# Lab 1 - Supplementary

Adam Shen

September 16, 2020

## Packages

A package is a collection of functions (and/or data sets). There are many packages not included in base-`R` that you may be interested in using. You must first install them before you can use them.

To install a package (only needs to be done once):

```r
install.packages("PackageName") # Don't forget the quotes
```

To load a package that you have already installed (needs to be done each time you open a new R session):

```r
library(PackageName) # Can be used with or without quotes
```

To use a function from a package without loading the whole package, use `::`. The package still needs to be installed beforehand.

```r
PackageName::theFunction()
```

Two useful packages for modelling are:

- `ggplot2`: Used for visualization; an alternative to base-`R` graphics
- `broom`: Used to transform model data into 'tidy' tables where the values are easily accessible

### Load packages

```r
library(ggplot2)
library(broom)
```

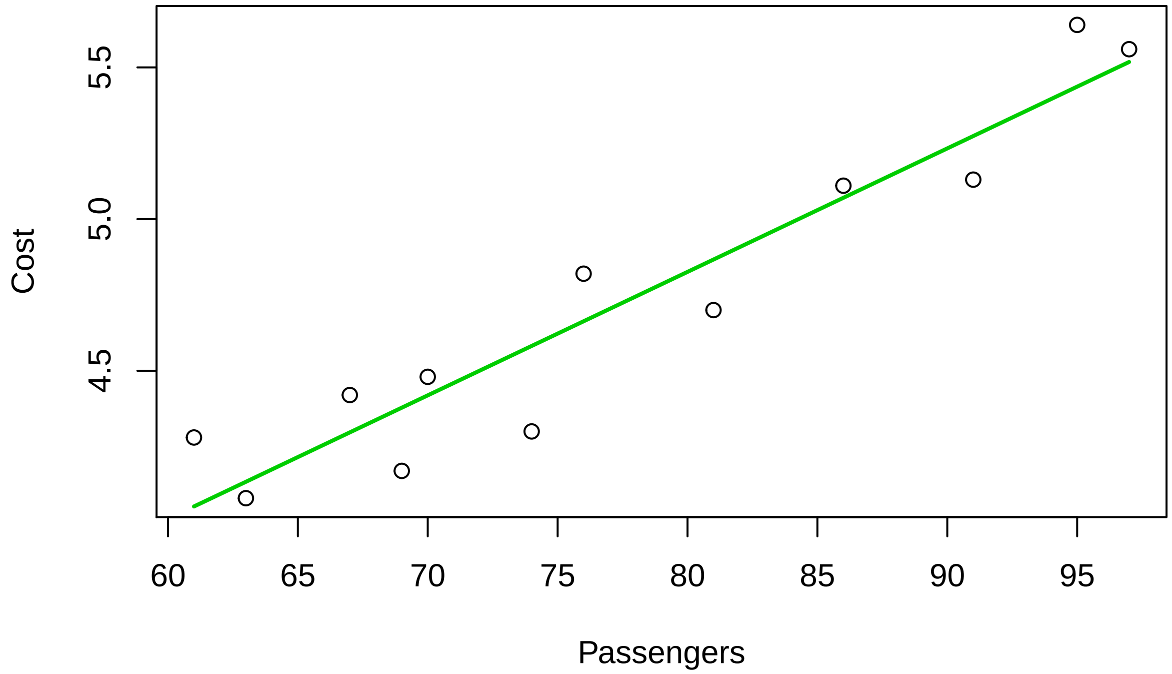
## Revisiting the airline data

```r
airline <- read.table("./boeing.txt", header=TRUE)
```

### Visualization Method 1

Previously, we had used the following commands to produce a scatterplot with the fitted regression line.

```r
m1 <- lm(Cost ~ Passengers, data=airline)
with(airline, plot(x=Passengers, y=Cost))
with(airline, lines(x=sort(Passengers), y=fitted(m1)[order(Passengers)], col="green3", lwd=2))
```



Although the data provided here was already sorted in the $x$-variable, when working with data that does not have a sorted $x$-variable, we have to remember to sort it prior to plotting the line. For straight-line data the difference is hard to see. For data that does not fall on a straight line (e.g. $\sin(x)$), you will get a bunch of overlapping lines as the points are joined in the order they appear in the data!

With `ggplot2`, we do not need to do any sorting of points. To illustrate this, we can rearrange the rows of the `airline` data so that `Passengers` is not sorted from least to greatest.

