Week 1 Problem Based Learning and Practical Solutions

Jeffrey O. Hanson[1]
[1]*School of Biological Sciences, The University of Queensland, Brisbane, QLD, Australia*
Correspondance should be addressed to jeffrey.hanson@uqconnect.edu.au

15 April 2016

Contents

Problem based learning workshop . . . 1
Activity 1 . . . 1
Activity 2 . . . 7
Activity 3 . . . 9
R practical session . . . 10
General notes on using the teaching manual . . . 10
P1. Using R as a calculator . . . 10
P2. Joining stuff together . . . 11
P3. Viewing data . . . 11
P4. Importing your own data . . . 15
P5. Graphing data . . . 20
P6. Simulating dice throws . . . 24
P7. Simulating data within a 'for loop' . . . 25

Problem based learning workshop

Activity 1

1. *Write down the probabilities for rolling a 1 to a 6 using a fair, and using an unfair die. Calculate the mean of their sum for each die.*

   - Fair dice
     - The following table has the probability for each outcome.

Die number	Probability
1	$\frac{1}{6} = 0.167$
2	$\frac{1}{6} = 0.167$
3	$\frac{1}{6} = 0.167$
4	$\frac{1}{6} = 0.167$
5	$\frac{1}{6} = 0.167$
6	$\frac{1}{6} = 0.167$

     –

$$
\begin{aligned}
\text{mean of sum} &= (\frac{1}{6} \times 1) + (\frac{1}{6} \times 2) + (\frac{1}{6} \times 3) + (\frac{1}{6} \times 4) + (\frac{1}{6} \times 5) + (\frac{1}{6} \times 6) \\
&= 3.5
\end{aligned}
$$

- Unfair dice
  - **This unfair die has the numbers 5 and 6 replaced with a 1.**
  - The following table has the probability for each outcome.

Die number	Probability
1	$\frac{3}{6} = 0.5$
2	$\frac{1}{6} = 0.167$
3	$\frac{1}{6} = 0.167$
4	$\frac{1}{6} = 0.167$
5	$\frac{0}{6} = 0$
6	$\frac{0}{6} = 0$

  -
  -

$$
\begin{aligned}
\text{mean of sum} &= (\frac{3}{6} \times 1) + (\frac{1}{6} \times 2) + (\frac{1}{6} \times 3) + (\frac{1}{6} \times 4) + (\frac{0}{6} \times 5) + (\frac{0}{6} \times 6) \\
&= 2
\end{aligned}
$$

2. *Write down the expected probabilities for the possible sums when throwing two fair die. Compare them to the expected probabilities for throwing possible sums when throwing one fair and one unfair die.*

- Fair dice
  - First, let's write out all possible outcomes.

```r
df1 <- expand.grid(D1=1:6, D2=1:6) %>%
    mutate(sum=D1+D2)
kable(
    df1,
    digits=2,
    align=c('c', 'c', 'c'),
    col.names=c('First dice', 'Second dice', 'Sum')
)
```

First dice	Second dice	Sum
1	1	2
2	1	3
3	1	4
4	1	5

First dice	Second dice	Sum
5	1	6
6	1	7
1	2	3
2	2	4
3	2	5
4	2	6
5	2	7
6	2	8
1	3	4
2	3	5
3	3	6
4	3	7
5	3	8
6	3	9
1	4	5
2	4	6
3	4	7
4	4	8
5	4	9
6	4	10
1	5	6
2	5	7
3	5	8
4	5	9
5	5	10
6	5	11
1	6	7
2	6	8
3	6	9
4	6	10
5	6	11
6	6	12

– Now let's use this to calculate the probability for throwing each sum. The total number of different outcomes is the number of rows in the table (36). The probability of a given sum occurring is the number of times the sum appears in the previous table (ie. the frequency) divided by the number of potential outcomes (ie. 36).

```r
df2 <- df1 %>% mutate(sum=factor(sum)) %>%
    group_by(sum) %>%
    summarize(freq=length(sum)) %>%
    ungroup() %>%
    mutate(prob=paste0('$\\frac{',freq,'}{',nrow(df1),'} = ',
        round(freq/nrow(df1),3),'$'))
kable(
    df2,
    digits=3,
    align=c('c', 'c', 'c'),
    col.names=c('Sum', 'Frequency', 'Probability')
)
```

Sum	Frequency	Probability
2	1	$\frac{1}{36} = 0.028$
3	2	$\frac{2}{36} = 0.056$
4	3	$\frac{3}{36} = 0.083$
5	4	$\frac{4}{36} = 0.111$
6	5	$\frac{5}{36} = 0.139$
7	6	$\frac{6}{36} = 0.167$
8	5	$\frac{5}{36} = 0.139$
9	4	$\frac{4}{36} = 0.111$
10	3	$\frac{3}{36} = 0.083$
11	2	$\frac{2}{36} = 0.056$
12	1	$\frac{1}{36} = 0.028$

- Unfair dice
  - **This unfair die has the numbers 5 and 6 replaced with a 1.**
  - First, let's write out all possible outcomes.

```r
df3 <- expand.grid(D1=1:6, D2=c(1,1,1,2,3,4)) %>% mutate(sum=D1+D2)
kable(
    df3,
    digits=3,
    align=c('c', 'c', 'c'),
    col.names=c('First dice', 'Second dice', 'Sum')
)
```

First dice	Second dice	Sum
1	1	2
2	1	3
3	1	4
4	1	5
5	1	6

First dice	Second dice	Sum
6	1	7
1	1	2
2	1	3
3	1	4
4	1	5
5	1	6
6	1	7
1	1	2
2	1	3
3	1	4
4	1	5
5	1	6
6	1	7
1	2	3
2	2	4
3	2	5
4	2	6
5	2	7
6	2	8
1	3	4
2	3	5
3	3	6
4	3	7
5	3	8
6	3	9
1	4	5
2	4	6
3	4	7
4	4	8
5	4	9
6	4	10

– Now let's use this to calculate the probability for throwing each sum.

```
df4 <- df3 %>% mutate(sum=factor(sum)) %>%
    group_by(sum) %>%
    summarize(freq=length(sum)) %>%
    ungroup() %>%
    mutate(
        prob=paste0('$\\frac{',freq,'}{',nrow(df3),'} = ',
        round(freq/nrow(df3),3),'$'))
kable(
    df4,
    digits=2,
    align=c('c', 'c', 'c'),
    col.names=c('Sum', 'Frequency', 'Probability')
)
```

