian mode.} \end{cases} \tag{23}$$

3.5 Depth Extrapolation

The survival prior already carries the error scale as a function of depth, and the causal backbone can process additional tokens without a fixed depth lookup table. In each Transformer block, let $\mathbf{q}_l^{(h)}, \mathbf{k}_l^{(h)}, \mathbf{v}_l^{(h)} \in \mathbb{R}^{d_h}$ be attention head h's learned query, key, and value projections of the normalized hidden states, with head dimension d_h. Following ALiBi [25] and suppressing the block index, the attention weights and head output are

$$\alpha_{l,l'}^{(h)} = \mathrm{softmax}_{l'}\left(\frac{(\mathbf{q}_l^{(h)})^\top \mathbf{k}_{l'}^{(h)}}{\sqrt{d_h}} - \kappa_h(l - l') + m_{l,l'}^{\mathrm{attn}}\right), \mathbf{u}_l^{(h)} = \sum_{l'=1}^{L} \alpha_{l,l'}^{(h)} \mathbf{v}_{l'}^{(h)}. \tag{24}$$

Here softmax is over key positions, $\kappa_h > 0$ is a fixed head-specific slope, and the additive mask $m^{\text{attn}}_{l,l'}$ is 0 when $l' \le l$ and $m^L_{l'} = 1$, and $-\infty$ otherwise (Sec. 3.3). MHA concatenates the head outputs and applies a learned output projection. The relative-position bias $-\kappa_h(l - l')$ remains well defined beyond training depths.

3.6 Temporal Noise Drift Adaptation

To accommodate deployment-time noise shifts, we combine measurement-based FiLM with a log gate-error term. Both use fixed trained parameters at inference.

FiLM-based adaptation. Using the measurement context $\boldsymbol{\mu}$ from eq. (9), FiLM [28] modulates a feature vector $\mathbf{h} \in \mathbb{R}^d$ as

$$\text{FiLM}(\mathbf{h}; \boldsymbol{\mu}) = \boldsymbol{\gamma}(\boldsymbol{\mu}) \odot \mathbf{h} + \boldsymbol{\beta}(\boldsymbol{\mu}), \tag{25}$$

where $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are learned scale and shift maps, and $\odot$ denotes elementwise multiplication. Disabled FiLM maps act as the identity, as in the base model. Independent FiLM maps adapt the rate and residual heads using the same $\boldsymbol{\mu}$ at every layer. The head-only and dense placement variants are detailed in Sec. A.2.

Log gate-error term. The log gate-error term in eq. (14) uses $\bar{p}_l$ from eq. (8). The coefficient W_{err} is a learned scalar shared across layers and generators. When the total logit is sufficiently negative and the probability floor is inactive, the rate scales approximately as $\bar{p}_l^{W_{\text{err}}}$ with the MLP output held fixed. This supplies a noise-strength scaling bias: updated gate-error inputs adjust the channel's noise scale, while FiLM adapts rate and residual predictions to the measurement context.

3.7 Training Objective

We train SPLiT end to end using the stabilized readout in eqs. (22) and (23) and minimize

$$\mathcal{L} = \mathcal{L}_{\text{mit}} + \alpha_{\text{den}}\mathcal{L}_{\text{den}} + \alpha_{L1}\mathcal{L}_{L1}, \tag{26}$$

where $\mathcal{L}_{\text{mit}}$ is the mean squared error between mitigated predictions $\hat{y}_l(O)$ and ideal targets $y_l(O)$ over valid layers. The denominator loss $\mathcal{L}_{\text{den}}$ is the mean squared error between $D^{\varepsilon}_{l,i}$ and the noisy-to-ideal ratio $y_l^{\text{noisy}}(O_i)/y_l(O_i)$ over valid layer–term pairs, excluding near-zero ideal labels and extreme ratios. For Hamiltonians, mitigation supervision acts on total energy, whereas denominator supervision acts on individual Pauli terms. The L1 penalty $\mathcal{L}_{L1}$ averages the nonnegative predicted rates to discourage unnecessary generator strength. Nonnegative coefficients α_{den} and α_{L1} weight the two auxiliary terms. Loss weights and training settings are given in Sec. A.6.

4 Experiment

4.1 Experimental Setup

Benchmark circuits. We evaluate four circuit families: Trotterized dynamics [32] of the transverse-field Ising model (TFIM) [33], random circuits, the quantum approximate optimization algorithm (QAOA) for MaxCut [3], and Fermi–Hubbard circuits [34]. Tab. 3 in Sec. A.1 lists their sizes and observables.

Noise models. Each family uses three noise configurations: (1) *Incoherent*: Pauli–Lindblad noise derived from T_1/T_2 relaxation and gate durations [27], consistent with the model assumptions; (2) *Coherent*: over-rotation coherent errors of amplitude 0.01 superimposed on the incoherent noise, with the Clifford-twirled channel [35] approximated using a finite average over 64 randomly sampled twirls; (3) *Provider*: per-qubit heterogeneous T_1/T_2 noise from the Qiskit [36] fake backend FakeHanoiV2; simulator rates follow this noise map, while gate scalar features remain uniform.

Datasets and splits. Each (circuit, noise) cell contains 800 training, 100 validation, and 300 test sequences, randomly truncated to cover depths 1–$L_{\max}$ ($L_{\max} = 20$, or 18 for QAOA); see Sec. A.6 for the full protocol.

Baselines. Six baselines span statistical regression, noise extrapolation, and neural prediction: *Random forest (RF)* [37; 21], *Ordinary least squares (OLS)* [38; 21], *Zero-noise extrapolation (ZNE)* [13; 14], *Clifford data regression (CDR)* [15], *Multilayer perceptron (MLP)* [38; 21] and *Graph neural network (GNN)* [21].