## Make a new data set with re-ordered rows

```
# For reproducibility
set.seed(5)

# Sample the row numbers without replacement, essentially re-ordering them
(ind <- sample(1:nrow(airline)))
```

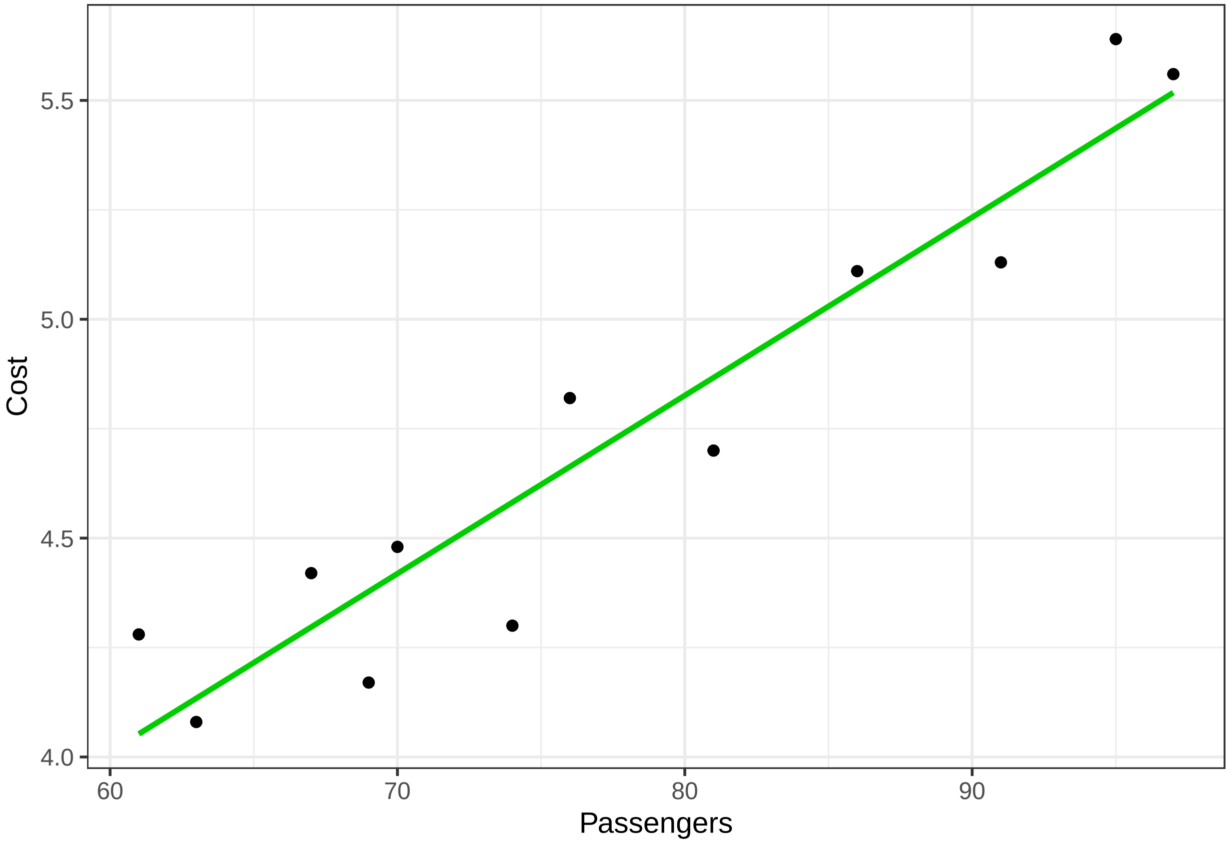
```
##  [1]  2 11  9 10  1  5  6  3  7 12  4  8
```

```
# Create a new data frame with rows corresponding to `ind`
airline_new <- airline[ind,]

# Reset the row names
rownames(airline_new) <- 1:nrow(airline_new)

# Compare them side by side - old on left, new on right
cbind(airline, airline_new)
```

```
##    Passengers Cost Passengers Cost
## 1          61 4.28         63 4.08
## 2          63 4.08          95 5.64
## 3          67 4.42          86 5.11
## 4          69 4.17          91 5.13
## 5          70 4.48          61 4.28
## 6          74 4.30          70 4.48
## 7          76 4.82          74 4.30
## 8          81 4.70          67 4.42
## 9          86 5.11          76 4.82
## 10         91 5.13          97 5.56
## 11         95 5.64          69 4.17
## 12         97 5.56          81 4.70
```

