Chapter 3
Hierarchical models: linear models, generalized linear models and random effects

Written by Marc Kéry, Sep 2013; this is material for a draft chapter of a book by Kéry and Royle. Version November 2013

Three essential statistical concepts are introduced here, and all three are typically present in hierarchical model: linear models, generalised linear models and random effects. We will illustrate all of them using one toy data set where we imagine we had measured wing span and body length and recorded the sex of a total of nine blue-eyed hooktail dragonflies (*Onychogomphus uncatus*; Fig. 1) in three populations. The objective of our analysis consists in investigating the relationships between wing span (the response), and population, sex and body length (three explanatory variables). The former two are factors with three and two levels, respectively, i.e., categorical explanatory variables in which numbers have no numerical meaning; they are mere labels or names. In contrast, wing length is a continuous explanatory variable, or covariate, where the numbers do have a quantitative meaning. Importantly, however, when fitting linear models, **R** and other stats packages *internally* do not make a difference between categorical and continuous explanatory variables (although the former must be declared a factor). Rather, all linear models are represented exactly as a multiple linear regression. Effects of factors are specified using two or more indicator or dummy variables that contain only 0s and 1s and that code for the presence or absence of an effect in the expected response of each dragonfly.

Fig. 1: Male blue-eyed hooktail (*Onychogomphus uncatus*), Aragon, Spain, 2013 (Photo: M. Kéry)

Here are the data and a plot of them (Fig. 2).

```
# Define and plot data
pop <- factor(rep(c(1, 1, 1, 2, 2, 2, 3, 3, 3), 10))
wing <- rep(c(10.82, 10.56, 11.84, 12.09, 11.74, 13.51, 11.39, 13.04, 12.85), 10)
length <- rep(c(6.77, 8.27, 9.20, 6.88, 7.69, 8.96, 6.88, 8.15, 9.20), 10)
sex <- factor(rep(c(1,2,1,2,1,2,1,2,1), 10))     # Assume males are 1, females 2


cbind(pop, wing, length, sex)   # Note that sex 1 = male and sex 2 = females
      pop  wing length sex
 [1,]   1 10.82   6.77   1
 [2,]   1 10.56   8.27   2
 [3,]   1 11.84   9.20   1
 [4,]   2 12.09   6.88   2
 [5,]   2 11.74   7.69   1
 [6,]   2 13.51   8.96   2
 [7,]   3 11.39   6.88   1
 [8,]   3 13.04   8.15   2
 [9,]   3 12.85   9.20   1

plot(length[sex == 1], wing[sex == 1], col = c("red", "red", "blue", "green",
"green"), xlim = c(6.5, 9.5), ylim = c(10.5, 13.5), cex = 1.5, lwd = 2,
frame.plot = FALSE, las = 1, pch = 17, xlab = "Body length", ylab = "Wing span")
points(length[sex == 2], wing[sex == 2], col = c("red", "blue", "blue", "green",
"green"), cex = 1.5, pch = 16)
```

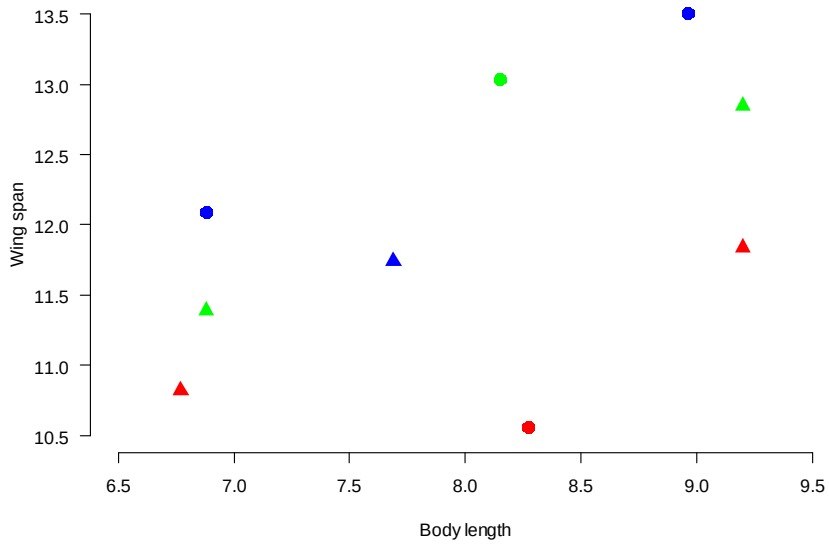

Fig. 2: Relationship between wing span, body length, sex and population in the *O. uncatus* toy data set. Colors code for the three populations; circles denote females and triangles males.

3.1 Linear models

In a linear model, effects of covariates on a response appear in a purely additive fashion. That is, the model is linear in the parameters, but does not necessarily represent a straight line when

shown as a graph. For instance, models with polynomial terms do not translate into a straight-line graph. A linear model can be described in many different ways, for instance:

(1) in words, e.g., when we say something like "`pop` and `length` act additively ...",

(2) using specific names or labels for least-squares techniques which *imply* a linear model of a certain form, e.g., "t-test", "linear regression", "ANOVA" or "ANCOVA",

(3) in graphs, e.g., using lines and bars,

(4) in algebra, e.g., $y_i \sim Normal(\alpha_j + \beta * x_i, \sigma^2)$,

(5) using matrix algebra, such as $\mathbf{y} = \mathbf{X}\beta + \varepsilon$,

(6) as a system of equations,

(7) in the R model definition language, e.g. `lm(wing ~ pop + length)`,

(8) and finally, using the BUGS language, e.g., `mu[i] <- alpha[pop[i]] + beta * length[i]`.

