Advanced Statistics

Bernhard Angele

Class 3, October 16, 2014

Your last serving of R basics

- Working with files
- R can open all kinds of data files
- You do have to tell it where to find them
- R's working directory
- This is where R looks for files to open
- You change it using 'setwd("C:/My_example_R_directory")
- No backslashes `\` allowed! Use forward slashes instead `/`

Getting more practice with R

- Here's a free introduction to R which might be helpful to you. It repeats a lot of the things I have showed you only briefly.

Setting your working directory

- Don't know where your working directory should be?
- Start by putting your script files in a location that is sensible (to you)
    - e.g. `C:\Some_Document_Folder\AdvStats\Class3\` or `/home/My_Name/Documents/MastersProject/Data`
    - That directory should also be your working directory for the current session
    - In RStudio, open your script file, then use the menu bar and go to Session –> Set Working Directory –> To Source File Location
    - RStudio will write a setwd command to the console.
    - Copy and paste this command to your script, and you're set. Every time you run the script, the working directory will be automatically set
- When you use R Markdown, Knitr sets your working directory to the location of the Rmd file automatically.

Opening files and importing data

- If you have data in Excel:
- Save them as comma-separated values (CSV)
- If you are using Excel formulas and other bells and whistles, check that your data are exported correctly
- Make sure the first row of every column contains its column name
- Save this file in your working directory (remember where that was?)
- Now you can import the file using `new_object_with_a_sensible_name <- read.csv(file = "my_filename.csv")`
- If there is an error/warning message, check that there are no missing cells
    - Same number of data points in each row

Data Frames

- A Data Frame is a combination of a list and a matrix
- It gets its row/column structure from the matrix
- But you can use columns of any data type (numeric, logical, factor (discrete), logical, character)
- Perfect for the kind of analyses we would like to do
- Very similar to SPSS data files
- Get a column by using `object_name$column_name`
- Get a column by slicing: `object_name[,1]`
- Get a row by slicing: `object_name[1,]`
- Get a cell by slicing: `object_name[1,1]`

Packages

- Many authors have worked on extending R and teaching it new tricks
- Packages are collections of functions that help you perform a certain task
- For example, the `ez` package has all kinds of useful functions for performing ANOVAs
- To install a package, type `install.packages("package_name")` at the Console.
- e.g. `install.packages("ez")`
- You can ask R to install multiple packages at once:
- Run this line in the console now: ‘install.packages(c(“ez”,“reshape”,“ggplot2”,“lme4”, “plyr”))

Loading packages

- Installed packages are not loaded automatically
- If you want to use a command from a package, you need to load it: `library(package_name)`
- e.g. `library(ez)`
- Once it is loaded, the package and all of its commands are ready to go until you restart R.

Starting a script

Good practice when starting a script:

```r
# My_testscript.R
# Author: Bernhard Angele
# Description: Demonstrates how to start a script
###########################

rm(list = ls()) # clear the workspace to avoid problems caused by old objects
library(ez) # load (only) those libraries that you need
?library(reshape)
library(ggplot2)
# Set your working directory
setwd("C:/I_know_my_working_directory_and_this_is_it/")
# Now load your data, do your analyses and win fame and fortune!
```

Starting an Rmd file

- Fill in the header:

```
---
title: "My fantastic results section"
output: pdf_document
---
```

- Then do the same things as above in the first code chunk
- Hint: you can tell R to hide this boring code by adding `hide = TRUE` to the header. It will still get evaluated

```
Start your code chunk like this: ```{r, hide = TRUE}
```

Comparing multiple groups

- t-tests are nice if you only have two groups that you want to compare.
- But maybe you have more groups

Example: A researcher wants to find out if there is a systematic difference in intelligence between MSc students from different universities. She performs intelligence tests on 10 students each from BU, University of Southampton and Oxford University and records the results.

Making fake data for our example

- Let's assume that the true state of affairs is that there is no difference in intelligence
- In that case, all intelligence scores would come from the same distribution: a normal distribution with mean = 100 and sd = 15
- Let's generate 3 data sets according to this criterion:

```
# The following line sets the random number generator to a specific state
# and ensures that you get the same numbers that I did.
set.seed("16102014")
bu <- rnorm(n = 10, mean = 100, sd = 15)
soton <- rnorm(n = 10, mean = 100, sd = 15)
oxford <- rnorm(n = 10, mean = 100, sd = 15)
```

What if we use t-tests?

- We could simply perform t-tests on these data
- How many would we need?
- 3: BU vs. Soton, BU vs. Oxford, Soton vs. Oxford

BU vs Soton

```r
t.test(bu, soton)
```

```
##
##  Welch Two Sample t-test
##
## data:  bu and soton
## t = -0.7242, df = 17.09, p-value = 0.4787
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -19.924   9.738
## sample estimates:
## mean of x mean of y
##     94.61     99.70
```

BU vs Oxford

```r
t.test(bu, oxford)
```

```
##
##  Welch Two Sample t-test
##
## data:  bu and oxford
## t = 0.5039, df = 17.34, p-value = 0.6207
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -11.44  18.63
## sample estimates:
## mean of x mean of y
##     94.61     91.01
```

Soton vs Oxford

```r
t.test(soton, oxford)
```

```
##
##  Welch Two Sample t-test
##
## data:  soton and oxford
## t = 1.383, df = 17.98, p-value = 0.1837
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -4.514 21.890
## sample estimates:
## mean of x mean of y
##     99.70     91.01
```

Anything wrong with that?

- We are doing three independent t-tests
- Each t-test has a 5% chance of producing a spurious result (α)
- What is the chance that we get at least one spuriously significant result?
- It's 1 - the chance that we get no spurious results
- $1 - .95 * .95 * .95 = .14$
- We have a problem: our α is almost three times as high as it should be.
- SPSS calls this LSD (least significant differences) - don't use it!

Exercise: Give it a try

- Instead of running the simulation on my computer, I'll run it on **you**
- What I mean by that:
- Generate three data sets like I've just shown you
- **Don't** use `set.seed()` now, or we'll all get the exact same data
- Make sure that all three data sets are samples from the same normal distribution
  - Same mean and sd, of course same n as well
- Run three two-sample t-tests comparing the means
- Afterwards, I will ask you and count how many of you found at least one significant result.
- If the α level isn't inflated, only one or two of you should find one

Solutions

- We can adjust the α level of each t-test:
- If we divide the alpha level by the number of tests, we get $.05/3 = .0167$
- Our total α is then: $1 - (1 - .05/3)^3 = .049$
- This is called a **Bonferroni correction**
- Problem solved?
- Yes, but the lower the α level, the lower the power.
- Better ways (but still lowering power):
- Holm-Bonferroni (same principle as Bonferroni, but better power)
- Tukey's HSD (honestly significant differences)
- Maybe we just want to know if there is a difference at all between any of these three means
- One-way ANOVA

