Module 4 — Optimization Regularization

Mathematical Intuition

hbar.university

Lesson 4.1: Parameterized States

Overview

Understanding parameterized quantum circuits (ansätze) as a way to explore the Hilbert space of quantum states.

Learning Objectives

- See ansätze as parameterized families of quantum states
- Understand how circuit depth and structure affect expressibility
- Recognize common ansatz types and their trade-offs
- Build intuition for the relationship between parameters and state space

Core Concepts

What Is an Ansatz?

Definition

- **Ansatz**: "educated guess" for the form of the solution
- **Parameterized circuit**: $\mathrm{U}(\theta)$ with tunable parameters $\theta = (\theta_1, \theta_2, \ldots, \theta_p)$
- **Trial state**: $|\psi(\theta)\rangle = \mathrm{U}(\theta)|0\rangle^n$ (starting from computational basis)
- **Goal**: find θ^* such that $|\psi(\theta^*)\rangle$ approximates ground state

Why Parameterize?

- **Exploration**: parameters let us search through state space
- **Optimization**: classical optimizer adjusts θ to minimize cost
- **Hardware efficiency**: shallow circuits reduce noise
- **Physical intuition**: structure can encode problem symmetries

Ansatz Structure

Single-Qubit Rotations

- $\mathbf{R}_x(\theta) = \mathbf{e\hat{}(-i\theta\sigma_x/2)}$: rotation around x-axis
- $\mathbf{R}_\gamma(\theta) = \mathbf{e\hat{}(-i\theta\sigma_\gamma/2)}$: rotation around y-axis
- $\mathbf{R}_u(\theta) = \mathbf{e\hat{}(-i\theta\sigma_u/2)}$: rotation around z-axis
- $\mathbf{U}_3(\theta,\varphi,\lambda)$: general single-qubit rotation

Entangling Gates

- **CNOT**: creates entanglement between qubits
- **CZ**: controlled-Z gate

- **RZZ(θ) = e^(-iθZZ/2)**: parameterized two-qubit gate

Layer Structure

- **Rotation layer**: apply $R_\gamma(\theta_i)$ to each qubit
- **Entangling layer**: apply CNOTs or other two-qubit gates
- **Repeat**: multiple layers increase expressibility

Common Ansatz Types

Hardware-Efficient Ansatz

- **Structure**: matches native gate set of quantum hardware
- **Pros**: shallow circuits, low noise
- **Cons**: may not capture problem structure
- **Example**: alternating single-qubit rotations and CNOTs

Unitary Coupled Cluster (UCC)

- **Structure**: based on quantum chemistry excitations
- **Pros**: physically motivated, systematic improvement
- **Cons**: deep circuits, many parameters
- **Example**: UCCSD (singles and doubles excitations)

QAOA-Inspired Ansatz

- **Structure**: alternating problem and mixer Hamiltonians
- **Pros**: good for combinatorial optimization
- **Cons**: requires problem-specific design
- **Example**: $e^{(-i\beta H\_mixer)}e$(-iγH_problem)

Symmetry-Preserving Ansatz

- **Structure**: respects problem symmetries (particle number, spin, etc.)
- **Pros**: reduces parameter space, improves optimization
- **Cons**: requires symmetry analysis
- **Example**: particle-conserving gates

Expressibility vs Trainability

Expressibility

- **How much of Hilbert space can the ansatz reach?**
- More parameters → more expressible
- Deeper circuits → more expressible
- Trade-off: expressibility vs circuit depth (noise)

Trainability

- **How easy is it to optimize the parameters?**
- Too many parameters → barren plateaus (vanishing gradients)
- Too few parameters → can't reach good solutions
- Sweet spot: enough expressibility, trainable gradients

Barren Plateaus

- **Problem**: gradients vanish exponentially with circuit depth
- **Cause**: random circuits have exponentially small gradients
- **Solutions**: problem-inspired ansätze, local cost functions, parameter initialization

Examples in Context

Quantum Chemistry

- **Ansatz**: UCC (unitary coupled cluster)
- **State**: $|\psi(\theta)\rangle = \mathrm{e\hat{}(T(\theta)\text{-}T\dagger(\theta))|HF\rangle}$
- **Parameters**: excitation amplitudes
- **Goal**: find molecular ground state energy

Combinatorial Optimization

- **Ansatz**: QAOA (Quantum Approximate Optimization Algorithm)
- **State**: $|\psi(\beta,\gamma)\rangle = \prod_p \mathrm{e}^{(\text{-i}\beta_p \mathrm{H\_mixer})}\mathrm{e}(\text{-i}\gamma_p \mathrm{H\_cost})|+\rangle^n$
- **Parameters**: mixing angles β, cost angles γ
- **Goal**: find optimal solution to combinatorial problem

Quantum Machine Learning

- **Ansatz**: data-encoding + variational layers
- **State**: $|\psi(\mathrm{x},\theta)\rangle = \mathrm{U}(\theta)\mathrm{U\_encode(x)}|0\rangle^n$
- **Parameters**: trainable weights θ
- **Goal**: learn function $\mathrm{f(x)} = \langle\psi(\mathrm{x},\theta)|\mathrm{M}|\psi(\mathrm{x},\theta)\rangle$

Designing an Ansatz

Step 1: Understand the Problem

- What symmetries does the problem have?
- What's the expected structure of the solution?
- What's the available quantum hardware?

