SPRING 2023

CS 378: INTRO TO SPEECH AND AUDIO PROCESSING

Gaussian Mixture Models

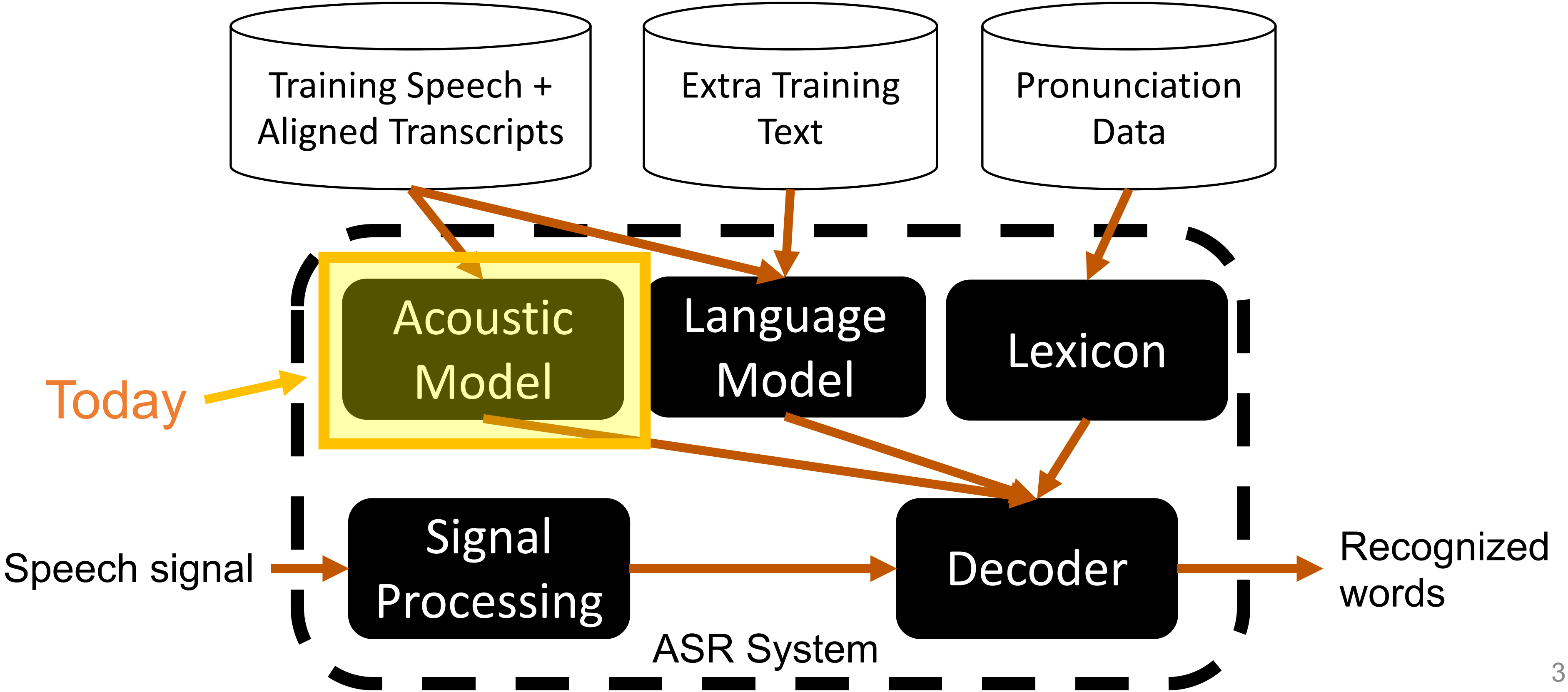
DAVID HARWATH
Assistant Professor, UTCS

The University of Texas at Austin
Department of Computer Science
College of Natural Sciences

Agenda

- Acoustic Modeling Overview
- Gaussian Distributions
- Gaussian Mixture Models
  - K-means
  - Training GMMs with Expectation-Maximization

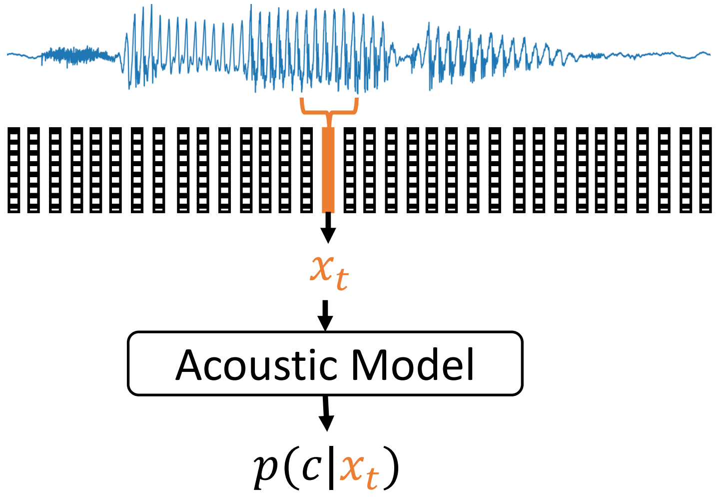
Components of an ASR system



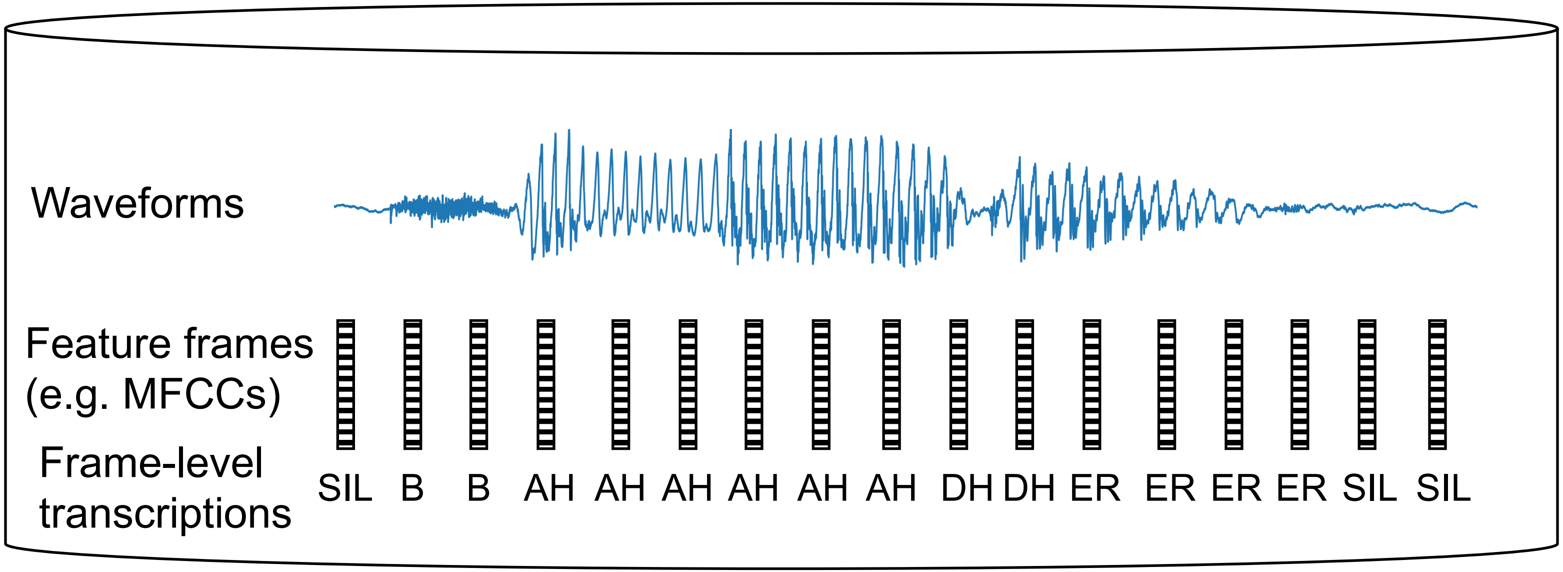
The Setting for Today

- How can we build a classifier to predict which particular speech sound is active at a particular point in time?

- What are Gaussian Mixture Models (GMMs) and how can we use them for this task?



