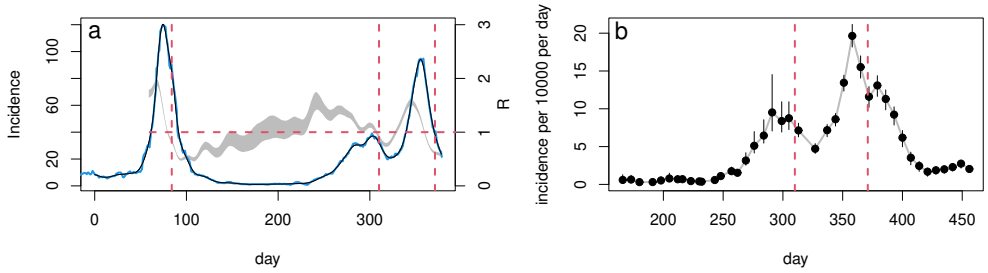
Practical 2: Covid and the hazards of opportunistic sampling

This practical is to be completed in groups of 3. What is submitted must be solely the work of the members of the submitting group. Code must not be shared between groups. We have automatic systems available for checking for this, but in addition students tend to make distinctive errors in coding, or use distinctively convoluted solutions to problems: these tend to stand out even if you do the obvious things to try and hide the code sharing. You can use code from the lecture notes without citation, but if you use any other code that you did not write yourself you should cite where it came from.

This practical is less prescriptive than the previous one. In the previous practical the programming steps were broken down for you. Here you have to do a bit more of that yourself. However the practical should be straightforward if you are up to date with the lecture material, and have put in the effort to understand it properly. The assignment has the 'real world' feature that you have to spend some pencil and paper time translating the problem description into something you can code (albeit the problem description in fact describes what is needed with unusual precision for the real world). Again, coding should use only base R (recommended packages and `ggplot2` are fine, although I don't think any are needed).

An interesting feature of the Covid-19 epidemic in the UK is that the estimates of the number of new cases per day based on properly sampled statistical surveys, or based on properly conducted reconstructions from death data, show infections in substantial decline before any of the three national lockdowns. For example the figure below shows new cases per day – *incidence* – plotted against time (day of year since Dec 31 2019), based on two separate surveys. On the left, REACT-2 asked people who tested positive for Covid in a random sample, when they had first experienced symptoms. Lagging the numbers experiencing symptom onset each day by the published average time from infection to symptom onset gives the blue/black incidence curve (grey is the corresponding 'R number'). On the right are the ONS published estimates from their random PCR test surveillance survey. The vertical dashed lines mark the start of full lockdowns.



These figures coincide closely with the reconstructions that result from applying a statistical deconvolution to the daily Covid deaths data, to work out when the corresponding infections occurred.

The advantage of these three estimates is that they are based on statistically valid measurements, where the relation between the data and the population being sampled is fairly clear. REACT-2 took random samples of the UK population, and of those that had had Covid, collected their time of symptom onset. Clearly the numbers with symptom onset on any day should be proportional to the total numbers in the UK with sympton onset. The ONS estimates are again based on a representative random sample of the UK population. The death data, although in some sense a census, can be viewed as a random sample from the population of death trajectories that could have occurred, given the infection to death time distribution, and a particular incidence trajectory.

But there is another source of data that could be used, and shows apparently similar patterns, albeit with the timings not exactly matching. These are data from the ZOE symptom tracker app. This is an app that people downloaded to log covid like symptoms, and over a million people did so. Such data also allows symptom onset trajectories to be reconstructed, but there is a catch. The ZOE sample is **not** a random sample of the UK population, so we can not know how its results will scale to the whole population. Sample size is **not** any sort of guarantee of representativeness.

One particular issue of concern is this: it is highly likely that people who choose to download a Covid symptom tracking app are more concerned about Covid than the average person. In that case they are likely to take more precautions than average to avoid catching or transmitting it. What might that mean about the representativeness of symptom onset trajectories from such a group? Will they track the whole population trajectory, or do something different? Sharpening this question to one that can be answered: what do the daily infection trajectories look like

for the 10% (say) of the population with the lowest individual transmission probabilities, and how do they compare to the infection trajectories for the whole population?

