

Speech Processing and Understanding

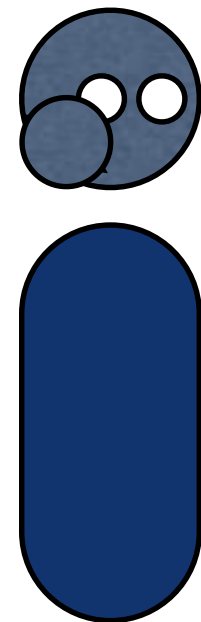
CSC401 Assignment 3

Agenda

- Background
 - Speech technology, in general
 - Acoustic phonetics
- Assignment 3
 - Speaker Recognition: Gaussian mixture models
 - Speech Recognition: Word-error rates with Levenshtein distance.

Applications of Speech Technology

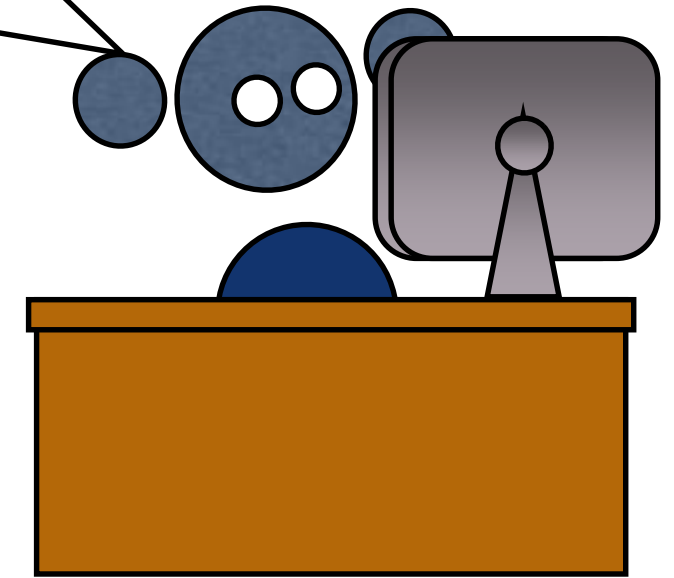
Telephony



Buy ticket...
AC490...
yes

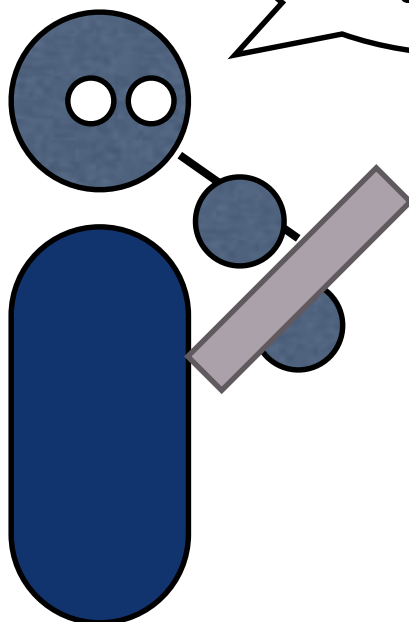
Dictation

My hands are
in the air.



Multimodality & HCI

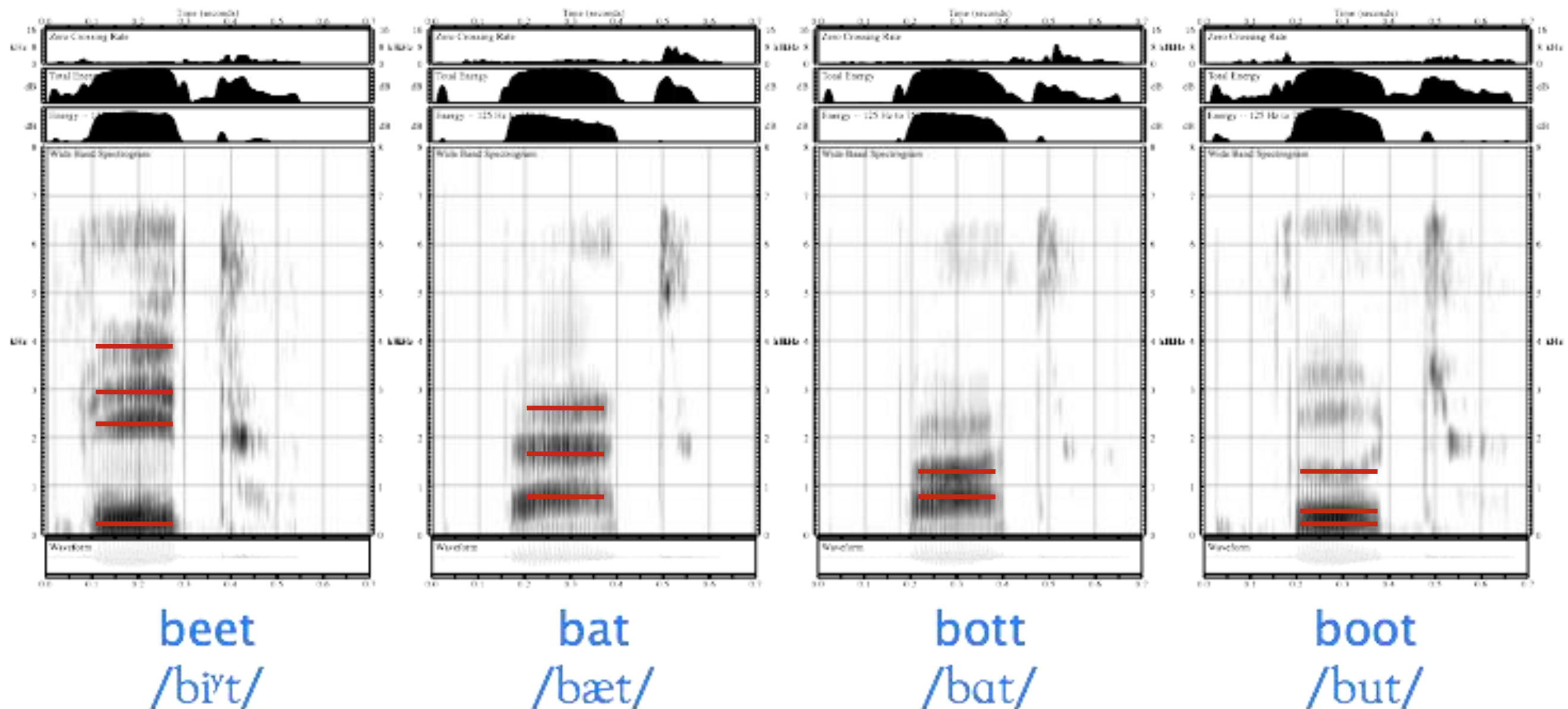
Put this
there.



Emerging...

- Data mining/indexing.
- Assistive technology.
- Conversation.

