**Home page: http://www.unc.edu/courses/2010fall/ecol/563/001/index.html**

**http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm**

**Lecture 29—Wednesday, December 8, 2010**

**Topics**

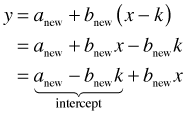
* [How centering removes the correlation between slopes and intercepts](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#how)
* [Multilevel models with three levels using the lme4 package](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#multilevel)
* [Analyzing repeated measures data with mixed effects models](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#repeated)
  + [Graphing the apple data](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#graphing)
  + [Constructing the basic level-1 model](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#constructing)
  + [Adding additional random coefficients](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#adding)
  + [Checking the adequacy of the correlation structure induced by the random effects](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#checking)
  + [Adding a correlation structure](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#structure)
* [Visualizing the final model](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#visualizing) [not done in class]
* [Cited references](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#cited)
* [R code](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#Rcode)

**How centering removes the correlation between slopes and intercepts**

We investigate how centering a predictor affects the correlation between a set of intercepts and slopes. Consider a generic ordinary linear regression equation.

old eqn

We elect to center the predictor with centering constant *k*.



The two equations represent the same linear relationship just parameterized differently. Comparing the two equations we see that

http://www.unc.edu/courses/2010fall/ecol/563/001/images/lectures/lecture29/compareparms.gif

The slope parameterizations are the same but the intercepts are parameterized differently. So, we can drop the subscript on *b* because *b* is the same in both equations. Solving for *a*new in the intercept equation we find the following.

http://www.unc.edu/courses/2010fall/ecol/563/001/images/lectures/lecture29/transformation.gif

So, the intercept in the centered model can be obtained by taking the intercept in the old model and adding to it the slope times the centering constant.

Fig. 1 is a scatter plot of the separate slopes and intercepts for all 74 invertebrate species from the log(PLD) versus log(temperature) model we considered last time. The ordinary least squares regression line is superimposed for reference. The lengths of the blue arrows indicate how much different intercepts will be affected by centering. The length of the arrow varies depending on the value of the slope with which that intercept is paired. Intercepts with bigger slopes will be shifted more in the plot. The overall effect is to impose a "shear" on the cloud of points with points at the bottom of the diagram being shifted more than points at the top.

inverts <- read.table( 'http://www.unc.edu/courses/2010fall/ecol/563/001/data/lectures/74species.csv',  
header=T, sep=',')

library(nlme)

out.lmlist <- lmList(log(PLD)~log(temp)|species, data=inverts)

plot(coef(out.lmlist)[,1], coef(out.lmlist)[,2], xlab='Intercept', ylab='Slope', xlim=c(-1.5,19))

abline(lm(coef(out.lmlist)[,2]~coef(out.lmlist)[,1]), lty=2, col=2)

arrows(3,-1,2,-1, code=2, angle=45, length=.08, lwd=2, col=4)

arrows(9,-3,4,-3, code=2, angle=45, length=.08, lwd=2, col=4)

arrows(6.5,-2,4,-2, code=2, angle=45, length=.08, lwd=2, col=4)

text(5.25,-2, expression(phantom()+ 2\*k), pos=3, col=4)

text(2,-1, expression(phantom()+ k), pos=2, col=4)

text(6.5,-3, expression(phantom()+ 3\*k), pos=3, col=4)

|  |
| --- |
| fig 1 |
| **Fig. 1**  Centering a predictor with a constant *k* will shift intercepts associated with bigger slopes more than intercepts associated with smaller slopes |

We can carry out an animation that shows the effect of centering on the relationship between slopes and intercepts with the animation package of R. We need to create a loop that generates the individual graphs and then include the Sys.sleep function to pause the display a specified amount of time. Fig. 2 displays the results.

library(animation)

for(k in c(1,1,1,1:35,35:1)) {

new.int <- coef(out.lmlist)[,1] + log(k)\*coef(out.lmlist)[,2]

r <- cor(new.int,coef(out.lmlist)[,2])

out <- lm(coef(out.lmlist)[,2]~new.int)

pval <- round(summary(out)$coefficients[2,4],3)

plot(new.int, coef(out.lmlist)[,2], xlab='Intercept', ylab='Slope', xlim=c(-1.5,13), ylim=c(-3.3,-.1))

abline(out, lty=2, col=2)

mtext(side=3, line=.5, substitute(list('k'==log(x), 'correlation' ==c, 'p'==p), list(x=k, c=round(r,3), p=pval)), cex=1, font=2)

Sys.sleep(1)

}

detach(package:animation)

|  |  |
| --- | --- |
| fig 2a | fig 2b |
| fig 2c | fig 2d |
| **Fig. 2**  The effect of different centering constants on the correlation of slopes and intercepts. The choice *k* = log(25) leaves them approximately uncorrelated. | |

As *k* increases the cloud of points shifts to the left but the points at the bottom are pulled to the left more than are the points at the top. As a result the original nearly perfect negative linear relationship is totally disrupted by the time *k* = log(25). As *k* increases further the points begin to become positively correlated.