Metrics. Following QEM-Bench [24], we measure accuracy by absolute error (AE), mean absolute error (MAE), and root mean squared error (RMSE), defined in Sec. A.7.

Implementation and training protocol. Training follows Sec. 3.7 (stabilized denominator, survival prior), selecting checkpoints by validation MAE. Tables give means and available standard deviations; Sec. A.6 gives hyperparameters and training details.

4.2 Main Results: Mitigation Accuracy

Tabs. 1 and 2 report MAE for TFIM/Random/QAOA and Hubbard, respectively, under three noise configurations, with methods in rows. SPLiT leads in 10 of 12 settings and ranks second on TFIM with incoherent and coherent noise. MAEs use original units and four decimals; red/blue mark best/second-best by unrounded means. Tabs. 5 and 6 give RMSE; Figs. 2 and 5 show selected and remaining standard per-sequence error distributions.

Tab. 1: Per-sequence MAE on TFIM, Random, and QAOA (mean ± std in original units, four decimals; lower is better). Inc./Coh./Prov.: incoherent/coherent/provider noise. Red/blue: best/second-best from unrounded means, excluding Noisy.

Method	TFIM			Random			QAOA		
	Inc.	Coh.	Prov.	Inc.	Coh.	Prov.	Inc.	Coh.	Prov.
Noisy	0.4210 ± 0.0050	0.4260 ± 0.0050	0.2960 ± 0.0030	0.0520 ± 0.0010	0.0510 ± 0.0030	0.0200 ± 0.0004	0.5210 ± 0.0320	0.5410 ± 0.0240	0.3570 ± 0.0190
RF	0.0305 ± 0.0018	0.0308 ± 0.0022	0.0271 ± 0.0012	0.00456 ± 0.00013	0.00517 ± 0.00038	0.00360 ± 0.00011	0.2100 ± 0.0090	0.2280 ± 0.0080	0.2010 ± 0.0080
OLS	0.1830 ± 0.0050	0.1880 ± 0.0050	0.1160 ± 0.0020	0.0098 ± 0.0003	0.0104 ± 0.0006	0.00568 ± 0.00010	0.3310 ± 0.0180	0.3600 ± 0.0080	0.2710 ± 0.0120
ZNE	0.3890 ± 0.0050	0.3950 ± 0.0050	0.2340 ± 0.0030	0.0449 ± 0.0008	0.0446 ± 0.0025	0.0201 ± 0.0004	0.3890 ± 0.0230	0.4190 ± 0.0180	0.2430 ± 0.0120
CDR	0.1580 ± 0.0050	0.1610 ± 0.0060	0.1020 ± 0.0040	0.00710 ± 0.00030	0.00789 ± 0.00037	0.0152 ± 0.0004	0.3130 ± 0.0160	0.3780 ± 0.0140	0.2530 ± 0.0130
MLP	0.0395 ± 0.0037	0.0399 ± 0.0038	0.0323 ± 0.0011	0.00527 ± 0.00009	0.00607 ± 0.00049	0.00365 ± 0.00008	0.2030 ± 0.0070	0.2260 ± 0.0090	0.2010 ± 0.0030
GNN	0.0576 ± 0.0087	0.0550 ± 0.0040	0.0473 ± 0.0038	0.0233 ± 0.0020	0.0241 ± 0.0016	0.0207 ± 0.0010	0.2410 ± 0.0120	0.2490 ± 0.0100	0.2260 ± 0.0140
SPLiT	0.0320 ± 0.0015	0.0397 ± 0.0032	0.0245 ± 0.0014	0.00259 ± 0.00021	0.00422 ± 0.00022	0.00244 ± 0.00012	0.1260 ± 0.0050	0.1760 ± 0.0100	0.1120 ± 0.0030

Multi-Pauli energy benchmarks. SPLiT leads all six QAOA/Hubbard settings, reducing MAE by 1.3×–2.3× over the strongest baselines: RF on Hubbard and MLP on QAOA. Under incoherent, coherent, and provider noise, reductions are 2.3×, 2.0×, and 2.1× on Hubbard, and 1.6×, 1.3×, and 1.8× on QAOA, respectively.

Comparison with neural baselines. MLP and GNN improve noisy multi-Pauli estimates; SPLiT further lowers errors in every QAOA/Hubbard setting. It shares a learned channel across terms, rescales each by its own denominator, and combines them into an energy estimate.

Single-Pauli benchmarks. SPLiT leads four of six TFIM/Random settings. On TFIM under incoherent and coherent noise, RF attains MAEs of 0.0305 and 0.0308, versus 0.0320 and 0.0397 for SPLiT, respectively (Tab. 1).

4.3 Challenging Settings

SPLiT leads all six columns of Tab. 2: three standard Hubbard settings, Hubbard temporal drift, TFIM depth extrapolation, and the 100-qubit TFIM Hamiltonian. RF is the strongest baseline on standard Hubbard and 100 qubits; MLP leads baselines under drift and depth extrapolation. Fig. 3 plots depth and noise-strength extrapolation.

Tab. 2: MAE on Hubbard and three challenging settings (mean ± std, four decimals). The drift magnitude is $\sigma = 0.05$; std is unavailable for 100 qubits. Noise labels and colors follow Tab. 1.

