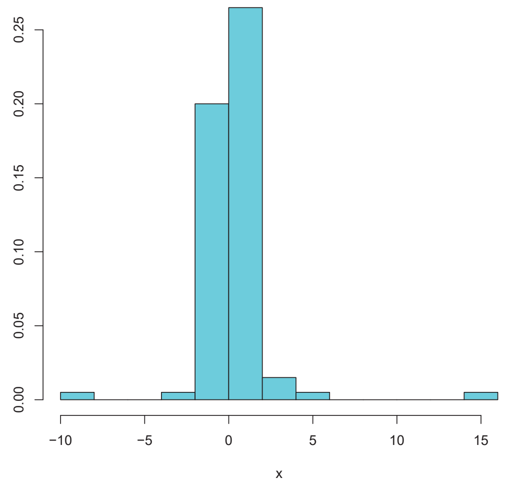
Chapter 1

Introduction

1.1 Generate a random sample $x_1, \ldots, x_{100}$ of data from the t_4 (df=4) distribution using the rt function. Use the MASS::truehist function to display a probability histogram of the sample.

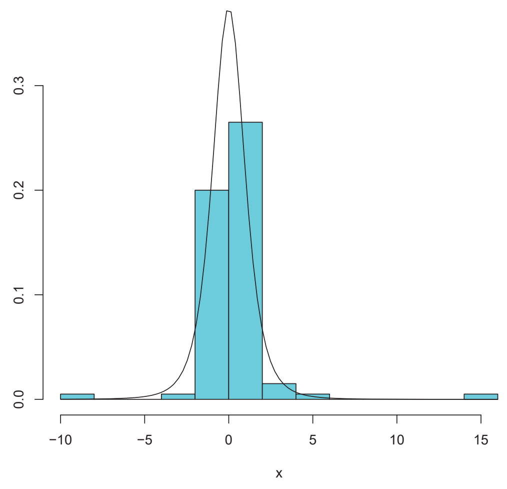
```r
library(MASS)
x <- rt(100, df = 4)
truehist(x)
```



1.2 Add the t_4 density curve (`dt`) to your histogram in Exercise 1.1 using the `curve` function with add=TRUE.

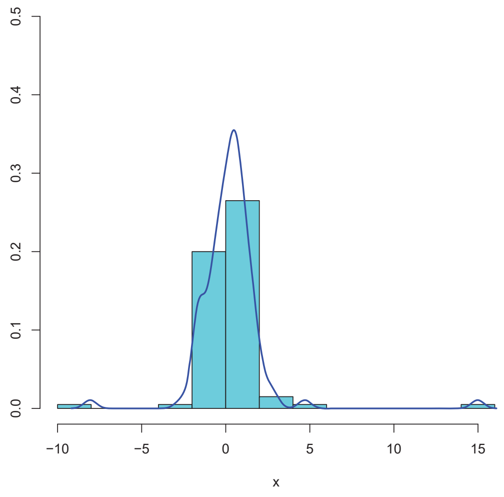
To avoid cutting off the top, we can set the y axis range to match the mode of the density at 0 using ylim in truehist.

```r
# using x from Exercise 1.1
y0 <- dt(0, df = 4)
truehist(x, ylim = c(0, y0))
curve(dt(x, df = 4), add = TRUE)
```



1.3 Add an estimated density curve to your histogram in Exercise 1.1 using `density`. Notice that the density estimate (`density`) is an approximation to the density of the sampled distribution (in this case the t_4 density). (Density estimation and the density function are covered in detail in Chapter 12.)

```r
# using x from Exercise 1.1
truehist(x, ylim = c(0, .5))
lines(density(x), col = 4, lwd = 2)
```



1.4 a. Write an R function f in R to implement the function

$$f(x) = \frac{x-a}{b}$$

that will transform an input vector x and return the result. The function should take three input arguments: x, a, b.

