# Mixed models in R using the lme4 package
# Part 6: Theory of linear mixed models, evaluating precision of estimates

Douglas Bates

University of Wisconsin - Madison
and R Development Core Team
<Douglas.Bates@R-project.org>

University of Lausanne
July 2, 2009

# Outline

# Outline

Definition of linear mixed models

The penalized least squares problem

The sparse Cholesky factor

Evaluating the likelihood

# Outline

Definition of linear mixed models

The penalized least squares problem

The sparse Cholesky factor

Evaluating the likelihood

# Outline

Definition of linear mixed models

The penalized least squares problem

The sparse Cholesky factor

Evaluating the likelihood

# Outline

# Definition of linear mixed models

- As previously stated, we define a linear mixed model in terms of two random variables: the $n$-dimensional $\mathcal{Y}$ and the $q$-dimensional $\mathcal{B}$
- The probability model specifies the conditional distribution

$$(\mathcal{Y}|\mathcal{B}=\boldsymbol{b}) \sim \mathcal{N}\left(\boldsymbol{Z}\boldsymbol{b}+\boldsymbol{X}\boldsymbol{\beta}, \sigma^2\boldsymbol{I}_n\right)$$

and the unconditional distribution

$$\mathcal{B} \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{\Sigma}(\boldsymbol{\theta})\right).$$

These distributions depend on the parameters $\boldsymbol{\beta}$, $\boldsymbol{\theta}$ and $\sigma$.

- The probability model defines the *likelihood* of the parameters, given the observed data, $\boldsymbol{y}$. In theory all we need to know is how to define the likelihood from the data so that we can maximize the likelihood with respect to the parameters. In practice we want to be able to evaluate it quickly and accurately.

# Properties of $\Sigma(\theta)$; generating it

- Because it is a variance-covariance matrix, the $q \times q$ $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ must be symmetric and *positive semi-definite*, which means, in effect, that it has a "square root" — there must be another matrix that, when multiplied by its transpose, gives $\boldsymbol{\Sigma}(\boldsymbol{\theta})$.
- We never really form $\boldsymbol{\Sigma}$; we always work with the *relative covariance factor*, $\boldsymbol{\Lambda}(\boldsymbol{\theta})$, defined so that

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}) = \sigma^2 \boldsymbol{\Lambda}(\boldsymbol{\theta}) \boldsymbol{\Lambda}'(\boldsymbol{\theta})$$

  where $\sigma^2$ is the same variance parameter as in $(\boldsymbol{\mathcal{Y}} | \boldsymbol{\mathcal{B}} = \boldsymbol{b})$.
- We also work with a $q$-dimensional "spherical" or "unit" random-effects vector, $\boldsymbol{\mathcal{U}}$, such that

$$\boldsymbol{\mathcal{U}} \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}_q\right), \; \boldsymbol{\mathcal{B}} = \boldsymbol{\Lambda}(\boldsymbol{\theta}) \boldsymbol{\mathcal{U}} \Rightarrow \mathsf{Var}(\boldsymbol{\mathcal{B}}) = \sigma^2 \boldsymbol{\Lambda} \boldsymbol{\Lambda}' = \boldsymbol{\Sigma}.$$

- The linear predictor expression becomes

$$\boldsymbol{Z}\boldsymbol{b} + \boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{Z}\boldsymbol{\Lambda}(\boldsymbol{\theta})\boldsymbol{u} + \boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{U}(\boldsymbol{\theta})\boldsymbol{u} + \boldsymbol{X}\boldsymbol{\beta}$$

  where $\boldsymbol{U}(\boldsymbol{\theta}) = \boldsymbol{Z}\boldsymbol{\Lambda}(\boldsymbol{\theta})$.

# The conditional mean $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=y}$

- Although the probability model is defined from $(\mathcal{Y}|\mathcal{U}=\boldsymbol{u})$, we observe $\boldsymbol{y}$, not $\boldsymbol{u}$ (or $\boldsymbol{b}$) so we want to work with the other conditional distribution, $(\mathcal{U}|\mathcal{Y}=\boldsymbol{y})$.
- The joint distribution of $\mathcal{Y}$ and $\mathcal{U}$ is Gaussian with density

$$\begin{aligned} f_{\mathcal{Y},\mathcal{U}}(\boldsymbol{y},\boldsymbol{u}) &= f_{\mathcal{Y}|\mathcal{U}}(\boldsymbol{y}|\boldsymbol{u})\, f_{\mathcal{U}}(\boldsymbol{u}) \\ &= \frac{\exp(-\frac{1}{2\sigma^2}\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}\boldsymbol{u}\|^2)}{(2\pi\sigma^2)^{n/2}}\,\frac{\exp(-\frac{1}{2\sigma^2}\|\boldsymbol{u}\|^2)}{(2\pi\sigma^2)^{q/2}} \\ &= \frac{\exp(-\left[\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}\boldsymbol{u}\|^2+\|\boldsymbol{u}\|^2\right]/(2\sigma^2))}{(2\pi\sigma^2)^{(n+q)/2}} \end{aligned}$$

- $(\mathcal{U}|\mathcal{Y}=\boldsymbol{y})$ is also Gaussian so its mean is its mode. I.e.

