

Do not edit
How to change the
design



**Join at slido.com
#9624861**

① The Slido app must be installed on every computer you're presenting from

slido

Lecture 5

Density Estimation and Gaussian Mixture Models

Introducing Maximum Likelihood Estimation, and
Gaussian Mixture Models

EECS 189/289, Fall 2026 @ UC Berkeley

Joseph E. Gonzalez and Narges Norouzi

Recap



9624861

In the last lecture we introduced basic probability rules and derived the Maximum Likelihood Estimate (MLE) and started to derive the Maximum a Posteriori (MAP) Estimate.

Today we will:

- Finish deriving the MAP estimate
- Introduce the **Multivariate Gaussian** and how to **compute Bias**
- Introduce the **Mixture of Gaussians model** that combines many of these ideas.



9624861

Estimating the Parameters of a Distribution

Given a collection of data $\mathcal{D} = \{x_1, \dots, x_N\}$, there are several ways to estimate the “best” parameters for a given distribution.

- **Maximum Likelihood Estimation (MLE):** choose the parameters that maximize the **likelihood** of the data. (Frequentist)

$$w_{\text{MLE}}^* = \arg \max_w p(\mathcal{D} | w)$$

- **Maximum A Posteriori (MAP) Estimation:** choose the parameters that maximize the posterior given the data and the **prior**. (Bayesian)

$$w_{\text{MAP}}^* = \arg \max_w p(\mathcal{D} | w) p(w)$$

Both methods rely on computing the **likelihood of the data**.



9624861

The Bernoulli MLE Is Just Counting

The maximum likelihood estimate (MLE) is:

$$\mu_{ML} = \frac{n_1}{n_0 + n_1} = \frac{n_1}{N}$$

This is **the fraction of ones in the data**, which is the **empirical distribution** defined earlier in lecture.

➤ μ_{ML} is the parameter value that makes the **observed data most probable** under a Bernoulli model.



9624861

The Parameter as a Random Variable

Maximum likelihood estimation treats μ as an **unknown constant**.

The Bayesian view treats μ as a **random variable** with a distribution. Applying Bayes' theorem to the parameter:

$$p(\mu | \mathcal{D}) = \frac{p(\mathcal{D} | \mu) p(\mu)}{p(\mathcal{D})}$$

- **Prior** $p(\mu)$: what we believe about μ before seeing data.
- **Likelihood** $p(\mathcal{D} | \mu)$: is the quantity we maximized for the MLE.
- **Posterior** $p(\mu | \mathcal{D})$: the updated belief after seeing the data.

Maximum a Posteriori (MAP) estimate maximizes the posterior:

$$\mu_{\text{MAP}} = \arg \max_{\mu} \ln p(\mathcal{D} | \mu) + \ln p(\mu)$$

- The denominator $p(\mathcal{D})$ does not depend on μ , so it does not affect the maximization.



9624861

Modeling the Prior

The **choice** and **parameterization** of model for the prior $p(\mu)$ is a **design decision**.

- Capture our prior beliefs about the μ .
- Model is often chosen for convenience (conjugate to the likelihood).
- The prior parameters are set using *prior knowledge* or *another prior*.

What would be a good prior $p(\mu)$ for the Bernoulli coin flip example?

- The parameter $0 \leq \mu \leq 1$ is the probability of landing heads
- We expect the coin (and the flipper) to be approximately fair.



9624861

The Beta Distribution

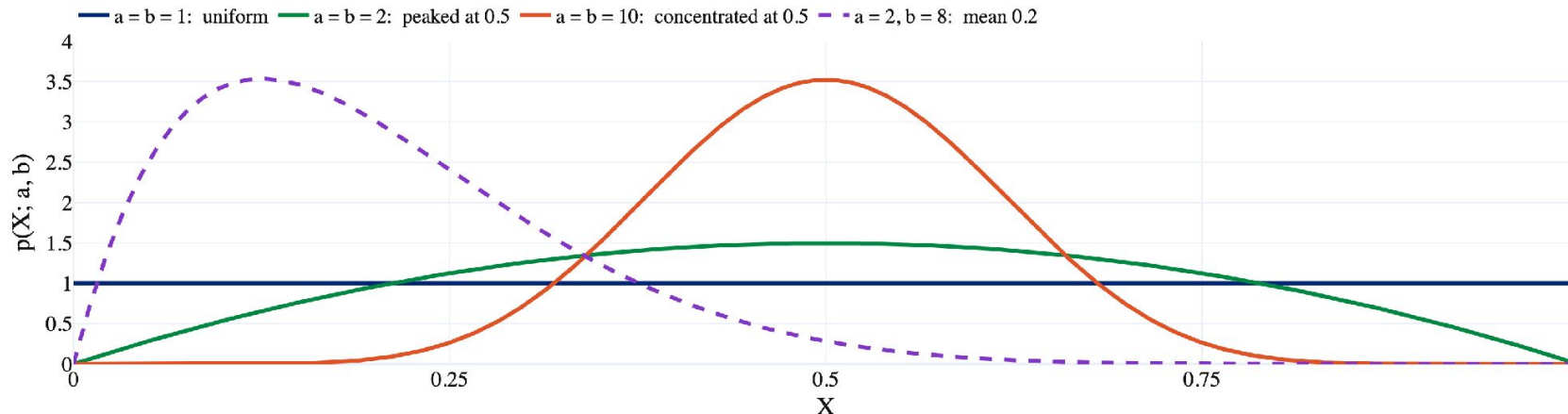
The **Beta distribution** for the **continuous random** variable $0 \leq X \leq 1$:

$$p(X; a, b) = \frac{1}{B(a, b)} X^{a-1} (1 - X)^{b-1}$$

with **parameters by α and β** with the following shape:

- $a = b = 1$: every value of X is equally plausible.
- $a = b = 2$: values of X close to 0.5 are more likely
- Large a and b concentrate the distribution near $\frac{a}{a+b}$

The beta function $B(a, b)$ is the normalizer (computed numerically) and outside the scope of this class.





9624861

Deriving the Posterior for Bernoulli + Beta

Recall we are trying to compute:

$$p(\mu | \mathcal{D}) = \frac{p(\mathcal{D} | \mu) p(\mu)}{p(\mathcal{D})}$$

Substituting the Bernoulli and the Beta distributions (without normalization):

$$p(\mu | \mathcal{D}) \propto p(\mathcal{D} | \mu) p(\mu) = \mu^{n_1} (1 - \mu)^{n_0} \mu^{(a-1)} (1 - \mu)^{(b-1)}$$

Collecting terms, we can see that the **posterior is also a Beta distribution**:

$$p(\mu | \mathcal{D}) \propto \mu^{(n_1+a-1)} (1 - \mu)^{(n_0+b-1)}$$

with parameters $a' = (n_1 + a)$ and $b' = (n_0 + b)$.

When a prior paired with a likelihood has the same form as the posterior it is called a **conjugate prior**.

➤ The Beta distribution is the conjugate prior for the Bernoulli likelihood.



9624861

Computing the MAP

Maximum a Posteriori (MAP) estimate maximizes the posterior:

$$\mu_{\text{MAP}} = \arg \max_{\mu} \ln p(\mathcal{D} | \mu) + \ln p(\mu)$$

When deriving the posterior we already showed (collecting terms):

$$\ln p(\mathcal{D} | \mu) + \ln p(\mu) = (n_1 + a - 1) \ln \mu + (n_0 + b - 1) \ln(1 - \mu)$$

When computing the MLE we derived the **stationary point** of:

$$\frac{n_1}{n_1 + n_0} = \arg \max_{\mu} n_1 \ln \mu + n_0 \ln(1 - \mu)$$

Substituting $n'_1 = (n_1 + a - 1)$ and $n'_0 = (n_0 + b - 1)$.

$$\mu_{\text{MAP}} = \frac{n_1 + a - 1}{n_1 + n_0 + a + b - 2} = \frac{n_1 + a - 1}{N + a + b - 2}$$



9624861

The Prior as Pseudo-Counts

The prior acts like a form of pseudo-counts (assumed binary observations) by contributing $a - 1$ imaginary successes $b - 1$ imaginary failures.

$$\mu_{\text{MAP}} = \frac{n_1 + a - 1}{n_1 + n_0 + a + b - 2} = \frac{n_1 + a - 1}{N + a + b - 2}$$

With $a = b = 2$ this becomes $\mu_{\text{MAP}} = \frac{n_1 + 1}{N + 2}$ one imaginary observation of each outcome.