The return statement is optional; the last expression evaluated is returned.

```
f <- function(x, a, b) {
  return ((x - a) / b)
}

# try the function
f(10, 3, 2)

## [1] 3.5
```

b. To transform x to the interval $[0, 1]$ we subtract the minimum value and divide by the range:

```
y <- f(x, a = min(x), b = (max(x) - min(x))
```

Generate a random sample of Normal($\mu = 2, \sigma = 2$) data using `rnorm` and use your function f to transform this sample to the interval $[0, 1]$. Print a `summary` of both the sample x and the transformed sample y to check the result.

```
n <- 100
x <- rnorm(n, mean = 2, sd = 2)
y <- f(x, a = min(x), b = max(x) - min(x))

# one by one - harder to compare
summary(x)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.8001  0.4539  2.2470  1.9429  3.3832  8.2517

summary(y)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.2944  0.4567  0.4292  0.5595  1.0000

# nicer for comparison to create a data frame
dat <- data.frame(x = x, y = y)
summary(dat)

##        x                 y
##  Min.   :-2.8001   Min.   :0.0000
##  1st Qu.: 0.4539   1st Qu.:0.2944
##  Median : 2.2470   Median :0.4567
##  Mean   : 1.9429   Mean   :0.4292
##  3rd Qu.: 3.3832   3rd Qu.:0.5595
##  Max.   : 8.2517   Max.   :1.0000
```

1.5 Refer to Exercise 1.4. Suppose that we want to transform the x sample so that it has mean zero and standard deviation one (*studentize* the sample). That is, we want

$$z_i = \frac{x_i - \bar{x}}{s}, \quad i = 1, \ldots, n,$$

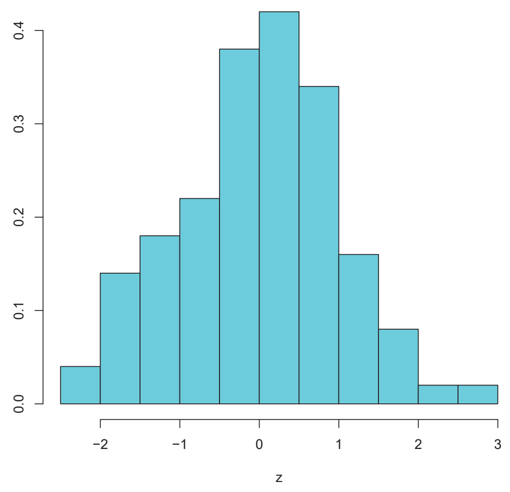
where s is the standard deviation of the sample. Using your function f this is

```
z <- f(x, a = mean(x), b = sd(x))
```

Display a summary and histogram of the studentized sample z. It should be centered exactly at zero. Use ‘sd(z)‘ to check that the studentized sample has standard deviation exactly 1.0.

```
# x sample from Exercise 1.4
z <- f(x, a = mean(x), b = sd(x))
summary(z)
```

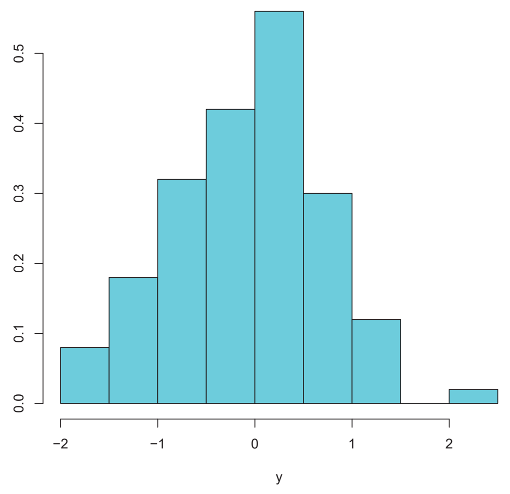
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.0890 -0.6558  0.1339  0.0000  0.6344  2.7786

truehist(z)
```



```
sd(z)

