Measurement-Optimizer Co-Design

Variational Quantum Algorithms · Module 10 — Vqa As Adaptive Control Systems

Node 10.8

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** VQA as Adaptive Control Systems **Node:** 10.8

This is the **closing node** of Module 10 and of the VQA course. It takes the most speculative position of the entire course: **measurement and optimization are not separate concerns; they should be co-designed**.

The standard VQA stack treats measurement as an oracle — you prepare a parameterized state, measure it, get an expectation value, pass it to the optimizer. The measurement is a fixed procedure (parameter shift rule or shot averaging), and the optimizer treats the output as an opaque scalar.

This separation is **suboptimal**. The optimizer *could* benefit from knowing the measurement structure: which Hamiltonian terms are being measured, which shots carry the most gradient information, how to weight the shots to make the update more informative. Conversely, the measurement *could* be adaptive to the optimizer's current state: "the optimizer is currently looking for a large gradient, so allocate more shots to terms with large coefficients".

Co-design means designing the measurement procedure and the optimizer *jointly*, as one system with shared information. It's the logical extension of the closed-loop architecture (Module 10.5) to the boundary between Levels 2 and 3.

This node is the **most forward-looking** of the course. The co-design paradigm is not yet standard; it's a research direction. But it's a direction that follows naturally from the thesis's principles, and it's where the next generation of VQA architectures is likely to go.

The Standard Decoupling

In the standard VQA architecture, measurement and optimization are cleanly decoupled:

Measurement side: - Fix a Hamiltonian decomposition $H = \sum_i c_i P_i$. - For each Pauli term P_i, compute $\langle P_i \rangle$ via a parameter-shift gradient or direct estimation. - Average over shots to reduce noise. - Return the cost $C = \sum_i c_i \langle P_i \rangle$.

Optimizer side: - Receive C (and perhaps gradient) as scalar input. - Compute the parameter update. - Return new parameters.

The interface: a scalar cost (and gradient vector). Simple, modular, easy to reason about.

What's lost in the decoupling:

1. The optimizer doesn't know *which* Pauli terms contributed most to the gradient. If it did, it could ask for more precision on the important terms.
2. The optimizer doesn't know *which* shots were the most informative. If it did, it could weight them differently.

3. The measurement doesn't know what the optimizer *needs*. It estimates the cost with a fixed precision regardless of whether the optimizer needs high precision (near convergence) or low precision (far from minimum).
4. The measurement doesn't adapt to the optimizer's state. If the optimizer is in a flat region, measurement could use different observables than if it's in a steep region.

Each lost piece of information is a missed optimization opportunity.

Co-Design Principles

A co-designed measurement-optimizer system shares information across the interface. The principles:

1. Shot allocation is coupled to the optimizer's current state

Not all Hamiltonian terms are equally important at each iteration. If the current gradient comes mostly from terms with coefficients c_1, c_3, c_5, more shots should go to those terms. If the gradient comes mostly from terms with small variance, those shots are more valuable.

Formalization. The shot allocation problem is:

$$\min_{\{n_i\}} \mathrm{Var}[\hat{g}] \quad \text{subject to} \quad \sum_i n_i = N_{\text{total}},$$

where $\hat{g}$ is the gradient estimator and n_i is the number of shots for term i. The optimal allocation is proportional to $|c_i|\sqrt{\mathrm{Var}(P_i)}$.

2. Observable selection is coupled to the optimizer's direction

The optimizer doesn't need to measure the full cost at every iteration — sometimes it only needs a gradient in a specific direction. If the direction is known, a single observable measurement in the right basis gives the gradient component.

Technique. *Rotation-aware measurement* — choose the measurement basis to align with the gradient direction rather than the standard computational basis.

3. Shot budget is coupled to the optimizer's uncertainty

When the optimizer is unsure (near a saddle point, in a flat region), more shots are needed per decision. When it's confident (steep gradient, clear direction), fewer shots suffice.

Technique. *Bayesian shot allocation* — compute the posterior over the cost value given current shots, continue measuring until the posterior is concentrated enough for the optimizer's needs.

4. Measurement noise is coupled to the optimizer's tolerance

Different optimizers tolerate different noise levels. Gradient descent is variance-sensitive; HOPSO's population averaging tolerates more noise per particle. The shot budget should match the optimizer's noise tolerance.

5. Mitigation is coupled to the cost landscape structure

If the landscape is known to be approximately linear near the current point, simple mitigation is enough. If it's steeply curved, more aggressive mitigation is worth the cost.

Concrete Techniques

Several research-stage techniques implement pieces of the co-design paradigm.

Shot-adaptive optimization (SAO)

Idea. At each iteration, use just enough shots to make a confident decision, then move on. Don't waste shots after the decision is certain.