The Setting for Today

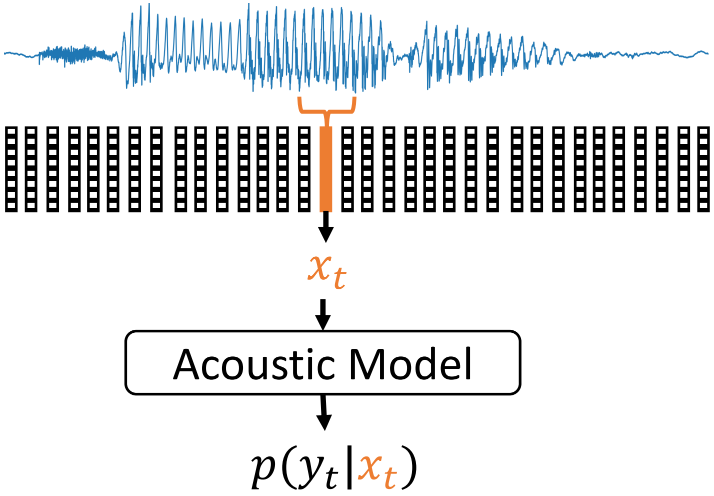


Assume we are given a collection of waveforms represented by features such as MFCCs, and that we have a phonetic state label for every single frame.

Our goal: Use this training data to build a classifier that can predict phonetic state labels for new (unseen) speech frames at test time.

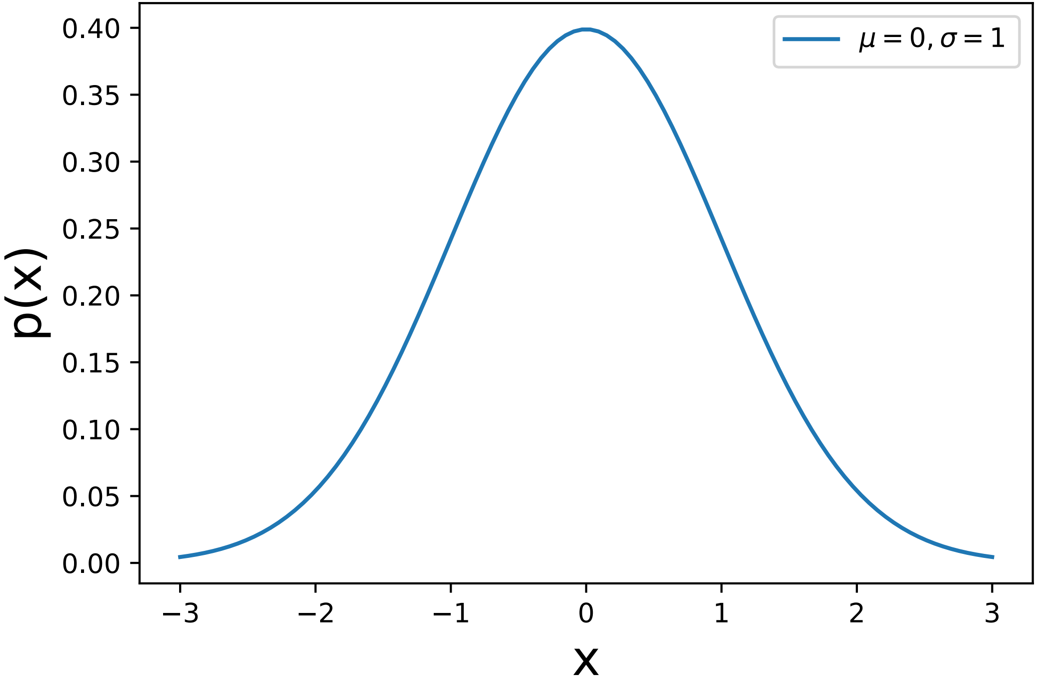
The Setting for Today

- We will discuss acoustic modeling today purely as a multi-class classification problem.



- In principle, any machine learning model capable of multi-class classification could be used here.
- However, we will focus our attention on Gaussian Mixture Models (GMMs) as they have historically been dominant. (we will look at neural net acoustic models next lecture)

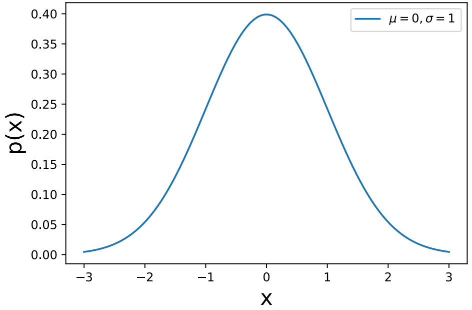
1-D Gaussian Distributions



1-D Gaussian PDF

- Probability density function:

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



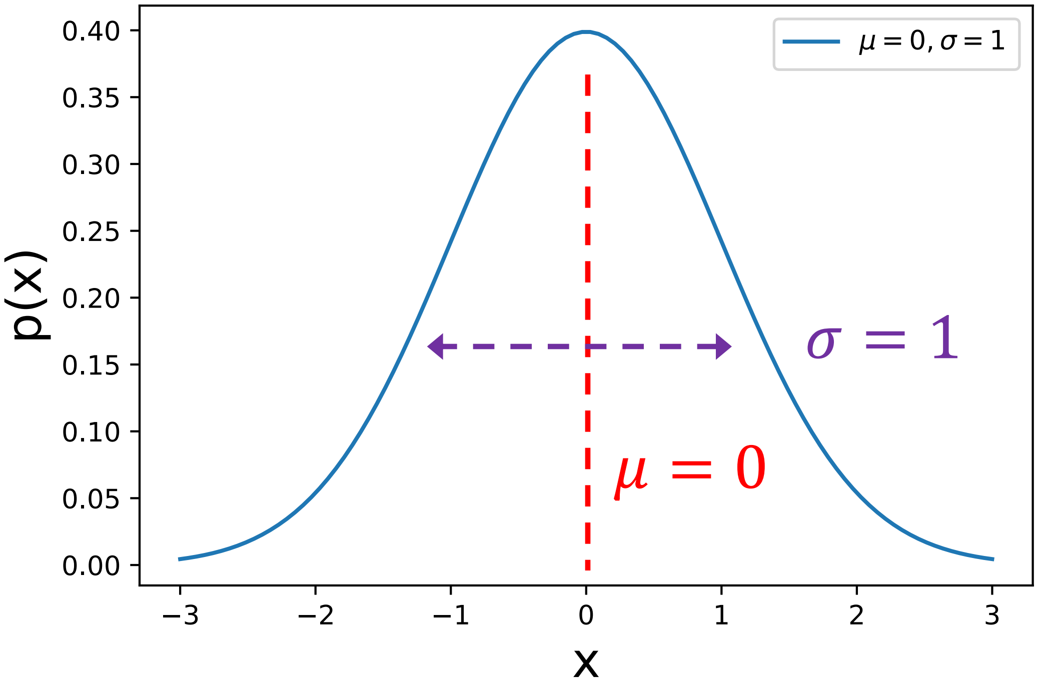
- Parameters:
  - μ : the mean of the distribution

