CS5339 Lecture Notes #11: Unsupervised Learning

Jonathan Scarlett

April 3, 2021

Useful references:

- MIT lecture notes,[1] lectures 15 and 16
- Supplementary notes lec16a.pdf and lec17a.pdf
- Chapter 9 of Bishop's "Pattern Recognition and Machine Learning" book
- Section 22.2, 24.1, and 24.4 of "Understanding Machine Learning" book

1 Clustering and the K-Means Algorithm

Clustering:

- So far we have focused on the problems of classification (i.e., predicting the label $y \in \{-1, 1\}$ associated with an input $\mathbf{x}$) and regression (i.e., predicting the label $y \in \mathbb{R}$ associated with an input $\mathbf{x}$). These are examples of *supervised learning*, in the sense that the true labels of the training inputs are known.
- *Unsupervised learning* problems seek to learn something about $\mathcal{D} = \{\mathbf{x}_t\}_{t=1}^n$ when no labels are available (e.g., learn some structure in the data, or estimate a data distribution $P_{\mathbf{X}}$).
- A particularly common example is *clustering*, which seeks to group the inputs such that those within the same group are "more similar" compared to those in different groups.
  - e.g., group similar users together in a recommendation system; these similarities could then be used to help provide movie recommendations, etc.
- Compared to supervised learning, clustering is not a clearly-defined problem. There is no "ground truth", and there may be multiple reasonable groupings (e.g., think of how many ways people can be categorized into different groups!). We will look at only the most basic mathematical formulation.

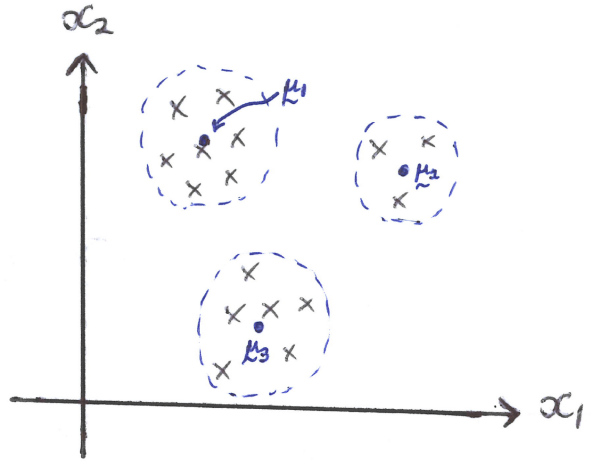
[1] `http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-867-machine-learning-fall-2006/lecture-notes/`

- If the number of clusters (groups) K is fixed in advance, then a natural goal is to seek a partition of the data set $\mathcal{D}_1 \cup \ldots \cup \mathcal{D}_K$, as well as an associated set of *cluster centers* $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$, such that the sum of distances

$$J(\{\mathcal{D}_j\}_{j=1}^K, \{\boldsymbol{\mu}_j\}_{j=1}^K) = \sum_{j=1}^{K} \sum_{\mathbf{x} \in \mathcal{D}_j} \|\mathbf{x} - \boldsymbol{\mu}_j\|^2.$$

  is as small as possible.

- A visual illustration:



- Unfortunately, finding such an optimal clustering is computationally challenging (NP-hard) in general.

The K-means algorithm:

- While finding the global minimum of $J(\{\mathcal{D}_j\}_{j=1}^k, \{\boldsymbol{\mu}_j\}_{j=1}^K)$ is hard, there are two related steps that are easy:

  1. Minimize $J(\{\mathcal{D}_j\}_{j=1}^k, \{\boldsymbol{\mu}_j\}_{j=1}^K)$ with respect to $\{\mathcal{D}_j\}_{j=1}^k$ for fixed $\{\boldsymbol{\mu}_j\}_{j=1}^K$;
  2. Minimize $J(\{\mathcal{D}_j\}_{j=1}^k, \{\boldsymbol{\mu}_j\}_{j=1}^K)$ with respect to $\{\boldsymbol{\mu}_j\}_{j=1}^K$ for fixed $\{\mathcal{D}_j\}_{j=1}^k$.

  The *K-means algorithm* alternates between these two steps until convergence. Note that "convergence" means that neither of the two steps produces any change in the centers $\{\boldsymbol{\mu}_j\}_{j=1}^K$ or assignments $\{\mathcal{D}_j\}_{j=1}^k$ (it can be shown that this always occurs eventually).

