# 1st Summer School of the Institute for Language, Communication and the Brain

## Applied mathematics, statistics and networks - Courses 2, 3 and 4 support

Bernard Giusiano

September 4-6, 2018 - Marseille, France

## Table of Contents



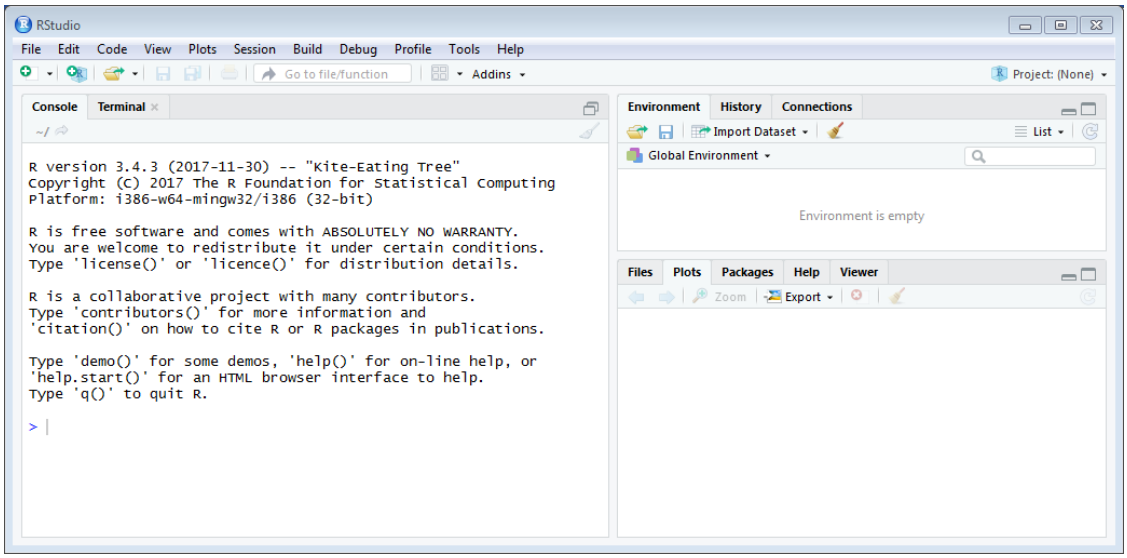
This course presents the basic principles of statistical inference (estimation, mean comparison, variance analysis and linear regression) as well as a practical introduction to the R language. It corresponds to Bernard Giusiano's classes on Tuesday, Wednesday and Thursday.

# PART I

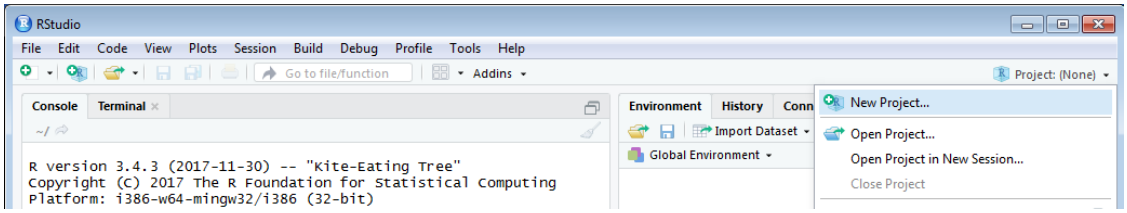
## Installation and handling

### R and RStudio installation

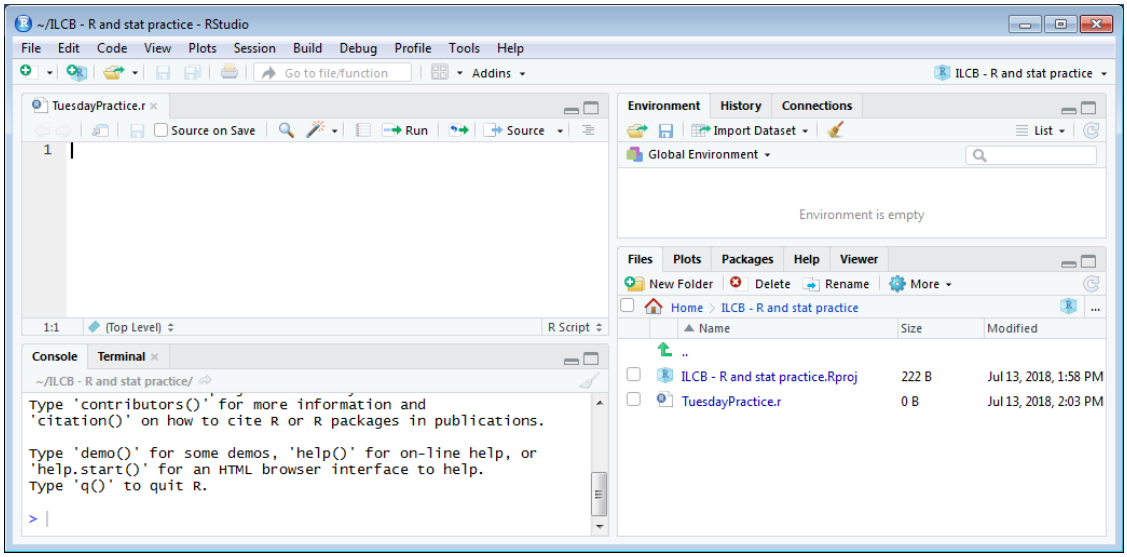
The first thing to do is the installation of the **R language** and **the RStudio development environment**.
Get the correct R download for your Operating System from The Comprehensive R Archive Network. Then get Rstudio (Open Source License) for your OS from Download RStudio.
After launching RStudio, you should see a window that looks like this:




Click on **Project** on the far right. Choose **New Project**, then **New Directory**, and again **New Project**:




Choose "*ILCB - R and stat practice*" as **Directory name**, for example. Then choose **File**, **New File**, **R Script**. And save (disk icon or File menu) this "*Untitled1*" script under the name you want. Here is the Rstudio window after these actions:




The RStudio window is divided into four panels:
1. top left: the code editor, with the possibility of having several open scripts (different tabs),
2. bottom left: the output console, where you can also directly enter R instructions (so, it's an input/output console !),

3. top right: the environment shows your variables and their values,
4. bottom right: your project files, your plots, etc. (cf. the tabs).

## First steps

Type *Hello!* in the console without quotation marks, then <ENTER>.
The answer appears in red under what you just typed: *Error: object 'Hello' not found*
Try *"Hello!"* with quotation marks. The answer: *[1] "Hello!"*. R recognized an object of type string.

You can type *"Hello!"* in the first line of your R script in the code editor. And save (disc icon), which allows you to correct without retyping and to keep what you have written. To run what is in the script, put the cursor anywhere in the line you want to run, then click **Run** at the top right of the script window. You can also execute all the contents of a selected script or set of lines: cf. items of the **Code** menu.

Good. Now let's start really!

