

## **31YB: Biologically Inspired Computing:**

### **Lab Sheet 3 (Thursday, 13<sup>th</sup> November)**

#### **SNNS: RBF Nets and Partially Recurrent Nets**

Remember, if in difficulty consult the WWW manual which is available from:  
<http://www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns/snns.html>

#### **Radial Basis Function Networks: The Two Spirals Problem**

Run SNNS and use the FILE browser to LOAD the spirals problem data set ('spirals.pat') from SNNS/examples.

The spirals problem is a 'toy' classification problem. There are two inputs and these represent co-ordinates in 2D space. (In this data set, these co-ordinates have values between -6.5 and 6.5.) The data points form two 'inter-woven' spiral shapes. The task is to classify the data points as belonging to one or other of the spirals. The target outputs are (0, 1) or (1, 0).

Take a look at the 'spirals.pat' file using the Unix **more** utility. Make sure you understand the format and content of this file. Note the ordering of the patterns.

Use the BIGNET utility in SNNS to create a feed-forward network with just two input units and two output units. Do not create a hidden layer. Do not connect the units. Use DISPLAY to check that the network has been created.

Now open the SNNS CONTROL panel and select the RBF-DDA learning algorithm. Set the LEARN parameters to: 0.4 0.2 0.0 0.0 0.0. Click the ALL button to train the network.

RBF-DDA stands for 'Radial Basis Function – Dynamic Decay Adjustment'. This learning algorithm will create an RBF network with a hidden layer of Gaussian units. It *automatically* decides the size of the hidden layer using a *constructive* algorithm. The number of hidden units chosen depends to some extent on the learning parameters you've just entered. (See the SNNS manual if you want to know more).

**Q1. How many hidden units did the RBF-DDA algorithm create ?**

Use the TEST button to cycle through the patterns.

**Q2. Roughly how many classification errors does the network make ?**

While testing, observe the activation values in the hidden layer.

**Q3. How many hidden units typically fire strongly for a given input pattern ?**

Now let's have a look at the spirals data set.

Open the network ANALYZER from the snns-manager.

Make sure that the **ON** button is selected and that the **LINE** button is not selected.

Click on the Analyzer's **SETUP** button.

Now instruct the analyser to plot an X-Y graph.

Set the x-axis to be the output of unit **1** and the y-axis to be the output of unit **2**.

Set the **min** and **max** for each axis to an appropriate range for the spirals data set.

Now click **DONE** on the Analyzer window.

Now when you cycle through the data by clicking **TEST**, you should see the data plotted by the Network Analyzer.

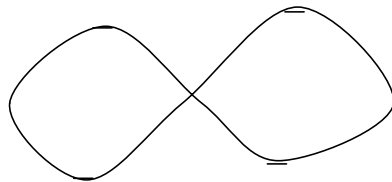
**Q4. Why is the spirals problem a difficult learning problem ?**

**Q5. How many hidden units do you estimate an MLP network would need in order to learn this problem ?**

**Q6. How has the RBF network used its Gaussian units to solve the problem ? (Hint: is the representation localised or distributed ?)**

## Partially Recurrent Networks: Predicting a Trajectory

Imagine an object moving in a figure-of-eight trajectory:



At any given time, the task is to predict the position of the object at the next time-step. The difficulty arises when the object passes through the central crossing point of the '8'. It passes through this point twice so in order to predict the next position, the network must know something about where the object has come from. We can use a partially recurrent network to learn this type of problem. The network will have two inputs (the current x,y position) and two outputs (the predicted x,y position).

Use the FILE browser to load the pattern file 'eight\_016.pat'. This contains a sequence of 16 data points distributed around the figure '8' trajectory. Now load the pattern file 'eight\_160.pat'. This contains the same data sequence repeated 10 times and will be useful for training.

You can load an Elman network which has already been trained on this problem by loading the NET and CFG files 'eight\_elman'. Load this network.

**Q7. How many hidden units and context units are there ?**

**Q8. How many feedback connections are there ?**

Now use the Network ANALYZER tool to analyse its behaviour as follows. In the Analyzer window:

Select ON and LINE

Press SETUP and choose an X-Y graph and the following values for axis, min, max, units, grid:

X	0.0	1.0	1	10
Y	0.0	1.0	2	10

Select DONE in the Network Analyzer Setup panel.

Now click TEST on the control panel to cycle through the data set. You will see the input patterns plotted.

Go back to the SETUP option on the Analyzer and change the selected units from 1, 2 to 11, 12 (output units). Set the multiple test option to  to test 16 patterns in sequence.

Click the M-TEST button to test the trained network. You should see the network's predictions being plotted.

Now create your own Elman and Jordan networks for this problem using the BIGNET utility.

You can choose from four different learning functions. Here they are with some values for the learning parameters that give reasonably good performance:

JE_BP (Backprop)	0.2			
JE_BP_Momentum		0.2	0.5	
JE_Quickprop	0.3	2.0	0.0001	
JE_Rprop	0.1	50.0		

You should set the UPDATE function to JE\_Order.

Set the INITIALISATION function to JE\_Weights.

Reasonable values for the five initialisation parameters are:

1.0	-1.0	0.3	1.0	0.5
-----	------	-----	-----	-----

**Q9. How many hidden/context units does an Elman net need for good prediction ?**

**Q10. How many hidden/context units does a Jordan net need for good prediction ?**