### Visualization Method 2

```
ggplot(airline_new, aes(x=Passengers, y=Cost))+
  geom_point()+
  geom_smooth(method="lm", formula=y~x, se=FALSE, colour="green3")+
  theme_bw()
```



Benefits:

- Computed a linear model but didn't store it
- Didn't need to sort points to draw the line

Explanation of code:

- `ggplot(airline_new, aes(x=Passengers, y=Cost))`: Initializes a ggplot canvas (blank) that will be taking data from `airline_new`. The $x$ and $y$-variables that will be used from `airline_new` are `Passengers` and `Cost`. `aes` stands for aesthetics, which are the plot layers' parameters.

- `geom_point()`: Adds a layer of points. Since there are no aesthetic parameters supplied within, it will inherit the ones from the initialization of the plot, namely `x=Passengers` and `y=Cost`.
- `geom_smooth(method="lm", formula=y~x, se=FALSE, colour="green3")`: This is used to draw a smooth line of a specified method.
    - `method="lm"`: The specified method is a linear regression model.
    - `formula=y~x`: The formula to be used for the linear regression model. Since our aesthetics were `x=Passengers` and `y=Cost`, the corresponding formula will be `Cost ~ Passengers`.
    - `se=FALSE`: We set this to `FALSE` otherwise it will draw a confidence band with the fitted line.
    - `colour="green3"`: Self-explanatory. Accepts `colour` or `color`.
- `theme_bw()`: Optional. Without this, the canvas background will be grey. White looks cleaner.
- Each line of code can be thought of as a layer. Layers are joined by a `+` symbol.

## Using broom with linear models

```
# Re-compute linear model with re-ordered data
m2 <- lm(Cost ~ Passengers, data=airline_new)

# Use tidy() to clean up the variable summary into a table with accessible values
(tidy_m2 <- tidy(m2))
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic    p.value
##   <chr>          <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)   1.57     0.338        4.64 0.000917
## 2 Passengers    0.0407   0.00431      9.44 0.00000269
```

```
# Use augment() to create a column for fitted values (and other things)
(augment_m2 <- augment(m2))
```

