STATS 306B: Unsupervised Learning **Spring 2014**

Lecture 9 — April 28

Lecturer: Lester Mackey *Scribe: Jiyao Kou, Shuo Xie*

9.1 Recap

In the last lecture, we discussed the limits of linearity in ordinary PCA and introduced **kernel PCA**, a non-linear dimensionality reduction method with little dependence on the dimension of non-linear feature space in which it operates. This lecture will present a few additional remarks on kernel PCA and then introduce **factor analysis**, a probabilistic approach to linear dimensionality reduction.

9.2 Remarks on Kernel PCA

9.2.1 Kernel PCA and spectral clustering

There is a close relationship between kernel PCA and spectral clustering, even though we motivated the two methods in quite different ways. To see this, note that kernel PCA finds the *bottom* k eigenvectors of $I - K$ for a kernel matrix K while Spectral Clustering finds the bottom k eigenvectors of a Laplacian matrix. The parallel becomes especially clear when we choose the symmetric normalized Laplacian, $L_{sym} = I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$. Notably, if W is a kernel matrix (meaning it is symmetric positive semidefinite) then $D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$ is also a kernel matrix. In this case, spectral clustering can be interpreted as embedding datapoints into an associated feature space and clustering using the recovered reconstruction weights α_j introduced in the context of kernel PCA. The normalization introduced in spectral clustering implies that even if $W = K$ (a Gaussian kernel matrix is a common choice for either) the resulting eigenvectors α_j could still be quite different. See the accompanying slides for an empirical comparison of spectral clustering and kernel PCA from ESL.

9.2.2 Mean centering

One must mean-center data in ϕ-space before applying kernel PCA. Fortunately, it is always possible to mean-center the data given access only to the kernel matrix K^{ϕ}. Indeed, one can check that $\tilde{K}^{\phi} = (I - \frac{\mathbf{11^T}}{n})K^{\phi}(1 - \frac{\mathbf{11^T}}{n})$ is a mean-centered version of K^{ϕ}.

9.2.3 Solving kernel PCA with PCA

Instead of solving kernel PCA as described last time, once K^ϕ is formed, we can factor it as $K^\phi = GG^T$ using, for instance, the Cholesky decomposition and then run standard PCA on $G \in \mathbb{R}^{n\times n}$. In this case G becomes an alternative representation for Φ, since $K^\phi = \Phi\Phi^T$.

9.2.4 Data modeling

PCA and kernel PCA are both model-free methods for dimensionality reduction: they are determined completely by their reduction objectives and make no attempt to explicitly model the data generating process. We will next explore a classical model-based approach to dimensionality reduction.

9.3 Factor Analysis

Factor Analysis is a probabilistic latent feature model for the linear dimensionality reduction of continuous data. The model is often employed when the coordinates of each datapoint x_i are measurements largely explainable by a small set of hidden factors. Common application areas include

- Educational and psychological testing, in which one aims to infer abilities or personal traits from test responses,
- EEG brain scanning to measure neuronal activities, and
- Financial data analysis, in which one attempts infer market confidence and external driving forces from stock prices.

9.3.1 Model formulation

The factor analysis generative model for datapoints $x_1, ..., x_n \in \mathbf{R}^p$ is formulated as

$$\begin{aligned} z_i &\overset{\text{iid}}{\sim} \mathcal{N}(0, I_{q\times q}) \\ x_i|z_i &\overset{\text{iid}}{\sim} \mathcal{N}(\mu + \Lambda z_i, \Psi), \end{aligned}$$

where z_i is a collection of **latent factors** in $\mathbf{R}^q$ associated with the datapoint x_i. We have the following unknown parameters:

- $\Psi \in \mathbf{R}^{p\times p}$, the diagonal covariance matrix,
- $\mu \in \mathbf{R}^p$, the global mean vector,
- $\Lambda \in \mathbf{R}^{p\times q}$, the **factor loading** matrix.

