

Using R for Estimating Longitudinal Student Achievement Models

by J.R. Lockwood, Harold Doran and Daniel F. McCaffrey

Overview

The current environment of test-based accountability in public education has fostered increased interest in analyzing longitudinal data on student achievement. In particular, "value-added models" (VAM) that use longitudinal student achievement data linked to teachers and schools to make inferences about teacher and school effectiveness have burgeoned. Depending on the available data and desired inferences, the models can range from straightforward hierarchical linear models to more complicated and computationally demanding cross-classified models. The purpose of this article is to demonstrate how R, via the `lme` function for linear mixed effects models in the `nlme` package (Pinheiro and Bates, 2000), can be used to estimate all of the most common value-added models used in educational research. After providing background on the substantive problem, we develop notation for the data and model structures that are considered. We then present a sequence of increasingly complex models and demonstrate how to estimate the models in R. We conclude with a discussion of the strengths and limitations of the R facilities for modeling longitudinal student achievement data.

Background

The current education policy environment of test-based accountability has fostered increased interest in collecting and analyzing longitudinal data on student achievement. The key aspect of many such data structures is that students' achievement data are linked to teachers and schools over time. This permits analysts to consider three broad classes of questions: what part of the observed variance in student achievement is attributable to teachers or schools; how effective is an individual teacher or school at producing growth in student achievement; and what characteristics or practices are associated with effective teachers or schools. The models used to make these inferences vary in form and complexity, and collectively are known as "value added models" (VAM, McCaffrey et al., 2004).

However, VAM can be computationally demanding. In order to disentangle the various influences on achievement, models must account simultaneously for the correlations among outcomes within students over time and the correlations among outcomes by students sharing teachers or schools in the current

or previous years. The simplest cases have student outcomes fully nested within teachers and schools, in which case standard hierarchical linear models (Raudenbush and Bryk, 2002; Pinheiro and Bates, 2000) are appropriate. When students are linked to changing teachers and/or schools over time, more complicated and computationally challenging cross-classified methods are necessary (Bates and DebRoy, 2003; Raudenbush and Bryk, 2002; Browne et al., 2001). Although the `lme` function of the `nlme` library is designed and optimized for nested structures, its syntax is flexible enough to specify models for more complicated relational structures inherent to educational achievement data.

Data structures

The basic data structures supporting VAM estimation are longitudinal student achievement data $Y_k = (Y_{k0}, \dots, Y_{kT})$ where k indexes students. Typically the data represent scores on an annual standardized examination for a single subject such as mathematics or reading, with $t = 0, \dots, T$ indexing years. For clarity, we assume that all students are from the same cohort, so that year is coincident with grade level. More careful consideration of the distinction between grade level and year is necessary when modeling multiple cohorts or students who are held back. We further assume that the scores represent achievement measured on a single developmental scale so that Y_{kt} is expected to increase over time, with the gain scores $(Y_{kt} - Y_{k,t-1})$ representing growth along the scale. The models presented here can be estimated with achievement data that are not scaled as such, but this complicates interpretation (McCaffrey et al., 2004).

As noted, the critical feature of the data underlying VAM inference is that the students are linked, over time, to higher-level units such as teachers and schools. For some data structures and model specifications, these linkages represent a proper nesting relationship, where outcomes are associated with exactly one unit in each hierarchical level (e.g. outcomes nested within students, who are nested within schools). For more complex data, however, the linkages represent a mixture of nested and cross-classified memberships. For example, when students change teachers and/or schools over time, the repeated measures on students are associated with different higher-level units. Complicating matters is that some models may wish to associate outcomes later in the data series with not only the current unit, but past units as well (e.g. letting prior teachers af-

stuid	schid	y0.tchid	y1.tchid	y2.tchid	y0	y1	y2
1	1	3	102	201	557	601	629
2	1	3	101	203	580	632	660
3	1	2	102	202	566	620	654
...							
287	4	14	113	215	590	631	667
288	4	14	114	213	582	620	644
289	4	13	113	214	552	596	622
...							

Table 1: Sample records from wdat

fect current outcomes).

Depending on the type of model being fit, the data will need to be in one of two formats for use by `lme`. In the first format, commonly called "wide" or "person-level" format, the data are stored in a `dataframe` where each student has exactly one record (row), and repeated measures on that student (i.e. scores from different years) are in different fields (columns). In the other form, commonly called "long" or "person-period" format, repeated measures on students are provided in separate records, with records from the same student in consecutive, temporally ordered rows. Interconversion between these formats is easily carried out via the `reshape` function. In some settings (particularly with fully nested data structures) making the `dataframe` a `groupedData` object, a feature provided by the `nlme` library, will simplify some model specifications and diagnostics. However, we do not use that convention here because some of the cross-classified data structures that we discuss do not benefit from this organization of the data.

Throughout the article we specify models in terms of two hypothetical `dataframes` `wdat` and `ldat` which contain the same linked student achievement data for three years in wide and long formats, respectively. For clarity we assume that the outcome data and all students links to higher level units are fully observed; we discuss missing data at the end of the article. In the wide format, the outcomes for the three years are in separate fields `y0`, `y1`, and `y2`. In the long format, the outcomes for all years are in the field `yt`, with year indicated by the numeric variable `year` taking on values of 0, 1 and 2. The student identifiers and school identifiers for both `dataframes` are given by `stuid` and `schid`, respectively, which we assume are constant across time within students (i.e. students are assumed to be nested in schools). For the wide format `dataframe`, teacher identifiers are in separate fields `y0.tchid`, `y1.tchid`, and `y2.tchid` which for each student link to the year 0, 1 and 2 teachers, respectively. These fields will be used to create the requisite teacher variables of the long format `dataframe` later in the discussion. All identifiers are coded as factors, as this is required by some of the syntax presented here. Finally, in order to

specify the cross-classified models, it is necessary to augment `ldat` with a degenerate grouping variable `dumid` that takes on a single value for all records in the `dataframe`. This plays the role of a highest-level nesting variable along which there is no variation in the data. Sample records from the two hypothetical `dataframes` are provided in Tables 1 and 2.

stuid	schid	dumid	year	yt
1	1	1	0	557
1	1	1	1	601
1	1	1	2	629
2	1	1	0	580
2	1	1	1	632
2	1	1	2	660
...				
287	4	1	0	590
287	4	1	1	631
287	4	1	2	667
288	4	1	0	582
288	4	1	1	620
288	4	1	2	644
...				

