OTDM: Pattern recognition with Single Layer Neural Network

Gabriel Zarate, Ximena Moure

October 2023

Contents

Seeds

The values used for the seeds are: tr_seed= 20344813 and te_seed= 2346539

1 Introduction

The aim of the project centers on the construction of a Single-Layer Neural Network (SLNN) for the recognition of digits, employing different first derivative optimization techniques. Each digit is represented by a 35-pixel matrix (7x5), where each pixel is designated a value of 10 or -10, followed by the application of Gaussian noise. The neural network is engineered to recognize all digits ranging from 0 to 9.

The optimization techniques are employed with the goal of minimizing the Loss function with L2 regularization. This function is pivotal in establishing the weight relationships between each neuron and the output neuron. The optimization methods being explored are the Gradient Method (GM), Quasi-Newton Method (QN-BFGS) and Stochastic Gradient Method (SGM).

2 Study of the convergence

2.1 Experiment Setup

We will examine the global and local convergence of the three algorithms solely based on the objective function $\tilde{L}$.

To guarantee the reproducibility of the results, the following seeds were employed:

```
tr_seed= 20344813; te_seed= 2346539; sg_seed=565544;
```

To generate the dataset the following parameters are used:

```
tr_p = 250; te_q = 250; tr_freq = 0.5;
```

The program will be executed for all combinations of λ, target digit and algorithm, as depicted below:

$$\lambda : \{0, 0.01, 0.1\} \quad \text{digit} : \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \quad \text{algorithm} : \{\text{GM}, \text{QNM}, \text{SGM}\}$$

Other fixed parameters:

```
epsG = 10^-6;   kmax= 1000
ils=3; ialmax = 1; kmaxBLS=30;
epsal=10^-3; c1=0.01; c2=0.45;
sg_al0 = 2; sg_be = 0.3; sg_ga = 0.01;
sg_emax = kmax
sg_ebest = floor(0.01*sg_emax);
```

2.2 Global convergence

Pattern Recognition with Single Layer Neural Network involves the use of computational models to identify patterns in data. The ultimate goal is to find a suitable solution to this pattern recognition problem. A key consideration in evaluating the effectiveness of different algorithms is the concept of Global Convergence.

Global Convergence is an algorithm's property, that reflects its capacity to reach a solution for a given problem. The Zoutendijk Theorem provides a rigorous foundation for the notion that an algorithm will ultimately find a solution, which proves the previous statement.

For this specific case of study, the objective function is the Loss function with L2 regularization ($\tilde{L}$), which is composed by two main elements:

- Training Set Prediction Difference: consists on the algorithm's predictions and the actual values in the training dataset. Measures how well the algorithm models the data relations.
- Regularization Term: promotes simplicity and prevents overcomplexity within the model by penalizing overly intricate or elaborate models. Depend on λ as a parameters.

As the goal is to minimize $\tilde{L}$, we will check how the value varies within the different algorithms and λ values. The detailed results for the global convergence studies are on Table 1.

Algorithm	GM			QNM			SGM		
Lambda	0	0.01	0.1	0	0.01	0.1	0	0.01	0.1
Mean Training Accuracy	99.96	99.40	96.80	99.52	99.40	96.80	99.36	98.36	73.12
Mean Test Accuracy	98.76	99.24	93.48	98.80	99.24	93.48	99.52	99.44	94.16
L*	0.0004	0.0479	0.1276	0.0048	0.0479	0.1276	0.0052	0.0604	0.2528

Table 1: Global Convergence Results

And also to visualize better the outcomes of the fist experimentation, a group of boxplots are shown on Figure 1.

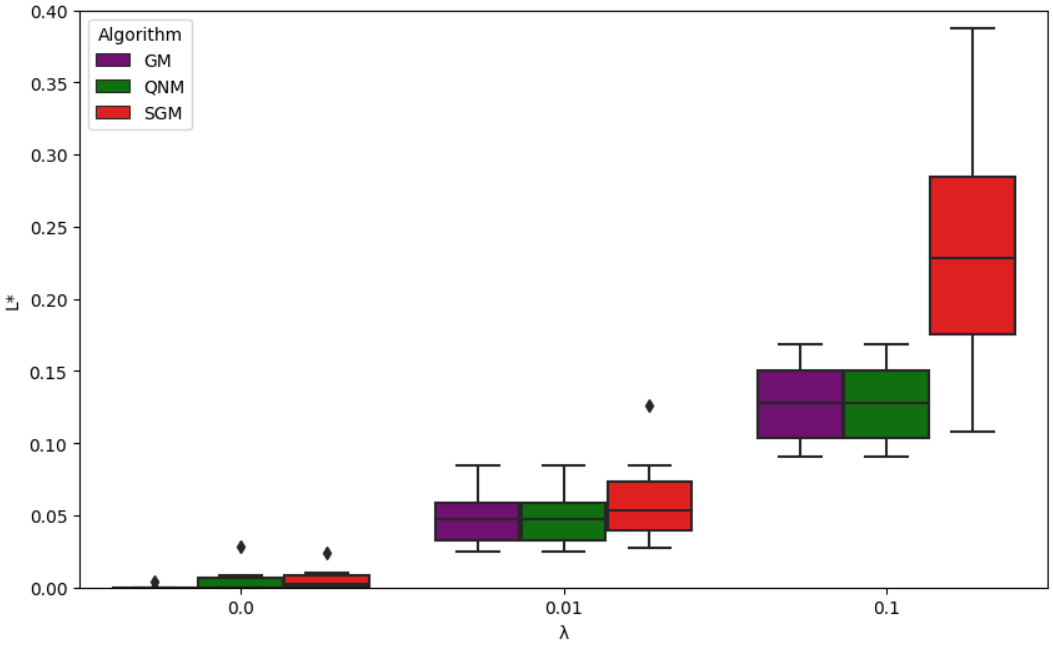

Figure 1: L* by Algorithm

From the results table and the plots we can see that:

- All three algorithms demonstrate achieving global convergence.
- GM and QNM accuracy tends to decrease as the λ increases
- SGM demonstrates to be sensitive to the choice of λ. At 0.01, it achieves high accuracy, while at 0.10 it drops significantly, suggesting that careful tuning of λ is necessary for SGM.

- There exists a dependency on λ, because for all the algorithms, it significantly affects performance. When it is set to 0 (no regularization), all algorithms perform exceptionally well with high training accuracies, but as it increases, the regularization term has a greater impact, leading to a trade-off between complexity and accuracy.
- Higher λ values encourage simpler models but may result in reduced accuracy.
- For SGM it is important to consider the Zoutendijk condition, because if this condition is satisfied, it means that SGM is making progress towards the best solution. This because the theorem states that the direction of progress, must be an approximation of the gradient of the loss function, ensuring it gets closer to the optimal solution with each step.

Finally, for choosing the best combination of algorithm and λ, that provides the best results in terms of global convergence, we conclude that for all the algorithms the accuracy tends to decrease as the λ increases, but still with 0.1 the accuracy is really good and the L* value is really small. So the best λ for these algorithms is 0.1, because this combination maintains high accuracy while still benefiting from regularization, striking a balance between complexity and performance.

2.3 Local convergence

2.3.1 Speed of convergence

The speed of convergence and the number of iterations for the three algorithms is depicted in Figure 2 and 3 respectively.

To better visualize the results, a logarithmic scale has been applied. This is due to the fact that the SGM algorithm has very low times so a logarithmic transformation can provide a clearer visualization of the differences and variabilities across all three algorithms.

SGM generally has the fastest execution times, followed by QNM, and then GM. GM and QNM have similar median execution times, but GM has a slightly narrower spread and a couple of high outliers. SGM not only has the lowest median execution time but also the least variability in execution times, although it does have a high outlier.

