Module 5 — Regularization

Variational Quantum Algorithms

hbar.university

Regularization as Prior Injection

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.1

This is the **standard machine-learning lens** on what regularization is. It is the lens you will encounter in any textbook on supervised learning, statistics, or inverse problems. We start here because Module 5 must first explain regularization in its conventional form before we can argue (in node 5.5) that the conventional form is incomplete in the VQA setting.

Formal Definition

Given an optimization problem

$$\min_{\boldsymbol{\theta}} \; C(\boldsymbol{\theta})$$

a **regularizer** is an auxiliary penalty function $R(\boldsymbol{\theta})$ added to the objective:

$$\min_{\boldsymbol{\theta}} \; C(\boldsymbol{\theta}) + \lambda \, R(\boldsymbol{\theta}),$$

where $\lambda \geq 0$ is the **regularization strength**.

The Bayesian interpretation is that this is a **maximum a posteriori (MAP)** estimate under a prior $p(\boldsymbol{\theta}) \propto \exp(-\lambda R(\boldsymbol{\theta}))$:

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} \; p(\text{data} \mid \boldsymbol{\theta}) \, p(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta}} \; \big[-\log p(\text{data} \mid \boldsymbol{\theta}) - \log p(\boldsymbol{\theta}) \big].$$

The negative log-likelihood is the cost C. The negative log-prior is λR.

Intuition

A regularizer is a way of saying:

> Among all parameter values that fit the data equally well, prefer the ones that are "small," "smooth," "sparse," or otherwise structured.

The prior is your belief about what counts as a "reasonable" $\boldsymbol{\theta}$ before you see any data.

The data tugs the optimizer in one direction. The prior tugs it in another. The MAP estimate is the equilibrium.

In the limit $\lambda \to 0$, the prior is irrelevant and you recover the maximum-likelihood estimate. In the limit $\lambda \to \infty$, the data is irrelevant and you recover the prior mode. The art is choosing λ so that the two are in productive tension.

Structural Role

This node is the **anchor** for the standard regularization techniques in nodes 5.2 (geometric), 5.3 (spectral), and 5.4 (generalization bounds). All three are concrete instances of the prior-injection framework.

It is also the **foil** for node 5.5 (regularization as dynamical modification). The thesis claim is that prior-injection is the wrong framing for VQA, because in VQA the optimizer never reaches the MAP solution. It lives in a stochastic transient regime where dynamics, not steady states, determine outcomes. You cannot understand 5.5 without first having 5.1 in hand.

Relations to Other Concepts

- **L2 / weight decay** — Gaussian prior on parameters. $R(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2$.
- **L1 / Lasso** — Laplace prior. $R(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_1$. Promotes sparsity.
- **Tikhonov regularization** — the inverse-problems name for L2.
- **Ridge regression** — linear regression with L2.
- **Dropout** — implicit regularization via stochastic pruning of computation paths.

Worked Derivations

L2 regularization is a Gaussian prior

Let $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{0}, \sigma^2 I)$. Then

$$-\log p(\boldsymbol{\theta}) = \frac{1}{2\sigma^2}\|\boldsymbol{\theta}\|_2^2 + \text{const.}$$

Setting $\lambda = 1/(2\sigma^2)$ gives the standard L2 penalty. Smaller prior variance is equivalent to stronger regularization.

MAP versus full posterior

The MAP estimate is a single point, not a distribution. It throws away all information about the posterior's spread. Bayesian methods that integrate over the full posterior (variational inference, MCMC) recover this information at higher computational cost. For VQA, even MAP is hard — see node 5.5 for why we don't actually reach MAP in practice.

Visualization

Pending. Diagram: 2D parameter space with the data-likelihood contour ellipse, the prior contour circle (centered at origin), and the MAP point at the intersection of the two pulls.

Exercises

1. Show that L2 regularization with strength λ on a linear regression problem shrinks each least-squares coefficient by a factor of $\sigma_i^2/(\sigma_i^2 + \lambda)$, where σ_i is the i-th singular value of the design matrix.
2. Derive the MAP estimate for a Bernoulli likelihood with a Beta prior. What does the regularization strength correspond to in terms of "effective prior counts"?
3. (VQA) Consider a parameterized circuit with parameters $\boldsymbol{\theta} \in [-\pi, \pi]^d$. Why does an L2 penalty have a less natural interpretation here than in unconstrained Euclidean parameter spaces?

Research Extensions

- **Empirical Bayes** — learning the prior hyperparameters from the data itself.
- **Hierarchical priors** — putting priors on the regularization strength λ.
- **Implicit regularization** — gradient descent itself acts as a regularizer in overparameterized models, even without an explicit penalty term. This is the bridge to node 5.5.

Cross-links to Other Courses

- **Statistics Module 6** — Bayesian Inference (priors, posteriors, MAP).
- **Statistics Module 10** — Model Selection (penalized likelihood, AIC/BIC, cross-validation as implicit regularization).
- **Math-Intuition** — Lagrangian formulation of constrained optimization as regularization.

Geometric Regularization

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.2

This node specializes the prior-injection framework of node 5.1 to the case where the prior is **geometric** — that is, defined on the manifold of reachable quantum states rather than on the raw parameter vector.

Formal Definition

Let $\mathcal{M} = \{|\psi(\boldsymbol{\theta})\rangle : \boldsymbol{\theta} \in \mathbb{R}^d\}$ be the ansatz manifold (see node 2.9).

A **geometric regularizer** is a penalty function

$$R_{\text{geom}}(\boldsymbol{\theta}) = d_{\mathcal{M}}\big(|\psi(\boldsymbol{\theta})\rangle, |\psi_0\rangle\big)^2$$

where $d_{\mathcal{M}}$ is a distance function on $\mathcal{M}$ and $|\psi_0\rangle$ is a reference state (often the previous iterate).

The most common choice is the **Fubini-Study metric**:

$$d_{\text{FS}}(|\psi\rangle, |\phi\rangle) = \arccos|\langle\psi|\phi\rangle|.$$

The regularized cost becomes

$$C_{\text{reg}}(\boldsymbol{\theta}) = C(\boldsymbol{\theta}) + \lambda\, d_{\text{FS}}(|\psi(\boldsymbol{\theta})\rangle, |\psi(\boldsymbol{\theta}_k)\rangle)^2,$$

evaluated at iteration k.

