Topic 4

Exploratory Data Analysis with R: Graphics

Sergey Mastitsky ©
Klaipeda, 28-30 September 2011

4. EDA with R: Graphics

4.1. A general protocol for EDA

Zuur et al. (2010) A protocol for data exploration to avoid common statistical problems. *Methods in Ecology & Evolution* 1: 3-14

An enormous expansion of statistical tools has occurred over the last 30 years…

- Linear regression and ANOVA-like analyses
- Generalized linear models (GLM)
- Generalized additive models (GAM)
- Classification trees
- Survival analysis
- Neural networks
- Principle components analysis (PCA)
- Multidimensional scaling (MDS)
- Time series analyses, etc.

All techniques have a common problem:

Garbage in, garbage out :(

In particular, researchers often ignore important mathematical assumptions that underlay the statistical methods in use

A simple linear regression example…

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

- For each Y_i , X_i should be normally distributed
- For all Y_i , the variance of X_i should be equal
- Y_i are supposed to be independent (i.e. there should be no temporal or spatial correlation)
- There should be no outliers
- Residuals are also supposed to be normally distributed

To avoid type I or type II errors…

… a detailed data exploration must be applied before the actual analysis begins

Exploratory Data Analysis (EDA) can take up to 50% of the time spent on the whole analysis!

A general protocol for EDA

- 1. Formulate biological hypothesis. Carry out experiment and collect data
- 2. Explore the data:
  - Outliers among Y and X
  - Homogeneity of variance of Y
  - Normality of Y and X
  - Troubles with zero values of Y
  - Collinearity among predictors (Xs)
  - Interactions between predictors
  - Independence of Ys
- 3. Apply statistical model

The power of graphics

"…there is no statistical tool that is as powerful as a well chosen graph"
(Chambers et al., 1983)

Always start your analyses by graphing data in some informative way

Chambers, J. M., Cleveland, W. S., Kleiner, B. and Tukey, P. A. (1983) *Graphical Methods for Data Analysis.* Wadsworth & Brooks/Cole

The power of R graphics: examples

- Demo of R graphics:

```
> demo(graphics)
```

- R Graph Gallery (with source code!):

http://addictedtor.free.fr/graphiques/

4. EDA with R: Graphics

4.2. Basics of R plotting: scatterplots

Reading our example dataset in:

```
> LWdata = read.table(
  file = "pH_down_experiment.txt",
  header = TRUE,
  sep = "\t")
> names(LWdata)
> head(LWdata)
> LWdata
```

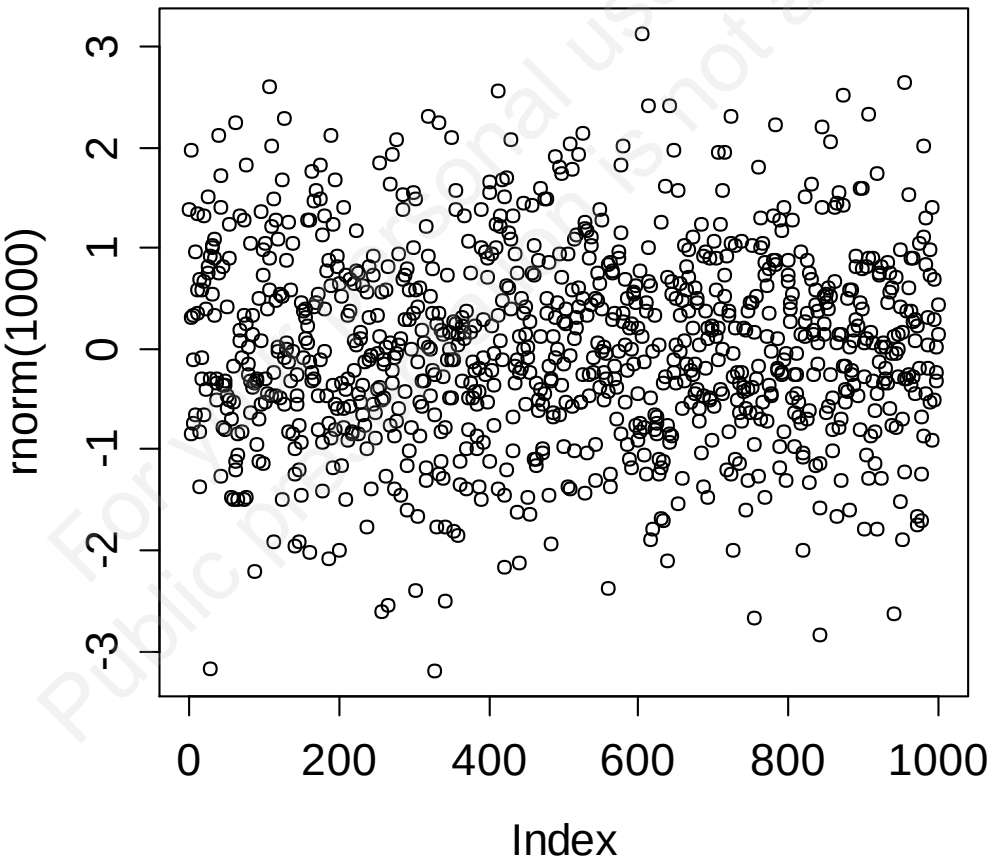

In **RStudio**, go to tab Workspace and click LWdata

The `plot()` function

- *Generic* R function, i.e. behaves differently depending on the class of the object being plotted
- Allows for fine control over the graph appearance => has many arguments
- We will only "scrape the surface" of the R graphics as this is a huge topic on its own

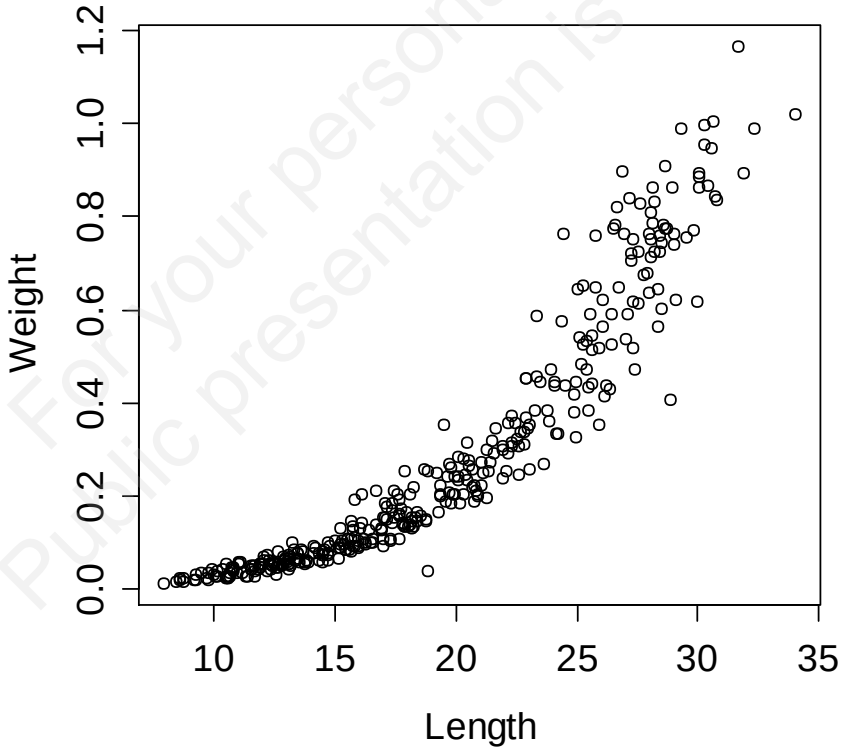
An example we already saw

1000 numbers randomly drawn from a normal distribution



Checking the relationship between *Dreissena* shell length and dry weight

```
> plot(LWdata$Length, LWdata$Weight)
> # or, alternatively, in a formula form
> plot(LWdata$Weight ~ LWdata$Length)
```



