

# Assignment Network Plasticity

Neural Computation 2016-2017. Mark van Rossum

30th November 2016

Version 2

## Practical info

Organize your answers according to the questions; don't merge them. Plots should include axis labels and units (either on the plot, or mentioned in the text), see the link on the course website. Some answers might require units as well.

You will find that some questions are quite open-ended. In order to receive full marks for those you will need to do more than running a simulation and making a plot. Instead, you should substantiate your explanations and claims, for instance by doing additional simulations or mathematical analysis. However, core-dumping (just writing down all you can think of) is discouraged, and incorrect claims can reduce marks. It should not be necessary to consult scientific literature, but if you do use additional literature, cite it. There will be a to-be-determined normalization factor between the number of points scored and the resulting percentage mark.

Copying results is absolutely not allowed and can lead to severe punishment. It's OK to ask for help from your friends. However, this help must not extend to copying code, results, or written text that your friend has written, or that you and your friend have written together. I assess you on the basis of what you are able to do by yourself. It's OK to help a friend. However, this help must not extend to providing your friend with code or written text. If you are found to have done so, a penalty will be assessed against you as well.

Deadline will be announced via email and the website. Hand in a paper copy to ITO (if you are out of town an email with a PDF to me is fine). Late policies are strict and are stated at [www.inf.ed.ac.uk/student-services/teaching-organisation/for-taught-students/coursework-and-projects](http://www.inf.ed.ac.uk/student-services/teaching-organisation/for-taught-students/coursework-and-projects). In case of illness etc, contact your personal tutor (CC me if you want).

## Model and setup

In this assignment we consider a highly abstracted version of a piece of sensory cortex. We consider  $N$  neurons connected via plastic synapses to  $M = 50$  inputs. For convenience, the inputs are arranged on a circle  $\frac{2\pi}{M}, 2\frac{2\pi}{M}, \dots, 2\pi$ .

The input is structured as follows: The inputs are binary (0/1). On a given timestep a random unit on the circle is chosen, and this unit as well as the next  $d - 1$  units in the clockwise direction are turned on. In other words, the stimulus is a randomly placed arc of fixed length. Every timestep a different stimulus is presented. (check for yourself your simulation of this is working correctly).

Initially, the neural activity of neuron  $j$ ,  $y_j$ , is given by  $y_j = [\sum_i w_{ji}x_i]_+$ . The synapses, described by a  $N \times M$  matrix, are every time-step updated according to the rule  $w_{ji} \rightarrow w_{ji} + \epsilon x_i (y_j^2 - y_j/10)$ . After this update the weights are multiplicatively normalized such that for each neuron  $j$ ,  $\sum_i w_{ij}^2 = 1$ .  
**Initialize the weights with uniform random numbers.**

Take the learning rate  $\epsilon = 0.02$ , and stimulus width  $d = 5$ .

In principle we could track the system after every time-step, but to reduce computation, we let the system run in chunks of 100 time-steps. We call such a chunk an 'episode'.

## Questions

**Question 1** (5 points) First we study independent neurons. You can set  $N = 1$ , or run multiple neurons in parallel. Plot the receptive field (i.e. the weight vector) after a few episodes. Describe it's shape. What happens if not all stimuli are equally likely?

We study the stability of the weights.

**Question 2** (10 points) Track the receptive field across episodes. What happens if you vary  $\epsilon$ ? Also compare your findings to the case of classical Hebbian learning in which case we have  $w_{ji} \rightarrow w_{ji} + \epsilon x_i y_j$  (and still normalize as above). Explain the difference and justify your claims.

Because the inputs are placed on a ring, we can describe the weight vector of each neuron by a population vector (see lecture notes). We are interested in its angle, called  $\phi$ .

**Question 3** (10 points) Describe how you calculated the angle. Plot the angle across episodes.

The auto-covariance of the angle is  $\langle (\phi(i) - \bar{\phi})(\phi(i+k) - \bar{\phi}) \rangle$  where  $\phi(i)$  is the angle after episode  $i$ ,  $\bar{\phi}$  is the average angle, and the average is over trials or episodes. Plot the auto-covariance as a function of **episode** ( $k$ ) for both cases of the previous question. Describe in interpret the various aspects of the plot (peak, limits, etc.).

Next, we set the number of neurons to  $N = 5$  and let them interact. The interactions are all-to-all and all the same strength, but self-interactions are excluded.

For the dynamics we use  $y_j = [\sum_i w_{ji}x_i + q \sum_{j' \neq j} \sum_i w_{j'i}x_i]_+$ . Thus when  $q$  is positive, the interaction is excitatory.

**Question 4** (5 points) The above equation for  $y_j$  is an approximation. **Assuming firing rate dynamics as in the lecture notes and** assuming that the neural dynamics are much faster than the duration of the stimulus, what would be a more correct equation for the value of  $y_j$  reached at the end of each stimulus?

**Question 5** (10 points) Study the angles that develop for both positive and negative values of  $q$ . Also plot the covariance function of the angles between pairs of neurons. Interpret the results.

## Matlab/Octave notes

- In Matlab it is useful to introduce a matrix:  $V=q*(ones(n)-eye(n))$ ;
- Some useful functions are: `atan2()`, `norm()`, `normr()`, `xcov()`. Read their documentation.