AGENT-VQE: LEARNING TRANSFERABLE QUANTUM CIRCUIT PROGRAMS FOR GRADIENT-FREE DEPLOYMENT AT SCALE

Anonymous authors
Paper under double-blind review

ABSTRACT

The high measurement overhead of gradient-based optimization is a bottleneck for deploying the variational quantum eigensolver at scale. We introduce *Agent-VQE*, an AI-assisted framework that learns transferable quantum circuit programs from small systems for gradient-free deployment. The agent analyzes local Hamiltonian features to propose circuit architectures and parameter-sharing rules, then iteratively refines these designs through small-system optimization and energy evaluation. The resulting circuit programs encode reusable circuit-building rules and shared circuit parameters. Larger circuits directly reuse these rules and coefficients without target-side gradient evaluations or parameter updates. Across six spin and fermionic many-body benchmarks spanning chain, ladder, and fully connected interaction structures, we fit circuit programs for ground-state energy estimation on systems of at most 16 qubits and evaluate the resulting circuits in noiseless simulations with 100–1000 qubits. Relative differences from reference energies remain below 1% for five of the six benchmarks, while Fermi–Hubbard has the largest difference of 5.31%. We further execute the learned one-dimensional transverse-field Ising model (TFIM) circuits on an OriginQ quantum computer. Measurements from 4–16-qubit circuits support measured-energy extrapolation up to 38 qubits. A separate comparison shows that the learned 32-qubit circuit achieves lower measured energy than randomized circuits of the same logical depth, despite hardware noise. These results support small-system circuit learning as a route to scalable ground-state energy estimation without costly target-side optimization.

1 INTRODUCTION

Preparing many-body ground states is a central application of quantum simulation. The variational quantum eigensolver (VQE) approaches this task using a classical optimizer to iteratively update variational circuit parameters based on quantum measurements. However, each parameter update requires repeated circuit executions to estimate energies and their derivatives when using gradient-based optimizers (Schuld et al., 2019). The cumulative measurement burden can grow with the number of variational parameters and optimization iterations, ultimately becoming a bottleneck to scaling this approach. Barren plateaus and hardware noise can further increase the difficulty of resolving useful gradients (McClean et al., 2018; Wang et al., 2021). Therefore, even when a variational circuit can be executed on a large-scale, low-noise quantum computer, optimizing its parameters to prepare a low-energy state can remain prohibitively costly.

Scalable Circuits ADAPT-VQE (SC-ADAPT-VQE) addresses the large-system training cost by exploiting translational symmetry and finite correlation lengths in the Schwinger-model ground state (Farrell et al., 2024). Ground-state preparation circuits are constructed and optimized on small systems using classical simulation, and the optimized parameters are extrapolated to generate circuits for larger systems without further optimization. Circuit construction automatically selects from parameterized building blocks designed by researchers to reflect the interaction structure and symmetries of the Hamiltonian. Therefore, adapting this workflow to other many-body models can require additional expert input to revise the building blocks and their circuit implementations.

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.

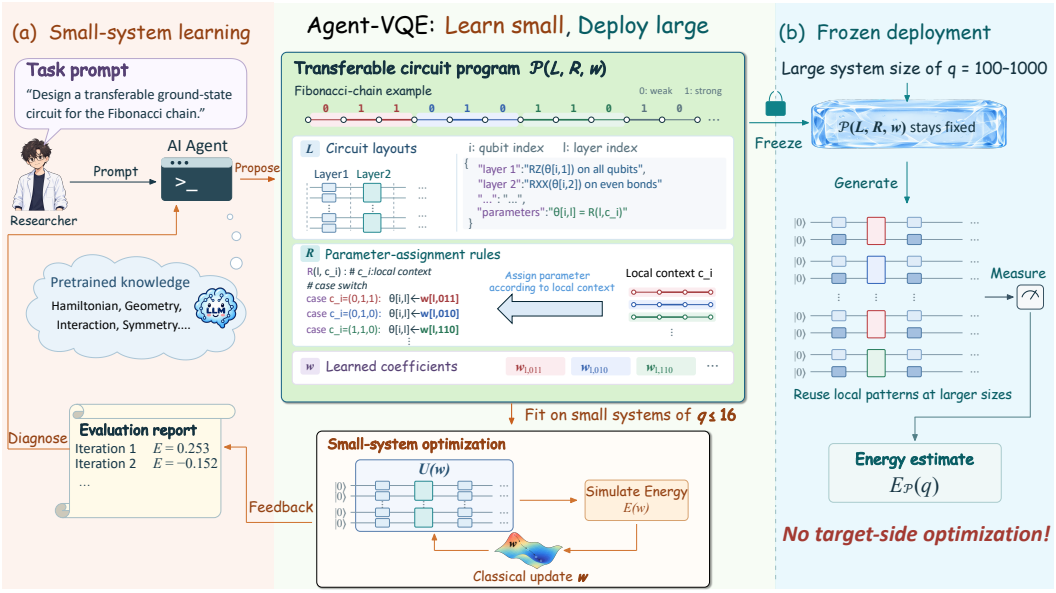

Figure 1: **Agent-VQE: Learn small, Deploy large. (a) Small-system learning.** The agent proposes and implements circuit layouts L and parameter-assignment rules R, drawing on the task specification and pretrained knowledge. An inner classical optimization loop fits shared coefficients $\mathbf{w}$ on systems with $q \leq 16$; evaluation feedback guides further design revisions. The Fibonacci-chain example illustrates bond-context sharing for two-qubit gates and parity-based sharing for single-qubit rotations. **(b) Frozen deployment.** The fixed program $\mathcal{P} = (L, R, \mathbf{w})$ generates larger circuits whose energies are evaluated without target-side parameter updates. The illustrated 100–1000-qubit regime uses noiseless simulation; hardware experiments are reported separately. Circuit patterns and evaluation-report values are schematic.

To address the challenge of automating VQE circuit design for ground-state preparation across different many-body models, we introduce *Agent-VQE*, an AI-assisted workflow for learning transferable quantum circuit programs (Figure 1). The agent draws on prior knowledge of quantum models and circuit construction to propose gate types, layouts, and rules for assigning parameters to gates. The agent then implements these designs as executable code for generating quantum circuits. Circuit parameters are optimized on small systems to lower the energy, and the resulting energies guide further revisions of the circuit design. This workflow aims to reduce repeated expert design and implementation effort when adapting ground-state preparation circuits to different many-body systems.

The resulting circuit program for each Hamiltonian family specifies reusable circuit layouts, parameter-assignment rules, and shared coefficients. Circuit layouts determine gate types, relative positions, and layer order. Parameter-assignment rules map local geometric or Hamiltonian features to gate angles through learned coefficients. This allows gate angles to adapt to spatial variations in local interactions, while keeping the number of learned coefficients independent of system size. After small-system fitting, the rules and coefficients are frozen, and the program generates larger circuits without target-side gradient evaluations or parameter updates. We call this *frozen deployment*. The transferability of these programs is motivated by the recurring local structure of the benchmark Hamiltonians. Reusing circuit rules in corresponding local environments can preserve energy accuracy per qubit at larger sizes, provided that small-system learning captures the relevant local correlations.

Our contributions are:

- We introduce Agent-VQE, an AI-assisted workflow that draws on Hamiltonian structure and prior scientific knowledge to design executable quantum circuit programs. Feedback from small-system optimization and energy evaluation guides the agent in refining their architectures and parameter-assignment rules.

- We construct transferable circuit programs that combine reusable circuit layouts and parameter-assignment rules conditioned on local geometry and Hamiltonian features. These programs allow spatially varying gate angles with a size-independent number of learned coefficients and support gradient-free deployment without target-side parameter updates.
- We evaluate six spin and fermionic many-body benchmarks spanning chain, ladder, and fully connected interaction structures. For programs fitted on at most 16 qubits and deployed at 100–1000 qubits, relative differences from reference energies remain below 1% for five benchmarks, while Fermi–Hubbard reaches 5.31%. Experiments with the learned one-dimensional TFIM circuits on an OriginQ quantum computer further demonstrate measured-energy extrapolation up to 38 qubits, and lower measured energy than randomized circuits at 32 qubits despite hardware noise.

2 Related work

Variational circuits and transferability. VQE combines parameterized quantum circuits with classical optimization to approximate ground states and their energies (Peruzzo et al., 2014; Wang et al., 2024). Incorporating Hamiltonian structure provides a principled basis for circuit design, as exemplified by Hamiltonian variational ansatzes that organize circuit operations around interaction terms (Wecker et al., 2015). Even with a suitable architecture, iterative optimization requires repeated quantum measurements (Schuld et al., 2019). To reduce repeated optimization across related Hamiltonians, Meta-VQE learns circuit parameterizations conditioned on Hamiltonian parameters, supporting energy estimation for Hamiltonians within the same family at new parameter values (Cervera-Lierta et al., 2021). Across system sizes, SC-ADAPT-VQE proposes to extrapolate circuits optimized on small systems to larger systems without further optimization under specific Hamiltonian assumptions (Farrell et al., 2024). Agent-VQE instead provides an automated workflow for adapting size-extensible circuit programs to different Hamiltonian families, using their interaction structures and small-system energy feedback to guide circuit design.

Automated quantum-circuit design. Automated circuit design aims to identify effective circuits without manually specifying the full architecture. ADAPT-VQE uses energy gradients to select which parameterized building block to append from a predefined set (Grimsley et al., 2019). Reinforcement learning instead learns a policy that chooses gate types and qubit locations (Ostaszewski et al., 2021). Both methods produce instance-specific circuits for a fixed number of qubits, requiring circuit search and parameter optimization to be repeated for a new system size. Beyond individual circuits, Lourens et al. (2025) combine a predefined symbolic language with evolutionary search to discover scalable circuit structures through small-system energy evaluations. Although their structures scale with system size, the variational parameters may still require size-specific optimization. Meta-design addresses quantum-state preparation by generating scalable circuit-construction programs from explicit small-system target-state expressions (Arlt et al., 2026). An LLM-based system automates molecular VQE circuit design, but still relies on classical parameter updates for each problem instance (Sakka et al., 2026, preprint). Agent-VQE takes the system Hamiltonian as input and uses small-system energy feedback to learn circuit-building rules, parameter-assignment rules, and shared coefficients. These rules and coefficients remain fixed when generating circuits for larger systems, enabling deployment without further parameter optimization.

LLM-assisted algorithm design. LLMs can draw on knowledge acquired during pretraining to propose candidate solutions to scientific problems and refine them using execution feedback. This paradigm extends beyond quantum computing to mathematical discovery and algorithm design. FunSearch addresses mathematical and combinatorial optimization problems by generating and evaluating heuristic functions, then reusing high-performing candidates in subsequent prompts (Romera-Paredes et al., 2024). For combinatorial optimization, ReEvo generates heuristic algorithms and uses reflections on their relative performance to guide further revisions (Ye et al., 2024). In scientific equation discovery, LLM-SR further separates structural proposals from parameter estimation: the LLM proposes equation structures in executable code, while numerical optimizers fit their coefficients to data (Shojaee et al., 2025). Agent-VQE similarly draws on an LLM's knowledge of quantum circuits within an agent workflow that proposes, implements, and revises circuit designs using feedback from small-system optimization and energy evaluation.

3 PROBLEM FORMULATION

Variational quantum eigensolver. Let H_q be the Hamiltonian of a q-qubit many-body system. The ground-state energy is the lowest eigenvalue,

$$E_0(q) = \min_{\langle\psi|\psi\rangle=1} \langle\psi|H_q|\psi\rangle, \tag{1}$$

which characterizes the system's zero-temperature energetics. VQE (Peruzzo et al., 2014) restricts this search to states prepared by a circuit $U_q(\boldsymbol{\theta})$ with P_q adjustable parameters $\boldsymbol{\theta} \in \mathbb{R}^{P_q}$, acting on a fixed initial state $|\psi_{\mathrm{init},q}\rangle$:

$$|\psi_q(\boldsymbol{\theta})\rangle = U_q(\boldsymbol{\theta})|\psi_{\mathrm{init},q}\rangle, \tag{2}$$

$$E_q(\boldsymbol{\theta}) = \langle\psi_{\mathrm{init},q}|U_q^\dagger(\boldsymbol{\theta})H_qU_q(\boldsymbol{\theta})|\psi_{\mathrm{init},q}\rangle. \tag{3}$$

A quantum processor estimates $E_q(\boldsymbol{\theta})$ from repeated measurements, and a classical optimizer updates $\boldsymbol{\theta}$ using these estimates. Two barriers arise in practice:

1. *Ansatz expressivity.* The circuit architecture, or ansatz, determines the accessible states and the best attainable energy $\min_{\boldsymbol{\theta}} E_q(\boldsymbol{\theta}) \geq E_0(q)$. Circuit constructions should reflect the Hamiltonian structure to reduce the gap between this minimum and $E_0(q)$.
2. *Optimization cost.* Iterative parameter updates require repeated energy measurements and, for gradient-based methods, gradient estimates. The resulting measurement overhead can become substantial at large q (Schuld et al., 2019).

