Exploring Length-Weight Data

In this document, you will use R to explore length and weight data recorded for 100 bluegills captured from Lake Mary, Minnesota. The objective for this analysis is to become familiar with the R interface and to see what types of analyses R can perform. Some of the specific functions used in this document will be discussed in more detail at the workshop. However, you should follow these commands to get a general feel for R.

1 Initialization

Nearly all R sessions will begin with you loading packages that you have previously installed. This R session will only require the `FSA` package (and the packages that it depends on). Thus, assuming that you have already installed R and the packages it depends on (if not then see this document), you should load R with,

```
> library(FSA)
```

The typical second step in most analyses is to load an external file that contains the data to analyze. The data that you will use in this session is the Lake Mary bluegill data found in the BluegillLM1.xls Excel spreadsheet that is linked to on the class webpage. This file should be downloaded from the webpage to a directory on your computer.

The first step in loading this file into R is to change the R working directory to the directory where you saved the downloaded file. This is accomplished with the `File..Change Dir` menu item. In the ensuing dialog boxes you should browse to the directory where you saved the file and then click `OK`. Note that you will not see the actual file; you will only see the directory that it is contained in.

The Excel spreadsheet can be loaded into R with `read.xls()` from the `xlsReadWrite` package (this will require you to download and install the `xlsReadWrite` package from CRAN – see this document). The first argument to this function is the name of the file (contained in quotes). Optional arguments include the name of the sheet on which the data resides and which row number the data begins. This particular set of data is on the first sheet and the data begins in row number 4. Thus,

```
> library(xlsReadWrite)
> bg <- read.xls("BluegillLM1.XLS", sheet = 1, from = 4)
```

The structure and a random sample of rows in this data frame can be observed with,

```
> str(bg)
```

```
'data.frame':        100 obs. of  5 variables:
 $ sernum: num  100 110 120 130 140 150 160 170 180 190 ...
 $ sl    : num  126 124 118 136 191 121 105 123 123 120 ...
 $ fl    : num  152 150 140 163 144 149 128 151 148 148 ...
 $ tl    : num  157 153 149 172 150 155 134 157 155 154 ...
 $ wght  : num  92 88 77 130 86 90 53 90 79 83 ...
```

```
> rhead(bg)
```

```
   sernum  sl  fl  tl wght
52    610 110 138 143   70
53    620 100 122 127   40
47    560 154 189 196  206
11    200 138 170 177  131
45    540 165 197 204  232
68    770 130 157 163  114
```

From this we see that the data frame has five numerical variables – *sernum* , *tl* , *fl* , *sl* , and *wght* . The left-most column of numbers are row number labels.

2 Simple Summaries

2.1 Basic Statistics

Simple summary statistics for the variables in the data frame can be computed with `Summary()` as follows,

```
> Summary(bg$tl)
```

```
       n       NAs   Valid n      Mean  St. Dev.      Min.   1st Qu.    Median   3rd Qu.
100.00000   0.00000 100.00000 162.41000  33.32275  55.00000 143.00000 155.00000 177.20000
     Max.
230.00000
```

or, rounded to two decimals,

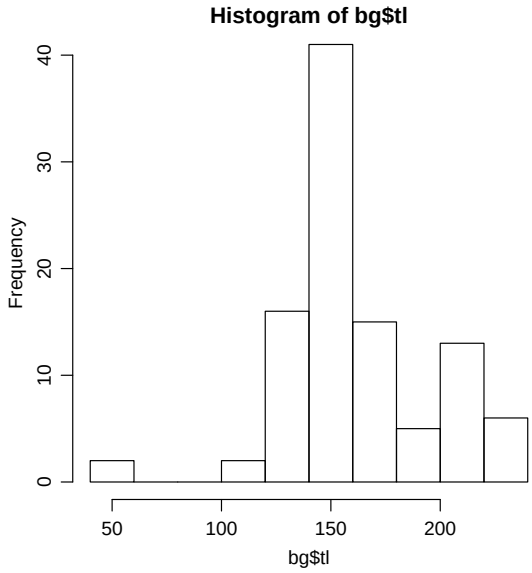
```
> round(Summary(bg$tl), 2)
```

```
      n      NAs  Valid n     Mean St. Dev.     Min.  1st Qu.   Median  3rd Qu.     Max.
 100.00     0.00   100.00   162.41    33.32    55.00   143.00   155.00   177.20   230.00
```

