Neuroinformatics — lab exercises manual

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

February 22, 2017

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

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.

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 $\langle 0, 5)$, 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 $t + \ Deltat$

[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.

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*.