Lecture 8: Classical Factor Analysis and PCA

Francis J. DiTraglia

April 14, 2014

These notes draw on material from Chapters 11–12 of Murphy's *Machine Learning: A Probabilistic Perspective*, Andrew Ng's lecture notes for CS229 at Stanford, and Jolliffe's *Principal Component Analysis*.

1 EM Algorithm

1.1 The Idea behind the EM Algorithm

For simplicity, we'll consider an iid setup for now although the EM can be used in situations with dependence. We'll also suppose that the latent variable is continuous. If it's discrete the idea is exactly the same but the integral is replaced by a sum.

$$\ell(\theta) = \sum_{t=1}^{T} \log p(\mathbf{x}_t; \theta) = \sum_{t=1}^{T} \log \left(\int p(\mathbf{x}_t, \mathbf{z}_t; \theta) \, d\mathbf{z} \right)$$

where $\mathbf{x}_t$ is observed and $\mathbf{z}_t$ is unobserved. In many interesting models there is no explicit formula for the MLE in terms of the marginal density $p(\mathbf{x}_t; \theta)$ but there *is* an explicit formula in terms of the *joint* density $p(\mathbf{x}_t, \mathbf{z}_t; \theta)$. This is exactly the setting in which the EM algorithm is useful. Rather than directly maximizing $\ell(\theta)$, the EM algorithm proceeds *iteratively* over the following two steps:

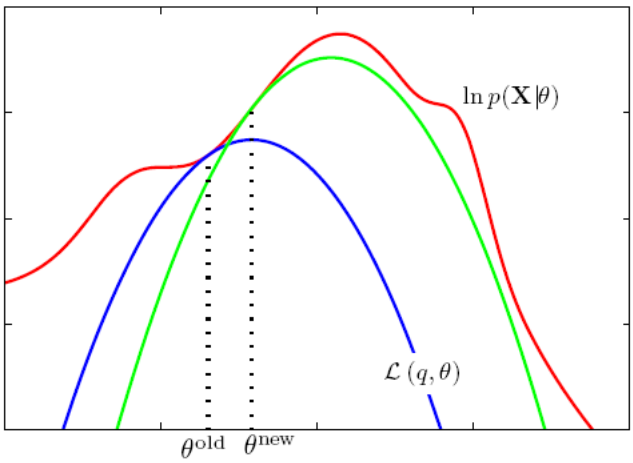

Figure 1: Illustration of the EM Algorithm: to maximize the log likelihood, the red curve, we create a sequence of successive approximations, the blue and green curves, and maximize these. This appears as Figure 9.14 in Bishop's (2006) *Pattern Recognitiion and Machine Learning*.

(E-step) Construct a *lower bound* for $\ell(\theta)$

(M-step) Optimize the lower bound over θ

Roughly speaking, the EM algorithm converts a single complicated optimization problem into a sequence of simple optimization problems. The trick is to ensure that the resulting sequence of estimators converges to the MLE. Jensen's Inequality is the key so I'll briefly remind you of a few importnat facts before proceeding.

1.2 Jensen's Inequality

Recall that a function is called *convex* if its Hessian matrix is positive semi-definite and *strictly convex* if its Hessian matrix is positive definite. For functions of a single variable the condition is $f''(x) \geq 0 \quad \forall x \in \mathbb{R}$ for *convex* and $f''(x) > 0 \quad \forall x \in \mathbb{R}$ for *strictly convex*. In statistics, one of the most useful results concerning convex functions is *Jensen's Inequality*

Proposition 1.1 (Jensen's Inequality)**.** *Let f be a convex function and X be*

a random variable. Then $E[f(X)] \geq f(E[X])$. If f is strictly convex then the inequality is strict unless $P(X = E[X]) = 1$, i.e. X is a constant. For the equivalent results for concave functions, simply reverse the inequality.

1.3 A Lower Bound for the Likelihood

Let $f_t(\mathbf{z}_t)$ be *some arbitrary* density function over the support of $\mathbf{z}_t$, that is any function satisfying $f_t(\mathbf{z_t}) \geq 0$ and

$$\int f_t(\mathbf{z}_t)\, d\mathbf{z}_t = 1$$

We have

$$\begin{aligned} \ell(\theta) = \sum_{t=1}^{T} \log p(\mathbf{x}_t; \theta) &= \sum_{t=1}^{T} \log \left(\int p(\mathbf{x}_t, \mathbf{z}_t; \theta)\, d\mathbf{z}_t \right) \\ &= \sum_{t=1}^{T} \log \left(\int f_t(\mathbf{z}_t) \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t(\mathbf{z}_t)} \right] d\mathbf{z}_t \right) \end{aligned}$$

Now we use Jensen's inequality and the fact that log is a concave function over its domain to find that

$$\log \left(\int f_t(\mathbf{z}_t) \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t(\mathbf{z}_t)} \right] d\mathbf{z}_t \right) \geq \int f_t(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t(\mathbf{z}_t)} \right] d\mathbf{z}_t$$

What's going on here? Since f_t is a *density* the integral inside the parentheses is *an expectation* of a particular function of the argument of integration $\mathbf{z}_t$. The parameter θ and the observed vector of realizations $\mathbf{x}_t$ are constants with respect to the integration. Substituting the preceding inequality into the sum, we have established that

$$\ell(\theta) \geq \sum_{t=1}^{T} \left(\int f_t(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t(\mathbf{z}_t)} \right] d\mathbf{z}_t \right)$$

for *any* density function f_t. This is the *lower bound* for the likelihood that we will use in the E-step. The question is, how should we choose f_t?

The key idea is to turn the *inequality* into an *equality* at a particular value of θ. Intuitively, we want to ensure that, in a given iteration of the algorithm, the actual likelihood and the lower bound *agree* at the value of θ that emerged from the *preceding* iteration. In this way, our sequence of approximating functions will "trace out a path" along the true likelihood, ultimately ensuring that the EM algorithm will converge to the MLE. Since log is in fact *strictly* concave, the only way for Jensen's inequality to hold with equality is if

$$\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t(\mathbf{z}_t)} = c$$

for some constant c that *does not depend* on $\mathbf{z}_t$. The question is, how should we choose f_t to achieve this? Rearranging, integrating, and using the fact that f_t is a density,

$$\begin{aligned} c f_t(\mathbf{z}_t) &= p(\mathbf{x}_t, \mathbf{z}_t; \theta) \\ c \int f_t(\mathbf{z}_t)\, d\mathbf{z}_t &= \int p(\mathbf{x}_t, \mathbf{z}_t; \theta)\, d\mathbf{z}_t \\ c &= p(\mathbf{x}_t; \theta) \end{aligned}$$

Substituting for c, solving for f_t and using the definition of a conditional density we have

$$f_t(\mathbf{z}_t) = \frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{p(\mathbf{x}_t; \theta)} = p(\mathbf{z}_t | \mathbf{x}_t; \theta)$$

In other words, to make the lower bound hold with equality at a particular value of θ, say θ^*, it suffices to set f_t equal to the *conditional* density of $\mathbf{z}_t$ *given* $\mathbf{x}_t$ *evaluated* at θ^*. Crucially this is a both a probability density and a function of $\mathbf{z}_t$ *only* since we plug in the observed value of $\mathbf{x}_t$.