Author: Sergey Mastitsky

Making life a bit easier with the `attach()` function

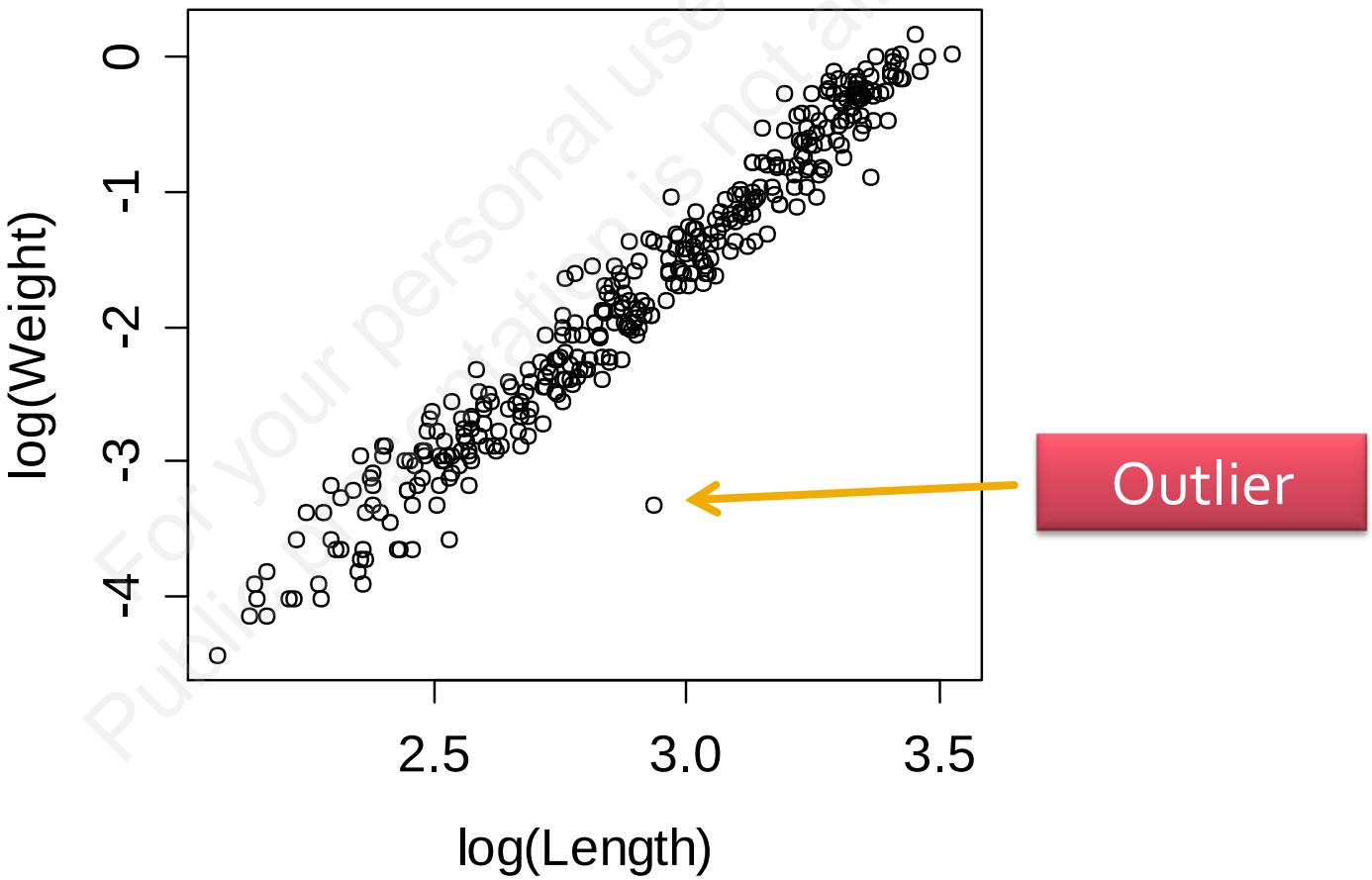
`attach()` makes the content of a data frame visible for the R's *search environment*
=> no need to use **`$`** for indexing

```
> attach(LWdata)

> plot(Weight ~ Length)
```

Plotting log-transformed data

```
> plot(log(Weight) ~ log(Length))

> # change the axis titles

> plot(log(Weight) ~ log(Length),
xlab = "log transformed Length",
ylab = "log transformed Weight")
```

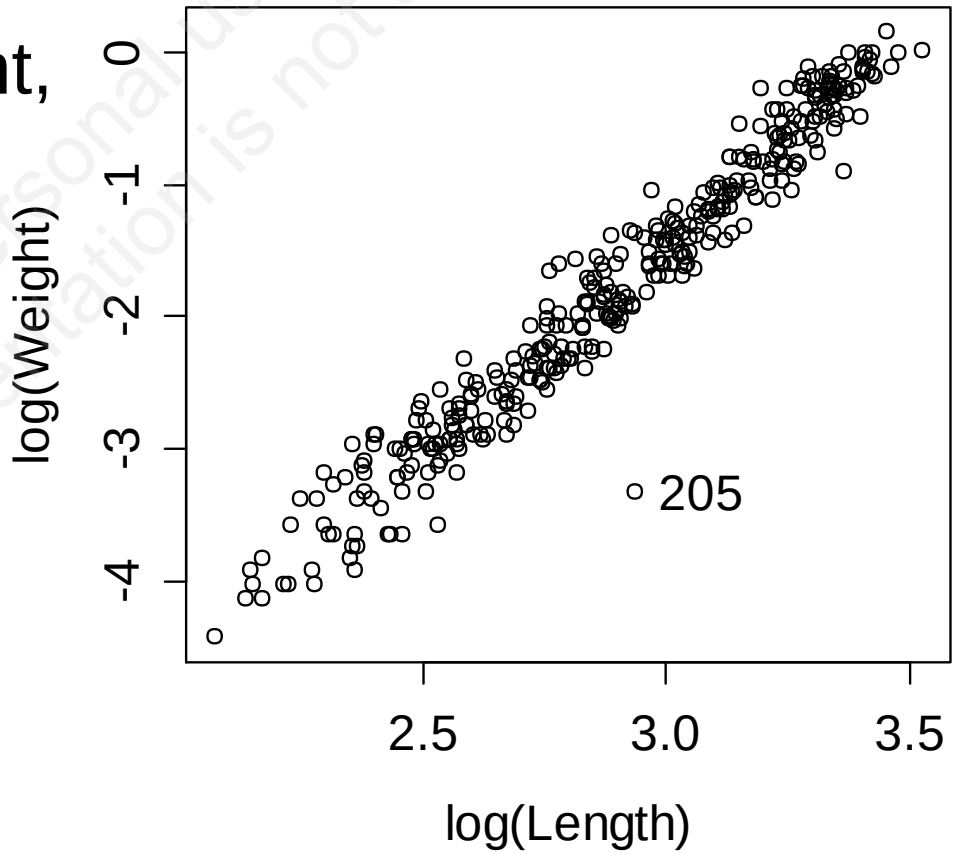
Log transformed data



How to identify an outlier?

```
> identify(log(Weight) ~ log(Length))
```

Then click near the point, then ESC…



Remove the outlier

```
> detach(LWdata)
> LWdata <- LWdata[-205, ]
> attach(LWdata)
> # check if it worked out
> plot(log(Weight) ~ log(Length))
```

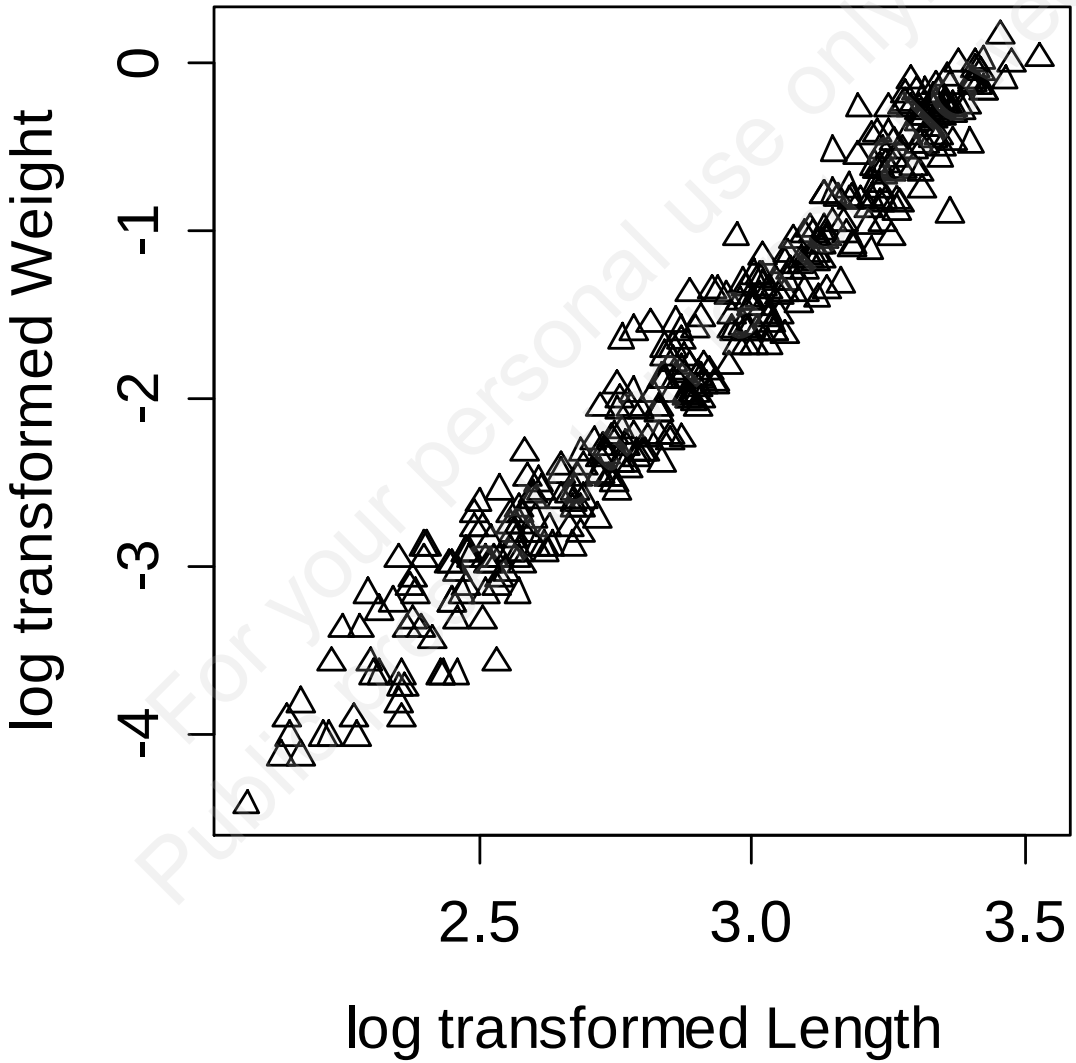
Changing the plotting character

```
> plot(log(Weight) ~ log(Length),
  xlab = "log transformed Length",
  ylab = "log transformed Weight",
  pch = 2)
```

Exercise:

Play around with other **pch** values…

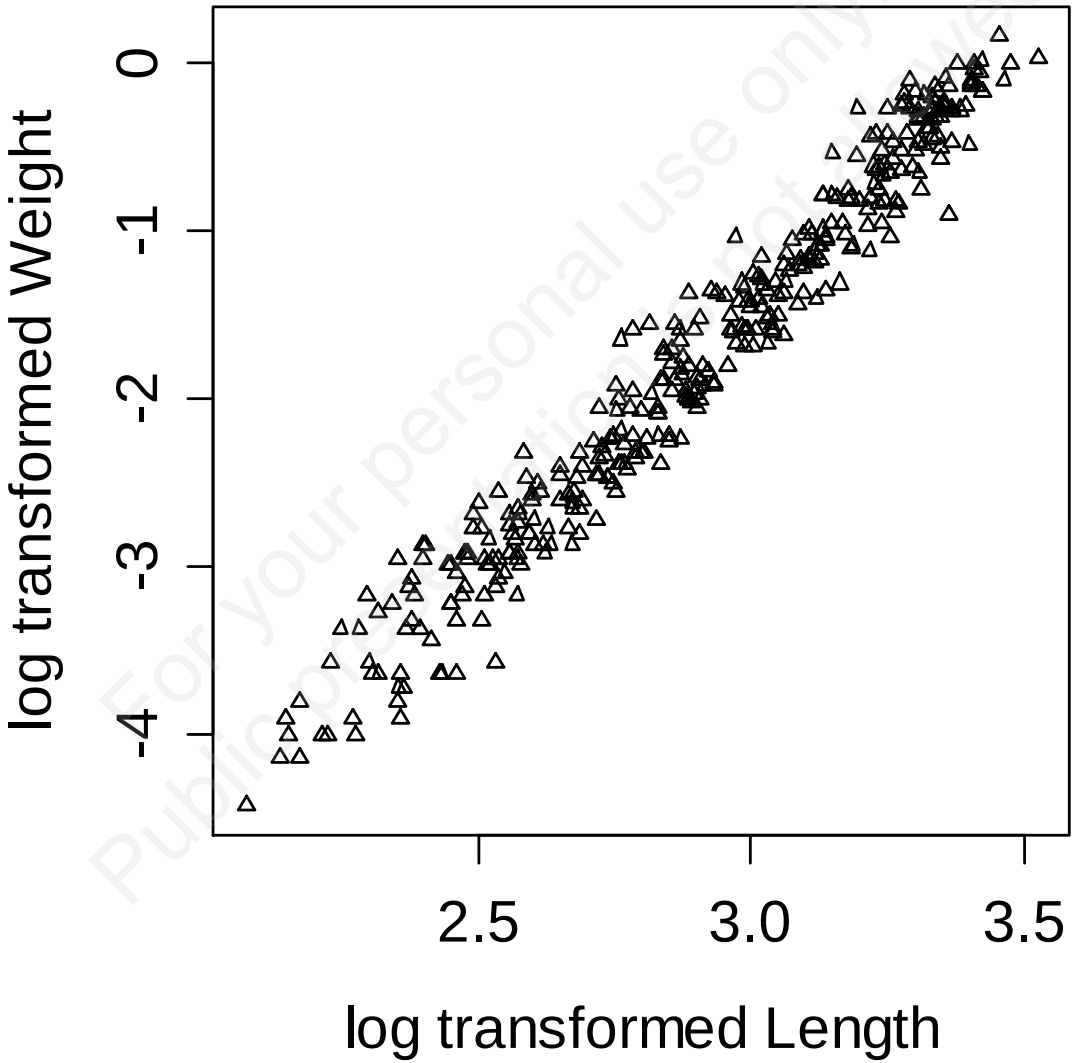
Plotting character changed



Changing the size of the plotting character

```
> plot(log(Weight) ~ log(Length),
  xlab = "log transformed Length",
  ylab = "log transformed Weight",
  pch = 2, cex = 0.6)
```

Size of the plotting character changed

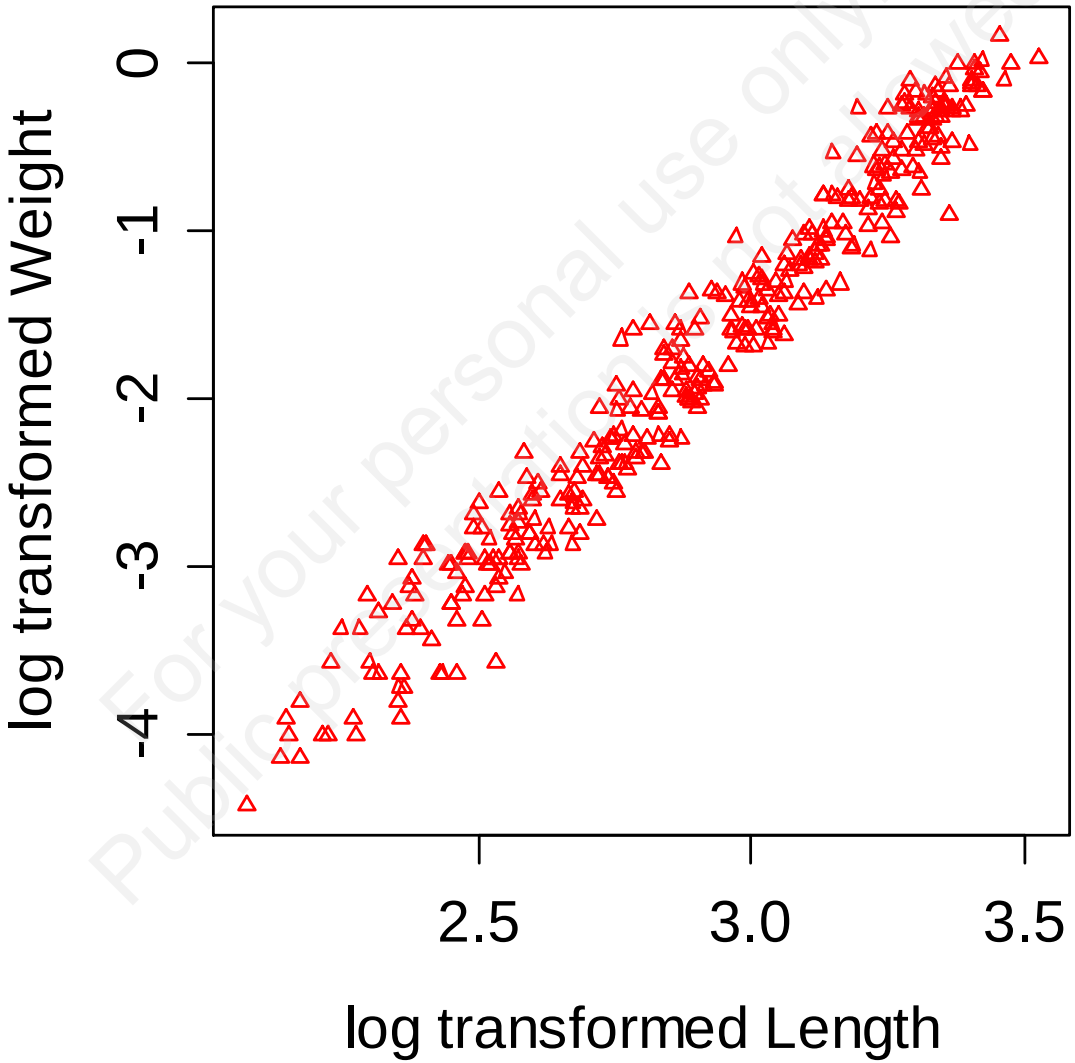


Changing the color of the plotting character

```
> plot(log(Weight) ~ log(Length),
  xlab = "log transformed Length",
  ylab = "log transformed Weight",
  pch = 2, cex = 0.6, col = 2)
```

Alternatively, could be `col = "red"`
Try `colors()` to see all options

Red triangle as a plotting character

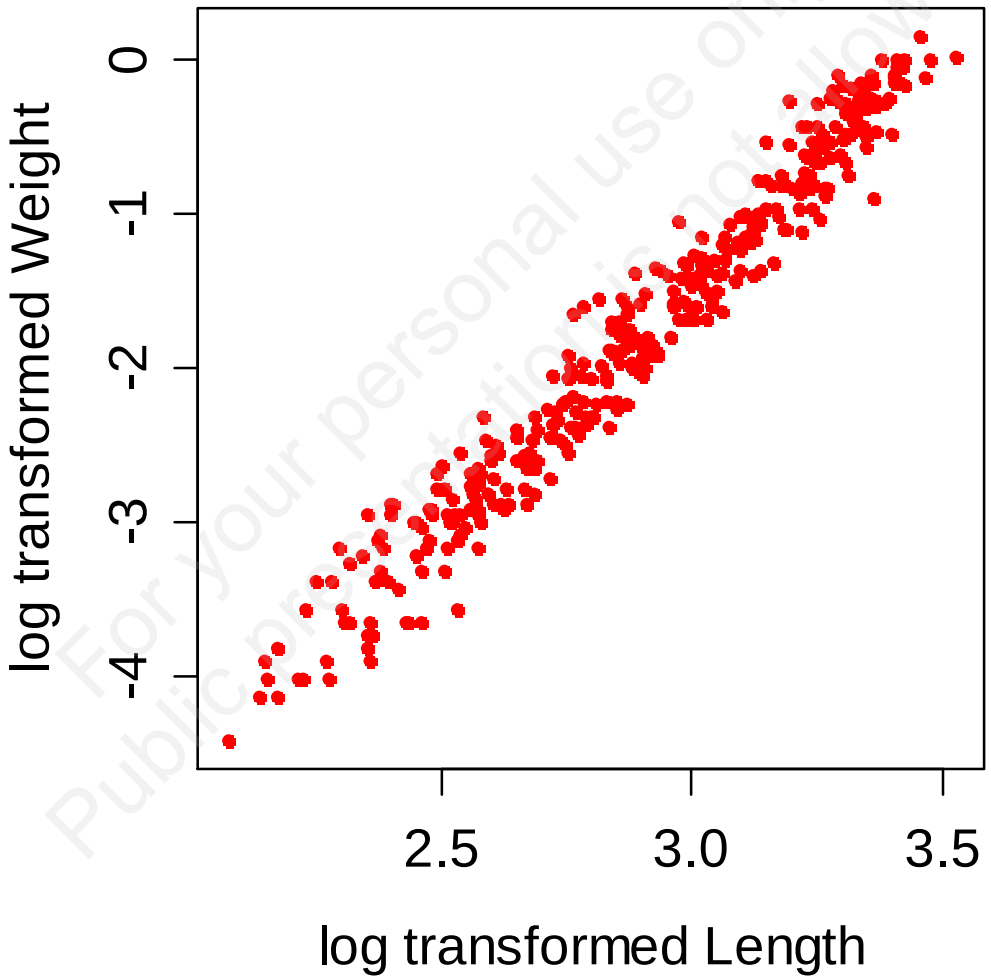


Author: Sergey Mastitsky

A slightly better version of the plot

```
> plot(log(Weight) ~ log(Length),
  xlab = "log transformed Length",
  ylab = "log transformed Weight",
  pch = 20, col = 2)
```

