CHAPTER

9

Normal One-Way ANOVA

OUTLINE

9.1 INTRODUCTION: FIXED AND RANDOM EFFECTS

Analysis of variance (ANOVA) is the generalization of a t-test to more than two groups. There are different kinds of ANOVA: one-way, with just a single factor, and two- or multiway, with two or more factors, and main- and interaction-effects models (see Chapter 10). Here, we present a one-way ANOVA and introduce the concept of random effects along the way. In random-effects models, a set of effects (e.g., group means) is constrained to come from some distribution, which is most often a normal, although it may be a Bernoulli (see Chapter 20), a Poisson (see Chapter 21) or yet another distribution. In this chapter, we will first generate and analyze a fixed-effects and then a random-effects ANOVA data set. In Chapters 12, 16, and 19–21, we will focus on mixed models, i.e., those containing both fixed and random effects. As a motivating example for this chapter, we assume that we measured snout–vent length

DOI: 10.1016/B978-0-12-378605-0.00009-0

FIGURE 9.1 Smooth snake (*Coronella austriaca*), France, 2006. (*Photo C. Berney*)

(SVL) in five populations of Smooth snakes (Fig. 9.1) and were interested in whether populations differ.

The one-way ANOVA can be parameterized in various ways (see Section 6.3.4). We adopt a means parameterization of the linear model for the fixed-effects, one-way ANOVA:

$$y_i = \alpha_{j(i)} + \varepsilon_i$$
$$\varepsilon_i \sim Normal\,(0,\sigma^2)$$

Here, y_i is the observed SVL of Smooth snake i in population j, $\alpha_{j(i)}$ is the expected SVL of a snake in population j, and residual ε_i is the random SVL deviation of snake i from its population mean $\alpha_{j(i)}$. It is assumed to be normally distributed around zero with constant variance σ^2.

Without any further assumption, the population means $\alpha_{j(i)}$ are simply some unknown constants that are estimated in a fixed-effects ANOVA. If, however, we add a distributional assumption about the population means $\alpha_{j(i)}$, we obtain a random-effects ANOVA:

$$y_i = \alpha_{j(i)} + \varepsilon_i$$
$$\varepsilon_i \sim Normal\,(0,\sigma^2)$$
$$\alpha_{j(i)} \sim Normal\,(\mu,\tau^2)$$

The interpretation of $\alpha_{j(i)}$ and ε_i as population mean SVL and residual, respectively, is unchanged. But now, the $\alpha_{j(i)}$ parameters are no longer assumed to be independent; rather, they come from a second normal

distribution with mean μ and variance τ^2. The latter are also called hyperparameters, because they are one level higher than the parameters $\alpha_{j(i)}$ that they govern.

Thus, the models for the fixed- and random-effects ANOVA differ only subtly; so how do we know when to apply which one in practice? Unfortunately, there are fairly differing views on this decision, see Gelman and Hill (2007, p. 245). The traditional view goes about as follows. When you have a particular interest in the studied factor levels and/or when you have included (nearly) all conceivable levels in a study, the associated factor should be viewed as having fixed effects. You estimate the effects of each level but are not interested in the variance among levels, except as part of an ANOVA table to construct an F-test statistic for that factor. Importantly, you cannot generalize to factor levels that you did not study. In contrast, you consider a factor as random when you don't have a particular interest in the levels that actually appear in your study and/or when these levels form a sample from a (much) larger set of possible levels that you *could* have included in your study. Typically, you want to generalize to this larger population and are more interested in the variation among the factor levels in that population, although you may still want to estimate the effects of the levels actually observed in your study. Thus, typical fixed-effects factors would be sex or cereal variety in an agricultural experiment. Typical random-effects factors might be time (e.g., year, month, or day) or location, such as experimental blocks or other spatial units on which repeated measurements are taken.

It has been argued (e.g., Robinson, 1991), that it doesn't make sense to claim that you studied only a sample of effects and then estimate them anyway, and that a more natural distinction between fixed and random effects is simply based on the question of whether they could plausibly have come from some *distribution* of effects. Such random effects are generated by the same homogeneous, stochastic process and statisticians also say they are *exchangeable*, which in common language means that they are similar, but not identical. This similarity is because of the common stochastic process that generated them and thus creates a stochastic relationship among the effects of the levels of a random-effects factor. In contrast, when factor levels are modeled as fixed they are considered unrelated or independent.

