

# 6

## Advancing Your Statistics in R

### 6.1 Getting started with more advanced statistics

Chapter 5 was pretty intense. We introduced statistics and interpretation of R output for four different examples: a  $\chi^2$  contingency table analysis, a *t*-test, a simple linear regression, and a one-way ANOVA. In this chapter we extend the complexity of our models to introduce analyses with more than one explanatory variable. In fact, we focus on models with two explanatory variables: the two-way ANOVA and the ANCOVA. Our emphasis, as usual, remains on how to accomplish these in R, rather than on the details of the statistical methods. As we said in Chapter 5, if you're not familiar with the statistical foundations of these tools, you should take the time to become so. Our goal is to introduce to you the formulation and interpretation of such models in R. And nothing changes about our workflow of *Plot -> Model -> Check Assumptions -> Interpret -> Plot Again*. Except that the benefits of *dplyr* and *ggplot2* may become more apparent.

### 6.2 The two-way ANOVA

Chapter 5 finished with the one-way ANOVA, and the major feature of this model was that the explanatory variable was categorical. Recall that

the explanatory variable was `parasite` and it had four levels. The two-way ANOVA is a logical extension of this, and involves experiments or structured data collection where two explanatory variables are involved. As with the one-way ANOVA, the explanatory variables are, in a two-way ANOVA, both categorical.

The two-way ANOVA thus corresponds to data that have structure in two dimensions. The response variable may vary with both variables. In fact, the most exciting and motivating reason for designing experiments and data collection associated with a two-way ANOVA analysis is that the way in which the response variable varies with one variable may depend on the other variable. This is a statistical interaction. The idea of an interaction is often central to hypotheses involving more than one explanatory variable. The word ‘depends’ is a great one to remember as we move on in the book.

### 6.2.1 THE COW (MOOOOO!) GROWTH DATA

Let’s get some example data and work through its structure to articulate the nature of a hypothesis associated with a two-way ANOVA. We are going to work with growth data for cows. The cows were fed one of three diets: barley, oats, and wheat (do you notice the alphabetical order?). The diets were also enhanced with one of four supplements (control and some funny names again). The important thing is that the data are associated with a fully factorial experimental design, where each combination of diet and supplement was replicated three times. This means that there are three diets  $\times$  four supplements = 12 treatment combinations, each with three cows in them (moooooo! 36 times).

Now, you can (and should) work out the degrees of freedom for the error, each ‘main effect’, and ‘the interaction’. If you have no idea what all the key phrases in that sentence mean, now is a very good time to do a little more reading. Let’s see what we can also glean from a picture of the data.

The name of the data file is `growth.csv` and it is, like the others, available from <http://www.r4all.org/the-book/datasets/>.

Set up a script, and, as usual, *Annotate, clear the decks, make libraries available, and import the data, followed by, for example, `glimpse()`*. We'll call the data `growth.moo`.

Check the data was imported correctly, and check their structure:

```
glimpse(growth.moo)

## Observations: 48
## Variables: 3
## $ supplement (fctr) supergain, supergain, supergain, su...
## $ diet       (fctr) wheat, wheat, wheat, wheat, wheat, ...
## $ gain       (dbl) 17.37125, 16.81489, 18.08184, 15.781...
```

As we expect for a two-way ANOVA, two of the variables are factors and one is labelled 'dbl', indicating numeric. This is to plan. We'll introduce a cool function from *base* R here, called `levels()`. This function allows us to see the levels associated with a factor. It's very handy. We use `levels()` with '`$`', grabbing the column from which we want the information:

```
levels(growth.moo$diet)

## [1] "barley" "oats"   "wheat"

levels(growth.moo$supplement)

## [1] "agrimore" "control"  "supergain" "supersupp"
```

Pay attention to the order of things. One of the things we can immediately see is that the supplements include a control level, but the agrimore level is in front of it alphabetically. Remember the reference level? We probably want the control to be the reference level. Here we employ that nifty function `relevel()`, which, for just this purpose, creates a new reference level by re-ordering the factor. We use it with the `mutate()` function from *dplyr* as follows:

```
# relevel the supplement column
growth.moo <-
  mutate(growth.moo,
    supplement = relevel(supplement, ref="control"))

# check it worked
levels(growth.moo$supplement)

## [1] "control" "agrimore" "supergain" "supersupp"
```

How cool! We've overwritten the supplement column in the imported data, forcing the control to be the reference. This is good. Can you guess what level of diet is the reference? That's right... Oats.

Now we are ready to make a fancy figure of the data. This is quite fun. We'll use *dplyr* to calculate the mean of the growth rates for each of the 12 combinations, and then use *ggplot()* to plot them. What we are producing is an elegant interaction plot. So, first, use *dplyr* to get means. Second, get these summary statistics onto an informative plot. Let's go.

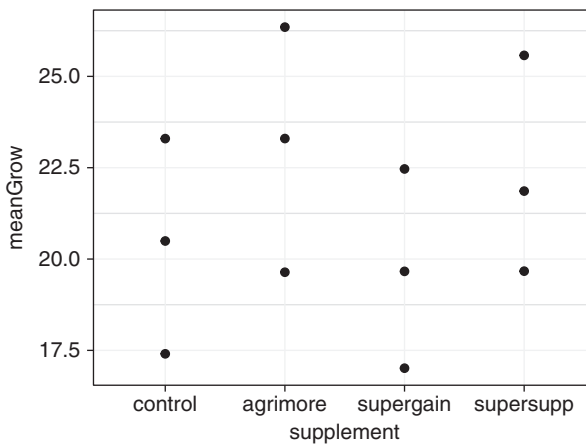
### 6.2.2 STEP 1: *dplyr* SUMMARY DATA

Let's stick with the piping syntax and recall that our data frame is called `growth.moo`. Let's also note that we have two grouping variables, both of which we need to give to the *group\_by()* function:

```
# calculate mean and sd of gain for all 12 combinations
sumMoo <- growth.moo %>%
  group_by(diet, supplement) %>%
  summarise(meanGrow = mean(gain))

# make sure it worked
sumMoo

## Source: local data frame [12 x 3]
## Groups: diet [?]
##
##      diet supplement meanGrow
##      (fctr)      (fctr)      (dbl)
## 1  barley   control  23.29665
## 2  barley  agrimore  26.34848
## 3  barley  supergain  22.46612
## 4  barley  supersupp  25.57530
## 5    oats   control  20.49366
## 6    oats  agrimore  23.29838
## 7    oats  supergain  19.66300
## 8    oats  supersupp  21.86023
## 9   wheat   control  17.40552
##10   wheat  agrimore  19.63907
##11   wheat  supergain  17.01243
##12   wheat  supersupp  19.66834
```



**Figure 6.1** Cow weight gain, unadorned.

### 6.2.3 STEP 2: `ggplot()` INTERACTION PLOT

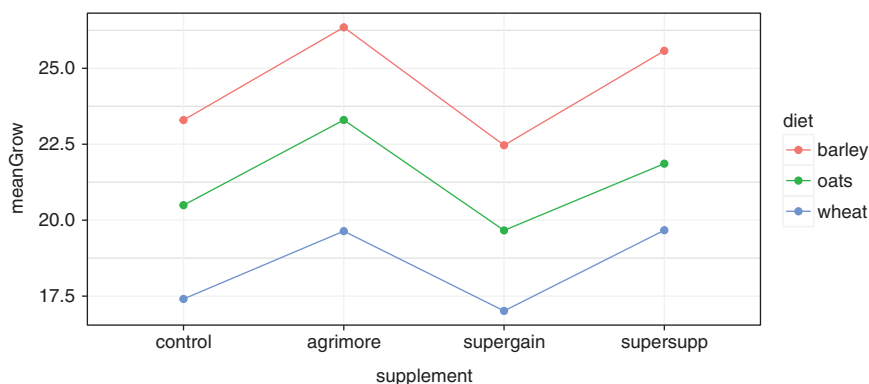
OK. Now for the fun bit. We are going to first add the points from `sumMoo` to a figure (Figure 6.1), and specify that the  $x$ -axis is `supplement`. The logic of this comes from thinking about this statement, to which we will return:

The effect of supplement type on cow weight gain depends on the diet.

```
ggplot(sumMoo, aes(x = supplement, y = meanGrow)) +
  geom_point() +
  theme_bw()
```

OK. The next step in this figure-making process involves connecting the dots (Figure 6.2). Our objective is to connect them according to diet, and perhaps to ask for colours to correspond to diet. We do this in the original aesthetics, specifying both `colour = diet` and `group = diet`, and by asking for another layer, of lines:

```
ggplot(sumMoo, aes(x = supplement, y = meanGrow,
                   colour = diet, group = diet)) +
  geom_point() +
  geom_line() +
  theme_bw()
```



**Figure 6.2** Does the effect of supplement type on cow weight gain depend on the diet?

#### 6.2.4 INTERPRETING THE FIGURE: BIOLOGICAL INSIGHT

Sweet mother of cows, milk! What might we conclude from Figure 6.2? Do you think that the effect of supplement type on cow weight gain depends on the diet? One way to think about this is to first look at the pattern that the supplements generate for the oat diet. Then, ask yourself, does this pattern differ for the other diets, or is it the same? Geometrically, are the effects of supplement generating parallel patterns among the diets, or not? If your answer is that the patterns are the same and the lines are roughly parallel, you have probably guessed that there is *no interaction*. The effect of supplement type on cow weight gain *does not* depend on the diet.

This does not mean that there is no effect of diet. In fact, we can see that, on average, if you are a cow, eating barley rocks the world. This also does not mean that there is no effect of supplement; two of them seem to improve growth no matter what the diet!

Together, the similarity of all the patterns, and the evidence that both variables seem to have an effect, suggests an additive rather than interactive set of effects. Grand. Now we are ready to build the model.

### 6.2.5 CONSTRUCTING THE TWO-WAY ANOVA

At this point, let us formally specify the null hypothesis we are testing. Wait! We have. But here we go again. The effect of supplement type on cow weight gain *does not* depend on the diet. We wish to emphasize that if one designs an experiment with two factors, often one *is* interested in the interaction—whether the effect of one variable on the response variable depends on a third. We note too that the null hypothesis is thus the ‘additive’ model. The alternative is that there is an interaction. All of this means that we want to fit a model *with* the interaction. As we’ll see, this allows us to test formally between the additive and interaction (non-additive) alternatives.

To build the model, we use a trick common to most statistical programming languages. We use a `*` symbol between the two explanatory variables. Specifically, if we write `diet * supplement`, it will expand the combination of variables to read `diet + supplement + diet:supplement`, such that we have an a (main) effect of diet, a (main) effect of supplement, and the interaction (`:`) between diet and supplement.

As with all linear models, we use the `lm()` function, specifying this trick formula and the data frame:

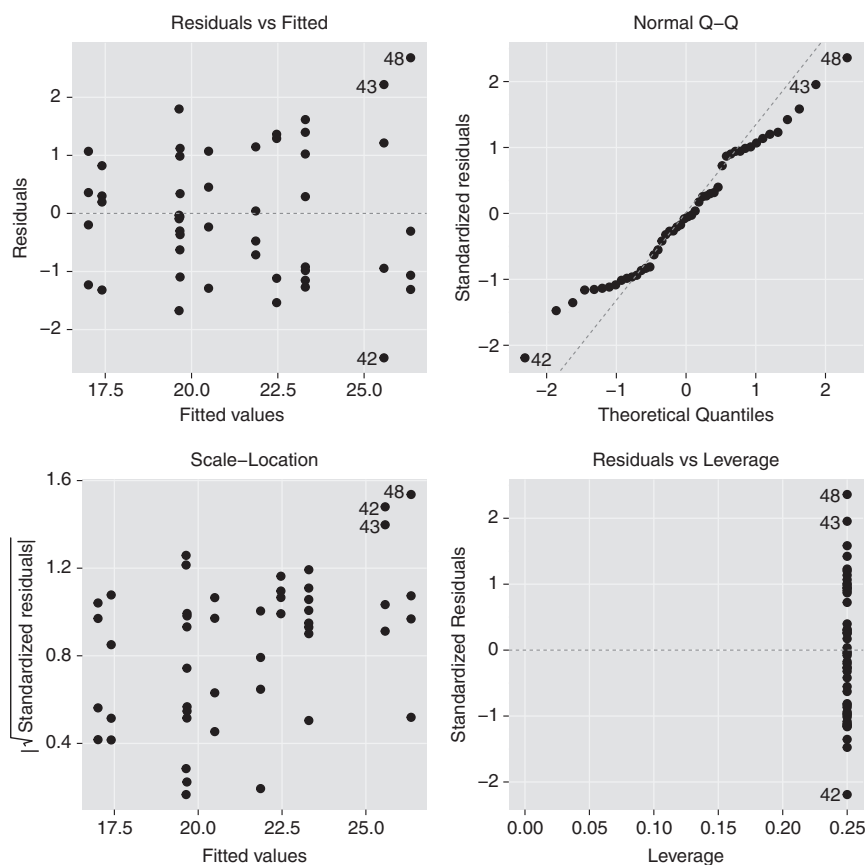
```
model_cow <- lm(gain ~ diet * supplement, data = growth.moo)
```

We will come back later to showing you that the expansion of `*` worked. But first, we must check the assumptions! Which are the same as they were for the regression and the one-way ANOVA. So let’s build those four plots:

```
autoplot(model_cow, smooth.colour = NA)
```

### 6.2.6 EXAMINE THE ASSUMPTIONS

These plots look OK (Figure 6.3), though they aren’t perfect. The residuals vs fitted values (top left) do not show any kind of pattern to suggest our model is inappropriate. In fact, since we built a full model that includes the interaction (a very flexible model), this plot really should be OK. The



**Figure 6.3** The diagnostic plots for the cow weight gain two-way ANOVA.

scale–location plot (bottom left) has almost no pattern, and as you’ll see later in Chapter 7, what we see here is nothing to worry about. There also don’t seem to be any serious outliers either (bottom right).

The normal Q–Q plot (top right) is not super-great. It isn’t disastrous either. The positive and negative residuals tend to be smaller in magnitude than they should be. If you are curious, this arises because the tails (the ‘ends’) of the residuals’ distribution are ‘squashed’ towards its middle, relative to what we expect. This is a rather unusual pattern. It is almost as though someone made up these data, and chose to do this in a strange way. . . The good news is that a departure from normality like this



shouldn't mess up our stats *too* much. The general linear model is actually quite robust to moderate deviations from normality, so let's keep going. This is a book about using R, after all.

## 6.2.7 MODEL OUTPUT AND MORE BIOLOGY

Now we're ready for the fun and games. Remember, there are two functions to aid in interpretation of a general linear model: **anova()** and **summary()**. **anova()** produces an ANOVA table listing sums of squares, mean squares, *F*-values, and *p*-values. It *does not* perform an ANOVA. **summary()** does many things, as you might recall. When provided with a model object from **lm()** as its argument, **summary()** returns a table of coefficients (slopes and intercepts), standard errors, and *t*-values.

