BACHELOR DEGREE IN DATA SCIENCE & ENGINEERING
UNIVERSITAT POLITÈCNICA DE CATALUNYA

UNCONSTRAINED OPTIMIZATION

LAB ASSIGNMENT
PATTERN RECOGNITION WITH SINGLE LAYER NEURAL NETWORK (SLNN)

REPORT

tr_seed: 4822010
te_seed: 21785256
sg_seed: 565544

Adrián Cerezuela Hernández
DNI: 48222010A
Ramon Ventura Navarro
DNI: 21785256R

INDEX

1. INTRODUCTION

In this project we are asked to develop an application, based on the unconstrained optimization algorithms studied in this course, GM, QNM and a new one called SGM, specifically, that allow us to recognize the numbers in a sequence of blurred digits through Single Layer Neural Network.
The first two parts consist on the development of the GM, QNM and SGM coding through Matlab. In this part, our purpose is to report and perform an analysis of all the results obtained.

Previously we have solved an instance of the SLNN problem through the given function *uo_nn_batch.m*, using a small dataset with tr_p = 250; te_q = 250 and tr_freq = 0.5. The results are in a csv file, and shown in the annexes of this document.

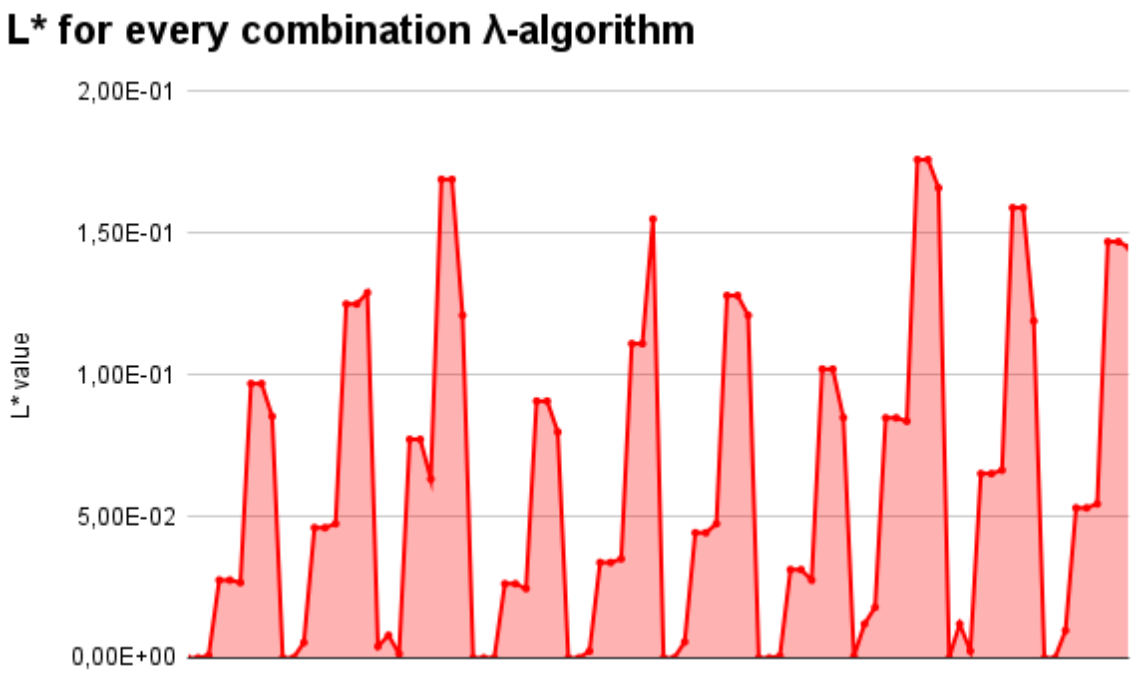
2. STUDY OF THE CONVERGENCE

In this first part, we are going to study the global and local convergence of the three algorithms only in terms of the objective function $\tilde{L}$.

a) Global convergence

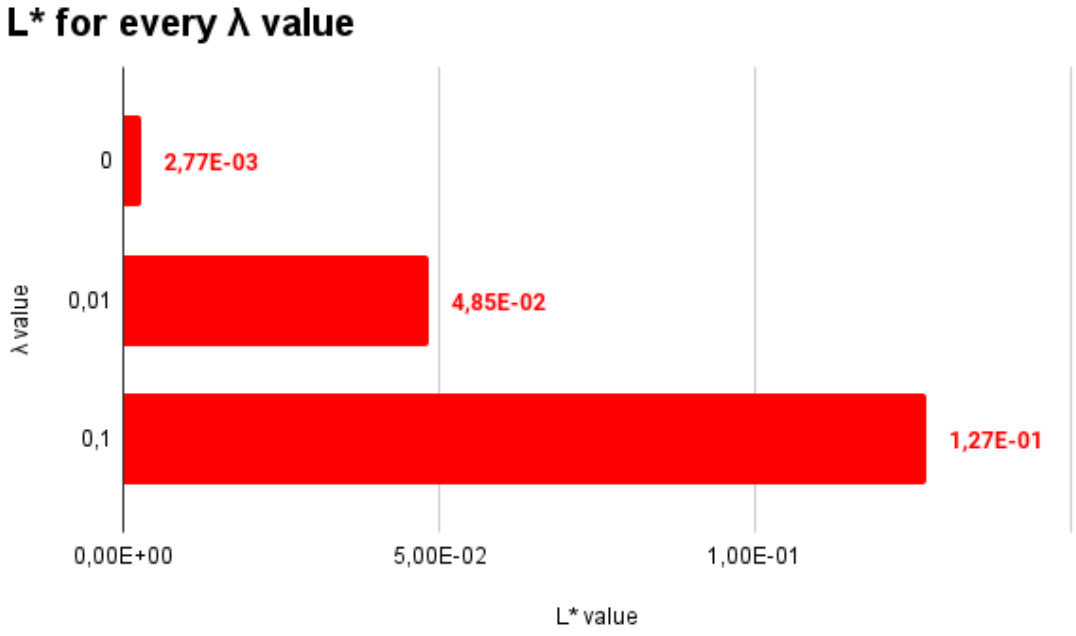
In this section we are asked to compare the value of the loss function $\tilde{L}$ at the optimal solution for every λ value and for every algorithm. Once we perform the dataset given through the function *uo_nn_batch.m* we obtain a csv file where the data is stored (see annexes).

