Neural Networks

Optimització Matemàtica

Grau en Ciència i Enginyeria de Dades
Deadline: 10/05/2022
Pol Lizaran Campano & Clàudia Mur Planchart

INDEX

Introduction

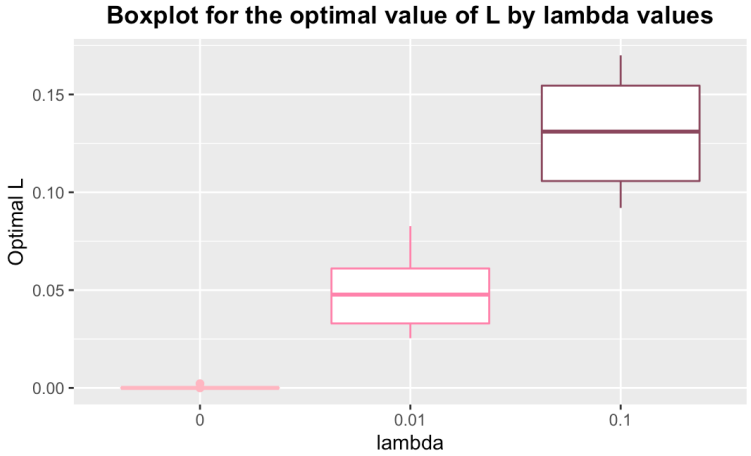
In order to study the results obtained in a more in-depth way, we've decided to import the data file to R and plot several results. By that, we've been able of analyzing the different outcomes generated by the three algorithms (GM, QNM, SGM).

The aim of this project is to be capable of determining which algorithms give better results for minimizing the error of pattern recognition, and see which combinations of λ-algorithms have better performance.

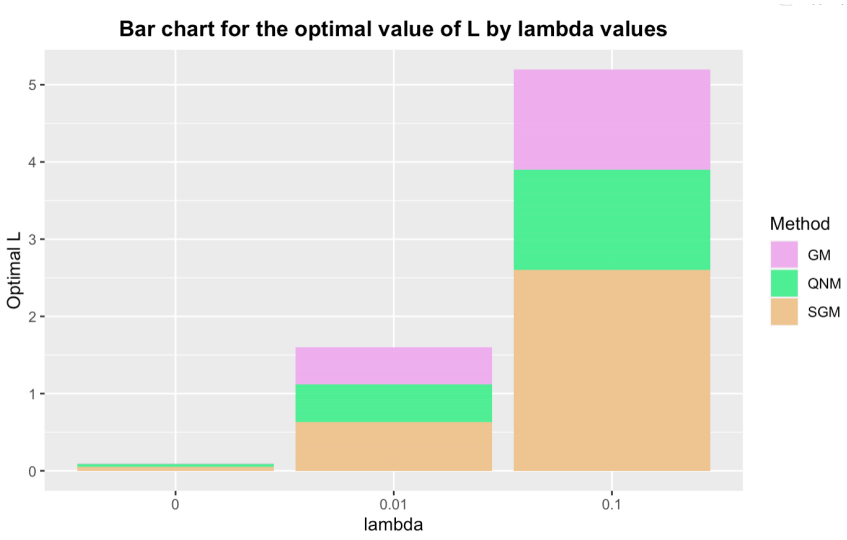
Study of convergence

a) Global convergence

First of all we will study the global convergence of the three algorithms in terms of the objective function.

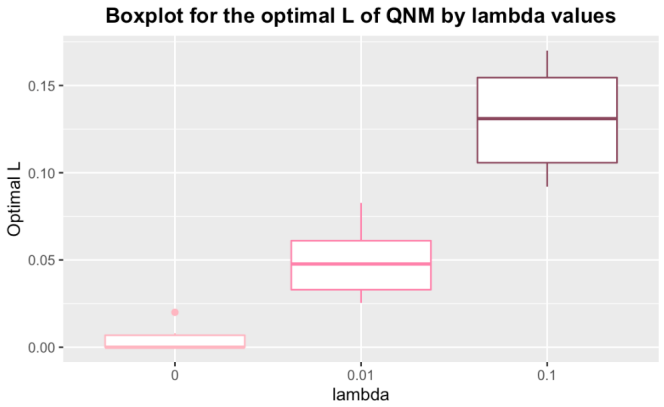


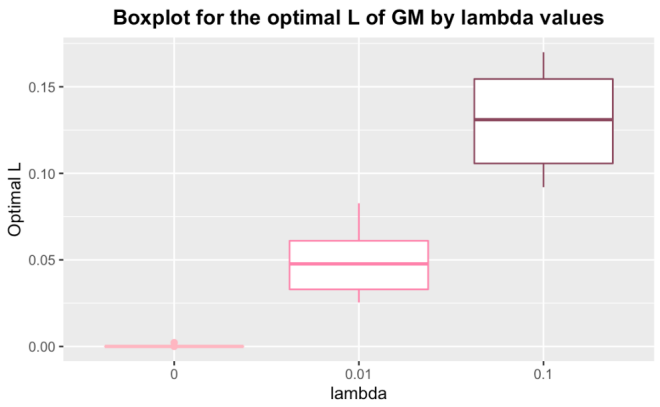
The above chart shows how the loss of the objective function at w* behaves, depending on the regularization parameter (λ) used. It is seen that when augmenting lambda, the optimal function value also gets bigger. That is, the optimal solution is worse since there are harder restrictions that unallow the best minimization of L. Furthermore, an increase of lambda makes the objective function more convex. Thus, the algorithms are able to find a solution faster.



This chart compares the optimal value of L that each algorithm reaches depending on lambda. As it has already been concluded, when lambda increases the solution gets away from the optimal solution.

