## Simulated Outbreak Exercise

|  |  |
| --- | --- |
| Scenario Date: | Tuesday 10 October (13:30-15:00) |
| Inject No: | 09 |
| Inject time: | 1.5 hours |
| Inject Title: | Data import and cleaning |

Facilitator’s note: Highlight to fellows that there are going to be a few injects where we will be working with data in R. So they don’t stress unnecessary, reassure them that we will give them a document with the R scripts midway the inject, so they can look at an example of a solution. Remind them that there are always different ways to get to the same solution in R.

Encourage them to look at their previous code, as well as at the R handbook and google.

In general, for all injects, remember that we want the fellows to manage on their own/as a group. You, as facilitator, should intervene only if they are drifting off, or if you feel they are forgetting or getting confused with basic/important concepts. But please, let them try on their own/as a group first. Curb your enthusiasm to jump in! Let them explore, and try, and think without your help.

Facilitators’ notes are a helping tool for you to assess whether the fellows are discussing the most important points. If you think you need to intervene, probe with questions so they can find the answer as a group. I am a fun of [Socratic questioning - Wikipedia](https://en.wikipedia.org/wiki/Socratic_questioning) to encourage deep-level reflection and critical thinking.

1. **Learning outcomes:**At the end of the session, participants will be able to:

- Import datasets in R  
- Apply basic data management techniques to explore and familiarise themselves with the dataset

- Apply strategies for data cleaning

1. **Story/plot description:**  
   You and your team have carried out hundreds of interviews using KoboCollect in a very short time. Well done!

You will now have your first look at the data. Note that in real life you would have ideally done a pilot of the questionnaire and have prepared in advance a codebook and be familiar with the names and format of all variables.

You have 1.5h to explore and clean the data you have collected and save a new version of the cleaned dataset in your computer. Be sure to make use of the R code that was provided to you last week, during the R practicals. You can also use the code you wrote during the Applied Epi course. You are expected to look at the help functions and google as well. Try to understand what the functions are doing and their arguments are.

Work on your own computer and help your fellow colleagues as needed. The lead of the outbreak will be in charge of keeping the time (unless you designate another person, but be sure that someone keeps track of time).

Open R Studio and set up a new R project, and create a data and a scripts folder.

Facilitator’s note: Note that this will be the first R exercise of the SimOb. Fellows had an “R introductory course” delivered by AppliedEpi that will cover the R functions used during the SimOb. Moreover, during the second week of the introductory course, they had two sessions that focused on using R to analyse outbreak data. Fellows are encouraged to use and adapt their previous code from these training opportunities into the SimOb, so they should be able to do these exercises almost independently. Fellows in the group with a high level of R-skills are encouraged to help their colleagues. R facilitators are available for questions. Remember to refer to them to the Rhandbook, ask colleagues for help, and in google.

1. **Questions/assignments:**

3.1 Install packages and load libraries.

* Load the R packages you will be using. Update this list of packages as you dig into the code: You may realise you need packages you didn't think about at the beginning. Remember that it is a good practice to load tydiverse at the end of your package list, to avoid [masking functions](https://www.ucl.ac.uk/~uctqiax/PUBLG100/2015/faq/masking.html).

3.2 Import your data.

* You can find the Copenhagen\_raw.csv file in EVA, under today’s folder. Feel free to explore the codebook as well.

3.3 Explore and clean your data.

* Have a look at the structure of your data. Here you have some questions that may help you get started:
  + How many observations and variables does the dataset contain?
  + What types of variables do you have and what types of values are recorded?
  + Do any of the values of the other variables look implausible? Which ones and why? What will you do about it?
* Go ahead and clean your data as you see fit.

Facilitator’s note: Most answers to these questions are in the “SolutionR\_Inj09”.

* As an example of implausible values: there are some age issues (ages that are likely to be typographic errors. Specifically: There is one teacher aged 16 (likely digit reversal - should be 61); There is one student aged 8 (likely missing a digit - should be 18); There is one student aged 180 (likely has an extra digit - should be 18). **NOTE**: The code we provide to them cleans most of the data.
* There are still some things that are not fixed with the code, for simplicity. Thus, the fellows may find errors in the data that are not fixed with the code we provide to them.
* Remind fellows that the solution code is an example of many different ways these could be done. If they’ve done it differently, doesn’t mean that it is wrong, not at all!



Hint: you could use head(), dim(), str(), or skim() to have a quick look at the data and get an idea of which variables you want to explore further. You can also do some tables.

Facilitator’s notes: Have the fellow discuss what the outputs of these functions show.

You could highlight that there are many missing values, probably because people who completed the questionnaire only checked the items of what they ate, or symptoms they had (they didn’t mark those they hadn’t). A face to face interview may have yielded less missing values in these questions. The questionnaire structure is very important to avoid unnecessary missing values.

Create summary table for the dose response columns (those finishing with a capital "D") to have a look at their distribution.



Hint: use gtsummary::tbl\_summary().

* There are some "column types" you would like to modify because either R needs some specific types of data when working with certain variables, or because it will be easier for you to visualise and interpret the data.

For example, the variable “group” is encoded as 0 and 1, when you create a table later on, you don’t want to see 0’s and 1’s because that doesn’t mean anything. It is better to change this variable to a factor, with explicit labels, so they actually mean something when they come out in your table.

You can use mutate to modify the type of the following variables:

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | Original | Desired | Hint: you could use… |
| sex | character | factor | mutate(), as.factor() |
| group | integer | factor |
| class | integer | factor |
| *All the clinical symptoms variables* | integer | logical | mutate(across()), as.logical() |
| meal | integer | logical |
| *All the food variables* | integer | logical |
| *Dose response: All the food variables representing amount eaten (those finishing with a capital "D") –* | integer | factor | - |
| dayonset | character | Date | lubridate::dmy() |
| starthour and dayonset together | integer | Date-time (POSIXct, POSIXt) | lubridate::ymd\_h() could have inside stringr::str\_glue() with dayonset and starthour |

* Why may you want to transform the variable starthour to date-time?

Facilitator’s note: Having a variable that defines “time” in an outbreak investigation can be very useful when creating a case definition and an epicurve. An hour of the day, without a day associated with it doesn’t help you much, thus, you should merge together into one variable day and time of onset of symptoms. Moreover, you will be using this combined variable later on to estimate an incubation period create your epicurve.

For some food outbreaks, if you have the date-time of exposure to the culprit food, and you have the date-time of the onset of clinical symptoms, you may be able estimate an incubation time and thus, hypothesise about the type of pathogen you are dealing with.

3.5 Export clean data.

* Save your clean data as a new file “Copenhagen\_clean1\_YOURINITIALS.rds”, under the data folder of your Rproject.



Hint: use rio::export()

**Notes for facilitators:**  
At some point during each R-inject (we recommend this happens in the middle of the exercise, so for this inject, which lasts 1.5h, that would be 45 min), exact time depending on the group, the facilitator will provide fellows with the final example code, so fellows can go through it and ask any question they may have about that code.

If the fellow’s laptop’s language settings are not English, it might not recognize the `ymd\_hm` input. They can either use a `ymd\_h` or `ymd\_hms` input, or switch their settings to English by running:

# Check current system settings:

Sys.getlocale()

# Set time “in English”:

Sys.setlocale("LC\_TIME", "English")

1. **Scenario for the role play:**  
   N/A
2. **Timing:**  
   3 hours
3. **Need for materials (logistics)**
4. **Deliverables**  
   A script with reproducible code and clean dataset named **Copenhagen\_clean1\_YOURINITIALS**.