1.4 The Algorithm

In the previous subsection we showed that if we set $f_t(\mathbf{z}_t) = p(\mathbf{z}_t|\mathbf{x}_t;\theta^*)$ then

$$\ell(\theta^*) = \sum_{t=1}^{T}\left(\int f_t(\mathbf{z}_t)\log\left[\frac{p(\mathbf{x}_t,\mathbf{z}_t;\theta^*)}{f_t(\mathbf{z}_t)}\right]d\mathbf{z}_t\right)$$

and, more generally for *any* value of θ

$$\ell(\theta) \geq \sum_{t=1}^{T}\left(\int f_t(\mathbf{z}_t)\log\left[\frac{p(\mathbf{x}_t,\mathbf{z}_t;\theta)}{f_t(\mathbf{z}_t)}\right]d\mathbf{z}_t\right)$$

by Jensen's Inequality. Now we are ready to state the EM algorithm:

Algorithm 1.1 (EM Algorithm)**.** First select a starting value $\theta^{(1)}$. Then repeat the following two steps repeatedly until convergence

(E-step) For each t set $f_t^{(j-1)}(\mathbf{z}_t) = p(\mathbf{z}_t|\mathbf{x}_t;\theta^{(j-1)})$ where $\theta^{(j-1)}$ is the solution from the M-step of the *preceding* iteration.

(M-step) $\theta^{(j)} = \arg\max_{\theta\in\Theta}\sum_{t=1}^{T}\left(\int f_t^{(j-1)}(\mathbf{z}_t)\log\left[\frac{p(\mathbf{x}_t,\mathbf{z}_t;\theta)}{f_t^{(j-1)}(\mathbf{z}_t)}\right]d\mathbf{z}_t\right)$

If $j = 2$ then $\theta^{(j-1)}$ is simply the starting value $\theta^{(1)}$.

Note that in the M-step the argument θ over which we maximize *only* enters the expression $p(\mathbf{x}_t,\mathbf{z}_t;\theta)$. The density $f^{(j-1)}(\mathbf{z}_t)$ does *not* depend on θ, it depends on the *constant* $\theta^{(j-1)}$ that solved the M-step of the *previous iteration*. The amazing thing about the EM algorithm is that it is *guaranteed* to converge to a local maximum of the likelihood function: each successive iteration *monotonically* improves the likelihood as we will see below. This fact along the the way we constructed our lower bound to hold with equality at the value of θ from the *previous* M-step gives us an excellent tool for debugging our code: simply plot

$$\ell(\theta^{(j)}) = \sum_{t=1}^{T}\left(\int f_t^{(j)}(\mathbf{z}_t)\log\left[\frac{p(\mathbf{x}_t,\mathbf{z}_t;\theta^{(j)})}{f_t^{(j)}(\mathbf{z}_t)}\right]d\mathbf{z}_t\right)$$

against j. The preceding expression is the *objective function* from the $(j+1)$th M-step evaulated at the *solution* from the jth M-step. By construction, this If the is equal to the likelihood evaulated at $\theta^{(j)}$. If the plot is *not* increasing monotonically in j, then there must be a bug in your code.

1.5 Why Does the EM Algorithm Converge?

Let $\theta^{(j)}$ and $\theta^{(j+1)}$ be two succesive solutions to the M-step of the EM algorithm. We will now show that $\ell(\theta^{(j)}) \leq \theta^{(j+1)}$. In other words, the EM algorithm *monotonically* improves the likelihood in each iteration. Since $\{\theta^{(j)}\}$ is a monotonic sequence, it converges as long as it is bounded (Rudin Theorem 3.14). Since $\ell(\theta^{(1)})$ is a lower bound, if follows that the EM algorithm is *guaranteed* to converge to a local maximum of the likelihood function provided that the likelihood function is bounded above. All that remains is to actually demonstrate that $\ell(\theta^{(j)}) \leq \theta^{(j+1)}$.

By definition,

$$\theta^{(j+1)} = \arg\max_{\theta \in \Theta} \sum_{t=1}^{T} \left(\int f_t^{(j)}(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t^{(j)}(\mathbf{z}_t)} \right] d\mathbf{z}_t \right)$$

Now let $\tilde{\theta}$ be some arbitrary value of θ. Since $\theta^{(j+1)}$ is the arg max, evaluating the objective function at $\tilde{\theta}$ cannot yield a greater value than evaluating it at $\theta^{(j+1)}$. Since this holds for *any* $\tilde{\theta}$ it holds in particular for $\theta^{(j)}$. Hence,

$$\begin{aligned} \sum_{t=1}^{T} \left(\int f_t^{(j)}(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta^{(j+1)})}{f_t^{(j)}(\mathbf{z}_t)} \right] d\mathbf{z}_t \right) &\geq \sum_{t=1}^{T} \left(\int f_t^{(j)}(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta^{(j)})}{f_t^{(j)}(\mathbf{z}_t)} \right] d\mathbf{z}_t \right) \\ &= \ell(\theta^{(j)}) \end{aligned}$$

since we chose $f_t^{(j)}(\mathbf{z}_t)$ to make Jensen's Inequality strict at $\theta^{(j)}$. Now, recall

from above that for *any density* $f_t(\mathbf{z}_t)$ and *any* value of θ,

$$\ell(\theta) \geq \sum_{t=1}^{T} \left(\int f_t(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t(\mathbf{z}_t)} \right] d\mathbf{z}_t \right)$$

by Jensen's Inequality. Since this holds in general, it also holds in particular for $\theta = \theta^{(j+1)}$ and $f_t(\mathbf{z}_t) = f_t^{(j)}(\mathbf{z}_t)$. Hence,

$$\ell(\theta^{(j+1)}) \geq \sum_{t=1}^{T} \left(\int f_t^{(j)}(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta^{(j+1)})}{f_t^{(j)}(\mathbf{z}_t)} \right] d\mathbf{z}_t \right)$$

Combining the two inequalities gives $\ell(\theta^{(j+1)}) \geq \ell(\theta^{(j)})$ as claimed.

2 Factor Analysis

Before we proceed, I'll just remind you of some key facts about normal distributions and we'll need below.

2.1 Facts about the Multivariate Normal Distribution

2.1.1 Linear Combinations

Suppose that $X \sim N(\mu, \Sigma)$ and $Y = a + BX$ where a is a vector and B a matrix of constants. Then $Y \sim (a + B\mu, B\Sigma B')$.