Looking at a particular lambda value, the worst optimal L is given by the SGM. Due to the fact that this bar chart doesn't allow to compare the QNM and GM the following graphics have been done.



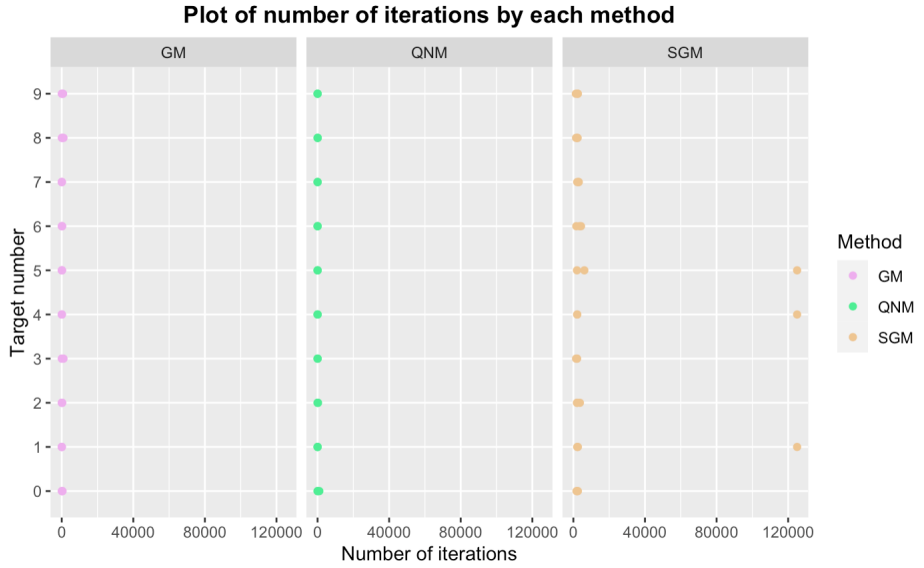


Surprisingly, it is observed that when using regularization, the optimal value of the objective function is exactly the same. However, when lambda is zero, the GM reaches a better solution thus being closer to the true optimal.

Therefore, the best combination regarding the optimal value of the objective function L is the pair 0.0-GM.

In addition, it will be analyzed whether the algorithms are able to identify a stationary solution of the loss function. Both GM and QNM stopping criteria take into account the number of iterations and the gradient norm. We can state that whenever the gradient norm of the Loss function is zero, or almost zero, the algorithm has reached a stationary solution. However, when the number of iterations at the end equals kmax, the algorithm stops without reaching a stationary solution.

On the other hand, the SGM has a particular stopping criteria that does not take into account the gradient norm. Nonetheless, the sg_ebest substitutes it. When it is the cause of the stop it implies that for the last sg_ebest epochs the solution has not been improved. Then, it is considered that a stationary solution has been found. However, when the number of iterations is sg_kmax the algorithm stops without reaching a stationary solution.



Mention that from now on SGM method will be displayed without the 3 extreme cases for better interpretation, although they will continue to be taken into account.

Using the criteria explained before and knowing that kmax = 1000 it is seen that the GM and the QNM reach a stationary solution except in some cases with the combination 0.0-GM and 0.0-QNM. Concerning sg_kmax = 125000 it is known that all the SGM executions converge to a stationary solution saving some cases without regularization, as happens with the other methods, due to the fact that the smaller lambda, the less convex is the objective function. Therefore, it is harder to reach the optimum.