➤ Regardless of the observations $0 < \mu_{\text{MAP}} = \frac{n_1 + 1}{N + 2} < 1$ the probability is never 0 or 1.

This is also called **Laplace Smoothing**

➤ Add **pseudo counts to empirical estimates** to act as a prior when there is **limited data** or **events are rare**.



9624861

How μ_{MAP} and μ_{ML} scale with N

What happens as N grows and how do μ_{MAP} and μ_{ML} compare?

$$\mu_{\text{ML}} = \frac{n_1}{n_0 + n_1} = \frac{n_1}{N}, \quad \mu_{\text{MAP}} = \frac{n_1 + a - 1}{n_1 + n_0 + a + b - 2} = \frac{n_1 + a - 1}{N + a + b - 2}$$

As N grows, the constants $a - 1$ and $a + b - 2$ become negligible and the **two estimates converge**.

➤ **Small N:** the prior dominates, and the estimate is pulled toward the prior mean.

➤ **Large N:** the data dominates, and MAP and MLE agree.

Today many deep learning applications operate in the Large N (Big Data) setting where the MLE method is preferred (simpler).

➤ In future lectures, we will discuss introduce **regularization methods** that are equivalent or related to MAP estimation with priors.



For $a, b > 2$ which has higher variance?

More Practice with MLE

Starting Lecture 5



9624861

Normal (Gaussian) Distribution

If a single real-valued variable X is **normally distributed**, then:

$$X \sim \mathcal{N}(\mu, \sigma^2) \quad \Rightarrow \quad p(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x - \mu)^2\right)$$

Where the parameters μ and σ^2 are called the mean and variance.

With some calculus you can show $\mathbb{E}[X] = \mu$ and $\text{var}[X] = \sigma^2$.



9624861

The MLE for IID Gaussian Samples (Part 1)

Example: Gaussian Distribution

We observe N continuous real valued samples $\mathcal{D} = \{x_1, \dots, x_N\}$

1. Modeling: Let's assume IID Gaussian trials:

$$\begin{aligned}\ln p(\mathcal{D}|\mu, \sigma^2) &= \sum_{n=1}^N \ln \left(\frac{1}{\sqrt{2\pi}\sigma} \exp \left(-\frac{1}{2\sigma^2} (x_n - \mu)^2 \right) \right) \\ &= -\frac{N}{2} \ln(2\pi) - \frac{N}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^N (x_n - \mu)^2\end{aligned}$$

2. Optimization: Maximize the log likelihood function:

$$\arg \max_{\mu, \sigma^2} -\frac{N}{2} \ln(2\pi) - \frac{N}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^N (x_n - \mu)^2$$

The MLE for IID Gaussian Samples (Part 2)



9624861

♥ Maximize the log likelihood function:

$$\arg \max_{\mu, \sigma^2} -\frac{N}{2} \ln(2\pi) - \frac{N}{2} \ln(\sigma^2) - \frac{1}{2\sigma^2} \sum_{n=1}^N (x_n - \mu)^2$$

Compute derivative with respect to μ and σ^2 and solve for the critical-points.

$$\frac{\partial}{\partial \mu} \ln p(\mathcal{D} | \mu, \sigma^2) = \frac{1}{\sigma^2} \sum_{n=1}^N (x_n - \mu)$$

Set derivative equal to 0 and solve for μ_{ML} :

$$\frac{1}{\sigma^2} \sum_{n=1}^N (x_n - \mu) = 0 \implies \mu_{\text{ML}} = \frac{1}{N} \sum_{n=1}^N x_n$$



9624861

The MLE for IID Gaussian Samples (Part 3)

Computing the derivative with respect to the variance parameter $v = \sigma^2$

$$\frac{\partial}{\partial v} \ln p(\mathcal{D}|\mu, v) = \frac{\partial}{\partial v} \left(-\frac{N}{2} \ln(v) - \frac{1}{2v} \sum_{n=1}^N (x_n - \mu)^2 \right) = -\frac{N}{2} \left(\frac{1}{v} \right) + \frac{1}{2v^2} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2$$

Set equal to zero and solving for $v_{\text{ML}} = \sigma_{\text{ML}}^2$:

$$\begin{aligned} -\frac{N}{2} \left(\frac{1}{v} \right) + \frac{1}{2v^2} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 &= 0 \\ \frac{v^2}{N} * N \left(\frac{1}{v} \right) &= \frac{1}{v^2} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 * \frac{v^2}{N} \\ v_{\text{ML}} = \sigma_{\text{ML}}^2 &= \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 \end{aligned}$$

Set equal to zero and solving for $v_{\text{ML}} = \sigma_{\text{ML}}^2$:

$$-\frac{N}{2} \left(\frac{1}{v} \right) + \frac{1}{2v^2} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 = 0$$

$$\frac{v^2}{N} * N \left(\frac{1}{v} \right) = \frac{1}{v^2} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 * \frac{v^2}{N}$$

$$v_{\text{ML}} = \sigma_{\text{ML}}^2 = \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2$$

$$\sigma_{\text{ML}}^2 = \frac{1}{N} \sum_{n=1}^N (x_n^2 - 2 x_n \mu_{\text{ML}} + \mu_{\text{ML}}^2) = \left(\frac{1}{N} \sum_{n=1}^N x_n^2 \right) - \left(2 \mu_{\text{ML}} \frac{1}{N} \sum_{n=1}^N x_n \right) + \left(\frac{N \mu_{\text{ML}}^2}{N} \right)$$

$$= \left(\frac{1}{N} \sum_{n=1}^N x_n^2 \right) - (2\mu_{\text{ML}}^2) + (\mu_{\text{ML}}^2) = \left(\frac{1}{N} \sum_{n=1}^N x_n^2 \right) - \mu_{\text{ML}}^2$$



9624861



9624861

Recap of the MLE for IID Gaussian Samples

We used the method of **Maximum Likelihood** to estimate the **mean** and **variance parameters** of a Gaussian from data:

$$\mu_{\text{ML}} = \frac{1}{N} \sum_{n=1}^N x_n$$

Sufficient Statistics

$$\sigma_{\text{ML}}^2 = \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 = \frac{1}{N} \left(\sum_{n=1}^N x_n^2 \right) - \mu_{\text{ML}}^2$$

What are the sufficient statistics?

Do they match our intuition for the mean and variance?

Are they good estimates?



9624861

Bias

- MAP Estimation
- Gaussians
 - Maximum Likelihood
 - Bias
- Gaussian Mixture Models

Questions



9624861

Bias of an Estimate

How close is the true (unknown) parameter value to our estimate in expectation?

Before we observe our data, the value of our estimate w_{ML} is a **random variable**.

We can compute the bias of the estimation procedure as:

$$\text{Bias}(w_{ML}) = \mathbb{E}[w_{ML}] - w$$

Relative to the true (but unknown) w .



9624861

Bias of μ_{ML} for the Gaussian Distribution

Assuming the data are drawn IID from $\mathcal{N}(x|\mu, \sigma^2)$ treating X_n, μ_{ML} ,

$$\mathbb{E}[\mu_{\text{ML}}] = \mathbb{E}\left[\frac{1}{N} \sum_{n=1}^N X_n\right] = \frac{1}{N} \sum_{n=1}^N \mathbb{E}[X_n] = \frac{1}{N} \sum_{n=1}^N \mu = \boxed{\mu} \text{ Unbiased}$$

This means that in expectation over randomly sampled datasets our **Maximum Likelihood estimate of the mean** (μ_{ML}) will match the mean μ of the distribution $\mathcal{N}(x|\mu, \sigma^2)$ from which we sampled.



Is the maximum likelihood variance estimate biased?



