DS Class Notes

Notes for CSC482/DSC205 Introduction to Advanced Data Science Spring 2022

Table of Contents

README

Instead of bugging you with emails, I opt to summarize my course observations regarding content, process, in this file. These often contain additional links, articles, and musings.

I usually update it after each class - it also contains the homework (if any). The first point of call for any questions should be the FAQ. There are two FAQs - a general one (for all my courses), and a FAQ for CSC 100.

You find the whiteboard photos here in GDrive.

The companion file to this file, with the agenda and much of the course content, is the agenda.org file.

Course introduction - w1s1 (01/12/22)

See also: Google Meet chat

Homework (until Tuesday, 18 Jan, 11:59 PM)

IF YOU DID NOT COMPLETE DSC101	IF YOU COMPLETED DSC101
Complete "Datascience for everyone" on DataCamp	
Complete "Introduction to R" on DataCamp	
Pass the Entry Quiz (Schoology) > 50%	Complete the Entry quiz (Schoology)

Stuff

Data science pipeline

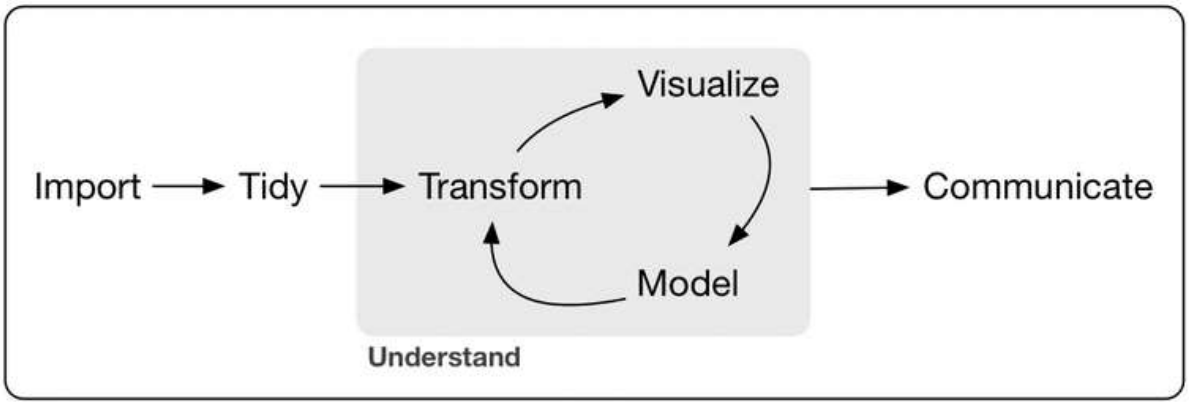


Figure 1: Data science pipeline (Source: Wickham/Grolemund 2017)

Books

None of which will be an exclusive, but I may use stuff from these books. They're all good in their own way but a little hard on one's own.

