Optimization Techniques for Data Mining: Unconstrained Optimization

Pietro Fronte, Carolina Jorge Centeio

October 2018

1 Introduction

The assignment was about building a shallow Neural Network (just input layer and output layer) and train it with different dataset and optimization algorithms studied during the Unconstrained Optimization part of the course:

1. Steepest Descent Gradient
2. Conjugate Gradient Fletcher-Reeves
3. Conjugate Gradient Polak-Ribière
4. Quasi-Newton method BFGS
5. Quasi-Newton method DFP

The values of two input independent variable X_1 and X_2 are fed into the two neurons of the input layer. Each neuron is equipped with the Sigmoid function as activation function defined as follow:

$$o_i = \sigma(x_i) = \frac{1}{1 + e^{-x_i}}, \quad i = 1, 2$$

The outputs of these neurons are now values between 0 and 1. They can be expressed as O = $\{o_1, o_2\}$.

These values are then weighted and summed giving a new input value for a new neuron (in another layer, the second one, which in this project is the output layer):

$$X_{n+1} = X_3 = \sum_{i=1}^{2} w_i * O_i = w_1 * o_1 + w_2 * o_2$$

The final output (from the output layer) will make use of another Sigmoid function, returning y (which is assumed to me binary).

$$y(x, w) = \sigma(X_3) = \frac{1}{1 + e^{-X_3}} = \frac{1}{1 + e^{-\sum_{i=1}^{2} w_i * O_i}}$$

As a usual ML problem we need a function to measure how good our classification is. In this case, we will use the Loss function for the training dataset. Since we aim to minimize the loss function, it would be our objective function to be optimized with respect to the parameter w. It can be defined as follows:

$$L(X^{TR}, Y^{TR}) = \min_{w \epsilon \mathbb{R}^2} L(w; X^{TR}, Y^{TR}) = \sum_{j=1}^{p} (y(X^{TR}, w) - Y^{TR})^2,$$

where p is the size of the dataset.

However, in order to increase the convexity of the function and to facilitate the convergence of the algorithms, we modified slightly the objective function, adding a new parameter (not to be optimized):

$$\tilde{L}(w; X^{TR}, Y^{TR}, \lambda) = L(X^{TR}, Y^{TR}) + \lambda \frac{||w||^2}{2}$$

Thus, this is the true objective function we aim to minimize, called Loss function with $L2$ regularization with parameter λ.

2 Results on basic classification

Results obtained running the algorithms of varying the linesearch method. We can notice that Quasi-Newton algorithms with fminbnd lineasearch outperformed the other algorithms candidate.

r	tr_p	tr_seed	la	w1(1)	w1(2)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	L*	tr_acc	te_acc	te_q	te_seed
1	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	GM	0	756	-9.35e+01	+7.61e+01	5.771e-06	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	GM	0	186	-9.37e+01	+7.62e+01	5.555e-06	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	534	-9.39e+01	+7.64e+01	5.372e-06	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	0	401	-9.30e+01	+7.56e+01	6.436e-06	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	0	347	-9.35e+01	+7.60e+01	5.837e-06	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	0	236	-9.30e+01	+7.57e+01	6.322e-06	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	BFGS	2	3	-5.36e+00	+2.48e+00	6.888e+01	74.8	74.9	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	6	-9.95e+03	+7.70e+03	0.000e+00	100.0	100.0	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	DFP	2	3	-5.35e+00	+2.48e+00	6.888e+01	74.8	74.9	50000	7891016
1	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	DFP	0	6	-1.08e+03	+8.57e+02	4.143e-97	100.0	100.0	50000	7891016

Figure 1: Results on classification rule 1 with $\lambda = 0$.