Imagine that we have the anatomical MRI of seven subjects from which we measured the volume of their brain (in ml): 1424 1467 1215 1338 1139 1308 1229. And we ask ourselves: what is the mean volume of these seven brains?

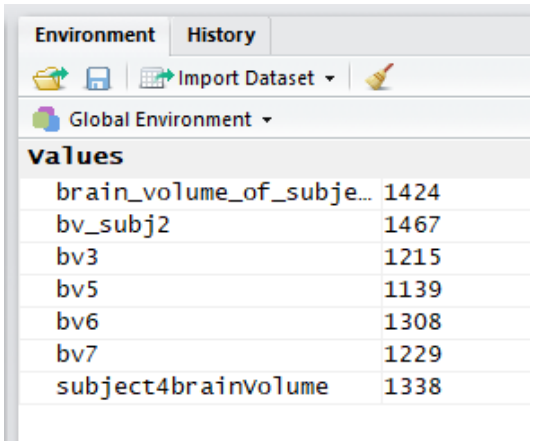
```
# We can calculate the mean in the following way, directly in R:
# Notice how to insert comments in the R code, with the # character
(1424 + 1467 + 1215 + 1338 + 1139 + 1308 + 1229) / 7
```

```
## [1] 1302.857
```

We can put all these numbers in *variables*.

```
brain_volume_of_subject_1 = 1424
bv_subj2 = 1467
bv3 <- 1215
subject4brainVolume <- 1338
bv5 = 1139
bv6 <- 1308
bv7 <- 1229
```

Look in the upper right panel: the variables we just created are there.



You can see that a variable can have a more or less long name, more or less significant. A valid variable name consists of letters, numbers and the dot or underline characters. The variable name starts with a letter or the dot not followed by a number.
You can also see that two signs can be used indifferently to indicate the assignment of a value to a variable: the sign "**=**" and the sign "**<-**". The latter is the historical sign of the assignment in R (cf. https://www.tutorialspoint.com/r/r_variables.htm).

But with these fanciful names variables our code will not be very elegant.

```
(brain_volume_of_subject_1 + bv_subj2 + bv3 + subject4brainVolume + bv5 + bv6 +
bv7) / 7

## [1] 1302.857
```

When we have several values of the same measurement, of the same type of observation, it is advisable to group these values in a *vector* to which we can give an evocative name.

```
brainVol <- c(1424, 1467, 1215, 1338, 1139, 1308, 1229)
brainVol

## [1] 1424 1467 1215 1338 1139 1308 1229
```

**c()** is a R function that combines multiple values into a vector or list. For help with this function, type **help (c)** or **? c**

Most functions available in R apply directly to structured data such as vectors, but also matrices, lists or data frames. These functions are developed in large numbers by R users and widely exchanged between users. This has been the success of this language in the community of statisticians and, more broadly, in that of researchers who use statistical methods.

```
n <- length(brainVol)
meanOfBrainVol <- sum(brainVol) / n
meanOfBrainVol

## [1] 1302.857

# or more simply:
mean(brainVol)

## [1] 1302.857
```

The mean is classically used to characterize a set of numbers. But consider another group of 7 numbers that have the same mean:

```
brainVol2 <- c(1805, 1376, 1200, 1654, 524, 620, 1941)
mean(brainVol2)

## [1] 1302.857
```

These values are very different from the first ones. A central tendency parameter such as the mean is not sufficient in itself to characterize a set of values; it is necessary to give also an idea of the dispersion of the values by using the standard deviation for example.

```
sd(brainVol)
```

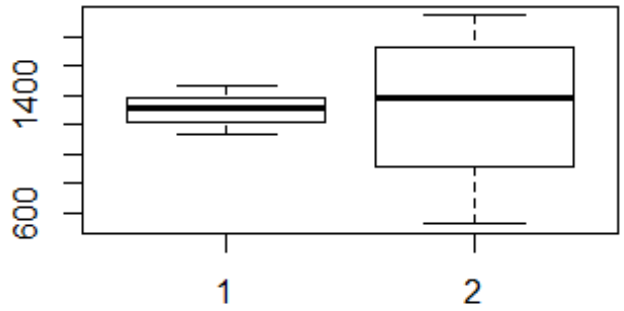
```
## [1] 117.5463
```

```
sd(brainVol2)
```

```
## [1] 558.4975
```

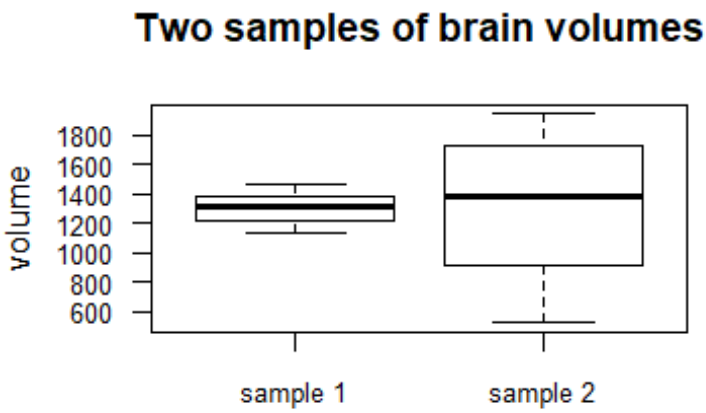
A boxplot chart shows how these two sets of data are different:

```
boxplot(brainVol,brainVol2)
```




Let's improve this figure (**? boxplot**, then click on **Graphical Parameters** to know more):

```
boxplot(brainVol, brainVol2, main="Two samples of brain volumes", names=c("sample
1","sample 2"), cex.axis=0.8, las=1, ylab="volume")
```




**summary ()** is a useful function to quickly describe the data ...

```
summary(brainVol)
```

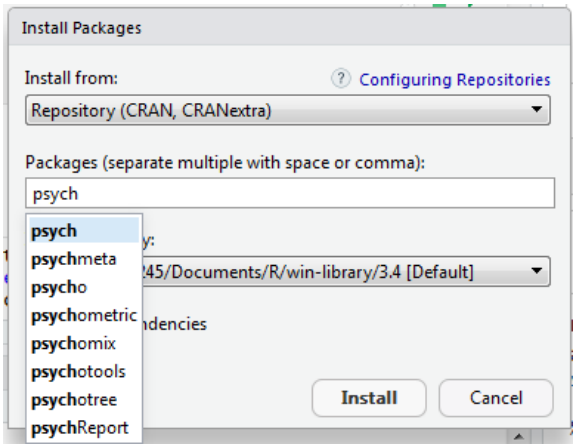
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1139    1222    1308    1303    1381    1467
```

```
summary(brainVol2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     524     910    1376    1303    1730    1941
```

... but it does not offer the standard deviation. If you search the Internet for an R function to better describe your data as **summary()**, you will surely reach the Quick-R site which is a real mine for beginners of R.
Check for yourself: Descriptive Statistics on Quick-R.
On this page you will find how to use the **describe** function of the **psych** package.

Before you can tell RStudio that you want to use the psych package, you must download this package. Choose the item **Install Packages ...** from the **Tools** menu and type *psych* in the appropriate field. This download on your machine is to be done once. Then, when you want to use the functions of this package in one of your scripts, just place **library(psych)** at the beginning of the script.