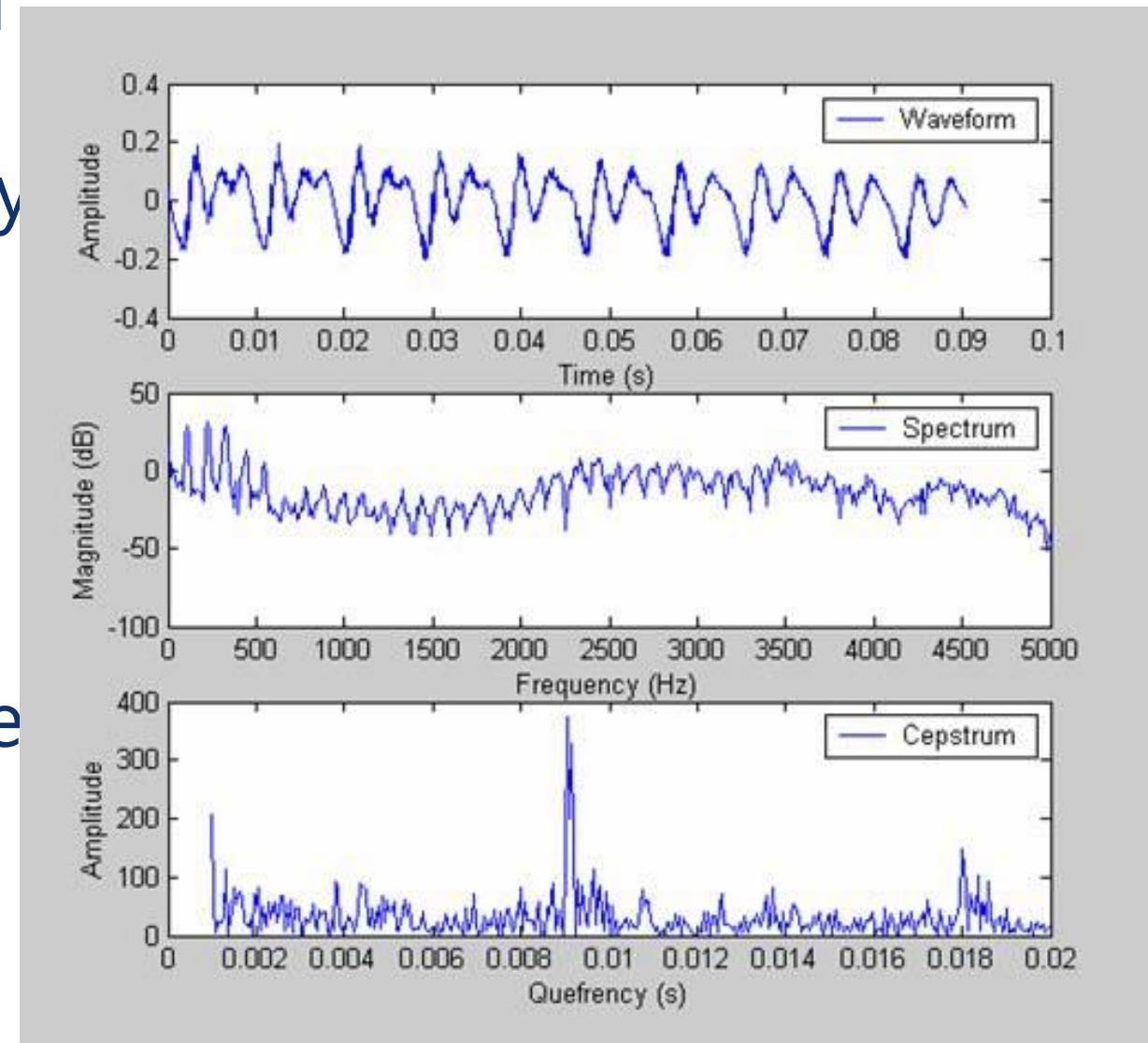
Formants in sonorants



- However, formants are insufficient features for use in speech recognition generally...

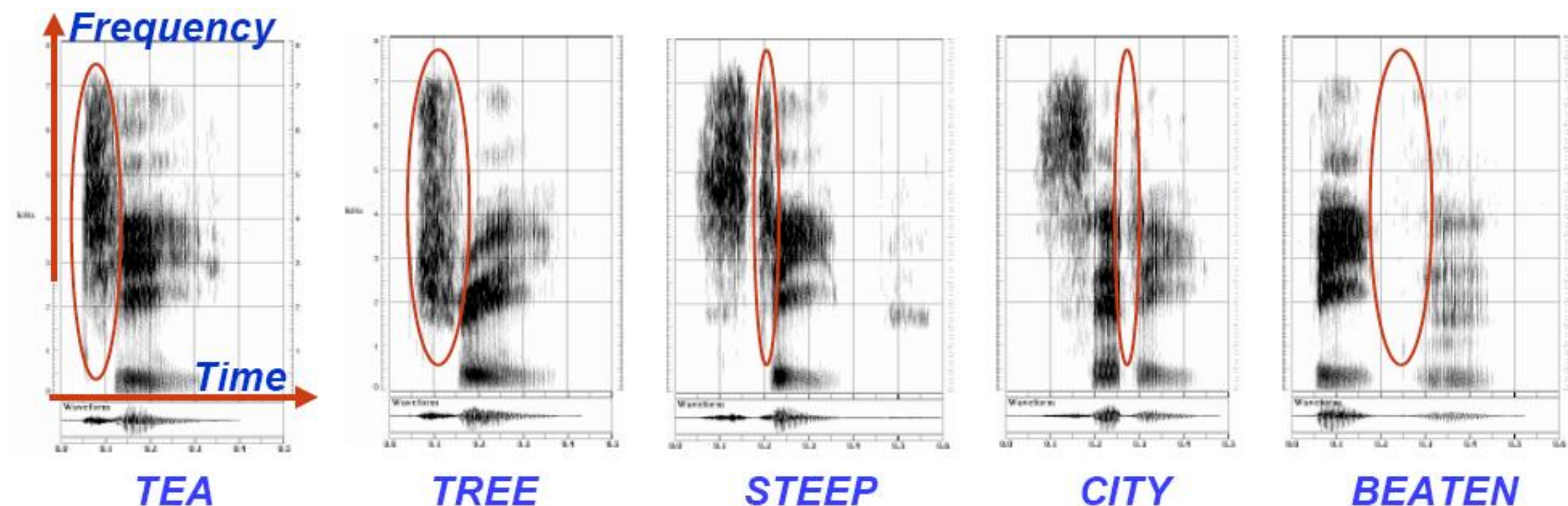
Mel-frequency cepstral coefficients

- In real speech data, the spectrogram is often transformed to a representation that more closely represents human auditory response and is more amenable to accurate classification.
- MFCCs are 'spectra of spectra'. They are the discrete cosine transform of the logarithms of the nonlinearly Mel-scaled powers of the Fourier transform of windows of the original waveform.



Challenges in speech data

- Co-articulation and dropped phonemes.
- (Intra-and-Inter-) Speaker variability.
- No word boundaries.
- Slurring, disfluency (e.g., 'um').
- Signal Noise.
- Highly dimensional.



Phonemes

- Words are formed by **phonemes** (aka 'phones'),
e.g., 'pod' = /p aa d/
- Words have different pronunciations. and in practice we can never be certain of which phones were uttered, nor their start/stop points.

	Sentence														
	Verb phrase														
Syntactic	Verb				Noun phrase										
					Det	Modifier				Noun (plu)					
						Noun		Noun							
Lexical	open				the		pod		bay		doors				
Phonemic	ow	p	ah	n	dh	ah	p	aa	d	b	ey	d	ao	r	z

Phonetic alphabets

- International Phonetic Association (IPA)
 - Can represent sounds in all languages
 - Contains non-ASCII characters
- ARPAbet
 - One of the earliest attempts at encoding English for early speech recognition.
- TIMIT/CMU
 - Very popular among modern databases for speech recognition.