Intuition

L2 regularization on $\boldsymbol{\theta}$ penalizes large parameter values. But two parameter vectors that differ a lot in $\boldsymbol{\theta}$-space may produce nearly identical quantum states — and conversely. **The geometry of the parameter manifold and the geometry of the state manifold do not agree.**

Geometric regularization fixes this by penalizing motion in *state space*, which is the physically meaningful place.

This is the same idea that motivates **quantum natural gradient** (node 6.2) — both replace the Euclidean parameter metric with the Fubini-Study metric on the state manifold.

Structural Role

Geometric regularization is the **manifold-aware** member of the standard regularization family. It connects Module 5 to Module 2 (the ansatz manifold) and Module 6 (natural gradient).

It is also a stepping stone toward node 5.5: once you accept that the parameter metric is the wrong unit of "distance," you are halfway to seeing regularization as something the optimizer dynamics need to consume — not a property of an idealized objective.

Relations to Other Concepts

- **Quantum Fisher Information** (node 6.2) — the metric tensor whose square root is the local Fubini-Study distance.
- **Trust regions** (node 5.6) — bounds the step size in some metric. When the metric is FS, this is geometric regularization in disguise.
- **Mirror descent** — generalized gradient methods that respect a non-Euclidean parameter geometry.

Worked Derivations

Fubini-Study to leading order

For an infinitesimal step $\delta\boldsymbol{\theta}$ from $\boldsymbol{\theta}_k$, the Fubini-Study distance squared is

$$d_{\mathrm{FS}}^2 = \delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta}_k)\,\delta\boldsymbol{\theta} + O(\|\delta\boldsymbol{\theta}\|^4),$$

where F is the quantum Fisher information matrix. At first order in $\delta\boldsymbol{\theta}$, geometric regularization is equivalent to a **state-space Tikhonov penalty** with the QFI matrix as the metric.

Equivalence to natural gradient damping

A geometrically regularized step minimizes

$$C(\boldsymbol{\theta}_k + \delta\boldsymbol{\theta}) + \lambda\,\delta\boldsymbol{\theta}^\top F(\boldsymbol{\theta}_k)\,\delta\boldsymbol{\theta}.$$

Setting the gradient to zero gives

$$\delta\boldsymbol{\theta} = -(2\lambda F + H)^{-1}\nabla C,$$

where H is the Hessian of C. For small H, this approaches the **damped natural gradient step** $-(2\lambda F)^{-1}\nabla C$.

Visualization

Pending. Diagram: parameter space (Euclidean grid) on the left, state manifold (curved surface) on the right, with the same parameter step producing different state distances depending on the local curvature.

Exercises

1. For a single-qubit ansatz $|\psi(\theta)\rangle = R_y(\theta)|0\rangle$, compute the Fubini-Study distance between $\theta = 0$ and $\theta = \pi$. Compare to the Euclidean distance $|\pi - 0| = \pi$.
2. Show that geometric regularization with the FS metric is invariant under reparameterization of the ansatz, while L2 is not.
3. Implement geometric regularization for a 2-qubit hardware-efficient ansatz and compare convergence to plain L2 on a small VQE instance.

Research Extensions

- **Information-geometric optimization** — full theory in Amari's framework.
- **Riemannian VQE** — treating the optimization as a Riemannian descent on the state manifold from the start.
- **Geodesic regularization** — penalizing total geodesic length traversed, not just final distance.

Cross-links to Other Courses

- **Statistics Module 5** — Information Geometry (Fisher information as a Riemannian metric).
- **Linear Algebra** — Riemannian metrics on submanifolds.

Spectral Regularization of Parameterized Circuits

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.3

The third standard regularization technique. Penalizes large spectral norms of intermediate operators in the parameterized circuit, controlling Lipschitz behavior of the cost as a function of $\boldsymbol{\theta}$.

Formal Definition

Let $U(\boldsymbol{\theta}) = U_L(\theta_L)\cdots U_2(\theta_2)U_1(\theta_1)$ be a parameterized unitary built from L layers.

The **spectral norm** of an operator A is

$$\|A\|_{\text{op}} = \sup_{|\psi\rangle \neq 0} \frac{\|A|\psi\rangle\|}{\||\psi\rangle\|} = \sigma_{\max}(A).$$

For a unitary, $\|U\|_{\text{op}} = 1$ trivially, so the interesting object is the spectral norm of **derivatives**:

$$R_{\text{spec}}(\boldsymbol{\theta}) = \sum_{k=1}^{L} \left\| \frac{\partial U}{\partial \theta_k} \right\|_{\text{op}}^2 .$$

Penalizing this quantity bounds how much a small change in any single parameter can perturb the output state.

Intuition

A circuit with high spectral sensitivity is one where small parameter wiggles cause large state changes. That sounds good for expressivity — but it also means large gradient *variance* under shot noise, and large *condition number* of the optimization problem. Spectral regularization is the trade: pay a small cost in expressivity to gain stability.

It is the **Lipschitz constant lens** on regularization: bound the worst-case rate of change.

Structural Role

Spectral regularization is the most "operator-theoretic" of the standard regularizers. It is the lens that talks about *what the circuit does*, not what its parameters look like. This makes it the natural bridge between regularization theory (Module 5) and expressivity theory (Module 8).

Relations to Other Concepts

- **Lipschitz constant** — $|C(\boldsymbol{\theta}) - C(\boldsymbol{\theta}')| \leq L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|$. Spectral regularization bounds L.
- **Trainability bounds** (node 8.4) — gradient variance scales with spectral quantities.
- **Dynamical Lie algebra** — the algebra generated by the circuit's generators determines its spectral richness.

Worked Derivations

Spectral norm of a Pauli rotation derivative

For $U_k(\theta_k) = e^{-i\theta_k P_k/2}$ with P_k a Pauli string,

$$\frac{\partial U_k}{\partial \theta_k} = -\frac{i}{2} P_k \, U_k.$$

Since P_k and U_k are both unitary,

$$\left\| \frac{\partial U_k}{\partial \theta_k} \right\|_{\text{op}} = \frac{1}{2}.$$

So for a depth-L circuit of Pauli rotations, $R_{\text{spec}} = L/4$, independent of $\boldsymbol{\theta}$. This is why spectral regularization is more interesting for non-Pauli generators or for compound operators.

