CS 181 Spring 2023 Section 6
Principal Component Analysis and Expectation Maximization

Charumathi Badrinath, Oliver Cheng

1 Latent Variable Models

A **latent variable model** is a probabilistic model that contains unobserved variables.

Example: Coin Flips

Suppose you have 2 identical biased coins, C_0 and C_1. You put them in a hat, shake them up, and then pick a coin from the hat to flip and observe whether the outcome is heads or tails. You repeat this process 100 times. You are interested in finding a maximum likelihood estimate for the probability that C_0 and C_1 land heads. It's not immediately clear how to do this using the methods we have learned so far in the class. What makes this different? In this scenario each coin flip is accompanied by a *latent variable* which is *which* coin we chose to flip each time (we don't know this because we chose a coin at random from the hat and can't visually distinguish between the two coins). Thus, we can model this using a latent variable model.

For flip $n \in \{1, 2, \cdots, 100\}$...

- Let Z_n denote *which* coin we flipped. This is our *latent variable* and its support is $\{0, 1\}$ since we could have flipped either coin 0 or coin 1.
- Let Y_n be the outcome at the coin flip. Its support is $\{0, 1\}$ where 1 denotes heads and 0 denotes tails

We can put a specify a reasonable probabilistic model that relates our random variables.

- We assume that each coin flip is independent.
- Let $Z_n \sim \text{Bern}(0.5)$ since for each toss we have a 0.5 probability of choosing coin 0 and a 0.5 probability of choosing coin 1.
- Let $Y_n|Z_n = 0 \sim \text{Bern}(\phi_0)$ and $Y_n|Z_n = 1 \sim \text{Bern}(\phi_1)$ since we have different probabilities of flipping heads based on which coin we chose.

1.1 Model Structure

For the purpose of the derivations in this section we assume we have N observations in our training dataset. We denote our *latent variables* as Z_n and our *observed variables* as Y_n for

$n \in \{1, 2, \cdots N\}$. We assume that the latent variable was generated by some distribution that depends on an unknown parameter θ and $Y_n|Z_n$ was generated by some distribution that depends on an unknown parameter ϕ. We will assume that each observation is independent (although each observation is dependent on its *corresponding* latent variable). In math,

$$Z_n \sim p(\cdot\ ; \theta)$$
$$Y_n|Z_n \sim p(\cdot\ |Z_n; \phi)$$

1.2 Finding the MLE

Note: While proofs and derivations presented in the remainder of this section are slightly different from those in the pre-lecture notes, the *details of the expectation maximization algorithm are exactly the same*. We hope this gives you more diverse material to deepen your understanding and build intuition.

Now that we have a model of our environment, let's try and find the MLE of our unknown parameters θ and ϕ. by maximizing the log likelihood of our observed data with respect to them. Lets start with ϕ:

$$\phi_{\text{MLE}} = \arg\max_{\phi} \log \prod_{n=1}^{N} p(y_n; \phi)$$

Since the distribution of Y_n depends on Z_n, we can apply LOTP to marginalize out over all possible values of Z_n:

$$= \arg\max_{\phi} \log \prod_{n=1}^{N} \int_{\text{support } Z_n} p(y_n|z_n; \phi)p(z_n; \theta)dz_n$$
$$= \arg\max_{\phi} \sum_{n=1}^{N} \log \int_{\text{support } Z_n} p(y_n|z_n; \phi)p(z_n; \theta)dz_n$$

Now, we have to take the derivative of this expression with respect to ϕ to evaluate the arg max.

Question: Why is this hard to do?
Answer: The gradient in the integrand may be hard to evaluate. The derivative is also complicated to compute because it requires chain ruling a log and an integral.

Let's try to find an *alternative expression* for the log likelihood of the observed data that doesn't suffer from these pitfalls.

1.3 Decomposing the log likelihood

Note: All of the things we will do in the following derivation are done with the goal of finding an easier-to-optimize expression for the log likelihood of the observed data. Don't worry if it doesn't make sense how we came up with this as long as all the steps make mathematical sense. An alternate derivation can be found in the pre-lecture notes.

Let $Z_n \overset{\text{iid}}{\sim} q(\cdot;\theta)$. Let Ω_Z represent the support of all the Z_ns. Since q is a valid probability distribution, we know $\int_{\Omega_Z} q(z;\theta)dz = 1$. Thus the log likelihood $\ell(y_{1\cdots N};\phi,\theta)$ is:

$$\log \prod_{n=1}^{N} p(y_n;\phi) = \sum_{n=1}^{N} \log p(y_n;\phi) = \sum_{n=1}^{N} \log p(y_n;\phi) \int_{\Omega_Z} q(z)dz$$

Let's move the log term into the integral and multiply and divide by $p(z|y_n;\phi,\theta)$:

$$\begin{aligned}
&= \sum_{n=1}^{N} \int_{\Omega_Z} q(z) \log p(y_n;\phi)dz \\
&= \sum_{n=1}^{N} \int_{\Omega_Z} q(z) \log \left(\frac{p(y_n;\phi,\theta)p(z|y_n;\phi,\theta)}{p(z|y_n;\phi,\theta)} \right) dz \\
&= \sum_{n=1}^{N} \int_{\Omega_Z} q(z) \log \left(\frac{p(y_n,z;\phi,\theta)}{p(z|y_n;\phi,\theta)} \right) dz \\
&= \sum_{n=1}^{N} \int_{\Omega_Z} q(z) \log \left(\frac{p(y_n,z;\phi,\theta)}{q(z)} \frac{q(z)}{p(z|y_n;\phi,\theta)} \right) dz \\
&= \sum_{n=1}^{N} \int_{\Omega_Z} q(z) \log \frac{p(y_n,z;\phi,\theta)}{q(z)} dz + \int_{\Omega_Z} q(z) \log \frac{q(z)}{p(z|y_n;\phi,\theta)} dz \\
&= \sum_{n=1}^{N} \mathbb{E}_{Z_n \sim q(Z_n)} \left[\log \frac{p(y_n,Z_n;\phi,\theta)}{q(Z_n)} \right] + \mathbb{E}_{Z_n \sim q(Z_n)} \left[\log \frac{q(Z_n)}{p(Z_n|y_n;\phi,\theta)} \right]
\end{aligned}$$

The first term in the sum is called the *evidence lower bound* or the **ELBO**. The second term is the KL-divergence between $p(Z|y_{1\cdots N},\theta)$ and $q(Z)$ (the KL-divergence is a measure of *distance* between two probability distributions). Note that there are *three* parameters here that we need to optimize in order to maximize the log likelihood: ϕ and θ, as before, and the probability distribution q. It doesn't immediately make sense how to do this, however, since q is a function and not just a variable that you can take the derivative with respect to. The *expectation maximization* algorithm (EM algorithm) gives us a clever way to bypass this issue.

