Stochastic Gradient Descent Optimization Methods

Duaa Alshareif, Leonard Sanya and Raham Jamal
African Masters for Machine Intelligence (AMMI)
dalshareif@aimsammi.org, lsanya@aimsammi.org
rsaleem@aimsammi.org

1 Introduction

Stochastic Gradient Descent (SGD) is a fundamental optimization algorithm widely used for training machine learning models, especially with large-scale datasets. By updating model parameters iteratively using a single data point or a small mini-batch, SGD significantly reduces computational overhead and enhances scalability. This project implements various SGD techniques to improve convergence and performance, including constant and shrinking step sizes, with and without replacement, averaging, and momentum. The effectiveness of these techniques is assessed on simulated datasets for the linear regression and logistic regression problems, focusing on convergence speed, stability, and solution accuracy.

2 Experimental Setup

To evaluate model performance, synthetic datasets were generated to replicate real-world conditions, providing a controlled yet challenging testing environment.

2.1 Data Generation for Linear Regression

A dataset with 50 features and 1000 samples was created. The true coefficients followed an exponential decay formula to establish a realistic relationship between features and the target variable. A correlation value of 0.1 introduced a mild interdependency among features.

2.2 Data Generation for Logistic Regression

A similar approach was applied to the logistic regression model, but with a higher correlation value of 0.7 to create a more complex scenario with stronger feature interdependencies.

2.3 Model Selection and Regularization

Ridge Regression (a regularized form of Linear Regression) and regularized Logistic Regression (LogReg) were used to prevent overfitting. Regularization controlled model complexity and enhanced generalization on unseen data.

2.4 Gradient Checking

Gradient checking was performed to ensure the accuracy of gradient computations by comparing analytical gradients with numerical approximations. This validation was crucial for reliable optimization.

2.5 Optimization with LBFGS

The LBFGS (Limited-memory Broyden-Fletcher-Goldfarb-Shanno) optimization algorithm was employed to achieve precise solutions. LBFGS efficiently minimized the loss function, serving as a benchmark to compare the performance of other optimization methods such as SGD and regular Gradient Descent (GD).

3 Experiments

3.1 SGD with Constant Vs. Shrinking Step Sizes

To explore the impact of different step size strategies, we first implemented SGD with a constant step size, which remained the same throughout all the iterations:

$$\text{step size} = \frac{1}{2L_{\max}} \tag{1}$$

This can provide rapid initial progress but may cause oscillations as the algorithm approaches the minimum, potentially slowing down convergence.

Next, we implemented a shrinking step size, which decreases over time according to the equation:

$$\text{step at iteration (t)} = \frac{1}{2L_{\max} + t} \tag{2}$$

This approach aims to combine rapid initial progress with stable and precise convergence in the later stages.

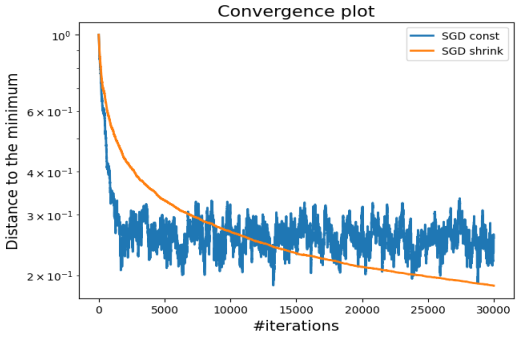

Figure 1: SGD with Constant & Shrinking distance to the Minimum vs. Iterations

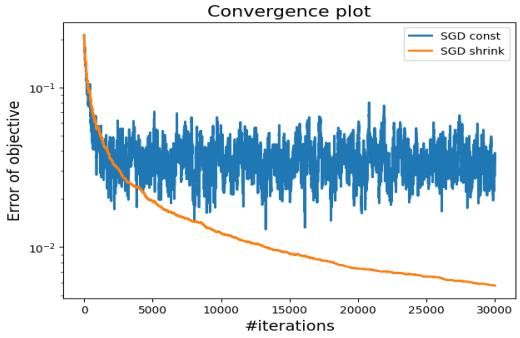

Figure 2: SGD with Constant & Shrinking loss Function vs. Iterations

From Figure 1 and Figure 2, we observe that SGD with a constant step size initially converges faster, as indicated by the steeper decline in both the distance to the minimum and the error of the objective within the first few thousand iterations. However, this method shows significant fluctuations and does not achieve as low a final error as the shrinking step size method. In contrast, SGD with a shrinking step size converges slightly slow at the beginning but ultimately achieves a lower final error and more stable convergence, demonstrating its effectiveness in achieving a more reliable optimization process.

3.2 SGD with switch to shrinking stepsizes

Switching to shrinking step sizes is an important strategy in the optimization process to ensure effective convergence. Initially, a larger constant stepsize is used to quickly reduce the loss, defined as:

$$\gamma_t = \frac{1}{2L_{\max}} \quad \text{for } t \leq 4\lceil\kappa\rceil \tag{3}$$

As the process nears the optimal solution, the stepsize shrinks to:

$$\gamma_t = \frac{2t+1}{(t+1)^2\mu} \quad \text{for } t > 4\lceil\kappa\rceil \tag{4}$$

The graphs compare the performance of two SGD variants: SGD with switching (SGD switch) and SGD with shrinking stepsizes (SGD shrink).

