t-tests

NRES 710

Fall 2020

Download the R code for this lecture!

To follow along with the R-based lessons and demos, right (or command) click on this link and save the script to your working directory

Overview of statistical methods

Okay before we delve further into t-tests, let's first give a broad overview of the analyses we will cover in this class (at least a tiny bit) and in which cases they are most appropriate:

Continuous response variable

If your response variable is continuous (ratio, interval), common classical parametric statistical analyses include: t-tests, ANOVA, linear regression. Each of these tests is associated with non-parametric alternatives. In each case, we are interested in testing if the *population mean* of the continuous response variable is affected by the predictor variables (null hypothesis: nope, there is no effect!). Predictor variables, in turn, can be continuous or categorical (factor variables in R).

Continuous response, categorical predictor If your categorical predictor is binary (two levels), you can use a **two-sample t-test**. The non-parametric alternative is the **Mann-Whitney test**.

Example: Are females larger than males? (null hypothesis: no difference)

If your categorical predictor has more than two levels, you can use an **ANalysis Of Variance (ANOVA)** followed by *pairwise comparisons* to test which categories differ from one another. You can visualize these relationships with (e.g.) a boxplot. The non-parametric alternative is the **Kruskal-Wallis test**.

Q: What is the predictor variable for a one-sample t-test?

Continuous response, continuous predictor If your predictor variable is continuous, you can use **linear regression**. If you just want to know if two variables are correlated but you are not interested in modeling one (the response) as a function of another (the predictor) you can run a **Pearson correlation test**. The non-parametric alternative is a **Spearman correlation test**.

Example: What is the relationship between tree diameter and age- can tree diameter be used to effectively predict the age of a tree?

Continuous response, both continuous and categorical predictors In this case you can use **Analysis of Covariance (ANCOVA)** or just use **multiple linear regression**. In truth, linear regression and ANOVA/ANCOVA are two sides of the same coin. You can run both analyses using the workhorse of linear modeling in R, the 'lm' function. Basically, if you have a continuous response variable, you can use the 'lm' function! The non-parametric alternative might be something like a **regression tree** analysis.

Discrete (count) response variable

With a discrete count response, you can use the same techniques as with a continuous response OR you can use **generalized linear models** with a Poisson error distribution (or other discrete probability distribution).

Categorical response variable

If your response variable is categorical (factor variable, ordinal variable, binary variable) then your choice of statistical methods changes:

Categorical response variable, categorical predictor If both your response variable and your predictor variable are categorical, then you can use a **chi-squared test** or a **Fisher exact test** to test for an association between the two variables. You may first summarize your data as a *contingency table* like we did with the 'lady tasting tea' example.

Example: test for an association between salamander color morph (melanistic vs wild-type) and mating behavior ('sneaker' vs territory holder).

Categorical response variable, continuous predictor If your response variable is binary (true/false, two levels) and your predictor variable is continuous, you can use **logistic regression** (which is a type of *generalized linear model (GLM)*).

If your categorical response variable has more than two levels, you can use **multinomial logistic regression** (for categorical responses) or **ordinal logistic regression** (for ordinal responses).

Overview: t-tests

Okay, now let's move to the main topic of this lecture: t-tests! Here we have a continuous response variable and either no predictor variable (one sample t-test) or a binary predictor variable (two-sample t-test).

We have already performed simple versions of the t-test (one-sample t-tests). But t-tests are quite flexible and can be applied to a wide variety of null-hypothesis testing scenarios:

One-sample t-test

A one-sample t-test tests the consistency of the sample data with the null hypothesis that the sample was generated from a population with a specified mean (often zero, but not necessarily). As we have seen before, the t-statistic in this case is expressed as:

$t = \frac{(\bar{x} - \mu_0)}{s.e.}$

Where $\bar{x}$ is the sample mean, $s.e.$ is the standard error of the mean, and μ_0 is the population mean under the null hypothesis.

The t-statistic can be interpreted as the difference between the sample mean and the null population mean in units of standard errors.

Under the null hypothesis, the sampling distribution of the t-statistic should follow a t distribution with n-1 degrees of freedom.

```
### one sample t-test (paired t-test is a type of one sample t-test)

sample.data <- rgamma(10,2,.1)
null.mean <- 10
```

```r
sample.size <- length(sample.data)
sample.mean <- mean(sample.data)
sample.sd <- sd(sample.data)
std.err <- sample.sd/sqrt(sample.size)
t.stat <- (sample.mean-null.mean)/std.err

t.crit <- abs(qt(0.025,sample.size-1))   # for 2-tailed test

p.val <- (1-pt(abs(t.stat),sample.size-1))*2   #


### alternatively use the t.test function:

t.test(sample.data,mu=null.mean)   # should get the same answer!
```

```
##
##  One Sample t-test
##
## data:  sample.data
## t = 2.156, df = 9, p-value = 0.05945
## alternative hypothesis: true mean is not equal to 10
## 95 percent confidence interval:
##   9.597101 26.772528
## sample estimates:
## mean of x
##  18.18481
```

Paired t-test A common version of the one-sample t-test is the 'paired t-test', in which a measurement is taken on a single individual before and after a treatment is applied. An example of a paired t-test is the 'weight loss drug' example from the 'basic concepts' lecture. In the weight-loss drug' example, we measured patients before and after taking a drug and measured the total weight change in each patient. Even though we have two measurements per patient (repeated measures!) we collapse the data for each patient into a single number (difference in weight) before running our t-test.

Two-sample t-test

A two-sample t-test tests the consistency of the sample data with the null hypothesis that sample A was generated from the same underlying population as sample B. The t-statistic in this case is expressed as:

$t = \frac{(\bar{x_A} - \bar{x_B})}{s.e._{pooled}}$

Where $\bar{x}$ is the sample mean, $s.e._{pooled}$ is the pooled standard error of the mean across both samples, and μ_0 is the population mean under the null hypothesis.

The t-statistic in the 2-sample case can be interpreted as the difference between the sample mean from population A and the sample mean from population B, in units of (pooled) standard errors.

The formula for the pooled standard deviation depends on whether your sample size is equal in the two samples and whether you are able to assume that the standard deviation is the same or similar in the two underlying populations.