2 Expectation Maximization

In section 1.3 we saw how:

$$\ell(y_{1\cdots N};\phi,\theta) = \sum_{n=1}^{N} \text{ELBO}(\theta,\phi,q(Z_n)) + \text{D}_{\text{KL}}(q(Z_n)||p(Z|Y_n;\theta,\phi))$$

One property of the KL-divergence metric is that it is always ≥ 0. Thus, we have that:

$$\ell(y_{1\cdots N};\phi,\theta) \geq \sum_{n=1}^{N} \text{ELBO}(\theta,\phi,q(Z_n))$$

$$\implies \arg\max_{\phi,\theta} \ell(y_{1\cdots N};\phi,\theta) \geq \arg\max_{\phi,\theta} \sum_{n=1}^{N} \text{ELBO}(\theta,\phi,q(Z_n))$$

In words, maximizing the ELBO yields a lower bound on the maximum value of the log likelihood. Equality is achieved for the distribution of $q(Z_n)$ that allows the ELBO to be as large as possible.

Since our new objective is to *maximize* the ELBO, and since the ELBO and KL divergence sum to the log likelihood (a fixed number for a given setting of parameters), maximizing the ELBO is equivalent to *minimizing* the KL-divergence. We choose which of these terms we want to maximize/minimize based on what's easier to do.

The EM algorithm is an *iterative optimization algorithm* where we alternate between maximizing the log likelihood with respect to ϕ and θ while holding the distribution q constant and then find the q that maximizes the log likelihood while holding ϕ and θ constant. We initialize ϕ and θ to a random number within the support of their distribution.

2.1 E-Step

In the E-Step we minimize the KL-divergence with respect to q holding ϕ and θ fixed. The KL-divergence between two distributions is minimized (equal to 0) when the distributions *are the same*. Thus, we set $q_{\text{new}}(Z_n) = p(Z_n|Y_n;\theta_{\text{old}},\phi_{\text{old}})$ where θ_{old} and ϕ_{old} are the values of θ and ϕ from the previous iteration (for the first iteration of the algorithm, these are the values that θ and ϕ were initialized to).

2.2 M-Step

Now, we maximize the ELBO with respect to ϕ and θ holding q fixed. Through some algebra, which you can find in the pre-lecture notes we show that since $q(Z_n)$ is a constant

with respect to ϕ and θ our objective becomes:

$$\theta_{\text{new}}, \phi_{\text{new}} = \arg\max_{\theta,\phi} \sum_{n=1}^{N} \mathbb{E}_{Z_n \sim q_{\text{new}}(Z_n)}[\log p(y_n, Z_n; \phi, \theta)]$$

where q_{new} is the value of q from the previous E-step. We can either solve the maximization problem analytically (if tractable) or via gradient descent.

2.3 Properties

The EM algorithm has the following properties:

- *Monotonicity* - repeating the E and M steps never decreases the ELBO.
- *Converge* - the EM algorithm will eventually converge to a local optimum.

Like any iterative optimization algorithm with a non-convex objective, EM is not guaranteed to converge to the global optimum.

Example: Coin Flips (Continued)

Let's see how we would apply the EM algorithm to learn the unknown parameters ϕ_0 and ϕ_1 step-by-step.

We initialize ϕ_0 and ϕ_1 each to random numbers between 0 and 1.

E-Step For each datapoint y_n...

$$\begin{aligned} q_{\text{new}}(Z_n) &= \begin{bmatrix} p(Z_n = C_0 | y_n; \phi) \\ p(Z_n = C_1 | y_n; \phi) \end{bmatrix} \\ &\propto \begin{bmatrix} p(y_n | Z_n = C_0; \phi) p(Z_n = C_0) \\ p(y_n | Z_n = C_1; \phi) p(Z_n = C_1) \end{bmatrix} \\ &\propto \begin{bmatrix} \phi_0^{y_n} (1 - \phi_0)^{1 - y_n} \cdot 0.5 \\ \phi_1^{y_n} (1 - \phi_1)^{1 - y_n} \cdot 0.5 \end{bmatrix} \end{aligned}$$

Note that during the M-step the normalizing constant for the distribution vanishes upon taking the derivative. Thus, we can just ignore it here and keep things in terms of proportions.

M-step: Let's start by computing the term inside the expectation of the M-Step objective

for a single datapoint:

$$\begin{aligned}\log p(y_n, z_n; \phi) &= \log p(y_n|z_n;\phi)p(z_n;\phi) \\ &= \log\left[[0.5\phi_0^{y_n}(1-\phi_0)^{1-y_n}]^{1-z_n}[0.5\phi_1^{y_n}(1-\phi_1)^{1-y_n}]^{z_n}\right] \\ &= (1-z_n)[y_n\log\phi_0 + (1-y_n)\log(1-\phi_0)] \\ &+ \log 0.5 + z_n[y_n\log\phi_1 + (1-y_n)\log(1-\phi_1)]\end{aligned}$$

Now we calculate the entire objective, simplifying for each possible value of Z_n (0, 1):

$$\begin{aligned}&\sum_{n=1}^{N} q_{\text{new},0}(Z_n)(\log 0.5 + y_n\log\phi_0 + (1-y_n)\log(1-\phi_0)) \\ &+ q_{\text{new},1}(Z_n)(\log 0.5 + y_n\log\phi_1 + (1-y_n)\log(1-\phi_1))\end{aligned}$$

where we index into $q(Z_n)$ since we represented it as a vector, but $q_{\text{new},i}(Z_n) = q(Z_n = i)$.