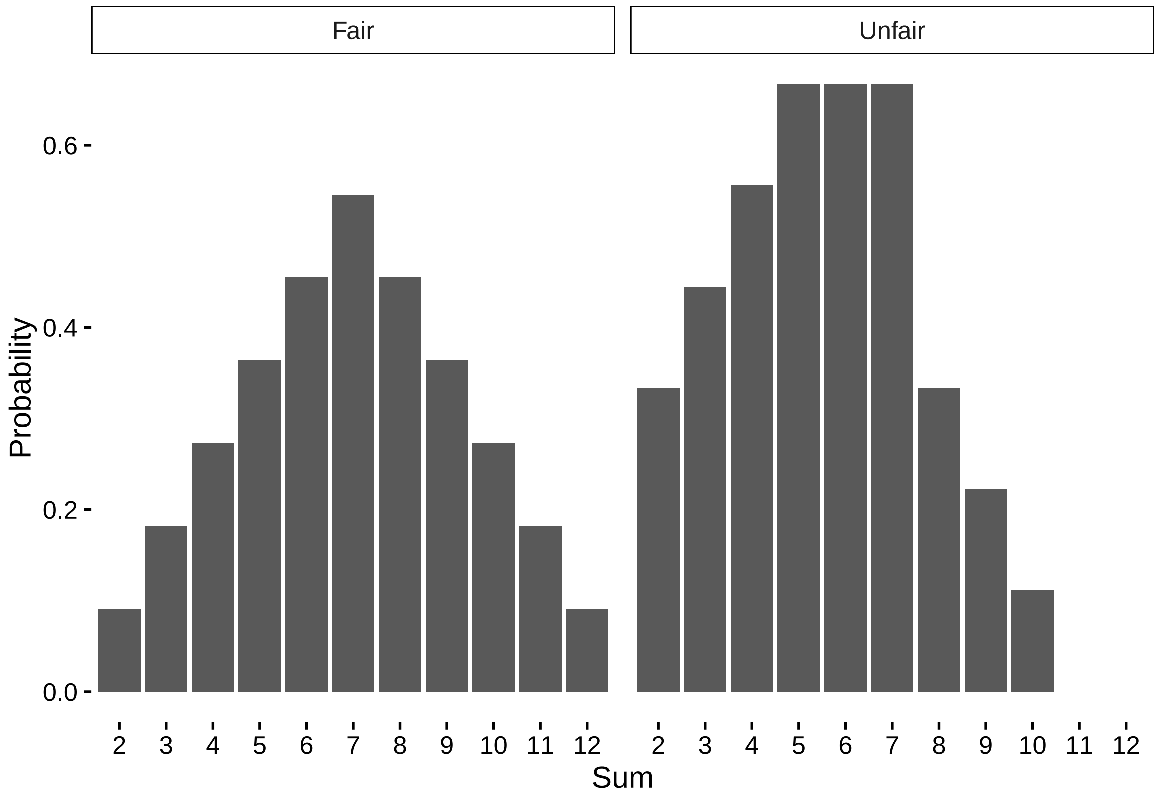
Sum	Frequency	Probability
2	3	$\frac{3}{36} = 0.083$
3	4	$\frac{4}{36} = 0.111$
4	5	$\frac{5}{36} = 0.139$
5	6	$\frac{6}{36} = 0.167$
6	6	$\frac{6}{36} = 0.167$
7	6	$\frac{6}{36} = 0.167$
8	3	$\frac{3}{36} = 0.083$
9	2	$\frac{2}{36} = 0.056$
10	1	$\frac{1}{36} = 0.028$

3. Draw the probability distribution for the sum of two fair dice and compare it to the probability distribution for the sum of one fair and one unfair die.

    - We will use the probabilities calculated in the previous question to draw these bars.

```
df5 <- rbind(
    mutate(df4, Probability=freq/nrow(df4), Die='Unfair'),
    mutate(df2, Probability=freq/nrow(df2), Die='Fair')) %>%
    rename(Sum=sum)

ggplot(data=df5, aes(x=Sum, y=Probability)) +
    geom_bar(stat='identity') +
    facet_wrap(~ Die) +
    theme_classic()
```



4. Discuss with your partner the idea of null distribution versus observed distribution.

   - The observed distribution is the data you collect
   - The null distribution represents the distribution you expect and want to compare the data to.

Activity 2

- Instructions
  - Roll 2 fair die 10 times

    ```
    outcomes.10 <- replicate(sample(1:6, 2, replace=TRUE), n=10)
    ```

  - Record the outcomes

    ```
    print(outcomes.10)
    ```

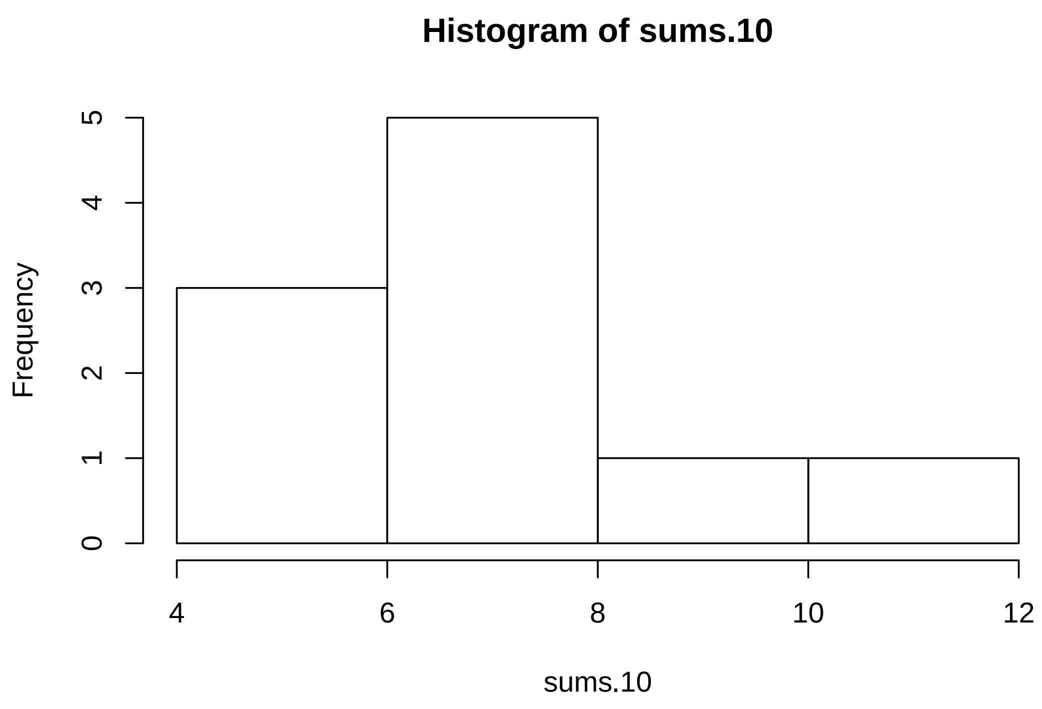
    ```
    ##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
    ## [1,]    4    6    2    2    6    6    1    5    2     2
    ## [2,]    4    1    6    4    3    6    5    3    2     6
    ```

  - Count how many times each sum occurs

    ```
    sums.10 <- apply(outcomes.10, 2, sum)
    ```

  - Plot a histogram showing the frequency of each sum.

    ```
    hist(sums.10)
    ```



– Repeat 1-3 but rolling the dice 40 times.

