An isolated word speech to British Sign Language translator

Pedro Caio Castro Cortes C Coutinho, Thomas Lorblanchès and Romain Tajan

April 24, 2023

Abstract

This project addresses the problem of automated translation of a voice signal into American/British Sign Language (ASL/BSL). Common speech processing tools are developed to analyze and extract features from microphone recorded signals. A word recognition algorithm compares these features to a database, to estimate the spoken word. This estimated word is converted into a picture from BSL and displayed to the deaf user.

Contents

1 Organization and project evaluation

The duration of the project is 20h, divided in 5 sessions of 4h. The work must be carried out by groups of 3 students **maximum**. The assessment relies on a personal grade depending on the **student's behaviour** (assiduity and reliability during the sessions), as well as **the code and report of the project**. The instructions for the report are given below :

- An electronic pdf version of the final report is to be sent using the Thor interface to the professor in charge of your group. The Matlab source code produced during the project must also be uploaded using the same interface and does not have to be included in the appendices of the report.
- The report should be typed using Latex or Word, in plain page format with a font size of **11 points**. **The maximum number of pages allowed is 10**. If you are using Word, all equations should be typed with the specific editor, and should be numbered if they are cited. All figures should have a number and a title.
- Concerning the organization of the report, a table of contents should be inserted at the beginning. In a first introductory part, the project topic must be rephrased **with your own words** (do not rewrite the subject !). The remainder of the report should follow the organization of the present document.
- If some documents have been used during the project, they should be cited in a "References" section, at the end of the report.

2 Introduction

Recent developments of connected objects can be used to improve the daily life of disabled people and in particular deaf or hard of hearing people. For hard of hearing people, even simplest tasks such as calling someone that cannot read British Sign Language (BSL) are difficult. The solution proposed today is to video call an intermediary person that can read BSL. This person will translate the BSL to speech in one direction and speech to BSL in the other direction.

In this work, we propose to study the part of the system that detects spoken words and translates them into BSL pictures. This system is depicted in Figure 1; it acts as a supervised classifier. We train this classifier with many examples of spoken words during a training phase. After this training phase, we use the classifier for new speakers. In fact the classifier extracts features that we hope uniquely determine which word is spoken, compare the features of the new signal with features from signals of the training phase to determine which word is spoken. When a corresponding word is found, the picture or video of the corresponding BSL sign is displayed to the hard of hearing user.

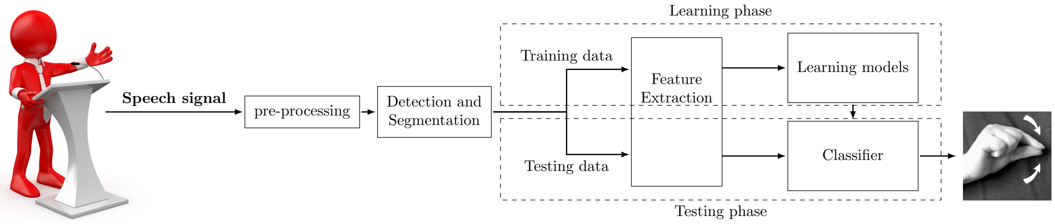


Figure 1: General framework

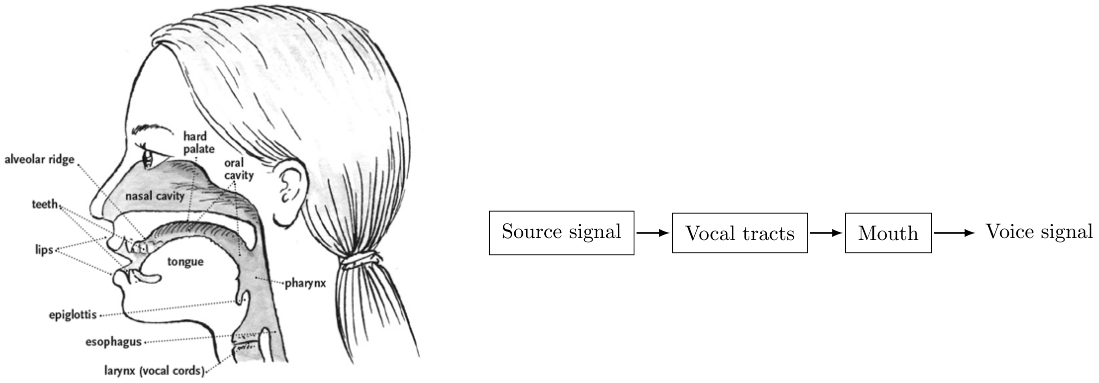
In Section 3, we derive general tools for time and frequency signal analysis. These tools will be used in Section 4 to extract features that will be used by the classifier to distinguish between words. In Section 5 we train and test the classifier. Finally, in Section 6 we implement a classifier for real time words recognition.

