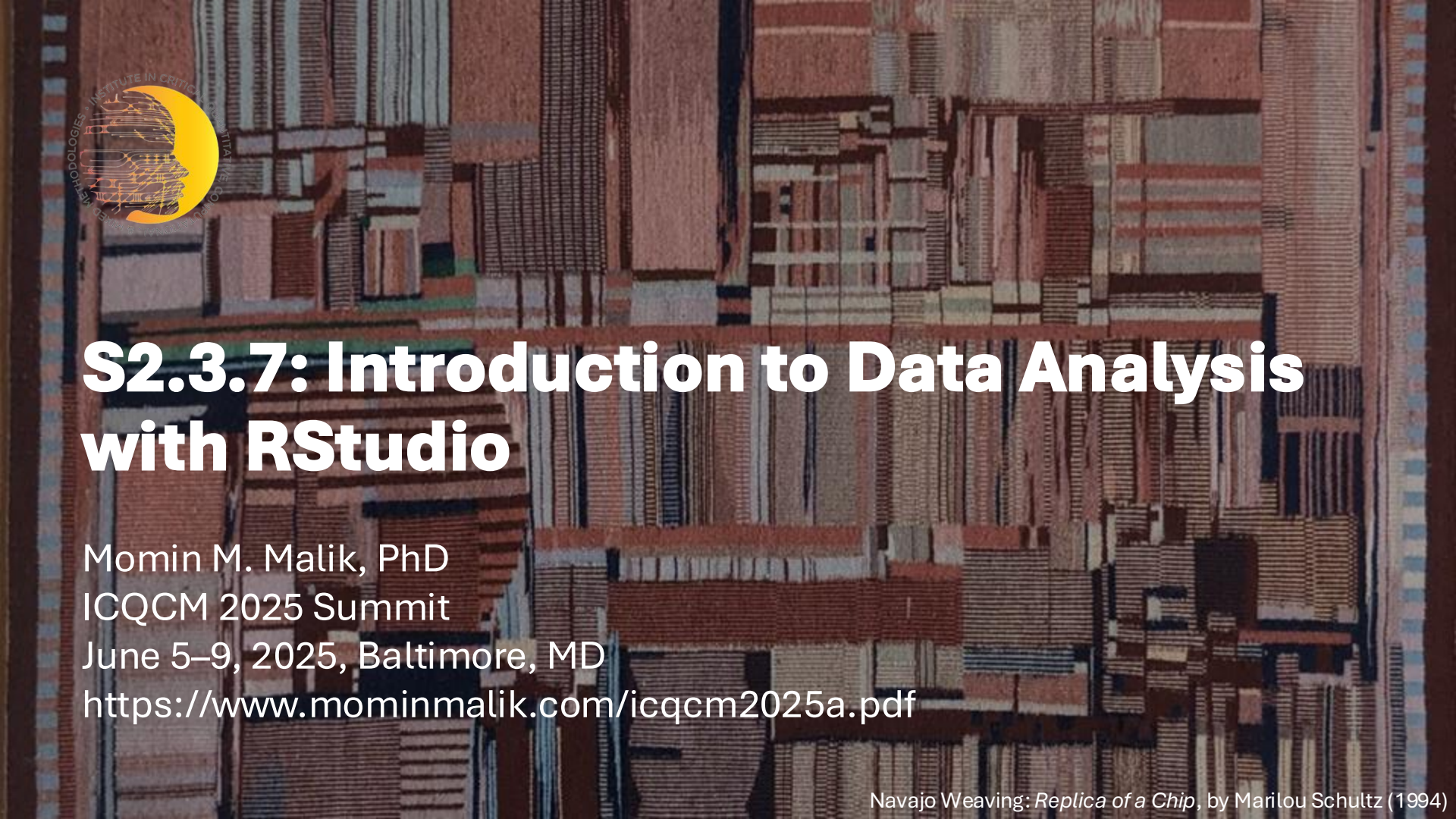

S2.3.7: Introduction to Data Analysis with RStudio

Momin M. Malik, PhD
ICQCM 2025 Summit
June 5–9, 2025, Baltimore, MD
https://www.mominmalik.com/icqcm2025a.pdf

