NOD_Codes

Mohit Mehndiratta

27/10/2021

Iraqi Refugees

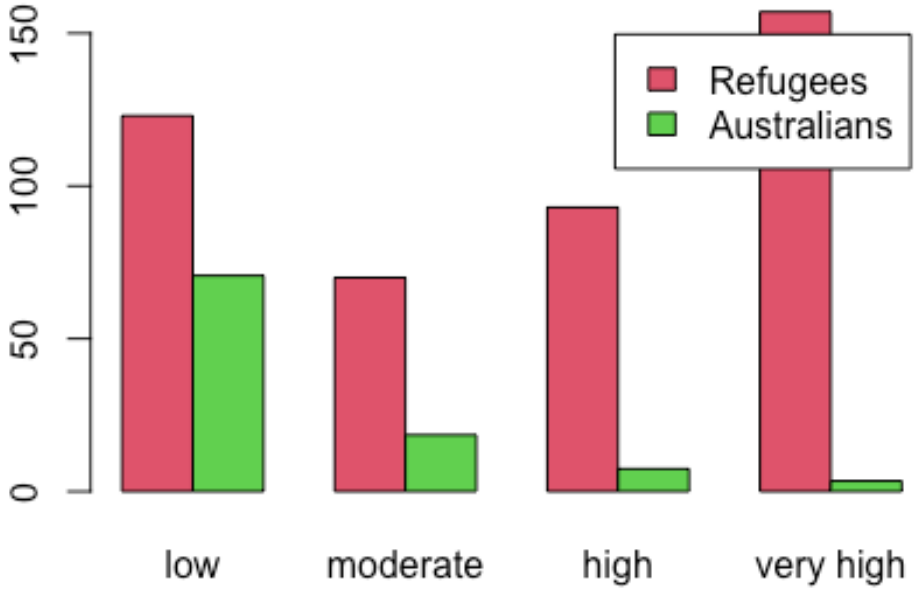
Loading the data

```
iraqi = c(123, 70, 93, 157)
aihw = c(70.65, 18.5, 7.41, 3.43)
levs = c("low", "moderate", "high", "very high")
names(iraqi) = levs
names(aihw) = levs

m <- rbind(iraqi,aihw)
m
```

```
##          low moderate  high very high
## iraqi 123.00     70.0 93.00    157.00
## aihw   70.65     18.5  7.41      3.43
```

Visualisation

```
barplot(m, beside = TRUE, col = 2:3,
        legend.text = c('Refugees', 'Australians'))
```



Hypothesis Testing

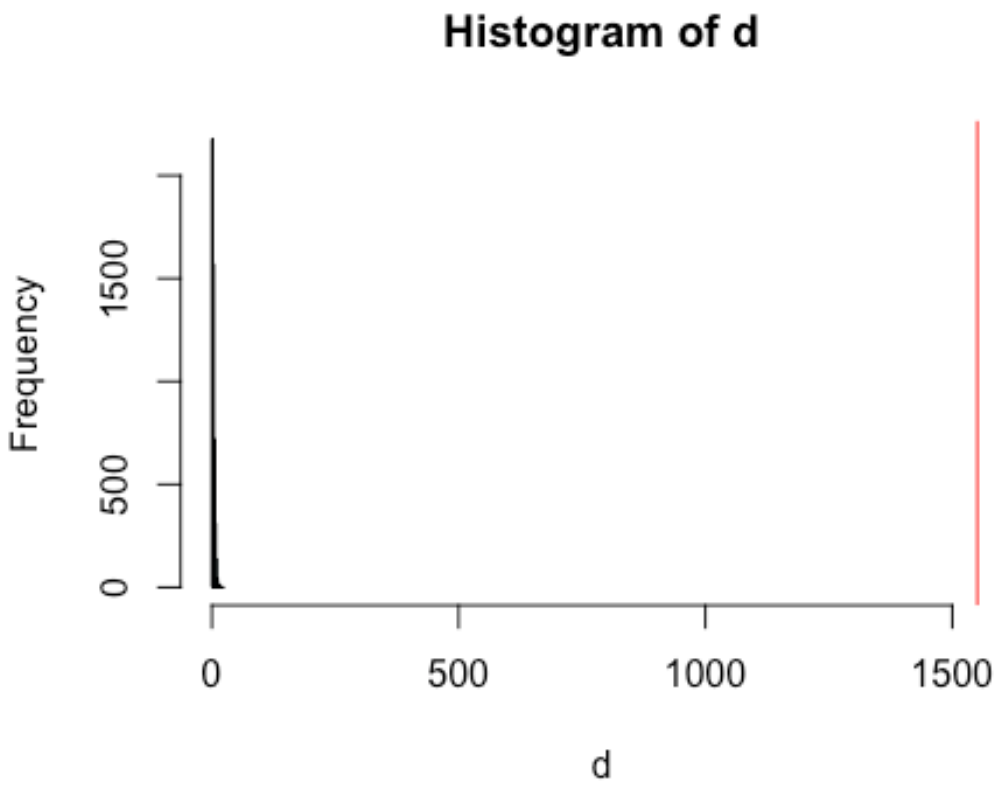
H0: There is no difference in distribution of distress between iraqi and aihw H1: There is a difference CV: 0.05

```
expected = aihw * 443 / 100
cs <- sum((iraqi - expected) ^ 2 / expected)
cs
```

```
## [1] 1550.75
```

```
d <- replicate(5000, {
  obs <- rmultinom(1, 443, expected)
  sum((obs - expected) ^2 / expected)
})
```

```
hist(d, col="lightblue", xlim = c(0,1580))
abline(v = cs, col = "red")
```




```
pVal <- mean(d > cs)
pVal
```

```
## [1] 0
```

And another method for hypothesis testing

```
chisq.test(iraqi, p = aihw, rescale.p = TRUE, simulate.p.value = TRUE,
B = 5000)
```

```
##
##  Chi-squared test for given probabilities with simulated p-value
(based
##  on 5000 replicates)
```

```
##
## data:  iraqi
## X-squared = 1550.6, df = NA, p-value = 2e-04
```

Eye Color:

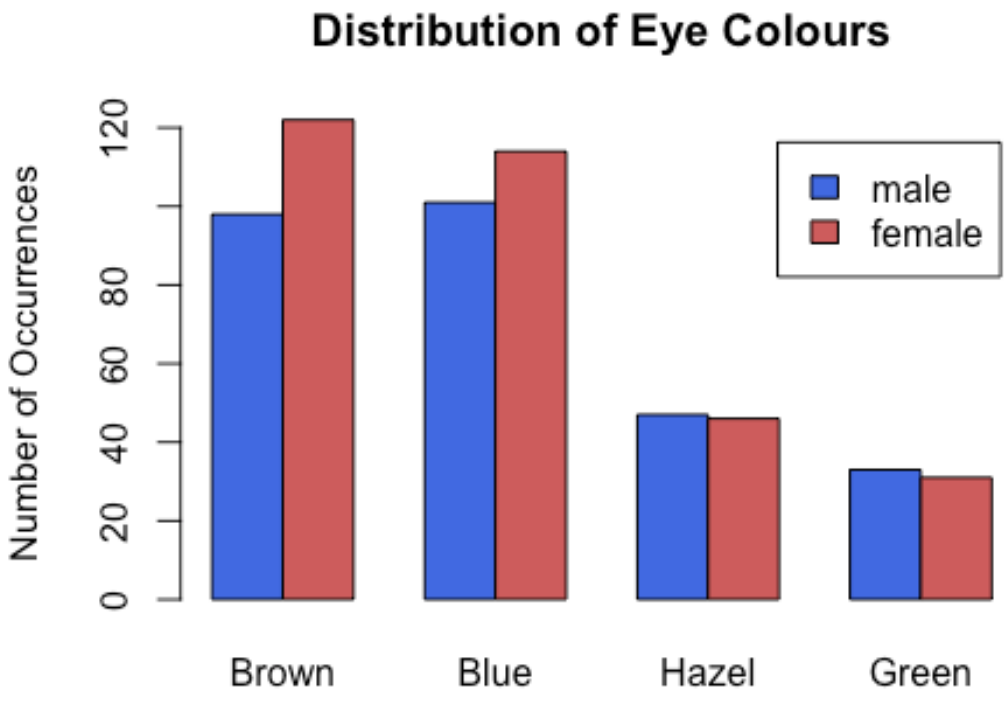
Loading the data:

```
male = c(98, 101, 47, 33)
female = c(122, 114, 46, 31)
levs = c("Brown", "Blue", "Hazel", "Green")
names(male) = levs
names(female) = levs

m <- rbind(male,female)
m
```

```
##        Brown Blue Hazel Green
## male      98  101    47    33
## female   122  114    46    31
```

Visualisation

```
barplot(m, beside = TRUE,col = c("Royalblue", "indianred"),
        main = "Distribution of Eye Colours",
        ylab = "Number of Occurrences",
        legend = TRUE)
```



simulation

```
chisq.test(m, rescale.p = TRUE, simulate.p.value = TRUE, B = 5000)

##
##  Pearson's Chi-squared test with simulated p-value (based on 5000
##  replicates)
##
## data:  m
## X-squared = 1.5298, df = NA, p-value = 0.6801
```

EELS

Loading the data:

```
eels <- matrix(c(264, 161, 127, 116, 99, 67), ncol = 3)

speciesLabels <- c('G.moringa', 'G.vicinus')
locationLabels <- c('Border', 'Grass', 'Sand')

dimnames(eels) <- list(species = speciesLabels,
                       location = locationLabels)
eels
```

```
##            location
## species     Border Grass Sand
##   G.moringa    264   127   99
##   G.vicinus    161   116   67
```

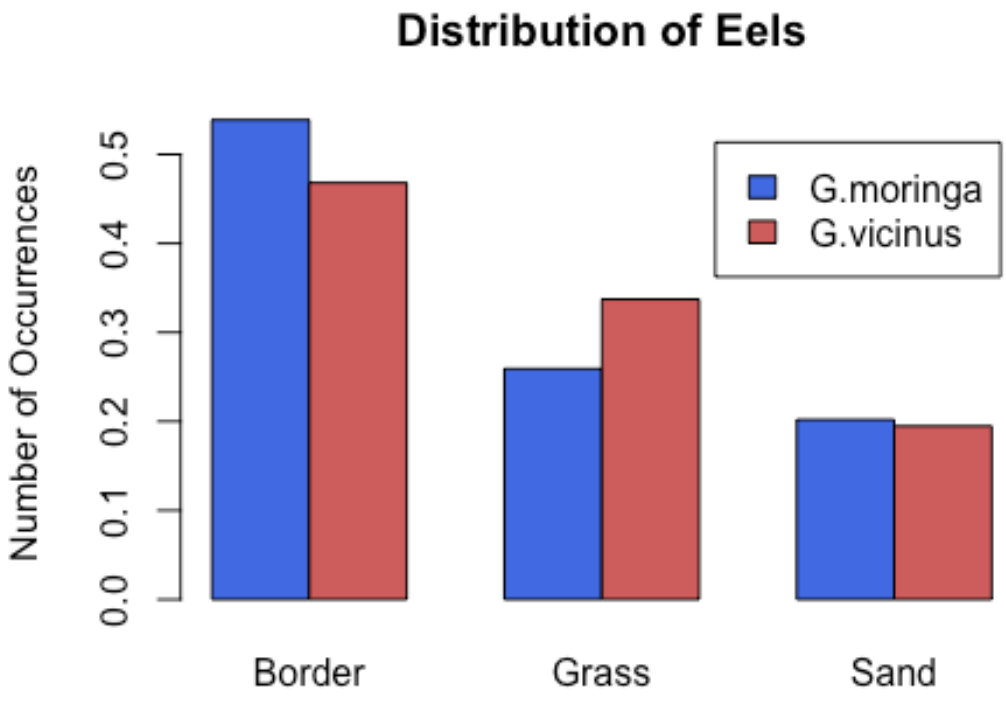
```
sampleSize = sum(eels)
```

Visualisation:

```
eels1 = eels ## Proportions
eels1[1,] = eels[1,] / sum(eels[1,])
eels1[2,] = eels[2,] / sum(eels[2,])
eels1
```

```
##            location
## species        Border     Grass      Sand
##   G.moringa 0.5387755 0.2591837 0.2020408
##   G.vicinus 0.4680233 0.3372093 0.1947674
```

