Week 1: Introduction to the course (and R)

Charlotte Hadley

1. Getting to know me

2. Getting to know you

3. Understanding the goals of the course

4. Why are we learning R?

Getting to know me

Charlotte Hadley

Please call me Charlotte or Charlie. My pronouns are she/her.

I don't have a doctorate so it's inaccurate to call me "Dr Hadley" or "Professor Hadley".

If you absolutely must call be by a title your only option is "Miss Hadley" but I'd really prefer you didn't.

What do I do?

I'm currently a full-time independent data science consultant and trainer through Visible Data Ltd.

Academic background

- 2015-2019: I worked at University of Oxford as a Research Support Officer and built the Interactive Data Network.
- 2010-2012: I began but ultimately quit a PhD in biomineralisation.
- 2006-2010: MPhys and BSc in physics from University of Leeds with a focus on biophysics

Industry background

- 2016-2022: I've been consulting and delivering training in industry both in-person and via LinkedIn Learning..
- 2012-2015: I was a senior consultant at Wolfram Research

Getting to know you

Can you tell me a little bit about yourselves?

- Your name
- Your pronouns
- Where you're from
- What have you studied before this Masters?

Introducing Etherpad

We're going to use "Etherpads" during lectures and workshops so I can ask you questions and share code.

Here's the link for today's pad: bit.ly/eng7218_week-1_lecture-slides

How does this course work?

Course Structure

We have 11 weeks of teaching and each week has:

- an 2 hour lecture
  - In the lectures I will introduce topics and theory
  - Two hours is too long for most people to pay 100% attention.
    I'll insert breaks and experiment with other ways to break up the lectures.
- an 2 hour workshop
  - These workshops are **crucial**[1] for you to do well in the workshops.
  - The workshops will mix together guided and exploratory work.

[1] Of course, the real world means that 100% attendance is an unrealistic expectation. **Please** do get in touch with me if you miss workshops or lectures and I'll help as much as I can.

Laptops

This is a practical data science course - please think of the whole course as a lab.

Please **bring your laptop to every lecture and workshop**.

If this isn't practical for you **please** speak to me[1] and I will find a solution.

In today's workshop I will take you through all of the steps necessary to setup your machine to use R and RStudio.

[1] See next slide!

Asking questions

> If you have a question during the lectures or workshops **please ask the question** when you think of them.
>
> There's no such thing as a 'pointless question' - particularly as in this course you'll be learning data science and using **3+** different programming languages[1].

If you want to ask me questions outside of our sessions please email me **charlotte.hadley@bcu.ac.uk**.

[1] You'll be learning R. But in order to use RMarkdown you will also need to use Markdown and YAML. You'll also likely end up using a little bit of HTML and CSS.

Course materials and website

I'd like to ask you **not** to read ahead in the lecture slides or workshops.

> This is because there are some exercises I'd like you to try in week `N` that are solved for you in week `N+1`.

As with all BCU modules you can find the lecture notes on Moodle.

However!

This course has a dedicated website (eng7218.netlify.app) that contains more materials than the Moodle page.

Course assessment

This module is 100% assessed with coursework that must be submitted **before 12:00 on Friday, 13 January 2023**.

Part of the coursework will require you to learn to use R and RMarkdown.

> I want to talk about the *goals* of this course before giving more details about how the assessment will work.

Understanding the goals of this course

So you know how to succeed

Course goals

I want you to succeed in this course.

I want you to succeed **after** this course in your career and/or research.

How to succeed in this course:

I want to break down each of these in turn:

- Feel confident in lectures
- Feel confident in workshops
- Feel confident in designing (and reading) data visualisations
- Feel confident in the assessment

Feel confident in lectures

For you to feel confident in the lectures I want you to:

- Ask questions if you feel lost or don't understand something.
- Understand **why** something is being taught in the context of the module goals.

Module goals

- Demonstrate a systematic understanding of the principles and approaches in data science to be used in healthcare.
- Critically appraise the key considerations for using healthcare data including ethics, information governance and security issues relevant to health data science.
- Apply knowledge of the R language to read and wrangle healthcare datasets into the R environment for analysis.
- Design data visualisations and tables with the R language to communicate properties of datasets and the conclusions of data analyses.

