SDM - Assignment 3, Question 1

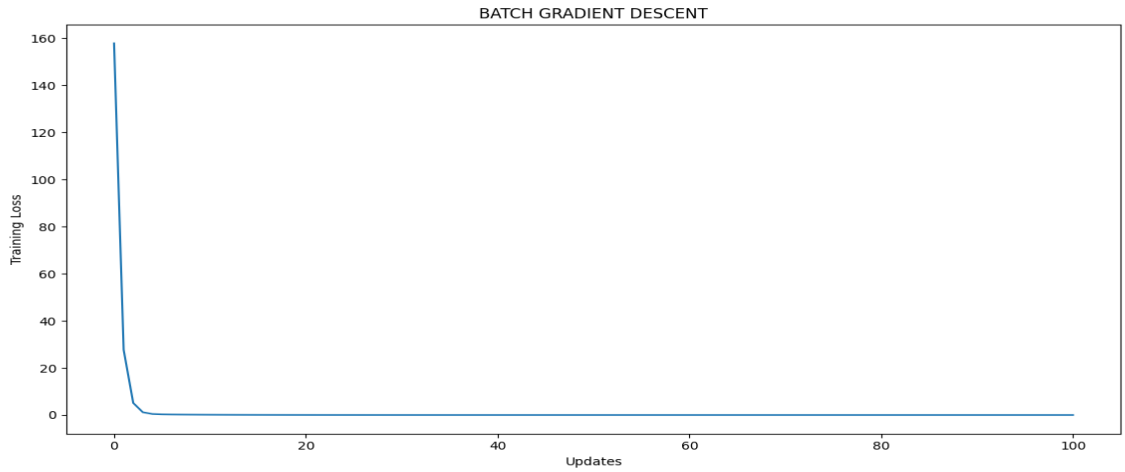
Kousshik Raj (17CS30022)

8-11-2020

Training a Linear Regression model with 5 different optimiser (Gradient Descent, Minibatch Gradient Descent, Stochastic Gradient Descent, Momentum Gradient Descent, Adam) and comparing their performance.

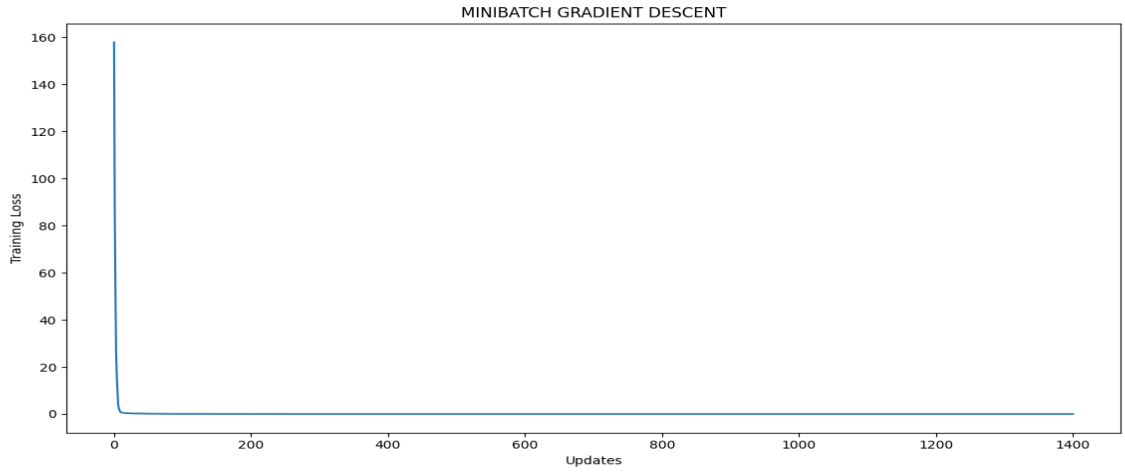
1 Training Loss After Every Update

1.1 Batch Gradient Descent



In the Batch GD, the number of updates is the same as the number of epoch as there is only 1 update per epoch. Because of the shallow network and small training data, we can see an early convergence (after around 4 or 5 epochs) of the parameters after which there is no considerable change in the training loss.