```
library(psych)
describe(brainVol)

##    vars n    mean     sd median trimmed    mad  min  max range skew
## X1    1 7 1302.86 117.55   1308 1302.86 137.88 1139 1467   328 0.07
##    kurtosis    se
## X1    -1.69 44.43

describe(brainVol2)

##    vars n    mean    sd median trimmed    mad min  max range skew kurtosis
## X1    1 7 1302.86 558.5   1376 1302.86 636.04 524 1941  1417 -0.3    -1.78
##        se
## X1 211.09
```

In R, a package is a set of functions around a theme shared by one of the many contributors who make this language live. It's easy to find one or more packages on the Internet to meet your needs. The Comprehensive R Archive Network is an interesting starting point for exploring packages, especially from its CRAN Task Views page.

## Exercices

- Find help on mean, sd, summary, describe, boxplot, ...
- Look for the differences between the data structures of R (vector, matrix, lists, data.frame):
  - Basic: Data Types on Quick-R
  - Stronger: Data structure on First Steps in R

  - Even stronger: Data Structures and Data Types
- Take a look at the features offered by the many packages available on the CRAN website.

### Brain break

A Marseille story: the man with the brain ribbon. WIRED headline: "Brain Not Necessary for French Civil Service Worker"

## Estimation and principles of statistical tests

### Population and sample

Researchers wanted to see if the brain volume was different in some psychiatric diseases, for example Martin Reite and col. in a BMC Psychiatry paper in 2010.

Inspired by these authors, suppose that we want to make a comparison between the brain volume of control subjects and the brain volume of patients with schizophrenia. To do such a comparison, it is conventional to compare the mean volume in the control population with the mean volume in the schizophrenic population. Unfortunately, it is impossible to get the brain volume of all control subjects like that of all schizophrenics in the world. We must limit ourselves to a few people for whom we have an MRI that allows us to calculate the volume: we therefore have only one *sample* for each of the two *populations* that we want to compare. Will the conclusions we draw from the comparison between these two samples be valid for the two populations?

We will see under what conditions this extrapolation, this *inference* is possible.

Let's leave aside the very important question of the quality of a sample and rather start with an experimental exploration. Nevertheless, here are some tips on how to extract a good sample from a population:
* Design, data analysis and sampling techniques for clinical research
* Brain imaging results skewed by biased study samples
* Sampling bias

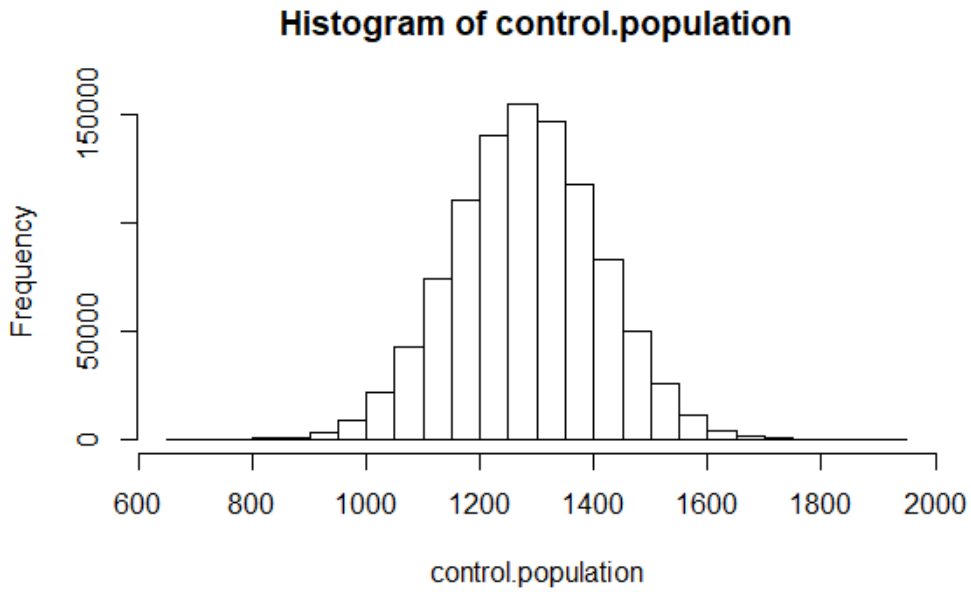
Thanks to R and our computer, we can create a population of one million people (and more if your RAM is big enough). We will draw samples from this simulated population to understand the magic of the estimate.

The volume of the brain is a random variable that most likely follows a normal distribution, that is, the histogram of the values of this volume follows a Gauss bell curve. A normal distribution is entirely defined by its mean and its standard deviation. In the article mentioned at the beginning of this chapter, we find values for these two parameters for the control subjects.

```
control.mean <- 1281
control.sd <- 128
```

Let's randomly generate 1,000,000 brain volume values from control subjects with the function **rnorm(n, mean = m, sd = s)** that is part of the default loaded stats package (see here and here). And let's trace the histogram.

```r
set.seed(427)    # to initialize random generator at the same value each time
control.population <- rnorm(1000000, mean = control.mean, sd = control.sd)
hist(control.population)
```

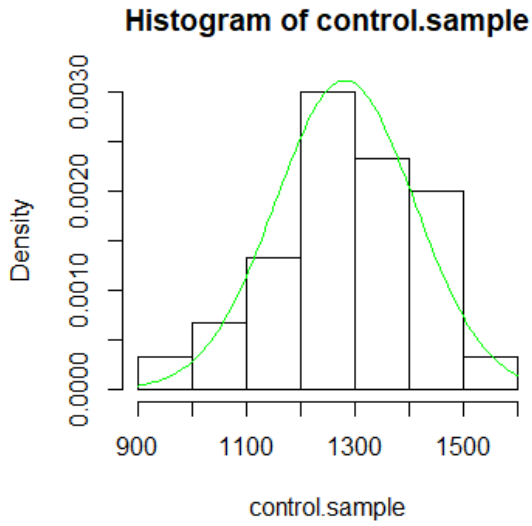



Let's check the mean and standard deviation of this population:

```r
mean(control.population)

## [1] 1281.17

sd(control.population)

## [1] 128.0894
```

These values are not quite what we expected. Is this an effect of the random generation algorithm? Whatever, let's continue. Let's randomly draw a size 30 sample from this population...