$$\mu = E[x] = \int x\, p(x)\, dx$$

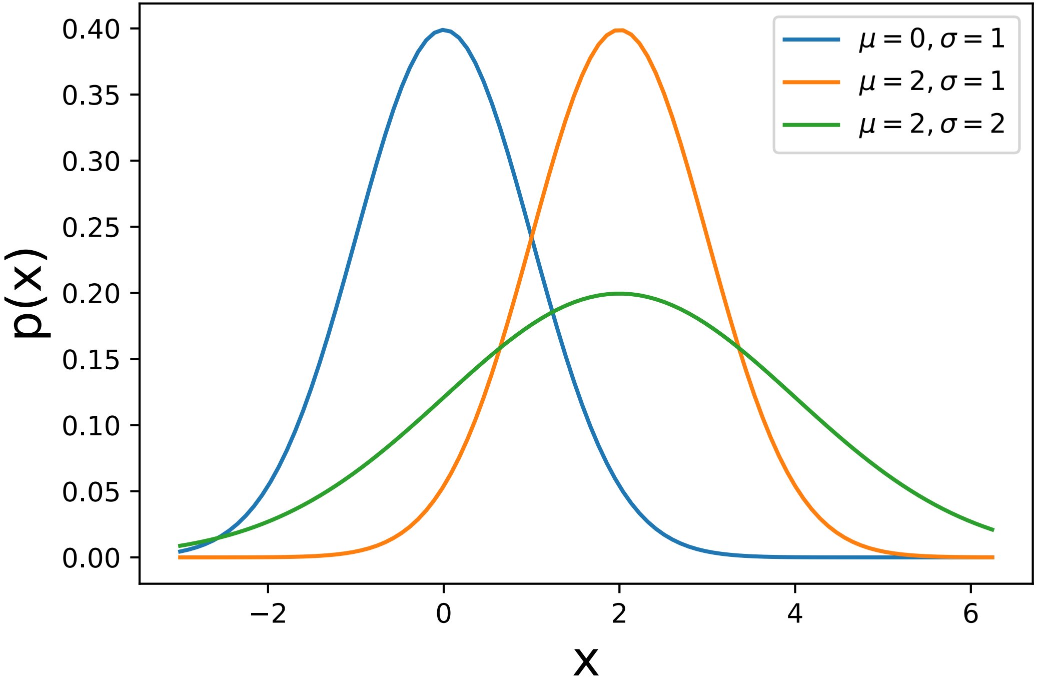
  - σ^2 : the variance of the distribution (σ is called the standard deviation)

$$\sigma^2 = E[(x-\mu)^2] = \int (x-\mu)^2\, p(x)\, dx$$

1-D Gaussian Distributions



1-D Gaussian Distributions



More on Gaussians

- To indicate a random variable x has a Gaussian distribution, we often write $x \sim \mathcal{N}(\mu, \sigma^2)$
- The PDF can also be written $p(x \mid \mu, \sigma^2) = \mathcal{N}(x; \mu, \sigma^2)$
- “Normal distribution” is another common name
- Possibly the most widely-used continuous probability distribution
  - Easy to mathematically analyze and fit to data
  - Many real-world random variables happen to be Gaussian (Central Limit Theorem: sum of many independent R.V.s tends Gaussian)
  - One way to conceptualize the distribution: perturbations around some average value μ

Fitting Distributions to Data

- Fitting distributions to data in general is often cast as an optimization problem where we have 3 items:
  - The datapoints we want to fit
  - The parameters of the distribution (model parameters)
  - An objective function that measures how well the parameters fit the data
- Fitting the distribution boils down to algorithmically adjusting the parameters to maximize the objective function.

Fitting 1-D Gaussians to Data

- Assume we are given a dataset $X = \{x_1, \dots x_N\}$
- We have model parameters $\theta = \{\mu, \sigma\}$
- Most common objective for fitting Gaussians is Maximum Likelihood (ML), or Maximum Likelihood Estimation (MLE)

$$\text{maximize}_\theta \; p(X \mid \theta)$$

- Assume x_i's are independent and identically distributed (i.i.d.):

$$p(X; \mu, \sigma) = p(x_1, \dots, x_N \mid \theta) = \prod_{i=1}^{N} p(x_i \mid \theta)$$

MLE for 1-D Gaussians

- We have that

$$p(X;\mu,\sigma) = \prod_{i=1}^{N} p(x_i \mid \theta) = \frac{1}{\sigma\sqrt{2\pi}} \prod_{i=1}^{N} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

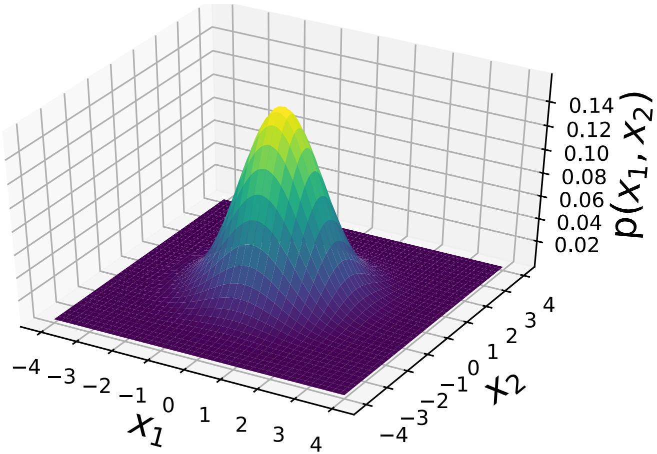
- Differentiating the logarithm of the above term (for mathematical convenience, does not change the solution) w.r.t μ and σ^2, setting it equal to 0 and solving for μ and σ^2 leads to their ML estimates:

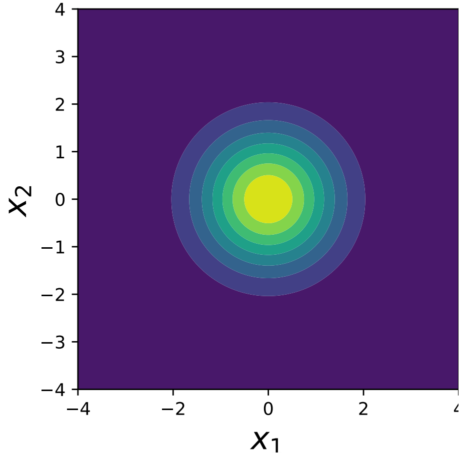
$$\mu^* = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\sigma^{2^*} = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu^*)^2$$

Multivariate Gaussian Distributions

- What if each $\boldsymbol{x}_i \in \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$ is a vector in $\mathbb{R}^D$?
  - Recall that typical MFCC vectors are in $\mathbb{R}^{39}$
- We can extend the Gaussian distribution to $\mathbb{R}^D$
- This is often called a *multivariate Gaussian*





Multivariate Gaussian Distributions

- Probability density function:

$$p(\boldsymbol{x}) = \frac{1}{(|\boldsymbol{\Sigma}|)^{\frac{1}{2}}(2\pi)^{\frac{D}{2}}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$$

- Parameters:
  - $\boldsymbol{\mu}$: the mean vector, with ML estimate

$$\boldsymbol{\mu}^* = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i$$

  - $\boldsymbol{\Sigma}$: the covariance matrix, with ML estimate

$$\boldsymbol{\Sigma}^* = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \boldsymbol{\mu}^*)(\boldsymbol{x}_i - \boldsymbol{\mu}^*)^T$$

Understanding $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

- $\boldsymbol{\mu}$ is simply the empirical mean vector of $\boldsymbol{X}$
- $\boldsymbol{\Sigma}$ captures the covariance between every pair of dimensions

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{11}^2 & \cdots & \sigma_{1N}^2 \\ \vdots & \ddots & \vdots \\ \sigma_{N1}^2 & \cdots & \sigma_{NN}^2 \end{bmatrix}$$

- $\boldsymbol{\Sigma}$ must be positive semi-definite to be valid

Types of Covariance Matrices

- Spherical (Isotropic) covariance matrix

$$\boldsymbol{\Sigma} = \boldsymbol{\sigma}^2 \boldsymbol{I}$$

- Diagonal covariance matrix

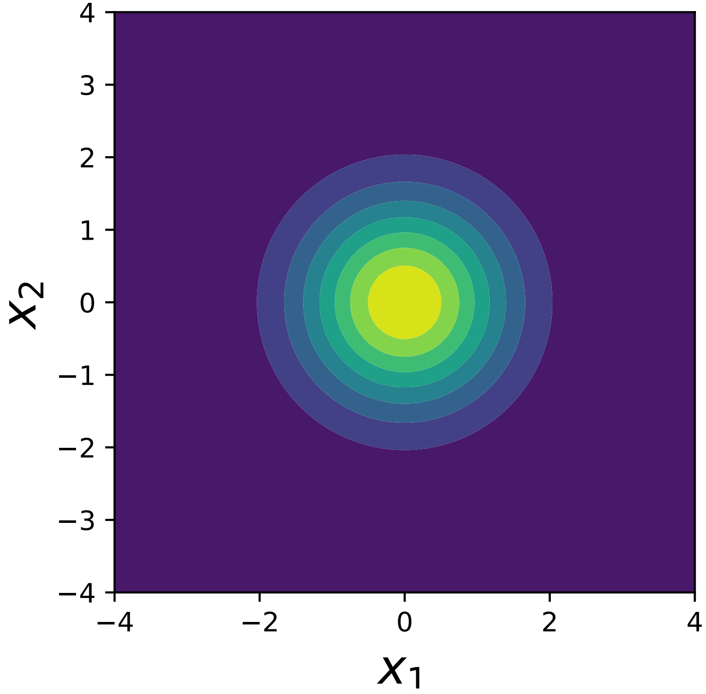
$$\boldsymbol{\Sigma} = diag([\sigma_1^2, \ldots, \sigma_N^2])$$