$$\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}} = \arg\min_{\boldsymbol{u}} \left[\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}(\boldsymbol{\theta})\boldsymbol{u}\|^2+\|\boldsymbol{u}\|^2\right]$$

# Outline

# Minimizing a penalized sum of squared residuals

- An expression like $\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}(\boldsymbol{\theta})\boldsymbol{u}\|^2+\|\boldsymbol{u}\|^2$ is called a *penalized sum of squared residuals* because $\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}(\boldsymbol{\theta})\boldsymbol{u}\|^2$ is a sum of squared residuals and $\|\boldsymbol{u}\|^2$ is a penalty on the size of the vector $\boldsymbol{u}$.
- Determining $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}}$ as the minimizer of this expression is a *penalized least squares* (PLS) problem. In this case it is a *penalized linear least squares problem* that we can solve directly (i.e. without iterating).
- One way to determine the solution is to rephrase it as a linear least squares problem for an extended residual vector

$$\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}}=\arg\min_{\boldsymbol{u}}\left\|\begin{bmatrix}\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}\\ \boldsymbol{0}\end{bmatrix}-\begin{bmatrix}\boldsymbol{U}(\boldsymbol{\theta})\\ \boldsymbol{I}_q\end{bmatrix}\boldsymbol{u}\right\|^2$$

This is sometimes called a *pseudo-data* approach because we create the effect of the penalty term, $\|\boldsymbol{u}\|^2$, by adding "pseudo-observations" to $\boldsymbol{y}$ and to the predictor.

# Solving the linear PLS problem

- The conditional mean satisfies the equations

$$[\boldsymbol{U}(\boldsymbol{\theta})\boldsymbol{U}'(\boldsymbol{\theta}) + \boldsymbol{I}_q]\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}} = \boldsymbol{U}'(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}).$$

- This would be interesting but not very important were it not for the fact that we actually can solve that system for $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}}$ even when its dimension, $q$, is very, very large.
- Recall that $\boldsymbol{U}(\boldsymbol{\theta}) = \boldsymbol{Z}\boldsymbol{\Lambda}(\boldsymbol{\theta})$. Because $\boldsymbol{Z}$ is generated from indicator columns for the grouping factors, it is sparse. $\boldsymbol{U}$ is also very sparse.
- There are sophisticated and efficient ways of calculating a sparse Cholesky factor, which is a sparse, lower-triangular matrix $\boldsymbol{L}(\boldsymbol{\theta})$ that satisfies

$$\boldsymbol{L}(\boldsymbol{\theta})\boldsymbol{L}'(\boldsymbol{\theta}) = \boldsymbol{U}(\boldsymbol{\theta})'\boldsymbol{U}(\boldsymbol{\theta}) + \boldsymbol{I}_q$$

and, from that, solving for $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}}$.

# Outline

# The sparse Choleksy factor, $\boldsymbol{L}(\boldsymbol{\theta})$

- Because the ability to evaluate the sparse Cholesky factor, $\boldsymbol{L}(\boldsymbol{\theta})$, is the key to the computational methods in the `lme4` package, we consider this in detail.
- In practice we will evaluate $\boldsymbol{L}(\boldsymbol{\theta})$ for many different values of $\boldsymbol{\theta}$ when determining the ML or REML estimates of the parameters.
- As described in Davis (2006), §4.6, the calculation is performed in two steps: in the *symbolic decomposition* we determine the position of the nonzeros in $\boldsymbol{L}$ from those in $\boldsymbol{U}$ then, in the *numeric decomposition*, we determine the numerical values in those positions. Although the numeric decomposition may be done dozens, perhaps hundreds of times as we iterate on $\boldsymbol{\theta}$, the symbolic decomposition is only done once.

# A fill-reducing permutation, $\boldsymbol{P}$

- In practice it can be important while performing the symbolic decomposition to determine a *fill-reducing permutation*, which is written as a $q \times q$ permutation matrix, $\boldsymbol{P}$. This matrix is just a re-ordering of the columns of $\boldsymbol{I}_q$ and has an orthogonality property, $\boldsymbol{P}\boldsymbol{P}' = \boldsymbol{P}'\boldsymbol{P} = \boldsymbol{I}_q$.
- When $\boldsymbol{P}$ is used, the factor $\boldsymbol{L}(\boldsymbol{\theta})$ is defined to be the sparse, lower-triangular matrix that satisfies

$$\boldsymbol{L}(\boldsymbol{\theta})\boldsymbol{L}'(\boldsymbol{\theta}) = \boldsymbol{P}\left[\boldsymbol{U}'(\boldsymbol{\theta})\boldsymbol{U}(\boldsymbol{\theta}) + \boldsymbol{I}_q\right]\boldsymbol{P}'$$

- In the `Matrix` package for *R*, the `Cholesky` method for a sparse, symmetric matrix (class `dsCMatrix`) performs both the symbolic and numeric decomposition. By default, it determines a fill-reducing permutation, $\boldsymbol{P}$. The `update` method for a Cholesky factor (class `CHMfactor`) performs the numeric decomposition only.