Why should one make distributional assumptions about a set of effects in a model, i.e., go from a fixed-effects ANOVA to the corresponding random-effects ANOVA? There are three reasons: extrapolation of inference to a wider population, improved accounting for system uncertainty, and efficiency of estimation. First, viewing the studied effects as a random sample from some population enables one to extrapolate to that population. This generalization can only be achieved by modeling the process that generates the realized values of

the random effects (i.e., by assuming a normal distribution for the $\alpha_{j(i)}$ above). Second, declaring factor effects as random acknowledges that when repeating our study, we obtain a different set of effects, so the resulting parameter estimates will differ from those in our current study. Random-effects modeling properly accounts for this added uncertainty in our inference about the analyzed system. Third, when making a random-effects assumption about a factor, these effects are no longer estimated independently; instead, estimates are influenced by each other and therefore are dependent. Specifically, individual estimates are "pulled in" toward the common mean μ, i.e., they are closer to μ than the corresponding fixed-effects estimates. This is why random-effects estimators are said to be "shrinkage estimators". Estimates that are more imprecise and are based on a smaller sample size are shrunk more. When effects are indeed exchangeable, shrinkage results in better estimates (e.g., with smaller prediction error) than the estimates obtained from a fixed-effects analysis. This is why one also says that a random-effects analysis "borrows strength".

Random-effects modeling can also be viewed as a compromise between assuming no effects and fully independent effects of the levels of a factor. When assuming a factor has no effect, you pool its effects, whereas when assuming it has fixed effects, you treat all effects as completely independent instead. When assuming a factor has random effects, you pool effects only partially, and the degree of pooling is based on the amount of information that you have about the effect of each level. According to this view, you might always want to assume all factors as random and let the data determine the degree of pooling; see Gelman (2005) and Gelman and Hill (2007).

Recently, statisticians seem to prefer the view of random-effects factors as those, whose effects result from a common stochastic process, with the resulting benefits of the ability to extrapolate, more honest accounting for uncertainty and shrinkage estimation. For instance, Sauer and Link (2002) assessed population trends in a large numbers of bird species and showed how imprecise estimates for species with little information borrowed strength from the "ensemble" (the group of species) and got pulled toward the group mean, and this yielded better predictions. Similarly, Welham et al. (2004) analyzed a huge wheat variety testing experiment and treated variety as random. Again, they found that this gave better predictions of future yield than treating variety as fixed.

Next, we generate one data set under a fixed-effects design and another under a random-effects design. We do this in a very "linear model" fashion, i.e., by first specifying a design matrix and choosing parameter values. For the fixed-effects analysis, we will arbitrarily select these values, whereas for the random-effects analysis, we will draw them from a normal distribution with specified hyperparameters. Then, we multiply the

design matrix by the parameter vector to obtain the linear predictor, to which we add residuals to obtain the actual measurements.

9.2 FIXED-EFFECTS ANOVA

9.2.1 Data Generation

We assume five groups with 10 snakes measured in each with SVL averages of 50, 40, 45, 55, and 60. This corresponds to a baseline population mean of 50 and effects of populations 2–5 of −10, −5, 5, and 10. We choose a residual standard deviation of SVL of 3 and assemble everything (Fig. 9.2). (Note that `%*%` denotes matrix multiplication in the R code below.)

```
ngroups <- 5                  # Number of populations
nsample <- 10                 # Number of snakes in each
pop.means <- c(50, 40, 45, 55, 60)   # Population mean SVL
sigma <- 3                    # Residual sd

n <- ngroups * nsample     # Total number of data points
eps <- rnorm(n, 0, sigma) # Residuals
x <- rep(1:5, rep(nsample, ngroups)) # Indicator for population
means <- rep(pop.means, rep(nsample, ngroups))
X <- as.matrix(model.matrix(~ as.factor(x) -1)) # Create design matrix
```