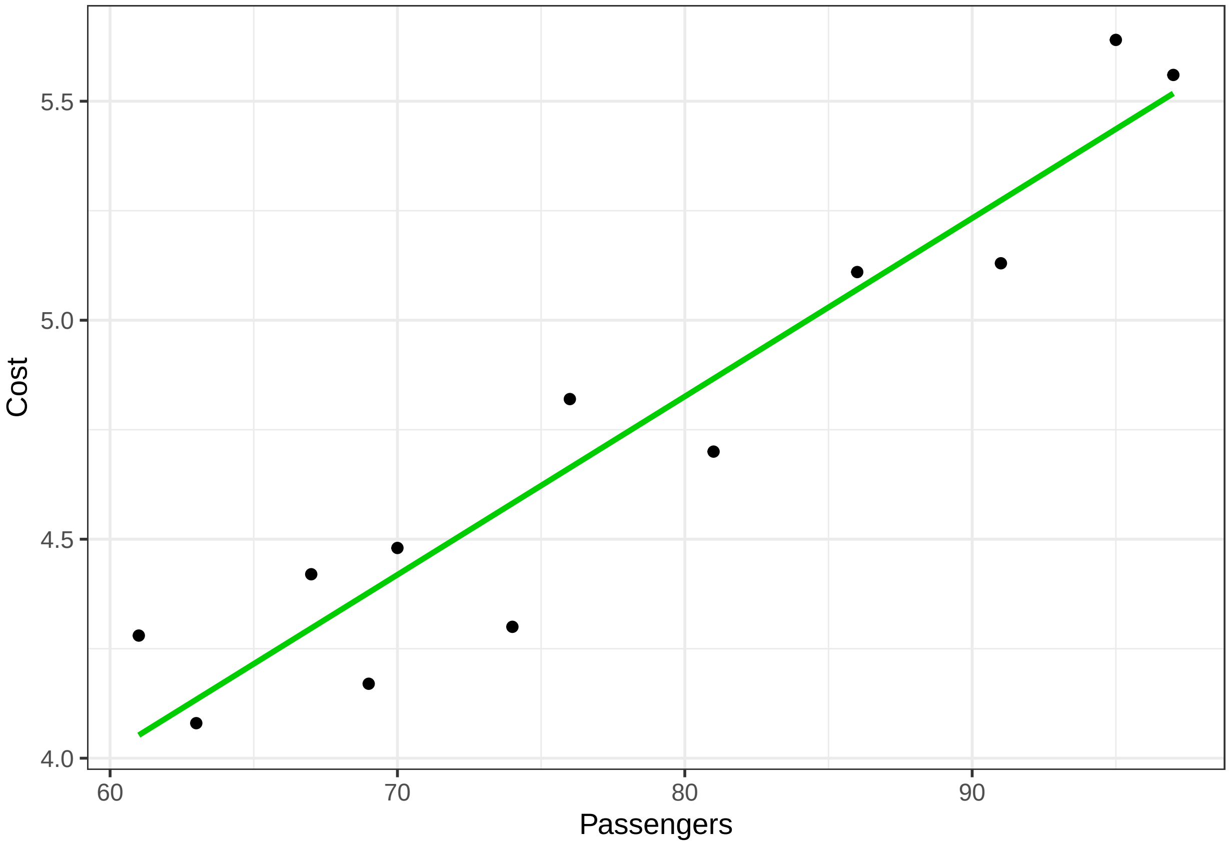
```
## # A tibble: 12 x 9
##     Cost Passengers .fitted .se.fit  .resid   .hat .sigma .cooksd .std.resid
##    <dbl>      <int>   <dbl>   <dbl>   <dbl>  <dbl>  <dbl>   <dbl>      <dbl>
##  1  4.08         63    4.13  0.0808 -0.0540 0.208   0.186 0.0154      -0.342
##  2  5.64         95    5.44  0.0912  0.204  0.265   0.169 0.323        1.34
##  3  5.11         86    5.07  0.0629  0.0399 0.126   0.186 0.00418      0.241
##  4  5.13         91    5.27  0.0775 -0.144  0.191   0.179 0.0960      -0.901
##  5  4.28         61    4.05  0.0876  0.227  0.245   0.165 0.353        1.48
##  6  4.48         70    4.42  0.0605  0.0611 0.117   0.186 0.00888      0.367
##  7  4.3          74    4.58  0.0533 -0.282  0.0906  0.159 0.138       -1.67
##  8  4.42         67    4.30  0.0683  0.123  0.149   0.181 0.0495       0.753
##  9  4.82         76    4.66  0.0516  0.157  0.0847  0.179 0.0396       0.925
## 10  5.56         97    5.52  0.0984  0.0422 0.308   0.186 0.0182       0.286
## 11  4.17         69    4.38  0.0629 -0.208  0.126   0.171 0.114       -1.26
## 12  4.7          81    4.87  0.0533 -0.167  0.0906  0.177 0.0484      -0.986
```

For an explanation of the column values, see the documentation:

```
?augment.lm # If `broom` was loaded
?broom::augment.lm # If `broom` wasn't loaded
```

### Visualization Method 3

```
ggplot(augment_m2, aes(x=Passengers))+
  geom_point(aes(y=Cost))+
  geom_line(aes(y=.fitted), colour="green3", size=1)+
  theme_bw()
```



Benefits:

- Didn't need to sort points to draw the line

Explanation of code:

- `ggplot(augment_m2, aes(x=Passengers))`: Initializes a ggplot canvas (blank) that will be taking data from `augment_m2`. The $x$-variable that will be used is `Passengers`. The $y$-variable has not been declared as our layers will be using different $y$-variables.
- `geom_point(aes(y=Cost))`: Draw a layer of points using `x=Passengers` and `y=Cost`, both found in the data.
- `geom_line(aes(y=.fitted), colour="green3", size=1)`: Draw a line using `x=Passengers` and `y=.fitted`. `geom_line()` automatically connects points from left to right, so there is no need for sorting. `colour="green3"` is self-explanatory. `size=1` was used because the default thickness of the line was a bit too thin for my liking.
- `theme_bw()`: Again, optional but because I wanted a white background rather than a grey background.
- Once again, all the layers are joined using `+`

## Additional Links

Hopefully, we will be able to use more packages from the tidyverse at a later date. For now, here are some useful links:

- A free book introducing R, tidy principles, and some tidyverse packages: https://r4ds.had.co.nz/
- About the tidyverse and its packages: https://tidyverse.tidyverse.org/
- `ggplot2` reference: https://ggplot2.tidyverse.org/reference/index.html
- `broom` reference: https://broom.tidymodels.org/reference/index.html