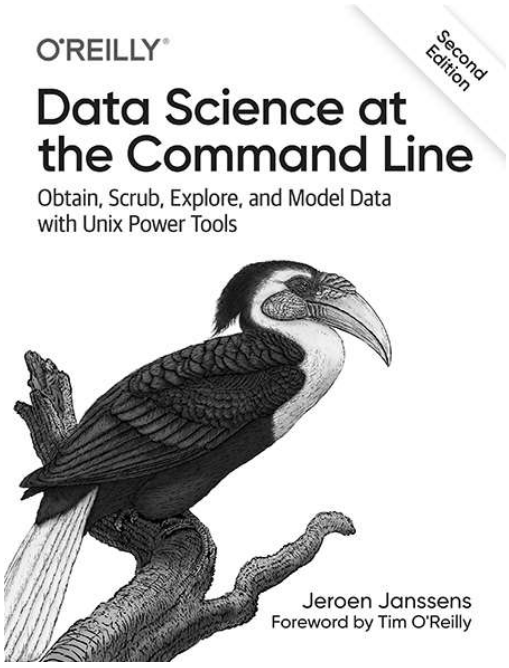


Figure 2: Data Science at the Command Line by Jeroen Janssens

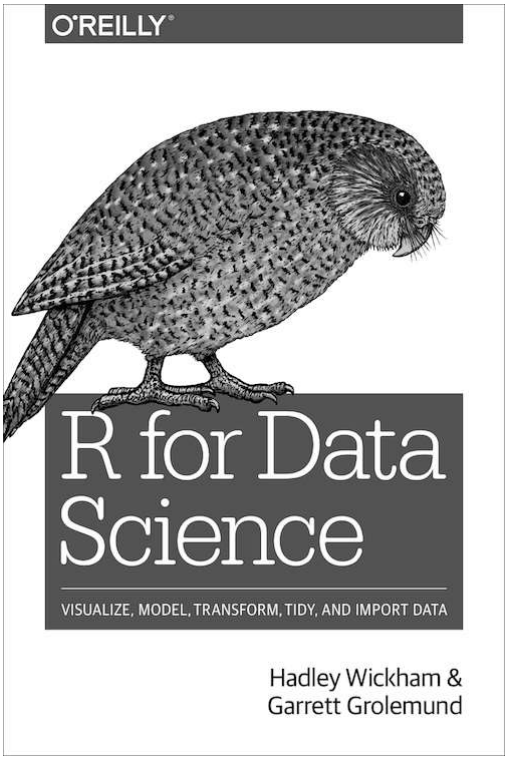


Figure 3: R for Data Science by Wickham/Grolemund

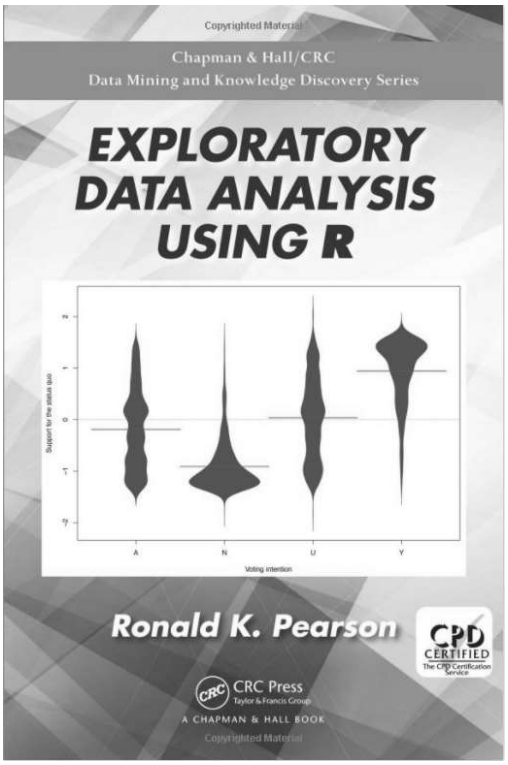


Figure 4: Exploratory Data Science Using R by Ron Pearson

Regular expressions

Important for efficient text mining and string manipulation, e.g. when doing data science on the command line, regexp are search patterns. Here is a complete, free, online tutorial (RegexOne, 2021), and here is a free book chapter explaining regexp as part of automating stuff with Python (Sweigart, 2019).

Examples for such regular expressions are the `*` in an SQL command like `SELECT * FROM t` to query all columns of the table `t`, or `^x` that matches any string starting with `x` etc.

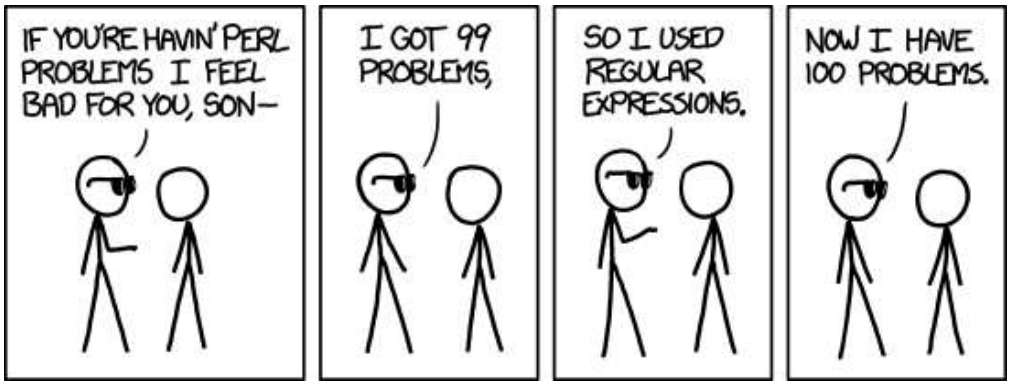
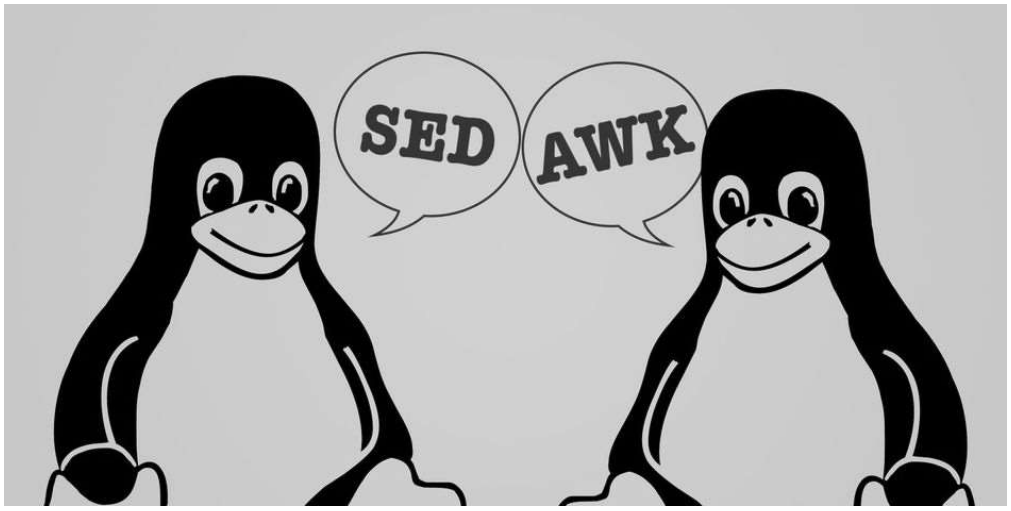


Figure 5: Perl problems (Source: xkcd).

`Perl` from the cartoon title, is another powerful language whose strength is pattern matching and manipulation. It's more high level than `awk` or `sed` though and lives on all operating systems.

`awk` (and `sed`)

This is an "awkward" language on GNU/Linux. They're natural languages for regexp use. Makes data wrangling on the command line reall easy. Not hard to learn, and we might take a look at it - I plan to present it in another class (operating systems) as part of the Linux layout. Here's a tutorial for `awk`, and an opinion piece (Hughes, 2015). Something else for a rainy afternoon.



Getting started with GNU Emacs

GNU Emacs is going to be our IDE and our environment for literate programming. This is an experiment that I'm running this term in all my courses - but this course (R) and the intro class on C/C++ are the two classes where Emacs should really pay off.

I suggested two short videos to get started while munching a bagel:

- First Steps With Emacs (Eddelbuettel, 2021). This is especially for RStats people (like you), with a focus on ESS ('Emacs Speaks Statistics').
- Literate programming demo (Birkenkrahe, 2022). Here I contrast Emacs Org-mode with an interactive shell using SQLite, an RDBMS.

We'll get deeply into this soon as we set up our infrastructure.

Notebooks and notebook platforms

There are many interactive notebooks and notebook platforms - they're especially popular in data science (and perhaps data science is so popular, and easier to learn because of them).

Some examples: Jupyter (originally only for Python), Google Colaboratory (for Python and R - though with "magic" commands, one can use other languages, but it's not straightforward), and Kaggle (owned by Google). Kaggle serves notebooks, datasets and (most importantly) data science competitions (strong focus on machine learning). These are often quite ideological ("Save the whales with data science") but what isn't these days? Which is why data science needs strong bias monitoring[1].

