# Gradient Descent Algorithms: An In-depth Analysis

Your Name

March 20, 2025

# Overview

- Introduction to Gradient Descent
- Types of Gradient Descent
- Mathematical Formulations
- Comparison of Different Variants
- Practical Considerations

# Introduction to Gradient Descent

- Gradient Descent is an optimization algorithm used to minimize a function by iteratively moving in the direction of the negative gradient.
- Given a function $f(\theta)$, the update rule is:

$$\theta_{t+1} = \theta_t - \eta \nabla f(\theta_t)$$

where $\eta$ is the learning rate.

# Batch Gradient Descent

- Uses the entire dataset to compute the gradient.
- Update rule:

$$\theta_{t+1} = \theta_t - \eta \frac{1}{m} \sum_{i=1}^{m} \nabla f_i(\theta_t)$$

- Pros: Converges smoothly.
- Cons: Computationally expensive for large datasets.

# Stochastic Gradient Descent (SGD)

- Uses one random sample at each step.
- Update rule:
$$\theta_{t+1} = \theta_t - \eta \nabla f_i(\theta_t)$$
- Pros: Faster updates, good for large datasets.
- Cons: More variance in updates.

# Mini-batch Gradient Descent

- Uses a small batch of data at each step.
- Update rule:

$$\theta_{t+1} = \theta_t - \eta \frac{1}{B} \sum_{i=1}^{B} \nabla f_i(\theta_t)$$

- Pros: Balance between batch and stochastic methods.
- Cons: Requires careful tuning of batch size.

# Momentum-based Gradient Descent

- Uses momentum to accelerate convergence.
- Update rule:

$$\begin{aligned} v_t &= \beta v_{t-1} + (1-\beta)\nabla f(\theta_t) \\ \theta_{t+1} &= \theta_t - \eta v_t \end{aligned}$$

- Pros: Reduces oscillations and speeds up learning.

# Adaptive Methods: Adagrad, RMSprop, Adam

- Adagrad: Adapts learning rates element-wise.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{G_t + \epsilon}} \nabla f(\theta_t)$$

- RMSprop: Uses exponentially weighted moving average of squared gradients.
- Adam: Combines momentum and RMSprop.

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) \nabla f(\theta_t) \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) \nabla f(\theta_t)^2 \\ \hat{m}_t &= \frac{m_t}{1 - \beta_1^t}, \quad \hat{v}_t = \frac{v_t}{1 - \beta_2^t} \\ \theta_{t+1} &= \theta_t - \frac{\eta \hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \end{aligned}$$

# Comparison of Algorithms

| Method | Speed | Convergence Stability | Scalability |
|---|---|---|---|
| Batch GD | Slow | Stable | Low |
| SGD | Fast | Unstable | High |
| Mini-batch GD | Medium | More stable | High |
| Momentum | Faster | Stable | Medium |
| Adam | Fastest | Very stable | High |

# Conclusion

- Choice of algorithm depends on dataset size, computational power, and convergence requirements.
- Adaptive methods like Adam are often preferred.
- Proper tuning of hyperparameters is essential.