Let's start with **anova()**:

```
anova(model_cow)

## Analysis of Variance Table
##
## Response: gain
##
##          Df Sum Sq Mean Sq F value    Pr(>F)
## diet      2 287.171  143.586  83.5201 2.999e-14 ***
## supplement 3  91.881   30.627  17.8150 2.952e-07 ***
## diet:supplement 6   3.406    0.568   0.3302  0.9166
## Residuals 36  61.890    1.719
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

As we promised, using **diet\*supplement** has indeed expanded to provide three rows, corresponding to the variation explained by diet, the supplement, and the interaction. Let's walk through the logic, biological and statistical, of this table.

### The ANOVA table

The ANOVA table presents a sums-of-squares analysis-of-variance table. This is interpreted as follows. The first line shows the variation explained by diet, captured in the Mean Sq value of 143.586. Having captured this

variation, we then see an estimate of the variation explained by the supplement, captured in the Mean Sq value of 30.627. Finally, having captured both of these estimates of variation, we finally ask whether any additional amount of variation is explained by allowing the effect of supplement to vary by diet. The answer is captured in a tiny (0.568) Mean Sq. The  $F$ -statistics for each of these parts of our explanation are calculated by dividing each Mean Sq by the residual Mean Sq (last line), generating two significant  $p$ -values for the main effects and a non-significant value for the interaction.

Because we conducted, and are analysing, an experiment, we had a specific hypothesis in mind—that the effect of supplement on weight gain depended on diet. Testing this hypothesis is embodied in the `diet:supplement` row. This table reveals that there is no additional, significant variation in weight gain explained by allowing the effect of supplement to vary by diet ( $F = 0.33$ ;  $df = 6, 36$ ;  $p = 0.92$ ). Maybe this was obvious from the figure we made, but now we have the statistical evidence we need to support this insight. Here is how you might report this result:

We tested the hypothesis that the effect of diet supplement on bovine weight gain depended on cereal diet. We found no evidence to support the presence of an interaction between diet and supplement ( $F = 0.33$ ;  $df = 6, 36$ ;  $p = 0.92$ ).

Hopefully, the explanations along the way have been helpful. The ANOVA table is all about explaining variation associated with main effects and interactions.

We note that the order of the variables in the ANOVA table in this example does not matter, because the data are from a designed experiment where each of the explanatory variables are independent by design. The data are from a fully factorial, balanced experiment. If a dataset does not possess these properties (e.g. in most observational studies), then the order of testing in the ANOVA table *does* matter. It matters because the mean squares,  $F$ -values, and ultimately the  $p$ -values will depend on the order of testing. This is not the place to discuss this idea in detail, but we will

repeat this important warning (it would be irresponsible not to): unless your data are balanced and orthogonal, the order of testing in an ANOVA table matters. This means you should not blindly rely on the output of `anova()` in these situations. There is a lot of good reading to do on this and we encourage you to seek some of it out if you're not sure what we're talking about (see Appendix 2 and, in particular, the treatment in *An R Companion to Applied Regression* by J. Fox).

We could actually stop the analysis here. We have analysed data from an experiment designed to test formally a hypothesis about whether the effect of supplements on cow weight gain depends on the diet. R has served us well, and provided an answer.

### The summary table

However, we may want more. Let's not forget that there is a summary table. This table is quite large. Let's look at it and then make some recommendations about how to proceed, if one feels compelled:

```
summary(model_cow)

##
## Call:
## lm(formula = gain ~ diet * supplement, data = growth.moo)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.48756 -1.00368 -0.07452  1.03496  2.68069
##
## Coefficients:
##              Estimate Std. Error t value
## (Intercept)    23.2966499   0.6555863   35.536
## dietoats       -2.8029851   0.9271390   -3.023
## dietwheat      -5.8911317   0.9271390   -6.354
## supplementagrimore  3.0518277   0.9271390    3.292
## supplementsupergain -0.8305263   0.9271390   -0.896
## supplementsupersupp  2.2786527   0.9271390    2.458
## dietoats:supplementagrimore -0.2471088   1.3111726   -0.188
## dietwheat:supplementagrimore -0.8182729   1.3111726   -0.624
## dietoats:supplementsupergain -0.0001351   1.3111726    0.000
## dietwheat:supplementsupergain  0.4374395   1.3111726    0.334
## dietoats:supplementsupersupp -0.9120830   1.3111726   -0.696
## dietwheat:supplementsupersupp -0.0158299   1.3111726   -0.012
##
##              Pr(>|t|)
## (Intercept)    < 2e-16 ***
## dietoats       0.00459 **
```

```
## dietwheat                2.34e-07 ***
## supplementagrimore       0.00224 **
## supplementsupergain      0.37631
## supplementsupersupp      0.01893 *
## dietoats:supplementagrimore 0.85157
## dietwheat:supplementagrimore 0.53651
## dietoats:supplementsupergain 0.99992
## dietwheat:supplementsupergain 0.74060
## dietoats:supplementsupersupp 0.49113
## dietwheat:supplementsupersupp 0.99043
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.311 on 36 degrees of freedom
## Multiple R-squared:  0.8607, Adjusted R-squared:  0.8182
## F-statistic: 20.22 on 11 and 36 DF,  p-value: 3.295e-12
```

The table looks rather scary. We argue that in fact this table, as is, is probably not what you want. It is in fact interpretable. We've set you up to understand at least parts of it. Recall our discussion of treatment contrasts in Chapter 5. And recall above our emphasis on alphabetical ordering of both explanatory variables. Together, these two points should help you to interpret at least the first three lines.

The '(Intercept)' is a reference point for the table. And it is a combination of levels, one for each variable. Can you figure it out? It has to be barley-control because barley is the first diet in the alphabetized list, and we used `relevel()` to force `Control` to be the reference level. You can find the value 23.29 on the graph, in fact in the top left.

Great. Now, recall that all the other estimates are differences between this reference level and whatever is labelled in the row. So, rows two and three specify the oats and wheat, indicating that these values are the differences between barley-control and oat-control, and between barley-control and wheat-control, respectively. Again, you can prove this to yourself with the numbers and the graph.

But, is this what you really want to do? We think not. In fact, we think that if you had designed such a large, multi-factorial experiment, you probably had some *specific, a priori* hypotheses in mind (we really hope you did). In stats speak, a priori hypotheses are typically described as *contrasts*. A contrast is just a difference between the means of two levels, or some

combinations of levels. You know about treatment contrasts, but you can use R to examine the statistical significance of any contrast you like (even nonsensical ones).

You can, in fact, specify contrasts before you fit the model, and just estimate the subset for which you designed the experiment. This is very good practice. We can recommend several packages to help you work with contrasts. The *contrast*, *rms*, and *multcomp* packages all contain excellent facilities for specifying custom contrasts. Furthermore, the *multcomp* package has numerous built-in post-hoc (e.g. Tukey) tests that you may be interested in learning how to use. We think a priori contrasts are much better. This stuff is a bit tricky, though, so it is probably another find-an-experienced-friend scenario.

## 6.2.8 STATS BACK TO GRAPHICS

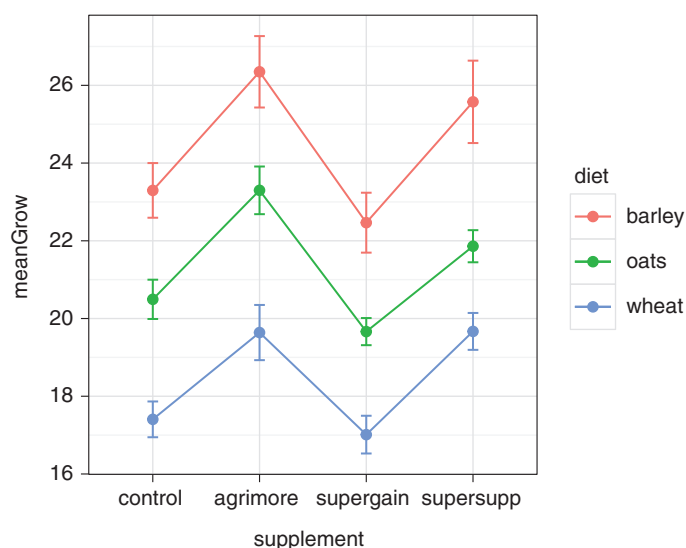
One might be tempted to conclude that the figure of the mean weight gain that we've already made is sufficient to understand the cow diet data. However, we really need to understand the variation around each mean value. There are several ways to do this, but we'll take this opportunity to show you a bit more *dplyr* and *ggplot* loveliness.

First, we return to the use of *dplyr*, where we calculated `meanGain`. We want to also include an estimate of the standard error. We do this via the following code:

```
# calculate mean and sd of gain for all 12 combinations
sumMoo <- growth.moo %>%
  group_by(diet, supplement) %>%
  summarise(
    meanGrow = mean(gain),
    seGrow = sd(gain)/sqrt(n())
  )
```

Note that the standard error of the mean is given by the standard deviation divided by the square root of the sample size, and *dplyr* provides a function `n()` that counts the rows in each group. Be careful with this, however. If you have missing values, `n()` will still count these rows.<sup>1</sup>

<sup>1</sup> Here is an advanced tool to alleviate this problem: `sd(gain)/sqrt(sum(!is.na(gain)))`... check the help files and internet!



**Figure 6.4** A quite nice graph of the cow growth data, finishing off our two-way ANOVA example.

With this summary table, we can now add three layers to the figure: points, lines, and error bars (Figure 6.4). To do this, we introduce `geom_errorbar()`, which has its own aesthetics: the lower and upper limits of the vertical lines that we call error bars. These limits are called `ymin` and `ymax`. Each vertical line is made by placing a line between these limits, passing through `meanGrow`. The magic here is that we've got 12 estimates of mean weight gain and 12 estimates of the standard error. **ggplot2** is going to add them ALL at once, correctly on each point. Note we also ask for the hat on the error bar to be small, using `width`. Did you know that error bars wear hats? Superb:

```
ggplot(sumMoo, aes(x = supplement, y = meanGrow,
                   colour = diet, group = diet)) +
  geom_point() +
  geom_line() +
  geom_errorbar(aes(ymin = meanGrow - seGrow,
                    ymax = meanGrow + seGrow), width = 0.1) +
  theme_bw()
```

## 6.3 Analysis of covariance (ANCOVA)

We hope you've taken a deep breath. And/or had a few more biscuits/cookies. Here we go for our final linear-model example. It is unique in that it combines a categorical explanatory variable with a continuous explanatory variable. What are we up to? We are combining regression and one-way ANOVA! Yes we are.

The dataset we will use is `limpet.csv` and is originally from Quinn and Keough's 2002 book *Experimental Design and Data Analysis for Biologists*. As usual, the data file is available at <http://www.r4all.org/the-book/datasets/>. The data relate egg production by limpets to four density conditions in two seasons. The response variable ( $y$ ) is egg production (EGGS) and the independent variables ( $x$ 's) are DENSITY (continuous) and SEASON (categorical). Because we are examining egg production along a continuous density gradient, this is essentially a study of density-dependent reproduction.<sup>2</sup> The experimental manipulation of density was implemented in spring and in summer. Thus, a motivation for collecting these data could be 'does the density dependence of egg production differ between spring and summer'?

### 6.3.1 THE LIMPET REPRODUCTION DATA

As with all of our examples, set up a new script. We'll refrain from repeating ourselves about what you should do... you know it already! We'll call the data frame 'limp':

```
glimpse(limp)

## Observations: 24
## Variables: 3
## $ DENSITY (int) 8, 8, 8, 8, 8, 8, 15, 15, 15, 15, 15, 1...
## $ SEASON (fctr) spring, spring, spring, summer, summer...
## $ EGGS (dbl) 2.875, 2.625, 1.750, 2.125, 1.500, 1.87...
```

<sup>2</sup> If you don't come from an ecology background, density dependence is the idea that as the number of mums sharing a food resource increases (e.g. their density goes up), they get a smaller and smaller portion, leading to reduced baby production.



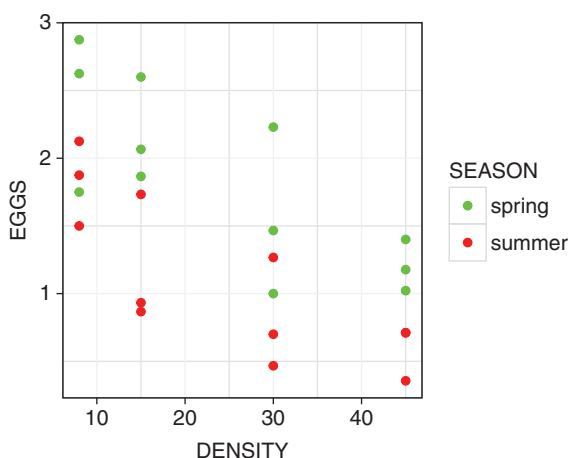
### 6.3.2 ALWAYS START WITH A PICTURE

Let's begin the analysis as we normally do—with a fabulous picture in R.

You should notice that the dataset contains three columns, two that are numeric (EGGS and DENSITY) and a third that is categorical (SEASON). The response ( $y$ ) variable is egg production, DENSITY is the continuous independent variable ( $x$ ), and the values are also classified by the category SEASON. This suggests that a plot of egg production against density, distinguishing the two seasonal categories, is what we need. This is readily achieved in `ggplot()`, isn't it! The code has the same structure as what we introduced in Chapter 4. Make sure you understand how we've used `scale_colour_manual()` and how we've made sure that, oh, we can't resist, spring comes before summer! Ha!

