INST0065 Practical Tasks – Week 1

This week's practical consists of two main tasks; firstly planning a manual data visualization, and secondly checking that you can use R, in preparation for subsequent practicals. We will start however with a short ice-breaker task, so that students can become more familiar with their classmates.

You will be allocated into small groups for the practical session, and most tasks will be based around these groups.

Ice – breaker task

In the first part of the practical session, you will be divided into break out rooms, so that you can meet your fellow students. This module is part of two programmes – *Digital Humanities* and *Information Science*. It is therefore likely that you will already know some of the people in your group, but not all.

Some of you are based in the UK, some of you are not. It will be useful, for planning future sessions, to have a good idea of the time zones in which you are all located.

The first task, therefore, is to collect data on which time zone you are based in.

You will have **five minutes in the breakout room** in which you should *introduce yourself to each other, and collect information about time zones*, in particular offsets from the time in London, where UCL is located. You may find the website https://www.timeanddate.com/worldclock/ useful to confirm this.

Within the group, you should also *choose a group member who will report back* to the whole class summarising this time zone data. This should be reported in aggregate – e.g. "We have three group members in the UK, two who are in China which is 8 hours ahead of London, and one member who is in Los Angeles which is 8 hours behind London."

We can then pool this information to create a graph for the whole class.

Creating a manual data visualization

Our first main task this week is to create a data visualization, inspired by those published in the book 'Dear Data' (Lupi and Posavec, 2016).

'Dear Data' tells the story of how its two authors - Giorgia Lupi and Stefanie Posavec spent a year exchanging postcards on which they drew a visualization documenting aspects of their life. Each week they would choose a topic, collect data, and then decide how to visualize it.

The project has been widely described and documented; you can learn more about it here:

- A keynote speech by Lupi and Posavec: https://vimeo.com/157474716
- A podcast: https://www.wnycstudios.org/podcasts/notetoself/episodes/dear-data-diary
- Another podcast: https://fivethirtyeight.com/features/dear-data-and-fivethirtyeight-want-you-to-visualize-your-podcast-habits/

In the project, Lupi and Posavec specifically focussed on 'small data'. Looking at the first few weeks of the project (http://www.dear-data.com/by-week/) can help to illustrate this, and also some of the data collection 'rules' that were used:

- Week 1: 'a week of clocks' – recording when each author looked at the time
- Week 2: 'a week of public transportation' – data on walks and public transport use
- Week 3: 'a week of thank yous' – when the authors thanked or were thanked by someone

This week, I'd like you to follow a similar model of observing data around you, and then drawing a visualization of it. The visualization does not need to be complex or artistically sophisticated, although of course you are free to present it however you wish.

For many of us, the idea of 'observing the world around us' is emotive, with our everyday activities curtailed due to the covid-19 pandemic. We will need to consider data that can be collected within the home.

The first aim will be choose a topic that group members will focus on. Each group member will then collect data on that topic. You should therefore try to find data collection topics that can be done relatively easily (i.e. data collection will not take a lot of time), and can be done by all members of the group (or, a version of it can be done by everyone), taking into account the places in which people live, pandemic restrictions, and other considerations such as mobility and access.

Each group will be given 10 minutes in a breakout room to discuss ideas on, and then *choose what data to collect*. As with the ice-breaker task, you should *select a group member to report back to the class*. This may be the same person as before, or a different person.

You will then need to think in more detail about how to collect the data, and how you will visualize it. Each group member should choose their own way to represent the data.

Over the following week, you should **collect relevant data**. This should be assembled in to data visualization (typically, hand drawn) and a photo of it should be posted on the INST0065 module Moodle forum.

Using R and R Studio

The second main task is to check that you can use R and R Studio. R Studio is separate from R – it provides an environment in which to use R, but it does not include R – instead, it requires that R is already installed.

R is available from https://www.r-project.org/ for a variety of platforms including Windows, iOS and Linux; it is already installed on the 'Desktop@UCL Anywhere' remote desktop. These notes will cover both using R via the remote desktop, and installing it yourself. You should ensure that you can use R via one of these methods, but do not necessarily need to do both.