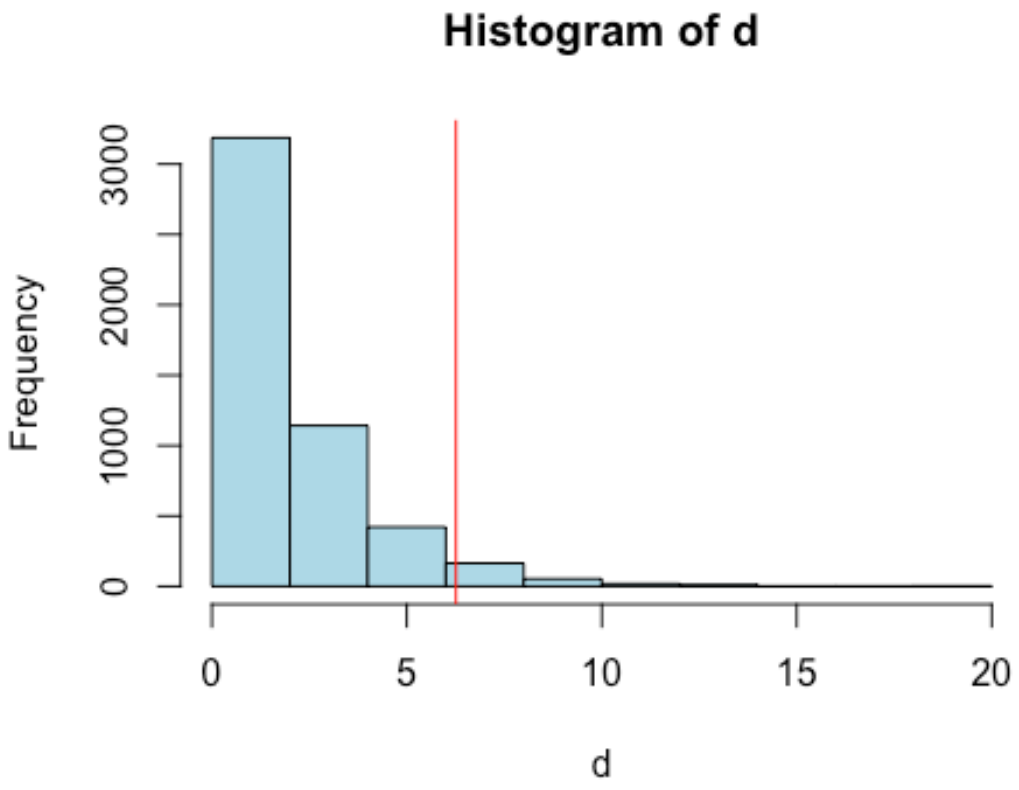
```
barplot(eels1, beside = TRUE,col = c("Royalblue", "indianred"),
        main = "Distribution of Eels",
        ylab = "Number of Occurrences",
        legend = TRUE)
```



Simulation

```
speciesCount = rowSums(eels)
locationProps = colSums(eels) / sampleSize

# our expected distribution
exp <- outer(speciesCount, locationProps)

cs <- sum((eels - exp)^2 / exp)
# Simulate / the distribution of differences
#
# simulate the data, assuming that the species
# does not effect the location
speciesProps <- speciesCount / sampleSize

d <- replicate(5000, {
```

```
  # sample of species
  sp <- sample(speciesLabels,
               size = sampleSize,
               replace = TRUE,
               prob = speciesProps)

  # sample of locations
  lc <- sample(locationLabels,
               size = sampleSize,
               replace = TRUE,
               prob = locationProps)

  # tabulate the results
  res <- table(sp, lc)


  # re-compute the expected
  r <- rowSums(res)
  c <- colSums(res) / sum(res)

  ex <- outer(r, c)


  # compute diff between sample and expected
  sum((res - ex)^2 / ex)
})

hist(d, col="lightblue")
abline(v = cs, col = "red")
```




```
pVal <- mean(d > cs)
pVal
```

```
## [1] 0.0444
```

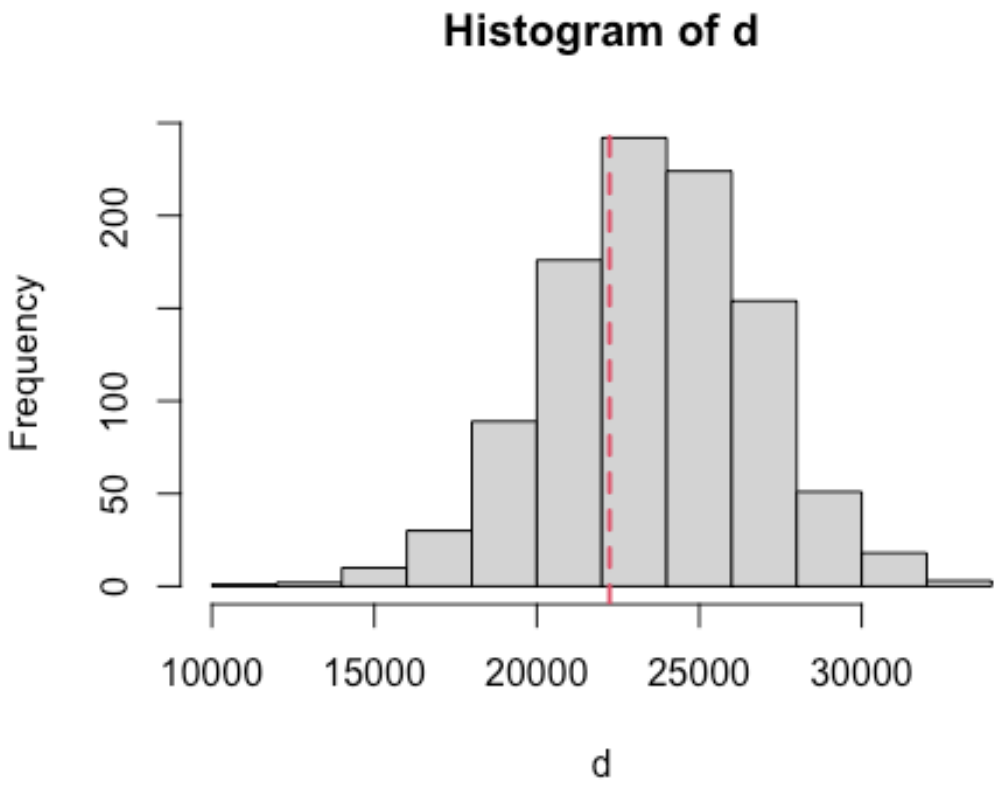
```
chisq.test(eels, simulate.p.value = TRUE, B = 5000)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 5000
##  replicates)
##
## data:  eels
## X-squared = 6.2621, df = NA, p-value = 0.04899
```

Card Piles

Simulation

```
x1 <- sample(1:52)
x2 <- sample(1:52)

# x1 <- x2 with this commented out, x1 and x2 are different, otherwise
identical

cs <- sum((x1 - x2)^2)

# Simulate what is supposedly random
d <- replicate(1000,
               {
                 a <- sample(1:52)
                 b <- sample(1:52)

                 sum((a - b)^2)
               })

hist(d)
abline(v = cs, col = 2, lwd = 2, lty = 2)
```



Birth Weight

Loading the data

```
birthWeight <- read.csv("../datasets/birthwt.csv")
head(birthWeight)
```

```
##    bwt smoke
## 1 3429    no
## 2 3229    no
## 3 3657   yes
## 4 3514    no
## 5 3086   yes
## 6 3886    no
```

```
table(birthWeight$smoke)
```

```
##
##  no yes
## 742 484
```

```
summary(birthWeight)
```

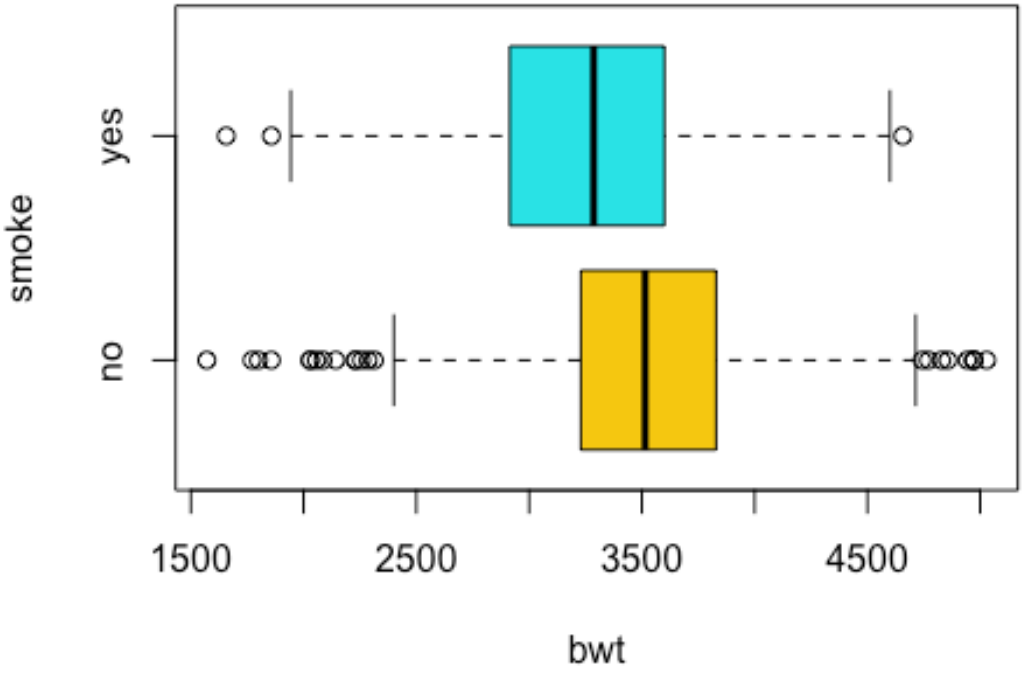
```
##       bwt          smoke
##  Min.   :1571   Length:1226
##  1st Qu.:3114   Class :character
##  Median :3429   Mode  :character
##  Mean   :3415
##  3rd Qu.:3743
##  Max.   :5029
```

```
aggregate(bwt~smoke, birthWeight, mean)
```

```
##   smoke      bwt
## 1    no 3515.639
## 2   yes 3260.285
```