Navajo Weaving: *Replica of a Chip*, by Marilou Schultz (1994)

Outline

- Getting started
- Background and context of R and RStudio:
  - Where they came from, what they do, how they compare to other statistical software and development environments
- Demo: Orientation to RStudio and basics of R
- Demo: Doing a bootstrap in R

Goals

- Not be intimidated by statistical programming!
- See enough to feel comfortable getting started yourself
- Have some sense of what is in RStudio and navigating around it
- Nothing more!!

How to start

- "R Programming Tutorial" from freeCodeCamp (2.5h video): https://youtu.be/_V8eKsto3Ug
  - Tacit knowledge/sociocultural learning theory: this live session may be similar *content*, but in a meaningful *context*
- Verzani, J. (2002). simpleR – Using R for introductory statistics. http://www.math.csi.cuny.edu/Statis tics/R/simpleR/printable/simpleR.pdf
  - More than 20 years old and still good!
- Cook, J. (2012). The R language: The good, the bad, & the ugly. https://youtu.be/6S9r_YbqHy8
  - How R fits into the landscape of programming languages (R is the best choice for "interactive data analysis", but not much else)
- Bodwin, K. (2024). Keep R weird. https://youtu.be/KOQBfC1WPwM
  - Good recent talk in defense of the things computer scientists hate about R
- Wickham, H., & Grolemund, G. (2017). *R for data science: Visualize, model, transform, tidy, and import data*. https://r4ds.had.co.nz/index.html

Foundational resources (probably not worthwhile)

- "Blue book": Becker, R. A., Chambers, J. M., Wilks, A. R. (1988). *The new S Language: A programming environment for data analysis and graphics*. Pacific Grove, CA: Wadsworth & Brooks/Cole.
- Wickham, H. (2019). *Advanced R* (2nd ed.). Chapman & Hall/CRC Press. https://adv-r.hadley.nz/

Installing R and RStudio

- https://cran.r-project.org/
- https://posit.co/download/rstudio-desktop/
- RStudio is an "Integrated Development Environment" (IDE): not the only option, but far better than anything for Python (jupyter notebooks can run R as well, but why would you?)

Background and context

- S language invented at Bell Labs in 1976
- R is an open source version of S invented in 1993 to teach introductory statistics at the University of Aukland, New Zealand
  - S programs can usually run in R; S still exists but all developments have happened in R
- Made for one purpose and one purpose only: *interactive data analysis*
- If you compare with SAS (developed between 1966 and 1976 with its initial release in 1972), you will appreciate how amazing R is
- R tries to "black box" as many computer things as possible for the purpose of data analysis (e.g., 1/3 works, don't need 1.0/3.0); but it has much more functionality than SPSS or Stata
- Who uses R? Statisticians (including Google's Data Science team); advanced econometricians, public policy, and some finance institutions

Context: Levels of abstraction

Likelihood principle

Solution

We need both the gradient and Hessian for the IRLS updates.

$$\frac{\partial \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = \mathbf{X}^T\left(\mathbf{y} - \frac{\exp(\mathbf{X}\boldsymbol{\beta})}{\mathbf{1}+\exp(\mathbf{X}\boldsymbol{\beta})}\right) = \mathbf{X}^T\mathbf{y} - \mathbf{X}^T\boldsymbol{\mu}(\mathbf{X},\boldsymbol{\beta})$$

$$\frac{\partial^2 \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}\partial \boldsymbol{\beta}^T} = \frac{-\partial(\mathbf{X}^T\boldsymbol{\mu}(\mathbf{X},\boldsymbol{\beta}))}{\partial \boldsymbol{\beta}^T} = -\mathbf{X}^T\mathbf{W}\mathbf{X}$$

