Multilevel linear models

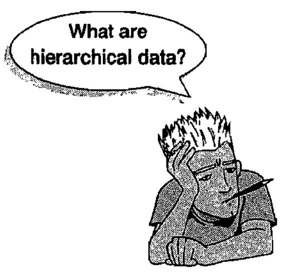
19

FIGURE 19.1 Having a therapy session in 2007

19.1. What will this chapter tell me? ①

Over the last couple of chapters we saw that I had gone from a child having dreams and aspirations of being a rock star, to becoming a living (barely) statistical test. A more dramatic demonstration of my complete failure to achieve my life's ambitions I can scarcely imagine. Having devoted far too much of my life to statistics, it was time to unlock the latent rock star once more. The second edition of the SPSS version of this book had left me in desperate need for some therapy and, therefore, at the age of 29 I decided to start playing the drums (there's a joke in there somewhere about it being the perfect instrument for a failed musician, but really they're much harder to play than people think). A couple of years later I had a call from an old friend of mine, Doug, who used to be in a band that my old band Scansion used to play with a lot: 'Remember the last time I saw you we talked about you coming and having a jam with us?' I had absolutely no recollection whatsoever of him saying this, so I responded 'Yes'. 'Well, how about it then?' he said. 'OK,' I said, 'you arrange it and I'll bring my guitar.' 'No, you whelk,' he said, 'we want you to drum and maybe you could learn some of the songs on the CD I gave you last year?' I'd played his band's CD and I liked it, but there was no way on this earth that I could play the drums as well as their drummer. 'Sure, no problem', I lied. I spent the next two weeks playing along to this CD as if my life depended on it and when the rehearsal came, much as I'd love to report that I drummed like a lord, I didn't. I did, however, nearly have a heart attack and herniate everything in my body that it's possible to herniate (really, the music is pretty fast!). Still, we had another rehearsal, and then another and, well, many years down the line we're still having them. The only difference is that now I can play the songs at a speed that makes their old recordings seem as though a sedated snail was on the drums (www.myspace.com/fracturepattern). The point is that it's never too late to learn something new. This is just as well because, as a man who clearly doesn't learn from his mistakes, I agreed to write a chapter on multilevel linear models, a subject about which I know absolutely nothing. I'm writing it last, when I feel mentally exhausted and stressed. Hopefully at some point between now and the end of writing the chapter I will learn something. With a bit of luck you will too.

19.2. Hierarchical data ②



In all of the analyses in this book so far we have treated data as though they are organized at a single level. However, in the real world, data are often hierarchical. This just means that some variables are clustered or *nested* within other variables. For example, when I'm not writing statistics books I spend most of my time researching how anxiety develops in children below the age of 10. This typically involves my running experiments in schools. When I run research in a school, I test children who have been assigned to different classes, and who are taught by different teachers. The classroom that a child is in could conceivably affect my results. Let's imagine I test in two different classrooms. The first class is taught by Mr. Nervous. Mr. Nervous is very anxious and often when he supervises children he tells them to be careful, or that things that they do are dangerous, or that they might hurt themselves. The second class is taught by Little Miss Daredevil.[1] She is very carefree and she believes that children in her class should have the freedom to explore new experiences. Therefore, she is always telling them not to be scared of things and to explore new situations. One day I go into the school to test the children. I take in a big animal carrier, which I tell them has an animal inside. I measure whether they will put their hand in the carrier to stroke the animal. Children taught by Mr. Nervous have grown up in an environment where their teacher reinforces caution, whereas children taught by Miss Daredevil have been encouraged to embrace new experiences. Therefore, we might expect Mr. Nervous's children to be more reluctant to put their hand in the box because of the classroom experiences that they have had. The classroom is, therefore, known as a *contextual variable*. In reality, as an experimenter I would be interested in a much more complicated situation. For example, I might tell some of the children that the animal is a bloodthirsty beast, whereas I tell others that the animal is

[1] Those of you who don't spot the Mr. Men references here, check out http://www.mrmen.com. Mr. Nervous used to be called Mr. Jelly and was a pink jelly-shaped blob, which in my humble opinion was better than his current incarnation.

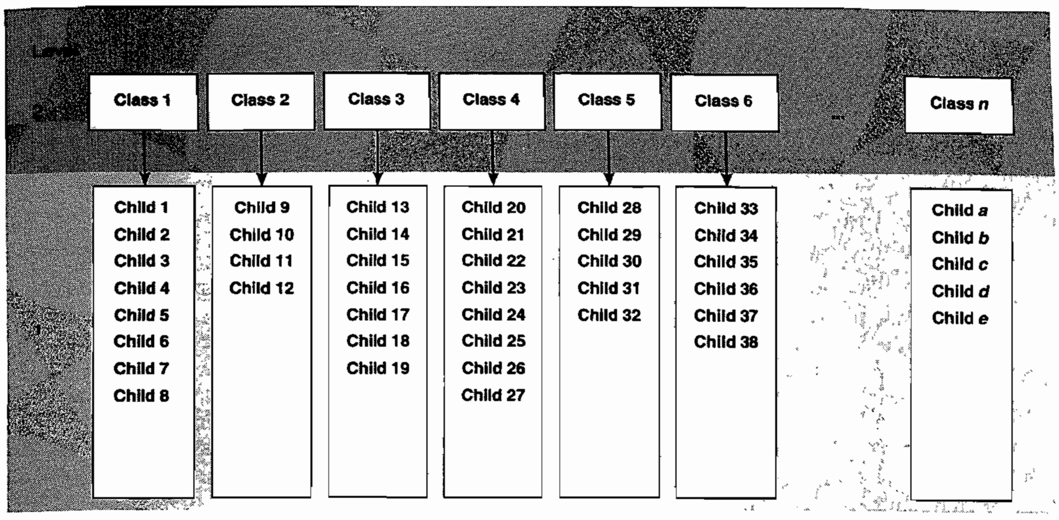


FIGURE 19.2
An example of a two-level hierarchical data structure: children (level 1) are organized within classrooms (level 2)

friendly. Now obviously I'm expecting the information I give the children to affect their enthusiasm for stroking the animal. However, it's also possible that their classroom has an effect. Therefore, my manipulation of the information that I give the children also has to be placed within the context of the classroom to which the child belongs. My threat information is likely to have more impact on Mr. Nervous's children than it will on Miss Daredevil's children. One consequence of this is that children within Mr. Nervous's class will be more similar to each other than they are to children in Miss Daredevil's class and vice versa.

Figure 19.2 illustrates this scenario more generally. In a big data set, we might have collected data from lots of children. This is the bottom of the hierarchy and is known as a *level 1* variable. So, children (or cases) are our level 1 variable. However, these children are organized by classroom (children are said to be *nested* within classes). The class to which a child belongs is a level up from the participant in the hierarchy and is said to be a level 2 variable.

The situation that I have just described is the simplest hierarchy that you can have because there are just two levels. However, you can have other layers to your hierarchy. The easiest way to explain this is to stick with our example of my testing children in different classes and then to point out the obvious fact that classrooms are themselves nested within schools. Therefore, if I ran a study incorporating lots of different schools, as well as different classrooms within those schools, then I would have to add another level to the hierarchy. We can apply the same logic as before, in that children in particular schools will be more similar to each other than to children in different schools. This is because schools tend to reflect their social demographic (which can differ from school to school) and they may differ in their policies also. Figure 19.3 shows this scenario. There are now three levels in the hierarchy: the child (level 1), the class to which the child belongs (level 2) and the school within which that class exists (level 3). In this situation we have two contextual variables: school and classroom.

Hierarchical data structures need not apply only to between-participant situations. We can also think of data as being nested within people. In this situation the case, or person, is not at the bottom of the hierarchy (level 1), but is further up. A good example is memory. Imagine that after giving children threat information about my caged animal I asked them a week later to recall everything they could about the animal. For each child there are many facts that they could recall. Let's say that I originally gave them 15 pieces of information; some children might recall all 15 pieces of information, but others might remember only

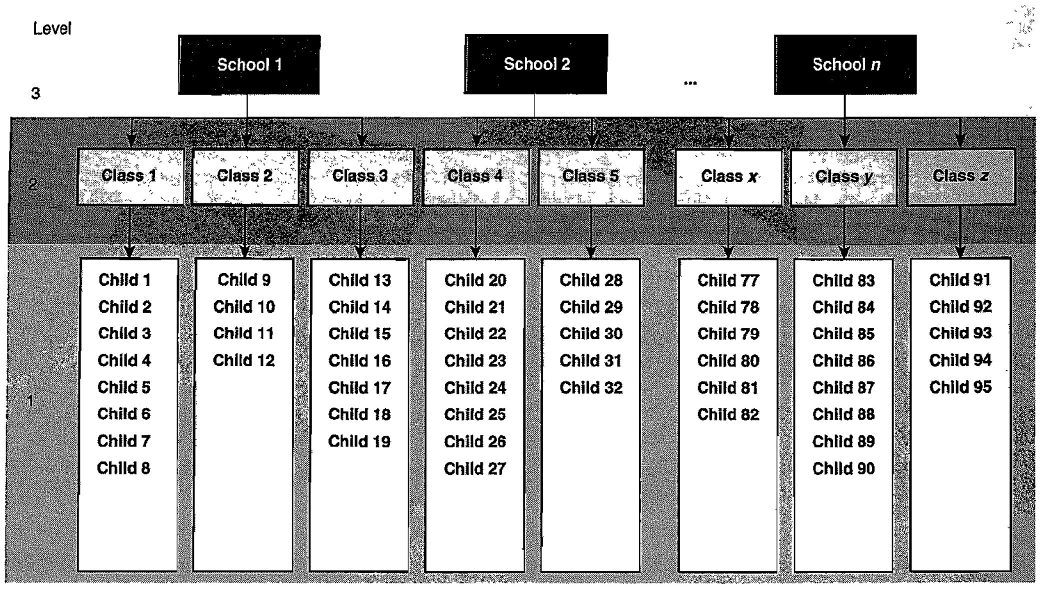

Level
3
2
1
School 1
School 2
...
School n
Class 1
Class 2
Class 3
Class 4
Class 5
Class x
Class y
Class z
Child 1 Child 2 Child 3 Child 4 Child 5 Child 6 Child 7 Child 8
Child 9 Child 10 Child 11 Child 12
Child 13 Child 14 Child 15 Child 16 Child 17 Child 18 Child 19
Child 20 Child 21 Child 22 Child 23 Child 24 Child 25 Child 26 Child 27
Child 28 Child 29 Child 30 Child 31 Child 32
Child 77 Child 78 Child 79 Child 80 Child 81 Child 82
Child 83 Child 84 Child 85 Child 86 Child 87 Child 88 Child 89 Child 90
Child 91 Child 92 Child 93 Child 94 Child 95

FIGURE 19.3 An example of a three-level hierarchical data structure

two or three bits of information. The bits of information, or memories, are nested within the person and their recall depends on the person. The probability of a given memory being recalled depends on what other memories are available, and the recall of one memory may have knock-on effects for what other memories are recalled. Therefore, memories are not independent units. As such, the person acts as a context within which memories are recalled (Wright, 1998).

Figure 19.4 shows the structure of the situation that I have just described. The child is our level 2 variable, and within each child there are several memories (our level 1 variable). Of course we can also have levels of the hierarchy above the child. So, we could still, for example, factor in the context of the class from which they came (as I have done in Figure 19.4) as a level 3 variable. Indeed, we could even include the school again as a level 4 variable.

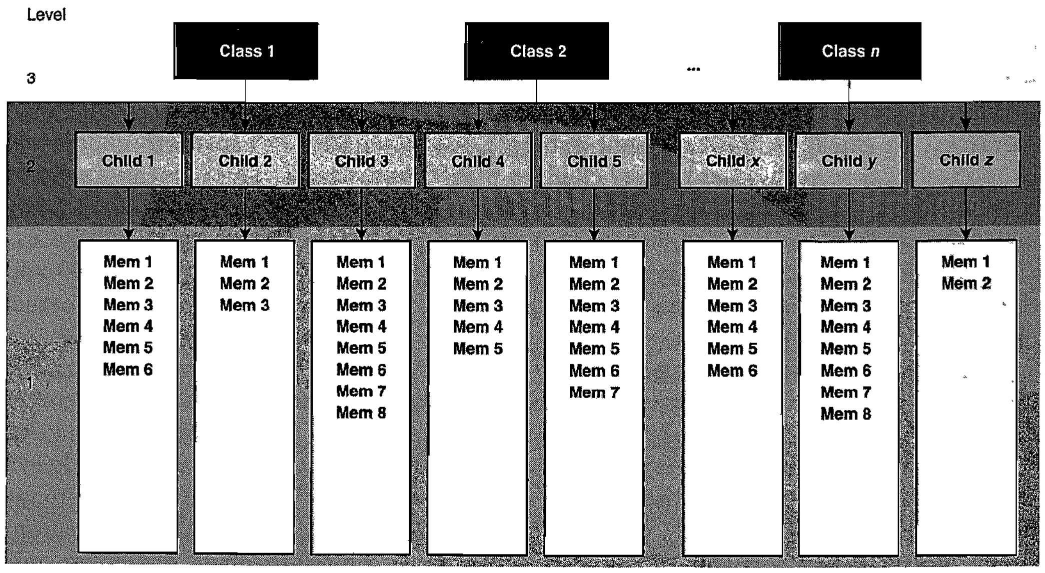

Level
3
2
1
Class 1
Class 2
...
Class n
Child 1
Child 2
Child 3
Child 4
Child 5
Child x
Child y
Child z
Mem 1 Mem 2 Mem 3 Mem 4 Mem 5 Mem 6
Mem 1 Mem 2 Mem 3
Mem 1 Mem 2 Mem 3 Mem 4 Mem 5 Mem 6 Mem 7 Mem 8
Mem 1 Mem 2 Mem 3 Mem 4 Mem 5
Mem 1 Mem 2 Mem 3 Mem 4 Mem 5 Mem 6 Mem 7
Mem 1 Mem 2 Mem 3 Mem 4 Mem 5 Mem 6
Mem 1 Mem 2 Mem 3 Mem 4 Mem 5 Mem 6 Mem 7 Mem 8
Mem 1 Mem 2

FIGURE 19.4 An example of a three-level hierarchical data structure, where the level 1 variable is a repeated measure (memories recalled)

19.2.1. The intraclass correlation ②

You might well wonder why it matters that data are hierarchical (or not). The main problem is that the contextual variables in the hierarchy introduce dependency in the data. In plain English, this means that residuals will be correlated. I have alluded to this fact already when I noted that children in Mr. Nervous's class would be more similar to each other than to children in Miss Daredevil's class. In some sense, having the same teacher makes children more similar to each other. This similarity is a problem because in nearly every test we have covered in this book we assume that cases are independent. In other words, there is absolutely no correlation between residual scores of one child and another. However, when entities are sampled from similar contexts, this independence is unlikely to be true. For example, Charlotte and Emily's responses to the animal in the carrier have both been influenced by Mr. Nervous's cautious manner, so their behaviour will be similar. Likewise, Kiki and Jip's responses to the animal in the box have both been influenced by Miss Daredevil's carefree manner, so their behaviour will be similar too. We have seen before that in ANOVA, for example, a lack of independence between cases is a huge problem that really affects the resulting test statistic – and not in a good way! (See section 10.3.)

By thinking about contextual variables and factoring them into the analysis we can overcome this problem of non-independent observations. One way that we can do this is to use the intraclass correlation (ICC). We came across this measure in section 17.9.3 as a measure of inter-rater reliability, but it can also be used as a measure of dependency between scores. We'll skip the formalities of calculating the ICC (but see Oliver Twisted if you're keen to know), and we'll just give a conceptual grasp of what it represents. In our two-level example of children within classes, the ICC represents the proportion of the total variability in the outcome that is attributable to the classes. It follows that if a class has had a big effect on the children within it then the variability within the class will be small (the children will behave similarly). As such, variability in the outcome within classes is minimized, and variability in the outcome between classes is maximized; therefore, the ICC is large. Conversely, if the class has little effect on the children then the outcome will vary a lot within classes, which will make differences between classes relatively small. Therefore, the ICC is small too. Thus, the ICC tells us that variability within levels of a contextual variable (in this case the class to which a child belongs) is small, but between levels of a contextual variable (comparing classes) is large. As such, the ICC is a good gauge of whether a contextual variable has an effect on the outcome.

OLIVER TWISTED

Please Sir, can I have some more ... ICC?

'I have a dependency on gruel', whines Oliver. 'Maybe I could measure this dependency if I knew more about the ICC.' Well, you're so high on gruel, Oliver, that you have rather missed the point. Still, I did write an article on the ICC once upon a time (Field, 2005a) and it's reproduced in the additional web material for your delight and amusement.

19.2.2. Benefits of multilevel models ②

Multilevel linear models have numerous uses. To convince you that trawling through this chapter is going to reward you with statistical possibilities beyond your wildest dreams, here are just a few (slightly overstated) benefits of multilevel models:

- **Cast aside the assumption of homogeneity of regression slopes.** We saw in Chapter 11 that when we use analysis of covariance we have to assume that the relationship between our covariate and our outcome is the same across the different groups that make up our predictor variable. However, this doesn't always happen. Luckily, in multilevel models we can explicitly model this variability in regression slopes, thus overcoming this inconvenient problem.
- **Say 'bye bye' to the assumption of independence.** We saw in Chapter 10 that when we use independent ANOVA we have to assume that the different cases of data are independent. If this is not true, little lizards climb out of your mattress while you're asleep and eat you. Again, multilevel models are specifically designed to allow you to model these relationships between cases. Also, in Chapter 7 we saw that multiple regression relies on having independent observations. However, there are situations in which you might want to measure someone on more than one occasion (i.e., over time). Ordinary regression turns itself into cheese and hides in the fridge at the prospect of cases of data that are related. Multilevel models eat these data for breakfast, with a piece of regression-flavoured cheese.
- **Laugh in the face of missing data.** I've spent a lot of this book extolling the virtues of balanced designs and not having missing data. Regression, ANOVA, ANCOVA and most of the other tests we have covered do strange things when data are missing or the design is not balanced. This can be a real pain. Missing data are a particular problem within clinical trials because it is common to attempt to collect follow-up data, often many months after treatment has ended when patients might be difficult to track down. Of course, there are ways to correct for and impute missing data, but these techniques are often quite complicated (Yang, Li, & Shoptaw, 2008), therefore, often when using repeated-measures designs if a single time point is missing the whole case usually needs to be deleted; missing data leads to more data being deleted. Multilevel models do not require complete data sets and so when data are missing for one time point they do not need to be imputed, nor does the whole case need to be deleted. Instead parameters can be estimated successfully with the available data, which offers a relatively easy solution to dealing with missing data. It is important to stress that no statistical procedure can overcome data that are missing. Good methods, designs and research execution should be used to minimize missing values, and reasons for missing values should always be explored. It is just that when using traditional statistical procedures for repeated-measures data additional procedures to account for missing data are usually necessary and can be problematic.

I think you'll agree that multilevel models are pretty funky. 'Is there anything they can't do?' I hear you cry. Well, no, not really.

