Landscape Geometry and Gradients

Variational Quantum Algorithms · Module 3 — Cost Landscapes

Node 3.5

Position in Knowledge Graph

System: Variational Quantum Algorithms **Structural Domain:** Cost Landscapes **Node:** 3.5

This node is about the **local** structure of a VQA cost landscape: gradient magnitude, curvature, the metric, and how these properties change as you move around parameter space.

Local geometry is what every optimizer "sees" — gradient descent uses the gradient, second-order methods use the Hessian, natural gradient uses the metric. Understanding local geometry is therefore the bridge between the topological vocabulary of nodes 3.2-3.4 and the operational behavior of an optimizer in Modules 5-7.

The Gradient

The cost function gradient is

$$\nabla C(\boldsymbol{\theta}) = \left(\frac{\partial C}{\partial \theta_1}, \ldots, \frac{\partial C}{\partial \theta_P} \right).$$

For a VQA with cost $C(\boldsymbol{\theta}) = \langle \psi(\boldsymbol{\theta}) | H | \psi(\boldsymbol{\theta}) \rangle$, each component is

$$\frac{\partial C}{\partial \theta_i} = \langle \partial_i \psi | H | \psi \rangle + \langle \psi | H | \partial_i \psi \rangle = 2\,\mathrm{Re}\langle \partial_i \psi | H | \psi \rangle,$$

where $|\partial_i \psi\rangle$ is the partial derivative of the state with respect to θ_i.

For circuit ansätze built from Pauli rotations $R_P(\theta) = e^{-i\theta P/2}$, this can be evaluated **exactly** using the parameter-shift rule (node 3.8):

$$\frac{\partial C}{\partial \theta_i} = \frac{1}{2} \left[C(\boldsymbol{\theta} + \tfrac{\pi}{2}\mathbf{e}_i) - C(\boldsymbol{\theta} - \tfrac{\pi}{2}\mathbf{e}_i) \right].$$

This identity is what makes gradient-based VQA optimization viable — exact gradients can be computed by *evaluating the cost function* at shifted parameter values, no finite-difference approximation needed.

The trade is that each gradient component costs two cost evaluations. For a P-parameter ansatz, computing the full gradient costs $2P$ cost evaluations — and each cost evaluation is itself M measurement bases $\times$ N shots.

Gradient Magnitude

How big is $\|\nabla C(\boldsymbol{\theta})\|$ at a typical point?

For shallow problem-inspired ansätze (UCCSD, HVA, ADAPT-VQE on chemistry instances), the gradient magnitude at the Hartree-Fock starting point is usually substantial — large enough to drive meaningful descent.

For random initializations of expressive HEAs, the gradient magnitude is *exponentially small in qubit count* — this is the barren plateau result, which Module 4 will treat in full. At $n = 20$ qubits, the typical gradient magnitude in a random HEA region can be 10^{-6} or smaller, far below the shot-noise floor of any reasonable budget.

At critical points, gradient is exactly zero by definition. Near a critical point, the gradient grows linearly with distance from the critical point, with the rate set by the Hessian eigenvalues.

The practical question is: **is the gradient distinguishable from shot noise?** Shot noise sets a floor on detectable gradient magnitudes:

$$\text{detectable iff} \quad \|\nabla C\| \gtrsim \frac{\sigma_{\text{shot}}}{\sqrt{\text{shots/component}}}.$$

When the gradient magnitude drops below this threshold, the optimizer effectively cannot distinguish "no signal" from "small signal" and progress stalls. This is the **shot noise floor** that Module 5 spends a lot of effort working around.

The Hessian

The second-order structure is captured by the Hessian:

$$\mathbf{H}_{ij}(\boldsymbol{\theta}) = \frac{\partial^2 C}{\partial\theta_i \partial\theta_j}.$$

For Pauli-rotation ansätze, the Hessian can also be computed via parameter shifts (a "double-shift rule"), but each component costs four cost evaluations, so the full $P \times P$ Hessian costs $O(P^2)$ cost evaluations. For $P = 50$, that's 10,000 cost evaluations per Hessian — generally prohibitive.