Our analysis will be carried out both graphically and analytically:



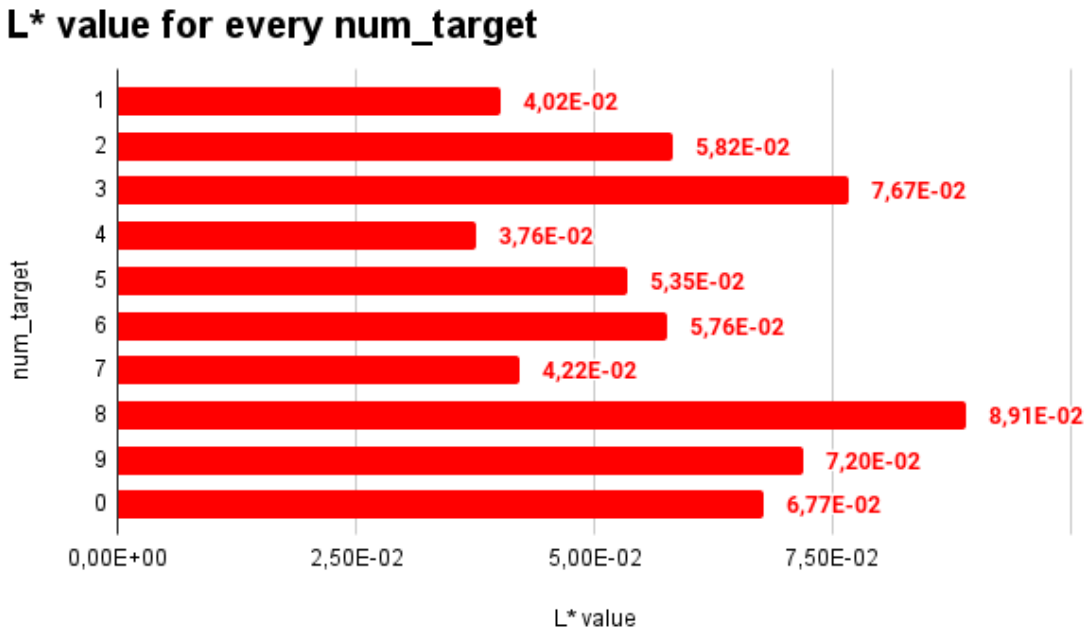
Here we can see a comparison of every single value for the loss function, where the order followed is the same as in the csv file: first by num_target, then by λ value and finally by method applied (GM, QNM or SGM). So, for every target nine consecutive values of $\tilde{L}$ are represented, three for $\lambda = 0$, three for $\lambda = 0.01$ and three for $\lambda = 0.1$ (GM, QNM and SGM), in that order. Every loss function peak corresponds to one target.

In terms of analysis, we want the minimum value for $\tilde{L}$ (i.e. we want the difference between the value obtained by the model and the data value to be the less possible). Firstly, we our hypothesis is that this value reaches its peak for every target when $\lambda = 0.1$ and its lowest peak when $\lambda = 0$. If we perform an average of $\tilde{L}$ for every λ value we obtain the next graphic:

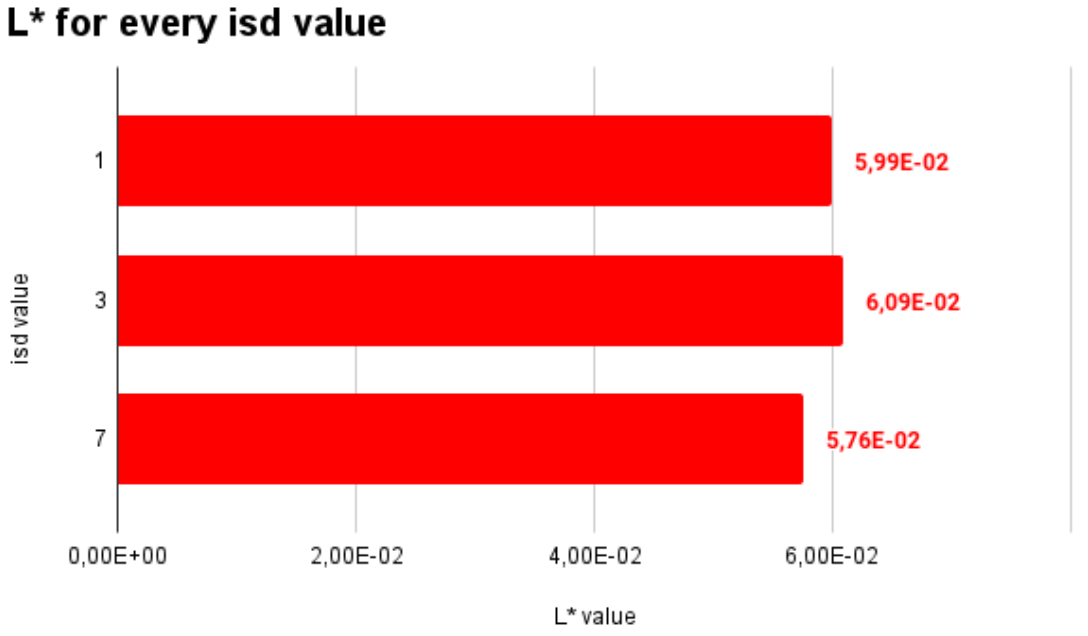


As we can see, our hypothesis was true, and the loss function value increases as the λ increases.

Taking this result into account and going back to the first graphic, we can observe that the loss function reaches its peak if the target is the number 8 (1,76E-01), having high values for targets 3 and 9 too. Also, on the other hand, we have the lowest peak (0,00E+00) if the target is number 7. These numbers are obtained by performing the minimum and maximum formulas on the loss function column on the csv file.
But do these peaks mean that targeting 8 and 7 respectively entails more or less difference between values?
This can be easily seen if we perform an average of every $\tilde{L}$ value for each target:



As it could be expected, number 8 is the target which has more difference between the model value and the data value. But as we can note from this second graphic, having the loss function peak doesn't necessarily mean that it is the target that "loses more information". It is the case of number 7. We have seen before that it has the lowest peak, but in general targeting number 4 makes us lose less information than any other target.



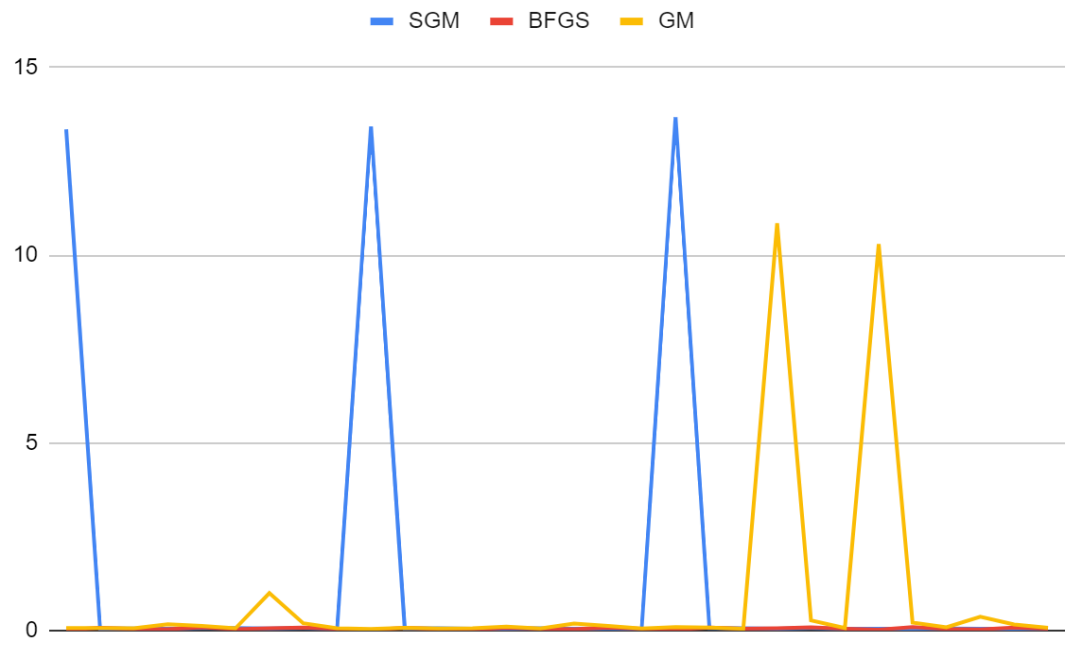
Finally, we compare how large is the $\tilde{L}$ value for every method. As we have done before for the lambda, we perform an average for every isd and we obtain that although all three methods have similar values for the loss function, the SGM seem to be the better one in this aspect, with the lowest one.