Module goals

- Demonstrate a systematic understanding of the principles and approaches in data science to be used in healthcare.
- Critically appraise the key considerations for using healthcare data including ethics, information governance and security issues relevant to health data science.
- Apply knowledge of the R language to read and wrangle healthcare datasets into the R environment for analysis.
- Design data visualisations and tables with the R language to communicate properties of datasets and the conclusions of data analyses.

Module goals

- Demonstrate a systematic understanding of the principles and approaches in data science to be used in healthcare.
- Critically appraise the key considerations for using healthcare data including ethics, information governance and security issues relevant to health data science.
- Apply knowledge of the R language to read and wrangle healthcare datasets into the R environment for analysis.
- Design data visualisations and tables with the R language to communicate properties of datasets and the conclusions of data analyses.

Module goals

- Demonstrate a systematic understanding of the principles and approaches in data science to be used in healthcare.
- Critically appraise the key considerations for using healthcare data including ethics, information governance and security issues relevant to health data science.
- Apply knowledge of the R language to read and wrangle healthcare datasets into the R environment for analysis.
- Design data visualisations and tables with the R language to communicate properties of datasets and the conclusions of data analyses.

Feel confident in workshops

The workshops will run a little differently to the lectures.

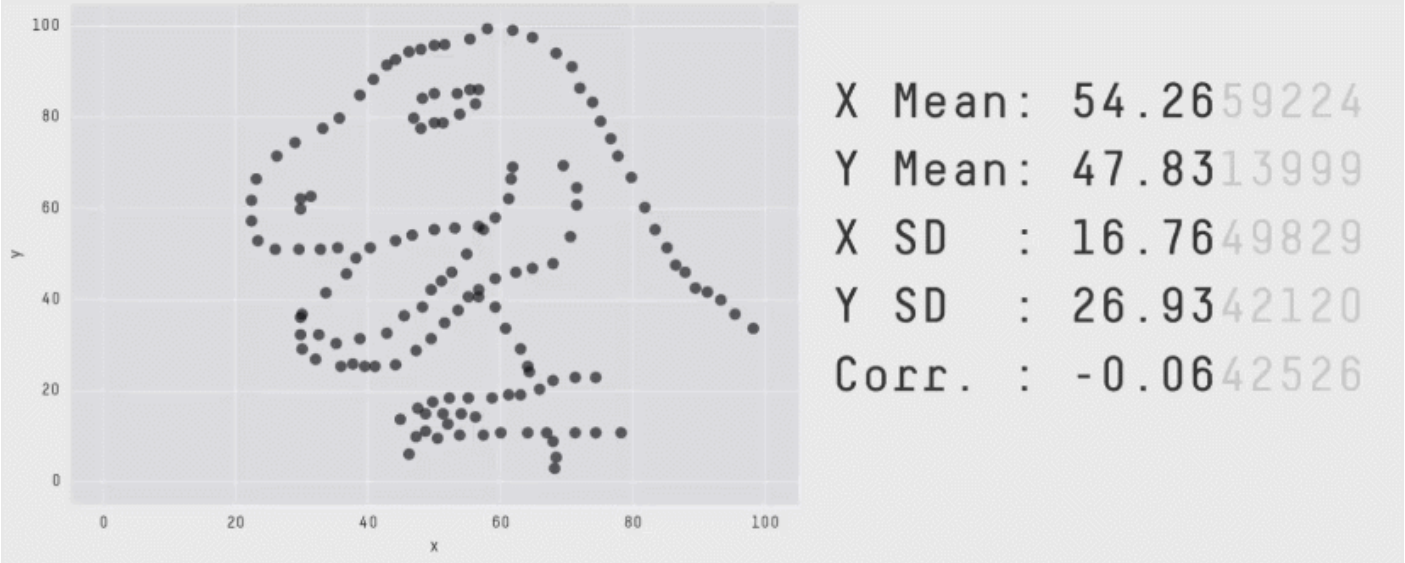
As we progress through the weeks we'll move away from you completing very specific tasks to more open eneded goals.

- Ask questions if you feel lost or don't understand something.
- Become confident in figuring out why your code doesn't work and problem solving it.
- Become confident in exploring new ideas, particularly different ways to explore and visualise data.