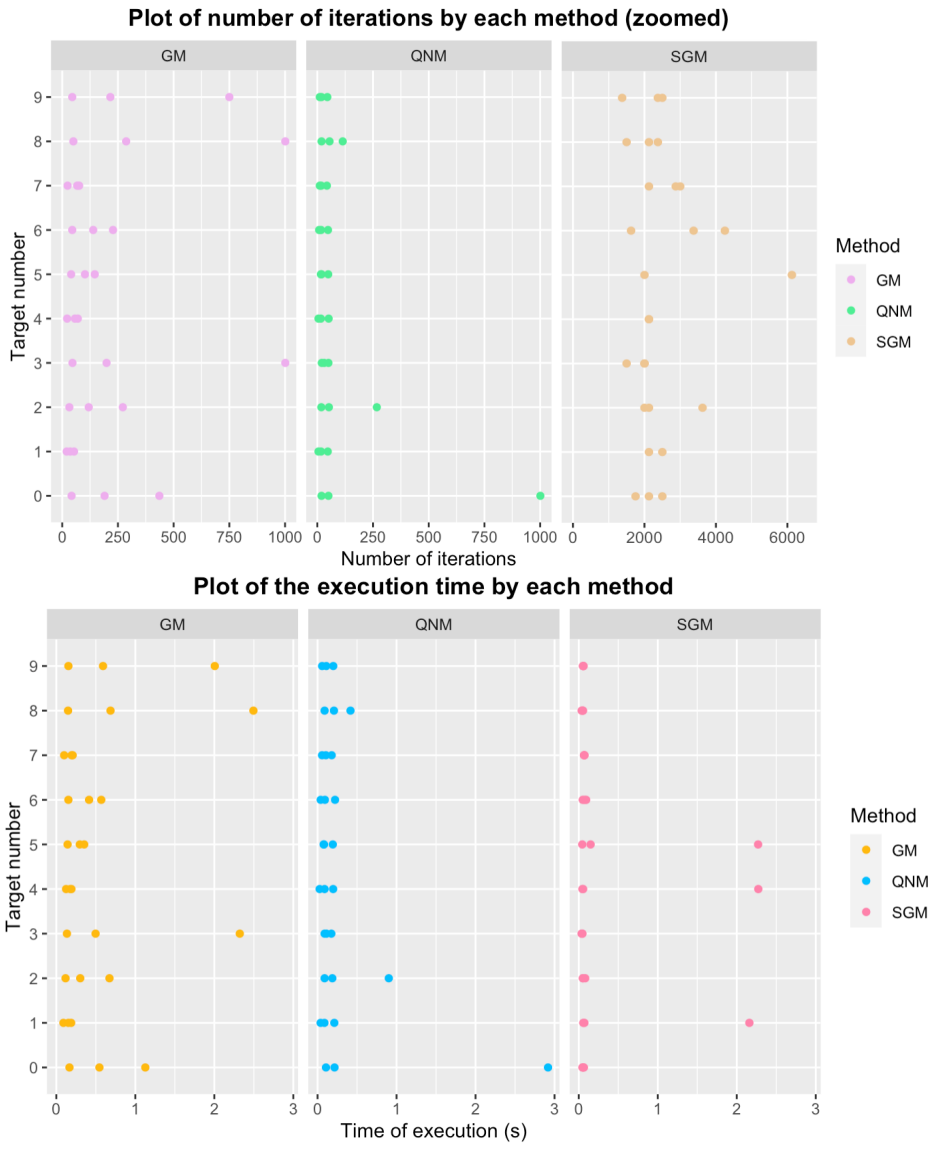
b) Local Convergence

i.) Speed of convergence (tex & niter)

The study of convergence can be done regarding the total number of iterations or by looking at the execution time. However, the comparison between algorithms has to be taken with care as their internal structure isn't the same.

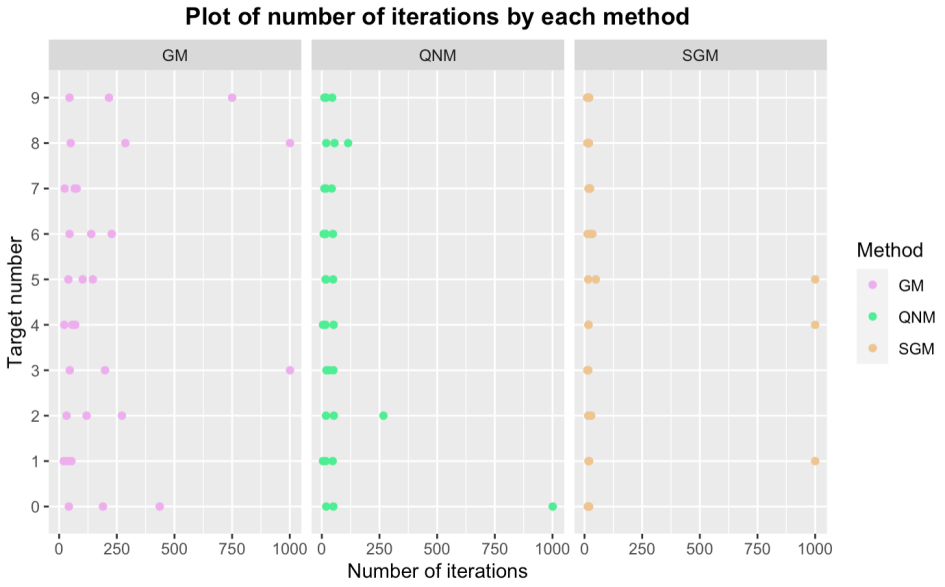
Algorithms	**GM**	**QNM**	**SGM**
Mean number of iterations	193.7	71.37	14705.17
Mean time of execution (sec)	0.51	0.25	0.28

Recall the above plot:



On the one hand, it is stated that the method that takes more iterations is the SGM. Nevertheless, when analyzing the plot of execution time it is clearly seen that the algorithm that lasts more is the GM despite not being the one with higher number of iterations because the SGM computes the iterations in a different way. In order to make a proper comparison, the number of iterations of the SGM will be normalized being divided by sg_kmax/kmax = 125, meanwhile the time of execution will remain unchanged.

Algorithms	**GM**	**QNM**	**SGM**
Mean number of iterations	193.7	71.37	117.64
Mean time of execution (sec)	0.51	0.25	0.28

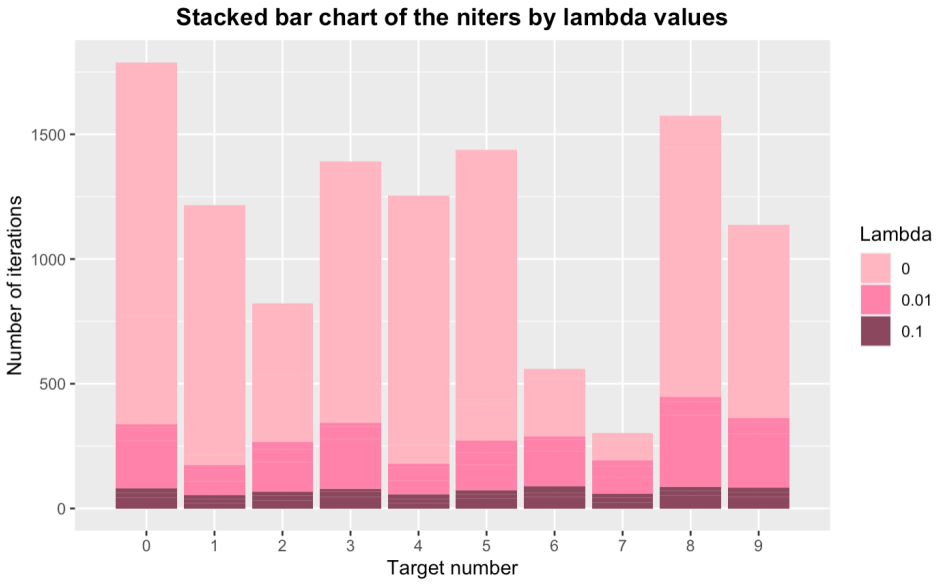


Now it is seen that the execution time and the number of iterations are proportional. The QNM is the method with less iterations and execution time thus being the fastest for little dimensions Neural Networks. Besides, the GM is the algorithm that lasts more and it takes more time than the SGM because it works with a smaller matrix.