where $\mathbf{W}$ is a diagonal matrix whose ith element is $\mu(\mathbf{x}_i,\boldsymbol{\beta})(1-\mu(\mathbf{x}_i,\boldsymbol{\beta}))$, which I am getting from *The Elements of Statistical Learning*. Then, the IRLS update is derived from Newton's method,

$$\begin{aligned}
\boldsymbol{\beta}^{(t+1)} \leftarrow & \boldsymbol{\beta}^{(t)} + (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{(t)}) \\
&= (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}(\mathbf{X}^T\mathbf{W}\mathbf{X})\boldsymbol{\beta}^{(t)} + (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{(t)}) \\
&= (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\left(\mathbf{W}^{(t)}\mathbf{X}\boldsymbol{\beta}^{(t)} + (\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{(t)})\right) \\
&= (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\left(\mathbf{W}^{(t)}\mathbf{X}\boldsymbol{\beta}^{(t)} + \mathbf{W}^{(t)}{\mathbf{W}^{(t)}}^{-1}(\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{(t)})\right) \\
&= (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^{(t)}\left(\mathbf{X}\boldsymbol{\beta}^{(t)} + {\mathbf{W}^{(t)}}^{-1}(\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{(t)})\right) \\
&= (\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}^{(t)}\mathbf{z}
\end{aligned}$$

where $\mathbf{z} = \left(\mathbf{X}\boldsymbol{\beta}^{(t)} + {\mathbf{W}^{(t)}}^{-1}(\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{(t)})\right)$ is the adjusted response.

Probability, linear algebra, convex optimization

```
loglik <- function(y, X, beta) t(y) %*% X %*% beta - sum(log(1 + exp(X %*% beta)))
logistic <- function(x) exp(x)/(1 + exp(x))
weight <- function(mu) diag(c(mu * (1 - mu)), nrow(mu), nrow(mu))
adjust <- function(y, X, beta, mu, W) X %*% beta + solve(W) %*% (y - mu)
update <- function(y, X, beta, mu, W, z) solve(t(X) %*% W %*% X) %*% t(X) %*% W %*% z

niter <- 20
beta <- rep(0, ncol(X))
objective <- rep(NA,niter)
ptm <- proc.time()
for (i in 1:niter) {
  b <- beta
  mu <- logistic(X%*%b)
  W <- weight(mu)
  z <- adjust(y, X, b, mu, W)
  beta <- update(y, X, b, mu, W, z)
  objective[i] <- loglik(y, X, beta)
  if (i > 1) if (objective[i] - objective[i-1] < 1e-6) break
}
```

Software & Code

Data: $\mathbf{X} \in \mathbb{R}^{n\times p}$, $\mathbf{y} \in [0,1]^n$, and boundary for convergence $\varepsilon > 0$.
Result: Estimated $\widehat{\boldsymbol{\beta}}$.
Add bias term $\mathbf{X} \leftarrow \begin{bmatrix} \mathbf{1} & \mathbf{X} \end{bmatrix} \in \mathbb{R}^{n\times(p+1)}$;
Initialize $\boldsymbol{\beta}^{new} \leftarrow \mathbf{0} \in \mathbb{R}^{p+1}$;
while $\ell(\boldsymbol{\beta}^{new}) - \ell(\boldsymbol{\beta}^{old}) > \varepsilon$ **do**
  $\boldsymbol{\beta}^{old} \leftarrow \boldsymbol{\beta}^{new}$;
  $\mathbf{W} \leftarrow \text{diag}\left(\mu(\mathbf{x}_i,\boldsymbol{\beta}^{old})\left(1-\mu(\mathbf{x}_i,\boldsymbol{\beta}^{old})\right)\right)$;
  $\mathbf{z} \leftarrow \left(\mathbf{X}\boldsymbol{\beta}^{old} + \mathbf{W}^{-1}(\mathbf{y}-\mu(\mathbf{X},\boldsymbol{\beta}^{old})\right)$;
  $\boldsymbol{\beta}^{new} \leftarrow (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}$;
end