Step 2: Choose Structure

- Hardware-efficient for NISQ devices
- Problem-inspired for better optimization
- Balance expressibility and trainability

Step 3: Initialize Parameters

- Random initialization can lead to barren plateaus
- Problem-inspired initialization (e.g., classical solution)
- Layerwise training

Step 4: Test and Iterate

- Check expressibility: can it reach good states?
- Check trainability: do gradients vanish?
- Adjust depth, structure, initialization

Practice Problems

1. How many parameters does a hardware-efficient ansatz with L layers and n qubits have?
2. Why might a UCC ansatz be better than a hardware-efficient ansatz for quantum chemistry?
3. What is a barren plateau, and how can you avoid it?

Key Takeaways

- Ansätze are parameterized quantum circuits that explore state space
- Structure matters: hardware-efficient vs problem-inspired
- Trade-off: expressibility (can reach good states) vs trainability (can optimize)
- Barren plateaus are a major challenge in deep variational circuits
- Good ansatz design requires understanding the problem and hardware

Next Steps

Next, we'll explore cost functions—how to define what we're trying to minimize in VQE.

Lesson 4.2: Cost Functions

Overview

Understanding cost functions in VQE as energy expectation values and how to design and measure them on quantum hardware.

Learning Objectives

- See cost functions as expectation values of Hamiltonians
- Understand how to decompose Hamiltonians into measurable terms
- Recognize the variational principle and energy bounds
- Build intuition for cost function landscapes

Core Concepts

Energy Expectation Value

Definition

- $\mathbf{E(\theta)} = \langle\psi(\theta)|\hat{\mathbf{H}}|\psi(\theta)\rangle$: energy of state $|\psi(\theta)\rangle$
- $\hat{\mathbf{H}}$: Hamiltonian (energy operator)
- **Goal**: minimize $E(\theta)$ to find ground state
- **Variational principle**: $E(\theta) \geq E_0$ (ground state energy)

Why Energy?

- Ground state has minimum energy
- Energy is a Hermitian observable (real-valued)
- Variational principle guarantees bound
- Physical meaning: total energy of quantum system

Hamiltonian Structure

Pauli Decomposition

- $\hat{\mathbf{H}} = \Sigma_i\ \mathbf{h}_i\mathbf{P}_i$: sum of Pauli strings
- $\mathbf{P}_i$: tensor product of Pauli operators (I, X, Y, Z)
- $\mathbf{h}_i$: real coefficients
- **Example**: $\hat{H} = 0.5 \cdot Z_0Z_1 + 0.3 \cdot X_0 - 0.2 \cdot Y_1$

Measuring Pauli Strings

- $\langle\mathbf{P}_i\rangle = \langle\psi|\mathbf{P}_i|\psi\rangle$: expectation of Pauli string
- Measure in eigenbasis of P_i
- Eigenvalues: ± 1 for Pauli operators
- Estimate: average over many measurements

Total Energy

- $\mathbf{E(\theta)} = \Sigma_i\ \mathbf{h}_i\langle\mathbf{P}_i\rangle$: sum of weighted expectations
- Each term measured separately
- Total measurements: O(number of terms)

Variational Principle

Upper Bound

- $\mathbf{E}(\theta) \geq \mathbf{E}_0$ for all θ
- Any trial state gives upper bound on ground energy
- Minimizing $E(\theta)$ approaches E_0

Excited States

- Can target excited states with constraints
- Orthogonalize to lower states
- Penalty terms for overlap

Cost Function Design

Standard VQE

- $\mathbf{C}(\theta) = \langle\psi(\theta)|\hat{\mathbf{H}}|\psi(\theta)\rangle$: just the energy
- Simple, direct
- May have many local minima

With Constraints

- $\mathbf{C}(\theta) = \mathbf{E}(\theta) + \lambda \cdot \mathbf{penalty}(\theta)$: energy + constraint
- Penalty enforces symmetries or other requirements
- Example: particle number conservation

Subspace VQE

- $\mathbf{C}(\theta) = \langle\psi(\theta)|\hat{\mathbf{H}}|\psi(\theta)\rangle$ in subspace
- Reduces problem size
- Exploits symmetries

Measurement Strategies

Individual Pauli Measurements

- Measure each Pauli string separately
- Rotate basis: apply gates before measurement
- Estimate expectation from measurement outcomes

Grouping Commuting Terms

- $[\mathbf{P}_i,\mathbf{P}_j] = \mathbf{0}$: can measure simultaneously
- Group commuting Pauli strings
- Reduces total measurements

Shadow Tomography

- Estimate many observables from few measurements
- Classical post-processing
- Efficient for large Hamiltonians

Examples in Context

Quantum Chemistry

- **Hamiltonian**: molecular electronic structure
- $\hat{\mathbf{H}} = \Sigma_{ij}\ \mathbf{h}_{ij}\hat{\mathbf{a}}_i\dagger\hat{\mathbf{a}}_j + \Sigma_{ijkl}\ \mathbf{h}_{ijkl}\hat{\mathbf{a}}_i\dagger\hat{\mathbf{a}}_j\dagger\hat{\mathbf{a}}_k\hat{\mathbf{a}}_l$
- Map to qubits: Jordan-Wigner, Bravyi-Kitaev
- Measure energy in Pauli basis

Combinatorial Optimization

- **Hamiltonian**: encodes optimization problem
- $\hat{\mathbf{H}} = \Sigma_{ij}\ \mathbf{J}_{ij}\sigma_i\sigma_j + \Sigma_i\ \mathbf{h}_i\sigma_i$ (Ising model)
- Ground state = optimal solution
- Measure cost function value