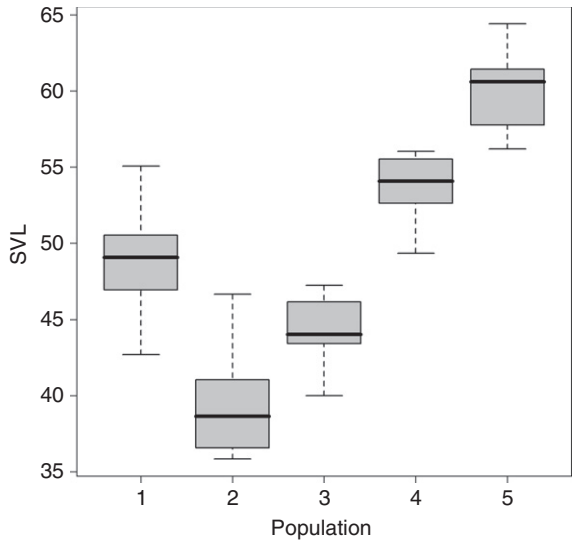

FIGURE 9.2 Snout–vent length (SVL) of Smooth snakes in five populations simulated under a fixed-effects model.

```
X                          # Inspect that
y <- as.numeric(X %*% as.matrix(pop.means) + eps)
     # assemble -- NOTE: as.numeric ESSENTIAL for WinBUGS
boxplot(y~x, col="grey", xlab="Population", ylab="SVL", main="", las = 1)
```

9.2.2 Maximum Likelihood Analysis Using R

```
print(anova(lm(y~as.factor(x))))
cat("\n")
print(summary(lm(y~as.factor(x)))$coeff, dig = 3)
cat("Sigma:    ", summary(lm(y~as.factor(x)))$sigma, "\n")
```

```
> print(summary(lm(y~as.factor(x)))$coeff, dig = 3)
              Estimate Std.  Error  t value   Pr(>|t|)
(Intercept)      48.84      0.864   56.51   1.96e-43
as.factor(x)2    -9.48      1.222   -7.75   7.89e-10
as.factor(x)3    -4.54      1.222   -3.71   5.67e-04
as.factor(x)4     4.76      1.222    3.90   3.20e-04
as.factor(x)5    11.25      1.222    9.20   6.56e-12
> cat("Sigma:         ", summary(lm(y~as.factor(x)))$sigma, "\n")
Sigma:            2.733362
```

Remembering that R fits an effects parameterization, we recognize fairly well the input values of the analysis.

9.2.3 Bayesian Analysis Using WinBUGS

We fit a means parameterization of the model and obtain effects estimates (i.e., population differences) as derived quantities. Note WinBUGS' elegant double-indexing `(alpha[x[i]])` to specify the expected SVL of snake *i* according to the *i*-th value of the population index x. We also add two lines to show how custom hypotheses can easily be tested as derived quantities. Test 1 examines whether snakes in populations 2 and 3 have the same size as those in populations 4 and 5. Test 2 checks whether the size difference between snakes in populations 5 and 1 is twice that between populations 4 and 1.

```
# Write model
sink("anova.txt")
cat("
model {

# Priors
  for (i in 1:5){          # Implicitly define alpha as a vector
     alpha[i] ~ dnorm(0, 0.001)
  }
  sigma ~ dunif(0, 100)
```

```
# Likelihood
  for (i in 1:50) {
     y[i] ~ dnorm(mean[i], tau)
     mean[i] <- alpha[x[i]]
  }

# Derived quantities
   tau <- 1 / ( sigma * sigma)
   effe2 <- alpha[2] - alpha[1]
   effe3 <- alpha[3] - alpha[1]
   effe4 <- alpha[4] - alpha[1]
   effe5 <- alpha[5] - alpha[1]

# Custom hypothesis test / Define your own contrasts
   test1 <- (effe2+effe3) - (effe4+effe5) # Equals zero when 2+3 = 4+5
   test2 <- effe5 - 2 * effe4 # Equals zero when effe5 = 2*effe4
}
",fill=TRUE)
sink()

# Bundle data
win.data <- list("y", "x")

# Inits function
inits <- function(){ list(alpha = rnorm(5, mean = mean(y)), sigma = rlnorm(1) )}

# Parameters to estimate
params <- c("alpha", "sigma", "effe2", "effe3", "effe4", "effe5", "test1", "test2")

# MCMC settings
ni <- 1200
nb <- 200
nt <- 2
nc <- 3

# Start Gibbs sampling
out <- bugs(win.data, inits, params, "anova.txt", n.thin=nt, n.chains=nc,
n.burnin=nb, n.iter=ni, debug = TRUE)

# Inspect estimates
print(out, dig = 3)
> print(out, dig = 3)
Inference for Bugs model at "anova.txt", fit using WinBUGS,
  3 chains, each with 1200 iterations (first 200 discarded), n.thin = 2
  n.sims = 1500 iterations saved
             mean     sd     2.5%      25%      50%      75%    97.5%   Rhat  n.eff
alpha[1]   48.837  0.877   47.049   48.240   48.840   49.420   50.636  1.011    180
alpha[2]   39.354  0.888   37.595   38.770   39.350   39.990   41.080  1.000   1500
```

```
alpha[3]   44.315  0.867   42.710   43.720   44.290   44.910   46.016  1.002   1500
alpha[4]   53.580  0.886   51.759   53.020   53.565   54.150   55.356  1.001   1500
alpha[5]   60.019  0.882   58.285   59.430   60.030   60.590   61.750  1.001   1500
sigma       2.811  0.303    2.296    2.594    2.777    3.015    3.451  1.002   1200
effe2      -9.482  1.273  -12.040  -10.320   -9.456   -8.665   -7.046  1.003    590
effe3      -4.521  1.275   -7.079   -5.356   -4.548   -3.677   -2.037  1.006    360
effe4       4.744  1.241    2.325    3.944    4.768    5.514    7.205  1.004    470
effe5      11.182  1.239    8.767   10.390   11.180   12.002   13.615  1.002    810
test1     -29.929  1.764  -33.425  -31.100  -29.900  -28.800  -26.415  1.000   1500
test2       1.694  2.158   -2.670    0.319    1.644    3.139    6.016  1.002    940
deviance  243.557  3.671  238.500  240.800  242.800  245.600  252.452  1.005    580
[ ... ]
DIC info (using the rule, pD = Dbar-Dhat)
pD = 5.8 and DIC = 249.3
DIC is an estimate of expected predictive error (lower deviance is better).
```