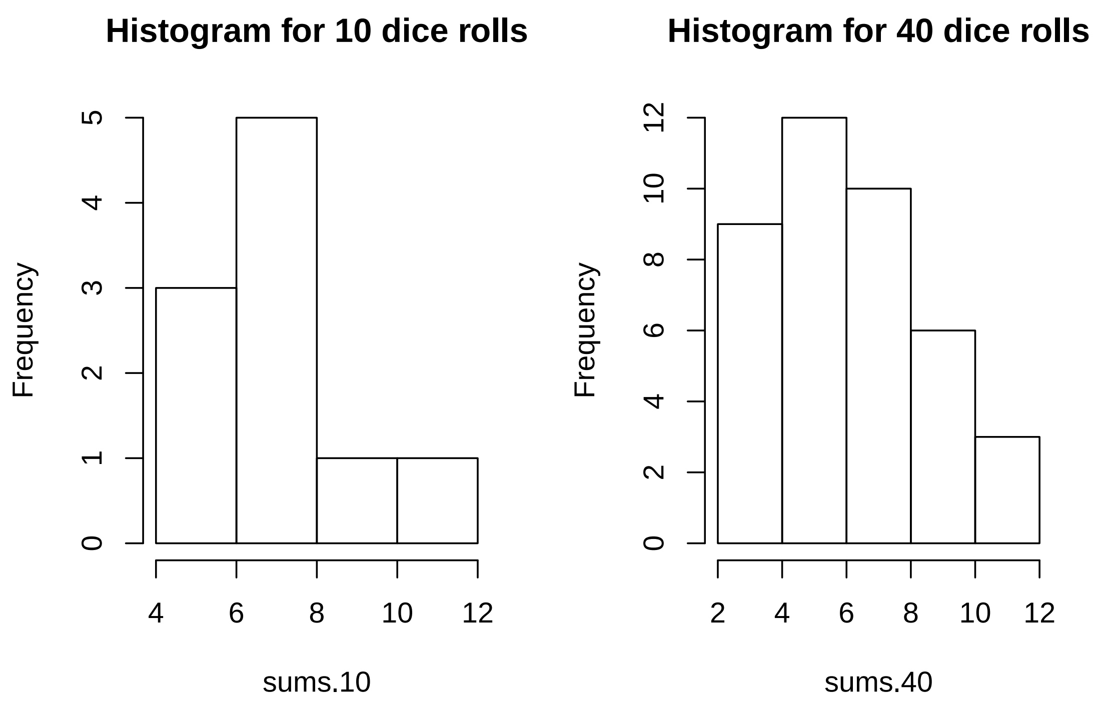
```
outcomes.40 <- replicate(sample(1:6, 2, replace=TRUE), n=40)
print(outcomes.40)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
## [1,]    2    2    2    6    6    2    2    1    3     1     1     2     5
## [2,]    4    2    3    6    3    3    4    1    6     3     6     6     3
##      [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
## [1,]     5     1     5     5     4     1     3     1     2     5     6
## [2,]     1     5     4     1     5     5     1     2     6     6     3
##      [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35]
## [1,]     1     2     5     1     1     1     3     2     4     1     6
## [2,]     5     2     2     5     1     5     6     2     3     6     1
##      [,36] [,37] [,38] [,39] [,40]
## [1,]     4     3     1     3     5
## [2,]     1     5     3     4     6
```

```
sums.40 <- apply(outcomes.40, 2, sum)
```

– Compare the two histograms

```
par(mfrow=c(1,2))
hist(sums.10, main='Histogram for 10 dice rolls')
hist(sums.40, main='Histogram for 40 dice rolls')
```



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

1. What is the mean value of the observed sums for each experiment.

   - The mean values are 7.6 and 6.55.

2. How does the mean value obtained (from rolling dice) change with increasing number of rolls?

   - The mean value gets closer to the estimate based on probability calculations with more rolls.

3. How does the width of the histogram of the means change with increasing number of rolls?

   - The width of the histogram gets smaller.

Activity 3

- Instructions
  - Divide into two teams of 8 per group.
  - Each team chooses a pair of dice.
  - Each team rolls the dice 30 times for the other team (do not let them see your dice).
  - The other team records the values of the two dice for each of the 30 rolls.
  - Exchange rolls.
  - Here are the rolls we recieved from an imaginary team:

1. Can you tell whether the other team used a modified die?
2. What would you do to test the hypothesis that the other team used a modified die?

R practical session

General notes on using the teaching manual

- network information
- always use Windows 7
  - Map to L drive: `\\\\sci-tl01.soe.uq.edu.au\\Teaching\\BIOL\\BIOL2006`
  - Program name: `BIOL2006PracManual_2016.exe`
- sign on
  - must enter valid id
  - my sign on is `"Jeffrey"`
- GUI
  - popup links are bright green
  - arrows to navigate
  - home icon is home
  - R icon saves code to desktop and open up with Rstudio
  - R code is in blue
  - comments in black
  - questions also in green
  - tooltips appear in brown bar along the bottom of program
  - minimize question popups by clicking on question again
  - assessment questions turn black after they're done, but note that you might have to mouse over questions for them to turn black due to bug
  - students must click logout to store results, they cannot just close the program
  - students must answer all practice questions to get to final assessment questions
  - each question in the assessment page refers to a specific page with info on how to answer it
- R scripts
  - right click on the scripts
  - open with -> Rstudio
- Potential issues
  - the computers don't associate .R scripts with Rstudio: need to open Rstudio first then open the script
  - attaching datasets: **never to do this even though it looks handy**

P1. Using R as a calculator

```r
1 + 2
```

```
## [1] 3
```

```r
log(10)/( 41 + exp(33.8))
```

```
## [1] 4.820168e-15
```

```r
x <- (48 + 34)/5.5       # store the result as 'x'
x         # look at the contents of 'x'
```

```
## [1] 14.90909
```

```r
y <- x + 3
x*y
```

```
## [1] 267.0083
```

P2. Joining stuff together

```r
stuff <- c(1,"cat", 4.5, 6, "elephant")     # create some "stuff"
class(stuff)                         # what type is it?
```

```
## [1] "character"
```

```r
stuff               # have a look ; note that "4.5" is not a number!
```

```
## [1] "1"        "cat"      "4.5"      "6"        "elephant"
```

```r
numbers <- c(1,2.8,3,4.5,12.22,13)  # create "numbers"
class(numbers)                # what type is it?
```

```
## [1] "numeric"
```

```r
paste(stuff,numbers)      # join things together as character strings
```

```
## [1] "1 1"           "cat 2.8"       "4.5 3"         "6 4.5"
## [5] "elephant 12.22" "1 13"
```

```r
3:7       # create a consecutive sequence of numbers
```

```
## [1] 3 4 5 6 7
```

```r
paste(1:4,stuff,numbers,"whatever!")
```

```
## [1] "1 1 1 whatever!"            "2 cat 2.8 whatever!"
## [3] "3 4.5 3 whatever!"          "4 6 4.5 whatever!"
## [5] "1 elephant 12.22 whatever!" "2 1 13 whatever!"
```

P3. Viewing data

```r
women # display the data set called 'women'
```

```
##    height weight
## 1      58    115
## 2      59    117
## 3      60    120
## 4      61    123
## 5      62    126
## 6      63    129
## 7      64    132
## 8      65    135
## 9      66    139
## 10     67    142
## 11     68    146
## 12     69    150
## 13     70    154
## 14     71    159
## 15     72    164
```

```r
dim(women) # display the numbers of rows and columns
```

```
## [1] 15  2
```

```r
names(women) # display the column headings
```

```
## [1] "height" "weight"
```