- In more detail, the first step assigns each point to the closest cluster center:

$$\mathcal{D}_j = \left\{ \mathbf{x} \in \mathcal{D} \,:\, j = \operatorname*{arg\,min}_{j'=1,\ldots,K} \|\mathbf{x} - \boldsymbol{\mu}_{j'}\|^2 \right\}, \tag{1}$$

  and the second step lets each cluster center be the average of all points in the cluster:

$$\boldsymbol{\mu}_j = \frac{1}{|\mathcal{D}_j|} \sum_{\mathbf{x} \in \mathcal{D}_j} \mathbf{x}. \tag{2}$$

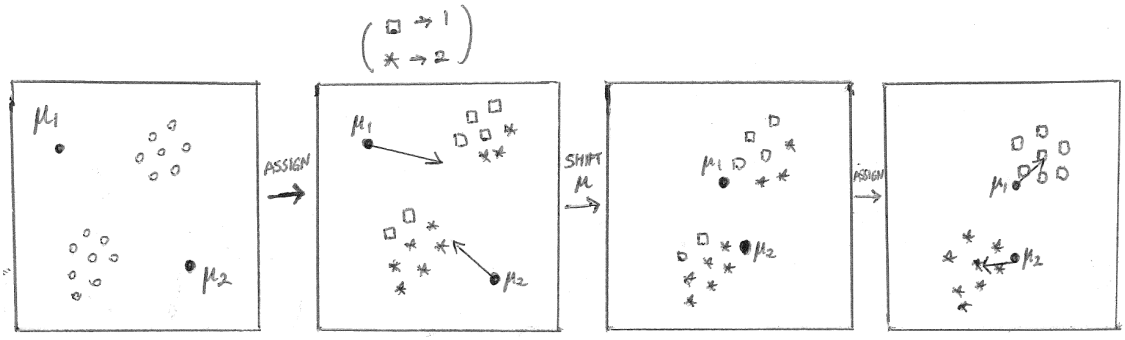
- The rule (1) clearly minimizes $J(\cdot, \{\boldsymbol{\mu}_j\}_{j=1}^K)$: Each $\mathbf{x}$ has to contribute one term of the form $\|\mathbf{x} - \boldsymbol{\mu}_j\|^2$, so we should choose the cluster index j to have the smallest such value.

- The optimality of (2) for minimizing $J(\{\mathcal{D}_j\}_{j=1}^k, \cdot)$ follows from a more general fact about second moments of random variables:

$$\begin{aligned}\mathbb{E}[\|\mathbf{W}-\mathbf{w}_0\|^2] &= \mathbb{E}\big[\|\mathbf{W}\|^2 - 2\langle \mathbf{W}, \mathbf{w}_0\rangle + \|\mathbf{w}_0\|^2\big] \\ &= \mathbb{E}\big[[\|\mathbf{W}\|^2\big] - 2\mathbf{w}_0^T\mathbb{E}[\mathbf{W}] + \|\mathbf{w}_0\|^2 \\ &\overset{(a)}{\geq} \mathbb{E}[\|\mathbf{W}-\mathbb{E}[\mathbf{W}]\|^2], \end{aligned} \tag{3}$$

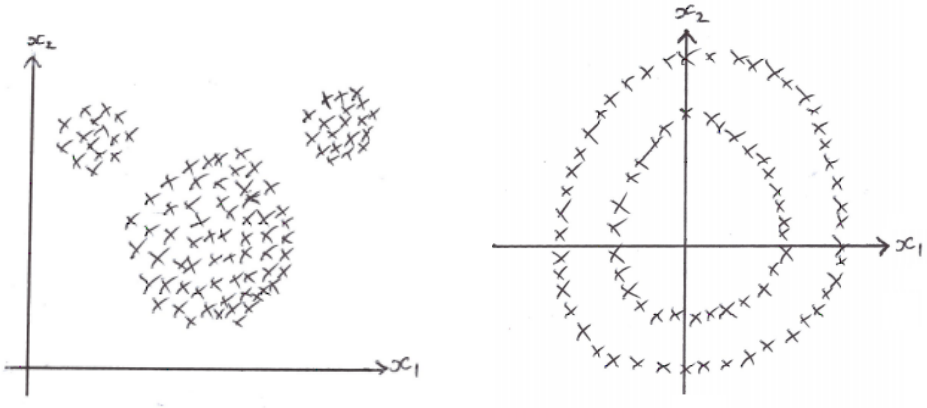
  where (a) follows since $\|\mathbf{w}_0\|^2 - 2\mathbf{w}_0^T\mathbb{E}[\mathbf{W}]$ is minimized at $\mathbf{w}_0 = \mathbb{E}[\mathbf{W}]$ (proved by setting the derivative to zero). To establish the optimality of (2), apply this property with $\mathbf{W}$ defined to be uniform over the points in the cluster (i.e., $\mathbf{W}$ equals each $\mathbf{x} \in \mathcal{D}_j$ with probability $\frac{1}{|\mathcal{D}_j|}$ each, and hence $\mathbb{E}[\mathbf{W}] = \frac{1}{|\mathcal{D}_j|}\sum_{\mathbf{x}\in\mathcal{D}_j}\mathbf{x}$ which matches (2)).