2.1.2 Marginals and Conditionals

Let X_1 and X_2 be random vectors such that $(X_1', X_2') \sim N(\mu, \Sigma)$ where

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \quad \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$$

Then,

$$\begin{aligned} X_1 &\sim N(\mu_1, \Sigma_{11}) \\ X_2 &\sim N(\mu_2, \Sigma_{22}) \\ X_1|X_2 &\sim N(\mu_{1|2}, \Sigma_{1|2}) \end{aligned}$$

where,

$$\begin{aligned} \mu_{1|2} &= \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2) \\ \Sigma_{1|2} &= \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21} \end{aligned}$$

2.2 The Factor Analysis Model

Classical Factor Analysis specifies a joint distribution on the observable random p-vector X and an unobserved or "latent" random k-vector Z, as follows

$$\begin{aligned} Z &\sim N_k(0_k, \mathbf{I}_k) \\ \epsilon &\sim N_p(0_p, \Psi) \\ Z &\perp \epsilon \\ X &= \mu + \Lambda Z + \epsilon \end{aligned}$$

where μ is a $p \times 1$ vector of parameters, Λ is a $p \times k$ matrix of parameters called the *factor loading matrix*, and Ψ is a $p \times p$ *diagonal* matrix of parameters. Factor Analysis can be viewed as a "low rank parameterization" of a multivariate normal distribution. The idea is that, while X is a random p-vector, its realizations lie *close* to a k-dimensional affine subspace: Λ maps Z from $\mathbb{R}^k$ to a linear subspace of $\mathbb{R}^p$, μ shifts this subspace away from the origin, and ϵ adds axis-aligned Gaussian noise. Hence it makes sense to require that k is strictly less than both p, the dimension of X, and T, the sample size.

The intution is as follows: Factor Analysis "forces" Z to "explain" the correlation structure of X. This is why Ψ is required to be diagonal. The

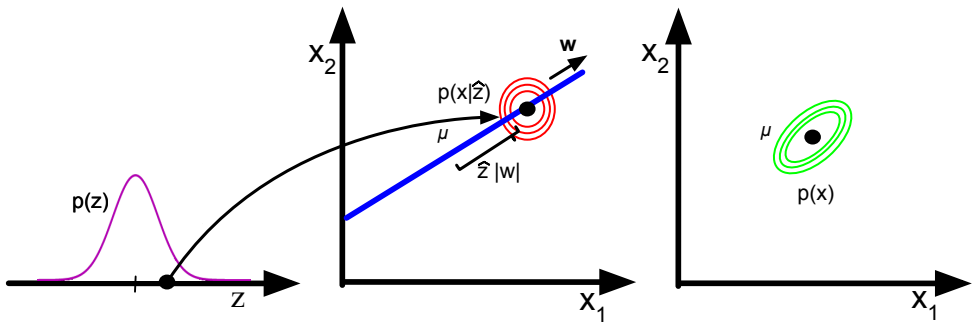

Figure 2: Illustration of Factor Analysis, although the notation is slightly different from mine. (I need to draw my own version of this.) This appears as figure 12.1 in Murphy (2012).

diagonal elements of Ψ are sometimes called the *idiosyncratic variance terms*, since each corresponds to a *single* component of X. This is the key point: *conditional* on the factors Z, the elements of X are *independent*.

The factor analysis model implies that the joint distribution of Z and X is normal. Specifically,

$$\begin{aligned}
\begin{bmatrix} Z \\ X \end{bmatrix} &= \begin{bmatrix} 0_k \\ \mu \end{bmatrix} + \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ \Lambda & I_p \end{bmatrix} \begin{bmatrix} Z \\ \epsilon \end{bmatrix} \\
&= \begin{bmatrix} 0_k \\ \mu \end{bmatrix} + \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ \Lambda & I_p \end{bmatrix} N\left(\begin{bmatrix} 0_k \\ 0_p \end{bmatrix}, \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ 0_{p\times k} & \Psi \end{bmatrix} \right) \\
&\sim N\left(\begin{bmatrix} 0_k \\ \mu \end{bmatrix}, \begin{bmatrix} I & \Lambda' \\ \Lambda & \Lambda\Lambda' + \Psi \end{bmatrix} \right)
\end{aligned}$$

The algebra for the variance matrix calculation is as follows:

$$\begin{aligned} V &= \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ \Lambda & I_p \end{bmatrix} \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ 0_{p\times k} & \Psi \end{bmatrix} \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ \Lambda & I_p \end{bmatrix}' \\ &= \begin{bmatrix} \mathbf{I}_k & 0_{k\times p} \\ \Lambda & \Psi \end{bmatrix} \begin{bmatrix} \mathbf{I}_k & \Lambda' \\ 0_{p\times k} & I_p \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{I}_k & \Lambda' \\ \Lambda & \Lambda\Lambda' + \Psi \end{bmatrix} \end{aligned}$$

2.3 The Factor Analysis Model is Not Identified

Suppose we want to estimated the parameters μ, Λ, Ψ of the factor analysis model. The first natural question is whether this model is even identified. The mean vector μ doesn't provide any problems for identification since we can always demean X before proceeding. Excluding μ, the factor analysis model has $k(p+1)$ free parameters: Λ is a $p \times k$ matrix and Ψ is a *diagonal* $p \times p$ matrix.

Unfortunately the Factor Analysis is not identified as given above. To see why, suppose that R is an orthogonal matrix, i.e. $RR' = R'R = I$. Geometrically, R is a rotation: it leaves the length of any vector v unchanged since

$$||Rv|| = \sqrt{(Rv)'(Rv)} = \sqrt{v'R'Rv} = \sqrt{v'v} = ||v||$$

And it leaves the *distance* between any two vectors v and w unchanged since

$$\begin{aligned} ||Rv - Rw|| &= ||R(v-w)|| = \sqrt{[R(v-w)]'\,[R(v-w)]} \\ &= \sqrt{(v-w)'R'R(v-w)} = \sqrt{(v-w)'(v-w)} = ||v-w|| \end{aligned}$$

From the joint distribution for X and Z that we derived above it follows that the marginal distribution of X is $N(\mu, \Lambda\Lambda' + \Psi)$. Thus if we observe realizations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ of a sequence of iid random vectors $X_1, X_2, \ldots, X_T$ generated

from the Factor Analysis model the log-likelihood is given by

$$\ell(\mu, \Lambda, \Psi) = \log \left[\prod_{t=1}^{T} \frac{\exp\left\{-\frac{1}{2}\left(\mathbf{x}_t - \mu\right)' \left(\Lambda\Lambda' + \Psi\right)^{-1} \left(\mathbf{x}_t - \mu\right)\right\}}{(2\pi)^{p/2} \left|\Lambda\Lambda' + \Psi\right|^{1/2}} \right]$$

Now suppose that we evaluate the log-likelihood at $\widetilde{\Lambda}R$ rather than Λ. Since Λ only enters through the outer product $\Lambda\Lambda'$ the likelihood is *unchanged*:

$$\widetilde{\Lambda}\widetilde{\Lambda}' = (\Lambda R)(\Lambda R)' = \Lambda R R' \Lambda' = \Lambda\Lambda'$$

We have shown that the matrix of factor loadings is *only identified up to a rotation*. Another way to think about this is in terms of the latent variable Z, Since $X = \mu + \Lambda Z + \epsilon$, post-multiplying Λ by R is the same as *pre-multiplying* Z by R. As explained above, this constitutes a *rotation* of the vector Z. But since Z is a *spherical* normal distribution, rotating it cannot change the likelihood.