# Applications to models with simple, scalar random effects

- Recall that, for a model with simple, scalar random-effects terms only, the matrix $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ is block-diagonal in $k$ blocks and the $i$th block is $\sigma_i^2\boldsymbol{I}_{n_i}$ where $n_i$ is the number of levels in the $i$th grouping factor.
- The matrix $\boldsymbol{\Lambda}(\boldsymbol{\theta})$ is also block-diagonal with the $i$th block being $\theta_i\boldsymbol{I}_{n_i}$, where $\theta_i = \sigma_i/\sigma$.
- Given the grouping factors for the model and a value of $\boldsymbol{\theta}$ we produce $\boldsymbol{U}$ then $\boldsymbol{L}$, using `Cholesky` the first time then `update`.
- To avoid recalculating we assign

| | |
|---|---|
| `flist` | a list of the grouping factors |
| `nlev` | number of levels in each factor |
| `Zt` | the transpose of the model matrix, $\boldsymbol{Z}$ |
| `theta` | current value of $\boldsymbol{\theta}$ |
| `Lambda` | current $\boldsymbol{\Lambda}(\boldsymbol{\theta})$ |
| `Ut` | transpose of $\boldsymbol{U}(\boldsymbol{\theta}) = \boldsymbol{Z}\boldsymbol{\Lambda}(\boldsymbol{\theta})$ |

# Cholesky factor for the Penicillin model

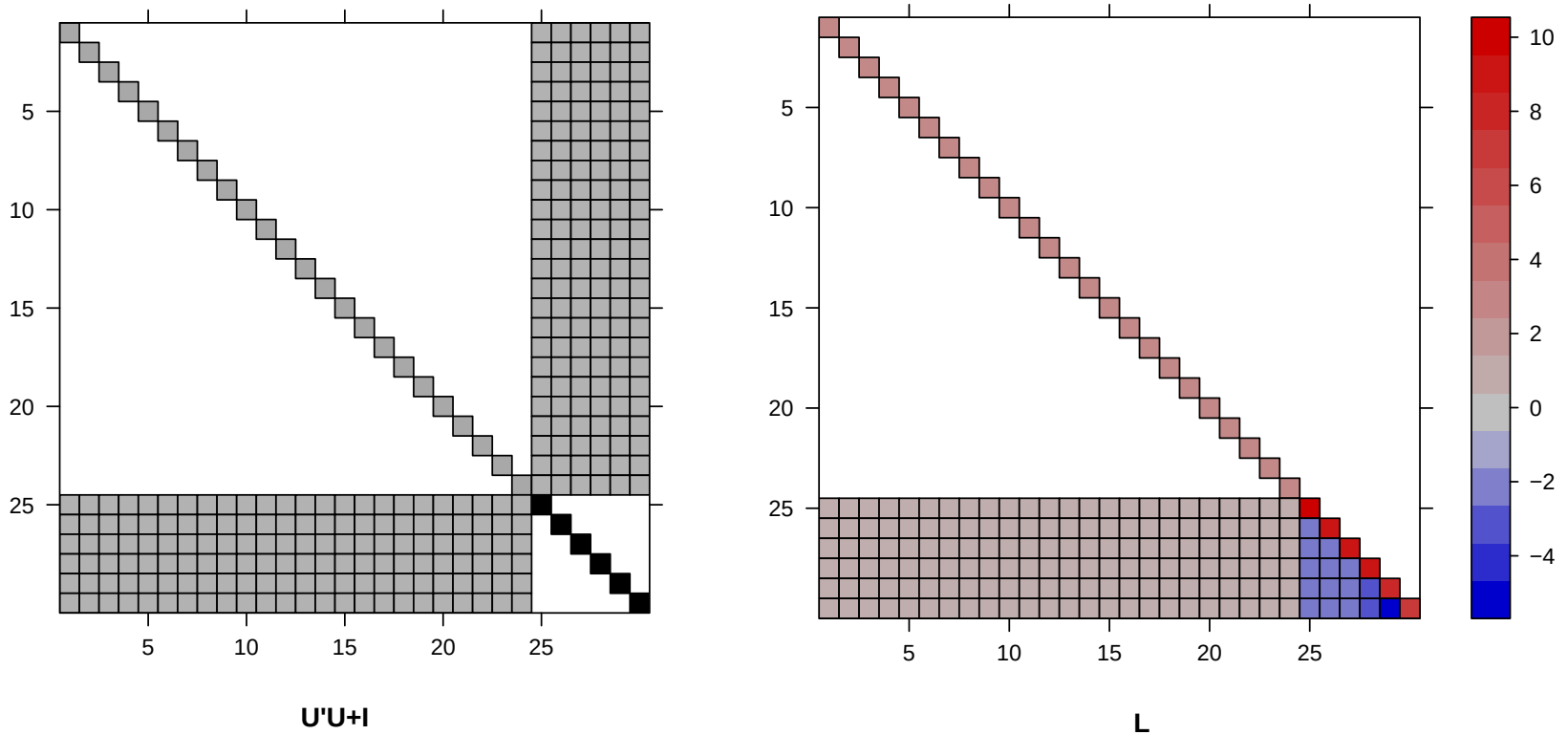
```
> flist <- subset(Penicillin, select = c(plate, sample))
> Zt <- do.call(rBind, lapply(flist, as, "sparseMatrix"))
> (nlev <- sapply(flist, function(f) length(levels(factor(f)))))
```

```
 plate sample
    24      6
```

```
> theta <- c(1.2, 2.1)
> Lambda <- Diagonal(x = rep.int(theta, nlev))
> Ut <- crossprod(Lambda, Zt)
> str(L <- Cholesky(tcrossprod(Ut), LDL = FALSE, Imult = 1))
```

```
Formal class 'dCHMsimpl' [package "Matrix"] with 10 slots
  ..@ x       : num [1:189] 3.105 0.812 0.812 0.812 0.812 ...
  ..@ p       : int [1:31] 0 7 14 21 28 35 42 49 56 63 ...
  ..@ i       : int [1:189] 0 24 25 26 27 28 29 1 24 25 ...
  ..@ nz      : int [1:30] 7 7 7 7 7 7 7 7 7 7 ...
  ..@ nxt     : int [1:32] 1 2 3 4 5 6 7 8 9 10 ...
  ..@ prv     : int [1:32] 31 0 1 2 3 4 5 6 7 8 ...
  ..@ colcount: int [1:30] 7 7 7 7 7 7 7 7 7 7 ...
  ..@ perm    : int [1:30] 23 22 21 20 19 18 17 16 15 14 ...
  ..@ type    : int [1:4] 2 1 0 1
  ..@ Dim     : int [1:2] 30 30
```