Lipschitz bound on the cost

If $R_{\text{spec}}(\boldsymbol{\theta}) \leq B$, then for any Hermitian H with $\|H\|_{\text{op}} \leq M$,

$$|C(\boldsymbol{\theta}) - C(\boldsymbol{\theta}')| \leq 2M\sqrt{B}\, \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|.$$

This gives an *a priori* bound on the Lipschitz constant of C, which feeds directly into the convergence theory of node 6.5.

Visualization

Pending. Diagram: same circuit drawn twice — once with low spectral norm derivatives (smooth response), once with high (jagged response) — and the corresponding cost-vs-parameter curves.

Exercises

1. Show that for a single layer $U(\theta) = e^{-i\theta H}$ with H Hermitian, the spectral regularizer reduces to $\|H\|_{\text{op}}^2$.
2. Compute R_{spec} for a 2-qubit hardware-efficient ansatz with depth 3.
3. Argue that spectral regularization can never *increase* the location of the optimum, only restrict the trajectory toward it.

Research Extensions

- **Lipschitz quantum machine learning** — formal generalization theory for VQA based on Lipschitz constants.
- **Spectral pruning** — using the spectrum of the Hessian (rather than derivatives of U) as the regularization signal.
- **Operator-norm penalties on the cost itself** — closely related to the cost-shaping techniques in node 5.7.

Cross-links to Other Courses

- **Linear Algebra** — operator norms, singular value decomposition.
- **Math-Intuition** — Lipschitz functions and continuous mappings.

Regularization and Generalization Bounds

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.4

The fourth standard regularization technique. Asks: *given* the optimizer succeeds in reducing the training cost, what does the regularizer guarantee about the cost on **unseen** problem instances?

For VQE specifically, "unseen instances" means: different shots, different noise realizations, or different molecular geometries.

Formal Definition

Let $\mathcal{D}$ be a distribution over problem instances and let $\hat{C}(\boldsymbol{\theta})$ be the empirical cost on a finite training set. A **generalization bound** is an inequality of the form

$$\mathbb{E}_{x\sim\mathcal{D}}[C(\boldsymbol{\theta}; x)] \leq \hat{C}(\boldsymbol{\theta}) + \mathcal{B}(\mathcal{H}, n, \delta)$$

with probability at least $1-\delta$ over training set draws, where $\mathcal{B}$ is a complexity term that depends on the hypothesis class $\mathcal{H}$ (the ansatz family) and the training-set size n.

A **regularizer reduces** $\mathcal{B}$ by restricting the effective complexity of $\mathcal{H}$.

Intuition

Generalization is the answer to: "My circuit got the right answer on the data I trained it on. Will it get the right answer on new data?"

Without regularization, an overparameterized ansatz can perfectly fit any training set — including the noise — and fail catastrophically on new instances. A regularizer says: "Don't fit the noise. Restrict yourself to a smaller class of functions, where overfitting is impossible."

The bound $\mathcal{B}$ is the price you pay in worst-case error for the freedom to choose any $\boldsymbol{\theta}$ in the unrestricted class. Regularization shrinks the class, which shrinks the price.

Structural Role

This is the most theoretical of the four standard regularization nodes. It is the lens that lets you state, with formal probability guarantees, *why* L2 / spectral / geometric regularization should work.

But it is also the lens that **fails most cleanly in the VQA setting**, which is why it sets up the polemic in node 5.5. The standard generalization story assumes (a) you have a well-defined data distribution, (b) you reach a fixed steady-state solution, (c) your optimizer is essentially noise-free. None of these hold for VQA. See 5.5 for the full critique.

Relations to Other Concepts

- **Rademacher complexity** — the complexity term $\mathcal{B}$ for many concrete hypothesis classes.
- **PAC-Bayes** — generalization bounds that incorporate a Bayesian prior over $\boldsymbol{\theta}$, making the connection to node 5.1 explicit.

- **Stability-based bounds** — generalization through algorithmic stability rather than hypothesis-class size.
- **VC dimension** — the combinatorial measure of capacity for binary classifiers.

Worked Derivations

A schematic Rademacher bound

For a hypothesis class $\mathcal{H}$ with Rademacher complexity $\mathfrak{R}_n(\mathcal{H})$ on samples of size n, the standard bound gives

$$\mathbb{E}[C] \leq \hat{C} + 2\,\mathfrak{R}_n(\mathcal{H}) + 3\sqrt{\frac{\log(2/\delta)}{2n}}$$

with probability $1 - \delta$.

Spectral regularization (node 5.3) bounds $\|U(\boldsymbol{\theta})\|_{\text{Lip}}$, which in turn bounds $\mathfrak{R}_n(\mathcal{H})$, tightening the inequality.

What goes wrong for VQA

The bound assumes $\hat{C}$ is the *minimum* of the empirical objective. In VQA, $\hat{C}$ is whatever the optimizer happened to converge to — typically a stochastic transient, not a minimum. The bound still holds in principle, but it ceases to be informative because the optimizer's actual behavior is governed by *dynamics*, not by the geometry of the loss surface. This is the gap that node 5.5 fills.

Visualization

Pending. Diagram: the bound as an envelope around the empirical-cost curve, shrinking as a regularizer is applied.

Exercises

1. Compute the Rademacher complexity of a linear classifier with bounded L2 weight norm.
2. Sketch why a depth-L hardware-efficient ansatz has Rademacher complexity at least exponential in L without regularization.
3. (Conceptual) Argue why "training cost" and "test cost" are not the right concepts in VQE — and what should replace them.

Research Extensions

- **Quantum PAC-Bayes** — recent work extending PAC-Bayes bounds to variational quantum models.
- **Implicit-bias generalization** — explaining generalization without explicit regularization, by analyzing the bias of the optimizer itself. This is a direct neighbor of node 5.5.
- **Distribution-free bounds** — generalization that does not depend on a fixed training distribution.

Cross-links to Other Courses

- **Statistics Module 10** — Model Selection (cross-validation, AIC/BIC, structural risk minimization).
- **Statistics Module 7** — Asymptotic Theory (uniform convergence, empirical processes).

Regularization as Dynamical Modification (and Why It Is Not Bias)

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.5

This is the **central claim** of the regularization thesis. It is the node that motivates why Module 5 is labelled "an original contribution" rather than "standard background." Everything in nodes 5.1 through 5.4 was standard ML / inverse-problems theory. This node argues that the standard theory **misframes what regularization is doing in the VQA setting**, and proposes a replacement framing.