The number of iterations for SGM is higher than both GM and QNM, which indicates that it requires the most iterations among the three algorithms. QNM converges in fewer iterations than GM.

When using SGM, the algorithm updates weights based on a subset of the training data, known as a mini-batch, at each iteration. These mini-batch updates are considered as individual "iterations". On the other hand, when using algorithms like GM or QNM, they typically use the entire dataset to compute the gradient and update the model parameters in each iteration. Thus, directly comparing the number of iterations between SGM and the other two algorithms can be misleading. The iterations in SGM are generally faster (since they are based on smaller subsets of the data) but might be more numerous, especially if the dataset is large.

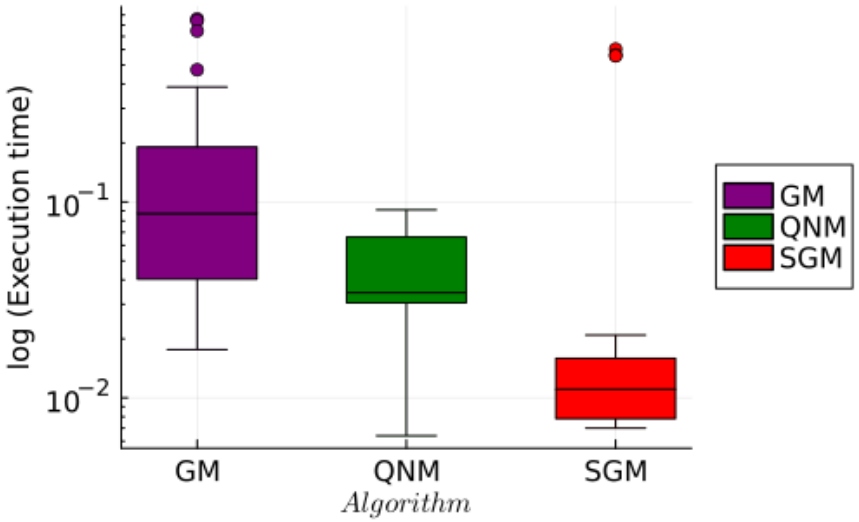

Figure 2: Execution time for each algorithm

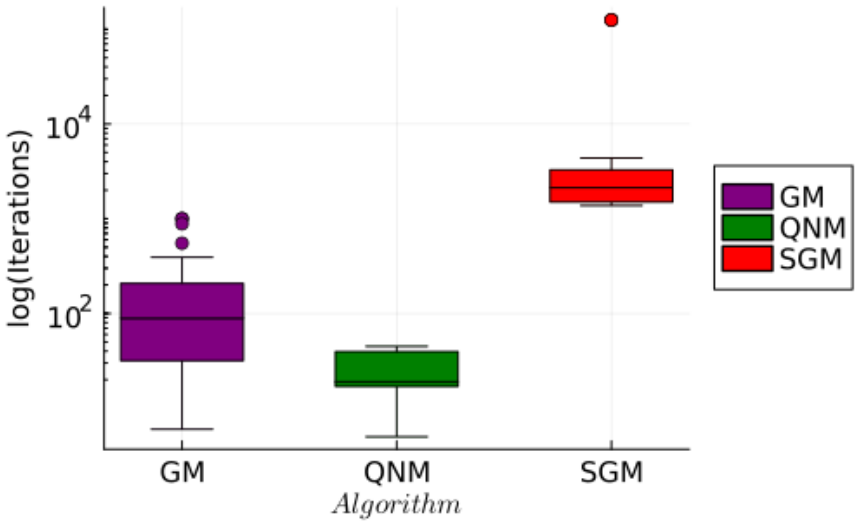

Figure 3: Iterations for each algorithm

2.3.2 Convergence speed in terms of λ

In the following section the analysis will be based on Figure 4 and Figure 5.