# Images of $\boldsymbol{U}'\boldsymbol{U} + \boldsymbol{I}$ and $\boldsymbol{L}$




- Note that there are nonzeros in the lower right of $\boldsymbol{L}$ in positions that are zero in the lower triangle of $\boldsymbol{U}'\boldsymbol{U} + \boldsymbol{I}$. This is described as "fill-in".

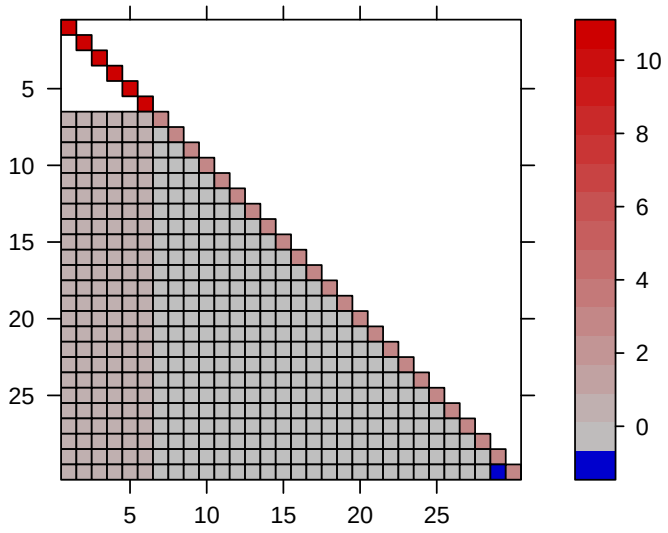
# Reversing the order of the factors

- To show the effect of a fill-reducing permutation, we reverse the order of the factors and calculate the Cholesky factor with and without a fill-reducing permutation.
- We evaluate `nnzero` (number of nonzeros) for `L`, from the original factor order, and for `Lnoperm` and `Lperm`, the reversed factor order without and with permutation

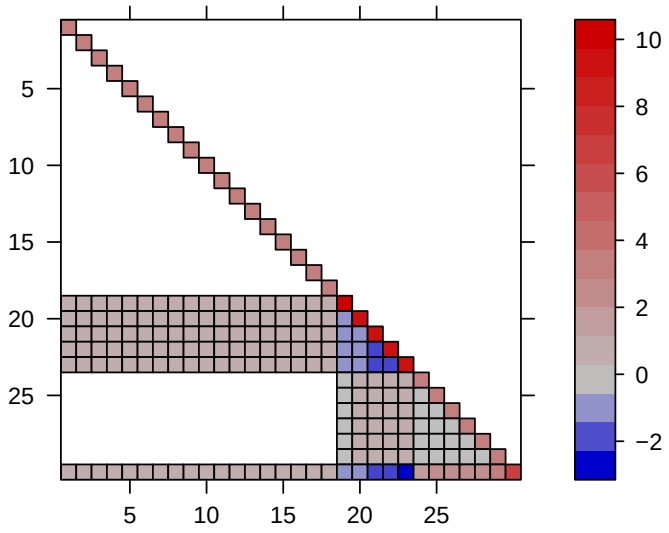
```
> Zt <- do.call(rBind, lapply(flist[2:1], as, "sparseMatrix"))
> Lambda <- Diagonal(x = rep.int(theta[2:1], nlev[2:1]))
> Ut <- crossprod(Lambda, Zt)
> Lnoperm <- Cholesky(tcrossprod(Ut), perm = FALSE, LDL = FALSE,
+     Imult = 1)
> Lperm <- Cholesky(tcrossprod(Ut), LDL = FALSE, Imult = 1)
> sapply(lapply(list(L, Lnoperm, Lperm), as, "sparseMatrix"),
+     nnzero)
```

```
[1] 189 450 204
```

# Images of the reversed factor decompositions




**Lnoperm**




**Lperm**

- Without permutation, we get the worst possible fill-in. With a fill-reducing permutation we get much less but still not as good as the original factor order.
- This is why the permutation is called "fill-reducing", not "fill-minimizing". Getting the fill-minimizing permutation in the general case is a very hard problem.