Feel confident in reading and designing data visualisations

Week 4 is meant to be when we focus on data visualisation.

However, we will start using data visualisations right from the beginning of the course.



Data visualisation produced by .

Feeling confident in the assessment

How the assessment is designed

There are two very different components to the assessment:

- the written component which requires you to explain concepts and critically analyse case studies.
  - You will use Microsoft Word (or your word processor of choice) for this part of the assessment.
- the coding component which requires you to write code to read, wrangle and visualise survey data.
  - You will use R and RMarkdown for this part of the assessment.

How the assessment is designed

There are 3 different sections in the assessment:

Section	Type of assessment
Part A) Open health data and anonymisation	Written component
Part B) Algorithms and health data	Written component
Part C) Analyse and visualize results from a health data survey	Coding component

The module website provides more details about these sections.

The colour coded timetable demonstrates which lectures and workshops will help you with each section.

Written component of the assessment

You will need to use *case studies* to demonstrate your understanding of concepts introduced in the lectures.

The course website's case studies section includes all the case studies we will cover in the course.

You are also strongly encouraged to research your own case studies (and please share them with the cohort!).

Coding component of the assessment

You will be making use of a tool called RMarkdown to answer the last part of the courswork

> Part C) Analyse and visualize results from a health data survey

RMarkdown allows you to write reports, presentations and even entire websites[1]. It's a very powerful tool that is widely used in industry and academia.

The thing that's powerful about it is that you can include (and run) R code in your documents.

[1] These slides and the entire module website is written with RMarkdown documents.

Coding component of the assessment

In the very first workshop I will thoroughly introduce you to R and RMarkdown.

In every subsequent workshop we will use RMarkdown.

You should have sufficient practice and expertise to answer the coding component of the coursework.

I will check in with you all about your confidence with the assessment in Week 8.

There is also a template GitHub repository that you can use for structuring your assessment.

How to succeed in this course:

Now I've covered these in more detail - do you have any questions?

- Feel confident in lectures
- Feel confident in workshops
- Feel confident in designing (and reading) data visualisations
- Feel confident in the assessment

Palate cleanser

Check out one of my favourite data visualisations ever:

bit.ly/3QISech

How to succeed after this course:

Now I've covered these in more detail - do you have any questions?

- Understand there are data science careers in both academia and industry
- Practice reproducible research from now onwards
- Appreciate and make use of open data standards where possible
- Protect people by protecting data
- Require Fairness, Accountability and Transparency for algorithms

Data Science Careers

What do you folks want to do in the future?

What do we mean by data science?

There's lots of discourse about the difference between "data science jobs" and "data analysis jobs" but most of this is gate keeping.

For our purposes:

> We successfully do data science when we write reproducible code that reads and analyses code in such a way that we can others stories about the data.
>
> Data science might involve statistics, but it does not necessarily require it.

> Reproducible code means that other people can run the code we write on their machines.

Academic careers using data science

Researchers across **all divisions and departments** use data science:

- Visualising close reading in poetry (and elsewhere)
- Deciphering lost languages
- Crowd sourced projects on Zooniverse,
  - Transcribing weather data from logbooks.
  - Classifying baby noises to explore language development

Of course, data science is being used prolifically in the collection and analysis of healthcare data.

Traditional academic research job route

1. PhD.
    - In the UK these are usually funded for 3 years, but funding *might* be extendable
2. Several "post doc" positions.
    - "Post doc" positions are difficult to define but are almost always fixed term contracts for 2-3 years.
3. Lecturer positions
    - Permanent jobs
    - These might be anywhere from 100% research to 100% teaching

> However there are too many PhD vacancies with too few research positions.
>
> doi.org/10.1038/d41586-019-03439-x

Non-traditional academic research route

There are many non-traditional routes into academia.

Research Software Engineering (RSE) is an excellent non-traditional route for folks with a data science background.

The RSE community is responsible for designing, building and maintaining the code/software that underpins academic research

> This is important because code/software is not traditionally celebrated or considered in the academic publishing industry

The Society of Research Software Engineering provides resources and career opportunities.

There are **many** folks in academic research positions that do not have PhD and/or post docs.