9624861

Bias of σ_{ML}^2 for the Gaussian Distribution

Assuming the data are drawn IID from $\mathcal{N}(x|\mu, \sigma^2)$ treating X_n , μ_{ML} , and σ_{ML}^2 as random variables

$$\begin{aligned}\mathbb{E}[\sigma_{\text{ML}}^2] &= \mathbb{E}\left[\left(\frac{1}{N} \sum_{n=1}^N X_n^2\right) - \mu_{\text{ML}}^2\right] = \frac{1}{N} \sum_{n=1}^N \mathbb{E}[X_n^2] - \mathbb{E}[\mu_{\text{ML}}^2] \\ &= \frac{1}{N} \sum_{n=1}^N (\sigma^2 + \mu^2) - (\text{var}(\mu_{\text{ML}}) + \mathbb{E}[\mu_{\text{ML}}]^2)\end{aligned}$$

Variance Identity

$$\begin{aligned}\text{var}(X_n) &= \mathbb{E}[X_n^2] - \mu^2 = \sigma^2 \\ \text{var}(\mu_{\text{ML}}) &= \mathbb{E}[\mu_{\text{ML}}^2] - \mathbb{E}[\mu_{\text{ML}}]^2\end{aligned}$$

$$\text{var}(\mu_{\text{ML}}) = \text{var}\left(\frac{1}{N} \sum_{n=1}^N X_n\right) = \frac{1}{N^2} \underbrace{\left(\sum_{n=1}^N \text{var}(X_n)\right)}_{\text{Independence of } X_n} = \frac{1}{N^2} \underbrace{N \sigma^2}_{\text{var}(X_n)} = \frac{\sigma^2}{N}$$



9624861

Bias of σ_{ML}^2 for the Gaussian Distribution

Assuming the data are drawn IID from $\mathcal{N}(x|\mu, \sigma^2)$ treating X_n , μ_{ML} , and σ_{ML}^2 as random variables

$$\mathbb{E}[\sigma_{\text{ML}}^2] = \mathbb{E}\left[\left(\frac{1}{N} \sum_{n=1}^N X_n^2\right) - \mu_{\text{ML}}^2\right] = \frac{1}{N} \sum_{n=1}^N \mathbb{E}[X_n^2] - \mathbb{E}[\mu_{\text{ML}}^2]$$

Variance Identity

$$\begin{aligned}\text{var}(X_n) &= \mathbb{E}[X_n^2] - \mu^2 = \sigma^2 \\ \text{var}(\mu_{\text{ML}}) &= \mathbb{E}[\mu_{\text{ML}}^2] - \mathbb{E}[\mu_{\text{ML}}]^2\end{aligned}$$

Sample Mean Variance

$$\text{var}(\mu_{\text{ML}}) = \frac{\sigma^2}{N}$$

$$= \frac{1}{N} \sum_{n=1}^N (\sigma^2 + \mu^2) - (\text{var}(\mu_{\text{ML}}) + \mathbb{E}[\mu_{\text{ML}}]^2)$$

$$= (\sigma^2 + \mu^2) - \left(\frac{\sigma^2}{N} + \mu^2\right) = \sigma^2 \left(\frac{N-1}{N}\right)$$



9624861

Bias of σ_{ML}^2 for the Gaussian Distribution

• We have just shown that σ_{ML}^2 is **biased** by a factor of $\left(\frac{N-1}{N}\right)$.

$$\mathbb{E}[\sigma_{\text{ML}}^2] = \sigma^2 \left(\frac{N-1}{N} \right)$$

How can we make σ_{ML}^2 unbiased?



9624861

Correcting the Bias of σ_{ML}^2

We showed that σ_{ML}^2 is biased ($\mathbb{E}[\sigma_{\text{ML}}^2] \neq \sigma^2$):

$$\mathbb{E}[\sigma_{\text{ML}}^2] = \sigma^2 \left(\frac{N-1}{N} \right)$$

We can correct the bias by multiplying by $\left(\frac{N}{N-1} \right)$:

Sometimes called the
**unbiased
sample variance.**

$$\left(\frac{N}{N-1} \right) \sigma_{\text{ML}}^2 = \left(\frac{N}{N-1} \right) \frac{1}{N} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2 = \frac{1}{N-1} \sum_{n=1}^N (x_n - \mu_{\text{ML}})^2$$

For large N the bias vanishes $\rightarrow \sigma_{\text{ML}}^2$ is **asymptotically unbiased**.

➤ In future lectures, we will show that there is a tradeoff between bias and variance and we may prefer the biased estimator.

