Assignment 5: Optimization Methods in Machine Learning

Math4AI: Calculus & Optimization

Fatima Alibabayeva

fatime.elibabayeva25@aiacademy.az

National AI Academy

December 18, 2025

Contents

1 Introduction

In machine learning, the goal is to minimize a loss function that represents the error between model predictions and ground truth. This process is known as optimization. While simple models can sometimes be solved analytically, most modern machine learning tasks require iterative optimization algorithms. This report implements and compares first-order gradient descent variants, adaptive momentum methods, and second-order methods like Newton's Method within the context of linear regression.

2 Mathematical Foundation

2.1 Mean Squared Error (MSE) Surface

For our model $y = mx + b$, we optimize the parameter vector $\theta = [m, b]^T$. The MSE loss is defined as:

$$L(m, b) = \frac{1}{N} \sum_{i=1}^{N} (y_i - (mx_i + b))^2 \tag{1}$$

Since this is a quadratic function, the loss surface forms a convex "bowl" (paraboloid), which guarantees that gradient-based methods will converge to the global minimum.

2.2 Gradient and Curvature

The gradient ∇L tells us which way is "uphill," so we move in the opposite direction.

$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial m} \\ \frac{\partial L}{\partial b} \end{bmatrix} = \begin{bmatrix} -\frac{2}{N} \sum (y_i - \hat{y}_i) x_i \\ -\frac{2}{N} \sum (y_i - \hat{y}_i) \end{bmatrix} \tag{2}$$

The Hessian matrix H provides the curvature information:

$$H = \begin{bmatrix} \frac{2}{N} \sum x_i^2 & \frac{2}{N} \sum x_i \\ \frac{2}{N} \sum x_i & 2 \end{bmatrix} \tag{3}$$

3 Detailed Analysis of Gradient Descent Variants

3.1 Batch Gradient Descent (BGD)

BGD calculates the gradient using every single sample in the dataset before making one update.

- **Pros:** Stable convergence and guaranteed to reach the minimum for convex surfaces.
- **Cons:** Extremely slow on large datasets because it requires a full pass through the memory for a single step.

3.2 Stochastic Gradient Descent (SGD)

SGD updates the parameters for *each* individual training example.

- **Pros:** Very fast and can be used for online learning. The "noise" in the updates can help the model jump out of shallow local minima in more complex non-convex landscapes.
- **Cons:** The loss does not settle down smoothly; it "wanders" around the minimum due to the high variance of single-sample gradients.

3.3 Mini-Batch Gradient Descent

This is the industry standard. It splits the data into small batches (e.g., 16 or 32).

- **Justification:** It balances the computational efficiency of vectorization (matrix operations) with the stochastic nature of SGD. It reduces the variance of the gradient updates compared to SGD, leading to more stable convergence.

4 Adaptive and Second-Order Methods

4.1 Momentum and Adam

Standard GD often gets stuck in "ravines" where the surface is much steeper in one dimension than another.

- **Momentum:** Introduces a velocity term β that accumulates past gradients, helping the optimizer "accelerate" down the right path.
- **Adam:** Adaptive Moment Estimation adjusts the learning rate individually for each parameter. It is widely considered the most robust optimizer for deep learning.

4.2 Newton's Method (Second-Order)

Newton's method uses a second-order Taylor approximation. By multiplying the gradient by the inverse Hessian (H^{-1}), it accounts for the curvature.

$$\theta_{t+1} = \theta_t - H^{-1}\nabla L(\theta_t) \tag{4}$$

In our implementation, since the MSE loss is exactly quadratic, Newton's method finds the global minimum in effectively one major iteration, showing a near-vertical convergence on the loss plot.

5 Verification Results

Below is the visual comparison of the convergence rates for all implemented optimizers.

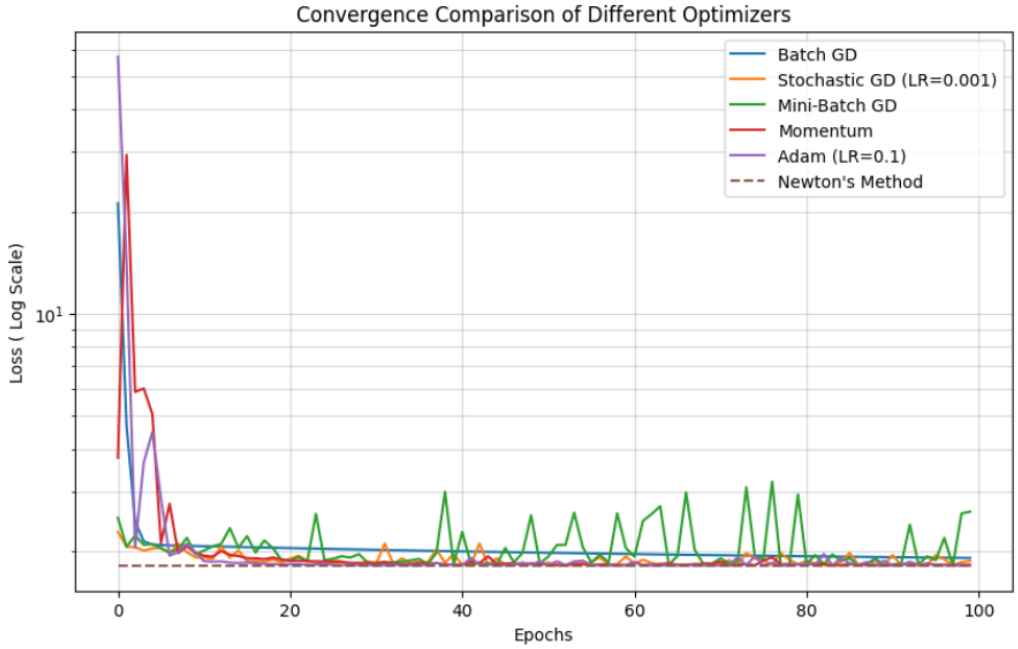

Figure 1: Loss Convergence Comparison (Logarithmic Scale)

6 Conclusion

This assignment verified that while Batch GD is mathematically straightforward, it is often inefficient. Adaptive methods like Adam provide a significant speedup by scaling learning rates. However, for low-dimensional quadratic problems, Newton's Method is the superior choice as it utilizes the full curvature information of the Hessian to reach the optimum instantly.