Speech processing
Automatic Speech Recognition (ASR)
Text-To-Speech (TTS)

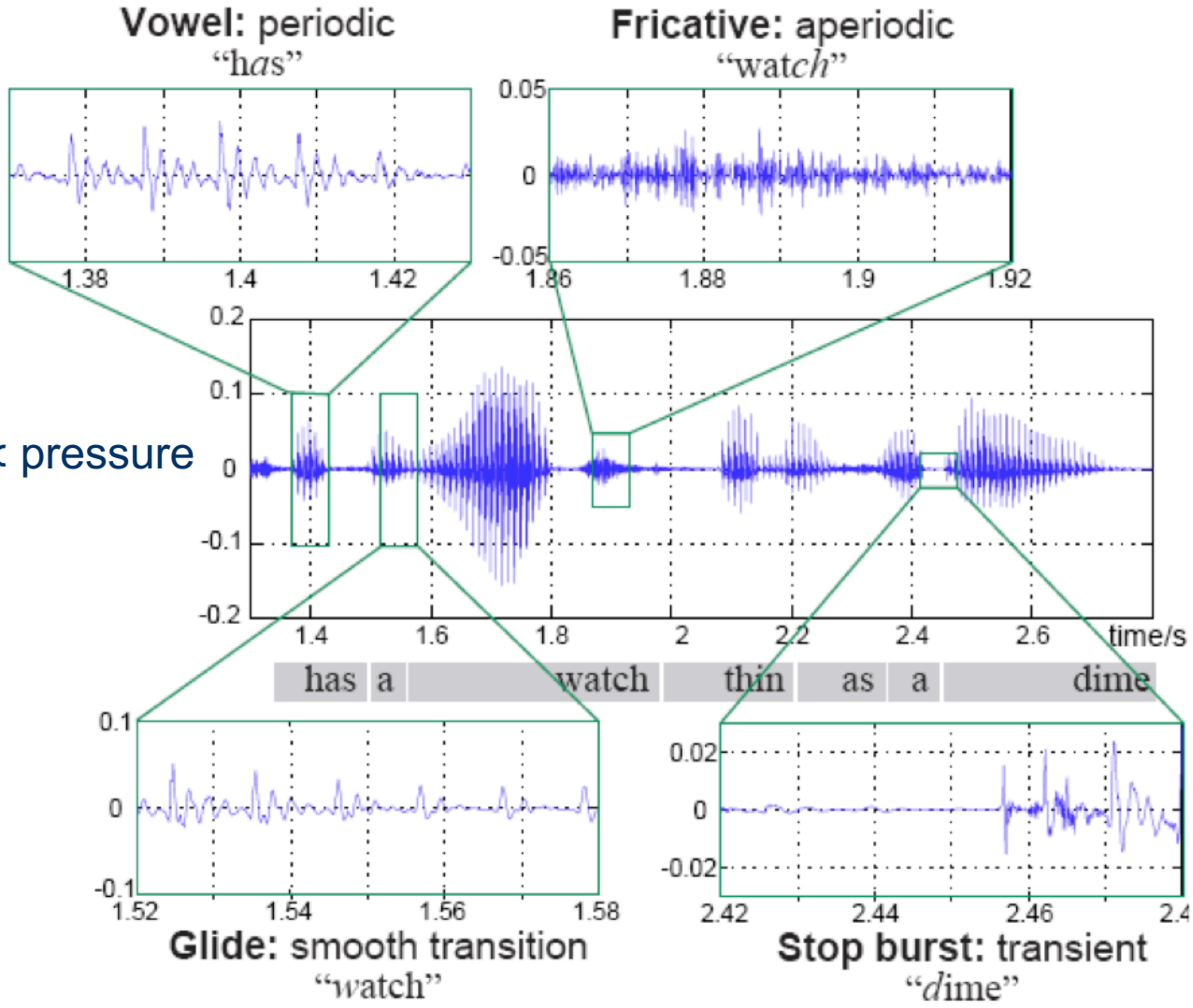
Ing. Roberto Tedesco, PhD
roberto.tedesco@polimi.it

NLP – AA 20-21

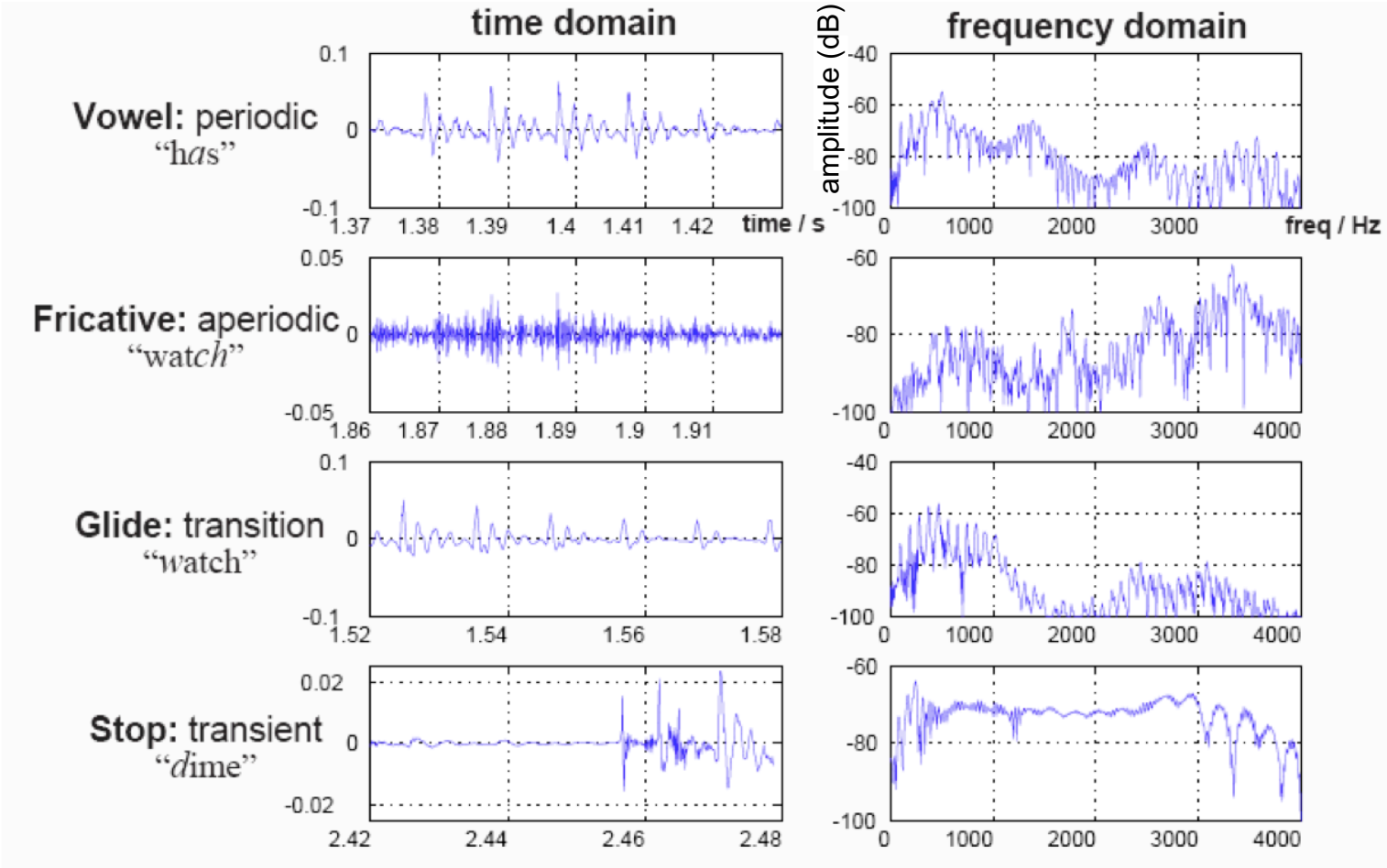
INTRO AND ASR

Time domain

intensity ∝ pressure



Frequency domain



$$\text{dB} = 20\log_{10}(\text{amplitude}) = 10\log_{10}(\text{power})$$

https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html

Pre-emphasis

- Pre-emphasis filter on the sampled signal $s(t)$ to amplify the high frequencies
- A pre-emphasis filter is useful in several ways:
  - Balances the frequency spectrum since high frequencies usually have smaller amplitudes compared to lower frequencies
  - Avoids numerical problems during the Fourier transform
  - May also improve the Signal-to-Noise Ratio (SNR)
- The pre-emphasis filter can be applied to the signal using a first order filter:

$$x(t)=s(t)-\alpha \cdot s(t-1)$$

where typical values for α are 0.95 or 0.97

Short time Fourier transform

- The vocal signal changes over time
  - Formants are only valid for a brief time interval
- STFT: to determine sinusoidal amplitude and phase of local sections of a signal as it changes over time
  - Signal divided into M chunks $[x_0[0{:}L-1],\ldots, x_m[0{:}L-1]$, $x_{M\text{-}1}[0{:}L-1]]$ each one composed of L samples
  - Each chunk is multiplied by a window function $w[0{:}L-1]$
  - STFT_m is a L-sized vector containing the Fourier transform of the m-th chunk (each k-th component of STFT_m is a complex number):

$$\mathrm{STFT}_m[k]\{x_m\} = \sum_{n=0}^{L-1} x_m[n]\cdot w[n]\cdot e^{-j\frac{2\cdot\pi}{L}\cdot k\cdot n}$$

STFT for the m-th chunk

$0 \le m \le M-1$

$0 \le k \le L-1$

$0 \le n \le L-1$

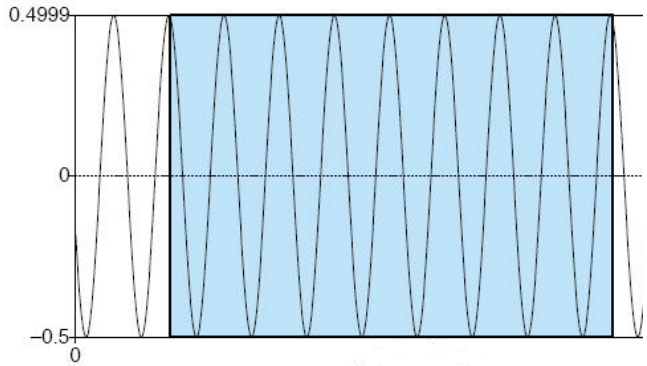
Hamming

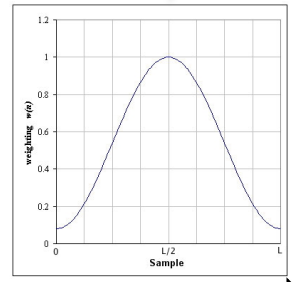
- The most used window function

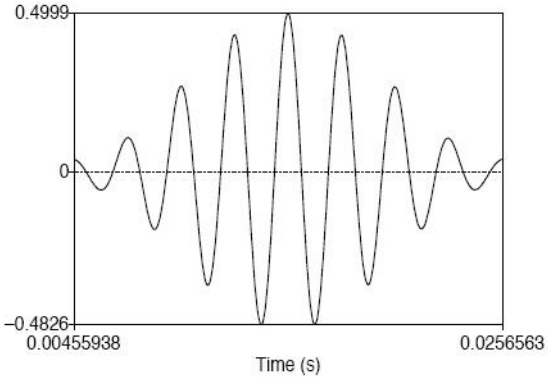
$$w[n] = 0.54 - 0.46 \cdot \cos\left(\frac{2 \cdot \pi \cdot n}{L}\right); \quad 0 \le n \le L - 1$$