2.2 Histograms or Length Frequencies

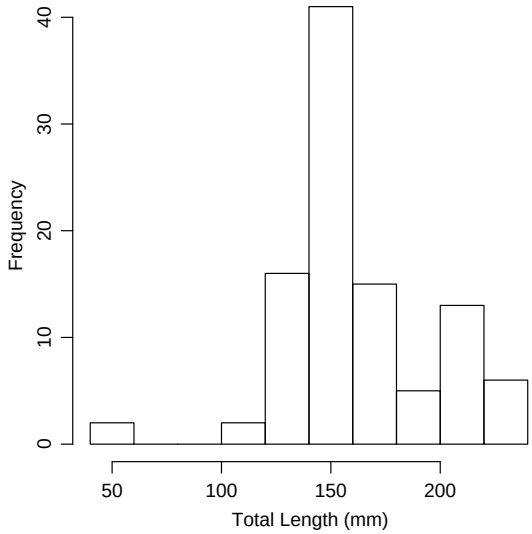
A simple histogram of the total lengths can be constructed with `hist()` as follows,

```
> hist(bg$tl)
```



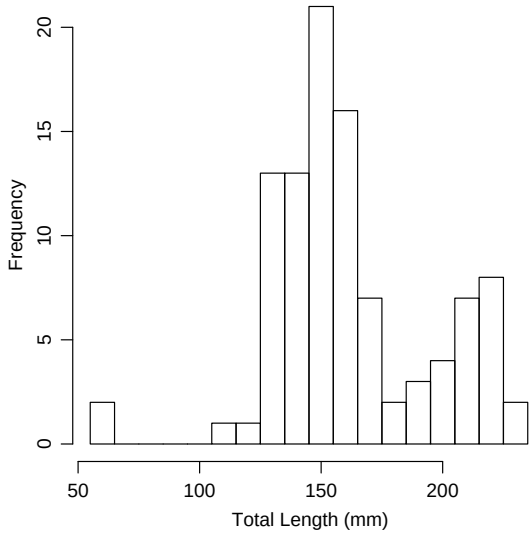
Of course, this has an irritating main title and poor axis labels. Both of these issues can be corrected with,

```
> hist(bg$tl, main = "", xlab = "Total Length (mm)")
```



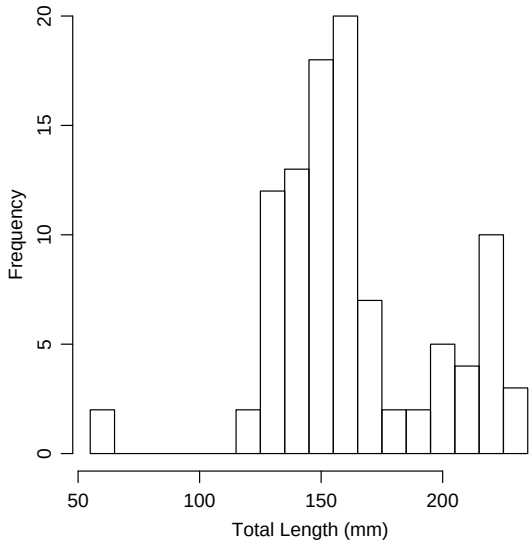
In addition, the breaks for the bars in the histogram may be too course to identify potential age-clases in the data. The breaks, for example, can be set to 10-mm wide beginning at 55 mm (and ending at 235 mm; these endpoints were chosen by looking at the minimum and maximum TL values in the summary statistics above) with,

```
> brks <- seq(55, 235, 10)
> hist(bg$tl, main = "", xlab = "Total Length (mm)", breaks = brks)
```



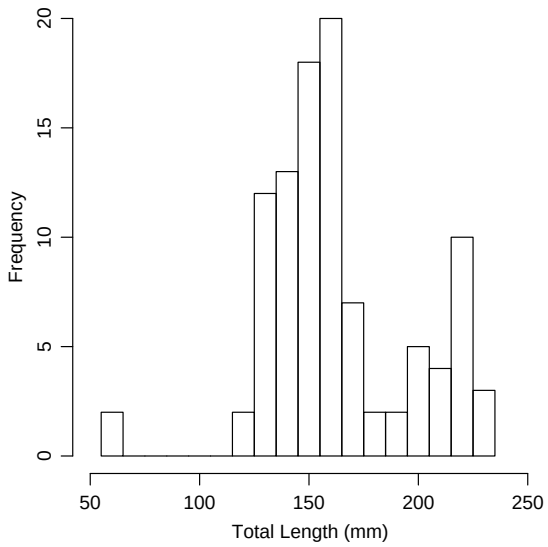
Furthermore, R can be told to be left-inclusive rather than right-inclusive (e.g., so that that a 65 mm fish is included in the 65-75 mm grouping rather than the 55-65 mm grouping) with,

```
> brks <- seq(55, 235, 10)
> hist(bg$tl, main = "", xlab = "Total Length (mm)", breaks = brks, right = FALSE)
```



