1-Factor GLM: ChickWeight example

Crispin Jordan

08/06/2021

Introducing the dataset

We will analyse the `ChickWeight` dataset, which appears in **R**'s memory. We obtain the data with this command:

```
data("ChickWeight")
```

Let's look at the first 30 lines of code to get a sense of what the data look like:

```
ChickWeight[1:30,]
```

```
##    weight Time Chick Diet
## 1      42    0     1    1
## 2      51    2     1    1
## 3      59    4     1    1
## 4      64    6     1    1
## 5      76    8     1    1
## 6      93   10     1    1
## 7     106   12     1    1
## 8     125   14     1    1
## 9     149   16     1    1
## 10    171   18     1    1
## 11    199   20     1    1
## 12    205   21     1    1
## 13     40    0     2    1
## 14     49    2     2    1
## 15     58    4     2    1
## 16     72    6     2    1
## 17     84    8     2    1
## 18    103   10     2    1
## 19    122   12     2    1
## 20    138   14     2    1
## 21    162   16     2    1
## 22    187   18     2    1
## 23    209   20     2    1
## 24    215   21     2    1
## 25     43    0     3    1
## 26     39    2     3    1
## 27     55    4     3    1
## 28     67    6     3    1
## 29     84    8     3    1
## 30     99   10     3    1
```

We see four columns: `weight`, `Time`, `Chick`, `Diet`.

Stage 1: Determining the appropriate analysis.

Before we begin an analysis, we must consider the experimental design to determine an appropriate approach. The `ChickWeight` data come from an experiment that examined the effect of four diets (1-4; see column `Diet`) upon the weight of chicks (see column `weight`). Examine the columns `Time`, `Chick` and `Diet`. Notice that each chick only experienced one diet, but each chick was also measured over a series of times (`Time`).

Notice that the experiment only has one general type of experimental manipulation (`Diet`) with 4 levels (Diets 1 to 4). Also, the measurements are on a continuous scale. These two facts suggest that we could analyse the data with a 1-Factor GLM. However, before we proceed we need to ensure that the data meet two essential assumptions of a 1-Factor GLM:

- Subjects are assigned randomly to treatments;
- Data within treatments (level of `Diet`, in our case) are independent.

Without knowing more about the experiment, we can't say whether the data meet the assumption of random assignment; we'll assume they were randomly assigned, for the sake of this learning exercise.

However, it is clear that the data do not meet the assumption of independence: individual subjects are measured multiple times within a type of treatment (i.e., within a level of `Diet`). This means that we cannot analyze the entire dataset with a 1-Factor GLM; instead, we'd use an approach like a mixed effects model. However, we've not yet learned mixed effects models. Therefore, to obtain a dataset appropriate for 1-Factor GLM, we will extract data for the last timepoint for each chick, which occurs at `Time` 21:

```
ch21 <- ChickWeight[which(ChickWeight$Time == 21),]
```

Now, let's look at the whole dataset, which we've renamed `ch21` (for **Ch**ickWeight at time 21):

```
ch21
```

```
##     weight Time Chick Diet
## 12     205   21     1    1
## 24     215   21     2    1
## 36     202   21     3    1
## 48     157   21     4    1
## 60     223   21     5    1
## 72     157   21     6    1
## 84     305   21     7    1
## 107     98   21     9    1
## 119    124   21    10    1
## 131    175   21    11    1
## 143    205   21    12    1
## 155     96   21    13    1
## 167    266   21    14    1
## 194    142   21    17    1
## 208    157   21    19    1
## 220    117   21    20    1
## 232    331   21    21    2
## 244    167   21    22    2
## 256    175   21    23    2
## 268     74   21    24    2
## 280    265   21    25    2
## 292    251   21    26    2
## 304    192   21    27    2
## 316    233   21    28    2
## 328    309   21    29    2
## 340    150   21    30    2
## 352    256   21    31    3
```

```
## 364    305   21    32    3
## 376    147   21    33    3
## 388    341   21    34    3
## 400    373   21    35    3
## 412    220   21    36    3
## 424    178   21    37    3
## 436    290   21    38    3
## 448    272   21    39    3
## 460    321   21    40    3
## 472    204   21    41    4
## 484    281   21    42    4
## 496    200   21    43    4
## 518    196   21    45    4
## 530    238   21    46    4
## 542    205   21    47    4
## 554    322   21    48    4
## 566    237   21    49    4
## 578    264   21    50    4
```

Notice that now the data are all independent, so far as we can tell (i.e., given that we don't know all details for this experiment). Now, our data are appropriate for a 1-Factor GLM and we can proceed.

Stage 2: Plot the data

It is important to begin an analysis by plotting your data. We do this with several goals in mind:

- Obtain a preliminary sense of whether the data will meet the assumption of normally distributed residuals;
- Obtain a preliminary sense of whether the data will meet the assumption of equal variance;
- Check for outliers;
- Make predictions with respect to differences between groups.