```
# plot window
ggplot(limp, aes(x = DENSITY, y = EGGS, colour = SEASON)) +
  geom_point() +
  scale_color_manual(values = c(spring="green", summer="red")) +
  theme_bw()
```

Let's consider the pattern that we see in this graph (Figure 6.5). There is clearly a decline in egg production with increasing limpet density, and



**Figure 6.5** Always begin analyses with a picture. Our analysis of egg production by limpets in two seasons and at four limpet densities.



it looks as though, for any particular density, there is a tendency for the production to be higher in spring than in summer. So, just from the picture, we have extracted some pretty useful information. Now the challenge is how to test this more formally; in other words, how to make a model for the data.

To think about how to do this, we can start by recalling the equation for a straight line:  $y = b + m \times x$ . On our graph, if we were to describe the relationship between egg production and density by a straight line (ignore the season for the moment), then  $y$  is egg production,  $x$  is density,  $b$  is where the line crosses the  $y$ -axis (i.e. the egg production at 0 density—admittedly an odd concept!), and  $m$  is the slope of the egg production density relationship. This slope represents the change in egg production per unit change in density—that is, the strength of density dependence.  $m$  and  $b$  are the parameters, or coefficients, of the line.

To work out the degrees of freedom for error, you need to observe that there were three observations at each of the four densities in each of the two seasons (i.e. Figure 6.5 has 24 data points). We have data for two different seasons, so don't forget we'll work with a model that is estimating coefficients for two lines. You should also work out guesses of the values of the two intercepts and two slopes (i.e. the four coefficients that are estimated when we make the model).

### 6.3.3 INTERPRETING THE FIGURE—IT'S ALL ABOUT LINES

With the understanding that all straight lines are composed of an intercept and a slope, we can reinterpret our figure, assigning biological meaning to intercepts and slopes. First, we can note that, overall, egg numbers decline with increasing density—that is, there is a negative slope. Second, we can begin to speculate that there is a seasonal difference—that is, that the intercept, the value of egg production at zero density, is different in each season.

Once we start to look at the data plotted on the figure this way, we immediately start to identify the patterns that stand out. But in order to think about testing these, and interpreting what they mean, it is important to

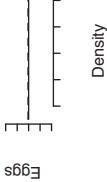
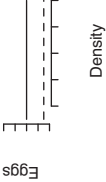
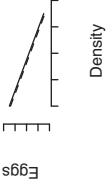
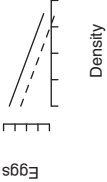
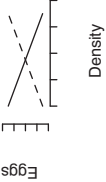
step back and consider all the possible patterns that we might have found, and the different hypotheses, or interpretations, they represent. We can do this using the two concepts of slope and intercept of the lines describing these data. Figure 6.6 shows the range of possibilities, and describes each in verbal terms, graphically, and as an equivalent model in R.

Looking at our data above (Figure 6.5), we can walk through the different hypotheses in Figure 6.6. We might ignore A and B—it seems clear from the data that there is at least a decline, on average, in egg production with density. However, the higher the variation in egg production, the less likely we are to detect such a trend as significant. C is clearly possible; again, more so if the variation in egg production in each season is high. D and E are compelling. In our data, if the slopes are indeed equivalent, scenario D would be the best explanation. However, if the slopes are different and the lines for each season cross (even if they are both negative), then scenario E is a compelling, competing hypothesis. Because we can't actually see whether there are different slopes and intercepts, we can use statistics.

### Let's go a bit deeper

For a number of reasons, we'd like you to understand that the difference in models D and E is the presence (E) or absence (D) of one particular term: an interaction term that specifies that the slopes are different. This interaction is embodied in the following sentence: 'The effect of density on egg production depends on the season.'

Try looking at Figure 6.5 and saying this out loud: 'The effect of density on egg production depends on the season.' If we deconstruct this sentence, we can reveal the features of a line. 'The effect of density on egg production' means the value of the slope(s); 'depends on the season' indicates that these values may be different depending on the season. Of course, if the slope depends on the season, we must also be estimating the effect of the season, in other words estimating intercepts as well as slopes. In fact, what statistical modelling does is to ask whether our data justify specifying more than one intercept and more than one slope. Put another way, we are

	VERBAL HYPOTHESIS	INTERCEPT/ SLOPE	MODEL IN R	GRAPHICAL
A	The number of eggs produced by limpets does not vary with density or season	Common intercept Zero slope(s)	<code>lm(Eggs~1, data=limp)</code>	
B	The number of eggs produced by limpets does not vary with density but is reduced in the summer season (dashed)	Different intercepts Zero slope(s) Parallel horizontal line	<code>lm(Eggs~Season, data=limp)</code>	
C	The number of eggs produced declines with density, but the maximum number of eggs (intercept) and the rate (slope) do not vary with season	Same intercept Same slope Same lines	<code>lm(Eggs~Density, data=limp)</code>	
D	The number of eggs produced declines with density and the number of eggs at density zero (intercept) differs between seasons but the rate (slope) does not vary with season	Different intercepts Same (negative) slope Parallel lines	<code>lm(Eggs~Density+Season, data=limp)</code>	
E	The number of eggs at density zero (intercept) and the rate (slope) vary with season	Different intercepts Different slopes 'Crossing' lines	<code>lm(Eggs~Density*Season, data=limp)</code>	

**Figure 6.6** Verbal, mathematical, R, and graphical interpretations of various hypotheses related to the ANCOVA that translate into specific linear models.

asking: do we get a better description of the data by using more than one intercept and more than one slope?

At this stage, we have a good figure (Figure 6.5) that reveals a pattern and allows us to speculate about a result—some of you might think that there is a common rate of density dependence (slope), while others might see an interaction—there is the distinct possibility that the effect of density on egg production depends on the season.

Before we explore these alternatives, consider one more feature of these data: they were collected as part of a manipulative experiment. Philosophically, we may want to limit the number of competing hypotheses we consider (i.e. A–E) because when one designs an experiment, one usually has a particular hypothesis in mind; we have an *a priori* hypothesis. In the case of these data, if we assume that the researchers were testing whether ‘The effect of density on egg production depends on the season’, then in fact we are not equally interested in all the hypotheses: we are starting with the belief that density does have an effect on egg production, and we are focusing our investigation on whether this effect differs between seasons; that is, we are really interested in whether the data are better described by model E than by model D.

### 6.3.4 CONSTRUCTING THE ANCOVA



To specify a general linear model (regression, ANOVA, ANCOVA), and one that would capture any of the above statistical hypotheses, we continue to rely on the function **lm()**, the workhorse of linear models in R:

```
limp.mod <- lm(EGGS ~ DENSITY * SEASON, data = limp)
```

We have assigned the model returned by **lm()** to the object `limp.mod`. And we have used a formula to specify the relationship between EGGS, DENSITY, and SEASON. As we discussed above, the formula’s right-hand side expression is a sort of shorthand: it specifies, all at once, that we want to include an effect of DENSITY (main effect), an effect of SEASON (main effect), and the potential for the effect of DENSITY to depend on SEASON (interaction). The specification expands to the full model of DENSITY + SEASON + DENSITY : SEASON.

We could specify a model that doesn't include all these things—the other examples in Figure 6.6 illustrate what these would look like. But since we are interested in whether the effect of density depends on season, we need a model where there is, in addition to the effects of density and season on their own, a specific term allowing for their interaction, that is, in which the effect of density on egg production *could* depend on season.

Why do we say 'could'? We are, fundamentally, testing the null hypothesis (Figure 6.6, row E) that this interaction term is not significant: there are no differences in the slopes for each season; there is no extra variation explained by fitting different slopes. The alternative is that, by allowing separate slopes to be fitted, we explain more variation in the data. That's it. Said another way, we are asking if, having explained a certain amount of variation with different intercepts and a common slope (i.e. hypothesis D), do we explain a significant amount of additional variation by allowing separate slopes?

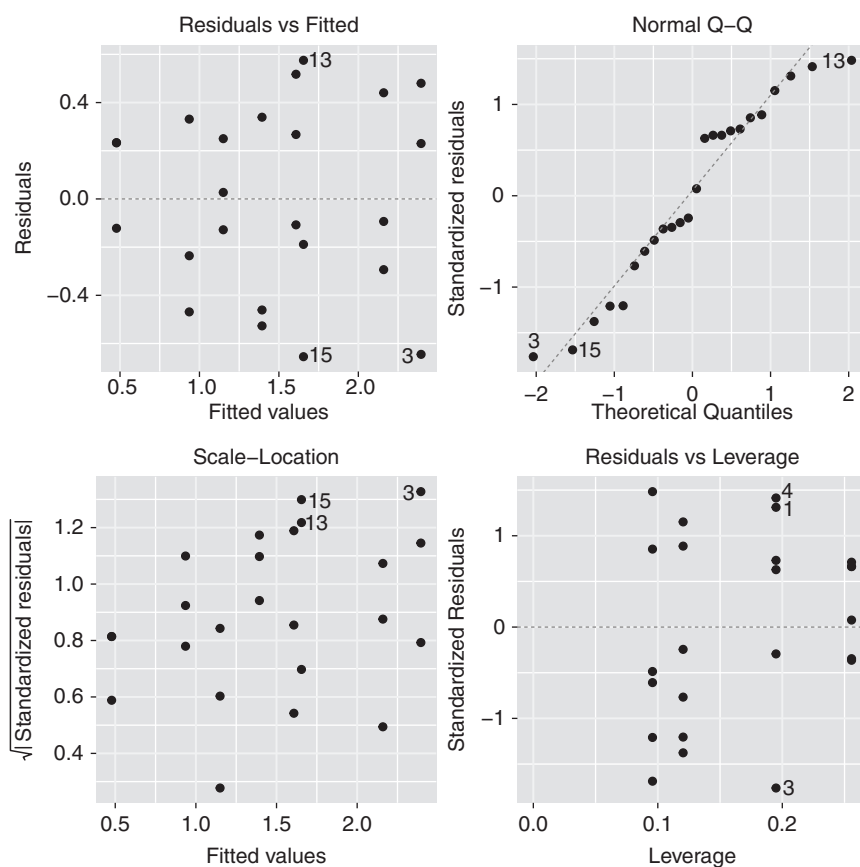
If you run the code above, all the good stuff has been done, and has been quietly collected and organized for you in the `limp.mod` object you specified.

As with all of the amazing R functions you have experienced thus far, by assigning the values returned by `lm()` to an object (`limp.mod`), we can see what has been collected for us; we've not done this for any of the previous examples, so let's have a look:

```
names(limp.mod)

## [1] "coefficients" "residuals"      "effects"
## [4] "rank"         "fitted.values"  "assign"
## [7] "qr"           "df.residual"    "contrasts"
## [10] "xlevels"      "call"           "terms"
## [13] "model"
```

R has stored for us the coefficients (intercept and slope estimates), the residuals, the fitted values, and a number of other values. Peruse the help file for `lm()`, specifically the Value section, to find out what `lm()` can return to you.



**Figure 6.7** Diagnostic graphs for the linear model of the limpet data.

### 6.3.5 ASSUMPTIONS FIRST, SO SAY THE LIMPETS

By now you know what to do, right? Right.