Later on we'll see that QNM will result in more time of execution due to the large computational effort that it has to approximate the Hessian, the fact that when working with small matrices is not that noticeable.

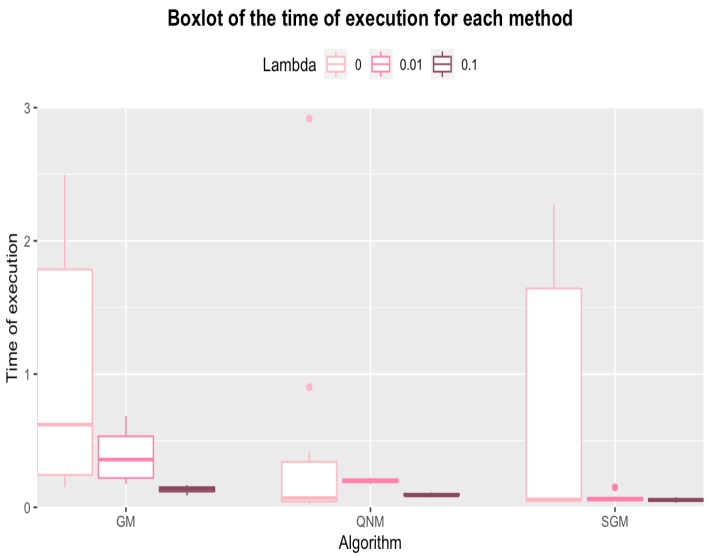
ii.) Speed of convergence (λ)

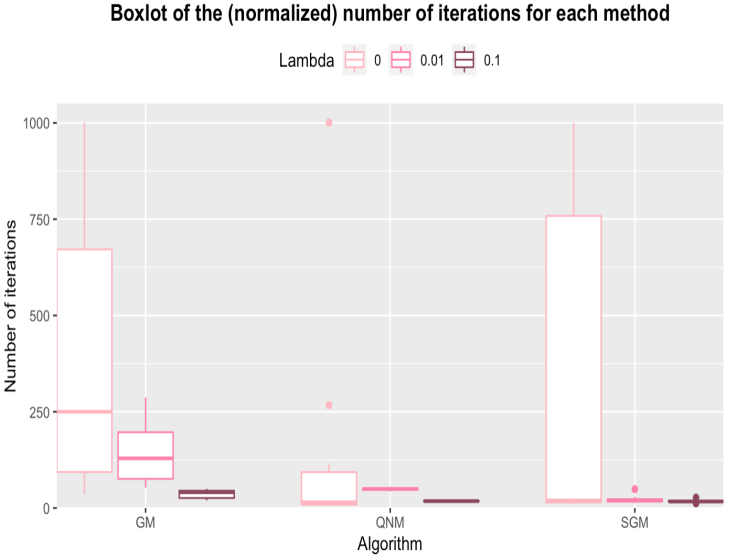
The main goal of this section is to study how does a change in λ value affect on the computation. This regularization allows us to control the total size of the vector of parameters by adding a penalty term ($\lambda \frac{||w||^2}{2}$ in Tikhonov's regularization). By that, we are capable of controlling overfitting the model. The following image shows an accumulated graphic of how do the algorithms used react to λ value modifications.



Whenever the function to minimize has a great amount of coefficients, $\lambda \frac{||w||^2}{2}$ will be bigger. Thus, when applying Tikhonov regularization, the function to minimize: $L(w;\ X^{TR},\ y^{TR})\ +\ \lambda \frac{||w||^2}{2}$, will be greater too.

Taking into account the above mentioned we can state that the bigger λ, the more convex the Loss function is, allowing the algorithm to find a stationary point faster. That's the reason why by looking at the graphics, we observe that $\lambda\ =\ 0.1$ is the one having less iterations.