If we merely plan to use Factor Analysis for *prediction* this lack of identification is irrelevant: it does not affect the predictive performance of the model in any way. If we ultimately hope to *interpret* the latent factors, however the lack of identification becomes problematic. There are various ways to get a unique solution for the factor loadings Λ that involve making various restrictions on the matrix of factor loadings Λ. The first question is: so how many restrictions do we need?

Since the lack of identication comes from rotational invariance, we need to count the number of free parameters in a $k \times k$ rotation matrix. Start with the first column: it has $k - 1$ free parameters since the only constraint is that it have length one. The second column must also have length one, but it has the further restriction that it must be orthogonal to the first column. Hence it has $k - 2$ free parameters. Continuing in this way, we see that there are $(k-1) + (k-2) + \ldots + (k-k+1) = k(k-2)/2$ free parameters in a general $k \times k$ rotation matrix.

There are a number of possible solutions to the lack of identification:

- **Constraining the columns of Λ to be orthonormal** This is essentially how PCA works, as we'll see below.
- **Constraining Λ to be lower triangular** This constraint imposes that the first element of X only depends on the first factor, the second element of X only depends on the first two factors, and so on. In this choice of which variables to list as the first elements of X can make a *big difference*.
- **Imposing Sparsity on Λ** There are a number of proposals for "sparse factor analysis," including using LASSO and imposing an ℓ_1 penalty on the factor loadings. Although this might not completely solve the identification problem, setting many of the elements of Λ to exactly zero can partially resolve it.
- **Choosing an Informative Rotation Matrix** If you read old textbooks on multivariate statistics, you'll see a number of suggestions, including something called the "varimax" method. Typically, these solutions involve some kind of sparsity condition.
- **Use a Non-Gaussian Distribution for the Factors** The lack of identification in the Factor Analysis Model comes from the rotational invariance of a multivariate normal distribution with an identity covariance matrix. Using a distribution other than the normal can partially eliminate this problem: a Laplace distribution, for example, has diamond-shaped contours. This is the idea behind "Independent Components Analysis" (ICA).

2.4 The Latent Factors

The unobserved random variables $Z_1, \ldots, Z_T$ that generate $X_1, \ldots, X_T$ under the Factor Analysis Model are called the *latent factors* or the *latent scores*. In some settings the factor scores are given a particular interpretation and we

may wish to infer them from the observable data. (Warning: interpreting the factors can be very difficult because of the lack of idenfication of the factor model!) Because this model is Gaussian, we can easily work out the conditional distribution of the latent factors using joint distribution of $(Z', X')'$. Indeed, this is precisely what we'll need to do to implement the EM algorithm, as we'll see below.

2.5 EM for Classical Factor Analysis

This is a great problem for the EM algorithm since it can be viewed as a case of missing data: if Z were observed, this would just be a standard multivariate regression problem!

2.5.1 The E-step: Inferring the Latent Factors

In this step we set $f_t^{(j-1)}(\mathbf{z}_t) = p(\mathbf{z}_t|\mathbf{x}_t; \theta^{(j-1)})$ for each t where $\theta^{(j-1)}$ is the value of θ that solved the optimization problem from the *preceding* M-step or, if $j = 2$, the starting value. In the notation of the factor analysis problem we need to calculate:

$$f_t^{(j-1)}(\mathbf{z}_t) = p(\mathbf{z}_t|\mathbf{x}_t; \mu^{(j-1)}, \Lambda^{(j-1)}, \Psi^{(j-1)})$$

As we showed above,

$$\begin{bmatrix} Z \\ X \end{bmatrix} \sim N\left(\begin{bmatrix} 0_k \\ \mu \end{bmatrix}, \begin{bmatrix} I & \Lambda' \\ \Lambda & \Lambda\Lambda' + \Psi \end{bmatrix}\right)$$

Hence, using the properties of the normal distribution reviewed earlier in this document:

$$\begin{aligned} Z|X &\sim N_k(\mu_{Z|X}, \Sigma_{Z|X}) \\ \mu_{Z|X} &= \Lambda'(\Lambda\Lambda' + \Psi)^{-1}(X - \mu) \\ \Sigma_{Z|X} &= \mathbf{I}_k - \Lambda'(\Lambda\Lambda' + \Psi)^{-1}\Lambda \end{aligned}$$

2.5.2 The M-Step: Optimizing the Lower Bound

In this step, we solve

$$\theta^{(j)} = \arg\max_{\theta \in \Theta} \sum_{t=1}^{T} \left(\int f_t^{(j-1)}(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \theta)}{f_t^{(j-1)}(\mathbf{z}_t)} \right] d\mathbf{z}_t \right)$$

Before carrying out the optimization problem, we'll first manipulate the objective function to simplify it and remove constant terms that don't depend on the model parameters. Substituting the notation of the Factor Analysis Model and rearranging, we can write the objective function for the jth M-step as follows:

$$\begin{aligned}
Q^{(j)}(\mu, \Lambda, \Psi) &= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log \left[\frac{p(\mathbf{x}_t, \mathbf{z}_t; \mu, \Lambda, \Psi)}{f_t^{(j-1)}(\mathbf{z}_t)} \right] d\mathbf{z}_t \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \left[\log p(\mathbf{x}_t, \mathbf{z}_t; \mu, \Lambda, \Psi) - \log f_t^{(j-1)}(\mathbf{z}_t) \right] d\mathbf{z}_t \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log p(\mathbf{x}_t, \mathbf{z}_t; \mu, \Lambda, \Psi) \, d\mathbf{z}_t - \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log f_t^{(j-1)}(\mathbf{z}_t) \, d\mathbf{z}_t \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log p(\mathbf{x}_t, \mathbf{z}_t; \mu, \Lambda, \Psi) \, d\mathbf{z}_t + C \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log p(\mathbf{x}_t, \mathbf{z}_t; \mu, \Lambda, \Psi) \, d\mathbf{z}_t - \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log f_t^{(j-1)}(\mathbf{z}_t) \, d\mathbf{z}_t \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log \left[p(\mathbf{x}_t | \mathbf{z}_t; \mu, \Lambda, \Psi) p(\mathbf{z}_t) \right] d\mathbf{z}_t + C \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log p(\mathbf{x}_t | \mathbf{z}_t; \mu, \Lambda, \Psi) \, d\mathbf{z}_t + \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log p(\mathbf{z}_t) \, d\mathbf{z}_t + C \\
&= \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \log p(\mathbf{x}_t | \mathbf{z}_t; \mu, \Lambda, \Psi) \, d\mathbf{z}_t + C
\end{aligned}$$

where C denotes an arbitrary constant and we have used the fact that $p(\mathbf{z}_t)$ *does not depend* on any of the model parameters since $Z \sim N_k(0, I)$.

