Imperial College London – Zambart

Workshop on *Analysing and modelling epidemic data*

Practical 2: Introducing Odin as a tool for modelling.

Dr Pablo N Perez-Guzman

Adapted from materials from Drs Lucy Okell & Lilith Whittles

The aims of the practical are:

- To familiarise yourselves with the modelling software interface odin/R.
- To understand the concept, utility and limitations of numerical integration.

In this hand-out, generally:

▶ Indicates an instruction.

▶ Indicates a useful tip or note.

▶ Indicates a question.

All practicals and projects for this short course will use a web-based interface. The underlying programming language is R. We will be using the "odin" R package with a Shiny interface during the next two weeks. A general help file about the odin interface is available in your folders and as a separate url. Each tab will have a  *(help) button which will contain more useful tips.*

Example 1: Exponential growth

► *Navigate to the odin interface* https://shiny.dide.ic.ac.uk/infectiousdiseasemodels-lusaka-2022/ *in Chrome or Safari.*

► *Click on "Practicals (Week 1), then "Go to App"* The first window you see will be the **[Editor tab]**.

The first model you will solve is that of a population whose size changes at a constant per capita rate proportional to the population size. We denote *N(t)* as the population size at time *t* and *a* the rate of change in population size.

In equation form, the model is

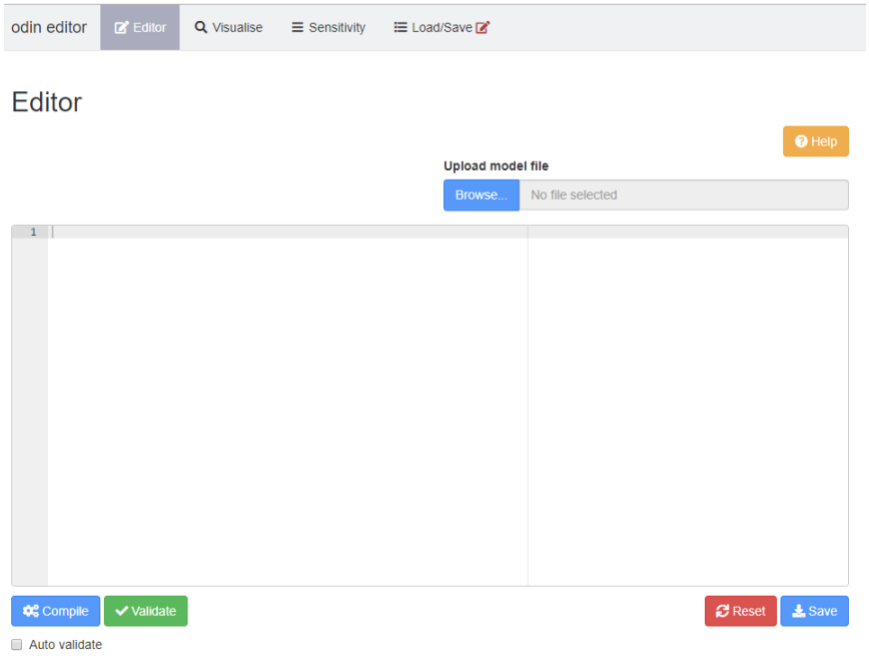
$$\frac{dN}{dt} = aN$$

with the initial condition $N = N_0$ when $t = 0$.

► *Type the following into the blank Editor window* ***exactly as follows****:*

```
# variable
deriv(N) <- a * N
```

Each variable, in this case "N" the population size, is specified by `deriv(variable)`. That is, in R language deriv(N) for dN/dt specifies that N is a quantity that changes over time.



To fully specify the model, you must also give the value of the population size at the start (in this case time 0) of the simulation (the initial value *N(0)*).

► The "#" is used to "comment out" code in the **Editor tab**. It is good practice to comment your work as much as possible so that you can understand what you did when you return to a file.

► **Caution: do not use the browser "back" button as you will lose any unsaved work.** *We recommend that you save your code regularly.*

► *Now type in the following:*

```
initial(N) <- N0
```

You must now give some numerical values to the parameters of the model.

► *Type:*

```
a <- user(0.5)
N0 <- user(1)
```

- The 'user' part means the *Shiny* "odin" app will allow you to vary the parameter.

