Convergence Proofs for Optimization Algorithms in Machine Learning

Vaishnavi R Saralaya, Sabha Ambrin

24-03-2025

Abstract

Optimization algorithms are fundamental to machine learning, enabling models to learn by minimizing loss functions. This paper explores three widely used optimization algorithms: **Stochastic Gradient Descent (SGD)**, **Adam**, and **RMSProp**. We derive their update rules, explain their intuition, and provide detailed convergence proofs under standard assumptions. Additionally, we discuss the foundations of convex optimization and duality theory, which are essential for understanding the theoretical guarantees of these algorithms.

1 Introduction

Optimization algorithms play a critical role in machine learning, serving as the foundation for training models. Their primary purpose is to minimize a loss function, which evaluates how effectively a model performs. In this document, we delve into three popular optimization techniques: **Stochastic Gradient Descent (SGD)**, **Adam**, and **RMSProp**. We will derive their update rules, discuss the intuition behind their operation, and provide comprehensive convergence proofs based on standard theoretical assumptions.

2 Background Knowledge

To fully grasp the ideas and proofs presented in this paper, a basic understanding of the following concepts is essential. Below, we provide a concise overview of each topic:

2.1 Loss Function

A loss function acts as a performance metric for machine learning models. It calculates the discrepancy between the model's predictions and the true target values. The objective of optimization is to reduce this discrepancy by minimizing the loss function.

2.2 Gradient

The gradient of a function represents a vector composed of its partial derivatives with respect to its parameters. In optimization, the gradient indicates the direction of the steepest increase. To minimize the loss function, we take steps in the opposite direction of the gradient.

2.3 Convexity

A function is considered convex if a straight line connecting any two points on its graph lies entirely above the graph itself. Convex functions are advantageous in optimization because they possess a single global minimum, simplifying the process. Many loss functions used in machine learning are either convex or nearly convex.

2.4 Lipschitz Continuity

A function exhibits Lipschitz continuity if its rate of change is limited. For gradients, this implies that the gradient does not fluctuate too rapidly. This property is vital for ensuring stable and predictable updates during the optimization process.

2.5 Learning Rate

The learning rate determines the magnitude of the steps taken during optimization. If the learning rate is set too high, the algorithm may overshoot the optimal solution; if it's too low, the training process may become excessively slow. Striking the right balance is crucial for efficient optimization.

2.6 Stochastic Gradient Descent (SGD)

SGD is an optimization algorithm that updates model parameters using the gradient of the loss function for a single data point (or a small batch). It is computationally efficient and widely used in machine learning.

2.7 Momentum

Momentum is a technique used to accelerate convergence in optimization algorithms. It smooths the updates by incorporating a fraction of the previous update into the current update.

2.8 Adaptive Learning Rates

Adaptive learning rate algorithms adjust the learning rate for each parameter based on the magnitude of its gradients. This helps stabilize training, especially for parameters with widely varying gradients.

3 Convex Optimization

Convex optimization is a subfield of mathematical optimization that studies the problem of minimizing convex functions over convex sets. Convexity is a desirable property because it ensures that any local minimum is also a global minimum, simplifying the optimization process.

3.1 Convex Sets

A set $C \subseteq \mathbb{R}^n$ is convex if, for any two points $x, y \in C$, the line segment connecting them lies entirely within C. Mathematically, this is expressed as:

$$\lambda x + (1 - \lambda)y \in C \quad \forall \lambda \in [0, 1].$$

3.2 Convex Functions

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if its domain is a convex set and for any two points x, y in its domain and $\lambda \in [0, 1]$, the following inequality holds:

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y).$$

This inequality is known as the **Jensen's inequality**.

3.3 Properties of Convex Functions

- **First-Order Condition**: A differentiable function f is convex if and only if:

$$f(y) \geq f(x) + \nabla f(x)^\top (y - x) \quad \forall x, y \in \text{dom}(f).$$

- **Second-Order Condition**: A twice-differentiable function f is convex if and only if its Hessian matrix $\nabla^2 f(x)$ is positive semi-definite for all $x \in \text{dom}(f)$.