In our statistical modeling, most of us have become accustomed to defining certain types of linear models using click-and-point techniques in many statistical packages or using a slick formula definition language. Especially the latter has undoubtedly advantages, because it allows one to define a large range of linear models quickly and error-free. The drawbacks are that it is easy to fit linear models without actually understandig them, or what the parameter estimates mean. In addition, some non-standard linear models may not be fitted or only in a very complicated way. When fitting models in BUGS software, no such shortcuts to defining linear (or nonlinear) models are available, so you must know exactly what kind of model you fit how and in what parameterisation. Therefore, we will now first look at how to describe a certain linear model in the different ways listed above. Then, we give a summary of some typical linear models that can be fitted for factors and continuous explanatory covariates. We will also show how to combine a factor and a continuous covariate, and two factors, in additive and interactive ways.

We will next fit a linear model where `pop` and `length` act in an additive fashion on wing span, namely according to the model underlying a technique that used to be called a "main-effects analysis of covariance (ANCOVA)". When plotted as a graph, this model is represented by three parallel straight lines for our data set, one for the dragonflies in each population (see also Fig. 3). We can fit this model in **R** by issuing the following commands:

```
summary(fm <- lm(wing ~ pop + length))

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.4801     1.7334   3.738   0.0135 *
pop2          1.5079     0.4854   3.107   0.0267 *
pop3          1.3552     0.4828   2.807   0.0377 *
length        0.5685     0.2103   2.703   0.0426 *
[ ... ]
Residual standard error: 0.5913 on 5 degrees of freedom
Multiple R-squared:  0.782,     Adjusted R-squared:  0.6512
F-statistic: 5.979 on 3 and 5 DF,  p-value: 0.04151
```

How many parameters does our model have ? Where are the estimates of these parameters presented in the **R** output ? What do these parameters mean ? Let's look at the description of this model in algebra. We can write the model just fitted as follows.

$$wing_i = \mu + \alpha_j + \beta * length_i + \varepsilon_i \text{, with } \varepsilon_i \sim Normal(0, \sigma^2)$$

Here, $wing_i$ is the response of unit (data point, individual dragonfly, row in the data set) i and $length_i$ is the value of the continuous explanatory variable `length` for dragonfly i. The intercept μ is a constant shared by the response of all nine dragonflies, while α_j is a vector of length three, corresponding to the number of levels in the factor `pop`, which contains the *effects* of each population. For population 1, we have α_1, for population 2 α_2 and population 3 α_3. Parameter β is the expected change in wing span per unit-change in body length, while ε_i is the unexplained "rest" in wing span, which is not accounted for by population membership or body length of dragonfly *i*, i.e., it is the residual. For all individuals, the ε_i are assumed to be independent draws from a single normal distribution with a common variance parameter that is estimated.

This model is overparameterised or parameter-redundant, i.e., we try to estimate one too many parameters. Specifically, we cannot estimate both the intercept μ and a full parameter vector α of length three. Rather, one of the population effects α_j has to be set to zero to make the model identifiable. Customarily in **R**, the parameter corresponding to the first level of a factor is set to zero, i.e., the constraint $\alpha_1 = 0$ is enforced. As a consequence, the intercept becomes the intercept of the regression line of dragonflies in population 1, i.e., the expected wing span of a (hypothetical) dragonfly of length zero in population 1. The remaining two elements of the parameter vector α, which are not fixed but estimated, become the differences between the intercepts of the regression lines in population 2 and 3, relative to the intercept in population 1 (this is why only estimates for pop2 and pop3 are shown in the R output above). Hence, in this parameterisation of the model, population 1 serves as a baseline or reference and the effects of the population factor represent comparisons with that population. This parameterisation of the effects of the factor `pop` can be called the treatment contrast or effects parameterisations.

There are other ways to write what is essentially the same linear model, and these are called different parameterisations of the model. One of them can be written as this:

$$wing_i = \alpha_j + \beta * length_i + \varepsilon_i \text{, with } \varepsilon_i \sim Normal(0, \sigma^2)$$

Here, everything is the same as before (except that there is no longer a μ), only the meaning of the parameters α_j representing the effects of the factor `pop` has changed somewhat. Now, all three elements of this parameter vector are estimable, since there is no longer a separate intercept (μ), and they represent directly the intercepts of the three regression lines, i.e., the expected wing spans of a dragonfly at length zero in population 1–3. This parameterisation of the effects of the factor `pop` can be called the means parameterisations. It is fit in **R** as follows, namely by "subtracting the intercept":

```
summary(fm <- lm(wing ~ pop-1 + length))
```

```
Coefficients:
       Estimate Std. Error t value Pr(>|t|)
pop1     6.4801     1.7334   3.738  0.01346 *
pop2     7.9879     1.6847   4.742  0.00514 **
pop3     7.8353     1.7328   4.522  0.00627 **
length   0.5685     0.2103   2.703  0.04264 *
[ ... ]
Residual standard error: 0.5913 on 5 degrees of freedom
Multiple R-squared:  0.9987,    Adjusted R-squared:  0.9976
F-statistic: 928.3 on 4 and 5 DF,  p-value: 2.319e-07
```

This is exactly the same model as before: the parameter estimates are either the same (for pop1, length and the residual standard error, the latter being the square root of the variance in the equations) or they can be obtained by adding two of the parameters in the former parameterisation. That is, the value of pop2 here is obtained by adding the value of the intercept and of pop2 in the previous parameterisation (i.e., 6.4801 + 1.5079 = 7.988) and that of pop3 here by adding the value of the intercept and of pop3 (i.e., 6.4801 + 1.3552 = 7.8353).

Different algebraic ways to write this model are the following:

(1) $wing_i \sim Normal(\alpha_j + \beta * length_i, \sigma^2)$

(2) $wing_i \sim Normal(\mu_i, \sigma^2)$, with $\mu_i = \alpha_j + \beta * length_i$

It is very important that you know how to write your linear models in algebra, because we will see below how a model written in the BUGS language very much resembles its representation in algebra, so much so, indeed, that once you have written it in algebra, you have almost written it in BUGS ! Furthermore, BUGS software can be sensitive to the parameterisation chosen, one may work well and the other not, hence, it is a good thing to be able to switch between different parameterisations of a model and try out which works best.