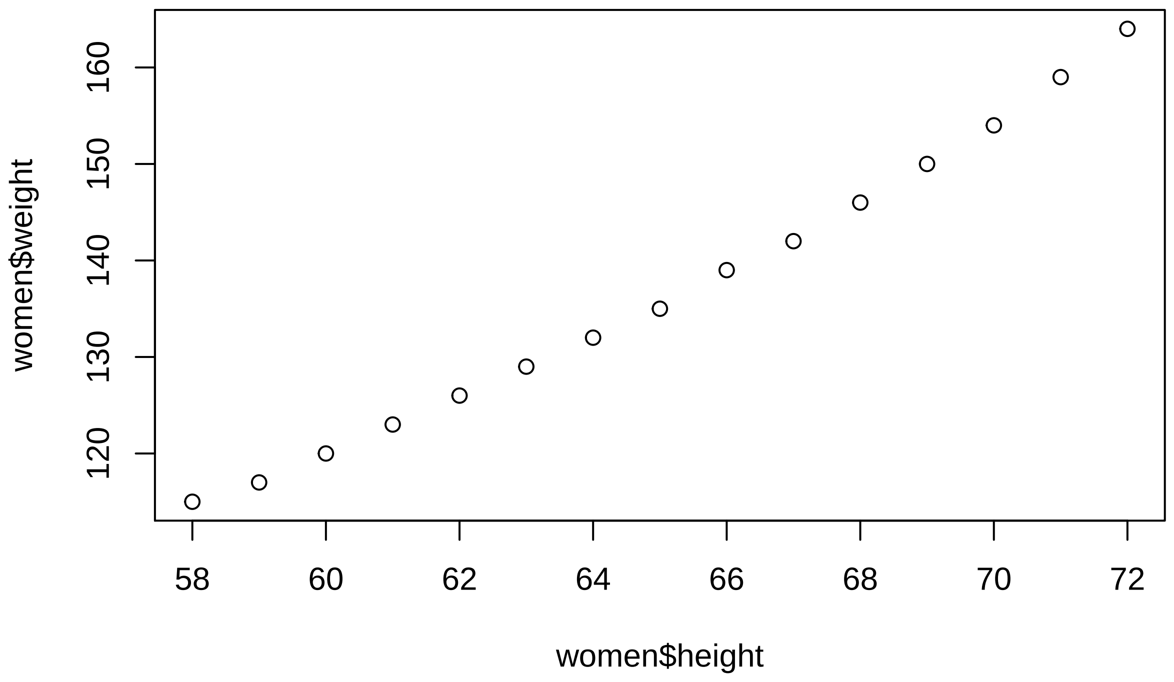
```r
women$weight # display the column called 'weight' within 'women'
```

```
##  [1] 115 117 120 123 126 129 132 135 139 142 146 150 154 159 164
```

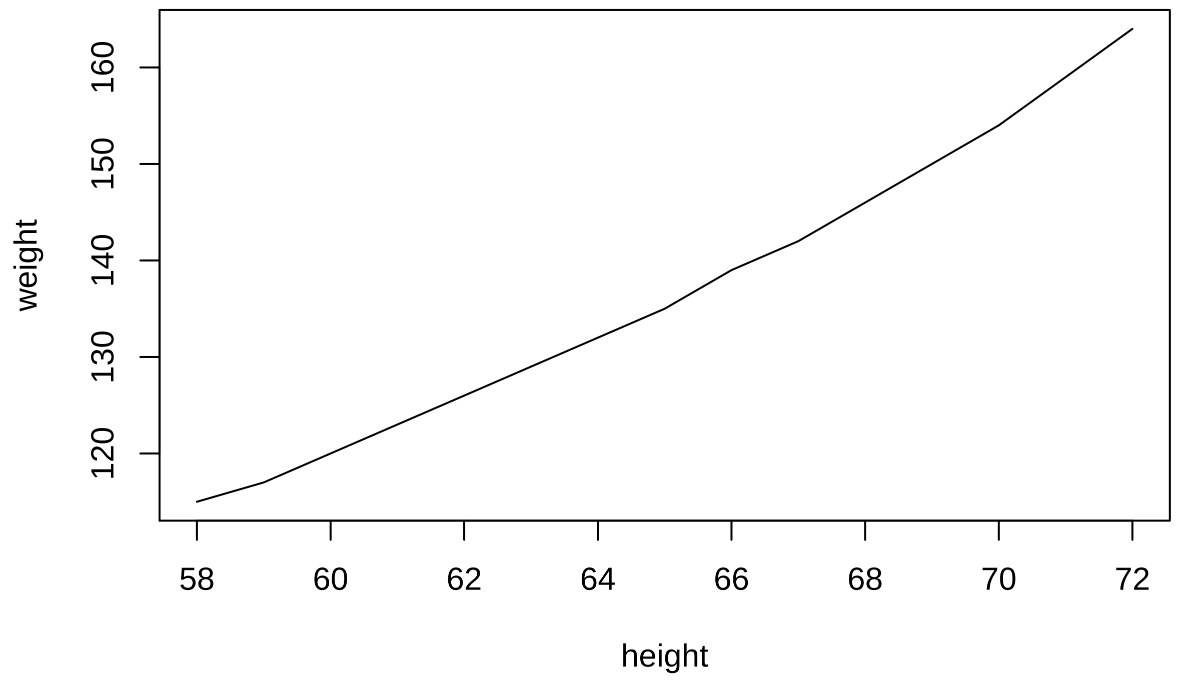
```r
women[4,2] # display row 4 column 2
```

```
## [1] 123
```

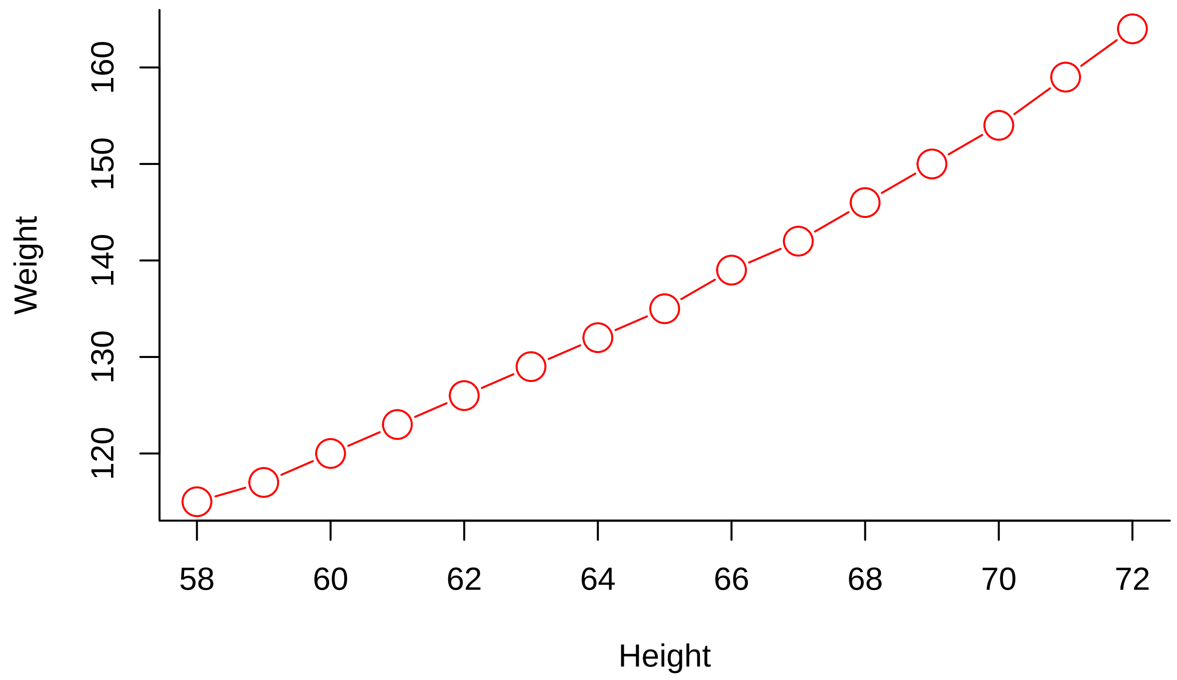
```r
plot(women$height, women$weight) # graph 'height' versus 'weight' for 'women'
```



```r
attach(women) # declare 'women' to be the default data set
plot(height, weight, type="l") # graph 'height' vs. 'weight', specifying the line type
```



```r
plot(height, weight, type="b", col="red", bty="l", cex=2, xlab="Height", ylab="Weight")
```



