HUMBOLDT-UNIVERSITÄT ZU BERLIN
BERNSTEIN CENTER FOR COMPUTATIONAL NEUROSCIENCE

Humboldt-Universität zu Berlin
Philippstr. 13 House 4
Prof. Richard Kempter
Prof. Benjamin Lindner

Phone: 030/2093-9110
Fax: 030/2093-6771
Webpage: http://www.bccn-berlin.de/

Models of Neural Systems, WS 2020/21
Computer Practical 4
Assignment due: December, 14th, 2020, 10 am

Ordinary Differential Equations (ODEs)

Differential equations describe the evolution of systems in continuous time and are widely used in science and engineering. Here, we will focus on equations of the form:

$$\dot{x}(t) = f\left(x(t), t\right), \tag{1}$$

where $\dot{x}(t) \equiv \frac{\mathrm{d}x}{\mathrm{d}t}$ is the first derivative of $x(t)$ with respect to time t. The solution is a function (dynamical variable x) that satisfies equation (1). The solution is completely determined if we prescribe a certain value of the function $x(t)$ at a certain time instant, for example in the form of an initial condition $x(0) = x_0$. Equation (1) involves only the first derivative with respect to a single independent variable t and therefore this equation is called a first-order ordinary differential equation. Here, you will recapitulate how to find analytical solutions to such equations and learn solving them numerically. Finally, you will apply the methods to model the potential across a lipid membrane.

Exercises

1. **Analytical solutions to ODEs**

   Remind yourself how to solve first order ODEs and solve explicitly (without using the computer) the following differential equations with the initial condition $x(0) = x_0$:

   (a) $\dot{x} = -x; \quad x_0 = 1$

   (b) $\dot{x} = x^{-1}; \quad x_0 = 1$

   (c) $\dot{x} = 1 - x; \quad x_0 = 0$

   (d) (optional) $\dot{x} = x(1-x); \quad x_0 = \frac{1}{2}$

2. **Numerical solutions to ODEs** The simplest numerical method of solving equation (1) is by discretization. Let's assume that the function $x(t)$ is constant over a short time interval Δt. We can then rewrite the equation using finite steps:

   $$x_{i+1} = x_i + f(x_i, t_i)\Delta t \tag{2}$$

   To find the solution, we start with the value x_0 (the initial condition) and then proceed iteratively by adding small increments to the function according to equation (2). This algorithm is called Euler method.

   (a) Define a Python function that takes as an argument the function $f(x, t)$, the initial condition, the stop time, and the integration step Δt:

   ```
   def euler(f_func, x_0, t_max, dt):
       ...
   #Example: solve the logistic ODE
   x_t=euler(logistic, 0.5, 5., 0.01)
   ```

   where `logistic` is a function, that returns the right-hand side of exercise 1(d).

   (b) Solve the problems from Exercise 1 with the Euler method. Plot and compare (discuss) the analytical and numerical solutions.

(c) Compare the results obtained by the Euler method with a more advanced numerical method (see the `scipy.integrate` package, e.g. a Runge-Kutta method). The standard usage case is given by the following example:

```
from scipy import integrate, arange
def f(x, t): return -x
x_0 = 1; t = arange(0, 5, 0.01)
x = integrate.odeint(f, x_0, t)
```

To compare the methods, choose one of the functions from exercise 1 and plot the deviation (squared error) from the analytic solution over time for both methods (choose several step sizes Δt for the Euler method). Discuss the result.

3. **Passive membrane**

The following equation describes a passive membrane subject to a current injection $I(t)$:

$$\tau_m \frac{\mathrm{d}V(t)}{\mathrm{d}t} = -V(t) + E_m + R_m I(t). \tag{3}$$

At time $t = 0$ the membrane voltage is at its resting state $V(t = 0) = E_m$. The variable τ_m describes the membrane constant, R_m is the membrane resistance, and $I(t)$ is external current injected into the membrane.

(a) Implement equation (3) using $R_m = 10^7$, $I(t) = I_0 = 1\,\mathrm{nA}$, $\tau_m = 10\,\mathrm{ms}$, $E_m = -80\,\mathrm{mV}$. Computers do not know about physical units, so you have to make a decision about the units of voltage and time you are using because these units determine the numerical values of parameters and functions. Use the Euler method to find $V(t)$ as a function of time and plot the region of interest. Label your axes with the appropriate units.

(b) Consider a sinusoidal time-dependent current

$$I(t) = I_0 \sin(2\pi\nu t) \tag{4}$$

with frequency ν and $I_0 = 1\,\mathrm{nA}$. Find and plot the numerical solutions for $V(t)$ for $\nu = 1\,\mathrm{Hz}, 10\,\mathrm{Hz}$ and $30\,\mathrm{Hz}$. Choose the respective time range corresponding to about ten signal periods. On each plot, superimpose the rescaled input $E_m + R_m I(t)$.

(c) After the initial condition has been forgotten, the voltage response approaches $A\sin(\nu t - \Delta\phi)$. Analyze how the voltage's amplitude A and phase lag $\Delta\phi$ depend on the driving frequency ν. To this end, plot the amplitude A of the resulting voltage oscillation versus frequency ν in the interval $[1\,\mathrm{Hz}, 100\,\mathrm{Hz}]$. Make also the equivalent plot for the phase differences. To check the validity of your solutions, compare them with the analytical results:

$$A(\nu) = I_0 R_m \sqrt{\frac{1}{1 + (2\pi\nu\tau_m)^2}}, \qquad \Delta\phi(\nu) = \arctan(2\pi\nu\tau_m).$$

(*Hint*: You can calculate the phase of the resulting oscillation by finding the maximum in the last period simulated, e.g. using `numpy.argmax`, and by comparing this value to the peak of the driving oscillation.)

Upload your solution to Moodle (see also submission checklist in the general information section). Please, comment your code and provide short answers to the questions in each task.

Contact	Location	Phone	Email
Gregory Knoll	Haus 2	2093 6247	gregory@bccn-berlin.de
Paula Kuokkanen	ITB, Haus 4, Room 107	2093 98407	paula.kuokkanen@hu-berlin.de
Eric Reifenstein	ITB, Haus 4, Room 013	2093 98413	eric.reifenstein@bccn-berlin.de
Natalie Schieferstein	ITB, Haus 4, Room 106	2093 98406	natalie.schieferstein@bccn-berlin.de