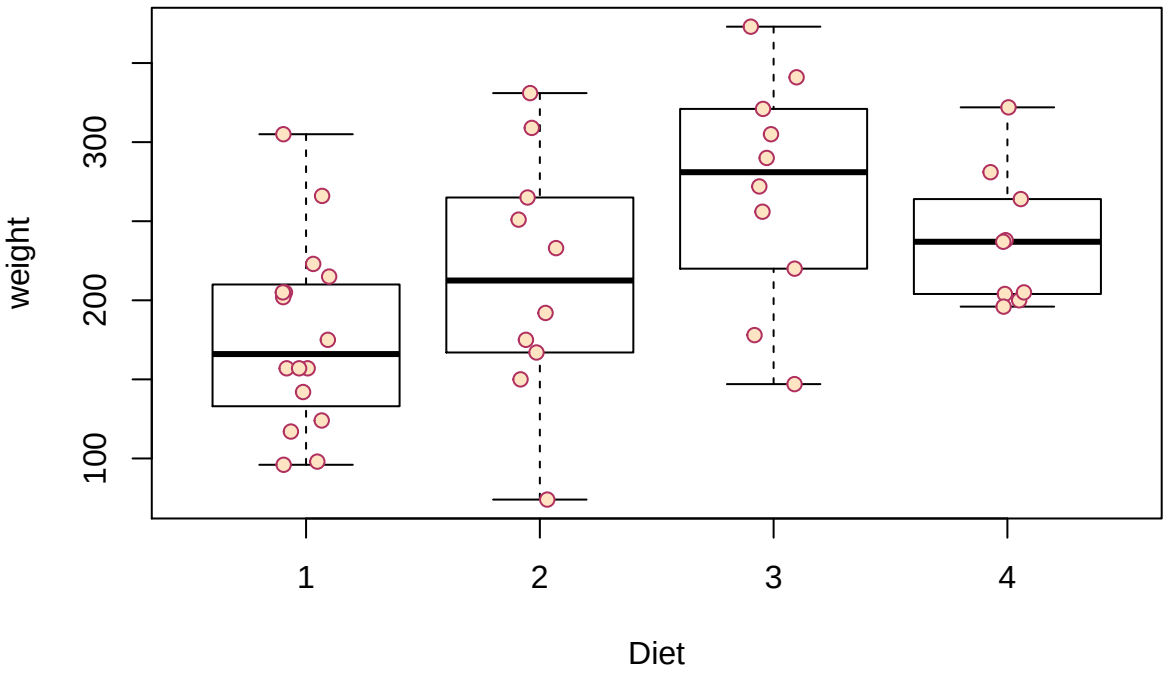
We will check each of these predictions at a later stage in our analysis, below. You might ask yourself, "If we're going to check these predictions later, why make the predictions in the first place?" Excellent question! We make these predictions early on as a 'validation' for results we obtain, below. If results we obtain later do not match our predictions then either: 1) our predictions were wrong, or 2) we made a mistake in our analysis. Therefore, if our predictions to not match our later findings, it behoves us to determine why.

We have a big decision to make before plotting the data: which data will go on the x-axis and y-axis? Traditionally, the x-axis displays the variable that we hypothesize to cause a change in the variable plotted along the y-axis. The `ChickWeight` experiment presumably was conducted to determine whether `Diet` affects `weight` (this was the hypothesis being tested). Therefore, we would plot `Diet` along the x-axis and `weight` along the y-axis. As another way of saying this, we refer to `weight` as the **dependent** variable and `Diet` as the **independent** variable because we hypothesize that `weight` *depends on* `Diet`. We will use this terminology a lot, so it is important to get used to it.

To plot `weight` as a dependent (y-) variable and `Diet` as the independent (x-) variable in a boxplot, we use this command: `boxplot(weight ~ Diet, data=ch21)`. When we call this command, we're telling **R** to obtain the data found in the column `weight` of the dataframe, `ch21`, and use those data for the y-axis; likewise for obtaining data in the column `Diet` in `ch21` for the x-axis.

Here is code to plot the `boxplot` as well as individual values:

```
boxplot(weight ~ Diet, data=ch21)
stripchart(weight ~ Diet, data=ch21, vertical = TRUE, method = "jitter", pch = 21,
           col = "maroon", bg = "bisque", add = TRUE)
```



What do we see? Let's assess our 4 criteria, above:

- Notice that the boxplots are roughly symmetrical around the median (dark, horizontal line); this implies that we expect the data will be (at least roughly) normally distributed. Note that this, casual, check of normality is not sufficient to be sure our data meet assumptions: we still need to check our assumptions, below.
- Notice that the spread of the data is generally similar among the levels of `Diet`; we predict that the assumption of equal variance will be met. If we find any evidence of unequal variance, we predict that it will arise from `Diet` 4, for which the data are slightly less variable (but they still look fine). Again, we must still perform a proper test of assumptions (i.e., examine residuals) to be certain.
- We see no evidence for outliers.
- We will predict differences between `Diet` 1 vs., `Diet` 2, 3 and 4. Based on this plot, I'd guess the the mean for `Diet` 1 equal 175. I'd similarly guess that the mean values for `Diet` 2, 3 and 4 equal 210, 275, and 250, respectively. Therefore, I would predict that the difference between `Diets` 1 & 2 equals `210 - 175 = 35`, 1 & 3 equals `275 - 175 = 100`, and 1 & 4 equals `250 - 175 = 75`. Note that the biggest difference lies between `Diet` types 1 and 3; therefore, **if we find evidence for a difference among levels of `Diet`, we expect it to occur here.**

With these predictions in mind, we'll proceed with our analysis.

Stage 3: Formulating a model, checking assumptions, and checking overall result

We'll analyze the data with a 1-Factor GLM by implementing the function, `lm()`. We use `lm()` in the same way as we used `boxplot()`: we specify our **dependent** variable, then add a 'tilda' (i.e., `~`), then we specify the **independent** variable, and specify where **R** can find these variables (i.e., `data = ch21`).

One important note before we proceed. Recent versions of **R** require that we tell a function when we want it to use a variable as a 'factor'. In our example, we want the information in the column, `Diet` to be considered as 4 *categories*, **not** as 4 *'numbers'*. To ensure that the `lm()` function correctly interprets the content of the column, `Diet`, (i.e., to treat it as a factor) we place the column name in the function, `factor()`, which converts the object passed to it to a factor. Note **that, with recent versions of R, we should do this even if the data in the column we wish to treat as a `factor` are not numeric (e.g., even if the data contain letters of the alphabet)**.