```r
control.sample <- sample(control.population,30)
hist(control.sample, prob=TRUE)  # with relative frequence (density) in y
# draw a green line to show the population distribution shape
curve(dnorm(x, mean(control.population), sd(control.population)), add=TRUE,
col="green")
```

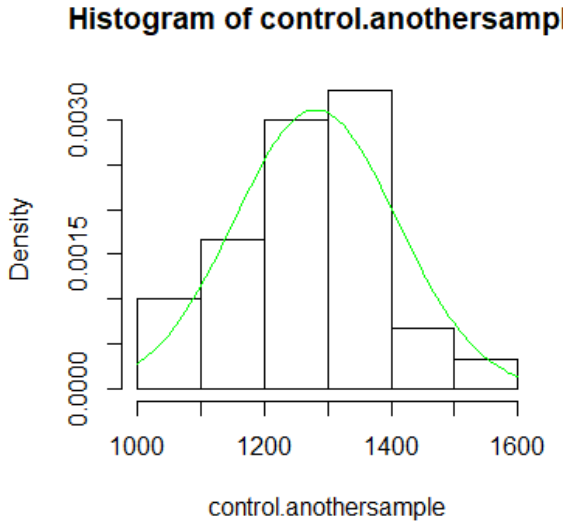



... and calculate its parameters.

```
mean(control.sample)

## [1] 1282.222

sd(control.sample)

## [1] 126.3743
```

Let's draw another sample of the same size.

```
control.anothersample <- sample(control.population,30)
hist(control.anothersample, prob=TRUE)
curve(dnorm(x, mean(control.population), sd(control.population)), add=TRUE,
col="green")
```



```r
mean(control.anothersample)
```

```
## [1] 1271.475
```

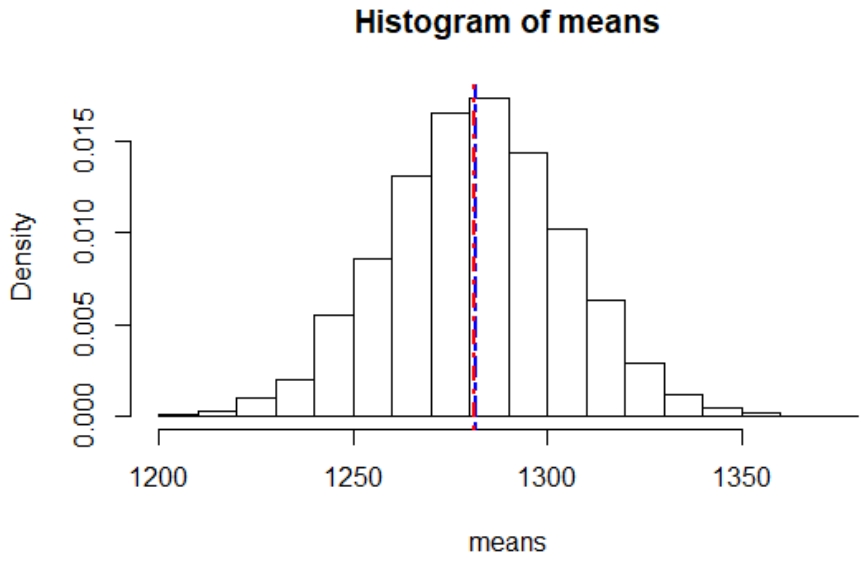
```r
sd(control.anothersample)
```

```
## [1] 123.6121
```

We can conclude that the parameters of a random sample with a size of 30 individuals (in the sense of statistical units) are not identical to the parameters of the population from which this sample is derived. But we are going to see that the parameters of the sample are not so far from those of the population.

Let's randomly draw 10,000 samples of size 30, calculate the mean of each and put these 10,000 means in the vector **means**. It takes a little while ... but it would be much longer to do it by hand!

