Chapter 9
More ANOVA: Repeated measures

James Myers
2022/5/14 draft

1. Introduction

Despite the cleverness and usefulness of independent-measures ANOVA, it's not the most commonly used type of ANOVA. This is because independent-measures ANOVA is designed for between-group study designs, where you compare independent samples, and researchers generally try to use within-group designs whenever they can. This kind of design is more powerful because each unit (e.g., experimental participants, speakers in a corpus study, experimental test items, and so on) acts as its own control. For example, if one group of people responds to nouns and another responds to verbs, it's hard to tell whether any noun-verb difference is due to the words or to the people, but if the same group of people responds to both nouns and verbs, then any difference must really be due to the words.

When each unit provides exactly two values, then we could do a paired *t* test. Just as the independent-measures ANOVA generalizes the unpaired *t* test, the **repeated-measures ANOVA** generalizes the paired *t* test. As you surely remember, the paired *t* test works by analyzing the paired *differences* within units instead of the raw data points themselves. The repeated-measures ANOVA just generalizes this logic to multi-level factors.

This chapter explains how to run repeated-measures ANOVA, mostly focusing on R, since Excel can only do one simple type. In particular, if you want to do a *two-way* repeated-measures ANOVA, which is quite common in experimental linguistics, Excel can't do it. On top of the increased complexity of using R compared with Excel (at least if you're still scared of R's command-based interface, and prefer Excel's GUI-based interface), ANOVA deserves a second chapter because ANOVA itself raises certain complexities that have to be dealt with. One concerns an issue related to the homoscedasticity issue we encountered when we were looking at unpaired *t* tests and correlation. Another is that in the typical linguistic experiment, we actually have two grouping units: speakers (experimental participants) and items (linguistic materials, like words or sentences). This means we have to build two separate ANOVA models, one for participants and one for items, and then somehow combine them again into one grand model, so we can conclude whether our overall results are or are not statistically significant. On top of all this, I still haven't told you how to estimate effect sizes for ANOVA. As we've seen, *p* values only say something about probability, not about real-life "significance". It turns out the most common effect size measure for ANOVA is closely related to the coefficient of

determination (r^2) that we saw with correlation. This will then lead us into the next chapter, when we see that ANOVA is just a special case of multiple regression.

2. Repeated-measures (and mixed) ANOVA

Whether our within-group study involves one multi-level factor or two or more factors and their interaction(s), using repeated-measures ANOVA feels very much like the independent-measures ANOVA that we discussed in the previous chapter. As usual we'll start by trying it out in Excel, then going further with R, and save the mathematical details for later. The underlying math has some new twists, however, which affects how you run it in Excel or R. Thus you do need to start with a vague sense of the math just to understand how to tell the programs what you want them to do, and to understand the results they give you.

As we saw in the previous chapter, the brilliant idea behind ANOVA is that it treats everything in terms of variance, comparing "interesting" (the **fixed** factors that we are trying to test) with "boring" (the **random** variables that are just noise to us). In technical terms, ANOVA **partitions the variance** into separate components so we can see which component is having a "significant" effect on the observed data. For example, if we are doing a two-way independent-measures ANOVA, with two independent variables A and B and their interaction, the variance is partitioned like so, with the interesting part (the A and B stuff) split off from the boring part (the "Error", i.e., the **residuals**: the variation that's in the observed data but not explained by the interesting part of the model):

Dependent variable = A + B + A × B + Residual error

But what if the levels (conditions, treatments) of A and B aren't sampled independently, but instead come grouped into units (like people or words)? This may make the math seem more annoying, but it actually has a wonderful advantage too: it allows us to partition the "Error" component itself, and get more information out of the whole analysis. For example, if our data come from an experiment where every participant gave us a response for all combinations of the A and B levels, we can write an equation more like this:

Dependent variable = A + B + A × B + Participant error + Residual error

Note that the box of randomness in the second equation isn't simply splitting up the one in the earlier equation, since if we don't factor out participant error we might accidentally confuse it for part of the effect of the fixed factors A and B (recall that similar bad things can happen if we run an unpaired *t* test on paired data).

To make this a bit more concrete, imagine that this is a priming experiment looking at the effects of phonology and semantics on reaction time (RT), so A = Homophone (i.e., the prime word and target word do or do not sound the same), and B = Synonym (i.e., the prime word and target word do or do not have similar meanings). Now suppose we randomly pull out two different RT values from the results. How might we explain the difference between these two values? Well, any specific RT in this experiment might have come from an experimental trial with a homophonous but non-synonymous prime-target pair from participant #23, or maybe from a trial with a prime-target pair that's both homophonous and synonymous from participant #12, or many other possibilities. But since the experiment is designed in a logical way, the only ways the two RTs could differ would be if they differed in the Homophone factor, or in the Synonym factor, or in the interaction of Homophone and Synonym (remember that an interaction is literally a product of two numbers), or if the two RTs come from the same ANOVA **cell** for two different participants (what I labeled "Participant error" in the above equation), or, finally, if they differed in some other totally random way that isn't captured by the model at all (residual error).

In this example, our research hypotheses relate to phonology and semantics, not cross-participant differences, but by including the random grouping variable of Participants, we can now **partial out** this bit of noisy variance, thereby reducing the size of the totally unexplained error (residuals). For example, maybe homophone priming speeds up the RT overall, but the participants probably also vary in their individual speeds, so unless we can pull out this "boring" influence on RT, we might not be able to see the homophone effect. Thus by shrinking the totally unexplained error, we explain more, and our repeated-measures ANOVA becomes more powerful than an independent-measures ANOVA would be.

This basic idea is not only clever, but as I said, it also affects the output reports given by Excel and R, and even R's command syntax. So let's make things even more concrete, and run some repeated-measures ANOVA models in Excel and R.

2.1 Four word types

Once upon a time, I saw an example in Gravetter & Wallnau (2004, p. 449), and decided to keep their numbers but change their description to something linguistic. So a wise old Chinese teacher wondered if syntactic category affects how easy it is for foreign students to learn Chinese words. She gave each of the five foreign students in one of her (tiny) classes four types of words to learn: nouns, verbs, adjectives, and adverbs. That is, each of those five students got all four types of words: this was a within-groups design, which the Chinese teacher (being wise) knew would give her greater statistical power than a between-groups design.

When she counted up how many words of each type that each student learned, she got the results shown in Table 1 (also available in the file **NVAA.txt**). Note that the numbers in the

Student column are just each person's arbitrary identification [ID] number, and each row of numbers for the four syntactic categories show how many words were learned by the student with that ID number.

Table 1. Results of a within-group experiment on word learning

Students	Nouns	Verbs	Adjectives	Adverbs
1	3	4	6	7
2	0	3	3	6
3	2	1	4	5
4	0	1	3	4
5	0	1	4	3

The wise old Chinese teacher thought that her sample was large enough (probably not, in real life, but let's ignore this for our demo), and the dependent variable (number of learned words) was continuous and normal enough (ditto), in order for her to run a parametric statistical test, and since she has one factor (word type) with four levels, and since each row counting learned words is grouped within a student, she also knows the particular type of parametric statistical test to use: a **one-way repeated-measures ANOVA**.

2.1.1 One-way repeated-measures ANOVA in Excel

As we saw in the previous chapter, Excel calls this test **Anova: Two-Factor Without Replication (雙因子變異數分析：無重複試驗)**. Now at last I can reveal where this weird name comes from: Excel calls it a "two-factor" ANOVA because one of the factors is the grouping units (here, students). That is, although there is just one **fixed variable** (word type, in our case), there is also one **random variable** (the students, in our case). Each quartet of word counts, on each row in Table 1, is associated with a specific student, so these values are not independent (e.g., maybe some students are better at learning all types of words than some other students). Excel refers to these rows, grouped by grouping units, as **blocks** (列), and since there's only one unit (student) per block, there's no "repetition".

Yes, the terminology is still confusing, but don't blame the linguists; blame Excel for using a different type of terminology (statistical terminology often varies across different disciplines, and maybe Excel is using terms more common in business or engineering).

So let's play along with the wise old Chinese teacher: put the above table into Excel, find the appropriate ANOVA tool in the Analysis ToolPak, select the entire table (you have to select the column for the grouping variable too, i.e. Students), and if you selected the column labels, you have to tell ANOVA that you included the labels too.

If you did it right, Excel should give you two tables. The first is a table of summary statistics, showing not just the means and standard deviations for each of the four levels of the

fixed word type factor, but also the means and variances for each of the five students (the random grouping variable). The second table is the ANOVA table shown in Table 2, showing the results for 列 (i.e., Students) and for 欄 (i.e., the columns, i.e., the word types).

Table 2. Excel's results for the one-way repeated-measures ANOVA

ANOVA

變源	SS	自由度	MS	F	P-值	臨界值
列	24	4	6	9	0.001343	3.25916
欄	50	3	16.66667	25	1.9E-05	3.4903
錯誤	8	12	0.666667			
總和	82	19				

Before we discuss how to report this result, imagine that the wise old Chinese teacher's slightly less wise colleague ran a very similar experiment, but gave each of the four word types to a separate group of five participants (i.e., she tested 5 × 4 = 20 independent participants), and through an amazing coincidence, got the exact same numerical results, as shown in Table 3 (note that in this table, the rows do not represent grouped values; I could just as well have reordered the values within each column some other way).

Table 3. Results of a between-group experiment on word learning (Excel style arrangement)

Nouns	Verbs	Adjectives	Adverbs
3	4	6	7
0	3	3	6
2	1	4	5
0	1	3	4
0	1	4	3

Then the less-wise colleague used Excel to run a one-way independent-measures ANOVA (correctly using the Analysis ToolPak tool ANOVA: One-Factor 單因子變異數分析), and got the ANOVA table shown in Table 4.

Table 4. Excel's results for the one-way independent-measures ANOVA

ANOVA

變源	SS	自由度	MS	F	P-值	臨界值
組間	50	3	16.66667	8.333333	0.001451	3.238872
組內	32	16	2			
總和	82	19				

Can you figure out how these two analyses relate to, and differ from, each other? Let's focus on the similarities first. First, both tables end with a row called 總和 (total), and the values on this row are identical for both types of ANOVA: total *SS* = 80 and total *df* = 19. This reflects the fact that both models are describing exactly the same total variance; the models differ only in how they partition this variance.

Second, both models have exactly the same fixed variable, namely the four-level word type factor. In the repeated-measures ANOVA table (Table 2), the information about this fixed variable is on the row called 欄 (column, for the word types), because the factor levels are listed in columns. In the independent-measures ANOVA table (Table 4), the information about this fixed variable is on the row called 組間 (between), because the variance here is between the factor levels. But the next three values on each of these rows are exactly the same: *SS* = 50, *df* = 3, *MS* = 16.66667. Moreover, you can probably guess where the *df* and *MS* values come from. Namely, *df* = *k*-1 (where *k* = 4, the number of levels) and *MS* = *SS*/*df* = 50/3 (try it!). So we can say that in a sense, a repeated-measures ANOVA actually contains a kind of independent-measures ANOVA inside of it.

However, something new happens with the random variation. There are only three rows in the independent-measures ANOVA table: the between-cell variance (組間, here the word types), and the within-cell variance (組內), and total (總和), which is just the sum of the first two. In other words, there is only one row for the random variation. By contrast, in the repeated-measures ANOVA table, there are two rows for random variation, namely that relating to the blocks (列, i.e., the grouping units, which here are the students), and the residual unexplained variation (錯誤: error).