Visualisation

```
boxplot(bwt~smoke, birthWeight, col = c(7,5), horizontal = TRUE)
```



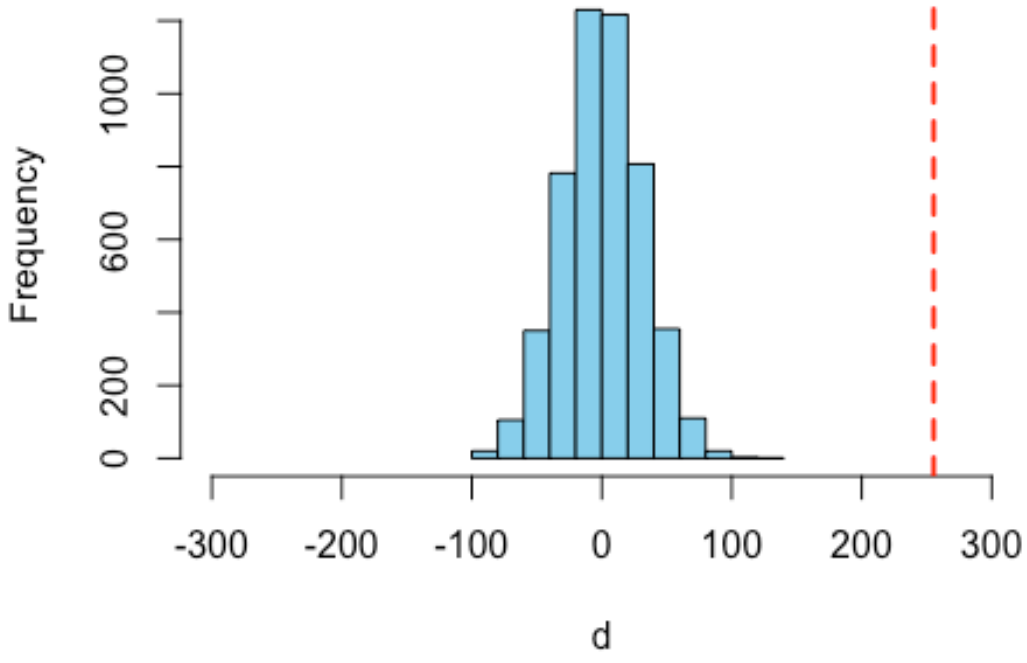
Hypothesis Testing

```
# H0: mu1 == mu2, There is no difference
# H1: mu1 < mu2, There is a difference

delta <- aggregate(bwt ~ smoke, birthWeight, mean)$bwt
delta
```

```
## [1] 3515.639 3260.285
```

```
cs <- -diff(delta)

d <- replicate(5000, {
  smoke.shuffle <- sample(birthWeight$smoke)
  del <- aggregate(bwt~smoke.shuffle, birthWeight, mean)$bwt
  -diff(del)
})
```

```r
hist(d, main = '', col = 'skyblue', xlim = c(-1, 1) * 300)
abline(v = cs, col = "red", lwd = 2, lty = 2)
```




```r
# One sided test, so p-value calculation uses one side of the
distribution
# The +ve side is used since mu2 - mu1 > 0
pVal <- mean(d > cs)
```

Drugs Data

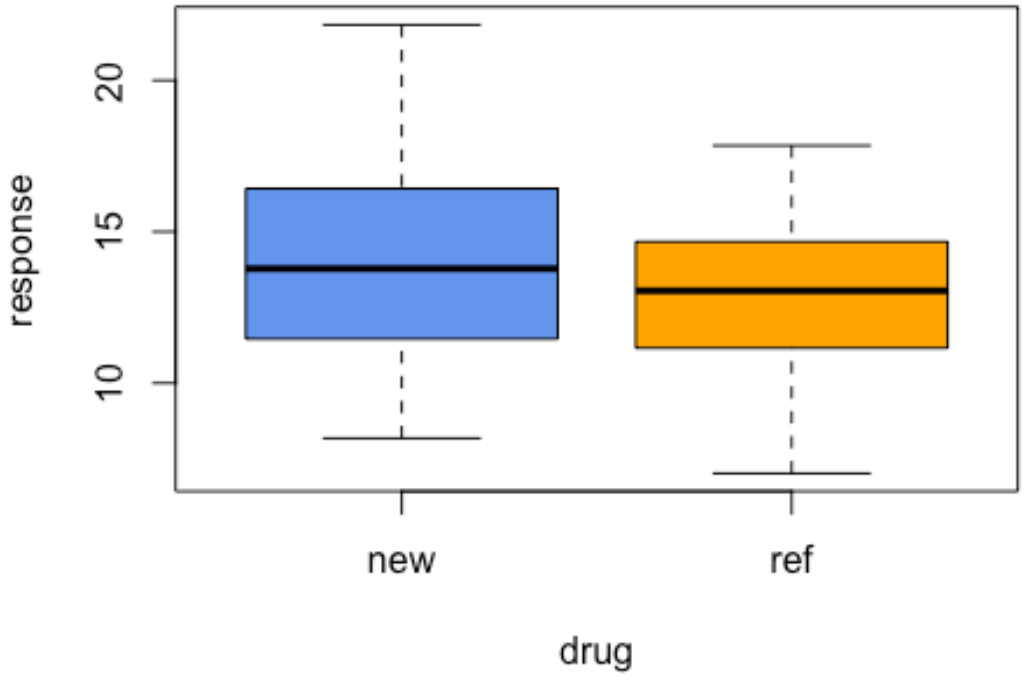
Loading the data

```r
# Loading the drug data from file "assignmentB_drugData.csv".
drug_data <- read.csv("../datasets/drugs.csv")
head(drug_data) # viewing the first 6 rows of the data
```

```
##    response drug
## 1  8.661020  new
## 2 11.431452  new
## 3 13.904322  ref
## 4  8.300664  new
## 5 14.662067  new
## 6  9.971347  ref
```

Visualisation

```
## Visualisation
boxplot(response~drug, data = drug_data, pch = 16,
        col = c("cornflowerblue","orange"))
```



Hypothesis Testing

```
# For the Hypothesis, we are using null and alternate hypothesis as
follows:

# H0: mu1 == mu2, There is no statistically significant improvement
exist for new drug over ref drug.
# H1: mu1 > mu2, There exists a statistically significant improvement
for new drug over ref drug.
# CV = 0.05 (5%)

cv <- 0.05 # critical value
replications <- 5000 # number of replications for simulation

# calculating mean data for new drug and ref drug
delta <- aggregate(response ~ drug, drug_data, mean)$response

# Calculating difference in means
cs <- -diff(delta)

cat("Difference in means of new drugs and ref drugs is:", cs, "\n")
```

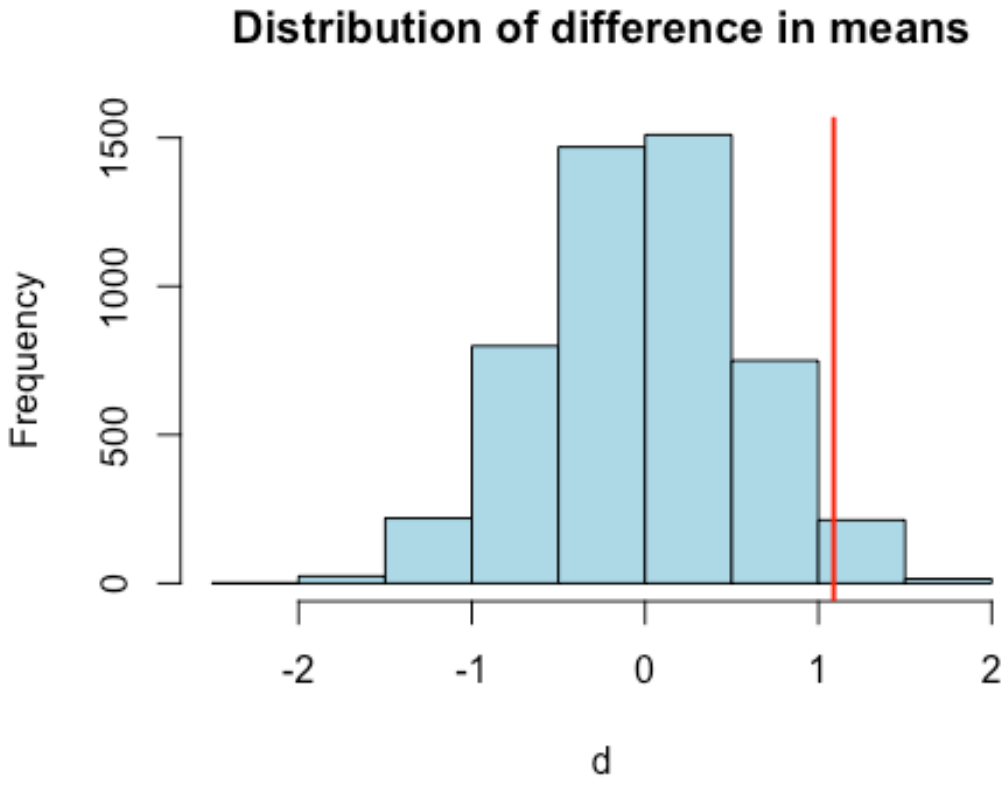
```
## Difference in means of new drugs and ref drugs is: 1.088462
```

```
# setting the seed value.
set.seed(2)
# Simulating the difference in means by 5000 times for the shuffled
drug categories within same data.
d <- replicate(replications, {
  shuffled_drug <- sample(drug_data$drug) ## Shuffling the drug
categories
  delta <- aggregate(response ~ shuffled_drug, drug_data, mean)
$response
  c <- -diff(delta)
})

cat("Average difference in means for simulated results",mean(d),
"\n\n")
```

```
## Average difference in means for simulated results -0.009275686
```

```
# Visualising the simulated outcome
hist(d, col = "lightblue", main = "Distribution of difference in
```

```
means")
abline(v = cs, col = "red", lwd = 2)
```




```
# counting the number of replications which have bigger difference
than original difference.
count <- sum(d > cs)

# calculating p-value
pvalue <- count/replications
cat("Calculated p-value from the simulation:", pvalue, "\n\n")

## Calculated p-value from the simulation: 0.0348

cat("Is critical value greater than p-value?", cv > pvalue, "\n")

## Is critical value greater than p-value? TRUE
```

```
## if true then reject the null hypothesis.
```

Spider Data set

Hypothesis testing

```
spider <- read.csv("../datasets/Spider.csv")
head(spider)
```

```
##     Group Anxiety
## 1 Picture      30
## 2 Picture      35
## 3 Picture      45
## 4 Picture      40
## 5 Picture      50
## 6 Picture      35
```

```
table(spider$Group)
```

```
##
##     Picture Real Spider
##          12          12
```

