# Practical: Investigating the t-test with simulated data

Polina Soloveva

2023-11-22

## Task 1

Establish the null hypothesis that the means of the two datasets are equal. The alternative hypothesis says that the means are not equal. If p-value is greater than 0.05, then there is no significant difference between the means of the two groups.

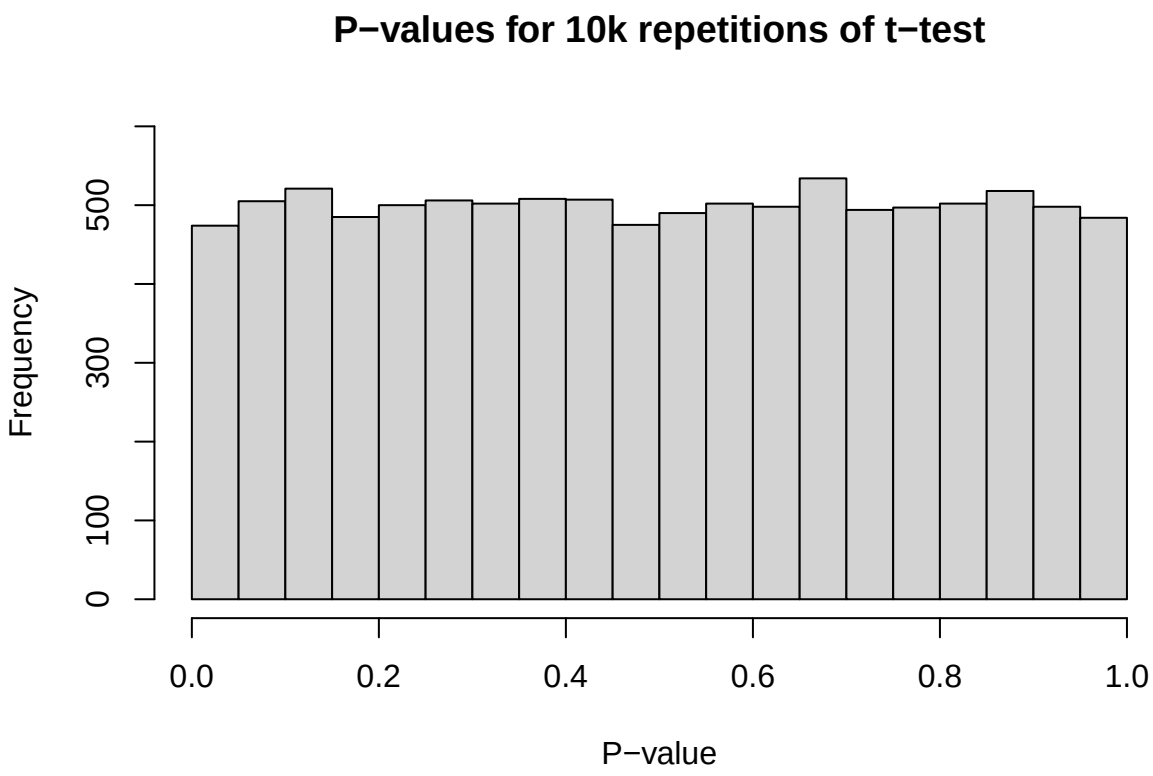
```
ds_1 <- rnorm(10)
ds_2 <- rnorm(10)
t.test(ds_1, ds_2)
```

```
##
##  Welch Two Sample t-test
##
## data:  ds_1 and ds_2
## t = 0.36445, df = 16.743, p-value = 0.7201
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.032390  1.462931
## sample estimates:
##  mean of x  mean of y
##  0.1453836 -0.0698873
```

## Task 2

Now repeat Task 1 for r = 10000 times

```
r <- 10000
pvalue_sum <- c()

for (i in 1:r) {
  ds_1 <- rnorm(10)
  ds_2 <- rnorm(10)
  pvalue <- t.test(ds_1, ds_2)$p.value
  pvalue_sum <- c(pvalue_sum, pvalue)
}

hist(pvalue_sum, xlab = "P-value", ylim = c(0,600),
     main = "P-values for 10k repetitions of t-test")
```




The resulting distribution of values is uniform without peaks

### Task 3

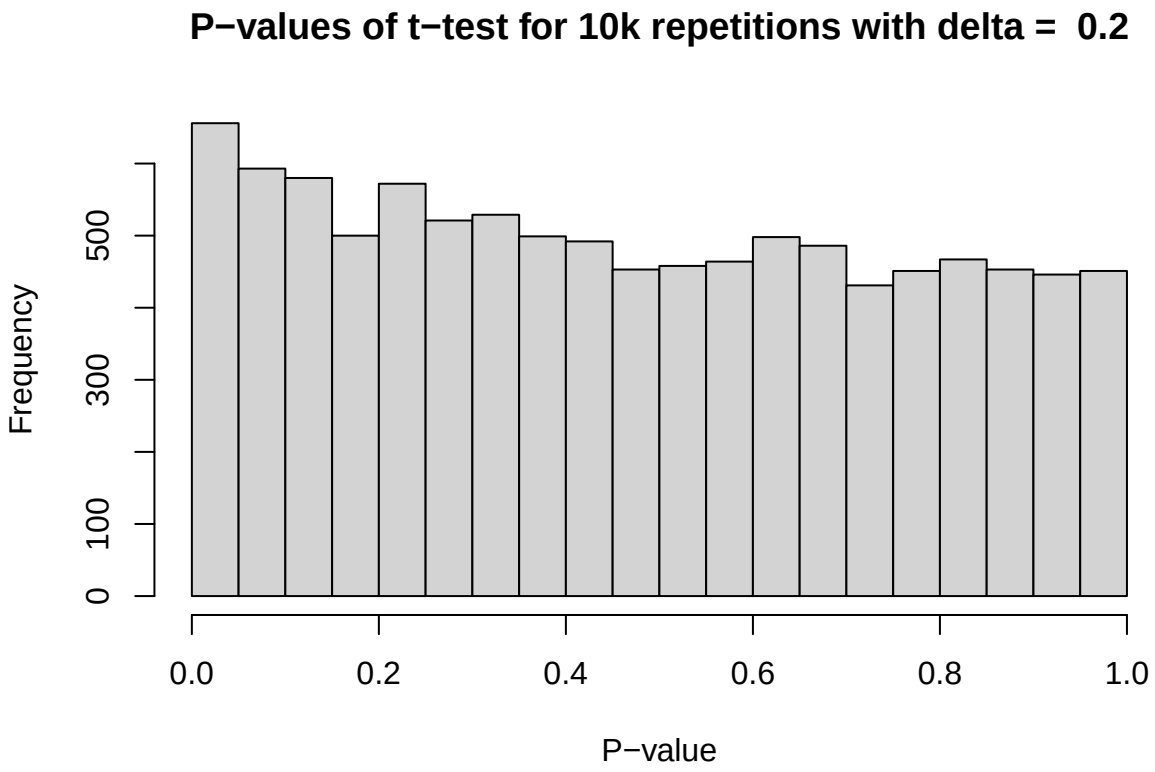
Create a function that takes the mean and variance parameters to construct a normal distribution.