Here's our model:

```
ch21.lm <- lm(weight ~ factor(Diet), data=ch21)
```

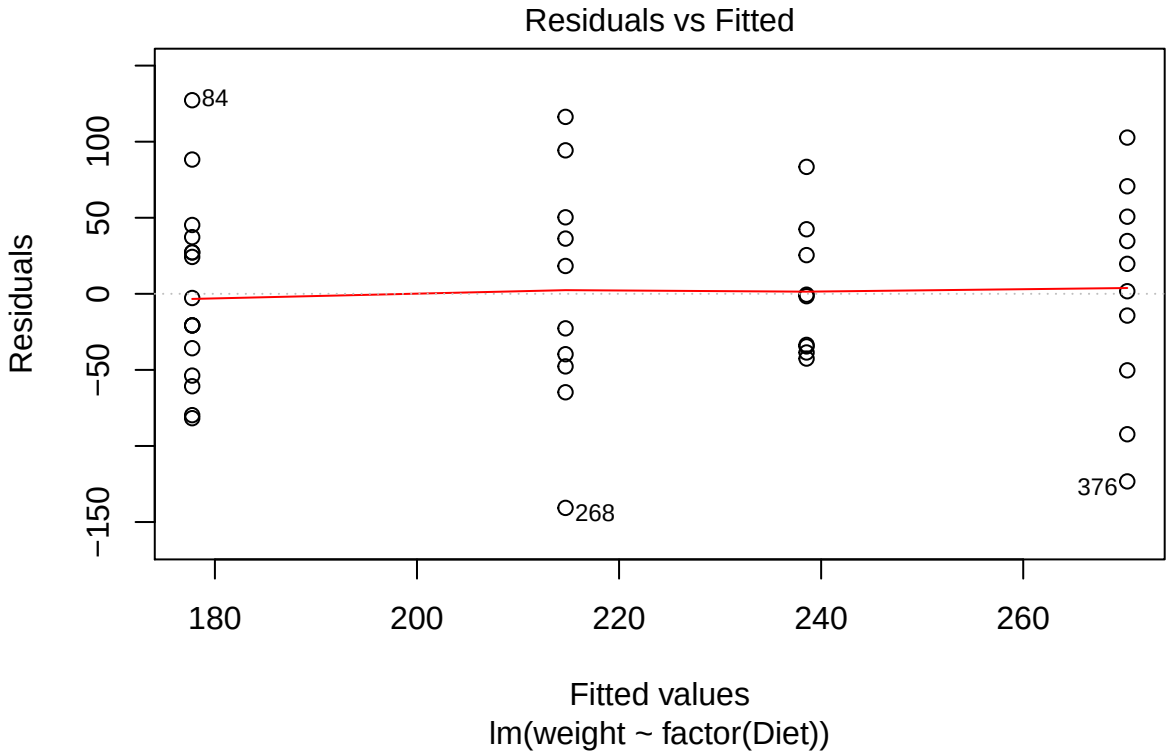
Notice that I saved the output in an object I called `ch21.lm`. I like this nomenclature because it helps me remember where I have saved various output, and this helps keep me organized. In my choice of the name, `ch21.lm`, the `ch21` refers to the dataset, and `.lm` indicates that this object contains the output from the `lm()` function. Please adopt this approach to keeping track of your work if you think it is helpful; if you find another way that works better for you then please use it! The important thing is to try to stay organized.

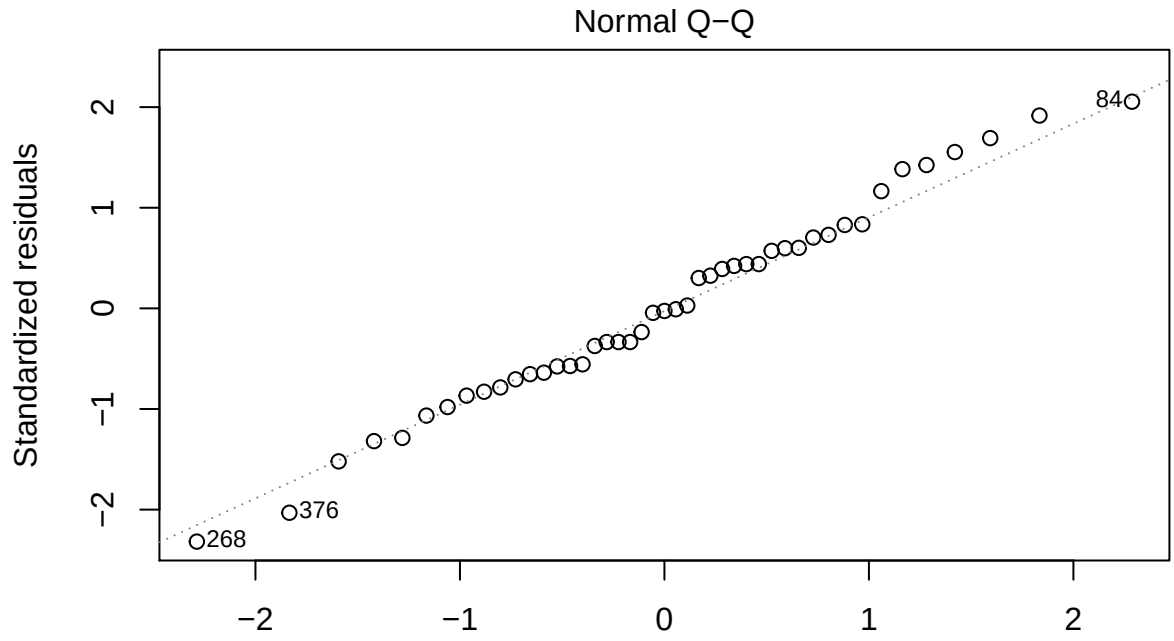
Our first task is to check the assumptions. Recall that the assumptions for 1-Factor GLM include:

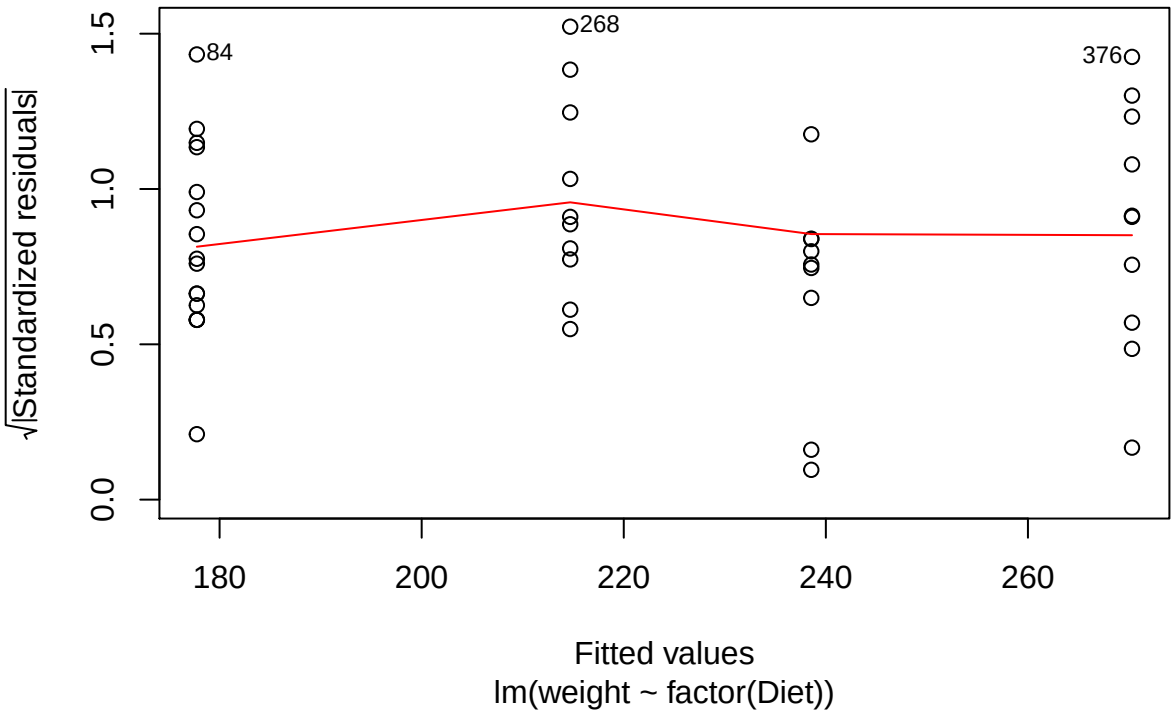
- Random allocation to treatments;
- Independence within treatment-levels;
- Equal variance;
- Normally distributed residuals.

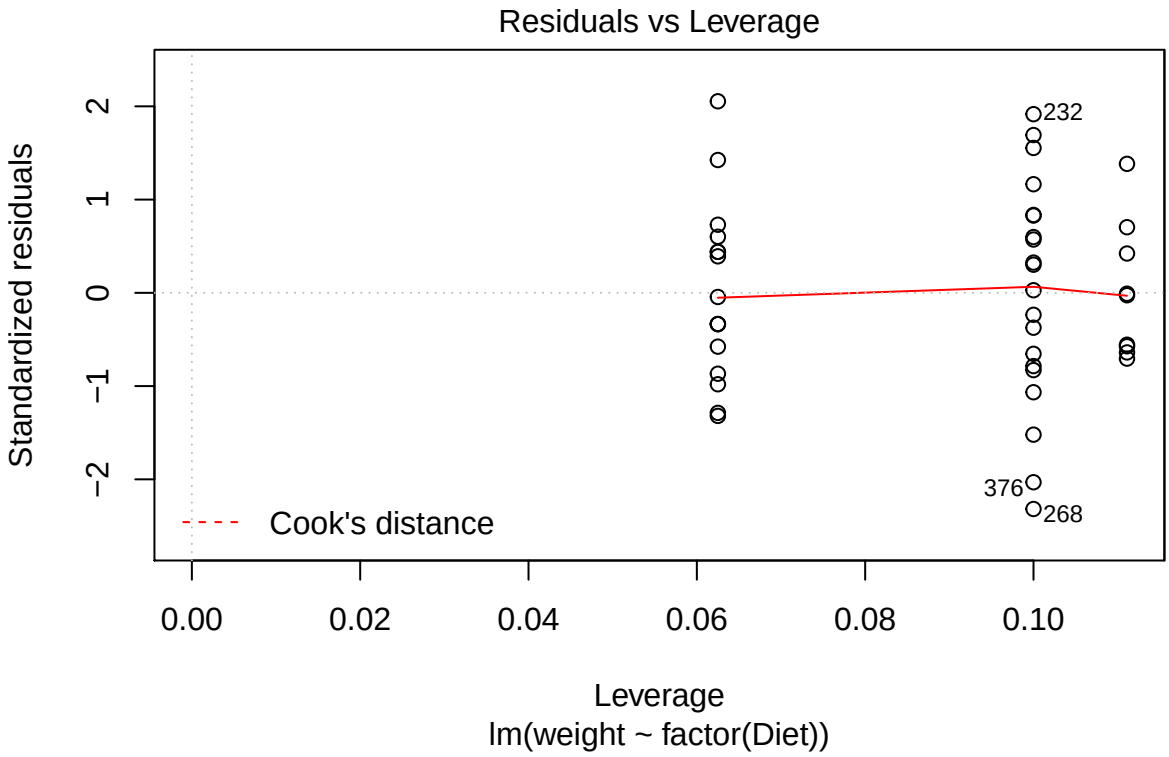
Based on our discussion, earlier, we're happy with the first two assumptions. We check the latter two by visualizing the residuals. We do this with the `plot()` function, where we `plot` the output from our model:

```
plot(ch21.lm)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(weight ~ factor(Diet))
84
376
268
−2
−1
0
1
2


Scale−Location
√|Standardized residuals|
Fitted values
lm(weight ~ factor(Diet))
84
268
376
0.0
0.5
1.0
1.5
180
200
220
240
260



We notice:

- The first plot displays the `Residuals` (y-axis) vs. the `Fitted values`. This plot allows us to check the assumption of equal variance. We see four columns of residuals, which correspond to residuals from the four levels of `Diet`; the placement of the residuals along the x-axis corresponds to the mean value of the treatment from which they came. Notice **(1) that the 'spread' of the residuals is very similar among the four treatments**; the residuals are slightly closer together for `Diet` 4 (at about 240 along the x-axis; notice that this mean is close to our prediction from the `boxplot`), but this is not a worry. Also, **(2) the red line is relatively horizontal**. Both observations suggest that the data meet the assumption of equal variance.
- The second plot (a 'qq plot') allows us to test the assumption of normally distributed residuals. Here, **the points fall beautifully along the dotted line**. This implies that the residuals are normally distributed.
- The third plot presents *Standardized* residuals along the y-axis, and `Fitted values` along the x-axis. Like the first plot, this one allows us to test the assumption of equal variance. However, this plot presents the residuals on a scale that allows us to check this assumption more reliably. Therefore, this plot is considered better than the first to check the assumption of equal variance. Here, we check **(1) whether the residuals for each treatment are roughly centered on the red line**. This is true for 3 of the 4 levels of `Diet`; the residuals for `Diet` 4 (at about 240 along the x-axis) are not terribly well centered, but overall this looks OK; **(2) whether the red line is horizontal**. This red line is relatively horizontal, so we're happy that the data meet the assumption of eaual variance.
- The final plot allows us to check for outliers. We will generally ignore this last plot because we have already checked for outliers in our `boxplot`, above. This final plot is most useful for other types of GLM, e.g., where we have multiple, continuous independent variables (i.e., what used to be called 'multiple regression'). We will use this plot in future analyses where it is helpful.

Now that we're satisfied our data meet the assumptions we can check our results. We'll examine our results from two perspectives, using the functions, `summary()` and `anova()`.

Let's start by looking at a summary of the output from `lm()`:

```
summary(ch21.lm)
```

```
##
## Call:
## lm(formula = weight ~ factor(Diet), data = ch21)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -140.700  -39.700   -1.556   37.250  127.250
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)     177.75      16.00  11.113 6.07e-14 ***
## factor(Diet)2    36.95      25.79   1.433  0.15955
## factor(Diet)3    92.55      25.79   3.588  0.00088 ***
## factor(Diet)4    60.81      26.66   2.281  0.02782 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 63.98 on 41 degrees of freedom
## Multiple R-squared:  0.2541, Adjusted R-squared:  0.1995
## F-statistic: 4.655 on 3 and 41 DF,  p-value: 0.006858
```

Notice that we obtain lots of output. At the top, we find a summary of our model; just below we find a description of the distribution of residuals; below again, we find a summary of the `Coefficients`, and at the bottom we find an assortment of information. We'll focus on the output in `Coefficients`.

Examine the four rows of output in the `Coefficients` section. We notice that the first row is called, `(Intercept)`, and the second through fourth rows are called, `factor(Diet)2`, `factor(Diet)3`, and `factor(Diet)4`. The latter three rows obviously refer to `Diet` levels 2, 3 and 4, but what happened for `Diet` 1? The `lm()` function selects one level of a factor to act as a *reference*, against which the other levels are compared; this reference is termed the `(Intercept)`. **R** selects this reference level (`(Intercept)`) in alpha-numeric order (i.e., it selects that level that comes first either numerically or alphabetically). Therefore, we infer that **R** has selected `Diet` 1 as the `(Intercept)`.

Now that we understand that the `(Intercept)` refers to data from `Diet` level 1, let's look more closely at the output for this row. First, we see an entry in the column, `Estimate`: This entry (177.75) equals the mean value (or the 'fitted value') for `Diet` 1. Notice that it closely matches our guess, above, of 175. (Good guessing!) Next, we see the value, 16.00, in the column `Std Error`. This value equals the standard error (SE) for the mean presented in the column, `Estimate` (177.75). In other words, this first row provides the mean and SE for the `Diet` 1. Next, we see the columns `t value` and `Pr(>|t|)`. These columns provide a test of whether the value in the `Estimate` column differs from the value, zero. We see that the p-value is very small. However, this is not interesting in the least: the small p-value implies that we have strong evidence that the mean `weight` in `Diet` 1 differs from zero. Given that a chick must have some non-zero mass to exist in our universe, it comes as no surprise that we have strong evidence that the mean `weight` in `Diet` 1 differs from zero!

The output in the next three rows differs qualitatively from the output in the first (`(Intercept)`) row. Let's focus on the output in the second row (`factor(Diet)2`). Here, `Estimate` refers to the **difference** between the mean value for `Diet` 2 vs. the `(Intercept)`(i.e., vs. `Diet` 1). Notice that this value is **positive** 36.95: this means that the mean value for `Diet` 2 is 36.95 units larger than the mean value of `Diet` 1. (Note that this value, 36.95, is very close to our predicted difference of 35, made from examining output from `boxplot()`; very reassuring.) Alternatively, we can think of '36.95' as the value we need to add to the `(Intercept)` (i.e., the mean of `Diet` 1) in order to obtain the mean for `Diet` 2; i.e., the mean of `Diet` 2 is `177.75 + 36.96 = 214.71`. **Note that the difference between the mean of `Diet` 2 and the mean of `Diet` 1 represents the effect size on mean weight for shifting between these two diets.** Also note that, if

the difference between Diet 2 and Diet 1 had been negative, this would imply that the mean of Diet 2 was less than that of Diet 1 and we would subtract a value from the mean of Diet 1 to obtain the mean for Diet 2.

Given that the value in the Estimate column for the row, factor(Diet)2, provided the difference between the (Intercept) (Diet 1) and Diet 2 (i.e., the **effect size** for switching between these two types of Diet), what do you think the value in the column, Std. Error represents? This value (25.79) represents the standard error (SE) for the estimate of the difference (i.e., effect size) between these two averages. **This effect size, and its SE, can be reported in your Results.**

Finally, consider the entries in columns, t value and Pr(>|t|), for our same row (factor(Diet)2). These entries are used to test whether the value in Estimate in this row (36.96, in this case) is likely different from zero. In other words, they provide evidence to judge whether we think there's a difference between the mean of Diet 1 (the (Intercept)) and Diet 2. Here, the p-value (0.15955) is pretty large, so we conclude that we have weak evidence for a difference between the mean values for these two types of Diet. **We will re-visit this p-value when we conduct a post-hoc test, below.**

The information in the third and fourth rows are interpreted similarly as we did for the second row, however their output refers to differences between Diet 1 (the (Intercept)) and Diet 3 (third row) and Diet 4 (fourth row). Notice that these differences generally match our predictions of 100 and 75, made using the output from boxplot(); this is reassuring. Also, as predicted from our boxplot, the p-value in the third row provides strong evidence for a difference between the mean values of Diet 1 and Diet 3.

Now that we have thoroughly discussed the Coefficients, let's look at the last row of this output. Here we find the **F-statistic** (4.655), **degrees of freedom (DF)** (3 and 41), and an overall **p-value** (0.006858). These values pertain to an overall test of whether mean values of weight differ between levels of Diet. Note that the p-value is around 0.005; in lecture (videos), I have suggested that p-values of this magnitude constitute 'strong' or 'substantial' evidence for an effect of Diet upon weight. However, this p-value does not indicate which levels of Diet likely differ from which; we will perform a post-hoc test, below, to obtain evidence for differences in weight among levels of Diet. **Note that you would report all values on this line of output (F-statistic: 4.655 on 3 and 41 DF, p-value: 0.006858) in your Results.**

Far above, we said we'd examine our results from two perspectives. We've examine results using summary() and now we will examine results using anova():

