Exploiting sparsity in model matrices

Douglas Bates and Martin Maechler

Department of Statistics
University of Wisconsin – Madison U.S.A.

Seminar fr Statistik
ETH Zurich Switzerland
(bates|maechler)@R-project.org (R-Core)

DSC2009, Copenhagen
July 14, 2009

Outline

1. Model matrices with many columns
2. Applications to linear mixed models
3. The penalized least squares problem
4. Using the sparse Cholesky for mixed models
5. Evaluating the likelihood

Outline

1. Model matrices with many columns
2. Applications to linear mixed models
3. The penalized least squares problem
4. Using the sparse Cholesky for mixed models
5. Evaluating the likelihood

Outline

1. Model matrices with many columns
2. Applications to linear mixed models
3. The penalized least squares problem
4. Using the sparse Cholesky for mixed models
5. Evaluating the likelihood

Outline

1. Model matrices with many columns
2. Applications to linear mixed models
3. The penalized least squares problem
4. Using the sparse Cholesky for mixed models
5. Evaluating the likelihood

Outline

1. Model matrices with many columns
2. Applications to linear mixed models
3. The penalized least squares problem
4. Using the sparse Cholesky for mixed models
5. Evaluating the likelihood

Outline

1. Model matrices with many columns
2. Applications to linear mixed models
3. The penalized least squares problem
4. Using the sparse Cholesky for mixed models
5. Evaluating the likelihood

Model matrices and sparsity

- In statistical models the effects of the covariates on the response are often expressed through *model matrices*.
- A common idiom in a model fitting function using a `formula` argument is a call to `model.frame()` followed by a call to `model.matrix()`.
- Many users feel frustrated that R does not transparently handle very large model matrices, failing to realize that a naive decomposition of an $n \times p$ dense model matrix requires np^2 flops. Large values of p are particularly problematic.
- Frequently large values of p are a consequence of incorporating factors with a large number of levels in the model. A factor with k levels generates at least $k - 1$ columns as do any interactions with such a factor.
- The model matrix columns are generated from the indicator columns for the factor, which are very sparse. The greater the number of levels, the more sparse the indicators become.

Sparse model matrices and regularization

- As stated at useR!2009, large, sparse model matrices usually require some amount of regularization for computationally feasible evaluation of coefficients and fitted values.
- Frequently the regularization parameter(s) is(are) chosen to optimize a criterion, requiring evaluation of the criterion for many different trial values of the regularization parameter(s).
- Usually the repeated evaluations of the criterion require decomposition of a matrix with a constant structure (including the positions of the non-zeros) and varying numeric values.
- The sparse Cholesky factorization is ideally suited to problems requiring many evaluations of a decomposition of a matrix with constant structure and varying numeric values.

The sparse Choelsky factorization

- The `Matrix` package for R provides sparse matrix methods, including the sparse Cholesky, by interfacting to Tim Davis' CHOLMOD library of C functions.
- This C library provides separate functions for the symbolic factorization, including determining a *fill-reducing permutation*, and the numeric factorization.
- The symbolic factorization determines the positions of the non-zeros in the result. The numeric factorization simply evaluates the numeric values. Generally it is much faster than the symbolic factorization.
- There are many beautiful mathermatical results associated with sparse matrix operations. See Tim Davis' 2007 SIAM book for some of these results.

Variations of the sparse Cholesky

- In the Matrix package we use the formulation from the CHOLMOD C library. Sparse matrices may be entered in the triplet formulation but operations are usually performed on the *compressed-column representation* (the CsparseMatrix class).
- If $\boldsymbol{A}$ is a positive-definite symmetric sparse matrix, the sparse Cholesky factorization consists of a permutation matrix $\boldsymbol{P}$ and a lower triangular matrix $\boldsymbol{L}$ such that

  $$\boldsymbol{L}\boldsymbol{L}^{\intercal} = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^{\intercal}.$$

  Note that $\boldsymbol{L}$ is the left factor (statisticians often consider the the right factor, $\boldsymbol{R} = \boldsymbol{L}^{\intercal}$). The permutation $\boldsymbol{P}$ is stored (as a vector) within the factorization.
- There are two variations: the *LDL* factorization, where the lhs is $\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\intercal}$ ($\boldsymbol{L}$ unit lower triangular; $\boldsymbol{D}$ diagonal), and a *supernodal* $\boldsymbol{L}\boldsymbol{L}^{\intercal}$ decomposition, which is a sparse/dense hybrid that collapses columns with similar structure to a "supernode" of the graph representation.