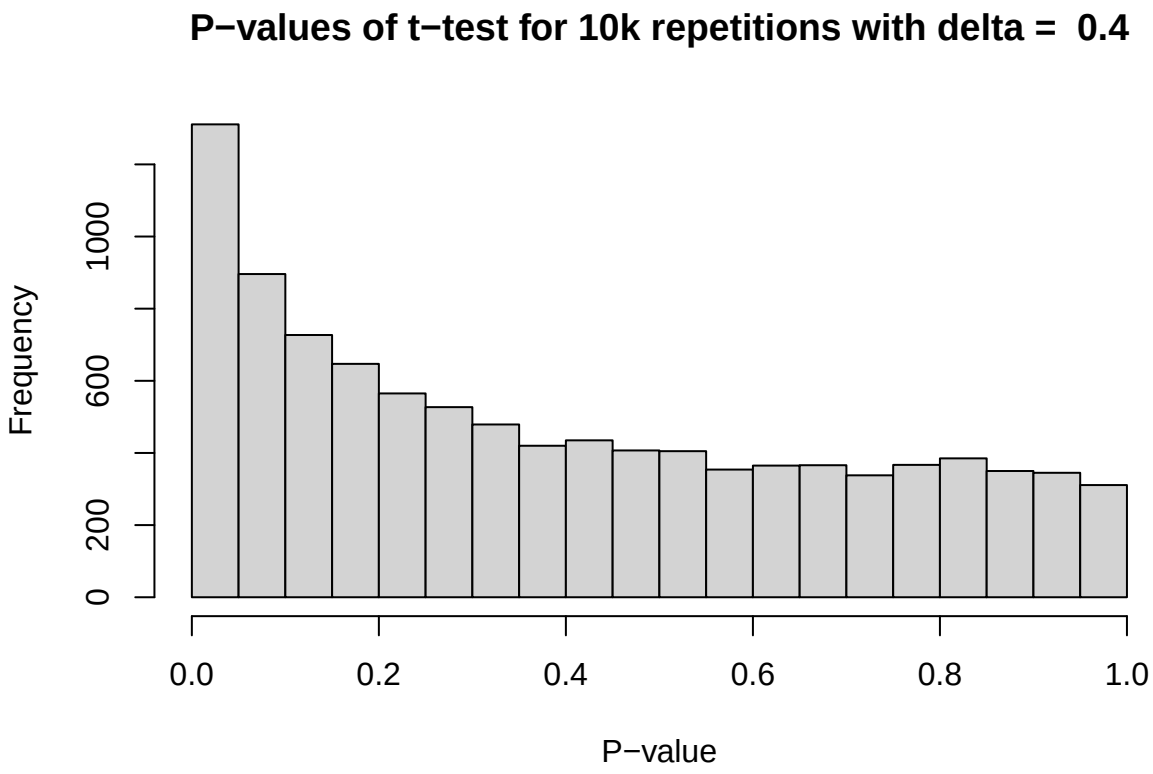
```
pvalue_function <- function(n, d, v1, v2){
  r <- 10000
  pvalue_sum <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, -d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- t.test(ds_1, ds_2)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }

  return(pvalue_sum)
  }
```

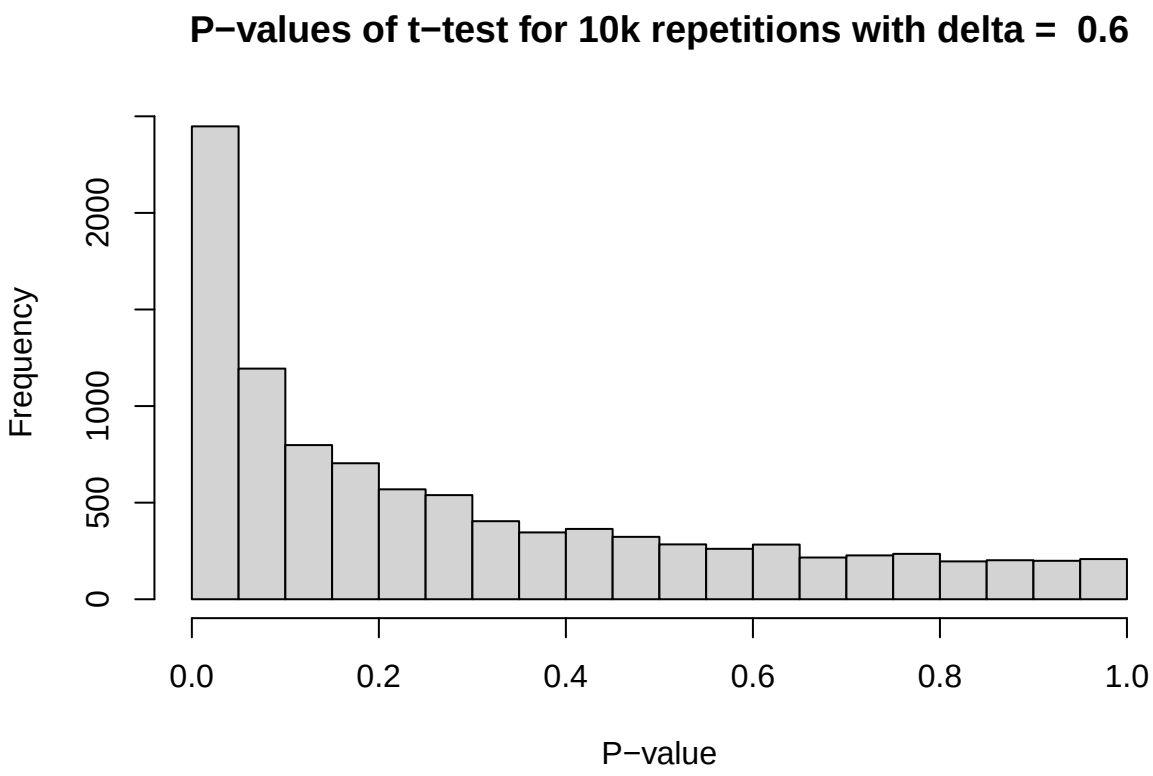
Build several histograms for different deltas and variations equal to 1

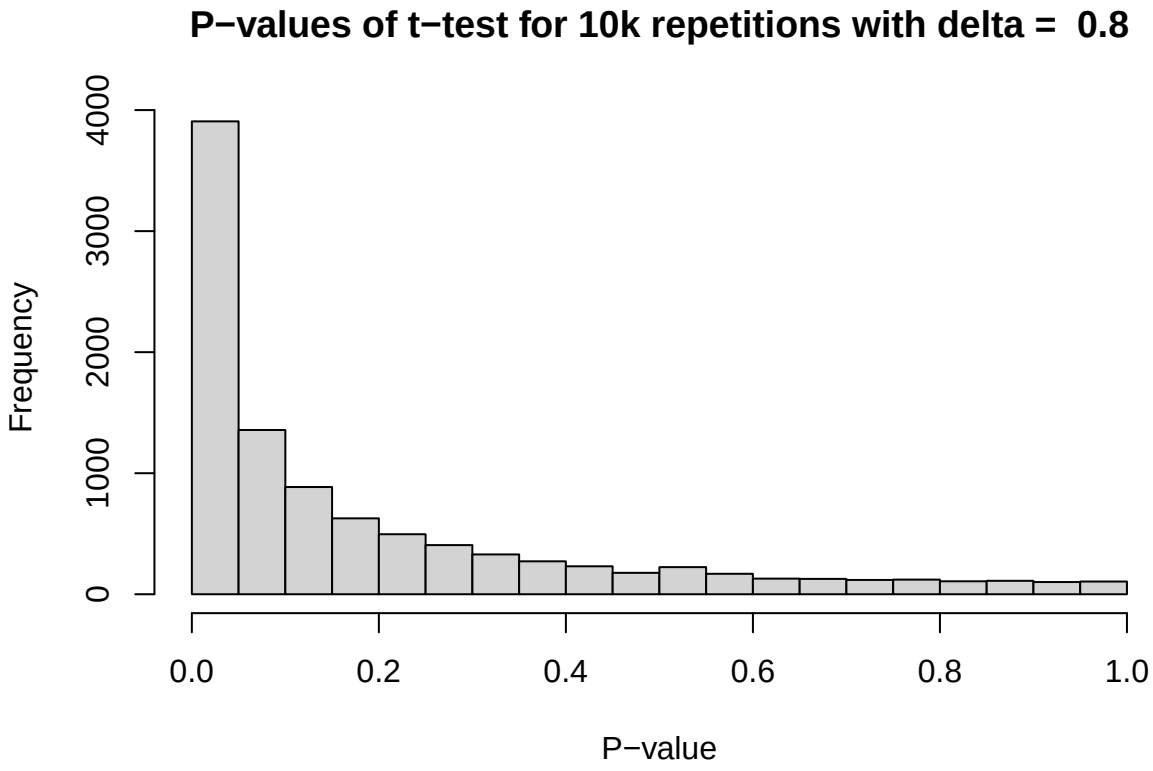
```
delta <- c(0.2, 0.4, 0.6, 0.8, 1)
n <- 10
v1 <- 1
```