Now we can solve for the values of ϕ_0 and ϕ_1 that maximize the objective.

$$\begin{aligned}\frac{\partial \text{ELBO}}{\partial \phi_0} &= \sum_{n=1}^{N} q_{\text{new},0}\left(\frac{y_n}{\phi_0} - \frac{1-y_n}{1-\phi_0}\right) = 0 \\ \frac{\partial \text{ELBO}}{\partial \phi_1} &= \sum_{n=1}^{N} q_{\text{new},1}\left(\frac{y_n}{\phi_1} - \frac{1-y_n}{1-\phi_1}\right) = 0\end{aligned}$$

Doing some algebra we find that:

$$\begin{aligned}\phi_0 &= \frac{\sum_{n=1}^{N} q_{\text{new},0}\cdot y_n}{\sum_{n=1}^{N} q_{\text{new},0}} \\ \phi_1 &= \frac{\sum_{n=1}^{N} q_{\text{new},1}\cdot y_n}{\sum_{n=1}^{N} q_{\text{new},1}}\end{aligned}$$

Thus, the full EM algorithm for this example works as follows:

1. Initialize ϕ_0 and ϕ_1 to random numbers between 0 and 1.
2. Update the distribution q according to its update equation we derived above.
3. Update ϕ_0 and ϕ_1 according to their update equations we derived above.
4. Repeat steps 2 and 3 until convergence.

Practice Problem: Hardest CS Class

You run a poll of the CS181 teaching staff asking how many hours a week they spend doing work for their hardest CS class. There are only 3 CS classes at Harvard and every member of the teaching staff is taking exactly one of the three. *Using this, can you figure out which staff members are in the same class?*

Staff Member	Weekly HW for hardest class
Charu	15
Skyler	21
Oliver	12
Alex	7
Kat	28
... (etc.)	

Data gathered

Here, we have some observed data $\{x_n\}_{n=1}^N$ corresponding to the hours reported by each staff member.

Exercise 1. *What are the latent variables $\mathbf{Z}_n$ in this scenario?*

Solution. $\mathbf{Z}_n$ is the class that that the nth staff member reported hours for. Note that while $\mathbf{Z}_n$ is *unknown*, it influences the observed data (ie. some classes have a heavier workload than others, so the CS class a staff member is taking impacts the hours they report). □

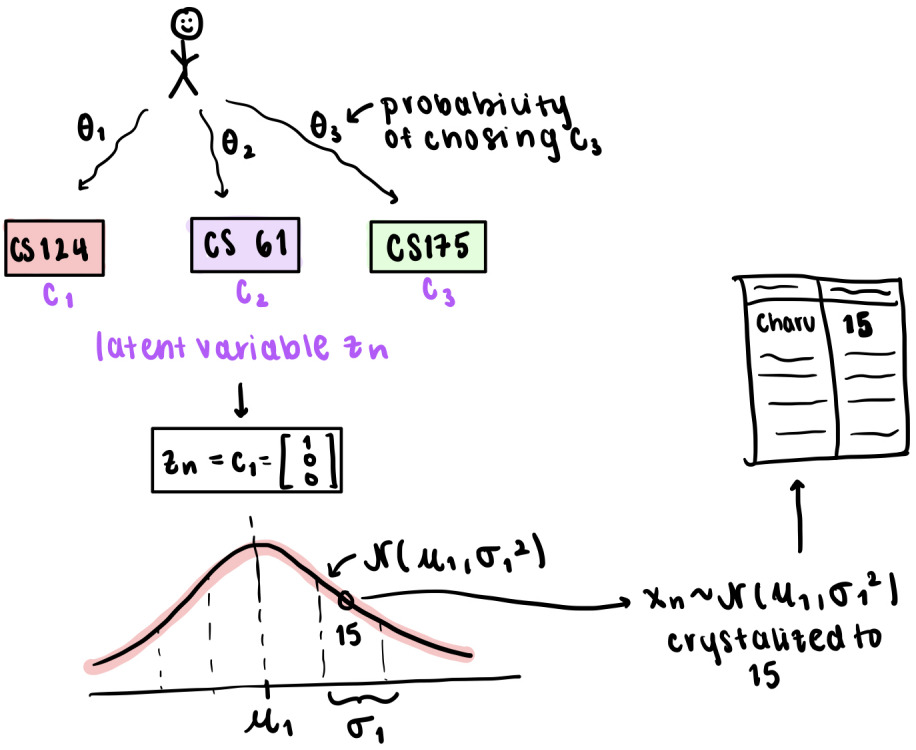
There are K possible values for each $\mathbf{Z}_n$, denoted $\{C_k\}_{k=1}^K$ where each C_k is a one-hot encoded vector of length K. In our example, $K = 3$ with $C_1 = [1, 0, 0]$ corresponding to the class CS124, $C_2 = [0, 1, 0]$ corresponding to the class CS61, and $C_3 = [0, 0, 1]$ corresponding to the class CS175. We can think of every observation x_n as being generated by the following process:

- Sample latent class $\mathbf{Z}_n$ from $\boldsymbol{\theta}$, the categorical distribution over $\{C_k\}_{k=1}^K$ s.t. $p(\mathbf{Z} = C_k; \boldsymbol{\theta}) = \theta_k$. Call this sampled latent class C_S.
- Given that $\mathbf{Z}_n = C_S$, sample y_n from the distribution

  $$p(x|\mathbf{z} = C_S; \mathbf{w})$$

  This conditional distribution is a modeling assumption (which means we will give it to you in this class), and is specified using unknown parameters $\mathbf{w}$.

  In our example, it might make sense to assume $x \sim p(x|\mathbf{z} = C_k) = \mathcal{N}(x; \mu_k, \sigma_k)$, where μ_k, Σ_k are the unknown mean and covariance of the number of hours the k-th class takes a week.