For example, the formula for the pooled standard deviation when we assume equal standard deviation in the two population but we have unequal sample size is:

$\sqrt{((N_1 - 1) * \sigma_1^2 + (N_2 - 1) * \sigma_2^2)/(N_{pooled} - 2)}$

For the 2-sample t-test, the degrees of freedom is equal to the total sample size across both samples minus 2.

```
### two sample t-test

sample.data.1 <- rnorm(15,55,10)
sample.data.2 <- rnorm(10,45,10)

sample.size.1 <- length(sample.data.1)
sample.size.2 <- length(sample.data.2)
sample.size.pooled <- length(sample.data.1) + length(sample.data.2)

sample.mean1 <- mean(sample.data.1)
sample.mean2 <- mean(sample.data.2)

sample.sd1 <- sd(sample.data.1)
sample.sd2 <- sd(sample.data.2)
sample.sd.pooled <- sqrt(((sample.size.1-1)*sample.sd1^2 + (sample.size.2-1)*sample.sd2^2)/(sample.size

std.err.pooled <- sample.sd.pooled*sqrt(1/sample.size.1+1/sample.size.2)
t.stat <- (sample.mean1-sample.mean2)/std.err.pooled

t.crit <- abs(qt(0.025,sample.size-1))   # for 2-tailed test

p.val <- (1-pt(abs(t.stat),sample.size.pooled-2))*2   # 2-tailed test



### alternatively use the t.test function:

t.test(sample.data.1,sample.data.2,var.equal = T)   # should get the same answer!
```

```
##
##  Two Sample t-test
##
## data:  sample.data.1 and sample.data.2
## t = 2.2062, df = 23, p-value = 0.03763
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##   0.5157696 16.0247347
## sample estimates:
## mean of x mean of y
##  55.64815  47.37790
```

Q What are the key assumptions of the 2-sample t-test?

Q What is the null hypothesis for the 2-sample t-test?

One tailed vs two-tailed tests Tailed-ness is a common point of confusion when running t-tests and z-tests, so it's worth taking a little time to review.

It all has to do with what we mean by 'more extreme than the observed test statistic' (from the definition of a p-value).

In a one-tailed test in which our (alternative) hypothesis is that our sample mean is *greater than* than the null hypothesis, we mean 'more positive than the observed test statistic'.

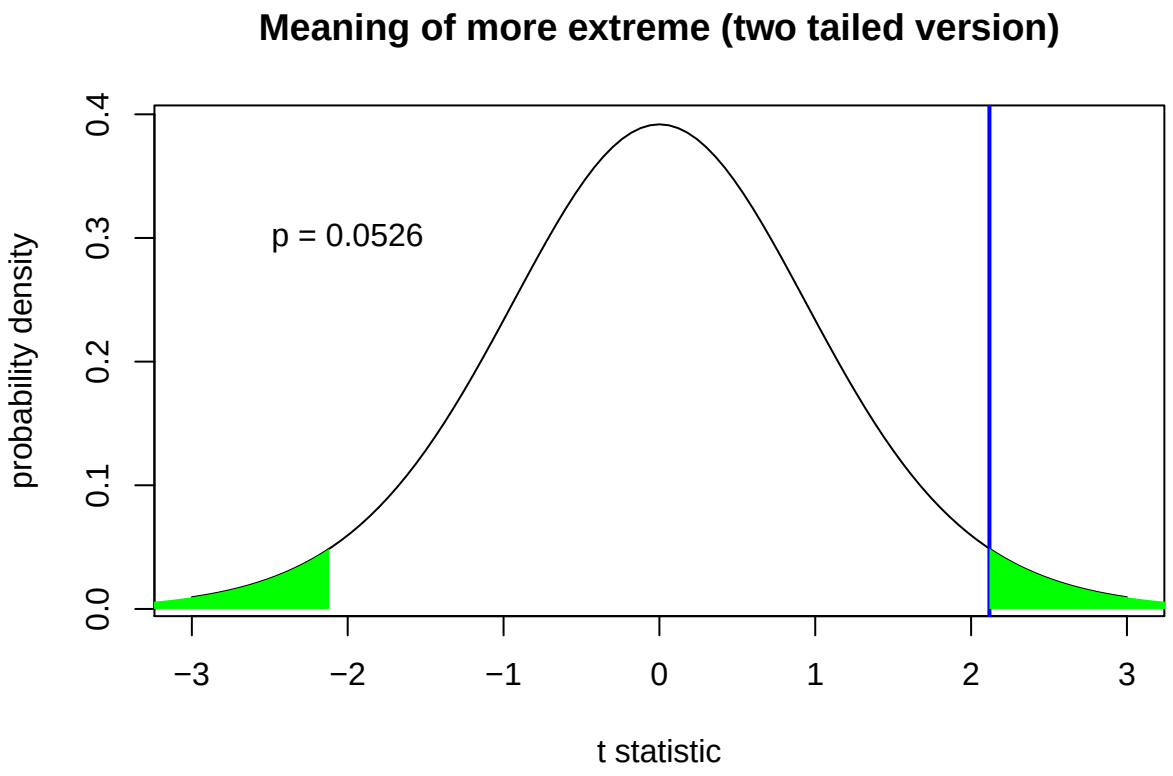
In a one-tailed test in which our (alternative) hypothesis is that our sample mean is *less than* than the null hypothesis, we mean 'more negative than the observed test statistic'.

In a two-tailed test, we mean 'more positive or more negative than the absolute value of our observed test statistic'

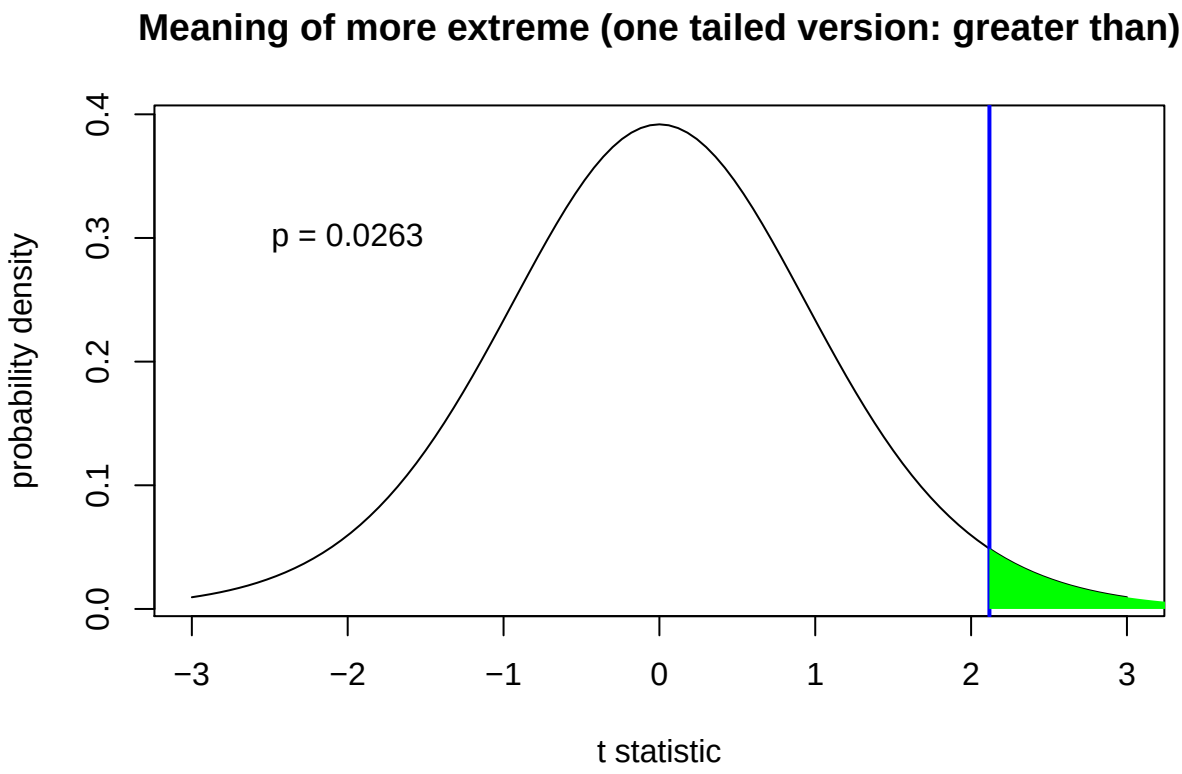
Here is an R demo to illustrate the concept:

We have the following data:

```
# one vs two tailed demo

#my.data <- rnorm(15, 0.5, 1)   # generate sample data
my.data <- c(0.20119786,1.41700898,-0.72426698,0.44006284,0.01487128,-0.19031680,1.75470699,-0.81992816
samp.mean <- mean(my.data)
samp.sd <- sd(my.data)
samp.n <- length(my.data)
std.err <- samp.sd/sqrt(samp.n)

null.mean <- 0

t.statistic <- (samp.mean-null.mean)/std.err

### Two-tailed
curve(dt(x,samp.n-1),-3,3, main="Meaning of more extreme (two tailed version)",
      ylab="probability density",xlab="t statistic")    # visualize the sampling distribution of the t-
abline(v=t.statistic,lwd=2,col="blue")

xs <- seq(abs(t.statistic),10,0.05)
ys <- dt(xs,samp.n-1)
polygon(x=c(xs,rev(xs)),y=c(ys,rep(0,times=length(ys))),col="green",border=NA)
polygon(x=c(-xs,rev(-xs)),y=c(ys,rep(0,times=length(ys))),col="green",border=NA)

p.twosided <- pt(-abs(t.statistic),samp.n-1)*2     # two-tailed p-value

text(-2,0.3,paste("p =",round(p.twosided,4)))
```

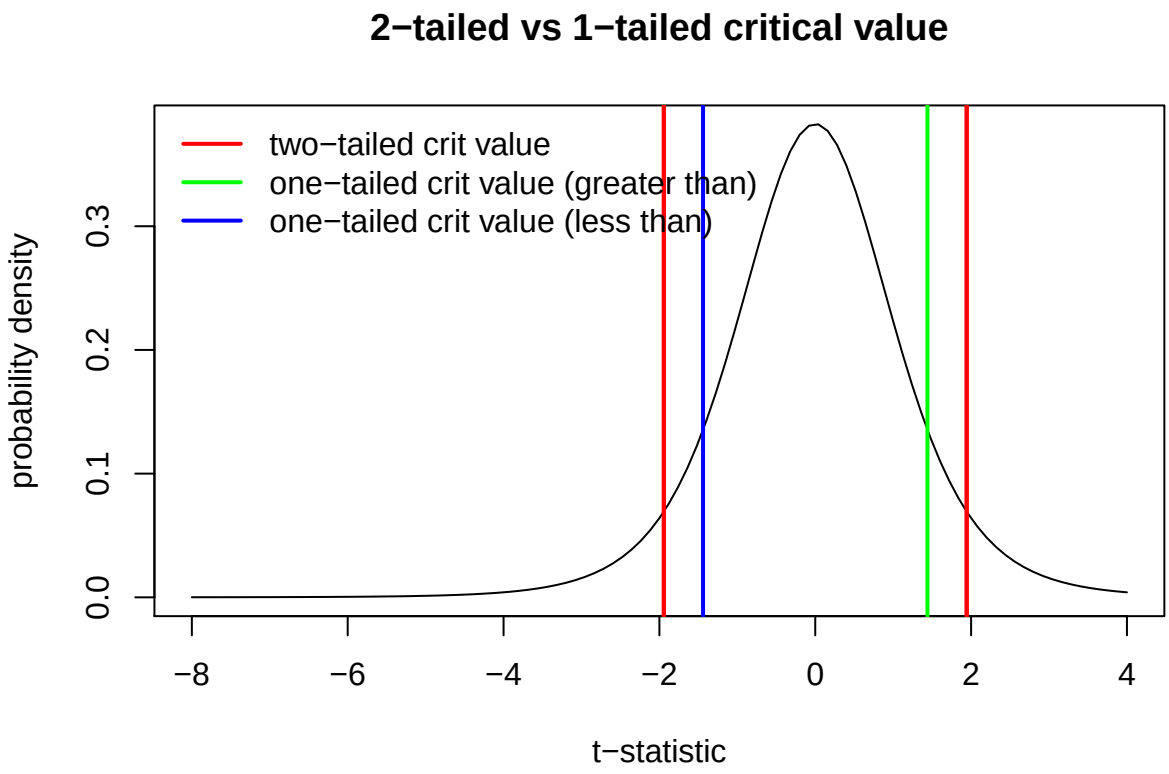



```r
### One-sided (alternative = 'greater')
curve(dt(x,samp.n-1),-3,3, main="Meaning of more extreme (one tailed version: greater than)",
      ylab="probability density",xlab="t statistic")    # visualize the sampling distribution of the t-
abline(v=t.statistic,lwd=2,col="blue")

xs <- seq(t.statistic,10,0.05)
ys <- dt(xs,samp.n-1)
polygon(x=c(xs,rev(xs)),y=c(ys,rep(0,times=length(ys))),col="green",border=NA)

p.onesided <- pt(-abs(t.statistic),samp.n-1)     # one-tailed p-value

text(-2,0.3,paste("p =",round(p.onesided,4)))
```



In the same way, the critical value for a t-statistic is different for a two-tailed vs a one-tailed test:

```
### t-crit in one tailed vs two tailed test

sample.size=7

curve(dt(x,sample.size-1),-8,4, main="2-tailed vs 1-tailed critical value",
      xlab="t-statistic",ylab="probability density")

alpha <- 0.1
t.crit.twosided <- qt(alpha/2,sample.size-1)

abline(v=c(t.crit.twosided,abs(t.crit.twosided)),col="red",lwd=2)


t.crit.twosided <- qt(alpha/2,sample.size-1)

abline(v=c(t.crit.twosided,abs(t.crit.twosided)),col="red",lwd=2)

t.crit.onesided <- qt(alpha,sample.size-1)

abline(v=abs(t.crit.onesided),col="green",lwd=2)
abline(v=t.crit.onesided,col="blue",lwd=2)

legend("topleft",lwd=c(2,2,2),col=c("red","green","blue"),bty="n",legend=c("two-tailed crit value","one
```



All of these critical values have the same percentage of the sampling distribution that are more extreme than them (the percentage is alpha, which is set at 0.1 in this example)– but the definition of 'extremeness' differs for the one-tailed and two-tailed cases!