Writing the joint distribution with X as the first block rather than Z, we have

$$\begin{bmatrix} X \\ Z \end{bmatrix} \sim N\left(\begin{bmatrix} \mu \\ 0_k \end{bmatrix}, \begin{bmatrix} \Lambda\Lambda' + \Psi & \Lambda \\ \Lambda' & I \end{bmatrix}\right)$$

Hence, using the properties of normal distributions

$$\begin{aligned} X|Z &\sim N_p(\mu_{X|Z}, \Sigma_{X|Z}) \\ \mu_{X|Z} &= \mu + \Lambda Z \\ \Sigma_{X|Z} &= \Psi \end{aligned}$$

so

$$p(\mathbf{x}_t|\mathbf{z}_t; \mu, \Lambda, \Psi) = \frac{\exp\left\{-\frac{1}{2}(\mathbf{x}_t - \mu - \Lambda\mathbf{z}_t)'\Psi^{-1}(\mathbf{x}_t - \mu - \Lambda\mathbf{z}_t)\right\}}{(2\pi)^{p/2}\left|\Psi\right|^{1/2}}$$

and hence

$$\log p(\mathbf{x}_t|\mathbf{z}_t; \mu, \Lambda, \Psi) = -\frac{1}{2}\left[\log|\Psi| + p\log(2\pi) + (\mathbf{x}_t - \mu - \Lambda\mathbf{z}_t)'\Psi^{-1}(\mathbf{x}_t - \mu - \Lambda\mathbf{z}_t)\right]$$

Exchanging integral and derivative in the objective function, deal with terms one at a time, f doesn't depend on parameters, etc...

Updating Λ: Differentiating,

$$\begin{aligned} \nabla_\Lambda \log p(\mathbf{x}_t|\mathbf{z}_t; \mu, \Lambda, \Psi) &= \nabla_\Lambda\left[\frac{1}{2}(\mathbf{x}_t - \mu)'\Psi^{-1}\Lambda\mathbf{z}_t + \frac{1}{2}\mathbf{z}_t'\Lambda'\Psi^{-1}(\mathbf{x}_t - \mu) - \frac{1}{2}\mathbf{z}_t'\Lambda'\Psi^{-1}\Lambda\mathbf{z}_t\right] \\ &= \nabla_\Lambda\left[\mathbf{z}_t'\Lambda'\Psi^{-1}(\mathbf{x}_t - \mu) - \frac{1}{2}\mathbf{z}_t'\Lambda'\Psi^{-1}\Lambda\mathbf{z}_t\right] \\ &= \nabla_\Lambda\left[\mathrm{tr}\left\{\mathbf{z}_t'\Lambda'\Psi^{-1}(\mathbf{x}_t - \mu)\right\} - \frac{1}{2}\mathrm{tr}\left\{\mathbf{z}_t'\Lambda'\Psi^{-1}\Lambda\mathbf{z}_t\right\}\right] \\ &= \nabla_\Lambda\mathrm{tr}\left\{\Lambda'\Psi^{-1}(\mathbf{x}_t - \mu)\mathbf{z}_t'\right\} - \frac{1}{2}\nabla_\Lambda\mathrm{tr}\left\{\Lambda'\Psi^{-1}\Lambda\mathbf{z}_t\mathbf{z}_t'\right\} \end{aligned}$$

where we have used the fact that each term is a scalar, and thus equals its trace, and $\text{tr}(AB) = \text{tr}(BA)$ with $\mathbf{z}_t$ playing the role of A.

It remains to calculate two matrix derivatives. For the first term we need to calculate $\nabla_X \text{tr}(X'A)$ where Λ plays the role of X and $\Psi^{-1}(\mathbf{x}_t - \mu)\mathbf{z}_t'$ plays the role of A. It turns out that[1]

$$\nabla_X \text{tr}(X'A) = A$$

For the second term we need to calculate $\nabla_A \text{tr}(X'BXC)$ where Λ plays the role of X, Ψ^{-1} plays the role of B, and $\mathbf{z}_t\mathbf{z}_t'$ plays the role of C. It turns out that[2]

$$\text{tr}(X'BXC) = BXC + BXC'$$

Finally, we have,

$$\begin{aligned} \nabla_\Lambda \log p(\mathbf{x}_t|\mathbf{z}_t; \mu, \Lambda, \Psi) &= \Psi^{-1}(\mathbf{x}_t - \mu)\mathbf{z}_t' - \frac{1}{2}\left(\Psi^{-1}\Lambda\mathbf{z}_t\mathbf{z}_t' + \Psi^{-1}\Lambda\mathbf{z}_t\mathbf{z}_t'\right) \\ &= \Psi^{-1}(\mathbf{x}_t - \mu)\mathbf{z}_t' - \Psi^{-1}\Lambda\mathbf{z}_t\mathbf{z}_t' \end{aligned}$$

Thus, the first order condition for Λ is

$$\sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t) \left[\Psi^{-1}(\mathbf{x}_t - \mu)\mathbf{z}_t' - \Psi^{-1}\Lambda\mathbf{z}_t\mathbf{z}_t'\right] d\mathbf{z}_t = 0$$

Rearranging,

$$\begin{aligned} \Psi^{-1} \sum_{t=1}^{T} (\mathbf{x}_t - \mu) \int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t'\ d\mathbf{z}_t &= \Psi^{-1}\Lambda \sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t\mathbf{z}_t'\ d\mathbf{z}_t \\ \sum_{t=1}^{T} (\mathbf{x}_t - \mu) \int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t'\ d\mathbf{z}_t &= \Lambda \left(\sum_{t=1}^{T} \int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t\mathbf{z}_t'\ d\mathbf{z}_t \right) \end{aligned}$$

[1] See, inter alia, Peterson & Pederson (2012) *The Matrix Cookbook*, Section 2.5.1 or the Wikipedia article on Matrix Calculus.

[2] Ibid.

Solving for Λ and substituting the result of the E-step,

$$\begin{aligned}
\Lambda^{(j)} &= \left(\sum_{t=1}^{T}(\mathbf{x}_t-\mu)\int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t'\ d\mathbf{z}_t\right)\left(\sum_{t=1}^{T}\int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t\mathbf{z}_t'\ d\mathbf{z}_t\right)^{-1} \\
&= \left(\sum_{t=1}^{T}(\mathbf{x}_t-\mu)\int \mathcal{N}\left(\mathbf{z}_t|\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)},\Sigma_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right)\ \mathbf{z}_t'\ d\mathbf{z}_t\right)\left(\sum_{t=1}^{T}\int \mathcal{N}\left(\mathbf{z}_t|\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)},\Sigma_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right)\ \mathbf{z}_t\mathbf{z}_t'\ d\mathbf{z}_t\right)^{-1} \\
&= \left[\sum_{t=1}^{T}(\mathbf{x}_t-\mu)\left(\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right)'\right]\left[\sum_{t=1}^{T}\left\{\left(\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right)\left(\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right)'+\left(\Sigma_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right)\right\}\right]^{-1}
\end{aligned}$$

where $\mathcal{N}(\mathbf{z}|\mu,\Sigma)$ denotes a multivariate normal density with argument $\mathbf{z}$, mean μ and variance matrix Σ and

$$\begin{aligned}
\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)} &= \left(\Lambda^{(j-1)}\right)'\left[\Lambda^{(j-1)}\left(\Lambda^{(j-1)}\right)'+\Psi^{(j-1)}\right]^{-1}\left(\mathbf{x}_t-\mu^{(j-1)}\right) \\
\Sigma_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)} &= \mathbf{I}_k-\left(\Lambda^{(j-1)}\right)'\left[\Lambda^{(j-1)}\left(\Lambda^{(j-1)}\right)'+\Psi^{(j-1)}\right]^{-1}\Lambda^{(j-1)}
\end{aligned}$$

Notice that the M-step update for Λ looks what would be the multivariate OLS estimator if Z were observed: $\Lambda' = (Z'Z)^{-1}Z'X$. Since we don't observe Z we substitute its conditional mean given X. The only twist is the conditional variance term in the term that serves as the analogue of $(Z'Z)^{-1}$. This accounts for the uncertainty in our estimate of Z based on observing X.