```
anova(ch21.lm)
```

```
## Analysis of Variance Table
##
## Response: weight
##              Df Sum Sq Mean Sq F value   Pr(>F)
## factor(Diet)  3  57164 19054.7  4.6547 0.006858 **
## Residuals    41 167839  4093.6
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The function, anova() is multi-purpose; i.e., it will do different things, depending on what kind of object(s) you provide the function. Here, we provided the output from our model using lm() and anova() provides an ANOVA table for these results.

Compare the output from summary(ch21.lm) to that from anova(ch21.lm). Notice that the p-values are identical, as are the F-values and degrees of freedom. From the output of anova() we obtain values for the Sum of Squares (Sum Sq) and Mean Square (Mean Sq) for between-group variation (row with factor(Diet)) and within-group variation (row with Residuals). These values summarize the calculations used to obtain the F value: we obtain the Mean Sq for each row by dividing the Sum Sq by Df, and we obtain the F value by dividing the Mean Sq for among-group variation by the Mean Sq for within-group variation. The value of F and the Df collectively determine the p-value (Pr(>F)). Please see the video lectures for discussion of how Sum Sq and Df are calculated.

OK, we're making progress! We've determined that we have strong evidence for an effect of `Diet` on `weight`, and we need to conduct a post-hoc test to understand this result further. We'll do that in a moment. First, I want to highlight one other bit of output from `ch21.lm`. We can extract the estimate of the residual (i.e., within-group) variation, expressed as a **standard deviation**. We do this like:

```
sigma(ch21.lm)
```

```
## [1] 63.98159
```

I want to note two things:

- We find this same value (63.98) in the output from `summary(ch21.lm)`, described as `Residual standard error`. This is an unfortunate term in `summary(ch21.lm)` because the value, 63.98, is not a standard error! This value (63.98) equals the **standard deviation** for the residual (i.e., within-group) variation.
- We can use this value when we perform a power-analysis. Power analyses require an estimate of residual variation, and we can use this value for this purpose (if we wanted to perform a power analysis.)

Stage 4: Obtaining evidence for differences between groups via post-hoc test (Tukey test)

Our next step is to further understand differences in `weight` among levels of `Diet`. We will use three functions from the library, `emmeans`, for our analyses. If you do not have this library installed, you can install it with `install.packages("emmeans")`.

We begin by opening the library:

```
library(emmeans)
```

Our post-hoc test will involve three steps (and three functions!). First, we will calculate the mean values for each group that we want to compare. Second, we will compare the mean values among levels of `Diet`; this will provide estimates of effect size, SE's for the effect sizes, and p-values for the comparisons among means. Third, we will further characterize the effect sizes by calculating their 95% Confidence Intervals.

Let's begin by calculating the mean values for each level of `Diet`:

```
ch21.emmeans <- emmeans(ch21.lm,"Diet")
```

What is happening here? We provided the function, `emmeans()` (found in the `emmeans` library) with two bits of input: `ch21.lm`, which contains the output from our model, and the name of the `factor` in our model for which we wish to obtain mean values (`Diet`). **Note that `emmeans` works with the model output for its calculations, not the original dataset (`ch21`).** We stored the output of `emmeans` in the object, `ch21.emmeans` (we continue to use our useful nomenclature). Let's look at the output from `emmeans`:

```
ch21.emmeans
```

```
##  Diet emmean   SE df lower.CL upper.CL
##  1       178 16.0 41      145      210
##  2       215 20.2 41      174      256
##  3       270 20.2 41      229      311
##  4       239 21.3 41      195      282
##
## Confidence level used: 0.95
```

This output is extremely useful. **Here, we find the mean value for each group, along with SE's and 95% CI's for each mean; we should report these values in our Results.** Compare these mean values to those we calculated using the `Coefficients` in the output of `summary(ch21.lm)`, above. You should see that they match (we did the calculations for `Diet`1 and `Diet` 2). Also notice that the SE for the mean of `Diet` 1 is the same as in `summary(ch21.lm)` (recall that mean for `Diet` 1 was termed the `(Intercept)` in the output of `summary(ch21.lm)`, above). It is comforting that these value match.