19.3. Theory of multilevel linear models ③

The underlying theory of multilevel models is very complicated indeed – far too complicated for my little peanut of a brain to comprehend. Fortunately, the advent of computers and software like R makes it possible for feeble-minded individuals such as myself to take advantage of this wonderful tool without actually needing to know the maths. Better still, this means I can get away with not explaining the maths (and really, I'm not kidding, I don't understand any of it). What I will do, though, is try to give you a flavour of what multilevel models are and what they do by describing the key concepts within the framework of linear models that has permeated this whole book. I also want to remind you that if you have worked through Chapters 13 and 14 then you have already done a multilevel model and used the *lme()* function that we discuss in this chapter, because we used it to analyse repeated-measures designs. In these repeated-measures designs can be thought of as a two-level hierarchy in which scores (level 1) are nested within participants (level 2).

19.3.1. An example ②

Throughout the first part of the chapter we will use an example to illustrate some of the concepts in multilevel models. Cosmetic surgery is on the increase at the moment. In the USA, there was a 1600% increase in cosmetic surgical and non-surgical treatments between 1992 and 2002, and in 2004, 65,000 people in the UK underwent privately and publicly funded operations (Kellett, Clarke, & McGill, 2008). With the increasing popularity of this surgery, many people are starting to question the motives of those who want to go under the knife. There are two main reasons to have cosmetic surgery: (1) to help a physical problem such as having breast reduction surgery to relieve backache; and (2) to change your external appearance, for example by having a face lift. Related to this second point, there is even some case for arguing that cosmetic surgery could be performed as a psychological intervention: to improve self-esteem (Cook, Rosser, & Salmon, 2006; Kellett et al., 2008). The main example for this chapter looks at the effects of cosmetic surgery on quality of life. The variables in the data file are (**Cosmetic Surgery.dat**):

- **Post_QoL**: This is a measure of quality of life after the cosmetic surgery. This is our outcome variable.
- **Base_QoL**: We need to adjust our outcome for quality of life before the surgery.
- **Surgery**: This variable is a dummy variable that specifies whether the person has undergone cosmetic surgery (1) or whether they are on the waiting list (0), which acts as our control group.
- **Surgery_Text**: This variable is the same as above but specifies group membership as text (we will use this variable when we create graphs but not for the main analysis).
- **Clinic**: This variable specifies which of 10 clinics the person attended to have their surgery.
- **Age**: This variable tells us the person's age in years.
- **BDI**: It is becoming increasingly apparent that people volunteering for cosmetic surgery (especially when the surgery is purely for vanity) might have very different personality profiles than the general public (Cook, Rossera, Toone, James, & Salmon, 2006). In particular, these people might have low self-esteem or be depressed. When looking at quality of life it is important to assess natural levels of depression, and this variable used the Beck Depression Inventory (BDI) to do just that.
- **Reason**: This dummy variable specifies whether the person had/is waiting to have surgery purely to change their appearance (0), or because of a physical reason (1).
- **Reason_Text**: This variable is the same as above but contains text to define each group rather than a number.
- **Gender**: This variable simply specifies whether the person was a man (1) or a woman (0).

When conducting hierarchical models we generally work up from a very simple model to more complicated models, and we will take that approach in this chapter. In doing so I hope to illustrate multilevel modelling by attaching it to frameworks that you already understand, such as ANOVA and ANCOVA.

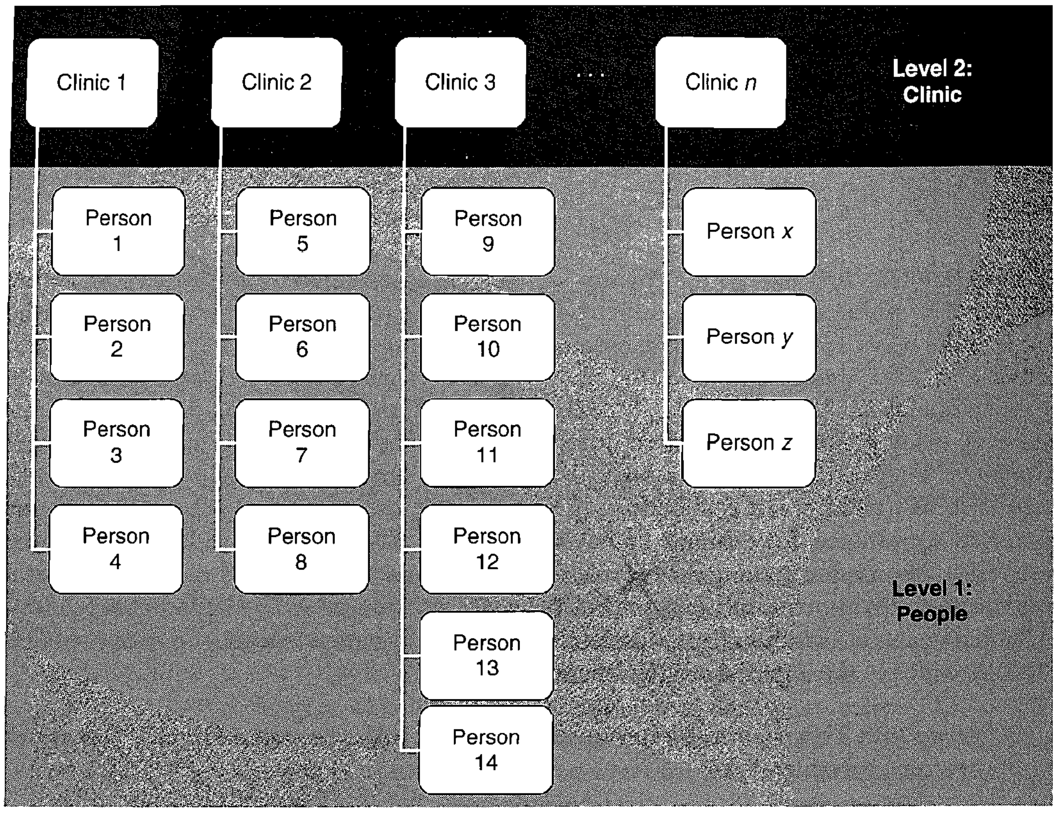


FIGURE 19.5 Diagram to show the hierarchical structure of the cosmetic surgery data set. People are clustered within clinics. Note that for each person there would be a series of variables measured: surgery, BDI, age, gender, reason and pre-surgery quality of life

Figure 19.5 shows the hierarchical structure of the data. Essentially, people being treated in the same surgeries are not independent of each other because they will have had surgery from the same surgeon. Surgeons will vary in how good they are, and quality of life will to some extent depend on how well the surgery went (if they did a nice neat job then quality of life should be higher than if they left you with unpleasant scars). Therefore, people within clinics will be more similar to each other than people in different clinics. As such, the person undergoing surgery is the level 1 variable, but there is a level 2 variable, a variable higher in the hierarchy, which is the clinic attended.

19.3.2. Fixed and random coefficients ③

Throughout this book we have discussed effects and variables, and these concepts should be very familiar to you by now. However, we have viewed these effects and variables in a relatively simple way: we have not distinguished between whether something is fixed or random.

What we mean by 'fixed' and 'random' can be a bit confusing because the terms are used in a variety of contexts. You hear people talk about **fixed effects** and **random effects.** An effect in an experiment is said to be a fixed effect if all possible treatment conditions that a researcher is interested in are present in the experiment. An effect is said to be random if the experiment contains only a random sample of possible treatment conditions. This distinction is important because fixed effects can be generalized only to the situations in your experiment, whereas random effects can be generalized beyond the treatment conditions in the experiment (provided that the treatment conditions are representative). For example, in our Viagra example from Chapter 10, the effect is fixed if we say that we are interested only in the three conditions that we had (placebo, low dose and high dose) and we can generalize our findings only to the situation of a placebo, low dose and high dose. However, if we were to say that the three doses were only a sample of possible doses (maybe we could have tried a very high dose), then it is a random effect and we can generalize beyond just placebos, low doses and high doses. All of the effects in this book so far we have treated as fixed effects. The vast majority of academic research that you read will treat variables as fixed effects.

People also talk about **fixed variables** and **random variables**. A fixed variable is one that is not supposed to change over time (e.g., for most people their gender is a fixed variable – it never changes), whereas a random one varies over time (e.g., your weight is likely to fluctuate over time).

In the context of multilevel models we need to make a distinction between **fixed coefficients** and **random coefficients**. In the regressions, ANOVAs and ANCOVAs throughout this book we have assumed that the regression parameters are fixed. We have seen numerous times that a linear model is characterized by two things: the intercept, b_0, and the slope, b_1:

$$Y_i = b_0 + b_1 X_{1i} + \varepsilon_i$$

Note that the outcome (Y), the predictor (X) and the error (ε) all vary as a function of i, which normally represents a particular case of data. In other words, it represents the level 1 variable. If, for example, we wanted to predict Sam's score, we could replace the is with her name:

$$Y_{\text{Sam}} = b_0 + b_1 X_{1,\text{Sam}} + \varepsilon_{\text{Sam}}$$

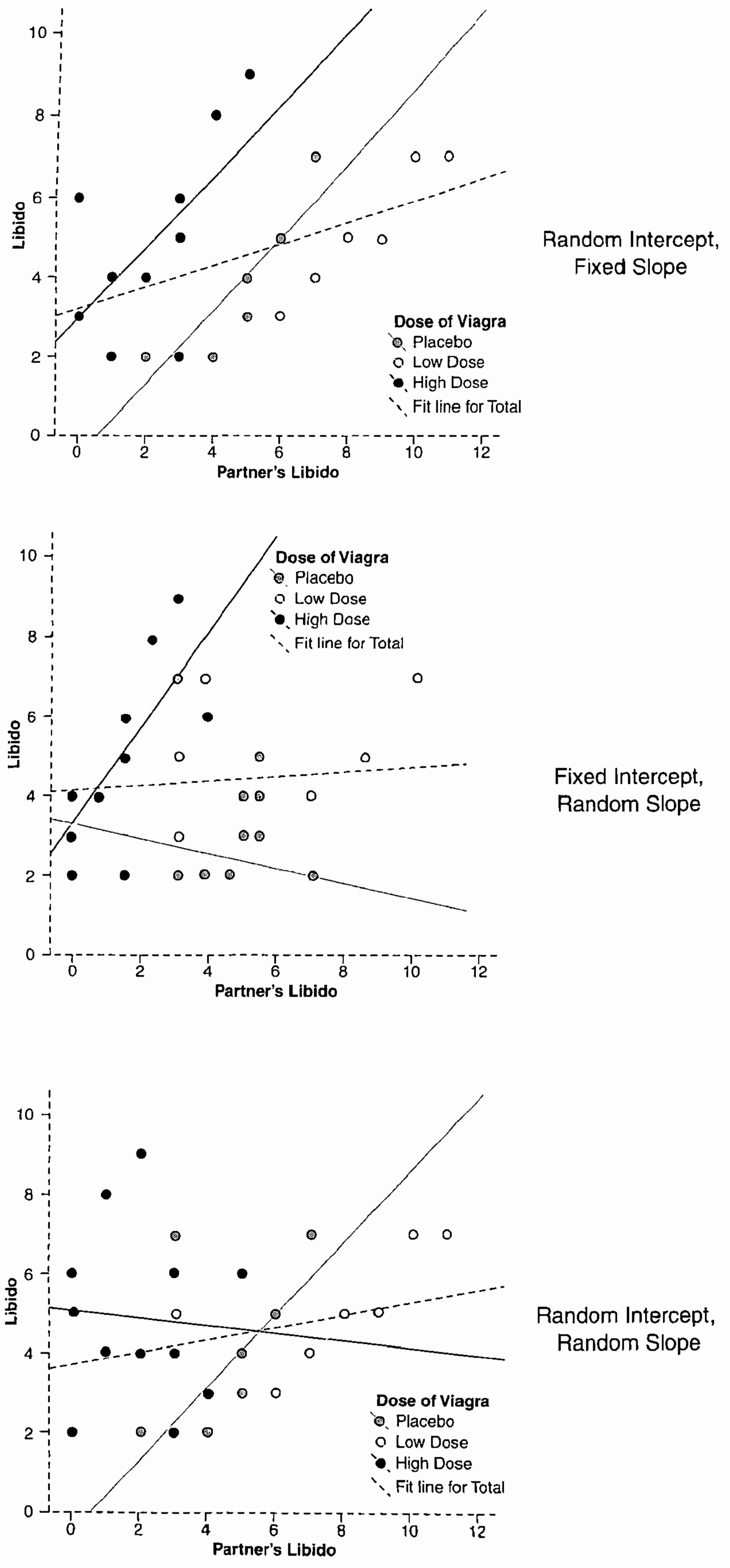
This is just some basic revision. Now, when we do a regression like this we assume that the bs are fixed and we estimate them from the data. In other words, we're assuming that the model holds true across the entire sample and that for every case of data in the sample we can predict a score using the same values of the gradient and intercept. However, we can also conceptualize these parameters as being random.[2] If we say that a parameter is random then we assume not that it is a fixed value, but that its value can vary. Up until now we have thought of regression models as having **fixed intercepts** and **fixed slopes**, but this opens up three new possibilities for us that are shown in Figure 19.6. This figure uses the data from our ANCOVA example in Chapter 11 and shows the relationship between a person's libido and that of their partner overall (the dashed line) and separately for the three groups in the study (a placebo group, a group that had a low dose of Viagra and a group that had a high dose).

19.3.2.1. The random intercept model ③

The simplest way to introduce random parameters into the model is to assume that the intercepts vary across contexts (or groups) – because the intercepts vary, we call them **random intercepts**. For our libido data this is like assuming that the relationship between libido and partner's libido is the same in the placebo, low- and high-dose groups (i.e., the slope is the same), but that the models for each group are in different locations (i.e., the intercepts are different). This is shown in the top panel of Figure 19.6, in which the models within the different contexts (colours) have the same shape (slope) but are located in different geometric space (they have different intercepts).

[2] In a sense 'random' isn't an intuitive term for us non-statisticians because it implies that values are plucked out of thin air (randomly selected). However, this is not the case – they are carefully estimated just as fixed parameters are.

FIGURE 19.6 Data sets showing an overall model (dashed line) and the models for separate contexts within the data (i.e., groups of cases)



19.3.2.2. Random slope model ③

We can also assume that the slopes vary across contexts – i.e., we assume **random slopes.** For our libido data this is like assuming that the relationship between libido and partner's libido is different in the placebo, low- and high-dose groups (i.e., the slopes are different), but that the models for each group are fixed at the same geometric location (i.e., the intercepts are the same). This is what happens when we violate the assumption of homogeneity of regression slopes in ANCOVA. Homogeneity of regression slopes is the assumption that regression slopes are the same across contexts. If this assumption is not tenable then we can use a multilevel model to explicitly estimate that variability in slopes. This is shown in the middle panel of Figure 19.6 in which the models within the different contexts (colours) converge on a single intercept but have different slopes. It's worth noting that it would be unusual in reality to assume random slopes without also assuming random intercepts because variability in the nature of the relationship (slopes) would normally create variability in the overall level of the outcome variable (intercepts). Therefore, if you assume that slopes are random you would normally also assume that intercepts are random.

19.3.2.3. The random intercept and slope model ③

The most realistic situation is to assume that both intercepts and slopes vary around the overall model. This is shown in the bottom panel of Figure 19.6 in which the models within the different contexts (colours) have different slopes but are also located in different geometric space and so have different intercepts.

19.4. The multilevel model ④

We have seen conceptually what a random intercept, random slope and random intercept and slope model looks like. Now let's look at how we actually represent the models. To keep things concrete, let's use our example. For the sake of simplicity, let's imagine first that we wanted to predict someone's quality of life (QoL) after cosmetic surgery. We can represent this as a linear model as follows:

$$\text{QoL After Surgery}_i = b_0 + b_1\text{Surgery}_i + \varepsilon_i \tag{19.1}$$

We have seen equations like this many times and it represents a linear model: regression, a *t*-test (in this case) and ANOVA. In this example, we had a contextual variable, which was the clinic in which the cosmetic surgery was conducted. We might expect the effect of surgery on quality of life to vary as a function of which clinic the surgery was conducted at because surgeons will differ in their skill. This variable is a level 2 variable. As such we could allow the model that represents the effect of surgery on quality of life to vary across the different contexts (clinics). We can do this by allowing the intercepts to vary across clinics, or by allowing the slopes to vary across clinics or by allowing both to vary across clinics.

To begin with, let's say we want to include a random intercept for quality of life. All we do is add a component to the intercept that measures the variability in intercepts, u_{0j}. Therefore, the intercept changes from b_0 to become $(b_0 + u_{0j})$. This term estimates the intercept of the overall model fitted to the data, b_0, and the variability of intercepts around that overall model, u_{0j}. The overall model becomes:[3]

$$Y_{ij} = (b_0 + u_{0j}) + b_1X_{ij} + \varepsilon_{ij} \tag{19.2}$$

[3] Some people use gamma (γ), not *b*, to represent the parameters, but I prefer *b* because it makes the link to the other linear models that we have used in this book clearer.

The *j*s in the equation reflect levels of the variable over which the intercept varies (in this case the clinic) – the level 2 variable. Another way that we could write this is to take out the error terms so that it looks like an ordinary regression equation except that the intercept has changed from a fixed, b_0, to a random one, b_{0j}, which is defined in a separate equation:

$$\begin{aligned} Y_{ij} &= b_{0j} + b_1 X_{ij} + \varepsilon_{ij} \\ b_{0j} &= b_0 + u_{0j} \end{aligned} \qquad (19.3)$$

Therefore, if we want to know the estimated intercept for clinic 7, we simply replace the *j* with 'clinic 7' in the second equation:

$$b_{0\text{Clinic 7}} = b_0 + u_{0\text{Clinic 7}}$$

If we want to include random slopes for the effect of surgery on quality of life, then all we do is add a component to the slope of the overall model that measures the variability in slopes, u_{1j}. Therefore, the gradient changes from b_1 to become $(b_1 + u_{1j})$. This term estimates the slope of the overall model fitted to the data, b_1, and the variability of slopes in different contexts around that overall model, u_{1j}. The overall model becomes (compare to the random intercept model above):

$$Y_{ij} = b_0 + (b_1 + u_{1j})X_{ij} + \varepsilon_{ij} \qquad (19.4)$$

Again we can take the error terms out into a separate equation to make the link to a familiar linear model even clearer. It now looks like an ordinary regression equation except that the slope has changed from a fixed, b_1, to a random one, b_{1j}, which is defined in a separate equation:

$$\begin{aligned} Y_{ij} &= b_{0i} + b_{1j} X_{ij} + \varepsilon_{ij} \\ b_{1j} &= b_1 + u_{1j} \end{aligned} \qquad (19.5)$$

If we want to model a situation with random slopes *and* intercepts, then we combine the two models above. We still estimate the intercept and slope of the overall model (b_0 and b_1) but we also include the two terms that estimate the variability in intercepts, u_{0j}, and slopes, u_{1j}. The overall model becomes (compare to the two models above):

$$Y_{ij} = (b_0 + u_{0j}) + (b_1 + u_{1j})X_{ij} + \varepsilon_{ij} \qquad (19.6)$$

We can link this more directly to a simple linear model if we take some of these extra terms out into separate equations. We could write this model as a basic linear model, except we've replaced our fixed intercept and slope (b_0 and b_1) with their random counterparts (b_{0j} and b_{1j}):

$$\begin{aligned} Y_{ij} &= b_{0j} + b_{1j} X_{ij} + \varepsilon_{ij} \\ b_{0j} &= b_0 + u_{0j} \\ b_{1j} &= b_1 + u_{1j} \end{aligned} \qquad (19.7)$$

The take-home point is that we're not doing anything terribly different from the rest of the book: it's basically just a posh regression.