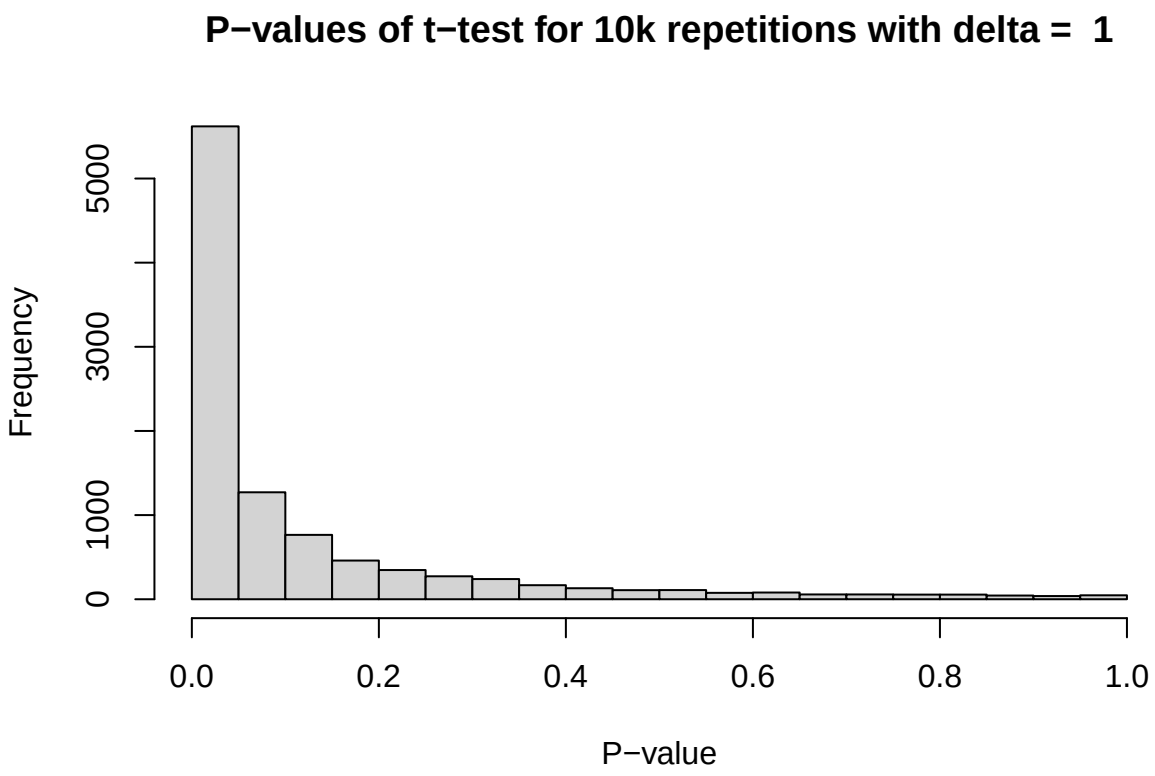
```r
v2 <- 1

for (d in delta){
  pvalue_sum <- pvalue_function(n, d, v1, v2)
  hist(pvalue_sum, xlab = "P-value",
       main = paste("P-values of t-test for 10k repetitions with delta = ", d))
}
```




P−values of t−test for 10k repetitions with delta = 0.4
Frequency
0
200
600
1000
0.0
0.2
0.4
0.6
0.8
1.0
P−value


P−values of t−test for 10k repetitions with delta = 0.6
Frequency
0
500
1000
2000
0.0
0.2
0.4
0.6
0.8
1.0
P−value


P−values of t−test for 10k repetitions with delta = 0.8
Frequency
0
1000
2000
3000
4000
0.0
0.2
0.4
0.6
0.8
1.0
P−value




The larger the delta, the farther the mean values of the datasets are from each other, and the statistical test is the more reliably.

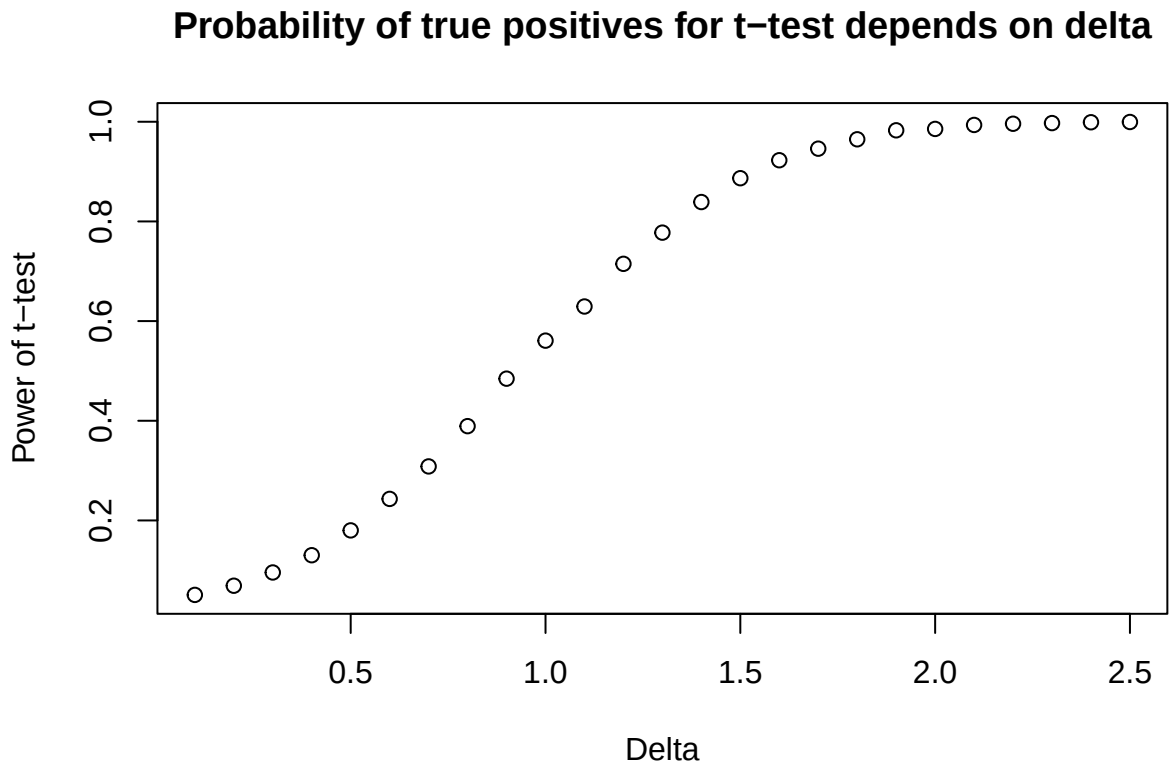
### Task 4

Probability of a "true positive", i.e., correctly rejecting the null hypothesis, is also known as the power of the test. Plot the power (probability of true positives) of the test as a function of deltas.