Method	Hubbard			Time drift	Depth extrap.	100 qubits
	Inc.	Coh.	Prov.	Hubbard	TFIM	TFIM
Noisy	1.0710 ± 0.0370	1.0800 ± 0.0150	0.7270 ± 0.0280	0.9988 ± 0.0181	0.4006 ± 0.0065	14.5350
RF	0.1360 ± 0.0030	0.1410 ± 0.0070	0.1340 ± 0.0070	0.2905 ± 0.0075	0.0398 ± 0.0004	1.5430
OLS	0.3990 ± 0.0080	0.4330 ± 0.0110	0.3310 ± 0.0060	0.4557 ± 0.0060	0.1771 ± 0.0026	15.5500
ZNE	0.8030 ± 0.0300	0.8250 ± 0.0110	0.4310 ± 0.0180	0.7630 ± 0.0160	0.3668 ± 0.0063	11.5200
CDR	0.4500 ± 0.0160	0.5080 ± 0.0140	0.3770 ± 0.0160	0.4380 ± 0.0155	0.1635 ± 0.0032	–
MLP	0.1440 ± 0.0050	0.1480 ± 0.0030	0.1530 ± 0.0050	0.2494 ± 0.0144	0.0389 ± 0.0027	2.3290
GNN	0.1800 ± 0.0090	0.1830 ± 0.0040	0.1600 ± 0.0050	0.2684 ± 0.0049	0.0553 ± 0.0014	2.3340
SPLiT	0.0605 ± 0.0090	0.0707 ± 0.0061	0.0625 ± 0.0029	0.1390 ± 0.0127	0.0209 ± 0.0016	1.4610

Depth extrapolation. Models train on TFIM depths 1–15 and are evaluated up to 30. SPLiT has the lowest MAE throughout unseen depths 16–30 (Fig. 3(a)). Sec. A.4 separately ablates positional encoding to assess ALiBi.

100-qubit Hamiltonian. On 100-qubit TFIM, SPLiT attains MAE 1.4610 versus RF's 1.5430 (5.3% lower), and the lowest reported RMSE, 3.611 versus the best baseline GNN's 4.097 (Tab. 6). Fig. 2 (rightmost) shows its per-sequence error distribution.

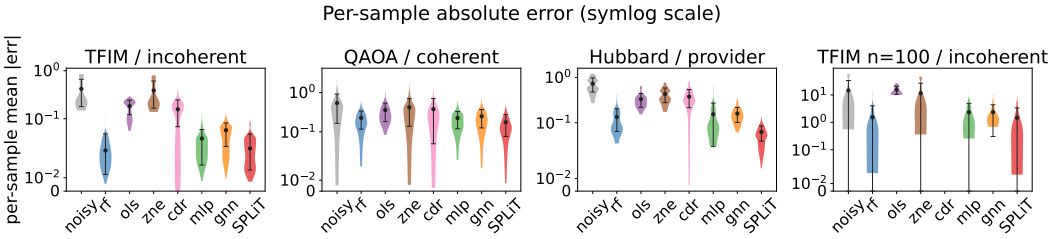


Fig. 2: Per-sequence mean absolute-error distributions for TFIM/incoherent, QAOA/coherent, Hubbard/provider, and the 100-qubit TFIM Hamiltonian (left to right). The remaining nine standard settings appear in Fig. 5.

Noise-strength extrapolation. On Hubbard, training uses noise multipliers $M \leq 2$ and testing uses $M = 3, 4$ (Fig. 3(b)). At $M = 4$, SPLiT has MAE 0.516 ± 0.072, below all competing curves, versus 1.720 ± 0.046 unmitigated.

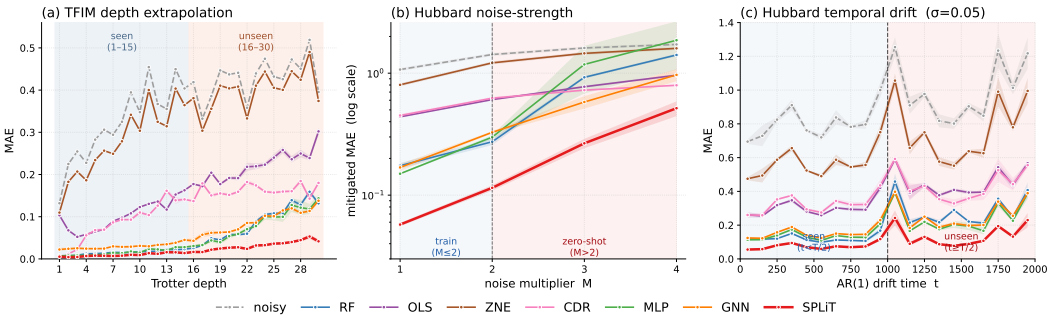


Fig. 3: MAE under (a) TFIM depth extrapolation, (b) Hubbard noise-strength extrapolation, and (c) Hubbard temporal drift ($\sigma = 0.05$). Blue and pink backgrounds mark the training and held-out ranges.

4.4 Temporal Calibration Drift

On three-site Hubbard, T_1, T_2 follow an AR(1) trajectory with $\sigma = 0.05$, split into a training first half and evaluation second half; Sec. A.1 specifies drift and calibration inputs. SPLiT maintains the lowest held-out MAE, including higher-error intervals (Fig. 3(c)): 0.1390 versus 0.2494 for the strongest baseline MLP (44.3% lower) and 0.9988 for Noisy (Tab. 2).

5 Conclusion

We presented SPLiT, a channel-aware Transformer that combines layerwise SPL rate prediction, a differentiable survival prior, and observable-conditioned residual refinement. Qubit-transfer dynamic programming makes the prior efficient to compute for local open-chain channels. Relative-position attention supports depth extrapolation, while measurement and calibration conditioning support adaptation to temporal noise drift. Across four circuit families and three noise settings, SPLiT achieves the lowest MAE in 10 of 12 standard settings, with further advantages under temporal drift, beyond training depths, and on a 100-qubit TFIM benchmark. These results demonstrate the value of combining structured channel modeling with learned observable-specific rescaling.

AI USE STATEMENT

In this work, we used generative AI tools for drafting assistance and editing support during manuscript preparation. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work develops learning-based quantum error mitigation methods and evaluates them on simulated quantum circuits with synthetic and hardware-inspired noise models. It involves no human subjects, personal data or hardware experiments, and we do not foresee direct ethical concerns; any societal impact is indirect, through progress towards more reliable quantum computation.