Now notice something perhaps unexpected in the output: the SE values for the means of `Diet` 2 & 3 are identical (20.2), whereas the distributions of these data are not identical in our boxplot - i.e., based on the boxplot, we expect the standard deviation to differ between `Diet` 2 and `Diet` 3, and therefore expect the SE's for their means to differ, too. What's going on? Recall that `emmeans` uses the *model output*, not the original data, for calculations. When `emmeans` calculates a standard error it uses the standard deviation of the residuals **from the model** to calculate SE (remember `SE = sd / sqrt(n)`). (This approach makes sense because our analyses assumes that variance is equal among groups.) We obtained this value of sd using `sigma(ch21.lm)`, above, which equaled 63.98159. Given that sd is the same for all levels of `Diet`, we will obtain the same SE for levels that have the same sample size. As it turns out, the sample sizes for `Diet` 1 to 4 equal 16, 10, 10, and 9. Therefore, the SE is the same for the means of `Diet` 2 & 3 because they have the same sample size (10). In fact, let's do the calculation: `63.98159 / sqrt(10) = 20.23276`, which is what we obtain from `emmeans`. Finally, note that the SE of `Diet` 1 and 4 are smaller and larger than this, respectively, because their sample sizes are larger and smaller, respectively (sample size of `Diet` 1 is 16; sample size of `Diet` 4 is 9).

Now that we have calculated these mean values and their SE's, we'll compare them using the `pairs()` function (also in the `emmeans` library):

```
ch21.pairs <- pairs(ch21.emmeans)
```

Notice that, to run `pairs`, we simply provide it the output from the `emmeans` function. We also stored the output in a sensibly-named object (`ch21.pairs`).

Let's look at the output:

```
ch21.pairs
```

```
##  contrast estimate   SE df t.ratio p.value
##  1 - 2       -37.0 25.8 41 -1.433  0.4868
##  1 - 3       -92.5 25.8 41 -3.588  0.0047
##  1 - 4       -60.8 26.7 41 -2.281  0.1193
##  2 - 3       -55.6 28.6 41 -1.943  0.2264
##  2 - 4       -23.9 29.4 41 -0.811  0.8487
##  3 - 4        31.7 29.4 41  1.080  0.7036
##
## P value adjustment: tukey method for comparing a family of 4 estimates
```

We will discuss several columns from this output:

- The column, `contrast`, indicates which levels of `Diet` being compared. For example, the first row provides information for the comparison between the means of `Diet` 1 vs. `Diet` 2. Notice that it shows `1 - 2`, which indicates that the difference between these two means is calculated by subtracting the mean of `Diet` 2 from the mean of `Diet` 1. Recall that the mean of `Diet` 2 was greater than the mean of `Diet` 1; therefore we expect this difference (`1 - 2`) to be negative.
- The column, **`estimate` provides the effect size** for each comparison between levels of `Diet`. This is calculated as described, immediately above, when we discussed the `contrast` column. For example, notice that, as expected, the difference between the mean of `Diet` 1 and `Diet` 2 is negative. Also notice that we have seen this value before (although it was positive when we first saw it because it was calculated differently). We found the same value in the output of `summary(ch21.lm)`, above (reported as 36.95). Hence, once again, we find output that matches the output from `summary(ch21.lm)` (and our prediction from observing the boxplot). The major difference from the output of `summary(ch21.lm)` is that the output from `pairs(ch21.emmeans)` provides more combinations of comparisons among levels of `Diet`. **We will report the effect sizes in the column `estimate` in our Results.**
- The column `SE` provides the standard error for the effect sizes displayed in the previous column, `estimate`. **We will also report these SE's of the effect sizes.** Notice, once again, that these SE's match the output from `summary(ch21.lm)`.
- the column `df` provides the degrees of freedom used to make the comparison between the two mean

values. **It is a good idea to report the `df` in Results.**

- The column `t.ratio` provides the test-statistic for the comparison between the means listed in the column, `contrast`. These values have the same meaning as the values in the column `t.value` in the output of `summary(ch21.lm)`, and, once again, they match between the outputs. **It is a good idea to report the `t.ratio` in Results.**
- Finally, the column `p.value` provides the p-value for the comparison between the mean values indicated in the column `contrast`. Notice, however, that **this time the p-values from the output of `pairs()` do NOT match the p-values in the output of `summary(ch21.lm)`.** Why is this? We find the answer at the very bottom of our output: `P value adjustment: tukey method for comparing a family of 4 estimates`. This comment tells us that `pairs()` altered the p-values using the tukey method (i.e., we used a Tukey test). It did this for an important reason: **the p-values were altered in a way that maintains a Type 1 error rate of 5%; i.e., it accounted for our making multiple comparisons.** Notice that the p-values are larger in the output from `pairs()` than they were in the output of `summary(ch21.lm)`. For example, the p-value for the difference between the means of `Diet` 1 and `Diet` 2 equaled 0.15955 in the output of `summary(ch21.lm)`, but was greater (0.4868) in the output of `pairs()`. `pairs()` increased the p-values to make it less likely that we would conclude that a difference occurs between means, to compensate for the fact that we've performed multiple comparisons.

Overall, what do we conclude here? If we focus on the p-values, we see that we only have strong evidence for differences between `Diet` 1 and `Diet` 3, as the p-value equals 0.0047. All other p-values exceed 0.1, and therefore provide only weak evidence for differences.

