The R System: From Open Source to Open Science
An Insider’s View

Achim Zeileis

https://eeecon.uibk.ac.at/~zeileis/
Overview

**R:**
- System for statistical computing.
- Open-source software under General Public License (GPL).
- [https://www.R-project.org/](https://www.R-project.org/)

**Insider:** Achim Zeileis.
- Statistician.
- Co-editor: Journal of Statistical Software.
- Ordinary member: R Foundation.
- Co-creator: useR! conference, R-Forge, ...
What is R?

**Based on:** ACM award-winning S language (core of commercial S-PLUS).

**Early 1990s:** Ross Ihaka and Robert Gentleman start reimplementations, with underlying semantics derived from Scheme, eventually called **R**.

**Since 1997:**
- Base system developed by R Core Team.
- Highly extensible through packages.
- Openly shared through Comprehensive R Archive Network.

**Since 2000s:** Lingua franca in statistics. Around \( \sim 100 \) CRAN packages in 2000, more than 15,000 today (\( \sim 25\% \) nominal growth rate per year).

What is R?

Vantage points:

- Data analysis vs. programming.
- Statistics vs. data science.
- Community vs. app.
- Science vs. commerce.
What is R used for?

Classically: Statistics and graphics.
Linear regression, two-sample tests, scatter plots, bar charts, …

slope = -0.29
p = 6.5e-09

Before
After
t-Test: t = 4.67, p = 2.9e-05
Wilcoxon–Mann–Whitney:
U = 635.5, p = 5.5e-05
What is R used for?

**Diversified methods:** Machine learning, social network analysis, econometrics, environmetrics, psychometrics, ...
What is R used for?

**Data structures:** Genomic data, spatial and space-time data, surveys, text corpora, connections to databases, ...
What is R used for?

**Specific applications:** Bioinformatics, business analytics, atmospheric sciences, finance, natural language processing, ...
Why is R so successful?

- Open source.
- By statisticians for statisticians (in a very broad sense).
- Highly modular and extensible.
- Many subcommunities.
- Spillovers through joint journals, conferences, ... 
- “Big Data Science.”
How does the R community work?

R Core/Foundation
Base system
CRAN
Mailing lists
How does the R community work?

**R Core/Foundation**
- Base system
- CRAN
- Mailing lists

**Scientific conferences**

**Scientific journals**
- *Journal of Statistical Software*
How does the R community work?

R Core/Foundation
Base system
CRAN
Mailing lists

(Scientific) conferences

Scientific journals

The Journal of Statistical Software

The R Journal
How does the R community work?

- **R Core/Foundation**
  - Base system
  - CRAN
  - Mailing lists

- **Scientific conferences**

- **Scientific journals**
  - *Journal of Statistical Software*

- **Code collaboration**
  - Bioconductor

- **The R Journal**
How does the R community work?

R Core/Foundation
- Base system
- CRAN
- Mailing lists

(Scientific) conferences

Scientific journals
- Journal of Statistical Software
- The R Journal

Code collaboration
- Bioconductor
- R-Forge
How does the R community work?

**R Core/Foundation**
- Base system
- CRAN
- Mailing lists

**Scientific journals**
- Journal of Statistical Software
- The R Journal

**Code collaboration**
- Bioconductor
- R-Forge
- GitHub
- OpenSci

**(Scientific) conferences**
How does the R community work?

R Core/Foundation
- Base system
- CRAN
- Mailing lists

(SCientific) conferences

Scientific journals
- Journal of Statistical Software
- The R Journal

Code collaboration
- Bioconductor
- R-Forge
- GitHub
- OpenSci

Communication
- R-bloggers
How does the R community work?