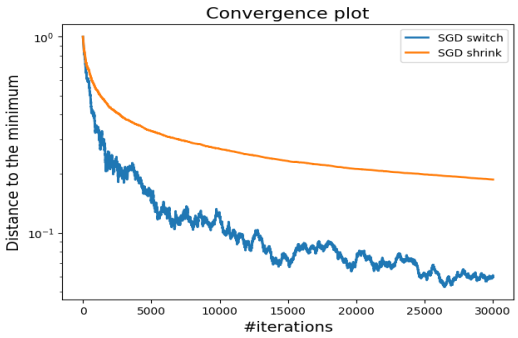

Figure 3: SGD switch to shrinking distance to the Minimum vs. Iterations

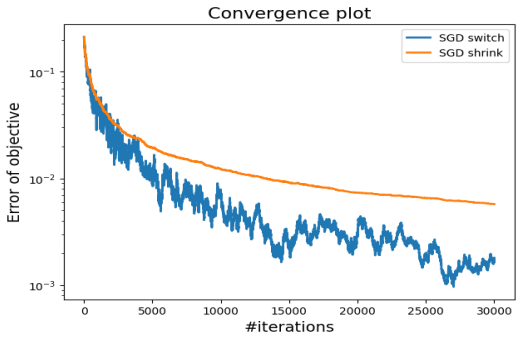

Figure 4: SGD switch to shrinking loss Function vs. Iterations

From Figure 3 and Figure 4, we observe that SGD with Constant Stepsizes is faster initially due to the large, unchanged stepsize that allows for rapid progress early on.In contrast, SGD with shrinking stepsizes starts slower but reaches a better solution in the long term due to its ability to fine-tune the convergence and reduce variability as the stepsize diminishes.

3.3 Stochastic gradient descent with averaging

Starting averaging only in the last n iterations (in this case, the last 25%) can indeed help stabilize the convergence process. This approach combines the benefits of rapid initial learning, achieved through larger step sizes that quickly reduce the loss, with the advantages of averaging in the later iterations, which smooths out fluctuations and promotes stability. By focusing on the most recent updates, this method allows the algorithm to capture the trends of the final stages of optimization, leading to a more robust and reliable solution.

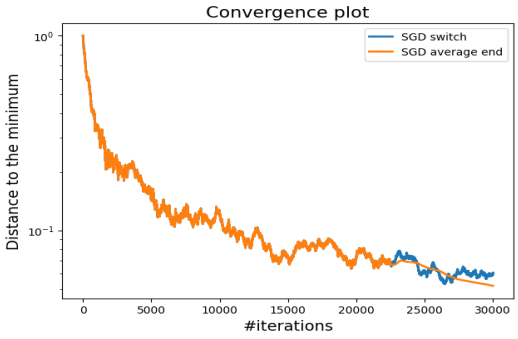

Figure 5: SGD with average distance to the Minimum vs. Iterations

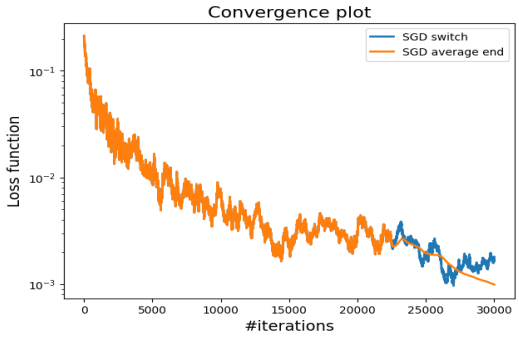

Figure 6: SGD with average loss Function vs. Iterations

From the graph 5 6it's clear that SGD with shrinking step sizes can get to a lower final loss more quickly, thanks to its larger step sizes early on, which help cut down the loss fast. On the other hand, SGD with averaging provides a steadier and smoother convergence, which can be better for generalization because it reduces the variability in updates.

3.4 SGD with momentum (SGDm) and Gradient Descent

We compared the model performance using the SGD with momentum, SGD with switch to shrinking stepsizes and the gradient descent. SGD with momentum aims at improving the efficiency and stability of the optimization process by accelerating convergence and reducing oscillations. It achieves this by accumulating a velocity vector that smooths out the updates and helps the optimizer move consistently towards the minimum, even in the presence of noisy gradients.