As an aside, the effective number of parameters, pD, is estimated quite correctly as we have five group means and one variance parameter. Comparison with the maximum likelihood solutions (see Section 9.2.2) shows how with vague priors, a Bayesian analysis yields numerically virtually identical inferences as a frequentist analysis. However, one of the most compelling things about a Bayesian analysis conducted using Markov chain Monte Carlo methods is the ease with which derived quantities can be estimated and custom tests conducted. In the above WinBUGS model code, we see how easily such custom contrasts (comparisons) can be estimated with full error propagation from all the involved random quantities. Of course, for a simple model such as a one-way ANOVA, this can also be done fairly easily in standard stats packages. However, in WinBUGS, this is *equally simple for any kind of parameter*, e.g., for variances (see Chapter 7), *and in any model type*, e.g., mixed models, generalized linear models (GLMs), or generalized linear mixed models (GLMMs).

9.3 RANDOM-EFFECTS ANOVA

9.3.1 Data Generation

For our second data set, we assume that population SVL means come from a normal distribution with selected hyperparameters. The code is only slightly different from the previous data-generating code. First, we choose the two sample sizes; the number of populations and that of snakes examined in each. As always, we generate balanced data for convenience only. The methods to "decompose" a data set in R and WinBUGS work just as well for unbalanced data.

```
npop <- 10              # Number of populations: now choose 10 rather than 5
nsample <- 12           # Number of snakes in each
n <- npop * nsample     # Total number of data points
```

We choose the hyperparameters of the normal distribution from which the random population means are thought to come from and use `rnorm()` to draw one realization from that distribution for each population. Then, we select the residual standard deviation and draw residuals.

```
pop.grand.mean <- 50      # Grand mean SVL
pop.sd <- 5               # sd of population effects about mean
pop.means <- rnorm(n = npop, mean = pop.grand.mean, sd = pop.sd)
sigma <- 3                # Residual sd
eps <- rnorm(n, 0, sigma)   # Draw residuals
```

We build the design matrix, expand the population effects to the chosen (larger) sample size, use matrix multiplication to assemble our data set and have a look at what we've created (Fig. 9.3):

```
x <- rep(1:npop, rep(nsample, npop))
X <- as.matrix(model.matrix(~ as.factor(x) -1))
y <- as.numeric(X %*% as.matrix(pop.means) + eps) # as.numeric is ESSENTIAL
boxplot(y ~ x, col = "grey", xlab = "Population", ylab = "SVL", main = "", las = 1)
                    # Plot of generated data
abline(h = pop.grand.mean)
```