If p-value is greater than 0.05, then there is no significant difference between the means of the two groups. At the same time, we know that the sample means differ (since we initially set them to be different). Therefore, we will consider those events in which p-value is less than 0.05 to be true positive, that is, the hypothesis of equality of means is not accepted, that is, the test gives a true answer.

```
delta <- seq(0.1, 2.5, 0.1)
n <- 10
v1 <- 1
v2 <- 1
r <- 10000
power <- c()

for (d in delta){
  pvalue_sum <- pvalue_function(n, d, v1, v2)
  TP <- length(pvalue_sum[pvalue_sum < 0.05])
  power <- c(power, TP/r)
}
```

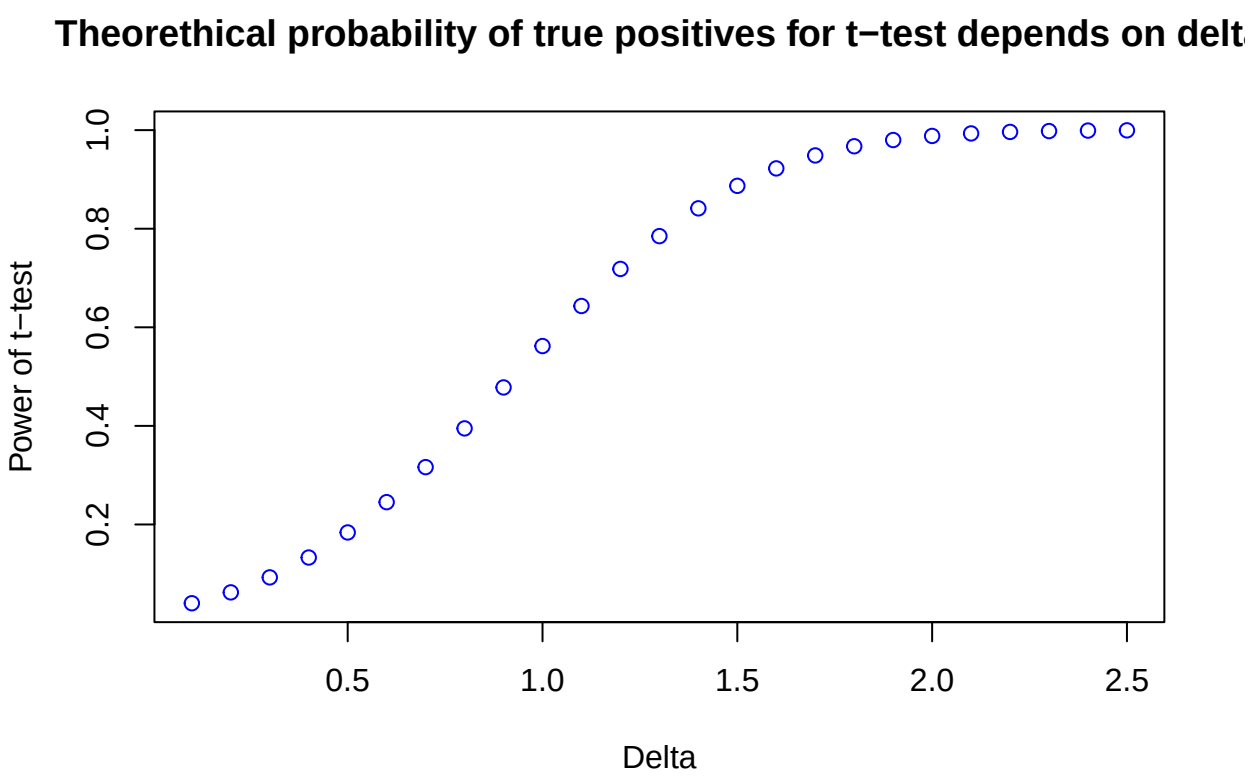
```
plot(delta, power, xlab = "Delta", ylab = "Power of t-test",
     main = "Probability of true positives for t-test depends on delta")
```




Thus, when the delta reaches a value of 2 (that is, the mean values of the datasets become -1 and 1, respectively), the power of the test reaches a plateau and becomes 100%.

Compare the results to the theoretical power curve derived from the t-distribution.

```
theor_power <- power.t.test(n = n, delta = delta, sd = v1)$power


plot(delta, theor_power, col = 'blue', xlab = "Delta", ylab = "Power of t-test",
     main = "Theorethical probability of true positives for t-test depends on delta")
```



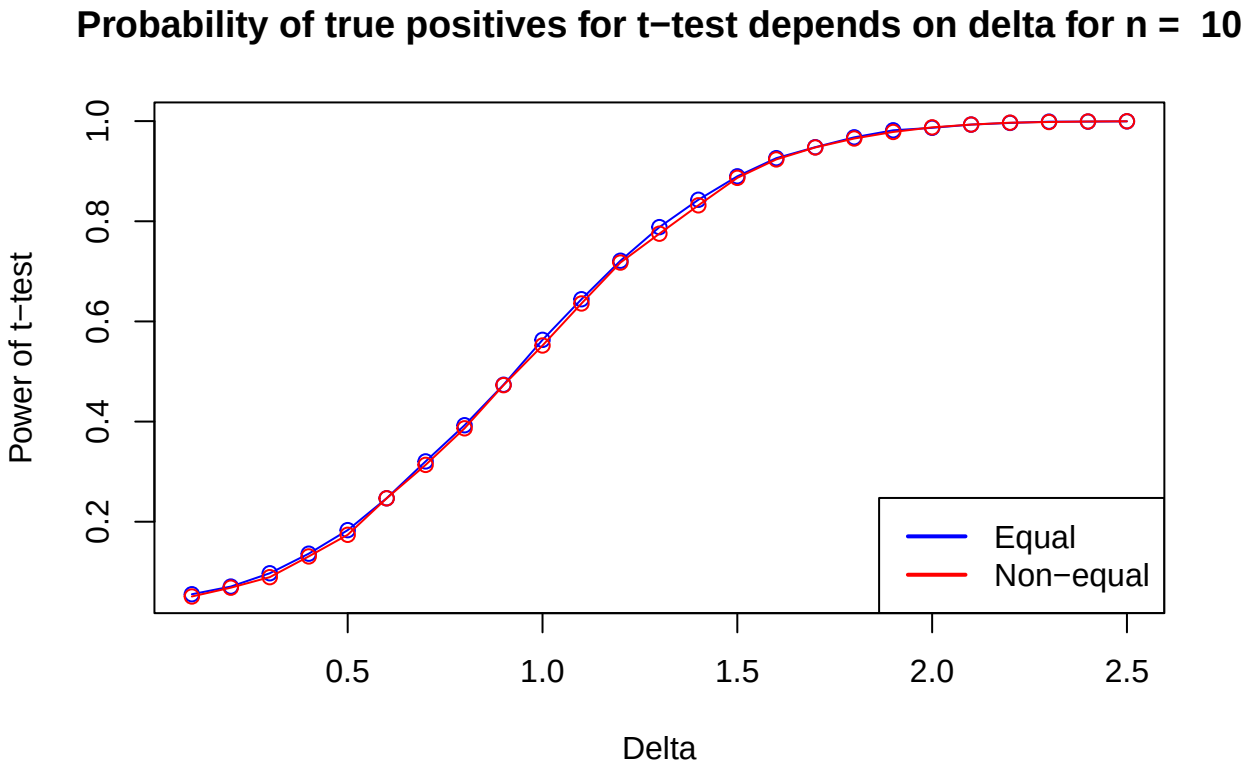

The obtained dependencies are very similar and we can say that the theoretical dependence is reproduced during observation.