We have mentioned above that the intercept is the expected wing span of a dragonfly of length zero. This is really nonsensical, and a much more sensible model is obtained when we regress wing span not on body length, but on *centered* body length, i.e., a transformation of original body length obtained by subtracting mean body length. In this way, the intercept is the expected wing span of a dragonfly of the average observed body length, which is much more meaningful parameter. Moreover, centering covariates can be required for numerical reasons both for maximum likelihood fits and for Bayesian MCMC analyses. Sometimes, we even have to scale them, i.e., center them and then divide the result by some amount, for instance, the covariate standard deviation.

In statistics books, you are likely to see all of linear models described in matrix algebra, for instance as $\mathbf{y} = \mathbf{X}\beta + \varepsilon$. What does this mean ? Of course, matrices (and vectors) are simply a manner in which numbers are stored in an orderly way in an array. Hence, here $\mathbf{y}$ is a vector of length nine containing the responses, i.e., the wing span of each of the nine dragonflies. $\mathbf{X}\beta$ is the design matrix (also called model or X matrix) of the fitted model, $\mathbf{X}$, which is matrix-multiplied by the parameter vector β, and ε is another vector containing the residuals. To understand what this means, it is useful to write this out with the numbers of our data set plugged in. We will do this first for the model in the effects parameterisation and then for the means parameterisation. You will

see that the structure of the linear model is exactly defined by the design matrix. Moreover, the structure of the design matrix determines the interpretation of the parameters fitted.

So for the effects parameterisation, we have

$$\begin{pmatrix} 10.82 \\ 10.56 \\ 11.84 \\ 12.09 \\ 11.74 \\ 13.51 \\ 11.39 \\ 13.04 \\ 12.85 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 6.77 \\ 1 & 0 & 0 & 8.27 \\ 1 & 0 & 0 & 9.20 \\ 1 & 1 & 0 & 6.88 \\ 1 & 1 & 0 & 7.69 \\ 1 & 1 & 0 & 8.96 \\ 1 & 0 & 1 & 6.88 \\ 1 & 0 & 1 & 8.15 \\ 1 & 0 & 1 & 9.20 \end{pmatrix} \times \begin{pmatrix} \mu \\ \alpha_2 \\ \alpha_3 \\ \beta \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \\ \varepsilon_7 \\ \varepsilon_8 \\ \varepsilon_9 \end{pmatrix}, \text{ with } \varepsilon_i \sim Normal(0, \sigma^2)$$

From left to right, we have response vector, design matrix, parameter vector and residual vector. The design matrix contains indicator variables for the effects of factors and the measured covariate values for continuous explanatory variables. The design matrix (also called model matrix or X matrix) multiplied with the parameter vector produces another vector, μ_i above, which is the expected wing span for each dragonfly. For dragonfly 1, it is given by $1*\mu + 0*\alpha_2 + 0*\alpha_3 + 6.77*\beta$, while for dragonfly 9, it is $1*\mu + 0*\alpha_2 + 1*\alpha_3 + 9.20*\beta$. With a fitted model, we can obtain these expected wing lengths "by hand" as follows:

```
model.matrix(~pop+length) %*% fm$coef       # %*% denotes matrix multiplication
      [,1]
1 10.32863
2 11.18134
3 11.71002
4 11.89904
5 12.35950
6 13.08146
7 11.74639
8 12.46835
9 13.06525
```

We can get the same using the `predict` function.

```
predict(fm)
       1        2        3        4        5
10.32863 11.18134 11.71002 11.89904 12.35950
       6        7        8        9
13.08146 11.74639 12.46835 13.06525
```

For the means parameterisation of the model, we have

$$\begin{pmatrix} 10.82 \\ 10.56 \\ 11.84 \\ 12.09 \\ 11.74 \\ 13.51 \\ 11.39 \\ 13.04 \\ 12.85 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 6.77 \\ 1 & 0 & 0 & 8.27 \\ 1 & 0 & 0 & 9.20 \\ 0 & 1 & 0 & 6.88 \\ 0 & 1 & 0 & 7.69 \\ 0 & 1 & 0 & 8.96 \\ 0 & 0 & 1 & 6.88 \\ 0 & 0 & 1 & 8.15 \\ 0 & 0 & 1 & 9.20 \end{pmatrix} \times \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \beta \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \varepsilon_4 \\ \varepsilon_5 \\ \varepsilon_6 \\ \varepsilon_7 \\ \varepsilon_8 \\ \varepsilon_9 \end{pmatrix}, \text{ with } \varepsilon_i \sim Normal(0, \sigma^2)$$

Now, the expected wing span of dragonfly 1 is still given by $1*\alpha_1 + 0*\alpha_2 + 0*\alpha_3 + 6.77*\beta$, which is the same as before, except for the small semantic change that we have now called the intercept in population 1 α_1 instead of μ. In contrast, for dragonfly 9 we get $0*\alpha_1 + 0*\alpha_2 + 1*\alpha_3 + 9.20*\beta$ and so we see that the intercept of the regression line in population 3 is now given by α_3, rather than by $\mu + \alpha_3$ as in the effects parameterisation of the model.

As a side-comment: the least-squares model fitting criterion chooses the values of the parameter vector such that the sum of the squared residuals becomes minimal (thence the name least squares). It does not make any distributional assumptions about the residuals. However, the parameter estimates are numerically equivalent to the more general maximum likelihood criterion.