- Reduces border effects
- E.g.: effect on a sinus

Applying Hamming window

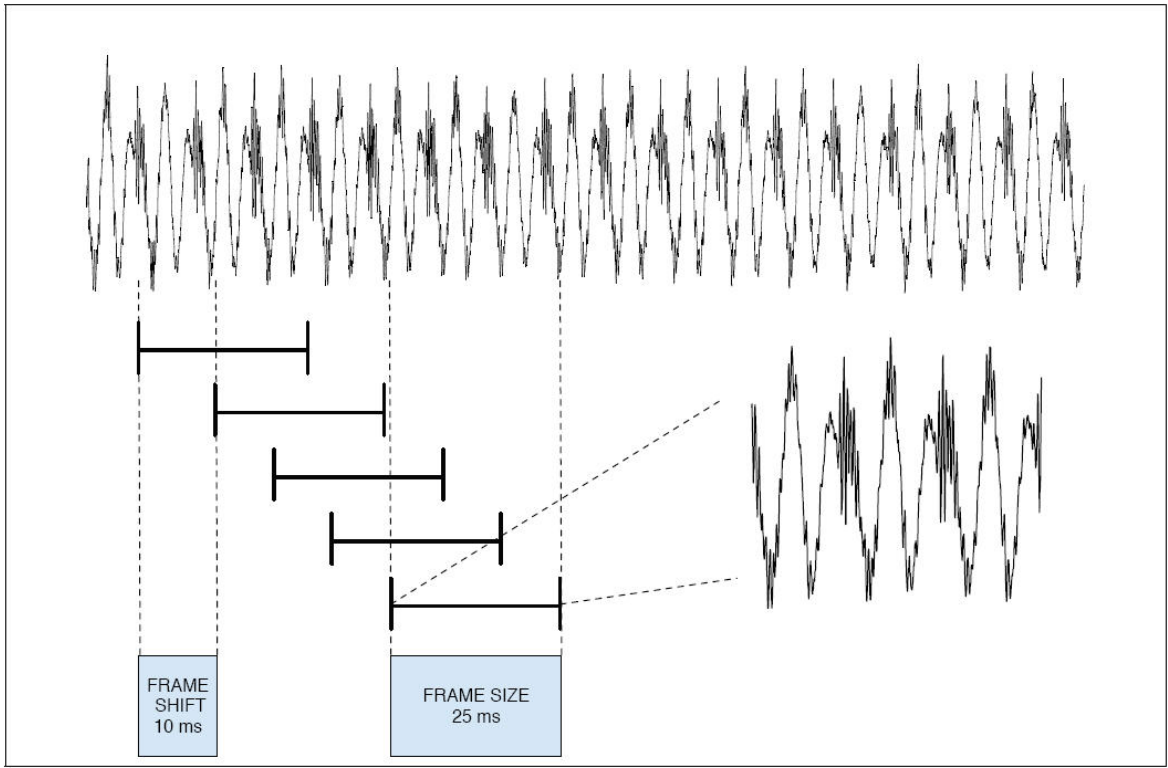




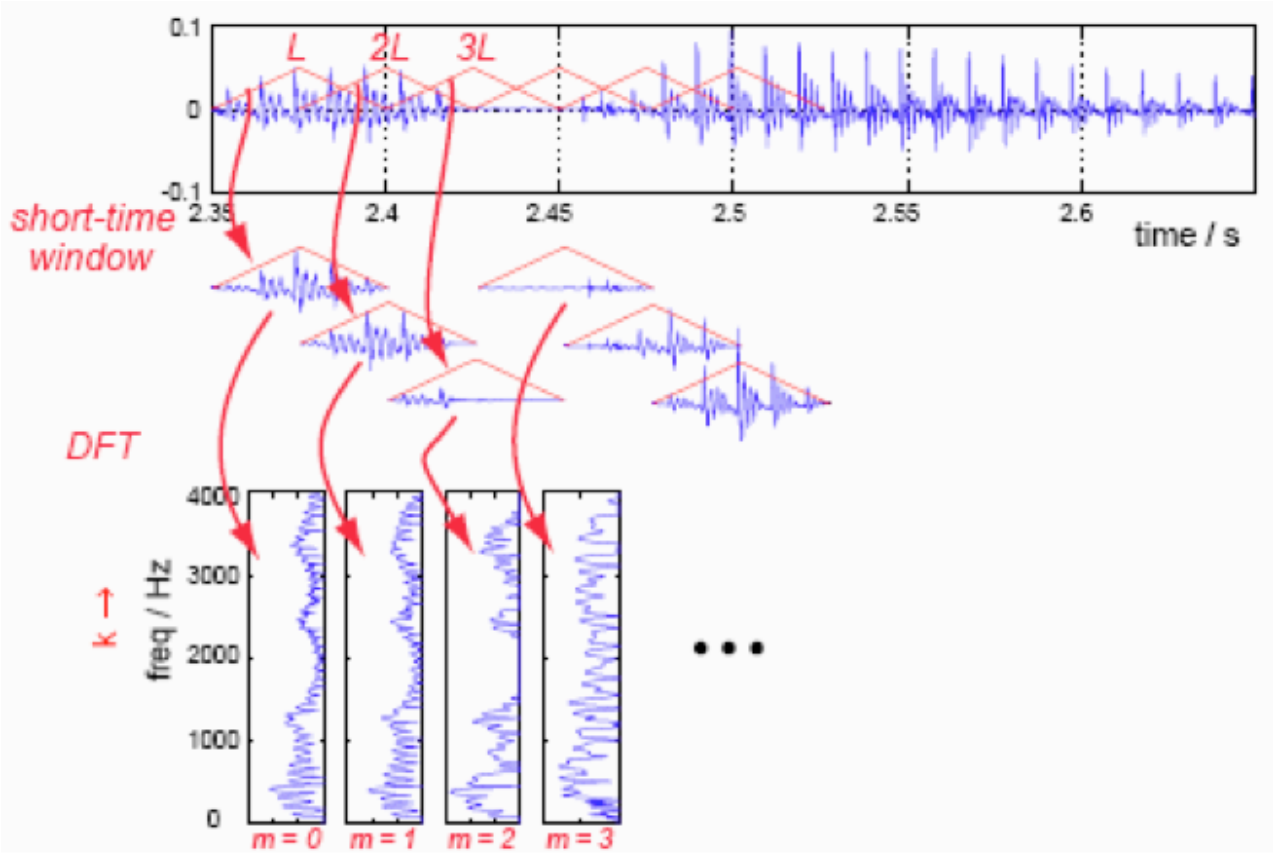


Chunks

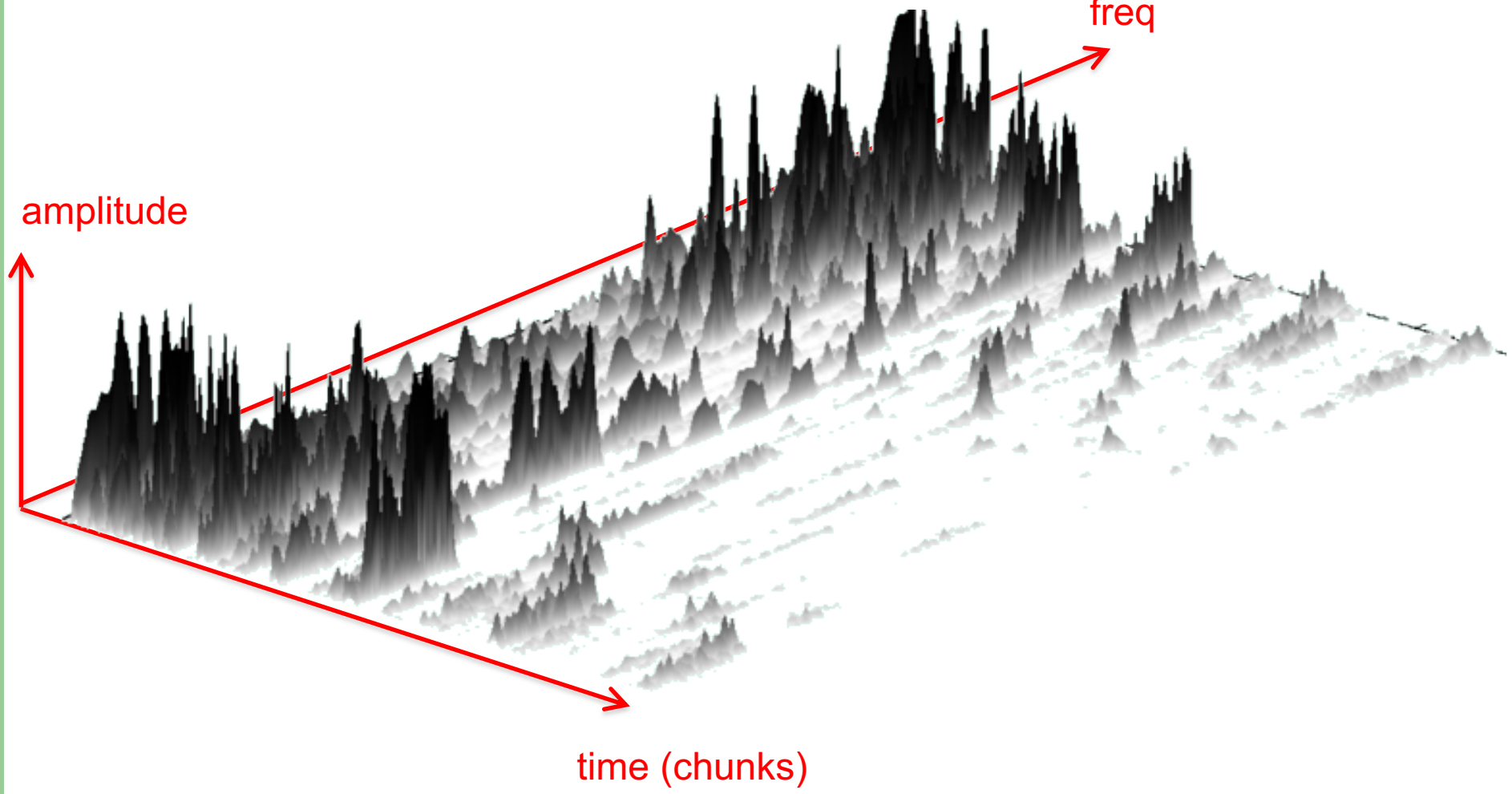
- Chunks (aka frames) are overlapped
  - To further reduce border effects
- Usually:
  - Each chunk lasts 25 ms
  - Chunks are separated by 10 ms