Installing R / Windows PATH - w1s2 (01/14/22)

R

TO DO	WINDOWS
Download base R from CRAN	R 4.1.2 "base"
Run installer	
Check files	C:/Program Files/R
Go to the binary folder	c:/Program Files/R/R-4.1.2/bin/x64
Open R GUI	Rgui.exe
Open R terminal	Rterm.exe
Check Rscript	Rscript test.R
Check PATH	

Log

- Short rant about Python vs R and why you learn R (vonjd)
- Showed R console and Rscript in DataCamp
- Showed R in a Windows (CMD) terminal
- Showed R inside Emacs in a terminal (no syntax highlighting)

- At CRAN, we want "base R" (without packages)
- The current version of R (Jan'22) is 4.1.2 "Bird Hippie"
- Normally, before running executables: check the "checksum" (Hoffman,2019)
- Run the installer, accept standard suggestions
- Start the launcher from the desktop
- GUI appears (`Rgui.exe`)
- Saving the workspace image stores `.RData`, `.Rhistory`, and `.Rplots` files containing (binary) data, command history, and PDF plots, respectively
- Update the `PATH` variable (search for PATH) using the string from the file explorer that contains the path to `bin/`
- Apparently, you don't have to do this in Windows 11 (but don't rely on it - better find out how to drive with stick shift!)
- Open a Windows terminal ("CMD")
- Start R (enter R)
- Test R with some commands like in the 1 code block.

```
plot(rnorm(100))
3 + 4
x <- rnorm(100)
str(x)
plot(x)
q()     # you can save your workspace image (don't)
```

- If you have any installation issues: check the R FAQ first

Installing and setting up GNU Emacs - w2s3 (01/19/22)

Emacs+ESS

TO DO	WINDOWS
Download Emacs+ESS	Download Installer
Run installer	Standard config
	Desktop shortcut
Check README	*Opens after installation*
Check Emacs	`emacs -nw` in terminal / desktop shortcut
Set PATH	*requires admin privileges*

Log

- If you don't have the modified GNU Emacs (with ESS already installed), you need to install and load the `ess` package
- See install.org (+ PDF) in the org/emacs GitHub repo for installation instructions if you want to put this on your own PC
- GNU Emacs layout: buffer window + modeline + minibuffer
- Commands begin with C-x (CTRL+x) or M-x (ALT+x)
- C-g interrupts any process
- List of open buffers: C-x C-b
- Change to other buffer: C-x o

- Close all visible buffers except one: C-x 1
- Start R (if installed and PATH set correctly): M-x R
- This opens an R session in the current directory (iESS mode)

Understanding Emacs Org-mode - w2s4 (01/21/22)

This class will get the most intense exposure and training for GNU Emacs, because of the need to work with interactive notebooks in data science. Getting to play around in Emacs in other courses (Databases, Operating Systems) will only improve your editor skills.

What we did using the instructions from tutor.org:

- Downloaded GitHub directory with `.org` files
- Opened `.org` files permanently with GNU Emacs
- We covered:
  - header options in Org-mode
  - moving around in Emacs buffers
  - opening/closing/suspending Emacs (also from the cmd line)
  - reading a file into Emacs, and saving it
  - opening buffer list and directory
  - switching buffers
  - creating a region, killing and yanking it
  - changing the font
  - opening the onboard tutorial
  - aborting commands
- We'll rehearse these in our weekly quiz on Monday!

- To get better, work through the tutorial (C-h t)

  See also the article "Getting started with Emacs" (Kenlon, 2020), and the video "The Absolute Beginner's Guide to Emacs" (System Crafters, 2020) with my notes.

Customizing Emacs (init file) - w3s5 (01/24/22)

Planned:

Practice	GNU Emacs Tutorial cont'd (tutor.org)
- Package manager	`M-x package-list-packages RET`
- Start R shell in Emacs	`M-x R` (R must be installed & in the `PATH`)
- Add init file	.emacs sample file (GitHub)
- Create `first.org` file	`C-x C-f ob.org RET`
- Create R code block	`#+begin_src R :session :results output ...#+end_src`
- Run R code block	`C-c C-c`

Captain's Log

See tutor.org for details:

- We added .emacs file in the ~/ HOME directory and discussed its content and structure (Emacs-Lisp) - especially the Org-babel packages.
- We talked about the Org-mode file assignment.
- After restarting Emacs (to load the configuration file), we opened the package manager with `M-x package-list-packages`. If the `.emacs` file is in the right location, the package manager should refresh its content.
- The package manager lists many downloadable packages. You downloaded the `org-beautify-theme` and `org-bullet` - both packages to improve the appearance of Emacs.
- Here is the Emacs documentation on the initialization file `.emacs` in the GNU Emacs manual: "How Emacs finds your init file".
- By default, Emacs will open to `default-directory`. This is a variable that you can set in your `.emacs` file. Here is an example where the working directory is set to `C:\Users\birkenkrahe\Emacs`

```
(setq default-directory "c:/Users/birkenkrahe/Emacs")
```

Notice how Windows requires backslashes, while Emacs (and Unix/Linux) use forward slashes.

Running code in Org-mode 1 - w3s6 (01/26/22)

- When you look at an Org file as a PDF or on GitHub, you will not see the meta data starting with `#+`. Org-mode files are meant to be edited/viewed in Emacs.
- The code block header has the following arguments:

HEADER ARGUMENT	MEANING
`:session *R*`	Run R in a session in the Emacs buffer `*R*`
`:results output`	insert output directly in the org file
`:tangle first.R`	export source code as R file `first.R` ("tangle")
`:exports both`	both result and source code will be exported
`:comments both`	link source code and org files, add comments to source

Running code in Org-mode 2 - w3s7 (01/28/22)

- Feel free to bring your own laptop to future sessions. If you want me to check installation because something did not work, come a little earlier or stay a little later.
- This concludes our "Emacs week". To get more practice in GNU Emacs, complete the onboard tutorial (`C-h t`), and of course there's still one (simple, text-only) Org-mode assignment outstanding.
- Solutions to the Org-mode assignment are posted here on GitHub. Note that submissions of programs as Org-mode files should always also be accompanied by references and sources.
- I told you an inaccuracy in class: when rendering the Org-mode file on GitHub, the `#+TITLE` meta information is displayed as the title of the file. If no such header is present, only the `README` file is displayed (with the file name as title).