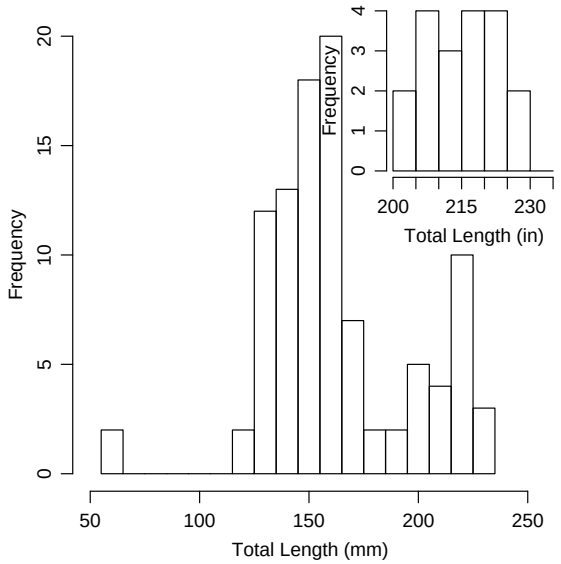
Also you may not like the fact that the x-axis does not extend as far to the right as the data. This can be corrected with,

```
> hist(bg$tl, main = "", xlab = "Total Length (mm)", breaks = brks, right = FALSE,
+     xlim = c(50, 250))
```



Finally, suppose that you want to focus on the fish greater than 200 mm. You could construct a plot such as this,

```
> hist(bg$tl, main = "", xlab = "Total Length (mm)", breaks = brks, right = FALSE,
+     xlim = c(50, 250))
> subplot(hist(bg$tl[bg$tl > 200], breaks = seq(200, 235, 5), main = "",
+     xlab = "Total Length (in)"), 225, 17.5, c(1.5, 1.5))
```



A length frequency table can be constructed by first adding a length categorization variable to the *bg* data frame. This variable can be easily constructed with `lencat()` . The length categorizations in the example below start with a value of 50-mm and each length class is 10-mm wide.

```
> bg.mod <- lencat(bg, "tl", startcat = 50, w = 10)
> rhead(bg.mod)
```

```
   sernum  sl  fl  tl wght LCat
34    430 137 164 170  125  170
```

```
40     490 169 201 210  236  210
60     690 175 215 221  277  220
22     310 123 148 155   92  150
55     640 106 128 134   55  130
51     600 130 158 163  106  160
```

The `table()` function can then be used to summarize the frequency of individuals in each of the length classes. In the example below this table is saved to an object and, thus, to see the length frequency the name of this object must be submitted to R. The proportin of individuals in each length category can be found by submitting the saved `table()` object to the `prop.table()` function.

```
> bg.tab <- table(bg.mod$LCat)
> bg.tab
```

```
 50 110 120 130 140 150 160 170 180 190 200 210 220 230
  2   1   5  11  20  20  10   7   2   3   4   8   6   1
```

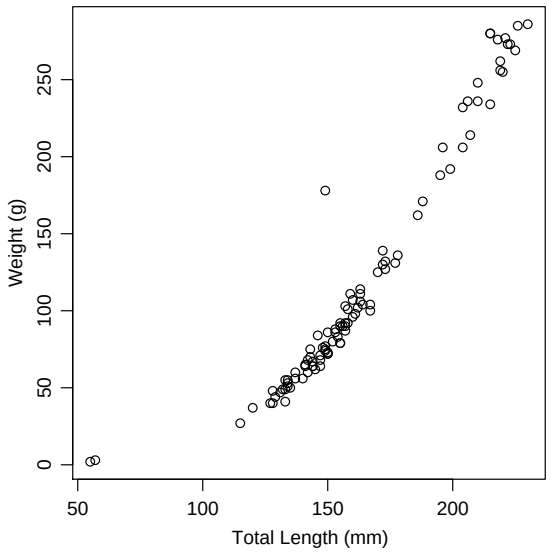
```
> prop.table(bg.tab)
```

```
   50   110   120   130   140   150   160   170   180   190   200   210   220   230
0.02 0.01 0.05 0.11 0.20 0.20 0.10 0.07 0.02 0.03 0.04 0.08 0.06 0.01
```