The hypothetical data generating process

Exercise 2. *In our example, what is an intuitive explanation of what the vector $\boldsymbol{\theta}$ represents?*

Solution. The proportion of people taking each class. □

Exercise 3. *You are told that $z_n = C_k | x_n \sim \frac{\theta_k \mathcal{N}(\mu_k, \sigma_k)}{Constant}$. Using this, write down the E-Step update.*

Solution.

$$q_{\text{new}}(Z_n) = \begin{bmatrix} p(z_n = C_1 | x_n) \\ p(z_n = C_2 | x_n) \\ p(z_n = C_3 | x_n) \end{bmatrix} \propto \begin{bmatrix} \theta_1 \mathcal{N}(\mu_1, \sigma_1) \\ \theta_2 \mathcal{N}(\mu_2, \sigma_2) \\ \theta_3 \mathcal{N}(\mu_3, \sigma_3) \end{bmatrix} \tag{1}$$

□

Exercise 4. *Write down the objective you will be optimizing in the M-Step.*

Solution.

For notational simplicity we will denote $q_{\text{new},k}(Z_n)$ as $q_{n,k}$. The following is the objective of the M-Step:

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{n,k}\ln\theta_k + q_{n,k}\ln\mathcal{N}(x_n;\mu_k,\sigma_k) \tag{2}$$

□

Exercise 5. *What are the parameters we are optimizing in the M-Step?*

Solution. We are optimising $\theta_1,\theta_2,\theta_3,\mu_1,\mu_2,\mu_3,\sigma_1,\sigma_2,\sigma_3$ in the M-Step. □

Exercise 6. *The MLE for μ_k and σ_k are $\frac{\sum_{n=1}^{N} q_{new,k}x_n}{\sum_{n=1}^{N} q_{new,k}}$ and $\frac{\sum_{n=1}^{N} q_{new,k}(x_n-\mu_k^{(t+1)})^2}{\sum_{n=1}^{N} q_{new,k}}$ respectively.*
(a) What are intuitive interpretations of these MLEs?
(b) Write out the steps of the EM algorithm given the MLEs above and your answers to the previous questions.

Solution. (a) The MLE for μ_k looks like an average of the reported hours of each TF weighted by the probability that the datapoint belongs to class k. The MLE for σ_k looks a bit like the empirical variance but with each squared difference between average reported hours and mean reported hours for that class weighted by the probability that the datapoint belongs to class k.
(b) The steps of the EM algorithm are as follows:

1. Initialize $\theta_2,\theta_3,\mu_1,\mu_2,\mu_3,\sigma_1,\sigma_2,\sigma_3$. Perhaps we initialize the μs to random numbers from the observations, the σs to be equal to the variance of the observed data, and the θ's to each be $\frac{1}{3}$ (a uniform distribution).
2. Calculate the q distribution as in Exercise 3 using the last computed values of $\theta_2,\theta_3,\mu_1,\mu_2,\mu_3,\sigma_1,\sigma_2,\sigma_3$.
3. Update the values of $\theta_2,\theta_3,\mu_1,\mu_2,\mu_3,\sigma_1,\sigma_2,\sigma_3$ using the MLE expressions given in this question where the $q_{\text{new},k}$s are the ones found in the previous step.
4. Repeat steps 2 and 3 until convergence (the MLE parameters don't change much anymore).

(Extra Content:) How did we actually find the MLEs? Since we require that $\sum_k \theta_k = 1$ in order to have a valid probability distribution, we must use Lagrange multipliers to incorporate this constraint before optimizing our parameters. Thus, the equation we want to optimize is:

$$\sum_{n=1}^{N}\sum_{k=1}^{K} q_{n,k}\ln\theta_k + q_{n,k}\ln\mathcal{N}(x_n;\boldsymbol{\mu}_k,\boldsymbol{\sigma}_k) - \lambda\left(\sum_{k=1}^{K}\theta_k - 1\right)$$

Now lets find the MLE of θ_k. We first take the derivative of the above expression wrt θ_k:

$$\sum_{n=1}^{N} q_{n,k} \cdot \frac{1}{\theta_k} - \lambda = 0 \implies \frac{\sum_{n=1}^{N} q_{n,k}}{\lambda} = \theta_k$$

Note that this implies $\sum_{k=1}^{K} \theta_k = \frac{\sum_{k=1}^{K}\sum_{n=1}^{K} q_{n,k}}{\lambda}$. By the constraint on mixture probabilities, $\sum_{k=1}^{K} \theta_k = 1$, so $\frac{\sum_{k=1}^{K}\sum_{n=1}^{K} q_{n,k}}{\lambda} = 1 \implies \lambda = \sum_{k=1}^{K}\sum_{n=1}^{N} q_{n,k}$. Plugging this back in we get

$$\theta_k = \frac{\sum_{n=1}^{N} q_{n,k}}{\sum_{k=1}^{K}\sum_{n=1}^{N} q_{n,k}} = \frac{\sum_{n=1}^{N} q_{n,k}}{N}$$

We can follow a similar process to find the MLE for μ_k and σ_k:

$$\mu_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{n,k} x_n}{\sum_{n=1}^{N} q_{n,k}} \tag{3}$$

$$\sigma_k^{(t+1)} = \frac{\sum_{n=1}^{N} q_{n,k} (x_n - \mu_k^{(t+1)})^2}{\sum_{n=1}^{N} q_{n,k}} \tag{4}$$

□

Exercise 7. *How could you use this to figure out which staff members are in the same class?*

Solution. Note that once the EM algorithm converges, $q_{\text{new}}(Z_n)$ gives us a pretty good estimate of the probability that a TF is in each of the three classes given the observed data. The index of the highest entry in this vector corresponds to the class that the TF is most likely to be in. Thus, we can do a hard-assignment of TFs to classes and then see who is in the same class! □

This latent variable model is called a Gaussian Mixture Model (GMM). GMMs with multivariate normals are discussed in section 9.5 of the textbook.

Example: Coin Flips 2 (Optional)

Consider the following data generation process: the setup is the same as above, but instead of flipping the chosen coin once, we flip it 10 times before choosing a new coin. Let our data be $Y_n \in \{0, 1, \ldots, 10\}$. Let our unknown parameters, as above, be $\boldsymbol{\phi} = (\phi_0, \phi_1)$.

