Neuroinformatics — lab exercises manual

authors:
Eduard Bakstein (TA), Daniel Novák (Lecturer)
http://nit.felk.cvut.cz, Prague 2015

March 10, 2016

Abstract

This is supportive text for labs of Neuroinformatics course at Czech Technical University in Prague, Faculty of Electrical Engineering. Exercises for each task are provided in Matlab language.
We further recommend a book [3], which is nice and easy to read. A good summary is provided by [1]. Particular topics are covered by [2], [4] Some advance material cis cover by [2], [4].

Contents

0 Mathematical apparatus

We will introduce numerical apparatus, which will be applied during activity modelling of neurons.

0.1 Numerical solution of differential equations

As in other sectors dealing with dynamic systems, we will use differential equations throughout this course. Due to the fact that their exact analytical solutions can often be difficult or even impossible, we introduce in this exercise the approximate numerical solutions.

Euler's method

Let's consider first order differential equation

$$\frac{\delta x}{\delta t} = f(x, t), \tag{0.1}$$

Euler's first order metoda consist of discretization $\frac{\delta x}{\delta t} = \frac{\Delta x}{\Delta t}$. Lets' take:

$$\begin{aligned} \Delta x &= x(t + \Delta t) - x(t) = x(t_2) - x(t) \\ \Delta t &= t_2 - t, \end{aligned}$$

then we can express eq (0.1) as

$$\frac{\Delta x}{\Delta t} = f(x(t), t)$$

finally

$$x(t + \Delta t) = x(t) + \Delta t f(x(t), t)$$

We can approximate solution by taking into account the slope of the line at that point. However, if the slope is dependent on t, this will be a very rough approximation. In this case, we can refine the solutions by other parameters of the Taylor series according to the formula:

$$x(t + \Delta t) = x(t) + \Delta t \frac{\delta x}{\delta t} + \frac{1}{2}(\Delta t)^2 \frac{\delta^2 x}{\delta t^2} + O, \tag{0.2}$$

where O represents all members of the higher orders. In the second order therefore we model in addition the slope curvature. This approximation should be closer to analytical solutions.

Runge-Kutta method

In the methods of higher orders we can additionally refine the solution by estimation not in the point x and $x + \Delta x$, but in the middle of this period - the so-called *„midpoint method"*. The resulting value should be more representative in a given interval and lead to a more accurate estimate. Moreover, if we do not have analytical expression of parameters of the Taylor expansion of higher orders, we can again estimate parameters of high orders using numerical methods.

The actual method of Runge-Kutta numerical integration is the 4th order, which combines estimation in the middle of the interval with a numerical estimate of higher order. It can therefore approximate solution of arbitrary functions. This leads to a higher computational cost, but - as we shall see - leads to very accurate estimates of the solution. Runge-Kutta method is implemented in Matlab function `ode45()`.

Exercise 0.1 Our task is numerical approximation of the following differential equation

$$\frac{dx}{dt} = t - x + 1, \tag{0.3}$$

initial conditions: x (0) = 1 using Euler's method (1 and 2 order) and Runge-Kutta methods.

The analytical solution of this equation has the form $x = t + e^{-t}$ [1]. Solve the equation in the interval $\langle 0, 5 \rangle$, choose $\Delta t = 0.02$.

---

[1] You can check e.g. at http://www.wolframalpha.com/, question: „solve differential equation x '= ... "

Task 0.1 (2 b) Plot the solution $x(t) = f(t)$ for $t = 0, \ldots, 2s$ using Euler's method of first and second order [2] (X_{Euler}) and plot into graph along with the analytical solution.

Task 0.2 (1 b) Solve the task using Runge-Kutta x_{Runge} and add the solution to the same graph.
    Hints: function `ode45` has as first parameter a callback to the function, which desribes the equation to be solved. In this case the callback can be easily defined using so called *function handle* and *anonymous function*[3] as `ode_func = @(t,x,flag) 1-x+t;`

Task 0.3 (1.5 b) Plot the dependence of relative error of each numerical method against the analytical solution. For example (for Euler's method) $(x_{Euler} - x_{exact})/x_{Euler} = f(\Delta t)$

Task 0.4 (1.5 b) Plot absolute error of each numerical method as a function of the size of the integration step $\Delta t \in (0.001, 1)s$ in time $t = 1s$.

