STAT 210
Applied Statistics and Data Analysis
Problem List 2 - Solution
(due on week 3)

Fall 2025

Exercise 1

This is an exercise on the use of `plot` and its arguments. You will use the `penguins` data set in library `palmerpenguins`, that has data on penguins from the Palmer Archipelago in the Antartica.

(a) Use `str` to explore the structure of the data set and look at the help. Give the class a description of the data set.

(b) **Boxplots:** Divide the plotting window into four sectors and draw boxplots for the four numerical variables (`bill_length_mm`, `bill_depth_mm`, `flipper_length_mm`, and `body_mass_g`) according to `species`. Use different colors for the species. The web page https://sites.stat.columbia.edu/tzheng/files/Rcolor.pdf shows all the colors that have a name. Choose one of these for each species. Comment on what you observe.

(c) **Histograms:** For this part of the question you will use the function `truehist` in the `MASS` package. Look at the help and get familiar with it. Divide the plotting window into a 3×2 matrix; The first row corresponds to the `Adelie` species, the second to `Chinstrap`, and the third to `Gentoo`. On the first column, plot histograms of relative frequencies for bill length, and on the second histograms of relative frequencies for flipper length. In all cases set `nbins = 10`. Use the variable name as a label for the x-axis and add a title to each plot indicating the species. For each species, use the same colors that you chose for (b). Since you want to compare the distribution of the variables according to species, the scales on the x-axis should be the same for plots on the same column. Comment on what you observe.

(d) Using the function `plot`, create a matrix of plots for the four numerical variables in `penguins` –those you used in (b). Use a solid square as the plotting symbol and color by the values of `species`. Comment on what you observe. Try to use the same colors as before for the species.

Solution:

The data for the `palmerpenguins` package contains size measurements for three penguin species observed on three islands in the Palmer Archipelago, Antarctica. These data were collected from 2007 - 2009 by Dr. Kristen Gorman with the Palmer Station Long Term Ecological Research Program, part of the US Long Term Ecological Research Network. Figure 1 gives a graphical representation of specimens of each of the three species included in the data set.

(a) Load the library `palmerpenguins` and the data set

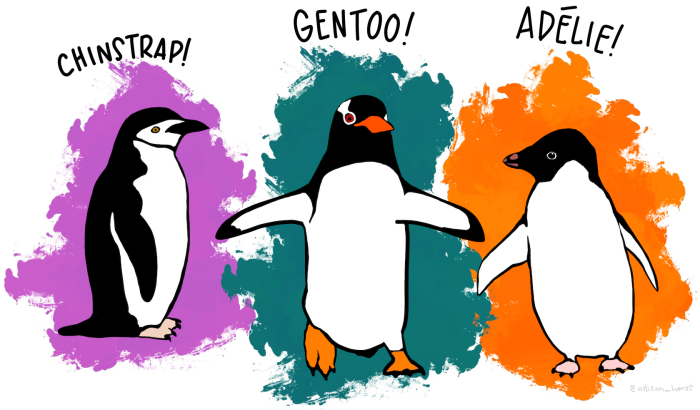


Figure 1: The Palmer penguins.

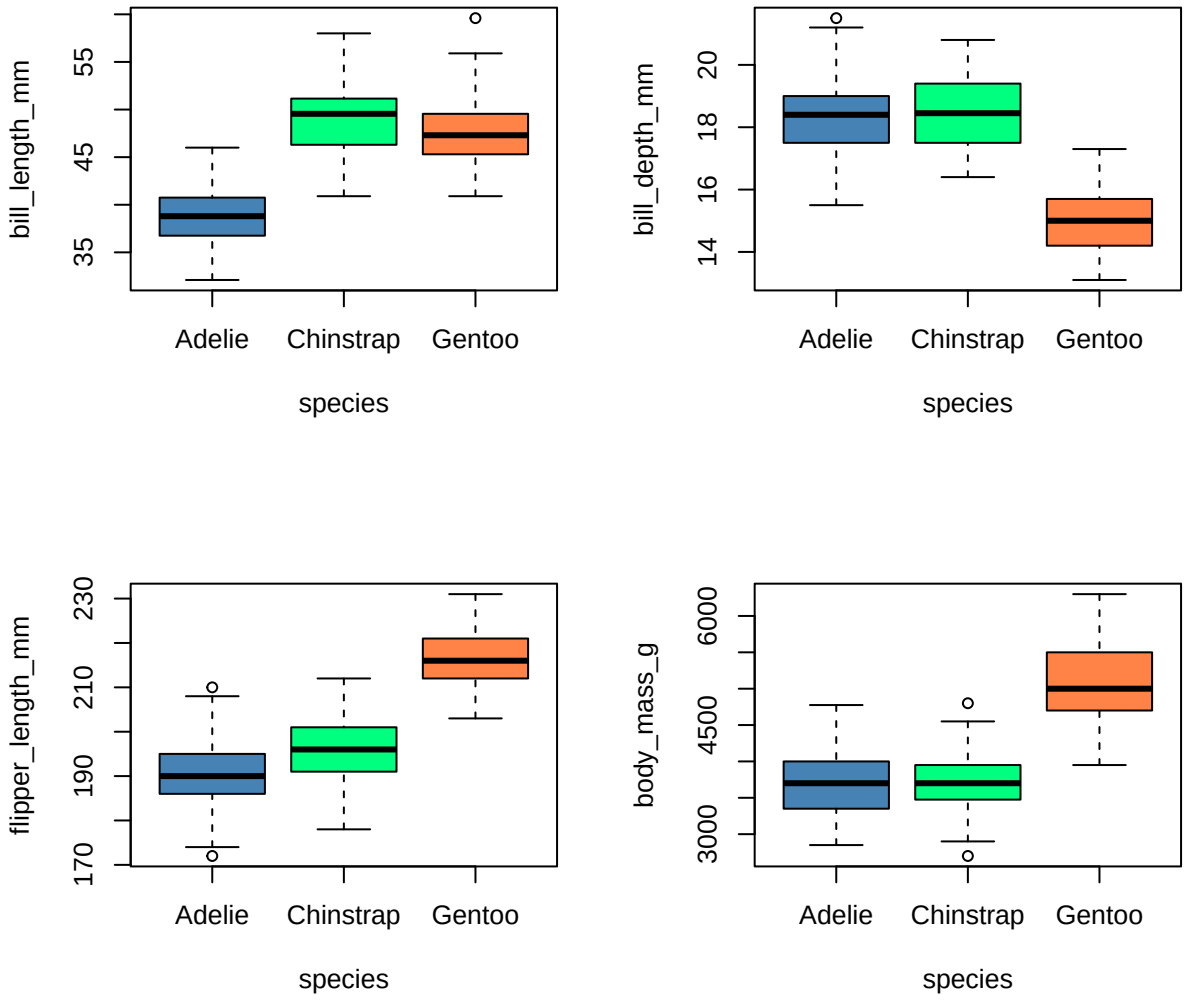
```
library(palmerpenguins)
data("penguins")
str(penguins)
```

```
## tibble [344 x 8] (S3: tbl_df/tbl/data.frame)
##  $ species          : Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ island           : Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ bill_length_mm   : num [1:344] 39.1 39.5 40.3 NA 36.7 39.3 38.9 39.2 34.1 42 ...
##  $ bill_depth_mm    : num [1:344] 18.7 17.4 18 NA 19.3 20.6 17.8 19.6 18.1 20.2 ...
##  $ flipper_length_mm: int [1:344] 181 186 195 NA 193 190 181 195 193 190 ...
##  $ body_mass_g      : int [1:344] 3750 3800 3250 NA 3450 3650 3625 4675 3475 4250 ...
##  $ sex              : Factor w/ 2 levels "female","male": 2 1 1 NA 1 2 1 2 NA NA ...
##  $ year             : int [1:344] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

We have a `tibble`, which is a modern data frame, with eight variables, five of which are numerical: `bill_length_mm`, `bill_depth_mm`, `flipper_length_mm`, `body_mass_g`, and `year`, and three categorical variables (factors), `species`, `island`, and `sex`. There are three levels for `species`, `Adelie`, `Chinstrap`, and `Gentoo`. There are a total of 344 observations with some missing data. If you want to learn more about `tibbles` and their relation to data frames look at https://science.nu/lektion/managing-tibbles/#:~:text=Tibbles%20are%20data%20frames%2C%20but,pronounced%20as%20\T1\textquotedbllefttibble%20diff\T1\textquotedblright.

(b) **Boxplots:** I chose three contrasting colors for the plots: `steelblue`, `springgreen`, and `sienna1`. In the first command below, I created a vector named `cols` with the names of the colors to facilitate their use.

```
cols <- c('steelblue','springgreen', 'sienna1')
par(mfrow = c(2,2))
with(penguins, plot(bill_length_mm ~ species, col = cols))
with(penguins, plot(bill_depth_mm ~ species, col = cols))
with(penguins, plot(flipper_length_mm ~ species, col = cols))
with(penguins, plot(body_mass_g ~ species, col = cols))
```




```
par(mfrow = c(1,1))
```

The boxplot for bill length shows that the `Adelie` species has shorter bill length, while the range of values for the other two species is similar. On the other hand, for bill depth, it is the `Gentoo` species that has smaller values, while the other two species have similar range of values.

For the other two variables, flipper length and body mass, it is the `Gentoo` species that has larger values while the other two have similar ranges.

(c) **Histograms:** We start by looking at the range of values of the variables we are considering, to determine the limits for the x-axis in the plots. Since there are missing data (`NA`'s) we need to set the option `na.rm` to `TRUE`. This removes the missing data from the calculation.

```
range(penguins$bill_length_mm, na.rm = T)
```

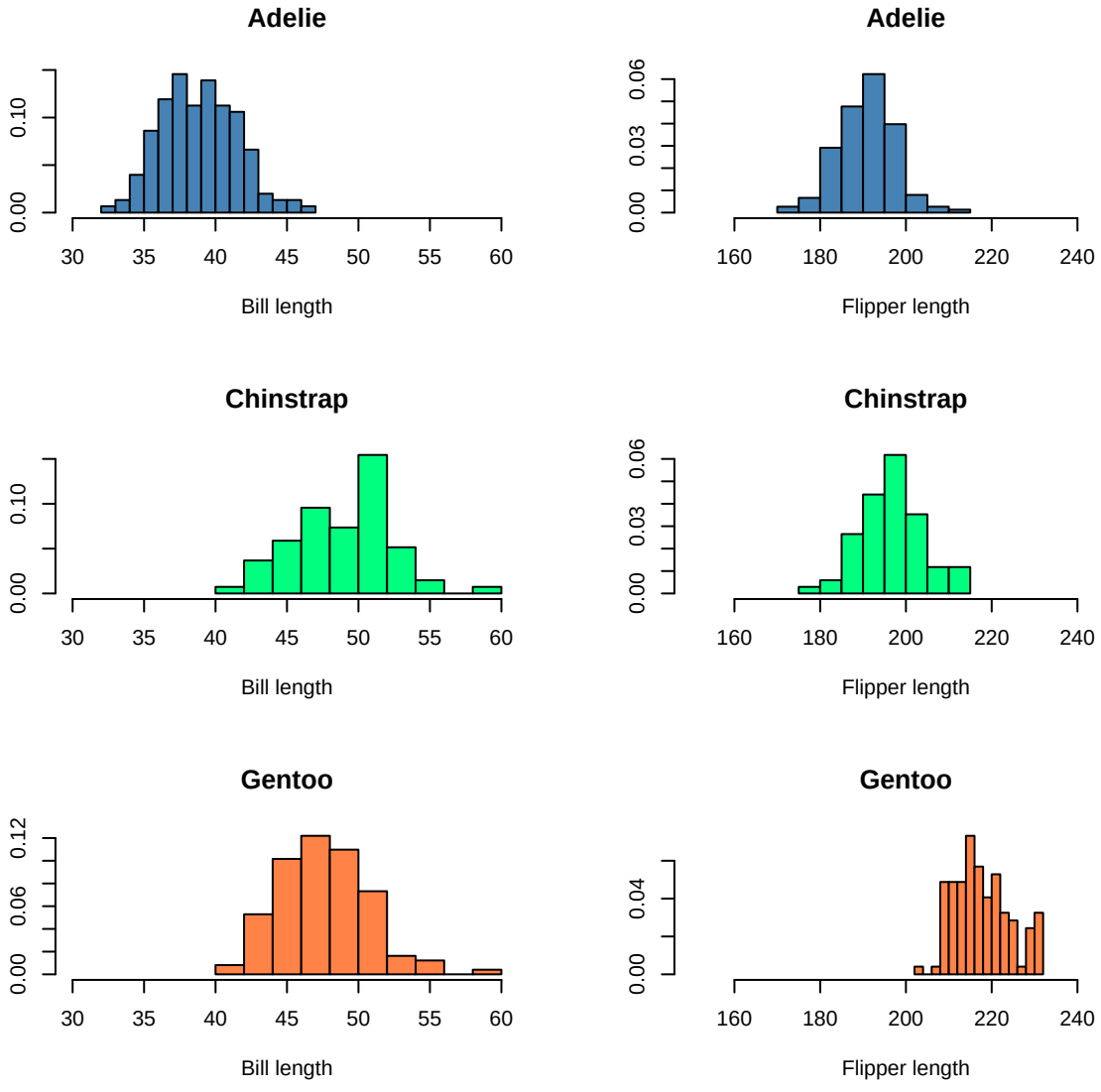
```
## [1] 32.1 59.6
```

