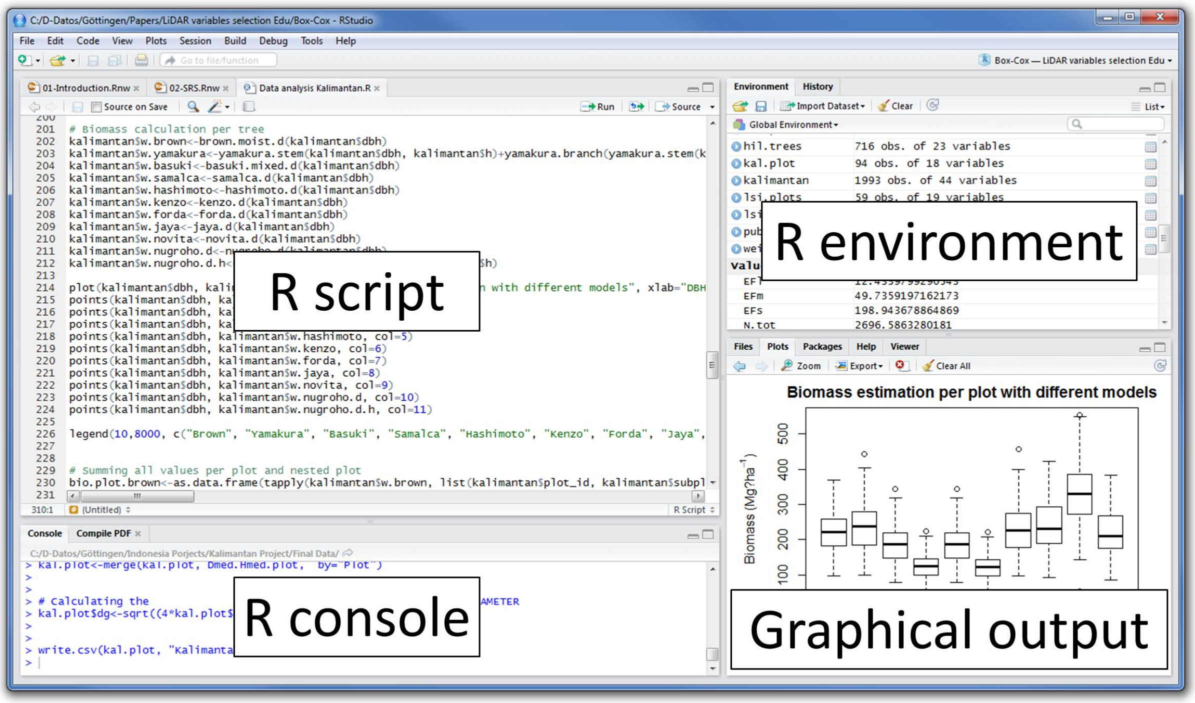
post01-Rui-Sun

Rui Sun

2017/10/31

Experience of using language R as an novice

Introduction


C:/D-Datos/Göttingen/Papers/LiDAR variables selection Edu/Box-Cox - RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function
Box-Cox — LiDAR variables selection Edu
01-Introduction.Rnw
02-SRS.Rnw
Data analysis Kalimantan.R
Source on Save
Run
Source
310:1
(Untitled)
R Script
Console
Compile PDF
C:/D-Datos/Göttingen/Indonesia Porjects/Kalimantan Project/Final Data/
Environment
History
Import Dataset
Clear
List
Global Environment
Files
Plots
Packages
Help
Viewer
Zoom
Export
Clear All
Biomass estimation per plot with different models
Biomass (Mg?ha^{-1})
100
200
300
400
500
R script
R environment
R console
Graphical output

```
# Biomass calculation per tree
kalimantan$w.brown<-brown.moist.d(kalimantan$dbh)
kalimantan$w.yamakura<-yamakura.stem(kalimantan$dbh, kalimantan$h)+yamakura.branch(yamakura.stem(k
kalimantan$w.basuki<-basuki.mixed.d(kalimantan$dbh)
kalimantan$w.samalca<-samalca.d(kalimantan$dbh)
kalimantan$w.hashimoto<-hashimoto.d(kalimantan$dbh)
kalimantan$w.kenzo<-kenzo.d(kalimantan$dbh)
kalimantan$w.forda<-forda.d(kalimantan$dbh)
kalimantan$w.jaya<-jaya.d(kalimantan$dbh)
kalimantan$w.novita<-novita.d(kalimantan$dbh)
kalimantan$w.nugroho.d<-nu
kalimantan$w.nugroho.d.h<                                 $h)

plot(kalimantan$dbh, kali                                 n with different models", xlab="DBH
points(kalimantan$dbh, ka
points(kalimantan$dbh, ka
points(kalimantan$dbh, ka
points(kalimantan$dbh, kalimantan$w.hashimoto, col=5)
points(kalimantan$dbh, kalimantan$w.kenzo, col=6)
points(kalimantan$dbh, kalimantan$w.forda, col=7)
points(kalimantan$dbh, kalimantan$w.jaya, col=8)
points(kalimantan$dbh, kalimantan$w.novita, col=9)
points(kalimantan$dbh, kalimantan$w.nugroho.d, col=10)
points(kalimantan$dbh, kalimantan$w.nugroho.d.h, col=11)

legend(10,8000, c("Brown", "Yamakura", "Basuki", "Samalca", "Hashimoto", "Kenzo", "Forda", "Jaya",


# Summing all values per plot and nested plot
bio.plot.brown<-as.data.frame(tapply(kalimantan$w.brown, list(kalimantan$plot_id, kalimantan$subpl
```

```
> kal.plot<-merge(kal.plot, Dmed.Hmed.plot,  by="Plot")
>
>
> # Calculating the                                AMETER
> kal.plot$dg<-sqrt((4*kal.plot$
>
>
> write.csv(kal.plot, "Kalimanta
> 
```

```
hil.trees       716 obs. of 23 variables
kal.plot        94 obs. of 18 variables
kalimantan      1993 obs. of 44 variables
lsi.plots       59 obs. of 19 variables
lsi
pub
wei
Valu
  EFl