Short-time Fourier transform

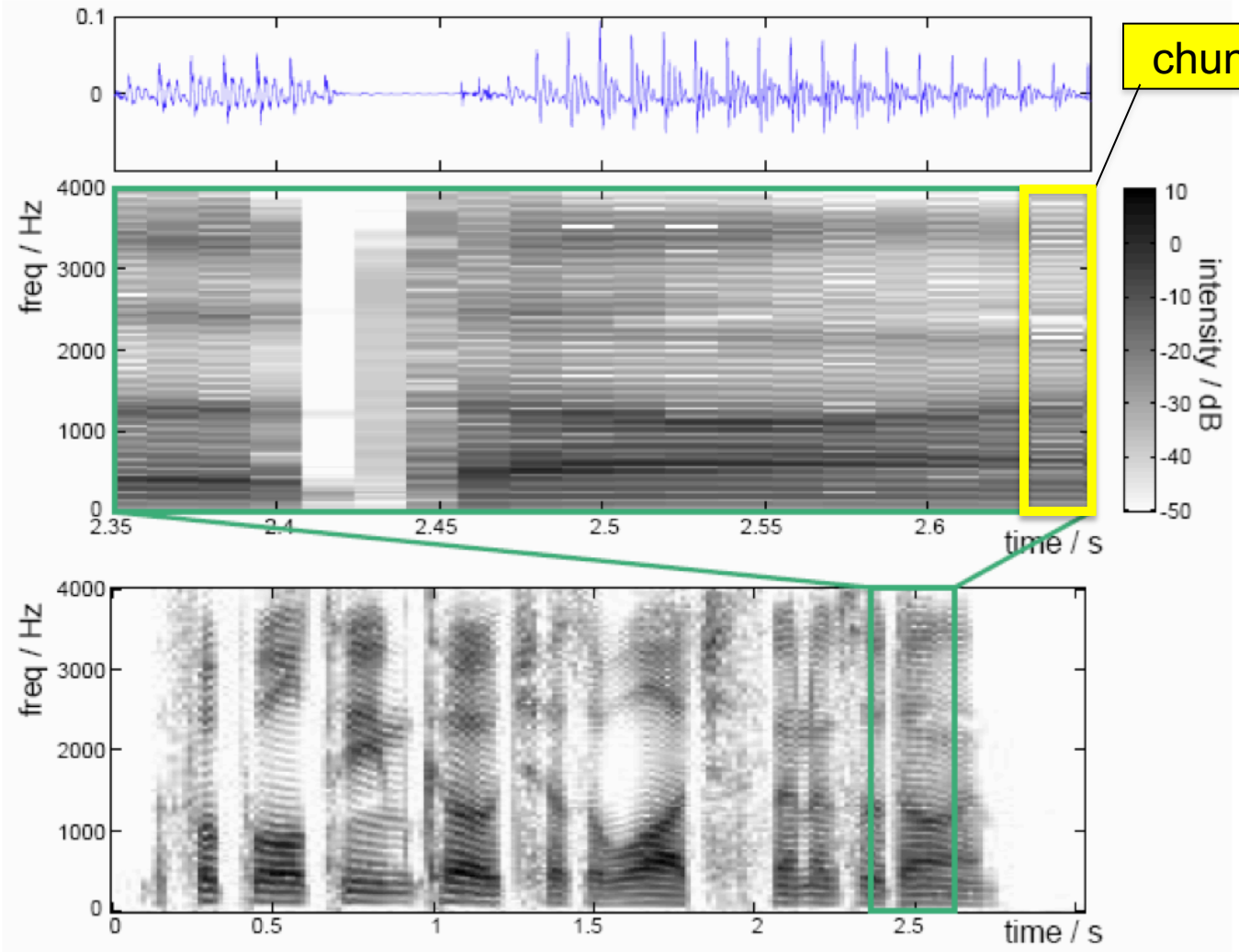


Spectrogram (3D view)



http://en.wikipedia.org/wiki/File:Spectrogram.png

Spectrogram (2D view)

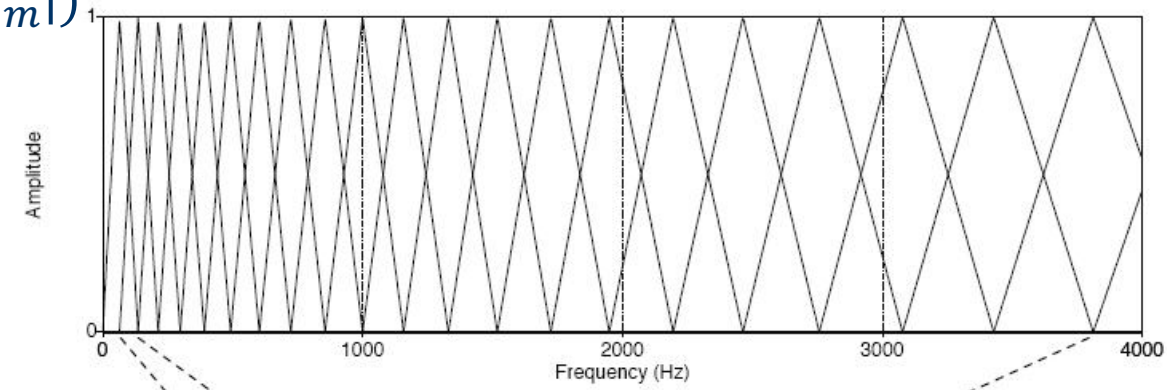


Amplitudes: shades of grey

Mel filter bank

- Human response to frequency is not flat
  - It is less sensitive to high frequencies (above 1000Hz)
- According to this idea, the mel filter bank collects *amplitudes* from varying frequency ranges
- Triangular filters with linear distance above 1000Hz, then logarithmic distance

$$\text{mel}_m = \text{mel\_filters}(|\text{STFT}_m|)$$

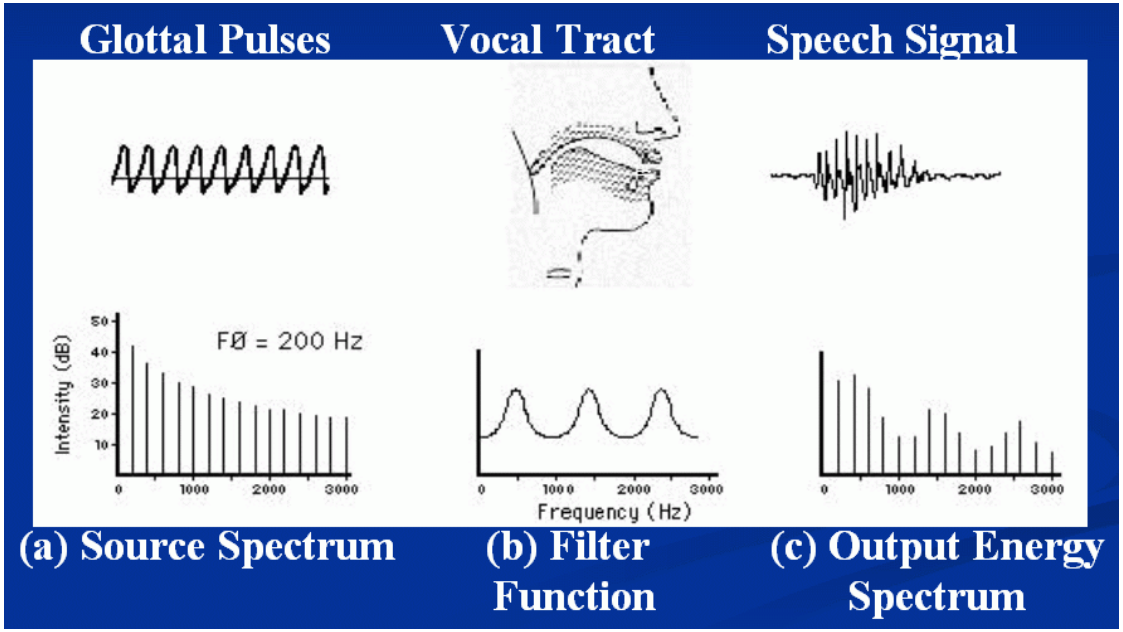


$\text{mel}_m[0{:}N{-}1]$

a vector of $N<L$ elements for the m-th chunk

The source-filter model

- A model of human phonation
- The glottis produces pulses (source), the vocal tract shapes such pulses (filter)



http://mirlab.org/jang/books/audiosignalprocessing/humanVoiceProduction.asp?title=3-3%20Human%20Voice%20Production

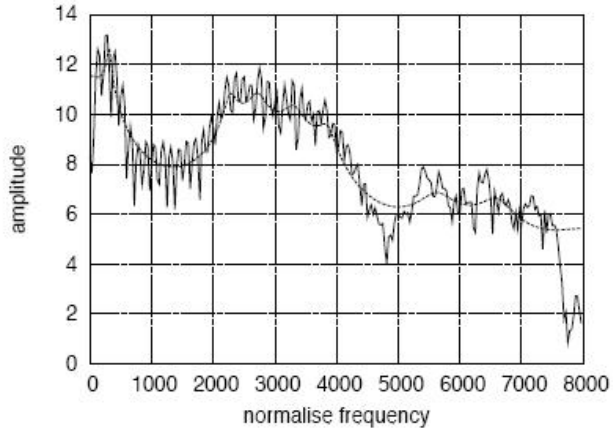
Cepstrum

- Source and filter are not equally useful
  - The source is not important for ASR
  - The filter carries the important info
- The cepstrum can be used to separate source and filter
  - Cepstrum: *the spectrum of the log of the spectrum*
    - Take a digitized waveform $x[n]$
    - Compute discrete Fourier transform $X[k]$ and consider amplitudes
    - Consider the discrete Fourier transform as a new digitized waveform
    - Compute the discrete Fourier transform of the log of such new waveform
    - The cepstrum coefficients are the amplitudes of such new spectrum
  - Cepstrum separates filter and source
- Mel-cepstrum: apply the cepstrum to the mel spectrum

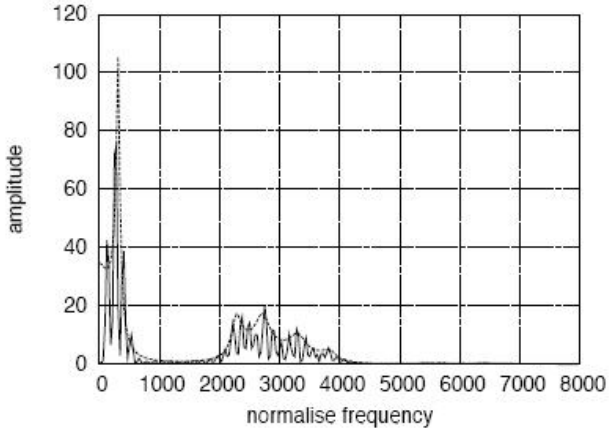
Mel-cepstrum

$$c'_m[n]\{\text{mel}_m\} = \sum_{k=0}^{N-1} \log(\text{mel}_m[k]) \cdot e^{j\frac{2\cdot\pi}{N}\cdot k\cdot n}$$