```
range(penguins$flipper_length_mm, na.rm = T)
```

```
## [1] 172 231
```

We set the limits for `bill_length` at 30 and 60, while for `flipper_length` the limits are 150 and 250. Since we are going to be working with this data set for the whole exercise, we `attach` it, to simplify the notation for the variables. At the end of the solution, the data set is detached with the command `detach`.

```r
attach(penguins)
library(MASS)
par(mfrow = c(3,2))
truehist(bill_length_mm[species=='Adelie'], main = 'Adelie', xlab = 'Bill length',
         xlim = c(30,60), col = cols[1], nbins = 10, prob = TRUE)
truehist(flipper_length_mm[species=='Adelie'], main = 'Adelie', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[1], nbins = 10, prob = TRUE)
truehist(bill_length_mm[species=='Chinstrap'], main = 'Chinstrap', xlab = 'Bill length',
         xlim = c(30,60), col = cols[2], nbins = 10, prob = TRUE)
truehist(flipper_length_mm[species=='Chinstrap'], main = 'Chinstrap', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[2], nbins = 10, prob = TRUE)
truehist(bill_length_mm[species=='Gentoo'], main = 'Gentoo', xlab = 'Bill length',
         xlim = c(30,60), col = cols[3], nbins = 10, prob = TRUE)
truehist(flipper_length_mm[species=='Gentoo'], main = 'Gentoo', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[3], nbins = 10, prob = TRUE)
```



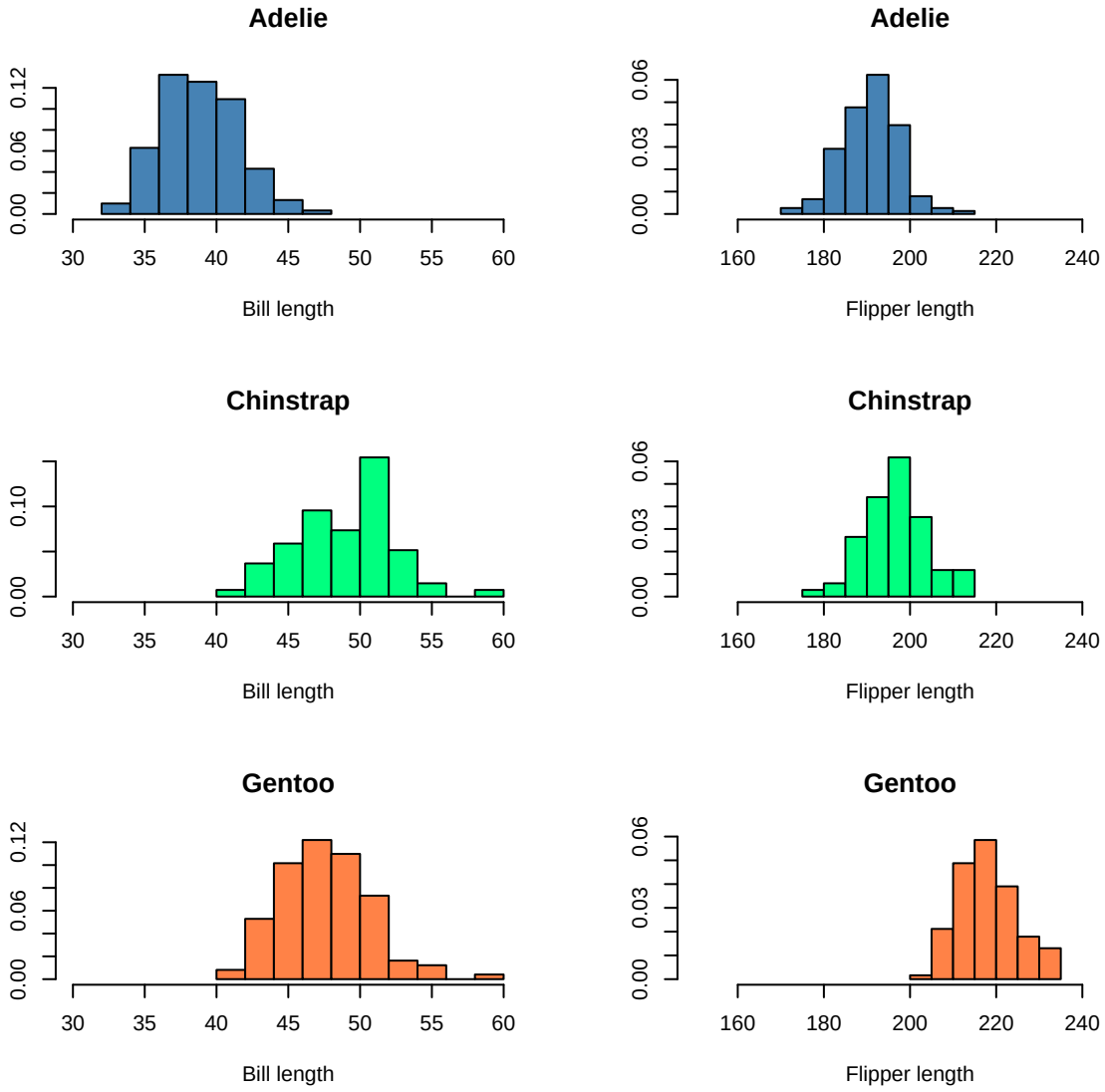

```r
par(mfrow = c(1,1))
```

On the first column we have bill length, and we see that the values are smaller for the `Adelie` species, while they are similar for `Chinstrap` and `Gentoo`, which agrees with what we observed in the boxplots, but with the histograms we get a better idea of the distributions. For flipper length, in the second column, the values

for `Adelie` and `Chinstrap` are simlar, while those for `Gentoo` are larger.

Observe also that, even though we set the number of breaks to be 10, the software has decided on the number of bins separately for each plot. One way to guarantee that the same bin width is used in all plots is to use the option `h =` instead of `nbins =`. This option controls the bin width:

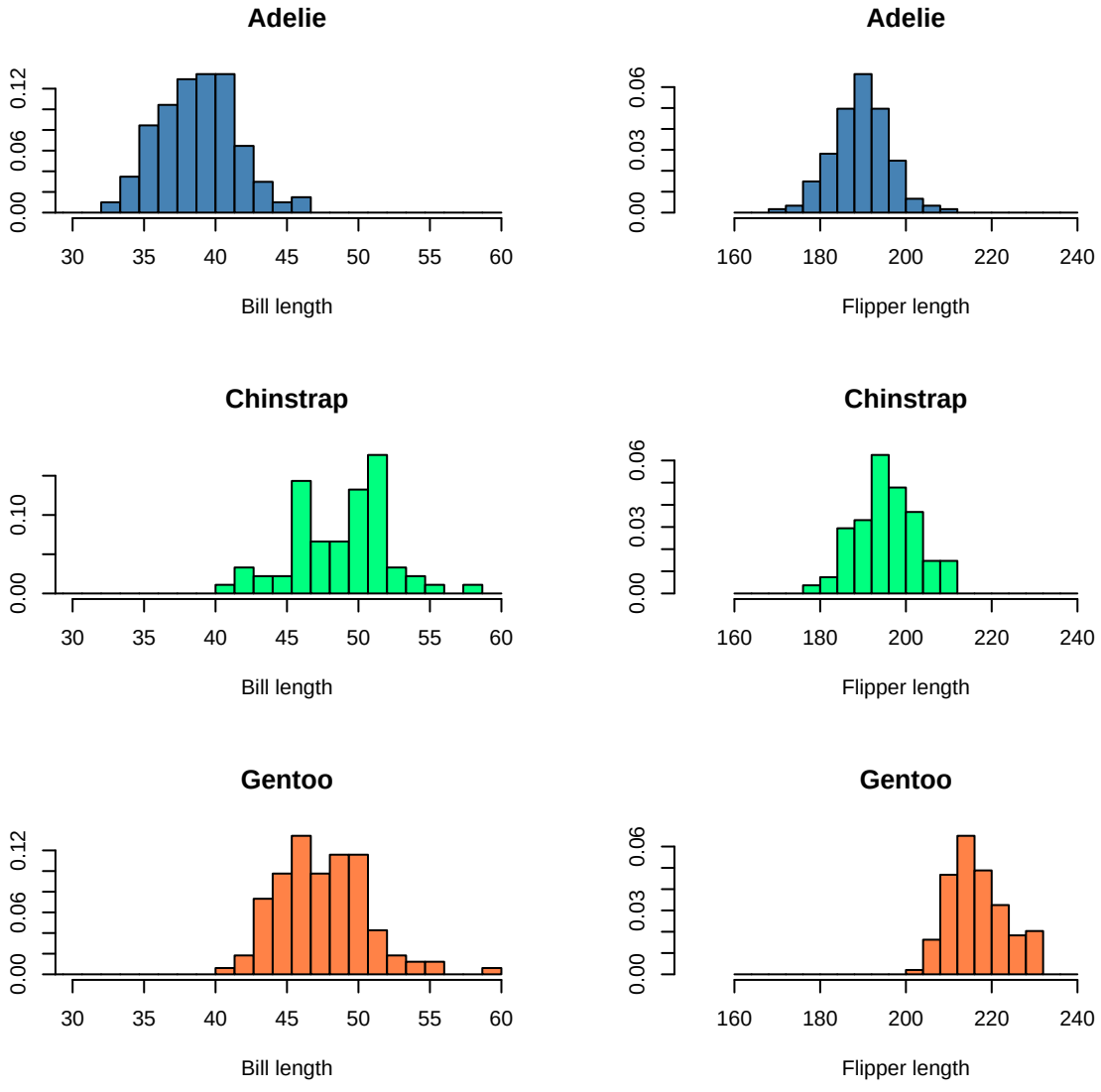
```
par(mfrow = c(3,2))
truehist(bill_length_mm[species=='Adelie'],main = 'Adelie', xlab = 'Bill length',
         xlim = c(30,60), col = cols[1], h = 2, prob = TRUE)
truehist(flipper_length_mm[species=='Adelie'],main = 'Adelie', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[1], h = 5, prob = TRUE)
truehist(bill_length_mm[species=='Chinstrap'],main = 'Chinstrap', xlab = 'Bill length',
         xlim = c(30,60), col = cols[2], h = 2, prob = TRUE)
truehist(flipper_length_mm[species=='Chinstrap'],main = 'Chinstrap', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[2], h = 5, prob = TRUE)
truehist(bill_length_mm[species=='Gentoo'],main = 'Gentoo', xlab = 'Bill length',
         xlim = c(30,60), col = cols[3], h = 2, prob = TRUE)
truehist(flipper_length_mm[species=='Gentoo'],main = 'Gentoo', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[3], h = 5, prob = TRUE)
```




