732A94 Advanced R Programming
Computer lab 4

Krzysztof Bartoszek, Héctor Rodriguez Déniz
(designed by Leif Jonsson and Måns Magnusson)

10 September 2019

Seminar date: **24 September 15:15** (C3)
Lab deadline: **29 September 23:59**

Instructions

- This lab should be conducted by students **two by two**.
- The lab constists of writing a package that is version controlled on github.com.
- Both student should **contribute equally much** to the package.
- Other significant collaborations/discussions should be acknowledged in the solution.
- Commit continously your addition and changes.
- Collaborations should be done using github (ie you should commit using your own github account).
- In the lab some functions can be marked with an *. Students **MUST do AT LEAST ONE** exercise marked with an * (if more than one is marked) for each of the Labs 3 - 6 and Bonus.
- The deadline for the lab is on the lab's title page.
- The lab should be turned in as an url to the repository containing the package on github using **LISAM**. This should also include name, github user names and liuid of the students behind the project. In case of problems the url may be emailed to `hector.rodriguez@liu.se` or `krzysztof.bartoszek@liu.se`.
- **NO resubmissions will be allowed for the Bonus lab.**
  NO late submissions will be allowed for the Bonus lab.

Contents

Chapter 1

A linear regression package in R

In this lab we will will create a package to handle linear regression models. We will use linear algebra to create the most basic functionality in the R package. We will also implement an object oriented system to handle special functions such as `print()`, `plot()`, `resid()`, `pred()`, `coef()` and `summary()`. We will finish the package up by writing a vignette on how the package can be used to conduct a simple regression analysis using a dataset included in the package.

There are different design choices to make with different levels of complexity.

1. Implement the calculations using ordinary linear algebra or (*) using the QR decomposition
2. Implement the results as an S3 class or (*) a RC class object
3. (*) Implement a `theme()` for the graphical profile of Linköping University

Students that take the advanced course must choose to use one of the advanced implementations (ie either using QR decomposition or using RC classes).

Remember to commit your changes continously to github during the lab.

1.1 Create a package on Github and Travis

Start out by creating your new package on github and Travis CI. See lab 3 for details.

1.2 Write the R code

In this R package we will write the code for a multiple regression model. The function should be called `linreg()` and have the two arguments formula and data. The function should return an object with of class `linreg` either as an S3 class or an RC class.

The formula argument should take a formula object. The first step in the function is to use the function `model.matrix()` to create the matrix $\mathbf{X}$ (independent variables) and the pick out the dependent variable $\mathbf{y}$ using `all.vars()`.

1.2.1 Computations using ordinary least squares

The simple way to calculate the different is to use ordinary linear algebra and calculate:

Regressions coefficients:

$$\hat{\beta} = \left(\mathbf{X}^{\mathbf{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathbf{T}}\mathbf{y}$$

The fitted values:

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$$

The residuals:

$$\hat{\mathbf{e}} = \mathbf{y} - \hat{\mathbf{y}} = \mathbf{y} - \mathbf{X}\hat{\beta}$$

The degrees of freedom:

$$df = n - p$$

The residual variance:

$$\hat{\sigma}^2 = \frac{\mathbf{e}^{\mathbf{T}}\mathbf{e}}{df}$$

where n is the number of observations and p is the number of parameters in the model.

The variance of the regression coefficients:

$$\hat{\mathbf{Var}}(\hat{\beta}) = \hat{\sigma}^2(\mathbf{X}^{\mathbf{T}}\mathbf{X})^{-1}$$

The t-values for each coefficient:

$$t_\beta = \frac{\hat{\beta}}{\sqrt{Var(\hat{\beta})}}$$

Using the function `pt()` it is then possible to calculate the p-values for each regression coefficient.

Calculate these statistics and store it in an object of class `linreg`. Document your function `linreg()` using `roxygen2`.

1.2.2 (*) Using the QR decomposition

In statistical software we use the QR decomposition to do the estimation. This is due to the fact that QR decomposition is faster and more robust than the more straight-forward way of the linear algebra computations.

Your job is to compute the $\hat{\beta}$ and $\text{Var}(\hat{\beta})$ using a QR decomposition of $\mathbf{X}$. See extra materials in LISAM, from here `https://www.stat.wisc.edu/courses/st849-bates/lectures/Orthogonal.pdf` and here `http://staff.www.ltu.se/~jove/courses/c0002m/least_squares.pdf` for detailed information how to estimate $\hat{\beta}$. You may have to derive yourself how to use the QR decomposition to calculate $\text{Var}(\hat{\beta})$. Remember that Q is an orthogonal matrix, and R is a triangular matrix.

1.3 Implementing methods for your class

You now have done the calculations and stored them in an object with class `linreg`. The next step is to implement different "methods" for your object. These implementations can be done in either the S3 object oriented system or (*) the RC objected oriented system.

The following methods should be implemented and be documented using `roxygen2`.

