Appendix: Derivations

Sophie Li

May 2025

1 Introduction

Hi! These are some motivations and derivations I write up as I develop on my project. No definite structure to this, but I think it's important as a mathematician and researcher to understand the theory behind what I implement in code.

2 Cholesky Decomposition

The Cholesky decomposition is a computational linear algebra technique. We have $C, L \in M_{n\times n}(R)$, with entries indexed as such:

$$C = \begin{bmatrix} C_{0,0} & \cdots & C_{0,n-1} \\ \vdots & \ddots & \vdots \\ C_{n-1,0} & \cdots & C_{n-1,n-1} \end{bmatrix}, L = \begin{bmatrix} L_{0,0} & \cdots & L_{0,n-1} \\ \vdots & \ddots & \vdots \\ L_{n-1,0} & \cdots & L_{n-1,n-1} \end{bmatrix}$$

We seek a lower triangular matrix L such that

$$C = LL^T$$

Since L is lower-triangular, any entry l_{ij} with $i < j$ is 0. So we can write it as

$$L = \begin{bmatrix} L_{0,0} & 0 & \cdots & 0 \\ L_{1,0} & L_{1,1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n-1,0} & L_{n-1,1} & \cdots & L_{n-1,n-1} \end{bmatrix}$$

Things get weird when entries are complex. For simplicity, I'll assume the decomposition matrix L is real-valued. Note, this is guaranteed when we C is a symmetric positive-definite matrice (see section 3).

2.1 Motivation on Gaussian Distributions

For motivation, I am using this technique in the context of sampling a d-dimensional multivariate Gaussian distribution, whose parameters are the mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ and the covariance matrix

$C \in \mathbb{R}^{d \times d}$. We seek to sample a random vector $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, C)$. Doing this directly is difficult since the correlation structure can be complicated. Instead, we can define a variable $\mathbf{U}$ that follows the standard normal, $\mathbf{U} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. Let L be the matrix resulting from the Cholesky Decomposition of C. It turns out, if we sample $\mathbf{U}$ and set

$$\mathbf{X} = \boldsymbol{\mu} + L\mathbf{U}$$

then $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, C)$ as desired. I believe this is actually how Pytorch does it internally.

It's worth proving this fact. Doing so definitely helped solidify the intuition for me since the multivariate case is highly non-obvious at first glance. We write out $\boldsymbol{\mu} = (\mu_0, .., \mu_{d-1}), \mathbf{U} = (U_0, ..., U_{d-1})$. Expanding out $\mathbf{X} = \boldsymbol{\mu} + L\mathbf{U}$ yields

$$X_0 = \mu_0 + L_{00}U_0$$
$$X_1 = \mu_2 + L_{10}U_0 + L_{11}U_1$$
$$\dots$$

Consider the general

$$X_k = \mu_k + L_{k0}U_0 + L_{k1}U_1 + ... + L_{kk}U_k$$

Applying linearity of expectation,

$$E[X_k] = \mu_k + L_{k0}E[U_0] + ... + L_{kk}E[U_k]$$

By assumption, each U_0 follows the standard normal, which is centered at 0. Thus, we've shown $E[X_k] = \mu_k$. Also, it's known that the sum of normally distributed random variables is normal, so X_k is normally distributed. No worries there!

It remains to prove the desired covariance structure. Let us take $i, j < d$ and without loss of generality, let $i \leq j$.

$$Cov(X_i, X_j) = Cov(L_{i0}U_0 + L_{i1}U_1 + ... + L_{ii}U_i, L_{j0}U_0 + L_{j1}U_1 + ... + L_{jj}U_j)$$

Applying bi-linearity, we get a nasty looking expression:

$$\sum_{k=0}^{i}[Cov(L_{ik}U_k, L_{j0}U_0) + Cov(L_{ik}U_k, L_{j1}U_1) + ... + Cov(L_{ik}U_k, L_{jj}U_j)]$$

$$\sum_{k=0}^{i}[L_{ik}L_{j0}Cov(U_k, U_0) + L_{ik}L_{j1}Cov(U_k, U_1) + ... + L_{ik}L_{jj}Cov(U_k, U_j)]$$