Exercise 8. *Find an appropriate choice of latent variables Z_n and determine the distribution $p(Y_n|Z_n; \boldsymbol{\phi})$.*

Solution. Let our unknown parameters be ϕ_0, ϕ_1. Upon knowing which coin was chosen out of the hat, we know which parameter ϕ_k our data is distributed according to. In particular, $Z_n \sim \text{Bin}(0.5)$. Given $Z_n = C_k$ we know that $Y_n|Z_n \sim \text{Bin}(10, \phi_k)$. □

Exercise 9. *Write down the E-Step update.*

Solution. For each datapoint Y_n, we have that

$$\begin{aligned} q(Z_n) &= \begin{bmatrix} p(Z_n = C_0 | Y_n; \phi) \\ p(Z_n = C_1 | Y_n; \phi) \end{bmatrix} \\ &\propto \begin{bmatrix} p(Y_n | Z_n = C_0; \phi) p(Z_n = C_0) \\ p(Y_n | Z_n = C_1; \phi) p(Z_n = C_1) \end{bmatrix} \\ &\propto \begin{bmatrix} \phi_0^{Y_n} (1 - \phi_0)^{10 - Y_n} \cdot 0.5 \\ \phi_1^{Y_n} (1 - \phi_1)^{10 - Y_n} \cdot 0.5 \end{bmatrix} \end{aligned}$$

□

Exercise 10. *Write down the objective you are optimizing in the M-step. Take the derivatives of this objective to determine the updates.*

Solution.

$$\begin{aligned} \sum_{n=1}^{N} \mathbb{E}_{Z_n \sim q_{\text{new}}(Z_n)} [\log p(Y_n, Z_n; \phi)] = \sum_{n=1}^{N} & q_{\text{new},0}(Z_n)(\log 0.5 + Y_n \log \phi_0 + (10 - Y_n) \log(1 - \phi_0)) \\ & + q_{\text{new},0}, 1(Z_n)(\log 0.5 + Y_n \log \phi_1 + (1 - Y_n) \log(10 - \phi_1)) \end{aligned}$$

Taking partial derivatives, we find that

$$\begin{aligned} \frac{\partial \text{ELBO}}{\partial \phi_0} &= \sum_{n=1}^{N} q_{\text{new},0} \left(\frac{Y_n}{\phi_0} - \frac{10 - Y_n}{1 - \phi_0} \right) \\ \frac{\partial \text{ELBO}}{\partial \phi_1} &= \sum_{n=1}^{N} q_{\text{new},1} \left(\frac{Y_n}{\phi_1} - \frac{10 - Y_n}{1 - \phi_1} \right) \end{aligned}$$

Hence, we can determine the update by setting the derivative to 0 and solving for ϕ_k.

$$\phi_{k,new} \leftarrow \frac{\sum_{n=1}^{N} q_{\text{new},k} \cdot Y_n}{\sum_{n=1}^{N} q_{\text{new},k}}$$

Interestingly enough, the update is the exact same as when we made a single coin flip! □

3 Principal Component Analysis

3.1 Motivation

In many supervised learning problems, we try to find rich features that increase the expressivity of our model. In practice, this often involves using basis functions to transform the model input into a higher dimensional space (eg. given data x, using x and x^2 as features, or using features learned by a neural network). However, sometimes we want to *reduce* the dimensionality of our data.

Exercise 11. *Why would we want to reduce the dimensionality of our data? Can you think of example cases?*

Solution. There can be several reasons:

- Fewer features are easier to interpret: we might want to know why our model outputs a certain diagnosis, and only some of the patient record details will be relevant.
- Models with fewer features are easier to handle computationally.
- Our data might be arbitrarily high-dimensional because of noise, so we would like to access the lower-dimensional signal from the data.

□

When working with high-dimensional data, it is likely that not every feature will give us completely new information, for example, multiple features may be correlated. The idea that we might be able to reduce the dimensionality of our data by eliminating redundant information is a powerful one, and is the essence of *Principal Component Analysis (PCA)*.

Recall that our data $\mathbf{X}$ is an N x D dimensional matrix where each row corresponds to a datapoint and each column corresponds to a feature. The goal of PCA is to re-express $\mathbf{X}$ in terms of a new basis such that every column of this transformed matrix now gives us completely new information (ie. the features are *linearly independent*). The columns of the corresponding change-of-basis matrix are called *principal components*. The principal components are linearly independent eigenvectors, whose eigenvalues correspond to the amount of variance that the eigenvector takes up of the original data. After re-expressing $\mathbf{X}$ in terms of the principal components, we can make a decision about how many columns of the new matrix we want to keep depending on how much of the original variation we want to preserve and what our use case is. For example, if we wanted to plot the resulting data we might only keep the first two principal components.

3.2 Finding Lower Dimensional Representations

How do we determine the subspace that of vectors that contain majority of the variance of our data? We answer this question by focusing on how we find the best 1-dimensional subspace of which to project our data. Recall the definition of sample variance, where for any sample of data $\{\mathbf{x}_n\}_{n=1}^N$, we have

$$\text{sample variance} = \frac{1}{N-1}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})^2$$

We formalize this idea of finding vectors that are along axis of highest variance by searching for the vector $\mathbf{v}^*$ such that, when our data is projected onto this vector, has the highest sample variance.

$$\begin{aligned}
\mathbf{v}^* &= \text{argmax}_{\mathbf{v}} \frac{1}{N-1}\sum_{n=1}^{N}\left(\mathbf{x}_n^\top \mathbf{v} - \bar{\mathbf{x}}^\top \mathbf{v}\right)^2 \\
&= \text{argmax}_{\mathbf{v}} \frac{1}{N-1}\sum_{n=1}^{N}\mathbf{v}^\top(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top \mathbf{v} \\
&= \text{argmax}_{\mathbf{v}} \frac{N}{N-1}\mathbf{v}^\top \mathbf{S}\mathbf{v} \\
&= \text{argmax}_{\mathbf{v}} \mathbf{v}^\top \mathbf{S}\mathbf{v}
\end{aligned}$$

Where $\mathbf{S}$ is the empirical covariance matrix.

