Assignment 1- Introduction to Mathematica and differential equations

Welcome to the Systems Biology tutorial. During this course you will solve 6 assignments that will help you repeat the material covered in the class and get some hand-on experience in computational modelling. For passing the course you must complete and hand in at least 5 out of 6 assignments.

Exercises will include computer simulations and "pen and paper" mathematics. You can use any software of your choice to run the simulations. I will provide templates written in Mathematica, which is very well suited for the purpose of this course.

Even if you are already fluent in programming, but do not know Mathematica, use this exercise to decide if it is worthwhile to learn it. Mathematica is a high-level language that is powerful for solving (differential) equations numerically, as well as algebraically. Its syntax is a bit different from other programming languages and may require some getting-used-to.

Mathematica offers a free version that can be used in the browser.

A. Introduction to Mathematica

Open "mathematica online on https://www.wolframcloud.com/ and create an account.

Click "New Notebook". The free account allows all functions needed in the course, but saving and re-opening your files to cloud or local storage is very limited. Student license are very cheap. If you want to do anything more involved, you should get such a license for the duration of the course, although it will not be strictly needed. The license will also allow you to install the software on your desktop.

Check the link "Language Intro" in the browser to get a quick tutorial and overview, and familiarize yourself with the language by testing the following functions. You can get help on each function by typing "F1" while your cursor is on the function.

Something important to know: To execute commands in a field press SHIFT+Enter. For line break without execution press ENTER.

1. Plotting function

Example:

```
Plot[x^2, {x,0,10}]
```

This command plots the function x^2 from 1 to 10.

Task: Change the function to plot $30 - x - 6x^2 + x^3$ from x = -5 to 10.

2. Algebraic Solve

Example:

```
Solve[y^2-3x+5 == 0, y]
```

This command gives the solution of the equation (first argument, `y^2-3x+5 == 0`) for the variable provided as second argument (`y`). Note the duplicated equal sign (==) . In Mathematica, one equal sign (=) is used to assign values to variables. Double equal sign is used in equations.

Task: Change the example to Solve the equation for x instead of y.

Solve (and functions below) can also be used to solve a system of equations when placed in curly brackets and separated by commas. For example:

```
Solve[{x^2-y==2, y+x==10},{x,y}]
```

3. Numeric Solve

```
NSolve[Exp[x*5]-x^2==0,x]
```

Some equations cannot be solved algebraically. NSolve provides a numerical solution of the equation for x. Note that there can also be complex solutions (of form a + b i). These are rarely relevant for our applications but keep this in mind. Change the function to solve $e^x + x == 0$ for x.

4. Differential equation solving (algebraic)

```
DSolve[{y'[t] == b - a y[t], y[0] == 0},y[t],t]//Simplify
```

The first argument is the equations system to be solved, including initial conditions (i.e. y[0] == 0). The second argument is the dependent variable (i.e. y[t]). The third variable is the independent variable (i.e., t). The term "//Simplify" converts the output in what Mathematica thinks is the easiest display of the result.

Often, we will use the following way to assign values to parameters:

```
DSolve[{y'[t] == b - a y[t], y[0] == 0}/.{b->1, a->0.5},y[t],t]//Simplify
```

"/." Means to apply what follows in curly brackets to the expression on the left. In this case: replace b with 1 and a with 0.5.

The term "//Simplify" converts the output left of "//" to what Mathematica thinks is the easiest display of the result.

You can plot the solution by the following commands:

```
sol = DSolve[{y'[t] == b - a y[t], y[0] == 0}/.{b->1, a->0.5},y[t],t]//Simplify
```

> this assigns the solution returned by DSolve to a variable called "sol".

```
Plot[y[t]/.sol, {t,0,10}]
```

> this command plots "y[t]", given the solution "sol" from t=0 to 10.

5. Differential equation solving (numerical)

Often differential equations cannot be solved algebraically, but only numerically. You can use the following command to do that. Note that you will need to also indicate the time span in which the differential equations should be solved. In this case, from t = 0 to 10.

```
sol = NDSolve[{y'[t] == b - a y[t], y[0] == 0}/.{b->1, a->0.5},y[t],{t,0,10}]
```

again, you can plot the solution as before.