2.3 Scatterplots

The relationship between total length and weight can be examined with a scatterplot constructed with `plot()` as follows,

```
> plot(bg$wght ~ bg$tl, xlab = "Total Length (mm)", ylab = "Weight (g)")
```



The outlier in the plot can be identified with `identify()` as long as the plot above is still active (i.e, you have not made another plot or closed this plot). This function suspends the activity of the R console until

you press the `Stop` button on the R toolbar. If you click on a "point" on the plot before you press the `Stop` button then R will display the row in which that individual if found. If you click on more than one plot it will show the rows for all selected individuals. This is an interactive function that cannot be displayed in this textual document. However, the command can be implemented with,

```
> identify(bg$wght ~ bg$tl)
```

A click on the plot shows that the outlier is individual number 41. The data for this individual can be found with (note that the comma with nothing behind it is intentional),

```
> bg[41, ]
```

```
   sernum  sl  fl  tl wght
41    500 119 143 149  178
```

Individuals within five millimeters total length of this individual can be found with,

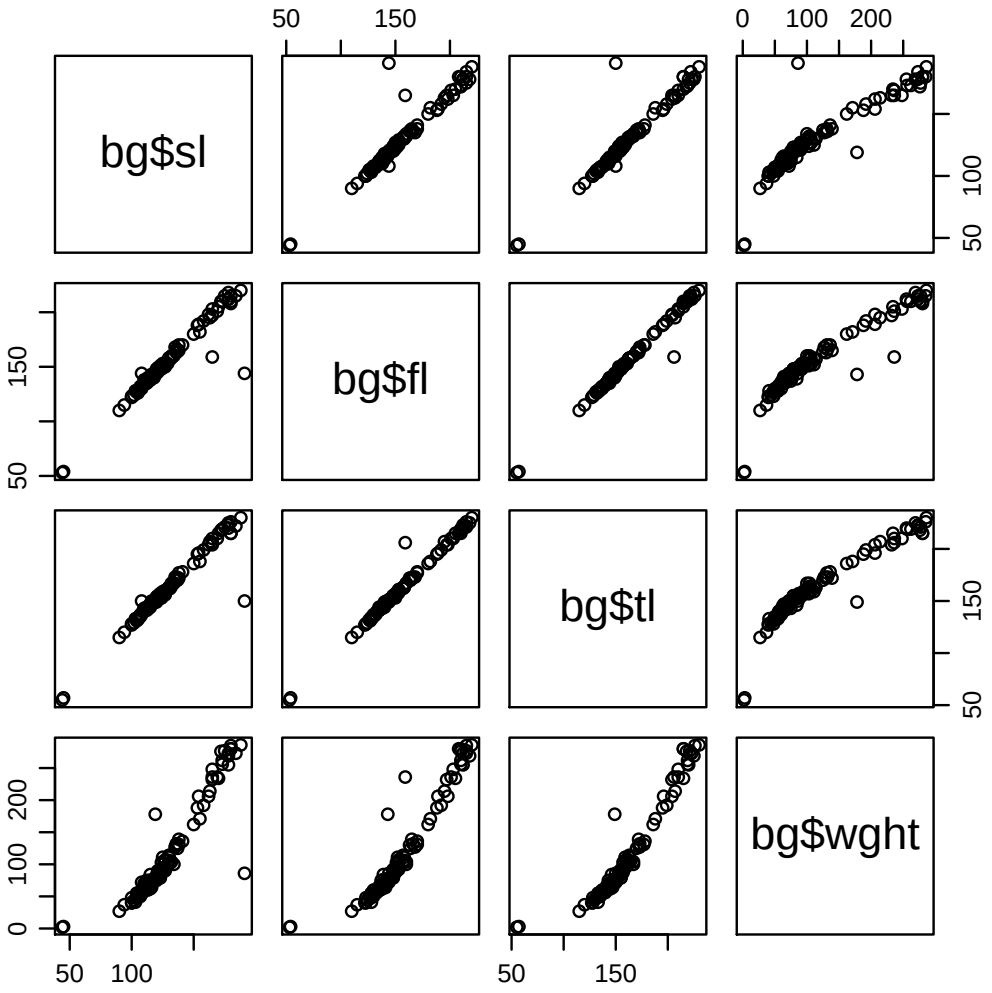
```
> bg[bg$tl > 144 & bg$tl < 154, ]
```

```
   sernum  sl  fl  tl wght
2     110 124 150 153   88
3     120 118 140 149   77
5     140 191 144 150   86
14    230 115 140 147   71
17    260 116 142 148   76
20    290 124 148 153   86
28    370 115 139 146   84
31    400 118 142 150   73
37    460 121 145 152   80
39    480 118 143 149   74
41    500 119 143 149  178
56    650 120 144 149   75
76    850 115 140 147   68
87    960 115 138 145   62
90    990 108 144 150   72
97   1060 116 143 150   72
98   1070 116 141 147   64
```