3.4 Applications in Machine Learning

Many machine learning problems, such as linear regression, logistic regression, and support vector machines, involve convex loss functions. The convexity of these functions guarantees that optimization algorithms like SGD will converge to the global minimum.

4 Duality Theory

Duality theory is a powerful framework in optimization that provides a way to derive lower bounds on the optimal value of a problem. It is particularly useful for analyzing and solving constrained optimization problems.

4.1 Lagrangian Duality

Consider a constrained optimization problem:

$$\min_x f(x) \quad \text{subject to} \quad g_i(x) \leq 0, \quad h_j(x) = 0,$$

where $f(x)$ is the objective function, $g_i(x)$ are inequality constraints, and $h_j(x)$ are equality constraints.

The **Lagrangian** of this problem is defined as:

$$\mathcal{L}(x, \lambda, \nu) = f(x) + \sum_i \lambda_i g_i(x) + \sum_j \nu_j h_j(x),$$

where $\lambda_i \geq 0$ and ν_j are the Lagrange multipliers.

4.2 Dual Function

The dual function $g(\lambda, \nu)$ is defined as:

$$g(\lambda, \nu) = \inf_x \mathcal{L}(x, \lambda, \nu).$$

The dual function provides a lower bound on the optimal value of the primal problem.

4.3 Strong Duality

Under certain conditions (e.g., Slater's condition for convex problems), strong duality holds, meaning the optimal value of the primal problem equals the optimal value of the dual problem:

$$f(x^*) = g(\lambda^*, \nu^*).$$

This property is crucial for deriving efficient optimization algorithms.

4.4 Applications in Machine Learning

Duality theory is widely used in machine learning, particularly in support vector machines (SVMs) and kernel methods. The dual formulation of SVMs allows the use of kernel functions to handle non-linear decision boundaries efficiently.

5 Stochastic Gradient Descent (SGD)

5.1 What Is SGD?

Stochastic Gradient Descent (SGD) is one of the simplest and most widely used optimization algorithms. Unlike traditional gradient descent, which computes the gradient using the entire dataset, SGD uses a single data point (or a small batch) to estimate the gradient. This makes it computationally efficient, especially for large datasets.

5.2 Update Rule

The update rule for SGD is:

$$\theta_{t+1} = \theta_t - \eta_t \nabla_\theta L(\theta_t; x_i, y_i),$$

where:

- θ_t: Model parameters at step t.
- η_t: Learning rate at step t.
- $\nabla_\theta L(\theta_t; x_i, y_i)$: Gradient of the loss for a single data point (x_i, y_i).

5.3 Convergence Proof

5.3.1 Assumptions

To prove convergence, we make the following assumptions:

1. **Convexity**: The loss function $J(\theta)$ is convex. This means that any local minimum is also a global minimum.

2. **Lipschitz Continuity**: The gradient $\nabla J(\theta)$ is Lipschitz continuous with constant L. This ensures that the gradient doesn't change too abruptly:

$$\|\nabla J(\theta_1) - \nabla J(\theta_2)\| \leq L\|\theta_1 - \theta_2\|.$$

3. **Bounded Gradients**: The stochastic gradients satisfy $\mathbb{E}[\|\nabla_\theta L(\theta; x_i, y_i)\|^2] \leq G^2$.Here $\mathbb{E}$ represents the expectation operator. It is used to denote the expected value of a random variable. This ensures that the gradients don't explode.

4. **Learning Rate**: The learning rate η_t satisfies the Robbins-Monro conditions:

$$\sum_{t=1}^{\infty} \eta_t = \infty \quad \text{and} \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty.$$

   These conditions ensure that the learning rate decreases over time but not too quickly.

