Part 3

Question 1:

- Implement stochastic gradient and

- compare its performance with that of your gradient descent implementation from Part 1 on the same problem and dataset.
- What happens when both methods are run using the same stepsize?
- Do your observations confirm what has been discussed during the lecture?

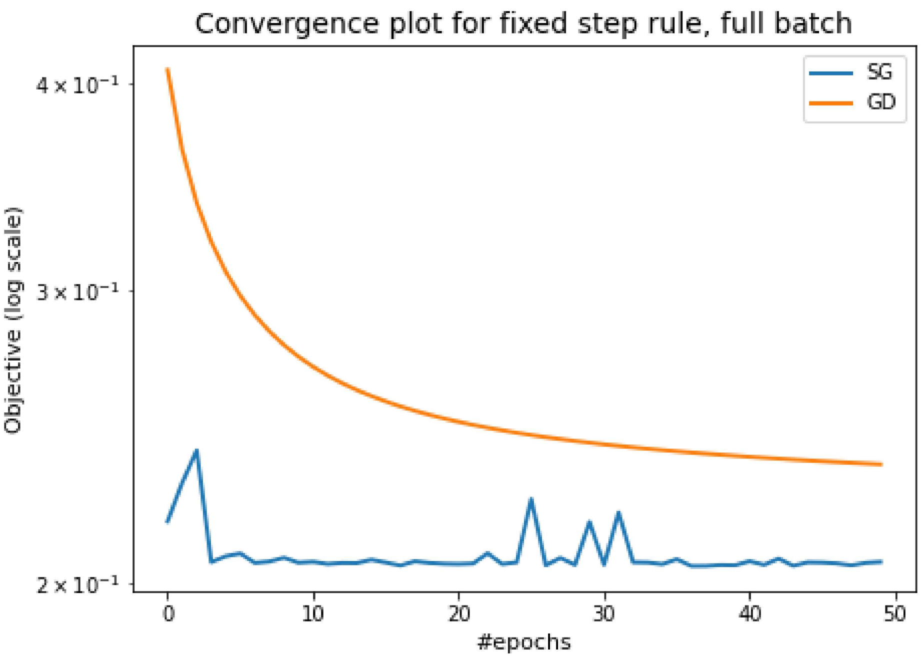
We consider the same dataset as in Part 1 and compare the performance of the stochastic descent and gradient descent technques. To ensure comparability, the following parameters are fixed:

- Number of epochs (eg. number of accesses to the gradient) is set to 30
- x_0 the initial estimate is set to all-zeros$.

As the SG method is not always guaranteed to converge, we opt to compare the methods by making use of a decreasing step size.

- The step parameter is fixed to $\alpha_k = \frac{0.2}{2^k}$ at iteration k.

Plot



This run beautifully illustrates typical characteristics of the 2 algorithms when compared one to the other:

As discussed in class, the stochastic gradient algorithm is not a descent method; behaviour which is clearly observed here in that the objective value is sometimes increasing over successive epochs. The oscillatory phase following a phase of quick descent corresponds to class discussion in that we observe the method stalling when the steps get too small to make a big improvment.

Conversely, gradient descent is guaranteed to descend over every successive iteration and we observe this behaviour clearly below.

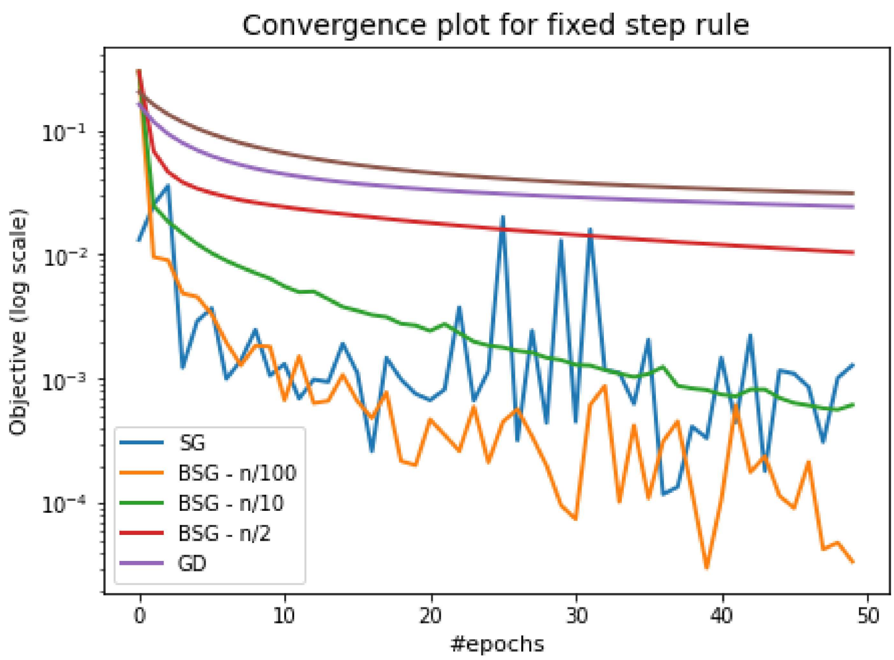
For this method, and for a fixed number of epochs, it seems that SG performs better than the method from which it was derived.