Automated design for frozen deployment. These challenges motivate the automated design of ground-state circuit programs that capture model-specific Hamiltonian structure and support deployment at larger sizes without further optimization. With the Hamiltonian of a many-body model as input, the desired output is an executable program $\mathcal{P}$, learned on small systems, that generates circuits for larger systems.

4 AGENT-VQE

Agent-VQE automates the design of ground-state VQE circuits that can be learned on small systems and instantiated at larger sizes. Given a many-body Hamiltonian specification and the admissible physical and gate constraints, the framework searches for a circuit program whose construction rules and number of learned coefficients are independent of the target system size.

4.1 TRANSFERABLE CIRCUIT PROGRAMS

We represent a circuit program as $\mathcal{P} = (L, R, \mathbf{w})$, where $L = (p_1, \ldots, p_d)$ contains the discrete layout rule for each block, R is a context-dependent parameter-assignment rule, and $\mathbf{w}$ contains the shared coefficients. Each p_ℓ specifies a size-independent pattern of gate types and relative qubit relations, such as applying a two-qubit gate to every nearest-neighbor pair of a given type. Instantiating this pattern on the interaction graph of a q-qubit system determines the concrete gate locations. The same L can therefore generate circuits for different system sizes without redesigning the layout.

For a gate of type τ_e at location e, the program-specific context rule $\mathcal{C}$ derives a local descriptor c_e from the Hamiltonian and interaction geometry. Depending on the benchmark, c_e encodes features such as interaction type or strength, site class, boundary status, or local field phase. The parameter-assignment rule R then determines the gate angle $\theta_{\ell,e}$:

$$c_e = \mathcal{C}(H_q, e), \qquad \theta_{\ell,e} = R(\ell, p_\ell, \tau_e, c_e; \mathbf{w}). \tag{4}$$

Here, $\mathcal{C}$ is a component of R specified during circuit design. Therefore, L specifies the gate types and their locations within each circuit block, while R maps the shared coefficients $\mathbf{w}$ and local context c_e to the corresponding gate parameters. The number of learned coefficients remains independent of q, even when the generated angles vary across sites to represent model inhomogeneity.

The construction is size-extensible by design: increasing q adds further instances of recurring local or symmetry-defined interaction motifs, to which the same layout and assignment rules apply. Transfer of energy accuracy is expected when the small training systems capture the local structures and correlations relevant to those motifs. We test this condition empirically in Section 5.1.

4.2 LEARNING CIRCUIT PROGRAMS

Quantum circuit optimization involves a discrete gate layout and continuous gate parameters, which we optimize via an outer discrete-design loop and an inner continuous-optimization loop. The outer loop searches over circuit layouts L and parameter-assignment rules R, while the inner loop fits their shared coefficients $\mathbf{w}$ on a set S of small systems. For each candidate (L, R), a numerical optimizer approximately minimizes the prescribed source-side energy objective $\mathcal{J}(L, R, \mathbf{w})$. The final design is selected from the candidates evaluated within a computational budget B:

$$\widehat{\mathbf{w}}(L, R) \approx \arg\min_{\mathbf{w}} \mathcal{J}(L, R, \mathbf{w}), \qquad (\widehat{L}, \widehat{R}) \in \arg\min_{(L,R)\in\mathcal{K}_{\mathrm{eval}}(B)} \mathcal{J}(L, R, \widehat{\mathbf{w}}(L, R)), \tag{5}$$

where $\mathcal{K}_{\mathrm{eval}}(B)$ denotes the set of evaluated designs eligible under the declared acceptance protocol. The selected program $(\widehat{L}, \widehat{R}, \widehat{\mathbf{w}})$ is then frozen. At deployment, its layout and parameter-assignment rules generate circuits for larger systems using the stored coefficients, without target-side parameter optimization or architecture revisions.

More specifically, for a source-size set S, each candidate $\mathcal{P} = (L, R, \mathbf{w})$ is instantiated at every $q \in S$ and fitted jointly. Writing $U_q(L, R, \mathbf{w})$ for the generated circuit and $|\psi_{\mathrm{init,q}}\rangle$ for its prescribed initial state, a canonical source-side objective is

$$\mathcal{J}(L, R, \mathbf{w}) = \sum_{q\in S} \langle\psi_{\mathrm{init,q}}| U_q^\dagger(L, R, \mathbf{w}) H_q U_q(L, R, \mathbf{w}) |\psi_{\mathrm{init,q}}\rangle. \tag{6}$$

Individual benchmarks may use benchmark-specific variants of this objective. The source sizes do not exceed 16 qubits. The same candidate rules and coefficient vector are used at all included sizes. Appendix B.1 specifies the source-side search, acceptance checks, and program-selection procedure.

4.3 AGENT DESIGN

An AI agent directs the outer search by deciding which circuit designs to evaluate, how to revise them, and where to allocate the remaining computational budget. We represent the agent as

$$A = (\mathcal{S}, \mathcal{A}, \mathcal{M}, \mathcal{W}) \tag{7}$$

where state $\mathcal{S}$ encodes the current problem, candidate program, evaluation feedback, and available resources; action $\mathcal{A}$ specifies the admissible design and experiment actions; memory $\mathcal{M}$ retains previous candidates and their outcomes; and world knowledge $\mathcal{W}$ provides pretrained scientific knowledge.

Given a Hamiltonian and its interaction geometry, the agent relies on $\mathcal{W}$ to propose physically motivated gate layouts, local-context descriptors, and parameter-sharing rules. After numerical coefficient fitting, it interprets the resulting energies across source sizes, convergence behavior, and circuit costs to choose the next intervention: revising the architecture, changing the sharing rule, continuing optimization from a promising checkpoint, or abandoning an unsuccessful direction. Memory allows these decisions to draw on the accumulated search history, including both successful and unsuccessful trials. Through these sequential decisions, the agent shapes the evaluated candidate set $\mathcal{K}_{\mathrm{eval}}(B)$ and allocates optimization effort among its members, aiming to concentrate a limited budget on designs supported by physical knowledge and empirical feedback. Numerical optimizers carry out the continuous coefficient updates requested by the agent. Appendix B.2 details the state representation, action space, memory, and execution loop of the agent used in Agent-VQE.

5 EXPERIMENTS

We evaluate Agent-VQE in three settings: size scaling across six many-body benchmarks, comparison with SC-ADAPT-VQE, and execution on quantum hardware. The first two settings use noiseless classical simulation; the hardware experiments test the learned one-dimensional TFIM circuit under real device noise.

5.1 SCALING ACROSS SIX MANY-BODY BENCHMARKS

Experimental setup. The benchmarks cover chain, ladder, and fully connected interaction structures: the one-dimensional TFIM, a two-leg Heisenberg ladder, a Fibonacci Heisenberg chain, an

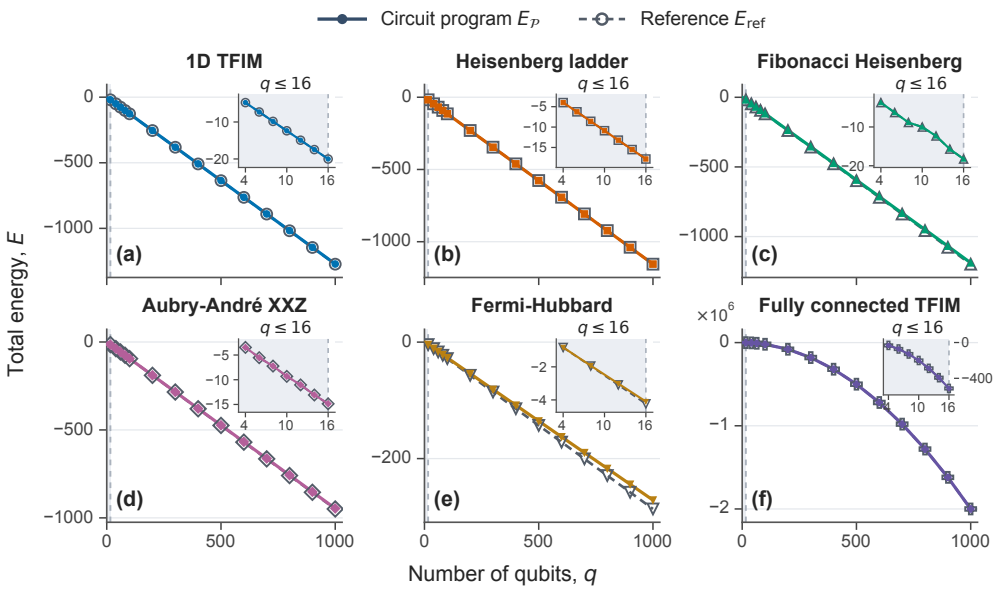

Figure 2: **Energy scaling of frozen circuit programs across six many-body benchmarks.** (a–f) Total circuit energies (colored solid curves, filled markers) and reference energies (gray dashed curves, open markers); insets enlarge the small-system training range $q \le 16$, marked by gray shading.

Aubry–André XXZ chain, a half-filled Fermi–Hubbard chain, and a fully connected TFIM. Each model has a separate circuit program whose coefficients are fitted on systems with at most 16 qubits. We then generate larger circuits with the layout rules, parameter-assignment rules, and learned coefficients fixed. We assess the generated circuits using the absolute energy difference per qubit ϵ_q and the relative energy difference r_q:

$$\epsilon_q = \frac{|E_{\mathcal{P}}(q) - E_{\text{ref}}(q)|}{q}, \qquad r_q = \frac{|E_{\mathcal{P}}(q) - E_{\text{ref}}(q)|}{|E_{\text{ref}}(q)|}, \tag{8}$$

where $E_{\mathcal{P}}(q)$ is the energy of the state prepared by the circuit generated by $\mathcal{P}$ at size q, and $E_{\text{ref}}(q)$ is the reference ground-state energy.

Following the small-system learning procedure in Section 4, we freeze the selected circuit program for each benchmark after training on systems with $q \le 16$ and instantiate it at larger sizes without further parameter optimization; Appendix A specifies the corresponding Hamiltonians, circuit layouts, and parameter-assignment rules. Circuit energies $E_{\mathcal{P}}(q)$ are evaluated exactly for the one-dimensional and fully connected TFIMs, and to high numerical precision for the remaining four benchmarks. Ground-state references $E_{\text{ref}}(q)$ are likewise obtained exactly when tractable, and otherwise approximated numerically. Appendix B.3 details the evaluation methods and numerical settings. All six benchmarks are evaluated at the large-system range of $q = 100, 200, \ldots, 1000$.

Energy scaling and accuracy. Figure 2 compares $E_{\mathcal{P}}(q)$ and $E_{\text{ref}}(q)$ across system sizes. The circuit energies $E_{\mathcal{P}}(q)$ follow the overall reference trends: approximately linear scaling for the five local-interaction benchmarks and quadratic scaling for the fully connected TFIM under the Hamiltonian convention specified in Appendix A. The insets resolve the small-system range $q \le 16$, showing the data used to learn each circuit program.

To quantify the deviations from these reference trends, Figure 3(a) reports the absolute energy differences per qubit. Over the 100–1000-qubit range, ϵ_q varies moderately with all circuit programs and learned coefficients held fixed. The maximum differences over this range are 1.236×10^{-3} for the one-dimensional TFIM, 7.752×10^{-4} for the Heisenberg ladder, and 2.900×10^{-3} for the Aubry–André chain. The Fibonacci difference reaches 1.161×10^{-2} and remains near 10^{-2}, while Hubbard exhibits the largest difference across all 60 evaluations, 1.52211×10^{-2} at $q = 900$. The

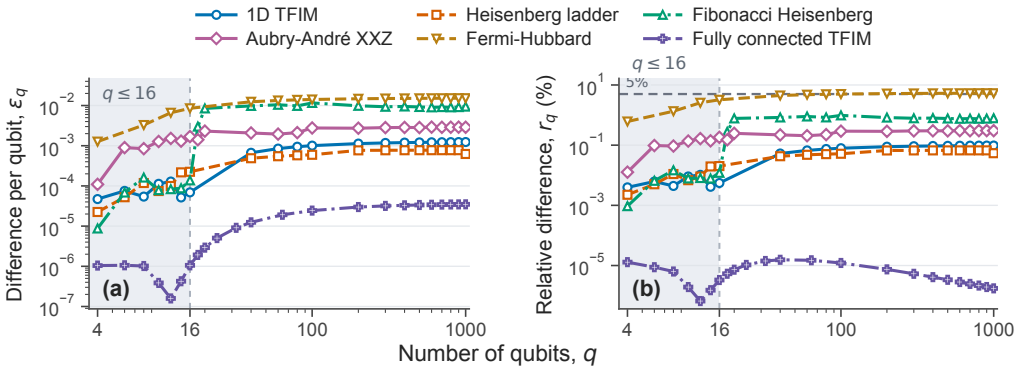

Figure 3: **Energy differences of frozen circuit programs across six many-body benchmarks.** (a) Absolute energy difference per qubit, ϵ_q. (b) Relative energy difference, r_q, displayed as a percentage. Both panels use logarithmic axes and share a legend; shading marks the small-system training range $q \leq 16$. The horizontal dashed line in (b) marks 5%.

