Least Squares Estimation with R: $\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$

```
> trees[1:4,] # First 4 rows, all columns
  Girth Height Volume
1   8.3     70   10.3
2   8.6     65   10.3
3   8.8     63   10.2
4  10.5     72   16.4
> n = dim(trees)[1]; n
[1] 31
> attach(trees)  # Makes variable names available
> int = numeric(n)+1  # Vector of ones, length n
> X = cbind(int, Girth, Height); y = Volume
> X
      int Girth Height
 [1,]   1   8.3     70
 [2,]   1   8.6     65
 [3,]   1   8.8     63
 [4,]   1  10.5     72
 [5,]   1  10.7     81
 [6,]   1  10.8     83
 [7,]   1  11.0     66
 [8,]   1  11.0     75
 [9,]   1  11.1     80
[10,]   1  11.2     75
[11,]   1  11.3     79
[12,]   1  11.4     76
[13,]   1  11.4     76
[14,]   1  11.7     69
[15,]   1  12.0     75
[16,]   1  12.9     74
[17,]   1  12.9     85
[18,]   1  13.3     86
[19,]   1  13.7     71
[20,]   1  13.8     64
[21,]   1  14.0     78
[22,]   1  14.2     80
[23,]   1  14.5     74
[24,]   1  16.0     72
[25,]   1  16.3     77
[26,]   1  17.3     81
```

```
[27,]   1  17.5     82
[28,]   1  17.9     80
[29,]   1  18.0     80
[30,]   1  18.0     80
[31,]   1  20.6     87
```

```
> XpX =  t(X) %*% X; XpX
```

```
          int    Girth   Height
int      31.0   410.70   2356.0
Girth   410.7  5736.55  31524.7
Height 2356.0 31524.70 180274.0
```

```
> Xpy = t(X) %*% y; Xpy
```

```
           [,1]
int      935.30
Girth  13887.86
Height 72962.60
```

```
> betahat = solve(XpX) %*% Xpy
> betahat
              [,1]
int    -57.9876589
Girth    4.7081605
Height   0.3392512
```

```
> # Predict volume for a tree 12 inches in diameter, 80 feet tall
> betahat[1] + betahat[2]*12 + betahat[3]*80
```

```
[1] 25.65037
```

```
> # R does not actually calculate X'X-inverse. It solves the system of
> # linear equations X'X beta = X'y numerically, like this:
> solve(XpX,Xpy)
```

```
              [,1]
int    -57.9876589
Girth    4.7081605
Height   0.3392512
```

```
> # It's better (not just more convenient) to let R do the calculation
> treefit = lm(Volume ~ Girth+Height) # Produces a linked list
> treefit$coefficients

(Intercept)       Girth      Height
-57.9876589   4.7081605   0.3392512

> sum(Volume) # Sum of y
[1] 935.3
> sum(treefit$fit) # Sum of y-hat
[1] 935.3
> sum(treefit$residuals) # Sum of epsilon-hat
[1] 4.662937e-15



> # Try a model with no intercept: y = beta1*x1 + beta2*x2 + epsilon
> treefit2 = lm(Volume ~ 0+Girth+Height)

> treefit2$coefficients
     Girth     Height
 5.0440083 -0.4773192
> sum(treefit2$residuals)
[1] -11.71008

> # Prediction made easy
> newdata = data.frame(Girth=12,Height=80) # Creating a data frame
> newdata
  Girth Height
1    12     80
> predict(treefit2,newdata) # With an intercept, got 25.65037

       1
22.34256

> # WHICH PREDICTION DO YOU LIKE MORE?
```

Reading from external files: see `help(read.table)`

By default, R expects a plain test data file to look like this:

```
  Subject Item Treatment ReactionTime
1      s1   w1      Long          466
2      s1   w2      Long          520
3      s1   w3      Long          502
```

```
> # Read a file in your working directory
> potatoes = read.table("potato.data.txt"); head(potatoes)
  BACTERIA TEMP OXYGEN ROT
1        1    1      1   7
2        1    1      1   7
3        1    1      1   9
4        1    1      2   0
5        1    1      2   0
6        1    1      2   0
```

```
> # Read data in an Excel spreadsheet (in the working directory)
> # install.packages("xlsx", dependencies=TRUE) # Only need to do this once
> library(xlsx) # Load the package
Loading required package: rJava
Loading required package: xlsxjars
```

```
> sleep = read.xlsx("sleep1.xlsx", sheetIndex=1) # Student's sleep data
> head(sleep)
   Patient Drug1 Drug2
1        1   0.7   1.9
2        2  -1.6   0.8
3        3  -0.2   1.1
4        4  -1.2   0.1
5        5  -0.1  -0.1
6        6   3.4   4.4
```

```
# Read a data file online
> cars =
read.table("http://www.utstat.toronto.edu/~brunner/data/legal/mcars4.data")
```