Welch's t-test Welch's t-test allows for unequal sample sizes AND unequal variance in the two populations. The Welch's t-test is like the two-sample t-test we already ran, but it is more flexible. Like other t-tests, this test is robust against violations of normality due to the CLT. This is the DEFAULT version of a t-test in R. If you want a classical Student's t-test, you need to tell it to do so (otherwise it will do the Welch's test; use the 'var.equal = TRUE' argument to ru the classical test)!

I won't go through the math for the Welch's t-test (you can look it up if you'd like) because the basic concepts and interpretation are the same. The differences lie in the computation of the pooled variance and the degrees of freedom.

t-tests in R

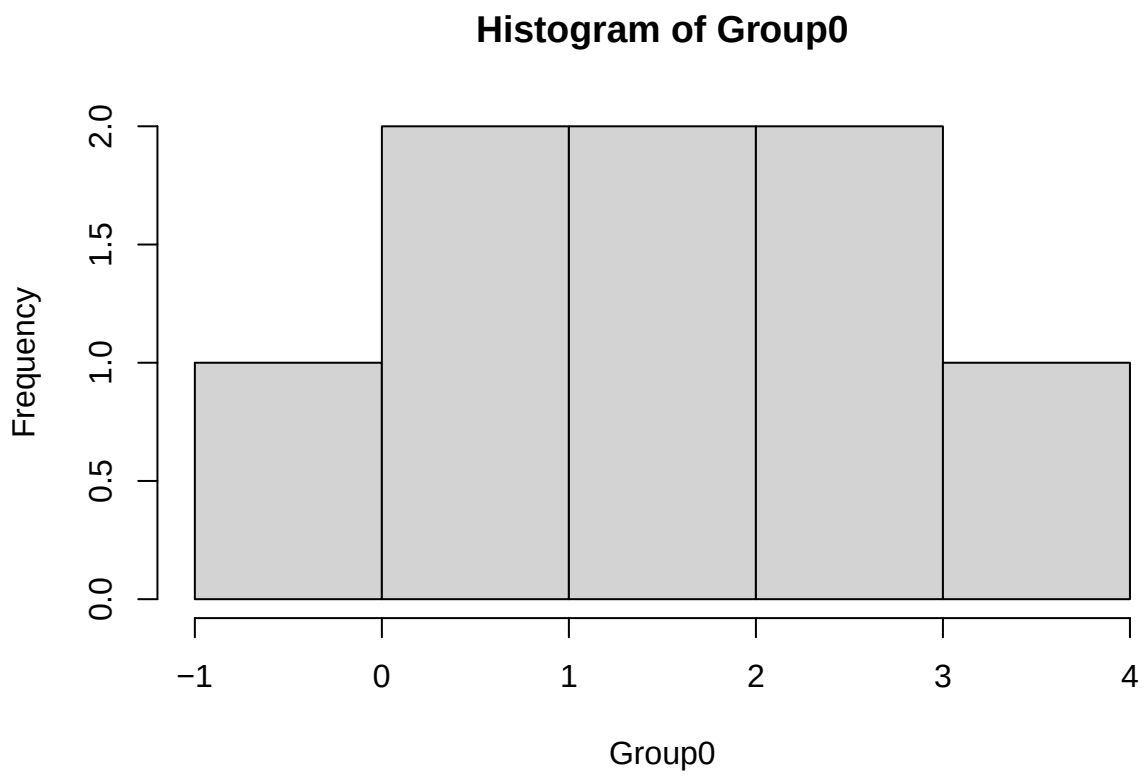
Previously, we have mostly avoided using R's built in 't-test' function. Let's use it now!

```
######
# More t-test examples


####
# T-tests
####
```

```r
#Ttest
# Are my data greater than zero?
Group0 <- c(0.5, -0.03, 4, 2.5, 0.89, 2.2, 1.7, 1.125)
hist(Group0)
t.test(Group0,alternative="greater") # This gets at directionality
```

```
##
##  One Sample t-test
##
## data:  Group0
## t = 3.5487, df = 7, p-value = 0.00468
## alternative hypothesis: true mean is greater than 0
## 95 percent confidence interval:
##  0.7507388       Inf
## sample estimates:
## mean of x
##  1.610625
```

```r
#Are my data different than zero?
Group0 <- c(0.5, -0.03, 4, 2.5, 0.89, 2.2, 1.7, 1.125)
hist(Group0)
```



```r
t.test(Group0) # Okay, that's to zero. What about if it's different than 1?
```

```
##
##  One Sample t-test
##
## data:  Group0
## t = 3.5487, df = 7, p-value = 0.00936
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.5374007 2.6838493
## sample estimates:
## mean of x
##  1.610625
```

```r
# are my data different than 1?
t.test(Group0, mu=1)
```

```
##
##  One Sample t-test
##
## data:  Group0
## t = 1.3454, df = 7, p-value = 0.2205
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  0.5374007 2.6838493
## sample estimates:
## mean of x
##  1.610625
```

```r
# Now let's test two groups.
# are the means equal?
group1 <- c(7,9,6,6,6,11,6,3,8,7)
group2 <- c(11,13,8,6,14,11,13,13,10,11)
t.test(group1, group2, var.equal=T) # Notice how we set equal variance? Look at the output - "Two Sampl
```

```
##
##  Two Sample t-test
##
## data:  group1 and group2
## t = -3.9513, df = 18, p-value = 0.000936
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -6.27997 -1.92003
## sample estimates:
## mean of x mean of y
##       6.9      11.0
```

```r
# is this one-tail or two?

group1 <- c(7,9,6,6,6,11,6,3,8,7)
group2 <- c(11,13,8,6,14,11,13,13,10,11)
t.test(group1, group2) #  "Welch's Two Sample t-test"
```

```
##
##  Welch Two Sample t-test
##
## data:  group1 and group2
## t = -3.9513, df = 17.573, p-value = 0.0009743
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -6.283771 -1.916229
## sample estimates:
## mean of x mean of y
##       6.9      11.0
```

```
# WELCH'S TEST IS THE DEFAULT IN R
```

t-test assumptions

Normality of the data

Like we have mentioned before, the t-test is quite robust to non-normality of the raw data – this is due to the Central Limit Theorem!