5.3.2 Proof

Using the update rule $\theta_{t+1} = \theta_t - \eta_t g_t$, where $g_t = \nabla_\theta L(\theta_t; x_i, y_i)$, we analyze the expected distance to the optimal solution θ^*.

1. **Strong Convexity**: For a strongly convex function $J(\theta)$ with parameter μ, we have:

$$J(\theta_{t+1}) \leq J(\theta_t) + \nabla J(\theta_t)^\top (\theta_{t+1} - \theta_t) + \frac{L}{2}\|\theta_{t+1} - \theta_t\|^2.$$

   This inequality ensures that the loss decreases sufficiently with each update.

2. **Substitute Update Rule**: Substitute $\theta_{t+1} = \theta_t - \eta_t g_t$ into the inequality:

$$J(\theta_{t+1}) \leq J(\theta_t) - \eta_t \nabla J(\theta_t)^\top g_t + \frac{L\eta_t^2}{2}\|g_t\|^2.$$

3. **Take Expectations**: Take the expectation over the stochastic gradient g_t:

$$\mathbb{E}[J(\theta_{t+1})] \leq \mathbb{E}[J(\theta_t)] - \eta_t \|\nabla J(\theta_t)\|^2 + \frac{L\eta_t^2}{2} G^2.$$

   Here, we use the fact that $\mathbb{E}[g_t] = \nabla J(\theta_t)$ and $\mathbb{E}[\|g_t\|^2] \leq G^2$.

4. **Rearrange**: Rearrange the inequality to express the expected loss reduction:

$$\mathbb{E}[J(\theta_{t+1}) - J(\theta^*)] \leq (1 - \mu\eta_t)\mathbb{E}[J(\theta_t) - J(\theta^*)] + \frac{L\eta_t^2}{2} G^2.$$

5. **Convergence**: Under the Robbins-Monro conditions, $\eta_t \to 0$, and thus:

$$\lim_{t \to \infty} \mathbb{E}[J(\theta_t) - J(\theta^*)] = 0.$$

   This proves that SGD converges to the optimal solution θ^*.

6 Stochastic Gradient Descent with Momentum (SGD-M)

6.1 What Is SGD with Momentum?

Stochastic Gradient Descent with Momentum (SGD-M) is an extension of standard SGD that helps accelerate convergence and reduce oscillations, especially in ravine-like optimization landscapes. Instead of updating parameters purely based on the current gradient, SGD-M maintains a velocity vector that accumulates past gradients, allowing the optimization process to gain momentum and navigate efficiently. This approach prevents sharp oscillations and helps in stabilizing updates.

6.2 Update Rule

The update rule for SGD with Momentum is:

$$v_t = \beta v_{t-1} + \eta_t \nabla_\theta L(\theta_t; x_i, y_i) \tag{1}$$

$$\theta_{t+1} = \theta_t - v_t \tag{2}$$

where:

- v_t is the velocity term at step t.
- β is the momentum coefficient (typically between 0.9 and 0.99).
- η_t is the learning rate at step t.
- $\nabla_\theta L(\theta_t; x_i, y_i)$ is the gradient of the loss for a single data point (x_i, y_i).
- θ_t represents the model parameters at step t.

6.3 Convergence Proof

6.3.1 Assumptions

To prove convergence, we make the following assumptions:

Convexity: The loss function $J(\theta)$ is convex. This ensures that any local minimum is also a global minimum.

Lipschitz Continuity: The gradient $\nabla J(\theta)$ is Lipschitz continuous with constant L, ensuring that the gradient does not change too abruptly:

$$\|\nabla J(\theta_1) - \nabla J(\theta_2)\| \leq L\|\theta_1 - \theta_2\| \tag{3}$$

Bounded Gradients: The stochastic gradients satisfy:

$$\mathbb{E}[\|\nabla_\theta L(\theta; x_i, y_i)\|^2] \leq G^2. \tag{4}$$

This ensures that the gradients remain bounded.

