Spare Matrix Representations of Linear Mixed Models

Douglas Bates
R Development Core Team
Douglas.Bates@R-project.org

April 5, 2004

Abstract

We describe a representation of linear mixed-effects models using positive semidefinite, symmetric, compressed, column-oriented, sparse matrices. This representation provides for efficient evaluation of the profiled log-likelihood or profiled restricted log-likelihood of the model, given the relative precision parameters of the random effects. The evaluation is based upon the Cholesky decomposition of the augmented sparse representation. Additionally, we can use information from this representation to evaluate ECME updates and the gradient and Hessian of the objective criterion.

The ordering of the columns (and, correspondingly, the rows) of a positive semidefinite, symmetric sparse matrix can have a substantial effect on the amount of fill-in generated by the Cholesky decomposition. For the particular matrices considered here the ordering will only become important when random effects are associated with more that one grouping factor and the grouping factors are neither nested nor fully crossed. We say that such factors are partially crossed, a situation that is very common in observational data.

Several methods for determining favorable orderings have been proposed but these generally reorder all the columns. In our case the columns are grouped. We show that we can reorder the columns while preserving the grouping and still attain acceptable levels of fill-in.

1 Introduction

This report is directed at two audiences: sparse matrix researchers and mixed-effects model researchers. To make the ideas accessible to both audiences I will need to introduce a formulation of mixed models for the sparse matrix researchers and also to introduce sparse matrix representation for the mixed model researchers.

Linear mixed-effects models are described in detail in Pinheiro and Bates (2000). In chapter 3 of that book we provide details of computational methods suitable for mixed models with a single level of random effects or with multiple, nested levels of random effects. (Detailed descriptions of terms like "levels" of "random effects" and "nesting" are given later.) Those methods were based on orthogonal-triangular (also called QR) decompositions. Later, in Bates and DebRoy (2004) we generalized these computational methods and showed that all of the important results can be calculated from the Cholesky decomposition of a large, sparse, positive definite, symmetric matrix. We also provide additional computational results for the specific cases considered in Pinheiro and Bates (2000, ch. 3).

In this report I formulate a general approach to linear mixed model calculations using sparse matrix techniques. For an important class of mixed model problems, those with partially crossed grouping factors, the sparse matrix methods will be competitive with, and probably quite superior to, any existing methods. I believe that even for the problems with simpler structure, either a single grouping factor or nested grouping factors, an efficient implementation of the sparse-matrix-based methods will be competitive with the best current methods.

Opportunities for efficiency exist because we must repeatedly perform Cholesky decompositions of matrices with the same pattern of non-zero entries (and somewhat different values of those non-zero entries) and because the non-zero entries occur in dense blocks. Many methods for sparse matrices have both a symbolic phase, where the pattern of the non-zero entries in the result is determined, and a numeric phase, where the actual results are determined. For decompositions part of the symbolic phase is determination of permutation of the rows and columns that reduces fill-in for the decomposition. In our problem the symbolic phase need only be done once and it can be performed based the positions of the blocks. The matrix giving the positions of the blocks is frequently much smaller and easier to manipulate than the matrix that is to be decomposed. Furthermore, the blocks provide

a natural way of using multifrontal techniques, that is, using dense matrix building blocks in a sparse matrix calculation, for this problem.

In the next section I introduce a general form of linear mixed-models and the notation I will use. I also introduce three sample data sets and models. These include a very simple example that can be used to illustrate details of the calculations, a moderate-sized example that has been used to illustrate other methods, and a very large example that can show the savings available with sparse matrix methods. In section In §5 I introduce sparse matrix storage schemes and computational methods and show how they can be applied to the examples. Throughout we illustrate these methods using the `Matrix` package for R (**?**), which provides an implementation of sparse matrix methods for linear mixed-effects models. This R package is based on code from the LDL, TAUCS, Metis, and UMFPACK packages.