- Illustration of the algorithm:



- Procedures of this type are known as *alternating optimization*. We cannot guarantee that we get to the optimal clustering, but we do at least guarantee that $J(\{\mathcal{D}_j\}_{j=1}^k, \{\boldsymbol{\mu}_j\}_{j=1}^K)$ decreases after each iteration. This means (at least under mild technical conditions that we won't worry about here) that we converge to a *local minimum*.
  - Intuition: We cannot decrease J forever, as it is lower bounded by 0 – hence we must converge.

- While the K-means objective in natural, it does not always produce the "desired" or "obvious" clustering. Some examples:



- Left example: K-means tends to favor similar-size clusters, a property that doesn't hold here
- Right example: K-means doesn't detect these seemingly obvious patterns (though after an appropriate feature mapping or kernelization it might!)

Further discussion, and some other clustering algorithms very different to K-Means, can be found in Chapter 22 of the "Understanding Machine Learning" book.

2 Distribution Learning

Introduction.

- Another widespread unsupervised learning problem is distribution learning: *Given an unlabeled data set $\mathcal{D} = \{\mathbf{x}_t\}_{t=1}^n$, estimate a distribution $\hat{p}(\mathbf{x})$ that models the data well.*
- The learned distribution $\hat{p}(\mathbf{x})$ may be a PMF (discrete case) or a PDF (continuous case); in the latter, the problem is also referred to as *density estimation*.
- The notion of "modeling the data well" varies depending on the precise goal. For instance, if we posit that there is a true underlying distribution $p^*(\mathbf{x})$, there are several popular measures for measuring the closeness of $p^*(\mathbf{x})$ and $\hat{p}(\mathbf{x})$ (e.g., total variation, KL divergence, etc.), but we will not cover these.
- Distribution learning may be very difficult when the input length d is large, e.g., even a 10-dimensional distribution with complicated dependencies among the variables may be very hard to learn. However, it becomes feasible when we suitably limit the types of distribution under consideration.

Parametric methods and maximum-likelihood.

- *Parametric methods* consider classes of distributions $p(\mathbf{x}; \boldsymbol{\theta})$ that have some associated parameters collected into $\boldsymbol{\theta}$ (whose size may differ from $\mathbf{x}$). The learning algorithm finds some $\hat{\boldsymbol{\theta}}$, and the estimate of the distribution is $\hat{p}(\mathbf{x}) = p(\mathbf{x}; \hat{\boldsymbol{\theta}})$.
  - Example 1: If we want to estimate the mean of a Gaussian random vector with identity covariance matrix, then we can let $\boldsymbol{\theta} \in \mathbb{R}^d$ denote the mean, and then $p(\mathbf{x}; \boldsymbol{\theta})$ is the $N(\boldsymbol{\theta}, \mathbf{I})$ density function.
  - Example 2: If we want to estimate the *mean and covariance matrix* of a Gaussian random vector, we can let $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$, and then $p(\mathbf{x}; \boldsymbol{\theta})$ is the $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ density function. In this case, we have $d + d(d+1)/2$ parameters (the second term being due to the symmetry of $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$).
  - Example 3: The *univariate* case $d = 1$ is usually simpler. For instance, we might consider distributions like $x_t \sim \text{Bernoulli}(\theta)$ or $x_t \sim \text{Exponential}(\theta)$ and try to estimate θ (or similarly for distributions with multiple parameters, such as Gamma).
- The "go to" approach in parametric models is maximum likelihood estimation, which selects

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \prod_{t=1}^{n} p(\mathbf{x}_t; \boldsymbol{\theta}),$$

where the product arises under a standard assumption of independent data samples. Notes:

  - Mathematical analysis is often done assuming that the samples are drawn from $p(\mathbf{x}; \boldsymbol{\theta}^*)$ for some "true parameters" $\boldsymbol{\theta}^*$.

- Classical statistics theory gives important guarantees on this estimate under suitable assumptions, such as *consistency*: If the data points are drawn from $p(\mathbf{x}; \boldsymbol{\theta}^*)$ then we have $\hat{\boldsymbol{\theta}} \to \boldsymbol{\theta}^*$ as $n \to \infty$.
- For some (but certainly not all) models, the estimate is *unbiased*: $\mathbb{E}[\hat{\boldsymbol{\theta}}] = \boldsymbol{\theta}^*$ (we also encountered this concept in linear regression).
- Previously we were interested in maximum likelihood for *conditional* transition laws of the form $p(y|\mathbf{x}; \boldsymbol{\theta})$ (e.g., linear or logistic regression). The ideas that we are looking at here are quite similar, but we are in the unsupervised setting, so instead have $p(\mathbf{x}; \boldsymbol{\theta})$.