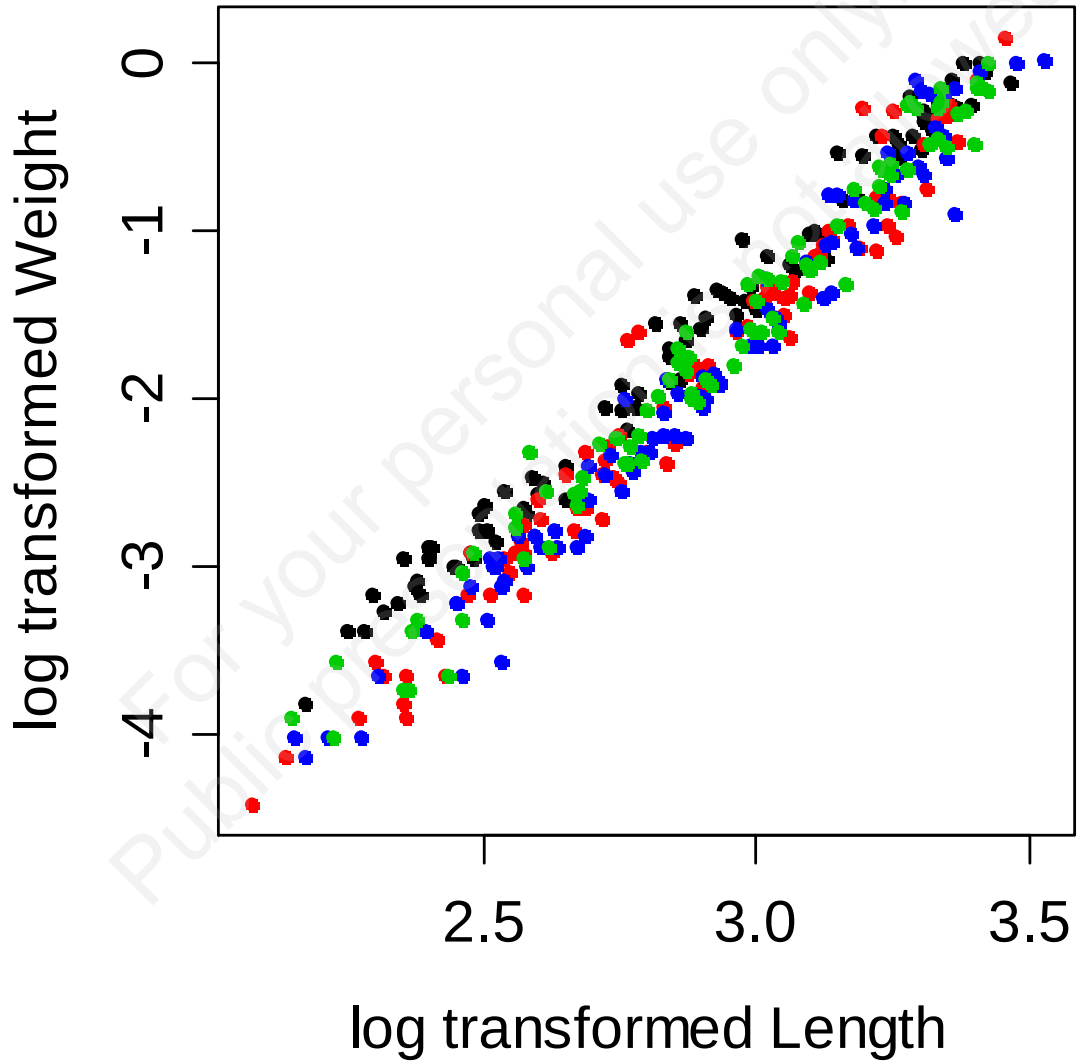
A slightly better version of the plot



Different color for each group

```
> plot(log(Weight) ~ log(Length),
 xlab = "log transformed Length",
 ylab = "log transformed Weight",
 pch = 20,
  col = as.numeric(Treatment))

> # check how it works:
> as.numeric(Treatment)
```

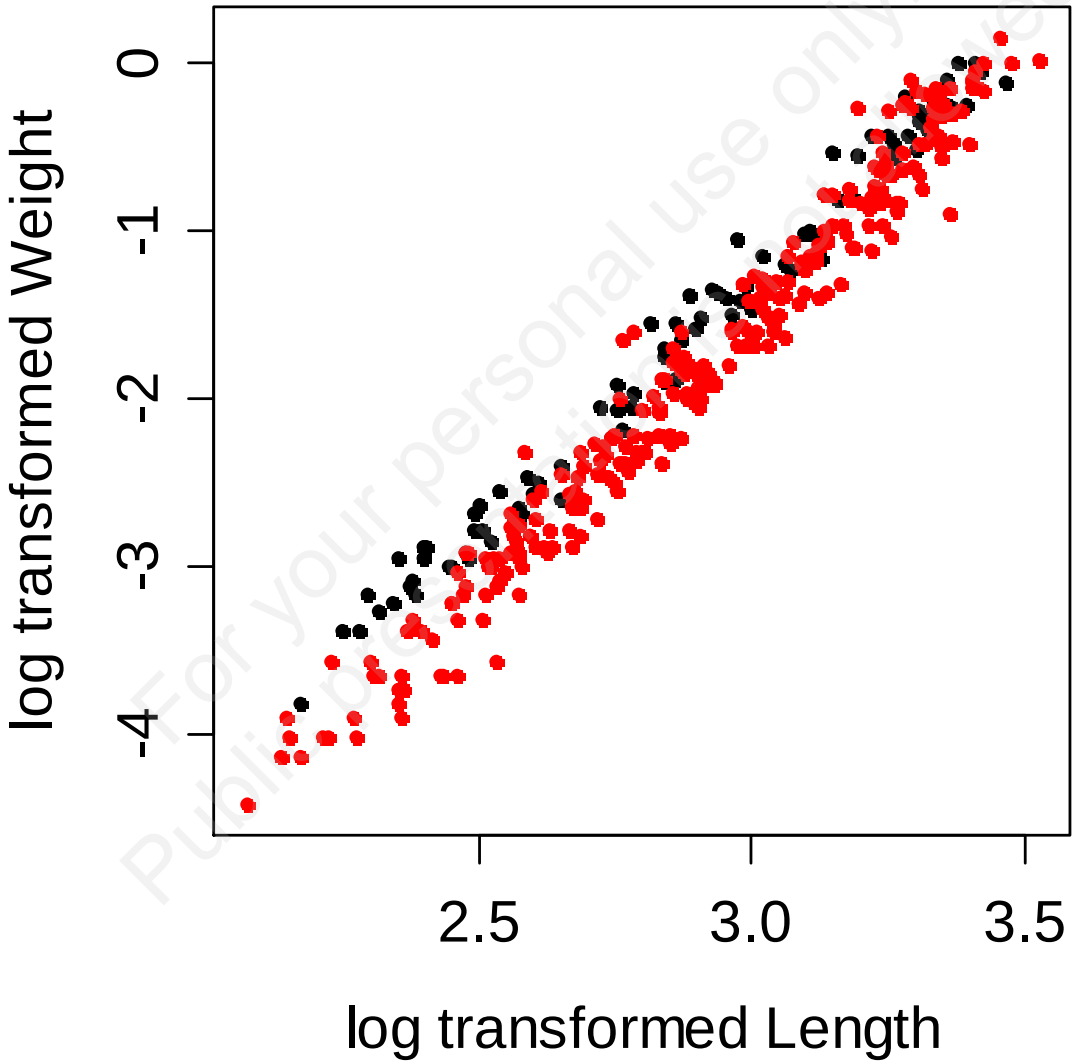
Different colors for all four groups



Highlighting the control group

```
> plot(log(Weight) ~ log(Length),
  xlab = "log transformed Length",
  ylab = "log transformed Weight",
  pch = 20, col = ifelse(
     Treatment == "Control", 1, 2))
```