The design matrix lies at the heart of a linear model and it is imperative that we obtain an understanding of it when fitting linear models. In **R**, there is the useful `model.matrix` function, which allows to print the design matrix for any linear model specified in **R**'s model definition language. This can be very useful to understand a model or can even be useful when fitting it in BUGS, for instance, by clarifying to us how a model looks like. Let's look at the design matrices of the two parameterisations of the main-effects ANCOVA model.

```
# Effects or treatment contrast parameterisation
model.matrix(~ pop + length)
  (Intercept) pop2 pop3 length
1           1    0    0   6.77
2           1    0    0   8.27
3           1    0    0   9.20
4           1    1    0   6.88
5           1    1    0   7.69
6           1    1    0   8.96
7           1    0    1   6.88
8           1    0    1   8.15
9           1    0    1   9.20

# Means parameterization
model.matrix(~ pop-1 + length)
  pop1 pop2 pop3 length
1    1    0    0   6.77
2    1    0    0   8.27
3    1    0    0   9.20
4    0    1    0   6.88
5    0    1    0   7.69
```

```
6    0    1    0   8.96
7    0    0    1   6.88
8    0    0    1   8.15
9    0    0    1   9.20
```

The form of the design matrix determines the interpretation of the parameters. Table 1 gives an overview of different linear models that can be fitted for two covariates, one of which is a factor. Importantly, we also discuss the case of an interaction between `pop` and `length`, which is represented graphically by a bundle of non-parallel regression lines, and for which we here give the design matrices.

```
model.matrix(~ pop*length)  # effects parameterisation
  (Intercept) pop2 pop3 length pop2:length pop3:length
1           1    0    0   6.77        0.00        0.00
2           1    0    0   8.27        0.00        0.00
3           1    0    0   9.20        0.00        0.00
4           1    1    0   6.88        6.88        0.00
5           1    1    0   7.69        7.69        0.00
6           1    1    0   8.96        8.96        0.00
7           1    0    1   6.88        0.00        6.88
8           1    0    1   8.15        0.00        8.15
9           1    0    1   9.20        0.00        9.20

model.matrix(~ pop*length-1-length)  # means parameterisation
  pop1 pop2 pop3 pop1:length pop2:length pop3:length
1    1    0    0        6.77        0.00        0.00
2    1    0    0        8.27        0.00        0.00
3    1    0    0        9.20        0.00        0.00
4    0    1    0        0.00        6.88        0.00
5    0    1    0        0.00        7.69        0.00
6    0    1    0        0.00        8.96        0.00
7    0    0    1        0.00        0.00        6.88
8    0    0    1        0.00        0.00        8.15
9    0    0    1        0.00        0.00        9.20
```

Note in Table 1 the similarity between the BUGS code and the algebraic description of a model. Thus, when you know how to write a model in algebra, you are almost there at writing it in the BUGS language. Nevertheless, here we won't go into more detail in terms of the BUGS language, because we will see so many examples later and especially, in the next chapter/module.

As a final topic in linear models, we will briefly look at how effects of two factors are combined. For one factor and one continuous covariate, as just shown, this is reasonably simple: the columns in the design matrix are simply combined. However, for two factors, we typically have to resolve some aliasing when fitting two factors, that is, we have to drop some columns in the design matrix, which correspond to redundant parameters in the parameter vector (i.e., parameters that cannot be estimated). This can be somewhat difficult to understand at first, so when fitting models with two or more factors in BUGS, it can be very helpful to inspect their design matrix in **R**, using `model.matrix`, where we can simply fit such models by specifying them as A+B or A*B. Let's do this for the two factors in our analysis, `pop` (with three levels) and `sex` (with two). We will fit the models which used to be called "two-way ANOVA with main effects" and "two-way ANOVA with interaction effects".

```
model.matrix(~ pop+sex)  # main-effects 2-way ANOVA
```

```
  (Intercept) pop2 pop3 sex2
1           1    0    0    0
2           1    0    0    1
3           1    0    0    0
4           1    1    0    1
5           1    1    0    0
6           1    1    0    1
7           1    0    1    0
8           1    0    1    1
9           1    0    1    0
```

The meaning of the parameters associated with the first through the last column of the design matrix are the following: the intercept is the expected wing span of a dragonfly with level 1 for both factors, that is, a male in population 1. The other three parameters are the difference for a male in population 2 and 3, respectively, and for a female.

```
model.matrix(~ pop+sex-1)  # main-effects 2-way ANOVA
  pop1 pop2 pop3 sex2
1    1    0    0    0
2    1    0    0    1
3    1    0    0    0
4    0    1    0    1
5    0    1    0    0
6    0    1    0    1
7    0    0    1    0
8    0    0    1    1
9    0    0    1    0
```

Here, the parameters associated with the first three columns in the design matrix directly correspond to the expected wing span of a male in populations 1, 2 and 3, respectively, and the last parameter is the difference between females and males.

Next, let's look at the design matrix of an interactive model. The following three design matrices are all different parameterisations of an interaction-effects 2-way ANOVA, which estimate a separate expected wing span for the six groups corresponding to the pairwise combination of all levels of the two factors pop and sex. Perhaps the last one is the easiest to understand in terms of the meaning of the parameters.

```
model.matrix(~ pop*sex)  # interaction-effects 2-way ANOVA
  (Intercept) pop2 pop3 sex2 pop2:sex2 pop3:sex2
1           1    0    0    0         0         0
2           1    0    0    1         0         0
3           1    0    0    0         0         0
4           1    1    0    1         1         0
5           1    1    0    0         0         0
6           1    1    0    1         1         0
7           1    0    1    0         0         0
8           1    0    1    1         0         1
9           1    0    1    0         0         0
```

```
model.matrix(~ pop*sex-1)  # interaction-effects 2-way ANOVA
  pop1 pop2 pop3 sex2 pop2:sex2 pop3:sex2
1    1    0    0    0         0         0
2    1    0    0    1         0         0
3    1    0    0    0         0         0
```

```
4    0    1    0    1         1         0
5    0    1    0    0         0         0
6    0    1    0    1         1         0
7    0    0    1    0         0         0
8    0    0    1    1         0         1
9    0    0    1    0         0         0

model.matrix(~ pop:sex-1)  # interaction-effects 2-way ANOVA
  pop1:sex1 pop2:sex1 pop3:sex1 pop1:sex2 pop2:sex2 pop3:sex2
1         1         0         0         0         0         0
2         0         0         0         1         0         0
3         1         0         0         0         0         0
4         0         0         0         0         1         0
5         0         1         0         0         0         0
6         0         0         0         0         1         0
7         0         0         1         0         0         0
8         0         0         0         0         0         1
9         0         0         1         0         0         0
```

