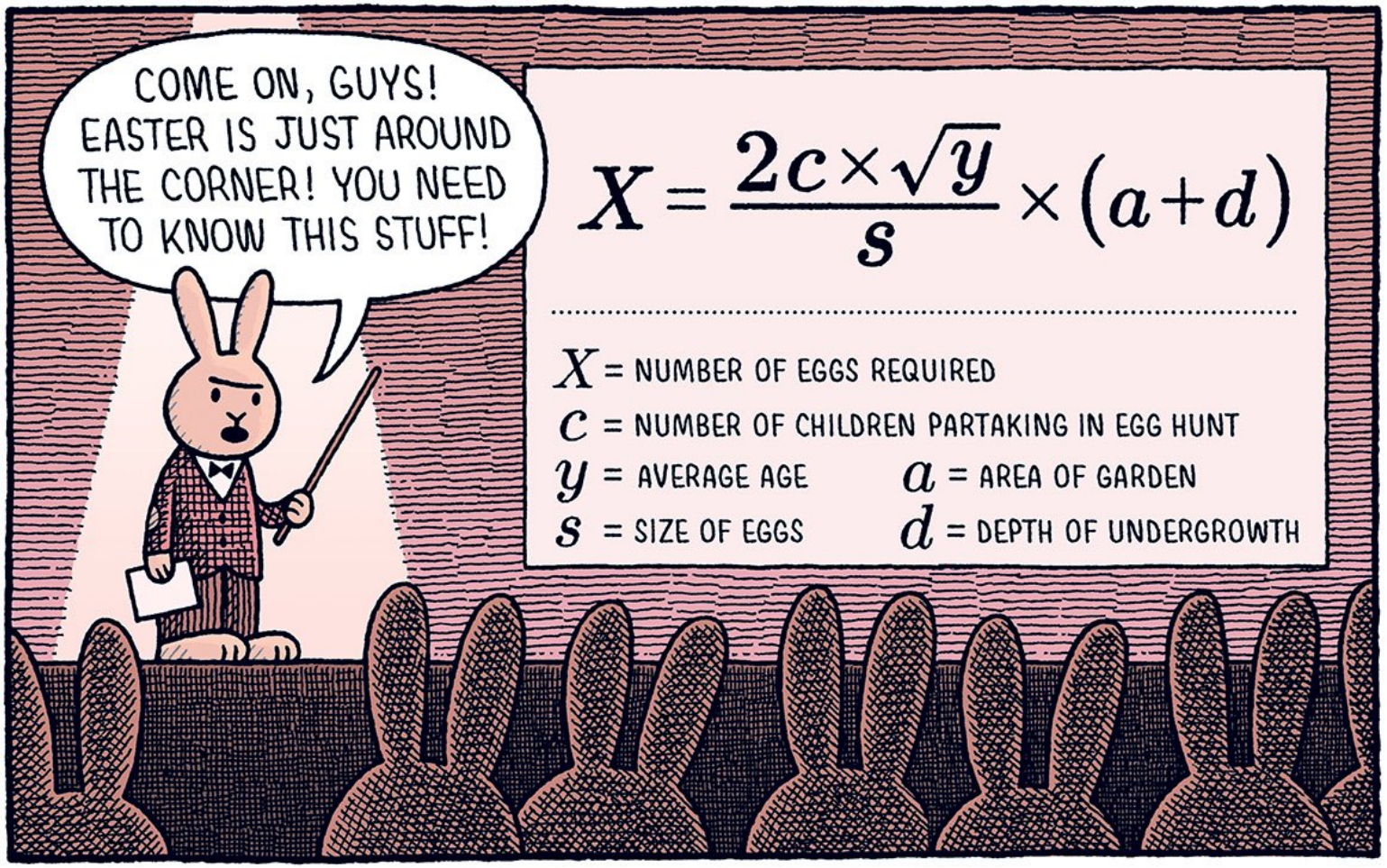
ACCORDS analyst stats meeting

Build a R package for Your Analysis

Randy Jin

2025-05-22

As a biostatiscian ...



As a biostatiscian ...

We need to speak in three different Languages, I guess:

As a biostatiscian ...

