Lab 2.4: Introduction to Linear Regression II

We will work with data on the fat and protein content of items on the Burger King menu.
In RStudio, Environment in Quadrant I, goto Import Data, and paste in the URL
http://statland.org/AP/R/BKmenu.txt

Console (Quadrant III) will show the following:

```
> BKmenu <-
read.delim("C:\\DOCUME~1\\Owner\\LOCALS~1\\Temp\\RtmpGMePYJ\\datab5c608d1044")
```

Click on BKmenu, in Quadranr I, and data appears in Quardant II.

	item	Total.Fat	Protein
1	Whopper	39.0	29
2	Whopper w/ Cheese	47.0	34
3	Double Whopper	57.0	48
4	Double Whopper w/ Cheese	65.0	53
5	Hamburger	14.0	18
6	Cheeseburger	18.0	20
7	Double Hamburger	26.0	31
8	Double Cheeseburger	34.0	35
9	Double Cheeseburger w/ Bacon	37.0	38
10	Veggie Burger	10.0	14
11	BK Big Fish	38.0	24
12	BK Broiler Chicken	25.0	30
13	Chicken Tenders Sandwich	27.0	14
14	Chicken Tenders (4pc)	9.0	11
15	Fries (med)	18.0	4
16	Onion rings (med)	16.0	4
17	Jalapeno poppers (4pc)	13.0	7
18	Mozzarella Sticks (4pc)	16.0	12
19	Apple Pie	14.0	2
20	Croissan'wich w/Sausage, Egg&Cheese	36.0	19
21	Biscuit	15.0	6
22	Biscuit w/Sausage, Egg&Cheese	46.0	20
23	Biscuit w/ Sausage	35.0	13
24	French Toast Stix (5)	20.0	6
25	Cini-minis (4)	23.0	6
26	Hash Brown Rounds (small)	15.0	2
27	Vanilla Shake (med)	8.0	12
28	Chocolate Shake (Med w/ Syrup)	8.0	13
29	Strawberry Shake (Med)	8.0	12
30	Coke (med)	0.0	0
31	Tropicana OJ	0.0	2
32	1% Milk	2.5	8

Here Protein is the dependent variable and Total.Fat is the independent variable.

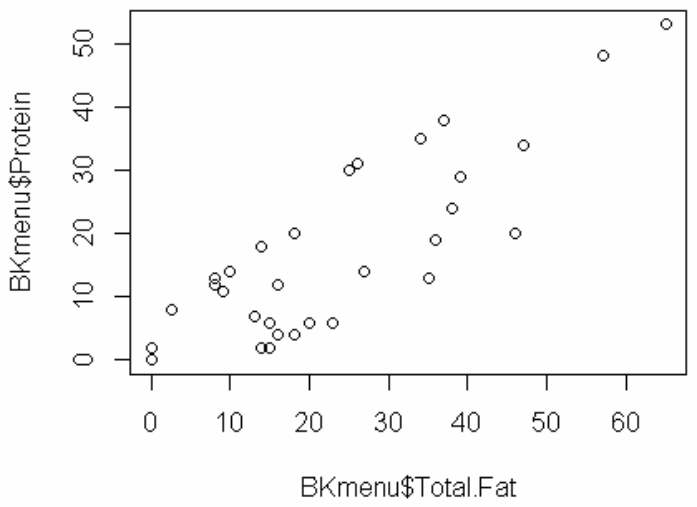
In Console type

```
BK <- lm(Protein ~ Total.Fat, data = BKmenu)
```

This tells you that Protein is dependent on Total Fat

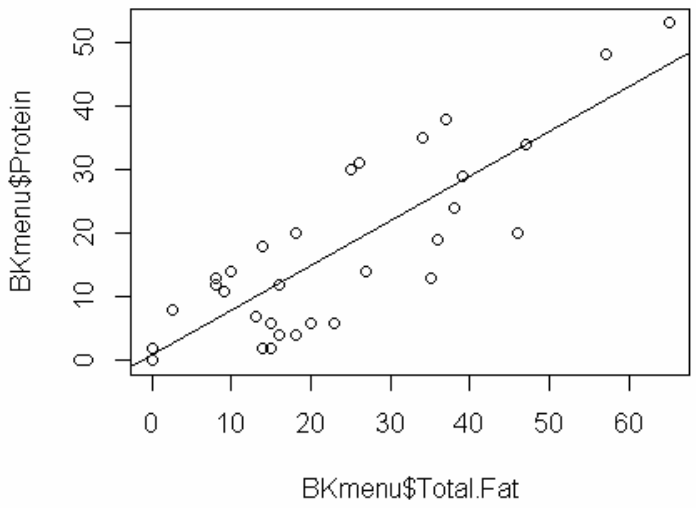
```
cor(BKmenu$Total.Fat, BKmenu$Protein)
[1] 0.8270104
```

```
plot(x=BKmenu$Total.Fat, y=BKmenu$Protein)
```