2 Linear mixed models

As described in Bates and DebRoy (2004) a linear mixed-effects model can be written as

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{b} + \boldsymbol{\epsilon} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{I}), \boldsymbol{b} \sim \mathcal{N}(\mathbf{0}, \sigma^2\boldsymbol{\Omega}^{-1}), \boldsymbol{\epsilon} \perp \boldsymbol{b} \tag{1}$$

where $\boldsymbol{y}$ is the n-dimensional response vector, $\boldsymbol{X}$ is an $n \times p$ model matrix for the p-dimensional fixed-effects vector $\boldsymbol{\beta}$, $\boldsymbol{Z}$ is the $n \times q$ model matrix for the q-dimensional random-effects vector $\boldsymbol{b}$ that has a Gaussian distribution with mean $\mathbf{0}$ and relative precision matrix $\boldsymbol{\Omega}$ (i.e., $\boldsymbol{\Omega}$ is the precision of $\boldsymbol{b}$ relative to the precision of $\boldsymbol{\epsilon}$), and $\boldsymbol{\epsilon}$ is the random noise assumed to have a spherical Gaussian distribution. The symbol $\perp$ indicates independence of random variables. We assume that $\boldsymbol{X}$ has full column rank and that $\boldsymbol{\Omega}$ is positive definite. Furthermore $\boldsymbol{\Omega}$ is a function of an (unconstrained) parameter vector $\boldsymbol{\theta}$.

Generally p, the dimension of $\boldsymbol{\beta}$, is moderate but q, the dimension of $\boldsymbol{b}$ can be huge. Using the response and the model matrices, $\boldsymbol{y}$, $\boldsymbol{X}$ and $\boldsymbol{Z}$, which are evaluated from the observed data, we determine the estimates of the model parameters; $\boldsymbol{\beta}$, $\boldsymbol{\theta}$, and σ^2, as those values that optimize the likelihood function or, more commonly, a variant called the restricted likelihood. Both the likelihood and the restricted likelihood must be positive and their logarithms, called the log-likelihood and the log-restricted-likelihood, are generally easier

to evaluate and provide a better quadratic approximation for optimization than the original functions, so we work on the log-likelihood scale.

The dimension of $\boldsymbol{\theta}$ is typically very small. Even in complex, "real-world" models applied to data sets with millions of observations the dimension of $\boldsymbol{\theta}$ can be as small as two or three. Conditional on a value of $\boldsymbol{\theta}$, the optimal values of $\boldsymbol{\beta}$ and σ^2 can be determined from a penalized least squares problem based on $\boldsymbol{y}$, $\boldsymbol{X}$, $\boldsymbol{Z}$, and $\boldsymbol{\Omega}$

$$\min_{\boldsymbol{b},\boldsymbol{\beta}} \left\| \tilde{\boldsymbol{y}} - \boldsymbol{\Phi}(\boldsymbol{\theta}) \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{\beta} \end{bmatrix} \right\|^2 \text{ where } \boldsymbol{\Phi}(\boldsymbol{\theta}) = \begin{bmatrix} \boldsymbol{Z} & \boldsymbol{X} \\ \boldsymbol{\Delta}(\boldsymbol{\theta}) & \boldsymbol{0} \end{bmatrix}, \tilde{\boldsymbol{y}} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{bmatrix}, \tag{2}$$

and $\boldsymbol{\Delta}(\boldsymbol{\theta})^\mathsf{T}\boldsymbol{\Delta}(\boldsymbol{\theta}) = \boldsymbol{\Omega}$. One way to solve problem (2) is form $\boldsymbol{L}_e$ and $\boldsymbol{D}_e$, the LDL form of the Cholesky decomposition,

$$\begin{bmatrix} \boldsymbol{Z}^\mathsf{T}\boldsymbol{Z}+\boldsymbol{\Omega} & \boldsymbol{Z}^\mathsf{T}\boldsymbol{X} & \boldsymbol{Z}^\mathsf{T}\boldsymbol{y} \\ \boldsymbol{X}^\mathsf{T}\boldsymbol{Z} & \boldsymbol{X}^\mathsf{T}\boldsymbol{X} & \boldsymbol{X}^\mathsf{T}\boldsymbol{y} \\ \boldsymbol{y}^\mathsf{T}\boldsymbol{Z} & \boldsymbol{y}^\mathsf{T}\boldsymbol{X} & \boldsymbol{y}^\mathsf{T}\boldsymbol{y} \end{bmatrix} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^\mathsf{T} \text{ where } \boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_{ZZ} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{L}_{XZ} & \boldsymbol{L}_{XX} & \boldsymbol{0} \\ \boldsymbol{\ell}_{yZ} & \boldsymbol{\ell}_{yZ} & 1 \end{bmatrix} \tag{3}$$