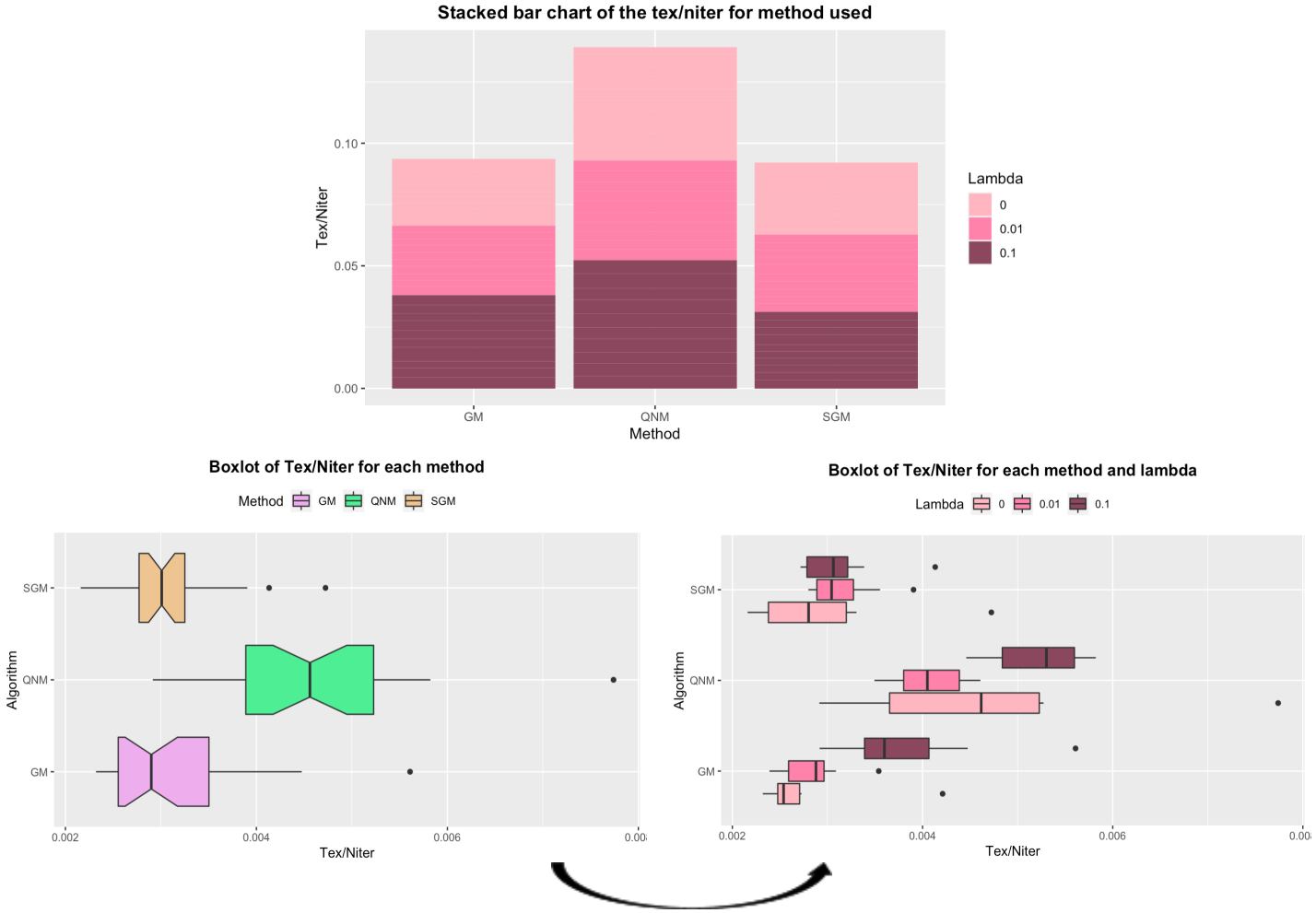




In the above graphics it is seen what was mentioned before. For each Algorithm used, the regularization with $\lambda = 0$ is the one taking more iterations and time of execution on average. Furthermore, it is seen that a big λ value implies a lower variance among experiments because the interquartile ranges are nearer.

iii.) Running time per iteration

Before starting analyzing the running time per iteration it is worth remembering that the total iterations of the SGM have been divided by the size of the matrix (125) in order to be able to compare algorithms.

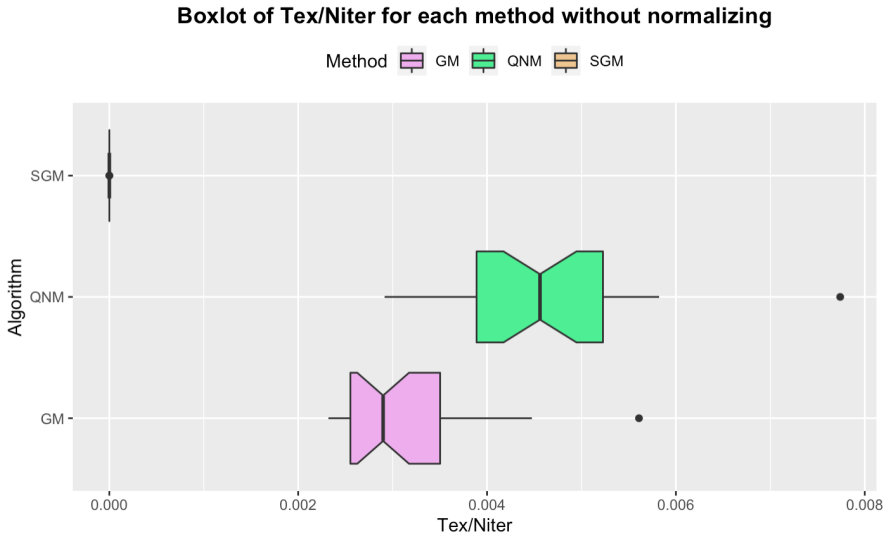


We clearly observe that the slowest algorithm in terms of iteration is the Quasi-Newton Method, BFGS. This is because at each iteration it approximates the Hessian matrix of the Loss function, resulting in heavy computations. Nonetheless, we have to take into account that it has been the fastest algorithm to find a stationary point among the three.

On the other hand, we see that the average time that lasts GM and SGM are very similar. Despite having a couple of outliers, the SGM method shows less variability. Thus, we can

state that SGM is more reliable than GM in terms of the time wasted per iteration, as the vast majority of its cases take similar time.

By last, if we hadn't divided the total iterations of the SGM method, we would observe that in terms of fastness is the best. This is because at each epoch it works with just a part of the training matrix, making the computations lighter than in the other algorithms. The graphic below shows what has been commented on.



Recall that this comparison is not entirely correct as the methods have distinct internal structures, but it has been made to highlight this fact.

c) General performance

Up to now it has been observed that the regularization parameter allows us to manipulate the Loss function in order to obtain the binary classification with different speeds of execution and levels of error. The greater the lambda, the worse optimal value of the objective function as there are harder restrictions that complicate the minimization of L. Furthermore, an increase on λ makes the objective function more convex. Thus, global convergence is achieved and the algorithms are able to find a solution in fewer steps but with more computational effort by iteration.