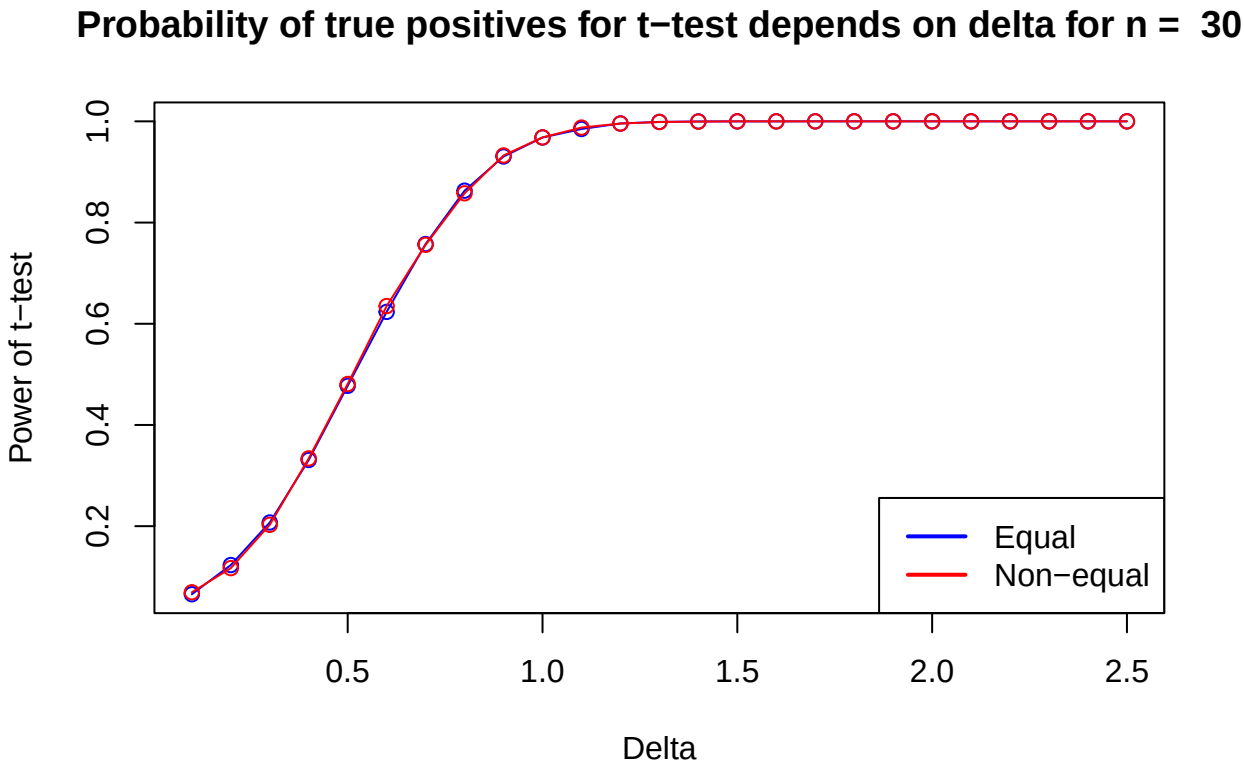
### Task 5

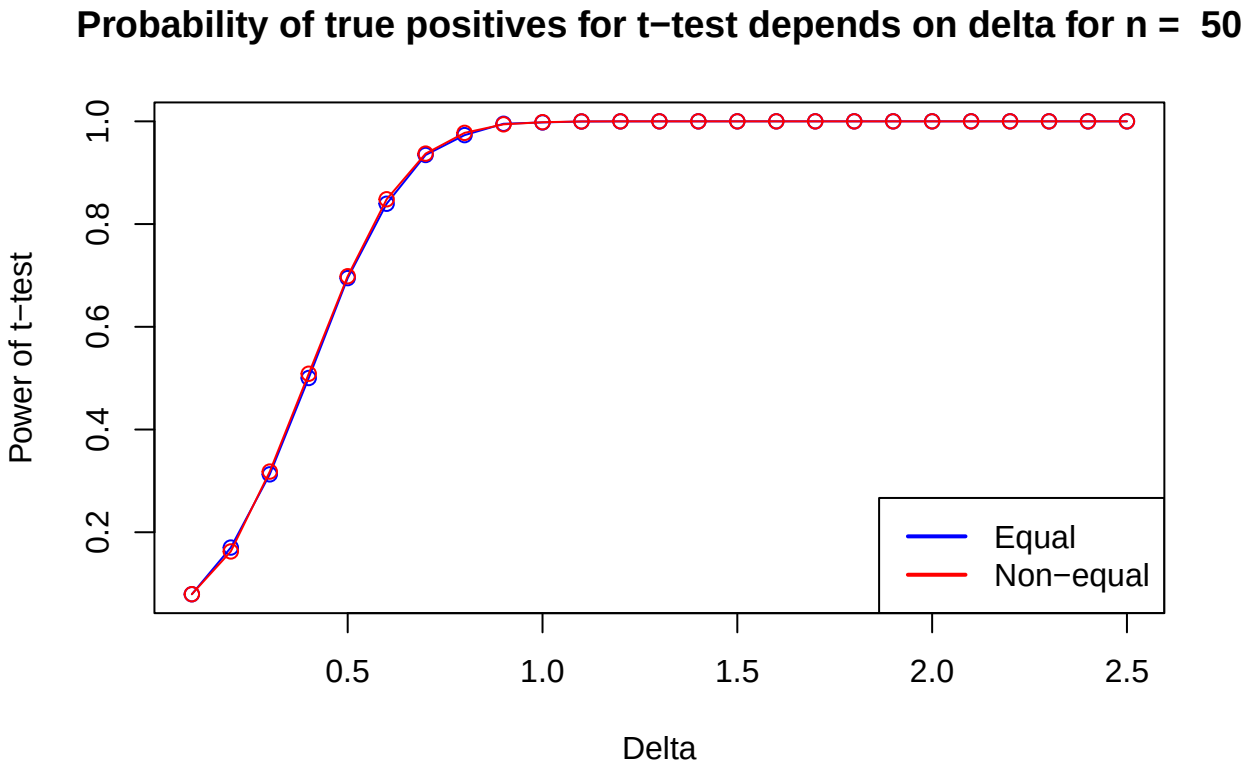
Assume that datasets distributions have equal variances and compare the power of the two versions of the t-test (for equal and unequal variances, respectively).

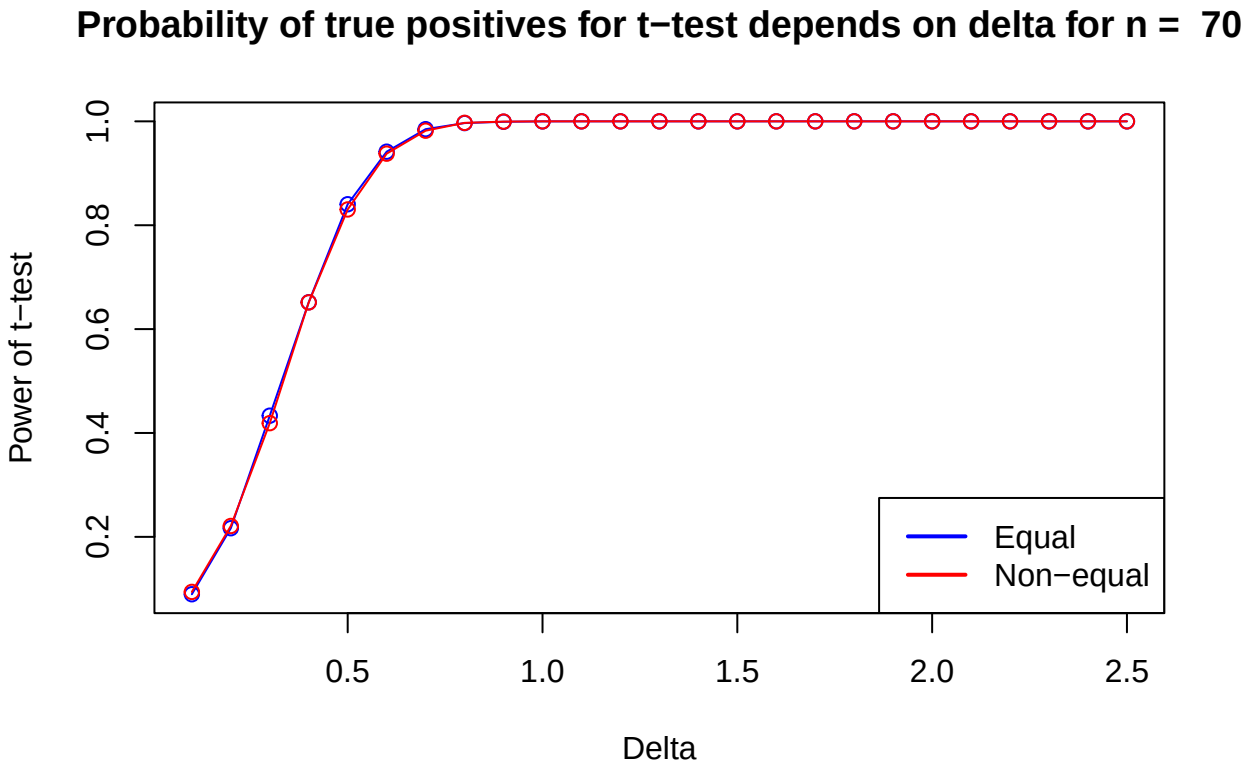
```
power_function_equal <- function(n, d, v1, v2){
  r <- 10000
  pvalue_sum <- c()
  power <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, -d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- t.test(ds_1, ds_2, var.equal = TRUE)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }

  TP <- length(pvalue_sum[pvalue_sum < 0.05])
  power <- c(power, TP/r)

  return(power)
  }
```