Putting our fake data in a different format

- Usually, we aren't getting our data neatly in different objects
- Instead, we'll have a big table with our data
- If we are going to put our data in this format, we need a variable (or column) identifying the group
- Let's add that to each of our data sets

```
bu <- data.frame(iq = bu, group = "BU")
soton <- data.frame(iq = soton, group = "Soton")
oxford <- data.frame(iq = oxford, group = "Oxford")
```

Looking at these data now

```
bu
```

```
##        iq group
## 1  123.53    BU
## 2  107.44    BU
## 3  102.43    BU
## 4   88.58    BU
## 5   76.41    BU
## 6   80.20    BU
## 7  113.23    BU
## 8   96.15    BU
## 9   90.62    BU
## 10  67.46    BU
```

Putting it all together

```
iqdata <- rbind(bu, soton, oxford)
str(iqdata)
```

```
## 'data.frame':    30 obs. of  2 variables:
##  $ iq   : num  123.5 107.4 102.4 88.6 76.4 ...
##  $ group: Factor w/ 3 levels "BU","Soton","Oxford": 1 1 1 1 1 1 1 1 1 1 ...
```

- Data frames are smart:
- R has automatically converted `group` into a factor
    - A factor is a discrete variable (nominal scale)
- R left `iq` as a numeric variable
    - Numeric variables are continuous (interval or ratio scale)

Running an ANOVA – by hand!

- One-way ANOVAs are so simple, all you need is a calculator (or R, even better)
- First, we need an estimate of the total variance in the data: calculate the total sum of squared deviations from the mean (or short, sum of squares, SS)
- $SS_{total} = \sum (x_i - \bar{x})^2$, where x_i is each individual value and $\bar{x}$ is the grand mean of the data.

```
(SS_tot <- sum((iqdata$iq - mean(iqdata$iq))^2))
```

```
## [1] 6674
```

Running an ANOVA – by hand! (2)

- Now we need an estimate of the variance explained by `group`
- $SS_{model} = n_k(\bar{x}_k - \bar{x})^2$, where $\bar{x}_k$ denotes the mean for each group, n_k is the number of subjects in each group, and $\bar{x}$ is the grand mean of the data.

```
bu_mean <- mean(subset(iqdata, group == "BU")$iq)
soton_mean <- mean(subset(iqdata, group == "Soton")$iq)
oxford_mean <- mean(subset(iqdata, group == "Oxford")$iq)
n <- 10

(SS_model <- sum(n*(bu_mean - mean(iqdata$iq))^2, n*(soton_mean - mean(iqdata$iq))^2, n*(oxford_mean - m
```

```
## [1] 381.1
```

Running an ANOVA – by hand! (3)

- Now get an estimate of the variance that is *not* explained by `group`, i.e. the error.
- Easy: Just subtract SS_{model} from SS_{total}.
- $SS_{error} = SS_{total} - SS_{model}$

```
(SS_error <- SS_tot - SS_model)
```

```
## [1] 6293
```

Running an ANOVA – by hand! (4)

- With SS_{model} and SS_Error, we can compute the ratio of explained variance to error (or unexplained) variance.
- First, we need to take into account the number of measurements which went into each SS
- These are called the degrees of freedom
- This is equivalent to calculating the variance as a descriptive statistic (SS/df)
- $df_{total} = n * k - 1$, where n is the number of observations per group (10) and k is the number of groups
- Why -1? Very good question. Short answer: Whenever we want to make conclusions about the population, we use n-1 instead of n.
- Long answer (if you are REALLY interested and won't shut up about it): Here

Running an ANOVA – by hand! (5)

- $df_{total} = n \cdot k - 1$
- $df_{model} = k - 1$, where k is the number of groups.
- $df_{error} = df_{total} - df_{error} = n \cdot k - k$

```
k <- 3
(df_total <- n * k - 1)
```

```
## [1] 29
```

```
(df_model <- k - 1)
```

```
## [1] 2
```

```
(df_error <- n * k - k)
```

```
## [1] 27
```

Running an ANOVA – almost done!

- Now we compute the mean squares (MS) as an estimate of the variance
- $MS_{model} = \frac{SS_{model}}{df_{model}}$
- $MS_{error} = \frac{SS_{error}}{df_{error}}$

```
(MS_model <- SS_model/df_model)
```

```
## [1] 190.6
```

```
(MS_error <- SS_error/df_error)
```

```
## [1] 233.1
```

Running an ANOVA – final steps!

- Finally, we take the ratio of the two.
- $F_(df_{model}, df_{error}) = \frac{MS_{model}}{MS_{error}}$

```
(F_value <- MS_model/MS_error)
```

```
## [1] 0.8177
```

What to do with this F-value

- It turns out that the ratio between model and error variance follows a specific distribution
- If there is no actual effect (!) and
- As long as certain assumptions are valid (more on that later)
- This distribution is called the F-distribution
- Occasionally you will get a high MS_{model} simply by chance, but such occurrences are quite rare
- The F-distribution is the probability density function for different values of the variance ratio, i.e. the F-value.
- We essentially want to test if the F-value we get is extreme enough that it could only have occurred by chance 5% of the time (our α level)