```
autoplot(limp.mod, smooth.colour = NA)
```

The diagnostics are very nice (Figure 6.7). We saw some that weren't so great in the last example, and we'll see some more that are even worse in the next chapter.

We will assume that you, too, find the diagnostics satisfying, illuminating, and downright good. Before we head to our interpretation phase, let's review. We've made an insightful and useful figure representing the

data, which helped us to formulate our ideas about how density and season might affect the egg production of limpets. We've made a model to capture our ideas, and, more importantly, an experimental design. And we've evaluated the core assumptions associated with a general linear model. Now you're ready for interpretation.

### 6.3.6 INTERPRETATION: THE `anova()` TABLE

As discussed throughout so far, we use two functions to aid in interpretation of a general linear model: `anova()` and `summary()`. Together, the resulting two tables help us interpret the relationships between egg production and the combined or independent impact of density and season. Sounds easy? If you know how to interpret the output of other packages, and understand contrasts and statistics deeply, it is. If you've never delved deeply into what a statistical package returns to you, hold on tight! In this example, the outputs both `anova()` and `summary()` are totally and immediately relevant.



Let's start with `anova()`:

```
anova(limp.mod)

## Analysis of Variance Table
##
## Response: EGGS
##              Df Sum Sq Mean Sq F value    Pr(>F)
## DENSITY         1  5.0241    5.0241  30.1971 2.226e-05 ***
## SEASON           1  3.2502    3.2502  19.5350 0.0002637 ***
## DENSITY:SEASON   1  0.0118    0.0118   0.0711 0.7925333
## Residuals       20  3.3275    0.1664
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here we see again a sequential sums-of-squares analysis-of-variance table. This is interpreted as follows. First, we estimate a common slope (DENSITY), using 1 df, and this explains a certain amount of variation,

captured in the Mean Sq value of 5.0241. Having done this, we then estimate different intercepts for the model (SEASON) and, having done so, explain an additional amount of variation (3.2502). Finally, we allow the slopes to vary (DENSITY:SEASON), and this explains an additional 0.0118 units of variation. This is the sequence of explanation.

Because we conducted, and are analysing, an experiment, we had a specific hypothesis in mind—that the effect of density on egg production depends on the season. Testing this hypothesis is embodied in the DENSITY:SEASON row.

This table reveals that there is no additional, significant variation in egg number explained by allowing different slopes for each season (i.e. the *p*-value for the DENSITY:SEASON row is 0.79). We fail to reject the null hypothesis that the slopes are the same, i.e. we have failed to find evidence for the alternative hypothesis. Our result could actually be written:

We tested the hypothesis that the effect of density on egg production in limpets depends on the season in which they are reproducing. We found no evidence for an interaction between density and season ( $F = 0.0711$ ;  $df = 1, 20$ ;  $p = 0.79$ ), indicating that the effects of density and season are additive.

We could stop the analysis there. We have analysed data from an experiment designed to test the hypothesis, and R has again provided an answer. However, we probably want more. Perhaps we want to know what the estimate of egg production at low density is in each season (intercepts; density-independent egg production). We may also want to know what the rate of density dependence is, given that it appears to be common.



### 6.3.7 INTERPRETATION: THE `summary()` TABLE

We can use the summary table to identify all of this:

```
summary(limp.mod)

##
## Call:
## lm(formula = EGGS ~ DENSITY * SEASON, data = limp)
##
```



```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.65468 -0.25021 -0.03318  0.28335  0.57532
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      2.664166   0.234118  11.380 3.45e-10
## DENSITY          -0.033650   0.008259  -4.074 0.000591
## SEASONsummer     -0.812282   0.331092  -2.453 0.023450
## DENSITY:SEASONsummer 0.003114   0.011680   0.267 0.792533
##
## (Intercept)          ***
## DENSITY              ***
## SEASONsummer         *
## DENSITY:SEASONsummer
## ---
## Signif. codes:
##  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4079 on 20 degrees of freedom
## Multiple R-squared:  0.7135, Adjusted R-squared:  0.6705
## F-statistic: 16.6 on 3 and 20 DF, p-value: 1.186e-05
```

The output from `summary()` is more manageable here than it was with the two-way ANOVA. We can see that it has four sections. First, it provides a restatement of your model by repeating your usage of `lm()`—the call. Next it provides the range, interquartile range, and median of the residuals as a quick, additional check of the residuals. These should not be relied upon as a sole diagnostic and should be used with the more substantial plotting methods described above.

The next section of the output is the coefficient table. This is one of the most interesting tables a statistics package can produce associated with an ANCOVA, as it provides, one way or another, insight into the estimates for the lines that you have fitted—these are the coefficients that quantify the important biological relationships. If you have used other packages, you will have seen similar tables. Here is how to interpret the table in R.

First, don't forget that R works alphanumerically. In this current example we should thus expect to see, for example, the season 'spring' reported before the season 'summer', because spring comes before summer. We do like saying that.

Let's start our interpretation from the bottom, where we find some generic statistics and a commonly used estimate of the variance explained by the model,  $R^2$ . The model we have fitted explains 67% of the variation in egg production, and has a significant fit to the data ( $F = 16.6$ ,  $df = 3, 20$ ,  $p < 0.001$ ), leaving a residual standard error of 0.4079 with 20 degrees of freedom. Hopefully, you were expecting 20 degrees of freedom for error: 24 data points and 4 estimated coefficients give  $20 - 4$  error degrees of freedom.

Now for the fun part. First, recall that R behaves alphanumerically. Spring comes before summer. We told you we like saying that. Second, note that R has labelled aspects of rows three and four with the word 'summer'—this should give something away.

Looking closely at the top two rows, we can see two words in the leftmost column under the word 'Coefficients': Intercept and Density. The equation for the line we are fitting is  $\text{EGGS} = \text{Intercept } (b) + \text{slope } (m) \times \text{DENSITY}$ . This should be reassuring. R has told us that here is the estimate of an intercept and a slope. And, we know that it is the intercept and slope estimate for the spring data. Spring comes before summer.

Specifically, the model (equation for a line) for egg production in spring is:

$$\text{EGGS}_{\text{spring}} = 2.66 - 0.033 \times \text{DENSITY}.$$

Looking at rows three and four, we think you may now know what we are supposed to do. You should recall from our previous discussions of treatment contrasts that these will be differences. Specifically, the third line, labelled `SEASONsummer`, is very specifically the difference between the spring and the summer intercept. More biologically, it is the change in egg production that arises from shifting the season from spring to summer. The value is  $-0.812$  eggs.

If this is the difference between the spring and summer intercepts, then adding this number of eggs to the estimate of the intercept for spring should provide our estimate of the intercept for summer.

Likewise, perhaps even logically because of the words ‘summer’ and ‘DENSITY’, the fourth line is the difference between the slopes for spring and summer. More biologically, it is the change in the rate of density dependence that arises from shifting from spring to summer. The value is 0.003. If this is the difference in slopes, then adding this number to our estimate of the slope for spring should provide our estimate for the summer slope. Here is the maths:

$$\begin{aligned}\text{EGGS}_{\text{spring}} &= 2.66 - 0.033 \times \text{DENSITY}, \\ \text{EGGS}_{\text{summer}} &= (2.66 - 0.812) + (-0.033 + 0.003) \times \text{DENSITY}, \\ \text{EGGS}_{\text{summer}} &= 1.84 - 0.03 \times \text{DENSITY}.\end{aligned}$$

We also highlight the  $t$ -values and  $p$ -values in this table. As we covered in Chapter 5, a  $t$ -test essentially evaluates whether two values are different by asking if the difference between the two values differs from zero. Here, the treatment contrasts, comparing for example the difference in the intercepts for each season, makes sense. The  $t$ -value and  $p$ -value reveal whether the season intercepts and slopes differ. If the  $t$ -value is small and the  $p$ -value large, we are unable to reject the null hypothesis that the two values are the same (there is no difference). We can reject this null hypothesis for intercepts, but not slopes.