- Full covariance matrix

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma_{11}^2 & \cdots & \sigma_{1N}^2 \\ \vdots & \ddots & \vdots \\ \sigma_{N1}^2 & \cdots & \sigma_{NN}^2 \end{bmatrix}$$

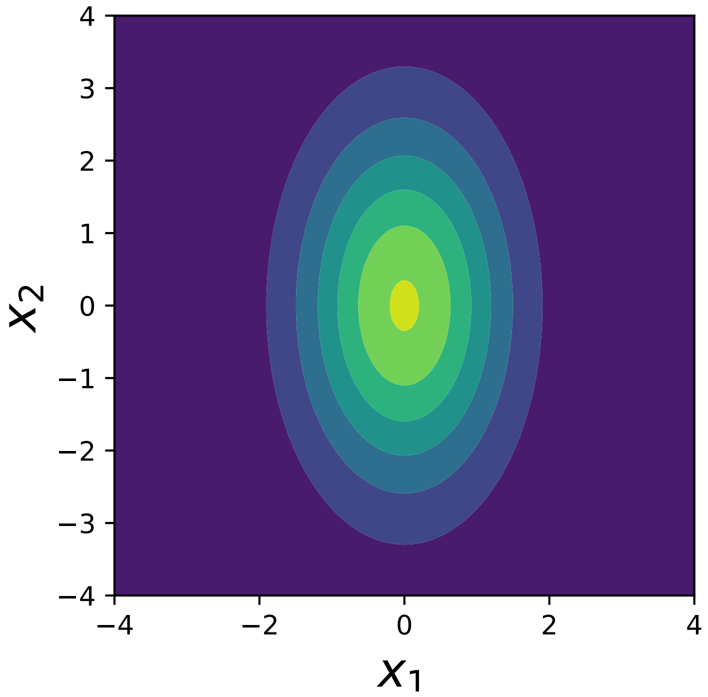
Spherical Covariance Matrix

- $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
- Individual dimensions assumed to be uncorrelated
- Each dimension assumed to have the same variance σ
- 1 parameter



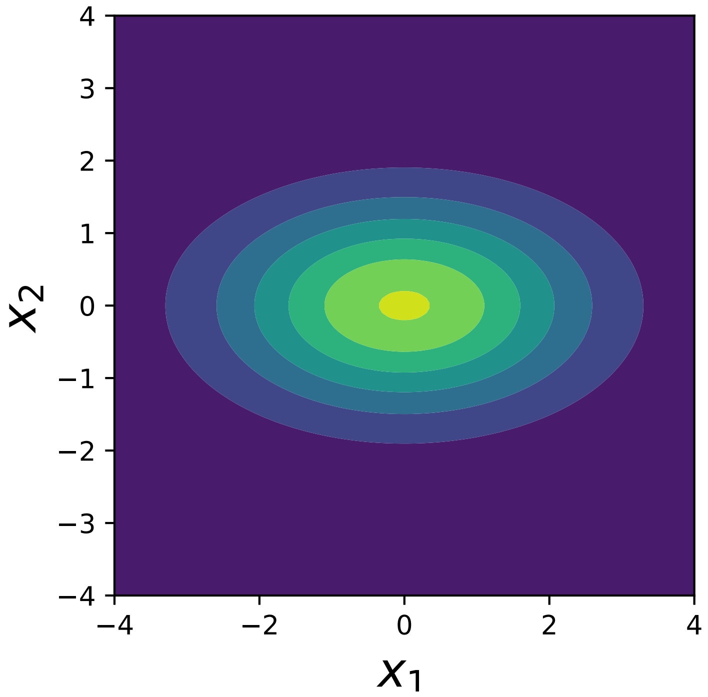
Diagonal Covariance Matrix

- $\mathbf{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$
- Individual dimensions assumed to be uncorrelated
- Each dimension has its own variance parameter σ_i
- N parameters



Diagonal Covariance Matrix

- $\mathbf{\Sigma} = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$
- Individual dimensions assumed to be uncorrelated
- Each dimension has its own variance parameter σ_i
- N parameters



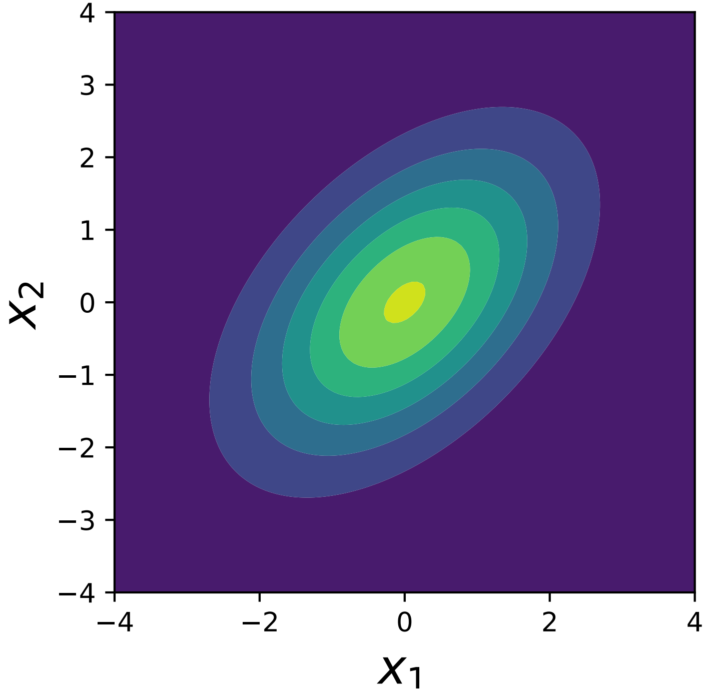
Full Covariance Matrix

- $\boldsymbol{\Sigma} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

- Dimensions may be correlated

- All covariance terms can vary (subject to PSD matrix)