Results obtained adding regularization parameter to the objective function.

r	tr_p	tr_seed	la	w1(1)	w1(2)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	L*	tr_acc	te_acc	te_q	te_seed
1	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	GM	0	471	-1.36e+01	+1.09e+01	2.930e+01	100.0	100.0	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	GM	1	801	-1.36e+01	+1.09e+01	2.930e+01	100.0	100.0	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	289	-1.36e+01	+1.09e+01	2.930e+01	100.0	100.0	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	177	-1.36e+01	+1.09e+01	2.930e+01	100.0	100.0	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	BFGS	2	3	-5.16e+00	+2.19e+00	7.416e+01	74.8	74.9	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	9	-1.36e+01	+1.09e+01	2.930e+01	100.0	100.0	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	DFP	2	3	-5.15e+00	+2.18e+00	7.416e+01	74.8	74.9	50000	7891016
1	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	DFP	0	8	-1.36e+01	+1.09e+01	2.930e+01	100.0	100.0	50000	7891016

Figure 2: Results on classification rule 1 with $\lambda = 0$.

3 First part

The first part of the assignment required the training of the NN on a training dataset built as follow

$$x_i \epsilon \{-1, 1\}, \quad y = \begin{cases} 1 \text{ if } x_2{=}x_1 \\ 0 \text{ otherwise} \end{cases} \qquad i = 1, 2$$

Notice that when building the training dataset X^{TR} we are also building the response for each training datapoint Y^{TR}.

Notice that all the attempts done follow this rules:

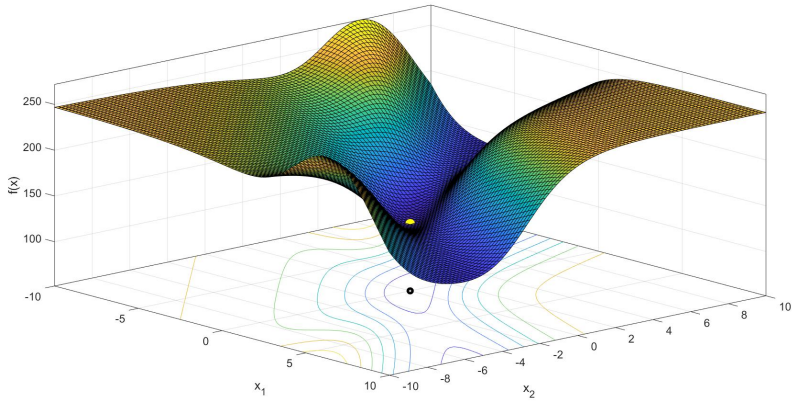
1. size training dataset: 500
2. seed training dataset : 1234566
3. size test dataset: 50000
4. seed test dataset: 7891016
5. starting point w_0={0,0}

Starting with a λ=0 (meaning no regularization of the objective function) the results obtained can be seen in Fig 3 and the respective plot in Fig 4:

r	tr_p	tr_seed	la	w1(1)	w1(2)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	L*	tr_acc	te_acc	te_q	te_seed
2	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	GM	0	13	-4.40e-03	-4.40e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	GM	0	3	+5.59e+02	+5.59e+02	2.520e+02	49.6	50.1	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	15	-4.40e-03	-4.40e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	0	3	+5.59e+02	+5.59e+02	2.520e+02	49.6	50.1	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	5	-4.40e-03	-4.40e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	0	3	+5.59e+02	+5.59e+02	2.520e+02	49.6	50.1	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	8	-4.40e-03	-4.40e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	3	+5.59e+02	+5.59e+02	2.520e+02	49.6	50.1	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	DFP	0	8	-4.40e-03	-4.40e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	DFP	0	3	+5.59e+02	+5.59e+02	2.520e+02	49.6	50.1	50000	7891016

Figure 3: Results of function Optimization with $\lambda = 0$.

Figure 4: Plot of function Optimization with $\lambda = 0$.