Now imagine we wanted to add in another predictor, for example quality of life before surgery. Knowing what we do about multiple regression, we shouldn't be invading the personal space of the idea that we can simply add this variable in with an associated beta:

$$\text{QoL After Surgery}_i = b_0 + b_1\text{Surgery}_i + b_2\text{QoL Before Surgery}_i + \varepsilon_i \tag{19.8}$$

This is all just revision of ideas from earlier in the book. Remember also that the *i* represents the level 1 variable, in this case the people we tested. Therefore, we can predict a given person's quality of life after surgery by replacing the *i* with their name:

$$\text{QoL After}_{\text{Sam}} = b_0 + b_1\text{Surgery}_{\text{Sam}} + b_2\text{QoL Before}_{\text{Sam}} + \varepsilon_{\text{Sam}}$$

Now, if we want to allow the intercept of the effect of surgery on quality of life after surgery to vary across contexts then we simply replace b_0 with b_{0j}. If we want to allow the slope of the effect of surgery on quality of life after surgery to vary across contexts then we replace b_1 with b_{1j}. So, even with a random intercept and slope, our model stays much the same:

$$\begin{aligned} \text{QoL After}_{ij} &= b_{0j} + b_{1j}\text{Surgery}_{ij} + b_2\text{QoL Before}_{ij} + \varepsilon_{ij} \\ b_{0j} &= b_0 + u_{0j} \\ b_{1j} &= b_1 + u_{1j} \end{aligned} \tag{19.9}$$

Remember that the *j* in the equation relates to the level 2 contextual variable (clinic in this case). So, if we wanted to predict someone's score we wouldn't just do it from their name, but also from the clinic they attended. Imagine our guinea pig Sam had her surgery done at clinic 7, then we could replace the *i*s and *j*s as follows:

$$\begin{aligned} &\text{QoL After Surgery}_{\text{Sam, Clinic 7}} \\ &\quad = b_{0\text{Clinic 7}} + b_{1\text{Clinic7}}\text{Surgery}_{\text{Sam, Clinic7}} \\ &\quad\quad + b_2\text{QoL Before Surgery}_{\text{Sam, Clinic7}} + \varepsilon_{\text{Sam, Clinic7}} \end{aligned}$$

I want to sum up by just reiterating that all we're really doing in a multilevel model is a fancy regression in which we allow either the intercepts or slopes, or both, to vary across different contexts. All that really changes is that for every parameter that we allow to be random, we get an estimate of the variability of that parameter as well as the parameter itself. So, there isn't anything terribly complicated; we can add new predictors to the model and for each one decide whether its regression parameter is fixed or random.

19.4.1. Assessing the fit and comparing multilevel models ④

As in logistic regression (Chapter 8) the overall fit of a multilevel model is tested using a chi-square likelihood ratio test (see section 18.4.3) and R reports the −2log-likelihood (−2*LL*, see section 8.3.1). Essentially, the smaller the value of the log-likelihood, the better. R also produces two adjusted versions of the log-likelihood value, both of which were described briefly in section 7.6.3. Both of these can be interpreted in the same way as the log-likelihood, but they have been corrected for various things:

- *Akaike's information criterion* (**AIC**): This is basically a goodness-of-fit measure that is corrected for model complexity. That just means that it takes into account how many parameters have been estimated.
- *Schwarz's Bayesian criterion* (**BIC**): This statistic is comparable to the AIC, although it is slightly more conservative (it corrects more harshly for the number of parameters being estimated). It should be used when sample sizes are large and the number of parameters is small.

Neither the AIC or BIC are intrinsically interpretable (it's not meaningful to talk about their values being large or small *per se*); however, they are useful as a way of comparing models. The value of AIC and BIC can be compared to their equivalent values in other models. In all cases smaller values mean better-fitting models.

Many writers recommend building up multilevel models starting with a 'basic' model in which all parameters are fixed and then adding in random coefficients as appropriate and exploring confounding variables (Raudenbush & Bryk, 2002; Twisk, 2006). One advantage of doing this is that you can compare the fit of the model as you make parameters random, or as you add in variables. To compare models we simply subtract the log-likelihood of the new model from the value for the old:

$$\begin{aligned}\chi^2_{\text{Change}} &= (-2\text{Log-Likelihood}_{\text{Old}}) - (-2\text{Log-Likelihood}_{\text{New}}) \\ df_{\text{Change}} &= \text{Number of Parameters}_{\text{Old}} - \text{Number of Parameters}_{\text{New}}\end{aligned} \tag{19.10}$$

This equation is the same as equations (18.5) and (8.6). There are two caveats to this equation: (1) it works only if full maximum-likelihood estimation is used (and not restricted maximal likelihood – see R's Souls' Tip 19.1); and (2) the new model must contain all of the effects of the older model.

19.4.2. Types of covariance structures ④

If you have any random effects or repeated measures in your multilevel model then you have to decide upon the *covariance structure* of your data. If you have random effects and repeated measures then you can specify different covariance structures for each. The covariance structure simply specifies the form of the variance–covariance matrix (a matrix in which the diagonal elements are variances and the off-diagonal elements are covariances). There are various forms that this matrix could take and we have to tell **R** what form we think it *does* take. Of course we might not know what form it takes (most of the time we'll be taking an educated guess), so it is sometimes useful to run the model with different covariance structures defined and use the goodness-of-fit indices (the AIC and BIC) to see whether changing the covariance structure improves the fit of the model (remember that a smaller value of these statistics means a better-fitting model).

The covariance structure is important because **R** uses it as a starting point to estimate the model parameters. As such, you will get different results depending on which covariance structure you choose. If you specify a covariance structure that is too simple then you are more likely to make a Type I error (finding a parameter is significant when in reality it is not), but if you specify one that is too complex then you run the risk of a Type II error (finding parameters to be non-significant when in reality they are). **R** can implement many different covariance structures. We will look at four of the commonest covariance structures to give you a feel for what they are and when they should be used. In each case I use a representation of the variance–covariance matrix to illustrate. With all of these matrices you could imagine that the rows and columns represents four different clinics in our cosmetic surgery data:

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Variance components: This covariance structure is very simple and assumes that all random effects are independent (this is why all of the covariances in the matrix are 0). Variances of random effects are assumed to be the same (hence why they are 1 in the matrix) and sum to the variance of the outcome variable. This covariance structure is sometimes called the independence model.

$$\begin{pmatrix} \sigma_1^2 & 0 & 0 & 0 \\ 0 & \sigma_2^2 & 0 & 0 \\ 0 & 0 & \sigma_3^2 & 0 \\ 0 & 0 & 0 & \sigma_4^2 \end{pmatrix}$$

Diagonal: This variance structure is like variance components except that variances are assumed to be heterogeneous (this is why the diagonal of the matrix is made up of different variance terms). This structure again assumes that variances are independent and, therefore, that all of the covariances are 0.

$$\begin{pmatrix} 1 & \rho & \rho^2 & \rho^3 \\ \rho & 1 & \rho & \rho^2 \\ \rho^2 & \rho & 1 & \rho \\ \rho^3 & \rho^2 & \rho & 1 \end{pmatrix}$$

AR(1): This stands for first-order autoregressive structure. In layman's terms this means that the relationship between variances changes in a systematic way. If you imagine the rows and columns of the matrix to be points in time, then it assumes that the correlation between repeated measurements is highest at adjacent time points. So, in the first column, the correlation between time points 1 and 2 is ρ; let's assume that this value is .3. As we move to time point 3, the correlation between time point 1 and 3 is ρ^2, or .09. In other words, it has decreased: scores at time point 1 are more related to scores at time 2 than they are to scores at time 3. At time 4, the correlation goes down again to ρ^3 or .027. So, the correlations between time points next to each other are assumed to be ρ, scores two intervals apart are assumed to have correlations of ρ^2, and scores three intervals apart are assumed to have correlations of ρ^3. So the correlation between scores gets smaller over time. Variances are assumed to be homogeneous, but there is a version of this covariance structure where variance can be heterogeneous. This structure is often used for repeated-measures data (especially when measurements are taken over time such as in growth models).

$$\begin{pmatrix} \sigma_1^2 & \sigma_{21} & \sigma_{31} & \sigma_{41} \\ \sigma_{21} & \sigma_2^2 & \sigma_{32} & \sigma_{42} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 & \sigma_{43} \\ \sigma_{41} & \sigma_{42} & \sigma_{43} & \sigma_4^2 \end{pmatrix}$$

Unstructured: This covariance structure is completely general. Covariances are assumed to be completely unpredictable: they do not conform to a systematic pattern.

CRAMMING SAM'S TIPS **Multilevel models**

- Multilevel models should be used to analyse data that have a hierarchical structure. For example, you might measure depression after psychotherapy. In your sample, patients will see different therapists within different clinics. This is a three-level hierarchy with depression scores from patients (level 1), nested within therapists (level 2) who are themselves nested within clinics (level 3).
- Hierarchical models are just like regression, except that you can allow parameters to vary (this is called a random effect). In ordinary regression, parameters generally are a fixed value estimated from the sample (a fixed effect).
- If we estimate a linear model within each context (the therapist or clinic, to use the example above) rather than the sample as whole, then we can assume that the intercepts of these models vary (a random intercepts model), or that the slopes of these models differ (a random slopes model) or that both vary.
- We can compare different models (assuming that they differ in only one additional parameter) by looking at the difference in the −*2LL*. Usually we would do this when we have changed only one parameter (added one new thing to the model).
- For any model we have to assume a covariance structure. For random intercepts models the default of *variance components* is fine, but when slopes are random an *unstructured* covariance structure is often assumed. When data are measured over time an autoregressive structure (AR(1)) is often assumed.

19.5. Some practical issues ③

19.5.1. Assumptions ③

Multilevel linear models are an extension of regression, so all of the assumptions for regression apply to multilevel models (see section 7.7.2). There is a caveat, though, which is that the assumptions of independence and independent errors can sometimes be solved by a multilevel model because the purpose of this model is to factor in the correlations between cases caused by higher-level variables. As such, if a lack of independence is being caused by a level 2 or level 3 variable then a multilevel model should make this problem go away (although not always). As such, try to check the usual assumptions in the usual way.

There are two additional assumptions in multilevel models that relate to the random coefficients. These coefficients are assumed to be normally distributed around the overall model. So, in a random intercepts model the intercepts in the different contexts are assumed to be normally distributed around the overall model. Similarly, in a random slopes model, the slopes of the models in different contexts are assumed to be normally distributed.

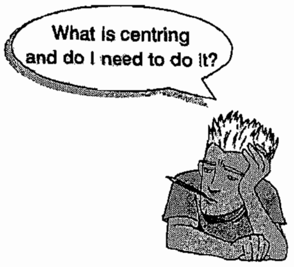
Also it's worth mentioning that multicollinearity can be a particular problem in multilevel models if you have interactions that cross levels in the data hierarchy (cross-level interactions). However, centring predictors can help matters enormously (Kreft & de Leeuw, 1998), and we will see how to centre predictors in section 19.5.3.

19.5.2. Sample size and power ③

As you might well imagine, the situation with power and sample size is very complex indeed. One complexity is that we are trying to make decisions about our power to detect both fixed and random effects coefficients. Kreft and de Leeuw (1998) do a tremendous job of making sense of things for us. Essentially, the take-home message is the more data, the better. As more levels are introduced into the model, more parameters need to be estimated and the larger the sample sizes need to be. Kreft and de Leeuw conclude that if you are looking for cross-level interactions then you should aim to have more than 20 contexts (groups) in the higher-level variable, and that group sizes 'should not be too small'. They conclude by saying that there are so many factors involved in multilevel analysis that it is impossible to produce any meaningful rules of thumb.

Twisk (2006) agrees that the number of contexts relative to individuals within those contexts is important. He also points out that standard sample size and power calculations can be used but then 'corrected' for the multilevel component of the analysis (by factoring, among other things, the intraclass correlation). However, there are two corrections that he discusses that yield very different sample sizes! He recommends using sample size calculations with caution.

19.5.3. Centring variables ④



Centring refers to the process of transforming a variable into deviations around a fixed point. This fixed point can be any value that you choose, but typically we use the grand mean. We have already come across a form of centring way back in Chapter 1, when we discovered how to compute *z*-scores. When we calculate a *z*-score we take each score and subtract from it the mean of all scores (this centres the values at 0), and then divide by the standard deviation (this changes the units of measurement to standard deviations). When we centre a variable around the mean we simply subtract the mean from all of the scores: this centres the variables around 0.

There are two forms of centring that are typically used in multilevel modelling: **grand mean centring** and **group mean centring**. Grand mean centring means that for a given variable we take each score and subtract from it the mean of all scores (for that variable). Group mean centring means that for a given variable we take each score and subtract from it the mean of the scores (for that variable) within a given group. In both cases it is usually only level 1 predictors that are centred (in our cosmetic surgery example this would be predictors such as age, BDI and pre-surgery quality of life). If group mean centring is used then a level 1 variable is typically centred around means of a level 2 variable (in our cosmetic surgery data this would mean that, for example, the age of a person would be centred around the mean age for the clinic at which the person had their surgery).

Centring can be used in ordinary multiple regression too, and because this form of regression is already familiar to you I'd like to begin by looking at the effects of centring in regression. In multiple regression the intercept represents the value of the outcome when all of the predictors take a value of 0. There are some predictors for which a value of 0 makes little sense. For example, if you were using heart rate as a predictor variable then a value of 0 would be meaningless (no one will have a heart rate of 0 unless they are dead). As such, the intercept in this case has no real-world use: why would you want to know the value of the outcome when heart rate was 0 given than no alive person would even have a heart rate that low? Centring heart rate around its mean changes the meaning of the intercept. The intercept becomes the value of the outcome when heart rate is its average value. In more general terms, if all predictors are centred around their mean then the intercept is the value of the outcome when all predictors are the value of their mean. Centring can, therefore, be a useful tool for interpretation when a value of 0 for the predictor is meaningless.

The effect of centring in multilevel models, however, is much more complicated. There are some excellent reviews that look in detail at the effects of centring on multilevel models

(Enders & Tofighi, 2007; Kreft & de Leeuw, 1998; Kreft, de Leeuw, & Aiken, 1995), and here I will just give a very basic précis of what they say. Essentially if you fit a multilevel model using the raw score predictors and then fit the same model but with grand mean centred predictors then the resulting models are equivalent. By this, I mean that they will fit the data equally well, have the same predicted values, and the residuals will be the same. The parameters themselves (the *b*s) will, of course, be different but there will be a direct relationship between the parameters from the two models (i.e., they can be directly transformed into each other). Therefore, grand mean centring doesn't change the model, but it would change your interpretation of the parameters (you can't interpret them as though they are raw scores). When group mean centring is used the picture is much more complicated. In this situation the raw score model is not equivalent to the centred model in either the fixed part or the random part. One exception is when only the intercept is random (which arguably is an unusual situation), and the group means are reintroduced into the model as level 2 variables (Kreft & de Leeuw, 1998).

The decision about whether to centre or not is quite complicated and you really need to make the decision yourself in a given analysis. Centring can be a useful way to combat multicollinearity between predictor variables. It's also helpful when predictors do not have a meaningful zero point. Finally, multilevel models with centred predictors tend to be more stable, and estimates from these models can be treated as more or less independent of each other, which might be desirable. If group mean centring is used then the group means should be reintroduced as a level 2 variable unless you want to look at the effect of your 'group' or level 2 variable uncorrected for the mean effect of the centred level 1 predictor, such as when fitting a model when time is your main explanatory variable (Kreft & de Leeuw, 1998).

The question arises of whether grand mean or group mean centring is 'better'. People doing statistics often fixate on their being a 'best' way to do things, but the 'best' method often depends on what it is that you're actually trying to do. Centring is a good example. Some people make a decision about whether to use group or grand mean centring based on some statistical criterion; however, there is no statistically correct choice between not centring, group mean centring and grand mean centring (Kreft et al., 1995). Instead, Enders and Tofighi (2007) recommend making decisions based on the substantive research question. In short, they make four recommendations when analysing data with a two-level hierarchy: (1) group mean clustering should be used if the primary interest is in an association between variables measured at level 1 (i.e., the aforementioned relationship between surgery and quality of life after surgery); (2) grand mean centring is appropriate when the primary interest is in the level 2 variable but you want to control for the level 1 covariate (i.e., you want to look at the effect of clinic on quality of life after surgery while controlling for the type of surgery); (3) both types of centring can be used to look at the differential influence of a variable at level 1 and 2 (i.e., is the effect of surgery on quality of life post-surgery different at the clinic level to the client level?); and (4) group mean centring is preferable for examining cross-level interactions (e.g., the interactive effect of clinic and surgery on quality of life after surgery).

OLIVER TWISTED

Please Sir, can I have some more … centring?

'Recentgin', babbles Oliver as he stumbles drunk out of Mrs Moonshine's alcohol emporium. 'I've had some recent gin.' I think you mean *centring*, Oliver, not *recentgin*. If you want to know how to centre your variables using **R**, then the additional material for this chapter on the companion website will tell you.

19.6. Multilevel modelling in R ④

Multilevel modelling can be done with specialist software such as MLwiN and HLM. There are several excellent books that compare R with various other packages (Tabachnick & Fidell, 2001; Twisk, 2006). R is more versatile than packages such as SPSS in that it can do multilevel modelling when the outcome variable is categorical. However, the packages that do multilevel models in R do not currently produce bootstrap estimates of the model parameters, and these can be a very useful way to circumvent pesky distributional assumptions (see section 5.8.4).

We saw in section 19.4.1 that it is useful to build up models starting with a 'basic' model in which all parameters are fixed and then add random coefficients as appropriate before exploring confounding variables. We will take this approach in our example.

19.6.1. Packages for multilevel modelling in R ①

There are several packages that can be used for multilevel models. Two of the most used are: *nlme* (Pinheiro, Bates, DebRoy, Sarkar, & R Development Core Team, 2010) and *lme4* (Bates & Maechler, 2010). I am going to focus on the package *nlme* (*non linear mixed effect*) because, unlike *lme4*, it enables you to model the covariance structure, which will be useful when we come to look at growth models towards the end of the chapter.

For the examples in this chapter you will need the packages *car* (to recode variables), *nlme* (for the multilevel analysis), *ggplot2* (for graphs), and *reshape* (to restructure the data). If you do not have these packages installed, you can install them by executing the following commands:

```
install.packages("car"); install.packages("ggplot2"); install.
packages("nlme"); install.packages("reshape")
```

You then need to load these packages by executing the commands:

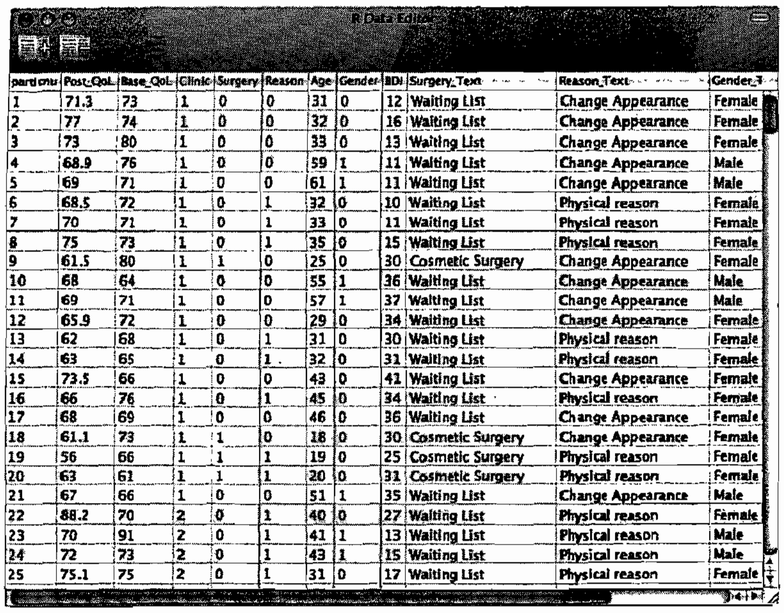
```
library(car); library(ggplot2); library(nlme); library(reshape)
```

19.6.2. Entering the data ②

Data entry depends a bit on the type of multilevel model that you wish to run: the data layout is slightly different when the same variables are measured at several points in time. However, we will look at the case of repeated-measures data in a second example. In this first example, the situation we have is very much like multiple regression in that data from each person who had surgery are not measured over multiple time points. Figure 19.7 shows the data layout. Each row represents a case of data (in this case a person who had surgery). Their scores on the various variables are simply entered in different columns. So, for example, the first person was 31 years old, had a BDI score of 12, was in the waiting list control group (**Surgery** = 0) at clinic 1, was female (**Gender** = 0) and was waiting for surgery to change her appearance (**Reason** = 0).