and $\boldsymbol{D}$ is diagonal with non-negative diagonal elements. The matrix $\boldsymbol{L}$ is unit lower triangular (i.e. it is lower triangular and its diagonal elements are all unity).

We divide the diagonal of $\boldsymbol{D}$ into three components; $\boldsymbol{d}_Z$ of length q, $\boldsymbol{d}_X$ of length p, and the scalar d_y. Our assumptions that $\boldsymbol{\Omega}$ be positive definite and that $\boldsymbol{X}$ have full column rank ensure that the elements of $\boldsymbol{d}_Z$ and $\boldsymbol{d}_X$ are positive. Because the only cases where $d_y = 0$ are trivial cases with exact fits to the response, we will assume that $d_y > 0$.

In Bates and DebRoy (2004) we wrote the Cholesky decomposition (3) as $\boldsymbol{R}^\mathsf{T}\boldsymbol{R}$ where $\boldsymbol{R}$ is upper triangular and, from that, derived expressions for the profiled log-likelihood $\tilde{\ell}(\boldsymbol{\theta})$ and the profiled log-restricted-likelihood, $\tilde{\ell}_R(\boldsymbol{\theta})$. These "profiled" functions are functions of $\boldsymbol{\theta}$ only. They are the values of the corresponding objective function at the conditionally optimal values of the other parameters, $\boldsymbol{\beta}$ and σ^2. Translating from the representation in Bates and DebRoy (2004) to the representation used here provides

$$-2\tilde{\ell}(\boldsymbol{\theta}) = \log\left(\frac{\left|\boldsymbol{Z}^\mathsf{T}\boldsymbol{Z}+\boldsymbol{\Omega}\right|}{|\boldsymbol{\Omega}|}\right) + n\left[1+\log\left(\frac{2\pi d_y}{n}\right)\right] \tag{4}$$

$$-2\tilde{\ell}_R(\boldsymbol{\theta}) = \log\left(\frac{\left|\boldsymbol{Z}^\mathsf{T}\boldsymbol{Z}+\boldsymbol{\Omega}\right|\left|\boldsymbol{L}_{XX}\right|^2}{|\boldsymbol{\Omega}|}\right) + (n-p)\left[1+\log\left(\frac{2\pi d_y}{n-p}\right)\right] \tag{5}$$

As discussed later, $\boldsymbol{\Omega}$ has a block diagonal structure and its determinant is easily evaluated. The determinant $\left|\boldsymbol{Z}^\mathsf{T}\boldsymbol{Z}+\boldsymbol{\Omega}\right|$ is the product of the elements of $\boldsymbol{d}_Z$ and $|\boldsymbol{L}_{XX}|^2$ is the product of the elements of $\boldsymbol{d}_X$.

The other results given in Bates and DebRoy (2004) can be calculated from $\boldsymbol{L}$ and $\boldsymbol{D}$. To make all this feasible the structure and, in particular, the sparsity of $\boldsymbol{Z}$ and $\boldsymbol{\Omega}$ must be exploited.

Sparsity in $\boldsymbol{Z}$ (and $\boldsymbol{\Omega}$) occurs when the random effects vector $\boldsymbol{b}$ is divided into small components associated with one or more factors that group the observations. It is easiest to illustrate this with some examples.