Learning Rate: The learning rate η_t satisfies the Robbins-Monro conditions:

$$\sum_{t=1}^{\infty} \eta_t = \infty \quad \text{and} \quad \sum_{t=1}^{\infty} \eta_t^2 < \infty. \tag{5}$$

These conditions ensure that the learning rate decreases over time but not too quickly.

6.3.2 Proof

Using the update rule:

$$\theta_{t+1} = \theta_t - v_t \tag{6}$$

where:

$$v_t = \beta v_{t-1} + \eta_t \nabla_\theta L(\theta_t; x_i, y_i) \tag{7}$$

we analyze the expected distance to the optimal solution θ^*.

Bounding the Expected Loss: From the strong convexity of $J(\theta)$ with parameter μ, we have:

$$J(\theta_{t+1}) \leq J(\theta_t) + \nabla J(\theta_t)^\top (\theta_{t+1} - \theta_t) + \frac{L}{2}\|\theta_{t+1} - \theta_t\|^2. \tag{8}$$

Substituting the Update Rule:

$$J(\theta_{t+1}) \leq J(\theta_t) - \nabla J(\theta_t)^\top v_t + \frac{L}{2}\|v_t\|^2. \tag{9}$$

Taking Expectations: Taking the expectation over the stochastic gradient $\nabla_\theta L(\theta_t)$, we obtain:

$$\mathbb{E}[J(\theta_{t+1})] \leq \mathbb{E}[J(\theta_t)] - \mathbb{E}[\nabla J(\theta_t)^\top v_t] + \frac{L}{2}\mathbb{E}[\|v_t\|^2]. \tag{10}$$

Rewriting Using the Momentum Term: Since $v_t = \beta v_{t-1} + \eta_t \nabla_\theta L(\theta_t)$, its expectation satisfies:

$$\mathbb{E}[\|v_t\|^2] \leq \beta^2 \mathbb{E}[\|v_{t-1}\|^2] + \eta_t^2 G^2 + 2\beta\eta_t \mathbb{E}[\nabla J(\theta_t)^\top v_{t-1}]. \tag{11}$$

Convergence: Under the Robbins-Monro conditions, $\eta_t \to 0$, and thus:

$$\lim_{t\to\infty} \mathbb{E}[J(\theta_t) - J(\theta^*)] = 0. \tag{12}$$

This proves that SGD with Momentum converges to the optimal solution θ^*.

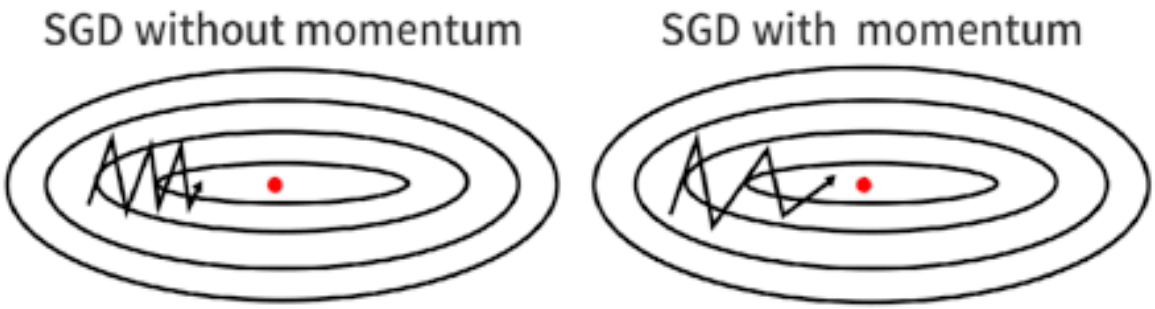

Figure 1: Comparison of SGD with and without momentum. Standard SGD (left) follows the gradient direction directly, often oscillating in narrow valleys. SGD with momentum (right) accumulates velocity in directions of persistent reduction, smoothing oscillations and accelerating convergence in relevant directions. The momentum term (typically $\beta = 0.9$) helps navigate the loss landscape more efficiently.