- Some examples are as follows (mostly explored in the tutorials):
  - The maximum-likelihood estimates of the mean and covariance matrix for the Gaussian case (i.e., $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$) are simply the *sample mean* and *sample covariance*:

$$\hat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{t=1}^{n} \mathbf{x}_i \tag{4}$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n} \sum_{t=1}^{n} (\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^T. \tag{5}$$

    A proof is given below for the special case that $d = 1$ (i.e., a univariate Gaussian, $N(\mu, \sigma^2)$).
  - The maximum-likelihood estimate of the parameter to the Bernoulli(θ) distribution ($d = 1$) is

$$\hat{\theta} = \frac{1}{n} \sum_{t=1}^{n} x_t,$$

    as one might expect intuitively (guess $\Pr[x = 1]$ to equal the proportion of 1's observed).
  - The maximum-likelihood estimate of the parameter to the Uniform($[0, \theta]$) distribution is

$$\hat{\theta} = \max_{t=1,\ldots,n} x_t.$$

    This is an example where the estimate is *biased*: $\mathbb{E}[\hat{\theta}] \neq \theta^*$ (more precisely, $\mathbb{E}[\hat{\theta}] < \theta^*$).

Derivation of the Gaussian maximum-likelihood estimator (1D case).

- In the 1D/univariate case ($d = 1$), the density of x is

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}.$$

- We are interested in $(\hat{\mu}, \hat{\sigma}^2) = \arg\max_{\mu,\sigma^2} \prod_{t=1}^{n} p(x_t; \mu, \sigma^2)$, but as usual it is more convenient to work with the log:

$$\begin{aligned}
(\hat{\mu}, \hat{\sigma}^2) &= \arg\max_{\mu,\sigma^2} \sum_{t=1}^{n} \log p(x_t; \mu, \sigma^2) \\
&= \arg\max_{\mu,\sigma^2} \sum_{t=1}^{n} \left(-\frac{1}{2} \log(2\pi\sigma^2) - \frac{(x_t - \mu)^2}{2\sigma^2} \right) \\
&= \arg\min_{\mu,\sigma^2} \frac{n}{2} \log(\sigma^2) + \frac{1}{2\sigma^2} \sum_{t=1}^{n} (x_t - \mu)^2,
\end{aligned}$$

where we switched the max to min by flipping the sign, and removed the $\log 2\pi$ term which just amounts to a constant that does not change the minimizer.

- Letting $v = \sigma^2$ and setting $\frac{\partial}{\partial v}\left(\frac{n}{2}\log v + \frac{1}{2v}\sum_{t=1}^{n}(x_t-\mu)^2\right) = 0$, we obtain $\sigma^2 = \frac{1}{n}\sum_{t=1}^{n}(x_t-\mu)^2$ (the details are omitted, but are straightforward).

- As for μ, notice that regardless of the value of $\sigma^2 > 0$, the above minimization is equivalent to minimizing $\frac{1}{n}\sum_{t=1}^{n}(x_t-\mu)^2$. By the result in (3) (with $\mathbf{W}$ equaling each x_t with probability $\frac{1}{n}$), it follows that the optimal choice is $\mu = \frac{1}{n}\sum_{t=1}^{n} x_t$ (alternatively, this can also be established by setting the derivative to zero).

- Combining the above, the maximum-likelihood estimate is

$$\hat{\mu} = \frac{1}{n}\sum_{t=1}^{n} x_t, \quad \hat{\sigma}^2 = \frac{1}{n}\sum_{t=1}^{n}(x_t-\hat{\mu})^2,$$

  which is a special case (namely, $d = 1$) of (4)–(5). The general case is proved similarly, but with some more tedious details.

Other concepts in unsupervised learning.

- Although we will not cover it in detail, unsupervised learning faces the very same challenges/concepts of overfitting, bias-variance trade-off, regularization, etc. as supervised learning.

- As an extreme example, if we try to fit a mixture of 10 Gaussians (see the next section) to 10 data points, the maximum likelihood solution will be attained in the limit of an infinitely-sharp peak (low-variance Gaussian) centered at each data point.

Non-parametric methods.

- *Non-parametric methods* seek to directly learn from data without adopting a specific parametric model.

- An example of such a method in *supervised* learning is K-nearest-neighbors.

- There are also important non-parametric methods in unsupervised learning, notably including *kernel density estimation* (KDE)[2] The idea is to let each data point increase the density more in nearby regions, while having less influence in further-away regions.

  – This can be viewed as generalizing the more familiar notion of *histograms*, in which each point influences the closest histogram bar, but has no influence on the other bars. In KDE, however, the influence tends to decay to zero more smoothly.

3 Hidden Variables and Mixture Models

- The distribution learning problem can be significantly complicated by the presence of *hidden variables*:[3]

$$p(\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}} p(\mathbf{x},\mathbf{z};\boldsymbol{\theta}).$$

[2]This is actually a different notion of "kernel" to our lecture on kernel methods.
[3]If $\mathbf{z}$ is continuous, then the summation should be replaced by an integral.

- Example 1: $\mathbf{x}$ is public information about a user, whereas $\mathbf{z}$ is (unknown) private information.
- Example 2: $\mathbf{x}$ contains pixels of a face image, $\mathbf{z}$ contains information about pose/scale/lighting.