---

[2]when solving the second order Euler method you can proceed in two ways. 1) Derivating analytic function $f(x,t)$.
2) estimate value of $f'(x,t)$ using the slope $f(x,t)$ between points t and $t + \mathit{Delta}t$
[3]see for example. `http://www.mathworks.com/help/matlab/matlab_prog/creating-a-function-handle.html`

1 Neuron models

In this part we will describe properties of the action potential and behaviour of individual neuron. We will implement several neuron models of differing level of complexity and investigate how their simplifying assumptions affect the resulting action potentials or spike-trains.

1.1 Model of membrane and synapse: simpler case

Taking into accout that nerve impulses propagate through changes in electrical potential on the cell membrane, it seems natural to create the basic model as an equivalent electrical circuit. The simplest model, that we introduce in this exercise, is working only with one type of channel - leakega chloride channels (always opened) - and takes the form of an RC circuit, the resting battery, replacing the resting potential of the membrane (Nerst potential) . As we shall see, the response to the input excitation is far away from the the real action potential but serve us in understanding the basic concept.

Exercise 1.1 (RC model) Model memrabe behavior using RC model - see figure 1. Input membrane current I_{stim} is rectungalar signal $10pA$ lasting $20ms$. It is neccesary to convert the stimulating currentI_{stim} to current density to be compatible with I_{Cl} and I_C. This can be done easily by dividing the input current by membrane area A[4]. The electrode is stimulating and at the same time recording the membrane current (expressed in cm^2). Hence $I'_{stim} = I_{stim}/A \approx 10^{-11} \cdot 10^6 \approx 10^{-5}$. The mebrane parameters are the following:

capacity: $C_m = 1\ \mu F/cm^2$,
conductance: $g_{Cl} = 0.3\ ms/cm^2$,
time constant : $\tau = C_m/g_{Cl}$,
mebrane surface : $A \approx 1 \cdot 10^{-6}\ cm^2$
Nerns potential of Cl: $V_{Cl} = -68\ mV$,
initial conditions: $V(0) = -68\ mV$, $I_{Cl}(0) = 0\ \mu A/cm^2$, $I_C(0) = 0\ \mu A/cm^2$.

Task 1.1 (2 b) Plot dependence $V(t)(= \phi_{in} - \phi_{out})$, $I_{Cl}(t)$, $I_C(t)$for these time interval $\langle 0, 40 \rangle ms$, $\Delta t = 0.01ms$
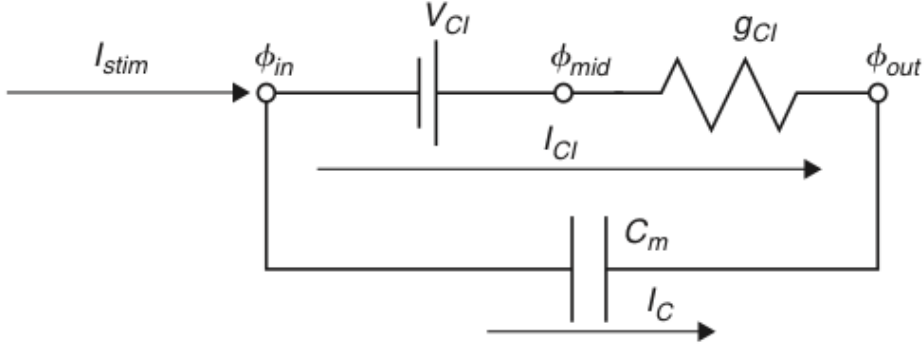

Figure 1: Membrane model with Cl leakage channel

Hence

$$I_C(t) = C_m \frac{dV}{dt}(t) \tag{1.1}$$

$$I_C(t) = \frac{I_{stim}(t)}{A} - I_{Cl}(t) \qquad \rightarrow I_{stim} = A \cdot I_C(t) + A \cdot I_{Cl}(t) \tag{1.2}$$

$$I_{Cl}(t) = g_{Cl}(V(t) - V_{Cl}) \tag{1.3}$$

We can express the time constant of the RC circuit as

---

[4]I_{Cl} and I_C are in fact current densities and it would be more appropriate to use ϱ instead of I. However, we will stick to the terminology used in [3] and use I.

$$\tau \quad = \quad \frac{C_m}{g_{Cl}} \tag{1.4}$$

Combining the above equations we get

$$C_m \frac{dV}{dt} \quad = \quad \frac{I_{stim}(t)}{A} - g_{Cl}(V(t) - V_{Cl}) \tag{1.5}$$

