HW2-REPORT

Qiu Wei

March 8, 2017

Contents

1 MLQP Analyse

1.1 Forward

$$x_i^k = f^k(n_i^k)$$

$$n_i^k = b_i^k + \sum_{j=1}^{N^{k-1}} (u_{i,j}^k (x_j^{k-1})^2 + v_{i,j}^k x_j^{k-1})$$

1.2 Backpropagation

Let:

$$s_i^k = \frac{\partial F}{\partial n_i^k}$$

Then:

$$u_{i,j}^k(l+1) = u_{i,j}^k(l) - \alpha s_i^k \frac{\partial n_i^k}{\partial u_{i,j}^k} = u_{i,j}^k(l) - \alpha s_i^k (x_j^{k-1})^2$$

$$v_{i,j}^k(l+1) = v_{i,j}^k(l) - \alpha s_i^k \frac{\partial n_i^k}{\partial v_{i,j}^k} = v_{i,j}^k(l) - \alpha s_i^k x_j^{k-1}$$

$$b_{i,j}^k(l+1) = b_{i,j}^k(l) - \alpha s_i^k \frac{\partial n_i^k}{\partial b_{i,j}^k} = b_{i,j}^k(l) - \alpha s_i^k$$

Now calculate s^k :

$$s^k = \frac{\partial F}{\partial n^k} = \frac{\partial F}{\partial n^{k+1}} \frac{\partial n^{k+1}}{\partial n^k} = s^{k+1} \frac{\partial n^{k+1}}{\partial n^k}$$

To calculate the Jacobbi's matrix

$$J^k = \frac{\partial n^{k+1}}{\partial n^k}$$

We have:

$$J_{i,j}^k = \frac{\partial n_i^{k+1}}{\partial n_j^k} = \frac{\partial (b_i^{k+1} + \sum_{l=1}^{N^k} (u_{i,l}^{k+1} (x_l^k)^2 + v_{i,l}^{k+1} x_l^k))}{\partial n_j^k} = (2x_j^k u_{i,j}^{k+1} + v_{i,j}^{k+1}) \frac{\partial x_j^k}{\partial n_j^k}$$

At last, calculate $\mathrm{s^m}$:

$$s_i^m = \frac{\partial \sum_{j=1}^{N^m} (t_j - x_j^m)^2}{\partial n_i^m} = 2(x_i^m - t_i) \frac{\partial x_i^m}{\partial n_i^m}$$

Let

$$\dot{F^k}_{i,j} = \begin{cases} \dot{f}^k(x_i^k), & i = j \\ 0, & i \neq j \end{cases}$$

$$X_{i,j}^k = \begin{cases} x_i^k, & i = j \\ 0, & i \neq j \end{cases}$$

Then

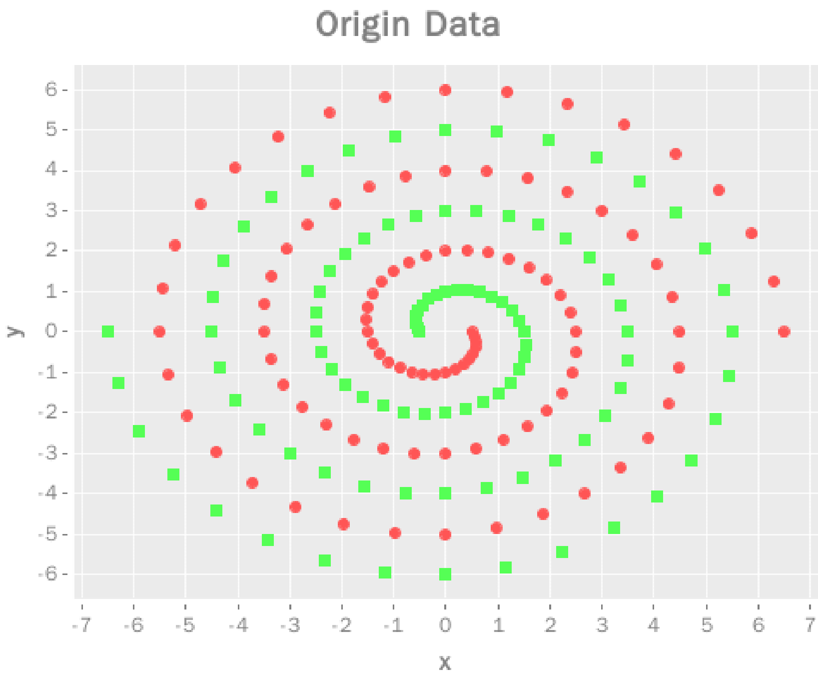
$$s^m = 2(x^m - t)\dot{F^m}$$

$$s^k = s^{k+1} J^k = s^{k+1} (\dot{F^k} V^{k+1} + 2\dot{F^k} X^k U^{k+1})$$

2 Two-spirals problem

2.1 Problem

The two-spirals problem is defined as follow:



It is extremely hard for traditional network to regress such a pattern. In the following I will try to use MLQP to solve this problem.