Table 2: Sample records from ldat

Model specifications

In this section we consider three increasingly complex classes of VAM, depending on the dimension of the response variable (univariate versus multivariate) and on the relational structure among hierarchical units (nested versus cross-classified). We have chosen this organization to span the range of models considered by analysts, and to solidify, through simpler models, some of the concepts and syntax required by more complicated models. All of the models discussed here specify student, teacher and school effects as random effects. We focus on specification of the random effects structures, providing a discussion of fixed effects for student, teacher or school-level characteristics in a later section. It is outside of the scope of this article to address the many complicated issues that might threaten the validity of the models (McCaffrey et al., 2004); rather we focus on issues of implementation only. It is also outside of the scope of this article to discuss the variety of functions, both in R in general and in the `nlme` library in

particular, that can be used to extract relevant diagnostics and inferences from model objects. The book by Pinheiro and Bates (2000) provides an excellent treatment of these topics.

In specifying the statistical models that follow, for clarity we use the same symbols in different models to draw correspondences among analogous terms (e.g. μ for marginal means, ϵ for error terms) but it should be noted that the actual interpretations of these parameters will vary by model. Finally, we use subscripts i, j, k to index schools, teachers and students, respectively. To denote linkages, we use notation of the form $m(n)$ to indicate that observations or units indexed by n are associated with higher-level units indexed by m . For example, $j(k)$ denotes the index j of the teacher linked to an outcome from student k .

1. Nested univariate models

Though VAM always utilizes longitudinal student level data linked to higher level units of teachers, schools and/or districts, analysts often specify simple models that reduce the multivariate data to a univariate outcome. They also reduce a potentially cross-classified membership structure to a nested one by linking that outcome with only one teacher or school. Such strategies are most commonly employed when only two years of data are available, though they can be used when more years are available, with models being repeated independently for adjacent pairs of years. As such, without loss of generality, we focus on only the first two years of the hypothetical three-year data.

The first common method is the "gain score model" that treats $G_{k1} = (Y_{k1} - Y_{k0})$ as outcomes linked to current year (year 1) teachers. The formal model is:

$$G_{k1} = \mu + \theta_{j1(k)} + \epsilon_{k1} \quad (1)$$

where μ is a fixed mean parameter, θ_{j1} are iid $N(0, \sigma_{\theta1}^2)$ random year one teacher effects and ϵ_{k1} iid $N(0, \sigma_{\epsilon1}^2)$ year one student errors. Letting $g1$ denote the gain scores, the model is a simple one-way random effects model specified in `lme` by

```
lme(fixed=g1~1,data=wdat,random=~1|y1.tchid)
```

In the traditional mixed effects model notation $Y_k = X_k\beta + Z_k\theta_k + \epsilon_k$ (Pinheiro and Bates, 2000), the `fixed` argument to `lme` specifies the response and the fixed effects model $X_k\beta$, and the `random` argument specifies the random effects structure $Z_k\theta_k$. The `fixed` argument in the model above specifies that the gain scores have a grand mean or intercept common to all observations, and the `random` argument specifies that there are random intercepts for each year one teacher.

The other common data reduction strategy for multivariate data is the "covariate adjustment"

model that regresses Y_{k1} on Y_{k0} and current year teacher effects:

$$Y_{k1} = \mu + \beta Y_{k0} + \theta_{j1(k)} + \epsilon_{k1} \quad (2)$$

This model is also easily specified by

```
lme(fixed=y1~y0,data=wdat,random=~1|y1.tchid)
```

As a matter of shorthand notation used throughout the article, in both the `fixed` and `random` arguments of `lme`, the inclusion of a covariate on the right side of a tilde implies the estimation of the corresponding intercept even though the +1 is not explicitly present.

Both of these models are straightforwardly extended to higher levels of nesting. For example, in the covariate adjustment model Equation (2) it may be of interest to examine what part of the marginal teacher variance lies at the school level via the model:

$$\begin{aligned} Y_{k1} &= \mu + \beta Y_{k0} + \theta_{j1(k)} + \epsilon_{k1} \\ \theta_{j1} &= \gamma_{i(j1)} + e_{j1} \end{aligned} \quad (3)$$

with γ_i , e_{j1} and ϵ_{k1} independent normal random variables with means 0 and variances σ_γ^2 , σ_{e1}^2 and $\sigma_{\epsilon1}^2$ respectively. Thus $\sigma_{\theta1}^2$ from the previous model is decomposed into the between-school variance component σ_γ^2 and the within-school variance component σ_{e1}^2 . Using standard nesting notation for statistical formulae in R this additional level of nesting is specified with

```
lme(fixed=y1~y0,data=wdat,
    random=~1|schid/y1.tchid)
```

where the `random` statement specifies that there are random intercepts for schools, and random intercepts for year one teachers within schools.

2. Nested multivariate models

The next most complicated class of models directly treats the longitudinal profiles $Y_k = (Y_{k0}, \dots, Y_{kT})$ as outcomes, rather than reducing them to univariate outcomes, and the challenge becomes modeling the mean and covariance structures of the responses. In this section we assume that the multivariate outcomes are fully nested in higher level units; in the hypothetical example, students are nested in schools, with teacher links ignored (in actual data sets teacher links are often unavailable). For multivariate modeling of student outcomes, it is common for analysts to parameterize growth models for the trajectories of the outcomes, and to examine the variations of the parameters of the growth models across students and schools.

The class of models considered here is of the form

$$Y_{kt} = \mu_t + s_{it(k)} + \epsilon_{kt} \quad (4)$$

where μ_t is the marginal mean structure, s_{it} are school-specific growth trajectories, and ϵ_{kt} are

student-specific residual terms which may themselves contain parameterized growth trajectories. For clarity and conciseness we focus on a limited collection of linear growth models only. Other linear growth models, as well as nonlinear models specified with higher-order or piecewise polynomial terms in time, can be specified using straightforward generalizations of the examples that follow.