Or expressed using time constant τ from 1.4 we get

$$\tau \frac{dV}{dt} \quad = \quad V_{Cl} - V(t) + \frac{I_{stim}(t)}{Ag_{Cl}} \tag{1.6}$$

Euler's method (forward)

$$\tau \frac{V(j) - V(j-1)}{\Delta t} \quad = \quad V_{Cl} - V(j-1) + \frac{I_{stim}(j-1)}{Ag_{Cl}} \tag{1.7}$$

$$V(j) \quad = \quad V(j-1) + \frac{\Delta t}{\tau}[V_{Cl} - V(j-1) + \frac{I_{stim}(j-1)}{Ag_{Cl}}] \tag{1.8}$$

Exercise 1.2 (EPSP model) The task is analysis of a model depicted in figure 2. This model incorporates so called „excitatory postsynaptic potential" (EPSP) simulating behavior of membrane dendrite of the postsynaptic's neuron after receiving excitation. Synapse is modelled by chanchagable conductance g_{syn}. In other words: the model contains additionally channels, which react to release to the neurotransmitter (literally they are opening, hence their conductance is decreasing - see result of eq (1.9)) with time constant $\tau_{syn} = 1 \ mS$. The parameters are the same like in the previous case. Stimulation current $I_{stim} = 0$, $V_{syn} = 10 \ mV$. In time $t = 1 \ ms$ the neurotransmitter is released, hence $g_{syn}(1 + \delta) = 1$. Initial conditions $V(1) = 0$, $I_{syn} = 0$, $g_{syn}(1) = 0$, $g_L = 1$.

$$\tau_{syn} \frac{dg_{syn}(t)}{dt} = -g_{syn}(t) + \delta(t - t_{pre} - t_{delay}) \tag{1.9}$$

Task 1.2 (0 b) Plot dependence $V(t)$, $I_{Cl}(t)$, $I_C(t)$, $I_{syn}(t)$. Explain trends and compare them with the previous task 1.1.
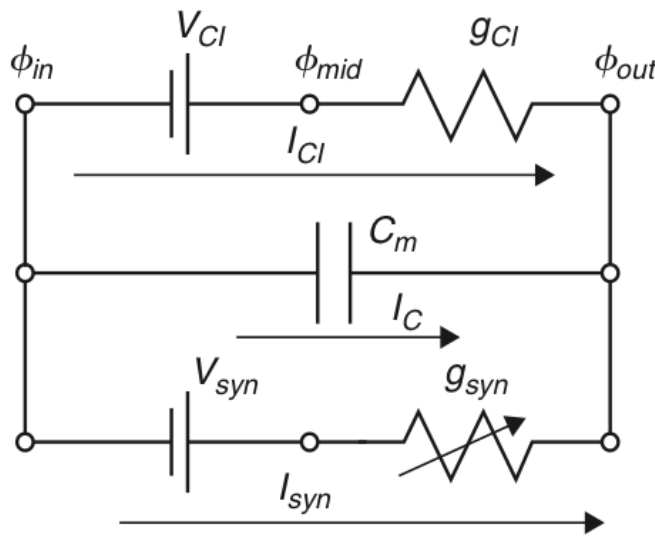

Figure 2: EPSP: Synapse model with leakage Cl channel

1.2   Mebrane and synapse modelling 2: Hodgkin-Huxley

Exercise 1.3 (Hodgkin-Huxley) The taks is analysis of Hodgkin-Huxley(HH) model depicted in figure 3. Compared to the previous models, HH incorporates variables which are dependent both on time model and voltage, hence Na^+ a K^+ channels are controlled by voltage. The model is based on the following set of eqautions
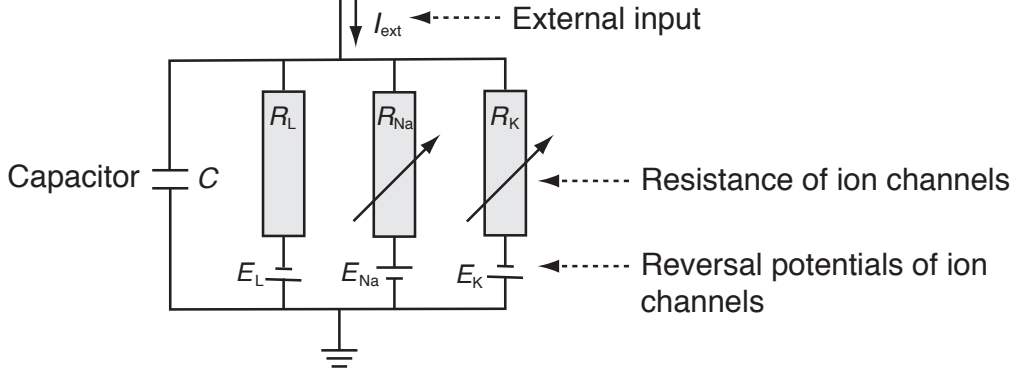