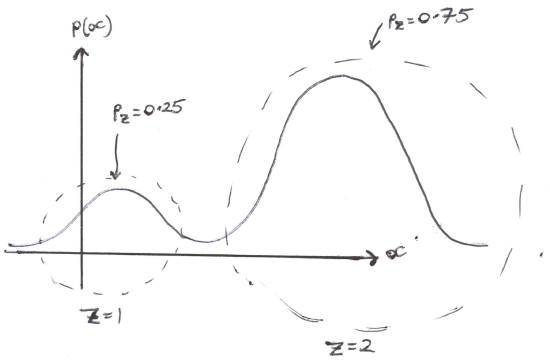
Sometimes the terminology *observed variables* and *latent variables* is used for $\mathbf{x}$ and $\mathbf{z}$ respectively. As usual, $\boldsymbol{\theta}$ is a set of parameters that the (joint) distribution depends on.

- A simple (but important) example is a *mixture model*, where $\mathbf{z}$ simply takes values in $\{1, \ldots, m\}$, $p(\mathbf{z}) = (p_1, \ldots, p_m)$ is some mass function, and

$$p(\mathbf{x}) = \sum_{z=1}^{m} p_z p(\mathbf{x}|z).$$

We use z in place of $\mathbf{z}$ here, since it is scalar (in general, $\mathbf{z}$ above could be a vector). We also omit the dependence on possible parameters $\boldsymbol{\theta}$ (for now).

- If each $p(\mathbf{x}|z)$ in the mixture model is a Gaussian distribution, then this is referred to as a *Gaussian mixture model* (GMM). A visual illustration:



- Toy example. Suppose $m = 2$, z indicates gender (so $p_z = 0.5$ for $z = 1, 2$), and $\mathbf{x}$ contains only a single feature indicating hair length. Conditioned on a given gender, a Gaussian distribution might be a reasonable assumption, but the *unconditional* distribution instead has two clear components.
- If gender is not specified in the data set, then it becomes a *hidden variable*.

- In the following, we will consider distributions $p(\mathbf{x}, \mathbf{z}; \boldsymbol{\theta})$ and $p(\mathbf{x}; \boldsymbol{\theta})$ parametrized by $\boldsymbol{\theta}$. It is useful to think of the Gaussian mixture model, in which the parameters include anything that is unknown:
  - e.g., if all the mixture weights, means, and covariance matrices are unknown, then $\boldsymbol{\theta} = \big(\{p_z\}_{z=1}^m, \{\boldsymbol{\mu}_z\}_{z=1}^m, \{\boldsymbol{\Sigma}_z\}_{z=1}^m\big)$, where $(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$ is the mean vector and covariance matrix of the z-th mixture component $p(\mathbf{x}|z)$.
  - e.g., if, on the other hand, we know in advance that $p_z = \frac{1}{m}$ for all z and $\boldsymbol{\Sigma}_z = \mathbf{I}$ for all z, then we would just have $\boldsymbol{\theta} = \{\boldsymbol{\mu}_z\}_{z=1}^m$.

- As usual, we consider a collection of n samples $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. We write the underlying hidden variables as $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_n)$, and assume that the samples are independent:

$$p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta}) = \prod_{t=1}^{n} p(\mathbf{x}_t, \mathbf{z}_t; \boldsymbol{\theta}).$$

By summing both sides over all $\mathbf{Z}$ sequences, we find that this implies $p(\mathbf{X};\boldsymbol{\theta}) = \prod_{t=1}^{n} p(\mathbf{x}_t;\boldsymbol{\theta})$.

- Fundamental question. Given the samples $\mathbf{X}$ (but not $\mathbf{Z}$), how do we perform maximum-likelihood estimation? (i.e., maximize $p(\mathbf{X};\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$)
  - Challenge 1: $p(\mathbf{X};\boldsymbol{\theta})$ is typically challenging to optimize directly. Observe that $p(\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}} p(\mathbf{x},\mathbf{z};\boldsymbol{\theta})$, so taking the log in $p(\mathbf{X};\boldsymbol{\theta}) = \prod_{t=1}^{n} p(\mathbf{x}_t;\boldsymbol{\theta})$ gives

$$\log p(\mathbf{X};\boldsymbol{\theta}) = \sum_{t=1}^{n} \log \sum_{\mathbf{z}_t} p(\mathbf{x}_t,\mathbf{z}_t;\boldsymbol{\theta}). \tag{6}$$

    The summation inside the log typically makes this computationally hard to maximize with respect to $\boldsymbol{\theta}$, especially if the length of $\mathbf{z}$ is large.
  - Challenge 2: Potentially large number of parameters, e.g., in the above example with $\boldsymbol{\theta} = \big(\{p_z\}_{z=1}^m, \{\boldsymbol{\mu}_z\}_{z=1}^m, \{\boldsymbol{\Sigma}_z\}_{z=1}^m\big)$, we have m scalars, m vectors, and m matrices to optimize.

4 Expectation Maximization (EM) Algorithm

The algorithm.

