# Lab 3

MTH 3220

10 October 2017

## Blood Coagulation Times Data Set

The table below shows coagulation times (seconds) for samples of blood drawn from 24 animals receiving four different diets, A, B, C, and D. The diets were randomly assigned to the animals, and the blood samples were taken and tested in random order.

| Diet (Treatment) | | | |
|---|---|---|---|
| A | B | C | D |
| 62 | 63 | 68 | 56 |
| 60 | 67 | 66 | 62 |
| 63 | 71 | 71 | 60 |
| 59 | 64 | 67 | 61 |
| 59 | 65 | 68 | 63 |
| 63 | 66 | 68 | 64 |
| 62 | 64 | 66 | 63 |
| 61 | 67 | 68 | 59 |

The data are in the file **blood.txt** *stacked* in the order in which the blood samples were drawn.

1. Use `read.table()` (with `header = TRUE`) to read the data from the **blood.txt** file into an R *data frame* called, say, `mydata`.

2. The square brackets operator `[ ]` can be used to extract values from a *data frame*. For example, to extract the value from the 3rd row and 1st column of `mydata` run:

   ```
   mydata[3, 1]
   ```

   To extract the entire 3rd row run:

```r
mydata[3, ]
```

To extract the entire 1st column run:

```r
mydata[, 1]
```

If we want to refer to a variable in a *data frame* by name, we need to use the dollar sign operator `$`:

```r
mydata$Diet
```

A *logical vector* before the comma will extract the rows of the *data frame* for which the *logical vector* is `TRUE`. Run the following to see an example of a logical vector (this checks if the `Diet` column of `mydata` is equal to `A` or not):

```r
mydata$Diet == "A"
```

so, to extract the Diet A treatment group from `mydata`, run:

```r
Adata <- mydata[mydata$Diet == "A", ]
Adata
```

Create four data sets, `Adata`, `Bdata`, `Cdata`, and `Ddata` corresponding to the four treatment groups.

3. Use `mean()`, the data sets from Step 5, and the square brackets to compute the four treatment group means $\bar{Y}_{1\cdot}$, $\bar{Y}_{2\cdot}$, $\bar{Y}_{3\cdot}$, and $\bar{Y}_{4\cdot}$. Assign these four means to objects titled `ybarA`, `ybarB`, `ybarC`, and `ybarD`, respectively, e.g.

```r
ybarA <- mean(Adata[, "CoagTime"])
```

4. Use the function `c()` to combine your four sample means into a vector and assign the vector to an object called `ybars`. Give this vector names and then sort the resulting object by running the following:

```r
names(ybars) <- c("ybarA", "ybarB", "ybarC", "ybarD")
sort(ybars)
```

Note that all of this can also be done just by sorting the output of a `tapply()` call, but now you're more familiar with how to do data selection!

5. Let $\mu_A$ be the average coagulation time for diet $A$ and label the other three means similarly. We want to test the hypotheses

$$\begin{aligned} H_0 : &\quad \mu_A = \mu_B = \mu_C = \mu_D \\ H_a : &\quad \text{not all } \mu_i\text{'s in } H_0 \text{ are equal.} \end{aligned}$$

   The function `aov()`, with arguments `formula=` (a *formula*) and `data=` (a *data frame*), will carry out a one-factor ANOVA without requiring you to attach the *data frame*. Use:

```
myanova <- aov(CoagTime ~ Diet, data = mydata)
```

   to carry out the ANOVA and save the results in an object `myanova`.

6. Now run the following to see the ANOVA table:

```
summary(myanova)
```

   *Before we do the hypothesis test, in the following three problems we will check whether the assumptions of normality and homoskedasticity seem plausible. Recall that you can get all plots in a single .pdf by running* `pdf("filename.pdf")` *before you make the plots and* `dev.off()` *after you make the plots.*

7. Make a boxplot of the coagulation times for each of the diets by running the following:

```
boxplot(CoagTime ~ Diet, data = mydata)
```

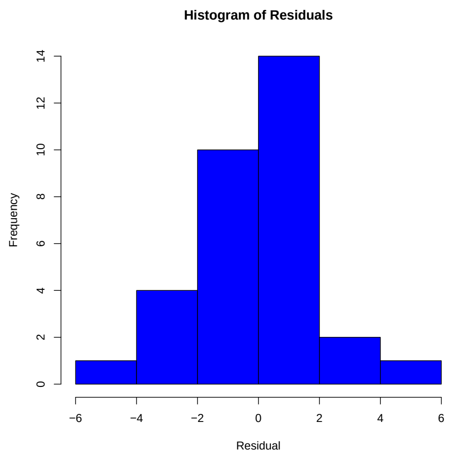
8. Note that the object `myanova` is a so-called *list* (run `is.list(myanova)`). A *list* is like a *vector*, but the elements of a *list* aren't necessarily numbers. They can be any R objects. Run:

```
names(myanova)
```

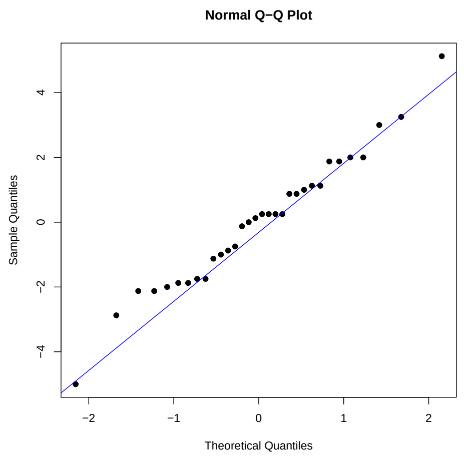
   to see the names of the objects stored in `myanova`.

   The operator `$` is used to extract an object from a *list* of objects. For example, `myanova$residuals` will extract the residuals.

Make a histogram of the residuals (using `hist()`, with `xlab=` and `main=` for an $x$ axis label and title). Your plot should look similar to this one:



9. Now make a normal probability plot of the residuals (use `qqnorm()` and add a line using `qqline()`). Your plot should look similar to this one:



10. Now carry out Tukey's multiple comparison procedure to decide which treatments differ from each other: pass your *aov* object from Step 5 to the `TukeyHSD()` function.