REPRODUCIBILITY STATEMENT

The circuit families, noise models, and dataset splits are described in Sec. 4.1, with benchmark sizes and calibration-drift settings in Sec. A.1. The architecture and training objective are specified in Secs. 3.3, 3.4 and 3.7; input features, optional conditioning paths, and the exact solver are detailed in Secs. A.2, A.3 and 3.2. Model hyperparameters and training protocols are summarized in Sec. A.6, and metric definitions and aggregation conventions are given in Sec. A.7. The temporal split and measurement-noise construction for the drift benchmark are specified in Sec. A.1. Baseline identities and checkpoint selection are described in Sec. 4.1; the result-table captions state the available standard deviations and the ranking convention.

REFERENCES

[1] Sam McArdle, Suguru Endo, Alán Aspuru-Guzik, Simon C. Benjamin, and Xiao Yuan. Quantum computational chemistry. *Reviews of Modern Physics*, 92:015003, 2020.

[2] Bela Bauer, Sergey Bravyi, Mario Motta, and Garnet Kin-Lic Chan. Quantum algorithms for quantum chemistry and quantum materials science. *Chemical Reviews*, 120(22):12685–12717, 2020.

[3] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[4] Amira Abbas, Andris Ambainis, Brandon Augustino, et al. Challenges and opportunities in quantum optimization. *Nature Reviews Physics*, 6:718–735, 2024.

[5] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549:195–202, 2017.

[6] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997.

[7] Peter W. Shor. Scheme for reducing decoherence in quantum computer memory. *Physical Review A*, 52:R2493–R2496, 1995.

[8] Barbara M. Terhal. Quantum error correction for quantum memories. *Reviews of Modern Physics*, 87:307–346, 2015.

[9] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. Surface codes: Towards practical large-scale quantum computation. *Physical Review A*, 86:032324, 2012.

[10] Google Quantum AI and Collaborators. Quantum error correction below the surface code threshold. *Nature*, 638:920–926, 2025.

[11] John Preskill. Quantum computing in the NISQ era and beyond. *Quantum*, 2:79, 2018.

[12] Zhenyu Cai, Ryan Babbush, Simon C. Benjamin, Suguru Endo, William J. Huggins, Ying Li, Jarrod R. McClean, and Thomas E. O'Brien. Quantum error mitigation. *Reviews of Modern Physics*, 95:045005, 2023.

[13] Kristan Temme, Sergey Bravyi, and Jay M. Gambetta. Error mitigation for short-depth quantum circuits. *Physical Review Letters*, 119:180509, 2017.

[14] Ying Li and Simon C. Benjamin. Efficient variational quantum simulator incorporating active error minimization. *Physical Review X*, 7:021050, 2017.

[15] Piotr Czarnik, Andrew Arrasmith, Patrick J. Coles, and Lukasz Cincio. Error mitigation with Clifford quantum-circuit data. *Quantum*, 5:592, 2021.

[16] Suguru Endo, Simon C. Benjamin, and Ying Li. Practical quantum error mitigation for near-future applications. *Physical Review X*, 8:031027, 2018.

[17] G. Lindblad. On the generators of quantum dynamical semigroups. *Communications in Mathematical Physics*, 48:119–130, 1976.

[18] Ewout van den Berg, Zlatko K. Minev, Abhinav Kandala, and Kristan Temme. Probabilistic error cancellation with sparse pauli-lindblad models on noisy quantum processors. *Nature Physics*, 19(8):1116–1121, 2023.

[19] Ryuji Takagi, Suguru Endo, Shintaro Minagawa, and Mile Gu. Fundamental limits of quantum error mitigation. *npj Quantum Information*, 8:114, 2022.

[20] Youngseok Kim, Luke C. G. Govia, Andrew Dane, Ewout van den Berg, David M. Zajac, Bradley Mitchell, Yinyu Liu, Karthik Balakrishnan, George Keefe, Adam Stabile, Emily Pritchett, Jiri Stehlik, and Abhinav Kandala. Error mitigation with stabilized noise in superconducting quantum processors. *Nature Communications*, 16:8439, 2025.

[21] Haoran Liao, Derek S. Wang, Iskandar Sitdikov, Ciro Salcedo, Alireza Seif, and Zlatko K. Minev. Machine learning for practical quantum error mitigation. *Nature Machine Intelligence*, 6(12):1478–1486, 2024.

[22] Manwen Liao, Yan Zhu, Giulio Chiribella, and Yuxiang Yang. Noise-agnostic quantum error mitigation with data augmented neural models. *npj Quantum Information*, 11:8, 2025.

[23] Tianyi Bao, Xinyu Ye, Hang Ruan, Chang Liu, Wenjie Wu, and Junchi Yan. Beyond circuit connections: A non-message passing graph transformer approach for quantum error mitigation. In *International Conference on Learning Representations*, 2025.

[24] Tianyi Bao, Ruizhe Zhong, Xinyu Ye, Yehui Tang, and Junchi Yan. QEM-bench: Benchmarking learning-based quantum error mitigation and QEMFormer as a multi-ranged context learning baseline. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 2953–2967. PMLR, 2025.

[25] Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations (ICLR)*, 2022.

[26] John Watrous. *The Theory of Quantum Information*. Cambridge University Press, 2018.

[27] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition, 2010.

[28] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 3942–3951, 2018.

[29] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

[31] Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001. doi: 10.1109/18.910572.

[32] H. F. Trotter. On the product of semi-groups of operators. *Proceedings of the American Mathematical Society*, 10:545–551, 1959.

[33] Subir Sachdev. *Quantum Phase Transitions*. Cambridge University Press, second edition, 2011.

[34] J. Hubbard. Electron correlations in narrow energy bands. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 276:238–257, 1963.