In terms of convergence, we can assume that GM and QNM converge globally if the number of iterations is less than 1000, and SGM converge globally as long as it finds a solution, regardless of the number of iterations.
With this information and according to the data stored in the log file, we can assume that:

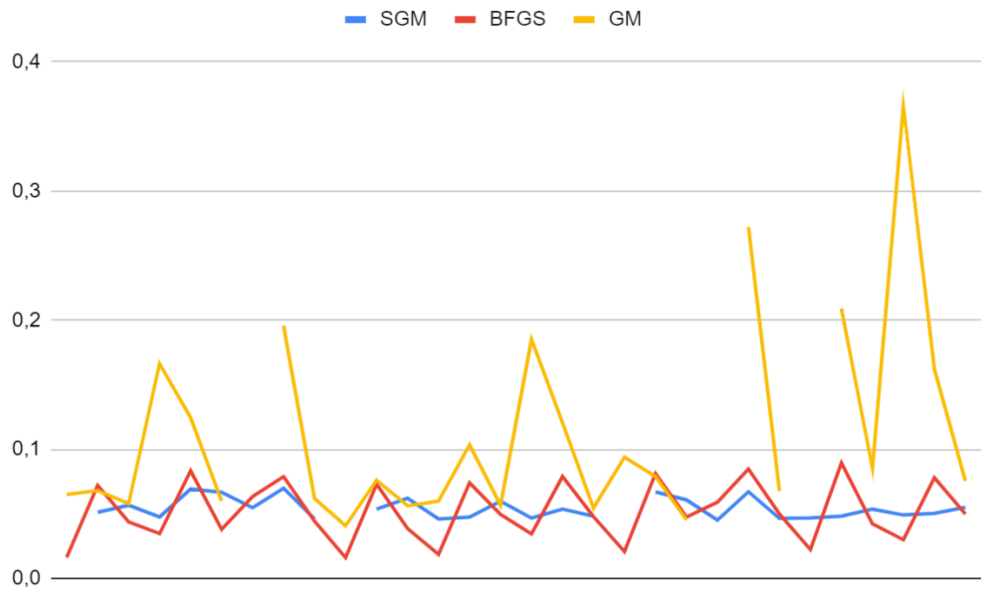
- SGM and QNM methods converge globally for all our cases.
- GM converges globally for every case except if we target number 3, number 8 or number 9, all with $\lambda = 0$.

b) Local convergence

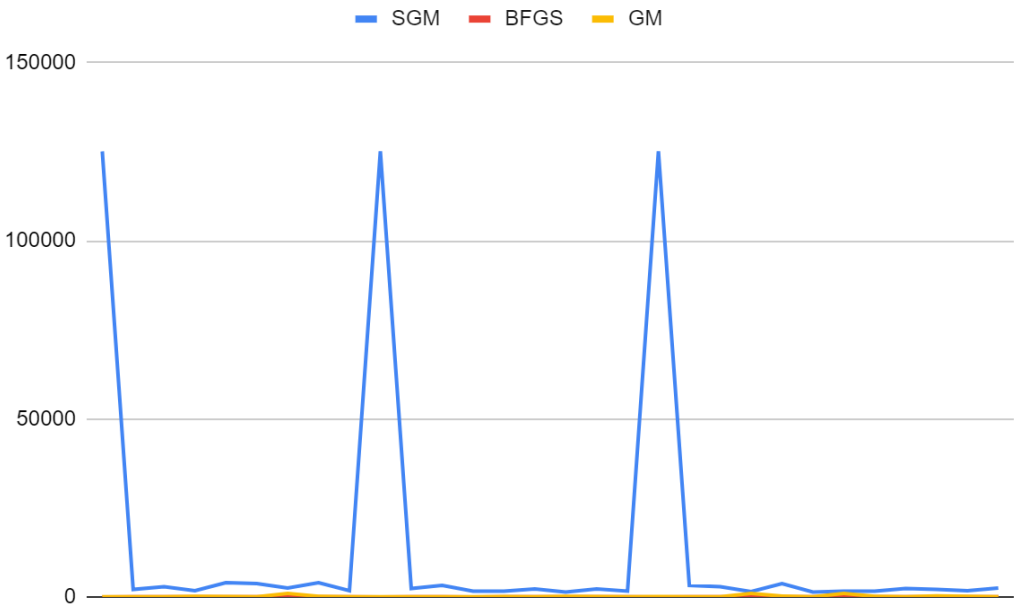
i. When taking a look to execution times, we can see in the chart some large times, which happen to be non-convergent executions.



If we remove those values, we can see in the new chart below how now SGM seems to be more stable than BFGS. However, due to the randomness of the chosen seeds and the different values of λ, we can see that in some cases BFGS is more efficient than any other algorithm. We can also discard GM as our candidate to reaching convergence the fastest because, pretty much for any execution, it shows the worst performance of them all.

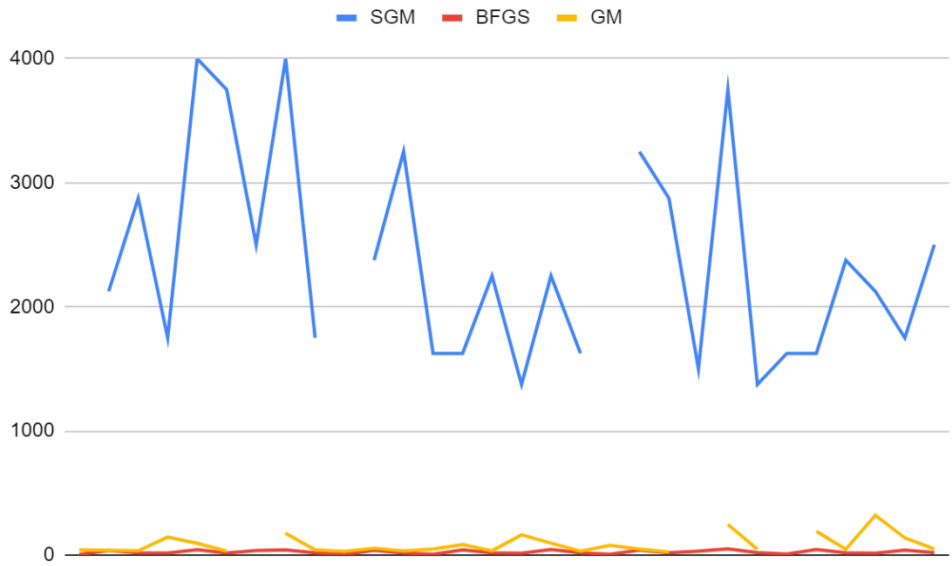


Now lets follow the same study along the number of iterations. As we saw before, non-convergent executions are being represented as large spikes. In this case, we can notoriously see SGM does not converge three times.



