Analysis of Variance Report 1

Melinda Higgins

October 21, 2016

R Exercises - One-Way Analysis of Variance

The following exercise comes from R Exercises at http://r-exercises.com/, which is a site created by Research for Decisions, a Dutch research/consulting firm. They provide many exercises for learning R and practicing R coding skills. They also provide a large list of courses available for learning R - both free and paid - with detailed summaries of each option at http://r-exercises.com/r-courses/.

Assumptions

For one way ANOVA results to be valid there are several assumptions that need to be satisfied:

1. The dependent variable is required to be continuous *(not ordinal, not continuous)*
2. The independent variable is required to be categorical with two or more categories *(these could also be ordinal levels being treated as non-sequential categories)*.
3. The dependent and independent variables have values for each row of data *(no missing data - any rows or subjects with missing data for the independent or dependent variables are removed before analysis)*.
4. Observations in each group are independent *(subjects cannot have data in more than 1 category - this cannot be used for dependent "repeated measures")*.
5. The dependent variable is approximately normally distributed in each group *(ideally at least symmetric, not skewed, no significant outliers)*.
6. There is approximate equality of variance in all the groups *(we'll see how to test for this below)*.
7. We should not have any *"significant"* outliers *(also tested below)*.

When the data shows non-normality, unequal variance or presence of outliers, the data can be "transformed" *(such as square-root or log transformations to help skewness)* or a non-parametric test like Kruskal-Wallis can be used. It is good to note Kruskal-Wallis does not require normality of data but still requires equal variance in your groups. *[Note: There are transformation alternatives for handling unequal variances which will not be covered in this exercise.]*

Load Data

For this exercise, the data come from patients having stomach, colon, ovary, brochus, or breast cancer. The objective of the study was to identify if the number of days a patient survived was influenced by the organ affected. Our dependent variable is Survival measured in days. Our independent variable is Organ.

The data is available at http://lib.stat.cmu.edu/DASL/Datafiles/CancerSurvival.html. The dataset we're going to upload is the `cancer-survival.csv` file.

Since this dataset is a CSV (comma-delimited) formatted file, we can use the r function `read.csv()` to read in the dataset. This function returns a `data.frame` which we "assign" to the object `cancer.survival`.

```
cancer.survival <- read.csv("cancer-survival.csv", header = TRUE)
```

Look at data

Let's "look" at the data. We can use the `head()` function to look at the top 5-6 rows of the dataset.

```
head(cancer.survival)
```

```
##   Survival   Organ
## 1      124 Stomach
## 2       42 Stomach
## 3       25 Stomach
## 4       45 Stomach
## 5      412 Stomach
## 6       51 Stomach
```

We can clean up the looks for this output by using the `kable()` function from the `knitr` package to put the output into a table.

```
knitr::kable(head(cancer.survival))
```

Survival	Organ
124	Stomach
42	Stomach
25	Stomach
45	Stomach
412	Stomach
51	Stomach

We can also look at the "structure" of the data using the `str()` function.

```
str(cancer.survival)
```

```
## 'data.frame':    64 obs. of  2 variables:
##  $ Survival: int  124 42 25 45 412 51 1112 46 103 876 ...
##  $ Organ   : Factor w/ 5 levels "Breast","Bronchus",..: 5 5 5 5 5 5 5 5 5 5 ...
```

Select data and get stats

To select one of the columns in the dataset we can use the `$` selector to "select" by the variable name. You can see the column names using the `names()` function.

```
names(cancer.survival)
```

```
## [1] "Survival" "Organ"
```

For example, we can find the mean survival time using the `mean()` function and selecting the 1st column which is `Survival` in the data frame `cancer.survival` using the followng code `mean(cancer.survival$Survival)`.

```
mean(cancer.survival$Survival)
```

```
## [1] 558.625
```

Another way to select a data in a data frame is to use the row and column indices. To get the data element located at row 5 in column 1 we type `cancer.survival[5,1]`. The value should be 412.

```
cancer.survival[5,1]
```

```
## [1] 412
```

To select a whole column, you leave the row index empty and enter the column number as `cancer.survival[,1]`. Let's do a `summary()` of the 1st column of this dataset.

```r
summary(cancer.survival[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    20.0   102.5   265.5   558.6   721.0  3808.0
```

Since the 1st column `Survival` is a continuous variable, the `summary()` function gives us the min, max, median, 1st and 3rd quartiles. Let's see what we get for the 2nd column, which is the list of Organs - it is a non-numeric text categorical variable.

```r
summary(cancer.survival[,2])
```

```
##   Breast Bronchus    Colon    Ovary  Stomach
##       11       17       17        6       13
```