[35] Christoph Dankert, Richard Cleve, Joseph Emerson, and Etera Livine. Exact and approximate unitary 2-designs and their application to fidelity estimation. *Physical Review A*, 80:012304, 2009.

[36] Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J. Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D. Nation, Lev S. Bishop, Andrew W. Cross, Blake R. Johnson, and Jay M. Gambetta. Quantum computing with Qiskit. *arXiv preprint arXiv:2405.08810*, 2024.

[37] Leo Breiman. Random forests. *Machine Learning*, 45:5–32, 2001.

[38] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, second edition, 2009.

[39] George E. P. Box, Gwilym M. Jenkins, and Gregory C. Reinsel. *Time Series Analysis: Forecasting and Control*. Wiley, fourth edition, 2008.

[40] X. Bonet-Monroig, R. Sagastizabal, M. Singh, and T. E. O'Brien. Low-cost error mitigation by symmetry verification. *Physical Review A*, 98(6):062339, 2018. doi: 10.1103/PhysRevA.98.062339. URL https://doi.org/10.1103/PhysRevA.98.062339.

[41] Sam McArdle, Xiao Yuan, and Simon Benjamin. Error-mitigated digital quantum simulation. *Physical Review Letters*, 122(18):180501, 2019. doi: 10.1103/PhysRevLett.122.180501. URL https://doi.org/10.1103/PhysRevLett.122.180501.

[42] Armands Strikis, Dayue Qin, Yanzhu Chen, Simon C. Benjamin, and Ying Li. Learning-based quantum error mitigation. *PRX Quantum*, 2:040330, 2021.

A Appendix

A.1 Benchmark Settings

Tab. 3 lists the sizes and observables of the four standard circuit families and the additional 100-qubit TFIM Hamiltonian benchmark. Tab. 4 collects the temporal calibration-drift settings described in Sec. 4.4.

Tab. 3: Benchmark circuit families.

Circuit family	Size	Observable
TFIM	$n = 6$, depth 1–20	$Z^{\otimes n}$
Random	$n = 6$, depth 1–20	$Z^{\otimes n}$
QAOA-MaxCut	$n = 6$, depth 1–18	MaxCut cost function
Fermi-Hubbard	3 sites, depth 1–20; interaction/hopping $= 4$	Hamiltonian energy
TFIM (100 qubits)	$n = 100$	TFIM Hamiltonian energy

Tab. 4: Temporal calibration-drift settings for three-site Hubbard.

Setting	Value	Setting	Value
Base T_1	$23.24\,\mu$s	Base T_2	$15.6\,\mu$s
Gate duration τ_{1q}	18 ns	Gate duration τ_{2q}	48 ns
Noise scale s_{noise}	3.0	Readout error	3.8%
Trajectory length T	2000	AR(1) persistence ϕ_{AR}	0.99
Drift magnitude σ	0.05	Measurement noise σ_{meas}	0.1
Floor on r_t	0.05	Train/test split	First/second half
Training sequences	900	Test sequences	300

Drift process and calibration input. The single-qubit Pauli error probabilities associated with a gate of duration $\tau \in \{\tau_{1q}, \tau_{2q}\}$ follow the amplitude-damping/dephasing model [27], with global noise scale s_{noise} multiplying the probabilities:

$$p_X = p_Y = \frac{s_{\text{noise}}}{4}(1 - e^{-\tau/T_1}), \qquad p_Z = \frac{s_{\text{noise}}}{2}(1 - e^{-\tau/T_2}) - p_X. \tag{27}$$

The base parameters, global noise scale, and readout error are listed in Tab. 4. A shared log-domain AR(1) factor [39], denoted r_t at time step t, scales both decoherence times,

$$T_1(t) = T_1 r_t, \quad T_2(t) = T_2 r_t, \qquad \log r_t = \phi_{\text{AR}} \log r_{t-1} + \sigma\omega_t, \quad \omega_t \sim \mathcal{N}(0,1), \quad r_0 = 1, \tag{28}$$

with a floor $r_t \geq 0.05$. At each time step, the physical noise and supervision targets are recomputed using $T_1(t), T_2(t)$ in eq. (27). For gate slot (l, g), let $p^{\text{phys}}_{l,g}(t)$ be the resulting physical error probability. Its calibration estimate supplies the gate-error input from Sec. 3.2:

$$p_{l,g}(t) = p^{\text{phys}}_{l,g}(t) e^{\nu_t}, \qquad \nu_t \sim \mathcal{N}(0, \sigma^2_{\text{meas}}). \tag{29}$$

The time argument is suppressed elsewhere for a fixed calibration context. The raw T_1, T_2 input features remain fixed at their base values. Times $t < T/2$ form the training/calibration split, and $t \geq T/2$ the held-out deployment split. The comparison includes Noisy, all six baselines, and SPLiT, as shown in Fig. 3(c).

A.2 Input Features and Configuration

The inference inputs and encoders are defined in Sec. 3.2. Gate and layer masks remove padding; ideal expectations are used only for supervision.

FiLM configurations. The head-only variant uses the conditioning in eqs. (14) and (20). The dense variant additionally constructs $\mathbf{h}_c = \text{MLP}_c(\boldsymbol{\mu}) \in \mathbb{R}^d$, modulates every Transformer block output with its own FiLM map conditioned on $\mathbf{h}_c$, and appends $\mathbf{h}_c$ to the residual input, giving $\mathbf{u}_{l,i} \in \mathbb{R}^{4d}$ instead of $\mathbb{R}^{3d}$. In simulations, $\boldsymbol{\mu}$ is computed before the separate readout-error model. Prefix consistency requires fixed prefix measurements and measurement context; recomputing $\boldsymbol{\mu}$ for a truncated circuit can change the hidden states.