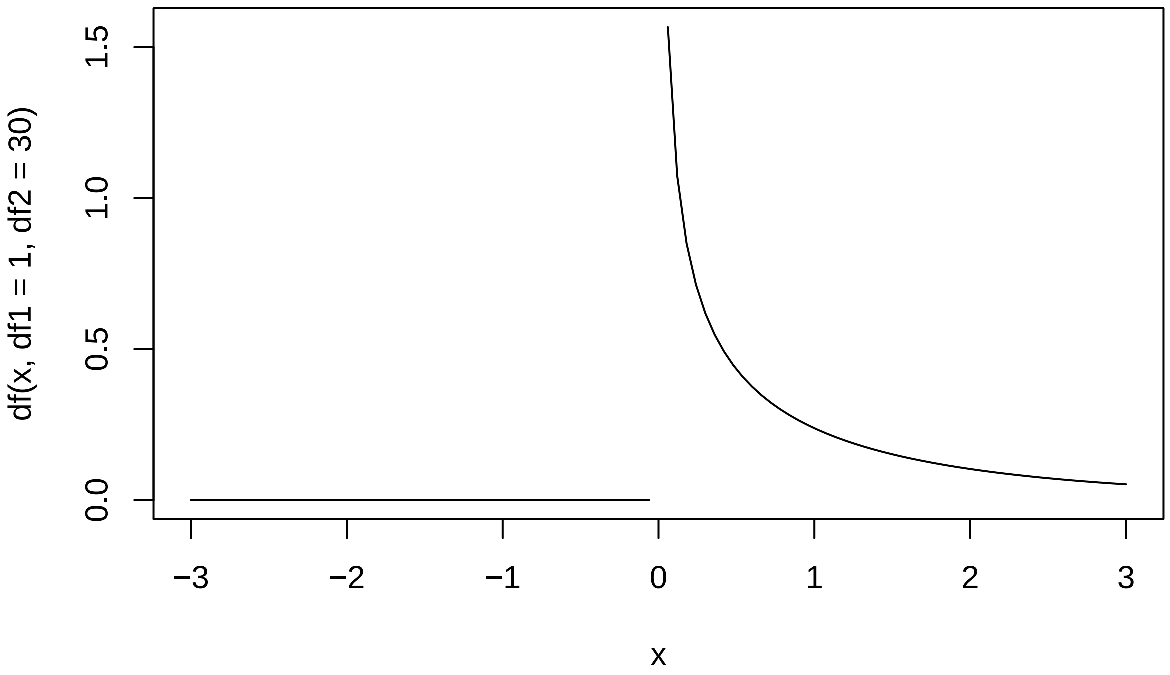
The F-distribution

- Like the *t*-distribution, the shape of the F-distribution varies depending on sample size (degrees of freedom).
- Remember that F is a *ratio* of two variances.
- Because of this, the F distribtion has *two* degrees of freedom parameters
  - $df_{numerator}$
  - $df_{denominator}$
- Guess what the R functions for the F distribution are called
- That's right: `df`, `pf`, `qf`, `rf`

Plotting it

- Let's take a look

```r
curve(df(x, df1 = 1, df2 = 30), from = -3, to = 3)
```

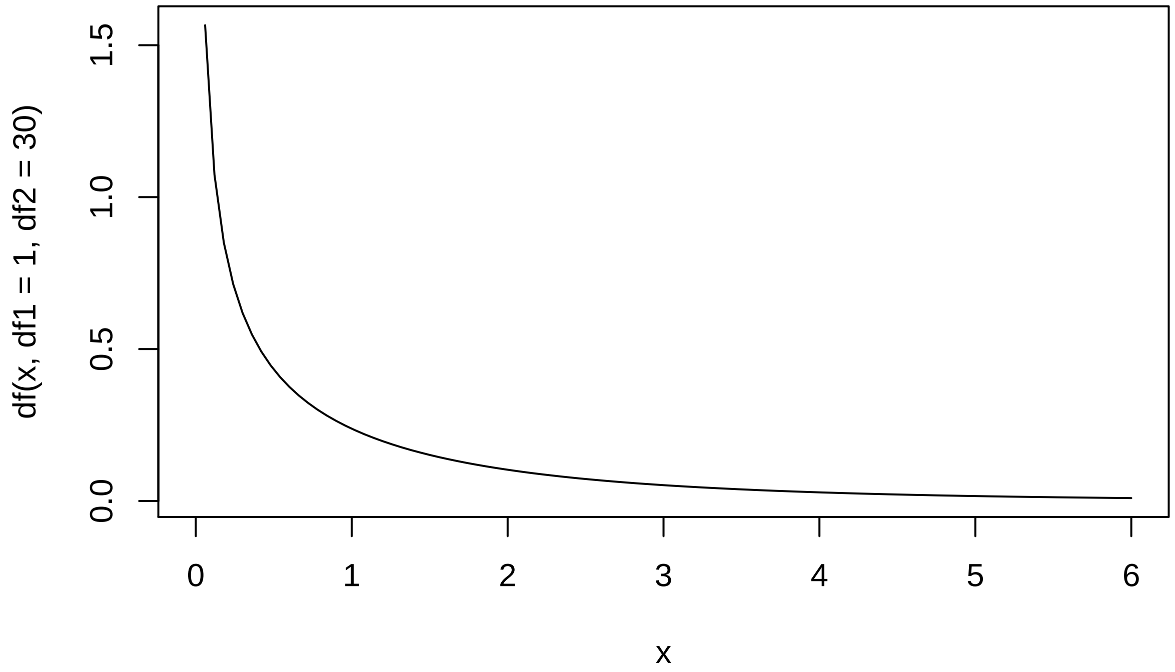


Looks like x can't be negative - this makes sense: you can't have negative variance

Plotting it (2)

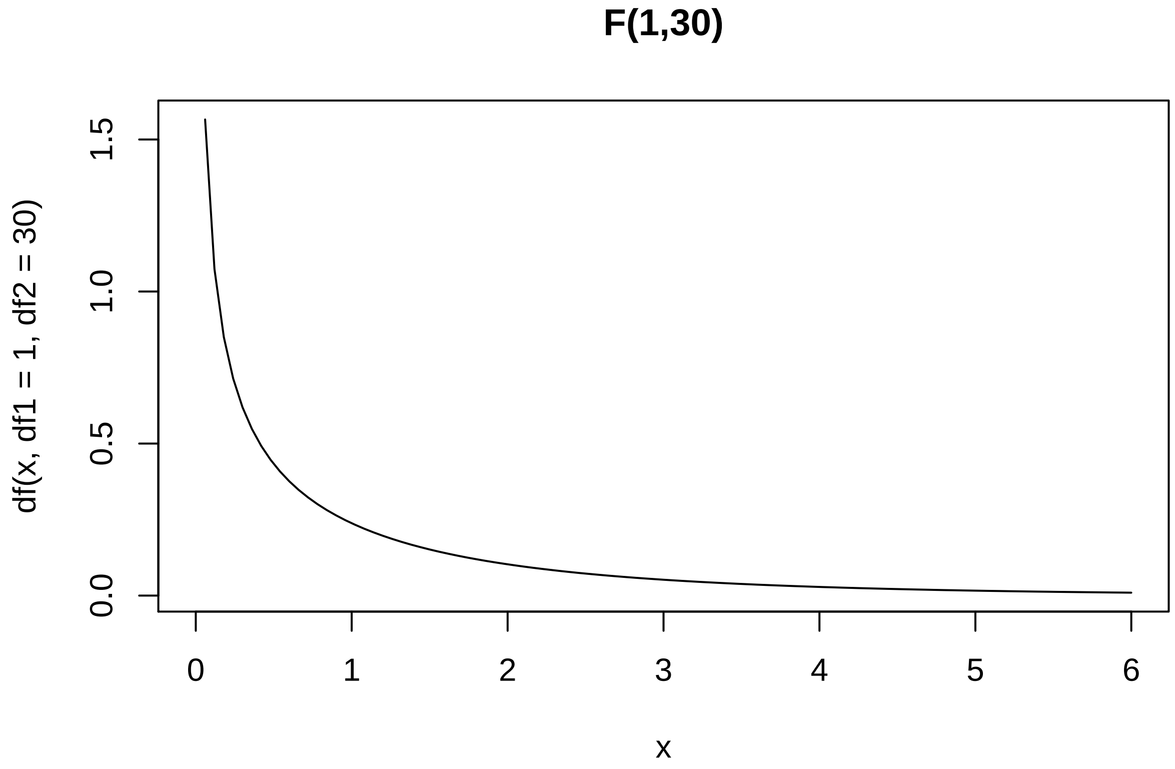
- Adjusting the x-axis

```r
curve(df(x, df1 = 1, df2 = 30), from = 0, to = 6)
```