Quantum Simulation

- **Hamiltonian**: physical system to simulate
- $\hat{\mathbf{H}}$ **= $\hat{\mathbf{H}}$_kinetic + $\hat{\mathbf{H}}$_potential + $\hat{\mathbf{H}}$_interaction**
- Find ground state or dynamics
- Measure observables of interest

Cost Function Landscape

Local Minima

- Cost function may have many local minima
- Optimization can get stuck
- Strategies: multiple initializations, simulated annealing

Barren Plateaus

- Gradients vanish in large regions
- Exponentially hard to escape
- Related to ansatz design (see Lesson 4.1)

Noise Effects

- Measurement noise adds variance
- Coherent errors bias energy estimates
- Regularization can help (see Lesson 4.4)

Practical Considerations

Shot Budget

- **N_shots**: number of measurements per Pauli string
- More shots → better estimate, but more time
- Trade-off: accuracy vs runtime

Error Mitigation

- Zero-noise extrapolation
- Probabilistic error cancellation
- Symmetry verification

Adaptive Measurements

- Allocate shots based on term importance
- More shots for large coefficients h_i
- Reduces total measurement cost

Practice Problems

1. If $\hat{H} = 2 \cdot Z_0 Z_1 + 3 \cdot X_0 X_1$, how many measurement bases do you need?
2. Why does the variational principle guarantee $E(\theta) \geq E_0$?
3. How does measurement noise affect the cost function estimate?

Key Takeaways

- Cost function is energy expectation value $E(\theta) = \langle \psi(\theta) | \hat{H} | \psi(\theta) \rangle$
- Hamiltonians decompose into measurable Pauli strings
- Variational principle: any trial state gives upper bound on ground energy
- Measurement strategies: individual, grouped, or shadow tomography
- Cost landscape has local minima and barren plateaus

Next Steps

Next, we'll explore gradients—how to compute derivatives of the cost function using parameter-shift rules.

Lesson 4.3: Gradients

Overview

Understanding how to compute gradients of quantum cost functions using parameter-shift rules and other techniques.

Learning Objectives

- See gradients as directions of steepest descent
- Understand the parameter-shift rule for quantum circuits
- Recognize when finite differences vs parameter-shift is appropriate
- Build intuition for gradient-based optimization on quantum hardware

Core Concepts

Why Gradients?

Optimization

- **Gradient descent**: θ_new = θ_old - $\alpha \cdot \nabla E(\theta)$
- Gradient points toward steepest increase
- Move opposite direction to minimize
- Requires computing $\partial E/\partial\theta_i$ for each parameter

Classical vs Quantum

- **Classical**: backpropagation (chain rule)
- **Quantum**: parameter-shift rule (analytical gradients)
- Can't directly access quantum state
- Must use measurement outcomes

Parameter-Shift Rule

Single-Parameter Gates

- **Gate**: $U(\theta) = e\hat{}(-i\theta\hat{G}/2)$ where $\hat{G}^2 = I$ (generator)
- **Gradient**: $\partial\langle\hat{H}\rangle/\partial\theta = [\langle\hat{H}\rangle(\theta + \pi/2) - \langle\hat{H}\rangle(\theta - \pi/2)]/2$
- **Intuition**: shift parameter by $\pm\pi/2$, take difference
- **Exact**: not an approximation (for Pauli generators)

Why It Works

- **Taylor expansion**: $\langle\psi(\theta)|\hat{H}|\psi(\theta)\rangle = a + b \cdot \cos(\theta) + c \cdot \sin(\theta)$
- **Derivative**: $d/d\theta = -b \cdot \sin(\theta) + c \cdot \cos(\theta)$
- **Shift formula**: recovers derivative exactly
- **Measurement-based**: only need expectation values

Multi-Parameter Case

- **Partial derivatives**: $\partial E/\partial\theta_i$ computed independently
- **Gradient**: $\nabla E = [\partial E/\partial\theta_1, \partial E/\partial\theta_2, \ldots, \partial E/\partial\theta_p]^T$
- **Cost**: 2p circuit evaluations (p parameters)

Finite Differences

Forward Difference

- $\partial \mathbf{E}/\partial\theta \approx [\mathbf{E}(\theta + \varepsilon) - \mathbf{E}(\theta)]/\varepsilon$
- Simple, but approximate
- Error: $O(\varepsilon)$

Central Difference

- $\partial \mathbf{E}/\partial\theta \approx [\mathbf{E}(\theta + \varepsilon) - \mathbf{E}(\theta - \varepsilon)]/(\mathbf{2}\varepsilon)$
- More accurate
- Error: $O(\varepsilon^2)$

When to Use

- Non-Pauli generators
- Quick prototyping
- Classical simulation

Gradient-Based Optimizers

Gradient Descent

- $\theta \leftarrow \theta - \alpha \cdot \nabla \mathbf{E}(\theta)$: basic update rule
- α is learning rate
- Can be slow, sensitive to learning rate

Momentum

- $\mathbf{v} \leftarrow \beta\mathbf{v} + \nabla \mathbf{E}(\theta)$: accumulate momentum
- $\theta \leftarrow \theta - \alpha \cdot \mathbf{v}$: update with momentum
- Helps escape shallow local minima