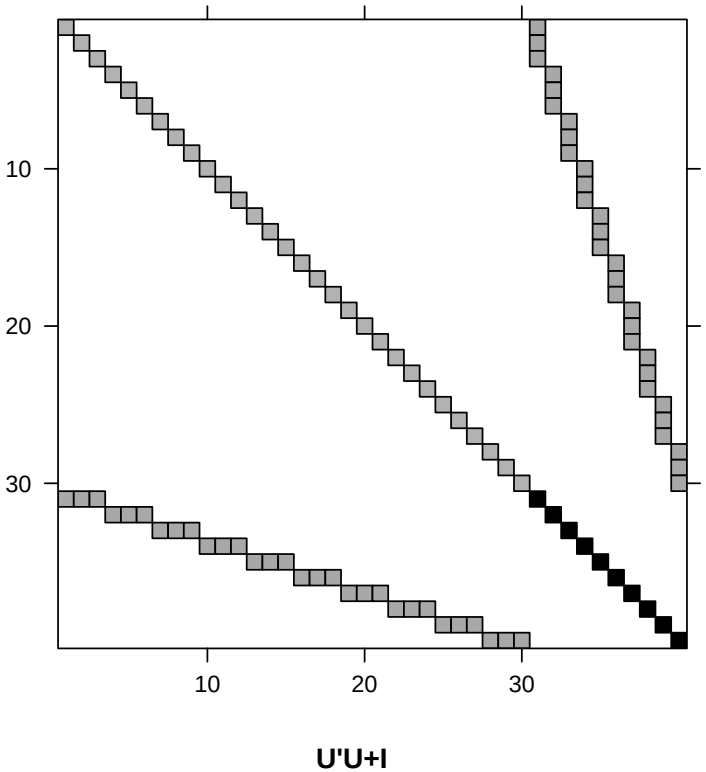
# Cholesky factor for the Pastes data

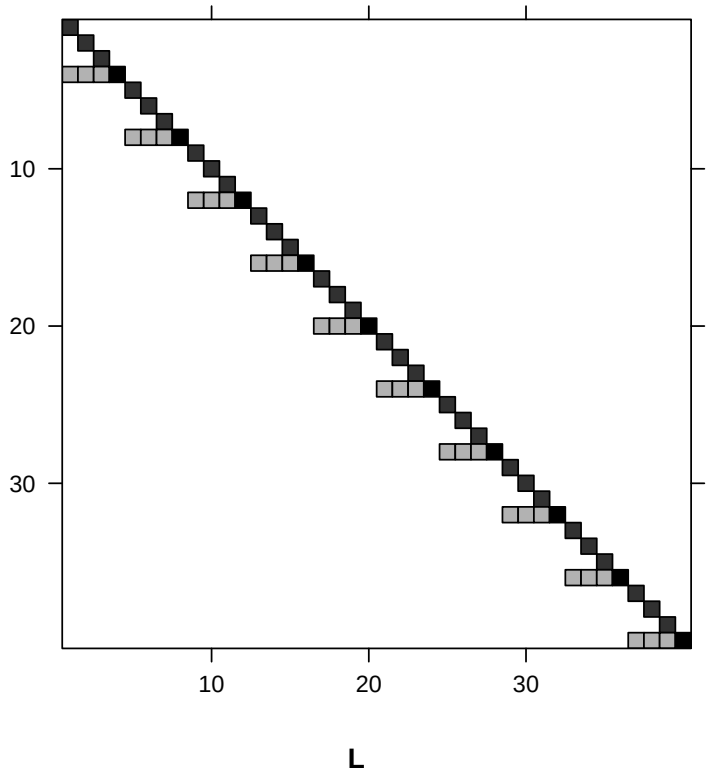
- For the special case of nested grouping factors, such as in the `Pastes` and `classroom` data, there is no fill-in, regardless of the permutation.
- A permutation is nevertheless evaluated but it is a "post-ordering" that puts the nonzeros near the diagonal.

```
> Zt <- do.call(rBind, lapply(flist <- subset(Pastes,
+     , c(sample, batch)), as, "sparseMatrix"))
> nlev <- sapply(flist, function(f) length(levels(factor(f))))
> theta <- c(0.4, 0.5)
> Lambda <- Diagonal(x = rep.int(theta, nlev))
> Ut <- crossprod(Lambda, Zt)
> L <- Cholesky(tcrossprod(Ut), LDL = FALSE, Imult = 1)
> str(L@perm)
```

```
 int [1:40] 2 1 0 30 5 4 3 31 8 7 ...
```

# Image of the factor for the Pastes data




- The image for the Cholesky factor from the `classroom` data model is similar but, with more than 400 rows and columns, the squares for the nonzeros are difficult to see.