**R Core/Foundation**
- Base system
- CRAN
- Mailing lists

**Scientific journals**
- Journal of Statistical Software
- The R Journal

**Code collaboration**
- Bioconductor
- R-Forge
- GitHub
- OpenSci

**Communication**
- R-bloggers
- stackoverflow [r]
- #rstats

**Scientific conferences**
How does the R community work?

R Core/Foundation
- Base system
- CRAN
- Mailing lists

Scientific journals
- Journal of Statistical Software
- The R Journal

(Scientific) conferences

Code collaboration
- Bioconductor
- R-Forge
- GitHub
- OpenSci

Communication
- R-bloggers
- stackoverflow [r]
- #rstats

Other players
- R Studio
- Microsoft
- Ladies
How does the R community work?

R Core/Foundation
- Base system
- CRAN
- Mailing lists

Scientific journals
- Journal of Statistical Software
- The Journal of Open Source Software

(Scientific) conferences
- eRum
- rstudio::conf

Scientific conferences
- Xi’an R at COBRA

Other players
- Bioconductor
- R-Forge
- GitHub
- OpenSci
- R-studio

Code collaboration
- Biocconduct
- R-Forge
- GitHub
- OpenSci

Communication
- R-bloggers
- stackoverflow [r]
- #rstats
Why do you contribute to the R community?

In 1999: Undergraduate.

- “Why do you use R? We do have an S-PLUS license.”
- Open source!
Why do you contribute to the R community?

**In 1999:** Undergraduate.
- “*Why do you use R? We do have an S-PLUS license.*”
- Open source!

**In 2002:** PhD student.
- “*Why do you publish in online-only journals? That’s just like a technical report.*”
- Open access (free for everyone)!
Why do you contribute to the R community?

**In 1999:** Undergraduate.
- "Why do you use R? We do have an S-PLUS license."
- Open source!

**In 2002:** PhD student.
- "Why do you publish in online-only journals? That’s just like a technical report."
- Open access (free for everyone)!

**Since 2004:** Postdoc onwards.
- "Why do you volunteer to edit a free journal and organize conferences? You should make some money."
- Open and reproducible science!
From open source to open science in R

**R package system:**

- Inspired by the Debian package system.
- Standard way of sharing R code with easy installation.
- Packages can contain: R code, source code (e.g., C, C++, Fortran, ...), data, manual pages, other documentation, examples, tests, demos, ...
- Package can **depend** on other packages.
- CRAN (Comprehensive R Archive Network) is the main repository for sharing R packages (sources plus binary versions). Daily checks of the entire network on different platforms.
- Other repositories include Bioconductor, R-Forge but also platforms such as GitHub or Bitbucket.
From open source to open science in R

Dynamic documents:

• **2002**: Sweave introduced in R 1.5.0 by Friedrich Leisch.
  Ideas: From literate programming to literate data analysis.
  Basis for: R package vignettes.
  First (?) contributed example: strucchange based on previous JSS paper.

• **2006**: weaver in Bioconductor by Seth Falcon.
  Ideas: Caching code chunks.

• **2012**: knitr by Yihui Xie.
  Initial ideas: Extended/improved Sweave alternative.
  Later: R/Markdown support.

• **2014**: rmarkdown by J.J. Allaire, Yihui Xie, et al.
  Ideas: Wide range of output formats via pandoc. Flexibility/extensibility.
From open source to open science in R


- Integrate computations, code, and data used in empirical analyses, simulations, etc.
- In R based on package infrastructure, leveraging dynamic documents (*Sweave, knitr, ...*) and automatic checking.
From open source to open science in R


- Integrate computations, code, and data used in empirical analyses, simulations, etc.
- In R based on package infrastructure, leveraging dynamic documents (*Sweave, knitr, ...*) and automatic checking.