3 Examples of mixed-effects models

3.1 A simple variance components model

In Pinheiro and Bates (2000, §1.1) we discuss measurements of the travel time of a certain type of ultrasonic wave in six different railway rails. Each rail was tested three times yielding a total of 18 observations. Each observation denotes the rail and the observed travel time. A simple data plot (e.g. Fig. 1.1 in Pinheiro and Bates (2000)) shows that the variation between responses on different rails is much greater than the variation between responses on the same rail. We model this as

$$y_{ij}=\mu+b_i+\epsilon_{ij}\quad i=1,\dots,6,\ j=1,\dots,3\quad b_i\sim\mathcal{N}(0,\sigma_b^2),\ \epsilon_{ij}\sim\mathcal{N}(0,\sigma^2) \tag{6}$$

where σ^2 is the within-rail variance and σ_b^2 is the between-rail variance. These are called the *variance components*.

The random variable b_i is the deviation of the mean travel time for rail i from the overall mean travel time μ. These are called the *random effects* associated with the rails.

We can express model (6) in the form (1) by setting

$$\begin{aligned}\boldsymbol{b}&=(b_1,b_2,\dots,b_6)^\mathsf{T}\\ \boldsymbol{\beta}&=\mu\boldsymbol{X}^\mathsf{T}\qquad\qquad=[1,1,\dots,1]^\mathsf{T}\end{aligned} \tag{7}$$

and $\boldsymbol{Z}$ to be the 18×6 matrix of indicators of the rail. The matrix

$$\boldsymbol{\Omega}=\frac{\sigma^2}{\sigma_a^2}\boldsymbol{I}_6 \tag{8}$$

where $\boldsymbol{I}$ is the 6×6 identity matrix. The multiple $\frac{\sigma^2}{\sigma_a^2}$ is the relative precision of the random effects and the parameter θ is a scalar that determines this multiple. To obtain an unconstrained θ we could use the logarithm of the ratio

$$\theta = \log\left(\sigma^2\right) - \log\left(\sigma_a^2\right) \tag{9}$$

The first few rows of $\boldsymbol{Z}$, $\boldsymbol{X}$, and $\boldsymbol{y}$ are

```
> data(Rail, package = "nlme")
> ZXy = cbind(model.matrix(~Rail - 1, Rail), X = 1, y = Rail$travel)
> ZXy[1:4, ]
```

```
  Rail2 Rail5 Rail1 Rail6 Rail3 Rail4 X  y
1     0     0     1     0     0     0 1 55
2     0     0     1     0     0     0 1 53
3     0     0     1     0     0     0 1 54
4     1     0     0     0     0     0 1 26
```

The matrix to be decomposed is obtained by adding $\boldsymbol{\Omega}$ to the 6×6 upper left submatrix of

```
> crossprod(ZXy)
```

```
      Rail2 Rail5 Rail1 Rail6 Rail3 Rail4    X     y
Rail2     3     0     0     0     0     0    3    95
Rail5     0     3     0     0     0     0    3   150
Rail1     0     0     3     0     0     0    3   162
Rail6     0     0     0     3     0     0    3   248
Rail3     0     0     0     0     3     0    3   254
Rail4     0     0     0     0     0     3    3   288
X         3     3     3     3     3     3   18  1197
y        95   150   162   248   254   288 1197 89105
```

For example, if $e^\theta = 0.1$ then the Cholesky decomposition is

```
> options(digits = 5)
> chol(crossprod(ZXy) + diag(c(rep(0.1, 6), 0, 0)))
```

```
       Rail2  Rail5  Rail1  Rail6  Rail3  Rail4      X       y
Rail2 1.7607 0.0000 0.0000 0.0000 0.0000 0.0000 1.7039  53.956
Rail5 0.0000 1.7607 0.0000 0.0000 0.0000 0.0000 1.7039  85.194
Rail1 0.0000 0.0000 1.7607 0.0000 0.0000 0.0000 1.7039  92.010
Rail6 0.0000 0.0000 0.0000 1.7607 0.0000 0.0000 1.7039 140.855
Rail3 0.0000 0.0000 0.0000 0.0000 1.7607 0.0000 1.7039 144.262
Rail4 0.0000 0.0000 0.0000 0.0000 0.0000 1.7607 1.7039 163.573
X     0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.7620  50.673
y     0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000  22.234
```

As seen in this example the cross-product matrix is sparse and only only the diagonal and the last two rows need to be stored. (It happens that $\boldsymbol{Z}^\top\boldsymbol{Z}$ is a multiple of the identity, as is $\boldsymbol{\Omega}$, and this could lead to further simplifications. However, this property depends on the fact that the data are balanced in the sense that there are the same number of observations made on each rail. Although data from a designed experiment may be balanced, observational data are almost never balanced so it is not worthwhile trying to exploit this special structure.)