Outline

1. Model matrices with many columns

2. Applications to linear mixed models

3. The penalized least squares problem

4. Using the sparse Cholesky for mixed models

5. Evaluating the likelihood

Definition of linear mixed models

- A linear mixed model consists of two random variables: the n-dimensional response, $\boldsymbol{\mathcal{Y}}$, and the q-dimensional random effects, $\boldsymbol{\mathcal{B}}$. We observe the value, $\boldsymbol{y}$, of $\boldsymbol{\mathcal{Y}}$; we do not observe the value of $\boldsymbol{B}$.
- The probability model defines one conditional and one unconditional distribution

  $$(\boldsymbol{\mathcal{Y}}|\boldsymbol{\mathcal{B}}=\boldsymbol{b}) \sim \mathcal{N}\left(\boldsymbol{Z}\boldsymbol{b}+\boldsymbol{X}\boldsymbol{\beta}, \sigma^2\boldsymbol{I}_n\right), \quad \boldsymbol{\mathcal{B}} \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{\Sigma}_\theta\right),$$

  which depend on parameters $\boldsymbol{\beta}$, $\boldsymbol{\theta}$ and σ.
- Although the dimension of $\boldsymbol{\Sigma}_\theta$ can be huge, the dimension of the *variance-component parameter vector*, $\boldsymbol{\theta}$, is usually very small.
- The model specification determines the $n \times q$ model matrix $\boldsymbol{Z}$ (generated from indicator columns and typically very sparse), the $n \times p$ model matrix $\boldsymbol{X}$, and the way in which $\boldsymbol{\theta}$ generates $\boldsymbol{\Sigma}_\theta$.

Properties of Σ_θ; generating it

- Because it is a variance-covariance matrix, the $q \times q$ $\boldsymbol{\Sigma}_\theta$ 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 $\boldsymbol{\Sigma}_\theta$.
- We never really form $\boldsymbol{\Sigma}_\theta$; we always work with the *relative covariance factor*, $\boldsymbol{\Lambda}_\theta$, defined so that

  $$\boldsymbol{\Sigma}_\theta = \sigma^2 \boldsymbol{\Lambda}_\theta \boldsymbol{\Lambda}_\theta^{\mathsf{T}}$$

  where σ^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

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

- The linear predictor expression becomes

  $$\boldsymbol{Z}\boldsymbol{b} + \boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{Z}\boldsymbol{\Lambda}_\theta\, \boldsymbol{u} + \boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{U}_\theta\, \boldsymbol{u} + \boldsymbol{X}\boldsymbol{\beta}$$

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

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

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

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

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

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

Outline

1. Model matrices with many columns

2. Applications to linear mixed models

3. The penalized least squares problem

4. Using the sparse Cholesky for mixed models

5. Evaluating the likelihood

Minimizing a penalized sum of squared residuals

- An expression like $\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{U}_\theta\,\boldsymbol{u}\|^2 + \|\boldsymbol{u}\|^2$ is called a *penalized sum of squared residuals* because $\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta} - \boldsymbol{U}_\theta\,\boldsymbol{u}\|^2$ is a sum of squared residuals and $\|\boldsymbol{u}\|^2$ is a penalty on the size of the vector $\boldsymbol{u}$.
- Determining $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}$ 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

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

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

$$(\boldsymbol{U}_\theta \boldsymbol{U}_\theta^\intercal + \boldsymbol{I}_q)\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}} = \boldsymbol{U}_\theta^\intercal(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}) = \boldsymbol{\Lambda}_\theta^\intercal(\boldsymbol{Z}\boldsymbol{y} - \boldsymbol{Z}\boldsymbol{X}\boldsymbol{\beta})$$

- This would be interesting but not very important were it not for the fact that we actually can solve that system for $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}$ even when its dimension, q, is very, very large.
- Recall that $\boldsymbol{U}_\theta = \boldsymbol{Z}\boldsymbol{\Lambda}_\theta$. Because $\boldsymbol{Z}$ is generated from indicator columns for the grouping factors, it is sparse. $\boldsymbol{U}_\theta$ is also very sparse.
- The fill-reducing permutation $\boldsymbol{P}$ and the structure of the Cholesky factor $\boldsymbol{L}$ are determined from $\boldsymbol{U}_{\theta^{(0)}}$ where $\boldsymbol{\theta}^{(0)}$ is the starting value. For subsequent values of $\boldsymbol{\theta}$ the update of the factor $\boldsymbol{L}_\theta$ satisfying

$$\boldsymbol{L}_\theta \boldsymbol{L}_\theta^\intercal = \boldsymbol{P}\left(\boldsymbol{U}_\theta^\intercal \boldsymbol{U}_\theta + \boldsymbol{I}_q\right)\boldsymbol{P}^\intercal$$

is direct from $\boldsymbol{U}_\theta$. (One of the CHOLMOD functions does the update, including virtually adding a multiple of the identity, from the sparse, rectangular $\boldsymbol{U}_\theta$.) From $\boldsymbol{L}_\theta$ we solve for $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}$.

