Lesson 4.3: Gradients

Mathematical Intuition · Module 4 — Optimization Regularization

Node 4.3

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**: $\partial \mathrm{E}/\partial\theta_i$ computed independently
- **Gradient**: $\nabla \mathrm{E} = [\partial \mathrm{E}/\partial\theta_1, \partial \mathrm{E}/\partial\theta_2, \ldots, \partial \mathrm{E}/\partial\theta_p]^T$
- **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: $\mathrm{O}(\varepsilon)$

Central Difference

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

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 $\hat{\mathrm{G}}$ 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

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

Step 2: Compute Shifted Costs

- For each θ_i: compute $E(\theta + \pi/2 \cdot e_i)$ and $E(\theta - \pi/2 \cdot e_i)$

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.