3 Speech signal analysis

3.1 Introduction to speech models

The goal of this section is to understand how a speech signal is structured. To do so, we will first present an anatomic model of the speech creation. From this anatomic model, we will present a linguistic model that will lead to a signal processing model.

How is speech created?
Anatomically, speech is formed by combining two phenomena. The first phenomenon is the formation of an input signal by breathing out air from lungs. This air vibrates or not vocal cords. In the first case, the input signal will be a periodic signal (think about some singer playing a note) whereas in the second case, the created signal will behave as a noise (think of a whispering). The second phenomenon is the modulation of the input signal by vocal tracts (composed by larynx, pharynx, and nasal cavity), the tongue and the mouth (cf Figure 2).



(a) Biological model

(b) Simplified block model

Figure 2: Anatomy of speech creation.

From this anatomic model we can understand that speech is created by a succession of small sounds that can use the vocal cords and that are shaped differently by the vocal tracts and the mouth. This way of representing speech is the linguistic model presented thereafter.

Linguistic model
From a linguistic perspective, speech is a succession of **phonemes**. Phonemes are "elementary" signals of 40 to 400 milliseconds composing the speech signal (see [2]). For the English language, a list of phonemes has been done for the ARPA Speech Understanding Project [1]. This list, called the ARPAbet, is provided in Appendix A.

Consider for example the sentence "Hello world". This sentence can be represented, using the ARPAbet, as "[HH AH L OW]. [W ER L D]." We can observe that not every phoneme uses the vocal chords. In the classification proposed in Appendix A, the phonemes named "vowel", "diphthongs", "glides", "liquids", and "nasals" uses vocal chords. These phonemes are said to be **voiced**. The other phonemes, namely "stops", "fricatives" and "affricates" do not use vocal chords. They are said to be **unvoiced** phonemes. In the "Hello world" example, phonemes AH, L, OW, W, and ER, are voiced whereas the phonemes H and D are unvoiced.

In this work, we will only distinguish voiced and unvoiced phonemes. We will not consider the finer classification of phonemes proposed in the ARPAbet. Also, we won't consider other linguistic aspects such as prosody.

From this linguistic model we can understand that speech is something that varies in time. Speech is then said to be non-stationary. In the next part we present the signal processing model of speech based on the two previous models.

Signal processing model

A signal processing model can be derived from the biologic model of speech creation. The source signal is $s(t)$ and model the air expelled from lungs. This source signal is filtered successively by two filters, the filter $h(t)$ representing the vocal tracts and the filter $g(t)$ representing the effect of mouth.These signal processing steps are depicted in Figure 3 where $x(t)$ represents the speech signal. The source signal $s(t)$ is either a **periodic source** for modeling voiced phonemes or a **white noise** process for representing unvoiced phonemes. Speech is composed of a succession of $40 - 400$ milliseconds phonemes. These phonemes are still non-stationary, however, one can show that speech signal is stationary for 10 to 30 milliseconds signals. The proposed model is only valid for pieces of the speech signal of length 10 milliseconds and changes with time. This means that filters $h(t)$ and $g(t)$ are time-varying.

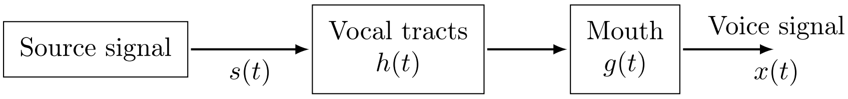


Figure 3: Signal processing model for speech creation

The acquisition of the speech signal $x(t)$ is performed by using a microphone (device that convert a sound to an electrical signal). The electric signal at the output of the microphone is considered to be equal to $x(t)$. $x(t)$ is sampled at a sampling rate $F_s = \frac{1}{T_s}$ giving the samples $x_n = x(nT_s)$. These samples x_n are the ones provided to MATLAB when reading a sound file. This acquisition chain is depicted in Figure 4.