Table 1: Syntax for specifying different linear models in the R and BUGS model definition languages, when `pop` is a factor (i.e., a categorical explanatory variable) with three levels (indexed by *j*) and `length` is a covariate (i.e., a continuous explanatory variable). The factor levels of `pop` can be numbers or letters (or names) in R, but they must be alphanumeric with the first level a 1 in the BUGS language. Individuals are indexed by *i* in BUGS (i.e., *i* = 1...9). Models 6 and 7 can also be reparameterised similar to the move from model 2 to 3, but this is not shown here. The full number of parameters in the fitted model is one higher, because it includes the variance also.

No.	Model in R	Model in algebra	Model in BUGS	Traditional name of technique based on that linear model	Number of parameters	Meaning
1	`1`	α	`alpha`	"Model of the mean"	1	Constant term (intercept) only
2	`pop`	α_j	`alpha[pop[i]]`	One-way ANOVA	3	Three constants, one for each population (called the level of the factor). Called a t-test if factor has only two levels.
3	`pop-1`	$\mu+\alpha_j$	`mu + alpha[pop[i]]`	One-way ANOVA	3	"Subtract the intercept"; this is a mere reparameterisation of model 2. with an intercept (= the value for the first population) and two constants that are the differences between the values of population 2 and 1 and 3 and 1. In BUGS, the first level of the vector alpha must be manually set to zero to avoid overparameterisation. In R this is done automatically.
4	`length`	$\alpha+\beta*x_i$	`alpha + beta * length[i]`	Simple linear regression	2	An intercept plus a slope, common to all three populations (i.e., no effect of pop)
5	`length-1`	$\beta*x_i$	`beta * length[i]`	Simple linear regression through the origin	1	"Subtract the intercept": this is NOT a mere reparameterisation of model 4. Regression through the origin; not usually a meaningful model.
6	`pop+length`	$\alpha_j+\beta*x_i$	`alpha[pop[i]] + beta * length[i]`	Main-effects ANCOVA	4	One separate intercept for each population and a common slope
7	`pop*length`	$\alpha_j+\beta_j*x_i$	`alpha[pop[i]] + beta[pop[i]] * length[i]`	Interaction-effects ANCOVA	6	Three separate intercepts and three separate slopes. That is, fully separate regression of wing on length for each population.

3.2 Generalised linear models

Generalised linear models (GLMs) carry over the concept of a linear effect of covariates to response variables that are not necessarily (but can be) normal. In GLMs, the linear effect of covariates is applied not to the expected response directly, but to a transformation of the expected response. That transformation is called the link function. Generally, we can write this as follows, by describing a GLM in terms of three things. For response y_i, we can write:

1. *Random part of the response*: a statistical distribution f with mean μ_i

$$y_i \sim f(\mu_i)$$

2. *A link function* g, which is applied to the mean μ_i:

$$g(\mu_i) = \eta_i$$

3. *Systematic part of the response*: called the linear predictor (η_i), e.g.

$$\eta_i = \alpha + \beta * x_i$$

By combining elements 2 and 3, we can define a GLM succinctly in just two lines:

$$y_i \sim f(\mu_i)$$

$$g(\mu_i) = \alpha + \beta * x_i$$

This is exactly the way in which GLMs are specified in the BUGS language and this is the reason why BUGS is so great if you want to really understand GLMs. The GLM concept gives you considerable creative freedom to combine the three components of a GLM, but there are typically pairs of response distributions and link functions that go together particularly well. These link functions are called canonical link functions and are the identity link for normal responses ($\eta_i = \mu_i$), the log link for Poisson responses ($\eta_i = \log(\mu_i)$) and the logit link for binomial responses ($\eta_i = \log(\mu_i)/\log(1-\mu_i)$). Together, these three standard GLMs make up a vast number of statistical methods used in population ecology and elsewhere; for an overview, see Kéry (2010).

The broad scope of the GLM is one reason for the great importance of the GLM for you. The other one, which we will see many times later in the book, is that the GLM represents the main building block for hierarchical models. Many of the most exciting ecological models for inference about populations can be viewed simply as a sequence of coupled GLMs (Royle and Dorazio, 2008; Kéry and Schaub, 2012).

Poisson GLM

For an illustration of a Poisson GLM, we can treat the integer part of the wing span variable as a count, for instance, imagining that it represented the number of some parasite counted on each dragonfly, and then use our toy data set to fit a Poisson GLM. Writing the truncated wing span variable of individual *i* as C_i, we can then define the "main-effects ANCOVA" linear model within the Poisson GLM as follows.

1. Random part of the response (statistical distribution):

$$C_i \sim \text{Poisson}(\lambda_i)$$

2. Link of random and systematic part (log link function):

$$\log(\lambda_i) = \eta_i$$

3. Systematic part of the response (linear predictor η_i):

$$\eta_i = \alpha_j + \beta * length_i$$

Here, λ_i is the expected count (the mean response) for dragonfly i on the arithmetic scale, η_i is the expected count on the log link scale (i.e., the linear predictor), $length_i$ is the length of dragonfly i and α and β are the two parameters (α being vector-value) of the log-linear regression of the counts on the factor `pop` and the covariate `length`.