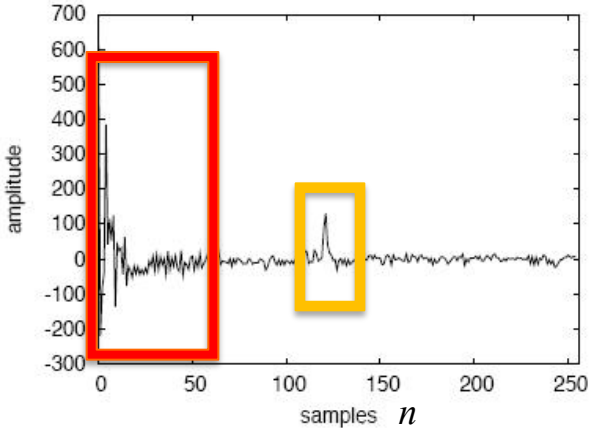
$|c'_m|$: vector of n mel-cepstrum coeff. for the m-th chunk $\quad 0 \le n \le N-1$



mel spectrum



log mel spectrum



cepstrum

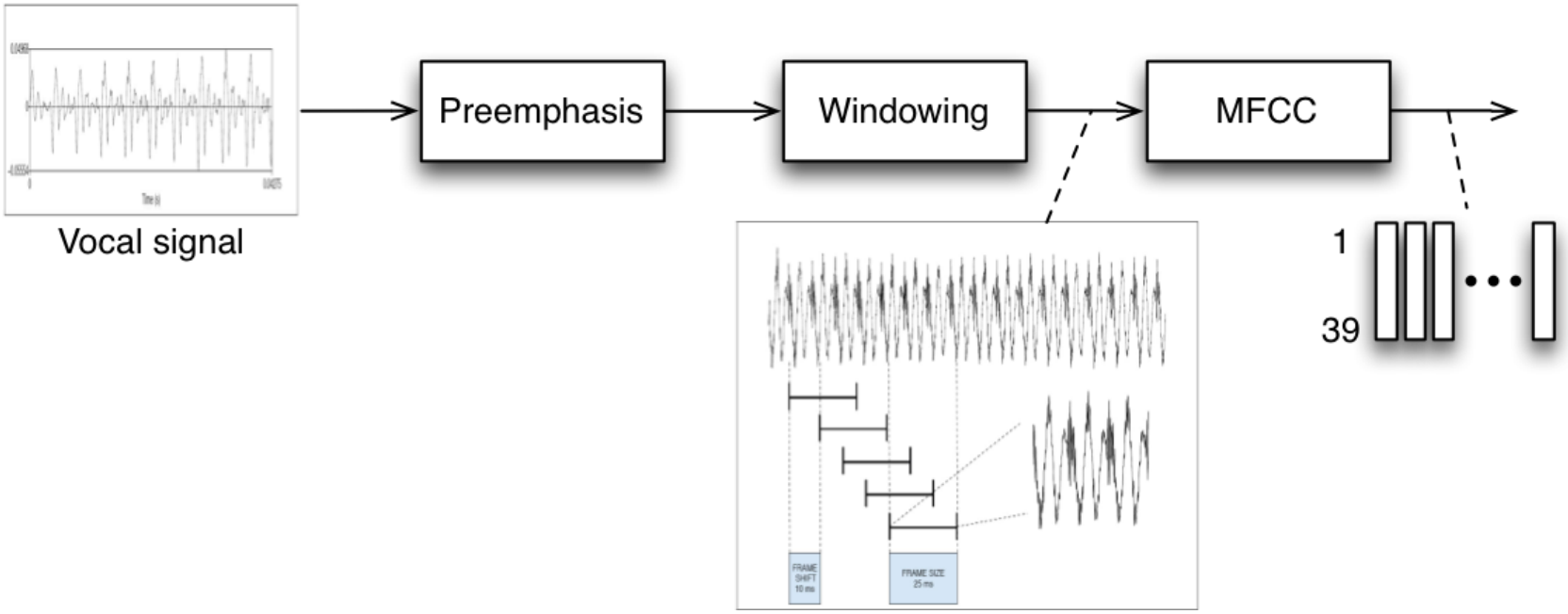
- Peak at 120 corresponds to F_0 in the mel spectrum → glottal pulse (the **source**) → F_0 is not an interesting feature!
- Lower components: the articulation (the **filter**)

MFCC coefficients

- First mel-cepstrum coefficients: $c_m[n] = |c'_m[n]|;\ 0 \le n \le 11$
  - Property: the variance of such coefficients tend to be uncorrelated
- Variations
  - Deltas: $$d(c_m[n]) = \frac{c_m[n+1] - c_m[n-1]}{2};\quad 0 \le n \le 11$$
  - Double-deltas: $$dd(c_m[n]) = \frac{d_m[n+1] - d_m[n-1]}{2};\quad 0 \le n \le 11$$
- Energy:
  - Energy of the samples: $E_m = \sum_{n=0}^{L-1} x^2[n]$
  - Energy of deltas: $E_d = \sum_{n=0}^{L-1} d^2(c_m[n])$
  - Energy of double-deltas: $E_{dd} = \sum_{n=0}^{L-1} dd^2(c_m[n])$
- 39 coefficients (tend to be uncorrelated) for each chunk

Summing-up so far

- Preemphasis boosts energy in the higher frequencies



- Every 10 ms (a chunk), we have a 39-vaues MFCC vector

The problem of ASR

- Given a vocal signal, composed of M MFCC vectors, compute the right sequence of M *subphones* s:

$$\hat{s}[0:M-1] = \underset{s[0:M-1]}{\operatorname{argmax}} \left(P(s[0:M-1] \mid \mathrm{MFCC}[0:M-1]) \right)$$

- The usual Bayesian approach:

$$P(s[0:M-1] \mid \mathrm{MFCC}[0:M-1]) = \frac{P(\mathrm{MFCC}[0:M-1] \mid s[0:M-1]) \cdot P(s[0:M-1])}{P(\mathrm{MFCC}[0:M-1])} =$$

$$= \frac{\prod_{m=0}^{M-1} P(\mathrm{MFCC}[m] \mid s[m]) \cdot P(s[0:M-1])}{P(\mathrm{MFCC}[0:M-1])} = \frac{\prod_{m=0}^{M-1} P(\mathrm{MFCC}[m] \mid s[m]) \cdot P(s[m] \mid s[m-1])}{P(\mathrm{MFCC}[0:M-1])}$$

Hp: MFCC independent
Hp: MFCC[m] only depends on s[m]

Markovian assumption

The problem of ASR

- Thus…

$$\hat{s}[0:M-1] = \underset{s[0:M-1]}{\operatorname{argmax}} \left(\frac{\prod_{m=0}^{M-1} P(\mathrm{MFCC}[m] \mid s[m]) \cdot P(s[m] \mid s[m-1])}{P(\mathrm{MFCC}[0:M-1])} \right) =$$

$$= \underset{s[0:M-1]}{\operatorname{argmax}} \prod_{m=0}^{M-1} P(\mathrm{MFCC}[m] \mid s[m]) \cdot P(s[m] \mid s[m-1])$$

Emission probability distribution B

Transition probability distribution A

- The model is an HMM

HMM for ASR

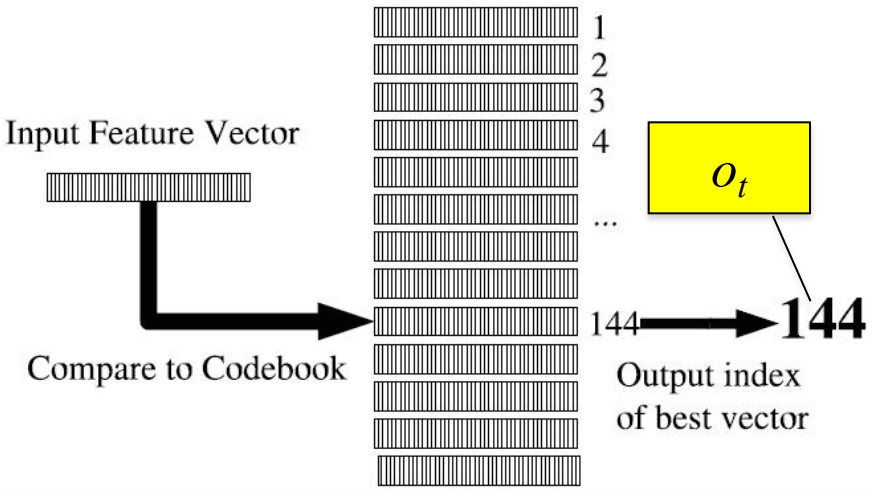
- $Q=\{s_1,s_2,\ldots s_{Ns}\}$: set of subphones
- O: set of observations (MFCC vectors)
  - each observation is a possible MFCC vector
- $A=\{a_{i,j}\}_{(Ns \times Ns)}$: transition probability matrix
  - probability to move from s_i to s_j: $a_{i,j} = P(s[m] = s_j \mid s[m-1] = s_i)$
  - It's usually called the **Language Model** (LM)
- $B=\{b_j(o_t)\}_{(Ns \times V)}$: emission probability matrix
  - probability that subphone j emits the observed MFCC vector o_t: $b_j(o_t) = P(MFCC[m] = o_t \mid s[m] = s_j)$
  - It's usually called the **Acoustic Model** (AM)

The emission probability distrib.

- $\mathrm{B}=\{b_j(o_t)\}_{(Ns \times V)}$: emission probability matrix
  - probability that subphone j emits the observed MFCC vector o_t
- MFCC vector contains 39 *real* numbers
- Two approaches here:
  - Discretize MFCC → discrete HMM
    - B stored ad an actual matrix $\{b_j(o_t)\}$
  - Continue MFCC → HMM-GMM uses Gaussian Mixture Model (GMM)
    - $b_j(o_t)$ is actually a function (no matrix B exist)
    - we store the *parameters* of $b_j(o_t)$

Discrete HMM

- Vector quantization
  - Codebook: a list of all V possible MFCC vectors $C_{book}=\{v_1,..,v_V\}$
  - Clustering algorithm: the codebook entries are created by clustering a set of sample MFCCs into V clusters (e.g., V=256)
    - Centroids will be the MFCCs to add to the codebook: *codevectors*
  - Distance metrics: for comparing codevectors in the codebook with a new MFCC, in order to select the closest one
  - The index of the selected MFCC is the o_t
- It is too simple…



HMM-GMM

- It's an HMM using GMM distributions to model emission probabilities
  - Model the entire MFCC vector with a single, multivariate Gaussian with D=39 dimensions
  - Assume that each parameter of the MFCC could have a very non-Gaussian distribution → weighted mixture of G Gaussians

$$b_j(o_t) = \sum_{n=0}^{G-1} c_n^{(j)} \cdot g(o_t \mid \mu_n^{(j)}, \Sigma_n^{(j)})$$

mixture of G weighted multivariate Gaussian, associated to the subphone j

$$g(o_t \mid \mu_n^{(j)}, \Sigma_n^{(j)}) = \frac{1}{(2\pi)^{D/2} \cdot \left|\Sigma_n^{(j)}\right|^{1/2}} \cdot \exp\left((o_t - \mu_n^{(j)})^T \cdot \left(\Sigma_n^{(j)}\right)^{-1} \cdot (o_t - \mu_n^{(j)})\right)$$

n-th multivariate Gaussian with D dim.