Org-mode lab session - w4s8 (01/31/22)

- Setting the default directory (the folder where Emacs "wakes up" when you open Dired with `C-x d`):

```
;; set default working directory to c:/Documents/GitHub/
(setq default-directory "c:\\Users\\birkenkrahe\\Documents\\GitHub\\")
```

2022 Data Trends - w4s9 (02/02/22)

Notes from the DataCamp webinar (DataCamp, 2022)

Overview

- Great acceleration (2020) - reaction
- Great transition (2021) - recognition

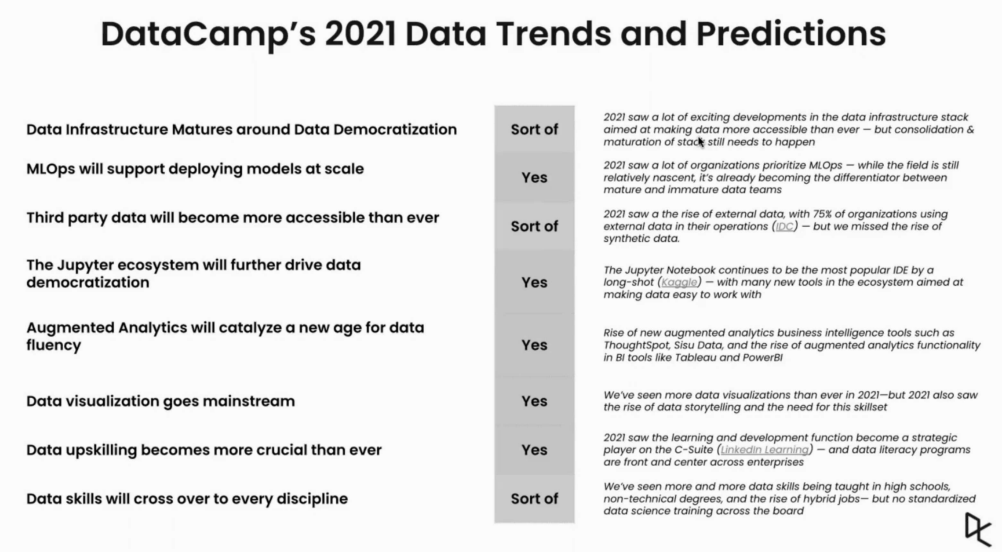


Figure 7: Prediction check 2021-2022

"Jupyter ecosystem"?

"Augmented analytics"?

- Operationalization of large language models: dashboards
- "Innovations in the data tooling stack" (more and better tools)

Trends

1. 100-fold increase prediction in data generated (2021-2025) according to Accenture
2. Data mesh vs data lake - Data PaaS - the issue is speed (infrastructure is complex and slow-moving)
3. MLOps mature - report: mostly startups (= economically irrelevant)
4. Data tool stack grows
5. Learning & Development - "upskilling becomes a mandate". Cotton: "People on this call are weird. Most people do not voluntarily join a webinar on data trends." Hairdresser asked him about his job…"does this mean that you work with computers and stuff." The knowledge divide is huge.
6. Data governance and quality

- data **catalog**
- data **observability** (freshness)

Documentation aids analysis (compare with Andrew Ng's initiative for more data tools and transparency). Independence of technical skill (no-coders).

7. NLP - e.g. PowerBI allows NLP descriptions of something you want to calculate, and it will auto-generate code/graphs for you. OpenAI: Codex allows for Python-from-description coding.

   Reverse: replit.com - don't have to read code anymore because the platform explains it to you.

8. Culture focus shift intensifies
9. Talent pool and talent generation will expand

Discussion / Groupwork results

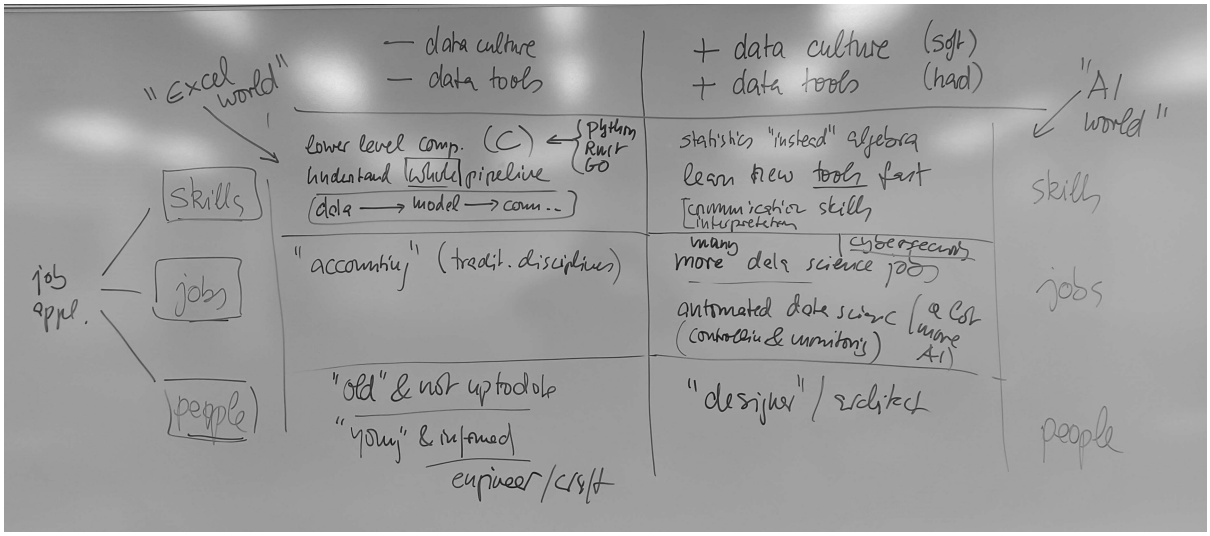


Figure 8: data trend scenarios

Studying with DataCamp - w5s10 (02/07/22)