With regard to the number of iterations, the QNM has been stated as the method with fewer iterations because of its superlineal order of convergence. Despite both the GM and SGM having linear order of convergence, the GM lasts more iterations because it works with a bigger matrix.

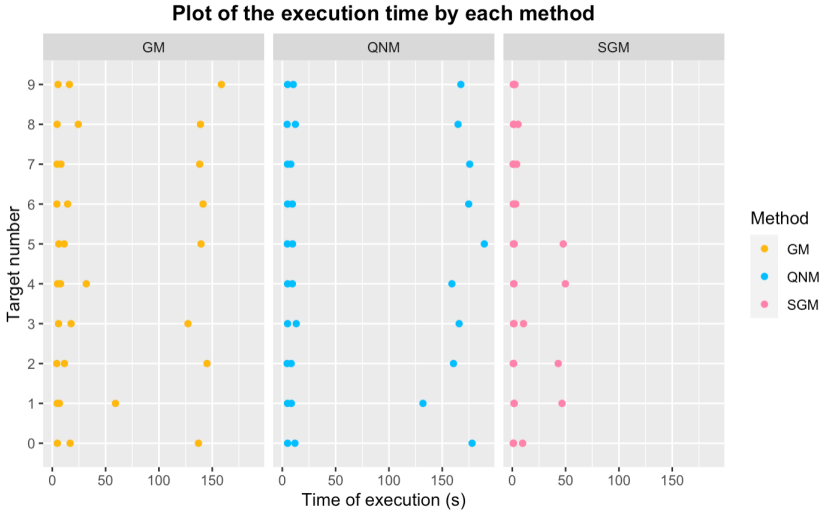
To select the most efficient algorithm for the minimization of L, we recall that QNM and SGM had similar time of execution. By looking at the minimization of the error of L, we have seen that SGM was the algorithm minimizing it, thus we have ended up with the choice of

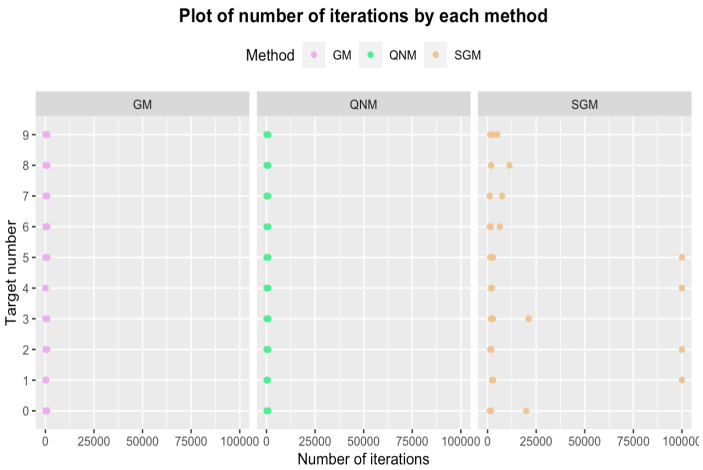
SGM as the one having better performance. In particular, we would chose this method using a regularization with $\lambda = 0.01$ or $\lambda = 0.1$ depending on how the increment of the error on L* is, as we've seen that those values reduced the mean time of execution.

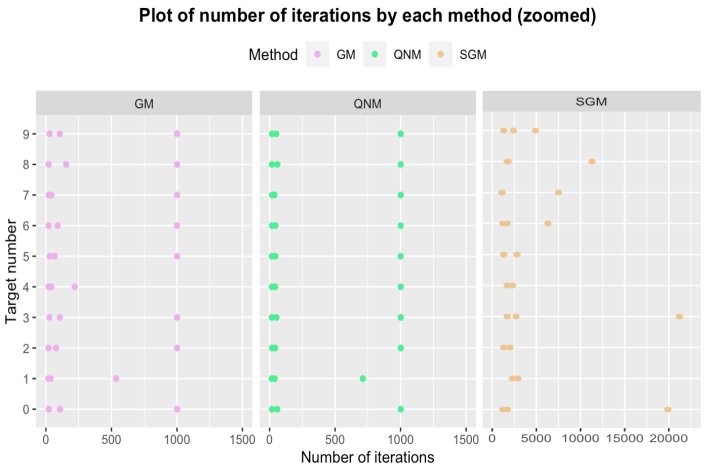
Study of the recognition accuracy

a) *Analysis of outperformance*

The main difference between this section and the first one is on the size of the training matrix. Now, *Xtr* has much more rows and columns, resulting in a notorious increase in the number of neurons of our Single Layer Neural Network. Thus, we are going to study if the algorithms behave the same way as in the previous case.

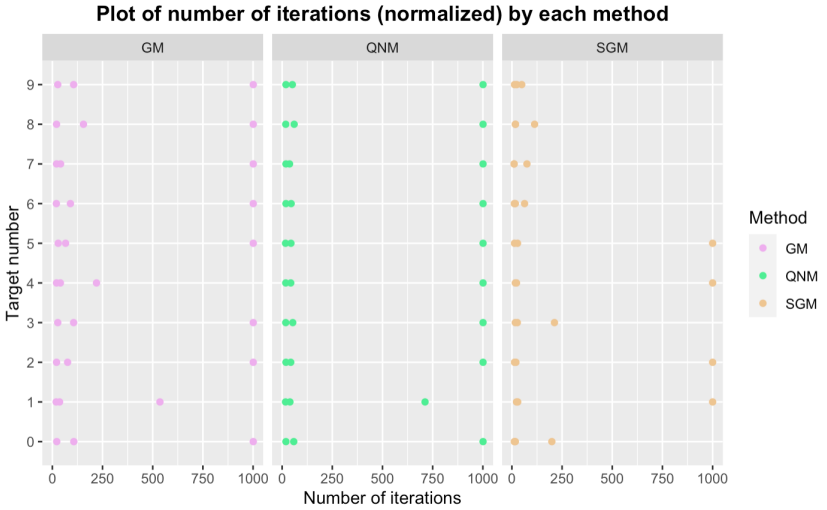






In contrast to what was seen before, we observe that the fastest algorithm when working with large matrices is SGM and not QNM. This is quite logical due to the fact that it works with mini batches.