To access these data we need to create a dataframe, which I have called *surgeryData*, that contains the data from the file **CosmeticSurgery.dat**. This file stores the data as tab-delimited text, so we can import it into the dataframe using the following command (I'm assuming as always that you have set the working directory to be where the file is stored):

```
surgeryData = read.delim("Cosmetic Surgery.dat",  header = TRUE)
```



partcnu	Post_QoL	Base_QoL	Clinic	Surgery	Reason	Age	Gender	BDI	Surgery_Text	Reason_Text	Gender_T
1	71.3	73	1	0	0	31	0	12	Waiting List	Change Appearance	Female
2	77	74	1	0	0	32	0	16	Waiting List	Change Appearance	Female
3	73	80	1	0	0	33	0	13	Waiting List	Change Appearance	Female
4	68.9	76	1	0	0	59	1	11	Waiting List	Change Appearance	Male
5	69	71	1	0	0	61	1	11	Waiting List	Change Appearance	Male
6	68.5	72	1	0	1	32	0	10	Waiting List	Physical reason	Female
7	70	71	1	0	1	33	0	11	Waiting List	Physical reason	Female
8	75	73	1	0	1	35	0	15	Waiting List	Physical reason	Female
9	61.5	80	1	1	0	25	0	30	Cosmetic Surgery	Change Appearance	Female
10	68	64	1	0	0	55	1	36	Waiting List	Change Appearance	Male
11	69	71	1	0	0	57	1	37	Waiting List	Change Appearance	Male
12	65.9	72	1	0	0	29	0	34	Waiting List	Change Appearance	Female
13	62	68	1	0	1	31	0	30	Waiting List	Physical reason	Female
14	63	65	1	0	1	32	0	31	Waiting List	Physical reason	Female
15	73.5	66	1	0	0	43	0	41	Waiting List	Change Appearance	Female
16	66	76	1	0	1	45	0	34	Waiting List	Physical reason	Female
17	68	69	1	0	0	46	0	36	Waiting List	Change Appearance	Female
18	61.1	73	1	1	0	18	0	30	Cosmetic Surgery	Change Appearance	Female
19	56	66	1	1	1	19	0	25	Cosmetic Surgery	Physical reason	Female
20	63	61	1	1	1	20	0	31	Cosmetic Surgery	Physical reason	Female
21	67	66	1	0	0	51	1	35	Waiting List	Change Appearance	Male
22	88.2	70	2	0	1	40	0	27	Waiting List	Physical reason	Female
23	70	91	2	0	1	41	1	13	Waiting List	Physical reason	Male
24	72	73	2	0	1	43	1	15	Waiting List	Physical reason	Male
25	75.1	75	2	0	1	31	0	17	Waiting List	Physical reason	Female

FIGURE 19.7 Data layout for multilevel modelling with no repeated measure

19.6.3. Picturing the data ②

Before we begin the analysis it's a good idea to have a look at the data. Our main example looks at **Surgery** and baseline quality of life (**Base_QoL**) as a predictors of quality of life after surgery (**Post_QoL**). Remember that the surgery was conducted at one of 10 clinics. Therefore, to begin with we could simply look at the relationship between baseline quality of life and post-surgery quality of life separately for the two surgery conditions (cosmetic surgery vs. waiting list). We might also want to graph this separately for the 10 clinics. We can use what we learnt about *ggplot2* in Chapter 4 to produce this plot; the resulting graph is shown in Figure 19.8.

SELF-TEST

✓ Using what you know about *ggplot2*, produce the graph described above. Display the levels of **Surgery_Text** in colours, and use **Clinic** to produce different graphs within a grid.

19.6.4. Ignoring the data structure: ANOVA ②

First of all, let's ground the example in something very familiar to us: ANOVA. Let's say for the time being that we were interested only in the effect that surgery has on post-operative quality of life. We could analyse this with a simple one-way independent ANOVA (or indeed a *t*-test), and the model is described by equation (19.1).

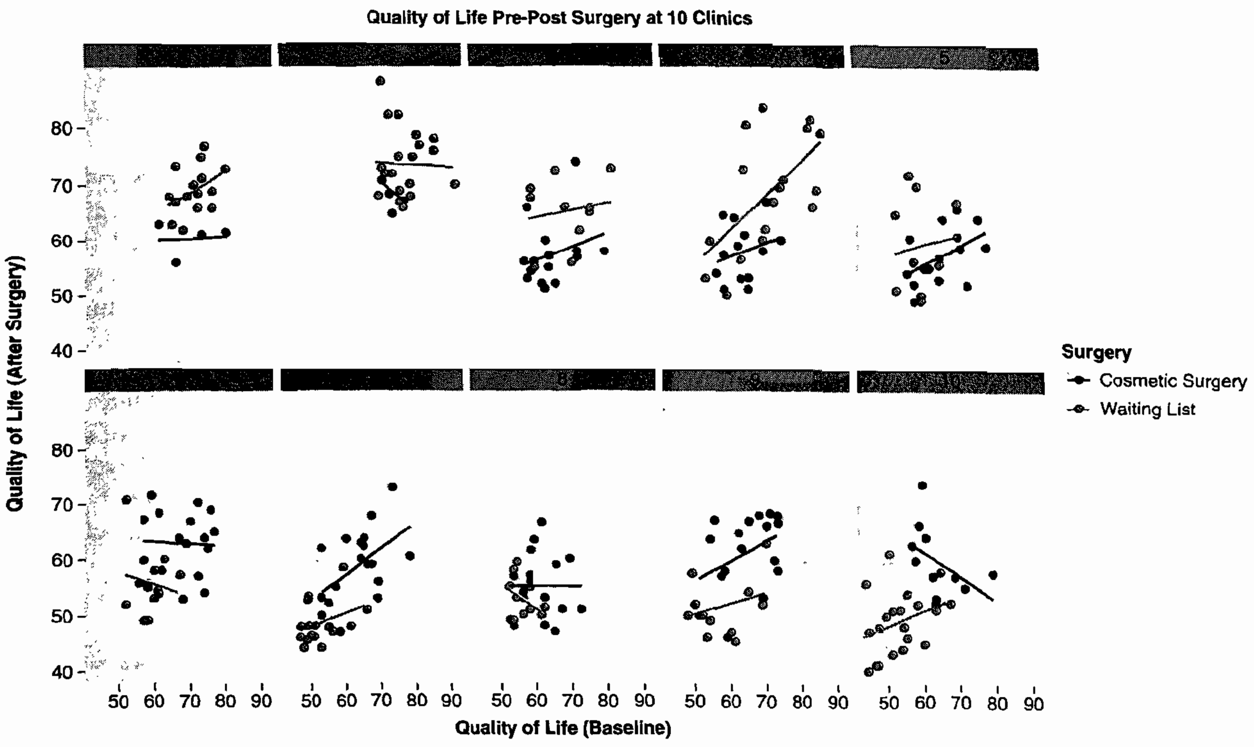


FIGURE 19.8
Graph of the relationship between baseline and post-surgery quality of life for people who had cosmetic surgery compared to those on the waiting list at 10 different clinics.

SELF-TEST

✓ Using what you know about ANOVA, conduct a one-way ANOVA using **Surgery** as the predictor and **Post_QoL** as the outcome.

In reality we wouldn't do an ANOVA, I'm just using it as a way of showing you that multilevel models are not big and scary, but are simply extensions of what we have done before. Output 19.1 shows the results of the ANOVA that you should get if you did the self-test. We find a non-significant effect of surgery on quality of life, $F(1, 274) = 0.33$, $p > .05$.

```
           Df  Sum Sq Mean Sq F value Pr(>F)
Surgery     1    28.6  28.620  0.3302  0.566
Residuals 274 23747.9  86.671
```

Output 19.1

We have also seen that we can think of ANOVA as a general linear model in which an outcome (in this case Post_QoL) is predicted from group membership (in this case

Surgery). Therefore, we could fit the same model but using the *lm()* function that we first encountered in Chapter 7.

```
surgeryLinearModel<-lm(Post_QoL ~ Surgery, data = surgeryData)
summary(surgeryLinearModel)
```

We have used the function *lm* (linear model) to create an object called *surgeryLinearModel*. The commands in brackets tell *lm()* what model we want to fit; as we have seen elsewhere in this book, the '~' means 'predicted from'. So, we have specified that we want **Post_QoL** predicted from **Surgery**. In other words, we have simply written out the linear model in equation (19.1) but without the *b*s. The rest of the options simply tell *lm()* to fit the model on the dataframe that we just created (*data = surgeryData*). Finally *summary(surgeryLinearModel)* prints the model parameters to the **R** console.

Output 19.2 shows the main table for the model. Compare this table with Output 19.1 and you'll see that there is basically no difference: we get a non-significant effect of surgery with an F of 0.33, and a p of .56. The point I want you to absorb here is that if we ignore the hierarchical structure of the data then what we are left with is something very familiar: an ANOVA/regression. The numbers are more or less exactly the same; all that has changed is that we have used different commands to get to the same end point.

```
Coefficients:
                         Estimate Std. Error t value Pr(>|t|)
(Intercept)               59.2710     0.8134  72.869   <2e-16 ***
Surgery[T.Waiting List]    0.6449     1.1222   0.575    0.566
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.31 on 274 degrees of freedom
Multiple R-squared: 0.001204,    Adjusted R-squared: -0.002442
F-statistic: 0.3302 on 1 and 274 DF,  p-value: 0.566
```

Output 19.2

19.6.5. Ignoring the data structure: ANCOVA ②

We have seen that there is no effect of cosmetic surgery on quality of life, but we did not take into account the quality of life before surgery. Let's, therefore, extend the example a little to look at the effect of the surgery on quality of life while taking into account the quality of life scores before surgery. Our model is now described by equation (19.8). You could do this analysis with an ANCOVA, using the *aov()* function or as a linear model using the *lm()* function. As in the previous section we'll run the analysis both ways, just to illustrate that we're doing the same thing when we run a hierarchical model.

SELF-TEST

✓ Using what you know about ANCOVA, conduct a one-way ANCOVA using **Surgery** as the predictor, **Post_QoL** as the outcome and **Base_QoL** as the covariate.

Output 19.3 shows the results of the ANCOVA that you should get if you did the self-test. The top output shows the Type I sums of squares, whereas the bottom is the same model but with Type III sums of squares (they differ slightly for baseline quality of life because we have an unbalanced design). With baseline quality of life included we find a significant effect of surgery on quality of life, $F(1, 273) = 4.04$, $p < .05$. Baseline quality of life also predicted quality of life after surgery, $F(1, 273) = 214.89$, $p < .001$.

```
             Df  Sum Sq Mean Sq  F value   Pr(>F)
Base_QoL      1 10291.4 10291.4 211.4321 < 2e-16 ***
Surgery       1   196.8   196.8   4.0435 0.04533 *
Residuals   273 13288.3    48.7
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Model:
Post_QoL ~ Base_QoL + Surgery
         Df Sum of Sq   RSS    AIC  F value    Pr(F)
<none>                13288 1075.3
Base_QoL  1   10459.6 23748 1233.5 214.8876 < 2e-16 ***
Surgery   1     196.8 13485 1077.3   4.0435 0.04533 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 19.3

We can also think of ANCOVA within the general linear model framework: the outcome (in this case **Post_QoL**) is predicted from group membership (in this case **Surgery**) and the covariate (**Base_QoL**). As with before the covariate was added, we can fit the model but using the *lm()* function by simply adding the baseline quality of life variable to the equation.

```
surgeryLinearModel<-lm(Post_QoL ~ Surgery + Base_QoL, data = surgeryData)
summary(surgeryLinearModel)
```

As before, the *lm()* function creates an object called *surgeryLinearModel*. We have specified that **Post_QoL** is the outcome variable and that it is predicted from (the ~ symbol) **Surgery** and baseline quality of life (**Base_QoL**). Again, notice how the *Post_QoL~Surgery + Base_QoL* is basically just equation (19.8) without the *b*s. The rest of the function specifies the data (*data = surgeryData*) and prints a summary of the model (*summary (surgeryLinearModel)*).

Output 19.4 shows the main table for the model. Compare this table with Output 19.3 and you'll see that again there is no difference: we get a significant effect of surgery with a t of –2.011, $p < .05$, and a significant effect of baseline quality of life with a t of 14.66, $p < .001$. We can also see that the regression coefficient for surgery is −1.70.

Hopefully this exercise, as well as being good revision, has convinced you that we're just doing a regression here, something you have been doing throughout this book. Multilevel models are not radically different, and if you think about it as just an extension of what you already know, then it's really relatively easy to understand. So, having shown you that we can do basic analyses through the linear models function, let's now use its power to factor in the hierarchical structure of the data.

```
Call:
lm(formula = Post_QoL ~ Surgery + Base_QoL, data = surgeryData)

Residuals:
     Min       1Q   Median       3Q      Max
-13.4142  -5.1326  -0.6495   4.0540  23.5005
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 18.14702    2.90767   6.241 1.65e-09 ***
Surgery     -1.69723    0.84404  -2.011   0.0453 *
Base_QoL     0.66504    0.04537  14.659  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.977 on 273 degrees of freedom
Multiple R-squared: 0.4411,     Adjusted R-squared: 0.437
F-statistic: 107.7 on 2 and 273 DF,  p-value: < 2.2e-16
```

Output 19.4

To sum up, we have seen that when we factor in the pre-surgery quality of life scores, which themselves significantly predict post-surgery quality of life scores, surgery seems to positively affect quality of life. However, at this stage we have ignored the fact that our data have a hierarchical structure. Essentially we have violated the independence assumption because scores from people who had their surgery at the same clinic are likely to be related to each other (and certainly more related than with people at different clinics). We have seen that violating the assumption of independence can have some quite drastic consequences (see section 10.3). However, rather than just panic and gibber about our *F*-ratio being inaccurate, we can model this covariation within clinics explicitly by including the hierarchical data structure in our analysis.

19.6.6. Assessing the need for a multilevel model ③

The first step in a multilevel analysis such as this is to assess the need to do it in the first place. If there is not significant variation across contexts in the first place then doing a multilevel model is simply a perverse exercise in mental flagellation. If there is little evidence of variation across contexts then save yourself a lot of pain and just do a regression/ANOVA/whatever variant of the general linear model that you feel like doing.

Ascertaining whether there is variation over your contexts is fairly straightforward. First, we need to fit a baseline model in which we include only the intercept; next, we fit a model that allows intercepts to vary over contexts; finally we compare these two models to see whether the fit has improved as a result of allowing intercepts to vary. If it has, we jump on the runaway train to multilevel insanity; if it has not, we do a little dance of joy into the loving arms of a simpler life.

In our surgery example then, we first need to get R to fit a baseline model that includes only the intercept. This is done using the *gls()* function (*generalized least squares*).[4]

```
interceptOnly <-gls(Post_QoL ~ 1, data = surgeryData, method = "ML")
summary(interceptOnly)
```

The format of the *gls()* function is very much like the *lm()* function that we have encountered before. In this example, we have asked R to create an object called *interceptOnly*, and we have specified that Post_QoL is the outcome variable and that it is predicted from (~)

[4] You might wonder why we don't simply use the *lm()* command. To compare models we need them to be computed in the same way. Multilevel models are estimated using maximum likelihood methods and generalized least squares use this method too, so we can compare these models. However, *lm()* used ordinary least squares methods and so can't be compared to a model estimated using maximum likelihood.

only the intercept (the '1' in the function translates as 'intercept'). The rest of the function specifies the data (*data = surgeryData*) and how to estimate the model (*method = "ML"*). The *method* option is very important. If you do not include it (i.e., use the default option), then **R** will use restricted maximum-likelihood methods (these can also be applied explicitly by writing *method = "REML"*). However, we have chosen to use maximum-likelihood estimation (*method = "ML"*). There are pros and cons to both (see R's Souls' Tip 19.1) but if you want to compare models as you build them up, you should use maximum-likelihood estimation. The final option (*summary(interceptOnly)*) is optional and prints the summary of the model shown in Output 19.5.

R's Souls' Tip 19.1 Estimation ③

R gives you the choice of two methods for estimating the parameters in the analysis: maximum likelihood (ML), which we have encountered before, and restricted maximum likelihood (REML). The conventional wisdom seems to be that ML produces more accurate estimates of fixed regression parameters, whereas REML produces more accurate estimates of random variances (Twisk, 2006). As such, the choice of estimation procedure depends on whether your hypotheses are focused on the fixed regression parameters or on estimating variances of the random effects. However, in many situations the choice of ML or REML will make only small differences to the parameter estimates. Also, if you want to compare models you must use ML.

```
Generalized least squares fit by maximum likelihood
  Model: Post_QoL ~ 1
  Data: surgeryData
       AIC      BIC    logLik
  2017.124 2024.365 -1006.562

Coefficients:
               Value Std.Error  t-value p-value
(Intercept) 59.60978 0.5596972 106.5036       0

Standardized residuals:
       Min         Q1        Med         Q3        Max
-2.1127754 -0.7875625 -0.1734394  0.7962286  3.0803354

Residual standard error: 9.281527
Degrees of freedom: 276 total; 275 residual
```

Output 19.5

Next, we need to fit the same model, but this time allowing the intercepts to vary across contexts, in this case we want them to vary across clinics. We do this by using the *lme* (linear mixed effect) function. In fact, this is the function that you'll use throughout the rest of the chapter. The format of this function is much the same as *lm()* and *gls()*; the only difference is that we need to specify the random part of the model using the option *random* = $x|y$, in which x is an equation specifying the random parts of the model and y is the contextual variable or variables across which we want to model variance. In the current example, we are trying to model intercepts that vary across clinics; therefore, we could add the instruction *random = ~1|Clinic*. Remember that we use '1' to denote the intercept, and that **Clinic** is the variable that contains information about the clinic that a given person attended. The resulting command is:

```
randomInterceptOnly <-lme(Post_QoL ~ 1, data = surgeryData, random =
~1|Clinic, method = "ML")
summary(randomInterceptOnly)
```

As before, executing this command creates a model (this time I've called it *randomInterceptOnly*), that predicts post-surgery quality of life from only the intercept (*Post_QoL~1*), but also allows intercepts to vary across clinics (*random = ~1|Clinic*). We have again asked for maximum-likelihood estimation (*method = "ML"*). You can use *summary(randomInterceptOnly)* to view the model summary (Output 19.6).

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
       AIC      BIC    logLik
  1911.473 1922.334 -952.7364

Random effects:
 Formula: ~1 | Clinic
        (Intercept) Residual
StdDev:    5.909691 7.238677

Fixed effects: Post_QoL ~ 1
               Value Std.Error  DF  t-value p-value
(Intercept) 60.08377  1.923283 266 31.24022       0

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.8828507 -0.7606631 -0.1378732  0.7075242  2.8607949

Number of Observations: 276
Number of Groups: 10
```

Output 19.6

To see whether allowing the intercepts to vary improves the model we can do several things. First, we can compare the fit of the model using indices such as AIC and BIC. If you compare Output 19.5 with Output 19.6 you'll see that the BIC when only the intercept was included is 2024.37 but decreases to 1922.33 when intercepts are allowed to vary. Remember that smaller values of BIC indicate a better fit of the data, so this gives us an indication that by allowing intercepts to vary the model fit has improved (BIC has decreased). This is all very well, but it does not give us an objective answer to whether the improvement in fit is 'significant' or big enough for us to continue down the multilevel path.