The first growth model that we consider assumes that in Equation (4), $\mu_t = \alpha + \beta t$ and $s_{it} = \alpha_i + \beta_i t$. This model allows each school to have its own linear growth trajectory through the random intercepts α_i and random slopes β_i , which are assumed to be bivariate normal with means zero, variances σ_α^2 and σ_β^2 , and covariance $\sigma_{\alpha\beta}$. This model is specified as

```
lme(fixed=yt~year,data=ldat,
    random=~year|schid)
```

Note that because the model is multivariate, the long data `ldat` is necessary.

This model is probably inappropriate because it assumes that deviations of individual student scores from the school-level trajectories are homoskedastic and independent over time within students. More realistically, the student deviations ϵ_{kt} are likely to be correlated within students, and potentially have different variances across time. These features can be addressed either through student-specific random effects, or through a richer model for the within-student covariance structure. To some extent there is an isomorphism between these two pathways; the classic example is that including random student intercepts as part of the mean structure is equivalent to specifying a compound symmetry correlation structure. Which of the two pathways is more appropriate depends on the application and desired inferences, and both are available in `lme`.

As an example of including additional random student effects, we consider random student linear growth so that $\epsilon_{kt} = \gamma_k + \delta_k t + \xi_{kt}$. Like (α_i, β_i) at the school level, (γ_k, δ_k) are assumed to be bivariate normal with mean zero and an unstructured covariance matrix. The random effects at the different levels of nesting are assumed to be independent, and the residual error terms ξ_{kt} are assumed to be independent and homoskedastic. The change to the syntax from the previous model to this model is very slight:

```
lme(fixed=yt~year,data=ldat,
    random=~year|schid/stuid)
```

The `random` statement is analogous to that used in estimating the model in Equation (3); in this case, because there are two random terms at each level, `lme` estimates two (2×2) unstructured covariance matrices in addition to the residual variance.

The other approach to modeling dependence in the student terms ϵ_{kt} is through their covariance structure. `lme` accommodates heteroskedasticity and within-student correlation via the `weights`

and correlation arguments. For the `weights` argument, a wide variety of variance functions that can depend on model covariates as described in Pinheiro and Bates (2000) are available. For the basic models we are interested in, the most important feature is to allow the residual terms to have different variances in each year. For the correlation argument, all of the common residual correlation structures such as compound symmetry, autoregressive and unrestricted (i.e. general positive definite) are available. The model with random growth terms at the school level and an unstructured covariance matrix for $\epsilon_k = (\epsilon_{k0}, \epsilon_{k1}, \epsilon_{k2})$ shared by all students is specified by

```
lme(fixed=yt~year,data=ldat,
    random=~year|schid,
    weights=varIdent(form=~1|year),
    correlation=corSymm(form=~1|stuid))
```

A summary of the model object will report the estimated residual standard deviation for year 0, along with the ratio of the estimated residual standard deviations from other years to that of year 0. It will also report the estimate of the within-student correlation matrix. The functions `varIdent` and `corSymm` combine to specify the unstructured covariance matrix; other combinations of analogous functions can be used to specify a broad array of other covariance structures (Pinheiro and Bates, 2000).

3. Cross classified multivariate models

All of the models in the previous two sections involved only fully nested data structures: observations nested within students, who are nested in higher level units such as schools. Modeling crossed data structures – e.g., accounting for the successive teachers taught by a student – is more challenging. The two most prominent examples of cross-classified models for longitudinal student achievement data are that of the Tennessee Value Added Assessment System (TVAAS) Ballou et al. (2004) and the cross-classified model of Raudenbush and Bryk (2002). Both models have the property that the effects of teachers experienced by a student over time are allowed to “layer” additively, so that past teachers affect all future outcomes.

The TVAAS layered model is the most ambitious VAM effort to date. The full model simultaneously examines outcomes on multiple subjects, from multiple cohorts, across five or more years of testing. We

consider a simplified version of that model for three years of testing on one subject for one cohort. The model for the outcomes of student k is as follows:

$$\begin{aligned} Y_{k0} &= \mu_0 + \theta_{j0(k)} + \epsilon_{k0} \\ Y_{k1} &= \mu_1 + \theta_{j0(k)} + \theta_{j1(k)} + \epsilon_{k1} \\ Y_{k2} &= \mu_2 + \theta_{j0(k)} + \theta_{j1(k)} + \theta_{j2(k)} + \epsilon_{k2} \end{aligned} \quad (5)$$

The random effects for teachers in year t , θ_{jt} , are assumed to be independent and normally distributed with mean 0 and variance $\sigma_{\theta t}^2$. The within student residual terms $\epsilon_k = (\epsilon_{k0}, \epsilon_{k1}, \epsilon_{k2})$ are assumed to be normally distributed with mean 0 and unstructured positive definite covariance matrix Σ_ϵ . The residuals are independent across students and are independent of the teacher effects.

Because the model deals directly with the cross-classified structure of the data, special syntax is required to fit the model with `lme`. The main challenges are building the teacher links and specifying the distribution of the teacher random effects. The first step is to augment the long dataframe with binary variables for each teacher with the property that for each record, the new variable takes on the value 1 if the teacher's random effect is part of that score under the model and zero otherwise. That is, if there are n students in the data set, and if there are J_t teachers in year t with $J = J_0 + J_1 + J_2$, then one must augment the long dataframe with a $(3n \times J)$ matrix Z with elements of 1 where linkages occur and 0 otherwise. Note that in most settings, the matrix is large with primarily entries of 0, which is why sparse matrix techniques are likely to be a beneficial line of computational development for these models.