We fit this model in **R**. In the next chapter/module, we will see how to fit Poisson and other GLMs in the BUGS language. Essentially, we will see once again how writing the model in algebra will almost directly lead to BUGS code for the same model.

$$C_i \sim \text{Poisson}(\lambda_i)$$
$$\log(\lambda_i) = \alpha_j + \beta * length_i$$

```
count <- trunc(wing)             # Keeps only the integer part of a number
summary(fm <- glm(count ~ pop-1 + length, family = poisson))
Call:
glm(formula = count ~ pop - 1 + length, family = poisson)

[ ... ]

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
pop1    2.00479    0.87703   2.286   0.0223 *
pop2    2.16420    0.84882   2.550   0.0108 *
pop3    2.15454    0.87371   2.466   0.0137 *
length  0.04081    0.10571   0.386   0.6994
[ ... ]
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 315.0174  on 9  degrees of freedom
Residual deviance:   0.2483  on 5  degrees of freedom
AIC: 46.82

Number of Fisher Scoring iterations: 3
```

Bernoulli GLM (= logistic regression for binary response)

We use a Bernoulli distribution to describe patterns in coin-flip like processes or the outcomes thereof. The Bernoulli is also important because the perhaps more commonly encountered Binomial distribution with trial size *N* is the sum of *N* Bernoulli outcomes. For our

example data set, we can model the event that the parasite load is 12 or greater as a so-called "success" (coded as 1) an the converse as a "failure" (coded as 0). We call `high.load` the indicator for a parasite load greater than 12; it contains a 1 if a dragonfly has more than 12 parasites and a 0 if it has between 0 and 12 parasites. Our Bernoulli GLM for the occurrence of a high parasite load in relationship to population and body length is then:

$$high.load_i \sim \text{Bernoulli}(p_i)$$

$$\text{logit}(p_i) = \log\left(\frac{p_i}{1-p_i}\right) = \alpha_j + \beta * length_i$$

Here, p_i is the expected proportion of dragonflies with high load on the probability scale. This is the mean response at the level of each individual dragonfly. The linear model is applied to the logit transformation of that expected response.

```
high.load <- wing>12
summary(fm <- glm(high.load ~ pop-1 + length, family = binomial))
Call:
glm(formula = high.load ~ pop - 1 + length, family = binomial)

[...]

Coefficients:
       Estimate Std. Error z value Pr(>|z|)
pop1    -33.074   5443.139  -0.006    0.995
pop2    -11.471     10.550  -1.087    0.277
pop3    -11.822     10.879  -1.087    0.277
length    1.592      1.403   1.135    0.257

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 12.4766  on 9  degrees of freedom
Residual deviance:  5.7571  on 5  degrees of freedom
AIC: 13.757

Number of Fisher Scoring iterations: 18
```

Binomial GLM (= logistic regression)

The Binomial distribution is customarily adopted as a model for random variables that can be considered as a sum of *N* independent Bernoulli trials, where *N* is called the sample or trial size, or the binomial denominator. We will see Binomial random variables many times throughout this book (for instance, in the next chapter), hence, here we are not going to spend more time on them, or on how they are fitted using R or BUGS.

3.3 Random effects

Any random effects model is already a hierarchical model, so there is some overlap between the three concepts in this chapter. As we have seen, a hierarchical model is a nested sequence of two or more random variables, of which one is typically observed and the other(s) unobserved or at

least only partly observed. We call random effects the realisations of those "higher-level", un- or only partly observed random variables. At the data level, that is, for the observed random variable, we may adopt a Normal distribution or any other distribution for a GLM, e.g., a Poisson or a Bernoulli or Binomial. We illustrate this here for a Normal model and afterwards for a Poisson GLM. For the unobserved random variable, that is, for the random effects, in the vast majority of cases a normal distribution is assumed, but not always. For instance, in occupancy models, we have binary random effects which are given a Bernoulli distribution and in N-mixture models, discrete random effects assumed to follow a Poisson or similar distribution.

Random effects for a normal data distribution: Normal GLMM

The distribution assumed for the observed random variable, that is, for the data, is often called the data distribution. We will use our toy dragonfly example to illustrate a random effects version of the linear regression model and, especially, the topic of shrinkage. After that, we do the same for the Poisson version of the model.

Here, we revisit the ANCOVA example from section 3.1 and (re-create) the figure from before, but now without keeping track of the sex of the nine individuals. This is a Normal-normal mixed or hierarchical model.

```
# Plot data without distinguishing sex
plot(length, wing, col = rep(c("red", "blue", "green"), each = 3), xlim = c(6.5,
9.5), ylim = c(10.5, 13.5), cex = 1.5, lwd = 2, frame.plot = FALSE, las = 1, pch
= 16, xlab = "Body length", ylab = "Wing span")
```

The model adopted in section 3.1. assumed a linear relationship between wing span and length with a different baseline in each population and with residuals ε_i coming from a zero-mean normal distribution with variance σ^2.

$$mass_i = \alpha_j + \beta * length_i + \varepsilon_i, \text{ with } \varepsilon_i \sim Normal(0, \sigma^2)$$

In this model, the population effects (the α_j) are estimated as completely unrelated numbers. If we assume that they have something in common, are "exchangeable", that is, similar but not identical, we can make the assumption that they are draws from a common distribution with (hyper-) parameters to be estimated. That is, we can add to the above model the following assumption:

$$\alpha_j \sim Normal(\mu_\alpha, \sigma_\alpha^2)$$

This line is the *only* difference between a fixed-effects version of the "ANCOVA" or clustered regression model and a random-effects version of it. We use **R** to fit both models and plot the resulting regression lines (Fig. 3-1). For the random-effects model we use the REML method, a variant of maximum likelihood that is better suited for mixed models, and fit the model with the function `lmer` in the R package lme4, which we need to load first.

```
# Fit fixed-effects model, print regression parameter estimates and plot
regression lines
summary(lm <- lm(wing ~ pop-1 + length))
abline(lm$coef[1], lm$coef[4], col = "red", lwd = 3, lty = 2)
```

```
abline(lm$coef[2], lm$coef[4], col = "blue", lwd = 3, lty = 2)
abline(lm$coef[3], lm$coef[4], col = "green", lwd = 3, lty = 2)

# Fit mixed model, print random effects and plot regression lines
library(lme4)
summary(lmm <- lmer(wing ~ length + (1|pop)))
ranef(lmm)
abline((lmm@fixef[1]+ranef(lmm)$pop)[1,], lmm@fixef[2], col = "red", lwd = 3)
abline((lmm@fixef[1]+ranef(lmm)$pop)[2,], lmm@fixef[2], col = "blue", lwd = 3)
abline((lmm@fixef[1]+ranef(lmm)$pop)[3,], lmm@fixef[2], col = "green", lwd = 3)
abline(lmm@fixef[1], lmm@fixef[2], col = "black", lwd = 5, lty = "dashed")
```