In both tables, the *SS* and *df* values add up to the totals in the bottom row, but the "within" *SS* and *df* values in the independent-measures ANOVA table are split up (partitioned) into separate grouping-unit and unexplained error values in the repeated-measures ANOVA table. Check it yourself: in the independent-measures table, 組內 *SS* = 32, and in the repeated-measures table, 列 *SS* = 24 and 錯誤 *SS* = 8, and 24 + 8 = 32. Likewise for the *df* values: $df_{within} = 16 = df_{unit} + df_{error} = 4 + 12$.

This difference affects the statistical power, because it makes the repeated-measures *F* value bigger than the independent-measures *F* value. You already know where the *F* value comes from in the independent-measures ANOVA table: it's the *MS* for the fixed variable divided by the *MS* for the within-group variation (i.e., *MSE*). Check yourself: 16.66667/2 = 8.333333. The exact same logic applies in the repeated-measures ANOVA table, except now that we've also partialed out the variability due to random cross-student differences, the *MSE* value is lower: it's only 0.666667. Since the *MS* for the fixed variable is the same, the ratio ends up bigger: *MS*/*MSE* = 16.66667/0.666667 = 25 (if you don't round the two divisors mid-computation). As before, getting the *p* value for this higher *F* value requires two *df* values, the first for the fixed factor (so here *df* = 3 for both types of ANOVA) and the second for the error

(*df* = 16 for the independent-measures ANOVA but only *df* = 12 for the repeated-measures ANOVA, since this type of ANOVA shrinks the residual error). Putting all this together, we can therefore confirm the *p* values in both ANOVA tables:

Independent-measures *p* value: **=FDIST(8.333333, 3, 16) = 0.0014506**
Repeated-measures *p* value: **=FDIST(25, 3, 12) = 1.89621E-05**

Unsurprisingly, given how much bigger the repeated-measures *F* value is, there's a big difference in statistical significance too: this type of ANOVA is truly more powerful.

The wise old Chinese teacher, I mean the one who wisely ran the within-group experiment and the repeated-measures ANOVA, can thus report her results like so: "There was a significant effect of word type on learning ($F(3,12) = 25$, $MSE = 0.67$, $p < .0001$)."

You may have noticed that I said nothing about the first row of Excel's repeated-measures ANOVA table for the random grouping variable (students) also gives *F* and *p* values, computed exactly as for the fixed variable. Namely, you divide this row's *MS* value of 6 by the overall model's *MSE* value of 0.66666667, to get this row's *F* value of 9, and then you use this row's *df* value of 4 and the error row's *df* value of 12 to get the *p* value of 0.001343. Try it yourself!

It should make perfect sense to you that a repeated-measures ANOVA can give you a *p* value even for a random variable, since this is just a side-effect of partitioning the total variance, but I'm not sure why we would ever need to know this *p* value. Sure, we now know that these five participants are significantly different from each other in their overall word learning results, but in a realistic experiment, we don't really care about random cross-participant differences (though we might care about cross-participant differences as a fixed factor, like females vs. males). I'll ignore this line in the rest of this chapter, though in later chapters we will come back to random variables when we get to a fancy generalization of regression called mixed-effects modeling (which mix the fixed and random variables rather than partitioning them).

Before we leave Excel, let's quickly discuss a claim I've made a few times already, namely that *t* tests are just a special case of ANOVA. Does that mean that the paired *t* test is a special case of repeated-measures ANOVA? Why yes, that's exactly what it means.

You can confirm this yourself. Go back to the four-word-types data, but this time only select the columns for Subjects, Nouns, and Verbs, and run a one-way repeated-measures ANOVA using Excel's misleadingly named "two-way ANOVA without replication". If you do it right, your ANOVA table will appear as in Table 5:

Table 5. Excel's results for a one-way repeated-measures ANOVA for a two-level factor

ANOVA

變源	SS	自由度	MS	F	P-值	臨界值
列	12	4	3	3	0.15625	6.388233
欄	2.5	1	2.5	2.5	0.189004	7.708647
錯誤	4	4	1			

Now analyze the same two columns using Excel's paired *t* test tool. If you do this right, your results will look like that in Table 6:

Table 6. Excel's results for a paired *t* test for the same two-level factor

t 檢定：成對母體平均數差異檢定

	Nouns	Verbs
平均數	1	2
變異數	2	2
觀察值個數	5	5
皮耳森相關係數	0.5	
假設的均數差	0	
自由度	4	
t 統計	-1.58114	
P(T<=t) 單尾	0.094502	
臨界值：單尾	2.131847	
P(T<=t) 雙尾	0.189004	
臨界值：雙尾	2.776445	

Study the two tables carefully. Do they really give the same *p* values? How are the other values related, like *df*, *t*, and *F*?

2.1.2 One-way repeated-measures ANOVA in R

We can do all of this in R too, of course, plus a lot more. As we saw in the previous chapter, the **aov()** function allows us to do all sorts of ANOVA-specific tricks (e.g., computing the post hoc Tukey test). It is also designed to indicate which variables are within-unit in a repeated-measures ANOVA or even in a **mixed ANOVA**, where some variables are between-group and others are within-group.

As usual, remember that R expects that each row in your data frame represents a single data point. So in the case of the wise old Chinese teacher's experiment, the data should be arranged as in Table 7 (I won't show the whole table, since there are 20 data points).

Table 7. Results of a within-group experiment on word learning (top of R style arrangement)

Student	WordType	Learning
1	Noun	3
…	…	...

Here's a bit of R code to create the data frame for you:

```
wordexp = data.frame(Student = rep(1:5,4),
  WordType=c(rep("Noun",5), rep("Verb",5), rep("Adj",5), rep("Adv",5)),
  Learning =c(c(3,0,2,0,0),c(4,3,1,1,1),c(6,3,4,3,4),c(7,6,5,4,3)))
wordexp # See what it looks like
```

The above R code arranges the data kind of weirdly, since to avoid making a mistake I kept the word types together (e.g., five Noun values in a row) and cycled through the student ID numbers (1, 2, 3, 4, 5, and then 1, 2, 3, 4, 5 again). In a real experiment, your experimental control program will actually group your data by participants, since when you run each person, you get a separate data file for that person, and then you need to stick them all together. R doesn't care about the order of rows, but if you want to reorder them for your human eyes, you can use the **order()** function to resort the rows by participants (Student):

```
wordexp.sort = wordexp[order(wordexp$Student),] # Sort wordexp (optional)
wordexp.sort # See what you did
```

Note the syntax: **order(Fact)** gives you the original row labels (numbers) sorted by factor **Fact**, and by putting this vector **V** of row numbers inside the square brackets with a data frame **Data** as **Data[V,]**, R knows to sort the rows of **Data** (remember R's rule that rows come before columns. Obviously it's much easier to arrange your data in Excel, and then just loading it into R the way you want....

Before we can run our repeated-measures ANOVA in R, however, we still have one more job to do. If your grouping units are identified by numbers, as they are here (with our student ID numbers), then we need to tell R to treat these as separate categorical levels of a random variable, instead of treating them as numbers. R has no problem building models with numerical random variables (one of the many things that Excel cannot do), but that's not what we want for a repeated-measures ANOVA. Since it's legal to do so, however, if we forget to

convert the Student vector into a categorical factor, R won't give us any error message to inform us of our mistake. Instead, we'll simply get a result that isn't the one we want.

So before we run the analysis, we need to convert the unit variable Student into a categorical factor. A factor is of course a very important type of object in R (since it's a programming language for statistics), so there's a function designed just for this job:

```
wordexp$Student = as.factor(wordexp$Student)
```

Now, here's one more thing to learn before we can run the actual analysis. In R, the syntax for creating the one-way repeated-measures ANOVA object for our data is as follows:

```
aov(Learning ~ WordType + Error(Student/WordType), data=wordexp)
```

Some parts of this syntax are exactly the same as for independent-measures ANOVA, namely the **Dependent ~ Independent** formula (here, **Learning ~ WordType**), but there's a new element too: the **Error()** argument. This element should make some sense to you, now that you understand that a repeated-measures ANOVA partitions the error variance into the portion related to some random grouping unit and the portion related to unexplained, residual error. The **Error(Student/WordType)** syntax tells R that **Student** is a random grouping variable of units (students), and that **WordType** is a fixed variable that is "nested within" this unit (i.e., each student gave data for each level of the **WordType** factor).

We could also have combined the last two steps like so:

```
aov(Learning ~ WordType + Error(as.factor(Student)/WordType), data=wordexp)
```

As always with the **aov()** function, merely creating an aov object doesn't tell us much; we need to put it inside the **summary()** function to see our ANOVA tables. So here's a one-line way to analyze and view our results:

```
summary(aov(Learning ~ WordType + Error(as.factor(Student)/WordType),
  data=wordexp))
```

Running this gives the output text shown below, which contains the same information as in Excel's ANOVA table, except that it splits the random variable (**Student**) from the fixed variable (**WordType**, nested within **Student**) into two tables. R also doesn't bother giving you the F and p values for the random variable (or the total values, or the critical values used for computing confidence intervals). This makes R a bit more practical than Excel, since we don't really need any of the information in the first table (for the random variable **Student**); all of the information we need for the report ($F(3,12) = 25$, $MSE = 0.67$, $p < .05$) is given in the

second table (for the interaction of the random variable **Student** with the fixed variable **WordType**, symbolized with the colon "**:**", just as with other types of interactions in R).

```
Error: Student
          Df  Sum Sq  Mean Sq  F value  Pr(>F)
Residuals  4      24        6

Error: Student:WordType
          Df  Sum Sq  Mean Sq  F value  Pr(>F)
WordType   3      50   16.667       25 1.9e-05 ***
Residuals 12       8    0.667
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

By the way, how can we be sure that we've reported the degrees of freedom correctly? In particular, how do we know that we should report *F*(3,12) for WordType? As we discussed in the previous chapter (and will review below), the first *df* value is based on the number of levels in the factor (4 levels, and df = 3). We might think that the other *df* value should relate to the number of random grouping units (i.e., the students): since we have 5 students, so shouldn't the other *df* be related to that, and thus be 4, which is the *df* value shown in the first row of the ANOVA table (in Excel) or the first mini-table (in R)?

No, *F*(3,12) is indeed the correct way to report it. We can see this by seeing what *p* value we get with different df values, though for that we need new fake data so the p values don't have so many zeroes, in order to look for subtle changes in value:

```
wordexp2 = data.frame(Student = rep(1:5,4),
  WordType=c(rep("Noun",5), rep("Verb",5), rep("Adj",5), rep("Adv",5)),
  Learning =c(c(3,3,2,3,5),c(4,3,3,3,5),c(6,3,4,3,4),c(7,6,5,4,3)))
summary(aov(Learning~WordType+Error(as.factor(Student)/WordType),
  data=wordexp2))
```

```
Error: Student
          Df  Sum Sq  Mean Sq  F value  Pr(>F)
Residuals  4     7.7    1.925

Error: Student:WordType
          Df  Sum Sq  Mean Sq  F value  Pr(>F)
WordType   3    8.95    2.983    2.196   0.141
Residuals 12   16.30    1.358
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Here's the *p* value we get for WordType when we plug in the given *F* value (2.196) and the two *df* values in the main table (3, 12):

pf(2.196, 3, 12, lower.tail=F)
[1] 0.1413609 ... right!