Example phonetic alphabets

IPA	CMU	TIMIT	Example	IPA symbol name
[ɑ]	AA	aa	f <u>ath</u> er, h <u>o</u> t	script a
[æ]	AE	ae	h <u>a</u> d	digraph
[ə]	AH0	ax	sof <u>a</u>	schwa (common in unstressed syllables)
[ʌ]	AH1	ah	b <u>u</u> t	turned v
[ɔ:]	AO	ao	ca <u>u</u> ght	open o – Note, many speakers of Am. Eng. do not distinguish between [ɔ:] and [ɑ]. If your “caught” and “cot” sound the same, you do not.
[ɛ]	EH	eh	h <u>e</u> ad	epsilon
[ɪ]	IH	ih	h <u>i</u> d	small capital I
[i:]	IY	iy	h <u>ee</u> d	lowercase i
[ʊ]	UH	uh	h <u>oo</u> d, b <u>oo</u> k	upsilon
[u:]	UW	uw	b <u>oo</u> t	lowercase u
[aɪ]	AY	ay	h <u>i</u> de	
[aʊ]	AW	aw	h <u>ow</u>	
[eɪ]	EY	ey	to <u>d</u> ay	
[oʊ]	OW	ow	h <u>oe</u> d	
[ɔɪ]	OY	oy	jo <u>y</u> , aho <u>y</u>	
[ər]	ER0	axr	h <u>e</u> rself	schwar (schwa changed by following r)
[ɜr]	ER1	er	b <u>ir</u> d	reverse epsilon right hook

IPA	CMU	TIMIT	Example	IPA symbol name
[ŋ]	NG	ng	si <u>ng</u> so <u>ng</u>	eng or angma
[ʃ]	SH	<u>sh</u>	<u>sh</u> ee <u>t</u> , wi <u>sh</u>	esh or long s
[tʃ]	CH	<u>ch</u>	<u>ch</u> ee <u>s</u> e	
[j]	Y	y	y <u>e</u> llow	lowercase j
[ʒ]	ZJ	zh	vi <u>si</u> on	long z or yogh
[dʒ]	JH	jh	ju <u>d</u> ge	
[ð]	DH	dh	<u>th</u> ee, <u>th</u> is	eth

- The other consonants are transcribed as you would expect
 - i.e., p, b, m, t, d, n, k, g, s, z, f, v, w, h

Agenda

- Background
 - Speech technology, in general
 - Acoustic phonetics
- Assignment 3
 - Speaker Recognition: Gaussian mixture models
 - Speech Recognition: Word-error rates with Levenshtein distance.

Assignment 3

- Two parts:
 - Speaker identification: Determine which of 30 speakers an unknown test sample of speech comes from, given Gaussian mixture models you will train for each speaker.
 - Speech recognition: Compute word-error rates for speech recognition systems using Levenshtein distance.

Speaker Data

- 32 speakers (e.g., S-3C, S-5A).
- Each speaker has up to 12 training utterances.
 - e.g., `/u/csc401/A3/data/S-3C/0.wav`
- Each utterance has 3 files:
 - `*.wav` : The original wave file.
 - `*.mfcc.npy` : The MFCC features in NumPy format
 - `*.txt` : Sentence-level transcription.

Speaker Data (cont.)

- All you need to know: A speech utterance is an $N \times d$ matrix
 - Each row represents the features of a d -dimensional point in time.
 - There are N rows in a sequence of N frames.
 - The data is in numpy arrays * `.mfcc.npy`
 - To read the files: `np.load('1.mfcc.npy')`

		data dimension			
		1	2		d
time ↓	frames	1	$X_1[1]$	$X_1[2]$	$X_1[d]$
		2	$X_2[1]$	$X_2[2]$	$X_2[d]$
		
		N	$X_N[1]$	$X_N[2]$	$X_N[d]$

Speaker Data (cont.)

- You are given human transcriptions in transcripts.txt
- You are also given Kaldi and Google transcriptions in `transcripts.*.txt`.
- Ignore any symbols that are not words.

Agenda

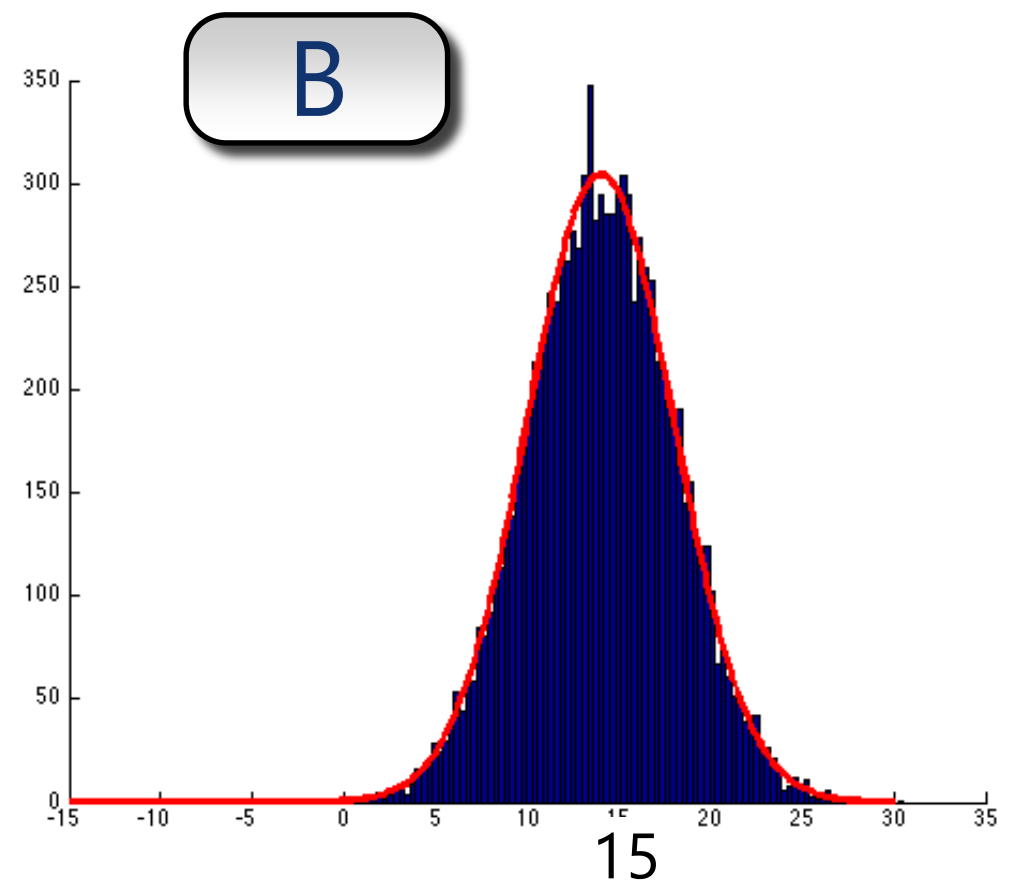
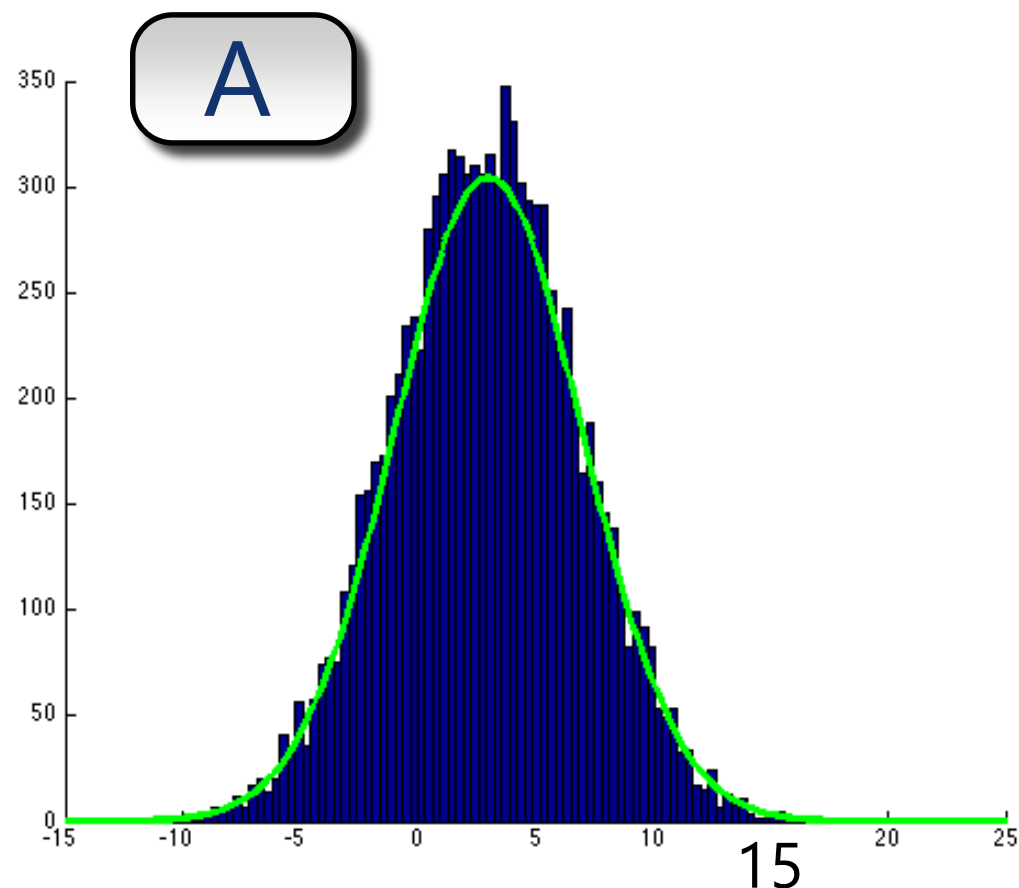
- Background
 - Speech technology, in general
 - Acoustic phonetics
- Assignment 3
 - Speaker Recognition: Gaussian mixture models
 - Speech Recognition: Word-error rates with Levenshtein distance.