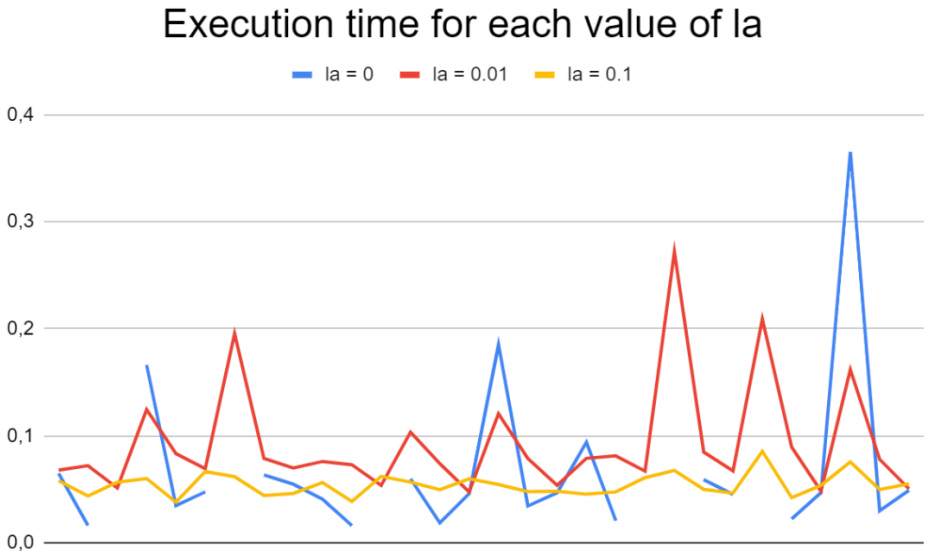
Deleting non-convergent executions, we can conclude that SGM is clearly the method that needs the most iterations with a big difference and, BFGS

the least. This being added to what we concluded earlier, we can definitely say that BFGS is the most efficient when referring to execution time and iterations, being the method with more speed of convergence, lowest execution time and lowest number of iterations.



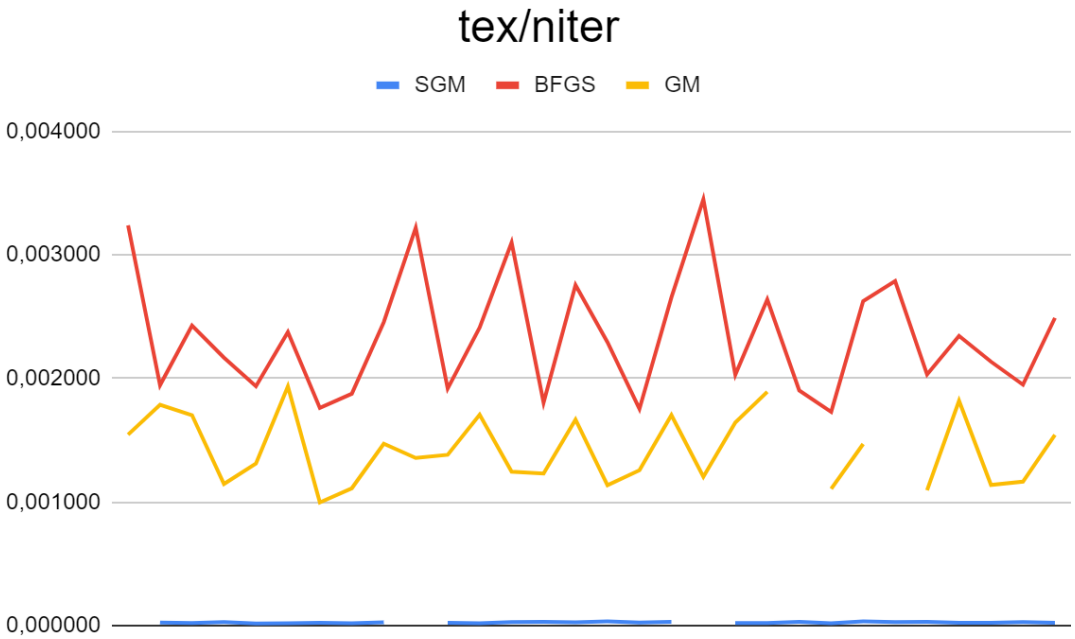
From now on, we will plot each chart only with convergent realizations.

ii. When plotting the execution time and the three different values of λ, we can conclude that $\lambda = 0.1$ is realistically the fastest one. This is because, even though for $\lambda = 0$ shows a better execution time, and we could say the higher λ the lower time of execution, this value does not always assure global convergence so choosing it wouldn't be the best answer.



iii. Finally, analyzing the running time per iteration for each of the methods, we can clearly see how SGM has the lowest *tex/niter* rate. This occurs because as we saw in class, each SGM iteration gets α and d way faster than GM and BFGS.

On the other hand, BFGS seems to be the algorithm with slowest running time per iteration, followed closely by GM. These two last algorithms are slower due to the fact that they use line search to calculate α, and additionally, the fact of using H matrix in BFGS makes it slower as it has to be calculated every iteration.



c) Discussion

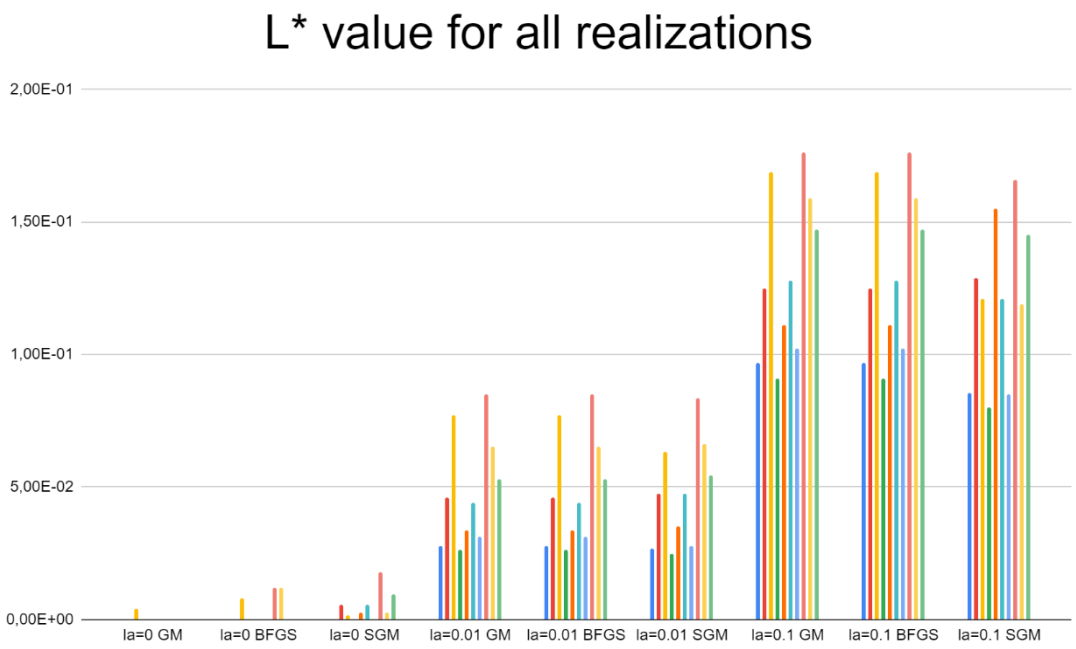
Let's discuss the general performance of all three algorithms.