**Multilevel models with three levels using the lme4 package**

I reload the apples data set and fit the variance components (unconditional means) model using the nlme package.

apples <- read.table( 'http://www.unc.edu/courses/2010fall/ecol/563/001/data/lectures/AppleData.csv',  
header=T, sep=',')

names(apples)

[1] "TREE"    "APPLE"   "size"    "appleid" "time"    "diam"

#variance components model--we can use APPLE or appleid as an identifier

out0 <- lme(diam~1, random=~1|TREE/APPLE, data=apples, method='ML', na.action=na.omit)

VarCorr(out0)

            Variance     StdDev      
TREE =      pdLogChol(1)             
(Intercept) 0.0001511123 0.01229278  
APPLE =     pdLogChol(1)             
(Intercept) 0.0093915920 0.09691023  
Residual    0.0030494623 0.05522194

As we've seen previously a second package in R that handles random effects and multilevel designs is lme4 and the function that does what lme does in this package is called lmer. Its way of handling 3-level models is quite different from that of nlme. First, to fit a 2-level model with random intercepts and no predictors we would proceed as follows.

detach(package:nlme)

library(lme4)

lmer(diam~1+(1|appleid), data=apples, REML=F, na.action= na.omit)

Notice that there is no random statement. Instead random effects are entered as if they were ordinary predictors with the exception that the grouping variable is included as part of the predictor name and the entire term, random effect followed by a vertical bar followed by the grouping factor, is enclosed in parentheses. When an additional level is added, for example TREE, that level requires another random predictor in the equation.

There is one complication when fitting 3-level models with lmer that doesn't arise with lme. The lmer function can handle both nested levels (the kind that lme handles) and crossed levels (not permitted with lme). A crossed level doesn't make any sense for the apple data set because it would mean that the same apple could simultaneously come from two trees. There are many situations where crossed random effects might make sense. For instance, suppose abundances of many different species are recorded on different transects at multiple times. If we wanted to analyze all the data simultaneously transect would define a nested factor (with measurement times nested in transect) whereas species could be treated as a crossed factor.

The problem with using the lme4 package for hierarchical designs is that lmer doesn't automatically assume that the levels it encounters are nested. As a result, if the identifier we use to identify the APPLE level takes on the same values for different trees, lmer will assume we want crossed levels and will estimate things accordingly. There are two variables that identify apples in the current data set, APPLE and appleid. The variable APPLE repeats its values on different trees.

apply(table(apples$APPLE, apples$TREE), 1, function(x) sum(x>0))

 1  2  3  4  5  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25   
 4  4  2  3  4  2  3  4  5  4  3  1  3  3  1  6  5  2  5  2  2  3  5  4

Here we see, for example, that APPLE=1 appears on four trees. On the other hand the variable appleid is unique to each tree.

apply(table(apples$appleid, apples$TREE), 1, function(x) sum(x>0))

  1   4   5  10  11  13  14  15  17  18  19  25  32  34  36  40  42  48   
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   
 49  50  51  60  66  67  68  70  72  73  74  76  77  78  83  86  92  93   
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   
 95  99 100 101 111 114 118 120 124 128 129 130 132 135 137 139 142 144   
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   
145 149 152 154 159 175 177 180 184 187 190 195 196 208 210 212 227 230   
  1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   
233 234 235 242 243 246 247 248   
  1   1   1   1   1   1   1   1

Using the lme function it doesn't matter which of APPLE or appleid we use to identify the level-2 units. With the lmer function it does matter. If we enter the following model using lmer we will get the wrong answer. This lmer model treats the intercepts of each APPLE as crossed rather than nested random effects.

#fits a non-nested model with crossed random effects

lmer(diam~1+(1|APPLE)+(1|TREE), data=apples, REML=F, na.action=na.omit)

On the other hand, if we use the variable appleid instead of APPLE to identify the level-2 units the results we get will match the ones we obtained using the lme function, out0, above.

#fits a nested model with appleid nested in TREE

lmer(diam~1+(1|appleid)+(1|TREE), data=apples, REML=F, na.action=na.omit)

detach(package:lme4)

**Analyzing repeated measures data with mixed effects models**

It was noted in [Lecture 26](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture26.htm#correlation) that when multiple random effects are included in a mixed effects model, the correlation that is induced by the random effects can be quite complicated if multiple random coefficients are specified. (In random intercepts models the correlation is quite simple.) While the correlation structure induced by random effects is often robust enough to account for most situations that arise in practice there exist a number of special cases where a special kind of correlation structure might be preferable. A simple example of this is in longitudinal studies where the level-1 observations are made repeatedly over time on the same level-2 units. Here one might expect correlations to decay over time, a structure that is typically not obtained merely by including random effects in the model.