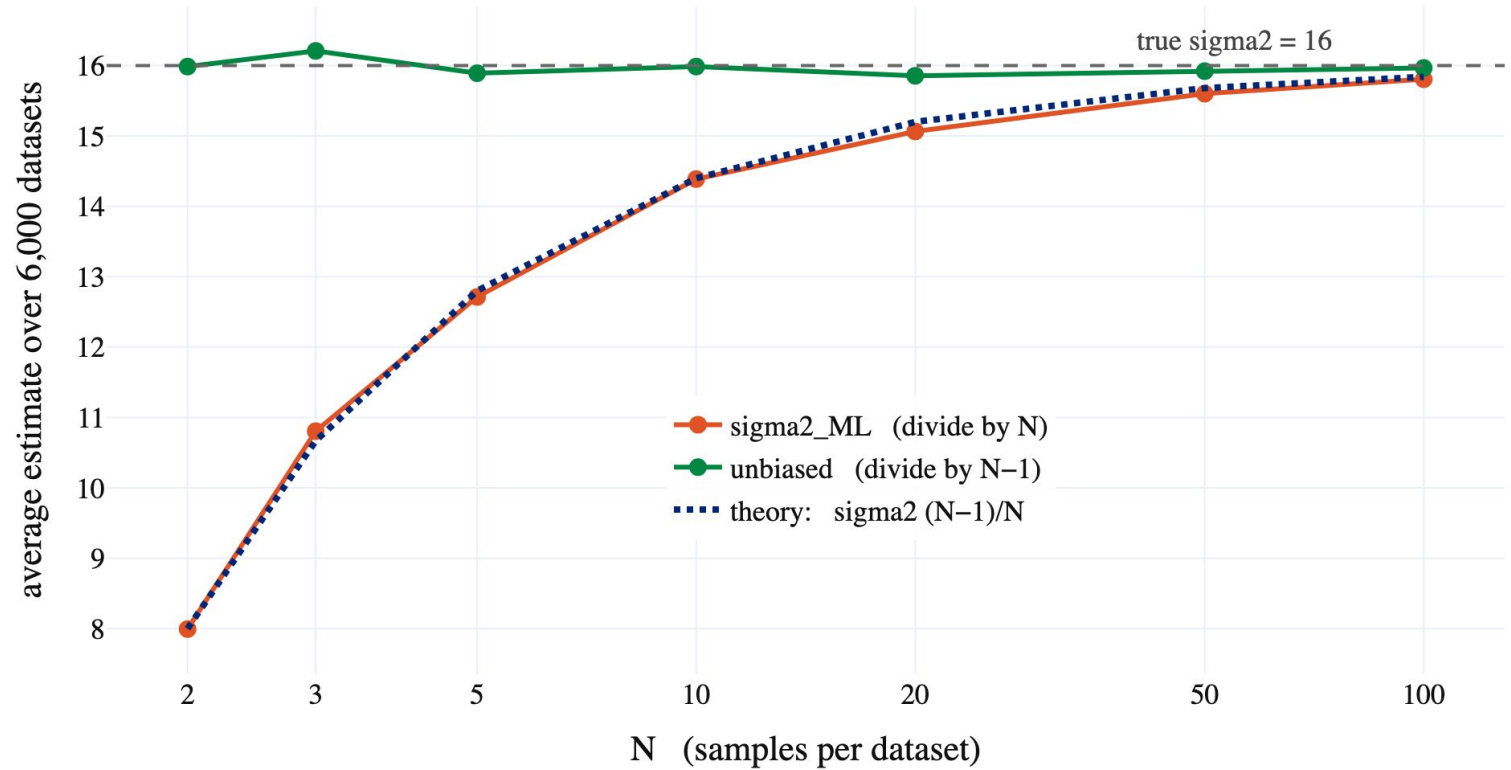


9624861

Demo

Why np.var has
a ddof argument

The MLE variance is too small by exactly $(N-1)/N$

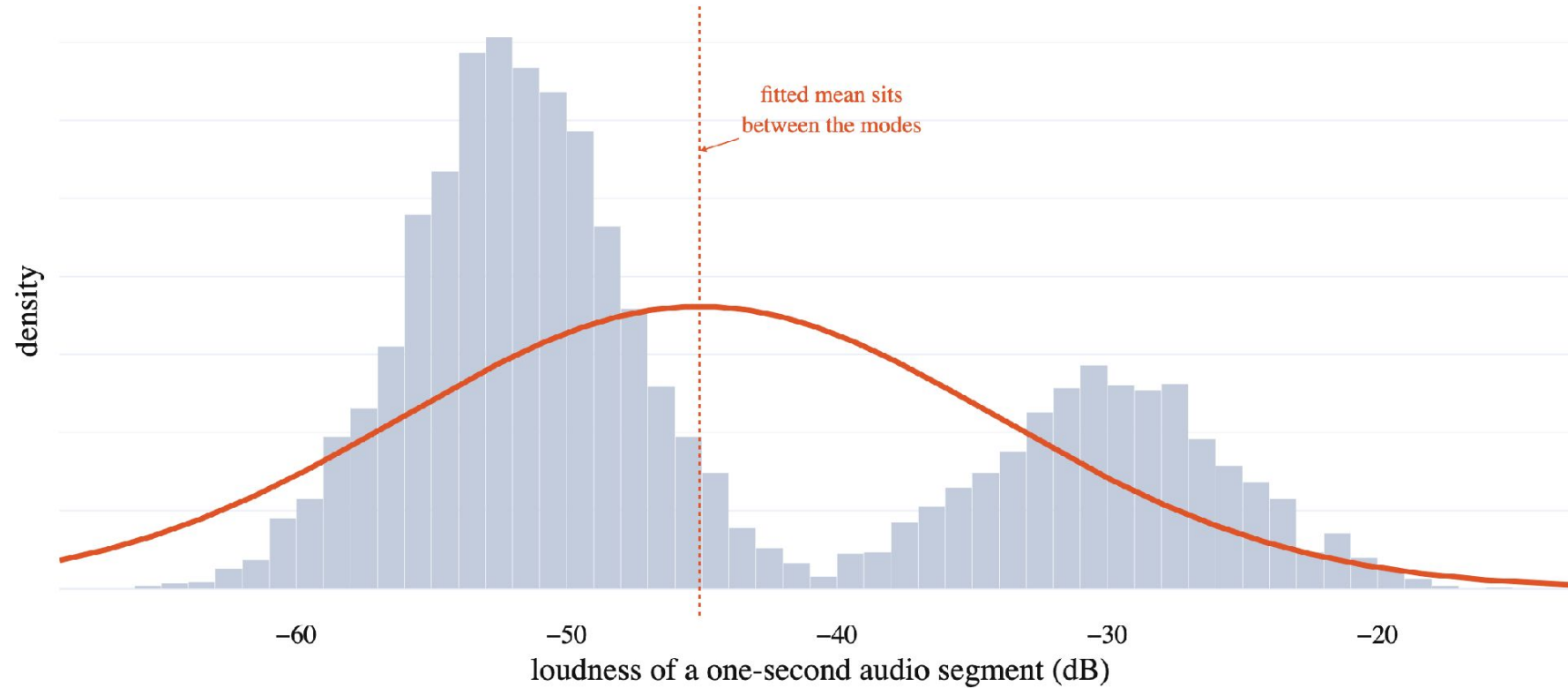


Fitting Data with Multiple Modes



9624861

Often data has multiple modes, and a single Gaussian is not a great model for the distribution.





9624861

Gaussian Mixture Models

- MAP Estimation
- Gaussians
 - Maximum Likelihood
 - Bias
- Gaussian Mixture Models

Questions

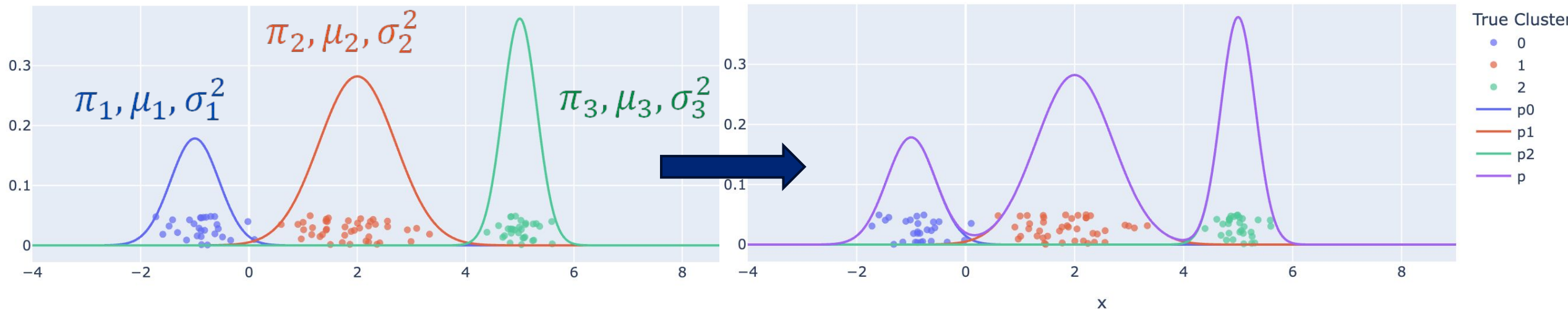
Gaussian Mixture Model (GMM)



9624861

The Gaussian mixture model defines the probability of sampling data from a **weighted combination** of Gaussians

- **Example: $K=3$**



- Note the data doesn't have a color or a y-value: both were added for visualization purposes.



9624861

Gaussian Mixture Model (GMM)

The Gaussian mixture model defines the probability of sampling data from a **weighted combination** of Gaussians:

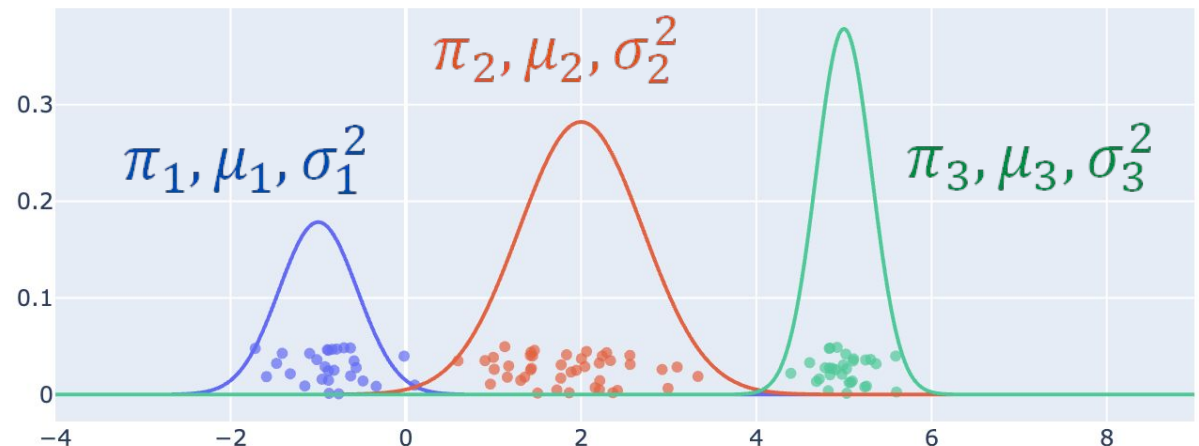
$$p(x | \pi, \mu_1, \sigma_1^2, \dots, \mu_K, \sigma_K^2) = \sum_{k=1}^K \pi_k \mathcal{N}(x | \mu_k, \sigma_k^2)$$

where $\sum_{k=1}^K \pi_k = 1$ is a tabular probability distribution over clusters.