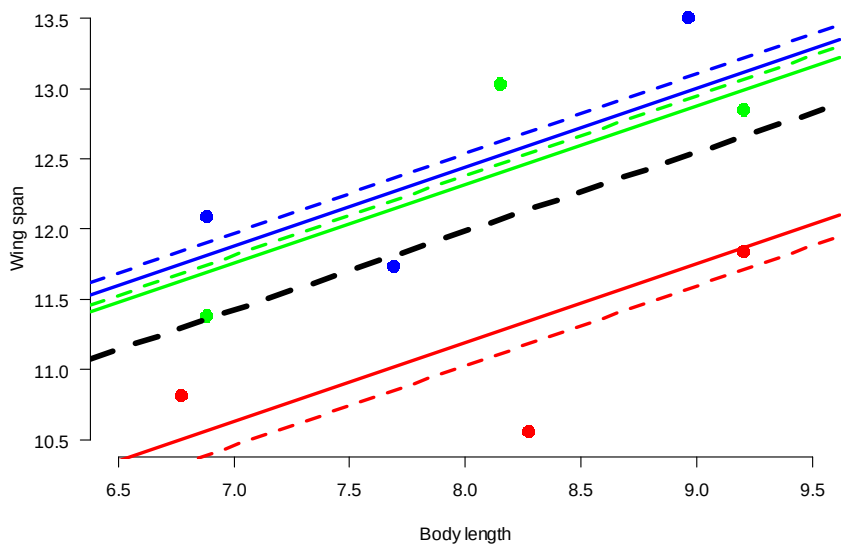

Fig. 3: An illustration of the shrinkage of random-effects parameters towards their common mean, i.e., the mean hyperparameter of their prior distribution. The dashed lines are the least-squares lines of best fit, which are equivalent to the line of best fit under a fixed-effects ANCOVA linear model (i.e., the population-specific intercepts α_j are estimated as unrelated parameters). The solid lines are the corresponding random-effects estimates; that is, the population-specific intercepts are estimated subject to the condition $\alpha_j \sim Normal(\mu_\alpha, \sigma_\alpha^2)$. The random-effects lines are pulled in towards the grand mean, μ_α, which is represented by the black dashed line. Colors blue, green and red code for the three populations.

Here are the parameter estimates under this model. The estimates of the mean and the variance of the random population effects are 7.5023 and 0.57208. We see that the random-effects regression lines are less extreme than the fixed-effects regression lines. This is shrinkage in action.

When parameters are estimated as random rather than fixed effects, they are no longer estimated as fully independent numbers, rather, their estimates are correlated somewhat. The reason for that is that they are assumed to be draws from a common distribution with (hyper)parameters that are estimated. That is, the random- and the fixed-effects estimates of a model that is otherwise the same, are not identical ! Random-effects estimates are always less extreme than are fixed-effects estimates. They are pulled in towards their mean (i.e., the mean

hyperparameter of the distribution assumed for the random effects). In a Normal-Normal random effects model such as here, the degree of shrinkage, or pulling in towards the grand mean, depends on the ratio between the population variance σ_α^2 and the residual variance σ^2. If the population variance σ_α^2 is relatively large, not much shrinkage happens, while if the residual variance σ^2 is relatively large, the degree of shrinkage is large. Fixed-effects estimates can be describes as random-effects estimates with infinite population variance σ_α^2.

```
summary(lmm <- lmer(wing ~ length + (1|pop)))
Linear mixed model fit by REML
Formula: wing ~ length + (1 | pop)
   AIC   BIC logLik deviance REMLdev
 28.33 29.12 -10.16    19.15   20.33
Random effects:
 Groups   Name        Variance Std.Dev.
 pop      (Intercept) 0.57208  0.75636
 Residual             0.34963  0.59130
Number of obs: 9, groups: pop, 3

Fixed effects:
            Estimate Std. Error t value
(Intercept)   7.5023     1.7475   4.293
length        0.5600     0.2101   2.666

Correlation of Fixed Effects:
       (Intr)
length -0.962
```

We could also treat the slopes as realisations from a random variable. This would result in the following model (note that now the slope β is indexed by *j*, that is, it can vary by population).

$$mass_i = \alpha_j + \beta_j * length_i + \varepsilon_i$$

$\alpha_j \sim Normal(\mu_\alpha, \sigma_\alpha^2)$ # Intercepts defined as random effects

$\beta_j \sim Normal(\mu_\beta, \sigma_\beta^2)$ # Slopes defined as random effects

$\varepsilon_i \sim Normal(0, \sigma^2)$ # same old residual "random effects"