P4. Importing your own data

```
schoolkids <- data.frame(height=rnorm(100), weight=rnorm(100),
    sex=sample(1:2, 100, replace=TRUE))
schoolkids            # look at the entire data set
```

```
##           height      weight sex
## 1    1.373550191  1.56043778   2
## 2   -2.456549185 -1.04694714   2
## 3   -0.280226770  0.17671936   2
## 4    1.069165668 -1.14993008   1
## 5   -0.311485976  1.46264745   1
## 6   -1.483965502 -0.47925421   1
## 7    0.582622827  0.68018406   2
## 8    0.104726272 -0.50981456   2
## 9    0.587122677 -2.30341264   2
## 10  -0.416952090 -0.33159178   2
## 11   0.888205698 -1.52164247   2
## 12   0.734037223 -0.43763518   2
## 13   0.586340468 -0.11239831   2
## 14   1.146180575  0.54544165   2
## 15   1.210617019 -0.87579820   1
## 16  -0.141016009  1.23950745   1
## 17  -0.525659139 -2.54425146   1
## 18  -0.458618103 -0.58864015   2
## 19  -0.217963258  1.90548993   2
## 20  -0.623661454  1.15562389   1
## 21  -0.546671658  1.05767395   1
## 22  -0.760097983 -0.80309416   2
## 23   1.319733104 -1.47085564   2
## 24  -1.593172983  2.08153495   1
## 25  -0.398714579  0.57165663   1
## 26   1.203837587  1.01102006   1
## 27   0.914796213 -0.74126700   2
## 28  -0.238222835 -2.30941216   1
## 29   0.235176935  0.32448721   2
## 30   0.437150999  0.02214456   2
## 31  -0.015509547 -1.15218367   1
## 32   0.284625493 -0.86532194   1
## 33  -0.132268646 -0.69885379   1
## 34  -0.120129395  1.58544726   1
## 35  -0.160259596 -0.88458976   2
## 36   0.412020800 -1.41173669   2
## 37  -1.132690090 -0.32227283   1
## 38   0.749225513 -1.31716982   2
## 39   0.090543131 -1.10866124   2
## 40   0.242690021 -2.34012109   1
## 41   0.977726668 -0.59186545   1
## 42  -1.762395432 -0.27709218   2
## 43  -0.062787297  0.26726381   1
## 44   0.508539518  1.67630965   1
## 45   0.992984038 -0.37948880   2
## 46   0.607488144 -0.91004450   2
## 47  -0.539066420 -0.48087507   2
## 48   0.318205635  0.22125546   2
## 49  -1.670780550 -0.25773561   1
## 50  -0.525877188  0.38295669   2
## 51   0.553176673  0.78888787   1
## 52   1.192956045  2.71053056   2
## 53  -1.877713501 -0.60609451   1
## 54  -0.795399596 -1.35904188   2
## 55   0.808993171 -0.66620264   2
## 56   0.381959391 -0.87146943   2
## 57  -0.452931919  1.67404890   2
## 58  -0.745241239  0.88332454   1
## 59   0.680957899 -1.05128032   2
## 60   0.688252072 -0.65756981   2
## 61  -1.207388426  0.54774485   1
## 62   0.059964011 -0.39907338   1
## 63  -1.215349060  0.35043615   1
## 64  -0.412802698 -1.07292596   1
## 65   0.054465059  1.99124099   2
## 66  -0.010018266  0.62230229   2
## 67   0.181897977 -0.34254443   1
## 68  -0.111165784  1.11895393   2
## 69  -0.024682523 -2.37115400   1
## 70  -0.401754933 -1.07406781   1
## 71  -1.492241867 -0.12740511   1
## 72   0.520530983 -2.53635495   2
## 73  -0.115213040 -0.80896870   2
## 74  -1.653091341  2.58258438   1
## 75  -1.795709005  0.30460407   2
## 76  -0.249056913 -0.42415039   2
## 77  -1.726436480 -0.07318773   2
## 78   2.284780962 -0.48290243   2
## 79   2.140926632 -1.03408947   1
## 80   0.210192005  0.73682346   2
## 81   0.316986130 -0.63143722   2
## 82   0.147634379  0.18885931   1
## 83   0.977808456 -0.24432579   2
## 84   0.663000704 -0.46067922   2
## 85   2.199534605  0.71942670   2
## 86   0.879772484  0.94444244   2
## 87   0.370199374 -0.63190497   2
## 88  -1.436260967 -1.02063585   1
## 89  -0.602455824 -1.46253302   1
## 90   0.079523387  1.04614227   1
## 91  -0.059161071 -0.31313364   2
## 92  -0.004033891 -0.42160945   1
## 93  -0.612150040  1.18051116   2
## 94   0.447203820  0.68842131   1
## 95  -0.229669251 -0.20115752   2
## 96   0.088323272 -0.66382047   2
## 97   0.473066808  1.79790461   2
## 98   0.161634058  1.04528904   2
## 99   0.671403069  1.96179056   1
## 100 -0.678891905 -0.70330518   2
```

```
names(schoolkids)       # list the variables
```

```
## [1] "height" "weight" "sex"
```

```
attach(schoolkids)      # Make it the default data set
```

```
## The following objects are masked from women:
##
##     height, weight
```

```
summary(schoolkids)
```

```
##      height              weight              sex
##  Min.   :-2.456549   Min.   :-2.5443   Min.   :1.00
##  1st Qu.:-0.475378   1st Qu.:-0.8669   1st Qu.:1.00
##  Median : 0.057215   Median :-0.3371   Median :2.00
##  Mean   : 0.003289   Mean   :-0.1112   Mean   :1.58
##  3rd Qu.: 0.586536   3rd Qu.: 0.6962   3rd Qu.:2.00
##  Max.   : 2.284781   Max.   : 2.7105   Max.   :2.00
```

```
schoolkids$sex <- factor(sex)
# Deduce what each of these commands does:
schoolkids[2,4]
```

```
## NULL
```

```
schoolkids[,2]
```

```
##   [1]  1.56043778 -1.04694714  0.17671936 -1.14993008  1.46264745
##   [6] -0.47925421  0.68018406 -0.50981456 -2.30341264 -0.33159178
##  [11] -1.52164247 -0.43763518 -0.11239831  0.54544165 -0.87579820
##  [16]  1.23950745 -2.54425146 -0.58864015  1.90548993  1.15562389
##  [21]  1.05767395 -0.80309416 -1.47085564  2.08153495  0.57165663
##  [26]  1.01102006 -0.74126700 -2.30941216  0.32448721  0.02214456
##  [31] -1.15218367 -0.86532194 -0.69885379  1.58544726 -0.88458976
##  [36] -1.41173669 -0.32227283 -1.31716982 -1.10866124 -2.34012109
##  [41] -0.59186545 -0.27709218  0.26726381  1.67630965 -0.37948880
##  [46] -0.91004450 -0.48087507  0.22125546 -0.25773561  0.38295669
##  [51]  0.78888787  2.71053056 -0.60609451 -1.35904188 -0.66620264
##  [56] -0.87146943  1.67404890  0.88332454 -1.05128032 -0.65756981
##  [61]  0.54774485 -0.39907338  0.35043615 -1.07292596  1.99124099
##  [66]  0.62230229 -0.34254443  1.11895393 -2.37115400 -1.07406781
##  [71] -0.12740511 -2.53635495 -0.80896870  2.58258438  0.30460407
##  [76] -0.42415039 -0.07318773 -0.48290243 -1.03408947  0.73682346
##  [81] -0.63143722  0.18885931 -0.24432579 -0.46067922  0.71942670
##  [86]  0.94444244 -0.63190497 -1.02063585 -1.46253302  1.04614227
##  [91] -0.31313364 -0.42160945  1.18051116  0.68842131 -0.20115752
##  [96] -0.66382047  1.79790461  1.04528904  1.96179056 -0.70330518
```

```
schoolkids[2,]
```

```
##      height    weight sex
## 2 -2.456549 -1.046947   2
```

```
schoolkids[,1:3]
```