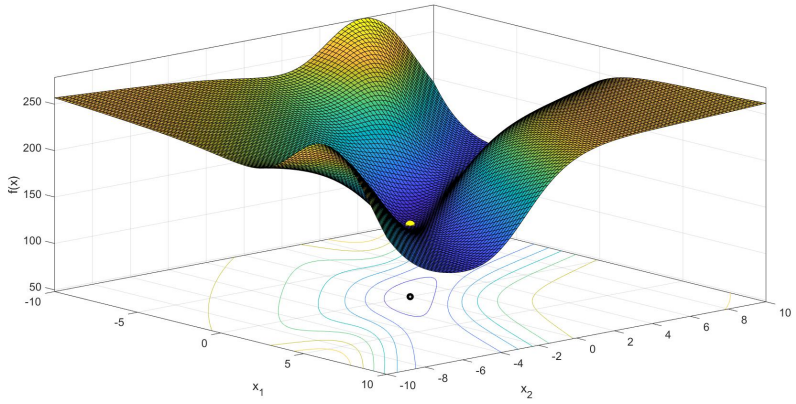


Moving to λ=0.1 the results can be seen in Fig 5 and the respective plot in Fig 6:

Figure 5: Results of function Optimization rule 2 with λ=0.1.

r	tr_p	tr_seed	la	w1(1)	w1(2)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	L*	tr_acc	te_acc	te_q	te_seed
2	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	GM	1	801	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	GM	0	3	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	71	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	0	3	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	5	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	0	3	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	9	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	3	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	DFP	0	9	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016
2	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	DFP	0	3	-4.39e-03	-4.39e-03	1.250e+02	50.4	49.9	50000	7891016

Figure 6: Plot of function Optimization rule 2 with λ=0.1.



3.1 Comments on results of First Part

In the not normalized problem all the algorithms behaved very well, even converging after just 3 iterations implementing the fminbnd line search. However not all of them manage to catch an universal minimum of the function..in particular the algorithms implemented with fminbnd stop always in W* = {+5.59e+0.2, +5.59e+0.2} while the algorithms that use backtracking line search converge in W* = {-4.40e-0.3, -4.40e-0.3}

The normalized problem show a universal point of convergence. All the algorithms, even with different line search methods and different time, catch the optimal point.

4 Second part

Since in the first part of this Assignment none of the algorithms used managed to achieve an acceptable value of test accuracy (a 50% in test accuracy it means that it is just a random classification), we tried to add more information in the input variables (a third feature), hoping to get better results.

The training (and test) dataset this time is built as follows:

$$x_i \epsilon \{-1, 1\}, \quad x_3 = x_2 * x_1, \quad y = \begin{cases} 1 \text{ if } x_2{=}x_1 \\ 0 \text{ otherwise} \end{cases} \qquad i = 1, 2, 3$$

This new problem leads to some changes in the previous set-up. Now we have a third neuron on the first layer (input layer), resulting on three outputs as follows:

$$o_i = \sigma(x_i) = \frac{1}{1 + e^{-x_i}}, \quad O = \{o_1, o_2, o_3\}$$

And the new input variable of the output layer will then be:

$$X_4 = \sum_{i=1}^{3} w_i * O_i = w_1 * o_1 + w_2 * o_2 + w_3 * o_3$$

Keeping the same objective function and algorithms the results obtained can be seen in Fig 12 and Fig 14 without regularization ($\lambda = 0$) and with regularization ($\lambda = 1$), respectively.

r	tr_p	tr_seed	la	w1(1)	w1(2)	w1(3)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	w*(3)	L*	tr_acc	te_acc	te_q	te_seed
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	GM	0	424	-2.73e+01	-2.74e+01	+6.72e+01	3.602e-06	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	GM	0	237	-2.74e+01	-2.74e+01	+6.73e+01	3.529e-06	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	167	-2.72e+01	-2.73e+01	+6.70e+01	3.880e-06	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	0	133	-2.73e+01	-2.73e+01	+6.71e+01	3.752e-06	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	194	-2.61e+01	-2.62e+01	+6.45e+01	7.905e-06	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	0	131	-2.71e+01	-2.72e+01	+6.67e+01	4.109e-06	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	14	-4.13e+01	-4.07e+01	+1.00e+02	4.619e-10	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	10	-3.92e+01	-3.27e+01	+8.77e+01	5.716e-08	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	DFP	0	22	-3.63e+02	-1.97e+02	+7.37e+02	3.924e-36	100.0	100.0	50000	7891016
21	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	DFP	0	10	-3.92e+01	-3.26e+01	+8.74e+01	6.285e-08	100.0	100.0	50000	7891016