# Outline

# The conditional density, $f_{\mathcal{U}|\mathcal{Y}=y}$

- We know the joint density, $f_{\mathcal{Y},\mathcal{U}}(\boldsymbol{y},\boldsymbol{u})$, and

$$f_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}}(\boldsymbol{u}|\boldsymbol{y}) = \frac{f_{\mathcal{Y},\mathcal{U}}(\boldsymbol{y},\boldsymbol{u})}{\int f_{\mathcal{Y},\mathcal{U}}(\boldsymbol{y},\boldsymbol{u})\,d\boldsymbol{u}}$$

  so we almost have $f_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}}$. The trick is evaluating the integral in the denominator, which, it turns out, is exactly the likelihood, $L(\boldsymbol{\theta},\boldsymbol{\beta},\sigma^2|\boldsymbol{y})$, that we want to maximize.

- The Cholesky factor, $\boldsymbol{L}(\boldsymbol{\theta})$ is the key to doing this because

$$\boldsymbol{P}'\boldsymbol{L}(\boldsymbol{\theta})\boldsymbol{L}'(\boldsymbol{\theta})\boldsymbol{P}\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}} = \boldsymbol{U}'(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}).$$

  Although the `Matrix` package provides a one-step `solve` method for this, we write it in stages:

  Solve $\boldsymbol{L}\boldsymbol{c}_{\boldsymbol{u}} = \boldsymbol{P}\boldsymbol{U}'(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})$ for $\boldsymbol{c}_{\boldsymbol{u}}$.
  Solve $\boldsymbol{L}'\boldsymbol{P}\boldsymbol{\mu} = \boldsymbol{c}_{\boldsymbol{u}}$ for $\boldsymbol{P}\boldsymbol{\mu}$ and $\boldsymbol{\mu}$ as $\boldsymbol{P}'\boldsymbol{P}\boldsymbol{\mu}$.

# Evaluating the likelihood

- The exponent of $f_{\mathcal{Y},\mathcal{U}}(\boldsymbol{y},\boldsymbol{u})$ can now be written

$$\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}\boldsymbol{u}\|^2+\|\boldsymbol{u}\|^2=r^2(\boldsymbol{\theta},\boldsymbol{\beta})+\|\boldsymbol{c}_{\boldsymbol{u}}-\boldsymbol{L}'\boldsymbol{P}\boldsymbol{u}\|^2.$$

  where $r^2(\boldsymbol{\theta},\boldsymbol{\beta})=\|\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta}-\boldsymbol{U}\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}\|^2+\|\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}\|^2$. The first term doesn't depend on $\boldsymbol{u}$ and the second is relatively easy to integrate.
- Use the change of variable $\boldsymbol{v}=\boldsymbol{c}_{\boldsymbol{u}}-\boldsymbol{L}'\boldsymbol{P}\boldsymbol{u}$, with $d\boldsymbol{v}=\operatorname{abs}(|\boldsymbol{L}||\boldsymbol{P}|)\,d\boldsymbol{u}$, in

$$\int\frac{\exp\left(\frac{-\|\boldsymbol{c}_{\boldsymbol{u}}-\boldsymbol{L}'\boldsymbol{P}\boldsymbol{u}\|^2}{2\sigma^2}\right)}{(2\pi\sigma^2)^{q/2}}\,d\boldsymbol{u}$$

$$=\int\frac{\exp\left(\frac{-\|\boldsymbol{v}\|^2}{2\sigma^2}\right)}{(2\pi\sigma^2)^{q/2}}\,\frac{d\boldsymbol{v}}{\operatorname{abs}(|\boldsymbol{L}||\boldsymbol{P}|)}=\frac{1}{\operatorname{abs}(|\boldsymbol{L}||\boldsymbol{P}|)}=\frac{1}{|\boldsymbol{L}|}$$

  because $\operatorname{abs}|\boldsymbol{P}|=1$ and $\operatorname{abs}|\boldsymbol{L}|$, which is the product of its diagonal elements, all of which are positive, is positive.

# Evaluating the likelihood (cont'd)

- As is often the case, it is easiest to write the log-likelihood. On the deviance scale (negative twice the log-likelihood) $\ell(\boldsymbol{\theta},\boldsymbol{\beta},\sigma|\boldsymbol{y}) = \log L(\boldsymbol{\theta},\boldsymbol{\beta},\sigma|\boldsymbol{y})$ becomes

$$-2\ell(\boldsymbol{\theta},\boldsymbol{\beta},\sigma|\boldsymbol{y}) = n\log(2\pi\sigma^2) + \frac{r^2(\boldsymbol{\theta},\boldsymbol{\beta})}{\sigma^2} + \log(|\boldsymbol{L}(\boldsymbol{\theta})|^2)$$

- We wish to minimize the deviance. Its dependence on $\sigma$ is straightforward. Given values of the other parameters, we can evaluate the conditional estimate

$$\widehat{\sigma^2}(\boldsymbol{\theta},\boldsymbol{\beta}) = \frac{r^2(\boldsymbol{\theta},\boldsymbol{\beta})}{n}$$

  producing the *profiled deviance*

$$-2\tilde{\ell}(\boldsymbol{\theta},\boldsymbol{\beta}|\boldsymbol{y}) = \log(|\boldsymbol{L}(\boldsymbol{\theta})|^2) + n\left[1 + \log\left(\frac{2\pi r^2(\boldsymbol{\theta},\boldsymbol{\beta})}{n}\right)\right]$$

- However, an even greater simplification is possible because the deviance depends on $\boldsymbol{\beta}$ only through $r^2(\boldsymbol{\theta},\boldsymbol{\beta})$.