Even with this short-coming, the random effects formulation is still attractive for repeated measures data because it readily deals with observational imbalance (different numbers of observations on the same unit), missing data, and an unequal spacing of observations within units. Fortunately, the nlme package allows one to specify an additional correlation structure for the level-1 observations if desired.

**Graphing the apple data**

The apple data set is a temporal data set. There are up to six time measurements per apple although not all apples were measured six times.

table(apples$appleid[!is.na(apples$diam)])

  1   4   5  10  11  13  14  15  17  18  19  25  32  34  36  40  42  48  49  50  51  60  66  67   
  6   6   6   6   6   6   1   6   5   6   6   6   5   6   6   3   6   6   6   6   4   6   6   6   
 68  70  72  73  74  76  77  78  83  86  92  93  95  99 100 101 111 114 118 120 124 128 129 130   
  6   6   6   6   6   6   6   6   6   6   5   6   5   6   5   6   4   6   6   6   6   6   6   3   
132 135 137 139 142 144 145 149 152 154 159 175 177 180 184 187 190 195 196 208 210 212 227 230   
  6   6   6   6   6   6   6   6   6   6   6   6   6   6   4   6   6   6   6   6   6   6   5   6   
233 234 235 242 243 246 247 248   
  4   5   6   5   6   6   5   5

table(table(apples$appleid[!is.na(apples$diam)]))

 1  3  4  5  6   
 1  2  4 10 63

All of the missing time measurements appear at the end of their time series probably occurring because the apple fell off the tree. We saw last time that there was little variability at the tree level so the observations on individual apples act almost independently of each other regardless of the tree on which they're found. The variable appleid uniquely identifies all the apples in the data set, so I use it in building models. I begin by obtaining a panel display of the data for each apple using lattice graphics.

library(lattice)

xyplot(diam~time|factor(appleid), data=apples, type='o', layout=c(16,5), cex=.7)

|  |
| --- |
| fig 3 |
| **Fig. 3**  Growth data for individual apples |

Although Fig. 3 shows a slight curvilinear trend for some apples, the trend appears linear to a first approximation. Also notice that sample size varies markedly between apples with some apples, apple 14 for instance, having only one data point.

**Constructing the basic level-1 model**

I begin in the usual fashion by first fitting an unconditional means model. This time I use a two-level model with no predictors in which the intercept is allowed to be random at level 2. I need to use appleid to identify the level-2 units because it uniquely identifies each apple.

model0 <- lme(diam~1, random=~1|appleid, data=apples, method='ML', na.action=na.omit)

Using the output of this model we can calculate the intra-class correlation coefficient (ICC), which here estimates the correlation among observations coming from the same apple.

VarCorr(model0)

appleid = pdLogChol(1)   
            Variance    StdDev   
(Intercept) 0.009545563 0.09770140  
Residual    0.003049309 0.05522054

0.009545563/(0.009545563+0.003049309)

[1] 0.7578928

A correlation of 0.76 is pretty strong evidence that we need to account for the data structure when analyzing these data. I next add time as a predictor at level 1. This is what we've previously called a random intercepts model.

model1 <- lme(diam~time, random=~1|appleid, data=apples, method='ML', na.action=na.omit)

VarCorr(model1)

appleid = pdLogChol(1)   
            Variance     StdDev   
(Intercept) 0.0105001410 0.10247020  
Residual    0.0004234962 0.02057902

By comparing the residual variance, σ2, of this model (model1) with that of the unconditional means model (model0), we obtain a pseudo-R2 for level 1.

(0.003049309-0.0004234962)/0.003049309

[1] 0.8611173

So 86% of the variability in apple diameter is explained by its linear relationship with time. This is quite high. Notice this truly is a "pseudo" R2 here because the level-2 variance, τ2, actually increased slightly in going from model0 to mode1l! I generally refer to this phenomenon as leakage. Part of the reason this is happening is simply due to the fact that the variances have a distribution and thus there is a degree of randomness in their estimation. A second reason we may see leakage is because the data aren't balanced. Time contains less information about apples that had fewer temporal observations. Thus although time is primarily a level-1 variable, it is also partially a level-2 variable.

Although the large value of the intra-class correlation coefficient provides a sufficient reason to account for the data structure when building a model, additional motivation can be obtained by comparing the random intercepts model to a linear model in which structure is completely ignored, an ordinary linear regression model.

#compare random intercepts to complete pooling model

modelOLS <- lm(diam~time, data=apples, na.action=na.omit)

sapply(list(modelOLS, model1), AIC)

[1] -780.0338 -1820.8447

So we see that the AIC is substantially lower for the model that accounts for the structure. Because we saw some evidence of curvature in the profiles plotted in Fig. 3, I try a quadratic model even though there is probably little theoretical justification for a quadratic growth model.