2.2 Net Structure

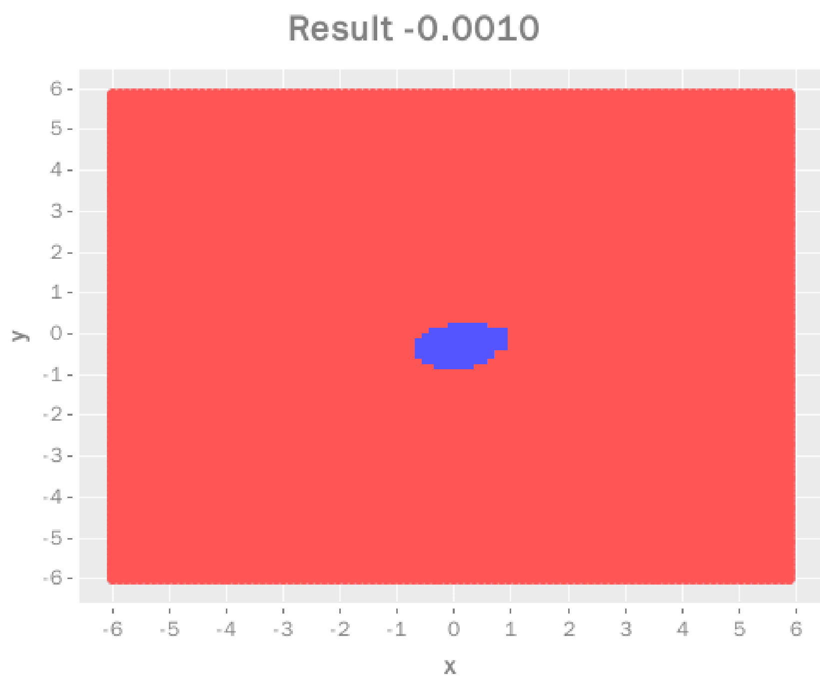
I try to use a two-layer MLQP network to solve the problem.The first layer contains 10 neurons with sigmoid function as transport function.The second layer contains 1 neuron with purelin function as transport function.

To avoid the weight matrixes always being symmetry,I gave each cell of them random initial values between -1 and 1.

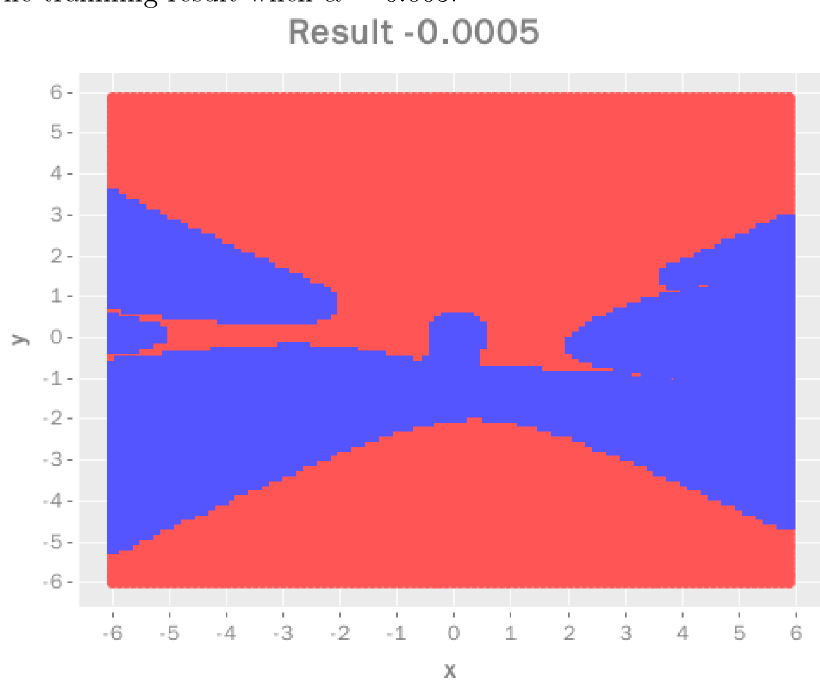
As sigmoid function almost become a constant when x $>$ 5, I use very small α s from 0.0001 to 0.001.

2.3 Result

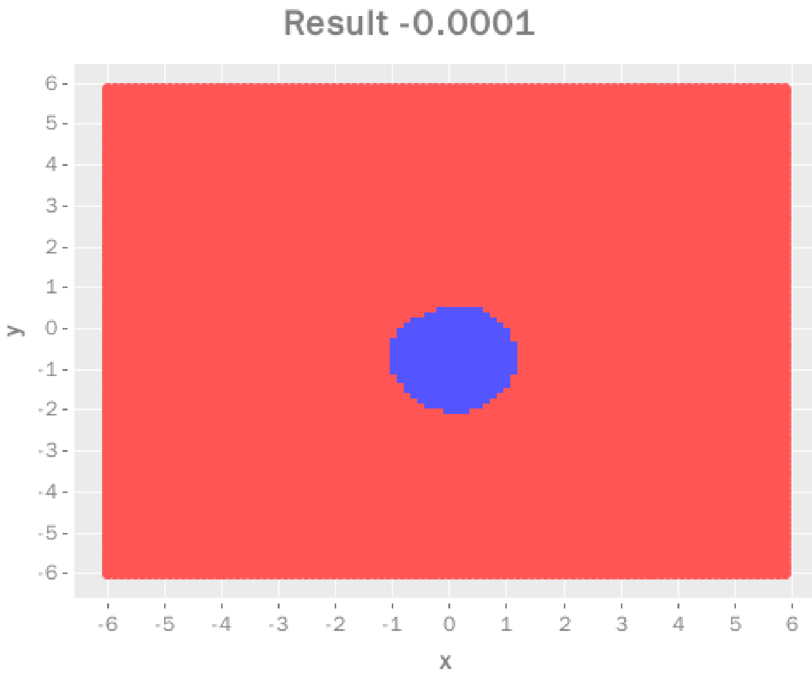
The trainning result when $\alpha = 0.01$:



The trainning result when $\alpha = 0.005$:



The trainning result when $\alpha = 0.001$:



Each result use a random net and looped for 1000 times