Calibration availability. Noise-strength extrapolation enables both gate metadata and measurement context. The log gate-error term uses unscaled gate-error probabilities, with W_{err} initialized to 1. Provider-noise metadata retain the base configuration and do not reveal per-qubit heterogeneity. Temporal calibration inputs follow Sec. A.1.

A.3 Qubit-transfer Dynamic Program

The local weights and chain factorization are defined in eqs. (16) and (17). Using the local Pauli factors from eq. (16), the generator supports give

$$\chi(Q, P_k) = \begin{cases} \chi(s_q, (P_k)_q), & \mathrm{supp}(P_k) = \{q\}, \\ \chi(s_q \otimes s_{q+1}, (P_k)_q \otimes (P_k)_{q+1}), & \mathrm{supp}(P_k) = \{q, q+1\}. \end{cases} \tag{30}$$

Substituting these identities into eq. (15) separates the exponential into site and edge factors. Distributing 4^{-n} as one factor of $1/4$ per site yields eq. (16).

The update in Alg. 1 multiplies the four accumulated boundary weights by a matrix with entries $B_{l,q}(s, s')u_{l,q+1}(s')$. Because the remaining factors depend on the processed sites only through their boundary label, this elimination preserves all contributions for arbitrary nonnegative, generator-specific rates in the stated basis. The batched implementation materializes the local site and edge factors, so its storage grows linearly with chain length.

The DP contracts the auxiliary Pauli sum in eq. (15). Exactness refers to the identity coefficient of the specified SPL model. Products of its local error generators can yield higher-weight Pauli errors, whose contributions are retained by this calculation.

The implementation requires the complete open-chain basis and rejects nonadjacent or missing generator supports; it does not implement periodic wrap edges or a general graph contraction.

A.4 Conditioning and Position Ablations

Noise-strength conditioning. Fig. 4(a) compares the full model with variants that remove FiLM or the log gate-error term on TFIM with the Z_{last} observable. The horizontal axis is the noise multiplier M, with $M \leq 2$ used for training and $M = 3, 4$ held out. At both held-out multipliers, the full model has the lowest MAE; removing the log gate-error term causes a larger increase than removing FiLM. The full model's performance under temporal drift is evaluated in Fig. 3(c).

Positional encoding. Fig. 4(b) compares learned positional embeddings with ALiBi on TFIM, using depths 1–15 for training and 16–30 for evaluation. Both variants yield small errors at seen depths. Beyond the training range, the learned variant degrades to approximately 0.70–0.76 MAE at depths 29–30, exceeding Noisy there, while ALiBi remains below 0.06 throughout the unseen range. This supports using relative positional bias for depth transfer.

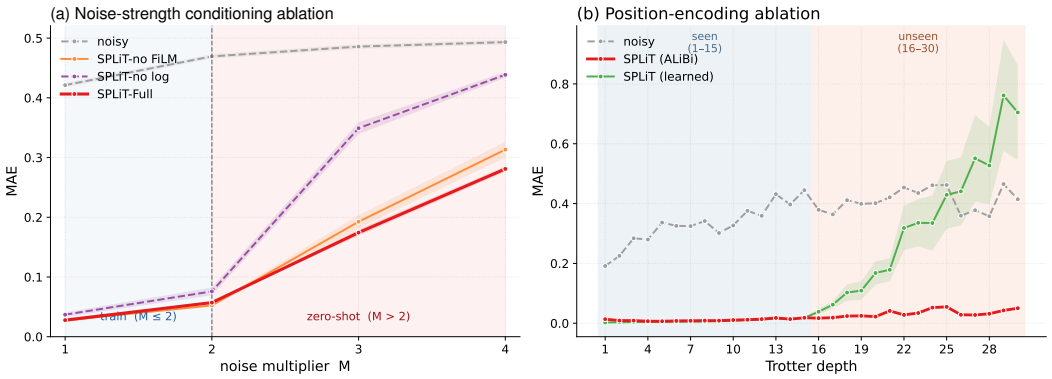


Fig. 4: (a) FiLM and log-channel ablations across noise strength (train $M \leq 2$; held out $M = 3, 4$). (b) Positional-encoding ablation on TFIM (train depths 1–15; held out 16–30). Shading denotes the standard deviation across three random seeds.

A.5 Additional RMSE Results

Tab. 5 and Tab. 6 report RMSE for the twelve standard settings and 100-qubit TFIM. MAE is used for the main comparisons in Tabs. 1 and 2 and Fig. 3.

Tab. 5: RMSE on TFIM and Random (mean ± std, lower is better). Noise labels and ranking colors follow Tab. 1.

Method	TFIM			Random		
	Inc.	Coh.	Prov.	Inc.	Coh.	Prov.
Noisy	0.521 ± 0.007	0.526 ± 0.007	0.373 ± 0.004	0.115 ± 0.001	0.115 ± 0.004	0.0441 ± 0.0006
RF	0.0548 ± 0.0064	0.0559 ± 0.0070	0.0449 ± 0.0029	0.0106 ± 0.0006	0.0124 ± 0.0014	0.00915 ± 0.00041
OLS	0.240 ± 0.005	0.245 ± 0.005	0.157 ± 0.002	0.0214 ± 0.0003	0.0231 ± 0.0016	0.0134 ± 0.0004
ZNE	0.486 ± 0.007	0.493 ± 0.007	0.301 ± 0.003	0.102 ± 0.001	0.102 ± 0.004	0.0459 ± 0.0006
CDR	0.204 ± 0.005	0.209 ± 0.006	0.158 ± 0.004	0.0156 ± 0.0008	0.0174 ± 0.0009	0.0326 ± 0.0007
MLP	0.0660 ± 0.0074	0.0635 ± 0.0045	0.0527 ± 0.0013	0.0129 ± 0.0006	0.0145 ± 0.0004	0.00916 ± 0.00052
GNN	0.0864 ± 0.0107	0.0850 ± 0.0050	0.0683 ± 0.0051	0.0427 ± 0.0040	0.0466 ± 0.0037	0.0374 ± 0.0024
SPLiT	0.0464 ± 0.0029	0.0587 ± 0.0046	0.0392 ± 0.0005	0.00791 ± 0.00070	0.0111 ± 0.0008	0.00706 ± 0.00013

Tab. 6: RMSE on QAOA, Hubbard, and 100-qubit TFIM (mean ± std, lower is better). Noise labels and ranking colors follow Tab. 1; dashes denote unavailable results.