Figure 7: Results of function Optimization with λ=0.

r	tr_p	tr_seed	la	w1(1)	w1(2)	w1(3)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	w*(3)	L*	tr_acc	te_acc	te_q	te_seed
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	GM	0	248	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	GM	0	216	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	148	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	0	53	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	0	82	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	0	199	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	17	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	10	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	DFP	0	16	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016
21	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	DFP	0	10	-5.36e+00	-5.43e+00	+1.31e+01	1.894e+01	100.0	100.0	50000	7891016

Figure 8: Results of function Optimization with λ=0.1.

Due to the dimensions of the problem, it is no longer possible to plot it.

4.1 Comments on second results

The first thing that pops up looking at the first table is that there is no optimal solution w:w_1,w_2,w_3 equals to any other in the table. Almost all the algorithms catch the optimal solution finally getting a 100% accuracy in the test set but there is no universal optimal solution.

Adding convexity to the problem (moving λ to 0.1) we fixed this issue. Now all the algorithm are able to converge, find the optimal solution in w:-5.36, -5.43, 13.1 and get a 100% accuracy in test set. Again the Quasi-Newton show their performance against the competitors.

5 Third part

In this classification rule we are adding more variability in the training dataset since we are allowing the variable X to assume also the 0-value. In details the new dataset is built as follow:

$$x_i \epsilon \{-1, 0, 1\}, \quad y = \begin{cases} 1 \text{ if } x_2 + x_1 > 0 \\ 0 \text{ otherwise} \end{cases} \qquad i = 1, 2$$

As said before with a greater freedom given to X, we expect a lower accuracy running our previous optimization methods

r	tr_p	tr_seed	la	w1(1)	w1(2)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	L*	tr_acc	te_acc	te_q	te_seed
3	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	GM	0	40	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	1	801	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	3	-6.38e-01	-4.15e-01	1.151e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	2	27	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	19	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	5	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	1	0.5	DFP	0	14	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016
3	500	1234566	0.000	+0.00e+00	+0.00e+00	2	0.5	DFP	0	5	-1.16e+00	-8.38e-02	1.146e+02	71.8	68.5	50000	7891016

Figure 9: Results of function Optimization rule 3 with λ=0.

