CS229 Lecture notes

原作者：[Andrew Ng](http://cs229.stanford.edu/)（[吴恩达](http://open.163.com/movie/2008/1/M/C/M6SGF6VB4_M6SGHFBMC.html)）

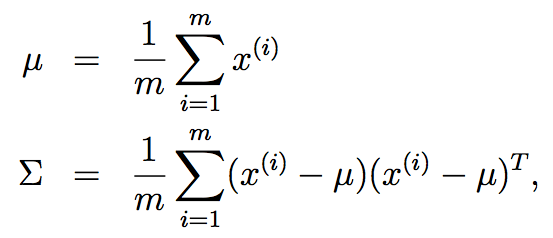
翻译：[CycleUser](https://zhuanlan.zhihu.com/python-kivy)

Part IX

# 因子分析（Factor analysis）

如果有一个多个高斯模型混合（a mixture of several Gaussians）而来的数据集 x(i) ∈ Rn ，那么就可以用期望最大化算法（EM algorithm）来对这个混合模型（mixture model）进行拟合。这种情况下，对于有充足数据（sufficient data）的问题，我们通常假设可以从数据中识别出多个高斯模型结构（multiple-Gaussian structure）。例如，如果我们的训练样本集合规模（training set size） m 远远大于（significantly larger than）数据的维度（dimension） n，就符合这种情况。

然后来考虑一下反过来的情况，也就是 n 远远大于 m，即 n ≫ m。在这样的问题中，就可能用单独一个高斯模型来对数据建模都很难，更不用说多个高斯模型的混合模型了。由于 m 个数据点所张开（span）的只是一个 n 维空间 Rn 的低维度子空间（low-dimensional subspace），如果用高斯模型（Gaussian）对数据进行建模，然后还是用常规的最大似然估计（usual maximum likelihood estimators）来估计（estimate）平均值（mean）和方差（covariance），得到的则是：



we would find that the matrix Σ is singular. This means that Σ−1 does not exist, and 1/|Σ|1/2 = 1/0. But both of these terms are needed in computing the usual density of a multivariate Gaussian distribution. Another way of stating this difficulty is that maximum likelihood estimates of the parameters result in a Gaussian that places all of its probability in the affine space spanned by the data,1 and this corresponds to a singular covariance matrix.

我们会发现这里的 Σ 是一个奇异（singular）矩阵。这也就意味着其逆矩阵 Σ−1 不存在，而 1/|Σ|1/2 = 1/0。 但这几个变量都还是需要的，要用来计算一个多元高斯分布（multivariate Gaussian distribution）的常规密度函数（usual density）。还可以用另外一种方法来讲述清楚这个难题，也就是对参数（parameters）的最大似然估计（maximum likelihood estimates）会产生一个高斯分布（Gaussian），其概率分布在由样本数据所张成的仿射空间（affine space）中，对应着一个奇异的协方差矩阵（singular covariance matrix）。

1This is the set of points x satisfying x = Σ mi=1 αix(i), for some αi’s so that Σ mi=1 α1 = 1.

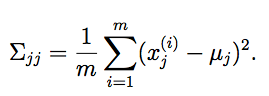
这是一个点集，对于某些 αi，此集合中的点 x 都满足 x = Σ mi=1 αix(i), 因此 Σ mi=1 α1 = 1。

通常情况下，除非 m 比 n 大出相当多（some reasonable amount），否则最大似然估计（maximum likelihood estimates）得到的均值（mean）和方差（covariance）都会很差（quite poor）。尽管如此，我们还是希望能用已有的数据，拟合出一个合理（reasonable）的高斯模型（Gaussian model），而且还希望能识别出数据中的某些有意义的协方差结构（covariance structure）。那这可怎么办呢？

在接下来的这一部分内容里，我们首先回顾一下对 Σ 的两个可能的约束（possible restrictions），这两个约束条件能让我们使用小规模数据来拟合 Σ，但都不能就我们的问题给出让人满意的解（satisfactory solution）。然后接下来我们要讨论一下高斯模型的一些特点，这些后面会用得上，具体来说也就是如何找到高斯模型的边界和条件分布。最后，我们会讲一下因子分析模型（factor analysis model），以及对应的期望最大化算法（EM algorithm）。