The second thing we can do is test the change in the $-2LL$ (equation (19.10)). We saw earlier that to be able to do this (1) full maximum-likelihood estimation must be used (and we have used this method); and (2) the new model contains all of the effects of the older model (this is true also, the models are identical except that we added a parameter reflecting the variability in intercepts across clinics). The log-liklihood is given in the outputs for each model and we could simply multiply these values by -2 to get the $-2LL$. We can also get **R** to do this for us for each model, and this has an advantage in that it will also tell us the degrees of freedom on which each log-liklihood is based. We can use the function *logLik()* for each model and then type '*−2' to multiply by −2. To obtain the $-2LL$ for our two models, we would therefore type:

```
logLik(interceptOnly)*-2
logLik(randomInterceptOnly)*-2
```

The resulting output should reveal that the model with random intercepts has a $-2LL$ of 2013.12 based on three degrees of freedom, whereas the model with only the intercept had a $-2LL$ of 19.05.47, based on two degrees of freedom. Therefore:

$$\chi^2_{\text{change}} = 2013.12 - 1905.47 = 107.65$$
$$df_{\text{change}} = 3 - 2 = 1$$

If we look at the critical values for the chi-square statistic with 1 degree of freedom in the Appendix, they are 3.84 ($p < .05$) and 6.63 ($p < .01$); therefore, this change is highly significant.

A simpler way to do much the same thing is to use the *anova()* function (section 7.8.4). For these types of models, this function compares the change in $-2LL$ without you having to compute it and produces an exact significance for this change. The same caveats as before apply: you should have used maximum likelihood and the models should be nested (that is, models higher up the chain need to contain all of the effects that were in models earlier in the chain). We can use this function as follows to compare our two models:

```
anova(interceptOnly, randomInterceptOnly)
```

The resulting Output 19.7 shows the fit indices for each model, but most important shows the value of change in the $-2LL$, the likelihood ratio, that we computed above (we can feel fairly smug that the value of 107.65 matches our earlier calculations). It also shows the degrees of freedom for each model (so that we can verify that the change in degrees of freedom is 1 as we previously calculated). Finally, it shows a p-value, which is highly significant and verifies our earlier conclusion that it is important that we model the variability in intercepts because when we do the fit of our model is significantly improved. The change in the $-2LL$ has a chi-square distribution, so we can report this statistic in the normal way, $\chi^2(1) = 107.65$, $p < .0001$. We can conclude then that the intercepts vary significantly across the different clinics. Multilevel madness must ensue.

```
Model df  AIC     BIC     logLik   Test  L.Ratio p-value
   1  2 2017.12 2024.36 -1006.56
   2  3 1911.47 1922.33  -952.73 1 vs 2 107.6517  <.0001
```

Output 19.7

19.6.7. Adding in fixed effects ③

We have seen that intercepts vary significantly across clinics. The model that we currently have is this:

$$\text{QoL After Surgery}_{ij} = b_{0j} + \varepsilon_{ij}$$
$$b_{0j} = b_0 + u_{0j}$$

However, we originally had hypotheses about how surgery and baseline quality of life will affect quality of life after surgery. Now that we have a baseline model with random intercepts, we can start to build up the final model by adding these predictors. Let's first add in the **Surgery** variable, which defines whether a person had surgery or was on the waiting list. Our model now becomes:

$$\text{QoL After Surgery}_{ij} = b_{0j} + b_1\text{Surgery}_{ij} + \varepsilon_{ij}$$
$$b_{0j} = b_0 + u_{0j}$$

To add this predictor we again create a new object in **R** (which I have called *randomInterceptSurgery*) using the *lme()* function. This function is exactly the same as before except that we have replaced *Post_QoL~1* with *Post_QoL~Surgery*. As such, the model now predicts quality of life after surgery from the variable **Surgery** and the intercept.[5]

```
randomInterceptSurgery <-lme(Post_QoL ~ Surgery, data = surgeryData, random
= ~1|Clinic, method = "ML")
summary(randomInterceptSurgery)
```

The resulting Output 19.8 shows this model. Note that the BIC has actually increased from 1922.33 in the previous model to 1924.62 in this one, which suggests that adding **Surgery** has not improved the fit of the model (consistent with this interpretation the log-likelihood has increased also). This interpretation is also borne out by the fixed effect of **Surgery**, which is not significant, $b = 1.66$, $t(265) = -1.83$, $p = .068$.

Although **Surgery** does not appear to be a significant predictor, we had a final model that also included baseline quality of life, so we should add that fixed effect too. Our model is now described by:

$$\text{QoL After Surgery}_{ij} = b_{0j} + b_1\text{Surgery}_{ij} + b_2\text{QoL Before Surgery}_{ij} + \varepsilon_{ij}$$
$$b_{0j} = b_0 + u_{0j}$$

To add this predictor we again use *lme()* to create a new object (which I have called *randomInterceptSurgeryQoL*). This function is exactly the same as before except that we have replaced *Post_QoL~Surgery* with *Post_QoL~Surgery + Base_QoL*. Hopefully, you can see that each time we specify a model we simply write out the equation that describes the model, but without the *b*s. As such, the model now predicts quality of life after surgery from the variables **Surgery**, **Base_QoL** and the intercept, with intercepts varying across clinics.

```
randomInterceptSurgeryQoL <-lme(Post_QoL ~ Surgery + Base_QoL, data = sur-
geryData, random = ~1|Clinic, method = "ML")
summary(randomInterceptSurgeryQoL)
```

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
       AIC      BIC    logLik
  1910.137 1924.619 -951.0686

Random effects:
 Formula: ~1 | Clinic
```

[5] You might wonder where the '1' representing the intercept has gone. The intercept is implied within the model so we don't need to specify it, although we could be explicit and write *Post_QoL ~ 1 + Surgery*. The end result would be the same. Similarly, we could write *Post_QoL ~ 0 + Surgery* if we wanted to remove the intercept from the model.

```
         (Intercept) Residual
StdDev:     6.099513  7.18542

Fixed effects: Post_QoL ~ Surgery
                Value Std.Error  DF  t-value p-value
(Intercept) 59.30517 2.0299632 265 29.21490   0.000
Surgery      1.66583 0.9091314 265  1.83233   0.068
 Correlation:
        (Intr)
Surgery -0.21

Standardized Within-Group Residuals:
       Min          Q1         Med          Q3         Max
-1.8904290 -0.7191399 -0.1420998  0.7177762  2.8644538

Number of Observations: 276
Number of Groups: 10
```

Output 19.8

R's Souls' Tip 19.2 Missing data ③

We saw in R's Souls' Tip 7.1 that missing data create problems for linear models (i.e., regression). Unlike ordinary regresison, multilevel models do not require balanced data sets, so you can have missing data, but you still need to tell **R** what to do with missing cases: just like ordinary regression, if you try to do a multilevel model with missing values in the dataframe you will get an error because *lme()* does not know what to do with these values. As with ordinary regression, you should add *na.action = na.exclude* to the *lme()* function to let it know that it can ignore any NAs it finds in the dataframe. In the surgery data we don't have missing values so we haven't had to use this instruction, but we could easily insert it. For example, the command for our model with Surgery and Base_QoL would become:

```
randomInterceptSurgeryQoL <-lme(Post_QoL ~ Surgery + Base_QoL, data = surgeryData,
random = ~1|Clinic, method = "ML", na.action = na.exclude)
```

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
      AIC      BIC   logLik
  1847.49 1865.592 -918.745

Random effects:
 Formula: ~1 | Clinic
        (Intercept) Residual
StdDev:    3.039264 6.518986

Fixed effects: Post_QoL ~ Surgery + Base_QoL
                Value Std.Error  DF   t-value p-value
(Intercept) 29.563601  3.471879 264  8.515160  0.0000
Surgery     -0.312999  0.843145 264 -0.371228  0.7108
Base_QoL     0.478630  0.052774 264  9.069465  0.0000
```

```
 Correlation:
         (Intr) Surgry
Surgery   0.102
Base_QoL -0.947 -0.222

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.8872666 -0.7537675 -0.0954987  0.5657241  3.0020852

Number of Observations: 276
Number of Groups: 10
```

Output 19.9

Output 19.9 shows the summary of the final model. Note that the BIC and AIC have both decreased since the previous model (Output 19.8); for example, the BIC has reduced from 1924.62 to 1865.59. This implies a better fitting model. We can have a look at how the fit of the models has improved using the *anova()* function that we used before; the following will compare all three models that we have so far fitted):

```
anova(randomInterceptOnly, randomInterceptSurgery,
randomInterceptSurgeryQoL)
```

Output 19.10 shows the resulting analysis. We start with just varying intercepts (model 1), and we can see that by including surgery as a fixed effect (model 2) we got no significant improvement ($p > .05$). In fact the change in the $-2LL$ is only 3.34, which has a significance of $p = .068$ (note that this is the same significance as for the t that tests the regression parameter of the fixed effect of surgery in Output 19.8). In model 3, we added the effect of baseline quality of life, and this had a dramatic impact. The $-2LL$ changed by 64.65 and this change was highly significant, $\chi^2(1) = 64.65$, $p < .0001$. As we have already noted the AIC and BIC also decrease from model 2 to model 3 showing that the fit is improving.

```
Model df      AIC      BIC    logLik   Test  L.Ratio p-value
1   3 1911.473 1922.334 -952.7364
2   4 1910.137 1924.619 -951.0686 1 vs 2  3.33564  0.0678
3   5 1847.490 1865.592 -918.7450 2 vs 3 64.64721  <.0001
```

Output 19.10

Given that including baseline quality of life has improved the fit of our model so much, let's briefly take stock of the model (shown in Output 19.9). The regression parameter for the effect of **Surgery** is –0.31, which is not significant, $t(264) = -0.37$, $p > .05$. However, baseline quality of life has a regression parameter of 0.48, which is highly significant $t(264) = 9.07$, $p < .001$. The standard deviation of the intercepts is 3.04 (we can square this value to get the variance of intercepts across clinics, which in this case is 9.24).

19.6.8. Introducing random slopes ④

We have seen that including a random intercept is important for this model (it changes the log-likelihood significantly). Figure 19.9 suggests that different clinics have different slopes; therefore, we could now look at whether adding a random slope will benefit the model. The model is now described by equation (19.9), which we saw earlier on; it can be specified in R with only minor modifications to the *lme()* function. All we are doing is adding another random term to the model, so, whereas before we specified random part of the model as *random = ~1 | Clinic*, we now need to change this to *random = ~Surgery | Clinic*. This change tells **R** that the model now allows the effect of **Surgery** (i.e., the slope) to vary across clinics. As when we specified the main model, the intercept is implied in this, so the change will give us both random intercepts over clinics, but also random slopes for the variable **Surgery**.[6]

```
addRandomSlope<-lme(Post_QoL ~ Surgery + Base_QoL, data = surgeryData, random
= ~Surgery|Clinic, method = "ML")
summary(addRandomSlope)
anova(randomInterceptSurgeryQoL,addRandomSlope)
```

The code above creates a new object called *addRandomSlope*, which is the same model as before but with a random slope added for the effect of **Surgery**. We then ask for a summary of this new model as we have done before, we then compare this new model to the previous one, using the *anova()* function, which we have used before. Output 19.11 shows the results of this model comparison. By allowing the effect of **Surgery** to vary across clinics we have reduced the BIC from 1865.59 to 1837.97, and the $-2LL$ has changed significantly,[7] $\chi^2(2) = 38.87$, $p < .0001$. In short, adding random slopes to the model has significantly improved its fit, which means there is significant variability in the effect of surgery across clinics.

Across this model and the previous one, we can conclude from the $-2LL$ as we built up the models that the intercepts, $\chi^2(1) = 64.65$, $p < .0001$, and slopes, $\chi^2(2) = 38.87$, $p < .0001$, for the relationship between surgery and quality of life (when controlling for baseline quality of life) vary significantly across the different clinics.

```
Model df      AIC      BIC    logLik   Test  L.Ratio p-value
1   5 1847.490 1865.592 -918.7450
2   7 1812.624 1837.966 -899.3119 1 vs 2 38.86626  <.0001
```

Output 19.11

Output 19.12 shows the summary of the model that contains both random slopes and intercepts. The regression parameter for the effect of **Surgery** is now $b = -0.65$, which is still not significant, $t(264) = -0.31$, $p > .05$. Baseline quality of life is still highly significant, $b = 0.31$, $t(264) = 5.80$, $p < .001$. The standard deviation of the intercepts is 6.13, and the effect of **Surgery** has a standard deviation of 6.20 (which we can square to find the variance, which is 38.41). The slopes and intercepts are highly correlated also ($r = -.97$).

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
       AIC      BIC    logLik
  1812.624 1837.967 -899.3119

Random effects:
 Formula: ~Surgery | Clinic
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev   Corr
(Intercept) 6.132655 (Intr)
Surgery     6.197489 -0.965
Residual    5.912335
```

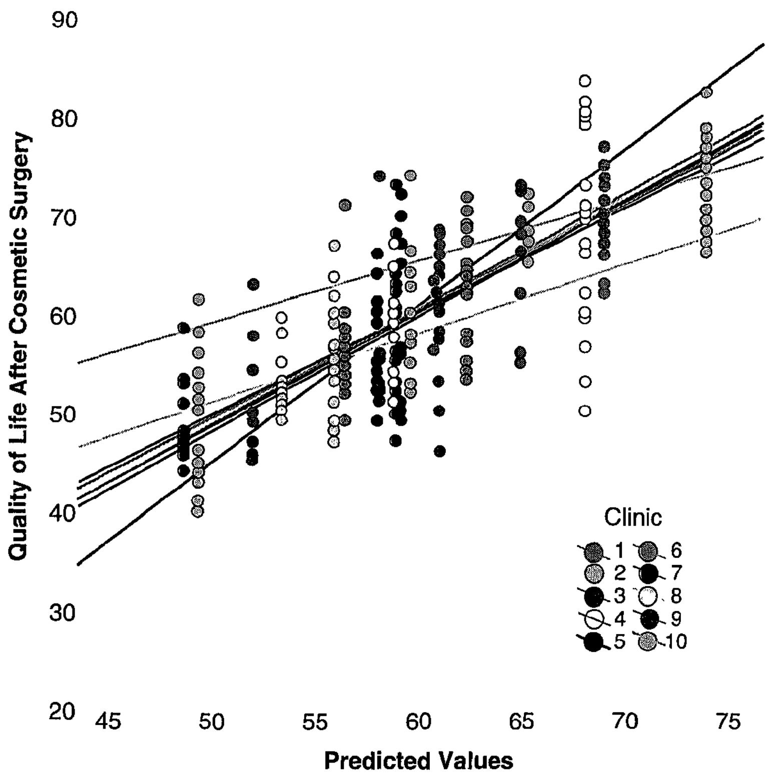
[6] It would be pretty unusual to want random slopes but not intercepts, because variability in the effect of a variable across contexts will tend to create variability in the intercepts across contexts too. However, should you want to have a model with random slopes but not intercepts you simply change the random effect to *~0 + Surgery | Clinic*; the 0 gets rid of the intercept.

[7] The observant among you will notice that even though we have added only one new term to the model (the random slope of surgery across clinics), the degrees of freedom have increased by 2 rather than 1. That's odd isn't it? Actually, it's not and it happens because by including random slopes we actually add *two* parameters to the model: the estimate of the variance of the effect of surgery across clinics, and also an estimate of the covariance between slopes and intercepts (i.e., the extent to which intercepts and slopes are dependent on each other).

```
Fixed effects: Post_QoL ~ Surgery + Base_QoL
                Value Std.Error  DF   t-value p-value
(Intercept) 40.10253  3.892945 264 10.301334  0.0000
Surgery     -0.65453  2.110917 264 -0.310069  0.7568
Base_QoL     0.31022  0.053506 264  5.797812  0.0000
 Correlation:
         (Intr) Surgery
Surgery  -0.430
Base_QoL -0.855 -0.063

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-2.4114778 -0.6628574 -0.1138411  0.6833110  2.8334730

Number of Observations: 276
Number of Groups: 10
```

Output 19.12

FIGURE 19.9 Predicted values from the model (surgery predicting quality of life after controlling for baseline quality of life) plotted against the observed values



19.6.9. Adding an interaction term to the model ④

We can now build up the model by adding in another variable. One of the variables we measured was the reason for the person having cosmetic surgery: was it to resolve a physical problem or was it purely for vanity? We can add this variable to the model, and also look at whether it interacts with surgery in predicting quality of life. Our model will simply expand to incorporate these new terms, and each term will have a regression coefficient (which we select to be fixed). Therefore, our final model can be described as in the equation below (note that all that has changed is that there are two new predictors):

$$
\begin{aligned}
\text{QoL After}_{ij} &= b_{0j} + b_{1j}\text{Surgery}_{ij} + b_2\text{QoL Before Surgery}_{ij} + b_3\text{Reason}_{ij} \\
&\quad + b_4(\text{Reason} \times \text{Surgery})_{ij} + \varepsilon_{ij} \\
b_{0j} &= b_0 + u_{0j} \\
b_{1j} &= b_1 + u_{1j}
\end{aligned}
\qquad (19.1)
$$

To set up this model in **R** is very easy; it just requires some minor changes to the code. First, we'll add in the effect of Reason. To do this we create a new model, which I have called *addReason*, and we set it up in exactly the same way as before, except that we add **Reason** to the model. So, our model changes from *Post_QoL~Surgery + Base_QoL* to *Post_QoL~Surgery + Base_QoL + Reason*. It's as simple as that. In fact, it can be even simpler if you use the update function (see R's Souls' Tip 19.3).

```
addReason<-lme(Post_QoL ~ Surgery + Base_QoL + Reason, data = surgeryData,
random = ~Surgery|Clinic, method = "ML")
```

As we've seen before, to add an interaction term we use a colon (i.e., *Surgery:Reason*). I have called this model *finalModel* and we specify it exactly the same as the previous model except that we add in the interaction term. Again, we could simplify this process by using the update function (see R's Souls' Tip 19.3).