```
par(mfrow = c(1,1))
```

Another possibility that gives even more control on the bins is the option **breaks**. The argument here is a vector with the endpoints for the bins. In the code below, we chose 30 bins for the first column and 20 for the second

```
par(mfrow = c(3,2))
truehist(bill_length_mm[species=='Adelie'],main = 'Adelie', xlab = 'Bill length',
         xlim = c(30,60), col = cols[1], breaks = seq(20, 60, length.out = 31), prob = TRUE)
truehist(flipper_length_mm[species=='Adelie'],main = 'Adelie', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[1], breaks = seq(160, 240, length.out = 21), prob = TRUE)
truehist(bill_length_mm[species=='Chinstrap'],main = 'Chinstrap', xlab = 'Bill length',
         xlim = c(30,60), col = cols[2], breaks = seq(20, 60, length.out = 31) , prob = TRUE)
truehist(flipper_length_mm[species=='Chinstrap'],main = 'Chinstrap', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[2], breaks = seq(160, 240, length.out = 21), prob = TRUE)
truehist(bill_length_mm[species=='Gentoo'],main = 'Gentoo', xlab = 'Bill length',
         xlim = c(30,60), col = cols[3], breaks = seq(20, 60, length.out = 31), prob = TRUE)
truehist(flipper_length_mm[species=='Gentoo'],main = 'Gentoo', xlab = 'Flipper length',
         xlim = c(150,250), col = cols[3], breaks = seq(160, 240, length.out = 21), prob = TRUE)
```




```
par(mfrow = c(1,1))
```

(d) We select the four columns that correspond to the variables of interest for the plot:

```r
plot(penguins[3:6],pch = 15, col = cols[as.numeric(species)])
```

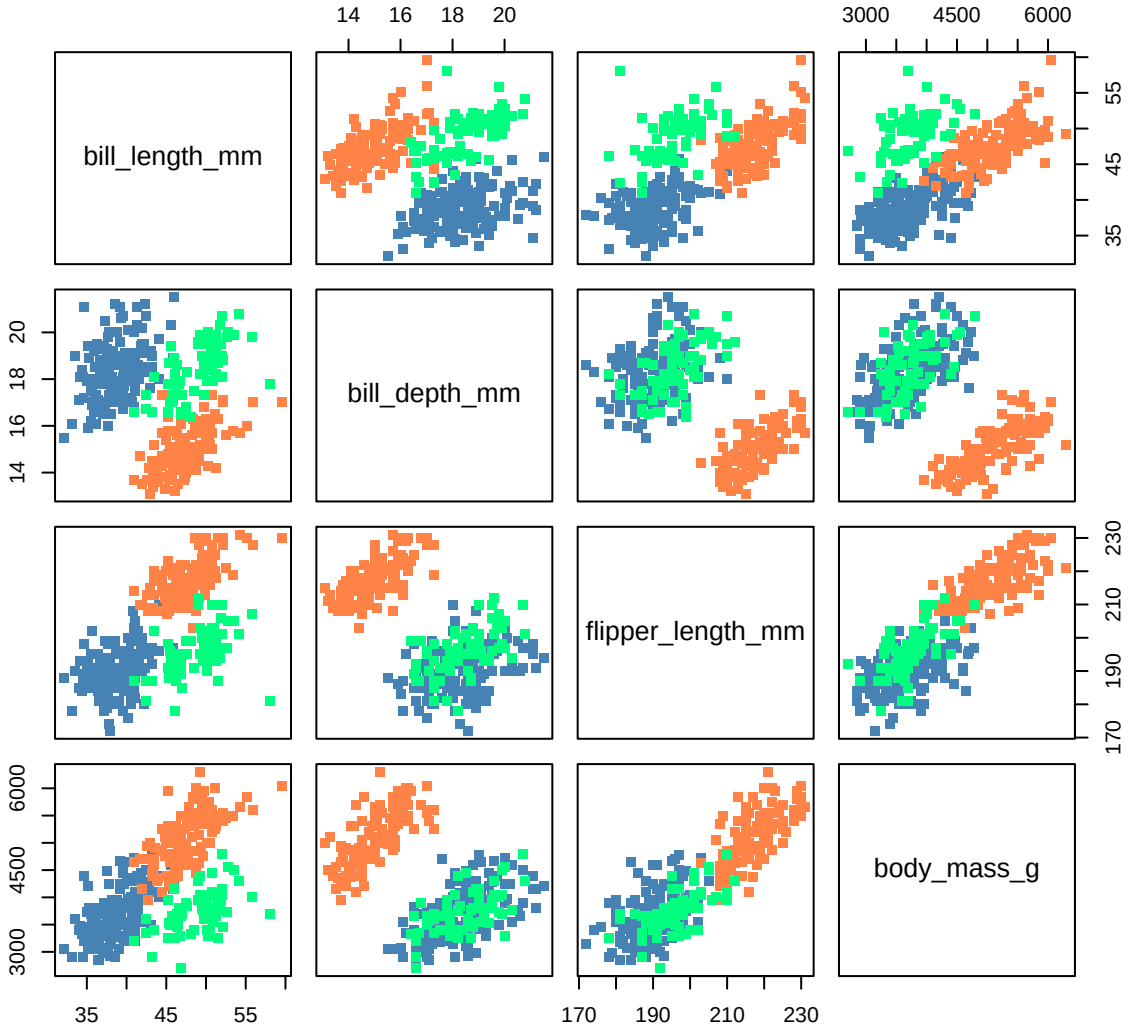


Figure 2: Scatterplot matrix for the Palmer penguin data. The species are coded by colors: Gentoo is orange, Chinstrap is green, and Adelie is blue

The first observation is that in all the plots we observe that there seems to be a linear relation between the corresponding variables, but, in general, this relation depends on the species. For instance, in the bill length vs. bill depth plot we see that there is an increasing relation between these two variables, bigger bill depth corresponds to bigger bill length, but this relation is different for each species.

If we look at the other two plots on the first row, we see that the cloud of points corresponding to the Chinstrap species is above the other two, but all the species seem to have similar slopes. Also, the clouds for Adelie and Gentoo seem to be aligned, although the range of values differ.

For the other three plots above the diagonal, the points for Adelie and Chinstrap overlap, indicating that we should expect a similar relation between the variables for these two species. In the two plots that have bill depth on the y-axis, the values for the Gentoo species are smaller and the cloud of points does not overlap the other two. This suggests that these variables could be used for classification purposes, as they seem to separate Gentoo penguins from the other two species.

Finally, in the plot of flipper length against body mass, the three couds seem to be aligned, although the values for the Gentoo species are bigger than for the other two.

We end the exercise detaching the data set:

```
detach(penguins)
```

Exercise 2

Histograms

For this exercise we are going to use simulated data from a mixture of three normal distribution. In the mixture, 40% of the points come from a normal distribution with mean -5 and standard deviation (sd) 1.8, 20% come from a normal with mean 0 and sd = 1.2, and the remaining 40% come from a normal with mean 4 and sd = 1.4:

$$0.4N(-5, 1.8) + 0.2N(0, 1.2) + 0.4N(4, 1.4)$$

The following code plots the density for this mixture

```
points_x <- seq(-10,10,length.out = 1001)
points_dens <- 0.4*(dnorm(points_x, mean=-5, sd = 1.8)) +
               0.2*(dnorm(points_x,mean=0, sd = 1.2)) +
               0.4*(dnorm(points_x,mean=4, sd = 1.4))
plot(points_x, points_dens, type = 'l', col = 'navyblue', lwd = 2,
     ylim = c(0,0.15), xlab = 'x', ylab = 'density', main = 'Mixture density')
abline(h=0, col = 'red'); abline(v=0, col = 'red')
```

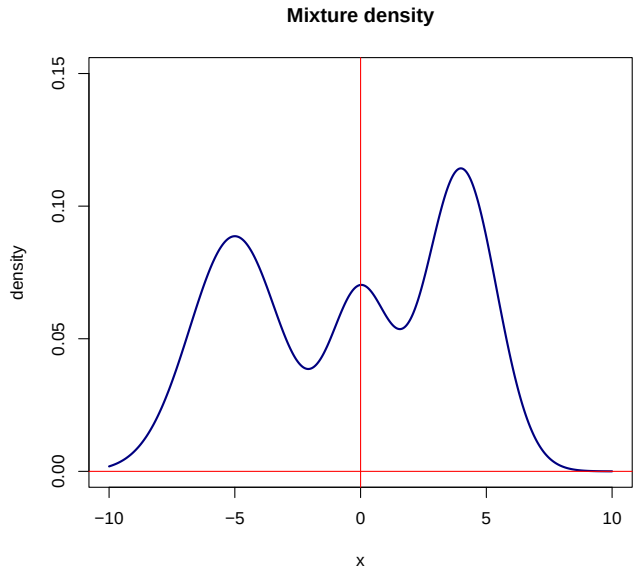


Figure 3: Density for mixture of normal distributions

The following commands draw a sample of size 1000 from this mixture, and print the range of values for the simulated data. The sample is stored in the vector `mix_sample`

```
n <- 1000
pops <- sample(x=c('1','2','3'), n, replace = T, prob = c(0.4,0.2,0.4))
mix_sample <- numeric(n)
mix_sample[pops == "1"] <- rnorm(sum(pops == "1"), mean = -5, sd = 1.8)
mix_sample[pops == "2"] <- rnorm(sum(pops == "2"), mean = 0, sd = 1.2)
mix_sample[pops == "3"] <- rnorm(sum(pops == "3"), mean = 4, sd = 1.4)
(rng <- range(mix_sample))
```

```
## [1] -9.942119  8.413861
```

We will use this sample to draw histograms with the function `truehist` in the `MASS` package. Look up the help for `truehist`. It is also a good idea to explore the use of the function `hist` on the base package by repeating this exercise using `hist`.

(a) Divide the plotting window into 4 using the function `par` with argument `mfrow`. Select four disjoint subsets of data of length 50 and draw histograms for them. Set the bin width to 1 in all plots. Make sure that the scales are the same for all plots. Are these plots similar to the density in Figure 1?

(b) Divide again the plotting window into 4. Draw successive histograms of relative frequency for the first 50, 100, 500, and 1000 points in `mix_sample`. Set the bin width to 1 in all plots. Make sure that the scales are the same for all plots. Are these plots similar to the density plotted above?

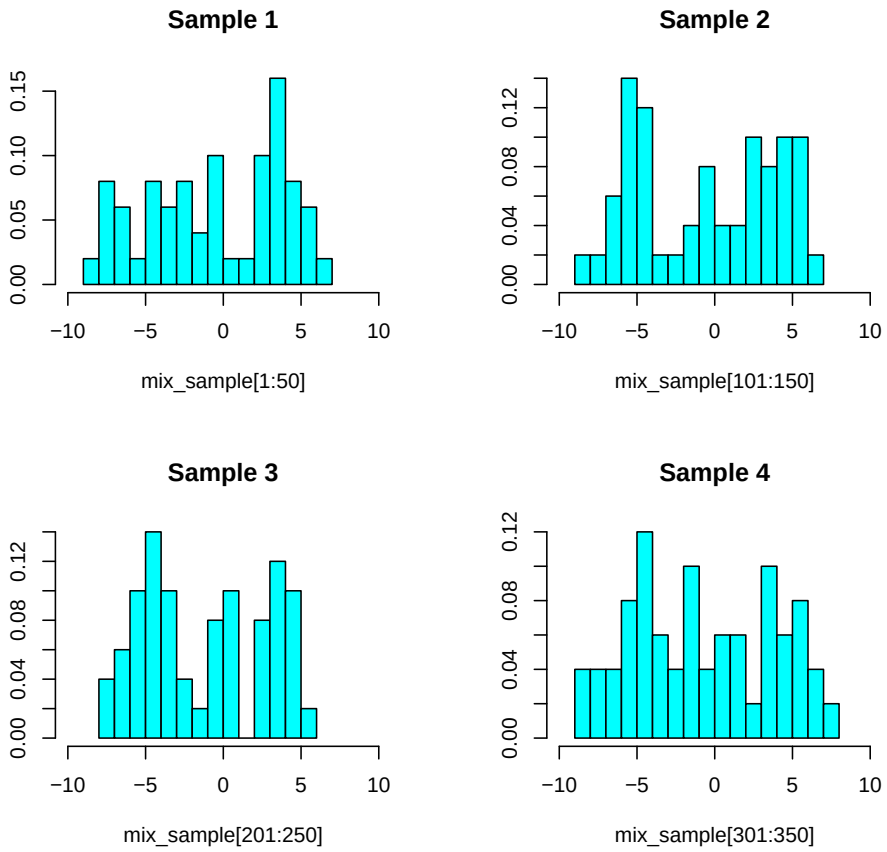
(c) Using again the function `par` with argument `mfrow`, set the graphical window to one row and three columns. Draw a histogram of relative frequency using all the points in `mix_sample`. Set the number of bins (`nbins`) to 10, 20, and 50, respectively. Comment.

(d) Now set the graphic window to just one graph and repeat the histograms using `FD` as value for `nbins`. Using the function `lines` with argument `density(mix_sample)`, add an estimate of the density for this sample. Color the line in blue. Add also a graph of the theoretical density in red (look back to the previous page to see how this density was plotted before and make the necessary changes). Add a legend for the two lines that you plotted. Comment on what you observe.

Solution:

(a) Using the range of values printed above, we set the limits for the plots at -10 and 10