We basically got the list of the Organs and the count (or frequency) of how many subjects survival times (i.e. number of data points) we have for each Organ. We can use the `summary()` command for the dataset as a whole and get the summary stats for both variables at the same time. If we use the `knitr::kable()` function around this summary, we get a nice summary stats table. Let's also add a caption.

```r
knitr::kable(summary(cancer.survival),
             caption = "Table of Summary Statistics for 2 Variables in Dataset")
```

Table 2: Table of Summary Statistics for 2 Variables in Dataset

Survival	Organ
Min. : 20.0	Breast :11
1st Qu.: 102.5	Bronchus:17
Median : 265.5	Colon :17
Mean : 558.6	Ovary : 6
3rd Qu.: 721.0	Stomach :13
Max. :3808.0	NA

Clean up your code with "pipes" (%>%) - brief introduction to dplyr package

Our code is starting to get a little long, let's learn how to use "pipes" in R with the `%>%` command. This is available from the `dplyr` package which utilizes this pipe command originally from the `magrittr` package. The idea behind this pipe command is that you place the commands in the order in which you use them. For example, to do the summary command above what we're really doing is saying get the dataset `cancer.survival` and then run `summary()` on it. This code will do that. First we have to install and load the `dplyr` package.

```r
# install.packages("dplyr")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
cancer.survival %>%
  summary()
```

```
##     Survival           Organ
##  Min.   :  20.0   Breast  :11
##  1st Qu.: 102.5   Bronchus:17
##  Median : 265.5   Colon   :17
##  Mean   : 558.6   Ovary   : 6
##  3rd Qu.: 721.0   Stomach :13
##  Max.   :3808.0
```

. . . and then to get that output into the `knitr::kable()` function, you simply add it as the next command in the sequence.

```
cancer.survival %>%
  summary() %>%
  knitr::kable(caption = "Table of Summary Statistics for 2 Variables in Dataset")
```

Table 3: Table of Summary Statistics for 2 Variables in Dataset

Survival	Organ
Min. : 20.0	Breast :11
1st Qu.: 102.5	Bronchus:17
Median : 265.5	Colon :17
Mean : 558.6	Ovary : 6
3rd Qu.: 721.0	Stomach :13
Max. :3808.0	NA

Let's try some inline code

Suppose I want to write a sentence providing the mean and standard deviation of the survival times seen across this dataset. We can use the functions `mean()` and `sd()` and use them "inline" - see example below.

The average survival time was 558.625 days with a standard deviation of 776.4786713 days.

Another way to do this is to assign the mean and standard deviation to objects and then just call those objects. This simplifies the inline code commands and can help in cleaning up your formatting. We can also use the `round()` function with the `digits=` option to set the number of digits to 1 or 2. Let's create 2 objects one for the mean `mnsurv` and one for the standard deviation `sdsurv` and then use these to re-write the sentence. We'll also use the `round()` function and set the number of digits (after the decimal place) to 2.

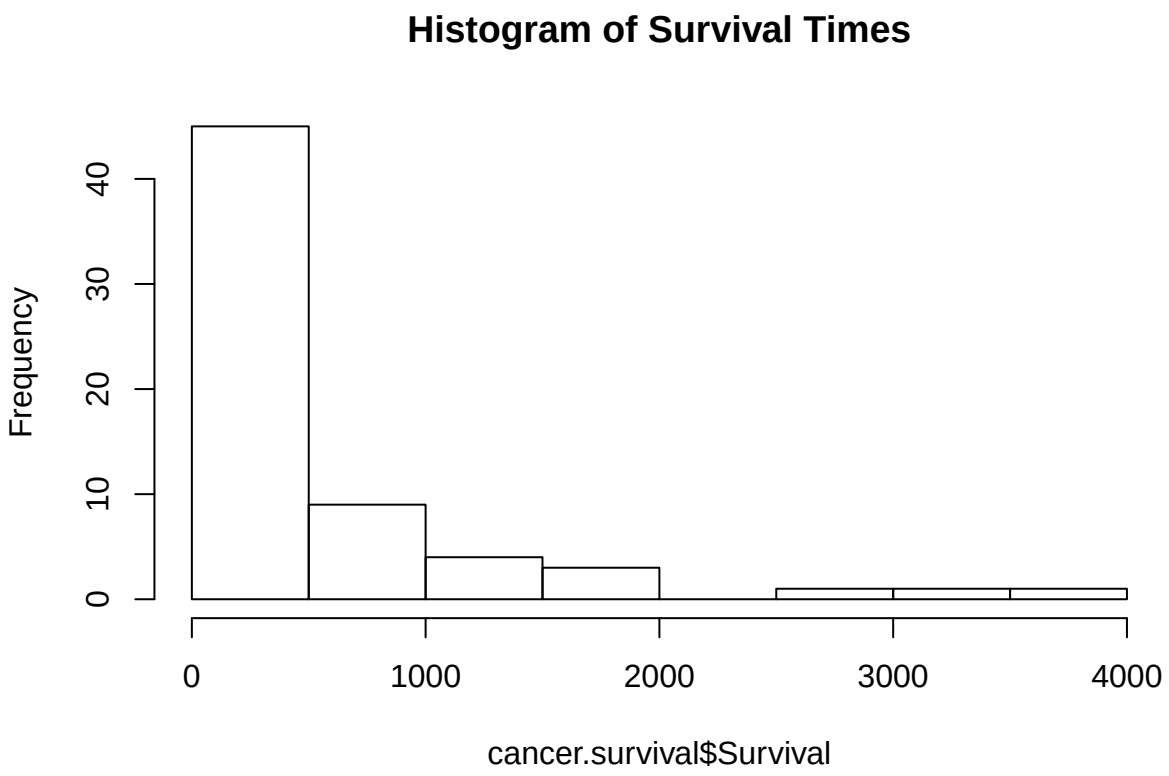
```
mnsurv <- mean(cancer.survival$Survival)
sdsurv <- sd(cancer.survival$Survival)
```