That's the same p value as shown in the table. But we get the wrong p value when use the other *df*:

pf(2.196, 3, 4, lower.tail=F)
[1] 0.2310651 ... wrong!

We'll come back to *df* issues when we discuss math later in the chapter, but it's nice to know that when you run repeated-measures ANOVA in R, the two *df* values that you need for your report are displayed right next to each other.

2.2 Other types of ANOVA in R

Since R is a command-based statistics package, we can build on the above syntax to test all sorts of other ANOVA models, far more complicated than the three types that are built into Excel.

For example, what if we want to do a two-way repeated-measures ANOVA, one of the most common type in experimental linguistics? No problem. Just be careful with the formula syntax. So if your ANOVA model has the factors **Fact1** and **Fact2** as main effects, and also their interaction **Fact1:Fact2**, then your fixed part of the model can be written as **Fact1 * Fact2**. But to tell R that it's nested within your grouping unit **Unit**, you have to make sure that you write **Error(Unit/(Fact1*Fact2))**, not **Error(Unit/Fact1*Fact2)**, just as 2/(3*4) isn't the same as 2/3*4 (try it!). Like the **as.factor(Unit)** business (where **Unit** is a vector of ID numbers), R will not give you an error message if you forget the parentheses inside **Error()**, since the model will still make mathematical sense; it just won't make real-world sense!

So running a two-way repeated-measures ANOVA, predicting the dependent variable **Dep** from **Fact1** and **Fact2** and their interaction, where they're grouped within categorical factor **Unit**, all in the data frame **Data**, would look like this:

summary(aov(Dep ~ Fact1*Fact2 + Error(Unit/(Fact1*Fact2)),data=Data))

Similarly, a three-way repeated-measures ANOVA would look like this:

summary(aov(Dep ~ Fact1*Fact2*Fact3 + Error(Unit/(Fact1*Fact2*Fact3)),data=Data))

What if you want to run a **mixed ANOVA**, where one factor is between-groups while another is within-groups? For example, suppose the wise old Chinese teacher suspects that men and women differ in their abilities to learn nouns and verbs. Each person is just one gender or

the other (so **Gender** is between-groups) but to increase statistical power, each person in this experiment could get both word types (**WordType** is within-groups). In that case, the wise old Chinese teacher could test for main effects of **Gender** and **WordType**, and also their interaction, by putting only the within-unit factor inside the **Error** component:

summary(aov(Learning ~ Gender*WordType + Error(Student/WordType))

Let's try a concrete example. Dorami (the sister of Doraemon) decides to run an experiment to test her hypothesis that education level, syntactic category, and lexical frequency all affect reaction time in some sort of processing experiment on Martians, possibly with various interactions among these factors (she treats them all as categorical variables, even frequency). Later on we'll play with her entire data set, but for now we'll just look at the by-participant mean reaction times derived from the whole set: **dorami_part.txt**.

Note that education is a between-group factor, since the participants are either high school graduates or college graduates, but syntactic category and lexical frequency are both within-group factors, since each person gets both nouns and verbs and both high and low frequency words (i.e., words with frequencies above or below some point), themselves arranged in a 2 × 2 material design. Putting this all together, we have a perfect situation for a mixed three-way (2 ×2 × 2) ANOVA, which we can analyze with the following R code:

```
ddat = read.delim("dorami_part.txt")
ddat$Participant = as.factor(ddat$Participant) # Don't forget!!!
summary(aov(RT~Education*SynCat*Freq
  +Error(Participant/(SynCat*Freq)), data = ddat)) # Watch the parentheses!
```

Because the variance is being partitioned in various ways, R splits up the results into four ANOVA tables: one for the between-groups factor alone (**Education**), one each for the interaction of this factor with the within-groups factors (**SynCat** and **Freq**), and one for the two within-groups factors. Here they are:

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Education	1	58	58	0.008	0.929
Residuals	18	127079	7060		

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
SynCat	1	26863	26863	5.818	0.0268 *
Education:SynCat	1	166	166	0.036	0.8516
Residuals	18	83111	4617		

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Freq	1	72381	72381	13.825	0.00157 **
Education:Freq	1	26639	26639	5.088	0.03677 *
Residuals	18	94240	5236		

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
SynCat:Freq	1	2168	2168	0.214	0.649
Education:SynCat:Freq	1	740	740	0.073	0.790
Residuals	18	182374	10132		

Dorami reported these results like so. Compare the following with the above table to see where all of the values come from, and note that she uses the full factor names (for human eyes), not the abbreviations that she used to run the statistics (for R's convenience).:

> "We analyzed the by-participant RTs using a three-way mixed ANOVA with one between-group factor (education) and two within-group factors (syntactic category and frequency). There were significant main effects of syntactic category ($F(1,18) = 5.82$, $MSE = 4617$, $p < .05$) and frequency ($F(1,18) = 13.83$, $MSE = 5236$, $p < .05$), but not of education ($F < 1$). There was also a significant two-way interaction between education and frequency ($F(1,18) = 5.09$, $MSE = 5236$, $p < .05$), but no other significant interactions. Moreover, my brother is annoying."

2.3 Plotting interactions

As usual with interactions, you would also have to explain what the interactions looked like by showing a graph or describing it in words. Let's start by looking at the means:

```
hs.hi = mean(ddat$RT[ddat$Education=="HighSchool"&ddat$Freq=="High"])
hs.lo = mean(ddat$RT[ddat$Education=="HighSchool"&ddat$Freq=="Low"])
co.hi = mean(ddat$RT[ddat$Education=="College"&ddat$Freq=="High"])
co.lo = mean(ddat$RT[ddat$Education=="College"&ddat$Freq=="Low"])
hs.lo - hs.hi # 96.65417: big effect of frequency for those with high school education
co.lo - co.hi # 23.6625: smaller effect of frequency for those with college education
```

In other words, although frequency sped up RT for everybody, the frequency effect was greater for those with a high school education (**hs.lo - hs.hi**) than for those with a college education (**co.lo - co.hi**).

Since Dorami's model implies that there could be up to three main effects and four interactions (two two-way interactions and one three-way interaction), you'll need a lot of plots to plot them all. I would suggest focusing on the plots that are most relevant for showing your reader whether or not your research hypotheses were supported by the data. For this demo, let's just focus on the interaction between education and frequency.

Since our independent variables are all categorical, the best type of plot would be a bar plot, though since we're also trying to display interactions, a line plot could make sense too, so the reader can see whether the lines are parallel (no interaction) or not parallel (interaction),

and if they're not parallel, whether the lines cross (opposite effects) or meet at one end (effect only in one sub-condition).

2.3.1 Plotting ANOVA results in Excel

Let's start in Excel, and make a bar plot. We first have to get the mean **RT** values for all four conditions defined by crossing **Education** and **Freq**. One way to do this would be to sort **dorami_part.txt** in Excel by the **Education** and **Freq** columns, so that they form four blocks of **RT** values, as in Table 8 (note that the SynCat column gets all jumbled, but we don't need it for our computations).

Table 8. Dorami's re-sorted data

Participant	Education	SynCat	Freq	RT
11	College	Noun	High	910.5
11	College	Verb	High	701
12	College	Noun	High	841.8
...	...	...	...	...
11	College	Noun	Low	722.4
11	College	Verb	Low	756.4
12	College	Noun	Low	858.8
...	...	...	...	...
1	HighSchool	Noun	High	656.8
1	HighSchool	Verb	High	647
2	HighSchool	Noun	High	759.75
1	HighSchool	Noun	Low	673.2
1	HighSchool	Verb	Low	809.3333
2	HighSchool	Noun	Low	867.4
...	...	...	...	...

Then we can select the four ranges of RT values, and apply **=AVERAGE()** to each one. In this case, the size of each range is the same (20 cells), so we only have to write the cell function once, and then copy/paste it.

A way to do this that doesn't require resorting the data (and works even if the cell sizes are different) is to use Excel's "database average" function **=DAVERAGE(database, field, criteria)**. This takes three arguments: **database** is a table of data arranged in columns (as we have here), **field** is a string naming one of the columns that we want to compute the averages for (here, "RT"), and **criteria** indicate which columns define the specific cell we want to look at, also arranged as a minitable with named columns, with the name of the desired level under

each column label. For example, if we want to look at the cell where Education = College and Freq = High, we use the minitable in Table 9:

Table 9. Minitable defining some of the criteria for **=DAVERAGE()**

Education	Freq
College	High

Try it yourself! Get **dorami_part.txt** into Excel, then create four minitables, one for each of the four cells in the Education × Freq interaction, and select the relevant cells and cell ranges to enter into **=DAVERAGE()**. If you do it right, you should end up with the four cell means in Table 10.

Table 10. The four cell means for the education × frequency interaction

Education	Freq	
College	High	732.5
Education	Freq	
College	Low	756.1625
Education	Freq	
HighSchool	High	694.3
Education	Freq	
HighSchool	Low	790.9542

Now that you have the cell means, rearrange them into a labeled matrix, as in Table 11.

Table 11. The four cell means in a matrix format

	High	Low
College	732.5	756.163
HighSchool	694.3	790.954

A bit more playing around with Excel's bar plot tool then gives you Figure 1. Note that I put Education on the bottom and marked Freq using colors because the first variable is between groups while the latter is within groups, and this conceptual difference is now reflected by grouping the pairs of bars together within each education level. This allows us to express a natural interpretation for the interaction: college educated Martians show a smaller frequency effect than Martians with only a high school education.

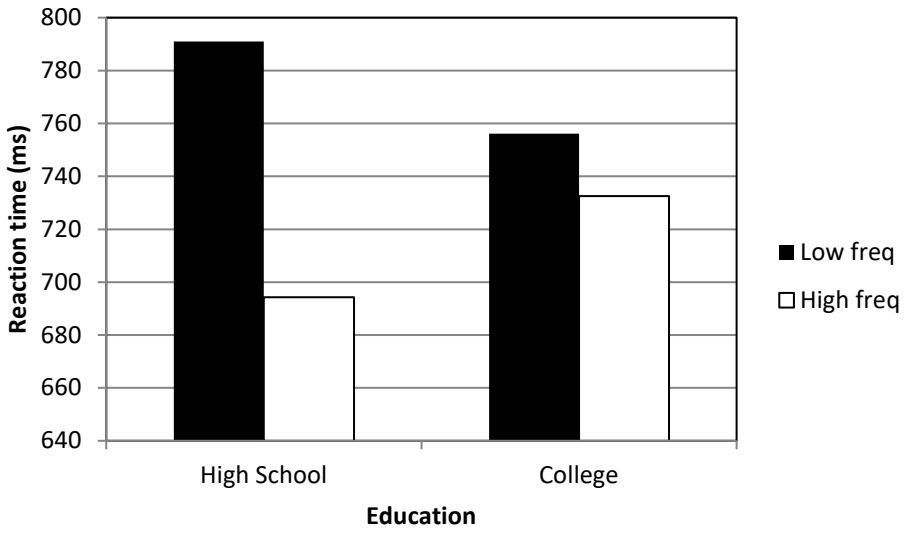

Figure 1. Interaction between education and frequency (bar plot in Excel)

2.3.2 Plotting ANOVA results in R

What about R? In the last chapter we used the **effects** package to make fancy plots of interactions, but unfortunately we can't use it for between-group variables analyzed using the **aov()** function, since the **effects** package is designed for linear models with regression-style formula syntax. Since the **aov()** includes that ANOVA-only **Error()** term for repeated-measures variables, the effects package gets all confused and can't run. In a later chapter we'll learn how to model Dorami's data in a more regression-style way, so the **effects** package can again be used to help with plotting.