Another way of describing this model is that we fit three separate normal distributions, of which one has the mean fixed at zero. We can fit this model in **R** as follows. (Note that the strange notation specifies a model with random intercepts and random slopes, but without a correlation parameter). For our minute data set, this complex model does not make sense, but we fit it here simply for illustration.

```
summary(lmm <- lmer(wing ~ length + (1|pop) + (0+length|pop)))
Linear mixed model fit by REML
Formula: wing ~ length + (1 | pop) + (0 + length | pop)
  AIC   BIC logLik deviance REMLdev
 29.9 30.89  -9.95    18.66    19.9
Random effects:
 Groups   Name        Variance   Std.Dev.
 pop      (Intercept) 1.6756e-14 1.2944e-07
```

```
 pop      length      9.3751e-03 9.6825e-02
 Residual             3.1655e-01 5.6263e-01
Number of obs: 9, groups: pop, 3

Fixed effects:
            Estimate Std. Error t value
(Intercept)   7.4317     1.6122    4.61
length        0.5694     0.2078    2.74

Correlation of Fixed Effects:
       (Intr)
length -0.957
```

One consequence of the sharing of information among random effects with the same prior distribution is that the aliasing observe in some fixed-effects models is resolved; that is, all parameters can be estimated. For instance, when fitting two factors as fixed effects such as sex and pop here, not all effects of all factor levels can be estimated, but rather, some have to be set to 0 to avoid an overparameterised model. This is no longer the case when these effects are treated as random. This is shown in the following random-effects, main-effects two-way ANOVA.

```
summary(lmm <- lmer(wing ~ (1|pop) + (1|sex)))
> summary(lmm <- lmer(wing ~ (1|pop) + (1|sex)))
Linear mixed model fit by REML
Formula: wing ~ (1 | pop) + (1 | sex)
   AIC   BIC logLik deviance REMLdev
 32.15 32.93 -12.07    24.35   24.15
Random effects:
 Groups   Name        Variance Std.Dev.
 pop      (Intercept) 0.38061  0.61694
 sex      (Intercept) 0.00000  0.00000
 Residual             0.71713  0.84684
Number of obs: 9, groups: pop, 3; sex, 2

Fixed effects:
            Estimate Std. Error t value
(Intercept)  11.9822     0.4545   26.36
```

We can obtain estimates of the random effects using the `ranef` function. Uncertainty intervals are not straightforward to get using a non-bayesian approach. Perhaps with the boostrap ?

```
ranef(lmm)
$pop
  (Intercept)
1  -0.5582674
2   0.2852760
3   0.2729914

$sex
  (Intercept)
1           0
2           0
```

OK, this is kind of pointless, because the sex effect is (correctly) estimated at zero, so we can't really see that in principle, they could be estimated when treated random.

Random effects for a Poisson data distribution: Poisson GLMM

Here we do the analogous thing for the Poisson version of our model, that is, we fit a Poisson generalised linear mixed model (GLMM). We only illustrate the random-intercepts model, which is this:

$$C_i \sim \text{Poisson}(\lambda_i)$$
$$\log(\lambda_i) = \alpha_j + \beta * length_i$$
$$\alpha_j \sim Normal(\mu_\alpha, \sigma_\alpha^2)$$

```
summary(lmm <- glmer(count ~ length + (1|pop), family = poisson))
Generalized linear mixed model fit by the Laplace approximation
Formula: count ~ length + (1 | pop)
   AIC   BIC  logLik deviance
 6.776 7.367 -0.3878   0.7755
Random effects:
 Groups Name        Variance Std.Dev.
 pop    (Intercept)  0        0
Number of obs: 9, groups: pop, 3

Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.15232    0.84536   2.546   0.0109 *
length       0.03558    0.10454   0.340   0.7336
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Correlation of Fixed Effects:
       (Intr)
length -0.993
```

We note that the model has not converged, or else, has had difficulty in estimating the variance of the random intercepts, because we obtain an estimate on the boundary of the parameter space, at 0. This is not surprising, since GLMMs with non-normal responses are more difficult to fit than normal GLMMs and especially, because we have only three groups (populations). If we created a version of our data set that was, say, 10 or 100 times larger, we would probably be able to get estimates under this model (you might want to try this as an exercise).

Summary

This completes our introduction to hierarchical models, which most typically are simply a combination of two or more GLMs; one for an observed random variable and the other(s) for unobserved random variable(s). Known structure in these GLMs can be accommodated with covariates or possibly with more complex modeling (e.g., distance-related variance-covariance matrices; this is not shown here, but see spatial modeling chapter). If you understand linear models, GLMs and random effects, you are in really good shape for much of the applied statistical analyses that are required for a modern ecologist. In this chapter, we have only looked at some of the simplest, traditional hierarchical models, or generalised linear mixed models (GLMM). They

could be called Normal-Normal mixed model or Normal-Poisson mixed model. That is, as in far more than 90% of the cases in applied statistical analysis in ecology, the distribution for the unobserved random variable, i.e., for the random effects, was assumed to be a normal. In most instances of HMs in this book, however, we will not have such traditional GLMMs, but rather will encounter non-standard GLMMs with random effects that have a more complicated structure, e.g., they will be assumed to come from a Bernoulli or Poisson distribution. This is perhaps surprising at first and may strike you as strange; but nevertheless, the principle of hierarchical modeling, that a complex model is built as a series of simpler models, typically GLMs, is retained also for these (more interesting ?) models.

Exercises

1. Fit random-effects two-way main-effects ANOVA sex + pop to a 10 times greater data set and see whether estimates are obtained by lme ?

```
# Define and plot data (10 times larger data set than the toy data set)
pop <- factor(rep(c(1, 1, 1, 2, 2, 2, 3, 3, 3), 10))
wing <- rep(c(10.82, 10.56, 11.84, 12.09, 11.74, 13.51, 11.39, 13.04, 12.85), 10)
length <- rep(c(6.77, 8.27, 9.20, 6.88, 7.69, 8.96, 6.88, 8.15, 9.20), 10)
sex <- factor(rep(c(1,2,1,2,1,2,1,2,1), 10))     # Assume males are 1, females 2
cbind(pop, wing, length, sex)   # Note that sex 1 = male and sex 2 = females

library(lme4)
summary(lmm <- lmer(wing ~ (1|pop) + (1|sex)))
ranef(lmm)
```

Yes, it seems we can.