Highlighting the control group



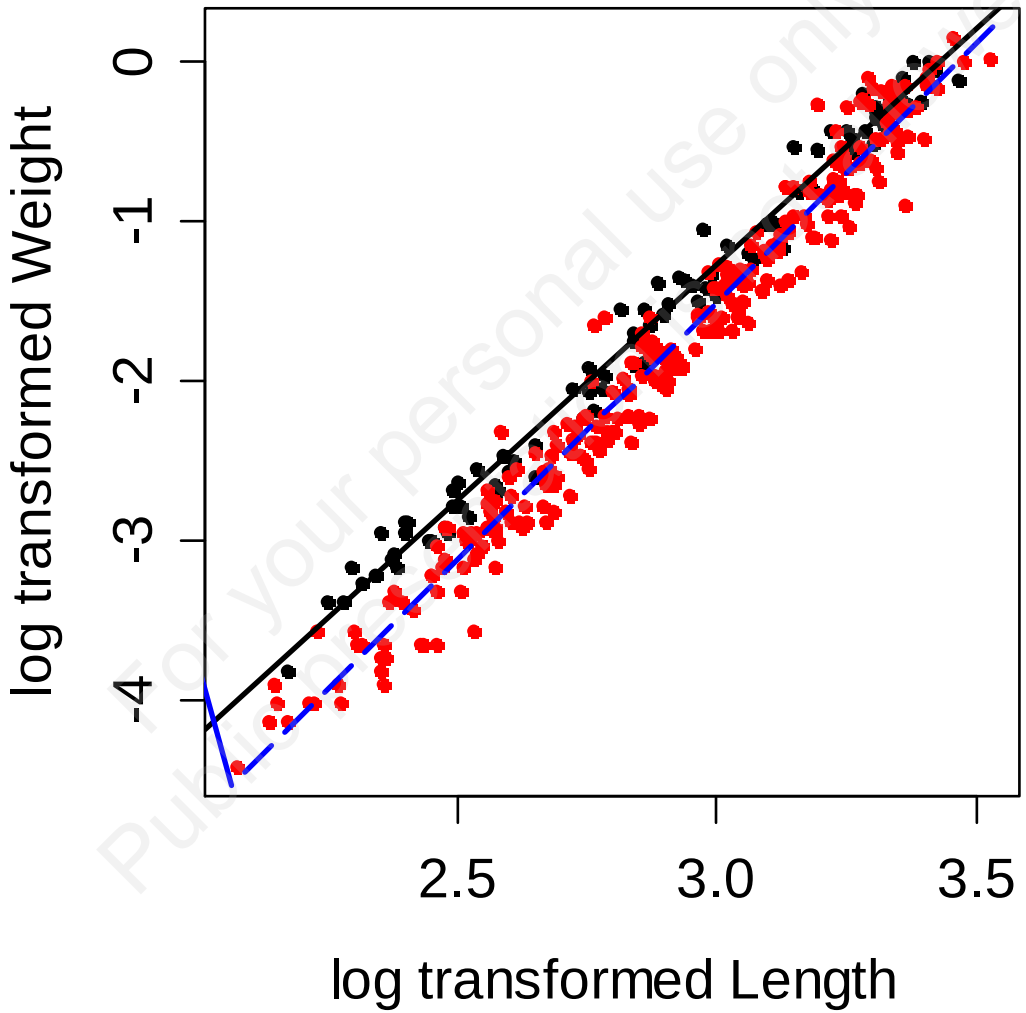
Adding regression lines to the plot

```
> contr <- lm(log(
Weight[Treatment == "Control"]) ~
log(
Length[Treatment == "Control"]))

> other <- lm(log(
Weight[Treatment != "Control"]) ~
log(
Length[Treatment != "Control"]))
```

Adding regression lines to the plot

```
> abline(contr, lty = 1, lwd = 2)

> abline(other, lty =5, lwd = 2,
         col = "blue")
```

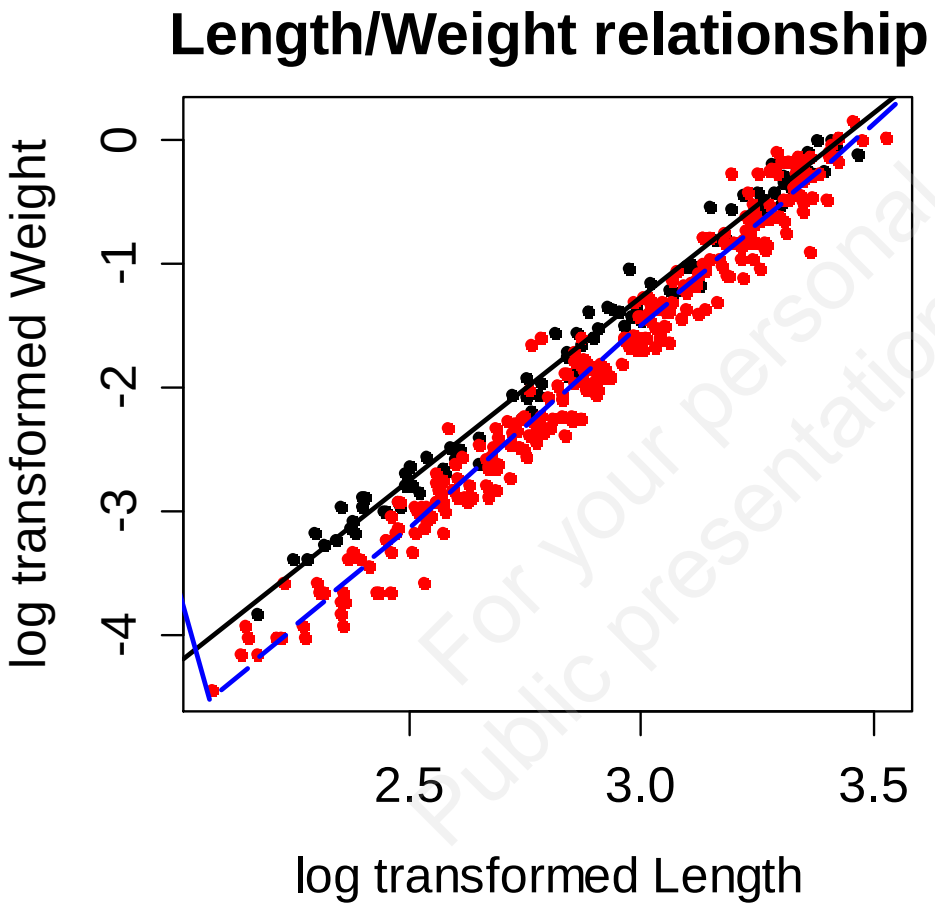
Regression lines added



Adding a title

```
> plot(log(Weight) ~ log(Length),
  xlab = "log transformed Length",
  ylab = "log transformed Weight",
  main = "Length/Weight relationship"
  pch = 20, col = ifelse(
     Treatment == "Control", 1, 2))

> abline(contr, lty = 1, lwd = 2)
> abline(other, lty = 5, lwd = 2,
     col = "blue")
```