Outline

1 Model matrices with many columns

2 Applications to linear mixed models

3 The penalized least squares problem

4 Using the sparse Cholesky for mixed models

5 Evaluating the likelihood

Applications to models with simple, scalar random effects

- For a model with simple, scalar random-effects terms only, the matrix $\boldsymbol{\Sigma}_\theta$ is block-diagonal in k blocks and the ith block is $\sigma_i^2 \boldsymbol{I}_{n_i}$ where n_i is the number of levels in the ith grouping factor.
- The matrix $\boldsymbol{\Lambda}_\theta$ is also block-diagonal with the ith 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 $\boldsymbol{U}_\theta$ then $\boldsymbol{L}_\theta$, 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 $\boldsymbol{\Lambda}_\theta$
`Ut`	transpose of $\boldsymbol{U}_\theta = \boldsymbol{Z}\boldsymbol{\Lambda}_\theta$

Cholesky factor for the Penicillin model

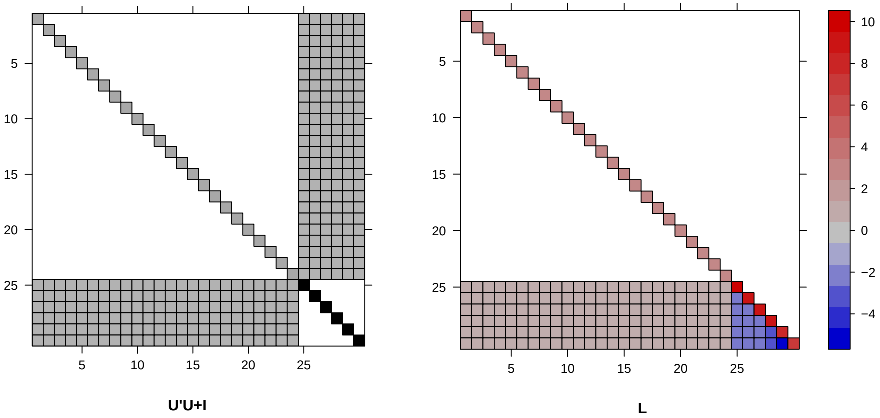
```
> flist <- subset(Penicillin, select = c(plate, sample))
> Zt <- do.call(rBind, lapply(flist, as, "sparseMatrix"))
> (nlev <- sapply(flist, function(f) length(levels(factor(f)))))

 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 $U^\intercal U + I$ and L



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

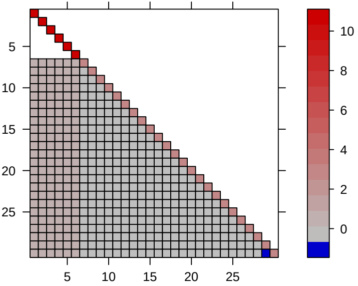
Reversing the order of the factors

- 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

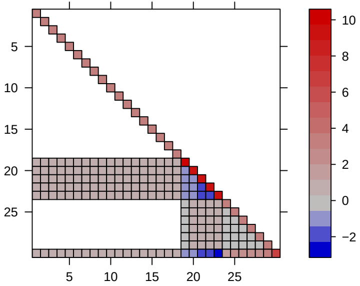
```
> Zt <- do.call(rBind, lapply(flist[2:1], as, "sparseMatrix"))
> Lambda <- Diagonal(x = rep.int(theta[2:1], nlev[2:1]))
> Ut <- crossprod(Lambda, Zt)
> Lnoperm <- Cholesky(tcrossprod(Ut), perm = FALSE, LDL = FALSE,
+     Imult = 1)
> Lperm <- Cholesky(tcrossprod(Ut), LDL = FALSE, Imult = 1)
> sapply(lapply(list(L,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