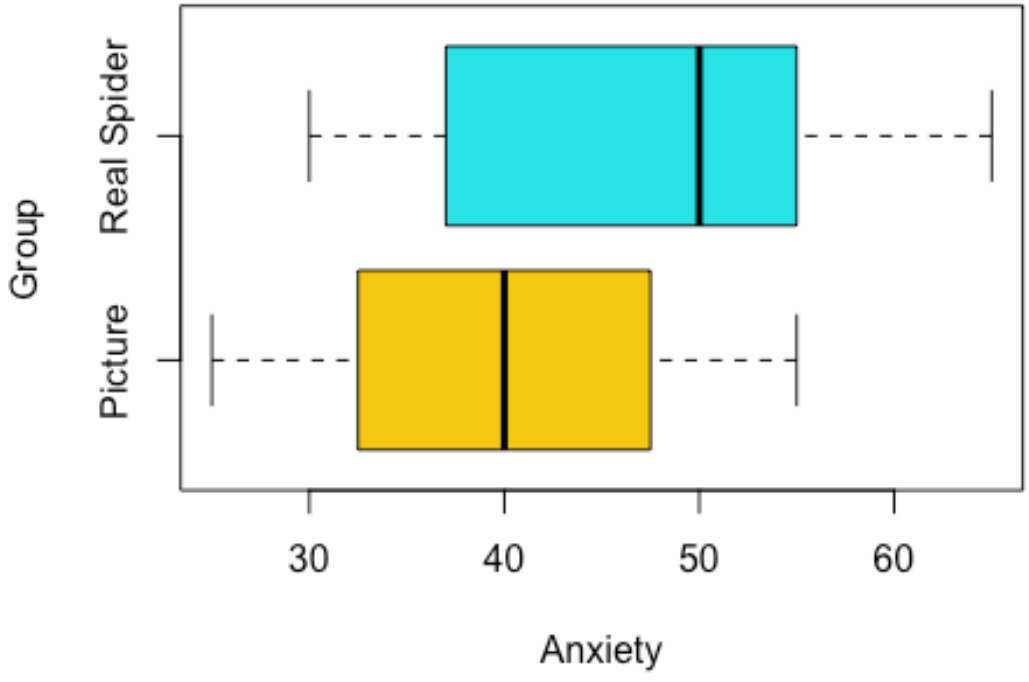
```
summary(spider)
```

```
##     Group               Anxiety
##  Length:24          Min.   :25.0
##  Class :character   1st Qu.:35.0
##  Mode  :character   Median :42.5
##                     Mean   :43.5
##                     3rd Qu.:50.0
##                     Max.   :65.0
```

```
aggregate(Anxiety~Group, spider, mean)
```

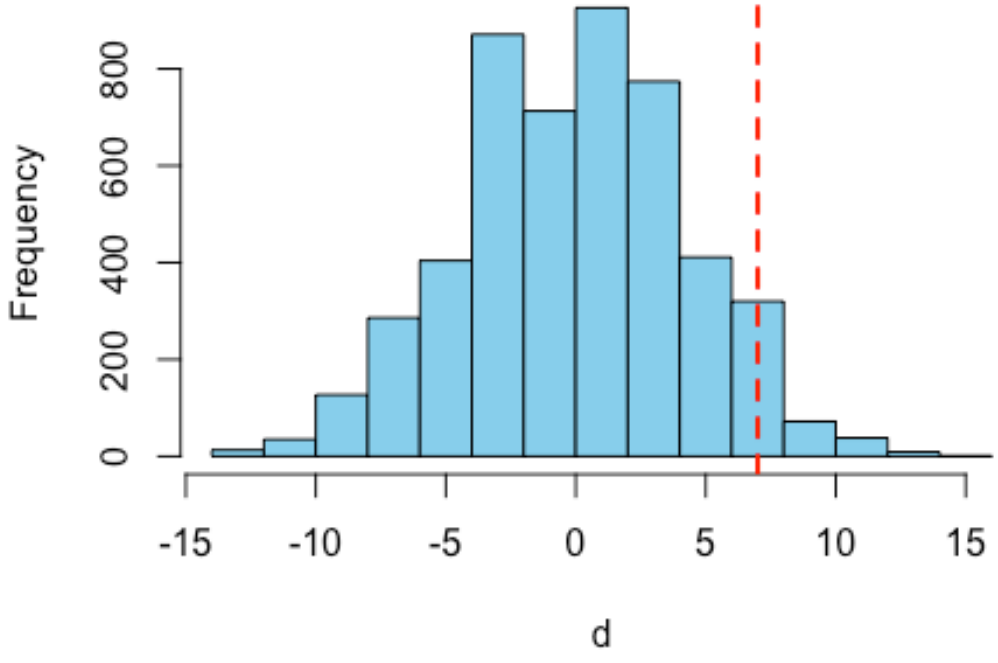
```
##         Group Anxiety
## 1     Picture      40
## 2 Real Spider      47
```

```
boxplot(Anxiety~Group, spider, col = c(7,5), horizontal = TRUE)
```




```
# H0: mu1 == mu2, There is no difference
# H1: mu1 < mu2, There is a difference

delta <- aggregate(Anxiety~Group, spider, mean)$Anxiety
delta

## [1] 40 47

cs <- diff(delta)

d <- replicate(5000, {
  Group.shuffle <- sample(spider$Group)
  del <- aggregate(Anxiety~Group.shuffle, spider, mean)$Anxiety
  diff(del)
})
```

```r
hist(d, main = '', col = 'skyblue')
abline(v = cs, col = "red", lwd = 2, lty = 2)
```




```r
# One sided test, so p-value calculation uses one side of the
distribution
# The +ve side is used since mu2 - mu1 > 0
pVal <- mean(d > cs)
```

Wilcoxon and Confidence Interval example

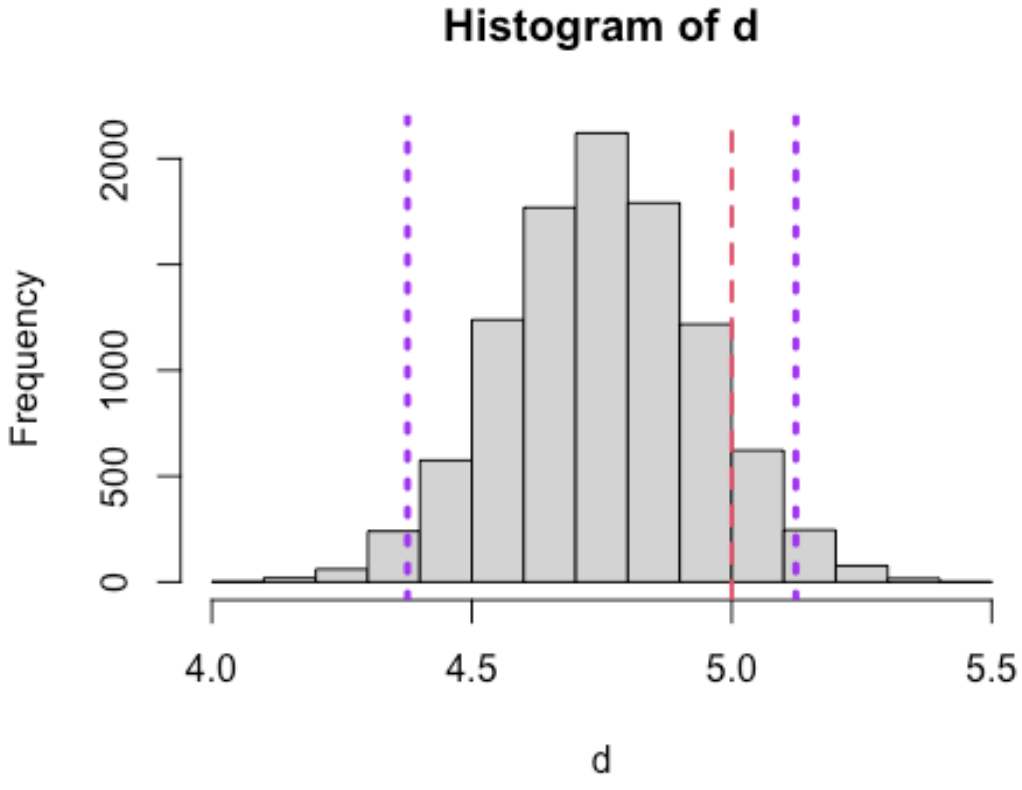
```r
# Confidence intervals
# The luxury of a known populations; use normal distribution
x <- rnorm(50, 15)  # actual pop mean = 15
y <- rnorm(50, 10)  # actual pop mean = 10
# difference in means is actually 5
# but we pretend we don't know that!
```

```r
wilcox.test(x, y)                     # Are they different; absolutely

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  x and y
## W = 2500, p-value < 2.2e-16
## alternative hypothesis: true location shift is not equal to 0

wilcox.test(x, y, alternative = 'l')  # Is x < y, no way

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  x and y
## W = 2500, p-value = 1
## alternative hypothesis: true location shift is less than 0

wilcox.test(x, y, alternative = 'g')  # Is x > y, absolutely

##
##  Wilcoxon rank sum test with continuity correction
##
## data:  x and y
## W = 2500, p-value < 2.2e-16
## alternative hypothesis: true location shift is greater than 0

# Generate estimate of population difference in means
p <- mean(x) - mean(y)
p

## [1] 4.752681

# Determine a confidence interval for the difference between x and y
# Generate confidence interval for true difference in population means
d <- replicate(10000,
               {
                 ix <- sample(1:length(x), replace = TRUE)
                 iy <- sample(1:length(y), replace = TRUE)

                 mean(x[ix]) - mean(y[iy])
               })
```

```r
hist(d)
abline(v = 5, col = 2, lwd = 2, lty = 2)

# calculate 95% confidence interval
q <- quantile(d, c(0.025, 0.975))
q
```

```
##     2.5%    97.5%
## 4.376405 5.123247
```

```r
abline(v = q, col = 'purple', lwd = 3, lty = 3)
```

