file: backpropex.mcd 10/15-20/06
file to give hand calculations for an example of backpropagation

$$p := \begin{bmatrix} 6 \\ -1.3 \end{bmatrix} \qquad t := \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}$$

choose a two layer feedforward NNET with 3 neurons for first layer.

$$W1 := \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0.5 & 0 \end{bmatrix} \qquad b1 := \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$W2 := \begin{bmatrix} 0.2 & 0 & 0.3 \\ 0 & 1 & 0 \end{bmatrix} \qquad b2 := \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

These fix the neuron inputs for the first layer:

$$n1 := W1 \cdot p + b1 \qquad n1 = \begin{bmatrix} 6 \\ -1.3 \\ 3 \end{bmatrix}$$

choose tanh(.) for each activation function

$$a1 := \tanh(n1) \qquad a1 = \begin{bmatrix} 1 \\ -0.862 \\ 0.995 \end{bmatrix}$$

The input to the second layer neurons are now fixed:

$$n2 := W2 \cdot a1 + b2 \qquad n2 = \begin{bmatrix} 0.499 \\ -0.862 \end{bmatrix}$$

$$a2 := \tanh(n2) \qquad a2 = \begin{bmatrix} 0.461 \\ -0.697 \end{bmatrix} \qquad t = \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}$$

output a2 is calculated:

$$e := (t - a2) \qquad e = \begin{bmatrix} 0.139 \\ 1.497 \end{bmatrix}$$

With that the quadratic "energy function" F is available

$$F := e^T \cdot e \qquad F = (2.261)$$

To start the backpropagation, the derivatives of the activation functions are needed

For the second layer these are (they go on the diagonal of the derivative matrix)

$df21 := 1 - \left(a2_1\right)^2 \qquad df22 := 1 - \left(a2_2\right)^2$

$$dF2 := \begin{bmatrix} df21 & 0 \\ 0 & df22 \end{bmatrix} \qquad dF2 = \begin{bmatrix} 0.788 & 0 \\ 0 & 0.514 \end{bmatrix}$$

This allows the sensitivites to be calculated.
for the output layer the sensitivity vector is

$s2 := -2 \cdot dF2 \cdot e$

In order to set the weights the convergence factor α is needed.
This is chosen somewhat arbitrarily between 0 and 1. The closer to 0 the longer it takes to converge but more accuracy is obtained.

$\alpha := 0.2$

Now the updated output layer weights and biases can be calculated

$W2new := W2 - \alpha \cdot s2 \cdot a1^T$

$$s2 = \begin{bmatrix} -0.219 \\ -1.539 \end{bmatrix} \qquad W2 = \begin{bmatrix} 0.2 & 0 & 0.3 \\ 0 & 1 & 0 \end{bmatrix}$$

$$W2new = \begin{bmatrix} 0.244 & -0.038 & 0.344 \\ 0.308 & 0.735 & 0.306 \end{bmatrix}$$

$b2new := b2 - \alpha \cdot s2$

$$b2new = \begin{bmatrix} 0.044 \\ 0.308 \end{bmatrix}$$

To update the first layer weights the derivative matrix is needed

$dF11 := 1 - \left(a1_1\right)^2 \qquad dF12 := 1 - \left(a1_2\right)^2 \qquad dF13 := 1 - \left(a1_3\right)^2$

$$dF1 := \begin{bmatrix} dF11 & 0 & 0 \\ 0 & dF12 & 0 \\ 0o & 0 & dF13 \end{bmatrix} \qquad dF1 = \begin{bmatrix} 2.458 \cdot 10^{-5} & 0 & 0 \\ 0 & 0.257 & 0 \\ 0 & 0 & 9.866 \cdot 10^{-3} \end{bmatrix}$$

Next the sensitivity vector for the first layer is needed; here is where the backward direction is used in that s1 comes from s2

$s1 := dF1 \cdot W2^T \cdot s2$

$$s1 = \begin{bmatrix} -1.077 \cdot 10^{-6} \\ -0.396 \\ -6.482 \cdot 10^{-4} \end{bmatrix}$$

$W1new := W1 - \alpha \cdot s1 \cdot p^T$

$$W1new = \begin{bmatrix} 1 & -2.799 \cdot 10^{-7} \\ 0.475 & 0.897 \\ 0.501 & -1.685 \cdot 10^{-4} \end{bmatrix}$$

$$b1new := b1 - \alpha \cdot s1 \qquad b1new = \begin{bmatrix} 2.153 \cdot 10^{-7} \\ 0.079 \\ 1.296 \cdot 10^{-4} \end{bmatrix}$$

The next step is to repeat by calculating the new signal forward propagation
All steps are as before.

The neuron inputs for the first layer:are

$$n1new := W1new \cdot p + b1new \qquad n1new = \begin{bmatrix} 6 \\ 1.766 \\ 3.005 \end{bmatrix}$$

using tanh(.) for each activation function

$$a1new := \tanh(n1new) \qquad a1new = \begin{bmatrix} 1 \\ 0.943 \\ 0.995 \end{bmatrix}$$

The new input to the second layer neurons are now fixed:

$$n2new := W2new \cdot a1new + b2new \qquad n2new = \begin{bmatrix} 0.594 \\ 1.613 \end{bmatrix}$$

$$a2new := \tanh(n2new) \qquad a2new = \begin{bmatrix} 0.533 \\ 0.924 \end{bmatrix} \qquad t = \begin{bmatrix} 0.6 \\ 0.8 \end{bmatrix}$$

At this point the new error enew between the desired output t and the actual output a2new is calculated:

$$enew := (t - a2new) \qquad enew = \begin{bmatrix} 0.067 \\ -0.124 \end{bmatrix} \qquad e = \begin{bmatrix} 0.139 \\ 1.497 \end{bmatrix}$$

With that the quadratic "energy function" F is available

$$Fnew := enew^{T} \cdot enew$$

$$Fnew = (0.02)$$

Comparing the new error with the old, we see that the new output is closer to the desired output with a much smaller sum of squares error

$$Fnew = (0.02) \qquad F = (2.261)$$