#quadratic model

model2 <- lme(diam~time+I(time^2), random=~1|appleid, data=apples, method='ML', na.action=na.omit)

sapply(list(model0,model1,model2),AIC)

[1] -1094.004 -1820.845 -1872.809

So there's fairly strong evidence for a quadratic model. I examine the summary table and find that the quadratic term is highly significant. From the sign of the quadratic term we conclude that the fitted model is a parabola opening down.

summary(model2)

Linear mixed-effects model fit by maximum likelihood  
 Data: apples   
        AIC       BIC   logLik  
  -1872.809 -1852.251 941.4044

Random effects:  
 Formula: ~1 | appleid  
        (Intercept)  Residual  
StdDev:   0.1025829 0.0191331

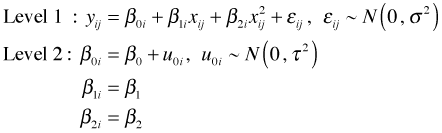
Fixed effects: diam ~ time + I(time^2)   
                 Value   Std.Error  DF   t-value p-value  
(Intercept)  2.8100104 0.012150117 369 231.27436       0  
time         0.0474213 0.002595541 369  18.27029       0  
I(time^2)   -0.0028030 0.000368966 369  -7.59682       0  
 Correlation:   
          (Intr) time    
time      -0.299         
I(time^2)  0.273 -0.977

Standardized Within-Group Residuals:  
         Min           Q1          Med           Q3          Max   
-4.101837057 -0.518966617 -0.009132923  0.536358461  3.209177767

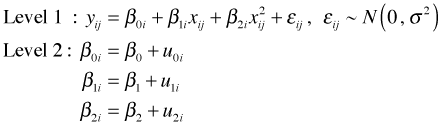
Number of Observations: 451  
Number of Groups: 80

**Adding additional random coefficients**

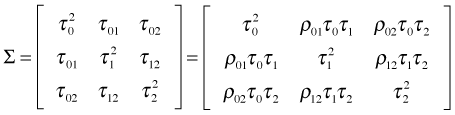
At this point the level-1 model contains three coefficients: an intercept, a linear coefficient, and a quadratic coefficient. Any of these coefficients can be allowed to be random. Previously we've considered only random slopes and intercepts, but in this model we can have random intercepts, linear terms, and quadratic terms. Our current model written in multilevel notation is the following.



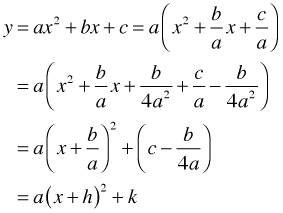
The most general random coefficients model possible is one in which all three coefficients are allowed to be random.



The random effects are assumed to have a joint multivariate normal distribution, random distribution, with



where the second matrix is the way in which R parameterizes the variance-covariance matrix. How does one interpret a model in which all three coefficients are random? Recall from analytical geometry that any parabola can be written in standard form by completing the square.



In standard form *a* controls whether the parabola is fat or skinny, *h* determines the lateral shift of the parabola (more formally –*h* is the *x*-coordinate of the vertex), and *k* determines the vertical shift of the parabola (the *y*-coordinate of the vertex). As we can see the original parameters *a*, *b*, and *c* appear together in a number of the terms so *b* and *c* don't have a clean interpretation. If having a clean interpretation of the parameters of the quadratic model is important then one should use the standard form. This requires fitting a nonlinear mixed effects model, something that is possible with the nlme package but not a topic we have time to discuss in this course.

Roughly speaking we can say that in the original multilevel model β2 controls the fatness of the parabola, β1 affects the horizontal location of the vertex, and β0 affects the height of the vertex. β0 also is the *y*-coordinate of the point where the parabola crosses the *y*-axis. There are six different quadratic random coefficient models we could consider depending on which combination of the three coefficients is allowed to be random. It's pretty clear from Fig. 3 that we need to have random intercepts because the vertical placements of the trajectories are quite variable. So, I only consider the three different ways we can add random linear and random quadratic terms to a random intercepts model.

#random linear terms and intercepts

out2a <- lme(diam~time+I(time^2), random=~time|appleid, data=apples, method='ML', na.action=na.omit)

#random quadratic terms and intercepts

out2b <- lme(diam~time+I(time^2), random=~I(time^2)|appleid, data=apples, method='ML', na.action=na.omit)

#random quadratic terms, linear terms, and intercepts

out2c <- lme(diam~time+I(time^2), random=~time + I(time^2)|appleid, data=apples, method='ML', na.action=na.omit)

sapply(list(model2, out2a, out2b, out2c), AIC)

[1] -1872.809 -2001.357 -1978.680 -2001.919

The random linear coefficients and intercepts appears to do nearly as well as random intercepts, linear terms, and quadratic terms. Because this last model adds three additional parameters, it probably makes sense to stick with the simpler model.