- $N(N+1)/2$ parameters



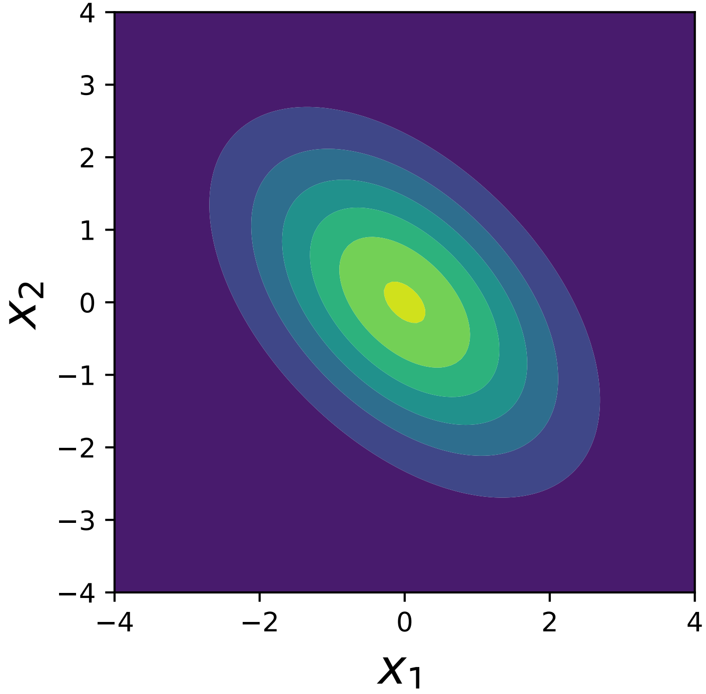
Full Covariance Matrix

- $\mathbf{\Sigma} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$

- Dimensions may be correlated

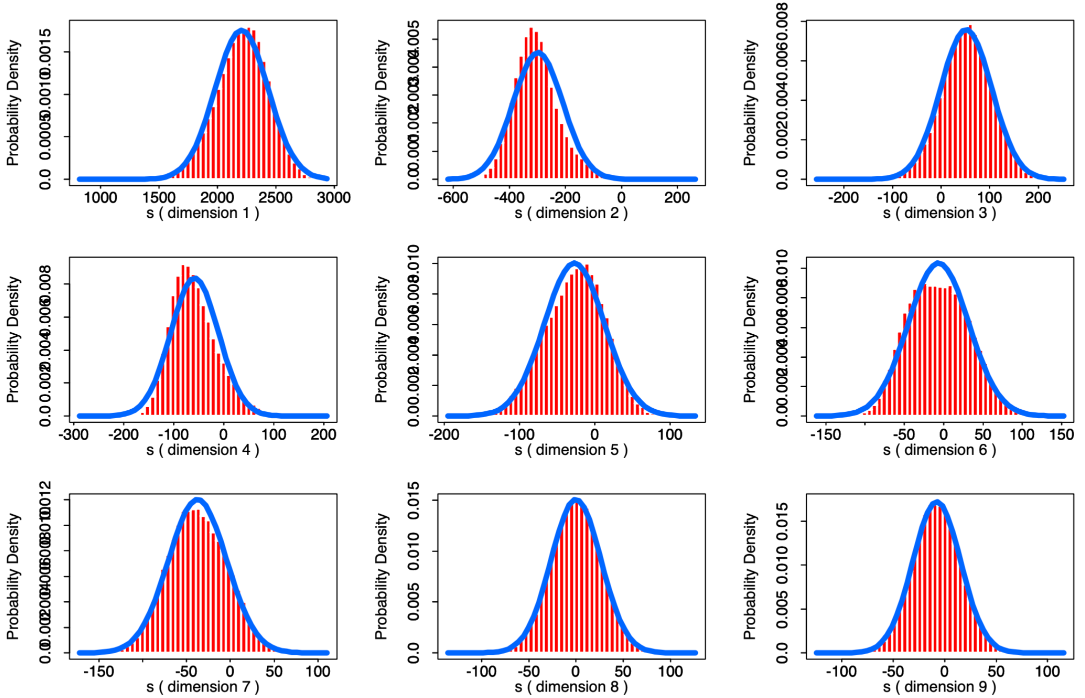
- All covariance terms can vary (subject to PSD matrix)

- $N(N+1)/2$ parameters



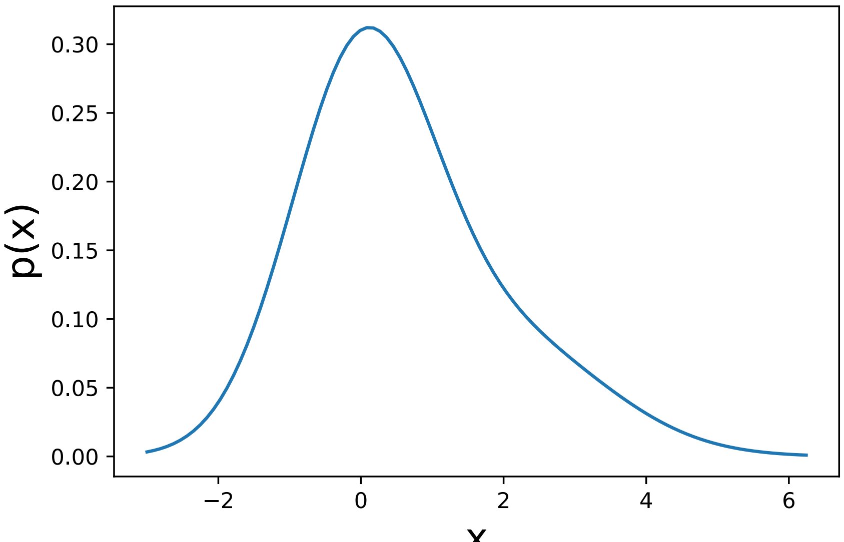
Fitting Gaussians to MFCCs

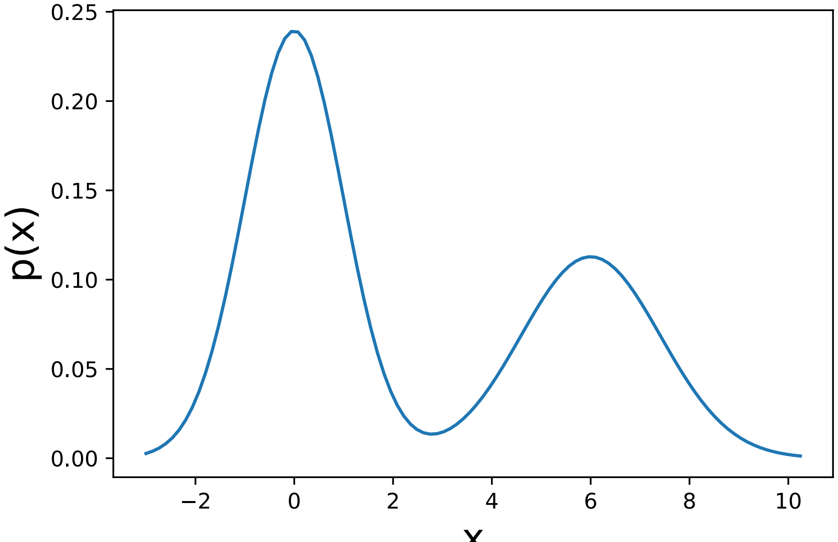
First 9 MFCC's from [s]: Gaussian PDF



Mixtures of Gaussians

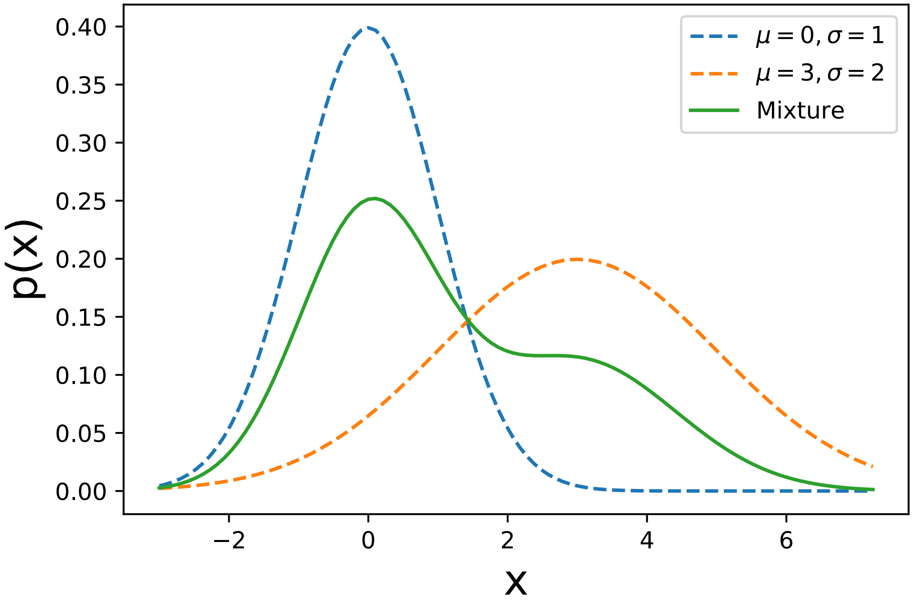
- Gaussians have convenient properties, but what if our data doesn’t *quite* follow a Gaussian distribution?