**In 2010s:**

- Support R package development: *devtools* (Wickham), *roxygen2* (Danenberg, Eugster, Wickham), ...
- Package reproducibility: *checkpoint* (Microsoft), *packrat* (RStudio), ...
- Pipelines and workflows: *drake* (Landau), ...
Scientific review of R packages

*Journal of Statistical Software*

- Founded in 1996 by Jan de Leeuw for publishing statistical software (any language).
- Publication: Software – along with paper plus replication materials.
- Papers: Explains statistical technique, code, and provides examples.
- Trick: Fit publication of software into classical journal publications.
- Review: Classical single-blind review.
- Audience: Broad community of practitioners, teachers, and researchers in the field of statistics
- Emphasis: Careful review of existing implementations and discussion of relative (dis)advantages. Interesting case study highlighting the typical workflow.
Scientific review of R packages

The R Journal

- Founded in 2001 as R News.
- Papers: Reasonably short, clearly written, not too technical, focused on R.
- Content: Not just packages – also reviews, comparisons, or applications.
- Review: Classical single-blind review.
- Audience: Users and developers of R.
Scientific review of R packages

- Founded in 2011 as non-profit initiative for reproducible data retrieval.
- Peer review process for R packages.
- Focus: Data processing and infrastructure.
- Submission: Open in GitHub repository.
- Review: Open conversation using GitHub issues.
- Recommendations for good practices: Naming conventions, code style, documentation, testing.
- Journal publication: Subsequently in *Journal of Open Source Software*. 
Implementation strategies

Task: Turn conceptual statistical tools into computational tools

Goals: Desirable features.

- Easy to use.
- Numerically reliable.
- Computationally efficient.
- Flexible and extensible.
- Reusable components.
- Object-oriented.
- Reflect features of the conceptual method.

Problem: Often antagonistic, e.g., computational efficiency vs. extensibility.
Implementation strategies

**Guiding principle:** The implementation should be guided by the properties of the underlying statistical methods while trying to ensure as much efficiency and accuracy as possible.

*The resulting functions should do what we think a method does conceptually.*

**In practice:** Many implementations are instead guided by the limitations that programming languages used to have where everything had to be represented by numeric vectors and matrices.

**Question:** What language features are helpful for improving this?
Implementation strategies

**Object orientation:** Create objects of a certain class with methods performing typical tasks. Particularly easy: S3 paradigm, based on generic functions.

**Functions as first-class objects:** Functions are a basic data type that can be passed to and returned by another function.

**Lexical scope:** Returned nested *lexically scoped functions* can have free variables stored in function closure.

**Compiled code:** Combine convenience of interpreted code and efficiency of (byte) compilation or dynamic linking.

**Reusable components:** Build on standard and widely-used tools. Likewise, provide tools that other implementations can build on.
Illustration: Heteroscedastic censored regression

Examples:

- Linear regression in base R.

Illustration:

- Precipitation forecasts for Innsbruck, Austria (\texttt{RainIbk}).
- Observed 3 day-accumulated precipitation amounts (\texttt{rain}) from SYNOP station Innsbruck Airport from 2000-01-01 to 2013-09-17.
- Corresponding GEFS 11-member ensemble reforecasts of total accumulated precipitation between 5 and 8 days in advance with mean \texttt{ensmean} and standard deviation \texttt{ensisd}. 
Illustration: Heteroscedastic censored regression

Hence:

• Gaussian model on square root scale.
• Account for zeros by “censoring”.
• Mean precipitation explained by GEFS ensemble mean.
• Standard deviation explained by GEFS standard deviation (using a log link).
Hence:

- Gaussian model on square root scale.
- Account for zeros by "censoring".
- Mean precipitation explained by GEFS ensemble mean.
- Standard deviation explained by GEFS standard deviation (using a log link).
Illustration: Heteroscedastic censored regression

Hence:

- Gaussian model on square root scale.
- Account for zeros by “censoring”.

$R$
Hence:

- Gaussian model on square root scale.
- Account for zeros by “censoring”.
- Mean precipitation explained by GEFS ensemble mean.
- Standard deviation explained by GEFS standard deviation (using a log link).
Illustration: Heteroscedastic censored regression
Illustration: Heteroscedastic censored regression
Illustration: Heteroscedastic censored regression
Illustration: Heteroscedastic censored regression
Implementation strategies in practice

**Example:** Linear regression in R.

- **Object orientation:** `lm()` returns an “lm” object with suitable methods and extractor functions.
- **Reusable components:** Underlying workhorse `lm.fit()` without pre- and postprocessing is also provided.
- **Compiled code:** At its core `lm.fit()` has a `.Fortran("dqrls", ...)` call.
Implementation strategies in practice

Example: Linear regression in R.