We see that, under this model, each datapoint is viewed as a random draw from a subspace spanned by the columns of Λ plus noise. Note moreover that all the coordinates of x_i depend on the same small set of hidden factors, which ultimately induces correlations amongst coordinates of x_i; since the noise matrix is diagonal, all correlations amongst the coordinates of x_i must be explained in this manner.

9.3.2 Marginal distribution of x_i

To better understand the correlation-inducing nature of the z_i's, let us consider the marginal distribution of x_i under the factor analysis model. First notice that $x_i = \mu + \Lambda z_i + w_i$ where $w_i \sim \mathcal{N}(0, \Psi)$ with $w_i \perp\!\!\!\perp z_i$. Thus, x_i is Gaussian, so it suffices to compute its mean and covariance to establish the distribution The mean of x_i is

$$\mathbf{E}[x_i] = \mu + \Lambda \mathbf{E}[z_i] + \mathbf{E}[w_i] = \mu,$$

while the covariance is

$$\begin{aligned}
\mathbf{Cov}(x_i) &= \mathbf{E}[(x_i - \mu)(x_i - \mu)^T] \\
&= \mathbf{E}[(\Lambda z_i + w_i)(\Lambda z_i + w_i)^T] \\
&= \Lambda \mathbf{E}[z_i z_i^T]\Lambda + \mathbf{E}[w_i w_i^T] + \mathbf{E}[\Lambda z_i w_i^T] + \mathbf{E}[w_i z_i^T \Lambda^T] \\
&= \Lambda\Lambda^T + \Psi + 0 + 0 \\
&= \Lambda\Lambda^T + \Psi
\end{aligned}$$

The fourth equality is due to the independence of w_i and z_i and the fact that they have mean 0. So marginally, $x_i \sim \mathcal{N}(\mu, \Lambda\Lambda^T + \Psi)$, where $\Lambda\Lambda^T$ represents low-rank latent factor covariance structure and Ψ is the diagonal noise variance.

9.3.3 Unsuperivsed learning goal

As usual, our unsupervised learning goal is to infer latent factors $z_{1:n}$ from observations $x_{1:n}$. We will first examine how to carry out probabilistic inference of $p(z_i|x_i)$ assuming all parameters $\theta = (\Lambda, \Psi, \mu)$ are known. Our strategy, which we appeal to frequently in the Gaussian context, is to first compute the joint distribution $p(z_i, x_i)$ and then derive the conditional distribution from this joint. A key observation is that $(z_i, x_i) = (z_i, \Lambda z_i + w_i + \mu)$ are jointly multivariate Gaussian, since they are affine functions of jointly Gaussian variables. Hence their joint distribution is determined by the mean value and covariance matrix. The mean value is

$$\mathbf{E}\left(\begin{bmatrix} z_i \\ x_i \end{bmatrix}\right) = \begin{bmatrix} 0 \\ \mu \end{bmatrix}$$

while the covariance matrix is

$$\mathbf{Cov}\left(\begin{bmatrix} z_i \\ x_i \end{bmatrix}\right) = \begin{bmatrix} \mathbf{Cov}(z_i) & \mathbf{Cov}(z_i, x_i) \\ \mathbf{Cov}(x_i, z_i) & \mathbf{Cov}(x_i) \end{bmatrix}$$

where we know that

- $\mathbf{Cov}(z_i) = \mathbf{I}$
- $\mathbf{Cov}(x_i) = \Lambda\Lambda^T + \Psi$
- $\mathbf{Cov}(z_i, x_i) = \mathbf{E}[z_i(x_i - \mu)^T] = \mathbf{E}[z_i(\Lambda z_i + w_i)^T] = \mathbf{E}[z_i z_i^T]\Lambda^T + \mathbf{E}[z_i w_i^T] = \Lambda^T + 0.$

Thus we have:

$$(z_i, x_i) \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ \mu \end{bmatrix}, \begin{bmatrix} \mathbf{I} & \Lambda^T \\ \Lambda & \Lambda\Lambda^T + \Psi \end{bmatrix} \right).$$

To obtain the conditional distribution of z_i given x_i, we use a key fact about Gaussian conditioning (which can be found in the assigned multivariate Gaussian chapter):