Starting with GM, it is easy to see that it does not ensure global convergence for every instance and is clearly the most inefficient and slowest in terms of assuring local convergence.

BFGS, instead, seems to always provide a good global convergence for all values of λ. Moreover, we can say it has a similar local convergence to GM. Both methods clearly show near to zero values for L* as we can see in the chart below.

In conclusion, we can say SGM appears to be the best algorithm when referring to local convergence but does not outperform the other two methods on global convergence.

Also, if we had to choose a λ, we would definitely choose $\lambda = 0.01$, as it does not give such a bigger error L* as in $\lambda = 0.1$ and it ensures global convergence, unlike $\lambda = 0$. This conclusion can be deducted from the following chart.



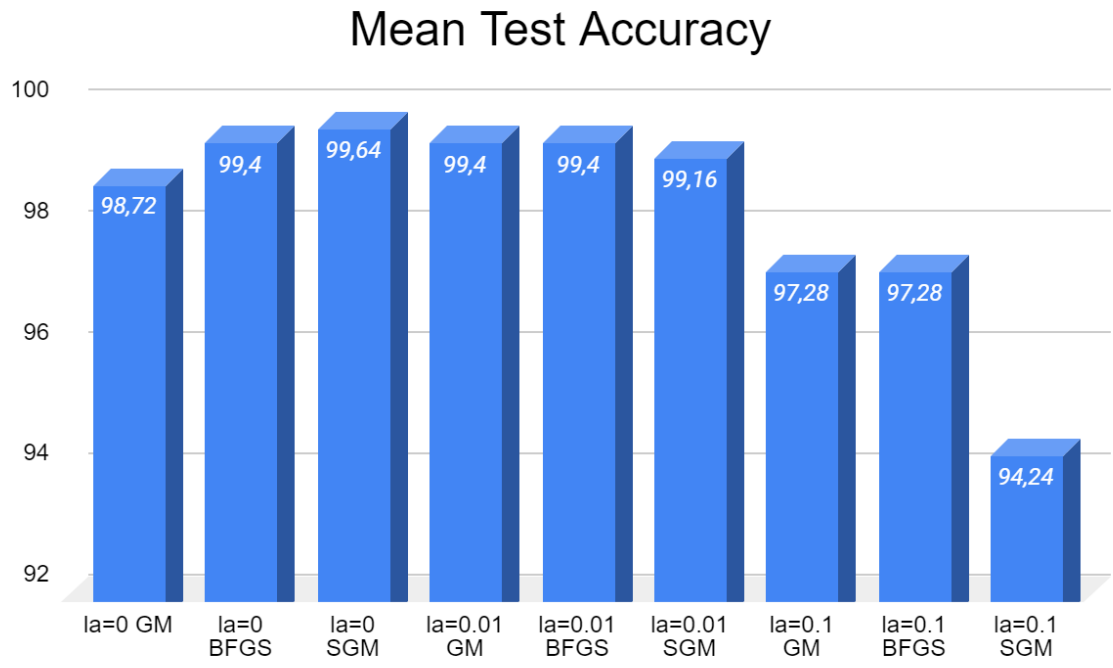
Finally, once analyzed each of the aspects for every algorithm, our chosen method for the most efficient minimization of L* is SGM for $\lambda = 0.01$, seems to clearly outperform GM and is better enough than BFGS.

3. STUDY OF THE RECOGNITION ACCURACY

In this second part, we are going to analyze the recognition accuracy of the SLNN for the different algorithmic options. Now we will perform our experiment with a larger dataset with tr_p = 20000; te_q = tr_p /10; tr_freq = 0.0.

Although we have run the function *uo_nn_batch.m* with all three values for λ, we will carry out our analysis with the values showing the best accuracy ($Accuracy^{TE}$) for each method.

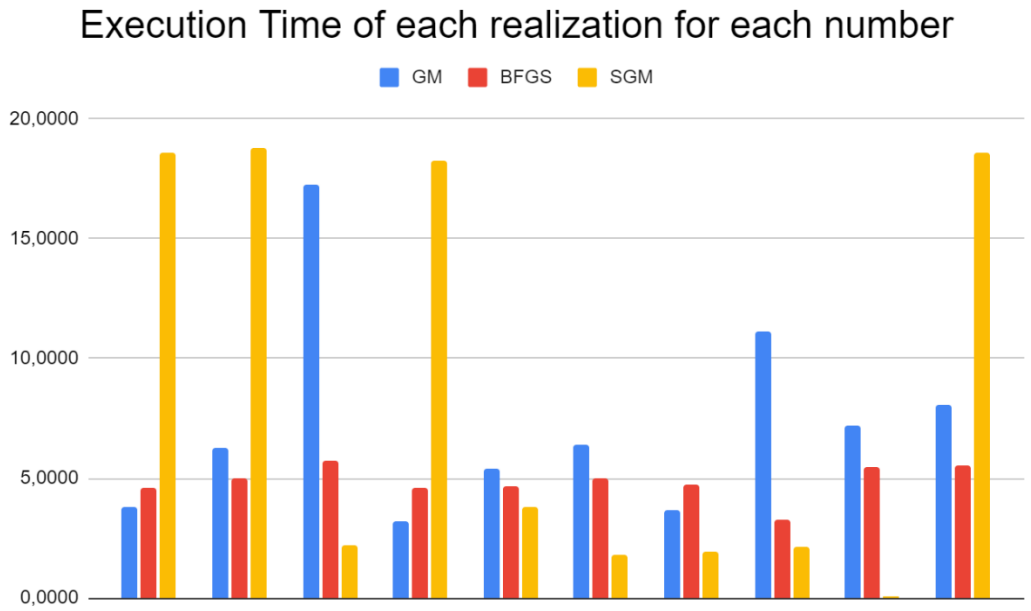
Basing our conclusions in the following chart, which represents the average mean $Accuracy^{TE}$ of each realization for the previous dataset, we deduct SGM for $\lambda = 0$, GM for $\lambda = 0.01$ and BFGS for $\lambda = 0.01$ are the best realizations. This last one is actually drew with it's same method for $\lambda = 0$, but we choose the other realization over this one as $\lambda = 0.01$ shows better properties and grants us global convergence.



