Package Testing in R

Hector Rodriguez-Deniz
Division of Statistics and Machine Learning (STIMA)
Linköping University

LINKÖPING UNIVERSITY

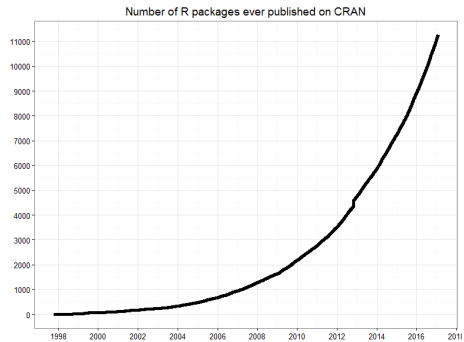
WASP | WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

R and RStudio

- R is a programming language for statistical computing and graphics
- Niche language for data analysis (Statistics, Machine Learning...)
- Strong community - vast amount of packages
- API's to high-performance languages as C/C++ and Java
- RStudio is the most popular open-source IDE for R (multiplatform)
- Code editor, visualization tools, debugging, profiling, Git...

R Packages

- The fundamental unit of shareable code in R
- Bundles together code, data, documentation, and tests
- Easy to share with others
- 14,000 packages (March ’19)
- Growing fast



Software Project

- R package with a simple implementation of Linear Regression
- Ordinary Least-Squares and Bayesian estimation
- Basic functionality: fitting the model, plot, coefficients, residuals...
- Developed in RStudio and checked with Travis-CI
- Tests implemented using *testthat* and Codecov
- ”Object Oriented Programming” via S3 classes
- Github project: https://github.com/hecro459/LinReg

Testing Package Consistency

- I used Travis-CI (https://travis-ci.org/) with my R Package
- Automatically build and test software projects hosted at GitHub
- Travis will do it for you every time you commit and push
- Will run R CMD check: automatically checks R package consistency
- Rebuild package from scratch, syntax, correct documentation...
- It will also automatically run your test suite!

Testing Framework: testthat

- A testing framework for R that easily integrates with RStudio
- *testthat* hierarchy: Expectations, Tests and Contexs.
- **Expectation**: Describes what the result of a computation should be.
  - Finest level of testing
  - Binary assertion about whether or not a value is as you expect
  - 11 types of built-in expectations
- **Test**: Groups together multiple expectations to test e.g. one function
- **Context**: Groups multiple tests that test related functionality

Testing Framework: expectations

Full	Short cut
`expect_that(x, is_true())`	`expect_true(x)`
`expect_that(x, is_false())`	`expect_false(x)`
`expect_that(x, is_a(y))`	`expect_is(x, y)`
`expect_that(x, equals(y))`	`expect_equal(x, y)`
`expect_that(x, is_equivalent_to(y))`	`expect_equivalent(x, y)`
`expect_that(x, is_identical_to(y))`	`expect_identical(x, y)`
`expect_that(x, matches(y))`	`expect_matches(x, y)`
`expect_that(x, prints_text(y))`	`expect_output(x, y)`
`expect_that(x, shows_message(y))`	`expect_message(x, y)`
`expect_that(x, gives_warning(y))`	`expect_warning(x, y)`
`expect_that(x, throws_error(y))`	`expect_error(x, y)`

Table 1: Expectation shortcuts

My test cases (I)

```
context("linreg")
data("iris")

#Testing for erroneous input in linreg_ols function
test_that("linreg_ols rejects errounous input", {
  expect_error(linreg_ols(Petal.Length~Sepdsal.Width+Sepal.Length, data=iris))
  expect_error(linreg_ols(Petal.Length~Sepdsal.Width+Sepal.Length, data=irfsfdis))
  expect_error(linreg_ols("this is not a formula!", data=iris))
})

#Testing that the returned object is from the correct class
test_that("class is correct", {
  linreg_mod1 <- linreg_ols(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_s3_class(linreg_mod1, "linreg")
  linreg_mod2 <- linreg_bayes(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_s3_class(linreg_mod2, "linreg")
})
```

My test cases (II)

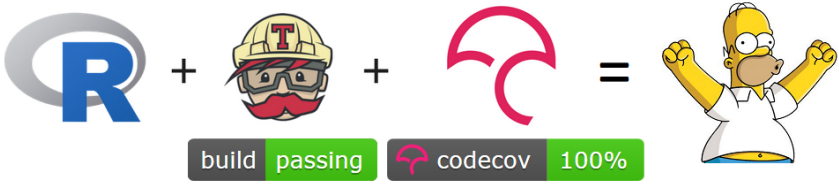
```
#Testing that the residuals from regression are correct
test_that("resid() works", {
  linreg_mod1 <- linreg_ols(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_equal(round(unname(resid(linreg_mod1)[c(7,13,27)]),2), c(0.31, -0.58, -0.20))
  linreg_mod2 <- linreg_bayes(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_equal(round(unname(resid(linreg_mod2)[c(7,13,27)]),2), c(0.31, -0.58, -0.20))
})

#Testing that we get correct regression estimated coefficients
test_that("coef() works", {
  linreg_mod1 <- linreg_ols(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_true(all(round(unname(coef(linreg_mod1)),2) %in% c(-2.52, -1.34, 1.78)))
  linreg_mod2 <- linreg_bayes(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_true(all(round(unname(coef(linreg_mod2)),2) %in% c(-2.52, -1.34, 1.78)))
})
```

My test cases (III)

```
#Testing that the output for the summary function is correct
test_that("summary() works", {
  linreg_mod1 <- linreg_ols(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_output(summary(linreg_mod1), ".*\\(Intercept\\).*0\\.31.*-4\\.48.*0\\.00.*")
  expect_output(summary(linreg_mod1), ".*Sepal\\.Width.*0\\.014.*-10\\.94.*0\\.00.*")
  expect_output(summary(linreg_mod1), ".*Sepal\\.Length.*0\\.004.*27\\.56.*0\\.00.*")
  linreg_mod2 <- linreg_bayes(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_output(summary(linreg_mod2), ".*\\(Intercept\\).*1\\.6.*0\\.0.*0\\.00.*")
  expect_output(summary(linreg_mod2), ".*Sepal\\.Width.*0\\.07.*0\\.00.*0\\.00.*")
  expect_output(summary(linreg_mod2), ".*Sepal\\.Length.*0\\.02.*0\\.00.*0\\.00.*")
})
#Testing that the plot function actually returns a plot
test_that("plot() works",{
  linreg_mod1 <- linreg_ols(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_is(plot(linreg_mod1),"gtable")
  linreg_mod2 <- linreg_bayes(Petal.Length~Sepal.Width+Sepal.Length, data=iris)
  expect_is(plot(linreg_mod2),"gtable")
})
```

Testing the Tests

- I used Codecov (https://codecov.io) for testing my test suite
- Automatically checks the coverage of your test suite (% of lines)
- Free for open-source projects and easy to integrate with Travis-CI



Some final thoughts

- Statisticians have not been user-oriented historically
  - We usually worry about reviewers not users!
- Tide may be changing however…
  - Number of package contributions to CRAN is rocketing
  - Dissemination of your research/models is paramount
  - Packaging your model with RStudio and Git is easy now
  - Testing your code naturally becomes part of this process
- Good news for the STATS/ML community in general
  - More models available for research and benchmarking

Questions?

hector.rodriguez@liu.se

LINKÖPING UNIVERSITY

WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM