ASSIGNMENT 6

COMPUTATIONAL PROBLEMS

Abhishek Sinha

1. The figures have been labelled as A_LBFGS_NCG_B_C where A could be MISCLA (misclassification) or LOSS(l(b) , B could be small or large for the 2 datasets, C could be ITERATIONS or TIME for iteration complexity and time complexity respectively.

Difference in Performance

For both the small as well as the large datasets, Non-Linear Conjugate Gradient Descent has a lower misclassification fraction when compared to LBFGS (with value of m = 15) if the plotting is done with respect to the number of iterations (iteration complexity) and the number of iterations are kept reasonably small . For smaller datasets, Non-Linear Conjugate Gradient Descent starts overfitting slightly when the number of iterations become large and LBFGS becomes better.

However when the performance is measured with respect to time, then LBFGS clearly outperforms Non-Linear Conjugate Gradient Descent since it achieve a comparable misclassification error with around 1/5th the time. But if we want very low misclassification, then either we can use non-linear conjugate gradient descent or we will have to increase the value of m for LBFGS which will then lead to an increase in the running time.

Difference in Function Calls During line search (FUNCTION_CALLS_X.fig)

For non linear conjugate gradient descent, the number of function calls BLTS takes to return decreases roughly monotonically with the iteration number of outer loop. Thus initially very high number of function calls are required before BLTS returns.

In case of LBFGS, for a very large number of iterations Backtracking line search returns almost immediately i.e. with eta = 1. But as the outer iteration number increases, BLTS starts having very high number of function calls (close to 400 function calls). Thus for small values of k, LBFGS proceeds very quickly but the later iterations become very slow.

Comparison to Gradient Descent (GD_LBFGS_NCG.fig)

Both the methods perform better than gradient descent as long as the number of iterations is not very high. If the number of iterations is very high then Gradient Descent becomes better than non-linear conjugate gradient descent.

The plot against time shows that gradient descent takes longer time to converge to a given error provided the dataset if small.

2. The figures have been labelled as MISCLA_BFGS_LBFGSX_Y where X is the look-back for LBFGS i.e. m and Y is TIME or ITERATIONS for time and iteration complexity respectively

m = 2

This a small value of m, so LBFGS and BFGS have significantly different plots.

When comparing misclassification as a function of iterations, BFGS clearly performs better for small number of iterations but for large number of iterations, LBFGS starts performing better probably because there is lesser overfitting.

When measured with respect to time, the lowest possible misclassification fraction LBFGS achieves is lesser and it takes an order of magnitude lesser time lesser time to achieve that.

Thus for small dataset, LBFGS with m = 2 is the preferred choice over BFGS.

m = 32

In this case, again, BFGS performs better for small number of iterations and for larger number of iterations LBFGS performs better.
However to achieve the highest accuracy, more time is needed in this case when compared to BFGS.

<u>m = 512</u>
With this high value of m (close to the value of n which is around 650), the graphs of LBFGS and BFGS become very similar. LBFGS approaches BFGS as m tends to n.