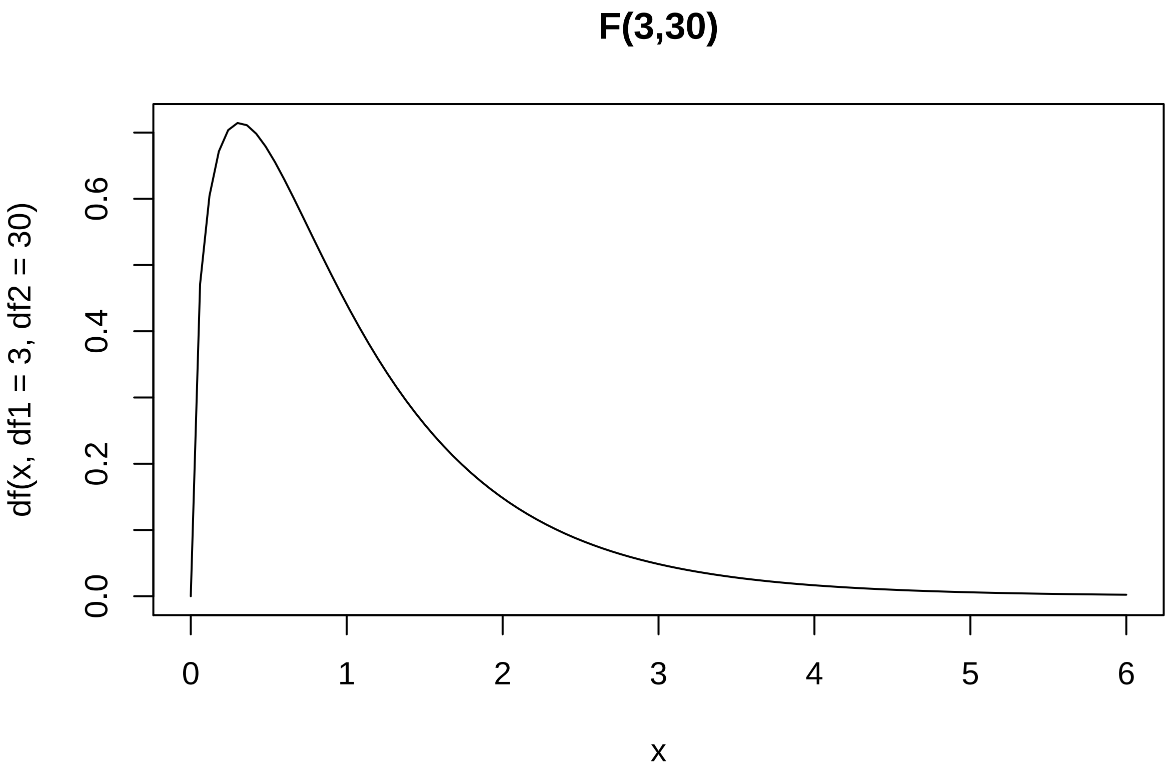
The F distribution

```
curve(df(x, df1 = 1, df2 = 30), from = 0, to = 6, main = "F(1,30)")
```



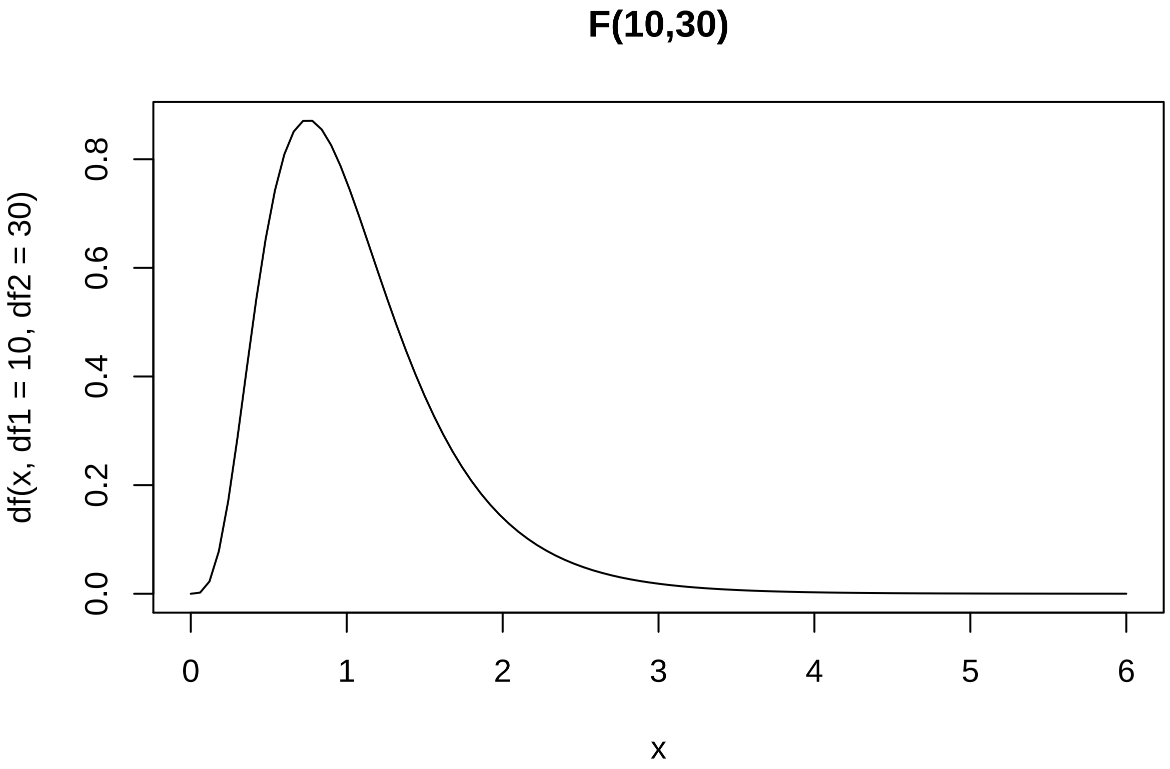
The F distribution (2)

```
curve(df(x, df1 = 3, df2 = 30), from = 0, to = 6, main = "F(3,30)")
```