The Hessian eigenvalues at a critical point classify it:

- **All eigenvalues positive**: local minimum.
- **All negative**: local maximum.
- **Mixed signs**: saddle point. The number of negative eigenvalues is the **Morse index** of the saddle.
- **Some zero eigenvalues**: degenerate critical point. Often signals a symmetry or gauge direction.

The **largest eigenvalue** of the Hessian is the **local Lipschitz constant** of the gradient and bounds how large a gradient-descent step you can take without overshooting.

The **condition number** of the Hessian (ratio of largest to smallest eigenvalue) determines convergence rate of gradient descent — large condition numbers mean slow convergence in the worst direction.

The Quantum Fisher Metric

The natural metric on the ansatz manifold is the **quantum Fisher information matrix** (QFIM):

$$g_{ij}(\boldsymbol{\theta}) = 4 \cdot \mathrm{Re}\left[\langle\partial_i\psi|\partial_j\psi\rangle - \langle\partial_i\psi|\psi\rangle\langle\psi|\partial_j\psi\rangle\right].$$

This is the **Fubini-Study metric** pulled back through the parameterization, scaled to make the Cramér-Rao bound clean.

The QFIM measures how distinguishable nearby parameter values are in terms of any possible quantum measurement. Two parameter directions with small g_{ii} produce nearly indistinguishable states; two with large g_{ii} produce very different states.

Crucially, the QFIM is **distinct from the Hessian of the cost function**. The Hessian depends on the cost function (and therefore on the Hamiltonian); the QFIM depends only on the ansatz.

The natural gradient update rule (Module 6) uses the QFIM to rescale the cost gradient:

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

This produces updates that are *invariant under reparameterization*: changing the way you label parameters doesn't change the trajectory in state space. Vanilla gradient descent does not have this property and can be wildly different under different parameterizations of the same ansatz.

Local Curvature and Step Size

The relationship between gradient magnitude and Hessian curvature determines the appropriate step size for gradient descent.

For a quadratic approximation $C(\boldsymbol{\theta}_k + \mathbf{d}) \approx C(\boldsymbol{\theta}_k) + \nabla C^T \mathbf{d} + \frac{1}{2}\mathbf{d}^T \mathbf{H} \mathbf{d}$, the optimal step in the gradient direction is

$$\eta^* = \frac{\|\nabla C\|^2}{\nabla C^T \mathbf{H} \nabla C}.$$

If you take a smaller step you make less progress; if you take a larger step you may overshoot. Adaptive learning rate methods (Adam, RMSProp) approximate this implicitly by scaling each component by an estimate of recent gradient variance, which is correlated with curvature.

In VQA practice, **the Hessian varies significantly across parameter space**, so a single global step size is rarely optimal. This is one reason adaptive methods often outperform vanilla gradient descent on VQAs.

Why Local Geometry Doesn't Tell You Everything

A subtle but important point: **local geometry is not sufficient to navigate a non-convex landscape.**

Even with perfect gradient and Hessian information at every point, a local optimizer: - Cannot tell whether the current basin is the global one. - Cannot find escape routes to better basins. - Cannot distinguish a long valley from a saddle. - Cannot detect concentration plateaus (Module 4) until it's already in one.

This is why **global** information — coming from population methods, surrogate models, or symbolic analysis — is often needed for hard VQA instances. Local geometry is the optimizer's microscope; it tells you the immediate slope but not the overall topology.

The thesis's choice of HOPSO as a representative population-based optimizer (Module 4-5 of the thesis) is motivated in part by this observation: HOPSO maintains a population of points that collectively sample the global structure, while each individual member follows local gradient information.

Structural Role

This node is the **local-structure** node of Module 3. It connects the topological vocabulary (3.2-3.4) to the gradient-computation machinery (3.8) and forwards to the statistical phenomena (3.6, 3.7) and barren plateaus (Module 4).

It is also the place where the QFIM is introduced informally; Module 6 will give it the full formal treatment as the source of the natural gradient.