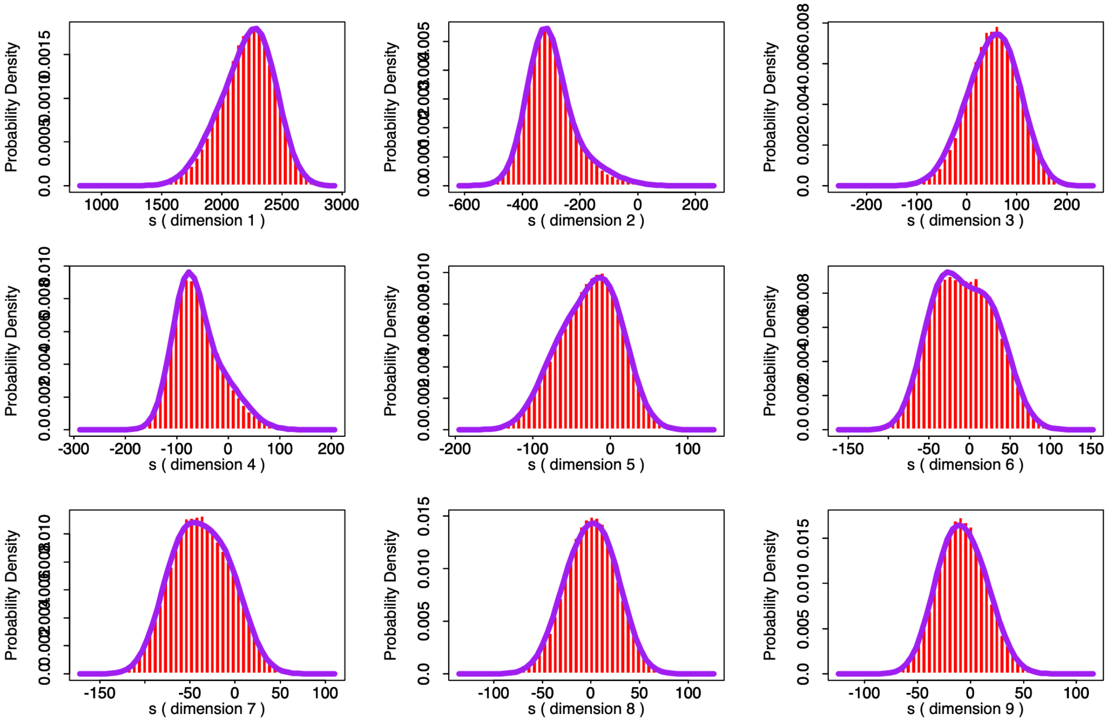
Mixtures of Gaussians

- We can model much more complex distributions by using a *mixture* (weighted sum) of several Gaussians



Fitting GMMs to MFCCs

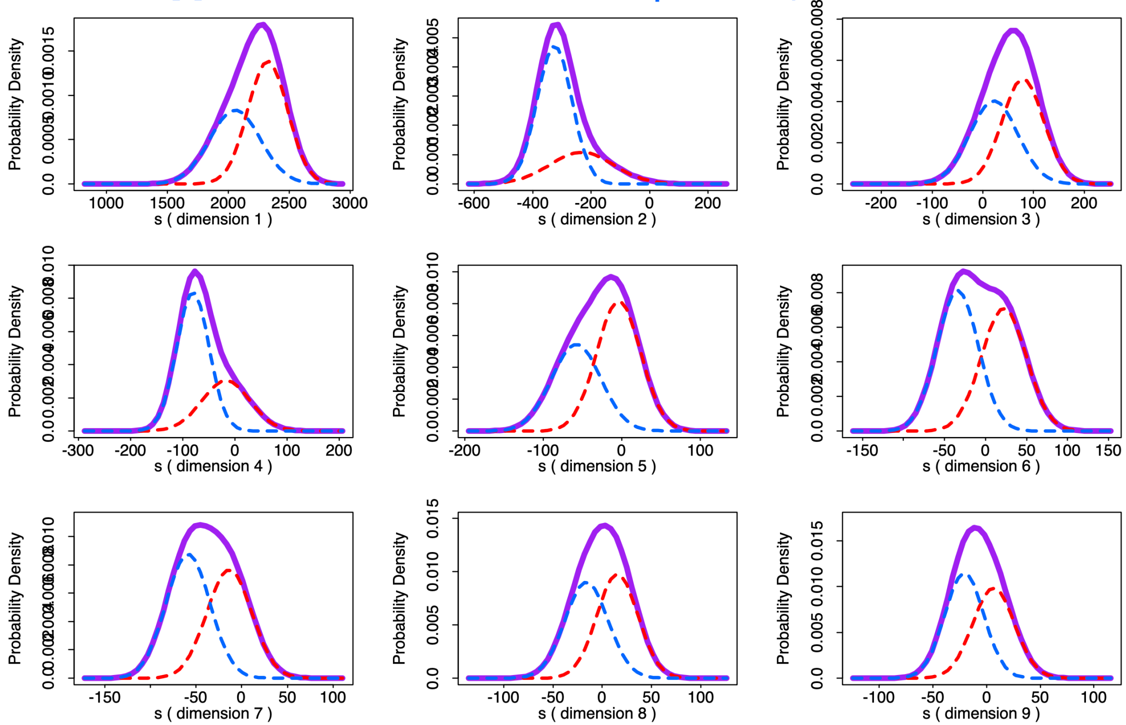
[s]: 2 Gaussian Mixture Components/Dimension



Adapted from *James Glass, and Victor Zue. 6.345 Automatic Speech Recognition. Spring 2003. Massachusetts Institute of Technology: MIT OpenCourseWare, https://ocw.mit.edu. License: [Creative Commons BY-NC-SA](https://creativecommons.org).*

Fitting GMMs to MFCCs

[s]: 2 Gaussian Mixture Components/Dimension



Gaussian Mixture Models

A Gaussian Mixture Model (GMM) is parameterized by:

1. A set of K Gaussian components, $\{(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \ldots, (\boldsymbol{\mu}_K, \boldsymbol{\Sigma}_K)\}$
2. A set of component weights $\{w_1, \ldots, w_K\}$ that form a categorical distribution, i.e. $\sum_k w_k = 1$

The probability density of the GMM is given by:

$$p(\boldsymbol{x}|\theta) = \sum_{k=1}^{K} w_k \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

The GMM Generative Story

We can tell a ”story” about how a GMM generated our dataset, given parameters $\{(w_1, \boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1), \dots, (w_K, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_K)\}$

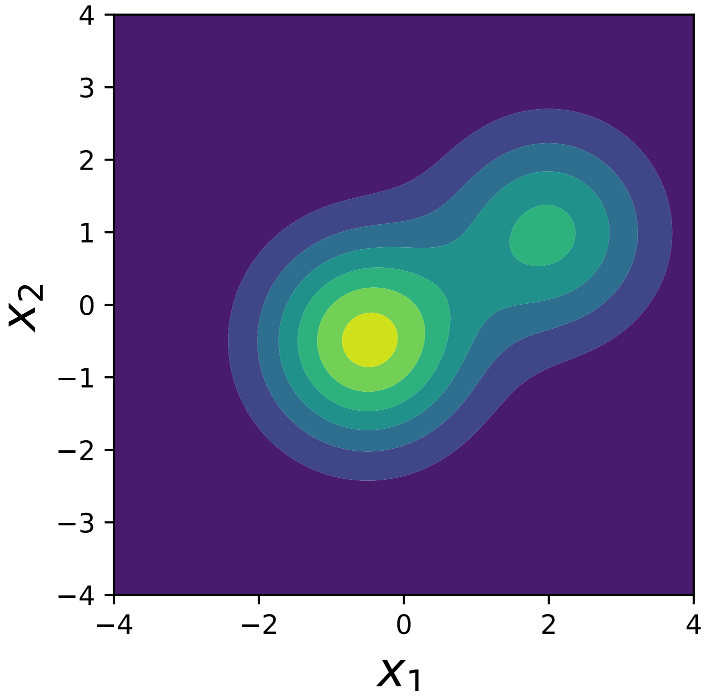
For $i = 1, \dots, N$:

1. Randomly sample a Gaussian component index $k \sim Categorical(w_1, \dots, w_K)$
2. Sample $\boldsymbol{x}_i \sim \mathcal{N}(x; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$

2-D Spherical Covariance GMM

- Component 1
  - $\mu = [-0.5, -0.5]^T$
  - $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
  - $w = 0.6$

