Neural Networks and Biological Modeling

Professor Wulfram Gerstner
Laboratory of Computational Neuroscience

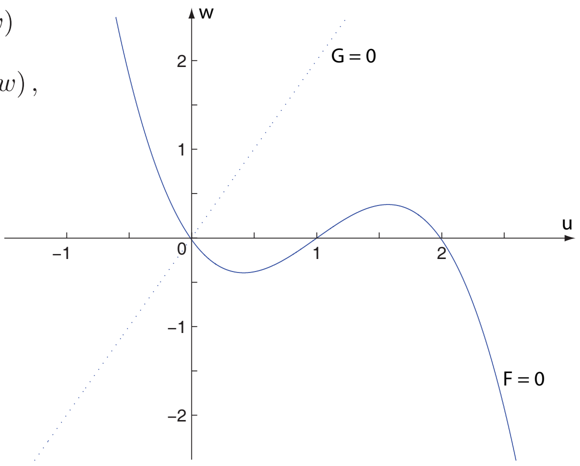
Question set 4

Exercise 1: Inhibitory rebound

Consider the following two-dimensional model:

$$\left[\begin{array}{rcll} \dfrac{du}{dt} & = & -u(u-1)(u-2) - w + I & \equiv F(u,w) \\ \dfrac{dw}{dt} & = & \varepsilon(2u - w) & \equiv \varepsilon G(u,w)\,, \end{array}\right. \tag{1}$$

where $\varepsilon = 0.01$.



Suppose that an inhibitory current step is applied,

$$I(t) = \left\{ \begin{array}{cc} -I_0 & t \leq 0 \\ 0 & t > 0 \end{array} \right. \tag{2}$$

1.1 How does the fixed point move?

1.2 What happens after the driving current is removed? Sketch the form of the trajectories for increasing values of I_0. What happens for large I_0?

Exercise 2: Impulse response (make as homework)

Consider the following system with separation of time scales:

$$\left[\begin{array}{rl} \dfrac{du}{dt} & = \frac{1}{\epsilon}\left(f(u) - w + I\right) \\ \dfrac{dw}{dt} & = bu - \gamma w \end{array}\right.$$

where $\epsilon \ll 1$ and

$$f(u) = \left\{ \begin{array}{ll} -u & \text{si } u < 1 \\ \dfrac{u-1}{a} - 1 & \text{si } 1 \leq u < 1 + 2a \\ 2(1+a) - u & \text{si } u > 1 + 2a \end{array} \right.$$

Discuss the behaviour of the trajectories of $u(t)$ in response to a current pulse $I(t) = q\delta(t)$. Sketch these trajectories in the phase plane and in the temporal domain for a few values of q. Does the model exhibit a threshold-like behaviour?

Exercise 3

Show that the learning rule

$$\frac{d}{dt}w_{ij} = a_2^{\text{corr}}(\nu_i^{\text{post}} - \vartheta)\nu_j^{\text{pre}} \tag{3}$$

is a special case of

$$\frac{d}{dt}w_{ij} = a_0 + a_1^{\text{pre}}\nu_j^{\text{pre}} + a_1^{\text{post}}\nu_i^{\text{post}} + a_2^{\text{corr}}\nu_j^{\text{pre}}\nu_i^{\text{post}}\,. \tag{4}$$

Exercise 4: BCM rule

The Bienenstock-Cooper-Munro rule takes the form

$$\frac{d}{dt}w_{ij} = a_2^{\text{corr}}\Phi(\nu_i^{\text{post}} - \vartheta)\nu_j^{\text{pre}}\,. \tag{5}$$