► *Use the Validate button to check for any errors in the model code (or click the Auto validate tick-box to do this as you type). You will then see 'Validation: success' or 'Validation: error'.*

- Note that you can load code from an existing model from file (from a previous practical, for example) using Browse and edit it if you wish, rather than write code from scratch.

► *When you are happy with your model, click "Compile".*

- If there are any errors, fix them and compile again until you have no more errors.

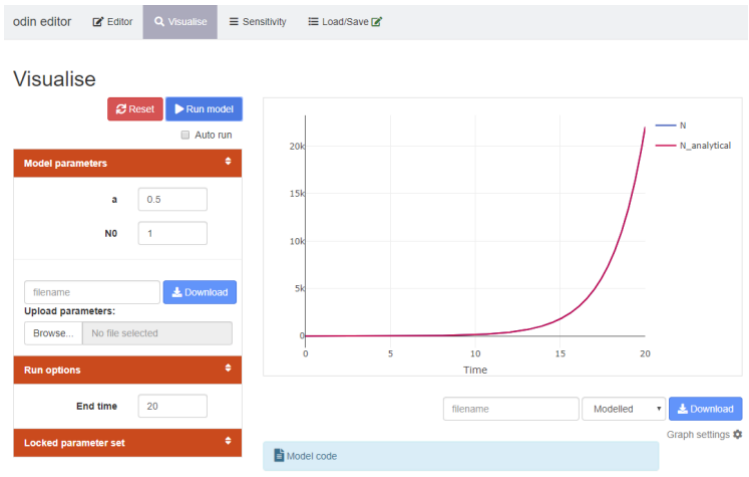
► You can save the model code using the Save button just below the code editor. This will download a file with extension ".R" containing your model code (which is usually stored in your local 'Downloads' folder depending on your browser and computer set up. Don't forget to save your work. NB disconnection from the internet can erase the current model if unsaved).

► *In the **Visualise tab** you can visualise the model output. Notice on the left under 'Model parameters', the values you entered for N0 and a are displayed.*

► *Under "Run options" specify the "End time" by typing 20.*

- This is how long the model will be run for (number of time steps).

► *Click Run model to visualise your model outputs*.



- Initially this will use default values for parameters, but you can manually change parameter values to see how they affect your model dynamics.

► *In the **Visualise tab** under "Model parameters", change the value of* a *from 0.5 to 0. Click Run model again to visualise your model outputs after changing your parameter values.*

- The growth rate is 0. The population size N, as seen in the Run window, is now constant over time.

► *Change the value of **a** again, this time to **-0.5** Click 'Run model'.*

- The population size N, as seen in the Run window, now declines exponentially. *a* is the decay rate.

► *At the top of the **Visualise tab**, click on 'Reset'.*

- This resets the value to the value of the parameter 'a' to be the same as in the code **Editor tab**, i.e. 0.5. This will also "reset" the "End time" so remember to re-enter it.

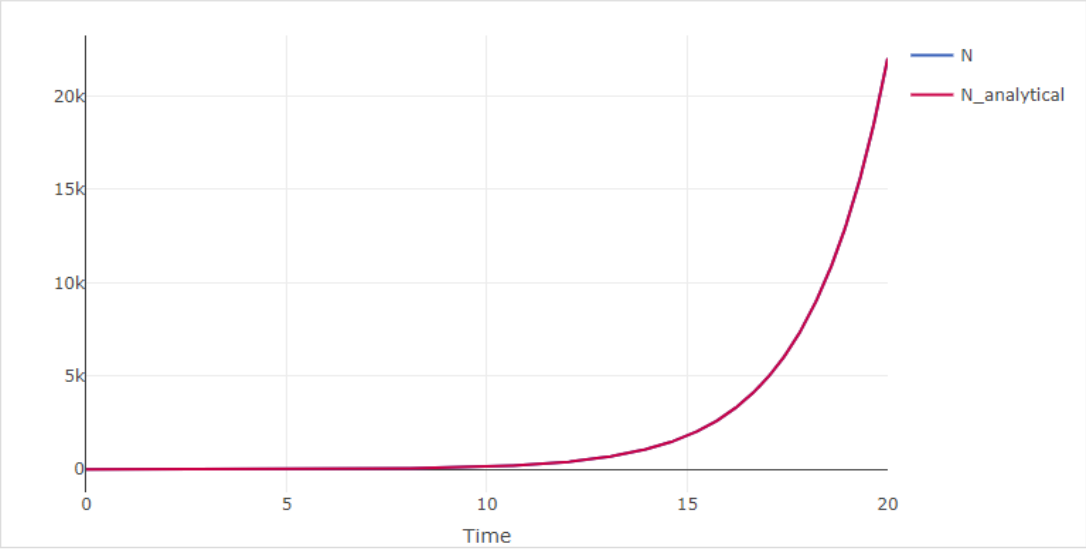
► *In the **Editor tab**, add an additional output variable. Type:*

```
output(N_analytical) < - N0 * exp(a * t)
```