## [1] 1
```

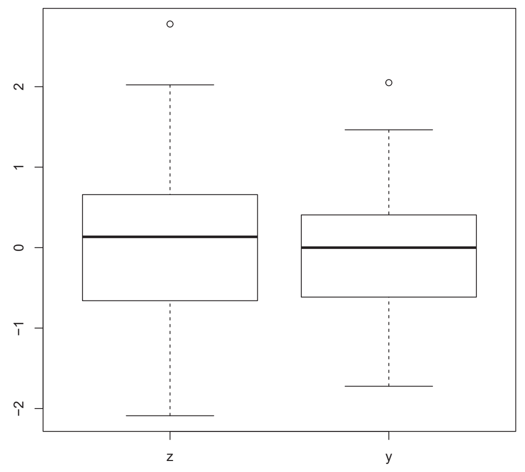
1.6 Using your function f of Exercise 1.4, center and scale your Normal($\mu = 2, \sigma = 2$) sample by subtracting the sample median and dividing by the sample interquartile range (IQR). Compare your results to Exercise 1.5.

```
# x sample from Exercise 1.4
y <- f(x, a = median(x), b = IQR(x))
summary(y)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.7229 -0.6121  0.0000 -0.1038  0.3879  2.0499

truehist(y)
```



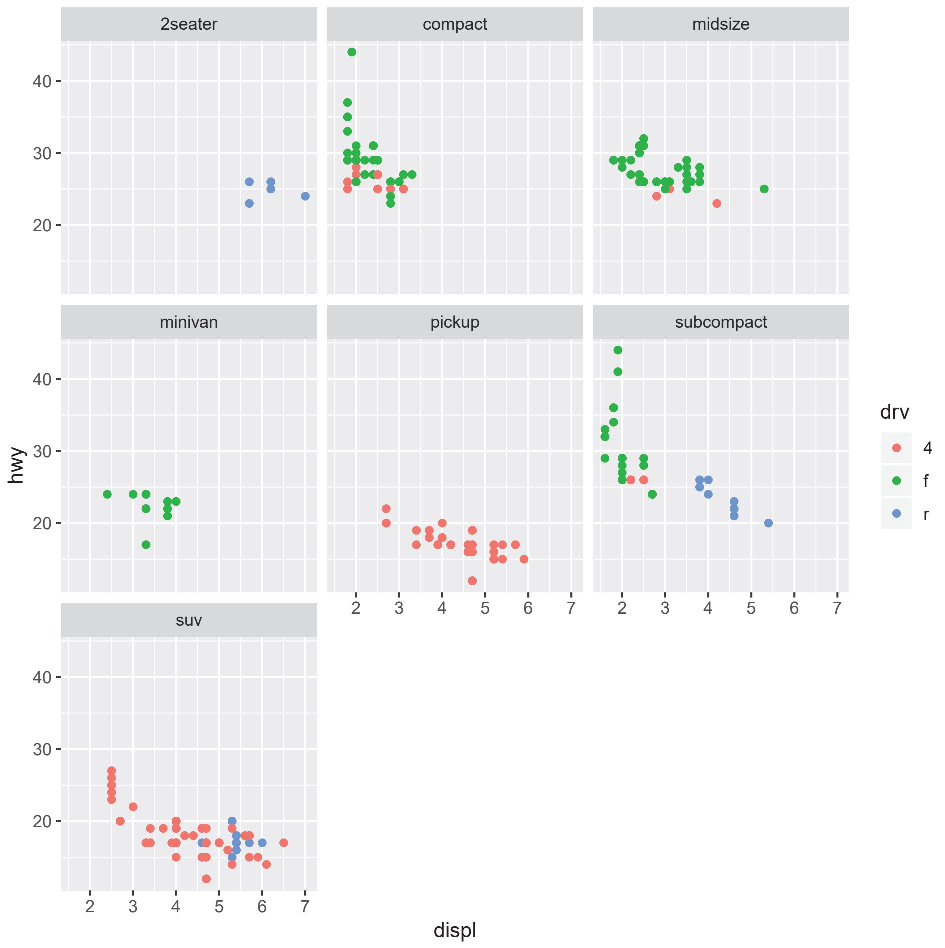
This transformation of the sample is centered close to zero with smaller range than the studentized sample. The sample median of y is zero but the sample mean of y is not zero. We can also compare with parallel boxplots.