Adam

- $\mathbf{m} \leftarrow \beta_1\mathbf{m} + (\mathbf{1}\text{-}\beta_1)\nabla\mathbf{E}$: first moment (mean)
- $\mathbf{v} \leftarrow \beta_2\mathbf{v} + (\mathbf{1}\text{-}\beta_2)(\nabla\mathbf{E})^2$: second moment (variance)
- $\theta \leftarrow \theta - \alpha \cdot \mathbf{m}/\sqrt{(\mathbf{v} + \varepsilon)}$: adaptive learning rate
- Popular in machine learning

Quantum Natural Gradient

- $\theta \leftarrow \theta - \alpha \cdot \mathbf{F}^{-1}\nabla\mathbf{E}$: use Fisher information matrix F
- Accounts for geometry of parameter space
- More efficient, but requires computing F

Challenges in Quantum Gradients

Barren Plateaus

- **Problem**: gradients vanish exponentially with system size
- **Cause**: random circuits have exponentially small gradients
- **Solutions**: problem-inspired ansätze, local cost functions

Shot Noise

- **Problem**: finite measurements add noise to gradient estimates
- **Effect**: noisy gradients slow optimization
- **Solutions**: more shots, adaptive shot allocation, gradient filtering

Hardware Noise

- **Problem**: gate errors and decoherence
- **Effect**: biased gradient estimates
- **Solutions**: error mitigation, shorter circuits

Examples in Context

VQE for Quantum Chemistry

- **Cost**: $\mathrm{E}(\theta) = \langle\psi(\theta)|\hat{\mathrm{H}}\_\mathrm{mol}|\psi(\theta)\rangle$
- **Gradient**: $\partial\mathrm{E}/\partial\theta_i$ via parameter-shift
- **Optimizer**: BFGS, Adam, or quantum natural gradient
- **Goal**: minimize molecular energy

QAOA for MaxCut

- **Cost**: $\mathrm{E}(\beta,\gamma) = \langle\psi(\beta,\gamma)|\hat{\mathrm{H}}\_\mathrm{cut}|\psi(\beta,\gamma)\rangle$
- **Gradient**: $\partial\mathrm{E}/\partial\beta_p$, $\partial\mathrm{E}/\partial\gamma_p$ via parameter-shift
- **Optimizer**: gradient descent with momentum
- **Goal**: maximize cut value (minimize -E)

Quantum Machine Learning

- **Cost**: $\mathrm{L}(\theta) = \Sigma_i(\mathrm{y}_i$ - $\mathrm{f}(\mathrm{x}_i,\theta))^2$
- **Gradient**: $\partial\mathrm{L}/\partial\theta_j$ via parameter-shift on each sample
- **Optimizer**: mini-batch gradient descent
- **Goal**: learn function f

Parameter-Shift Variants

General Shift Rule

- **For $\hat{\mathbf{G}}$ with eigenvalues** $\pm\lambda$: shift by $\pm\pi/(4\lambda)$
- **Multiple eigenvalues**: more shift terms needed
- **Example**: $\hat{\mathrm{G}} = \mathrm{X} + \mathrm{Y}$ requires 4 shifts

Stochastic Parameter-Shift

- **Randomly sample shift directions**
- Reduces measurement cost
- Unbiased gradient estimate

Simultaneous Perturbation

- **Perturb all parameters at once**
- Estimate gradient direction
- Fewer circuit evaluations

Practical Implementation

Step 1: Define Cost Function

- $\mathrm{E}(\theta) = \langle\psi(\theta)|\hat{\mathrm{H}}|\psi(\theta)\rangle$

Step 2: Compute Shifted Costs

- For each θ_i: compute $\mathrm{E}(\theta + \pi/2 \cdot \mathrm{e}_i)$ and $\mathrm{E}(\theta$ - $\pi/2 \cdot \mathrm{e}_i)$

Step 3: Calculate Gradient

- $\partial E/\partial \theta_i = [E(\theta + \pi/2 \cdot e_i) - E(\theta - \pi/2 \cdot e_i)]/2$

Step 4: Update Parameters

- $\theta \leftarrow \theta - \alpha \cdot \nabla E(\theta)$

Step 5: Repeat

- Until convergence or max iterations

Practice Problems

1. How many circuit evaluations are needed to compute the gradient for p parameters?
2. Why is the parameter-shift rule exact for Pauli generators?
3. What is the advantage of quantum natural gradient over standard gradient descent?

Key Takeaways

- Gradients guide optimization toward lower cost
- Parameter-shift rule gives exact gradients from measurements
- Requires 2p circuit evaluations for p parameters
- Barren plateaus and shot noise are major challenges
- Various optimizers: gradient descent, momentum, Adam, quantum natural gradient

Next Steps

Next, we'll explore regularization—how to add penalty terms to control the optimization landscape and improve convergence.

Lesson 4.4: Regularization

Overview

Understanding regularization as a way to control optimization landscapes, enforce constraints, and improve VQE performance.

Learning Objectives

- See regularization as penalty terms added to cost functions
- Understand L1, L2, and physics-inspired regularization
- Recognize how regularization shapes optimization landscapes
- Build intuition for when and how to regularize VQE

Core Concepts

What Is Regularization?

Definition

- **Regularized cost**: C_total(θ) = E(θ) + $\lambda \cdot$ R(θ)
- **E(θ)**: original cost (energy)
- **R(θ)**: regularization term (penalty)
- λ: regularization strength (hyperparameter)

Why Regularize?