Table 1: Schwinger-model vacuum energy density E/N, where N is the number of spatial lattice sites. SC-MPS denotes the published MPS evaluation of SC-ADAPT-VQE; SC-MPS and DMRG values are from Farrell et al. (2024). Lower variational energy is better.

N	q	DMRG	SC-MPS	Agent-VQE
50	100	-0.32805	-0.32790	-0.32799109
100	200	-0.32942	-0.32928	-0.32935673
200	400	-0.33011	-0.32996	-0.33002921
300	600	-0.33034	-0.33019	-0.33024418
400	800	-0.33045	-0.33031	-0.33034477
500	1000	-0.33052	-0.33038	-0.33039961

Fermi–Hubbard benchmark is structurally distinct from the spin models, combining spin-resolved fermionic hopping and onsite interactions over a four-dimensional local state space. Despite this richer structure, the frozen 96-parameter circuit program preserves the reference energy-scaling trend through 1000 qubits and achieves $r_q = 5.31\%$ at this size without target-side optimization. For the fully connected TFIM, the absolute differences range from 2.43×10^{-5} to 3.49×10^{-5}. Figure 3(b) summarizes the relative differences at 1000 qubits: r_q ranges from 0.054% to 0.785% across the one-dimensional TFIM, Heisenberg ladder, Fibonacci, and Aubry–André benchmarks, and is $1.74 \times 10^{-6}\%$ for the fully connected TFIM.

These results support the use of learned circuit programs for ground-state energy estimation at up to 1000 qubits without target-side parameter optimization. Across the tested benchmarks with approximately linear and quadratic energy scaling, the energy differences are model dependent but exhibit limited growth over the large-system range.

5.2 Comparison with SC-ADAPT-VQE

We compare with SC-ADAPT-VQE on the open-boundary Schwinger model used in its original evaluation, matching $m = 0.5$, $g = 0.3$, and all six reported volumes (Farrell et al., 2024). Denoting the number of spatial lattice sites by N, the qubit count is $q = 2N$. We fit a depth-32 circuit program on $q \in \{12, 16, 20\}$ and then deploy it without further optimization at $q \in \{100, 200, 400, 600, 800, 1000\}$. Table 1 compares the resulting energy densities with the published SC-MPS and DMRG values.

Agent-VQE yields a lower variational energy than SC-MPS at all six volumes. The mean absolute difference from the listed DMRG values is 8.74×10^{-5} per spatial lattice site, compared with 1.45×10^{-4} for SC-MPS. These results show that the agent-assisted workflow produced a transfer-

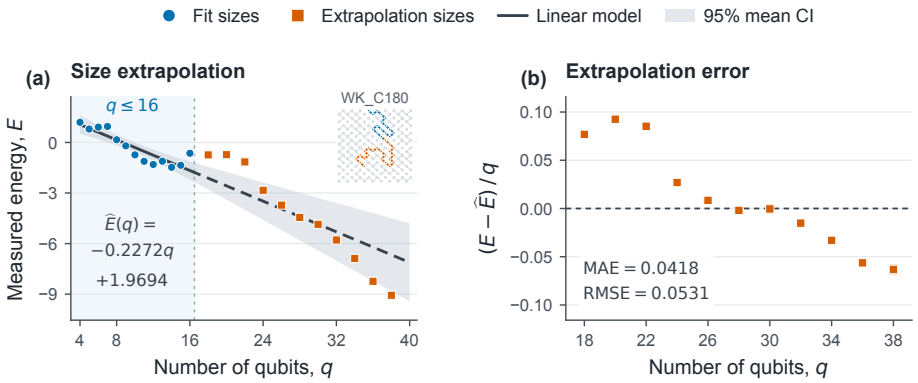

Figure 4: **Measured-energy extrapolation on quantum hardware.** (a) Blue circles show the measured means at the fit sizes $q = 4, 5, \ldots, 16$ (shaded region). The affine model is fitted only to these measurements, shown as a solid line within the fitting range and extrapolated as a dashed line without refitting. Orange squares show measurements at the larger extrapolation sizes $q = 18, 20, \ldots, 38$. The inset shows the 38-qubit chain on WK_C180. A circuit of size q uses the first q qubits of this chain. The gray band denotes the pointwise 95% confidence interval for the fitted mean response, estimated solely from the measurements at the fit sizes and extrapolated to larger sizes. It quantifies uncertainty in the fitted line under the linear-model assumptions. (b) Signed per-qubit deviations, $(E - \widehat{E})/q$, at the extrapolation sizes.

able circuit program whose MPS-evaluated energies are slightly lower than those of the published SC-ADAPT-VQE circuits constructed from a physics-informed operator pool, demonstrating the promise of agent-assisted circuit-program design.

We further test, on the nonuniform Fibonacci chain, a simplified baseline that shares the same parameters for entangling gates across all active bonds in each layer, motivated by the frozen-extrapolation strategy of SC-ADAPT-VQE. Programs learned on $q \in \{6, 8, 10, 12\}$ are deployed through $q = 100$ without reoptimization. Under the same budget of 144 learned coefficients, the baseline yields $\epsilon_{100} = 2.66 \times 10^{-1}$, whereas the Agent-VQE program conditions these angles on weak and strong local bonds and reduces the difference to 2.70×10^{-2}, an 89.8% reduction. This size-independent, context-dependent parameterization can facilitate transfer to quasiperiodic interactions while reducing the need for model-specific expert design. Appendix B.4 details this controlled test.

5.3 Evaluation on quantum hardware

We execute the frozen one-dimensional TFIM circuit on OriginQ's Wukong 180 superconducting quantum computer (backend `WK_C180`) (Origin Quantum, 2026) in two experiments: measured-energy extrapolation from small to larger circuits, and comparison with randomized circuits at 32 qubits. These experiments assess the behavior of circuit programs under realistic hardware noise.

Measured-energy extrapolation. We first execute the circuits on chains of $q = 4, 5, \ldots, 16$ qubits and measure their energies. For each system size q, we use the first q qubits of a fixed physical chain and repeat the energy measurement five times, using 2000 shots in each of the X and Z measurement bases per repeat. We fit a straight line $\widehat{E}(q) = aq + b$ to the measured mean energies by ordinary least squares, giving each size equal weight. We then execute and measure the circuits at $q = 18, 20, \ldots, 38$, comparing these mean energies with the fixed extrapolated line. Measurement and fitting details are provided in Appendix C.2.

The small-system measurements yield $\widehat{E}(q) = -0.2272q + 1.9694$. For comparison, fitting all reference energies plotted in Figure 2(a) gives $\widehat{E}_{\mathrm{ref}}(q) = -1.2732q + 0.3536$. Relative to the reference trend, hardware measurements yield higher energies at the same system size and a shallower negative slope, due to realistic device noise. Despite these shifts, the measured energies retain an approximately linear decrease with system size. Across the larger tested sizes $q = 18, 20, \ldots, 38$, the

extrapolated energies differ from the measured means by a per-qubit mean absolute error (MAE) of 0.0418 and root mean squared error (RMSE) of 0.0531. The measured energies retain a predictable size dependence under device noise.

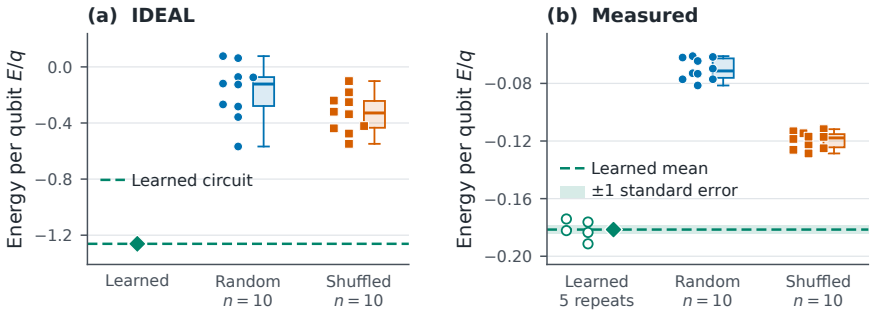

Figure 5: **Learned and randomized circuits at 32 qubits.** (a) Exact free-fermion evaluations of the learned and randomized circuits. The green diamond and dashed line indicate the energy of the learned frozen circuit. (b) Measurements on OriginQ hardware. Green open circles show five repeated evaluations of the learned frozen circuit; the diamond, dashed line, and shaded band show their mean and ± 1 standard error of the mean (SEM). The random-angle and shuffled-layer groups each contain ten circuits with the same logical architecture as the learned circuit. Boxes indicate the median and interquartile range (IQR), with whiskers extending to the most extreme values within 1.5 IQR.

Comparison with randomized circuits. To assess whether the learned parameters and layer ordering remain effective under hardware noise, we deploy the TFIM circuit on the 32-qubit physical chain and compare its measured energy with two groups of randomized circuits. Each group contains ten circuits with the same layouts as the learned circuit. In the random-angle group, the learned ZZ and X angles are replaced by independent samples from $\mathcal{U}[-\pi, \pi)$. In the shuffled-layer group, the learned angle pairs are retained, but the paired ZZ and X layer blocks are randomly reordered. Each randomized circuit is evaluated once with 2000 shots per measurement basis, while the learned circuit is evaluated five times.

Figure 5 compares exact free-fermion evaluations with hardware results. The learned circuit has an ideal circuit energy of -1.2615 per qubit, versus group means of -0.1728 for random angles and -0.3315 for shuffled layers. On hardware, the corresponding values are -0.1815 ± 0.0030, -0.0701, and -0.1194, where the uncertainty of the learned-circuit mean is one SEM across five repeated evaluations. Although hardware noise reduces the energy accuracy, the learned circuit achieves a lower mean measured energy than all 20 randomized circuits with the same logical layout and depth. The consistent advantage in both ideal and hardware evaluations provides evidence that the learned parameters and layer ordering preserve a measurable performance benefit under real-device noise.

6 Conclusion

We introduced Agent-VQE, an AI-assisted framework that designs size-transferable circuit programs and fits shared coefficients on small systems for frozen deployment at larger sizes. Across six spin and fermionic benchmarks, programs learned on at most 16 qubits were deployed at 100–1000 qubits without target-side optimization; relative energy differences remained below 1% for five benchmarks and reached 5.31% for Fermi–Hubbard. Agent-VQE also achieved lower variational energies than the published SC-MPS values at all six Schwinger-model volumes (Farrell et al., 2024). On quantum hardware, the learned TFIM program preserved the measured-energy scaling trend through 38 qubits and achieved lower measured energy than matched randomized circuits at 32 qubits despite device noise. Together, these results suggest that agent-guided design can reduce model-specific manual effort and amortize circuit-development and optimization costs across system sizes. Future work can extend the framework to broader Hamiltonian variations, additional observables, and hardware-aware feedback.

Reproducibility statement

Appendix A describes the Hamiltonians, circuit layouts, parameter rules, and gate counts. Appendix B covers the Agent-VQE learning and deployment framework, and Appendix C documents the hardware experiments. Appendix D presents a worked Agent-VQE trajectory for the Aubry–André XXZ chain.

AI use statement

An AI coding-and-experiment agent assisted with circuit-program design, implementation, experiment execution, and result analysis. Conventional numerical optimizers fitted the continuous coefficients. Generative AI also assisted with manuscript drafting and language editing. The authors are responsible for the scientific claims, source attribution, and final verification of the reported results.

References

Sören Arlt, Haonan Duan, Felix Li, Sang Michael Xie, Yuhuai Wu, and Mario Krenn. Meta-designing quantum experiments with language models. *Nature Machine Intelligence*, 8(2):148–157, 2026. doi: 10.1038/s42256-025-01153-0.

Alba Cervera-Lierta, Jakob S. Kottmann, and Alán Aspuru-Guzik. Meta-variational quantum eigensolver: Learning energy profiles of parameterized hamiltonians for quantum simulation. *PRX Quantum*, 2(2):020329, 2021. doi: 10.1103/PRXQuantum.2.020329.

Tetsuo Deguchi and Ruihong Yue. Exact solutions of 1-D Hubbard model with open boundary conditions and the conformal dimensions under boundary magnetic fields, 1997. URL `https://arxiv.org/abs/cond-mat/9704138`.

Sébastien Dusuel and Julien Vidal. Continuous unitary transformations and finite-size scaling exponents in the Lipkin–Meshkov–Glick model. *Physical Review B*, 71(22):224420, 2005. doi: 10.1103/PhysRevB.71.224420.

Roland C. Farrell, Marc Illa, Anthony N. Ciavarella, and Martin J. Savage. Scalable circuits for preparing ground states on digital quantum computers: The Schwinger model vacuum on 100 qubits. *PRX Quantum*, 5:020315, 2024. doi: 10.1103/PRXQuantum.5.020315.

Thierry Giamarchi. *Quantum Physics in One Dimension*. Oxford University Press, 2003. ISBN 9780198525004. doi: 10.1093/acprof:oso/9780198525004.001.0001.

Johnnie Gray. quimb: A Python package for quantum information and many-body calculations. *Journal of Open Source Software*, 3(29):819, 2018. doi: 10.21105/joss.00819.

Harper R. Grimsley, Sophia E. Economou, Edwin Barnes, and Nicholas J. Mayhall. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nature Communications*, 10:3007, 2019. doi: 10.1038/s41467-019-10988-2.