```r
means <- numeric(10000)                          # initialize a vector of numeric values
for (i in 1:10000){                              # loop 10,000 times
  means[i] <- mean(sample(control.population,30))  # draw a sample, calculate mean, put it in vector
}
hist(means, prob=TRUE)                           # draw histogram
means.mean <- mean(means)                        # the mean of means
abline(v=means.mean, lwd=2, lty=6, col="blue")   # draw in blue the mean of means of the 10,000 samples
abline(v=control.mean, lwd=2, lty=4, col="red")  # mark in red the place of the population mean
```

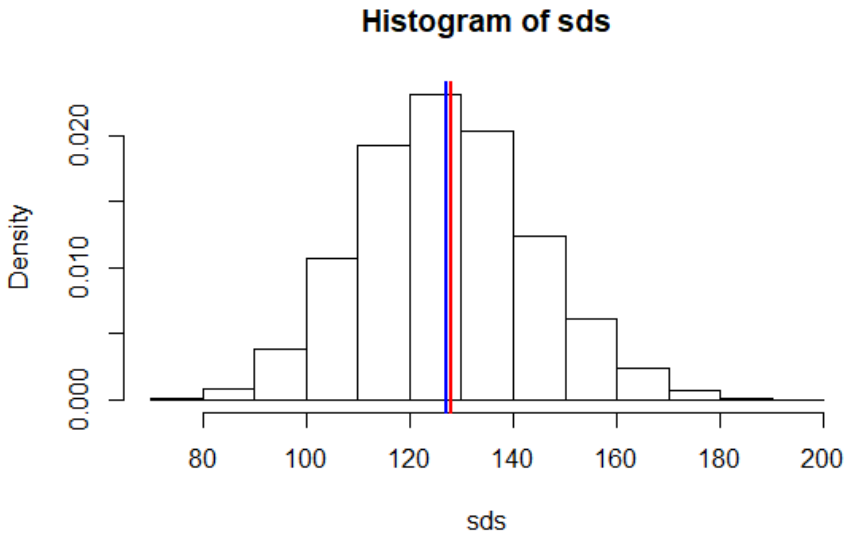



Surprise! The mean of the means of the 10,000 samples (blue) is equal to the mean of the population (red) from which these samples are drawn. As a result, it is said that the mean of a sample is a *good estimator* of the mean of the population from which it is drawn.

In fact, *means.mean*, the mean of the 10,000 sample means, is probably slightly different from the population mean. A good estimator is an estimator whose *mathematical expectation* (the mean over a large number of samples) *tends towards* the parameter of the population.

The same approach with standard deviation (or variance) shows us that the standard deviation (or variance) of a sample is not a good estimate of the standard deviation (or variance) of the population from which he is drawn. It is necessary to use the Bessel's correction.

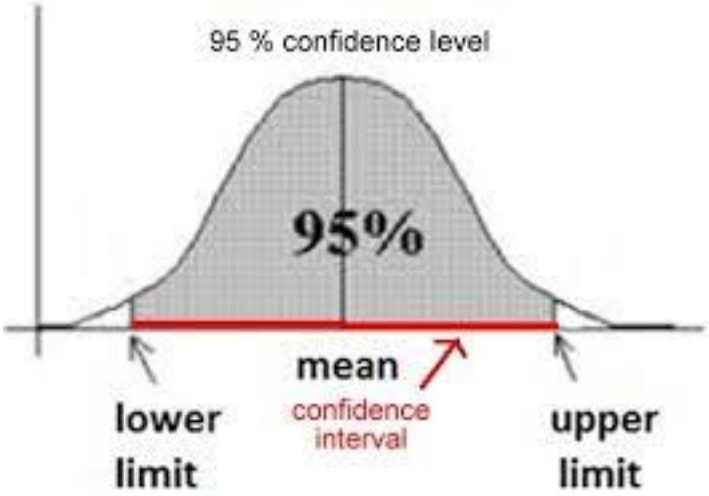
```
sds <- numeric(10000)
for (i in 1:10000){
  sds[i] <- sd(sample(control.population,30))
}
hist(sds, prob=TRUE)
sds.mean <- mean(sds)   # mathematical expectation of sample standard deviation
abline(v=sds.mean, lwd=2, lty=1, col="blue")
abline(v=control.sd, lwd=2, lty=1, col="red")
```




## Confidence interval

Consider the figure "Histogram of means" which shows the distribution of the means of 10,000 random size 30 samples drawn from the same population. This distribution shows the *sampling fluctuations* of the mean.

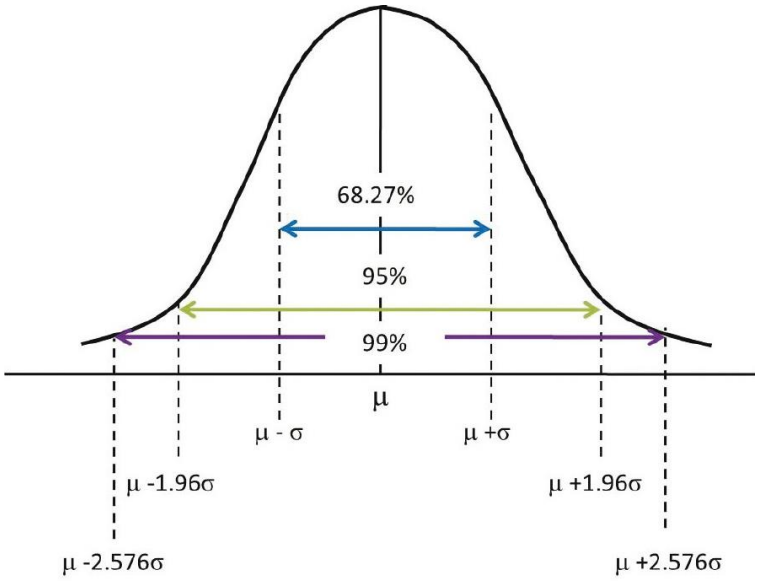
If we take at random one of these samples we could pull out a sample whose mean is equal to 1200, a value quite far from that of the population mean (1281). Chance could also cause us to get a sample whose mean is equal to or near the mean of the population. Estimating the mean of a population using the mean of a sample is a *point estimate*. To specify the credibility of this estimate, we use the *confidence interval*, which is an *interval estimate*.




*http://www.learningaboutelectronics.com/Articles/Confidence-interval-calculator.php*

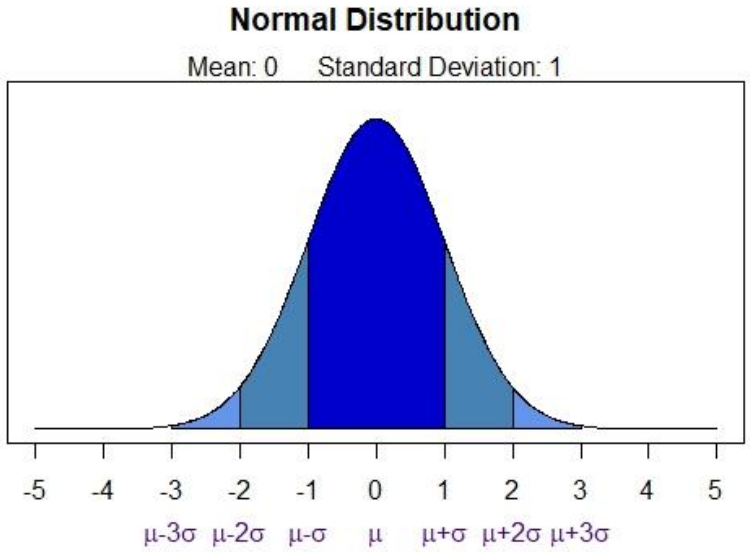
Watch out for the interpretation of the confidence interval! For a 95% interval, for example, we can say "there is a 95% probability that the calculated confidence interval of some future experimentation encompasses the true value of the population parameter." Read the confidence interval page on Wikipedia for more details. And see the difference with prediction interval.

To calculate the confidence interval of a parameter, that is, the lower bound and the upper bound for a chosen probability, it is necessary to rely on the distribution of the parameter. For example, to calculate the confidence interval of the mean, it is necessary to rely on the distribution of the means of the samples that can be drawn from the population. However, if the samples are large enough (n >= 30), their means are normally distributed regardless of the distribution of the original variable (for us, the volume of the brain).

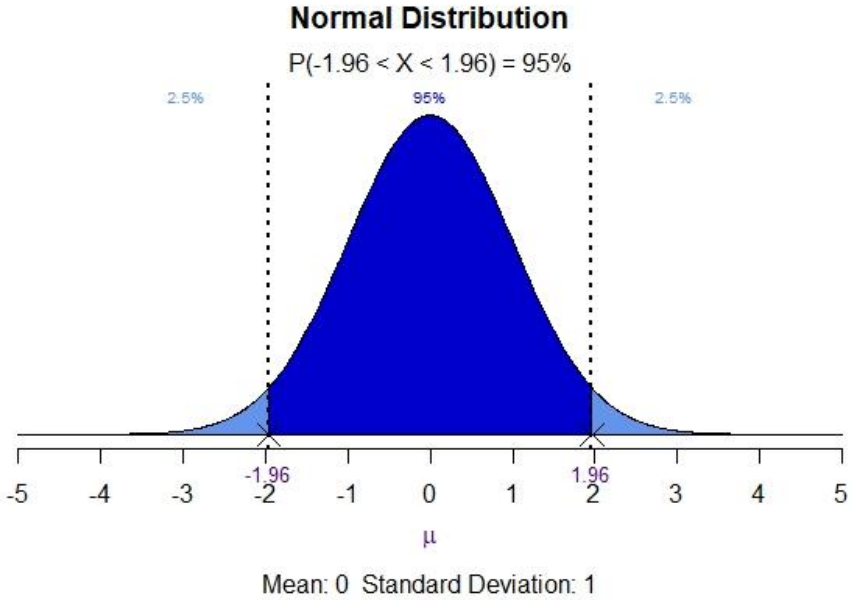



Look at the figure above: in a normal distribution, 68.27% of the values are between $\mu - \sigma$, i.e. the mean minus the standard deviation, and $\mu + \sigma$; 95% of the values, hence the area under the curve, lie between $\mu - 1.96\sigma$ and $\mu + 1.96\sigma$ ; and 99% of the values are between $\mu - 2.576\sigma$ and $\mu + 2.576\sigma$ . This is true regardless of the normal distribution, that is, regardless of its mean and standard deviation.

But where do these bounds 1, 1.96 and 2.576, come from? They are often called *z-values* and are calculated on the *standard normal distribution* (reduced centered normal distribution), which is the normal distribution with mean 0 and standard deviation 1.




Classically, tables have been used to find the relationship between different bound values and different percentile values calculated from the standard normal distribution (to which any normal distribution can be reduced, see here). For example, below, the probability of randomly drawing a value between -1.96 and +1.96 in a set of values normally distributed with an mean of 0 and a standard deviation of 1, is 95%.




With R, you do not need these tables: the **qnorm()** function returns the bound value for a given percentile. Caution: qnorm() gives P(X < bound) and not P(-bound < X < +bound).