Finalized plot



! Don't forget to **`detach(LWdata)`**

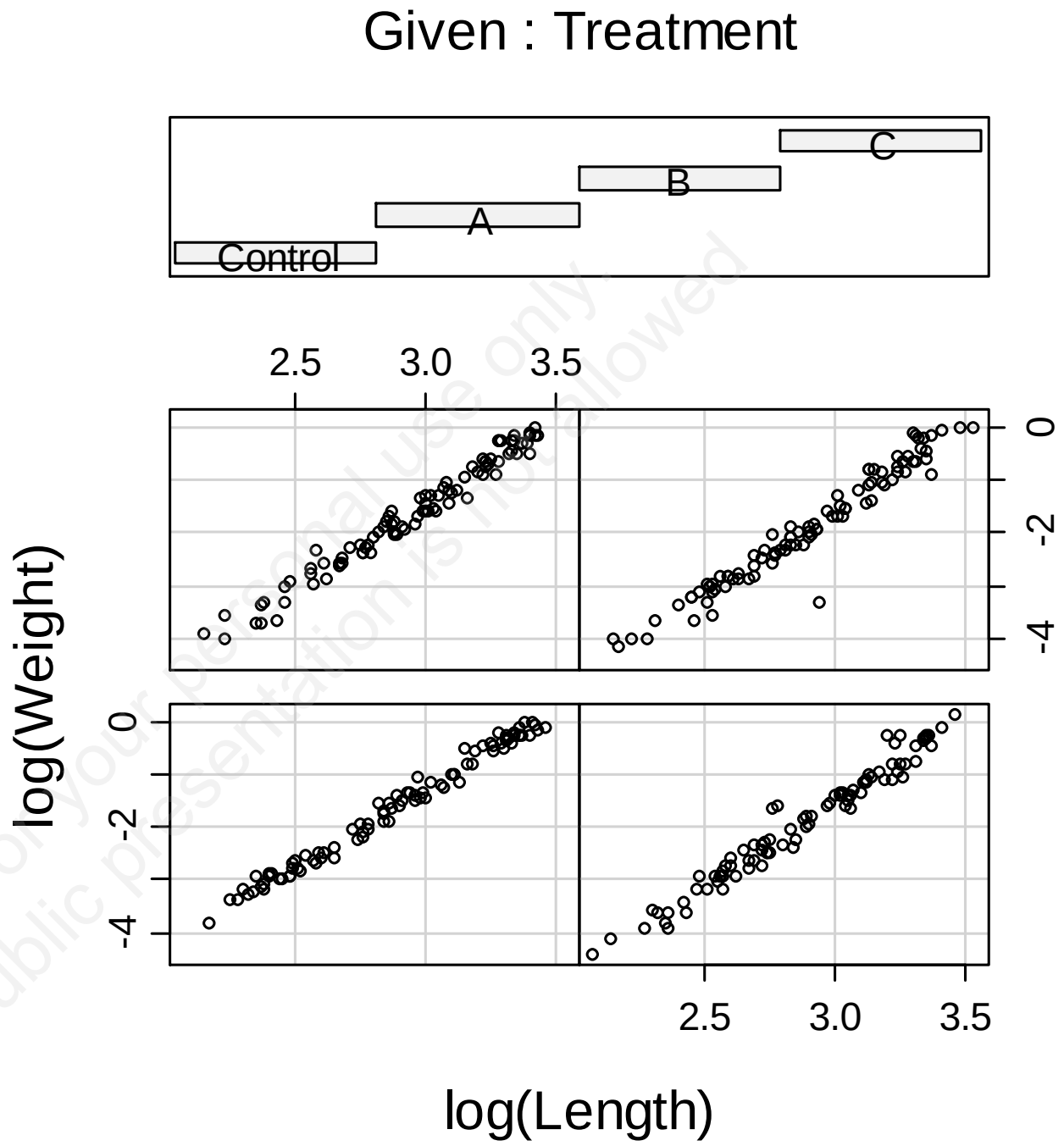
4. EDA with R: Graphics

4.2. Basics of R plotting: conditional scatterplots

The coplot() function

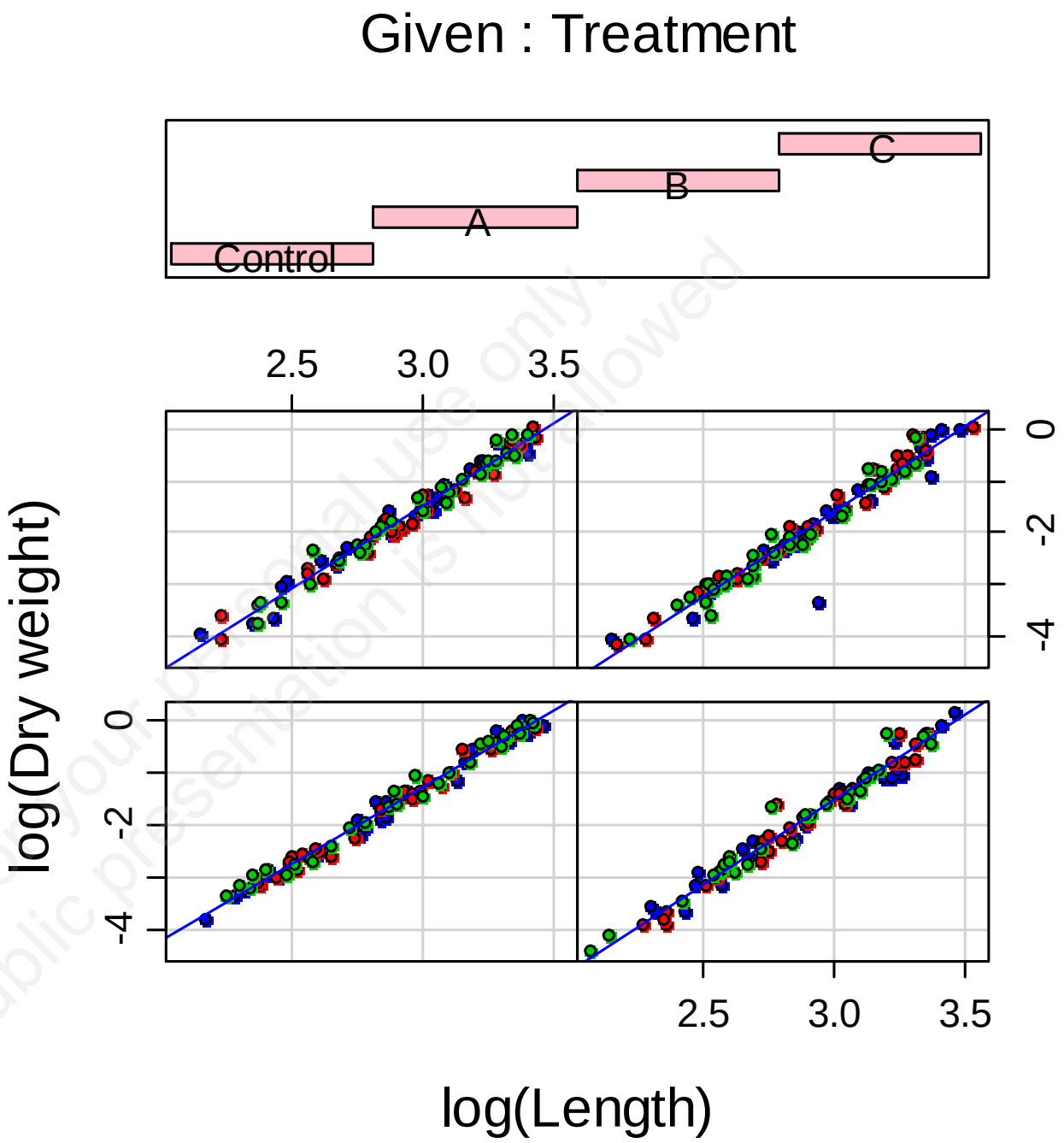
```
> coplot(log(Weight) ~
  log(Length) | Treatment)

# allows you to check what
# is going on in each
# Treatment group
```

A coplot



It could look even better…

see ?coplot



4. EDA with R: Graphics

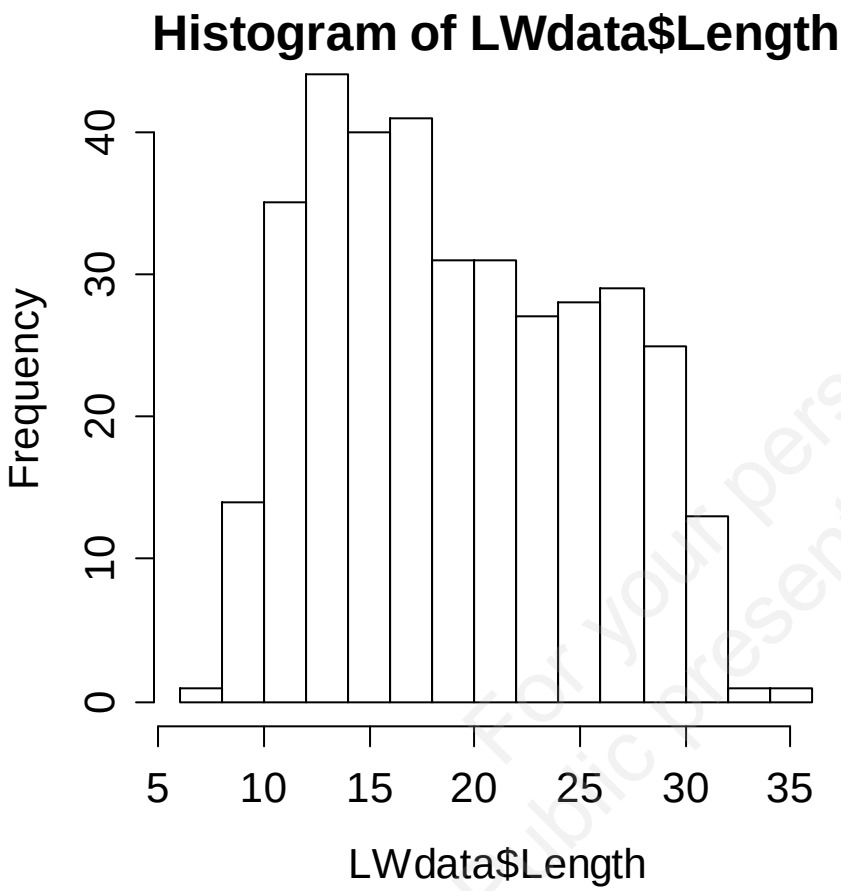
4.3. Basics of R plotting: histograms and density plots

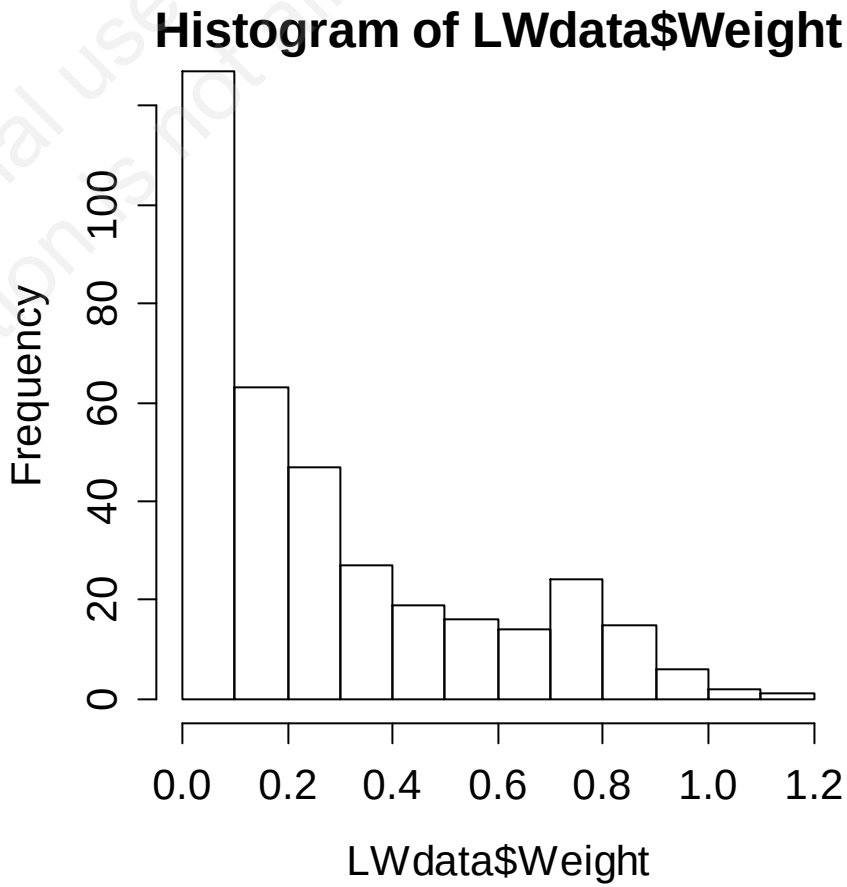
Histograms

- Histograms provide a reasonable visual summary of the shape of a distribution
- In R, a simple histogram can be drawn as follows:

```
> hist(LWdata$Length)

> hist(LWdata$Weight)
```