- As with K-means, it is possible to alternate between two steps that always improve things (i.e., the updated $\boldsymbol{\theta}$ has a higher value of $p(\mathbf{X};\boldsymbol{\theta})$). The analogy with K-means is as follows:
  - The vector $\mathbf{z}$ will be analogous to a cluster index, but instead of assigning a fixed $\mathbf{z}$ to each $\mathbf{x}$ (a "hard assignment"), we assign a distribution $p(\mathbf{z}|\mathbf{x})$ a (a "soft assignment").
  - The parameter vector $\boldsymbol{\theta}$ will be analogous to the collection of cluster centers, and in each iteration we will seek to update these to obtain improved parameters.
- More precisely, the steps are as follows:
  - (E-Step) Compute the conditional distribution $p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}^{(\ell)})$ of $\mathbf{z}$ given $\mathbf{x}$ corresponding to the current estimate $\boldsymbol{\theta}^{(\ell)}$ *(or more generally, calculate the suitable averages with respect to $p(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}^{(\ell)})$ that are needed in the M-Step)*
  - (M-Step) Update the current estimate by choosing $\boldsymbol{\theta}^{(\ell+1)}$ equaling the $\boldsymbol{\theta}$ that maximizes the average log-likelihood:

$$\begin{aligned}\boldsymbol{\theta}^{(\ell+1)} &= \arg\max_{\boldsymbol{\theta}} \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X};\boldsymbol{\theta}^{(\ell)}) \log p(\mathbf{X},\mathbf{Z};\boldsymbol{\theta}) \\ &= \arg\max_{\boldsymbol{\theta}} \sum_{t=1}^{n} \sum_{\mathbf{z}_t} p(\mathbf{z}_t|\mathbf{x}_t;\boldsymbol{\theta}^{(\ell)}) \log p(\mathbf{x}_t,\mathbf{z}_t;\boldsymbol{\theta}),\end{aligned} \tag{7}$$

    where the equality is due to independent data samples: $p(\mathbf{X},\mathbf{Z}) = \prod_{t=1}^{n} p(\mathbf{x}_t,\mathbf{z}_t)$.
- The intuition. (i) In the M-step, act like our belief on the distribution $(\mathbf{Z}|\mathbf{X})$ is correct, and choose $\boldsymbol{\theta}$ to maximize the log-likelihood $\log p(\mathbf{X},\mathbf{Z})$ "on average" accordingly. (ii) In the E-step, update our belief on $(\mathbf{Z}|\mathbf{X})$ due to the change in the estimate of $\boldsymbol{\theta}$.

– In simpler terms: If we knew $\mathbf{Z}$ we would want to maximize $\log p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta})$, but since we don't know it, we integrate over our current belief of it, namely $p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)})$.

– While it may not be immediately obvious, (7) is typically a much easier maximization problem to solve (exactly or approximately) compared to a direct maximization of (6).

Notes.

- In the form written above, EM is limited to situations in which averaging over $p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)})$ is computationally feasible, which is not always the case.
- Because of this, approximation methods (e.g., variational methods, Monte Carlo methods) are typically used. These are not covered in this course.
- Similarly, the optimization over $\boldsymbol{\theta}$ in (7) might only be done approximately (e.g., using gradient descent).

5 EM Guarantee via Majorization-Minimization

The maximization-minimization principle..

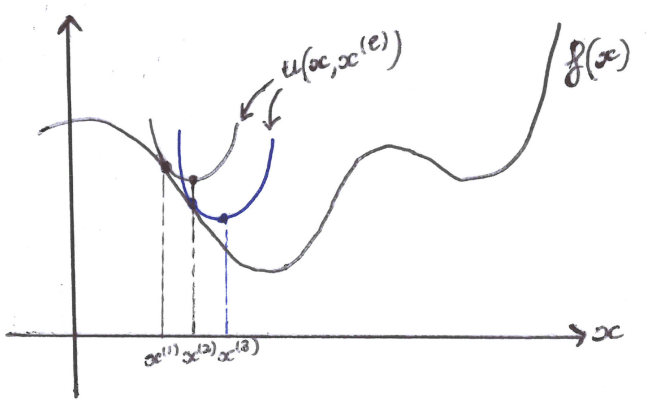
- Momentarily let's consider a generic optimization problem $\min_{\mathbf{x}} f(\mathbf{x})$ (where $\mathbf{x} \in \mathbb{R}^d$ here is distinct from the other sections of this document)
- When direct optimization is difficult, one approach is to successively minimize an *auxiliary function*:

$$\mathbf{x}^{(\ell+1)} = \arg\min_{\mathbf{x}} u(\mathbf{x}, \mathbf{x}^{(\ell)}).$$

- We will show that two simple properties suffice to ensure that $f(\mathbf{x}^{(\ell)})$ is non-increasing in ℓ:

  1. $u(\mathbf{x}, \mathbf{x}) = f(\mathbf{x})$ for all $\mathbf{x}$;
  2. $u(\mathbf{x}, \mathbf{x}') \geq f(\mathbf{x})$ for all $\mathbf{x}, \mathbf{x}'$.