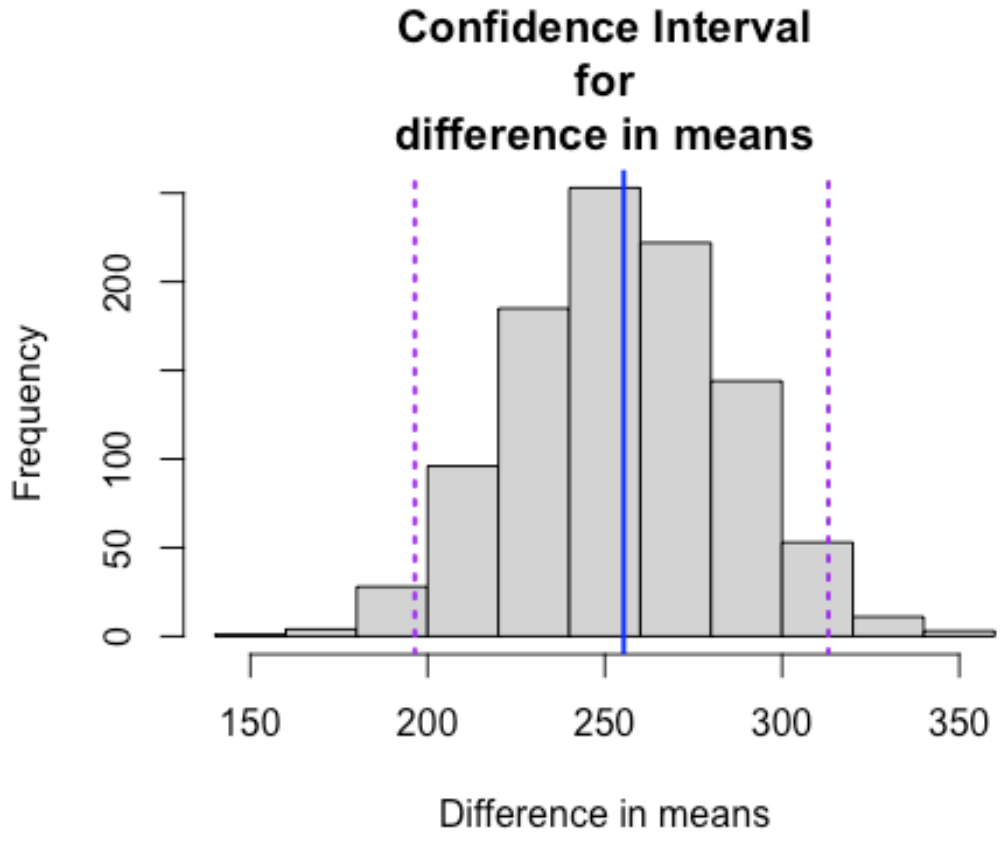


Confidence interval - Birth Weight

```
# Confidence interval for maternal smoking dataset
df <- read.csv('../datasets/birthwt.csv')

# Extract entire data set into the two groups
no <- subset(df$bwt, df$smoke == 'no')
yes <- subset(df$bwt, df$smoke == 'yes')

# Difference means for the sample;
# estimate of difference in population
res <- mean(no) - mean(yes)

## Alternal approach: delta <- aggregate(bwt~smoke, df, mean)
# resample using "boostrapping"
d <- replicate(1000,
               {
                 # bootstrapping means doing the following:
                 #  - create new sample of same size as original
                 #  - must use replacement;
                 #    otherwise we generate a shuffled original!
                 #  - must preserve group sizes; 472 no smoke, 484
smoke
                 #    so doing the two samples! Don't want to induce
differences
                 ns <- sample(no, replace = TRUE)
                 s <- sample(yes, replace = TRUE)

                 # difference in means
                 mean(ns) - mean(s)
               })

# find boundaries for central 95% of the data
q <- quantile(d, c(0.025, 0.975))
q     # conf interval, range within which true pop. difference is
expected to reside
```

```
##     2.5%    97.5%
## 196.4152 313.0799
```

```r
# visualise as matter of interest
hist(d,
     main = 'Confidence Interval\nfor\ndifference in means',
     xlab = 'Difference in means')
abline(v = res, col = 'blue', lwd = 2) # point estimate of pop mean
abline(v = q, col = 'purple', lwd = 2, lty = 3) # show confidence
interval
```



Binomial Confidence Interval - Method 1

```r
# Bootstrap binomial confidence intervals for true rate in pop.
# Method 1
germinate <- 1
notGerminate <- 0
```

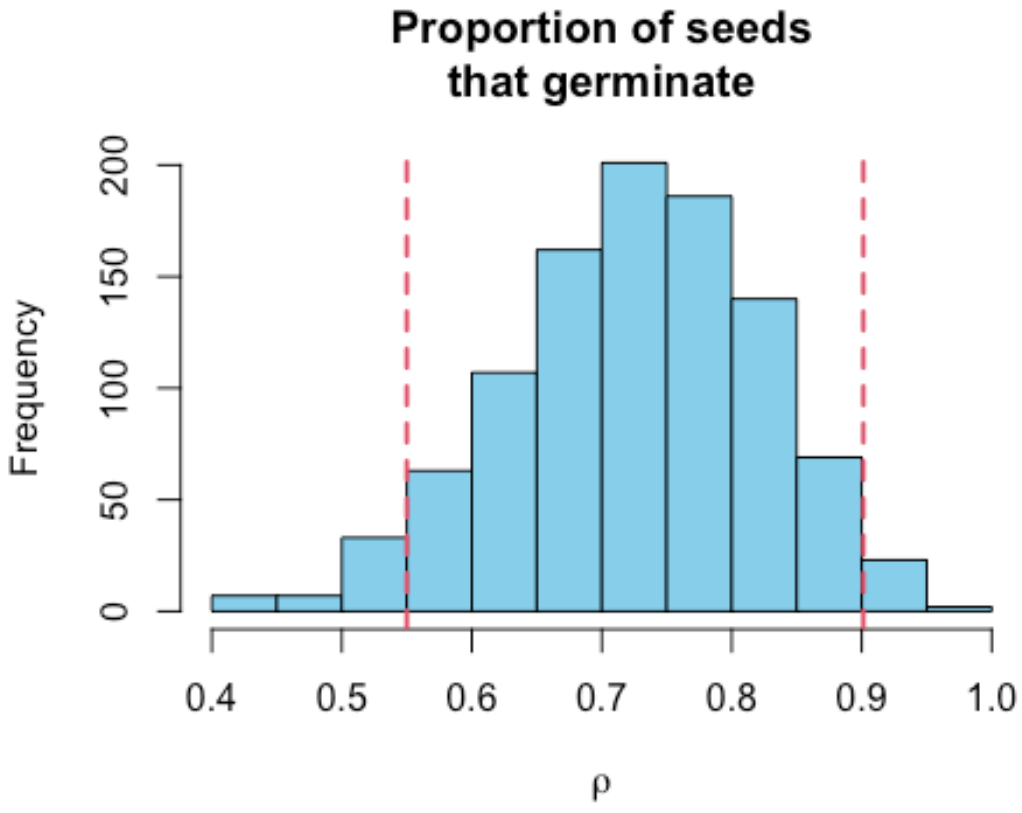
```r
seeds <- c(rep(germinate, 15),
           rep(notGerminate, 5))

d <- replicate(1000,
               {
                 res <- sample(seeds, replace = TRUE)
                 mean(res)
               })
mean(d)
```

```
## [1] 0.74995
```

```r
q <- quantile(d, c(0.025, 0.975))
q
```

```
##    2.5%   97.5%
## 0.55000 0.90125
```

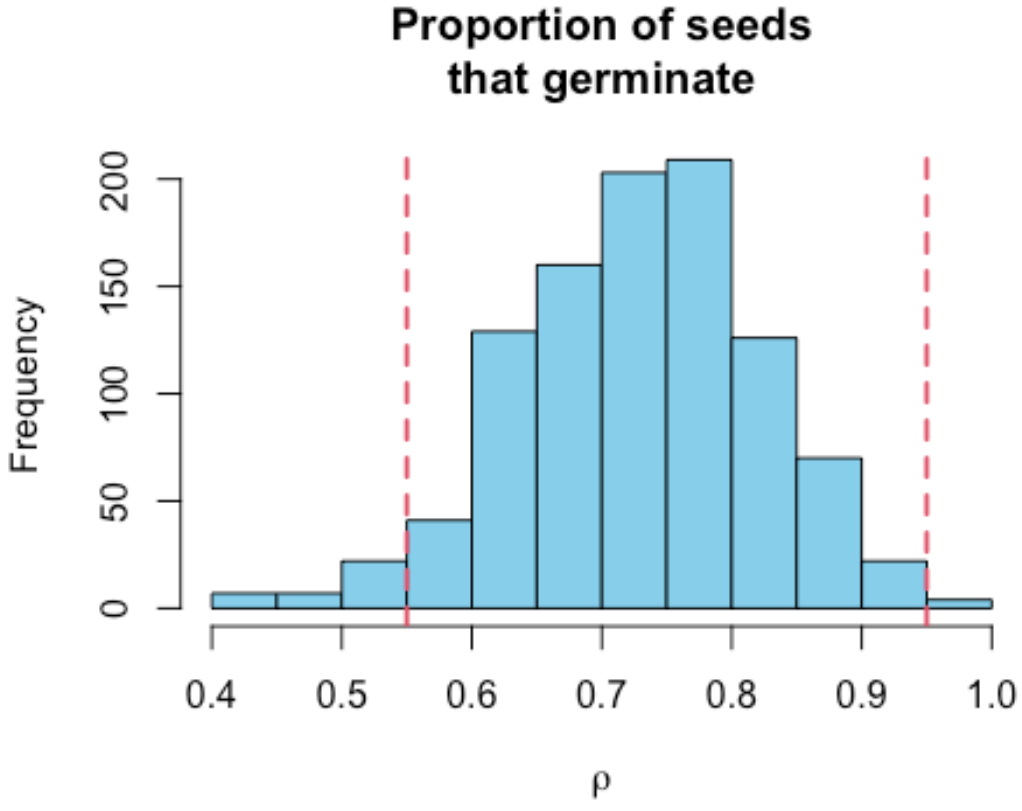
```r
# use of expression allows showing Greek letter
hist(d, col = 'skyblue', xlab = expression(rho),
     main = 'Proportion of seeds\nthat germinate')
abline(v = q, col = 2, lwd = 2, lty = 2)
```



Binomial Confidence Interval - Method 2

```
# Method 2
d <- rbinom(1000, size = 20, prob = 15/20)
d <- d / 20

mean(d)

## [1] 0.7536

q <- quantile(d, c(0.025, 0.975))
q

##  2.5% 97.5%
##  0.55  0.95
```