This analysis shows, as did the plot, that this individual has an abnormally high weight for it's measured length. In fact, it appears as if the weight should be 78 g rather than 178 g.

The relationships between all pairs of variables can be examined at one time with `plot()` as follows,

```
> plot(~bg$sl + bg$fl + bg$tl + bg$wght)
```



3 Length-Weight Regression

Traditionally, length-weight data follows a power function (notice curvature and increasing variance in the *wght* versus *tl* plot above). Thus, the relationship between the length and weight of fish is often modeled on the log-log scale. Variables can be transformed to the log scale and saved into the original data frame with,

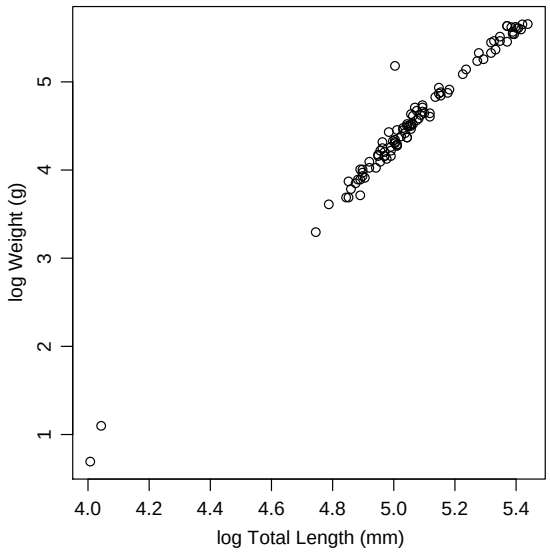
```
> bg$logW <- log(bg$wght)
> bg$logTL <- log(bg$tl)
> rhead(bg)
```

```
   sernum  sl  fl  tl wght     logW    logTL
58    670 136 166 173  132 4.882802 5.153292
35    440 180 208 215  280 5.634790 5.370638
50    590 111 136 142   68 4.219508 4.955827
24    330 102 124 129   44 3.784190 4.859812
```

```
93    1020 109 133 140   56 4.025352 4.941642
64     730 138 165 172  139 4.934474 5.147494
```

and the plot on the log-log scale constructed with,

```
> plot(bg$logW ~ bg$logTL, xlab = "log Total Length (mm)", ylab = "log Weight (g)")
```



Of course, the outlier is still evident on this plot. There are at least three ways to address this problem. First, the individual can be entirely removed (this removes all information for this indivdual) as follows,

```
> bg1a <- bg[-41, ]
```

Second, the weight can be "corrected" with,

```
> bg1b <- bg
> bg1b$wght[41] <- 78
```

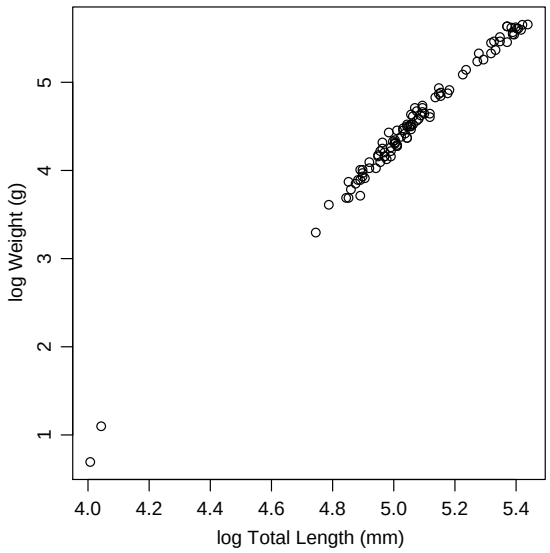
Third, the weight can be replaced with `NA` , which means "not available", with,

```
> bg1c <- bg
> bg1c$wght[41] <- NA
```