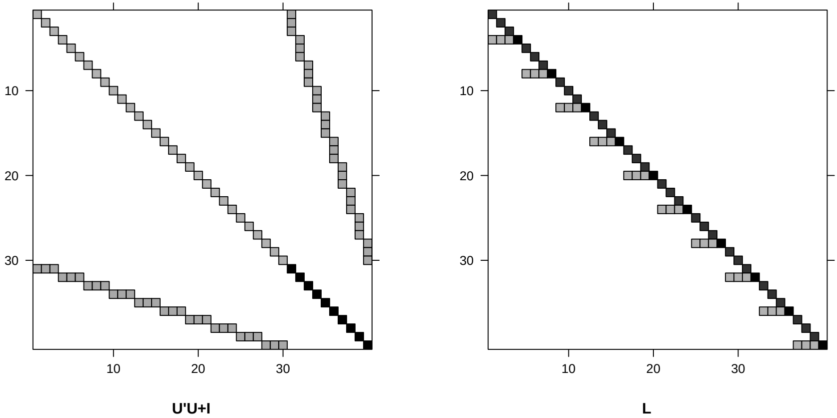
Cholesky factor for the Pastes data

- 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

1. Model matrices with many columns

2. Applications to linear mixed models

3. The penalized least squares problem

4. Using the sparse Cholesky for mixed models

5. Evaluating the likelihood

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

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

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

  we almost have $f_{\mathcal{U}|\mathcal{Y}}$. 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, $\boldsymbol{L}_\theta$ is the key to doing this because

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

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

  1. Solve $\boldsymbol{L}_\theta\boldsymbol{c}_{\boldsymbol{u}} = \boldsymbol{P}\boldsymbol{U}_\theta^\intercal(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})$ for $\boldsymbol{c}_{\boldsymbol{u}}$.
  2. Solve $\boldsymbol{L}_\theta^\intercal\boldsymbol{P}\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}} = \boldsymbol{c}_{\boldsymbol{u}}$ for $\boldsymbol{P}\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}$. Evaluate $\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}} = \boldsymbol{P}^\intercal\boldsymbol{P}\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}}$.

Evaluating the likelihood

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

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

  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 $\boldsymbol{v}=\boldsymbol{L}^{\intercal}\boldsymbol{P}(\boldsymbol{u}-\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}})$, with $d\boldsymbol{v}=\mathrm{abs}(|\boldsymbol{L}||\boldsymbol{P}|)\,d\boldsymbol{u}$, in

$$\int\frac{\exp\left(\frac{-\|\boldsymbol{L}^{\intercal}\boldsymbol{P}(\boldsymbol{u}-\boldsymbol{\mu}_{\mathcal{U}|\mathcal{Y}})\|^2}{2\sigma^2}\right)}{(2\pi\sigma^2)^{q/2}}\,d\boldsymbol{u}$$

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

  because $\mathrm{abs}\,|\boldsymbol{P}|=1$ and $\mathrm{abs}\,|\boldsymbol{L}|$, 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

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

- We wish to minimize the deviance. Its dependence on σ 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*

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

- 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, $\widehat{\boldsymbol{\beta}}_\theta$, by extending the PLS problem to

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

with the solution satisfying the equations

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

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

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

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 $\boldsymbol{L}\boldsymbol{L}^\intercal = \boldsymbol{P}(\boldsymbol{U}^\intercal\boldsymbol{U} + \boldsymbol{I})\boldsymbol{P}^\intercal$ and $\boldsymbol{c_u}$, the solution to $\boldsymbol{L}\boldsymbol{c_u} = \boldsymbol{P}\boldsymbol{U}^\intercal\boldsymbol{y}$.
- 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

$$\begin{aligned}
\boldsymbol{L}\boldsymbol{R}_{ZX} &= \boldsymbol{P}\boldsymbol{U}^\intercal\boldsymbol{X}\\
\boldsymbol{R}_X^\intercal\boldsymbol{R}_X &= \boldsymbol{X}^\intercal\boldsymbol{X} - \boldsymbol{R}_{ZX}^\intercal\boldsymbol{R}_{ZX}\\
\boldsymbol{R}_X^\intercal\boldsymbol{c_\beta} &= \boldsymbol{X}^\intercal\boldsymbol{y} - \boldsymbol{R}_{ZX}^\intercal\boldsymbol{c_u}
\end{aligned}$$

so that

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

Solving the extended PLS problem (cont'd)

- Finally we solve

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

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

- 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 the environment of `setPars()` contains `y`, `X`, `Zt`, `REML`, `L`, `nlev` and `XtX` ($\boldsymbol{X}^\intercal\boldsymbol{X}$).

Code for evaluating the profiled deviance

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

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 Λ_θ 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 $\boldsymbol{\theta}$.