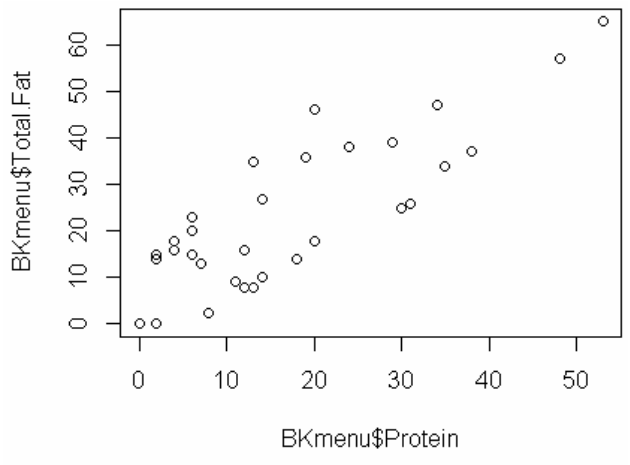

```
summary(BK)
Call:
lm(formula = Protein ~ Total.Fat, data = BKmenu)

Residuals:
     Min       1Q   Median       3Q      Max
-13.1207  -7.3684   0.4798   6.4840  11.8824

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.9136     2.4443   0.374    0.711
Total.Fat     0.7002     0.0869   8.057  5.4e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 7.883 on 30 degrees of freedom
Multiple R-squared:  0.6839,   Adjusted R-squared:  0.6734
F-statistic: 64.92 on 1 and 30 DF,  p-value: 5.402e-09
```

```
abline(BK)
```



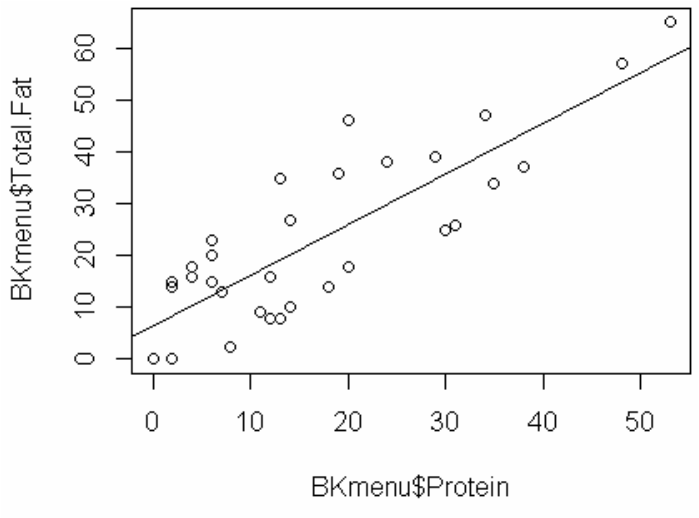
We could do it differently; we could make Total.Fat the dependent variable and Protein the independent or explanatory variable:

```
BK <- lm(Total.Fat ~ Protein, data = BKmenu)
```



This tells you that Total Fat is dependent on Protein

```
plot(x=BKmenu$Protein, y=BKmenu$Total.Fat)

cor(BKmenu$Protein, BKmenu$Total.Fat)
[1] 0.8270104

summary(BK)
Call:
lm(formula = Total.Fat ~ Protein, data = BKmenu)

Residuals:
    Min      1Q  Median      3Q     Max
-11.726  -8.772   1.239   7.029  20.052

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.4113     2.6466   2.423   0.0217 *
Protein       0.9769     0.1212   8.057  5.4e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.311 on 30 degrees of freedom
Multiple R-squared:  0.6839,   Adjusted R-squared:  0.6734
F-statistic: 64.92 on 1 and 30 DF,  p-value: 5.402e-09

abline(BK)
```