The thesis claim

> In variational quantum algorithms, regularization should be understood not as injecting a prior into the objective, but as **modifying the dynamics of the optimizer** under stochastic feedback. The objective remains unchanged. Only the trajectory through parameter space is altered.

Source: thesis Chapter 6.2 [@thesis-ch6].

Formal Definition

Let the unregularized parameter update rule be the discrete-time dynamical system

$$\boldsymbol{\theta}_{k+1} = \mathcal{A}\big(\boldsymbol{\theta}_k,\, \hat{C}(\boldsymbol{\theta}_k),\, \mathcal{H}_k\big),$$

where $\mathcal{A}$ is the classical optimizer, $\hat{C}$ is the noisy cost estimate, and $\mathcal{H}_k$ is the optimizer's internal state at iteration k.

A **regularizer** is a modification of this map:

$$\boldsymbol{\theta}_{k+1} = \mathcal{A}_{\text{reg}}\big(\boldsymbol{\theta}_k,\, \hat{C}(\boldsymbol{\theta}_k),\, \mathcal{H}_k;\, \lambda\big),$$

where λ controls the strength.

Crucially, $\mathcal{A}_{\text{reg}}$ is not derived from a modified cost. It is a direct modification of the update rule itself — adding a damping term, smoothing the input signal, bounding the step size, etc. The fixed points of $\mathcal{A}_{\text{reg}}$ are *not* generally those of $\arg\min C + \lambda R$.

Intuition

In a noise-free convex setting, gradient descent on $C(\boldsymbol{\theta}) + \lambda R(\boldsymbol{\theta})$ and gradient descent on C with a damping term added to the update rule are mathematically the same thing in the limit $k \to \infty$. **They converge to the same point.** This is why the standard ML literature can use prior-injection and dynamical-modification interchangeably — at convergence, they agree.

In VQA, you never reach convergence. Shot noise, NISQ-era circuit depth limits, and finite measurement budgets mean the optimizer lives in a permanent transient regime. **The two framings disagree about what is happening in the transient.**

The prior-injection framing says: "we are searching for the regularized minimum." The dynamical-modification framing says: "we are flowing along a stochastic trajectory whose distribution at iteration k is what we care about."

The latter is the right description for VQA. The former is a useful classical limit, but it does not predict what an actual VQA run looks like at iteration 50 with 1000 shots per evaluation.

Why it is not bias

A common objection: "if you modify the dynamics, surely you are introducing bias?"

No. Or rather: not in the sense the objector means.

A *biased estimator* is one whose expected steady-state value is not the true minimum. A *dynamical modification* is one whose **trajectory** is different but whose **steady-state** (in the limit $k \to \infty$ and $\lambda \to 0$) is the same as the unmodified system.

Schematically:

$$\lim_{k\to\infty} \lim_{\lambda\to 0} \mathbb{E}[\boldsymbol{\theta}_k^{(\lambda)}] = \boldsymbol{\theta}^\star$$

where $\boldsymbol{\theta}^\star$ is the true minimizer of C.

The order of limits matters. You have to take $\lambda \to 0$ **before** $k \to \infty$ for the unbiased property to hold. This is the regime in which **the cosine schedule (node 5.8) operates**: λ is annealed to zero as k grows, ensuring that the regularizer fades out before the steady state is reached.

This is the polemic point. A static regularizer (constant λ) does introduce bias — it shifts the steady state. A scheduled regularizer that vanishes at convergence does not introduce bias — it only shapes the path. The thesis argument is that **only the scheduled, dynamical interpretation makes sense in VQA**, because static regularization in a transient regime is just adding bias for no asymptotic benefit.

Structural Role

This is the load-bearing node of Module 5. Nodes 5.6, 5.7, 5.8 are all instances of dynamical modification. Node 5.9 is the empirical evidence that the framing pays off.

It is also the link between Module 5 and Module 6 (Optimization Dynamics). The dynamical-modification framing only makes sense if you have already accepted that the optimizer is a dynamical system and not a function evaluator — see node 6.1 for the foundation.

Relations to Other Concepts

- **Stochastic approximation theory** (Robbins-Monro) — the closest classical analogue. Annealed step sizes are conceptually identical to scheduled regularization.
- **Implicit bias of SGD** — the observation that gradient descent itself produces solutions that are "regularized" without any explicit penalty, depending on the optimizer.
- **Dynamical systems / control theory** — the natural language for describing the modified update map.
- **Effective potential** — viewing $\mathcal{A}_{\text{reg}}$ as gradient flow on a modified energy landscape that depends on k and λ.

Worked Derivations

Damped gradient descent as dynamical modification

Consider gradient descent with a damping term:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta \nabla C(\boldsymbol{\theta}_k) - \eta \lambda_k \boldsymbol{\theta}_k.$$

If λ_k is constant, this is L2-regularized gradient descent and converges to $\arg\min C + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2$ — a biased steady state.

If $\lambda_k \to 0$ such that $\sum_k \lambda_k = \infty$ but $\sum_k \lambda_k \eta_k < \infty$, the steady state is the unbiased $\arg\min C$. The damping shapes the path but vanishes asymptotically.

Why constant λ is bad in VQA

For constant λ, the steady state of the regularized dynamics is

$$\boldsymbol{\theta}^\star_\lambda \neq \arg\min C.$$

The displacement $\|\boldsymbol{\theta}^\star_\lambda - \arg\min C\|$ is bounded by something proportional to λ/μ, where μ is the local strong-convexity constant. In VQA, μ is small (weakly identified parameters) and λ is non-trivial (you need it for stability), so the displacement is *not negligible*. You are paying a permanent bias for a temporary stability gain. **This is the core argument for scheduled regularization.**

Visualization

Pending. Diagram: two trajectory plots in 2D parameter space — one with constant λ (ends at biased point), one with scheduled λ (ends at the true minimum, but visits a more stable transient region).

Exercises

1. Construct a 1D quadratic example where constant L2 regularization shifts the minimum by an amount you can compute exactly.
2. Show that for the same example, a cosine-scheduled L2 with $\lambda_k = \lambda_0 \cos^2(\pi k/2K)$ reaches the true minimum at iteration K.
3. Argue (without proof) why the dynamical-modification framing is unnecessary in noise-free convex optimization, and necessary in noisy non-convex settings.