- **Example:** $K=3$

Parameters

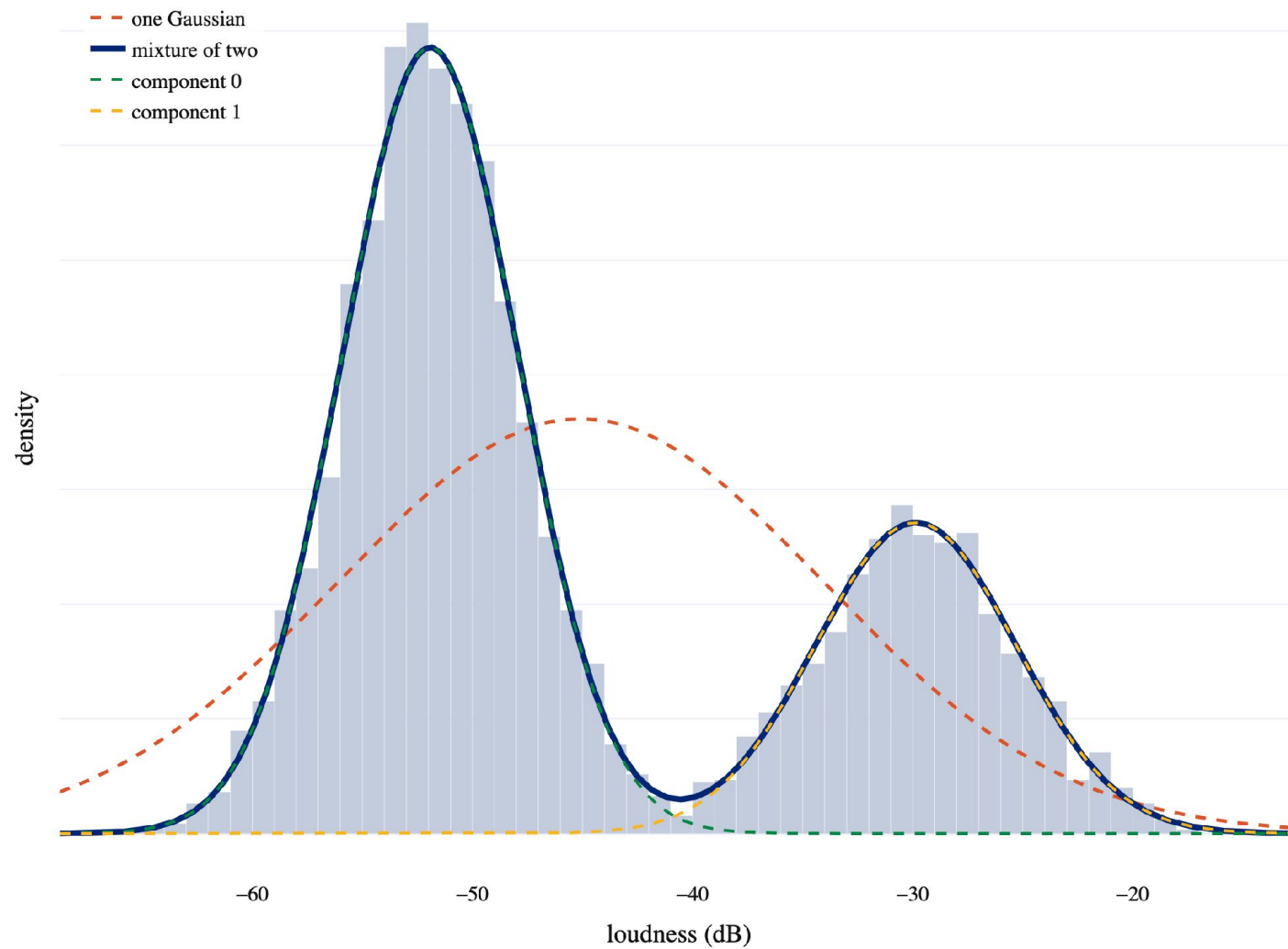
```
pi = np.array([0.2, 0.5, 0.3])  
mu = np.array([-1, 2, 5])  
sigma = np.array([0.2, 0.5, .1])
```



Demo

Gaussian Mixture Model

A mixture of two Gaussians fits what one cannot





9624861

The GMM is a Latent Variable Model

We introduce a **latent** (unobserved) **assignment variable** z_n for each x_n .

$$p(z_n) = \pi_{z_n}$$

Think of this as the color in the earlier visualization

We then define the likelihood of x_n given z_n as:

Density for is based on it's Gaussian.

$$p(x_n | z_n) = \mathcal{N}(x_n | \mu_{z_n}, \sigma_{z_n}^2)$$

The joint probability of the x_n and z_n is then:

$$p(x_n, z_n) = p(x_n | z_n) p(z_n) = \mathcal{N}(x_n | \mu_{z_n}, \sigma_{z_n}^2) \pi_{z_n}$$

If we **marginalize** over the unknown z_n we obtain the **GMM**:

$$p(x_n) = \sum_{k=1}^K p(x_n, z_n = k) = \sum_{k=1}^K \mathcal{N}(x_n | \mu_k, \sigma_k^2) \pi_k$$

1.3	1
9.6	3
2.4	1
5.7	2
...	...



9624861

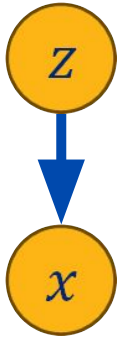
The GMM is a Generative Model

A **generative model** is a model that can **generate the data**.

$$p(x_n, z_n) = p(z_n) p(x_n | z_n)$$

We can construct **graph** where the parents of a variable are the variables in the **conditional distribution**.

- This is a **graphical model** and is used to represent the **conditional structure** of a probability model.



We can sample data from a graphical model using **ancestor sampling**.

- Sample all the variables that have no parents: $p(z_n)$
- Sample variables after all their parents have been sampled: $p(x_n | z_n)$

Example

Sample z_n from **Discrete**($\pi = [.1, .3, .6]$)
 $\rightarrow z_n = 2$

Sample x_n from $\mathcal{N}(x_n | \mu_2, \sigma_2^2)$
 $\rightarrow x_n = 4.8$



9624861

Demo

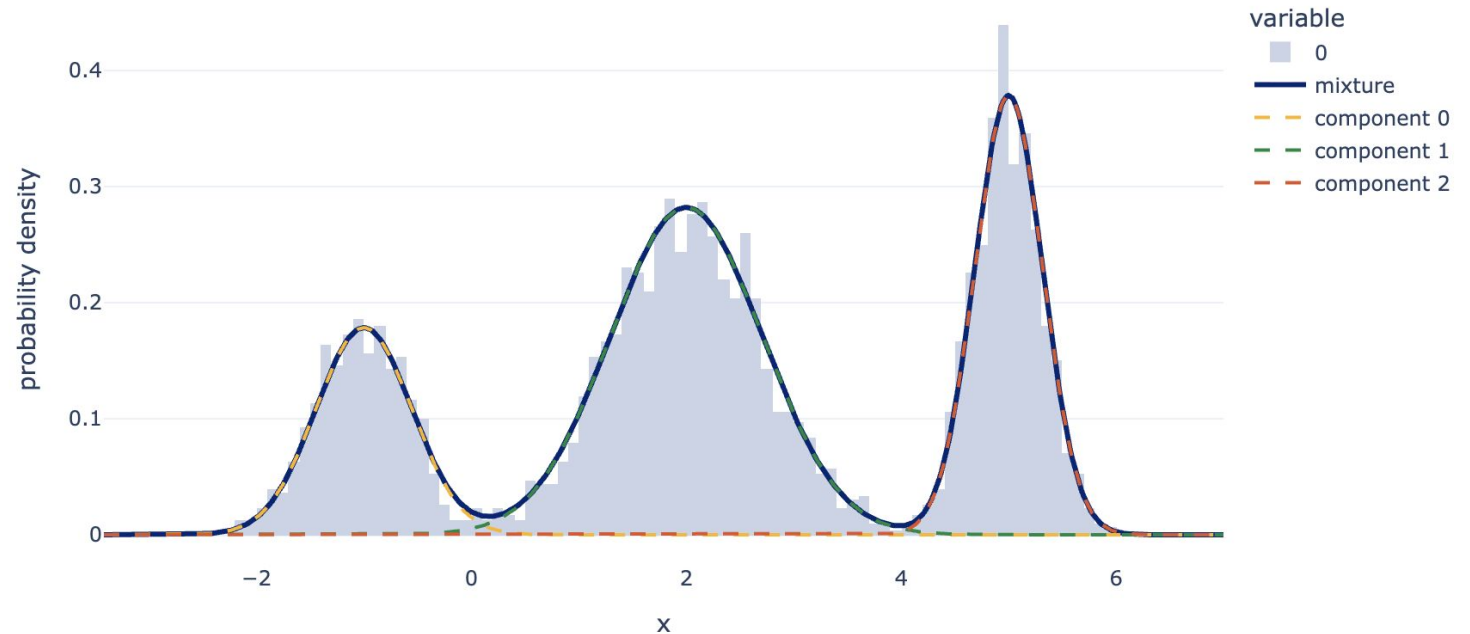
Sampling from a GMM

```
pi_true = np.array([0.2, 0.5, 0.3])
mu_true = np.array([-1.0, 2.0, 5.0])
var_true = np.array([0.2, 0.5, 0.1])

N = 3000
z = rng.choice(3, size=N, p=pi_true) # first the latent component
x = rng.normal(mu_true[z], np.sqrt(var_true[z])) # then the observation

pd.Series(z).value_counts(normalize=True).sort_index().round(3)
```

Samples drawn by ancestor sampling, with the density that generated them





9624861

The Connection to K-Means

In Lecture 2, k-means assigned points to clusters using the same symbol $z_n \in \{1, \dots, K\}$ as today.