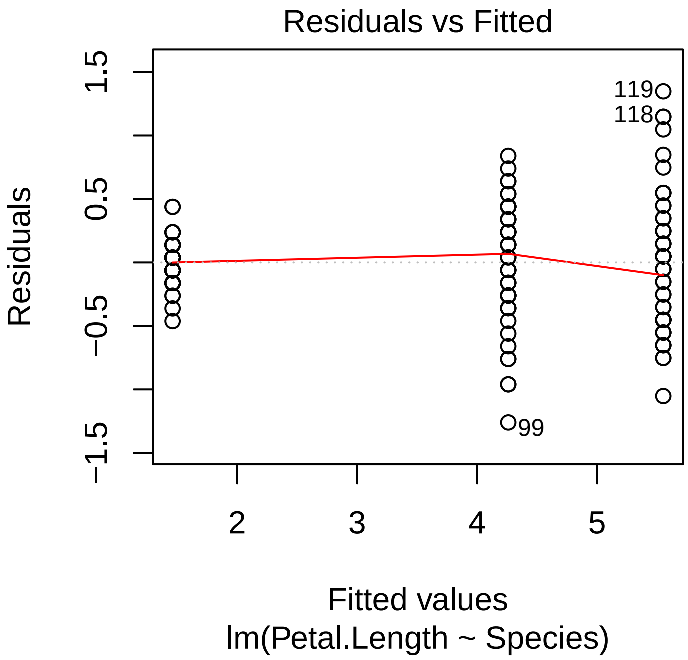
print() should **print out** the coefficients and coefficient names, similar as done by the lm class.

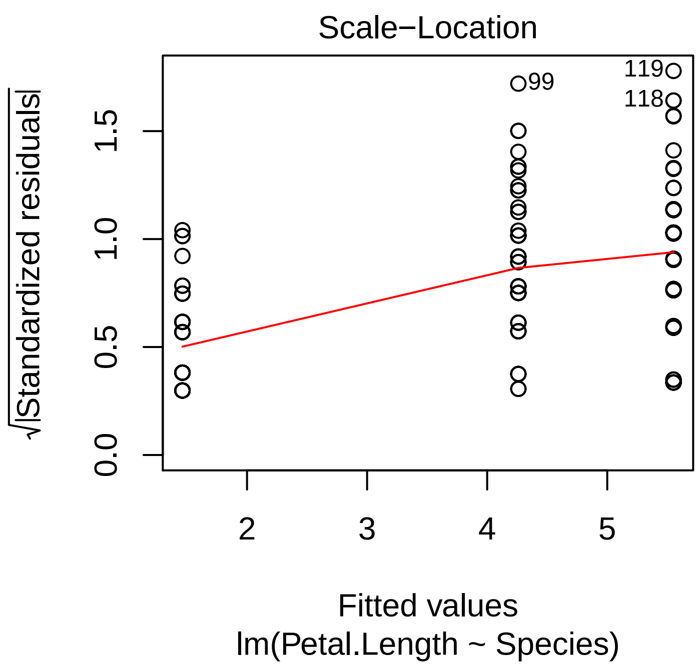
```
data(iris)
mod_object <- lm(Petal.Length~Species, data = iris)
print(mod_object)


Call:
lm(formula = Petal.Length ~ Species, data = iris)

Coefficients:
      (Intercept)  Speciesversicolor   Speciesvirginica
             1.46               2.80               4.09
```

plot() should plot the following two plots using `ggplot2`. Remember to include `ggplot2` in your package.





resid() should return the vector of residuals $\mathbf{e}$.

pred() should return the predicted values $\hat{\mathbf{y}}$

coef() should return the coefficients as a **named** vector.

summary() should return a similar printout as printed for lm objects, but you only need to present the coefficients with their standard error, t-value and p-value as well as the estimate of $\hat{\sigma}$ and the degrees of freedom in the model.

1.4 Add unit test suites to your package

Add unit tests found at `https://github.com/STIMALiU/AdvRCourse/tree/master/Testsuites` for your package depending on the OO system you are using. See lab 3 for details.

1.5 Write a vignette on how to use your package

Write a vignette that describes your function and how to use it on a well known dataset such as iris or faithful. A vignette should be included and be possible to read using the `browseVignettes(‘‘[yourpackagenamehere]’’)` command and show a simple walkthrough of your package showing how to use the function linreg together with all the implemented methods. For more information on how to write a vignette and include them in your package see “Vignettes” in [1].

1.6 (*) Create a `theme()` for Linköping university

It is common that different companies have a graphical profile and that all graphs created should follow this graphical profile. To solve this problem using `ggplot2` we have what is called `theme()` that containes all graphical information and that you simply “add” to your ggplot to follow the graphical profile.

Create a `theme()` for Linköping university. The guidelines can be found in the pdf **here** `https://liu.se/insidan/kommunikationsstod/grafiskprofil?l=en`. You can implement as much as you like (adding logotype, typesetting etc.) but as a minimum the colors should be used that are the colors mentioned in the graphical manual.

1.7 Seminar and examination

During the seminar you will bring your own computer and demonstrate your package and what you found difficult in the project.

We will present as many packages as possible during the seminar and you should

1. Show that the package can be built using R Studio and that all unit tests is passing.
2. Present the unit tests you’ve written.
3. Show your vignette/run the examples live.

1.7.1 Examination

Turn in a the adress to your github repo with the package using LISAM. To pass the lab you need to:

1. Have the R package up on GitHub with a Travis CI pass/fail badge.
2. The test suites for the implemented function(s) should be included in the package.
3. The package should build without warnings (pass) on Travis CI.
4. All issues raised by Travis CI should be taken care or justified why they are not a problem or cannot be corrected. Be careful with namespace issues, these you HAVE to take care of.

Bibliography

[1] Hadley Wickham. *R packages*. " O'Reilly Media, Inc.", 2015.