Note that we must include the constraint that $\|\mathbf{v}\|^2 = 1$, otherwise the optimization problem has no solution.

Seeing that this is an optimization problem with constraints, we know that the optima are at the stationary points of the Lagrangian function $\mathcal{L}(\mathbf{v}, \alpha) = \mathbf{v}^\top \mathbf{S}\mathbf{v} - \alpha(1 - \|v\|^2)$. Thus by Lagrange's Theorem we have that for some scalar α, the optimum must satisfy

$$\frac{\partial \mathcal{L}(\mathbf{v}, \alpha)}{\partial \mathbf{v}} = \mathbf{S}\mathbf{v} - \alpha\mathbf{v} = 0$$

which implies that all stationary points are eigenvectors of the empirical covariance matrix $\mathbf{S}$. From linear algebra, we know that the covariance matrix is always diagonalizable by the set of eigenvectors. Each eigenvalue corresponds to the variance that the corresponding eigenvector takes up. Hence, in order to obtain transform the data into the $K < D$ most expressive features, one should choose the eigenvectors that correspond to the K largest eigenvalues.

To summarize, to perform PCA on a dataset $\mathbf{X} = \{\mathbf{x}\}$ and work with transformed data $\{\mathbf{z}\}$:

1. Calculate the empirical covariance matrix:

$$\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$$

2. Decide how many dimensions K out of the original D we want to keep in the final representation.

3. Find the K largest eigenvalues of $\mathbf{S}$. The $D \times 1$ eigenvectors $(\mathbf{u}_1, \ldots, \mathbf{u}_K)$ corresponding to these eigenvalues will be our lower-dimensional basis. The corresponding basis transformation/projection matrix is thus

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_K \end{bmatrix}$$

4. Reduce the dimensionality of a data point $\mathbf{x}$ by projecting it onto the new basis, yielding a new reconstructed vector $\mathbf{z}$:

$$\mathbf{z} = \mathbf{U}^T\mathbf{x}$$

3.3 Reconstruction Loss

We define the *reconstruction loss* to be the distance from the approximation of each $\mathbf{x}_n$ using the new basis to $\mathbf{x}_n$ itself:

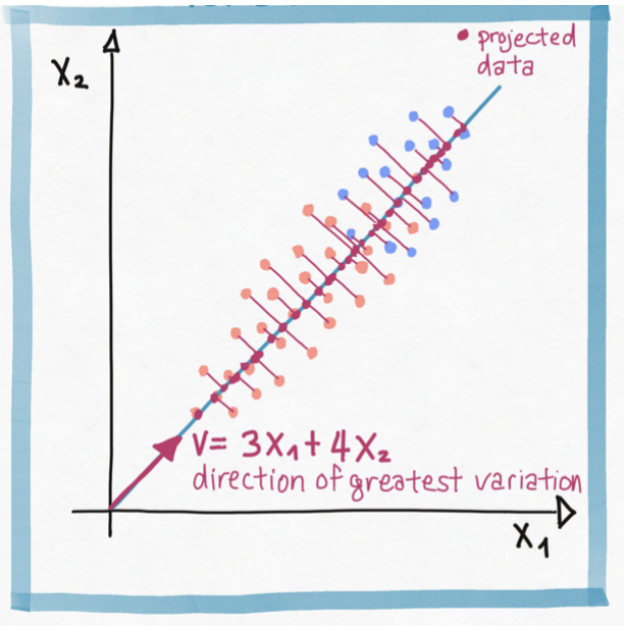
$$\mathcal{L}(\mathbf{U}) = \frac{1}{N}\sum_{n=1}^{N}||\mathbf{x}_n - \hat{\mathbf{x}}_n||^2$$

Where $\hat{\mathbf{x}}_n$ is the reconstructed vector from a set of linear transformations. We will show that minimizing the reconstruction loss yields the same results as maximizing the variance captured in a subspace. Note that a reconstruction will never be accurate if our data is not centered. Hence, we will work with centered data $\mathbf{x}_n \leftarrow \mathbf{x}_n - \bar{\mathbf{x}}$. This way, we can have that

$$\hat{\mathbf{x}}_n = \mathbf{U}\mathbf{U}^\top\mathbf{x}_n$$

First note that the solution to the above loss is not unique! To find a unique solution, we impose the constraint as before: each column must be normalized. Furthermore, let us impose that all columns are perpendicular to each other, namely any distinct columns k, k' in $\mathbf{U}$, $\mathbf{u}_k \cdot \mathbf{u}_k = 1$ and $\mathbf{u}_k \cdot \mathbf{u}_{k'} = 0$. We have that $\mathbf{u}_k^T\mathbf{x}_n = z_{n,k}$.

$$\mathcal{L}(\mathbf{U}) = \frac{1}{N}\sum_{n=1}^{N}\|\mathbf{x}_n - \mathbf{U}\mathbf{U}^\top\mathbf{x}_n\|^2 \text{ s.t. } \mathbf{U} \text{ is orthonormal}$$



Source: Weiwei Pan. The red lines sum to the reconstruction loss.

Note that if $K = D$, then we could perfectly reconstruct $\mathbf{x}_n$ since we'd be able to preserve all the features. Thus, $\mathbf{x}_n = \sum_{k=1}^{D} z_{n,k}\mathbf{u}_k$. Thus, we can simplify our loss as follows:

$$\begin{aligned}
\mathcal{L}(\{\mathbf{z}_n\}, \mathbf{U}) &= \frac{1}{N}\sum_{n=1}^{N}\left\|\sum_{k=1}^{D} z_{n,k}\mathbf{u}_k - \sum_{k=1}^{K} z_{n,k}\mathbf{u}_k\right\|^2 \\
&= \frac{1}{N}\sum_{n=1}^{N}\left\|\sum_{k=K+1}^{D} z_{n,k}\mathbf{u}_k\right\|^2 \\
&= \frac{1}{N}\sum_{k=K+1}^{D}\sum_{n=1}^{N}\mathbf{u}_k^\top\mathbf{x}_n\mathbf{x}_n^\top\mathbf{u}_k \\
&= \sum_{k=K+1}^{D}\mathbf{u}_k^\top\left[\frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^\top\right]\mathbf{u}_k
\end{aligned}$$

Which is exactly the same objective as when we were looking for the vectors to maximize explained variance!