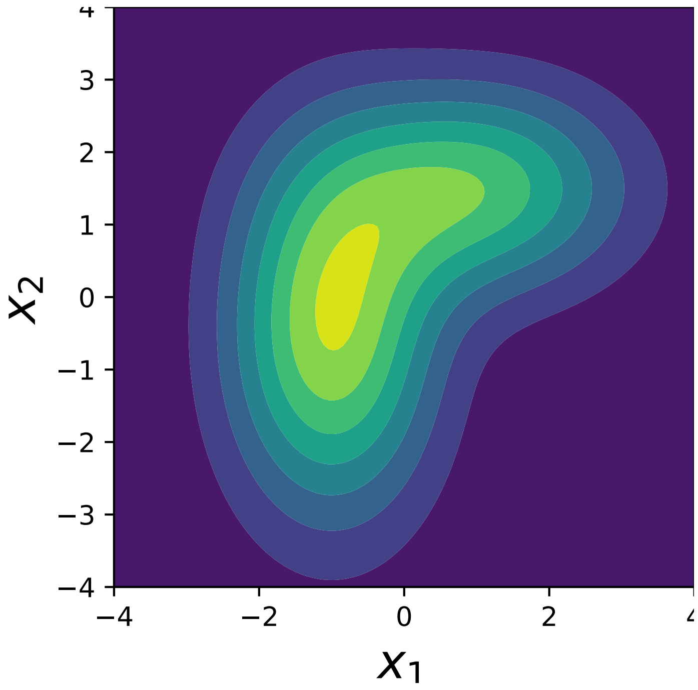
- Component 2
  - $\mu = [2,\ 1]^T$
  - $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
  - $w = 0.4$



2-D Diagonal Covariance GMM

- Component 1
  - $\mu = [-1, -0.5]^T$
  - $\Sigma = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$
  - $w = 0.6$

- Component 2
  - $\mu = [1,\ 0.5]^T$
  - $\Sigma = \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix}$
  - $w = 0.4$



Fitting GMMs to Data

- Assume we are given a dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \dots \boldsymbol{x}_N\}$
- We have model parameters $\theta = \{\theta_1, \dots, \theta_K\}, \theta_k = \{w_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$
- We can still use MLE for fitting a GMM

$$\text{maximize}_\theta \; p(\boldsymbol{X} \mid \theta)$$

- Assume $\boldsymbol{x}_i$'s are independent and identically distributed (i.i.d.):

$$p(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \mid \theta) = \prod_{i=1}^{N} \sum_{k=1}^{K} w_k \, \mathcal{N}(\boldsymbol{x}_i \,; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Fitting GMMs to Data

- Data likelihood under a GMM:

$$p(\boldsymbol{x}_1, \dots, \boldsymbol{x}_N \mid \theta) = \prod_{i=1}^{N} \sum_{k=1}^{K} w_k \, \mathcal{N}(\boldsymbol{x}_i \, ; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Directly maximizing this likelihood turns out to be intractable!

- However, think back to our generative story and pretend for a moment that we know *which* Gaussian component k was responsible for generating each $\boldsymbol{x}_i$…

Fitting GMMs to Data

- Define the indicator variable $z_i^k = 1$ if the k^{th} Gaussian component generated datapoint $\boldsymbol{x}_i$, and 0 otherwise.
- If we knew the value of each z_i^k, we could easily estimate the parameters of the GMM (in a ML sense) like so:

$$w_k = \frac{1}{N}\sum_{i=1}^{N} z_i^k \qquad \boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} z_i^k \, \boldsymbol{x}_i}{\sum_{i=1}^{N} z_i^k}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} z_i^k (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^{N} z_i^k}$$

Fitting GMMs to Data

Now let's look at this from the opposite angle.

Assume that we *didn't* know z_i^k, but we *do* know each $w_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i$.

We can then use the model + Bayes' Rule to *infer* z_i^k given $\boldsymbol{x}_i$:

$$P\left(z_i^k \middle| \boldsymbol{x}_i\right) = \frac{p\left(\boldsymbol{x}_i \middle| z_i^k\right) P(z_i^k)}{p(\boldsymbol{x}_i)} = \frac{\mathcal{N}(\boldsymbol{x}_i\,; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_\mathrm{k}) w_k}{\sum_{l=1}^{K} \mathcal{N}(\boldsymbol{x}_i\,; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) w_l}$$

Fitting GMMs to Data

To summarize:

- We want to estimate $\theta = \{\theta_1, \dots, \theta_K\}$ from $\{\boldsymbol{x}_1, \dots \boldsymbol{x}_N\}$, but naïve MLE ends up being intractable
- If we knew each z_i^k, we could estimate θ
- If we knew θ, we could infer each $P\left(z_i^k \middle| \boldsymbol{x}_i\right)$

This gives rise to an iterative algorithm (Expectation-Maximization, or EM) in which we alternate between 2 steps until convergence:

1. Given the current value of θ, compute $P\left(z_i^k \middle| \boldsymbol{x}_i\right)$
2. Substitute $P\left(z_i^k \middle| \boldsymbol{x}_i\right)$ in place of z_i^k, and update our estimate for θ

The E-M Algorithm

- Expectation-Maximization (E-M) can be thought of as a *genre* of algorithms that are used to solve problems that involve *missing information*, AKA hidden variables.
- Typical properties of problems for which E-M is useful:
    - Jointly solving for the parameters and hidden variables is hard
    - If the hidden variables were known, estimating the model parameters would be easy
    - If the model parameters were known, inferring the hidden variables would be easy

The E-M Algorithm for GMMs

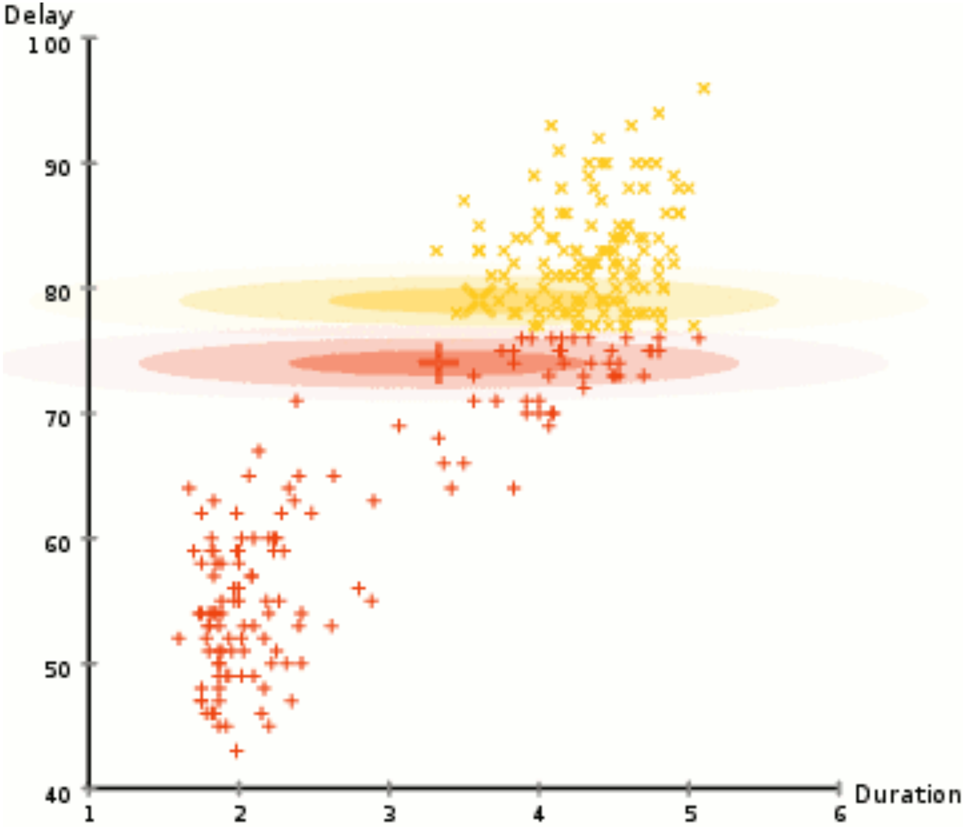
1. E-Step: $$P\left(z_i^k \middle| \boldsymbol{x}_i\right) = \frac{P\left(\boldsymbol{x}_i \middle| z_i^k\right) P\left(z_i^k\right)}{P(\boldsymbol{x}_i)} = \frac{\mathcal{N}(\boldsymbol{x}_i\,; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_\mathrm{k}) w_k}{\sum_{l=1}^{K} \mathcal{N}(\boldsymbol{x}_i\,; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) w_l}$$

2. M-Step: $$w_k = \frac{1}{N} \sum_{i=1}^{N} P\left(z_i^k \middle| \boldsymbol{x}_i\right) \qquad \boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} P\left(z_i^k \middle| \boldsymbol{x}_i\right) \boldsymbol{x}_i}{\sum_{i=1}^{N} P\left(z_i^k \middle| \boldsymbol{x}_i\right)}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} P\left(z_i^k \middle| \boldsymbol{x}_i\right)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^{N} P\left(z_i^k \middle| \boldsymbol{x}_i\right)}$$

Repeat until convergence! E-M provably finds a *local* maximum of the data likelihood.