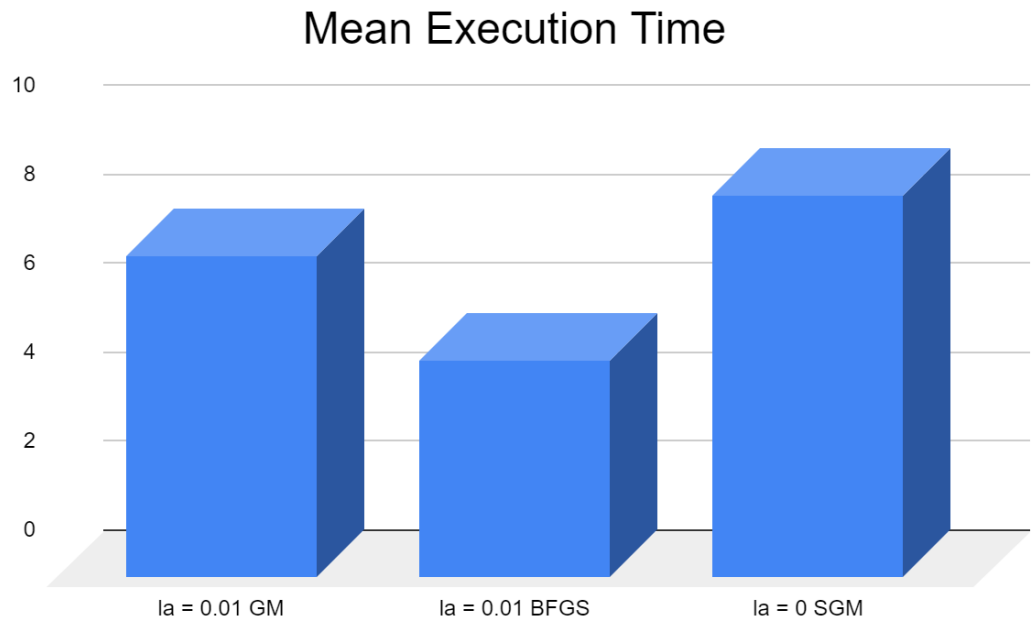
num_target	la	isd	niter	tex	tr_acc	te_acc	L*
1	0.0100	1	36	38.173	99.8	99.8	2.14e-02
1	0.0100	3	34	45.917	99.8	99.8	2.14e-02
1	0.0000	7	100100	185.579	100.0	100.0	1.82e-04
2	0.0100	1	69	62.745	99.1	99.5	4.02e-02
2	0.0100	3	38	49.778	99.1	99.5	4.02e-02
2	0.0000	7	100100	187.987	99.7	99.8	1.52e-03
3	0.0100	1	73	172.710	97.9	97.9	5.16e-02
3	0.0100	3	44	57.353	97.9	97.9	5.16e-02
3	0.0000	7	10000	22.301	99.2	99.2	7.14e-03
4	0.0100	1	28	32.043	99.9	100.0	2.21e-02
4	0.0100	3	35	46.314	99.9	100.0	2.21e-02
4	0.0000	7	100100	182.231	100.0	100.0	5.63e-05
5	0.0100	1	56	53.937	99.7	99.7	2.98e-02
5	0.0100	3	36	46.940	99.7	99.7	2.98e-02
5	0.0000	7	19100	37.985	100.0	100.0	5.43e-04
6	0.0100	1	69	64.015	99.3	99.5	3.66e-02
6	0.0100	3	39	50.197	99.3	99.5	3.66e-02
6	0.0000	7	7800	17.878	99.8	99.7	2.38e-03
7	0.0100	1	34	36.500	99.8	99.8	2.51e-02
7	0.0100	3	36	47.530	99.8	99.8	2.51e-02
7	0.0000	7	8500	19.116	99.9	100.0	4.57e-04
8	0.0100	1	129	111.420	97.1	97.0	6.21e-02
8	0.0100	3	45	329.486	97.1	97.0	6.21e-02
8	0.0000	7	9900	21.238	98.6	98.7	1.05e-02
9	0.0100	1	79	72.297	98.4	98.8	4.95e-02
9	0.0100	3	41	54.396	98.4	98.8	4.95e-02
9	0.0000	7	3300	0.9810	99.3	99.0	6.59e-03
0	0.0100	1	93	80.641	98.9	98.7	4.25e-02
0	0.0100	3	43	55.391	98.9	98.7	4.25e-02
0	0.0000	7	100100	185.948	99.7	99.8	1.98e-03

a) Best method for large problems

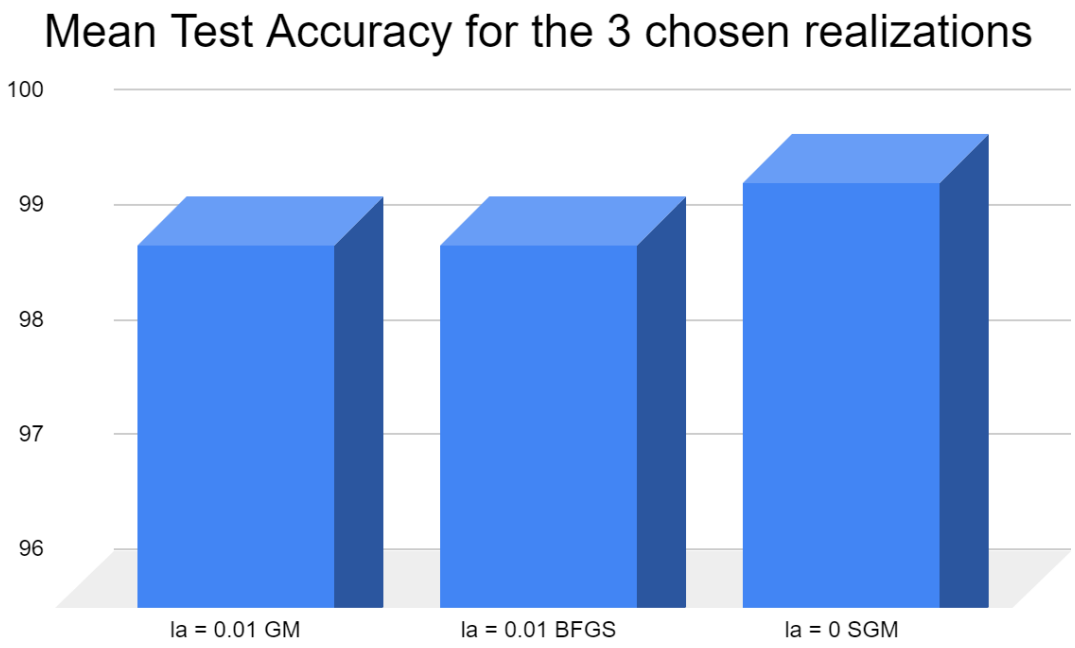
When taking a look to the execution time of each method for this new dataset, we can conclude that SGM is the fastest and outperforms the other two methods in the majority of cases but due to $\lambda = 0$, it doesn't guarantee global convergence, which explains the 4 largest amounts of time shown in the following chart.



Due to the same fact ($\lambda = 0$), the mean execution time of SGM looks overvalued.