We need to speak in three different Languages, I guess:

- Statistical language
  - notations, equations, and distributions
  - such as $\boldsymbol{X\beta} \sim \textit{Normal}\left(\boldsymbol{\mu} = \left(\begin{smallmatrix}\mu_1\\ \mu_2\end{smallmatrix}\right), \Sigma = \begin{pmatrix}\sigma_1^2 & \rho\sigma_1\sigma_2\\ \rho\sigma_1\sigma_2 & \sigma_2^2\end{pmatrix}\right)$

As a biostatiscian ...

We need to speak in three different Languages, I guess:

- Statistical language
  - notations, equations, and distributions
  - such as $\boldsymbol{X\beta} \sim \textit{Normal}\left(\boldsymbol{\mu} = \left(\begin{smallmatrix}\mu_1\\ \mu_2\end{smallmatrix}\right), \Sigma = \begin{pmatrix}\sigma_1^2 & \rho\sigma_1\sigma_2\\ \rho\sigma_1\sigma_2 & \sigma_2^2\end{pmatrix}\right)$
- Programming language
  - R and SAS
  - python
  - git and many more

As a biostatiscian ...

We need to speak in three different Languages, I guess:

- Statistical language
  - notations, equations, and distributions
  - such as $\boldsymbol{X\beta} \sim \textit{Normal}\left(\boldsymbol{\mu} = \left(\begin{smallmatrix}\mu_1\\ \mu_2\end{smallmatrix}\right), \Sigma = \begin{pmatrix}\sigma_1^2 & \rho\sigma_1\sigma_2\\ \rho\sigma_1\sigma_2 & \sigma_2^2\end{pmatrix}\right)$
- Programming language
  - R and SAS
  - python
  - git and many more
- Communicating language
  - write a report
  - meet with team
  - do a presentation

As a biostatiscian ...

- Statistical thinking
  - guides our approach to data and uncertainty

As a biostatiscian ...

- Statistical thinking
  - guides our approach to data and uncertainty
- Programming
  - enables us to implement models, wrangle data, and build reproducible workflows

As a biostatiscian ...

- Statistical thinking
  - guides our approach to data and uncertainty
- Programming
  - enables us to implement models, wrangle data, and build reproducible workflows
- Communication
  - allows us to connect with collaborators, translate results, and support decision making

These are not ends in themselves!

How can I put all the things together?

Why R package + qmd?

- Standardization:
  - Packages follows established conventions
  - Packages keeps consistent development practices and tools

R package structure

An R package usually includes the following components:

- Functions
- Data (optional)
- Documentation (function manuals and examples)
- Vignettes (longer descriptions of package function usage)
- Tests (R scripts to test the package functions)

Why R package + qmd?

- Standardization:
  - Packages follows established conventions
  - Packages keeps consistent development practices and tools

R package structure

```
R Project directory structure:
| - DESCRIPTION
| - LICENSE
| - NAMESPACE
| - R
|   | - script1.R
| - data_raw
|   | - data1.csv
| - man
|   | - function1.Rd
|   | - function2.Rd
| - randy_package.Rproj
```

Why R package + qmd?

- Organization and Reusability
  - Package helps structure and organize code
    - ggplot color theme and style
    - figure size and font
    - multiple reports with the same directory
  - Package makes it easier to reuse data across multiple projects
  - Package makes it easier to build a package
    - as well as keep them organized
    - do not even need to know that much R

```
library(devtools)
library(use_this)
library(testthat)

devtools::load_all() ## library(scope)

## create a new package
usethis::create_package("randy_package")

## add new functions in R/
use_r("my_summaries")

devtools::document()

## add a raw dataset
usethis::use_data_raw("mydataset0")
## add a new dataset with each version
usethis::use_data(mydataset1)
```

Why R package + qmd?