```
finalModel<-lme(Post_QoL ~ Surgery + Base_QoL + Reason + Surgery:Reason, data =
surgeryData, random = ~Surgery|Clinic, method = "ML")
```

R's Souls' Tip 19.3 The *update* function ③

Throughout the first example in this chapter I have built the models up piece by piece because I think it's useful to see how the code relates to the equation that describes the model. However, as we have seen before (see R's Souls' Tip 7.2) the *update()* function is a quicker way to add new things to old models. Let's start with the random slopes example from the main text. Our model is as follows:

```
addRandomSlope<-lme(Post_QoL ~ Surgery + Base_QoL, data = surgeryData, random =
~Surgery|Clinic, method = "ML")
```

In the text, we added the variable **Reason** to this model, using the longhand method:

```
addReason<-lme(Post_QoL ~ Surgery + Base_QoL + Reason, data = surgeryData, random =
~Surgery|Clinic, method = "ML")
```

Using the *update()* function we can do the same thing in much less text:

```
addReason<-update(addRandomSlope, .~. + Reason)
```

This function, like the longhand one, creates a new model called *addReason*, and it does this by updating an existing model. The first part of the parenthesis tells **R** that we want to update the model called *addRandomSlope*; the *.~. + Reason* tells **R** to keep the outcome variable and all of the previous predictors and to add **Reason** as a predictor.

Similarly, having created the model *addReason*, we can update this model to include the **Surgery** × **Reason** interaction (as we did in the main text). We can again use the update function as follows:

```
finalModel<-update(addReason, .~. + Reason:Surgery)
```

This command creates a new model called *finalModel*. Note that we have specified *addReason* in the *update()* function because we want to update the model that includes **Reason** as a predictor, and have added the interaction term typing + *Reason:Surgery*.

We need to see whether adding these new terms has improved the fit of the model and again we can simply use the *anova()* function. We have two new models (*addReason* and *finalModel*) that we want to compare to our previous model (*addRandomSlope*). We can do this comparison in a single function, remembering to order our models in the same order that they were built (so each time we have added only a single parameter):

```
anova(addRandomSlope, addReason, finalModel)
```

Output 19.13 presents the resulting output, which shows that adding **Reason** to the model reduces the $-2LL$ by 3.80, which is not quite a significant change, $p = .0513$; however, adding the **Surgery** × **Reason** interaction reduces the $-2LL$ by 5.78, which is a significant change, $p < .05$.

```
               Model df    AIC     BIC   logLik  Test  L.Ratio p-value
addRandomSlope 1  7 1812.62 1837.96 -899.31
addReason      2  8 1810.82 1839.78 -897.41 1 vs 2 3.7989  0.0513
finalModel     3  9 1807.04 1839.62 -894.52 2 vs 3 5.7795  0.0162
```

Output 19.13

Output 19.14 shows the summary of the final model. Quality of life before surgery significantly predicted quality of life after surgery, $t(262) = 5.75$, $p < .001$, surgery still did not significantly predict quality of life, $t(262) = -1.46$, $p = .15$, but the reason for surgery, $t(262) = -3.08$, $p < .01$, and the interaction of the reason for surgery and surgery, $t(262) = 2.48$, $p < .05$, both did significantly predict quality of life. The table of estimates also gives us the regression coefficients. However, if we want to get confidence intervals for these parameters we need to use the function *intervals()*, within which we simply specify the model for which we would like confidence intervals, and the level of the confidence intervals as a proportion (i.e., 0.99 produces 99% confidence intervals). For example:

```
intervals(finalModel, 0.90)
intervals(finalModel, 0.95)
intervals(finalModel, 0.99)
```

produce 90%, 95% and 99% intervals. If you insert only the model name into the function you will get 95% CIs by default.

Output 19.15 shows the 95% confidence intervals for our final model. We can see, for example, that **Surgery** had a $b = -3.19$, with a 95% confidence interval of -7.45 (lower) and 1.08 (upper). This interval crosses zero and so is not significant at $p < .05$, which of course we knew already from the main summary. These confidence intervals are very useful for establishing whether the variance of the intercepts and slopes is significant. For example, we can see that the standard deviation for intercepts was 5.48 with a 95% confidence interval from 3.31 to 9.07, for the slope of **Surgery** we get 5.42 (3.13, 9.37); because both confidence intervals do not cross zero we can see that the variability in both slopes and intercepts was significant, $p < .05$.

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
       AIC      BIC    logLik
  1807.045 1839.629 -894.5226

Random effects:
 Formula: ~Surgery | Clinic
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev   Corr
(Intercept) 5.482366 (Intr)
Surgery     5.417501 -0.946
Residual    5.818910
```

```
Fixed effects: Post_QoL ~ Surgery + Base_QoL + Reason + Reason:Surgery
                    Value Std.Error  DF   t-value p-value
(Intercept)      42.51782  3.875318 262 10.971440  0.0000
Surgery          -3.18768  2.185369 262 -1.458645  0.1459
Base_QoL          0.30536  0.053125 262  5.747833  0.0000
Reason           -3.51515  1.140934 262 -3.080938  0.0023
Surgery:Reason    4.22129  1.700269 262  2.482717  0.0137

Correlation:
               (Intr) Surgry Bas_QL Reason
Surgery        -0.356
Base_QoL       -0.865 -0.078
Reason         -0.233  0.306  0.065
Surgery:Reason  0.096 -0.505  0.024 -0.661

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-2.2331485 -0.6972193 -0.1541074  0.6326387  3.1641797

Number of Observations: 276
Number of Groups: 10
```

Output 19.14

```
Approximate 95% confidence intervals

 Fixed effects:
                    lower       est.      upper
(Intercept)    34.9565211 42.5178190 50.0791170
Surgery        -7.4516428 -3.1876767  1.0762895
Base_QoL        0.2017008  0.3053561  0.4090114
Reason         -5.7412731 -3.5151479 -1.2890227
Surgery:Reason  0.9038206  4.2212885  7.5387563
attr(,"label")
[1] "Fixed effects:"

 Random Effects:
  Level: Clinic
                               lower       est.      upper
sd((Intercept))            3.3138275  5.4823658  9.0699757
sd(Surgery)                3.1331192  5.4175011  9.3674439
cor((Intercept),Surgery) -0.9937813 -0.9455545 -0.5986153

 Within-group standard error:
   lower     est.    upper
5.331222 5.818910 6.351211
```

Output 19.15

As this is our final model, let's now give some thought to interpretation. The effect of the reason for surgery is easy to interpret. Given that we coded this predictor as 1 = physical reason and 0 = change appearance, the negative coefficient tells us that as reason increases (i.e., as a person goes from having surgery to change their appearance to having it for a physical reason) quality of life decreases. However, this effect in isolation isn't that interesting because it includes both people who had surgery and the waiting list controls. More interesting is the interaction term, because this takes account of whether or not the person had surgery. To break down this interaction we could rerun the analysis separately for the two 'reason groups'. Obviously we would remove the interaction term and the main effect of **Reason** from this analysis (because we are analysing the physical reason group separately from the group who wanted to change their appearance). As such, you need to fit the model in the previous section, but separately for those who had cosmetic surgery and those who had surgery for a physical reason.

Fortunately, this is fairly easy to do because *lme()* has a *subset* option that enables you to select a variable that specifies which rows of the data file that you want to use. First we need to create a variable that returns 'True' if the person had surgery for physical reasons and 'False' for those who had cosmetic surgery:

```
physicalSubset<- surgeryData$Reason==1
```

This function simply creates a variable called *physicalSubset* that will be set to TRUE only if the variable **Reason** is equal to 1. In our data set people who have a '1' for **Reason** are those who had surgery for a physical reason, so we are asking **R** to set *physicalSubset* to TRUE if the person had surgery for a physical reason. (Remember that *surgeryData$Reason* means 'the variable called *Reason* in the data set called *surgeryData*', and that in **R** '==' means 'equal to'.) We also create another variable called *cosmeticSubset* that returns TRUE for any people who had cosmetic surgery. We create this variable in much the same way (but now we specify that the variable **Reason** must be equal to 0, because in our data set 0 represents people who had cosmetic surgery):

```
cosmeticSubset<-surgeryData$Reason==0
```

Next, we create two new models that contain **Base_QoL** and **Surgery** as predictors and have random slopes and intercepts. The first one is for people who had surgery for a physical reason, so we set the subset option to be *physicalSubset*, which specifies all of the people who had surgery for this reason:

```
physicalModel<-lme(Post_QoL ~ Surgery + Base_QoL, data = surgeryData, random =
~Surgery|Clinic, subset= physicalSubset, method = "ML")
```

This creates a model (called *physicalModel*). Note that I have used *subset = physicalSubset*, which uses the variable **physicalSubset** to determine whether or not to include a particular case of data. By doing this our resulting model will include only those who had surgery for a physical reason. Similarly, we can use the variable **cosmeticSubset** to create another model that is fit using only those who had cosmetic surgery:

```
cosmeticModel<-lme(Post_QoL ~ Surgery + Base_QoL, data = surgeryData, random =
~Surgery|Clinic, subset= cosmeticSubset, method = "ML")
```

We can get a summary of these models in the usual way:

```
summary(physicalModel)
summary(cosmeticModel)
```

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
  Subset: physicalSubset
       AIC      BIC    logLik
  1172.560 1194.832 -579.2798

Random effects:
 Formula: ~Surgery | Clinic
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev   Corr
(Intercept) 5.773827 (Intr)
Surgery     5.804865 -0.948
Residual    5.798764
```

```
Fixed effects: Post_QoL ~ Surgery + Base_QoL
                Value Std.Error  DF  t-value p-value
(Intercept) 38.02079  4.705980 166 8.079250  0.0000
Surgery      1.19655  2.099769 166 0.569848  0.5696
Base_QoL     0.31771  0.069471 166 4.573271  0.0000
 Correlation:
          (Intr) Surgry
Surgery  -0.306
Base_QoL -0.908 -0.078

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.2447342 -0.6505340 -0.1264188  0.6111506  2.9472101

Number of Observations: 178
Number of Groups: 10
```

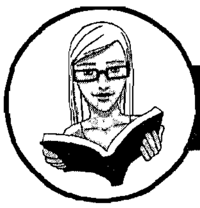
Output 19.16

```
Linear mixed-effects model fit by maximum likelihood
 Data: surgeryData
  Subset: cosmeticSubset
       AIC      BIC    logLik
  650.9469 669.0417 -318.4734

Random effects:
 Formula: ~Surgery | Clinic
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev   Corr
(Intercept) 5.006026 (Intr)
Surgery     5.292027 -0.969
Residual    5.738551

Fixed effects: Post_QoL ~ Surgery + Base_QoL
                Value Std.Error DF   t-value p-value
(Intercept) 41.78605  5.573849 87  7.496802  0.0000
Surgery     -4.30702  2.275002 87 -1.893193  0.0617
Base_QoL     0.33849  0.080274 87  4.216720  0.0001
 Correlation:
          (Intr) Surgry
Surgery  -0.252
Base_QoL -0.937 -0.058
Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.8945645 -0.6616222 -0.1461451  0.6460834  2.6741347
Number of Observations: 98
Number of Groups: 9
```

Output 19.17

The model for those who had surgery for a physical reason is shown in Output 19.16, whereas the model for those who had cosmetic surgery is in Output 19.17. It shows that for those operated on only to change their appearance, surgery almost significantly predicted quality of life after surgery, $b = -4.31$, $t(87) = -1.89$, $p = .06$. The negative gradient shows that for these people quality of life after surgery was lower compared to the control group. However, for those who had surgery to solve a physical problem surgery did not significantly predict quality of life, $b = 1.20$, $t(166) = 0.57$, $p = .57$. However, the slope was positive, indicating that people who had surgery scored higher on quality of life than those on the waiting list (although not significantly so!). The interaction effect, therefore, reflects the difference in slopes for surgery as a predictor of quality of life in those who had surgery for physical problems (slight positive slope) and those who had surgery purely for vanity (a negative slope).

We could sum up these results by saying that quality of life after surgery, after controlling for quality of life before surgery, was lower for those who had surgery to change their appearance than those who had surgery for a physical reason. This makes sense because for those having surgery to correct a physical problem, the surgery has probably brought relief and so their quality of life will improve. However, for those having surgery for vanity they might well discover that having a different appearance wasn't actually at the root of their unhappiness, so their quality of life is lower.



CRAMMING SAM'S TIPS **Multilevel models R output**

- The $-2LL$ and its significance can be used to compare models that are the same in all but one parameter. The AIC and BIC can also be compared across models (but not significance tested).
- The *fixed effects* tell you whether your predictors significantly predict the outcome. If the significance value is less than .05 then the effect is significant.
- Interpret the nature of the effect using the regression coefficient and its confidence interval. The direction of these coefficients tells us whether the relationship between each predictor and the outcome is positive or negative. To get the confidence intervals you need to use the *intervals()* function.
- The standard deviation of random effects can tell us how much intercepts and slopes varied over our level 1 variable. The significance of these estimates can be ascertained from their confidence intervals, obtained using the *intervals()* function.

19.7. Growth models ④

Growth models are extremely important in many areas of science, including psychology, medicine, physics, chemistry and economics. In a growth model the aim is to look at the rate of change of a variable over time: for example, we could look at white blood cell counts, attitudes, radioactive decay or profits. In all cases we're trying to see which model best describes the change over time.

19.7.1. Growth curves (polynomials) ④

Figure 19.10 gives some examples of possible **growth curves**. This diagram shows three **polynomials** representing a linear trend (the dark blue line) otherwise known as a first-order polynomial, a quadratic trend (the light blue line) otherwise known as a second-order polynomial, and a cubic trend (the black line) otherwise known as a third-order polynomial. Notice first that the linear trend is a straight line, but as the polynomials increase they get more and more curved, indicating more rapid growth over time. Also, as polynomials increase, the change in the curve is quite dramatic (so dramatic that I adjusted the scale of

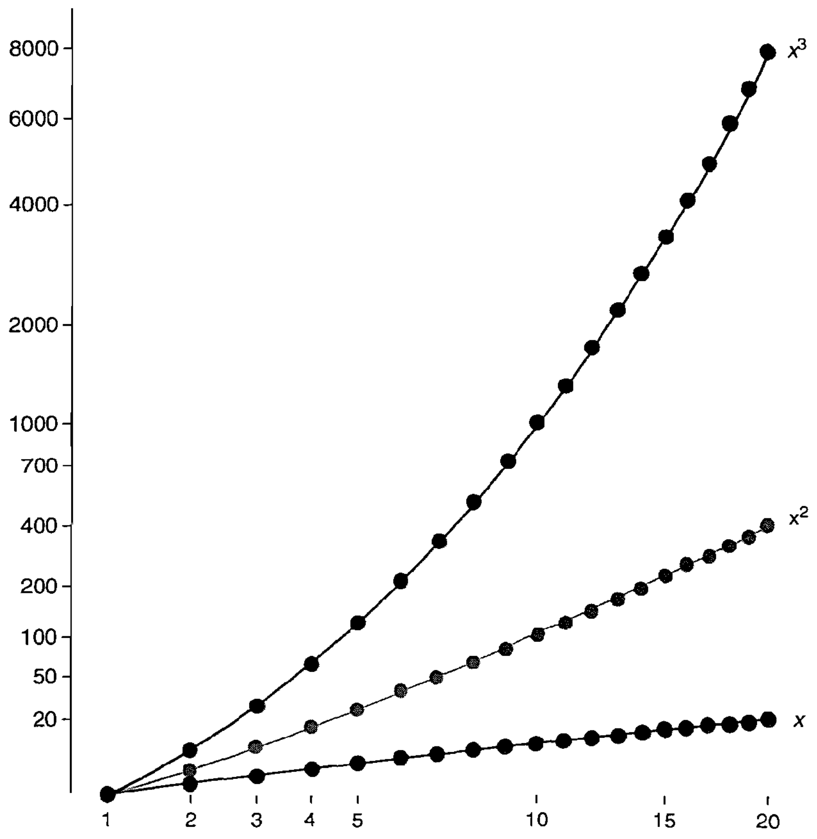
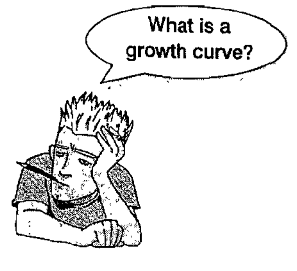


FIGURE 19.10 Illustration of a first-order (linear, blue), second-order (quadratic, light blue) and third-order (cubic, black) polynomial

the graph to fit all three curves on the same diagram). This observation highlights the fact that any growth curve higher than a quadratic (or possibly cubic) trend is very unrealistic in real data. By fitting a growth model to the data we can see which trend best describes the growth of an outcome variable over time (though no one will believe that a significant fifth-order polynomial is telling us anything meaningful about the real world!).

The growth curves that we have described might seem familiar to you: they are the same as the trends that we described for ordered means in section 10.4.5. What we are discussing now is really no different. There are just two important things to remember when fitting growth curves: (1) you can fit polynomials up to one less than the number of time points that you have; and (2) a polynomial is defined by a simple power function. On the first point, this means that with three time points you can fit a linear and quadratic growth curve (or a first- and second-order polynomial), but you cannot fit any higher-order growth curves. Similarly, if you have six time points you can fit up to a fifth-order polynomial. This is the same basic idea as having one less contrast than the number of groups in ANOVA (see section 10.4).



On the second point, we have to define growth curves manually in multilevel models in **R**: there is not a convenient option that we can select to do it for us. However, this is quite easy to do. If *time* is our predictor variable, then a linear trend is tested by including this variable alone. A quadratic or second-order polynomial is tested by including a predictor that is *time*^{2}, a cubic or third-order polynomial is tested by including a predictor that is *time*^{3} and so on. So any polynomial is tested by including a variable that is the predictor to the power of the order of polynomial that you want to test: for a fifth-order polynomial we need a predictor of *time*^{5} and for an *n*-order polynomial we would have to include *time*^{n} as a predictor. Hopefully you get the general idea.

19.7.2. An example: the honeymoon period ②

I once saw a brilliant talk given by Professor Daniel Kahneman, who won the 2002 Nobel Prize for Economics. In this talk Kahneman brought together an enormous amount of research on life satisfaction (he explored questions such as whether people are happier if they are richer). There was one graph in this talk that particularly grabbed my attention. It showed that leading up to marriage people reported greater life satisfaction, but by about two years after marriage this life satisfaction decreased back to its baseline level. This graph perfectly illustrated what people talk about as the 'honeymoon period': a new relationship/marriage is great at first (no matter how ill suited you may be) but after six months or so the cracks start to appear and everything turns to elephant dung. Kahneman argued that people adapt to marriage; it does not make them happier in the long run (Kahneman & Krueger, 2006).[8] This got me thinking about relationships not involving marriage (is it marriage that makes you happy, or just being in a long-term relationship?). Therefore, in a completely fictitious parallel world where I don't research child anxiety, but instead concern myself with people's life satisfaction, I collected some data. I organized a massive speed-dating event (see Chapter 14). At the start of the night I measured everyone's life satisfaction (**Satisfaction_Baseline**) on a 10-point scale (0 = completely dissatisfied, 10 = completely satisfied) and their gender (**Gender**). After the speed dating I noted all of the people who had found dates. If they ended up in a relationship with the person that they met on the speed-dating night then I stalked these people over the next 18 months of that relationship. As such, I had measures of their life satisfaction at 6 months (**Satisfaction_6_Months**), 12 months (**Satisfaction_12_Months**) and 18 months (**Satisfaction_18_Months**), after they entered the relationship. None of the people measured were in the same relationship (i.e., I measured only life satisfaction from one of the people in the couple).[9] Also, as is often the case with longitudinal data, I didn't have scores for all people at all time points because not everyone was available at the follow-up sessions. One of the benefits of a multilevel approach is that these missing data do not pose a particular problem. The data are in the file **Honeymoon Period.dat**.

Load these data into **R** by executing the following command:

```
satisfactionData = read.delim("Honeymoon Period.dat",  header = TRUE)
```

Figure 19.11 shows the data. Each dot is a data point and the line shows the average life satisfaction over time. Basically, from baseline, life satisfaction rises slightly at time 2 (6 months) but then starts to decrease over the next 12 months. There are two things to note about the data. First, time 0 is before the people enter into their new relationship, yet already there is a lot of variability in their responses (reflecting the fact that people will vary in their satisfaction due to other reasons such as finances, personality and so on). This suggests that intercepts for life satisfaction differ across people. Second, there is also a lot of variability in life satisfaction after the relationship has started (time 1) and at all subsequent time points, which suggests that the slope of the relationship between time and life satisfaction might vary across people also. If we think of the time points as a level 1 variable that is nested with people (a level 2 variable) then we can easily model this variability in intercepts and slopes within people. We have a situation similar to Figure 19.4 (except

[8] The romantics among you might be relieved to know that others have used the same data to argue the complete opposite: that married people are happier than non-married people in the long term (Easterlin, 2003).

[9] However, I could have measured both people in the couple because, using a multilevel model I could have treated people as being nested within 'couples' to take account of the dependency in their data.

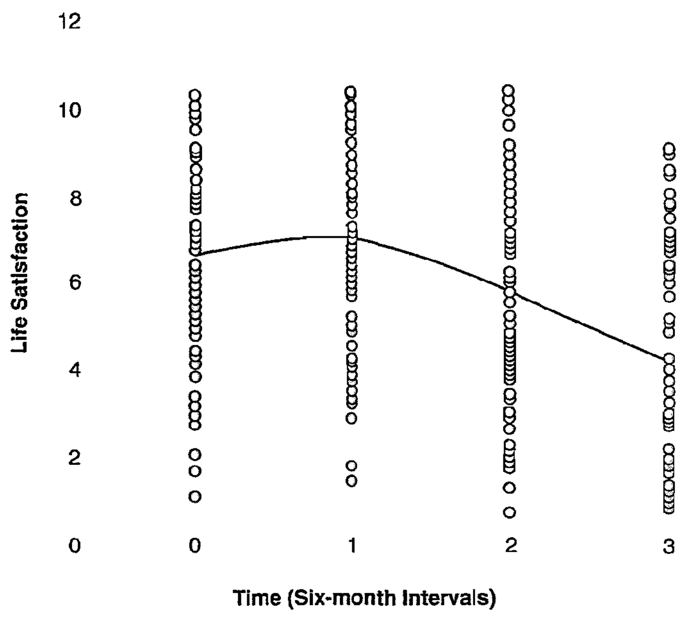


FIGURE 19.11 Life satisfaction over time

with two levels instead of three, although we could add in the location of the speed dating event as a level three variable if we had that information).

19.7.3. Restructuring the data ③

The first problem with having data measured over time is that to do a multilevel model the data need to be in a different format than what we are used to. For a repeated-measures design we normally set up the data with each row representing a person: in this case, the repeated-measures variable of time will be represented by four different columns (see section 3.9.4). We saw in Chapter 3 that this is called the 'wide' format. If we were going to run an ordinary repeated-measures ANOVA, this data layout would be fine; however, for a multilevel model we need the variable Time to be represented by a single column. We refer to this format as the 'long' format. As such we need to restructure the data.

SELF-TEST

✓ Thinking back to Chapter 3, use the *melt()* function to restructure the data into long format. If you get stuck, section 3.9.4 shows you how. Call your new dataframe *restructuredData*.

19.7.4. Setting up the basic model ④

Now that we have our data set up, we can run the analysis. Essentially, we can set up this analysis in a very similar way to the previous example. There is only one important difference: because we are working with time series data we have to model the covariance structure (see section 19.4.2). The most common way to do this is to assume a first-order autoregressive covariance structure; to remind you, this means that data points close in time are assumed to be more highly correlated than data points distant in time. In all other respects we set up the model in the same way as in the previous example.

First we fit a baseline model in which we include only the intercept. As in the previous example, this is done using the *gls()* function:

```
intercept <-gls(Life_Satisfaction ~ 1, data = restructuredData, method =
"ML", na.action = na.exclude)
```

Note that we have asked **R** to create an object called *intercept*, and we have specified that **Life_Satisfaction** is the outcome variable and that it is predicted from only the intercept (the '~1' in the function). The rest of the function specifies the data (*data = restructured-Data*) and how to estimate the model (*method = "ML"*). There is one important difference compared to the previous example. We have included a new option *na.action = na.exclude*. This is because the current data file (unlike the last example) has missing data and these missing values are specified as 'NA' in the data file. This option tells **R** what to do when it encounters an 'NA', and we have set it to exclude these cases (see R's Souls' Tip 19.2). Without this option the model would return an error.

Next, we need to fit the same model, but this time allowing the intercepts to vary across contexts (in this case we want them to vary across people). As in the previous example, we use the *lme()* function:

```
randomIntercept <-lme(Life_Satisfaction ~ 1, data = restructuredData,
random = ~1|Person, method = "ML",  na.action = na.exclude, control =
list(opt="optim"))
```

The format of this command is the same as the previous example. We create a model (called *randomIntercept*) that predicts life satisfaction from only the intercept (*Life_Satisfaction~1*), but also allows intercepts to vary across people (*random = ~1|Person*). Remember that the variable **Person** in the data set is a numeric variable that indicates whether data come from the same person. We have again asked for maximum-likelihood estimation (*method = "ML"*) and to exclude data points that are missing (*na.action = na.exclude*). However, note that there is a new option that we have not encountered before: *control = list(opt="optim")*. This option changes the optimizer that **R** uses to estimate the model. Normally the default optimizer is fine, but for these data some of the models cannot be computed using the default, so I have changed it to one that succeeds (see R's Souls' Tip 19.4).

R's Souls' Tip 19.4 Advanced options for *lme()* ③

You can use the control option in *lme()* to change the default options for the estimation procedure. A couple of the parameters you might want to change if your models won't converge are:

- *maxIter*: This sets the maximum number of iterations that **R** will use to reach a solution. The default is 50, but if your model fails to converge then you can increase this value, for example to 100, using *control = list(maxIter = 100)*.
- *opt*: This sets the optimizer that is used. The default (from **R** 2.2.0) is called *nlminb*, but there is an alternative optimizer called *optim*. If your model fails to converge it can be useful to try using a different optimizer. For example, some of the models in the honeymoon period example do not converge using the defaults, which is why we changed from the default optimizer by using *control = list(opt = "optim")*.

You can change both parameters using the same option by simply including both in the *list()* that you specify. For example, to increase the number of iterations to 2000 and to change the optimizer from *nlminb* to *opt* you would use *control = list(maxIter = 2000, opt = "optim")*. For a full list of advanced options execute *?lmeControl*.

19.7.5. Adding in time as a fixed effect ③

In a growth curve analysis, we are primarily interested in one fixed effect: time. This variable in our data set is the index variable (Time), which specifies whether the life satisfaction score was recorded at baseline (0), 6 months (1), 12 months (2) or 18 months (3). In the previous example we built up our models individually using the *lme()* function; however, for this example our models have a lot of options (*method* = *"ML"*, *na.action* = *na.exclude*, *control* = *list(opt="optim")*), which we must specify in each new model. This typing would be tedious, so we will use the *update()* function to retain everything from a previous model (including options such as the method, how to deal with missing cases, and the optimization method) but add things to it (R's Souls' Tip 19.3). We can quickly update the previous model (*randomIntercept*) to include Time as a predictor by executing:

```
timeRI<-update(randomIntercept, .~. + Time)
```

This command creates a new object in R called *timeRI*. The first part of the parenthesis tells **R** *which* model to update (in this case we have updated *randomIntercept*). The remainder tells **R** *how* to update this model: .~. simply means 'keep the previous outcome and all of the previous predictors' and + *Time* means 'add Time as a predictor'. If you want to have a look at the new model you can use *summary(timeRI)*.

19.7.6. Introducing random slopes ④

We can add a random slope to the model very simply using the *update()* function and respecifying the random part of the model. At the moment, the random part of the model is specified as *random* = *~1|Person*, which means that intercepts (~1) vary across people (**Person**). If we want slopes to vary across people as well, then we're saying that the effect of **Time** is different in different people. This is a standard growth model scenario: the rate of development or growth over time differs within entities (in this case people, but it could be companies, mice, states, hospitals, schools, geographical areas, etc.). As such, we want to update the random part of the model to be *random* = *~Time|Person*, which means that intercepts and the effect of time (*~Time*) vary across people (Person). We use the *update()* function to create a new model (called *timeRS*) which is identical to the previous model (*timeRI*) but updates the random part of the model to be *random* = *~Time|Person*:

```
timeRS<-update(timeRI, random = ~Time|Person)
```

19.7.7. Modelling the covariance structure ④

Now we have a basic random effects model, we can introduce a term that models the covariance structure or errors (see section 19.4.2). We do this by using the option *correlation* = *x*, in which *x* is one of several pre-defined covariance structures. The most likely covariance structures that'll you'll use will be (for a full list execute *?corClasses*):

- *corAR1()*: This is a first-order autoregressive covariance structure (see Jane Superbrain Box 19.1). It should be used when time points are equally spaced (as is the case in the current example).
- *corCAR1()*: This is similar to the above but for use with a continuous time covariate. Basically you should use this covariance structure if your time points are not equally spaced.
- *corARMA()*: Another autoregressive error structure, but this one allows the correlation structure to involve a moving average of error variance (see Jane Superbrain Box 19.1).

We can add a covariance structure to the model using the *update()* function to create a new model (called *ARModel*) which is identical to the previous model (*timeRS*) but adds in a first-order autoregressive covariance structure:

```
ARModel<-update(timeRS, correlation = corAR1(0, form = ~Time|Person))
```

Note that we have used *correlation = corAR1(0, form = ~Time | Person)*. We could have used the default setting, which would be to use *correlation = corAR1()*, but this would include only the random intercept (it would be the same as specifying *correlation = corAR1(0, form = ~1 | Person)*.

JANE SUPERBRAIN 19.1

Autoregressive and moving average models ④

Autoregressive models (AR) are very difficult to understand. I do not really understand them, and I doubt I ever will. Imagine that we have a time series (Y_t) and we adjust this by subtracting the mean $(y_t = Y_t - \bar{Y})$. The t in these equations just represents different points in time. As far as I can gather, if we have a first-order autoregressive model, AR(1), then we predict these adjusted values from the adjusted *values at the previous time point* (i.e., $t - 1$). We can use a standard linear model to do this:

$$y_t = -a_1 y_{t-1} + e_t$$

The $-a$ is known as the lag or autoregressive coefficient, and e_t is the residual or error at time t. You can hopefully see that this is a simple linear model in which values at one time are predicted from values at a previous time (the word *autoregressive* reflects the fact that values are predicted from themselves).

A second-order autoregressive model. AR(2), is much the same except that we're interested not just in the previous time point, but in the previous two time points. The model simply expands to include this extra time point:

$$y_t = -a_1 y_{t-1} - a_2 y_{t-2} + e_t$$

You should be able to extend this basic logic to understand third- and fourth-order autoregressive models. In these models, residuals are assumed not to correlate; in other words, errors at one time point are not believed to correlate with errors at another time point. The data themselves correlate at different time points, but the errors don't.

However, we might want to assume that the residuals also correlate over time. In other words, our data at time t can be predicted not just from the data at the previous time point but also the error at previous time points. This is known as a moving average (MA) model. Like AR models, a first-order moving average model factors in residuals at the current time point (e_t) and also residuals from the previous time point (e_{t-1}). A second-order MA model would factor in residuals for the current time point, the previous time point, and two points back in time (e_{t-2}), and so on. So, for example, if we had an AR(1) and MA(1) our model becomes:

$$y_t = -a_1 y_{t-1} + e_t + c_1 e_{t-1}$$

Note that this is the same as the AR(1) model except that there is an extra term representing the error from the previous time point, e_{t-1}, which is multiplied by c_1, a coefficient representing the first-order moving average. Models that combine autoregressive and moving average models are known as ARMA models. ARMA models have two parameters: p defines the order of the autoregressive part of the model, q specifies the order of the moving average part of the model. In both cases 1 = first-order, 2 = second-order and so on. Therefore, ARMA($p = 2$, $q = 1$) would be a second-order autoregressive model with a first-order moving average, ARMA($p = 2$, $q = 2$) would be a second-order autoregressive model with a second-order moving average. Hopefully you get the general gist because my brain is literally about to explode all over my screen.

19.7.8. Comparing models ③

So far we have created five models: (1) a baseline model predicting life satisfaction from only the intercept (*interceptOnly*); (2) a model with random intercepts across people (*randomIntercept*); (3) a model with time as a predictor of life satisfaction and random intercepts across people (*timeRI*); (4) a model with time as a predictor, a random effect of time over people and random intercepts (*timeRS*); and (5) a model with time as a predictor, random effects of time across people, a random effect of intercepts across people, and a first-order autoregressive covariance structure (*ARModel*). Let's now look at how these models fit the data. Each time we have added only one new component to the model so we can compare them with the log-likelihood as we did in the previous example. We can compare all five models by executing:

```
anova(intercept, randomIntercept, timeRI, timeRS, ARModel)
```

```
           Model df  AIC      BIC       logLik   Test   L.Ratio p-value
intercept    1   2 2064.053 2072.217 -1030.026
randomInt    2   3 1991.396 2003.642  -992.698 1 vs 2  74.657  <.0001
timeRI       3   4 1871.728 1888.057  -931.864 2 vs 3 121.667  <.0001
timeRS       4   6 1874.626 1899.120  -931.313 3 vs 4   1.102  0.5763
ARModel      5   7 1872.891 1901.466  -929.445 4 vs 5   3.736  0.0533
```

Output 19.18

The resulting output, in Output 19.18, shows that adding a random intercept significantly improved the fit of the model, $\chi^2(1) = 74.66$, $p < .0001$. Similarly, adding the fixed effect of time to the model significantly improved the fit compared to the previous model, $\chi^2(1) = 121.67$, $p < .0001$. However, adding a random slope for the effect of time across participants did not significantly improve the model, $\chi^2(2) = 1.10$, $p = .576$. Finally, adding a first-order autoregressive covariance structure did more or less significantly improve the model, $\chi^2(1) = 3.74$, $p = .053$. Note that for each model the degrees of freedom change by 1 because we have added only a single parameter;[10] this change in degrees of freedom is used for the log-likelihood test.

We can take a quick look at the final model, and the confidence intervals for the parameter estimates within it by using:

```
Summary(ARModel); intervals(ARModel)
```

Output 19.19 shows the resulting model summary and Output 19.20 shows the 95% confidence intervals. The effect of time, $b = -0.87$ $(-1.03, -0.71)$, $t(322) = -10.97$, $p < .001$, was highly significant, indicating that life satisfaction significantly changed over the 18 month period (see Figure 19.11). In addition, the standard deviation of intercepts was 1.62 (1.31, 2.02), and for the effect of time across people (slopes) was 0.05 (0.00, 41.83). Neither of the confidence intervals crosses zero, implying that this variance in slopes and intercepts was significant. Note that in the case of the slopes, this finding contradicts the results of the log-likelihood statistic, which implied that adding random slopes did not significantly improve the model (Output 19.18). The approximate confidence interval for slopes is very wide and not symmetrical, which implies that we might be wise to give more weight to the log-likelihood.

[10] The exception is the model where we add random slopes. See earlier for an explanation of why the change in degrees of freedom is 2 in this case.

```
Linear mixed-effects model fit by maximum likelihood
 Data: restructuredData
       AIC      BIC    logLik
  1872.891 1901.466 -929.4453

Random effects:
 Formula: ~Time | Person
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev     Corr
(Intercept) 1.62767553 (Intr)
Time        0.04782877 -0.062
Residual    1.74812486

Correlation Structure: AR(1)
 Formula: ~Time | Person
 Parameter estimate(s):
      Phi
0.2147812
Fixed effects: Life_Satisfaction ~ Time
                Value  Std.Error  DF   t-value p-value
(Intercept)  7.131470 0.21260192 322  33.54377       0
Time        -0.870087 0.07929275 322 -10.97310       0
 Correlation:
     (Intr)
Time -0.527

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.08400991 -0.62083911  0.06392492  0.59512953  2.49161500

Number of Observations: 438
Number of Groups: 115
```

Output 19.19

```
Approximate 95% confidence intervals

 Fixed effects:
                lower       est.      upper
(Intercept)  6.714162  7.1314700  7.5487782
Time        -1.025728 -0.8700874 -0.7144467
attr(,"label")
[1] "Fixed effects:"

 Random Effects:
  Level: Person
                               lower        est.      upper
sd((Intercept))         1.314720e+00  1.62767553  2.0151263
sd(Time)                5.468419e-05  0.04782877 41.8327717
cor((Intercept),Time) -6.937502e-01 -0.06192455  0.6237635

 Correlation structure:
          lower      est.    upper
Phi 0.002025179 0.2147812 0.408935
attr(,"label")
[1] "Correlation structure:"
```

```
Within-group standard error:
   lower      est.    upper
1.542913 1.748125 1.980630
```

Output 19.20

19.7.9. Adding higher-order polynomials ③

We have seen that the main effect of time is significant. This main effect is the linear trend of time. However, Figure 19.11, seems to show a more curvilinear change over time (satisfaction first increases from baseline to 6 months before declining after 6 months). To capture this trend we would need to add a quadratic or perhaps even cubic trend (refer back to Figure 19.10). There are several ways in R to look for trends. One way to add quadratic trends is to do it manually. We saw in Figure 19.10 that a quadratic trend equates to $time^2$ and that a cubic trend is simply $time^3$. We could, therefore, simply create new predictor variables in our dataframe that are time multiplied by itself (time^2) or time multiplied by itself twice ($time^3$). We could then enter these new variables as predictors into the model.

Fortunately, rather than computing new variables, R can create these new predictors 'on the fly'. To create the quadratic term we simply specify *I(Time ^ 2)* as a new predictor. 'Time ^ 2' is **R**'s way of writing 'time^2' (the ^ means 'to the power of'); because arithmetic operators such as +, *, – and ^ can be used to define the form of a model (e.g., satisfaction~gender + age + age*gender) we need to enclose 'Time ^ 2' within the *I()* function so that **R** knows to treat it as an arithmetic operator rather than part of the model specification. The last model we looked at was called *ARModel*, and included the main effect of *Time* as a predctor. We can use *update()* to create a new model (*timeQuadratic*) that adds the quadratic term to this model:

```
timeQuadratic<-update(ARModel, .~. + I(Time^2))
```

We can create a model (*timeCubic*) that adds a cubic term in exactly the same way as for the quadratic trend. This time, we update the quadratic model (*timeQuadratic*) so that it includes $time^3$, which is done the same as for the quadratic trend except that we specify time cubed rather than squared, 'I(Time ^ 3)'. We can compare these two new models with the model that included only the linear trend of time (*ARModel*) using the *anova()* function and ask for a summary of the final model using the *summary()* and *intervals()* functions.

```
timeCubic <-update(timeQuadratic, .~. + I(Time^3))
anova(ARModel, timeQuadratic, timeCubic)
summary(timeCubic)
intervals(timeCubic)
```

Output 19.21 shows the model comparison. It is clear from this that adding the quadratic term to the model significantly improves the fit, $\chi^2(1) = 57.35, p < .0001$; however, adding in the cubic trend does not, $\chi^2(1) = 3.38, p = .066$.