Mean Test Accuracy, however, shows SGM has the best test accuracy. Also, for all of the aspects studied GM seems to be worse than BFGS. Leaving BFGS in an intermediate spot and having a tricky better mean execution time than SGM but again, it is due to non-convergence of SGM.



b) Discussion $Accuracy^{TE}$-$\tilde{L}$

We can say methods don not coincide. Method chosen in the first exercise was SGM $\lambda = 0.01$ which showed the best minimization of L*, whereas in this second exercise we have chosen SGM $\lambda = 0$ which maximizes the test accuracy.
The reason they are not the same realization is because, the fact that they have different λ makes them different, fluctuating the test accuracy. Also, the fact that we aren't assuring global convergence in this second realization affects. As explained earlier, choosing $\lambda = 0.01$ is safer beacuse it assures global convergence, but if what we want is a better test accuracy, $\lambda = 0$ would be the best answer.

4. ANNEXES

a) Results of the performance of the experiment that corresponds to part 1:

num_target	la	isd	niter	tex	tr_acc	te_acc	L*
1	0	1	42	0,0649	100.0	100.0	4,20E-07
1	0	3	5	0,0162	100.0	100.0	8,02E-23
1	0	7	125125	13.353	100.0	100.0	8,39E-04
1	0,01	1	38	0,0679	100.0	100.0	2,75E-02
1	0,01	3	37	0,072	100.0	100.0	2,75E-02
1	0,01	7	2125	0,0512	99.6	99.6	2,66E-02
1	0,1	1	34	0,0579	100.0	100.0	9,69E-02
1	0,1	3	18	0,0437	100.0	100.0	9,69E-02
1	0,1	7	2875	0,0566	96.0	97.6	8,53E-02
2	0	1	145	0,1662	100.0	99.2	8,51E-07
2	0	3	16	0,0347	100.0	99.6	5,84E-08
2	0	7	1750	0,0476	100.0	99.6	5,47E-03
2	0,01	1	95	0,1245	100.0	99.6	4,60E-02
2	0,01	3	43	0,0833	100.0	99.6	4,60E-02
2	0,01	7	4000	0,0692	98.8	99.6	4,74E-02
2	0,1	1	31	0,06	95.2	92.0	1,25E-01
2	0,1	3	16	0,038	95.2	92.0	1,25E-01
2	0,1	7	3750	0,0666	48.8	84.8	1,29E-01
3	0	1	1000	0,9964	99.6	98.0	4,07E-03
3	0	3	36	0,0634	99.2	99.2	8,00E-03
3	0	7	2500	0,0547	98.0	99.6	1,44E-03
3	0,01	1	176	0,1956	98.0	99.6	7,72E-02
3	0,01	3	42	0,0788	98.0	99.6	7,72E-02
3	0,01	7	4000	0,0698	90.4	100.0	6,32E-02
3	0,1	1	42	0,0618	96.8	97.2	1,69E-01
3	0,1	3	18	0,0442	96.8	97.2	1,69E-01
3	0,1	7	1750	0,0461	52.0	93.2	1,21E-01
4	0	1	30	0,0407	100.0	100.0	5,23E-07
4	0	3	5	0,0161	100.0	100.0	2,80E-23
4	0	7	125125	13.432	100.0	100.0	2,91E-05
4	0,01	1	55	0,076	100.0	100.0	2,62E-02
4	0,01	3	38	0,0729	100.0	100.0	2,62E-02
4	0,01	7	2375	0,0536	100.0	100.0	2,45E-02
4	0,1	1	33	0,0563	100.0	100.0	9,07E-02
4	0,1	3	16	0,0386	100.0	100.0	9,07E-02
4	0,1	7	3250	0,0621	97.2	99.2	7,98E-02

5	0	1	48	0,0598	100.0	99.6	5,75E-07
5	0	3	6	0,0186	100.0	99.2	3,82E-79
5	0	7	1625	0,046	100.0	100.0	2,40E-03
5	0,01	1	84	0,1034	100.0	100.0	3,37E-02
5	0,01	3	41	0,074	100.0	100.0	3,37E-02
5	0,01	7	1625	0,0475	98.8	99.6	3,50E-02
5	0,1	1	34	0,0567	99.6	100.0	1,11E-01
5	0,1	3	18	0,0496	99.6	100.0	1,11E-01
5	0,1	7	2250	0,0598	99.2	99.6	1,55E-01
6	0	1	163	0,1852	100.0	99.2	8,60E-07
6	0	3	15	0,0344	100.0	99.6	2,83E-20
6	0	7	1375	0,0467	100.0	99.6	5,68E-03
6	0,01	1	96	0,1207	100.0	99.6	4,42E-02
6	0,01	3	45	0,0789	100.0	99.6	4,42E-02
6	0,01	7	2250	0,0537	99.2	99.6	4,74E-02
6	0,1	1	32	0,0545	99.6	99.2	1,28E-01
6	0,1	3	18	0,0478	99.6	99.2	1,28E-01
6	0,1	7	1625	0,0482	54.4	92.8	1,21E-01
7	0	1	78	0,0939	100.0	100.0	5,32E-07
7	0	3	6	0,0207	100.0	100.0	0,00E+00
7	0	7	125125	13.673	100.0	100.0	6,78E-04
7	0,01	1	48	0,0789	100.0	100.0	3,12E-02
7	0,01	3	40	0,0812	100.0	100.0	3,12E-02
7	0,01	7	3250	0,067	96.0	99.6	2,75E-02
7	0,1	1	24	0,0454	100.0	99.6	1,02E-01
7	0,1	3	18	0,0475	100.0	99.6	1,02E-01
7	0,1	7	2875	0,0608	98.0	99.2	8,49E-02
8	0	1	1000	10.849	100.0	96.0	5,58E-04
8	0	3	31	0,059	98.8	98.0	1,20E-02
8	0	7	1500	0,0452	96.4	98.4	1,79E-02
8	0,01	1	246	0,2721	98.4	97.2	8,48E-02
8	0,01	3	49	0,0847	98.4	97.2	8,48E-02
8	0,01	7	3750	0,0671	84.4	96.0	8,36E-02
8	0,1	1	46	0,0676	90.8	88.0	1,76E-01
8	0,1	3	19	0,0499	90.8	88.0	1,76E-01
8	0,1	7	1375	0,0465	52.4	89.2	1,66E-01

9	0	1	1000	10.296	100.0	96.4	6,71E-06
9	0	3	8	0,0223	98.8	99.2	1,20E-02
9	0	7	1625	0,0468	98.8	100.0	2,47E-03
9	0,01	1	191	0,2091	98.4	98.8	6,51E-02
9	0,01	3	44	0,0894	98.4	98.8	6,51E-02
9	0,01	7	1625	0,0482	91.6	98.4	6,63E-02
9	0,1	1	47	0,0855	96.8	97.6	1,59E-01
9	0,1	3	18	0,0422	96.8	97.6	1,59E-01
9	0,1	7	2375	0,0536	51.6	89.2	1,19E-01
0	0	1	321	0,3653	100.0	98.8	1,16E-06
0	0	3	14	0,0299	100.0	99.2	4,55E-09
0	0	7	2125	0,0491	100.0	99.2	9,68E-03
0	0,01	1	139	0,1619	99.2	99.2	5,30E-02
0	0,01	3	40	0,078	99.2	99.2	5,30E-02
0	0,01	7	1750	0,0503	99.2	99.2	5,44E-02
0	0,1	1	49	0,0756	99.2	99.2	1,47E-01
0	0,1	3	20	0,0498	99.2	99.2	1,47E-01
0	0,1	7	2500	0,0551	92.8	97.6	1,45E-01