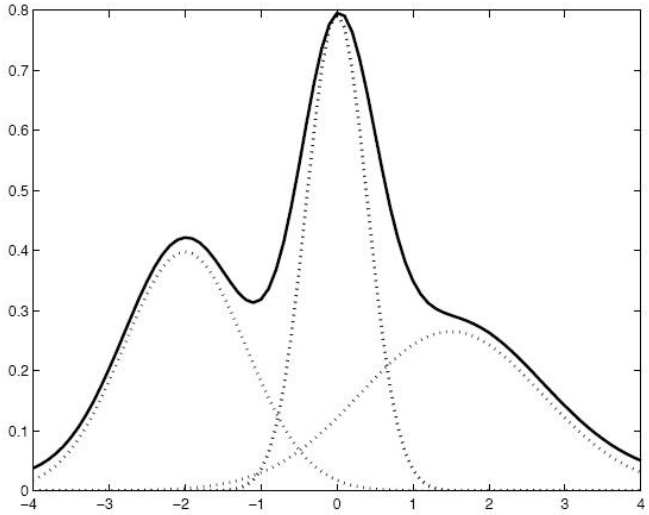
$c^{(j)}$: vector of G weights, $c_0^{(j)} \ldots c_{G-1}^{(j)}$, associated to the j-th subphone

$\Sigma_n^{(j)}$: covariance matrix ($D \times D$) for the n-th multivariate, associate to the j-th subphone

$\mu_n^{(j)}$: mean vector (D elements), for the n-th multivariate, associate to the j-th subphone

Gaussian Mixture Model

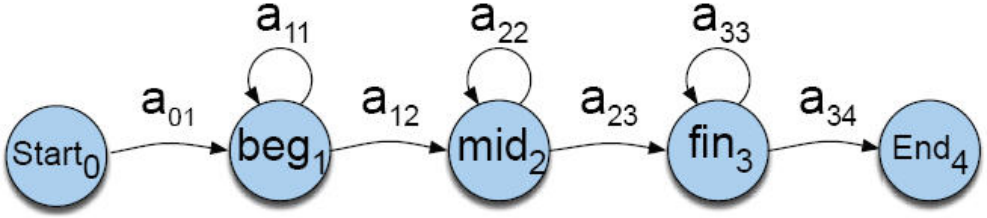
- Usually, covariance matrices are diagonal
  - remember the indipendence of MFCC values…
- Usually $b_j(o_t)$ are computed as logprob
- Why a mixture of Gaussians…



A mixture of three Gaussians approximating an arbitrary function

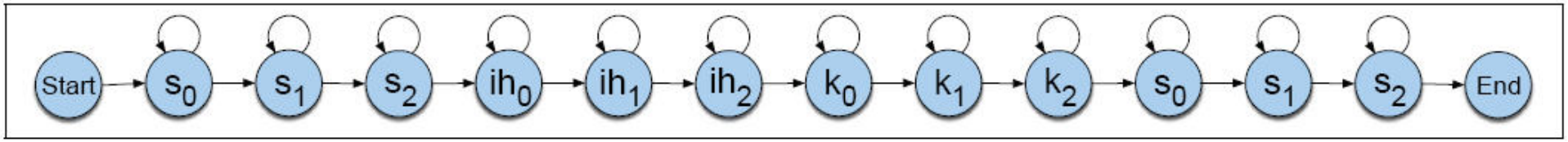
Phones and subphones

- Each phone is represented with five states:
  - Three emitting (beginning, middle, final) subphone states
  - Two non-emitting (start, end) states



$a_{i,j}$: transition probabilities

  - From each state, it is possible either to loop or to go to the next one
- Composing phones (e.g., “six” = [s ih k s]):
  - Replaces start and end with the emitting state of preceding and following subphones



The transition probability distrib.

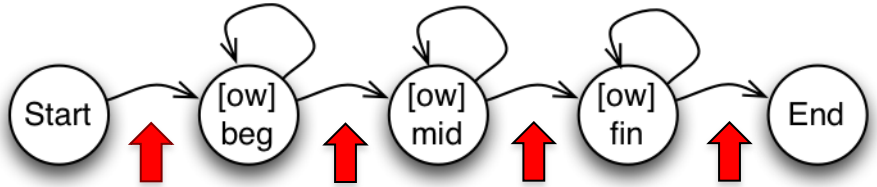
- $A=\{a_{i,j}\}_{(Ns \times Ns)}$: transition probability matrix
  - probability to transit from s_i to s_j
- The set of subphones $Q=\{s_1, s_2, \ldots s_{Ns}\}$ can be seen as containing all subphones, for all words, plus some special subphones:

$$Q = \{SIL, \ldots, [ow]^{OH}_{beg}, [ow]^{OH}_{mid}, [ow]^{OH}_{fin}, [ow]^{ZERO}_{beg}, [ow]^{ZERO}_{mid}, [ow]^{ZERO}_{fin}, \ldots$$
$$, [t]^{TWO}_{beg}, [t]^{TWO}_{mid}, [t]^{TWO}_{fin}, [t]^{THREE}_{beg}, [t]^{THREE}_{mid}, [t]^{THREE}_{fin}, \ldots, START, END\}$$

- E.g., [ow], for words “oh”, “zero”, are different subphones
- E.g., [t] for words “two”, and “three” are different subphones
- Probabilities:
  - Intra-phone, intra-word, inter-word

The transition probability distrib.

- Intra-phone probabilities are equal for all words (e.g., "oh", "zero")



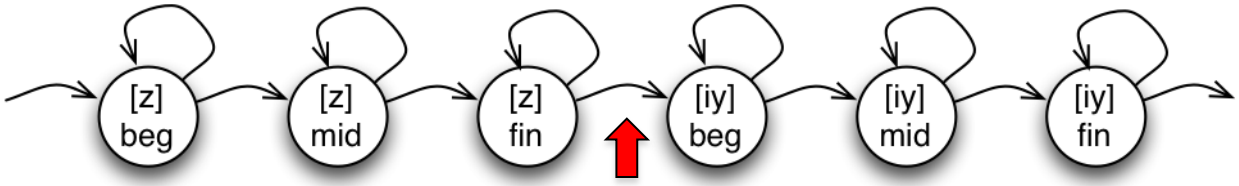
$$P(END \mid [ow]_{fin}^{OH}) = P(END \mid [ow]_{fin}^{ZERO}) = P(END \mid [ow]_{fin})$$

$$P([ow]_{fin}^{OH} \mid [ow]_{mid}^{OH}) = P([ow]_{fin}^{ZERO} \mid [ow]_{mid}^{ZERO}) = P([ow]_{fin} \mid [ow]_{mid})$$

$$P([ow]_{mid}^{OH} \mid [ow]_{beg}^{OH}) = P([ow]_{mid}^{ZERO} \mid [ow]_{beg}^{ZERO}) = P([ow]_{mid} \mid [ow]_{beg})$$

$$P([ow]_{beg}^{OH} \mid START) = P([ow]_{beg}^{ZERO} \mid START) = P([ow]_{beg} \mid START)$$

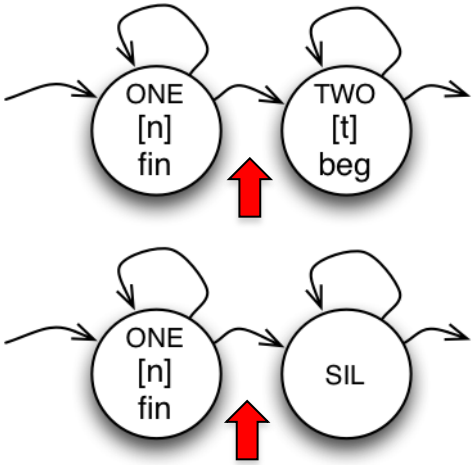
- Intra-word probabilities are equal for all words (e.g., "zero" "zeroth")



$$P([iy]_{beg}^{ZERO} \mid [z]_{fin}^{ZERO}) = P([iy]_{beg}^{ZEROTH} \mid [z]_{fin}^{ZEROTH}) = P(END \mid [z]_{fin})$$

The transition probability distrib.

- Inter-word probabilities are *not* equal for all words



$$P([t]_{beg}^{TWO} \mid [n]_{fin}^{ONE}) = P(TWO \mid ONE)$$

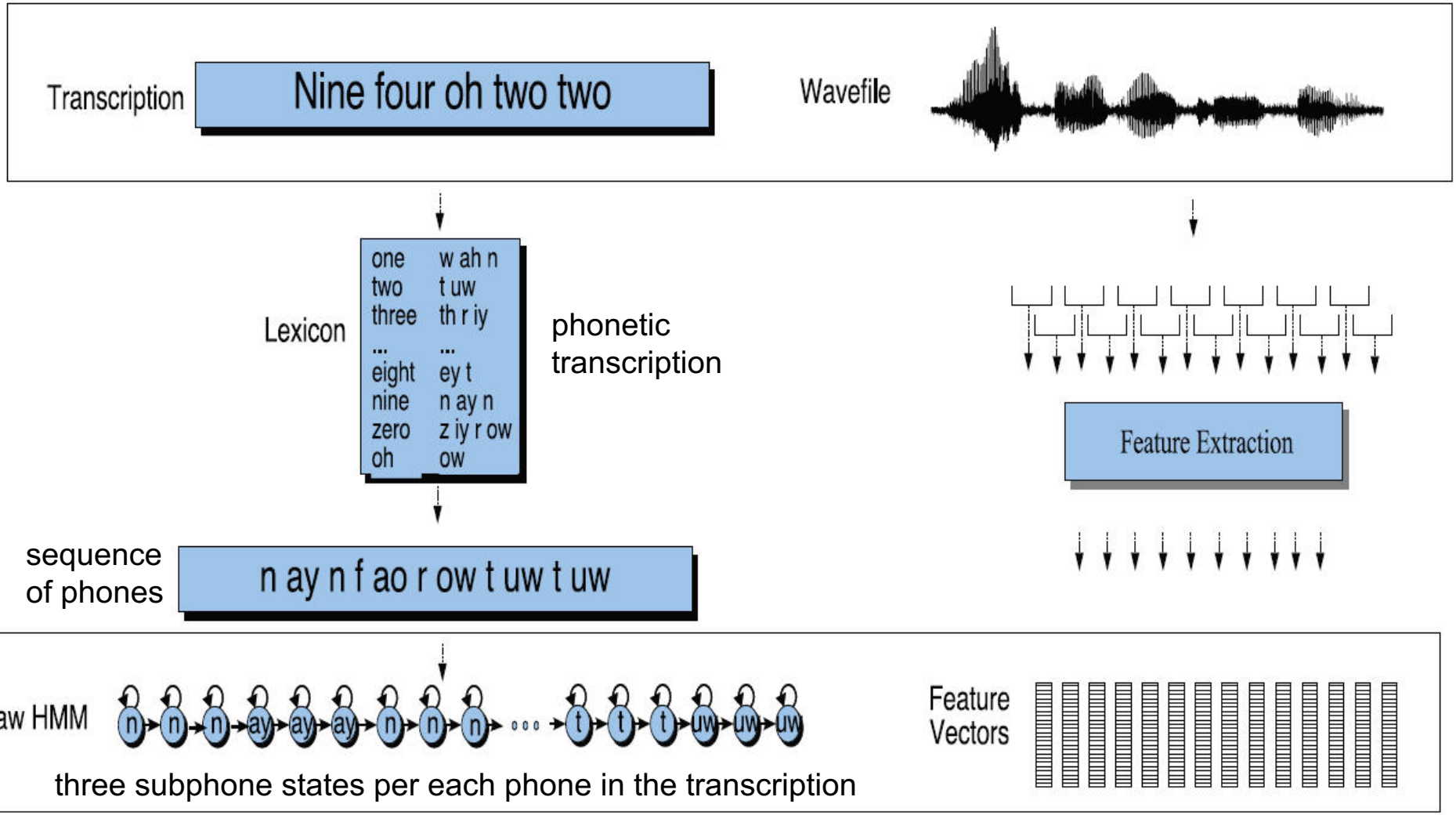
$$P(SIL \mid [n]_{fin}^{ONE}) = P(SIL \mid ONE)$$

- Summing up, A does not contain transitions for all subphones, as only inter-word probabilities are word-dependent
- Thus, Ns is actually:

$$Ns = |\mathrm{Q}| = \text{number of unique subphones} + 2 \times |\mathrm{V}|$$

Embedded training

- Training with Baum-Welch
- No annotated corpus is needed
- initial values for $a_{i,j}$ and $b_j(o_t)$ are needed
  - zero for $a_{i,j}$ we want to remain zero
  - 0.5 for other probabilities