- Simplify your Org-mode setup with `PROPERTY` settings
  - Add this at the top of your Org-mode file with code:

```
#+PROPERTY: header-args:R :session
#+PROPERTY: header-args:R :results output
```

Restart the file or refresh with `C-c C-c` on the `PROPERTY` line, and now a code chunk like this should work fine:

```
str(mtcars)
```

Try it now! (Documentation)

- Type the DataCamp exercises out as Org-mode files to get practice - both in Emacs Org-mode, but also in R (example)

The Anti-IF Campaign

You may have been mystified by my mentioning Cirillo and the Pomodoro time management technique but also IF-THEN-ELSE as a No-No in software engineering. This last issue was related to the "Anti-IF Campaign" launched (not tongue-in-cheek) by my friend Francesco Cirillo of Berlin (Cirillo, 2022):

#+begin_quote "The Campaign is against the use of the IF statement as a regular design strategy to deal with growth, change and complexity ("Let's Put an IF Syndrome") in an evolutionary context. Despite being an "easy," and apparently effective, way of delivering the value requested by the customer, this "design strategy" has negative repercussions when applied regularly as the main strategy to deal with change, growth and complexity. By applying the "IF Strategy" in an evolutionary context, software systems becomes more complex to be read, tested, even debugged. It becomes easier to duplicate code, accumulate technical debt and spend more time fixing bugs. In the result, the software system become more complex. New features and changes will cost more and more." #+end_quote.

Installing packages, using index vectors - w5s11 (02/09/22)

- Under Windows, when trying to install a package with `install.packages()`, you're prompted for a CRAN mirror to use.
- Put the following lines into a file `.Rprofile` in your `$HOME` directory to avoid having to answer this question every time:

```
options(repos = c("https://cloud.r-project.org/"))
```

The installation of packages from within Emacs should also work now. The screenshot shows an example (Emacs 27.2 under Windows 10).

```
#+begin_src R :session
  install.packages("dslabs")
  library(dslabs)
  data(murders)
#+end_src

#+RESULTS:
#+begin_example
Installing package into 'C:/Users/birkenkrahe/R/win-library/4.1'
(as 'lib' is unspecified)
trying URL 'https://cran.microsoft.com/bin/windows/contrib/4.1/dslabs_0.7.4.zip'
Content type 'application/zip' length 4699606 bytes (4.5 MB)
downloaded 4.5 MB

package 'dslabs' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\birkenkrahe\AppData\Local\Temp\RtmpInnqOD\downloaded_packages
#+end_example
```

Figure 9: R package "dslabs" installed in Org-mode file

Reviewing test 1, xkcd, plots - w6s14 (02/16/22)

- "p-hacking" refers to the practice of reanalyzing data in order to get a predetermined, desired result that confirms ones bias. More specifically, it is a misuse of the p-value, which is a measure for statistical relevance (however not for practical relevance). A small p-value corresponds to a greater chance that the observed results are not just due to random effects. For a critical discussion, see e.g. Statistical Bullshit (2017).
- I dropped those DataCamp assignments that dealt with the "Tidyverse", a bundle of packages including the popular `dplyr` package. In my view, the TidyVerse is not suited for beginners (Matloff, 2020). However, if you have time, it would probably be useful to take a look at the DataCamp course (it's not difficult).
- `ggplot2` is a very popular graphics package, especially for creating pretty graphs. It is distributed with the "Tidyverse", which it predates by several years however. We will take a look at `ggplot2` in the course, in preparation for the EDA in R DataCamp course.

Guest talk - Stone Ward - w6s15 (02/18/22)

- Ask Matthew Stewart for examples of the code he collects - a high end and a low end example. Also, ask about RStudio vs. Emacs vs. R console as his working environment[2].

#_attr_html: :width 600px

Marcus Birkenkrahe <birkenkrahe@lyon.edu> 1:51 PM (0 minutes ago)
to Matthew

Hi Matthew -

> Great questions.

Yes, I was pleased that the talk caught on so well. This is a great cohort of students. I asked them why they didn't ask more in class and it turns out that they think that's more respectful... :-)

> 1) Some simple examples of templates in my library are time series forecasts/multi-linear regression models. I mostly keep a compendium of tests/transformations for unique situations as the code itself is fairly basic (think testing for heteroskedasticity). The more complex templates range in topic from a Google Analytics API that is designed to get around sampling to propensity models, market basket analysis, and complex ggplots or gganimation. Essentially, any type of code that took significant time to understand/write initially I keep so that I can easily replicate the framework in the future.

Thank you. A couple of students are exploring gganimate now - I think I might leave that for the fall course on data visualization. Also, I am not a friend of Tidyverse or RStudio for beginners, and I prefer base R when possible (and data.table instead of dplyr because of syntax and speed). Instead, I've begun to use Emacs + ESS in class, and found it totally doable at any level. I'm impressed with the students. Your library advice was very well received!

> 2) I use RStudio and run rstudioapi::addTheme("https://raw.githubusercontent.com/batpigandme/night-owlish/master/rstheme/night-owlish.rstheme", apply = TRUE) to set dark mode as I find the program much easier to read with this theme.

I use Emacs + ESS (I'm hacker hippie and free software freak at the core), and Org-mode literate notebooks, which favor productivity while still allowing for great communication and sharing. I also use the dark theme - night-owl in Emacs!

> 3) Finally, one correction I did want to share regarding Excel is that in business it is essential to be able to use it since you can quickly build easy tools for users to employ versus testing out an entire Rshiny development cycle or do some quick analysis that remains in the spreadsheet. I find companies like to have the ability to open a spreadsheet with data highlighted or graphs already included occasionally. However, as data scientists, we should aim to move that data into SQL databases or clean/analyze with R/Python/etc. whenever possible. Hope that makes more sense!

That's why I use Emacs - multilingual spreadsheets with documentation, code and output included. Also for presentations. But I digress...because of your remark, I'm now looking at Excel too. Though I favor linking Excel + R + SQL...something to figure out first. Emacs Org-mode tables have spreadsheet capabilities, and I never need Excel so I'm quite rusty myself...