# Profiling the deviance with respect to $\boldsymbol{\beta}$

- Because the deviance depends on $\boldsymbol{\beta}$ only through $r^2(\boldsymbol{\theta},\boldsymbol{\beta})$ we can obtain the conditional estimate, $\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta})$, by extending the PLS problem to

$$r^2(\boldsymbol{\theta}) = \min_{\boldsymbol{u},\boldsymbol{\beta}} \left[ \|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{U}(\boldsymbol{\theta})\boldsymbol{u}\|^2 + \|\boldsymbol{u}\|^2 \right]$$

  with the solution satisfying the equations

$$\begin{bmatrix} \boldsymbol{U}(\boldsymbol{\theta})'\boldsymbol{U}(\boldsymbol{\theta}) + \boldsymbol{I}_q & \boldsymbol{U}(\boldsymbol{\theta})'\boldsymbol{X} \\ \boldsymbol{X}'\boldsymbol{U}(\boldsymbol{\theta}) & \boldsymbol{X}'\boldsymbol{X} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}=\boldsymbol{y}} \\ \widehat{\boldsymbol{\beta}}(\boldsymbol{\theta}) \end{bmatrix} = \begin{bmatrix} \boldsymbol{U}(\boldsymbol{\theta})'\boldsymbol{y} \\ \boldsymbol{X}'\boldsymbol{y}. \end{bmatrix}$$

- The profiled deviance, which is a function of $\boldsymbol{\theta}$ only, is

$$-2\tilde{\ell}(\boldsymbol{\theta}) = \log(|\boldsymbol{L}(\boldsymbol{\theta})|^2) + n\left[1 + \log\left(\frac{2\pi r^2(\boldsymbol{\theta})}{n}\right)\right]$$

# Solving the extended PLS problem

- For brevity we will no longer show the dependence of matrices and vectors on the parameter $\boldsymbol{\theta}$.
- As before we use the sparse Cholesky decomposition, with $\boldsymbol{L}$ and $\boldsymbol{P}$ satisfying $\boldsymbol{LL}' = \boldsymbol{P}(\boldsymbol{U}'\boldsymbol{U} + \boldsymbol{I})$ and $\boldsymbol{c_u}$, the solution to $\boldsymbol{Lc_u} = \boldsymbol{PU}'\boldsymbol{y}$.
- We extend the decomposition with the $q \times p$ matrix $\boldsymbol{R}_{ZX}$, the upper triangular $p \times p$ matrix $\boldsymbol{R}_X$, and the $p$-vector $\boldsymbol{c_\beta}$ satisfying

$$\begin{aligned}\boldsymbol{LR}_{ZX} &= \boldsymbol{PU}'\boldsymbol{X}\\ \boldsymbol{R}'_X\boldsymbol{R}_X &= \boldsymbol{X}'\boldsymbol{X} - \boldsymbol{R}'_{ZX}\boldsymbol{R}_{ZX}\\ \boldsymbol{R}'_X\boldsymbol{c_\beta} &= \boldsymbol{X}'\boldsymbol{y} - \boldsymbol{R}'_{ZX}\boldsymbol{c_u}\end{aligned}$$

so that

$$\begin{bmatrix}\boldsymbol{P}'\boldsymbol{L} & \boldsymbol{0}\\ \boldsymbol{R}'_{ZX} & \boldsymbol{R}'_X\end{bmatrix}\begin{bmatrix}\boldsymbol{L}'\boldsymbol{P} & \boldsymbol{R}_{ZX}\\ \boldsymbol{0} & \boldsymbol{R}_X\end{bmatrix} = \begin{bmatrix}\boldsymbol{U}'\boldsymbol{U} + \boldsymbol{I} & \boldsymbol{U}'\boldsymbol{X}\\ \boldsymbol{X}'\boldsymbol{U} & \boldsymbol{X}'\boldsymbol{X}\end{bmatrix}.$$

# Solving the extended PLS problem (cont'd)

- Finally we solve

$$\boldsymbol{R}_X\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta}) = \boldsymbol{c}_{\boldsymbol{\beta}}$$

$$\boldsymbol{L}'\boldsymbol{P}\boldsymbol{\mu}_{\boldsymbol{\mathcal{U}}|\boldsymbol{\mathcal{Y}}} = \boldsymbol{c}_{\boldsymbol{u}} - \boldsymbol{R}_{ZX}\widehat{\boldsymbol{\beta}}(\boldsymbol{\theta})$$

- The profiled REML criterion also can be expressed simply. The criterion is

$$L_R(\boldsymbol{\theta}, \sigma^2|\boldsymbol{y}) = \int L(\boldsymbol{\theta}, \boldsymbol{\beta}, \sigma^2|\boldsymbol{y})\, d\boldsymbol{\beta}$$

  The same change-of-variable technique for evaluating the integral w.r.t. $\boldsymbol{u}$ as $1/\operatorname{abs}(|\boldsymbol{L}|)$ produces $1/\operatorname{abs}(|\boldsymbol{R}_X|)$ here and removes $(2\pi\sigma^2)^{p/2}$ from the denominator. On the deviance scale, the profiled REML criterion is

$$-2\tilde{\ell}_R(\boldsymbol{\theta}) = \log(|\boldsymbol{L}|^2)+\log(|\boldsymbol{R}_x|^2)+(n-p)\left[1+\log\left(\frac{2\pi r^2(\boldsymbol{\theta})}{n-p}\right)\right]$$

- These calculations can be expressed in a few lines of *R* code. Assume `rho` contains `y`, `X`, `Zt`, `REML`, `L`, `nlev` and `XtX` ($\boldsymbol{X}'\boldsymbol{X}$).

