Biometry. Lecture 7

Alexey Shipunov

Minot State University

February 10, 2016

Outline

1. Questions and answers
2. Basics of R
   - The basics of R graphics
3. Types of data
   - Measurement (interval) data
   - Ranked (ordinal) data
   - Nominal (categorical) data

Outline

1. Questions and answers
2. Basics of R
   - The basics of R graphics
3. Types of data
   - Measurement (interval) data
   - Ranked (ordinal) data
   - Nominal (categorical) data

Outline

1. Questions and answers
2. Basics of R
   - The basics of R graphics
3. Types of data
   - Measurement (interval) data
   - Ranked (ordinal) data
   - Nominal (categorical) data

Starting...

`> setwd("<working folder>")`

or

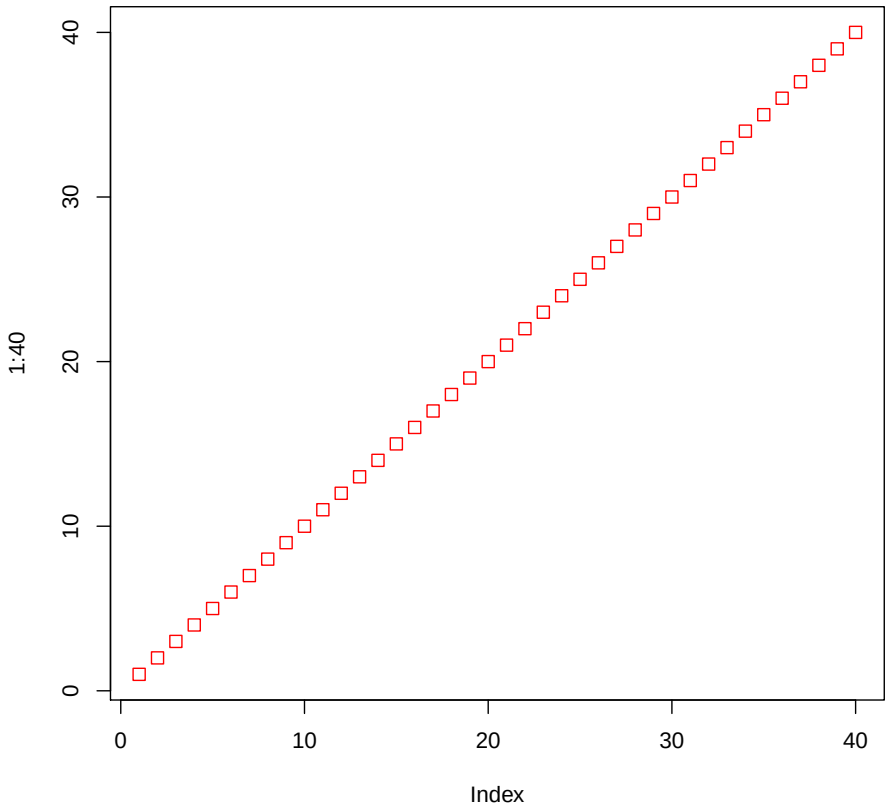
“`Change dir`”

in menu!

On Mac, be sure that startup option is working: `getwd()` (`getwd()` checks if R is in working folder, `dir()` checks the folder content)

Previous last question

Which command will produce this plot?




```
plot(1:40, pch=0, col=2)
```

Enhance the plot from Lab 3

```
> scatter.smooth() # plots the dots and line
> log10() # logarithmic transformation
```

Basics of R

The basics of R graphics

How to save current plot into the file

```
> pdf("1.pdf")
> plot(1:20)
> dev.off() # until you run it, your PDF does not exist!
> # OR (less appropriate)
> plot(1:20)
> dev.copy(pdf, "1.pdf") # works only interactively!
> dev.off() # always close device!
```

Graphical options

```
> oldpar <- par(mfrow=c(2,1))
> hist(cars$speed)
> hist(cars$dist)
> par(oldpar)
```

`mfrow` by default is `c(1,1)`

`par()` should be kept in the object and then restored

Interactive graphics

```
> plot(1:20)
> text(locator(), "My beloved point", pos=4)
```

Click left mouse button, then right mouse button

Interactive graphics

```
> i <- read.table("http://ashipunov.info/data/islets.txt",
+ sep="\t", h=T)
> square <- i$width * i$length * .6 * .6
> scatter.smooth(log10(square), i$species)
> identify(log10(square), i$species)
```

Types of data

Measurement (interval) data

Measurement (interval) data

- For any two measurements, the third between them also has sense
- Best example: location on the ruler. Continuous, could be zero, positive and negative.
- Temperature has a restriction: there is a minimal temperature
- Angle is worse: there are both minimal and maximal angles

Discrete measurement data: counts

- Tis is the other kind of measurement data
- Number of items is always a whole number so there is the third between 2 and 4
- But the third number between 2 and 3 is a nonsense

“Parametric” and “non-parametric” data

- (a) Only *continuous measurement* data may be parametric
- In addition, parametric methods require: (b) suspected *normal distribution* of data and (c) sample ≥ 30
- Everything else should be studied with non-parametric methods

Measurement data in R

```
> x <- c(174, 162, 188, 192, 165, 168, 172)
> str(x)
 num [1:7] 174 162 188 192 165 168 172
> is.numeric(x)
 [1] TRUE
> is.vector(x)
 [1] TRUE
```

Types of data

Ranked (ordinal) data

What if we cannot measure?