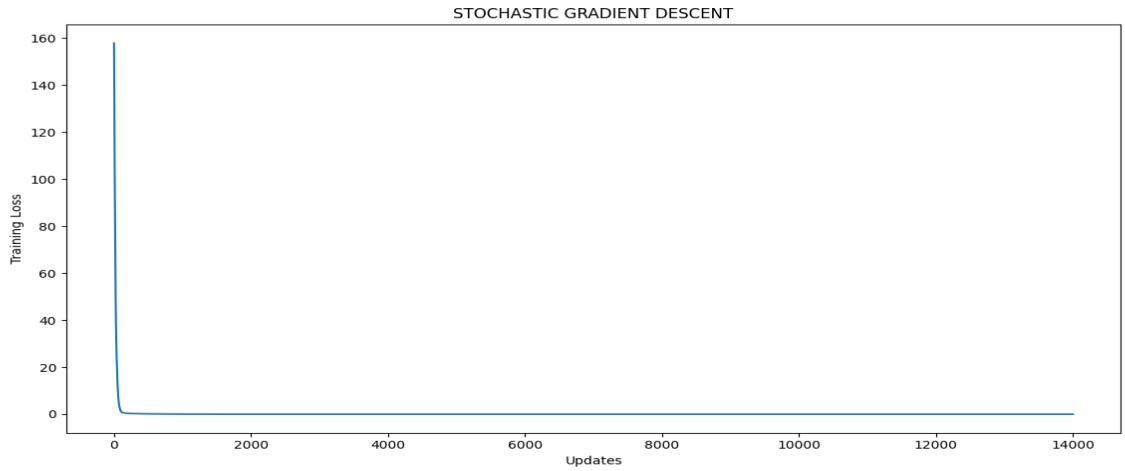
1.2 Minibatch Gradient Descent



In the Minibatch GD, there is an update after processing every batch, which results in nearly 15 times more updates than epochs. From the plot, it can be seen that the model converges after the initial few (around 10) updates, which is almost the same as the Batch GD. This again can be attributed to the fact that the linear model we are currently using exactly models

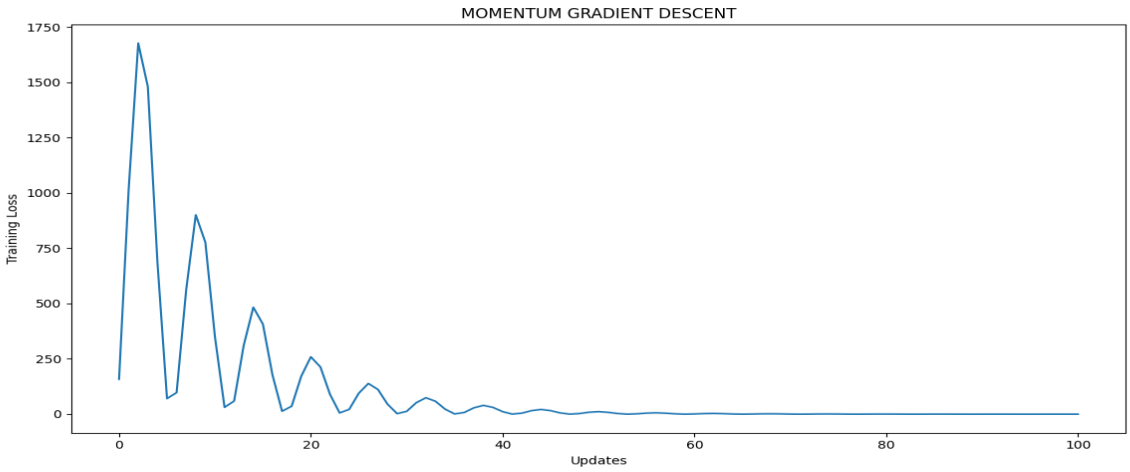
the training data. Because of this, the model quickly converges on the optimal function which generates the training data, similar to the Batch GD.