There are several ways to construct this matrix but one straightforward method is to build it from simple assignment matrices of size $(n \times J_t)$ using the teacher identifier fields `y0.tchid`, `y1.tchid` and `y2.tchid` from the wide format dataframe. The following code carries this out for the hypothetical completely observed dataset; modifications are necessary in the presence of missing data:

```
it<-rep(1:n,each=3)
z0<-model.matrix(~y0.tchid-1,data=wdat)[it,]
z1<-model.matrix(~y1.tchid-1,data=wdat)[it,]
z2<-model.matrix(~y2.tchid-1,data=wdat)[it,]
i0<-seq(from=1,to=(3*n-2),by=3)
z1[i0,]<-0
z2[c(i0,i0+1),]<-0
ldat<-data.frame(ldat,cbind(z0,z1,z2))
```

The calls to `model.matrix` result in $(n \times J_t)$ assignment matrices, which are then expanded to duplicated rows via the index `it`. Then, the appropriate rows of `z1` and `z2` are set to zero to indicate that later teachers are not associated with prior scores. The final call augments the long dataframe with the new variables.

The next step is to specify the distribution of the J teacher random effects. Recall that the teacher effects

are assumed to be independent and normally distributed, with different variances in each year. That is, the covariance matrix of the joint distribution of all teacher random effects (where teachers are blocked by year) is

$$\text{Var}(\theta) = \begin{bmatrix} \sigma_{\theta 0}^2 I_{J_0} & 0 & 0 \\ 0 & \sigma_{\theta 1}^2 I_{J_1} & 0 \\ 0 & 0 & \sigma_{\theta 2}^2 I_{J_2} \end{bmatrix} \quad (6)$$

Communicating this structure to `lme` requires a relatively sophisticated random argument. To construct this argument it is necessary to make use of `nlme`'s `pdMat` classes, which allow specification of a flexible class of positive definite covariance structures for random effects. The specific classes that are required are `pdIdent`, which refers to matrices of the form $\sigma^2 I$, and `pdBlocked`, which creates a block diagonal matrix from its `list` argument. To build the `pdIdent` blocks it is necessary to construct a formula of the form

```
~tchid1 + tchid2 + ... + tchidJ -1
```

for each year; this is used to allow a separate random effect for each teacher. A common way to build such a formula from a vector of character strings is

```
fmla0<-as.formula(paste("~",
  paste(colnames(z0),collapse="+"),"-1"))
```

Letting `fmla1` and `fmla2` refer to the analogous formulas for years 1 and 2, the covariance structure in Equation (6) is specified as:

```
mat<-pdBlocked(list(pdIdent(fmla0),
  pdIdent(fmla1),pdIdent(fmla2)))
```

This covariance structure is then passed to `lme` in the call to fit the layered model:

```
lme(fixed=yt~factor(year)-1,data=ldat,
  random=list(dumid=mat),
  weights=varIdent(form=~1|year),
  correlation=corSymm(form = ~1|dumid/stuid))
```

Because the layered model allows a separate mean for each year, it is necessary to treat year as a factor; including the `-1` in the fixed effects specification removes the intercept so that the yearly means are estimated directly. The random argument makes use of a trick that is necessary to force `lme` to deal with cross-classified data when it is designed for nested data. In brief, `lme` assumes that units are nested in hierarchical levels, and it allows a vector of random coefficients for each higher-level unit that affects the outcomes of all lower-level units nested in that higher-level unit. Because students are not nested in teachers, and thus in principle the random effects for any teacher are eligible to contribute to the responses for any student, it is necessary to impart an artificial nesting structure on the data. This is the role of the

dumid variable which takes on the same value for all records in the dataset. Students are trivially nested in dumid, and teachers are assumed to be attributes of dumid that have random coefficients, thus allowing any teacher to affect any student. Finally, as in the previous section, the weights and correlation arguments allow an unstructured residual covariance matrix within students; note that in the correlation argument it is necessary to respect the artificial nesting of students in dumid.

The cross-classified specification of the model affects the form of the model output in one notable way: a summary of the model object will report a separate estimated standard deviation component for each teacher. However, those values will be common across all teachers in the same year, as required by the pdIdent specifications.

The Raudenbush and Bryk cross-classified model uses the same layering of the teacher effects as the TVAAS layered model, but specifies an explicit model for student growth rather than an unstructured student-level covariance matrix. Specifically, the model for the outcomes of student k is

$$\begin{aligned} Y_{k0} &= (\mu + \mu_k) + \theta_{j0(k)} + \epsilon_{k0} \\ Y_{k1} &= (\mu + \mu_k) + (\beta + \beta_k) + \theta_{j0(k)} + \theta_{j1(k)} + \epsilon_{k1} \\ Y_{k2} &= (\mu + \mu_k) + 2(\beta + \beta_k) \\ &\quad + \theta_{j0(k)} + \theta_{j1(k)} + \theta_{j2(k)} + \epsilon_{k2} \end{aligned} \quad (7)$$

Here the residuals ϵ_{ki} are assumed to be independent and homoskedastic both within and across students. The student random intercepts and slopes (μ_k, β_k) have a bivariate normal distribution with mean zero, variances σ_μ^2 and σ_β^2 and covariance $\sigma_{\mu\beta}$. The distributions of teacher random effects are the same as those assumed by the TVAAS layered model.

In order to specify this model in lme one needs to perform the same steps regarding the teacher random effects as were necessary with the layered model, culminating in the creation of the object mat above. The resulting call to lme is:

```
lme(fixed=yt~year, data=lmat,
    random=list(dumid=mat, stuid=~year))
```

Note that here year is treated as numeric, with the model estimating a marginal linear trend in time. The random statement needs to specify the random effects structure at each of the two levels of nesting: the teacher effects at the degenerate highest-level of nesting have covariance matrix mat, and the random intercepts and slopes for the students have an unrestricted covariance matrix.