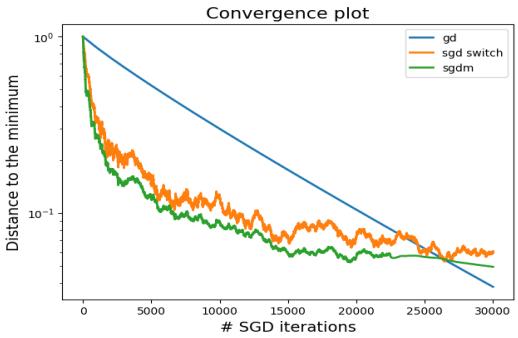

Figure 7: SGDm and GD distance to the Minimum vs. Iterations

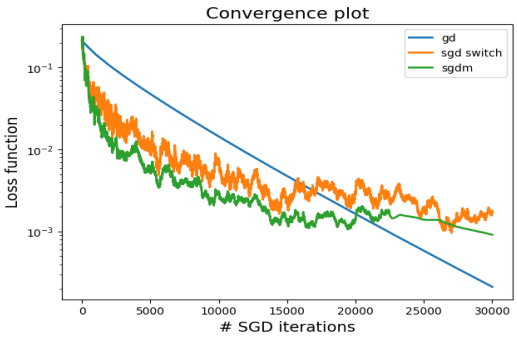

Figure 8: SGDm and GD loss function vs. Iterations

From Figures 7 and 8, we observe that SGD with momentum (SGDm) converges faster than standard SGD with shrinking step sizes. On the other hand, while the gradient descent method takes more iterations to converge, it ultimately reaches the optimal solution more effectively than the SGD methods. This is evident in the plot for the loss function, where SGDm takes 15,000 iterations to achieve an error below 10^{-2}. In contrast, Gradient Descent requires around 23,000 iterations to reach a similar error.

3.5 SGD without replacement

Using indices without replacement in SGD implementation encourages diversity in the learning process by ensuring that each sample is used only once per epoch. This approach is especially helpful for smaller datasets, as it can reduce the risk of overfitting and lead to better generalization. Additionally, it often results in faster convergence since the model learns more effectively from a varied set of samples.

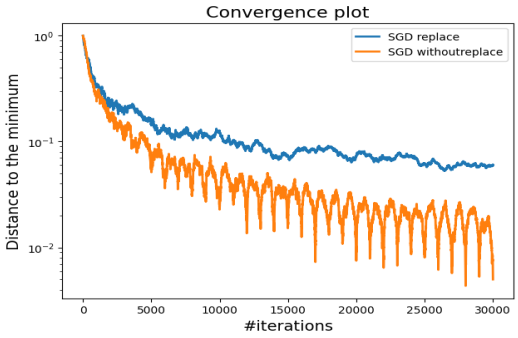

Figure 9: SGD without replacement distance to the Minimum vs. Iterations

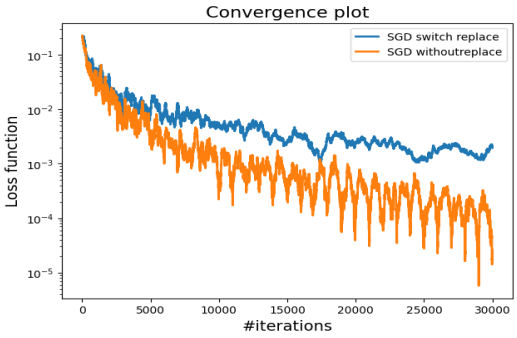

Figure 10: SGD without replacement loss Function vs. Iterations

From Figures 9 and 10 we observe that SGD without replacemen faster initial decrease in distance to the minimum and maintains a smoother, more stable convergence path. It achieves a lower distance to the minimum, demonstrating better and more consistent performance over time.

3.6 Comparing SGD and GD using the ground truth

In this last experiment, a comparison was made between SGD (without replacement and with averaging at the end), SGD (without replacement with decreasing step size and no averaging), and Gradient Descent (GD) using the true model parameters, which are the ground truth coefficients of the model.

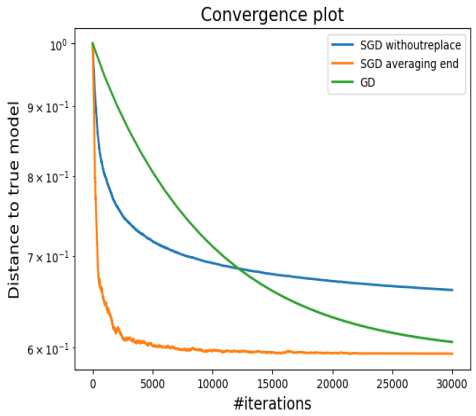

Figure 11: SGD without replace, SGD averaging-at-end, and GD distance to the True Model Comparison

From the convergence Figure 11 we notice the SGD without replacement initially converges rapidly but shows slower long-term progress, indicating instability. In contrast, SGD with averaging at the end demonstrates the fastest and most stable convergence, effectively reducing the distance to the true model. While GD converging more slowly and steadily, ultimately achieves a consistent and reliable solution.

4 Conlusion

In this project, we implemented various SGD techniques to improve convergence and performance, including constant and shrinking step sizes, with and without replacement, averaging, and momentum. We observed that SGD techniques significantly impact model optimization: shrinking step sizes outperform constant step sizes by achieving lower final errors and more stable convergence; combining initial constant step sizes with later shrinking ones improves precision and reduces variability, averaging in the final iterations stabilizes convergence and smooths out fluctuations, SGD with momentum accelerates convergence and reduces oscillations, outperforming standard SGD and traditional GD, and using indices without replacement enhances generalization and speeds up convergence, especially for smaller datasets. Overall, these methods improve training efficiency and model performance by balancing rapid initial progress with precise long-term convergence.