- **Object orientation:** `lm()` returns an “lm” object with suitable methods and extractor functions.
- **Reusable components:** Underlying workhorse `lm.fit()` without pre- and postprocessing is also provided.
- **Compiled code:** At its core `lm.fit()` has a `.Fortran("dqrls", ...)` call.

Application:
```
R> m1 <- lm(rain ~ ensmean, data = RainIbk)
R> coef(m1)
   (Intercept)   ensmean
      0.147       0.582
R> vcov(m1)
   (Intercept)   ensmean
   (Intercept)  0.003026 -0.000772
    ensmean    -0.000772  0.000240
```
Implementation strategies in practice

**Note:** S3 uses naming convention for methods (plus registration).

**Here:** `coef.lm()` and `vcov.lm()`.
Implementation strategies in practice

Note: S3 uses naming convention for methods (plus registration).

Here: coef.lm() and vcov.lm().

Also: Smart default methods like confint.default() that reuse other methods.

R> confint(m1)

2.5 % 97.5 %
(Intercept) 0.0395 0.255
ensmean 0.5512 0.612
Implementation strategies in practice

Note: S3 uses naming convention for methods (plus registration).

Here: `coef.lm()` and `vcov.lm()`.

Also: Smart default methods like `confint.default()` that reuse other methods.

```
R> confint(m1)
     2.5 % 97.5 %
(Intercept) 0.0395 0.255
ensmean 0.5512 0.612
```

Furthermore: Methods for inference, prediction, data handling, etc.
Implementation strategies in practice

**Package:** *crch* for censored regression with conditional heteroscedasticity.

- **Object orientation:** Object structure and methods mimic `lm()`/`glm()`.
- **Functions as first-class objects:** Model components can be supplied as functions, e.g., the log-likelihood (and its gradient and Hessian) or the link function (and its inverse and derivative).
- **Lexical scope:** Log-likelihood (and its gradient and Hessian) may access data through lexical scoping.
- **Compiled code:** Density/score/Hessian functions for censored distributions are implemented in C.
- **Reusable components:**
  - Uses `lm.fit()` for starting values.
  - Provide: `crch.fit()` and `dcnorm()` for other applications.
  - Optional: CRPS estimation from *scoringRules*. 
Implementation strategies in practice

```r
R> library("crch")
R> m2 <- crch(rain ~ ensmean | log(enssd), data = RainIbk, left = 0)
R> coef(m2)

  (Intercept) ensmean (scale)_(Intercept) (scale)_log(enssd)
     -0.840       0.783       0.687         0.220

R> confint(m2)

          2.5 % 97.5 %
(Intercept)  -0.983  -0.697
ensmean      0.743    0.822
(scall)_(Intercept) 0.662    0.712
(scall)_log(enssd) 0.161    0.279
```
Implementation strategies in practice

R> library("crch")
R> m2 <- crch(rain ~ ensmean | log(enssd), data = RainIbk, left = 0)
R> coef(m2)

(Intercept)      ensmean (scale)_(Intercept)
   -0.840          0.783          0.687

(scaled) log(enssd)
     0.220

R> confint(m2)

          2.5 % 97.5 %
(Intercept) -0.983 -0.697
ensmean      0.743  0.822
(scaled)_(Intercept) 0.662  0.712
(scaled) log(enssd)  0.161  0.279

Reproducibility: Paper from the The R Journal is provided in the package as a dynamic Sweave vignette.
Where are we going from here?

**Quite certainly:** More growth and more diversity.

**High potential:** Exciting and innovative collaborations across disciplines.

**Unclear:** Whether “one” R community will persist.

**Crucial:** Communication and exchange within and beyond the community.

**Strength:** Rooting in academia, tools for reproducibility and open science.