In general the trailing $p + 1$ rows (and columns) of the cross-product matrix will be dense. The structure of the other summarized as

In §4 we describe two of the popular sparse matrix representations and formation of the Cholesky decomposition of sparse, symmetric, positive semidefinite matrices. The number of non-zero entries in the Cholesky factor can depend on the ordering of the columns (and, correspondingly, the rows) of the original matrix. Various methods have been proposed to choose optimal or near-optimal reorderings. This is an example of symbolic analysis that can be used before the numeric computation to reduce the amount of numeric computation. We describe others.

4 Sparse matrix classes and methods in the Matrix package for R

The simplest representation of a sparse matrix $\boldsymbol{X}$ is to store a triplet (i, j, x_{ij}) for each non-zero element. If the triplets are sorted, say by column order, the column indices will occur in blocks of equal values. In the *compressed, sparse, column-oriented* format the entries are sorted in increasing column order and a set of pointers to the beginning of each column are used instead of

the column values themselves. The *tripletMatrix* class in the *Matrix* package provides the triplet format and the *cscMatrix* class provides the compressed, sparse column-oriented format. In both these classes indices are 0-based (for compatibility with the underlying C code) and not 1-based as is common in R.

```
> library(Matrix)
> mm = new("tripletMatrix", i = as(c(0, 2, 3, 1, 2, 0, 3, 4,
+     3, 4), "integer"), j = as(c(0, 0, 0, 1, 1, 2, 2, 2, 3,
+     3), "integer"), x = (1:10)/10, Dim = as(c(5, 4), "integer"))
> m1 = as(mm, "cscMatrix")
> str(m1)

 list()
 - attr(*, "p")= int [1:5] 0 3 5 8 10
 - attr(*, "i")= int [1:10] 0 2 3 1 2 0 3 4 3 4
 - attr(*, "x")= num [1:10] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
 - attr(*, "Dim")= int [1:2] 5 4
 - attr(*, "factorization")= list()
 - attr(*, "class")= atomic [1:1] cscMatrix
  ..- attr(*, "package")= chr "Matrix"

> as(m1, "matrix")

     [,1] [,2] [,3] [,4]
[1,]  0.1  0.0  0.6  0.0
[2,]  0.0  0.4  0.0  0.0
[3,]  0.2  0.5  0.0  0.0
[4,]  0.3  0.0  0.7  0.9
[5,]  0.0  0.0  0.8  1.0

> diff(m1@p)

[1] 3 2 3 2
```

We see that the p slot in a *cscMatrix* with 4 columns has 5 elements. The first element is always zero and the successive differences are the numbers of non-zero elements in each column. The total number of non-zero elements is the value of the last element of the p slot. This is also the length of the vector of row indices in the i slot.

The validation method for the *cscMatrix* class ensures that the row indices are increasing within columns and reorders the i and x slots if necessary to achieve this. Technically, the objects in this class can be described as sorted, compressed, sparse, column-oriented matrices.