Updating Ψ: Recall from above that Ψ is a diagonal matrix. Let ψ_{ii} denote its ith diagonal element. Since the determinant of a diagonal matrix is simply the product of its diagonal elements and the log of a product equals the sum of the logs, it follows that $\log|\Psi| = \sum_{i=1}^{p}\log\psi_{ii}$. Similarly, if c is a $p\times 1$ vector then $c'\Psi^{-1}c = \sum_{i=1}^{p}c_i^2/\psi_{ii}$. It follows that

$$\nabla_\Psi \log|\Psi| = \Psi^{-1}$$

and

$$\nabla_\Psi c'\Psi^{-1}c = -\Psi^{-1}cc'\Psi^{-1} = \Psi^{-2}cc'$$

hence

$$\nabla_\Psi \log p(\mathbf{x}_t|\mathbf{z}_t;\mu,\Lambda,\Psi) = -\frac{1}{2}\left[\Psi^{-1} - \Psi^{-2}(\mathbf{x}_t-\mu-\Lambda\mathbf{z}_t)(\mathbf{x}_t-\mu-\Lambda\mathbf{z}_t)'\right]$$

Multiply through to eliminate -1/2 and one of the Ψ^{-1} then should be relatively straightforward to plug in the M-step as before. Just need to impose diagonality at the end since we're carrying around the full Ψ matrix.

Updating μ: Differentiating and rearranging,

$$\begin{aligned}
\nabla_\mu \log p(\mathbf{x}_t|\mathbf{z}_t;\mu,\Lambda,\Psi) &= -\frac{1}{2}\nabla_\mu\left(-\mu'\Psi^{-1}\mathbf{x}_t + \mu'\Psi^{-1}\mu + \mu'\Psi^{-1}\Lambda\mathbf{z}_t - \mathbf{x}_t'\Psi^{-1}\mu + \mathbf{z}_t'\Lambda'\Psi^{-1}\mu\right) \\
&= \nabla_\mu\left(\mathbf{x}_t'\Psi^{-1}\mu - \mathbf{z}_t'\Lambda'\Psi^{-1}\mu - \frac{1}{2}\mu'\Psi^{-1}\mu\right) \\
&= \left(\mathbf{x}_t'\Psi^{-1}\right)' - \left(\mathbf{z}_t'\Lambda'\Psi^{-1}\right)' - \Psi^{-1}\mu \\
&= \Psi^{-1}\left(\mathbf{x}_t - \Lambda\mathbf{z}_t - \mu\right)
\end{aligned}$$

where we have used the results $\nabla_{\mathbf{x}}\mathbf{a}'\mathbf{x} = \mathbf{a}$ and$\nabla_{\mathbf{x}}\mathbf{x}'A\mathbf{x} = (A+A')\mathbf{x}$ along with the fact that Ψ^{-1} is symmetric.[3] Hence the first-order condition for μ is

$$\sum_{t=1}^{T}\int f_t^{(j-1)}(\mathbf{z}_t)\left[\Psi^{-1}\left(\mathbf{x}_t - \Lambda\mathbf{z}_t - \mu\right)\right]d\mathbf{z}_t = 0$$

[3]See, inter alia, Peterson & Pederson (2012) *The Matrix Cookbook*, Section 2.4.1–2 or the Wikipedia article on Matrix Calculus.

Left-multiplying both sides by Ψ, using the fact that $f_t^{(j-1)}(\mathbf{z}_t)$ is a density, and substituting the E-step gives

$$\begin{aligned}
\sum_{t=1}^{T}\left(\mathbf{x}_t - \Lambda\int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t\ d\mathbf{z}_t - \mu\right) &= 0 \\
T\bar{\mathbf{x}} - \Lambda\sum_{t=1}^{T}\int f_t^{(j-1)}(\mathbf{z}_t)\ \mathbf{z}_t\ d\mathbf{z}_t &= T\mu \\
\bar{\mathbf{x}} - \Lambda\left(\frac{1}{T}\sum_{t=1}^{T}\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}\right) &= \mu
\end{aligned}$$

We see that, provided that conditional expectations $\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)}$ sum to zero over t, the M-step update for μ is simply $\mu^{(j)} = \bar{\mathbf{x}}$ which doesn't depend on j. From above,

$$\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)} = \left(\Lambda^{(j-1)}\right)'\left[\Lambda^{(j-1)}\left(\Lambda^{(j-1)}\right)' + \Psi^{(j-1)}\right]^{-1}\left(\mathbf{x}_t - \mu^{(j-1)}\right)$$

and hence, summing over t

$$\sum_{t=1}^{T}\mu_{\mathbf{z}_t|\mathbf{x}_t}^{(j-1)} = \left(\Lambda^{(j-1)}\right)'\left[\Lambda^{(j-1)}\left(\Lambda^{(j-1)}\right)' + \Psi^{(j-1)}\right]^{-1} T\left(\bar{\mathbf{x}} - \mu^{(j-1)}\right)$$

So as long as $\mu^{(j-1)} = \bar{\mathbf{x}}$, the conditional expectations will sum to zero so that $\mu^{(j)} = \bar{\mathbf{x}}$. This makes perfect sense: we know that $\bar{\mathbf{x}}$ is the MLE for the mean of a normal distribution and we have shown that if we set $\mu^{(1)} = \bar{\mathbf{x}}$, the M-step will *never update* μ. This is just a very complicated way of saying that we can demean $\mathbf{x}_t$ before carrying out Factor Analysis and then proceed as though μ were zero.

2.5.3 EM for Factor Analysis: Summary of the Algorithm

2.6 Estimating the Factor Scores

We saw above that

$$\begin{aligned} Z|X &\sim N_k(\mu_{Z|X}, \Sigma_{Z|X}) \\ \mu_{Z|X} &= \Lambda'(\Lambda\Lambda' + \Psi)^{-1}(X - \mu) \\ \Sigma_{Z|X} &= \mathbf{I}_k - \Lambda'(\Lambda\Lambda' + \Psi)^{-1}\Lambda \end{aligned}$$

So if we want to *estimate* the realizations $\mathbf{z}_i$ of the latent vector X that correspond to the observations $\mathbf{x}_i$, the obvious choice is the conditional mean evaulated at the maximum likelihood estimators, namely

$$\widehat{\mathbf{z}}_i = \widehat{\Lambda}' \left(\widehat{\Lambda}\widehat{\Lambda}' + \widehat{\Psi}\right)^{-1} (\mathbf{x}_i - \bar{\mathbf{x}})$$

Notice that this is *precisely* the collection of values that emerge from the *final* M-step since $\bar{x}$ is the MLE for μ and we showed that it is never updated.