  In this case, we say that u is a *majorizer* of f. An illustration:



- The proof of monotone improvement for f is very simple:

$$f(\mathbf{x}^{(\ell+1)}) \overset{(a)}{\leq} u(\mathbf{x}^{(\ell+1)}, \mathbf{x}^{(\ell)}) \overset{(b)}{\leq} u(\mathbf{x}^{(\ell)}, \mathbf{x}^{(\ell)}) = f(\mathbf{x}^{(\ell)}),$$

  where (a) uses property 2 above, (b) uses the definition of $\mathbf{x}^{(\ell+1)}$, and (c) uses property 1 above.

- Based on the above alone, it still remains plausible the improvement could get increasingly slow and then just stop even when further local improvements remain possible. However, under some technical conditions, one can show that $\mathbf{x}^{(\ell)}$ converges to a local minimum (see lec16a.pdf).

Application to EM.

- It is convenient to note that maximizing (7) is equivalent to minimizing

$$\begin{aligned} u(\boldsymbol{\theta}, \boldsymbol{\theta}^{(l)}) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)}) \log \frac{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)})}{p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta})} \\ &= -\sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)}) \log p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta}) + \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)}) \log p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}^{(\ell)}), \end{aligned}$$

  since the second term does not depend on the optimization variable $\boldsymbol{\theta}$.

- We claim that the required properties of u hold (with $\boldsymbol{\theta}$ now being the optimization variable and $f(\boldsymbol{\theta}) = -\log p(\mathbf{X}; \boldsymbol{\theta})$ being the function we want to minimize).

- To check the first property, we write

$$\begin{aligned} u(\boldsymbol{\theta}, \boldsymbol{\theta}) &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}) \log \frac{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta})}{p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta})} \\ &\overset{(a)}{=} \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}) \log \frac{1}{p(\mathbf{X}; \boldsymbol{\theta})} \\ &\overset{(b)}{=} -\log p(\mathbf{X}; \boldsymbol{\theta}) \\ &= f(\boldsymbol{\theta}), \end{aligned}$$

  where (a) uses $p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta}) = p(\mathbf{X}; \boldsymbol{\theta}) p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta})$, and (b) uses $\sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}) = 1$.

- For the second property, we write

$$\begin{aligned} u(\boldsymbol{\theta}, \boldsymbol{\theta}') &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}') \log \frac{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}')}{p(\mathbf{X}, \mathbf{Z}; \boldsymbol{\theta})} \\ &= \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}') \log \frac{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}')}{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}) p(\mathbf{X}; \boldsymbol{\theta})} \\ &= -\log p(\mathbf{X}; \boldsymbol{\theta}) + \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}') \log \frac{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta}')}{p(\mathbf{Z}|\mathbf{X}; \boldsymbol{\theta})}. \end{aligned}$$

  So we only need to show that the second term is non-negative.

  - The quantity $D(p_1 \| p_2) = \sum_z p_1(z) \log \frac{p_1(z)}{p_2(z)}$ is known as the *Kullback-Leibler (KL) divergence*, and is fact non-negative for *any* PMFs p_1 and p_2.

– A short proof is as follows:

$$\begin{aligned} -D(p_1\|p_2) &= \sum_z p_1(z)\log\frac{p_2(z)}{p_1(z)} \\ &\overset{(a)}{\leq} \sum_z p_1(z)\left(\frac{p_2(z)}{p_1(z)}-1\right) \\ &= \sum_z p_2(z) - \sum_z p_1(z) \\ &= 0, \end{aligned}$$

where (a) uses the inequality $\log\alpha \leq \alpha - 1$, which is easily verified graphically.

– Note: This property and proof extend immediately to the case that p_1 and p_2 are PDFs (on $\mathbb{R}^d$, say) instead of PMFs, in which case $D(p_1\|p_2) = \int_{\mathbb{R}^d} p_1(z)\log\frac{p_1(z)}{p_2(z)}dz$.

- In summary, we have proved that the likelihood $p(\mathbf{X};\boldsymbol{\theta}^{(\ell)})$ increases in the iteration number ℓ under the EM algorithm.

  – As with K-means, although we approach a *local maximum*, we are certainly not guaranteed to get close to the *global maximum*.

6 Example: EM Applied to Gaussian Mixture Models

- Recall that $\boldsymbol{\theta} = \left(\{p_z\}_{z=1}^m, \{\boldsymbol{\mu}_z\}_{z=1}^m, \{\boldsymbol{\Sigma}_z\}_{z=1}^m\right)$, and let $\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)$ be the multivariate Gaussian PDF with mean vector $\boldsymbol{\mu}_j$ and covariance matrix $\boldsymbol{\Sigma}_j$.