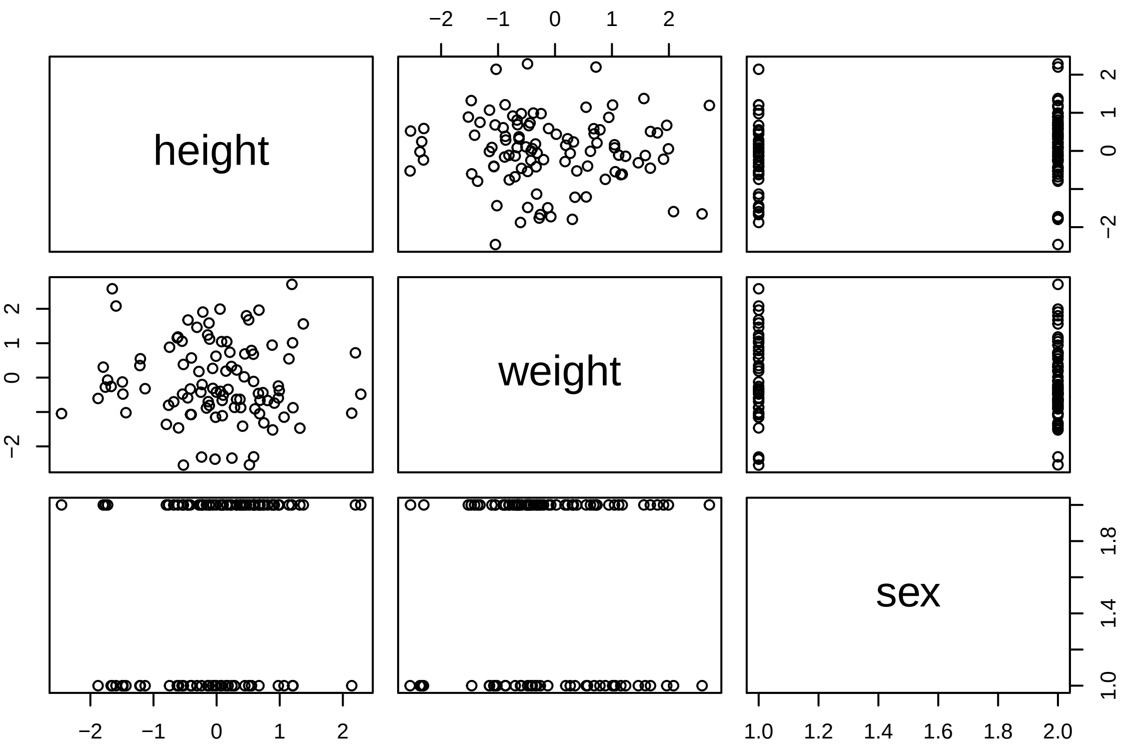
```
##           height      weight sex
## 1    1.373550191  1.56043778   2
## 2   -2.456549185 -1.04694714   2
## 3   -0.280226770  0.17671936   2
## 4    1.069165668 -1.14993008   1
## 5   -0.311485976  1.46264745   1
## 6   -1.483965502 -0.47925421   1
## 7    0.582622827  0.68018406   2
## 8    0.104726272 -0.50981456   2
## 9    0.587122677 -2.30341264   2
## 10  -0.416952090 -0.33159178   2
## 11   0.888205698 -1.52164247   2
## 12   0.734037223 -0.43763518   2
## 13   0.586340468 -0.11239831   2
## 14   1.146180575  0.54544165   2
## 15   1.210617019 -0.87579820   1
## 16  -0.141016009  1.23950745   1
## 17  -0.525659139 -2.54425146   1
## 18  -0.458618103 -0.58864015   2
## 19  -0.217963258  1.90548993   2
## 20  -0.623661454  1.15562389   1
## 21  -0.546671658  1.05767395   1
## 22  -0.760097983 -0.80309416   2
## 23   1.319733104 -1.47085564   2
## 24  -1.593172983  2.08153495   1
## 25  -0.398714579  0.57165663   1
## 26   1.203837587  1.01102006   1
## 27   0.914796213 -0.74126700   2
## 28  -0.238222835 -2.30941216   1
## 29   0.235176935  0.32448721   2
## 30   0.437150999  0.02214456   2
## 31  -0.015509547 -1.15218367   1
## 32   0.284625493 -0.86532194   1
## 33  -0.132268646 -0.69885379   1
## 34  -0.120129395  1.58544726   1
## 35  -0.160259596 -0.88458976   2
## 36   0.412020800 -1.41173669   2
## 37  -1.132690090 -0.32227283   1
## 38   0.749225513 -1.31716982   2
## 39   0.090543131 -1.10866124   2
## 40   0.242690021 -2.34012109   1
## 41   0.977726668 -0.59186545   1
## 42  -1.762395432 -0.27709218   2
## 43  -0.062787297  0.26726381   1
## 44   0.508539518  1.67630965   1
## 45   0.992984038 -0.37948880   2
## 46   0.607488144 -0.91004450   2
## 47  -0.539066420 -0.48087507   2
## 48   0.318205635  0.22125546   2
## 49  -1.670780550 -0.25773561   1
## 50  -0.525877188  0.38295669   2
## 51   0.553176673  0.78888787   1
## 52   1.192956045  2.71053056   2
## 53  -1.877713501 -0.60609451   1
## 54  -0.795399596 -1.35904188   2
## 55   0.808993171 -0.66620264   2
## 56   0.381959391 -0.87146943   2
## 57  -0.452931919  1.67404890   2
## 58  -0.745241239  0.88332454   1
## 59   0.680957899 -1.05128032   2
## 60   0.688252072 -0.65756981   2
## 61  -1.207388426  0.54774485   1
## 62   0.059964011 -0.39907338   1
## 63  -1.215349060  0.35043615   1
## 64  -0.412802698 -1.07292596   1
## 65   0.054465059  1.99124099   2
## 66  -0.010018266  0.62230229   2
## 67   0.181897977 -0.34254443   1
## 68  -0.111165784  1.11895393   2
## 69  -0.024682523 -2.37115400   1
## 70  -0.401754933 -1.07406781   1
## 71  -1.492241867 -0.12740511   1
## 72   0.520530983 -2.53635495   2
## 73  -0.115213040 -0.80896870   2
## 74  -1.653091341  2.58258438   1
## 75  -1.795709005  0.30460407   2
## 76  -0.249056913 -0.42415039   2
## 77  -1.726436480 -0.07318773   2
## 78   2.284780962 -0.48290243   2
## 79   2.140926632 -1.03408947   1
## 80   0.210192005  0.73682346   2
## 81   0.316986130 -0.63143722   2
## 82   0.147634379  0.18885931   1
## 83   0.977808456 -0.24432579   2
## 84   0.663000704 -0.46067922   2
## 85   2.199534605  0.71942670   2
## 86   0.879772484  0.94444244   2
## 87   0.370199374 -0.63190497   2
## 88  -1.436260967 -1.02063585   1
## 89  -0.602455824 -1.46253302   1
## 90   0.079523387  1.04614227   1
## 91  -0.059161071 -0.31313364   2
## 92  -0.004033891 -0.42160945   1
## 93  -0.612150040  1.18051116   2
## 94   0.447203820  0.68842131   1
## 95  -0.229669251 -0.20115752   2
## 96   0.088323272 -0.66382047   2
## 97   0.473066808  1.79790461   2
## 98   0.161634058  1.04528904   2
## 99   0.671403069  1.96179056   1
## 100 -0.678891905 -0.70330518   2
```

```
schoolkids[height>70,]
```