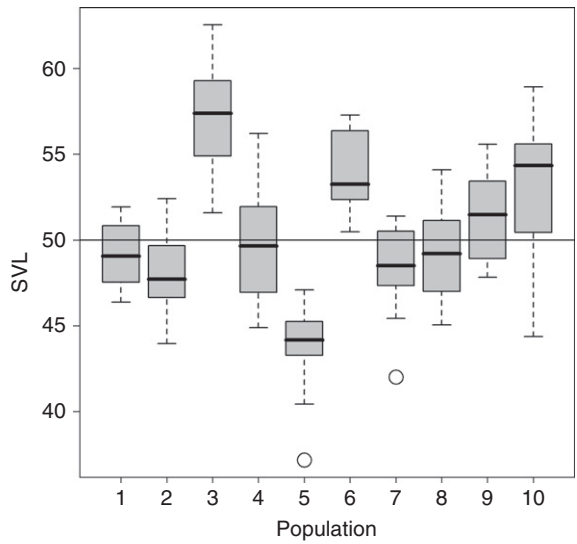

FIGURE 9.3 Simulated snout–vent length (SVL) of Smooth snakes in 10 populations. We can't tell from this graph that the population effects are random now rather than fixed as in Fig. 9.2. The horizontal line shows the grand mean.

9.3.2 Restricted Maximum Likelihood Analysis Using R

The functions contained in the R package lme4 allow one to fit, using approximate methods (Gelman and Hill, 2007), a wide range of linear, nonlinear, and generalized linear mixed models. We use this package for fitting a random-effects ANOVA by restricted maximum likelihood (REML). lme4 requires package Matrix, which we need to download and install in case we haven't done so already.

```
library('lme4')          # Load lme4

pop <- as.factor(x)      # Define x as a factor and call it pop

lme.fit <- lmer(y ~ 1 + 1 | pop, REML = TRUE)
lme.fit                  # Inspect results
ranef(lme.fit)           # Print random effects

> lme.fit# Inspect results
Linear mixed model fit by REML
Formula: y ~ 1 + 1 | pop
   AIC   BIC   logLik   deviance   REMLdev
 630.1 638.5   -312.1      626.3     624.1
Random effects:
   Groups    Name          Variance   Std.Dev.
   pop       (Intercept)   13.1244    3.6228
   Residual                8.5172     2.9184
Number of obs: 120, groups: pop, 10

Fixed effects:
            Estimate   Std. Error   t value
(Intercept) 50.318     1.176        42.78
> ranef(lme.fit)#   Print random effects
$pop
       (Intercept)
1       -1.0784560
2       -2.2585587
3        6.5857470
4       -0.5906953
5       -6.2970554
6        3.3997612
7       -1.9525175
8       -1.0407413
9        0.9989122
10       2.2336038
```