Objects in the *tscMatrix* class represent triangular, sparse, column-oriented matrices and those in the *sscMatrix* class represent symmetric, sparse, column-oriented matrices. Only the upper triangle or the lower triangle, as indicated by `"U"` or `"L"` in the *uplo* slot, of a symmetric matrix is stored. The *crossprod* function applied to a *cscMatrix* produces an *sscMatrix*.

```
> class(m2 <- crossprod(m1))

[1] "sscMatrix"
attr(,"package")
[1] "Matrix"

> as(m2, "matrix")

     [,1] [,2] [,3] [,4]
[1,] 0.14 0.00 0.00 0.00
[2,] 0.10 0.41 0.00 0.00
[3,] 0.27 0.00 1.49 0.00
[4,] 0.27 0.00 1.43 1.81
```

In Statistics we usually define the Cholesky decomposition of a positive semidefinite, symmetric matrix $\boldsymbol{A}$ as an upper triangular matrix $\boldsymbol{R}$ such that $\boldsymbol{A} = \boldsymbol{R}^\mathsf{T}\boldsymbol{R}$ but it is also frequently defined as a lower triangular matrix $\boldsymbol{L}$ such that $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^\mathsf{T}$. Naturally, $\boldsymbol{L}$ and $\boldsymbol{R}$ are transposes of each other. On occasion there are advantages to working with the left factor $\boldsymbol{L}$ instead of the right factor $\boldsymbol{R}$.

For a sparse, symmetric, semidefinite matrix reordering the columns (and, correspondingly, the rows) of $\boldsymbol{A}$ can change the number of non-zero elements in the Cholesky factor. The number of elements in the Cholesky factor is at least the number of non-zero elements in the lower triangle of $\boldsymbol{A}$. Additional non-zeros can be generated during the decomposition. This process is called "fill-in". Various methods of determining a fill-minimizing order have been proposed. We use a graph-based method implemented in the Metis package.

The *chol* function generates the Cholesky decomposition. When applied to an *sscMatrix* object it defaults to generating a fill-reducing permutation and the Cholesky factor of the permuted matrix.

```
> m3 = chol(m2)
> as(m3, "matrix")

        [,1]     [,2]    [,3]    [,4]
[1,] 1.34536 0.000000 0.00000 0.00000
[2,] 1.06291 0.600184 0.00000 0.00000
[3,] 0.00000 0.000000 0.64031 0.00000
[4,] 0.20069 0.094446 0.15617 0.25771

> m3@perm

[1] 3 2 1 0
```

If we set the optional argument *pivot* to *FALSE*, calculation of the fill-reducing permutation is suppressed.

```
> as(chol(m2, pivot = FALSE), "matrix")

        [,1]     [,2]    [,3]    [,4]
[1,] 0.37417  0.00000 0.00000 0.00000
[2,] 0.26726  0.58187 0.00000 0.00000
[3,] 0.72161 -0.33144 0.92705 0.00000
[4,] 0.72161 -0.33144 0.86233 0.66016
```

In this example the fill-reducing permutation reverses the order of the columns and rows of *m2* before taking the decomposition. It results in two fewer non-zero elements in the decomposition than when we suppress the permutation.

4.1 Symbolic versus numeric computation

Calculation of the fill-reducing ordering is an example of a symbolic computation on sparse matrices in that it is based only on the positions of the non-zero elements, not upon their values. Frequently a sparse-matrix computation has both a symbolic phase, which typically determines the number and positions of the non-zero entries in the result, and a numeric phase that actually calculates these non-zero elements.

Evaluation of the profiled log-likelihood or profiled log-restricted-likelihood requires updating the diagonal blocks in $\boldsymbol{Z}^\mathsf{T}\boldsymbol{Z}$ and taking the Cholesky decomposition of the resulting matrix. The symbolic phases, including calculation of a fill-reducing ordering only need to be done once.

Recently Tim Davis has released the LDL package that provides a concise Cholesky factorization of the form $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathsf{T}}$ where $\boldsymbol{L}$ is a unit lower triangular matrix (i.e. all the diagonal elements are unity) and $\boldsymbol{D}$ is diagonal and stored as a single vector. This representation is particularly convenient for us because the diagonal elements (which must be non-zero when $\boldsymbol{A}$ is positive definite) often constitute a substantial portion of the total number of non-zero elements in the Cholesky factor and, in this representations, we do not encounter the extra indexing overhead when accessing these elements. Also the determinant of $\boldsymbol{A}$ (or, for our purposes, the determinants of leading diagonal submatrices of $\boldsymbol{A}$) can be calculated directly from the diagonal of $\boldsymbol{D}$.