1.3 Stochastic Gradient Descent



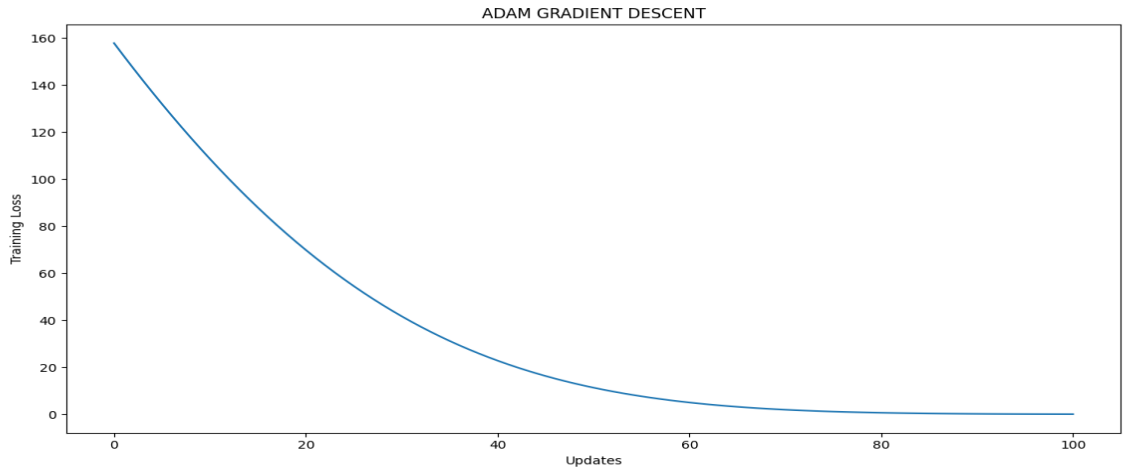
In the Stochastic GD, after processing every training sample there is an update which results in the high number of updates seen. Moreover, similar to the above 2 optimisers, the stochastic GD also converges to the optimal performance just after 3 or 4 epochs, because the data could be actually generated from a linear function which is what our model is trying to find. Adding on the fact there could be less noise, the model quickly converges to the global optimum of the loss function, despite updating the parameters for each and every training data.

1.4 Momentum Gradient Descent



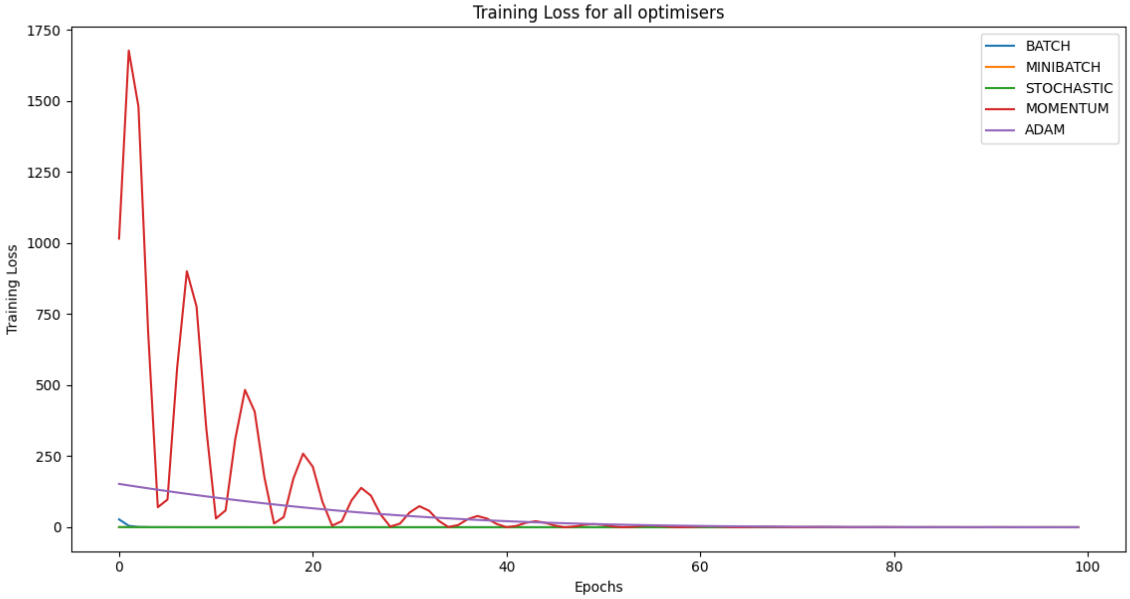
This plot has quite a different convergence pattern because of the added momentum factor in the updates. The momentum remembers the previous direction of updates of the parameters and tries to retain a portion of it for the current update, resulting in an oscillating convergence of the parameters. The magnitude of oscillation keeps decreasing, similar to the oscillation of a stretched rubber band after release.