Practical Considerations for Cross-Classified Models: Although this article has shown how lme can be used to estimate the most prominent cross-classified models in educational achievement modeling, it is important to bear in mind that it is nei-

ther designed nor optimized for such models. Sometimes, particularly with larger cross-classified models, lme will be exceedingly memory intensive and may not converge. To increase the likelihood of convergence in practical settings, particularly with unstructured student-level correlation, it is useful to start the algorithm at the empirical correlation of the data via the value argument of the corClasses constructor functions. Also in some circumstances we have found that replacing the unstructured student correlation matrix with compound symmetry (via corCompSymm) greatly enhances the stability of the estimation. In our experience with actual student achievement data, compound symmetry is often a reasonable assumption and the impact of this restriction on model inferences is generally minimal. Finally, in our experience with lme, the TVAAS layered model is more likely to converge than the Raudenbush and Bryk cross-classified model.

Another practical constraint is that to our knowledge, lme does not allow there to be more than 200 random effects associated with any one unit at any one level of nesting. This implies, for example, that the layered or cross-classified cannot be fit as specified above when the total number of teachers exceeds 200, because teachers are assumed to be attributes of dumid each with a separate random coefficient. A clever way to sidestep this boundary is to make use of additional, "higher-level" degenerate nesting variables. Although all such variables take on only a single value in the dataset, they are still trivially nested, and it is technically possible to specify up to 200 random effects at each of these levels. With three years of cross-classified data in (for example) the TVAAS layered model, this can be used to expand the capabilities of lme from 200 total teachers to 600 via the following syntax, where the additional degenerate nesting variables dumid2 and dumid3 have been appended to ldat:

```
lme(fixed=Y~factor(year)-1, data=lmat,
    random=list(dumid=pdIdent(fmla1),
                dumid2=pdIdent(fmla2),
                dumid3=pdIdent(fmla3)),
    weights=varIdent(form=~1|year),
    correlation=corSymm(form =
~1|dumid/dumid2/dumid3/stuid))
```

Although such structuring in principle increases the capabilities lme, our experience is that models with significantly more than 200 teachers often do not converge.

Extensions and other issues

Incomplete data

The hypothetical examples assumed fully observed data: all student outcomes and all appropriate links

of units to higher level units are known. In practical applications this will seldom be true. Students may be missing scores, links to teachers, or both. For the nested univariate models, it is common (though potentially not advisable) to use only complete cases; this is achieved by passing `na.action=na.omit` as an additional argument to `lme`. The multivariate models can use incomplete records, though the specific assumptions and models necessary to do so require careful substantive consideration. Incomplete data also make some of the pre-processing operations required to fit cross-classified models somewhat more tedious; in particular, adjustments to the **Z** matrix to account for missing scores and/or missing teacher links are required.

Multivariate annual outcomes

In some cases it might be of interest to model simultaneously the outcomes on multiple assessments in a given year, either on the same or different subjects. Generally this would be done only in the context of the multivariate models, where then one must consider appropriate mean and covariance structures for outcomes both contemporaneously and across time. It is beyond the scope of this article to explore the details of such specifications, but for some data and model structures (particularly with nested rather than cross-classified data) it is possible to specify and estimate such models with `lme`.

Covariates

Although the bulk of this article focused on the specifications of the random effects structures of the models, which can be used to estimate student, teacher or school effects, in practical settings it is often of interest to augment the models with covariates at one or more levels of the data. The relationships of such variables to outcomes may be of substantive interest, they may be used to explain variation in random effects, or they may be used to make more normalized comparisons among estimated unit-specific effects. In most applications, such variables are modeled with fixed effects, and thus are simply specified by passing standard R model formulae as the `fixed` argument to `lme` for any of the models specified above. The syntax seamlessly handles variables at any level of the multilevel data structure.

Conclusion

`lme` is an excellent tool for the many applications of longitudinal student achievement modeling in education research. In many research problems involving moderately sized data sets, it is possible to es-

timate all of the models that are commonly used to make inferences about teacher and school effects as well as the relationships of outcomes with educator practices and characteristics. The current primary limitation is the size of the dataset, particularly with cross-classified models. Although large cross-classified models are outside the current scope of the `nlme` package, theoretical and practical developments are underway to broaden the size and scope of such models that can be estimated (Bates and DeBroy, 2003; Browne et al., 2001).

The examples presented previously can be generalized to handle some additional complications of the data such as team teaching and multiple student cohorts. More complex models involving student-by-teacher interactions and unknown persistence of teacher effects (such as that described in McCaffrey et al. (2004)) are currently beyond the reach of `lme` as well as other standard packages.

Bibliography

- D. Ballou, W. Sanders, and P. Wright. Controlling for student background in value-added assessment of teachers. *Journal of Educational and Behavioral Statistics*, 2004. to appear. 20
- D. Bates and S. DeBroy. Linear mixed models and penalized least squares. *Journal of Multivariate Analysis*, 2003. Submitted for publication. 17, 23
- W. J. Browne, H. Goldstein, and J. Rasbash. Multiple membership multiple classification (MMMC) models. *Statistical Modelling: An International Journal*, 1(2):103–124, 2001. 17, 23
- D. McCaffrey, J. Lockwood, K. D., T. Louis, and L. Hamilton. Models for value-added modeling of teacher effects. *Journal of Educational and Behavioral Statistics*, 2004. to appear. 17, 18, 23
- J. C. Pinheiro and D. M. Bates. *Mixed-Effects Models in S and S-PLUS*. Statistics and Computing. Springer, 2000. 17, 19, 20
- S. Raudenbush and A. Bryk. *Hierarchical Linear Models: Applications and Data Analysis Methods*. Sage Publications, Newbury Park, CA, second edition, 2002. 17, 20

J.R. Lockwood, Daniel F. McCaffrey
The RAND Corporation
lockwood@rand.org, danielm@rand.org

Harold C. Doran
New American Schools
hdoran@nasdc.org

lmeSplines

An R package for fitting smoothing spline terms in LME models

by Rod Ball

Smoothing splines can be formulated in terms of linear mixed models. Corresponding to any smoothing spline term there is a mixed model with a set of random effects, a covariance structure, and a Z-matrix linking the random effects back to observations in the users dataframe. For a smoothing spline model a single variance component is estimated which is inversely proportional to the smoothing parameter for the smoothing spline. The `lmeSplines` package provides functions for generating and manipulating the necessary Z-matrices corresponding to sets of random effects. Using the Choleski decomposition of the covariance matrix, the random effects are transformed to an independent set of random effects, enabling the model to be fitted in `lme` with existing `pdMat` classes. Model predictions can be obtained at the data points and, by interpolation in the Z-matrices, at alternate points using `predict.lme`.