When λ is set to 0 the execution times vary significantly among the algorithms, with QNM emerging as the quickest. As λ increases, both GM and SGM show a decreasing trend in their execution times, with GM showing a consistent decrease and SGM having a more pronounced drop between $\lambda = 0.0$ and $\lambda = 0.01$. On the other hand, $QNM's$ execution time remains relatively stable with a slight increase observed between $\lambda = 0.0$ and $\lambda = 0.01$, after which it stabilizes.

It can be seen that the bars representing execution times become shorter and closer in height to each other, indicating that the execution times are more consistent across the different algorithms

when a regularizer is applied.

As λ increases, the number of iterations required by all algorithms tends to decrease or remain relatively stable. This suggests that regularization can help in faster convergence, requiring fewer iterations for the optimization process. QNM, which starts with the fewest iterations at $\lambda = 0$ experiences an increase in iterations with rising λ but remains efficient in terms of iteration count.

Higher values of λ can further smooth out the objective function. This might explain faster convergence or shorter execution times for some algorithms with increasing λ.

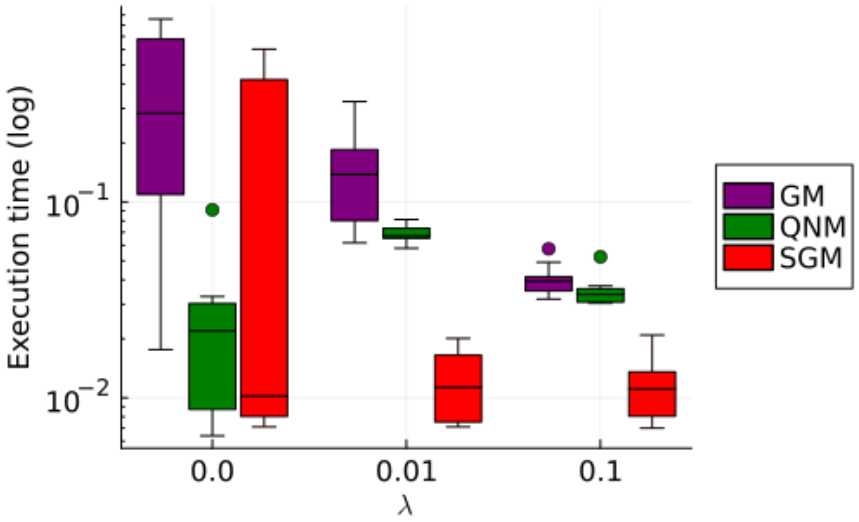

Figure 4: Execution time for each algorithm and each λ

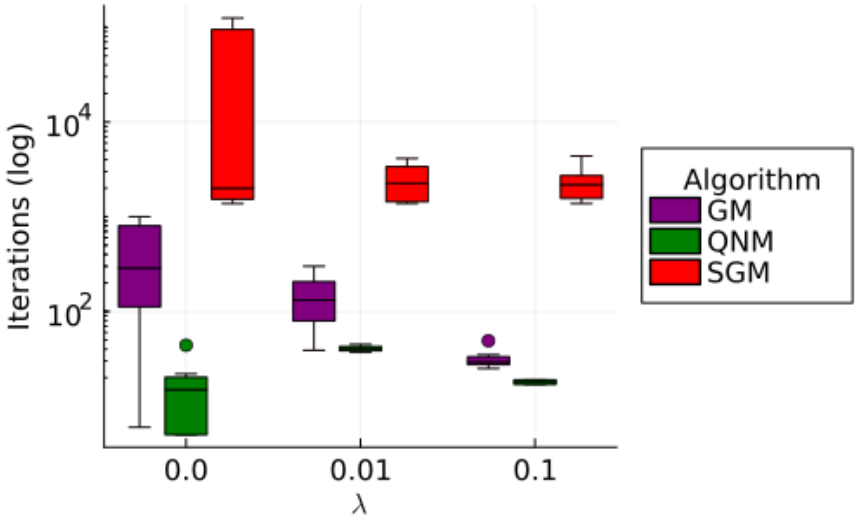

Figure 5: Number of iterations for each algorithm and each λ

2.3.3 Running time per iteration

In this section, we will investigate how the execution time per iteration for each algorithm varies with λ. It is worth noticing that SGM spends a remarkably short amount of time on each iteration. This is why we do not plot it with GM and QNM.