- **Prevent overfitting**: in quantum machine learning
- **Enforce constraints**: symmetries, particle number, etc.
- **Smooth landscapes**: reduce local minima, improve gradients
- **Bias toward solutions**: prefer certain parameter regions

Types of Regularization

L2 Regularization (Weight Decay)

- **R(θ) = $||\theta||^2 = \Sigma_i {\theta_i}^2$**: penalize large parameters
- **Effect**: keeps parameters small, smooth solutions
- **Gradient**: $\partial R/\partial\theta_i = 2\theta_i$ (simple to compute)
- **Use case**: prevent parameter explosion, improve generalization

L1 Regularization (Sparsity)

- **R(θ) = $||\theta||_1 = \Sigma_i|\theta_i|$**: penalize non-zero parameters
- **Effect**: encourages sparse solutions (many $\theta_i = 0$)
- **Gradient**: $\partial R/\partial\theta_i$ = sign(θ_i) (non-differentiable at 0)
- **Use case**: feature selection, circuit compression

Entropy Regularization

- **R(θ) = -S($\rho(\theta)$)**: penalize low entropy (pure states)
- **S(ρ) = -Tr(ρ log ρ)**: von Neumann entropy
- **Effect**: encourages mixed states, exploration
- **Use case**: quantum machine learning, thermal states

Physics-Inspired Regularization

Symmetry Enforcement

- $\mathbf{R}(\theta) = ||\hat{\mathbf{S}}|\psi(\theta)\rangle$ - $\mathbf{s}|\psi(\theta)\rangle||^2$: penalize symmetry violation
- $\hat{\mathbf{S}}$: symmetry operator (e.g., total spin, particle number)
- $\mathbf{s}$: target eigenvalue
- **Effect**: keeps state in correct symmetry sector
- **Use case**: quantum chemistry, many-body physics

Overlap Penalties

- $\mathbf{R}(\theta) = |\langle\psi\_\mathbf{ref}|\psi(\theta)\rangle|^2$: penalize overlap with reference state
- **Effect**: orthogonalize to known states (for excited states)
- **Use case**: excited state VQE

Gradient Regularization

- $\mathbf{R}(\theta) = ||\nabla\mathbf{E}(\theta)||^2$: penalize large gradients
- **Effect**: smooths cost landscape
- **Use case**: mitigate barren plateaus

Regularization in VQE

Standard VQE

- $\mathbf{C}(\theta) = \langle\psi(\theta)|\hat{\mathbf{H}}|\psi(\theta)\rangle$: no regularization
- May have rough landscape, local minima

Regularized VQE

- $\mathbf{C}(\theta) = \langle\psi(\theta)|\hat{\mathbf{H}}|\psi(\theta)\rangle + \lambda||\theta||^2$: L2 regularization
- Smoother landscape, easier optimization
- Trade-off: may not reach exact ground state

Adaptive Regularization

- $\lambda(\mathbf{t})$: decrease λ over optimization
- Start with strong regularization (smooth landscape)
- Gradually reduce to recover exact solution
- Similar to simulated annealing

Choosing Regularization Strength

Too Small ($\lambda \to 0$)

- Regularization has no effect
- Original rough landscape
- May get stuck in local minima

Too Large ($\lambda \to \infty$)

- Regularization dominates
- Solution biased away from true minimum
- May not find good solution

Just Right

- Balances original cost and regularization
- Improves optimization without biasing too much
- Often found via hyperparameter search

Examples in Context

Quantum Chemistry VQE

- **Regularization**: particle number conservation
- $\mathbf{R}(\theta) = (\langle \mathbf{N} \rangle$ **- N_target)**^{2}: penalize wrong particle number
- **Effect**: keeps state in correct Fock space sector
- **Result**: faster convergence, physically valid states

QAOA for Optimization

- **Regularization**: L2 on angles
- $\mathbf{R}(\beta,\gamma) = ||\beta||^2 + ||\gamma||^2$: keep angles moderate
- **Effect**: prevents extreme parameter values
- **Result**: more stable optimization

Quantum Machine Learning

- **Regularization**: L2 on circuit parameters
- $\mathbf{R}(\theta) = \lambda||\theta||^2$: prevent overfitting
- **Effect**: simpler models, better generalization
- **Result**: improved test set performance

Excited State VQE

- **Regularization**: orthogonality to ground state
- $\mathbf{R}(\theta) = |\langle \psi_0 | \psi(\theta) \rangle|^2$: penalize overlap with $|\psi_0\rangle$
- **Effect**: pushes state away from ground state
- **Result**: finds first excited state

Regularization and Noise

Noise as Implicit Regularization

- Hardware noise acts like regularization
- Decoherence smooths cost landscape
- Can help escape local minima (but biases solution)

Explicit Regularization for Noise

- Add regularization to counteract noise effects
- Example: penalize states sensitive to noise
- Improves robustness on NISQ devices

Practical Implementation

Step 1: Choose Regularization Type

- L2 for smoothness
- L1 for sparsity
- Physics-inspired for constraints

Step 2: Set Initial λ

- Start with small λ (e.g., 0.01)
- Increase if optimization struggles
- Decrease if solution is biased

Step 3: Monitor Both Terms

- Track $E(\theta)$ and $R(\theta)$ separately
- Ensure regularization isn't dominating
- Adjust λ if needed

Step 4: Adaptive Schedule (Optional)

- Start with large λ
- Decrease over optimization
- Final $\lambda \to 0$ for exact solution

Practice Problems

1. How does L2 regularization affect the gradient $\nabla C(\theta)$?
2. Why might you want to enforce particle number conservation in quantum chemistry VQE?
3. What's the trade-off between regularization strength and solution accuracy?