However, if the data are strongly skewed or otherwise clearly non-normal you may want to go with a non-parametric alternative (see below)!

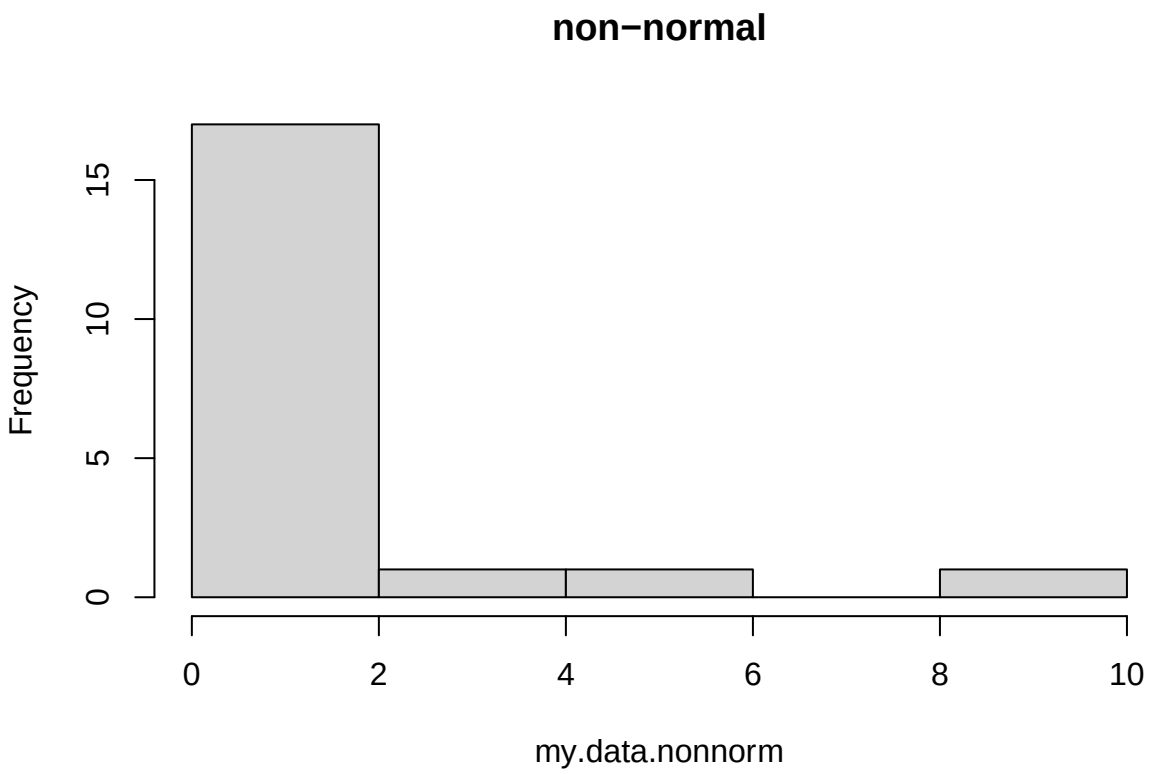
Testing for normality! Before 'accepting' the results of a statistical test, we often want to run 'goodness-of-fit' tests to see if there is any reason to think we may be violating key assumptions of our statistical analyses. For the t-test and many other parametric tests, one 'goodness-of-fit' test we might want to run is a test for normality. For linear regression analyses, we test for normality of the residuals. For the t-test, we test for normality of the raw data.

Here we will discuss two tests- one visual and one quantitative. The visual test is called a 'quantile-quantile (Q-Q) plot' and the quantitative test is called a Shapiro-Wilk test.

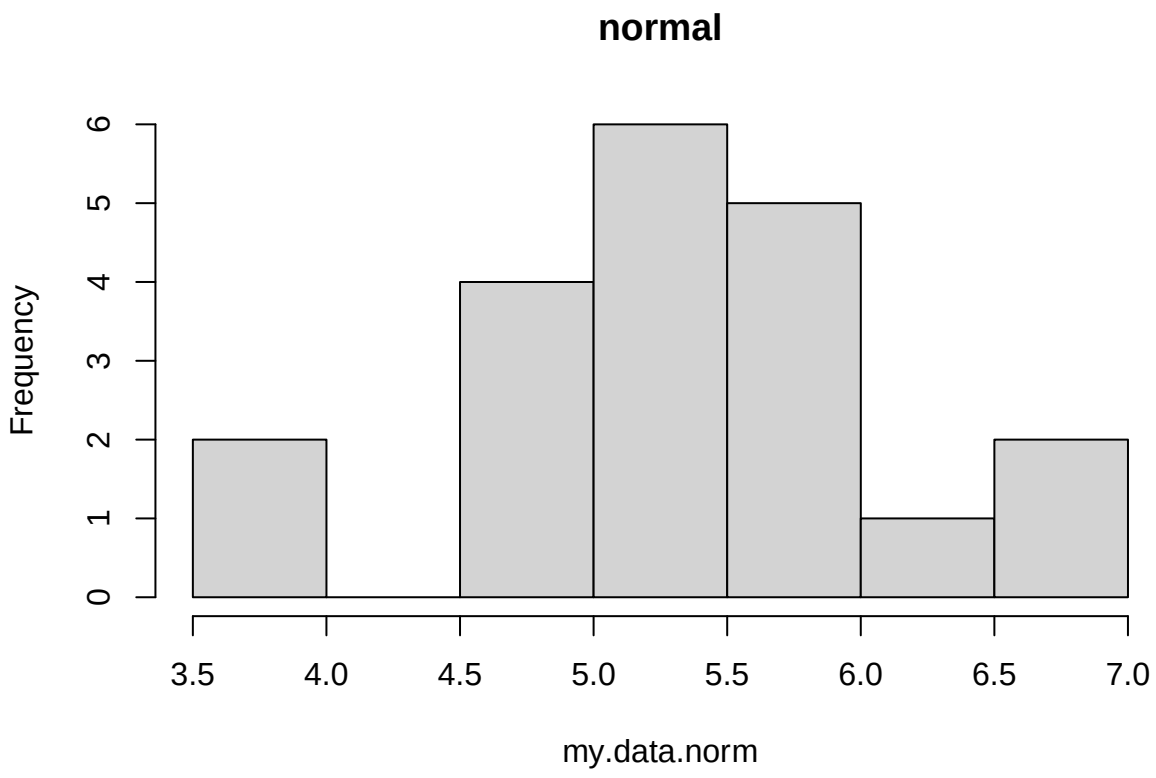
The quantile-quantile plot compares the quantiles of the observed data (y axis) with the theoretical quantiles from a normal distribution (x axis). If the q-q plot is highly non-linear, your data are probably not normally distributed and you might want to run a non-parametric alternative.