9.3.3 Bayesian Analysis Using WinBUGS

Now, WinBUGS. Note that adopting a common prior distribution for the 10 population means represents the random-effects assumption in this model.

```
sink("re.anova.txt")
cat("
model {

# Priors and some derived things
for (i in 1:npop){
    pop.mean[i] ~ dnorm(mu, tau.group) # Prior for population means
    effe[i] <- pop.mean[i] - mu # Population effects as derived quant's
  }
  mu ~ dnorm(0,0.001)         # Hyperprior for grand mean svl
  sigma.group ~ dunif(0, 10)  # Hyperprior for sd of population effects
  sigma.res ~ dunif(0, 10)    # Prior for residual sd

# Likelihood
  for (i in 1:n) {
    y[i] ~ dnorm(mean[i], tau.res)
    mean[i] <- pop.mean[x[i]]
  }

# Derived quantities
  tau.group <- 1 / (sigma.group * sigma.group)
  tau.res <- 1 / (sigma.res * sigma.res)
}
",fill=TRUE)
sink()

# Bundle data
win.data <- list(y=y, x=x, npop = npop, n = n)

# Inits function
inits <- function(){ list(mu = runif(1, 0, 100), sigma.group = rlnorm(1), sigma.res
= rlnorm(1) )}

# Params to estimate
parameters <- c("mu", "pop.mean", "effe", "sigma.group", "sigma.res")

# MCMC settings
ni <- 1200
nb <- 200
nt <- 2
nc <- 3
```

```
# Start WinBUGS
out <- bugs(win.data, inits, parameters, "re.anova.txt", n.thin=nt, n.chains=nc,
n.burnin=nb, n.iter=ni, debug = TRUE)

# Inspect estimates
print(out, dig = 3)
Inference for Bugs model at "re.anova.txt", fit using WinBUGS,
  3 chains, each with 1200 iterations (first 200 discarded), n.thin = 2
  n.sims = 1500 iterations saved
               mean    sd    2.5%     25%     50%     75%   97.5%  Rhat n.eff
mu           50.199 1.438  47.309  49.337  50.195  51.070  53.076 1.002  1500
pop.mean[1]  49.247 0.834  47.589  48.670  49.250  49.810  50.875 1.001  1500
pop.mean[2]  48.054 0.852  46.405  47.490  48.020  48.640  49.655 1.001  1500
pop.mean[3]  56.933 0.871  55.275  56.340  56.950  57.520  58.590 1.001  1500
pop.mean[4]  49.724 0.839  48.100  49.180  49.710  50.290  51.290 1.001  1500
pop.mean[5]  43.962 0.847  42.290  43.380  43.980  44.540  45.510 1.001  1500
pop.mean[6]  53.705 0.849  52.070  53.120  53.685  54.300  55.370 1.000  1500
pop.mean[7]  48.349 0.799  46.715  47.800  48.360  48.920  49.875 1.001  1500
pop.mean[8]  49.289 0.829  47.650  48.720  49.310  49.842  50.846 1.003  1000
pop.mean[9]  51.267 0.844  49.650  50.700  51.240  51.860  52.920 1.001  1500
pop.mean[10] 52.567 0.806  51.020  52.000  52.580  53.130  54.115 1.004   540
effe[1]      -0.952 1.589  -4.156  -1.990  -0.926   0.095   2.189 1.001  1500
effe[2]      -2.145 1.631  -5.513  -3.197  -2.126  -1.113   1.171 1.002  1200
effe[3]       6.734 1.639   3.558   5.680   6.781   7.736  10.055 1.001  1500
effe[4]      -0.475 1.617  -3.716  -1.461  -0.493   0.532   2.744 1.001  1500
effe[5]      -6.237 1.628  -9.426  -7.231  -6.255  -5.202  -3.169 1.001  1500
effe[6]       3.506 1.616   0.458   2.490   3.435   4.542   6.749 1.001  1500
effe[7]      -1.850 1.576  -5.136  -2.832  -1.858  -0.832   1.124 1.000  1500
effe[8]      -0.910 1.611  -4.105  -1.940  -0.898   0.074   2.352 1.003  1300
effe[9]       1.067 1.622  -1.975  -0.008   1.096   2.121   4.104 1.001  1500
effe[10]      2.368 1.582  -0.729   1.342   2.383   3.392   5.595 1.001  1500
sigma.group   4.287 1.254   2.546   3.380   4.036   4.941   7.305 1.002  1400
sigma.res     2.946 0.199   2.584   2.804   2.931   3.077   3.376 1.000  1500
deviance    598.756 5.010 591.347 595.200 597.900 601.600 610.210 1.005   590
[ ... ]
DIC info (using the rule, pD = Dbar-Dhat)
pD = 10.6 and DIC = 609.3
DIC is an estimate of expected predictive error (lower deviance is better).
```

On comparing the inference using REML in lme4 (see Section 9.3.2) with our Bayesian analysis, we find similar, but not identical, results of the two analyses. For instance, the among-population SVL variation (sigma.group), is estimated better in the Bayesian analysis (remember that truth is 5). The classical analysis, using `lmer()`, appears to be more biased and

does not give a standard error of the estimated random-effects standard deviation. This illustrates that `lmer()` may yield more biased estimates for small samples, i.e., when there are few levels only (Gelman and Hill, 2007). However, it has to be said that estimation of a between-group variance is always difficult when there are few levels and/or that variance is small (Lambert et al., 2005).

9.4 SUMMARY

We have introduced fixed (independent) and random (dependent) factor effects and fitted the corresponding one-way ANOVA models using WinBUGS and R, using lme4. We found that the WinBUGS solution is presumably less biased for the random-effects variance than the solution by lme4 when sample sizes are small.

EXERCISES

1. *Convert the ANOVA to an ANCOVA* (analysis of covariance): Within the fixed-effects ANOVA, add the effect on SVL of a continuous measure of habitat quality that varies by individual snake (perhaps individuals in better habitat are larger and more competitive). You may either recreate a new data set that contains such an effect or simply create a habitat covariate (e.g., by drawing random numbers) and add it as a covariate into the previous analysis.
2. *Watch shrinkage happen*: Population mean estimates under the random-effects ANOVA are shrunk toward the grand mean when compared with those under a fixed-effects model. First, watch this shrinkage by fitting a fixed-effects ANOVA to the random-effects data simulated in Section 9.3.1. Second, discard the data from 8 out of 10 snakes in one population and see what happens to the estimate of that population mean.
3. *Swiss hare data*: Compare mean observed population density among all surveyed sites when treating years as replicates. Do this once assuming that these populations corresponded to fixed effects; then repeat the analysis assuming they are random effects.