Thankfully, all terms here except $Cov(U_k, U_k) = 1$ equal 0, so we get

$$\sum_{k=0}^{i} L_{ik}L_{jk}Cov(U_k, U_k) = \sum_{k=0}^{i} L_{ik}L_{jk}$$

This shows that

$$Cov(X_i, X_j) = \sum_{k=0}^{i} L_{ik}L_{jk}$$

Now, going back to our matrix $C = LL^T$.

$$C_{ij} = \sum_{k=0}^{n-1} L_{ik} L_{kj}^T = \sum_{k=0}^{n-1} L_{ik} L_{jk}$$

In other words, the (i, j)th entry of C is just the dot product of row i and row j of matrix L, whose entries we must solve for. Once $k > i$, then $L_{ik} = 0$ and analogously for j. We re-write the above sum as

$$C_{ij} = \sum_{k=0}^{min(i,j)} L_{ik} L_{jk}$$

Here, we assumed $i \leq j$ so indeed,

$$C_{ij} = \sum_{k=0}^{i} L_{ik} L_{jk} = Cov(X_i, X_j)$$

proving that the covariance matrix is indeed correct! This shows that in the multi-variate case, any Gaussian can be reduced to sampling from the standard form.

The takeaway here is that intuitively, the L matrix makes the matrix multiplication work in really similar way that Covariance "splits" across terms. Therefore the structure is preserved. The explicit density function can be written as such (but it's really nasty):

$$\frac{1}{(2\pi)^{d/2}} (\det C)^{-1/2} \exp\{-(1/2)(\mathbf{x} - \boldsymbol{\mu}) C^{-1} (\mathbf{x} - \boldsymbol{\mu})^T\}$$

Anyways, onto the actual derivation of Cholesky. We have C, L defined as above. Recall we derived

$$C_{ij} = \sum_{k=0}^{min(i,j)} L_{ik} L_{jk}$$

Since $C^T = (LL^T)^T = (L^T)^T L^T = LL^T = C$, then C is symmetric. Without loss of generality, assume $i \leq j$.

Case 1 (Equality): $i = j$:

$$C_{00} = L_{00}^2$$

$$C_{11} = L_{00}^2 + L_{11}^2$$

$$\dots$$

$$C_{(n-1),(n-1)} = \sum_{i=0}^{n-1} L_{ii}^2$$

We easily solve, always taking the positive root, to get

$$L_{00} = \sqrt{C_{00}}$$

$$L_{11} = \sqrt{C_{11} - C_{00}}$$

$$\dots$$

$$L_{kk} = \sqrt{C_{kk} - C_{(k-1),(k-1)}}$$

This initializes all n diagonal entries of L.
Case 2 (Strict Inequality): $i < j$

$$C_{ij} = \sum_{k=0}^{i} L_{ik}L_{jk} = \sum_{k=0}^{i-1} L_{ik}L_{jk} + L_{ii}L_{ji}$$

We isolate for L_{ji}:

$$L_{ji} = \frac{1}{L_{ii}}(C_{ij} - \sum_{k=0}^{i-1} L_{ik}L_{jk})$$

This expresses L_{ji} in terms of $C_{i,j}$ and entries L_{km}, where $k \leq j, m \leq i$. This actually gets us the numerical solution – not in closed form, but in dynamic-programming style.

We go row by row. First, we set $L_{00} = \sqrt{C_{00}}$. Then, we can set L_{10}, L_{11} using the formula above. L_{10} only relies on L_{00}, and then once we get that value, L_{11} only relies on L_{00}, L_{10}. We go to the next row, and so forth, until the last entry we solve is $L_{(n-1),(n-1)}$ giving us the entire L matrix!

3 Positive Semi-definiteness

As mentioned before, the Cholesky Decomposition requires a matrix that is positive semidefinite. Fortunately, the covariance matrix turns out to have this property so we don't need to check separately. In this section, I first give some definitions of positive semi-definite and prove that they are equivalent.
I then show that the covariance matrix is positive semi-definite (abbrev: pos semi-def).