- Since the data samples are independent, the log-likelihood function *including hidden variables* is

$$\begin{aligned} \log p(\mathbf{X},\mathbf{Z};\boldsymbol{\theta}) &= \sum_{t=1}^n \log p(\mathbf{x}_t, z_t;\boldsymbol{\theta}) \\ &= \sum_{t=1}^n \log\left(p_{z_t}\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t},\boldsymbol{\Sigma}_{z_t})\right) \\ &= \sum_{t=1}^n \log p_{z_t} + \sum_{t=1}^n \log\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t},\boldsymbol{\Sigma}_{z_t}). \end{aligned} \tag{8}$$

- The distribution $p(z_t|\mathbf{x}_t;\boldsymbol{\theta}^{(\ell)})$ for a single sample (indexed by t), denoted by $p_t^{(\ell)}(z_t)$, is given by

$$\begin{aligned} p_t^{(\ell)}(z_t) &= p(z_t|\mathbf{x}_t;\boldsymbol{\theta}^{(\ell)}) && (9) \\ &= \frac{p(\mathbf{x}_t, z_t;\boldsymbol{\theta}^{(\ell)})}{p(\mathbf{x}_t;\boldsymbol{\theta}^{(\ell)})} \\ &= \frac{p_{z_t}^{(\ell)}\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t}^{(\ell)},\boldsymbol{\Sigma}_{z_t}^{(\ell)})}{\sum_z p_z^{(\ell)}\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_z^{(\ell)},\boldsymbol{\Sigma}_z^{(\ell)})}, && (10) \end{aligned}$$

and the full posterior is $p(\mathbf{Z}|\mathbf{X};\boldsymbol{\theta}^{(\ell)}) = \prod_{t=1}^n p(z_t|\mathbf{x}_t;\boldsymbol{\theta}^{(\ell)})$.

- Substituting (8) and (10) into (7) gives

$$\begin{aligned}
&\sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X};\boldsymbol{\theta}^{(\ell)}) \log p(\mathbf{X},\mathbf{Z};\boldsymbol{\theta}) \\
&\quad = \mathbb{E}_{\mathbf{Z}\sim p(\cdot|\mathbf{X},\boldsymbol{\theta}^{(\ell)})}\left[\sum_{t=1}^{n} \log p_{z_t} + \sum_{t=1}^{n} \log \mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t},\boldsymbol{\Sigma}_{z_t})\right] \\
&\quad = \sum_{t=1}^{n}\sum_{z_t} \frac{p_{z_t}^{(\ell)}\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t}^{(\ell)},\boldsymbol{\Sigma}_{z_t}^{(\ell)})}{\sum_z p_z \mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_z^{(\ell)},\boldsymbol{\Sigma}_z^{(\ell)})} \log p_{z_t} + \sum_{t=1}^{n}\sum_{z_t} \frac{p_{z_t}^{(\ell)}\mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t}^{(\ell)},\boldsymbol{\Sigma}_{z_t}^{(\ell)})}{\sum_z p_z \mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_z^{(\ell)},\boldsymbol{\Sigma}_z^{(\ell)})} \log \mathcal{N}(\mathbf{x}_t;\boldsymbol{\mu}_{z_t},\boldsymbol{\Sigma}_{z_t}).
\end{aligned}$$

  Solving the M-step according to (7) amounts to maximizing this expression with respect to $\{p_z\}_{z=1}^m$, $\{\boldsymbol{\mu}_z\}_{z=1}^m$, and $\{\boldsymbol{\Sigma}_z\}_{z=1}^m$. The details of this are tedious, so we will not go through them, but the solution is intuitive: Recalling the definition $p_t^{(\ell)}(z_t)$ in (10), the maximizing parameters are

$$p_z^* = \frac{\hat{n}(z)}{n}, \quad \text{where } \hat{n}(z) = \sum_{t=1}^{n} p_t^{(\ell)}(z) \tag{11}$$

$$\hat{\boldsymbol{\mu}}_z^* = \frac{1}{\hat{n}(z)} \sum_{t=1}^{n} p_t^{(\ell)}(z)\mathbf{x}_t \tag{12}$$

$$\hat{\boldsymbol{\Sigma}}_z^* = \frac{1}{\hat{n}(z)} \sum_{t=1}^{n} p_t^{(\ell)}(z)(\mathbf{x}_t - \hat{\boldsymbol{\mu}}_z^*)(\mathbf{x}_t - \hat{\boldsymbol{\mu}}_z^*)^T. \tag{13}$$

  In words, p_z^* is just the total expected number of z according to $p_t^{(\ell)}(z_t) = p(z_t|\mathbf{x}_t;\boldsymbol{\theta}^{(\ell)})$, and $\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z$ are just a "weighted empirical" mean vector and covariance matrix.

- If the covariance matrices were not optimized but instead fixed to be the identity matrix, the updates would look a lot like K-means:
  - The weighted posterior (10) acts as a "soft" assignment to clusters;
  - The updated mean parameter $\hat{\boldsymbol{\mu}}_z^*$ acts a weighted counterpart to (2).