```
library(MASS)
par(mfrow = c(2,2))
truehist(mix_sample[1:50],    h=1, xlim = c(-10,10), ylab = '', main = 'Sample 1')
truehist(mix_sample[101:150], h=1, xlim = c(-10,10), ylab = '', main = 'Sample 2')
truehist(mix_sample[201:250], h=1, xlim = c(-10,10), ylab = '', main = 'Sample 3')
truehist(mix_sample[301:350], h=1, xlim = c(-10,10), ylab = '', main = 'Sample 4')
```

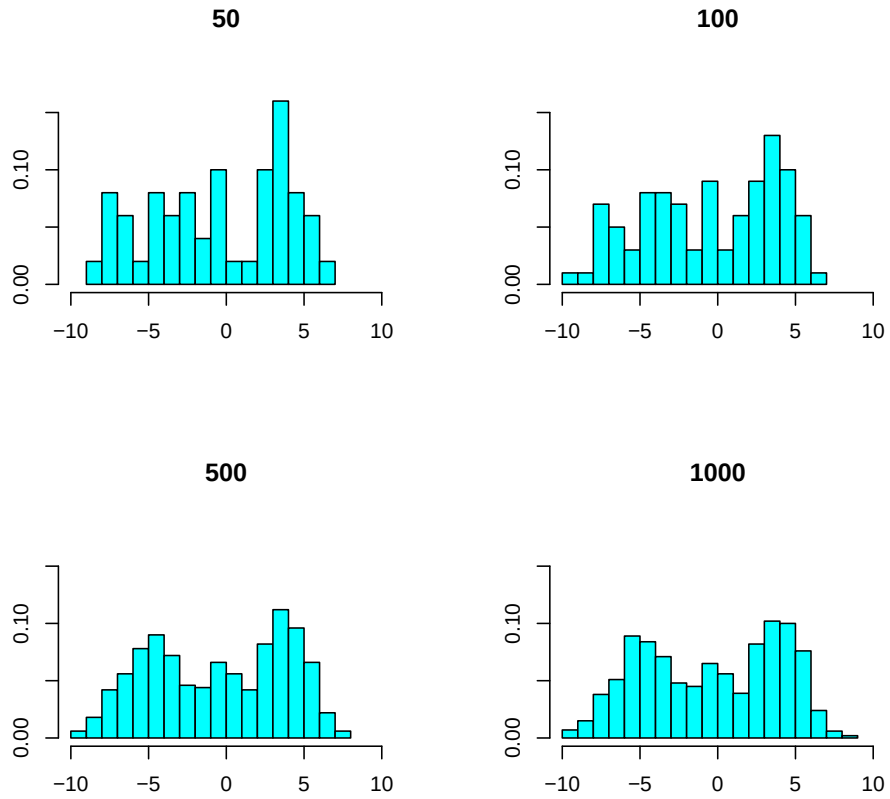



```
par(mfrow = c(1,1))
```

None of the plots resembles the density, the sample size is too small.

(b) Increasing sample size.

```
par(mfrow = c(2,2))
truehist(mix_sample[1:50],h=1,   xlim = c(-10,10), ylim = c(0,0.18),
         xlab = '', main = '50')
truehist(mix_sample[1:100],h=1,  xlim = c(-10,10), ylim = c(0,0.18),
         xlab = '', main = '100')
truehist(mix_sample[1:500],h=1,  xlim = c(-10,10), ylim = c(0,0.18),
         xlab = '', main = '500')
truehist(mix_sample[1:1000],h=1, xlim = c(-10,10), ylim = c(0,0.18),
         xlab = '', main = '1000')
```

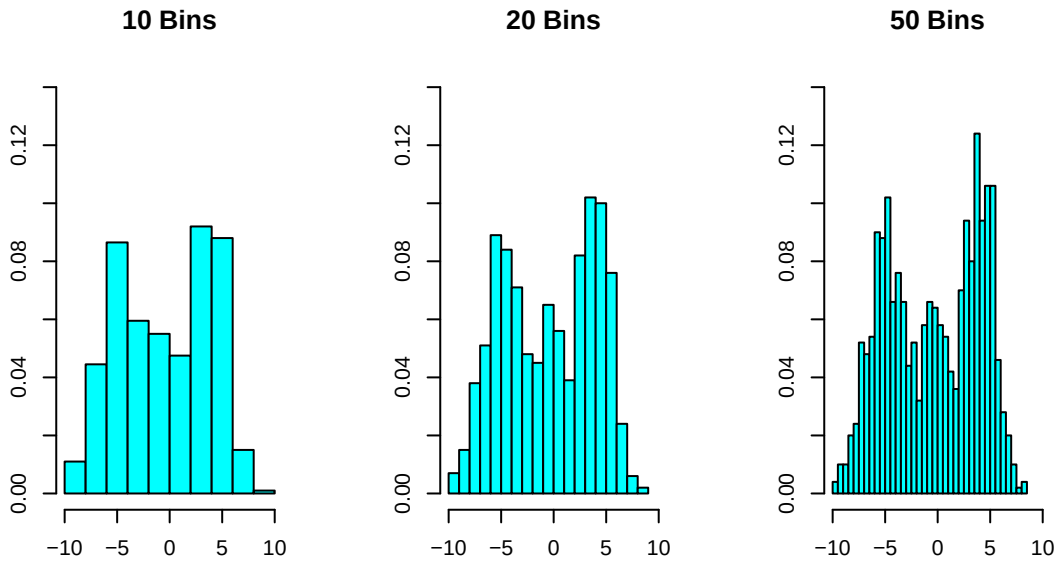



```
par(mfrow = c(1,1))
```

We see that as the sample size increases, the histograms approach the shape of the density. The histogram for the whole sample has a good resemblance to the density.

(c) Different number of bins:

```
par(mfrow = c(1,3))
truehist(mix_sample[1:1000], nbins = 10,  xlim = c(-10,10), ylim = c(0,0.14),
         main = '10 Bins', xlab = '')
truehist(mix_sample[1:1000], nbins = 20,  xlim = c(-10,10), ylim = c(0,0.14),
         main = '20 Bins', xlab = '')
truehist(mix_sample[1:1000], nbins = 50,  xlim = c(-10,10), ylim = c(0,0.14),
         main = '50 Bins', xlab = '')
```

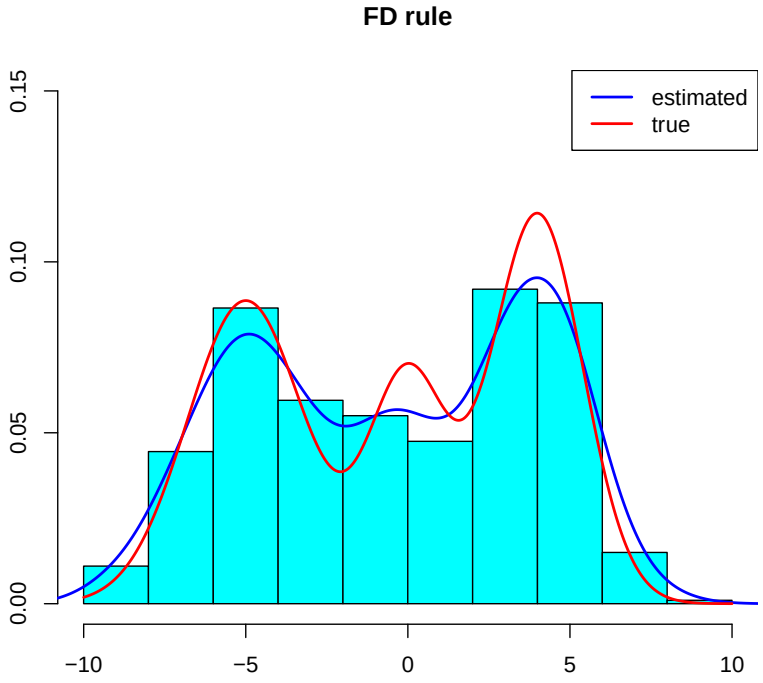



```
par(mfrow = c(1,1))
```

The first plot has too few bins, and misses the central mode of the distribution. The other two plots show a better resemblance with the density.

(d) Histogram with estimated and true densities.

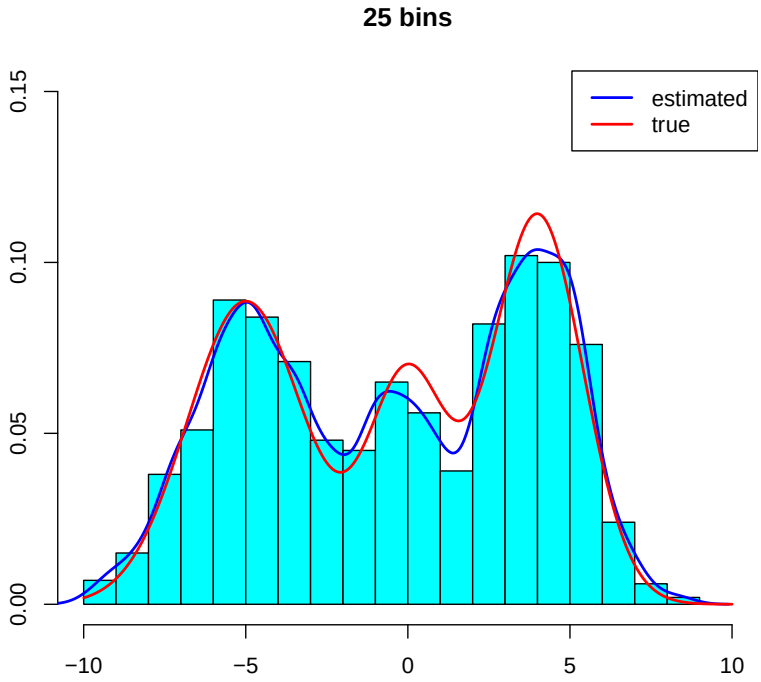
```
truehist(mix_sample,nbins = 'FD', xlim = c(-10,10), ylim = c(0,0.15),
         xlab = '', main = 'FD rule')
lines(density(mix_sample), col = 'blue', lwd = 2)
lines(points_x, points_dens, col = 'red', lwd = 2)
legend('topright',c('estimated','true'), col = c('blue','red'), lwd = 2)
```



We see that both the histogram and the estimated density miss the central bump. It is possible to improve the histogram by increasing the number of bins. It is also possible to improve the estimated density by reducing the 'bandwidth' for the kernel estimation, something we will study later on in the course. We do

this by setting the `adjust` parameter to 0.5 (the default value is 1).

