Lab 3 Report

Jack Hilton-Jones(jhj1g23)

I. Exploring optimisation of analytic functions

Within this lab the Rastrigin Function is optimised, it consists of many local minima and a single global minimum. The function is implemented with A=0.5 and a starting point of $[5, 4]$ is used. Optimisers are used to compute the point after 100 iterations and a loss plot shows the function values at each iteration for each of the optimisers. The optimisers used are Stochastic Gradient Descent (SGD) with and without momentum, Adagrad and Adam. The learning rate is set to $lr = 0.01$ and the momentum is set to 0.9, allowing for a comparison of each of the optimisers. The global minimum of the Rastrign function is at (0,0) which will be used to evaluate the final point the optimisers arrive at, the results are listed in Table I, a loss plot for each optimiser is seen in Figure 1.

Optimiser	Final Point
SGD	(1.61, 1.30)
SGD + Momentum	(-0.01, -0.90)
Adagrad	(4.83, 3.83)
Adam	(3.95, 2.95)

TABLE I
Final points arrived at by different optimisers

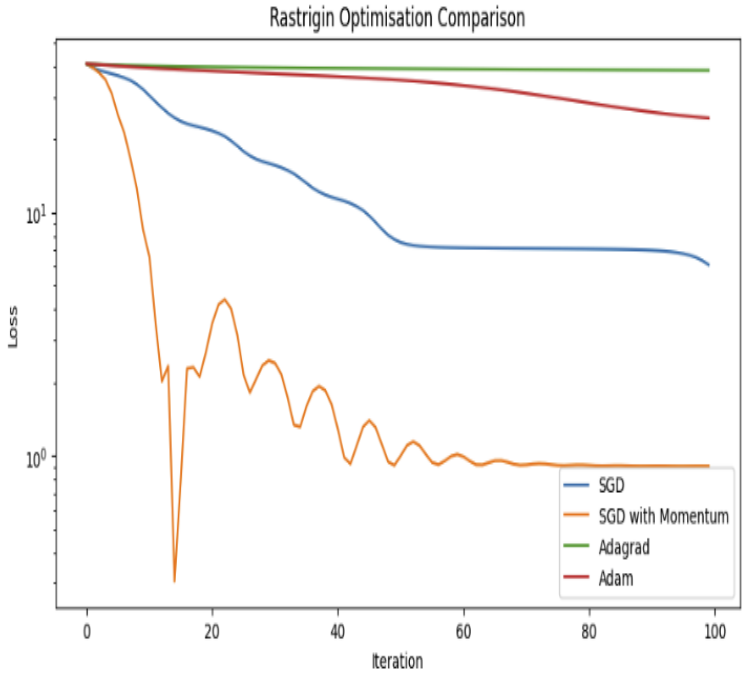

Fig. 1. Comparison of SGD and Adam on 2D Rastrigin

The loss plot and the final point are correlated, it is observed that the closer the final point is to the global minimum of (0,0) the lower the loss. Adagrad is an optimiser which decreases the learning rate dynamically per weight, this squares the magnitude of the gradient to adjust how quickly progress is made. Adam is different as it combines the advantages of Adagrad and RMSProp, using adaptive learning rates for each parameter by storing an exponentially decaying average of past squared gradients and an exponentially decaying average of past gradients. The SGD optimiser is shown to have a lower loss than both Adam and Adagrad, this updates the parameters of the model in the opposite direction of the gradient of the loss function and uses a learning rate with controls the size of the steps taken during the optimisation process. Using a higher learning rate will ensure the model takes bigger steps and vice versa. Momentum can be added to help accelerate in the correct direction while dampening oscillations, this is seen in the results of the loss plot and the final point the optimiser arrived at after 100 iterations. The SGD with momentum outperforms normal SGD, Adagrad and Adam, for the loss function and the final position computed.

II. Optimisation of an SVM on real data

In this part of the lab, SGD and Adam are used to train Soft-margin SVMs with a weight decay of 0.0001 on training data of the Iris dataset. SGD and Adam are trained using a batch size of 25, for 100 epochs.

A. Iris SVM

SGD and Adam are trained at 3 learning rates, 0.01, 0.001 and 0.0001. The validation accuracy and variance are plotted for the different models in Table II

Optimiser	Learning Rate	Mean Accuracy	Variance
SGD	0.01	0.8964	0.00343
SGD	0.001	0.6208	0.04949
SGD	0.0001	0.5024	0.07444
Adam	0.01	0.9032	0.0024
Adam	0.001	0.6480	0.06053
Adam	0.0001	0.4884	0.06846

TABLE II
SVM Training Results with Different Optimizers and Learning Rates

We expect the variance and accuracy to be highly dependent on the number of epochs. If the epochs are high enough for the learning rates to converge, then we would expect a low variance and high accuracy. However, if the number of epochs is too low for the learning rate, the model will not converge and therefore produce a low accuracy and high variance as this will be more dependent on the initial vectors. In addition, if the learning rate is too big the model will not converge producing a low accuracy and high variance, producing an unstable model. All optimisers were trained and run over 100 trials, and the mean and variance were calculated. Both optimisers perform with the highest accuracy at a learning rate of 0.01 with a decrease in performance as the learning rate decreases to 0.001 and 0.0001. This suggests that a lower learning rate might be too slow or insufficient for the optimisers to converge to the best solution within 100 iterations. The variance is lowest for both models at a learning rate of 0.01 compared to the lower learning rates. This is because the accuracy of the lower learning rates is more dependent on the random initial weights. The model cannot converge to a higher accuracy as the learning rate is too small.r This is seen in Table II, the variance increases as the learning rates decrease. The variance is lower for 0.01 learning rate as the model is not dependant on the initial weights and has a high enough learning rate to converge to the optimal accuracy. The accuracy across the optimisers are very similar and show similar patterns.