However, R's **interaction.plot()** function still works, creating Figure 2. This function expresses the interaction with lines, to help see whether they are parallel or not (here, they're not, consistent with the statistical significance of the interaction).

```
interaction.plot(ddat$Education, # Variable on x-axis
  ddat$Freq, # Variable in the legend
  ddat$RT, # Variable on y-axis
  main = "Education x Frequency",
  xlab="Education", ylab="RT", # Default labels are ugly
  ylim=c(0,900), # It's more honest to put 0 on the bottom, to show the actual effect size
  legend=F, # Default legend style for interaction.plot is ugly, so I turned it off
  lwd=2) # Makes the lines thicker
  legend("bottomright",lty=c(1,2),lwd=c(2,2),
    legend=c("Low freq","High freq")) # Prettier legend
```

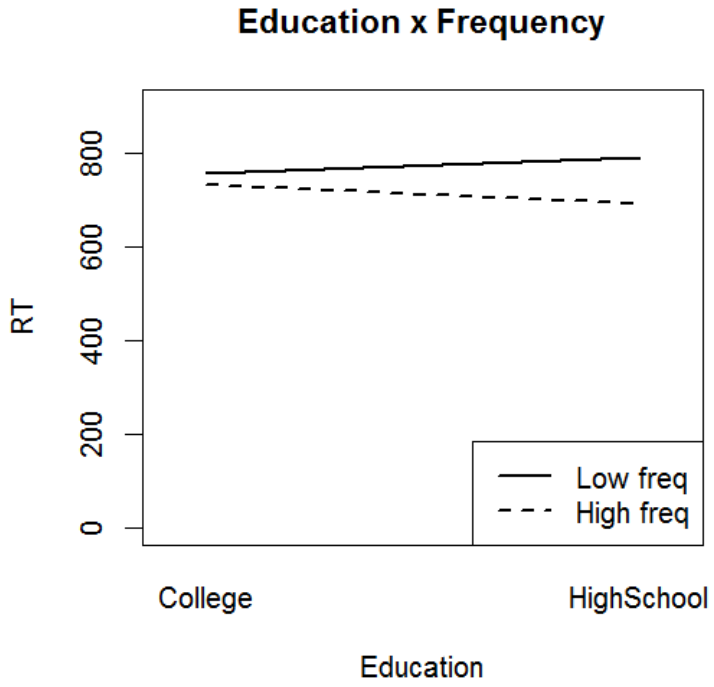

Figure 2. Interaction between education and frequency (ugly line interaction plot in R)

What if we wanted to make a bar plot in R? As in Excel, the first step is to compute the four cell means. Some earlier R code in this chapter does it manually, selecting the cells with the general-purpose function for extracting subsets of vectors, but we can make our lives a bit easier if we use a function in the **apply()** family, specifically designed for tables: **tapply()**. This acts like Excel's **=DAVERAGE()** function, except that it can applied to a wide variety of functions, not just **mean()** (though most of the time we'll just use it for means). If you're a fan of the tidyverse (https://www.tidyverse.org/), you could also use **summarize(group_by(...))**.

This function takes a table (here, our data frame), looks at one vector in it (in our case, the dependent variable **RT**, since that's what we'll be plotting), divides it up according to other variables (in our case, those defined by crossing **Education** and **Freq**), and then computes a function across it (in our case, **mean()**). So the following will create a table of mean RTs, where each mean is computed for each of the four cells defined by crossing **Education** and **Freq** (grouped together in a **list** object, which you may remember is like a vector created with **c()**, except that it permits any element type, not just elements all of one type). Since **Education** is listed first, this factor defines the rows, due to R's "rows first" rule. In any case, it gives us the same four means that we got with Excel.

tapply(ddat$RT, list(ddat$Education, ddat$Freq), mean) # Education on rows

	High	Low
College	732.5	756.1625
HighSchool	694.3	790.9542

Since the output is arranged like a matrix, it's not too hard to plot these means using the **barplot()** function (though harder than Excel, of course). The first thing to remember is how the arrangement of matrix cell values correspond to the bars that we want to plot. Suppose your matrix of means is arranged as in Table 12 (with the cells coded as for our contingency tables).

Table 12. Schematic matrix of cell means

		Factor F	
		Level F1	Level F2
Factor G	Level G1	A	B
	Level G2	C	D

Then **barplot()** will arrange your four bars as in Figure 3 (pay close attention to how the bars correspond to the table cells).

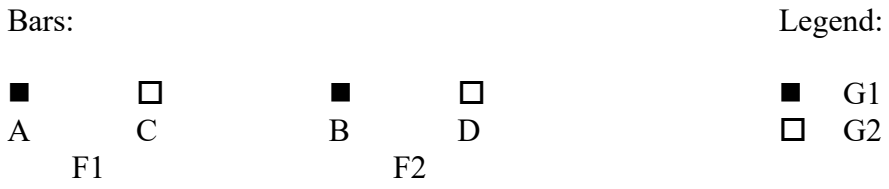

Figure 3. How **barplot()** arranges the bars for the cell means in the above matrix

Currently, our matrix is arranged with **Freq** defining the columns, so if we just use **barplot()** on it, it will put **Freq** at the bottom. But we want **Education** there, for the reasons I mentioned earlier. So the first we need to do is to transpose the matrix of cell means, using the **t()** function, or more reasonably, just doing **tapply()** with the factors in the other order:

```
tapply(ddat$RT, list(ddat$Freq, ddat$Education), mean) # Freq on rows
```

	College	HighSchool
High	732.5000	694.3000
Low	756.1625	790.9542

Note also that R sorts the levels alphabetically ("College" before "HighSchool", "High" before "Low"), both exactly backwards from what we logically would like them to be (as rearranged in my Excel plot in Figure 1). To make a really nice R plot, then, we'd have to do

a bit more work, but I'll just skip this part (just as I skipped it with the interaction line plot in Figure 2).

Finally, you have to remember (or look up) all of the extra arguments needed to make the plot look nice (Figure 4 doesn't look as nice as Figure 1):

```
barplot(tapply(ddat$RT, list(ddat$Freq, ddat$Education), mean),    # matrix of values
  beside=T,                              # draw bars next to each other, not on top
  names.arg=c("College","High school"),  # names at the bottom (in matrix order)
  legend.text=c("High freq","Low freq"),  # names in the legend box (in matrix order)
  ylim = c(0,1100),                      # min & max y-axis (so legend doesn't cover bars)
  ylab = "RT (ms)"                           # y-axis label
)
```

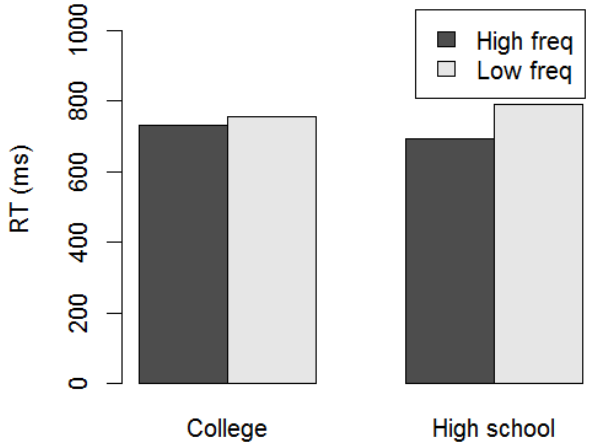

Figure 4. Interaction between education and frequency (ugly R bar plot)

2.4 More about the math of repeated-measures ANOVA

How does a repeated-measures ANOVA actually work? I've already sketched out the core ideas, but let's review them in mathematical terms (though we won't ever use this math).

Remember that the basic idea is that we want to understand as much of the variability in our data as possible, and so we want the "pure error" part (total residuals) to be as small as possible. Remember also that the independent-measures ANOVA partitions the variance into two parts: the variance between and within factor levels (treatments). Doing this lets us compute a single *F* ratio: $\text{variability}_{\text{between}}$ / $\text{variability}_{\text{within}}$.

If the data are repeated-measures, $\text{variability}_{\text{within}}$ itself has two parts: $\text{variability}_{\text{between\_treatments}}$ (e.g., nouns vs. verbs for one group of people) and "pure error" (e.g., random differences across people in this group). Since we've pulled out some more of the non-

error variability$_{between_treatments}$, the ratio of between-levels variability (in the numerator of the *F* ratio at the top) to error (in the denominator of the *F* ratio at the bottom) gets bigger than in an independent-measures ANOVA, increasing the power of the test.

Since the core logic is the same as for independent-measures ANOVA, the general formulas for *F* and *MS* don't change:

$$F = \frac{MS_{between}}{MS_{error}} \qquad \text{where } MS = s^2 = \frac{SS}{df}$$

The partitioning of the variance starts with the total *SS*, computed just as before:

$SS_{total} = \Sigma(x - M)^2$

Moreover, because the extra partitioning step occurs in the "within treatments" part, the formulas for the "between treatments" part are also the same as the "between" parts for independent-measures ANOVA:

$df_{between\_treatments} = k - 1$, where k = the number of treatments (levels) of the factor
$SS_{between\_treatments} = n \cdot SS_M$ (again, this simple version assumes the cells have the same size n)

The formulas for the "between units" part are exactly parallel to the "between treatments" formulas:

$df_{between\_units} = n - 1$, where n = the number of units
$SS_{between\_units} = k \cdot SS_P$ (where SS_P is computed across units)

The degrees of freedom for the residual, df_{error}, is calculated by multiplying the other two *df* values:

$df_{error} = df_{between\_treatments} \times df_{between\_units} = (k - 1) \cdot (n - 1)$

Finally (for our purposes anyway), the value of SS_{error}, for residual error, is found by subtracting all the "known" variability from the total variability:

$SS_{error} = SS_{total} - SS_{between\_treatments} - SS_{between\_units}$

And that's basically it! There only two bits of this that you might need in real life. First, it's useful to know that the core trick underlying repeated-measures ANOVA is that it partitions

up the "noisy" variance into two types of noise: the noise due to the grouping units (e.g., participants) and the "pure" noise of the residuals. More generally, an ANOVA model, like any statistical model, "explains" some of the data (e.g., the part captured by $SS_{between\_treatments}$) and doesn't explain the rest (the residuals). Since we can also compute the total amount of variability in the data as a whole, we can therefore also compute what proportion of the data is "explained" by our model. This is just like in correlation, where the regression line reflects how much of the dependent variable is "explained" by the independent variable (though "explains" doesn't have all the implications of this word in ordinary life, here referring only to how noisy the data are around the regression line). We'll come back to these ideas in the next section.

Second, the *df* values that you report in your ANOVA report are $df_{between\_treatments}$ and df_{error}, both because of the logic just sketched in the previous paragraph, and also because, as we saw earlier when we analyzed the fake **wordexp2** results, those are indeed the *df* values used by Excel and R to derive each *p* value from its associated *F* value.