Models with multiple random coefficients often suffer from convergence problems. We've already noted that centering can help in obtaining convergence because it can reduce the correlation between some of the random effects. Something else that sometimes works is to switch optimization functions. The lme function uses the nlmnb function by default but we can specify that it use the optim function instead. This is done by including the control argument in the function call as follows: control=lmeControl(opt='optim'). In the runs above this wasn't necessary. Examining the output from VarCorr for the model with three sets of random coefficients reveals that none of the reported correlations are exceedingly large. Only the large negative correlation between the linear and quadratic random effects comes close. Also none of the reported variances estimates are astronomically small. So there are no red flags here signifying convergence problems.

VarCorr(out2c)

appleid = pdLogChol(time + I(time^2))   
            Variance     StdDev      Corr           
(Intercept) 0.0080250980 0.089582911 (Intr) time    
time        0.0001968539 0.014030462  0.186         
I(time^2)   0.0000015495 0.001244789  0.041 -0.901  
Residual    0.0001836899 0.013553226

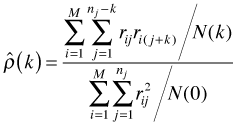
Another red flag to watch out for is when the reported log-likelihood doesn't change or it actually goes down when additional parameters are added to a model. That doesn't happen here either.

sapply(list(model2, out2a, out2b, out2c), logLik)

[1] 941.4044 1007.6786 996.3401 1010.9595

**Checking the adequacy of the correlation structure induced by the random effects**

Because the apple observations were measured over time, the question arises as to whether the correlation induced by the random effects is adequate to handle the serial correlation that probably exists between these observations. Data are serially correlated if observations measured closer in time are more similar to each other than are observations measured further apart in time. We can examine this possibility by plotting the autocorrelation function (ACF). The ACF function from the nlme package calculates the correlation between pairs of observations when separated by increasing time lags. Calculating the autocorrelation of the residuals from a model is a way to assess whether there is any residual correlation remaining that was unaccounted for by the inclusion of predictors and random effects. The correlation between residuals separated by *k* time units, http://www.unc.edu/courses/2010fall/ecol/563/001/images/lectures/lecture29/lagk%20correlation.gif, is estimated with the following formula.



Here http://www.unc.edu/courses/2010fall/ecol/563/001/images/lectures/lecture29/rij.gifis the *j*th level-1 residual for level-2 unit *i* and http://www.unc.edu/courses/2010fall/ecol/563/001/images/lectures/lecture29/rijplusk.gifis the residual from the same unit but *k* time units later. *N*(*k*) is the number of terms that occur in the numerator sum for lag *k* and *N*(0) is the total number of different residuals used in the numerator calculation.

Applying ACF to an lme model yields a list of correlations at various lags. The function assumes that the observations within a group occur in their temporal order in the original data frame with equal time periods between observations. If this is not the case, the results returned by ACF will not be sensible. There is also an acf function in base R but it doesn't understand grouped data and it needs to work with the residuals from the model directly. To use acf here we would need to insert a sufficient number of missing values between the residuals from the different groups to prevent cross-group lags from being calculated and affecting the results. One advantage of using the acf function is that it has capabilities for dealing with gaps in time series whereas the ACF function will get the wrong answer if there are gaps.

ACF(out2a)

  lag          ACF  
1   0  1.000000000  
2   1 -0.295485673  
3   2 -0.319477548  
4   3 -0.151074183  
5   4 -0.006962569  
6   5  0.192215171

To determine if any of these correlations are unusually large we can compare them against approximate two-sided critical bounds at a specified significance level. The recommended bounds are

ACF bounds

where *z*(*p*) denotes the quantile of a standard normal distribution such that *P*(*Z* ≤ *z*(*p*)) = *p* and *N*(*k*) is as defined above.

We don't have to worry about constructing these bounds ourselves. The plot function when applied to an ACF object does this automatically when the α-level is specified with the alpha= argument. Below I plot the ACF with α = .05 and α = .01, the latter being the Bonferroni-adjusted α-level to account for the fact that there are five separate correlations (five nonzero lags) whose significance we wish to test.

plot(ACF(out2a), alpha=.05, key=list(text=list(expression(alpha==.05), cex=1), space='top'))

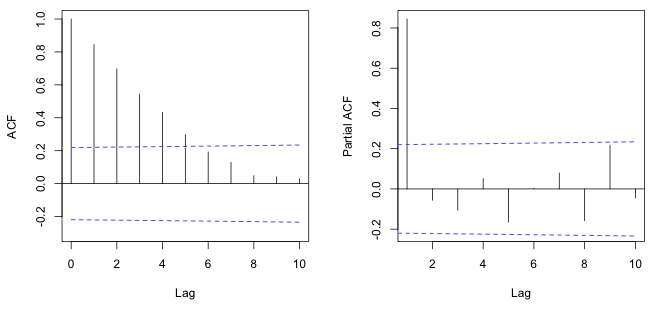
plot(ACF(out2a), alpha=.01, key=list(text=list(expression(alpha==.01), cex=1), space='top'))

|  |  |
| --- | --- |
| fig 4a | fig 4b |
| **Fig. 4**   ACF results for α = .05 (left) and α = .01 (right) | |

Observe that even with the Bonferroni correction there are significant autocorrelations at the first two lags. Also notice that contrary to expectation the correlations between nearby observations (in time) are negative!