```
df <- data.frame(z = z, y = y)
boxplot(df)
```



1.7 (ggplot) Refer to Example 1.14 where we displayed an array of scatterplots using `ggplot` with facet_wrap. One of the variables in the mpg data is drv, a character vector indicating whether the vehicle is front-wheel drive, rear-wheel drive, or four-wheel drive. Add color = drv in aes and display the revised plot. Your scatterplots should now have the three levels of drv coded by color and the plot should have automatically generated a legend for drv color.

```
library(ggplot2)

## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang

ggplot(mpg, aes(displ, hwy, color = drv)) +
  geom_point() +
  facet_wrap(~ class)
```



1.8 (RStudio and knitr) This exercise is intended to serve as an introduction to report writing with R Markdown. Install the *knitr* package if it is not installed. Create an html report using R Markdown and knitr in RStudio. The report should include the code and output of Examples 1.12 and 1.14 with appropriate headings and a brief explanation of each example.

- The knitr package should be installed from the Packages tab in RStudio.
- From the File menu in RStudio, select ”New File” and ”R Markdown ...” to open a basic template.
- Modify the title and author name.
- Replace the examples with text and code chunks for Examples 1.12 and 1.14.

- Save the file with file extension .Rmd (R Markdown).
- Click "Knit" on the editor toolbar to display the report.
- If needed, the html report is automatically saved in the current working directory.
- The report can be generated at any time from within RStudio by knitting the Rmd source file.

Note: It is not necessary to use labels in code chunks. For example, the label **cars** is not needed in

~~~
```{r cars}
summary(cars)
```
~~~

and

~~~
```{r}
summary(cars)
```
~~~

produces the same output.

Extra Question

(a)

$E(X) = E(E(X|N)) = E(N+1) = \lambda + 1 = 5 + 1 = 6$

$V(X) = E(V(X|N)) + V(E(X|N)) = E(2 \times (N+1)) + V(N+1) = 3\lambda + 2 = 3 * 5 + 2 = 17$

R:

```r
lambda <- 5
n <- 100
nrep <- 10^4

X.m <- rep(0,0)
X.v <- rep(0,0)
for(i in 1:nrep){
  N <- rpois(n, lambda)
  X <- rchisq(n, N+1)
  X.m[i] <- mean(X)
  X.v[i] <- var(X)
}

E_X <- mean(X.m)
var_X <- mean(X.v)

E_E_X_given_N <- lambda + 1
```

```r
E_var_X_giv_N <- 2*(lambda+1)
var_E_X_giv_N <- lambda
RHS <- var_E_X_giv_N + E_var_X_giv_N

A <- cbind(E_X,E_E_X_given_N,var_X,RHS)
 colnames(A)<- c("E(X)_simul" ,"E(E(X|N))_true","var(X)_simul" ,"E(var(X|N))+var(E(X|N))_true")
pander::pandoc.table(A,caption = "Check if E(X) = E(E(X|N)) and var(X) = E(var(X|N))+var(E(X|N))")
```

```
##
## ---------------------------------------------------------------------------
##  E(X)_simul   E(E(X|N))_true   var(X)_simul   E(var(X|N))+var(E(X|N))_true
## ------------ ---------------- -------------- ------------------------------
##    5.996            6             16.98                    17
## ---------------------------------------------------------------------------
##
## Table: Check if E(X) = E(E(X|N)) and var(X) = E(var(X|N))+var(E(X|N))
```

Python:

```python
import numpy as np
import pandas as pd
lamb, n,nrep =  5, 100, 10^5

X_m = []
X_v = []

for i in range(nrep):
  N = np.random.poisson(lamb, n)
  X = np.random.chisquare(N+1,n)

  X_m.append(np.mean(X))
  X_v.append(np.var(X))

E_X = np.mean(X_m)
var_X = np.mean(X_v)

E_E_X_given_N = lamb + 1
E_var_X_giv_N = 2*(lamb+1)
var_E_X_giv_N = lamb
RHS = var_E_X_giv_N + E_var_X_giv_N

df2 = pd.DataFrame({'E(X)_sim': [E_X], 'E(E(X|Y))_true': [E_E_X_given_N],
          'v(X)_sim': [var_X],'E(var(X|Y))+var(E(X|Y)_true': [RHS]})

print(df2)
```

```
##    E(X)_sim  E(E(X|Y))_true   v(X)_sim  E(var(X|Y))+var(E(X|Y)_true
## 0  6.069397               6  16.441164                            17
```

(b)

$E(X) = E(E(X|Y, P)) = E(YP) = E(Y)E(P) = \lambda\alpha/(\alpha + \beta) = 5 * 2/(2 + 3) = 2$

$V(X) = E(V(X|Y, P)) + V(E(X|Y, P)) = E(YP(1 - P)) + V(YP)$

$E(YP(1-P)) = E(Y)E(P-P^2) = \lambda(\alpha/(\alpha+\beta) - [(\alpha\beta/(\alpha+\beta)^2(\alpha+\beta+1)) + (\alpha/(\alpha+\beta))^2]))$

$V(YP) = V(Y)V(P) + V(Y)(E(P))^2 + V(P)(E(Y))^2 = \lambda\alpha\beta/((\alpha+\beta)^2(\alpha+\beta+1)) + \lambda(\alpha/(\alpha+\beta))^2) + (\alpha\beta/(\alpha+\beta)^2(\alpha+\beta+1))\lambda^2$

```r
lambda <- 5
a <- 2
b <- 3
n <- 100
nrep <- 10^4

X.m <- rep(0,0)
X.v <- rep(0,0)
for(i in 1:nrep){
  P <- rbeta(n,a,b)
  Y <- rpois(n,lambda)
  X <- rbinom(n,Y,P)
  X.m[i] <- mean(X)
  X.v[i] <- var(X)
}

E_X <- mean(X.m)
var_X <- mean(X.v)
E_P <- a/(a+b)
V_P <- a*b/(((a+b)^2)*(a+b+1))
E_Y <- V_Y <- lambda
E_P2 <- V_P+(E_P)^2

E_E_X_given_YP<- E_Y *E_P
E_var_X_giv_YP <- E_Y*(E_P-E_P2)
var_E_X_giv_YP <- V_Y*V_P + V_Y*(E_P^2)+V_P*(E_Y^2)
RHS <- var_E_X_giv_YP + E_var_X_giv_YP

A <- cbind(E_X,E_E_X_given_YP,var_X,RHS)
 colnames(A)<- c("E(X)_simul" ,"E(E(X|Y,N))_true","var(X)_simul" ,
                 "E(var(X|Y,P))+var(E(X|Y,P))_true")
pander::pandoc.table(A,caption = "Check if E(X) = E(E(X|Y,P)) and
                     var(X) = E(var(X|Y,P))+var(E(X|Y,P))")
```

```
##
## ---------------------------------------------------------------------------------
##  E(X)_simul   E(E(X|Y,N))_true   var(X)_simul   E(var(X|Y,P))+var(E(X|Y,P))_true
## ------------ ------------------ -------------- ----------------------------------
##    2.003             2              3.004                      3
## ---------------------------------------------------------------------------------
##
## Table: Check if E(X) = E(E(X|Y,P)) and
##                      var(X) = E(var(X|Y,P))+var(E(X|Y,P))
```