COURSE 02457

Non-Linear Signal Processing: Answers to Exercise 5

As the error function for neural network in general are non-linear and non-convex, an iterative optimization routine must be applied in order to train train the weights of the network by minimizing the error function.

When implementing a optimization algorithm, there are several important parameters that needs to be chosen carefully.

The optimization part of the training involves minimizing the error function with respect to the weights of the connections in the network.

Checkpoint 5.1

Initialization range

In order to train a network we first need some data. The script **main5a.m** creates both a training set and a test set using a two layer "teacher network" that has **tanh()** as activation function

$$z_j = tanh\left(a_j\right) = tanh\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i\right) \tag{1}$$

(included bias) for the hidden units and a linear function as the activation function for the output layer

$$y_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j = \sum_{j}^{M}\left(w_{kj}^{(2)} tanh\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i\right)\right) \tag{2}$$

(bias included). The output is corrupted with gaussian noise having zero mean and the same variance as the training data.

We are now going to train another network to learn the mapping from x to t by back propagation with an iterative procedure.

First the weights are initialized to gaussian random numbers $\mathcal{N}(0, range)$. It is the value of *range* that we will study in more detail later in this checkpoint.

while iter < maxiter:

The gradient of the error function with respect to the weights is calculated by the function **gradient()**, which in turn calls the function **forward()** that calculates the output from the units. Then the back propagation error are calculated by

$$\delta_k^{(2)} = y_{kj} - t_{kj} \tag{3}$$

$$\delta_j^{(1)} = (1 - z_j^2) \sum_{j=1}^{M} w_{kj} \delta_k \tag{4}$$

The **gradient()** function returns the derivative of the error function with respect to the weights.

$$\frac{\partial E_n}{\partial w_j^{(1)}} = \delta_j^{(1)} x_i + \alpha^{inp} w_j^{(1)} \tag{5}$$

$$\frac{\partial E_n}{\partial w_k^{(2)}} = \delta_k^{(2)} z_j + \alpha^{(2)} w_k^{(2)} \tag{6}$$

$$\frac{\partial E_n}{\partial w_j^{(1)}} = \sum_{n=1}^{N} \left((1 - z_j^2) \sum_{j=1}^{M} w_{kj}^{(2)} (y_{kj} - t_{kj}) \right) x_i + \alpha^{(1)} w_{ji} \tag{7}$$

$$\frac{\partial E_n}{\partial w_j^{(2)}} = \sum_{j=1}^{M} (y_{kj} - t_{kj}) z_j + \alpha^{(2)} w_k^{(2)} \tag{8}$$

where $\alpha^{(1)}, \alpha^{(2)}$ are the regularization parameters.

Now the new estimate of the weight vector can be calculated

$$w_{it+1}^{(1)} = w_{it}^{(1)} - \eta \mathrm{d}w_{it}^{(1)} \tag{9}$$

$$w_{it+1}^{(2)} = w_{it}^{(2)} - \eta \mathrm{d}w_{it}^{(1)} \tag{10}$$

where η is the learning rate.

The training and test error is then computed by calculating the output by forward propagation using the new weights, and the errors are calculated as