The Shapiro-Wilk test is a way of quantitatively assessing the linearity of the q-q plot such that you can get a p-value. The p-value here is interpreted the same as any other p-value. The catch is that the null hypothesis is that the data are normally distributed. If p>alpha you fail to reject the null - that is, you can move forward assuming that the data are normally distributed!

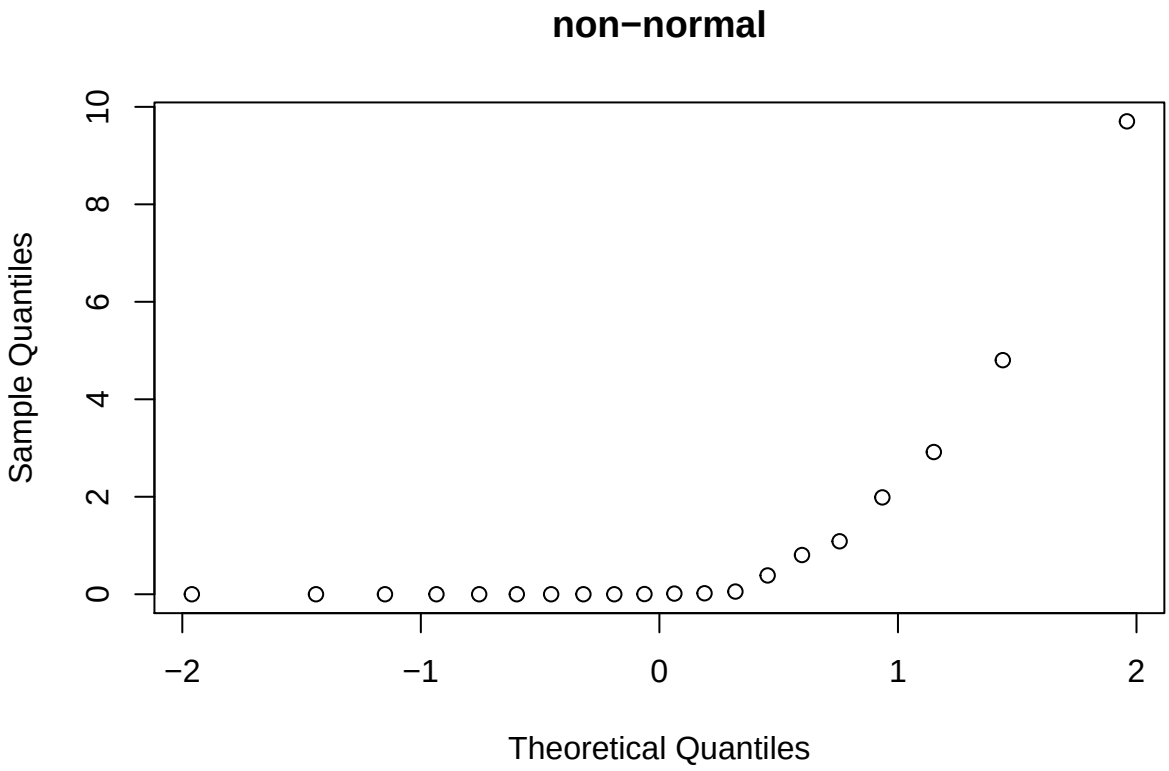
```
## testing for normality

my.data.nonnorm <- rgamma(20,0.1,0.1)     # simulate some non-normal data
hist(my.data.nonnorm,main="non-normal")      # visualize the data distribution
```



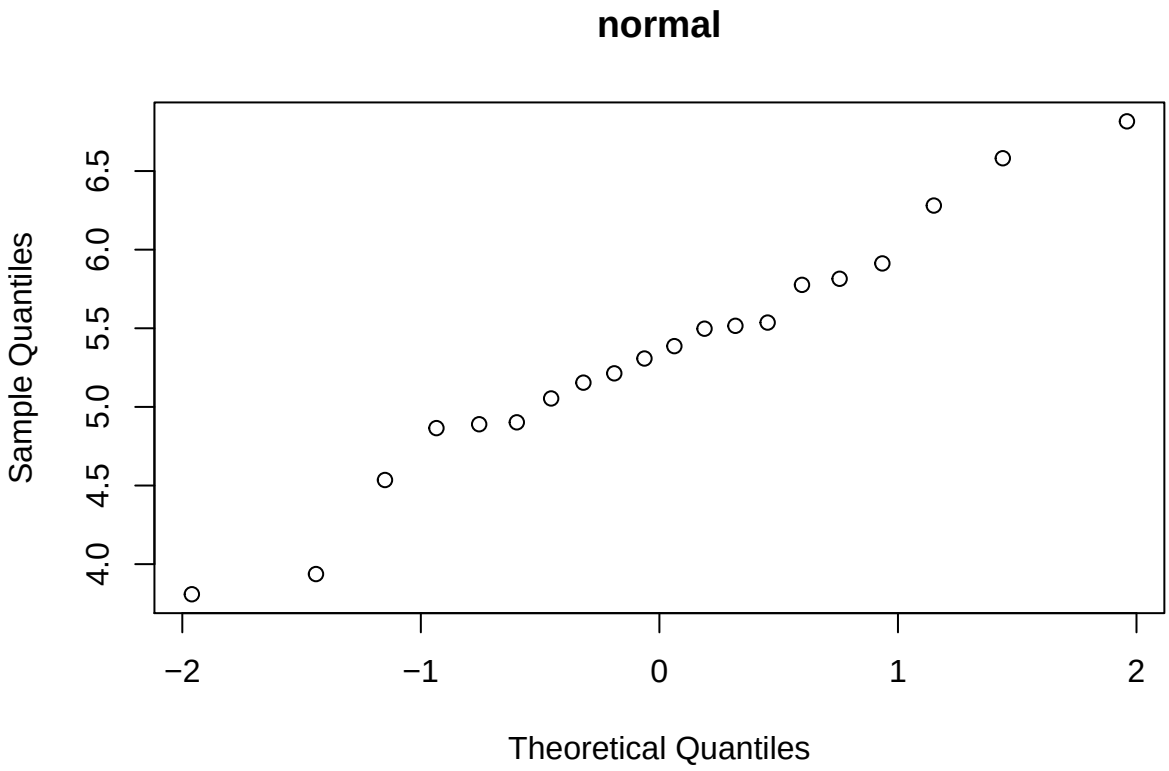

```
my.data.norm <- rnorm(20,6,0.9)     # simulate some normal data
hist(my.data.norm,main="normal")      # visualize the data distribution
```




```
# visualize q-q plot

qqnorm(my.data.nonnorm,main="non-normal")    # visual test for normality
```




```
qqnorm(my.data.norm,main="normal")
```




```
# run shapiro-wilk test

shapiro.test(my.data.nonnorm)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  my.data.nonnorm
## W = 0.53978, p-value = 7.435e-07
```

```
shapiro.test(my.data.norm)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  my.data.norm
## W = 0.97558, p-value = 0.8654
```

Outliers

The t-test can be highly affected by outliers. If you suspect your data contain highly influential outliers, you might want to run a non-parametric alternative, as non-parametric tests are much less influenced by outliers.