Using the Desktop@UCL Anywhere' remote desktop

The UCL remote desktop should be accessible to all students, and information on how to access it can be found at::

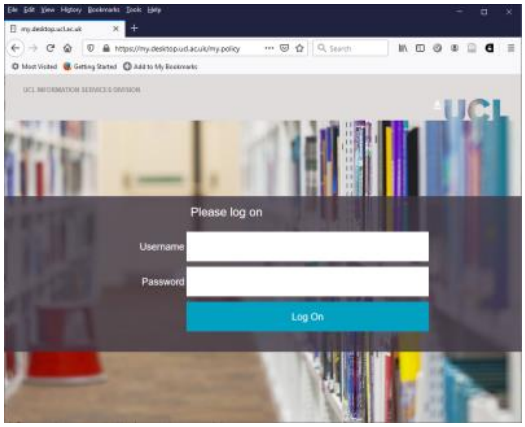
https://www.ucl.ac.uk/isd/services/computers/remote-access/desktopucl-anywhere

You may need to download and install 'Citrix Workspace'.

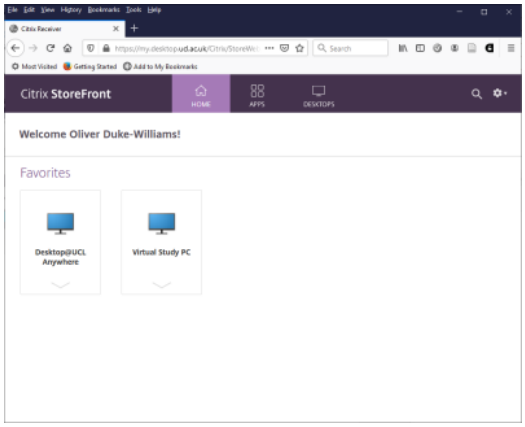
Students who are based in China may find that it is easier to connect using the UCL China Connect service; information about this is available here: https://www.ucl.ac.uk/isd/how-to/connect-to-ucl-china-connect-service

When using the Desktop@UCL service, you should follow these steps:

1. Go to https://www.ucl.ac.uk/isd/services/computers/remote-access/desktopucl-anywhere
2. If you have not previously done so, follow the 'Download and install Citrix Workspace' instructions
3. Click on 'Log in to Desktop@UCL Anywhere': https://my.desktop.ucl.ac.uk/ (this can be done without step 1)
4. Log in using your UCL username and password

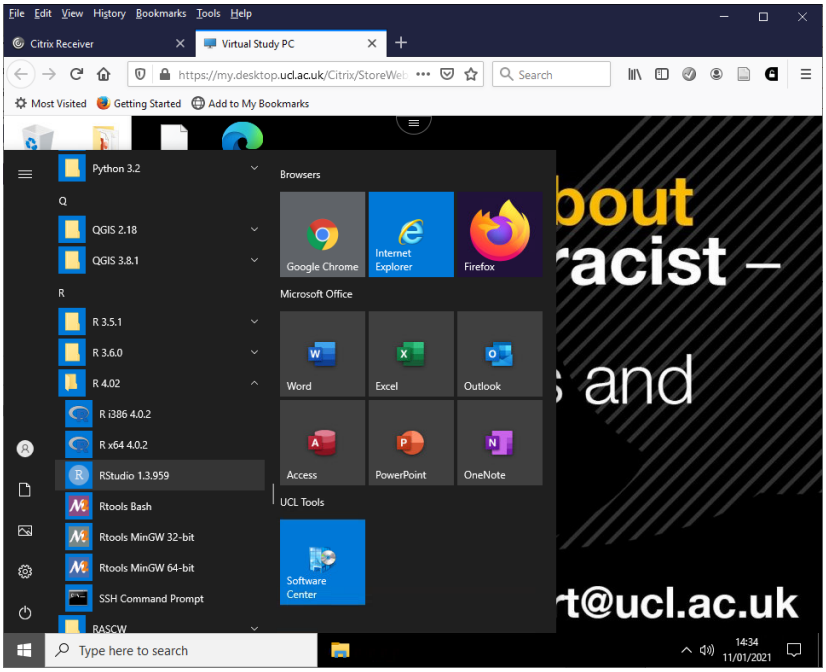


5. You will see the option 'Desktop@UCL Anywhere', and might see 'Virtual Study PC' (this should be available to you during the time of the practical session. The 'Virtual Study PC' should be more responsive, so use this if available. Use 'Desktop@UCL Anywhere' if the virtual study pc link is not shown.