Key Takeaways

- Regularization adds penalty terms to cost functions
- L2 smooths landscapes; L1 encourages sparsity
- Physics-inspired regularization enforces constraints
- Regularization strength λ must be tuned carefully
- Can improve optimization but may bias solutions

Next Steps

Next, we'll explore how noise affects VQE cost landscapes and strategies to navigate noisy optimization.

Lesson 4.5: Noise & Landscapes

Overview

Understanding how quantum noise affects VQE cost landscapes and developing strategies to navigate noisy optimization.

Learning Objectives

- See how noise transforms ideal cost landscapes
- Understand different types of quantum noise and their effects
- Recognize noise-induced local minima and bias
- Build intuition for error mitigation strategies

Core Concepts

Types of Quantum Noise

Decoherence

- **T_1 decay**: energy relaxation ($|1\rangle \to |0\rangle$)
- **T_2 decay**: dephasing (loss of phase coherence)
- **Effect**: mixed states, reduced purity
- **Impact on VQE**: biased energy estimates, reduced expressibility

Gate Errors

- **Over/under-rotation**: θ_actual $\neq$ θ_intended
- **Crosstalk**: unintended interactions between qubits
- **Effect**: wrong unitary applied
- **Impact on VQE**: incorrect state preparation, biased gradients

Measurement Errors

- **Readout errors**: measure $|0\rangle$ when state is $|1\rangle$ (and vice versa)
- **SPAM errors**: state preparation and measurement
- **Effect**: incorrect measurement outcomes
- **Impact on VQE**: noisy cost function estimates

How Noise Affects Cost Landscapes

Ideal Landscape

- **Smooth**: well-defined minima
- **Deterministic**: same θ gives same E(θ)
- **Gradient**: points toward minimum

Noisy Landscape

- **Rough**: noise adds high-frequency oscillations
- **Stochastic**: E(θ) varies between runs (shot noise)
- **Biased**: systematic errors shift minimum
- **Flattened**: decoherence reduces landscape features

Noise-Induced Phenomena

Local Minima

- Noise can create spurious local minima
- Optimizer gets trapped
- Solution: multiple initializations, simulated annealing

Barren Plateaus

- Gradients vanish in large regions
- Exponentially hard to escape
- Worse with noise and deep circuits

Gradient Bias

- Noise biases gradient estimates
- Optimizer moves in wrong direction
- Solution: error mitigation, more shots

Energy Bias

- Decoherence typically increases energy
- Ground state estimate is too high
- Solution: extrapolation, error mitigation

Error Mitigation Strategies

Zero-Noise Extrapolation (ZNE)

- **Idea**: run at different noise levels, extrapolate to zero noise
- **Method**: amplify noise (e.g., pulse stretching), fit curve, extrapolate
- **Effect**: reduces systematic bias
- **Cost**: multiple circuit runs

Probabilistic Error Cancellation (PEC)

- **Idea**: represent noisy channel as linear combination of noiseless operations
- **Method**: sample from quasi-probability distribution
- **Effect**: unbiased estimates (but high variance)
- **Cost**: many more measurements

Symmetry Verification

- **Idea**: check if state respects known symmetries
- **Method**: measure symmetry operators, post-select or correct
- **Effect**: filters out unphysical states
- **Cost**: additional measurements

Measurement Error Mitigation

- **Idea**: characterize readout errors, invert confusion matrix
- **Method**: calibration circuits, linear inversion
- **Effect**: corrects measurement outcomes
- **Cost**: calibration overhead

Noise-Aware Optimization

Adaptive Shot Allocation

- **Idea**: allocate more shots where gradient is uncertain

- **Method**: estimate gradient variance, increase shots accordingly
- **Effect**: reduces total shots for same accuracy
- **Benefit**: faster optimization

Noise-Aware Ansatz Design

- **Idea**: design circuits robust to noise
- **Method**: shorter circuits, native gates, symmetry-preserving
- **Effect**: less noise accumulation
- **Trade-off**: may reduce expressibility

Robust Cost Functions

- **Idea**: cost functions less sensitive to noise
- **Method**: local observables, symmetry-protected quantities
- **Effect**: more stable optimization
- **Example**: QAOA with local cost terms

Examples in Context

VQE on NISQ Hardware

- **Challenge**: decoherence limits circuit depth
- **Strategy**: shallow ansätze, error mitigation
- **Result**: approximate ground state energies

Noisy QAOA

- **Challenge**: gate errors accumulate over layers
- **Strategy**: optimize layer count, use ZNE
- **Result**: better approximation ratios

Quantum Machine Learning

- **Challenge**: training requires many circuit evaluations
- **Strategy**: noise-aware training, regularization
- **Result**: models robust to noise

Noise Models

Depolarizing Channel

- **Effect**: state $\rightarrow$ (1-p)ρ + p $\cdot$ I/d (mixed with maximally mixed state)
- **Parameter**: p (depolarizing probability)
- **Impact**: reduces purity, flattens landscape

Amplitude Damping

- **Effect**: $|1\rangle \rightarrow |0\rangle$ with probability γ
- **Parameter**: γ (damping rate)
- **Impact**: biases toward $|0\rangle$ states

Phase Damping

- **Effect**: loses phase coherence
- **Parameter**: λ (dephasing rate)
- **Impact**: reduces off-diagonal density matrix elements