Research Extensions

- **Stochastic differential equation models of SGD** — continuous-time limit of the discrete dynamics; turns regularization into a drift term.
- **Mean-field theory of optimizer dynamics** — averaging over noise realizations to predict trajectory distributions.
- **Connection to quantum annealing schedules** — the cosine schedule in 5.8 is closely related to adiabatic quantum schedules.

Cross-links to Other Courses

- **Statistics Module 9** — Stochastic Optimization (Robbins-Monro, annealed step sizes).
- **Math-Intuition** — Dynamical systems, fixed points, and basins of attraction.
- **Quantum Foundations** — Adiabatic theorem (analogous schedule structure).

Parameter-Space Regularization Techniques

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.6

This node catalogs the **parameter-space** family of regularization techniques: methods that act directly on the evolution of $\boldsymbol{\theta}$ by adding restoring forces, bounds, or step-size limits. It is one of the two operational halves of the dynamical-modification view introduced in node 5.5. Its sibling is node 5.7, which catalogs **objective-space** techniques.

The Common Mechanism

Every parameter-space technique modifies the update rule

$$\boldsymbol{\theta}_{k+1} = \mathcal{A}\big(\boldsymbol{\theta}_k, C(\boldsymbol{\theta}_k), \mathcal{H}_k\big)$$

into

$$\boldsymbol{\theta}_{k+1} = \mathcal{A}_{\text{reg}}\big(\boldsymbol{\theta}_k, C(\boldsymbol{\theta}_k), \mathcal{H}_k; \lambda\big),$$

without changing C. The intervention happens on the **dynamics side**, not the **objective side**. This is the key contrast with node 5.7.

Technique 1 — Quadratic Penalty (L2 / Weight Decay)

Adds a penalty proportional to $\|\boldsymbol{\theta}\|_2^2$ inside the optimizer's update. For gradient methods this becomes a restoring force toward the origin:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta\big[\nabla C(\boldsymbol{\theta}_k) + \lambda\boldsymbol{\theta}_k\big].$$

In a periodic parameter space — which is what VQA actually has, since rotation angles live on a torus — the quadratic penalty has no natural origin and is conceptually awkward. This is why parameter-space regularization in VQA is usually applied as a **schedule** that vanishes near convergence (see node 5.8) rather than as a fixed prior.

Technique 2 — Trust Regions

Rather than penalizing parameter values, **bound the step**:

$$\|\boldsymbol{\theta}_{k+1} - \boldsymbol{\theta}_k\| \leq \Delta_k.$$

Trust regions are particularly natural for VQA because objective evaluations far from the current iterate are unreliable — the stochastic estimate of C is only locally meaningful. A trust region encodes the statement: *do not trust the optimizer's predicted descent direction beyond the radius where the cost estimate is statistically valid.*

Technique 3 — Parameter Clipping

Hard bounds on parameter values:

$$\theta_i \leftarrow \max(\theta_{\min}, \min(\theta_{\max}, \theta_i)).$$

Simple but introduces non-smooth behavior at the boundaries. In VQA, clipping is most often used as a safety net against runaway behavior, not as a primary regularizer. The natural domain $\theta_i \in [-\pi, \pi]$ for rotation angles makes clipping degenerate with the periodic structure of the manifold.

Technique 4 — Step-Size Control and Velocity Damping

Adaptive learning rates, momentum decay, and (in population-based methods) velocity caps all act as parameter-space regularizers. For HOPSO specifically, the harmonic damping term

$$\boldsymbol{v}_{k+1} = \alpha \boldsymbol{v}_k + \omega^2(\boldsymbol{\theta}_k - \boldsymbol{\theta}_k^\star)$$

is itself a parameter-space stabilizer baked into the dynamics. This is why HOPSO needs less *added* regularization than naive optimizers — the structure already does some of the work (see Module 6 for the full HOPSO treatment).

Structural Role

This node enumerates the **levers** that node 5.5 abstractly named "modifications to the dynamical map." Together with node 5.7 (objective-space techniques), it provides the operational vocabulary that node 5.8 (cosine schedule and two-stage protocol) and node 5.9 (empirical behavior) build on.

Without 5.6 and 5.7 in hand, the cosine schedule of 5.8 would look like an arbitrary trick. With them, it becomes a natural choice: a parameter-space penalty that anneals to zero, preserving the original optimum.

Relations to Other Concepts

- **Tikhonov regularization** — the classical-statistics name for L2.
- **Levenberg-Marquardt** — adaptive trust-region for nonlinear least squares.
- **Gradient clipping** — common in deep learning, conceptually similar to step-size control.
- **Projected gradient descent** — clipping interpreted as projection onto a feasible set.
- **HOPSO damping** — see Module 6; structural parameter-space regularization built into the optimizer.

Worked Example — L2 with annealing schedule

Consider $\lambda_k = \lambda_0 \cdot s(k)$ with $s(k) \to 0$. The update rule

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta\big[\nabla C(\boldsymbol{\theta}_k) + \lambda_k \boldsymbol{\theta}_k\big]$$

acts as a damped oscillator early (when λ_k is large) and reduces to vanilla gradient descent late (when $\lambda_k \to 0$). The asymptotic optimum is unchanged because the regularization vanishes; only the **trajectory** is altered. This is the dynamical-modification view in action — see node 5.5 for the formal limit ordering.

Visualization

Pending. Diagram: side-by-side trajectories on a 2D noisy quadratic. Left: vanilla SGD with high variance. Right: SGD + L2 with annealing — same final point, smoother path.

Exercises

1. Show that adding a constant L2 penalty $\lambda\|\boldsymbol{\theta}\|_2^2$ to the cost shifts the minimum away from the unregularized optimum, but a vanishing schedule $\lambda_k \to 0$ does not.
2. Derive the update rule for trust-region gradient descent on a quadratic. What happens when the trust-region radius is much smaller than the gradient magnitude?
3. (VQA) Suppose your parameters live on a torus $[-\pi, \pi]^d$. Design a parameter-space regularizer that respects the periodicity. (Hint: think about what "small" should mean on a circle.)