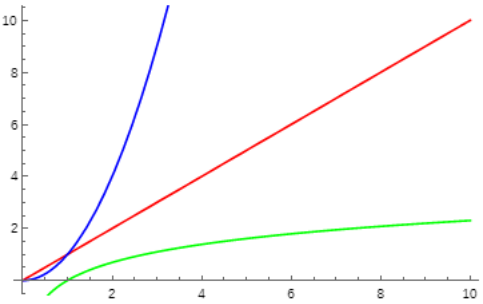
```
Plot[y[t]/.sol, {t,0,10}]
```

6. Plotting multiple functions in one graph:

To plot multiple functions in one graph, assign the plots to variables and supress their display by placing a semicolon (;) at the end. Then use the command “Show” to display all plots as in the example below.

```
In[5]:= pl1 = Plot[x, {x, 0, 10}, PlotStyle → Red];
        pl2 = Plot[x^2, {x, 0, 10}, PlotStyle → Blue];
        pl3 = Plot[Log[x], {x, 0, 10}, PlotStyle → Green];
        Show[pl1, pl2, pl3]
```

Out[8]=



7. Saving a graph to the cloud

Use the function `CloudExport` to save the plots in your Cloud home directory.

e.g. `CloudExport[pl1, "PNG", "myGraph"]`

8. Read the documentation of a function

You can directly access the documentation of any function using `?`

```
In[318]:= ? Plot
```

Out[318]=

Symbol

Plot[f, {x, x_{min}, x_{max}}] generates a plot of f as a function of x from x_{min} to x_{max}.
Plot[{f_1, f_2, …}, {x, x_{min}, x_{max}}] plots several functions f_i.
Plot[{…, w[f_i], …}, …] plots f_i with features defined by the symbolic wrapper w.
Plot[…, {x} ∈ reg] takes the variable x to be in the geometric region reg.

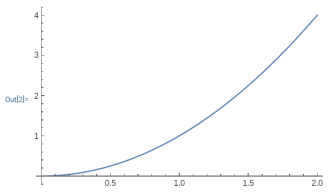
9. Commenting and making titles in your notebook

For improved readability use `(* your text here *)` to comment your code
Or use ALT + number (1 to 9) to insert headings at different levels

Excercise 1

```
In[2]:= Plot[x^2, {x, 0, 2}] (* this function plots a quadratic function in the range 0..2 *)
```

Out[2]=



B. Simulation of ON and OFF switch

In the lecture we modelled gene expression by the simple model

$$y'[t] == \beta - \alpha\, y[t]$$

a. Simulate and plot the dynamics of ON switches for production rates β = 1 mol/h, 10 mol/h, and 100 mol/h with α = 1/h

b. Simulate and plot the dynamics of ON switches for removal rates α = 1 mol/h, 10 mol/h, and 100 mol/h with β = 10/h.

c. Do the same as in (a) and (b), but for OFF switch from steady state β/α.

d. Calculate the response times for the above examples using the general analytical solution derived in the lecture.

Use this code as a starting point and modify accordingly:

```
sol1 = NDSolve[{y'[t] == β - α y[t], y[0] == 0}/.{α ->1, β ->1},y[t],{t,0,10}]
sol2 = NDSolve[{y'[t] == β - α y[t], y[0] == 0}/.{α->1, β ->10},y[t],{t,0,10}]

p1 = Plot[y[t]/.sol1, {t,0,10}, PlotStyle->Red, PlotRange->All]
p2 = Plot[y[t]/.sol2, {t,0,10}, PlotStyle->Green, PlotRange->All]
Show[p1,p2]
```

C. Separation of time scales

We used the concept of separation of time scales, ignoring the time needed to produce and degrade mRNAs. Let's see how well this assumption is justified by modelling the entire process.

```
eqs = {m'[t] == b - a*m[t], p'[t] == d*m[t]- c *p[t]}
initialconds = {m[0] == 0, p[0] == 0}
sol = DSolve[Join[eqs, initialconds], {m[t],p[t]},t]//Simplify
params = {a-> 1, b-> 2, c->1.1, d->2}
Plot[Evaluate[{m[t],p[t]}/.sol/.params], {t,0,10},PlotLegends ->{"mRNA", "Protein"}]
```

Choose parameters such that

- the degradation rate of the mRNA is 100 fold faster than the degradation rate of the protein
- the degradation rate of the mRNA is 10% faster than degradation rate of the protein
- the degradation rate of the mRNA is 10 fold slower than the degradation rate of the protein
- the same as the above three for production rates

a. What do you observe? explain in words what you see and why this happens.
b. Which of these scenarios is closest to the situation we assumed in the lecture (separation of time scales)?
c. Which rates (degradation, production rates of mRNA/protein) have to be different, so we can safely ignore mRNA dynamics and apply the separation of time scale simplification used in the lecture? When is this assumption not justified?