Figure 3: Hodgkin-Huxley model. Each branch represents resting potential a and resistivity of a particular channel: E_L, R_L - channels always opened („Leakage channels"), E_{Na}, R_{Na} - sodium channels controlled by voltage , E_K, R_K - calcium channels controlled by voltage.

Current in ion channels can be described by Ohm's law:

$$I_{ion} = \hat{g}_{ion}(V - E_{ion}) \qquad \left(= (V - E_{ion})/\hat{R}_{ion}\right), \tag{1.10}$$

where $\hat{g}_{ion}$ is the maximum conductance of ion channel

Next, we introduce auxilary variables dependent on voltage and time $n(V,t)$, $m(V,t)$, $h(V,t)$, resulting in the following conductances:

$$\hat{g_K}(V,t) = g_K n^4 \tag{1.11}$$
$$\hat{g_{Na}}(V,t) = g_{Na} m^3 h \tag{1.12}$$

Combining eq.1.12-1.10 we get

$$C\frac{\mathrm{d}V}{\mathrm{d}t} = -g_K n^4 (V - E_K) - g_{Na} m^3 h (V - E_{Na}) - g_L (V - E_L) + I_{ext}(t) \tag{1.13}$$

Next, we define time constants

$$\tau_n(V)\frac{\mathrm{d}n}{\mathrm{d}t} = -[n - n_0(V)] \tag{1.14}$$
$$\tau_m(V)\frac{\mathrm{d}m}{\mathrm{d}t} = -[m - m_0(V)] \tag{1.15}$$
$$\tau_h(V)\frac{\mathrm{d}h}{\mathrm{d}t} = -[h - h_0(V)] \tag{1.16}$$

For each variable we get

$$\frac{dx}{dt} = -\frac{1}{\tau_x(V)}[x - x_0(V)], \qquad x \in \{n, m, h\} \tag{1.17}$$

and after substituing Euler's numerical integration we get

$$x(t + \Delta t) = \left(1 - \frac{\Delta t}{\tau_x}\right)x(t) + \frac{\Delta t}{\tau_x}x_0. \tag{1.18}$$

Initial conditions hold:

$$x(0) = \frac{\alpha}{\alpha + \beta}, \qquad \tau_x = \alpha\beta, \qquad x \in \{n, m, h\} \tag{1.19}$$

$$\alpha_n = \frac{10 - V}{100\left(e^{\frac{10-V}{10}} - 1\right)}, \qquad \beta_n = 0.125e^{-\frac{V}{80}} \tag{1.20}$$

$$\alpha_m = \frac{25 - V}{10\left(e^{\frac{25-V}{10}} - 1\right)}, \qquad \beta_m = 4e^{-\frac{V}{18}} \tag{1.21}$$

$$\alpha_h = 0.07e^{\frac{V}{20}}, \qquad \beta_h = \frac{1}{e^{\frac{30-V}{10}} + 1} \tag{1.22}$$

Task 1.3 (0 b) Download model implementation file `cv3_HH.m` from course website. Visualize time output of the model (membrane voltage V), which is excited by the constant current $I_{ext} = 30\mu A/cm^2$ (default settings). Implementation works from time -30s to get steady system. Plot output for time $0 - 100ms$.

Task 1.4 (1 b) Expand visualization by adding time trend of conductance g for each channel type (describe different time series using `legend`)

Task 1.5 (2 b) Visualize dependence of firing frequency on external constant current I_{ext} - so called *activation or transfer function*. Set the input current I_{ext} in the range $0 - 15 \ \mu A/cm^2$. You should compute firing frequency after reaching the steady state (hint: use fce `diff`, `find` alt. `findpeaks`). Which shape does the activation function resamble? For which values of the current I_{ext} will the shape of the activation function change significantly (perform a discussion)?