# 1 Σ 的约束条件（Restriction）

如果我们没有充足的数据来拟合一个完整的协方差矩阵（covariance matrix），就可以对矩阵空间 Σ 给出某些约束条件（restrictions）。例如，我们可以选择去拟合一个对角（diagonal）的协方差矩阵 Σ。这样，读者很容易就能验证这样的一个协方差矩阵的最大似然估计（maximum likelihood estimate）可以由对角矩阵（diagonal matrix） Σ 满足：

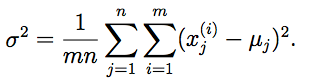


因此，Σjj 就是对数据中第 j 个坐标位置的方差值的经验估计（empirical estimate）。

Recall that the contours of a Gaussian density are ellipses. A diagonal Σ corresponds to a Gaussian where the major axes of these ellipses are axis- aligned.

回忆一下，高斯模型的密度的形状是椭圆形的。对角线矩阵 Σ 对应的就是椭圆长轴（major axes）对齐（axis- aligned）的高斯模型。

有时候，我们还要对这个协方差矩阵（covariance matrix）给出进一步的约束，不仅设为对角的（major axes），还要求所有对角元素（diagonal entries）都相等。这时候，就有 Σ = σ2I，其中 σ2 是我们控制的参数。对这个 σ2 的最大似然估计则为：



这种模型对应的是密度函数为圆形轮廓的高斯模型（在二维空间也就是平面中是圆形，在更高维度当中就是球（spheres）或者超球体（hyperspheres））。

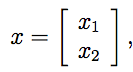
如果我们对数据要拟合一个完整的，不受约束的（unconstrained）协方差矩阵 Σ，就必须满足 m ≥ n + 1，这样才使得对 Σ 的最大似然估计不是奇异矩阵（singular matrix）。在上面提到的两个约束条件之下，只要 m ≥ 2，我们就能获得非奇异的（non-singular） Σ。

然而，讲 Σ 限定为对角矩阵，也就意味着对数据中不同坐标（coordinates）的 xi，xj建模都将是不相关的（uncorrelated），且互相独立（independent）。通常，还是从样本数据里面获得某些有趣的相关信息结构比较好。如果使用上面对 Σ 的某一种约束，就可能没办法获取这些信息了。在本章讲义里面，我们会提到因子分析模型（factor analysis model），这个模型使用的参数比对角矩阵 Σ 更多，而且能从数据中获得某些相关性信息（captures some correlations），但也不能对完整的协方差矩阵（full covariance matrix）进行拟合。

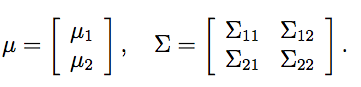
# 2 Marginals and conditionals of Gaussians

Before describing factor analysis, we digress to talk about how to find conditional and marginal distributions of random variables with a joint multivariate Gaussian distribution.

Suppose we have a vector-valued random variable

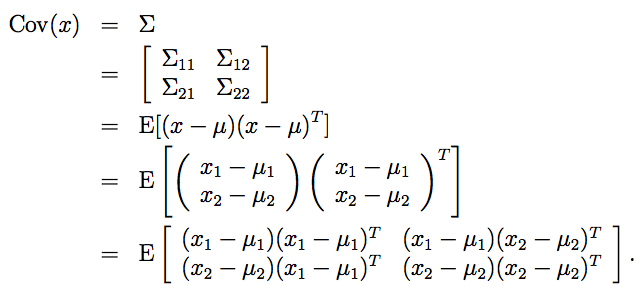


where x1 ∈ Rr, x2 ∈ Rs, and x ∈ Rr+s. Suppose x ∼ N(μ,Σ), where



Here, μ1 ∈ Rr, μ2 ∈ Rs, Σ11 ∈ Rr×r, Σ12 ∈ Rr×s, and so on. Note that since covariance matrices are symmetric, Σ12 = ΣT21.