On the other hand, we will need to rescale the number of iterations of SGM in order to be able to compare them with GM and QNM. Here, we will divide by sg_kmax/kmax = 100.

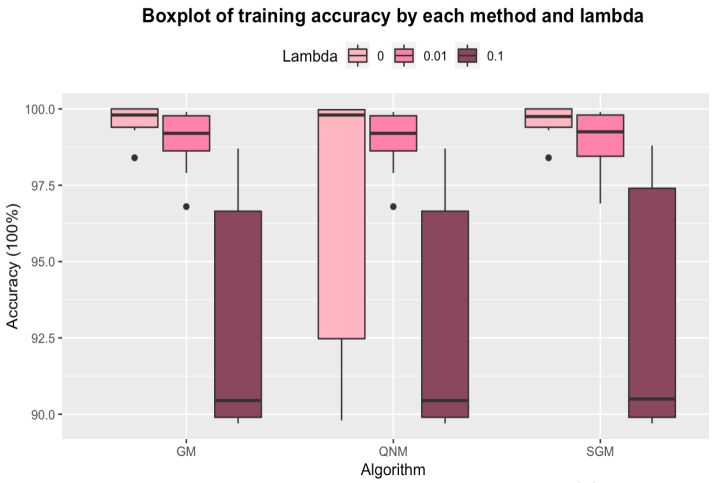


Giving us:

Algorithms	GM	QNM	SGM	
Mean number of iterations	327.17	345.9	16911	169.11
Mean time of execution (sec)	46.67	60.56	8.24	
Training accuracy (100%)	97.17	96.25	97.22	

Indeed, SGM is also the fastest one in terms of the number of iterations done, thus being the one having better performance. Moreover, we observe that it is the one converging in more cases, as fewer iterations reach the limit imposed (kmax = 1000).

The following plots show the accuracy of the methods used when classifying:



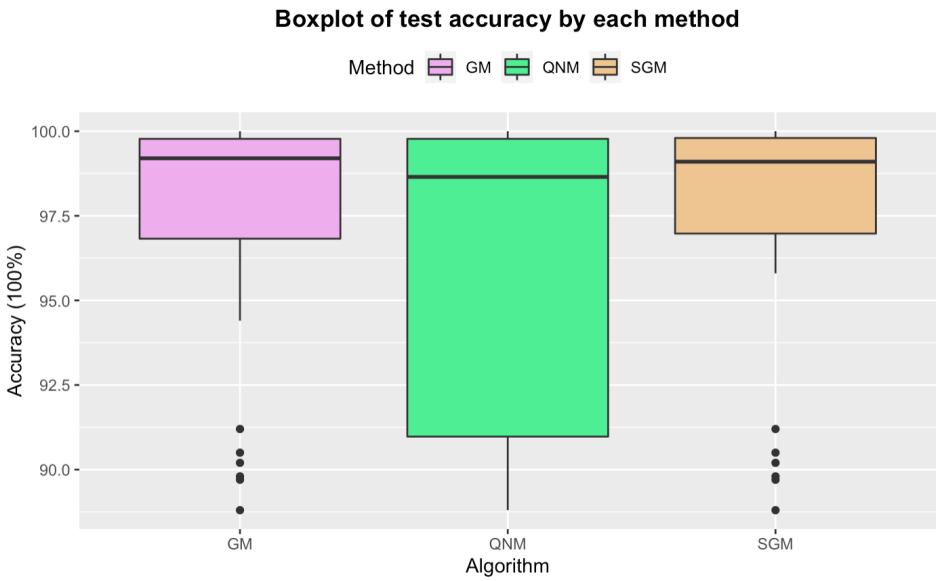
We see that all the methods present the same training accuracy for all lambda values except for the QNM with $\lambda = 0$. Therefore, we can state that when working with big matrices, all the algorithms hit the true value of the pattern much the same way. That's why, when choosing the best one, training accuracy won't be a crucial factor.

b) Minimization of L by λ values

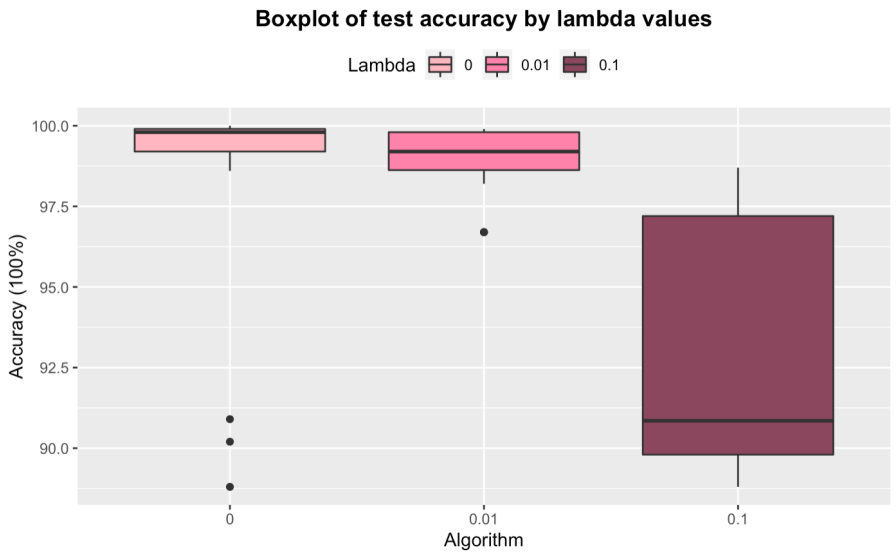
In this section we will study which is the best combination of lambda-algorithms with respect to the maximization of the test accuracy. Then, we will compare our conclusion with the one obtained in the previous study 1.c.