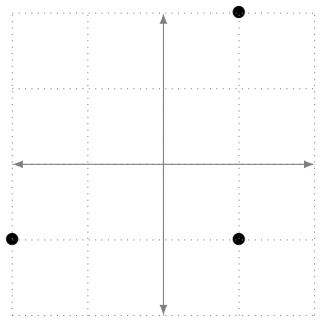
Exercise 12. *You are given the following data set:*

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} -2 \\ -1 \end{bmatrix}$$

You would like to use PCA to find a 1-dimensional representation of the data.

1. *Plot the data set.*
2. *Compute the empirical covariance matrix* $\mathbf{S}$.
3. *You find that* $\mathbf{S}$ *has eigenvector* $[-1\ 1]^\top$ *with eigenvalue* 1 *and eigenvector* $[1\ 1]^\top$ *with eigenvalue* 3. *What is the (normalized) basis vector* $\mathbf{u}_1$ *of your 1-dimensional representation? Add the basis vector* $\mathbf{u}_1$ *to your plot.*
4. *Compute the coefficients* z_1, z_2, z_3. *Add the lower-dimensional representations* $z_1\mathbf{u}_1, z_2\mathbf{u}_1, z_3\mathbf{u}_1$ *to your plot. Based on your plot, what is the relationship between* $z_i\mathbf{u}_1$ *and* $\mathbf{x}_i$ *with respect to the new basis?*
5. *Based on your plot, what would happen if you chose the unused eigenvector to be your basis vector?*

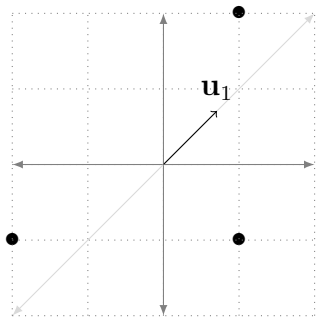
Solution. 1.



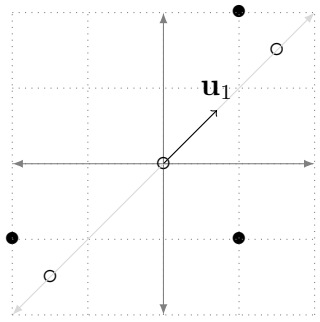
2.

$$\mathbf{S} = \frac{1}{3}\mathbf{X}^\top\mathbf{X} = \frac{1}{3}\begin{bmatrix} 1 & -1 \\ 1 & 2 \\ -2 & -1 \end{bmatrix}^\top \begin{bmatrix} 1 & -1 \\ 1 & 2 \\ -2 & -1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

3. We select the eigenvectors with the largest eigenvalues for our basis, so our basis will contain a scalar multiple of $[1\ 1]^\top$. Normalizing $[1\ 1]^\top$ gives us that $\mathbf{u}_1 = [\frac{\sqrt{2}}{2}\ \frac{\sqrt{2}}{2}]^\top$.



4.
$$z_1 = \mathbf{x}_1^\top \mathbf{u}_1 = 0, \quad z_2 = \mathbf{x}_2^\top \mathbf{u}_1 = \frac{3\sqrt{2}}{2}, \quad z_3 = \mathbf{x}_3^\top \mathbf{u}_1 = -\frac{3\sqrt{2}}{2}$$
The open circles in the plot represent the lower-dimensional representation:



$z_i \mathbf{u}_1$ is the projection of $\mathbf{x}_i$ onto the basis vector.

5. If we chose $[-1\ 1]^\top$ to be the basis of our new representation, then the representation would capture less of the variance in the data. For example, $\mathbf{x}_2$ and $\mathbf{x}_3$ would be represented by the same point.

□

Exercise 13. *(Optional) Suppose that our data are centered (i.e., have sample mean 0). Recall that in lecture, we showed that, when optimizing over (semi)-orthogonal matrices $\mathbf{U} \in \mathbb{R}^{m \times d}$ (i.e., where $\mathbf{U}^T\mathbf{U} = I$) to minimize the reconstruction loss,*

$$\mathcal{L}(\mathbf{U}) = \frac{1}{n}\sum_{i=1}^{n} ||\mathbf{x_i} - \mathbf{U}\mathbf{U}^T\mathbf{x_i}||_2^2,$$

we found that $\mathbf{U}_d$, the matrix whose first d columns are (in order) the top d eigenvectors of the empirical covariance matrix $\mathbf{\Sigma} = \frac{1}{n}\mathbf{X}^T\mathbf{X}$, will achieve the minimum (i.e., $\mathbf{z} = \mathbf{U}^T\mathbf{x}$ is the projection of $\mathbf{x}$ into d dimensions, and $\mathbf{U}\mathbf{z}$ is its reconstruction in $\mathbb{R}^m$). Show this, in general, for any d.

Hint: you may use the following theorem: [Courant-Fischer] Let $\mathbf{A}$ be a symmetric $n \times n$ matrix with eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$ and corresponding eigenvectors $\mathbf{v_1}, \ldots, \mathbf{v_n}$. Then

$$\lambda_1 = \min_{||x||=1} \mathbf{x}^T\mathbf{A}\mathbf{x}$$

$$\lambda_2 = \min_{||\mathbf{x}||=1, \mathbf{x}\perp\mathbf{v}_1} \mathbf{x}^T\mathbf{A}\mathbf{x}$$

$$\vdots$$

$$\lambda_i = \min_{||\mathbf{x}||=1, \mathbf{x}\perp\mathbf{v}_1, \mathbf{x}\perp\mathbf{v}_2, \ldots, \mathbf{x}\perp\mathbf{v}_{i-1}} \mathbf{x}^T\mathbf{A}\mathbf{x}$$

$$\vdots$$

$$\lambda_n = \min_{||\mathbf{x}||=1, \mathbf{x}\perp\mathbf{v}_1, \mathbf{x}\perp\mathbf{v}_2, \ldots, \mathbf{x}\perp\mathbf{v}_{n-1}} \mathbf{x}^T\mathbf{A}\mathbf{x}.$$