Fact 1. *If* $(z, x) \sim \mathcal{N}\left(\begin{bmatrix} \mu_z \\ \mu_x \end{bmatrix}, \begin{bmatrix} \Sigma_{zz} & \Sigma_{zx} \\ \Sigma_{xz} & \Sigma_{xx} \end{bmatrix} \right)$ *and* $\Sigma_{xx} > 0$*, then* $z|x$ *is Gaussian distributed with mean*

$$\mu_{z|x} = \mu_z + \Sigma_{zx}\Sigma_{xx}^{-1}(x - \mu_x)$$

and covariance:

$$\Sigma_{z|x} = \Sigma_{zz} - \Sigma_{zx}\Sigma_{xx}^{-1}\Sigma_{xz}$$

Remark 2. We see that the conditional mean $\mu_{z|x}$ is the marginal mean μ_z adjusted by a term $\Sigma_{zx}\Sigma_{xx}^{-1}(x - \mu_x)$ that accounts for the covariance between x and z.

Moreover, the conditional variance $\Sigma_{z|x}$ is a reduced form of the marginal covariance Σ_{zz}, reduced because observing more information x can only decrease the variance. The matrix $\Sigma_{zz} - \Sigma_{zx}\Sigma_{xx}^{-1}\Sigma_{xz}$ can also be identified as the **Schur complement** of the full joint covariance with respect to Σ_{xx}.

Using the above fact in our setting, we see that $z_i|x_i$ is Gaussian distributed with mean

$$\begin{aligned} \mathbf{E}[z_i|x_i] &= 0 + \Lambda^T(\Lambda\Lambda^T + \Psi)^{-1}(x_i - \mu) \\ &= (\mathbf{I} + \Lambda^T\Psi^{-1}\Lambda)^{-1}\Lambda^T\Psi^{-1}(x_i - \mu) \end{aligned} \tag{9.1}$$

and covariance

$$\begin{aligned} \mathbf{Cov}(z_i|x_i) &= \mathbf{I} - \Lambda^T(\Lambda\Lambda^T + \Psi)^{-1}\Lambda \\ &= (\mathbf{I} + \Lambda^T\Psi^{-1}\Lambda)^{-1} \end{aligned} \tag{9.2}$$

Note the initial expressions in (9.1) and (9.2) both involve $p \times p$ matrix inversions, while the final expressions only require $q \times q$ matrix inversions. This rewriting is achieved by the Sherman-Morrison-Woodbury formula. Moreover, in (9.2) we see that variance has been reduced from $\mathbf{I}$ to $(\mathbf{I} + \Lambda^T\Psi^{-1}\Lambda)^{-1}$ by observing x_i. This distribution enables soft inference of z_i. For hard inference (point estimation) of z_i, it is common to use the conditional mode $\mathbf{E}[z_i|x_i]$.

9.3.4 Estimation of unknown parameters θ

In the previous section we assumed that all parameters were known, but we often need to estimate θ in practice. Consider for instance maximizing the log-likelihood of $x_{1:n}$,

$$\begin{aligned}\log p(x_{1:n};\theta) &= \sum_{i=1}^{n} \log p(x_i;\theta) \\ &= -\frac{n}{2}\log|\Lambda\Lambda^T+\Psi| - \frac{1}{2}\sum_i (x_i-\mu)^T(\Lambda\Lambda^T+\Psi)^{-1}(x_i-\mu) + const\end{aligned}$$

where *const* is a parameter-free term. Fortunately, this admits a closed-form maximum over μ: $\hat{\mu}_{MLE} = \frac{1}{n}\sum_{i=1}^{n} x_i$. Thus, when estimating Ψ and Λ, we can subsitute $\hat{\mu}_{MLE}$ for μ in likelihood. In fact, this is equivalent to modeling mean centered data $x_i \leftarrow x_i - \hat{\mu}_{MLE}$ with fixed $\mu = 0$. However, even with this simplification, no closed form Ψ and Λ maxima exist, since they are coupled by the inverse and determinant in the log likelihood. To conquer this difficulty, we will decouple these parameters via the EM algorithm in the next class.