Embedded training

- Given: phoneset, pronunciation lexicon, and the transcribed wavefiles
- Build a “whole sentence” HMM for each sentence
- Initialize A probabilities to 0.5 (for loop-backs or for the correct next subphone) or to zero (for all other transitions)
- Initialize B probabilities
  - By setting the mean and variance for each Gaussian to the global mean and variance for the entire training set, in case of HMM-GMM
  - Random, in case of discrete B
- Run multiple iterations of the Baum-Welch algorithm

Training

- Embedded training
  - → intra-phone and intra-word probabilities of A
  - → emission probabilities (matrix B or parameters of GMM)
- Using a language model (a bigram)
  - → inter-word probabilities of A
  - → usual MLE approach, from the corpus

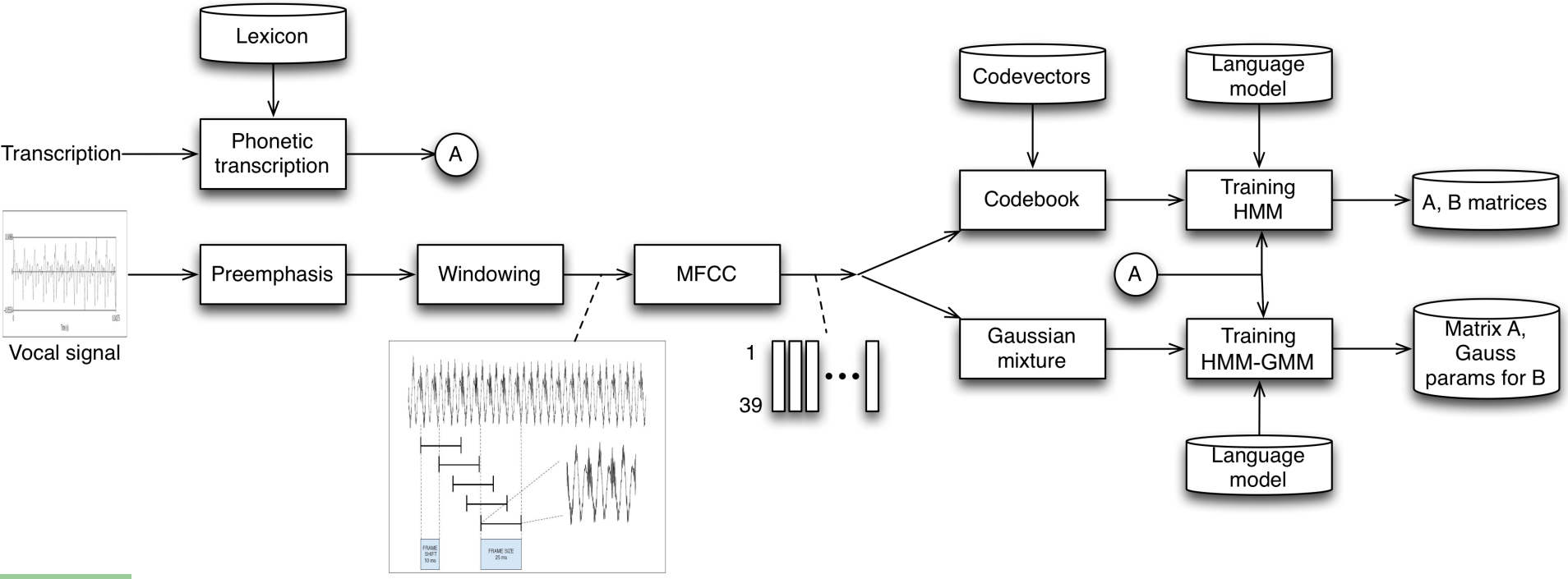
Decoding the HMM/HMM-GMM

- Viterbi; temporal complexity: $\mathrm{O}(M \times |\mathrm{Q}|^2)$
- Beam search
  - low probability paths are pruned at each time step
- Example output:

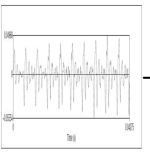
$$\text{"zero, oh"} \rightarrow \; [z]^{ZERO}_{beg}, [z]_{mid}, [z]_{fin}, [iy]_{beg}, [iy]_{mid}, [iy]_{fin}, [r]_{beg}, [r]_{mid}, [r]_{fin},$$
$$[ow]_{beg}, [ow]_{mid}, [ow]_{mid}, [ow]_{fin}, SIL,$$
$$[ow]^{OH}_{beg}, [ow]_{mid}, [ow]_{mid}, [ow]_{mid}, [ow]_{end}$$

Applying a lexicon… → zero oh

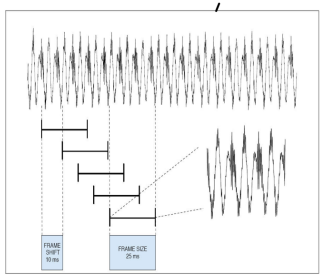
Recap: training

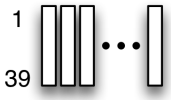


Recap: recognition



Vocal signal





Evaluation

- Word Error Rate (WER) in a word string:
  - How much the word string returned by the system (the *hypothesized* word string) differs from the *reference* word string
  - Based of the edit distance

$$WER = 100 \cdot \frac{\text{Insertions} + \text{Substitutions} + \text{Deletions}}{\text{Total words in correct transcript}}$$

- Sentence Error Rate (SER):
  - How many sentences had at least one error

$$SER = 100 \cdot \frac{\text{\# of sentences with at least one errror}}{\text{Total number of sentences}}$$

Evaluation

- Comparing two ASRs
- Matched-Pair Sentence Segment Word Error (MAPSSWE) is a significance t-test

	I	II	III	IV
REF:	it was	the best	of	times it	was the worst	of times		it was
SYS A:	ITS	the best	of	times it	IS the worst	of times	OR	it was
SYS B:	it was	the best		times it	WON the TEST	of times		it was

Align word string generated by the two system, with the correct transcription

N_A^i : number of errors made on segment i by system A

N_B^i : number of errors made on segment i by system B

D : $N_A^i - N_B^i$, $i = 1,2...n$; where n is the number of segments

$$\bar{D} = \frac{\sum_i^N N_A^i - N_B^i}{n}$$

$$S_D = \sqrt{\frac{\sum_i^N \left(D_i - \bar{D}\right)^2}{n-1}}$$

$$t = \frac{\bar{D}}{S_D / \sqrt{n}}$$

- Them compute the p-value: $2 \cdot P(t,n)$
- If $2 \cdot P(t,n) < 0.01$ (or 0.05), difference is significant

Advanced ASR: Problems with Viterbi

- Suppose the correct word sequence includes a word with very many pronunciations
  - The probabilities leaving the *start* arc of each word must sum to 1.0 → Viterbi may ignore this many-pronunciation word in favor of an incorrect word with only one pronunciation path
  - Viterbi penalizes words with many pronunciations
  - In fact, for decoding, we want to compute the likelihood:

$$P(acoustic \mid word\_sequence) = \sum_{s \in subphones} P(acoustic, s \mid word\_sequence)$$

  - But Viterbi actually computes:

$$P(acoustic \mid word\_sequence) \approx \max_{s \in subphones} P(acoustic, s \mid word\_sequence)$$

- Viterbi not good for *N*-grams with *N* > 2

Advanced ASR: alternatives to Viterbi

- Stack decoder or A*
  - Take into account multiple pronunciations
  - Work for *N*-grams
  - Slow
- Multiple-pass decoding: break up the decoding process into two stages
  - First stage: use a bigram, or use simplified acoustic models → collect *N*-best solutions
  - Second stage: apply more sophisticated but slower decoding algorithms on such reduced search space

Advanced ASR: triphones

- Phones vary enormously based on the phones on either side
- Triphone model: represents a phone in a particular left and right context
  - E.g., [y-eh+l] means: "[eh] preceded by [y] and followed by [l]"
- For a phoneset with 50 phones, in principle we would need 50^3 or 125000 triphones
  - But actually only a small subset exist, in any given language…

Advanced ASR: posterior classifiers

- The posterior classifier (Neural Net or SVM) is generally integrated with an HMM architecture
  - It is often called a HMM-SVM or HMM-MLP hybrid approach
- Posterior classifiers calculate probability of state q_j given o_t observation: $P(q_j \mid o_t)$
  - but HMM needs the *likelihood* (i.e., the emission probability $P(o_t \mid q_j)$
  - Using Bayes, we compute the *scaled likelihood*:

$$\frac{P(o_t \mid q_j)}{P(o_t)} = \frac{P(q_j \mid o_t)}{P(q_j)}$$

Scaled likelihood is just as good as the regular likelihood (the emission probab.), since $P(o_t)$ does not depend on q_j

Advanced ASR: posterior classifiers

- Unlike the Gaussian model, the posterior approaches uses acoustic information from neighboring time periods
  - Feature vectors for the current frame
  - Plus the four previous and four following frames
- i.e. a total of 9 cepstral feature vectors instead of the single one that the Gaussian model uses

Advanced ASR: noise and mic independence

- Spectral subtraction: to deal with *additive noise* (external sound source that is relatively constant)
  - We estimate the average noise during non-speech regions and then subtract this average value from the speech signal
- Cepstral mean normalization: to deal with "*convolutional noise*" (introduced by channel characteristics like different microphones)
  - We compute the average of the cepstrum over time and subtract it from each frame
  - The average cepstrum models the spectral characteristics of the microphone and the room acoustics

Advanced ASR: speaker independence

- Vocal tract length normalization [1]
  - Warping the frequency axis of the speech power spectrum to account for the fact that the precise locations of vocal-tract resonances vary roughly monotonically with the physical size of the speaker
  - Normalize on speakers' age
- Pitch normalization [2]
  - MFCC does not make use of the pitch
  - But MFCC values can be distorted by the pitch
  - Normalization could help improving accuracy for children
  - Not easy…

Advanced ASR: non-words

- Short non-verbal sounds (coughs, loud breathing, throat clearing, …) → filled pauses
- Environmental sounds (beeps, telephone rings, door slams, …)
- For each non-verbal sound:
  - We create a special phone and add to the lexicon a special word consisting only of that phone
- We use normal training to train these phones
  - Training data transcripts include labels for these new special words
  - These words need to be added to the language model; often by just allowing them to appear in between any word

Advanced ASR: speaker adaptation

- Modify the acoustic model (several methods)
- An example: MLLR (Maximum Likelihood Linear Regression), for HMM-GMM
  - We start with a trained acoustic model and a small adaptation dataset from a new speaker
  - The idea is to learn a linear transform matrix (W) and a bias vector (ω) to transform the means of the acoustic model Gaussians; for a given $b_j(o_t)$, the new average is $\hat{\mu}^{(j)} = W \cdot \mu^{(j)} + \omega$
  - The transform is learned by using linear regression to maximize the likelihood of the adaptation dataset
  - Simplest case: one transform for all subphones
  - Having more data: different transforms