Task 1.6 (2 b) Adding noise to the input current will change behaviour of the model. Visualize membrane voltage V and activation function as a response to input current with added noise. Use $I_{ext} = 30\mu A/cm^2$ and white noise (`randn`) with the standard deviation 60 and zero mean, for plotting activation function use $I_{ext} = \langle 0, 15 \rangle \mu A/cm^2$, noise with standard deviation 30. How did the resulting output change? Try out for different noise levels.

Task 1.7 (1 b) Plot ISI histogram for the input current $I_{ext} = 30\mu A/cm^2$ and noise with standard deviation 60. Compare to the case with zero noise and discuss the results.

1.3   Spike train modelling

We showed in the previous tasks, how does noise on the input affect temporal and frequency characteristics of a neuron. Noise signal on neuron input is much more realistic than constant current - mainly due to the fact that neuron input is formed by a summary signal from many synapses of pre-synaptic neurons, connected to its dendrites. In this section, we will simulate such signal and use it as input to a neuron model.

You have learned in the lectures, that interspike intervals (ISI) of typical cortical neurons exhibit log-normal distribution. Before we start modelling spiketrains, let us verify this on a model.

Exercise 1.4 (LIF neuron model and Poisson spiketrain) The complex models from previous exercises, such as Hodgkin-Huxley, are able to model the action potential shape based on physiological modelling of ion channels' behavior. If we want to model more complex neural networks, even simpler models (such as Wilson model) are overly complex for such application. When modelling neural networks, the accurate shape of action potential is not as important (we will see it can be simply modelled using α functions) as the spike timing. For these purposes, the *Leaky integrate and fire* neuron is often used. The basics of this model includes simple integrator, which is reset to resting potential when a preset threshold is exceeded.

Task 1.8 (1 b) Download LIF model from the course website - `cv4_lif.m`. Show model output for constant input current (default setting). Plot output voltage, along with spike times (= reset times) over time. Compute firing rate and display an ISI histogram.

Task 1.9 (2 b) Modify previous task by applying input noise with mean 12 and standard deviation 100, instead of constant current (function `randn`). Simulate for times in the range $0 - 10^5$ and integration step $dt = 0.01$. Set bin number to 60 when computing histograms. Convert histograms to probability density function estimates ($\sum pdf = 1$).

Task 1.10 (3 b) Fit lognormal distribution to the data from the previous task. The lognormal distribution is given by:

$$pdf^{lognormal}(x; \mu, \sigma) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-\frac{(log(x) - \mu)^2}{2\sigma^2}} \tag{1.23}$$

One possible solution would be to fit the analytical pdf to ISI histogram from the previous task. This, however, would be completely wrong, as one point in the histogram does not mean one, but many data points (think about this for a while!).

Thus, we will use the function `lognfit`, which estimates lognormal distribution parameters using maximum likelihood estimates. Based on these estimates, calculate the lognormal pdf and display along with the histogram. (You will probably have to adjust the bin number to get a good representation)

2 Real data analysis, modelling of neuronal populations

In this block, we will play with analyzing of real μEEG (*micro-EEG*) data, recorded from brains of Parkinson's disease patients. We will compare the data to our simulations from the previous block and we will analyze correlation with other biosignal - the electrooculogram (EOG).

Further in this section we will simulate more complicated network of artificial neirons and (so called *spiking network*) and analyze its behavior under varying conditions.

3 Information coding and transfer in the brain

In this block, we will model some of the processes taking part in transferring and storing information in the brain. We will go through the basics of cognitive modelling and we will try out the so-called *Self-organizing maps*.

Bibliography

[1] David Fitzpatrick William C. Hall Anthony-Samuel LaMantia Leonard E. White Dale Purves, George J. Augustine. *Neuroscience.* Sinauer Associates, Inc., 5th. edition edition, 2011.

[2] Michael L. Hines Nicholas T. Carnevale. *The Neuron Book.* Cambridge University Press, 2006.

[3] Thomas Trappenberg. *Fundamentals of Computational Neuroscience.* Oxford University Press, USA, June 2010.

[4] Werner M. Kistler Wulfram Gerstner. *Spiking Neuron Models: Single Neurons, Populations, Plasticity.* Cambridge University Press, 2002.