Data science careers in industry

I highly recommend the Build a Career in Data Science book by Emily Robinson and Jacqueline Nolis. Both these authors are part of the R community.

- A lot of the advice from the book is available in a blogpost from Emily Robinson.

I also recommend this great thread from Jesse Mostipak from RStudio (and previously Kaggle).

There's lots of other great advice out there.

Practice reproducible ~~researcH~~ code from now onwards

Assume all code you write might be useful for someone else

I've mentioned *reproducibile code* several times and described it as code that other people can use.

It's actually quite difficult to make code reproducible half-way through a project - **always start with best practices**.

... reproducible code makes for an awesome portfolio

In week 2's lecture I'm going to introduce GitHub and recommend you use it as a portfolio for future job applications.

> You do **not** need to use GitHub in the assessment for this module.

Appreciate and make use of open data standards where possible

Open Data is good for everyone

We'll talk a lot about Open Data in Week 2.

I want to encourage you to consider using open data standards **where possible** as it can benefit:

- You.

> 25.36% (± 1.07%) higher citation impact [for articles linking to a data repository]
>
> 10.1371/journal.pone.0230416

- Other researchers
- Society

Protect people by protecting data

We'll be looking at data anonymisation in lots of detail in Week 6's lecture.

Whenever we're working with data about people (or groups) we **must** keep in mind protecting their identities.

> Privacy itself is valuable.
>
> We need to protect individuals and groups from harm that could result from private data is published.
>
> There are legal requirements for data protection, including GDPR in the UK.

In Week 2 when I talk about Open Data I will also mention minimum requirements for privacy.

Require Fairness, Accountability and Transparency for algorithms

In Week 7 we will look at the ethics in algorithms which requires us to consider 3 different concepts:

- Fairness: Is the *training data* behind the data fair (does it look at what we think it looks at)?.
- Accountability: What are the impacts and secondary consequences of applying an algorithm?
- Transparency: Understanding of how algorithms are used in decision making.

> If you help develop an algorithm you're intrinsically connected with how that algorithm is applied.[1]

[1] For clarity, I'm not saying that you are **responsible** for how the algorithm is applied. By ensuring documented fairness in algorithm development this will aide in the future accountability and transparency of the algorithm.

How to succeed after this course:

Now I've covered these in more detail - do you have any questions?

- Understand there are data science careers in both academia and industry
- Practice reproducible research from now onwards
- Appreciate and make use of open data standards where possible
- Protect people by protecting data
- Require Fairness, Accountability and Transparency for algorithms

Why are we learning R?

Firstly - what's your starting point?

Let's answer some of the questions in the etherpad: pad.riseup.net/p/eng7218_week-1_lecture

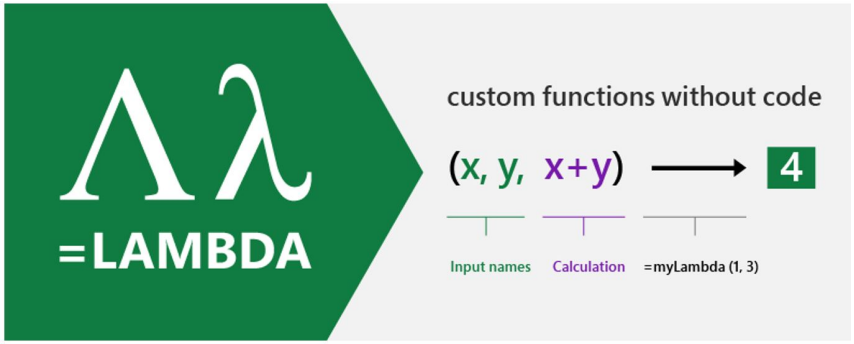
Programming vs GUI

You might expect me to bash all GUI based tools at the point, but there are incredibly powerful tools out there.

Programming vs GUI

You might expect me to bash all GUI based tools at the point, but there are incredibly powerful tools out there.