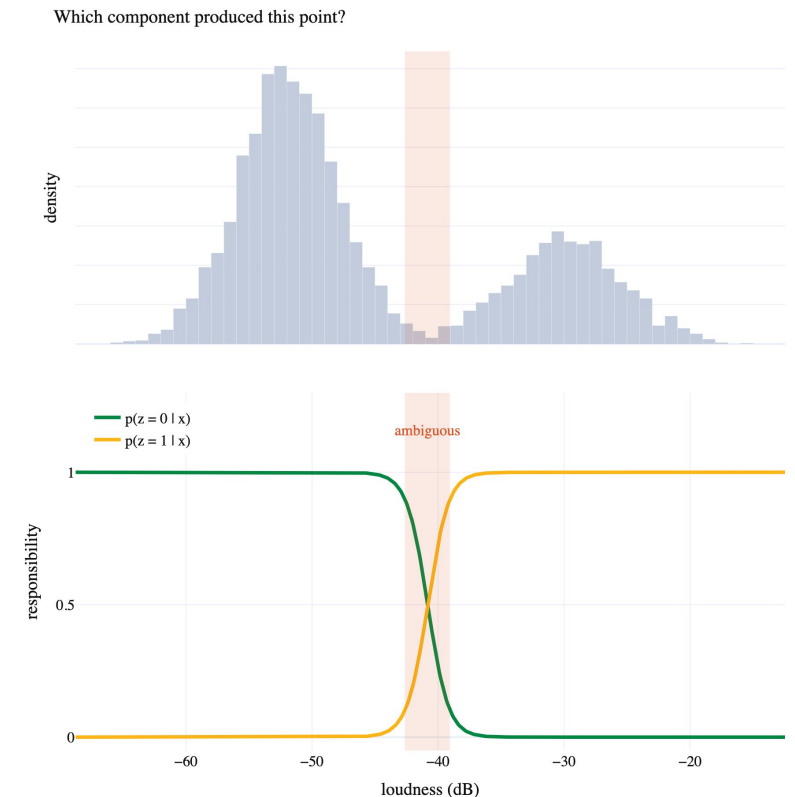
➤ k-means gives a **hard assignment**. Each point belongs to one cluster.

For a point midway between two clusters, that is a poor description.

A Gaussian mixture model keeps the same latent variable z_n but treats it as **random**.

➤ Instead of a single assignment, we obtain a distribution over the clusters.

➤ Instead of centers, each cluster has a mean, a variance, and a weight.



Latent Variable Posteriors

We are interested in modeling the **distribution** (*the uncertainty*) over the **cluster assignment** z .

- **Prior:** $p(z_n) = \pi_{z_n}$
- **Likelihood:** $p(x_n | z_n) = \mathcal{N}(x_n | \mu_z, \sigma_z^2)$

We can use Bayes rule to compute the posterior:

$$p(z_n = k | x_n, \{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K) = \frac{p(x_n | z_n = k)p(z_n = k)}{\sum_{j=1}^K p(x_n | z_n = j)p(z_n = j)} = \frac{\mathcal{N}(x_n | \mu_k, \sigma_k^2)\pi_k}{\sum_{j=1}^K \mathcal{N}(x_n | \mu_j, \sigma_j^2)\pi_j}$$

If we **could estimate** $\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K$ **from the data** we could compute this distribution over cluster assignments for each data point.

We are interested in modeling the **distribution** (*the uncertainty*) over the **cluster assignment** z .

- **Prior:** $p(z_n) = \pi_{z_n}$

- **Likelihood:** $p(x_n | z_n) = \mathcal{N}(x_n | \mu_z, \sigma_z^2)$

We can use Bayes rule to compute the posterior:

$$p(z_n = k | x_n, \{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K) = \frac{p(x_n | z_n = k)p(z_n = k)}{\sum_{j=1}^K p(x_n | z_n = j)p(z_n = j)} = \frac{\mathcal{N}(x_n | \mu_k, \sigma_k^2)\pi_k}{\sum_{j=1}^K \mathcal{N}(x_n | \mu_j, \sigma_j^2)\pi_j}$$

If we **could estimate** $\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K$ **from the data** we could compute this distribution over cluster assignments for each data point.



9624861



9624861

Trying the Stationary Point Method

Can we just apply the stationary point approach to maximize the likelihood?

$$\arg \max_{\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K} \sum_n \ln \left(\sum_k \pi_k \mathcal{N}(x_n | \mu_k, \sigma_k^2) \right)$$

Taking the derivative with respect to μ_k and applying the **calculus chain** rule:

$$\frac{\partial}{\partial \mu_k} \sum_n \ln \left(\sum_j \pi_j \mathcal{N}(x_n | \mu_j, \sigma_j^2) \right) = \sum_n \frac{\pi_k \mathcal{N}(x_n | \mu_k, \sigma_k^2)}{\sum_j \pi_j \mathcal{N}(x_n | \mu_j, \sigma_j^2)} \cdot \frac{x_n - \mu_k}{\sigma_k^2}$$

To make math cleaner let's define $\gamma_{nk} = \frac{\pi_k \mathcal{N}(x_n | \mu_k, \sigma_k^2)}{\sum_j \pi_j \mathcal{N}(x_n | \mu_j, \sigma_j^2)}$ and set the derivative equal to 0:

$$\sum_n \gamma_{nk} \frac{x_n - \mu_k}{\sigma_k^2} = 0 \Rightarrow \mu_k = \frac{\sum_n \gamma_{nk} x_n}{\sum_n \gamma_{nk}}$$

K
 N
 γ

➤ Unfortunately, γ_{nk} depends on μ_k and so **we can't solve for μ_k analytically.**



9624861

Estimating the GMM Parameters

We could extend the model to include the **latent variable** z directly instead of marginalizing out their assignments:

$$\ln \left(p \left(\{x_n, z_n\}_{n=1}^N \mid \{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K \right) \right) = \sum_{n=1}^N \left(\ln(\pi_{z_n}) + \ln \left(\mathcal{N}(x_n \mid \mu_{z_n}, \sigma_{z_n}^2) \right) \right)$$

- We can now optimize the parameters directly (sum of log Gaussian terms).
- **But we don't know the values of $\{z_n\}_{n=1}^N$**

But, if we knew the parameters $\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K$ than we could estimate $\{z_n\}_{n=1}^N$:

$$p \left(z_n = k \mid x_n, \{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K \right) = \frac{\mathcal{N}(x_n \mid \mu_k, \sigma_k^2) \pi_k}{\sum_{j=1}^K \mathcal{N}(x_n \mid \mu_j, \sigma_j^2) \pi_j} = \gamma_{nk}$$

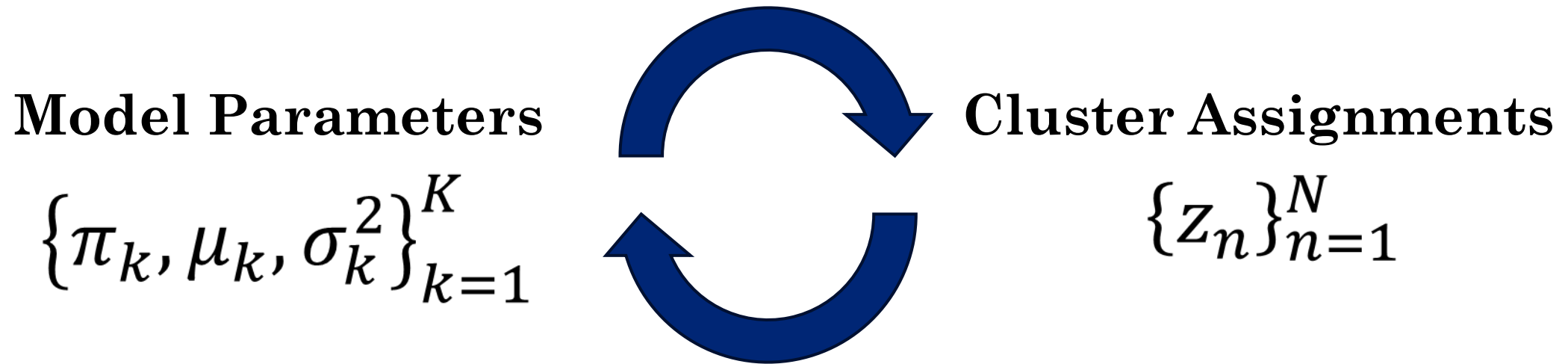
N K
 γ

Quick Recap



9624861

If we knew the model parameters, we could easily compute the cluster assignments.



If we knew the cluster assignments, we could easily estimate the model parameters.

How can we solve this cyclic dependency?



9624861

Intuition Behind Expectation-maximization (EM)

The **expectation-maximization algorithm** breaks the circle **by alternating**, in the same way **Lloyd's algorithm did for k-means** from Lecture 2.