Research Extensions

- **Manifold-aware regularization** — penalizing motion in the geometry of the Bures or Fubini-Study metric (links back to node 5.2).
- **Adaptive trust regions for noisy oracles** — adjusting Δ_k based on the variance of the cost estimate.
- **Implicit regularization of momentum** — even without explicit penalty, momentum-based optimizers exhibit a regularizing effect on the trajectory.

Cross-links to Other Courses

- **Optimization Theory** — trust-region methods, projected gradient, Levenberg-Marquardt.
- **Math-Intuition** — Lagrangian formulation of constrained optimization.
- **Statistics Module 6** — Bayesian view of the (now-rejected) L2-as-Gaussian-prior mapping (see node 5.1 for the lens, 5.5 for why we reject it in VQA).

Objective-Space Regularization Techniques

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.7

This node catalogs the **objective-space** family of regularization techniques: methods that modify the cost signal $C(\boldsymbol{\theta})$ **before** it reaches the optimizer, without touching the parameter update rule. It is the sibling of node 5.6 and the second operational half of the dynamical-modification view from node 5.5.

The Common Mechanism

The optimizer sees not the raw cost $C(\boldsymbol{\theta}_k)$ but a transformed signal

$$\widetilde{C}_k = \mathcal{F}\big(C(\boldsymbol{\theta}_k), C(\boldsymbol{\theta}_{k-1}), \ldots; \lambda\big).$$

The transformation $\mathcal{F}$ might smooth, weight, or augment the cost signal. The optimizer's update rule is **unchanged** — it still consumes a number called "the cost" — but the number it consumes has been filtered.

This contrast with node 5.6 is the entire point: same optimizer, different feedback.

Technique 1 — Temporal Smoothing

Replace the instantaneous cost with a moving average or exponentially-weighted average:

$$\widetilde{C}_k = (1-\beta)\widetilde{C}_{k-1} + \beta\, C(\boldsymbol{\theta}_k).$$

This introduces inertia into the feedback loop. Transient shot-noise spikes are damped before they can cause large parameter excursions. The price is delayed responsiveness: a genuine improvement in C takes a few iterations to show up in $\widetilde{C}$.

In VQA this is one of the cheapest stabilizers available — no extra circuit evaluations, just a running average — and it directly attacks the dominant noise source on NISQ hardware.

Technique 2 — Variance Penalization

Penalize cost evaluations with high statistical uncertainty:

$$\widetilde{C}(\boldsymbol{\theta}) = C(\boldsymbol{\theta}) + \mu\, \mathrm{Var}\big[\hat{C}(\boldsymbol{\theta})\big].$$

This biases the optimizer toward regions where the cost can be **trusted**, not just where it is small. In VQA the variance of a Pauli expectation estimate depends on $\langle\psi(\boldsymbol{\theta})|H^2|\psi(\boldsymbol{\theta})\rangle - \langle\psi(\boldsymbol{\theta})|H|\psi(\boldsymbol{\theta})\rangle^2$, so this is a real, measurable quantity — not an abstract penalty.

As with all objective-space techniques in VQA, the penalty is typically scheduled to vanish near convergence so the location of the optimum is preserved.

Technique 3 — Shot-Adaptive Weighting

Cost evaluations carry different reliabilities depending on how many measurement shots were used to estimate them. Weight each evaluation by its precision:

$$\widetilde{C}_k = \frac{\sum_j w_j C_j}{\sum_j w_j}, \qquad w_j \propto \frac{1}{\sigma_j^2}.$$

This lets the optimizer use cheap, high-variance estimates exploratorily while still anchoring on rare high-shot evaluations near promising regions.

The resource-allocation question — *how to spend a finite shot budget across iterations* — is where this technique shades into the larger topic of measurement-optimizer co-design (see Module 10).

Technique 4 — Auxiliary Penalty on Cost-Change Magnitude

Penalize rapid swings in successive cost values:

$$\widetilde{C}_k = C(\boldsymbol{\theta}_k) + \nu \left(C(\boldsymbol{\theta}_k) - C(\boldsymbol{\theta}_{k-1})\right)^2.$$

This discourages oscillatory behavior driven by noise. Like temporal smoothing, it adds inertia, but here the inertia is on the **cost signal's volatility** rather than its level.

Structural Role

Objective-space regularization is the natural intervention point when **the optimizer is fine but the signal is noisy**. In VQA this is the typical failure mode: gradient methods, HOPSO, COBYLA — they all behave reasonably given a clean cost signal. The instability comes from shot noise corrupting the feedback.

This node, together with 5.6, gives the practitioner the full menu of dynamical-modification options. Node 5.8 then shows how to **schedule** these techniques over training so they vanish near convergence.

Relations to Other Concepts

- **Polyak averaging** — averaging iterates is a parameter-space analogue of cost averaging.
- **Kalman filtering** — temporal smoothing as a special case of state estimation under noise.
- **CMA-ES** — uses a form of variance-aware weighting in its sample selection.
- **Importance-weighted gradients** — the precision-weighting trick generalized.

Worked Example — Temporal smoothing as a low-pass filter

Treat the cost iterates $\{C(\boldsymbol{\theta}_k)\}$ as a discrete-time signal. The exponential moving average

$$\widetilde{C}_k = (1-\beta)\widetilde{C}_{k-1} + \beta C_k$$

has the transfer function

$$H(z) = \frac{\beta}{1-(1-\beta)z^{-1}},$$

a first-order low-pass filter with cutoff $\omega_c \approx \beta$ (in radians per iteration). Smaller β means heavier smoothing and slower response. The effective cost landscape seen by the optimizer is the original landscape convolved with this filter — high-frequency shot-noise components are suppressed; low-frequency descent directions pass through.

Visualization

Pending. Diagram: cost time series with high-frequency noise overlaid on a slow descent. Two traces: raw C_k (jagged) and smoothed $\widetilde{C}_k$ (clean). Annotate where the optimizer would have fired a destabilizing update on the raw signal.

Exercises

1. Show that exponential smoothing with parameter β is equivalent to a first-order IIR low-pass filter, and derive its 3 dB cutoff frequency.
2. Suppose the variance of your cost estimate scales as $1/N$ where N is the shot count. Design a shot-adaptive weighting scheme that minimizes the variance of $\widetilde{C}_k$ given a total shot budget $N_{\text{tot}} = \sum_j N_j$. (Hint: Lagrange multipliers.)
3. (VQA) Why does temporal smoothing interact badly with population-based optimizers like vanilla PSO? (Hint: think about how attractor positions get updated.)