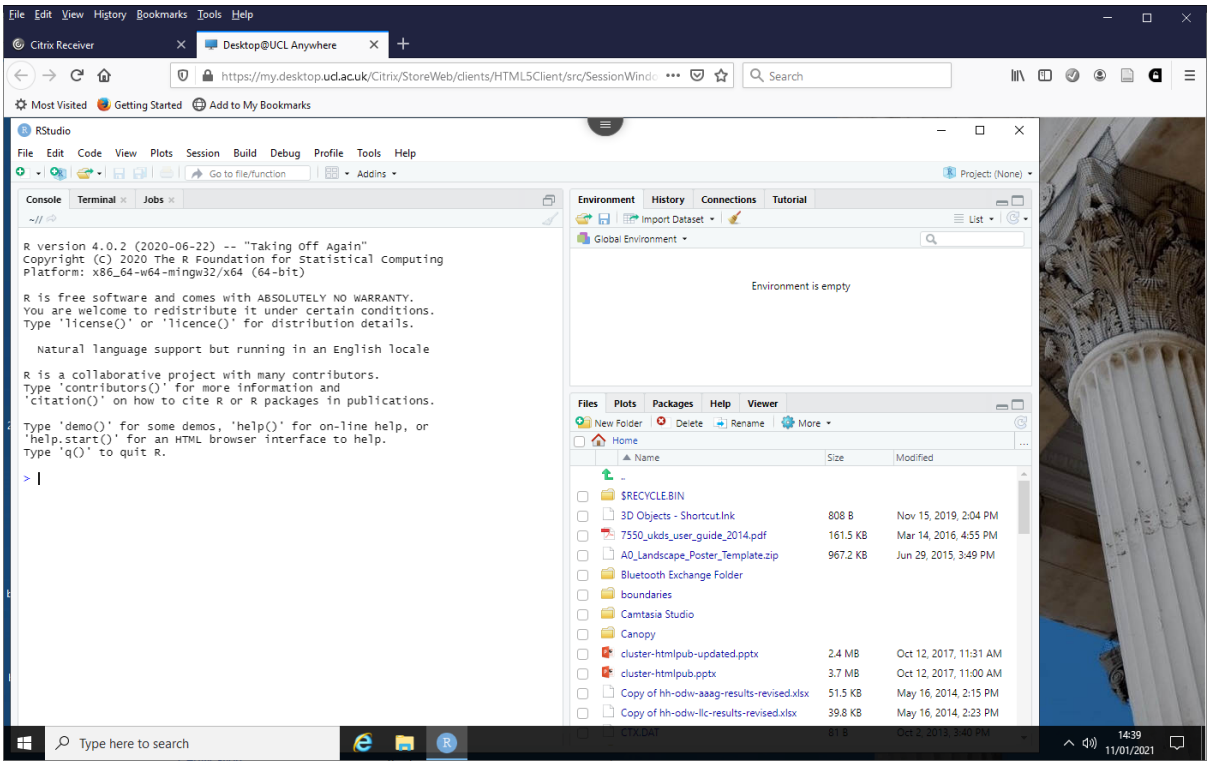


6. This will open a remote desktop (it will take a little while to complete). Click on the windows icon at the bottom left. You should see a list of applications. Scroll down to find 'R'. Select the version with the highest version number (at present, 4.0.2) and then click on the ' R

Studio' option.



7. This will cause R Studio to open. This part of the instructions is now complete - the section 'Using R' below gives some sample commands to try.



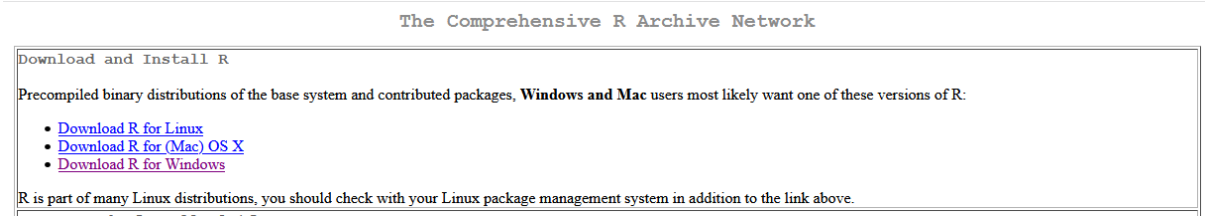
Downloading and installing R and RStudio

If you do not wish to, or are unable to use the UCL remote desktop, you can also install R on your own machine. You might find that this is a more reliable (and less prone to network issues).