$$E^{train} = \frac{1}{2} \sum_k (t_k^{train} - y_k^{train})^2 \tag{11}$$

$$E^{test} = \frac{1}{2} \sum_k (t_k^{test} - y_k^{test})^2 \tag{12}$$

end while

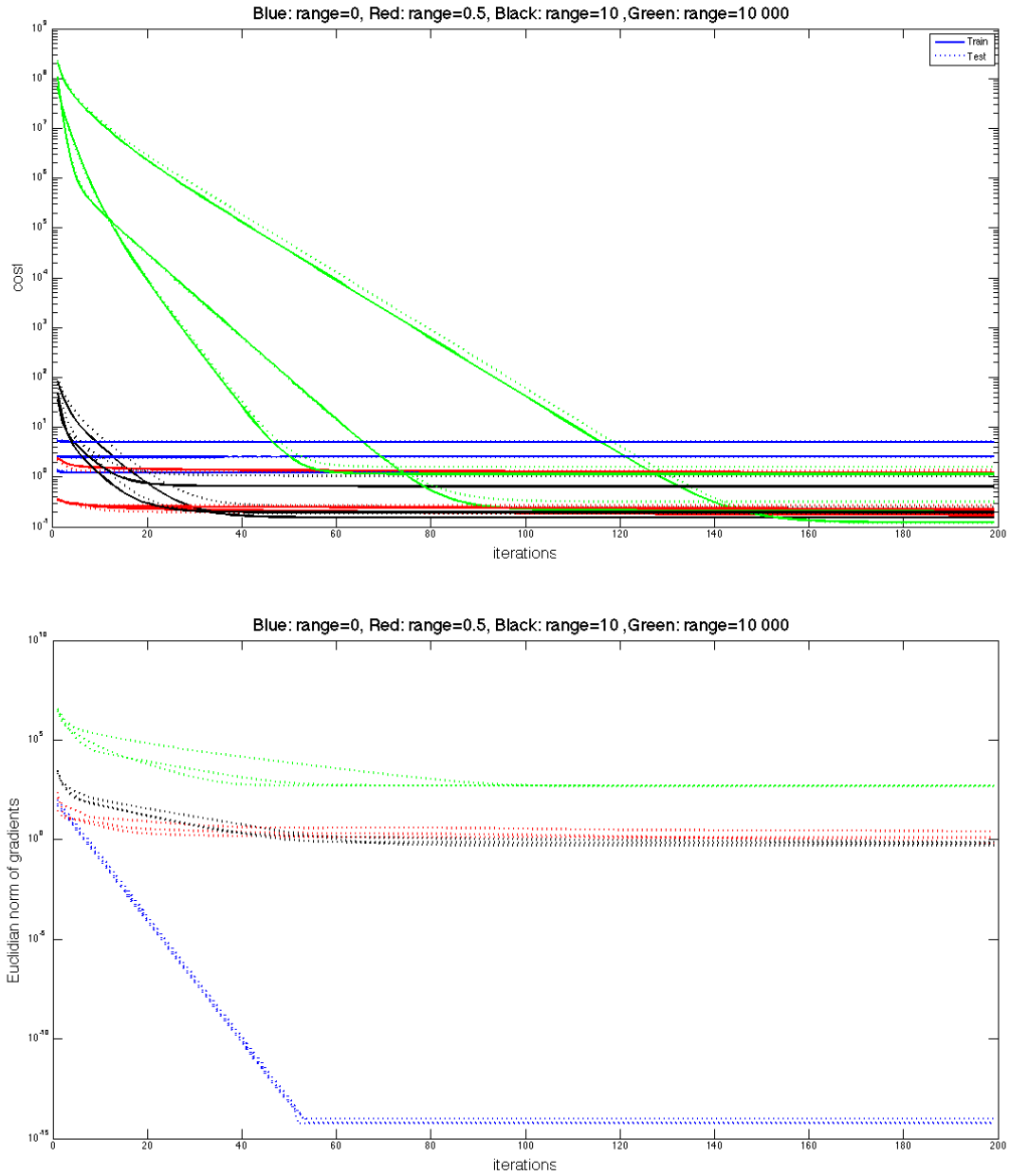

Figure 1: **Top:** The training and test error for a number of different runs with initialization ranges from 0 - 10 000, **Bottom:** The corresponding 2-norms of the gradients during training.

In Fig. 1 the training and test error is plotted as a function of the number of iterations in the training. It is seen that if the initialization range is too high