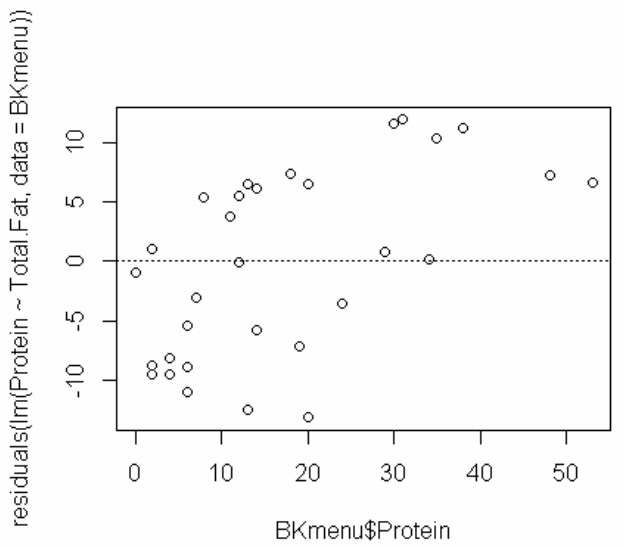


You can also compute residuals

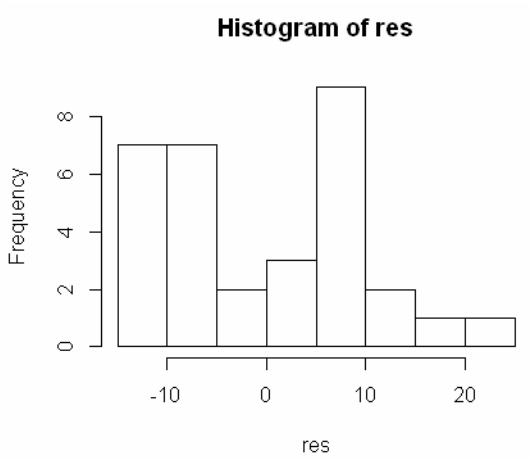
```
residuals(lm(Total.Fat ~ Protein, data = BKmenu))
```

```
         1          2          3          4          5
 4.2599876  7.3757304  3.6998103  6.8155532 -9.9946466
         6          7          8          9         10
-7.9483495 -10.6937153 -6.6011210 -6.5316753 -10.0872409
        11         12         13         14         15
 8.1442448 -10.7168638  6.9127591 -8.1566866  7.6812735
        16         17         18         19         20
 5.6812735  -0.2492808 -2.1335380  5.6349763 11.0285020
        21         22         23         24         25
 2.7275706 20.0516505 15.8896106  7.7275706 10.7275706
        26         27         28         29         30
 6.6349763 -10.1335380 -11.1103894 -10.1335380 -6.4113208
        31         32
-8.3650237 -11.7261323
```

```
plot(x=BKmenu$Protein,y=residuals(lm(Protein ~ Total.Fat, data = BKmenu)))
abline(h = 0, lty = 3) # adds a horizontal dashed line at y = 0
```



These look reasonably random but not clumped around zero. Instead there seems to be a group of residuals around 10 and another around -10.



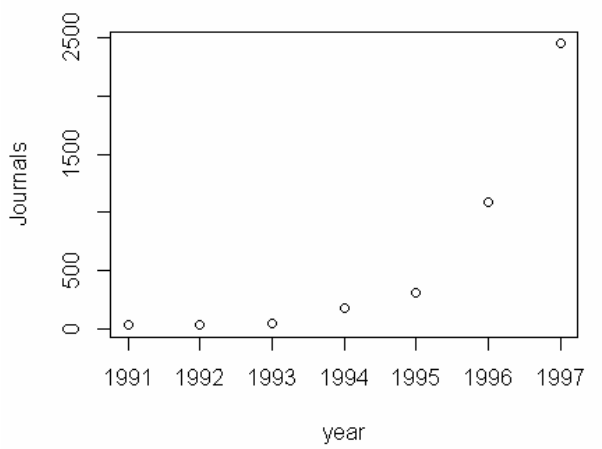
If you plan to do much with the residuals, you may wish to store them in a variable for further work. For example, here they are stored in a variable res and then a histogram is made.

```
res = residuals(lm(Total.Fat ~ Protein, data = BKmenu))
hist(res)
```

Transformations in R

For this example we will use data on the number of electronic academic journals over a seven-year period. Note the shortcut for entering consecutive years. The journal counts were cut and pasted from another statistical software package after invoking the scan function (and hitting Return). There are so few you could just type them in.

```
year = c(1991:1997)
year
[1] 1991 1992 1993 1994 1995 1996 1997

Journals <- scan()
1: 27
2: 36
3: 45
4: 181
5: 306
6: 1093
7: 2459
8:
Read 7 items

plot(year,Journals)
```



Not surprisingly, the number of electronic journals really took off during this period. Sometimes "exponential growth" is used to describe any kind of rapid growth, but technically it refers to a specific mathematical pattern. If we have true exponential growth, then plotting the logarithms of the growing variable versus time should give a straight line. First take the logarithms, then make the plot.

```
logJ=log(Journals)
plot(year,logJ)
```

The original graph shows strong curvature. The logarithms of the journal counts plot as much more linear versus year. We might say that the growth is approximately exponential, especially after the first year.

It might be interesting to see the effect of the transformation on the journal counts considered by themselves.

```
hist(Journals)
hist(logJ)
```

Here the transformation makes the data much less skewed.

Logarithms are a common transformation but certainly not the only one. We can do simple arithmetic transformations at the command line.

Lab 2.4: On Your Own

***Name*_____ *Score*_____**

1. Choose another traditional variable from mlb11 that you think might be a good predictor of runs. Produce a scatterplot of the two variables and fit a linear model. At a glance, does there seem to be a linear relationship?

2. How does this relationship compare to the relationship between runs and at_bats? Use the R^2 values from the two model summaries to compare. Does your variable seem to predict runs better than at_bats? How can you tell?

3. Now that you can summarize the linear relationship between two variables, investigate the relationships between runs and each of the other five traditional variables. Which variable best predicts runs? Support your conclusion using the graphical and numerical methods we've discussed (for the sake of conciseness, only include output for the best variable, not all five).

4. Now examine the three newer variables. These are the statistics used by the author of Moneyball to predict a teams success. In general, are they more or less effective at predicting runs that the old variables? Explain using appropriate graphical and numerical evidence. Of all ten variables we've analyzed, which seems to be the best predictor of runs? Using the limited (or not so limited) information you know about these baseball statistics, does your result make sense?

5. Give the model diagnostics for the regression model with the variable you decided was the best predictor for runs.