STA 141C - Big Data & High Performance Statistical Computing **Spring 2022**

Week 4-2: Examples using Cholesky decomposition

Lecturer: Bo Y.-C. Ning *April 21, 2022*

Disclaimer: *My notes may contain errors, distribution outside this class is allowed only with the permission of the Instructor.*

Announcement

- Team members assigned

Last time

- Pivoting
- Cholesky decomposition

Today

- Examples using Cholesky

1 Examples using Cholesky decomposition

Notes about Cholesky:

Python NumPy and SciPy have two incompatible implementations of the Cholesky decomposition: scipy.linalg.cholesky and numpy.linalg.cholesky. NumPy version numpy.linalg.cholesky uses the definition used here while scipy.linalg.cholesky returns by default the upper-triangular matrix $U = L'$ scipy.linalg.cholesky can be made to return L using scipy.linalg.cholesky(..., lower=True)

1.1 Evaluating a multivariate normal log-likelihood function

Consider the multivariate normal density you learned in statistical methods course

$$y \sim \mathcal{N}(\mu, \Sigma), \quad \mu \in \mathbb{R}^n, \quad \Sigma \in \mathbb{R}^{n \times n}$$

Assuming Σ is p.d., the log-likelihood function is given by

$$-\frac{n}{2}\log(2\pi) - \frac{\log(det(\Sigma))}{2} - \frac{1}{2}(y-\mu)'\Sigma^{-1}(y-\mu).$$

Here are some questions to consider:

- Suppose we know μ and Σ, how do you evaluate the red part? (thinking about taking inverse?) Whenever we see matrix inverse, we should think in terms of solving linear equations. Consider using Cholesky decomposition.
- How about the blue part? Using the fact $\log(det(\Sigma)) = 2\sum_{k=1}^{n} \log L_{kk}$, why?

Therefore, consider Method 1 (for someone does not take this class): Compute Σ^{-1} → Compute quadratic form → Compute determinant

Consider Method 2 using Cholesky: 1) Do Cholesky decomposition $\Sigma = LL'$, 2) solve $x : Lx = y - \mu$ by forward substitution, 3) compute quadratic form $x'x$; 4) Compute determinant from Cholesky factor. How many flops for Method 1 & 2?

- Method 1: 1) $2n^3/3$ flops; 2) $O(n^2)$ flops; 3) $2n^3/3 + O(n^2)$ flops $4n^3/3 + O(n^2) = \mathcal{O}(4n^3/3)$
- Method 2: 1) $n^3/3$ flops; 2) n^2 flops; 3) $O(n)$ flops; 4) $O(n)$ flops $n^3/3 + n^2 + O(n) = \mathcal{O}(n^3/3)$

Note: the forward substitution can be solve using

```
import scipy.linalg as slg
z = slg.solve_triangular(L, x-mu, lower=True)
```

```
z = forwardsolve(L, x-mu, lower=True)
```

Another function is backsolve.

Example `https://tomroth.com.au/decomp/`

1.2 Linear regression by Cholesky

The goal is to solve

$$X'X\beta = X'y$$

Assume $X \in \mathbb{R}^{n\times p}$ is a full column rank matrix. It is easy to work with the **augmented matrix**

$$\begin{pmatrix} X'X & X'y \\ y'X & y'y \end{pmatrix} = \begin{pmatrix} L & 0 \\ \ell' & d \end{pmatrix} \times \begin{pmatrix} L' & \ell \\ 0' & d \end{pmatrix} = \begin{pmatrix} LL' & L\ell \\ \ell' L' & \ell'\ell + d^2 \end{pmatrix}$$

By Cholesky, $X'X\beta = LL'\beta = X'y = L\ell$ and $L'\beta = \ell$. This solves β in p^2 flops.

Next, since $\ell = L^{-1}X'y$, we have

$$\ell'\ell = yX(LL')^{-1}X'y = y'X(X'X)^{-1}X'y = y'P_x y = y'P_x P_x y = \hat{y}'\hat{y}$$

and

$$d^2 = y'y - \ell'\ell = y'(I - P_x)y = (y - \hat{y})'(y - \hat{y}) = SSE$$

We thus solved β, $\hat{y}$, and SSE all altogether. In R: chol2inv() and in python: cho_solve()

Some notes: We only do inversion if standard errors are needed, $(X'X)^{-1} = (LL')^{-1} = (L^{-1})'L^{-1}$

To summarize, how many flops for using Cholesky to solve linear regression?

- Form the lower triangular part of $(X, y)'(X, y)$: $n(p + 1)^2/2$ flops (why?)

- Cholesky decomposition of the augmented matrix, $(p+1)^3/3$ flops
- Solve $L'\beta = l$ for $\hat{\beta}$: p^2 flops
- If need the standard error, estimate $\hat{\sigma} = d^2/(n-p)$ and compute $\hat{\sigma}^2(X'X)^{-1} = \hat{\sigma}^2(LL')^{-1}$: $2p^3/3$ flops

When n, p are large, total cost is $O(p^3/3) + O(np^2/2)$ flops (without s.e.) and $O(p^3) + O(np^2/2)$ flops (with s.e.).

What about using LU? The total flops is $2p^3/3 + p^2 + np^2/2$ (without s.e.) and $> 2p^3 + np^2/2$ (with s.e.)