Smoothing splines

Smoothing splines are a parsimonious representation (only one variance parameter is fitted) of a non-parametric curve. Compared with non-linear models, smoothing splines offer a number of advantages:

- Avoiding the need to find a parametric model.
- Avoiding strong model assumptions about the form of the curve.
- Avoiding problems e.g. in longitudinal data where some individuals don't follow the standard curves.
- Can be used to test for smooth departures from a given model.
- A non-linear mixed model can be replaced with a linear model giving more rapid convergence of model fits, and hence enabling more complex and realistic variance structures to be fitted.

Smoothing splines as mixed models

Our package is based on the formulation given in Verbyla *et al* (1999), and extends the capabilities of the NLME package for fitting linear and non-linear mixed models. Readers interested in trying NLME are recommended to read the excellent book (Pinheiro and Bates 2000). The NLME package is substantial, with comprehensive on-line documentation,

however it helps to have worked through the examples, and have an overview understanding of the system to fully appreciate its potential.

In the one dimensional case, the smoothing spline for fitting a function of the form $y = g(t) + \epsilon$, is found by maximising the penalised likelihood:

$$\text{penalised likelihood} = \log \text{likelihood} - \lambda \int g''(t)^2 dx \quad (1)$$

Restricting to the observed data points, the choice of g from an infinite dimensional space reduces to a finite dimensional problem, which can be expressed as a linear mixed model. The mixed model has the form:

$$y = X_s \beta_s + Z_s u_s + \epsilon \quad (2)$$

where β are fixed effects with intercept and slope coefficients, u_s is a set of random effects with $u_s \sim N(0, G_s \sigma_s^2)$, and $\epsilon \sim N(0, \sigma^2)$. The matrices Q and G_s depend only on the the spacings between the time points (see Verbyla *et al* p. 278), where Q is denoted by Δ , Z_s is given by

$$Z_s = Q(Q^t Q)^{-1}, \quad (3)$$

and the smoothing spline parameter λ is related to the mixed model parameters by

$$\lambda = \frac{\sigma^2}{\sigma_s^2}. \quad (4)$$

We transform to a set of independent random effects u'_s with

$$u_s = L u'_s, \quad Z'_s = Z_s L, \quad (5)$$

where L is the Choleski decomposition of G_s , $LL' = G_s$, giving

$$y = X_s \beta_s + Z'_s u'_s + \epsilon \quad (6)$$

with $u'_s \sim N(0, I \sigma_s^2)$.

For further information on the historical context of representing smoothing splines as mixed models and related approaches to smoothing see the discussion (Verbyla *et al* p. 276), and the other references.

Fitting smoothing spline models

The package provides 3 functions: the function `smspline()` calculates the matrix Z'_s in (6), (henceforth referred to as the Z-matrix, or simply Z), with columns corresponding to the unique values of the 'time' covariate, except for the two endpoints. The function `smspline.v()` returns a list containing the matrices X, Q, Z, G_s which can be used for further calculations in the smoothing spline framework. A

third function `approx`. `Z` is used for transforming the spline basis (columns of the `Z`-matrix).

The function `smspline()` is used to calculate the `Z`-matrix. A smoothing spline is fitted by including a term (or block) of the form `pdIdent(~ Z - 1)` in the LME random effects structure.

In the first example a smoothing spline is fit to a curve with measurements made on 100 time points. This takes less than a second to fit on a 2 Ghz Linux PC.

Example 1.

```
> library{lmeSplines}
> data(smSplineEx1)
> # variable 'all' for top level grouping
> smSplineEx1$all <- rep(1,nrow(smSplineEx1))
> # setup spline Z-matrix
> smSplineEx1$Zt <- smspline(~ time,
+ data=smSplineEx1)
> fit1s <- lme(y ~ time, data=smSplineEx1,
+ random=list(all=pdIdent(~Zt - 1)))
```

Note:

1. LME has several methods for specifying the covariance structure for random effects which can be confusing to new users. We will use only one: consisting of a named list with each element corresponding to a level of grouping, specified by a formula, such as `~ time`, or a 'pdMat' object such as `pdIdent(~Zt - 1)`, or a list of such objects wrapped up with `pdBlocked` e.g. `pdBlocked(list(~time,pdIdent(~Zt - 1)))`.
2. Where there is no grouping structure, or we wish to use the 'top-level' grouping, consisting of the whole dataset, we introduce the variable `all` consisting of a vector of ones.
3. The structure `smSplineEx1` is a dataframe with 100 rows. We have added the matrix `Zt` which is a matrix with 100 rows.

```
> str(smSplineEx1)
'data.frame': 100 obs. of 5 variables:
 $ time : num 1 2 3 4 5 6 7 8 9 10 ...
 $ y : num 5.80 5.47 4.57 3.65 ...
 $ y.true: num 4.24 4.46 4.68 4.89 ...
 $ all : num 1 1 1 1 1 1 1 1 1 1 ...
 $ Zt : num [1:100, 1:98] 1.169 ...
```

Fortunately the R dataframes and model formulae can contain matrix terms. Using `~ Zt` in a model formula is equivalent to specifying every column of `Zt` as a term.

4. The `'-1'` in `pdIdent(~Zt - 1)` stops R from adding an intercept term. Thus there are 98 random effects specified, one for every column

of `Zt`, i.e. one for every unique time point except the ends.