3. ANOVA complexities

Though I've tried to keep the basic ideas simple, when you start to use ANOVA for real-life data, you come across a variety of tricky issues. In this section I focus on just four of them, all of which can only be dealt with in R, not Excel. The first is a concept called **sphericity**, which is like homoscedasticity, but is relevant for within-group analyses like repeated-measures ANOVA rather than between-group analyses like the unpaired *t* test. The second is a concept called ***minF'***, which is related to our old friend the *F* value, but designed to make it possible to combine by-participant analyses and by-item analyses together, since language experiments usually give us both. The third is a concept called **eta-squared**, which is a measure of effect size that follows from the mathematical relationship between ANOVA and regression. Finally, the fourth relates to a subtle difference between ANOVA and regression that can confuse people: the usual way ANOVA is calculated depends on the order of the variables, unlike what we'll see for multiple regression in the next chapter.

3.1 Sphericity

It turns out that despite its incredible usefulness for linguistic research, repeated-measures ANOVA has a weakness, one that some linguists worry enough about to try to fix (e.g., Zhang & Lai, 2010), but which many textbooks (e.g., Baayen, 2008, Johnson, 2008) ignore, and which is not even handled very easily in the base package of R (maybe because it really is a problem that can safely be ignored). This is the problem of **sphericity**, or more intuitively: **homogeneity**

of covariance. (Perhaps the term "sphericity" is meant to apply that the variance is "the same in all directions", like the radius of a sphere?)

The homogeneity of covariance is a generalization of the assumption of equal variance made in the ordinary unpaired *t* test. Namely, in repeated-measures ANOVA, the variances of the *differences* between any two factor levels are assumed to be statistically equal (that is, come from the same population values). If this assumption is violated (i.e., if your data show a statistically significant difference in sphericity), then the *p* values you get in your ANOVA will be too low, and so you'll make Type I errors, mistaking randomness for a real pattern (i.e., rejecting the null hypothesis when it's actually true).

Fortunately, since sphericity involves comparing differences across factor levels, it's not an issue if your factor only has two levels. So the problem doesn't come up in the paired *t* test, nor does it arise if you have a two-way or three-way or any-way ANOVA where none of the within-group variables has more than two levels. For example, if the wise old Chinese teacher tests three of her classes (Beginner, Intermediate, and Advanced) and gives everybody only two word types (nouns vs. verbs), then she could run a two-way mixed ANOVA with one between-group variable (the three-level factor of Class) and one within-group variable (the two-level factor of Word Type), and wouldn't have to worry about sphericity violations.

This gives you another reason to avoid multi-level factors when you design an experiment (if you can manage it). Not only are multi-level factors hard to interpret (at a minimum, you have to use planned comparisons or post-hoc tests like the Tukey test), but they raise this annoying problem of sphericity as well.

Even more annoyingly, statistical tests for sphericity aren't reliable, so some people (e.g., Max & Onghena, 1999) say that you should *always* assume that it's violated, and *always* correct for it, to be extra-sure that you're not committing a Type I error (though of course, the more you try to avoid Type I errors, the greater the risk of Type II errors: missing real patterns).

Just as with the unpaired *t* test heteroscedicity problem, the most common solution to the repeated-measures ANOVA sphericity problem is to compute the test statistic as usual (here, the *F* ratio), but then to compute *p* with lowered *df* values (giving a higher *p* value). As with the Welch unpaired *t* test not assuming equal variance, the lowered *df* is computed in a really ugly way (you can see the equations in Baron & Li, 2006), but the basic idea is simple: you compute a ratio called epsilon (the Greek letter ε, for "error", I guess) that takes the sphericity into account, and multiply both of your *df* values by it, to lower them and thereby reduce Type I error risk. The most popular correcting value seems to be the **Huynh-Feldt epsilon factor**. A common alternative is the Greenhouse-Geisser epsilon, but it is said that it doesn't balance Type I and Type II errors quite as well. Most statistics programs, including R and SPSS, give you both values (and the adjusted *p* values that result) anyway.

We can see how this works by looking at a toy example designed by Max & Onghena (1999, p. 264) to demonstrate the Type I error risk that arises when you commit the horrible sin of ignoring sphericity violations. As usual, I'll change it into a linguistic example.

Table 13 shows yet another experiment by that wise old Chinese teacher. This time she gives three tests to each of five of her students, one testing their syntactic abilities, one testing their phonological abilities (pronunciation), and one testing their lexical abilities (vocabulary). This creates "responses" to three "conditions" that are within-group, so it seems to be an appropriate situation for running a one-way repeated-measures ANOVA (the simple kind that even Excel can run).

You can copy/paste Table 13 below into Excel to analyze it, or download **maxongR.txt** to analyze it in R. Either way, you end up getting $F(2,8) = 4.73$, $p = .044$: significant. Try it!

However, these data seem to violate sphericity: the variance for the Condition 1 minus Condition 2 differences is only 70, but for Condition 1 vs. Condition 3 it's 191.3 and for Condition 2 vs. Condition 3 it's 328.3. Try it!

So as Max & Onghena (1999) show, to correct for the sphericity violation we need a high Huynh-Feldt epsilon factor here, namely .795 (I'll explain in a moment how to do this in R). We multiply each of our original *df* values (2, 8) by this factor, which gives us a new member of the *F* distribution family: $F(1.59, 6.36) = 4.73$, $p = .060$: not significant. You can confirm this part yourself: **pf(4.73, 2*0.795, 8*0.795, lower.tail=F)** == 0.06.

Table 13. Scores on three within-group tests

Subject	Condition 1	Condition 2	Condition 3
1	100	90	130
2	90	100	100
3	110	110	109
4	100	90	109
5	100	100	130

The base package of R can compute the Huynh-Feldt epsilon factor, but only in a complex and confusing way that I don't recommend. Namely, you have to redo the ANOVA analysis as a **MANOVA** (Multiple Analysis of Variance), which is a generalization of ANOVA where the dependent variable is a *vector* instead of a number, so we can treat the *rows* in the above table as a dependent *vector*. In the unlikely event you want to try this, here's some R code that works (you first have to download the file **maxong.txt**, which contains the values as a matrix):

```
maxong.mat = matrix(scan("maxong.txt"),ncol=3,byrow=T,      # Must load as matrix
  dimnames = list(Subject=1:5,Condition=c("Cond1","Cond2","Cond3"))) # Labels
mlm1 = lm(maxong.mat ~1) # Base model (intercept only: see next regression chapter)
mlm0= lm(maxong.mat ~0) # Random model (no predictors at all: ditto)
anova(mlm1,mlm0, X=~1, test="Spherical")
```

The last step uses the **anova()** function, which is that general-purpose function for running ANOVA-like analyses (we'll see it again in later chapters in more useful applications). Running the above code gives the following text output:

```
Greenhouse-Geisser epsilon: 0.6343
Huynh-Feldt epsilon:        0.7951

  Res.Df Df Gen.var.      F num Df den Df Pr(>F) G-G Pr H-F Pr
1      4    36.921
2      5  1  47.633 4.7246      2      8  0.044  0.077  0.060
```

If you study this report, you will recognize several values from the ones we saw above from Max & Onghena (1999): the *F* value, the original *df* and *p* values, the Huynh-Feldt epsilon factor, and the adjusted *p* value, here called "H-F Pr" (and also the Greenhouse-Geisser epsilon and its *p* value [G-G Pr]). To get the adjusted *df* values, you have to multiply the epsilon factor by the original *df* values; that gives 2*0.7951 = 1.59 and 8*0.7951 = 6.36, just as Max & Onghena (1999) gave.

Fortunately, we can avoid all this mess by using a much more flexible and user-friendly function available in a happy little package (created by Lawrence, 2016) called **ez**, which is short for "easy" (get it?).

The **ez** function that we need is called **ezANOVA()**. If you like, you can just use this function for running all of your ANOVA models; check the help with **?ezANOVA** to learn more about its syntax. For now, I'll focus on one specific benefit of this function: it automatically corrects for sphericity violations.

Let's start by analyzing the data contrary to what Max and Ongehena recommend:

```
maxong = read.delim("maxongR.txt") # Download same data, now in a data frame
summary(aov(Response~Condition+Error(as.factor(Subject)/Condition),
  data=maxong)) # This ANOVA doesn't correct for the sphericity violation
```

```
Error: as.factor(Subject)
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  4  439.1   109.8

Error: as.factor(Subject):Condition
          Df Sum Sq Mean Sq F value   Pr(>F)
Condition  2  928.5   464.3   4.725 0.0442 *
Residuals  8  786.1    98.3
```

There's that significant result. Now, let's redo it with the sphericity correction:

```
library(ez)
ezANOVA(data=maxong, dv = Response,      # dv = "dependent variable"
  wid = Subject, # "within identifier" (converted to a factor with a warning)
  within = Condition, # within-group factor (also warns you to make sure it's a factor)
  # between = ... Not relevant here, but this is how to mark between-group factors
)
```

Here are the results:

```
Warning: Converting “Subject” to factor for ANOVA.
Warning: Converting “Condition” to factor for ANOVA.
$ANOVA
          Effect DFn DFd        F         p p<.05       ges
2      Condition   2   8 4.724559 0.0441841     * 0.4311273
$`Mauchly’s Test for Sphericity`
          Effect         W         p p<.05
2      Condition 0.4235129 0.2756131


$`Sphericity Corrections`
          Effect       GGe      p[GG] p[GG]<.05       HFe      p[HF] p[HF]<.05
2      Condition 0.6343217 0.07726439           0.7950659 0.06027022
```

Note that **ezANOVA()** automatically converts Subject to a factor (though it gives you a warning, to remind you to be more careful next time). Then it gives the ANOVA table (DFn = numerator *df*, DFd = denominator *df*, ges = generalized eta-squared, which we’ll explain soon). So among other nice things, it shows just the two *df* values that you need for your report (here, $F(2,8)$).

Then it gives a test for sphericity (a generalization of the *F* test that we used for preparing unpaired *t* tests); since $p > .05$, this test says that we can assume that the sphericity assumption isn’t violated. But Max & Onghena (1999) say that this test is unreliable, so we’ll ignore the results of this test as they recommend.

Finally, at the bottom of the results, we can see the Huynh-Feldt epsilon (HFe) and its *p* value (and also the Greenhouse-Geisser epsilon [GGe], and its *p* value). These are the same values as what we got with the complicated and confusing MANOVA, but the **ezANOVA()** makes it a lot easier to get them.

Is all of this trouble worth it? Well, on the one hand, most people who do repeated-measures ANOVAs don’t do any of this; as with unpaired *t* tests, it usually doesn’t matter whether or not we correct for violations of homogeneous (co)variance, since often (usually?) the *p* value is far above or far below .05, instead of right on the edge as in Max & Onghena’s (1999) example. On the other hand, some people *do* worry about the “evils” of sphericity, and will criticize you if you don’t worry too.

3.2 *minF'*

And now things get more complicated. I'm sorry, but that's how math works: we keep building on top of what we've already built. But in this case, it's a much more serious issue, one that affects almost all experimentation in linguistics, and it's definitely an issue that all linguists who know statistics will complain about if you do it wrong.