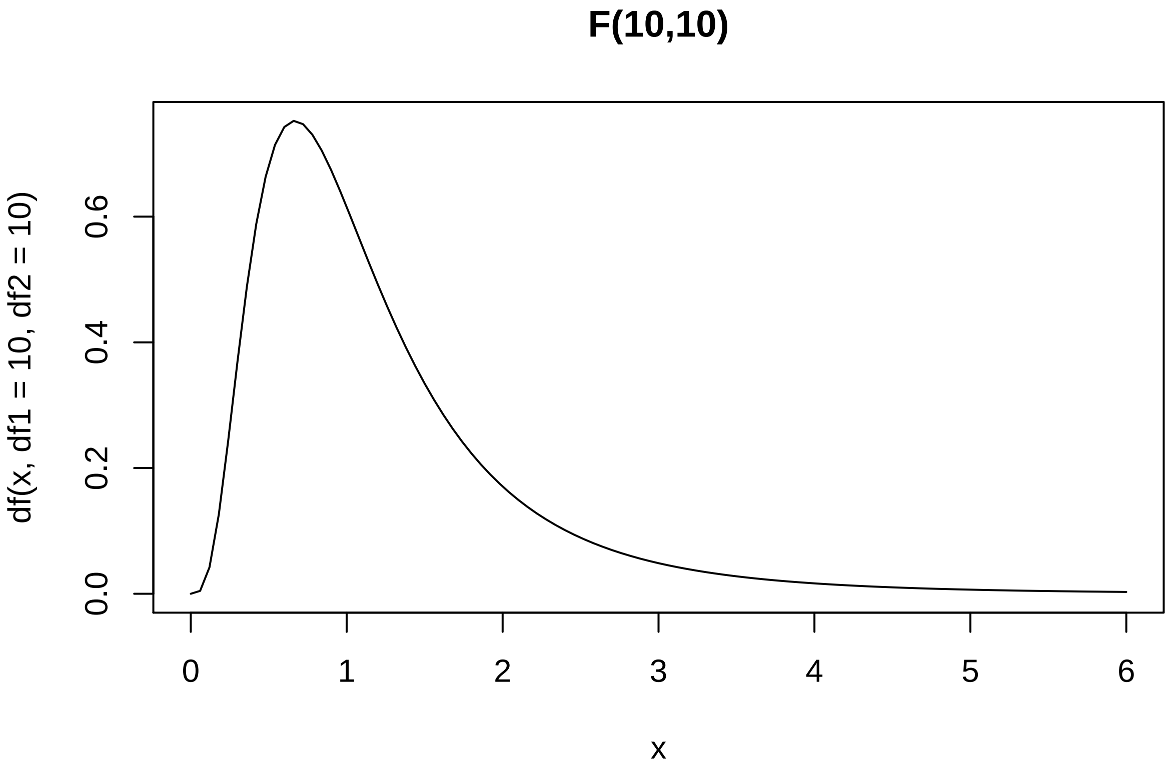
The F distribution (3)

```
curve(df(x, df1 = 10, df2 = 30), from = 0, to = 6, main = "F(10,30)")
```



The F distribution (4)

```
curve(df(x, df1 = 10, df2 = 10), from = 0, to = 6, main = "F(10,10)")
```



Finally finishing that ANOVA

- So, how extreme is our F-value given the H_0?
- Let's get the p-value for our F-value

```
(p_value <- 1 - pf(F_value, df1 = df_model, df2 = df_error))
```

```
## [1] 0.4521
```

It's 1 - pf since pf gives you the probability for an F value to be smaller than the value given. We want the opposite. If $p \leq .05$, the effect of group is significant (spuriously in this case, since we know there was no effect).

Can we just let R do it?

- Remember the formula format for the t-test command:

```
summary(aov(formula = iq ~ group, data = iqdata))
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## group        2    381     191    0.82   0.45
## Residuals   27   6293     233
```

- `summary()` is necessary here to get the p-value
- This is just a convention in r

A different way, using a package

```
library(ez)
# ezANOVA wants a subject identifier, so let's generate one
# every row is a different subject, so we just need to make a new variable that numbers the rows
iqdata$subnum <- 1:30
```

ezANOVA

```
(iq_anova <- ezANOVA(iqdata, dv = iq, wid = subnum, between = group))
```

```
## Warning: Converting "subnum" to factor for ANOVA.
```

```
## $ANOVA
##   Effect DFn DFd      F      p p<.05     ges
## 1  group   2  27 0.8177 0.4521       0.05711
##
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd   SSn  SSd      F      p p<.05
## 1   2  27 52.68 1887 0.3769 0.6895
```

- Nice. What's `ges` and what is that second output?

Effect size

- `ges` = generalised eta squared
- $\eta^2 = \frac{SS_{model}}{SS_{total}}$
- η^2_G is η^2 with a correction for repeated measures designs (if applicable; we'll talk about those later)
- η^2 is an estimate of the relationship between variance explained by the ANOVA model and total variance in the data
- Compare this to Cohen's d, another estimate of effect size:
- $d = \frac{\bar{x_1} - \bar{x_2}}{s}$
- Cohen's d is an estimate of how large a difference in means is (in standard deviations)
- What is s? The standard deviation of the difference in means (from your sample)

Levene's test for homogeneity of variance

```
iq_anova[2] # get the second element of the list iq_anova
```

```
## $`Levene's Test for Homogeneity of Variance`
##   DFn DFd   SSn  SSd      F      p p<.05
## 1   2  27 52.68 1887 0.3769 0.6895
```

- When you perform an ANOVA, you make the assumption that the variances within each group are similar
- For example, the IQs *within* the BU group should not be more variable than the IQs *within* the Soton and the Oxford groups
- Levene's test compares the group variances
    - What's the test statistic for a test that compares variances? It's F!
    - The p-value tells you if the variances are significantly different between groups.

Levene's test for homogeneity of variance

- You do not need Levene's test if you only have two independent groups (and ezANOVA won't do it then)
- If Levene's test is not significant, all is well
- If Levene's test is significant, you're violating the homogeneity of variance assumption
- Not a big issue if the sample sizes are equal for all groups (balanced design)
- If sample sizes aren't equal (unbalanced design) and the larger groups have higher variance, your ANOVA loses power
- If sample sizes aren't equal and the larger groups have lower variance, your ANOVA becomes anti-conservative (α increases)!

Alternative: Bartlett's test

```
bartlett.test(formula = iq ~ group, data = iqdata)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  iq by group
## Bartlett's K-squared = 0.5702, df = 2, p-value = 0.7519
```

- Bartlett's test is more sensitive to deviations from normality than Levene's test
- It's up to you which one you want to use (with ezANOVA, Levene's test is more convenient)

What to do if Levene's or Bartlett's tests are significant

- If your group sizes are equal, nothing to worry about.
- If not:
- Calculate the variance for each group and see if you're dealing with just a power issue or an α issue
- If the largest group variance is less than 4 times the smallest group variance, you're good.
- If you have huge variance differences and there might be an α issue:
    - Easiest solution: Fix the sample size issue (e.g. run more participants)!
    - Use linear mixed models (LMMs; more on that later)
    - Use specialised tests (this is SPSS's approach):
    - Welch's t-test (R has this test as `oneway.test()`)
    - Brown-Forsythe instead of ANOVA (R has this test in the `lawstats` package)