Relations to Other Concepts

- **Calculus of variations** — first- and second-order necessary conditions for optimality.
- **Information geometry** — the Fisher metric and its quantum analogue.
- **Riemannian optimization** — optimizing on manifolds with non-trivial metrics.
- **Lipschitz constants** — the bound on gradient change that controls step size.
- **Newton's method** — second-order optimization that uses the inverse Hessian.

Worked Example — Gradient Of The Single-Qubit $H = Z$ Cost

For $|\psi(\theta)\rangle = R_Y(\theta)|0\rangle$ and $H = Z$:

$$C(\theta) = \cos(\theta), \quad C'(\theta) = -\sin(\theta), \quad C''(\theta) = -\cos(\theta).$$

At $\theta = 0$: $C = 1$, $C' = 0$, $C'' = -1$. Local maximum. At $\theta = \pi$: $C = -1$, $C' = 0$, $C'' = 1$. Local minimum (also global). At $\theta = \pi/2$: $C = 0$, $C' = -1$, $C'' = 0$. Inflection point.

Gradient magnitude at $\theta = \pi/2$ is 1, the largest possible. At $\theta = 0$ and $\theta = \pi$ it is 0.

The Hessian eigenvalue at $\theta = \pi$ is $+1$, so the local Lipschitz constant of the gradient is 1, and gradient descent with step size $\eta < 1$ is guaranteed to converge near this minimum.

For shot-noisy estimation with $N = 100$ shots and $\mathrm{Var}[\widehat{Z}] = 1 - \cos^2(\theta) = \sin^2(\theta)$, the noise floor is $\sin(\theta)/\sqrt{100} = 0.1\sin(\theta)$. At θ near π, this floor approaches 0 — the noise vanishes near the eigenstate. But the gradient also vanishes there, so the SNR doesn't necessarily improve.

This is a clean toy example of the relationship between gradient magnitude, curvature, and shot noise. **The optimizer is happy near the minimum because both signal and noise are small in absolute terms — what matters is their ratio.**

Visualization

Pending. A 1D plot of $C(\theta) = \cos(\theta)$ over $[-\pi, \pi]$, with overlay annotations: gradient arrow at $\theta = \pi/2$, curvature label at $\theta = \pi$, noise band around the curve. Caption: "Gradient and curvature, the optimizer's microscope."

Exercises

1. For $C(\theta_1, \theta_2) = \cos(\theta_1)\cos(\theta_2)$, compute the Hessian at the four critical points $(0,0), (0,\pi), (\pi,0), (\pi,\pi)$. Classify each.
2. For the same function, compute the QFIM g_{ij} for the parameterization $R_Y(\theta_1) \otimes R_Y(\theta_2)|00\rangle$. Compare to the Hessian of C. Are they the same? Why or why not?

3. (VQA) Estimate the typical gradient magnitude in a random HEA with 10 qubits and 5 layers from the Pauli-shift formula (without computing the full barren plateau theorem). Compare to the shot noise floor at $N = 1024$ shots per evaluation.

Research Extensions

- **Curvature-aware schedules** — using local Hessian information to adapt the step size online.
- **Approximate natural gradient** — block-diagonal or low-rank approximations to the QFIM that are cheap to compute.
- **Hessian estimation via parameter shifts** — efficient protocols that don't require $O(P^2)$ circuit runs.
- **Landscape topography from local probes** — reconstructing the global topology from local gradient/Hessian samples.

Cross-links to Other Courses

- **Calculus / Differential Geometry** — gradients, Hessians, Riemannian metrics.
- **Information Geometry** — Fisher information, quantum Fisher information.
- **Numerical Optimization** — Newton's method, quasi-Newton, trust regions.
- **Module 6 (Optimization Dynamics)** — the natural gradient and other geometric methods.

Variational Quantum Algorithms · Module 3 — Cost Landscapes · Node 3.5
Concepts: gradient-descent, cost-function, optimization-landscape, fisher-information
Prerequisites: 3.2, 3.3
Enables: 3.6, 3.8

hbar.university — own.your.cognition