Start with an initial guess for the parameters $\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K$:

Repeat until convergence:

- **E-step (Cluster Soft Assignment):** With the parameters fixed, compute the distribution over each point's **assignment using Bayes Theorem**.

$$p(z_n = k | x_n, \{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K) = \frac{\mathcal{N}(x_n | \mu_k, \sigma_k^2) \pi_k}{\sum_{j=1}^K \mathcal{N}(x_n | \mu_j, \sigma_j^2) \pi_j} = \gamma_{nk}$$

- **M-step (Move Clusters).** With the assignment probabilities fixed, choose the parameters that **maximize the expected log likelihood**.

Each iteration **increases the log likelihood**, so the procedure **converges**.



9624861

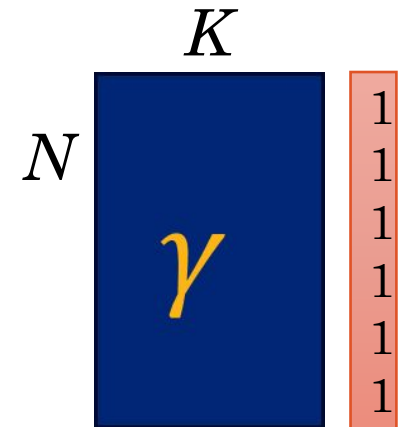
The EM Algorithm: **E**-step

E-Step: We compute “soft” assignments for the unknown $\{z_n\}_{n=1}^N$

$$\begin{aligned}\gamma_{nk} &= p\left(z_n = k \mid x_n, \{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K\right) = \frac{p(x_n \mid z_n = k)p(z_n = k)}{\sum_{j=1}^K p(x_n \mid z_n = j)p(z_n = j)} \\ &= \frac{\mathcal{N}(x_n \mid \mu_k, \sigma_k^2) \pi_k}{\sum_{j=1}^K \mathcal{N}(x_n \mid \mu_j, \sigma_j^2) \pi_j}\end{aligned}$$

The soft assignments sum to $N = \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk}$ and we define the **cluster pseudo counts**:

$$N_k = \sum_{n=1}^N \gamma_{nk}$$

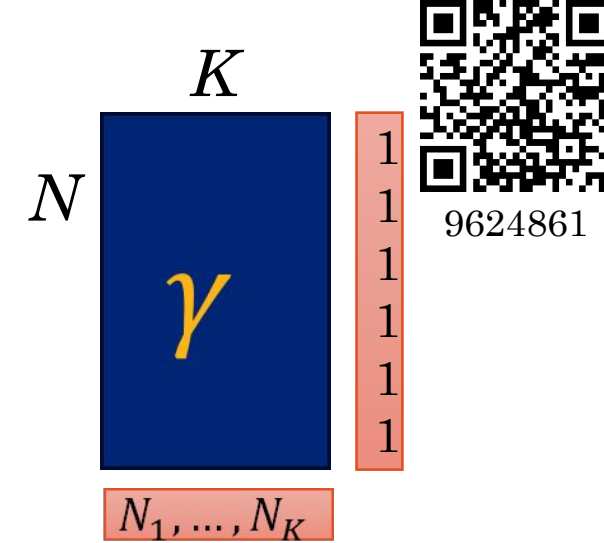


Cluster “pseudo counts” $N_1, \dots, N_K$

The EM Algorithm: **E**-step

E-Step: We compute “soft” assignments for the unknown $\{z_n\}_{n=1}^N$

$$\gamma_{nk} = \frac{\mathcal{N}(x_n \mid \mu_k, \sigma_k^2) \pi_k}{\sum_{j=1}^K \mathcal{N}(x_n \mid \mu_j, \sigma_j^2) \pi_j}$$



Using the γ_{nk} we can compute the **expected log likelihood**:

$$\begin{aligned} Q\left(\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K\right) &= \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \ln \left(p\left(x_n, z_n = k \mid \pi, \{\mu_k, \sigma_k^2\}_{k=1}^K\right) \right) \\ &= \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln \left(\mathcal{N}(x_n \mid \mu_k, \sigma_k^2) \right) \right) \end{aligned}$$

We can now **maximize** the expected log likelihood.



9624861

The EM Algorithm: **M-Step** (π_k)

M-Step: Compute the parameters that **maximize** the expected log likelihood Q :

$$Q\left(\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K\right) = \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln\left(\mathcal{N}(x_n | \mu_k, \sigma_k^2)\right) \right)$$

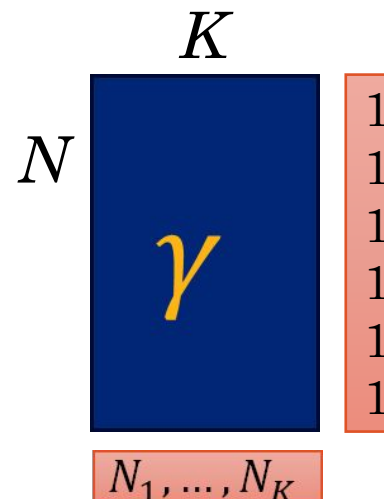
Optimizing with respect to π_k :

Method of Lagrange Multipliers (Lecture 3)
for the **normalization constraint**

$$\frac{\partial}{\partial \pi_k} \left(\sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln\left(\mathcal{N}(x_n | \mu_k, \sigma_k^2)\right) \right) - \lambda \left(\sum_{k=1}^K \pi_k - 1 \right) \right)$$

$$= \sum_{n=1}^N \frac{\gamma_{nk}}{\pi_k} - \lambda = 0 \quad \Rightarrow \quad \pi_k = \frac{1}{\lambda} \sum_{n=1}^N \gamma_{nk} = \frac{N_k}{\lambda}$$

$$N_k = \sum_{n=1}^N \gamma_{nk}$$





9624861

The EM Algorithm: **M-Step** (π_k)

M-Step: Compute the parameters that **maximize** the expected log likelihood Q :

$$Q\left(\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K\right) = \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln\left(\mathcal{N}(x_n | \mu_k, \sigma_k^2)\right) \right)$$

Optimizing with respect to π_k :

$$\pi_k = \frac{1}{\lambda} \sum_{n=1}^N \gamma_{nk} = \frac{N_k}{\lambda}$$

Need figure out the
Lagrange Multiplier

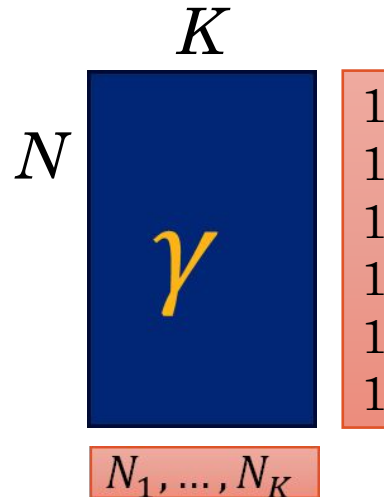
We know that $\sum_{k=1}^K \pi_k = 1$ so:

$$\sum_{k=1}^K \pi_k = \sum_{k=1}^K \frac{N_k}{\lambda} = \frac{N}{\lambda} = 1$$

$$\pi_k = \frac{N_k}{N}$$

Intuitively this is the
proportion of points
in cluster k

therefore $\lambda = N$





9624861

The EM Algorithm: **M-Step** (μ_k)

M-Step: Compute the parameters that maximize Q :

$$Q\left(\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K\right) = \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln\left(\mathcal{N}(x_n | \mu_k, \sigma_k^2)\right) \right)$$

Optimizing with respect to μ_k :

$$\frac{\partial}{\partial \mu_k} \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln\left(\mathcal{N}(x_n | \mu_k, \sigma_k^2)\right) \right) = \sum_{n=1}^N \gamma_{nk} \sigma_k^{-2} (x_n - \mu_k) = 0$$

$$\sigma_k^{-2} \sum_{n=1}^N \gamma_{nk} x_n = \sigma_k^{-2} \mu_k \sum_{n=1}^N \gamma_{nk} = \sigma_k^{-2} \mu_k N_k$$

$$\mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} x_n$$



9624861

The EM Algorithm: **M-Step** (σ_k^2)

M-Step: Compute the parameters that maximize Q :

$$Q\left(\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K\right) = \sum_{n=1}^N \sum_{k=1}^K \gamma_{nk} \left(\ln(\pi_k) + \ln\left(\mathcal{N}(x_n | \mu_k, \sigma_k^2)\right) \right)$$