3 Principal Components Analysis

If asked to summarize a k-dimensional random vector $\mathbf{x}$ our first inclination might be to examine its moments: the mean vector $\boldsymbol{\mu}$ and variance-covariance matrix Σ. As mentioned above, it might be difficult or impossible to estimate Σ if k is large relative to the sample size. More fundamentally, however, even if it were *known* rather than estimated, Σ would still be challenging to *interpret* unless k is fairly small. Principal Components Analysis (PCA) is a classical statistical technique that is designed to help us discover "structure" in a variance-covariance matrix by considering particular linear combinations of the elements of $\mathbf{x}$. We can apply PCA either to a population covariance matrix or a sample covariance matrix: the basic idea is the same in either case. To bring out some important relationships between ideas we will examine PCA

from several different, but equivalent perspectives. To begin, we will consider what is sometimes called the "analysis view" of PCA which amounts to solving a simple optimization problem.

3.1 The "Analysis View" of PCA

Suppose that $\boldsymbol{\alpha}$ is a constant vector. Then $\boldsymbol{\alpha}'\mathbf{x}$ is a *scalar* random variable that summarizes $\mathbf{x}$ via a weighted average. The question is, what weights provide an "interesting" summary of $\mathbf{x}$? One idea would be to choose $\boldsymbol{\alpha}$ to maximize the variance of the linear combination $\boldsymbol{\alpha}'\mathbf{x}$. Although this idea is appealing, there is an obvious problem: we can make the variance of the linear combination arbitrarily large! To turn this into a well-defined problem, we need to constrain $\boldsymbol{\alpha}$ in some way. There are many possible constraints we could use. PCA imposes a particularly simple one by requiring that $\boldsymbol{\alpha}$ has *unit norm*, in other words $\boldsymbol{\alpha}'\boldsymbol{\alpha} = 1$.

The First Principal Component The linear combination $\boldsymbol{\alpha}_1\mathbf{x}$ formed from the solution to

$$\max_{\alpha_1} Var(\boldsymbol{\alpha_1}'\mathbf{x}) \text{ subject to } \boldsymbol{\alpha_1}'\boldsymbol{\alpha_1} = 1$$

is called the *first principal component* of Σ. For a general linear combination of $\mathbf{x}$ we have

$$\begin{aligned} Var(\boldsymbol{\alpha}'\mathbf{x}) &= E[\boldsymbol{\alpha}'\mathbf{x}\mathbf{x}'\boldsymbol{\alpha}] - E[\boldsymbol{\alpha}'\mathbf{x}]E[\mathbf{x}'\boldsymbol{\alpha}] \\ &= \boldsymbol{\alpha}\left('E[\mathbf{x}\mathbf{x}'] - E[\mathbf{x}]E[\mathbf{x}']\right)\boldsymbol{\alpha}' \\ &= \boldsymbol{\alpha}'\Sigma\boldsymbol{\alpha} \end{aligned}$$

Thus, to find $\boldsymbol{\alpha}_1$ we maximize the Lagrangian

$$\mathcal{L}(\boldsymbol{\alpha}_1, \lambda) = \boldsymbol{\alpha}_1'\Sigma\boldsymbol{\alpha}_1 - \lambda(\boldsymbol{\alpha_1}'\boldsymbol{\alpha_1} - 1)$$

where λ is a *scalar* Lagrange Multiplier since there is only a *one* constraint. The first-order condition for α_1 is

$$2(\Sigma\boldsymbol{\alpha}_1 - \lambda\boldsymbol{\alpha}_1) = \mathbf{0}$$

since Σ is a symmetric matrix.[4] Rearranging,

$$(\Sigma - \lambda\mathbf{I}_k)\boldsymbol{\alpha}_1 = \mathbf{0}$$

so we see at once that $\boldsymbol{\alpha}_1$ must be an *eigenvector* of Σ and λ the corresponding *eigenvalue*. But which one? Rearranging the first-order condition for $\boldsymbol{\alpha}_1$ gives

$$\Sigma\boldsymbol{\alpha}_1 = \lambda\boldsymbol{\alpha}_1$$

Substituting this into the objective function,

$$Var(\boldsymbol{\alpha}_1'\mathbf{x}) = \boldsymbol{\alpha}_1'\Sigma\boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_1'\lambda\boldsymbol{\alpha}_1 = \lambda\boldsymbol{\alpha}_1'\boldsymbol{\alpha}_1 = \lambda$$

since λ is a scalar and $\boldsymbol{\alpha}_1'\boldsymbol{\alpha}_1 = 1$. In other words, the variance of the first principal component equals λ. Since this is what we want to *maximize*, we should make λ as large as possible. But λ must be one of the eigenvalues of Σ. Therefore, $\boldsymbol{\alpha}_1$ is the eigenvector corresponding to the *largest eigenvalue* of Σ. Recall that since Σ is a variance-covariance matrix, is must be positive semi-definite hence all its eigenvalues must be non-negative.

The Second Principal Component

Reconstruction Error

[4] In general, $\nabla_{\mathbf{x}}\mathbf{x}'A\mathbf{x} = (A + A')\mathbf{x}$. See, for example, Harville (1997) section 15.3.

3.2 PCA for the Sample Covariance Matrix

Let X be a design matrix from which we have subtracted the column means. Then the **sample covariance matrix** S is defined as

$$S = \frac{X'X}{T}$$

This is the MLE for Σ under multivariate normality. If you prefer the unbiased estimator, simply divide by $(T-1)$ rather than T. Now we can simply proceed as above with S playing the role of Σ.

Computing Sample PCs The best way to calculate the sample PCs is to use the singular value decomposition (SVD) of the centered design matrix X.[5] We have $X = UDV'$ and hence

$$X'X = VDU'UDV' = VD^2V'$$

Right-multiplying by V gives $(X'X)V = VD^2$. Thus, letting $\mathbf{v}_i$ denote the ith column of V, we have $(X'X)\mathbf{v}_i = d_i^2\mathbf{v}_i$. Dividing both sides by T gives $S\mathbf{v}_i = T^{-1}d_i^2\mathbf{v_i}$. Thus, $(\mathbf{v}_i, T^{-1}d_i^2)$ are the eigenvector-eigenvalue pairs of the sample covariance matrix.