Anuradha Jagannathan. The Fibonacci quasicrystal: Case study of hidden dimensions and multifractality. *Reviews of Modern Physics*, 93(4):045001, 2021. doi: 10.1103/RevModPhys.93.045001.

Matt Lourens, Ilya Sinayskiy, Johannes N. Kriel, and Francesco Petruccione. Generating generalized ground-state ansätze from few-body examples. *Physical Review Letters*, 135(21):210401, 2025. doi: 10.1103/5vc5-9f4d.

Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nature Communications*, 9:4812, 2018. doi: 10.1038/s41467-018-07090-4.

OpenAI. GPT-5.6 Sol model, 2026. URL `https://developers.openai.com/api/docs/models/gpt-5.6-sol`. Accessed September 26, 2026.

Origin Quantum. Origin Wukong 180: 4th gen superconducting quantum computer, 2026. URL `https://qcloud.originqc.com.cn/en/wukong180`. Official product page. Accessed September 21, 2026.

Mateusz Ostaszewski, Lea M. Trenkwalder, Wojciech Masarczyk, Eleanor Scerri, and Vedran Dunjko. Reinforcement learning for optimization of variational quantum circuit architectures. In *Advances in Neural Information Processing Systems*, volume 34, pp. 18182–18194, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/9724412729185d53a2e3e7f889d9f057-Abstract.html.

Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J. Love, Alán Aspuru-Guzik, and Jeremy L. O'Brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5:4213, 2014. doi: 10.1038/ncomms5213.

Pierre Pfeuty. The one-dimensional Ising model with a transverse field. *Annals of Physics*, 57(1): 79–90, 1970. doi: 10.1016/0003-4916(70)90270-8.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024. doi: 10.1038/s41586-023-06924-6.

Subir Sachdev. *Quantum Phase Transitions*. Cambridge University Press, second edition, 2011. doi: 10.1017/CBO9780511973765.

Kenya Sakka, Wataru Mizukami, and Kosuke Mitarai. An LLM system for autonomous variational quantum circuit design. *arXiv preprint arXiv:2606.13380*, 2026. URL https://arxiv.org/abs/2606.13380.

Maria Schuld, Ville Bergholm, Christian Gogolin, Josh Izaac, and Nathan Killoran. Evaluating analytic gradients on quantum hardware. *Physical Review A*, 99:032331, 2019. doi: 10.1103/PhysRevA.99.032331.

H. Schulz. Hubbard chain with reflecting ends. *Journal of Physics C: Solid State Physics*, 18(3): 581–601, 1985. doi: 10.1088/0022-3719/18/3/010.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K. Reddy. LLM-SR: Scientific equation discovery via programming with large language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/28df8e730c054c5331855fd4d5403ba9-Abstract-Conference.html.

Guifré Vidal. Efficient classical simulation of slightly entangled quantum computations. *Physical Review Letters*, 91(14):147902, 2003. doi: 10.1103/PhysRevLett.91.147902.

Samson Wang, Enrico Fontana, M. Cerezo, Kunal Sharma, Akira Sone, Lukasz Cincio, and Patrick J. Coles. Noise-induced barren plateaus in variational quantum algorithms. *Nature Communications*, 12:6961, 2021. doi: 10.1038/s41467-021-27045-6.

Xin Wang, Bo Qi, Yabo Wang, and Daoyi Dong. Entanglement-variational hardware-efficient ansatz for eigensolvers. *Physical Review Applied*, 21(3):034059, 2024. doi: 10.1103/PhysRevApplied.21.034059.

Dave Wecker, Matthew B. Hastings, and Matthias Troyer. Progress towards practical quantum variational algorithms. *Physical Review A*, 92:042303, 2015. doi: 10.1103/PhysRevA.92.042303.

Steven R. White. Density matrix formulation for quantum renormalization groups. *Physical Review Letters*, 69(19):2863–2866, 1992. doi: 10.1103/PhysRevLett.69.2863.

Haoran Ye, Jiarui Wang, Zhiguang Cao, Federico Berto, Chuanbo Hua, Haeyeon Kim, Jinkyoo Park, and Guojie Song. ReEvo: Large language models as hyper-heuristics with reflective evolution. In *Advances in Neural Information Processing Systems*, volume 37, pp. 43571–43608, 2024. doi: 10.52202/079017-1381.

Marko Žnidarič and Marko Ljubotina. Interaction instability of localization in quasiperiodic systems. *Proceedings of the National Academy of Sciences*, 115(18):4595–4600, 2018. doi: 10.1073/pnas.1800589115.

A BENCHMARK HAMILTONIANS AND CIRCUIT PROGRAMS

This appendix specifies the six saved programs evaluated in Section 5.1. We use open boundaries for the five local-interaction models and all-to-all interactions for the fully connected TFIM. For $P \in \{X, Y, Z\}$, P_i denotes the Pauli operator P acting on qubit (or site) i. For a Pauli string P, $R_P(\theta) = \exp(-i\theta P/2)$. Given d blocks U_ℓ, $\ell = 1, \ldots, d$, the full circuit is $U_q = U_d \cdots U_1$ acting on the stated product state. We denote by $\mathcal{E}_\ell(q)$ the *active edge set* of block ℓ: the qubit pairs (i, j) selected by its layout rule when expanded to system size q. Within each displayed block, gates are applied from left to right. The topology and block diagrams show the program rules rather than numerical values of the learned coefficients.

A.1 ONE-DIMENSIONAL TFIM

The Hamiltonian of the one-dimensional TFIM (Sachdev, 2011) is

$$H_{\text{TFIM}} = -\sum_{i=0}^{q-2} Z_i Z_{i+1} - \sum_{i=0}^{q-1} X_i. \tag{9}$$

Starting from $|+\rangle^{\otimes q}$, the circuit repeats

$$U_\ell(q) = \left[\prod_{i=0}^{q-1} R_X(\beta_\ell)_i\right] \left[\prod_{i=0}^{q-2} R_{ZZ}(\gamma_\ell)_{i,i+1}\right], \qquad \ell = 1, \ldots, 24. \tag{10}$$

The same two coefficients $(\gamma_\ell, \beta_\ell)$ are shared across all positions in block ℓ, while different blocks have independent coefficients.

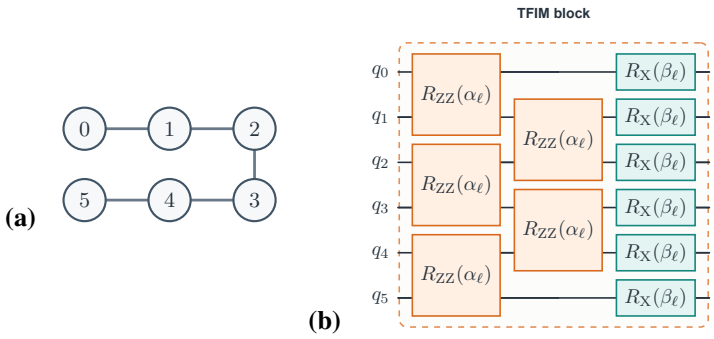

Figure 6: **One-dimensional TFIM program.** Representative six-qubit example. (a) Open nearest-neighbor interaction graph, folded for compact representation. (b) The corresponding circuit layout for one of the 24 blocks. Nearest-neighbor R_{ZZ} gates share γ_ℓ, and all R_X gates share β_ℓ. Commuting R_{ZZ} gates are placed in two columns to make the layout readable.

A.2 HEISENBERG LADDER

For the two-leg Heisenberg ladder with N rungs, $q = 2N$ and

$$H_{\text{ladder}} = \frac{1}{2} \sum_{\langle i,j \rangle} (X_i X_j + Y_i Y_j + Z_i Z_j). \tag{11}$$

Here $\langle i, j \rangle$ runs over all nearest-neighbor qubit pairs along the legs and rungs of the open two-leg ladder. The circuit starts from a checkerboard Néel product state (Giamarchi, 2003, Sec. 6.4.1). For the ladder, $\mathcal{E}_\ell(q)$ is generated by one of three fixed layouts: all rung edges, even-start leg edges, or odd-start leg edges. For a qubit i at row $r \in \{0, 1\}$ and column x, we define its checkerboard sublattice $s(i) \equiv s(r, x)$ as A when $r + x$ is even and B otherwise. Under this partition, every

nearest-neighbor ladder edge connects opposite sublattices. The block has the form

$$U_\ell(q) = \prod_{(i,j)\in\mathcal{E}_\ell(q)} R_{ZZ}(\gamma_\ell^{ZZ})_{ij} R_{YY}(\gamma_\ell^{YY})_{ij} R_{XX}(\gamma_\ell^{XX})_{ij} \times \prod_{i=0}^{q-1} R_Z(\delta_{\ell,s(i)})_i R_Y(\beta_{\ell,s(i)})_i R_Z(\alpha_{\ell,s(i)})_i. \tag{12}$$

Single-qubit $R_Z R_Y R_Z$ rotations share one three-angle vector within sublattice A and another within sublattice B. On every active edge, the block applies R_{XX}, R_{YY}, and R_{ZZ} rotations whose three angles are shared across all active edges. Each block therefore contains nine learned coefficients. Denoting the rung, even-start leg, and odd-start leg layouts by R, E, and O, respectively, the learned 54-block schedule is

1:9	R	E	R	O	R	O	E	R	R
10:18	O	E	R	E	O	R	E	O	R
19:27	E	R	O	E	R	O	E	R	O
28:36	E	R	O	R	O	E	R	O	E
37:45	R	O	E	R	O	E	O	E	R
46:54	O	E	R	R	O	E	R	O	E

(13)

Each symbol generates its corresponding active edge set at the target size q: R activates all rung edges, while E and O activate the alternating leg edges that start at even and odd columns, respectively.

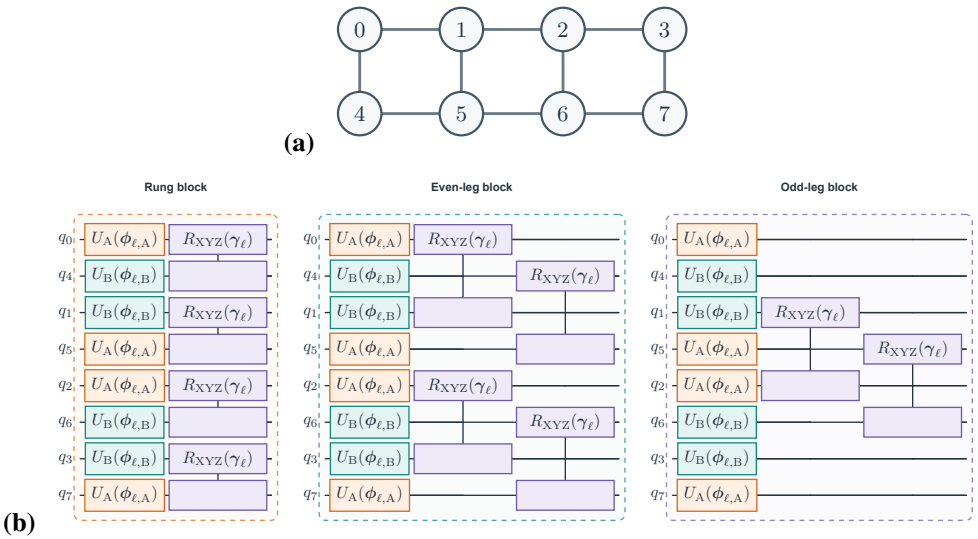

Figure 7: **Heisenberg-ladder program.** Representative eight-qubit example. (a) Two-leg ladder and qubit indexing. (b) The three block layouts used in the learned program. $U_A(\boldsymbol{\phi}_{\ell,A})$ and $U_B(\boldsymbol{\phi}_{\ell,B})$ denote $R_Z R_Y R_Z$ rotations, where $\boldsymbol{\phi}_{\ell,s} = (\alpha_{\ell,s}, \beta_{\ell,s}, \delta_{\ell,s})$ contains three independent angles for sublattice $s \in \{A, B\}$. Each $R_{\mathrm{XYZ}}(\boldsymbol{\gamma}_\ell)$ denotes consecutive R_{XX}, R_{YY}, and R_{ZZ} gates, where $\boldsymbol{\gamma}_\ell = (\gamma_\ell^{XX}, \gamma_\ell^{YY}, \gamma_\ell^{ZZ})$ contains three independent angles shared by active edges in block ℓ.

A.3 Fibonacci Heisenberg chain

The Fibonacci Heisenberg chain (Jagannathan, 2021, Sec. XI.A) uses

$$H_{\mathrm{Fib}} = \frac{1}{2}\sum_{i=0}^{q-2} J_i(X_iX_{i+1} + Y_iY_{i+1} + Z_iZ_{i+1}), \tag{14}$$

$$b_i = \lfloor (i+1)\beta + \phi_0 \rfloor - \lfloor i\beta + \phi_0 \rfloor, \qquad J_i = 0.3 + 1.4b_i, \tag{15}$$

with $\beta = (\sqrt{5}-1)/2$ and $\phi_0 = \sqrt{3}/10$. Here $b_i = 0$ denotes a weak bond with $J_i = J_{\mathrm{W}} = 0.3$, whereas $b_i = 1$ denotes a strong bond with $J_i = J_{\mathrm{S}} = 1.7 > J_{\mathrm{W}}$. Fixing ϕ_0 and indexing bonds from zero ensures that extending the chain only appends new bonds, while preserving the complete bond pattern of every smaller system as a prefix.