Optimizing with respect to σ_k^2 follows the same stationary point derivation as the single Gaussian, now weighted by the responsibilities γ_{nk} :

$$\sigma_k^2 = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} (x_n - \mu_k)^2$$

Summarizing the other parameters:

$$N_k = \sum_{n=1}^N \gamma_{nk}, \quad \pi_k = \frac{N_k}{N}, \quad \mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} x_n$$



9624861

The EM Algorithm for GMMs

1. **Initialize** the model parameters $\{\pi_k, \mu_k, \sigma_k^2\}_{k=1}^K$ (often using k-means)
2. **Iterate** until convergence

- **E-Step:** compute expected log-likelihood

$$\gamma_{nk} = \frac{\mathcal{N}(x_n | \mu_k, \sigma_k^2) \pi_k}{\sum_{j=1}^K \mathcal{N}(x_n | \mu_j, \sigma_j^2) \pi_j}$$

- **M-Step:** maximize the expected log-likelihood

$$N_k = \sum_{n=1}^N \gamma_{nk}, \quad \pi_k = \frac{N_k}{N}, \quad \mu_k = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} x_n$$
$$\sigma_k^2 = \frac{1}{N_k} \sum_{n=1}^N \gamma_{nk} (x_n - \mu_k)^2$$



9624861

The EM Algorithm

The **Expectation-Maximization (EM)** algorithm is a standard technique for fitting **latent variable** models like the Gaussian Mixture Model.

1. **Initialize** the model parameters θ .
2. **Iterate** until convergence.
 - **E-Step:** Compute expected log-likelihood.

$$Q(\theta') = \sum_Z \ln(p(X, Z | \theta')) p(Z | X, \theta)$$

Easy to optimize
joint probability

Current distribution
over the latent Z

- **M-Step:** Maximize the expected log-likelihood.

$$\theta = \arg \max_{\theta'} Q(\theta')$$

Updates distribution over Z .



9624861

Convergence and Local Optima

Except for the **variance collapse failure modes**, the likelihood is bounded from above and the EM Algorithm **only increases** the likelihood with each change in parameters:

➤ typically to a **local** maximum, not necessarily the global one.

Initialization matters.

➤ Different starting points can land on different local optima

➤ Run k-means first: use its centers for the initial μ_k , the whole-dataset variance for σ_k^2 , and $\frac{1}{K}$ for π_k (default for sklearn's GaussianMixture model).

A failure mode specific to mixtures: if a component's mean lands on a single data point and $\sigma_k^2 \rightarrow 0$, that term's likelihood diverges.

➤ Often set a variance floor (or a MAP/Bayesian prior penalizing tiny variances) and sensible initialization; (usually enough)

Convergence, Initialization, and Local Optima



9624861

Each EM iteration increases the log likelihood, and the likelihood is bounded above, so the algorithm converges.

It converges to a **local** maximum. Different starting points give different answers, exactly as Lloyd's algorithm did in Lecture 2.

Initialization matters.

- Running k-means first and using its centers as the initial μ_k is standard practice.
- The initial σ_k^2 are often set to the variance of the whole dataset, and the π_k to $1/K$.
- `sklearn.mixture.GaussianMixture` defaults to k-means initialization and `n_init=1`.

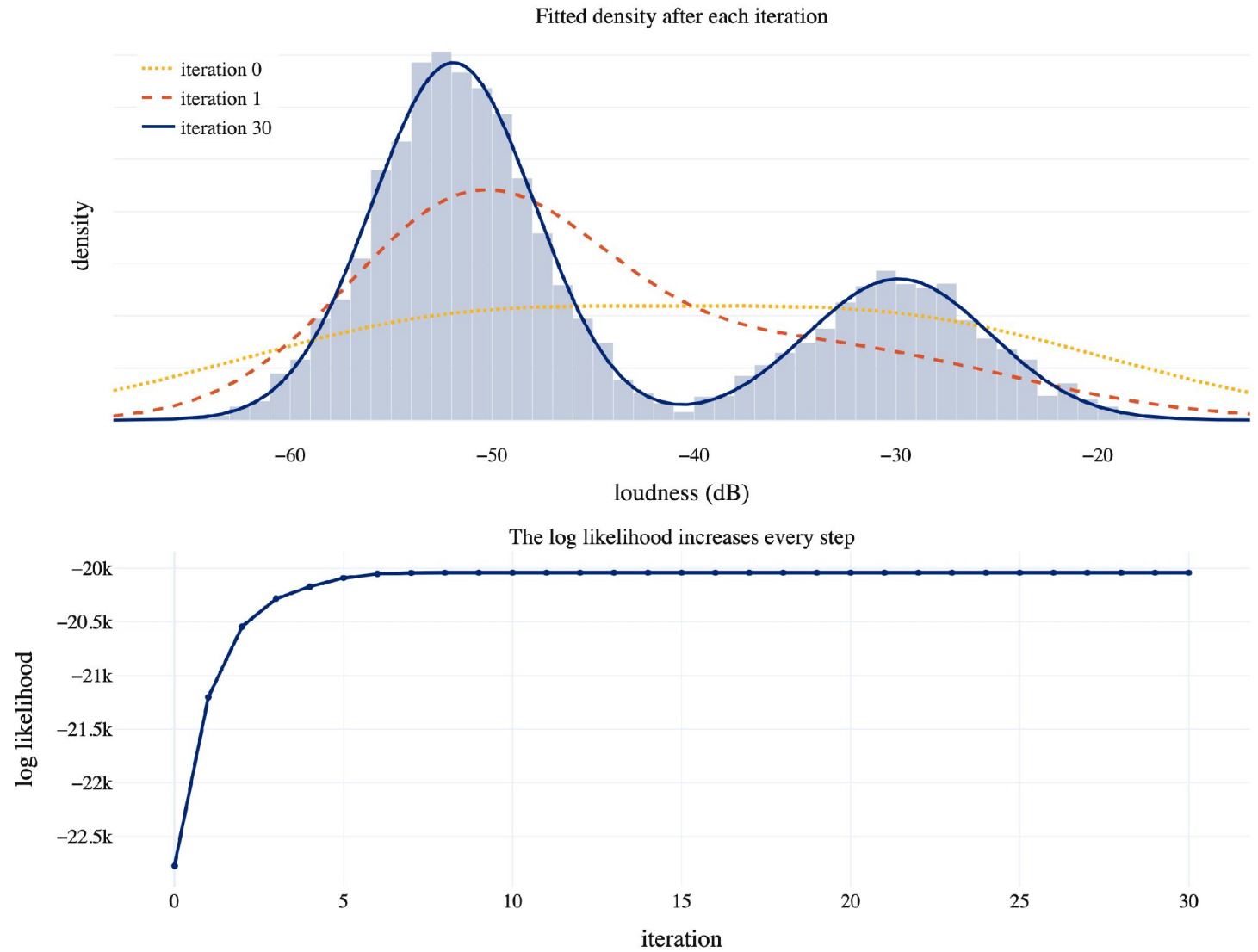
A failure mode specific to mixtures. If one component collapses onto a single point, its variance σ_k^2 shrinks toward zero and the likelihood diverges.

- The likelihood is unbounded, so a global maximum does not exist in any useful sense.
- Implementations add a small constant to σ_k^2 to prevent it.



Demo

Implementing EM for GMMs





9624861

What We Did Today

Fitting a Gaussian. Maximum likelihood returns the sample mean and the sample variance, which depend on the data only through $\sum_n x_n$ and $\sum_n x_n^2$.

Bias. The maximum likelihood variance is low by a factor of $\frac{N-1}{N}$, because deviations are measured from a mean estimated on the same data.

Mixture models. A weighted sum of Gaussians describes data that a single Gaussian cannot. The component that generated each point is a latent variable z_n .

Why EM is necessary. The log likelihood of a mixture puts a sum inside the logarithm, so the stationary point method yields no closed-form solution.

EM. Alternate between computing the responsibilities γ_{nk} by Bayes' theorem and maximizing the expected complete-data log likelihood. Each step is closed form; together they converge to a local maximum.

Lecture 5

Density Estimation and Gaussian Mixture Models

Credit: Joseph E. Gonzalez and Narges Norouzi

Reference Book Chapters:

- **Probability:** Chapter 2.[1-2], 2.3.[1-3] (we will return to 2.4 onward later)
- **Distributions:** Chapter 3.1, 3.2.1 (the rest of 3 is more advanced Gaussians)
- **Clustering:** Chapter 15.1 (k-means), 15.2 (Gaussian Mixture Models)