- I discovered this free toolkit to connect Excel and R, BERT. Confusingly, it is named like Google's AI algorithm, the "Bidirectional Encoder Representations from Transformers" with whom it has nothing to do.

- Matthew commented further re: Excel prowess required at Stone Ward:

  > "For excel, if students can make a pivot table, use sumif, xlookup, and make a chart quickly then they're in a good spot with the program."

- I had another exchange on Excel with Lyon Prof of Exercise Science, Matthew Peterson. He asked:

  > "During the data science talk yesterday, you mentioned that Excel was error prone. As someone who uses Excel quite a bit, this caught my attention. Would you mind expanding on that point for me?"

  I answered his question in some detail:

  > Good morning, Matthew - Sure, no problem. Here is a series of more or less famous cases: "The 7 Biggest Excel Mistakes of All Time" (teampay, 2022).
  >
  > Here is a more high profile case that I often discuss with my students. Though at Lyon, unless they are business students, they won't be able to appreciate it as much perhaps: "The Reinhart-Rogoff error - or how not to Excel at economics" (Borwein/Bailey, 2013).
  >
  > My main point here re: Excel - 1) find alternatives (esp. for visualization, R is great and really much easier to learn than Excel) 2) check all results using original data (or probes) if you can. (2) can lead to sudden fame - one of them could be the next Thomas Herndon!
  >
  > At the conceptual level, Excel does what most other dashboard apps do, too - hide the inner workings from the user. Obviously, we need some level of abstraction to use computers at all, but for scientific purposes, this is not appropriate (and also not for research-based teaching and learning). Which is why, in my classes, it "foundations above all", mixed with the latest research. The students take to the foundations easily enough once they have taken leave of the drag-and-drop paradigm and acquainted themselves with the command line, but they have difficulty with research because, alas, they have not learnt how to read. I try to address this in my classes (see my FAQ here - "How should you read?").
  >
  > In fact, this is likely going to be one of my points for my April lecture! You may recognize the thought there…sorry to be boring then already :-)
  >
  > Thanks for attending and for asking! Cheers, Marcus

- FAQ: Animations - including GIFs - cannot be rendered inside Emacs, I believe, (you can see open devices in R with `dev.list()`). Those are devices outside of Emacs run by the windows manager. Also, I might do gganimate now or I might leave it for the DataViz course next term. Simply because it isn't all that important. Most data are difficult enough to understand without animation, and the added value is often not there. It's a superficial pleasure to see things moving (the eye likes it - but visualization is only superficially for the eye). There are exceptions but one needs to check in each case if running images add value.

Guest talk post mortem: Google Analytics, Excel, Tidyverse - w7s16 (21-Feb)

The Google Data Analytics certificate (and other courses)

- This is a Coursera MOOC consisting of 8 courses
- The course promises to have you "job-ready in less than 6 months" for under $300

- > 600k students have been enrolled[3]
- It features SQL querying and R programming (not Python or Excel)

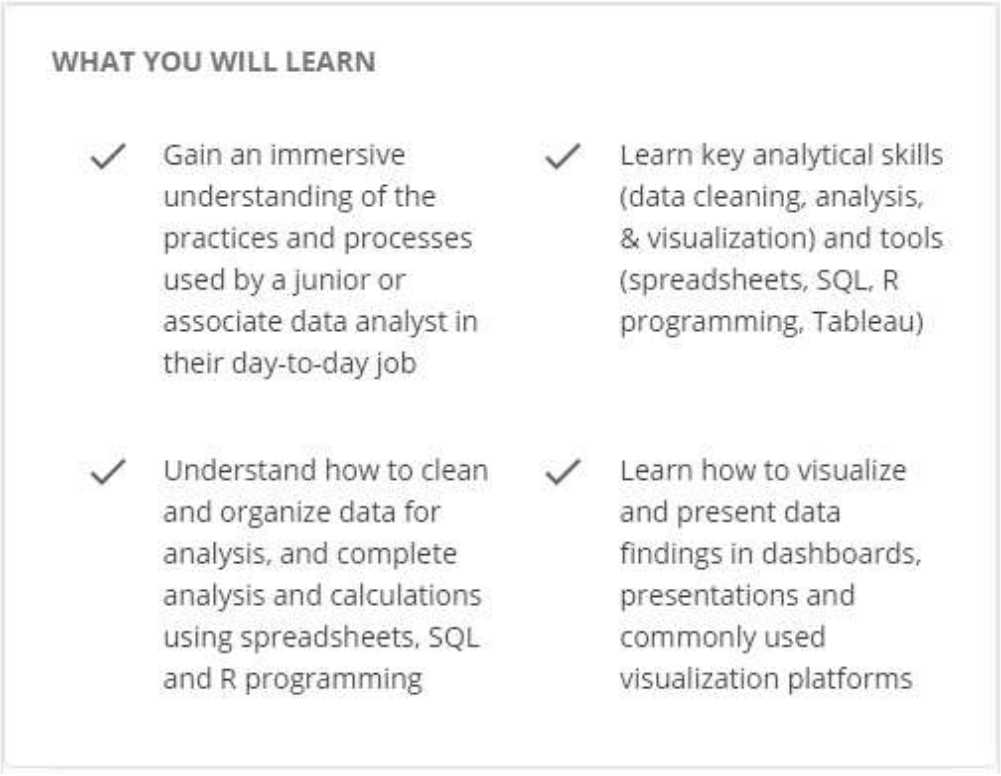


Figure 10: Google Data Analytics Certificate @coursera.org

- Elements of R - language instruction, visualization, and documentation (with RStudio and RMarkdown) are mixed
- The R course has had > 100k students enrolled
- What's the difference between "data science" and "data analytics"?[4]

Excel

Question:

> "During the data science talk yesterday, you mentioned that Excel was error prone. As someone who uses Excel quite a bit, this caught my attention. Would you mind expanding on that point for me?" (Prof Matthew Peterson)

Answer:

> "Here is a series of more or less famous cases: https://www.teampay.co/insights/biggest-excel-mistakes-of-all-time/
>
> Here is a more high profile case that I often discuss with my students. Though at Lyon, unless they are business students, they won't be able to appreciate it as much perhaps.
>
> My main point here re: Excel - 1) find alternatives (esp. for visualization, R is great and really much easier to learn than Excel) 2) check all results using original data (or probes) if you can. (2) can lead

to sudden fame - one of them could be the next Thomas Herndon!