  EFm           49.7359197162173
  EFs           198.943678864869
  N.tot         2696.5863280181
```

Ross Ihaka

R as some of you may know is a statistical analysis system created by Ross Ihaka and Robert Gentleman ("R language definition", 2000) whose features are: * effective data processing and simple tools for saving results * a set of operators for processing arrays, matrices, and other complex constructions * large, consistent, integrated collection tools for statistical analysis * numerous graphic tools * a simple and effective programming language that includes a lot of possibilities

I have been fortunate to work with such a programming language, even though I haven't had any prior experience working with such a programming language and moreover with Data Scientists. Since I decided to major in Statistics, I started to learn it and I found that R is powerful and convenient for data analysing and data visualizing. After last weeks learing and practicing on Rstudio, I have some experience and opinions whcih are related to some extended reading about using Language R.

R can easily load and save data

In the begining, I went through a few hours of tutoring by getting an understanding of the R environment, how to install it, and an overview of RStudio and I realized how amazing it is! What fascinates me is that you can esaily import a local file and store the data into one data variable. For example, the syntax would be:

```
mydata <- read.csv("nba2017-player-statistics.csv")
```

Furthermore, after you load objects into memory and play with it and when you shutdown your environment your data is not cleared! Rather you can save it and it retains such information per project.

R is a language with many functions

The another reason that I like to use R is that there is a wide range of functions is available in the base package. Since I am taking stat134 this semester, I found that there are some concepts and model of basic distributions can be simulated in R by various functions. There is also a large number of other packages that increase the potential of R. They are located separately and must be loaded into memory. A comprehensive list of such packages, along with their descriptions, can be found on the Internet at: URL: http://cran.rproject.org/src/contrib/PACKAGES.html.

Those packages are useful in many different uses: * To install package For example:

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(Rcpp)
```

```
## Warning: package 'Rcpp' was built under R version 3.4.2
```

- To load data For example:

```
mydata <- read.csv("nba2017-player-statistics.csv")
```

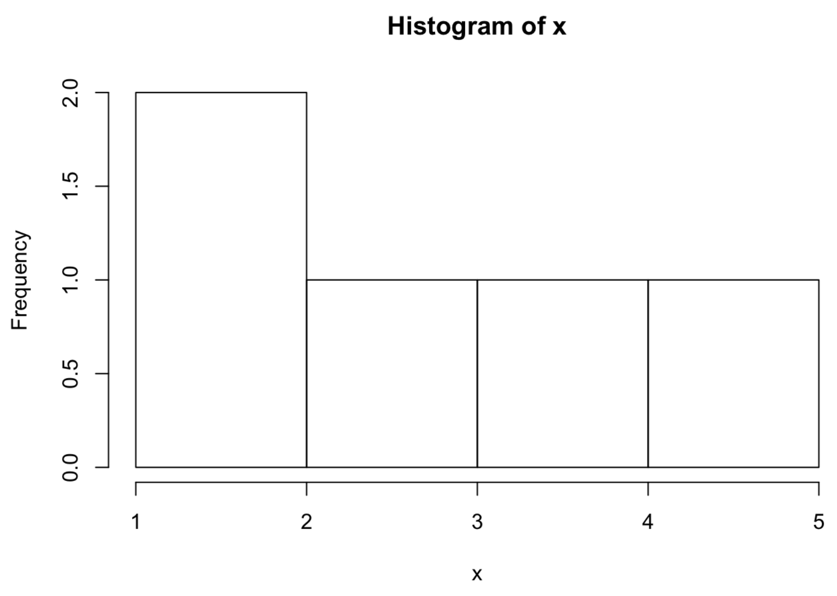
- To manipulate data For example:

```
str <- str(mydata)
```

```
## 'data.frame':    441 obs. of  24 variables:
##  $ Player      : Factor w/ 441 levels "A.J. Hammons",..: 5 16 30 103 146 161 169 181 189 213 ...
##  $ Team        : Factor w/ 30 levels "ATL","BOS","BRK",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Position    : Factor w/ 5 levels "C","PF","PG",..: 1 2 5 3 4 3 4 5 4 2 ...
##  $ Experience  : Factor w/ 19 levels "1","10","11",..: 18 3 15 19 18 14 13 11 19 15 ...
##  $ Salary      : num  26540100 12000000 8269663 1450000 1410598 ...
##  $ Rank        : int  4 6 5 15 11 1 3 13 8 10 ...
##  $ Age         : int  30 29 26 22 31 27 26 21 20 29 ...
##  $ GP          : int  68 80 55 5 47 76 72 29 78 78 ...
##  $ GS          : int  68 77 55 0 0 76 72 0 20 6 ...
##  $ MIN         : int  2193 1608 1835 17 538 2569 2335 220 1341 1232 ...
##  $ FGM         : int  379 213 359 3 95 682 333 25 192 114 ...
##  $ FGA         : int  801 370 775 4 232 1473 720 58 423 262 ...
##  $ Points3     : int  86 27 108 1 39 245 157 12 46 45 ...
##  $ Points3_atts: int  242 66 277 1 111 646 394 35 135 130 ...
##  $ Points2     : int  293 186 251 2 56 437 176 13 146 69 ...
##  $ Points2_atts: int  559 304 498 3 121 827 326 23 288 132 ...
##  $ FTM         : int  108 67 68 3 33 590 176 6 85 26 ...
##  $ FTA         : int  135 100 93 6 41 649 217 9 124 37 ...
##  $ OREB        : int  95 117 65 2 17 43 48 6 45 60 ...
##  $ DREB        : int  369 248 269 2 68 162 367 20 175 213 ...
##  $ AST         : int  337 140 121 3 33 449 155 4 64 71 ...
##  $ STL         : int  52 52 68 0 9 70 72 10 35 26 ...
##  $ BLK         : int  87 62 11 0 7 13 23 2 18 17 ...
##  $ TO          : int  116 77 88 0 25 210 79 4 68 39 ...
```

- To visualize data For example:

```
x <- 1 : 5
hist(x)
```



There is a bunch of packages that we have not cover but they are still useful in the folllowing uses: * To model data For example:

```
x <-1:5
y <-rnorm (5)
lm(y~x)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Coefficients:
## (Intercept)            x
##     -1.2716       0.1815
```

- To report results For example:

```
mymodel <- lm(y~x)
summary(mymodel)
```

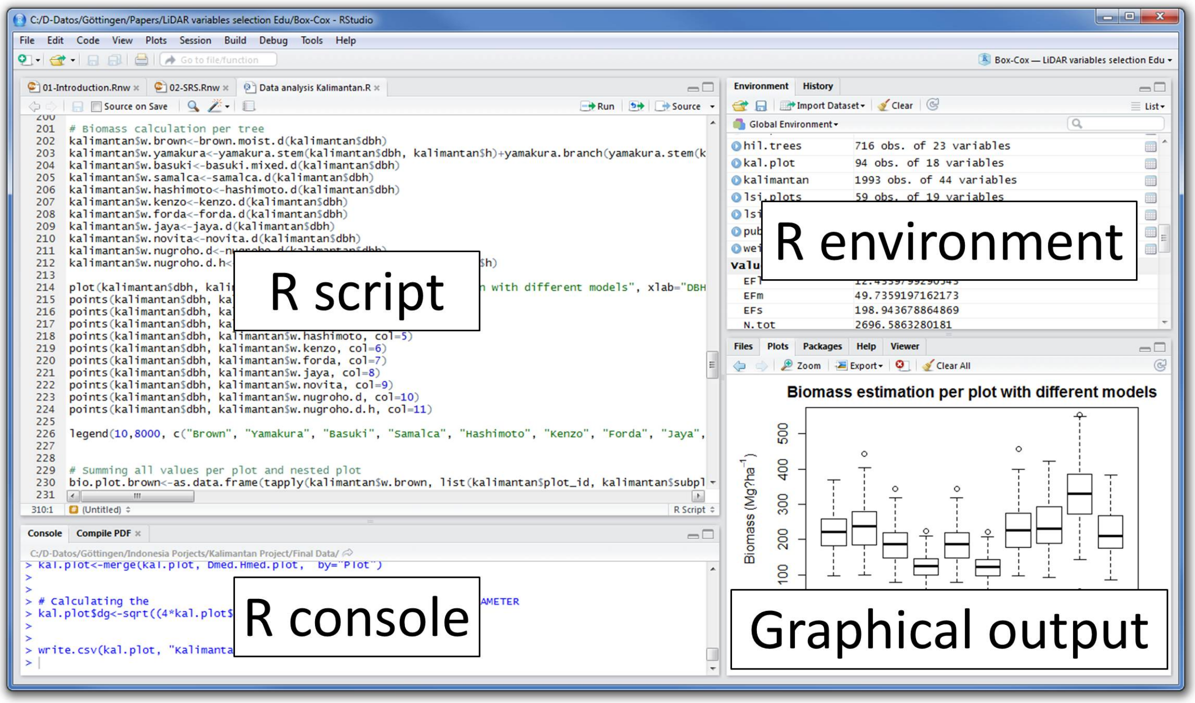
```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##        1        2        3        4        5
##  0.02763 -0.19558 -0.44810  1.37245 -0.75639
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.2716     0.9942  -1.279    0.291
## x             0.1815     0.2998   0.605    0.588
##
## Residual standard error: 0.9479 on 3 degrees of freedom
## Multiple R-squared:  0.1089, Adjusted R-squared:  -0.1882
## F-statistic: 0.3664 on 1 and 3 DF,  p-value: 0.5877
```

- To write high performance R code

```
cppFunction('int add(int x, int y, int z) {
  int sum = x + y + z;
  return sum;
}')
```

- To work with the web
- To write your own R packages R performs a wide variety of functions, such as data manipulation, statistical modeling, and graphics. One really big advantage of R, however, is its extensibility. People can easily write their own software and distribute it in the form of add-on packages. Because of the relative ease of creating these packages, literally thousands of them exist. In fact, many new statistical methods are published with an R package attached.
- Etc

Rstudio is easy to use


C:/D-Datos/Göttingen/Papers/LiDAR variables selection Edu/Box-Cox - RStudio
File Edit Code View Plots Session Build Debug Tools Help
Go to file/function
01-Introduction.Rnw
02-SRS.Rnw
Data analysis Kalimantan.R
Source on Save
Run
Source
R script
Environment
History
Import Dataset
Clear
List
Global Environment
R environment
Box-Cox — LiDAR variables selection Edu
Files
Plots
Packages
Help
Viewer
Zoom
Export
Clear All
Biomass estimation per plot with different models
Biomass (Mg?ha^{-1})
100
200
300
400
500
Graphical output
Console
Compile PDF
C:/D-Datos/Göttingen/Indonesia Porjects/Kalimantan Project/Final Data/
R console
310:1
(Untitled)
R Script