Speaker Recognition

- The data is randomly split into training and testing utterances. We don't know which speaker produced which test utterance.
- Every speaker occupies a characteristic part of the acoustic space.
- We want to learn a probability distribution for each speaker that describes their acoustic behaviour.
 - Use those distributions to identify the speaker-dependent features of some unknown sample of speech data.

Some background: fitting to data

- Given a set of observations X of some random variable, we wish to know how X was generated.
- Here, we assume that the data was sampled from a Gaussian Distribution (validated by data).
- Given a new data point ($x=15$), It is more likely that x was generated by B.



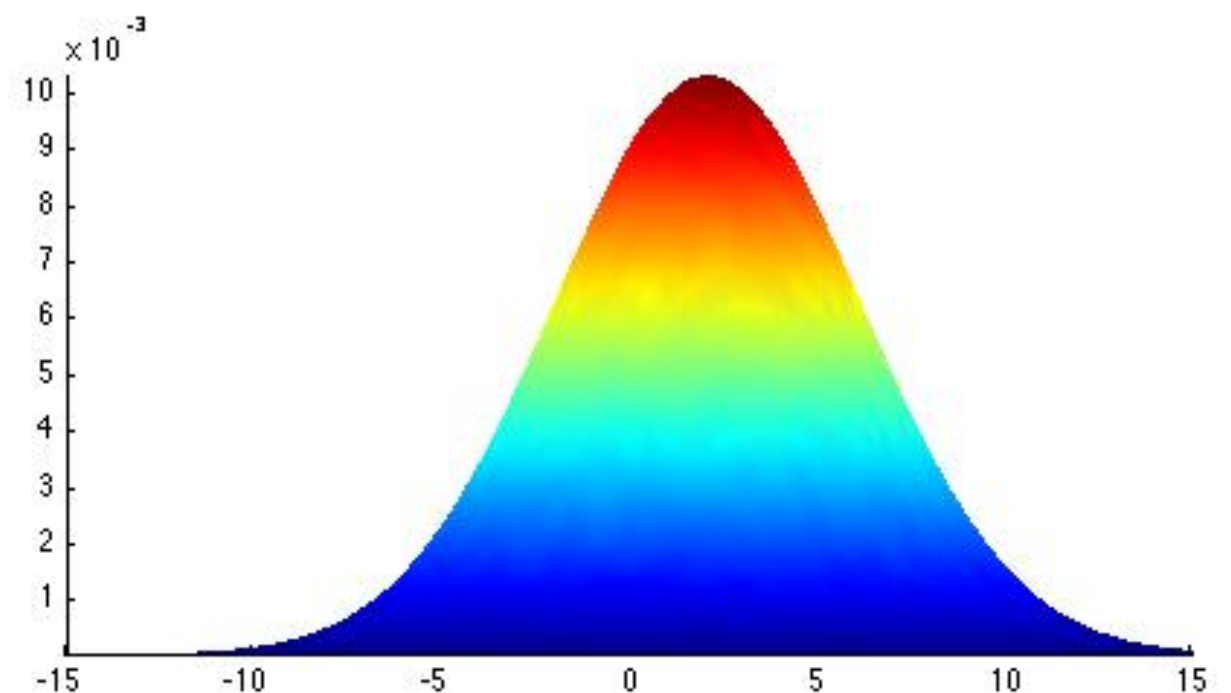
Finding parameters: 1D Gaussians

- Often called *Normal* distributions

$$p(x) = \frac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi}\sigma}$$

$$N(\mu, \sigma)$$

$$\mu = E(x) = \int xp(x)dx$$



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

- The parameters we can adjust to fit the data are μ and σ^2 :

$$\theta = \langle \mu, \sigma \rangle$$

Maximum likelihood estimation

- Given data: $X = \{x_1, x_2, \dots, x_n\}$
- and Parameter set: θ
- Maximum likelihood attempts to find the parameter set that maximizes the likelihood of the data.

$$L(X, \theta) = p(X \mid \theta) = p(x_1, x_2, \dots, x_n \mid \theta) = \prod_{i=1}^n p(x_i \mid \theta)$$