The LME fitted model is summarised as:-

```
> summary(fit1s)
Linear mixed-effects model fit by REML
Data: smSplineEx1
AIC BIC logLik
282 292 -137
Random effects:
Formula: ~Zt - 1 | all
Structure: Multiple of an Identity
Zt1 Zt2 Zt3 Zt4
StdDev: 0.0163 0.0163 0.0163 0.0163
. . . .
Zt97 Zt98 Residual
StdDev: 0.0163 0.0163 0.86
```

```
Fixed effects: y ~ time
Value Std.Error DF t-value
(Intercept) 6.50 0.173 98 37.5
time 0.04 0.003 98 13.0
p-value
(Intercept) <.0001
time <.0001
Correlation:
(Intr)
time -0.868
```

```
Standardized Within-Group Residuals:
Min Q1 Med Q3 Max
-2.889 -0.581 0.116 0.659 1.784
```

```
Number of Observations: 100
Number of Groups: 1
```

We read off the estimate of $\hat{\sigma}_s = 0.0163$ (standard deviation parameters for `Zt1`, `Zt2`, ...), and the estimate of $\hat{\sigma} = 0.86$ (residual standard deviation), and hence the smoothing parameter is estimated as $\hat{\lambda} = (\hat{\sigma}/\hat{\sigma}_s)^2 = (0.0163/0.86)^2 = 3.6 \times 10^{-4}$ representing quite a smooth curve. (Cf Figure 1).

Comparison with B-splines and Flexi

We compare the smoothing spline fit to B-spline models with 3 and 5 knots (`fit1bs3`, `fit1bs5` respectively) viz:-

```
> fit1bs3 <- update(fit1s, random=list(all=
+ ~bs(time,3)-1))
> fit1bs5 <- update(fit1s, random=list(all=
+ ~bs(time,5)-1))
> anova(ex1.fit1s,ex1.fit1bs3,ex1.fit1bs5)
Model df AIC BIC logLik L.Rat Pval
fit1s 1 4 282 292 -137
fit1bs3 2 13 298 332 -136 1.86 0.99
fit1bs5 3 24 316 378 -134 4.54 0.95
```

A plot of the smoothing spline fit is shown in Figure 1, with B-spline fits with 3 and 5 knots are shown for comparison. Output from Flexi, a Bayesian smoother (Upsdell 1994,1996; Wheeler and Upsdell 1997) is also shown. The Flexi and smoothing spline curves appear similar with Flexi giving slightly less of a 'hump' near the end. The B-spline models used many more degrees of freedom, took longer to fit (2 sec and 10 sec for 3 and 5 knot points respectively) and gave poorer fits by either AIC or BIC criteria—note the polynomial type wobbles.

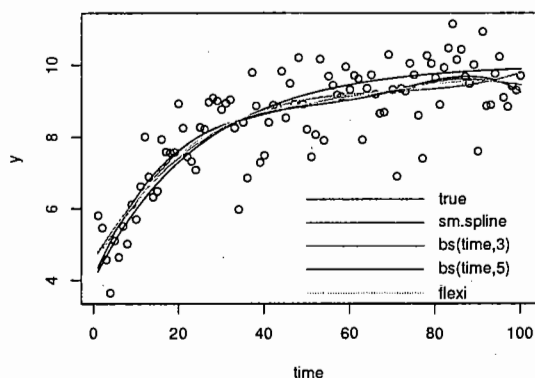


Figure 1: Spline fits. Curves fitted include: the 'true' curve, from which deviations were simulated, a smoothing spline, B-splines with 3,5 knots, and Flexi, a Bayesian smoother.

Aside: Discussion of mixed model formulation in LME, GENSTAT and SAS

By using the variable 'all' we see that models with no grouping structure are a special case of models with a grouping structure. This makes it possible to fit any models, such as commonly fitted in GENSTAT or SAS, where there is no grouping structure (unless specifically requested using a 'repeated' or 'random' statement in SAS), albeit without taking advantage of the efficiency gains possible from LME's treatment of grouping. Model formulae in LME are often more cumbersome than the corresponding GENSTAT REML or SAS PROC MIXED formulae, because LME assumes a general positive definite symmetric covariance matrix for all effects in a formula, while GENSTAT and SAS assume independent sets of random effects for each term in a formula. Suppose that a, b are factors, and we want random terms for a, b, and the interaction, with different variances for different levels of b. In GENSTAT this would be fitted with:

```
vcomp random = a + b + a.b
vstruct[term=b]factor=b; model=diag
vstruct[term=a.b]factor=a,b; model=id,diag
```

In LME, using the levels of a as groups, this would become

```
random=list(
  all=pdBlocked(list(
    pdIdent(~1),pdIdent(~a-1),pdDiag(~b-1))),
  a = pdDiag(~b-1))
```

LME is still lacking some 'pdMat' classes corresponding to the separable (i.e. tensor product) structures in GENSTAT. In many cases, one of the covariance matrices in a tensor product is the identity, and can be modelled using the factor with the identity structure as a grouping factor, as above. If, however, a in the GENSTAT model had a general symmetric covariance matrix V_a and b had a diagonal matrix D_b , the tensor product covariance matrix $V_a \otimes D_b$ for the interaction between a and b would be fitted in GENSTAT with:

```
vcomp random = a + b + a.b
vstruct a; symm
vstruct b; diag
vstruct a.b; symm,diag
```

which currently has no equivalent in LME.

Of course LME has the major advantage that anyone can write a new pdMat class.

Fitting with alternate sets of time points

The fitted curve in Fig. 1 was obtained using fitted values from the NLME fitted model object viz:-

```
lines(smSplineEx1$time,fitted(fit1s),col=2)
```

If the observed data are irregular or more points are needed, predictions at other points can be obtained in principle from the spline model using the matrices X_s, Q, G_s . We have not implemented these calculations, rather, we use a third function `approx.Z()` for transforming the spline basis to alternate sets of points (e.g. finer or coarser grids) for modelling and/or prediction. For example to fit the model on a coarser grid:-

```
# fit model on coarser grid
times20 <- seq(1,100,length=20)
Zt20 <- smspline(times20)
smSplineEx1$Zt20 <- approx.Z(Zt20,times20,
  smSplineEx1$time)
fit1s20 <- lme(y ~ time, data=smSplineEx1,
  random=list(all=pdIdent(~Zt20 - 1)))
```

Predictions with alternate sets of time points

For model predictions we need to generate a newdata argument for `predict.lme()` which contains all the terms in the model including the Z-matrix (here Zt20). This means we need a replacement for

Zt20 with values for each time point in the prediction dataset. This is obtained by calling the function `approx.Z` which uses linear interpolation in the columns of the Z-matrix.

```
# get model predictions on a finer grid
times200 <- seq(1,100,by=0.5)
pred.df <- data.frame(all=rep(1,
  length(times200)),time=times200)
pred.df$Zt20 <- approx.Z(Zt20,times20,
  times200)
yp20.200 <- predict(fit1s20,newdata=pred.df)
```

Note: Linear interpolation is a good approximation here because the spline basis functions (columns of the Z-matrix, i.e. the matrix Z'_s after the transformation (5)) are approximately piecewise linear.