```
truehist(mix_sample,nbins = 25, xlim = c(-10,10), ylim = c(0,0.15),
         xlab = '', main = '25 bins')
lines(density(mix_sample, adjust = 0.5), col = 'blue', lwd = 2)
lines(points_x, points_dens, col = 'red', lwd = 2)
legend('topright',c('estimated','true'), col = c('blue','red'), lwd = 2)
```



The density estimate improves, although the estimated density is rougher with a smaller bandwidth.

Exercise 3

A group of teachers wanted to understand better what the optimal duration of study is for students to achieve satisfactory results. They decided to survey the students in the school about the approximate number of hours they were studying, and then compare to the student's test scores. The information they collected is in the file `pl2ex3_25.csv`. The variables in this file are `hours`, `grade`, and `subject`.

(a) Load the data and store it in an object named `df3`. Use the function `str` to explore this data set. Transform `subject` into a factor.

(b) Some teachers thought that more data means better results, and they integrated their cross-course data for the analysis. Do a scatterplot of `grade` as a function of `hours` for all the data. Use adequate names for the variables in the plot and a filled circle as plotting symbol. Comment on what you observe. Is this what you expected? Calculate the correlation between these two variables using the command `cor` and comment.

(c) The teachers were confounded with the graph and the conclusion that they drew from it, so they consulted the statistics teacher, who suggested that they analyze each course's data individually. Plot the data for `Physical Education` using the same instructions as in (b). Comment on what you obtain. Calculate the correlation between `grade` and `hours` for this subject and comment.

(d) Repeat the plot in (b) but now color the data according to `subject`. Add a legend. Comment.

(e) Use the function `split` to create a list with five components, one for each subject. In each component, you want to have the variables `hours` and `grade` (but not `subject`). Call this list `lst3`. Use the command

`lapply(lst3, cor)` to obtain the correlation between `grades` and `hours` per subject. Compare with the correlation you obtained in (b) and comment.

This problem is an example of what is known as Simpson's paradox. If you want to know more, look at the Wikipedia page for this topic.

Solution:

(a) Read the data

```
df3 <- read.csv('pl2ex3_25.csv')
str(df3)
```

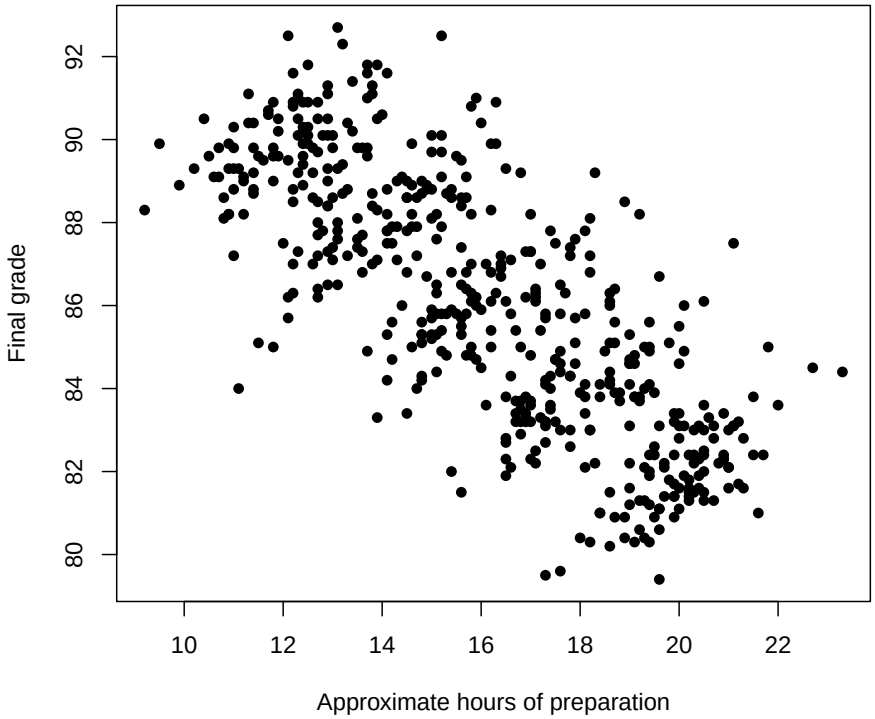
```
## 'data.frame':    500 obs. of  3 variables:
##  $ hours  : num  18.9 19.7 18.6 21.1 19.7 20 19.9 19.9 20.2 18.9 ...
##  $ grade  : num  80.4 82.2 81.5 83.1 81.4 81.6 81.7 81.4 81.8 80.9 ...
##  $ subject: chr  "Algebra" "Algebra" "Algebra" "Algebra" ...
```

```
df3$subject <- factor(df3$subject)
str(df3)
```

```
## 'data.frame':    500 obs. of  3 variables:
##  $ hours  : num  18.9 19.7 18.6 21.1 19.7 20 19.9 19.9 20.2 18.9 ...
##  $ grade  : num  80.4 82.2 81.5 83.1 81.4 81.6 81.7 81.4 81.8 80.9 ...
##  $ subject: Factor w/ 5 levels "Algebra","Art",..: 1 1 1 1 1 1 1 1 1 1 ...
```

(b) Plot the graph for all the data.

```
with(df3,plot(grade ~ hours, pch = 16,
              xlab = 'Approximate hours of preparation',
              ylab = 'Final grade'))
```



This is certainly not what one would expect. The final grade seems to get worse the more hours of preparation a student has! No wonder the teachers were confounded by this because, if this conclusion is right, students

should work as little as possible! Let's calculate the correlation

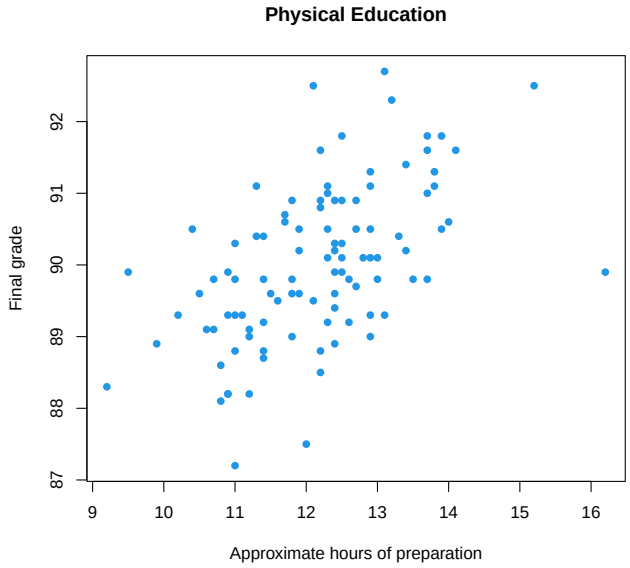
```
cor(df3$hours, df3$grade)
```

```
## [1] -0.767551
```

We get a negative correlation, which agrees with the plot.

(c) Plot for `Physical Education`

```
with(df3,plot(grade[subject=='Physical Education'] ~ hours[subject=='Physical Education'],
              pch = 16, main = 'Physical Education', ylab = 'Final grade',
              xlab = 'Approximate hours of preparation', col = 4))
```



This plot seems to indicate the opposite of the previous graph: The final grade increases as the number of study hours increases, which is what one would expect. Let's calculate the correlation in this case

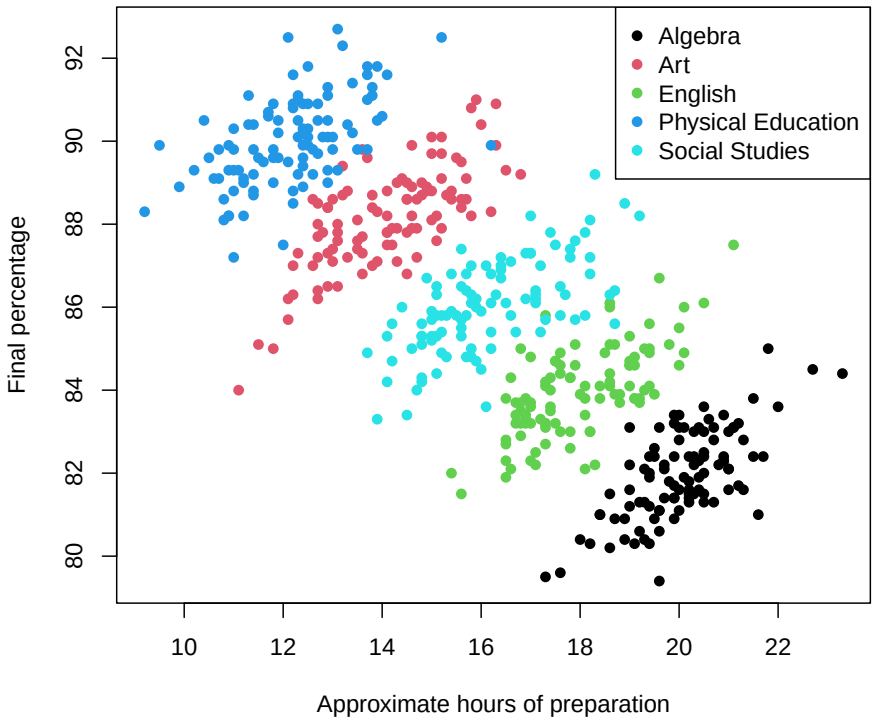
```
cor(df3$hours[df3$subject=='Physical Education'], df3$grade[df3$subject=='Physical Education'])
```

```
## [1] 0.5356516
```

Now we get a positive correlation.

(d) Plot with different colors

```
with(df3, plot(grade ~ hours, pch = 16, col = as.numeric(as.factor(subject)),
               xlab = 'Approximate hours of preparation',
               ylab = 'Final percentage'))
legend('topright', c('Algebra','Art','English','Physical Education','Social Studies'),
       col = 1:5, pch = 16)
```



In this graph we see that even though there is a decreasing relation for the whole data set, when we consider each subject separately, the relation between the final score and the hours of preparation is increasing, as we would expect.

(e) Use `split` on the first two columns of `df3`, and group by `subject`:

```
lst3 <- split(df3[,1:2], df3$subject)
str(lst3)
```

```
## List of 5
##  $ Algebra           :'data.frame':  100 obs. of  2 variables:
##   ..$ hours: num [1:100] 18.9 19.7 18.6 21.1 19.7 20 19.9 19.9 20.2 18.9 ...
##   ..$ grade: num [1:100] 80.4 82.2 81.5 83.1 81.4 81.6 81.7 81.4 81.8 80.9 ...
##  $ Art               :'data.frame':  100 obs. of  2 variables:
##   ..$ hours: num [1:100] 12.7 14.4 12.7 14.1 13.7 12.2 14.6 16.3 13 15.1 ...
##   ..$ grade: num [1:100] 87.2 89.1 88 88.8 89.6 87 89.9 89.9 87.4 88.2 ...
##  $ English           :'data.frame':  100 obs. of  2 variables:
##   ..$ hours: num [1:100] 20 16.7 16.6 17.3 16.8 17.3 19.4 17.8 17.4 17 ...
##   ..$ grade: num [1:100] 84.6 83.4 84.3 85.8 83.5 83.1 84.9 84.3 83.6 83.2 ...
##  $ Physical Education:'data.frame':  100 obs. of  2 variables:
##   ..$ hours: num [1:100] 10.9 11.8 11.9 10.2 12.3 12.4 12.1 11.6 10.7 12.2 ...
##   ..$ grade: num [1:100] 88.2 89.8 90.2 89.3 91.1 90.3 89.5 89.5 89.8 90.8 ...
##  $ Social Studies    :'data.frame':  100 obs. of  2 variables:
##   ..$ hours: num [1:100] 15.1 16.1 18.7 18.3 17.9 15.5 17.7 15.9 17.9 14.2 ...
##   ..$ grade: num [1:100] 85.3 83.6 86.4 89.2 87.6 85.8 86.3 86 85.7 85.6 ...
```