At the conceptual level, Excel does what most other dashboard apps do, too - hide the inner workings from the user. Obviously, we need some level of abstraction to use computers at all, but for scientific purposes, this is not appropriate (and also not for research-based teaching and learning). Which is why, in my classes, it "foundations above all", mixed with the latest research. The students take to the foundations easily enough once they have taken leave of the drag-and-drop paradigm and acquainted themselves with the command line, but they have difficulty with research because, alas, they have not learnt how to read. I try to address this in my classes (see my FAQ here - "How should you read?")."

"Tidyverse" vs Base R

- Read: TidyverseSceptic by Norman Matloff
- Work through lessons on DataCamp mobile & practice 5-20 min per day

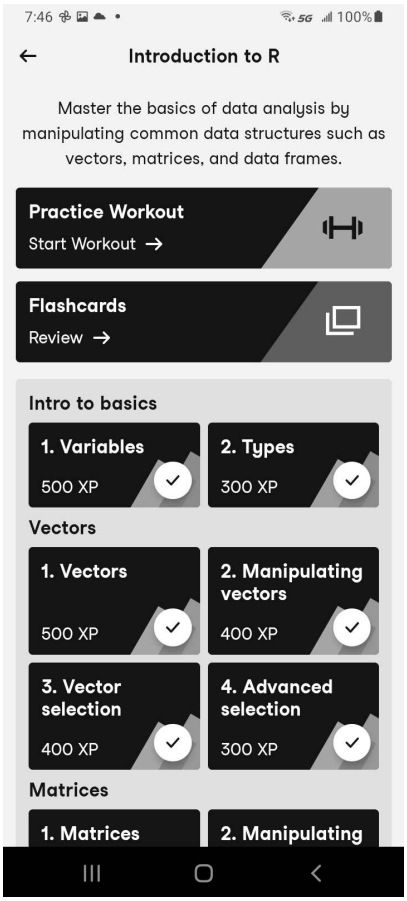


Figure 11: Introduction to R on DataCamp mobile

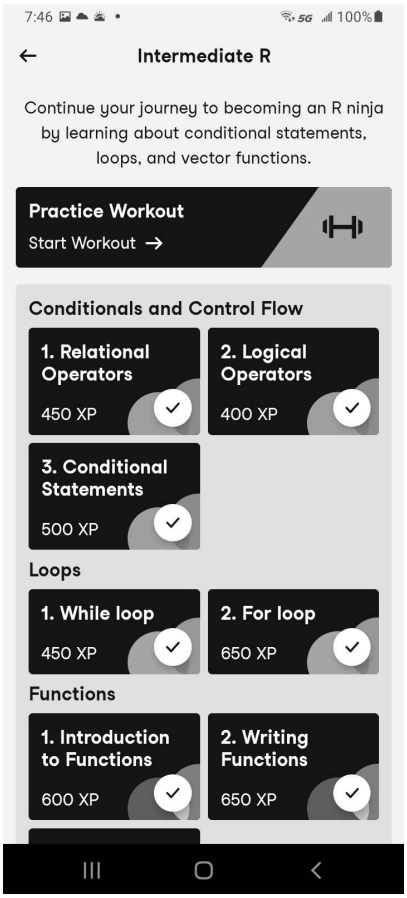


Figure 12: Intermediate R on DataCamp mobile

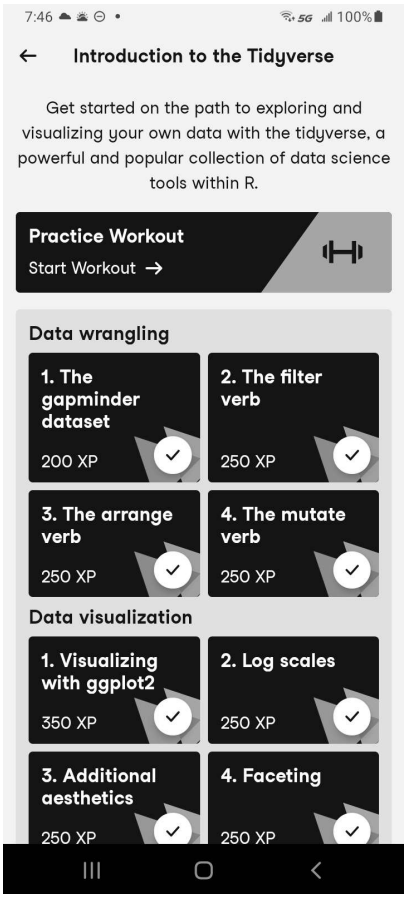


Figure 13: Introduction to the Tidyverse on DataCamp mobile

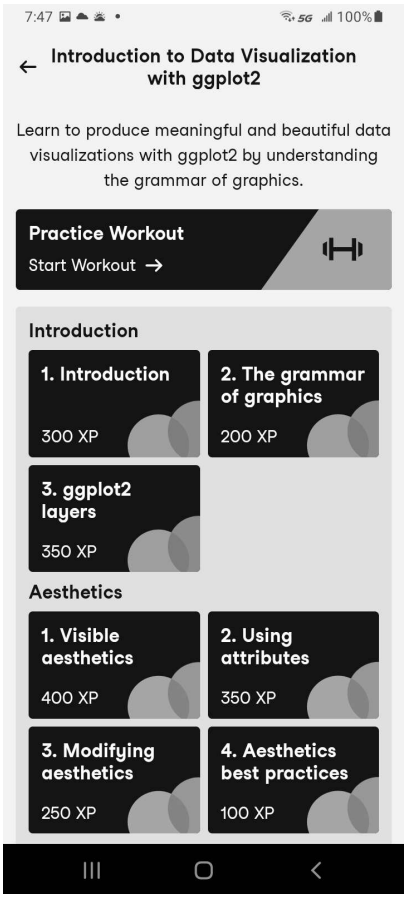


Figure 14: Introduction to ggplot2 on DataCamp mobile

Figure 15: R Programming Compiler for Android - console

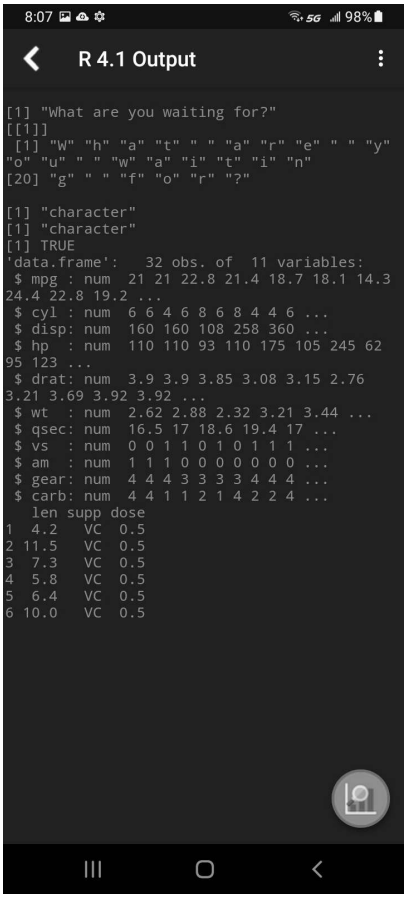


Figure 16: R Programming Compiler for Android - output

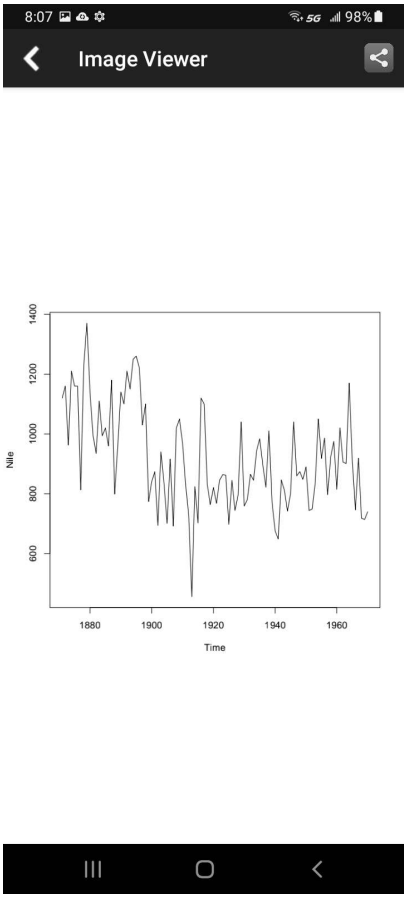


Figure 17: R Programming Compiler for Android - image viewer

Graphics with `ggplot2`

- `ggplot2` is not part of the "Tidyverse" but its creator (Hadley Wickham - yes, like the villain in Jane Austen's "Pride and Prejudice") are identical.
- We're going to do a quick tour of `ggplot2`. It is commendable for final plots, and less for interactive explorative plotting
- Plots made with `ggplot2` are harder to debug (as you can easily find out if you take a look at the DataCamp course exercises)
- Cedric Scherer has gathered `ggplot2` ressources here
- Wickham/Grolemund's "R for Data Science" book is all about `ggplot2`, visualization and EDA. It's not an easy book.

References

- Birkenkrahe (Jan 11, 2022). Interactive shell vs. interactive notebook (literate programming demo). URL: youtu.be/8HJGz3IYoHI.