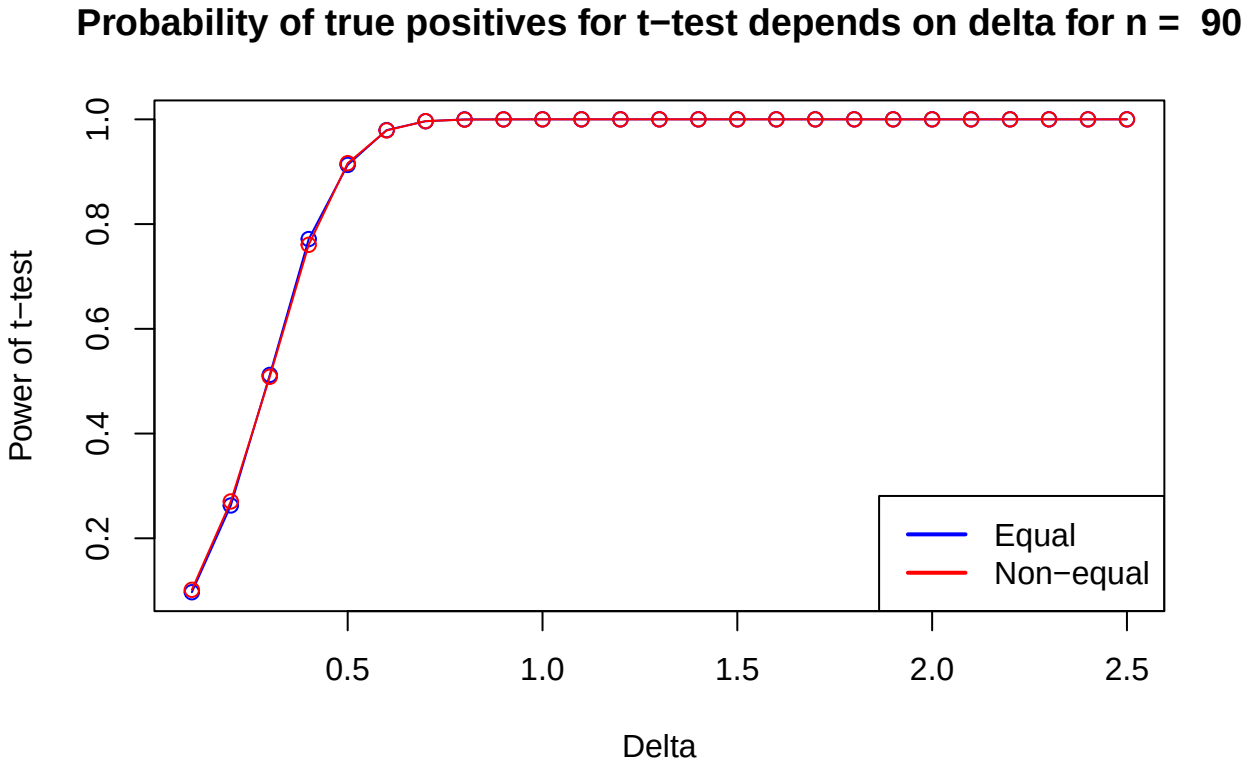
```r
power_function_nonequal <- function(n, d, v1, v2){
  r <- 10000
  pvalue_sum <- c()
  power <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, -d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- t.test(ds_1, ds_2, var.equal = FALSE)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }

  TP <- length(pvalue_sum[pvalue_sum < 0.05])
  power <- c(power, TP/r)

  return(power)
  }

delta <- seq(0.1, 2.5, 0.1)
N <- seq(10, 110, 20)
v1 <- 1
v2 <- 1

for (n in N){
  power_equal <- c()
  power_nonequal <- c()

  for (d in delta){
    pw_equal <- power_function_equal(n, d, v1, v2)
    power_equal <- c(power_equal, pw_equal)
    pw_nonequal <- power_function_nonequal(n, d, v1, v2)
    power_nonequal <- c(power_nonequal, pw_nonequal)
  }

  plot(delta, power_equal, col = 'blue', type = "o", xlab = "Delta",
       ylab = "Power of t-test",
       main = paste("Probability of true positives for t-test depends on delta for n = ", n))
  lines(delta, power_nonequal, col = 'red', type = "o")
  legend("bottomright", legend = c("Equal", "Non-equal"),
         col = c("blue", "red"), lwd = 2)
}
```

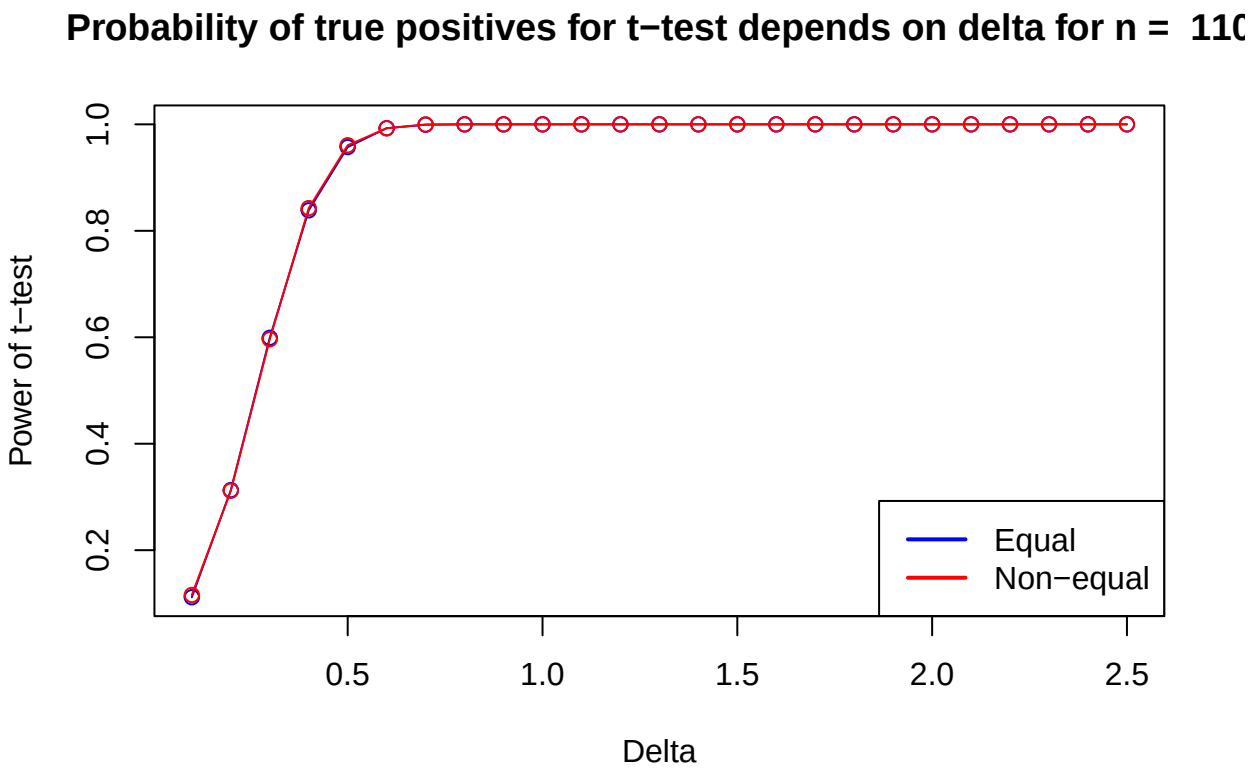

Probability of true positives for t−test depends on delta for n = 10
Power of t−test
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
Delta
Equal
Non−equal


Probability of true positives for t−test depends on delta for n = 30
Power of t−test
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
Delta
Equal
Non−equal


Probability of true positives for t−test depends on delta for n = 50
Power of t−test
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
Delta
Equal
Non−equal


Probability of true positives for t−test depends on delta for n = 70
Power of t−test
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
Delta
Equal
Non−equal


Probability of true positives for t−test depends on delta for n = 90
Power of t−test
0.2
0.4
0.6
0.8
1.0
0.5
1.0
1.5
2.0
2.5
Delta
Equal
Non−equal




As we can see, with equal variations, both versions of the t-test show the same power, that is, equal variations are a softer condition. At the same time, increasing the sample leads to the ability to resolve increasingly closer datasets (with a smaller delta), so with a datasets size of 110 values, already with a delta of approximately 0.6, 100% power will be achieved. With a datasets size of 10, this power was achieved only with a delta of 2.