From Figure 6 we can see that across all λ values, the QNM algorithm generally has a longer running time per iteration compared to the GM algorithm. While QNM determines its direction by multiplying the approximation of the actual Hessian with the gradient, this introduces additional computations, making it less speedy than the gradient method.

The shortest per-iteration execution time belongs to SGM, which may be due to the fact that it only uses a random fraction of the data. The regularizer's influence on SGM appears minimal.

Using the theoretical foundations of how each algorithm functions as a guide, we can deduce certain performance characteristics. The GM algorithm employs relatively straightforward computations in each iteration, accounting for its minimal per-iteration runtime. However, this simplicity can lead to more iterations being required to converge to a solution, hence a potentially longer overall execution time. On the other hand, QNM spends more time on each iteration due to its necessity to approximate the Hessian matrix. This added complexity, however, often enables it to find a solution in fewer iterations, making it potentially more time-efficient overall. From a theoretical standpoint, GM typically exhibits linear local convergence, while QNM can achieve superlinear convergence under specific conditions. This generally translates to QNM reaching a solution more rapidly than GM.

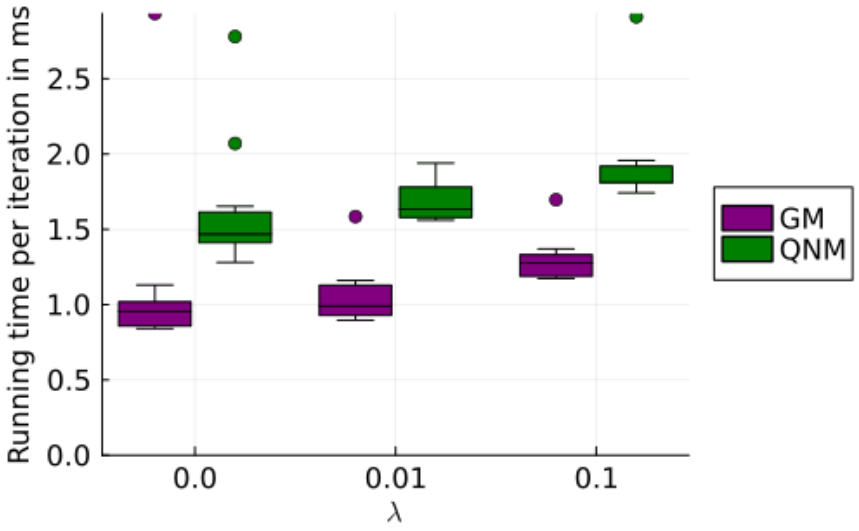

Figure 6: Running time per iteration

2.3.4 Discussion

- Global convergence:
  - The best λ for all the algorithms in terms of global convergence is 0.1, because this combination maintains high accuracy while still benefiting from regularization, striking a balance between complexity and performance
- Local convergence:

- For GM and SGM, $\lambda = 0.01$ seems to be the optimal choice as it balances a lower number of iterations with a stable running/execution time per iteration.
- For QNM, $\lambda = 0$ is the best choice as it offers the lowest number of iterations with consistent running time per iteration.
- When evaluating the efficiency of algorithms, we should consider the specific application and weight the relative importance of computation time against the number of iterations.

3 Study of Accuracy

3.0.1 Experiment Setup

The following study is set to analyse the recognition accuracy of the SLNN in a more realistic case, so some execution parameters will be changed. First the data set sizes will change, the training will increase to 20000 and the test will be the training size divided by 10. But the main change is setting tr_q as 0, which will make that all the digits are represented equally in the dataset.