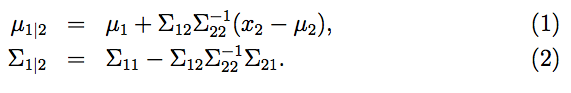
Under our assumptions, x1 and x2 are jointly multivariate Gaussian. What is the marginal distribution of x1? It is not hard to see that E[x1] = μ1, and that Cov(x1) = E[(x1 − μ1)(x1 − μ1)] = Σ11. To see that the latter is true, note that by definition of the joint covariance of x1 and x2, we have that



Matching the upper-left subblocks in the matrices in the second and the last lines above gives the result.

Since marginal distributions of Gaussians are themselves Gaussian, we therefore have that the marginal distribution of x1 is given by x1 ∼ N (μ1, Σ11).

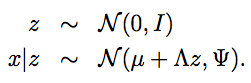
Also, we can ask, what is the conditional distribution of x1 given x2? By referring to the definition of the multivariate Gaussian distribution, it can be shown that x1|x2 ∼ N (μ1|2, Σ1|2), where



When working with the factor analysis model in the next section, these formulas for finding conditional and marginal distributions of Gaussians will be very useful.

# 3 The Factor analysis model

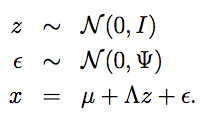
In the factor analysis model, we posit a joint distribution on (x, z) as follows, where z ∈ Rk is a latent random variable:



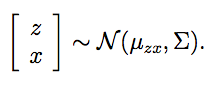
Here, the parameters of our model are the vector μ ∈ Rn, the matrix Λ ∈ Rn×k, and the diagonal matrix Ψ ∈ Rn×n. The value of k is usually chosen to be smaller than n.

Thus, we imagine that each data point x(i) is generated by sampling a k dimension multivariate Gaussian z(i). Then, it is mapped to a k-dimensional affine space of Rn by computing μ+Λz(i). Lastly, x(i) is generated by adding covariance Ψ noise to μ + Λz(i).

Equivalently (convince yourself that this is the case), we can therefore also define the factor analysis model according to

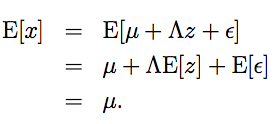


where ε and z are independent. Let’s work out exactly what distribution our model defines. Our random variables z and x have a joint Gaussian distribution

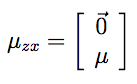


We will now find μzx and Σ.

We know that E[z] = 0, from the fact that z ∼ N(0,I). Also, we have that

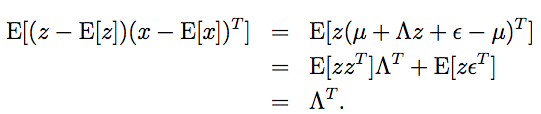


Putting these together, we obtain



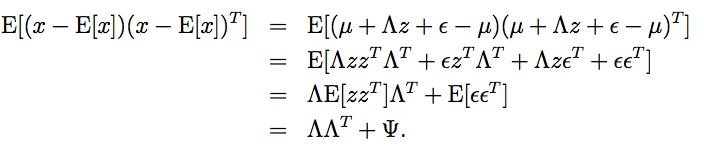
Next, to find, Σ, we need to calculate Σzz = E[(z − E[z])(z − E[z])T ] (the upper-left block of Σ), Σzx = E[(z − E[z])(x − E[x])T ] (upper-right block), and Σxx = E[(x − E[x])(x − E[x])T ] (lower-right block).

Now, since z ∼ N (0, I), we easily find that Σzz = Cov(z) = I. Also,

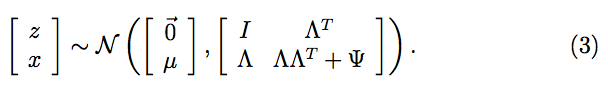


In the last step, we used the fact that E[zzT ] = Cov(z) (since z has zero mean), and E[zεT ] = E[z]E[εT ] = 0 (since z and ε are independent, and hence the expectation of their product is the product of their expectations).