- The likelihood function $L(X, \theta)$ provides a surface over all possible parameterizations. In order to find the Maximum Likelihood, we set the derivative to zero:

$$\frac{\partial}{\partial \theta} L(X, \theta) = 0$$

MLE - 1D Gaussian

- Estimate $\hat{\mu}$

$$L(X, \mu) = p(X \mid \mu) = \prod_{i=1}^n p(x_i \mid \mu) = \prod_{i=1}^n \frac{\exp\left(-\frac{(x_i - \mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi}\sigma}$$

$$\log L(X, \mu) = -\frac{\sum_i (x_i - \mu)^2}{2\sigma^2} - n \log \sqrt{2\pi}\sigma$$

$$\frac{\partial}{\partial \mu} \log L(X, \mu) = \frac{\sum_i (x_i - \mu)}{\sigma^2} = 0$$

$$\hat{\mu} = \frac{\sum_i x_i}{n}$$

- A similar approach gives the MLE estimate of $\hat{\sigma}^2$:

$$\hat{\sigma}^2 = \frac{\sum_i (x_i - \hat{\mu})^2}{n}$$

Multidimensional Gaussians

- When your data is d-dimensional, the input variable is

$$\vec{x} = \langle x[1], x[2], \dots, x[d] \rangle$$

the mean vector is

$$\vec{\mu} = E(\vec{x}) = \langle \mu[1], \mu[2], \dots, \mu[d] \rangle$$

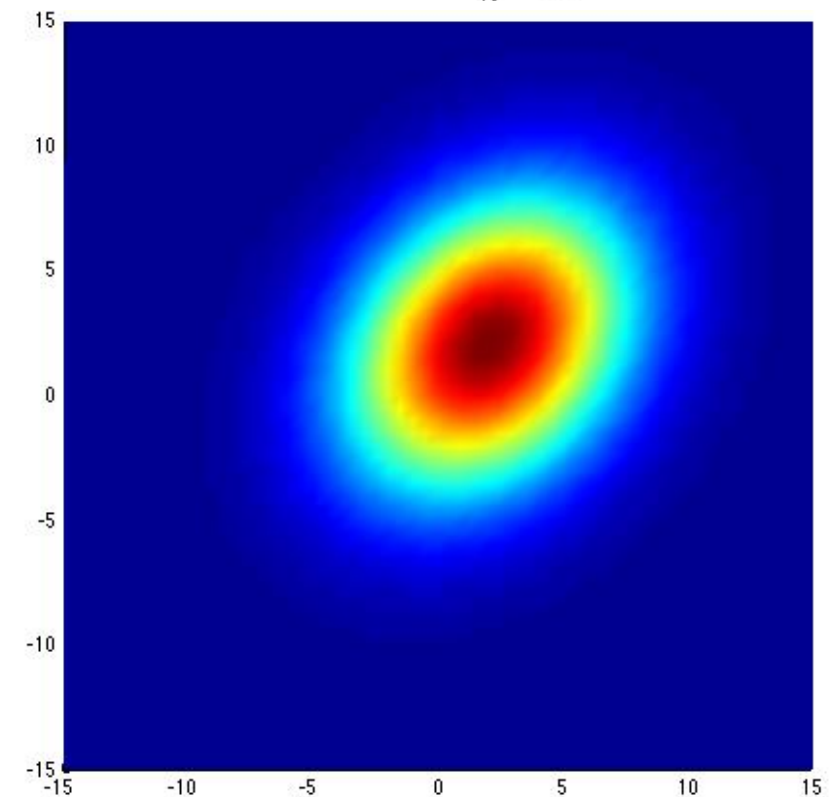
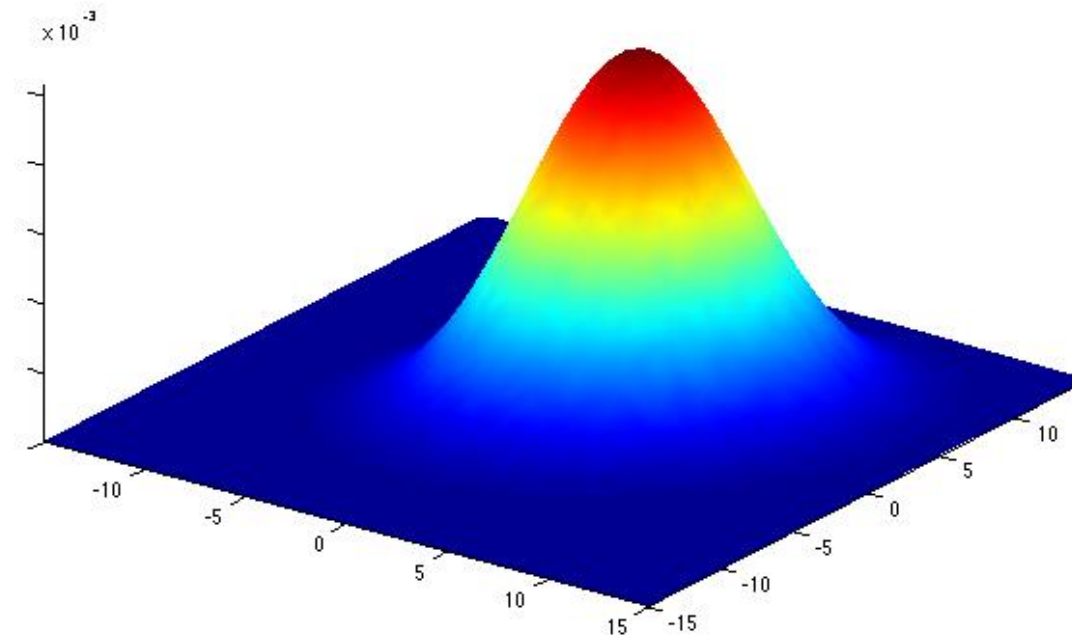
the covariance matrix is