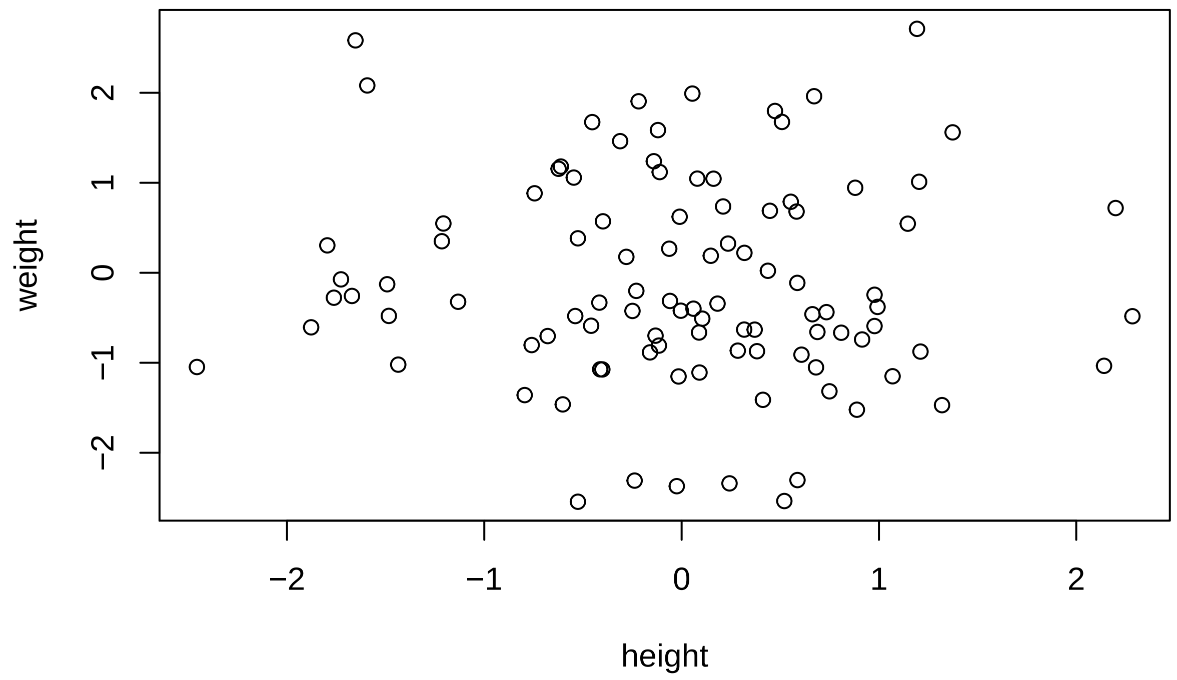
```
## [1] height weight sex
## <0 rows> (or 0-length row.names)
```

P5. Graphing data

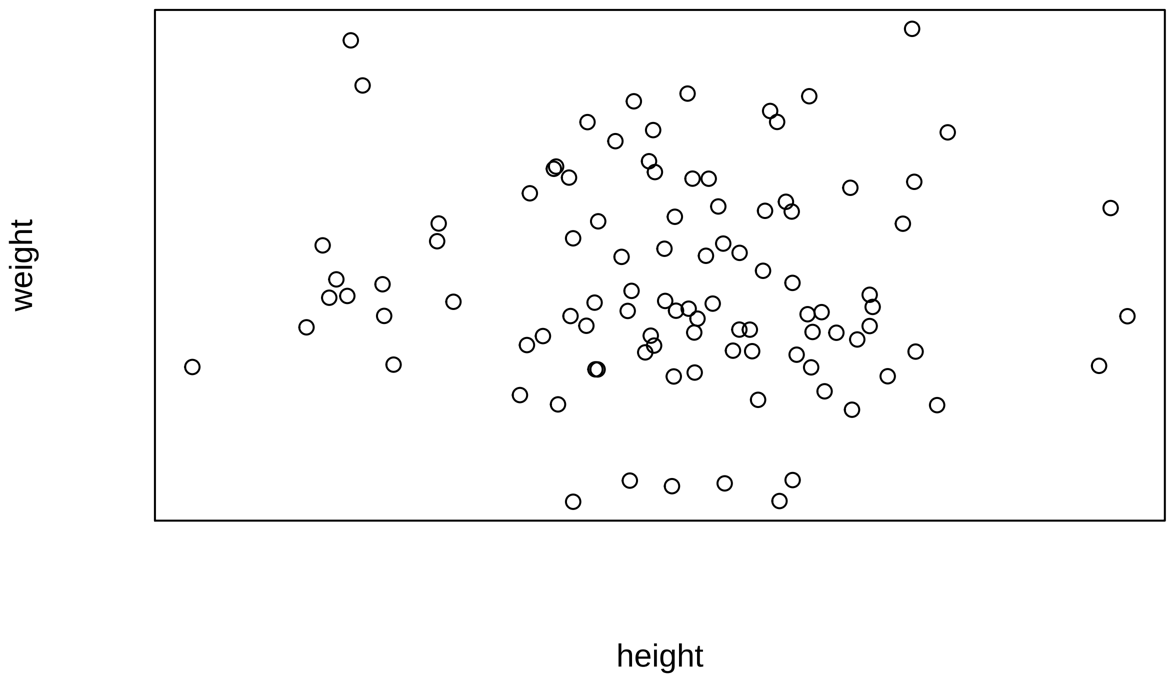
```
pairs(schoolkids)  # make pair-wise plots with columns in 'schoolkids'
```