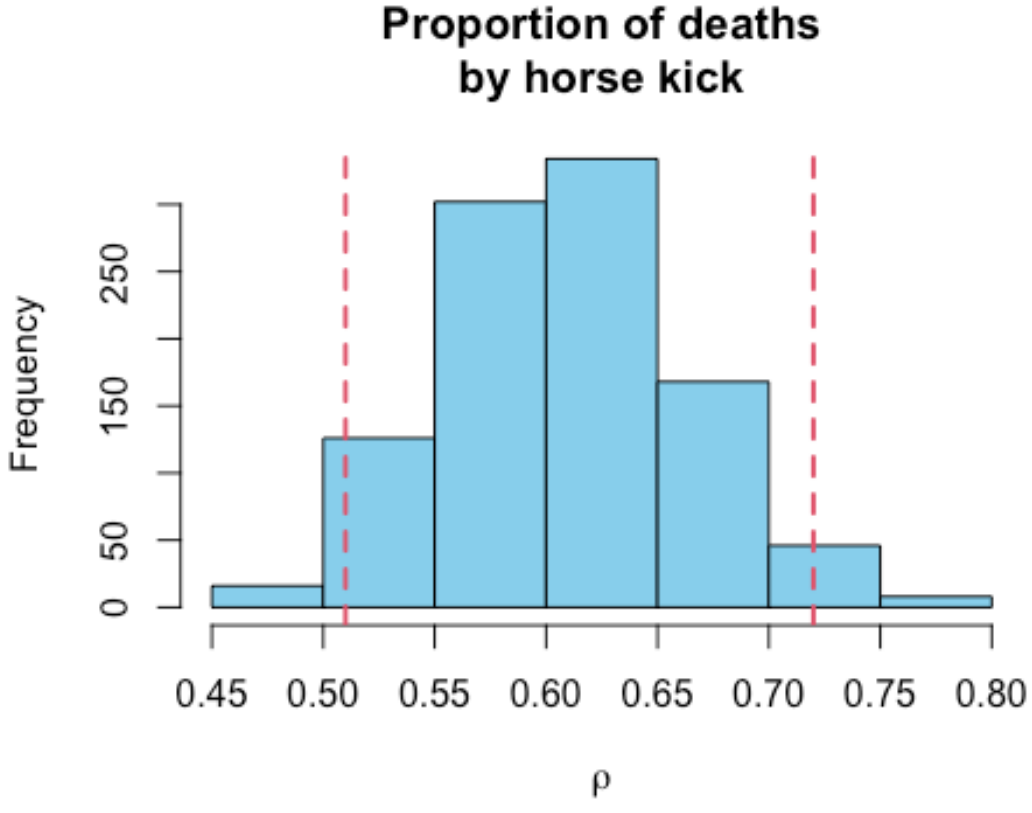
```
hist(d, col = 'skyblue', xlab = expression(rho),
     main = 'Proportion of seeds\nthat germinate')
abline(v = q, col = 2, lwd = 2, lty = 2)
```



Poison distribution confidence interval

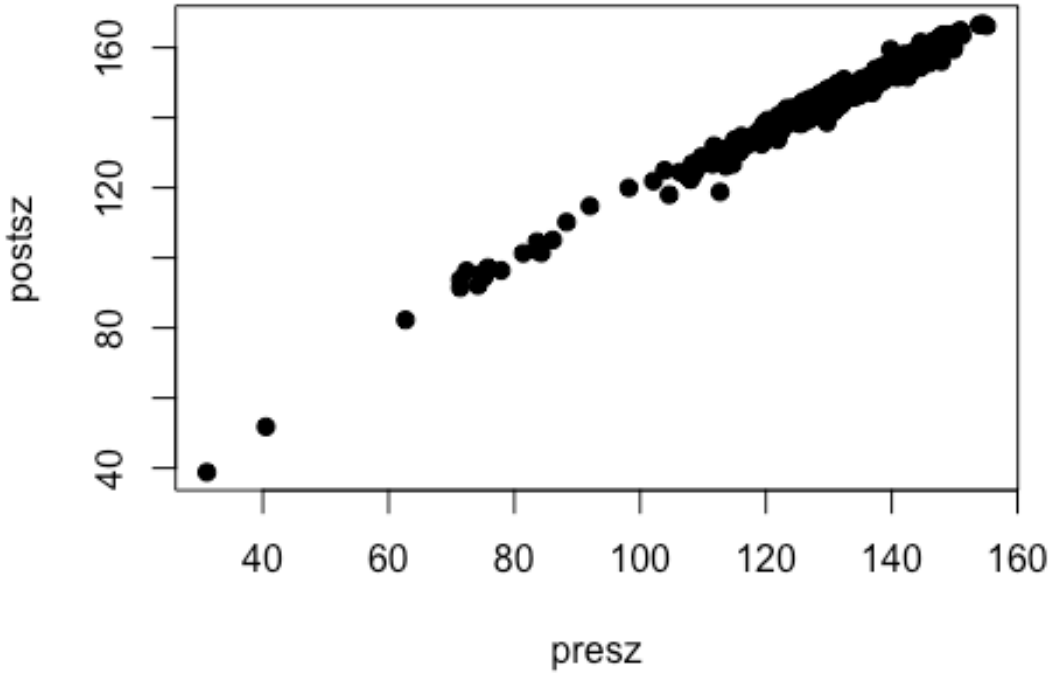
```
# Calculate Poisson confidence interval for true pop. lambda /
expected death rate
horsekick <- c(109, 65, 22, 3, 1)
deaths <- rep(0:4, horsekick)

d <- replicate(1000,
               {
                 res <- sample(deaths, replace = TRUE)
                 mean(res)
```

```
                })
mean(d)

## [1] 0.611905

q <- quantile(d, c(0.025, 0.975))
q

##  2.5% 97.5%
##  0.51  0.72

hist(d, col = 'skyblue', xlab = expression(rho),
     main = 'Proportion of deaths\nby horse kick')
abline(v = q, col = 2, lwd = 2, lty = 2)
```



Crabs data - Correlation hypothesis

```
# Slide 4
df <- read.csv('../datasets/crabsmolt.csv')

plot(df$postsz ~ df$presz, pch = 19, col = 1,
     xlab = 'presz', ylab = 'postsz')
```




```
cor(df$presz, df$postsz)
```

```
## [1] 0.9903699
```

```
obs.cor = cor(df$presz, df$postsz)
x= replicate(1000, {
  post.perm = sample(df$postsz)
  cor(df$presz, post.perm)
})
```

```
sum(abs(x) > abs(obs.cor))/1000
```

```
## [1] 0
```

```
cor.test(df$presz, df$postsz, method = "pearson")
```

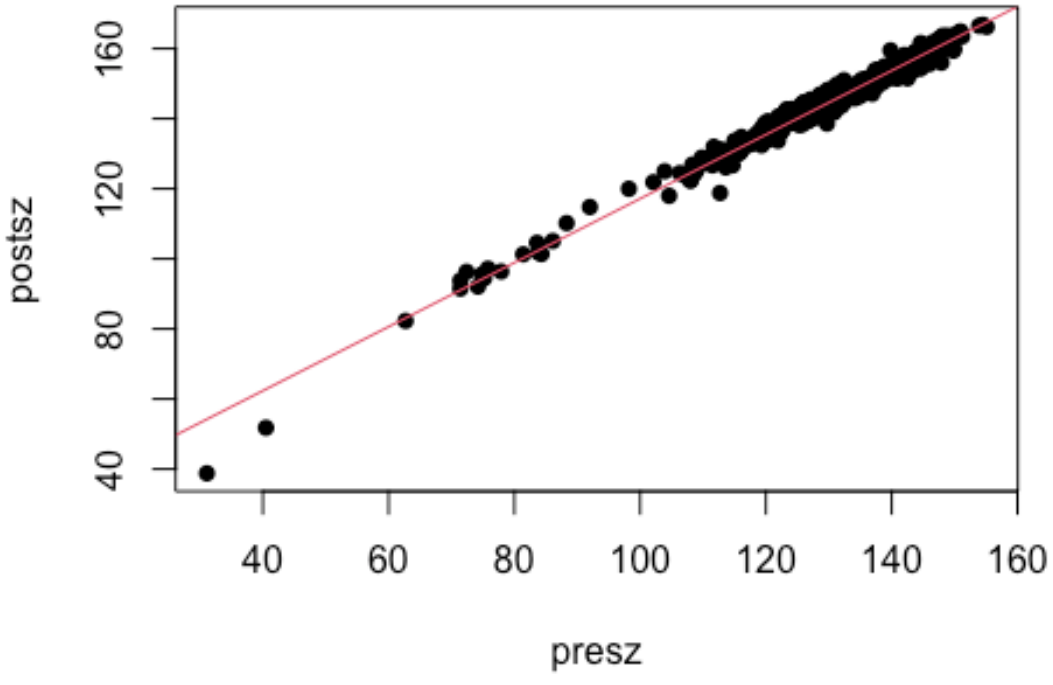
```
##
##  Pearson's product-moment correlation
##
## data:  df$presz and df$postsz
## t = 155.08, df = 470, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9884701 0.9919580
## sample estimates:
##       cor
## 0.9903699
```

Linear Models - Hypothesis testing - slope = 0

```
# Slide 4
df <- read.csv('../datasets/crabsmolt.csv')

fit = lm(postsz~presz, data = df)
summary(fit)
```

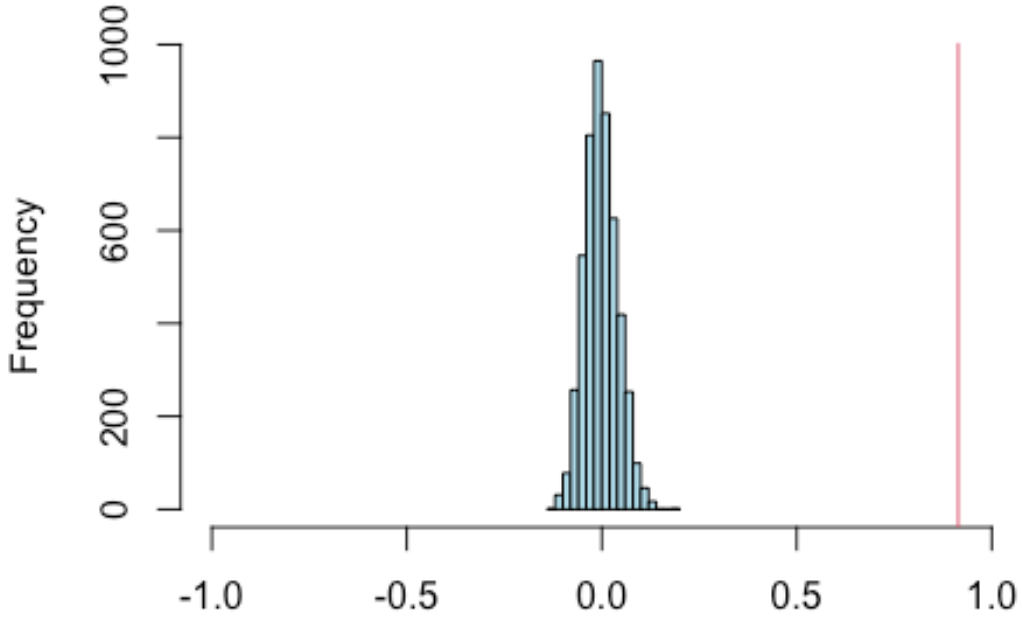
```
##
## Call:
## lm(formula = postsz ~ presz, data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.4269  -1.1611  -0.0669   1.2169   5.9251
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 25.802580   0.767201   33.63   <2e-16 ***
## presz        0.913965   0.005893  155.08   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 2.029 on 470 degrees of freedom
## Multiple R-squared:  0.9808, Adjusted R-squared:  0.9808
## F-statistic: 2.405e+04 on 1 and 470 DF,  p-value: < 2.2e-16

plot(postsz ~ presz, data=df, pch=16)
abline(fit, col = 2)
```




```
## Hypothesis for slope = 0

## compute the slope of the data
cs <- coef(fit)[2]
## compute the slope if the population b = 0
x= replicate(5000, {
  presz.perm = sample(df$presz) # shuffle one variable to force
population b = 0
```

```
  fit = lm(postsz ~ presz.perm, data=df) # fit the straight line model
  coef(fit)[2] # return the fitted b
})
## examine the distribution of b, when the population b = 0
hist(x, col="lightblue", main="", xlab="", xlim = c(-1,1))
abline(v = cs, col = 2)
```