Research Extensions

- **Adaptive smoothing** — adjusting β based on observed cost variance.
- **Bayesian optimization with noisy oracles** — explicit uncertainty modeling as objective-space regularization.
- **Joint shot-and-step adaptation** — co-designing the optimizer step size and the shot budget per iteration.

Cross-links to Other Courses

- **Signal Processing** — IIR filters, low-pass design, group delay.
- **Statistics Module 6** — Kalman filters and state estimation.
- **Optimization Dynamics (Module 6)** — the input side of the gradient-flow analysis.

The Cosine Schedule and Two-Stage Protocol

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.8

This node introduces the **two-stage protocol** that operationalizes the dynamical-modification view (node 5.5) using a cosine-annealed regularization schedule. It is the most concrete recommendation in Module 5: a specific recipe with specific hyperparameters that has been used in our own experiments. It depends on 5.5 (the framing), 5.6 (parameter-space techniques), and 5.7 (objective-space techniques).

The Idea in One Sentence

> Apply regularization aggressively early to stabilize trajectories, then anneal it to zero so the final iterates converge to the original (unregularized) optimum.

This is the obvious move once you accept the dynamical-modification view of node 5.5: regularization is for the path, not for the destination. If you keep regularization on at convergence you have changed the problem; if you anneal it away you have only changed the route.

The Schedule

Let T be the total iteration budget. Define a cosine schedule on the regularization weight:

$$\lambda_k = \frac{\lambda_0}{2}\left(1 + \cos\frac{\pi k}{T}\right), \qquad k = 0, 1, \ldots, T.$$

At $k = 0$, $\lambda_k = \lambda_0$. At $k = T$, $\lambda_k = 0$. The decay is smooth and monotone with a flat shoulder at the start (so early training experiences nearly the full regularization) and a flat tail at the end (so the optimizer has time to settle into the unregularized optimum).

Cosine schedules are popular in deep learning for the **learning rate**. Here we apply the same shape to the **regularization weight**, for analogous reasons: early phase wants exploration with stabilization, late phase wants fine-tuning without distortion.

The Two-Stage Protocol

The protocol splits the optimizer's run into two phases:

Stage A — Stabilized exploration. For $k \in [0, T_A]$, run the optimizer with the regularization schedule active. Use a relatively small per-iteration budget ($\texttt{maxiter}_A$ inner steps per outer iteration if the optimizer is hierarchical). The goal here is **not** to reach the minimum. The goal is to reach a stable region of parameter space that is robust to shot noise.

Stage B — Unregularized refinement. For $k \in [T_A, T]$, the schedule has effectively decayed; run the optimizer with regularization disabled (or near-zero). Use a longer per-iteration budget ($\texttt{maxiter}_B$). The goal here **is** to reach the minimum. Because Stage A delivered a good initialization, the unregularized phase no longer suffers the instabilities that motivated regularization in the first place.

A Concrete Configuration (Chernyak D30)

The configuration used in our own VQE experiments on a depth-30 problem instance:

Component	Setting
Ansatz	`TwoLocal(rotation_blocks=['rz','ry'], reps=4)`
Parameters	40
Optimizer	HOPSO (see Module 6)
Stage A iterations T_A	15
Stage A inner budget $\texttt{maxiter}_A$	15
Stage B inner budget $\texttt{maxiter}_B$	10
Schedule	cosine on regularization weight
λ_0	tuned per instance, typically small

These numbers are not magic. They were chosen empirically on a specific class of Hamiltonians and they reflect a tradeoff between *time spent stabilizing* and *time spent converging*. They are reported here as a worked example, not as a universal recommendation. Node 5.9 (empirical behavior) discusses how performance depends on these knobs.

Why Two Stages and Not One?

A natural alternative is to keep regularization on continuously, anneal slowly, and run a single-phase optimizer. The two-stage version is operationally cleaner for three reasons:

1. **Hyperparameter independence.** The Stage A budget can be tuned for stabilization, the Stage B budget for refinement. They optimize different quantities and you do not have to find a single setting that does both jobs at once.
2. **Logging clarity.** It is easier to read a training curve when the role of each iteration is explicit. "Why did the loss spike?" is answered differently in Stage A (stabilization phase, expected) and Stage B (refinement phase, suspicious).
3. **Optimizer compatibility.** Some optimizers (HOPSO included) have internal state — particle velocities, momentum buffers — that benefit from a clean reset between phases. The two-stage protocol gives you a natural moment to reset.

Structural Role

This node is the **operational climax** of Module 5. Everything in 5.1–5.7 leads here: - 5.1–5.4 establish the standard view and its limits. - 5.5 reframes regularization as dynamical modification. - 5.6 and 5.7 enumerate the levers. - 5.8 picks one specific lever (annealed parameter-space penalty) and shows how to schedule it.

Node 5.9 (empirical behavior under shot noise) then asks the question this node provokes: *does the protocol actually work?*

Relations to Other Concepts

- **Cosine annealing for learning rate** — same shape, different quantity. Loshchilov-Hutter, SGDR.
- **Curriculum learning** — staging the optimization problem from easier to harder.
- **Trust-region adaptation** — Stage B is essentially a relaxed trust region after Stage A has positioned the iterate.

- **Simulated annealing** — temperature schedule analogy; here λ plays the role of an inverse temperature.

Visualization

Pending. Diagram: a training curve with two color-coded segments. Left half: Stage A, jagged but bounded, with the cosine λ_k curve overlaid. Right half: Stage B, smooth descent to a lower asymptote. Annotate T_A, $\lambda_0 \to 0$, and the final unregularized minimum.

Exercises

1. Plot the cosine schedule $\lambda_k = \frac{\lambda_0}{2}(1 + \cos(\pi k/T))$ and verify that $\lambda_0 = \lambda_0$ and $\lambda_T = 0$. What is the iteration at which $\lambda_k = \lambda_0/2$?
2. Suppose Stage A is too short. Predict qualitatively what the training curve looks like and which failure mode dominates.
3. (VQA) Why is it inappropriate to evaluate the *final* energy of a Stage A run as if it were the answer? What metric should you report instead from Stage A?

Research Extensions

- **Adaptive T_A** — switching from Stage A to Stage B when a stability criterion is met instead of at a fixed iteration.
- **Schedule shapes beyond cosine** — linear, polynomial, step. Empirically, the differences are small once the total integral is matched.
- **Multi-stage protocols** — three or more phases for hierarchical optimization problems.