As shown in the examples in the next section we can determine fill-reducing orderings and sizes of Cholesky factors of the matrices that we wish to decompose by considering first the pairwise cross-tabulations of the grouping factors.

5 Some examples

Data on achievement scores of Scottish secondary school students is described in Paterson (1991) and included as the data set *ScotsSec* in the *mlmrev* package for R that provides the data sets from the multilevel modelling software review.

```
> data(ScotsSec, package = "Matrix")
> dim(ScotsSec)

[1] 3435    6

> summary(ScotsSec)

     verbal           attain         primary     sex          social
 Min.   :-30.00   Min.   : 1.00   61     :  72   M:1739   Min.   : 0.00
 1st Qu.:-11.00   1st Qu.: 3.00   122    :  68   F:1696   1st Qu.: 0.00
 Median : -2.00   Median : 5.00   32     :  58            Median : 0.00
 Mean   : -2.20   Mean   : 5.68   24     :  57            Mean   : 6.84
 3rd Qu.:  7.00   3rd Qu.: 9.00   6      :  55            3rd Qu.:20.00
 Max.   : 40.00   Max.   :10.00   1      :  54            Max.   :31.00
                                  (Other):3071
```

```
     second
14      : 290
18      : 257
12      : 253
6       : 250
11      : 234
17      : 233
(Other):1918
```

These data contain the achievement scores (*attain*) of 3435 secondary school students in Scotland. Along with demographic data (sex and social class) and a verbal reasoning score based on tests taken at entry to secondary school, the primary school and the secondary school that the student attended are recorded.

There are 148 distinct primary schools and 19 distinct secondary schools represented in these data and, in common models for such data (Lockwood et al., 2003), there would be one or more coefficients in $\boldsymbol{b}$ associated with each school. We say that *primary* and *second* are *grouping factors* for the random effects.

When the random effects are grouped according to more than one grouping factors, as here, it is important to determine if the grouping factors are *crossed* (every level of factor 1 occurs with every level of factor 2) or nested (each level of factor 1 occurs with only one level of factor 2) or partially crossed, which is how we describe grouping factors that are neither (fully) crossed nor strictly nested.

In this case the grouping factors *primary* and *second* are partially crossed. We can express this graphically through the image of the cross-tabulation of the grouping factors. We can generate such a cross-tabulation as a symmetric, sparse, compressed column matrix with the *sscCrosstab* function and use *image* to show the non-zero elements graphically. (Only the non-zero elements in the lower triangle are shown.)

```
> ctab = sscCrosstab(ScotsSec[, c("primary", "second")])
> str(ctab)

 list()
 - attr(*, "Gp")= int [1:3] 0 148 167
 - attr(*, "perm")= int(0)
 - attr(*, "uplo")= chr "U"
```

```
 - attr(*, "p")= int [1:168] 0 1 2 3 4 5 6 7 8 9 ...
 - attr(*, "i")= int [1:470] 0 1 2 3 4 5 6 7 8 9 ...
 - attr(*, "x")= num [1:470] 54 7 3 7 53 55 22 15 33 18 ...
 - attr(*, "Dim")= int [1:2] 167 167
 - attr(*, "factorization")= list()
 - attr(*, "class")= atomic [1:1] sscCrosstab
  ..- attr(*, "package")= chr "Matrix"
```

References

Douglas M. Bates and Saikat DebRoy. Linear mixed models and penalized least squares. *J. of Multivariate Analysis*, 2004. to appear. 2, 3, 4, 5

J.R. Lockwood, Harold Doran, and Daniel F. McCaffrey. Using r for estimating longitudinal student achievement models. *R News*, 3(3):17–23, December 2003. URL http://CRAN.R-project.org/doc/Rnews/. 12

L. Paterson. Socio economic status and educational attainment: a multidimensional and multilevel study. *Evaluation and Research in Education*, 5: 97–121, 1991. 11

José C. Pinheiro and Douglas M. Bates. *Mixed-Effects Models in S and S-PLUS*. Springer, 2000. ISBN 0-387-98957-9. 2, 5