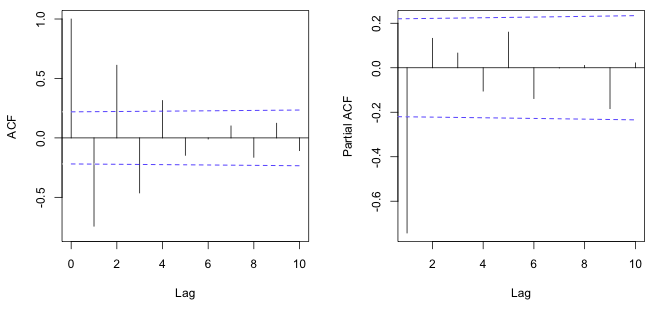
**Adding a correlation structure**

Although the graph does not exhibit a gradual decay in correlation with time as would be expected with temporally correlated data, we can still try to add such a correlation structure to these data. A complete list of all the available correlation structures can be found in Pinheiro and Bates (2000), p. 234. Additional examples can be found in Zuur et al. (2009). The simplest such correlation model is AR(1), an autoregressive structure of order 1, in which the correlation at lag 1 is given by φ and correlations at subsequent lags are given by AR1. Two plots that are useful in characterizing autoregressive processes are plots of the ACF and the partial ACF. For an AR(1) process with a positive correlation the ACF and PACF look as shown in Fig. 5. The ACF decreases gradually to zero while the PACF has a single significant spike followed by non-significant spikes.



**Fig. 5**  Typical ACF and PACF for an AR(1) process with φ > 0 [(R code](http://www.unc.edu/courses/2010fall/ecol/563/001/notes/lec29fig5Rcode.txt))

If the correlation is negative the pattern is similar except that the spikes alternate positive to negative (Fig. 6)



**Fig. 6**   Typical ACF and PACF for an AR(1) process with φ < 0

Another class of temporal correlation structures that have characteristic plots are the moving average processes. For these the patterns in the ACF and PACF are reversed. There is a pacf function in base R that can be used to calculate the partial autocorrelation function for a vector of data.

A correlation structure is added to a linear mixed effects model in R by specifying the correlation= argument in the lme function call. The AR(1) correlation structure is indicated by corAR1( ), or if it is necessary to make the time variable explicit, corAR1(form=~time|appleid). We can use the update function to add a residual correlation structure to the random intercepts and linear coefficients model.

out3a <- update(out2a, correlation=corAR1())

sapply(list(out2a, out3a), AIC)

[1] -2001.357 -2000.999

The AIC has increased indicating that adding the correlation structure has not improved the model. This is not surprising given that the displayed ACF does not match what we'd expect from an AR(1) process. Other temporal correlation models can be obtained with the corARMA function and are specified corARMA(p=, q=) where *p* is the order of the autoregressive process and *q* is the order of the moving average process. For instance the AR(1) process could also be specified as corARMA(p=1, q=0). As the notation suggests autoregressive and moving average processes can be combined to produce what's called an autoregressive moving average (ARMA) model.

I think the use of the ACF and PACF plots is important because when we specify a function in the correlation argument of lme we're trying to model the correlation structure not remove it. If we display the autocorrelation plot after adding a correlation structure to a model, the plot will look different but significant temporal correlations will most likely still be present. So, if we don't use diagnostic plots to screen possible correlation models we're left with no guidelines on how to proceed. My standard strategy is to use ACF and PACF plots to suggest possible models and then I restrict myself to some small set of variations of these. I vary both *p* and *q* slightly from the values suggested by the graph and in the end choose the correlation model that yields the lowest AIC. Zuur et al. (2009) on the other hand routinely fit a large number of correlation models using a wide range of values for *p* and *q* and then use AIC to select a final model.

**Visualizing the final model**

Because the displayed pattern doesn't match any of the standard temporal correlation structures it points to a systemic problem in the model. We could try to fix this by including additional time-varying predictors or by choosing a different functional form for the model such as a nonlinear model. To visualize the final model and perhaps understand the source of the residual temporal correlation, I modify the lattice code we used previously to graph the invertebrate development models. I display the OLS line (not the parabola) as well as the subject-specific and population-average quadratic models. I use the panel.curve function to add the quadratic model to a panel. I set things up as a function so that I can plot the individual apples from one tree at a time.