$$\Sigma = E((\vec{x} - \vec{\mu})(\vec{x} - \vec{\mu})^T)$$

with $\Sigma[i, j] = E(x[i]x[j]) - \mu[i]\mu[j]$

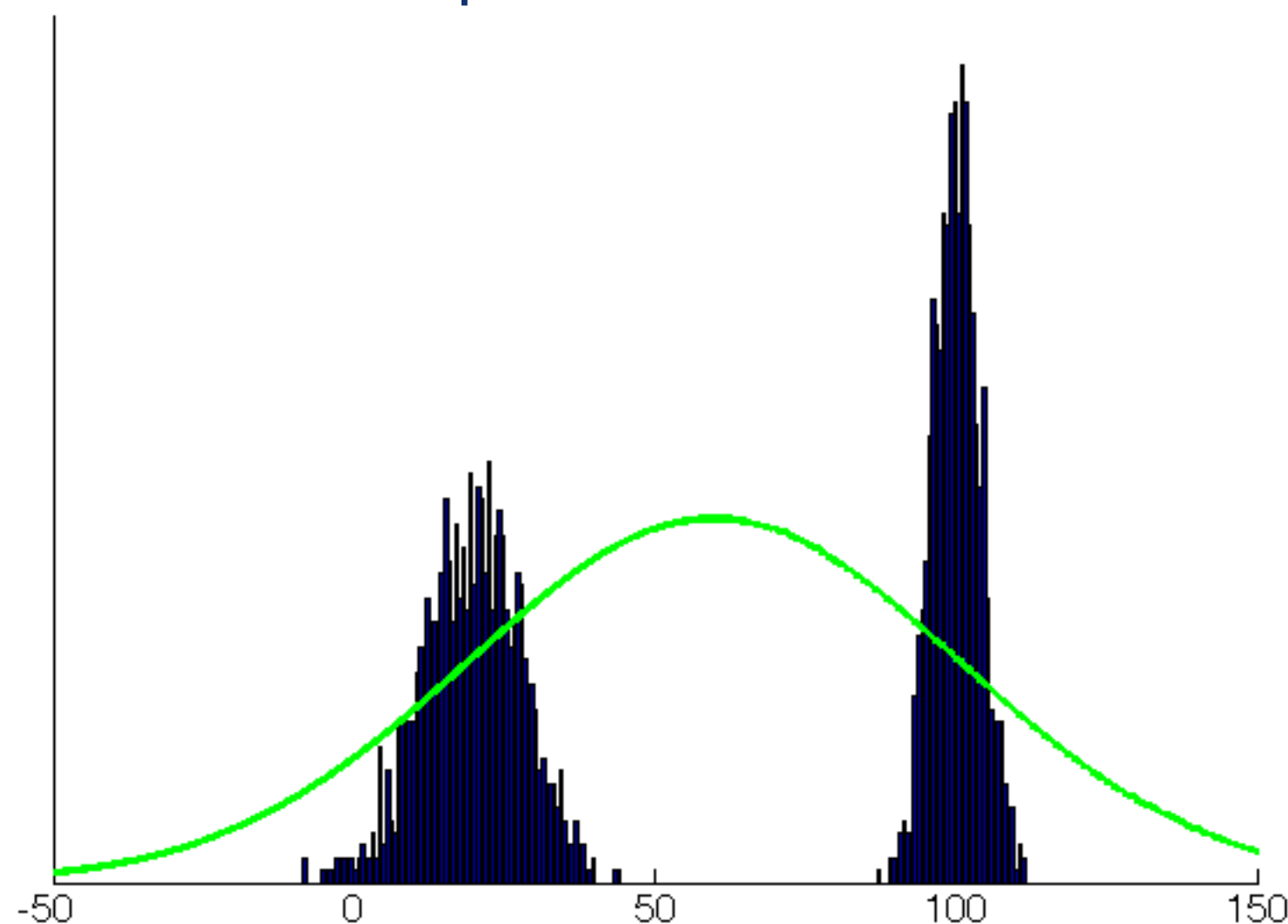
and

$$p(\vec{x}) = \frac{\exp\left(-\frac{(\vec{x} - \vec{\mu})^T \Sigma^{-1} (\vec{x} - \vec{\mu})}{2}\right)}{(2\pi)^{d/2} |\Sigma|^{1/2}}$$



Non-Gaussian data

- Our speaker data does not behave unimodally.
 - i.e., we can't use just 1 Gaussian per speaker.
- E.g., observations below occur mostly bimodally, so fitting 1 Gaussian would not be representative.

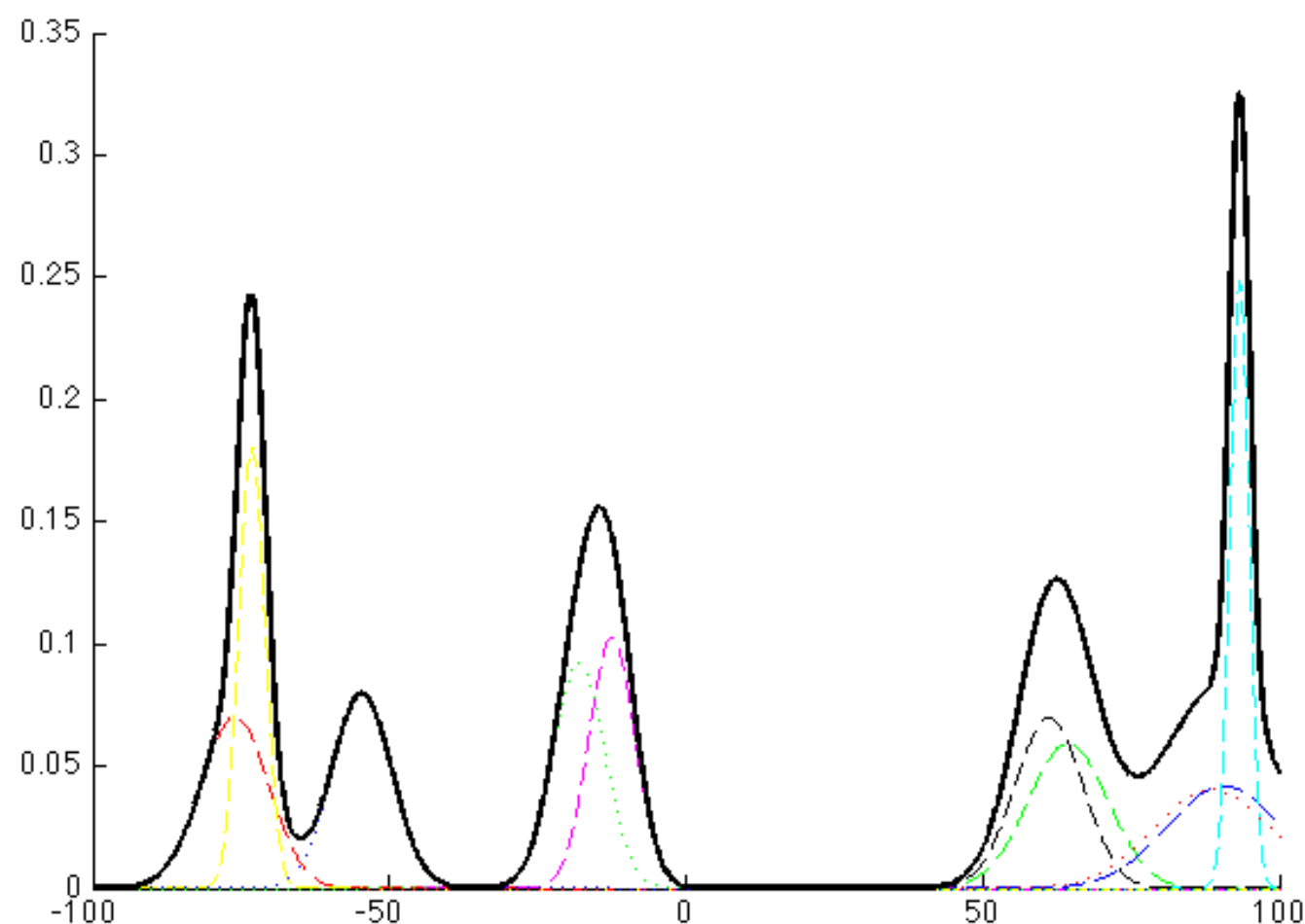


Gaussian mixtures

- Gaussian mixtures are a weighted linear combination of M component gaussians.

$$\langle \Gamma_1, \dots, \Gamma_M \rangle$$

$$p(\vec{x}) = \sum_{j=1}^M p(\Gamma_j) p(\vec{x} \mid \Gamma_j)$$



MLE for Gaussian mixtures

- For notational convenience, $\omega_m = p(\Gamma_m)$, $b_m(\vec{x}_t) = p(\vec{x}_t | \Gamma_m)$

- So

$$p_{\Theta}(\vec{x}_t) = \sum_{m=1}^M \omega_m b_m(\vec{x}_t), \quad \Theta = \langle \omega_m, \vec{\mu}_m, \Sigma_m \rangle, \quad m = 1, \dots, M$$

$$b_m(\vec{x}_t) = \frac{\exp\left(-\frac{1}{2} \sum_{i=1}^d \frac{(x_t[i] - \mu_m[i])^2}{\sigma_m^2[i]}\right)}{(2\pi)^{d/2} \left(\prod_{i=1}^d \sigma_m^2[i]\right)^{1/2}}$$

- To find $\hat{\Theta}$, we solve $\nabla_{\Theta} \log L(X, \Theta) = 0$ where

$$\log L(X, \Theta) = \sum_{t=1}^N \log p_{\Theta}(\vec{x}_t) = \sum_{t=1}^N \log \left(\sum_{m=1}^M \omega_m b_m(\vec{x}_t) \right)$$

...see Appendix for more

MLE for Gaussian mixtures (pt. 2)