Method	QAOA			Hubbard			100 qubits
	Inc.	Coh.	Prov.	Inc.	Coh.	Prov.	TFIM
Noisy	0.756 ± 0.046	0.783 ± 0.027	0.534 ± 0.026	1.324 ± 0.042	1.352 ± 0.017	0.950 ± 0.033	26.408
RF	0.324 ± 0.012	0.344 ± 0.010	0.309 ± 0.009	0.220 ± 0.005	0.231 ± 0.011	0.220 ± 0.008	4.148
OLS	0.490 ± 0.024	0.521 ± 0.015	0.401 ± 0.015	0.560 ± 0.009	0.607 ± 0.018	0.456 ± 0.006	18.748
ZNE	0.577 ± 0.033	0.620 ± 0.023	0.374 ± 0.015	1.021 ± 0.034	1.064 ± 0.014	0.595 ± 0.022	21.455
CDR	0.463 ± 0.022	0.594 ± 0.031	0.378 ± 0.020	0.595 ± 0.019	0.691 ± 0.016	0.516 ± 0.020	–
MLP	0.317 ± 0.007	0.343 ± 0.014	0.314 ± 0.007	0.244 ± 0.004	0.258 ± 0.020	0.250 ± 0.013	4.775
GNN	0.367 ± 0.024	0.363 ± 0.002	0.341 ± 0.025	0.256 ± 0.010	0.270 ± 0.016	0.226 ± 0.007	4.097
SPLiT	0.197 ± 0.008	0.272 ± 0.024	0.179 ± 0.003	0.0941 ± 0.0137	0.111 ± 0.013	0.0948 ± 0.0041	3.611

A.6 Experimental Details

Datasets and splits. Each (circuit, noise) cell generates 800 training, 100 validation, and 300 test sequences of depth 1–$L_{\max}$ ($L_{\max} = 20$, or 18 for QAOA), with each sequence truncated at a random intermediate depth to cover the depth distribution.

Baselines. We compare against six baselines covering statistical regression, noise extrapolation, and neural prediction: *Random forest (RF)* [37; 21], *Ordinary least squares (OLS)* [38; 21], *Zero-noise extrapolation (ZNE)* [13; 14], *Clifford data regression (CDR)* [15], *Multilayer perceptron (MLP)* [38; 21] and *Graph neural network (GNN)* [21].

Model hyperparameters. Unless otherwise stated, we use $d = 96$, $N_{\text{blk}} = 3$ Transformer blocks, 4 attention heads, $\lambda_{\text{scale}} = 0.05$, $\delta_{\max} = 3.0$, and $\varepsilon = 10^{-3}$. The reference survival prior configuration uses the stabilized denominator in eq. (22).

Objective settings. For eq. (26), the main survival prior recipe uses $\alpha_{\text{den}} = 0.2$ and $\alpha_{L1} = 10^{-4}$. TFIM and Random use single-Pauli readout, whereas QAOA and Hubbard use Hamiltonian readout.

Implementation and training protocol. Models are trained for 30 epochs. Training follows the recipe of Sec. 3.7 (stabilized denominator, survival prior), with checkpoint selection by validation MAE. Tables report means with the available standard deviations; the aggregation conventions are defined in Sec. A.7.

A.7 Evaluation Metrics

We evaluate prediction accuracy using absolute error (AE), mean absolute error (MAE), and root mean squared error (RMSE), as in QEM-Bench [24].

Let $\mathcal{E}$ be a collection of N_{eval} evaluated circuit–observable–depth triples $(\mathcal{C}, O, l)$. We use the output notation from Sec. 2, evaluated for the corresponding circuit $\mathcal{C}$. Each target is a Pauli expectation or an aggregated Hamiltonian energy. The error metrics are

$$\text{AE}(\mathcal{C}, O, l) = |\hat{y}_l(O) - y_l(O)|, \qquad \text{MAE}(\mathcal{E}) = \frac{1}{N_{\text{eval}}} \sum_{(\mathcal{C},O,l)\in\mathcal{E}} \text{AE}(\mathcal{C}, O, l), \tag{31}$$

$$\text{RMSE}(\mathcal{E}) = \sqrt{\frac{1}{N_{\text{eval}}} \sum_{(\mathcal{C},O,l)\in\mathcal{E}} \text{AE}(\mathcal{C}, O, l)^2}. \tag{32}$$

The Noisy reference replaces $\hat{y}_l(O)$ with $y_l^{\text{noisy}}(O)$ in these expressions. For a per-sequence MAE, $\mathcal{E}$ contains that sequence's evaluated depths. The standard-benchmark tables report the mean and standard deviation across these sequence-level MAEs; the violin plots show their distributions.

A.8 Absolute-error Distributions

Fig. 5 shows the distributions of per-sequence mean absolute errors for NOISY, RF, OLS, MLP, GNN, ZNE, CDR, and SPLiT in the nine standard settings complementing the first three panels of Fig. 2. Together, the two figures cover all twelve standard circuit–noise settings; Fig. 2 also includes the 100-qubit TFIM benchmark. Each plotted sample is one circuit sequence, represented by its mean absolute error over evaluated depths; lower distributions indicate more accurate mitigation.