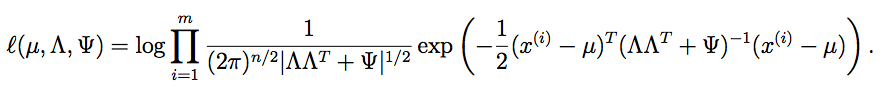
Similarly, we can find Σxx as follows:



Putting everything together, we therefore have that



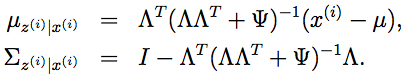
Hence, we also see that the marginal distribution of x is given by x ∼ N(μ,ΛΛT +Ψ). Thus, given a training set {x(i);i = 1,...,m}, we can write down the log likelihood of the parameters:



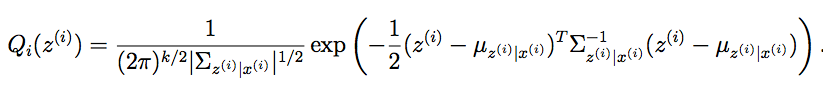
To perform maximum likelihood estimation, we would like to maximize this quantity with respect to the parameters. But maximizing this formula explicitly is hard (try it yourself), and we are aware of no algorithm that does so in closed-form. So, we will instead use to the EM algorithm. In the next section, we derive EM for factor analysis.

# 4 EM for factor analysis

The derivation for the E-step is easy. We need to compute Qi(z(i)) = p(z(i)|x(i); μ, Λ, Ψ). By substituting the distribution given in Equation (3) into the formulas (1-2) used for finding the conditional distribution of a Gaussian, we find that z(i)|x(i); μ, Λ, Ψ ∼ N (μz(i)|x(i) , Σz(i)|x(i) ), where

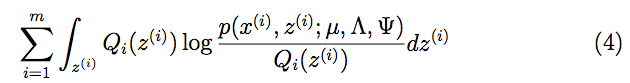


So, using these definitions for μz(i)|x(i) and Σz(i)|x(i), we have



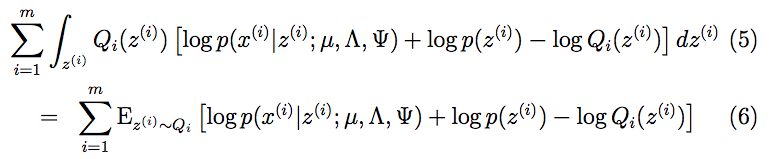
 

Let’s now work out the M-step. Here, we need to maximize

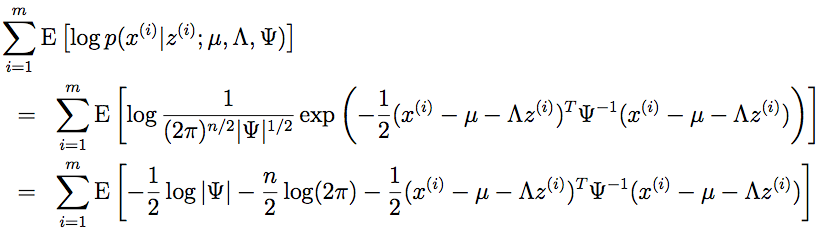


with respect to the parameters μ, Λ, Ψ. We will work out only the optimization with respect to Λ, and leave the derivations of the updates for μ and Ψ as an exercise to the reader.

We can simplify Equation (4) as follows:

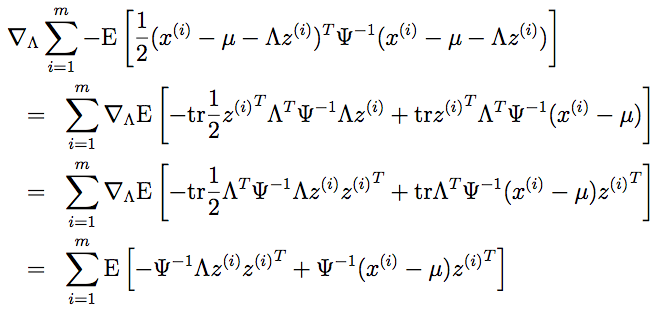


Here, the “z(i) ∼ Qi” subscript indicates that the expectation is with respect to z(i) drawn from Qi. In the subsequent development, we will omit this subscript when there is no risk of ambiguity. Dropping terms that do not depend on the parameters, we find that we need to maximize:

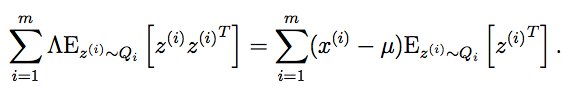


Let’s maximize this with respect to Λ. Only the last term above depends

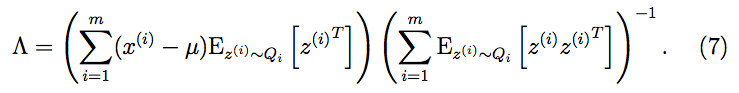
on Λ. Taking derivatives, and using the facts that tr a = a (for a ∈ R), trAB = trBA, and ∇AtrABAT C = CAB + CT AB, we get:



Setting this to zero and simplifying, we get:



Hence, solving for Λ, we obtain

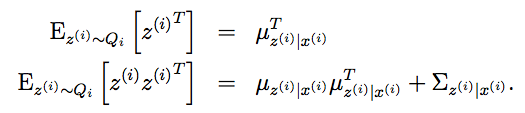


It is interesting to note the close relationship between this equation and the normal equation that we’d derived for least squares regression,

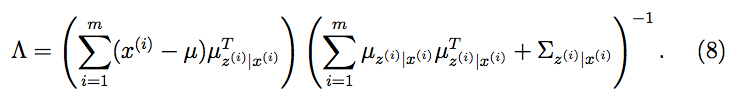


The analogy is that here, the x’s are a linear function of the z’s (plus noise). Given the “guesses” for z that the E-step has found, we will now try to estimate the unknown linearity Λ relating the x’s and z’s. It is therefore no surprise that we obtain something similar to the normal equation. There is, however, one important difference between this and an algorithm that performs least squares using just the “best guesses” of the z’s; we will see this difference shortly.

To complete our M-step update, let’s work out the values of the expectations in Equation (7). From our definition of Qi being Gaussian with mean μz(i)|x(i) and covariance Σz(i)|x(i) , we easily find

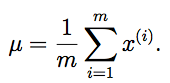


The latter comes from the fact that, for a random variable Y , Cov(Y ) = E[Y Y T ]−E[Y ]E[Y ]T , and hence E[Y Y T ] = E[Y ]E[Y ]T +Cov(Y ). Substituting this back into Equation (7), we get the M-step update for Λ:



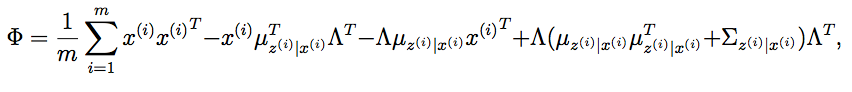
It is important to note the presence of the Σz(i)|x(i) on the right-hand side of this equation. This is the covariance in the posterior distribution p(z(i)|x(i)) of z(i) give x(i), and the M-step must take into account this uncertainty about z(i) in the posterior. A common mistake in deriving EM is to assume that in the E-step, we need to calculate only expectation E[z] of the latent random variable z, and then plug that into the optimization in the M-step everywhere z occurs. While this worked for simple problems such as the mixture of Gaussians, in our derivation for factor analysis, we needed E[zzT ] as well E[z]; and as we saw, E[zzT ] and E[z]E[z]T differ by the quantity Σz|x. Thus, the M-step update must take into account the covariance of z in the posterior distribution p(z(i)|x(i)).

Lastly, we can also find the M-step optimizations for the parameters μ and Ψ. It is not hard to show that the first is given by



Since this doesn’t change as the parameters are varied (i.e., unlike the update for Λ, the right hand side does not depend on Qi(z(i)) = p(z(i)|x(i); μ, Λ, Ψ), which in turn depends on the parameters), this can be calculated just once and needs not be further updated as the algorithm is run. Similarly, the diagonal Ψ can be found by calculating





and setting Ψii = Φii (i.e., letting Ψ be the diagonal matrix containing only the diagonal entries of Φ).