  - Post-hoc tests:
    - Games-Howell for unequal variance
    - Hochberg's GT2 for non-equal sample sizes

The SPSS approach to statistics

- Throw as many obscure tests at the problem as you can
- This is a sales strategy: "We need to buy SPSS since no other program has the Games-Howell test!"
- In reality, the standard ANOVA is remarkably robust to all but the most extreme violations of its assumptions
- Specialised tests often come at a huge cost in terms of power
- This doesn't mean that you shouldn't test the assumptions
- But a simplistic strategy where you run one type of test if the assumption test is significant and another one if it isn't is not helpful
- Take a good look at your data
  - Be aware of potential issues
  - Interpret the data accordingly.
  - Only use specialised and non-parametric tests as a last resort if your data massively violate the assumptions

Just so we're clear

- Inflated α is not harmless
- But "researcher degrees of freedom" inflate α much more than all but the most extreme assumption violations
- Stopping rules (test after every X participants, then stop as soon as you have a significant result)
- Failing to report non-significant conditions
- Failing to correct for multiple comparisons
- Don't let SPSS (or over-cautious textbooks) discourage you from running plain, simple ANOVAs
- Be honest and transparent about your data and how you collected them and you'll be fine.

Reporting a one-way ANOVA

- Let's walk through it together
- I stole this data set from Andy Johnson
- We are investigating the effect of swearing on pain tolerance (see Stephens et al., 2009)
- Three groups: continuous use of swear word, neutral word, or no word whilst hand in cold water (DV = time until participant can't stand the pain and pulls hand from water)

```
# open the data file (available on GitHub; don't forget to use setwd to set the working directory to wh
# setwd("C:/my_data/")
pain <- read.csv("pain.csv")
```

Get an idea of what's in the file

```r
head(pain)
```

```
##   X Swear.Condition Time.In.Cold.Water
## 1 1      Swear Word                 55
## 2 2      Swear Word                 64
## 3 3      Swear Word                 70
## 4 4      Swear Word                 60
## 5 5      Swear Word                 54
## 6 6      Swear Word                 66
```

```r
str(pain)
```

```
## 'data.frame':    60 obs. of  3 variables:
##  $ X                 : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Swear.Condition   : Factor w/ 3 levels "Neutral Word",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Time.In.Cold.Water: int  55 64 70 60 54 66 48 72 63 65 ...
```

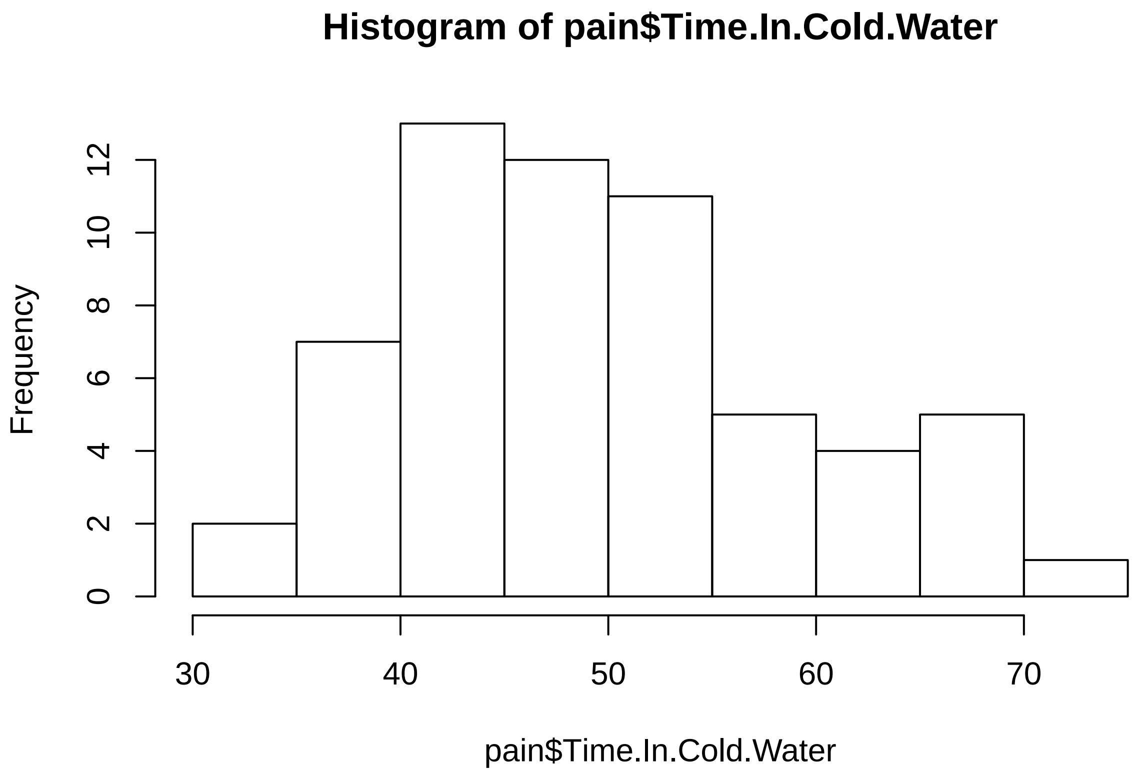
Is the design balanced?

```r
table(pain$Swear.Condition)
```

```
## 
## Neutral Word        Quiet   Swear Word 
##           20           20           20
```

How is the DV distributed?

```r
hist(pain$Time.In.Cold.Water)
```



Is the DV normal?

- Note: The raw DV does not have to be perfectly normal!
- In fact, your alternative hypothesis is that it's not, since it is influenced by the treatment effect!
- The error should be normal, though.
- That is, the residual data that you get when you take the treatment effect out should be normal.
- I'll show you how to test that in a second.

Calculate means and SDs

```
library(ez)
# for ez, we need a subject identifier so the function can tell if
# it's a between or within or mixed subject design

# nrow(pain) gives you the number of observations.
# Each observation is a subject
# 1:nrow(pain) gives you a vector with 1,2,3,...,number of observations

pain$subject <- factor(1:nrow(pain))

(pain_stats <- ezStats(data = pain,
                       dv = Time.In.Cold.Water,
                       wid = subject,
                       between =  Swear.Condition))
```

```
##   Swear.Condition  N  Mean    SD  FLSD
## 1    Neutral Word 20 44.70 6.906 4.231
## 2           Quiet 20 45.75 5.711 4.231
## 3      Swear Word 20 60.45 7.323 4.231
```