```r
qnorm(0.95)
```

```
## [1] 1.644854
```

```r
qnorm(1-0.05/2)   # 2.5% on each side out of the interval
```

```
## [1] 1.959964
```

Let's go back to the confidence interval of the mean that we want to calculate on a sample drawn at random in our population of control subjects. We put ourselves in the realistic case where we only have access to a sample of the population, that is to say forgetting that we have access to all the individuals of the population thanks to the simulation.

Here, our statistical unit is no longer the volume of a brain but the mean volume of a brain sample. We therefore need the mean of the sample means as well as the standard deviation of these means.

We have seen that the mean of the sample means tends towards the mean of the population from which they are drawn. And that the mean of a sample is a good estimator of the mean of the population. So we will use it.

If we start from a single sample, and therefore with only one mean value, how can we calculate the standard deviation of the population of means of samples of the same size that can be drawn from the original population? Commonly referred to as the standard error of the mean, this standard deviation of the means is obtained by dividing the standard deviation of the sample by the square root of the sample size.

```r
n = 30
mysample <- sample(control.population,n)   # draw a sample
mysample.mean <- mean(mysample)
mysample.sd <- sd(mysample)
means.mean <- mysample.mean        # a good estimate
means.sd <- mysample.sd / sqrt(n)  # the standard error of the mean
means.mean.ci_lower <- means.mean - 1.96 * means.sd  # 2.5% lower
means.mean.ci_upper <- means.mean + 1.96 * means.sd  # 2.5% upper
sprintf("Recall: true mean of population : %.2f", control.mean)
```

```
## [1] "Recall: true mean of population : 1281.00"
```

```r
sprintf("Mean estimate of the population : %.2f", means.mean)
```

```
## [1] "Mean estimate of the population : 1270.83"
```

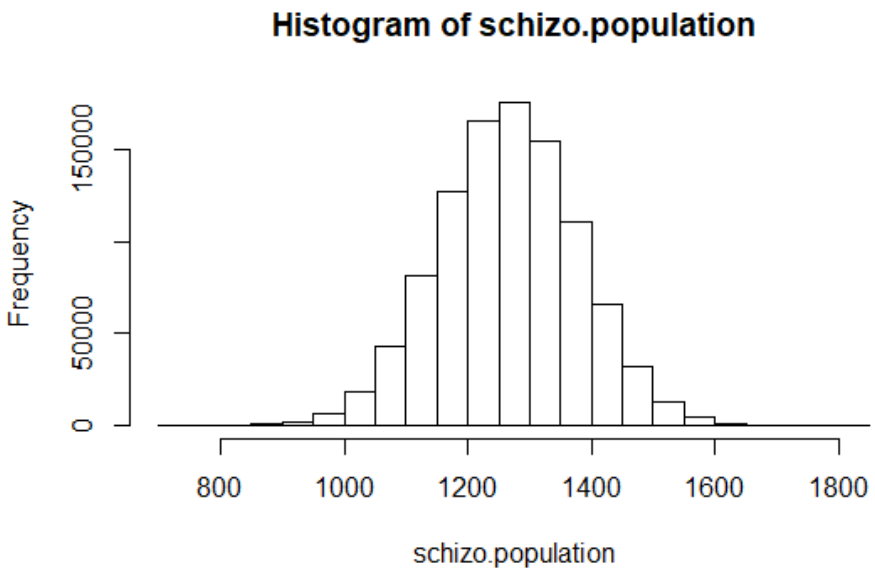
```r
sprintf("95%% confidence interval of this mean : (%.2f, %.2f)",
means.mean.ci_lower, means.mean.ci_upper)
```

```
## [1] "95% confidence interval of this mean : (1229.95, 1311.72)"
```

## Mean comparison test

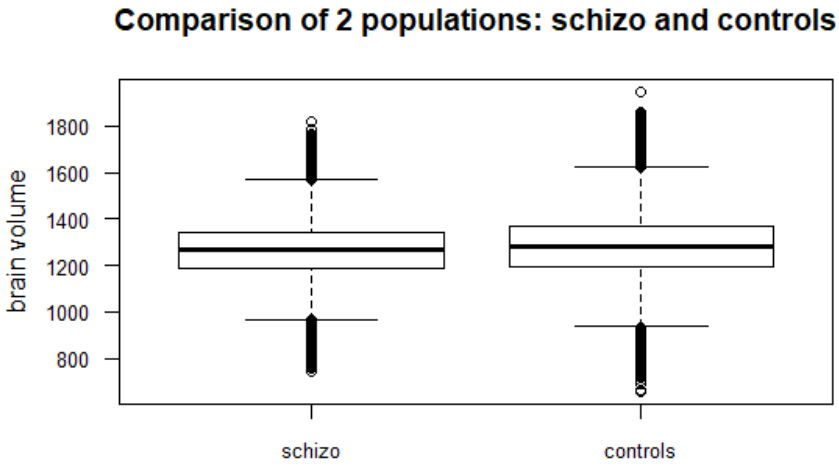
Remember that we wanted to compare the brain volume of control subjects with the brain volume of patients with schizophrenia. Let's generate the population of schizophrenics, or at least the values of the volume of their brain based on the paper mentioned above.

```
schizo.mean <- 1266
schizo.sd <- 112
schizo.population <- rnorm(1000000, mean = schizo.mean, sd = schizo.sd)
hist(schizo.population)
```




Let's compare the two distributions:

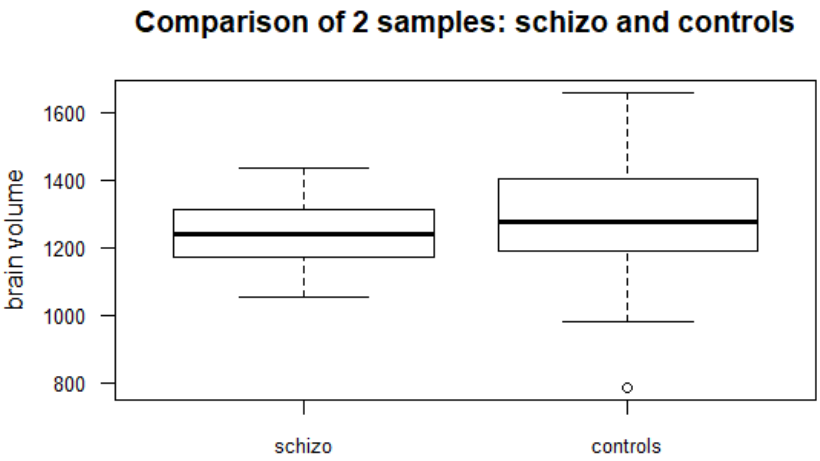
```
boxplot(schizo.population, control.population, main="Comparison of 2 populations:
schizo and controls", names=c("schizo","controls"), cex.axis=0.8, las=1,
ylab="brain volume")
```




On the boxplot above the two populations are very similar. Yet we know that their true means are not equal: 1266 for schizo, 1281 for controls.

Let us remember these values and put ourselves in the real context where only two samples are available to us.