Histograms of Length and Weight

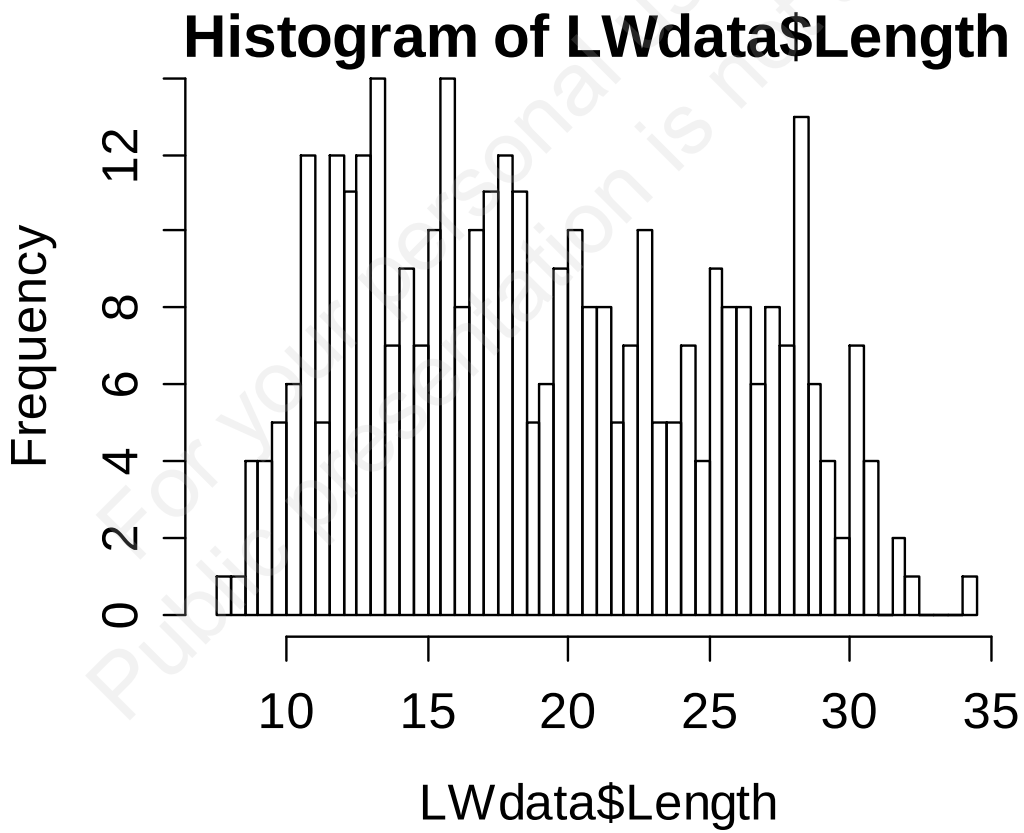




Author: Sergey Mastitsky

Changing the number of bars

```
> hist(LWdata$Length, breaks = 50)
```



Density function plots

- Often better than and not as misleading as histograms
- In R density curves are calculated with the `density()` function
- Individual density curves can be plotted on the same graph using the `lines()` function

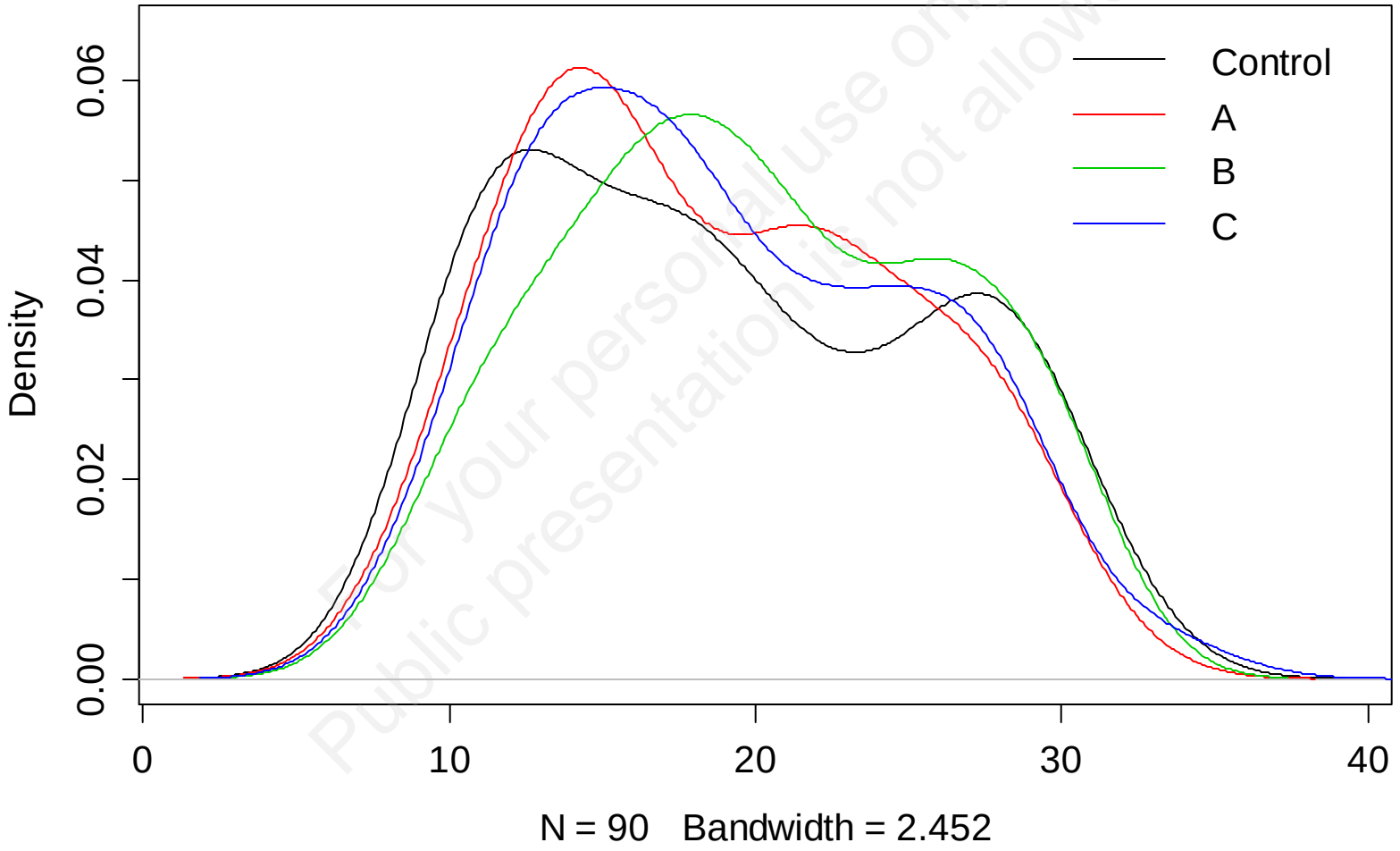
Example of a density plot

```
# Calculate density curves for each treatment:
> attach(LWdata)
> trControl = density(Length[Treatment ==
          "Control"])
> trA = density(Length[Treatment == "A"])
> trB = density(Length[Treatment == "B"])
> trC = density(Length[Treatment == "C"])

# Plot the graph:
> plot(trControl, main =" ",
  ylim = c(0, 0.065), lty = 1)
```

Example of a density plot

```
# Add the remaining three lines:
> lines(trA, col = 2)
> lines(trB, col = 3)
> lines(trC, col = 4)

# Add a legend
> legend("topright",
  lty = c(1, 1, 1, 1),  col = c(1:4),
  legend = c("Control", "A", "B", "C"),
  bty = "n")
```

Example of a density plot



Author: Sergey Mastitsky

4. EDA with R: Graphics

4.4. Basics of R plotting: normal probability plots

Normal probability plot

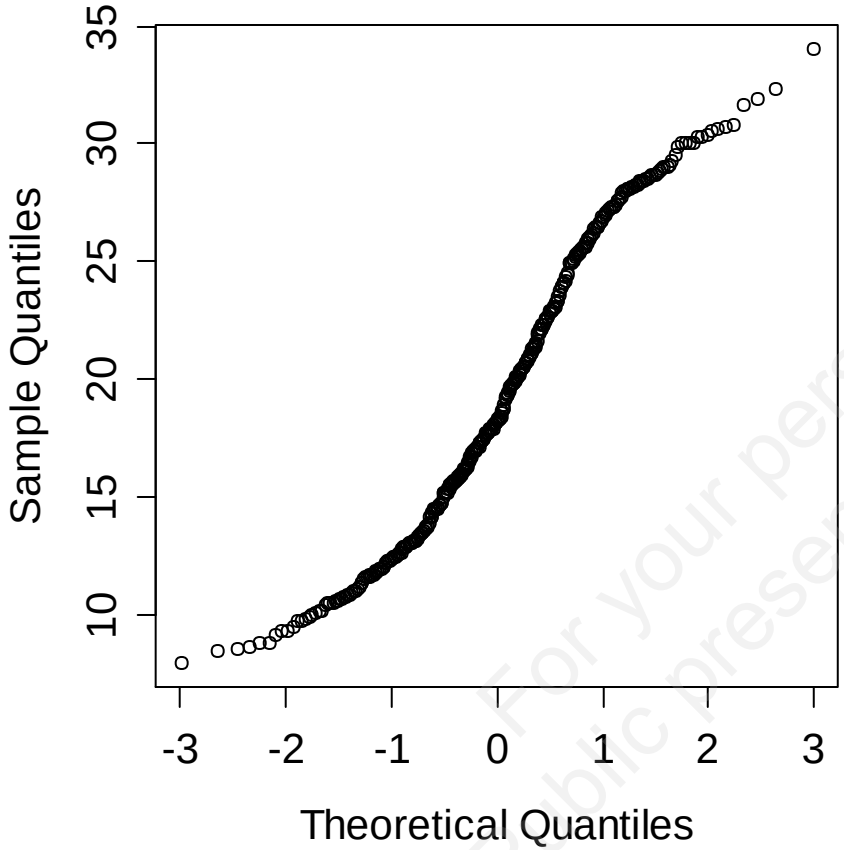
- Allows one to assess the normality
- Plots empirical data vs. those that would be expected in a standard normal distribution