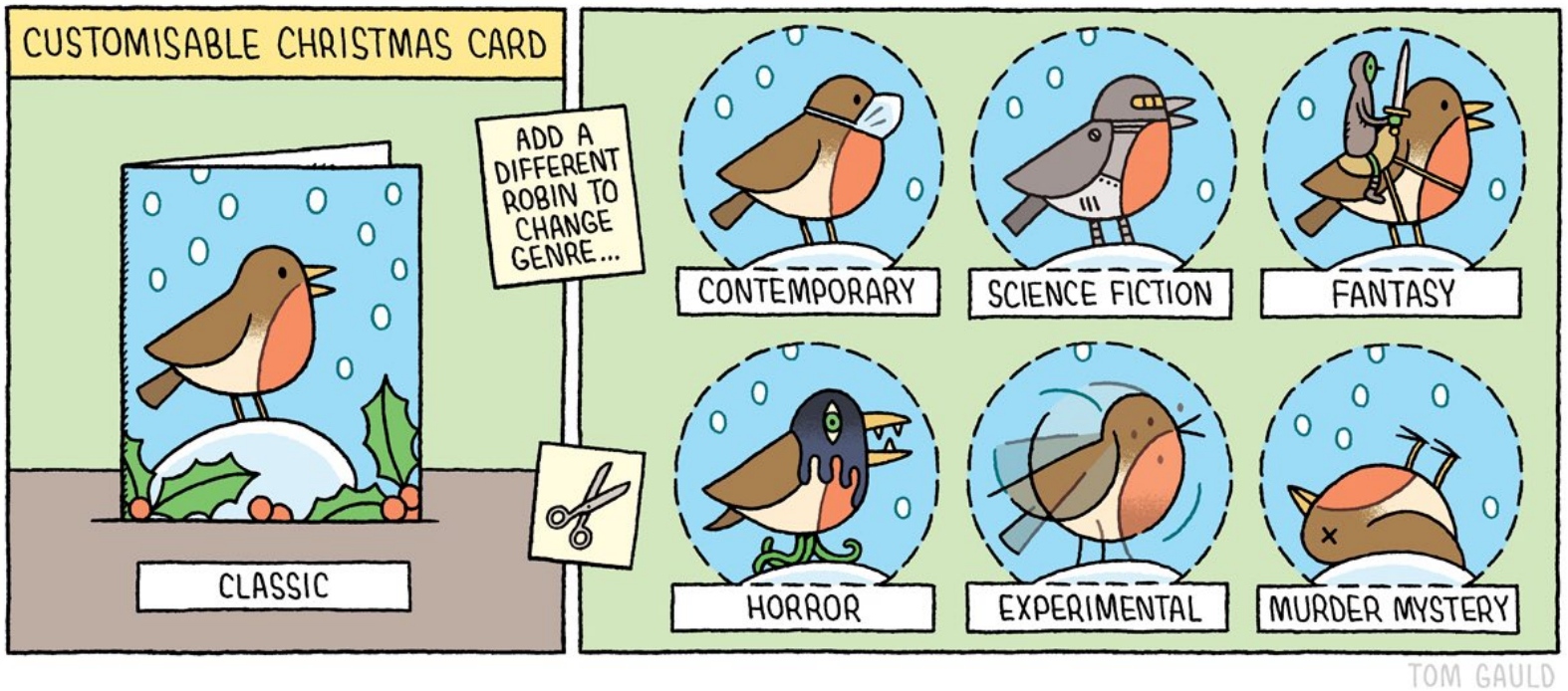
- Sharing and Maintaining:
  - They facilitate the sharing of code and report with others
  - They are easy work with Github
- Reproducibility:
  - They help ensure reproducibility
  - They make it easier to do a report
    - `html` format with shiny app
    - `pdf` format with Latex
    - `docx` format with quarto/rmarkdown
- Testing and Debugging
  - Package makes test and debug functions easier
  - Package keeps the reliability of the code (test files)

Small Tweaks that Make a Big Difference

- Never use default!!!
- And make it automatic, if possible “scope::template”
- Add something special for your package
  - ☒ Build a checklist here
    - Say something positive to yourself “r praise::praise()” You are primo!
    - Add an emoji to yourself
- Everybody should has their own theme (style)
  - Use the font you and your team like
  - Find the color theme for your organization
  - Put your theme in the package

Small Tweaks that Make a Big Difference

- DIY (base on other people's Codes)
  - David Keyes
  - Hadley Wickham
  - Jenny Bryan



Finally

- "If you hate it, you should do it more often." – Jenny Bryan
- "If you hate it, you should automate it." – UiPath