T. Kärkkäinen
Course: Application Programming in MATLAB Environment 2000

Exercises 7–8

Mainly some preliminaries for training MLPs using MATLAB.

Problem 1

a) Make an m-function that returns the values of $s_k(a)$ for a given vector **a** and positive integer $k \geq 1$. Draw plots of functions $s_k(a)$ on a suitable grid for different values of k and study their behaviour when k tends to infinity.

b) Modify the previous m-function so that it returns also the values of the derivatives $s_k'(a)$ and plot some of them.

c) Modify the previous m-function so that for a given input-vector **a** it returns two vectors **b** and **c** with components

$$\begin{cases} b_k = \frac{1}{1+\exp(-k a_k)} \\ c_k = \frac{\partial b_k}{\partial a_k} \end{cases}, \quad k = 1, \ldots, \texttt{length(a)}.$$

Problem 2

Using the homepage of this course, go to the UCI-repository and pick up the so-called Iris data set. Store it to a suitable matrix and illustrate it using the m-function `matrix_graph` (your own or Erkki's 'official' version). Program m-functions `scaledata` and `inv_scaledata`, which scale the rows of a given matrix from their minimal and maximal values onto the given interval `[c,d]` and back. Check the correctness of your codes by using `matrix_graph` and the pre- and post-scaling of the Iris-data.

Problem 3

Make an m-function `mlp_out` which for given inputs `x,w1,w2` returns the output `o` of the MLP-network using the last activation routine (c) in Problem 1.

Let us generate a test problem for approximating a noisy sin-function on the interval $[0, 2\pi]$ as follows:

```
delta = 0.3; x = 0:0.15:2*pi; [n0,N] = size(x);
ye = sin(x); y = ye + delta*randn(1,N); n2 = size(y,1);
```

Plot functions (vectors) `ye` and `y`.
Prescale the input-output data into the interval $[0, 1]$.
Train an MLP-approximator for the prescaled data by solving the corresponding optimization problem with a suitable MATLAB routine from the *Optimization Toolbox* by using the sizes $n_1 = 2$–4 for the hidden layer.
Illustrate the obtained solution by adding the MLP-function on different grid for $[0, 2\pi]$ to the initial plots (remember to pre-scale the input and post-scale the output of trained MLP to coincide with the initial scales).
What kind of observations can you make from the different test runs performed during this problem?