In fact, **Excel** now has a new way to program with Excel formulae

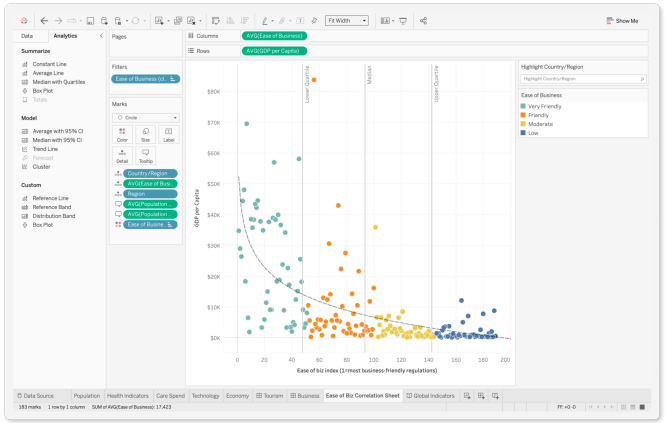


microsoft.com/en-us/research/blog/lambda-the-ultimatae-excel-worksheet-function/

Programming vs GUI

You might expect me to bash all GUI based tools at the point, but there are incredibly powerful tools out there.

Tableau has patented UX and a query language for building complex dashboards.



University of Oxford admissions dashboard

Programming vs GUI

Most tools that provide a 100% GUI are used for

- Data analysis without modelling
- Business intelligence (BI)
- Dashboard design and development

It is difficult to design fully UX-led tools for anything we might term "machine learning".

AWS and platforms like it market tools like SageMaker with glossy diagrams like this... **but there's always coding behind them**.

This includes using R & RMarkdown through the RStudio interface.

Benefits of programming over GUI

- If you're writing code **you can theoretically do anything**.
  - The ratio of code you can re-use and new code you need to write depends on the task.
  - In the R universe we're lucky to have a vast range of *packages*.

Benefits of programming over GUI

- If you're writing code you can theoretically do anything.
- Using code you can *programmatically* do things:
  - Import multiple data files
  - Run **many** different models with different parameters
  - Generate a report for every single patient

Benefits of programming over GUI

- If you're writing code **you can theoretically do anything.**
- Using code you can *programmatically* do things
- You can write tests to ensure the reproducibility of your code

Excel vs code: Reinhart & Ragoff

This is a screenshot of an Excel workbook by Reinhart and Ragoff (Reinhart and Rogoff [1]) which was used in 2010 to conclude

> countries with a debt-to-GDP ratio of 90% or higher see average growth of -0.1%

However, this was based on a faulty formula in cell M51

```
=AVERAGE(L30:L44)
```

Instead of

```
=AVERAGE(L30:L49)
```

	B	C	I	J	K	L	M
2			Real GDP growth				
3			Debt/GDP				
4	Country	Coverage	30 or less	30 to 60	60 to 90	90 or above	30 or less
26			3.7	3.0	3.5	1.7	5.5
27	Minimum		1.6	0.3	1.3	-1.8	0.8
28	Maximum		5.4	4.9	10.2	3.6	13.3
29							
30	US	1946-2009	n.a.	3.4	3.3	-2.0	n.a.
31	UK	1946-2009	n.a.	2.4	2.5	2.4	n.a.
32	Sweden	1946-2009	3.6	2.9	2.7	n.a.	6.3
33	Spain	1946-2009	1.5	3.4	4.2	n.a.	9.9
34	Portugal	1952-2009	4.8	2.5	0.3	n.a.	7.9
35	New Zealand	1948-2009	2.5	2.9	3.9	-7.9	2.6
36	Netherlands	1956-2009	4.1	2.7	1.1	n.a.	6.4
37	Norway	1947-2009	3.4	5.1	n.a.	n.a.	5.4
38	Japan	1946-2009	7.0	4.0	1.0	0.7	7.0
39	Italy	1951-2009	5.4	2.1	1.8	1.0	5.6
40	Ireland	1948-2009	4.4	4.5	4.0	2.4	2.9
41	Greece	1970-2009	4.0	0.3	2.7	2.9	13.3
42	Germany	1946-2009	3.9	0.9	n.a.	n.a.	3.2
43	France	1949-2009	4.9	2.7	3.0	n.a.	5.2
44	Finland	1946-2009	3.8	2.4	5.5	n.a.	7.0
45	Denmark	1950-2009	3.5	1.7	2.4	n.a.	5.6
46	Canada	1951-2009	1.9	3.6	4.1	n.a.	2.2
47	Belgium	1947-2009	n.a.	4.2	3.1	2.6	n.a.
48	Austria	1948-2009	5.2	3.3	-3.8	n.a.	5.7
49	Australia	1951-2009	3.2	4.9	4.0	n.a.	5.9
50							
51			4.1	2.8	2.8	=AVERAGE(L30:L44)	