In each case, we will have to re-created the *logW* variable. We will assume that the erroneous weight measurement for individual 41 cannot be "corrected" and will replace it with a `NA` (i.e., we will use the `bg1c` data frame).

Then the scatterplot can be reconstructed with,

```
> bg1c$logW <- log(bg1c$wght)
> plot(bg1c$logW ~ bg1c$logTL, xlab = "log Total Length (mm)", ylab = "log Weight (g)")
```

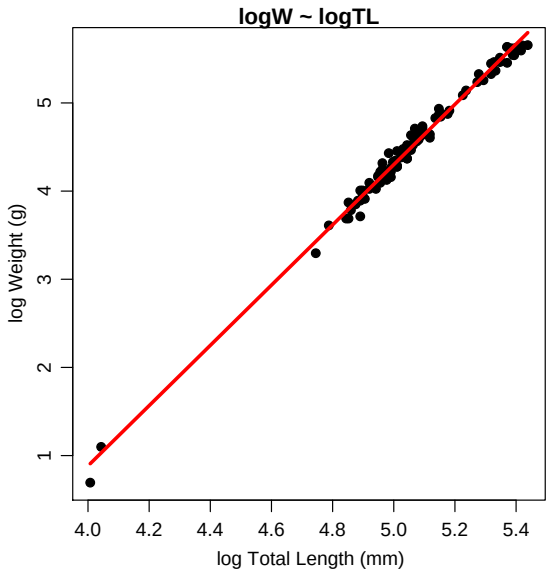


We can now try to fit the usual linear regression model to these data on the log-log scale with,

```
> attach(bg1c)
> lw.lm <- lm(logW ~ logTL)
```

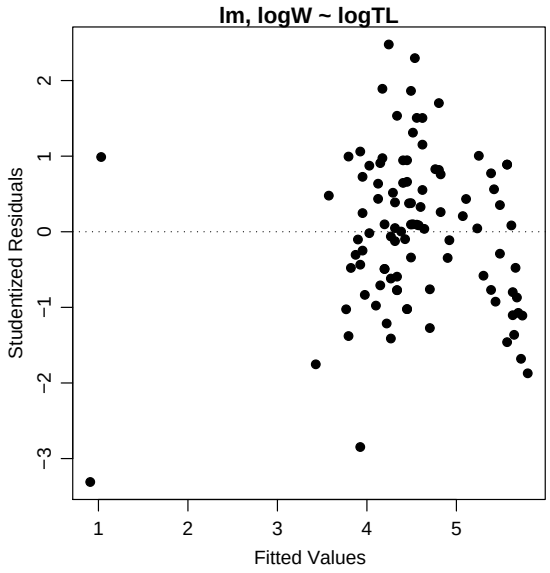
A visual of this model is then constructed with,

```
> fit.plot(lw.lm, xlab = "log Total Length (mm)", ylab = "log Weight (g)")
```



and a corresponding residual plot is constructed with,

```
> residual.plot(lw.lm)
```



It is apparent from the residual plot that these data are NOT linear and the assumptions of the linear regression model have not been met. At this point, other transformations should be considered for these data.

Because the linear model assumptions have not been met it is inappropriate to look at the model results. However, AS AN ILLUSTRATION ONLY, the summary statistics of the model fit can be extracted with,

```
> summary(lw.lm)


Call:
lm(formula = logW ~ logTL)

Residuals:
      Min        1Q    Median        3Q       Max
-0.215245 -0.059123  0.003816  0.057525  0.186642

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -12.78408    0.17394  -73.50   <2e-16 ***
logTL         3.41685    0.03429   99.64   <2e-16 ***
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1

Residual standard error: 0.07775 on 97 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-Squared: 0.9903,        Adjusted R-squared: 0.9902
F-statistic:  9929 on 1 and 97 DF,  p-value: < 2.2e-16
```

Thus, the model has a slope of 3.42 and an intercept of -12.78.