TTS

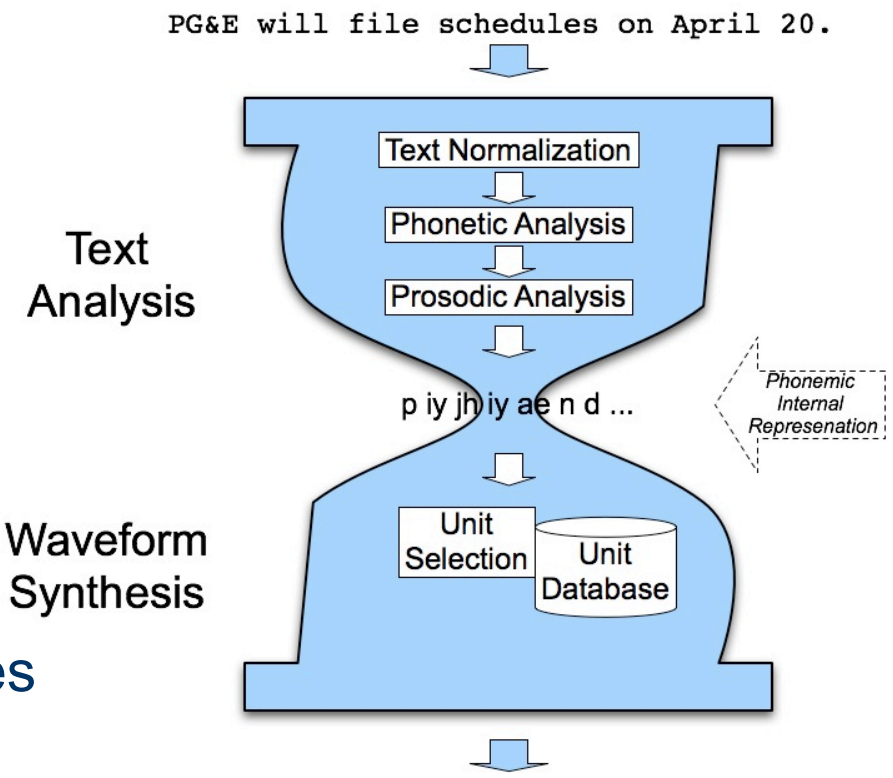
TTS

- Goal: transforming a text string into a waveform
- Steps:
  1. Text analysis: text string → phonetic representation
  2. Waveform synthesis: phonetic representation → waveform
- Approaches for waveform synthesis:
  - Formant synthesis: rules/filters are used to create speech using additive synthesis and an acoustic model (physical modelling synthesis) → "robotic" voice…
  - Articulatory synthesis: modelling and simulating movements of articulators and acoustics of vocal tract → complex
  - Concatenative synthesis: the most used approach in modern TTS

Concatenative TTS

- The hourglass model
- Two steps
  - Text analysis
  - Waveform synthesis

1. Text analysis
   - From text to phonemes
2. Waveform synthesis
   - From phonemes to waves
   - Concatenates small, prerecorded wave units



1. Text analysis

- Text normalization
  - Tokenization
  - Dealing with non-standard words
  - Homograph disambiguation
- Phonetic analysis
  - Producing the string of phonemes
- Prosodic analysis
  - Add information encoding prosody

Text normalization

- Sentence tokenization
  - Can be difficult if no punctuation is present
  - Often rely on machine learning
- Non-standard words (NSW)
  - Numbers, abbreviations, etc., that need to be expanded for pronunciation
  - Difficult, as pronouncing NSW can be ambiguous (e.g., how to pronounce 1970 (is it a date? a price? a string? …)
- Homograph disambiguation
  - Like “bass” as fish→/b ae s/ or as an instrument→/b ey s/
  - Or like Italian “àncora” and “ancóra” (accents are not mandatory and are usually not written…)
  - Leverage the context (i.e., use a WSD-like algorithm)

Phonetic analysis

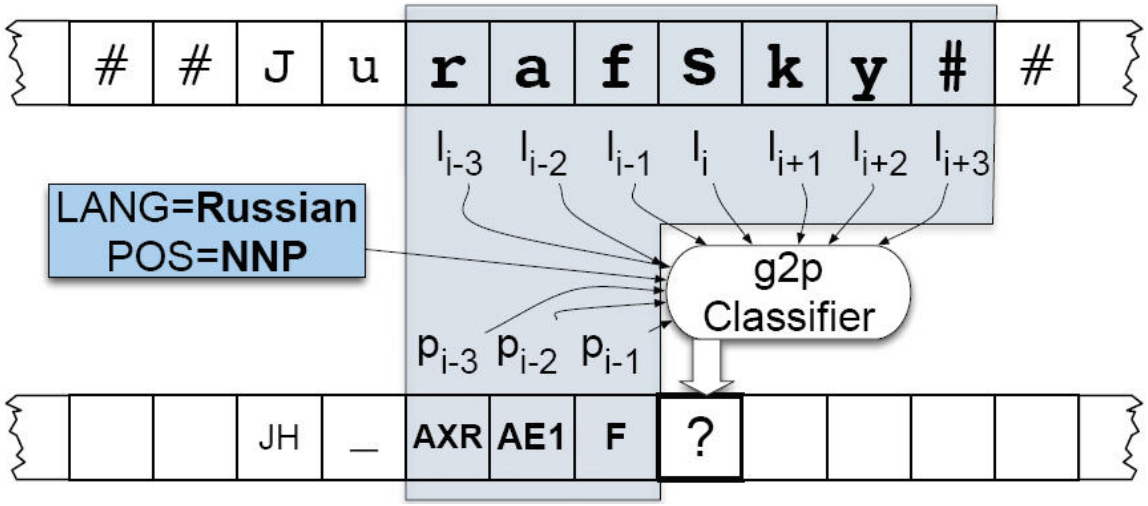
- From the preceding phase
- Transform a word into a list of phonemes
- Two general approaches
  - Dictionary-based: collection of pronunciations, for each word
  - Grapheme-to-phoneme (aka g2p) conversion: for words non present in the dictionary (mostly, names)
- For transparent languages (e.g., Italian)
  - Use pronunciation rules
  - Dictionary for irregular forms and foreign words

Phonetic analysis

- Dictionary contains the phonetic representation of words
  - For example, the CMU dictionary

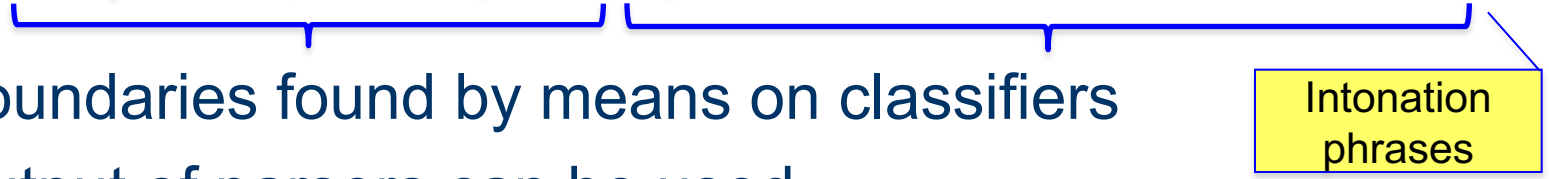
ANTECEDENTS	AE2 N T IH0 S IY1 D AH0 N T S	*PAKISTANI*	P AE2 K IH0 S T AE1 N IY0
CHANG	CH AE1 NG	*TABLE*	T EY1 B AH0 L
DICTIONARY	D IH1 K SH AH0 N EH2 R IY0	*TROTSKY*	T R AA1 T S K IY2
DINNER	D IH1 N ER0	*WALTER*	W AO1 L T ER0
LUNCH	L AH1 N CH	*WALTZING*	W AO1 L T S IH0 NG
MCFARLAND	M AH0 K F AA1 R L AH0 N D	*WALTZING(2)*	W AO1 L S IH0 NG

- Names and other unknown words:
  - g2p: a classifier
  - Uses several features



Prosodic analysis

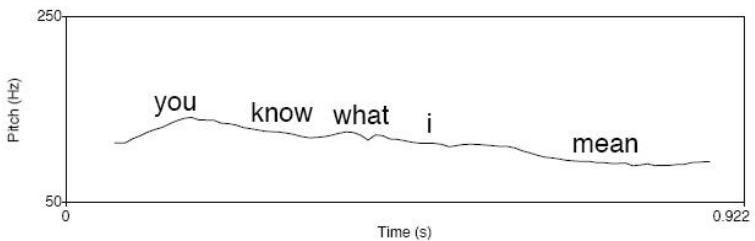
- Prosody: intonation, stress, and rhythm
  - F_0, phone duration, energy
  - Prosody is a *suprasegmental* aspect of spoken language
- Utterances have prosodic structure
  - Intonation phrases
  - Intermediate phrases
  - *I wanted to go to London, but could only get tickets for France*

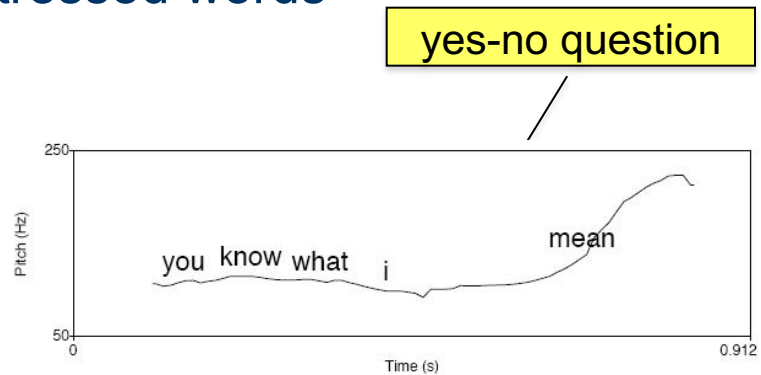


- Boundaries found by means on classifiers
- Output of parsers can be used
  - There is correlation between syntactic and prosodic structures

Prosodic analysis: prominence

- Prosodic prominence
  - In a phrase, some words are more prominent → "pitch accent"
    - Actually, realized by means of pitch and/or rhythm and/or energy
    - Pitch accent indicates "stress"
    - The pitch accent on a word is realized on its stressed syllable (where tonic accent lies)
    - The most prominent pitch accent is the phrase *nuclear accent*
  - Empathic accent: stress related to semantic reasons
  - Unaccented words: "normal" stress
  - Reduced accent: less than usual stressed words
- Tune: rise or fall of F_0