Models with multiple levels of grouping

More general models can contain multiple smoothing spline terms at different levels of grouping and/or with different time covariates.

This is illustrated in the second example. The Spruce dataset contains size measurements taken over time on 13 different trees from each of 4 different plots.

Example 2.

```
data(Spruce)
Spruce$Zday <- smspline(~days, data=Spruce)
spruce.fit2 <- lme(logSize ~ days,
  data=Spruce, random=list(
    all=pdIdent(~Zday - 1),
    plot=pdBlocked(list(
      ~ days, pdIdent(~Zday - 1))),
  Tree = ~1))
```

Fixed or random? Note the inclusion of intercept and slope terms as *random* effects at the plot level. The smoothing spline model (6) specifies *fixed* effects for the slope and intercept. We recommend that the choice of random or fixed effects should be made prior to consideration of fitting a smoothing spline term—whether these should be fixed or random follows the same logic regardless of whether a smoothing spline term is added. We specify random effects since plot effects come from a population of plots, which gives rise to a population of spline curves.

The `pdBlocked` construction at the plot level gives a variance structure in two mutually independent blocks. The first block contains two random effects (intercept and slope) for each plot with a general symmetric 2×2 covariance matrix, and the second contains 11 independent random effects for the smoothing spline.

Alternate possible models include adding linear terms at the tree level, or linear and spline terms. However, for these data a single overall spline term gave the best fit.

Much more complex models can be fitted. Recently we have fitted models for the spatial distribution of wood density in tree stems. There were measurements made for each annual ring of each of a number of discs taken from log ends (at approximately 5m intervals) on 2 trees per clone for each of 10 clones, for a total sample size of around 2650. There were four levels of grouping: all/clone/tree/disc. The model contained linear terms, smooth trends represented by `pdIdent(~Zr-1)`, `pdIdent(~Zh-1)`, in ring number and height, and deviations represented by e.g. `pdIdent(~factor(ring)-1)`, at each level, as appropriate, additionally with an AR(1) correlation structure for within group errors. The LME model fitted was:

Example 3.

```
lme(density ~ ring + height, random=list(
  all = pdBlocked(list(
    pdIdent(~Zr-1),
    pdIdent(~factor(ring) -1),
    pdIdent(~Zh-1),
    pdIdent(~factor(height)-1))),
  clone = pdBlocked(list(
    ~ ring + height,
    pdIdent(~factor(ring) -1),
    pdIdent(~Zh-1),
    pdIdent(~factor(height)-1))),
  tree = pdBlocked(list(
    ~ ring + height,
    pdIdent(~factor(ring) -1),
    pdIdent(~Zh-1),
    pdIdent(~factor(height)-1))),
  disc = pdBlocked(~ ring, pdIdent(~Zr -1)),
  cor=corAR1(form= ~ring))
```

The smoothing spline models fit relatively quickly, but with additional within group correlation structures the model fits take a number of hours. Further details will be reported elsewhere.

References

- Ball, R.D. (2003) Experiences with LME, smoothing splines, and application to *Pinus radiata* wood density. New Zealand Statistical Association Annual Conference, Massey University, Palmerston North.
- Green, P.J. and Silverman, B.W. (1994) *Nonparametric Regression and Generalized Linear Models*. Chapman and Hall, London.
- Guo, W. (2002) Inference in smoothing spline analysis of variance. *J. Royal Stat. Soc. B* 64(4) 887–898.

- Kimeldorf, G.S. and Wahba, G. (1970) A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, 41, 495-502.
- Pinheiro, J.C. and Bates, D.M. (2000) *Mixed-Effects Models in S and S-PLUS* Springer-Verlag, New York.
- Silverman, B.W. (1985) Some aspects of the spline smoothing approach to non parametric regression curve fitting (with discussion). *J.R. Statist. Soc. B*, 47, 1-52.
- Speed, T.P. (1991) Discussion of "That BLUP is a good thing: the estimation of random effects" by G.K. Robinson. *Statist. Sci.*, 6, 42-44.
- Upsdell, M.P. (1994) Bayesian smoothers as an extension of nonlinear regression. *The New Zealand Statistician*, 29, pp.66-81
- Upsdell, M.P. (1996) Choosing an Appropriate Covariance Function in Bayesian Smoothing, *Bayesian Statistics 5*, pp.747-756, Oxford University Press, Oxford.
- Verbyla, A., Cullis, B.R., Kenward, M.G., and Welham, S.J. (1999) The analysis of designed experiments and longitudinal data by using smoothing splines. *Appl. Statist.* 48(3) 269-311.
- Wahba, G. (1978) Improper priors, spline smoothing and the problem of guarding against model errors in regression. *J.R. Statist. Soc. B*, 40, 364-372.
- Wahba, G. (1983) Bayesian "confidence intervals" for the cross-validated smoothing spline. *J.R. Statist. Soc. B* 45, 2133-150.
- Wecker, W.P. and Ansley, C.F. (1983) The signal extraction approach to nonlinear regression and spline smoothing. *J. Am. Statist. Ass.*, 78, 81-89.
- Wheeler, D.M. and Upsdell, M.P. (1997) *Flexi 2.4 reference manual*. New Zealand Pastoral Agriculture Research Institute Ltd, Ruakura, Hamilton, New Zealand. martin.upsdell@agresearch.co.nz
www.agresearch.co.nz
- Rod Ball,
Forest Research, Rotorua, New Zealand
Rod.Ball@forestresearch.co.nz