#add coefficient estimates to data set

subj.specific.parms <- coef(out2a)

pop.average.parms <- fixef(out2a)

#create data frame of subject-specific estimates

ran.data <- data.frame(rownames(subj.specific.parms), subj.specific.parms)

colnames(ran.data) <- c('appleid', 'int', 'time1', 'time2')

#add model estimates to data set

apples2 <- merge(apples,ran.data)

#plotting function

plot.lattice2 <- function(tree) {

#select one tree at a time

data.temp <- apples2[apples2$TREE==tree,]

#reject apples with insufficient data (fewer than 2 observations)

tapply(data.temp$diam, data.temp$appleid, function(x) sum(!is.na(x))) -> guys

good.guys <- names(guys)[guys>1]

data.combined <- data.temp[data.temp$appleid %in% as.numeric(good.guys),]

#generate graph

xyplot(diam~time|factor(appleid), data=data.combined, subscripts=T, panel=function(x, y, subscripts) {

#raw data

panel.xyplot(x, y, pch=16)

#separate regressions OLS line

panel.abline(lm(y~x), lty=1, col=4)

#population-average parabola

func1 <- function(x) pop.average.parms[1]+ pop.average.parms[2]\*x+ pop.average.parms[3]\*x^2

panel.curve(func1, lty=1, lwd=2, col=2)

#subject-specific parabola

func2 <- function(x) data.combined$int[subscripts][1]+ data.combined$time1[subscripts][1]\*x + data.combined$time2[subscripts][1]\*x^2

panel.curve(func2, lty=2)

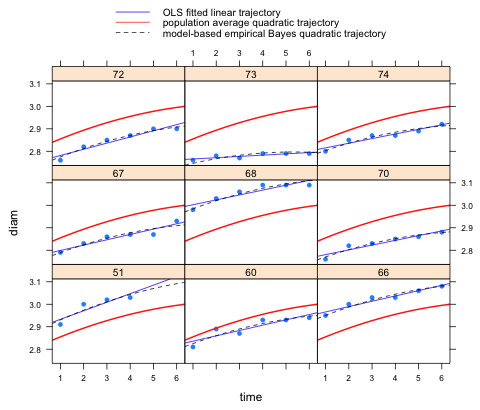
},

strip = strip.custom(par.strip.text = list(cex = 0.85)), par.settings = list(axis.text = list(cex = 0.7)) , key=list(lines=list(lty=c(1,1,2), col=c(4,2,1)), text=list(c("OLS fitted linear trajectory", "population average quadratic trajectory", "model-based empirical Bayes quadratic trajectory"), cex=rep(.85,3)), border=0, space='top' ))}

#results for tree #3

library(lattice)

plot.lattice2(3)



**Fig. 7**  Quadratic model results for tree #3

From the graph of tree 3 we get some understanding of the possible source of the negative lag-1 correlations in the ACF. Notice that in all of the apples shown, the subject-specific model (empirical Bayes trajectory) over-estimates the size of the apple at time 1 and then underestimates it at time 2. A pattern like this if repeated in other apples could be the source of the negative lag 1 correlation. A solution might be to switch to a model that is more flexible than the quadratic model, one with an inflection point that yields an initial S-shaped pattern. Regardless of the autocorrelation problems it is pretty clear that a quadratic model yields a rather close fit to the data.

**Cited References**

* Pinheiro, J. C. and Bates, D. M. 2000. *Mixed-Effects Models in S and S-Plus* . Springer-Verlag, New York.
* Zuur, Alain F., Elena N. Ieno, Neil J. Walker, Anatoly A. Savelieve, and Graham M. Smith. 2009. *Mixed Effects Models and Extensions in Ecology with R.* Springer, New York.

**R Code**

A compact collection of most of the R code displayed in this document appears [here](http://www.unc.edu/courses/2010fall/ecol/563/001/notes/lecture29%20Rcode.html).

[Course Home Page](http://www.unc.edu/courses/2010fall/ecol/563/001/index.html)

|  |
| --- |
| Jack Weiss *Phone:* (919) 962-5930 *E-Mail:* jack\_weiss@unc.edu *Address:* Curriculum in Ecology, Box 3275, University of North Carolina, Chapel Hill, 27599 Copyright © 2010 Last Revised--December 15, 2010 URL: [http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm](http://www.unc.edu/courses/2010fall/ecol/563/001/docs/lectures/lecture29.htm#lecture29) |

http://www.unc.edu/courses/2010fall/ecol/563/001/notes/lecture29%20Rcode.html

### R code for lecture 29 ###

#centering revisited

inverts<-read.table('http://www.unc.edu/courses/2010fall/ecol/563/001/data/lectures/74species.csv',

header=T,sep=',')

library(nlme)

lmList(log(PLD)~log(temp)|species,data=inverts)->out.lmlist

#plot of slopes versus intercepts for all 74 species

plot(coef(out.lmlist)[,1],coef(out.lmlist)[,2],xlab='Intercept',ylab='Slope',xlim=c(-1.5,19))

abline(lm(coef(out.lmlist)[,2]~coef(out.lmlist)[,1]),lty=2,col=2)