b) Results of the performance of the experiment that corresponds to part 2:

num_target	la	isd	niter	tex	tr_acc	te_acc	L*
1	0.0000	1	288	219.897	100.0	100.0	2.10e-06
1	0.0000	3	40	50.087	100.0	100.0	2.00e-11
1	0.0000	7	100100	185.579	100.0	100.0	1.82e-04
1	0.0100	1	36	38.173	99.8	99.8	2.14e-02
1	0.0100	3	34	45.917	99.8	99.8	2.14e-02
1	0.0100	7	1800	0.7166	99.9	99.8	2.12e-02
1	0.1000	1	22	28.461	97.7	98.0	7.00e-02
1	0.1000	3	17	27.843	97.7	98.0	7.00e-02
1	0.1000	7	1500	0.6764	98.1	98.3	7.06e-02
2	0.0000	1	1000	1.040.279	99.8	99.8	2.02e-03
2	0.0000	3	15	19.343	99.4	99.3	6.00e-03
2	0.0000	7	100100	187.987	99.7	99.8	1.52e-03
2	0.0100	1	69	62.745	99.1	99.5	4.02e-02
2	0.0100	3	38	49.778	99.1	99.5	4.02e-02
2	0.0100	7	1100	0.6046	99.0	99.4	3.91e-02
2	0.1000	1	21	28.367	89.7	90.7	9.70e-02
2	0.1000	3	17	26.775	89.7	90.7	9.70e-02
2	0.1000	7	2300	0.9157	89.7	90.7	9.41e-02
3	0.0000	1	1000	790.653	99.2	99.2	6.45e-03
3	0.0000	3	160	190.975	99.2	99.2	6.43e-03
3	0.0000	7	10000	22.301	99.2	99.2	7.14e-03
3	0.0100	1	73	172.710	97.9	97.9	5.16e-02
3	0.0100	3	44	57.353	97.9	97.9	5.16e-02
3	0.0100	7	1100	0.6403	97.5	97.7	4.99e-02
3	0.1000	1	35	41.817	89.9	91.7	1.06e-01
3	0.1000	3	17	28.018	89.9	91.7	1.06e-01
3	0.1000	7	1100	0.6047	89.9	91.7	9.83e-02
4	0.0000	1	299	425.829	100.0	100.0	1.77e-06
4	0.0000	3	24	33.027	100.0	100.0	5.00e-05
4	0.0000	7	100100	182.231	100.0	100.0	5.63e-05
4	0.0100	1	28	32.043	99.9	100.0	2.21e-02
4	0.0100	3	35	46.314	99.9	100.0	2.21e-02
4	0.0100	7	2600	0.8507	99.9	100.0	2.22e-02
4	0.1000	1	23	28.805	98.9	98.8	7.42e-02
4	0.1000	3	17	26.905	98.9	98.8	7.42e-02
4	0.1000	7	2600	0.8495	98.8	98.8	7.50e-02

5	0.0000	1	1000	757.557	100.0	99.9	2.00e-04
5	0.0000	3	49	58.854	100.0	100.0	3.50e-04
5	0.0000	7	19100	37.985	100.0	100.0	5.43e-04
5	0.0100	1	56	53.937	99.7	99.7	2.98e-02
5	0.0100	3	36	46.940	99.7	99.7	2.98e-02
5	0.0100	7	1700	0.6879	99.7	99.8	3.09e-02
5	0.1000	1	25	31.084	95.1	93.8	8.97e-02
5	0.1000	3	18	28.710	95.1	93.8	8.97e-02
5	0.1000	7	1800	0.7204	96.1	95.1	9.38e-02
6	0.0000	1	1000	758.038	99.8	99.7	1.28e-03
6	0.0000	3	151	172.974	99.9	99.7	1.10e-03
6	0.0000	7	7800	17.878	99.8	99.7	2.38e-03
6	0.0100	1	69	64.015	99.3	99.5	3.66e-02
6	0.0100	3	39	50.197	99.3	99.5	3.66e-02
6	0.0100	7	1200	0.6138	99.3	99.4	3.64e-02
6	0.1000	1	29	34.709	90.2	90.8	9.62e-02
6	0.1000	3	19	29.518	90.2	90.8	9.62e-02
6	0.1000	7	2200	0.7874	90.2	90.8	9.51e-02
7	0.0000	1	1000	916.001	100.0	99.9	2.76e-04
7	0.0000	3	84	97.037	100.0	99.9	4.00e-04
7	0.0000	7	8500	19.116	99.9	100.0	4.57e-04
7	0.0100	1	34	36.500	99.8	99.8	2.51e-02
7	0.0100	3	36	47.530	99.8	99.8	2.51e-02
7	0.0100	7	1400	0.6619	99.8	99.8	2.58e-02
7	0.1000	1	19	25.771	97.0	96.2	7.58e-02
7	0.1000	3	17	27.425	97.0	96.2	7.58e-02
7	0.1000	7	1600	0.7069	97.3	96.6	7.79e-02
8	0.0000	1	1000	765.998	98.6	98.6	1.10e-02
8	0.0000	3	1000	1.365.503	89.8	89.6	9.75e-02
8	0.0000	7	9900	21.238	98.6	98.7	1.05e-02
8	0.0100	1	129	111.420	97.1	97.0	6.21e-02
8	0.0100	3	45	329.486	97.1	97.0	6.21e-02
8	0.0100	7	2800	0.9590	97.1	97.2	6.15e-02
8	0.1000	1	31	36.382	89.8	89.6	1.14e-01
8	0.1000	3	18	28.282	89.8	89.6	1.14e-01
8	0.1000	7	1800	0.7137	89.8	89.6	1.14e-01

9	0.0000	1	1000	749.671	99.4	99.0	5.32e-03
9	0.0000	3	1000	1.352.691	99.1	98.8	8.25e-03
9	0.0000	7	3300	0.9810	99.3	99.0	6.59e-03
9	0.0100	1	79	72.297	98.4	98.8	4.95e-02
9	0.0100	3	41	54.396	98.4	98.8	4.95e-02
9	0.0100	7	1100	0.5939	98.3	98.8	4.80e-02
9	0.1000	1	20	26.009	90.0	89.6	1.07e-01
9	0.1000	3	16	26.088	90.0	89.6	1.07e-01
9	0.1000	7	4000	11.054	90.0	89.6	1.09e-01
0	0.0000	1	1000	762.559	99.7	99.7	2.30e-03
0	0.0000	3	131	160.845	99.8	99.8	2.10e-03
0	0.0000	7	100100	185.948	99.7	99.8	1.98e-03
0	0.0100	1	93	80.641	98.9	98.7	4.25e-02
0	0.0100	3	43	55.391	98.9	98.7	4.25e-02
0	0.0100	7	2700	0.8766	98.9	98.7	4.24e-02
0	0.1000	1	22	27.466	89.7	89.9	1.04e-01
0	0.1000	3	18	28.619	89.7	89.9	1.04e-01
0	0.1000	7	2100	0.7740	89.7	89.9	1.04e-01