Use 'output(variable)' for variables which do not need to be calculated by a differential equation. Here "t" is implicitly time and does not have to be defined.

Compile, then in the ***Visualise tab,*** run the model. You should see the output "N_analytical". Click on 'N_analytical' at the top right of the graph to remove it from the plot, then click again to add it back. Compare it with 'N' - what do you notice?

► In general, to add or remove variables on the plot, simply click the variable name in the top right of the graph.



► *See the Appendix at the end of this practical if you are interested in the derivation of N_analytical.*

► You can separately save parameter sets or tables of model outputs from the **Visualise tab** (using the Download buttons in the bottom right corner or at the bottom of the model parameters section).

► You can save plots by clicking on the icon. This will download a .png file of your plot.

► *From the 'Graph Settings" underneath the plot window* Graph settings *, select 'log scale y-axis'*

Now, with a log-scale axis, the population changes as a straight line. Notice however how the large numbers on the left axis now increase in multiples of 10. This is in fact the definition of a log-scale: exponential growth and decline appears as a straight line. If you want to see whether your data grows or decline exponentially, plot it on a log scale.

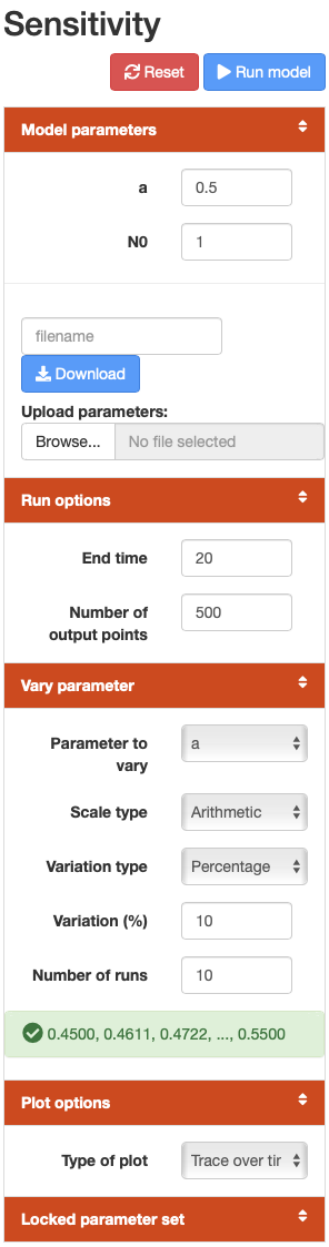
► The "graph settings" dialog box under the plot will also let you choose whether to plot variables on the right or left axis.

► Clicking on "Model Code" underneath the plot will expand a window that allows you to see your model code in the "**Editor tab**" in the same window as your model run.

Running sensitivity analysis

► *In the **Sensitivity tab** under "Model parameters", vary the value of a from -10% to +10%. Remember to set a value for "End time" (e.g. to 20). Click Run model to visualise your model outputs.*

- Note that, to visualise an output variable, you will need to click the variable name in the top right of the graph.



Example 2. *Modelling an infected cohort*

As you saw in practical 1 today, it is always good practice to start your model simple and then build in complexity as and when needed. The simplest compartmental model you can conceptualise is that of an infected cohort.