Algorithm

RStudio Background and context

- Open source software from RStudio PBC (which stands for Public Benefit Company), later renamed RStudio, Inc., and most recently again renamed to “Posit” to emphasize that it will focus not just on R
- Private company, but RStudio is open-source; desktop is free, they make their money from RStudio servers for businesses
- Maintains the “tidyverse” of packages, by New Zealand statistician Hadley Wickham (chief data scientist of RStudio)

Politics of R

- Judging things by what they do, R is ultimately still oppressive; it is used by economists (alongside Excel and Stata) for neoliberalism.[1] Maybe less bad than Python, C/C++ and FORTRAN themselves, COBOL (global banking), Java, etc., but not by much
- The good part of R politics: open-source, collaborative, (relatively) accessible, supportive within its borders
- There is no (digital computer) programming language that comes from a better place. LISP is maybe too obscure to have done direct harm, but is still super elite; same with "esolangs"
- They all still run on microchips (requiring rare mineral extraction), and follow the von Neumann architecture

[1]Richardson, E. (2020). Redescription 8: The race-PrEP study (counterhegemonic modeling). In *Epidemic illusions: On the coloniality of public health* (pp. 103–110). MIT Press. https://doi.org/10.7551/mitpress/12550.003.0013

Pros/cons of R (or, are you wasting your time?)

- R is the best environment for *interactive data analysis*
  - \+ RStudio is a fantastic environment, as an add-on to R
  - \+ R and RStudio are **self-contained**, unlike Python
  - \+ R has some of the best packages for specialized statistical applications, including the ”tidyverse”
  - \+ It is open source, with a fantastic community for support
  - \+ Its core linear algebra routines are written in FORTRAN, so are extremely efficient (more than Python!!)
  - – It is terrible *as a programming language*
  - – It is not great at things *other* than interactive data analysis
  - – Python is better for integrating with business applications and web development
  - – Machine learning is happening more and more in Python

R style guide

- Many choices around things like line breaks, spacing, and comments (especially in R, as opposed to Python) are arbitrary. For the sake of consistency, we should pick some convention and stick to it
  - Following a style guide will make your code look more professional, and even (for a very small set of style guide recommendations) make it run better
- The most comprehensive style guide is by Hadley Wickham, at https://style.tidyverse.org/. Most of this guide is about the tidyverse of packages, but some sections apply to base R.

Advice on mistakes

- You will make TONS of typos or other minor mistakes
  - This means nothing. I still make the same mistakes; the main difference is that I can correct them much more quickly
- You will sometimes spend hours on something really small, only to eventually find a one-line solution or function
  - *This is part of learning*. Don't see that time as wasted. You have to go through lots of those, repeatedly, to learn
- Your code will start sloppy. I say, don't worry about it! Don't be afraid to copy bits and pieces! (and lots of versions: "final.R", "final_v2.R") So long as it works.
- Rely on searching for errors/tasks online!! I still do this all the time; again, the main difference is I can quickly tell if something is what I am looking for

“Cheat sheets”

- Programming “languages” have something like grammar, syntax and vocabulary
- Vocabulary is based on English, but there’s no way to know vocabulary beforehand. E.g., Python: `len()` and `histogram()`, R: `length()` and `hist()`. Eventually you memorize
- “Cheat sheets” can help.
  - https://github.com/rstudio/cheatsheets/
  - “Base R”: https://github.com/rstudio/cheatsheets/blob/main/base-r.pdf
  - RStudio interface: https://github.com/rstudio/cheatsheets/blob/main/rstudio-ide.pdf
  - Base R plotting:
    - https://www.gastonsanchez.com/r-graphical-parameters-cheatsheet.pdf
    - https://www.r-graph-gallery.com/6-graph-parameters-reminder.html (a page, not a cheat sheet)
    - http://publish.illinois.edu/johnrgallagher/files/2015/10/BaseGraphicsCheatsheet.pdf

Demo!!


```
df <- read.csv(
"https://github.com/embruna/cruz_nsf_database/raw/refs/heads/main/data_clean/cruz_data_clean.csv"
)
```