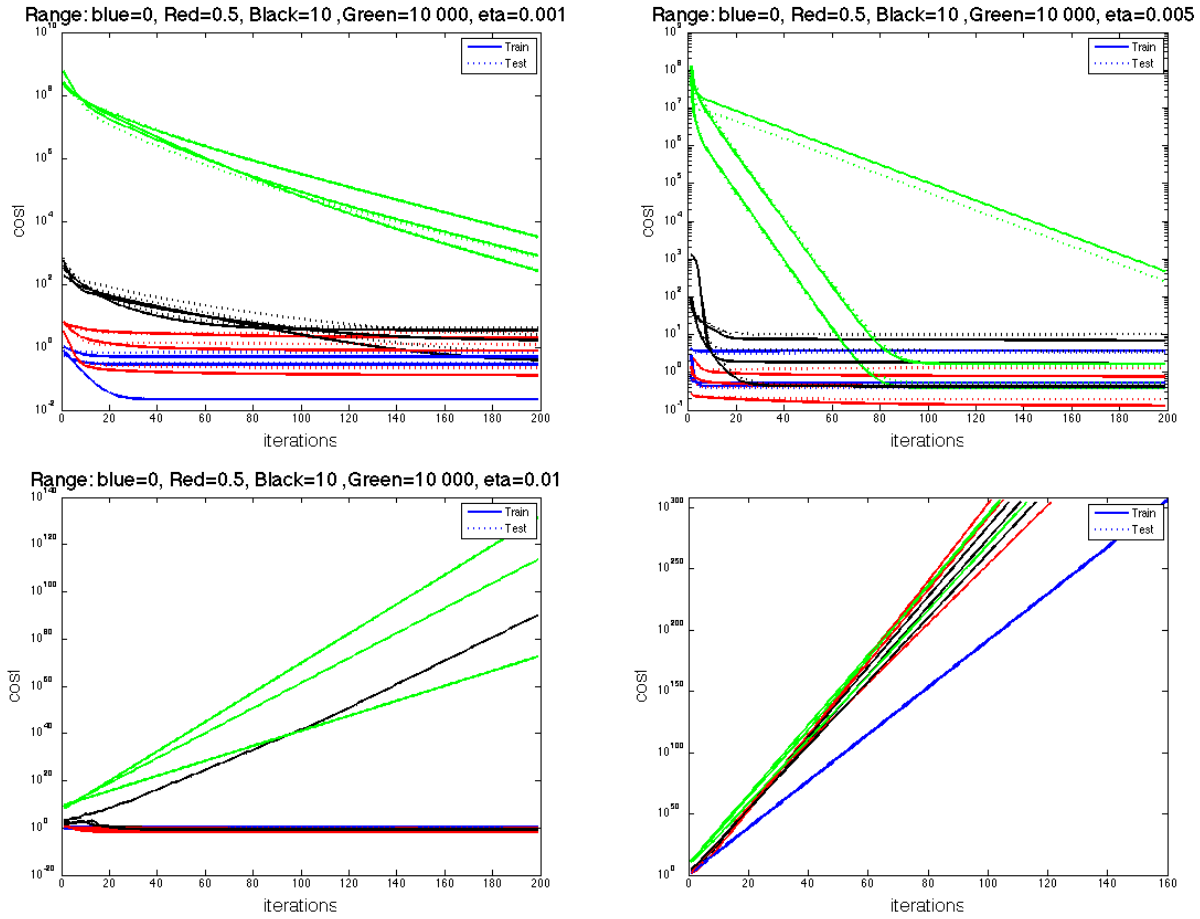

Figure 2: Both alpha's are in this case set to 0.1.

Checkpoint 5.2

Step Size

The step size is important for how fast an optimization method converges, and in Fig. 2 we see the training error plotted as a function of the iterations in the training procedure for four different (initialization) ranges. here it is easily seen that if the step size η is too low, the optimization method converges very slow, but if it is too high, the solution diverges.

Checkpoint 5.3

Stopping criteria

In this check point we investigate what would be a good stopping criteria for the optimization procedure during training of a neural network. This time we apply a (4 - 5 -1) network to the sunspot time series, where the training set is 220 years long and the test set is the rest. In this assignment, the model order of the AR is set to 9, since this has been previously shown to be the optimal order.