The circuit begins in an alternating Néel state. The single-qubit part uses the parity classes $p(i) = i \bmod 2$. The active edge sets for even- and odd-bond blocks are given by:

$$\mathcal{E}_\ell(q) = \begin{cases} \{(i, i+1) \mid 0 \le i < q-1,\ i \bmod 2 = 0\}, & \text{even-bond block,} \\ \{(i, i+1) \mid 0 \le i < q-1,\ i \bmod 2 = 1\}, & \text{odd-bond block.} \end{cases} \tag{16}$$

We label each bond $(i, i+1)$ by its left endpoint i and assign it a three-entry local context centered on that bond. Let ∂ denote a missing neighbor at an open boundary and let $*$ denote a neighboring bond type that is deliberately not distinguished by the parameter-sharing rule. The possible context is listed as

$$c_i = \begin{cases} (\partial, 0, 1), & i = 0, \\ (b_{i-1}, b_i, b_{i+1}), & 1 \le i \le q-3, \\ (1, 0, \partial), & i = q-2 \text{ and } b_i = 0, \\ (*, 1, \partial), & i = q-2 \text{ and } b_i = 1. \end{cases} \tag{17}$$

The resulting context set is

$$\mathcal{C}_{\mathrm{Fib}} = \{(\partial,0,1), (0,1,0), (0,1,1), (1,0,1), (1,1,0), (1,0,\partial), (*,1,\partial)\}. \tag{18}$$

Thus, an interior context records the types of the preceding, current, and following bonds, while the three boundary contexts retain the sharing convention implemented for the open chain. The block has the form

$$\begin{aligned} U_\ell(q) = &\prod_{(i,i+1)\in\mathcal{E}_\ell(q)} R_{ZZ}(\gamma^{ZZ}_{\ell,c_i}) R_{YY}(\gamma^{YY}_{\ell,c_i}) R_{XX}(\gamma^{XX}_{\ell,c_i}) \\ &\times \prod_{i=0}^{q-1} R_Z(\delta_{\ell,p(i)}) R_Y(\beta_{\ell,p(i)}) R_Z(\alpha_{\ell,p(i)}). \end{aligned} \tag{19}$$

Each block stores six parity-shared Euler coefficients and three coefficients for each of the seven possible bond contexts, giving 27 coefficient slots. Only the coefficients associated with contexts present in the block's active edge set are used at a given system size. Beginning with the even-bond block, the 24-block program strictly alternates between even- and odd-bond blocks.

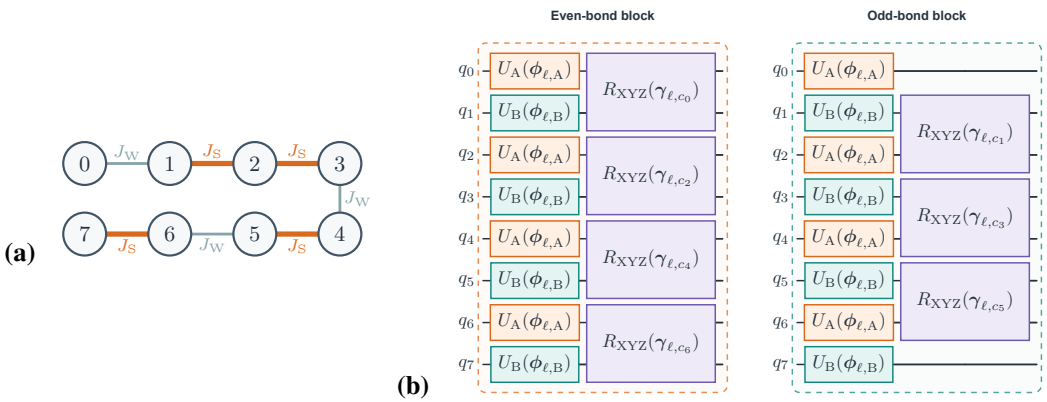

Figure 8: **Fibonacci-chain program.** Representative eight-qubit example. (a) Quasiperiodic open chain, folded for compact presentation; line styles distinguish weak and strong couplings. (b) Even- and odd-bond blocks. $U_A(\boldsymbol{\phi}_{\ell,A})$ and $U_B(\boldsymbol{\phi}_{\ell,B})$ denote $R_Z R_Y R_Z$ rotations, where $\boldsymbol{\phi}_{\ell,s} = (\alpha_{\ell,s}, \beta_{\ell,s}, \delta_{\ell,s})$ contains three independent angles for parity class $s \in \{A, B\}$, with A and B corresponding to even and odd sites. Each $R_{\mathrm{XYZ}}(\boldsymbol{\gamma}_{\ell,c_i})$ comprises R_{XX}, R_{YY}, and R_{ZZ} rotations, where $\boldsymbol{\gamma}_{\ell,c_i} = (\gamma^{XX}_{\ell,c_i}, \gamma^{YY}_{\ell,c_i}, \gamma^{ZZ}_{\ell,c_i})$ contains three independent angles assigned according to the active bond's local context c_i.

A.4 Aubry–André XXZ chain

The Aubry–André XXZ chain (Žnidarič & Ljubotina, 2018) is

$$H_{\mathrm{AA}} = \frac{1}{2}\sum_{i=0}^{q-2}(X_iX_{i+1} + Y_iY_{i+1} + \Delta Z_iZ_{i+1}) + W\sum_{i=0}^{q-1}\cos(2\pi\beta i + \phi_0)Z_i, \tag{20}$$

with $\Delta = 1$, $W = 0.5$, $\beta = (\sqrt{5}-1)/2$, and the same ϕ_0 as the Fibonacci chain. Define $\varphi_i = 2\pi\beta i + \phi_0$. Starting from an alternating Néel state, block ℓ assigns the site-dependent angle

$$\vartheta_{\ell,i} = a_{\ell,p(i)} + b_{\ell,c}\cos\varphi_i + b_{\ell,s}\sin\varphi_i, \qquad p(i) = i \bmod 2, \tag{21}$$

and applies

$$U_\ell(q) = \left[\prod_{(i,i+1)\in\mathcal{E}_\ell(q)} R_{ZZ}(\zeta_\ell)R_{YY}(\eta_\ell)R_{XX}(\eta_\ell)\right]\left[\prod_{i=0}^{q-1} R_Z(\vartheta_{\ell,i})\right]. \tag{22}$$

The two parity bases $a_{\ell,0}, a_{\ell,1}$ and the cosine and sine responses generate inhomogeneous site angles from four coefficients. All active bonds share η_ℓ for both XX and YY, preserving total magnetization, and share a separate ζ_ℓ for ZZ. Each block therefore has six coefficients. As in the Fibonacci program, 24 blocks alternate even- and odd-bond layouts, beginning with the even-bond block.

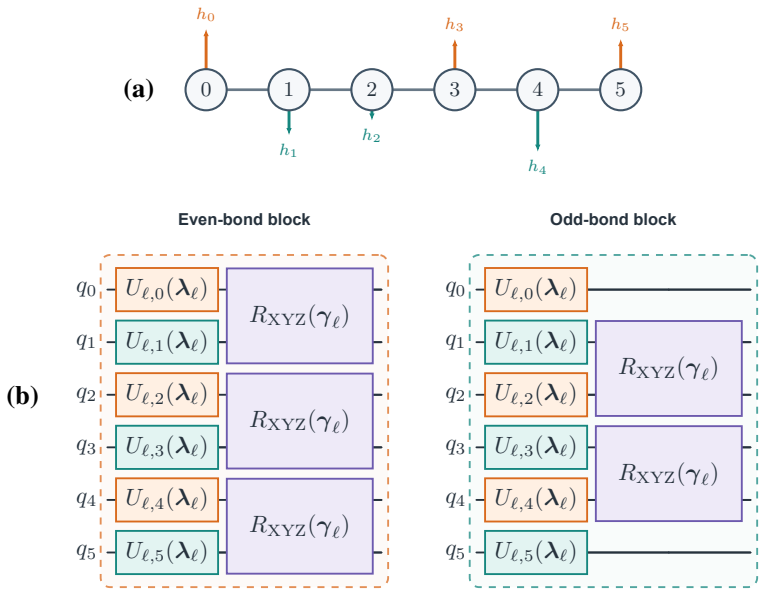

Figure 9: **Aubry–André XXZ program.** Representative six-qubit example. (a) Open XXZ chain with quasiperiodic onsite fields; arrow direction gives the field sign and arrow length indicates relative magnitude. (b) Alternating even- and odd-bond blocks. Each $U_{\ell,i}(\boldsymbol{\lambda}_\ell)$ denotes $R_Z(\vartheta_{\ell,i})$, where $\boldsymbol{\lambda}_\ell = (a_{\ell,0}, a_{\ell,1}, b_{\ell,c}, b_{\ell,s})$ contains the four coefficients that generate the site-dependent angle. Each $R_{\mathrm{XYZ}}(\boldsymbol{\gamma}_\ell)$ comprises R_{XX}, R_{YY}, and R_{ZZ} rotations with $\boldsymbol{\gamma}_\ell = (\eta_\ell, \eta_\ell, \zeta_\ell)$; the shared η_ℓ preserves total magnetization, while ζ_ℓ is learned independently.

A.5 Fermi–Hubbard chain

The Fermi–Hubbard chain (Giamarchi, 2003, Sec. 7.1.1) has $N = q/2$ physical sites and

$$H_{\mathrm{Hub}} = -t \sum_{j=0}^{N-2} \sum_{\sigma=\uparrow,\downarrow} \left(c^\dagger_{j,\sigma} c_{j+1,\sigma} + c^\dagger_{j+1,\sigma} c_{j,\sigma} \right) + U \sum_{j=0}^{N-1} n_{j,\uparrow} n_{j,\downarrow}, \tag{23}$$

with $t = 1$, $U = 4$, N particles, and $S_z = 0$. Qubits q_{2j} and q_{2j+1} encode the spin-up and spin-down modes at site j. The initial computational-basis state alternates occupied spin-up and spin-down modes across sites, placing the circuit at half filling with zero spin projection. This sector requires even N, so the supported qubit counts are divisible by four. Reported energy differences are divided by $q = 2N$, rather than by the number of physical sites.

Let $p(j) = j \bmod 2$, which assigns separate shared coefficients to even and odd sites. Superscripts $\uparrow$ and $\downarrow$ label the two spin orbitals. We define

$$\begin{aligned} F^\sigma_{s,j}(\theta) &= \exp\left[\theta\left(c^\dagger_{j+1,\sigma} c_{j,\sigma} - c^\dagger_{j,\sigma} c_{j+1,\sigma}\right)\right], \qquad \sigma \in \{\uparrow,\downarrow\}, \\ F_{d,j}(\kappa) &= \exp\left[\kappa\left(c^\dagger_{j+1,\downarrow} c^\dagger_{j+1,\uparrow} c_{j,\downarrow} c_{j,\uparrow} - \mathrm{H.c.}\right)\right]. \end{aligned} \tag{24}$$

Here, $F^\sigma_{s,j}$ coherently transfers one fermion with spin σ between sites j and $j+1$. Under the chosen Jordan–Wigner ordering, $F^\uparrow_{s,j}$ acts on $(q_{2j}, q_{2j+1}, q_{2j+2})$ and $F^\downarrow_{s,j}$ on $(q_{2j+1}, q_{2j+2}, q_{2j+3})$; the middle wire records fermionic parity. The four-wire gate $F_{d,j}$ transfers an onsite $\uparrow\downarrow$ pair between the same sites. One block applies the two spin-dependent R_Z phases and the onsite R_{ZZ} rotation at every site, followed by both single-excitation gates and the double-excitation gate on every

neighboring site pair. In operator form,

$$U_\ell(q) = \left[\overleftarrow{\prod}_{j=0}^{N-2} F_{d,j}(\kappa_{\ell,p(j)})\right]\left[\overleftarrow{\prod}_{j=0}^{N-2} F^{\downarrow}_{s,j}(\theta^{\downarrow}_{\ell,p(j)}) F^{\uparrow}_{s,j}(\theta^{\uparrow}_{\ell,p(j)})\right] \times \left[\prod_{j=0}^{N-1} R_{ZZ}(\lambda_{\ell,p(j)})_{2j,2j+1}\right]\left[\prod_{j=0}^{N-1} R_Z(\phi^{\downarrow}_{\ell,p(j)})_{2j+1} R_Z(\phi^{\uparrow}_{\ell,p(j)})_{2j}\right]. \tag{25}$$

Since excitation gates on adjacent bonds share qubits and need not commute, their application order must be specified. The left-arrow product indicates that the implementation visits the bonds in increasing order, $j = 0, 1, \ldots, N-2$, so that G_0 acts first in $\overleftarrow{\prod}_{j=0}^{N-2} G_j = G_{N-2} \cdots G_0$. Each block contains six parameterized gate families: $R_Z(\phi^{\uparrow})$, $R_Z(\phi^{\downarrow})$, $R_{ZZ}(\lambda)$, $F^{\uparrow}_{s,j}(\theta^{\uparrow})$, $F^{\downarrow}_{s,j}(\theta^{\downarrow})$, and $F_{d,j}(\kappa)$. Each family has separate coefficients for even and odd sites or bond origins, giving $6 \times 2 = 12$ coefficients per block. The deployed program repeats this layout for eight independently parameterized blocks, resulting in 96 active coefficients in total.

