Tips & techniques for a more reproducible R environment

2023-06-27 Keboola

About me

Jindra Lacko

PhD student KEKO VŠE

jindra@jla-data.net

www.jla-data.net

@jladata

jlacko

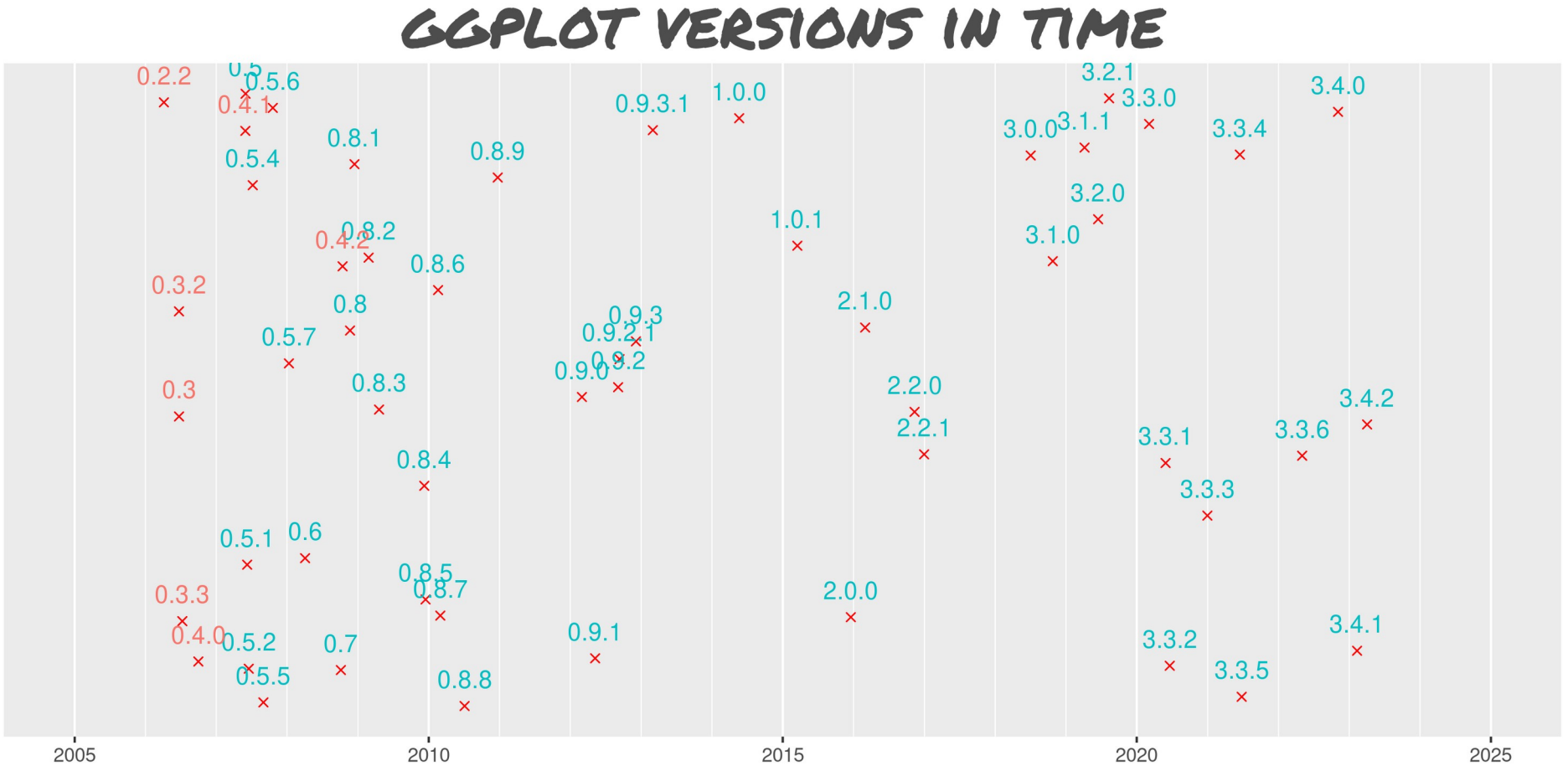
The why of reproducibility

- Tempora mutantur et nos mutamur in illis
- The Times They Are A-Changin'
- Both R itself, and the package ecosystem surrounding it, are actively developed
- Resources – including maintainer's time – are finite, and complexity comes at a cost
- One dev's errors are another dev's features

Evolution of R itself

- x.y.0 release each spring, patch releases "as required"
- Each x.y.0 release forces fresh install of packages
- Complicated system of checks & balances ensuring all current CRAN package versions play along nicely

Evolution of a package



Evolution of a package

- A nova-like outburst of initial creativity
  - few early adopters
  - breaking changes are common, including a fresh start
- Growing up period
  - user base builds up
  - breaking changes are infrequent (but possible)
- Plateau of stability
  - package becomes a part of the canon
  - active development happens in extending packages

3 levels of reproduciblity

1) A piece of code (often someone else's) fails to run as expected
2) Bulletproofing a piece of code for future you to run the same forever
3) Preparing a piece of code for anyone to run & get expected output

Making sense of code past

- Encountering old & buggy code in:
  - Company codebase (including yours!)
  - Supplement to a published paper
  - Blog posts / Stack Overflow answers
- Ask for `sessionInfo()` output (if possible)
- Look for the date of creation / publishing

Turning back the time…

- `{groundhog}` package
  - groundhog = genus *Marmota* / svišť
  - `groundhog.library(package, "date")`
- Historized RStudio package archive
  - `install.packages(package, repos = "https://packagemanager.rstudio.com/cran/YYYY-MM-DD")`

Known limitations

- Both approaches work within the current R version (can't go safely too far) + operating system
- groundhog will work in current session only (for good or bad)
- RStudio package manager will install packages globally (for good or bad)
- A chance of messing up your package install

Fixing R Environment

- `{renv}` package
  - Creates lockfile `renv.lock` (a JSON) describing currently used packages
  - Works on project basis = tied to RStudio IDE
  - Call `renv::init()` to set up
  - `renv::snapshot()` to create and
  - `renv::restore()` to restore from lockfile

Stabilizing at OS level

- Docker
  - Eliminates the "it works on my machine" excuse
  - Creates a fully isolated image of OS level libraries, R, installed packages, your data + scripts (full stop)
  - Plays nicely with other IDEs (VS Code)

Benefits of fixing versions

- The chance of unexpected code output is greatly minimized
- Less work on maintaining old code, more time to play
- Fewer surprises – especially in public

Costs of fixing versions

- Things happen for a reason – including breaking changes in R packages
- No shiny new features!
- Loss of contact with R community
- Complexity of living both in the past and present (maintaining old & creating new code)

Summary


EXPLORE THE PAST
BEND FUTURE TO YOUR WILL
SHIP YOUR COMPUTER

Thank you

https://github.com/jlacko/r-meetup-reproducibility