- Borwein/Bailey (April 22, 2013). The Reinhart-Rogoff error – or how not to Excel at economics [article]. URL: theconversation.com
- DataCamp (January 2022). Data Trends and Predictions 2022 [webinar]. URL: www.datacamp.com.
- Cirillo (2022). The Anti-IF Campaign [website}. URL: francescocirillo.com.
- Emacs Speaks Statistics (Mar 19, 2021). First Steps With Emacs [video]. URL: youtu.be/1YOrd7NCGkg.

- Hoffman (Sep 30, 2019). What is a checksum (and why should you care)? [blog]. URL: www.howtogeek.com.
- Hughes (Oct 30, 2015). Every Linux Geek Needs To Know Sed and Awk. Here's Why…[blog]. URL: www.makeuseof.com.
- Kenlon (March 10, 2020). Getting started with Emacs [blog]. URL: opensource.com.
- Matloff (2020). Tidyverse Sceptic - An opinionated view of the Tidyverse "dialect" of the R language. URL: github.com.
- Pearson (2019). Exploratory Data Analysis Using R. CRC Press. URL: routledge.com.
- RegexOne (2021). Lesson 1: An Introduction, and the ABCs [tutorial]. URL: regexone.com.
- Sweigart (2019). Automating the boring stuff with Python. NoStarch. URL: nostarch.com/automatestuff2.
- System Crafters (March 8, 2021). The Absolute Beginner's Guide to Emacs [video]. URL: youtu.be/48JlgiBpw_I.
- teampay (2022). The 7 Biggest Excel Mistakes of All Time [blog]. URL: teampay.co.
- vonjd (n.d). Why R for Data Science – and *Not* Python! [blog]. URL: blog.ephorie.de.
- Wickham/Grolemund (2017). R for Data Science. O'Reilly. URL: r4ds.had.co.nz.
- xkcd (n.d.). Perl Problems [cartoon]. URL: xkcd.com.

Footnotes:

[1] Who wouldn't want to save the whales! Still, even a seeminly harmless ideological thrust can lead to conflict. E.g. what if you only have enough project budget to either save the whales or starving children? That used to be a question for philosophy class - in data science, it's everybody's task - because data science is decision science.

[2] As a graduate of the Google certificate course, he is likely to be an RStudio user.

[3] However, the completion rate of MOOCs is famously low, below 15%.

[4] As the names suggest, data science is an umbrella term. The science, or research workflow includes analysis. Data analysis emphasizes the service character - you address a client's business problem using analytical methods.

Author: Marcus Birkenkrahe
Created: 2022-03-11 Fri 13:39
Validate