So, via just the summary table, we can draw the following conclusions. Summer reduces egg production compared with spring, on average, by 0.812 eggs. In summer, the rate of decline in egg production with density was slightly less than in spring (+0.003 eggs/density) but we could not conclude that this change was different from 0. Finally, as a result of this last piece of information, there is no evidence that the effect of density depends on season. Good, the results of the summary table and ANOVA agree, as they should!

Having *a priori* defined the hypothesis we were testing as ‘the effect of density on egg production depends on season’, with an associated null hypothesis that it does not, we find that we cannot reject the null hypothesis.

There is no evidence in these data that the effect of density on egg production depends on season. You never thought you'd know this much about limpets.

### 6.3.8 PUTTING THE LINES ONTO THE FIGURE

Our final instalment of the linear model centres on a robust method for producing a figure that combines the raw data with model fits—that is, the lines estimated by the ANCOVA. The method builds on your understanding that R provides you, in the summary table, with the coefficients for two lines. In contrast to our approach for the simple linear regression (Chapter 5), we present here a very generic method, useful for models of arbitrary complexity (i.e. more than two lines, non-linear models, etc.).

We have used R to fit a model, and the object has stored in it the coefficients for the lines. If you want to see them, use `coef(limp.mod)`. In order to add lines to a graph, we need to create (predict) some 'y' values from a sensible range of 'x' values, with the coefficients defining the new y values. We have to take, for example,

$$\text{EGGS}_{\text{Spring}} = 2.66 - 0.033 \times \text{DENSITY}$$

and provide some reasonable estimates of DENSITY. You've probably done something like this in Excel. It is worth bearing in mind now that if we 'just' wanted the lines, all we would need would be two points. But if we want to be flash and fancy, as is expected these days, and add transparent bands representing the 95% confidence interval around the fitted values (doesn't that sound flash?), we need more than two points, because the standard errors around a fitted line are decidedly non-linear. We'll see that below. To start this process, we invoke two functions: `expand.grid()` and `predict()`.



`expand.grid()` is a function that generates a grid of numbers—essentially it builds a factorial representation of any variables you provide to it, and returns a data frame. For example:

```
expand.grid(FIRST = c("A", "B"), SECOND = c(1, 2))
```

```
##      FIRST SECOND
## 1      A      1
## 2      B      1
## 3      A      2
## 4      B      2
```

Here, the first and second columns are each of the variables and the grid is expanded like a factorial design.

**predict()** is a function that generates fitted values from a model. It requires at least one argument, but works most effectively for us with three: a model, a set of new 'x' (explanatory) values at which we want to know 'y' values, generated according to our model, and another argument, interval, which provides a rapid way to get the 95% confidence interval, doing some hard work for us.



If we use **predict()** with only a model object as an argument, it returns to us the predicted value of 'y' at each of the 'x's in our original data frame. For our `limp.mod` model and the original data frame of 2 seasons  $\times$  4 density treatments  $\times$  3 replicates, we should see 24 predicted values returned, with sets of 3 (the replicates) being identical:

```
predict(limp.mod)
```

```
##      1      2      3      4      5      6
## 2.3949692 2.3949692 2.3949692 1.6075953 1.6075953 1.6075953
##      7      8      9     10     11     12
## 2.1594217 2.1594217 2.1594217 1.3938428 1.3938428 1.3938428
##     13     14     15     16     17     18
## 1.6546769 1.6546769 1.6546769 0.9358016 0.9358016 0.9358016
##     19     20     21     22     23     24
## 1.1499321 1.1499321 1.1499321 0.4777604 0.4777604 0.4777604
```

However, let's assume that we would specifically like a prediction of the number of eggs at a set of specific densities. First, we create the 'new x's' (DENSITY) at which we want values of y (EGGS). Here we use **expand.grid()** to generate a set of 'new x' values, *and* we take special care to label them as the column name in the original dataset, DENSITY.

We know the density varies from 8 to 45, so let's use the function `seq()` to get ten numbers in between 8 and 45:

```
# make some new DENSITY values at which we request predictions
new.x <- expand.grid(DENSITY =
  seq(from = 8, to = 45, length.out = 10))

# check it worked
head(new.x)

##      DENSITY
## 1  8.00000
## 2 12.11111
## 3 16.22222
## 4 20.33333
## 5 24.44444
## 6 28.55556
```



Let's add SEASON to this grid now. SEASON has two levels, spring and summer. We can add these levels to the 'new x's' as follows. Note how we use the function `levels()`, against the dataset, to extract the information we desire:

```
# make some new DENSITY values at which we request predictions
new.x <- expand.grid(
  DENSITY = seq(from = 8, to = 45, length.out = 10),
  SEASON = levels(limp$SEASON))

# check it worked
head(new.x)

##      DENSITY SEASON
## 1  8.00000 spring
## 2 12.11111 spring
## 3 16.22222 spring
## 4 20.33333 spring
## 5 24.44444 spring
## 6 28.55556 spring
```

Notice that `expand.grid()` has now created a factorial representation of DENSITY (four levels) and SEASON (two levels), such that each and every DENSITY:SEASON combination is represented. You may want to just type `new.x` to see it all. Now, we can embed these 'new x's' into the function `predict()`, in the argument known as ... wait ... `newdata`!



We will use **predict()** with three arguments: a model, a value for newdata, and a request for confidence intervals. We love this part. And we assign what predict returns to an object called new.y:

```
# generate fits and confidence interval at new.x values.
new.y <- predict(limp.mod, newdata=new.x,
                 interval = 'confidence')

# check it!
head(new.y)

##          fit          lwr          upr
## 1 2.394969 2.019285 2.770654
## 2 2.256632 1.931230 2.582034
## 3 2.118294 1.834274 2.402315
## 4 1.979957 1.724062 2.235852
## 5 1.841619 1.595998 2.087241
## 6 1.703282 1.447918 1.958646
```

This is just awesome. We have new.x, which looks like a version of the explanatory variables. And we have new.y, which is estimated using the coefficients from the model we have actually fitted, along with the 95% confidence interval around each value, with nice names — fit, lwr, and upr. Wowsa!



The next step we call ‘housekeeping’. Housekeeping is an important part of using R, and at this point in the plotting cycle we advocate a bit of it. What we suggest is that you combine the new y’s with the new x’s, so that we have a clear picture of what it is you’ve made. We can do that with the function **data.frame()**. We call the data frame addThese, ‘cause we are gonna add these to the plot.

And we make one *very important change*: we rename fit produced by **predict** to EGGS to match the original data. . . **rename()** is another cool little **dplyr** function!

```
# housekeeping to bring new.x and new.y together note that we
# rename fit to be EGGS matching the original data
addThese <- data.frame(new.x, new.y)
addThese <- rename(addThese, EGGS = fit)
# check it!
head(addThese)

##      DENSITY SEASON      EGGS      lwr      upr
## 1  8.00000 spring 2.394969 2.019285 2.770654
## 2 12.11111 spring 2.256632 1.931230 2.582034
```

```
## 3 16.22222 spring 2.118294 1.834274 2.402315
## 4 20.33333 spring 1.979957 1.724062 2.235852
## 5 24.44444 spring 1.841619 1.595998 2.087241
## 6 28.55556 spring 1.703282 1.447918 1.958646
```

So, now you have a new, small data frame that contains the grid of seasons and densities, as well as the predicted values and 95% CI at each of these combinations. You did not need to specify the coefficients, or the equations for the lines. **predict()** does all of that for you. You did not need to add and subtract  $1.96 \times \text{standard error}$  to and from each value either to generate the CI... **predict()** does all of that for you. **predict()** is your super-friend.