```
n = 30
mycontrol <- sample(control.population,n)
myschizo <- sample(schizo.population,n)
boxplot(myschizo, mycontrol, main="Comparison of 2 samples: schizo and controls",
names=c("schizo","controls"), cex.axis=0.8, las=1, ylab="brain volume")
```




To calculate the estimate of the mean and its confidence interval more simply than before, we will use the **CI** function of the **Rmisc** package.

```
library(Rmisc)

## Warning: package 'Rmisc' was built under R version 3.3.3

## Loading required package: lattice

## Loading required package: plyr

CI(myschizo, 0.95)

##    upper     mean    lower
## 1277.983 1240.585 1203.188

CI(mycontrol, 0.95)

##    upper     mean    lower
## 1347.313 1278.330 1209.346
```

The means of the two samples are not equal. Since we are not supposed to know the true means of the two populations, we can ask ourselves if this inequality is not only due to sampling fluctuations. Maybe if we randomly draw two more samples, their means will be closer ... or farther away?

The difference between the two estimates is:

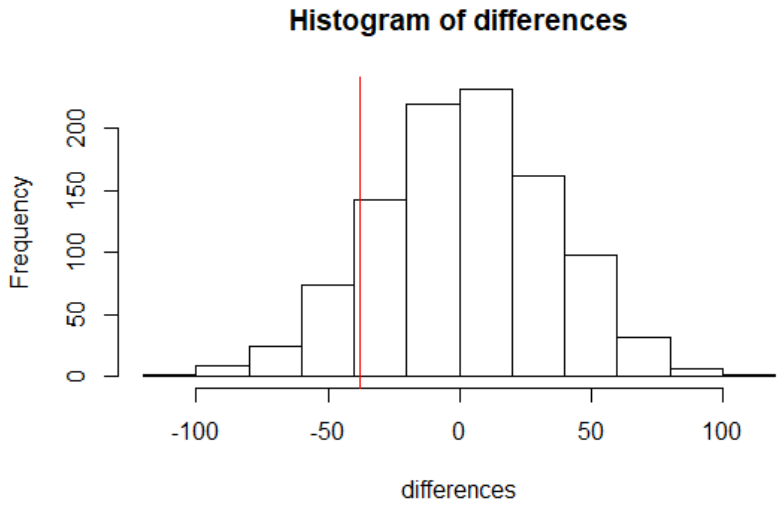
```
mydiff <- mean(myschizo) - mean(mycontrol)
mydiff

## [1] -37.74431
```

Is it possible that this difference *mydiff* is due only to sampling fluctuations? What is the probability that this is the case?

Well, to find out, we will use the simulation again.

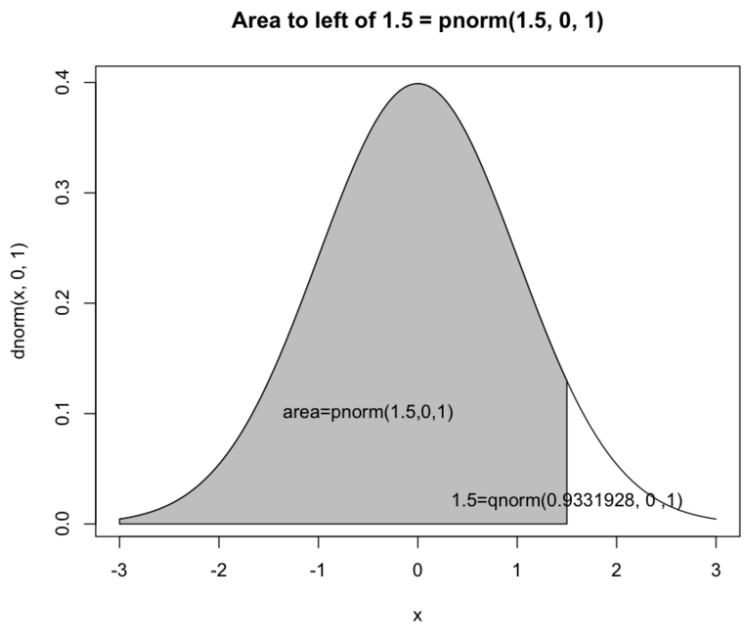
Let's assume that we have two identical populations, that's our *null hypothesis*. So their means are equal. If we draw a large number of pairs of samples, one in one population, the other in the second, and calculate the difference between the two means of each pair, what is the probability that this difference is greater, in absolute value, to our previous result?

```
n <- 30                          # the size of the samples
differences <- numeric(1000)     # loop on 1000 rather than 10000 for speed
for (i in 1:1000){
  # To simply simulate two identical populations,
  # we can use the same population twice!
  differences[i] <- mean(sample(control.population,n)) -
mean(sample(control.population,n))
}
hist(differences)
abline(v=mydiff, col="red")      # mark the place of mydiff with red vertical line
```




We can see that in the case of the null hypothesis, where the two populations have the same mean, the mean of the differences between the two means of the pairs of samples is equal to 0, but the distribution spreads largely around this value 0. Let's calculate the probability that the difference is smaller (one-sided test) or that the absolute value of the difference is larger (two-sided test).

The **pnorm()** function returns the probability than a value is less than a bound value.




```
# one-sided test
prob1 <- 1 - pnorm(abs(mydiff), mean = mean(differences), sd=sd(differences))
prob1

## [1] 0.1402023

# symmetrical two-sided test
prob2 <- 2 * (1 - pnorm(abs(mydiff), mean = mean(differences), sd=sd(differences)))
prob2

## [1] 0.2804046
```

The probability of obtaining a difference value smaller than or equal to -37.744 simply because of sampling fluctuations is therefore 14.02%. It's safer to consider the bilateral case: the probability of falling on an absolute value greater than or equal to 37.744, whether on one side or the other of the mean, is 28.04%.

Apply this approach to our original question with our two samples *mycontrol* and *myschizo*.

For this, whether to read the probability corresponding to the difference in a standard normal distribution table or to use the R *pnorm()* function, we need the difference between the two sample means, the mean of (the population of) the differences and the standard deviation of (the population of) the differences. The last two are needed to center and reduce the difference between the means of the two samples and make them a standardized value, i.e. a z-score.

1. Difference between the two sample means: *mydiff*
2. Mean of (the population of) the differences: 0 in the null hypothesis.
3. Standard deviation of (the population of) the differences: it is the square root of the sum of the variances of the two populations, that we can calculate using estimates.