By the computations we've obtained:

Algorithms	**GM**	**QNM**	**SGM**
Test accuracy (100%)	97.17	96.25	97.22

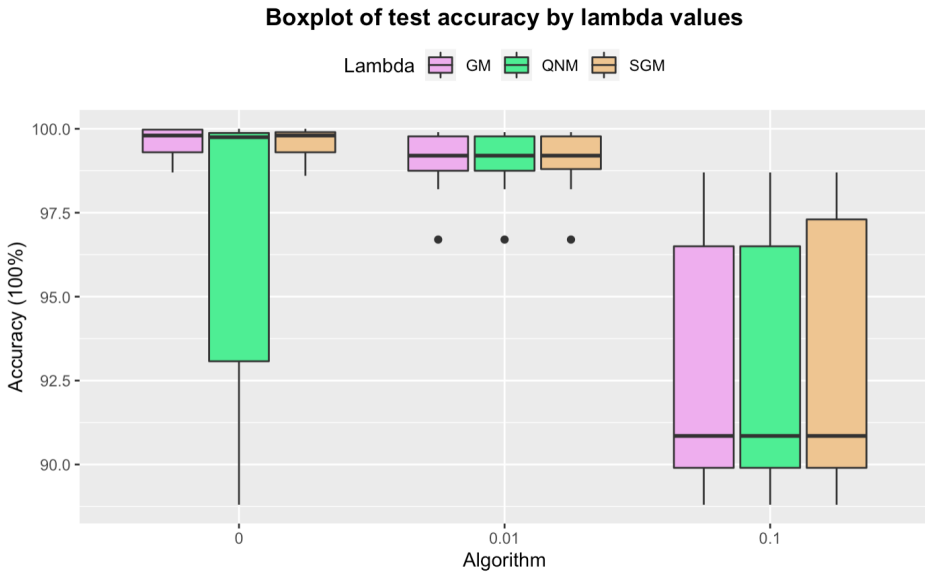


First of all, this chart shows the test accuracy depending on the method used. Comparing it with the previous exercise, it is clear that test accuracy and training accuracy behave the same way in our executions. It is seen that the GM is very similar to the SGM in terms of accuracy since the computation of the optimal in both methods is akin. The only difference is that the SGM works with smaller matrices. However, the SGM has greater accuracy. The QNM is the algorithm with least accuracy on average although the three algorithms report good results. Furthermore, the QNM has bigger variance among experiments because the interquartile ranges are further.



Secondly, looking at this chart that represents the test accuracy depending on the regularization parameter used, it is clear that the bigger lambda, the less test accuracy. This makes sense because increasing lambda means imposing harder restrictions on the objective function, which makes the algorithms find a worse optimal solution.

Furthermore, a big λ value implies a higher variance among experiments because the interquartile ranges are further.



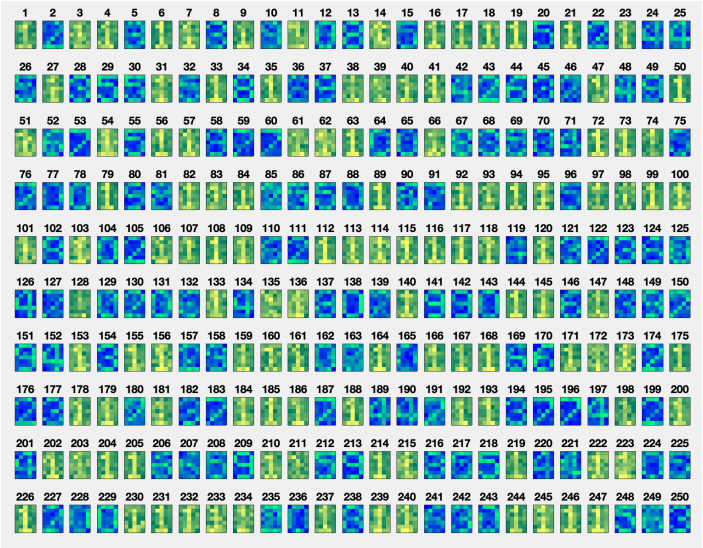
te_acc	GM	QNM	SGM
0.0	99.66	96.87	99.59
0.01	99.03	99.03	99.03
0.1	92.85	92.85	93.04

Last but not least, this chart joins all the above conclusions and allows the comparison between each lambda-algorithm combination. It is stated that the worst accuracy is due to the combination 0.1-QNM. By contrast, the best accuracy is attained without regularization (lambda = 0.0) and the GM algorithm despite the fact that the SGM is the best.

These conclusions coincide with the ones in the study 1.c because SGM is again the best algorithm. This time it is not just the best in terms minimization of L* but also in terms of fastness and maximization of test accuracy, allowing us to conclude SGM has been the best algorithm among GM, QNM and itself.

Lastly, the following plots are an example of the results of one case of the first part where the algorithm has guessed all numbers correctly and one that has made some mistakes.

CORRECT GUESSES



GUESSES WITH MISTAKES

Green -> True positive, Pink -> False positive,
Blue -> True negative, Red -> False negative