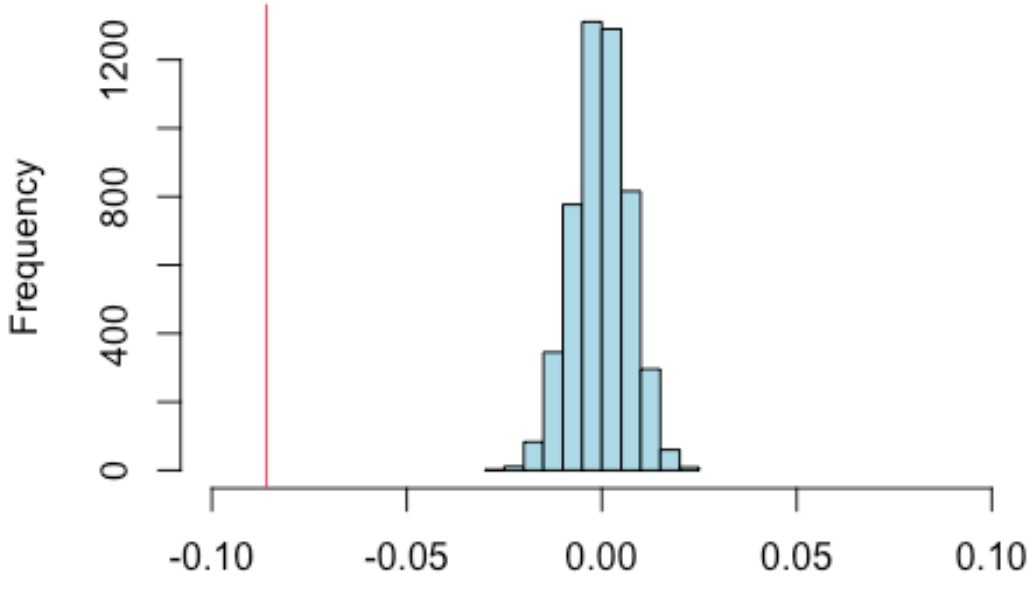



```
## compute the chance of getting the data b, if the population b = 0
(pValue = mean(x > abs(cs)) + mean(x < -abs(cs)))

## [1] 0
```

Linear Models - Hypothesis testing - Slope = 1

```
# Slide 4
df <- read.csv('../datasets/crabsmolt.csv')
```

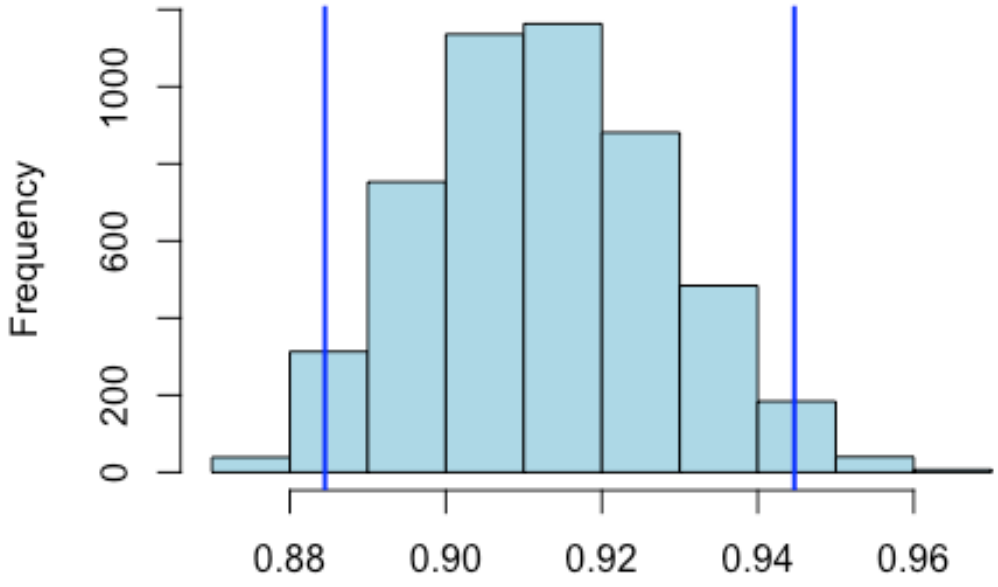
```r
## Hypothesis for slope = 1

## compute the estimate of b - 1 from the data
fit = lm((postsz - presz) ~ presz, data = df)
cs = coef(fit)[2]
## compute many sample gradients, when the population gradient is 1
x= replicate(5000, {
  presz.perm = sample(df$presz) # shuffle one variable
  fit = lm((postsz - presz) ~ presz.perm, data = df) # fit the model
  coef(fit)[2] # return the estimate of b
})
## examine the distribution of b - 1, when the population b = 1
hist(x, col="lightblue", main="", xlab="", xlim = c(-0.1,0.1))
abline(v = cs, col = 2)
```



```
## compute the chance of getting the data b, if the population b = 1
(pValue = mean(x > abs(cs)) + mean(x < -abs(cs)))

## [1] 0

## Conclusion - Assuming the slope is 1, the probability of seeing
## a slope at least this extreme by mere chance is practically 0.
## It is concluded that the slope is not equal to 1.
```

Linear Models - Confidence interval

```
# Slide 4
df <- read.csv('../datasets/crabsmolt.csv')

n = nrow(df) # store the number of observations n
## compute a set of bootstrap samples of b
x= replicate(5000, {
  samp = sample(1:n, replace = TRUE, size = n) # sample the row
numbers (with replacement)
  # fit the regression model to the selected rows (samp) of the data
  fit = lm(postsz ~ presz, data = df[samp,])
  coef(fit)[2] # extract the estimate of b
})
## examine the bootstrap distribution of b
hist(x, col = "lightblue", main = "", xlab = "")
## add the interval lines
abline(v = quantile(x,c(0.025, 0.975)), col = "blue", lwd = 2)
```

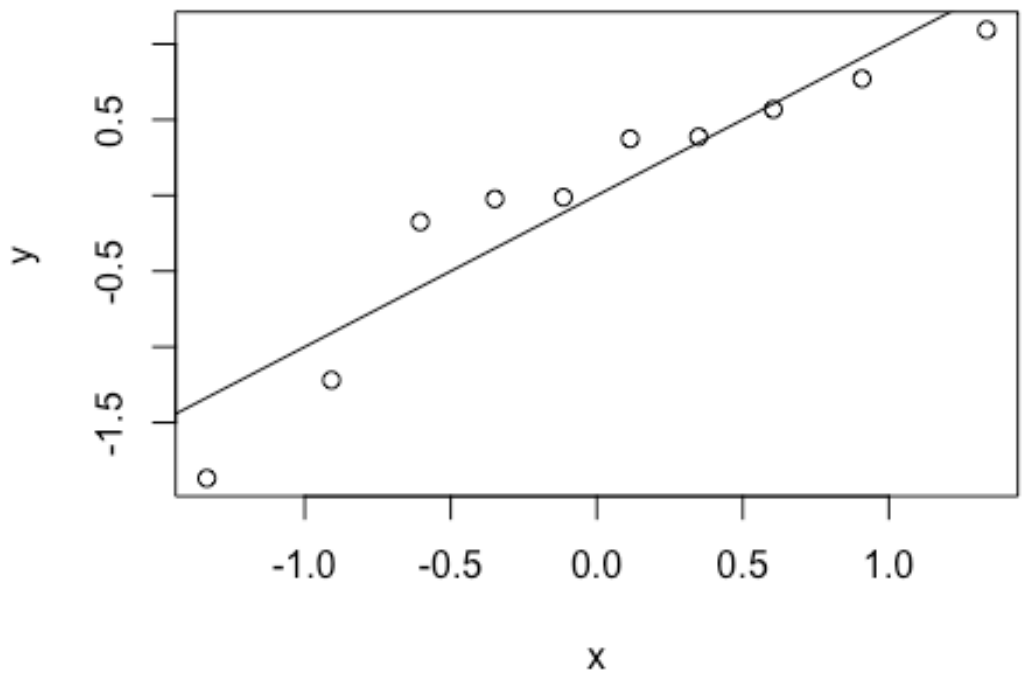



```
## print out the interval boundaries (95% interval)
quantile(x, c(0.025, 0.975))

##      2.5%     97.5%
## 0.8845042 0.9446943
```

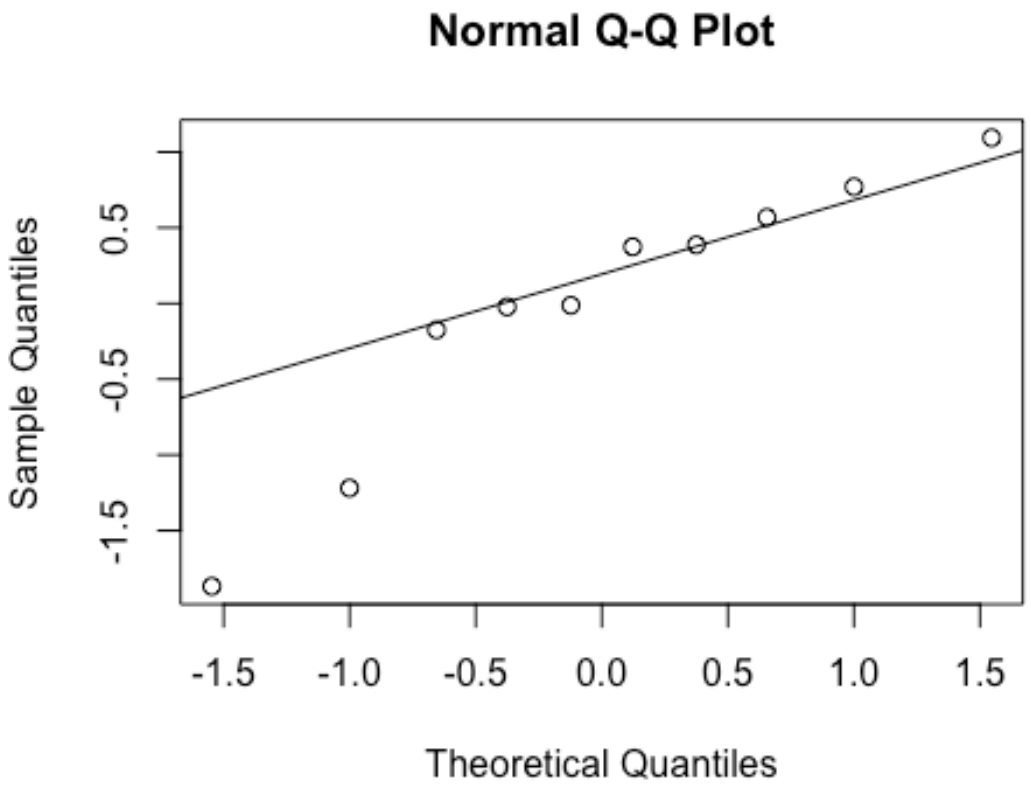
QQ - Plot

```
# Slide 12
# QQ plot
#
# y is the data being considered evaluated
# x is the normally distributed data being used as a reference

n <- 10  # Try different sample sizes
```