Algorithm. 1. Start with a small shot count. 2. Measure the cost and gradient. 3. Compute the decision (parameter update direction). 4. Estimate the uncertainty in the decision. 5. If uncertainty is below threshold, commit. Otherwise, take more shots and repeat.

Reference. Kübler, Arrasmith, Cincio, Coles (2020) — iCANS (Individual Coupled Adaptive Number of Shots).

Rosalin — Rotoselect and related

Idea. For each parameter, select the optimal rotation axis *and* measurement axis jointly. Reduces the number of shots needed per gradient.

Reference. Ostaszewski et al. (2021) — Rotoselect.

Adaptive observable compilation

Idea. Combine multiple Pauli observables into a single measurement circuit by diagonalizing them in a shared basis. Reduces the number of distinct circuits needed.

Technique. Use Clifford compilation to rotate multiple Paulis into the computational basis simultaneously, then measure all qubits once. The key question is which Paulis can be measured together (they must commute).

Reference. Verteletskyi et al. (2020), Yen et al. (2020) — grouping strategies for VQE measurement.

Information-weighted shot allocation

Idea. Weight each shot by its contribution to gradient information, using Fisher information or related quantities.

Formalization. The Fisher information $\mathcal{I}(\boldsymbol{\theta})$ tells you how informative a measurement is about the parameters. Allocate shots to maximize $\mathrm{Tr}(\mathcal{I})$ subject to a budget constraint.

Adaptive mitigation per iteration

Idea. Apply ZNE only when needed — near convergence when bias matters. Skip ZNE early when raw gradient signal is strong.

The Full Co-Designed System

A fully co-designed VQA looks like this:

Information flow.

1. Optimizer states its current needs: "I need the gradient in direction $\boldsymbol{d}$ with precision ϵ".
2. Measurement system computes the optimal way to satisfy the request: shot budget, observable selection, mitigation.
3. Measurement system executes, returns the requested information.
4. Optimizer updates parameters.
5. Repeat.

What the optimizer exposes: - Current parameters $\boldsymbol{\theta}$. - Current direction of interest $\boldsymbol{d}$ (possibly all directions, if doing full gradient). - Required precision ϵ. - Available budget (time, shots).

What the measurement system exposes: - Available observables and their cost. - Mitigation options and their overhead. - Hardware state (fidelity, queue).

What they negotiate: the exact shot allocation, observable choice, and mitigation for this specific iteration, given the optimizer's needs and the measurement's constraints.

This is a cleanly specified interface, and it's dramatically more efficient than the standard decoupled architecture. The research question is whether the engineering complexity is worth the savings — which depends on the specific problem and hardware.

Why Co-Design Matters

A few reasons co-design is likely to become important:

1. Shot cost dominates runtime

On real hardware, cost evaluation (running the circuit many times) dominates the total wall-clock time of a VQA. Optimizing shot allocation gives direct runtime improvements — sometimes 5-10x faster convergence.

2. Hardware is expensive

Quantum cloud time is expensive (on the order of $1-$10 per second as of 2026). Every shot counts. Co-design squeezes more optimization per shot.

3. The theoretical savings are large

Information-theoretic lower bounds (Cramer-Rao) show that optimal shot allocation can improve gradient variance by 5-100x over naive allocation, depending on the Hamiltonian structure. Standard VQA is far from the bound.

4. It enables new architectures

Once measurement and optimization are co-designed, entirely new architectures become possible: continuous-shot optimizers, shot-streaming training, online measurement adaptation, etc. These don't work in the decoupled paradigm.

5. It aligns with the control-theoretic view

In control theory, the sensor and the controller are not always decoupled. Many optimal controllers integrate sensor design with control design (observability, sensor placement). Co-design is the same principle applied to VQAs.

The Thesis's Claim On Co-Design

The thesis's specific claim:

HOPSO is structurally well-suited to co-design because its explicit state and decomposed dynamics make it natural to expose "what the optimizer currently needs" to the measurement side. A co-designed HOPSO is a straightforward extension: the meta-controller tracks population state; the estimator tracks shot needs; the interface between them carries the information needed to optimize the measurement.

In contrast, Adam or plain gradient descent has an opaque state (just the current parameters and a moving average), making co-design more awkward. The optimizer doesn't know what it needs because the optimizer has no explicit state to articulate needs from.

The prediction: co-design will become standard by the end of the NISQ era, and the optimizer side of co-design will look more like HOPSO (structured, exposed state) than like Adam (minimal state).

Where Co-Design Meets Fault-Tolerance

A final note on the longer view: **co-design is especially valuable in the transition to early fault-tolerance**.

In the early-FTQC regime, you have a small number of logical qubits (very expensive) and many physical qubits (cheaper). Co-design is the right framework for deciding: - Which observables get measured on logical qubits (for precision). - Which get measured on physical qubits with mitigation. - How to allocate the logical qubit budget across the VQA iterations.