```
lapply(lst3, cor)
```

```
## $Algebra
##           hours     grade
## hours 1.0000000 0.6790915
## grade 0.6790915 1.0000000
##
```

```
## $Art
##           hours     grade
## hours 1.0000000 0.7151211
## grade 0.7151211 1.0000000
##
## $English
##           hours     grade
## hours 1.0000000 0.6663983
## grade 0.6663983 1.0000000
##
## $`Physical Education`
##           hours     grade
## hours 1.0000000 0.5356516
## grade 0.5356516 1.0000000
##
## $`Social Studies`
##           hours     grade
## hours 1.0000000 0.6472982
## grade 0.6472982 1.0000000
```

The correlation are all positive and range from 0.54 to 0.71. This agrees with the positive relation we see in the colored plot.

This problem is an example of Simpson's paradox. The example comes from https://www.quora.com/What-is-Simpsons-paradox/answer/Jon-Wayland (see also https://math.stackexchange.com/questions/83756/more-examples-of-simpsons-paradox-barring-the-ones-on-wikipedia-titanic-and). The point is that the subjects that require more hours of study, produce on average lower grades than other subjects which require less preparation.

Exercise 4

In this exercise we look at quantile plots. In all cases we will consider samples simulated from the normal distribution. We explore the effect of size, mean, and variance, and also use `qqplot` to compare samples.

(a) Divide the graphical window into four regions using `par` and `mfrow`. Generate four samples from the standard normal distribution of size 10 and draw normal quantile plots. Add lines with `qqline`. Comment on what you observe.

(b) Repeat for sample sizes 20, 50, and 100. Comment on what you observe.

(c) Draw samples of size 50 from normal distributions with means -6, -2, 2, and 6, all with variance 1 and draw the corresponding quantile plots. To be able to compare the four graphs, find a suitable common scale for the axes for all plots. Comment on the similarities and differences between the plots.