King [2]

Excel vs code: Reinhart & Ragoff

By fixing this error the conclusion changed from:

> countries with a debt-to-GDP ratio of 90% or higher see average growth of -0.1%

To a completely different story:

> countries with a debt-to-GDP ratio of 90% or higher see average growth of 2.2%

	B	C	I	J	K	L	M
2			Real GDP growth				
3			Debt/GDP				
4	Country	Coverage	30 or less	30 to 60	60 to 90	90 or above	30 or less
26			3.7	3.0	3.5	1.7	5.5
27	Minimum		1.6	0.3	1.3	-1.8	0.8
28	Maximum		5.4	4.9	10.2	3.6	13.3
29							
30	US	1946-2009	n.a.	3.4	3.3	-2.0	n.a.
31	UK	1946-2009	n.a.	2.4	2.5	2.4	n.a.
32	Sweden	1946-2009	3.6	2.9	2.7	n.a.	6.3
33	Spain	1946-2009	1.5	3.4	4.2	n.a.	9.9
34	Portugal	1952-2009	4.8	2.5	0.3	n.a.	7.9
35	New Zealand	1948-2009	2.5	2.9	3.9	-7.9	2.6
36	Netherlands	1956-2009	4.1	2.7	1.1	n.a.	6.4
37	Norway	1947-2009	3.4	5.1	n.a.	n.a.	5.4
38	Japan	1946-2009	7.0	4.0	1.0	0.7	7.0
39	Italy	1951-2009	5.4	2.1	1.8	1.0	5.6
40	Ireland	1948-2009	4.4	4.5	4.0	2.4	2.9
41	Greece	1970-2009	4.0	0.3	2.7	2.9	13.3
42	Germany	1946-2009	3.9	0.9	n.a.	n.a.	3.2
43	France	1949-2009	4.9	2.7	3.0	n.a.	5.2
44	Finland	1946-2009	3.8	2.4	5.5	n.a.	7.0
45	Denmark	1950-2009	3.5	1.7	2.4	n.a.	5.6
46	Canada	1951-2009	1.9	3.6	4.1	n.a.	2.2
47	Belgium	1947-2009	n.a.	4.2	3.1	2.6	n.a.
48	Austria	1948-2009	5.2	3.3	-3.8	n.a.	5.7
49	Australia	1951-2009	3.2	4.9	4.0	n.a.	5.9
50							
51			4.1	2.8	2.8	=AVERAGE(L30:L44)	

Excel vs code: Reinhart & Ragoff

If we translated this into code we could guarantee that all rows were included.

> This is not to say code is free of errors!

But code gives us more ways to replicate, reproduce and test our analyses.

	B	C	I	J	K	L	M
2			Real GDP growth				
3			Debt/GDP				
4	Country	Coverage	30 or less	30 to 60	60 to 90	90 or above	30 or less
26			3.7	3.0	3.5	1.7	5.5
27	Minimum		1.6	0.3	1.3	-1.8	0.8
28	Maximum		5.4	4.9	10.2	3.6	13.3
29							
30	US	1946-2009	n.a.	3.4	3.3	-2.0	n.a.
31	UK	1946-2009	n.a.	2.4	2.5	2.4	n.a.
32	Sweden	1946-2009	3.6	2.9	2.7	n.a.	6.3
33	Spain	1946-2009	1.5	3.4	4.2	n.a.	9.9
34	Portugal	1952-2009	4.8	2.5	0.3	n.a.	7.9
35	New Zealand	1948-2009	2.5	2.9	3.9	-7.9	2.6
36	Netherlands	1956-2009	4.1	2.7	1.1	n.a.	6.4
37	Norway	1947-2009	3.4	5.1	n.a.	n.a.	5.4
38	Japan	1946-2009	7.0	4.0	1.0	0.7	7.0
39	Italy	1951-2009	5.4	2.1	1.8	1.0	5.6
40	Ireland	1948-2009	4.4	4.5	4.0	2.4	2.9
41	Greece	1970-2009	4.0	0.3	2.7	2.9	13.3
42	Germany	1946-2009	3.9	0.9	n.a.	n.a.	3.2
43	France	1949-2009	4.9	2.7	3.0	n.a.	5.2
44	Finland	1946-2009	3.8	2.4	5.5	n.a.	7.0
45	Denmark	1950-2009	3.5	1.7	2.4	n.a.	5.6
46	Canada	1951-2009	1.9	3.6	4.1	n.a.	2.2
47	Belgium	1947-2009	n.a.	4.2	3.1	2.6	n.a.
48	Austria	1948-2009	5.2	3.3	-3.8	n.a.	5.7
49	Australia	1951-2009	3.2	4.9	4.0	n.a.	5.9
50							
51			4.1	2.8	2.8	=AVERAGE(L30:L44)	