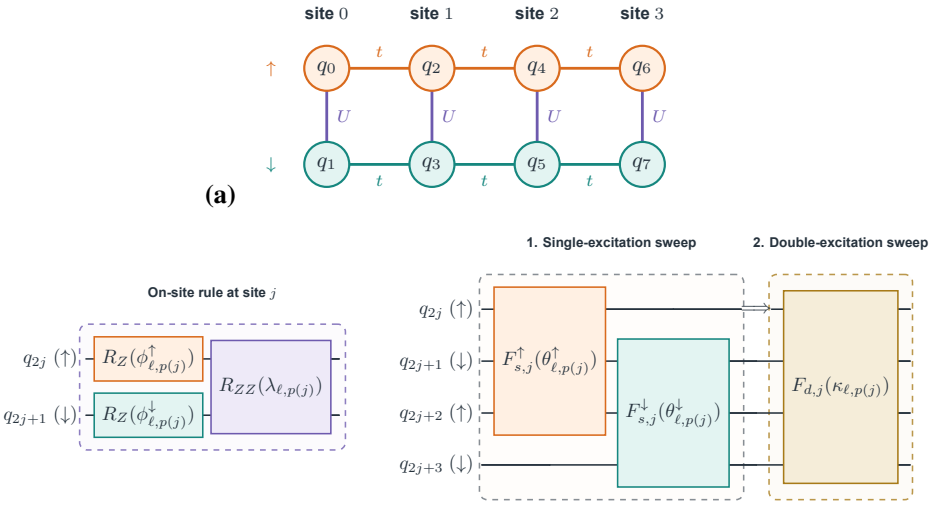

Figure 10: **Fermi–Hubbard program.** Representative four-site example encoded in eight qubits. (a) Four physical sites mapped to eight spin orbitals. Horizontal edges represent spin-preserving hopping t and vertical edges the onsite interaction U. (b) Onsite rule and the two successive bond sweeps in one program block. On bond $(j, j+1)$, the circuit applies $F^{\uparrow}_{s,j}$ and $F^{\downarrow}_{s,j}$ during the single-excitation sweep, followed by $F_{d,j}$ during the double-excitation sweep. Both sweeps visit all bonds in increasing j, while $p(j) = j \bmod 2$ selects the corresponding even- or odd-bond coefficient.

A.6 Fully connected TFIM

The fully connected TFIM is a rescaled Lipkin–Meshkov–Glick model (Dusuel & Vidal, 2005):

$$H_{\mathrm{FC}} = -J \sum_{i<j} Z_i Z_j - hq \sum_i X_i = q\left[-\frac{J}{q}\sum_{i<j} Z_i Z_j - h\sum_i X_i\right], \tag{26}$$

with $J = 1$ and $h = 2$. The prefactor q leaves the ground state unchanged relative to the mean-field-normalized Hamiltonian in brackets but makes the reported total energy scale quadratically with q.

Starting from $|+\rangle^{\otimes q}$, each block applies

$$U_\ell(q) = \left[\prod_{i=0}^{q-1} R_X(2h\beta_\ell)_i\right]\left[\prod_{0\le i<j<q} R_{ZZ}\left(-\frac{2J\gamma_\ell}{q}\right)_{ij}\right]. \tag{27}$$

All $q(q-1)/2$ pair gates share γ_ℓ, and all mixer gates share β_ℓ. The known $1/q$ factor changes the instantiated pair angle with system size without introducing another learned coefficient. Four blocks are concatenated in increasing ℓ, giving eight learned coefficients in total.

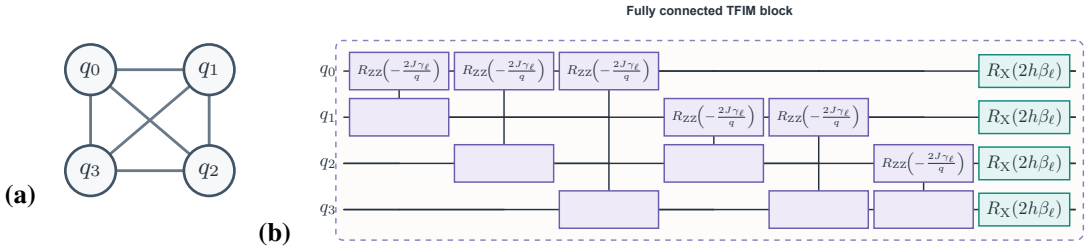

Figure 11: **Fully connected TFIM program.** Representative four-qubit example. (a) Complete interaction graph. (b) One block showing all six pairwise R_{ZZ} gates followed by the shared R_X mixer. The same rule generates $q(q-1)/2$ pair gates at size q.

A.7 Gate-count scaling

To compare the six heterogeneous programs under one convention, we decompose every complete circuit into the common gate set $\{H, X, R_X, R_Y, R_Z, \mathrm{CNOT}\}$ using PennyLane 0.44.1. Counts therefore separate single-qubit gates from CNOTs. Initial-state preparation is included, whereas measurements, repeated shots, and hardware routing are excluded. We apply no gate cancellation, rotation merging, zero-angle pruning, connectivity-aware routing, or other compiler optimization. The results characterize a consistent logical decomposition rather than minimum gate counts or device-specific execution costs.

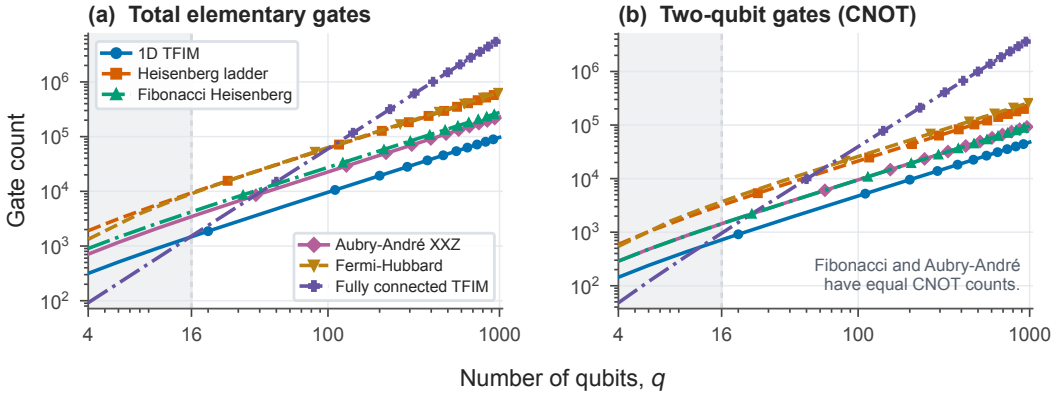

Figure 12: **Gate counts after a common logical decomposition.** (a) Total single-qubit and CNOT gates; (b) CNOT gates alone. Every curve includes all saved program blocks and product-state preparation. Both axes are logarithmic, and shading marks $q \leq 16$.

Figure 12 shows approximately linear growth for the five local-interaction programs: at fixed symbolic depth, enlarging the system adds a number of local operations proportional to q. The fully connected TFIM instead applies $q(q-1)/2$ pairwise R_{ZZ} gates in each of four blocks. Since the selected decomposition uses two CNOTs per R_{ZZ}, its CNOT count is $4q(q-1)$ and grows quadratically.

B Agent-VQE learning and frozen deployment

B.1 Learning procedure

For the reported experiments, we use GPT-5.6 Sol as LLM agent (OpenAI, 2026).

Algorithm 1 specifies source-side program search and frozen deployment. The source-size set S is used for coefficient fitting and program selection, while S_{target} contains the larger sizes evaluated after freezing; the two sets are nonempty and disjoint. A candidate program is $\mathcal{P} = (L, R, \mathbf{w})$, with context extractor $\mathcal{C}$ included in R. Candidate layouts and parameter-assignment rules must satisfy the specified physical and gate constraints. We write Ref for the reference solver, reserving R for parameter assignment, and obtain $E_{\mathrm{ref}}(q) = \mathsf{Ref}(H_q)$ under the locked reference protocol.

Source objectives, acceptance checks, and search budget. To clarify the two stages of Algorithm 1, we specify the source-side optimization objective, the acceptance checks for S and S_{target}, and the search budget B.

Coefficient fitting is performed on the source-size set S using the total-energy objective $\mathcal{J}$ in Equation 6 or a benchmark-specific variant. To express the source and target checks compactly, let T denote either S or S_{target}, with the corresponding threshold $\epsilon_{\mathrm{gate}} \in \{\epsilon_{\mathrm{src}}, \epsilon_{\mathrm{tgt}}\}$. Candidates are first required to produce finite outputs and satisfy the prescribed physical and gate constraints. For candidates passing these validations, the following predicate defines the energy-error check:

$$\mathrm{Pass}(\mathcal{P}, T, \epsilon_{\mathrm{gate}}) \quad \Longleftrightarrow \quad \epsilon_q(\mathcal{P}) \le \epsilon_{\mathrm{gate}} \quad \text{for every } q \in T. \tag{28}$$

Candidates passing the source check are stored in $\mathcal{K}_{\mathrm{src}}$. After the source-side search ends, the final program is selected from $\mathcal{K}_{\mathrm{src}}$ according to the source objective value and frozen before evaluation on S_{target}.

The source-side budget B covers all design trials and coefficient-optimization restarts. Each trial receives a bounded allocation of the remaining budget, and proposing a new design or restarting optimization does not reset the total budget B. For every trial, $\mathcal{M}$ records the candidate design, initialization or checkpoint, source objective, per-size errors, resource use, and termination status, while $\mathcal{S}$ tracks the latest feedback and remaining budget. If $\mathcal{K}_{\mathrm{src}}$ remains empty when the search ends, the procedure returns FAIL with the recorded trial history, and no program is deployed.

Algorithm 1 Agent-VQE

Input: $\{H_q\}$; physical and gate constraints; source sizes S and target sizes S_{target}; thresholds ϵ_{src}, ϵ_{tgt}; reference solver Ref; source-side budget B.
Output: Frozen program, source report, target audit, and pass/fail status.

Human protocol lock

Lock the physical and gate constraints, initial states, splits, estimator, gates, and reference protocol.
Declare actions $\mathcal{A}$, trial limits, optimizer/restart settings, budget B, and selection/stopping rules.
Prohibit target-side gradients, updates, design changes, or reselection.

Agent initialization

Action: Check that source and target sets are nonempty and disjoint; build H_q, $E_{\text{ref}}(q)$ for $q \in S$; fail on invalid input.
State: Initialize $A = (\mathcal{S}, \mathcal{A}, \mathcal{M}, \mathcal{W})$ with task, candidate, source feedback, and resources in $\mathcal{S}$; set $\mathcal{M}, \mathcal{K}_{\text{src}} \leftarrow \emptyset$.
[$\mathcal{W}$] marks pretrained scientific knowledge used to suggest designs, not to compute energies.
For a fixed design, L, R, and the number of shared coefficients are independent of q.

Stage A: size-extensible program search on small systems

while a source trial is permitted within the remaining budget B **do**
  State: Read source feedback, resources, and $\mathcal{M}$; **Action:** choose a new design, revision, or refit.
  if proposing or revising a design **then**
    Action: [$\mathcal{W}$] Use interaction structure and symmetries to propose relative gate patterns L.
    Action: [$\mathcal{W}$] Use local site/bond structure to propose $\mathcal{C}$ and sharing rule R, informed by source feedback; check size extensibility and physical constraints.
  end if
  Action: Fix (L, R); initialize or restore one shared $\mathbf{w}$ and call the numerical optimizer with declared restarts and a bounded trial allocation.
  for each objective query $\mathbf{w}$ from the optimizer within that allocation **do**
    for each source size $q \in S$ **do**
      Action: Expand the relative patterns p_ℓ in L on the q-qubit interaction graph.
      Action: Set $c_e = \mathcal{C}(H_q, e)$ and $\theta_{\ell,e} = R(\ell, p_\ell, \tau_e, c_e; \mathbf{w})$ as in Eq. (4).
      Action: Evaluate $E_q(L, R, \mathbf{w})$ using the generated circuit and prescribed initial state.
    end for
    Action: Return the declared source objective $\mathcal{J}_S(L, R, \mathbf{w})$, defined by Equation 6 or its benchmark-specific variant, to the numerical optimizer, which updates the shared $\mathbf{w}$.
  end for
  State: Update feedback, remaining budget, and trial count; **Memory:** log all actions, checkpoints, results, and failures.
  Action: Keep the best admissible fitted $\mathcal{P}$; if none exists, continue. Reevaluate all source errors and record them in $\mathcal{S}$ and $\mathcal{M}$.
  if $\text{Pass}(\mathcal{P}, S, \epsilon_{\text{src}})$ **then**
    Memory: Add an immutable snapshot of $\mathcal{P}$ and its source score to $\mathcal{K}_{\text{src}}$.
  end if
end while
Memory: Lock $\mathcal{K}_{\text{src}}$ and end source search; fail if the pool is empty.