Cross-links to Other Courses

- **Optimization Dynamics (Module 6)** — interaction with HOPSO's intrinsic damping.
- **Hardware Noise Models (Module 9)** — Stage A's stabilization is most beneficial under high noise.
- **Math-Intuition** — the cosine schedule as a smooth interpolant between two regimes.

Empirical Behavior Under Shot Noise

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Regularization **Node:** 5.9

This is the **closing node** of Module 5. It asks the empirical question that nodes 5.5–5.8 set up: *under realistic shot noise, does the dynamical-modification view of regularization actually deliver more stable, more reliable variational optimization than the unregularized baseline?*

The answer is "yes, in specific regimes," and this node makes the regimes precise.

Why This Is the Right Question

Module 5's argument has been theoretical:

- Standard prior-injection (node 5.1) is the wrong frame for VQA because the optimizer never reaches MAP (node 5.5).
- Regularization should be understood as modifying the **trajectory**, not the **target** (5.5).
- This unlocks specific tools: parameter-space techniques (5.6), objective-space techniques (5.7), and the two-stage protocol with cosine annealing (5.8).

A theoretical argument is necessary but not sufficient. The cash value is whether running real VQE instances with the protocol gives better outcomes than running them without. "Better" here means: lower variance across seeds, more reliable convergence, comparable or better final energies, and improved measurement efficiency.

Experimental Setup

The setup mirrors the empirical chapters of the underlying thesis (see node 5.5 for the framing).

- **Problem class:** variational quantum eigensolver instances on fixed Hamiltonians.
- **Ansatz:** hardware-efficient `TwoLocal` circuits, varying depth and qubit count.
- **Noise model:** finite-shot sampling noise on every cost evaluation.
- **Optimizer:** HOPSO (see Module 6) with and without the two-stage protocol.
- **Replicates:** multiple independent random seeds per condition.

The independent variable is the regularization configuration (off / parameter-space only / objective-space only / two-stage protocol). The dependent variables are stability and efficiency metrics defined below.

Metrics

Convergence speed alone is the wrong yardstick under stochastic noise. The relevant metrics are:

1. **Best-energy variance across seeds.** How much does the final reported energy depend on initialization? Low variance means the protocol is reliable; high variance means the optimizer is finding different local minima or failing in seed-dependent ways.
2. **Worst-case run quality.** The 90th-percentile worst seed, not the median seed, is what determines whether you can trust a single production run.
3. **Iteration-vs-energy trajectory variance.** Even within a single seed, how jittery is the descent? A noisy trajectory is a sign of unstable feedback.

4. **Measurement-budget efficiency.** Energy as a function of total shots consumed, not iteration count. In NISQ, shots are the resource; iterations are an internal accounting unit.

Expected Results (Predictive)

Note: results pending the next vqe-lab sweep. The predictions below come from the dynamical analysis of nodes 5.5–5.8 and from preliminary observations on smaller instances.

Prediction 1. Under high shot noise, the two-stage protocol reduces best-energy variance across seeds compared to the unregularized baseline.

Prediction 2. Under low shot noise, the protocol confers little or no benefit and may slightly slow convergence due to the Stage A overhead.

Prediction 3. Parameter-space techniques (5.6) help most when instability comes from large parameter excursions; objective-space techniques (5.7) help most when instability comes from cost-signal volatility. The two-stage protocol blends both.

Prediction 4. The improvement in **worst-case** seeds is larger than the improvement in **median** seeds. This is the practically important effect: regularization tightens the distribution rather than shifting its center.

Prediction 5. Measurement-budget efficiency (metric 4) improves under the protocol whenever stability improves, because fewer wasted iterations chasing noise-induced bad updates means fewer wasted shots.

What a Negative Result Would Mean

If the protocol fails to improve any of the metrics, the dynamical-modification framing is not falsified — but its operational utility is. The framing would still be **correct** as a description of what regularization does in VQA; it would just turn out that under the specific noise levels and ansatz classes tested, the standard unregularized optimizer is already good enough.

In that case the recommendation would shift: skip the two-stage protocol in low-noise regimes, retain it as a tool for high-noise hardware-on-loop runs. This is a more nuanced position than "the protocol works" or "the protocol fails."

Structural Role

This is the empirical capstone of Module 5. It is also the **bridge** to Module 6 (Optimization Dynamics), where HOPSO is treated in full and the interaction between structured optimizer dynamics and regularization is analyzed. Many of the regimes in which regularization helps are regimes where HOPSO already does part of the job — the question of how the two interact is the natural next step.

Relations to Other Concepts

- **A/B testing under stochasticity** — how to compare two optimizers when their outputs are random variables.
- **Reproducibility in VQA literature** — many published VQE results are single-seed; this section argues for distributional reporting.
- **Measurement-optimizer co-design** — Module 10 picks this up.

Visualization

Pending. Diagram set: 1. *Box-and-whisker plot of final energies across seeds, four conditions side by side.* 2. *Energy-vs-shots curves with confidence bands, showing the protocol's narrower band.* 3. *Per-seed trajectories overlaid, illustrating the variance reduction visually.*

Exercises

1. Design a statistical test for whether two optimizers give significantly different best-energy distributions, given 20 seeds per condition. What is the appropriate test and why?
2. Suppose the protocol improves the median final energy by 1% but reduces the 90th-percentile-worst final energy by 10%. Which result is more important for production deployment, and why?
3. (VQA) Why is *iteration count* a misleading x-axis when comparing optimizers in the shot-noisy regime? What should the x-axis be instead?

Research Extensions

- **Joint shot-and-regularization scheduling** — co-adapting the shot budget and the regularization weight per iteration.
- **Hardware-on-loop validation** — running the protocol on real quantum hardware, where noise is correlated and time-varying rather than i.i.d.
- **Generalization to other ansatz families** — the protocol was tuned on `TwoLocal`; how transferable is it to UCCSD, ADAPT-VQE, or QAOA?

Cross-links to Other Courses

- **Statistics Module 8** — hypothesis testing for stochastic optimizers.
- **Optimization Dynamics (Module 6)** — the next chapter in the same story.
- **Hardware Noise Models (Module 9)** — the source of the noise this node fights against.