E-M Animation



K-Means as "Hard" E-M

Want to group datapoints $\{x_1, x_2, \dots, x_N\}$ into K clusters

Algorithm:

1. Randomly initialize centroids $\{\mu_1, \mu_2, \dots \mu_K\}$
2. Assign each x_i to the closest cluster C_{j^*} according to

$$j^* = \mathrm{argmin}_j \left\| x_i - \mu_j \right\|_2$$

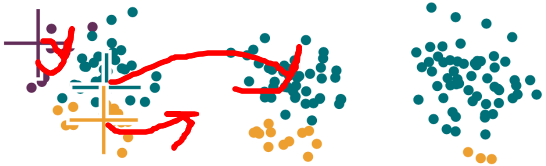
3. Update the centroid locations according to

$$\mu_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} x_i$$

4. Repeat steps 2 and 3 until convergence

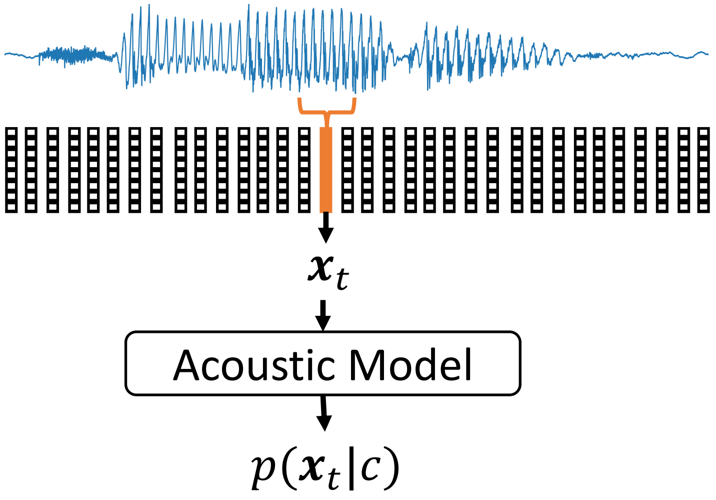
K-Means as “Hard” E-M

0



Multi-Class Classification with GMMs

- Back to our original picture, how do we model the acoustic state at time t with GMMs?
- Assume we have C classes, where each $c_j \in \{c_1, \ldots, c_C\}$ is an acoustic state (e.g. a phone)



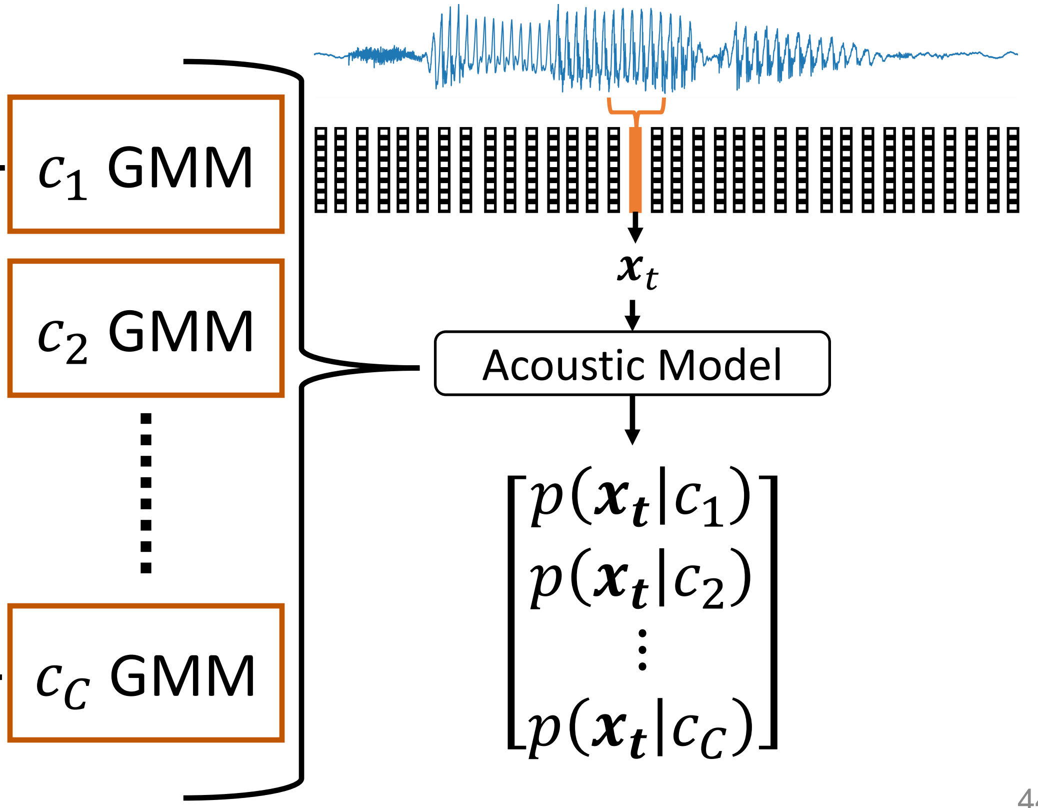
- Assume we know the ground-truth class label y_t for every frame. To create a model for class c_j, we can simply collect all the frames that satisfy $y_t = c_j$ and train a GMM with them.

Multi-Class Classification with GMMs

$$p(\boldsymbol{x}|c_1) = \sum_{k=1}^{K^{(1)}} w_k^{(1)} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k^{(1)}, \boldsymbol{\Sigma}_k^{(1)})$$

$$\vdots$$

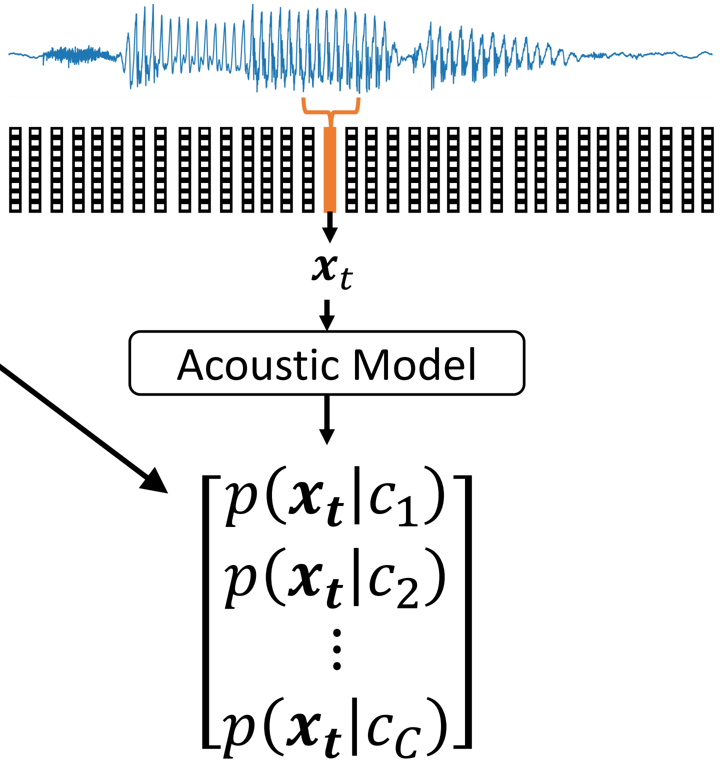
$$p(\boldsymbol{x}|c_C) = \sum_{k=1}^{K^{(C)}} w_k^{(C)} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k^{(C)}, \boldsymbol{\Sigma}_k^{(C)})$$



Multi-Class Classification with GMMs

We call these the acoustic state likelihoods

We could use Bayes' Rule to obtain the posterior $p(\boldsymbol{c}|\boldsymbol{x}_t)$ instead. But as we'll see later on, for an ASR system we generally use the likelihoods as-is



Remaining Questions

- How to integrate GMMs into an ASR system?
  - HMM lectures
- What about other acoustic models, like DNNs?:
  - Thursday + Next Week
- How do you get the frame-level labels in the first place?
  - HMM lectures
- Other training tricks
  - Adaptation/Discriminative Training lectures