### Task 6

Assume that datasets distributions have unequal variances and compare the false-positive rates of the two versions of the t-test.

False positive ratio is the probability of falsely rejecting the null hypothesis. The false positive rate is calculated as the ratio between the number of negative events wrongly categorized as positive (false positives) and the total number of actual negative events (regardless of classification).

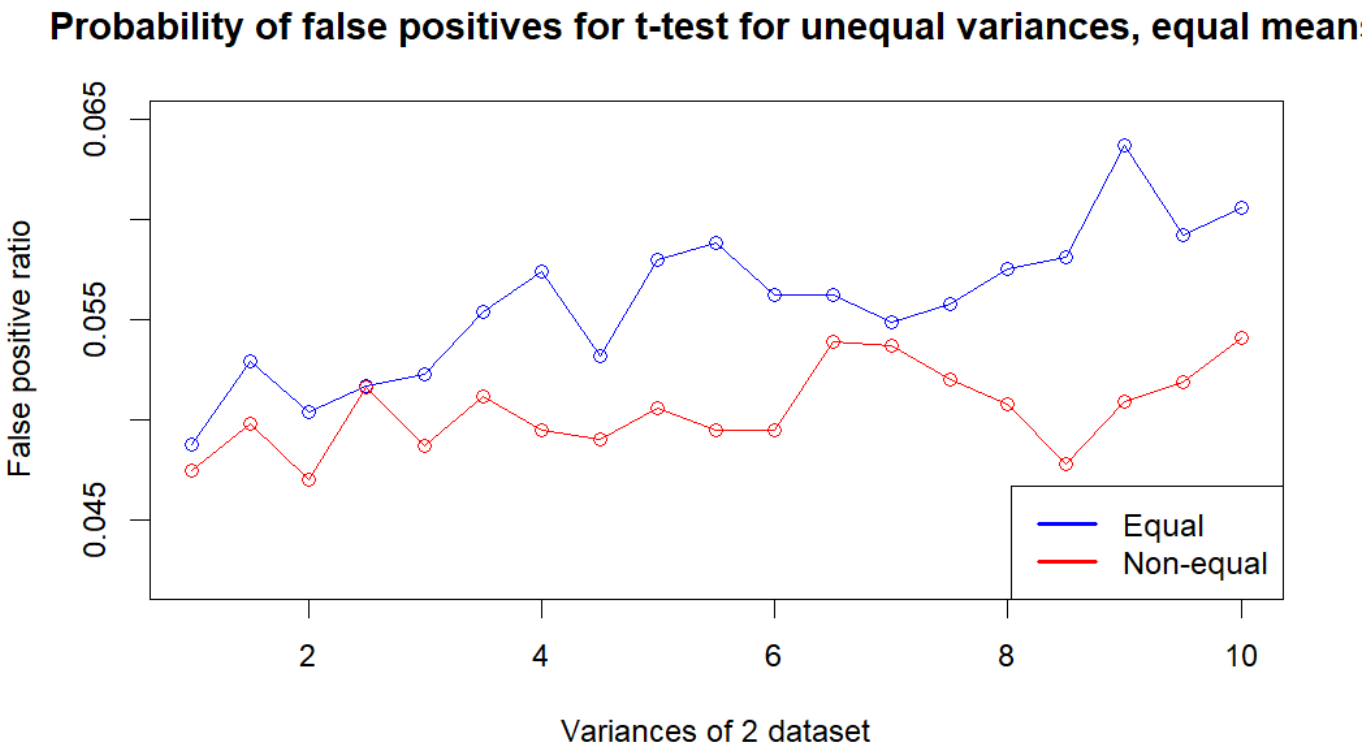
To comply with the definition, we need to have a non-zero chance of falsely rejecting the null hypothesis, and for this we need to use equal means (otherwise it will be impossible to estimate in how many cases the test result did not coincide with reality). Let's consider the false positive rate with initially equal averages d/2.

If p-value is greater than 0.05, then there is no significant difference between the means of the two groups. At the same time, we know that the sample means are completely the same. It turns out that in a situation where p-value is less than 0.05, the test made a mistake and gave an incorrect answer (namely, it rejects the null hypothesis that the means are equal), that is, such answers will be false positive.

```
false_function_equal <- function(n, d, v1, v2){
  r <- 10000
```

```
  pvalue_sum <- c()
  false <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- t.test(ds_1, ds_2, var.equal = TRUE)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }

  FP <- length(pvalue_sum[pvalue_sum < 0.05])
  false <- c(false, FP/r)

  return(false)
  }

false_function_nonequal <- function(n, d, v1, v2){
  r <- 10000
  pvalue_sum <- c()
  false <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- t.test(ds_1, ds_2, var.equal = FALSE)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }

  FP <- length(pvalue_sum[pvalue_sum < 0.05])
  false <- c(false, FP/r)

  return(false)
  }

d <- 1
n <- 10
v1 <- 1
V <- seq(1, 10, 0.5)

false_equal <- c()
false_nonequal <- c()

for (v2 in V){
    fp_equal <- false_function_equal(n, d, v1, v2)
    false_equal <- c(false_equal, fp_equal)
    fp_nonequal <- false_function_nonequal(n, d, v1, v2)
    false_nonequal <- c(false_nonequal, fp_nonequal)
  }

  plot(V, false_equal, col = 'blue', type = "o", ylim = c(0.042, 0.065),
       xlab = "Variances of 2 dataset", ylab = "False positive ratio",
       main = "Probability of false positives for t-test for unequal variances, equal means")
  lines(V, false_nonequal, col = 'red', type = "o")
```

```
  legend("bottomright", legend = c("Equal", "Non-equal"),
         col = c("blue", "red"), lwd = 2)
```




It can be noted that the test for equal variations in the majority gives a higher probability of a false positive than the test for unequal variations. That is, in the case of unequal variations, it will be more reliable to use the corresponding test for unequal variations, which will give a lower percentage of error in rejecting the null hypothesis.