To look for outliers, the first step is to simply visualize a histogram of the raw data and look for any observations that fall far from the others.

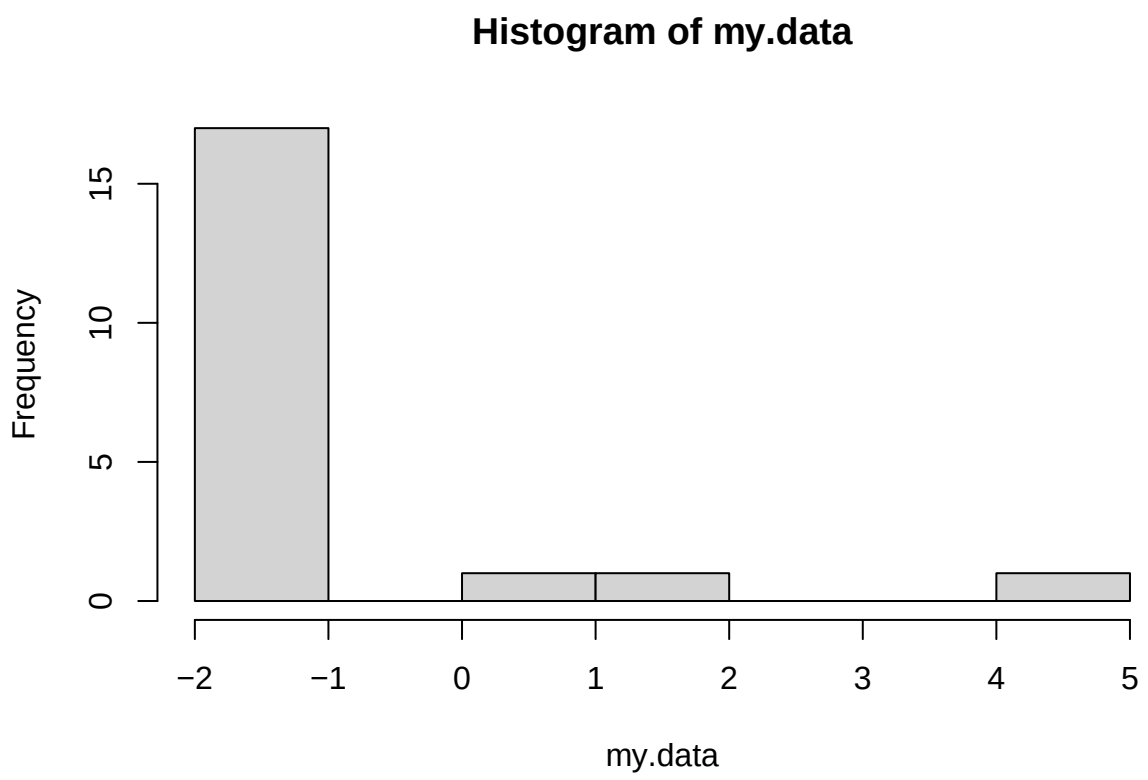
Non-parametric alternatives

If your data are highly non-normal or contain obvious outliers, you probably want to run one of the non-parametric alternatives to the classical t-test. These tests do not assume normality of the data nor are they highly affected by outliers!

One sample t-test

The non-parametric alternative to the one-sample t-test is called the **one sample Wilcoxon signed rank test**. This analysis first reduces the data to signs- negative 1 if the observation is greater than the null and -1 if the data are less than the null. Then, like many non-parametric tests, you sort the data from smallest to largest absolute value. You then multiply the ranks times the signs and sum across all observations. This test statistic (W) has a known sampling distribution (approximately anyway!) under the null hypothesis and therefore you can compute a p-value. The null hypothesis for the Wilcoxon test refers to the median of the distribution and not the mean (the null hypothesis for the t-test refers to the mean). Therefore the two tests are not strictly comparable!

```
# Wilcoxon signed rank test

my.data <- rgamma(20,0.1,0.1)-2

hist(my.data)
```



```r
wilcox.test(my.data)
```

```
## Warning in wilcox.test.default(my.data): cannot compute exact p-value with ties
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  my.data
## V = 26, p-value = 0.00338
## alternative hypothesis: true location is not equal to 0
```

```r
t.test(my.data)   # t-test for comparison
```

```
##
##  One Sample t-test
##
## data:  my.data
## t = -3.3774, df = 19, p-value = 0.003162
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -2.137474 -0.501834
## sample estimates:
## mean of x
## -1.319654
```

Two sample t-test

The non-parametric alternative to the two-sample t-test is called the **Mann Whitney U test (or Wilcoxon rank sum test)**. This analysis first reduces the data to ranks- that is, you first sort the data from smallest to largest absolute value. You then essentially test to see if the ranks of one group are higher on average than the ranks of the other group. This test statistic (U) has a known sampling distribution (approximately anyway!) under the null hypothesis and therefore you can compute a p-value. The null hypothesis for the Mann-Whitney test is that there is a 50% probability that a single sample from one distribution is larger than a single sample drawn from the other distribution. The null hypothesis for the t-test is that the means of two distributions are the same. Therefore the two tests are not strictly comparable!

```r
# Wilcoxon signed rank test

my.data1 <- rgamma(20,0.1,0.1)-2
my.data2 <- rgamma(20,0.2,0.1)-2

median(my.data1)
```

```
## [1] -1.99929
```

```r
median(my.data2)
```

```
## [1] -1.889916
```

```r
wilcox.test(my.data1,my.data2)
```

```
##
##  Wilcoxon rank sum exact test
##
## data:  my.data1 and my.data2
## W = 132, p-value = 0.0675
## alternative hypothesis: true location shift is not equal to 0
```

```r
t.test(my.data1,my.data2)   # t-test for comparison
```

```
##
##  Welch Two Sample t-test
##
## data:  my.data1 and my.data2
## t = -1.5486, df = 19.514, p-value = 0.1375
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -5.2782584  0.7845053
## sample estimates:
## mean of x mean of y
## -1.656629  0.590248
```

```r
#### perform rank test from scratch!

allobs <- c(my.data1,my.data2)
inorder <- order(allobs)

rank1 <- inorder[1:20]
rank2 <- inorder[21:40]

t.test(rank1,rank2,var.equal = T)    # perform t-test on the ranks (usually similar to Mann-Whitney tes
```

```
##
##  Two Sample t-test
##
## data:  rank1 and rank2
## t = -0.53604, df = 38, p-value = 0.5951
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -9.553202  5.553202
## sample estimates:
## mean of x mean of y
##      19.5      21.5
```