```
              Model df  AIC       BIC      logLik   Test  L.Ratio p-value
ARModel           1  7 1872.89 1901.466 -929.445
timeQuadratic 2    8 1817.54 1850.202 -900.772 1 vs 2 57.347  <.0001
timeCubic     3    9 1816.16 1852.901 -899.081 2 vs 3  3.382  0.0659
```

Output 19.21

Looking at the summary of the final model, the fixed effects (Output 19.22) and the confidence intervals (Output 19.23) tell us that the linear, $b = 1.55$ (0.61, 2.48), $t(320) = 3.24$, $p < .01$, and quadratic, $b = -1.33$ (−2.15, −0.50), $t(320) = -3.15$, $p < .01$, both significantly described the pattern of the data over time; however, the cubic trend was not significant, $b = 0.17$ (−0.01, 0.35), $t(320) = 1.84$, $p > .05$. This confirms what we already know from comparing the fit of successive models. The trend in the data is best described by a second-order polynomial, or a quadratic trend. This reflects the initial increase in life satisfaction 6 months after finding a new partner but a subsequent reduction in life satisfaction at 12 and 18 months after the start of the relationship (Figure 19.11). It's worth remembering that this quadratic trend is only an *approximation*: if it were completely accurate then we would predict from the model that couples who had been together for 10 years would have negative life satisfaction, which is impossible given the scale we used to measure it.

```
Linear mixed-effects model fit by maximum likelihood
 Data: restructuredData
      AIC      BIC    logLik
  1816.162 1852.902 -899.0808

Random effects:
 Formula: ~Time | Person
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 1.8826725 (Intr)
Time        0.4051351 -0.346
Residual    1.4572374

Correlation Structure: AR(1)
 Formula: ~Time | Person
 Parameter estimate(s):
      Phi
0.1326346
Fixed effects: Life_Satisfaction ~ Time + I(Time^2) + I(Time^3)
                Value Std.Error  DF   t-value p-value
(Intercept)  6.634783 0.2230273 320 29.748744  0.0000
Time         1.546635 0.4772221 320  3.240913  0.0013
I(Time^2)   -1.326426 0.4209411 320 -3.151098  0.0018
I(Time^3)    0.171096 0.0929297 320  1.841131  0.0665
 Correlation:
          (Intr) Time   I(T^2)
Time      -0.278
I(Time^2)  0.139 -0.951
I(Time^3) -0.098  0.896 -0.987

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-2.58597365 -0.54411056 -0.04373592  0.50525444  2.78413461

Number of Observations: 438
Number of Groups: 115
```

Output 19.22

The outputs for the final model also tell us about the random parameters in the model. First of all, the standard deviation of the random intercepts was 1.88 (1.49, 2.39). The fact that the 95% confidence interval doesn't cross zero suggests that we were correct to assume that life satisfaction at baseline varied significantly across people. Also, the variance of slope of time varied significantly across people, $SD = 0.41$ (0.17, 0.96). The confidence interval again does not cross zero, suggesting that the change in life satisfaction over time varied significantly across people too. Finally, the correlation between the slopes and intercepts, –0.35 (–0.67, 0.10) suggests that as intercepts increased, the slope decreased (although the confidence interval crosses zero so this trend is not significant).

```
Approximate 95% confidence intervals


 Fixed effects:
                   lower       est.      upper
(Intercept)  6.19800573  6.6347826  7.0715595
Time         0.61204293  1.5466350  2.4812271
I(Time^2)   -2.15079781 -1.3264264 -0.5020551
I(Time^3)   -0.01089790  0.1710958  0.3530895
attr(,"label")
[1] "Fixed effects:"

 Random Effects:
  Level: Person
                            lower       est.      upper
sd((Intercept))         1.4852030  1.8826725 2.38651276
sd(Time)                0.1705194  0.4051351 0.96255585
cor((Intercept),Time) -0.6738687 -0.3461486 0.09538264

 Correlation structure:
         lower      est.     upper
Phi -0.1856231 0.1326346 0.4257069
attr(,"label")
[1] "Correlation structure:"

 Within-group standard error:
   lower      est.    upper
1.173241 1.457237 1.809978
```

Output 19.23

Another way to test for trends over time is by converting Time to power polynomials. This is achieved with a simple function *poly()*. Within this function you specify the variable that you want to be converted, and the number of polynomials you want (up to the number of time points that you measured minus 1). For example, *poly(Time, 1)* will create a linear trend, *poly(Time, 2)* creates a linear and quadratic and *poly(Time, 3)* creates a linear, quadratic and cubic trend. In our current example we had four time points so a cubic trend is the highest order polynomial that we can have; if we, for example, specified *poly(Time, 4)* we would get an error.

```
Linear mixed-effects model fit by maximum likelihood
 Data: restructuredData
        AIC      BIC    logLik
  1816.162 1852.902 -899.0808

Random effects:
 Formula: ~Time | Person
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 1.8826725 (Intr)
Time        0.4051351 -0.346
Residual    1.4572374

Correlation Structure: AR(1)
 Formula: ~Time | Person
 Parameter estimate(s):
      Phi
0.1326346
Fixed effects: Life_Satisfaction ~ poly(Time, 3)
                     Value Std.Error  DF   t-value p-value
(Intercept)       5.938943  0.182953 320  32.46157  0.0000
poly(Time, 3)1 -20.615766  1.759420 320 -11.71736  0.0000
poly(Time, 3)2 -11.682913  1.418904 320  -8.23376  0.0000
poly(Time, 3)3   2.439191  1.324833 320   1.84113  0.0665
 Correlation:
               (Intr) p(T,3)1 p(T,3)2
poly(Time, 3)1 -0.009
poly(Time, 3)2 -0.016  0.027
poly(Time, 3)3  0.004 -0.035   0.014

Standardized Within-Group Residuals:
        Min          Q1          Med          Q3         Max
-2.58597365 -0.54411056 -0.04373592  0.50525444  2.78413461

Number of Observations: 438
Number of Groups: 115
```

Output 19.24

The advantage of this method of creating polynomials is that the resulting predictors are orthogonal (i.e., independent). By using *Time*, *Time*^{2} and *Time*^{3} we create predictors that are highly correlated, but by using *poly()* we create predictors that are completely uncorrelated. This means that we can evaluate one trend without it being affected by another. If we wanted to add our trends using this method we can again use the *update()* function to respecify the model with AR(1) covariance structure (*ARModel*). Remember that this model already has time as a predictor, and we need to get rid of this predictor and replace it with our polynomials. To do this we just respecify the outcome and predictors within the update function and this will overwrite the previous predictors:

```
polyModel<-update(ARModel, .~ poly(Time, 3))
```

Remember that '~' means 'predicted from'; the '.' means 'use the same thing as in the existing model'. As such, .~ *poly(Time, 3)* translates as 'use the same outcome as in the existing model but predict it from *poly(Time, 3)*'. In other words, the previous predictor of **Time** that was specified for the *ARModel* will be replaced by the linear, quadratic and cubic polynomials for **Time** that are created by the *poly()* function. All other parts of the *ARModel* (i.e., the random effects and covariance structure) remain unchanged.

Output 19.24 shows the model summary (*summary(polymodel)*) for the model including the polynomials. We won't dwell on this output other than to say that if you compare it to Output 19.22 it shows the same profile of results: the linear (*poly(Time, 3)1*) and quadratic (*poly(Time, 3)2*) trends are significant, whereas the cubic (*poly(Time, 3)3*) was just non-significant ($p = .067$). The regression coefficients are different (because these contrasts are orthogonal), but basically the same pattern or results emerges.

SELF-TEST

✓ We have used the *update()* function in this second example. To get some practice at specifying multilevel models, try building each of the models in this example but specifying each one in full.

19.7.10. Further analysis ④

It's worth pointing out that I've kept this growth curve analysis simple to give you the basic tools. In the example I allowed only the linear term to have a random intercept and slopes, but given that we discovered that a second-order polynomial described the change in responses, we could redo the analysis and allow random intercepts and slopes for the second-order polynomial also. To do these we would just have to specify these terms in the random part of the model. If we were to do this it would make sense to add the random components one at a time and test whether they have a significant impact on the model by comparing the log-likelihood values or other fit indices.

Also, the polynomials I have described are not the only ones that can be used. You could test for a logarithmic trend over time, or even an exponential one.

CRAMMING SAM'S TIPS Growth models

- Growth models are multilevel models in which changes in an outcome over time are modelled using potential growth patterns.
- These growth patterns can be linear, quadratic, cubic, logarithmic, exponential, or anything you like really.
- The hierarchy in the data is that time points are nested within people (or other entities). As such, it's a way of analysing repeated-measures data that have a hierarchical structure.
- The *anova()* function can be used to compare the overall fit of hierarchical models. The resulting change in the log-likelihood and the significance of this change can be used to ascertain if the fit has been improved (a significant change equates to a significant improvement). The AIC and BIC can also be compared across models (but not significance tested).
- The *fixed effects* tell you whether the growth functions that you have entered into the model significantly predict the outcome. If the *p*-value is less than .05 then the effect is significant.
- The *intervals()* function can be used to get confidence intervals for model parameters. These intervals can tell us how much intercepts and slopes varied over our level 1 variable, and whether this variance is significant (if the interval does not cross zero, it is significant).
- An autoregressive covariance structure, AR(1), is often assumed in time course data such as that in growth models.

Labcoat Leni's Real Research 19.1 A fertile gesture ③

Miller, G., Tybur, J. M., & Jordan, D. B. (2007). *Evolution and Human Behavior, 28*, 375–381.

Most female mammals experience a phase of 'estrus' during which they are more sexually receptive, proceptive, selective and attractive. As such, the evolutionary benefit to this phase is believed to be to attract mates of superior genetic stock. However, some people have argued that this important phase became uniquely lost or hidden in human females. Testing these evolutionary ideas is exceptionally difficult, but Geoffrey Miller and his colleagues came up with an incredibly elegant piece of research that did just that. They reasoned that if the 'hidden-estrus' theory is incorrect then men should find women most attractive during the fertile phase of their menstrual cycle compared to the pre-fertile (menstrual) and post-fertile (luteal) phase.

(Continued)

(Continued)

To measure how attractive men found women in an ecologically valid way, they came up with the ingenious idea of collecting data from women working at lap-dancing clubs. These women maximize their tips from male visitors by attracting more dances. In effect the men 'try out' several dancers before choosing a dancer for a prolonged dance. For each dance the male pays a 'tip'. Therefore, the greater the number of men choosing a particular woman, the more her earnings will be. As such, each dancer's earnings are a good index of how attractive the male customers have found her. Miller and his colleagues argued, therefore, that if women do have an estrus phase then they will be more attractive during this phase and therefore earn more money. This study is a brilliant example of using a real-world phenomenon to address an important scientific question in an ecologically valid way.

The data for this study are in the file **Miller et al. (2007).dat**. The researchers collected data via a website from several dancers (**ID**), who provided data for multiple lap-dancing shifts (so for each person there are several rows of data). They also measured what phase of the menstrual cycle the women were in at a given shift (**Cyclephase**), and whether they were using hormonal contraceptives (**Contraceptive**) because this would affect their cycle. The outcome was their earnings on a given shift in dollars (**Tips**).

A multilevel model can be used here because the data are unbalanced: the women differed in the number of shifts they provided data for (the range was 9 to 29 shifts); multilevel models can handle this problem.

Labcoat Leni wants you to carry out a multilevel model to see whether **Tips** can be predicted from **Cyclephase**, **Contraceptive** and their interaction. Is the 'estrus-hidden' hypothesis supported? Answers are in the additional material on the companion website (or look at page 378 in the original article).

19.8. How to report a multilevel model ③

Specific advice on reporting multilevel models is hard to come by. Also, the models themselves can take on so many forms that giving standard advice is hard. If you have built up your model from one with only fixed parameters to one with a random intercept, and then random slope, it is advisable to report all stages of this process (or at the very least report the fixed-effects-only model and the final model). For any model you need to say something about the random effects. For the final model of the cosmetic surgery example you could write something like:

✓ The relationship between surgery and quality of life showed significant variance in intercepts across participants, *SD* = 5.48 (95% CI: 3.31, 9.07), $\chi^2(1) = 107.65$, $p < .0001$. In addition, the slopes varied across participants, *SD* = 5.42 (3.13, 9.37), $\chi^2(2) = 38.87$, $p < .0001$, and the slopes and intercepts were negatively and significantly correlated, cor = −.95 (−.99, −.60).

For the model itself, you have two choices. The first is to report the results in the text, with the *b*-values, *t*s and degrees of freedom for the fixed effects, and then report the parameters for the random effects in the text as well. The second is to produce a table of parameters as you would for regression. For example, we might report our cosmetic surgery example as follows:

✓ Quality of life before surgery significantly predicted quality of life after surgery, $b = 0.31$, $t(262) = 5.75$, $p < .001$, surgery did not significantly predict quality of life, $b = -3.19$, $t(262) = -1.46$, $p = .15$, but the reason for surgery, $b = -3.52$, $t(262) = -3.08$, $p < .01$, and the interaction of the reason for surgery and surgery, $b = 4.22$, $t(262) = 2.48$, $p < .05$, both did significantly predict quality of life. This interaction was broken down by conducting separate multilevel models on the 'physical reason' and 'attractiveness reason'. The models specified were the same as the main model but excluded the main effect and interaction term involving the reason for surgery. These analyses showed that for those operated on only to change their appearance, surgery almost significantly predicted quality of life after surgery, $b = -4.31$, $t(87) = -1.89$, $p = .06$:

quality of life was lower after surgery compared to the control group. However, for those who had surgery to solve a physical problem, surgery did not significantly predict quality of life, $b = 1.20$, $t(166) = 0.57$, $p = .57$. The interaction effect, therefore, reflects the difference in slopes for surgery as a predictor of quality of life in those who had surgery for physical problems (slight positive slope) and those who had surgery purely for vanity (a negative slope).

Alternatively we could present parameter information in a table:

	b	*SE b*	*95% CI*
Baseline QoL	0.31	0.05	0.20, 0.41
Surgery	−3.19	2.19	−7.45, 1.08
Reason	−3.52	1.14	−5.74, −1.29
Surgery × Reason	4.22	1.70	0.90, 7.54

What have I discovered about statistics? ②

Writing this chapter was quite a steep learning curve for me. I've been meaning to learn about multilevel modelling for ages, and now I finally feel like I know something. This is pretty amazing considering that the bulk of the reading and writing was done between 11pm and 3am over many nights. However, despite now feeling as though I understand them, I don't, and if you feel like you now understand them then you're wrong. This sounds harsh, but sadly multilevel modelling is very complicated and we have scratched only the surface of what there is to know. Multilevel models often fail to converge, with no apology or explanation, and trying to fathom out what's happening can feel like hammering nails into your head.

Needless to say, I didn't mention any of this at the start of the chapter because I wanted you to read it. Instead, I lulled you into a false sense of security by looking gently at how data can be hierarchical and how this hierarchical structure can be important. Most of the tests in this book simply ignore the hierarchy. We also saw that hierarchical models are just basically a fancy regression in which you can estimate the variability in the slopes and intercepts within entities. We saw that you should start with a model that ignores the hierarchy and then add in random intercepts and slopes to see if they improve the fit of the model. Having submerged ourselves in the warm bath of standard multilevel models, we moved on to the icy lake of growth curves. We saw that there are ways to model trends in the data over time (and that these trends can also have variable intercepts and slopes). We also discovered that these trends have long confusing names like fourth-order polynomial. We asked ourselves why they couldn't have a sensible name, like Kate. In fact, we decided to ourselves that we'd secretly call a linear trend Kate, a quadratic trend Benjamin, a cubic trend Zoë, and a fourth-order trend Doug. 'That will show the statisticians' we thought to ourselves, and felt a little bit self-satisfied too.

We also saw that after years of denial, my love of making a racket got the better of me. This brings my life story up to date. Admittedly I left out some of the more colourful bits, but only because I couldn't find an extremely tenuous way to link them to statistics. We saw that over my life I managed to completely fail to achieve any of my childhood dreams. It's OK, I have other ambitions now (a bit smaller scale than 'rock star') and I'm looking forward to failing to achieve them too. The question that remains is whether there is life after *Discovering Statistics*. What effect does writing a statistics book have on your life?

R packages used in this chapter

car	nlme
ggplot2	reshape

R functions used in this chapter

aov()	lme()
anova()	loglik()
ggplot()	melt()
gls()	poly()
I()	summary()
intervals()	update()
lm()	

Key terms that I've discovered

AIC	Group mean centring
AR(1)	Growth curve
BIC	Multilevel linear model
Centring	Polynomial
Diagonal	Random coefficient
Fixed coefficient	Random effect
Fixed effect	Random intercept
Fixed intercept	Random slope
Fixed slope	Random variable
Fixed variable	Unstructured
Grand mean centring	Variance components

Smart Alex's tasks

- Task 1: Using the cosmetic surgery example, run the analysis described in section 19.6.9 but also including **BDI**, **Age** and **Gender** as fixed effect predictors. What differences does including these predictors make? ④
- Task 2: Using our growth model example in this chapter, analyse the data but include **Gender** as an additional covariate. Does this change your conclusions? ④
- Task 3: **Getting kids to exercise (Hill, Abraham, & Wright, 2007)**: The purpose of this research was to examine whether providing children with a leaflet based on the 'theory of planned behaviour' increases children's exercise. There were four different interventions (**Intervention**): a control group, a leaflet, a leaflet and quiz, and a leaflet and plan. A total of 503 children from 22 different classrooms were sampled (**Classroom**). It was not practical to have children in the same classrooms in different

conditions, therefore the 22 classrooms were randomly assigned to the four different conditions. Children were asked 'On average over the last three weeks, I have exercised energetically for at least 30 minutes ______ times per week' after the intervention (**Post_Exercise**). Run a multilevel model analysis on these data (**Hill et al. (2007).dat**) to see whether the intervention affected the children's exercise levels (the hierarchy in the data is: children within classrooms within interventions). ④

- **Task 4**: Repeat the above analysis but include the pre-intervention exercise scores (**Pre_Exercise**) as a covariate. What difference does this make to the results? ④

Answers can be found on the companion website.

Further reading

Kreft, I., & de Leeuw, J. (1998). *Introducing multilevel modeling*. London: Sage. (This is a fantastic book that is easy to get into but has a lot of depth too.)

Tabachnick, B. G., & Fidell, L. S. (2001). *Using multivariate statistics* (4th ed.). Boston: Allyn & Bacon. (Chapter 15 is a fantastic account of multilevel linear models that goes a bit more in depth than I do.)

Twisk, J. W. R. (2006). *Applied multilevel analysis: A practical guide*. Cambridge: Cambridge University Press. (An absolutely superb introduction to multilevel modelling. This book is exceptionally clearly written and is aimed at novices. Without question, this is the best beginner's guide that I have read.)

Interesting real research

Cook, S. A., Rosser, R., & Salmon, P. (2006). Is cosmetic surgery an effective psychotherapeutic intervention? A systematic review of the evidence. *Journal of Plastic, Reconstructive & Aesthetic Surgery*, *59*, 1133–1151.

Miller, G., Tybur, J. M., & Jordan, B. D. (2007). Ovulatory cycle effects on tip earnings by lap dancers: Economic evidence for human estrus? *Evolution and Human Behavior*, *28*, 375–381.