# Code for evaluating the profiled deviance

```
profDev <- function(rho, theta) {
  stopifnot(is.numeric(theta), length(theta)==length(rho$nlev))
  Ut <- crossprod(Diagonal(x=rep.int(theta,rho$nlev)),rho$Zt)
  L <- update(rho$L, Ut, mult = 1)
  cu <- solve(L, solve(L, Ut %*% rho$y, sys = "P"), sys = "L")
  RZX <- solve(L, solve(L, Ut %*% rho$X, sys = "P"), sys = "L")
  RX <- chol(rho$XtX - crossprod(RZX))
  cb <- solve(t(RX),crossprod(rho$X,rho$y)- crossprod(RZX, cu))
  beta <- solve(RX, cb)
  u <- solve(L,solve(L,cu - RZX %*% beta, sys="Lt"), sys="Pt")
  fitted <- as.vector(crossprod(Ut, u) + rho$X %*% beta)
  prss <- sum(c(rho$y - fitted, as.vector(u))^2)
  n <- length(fitted);  p <- ncol(RX)
  if (rho$REML) return(determinant(L)$mod +
                       2 * determinant(RX)$mod +
                       (n-p) * (1+log(2*pi*prss/(n-p))))
  determinant(L)$mod + n * (1 + log(2*pi*prss/n))
}
```

# Checking profDev, lmer version of fit

```
> invisible(lmer(mathgain ~ mathkind + minority + ses +
+     (1 | classid) + (1 | schoolid), classroom, verbose = 1,
+     REML = FALSE))
  0:     11466.913: 0.836158 0.489669
  1:     11447.349:  0.00000 0.0605665
  2:     11415.114: 0.000182491 0.530484
  3:     11402.540: 0.0597687 0.347260
  4:     11392.970: 0.287979 0.373802
  5:     11391.654: 0.325504 0.300787
  6:     11391.532: 0.335765 0.315397
  7:     11391.532: 0.336712 0.313883
  8:     11391.532: 0.336117 0.314549
  9:     11391.532: 0.336004 0.314503
 10:     11391.532: 0.335972 0.314546
```

# Checking profDev, nlminb applied to profDev

```
> ls.str(rho <- with(classroom, simplemer(list(classid,
+     schoolid), mathgain, model.matrix(~mathkind + minority +
+     ses), REML = FALSE)))

L : Formal class 'dCHMsimpl' [package "Matrix"] with 10 slots
nlev :  int [1:2] 312 107
REML :  logi FALSE
X :  num [1:1190, 1:4] 1 1 1 1 1 1 1 1 1 1 ...
XtX :  num [1:4, 1:4] 1190 555324 806 -15.4 555324 ...
y :  int [1:1190] 32 109 56 83 53 65 51 66 88 -7 ...
Zt : Formal class 'dgCMatrix' [package "Matrix"] with 6 slots

> invisible(nlminb(c(0.836158, 0.489669), function(x) profDev(rh
+     x), lower = c(0, 0), control = list(trace = 1)))

  0:     11466.913: 0.836158 0.489669
  1:     11447.349:  0.00000 0.0605677
  2:     11415.114: 0.000182488 0.530485
  3:     11402.540: 0.0597688 0.347263
  4:     11392.970: 0.287969 0.373786
  5:     11391.654: 0.325506 0.300795
  6:     11391.532: 0.335765 0.315398
  7:     11391.532: 0.336710 0.313884
```

# How lmer works

- Essentially `lmer` takes its arguments and creates a structure like the `rho` environment shown above. The optimization of the profiled deviance or the profiled REML criterion happens within this environment.
- The creation of $\Lambda(\theta)$ is somewhat more complex for models with vector-valued random effects but not excessively so.
- Some care is taken to avoid allocating storage for large objects during each function evaluation. Many of the objects created in `profDev` are updated in place within `lmer`.
- Once the optimizer, `nlminb`, has converged some additional information for the summary is calculated.

# Recap

- For a linear mixed model, even one with a huge number of observations and random effects like the model for the grade point scores, evaluation of the ML or REML profiled deviance, given a value of $\boldsymbol{\theta}$, is straightforward. It involves updating $\boldsymbol{\Lambda}$, $\boldsymbol{U}$, $\boldsymbol{L}$, $\boldsymbol{R}_{ZX}$, $\boldsymbol{R}_X$, calculating the penalized residual sum of squares, $r^2(\boldsymbol{\theta})$ and two determinants of triangular matrices.
- The profiled deviance can be optimized as a function of $\boldsymbol{\theta}$ only. The dimension of $\boldsymbol{\theta}$ is usually very small. For the grade point scores there are only three components to $\boldsymbol{\theta}$.