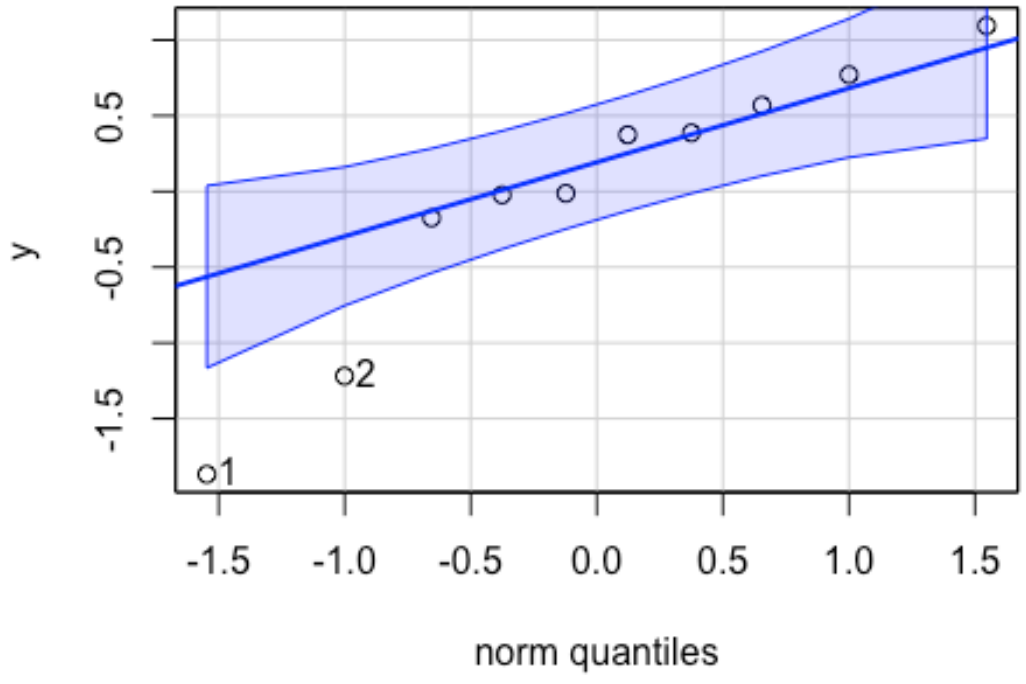
```r
y <- rnorm(n)
# y <- rexp(n)  # could try other distributions

# Order / sort the values
y <- sort(y)

# P(Z < zi) = i / (n + 1)
# x represents probabilities (0, 1)
# BUT zero and one are not included
x <- 1:length(y) / (length(y) + 1)

# Convert the x values (probabilities) to x axis locations
x <- qnorm(x)

# Our crude version of a QQ plot
plot(y ~ x)

# intercept of zero, slope of one
# crude, probability not the best fit
abline(coef = 0:1)
```




```
# built in functions to do this
# compare with above
qqnorm(y) # plot the QQ-plot
qqline(y) # add the line to the plot showing Normality
```



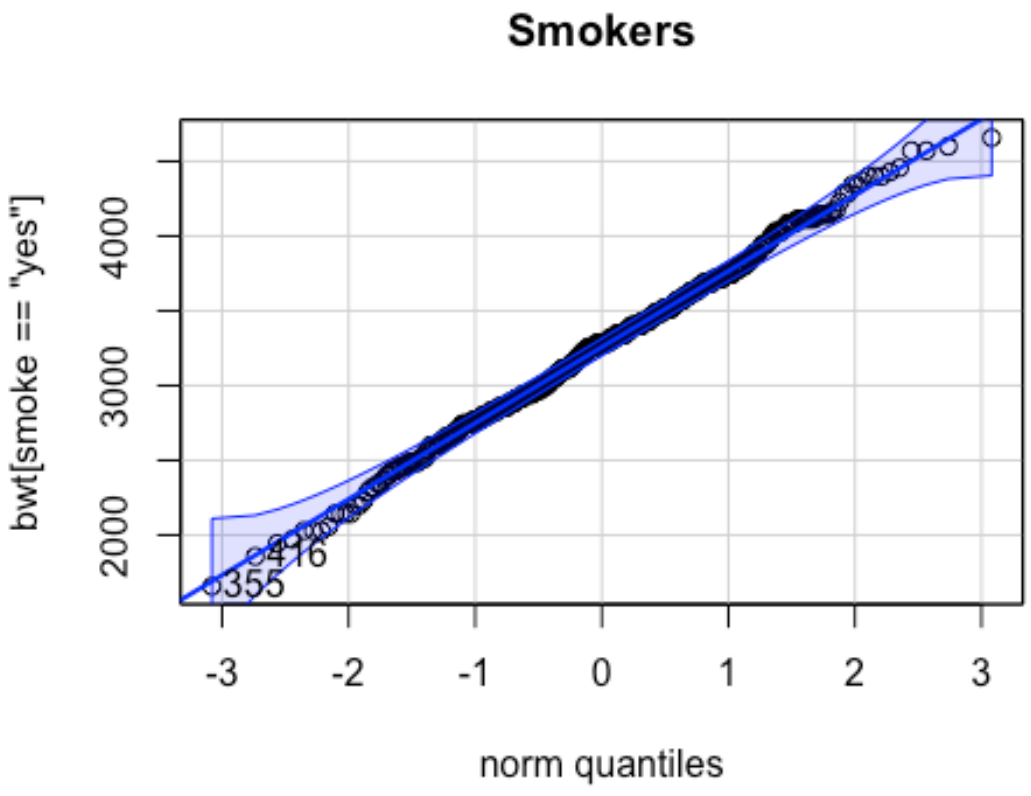

```
# better version
# An alternative is to use the car add-on library:
library(car)

## Loading required package: carData

qqPlot(y) # plot a QQ-plot with a line
```




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

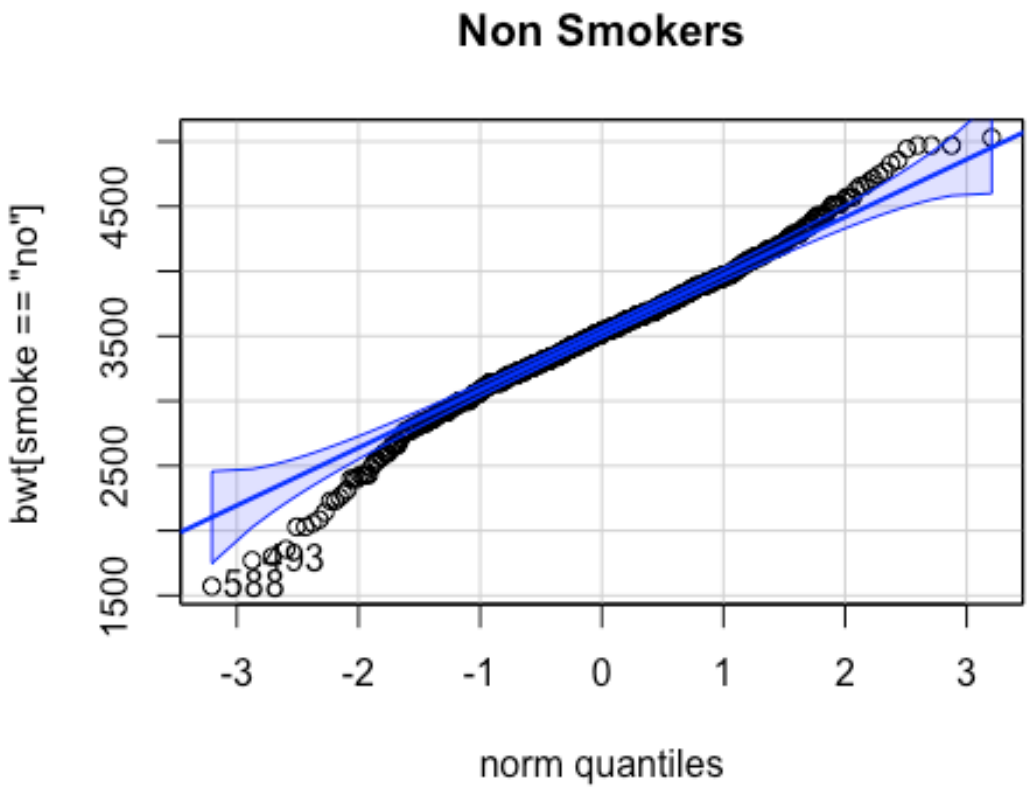
Birth weight QQ - Normal plot

```
birthwt = read.csv("../datasets/birthwt.csv")
with(birthwt, qqPlot(bwt[smoke=="yes"], main="Smokers"))
```




```
## [1] 355 416
```

```
with(birthwt, qqPlot(bwt[smoke=="no"], main="Non Smokers"))
```




```
## [1] 588 493
```

T-test Birth weight

```
df <- read.csv('../datasets/birthwt.csv')

aggregate(df$bwt, list(df$smoke), length)

##   Group.1   x
## 1      no 742
## 2     yes 484

aggregate(df$bwt, list(df$smoke), mean)
```

```
##   Group.1        x
## 1      no 3515.639
## 2     yes 3260.285

aggregate(df$bwt, list(df$smoke), sd)

##   Group.1        x
## 1      no 497.0966
## 2     yes 517.1097

# or
aggregate(bwt ~ smoke, df, sd)

##   smoke      bwt
## 1    no 497.0966
## 2   yes 517.1097

# Assuming equal variances in the populations
t.test(df$bwt ~ df$smoke, var.equal = TRUE,
       alternative = 't')  # H1: mu1 <> mu2

##
##  Two Sample t-test
##
## data:  df$bwt by df$smoke
## t = 8.6527, df = 1224, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group no
and group yes is not equal to 0
## 95 percent confidence interval:
##  197.4554 313.2520
## sample estimates:
##  mean in group no mean in group yes
##          3515.639          3260.285

t.test(df$bwt ~ df$smoke, var.equal = TRUE,
       alternative = 'l')  # H1: mu1 < mu2

##
##  Two Sample t-test
##
## data:  df$bwt by df$smoke
## t = 8.6527, df = 1224, p-value = 1
## alternative hypothesis: true difference in means between group no
```

```
and group yes is less than 0
## 95 percent confidence interval:
##      -Inf 303.9322
## sample estimates:
##  mean in group no mean in group yes
##          3515.639          3260.285

t.test(df$bwt ~ df$smoke, var.equal = TRUE,
       alternative = 'g')  # H1: mu1 > mu2

##
##  Two Sample t-test
##
## data:  df$bwt by df$smoke
## t = 8.6527, df = 1224, p-value < 2.2e-16
## alternative hypothesis: true difference in means between group no
and group yes is greater than 0
## 95 percent confidence interval:
##  206.7752      Inf
## sample estimates:
##  mean in group no mean in group yes
##          3515.639          3260.285
```