The average survival time was 558.62 days with a standard deviation of 776.48 days.

Let's make a histogram of the survival times.

Using the `hist()` function, let's make a histogram of the survival times overall and add a title.

```
hist(cancer.survival$Survival,
     main="Histogram of Survival Times")
```



We can also do this using the pipe command `%>%` and we'll add include the title we used above. This time let's add a Figure caption using a chunk option `fig.cap`. Learn more about `knitr` chunk options at http://yihui.name/knitr/options/.

```
cancer.survival$Survival %>%
  hist(main="Histogram of Survival Times")
```

Make side-by-side boxplots of survival times by organ

For this plot we are going to use the `ggplot2` package and functions. So we have to install the package and load it.

```
# install.packages("ggplot2")
library(ggplot2)
ggplot(cancer.survival,
       aes(x = Organ,y=Survival, color = Organ)) +
  geom_boxplot() +
  stat_summary(fun.y=mean,
               geom="point",
               shape=23,
               size=4) +
  ggtitle("Survival time of patients affected by different cancers")
```

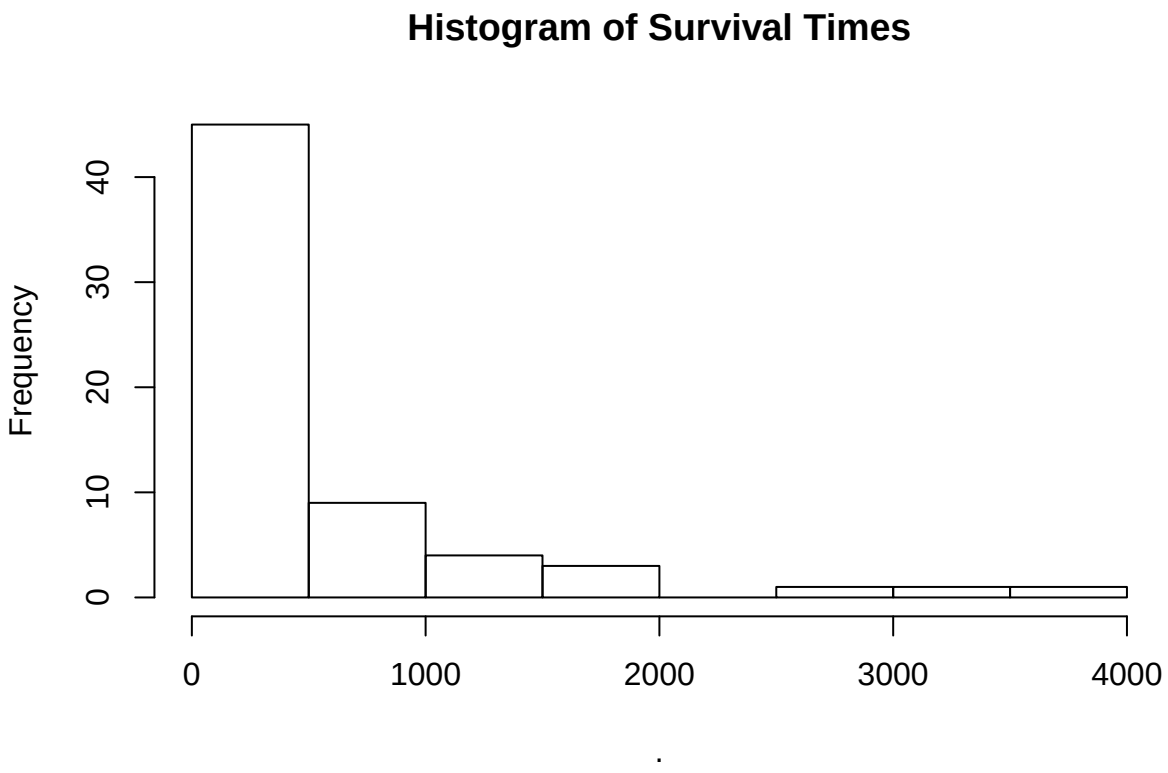
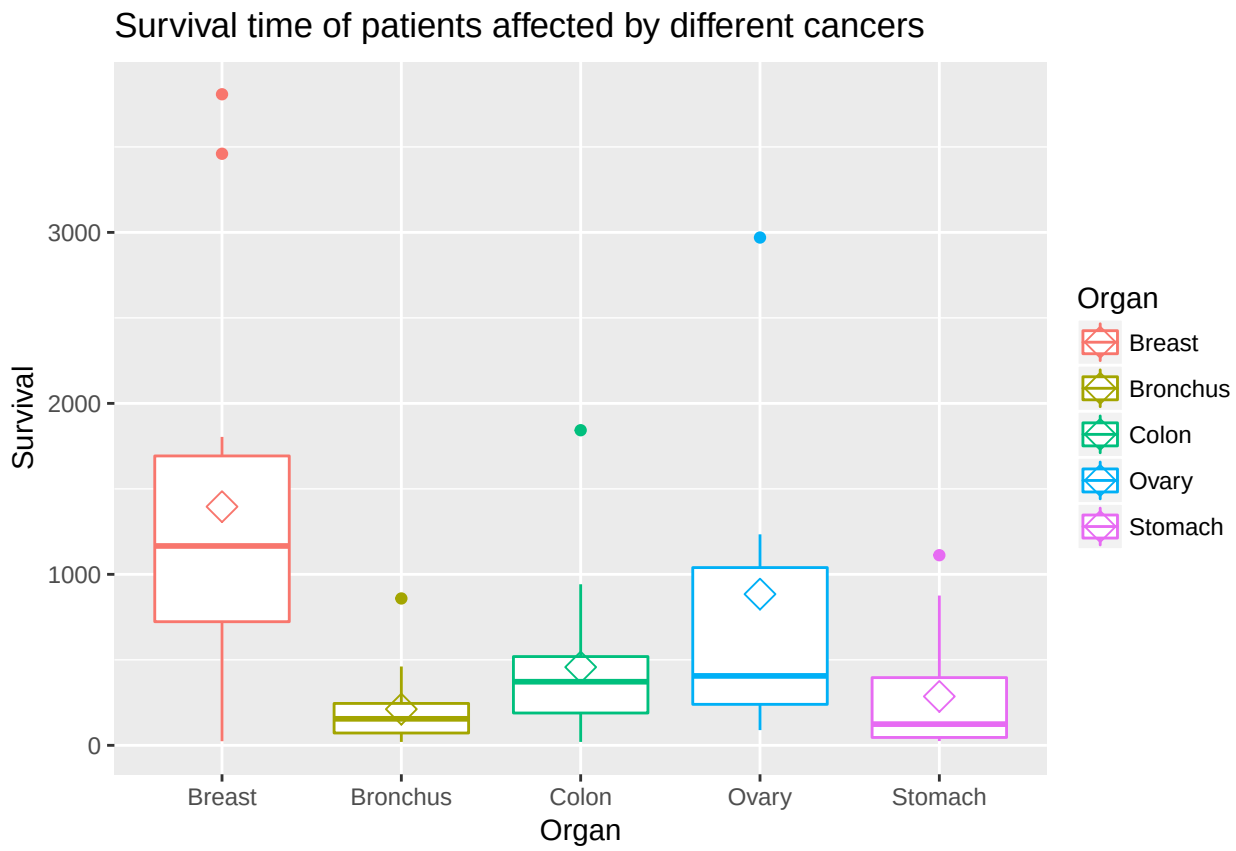


Figure 1: Figure: Histogram of Survival Times


Survival time of patients affected by different cancers
Survival
3000
2000
1000
0
Breast
Bronchus
Colon
Ovary
Stomach
Organ
Organ
Breast
Bronchus
Colon
Ovary
Stomach