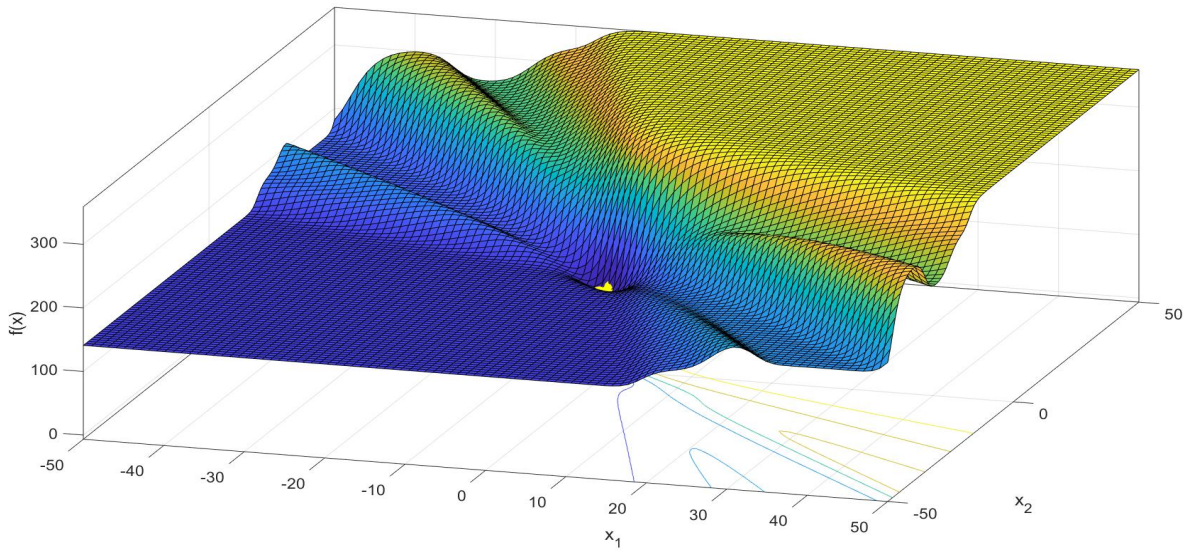


Figure 10: Plot function Optimization rule 3 with λ=0.

r	tr_p	tr_seed	la	w1(1)	w1(2)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	L*	tr_acc	te_acc	te_q	te_seed
3	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	GM	0	36	-1.13e+00	-1.15e-01	1.147e+02	71.8	68.5	50000	7891016
3	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	0	26	-1.13e+00	-1.15e-01	1.147e+02	71.8	68.5	50000	7891016
3	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	3	-6.55e-01	-4.39e-01	1.150e+02	71.8	68.5	50000	7891016
3	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	15	-1.13e+00	-1.15e-01	1.147e+02	71.8	68.5	50000	7891016
3	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	5	-1.13e+00	-1.15e-01	1.147e+02	71.8	68.5	50000	7891016
3	500	1234566	0.100	+0.00e+00	+0.00e+00	1	0.5	DFP	0	11	-1.13e+00	-1.15e-01	1.147e+02	71.8	68.5	50000	7891016
3	500	1234566	0.100	+0.00e+00	+0.00e+00	2	0.5	DFP	0	5	-1.13e+00	-1.15e-01	1.147e+02	71.8	68.5	50000	7891016

Figure 11: Results of function Optimization rule 3 with λ=0.1.