The PC *loadings* for S are the $\mathbf{v}_i$. The PC *scores* of the dataset are $\mathbf{v}_i'\mathbf{x}_t$ where $\mathbf{x}_t$ is the vector of observations for individual (or time period) t. Collecting these for all individuals in the dataset gives the vector of PC *scores* for the ith PC:

$$\mathbf{z}_i = \begin{bmatrix} z_{i1} \\ \vdots \\ z_{iT} \end{bmatrix} = \begin{bmatrix} \mathbf{v}_i'\mathbf{x}_1 \\ \vdots \\ \mathbf{v}_i'\mathbf{x}_T \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 \\ \vdots \\ \mathbf{x}_T \end{bmatrix} \mathbf{v}_i = X\mathbf{v}_i$$

[5]For details on this decomposition, see the lecture notes for shrinkage estimation.

Since X has been demeaned, we have

$$\bar{z}_i = \frac{1}{T}\sum_{t=1}^{T} \mathbf{v}_i' \mathbf{x}_t = \mathbf{v}_i' \left(\frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_t \right) = \mathbf{v}_i' \mathbf{0} = 0$$

Thus, we can calculate the variance of the ith PC score as follows:

$$\frac{1}{T}\sum_{t=1}^{T}(z_{it} - \bar{z}_i)^2 = \frac{1}{T}\sum_{i=1}^{T} z_{it}^2 = \frac{1}{T}\mathbf{z}_i'\mathbf{z}_i = \frac{1}{T}(X\mathbf{v}_i)'(X\mathbf{v}_i) = T^{-1}d_i^2$$

since $V'(X"X)V = D$. In fact, we do not need to calculate $X\mathbf{v}_i$ to get the PC scores: we get them for free from the SVD! To see this, note that $XV = UDV'V = UD$. That is,

$$X \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{v}_p \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_p \end{bmatrix} \begin{bmatrix} d_i & & 0 \\ & \ddots & \\ 0 & & d_p \end{bmatrix}$$

In other words, $\mathbf{z}_i = d_i \mathbf{u}_i$. And we're done!

3.3 Probabilistic PCA

Comparison of PCA and Factor Analysis So far we have examined factor analysis and PCA. Both procedures yield a lower-dimensional approximation to a covariance matrix Σ, but they behave in very different ways. As we saw above, the goal in PCA is to find directions of *maximal variance*. In essence, PCA directs its attention to the *diagonal* elements of Σ. Indeed, one can show that the solution to $\max_B \text{trace}\{Var(B'\mathbf{x})\}$ where B is a $p \times q$ matrix is given by placing the first q Principal Components of Σ into the columns of B. The transformed variables $B'\mathbf{x}$ attempt to "preserve" as much variance as possible. In many cases, *it turns out* that PCA also does a reasonable job of summarizing the off-diagonal elements of Σ. By the Spectral Decomposition,

we can write

$$\Sigma = \sum_{k=1}^{p} \lambda_k \alpha_k \alpha_k'$$

Now, because the λ_k are decreasing and we have imposed the normalization constraint $\alpha_k'\alpha_k = 1$, the elements of $\lambda_k \alpha_k \alpha_k'$ *tend* to decrease as k increases. However, they are not *guaranteed* to decrease. In particular, if the elements of $\mathbf{x}$ have different variances, PCA will *miss* much of the covariance structure in its attempt to maximize variance. In contrast, Factor Analysis is *only* concerned with the off-diagonal elements of Σ. The diagonal elements are modeled by a vector of idiosyncratic errors, so the factor loadings are only concerned with the correlations *between* elements of $\mathbf{x}$.

A related issue is scale-invariance. Because variances are not scale-invariant, PCA is not scale invariant. In contrast, suppose we were to define a new random vector Y obtained by multiplying $\mathbf{x}$ by a diagonal matrix C. The transformed factor model would be

$$\begin{aligned} Y &= CX \\ &= C\mu + C\Lambda Z + C\epsilon \\ &= \widetilde{\mu} + \widetilde{\Lambda} Z + \widetilde{\epsilon} \end{aligned}$$

where $\widetilde{\epsilon}$ has variance matrix $C\Psi$. In other words, the re-scaling is simply *absorbed* into the model coefficients, as in linear regression, and the factors themselves, Z, remain as before. Thus, we see that Factor Analysis is scale-invariant. As a consequence we do *not* need to normalize variables before carrying out Factor Analysis: the idiosyncratic variances Ψ "handle it for us."

Another difference between PCA and Factor Analysis concerns the relationships between the estimated "factors." Suppose we fit a k-factor model and then change our minds and decide to fit a $(k+1)$-factor model. We will *not* get the same values for the first k factor scores, the estimates of Z, as we did before! This is in stark contrast to PCA. If I decide to use, say, k PCs in PCR and then change my mind and use $k+1$, the first k "factors," the PC

scores, *remain unchanged*.

The most fundamental distinction between PCA and Factor Analysis, however, is that Factor Analysis provides a *generative probabilistic model* for the data. If you know the parameters, you can *simulate* data that has a factor structure. PCA, on the other hand, is merely an *algorithm*. There's no likelihood involved. It *is possible*, however, to *construct* a probabilistic model that behaves like PCA. This provides a very helpful way of relating PCA to Factor Analysis and drawing out their differcenes.

A Generative Model for PCA: Tipping & Bishop (1999) To construct a probabilistic model for PCA, Tipping and Bishop (1999) take the standard factor model considered above and restrict Ψ to be *isotropic*. That is, they assume $\Psi = \sigma^2 I$ so that the idiosyncratic variances are *equal* across components of X. Under this simplification, they derive an *explicit* formula for the maximum likelihood estimators of the model parameters, namely:

$$\widehat{\Lambda}_{ML} = V_q(L_q - \sigma^2 I)^{1/2} R$$

where V_q is a matrix containing the first q eigenvectors of the sample covariance matrix S, $L_q = \text{diag}\{\lambda_i\}_{i=1}^q$constains the corresponding eigenvalues, and R is an *arbitrary* $q \times q$ orthogonal rotation matrix. Since it's arbitrary, there is no loss in generality from settting $R = I$. The MLE for σ^2 is shown to be

$$\widehat{\sigma}^2_{ML} = \frac{1}{p-q} \sum_{j=q+1}^{p} \lambda_j$$

This is the *average variance* of the components that are discarded by ordinary PCA!

$$\widehat{\mathbf{z}}_i = \widehat{\Lambda}' \left(\widehat{\Lambda}\widehat{\Lambda}' + \widehat{\Psi} \right)^{-1} (\mathbf{x}_i - \bar{\mathbf{x}})$$

We see that as $\sigma^2 \to 0$, $\widehat{\Lambda}_{ML} \to V_q L_q^{1/2} R$ and $\widehat{\Psi} = \widehat{\sigma}^2 I \to 0$. Hence, we have

$$\begin{aligned}
\left(\widehat{\Lambda}\widehat{\Lambda}' + \widehat{\Psi}\right)^{-1} \to \left(\widehat{\Lambda}\widehat{\Lambda}'\right)^{-1} &= \left[(V_q L_q^{1/2} R)(V_q L_q^{1/2} R)'\right]^{-1} \\
&= \left[V_q L_q^{1/2} R R' L_q^{1/2} V_q'\right]^{-1} \\
&= \left[V_q L_q V_q'\right]^{-1}
\end{aligned}$$