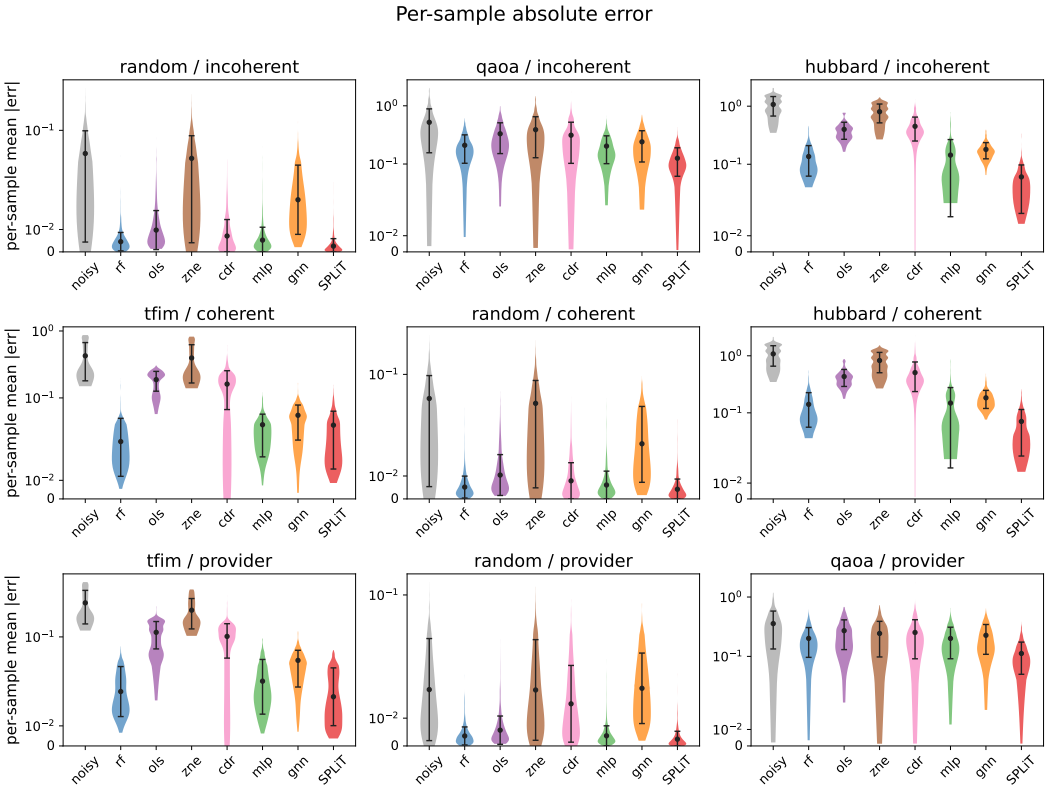


Fig. 5: Per-sequence mean absolute-error distributions for nine additional standard circuit–noise settings. The other three standard settings appear in Fig. 2.

A.9 Related Work

Noise extrapolation and probabilistic cancellation. ZNE estimates the zero-noise limit by measuring logically equivalent circuits at several effective noise strengths and extrapolating their observable estimates [13; 14]. Its accuracy depends on the noise-scaling procedure and extrapolation model. PEC represents ideal operations through signed linear combinations of implementable noisy operations and forms a weighted estimator from sampled circuits [13; 16]. This estimator is unbiased

when the representation exactly reproduces the ideal operations. SPL-based PEC parameterizes the noise channel as the exponential of a Lindblad generator with a sparse set of local Pauli terms and learns their nonnegative rates from characterization data [18]. This compact parameterization supports scalable noise learning and inverse-channel sampling; its accuracy depends on how closely the fitted model matches the deployed noise. Sampling overhead remains noise dependent. For layered circuits with local depolarizing noise, worst-case bounds exhibit exponential depth dependence in the sampling resources required for fixed accuracy [19].

Symmetry verification. SV uses a known target symmetry sector to suppress contributions from sector-changing errors. It can be implemented through symmetry measurements and postselection, or through equivalent projection in classical post-processing [40; 41]. Errors within the accepted sector can remain, and the sampling cost of postselection grows as the acceptance probability decreases.

Regression and learned sampling. RF averages decision-tree predictions, while OLS fits a linear predictor by minimizing squared residuals. In ML-QEM, both use encoded circuit and observable features together with noisy expectations; device-specific noise parameters are optional inputs [21]. CDR fits an affine map to paired noisy and classically computed expectations from tractable near-Clifford variants of the target circuit [15]. Its transfer to the target depends on how well these references capture the target's noise response. Learning-based PEC learns quasiprobability coefficients from Clifford training circuits with a shared multi-qubit-gate layout, then uses those weights to combine circuit executions during mitigation [42].

Neural regression. MLPs apply nonlinear feed-forward maps to circuit-level feature vectors. GNNs aggregate gate information by message passing on circuit graphs and combine the resulting representation with noisy measurements [21]. GTranQEM combines attention over all circuit nodes with graph-structure biases, execution-order encodings, and qubit-wise noisy measurements [23]. QEMFormer fuses an MLP branch over pooled gate features with a graph-Transformer branch and global circuit and measurement features [24]. ML-QEM learns from simulated targets or another mitigation protocol such as ZNE, and reports hardware experiments up to 100 qubits. Its depth extrapolation depends on noise, and fine-tuning supports drift adaptation [21]. DAEM instead learns noise response from fiducial processes with efficiently generated ideal statistics, requiring no noise-free data from the target process [22].

SPLiT structures the learned mapping through nonnegative layerwise SPL rates and a survival prior accumulated across layers. An observable-conditioned residual accounts for term-dependent responses, including coherent errors and channel mismatch, while the inferred channels are shared across Pauli terms within a measurement context. Gate metadata and the optional single-qubit measurement-conditioning paths connect these predictions to deployment noise. This design combines explicit layerwise accumulation with deterministic term-wise mitigation, and its depth and drift generalization are evaluated in Sec. 4.