Note that this is not yet a transmission model (i.e. there is no infectious disease transmission going on). Rather, this is just simulating a cohort of infected people that will go on to recover at an exponential *per capita* rate γ.

Step 1: the model.

This basic cohort model has only two compartments, *I* and *R*, and is written:

$$\frac{dI}{dt} = -\gamma I$$

$$\frac{dR}{dt} = \gamma I$$

Recall that:

- Mean duration of infectiousness = $1 / \gamma$
- So, the inverse will be the recovery rate

$$\gamma = \frac{1}{mean\ duration\ of\ infectiousness}$$

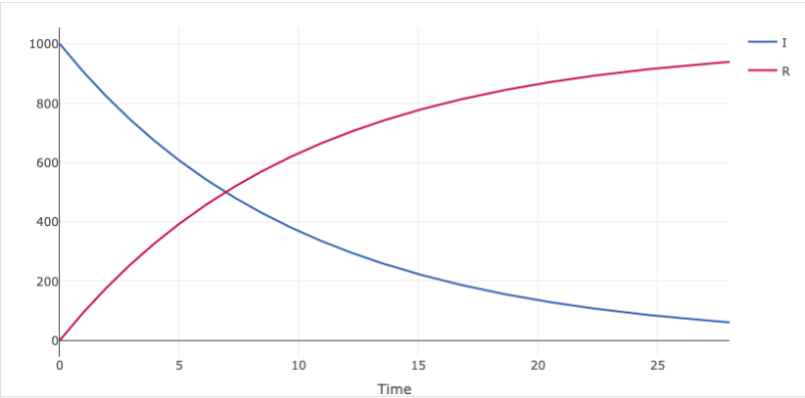
We are going to write the model from scratch.

► *Navigate back to the odin interface* https://shiny.dide.ic.ac.uk/infectiousdiseasemodels-lusaka-2022/ *in Chrome or Safari.*

► *Open a new* ***Editor tab*** *and write out the equations for the model above. Each state variable (i.e. I and R) should be initialised (e.g.* initial(S) < - ...*) and defined (e.g. with the* deriv(I) < - ...*). The initial values add up to the total population size, for example 1,000 (you can choose a number). Variables can take any name (eg.* gamma *for* γ*) just as long as there is no repetition and that they are given a value (see section below).*

- Assume the time to recovery from this infection is, on average, 10 days

► *Once you are happy with your code, go to the* ***Visualise tab*** *and click "Run model". You should see something like this*



Appendix. Population with constant per-capita growth rate model: analytic solution

In the first part of the practical, you were given a model of constant per-capita population growth. This is an example of a model which can be solved analytically (i.e. by hand using algebra) as well as numerically (e.g. using odin), unlike more complex compartmental models (e.g. SIR model) which we will cover later in this workshop.

Here we show workings for solving a differential equation model via integration to derive the analytic solution shown in the lecture:

The model equation is:

$$\frac{dN}{dt} = aN$$

where *N* is the number of individuals in the population, *t* is time, and *a* is the constant per-capita growth rate.

First, we rearrange to put all the terms containing *N* on one side of the equation (multiply both sides by *dt*, and divide both sides by *N*):

$$\frac{1}{N}dN = adt$$

Then apply the indefinite integral to both sides:

$$\int \frac{1}{N}dN = \int adt$$

Note that on the left hand side equation, we will integrate with respect to *N*, whilst on the right hand side, we will integrate with respect to *t*.

$$\int \tfrac{1}{N}dN = \ln(N) \text{ and } \int adt = at$$

Using these standard integrals, we obtain $\ln(N) = at + C$, where *C* is the constant of integration. We exponentiate both sides of the equation to obtain:

$$\mathrm{N} = e^{at+C}$$

By the laws of indices and by noting that e^C is still a constant, we can rewrite this as:

$$\mathrm{N} = e^{at}C$$

Now we set initial value (boundary) conditions in order to solve for the constant C. To obtain a simple result, we choose *t = 0*, which gives the following equation for *N* at time *0*:

$$N(0) = e^{a\cdot 0}C$$

Since $e^{a\cdot 0} = e^0 = 1$, $C = N(0)$ and replacing C in our above equation with N(0), we obtain our general solution:

$$N = N(0)e^{at}$$