As we've seen, the typical experiment in an introductory statistics textbook (e.g., Gravetter & Wallnau, 2004, where I got a lot of the fake data for my two ANOVA chapters) involves a set of participants responding to two or more types of stimuli, arranged as one or more factors. In the case of the repeated-measures ANOVA, the participants are treated as the **random variable**, since we imagine that the participants are a random sample representing a population. The other factors (e.g., word types or frequencies) are **fixed variables**: their levels aren't treated as a random sample, but are the things that we really do care about.

For example, if we ask twenty people to do some sort of experiment comparing one noun with one verb, the people are the random variable and the noun/verb variable is the fixed variable. Our experiment will allow us to generalize to the population of all people in how they respond to that specific noun and that specific verb. However, we can't generalize to the whole populations of nouns and verbs, since we only tested one of each.

In a real experiment, of course, we would actually test multiple nouns and multiple verbs. But now we have *two* random variables: both the participants and the items. What can we do? All of the statistical analyzes that we've seen so far assumes there is only *one* random variable.

To see why this is a problem, remember that computing an ANOVA involves partitioning the variance in a series of steps. So in a one-way repeated-measures ANOVA, we first partition out the variance relating to the fixed factor. Whatever is left is the random part of the data. Within this random part, we then partition out the variance due to cross- participant differences. This leaves the residuals for the model.

However, if there are *two* sources of random variation, both participants and items, their effects will be mixed together: we won't be able to partition out one from the other. Every trial in an experiment involves both a random participant and a random item; the response depends crucially on both at the same time. Thus we can't compute the proper *MSE*, needed in the denominator (bottom) for the *F* ratio (for more information on this point, see Raaijmakers et al., 1999, p. 417).

Until the early 1970s, experimental linguists would only run a **by-participant analysis**. That is, people averaged the scores within each item cell (e.g., get an average noun score and an average verb score), separately for each participant, and then treated only the participants as a random variable. They didn't do any **by-item analysis** at all.

But then along came Clark (1973), one of the most influential papers in the statistics of linguistic experiments. He pointed out that ignoring the by-item analysis falsely treats the items

as a *fixed* factor, just as in the unrealistic textbook experiments. He called this the **language-as-fixed-effect fallacy**, and it causes an overly high Type I error rate.

Clark's solution, described in detail below, involved conducting separate ANOVAs by participant (using participants as random variable, averaged across each type of item) and by item (using items as random variable, averaged across all participants), and then combining the two analyses together in a special way at the end.

Over the years, however, Clark's advice was misinterpreted as meaning that we should do both by-participant and by-item analyses, and just stop there, without the crucial putting-together-again step at the end. The convention came to be that we should only count an effect as significant if it's significant both by participants and by items. This is mathematically problematic, but it remains a common way to do ANOVA. The F value for the by-participant analysis is often labeled F_1, and F for the by-item analysis is labeled F_2, so Raaijmakers et al. (1999) call this **the $F_1 \times F_2$ fallacy**.

The problem is that the $F_1 \times F_2$ method still has an overly high Type I rate: it is quite possible to get a significant result both by participant and by item, but not really have a significant effect if participants and items are treated as random at the same time, as they should be. Thus in the early 2000s psychology journals started to go back to Clark's original advice. (More recently they've moved to another method, mixed-effects models, that we'll learn in a later chapter.)

So how did Clark recommend that we recombine the by-participants and by-items analyses? Although no true F ratio exists with two random variables, it is often possible to compute something similar to an F ratio, called **quasi F**, or **F'** (pronounced "F-prime"). F' has a distribution approximately the same shape as a real F distribution. In the formulas below, $something_1$ relates to the by-participant analysis and $something_2$ to the by-item analysis; remember that df_{num} is for the numerator (at the top of the ratio) and df_{denom} is for the denominator (at the bottom of the ratio).

$$F' = \frac{MS_{treatments} + MS_{items \times participants}}{MS_{treatments \times participants} + MS_{items}}$$

$$df_{num} = df_{treatments}$$

$$df_{denom} = \frac{(F_1 + F_2)^2}{\frac{{F_1}^2}{df_2} + \frac{{F_2}^2}{df_1}}$$

Since the equation for F' is pretty hard to work with, Clark (1973) suggested simplifying the procedure by pretending that $MS_{items \times participants} = 0$ (i.e., pretending that there is no interaction between items and participants). Through the magic of algebra, this gives the

minimum *F'*, or ***minF'***, which is much easier to calculate, and still uses the same *df* values as shown above:

$$minF' = \frac{MS_{treatments}}{MS_{treatments \times participants} + MS_{items}} = \frac{F_1 F_2}{F_1 + F_2}$$

If you think about it, you can see that *minF'* gives higher *p* values than the by-item or by-subject analyses alone. Since $minF' = (F_1 \times F_2)/(F_1+F_2)$, a bit of algebra shows that $minF' = F_1 \times (F_2/(F_1+F_2))$, and since F_1 and F_2 both must be greater than zero (since they're ratios of variances, which are squares, remember?), then $F_2 < F_1+F_2$, so therefore $(F_2/(F_1+F_2)) < 1$, and that means that $F_1 \times (F_2/(F_1+F_2)) = minF' < F_1$ (and likewise for F_2). If the *F* is smaller, then the *p* must be bigger. Got it?

Using these formulas, it's not difficult to invent a situation where both by-participant and by-item ANOVAs are significant, but *minF'* is not. For example, try out this code:

```
F1 = 4.5; df1= 23; F2 = 4.5; df2 = 34; df = 1
pf(F1,1,df1,lower.tail=F) # Significant by participants
pf(F2,1,df2,lower.tail=F) # Significant by items
minF = (F1*F2)/(F1+F2)
df.minF = (F1+F2)^2/(F1^2/df2 + F2^2/df1) # Using the correct df formula
pf(minF,1,df.minF,lower.tail=F) # Not significant by minF'
```

Thus when Clark recommended computing both by-participant and by-item ANOVAs, he meant that these values were supposed to be used as steps on the way to computing *minF'*, which should be the real final step.

Let's try doing it the right way with Dorami's experiment, this time starting with the full data set so we can run not just the by-participants analysis, but also the by-items analysis, and then combine them together using *minF'*.

3.2.1 Computing by-participant and by-item means

To play along at home, please download **doramiR.txt**. Let me start by explaining how I created the by-participant version in **dorami_part.txt** that we were using earlier. Since the data are in R-style columns, we can use the **aggregate()** function. Like **tapply()**, it does something similar to Excel's **=DAVERAGE()** function, but more flexibly, but unlike **tapply()**, it creates a data frame rather than a matrix. Namely, **aggregate()** takes the vector that we want to process (here, **RT**), a list of variables defining the subsets we care about (here, everything except **Item**, since this is for the by-participants analysis, not the by-items analysis), and a one-argument function (here, **mean**), and then creates a new data frame with the means computed for those subsets. Best of all, we can define our dependent variable and crossed independent variables

using a formula, similar to what we'll need when we run the ANOVA (except here we include the random variable Participant as the first variable):

```
ddat.all = read.delim("doramiR.txt")
ddat.all = na.omit(ddat.all) # Some data is missing, so let's clear out the NA's
ddat.part = aggregate(RT~Participant*Education*SynCat*Freq, data=ddat.all, mean)
head(ddat.part) # See what it looks like
```

	Participant	Education	SynCat	Freq	RT
1	11	College	Noun	High	910.5
2	12	College	Noun	High	841.8
3	13	College	Noun	High	794.6
4	14	College	Noun	High	853.0
5	15	College	Noun	High	602.8
6	16	College	Noun	High	603.0

We already did the by-participants analysis above, using the data in **dorami_part.txt**, but let's see if we come up with the same results with **ddat.part**.

```
ddat.part$Participant = as.factor(ddat.part$Participant) # Don't forget!!!
bypart.aov = summary(aov(RT~Education*SynCat*Freq
  +Error(Participant/(SynCat*Freq)), data = ddat.part)) # Summary of aov object
bypart.aov # Show summary: yes, it's the same as before!
```

We can create the by-item data frame the same way (you can compare this with **dorami_item.txt**, which I created like this):

```
ddat.item = aggregate(RT~Item*Education*SynCat*Freq, data=ddat.all, mean)
```

Now it's time to run the by-item analysis. Note that from the perspective of the items, **Education** is now a *within*-unit factor (each word is given to Martians with both education levels), while **SynCat** and **Freq** are now *between*-unit factors (since each word has its own syntactic category and frequency).

```
ddat.item$Item = as.factor(ddat.item$Item) # Don't forget!!!
byitem.aov = summary(aov(RT~Education*SynCat*Freq
  +Error(Item/Education), data = ddat.item))
byitem.aov # What the results look like
```

The results of the by-item analysis come in two ANOVA tables: one for the between-word factors **SynCat** and **Freq**, and one for the within-word factor **Education** and its interactions. I put all the results together below.

If we compare these by-item results with the by-participant results we got earlier, the only effect that's significant for both is **Freq**. So if we follow the common convention, we could write something like: "Only frequency had a significant effect in both the by-participant analysis ($F_1(1,18) = 13.83$, $MSE = 5236$, $p < .05$) and the by-item analysis ($F_2(1,16) = 7.39$, $MSE = 5144$, $p < .05$)." This is where most published papers stop, but technically it's not enough.

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
SynCat	1	13773	13773	2.677	0.1213	
Freq	1	38029	38029	7.393	0.0152	*
SynCat:Freq	1	1828	1828	0.355	0.5594	
Residuals	16	82306	5144			
Education	1	29	29	0.006	0.938	
Education:SynCat	1	45	45	0.010	0.923	
Education:Freq	1	13428	13428	2.923	0.107	
Education:SynCat:Freq	1	396	396	0.086	0.773	
Residuals	16	73514	4595			

3.2.2 Combining the by-participant and by-item ANOVAs

As you now know, the $F_1 \times F_2$ method has a high Type I error risk: there should be only one p value for **Freq**, namely the one we get from *minF'*. There's no point computing *minF'* for the factors that aren't significant in both analyses (since *minF'* can't give lower p values than the original by-participant and by-item p values), so let's just look at **Freq**.

For this we need the F_1, df_1, F_2, df_2 values for **Freq**. We could just copy and paste them from R's output, but the values in these outputs have been rounded; it's safer to extract the original values computed during the actual ANOVA. This is kind of hard to do, since as we've seen, **summary(aov(...))** creates different sets of ANOVA tables depending on the model being tested. Technically, these tables are put together in a **list** object, since they are different sizes and only lists can combine elements of different types. Just as vectors created with the **c()** function indicate their elements with single square brackets like **[...]**, list objects indicate their elements with double square brackets like **[[...]]**. So with that background, here's how to extract the values we need to compute *minF'*:

```
F1.Freq = bypart.aov[[3]][[1]][1,4] # 3rd table, 1st part of it, 1st row, 4th column
dfnum1 = bypart.aov[[3]][[1]][1,1] # 3rd table, 1st part of it, 1st row, 1st column
dfdenom1 = bypart.aov[[3]][[1]][3,1] # 3rd table, 1st part of it, 3rd row, 1st column
F2.Freq = byitem.aov[[1]][[1]][2,4] # 1st table, 1st part of it, 2nd row, 4th column
dfnum2 = dfnum1 # They're the same, and we don't really need both of them
dfdenom2 = byitem.aov[[1]][[1]][4,1] # 1st table, 1st part of it, 4th row, 1st column
minF = (F1.Freq*F2.Freq)/(F1.Freq+F2.Freq) # Apply minF' formula (can't use ' symbol)
dfnum.minF = dfnum1 # df for treatments (here, High vs. Low minus 1 = 1)
dfdenom.minF = (F1.Freq+F2.Freq)^2/(F1.Freq^2/dfdenom2 + F2.Freq^2/dfdenom1)
pf(minF,dfnum.minF,dfdenom.minF,lower.tail=F) # p = .03604612
```

After running this horrible-looking code, we can now add the following to the end of our report: "Frequency was also significant by a *minF′* analysis (*minF′*(1,30.05) = 4.82, $p < .05$)." (We can't include *MSE* here, since *minF′* doesn't give a single *MSE* value, as far as I know.)