This is a question that can usefully be addressed by simulation. In particular suppose that the ith person in the whole population has a relative contact rate with other people of β_i, so that the daily chance of a transmission between infected person i and uninfected person j is $\beta_i\beta_j\lambda$ where λ is an overall viral infectivity parameter. The average value of β_i over the whole population is 1. Each day an uninfected person, j, has a probability $\lambda\beta_j \sum_{i\in\mathcal{I}} \beta_i$ of being infected, and entering the exposed (E) state. $\mathcal{I}$ is the set of all currently infected people. Once exposed they have a daily probability of $1/3$ of entering the infectious state. The infectious have a daily probability of 1/5 of leaving the infectious state (recovery or transition to serious disease). Over the course of the simulation we do not want to consider re-infection.

Write a function to simulate from this model assuming the following default values:

1. The initial susceptible population is of size, n. Set $n = 5.5$ million (the population of Scotland - we should really do the whole UK, but that makes simulation slightly more time-consuming than is helpful here).
2. Generate the β_i using `beta <- rlnorm(n,0,0.5); beta <- beta/mean(beta).`
3. Assume $\lambda = 0.4/n$.
4. Start the epidemic by setting 10 randomly chosen people to the E state.

Note that although the simulation principles are similar, this model is not the same as the SEIR example in the notes. There are important differences.

Simulating for 100 model days, you should record the number of new infections each day, and the number of new infections among the 10% of the population with the lowest β_i values. Also record the number of new infection in a random sample of 0.1% of the population.

Using the `plot`, `lines` and `text` functions produce a plot showing how the daily infection trajectories compare between the whole population, the 'cautious 10%' and the 0.1% random sample. You will need to suitably standardize each trajectory to plot them on the same plot. Label the plot so that it is clear which trajectory is which, and also provide labels giving the day on which each trajectory peaks.

Finally write code to visualize the variability in the results from simulation to simulation, by running 10 replicate simulations, and suitably plotting these.

In the code comments, write a couple of lines on what the implications of these results might be for interpreting daily infection trajectories reconstructed using the ZOE app data.

What to submit: One text file of carefully commented code for producing the required simulations and plots should be submitted on Learn. The file should be called WGxxP2.R where 'xx' is replaced by your workgroup number (I don't care about the case: wgxxp2.r is also fine). The first line of the code should include your group number and the names of the people in your group. The second line should include the address of your github repo. You must also send an invite to your github repo to `simonnwood`. Do not modify the repo after submission. The deadline is 16:00 Friday 22nd October. It is strongly recommended that you submit well in advance of this, as last minute computer glitches will not be taken as an excuse for late submission[1].

Marking Scheme: Full marks will be obtained for code that:

1. does what it is supposed to do, and has been coded in R approximately as indicated (that is marks will be lost for simply finding a package or online code that simplifies the task for you).
2. is carefully commented, so that someone reviewing the code can easily tell exactly what it is for, what it is doing and how it is doing it without having read this sheet, or knowing anything else about the code. Note that *easily tell* implies that the comments must also be as clear and *concise* as possible. You should assume that the reader knows basic R, but not that they know exactly what every function in R does.
3. is well structured and laid out, so that the code itself, and its underlying logic, are easy to follow.
4. is reasonably efficient. As a rough guide one simulation run should take less than half a minute to run - much longer than that and something is probably wrong.

[1] This is for reasons of fairness - otherwise it becomes too easy to game the system for an extension.

5. includes a comment briefly discussing the implications of the results for using incidence reconstructed from the ZOE data, in a way that makes sense.

6. was prepared collaboratively using git and github in a group of 3.

7. has been tidied up so that it is a 'finished product' - no commented out experimental code, code purely for your own checking purposes, or printing out of large objects. Put functions first, and code using the functions after that.