In other words, co-design is not just for shots — it's for *any* resource that the measurement and optimizer share. When the resources are more heterogeneous (logical vs physical, fast vs slow, precise vs coarse), the co-design paradigm becomes more valuable.

This is the thesis's ultimate forward-looking claim: **the right architecture for VQAs in the early-FTQC era is a co-designed measurement-optimizer system, built on the closed-loop control framework, with HOPSO-like structured optimizers and noise-adaptive objective generators.** Module 10 has been building up to this claim; this node states it explicitly.

Structural Role

This node is the **closing synthesis** of Module 10 and of the VQA course. It states the thesis's most forward-looking architectural claim: measurement and optimization should be co-designed, not decoupled.

It is the node that points the way to what comes *after* the course — the research frontier where the thesis's framework can be applied to problems that haven't been solved yet.

Relations to Other Concepts

- **Kübler et al. (2020)** — iCANS, shot-adaptive optimization.
- **Ostaszewski et al. (2021)** — Rotoselect, co-designed rotation and measurement.
- **Verteletskyi et al. (2020), Yen et al. (2020)** — Pauli grouping for measurement cost reduction.
- **Module 10.1 (Adaptive architectures)** — the parent framework.
- **Module 10.5 (Closing the loop)** — co-design at the interface of Levels 2 and 3.
- **Module 10.7 (HOPSO as controller)** — the optimizer side of co-design.
- **Information theory (Fisher information, Cramer-Rao bound)** — the theoretical foundation.

Worked Example — Co-Designed VQE For A 20-Qubit Chemistry Problem

Setup. 20-qubit chemistry VQE, Hamiltonian with 500 Pauli terms, coefficients spanning 5 orders of magnitude.

Standard VQA. Equal shots per term. Total shot budget: 10^6. Time per iteration: 30 seconds. Gradient variance: $\sigma^2 = 0.01$.

Co-designed VQA. - Shot allocation proportional to $|c_i|$. Terms with small coefficients get very few shots. - Observable grouping: reduce 500 Pauli terms to ~50 measurement groups (10x reduction). - Adaptive ZNE: applied only when gradient norm is small (final 30% of iterations). - HOPSO as optimizer with shot request interface.

Results: - Time per iteration: 8 seconds (4x speedup from fewer circuits + smarter allocation). - Gradient variance: 0.003 (3x improvement from better allocation). - Total iterations to convergence: similar (100-200). - **Total runtime reduction: ~10x.**

Observation. Co-design gives a large (~order of magnitude) runtime improvement over standard VQA at zero cost in final solution quality. This is the kind of result that makes co-design a research priority.

Visualization

Pending. A comparison diagram showing standard VQA (measurement and optimizer as separate boxes with a thin arrow between them) vs co-designed VQA (multiple information channels between the boxes, including shot budget, observable selection, mitigation settings, precision requirements). Caption: "Co-design: a richer interface between measurement and optimization."

Exercises

1. For a Hamiltonian with 10 Pauli terms with coefficients $c_i = 1, 2, 5, 10, 20, 50, 100, 200, 500, 1000$, compute the optimal shot allocation for estimating the cost with minimum variance. Compare to equal allocation.
2. For a VQA optimizer that requires gradient variance below ϵ, derive the relationship between ϵ and the total shot count needed under optimal vs. naive allocation.
3. (Capstone project) Implement a co-designed mini-VQA for a 4-qubit VQE: the optimizer requests gradients with specific precision; the measurement subsystem computes the optimal shot allocation. Compare runtime to a standard VQA.

Research Extensions

- **Information-theoretic bounds on co-designed VQAs** — how much speedup is achievable in theory?
- **Online Pauli grouping** — re-compute measurement groups during training as the state evolves.
- **Co-designed mitigation** — mitigation techniques that are aware of the optimizer's state.
- **Co-designed adaptive ansätze** — ansatz growth that uses measurement-side information.
- **Co-designed error correction + VQA** — in the early FTQC regime, deciding which operations to logical-qubit and which to mitigate.

Cross-links to Other Courses

- **Information Theory** — Fisher information, Cramer-Rao bounds.
- **Statistical Estimation Theory** — optimal allocation, sufficient statistics.
- **Module 10.5 (Closing the loop)** — the feedback architecture.
- **Module 10.7 (HOPSO as controller)** — the optimizer side.
- **Quantum Metrology** — where measurement-estimator co-design is already standard.
- **All of Modules 1-10** — this node is the synthesis of the whole course into a single forward-looking claim.

Concepts: expectation-value, adaptive-systems, feedback, variance-and-covariance
Prerequisites: 10.1, 10.2, 10.5, 10.7