As usual for these special kinds of analyses, we have to ask: Is all of that extra work worth it?

It partly depends on how our data are structured. As Raaijmakers et al. (1999) point out, Clark's approach is only relevant if the item-based variable represents independent samples, such as separate random collections of nouns vs. verbs. In many linguistic experiments, however, the items are actually **matched** in some way. For example, we might select nouns randomly, but for each noun we choose a verb identical to it in every psychologically relevant way: same length, same frequency, etc. Thus if we find any effect of the noun vs. verb treatment, it must really be due to the treatment, not some hidden random effect of the items.

Matching items like this has the effect of making MS_{items} and $MS_{items \times participants}$ very small, so the *F′* becomes the same as the usual *F* ratio for the by-participant analysis. So in this kind of experiment, there's no point in doing a by-item analysis at all, just as in the pre-Clark tradition!

$$F' \approx F_1 = \frac{MS_{treatments}}{MS_{treatments \times participants}}$$ for a matched-item experiment

Besides the extra effort, there is also another cost to using *minF′*: it reduces the risk of Type I errors only by increasing the risk of Type II errors! It also doesn't do anything to solve our two other big problems with ANOVA, namely how to compare levels within a factor, and how to deal with violations of sphericity in repeated-measures analyses.

Hence in the past ten years or so, experimental linguists (starting with Baayen, 2004) have advocated a newly invented regression-based approach that solves all of these problems (**mixed-effects modeling**, which we'll learn about in a later chapter). Clark's problem doesn't disappear entirely, but it becomes much easier to deal with it. For this reason, I know of no R package that computes *minF′* for you: you have to write your own code, as I did above.

3.3 Effect sizes for ANOVA

Just as with statistical hypothesis test, the *p* values given by ANOVA only represent "significance" in a narrow technical sense, namely the probability that your results are due to chance (well, the probability that your sample comes from the null hypothesis population). If you want to estimate how "significant" your results are in the normal meaning of the word, you need to measure effect size: how much effect do the independent variables have on the

dependent variable? As with *t* tests, there are two general ways to do this: with a point estimate (a single number) and with a confidence interval.

3.3.1 A point-estimate for ANOVA effect size: eta-squared

Because an ANOVA is a kind of regression, you can measure the overall fit of an ANOVA model using something related to the coefficient of determination r^2 that we looked at with simple regression. For some reason, when the r^2 logic is applied to ANOVA, it's called **eta-squared** (η^2: yes, that Greek letter is actually a kind of "e", even though it looks like a velar nasal *ŋ*). Similar to r^2, eta-squared represents the proportion of the variance in the dependent variable that is predicted by the independent variable(s) (that is, the proportion of the variance that isn't dumped into the garbage-can category of the residuals).

For independent-measures one-way ANOVA, η^2 is the variance predicted by your independent variables divided by the total amount of variance. Because variance is *MS*, which is closely related to *SS*, you can compute eta-squared directly from the *SS* values reported in the ANOVA table:

$$\eta^2 = \frac{SS_{between}}{SS_{total}} = \frac{SS_{between}}{SS_{beween}+SS_{within}} = \frac{SS_{between}}{SS_{beween}+SS_{error}}$$

Earlier in this chapter we already saw one way we can calculate this automatically in R. Remember? No? You just run your ANOVA using the **ezANOVA()** function in the **ez** package, and that will automatically give you a value called "ges", which stands for **generalized eta-squared** (an adjusted η^2 that also takes the number of fixed variables into account; we'll come back to this concept we look at multiple regression).

Let's see if that formula works for the first colored room experiment discussed in the previous chapter. First we'll do it by hand, using R's base **aov()** function:

```
exp1 = data.frame(Color = c(rep("Red",5), rep("Blue",5), rep("Yellow",5)),
  Learning=c(c(0,1,3,1,0),c(4,3,6,3,4),c(1,2,2,0,0))) # To keep track of the 3 samples
summary(aov(Learning~Color,data=exp1)) # Use this to copy/paste the values below
eta2 = 30/(30+16) # (Sum Sq for Color = 30, Sum Sq for Residuals = 16)
eta2
```

```
[1] 0.6521739
```

Now we'll redo it using a regression analysis, just to prove I wasn't lying when I said that eta-squared for an ANOVA is the same as the r^2 you get from regression. Look for the value reported in the text report as "Multiple R-squared":

summary(lm(Learning~Color,data=exp1)) # Multiple R-squared: 0.6522

Next, let's use the **ezANOVA()** function in the **ez** package (note that we have to tell it that all of the subjects were different). Look in the text report for "ges" (generalized eta-squared):

```
library(ez) # In case it's not already running
exp1$Subj = 1:15
ezANOVA(data=exp1,dv=Learning,between=Color,wid=Subj) # ges: 0.6521739
```

Since there's only one independent variable, the generalized eta-squared is the same as the simple eta-squared that we computed ourselves. In more complex models, the generalized eta-squared will be a bit lower (again, see the multiple regression chapter, coming up next).

That's three ways to compute eta-squared in R, and here's a fourth way. Remember the **lsr** package (Navarro, 2014), that we used to compute Cohen's *d* for *t* tests? Cohen's *d* is also a measure of effect size, so it's not surprising that this package also has a function for computing the effect size for ANOVA: **etaSquared()**, which operates on **aov()** models:

```
library(lsr)
etaSquared(aov(Learning~Color,data=exp1))
```

```
              eta.sq    eta.sq.part
Color      0.6521739    0.6521739
```

The "eta.sq" part is the overall eta-squared for the ANOVA model, and the "eta.sq.part" part is the eta-squared just for Color. They're the same here because the model only has this one independent variable.

3.3.2 Confidence intervals for ANOVA

It's also possible to compute confidence intervals for ANOVA, and as usual, the main benefit is to add error bars to graphs. Loftus & Masson (1994) explain how to do this, for any type of ANOVA (the journal editor asked me to follow their advice for Myers et al., 2006).

The basic idea builds on the formula we used for the one-sample *t* tests:

$$\mu = M \pm (t(df)_{95\%conf})(SE) = M \pm (t(df)_{95\%conf})(s/\sqrt{n}) = M \pm (t(df)_{95\%conf})\sqrt{(s/n)}$$

To generalize from a *t* test to ANOVA, remember that s^2 is the measure of randomness (error) in the data, and that in ANOVA, the measure of randomness is *MSE*. This gives the following general formula for confidence intervals in ANOVA, where *n* is the size of each cell:

$$\mu = M \pm t(df_{error})_{95\%conf}\sqrt{\frac{MSE}{n}}$$

I'm still not sure how useful this is, though, since you have to add the same error bars to everything in your plot, but unlike a *t* test, an ANOVA actually tests multiple things at the same time. So in a two-way ANOVA, you're testing two main effects and an interaction, but the 95% confidence intervals can only reflect one of these comparisons. Moreover, if your factor has more than two levels, testing statistical differences between levels requires a post-hoc test of some kind, so it seems to me that it would make sense to plot confidence intervals derived from something like Tukey's HSD (remember the lower [lwr] and upper [upr] bounds that you get when you use R's **TukeyHSD** function). But I have to admit that I've never seen anybody actually doing this in a published paper.

A different approach towards computing confidence intervals for ANOVA is reviewed in Morey (2008). This approach aims to combine information about the model with information about the reliability of each measurement point, so different means may be surrounded by different-sized error bars. In technical terms, it does this by taking each data point, subtracting the individual unit's overall mean (e.g., each participant's mean), adding the overall mean, multiplying by the number of within-group conditions, and dividing by the *df* for conditions, and then finally putting these values through the Loftus & Masson (1994) method.

I find it interesting that even the experts continue to disagree about what seems to be pretty basic stuff. This controversy highlights my own concern with error bars: it is unclear what they represent. Are they telling your readers about your statistical analysis (i.e., clarifying what you believe) or telling your readers about the objective results (i.e., clarifying the data so the readers can decide what to believe themselves)? Maybe such confusions explain why some good R-for-linguists books (Baayen, 2008; Johnson, 2008; Gries, 2013; Levshina, 2015) pay very little attention to confidence intervals; Winter (2019) discusses them a bit more).

If you do want to add error bars to your ANOVA-related plot, R can help, perhaps most simply and generally using the **emmeans** package (Lenth, 2016, 2022), which stands for "estimated marginal means" (because ANOVA is a kind of regression, and regression lines are found by estimating mean values conditioned on other values; remember the "margins" from chi-square tests?). As usual with R packages, it's not 100% clear what math lies behind this function, but since it gives you separate confidence intervals for each ANOVA cell, I assume it does something related to Morey (2008). Here's a demo using Dorami's by-participant ANOVA. As usual, we need the output of **aov()**, not **summary(aov())**. The endpoints of each cell's confidence interval are indicated by "lower.CL" and "upper.CL", which are centered around the cell means in "emmean".

```
library(emmeans)
bypart.aov.model = aov(RT~Education*SynCat*Freq
  +Error(Participant/(SynCat*Freq)), data = ddat.part)
emmeans(bypart.aov.model,c("Education","SynCat","Freq"))
```

Note: re-fitting model with sum-to-zero contrasts

Education	SynCat	Freq	emmean	SE	df	lower.CL	upper.CL
College	Noun	High	754	26	65.4	703	806
HighSchool	Noun	High	719	26	65.4	668	771
College	Verb	High	711	26	65.4	659	762
HighSchool	Verb	High	669	26	65.4	617	721
College	Noun	Low	774	26	65.4	722	826
HighSchool	Noun	Low	800	26	65.4	748	852
College	Verb	Low	739	26	65.4	687	790
HighSchool	Verb	Low	782	26	65.4	730	834

Warning: EMMs are biased unless design is perfectly balanced
Confidence level used: 0.95

In addition to the table, the function outputs some fancy mathematical notes, once again relating to stuff in the next chapter, since **emmeans()** secretly exploits the ANOVA-as-regression trick: "sum-to-zero contrasts" refers to what I'll call "sum coding" or "effect coding" in the next chapter, and "EMMs [estimated marginal means] are biased unless design is perfectly balanced" means that the ANOVA-as-regression trick only works if all of the cell sizes are equal.

But forget all that: crucially, the table gives you estimates for the lower and upper ends of this ANOVA model's confidence intervals, and just like the *t* test confidence intervals discussed in the t test chapter, they're all the same size (103, give or take some rounding), since they all come from the same statistical model. To plot them, I gave you some base R code in the *t* test chapter. The **ggplot2** package also has syntax for adding error bars (search the web yourself for examples, particularly relating to the function **ggplot()** and its argument called **geom_errorbar**).