In order to do this, you need to download and install R and RStudio separately.

To download R, follow these steps:

1. Go to https://cran.r-project.org/mirrors.html This is a list of mirrors, from which R can be downloaded; you can use any, but it should be fastest to download from a mirror near to where you are located
2. After selecting a mirror, you should see a list of links for different operating systems. You should select the appropriate link; these notes illustrate the path using Windows


The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

3. You should only need the basic package; for Windows this is 'base', for Mac OS it is (currently) 'R-4.0.3.pkg'. The Mac OS version should download directly, the Windows 'base' link takes you to a further screen in which the difference between the 32 and 64 bit versions is described.
4. The software should trigger the standard installation process when it downloads. (Depending on your machine configuration, you might need to locate the download and double click it, or it might open an installation window after it has downloaded).
5. When you have installed R, you will then need to install RStudio. Go to: https://rstudio.com/products/rstudio/download/#download and you should be prompted to download an appropriate version of R Studio desktop for your machine.
6. Again, download and install the RStudio as you would normally do with other software.
7. After installing RStudio, you should then find it and start it in your list of applications. RStudio should open as in the image above for the remote desktop approach. Note that the console message should show you the version number of R that you have just installed.

Using R

We will explore R properly next week, but for now you can use some sample commands just to check that things work. These commands will work in the same way whether you are using R on your own machine, or on the remote desktop.

Commands in R are typed at the prompt; you will see this in the left hand 'console' pane in RStudio. You should see the prompt '>' and a flashing cursor.

Type: 1+2

This is a very simple expression. You should see the result:

[1] 3

Type: x <- 1+2

This is an assignment operation: you are asking R to calculate '1+2', and then assign the answer to the object x. You will not see an immediate result as above, but will be shown the prompt again. You may notice that 'x' is now listed in the 'environment' pane on the right hand side.

Now, type: x

You will see the result:

[1] 3

If we type the name of an object, we will see the value of that object. We can perform operations on the value.

Type: x*2

You will see the result:

[1] 6

This is the result of x * 2, you should note that we have not changed the value of *x* itself.

You should experiment with other calculations, but for the moment we just want to check that R actually works. We can try reproducing a simple plotted graph shown in this week's slides.

Type: t <- seq(1,10,0.1)

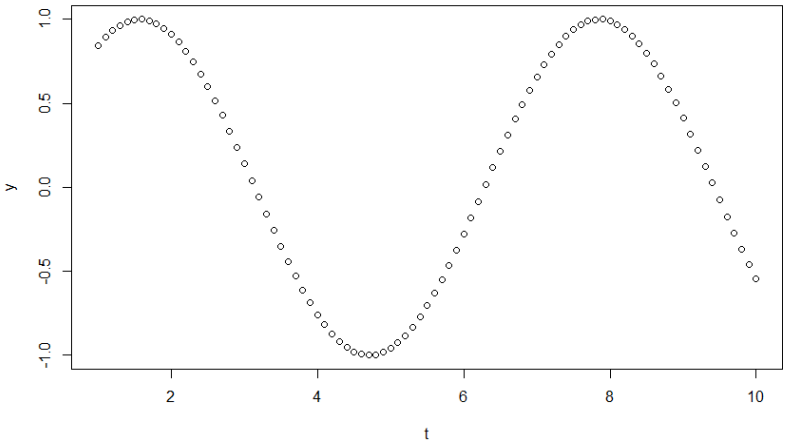
This creates a sequence of numbers (assigned to *t*) from 1 to 10, with an increment of 0.1 between values.

Type: y <- sin(t)

This creates a second vector of numbers, *y*. y has the same number of values as t, and each value is the value of sin(t) (That is, the first value of y is sin(1), the second value is sin (1.1) and so on).

Now, type: plot(t,y)

This creates a simple plot of t against y:



Try plotting some other values. R has many in-built functions – as well as sin(), try cos(), tan(), atan() etc etc – try using some of these. You might also try changing the parameters in the seq() function.