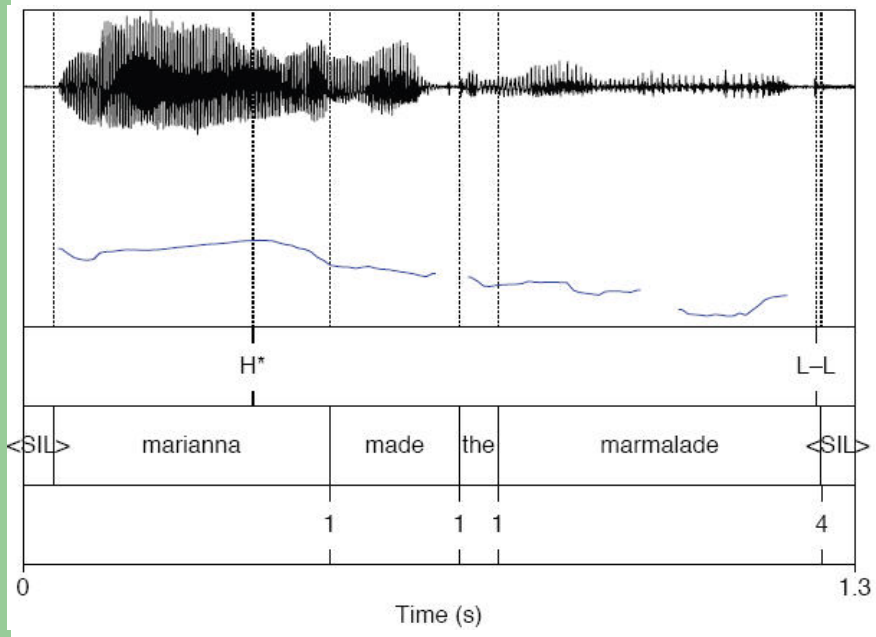
Prosodic analysis: ToBI

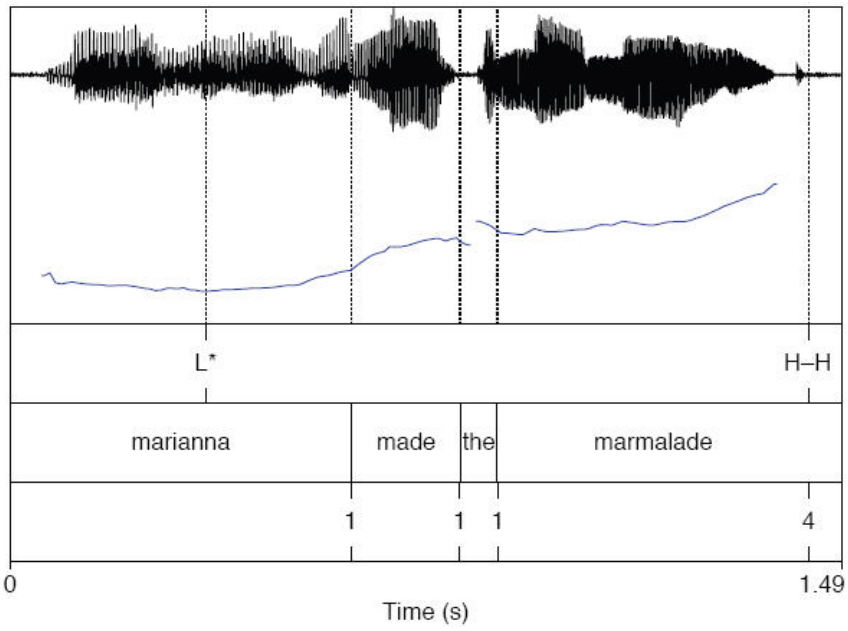
- ToBI: a phonological theory of intonation
  - Models prominence, tune, and boundaries
- Five pitch accents and four boundary tones
- Four phrasal breaks (intonational phrase, intermediate phrase, word breaks, word boundaries)

Pitch Accents		Boundary Tones	
H*	peak accent	**L-L%**	"final fall": "declarative contour" of American English
L*	low accent	**L-H%**	continuation rise
L*+H	scooped accent	**H-H%**	"question rise": cantonical yes-no question contour
L+H*	rising peak accent	**H-L%**	final level plateau (plateau because H- causes "upstep" of following)
H+!H*	step down		

Prosodic analysis: ToBI

- An example of ToBI annotation





Prosodic analysis: phone duration

- *Rules* that look at the context and change the typical duration of the phone (Klatt, 1979) → factor weights f_i
  1. Prepausal Lengthening: The vowel or syllabic consonant in the syllable before a pause is lengthened by 1.4.
  2. Non-phrase-final Shortening: Segments which are not phrase-final are shortened by 0.6.
  3. …

  Given the set of N factor weights f_i , the Klatt's formula for the duration d of a phone is:

$$d = d_{\min} + \prod_{i=1}^{N} f_i \times (\bar{d} - d_{\min})$$

- Or *Machine Learning*

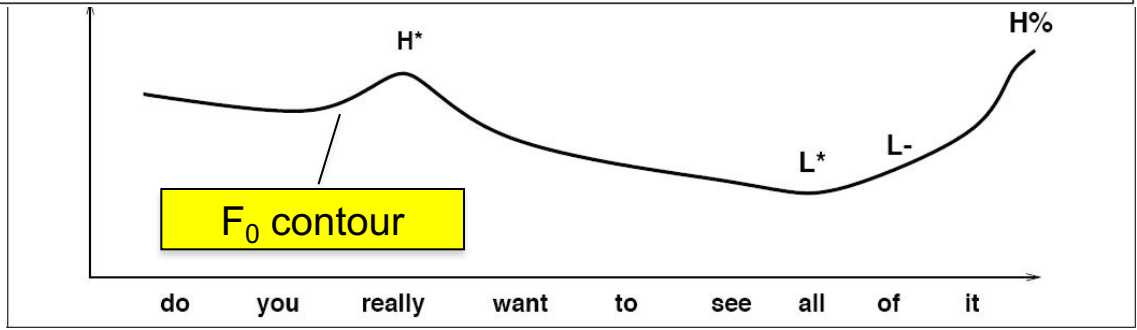
Prosodic analysis

- From the annotation, F_0 is generated in two steps:
  - Defining "key" F_0 values for pitch accents and breaks
  - Generate the F_0 contour interpolating such F_0 values
- The "key" F_0 values are defined as a percentage of a reference F_0

				H*										L*		L- H%			
do		you		really				want				to		see		all		of		it	
d	uw	y	uw	r	ih	l	iy	w	aa	n	t	t	ax	s	iy	ao	l	ah	v	ih	t
110	110	50	50	75	64	57	82	57	50	72	41	43	47	54	130	76	90	44	62	46	220

Phonemes with ToBI and durations

F_0 Generation



2. Waveform synthesis

- Two models for concatenative waveform synthesis
  - Diphone synthesis
  - Unit selection synthesis

ciao → /tS a o/ → (<s>, tS), (tS, a), (a, o), (o, </s>)

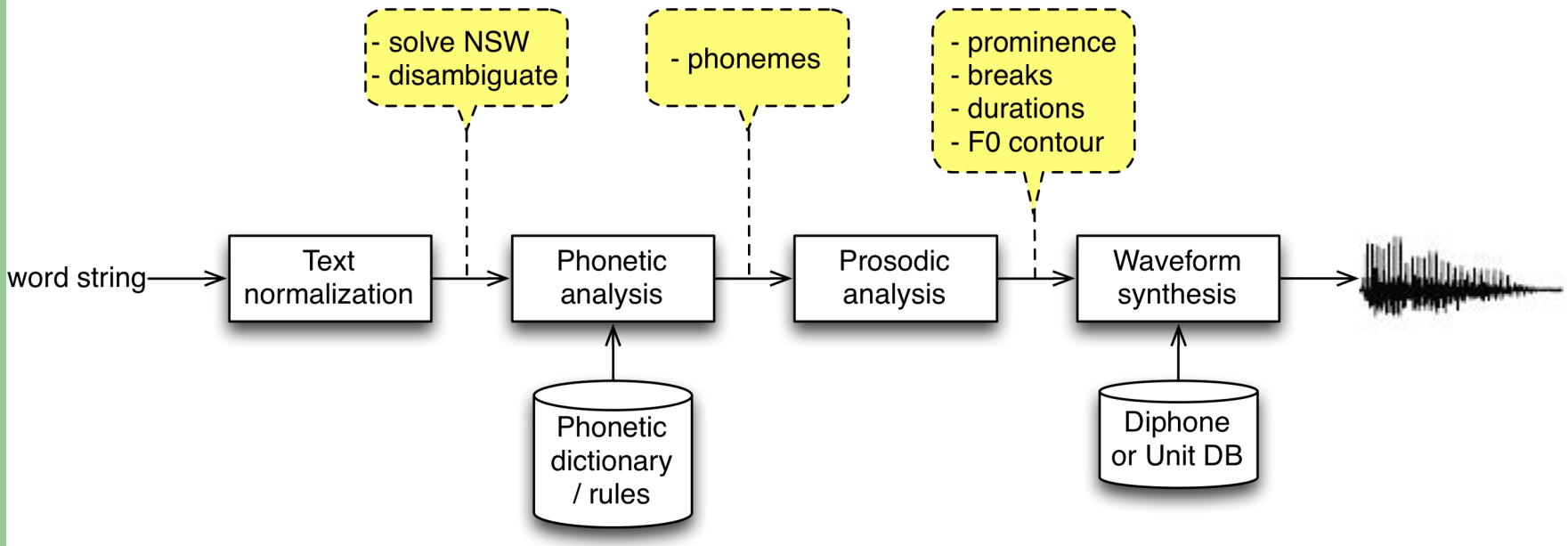
Diphone synthesis

- Diphone: phone-like unit going from the middle of one phone to the middle of the following one
  - C*oarticulation* phenomenon: each phone differs slightly, depending on the preceding and following ones
- A diphone database is a collection of recorded diphones
  - Record a speaker saying one example of each diphone
  - Mark the boundaries of each diphones
  - Cut each diphone out and create a diphone database
- Synthesizing an utterance:
  - Grab relevant sequence of diphones from database
  - Concatenate the diphones; clean boundaries
  - Use signal processing to change the prosody

Unit selection synthesis

- Database contains *units*: any piece of speech that can be concatenated
  - Diphones, syllables, …
- Given the list of phonemes with prosodic annotation, find in DB the best list of units → cost function:
  - "Target cost": Closest match to the target description, in terms of
    - Phonetic context
    - F_0, stress, phrase position
  - "Join cost": Best join with neighboring units
    - Matching formants + other spectral characteristics
    - Matching energy
    - Matching F_0
- Solved using Viterbi or beam search

Recap


- solve NSW
- disambiguate
- phonemes
- prominence
- breaks
- durations
- F0 contour
word string
Text normalization
Phonetic analysis
Prosodic analysis
Waveform synthesis
Phonetic dictionary / rules
Diphone or Unit DB

Evaluation

- Done by human testers
- Indexes:
  - Intelligibility: the ability of the tester to correctly interpret the words and the meaning of the utterance
    - Phone discrimination
    - Diagnostic Rhyme Test (DRT): intelligibility of initial consonants, in a standard word set
    - Modified Rhyme Test (MRT): identify words in a standard set
  - Quality: measure of naturalness, fluency, clarity of the speech
    - Mean Opinion Score (score 1—5 given to each system to compare)
    - AB test: the same utterance is produced by the two system; testers choose the best system. Repeat for 50 utterances

REFERENCES

References

- B. Gold, N. Morgan and D. Ellis. *Speech and Audio Signal Processing: Processing and Perception of Speech and Music.* Wiley-Interscience

ASR

- CMU Sphinx `http://cmusphinx.sourceforge.net`
- Julius `http://julius.sourceforge.jp/en_index.php`
- HTK `http://htk.eng.cam.ac.uk`
- Bavieca `http://www.bavieca.org`
- Kaldi `http://kaldi.sourceforge.net`
- SRILM `http://www.speech.sri.com/projects/srilm`
- DeepSpeech `https://github.com/mozilla/DeepSpeech`

References

TTS

- eSpeak `http://espeak.sourceforge.net`
  - Uses formant synthesis
- GnuSpeech `http://www.gnu.org/software/gnuspeech`
  - Uses articulatory synthesis
- FreeTTS `http://freetts.sourceforge.net/docs/index.php`
  - Uses concatenative synthesis
- MARY `http://mary.dfki.de`
  - Uses concatenative synthesis

References

[1] R. Serizel, D. Giuliani. *Vocal tract length normalisation approaches to DNN-based children's and adults' speech recognition*, IEEE Spoken Language Technology Workshop, 2014

[2] R. Sinha, S. Ghai. *On the use of pitch normalization for improving children's speech recognition*. INTERSPEECH, 2009