1.5 Adam Optimiser



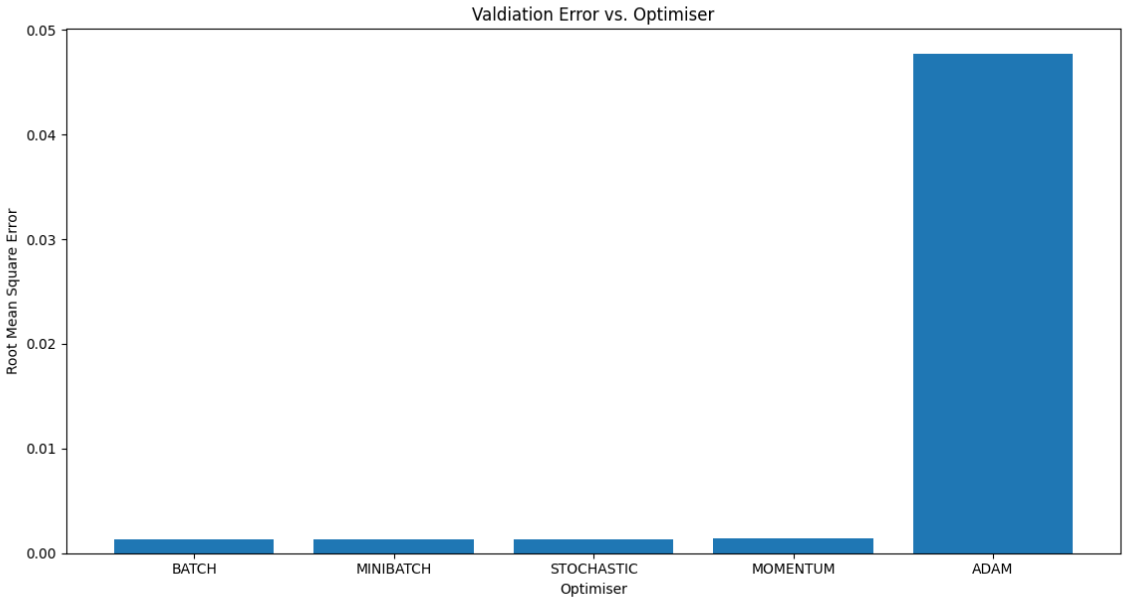
In this plot, we can see a decreasing curve whose rate of decrease also decreases. This is because, in the Adam optimiser, after every update the gradient accumulates, which thereby decreases the rate of update of the parameters. This results in a slow convergence of the parameters. Nevertheless, the model finally converges to the best performance having a very small loss at the end.

2 Training Loss After Every Epoch



In this plot, one can see that Momentum GD has a comparably different convergence pattern compared to the rest. This can be attributed to the momentum of the updates which tends to maintain a portion of the previous update for the current update as well, resulting in its oscillating performance. Another observation is the slower convergence of the Adam Optimiser despite its higher learning rate (ten times) which is a result of the decreasing rate of updates of the parameters as the gradient accumulates. All of the other optimisers have nearly the same convergence pattern which can be because of the small number of training data and small batch size. Under these conditions, the updates are almost the same for each other. We can observe a difference in performance between them in such a case only if we set the learning rate different for all of them.

3 Validation Loss



From the plot it can be observed that Batch GD, Minibatch GD, Stochastic GD and Momentum GD all have nearly the same result when tested on the validation data. This is mainly because, the model network is very shallow (its linear), and also because the amount of training data is very less. These factors result in the convergence of the parameters in the initial few epochs and all of them finally reach the optimal performance. The reason Adam Optimiser has a poorer performance as compared to its counterparts is because it did not have enough epochs to converge on the optimal parameters owing to the decaying learning rate as the gradient accumulates in it.