Stage B: source selection and frozen deployment

Action: Select $\widehat{\mathcal{P}}$ from $\mathcal{K}_{\text{src}}$ by the locked source-score rule; freeze all rules and coefficients before target evaluation. Never return to A.
Action: Serialize $\widehat{\mathcal{P}} = (\widehat{L}, \widehat{R}, \widehat{\mathbf{w}})$, including $\widehat{\mathcal{C}}$, hashes, configuration, logs, and zero-update declaration.
for each $q \in S_{\text{target}}$ **do**
  Action: Expand the frozen patterns in $\widehat{L}$ on the larger interaction graph, adding gate instances but no learned coefficients.
  Action: Use $\widehat{\mathcal{C}}$, $\widehat{R}$, and the unchanged $\widehat{\mathbf{w}}$ to assign every gate angle by Eq. (4).
  Action: Evaluate $E_{\widehat{\mathcal{P}}}(q)$ by forward calls only; obtain $E_{\text{ref}}(q)$ independently and compute errors as in Eq. (8); mark r_q undefined if $E_{\text{ref}}(q) = 0$.
  Memory: Log energies, errors, gates, hashes, and prohibited-operation counts; never refit or reselect.
end for
return the frozen artifact and reports; report PASS if $\text{Pass}(\widehat{\mathcal{P}}, S_{\text{target}}, \epsilon_{\text{tgt}})$ holds, hashes match, and prohibited-operation counts are zero; report FAIL otherwise.
Size extensibility is structural; energy accuracy is checked against independent references.

B.2 AGENT STATE, MEMORY, AND ACTIONS

This subsection specifies the state $\mathcal{S}$, action space $\mathcal{A}$, and memory $\mathcal{M}$ introduced in Section 4. The state $\mathcal{S}$ contains the problem specification, current circuit program, source-system feedback, and remaining computational budget. The memory $\mathcal{M}$ stores evaluated designs, optimized coefficients, per-size energies, convergence diagnostics, and circuit costs. Tables 2 and 3 summarize the information and actions available to the agent.

Table 2: Information available to the Agent-VQE design loop.

Component	Contents	Role
Problem specification	Hamiltonians, constraints, source sizes, and objective.	Define the task and admissible design space.
Current program	Layout rules L, parameter-assignment rule R, and shared coefficients $\mathbf{w}$.	Describe the candidate under evaluation.
Feedback, memory, and resources	Per-size energies, optimization diagnostics, previous candidates and checkpoints, circuit costs, tools, and remaining budget.	Support candidate comparison and resource allocation.
Scientific knowledge $\mathcal{W}$	Pretrained knowledge of Hamiltonian structure, symmetries, and circuit-design principles.	Inform candidate layouts and parameter-assignment rules.

Table 3: Principal actions in the Agent-VQE design loop.

Action	Operations	Purpose
Modify design	Add, replace, or remove layouts; revise parameter assignment.	Explore discrete circuit programs.
Fit coefficients	Select initialization, checkpoint, optimizer, and budget.	Evaluate a design after coefficient fitting.
Evaluate and compare	Re-evaluate the fitted candidate across the source sizes and compare its energies, errors, and diagnostics with previous candidates.	Compare candidate performance across source sizes.
Select or stop	Retain, reject, restore, or freeze a candidate.	Produce the final circuit program.

Execution loop. The agent proposes a discrete circuit design and its executable implementation, while the numerical optimizer fits only the shared coefficients $\mathbf{w}$ on the source systems. The returned energies and diagnostics update $\mathcal{S}$ and $\mathcal{M}$ for the next design decision. This cycle continues until the stopping condition or computational budget is reached. Once the final program is selected and frozen, target evaluations do not re-enter the design loop or trigger parameter updates.

B.3 FROZEN DEPLOYMENT AND NUMERICAL EVALUATION

For a selected circuit program $\mathcal{P} = (L, R, \mathbf{w})$ and target size q, deployment instantiates the layout L and parameter-assignment rules R at that size while keeping the learned coefficients $\mathbf{w}$ fixed. No target-size gradients, parameter updates, or architecture revisions are performed within this evaluation. The resulting circuit prepares $|\psi_{\mathcal{P},q}\rangle$, whose energy is

$$E_{\mathcal{P}}(q) = \langle \psi_{\mathcal{P},q} | H_q | \psi_{\mathcal{P},q} \rangle. \tag{29}$$

We compare this value with a ground-state reference $E_{\mathrm{ref}}(q)$ using $\epsilon_q = |E_{\mathcal{P}}(q) - E_{\mathrm{ref}}(q)|/q$ and $r_q = |E_{\mathcal{P}}(q) - E_{\mathrm{ref}}(q)|/|E_{\mathrm{ref}}(q)|$.

For the circuit energies $E_{\mathcal{P}}(q)$, the one-dimensional TFIM is simulated exactly through its free-fermion representation, and the fully connected TFIM is simulated exactly in the permutation-symmetric subspace (Pfeuty, 1970; Dusuel & Vidal, 2005). The remaining four models use MPS circuit simulation (Vidal, 2003). Reference ground-state energies $E_{\mathrm{ref}}(q)$ are computed exactly for the two TFIM benchmarks. For the open Fermi–Hubbard chain, we numerically solve the finite-size Bethe-ansatz equations to derive $E_{\mathrm{ref}}(q)$ (Schulz, 1985; Deguchi & Yue, 1997). For the Heisenberg ladder, Fibonacci, and Aubry–André models, we use DMRG approximations (White, 1992).

The frozen rules may assign different gate angles to different local environments at a target size, but neither the rules nor the learned coefficients are updated during deployment. Reference solvers are used only to evaluate the resulting circuit and do not provide feedback to the frozen program. Reported errors are measured relative to $E_{\mathrm{ref}}(q)$ and therefore do not constitute an exact ground-state guarantee when the reference energy is approximate.

Table 4 reports the complete large-system data underlying Figures 2 and 3.

Table 4: **Large-system frozen-deployment results.** We report total circuit and reference energies at representative target sizes, together with the absolute difference per qubit ϵ_q and relative difference r_q. Differences are computed from the unrounded energies.

q	$E_{\mathcal{P}}(q)$	$E_{\mathrm{ref}}(q)$	ϵ_q	r_q (%)
One-dimensional TFIM				
100	−126.8613	−126.9619	0.00101	0.0792
200	−254.0590	−254.2852	0.00113	0.0890
300	−381.2566	−381.6089	0.00117	0.0923
400	−508.4543	−508.9328	0.00120	0.0940
500	−635.6520	−636.2567	0.00121	0.0950
600	−762.8496	−763.5806	0.00122	0.0957
700	−890.0473	−890.9045	0.00122	0.0962
800	−1017.245	−1018.228	0.00123	0.0966
900	−1144.443	−1145.552	0.00123	0.0969
1000	−1271.640	−1272.876	0.00124	0.0971
Heisenberg ladder				
100	−114.8826	−114.9421	5.95×10^{-4}	0.0518
200	−230.2716	−230.4251	7.67×10^{-4}	0.0667
300	−345.9673	−345.7361	7.71×10^{-4}	0.0669
400	−461.5096	−461.2006	7.72×10^{-4}	0.0670
500	−576.6651	−577.0518	7.73×10^{-4}	0.0671
600	−692.5941	−692.1296	7.74×10^{-4}	0.0671
700	−808.1363	−807.5941	7.75×10^{-4}	0.0671
800	−923.6786	−923.0586	7.75×10^{-4}	0.0672
900	−1038.523	−1039.221	7.75×10^{-4}	0.0672
1000	−1154.763	−1155.387	6.24×10^{-4}	0.0540
Fibonacci Heisenberg chain				
100	−117.7961	−118.9574	0.0116	0.976
200	−236.6975	−238.6681	0.00985	0.826
300	−354.7809	−357.5815	0.00934	0.783
400	−473.7606	−477.5538	0.00948	0.794
500	−594.0538	−598.6571	0.00921	0.769
600	−712.9591	−718.3722	0.00902	0.754
700	−831.6775	−838.0829	0.00915	0.764
800	−949.7609	−956.9959	0.00904	0.756
900	−1068.741	−1076.969	0.00914	0.764
1000	−1188.668	−1198.072	0.00940	0.785
Aubry–André XXZ chain				
100	−94.47564	−94.75212	0.00276	0.292

Continued on next page

Table 4 continued

q	$E_{\mathcal{P}}(q)$	$E_{\mathrm{ref}}(q)$	ϵ_q	r_q (%)
200	-189.1188	-189.6648	0.00273	0.288
300	-283.7129	-284.5607	0.00283	0.298
400	-378.5744	-379.7145	0.00285	0.300
500	-473.0709	-474.4872	0.00283	0.298
600	-567.8540	-569.5940	0.00290	0.305
700	-662.5395	-664.5503	0.00287	0.303
800	-757.1019	-759.4088	0.00288	0.304
900	-851.9736	-854.5791	0.00289	0.305
1000	-946.4822	-949.3618	0.00288	0.303
Fermi–Hubbard chain				
100	-26.90581	-28.32022	0.0141	4.99
200	-54.05637	-57.00531	0.0147	5.17
300	-81.20695	-85.69130	0.0149	5.23
400	-108.3575	-114.3775	0.0151	5.26
500	-135.5082	-143.0638	0.0151	5.28
600	-162.6587	-171.7502	0.0152	5.29
700	-189.8093	-200.4366	0.0152	5.30
800	-216.9598	-229.1230	0.0152	5.31
900	-244.1105	-257.8095	0.0152	5.31
1000	-271.2790	-286.4959	0.0152	5.31
Fully connected TFIM				
100	-20008.41	-20008.41	2.43×10^{-5}	1.22×10^{-5}
200	-80016.98	-80016.99	2.98×10^{-5}	7.45×10^{-6}
300	-1.800256×10^{5}	-1.800256×10^{5}	3.18×10^{-5}	5.31×10^{-6}
400	-3.200341×10^{5}	-3.200341×10^{5}	3.29×10^{-5}	4.11×10^{-6}
500	-5.000427×10^{5}	-5.000427×10^{5}	3.35×10^{-5}	3.35×10^{-6}
600	-7.200513×10^{5}	-7.200513×10^{5}	3.40×10^{-5}	2.83×10^{-6}
700	-9.800599×10^{5}	-9.800599×10^{5}	3.43×10^{-5}	2.45×10^{-6}
800	-1.280068×10^{6}	-1.280068×10^{6}	3.45×10^{-5}	2.16×10^{-6}
900	-1.620077×10^{6}	-1.620077×10^{6}	3.47×10^{-5}	1.93×10^{-6}
1000	-2.000086×10^{6}	-2.000086×10^{6}	3.49×10^{-5}	1.74×10^{-6}

Tensor-network settings and bond-dimension checks. We control the accuracy of the MPS simulations of circuit energies $E_{\mathcal{P}}(q)$ through Quimb's truncation cutoff and maximum bond dimension (Gray, 2018). Under the `rsum2` convention in Quimb 1.15, the cutoff bounds the relative sum of squared Schmidt coefficients discarded at each tensor split, while χ sets the maximum number of Schmidt coefficients retained at each bond. We use a cutoff of 10^{-8}, $\chi = 128$, and complex128 arithmetic. At $q = 1000$, we repeat these simulations with $\chi \in \{128, 256, 512, 1024\}$. The Fibonacci, Aubry–André, and Fermi–Hubbard energies are unchanged at the reported precision from $\chi = 128$ onward. For the Heisenberg ladder, the $\chi = 128$ result differs from $\chi = 1024$ by 1.69×10^{-6} per qubit and is unchanged from $\chi = 256$ onward. This bond-dimension effect is substantially smaller than the corresponding circuit–reference difference, $\epsilon_{1000} = 6.24 \times 10^{-4}$; we therefore use $\chi = 128$ throughout to balance numerical accuracy and computational cost.

For the DMRG references, we use a local compression cutoff of 10^{-8} under Quimb's `sum2` convention. DMRG estimates the ground-state energy by iteratively optimizing the MPS over successive sweeps. We set the energy-convergence tolerance to 10^{-7} and the maximum number of sweeps to 150; the optimization terminates earlier when the absolute change in total energy between consecutive sweeps falls below this tolerance. To balance numerical accuracy and computational cost in the dense $q = 100$–1000 evaluation, the plotted references use $\chi = 16$ for the Heisenberg ladder at $q = 200, \ldots, 1000$ (and $\chi = 128$ at $q = 100$), $\chi = 8$ for Fibonacci, and $\chi = 128$ for Aubry–André. At $q = 1000$, we evaluate bond-dimension sensitivity using $\chi \in \{8, 16, 32, 64, 128, 256, 512, 1024\}$. Table 5 compares each plotted reference with the corresponding $\chi = 1024$ result and reports the smallest tested bond dimension that agrees with it

within 10^{-8} per qubit. The finite-bond residuals are substantially smaller than the corresponding circuit–reference differences in Figure 3, indicating that DMRG truncation does not account for the reported circuit errors.