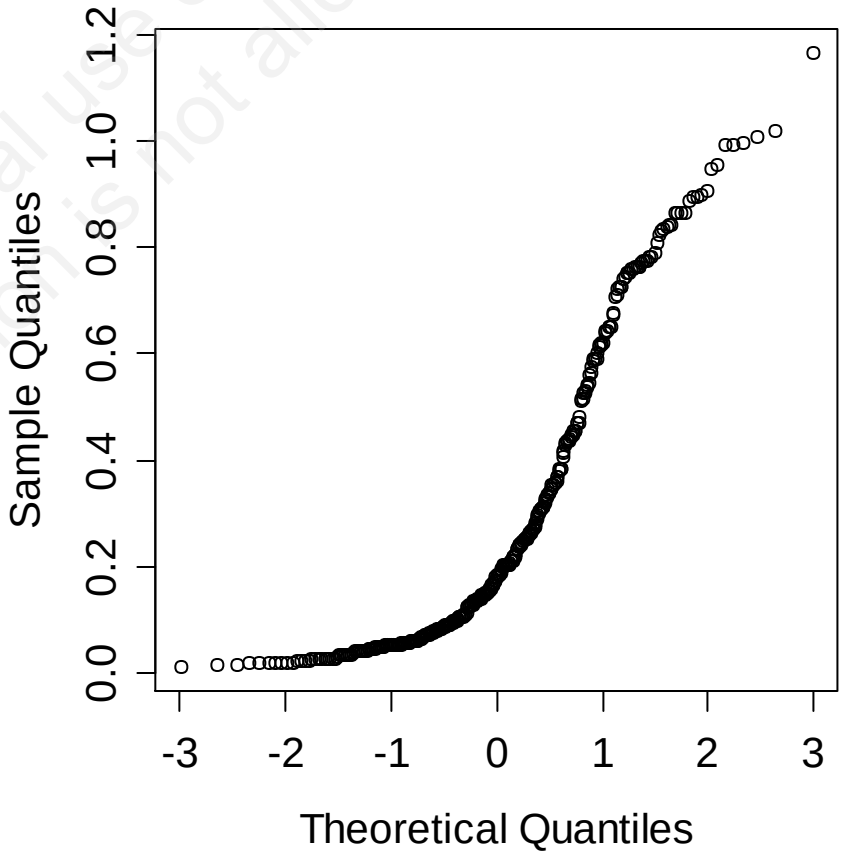
=> ideally should be a straight diagonal line

- Q-Q plot: a version of the normal probability plot that uses the empirical and expected quantiles

```
> qqnorm(LWdata$Length)
> qqnorm(LWdata$Weight)
```

Q-Q plots

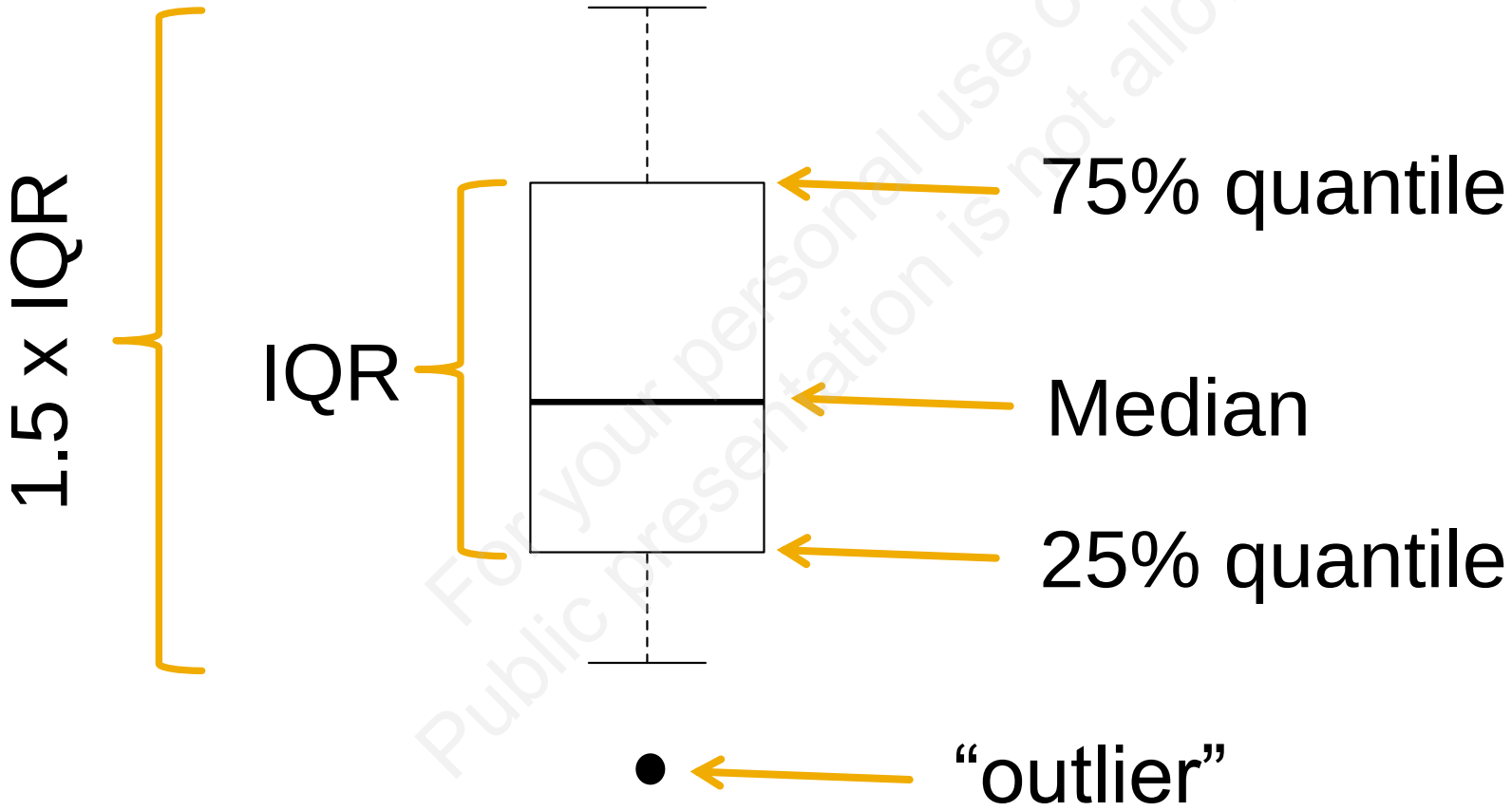




4. EDA with R: Graphics

4.5. Basics of R plotting: boxplots

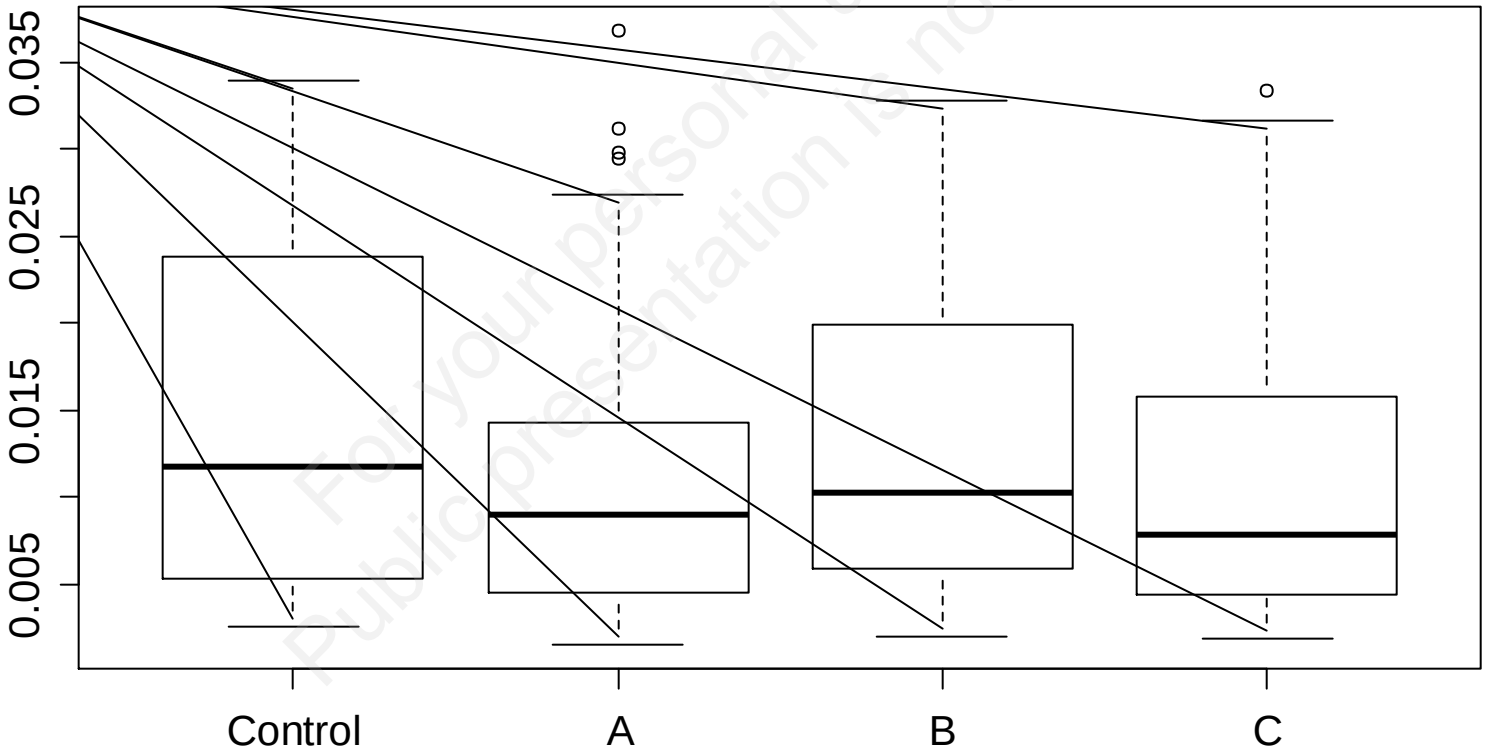
Boxplot, a graphical summary of a distribution



Author: Sergey Mastitsky

The boxplot() function

```
> boxplot(Weight/Length ~ Treatment)
```



4.6. Basics of R plotting: barplots

Barplots

- Provide graphical summary for categorical variables
- Produced with the **`barplot()`** function (see `?barlot` for detail)
- Often, the **`xtabs()`** function is needed to prepare a *contingency table*, e.g.:

```
> xtabs(~ Treatment)
Treatment
Control       A       B       C
     90      89      90      92
```

A [dummy] example of a bar chart

```
> barplot(xtabs(~ Treatment))
```