```
n <- 30                                          # recall: sample size
mydiff <- mean(myschizo) - mean(mycontrol)       # already calculated
meanOfDifferences <- 0                           # by definition of null
```

```
hypothesis
myschizo.var <- var(myschizo)                          # var() gives the variance
mycontrol.var <- var(mycontrol)
varOfControl_estimate <- mycontrol.var * n / (n-1)   # Bessel's correction
varOfSchizo_estimate <- myschizo.var * n / (n-1)
varOfControlMeans_estimate <- varOfControl_estimate/n  # from variance of volume to
variance of means of volume
varOfSchizolMeans_estimate <- varOfSchizo_estimate/n
varOfDifferences_estimate <- varOfControlMeans_estimate +
varOfSchizolMeans_estimate
sdOfDifferences_estimate <- sqrt(varOfDifferences_estimate)
mydiff.zscore <- (mydiff - meanOfDifferences) / sdOfDifferences_estimate
mydiff.zscore
```

```
## [1] -0.967246
```

What is the probability (the famous **p**) of reaching or exceeding this standardized absolute value of difference in the null hypothesis?

```
p <- 2 * (1 - pnorm(abs(mydiff.zscore), mean = 0, sd = 1))
p
```

```
## [1] 0.3334211
```

Personally, I would not take the risk of saying that the two populations have a different mean in view of this difference calculated on my two samples: the risk of being wrong is 33%!

Without further evidence in favor of the mean difference of the two populations, I keep the null hypothesis ... knowing that I may be wrong: if such a difference may be due to sampling fluctuations alone, could also be due to a real mean difference between the two populations.

In general, the decision threshold (*α risk*) is set at 5%. In our case, p > threshold, we maintain the null hypothesis of equality of the means of the two populations.

What would have been obtained with much larger samples? Let's try with samples of size 2000. *We give these samples the number 2 so as not to get confused in variables that would have the same name in two different experiments.*

```
n2 = 2000
mycontrol2 <- sample(control.population,n2)
myschizo2 <- sample(schizo.population,n2)
mydiff2 <- mean(myschizo2) - mean(mycontrol2)
meanOfDifferences <- 0
myschizo2.var <- var(myschizo2)
mycontrol2.var <- var(mycontrol2)
varOfControl_estimate2 <- mycontrol2.var * n2 / (n2-1)
varOfSchizo_estimate2 <- myschizo2.var * n2 / (n2-1)
varOfControlMeans_estimate2 <- varOfControl_estimate2 / n2
varOfSchizolMeans_estimate2 <- varOfSchizo_estimate2 / n2
varOfDifferences_estimate2 <- varOfControlMeans_estimate2 +
varOfSchizolMeans_estimate2
sdOfDifferences_estimate2 <- sqrt(varOfDifferences_estimate2)
```

```
mydiff2.zscore <- (mydiff2 - meanOfDifferences) / sdOfDifferences_estimate2
mydiff2.zscore

## [1] -3.440148

p2 <- 2 * (1 - pnorm(abs(mydiff2.zscore), mean = 0, sd = 1))
p2

## [1] 0.000581396
```

Hey! Now, p < 5%! We can reject the null hypothesis and adopt the alternative hypothesis: the means of the two populations are different.

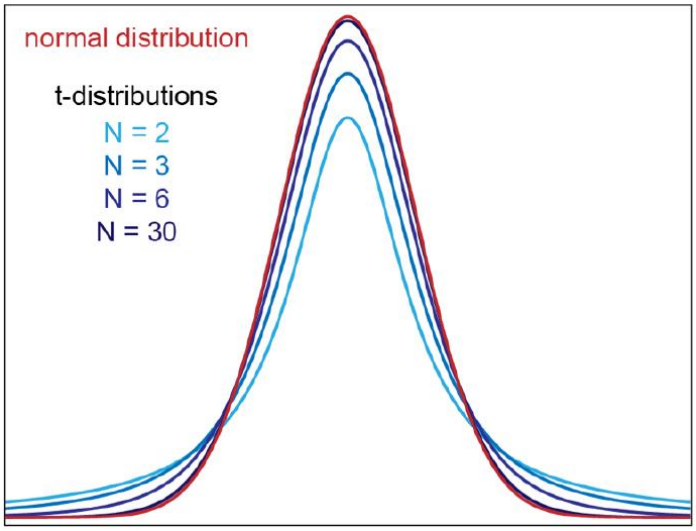
Note that we already knew it since we ourselves made the populations control and schizo with different means. However, the difference was too small compared to the dispersion (variance) of the data for small size samples (n=30) to highlight it. With these small samples, we had fallen into *β risk*.

| | | Truth (unknown) | |
|---|---|---|---|
| | | $H_0$ true | $H_A$ true |
| Decision taken on the basis of the result of the statistics (z-score) | $H_0$ true | No error | $\beta$ risk |
| | $H_A$ true | $\alpha$ risk | No error |

In the article that inspired our example, the sample sizes were 58 for schizophrenics and 89 for control subjects: *"In conclusion this study, although including a sizeable number of subjects, failed to demonstrate statistically significant differences in total brain volume between the three major groups of severe mental illness studied". "Although absolute raw data indicated brains in male schizophrenic subjects were about 3% smaller than control brains, this failed to reach formal statistical significance."*

To understand the importance of sample size and the concepts of power of a statistical test and effect size, look here and here. A very good software to help you calculate the size of your samples to highlight a difference is G*Power.

If we draw samples smaller than 30, the distribution of means will not necessarily be distributed normally, especially if the original variable is not normally distributed. We will no longer be able to rely on the standard normal distribution to calculate the probability p necessary for the decision of the comparison test. In this case we must rely on Student's t law, according to which the distribution depends not only on the mean and the standard deviation but also on the number of degrees of freedom (related to the size of the population).




From a size of 30, the Student's t distribution merges with the normal distribution. This explains why one does not find in the basic package of R a function to calculate the comparison test based on the standard normal distribution. In general, we use the **t.test()** function based on Student's t.

```
t.test(myschizo,mycontrol)

##
##  Welch Two Sample t-test
##
## data:  myschizo and mycontrol
## t = -0.98378, df = 44.691, p-value = 0.3305
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -115.03329   39.54467
## sample estimates:
## mean of x mean of y
##  1240.585  1278.330

t.test(myschizo2,mycontrol2)

##
##  Welch Two Sample t-test
##
## data:  myschizo2 and mycontrol2
## t = -3.441, df = 3929.9, p-value = 0.0005856
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -20.633671  -5.655187
## sample estimates:
## mean of x mean of y
##  1266.163  1279.307
```

## Exercices

- Search on Internet how to calculate the confidence interval of a percentage. If you cannot find the function in R, program it!

- Is it reasonable for a surgeon to say that he gets more than 50% good results when he operated on 11 patients and 5 patients died?
- Watch this video on YouTube to understand how to compare percentages. Rest assured, there is a function in R.

## Brain break

Read R's news on R-Bloggers, the blog of R blogs. Look at the "Most visited articles of the week" section in the right side.