Calculate means and SEs

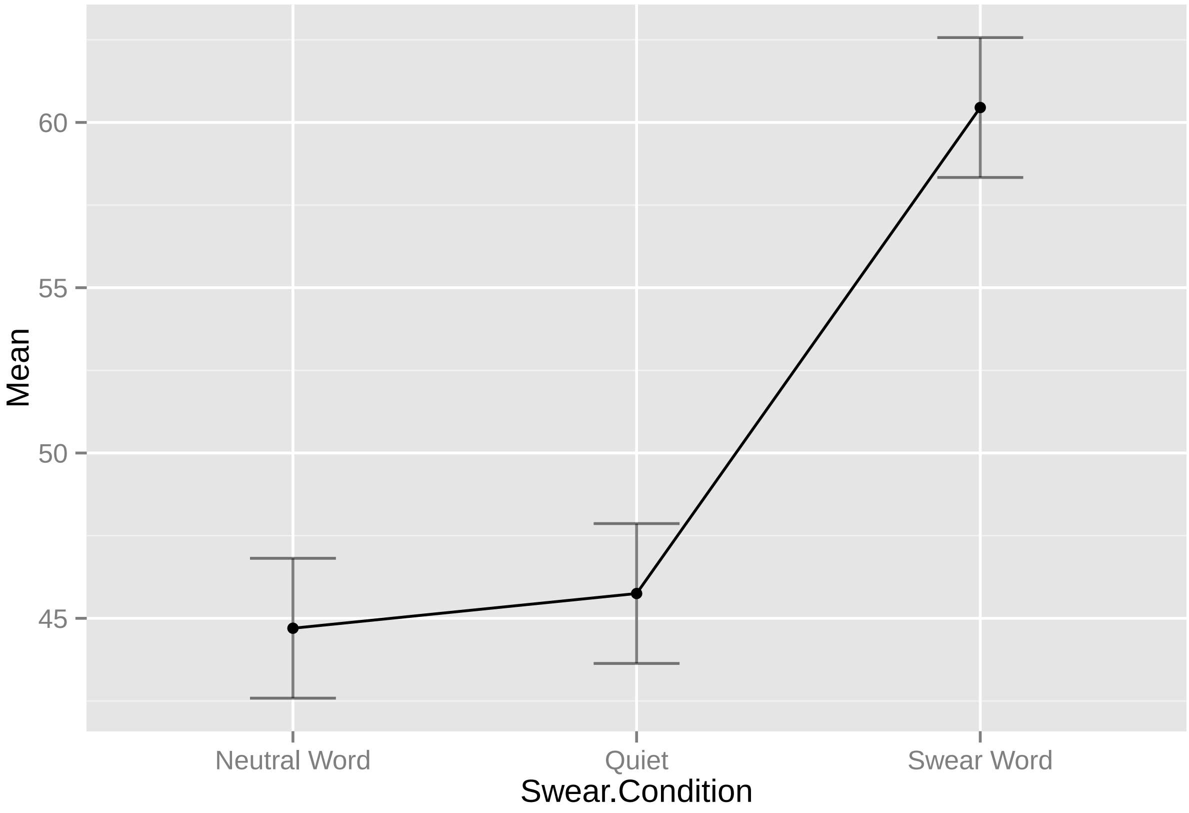
```
pain_stats
```

```
##   Swear.Condition  N  Mean    SD  FLSD
## 1    Neutral Word 20 44.70 6.906 4.231
## 2           Quiet 20 45.75 5.711 4.231
## 3      Swear Word 20 60.45 7.323 4.231
```

- This gives you a nice overview of the group sizes, group means, and group variance/
- N is the group size. It's the same for each group, so your design is balanced.
- The mean is almost the same for the neutral and the quiet conditions, but higher in the swear word condition.
- SDs are quite similar, so it doesn't seem like the homogeneity of variance assumption is an issue
- Fisher's Least Significant Difference (FLSD): gives you an idea of the minimum effect size that would be significant in a t-test between any of these groups. This is just an estimate, not a replacement for the ANOVA and a post-hoc test.
- Still useful to see which differences might be significant and which definitely aren't.

Plot means

```
ezPlot(data = pain,
       dv = Time.In.Cold.Water,
       wid = subject,
       between = Swear.Condition,
       x = Swear.Condition)
```



Notes about the plot

- This plot is pretty good (better than anything SPSS gives you), but it is not perfect
- Discrete factors should be plotted as bars, not lines
- The error bars are Fisher's LSD, which is helpful in interpreting the differences, but unusual
- In a plot for publication, you would also want to change the axis labels
- I will give you a better function to use later
- For now, this is perfectly fine

Perform the ANOVA

```
pain_anova <- ezANOVA(data = pain,
                      dv = Time.In.Cold.Water,
                      wid = subject,
                      between = Swear.Condition,
                      return_aov = TRUE)
```

- Note the `return_aov = TRUE`.
- We need that because we want to look at the residuals later.

Look at the results

```
pain_anova$ANOVA
```

```
##            Effect DFn DFd     F         p p<.05    ges
## 1 Swear.Condition   2  57 34.74 1.367e-10     * 0.5493
```

- Looks like there is a significant effect of swear condition.

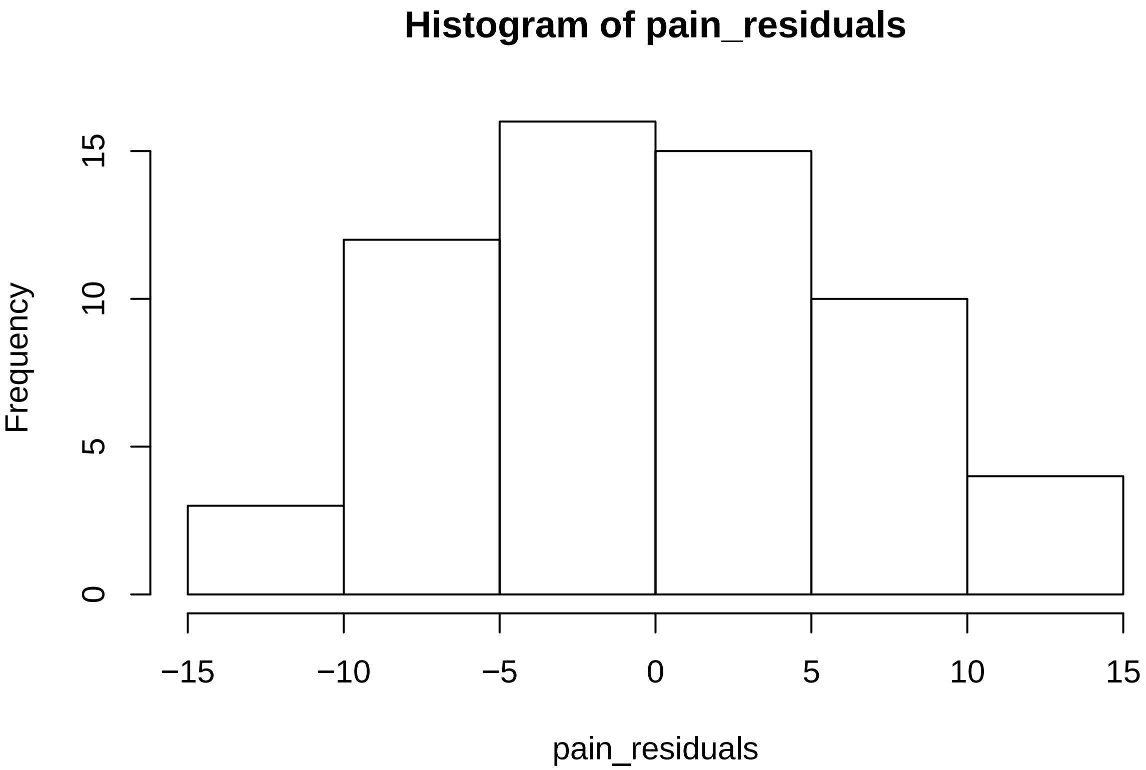
```
pain_anova$"Levene's Test for Homogeneity of Variance"
```

```
##   DFn DFd   SSn   SSd     F      p p<.05
## 1   2  57 33.23 817.7 1.158 0.3213
```

- No issues with homogeneity of variance.