#animation to show the effect of centering on the correlation

library(animation)

for(k in c(1,1,1,1:35,35:1)) {

new.int<-coef(out.lmlist)[,1]+log(k)\*coef(out.lmlist)[,2]

r<-cor(new.int,coef(out.lmlist)[,2])

lm(coef(out.lmlist)[,2]~new.int)->out

pval<-round(summary(out)$coefficients[2,4],3)

plot(new.int,coef(out.lmlist)[,2],xlab='Intercept',ylab='Slope',xlim=c(-1.5,13),ylim=c(-3.3,-.1))

abline(out,lty=2,col=2)

mtext(side=3,line=.5,substitute(list('k'==log(x), 'correlation' ==c, 'p'==p),list(x=k,c=round(r,3),p=pval)), cex=1.2,font=2)

Sys.sleep(1)

}

detach(package:animation)

#3-level models revisited

apples<-read.table('http://www.unc.edu/courses/2010fall/ecol/563/001/data/lectures/AppleData.csv',

header=T,sep=',')

names(apples)

#variance components model--we can use APPLE or appleid as an identifier

lme(diam~1,random=~1|TREE/APPLE,data=apples,method='ML',na.action=na.omit)->out0

VarCorr(out0)

detach(package:nlme)

#fitting a 3-level model with the lme4 package

library(lme4)

#if we use APPLE we get a crossed random effects model

lmer(diam~1+(1|TREE)+(1|APPLE),data=apples,REML=F,na.action=na.omit)->out0.lmer

#if we use appleid we get a nested random effects model--this is the one we want

lmer(diam~1+(1|TREE)+(1|appleid),data=apples,REML=F,na.action=na.omit)->out0.lmer1

#not all estimates match nlme: lmer failed to find the TREE variance

VarCorr(out0.lmer1)

#crossed random effects

VarCorr(out0.lmer)

detach(package:lme4)

library(nlme)

#try dropping the TREE level--it seems unimporant. Need to use appleid now

#because APPLE without TREE does not uniquely identify the different apples

lme(diam~1,random=~1|appleid,data=apples,method='ML',na.action=na.omit)->out0a

sapply(list(out0,out0a),AIC)

#we have six time observations per apple

table(apples$appleid)

#examine how apple diameter varies over time--some saturation in growth is seen

library(lattice)

xyplot(diam~time|appleid,data=apples,type='o')

#fit linear and quadratic models of time

lme(diam~time,random=~1|appleid,data=apples,method='ML',na.action=na.omit)->out1

lme(diam~time+I(time^2),random=~1|appleid,data=apples,method='ML',na.action=na.omit)->out2

sapply(list(out0a,out1,out2),AIC)

summary(out2)

#time is a level-1 predictor. We can use the change in level-1 variance as a pseudo-R2 measure

VarCorr(out0a)

VarCorr(out1)

(0.003049309-0.0003660755)/0.003049309

#unfortunately the level-2 variance change a little bit too--hence pseudo-R2

#this may be due to chance but also because the data are unbalanced. As a result

#time is more informative about level-2 units with more data--a level-2 characteristic!

table(apples$appleid[!is.na(apples$diam)])

#random linear terms and intercepts

lme(diam~time+I(time^2),random=~time|appleid,data=apples,method='ML',na.action=na.omit,control=lmeControl(opt='optim'))->out2a

#random linear terms, intercepts, and quadratic terms

lme(diam~time+I(time^2),random=~time+I(time^2)|appleid,data=apples,method='ML',na.action=na.omit,control=lmeControl(opt='optim'))->out2b

sapply(list(out2,out2a,out2b),AIC)

#calculating for temporal correlation in the residuals

ACF(out2a)

#plot the autocorrelation function

plot(ACF(out2a))

#add 95% confidence bands

plot(ACF(out2a),alpha=.05)

#add 95% bands with Bonferroni correction for multiple testing (5 tests)

plot(ACF(out2a),alpha=.01)

#the plot indicates it may be AR(2) or MA(2) or whatever

#add an AR1 structure

update(out2a,correlation=corAR1(form=~time|appleid))->out3

#without the form argument it assumes observations are sorted in time order and equally spaced

update(out2a,correlation=corAR1())->out3

AIC(out2a)

AIC(out3)

#try an AR(2) structure

update(out2a,correlation=corARMA(p=2,q=0))->out4

AIC(out4)

#try an ARMA(1,1) structure

update(out2a,correlation=corARMA(p=1,q=1))->out5

AIC(out5)

#try an MA(2) structure

update(out2a,correlation=corARMA(p=0,q=2))->out5

AIC(out5)

#nothing works--probably a mis-specified model

#a nonlinear model with an S-shaped structure might correct the negative correlations at early lags.