3.4 Ordered variance partition in ANOVA

Let me show you something. First, let's redo the two-way independent-measures ANOVA we did in the previous chapter. I'll keep the old name too (**exp2**, since it was the second experiment we analyzed in that chapter):

```
exp2 = data.frame(Gender = c(rep("Female",15),rep("Male",15)),    # F+M
  Color = rep(c(rep("Red",5), rep("Blue",5), rep("Yellow",5)),2),    # RBY+RBY
  Learning=c(c(3,1,1,6,4), c(2,5,9,7,7), c(9,9,13,6,8),    # F: RBY
                c(0,2,0,0,3), c(3,8,3,3,3), c(0,0,0,5,0)))      # M: RBY
head(exp2) # See what it looks like (try it!)
colorgender.aov = aov(Learning ~ Gender * Color, data = exp2)
summary(colorgender.aov) # Try it!
```

We'll do the analysis again, but change the order of the independent variables Gender and Color in the formula. Logically, this should make no difference, right? Let's see:

```
colorgender2.aov = aov(Learning ~ Color * Gender, data = exp2)
summary(colorgender2.aov) # Try it!
```

Look at the results carefully. Do you see any differences? Not really; the only thing that changes is the order of the output report: originally the ANOVA table reported the main effect of Gender before Color, and now it reports Color before Gender. But the numbers are all the same as before.

But now let's modify the data slightly. Suppose that on the day of the wise old Chinese teacher's experiment, one of the students in one of the classes was home sick. We'll simulate that by removing the last row from the data frame:

```
exp2a = exp2[-nrow(exp2),]
tail(exp2); tail(exp2a) # Compare how each data frame ends
```

See how I did this? The **[row,col]** part describes the rows and columns of **exp2**; there's nothing in the **col** position because we want to keep all of the columns, and since **nrow(exp2)** gives the number of the last row, putting a - (minus sign) in front removes it.

Now we'll run our differently ordered ANOVAs again. I'll put the results on the page so we can look at them together:

```
colorgendera.aov = aov(Learning ~ Gender * Color, data = exp2a)
summary(colorgendera.aov)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Gender	1	107.73	107.73	20.866	0.000137 ***
Color	2	66.1	33.05	6.401	0.006164 **
Gender:Color	2	50.87	25.43	4.926	0.016574 *
Residuals	23	118.75	5.16		

```
colorgender2a.aov = aov(Learning ~ Color * Gender, data = exp2a)
summary(colorgender2a.aov)
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Color	2	71.23	35.61	6.898	0.0045 **
Gender	1	102.61	102.61	19.873	0.00018 ***
Color:Gender	2	50.87	25.43	4.926	0.01657 *
Residuals	23	118.75	5.16		

Now it's not just a matter of differently ordered rows in the ANOVA table: the values shown in the first two rows are also different! What's going on?

There's a subtle hint if you look at the new values more carefully: the *SS*, *MS*, and *F* values are all larger when they are in the first row. For example, when Gender is in the first row in the first table, $F = 20.866$, but when it's in the second row in the second table, $F = 19.873$. You can see similar differences in *SS* and *MS*, and likewise for *SS*, *MS*, and *F* for Color. However, these three values remain the same for the interaction Color:Gender across both tables, and the *SS* and *M* values for the residuals also remain the same.

Does this hint help? Well, think about how ANOVA works: it partitions the variance, separating the "interesting" variance from the "boring" variance. It also partitions different kinds of "interesting" variance, so it can give us results not just about Gender, but also about Color, and their interaction.

What this little exercise shows is that the ANOVA algorithm used by R's **aov()** function partitions the variance in order. So if the formula says **Learning ~ Gender * Color**, it first partitions out the effect of Gender on Learning, leaving a temporary sort of residual that Gender cannot explain, and then it passes these values on to Color and partitions out the effect of this factor, and then after that it passes the remainder to partition out the interaction Gender:Color. But if the formula says **Learning ~ Color * Gender**, it first partitions out Color. That's why the *F* value is higher for any given factor when that factor comes first: the ANOVA procedure is trying to maximize the amount of variation that this factor can explain. But after the two main effects are partitioned, all that's left is the same interaction (Gender:Color = Color:Gender) and overall residuals, so the last two ANOVA table rows end up the same.

But then why didn't see see an effect of order in the original data set? Because that data set was **balanced**: all of the cell sizes were exactly the same. But because of that missing student, the other data set is **unbalanced**: one of the cells is smaller than the others:

xtabs(~Gender + Color, data = exp2)

	Color		
Gender	Blue	Red	Yellow
Female	5	5	5
Male	5	5	5

xtabs(~Gender + Color, data = exp2a)

	Color		
Gender	Blue	Red	Yellow
Female	5	5	5
Male	5	5	4

When the data set is balanced, the order of factors in the model doesn't matter, since partitioning out any factor explains the same number of data points as partitioning out any other factor. This is another reason why ANOVA should ideally be run on cells with equal sample sizes, in addition to the point mentioned in the previous chapter, namely that the greater the difference in sample sizes, the greater the Type I error risk if you violate other basic ANOVA assumptions, like the normality of the dependent variable. Of course, in real life, you can't always count on there being equal sample sizes, and you just have to make do with the data you have.

There are three different ways to partition out variance in an ANOVA, and unfortunately they are called Type I, Type II, and Type III, reusing terms that you already know as meaning false alarm error, miss error, and "can't remember which" error, but now used for a totally different purpose. At the other extreme, Type III ANOVA tests each component of your model (including interactions) by removing just that one component to create a simplified model, and then comparing it with the full model (we'll be doing this kind of thing ourselves in the chapter on multiple regression). Type II ANOVA is a compromise between the two approaches that avoids having to test simplified models that don't make much sense (like for the full model Y ~ X1*X1, the simplified model Y ~ X1 + X1:X2 would include an interaction with X2 but not X2 itself, which is weird). Statistical programs tend to favor one or another of these other types, or show you all three for you to choose. Both Excel and R's **aov()** function computes Type I ANOVA. To run the other types in R, you need to use a different function called **anova()**, which is for comparing models. We'll use it to compare regression models in the next chapter, but won't ever bother with Type II or Type III ANOVA. But if you're curious about them anyway, you can read more on R's FAQ page:

https://cran.r-project.org/doc/FAQ/R-FAQ.html#Why-does-the-output-from-anova_0028_0029-depend-on-the-order-of-factors-in-the-model_003f

4. Conclusions

Almost every time you see an ANOVA in a linguistics study, it is a repeated-measures ANOVA or a mixed ANOVA (where some independent variables are between group and some are within group). This chapter explained how to run these kinds of analyses in Excel (only the

simplest kind: one-way repeated-measures ANOVA) and R (every kind, including two- or more-way repeated-measures ANOVA or mixed ANOVA). In both cases, the key to understanding the commands and the output is to see that repeated-measures ANOVA generalizes paired *t* tests by treating the variation across random grouping units (e.g., experimental participants) as part of the analysis, partialing it out from the completely unexplained variation left in the residuals. In R, the key syntactic things to remember are to use the **aov()** function, to treat your random grouping variable as a factor (even if your participants have numerical IDs), and to get your parentheses right inside the **Error()** term. Other complexities arise with repeated-measures or mixed ANOVA that Excel cannot deal with. If your within-groups factor has three or more levels, you might consider correcting for violations of sphericity (related to homoscedasticity), using the Huynh-Feldt epsilon to adjust the *df* values; this is computed automatically by the **ezANOVA()** function in the **ez** package. If your experiment involves not just multiple speakers but also multiple items per speaker, then you have two random variables, and you should, at the very least, compute separate ANOVA results, both by participants and by items, and ideally, recombine the two results together using *minF'*. To see whether your model is not only statistically significant, but also captures an impressive amount of variance in the data, you should compute eta-squared (which is just like r^2), again most easily done with the **ezANOVA()** function, or plot confidence intervals with the **emmeans()** function in the **emmeans** package. Finally, even though we'll see in the next chapter how ANOVA is related to regression, there is a subtle difference: in the most commonly used species of ANOVA (Type I), it matters what order you put them into the model, though only if your design is unbalanced.

References

Baayen, R. H. (2004). Statistics in psycholinguistics: A critique of some current gold standards. In G. Libben & K. Nault (eds.) *Mental Lexicon Working Papers 1*, 1-45.

Baayen, R. H. (2008). *Analyzing linguistic data: A practical introduction to statistics using R*. Cambridge, UK: Cambridge University Press.

Baron, J., & Li, Y. (2006). Notes on the use of R for psychology experiments and questionnaires. University of Pennsylvania and Children's Hospital of Philadelphia ms. http://www.psych.upenn.edu/~baron/rpsych/rpsych.html

Clark, H. (1973). The language-as-fixed-effect fallacy: A critique of language statistics in psychological research. *Journal of Verbal Learning and Verbal Behavior, 12*, 335-359.

Gravetter, Frederick J., & Wallnau, Larry B. (2004). *Statistics for the behavioral sciences* (6th edition). Wadsworth. [Newer editions have the same examples on different pages.]

Gries, S. T. (2013). *Statistics for linguistics with R: A practical introduction* (2nd edition). Berlin: De Gruyter.

Johnson, K. (2008). *Quantitative methods in linguistics*. Wiley.

Lawrence, M. A. (2016). ez: Easy analysis and visualization of factorial experiments. R package version 4.4-0. https://CRAN.R-project.org/package=ez

Lenth, R. V. (2016). Least-squares means: The R package lsmeans. *Journal of Statistical Software, 69*(1), 1-33.

Lenth, R. V. (2022). emmeans: Estimated Marginal Means, aka Least-Squares Means. R package version 1.7.3. https://CRAN.R-project.org/package=emmeans

Levshina, N. (2015). *How to do linguistics with R: Data exploration and statistical analysis*. John Benjamins.

Loftus, G. R., & Masson, M. E. J. (1994). Using confidence intervals in within-subject designs. *Psychonomic Bulletin & Review, 1*, 476-490.

Lorch, R. F., & Myers, J. L. (1990). Regression analyses of repeated measures data in cognitive research. *J. of Experimental Psychology: Learning, Memory, and Cognition, 16* (1), 149-157.

Max, L., & Onghena, P. (1999). Some issues in the statistical analysis of completely randomized and repeated measures designs for speech, language, and hearing research. *Journal of Speech, Language, and Hearing Research, 42*, 261-270.

Morey, R. D. (2008). Confidence intervals from normalized data: A correction to Cousineau (2005). *Tutorial in Quantitative Methods for Psychology, 4*(2), 61-64.

Myers, J., Huang, Y.-C., & Wang, W. (2006). Frequency effects in the processing of Chinese inflection. *Journal of Memory and Language, 54*, 300-323.

Navarro, D. (2014). *Learning statistics with R: A tutorial for psychology students and other beginners*. University of Adelaide ms. (See link to e-book on statistics class page.)

Raaijmakers, J. G. W., Schrijnemakers, J. M. C., & Gremmen, F. (1999). How to deal with "the language-as-fixed-effect fallacy": Common misconceptions and alternative solutions. *Journal of Memory and Language, 41*, 416-426.

Winter, B. (2019). *Statistics for linguists: An introduction using R*. Routledge.

Zhang, J., & Lai, Y. (2010). Testing the role of phonetic knowledge in Mandarin tone sandhi. *Phonology, 27*(01), 153-201.