Consider a linear neuron,

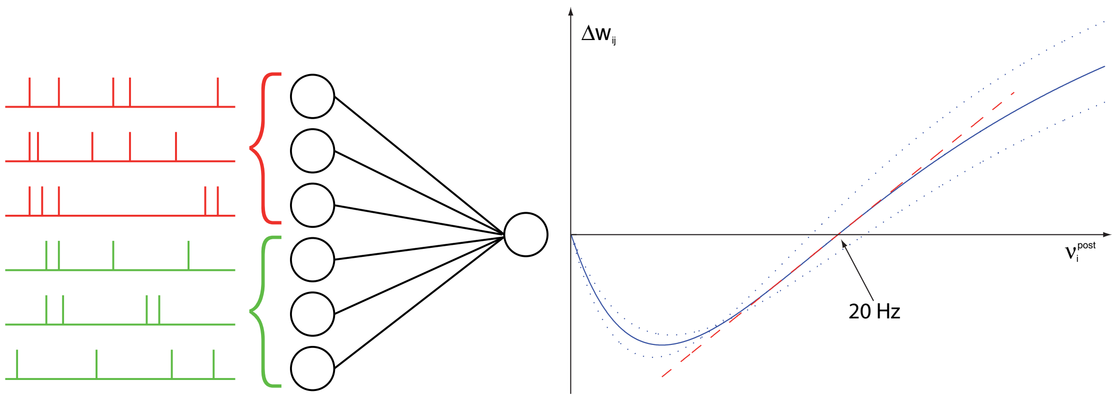
$$\nu_i^{\text{post}} = \sum_j w_{ij}\nu_j^{\text{pre}}\ , \tag{6}$$

and suppose that the presynaptic population consists of two groups that fire *one after the other* repetitively. Each group contains 10 neurons, and initially all synapses have unit weight. In the first two questions, assume $\vartheta = 20$ Hz.

4.1 Suppose that the first group fires at 3 Hz and the second at 1 Hz. How do the weights evolve?

4.2 Suppose that the first group fires at 3 Hz and the second at 2.5 Hz. How do the weights evolve?

4.3 In the same situation as 4.2, how would you make the threshold ϑ slide so as to allow group discrimination?



(a) (b)

Figure 1

Exercise 5: Numerical simulation of Oja's hebbian learning rule

By now you should have python running correctly. If not, please ask the assistants for help. You should also be able to edit python scripts in your favorite code editor, and to execute them from ipython. Download `set4.py` from moodle. Create a script file `set4_answers.py` in the same directory, and give it the following header:

```
from numpy import *
from scipy import *
from set4 import *
```

You will type the code for your answers right below. When you want to execute your code, open ipython (pylab) in a terminal and type:

```
>> import set4_answers (or reload set4_answers if already imported)
```

The `set4` module contains two documented functions:

- `make_cloud(...)` which generates a 2D elliptic gaussian cloud of datapoints
- `learn(...)` which runs Oja's learning rule on the data and returns the time course of the weight vector

In your answers you will need to plot data. Plotting functions are already loaded with pylab, and you will find example-based tutorials at

```
http://matplotlib.sourceforge.net/gallery.html
```

5.1 Run the learning rule on a *circular* data cloud (`ratio=1`). Plot the time course of both components of the weight vector. Do it many times. What happens?

5.2 Repeat the previous question with an *elongated* elliptic data cloud (e.g. `ratio=0.3`). Again, do it several times. What is the difference with a circular data cloud, in terms of learning curve?

5.3 Try to change the orientation of the ellipsoid (try several different angles). What does Oja's rule do? Hint: plot the learned weight vector in 2D space, and relate its orientation to that of the ellipsoid.

5.4 The above work assumes that the input activities can be negative (indeed the input were statistically centered). In real neurons, if we think of the "activity" as the firing rate, this cannot be less than zero. Try again the previous question, applying the learning rule on `5 + make_cloud(...)`, which centers the data around `(5,5)`. Draw your conclusions. Can you think of a modification of the rule?

Exercise 6: Numerical simulation of the BCM learning rule

6.1 Create a function `bcm_learn(...)` which implements the quadratic BCM learning rule (get inspired from the code for Oja's rule):

$$\Delta w = \eta y \left(y - \frac{\langle y^2 \rangle}{y_0} \right) x$$

where $y = w \cdot x$. In order to test the rule, we provide in module `set4` the documented function `make_image(...)` which generates small images (m-by-m where m is an argument of the function) made of gaussian bumps centered randomly. You can therefore run the learning rule on thousands of such images, and plot the resulting weights (as an m-by-m image). Make sure the synaptic weights stay positive. Choose y_0 to be the average output activity resulting from the initial weights (choose random initial weights, and check the average output activity without learning).