Simulating Noisy VQE

Noise Simulation

- **Tool**: Qiskit, Cirq with noise models
- **Method**: add noise channels after each gate
- **Purpose**: predict performance on real hardware

Benchmarking

- **Metrics**: fidelity, energy error, convergence rate
- **Comparison**: ideal vs noisy, different error rates
- **Goal**: understand noise impact, test mitigation

Practical Considerations

Circuit Depth vs Noise

- **Trade-off**: deeper circuits → more expressive but more noise
- **Strategy**: find optimal depth for hardware
- **Rule of thumb**: keep depth < coherence time / gate time

Shot Budget

- **Trade-off**: more shots → better estimates but more time
- **Strategy**: adaptive allocation, prioritize important terms
- **Typical**: 1000-10000 shots per circuit

Calibration

- **Importance**: gate errors vary over time
- **Strategy**: regular calibration, use recent calibration data
- **Impact**: more accurate noise models, better mitigation

Practice Problems

1. How does decoherence typically affect ground state energy estimates?
2. Why might zero-noise extrapolation help with systematic errors but not shot noise?
3. What's the trade-off between circuit depth and noise in ansatz design?

Key Takeaways

- Noise transforms cost landscapes: rougher, biased, flattened
- Different noise types: decoherence, gate errors, measurement errors
- Error mitigation: ZNE, PEC, symmetry verification, readout correction
- Noise-aware strategies: adaptive shots, robust ansätze, local cost functions
- Trade-offs: expressibility vs noise, shots vs time, accuracy vs cost

Next Steps

Next, we'll put it all together and walk through the complete VQE pipeline from problem setup to solution.

Lesson 4.6: VQE Pipeline

Overview

Putting it all together: the complete VQE workflow from problem definition to solution, with practical considerations.

Learning Objectives

- Understand the full VQE pipeline end-to-end
- Recognize decision points and trade-offs at each stage
- Build intuition for debugging and improving VQE performance
- Apply VQE to real problems in chemistry, optimization, and beyond

Core Concepts

The VQE Pipeline

Stage 1: Problem Definition

1. **Define the problem**: What are you trying to solve?
2. **Formulate Hamiltonian**: $\hat{H}$ that encodes the problem
3. **Choose target**: Ground state? Excited state? Thermal state?

Stage 2: Ansatz Design

1. **Select ansatz type**: Hardware-efficient? Problem-inspired?
2. **Determine depth**: Balance expressibility and noise
3. **Initialize parameters**: Random? Classical guess? Layerwise?

Stage 3: Cost Function Setup

1. **Decompose Hamiltonian**: $\hat{H} = \Sigma_i h_i P_i$ (Pauli strings)
2. **Group measurements**: Commuting terms measured together
3. **Add regularization** (optional): Constraints, smoothing

Stage 4: Gradient Computation

1. **Choose method**: Parameter-shift? Finite differences?
2. **Compute gradients**: $\partial E/\partial \theta_i$ for each parameter
3. **Handle noise**: Error mitigation, adaptive shots

Stage 5: Optimization

1. **Select optimizer**: Gradient descent? Adam? BFGS?
2. **Set hyperparameters**: Learning rate, momentum, etc.
3. **Iterate**: Update θ, evaluate cost, compute gradients, repeat

Stage 6: Validation

1. **Check convergence**: Has $E(\theta)$ stabilized?
2. **Verify solution**: Does state have expected properties?
3. **Benchmark**: Compare to classical methods, exact solutions

Detailed Walkthrough

Example: H_2 Molecule VQE **Step 1: Problem Definition** - **Goal**: Find ground state energy of H_2 molecule - **Hamiltonian**: Electronic structure Hamiltonian - **Mapping**: Jordan-Wigner or Bravyi-Kitaev to qubits

Step 2: Hamiltonian Construction - **Molecular geometry**: H-H distance = 0.74 Å - **Basis set**: STO-3G (minimal basis) - **Result**: $\hat{H} = \Sigma_i h_i P_i$ (sum of Pauli strings) - **Example**: $\hat{H}$ = -1.05 · I + 0.18 · Z_0 - 0.48 · Z_1 + 0.17 · Z_0Z_1 + ...

Step 3: Ansatz Selection - **Choice**: UCC singles and doubles (UCCSD) - **Qubits**: 2 qubits (2 spin orbitals) - **Parameters**: 1 parameter (single excitation amplitude) - **Circuit**: $R_\gamma(\theta)$ on qubit 0, CNOT, $R_\gamma(-\theta)$ on qubit 1, CNOT

Step 4: Initial State - **Hartree-Fock**: $|01\rangle$ (one electron in each orbital) - **Preparation**: X gate on qubit 1

Step 5: Cost Function - $\mathbf{E(\theta) = \langle\psi(\theta)|\hat{H}|\psi(\theta)\rangle}$ - **Measurement**: Measure each Pauli string - **Shots**: 1000 per Pauli string

Step 6: Gradient Computation - **Method**: Parameter-shift rule - $\mathbf{E(\theta + \pi/2)}$ and $\mathbf{E(\theta - \pi/2)}$ - **Gradient**: $\partial E/\partial\theta = [E(\theta + \pi/2) - E(\theta - \pi/2)]/2$

Step 7: Optimization - **Optimizer**: Gradient descent - **Learning rate**: $\alpha = 0.1$ - **Update**: $\theta \leftarrow \theta - \alpha \cdot \partial E/\partial\theta$ - **Iterations**: ~20-50 iterations