- Given
$$\frac{\partial \log L(X, \Theta)}{\partial \mu_m[n]} = \sum_{t=1}^N \frac{1}{p_{\Theta}(\vec{x}_t)} \left[\frac{\partial}{\partial \mu_m[n]} \omega_m b_m(\vec{x}_t) \right]$$
- Since
$$\frac{\partial}{\partial \mu_m[n]} b_m(\vec{x}_t) = b_m(\vec{x}_t) \frac{x_t[n] - \mu_m[n]}{\sigma_m^2[n]}$$
- We obtain $\hat{\mu}_m[n]$ by solving for $\mu_m[n]$ in :

$$\frac{\partial \log L(X, \Theta)}{\partial \mu_m[n]} = \sum_{t=1}^N \frac{\omega_m}{p_{\Theta}(\vec{x}_t)} b_m(\vec{x}_t) \frac{x_t[n] - \mu_m[n]}{\sigma_m^2[n]} = 0$$

and:

$$b_m(\vec{x}_t) = p(\vec{x}_t \mid \Gamma_m)$$
$$p(\Gamma_m \mid \vec{x}_t, \Theta) = \frac{\omega_m}{p_{\Theta}(\vec{x}_t)} b_m(\vec{x}_t)$$

$$\hat{\mu}_m[n] = \frac{\sum_t p(\Gamma_m \mid \vec{x}_t, \Theta) x_t[n]}{\sum_t p(\Gamma_m \mid \vec{x}_t, \Theta)}$$

Recipe for GMM ML estimation

- Do the following for each speaker individually. Use all the frames available in their respective **Training** directories

1.	<u>Initialize</u> : Guess $\Theta = \langle \omega_m, \vec{\mu}_m, \Sigma_m \rangle$, $m = 1, \dots, M$ with M random vectors in the data, or by performing M-means clustering.	
2.	<u>Compute likelihood</u> : Compute $b_m(\vec{x}_t)$ and $P(\Gamma_m \vec{x}_t, \Theta)$	
3.	<u>Update parameters</u> :	$\hat{\omega}_m = \frac{1}{T} \sum_{t=1}^T p(\Gamma_m \vec{x}_t, \Theta)$
	$\hat{\vec{\sigma}}_m^2 = \frac{\sum_t p(\Gamma_m \vec{x}_t, \Theta) \vec{x}_t^2}{\sum_t p(\Gamma_m \vec{x}_t, \Theta)} - \hat{\vec{\mu}}_m^2$	$\hat{\vec{\mu}}_m = \frac{\sum_t p(\Gamma_m \vec{x}_t, \Theta) \vec{x}_t}{\sum_t p(\Gamma_m \vec{x}_t, \Theta)}$
		$\log p(X \hat{\Theta}_{i+1}) - \log p(X \hat{\Theta}_i) < \epsilon$
	Repeat 2&3 until converges	

Cheat sheet

$$b_m(\vec{x}_t) = p(\vec{x}_t \mid \Gamma_m)$$

$$b_m(\vec{x}_t) = \frac{\exp\left(-\frac{1}{2} \sum_{i=1}^d \frac{(x_t[i] - \mu_m[i])^2}{\sigma_m^2[i]}\right)}{(2\pi)^{d/2} \left(\prod_{i=1}^d \sigma_m^2[i]\right)^{1/2}}$$

Probability of observing x_t in the m^{th} Gaussian

$$\omega_m = p(\Gamma_m)$$

Prior probability of the m^{th} Gaussian

$$p(\Gamma_m \mid \vec{x}_t, \Theta) = \frac{\omega_m}{p_{\Theta}(\vec{x}_t)} b_m(\vec{x}_t)$$

Probability of the m^{th} Gaussian, given x_t

$$p_{\Theta}(\vec{x}_t) = \sum_{m=1}^M \omega_m b_m(\vec{x}_t)$$

Probability of x_t in the GMM

Initializing theta

$$\Theta = \langle \omega_1, \mu_1, \Sigma_1, \omega_2, \mu_2, \Sigma_2, \dots, \omega_M, \mu_M, \Sigma_M \rangle$$

- Initialize each μ_m to a random vector from the data.
- Initialize Σ_m to a 'random' diagonal matrix (or identity matrix).
- Initialize ω_m randomly, with these constraints:

$$0 \leq \omega_m \leq 1$$

$$\sum_m \omega_m = 1$$

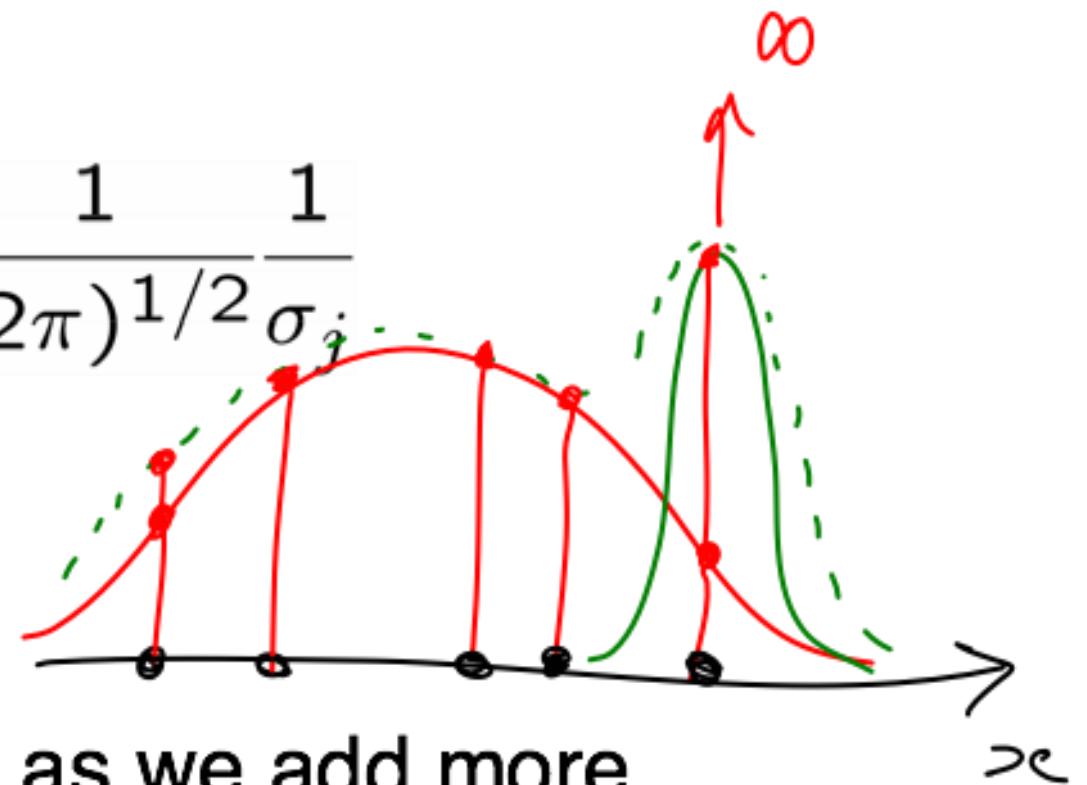
- A good choice would be to set ω_m to $1/M$

Over-fitting in Gaussian Mixture Models

- Singularities in likelihood function when a component ‘collapses’ onto a data point:

$$\mathcal{N}(\mathbf{x}_n | \mathbf{x}_n, \sigma_j^2 \mathbf{I}) = \frac{1}{(2\pi)^{1/2}} \frac{1}{\sigma_j}$$

then consider $\sigma_j \rightarrow 0$



- Likelihood function gets larger as we add more components (and hence parameters) to the model
 - not clear how to choose the number K of components