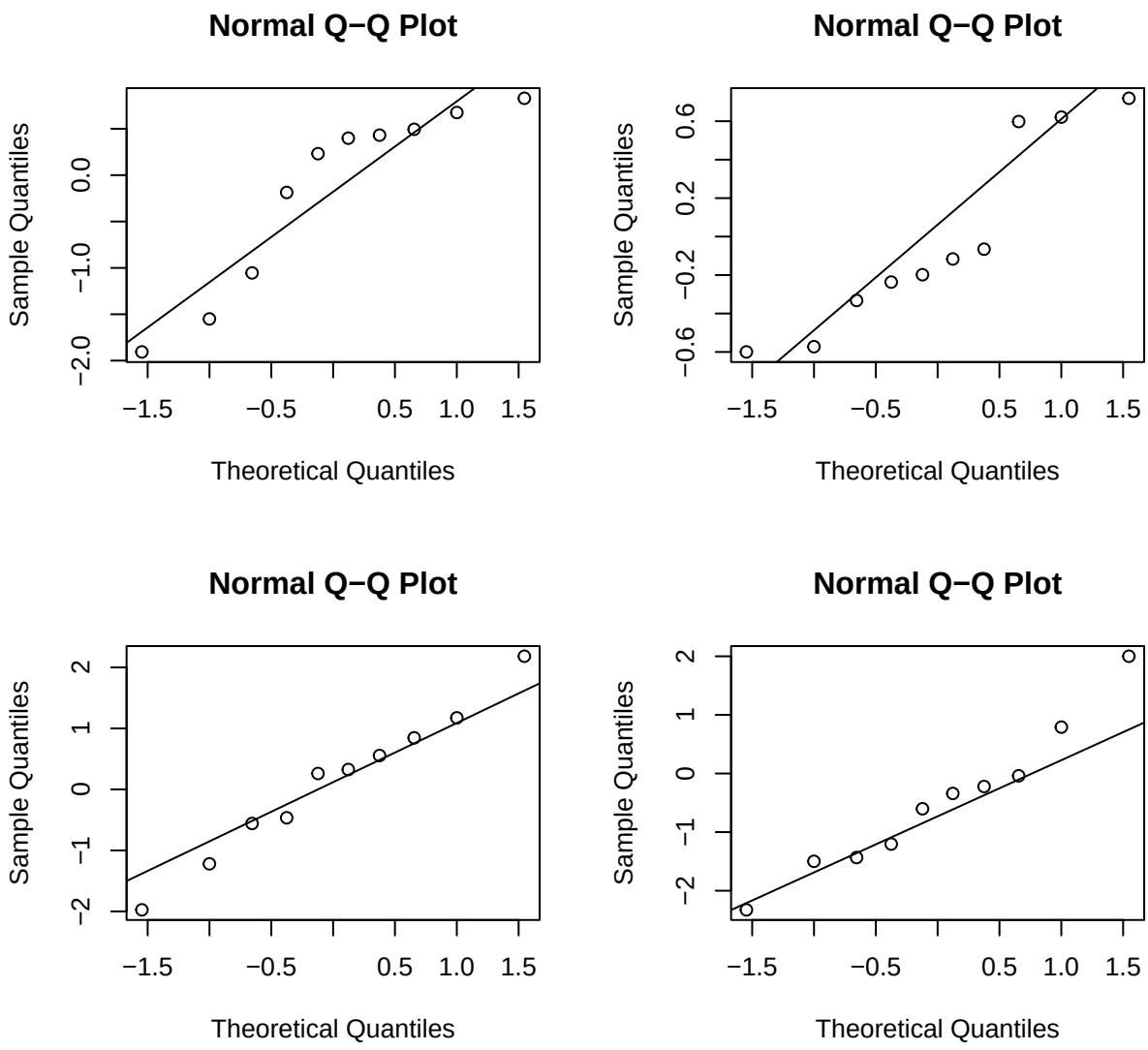
(d) Draw samples of size 50 from normal distributions with mean 1 and standard deviations 0.5, 2, 4, and 6, and draw the corresponding quantile plots. To be able to compare the four graphs, find a suitable common scale for the axes for all plots. Comment on the similarities and differences between the plots.

(e) Draw two samples of size 10 from the standard normal distribution and compare them using `qqplot`. Repeat a total of four times. Plot the four graphs on the same window. Comment on what you see.

Solution:

(a) Size 10:

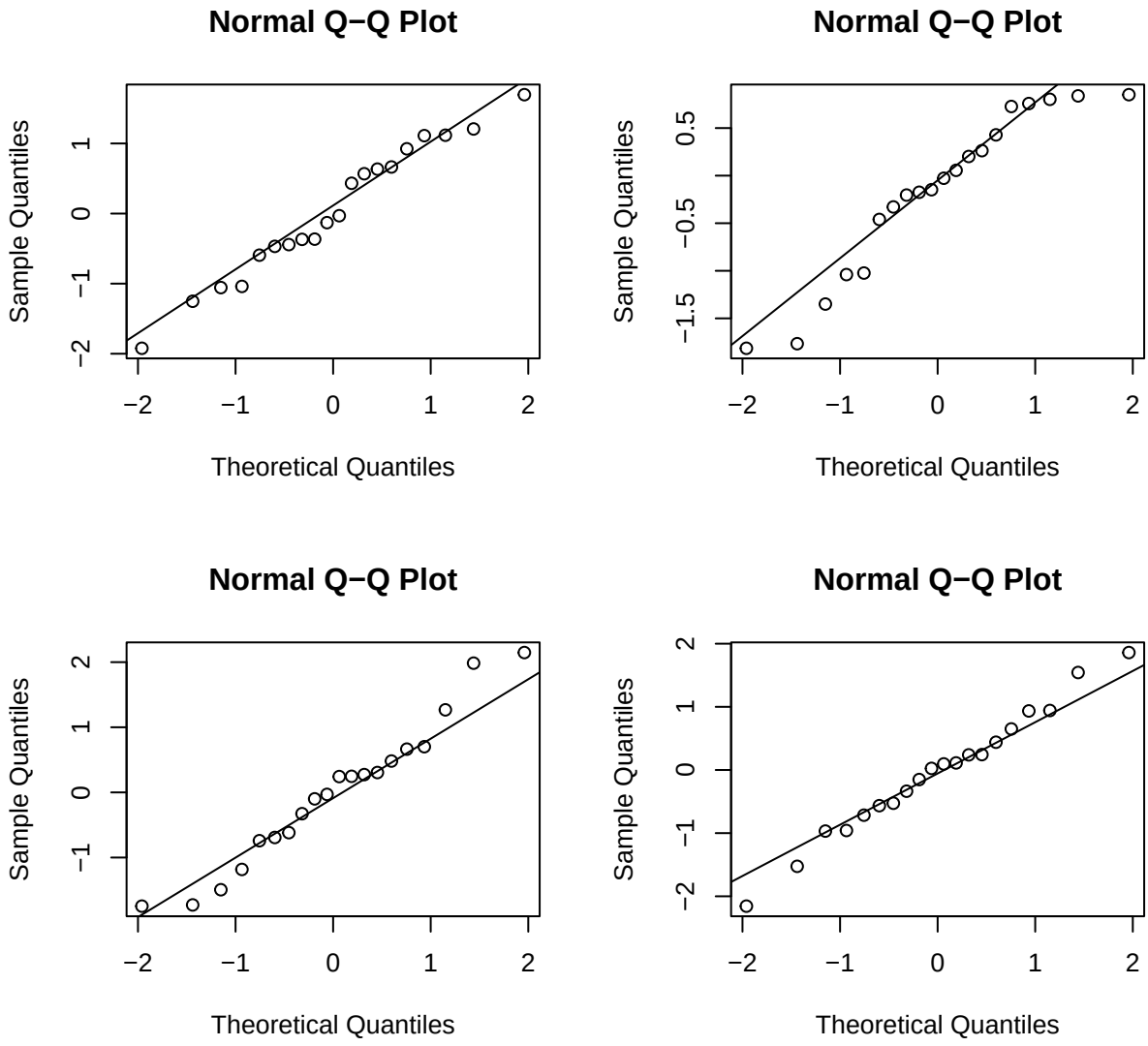
```
par(mfrow=c(2,2))
for(i in 1:4) {samp1 <- rnorm(10); qqnorm(samp1); qqline(samp1)}
```



Plots 1, 3, and 4 show important deviations from linearity. Plot 2 is reasonable.

(b) Size 20:

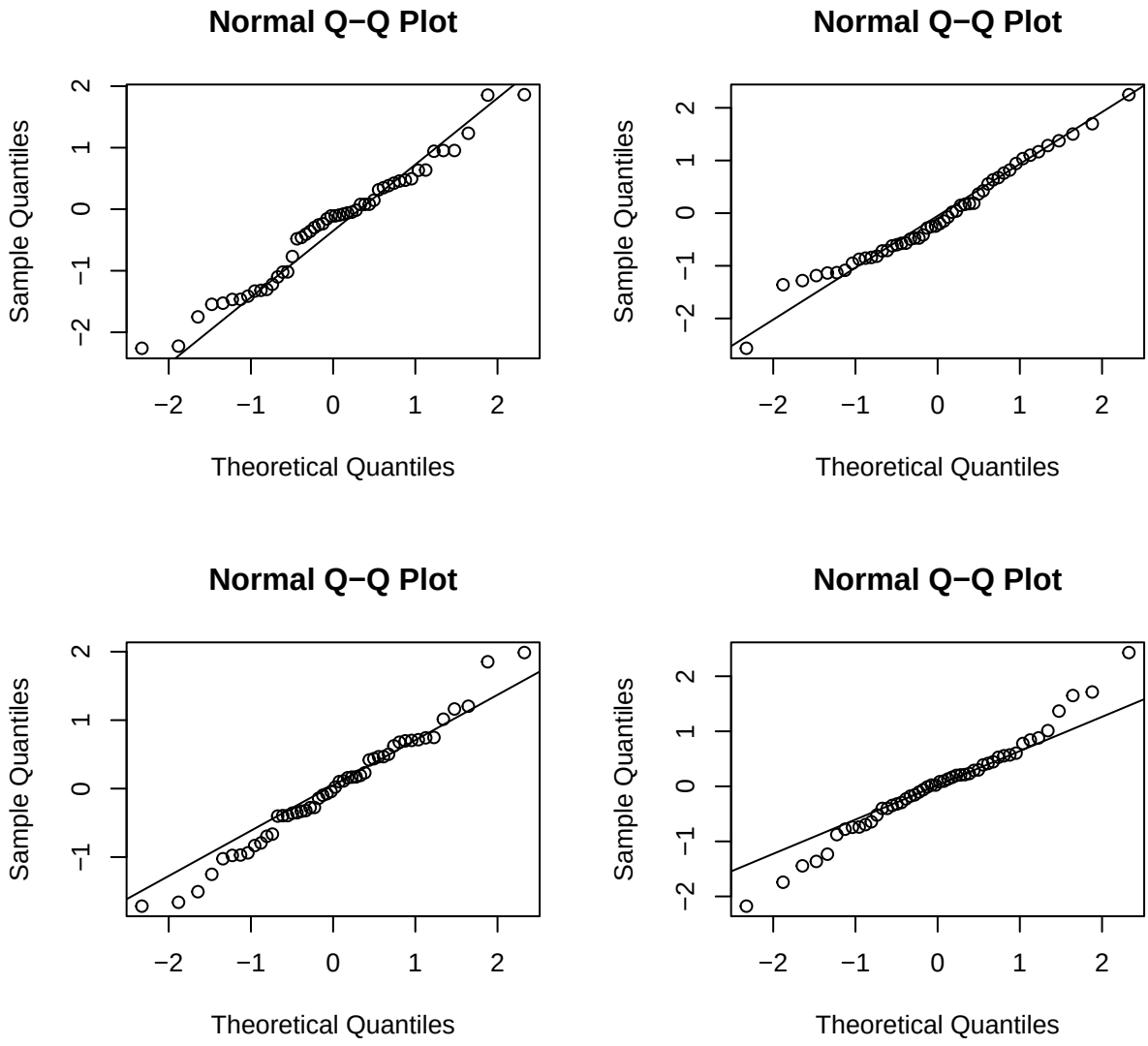
```
par(mfrow=c(2,2))
for(i in 1:4) {samp1 <- rnorm(20); qqnorm(samp1); qqline(samp1)}
```



For sample size 20 the fit is reasonable in plots 1, 3, and 4. Plot 2 shows important deviations from linearity.

Size 50:

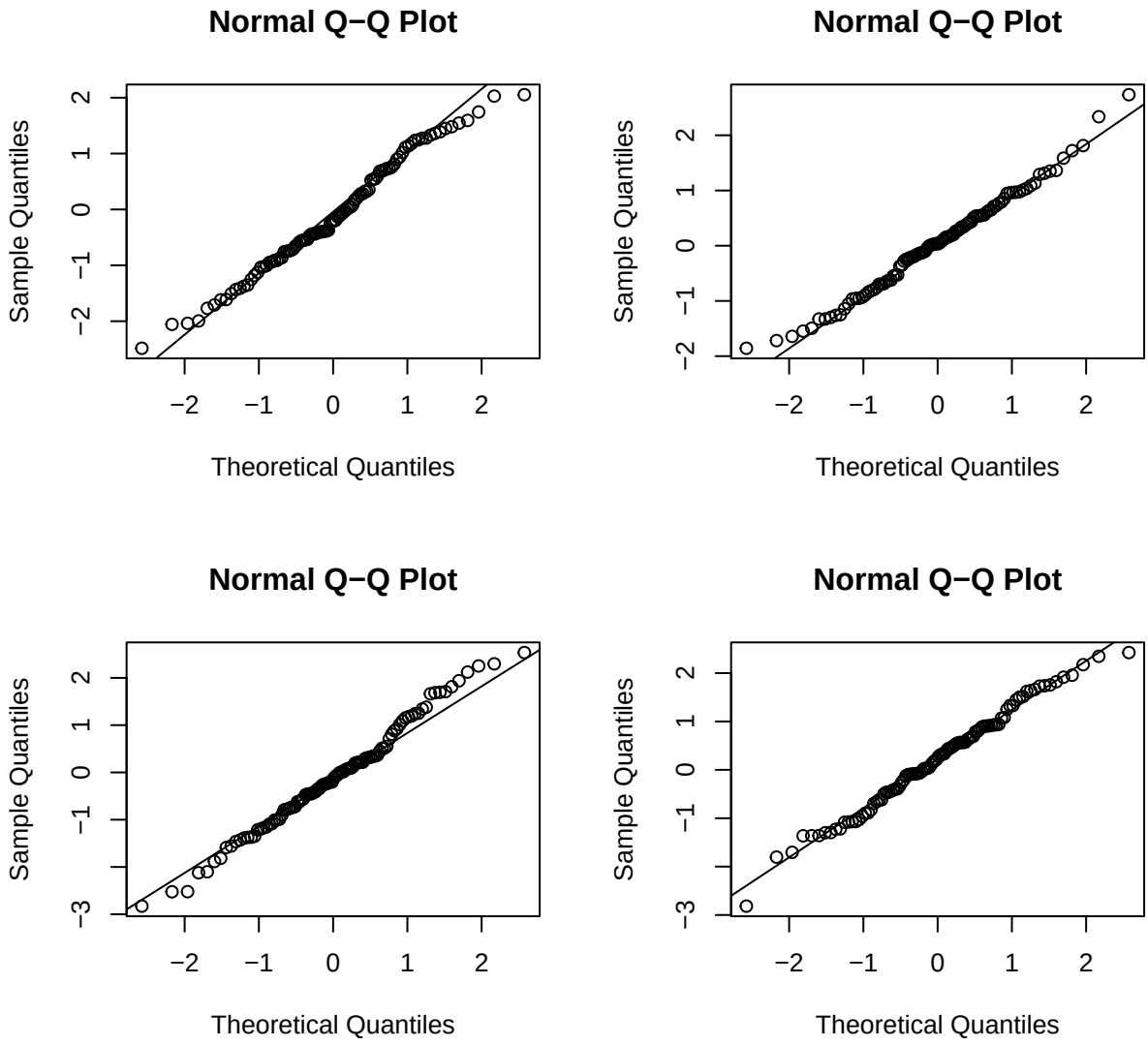
```
par(mfrow=c(2,2))
for(i in 1:4) {samp1 <- rnorm(50); qqnorm(samp1); qqline(samp1)}
```



For sample size 50 the fit is better. Plots 1 and 2 are very good, in plot 3 the fit is reasonable but there are deviations in the tails. Finally, in plot 4 the central part fits quite well, but there are large deviations in the tails.

Size 100:

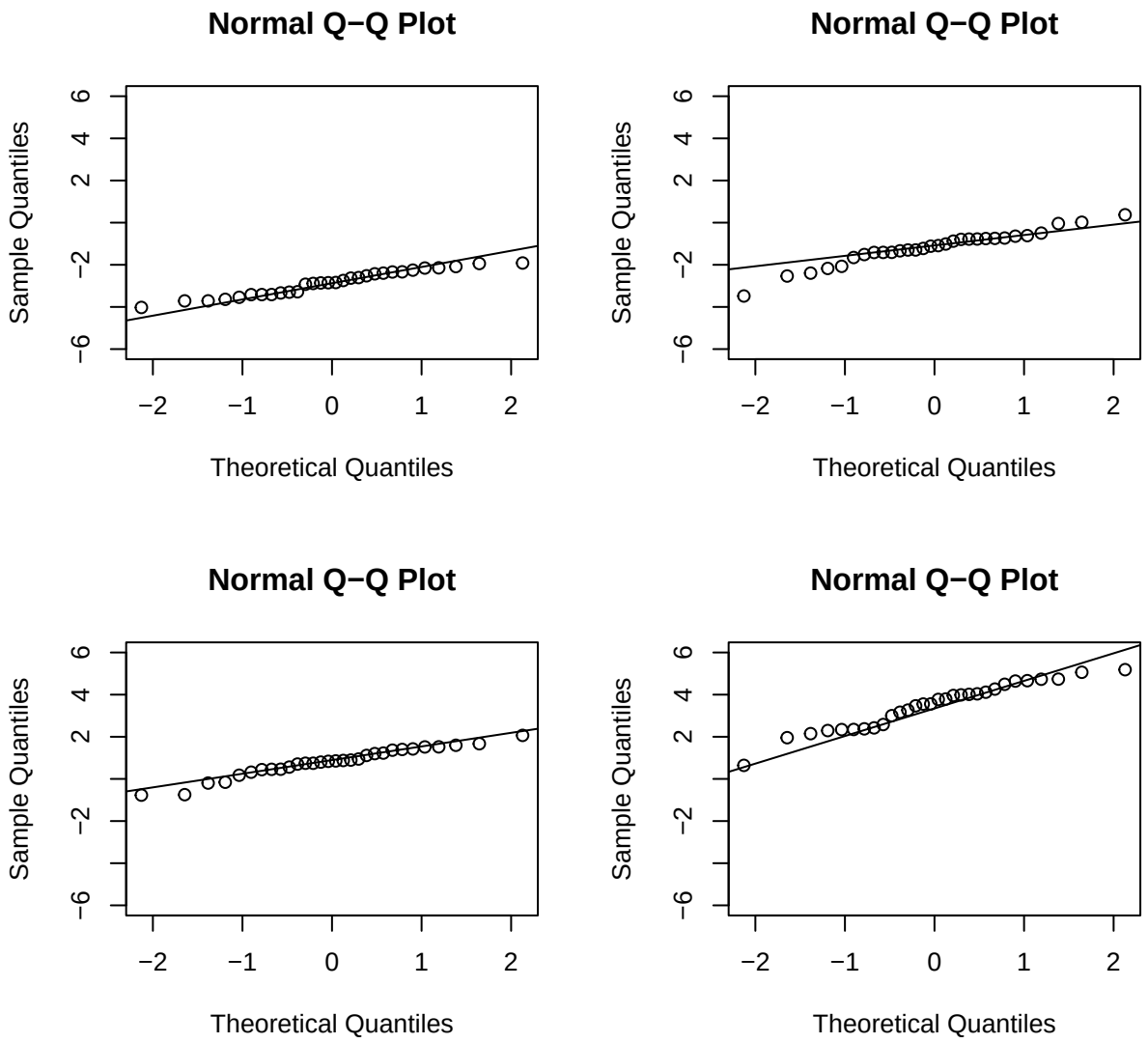
```
par(mfrow=c(2,2))
for(i in 1:4) {samp1 <- rnorm(100); qqnorm(samp1); qqline(samp1)}
```



Now the fit is very good iin all cases. We see that, as the sample size grows, the fit improves.

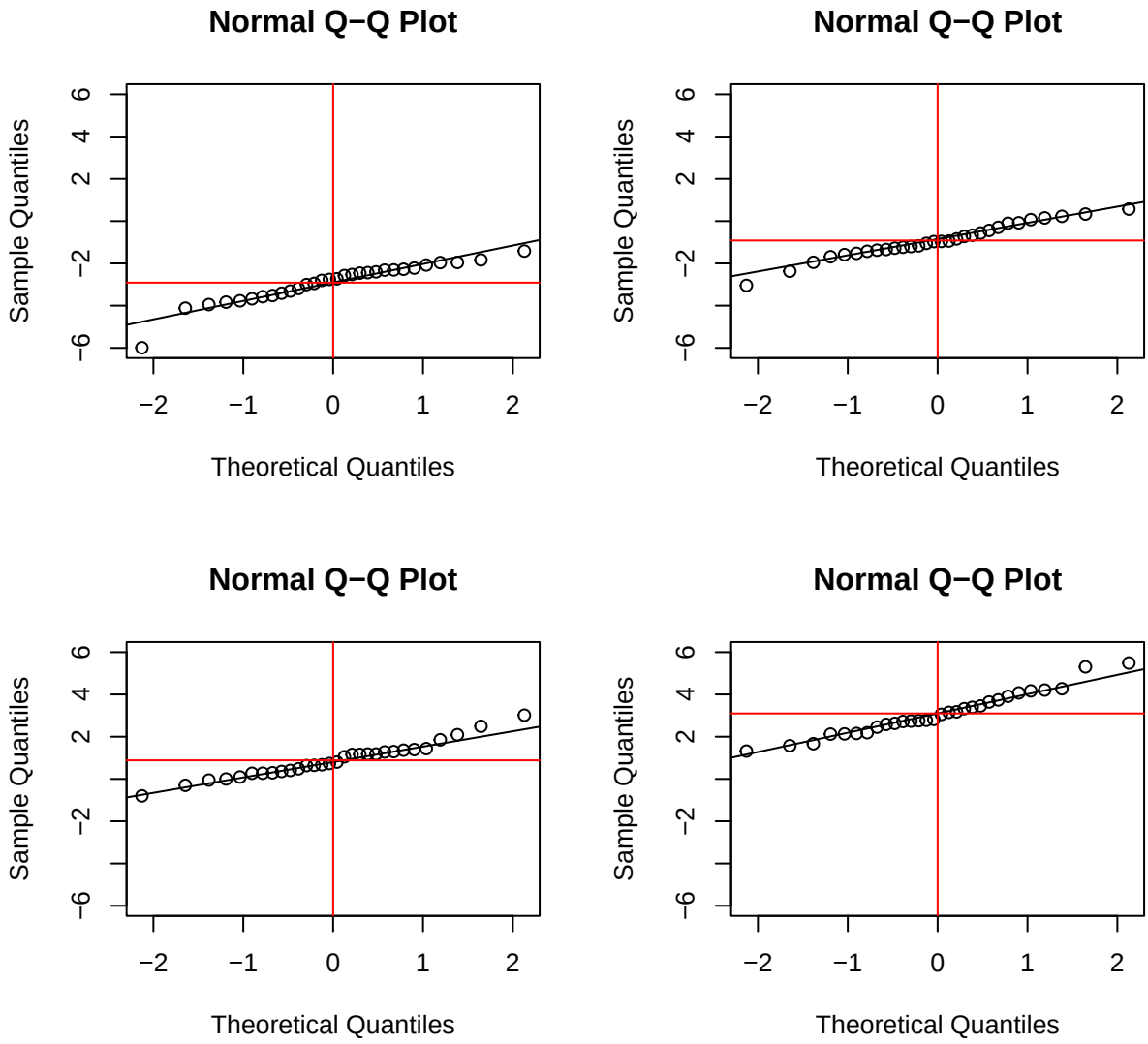
(c) Change in the mean:

```
par(mfrow=c(2,2))
for (i in c(-3,-1,1,3)) {
  dat <- rnorm(30,i);qqnorm(dat,ylim=c(-6,6));qqline(dat)}
```



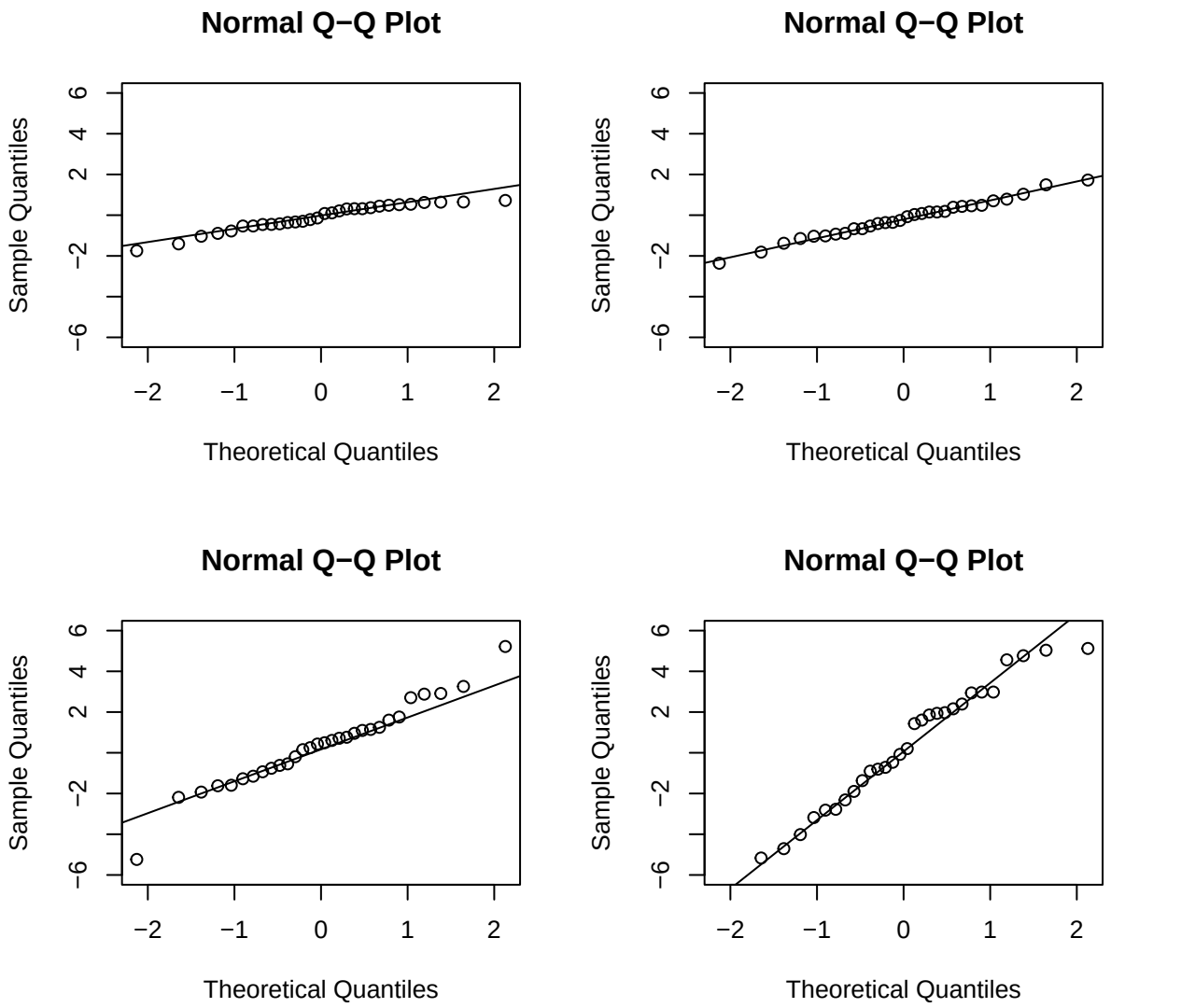
In the plots we see that the slope of the lines remain constant, but the lines shift upwards as the mean increases.

```r
par(mfrow=c(2,2))
for (i in c(-3,-1,1,3)) {
  dat <- rnorm(30,i);qqnorm(dat,ylim=c(-6,6));qqline(dat)
  abline(v=0,col='red'); abline(h=mean(dat),col = 'red')}
```



(d) Change in the variance:

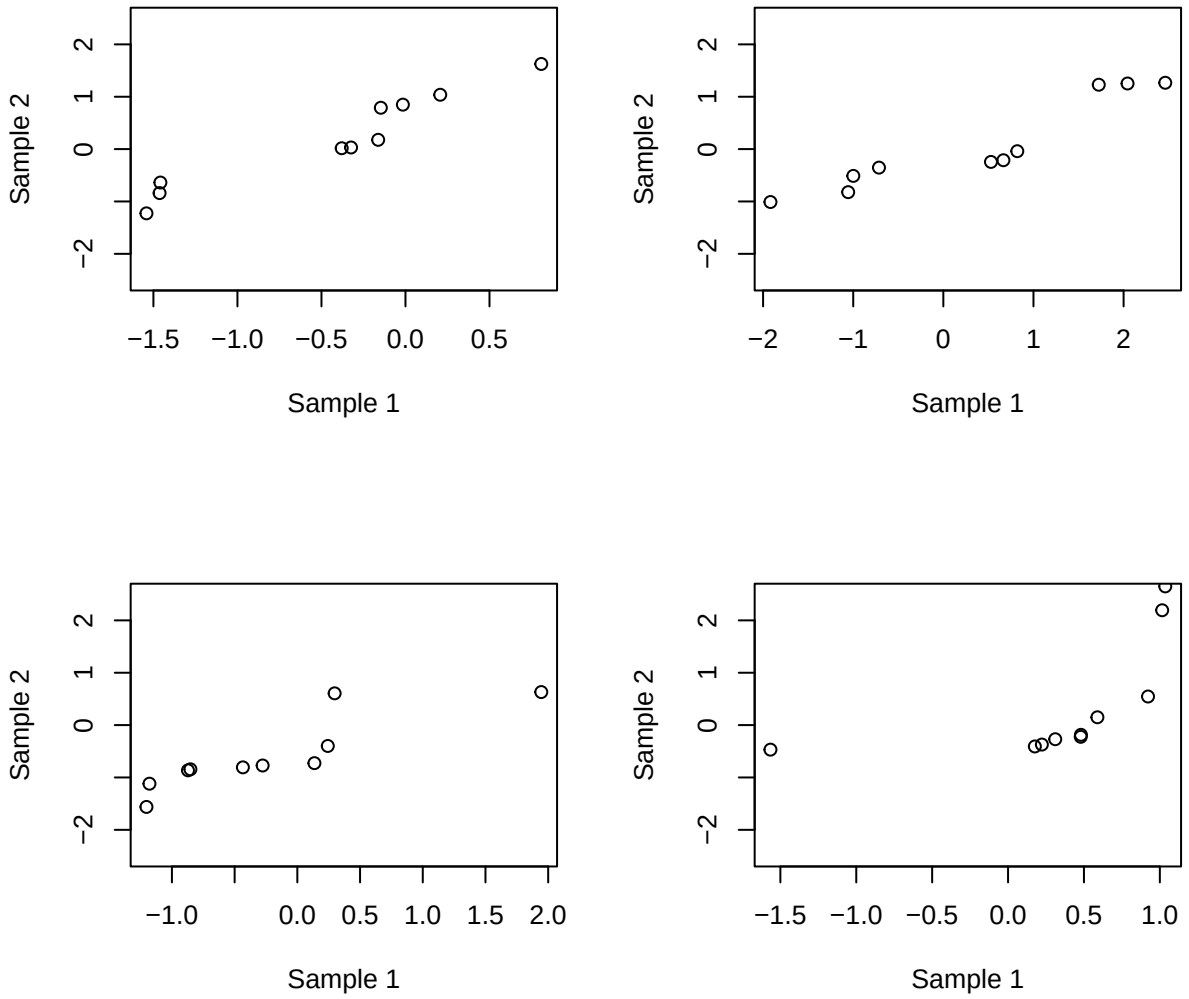
```
par(mfrow=c(2,2))
for (i in c(0.5,1,2,3)) {
  dat <- rnorm(30,0,i);qqnorm(dat,ylim=c(-6,6));qqline(dat)}
```



In these plots we see that the height of the central points remains constant, but the slope of the lines increases as the variance increases.

(e) Comparing two samples of size 10

```
par(mfrow=c(2,2))
for (i in 1:4) {
  dat <- rnorm(20);qqplot(dat[1:10],dat[11:20], ylim=c(-2.5,2.5),
                          xlab = 'Sample 1', ylab = 'Sample 2')}
```



We see that the first two plots show adequate alignment, and we would probably conclude that the two samples come from the same distribution. The last two plots show points that are not reasonably aligned, and we would conclude that they come from different distribution functions.