Why are we learning R?

Why are there so many languages?

There are easily 100s of "programming languages" in use today.

To make any kind of sense of them we need to start splitting them into categories.

Markup Languages

`HTML` is the most commonly used markup language - it's used to describe a web page.

You then use the `CSS` language to style the HTML page.

We'll use another markup language - `Markdown`.

> Add these to your CV as markup languages! I'll talk more about CVs at the

Programming Languages

> We need to go deeper

Why are there so many languages?

Machine Language

The lowest level languages. There is no abstraction.

Assembly language

Architecture-specific language with minimal abstraction.

> This is completely outside of scope for us.

Programming Languages

> We need to go higher

Why are there so many languages?

Compiled Languages

C, C++, C# are among the most widely used compiled languages.

In modern times we consider these "low-level" languages, but people have opinions about this.

Code is written and then compiled directly to machine code - it therefore runs quickly.

> Sometimes R code is re-written in C++ with {Rcpp} for efficiency.

Interpreted Languages

> There's an explosion of choice in this space!
>
> Some interpreted languages are software specific, eg VBA.

Why are there so many languages?

Scripting Languages

Most people today use scripting language to mean

- JavaScript
- Python
- R

But there are **many** other examples and **zero** strict definition of what is meant by a scripting language.

> We can choose to think of these as languages that are good for doing data science.

Every Other Language

> Folks are sometimes dismissive of scripting languages. Let them. We'll do fun and important stuff with them.

R vs Python

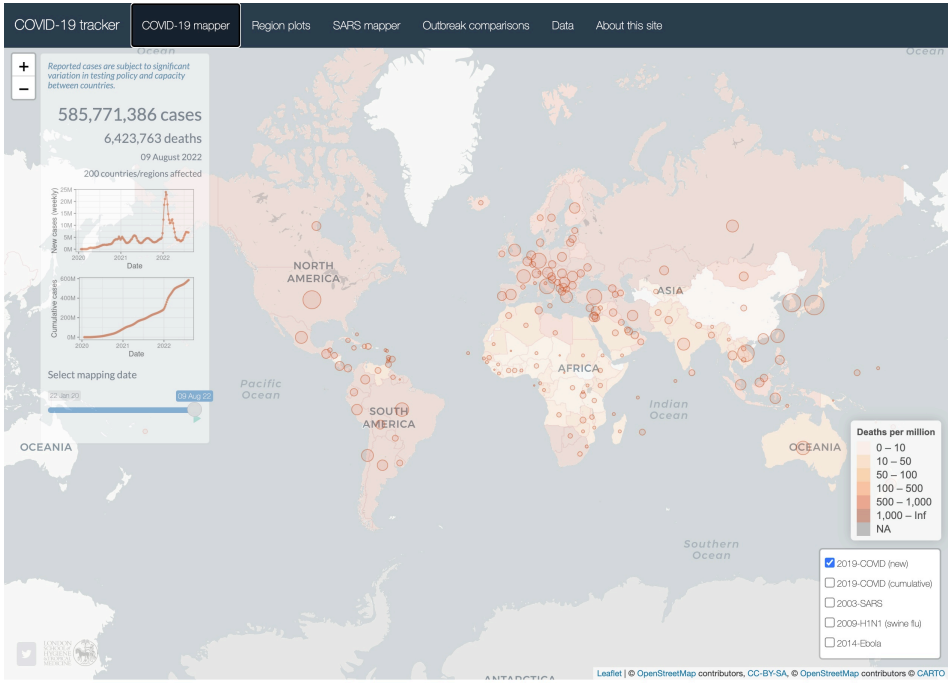
You'll sometimes hear

> R is a language for statistical analysis and data visualisation, it's not a general purpose language.