```
# Biomass calculation per tree
kalimantan$w.brown<-brown.moist.d(kalimantan$dbh)
kalimantan$w.yamakura<-yamakura.stem(kalimantan$dbh, kalimantan$h)+yamakura.branch(yamakura.stem(k
kalimantan$w.basuki<-basuki.mixed.d(kalimantan$dbh)
kalimantan$w.samalca<-samalca.d(kalimantan$dbh)
kalimantan$w.hashimoto<-hashimoto.d(kalimantan$dbh)
kalimantan$w.kenzo<-kenzo.d(kalimantan$dbh)
kalimantan$w.forda<-forda.d(kalimantan$dbh)
kalimantan$w.jaya<-jaya.d(kalimantan$dbh)
kalimantan$w.novita<-novita.d(kalimantan$dbh)
kalimantan$w.nugroho.d<-n
kalimantan$w.nugroho.d.h<                                    $h)

plot(kalimantan$dbh, kali                              n with different models", xlab="DBH
points(kalimantan$dbh, ka
points(kalimantan$dbh, ka
points(kalimantan$dbh, ka
points(kalimantan$dbh, kalimantan$w.hashimoto, col=5)
points(kalimantan$dbh, kalimantan$w.kenzo, col=6)
points(kalimantan$dbh, kalimantan$w.forda, col=7)
points(kalimantan$dbh, kalimantan$w.jaya, col=8)
points(kalimantan$dbh, kalimantan$w.novita, col=9)
points(kalimantan$dbh, kalimantan$w.nugroho.d, col=10)
points(kalimantan$dbh, kalimantan$w.nugroho.d.h, col=11)

legend(10,8000, c("Brown", "Yamakura", "Basuki", "Samalca", "Hashimoto", "Kenzo", "Forda", "Jaya",


# Summing all values per plot and nested plot
bio.plot.brown<-as.data.frame(tapply(kalimantan$w.brown, list(kalimantan$plot_id, kalimantan$subpl
```

```
hil.trees      716 obs. of 23 variables
kal.plot       94 obs. of 18 variables
kalimantan     1993 obs. of 44 variables
lsi.plots      59 obs. of 19 variables
lsi
pub
wei
Valu
EFl
EFm            49.7359197162173
EFs            198.943678864869
N.tot          2696.5863280181
```

```
> kal.plot<-merge(kal.plot, Dmed.Hmed.plot,  by="Plot")
>
>
> # Calculating the                          AMETER
> kal.plot$dg<-sqrt((4*kal.plot$
>
>
> write.csv(kal.plot, "Kalimanta
> |
```

Rstudio GUI

When I first started to learn to use R, I was bound and determined to use the basic R GUI. Without knowing much about it, I also didn't want to use RStudio. It seemed overly complicated to download an additional software package for something that already functioned on its own. One day though, I found my thought was wrong because Rstudio is easy and convient to use.

- It is easy to write scipts As soon as you create a new script, the windows within your RStudio session adjust automatically so you can see both your script and the results in your console when you run your syntax.

Even better is the ability to call up potential syntax options while you are writing just by using the tab key.

- RStudio makes it convenient to view and interact with the objects stored in your environment. In the basic R GUI, you can always list the objects you have stored in your environment. But RStudio has a very useful “Environment” window available.

This shows all of the objects that you have stored, including data; scalars, vectors, and matrices; model outputs; etc., along with a summary of the information that is stored in those objects.

You can even click on your data sets directly to open them and view them as spreadsheets.

- RStudio makes it easy to set your working directory and access files on your computer. Especially if you are working in Windows, one of the most tedious parts of programming in R is setting your working directory to access your files.

With RStudio, you can navigate to folders on your computer in the “Files” window, view any files you have in that folder, and set that folder as the working directory.

Summary(Take-home message)

Since I am just a beginner of using command-based software, I cannot compare R language with other softwares. However, after learning and practicing R on Rstudio in past weeks, I found that R is a very useful method of data analysising because it can easily load and save date, has lots of useful functions and its GUI is easy and convient to use.

Works Cited

Ihaka, Ross, and Robert Gentleman. “R: A Language For Data Analysis And Graphics.” Journal Of Computational And Graphical Statistics, vol 5, no. 3, 1996, pp. 299-314. Informa UK Limited, doi:10.1080/10618600.1996.10474713. “Mean In R.” Youtube, 2017, https://www.youtube.com/watch?v=MeDxJixwaPs. Team, R. Core. “R language definition.” Vienna, Austria: R foundation for statistical computing (2000). “Rproject.Org.” Cran.Rproject.Org, 2017, http://cran.rproject.org/doc/FAQ/R-FAQ.html. “Rproject.Org.” Cran.Rproject.Org, 2017, http://cran.rproject.org/src/contrib/PACKAGES.html. “R - Mean, Variance And Much More.” Youtube, 2017, https://www.youtube.com/watch?v=Fl0Ygu9_bpA. “Central Tendency In R: Mean, Median, And Mode.” Youtube, 2017, https://www.youtube.com/watch?v=UEBzPCT9oGA.