Recall that p-values provide only limited insight into our results. We obtain greater insight from the effect sizes. We can use the `estimate` column from `pairs()` to obtain a mean effect size. However, it is desirable to obtain 95% confidence intervals for the effect sizes. We do this using the `confint()` function, where we provide it the output from `pairs()`:

```
confint(ch21.pairs)
```

```
##  contrast estimate   SE df lower.CL upper.CL
##  1 - 2       -37.0 25.8 41     -106     32.1
##  1 - 3       -92.5 25.8 41     -162    -23.5
##  1 - 4       -60.8 26.7 41     -132     10.6
##  2 - 3       -55.6 28.6 41     -132     21.0
##  2 - 4       -23.9 29.4 41     -103     54.9
##  3 - 4        31.7 29.4 41      -47    110.5
##
## Confidence level used: 0.95
## Conf-level adjustment: tukey method for comparing a family of 4 estimates
```

Notice that all of the output here matches the output from `pairs()`, except we now have 95% CI's for the effect sizes, instead of a `t.ratio` and p-value. Also, note the comment at the base of the output, which states `Conf-level adjustment: tukey method for comparing a family of 4 estimates`. This indicates that the 95% CI's were adjusted (made larger) for the same reason that the p-values were increased in the output of `pairs()`.

How might we interpret these 95% CI's? Let's use the difference between the means of `Diet` 1 and `Diet` 3 as an example. Here, the 95% CI's range from -162 to -23.5; again, the values are negative because R subtracted a big mean (`Diet` 3) from a small mean (`Diet` 1). It is more important to notice that both ends of the 95% CI are the same sign (i.e.; negative); this means that the 95% CI's do not cross zero, so 'zero' is not a very plausible value for the difference between these means. These 95% CI's imply that we have good reason to think (given assumptions of 95% CI's, as opposed to, say, 99% CI's) that the mean of `Diet` 3 is greater than that for `Diet` 1 by as little as 23.5 and as great as 162. Is this effect size biologically important? That's hard to judge without knowing more about the reasons for performing the experiment. For example, imagine that `Diet` 3 costs more than `Diet` 1: a farmer might only be interested in the difference between `Diet` 1 and `Diet` 3 if increase in growth under `Diet` 3 offsets its additional cost. If the minimum 'plausible' increase in `weight` in `Diet` 3 (i.e., lower 95% CI, 23.5) is greater than the amount of increased `weight` needed to offset the cost,

farmers may be excited by these results.

It is a good idea to report 95% CI's for the effect sizes and interpret your results in terms of the 95% CI's, as suggested, above. If you report the 95% CI's reported from `confint()`, you should indicate that they have been adjusted (Tukey method) to account for multiple comparisons.

For practice, let's consider another example. Consider the contrast in the first row: `1 - 2`. Here, the 95% CI's range from -106 to 32.1. This implies that, relative to `Diet` 1, `Diet` 2 may plausibly increase mean `weight` by 106, or decrease in mean `weight` by 32.1, and anything in-between. (Note that the value 0 lies between these values, implying that 'no difference' is plausible - this explains why the p-value is so large for this comparison). Our job now would be to interpret these values at the end of the 95% CI's in terms of their 'biological' significance: if `Diet` 2 increased weight by 106, relative to `Diet` 1, would this be interesting and useful to us? We similarly interpret the biological / practical importance if `Diet` 2 decreased `weight` by 32.1, relative to `Diet` 1. We do not do that, here, because we do not know enough about the background for this experiment to do so. Remember that these 95% CI's (-106 to 32.1) are not hard limits; i.e., values lying just outside the limits (e.g., -107) are also plausible. Also, this range of 'plausible' values is contingent upon the assumptions of 95% CI's; we would have a wider range of 'plausible' values if we considered instead, say, 99% CI's.

Stage 5: Reporting your results

Please refer to the (video) analysis of the `ChickWeight` dataset (by 1-Factor GLM) for guidance to present your results.

An Important Comment Regarding Multiple Comparisons

Statisticians do not all agree on whether p-values and 95% CI's for effect sizes should be adjusted to account for multiple comparisons. We will not discuss the reasons for this debate here. Here is an example where authors believe that accounting for multiple comparisons is a mistake: *Hurlbert, S. H., and Lombardi, C. M. (2012), "Lopsided Reasoning on Lopsided Tests and Multiple Comparisons," Australian & New Zealand Journal of Statistics, 54, 23–42* https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-842X.2012.00652.x.

If, when implementing pairwise comparisons using the `emmeans` library, you do not wish to correct p-values for multiple comparisons, you can suppress the p-value adjustment like this:

```
summary(ch21.pairs, adjust = "none")
```

```
##  contrast estimate   SE df t.ratio p.value
##  1 - 2       -37.0 25.8 41 -1.433  0.1595
##  1 - 3       -92.5 25.8 41 -3.588  0.0009
##  1 - 4       -60.8 26.7 41 -2.281  0.0278
##  2 - 3       -55.6 28.6 41 -1.943  0.0589
##  2 - 4       -23.9 29.4 41 -0.811  0.4218
##  3 - 4        31.7 29.4 41  1.080  0.2865
```

Notice that 1) these p-values are smaller than those obtained, earlier, when using the `pairs()` function, and 2) these p-values match the output from `summary(ch21.lm)`.

Similarly, when using the `emmeans` library, you can obtain 95% CI's for effect sizes that have not been adjusted for multiple comparisons like this:

```
confint(ch21.pairs, adjust = "none")
```

```
##  contrast estimate   SE df lower.CL upper.CL
##  1 - 2       -37.0 25.8 41    -89.0    15.14
##  1 - 3       -92.5 25.8 41   -144.6   -40.46
##  1 - 4       -60.8 26.7 41   -114.6    -6.97
```

```
##  2 - 3       -55.6 28.6 41   -113.4     2.19
##  2 - 4       -23.9 29.4 41    -83.2    35.51
##  3 - 4        31.7 29.4 41    -27.6    91.11
##
## Confidence level used: 0.95
```

When you report your results, remember to comment upon whether or not values were adjusted for multiple comparisons.