Solution. Let $\tilde{\mathbf{U}} \in \mathbb{R}^{m \times m}$ be an orthogonal matrix, and we will let $\mathbf{U}$ be the matrix comprising $\tilde{\mathbf{U}}$'s first d columns. Letting $\tilde{\mathbf{U}}^{(k)}$ denote the kth column of $\tilde{\mathbf{U}}$, the orthogonality of $\tilde{\mathbf{U}}$ implies that the $\tilde{\mathbf{U}}^{(k)}$ form a basis in $\mathbb{R}^m$ so that, for each $\mathbf{x}_i$, we can find $\mathbf{z}_i \in \mathbb{R}^m$ such that

$$\mathbf{x}_i = \sum_{k=1}^{m} \mathbf{z}_{i,k}\tilde{\mathbf{U}}^{(k)}.$$

The orthogonality of $\tilde{\mathbf{U}}$ implies that $\tilde{\mathbf{U}}^T\mathbf{x}_i = (\mathbf{z}_{i,1}, \ldots, \mathbf{z}_{i,m})$ and $\mathbf{U}^T\mathbf{x}_i = (\mathbf{z}_{i,1}, \ldots, \mathbf{z}_{i,d})$. Hence, we can write the reconstruction loss as

$$\mathcal{L}(\mathbf{U}) = \frac{1}{n}\sum_{i=1}^{n} ||\sum_{k=1}^{m} \mathbf{z}_{i,k}\tilde{\mathbf{U}}^{(k)} - \sum_{k=1}^{d} \tilde{\mathbf{U}}^{(k)}\mathbf{z}_{i,k}||_2^2 = \frac{1}{n}\sum_{i=1}^{n} ||\sum_{k=1}^{m} \mathbf{z}_{i,k}\tilde{\mathbf{U}}^{(k)} - \sum_{k=1}^{d} \mathbf{z}_{i,k}\tilde{\mathbf{U}}^{(k)}||_2^2$$

$$= \frac{1}{n}\sum_{i=1}^{n} ||\sum_{k=d+1}^{m} \mathbf{z}_{i,k}\tilde{\mathbf{U}}^{(k)}||_2^2 = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=d+1}^{m} \mathbf{z}_{i,k}^2 = \frac{1}{n}\sum_{i=1}^{n}\sum_{k=d+1}^{m} \left(\tilde{\mathbf{U}}^{(k)T}\mathbf{x}_i\right)\left(\mathbf{x}_i^T\tilde{\mathbf{U}}^{(k)}\right)$$

$$= \sum_{k=d+1}^{m} \tilde{\mathbf{U}}^{(k)T} \left(\frac{1}{n} \mathbf{X}^T \mathbf{X} \right) \tilde{\mathbf{U}}^{(k)} = \sum_{k=d+1}^{m} \tilde{\mathbf{U}}^{(k)T} \mathbf{\Sigma} \tilde{\mathbf{U}}^{(k)},$$

where $\mathbf{\Sigma}$ denotes the empirical covariance matrix.

Let $\mathbf{u}_1, \ldots, \mathbf{u}_m$ denote the (unit length) eigenvectors of $\mathbf{\Sigma}$ (ordered by increasing corresponding eigenvalue). When $d + 1 = m$, there's only one element in the sum, and since we know that $\tilde{\mathbf{U}}$ is orthogonal, we must have that $||\tilde{\mathbf{U}}^{(k)}||_2 = 1$ and thus Courant-Fischer says that $\tilde{\mathbf{U}}^{(k)} = \mathbf{u}_1$ will minimize. When $d + 1 = m - 1$, there will be two elements in the sum:

$$\tilde{\mathbf{U}}^{(m-1)T} \mathbf{\Sigma} \tilde{\mathbf{U}}^{(m-1)} + \tilde{\mathbf{U}}^{(m)T} \Sigma \tilde{\mathbf{U}}^{(m)}.$$

Consider solving this optimization problem by first picking $\tilde{\mathbf{U}}^{(m)}$ and then $\tilde{\mathbf{U}}^{(m-1)}$. Now, we either choose $\tilde{\mathbf{U}}^{(m)}$ to be $\mathbf{u}_1$ or we do not. If we choose $\tilde{\mathbf{U}}^{(m)} = \mathbf{u}_1$, then Courant-Fischer implies that picking $\tilde{\mathbf{U}}^{(m-1)} = \mathbf{u}_2$ will minimize. If we do not choose $\tilde{\mathbf{U}}^{(m)}$ to be $\mathbf{u}_1$, then Courant-Fischer implies that picking $\tilde{\mathbf{U}}^{(m-1)} = \mathbf{u}_1$ will minimize and that, if we did not pick $\tilde{\mathbf{U}}^{(m)}$ to be $\mathbf{u}_2$, we could've achieved an even lower value more by picking it to be so. Thus, in either case, we will take $\tilde{\mathbf{U}}^{(m)}$ and $\tilde{\mathbf{U}}^{(m-1)}$ to be $\mathbf{u}_1$ and $\mathbf{u}_2$.

Proceeding iteratively like this shows that we will chose $\tilde{\mathbf{U}}^{(d+1)}, \ldots, \tilde{\mathbf{U}}^{(m)}$ to be $\mathbf{u}_1, \ldots, \mathbf{u}_{m-d}$. Now, note that these are the columns we *exclude* from the matrix $\tilde{\mathbf{U}}$ when we construct $\mathbf{U}$ (which will be the first d column of $\tilde{\mathbf{U}}$). Since $\mathbf{\Sigma}$ is symmetric, the eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_n$ are orthogonal (or can be chosen to be in the case of repeated eigenvalues), taking $\mathbf{U}$'s columns to be $\mathbf{u}_{m-d+1}, \ldots, \mathbf{u}_m$ (i.e., the top d eigenvectors) indeed will minimize the reconstruction loss, as desired. □