Before we finish with adding these data to the figure, we want to show how flexible this method is. Let's modify the code so that we get the predicted number of eggs for each season at the mean limpet density:

```
# new.x with DENSITY set to mean
new.x <-expand.grid(DENSITY=mean(limp$DENSITY),
                    SEASON = levels(limp$SEASON))

# predictions
predEgg <- predict(limp.mod, newdata = new.x)

# housekeeping
EggAtMeanDens <-data.frame(new.x, predEgg)
head(EggAtMeanDens)

##   DENSITY SEASON predEgg
## 1    24.5  spring  1.83975
## 2    24.5  summer  1.10375
```

### 6.3.9 THE FINAL PICTURE USING **ggplot()**

You are nearly there. You've carried out quite a substantial ANCOVA of an experiment. You've plotted the data, you've made a model, you've checked the assumptions, and you've interpreted the model. You've got some predictions as well, detailing how the model has fitted lines to your data. The final step, the last effort you may want to make, is to add these lines to the figure (e.g., Figure 6.8). An informative figure is worth a thousand words. If you can provide a figure that a reader looks at and, as a result, knows the question *and* the answer to the question, you will have communicated science effectively.





Here is the last bit of code to do this:

```
# raw data plot (you don't need to write this again...)

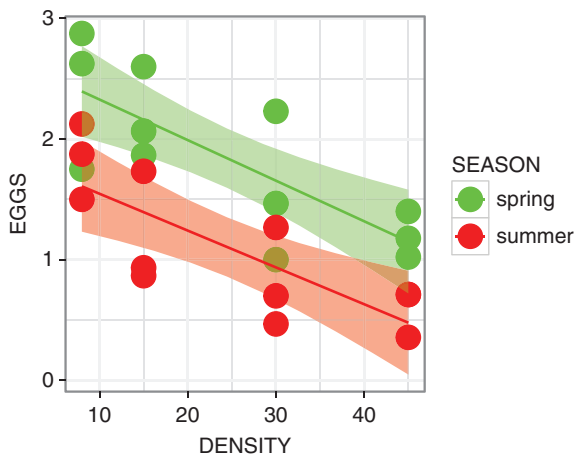
ggplot(limp, aes(x = DENSITY, y = EGGS, colour = SEASON)) +

  # first add the points
  geom_point(size = 5) +

  # now add the fits and CIs
  # note we don't need to specify DENSITY AND EGG
  # they are inherited from above!
  geom_smooth(data = addThese,
              aes(ymin = lwr, ymax = upr,
                  fill = SEASON), stat = 'identity') +

  #now adjust the colours
  scale_colour_manual(values = c(spring="green", summer="red")) +
  scale_fill_manual(values = c(spring="green", summer="red")) +

  # theme it
  theme_bw()
```



**Figure 6.8** Fitted lines for both seasons, added to the raw data for egg production by limpets. These lines we produced using a workflow that involves making predictions using the model via the function `predict()`. This produces lines that do not extend beyond the range of the data, which is good. The workflow to prepare this figure is very general and can be applied to any linear model.

### That ggplot code...

There is quite a bit going on in that code, but we've made every attempt to break it up and add annotation. The original raw data-plotting syntax is in there, but we've added the use of `geom_smooth()` to add the fits and CIs. We draw your attention to the annotation that says:

```
# note we don't need to specify DENSITY AND EGG
# they are inherited from above!
```

`geom_smooth()` is quite smoooooth. So were you. You made sure the names of ALL variables in the addThese data frame matched exactly what are in the original data. So, while we provide `geom_smooth()` with some new aesthetic boundaries for the CIs, it inherits (i.e. assumes) that the 'x =' and 'y =' variables are the same. Woot! So all we needed to do was specify ymin and ymax.

Oh, and `stat = 'identity'`. We covered that in the  $\chi^2$  example. It tells `geom_smooth()` to use what we give it, and not to try and calculate anything fancy-shmancy for us. That's all.

Just to help you along, a concise ANCOVA analysis script is presented in Box 6.1. It probably matches your own if you've been working along! The script is short and provides an archived (save it!), annotated (for you and your colleagues), and cross-platform record of your analysis.

## 6.4 Overview: an analysis workflow

Hopefully, it is clear that we have introduced you to a workflow for the analysis of your data using R. We have used this workflow consistently for a  $\chi^2$  contingency table test, a  $t$ -test, a linear regression, a one- and two-way ANOVA, and an ANCOVA. You've been introduced to several tools and tricks and insights into how to fit and interpret R's code and output. We review here nine steps to a happy R-life:

1. Enter your data in a spreadsheet program, check it, and save it as a comma-separated values file.

2. Start R, wipe its brain of all previous data—`rm(list=ls())`—and get your libraries sorted.
3. Import your data into R and check this has worked, for example using `glimpse()`.
4. Make a picture that reflects your questions and that will help you specify your model to answer your question.
5. Specify a statistical model to capture the hypothesis you are testing.
6. Assess whatever assumptions are important to the type of modelling you have decided to implement (use `autoplot()`).
7. Interpret the model with `anova()` and `summary()`.
8. Add fitted lines and CIs, when appropriate, to the plot; make the figure that is worth 1000 words.
9. Keep your data file and script file very safe (and backed up in a separate location). With these, you can recreate all your graphs and analyses with a few keystrokes in a few minutes at most.

If you adhere to this basic workflow for data analysis, and stick to *Plot -> Model -> Check Assumptions -> Interpret -> Plot Again*, you will have a very solid and efficient foundation for using R. Remember that nothing beats an effective picture for communicating your findings, and that the assumptions are just as important as the predictions.

#### Box 6.1: The final ANCOVA script

```
# ANCOVA Analysis of limpet reproduction
#libraries
library(dplyr)
library(ggplot2)
library(ggfortify)
#clear the decks
rm(list=ls())
# Read the limpet data and check the structure
limp <- read.csv("limpet.csv")
glimpse(limp) # make the first plot to explore the data
ggplot(limp, aes(x = DENSITY, y = EGGS, colour = SEASON)) +
  geom_point() +
```

## Box 6.1: (continued)

```

scale_color_manual(values = c(spring="green", summer="red")) +
  theme_bw() # make the ANCOVA model using lm; confirm
             what values are returned
limp.mod <- lm(EGGS ~ DENSITY*SEASON, data = limp)
# check the diagnostic plots for this model
autoplot(limp.mod, smooth.colour=NA)
# use anova() and summary() to interpret the model
anova(limp.mod) # sequential sums of squares table
summary(limp.mod) # coefficients table
# re-make the figure and add fitted lines
# FIRST--new x values--where do we want predicted eggs es-
#   timated?
new.x <- expand.grid(
  DENSITY = seq(from = 8, to = 45, length.out = 10),
  SEASON = levels(limp$SEASON))
# use predict() and data.frame() to generate the new y's
# collect them (housekeeping) for plotting in object called
preds.for.plot
new.y <- predict(limp.mod, newdata = new.x, interval = 'con-
fidence')
# housekeeping to bring new.x and new.y together
addThese <- data.frame(new.x, new.y)
addThese <- rename(addThese, EGGS = fit)
# remake the figure and add lines and CI
ggplot(limp, aes(x = DENSITY, y = EGGS, colour = SEASON)) +
  geom_point(size = 5) +
  geom_smooth(data = addThese,
    aes(ymin = lwr, ymax = upr, fill = SEASON),
    stat = 'identity') +
  scale_colour_manual(values = c(spring="green",
summer="red")) +
  scale_fill_manual(values = c(spring="green",
summer="red")) +
  theme_bw()

```