Table 5: DMRG bond-dimension audit at $q = 1000$. Here χ_{main} is the bond cap used for the plotted reference, $\chi_{10^{-8}}$ is the smallest tested cap within 10^{-8} per qubit of the $\chi = 1024$ result, and $\delta_{\mathrm{main}} = |E_{\mathrm{ref}}^{(\chi_{\mathrm{main}})} - E_{\mathrm{ref}}^{(1024)}|/q$.

Model	χ_{main}	$\chi_{10^{-8}}$	δ_{main}
Heisenberg ladder	16	64	3.28×10^{-5}
Fibonacci chain	8	32	1.70×10^{-5}
Aubry–André	128	256	2.00×10^{-8}

B.4 Controlled test of the SC-ADAPT extrapolation rule

The Fibonacci experiment in Section 5.2 tests a simplified layerwise bond-sharing proxy motivated by the frozen-extrapolation rule of SC-ADAPT-VQE (Farrell et al., 2024); it does not reproduce the SC-ADAPT-VQE operator-pool search on the Fibonacci Hamiltonian. All comparison arms use the same local gate families, alternating even- and odd-bond layouts, and parity-shared single-qubit Euler rotations. In layer ℓ, the layerwise bond-shared baseline applies the same angle triplet $\boldsymbol{\gamma}_\ell = (\gamma_\ell^{XX}, \gamma_\ell^{YY}, \gamma_\ell^{ZZ})$ to the R_{XX}, R_{YY}, and R_{ZZ} gates on every active bond. Agent-VQE instead assigns $\boldsymbol{\gamma}_{\ell,b_i}$ according to whether the active bond is weak ($b_i = 0$) or strong ($b_i = 1$). This two-context rule is used only for the controlled comparison; the final Fibonacci program in Appendix A uses the richer context c_i defined in Equation 17.

Both depth-12 programs are optimized on $q \in \{6, 8, 10, 12\}$. Each bond-shared layer stores six single-qubit angles and one shared $R_{XX}/R_{YY}/R_{ZZ}$ angle triplet, giving $12 \times 9 = 108$ coefficients; each conditioned layer stores separate triplets for weak and strong bonds, giving $12 \times 12 = 144$. To control for parameter count, a second layerwise bond-shared baseline uses 16 layers and $16 \times 9 = 144$ coefficients. All coefficients and layouts are then frozen and evaluated at $q \in \{14, 16, 20, 40, 60, 80, 100\}$, with no target-side gradients, parameter updates, or structure search. Energy differences use exact references through $q = 20$ and DMRG references at larger sizes. The same-depth and equal-parameter baselines both remain substantially less accurate than the conditioned rule; at $q = 100$, their energy differences are 2.44×10^{-1} and 2.66×10^{-1} per qubit, respectively, compared with 2.70×10^{-2} for the conditioned program.

C HARDWARE EXPERIMENTS

C.1 DEVICE AND PHYSICAL-QUBIT CHAINS

All hardware experiments use the OriginQ WK_C180 backend and the following ordered path through its coupling graph. Each integer is a physical-qubit index:

$$\mathcal{C}_{38} = (3, 13, 4, 14, 23, 32, 41, 51, 60, 69, 59, 50, 40, 49, 58, 68, 77, 87, 96,\\ 105, 114, 123, 132, 141, 131, 122, 113, 104, 94, 103, 93, 102, 111, 120, 110, 119, 128, 138).$$

Each q-qubit experiment uses the first q entries of $\mathcal{C}_{38}$.

C.2 MEASURED-ENERGY FITTING AND EXTRAPOLATION

This subsection details the measured-energy extrapolation experiment in Section 5.3 and Figure 4. At each size q, E_q is the mean of five repeated measurements, with sample standard deviation s_q and standard error of the mean (SEM) $s_q/\sqrt{5}$. An ordinary-least-squares fit of $\widehat{E}(q) = aq + b$ to the 13 means at $q = 4, 5, \ldots, 16$ gives $a = -0.22724066$, $b = 1.96942198$, $R^2 = 0.7944$, and RMSE 0.4325. The shaded region is the pointwise 95% confidence interval for the fitted mean, estimated from the fit residuals.

For the 11 held-out sizes $\mathcal{T} = \{18, 20, \ldots, 38\}$, Figure 4(b) reports the signed per-qubit residual $\delta_q = (E_q - \widehat{E}(q))/q$. Averaging over $\mathcal{T}$ gives $\mathrm{MAE}_{/q} = |\mathcal{T}|^{-1} \sum_q |\delta_q| = 0.041832$ and $\mathrm{RMSE}_{/q} = (|\mathcal{T}|^{-1} \sum_q \delta_q^2)^{1/2} = 0.053102$. Table 6 lists the underlying measurements and predictions.

Table 6: OriginQ hardware measurements and affine predictions. The *Fit* rows determine $\widehat{E}(q) = aq + b$, and the *Held-out* rows test its fixed extrapolation. Each energy and SEM is computed from five repeats; the final column gives the signed residual per qubit.

Regime	q	E_q	Repeat SEM	$\widehat{E}(q)$	$(E_q - \widehat{E}(q))/q$
Fit	4	1.1964	0.0152	1.0605	+0.03399
	5	0.7970	0.0337	0.8332	−0.00724
	6	0.9092	0.0647	0.6060	+0.05054
	7	0.9480	0.0375	0.3787	+0.08132
	8	0.1594	0.0489	0.1515	+0.00099
	9	−0.2024	0.0465	−0.0757	−0.01407
	10	−0.7358	0.0337	−0.3030	−0.04328
	11	−1.1186	0.0631	−0.5302	−0.05349
	12	−1.3076	0.0495	−0.7575	−0.04584
	13	−1.1122	0.0673	−0.9847	−0.00981
	14	−1.4754	0.0522	−1.2119	−0.01882
	15	−1.3556	0.0451	−1.4392	+0.00557
	16	−0.6412	0.0292	−1.6664	+0.06408
Held-out	18	−0.7382	0.0583	−2.1209	+0.07682
	20	−0.7248	0.0473	−2.5754	+0.09253
	22	−1.1546	0.0601	−3.0299	+0.08524
	24	−2.8374	0.0404	−3.4844	+0.02696
	26	−3.7186	0.0582	−3.9388	+0.00847
	28	−4.4486	0.0878	−4.3933	−0.00197
	30	−4.8610	0.1199	−4.8478	−0.00044
	32	−5.7878	0.0534	−5.3023	−0.01517
	34	−6.8800	0.0663	−5.7568	−0.03304
	36	−8.2388	0.1207	−6.2112	−0.05632
	38	−9.0672	0.1659	−6.6657	−0.06320

D WORKED AGENT TRAJECTORY FOR THE AUBRY–ANDRÉ XXZ CHAIN

This case study summarizes a human-guided, agent-assisted development trajectory for the Aubry–André XXZ chain using the program representation $\mathcal{P} = (L, R, \mathbf{w})$ and agent definition $A = (\mathcal{S}, \mathcal{A}, \mathcal{M}, \mathcal{W})$. Its pilot experiments and optimization objectives are specific to this run rather than required components of the general framework. The held-out checks below are diagnostics of an already frozen program, not a separate candidate-selection stage.

The colored labels distinguish observed state, agent actions, and stored memory. The $[\mathcal{W}]$ labels identify the scientific knowledge relevant to the design choices.

Example: Aubry--André XXZ agent trace

[HUMAN / 1. Protocol]

Human protocol: Select the open Aubry--André XXZ chain with exchange scale $J = 1$ in the Hamiltonian convention of Appendix A, $\Delta = 1$, $W = 0.5$, $\beta = (\sqrt{5}-1)/2$, and $\phi_0 = \sqrt{3}/10 \approx 0.17320508075688773$. Here W is the field amplitude, whereas $\mathcal{W}$ denotes the agent's scientific knowledge. Specify source sizes $S = \{4, 6, 8, 10, 12, 14, 16\}$ and the strict source criterion $\epsilon_q < \epsilon_{\text{src}} = 1.7 \times 10^{-3}$ for every $q \in S$. Throughout this record, ϵ_q is the reference-based error in Equation 8; numerical reference energies are denoted by $E_{\text{ref}}(q)$ rather than identified with exact $E_0(q)$.

[AGENT / 2. Model implementation and size extensibility]

Action: Implement H_q, an alternating Néel initial state, nearest-neighbor brickwork layouts, and first-order Fourier context. **[$\mathcal{W}$]** The relevant design priors are the chain's local interactions, its two site-parity classes, and the sinusoidal onsite field. Use $p(i) = i \bmod 2$ and $\varphi_i = 2\pi\beta i + \phi_0$, as in Equation 21, rather than using a gate angle to denote the field phase. The rule $\mathcal{C}$ supplies these local descriptors to R. **State:** The resulting program depends on relative nearest-neighbor relations, parity, and $\cos\varphi_i, \sin\varphi_i$. Expanding it to a new q therefore creates gate instances without introducing an independently trained coefficient for every new qubit.

[AGENT / 3. Shared-parameter pilot]

Action: Propose a depth-six shared-parameter program on pilot sizes $q \in \{6, 8, 10\}$. Each layer has six parity-shared single-qubit Euler coefficients and three shared entangler coefficients, giving $6 \times 9 = 54$ coefficients. Invoke numerical optimization with one recorded initialization/restart. The pilot objective is the mean energy per qubit,

$$\mathcal{J}_{\text{pilot}}(L, R, \mathbf{w}) = \frac{1}{3} \sum_{q \in \{6,8,10\}} \frac{E_q(L, R, \mathbf{w})}{q}. \tag{30}$$

State: The reported objective decreases from -0.4431495858 to -0.8556790192. **Memory:** Store the fitted program, its coefficients, and the pilot results for subsequent initialization. This pilot-specific normalization differs from the total-energy objective in Equation 6.

[AGENT / 4. Fourier-conditioned pilot]

Action: Copy the parity-shared rotation and entangler coefficients from the first pilot. Extend R with a cosine and sine response for each of the three Euler angles. **[$\mathcal{W}$]** First-order Fourier responses provide a parameter-sharing rule adapted to the sinusoidal field while retaining a coefficient count independent of q. The six added response coefficients give 15 coefficients per layer and $6 \times 15 = 90$ at depth six. In the first recorded restart, initialize the response coefficients to zero and optimize the same pilot objective on the same sizes. **State:** The mean energy per qubit decreases further to

Example: Aubry–André XXZ agent trace (continued)

-0.9120881489. **Memory:** Retain this conditioned checkpoint for the final program construction.

[AGENT / 5. Full-source program construction]

Action: Reparameterize the conditioned pilot as the Fourier-conditioned, magnetization-preserving program in Equations 21 and 22, and expand its depth to 24. **[$\mathcal{W}$]** The conserved total Z magnetization motivates R_Z site rotations and a common coefficient η_ℓ for paired R_{XX} and R_{YY} rotations. Each layer stores

$$(a_{\ell,0}, a_{\ell,1}, b_{\ell,c}, b_{\ell,s}, \eta_\ell, \zeta_\ell), \qquad \ell = 1, \ldots, 24, \tag{31}$$

giving a 24×6 table of 144 trainable coefficients. Initialize the first six layers by the recorded pilot-to-final parameter map and set all coefficients in the 18 added layers to zero.

[AGENT / 6. Joint source optimization]

Action: Fit the same 144 coefficients jointly on all $q \in S$ using L-BFGS-B, coefficient bounds $[-\pi, \pi]$, a smooth-max objective with temperature 5×10^{-4}, and adjoint gradients. **State:** The run records 224 objective/gradient evaluations and stops with

$$\max_{q \in S} \epsilon_q(\mathcal{P}) = 1.6920427687 \times 10^{-3} < 1.7 \times 10^{-3}. \tag{32}$$

Memory: Save the source-admitted program and its optimization record, then freeze its layout, context rule, parameter-assignment rule, and coefficients.

[AGENT / 7. Zero-update deployment]

Action: Load the frozen program JSON and instantiate it at $q \in S_{\text{target}} = \{100, 200, \ldots, 1000\}$ using the same recorded r1_o0/r1_o1 nearest-neighbor brickwork rules, corresponding to the alternating even- and odd-bond layouts. Compute site angles with the frozen Fourier rule and coefficients from Equation 21. Only MPS forward circuit-energy evaluation is performed: there is no target-side architecture search, gradient evaluation, L-BFGS-B, Rotosolve, or parameter update. Increasing q changes the number of gate instances, not the 144 learned coefficients or the program rules.

[AGENT / 8. Independent target audit]

Action: Evaluate the frozen circuit energies using MPS and compare them with independently computed DMRG reference energies, following the numerical settings in Appendix B.3. Both target-energy and reference calculations use a cutoff of 10^{-8}; the reference energy-convergence tolerance is 10^{-7}, and the formal dense audit uses reference bond dimension $\chi = 128$. **State:** All ten target sizes pass the relaxed criterion $\epsilon_q < \epsilon_{\text{tgt}} = 10^{-1}$. The largest error is $\epsilon_{600} = 2.9000778454 \times 10^{-3}$, while $\epsilon_{1000} = 2.8795350372 \times 10^{-3}$. **Memory:** Report these target outcomes.