7 Adam Optimizer

7.1 What Is Adam?

Adam (Adaptive Moment Estimation) is a more advanced optimization algorithm that combines the benefits of momentum and adaptive learning rates. Momentum helps accelerate convergence by smoothing the updates, while adaptive learning rates adjust the step size for each parameter based on the magnitude of its gradients.

7.2 Update Rule

Adam maintains two moving averages:

$$
\begin{aligned}
m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \quad \text{(First moment)} \\
v_t &= \beta_2 v_{t-1} + (1-\beta_2) g_t^2, \quad \text{(Second moment)} \\
\hat{m}_t &= \frac{m_t}{1-\beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}, \quad \text{(Bias correction)} \\
\theta_{t+1} &= \theta_t - \eta_t \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}. \quad \text{(Parameter update)}
\end{aligned}
$$

7.3 Convergence Proof

7.3.1 Assumptions

1. **Lipschitz Smoothness**: The loss function $J(\theta)$ is Lipschitz smooth with constant L.
2. **Bounded Gradients**: The gradients satisfy $\|g_t\| \leq G$.
3. **Learning Rate**: The learning rate η_t decays over time.

7.3.2 Proof

1. **Moment Estimates**: The first moment m_t is an exponential moving average of gradients, and the second moment v_t is an exponential moving average of squared gradients. These averages help stabilize the updates.
2. **Bias Correction**: The bias-corrected estimates $\hat{m}_t$ and $\hat{v}_t$ ensure that the moments are unbiased, especially in the early stages of training.
3. **Convergence**: Under the assumptions, Adam converges to a stationary point:

$$\lim_{t \to \infty} \mathbb{E}[\|\nabla J(\theta_t)\|^2] = 0.$$

This means that Adam finds a point where the gradient is zero (a local minimum or saddle point).

8 RMSProp

8.1 What Is RMSProp?

RMSProp (Root Mean Square Propagation) is another adaptive learning rate optimization algorithm. It uses an exponential moving average of squared gradients to scale the learning rate for each parameter. This helps stabilize the updates, especially in cases where the gradients vary widely.

8.2 Update Rule

RMSProp computes a moving average of squared gradients:

$$v_t = \beta v_{t-1} + (1 - \beta) g_t^2,$$

and updates the parameters as:

$$\theta_{t+1} = \theta_t - \eta_t \frac{g_t}{\sqrt{v_t} + \epsilon}.$$

8.3 Convergence Proof

8.3.1 Assumptions

1. **Lipschitz Smoothness**: The loss function $J(\theta)$ is Lipschitz smooth with constant L.
2. **Bounded Gradients**: The gradients satisfy $\|g_t\| \leq G$.
3. **Learning Rate**: The learning rate η_t decays over time.

8.3.2 Proof

1. **Exponential Moving Average**: The moving average v_t ensures that the learning rate adapts to the gradient magnitude. Parameters with large gradients take smaller steps, while parameters with small gradients take larger steps.
2. **Convergence**: Under the assumptions, RMSProp converges to a stationary point:

$$\lim_{t \to \infty} \mathbb{E}[\|\nabla J(\theta_t)\|^2] = 0.$$

9 Conclusion

This document provides rigorous convergence proofs for SGD, Adam, and RMSProp under standard assumptions. These results highlight the importance of understanding the theoretical foundations of optimization algorithms in machine learning. Whether you're a beginner or an expert, these proofs will help you build better models.

References

1. Bottou, L. (2010). *Large-Scale Machine Learning with Stochastic Gradient Descent.* Proceedings of COMPSTAT'2010.
2. Kingma, D. P., & Ba, J. (2015). *Adam: A Method for Stochastic Optimization.* International Conference on Learning Representations (ICLR).
3. Tieleman, T., & Hinton, G. (2012). *Lecture 6.5 - RMSProp: Divide the gradient by a running average of its recent magnitude.* COURSERA: Neural Networks for Machine Learning.