Additionally, the selected lambda per each algorithm will be the one that gave the best results in the previous section, with the smaller data set. So a summary table with the mean test accuracy results is shown on Table 2.

Algorithm	GM			QNM			SGM		
Lambda	0	0.01	0.1	0	0.01	0.1	0	0.01	0.1
Mean Test Accuracy	98.76	99.24	93.48	98.80	99.24	93.48	99.52	99.44	94.16

Table 2: Mean Test Accuracy Summary

And as it can be seen, the values that will be used for the lambdas are: 0.01 for GM and QNM, and 0 for SGM.

3.0.2 Accuracy and Training Speed

So as it was defined, the experimentation was done over a bigger dataset. The obtained results of the accuracy and training speed tests are shown on Table 3

Algorithm	GM	QNM	SGM
Mean Test Accuracy	99.05	99.05	99.54
Mean Training Time (sec)	6.30	4.76	4.73
Mean Iterations Number	80.7	39.2	26800.0

Table 3: Accuracy and Training Speed Results

From those results we obtain the following insights:

- SGM is the best-performing algorithm in terms of recognition accuracy, surpassing both GM and QNM by a little.
- SGM also outperforms GM and QNM in training speed, with the lowest mean training time.
- However, SGM has the highest mean number of iterations, which suggests that it might be computationally intensive or potentially inefficient in terms of convergence compared to GM and QNM.

In summary, if you prioritize both high accuracy and reasonable training speed, SGM is the best choice. However, if computational efficiency and a lower number of iterations are factors that will be taken into account, either because there is not much computational power, GM and QNM could be better options, especially QNM given its shorter training time. Finally the final decision will depend on the specific requirements and limitations of its use.

3.0.3 Differences with the first experiment

For this part first we will compare if the combination of λ-algorithm obtained on the second experiment with the bigger dataset focusing on the maximization of the test accuracy differs the ones obtained in the first experiment, which focuses on the minimization of $\tilde{L}$.

The comparison of the combination of λ-algorithm per experiment is shown on Table 4

Algorithm	GM	QNM	SGM
Lambda Exp. 1	0.01	0	0.01
Lambda Exp. 2	0.01	0.01	0

Table 4: λ-algorithm per experiment

As it can be seen, only the GM with $\lambda = 0.01$ coincides. But this is only because the λ values for this experiment were chosen only focusing on the test accuracy over the small dataset, ignoring other important factors such as low number of iterations or convergence speed. This is because the accuracy and L* are two different criteria. the first one represents how well the model classifies data and the second one measures the goodness of fit with an additional regularization term to control complexity. So the best λ for maximizing accuracy would not be the same for the L* minimization, because for accuracy it will try to fit the data as perfectly as it can, and the other case will try to ensure regularization. Having two different decision criteria, so it makes sense that the combinations does not coincide.

The important thing that we can compare from the two experiments are the best algorithms in terms of accuracy: for a big dataset is the SGM, achieving the highest test accuracy, and in experiment 1, the best algorithm was also SGM and also using $\lambda = 0$. This can mean that in SGM, when no regularisation is used, the dataset size doesn't impact highly to the performance, because it performs well in a small and in a big dataset.

Also it can be seen that $\lambda = 0.1$ seems to woks slightly better in smaller datasets, meaning that the value of this parameter needs to be tuned differently as long as the dataset size increases. A larger dataset may require more regularization to prevent overfitting.

3.0.4 Conclusions

- SGM is the best choice if you prioritize both high accuracy and reasonable training speed for a classification task and a big dataset .
- If computational efficiency and a lower number of iterations are the main factors that will be taken into account, QNM is the best choice.
- In both experiments, in terms of test accuracy, the best algorithm was SGM using $\lambda = 0$. This can mean that in SGM, when no regularisation is used, the dataset size doesn't impact highly to the performance.
- A larger dataset may require more regularization to prevent overfitting.

References

[1] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2006.

[2] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*. Springer, 3rd ed., 2008.