- In this case, we cal use scale-like representation
- E.g., we can rank the student success from 1 to 5 (“very bad” to “excellent”)
- Or softness of mattress from 0 to 10 (“hard as a plank” to “soft as a cloud”)

Ranked and measurement data

- Similarity: for every two ranks, the third between them has sense
- E.g., it is possible to imagine mattress with softness between 2 and 3
- However, ranks are not represent intervals correctly!
- Ranked data should be studied with non-parametric methods

How to create ranked data

In R, ranked data is normally represented by the same numerical vector or *ordered factor*. Command `cut()` will break continuous data into ranks:

```
> height <- trees[,2]
> cut(height, 3, labels=c(1:3), ordered=T)
> cut(height, 3, ordered=T)
```

Types of data

Nominal (categorical) data

Just observations

- Some data cannot be ordered at all
- Sex, color, absence/presence are good examples
- If even we label red color as “1” and green color as “2” the “1.5” is a nonsense.
- Therefore, if we use numbers for categorical data, they are only *labels*.

Binary data

- Absence/presence is a specific subset of categorical data which only two possible values
- One of the easiest representation is with numbers 0 and 1
- Computers normally prefer binary data over non-binary

Logical data

Practically, it is just a kind of binary data:

```
> height < 72
> height >= 72
> height == 72 # not "="!
> presence <- c(F, T, T, F, F)
> presence
> presence * 1 # convert to 1/0
> (presence * 1) == 1 # convert back
```

"==" is a logical test: "Is equal?". In R, "=" has a different meaning, it is a replacement for "<−".

Categorical data in R

Character and logical vectors may be used for categorical data:

```
> sex <- c("male", "female", "male", "male",
+ "female", "male", "male")
> is.character(sex)
> is.vector(sex)
> str(sex)
> str(presence)
```

Squeezing numbers from the categorical data

```
> sex <- c("male", "female", "male", "male",
+ "female", "male", "male")
> presence <- c(F, T, T, F, F)
> table(sex)
> table(presence)
```

The `table()` command will let us to have some numbers even from categorical data!

Character to factor

```
> plot(sex) # error!
> sex.f <- as.factor(sex)
> plot(sex.f) # makes bar plot
```

Factor is a special type of object used to represent nominal or ranked data

Features of factors

```
> is.factor(sex.f)
> is.character(sex.f)
> str(sex.f)
> levels(sex.f)
> sex.f[6:7] # two levels!
> sex.f[6:7, drop=TRUE] # one level
```

Factor has levels which will not automatically drop with a sub-setting.

Factors to numbers

```
> as.numeric(sex.f)
> w <- c(69, 68, 93, 87, 59, 82, 72)
> x <- c(174, 162, 188, 192, 165, 168, 172)
> plot(x, w, pch=as.numeric(sex.f), col=as.numeric(sex.f))
> legend("topleft", pch=1:2, col=1:2, legend=levels(sex.f))
```

Objects `x`, `sex` and `w` could be height, gender and weight of seven people in small office, respectively.

Factors to ranks

```
> m <- c("L", "S", "XL", "XXL", "S", "M", "L") # t-shirts
> m.f <- factor(m)
> levels(m) # Wrong order, alphabetical
> m.o <- ordered(m.f, levels=c("S", "M", "L", "XL", "XXL"))
> levels(m.o)
```

The danger of factors

```
> a <- factor(3:5)
> a
> as.numeric(a) # wrong!!!
> as.numeric(as.character(a)) # correct
```

Some rules about vectors

- For every type of R object, there are functions `is.<something>()` and `as.<something>()` (e.g., `as.vector()` and `as.numeric()` will convert to vector and to numeric vector, respectively).
- Object names must not start with a number
- R is case-sensitive
- Please avoid to use names of popular functions (like `c()`) and reserved keywords: `T` (TRUE), `F` (FALSE), `NA` (missing data), `NaN` (not a number), `Inf` (result of dividing by zero), also constants like `pi`, `letters` and `LETTERS`

If you want *e* constant, use `exp(1)`

Finishing...

Save your commands!

(`savehistory(<todaysdate>.r)` or `File -> Save as...` on Mac)

Summary: most important commands

- `par()` regulates plots parameters
- `cut()` makes ranked data
- `as.<something>()`—converts objects
- `table()`—summarizes categorical data

For Further Reading

A. Shipunov.
Biometry [Electronic resource].
2012—onwards.
Mode of access:
http://ashipunov.info/shipunov/school/biol_240

A. Shipunov, and many others.
Visual statistics. Use R!
2016—onwards.
Mode of access: http://ashipunov.info/shipunov/school/biol_240/en/visual_statistics.pdf