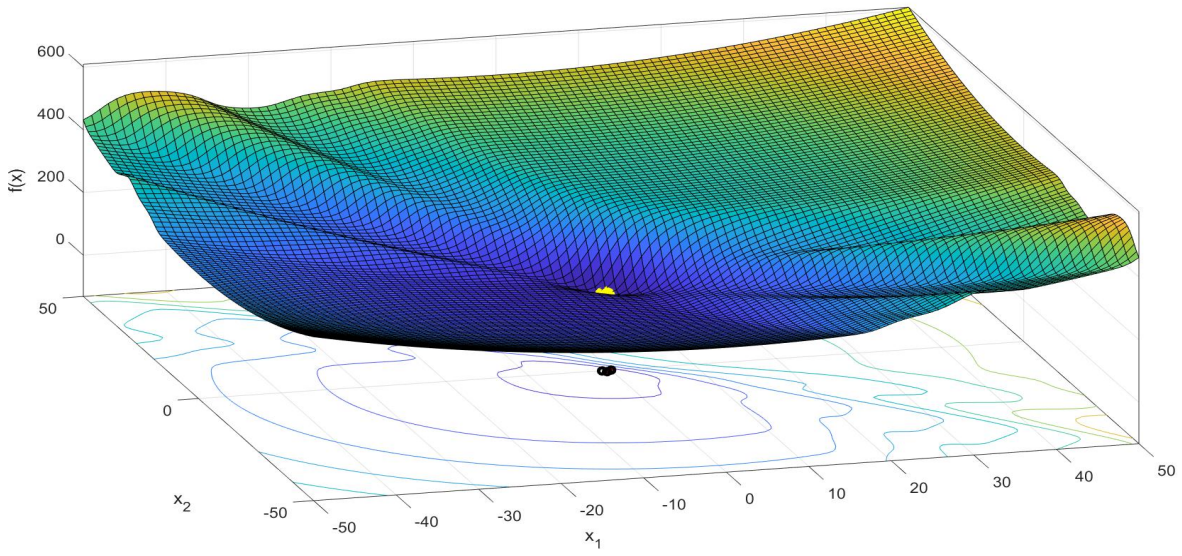


Figure 12: Plot function Optimization rule 3 with λ=0.1.

Now the true task asked..try to improve the accuracy with the new classification rule adding a new neuron x_3. The goal here is declare the value of x_3 in order to pass to the NN enough information to carry out a better classification. Our dataset was then built in this way:

$$x_i \epsilon \{-1, 0, 1\}, \quad i = 1, 2 \quad x_3 = \begin{cases} 1 \text{ if } x_2{}^* x_1 >= 0 \\ 0 \text{ otherwise} \end{cases} \qquad y = \begin{cases} 1 \text{ if } x_2 + x_1 > 0 \\ 0 \text{ otherwise} \end{cases}$$

r	tr_p	tr_seed	la	w1(1)	w1(2)	w1(3)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	w*(3)	L*	tr_acc	te_acc	te_q	te_seed
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	GM	1	801	+3.09e+01	+7.48e+00	-2.83e+01	4.631e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	GM	1	801	+5.24e+01	+1.08e+01	-4.57e+01	3.700e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	1	801	+5.25e+01	+1.08e+01	-4.57e+01	3.698e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	1	801	+1.09e+02	+2.01e+01	-9.19e+01	2.693e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	2	-4.89e-01	-5.42e-01	-1.12e+00	1.131e+02	71.4	68.6	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	1	801	+5.08e+01	+1.06e+01	-4.44e+01	3.756e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	10	+4.50e+03	+8.66e+02	-3.85e+03	2.500e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	9	+5.71e+02	+1.01e+02	-4.76e+02	2.500e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	DFP	0	10	+3.57e+04	+7.32e+03	-3.12e+04	2.500e+01	95.0	93.9	50000	7891016
32	500	1234566	0.000	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	DFP	0	9	+6.25e+02	+1.10e+02	-5.22e+02	2.500e+01	95.0	93.9	50000	7891016

Figure 13: Results of function Optimization rule 31 with λ=0.

r	tr_p	tr_seed	la	w1(1)	w1(2)	w1(3)	ls	c2	sdm	iout	iter	w*(1)	w*(2)	w*(3)	L*	tr_acc	te_acc	te_q	te_seed
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	GM	1	801	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-FR	3	576	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-FR	0	41	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	CGM-PR	2	2	-4.89e-01	-5.42e-01	-1.12e+00	1.132e+02	71.4	68.6	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	CGM-PR	2	37	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	BFGS	0	13	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	BFGS	0	10	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	1	0.5	DFP	0	15	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016
32	500	1234566	0.100	+0.00e+00	+0.00e+00	+0.00e+00	2	0.5	DFP	0	10	+7.46e+00	+3.61e+00	-8.96e+00	7.208e+01	77.2	75.2	50000	7891016

Figure 14: Results of function Optimization rule 31 with λ=0.1.

5.1 Comments on third results

According to the first table of results (lambda =0) the accuracy obtained seems very high but looking at the other info we notice that GM and the CG methods were not able to find any interesting result within 800 iterations. Quasi-Newton did better but there is a lot variability among the coordinates of the optimal point W*.

Adding convexity to the problem (moving λ to 0.1) things change. Most of the algorithms converged, find the optimal solution in w:+7.46, 3.61, -8.96 increasing the test accuracy from a previous 68.5% without third neuron to a 75.2%. Again the Quasi-Newton showed their strength in reaching the optimal with any line search and in few iterations.