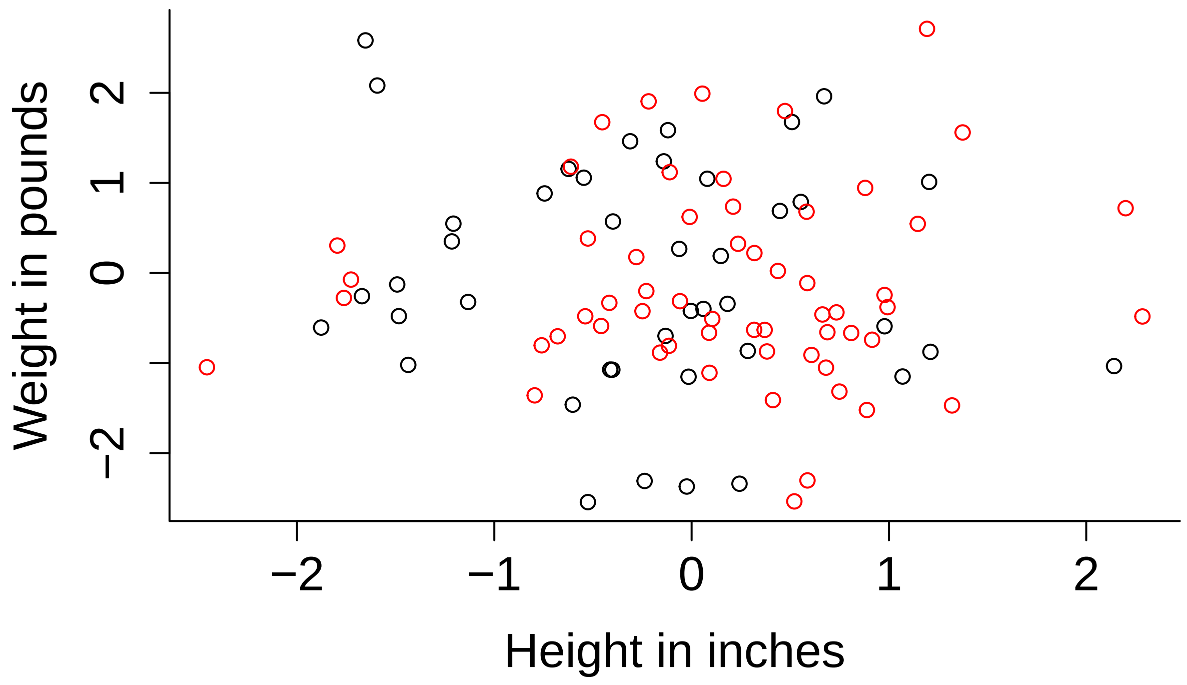
```
plot(height, weight)          # Plot height (x-axis) against weight (y-axis)
```



```
plot(height,weight,xaxt="n",yaxt="n")    # set up a blank graph with axes
```



```r
plot(height,weight,type="n",
 bty="l",  # Remove the surrounding box
 xlab="Height in inches",  # set a label for the horizontal axis
 ylab="Weight in pounds",  # set a label for the vertical axis
 cex.axis=1.5,  # multiply the axis size by 1.5
 cex.lab=1.5)  # multiply the axis label size by 1.5
points(height[sex==1], weight[sex==1])  # add points for the 1st sex
points(height[sex==2],weight[sex==2],col="red")  # add red points for the 2nd sex
legend(52,160,c("Male","Female"),col=c("black","red"),pch=c(1,1) ) # add  a legend
```



P6. Simulating dice throws

```
die <- c(1:6)                 # define the possible outcomes for a 6 sided die
die                           # have a look at your 'die'
```

```
## [1] 1 2 3 4 5 6
```

```
sample(die, 1, replace = TRUE)  # take a random 'throw of the of 'die'
```

```
## [1] 3
```

```
sample(die, 10, replace = TRUE)     # observe 10 simulated dice throws
```

```
##  [1] 4 2 5 2 2 6 4 2 4 4
```

```
table(sample(die, size = 10000, replace = TRUE))
```

```
## 
##    1    2    3    4    5    6 
## 1683 1676 1599 1682 1693 1667
```

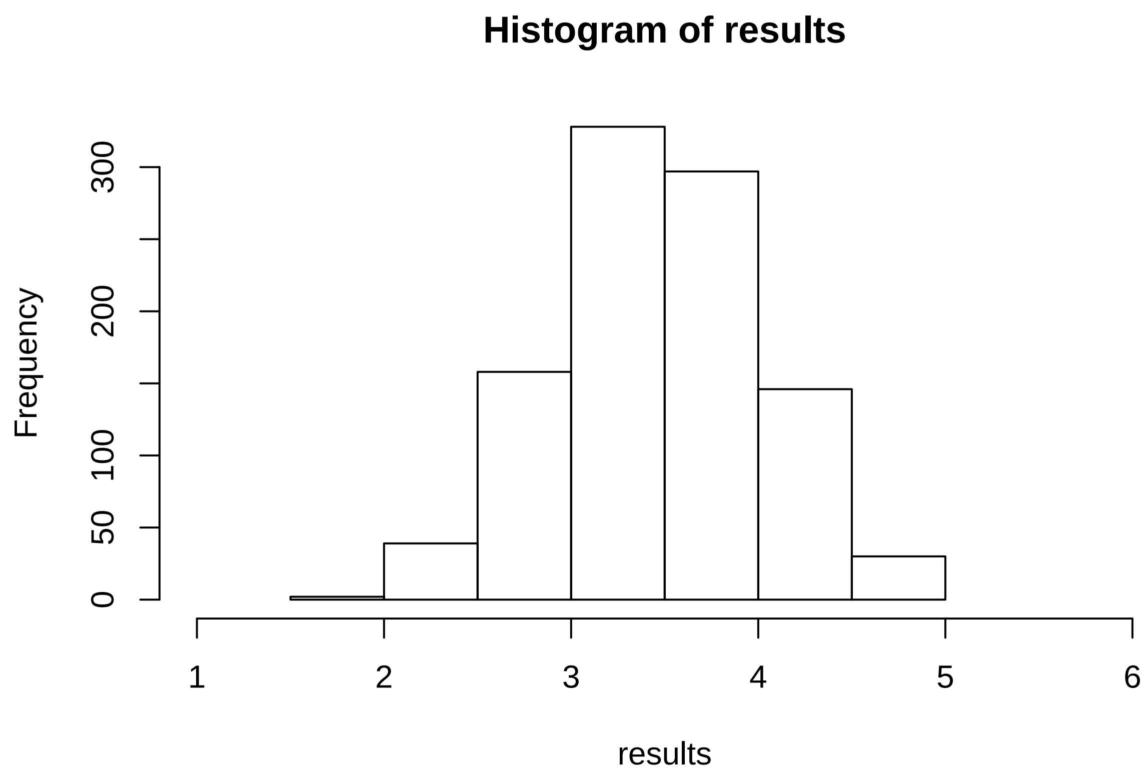
```
# construct a frequency table for 10,000 dice throws
mean(sample(die, 10, replace = TRUE))   # observe the mean (average) of ten throws
```

```
## [1] 3.7
```

```
# Repeat the previous command several times
# Observe that the sample mean is "scattered around" a "true mean" of 3.5
# Try this again, but with sample size increased to 100
# The "scatter around the true mean" decreases!
```

P7. Simulating data within a 'for loop'

```
# In order to visualize the "scatter around a true mean",
# the following code will generate and store 1000 such means
results <- vector()              # set up space to hold results
length(results) <- 1000          # size it to hold exactly 1000 means
for (i in 1:1000) { # set up a 'for loop' that will iterate 1000 times
    # save the mean at location 'i' of 'results'
    results[i] <- mean(sample(die, 10, replace = TRUE))
}                # repeat 'for loop' at the next value of 'i'
hist(results, xlim = c(1,6))    # A histogram of the 1000 means
```



```
# Repeat, but with sample size increased to 100
# The "scatter around the true mean" decreases!
```