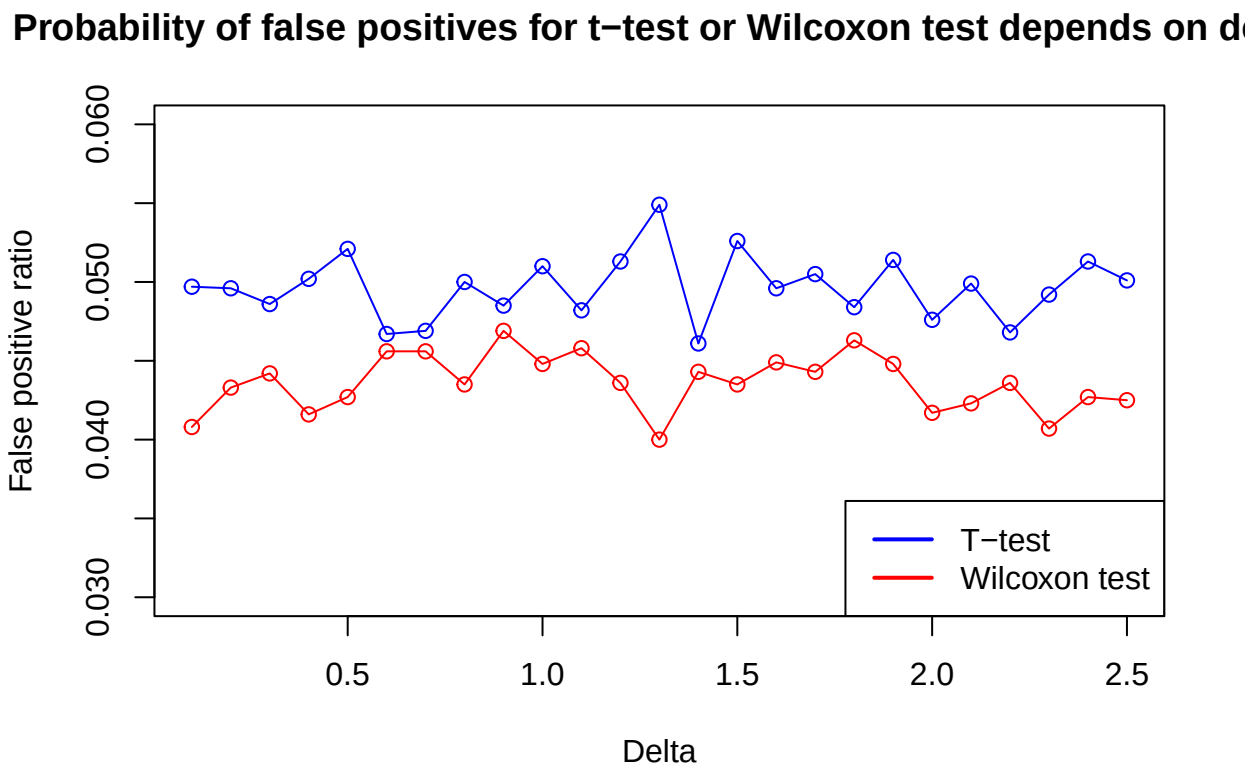
### Task 7

Returning to equal variances, compare the power and false-positive rates of the t-test and its non-parametric alternative, the Wilcoxon rank-sum test.

False-positive rate (for equal means):

```
false_function_wilcoxon <- function(n, d, v1, v2){
  r <- 10000
  pvalue_sum <- c()
  false <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- wilcox.test(ds_1, ds_2)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }
```

```
  FP <- length(pvalue_sum[pvalue_sum < 0.05])
  false <- c(false, FP/r)

  return(false)
  }


delta <- seq(0.1, 2.5, 0.1)
n <- 10
v1 <- 1
v2 <- 1

false_wilcoxon <- c()
false_ttest <- c()

for (d in delta){
    fp_ttest <- false_function_equal(n, d, v1, v2)
    false_ttest <- c(false_ttest, fp_ttest)
    fp_wilcoxon <- false_function_wilcoxon(n, d, v1, v2)
    false_wilcoxon <- c(false_wilcoxon, fp_wilcoxon)
  }


  plot(delta, false_ttest, col = 'blue', type = "o",
       ylim = c(0.03, 0.06), xlab = "Delta", ylab = "False positive ratio",
       main = "Probability of false positives for t-test or Wilcoxon test depends on delta")
  lines(delta, false_wilcoxon, col = 'red', type = "o")
  legend("bottomright", legend = c("T-test", "Wilcoxon test"),
         col = c("blue", "red"), lwd = 2)
```

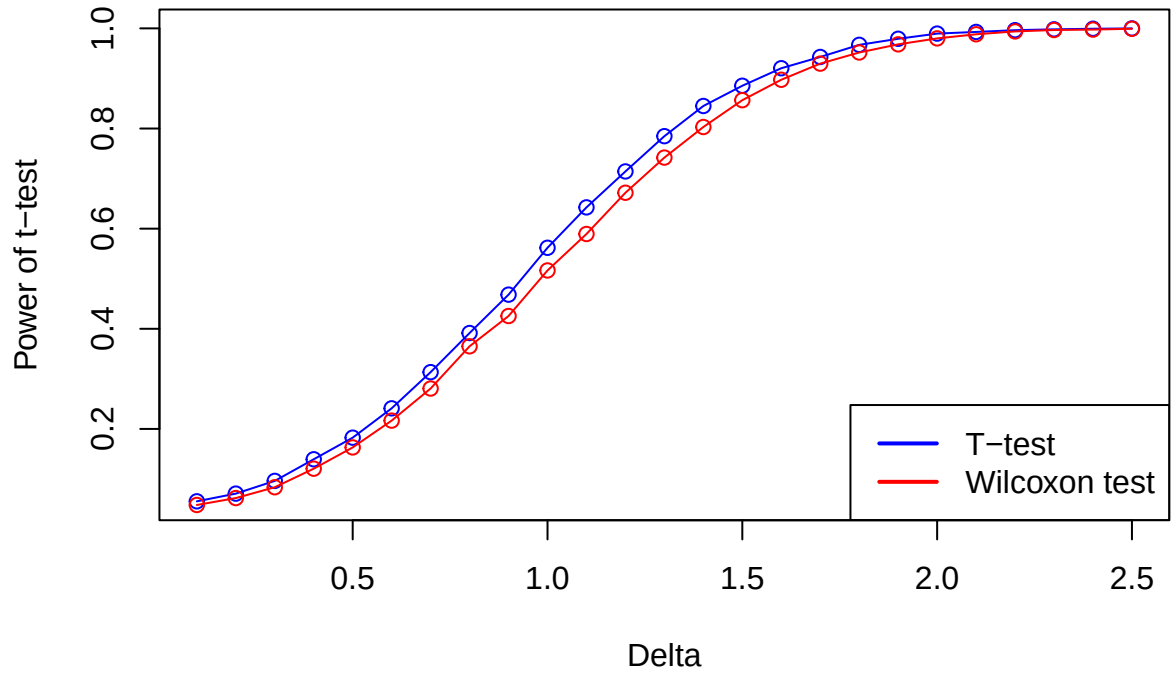



The t-test gives a higher probability of incorrectly rejecting the null hypothesis.

Power (for unequal means):

```
power_function_wilcoxon <- function(n, d, v1, v2){
  r <- 10000
  pvalue_sum <- c()
  power <- c()

  for (i in 1:r) {
    ds_1 <- rnorm(n, -d/2, sqrt(v1))
    ds_2 <- rnorm(n, d/2, sqrt(v2))
    pvalue <- wilcox.test(ds_1, ds_2)$p.value
    pvalue_sum <- c(pvalue_sum, pvalue)
  }

  TP <- length(pvalue_sum[pvalue_sum < 0.05])
  power <- c(power, TP/r)

  return(power)
  }


delta <- seq(0.1, 2.5, 0.1)
n <- 10
v1 <- 1
v2 <- 1
```

```r
power_wilcoxon <- c()
power_ttest <- c()

for (d in delta){
    tp_ttest <- power_function_equal(n, d, v1, v2)
    power_ttest <- c(power_ttest, tp_ttest)
    tp_wilcoxon <- power_function_wilcoxon(n, d, v1, v2)
    power_wilcoxon <- c(power_wilcoxon, tp_wilcoxon)
  }


  plot(delta, power_ttest, col = 'blue', type = "o",
       xlab = "Delta", ylab = "Power of t-test",
       main = "Probability of true positives for for t-test or Wilcoxon test depends on delta")
  lines(delta, power_wilcoxon, col = 'red', type = "o")
  legend("bottomright", legend = c("T-test", "Wilcoxon test"),
         col = c("blue", "red"), lwd = 2)
```




Non-parametric methods can also be used for normally distributed datasets, but they will give less accuracy than parametric ones, as can be seen from the graphs. Therefore, if you are confident in the normal distribution of the data, it is more reliable to use the parametric method. Like, for example, the t-test and wilcoxon test were compared here.