- R is a fully featured scripting language used by millions of people every day.
- R has multiple solutions for making it available on a server.
- R is widely used in production
- R is deeply integrated into AWS, Azure, Google Cloud and other Cloud-based infrastructure.
- R is deeply integrated into BI tools like Tablea and PowerBI

R vs Python (Healthcare)

- R has a very strong presence in the bioinformatics community thanks to bioconductor.org
- There is a very active R community in the NHS - nhsrcommunity.com/
- R users have used {shiny} to build several widely used tools for studying COVID-19 data



R vs Python (ease of learning)

In general, there are fewer "computer science" things you'll need to learn to pick up R and run.

This is particularly true thanks to the `tidyverse` ecosystem of packages.

The R community also has a very healthy collection of free to read (open source) books, eg

- R for Data Science
- R for Health Data Science
- Text Mining with R
- Tidy Modelling with R

R vs Python (data visualisation)

The `{ggplot2}` package is an incredibly easy to learn tool for building rich and beautiful static charts. It also has 100+ extensions.

In the Python world there are several competing and non-compatible visualisation packages.

Through htmlwidgets.org it's also possible to create interactive charts, maps and tables without learning JavaScript.

R vs Python (community)

The R community is very friendly and welcoming to new users

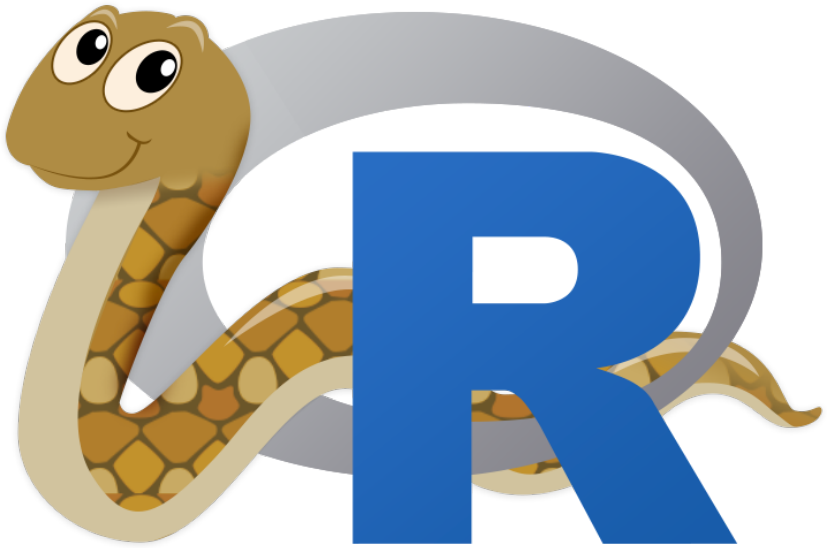
- R Forwards is dedicated to elevating voices often missing in other programming communities.
- Twitter is often the best place to join in the community via the `#rstats` hashtag.
- Tidy Tuesday is a social data project in R that's an awesome excuse to experiment and get involved

It's actually R + Python

It's unfair to put these two tools in competition.

There are lots of folks using both R and Python interchangeably and positively.

- The {reticulate} package provides tools for interopability between Python and R.



References Page 1

[1] C. M. Reinhart et al. "Growth in a Time of Debt". In: *American Economic Review* 100.2 (May. 2010), pp. 573-578. ISSN: 0002-8282. DOI: 10.1257/aer.100.2.573.

[2] R. King. *How to Avoid Making an Excel Mistake like Rogoff and Reinhart*. https://qz.com/75119/how-to-avoid-making-an-excel-mistake-like-rogoff-and-reinhart/. Apr. 2013.