Visually examine residuals

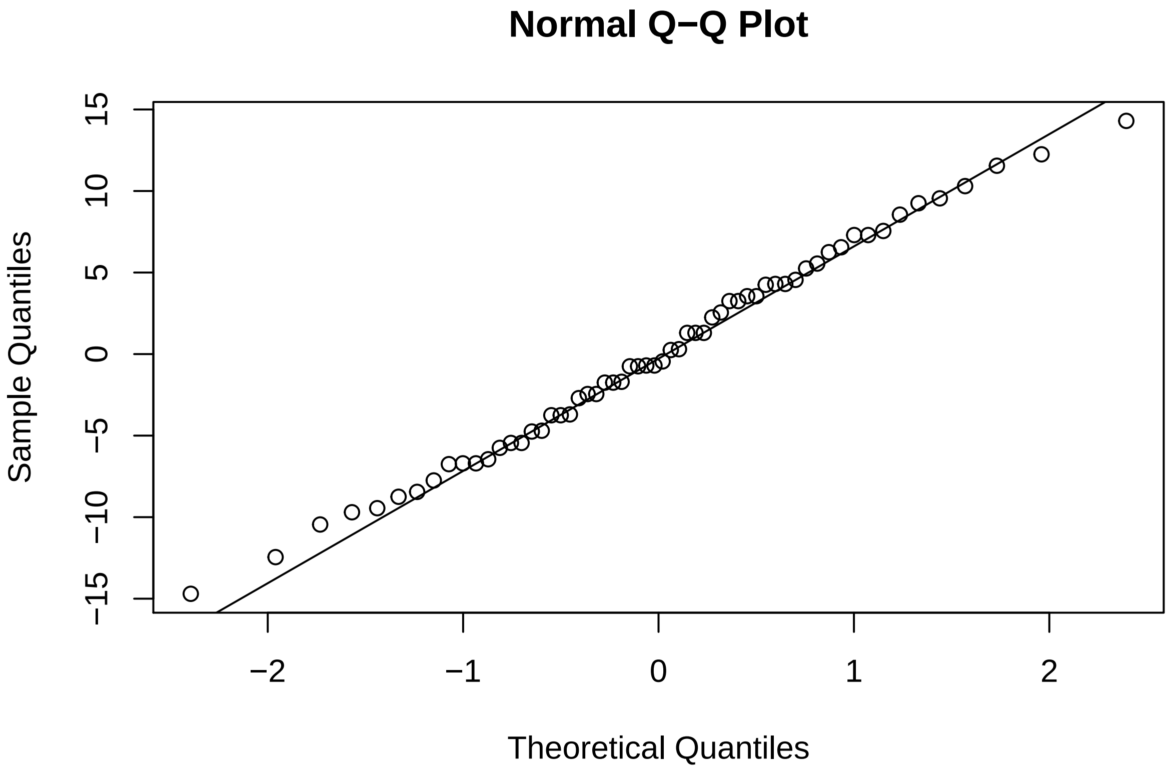
```
pain_residuals <- resid(pain_anova$aov)
hist(pain_residuals)
```



- Looks quite normal.

Quantile-Quantile Plot

```
qqnorm(pain_residuals)
qqline(pain_residuals)
```



- Plots the distribution of the variable in question against a hypothetical perfect normal distribution - Deviations from the line are deviations from normality - Our residuals are very slightly deviated. Is this an issue? Probably not.

Formally test normality of residuals

- Shapiro-Wilk normality test

```
shapiro.test(pain_residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  pain_residuals
## W = 0.9918, p-value = 0.9596
```

- Not significant at all. No issues with non-normality.

Perform post-hoc pairwise comparisons

```r
(pain_pairwise_p_values <- with(pain, pairwise.t.test(x = Time.In.Cold.Water,
                                                       g = Swear.Condition,
                                                       pool.sd = FALSE,
                                                       p.adj = "bonferroni")))
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  Time.In.Cold.Water and Swear.Condition
##
##            Neutral Word Quiet
## Quiet      1            -
## Swear Word 7.6e-08      7.9e-08
##
## P value adjustment method: bonferroni
```

- These are the p-values for all the possible t-tests (Bonferroni correction applied)

Perform post-hoc pairwise comparisons

- Better: Use Holm's correction instead of Bonferroni's correction for higher power

```r
(pain_pairwise_p_values <- with(pain, pairwise.t.test(x = Time.In.Cold.Water,
                                                       g = Swear.Condition,
                                                       pool.sd = FALSE,
                                                       p.adj = "holm")))
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  Time.In.Cold.Water and Swear.Condition
##
##            Neutral Word Quiet
## Quiet      0.6          -
## Swear Word 7.6e-08      7.6e-08
##
## P value adjustment method: holm
```

Perform post-hoc pairwise comparisons

- If you have no issues with homogeneity of variance, you can use the same pooled sd for all the tests:

```r
(pain_pairwise_p_values <- with(pain, pairwise.t.test(x = Time.In.Cold.Water,
                                                       g = Swear.Condition,
                                                       pool.sd = FALSE,
                                                       p.adj = "holm")))
```

```
##
##  Pairwise comparisons using t tests with non-pooled SD
##
## data:  Time.In.Cold.Water and Swear.Condition
##
##            Neutral Word Quiet
## Quiet      0.6          -
## Swear Word 7.6e-08      7.6e-08
##
## P value adjustment method: holm
```

Alternative: Tukey's HSD

```
TukeyHSD(x = pain_anova$aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = formula(aov_formula), data = data)
##
## $Swear.Condition
##                          diff    lwr    upr  p adj
## Quiet-Neutral Word       1.05 -4.035  6.135 0.8731
## Swear Word-Neutral Word 15.75 10.665 20.835 0.0000
## Swear Word-Quiet        14.70  9.615 19.785 0.0000
```

- Another way of correcting for multiple comparisons. Assumes equal variance.

Now: report it

I'll show you how in Homework 3.