Power analysis for t-test

Okay, before we leave the t-test, let's run a power analysis! Remember that a power analysis is a way of assessing whether you are likely to detect a signal given your null hypothesis is false (i.e., there is a signal there to detect!). Power analyses can be a great way to assess if your sampling design is adequate to detect a signal. If not, you may want to alter your sampling design by adding more replicates!

```r
###### First we will set the population parameters:

true.mean.A <- 13.5
true.mean.B <- 13.9

true.sd <- 3.4

# Assume a two-tailed test- alternative hypothesis is that the mean of A is different from the mean of

###### Now let's set the sampling scenario

sampsize.A <- 10
sampsize.B <- 12

###### now we will simulate a single 'experiment'

samp.A <- rnorm(sampsize.A,true.mean.A,true.sd)
samp.B <- rnorm(sampsize.B,true.mean.B,true.sd)

###### and now we can run a test!

this.test <- t.test(samp.A,samp.B,var.equal = T)

###### and determine if we rejected our null hypothesis (which we know is not true!)

this.test$p.value < 0.05
```

```
## [1] FALSE
```

Okay, but if we really want to compute power we need to do this many times and see what percent of samples allow us to reject the null!

```r
### full power analysis:

###### First we will set the population parameters:

true.mean.A <- 13.5
true.mean.B <- 13.9

true.sd <- 3.4

# Assume a two-tailed test- alternative hypothesis is that the mean of A is different from the mean of

###### Now let's set the sampling scenario

sampsize.A <- 10
sampsize.B <- 12

###### now we will simulate LOTS of  'experiments'

pvals <- numeric(1000)

for(i in 1:1000){
```

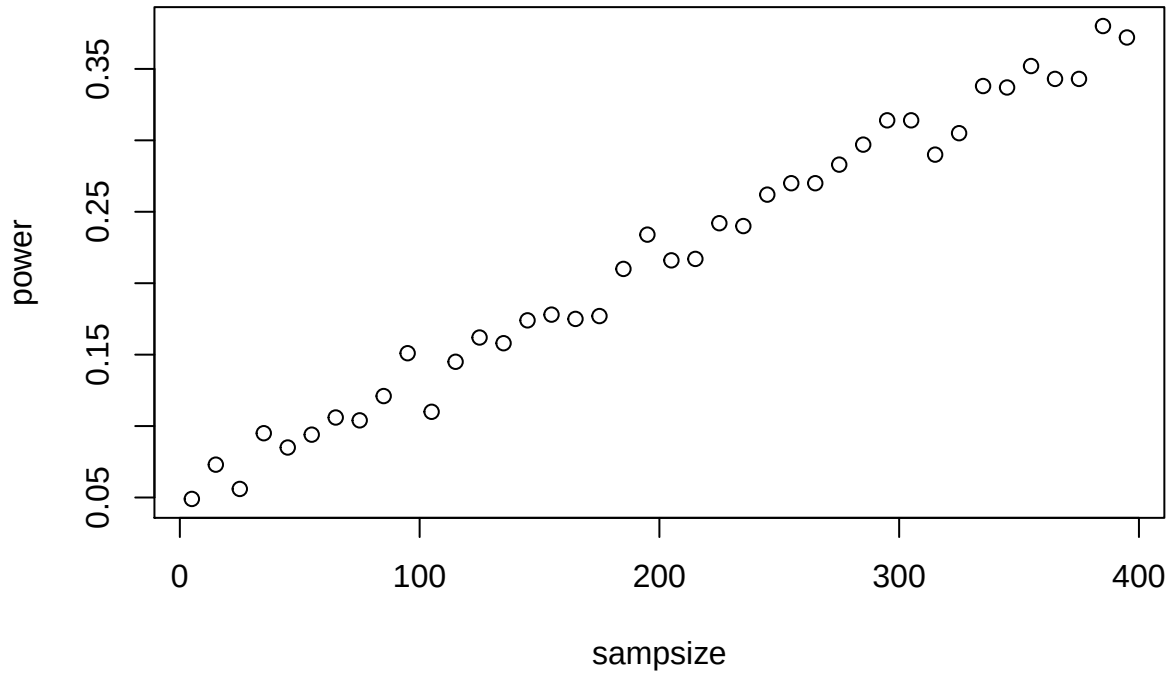
```
  samp.A <- rnorm(sampsize.A,true.mean.A,true.sd)
  samp.B <- rnorm(sampsize.B,true.mean.B,true.sd)

  ###### and now we can run a test!

  this.test <- t.test(samp.A,samp.B,var.equal = T)

  ###### and determine if we rejected our null hypothesis (which we know is not true!)

  pvals[i] <- this.test$p.value
}

 # hist(pvals)

length(which(pvals<0.05))/1000
```

```
## [1] 0.052
```

Of course, if your power is not adequate, you may want to increase sample size. But what is the minimum sample size that gives you sufficient power? We can use simulation to answer this question!

```
### full power analysis WITH SAMPLE SIZE DETERMINATION:

###### First we will set the population parameters:

true.mean.A <- 13.5
true.mean.B <- 13.9

true.sd <- 3.4

# Assume a two-tailed test- alternative hypothesis is that the mean of A is different from the mean of

###### now we will simulate LOTS of  'experiments' under different sample sizes

sampsize <- seq(5,400,10)

power <- numeric(length(sampsize))

for(j in 1:length(sampsize)){

  pvals <- numeric(1000)
  for(i in 1:1000){
    samp.A <- rnorm(sampsize[j],true.mean.A,true.sd)
    samp.B <- rnorm(sampsize[j],true.mean.B,true.sd)

    ###### and now we can run a test!

    this.test <- t.test(samp.A,samp.B,var.equal = T)

    ###### and determine if we rejected our null hypothesis (which we know is not true!)

    pvals[i] <- this.test$p.value
```

```
  }

 # hist(pvals)

  power[j] <- length(which(pvals<0.05))/1000
}

names(power) <- sampsize

plot(power~sampsize)
```



So what sample size would you need (in each of the two groups) to detect a signal in this case?

–go to next lecture–