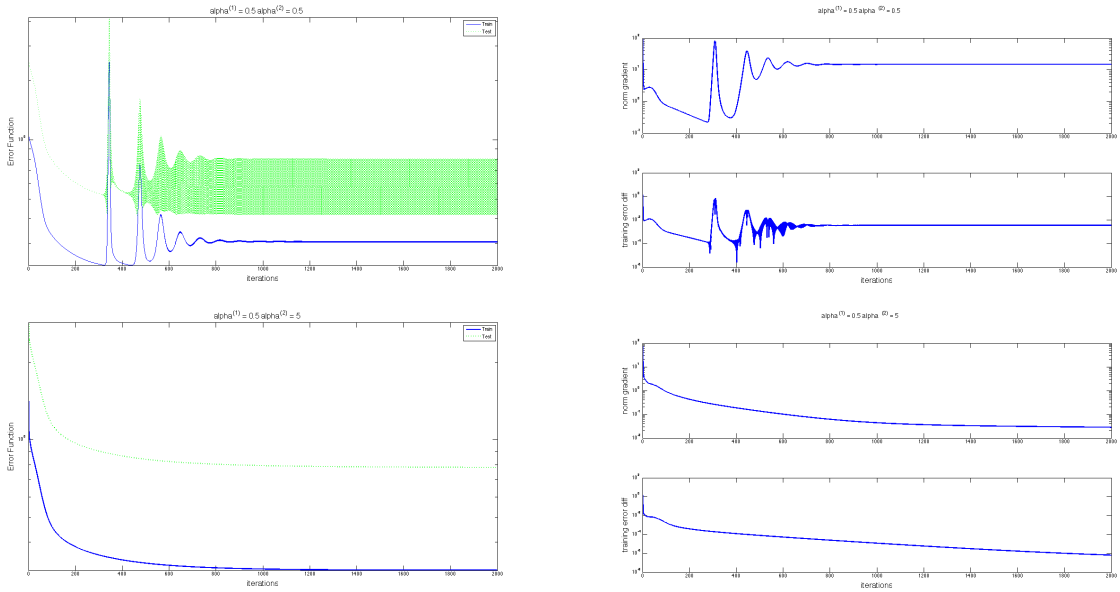

Figure 3: Neural network training using the sunspot series.

In general, it is not a good idea to use the value of the training and test error as a stopping criterion because one typically does not know the values they take at the minimum. Instead, the norm of the gradient or the difference in the error function between each iteration.

Checkpoint 5.3

Convergence Rate

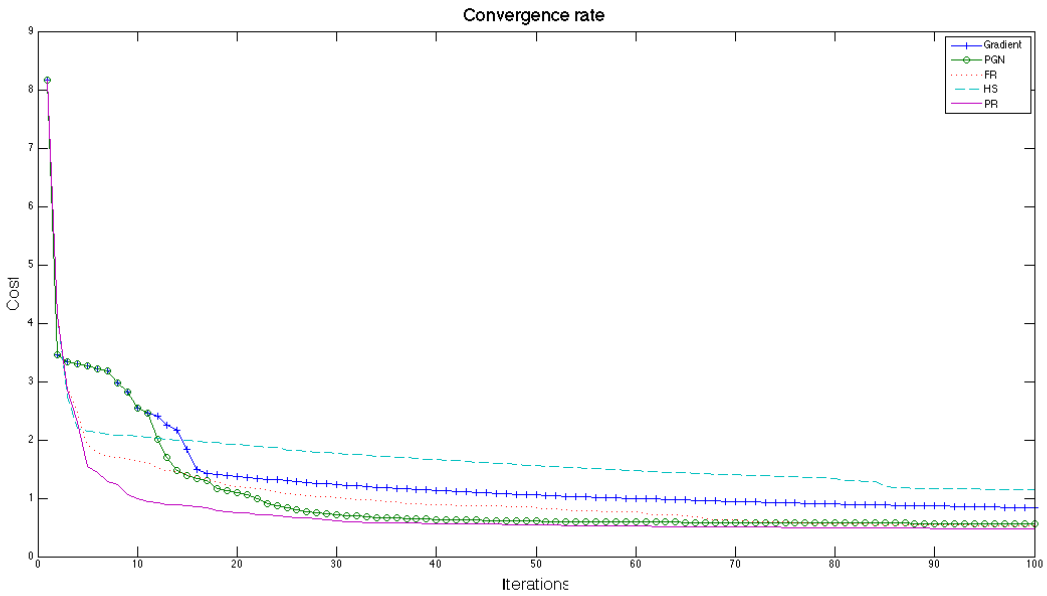

Figure 4: Convergence rate for a range of different optimization algorithms.