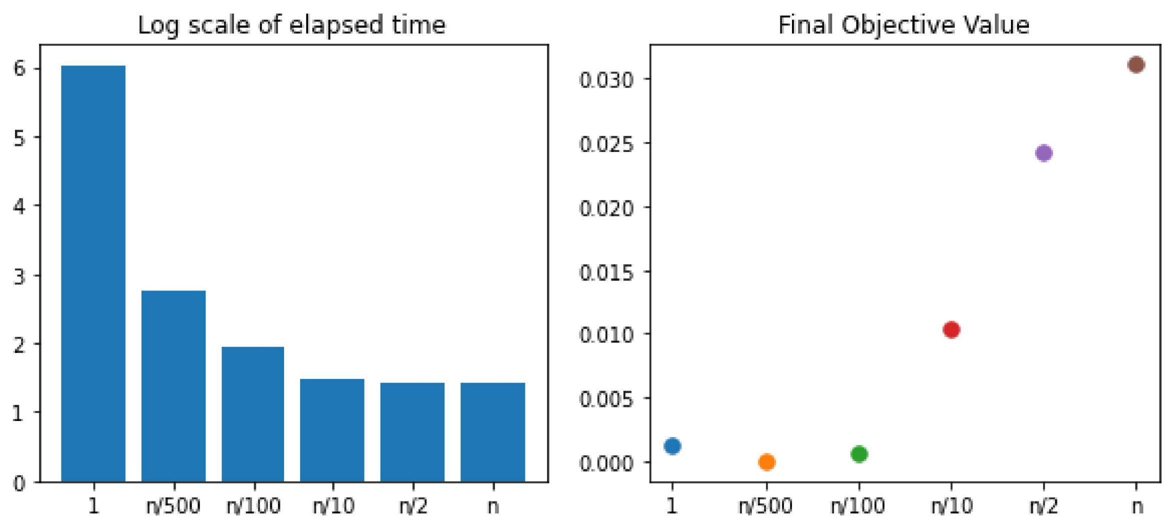
Question 2:

Find a value for the batch size that realizes a good compromise between gradient descent and stochastic gradient.

We consider several different batch sizes, each for a fixed number of epochs and compare this to the gradient descent method (where the batch size is merely n, the number of entries).

$$\text{batch} = \left\{1, \frac{n}{100}, \frac{n}{10}, \frac{n}{5}, n\right\}$$





BSG with n/100 seems to perform almost exactly as well as vanilla SG, but with a runtime comparable to smaller batch size.

We conclude that batch stochastic gradient with a batch size of n/100 would be satisfactory as a compromise between SG and GD.

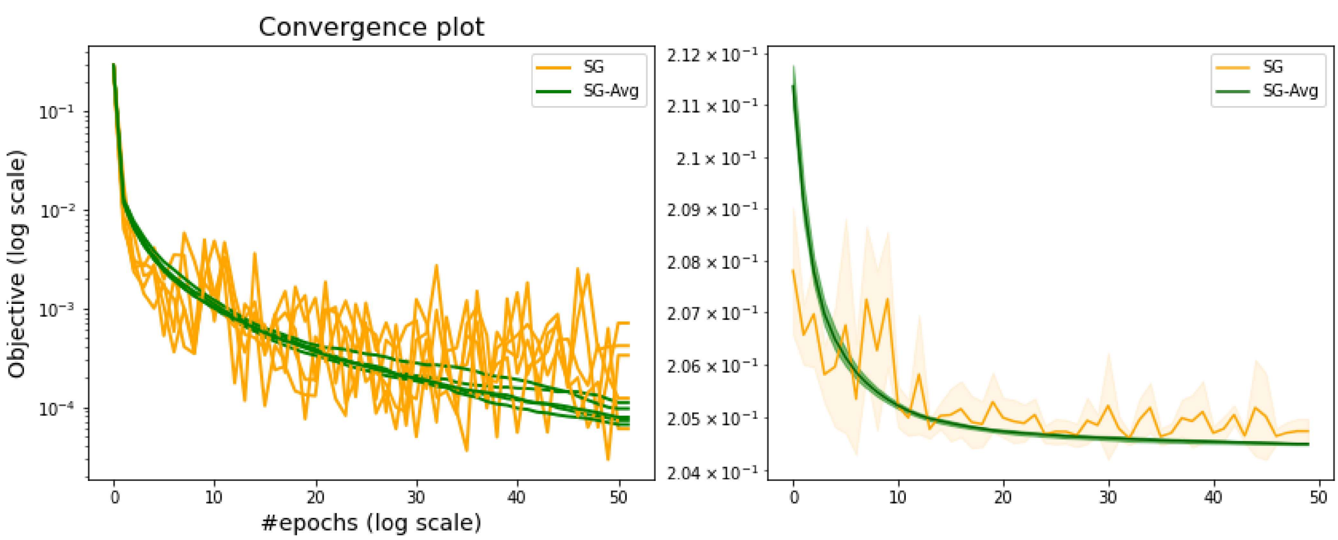
Question 3:

Compare your stochastic gradient method with one of the advanced variants seen in class on your selected problem. Discuss your results, and what interpretation you can draw from them.

We consider SG and compare it with the technique of averaging - a technique designed to reduce the variance. To compare the methods, run 5 repetitions of 50 epochs each for a batch-size fixed to $\frac{n}{100}$ as determined above and a step size rule as before.

Plotted below on the left is the convergence of each of the 5 runs for SG and SG-averaging respectively.

To the right is the average of each method, with the standard deviation coloured in. We observe that SG-Avg has markedly less variance than that of vanilla SG



As mentioned before, SG is *not* a descent method and so not guaranteed to descend at each iteration. It will, however, descend on average. By averaging over iterates, the SG-Avg method is essentially enforcing the 'descent' behaviour of SGD but still retaining its very good convergence results. Moreover, averaging also decreases the variance in the method.