Solutions:

- Ensure that the variances don't get too small.
- Bayesian GMMs

Your Task

- For each speaker, train a GMM, using the EM algorithm, assuming diagonal covariance.
- Identify the speaker of each test utterance.
- Experiment with the number of mixture elements in the models, the improvement threshold, number of possible speakers, etc.
- Comment on the results

Practical tips for MLE of GMMs

- We assume diagonal covariance matrices. This reduces the number of parameters and can be sufficient in practice given enough components.
- Numerical Stability: Compute likelihoods in the log domain (especially when calculating the likelihood of a sequence of frames).

$$\log b_m(\vec{x}_t) = - \sum_{n=1}^d \frac{(\vec{x}_t[n] - \vec{\mu}_m[n])^2}{2\sigma_m^2[n]} - \frac{d}{2} \log 2\pi - \frac{1}{2} \log \prod_{n=1}^d \sigma_m^2[n]$$

- Here, \vec{x}_t , $\vec{\mu}_m$ and σ_m^2 are d-dimensional vectors.

Practical tips (pt. 2)

- Efficiency: Pre-compute terms not dependent on \vec{x}_t

$$\begin{aligned} \log b_m(\vec{x}_t) = & - \sum_{n=1}^d \left(\frac{1}{2} \vec{x}_t[n]^2 \sigma_m^{-2}[n] - \mu_m[n] \vec{x}_t[n] \sigma_m^{-2}[n] \right) \\ & - \left(\sum_{n=1}^d \frac{\mu_m[n]^2}{2\sigma_m^2[n]} + \frac{d}{2} \log 2\pi + \frac{1}{2} \log \prod_{n=1}^d \sigma_m^2[n] \right) \end{aligned}$$

Agenda

- Background
 - Speech technology, in general
 - Acoustic phonetics
- Assignment 3
 - Speaker Recognition: Gaussian mixture models
 - Speech Recognition: Word-error rates with Levenshtein distance.

Word-error rates

- If somebody said
REF: how to recognize speech
but an ASR system heard
HYP: how to wreck a nice beach
how do we measure the error that occurred?
- One measure is $\frac{\text{\#CorrectWords}}{\text{\#HypothesisWords}}$
e.g., 2/6 above
- Another measure is $\frac{S+I+D}{\text{\#ReferenceWords}}$
 - S: # Substitution errors (one word for another)
 - I: # Insertion errors (extra words)
 - D: # Deletion errors (words that are missing).

Computing Levenshtein Distance

- In the example
REF: how to recognize speech.
HYP: how to wreck a nice beach
How do we count each of S, I, and D?
- If “wreck” is a substitution error, what about “a” and “nice”?

Computing Levenshtein Distance

- In the example
REF: how to recognize speech.
HYP: how to wreck a nice beach
How do we count each of S, I, and D?
If “wreck” is a substitution error, what about “a” and “nice”?

- Levenshtein distance:
Initialize $R[0,0] = 0$, and $R[i,j] = \infty$ for all $i=0$ or $j=0$
for $i=1..n$ (#ReferenceWords)
 for $j=1..m$ (#Hypothesis words)
 $R[i,j] = \min(\begin{matrix} R[i-1,j] + 1 & \text{(deletion)} \\ R[i-1,j-1] & \text{(only if words match)} \\ R[i-1,j-1] + 1 & \text{(only if words differ)} \\ R[i,j-1] + 1 & \text{(insertion)} \end{matrix})$

Return $100 * R(n,m) / n$

Levenshtein example

		how	to	wreck	a	nice	beach
	0	∞	∞	∞	∞	∞	∞
how	∞	0	1	2	3	4	5
to	∞						
recognize	∞						
speech	∞						

Levenshtein example

		how	to	wreck	a	nice	beach
	0	∞	∞	∞	∞	∞	∞
how	∞	0	1	2	3	4	5
to	∞	1	0	1	2	3	4
recognize	∞						
speech	∞						

Levenshtein example

		how	to	wreck	a	nice	beach
	0	∞	∞	∞	∞	∞	∞
how	∞	0	1	2	3	4	5
to	∞	1	0	1	2	3	4
recognize	∞	2	1	1	2	3	4
speech	∞						

Levenshtein example

		how	to	wreck	a	nice	beach
	0	∞	∞	∞	∞	∞	∞
how	∞	0	1	2	3	4	5
to	∞	1	0	1	2	3	4
recognize	∞	2	1	1	2	3	4
speech	∞	3	2	2	2	3	4

Word-error rate is $4/4 = 100\%$

2 substitutions, 2 insertions

Appendices

Multidimensional Gaussians, pt. 2

- If the i^{th} and j^{th} dimensions are statistically independent,

$$E(x[i]x[j]) = E(x[i])E(x[j])$$

and

$$\Sigma[i, j] = 0$$

- If all dimensions are statistically independent, $\Sigma[i, j] = 0, \forall i \neq j$ and the covariance matrix becomes diagonal, which means

$$p(\vec{x}) = \prod_{i=1}^d p(x[i])$$

where

$$p(x[i]) \sim N(\mu[i], \Sigma[i, i])$$
$$\Sigma[i, i] = \sigma^2[i]$$

MLE example - dD Gaussians

- The MLE estimates for parameters $\Theta = \langle \theta_1, \theta_2, \dots, \theta_d \rangle$ given i.i.d. training data $X = \langle \vec{x}_1, \dots, \vec{x}_n \rangle$ are obtained by maximizing the joint likelihood

$$L(X, \Theta) = p(X \mid \Theta) = p(\vec{x}_1, \dots, \vec{x}_n \mid \Theta) = \prod_{i=1}^n p(\vec{x}_i \mid \Theta)$$

- To do so, we solve $\nabla_{\Theta} L(X, \Theta) = 0$ where

$$\nabla_{\Theta} = \left\langle \frac{\partial}{\partial \theta_1}, \dots, \frac{\partial}{\partial \theta_d} \right\rangle$$

- Giving

$$\hat{\vec{\mu}} = \frac{\sum_{t=1}^n \vec{x}_t}{n} \quad \hat{\Sigma} = \frac{\sum_{t=1}^n \left(\vec{x}_t - \hat{\vec{\mu}} \right) \left(\vec{x}_t - \hat{\vec{\mu}} \right)^T}{n}$$

MLE for Gaussian mixtures (pt1.5)

- Given $\log L(X, \Theta) = \sum_{t=1}^N \log p_{\Theta}(\vec{x}_t)$ and $p_{\Theta}(\vec{x}_t) = \sum_{m=1}^M \omega_m b_m(\vec{x}_t)$
- Obtain an ML estimate, $\hat{\mu}_m$, of the mean vector by maximizing $\log L(X, \mu_m)$ w.r.t. $\mu_m[n]$

$$\frac{\partial \log L(X, \Theta)}{\partial \mu_m[n]} = \sum_{t=1}^N \frac{\partial}{\partial \mu_m[n]} \log p_{\Theta}(\vec{x}_t) = \sum_{t=1}^N \frac{1}{p_{\Theta}(\vec{x}_t)} \left[\frac{\partial}{\partial \mu_m[n]} \omega_m b_m(\vec{x}_t) \right]$$

- Why?

d of sum = sum of d

d rule for log_e

d wrt μ_m is 0 for all other mixtures in the sum in $p_{\Theta}(\vec{x}_t)$