Step 8: Result - **Converged energy**: E ≈ -1.137 Ha (Hartree) - **Exact**: -1.137 Ha - **Error**: < 0.001 Ha (chemical accuracy)

Decision Points and Trade-offs

Ansatz Depth

- **Shallow**: Less noise, less expressive
- **Deep**: More expressive, more noise
- **Decision**: Match to hardware coherence time

Shot Budget

- **Few shots**: Fast, noisy estimates
- **Many shots**: Slow, accurate estimates
- **Decision**: Adaptive allocation, prioritize important terms

Optimizer Choice

- **Gradient-free** (Nelder-Mead, COBYLA): No gradients needed, slower
- **Gradient-based** (Adam, BFGS): Faster, requires gradients
- **Decision**: Use gradients if available (parameter-shift)

Error Mitigation

- **None**: Fastest, biased results
- **ZNE**: Moderate cost, reduces bias
- **PEC**: High cost, unbiased
- **Decision**: Balance accuracy and runtime

Common Pitfalls and Solutions

Pitfall 1: Stuck in Local Minimum

- **Symptom**: Optimization stops improving, energy above expected

- **Solution**: Multiple random initializations, simulated annealing

Pitfall 2: Barren Plateau

- **Symptom**: Gradients vanish, no progress
- **Solution**: Shallower ansatz, problem-inspired structure, local cost

Pitfall 3: Noisy Gradients

- **Symptom**: Optimization oscillates, doesn't converge
- **Solution**: More shots, gradient filtering, smaller learning rate

Pitfall 4: Wrong Symmetry Sector

- **Symptom**: Solution violates known symmetries
- **Solution**: Symmetry-preserving ansatz, regularization

Pitfall 5: Measurement Overhead

- **Symptom**: Too many Pauli strings, too slow
- **Solution**: Group commuting terms, shadow tomography

Performance Metrics

Energy Accuracy

- **Chemical accuracy**: Error < 1.6 mHa (0.001 Ha)
- **Relative error**: |E_VQE - E_exact|/|E_exact|

Convergence Rate

- **Iterations to convergence**: Fewer is better
- **Wall-clock time**: Total runtime on hardware

Circuit Depth

- **Gate count**: Fewer gates $\rightarrow$ less noise
- **Depth**: Shorter circuits $\rightarrow$ more robust

Measurement Cost

- **Total shots**: Fewer shots $\rightarrow$ faster
- **Circuit evaluations**: Fewer evaluations $\rightarrow$ cheaper

Examples in Context

Quantum Chemistry

- **Problem**: Molecular ground state energies
- **Ansatz**: UCC (unitary coupled cluster)
- **Success**: H_2, LiH, BeH_2 within chemical accuracy

Combinatorial Optimization

- **Problem**: MaxCut, TSP, graph coloring
- **Ansatz**: QAOA (Quantum Approximate Optimization Algorithm)
- **Success**: Small instances, competitive with classical

Condensed Matter Physics

- **Problem**: Lattice models (Hubbard, Heisenberg)
- **Ansatz**: Hardware-efficient or symmetry-adapted
- **Success**: Phase diagrams, ground state properties

Machine Learning

- **Problem**: Classification, regression
- **Ansatz**: Data-encoding + variational layers
- **Success**: Proof-of-concept, exploring quantum advantage

Practical Implementation Checklist

Before Running

☐ Hamiltonian defined and decomposed
☐ Ansatz chosen and implemented
☐ Initial parameters set
☐ Optimizer configured
☐ Shot budget allocated
☐ Error mitigation strategy decided

During Optimization

☐ Monitor energy convergence
☐ Track gradient magnitudes
☐ Check for symmetry violations
☐ Adjust learning rate if needed
☐ Save intermediate results

After Convergence

☐ Verify solution properties
☐ Compare to benchmarks
☐ Analyze error sources
☐ Document results
☐ Consider improvements

Advanced Topics

Adaptive VQE

- **Idea**: Grow ansatz during optimization
- **Method**: Add layers when gradients plateau
- **Benefit**: Balance expressibility and depth

Subspace VQE

- **Idea**: Solve in reduced subspace
- **Method**: Exploit symmetries, active space
- **Benefit**: Fewer qubits, faster optimization

Excited State VQE

- **Idea**: Find excited states, not just ground
- **Method**: Orthogonality constraints, penalty terms

- **Benefit**: Access to spectroscopy, dynamics

Quantum Natural Gradient

- **Idea**: Use Fisher information metric
- **Method**: Compute F, solve $F \cdot \Delta\theta = -\nabla E$
- **Benefit**: Faster convergence, better geometry

Practice Problems

1. Design a VQE pipeline for finding the ground state of a 3-qubit Heisenberg model.
2. How would you debug a VQE run that's stuck at high energy?
3. What's the expected scaling of VQE runtime with number of qubits?

Key Takeaways

- VQE pipeline: problem → Hamiltonian → ansatz → cost → gradients → optimization → validation
- Key decisions: ansatz depth, shot budget, optimizer, error mitigation
- Common pitfalls: local minima, barren plateaus, noisy gradients, symmetry violations
- Success metrics: energy accuracy, convergence rate, circuit depth, measurement cost
- VQE is a flexible framework applicable to chemistry, optimization, physics, ML

Module 4 Complete

You now understand the full VQE pipeline and how to apply variational quantum algorithms. In Module 5, we'll sharpen your mathematical reading skills with practical clarity drills.