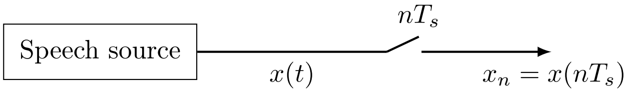


Figure 4: Acquisition of speech

Consider again the example of the sentence "Hello world", the sampled speech signal x_n for a sampling rate of $F_s = 44100$ samples per seconds is given in Figure 5 By looking at the signal depicted in Figure 5, one can notice that it exhibits different behaviors:

- the shape of a periodic signal, see for example Figure 6a,
- the shape of a noisy signal, see for example Figure 6b.

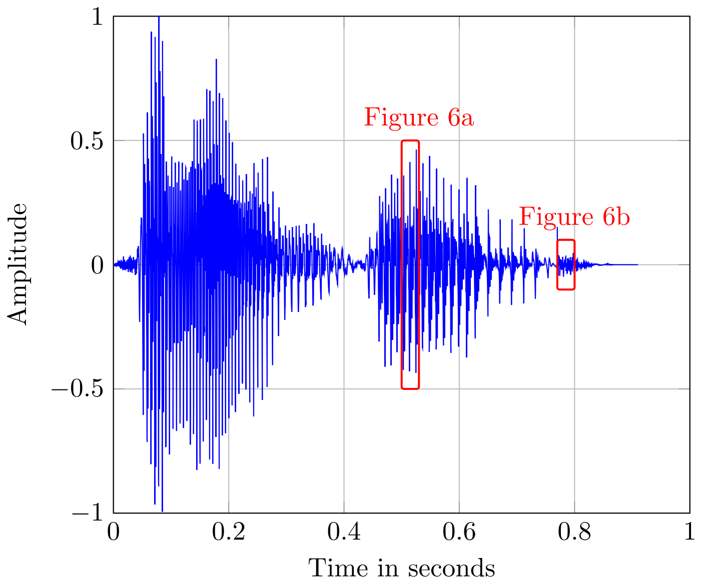
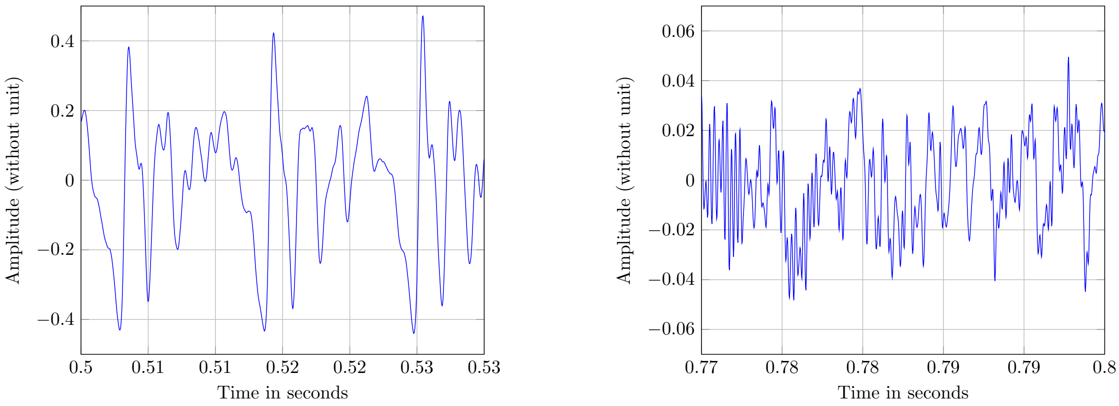


Figure 5: Speech signal containing the sentence "Hello world".



(a) A "periodic" part of the signal of Figure 5.

(b) A "noisy" part of the signal of Figure 5.

Figure 6: Two different kind of signals found in the speech signal "Hello world"

3.2 Time analysis

Write a MATLAB code named `part3.m` to answer the following questions. Your code should begin with the following lines:

part3.m

```
close all; % Resets matlab workspace
close all; % Close every open figure
clc; % clear the command window

%% Initialisation of some variables

%% Start of your processing
% Read the samples from HelloWorld.wav
[x, Fs] = wavread('HelloWorld.wav');
```

Question 1. Let $x_1(t)$ be $x(t)$ restricted to $t \in [0.01\ s, 0.04\ s]$. Plot $x_1(t)$. Is this part of $x(t)$ voiced or unvoiced?

Question 2. Let $x_2(t)$ be $x(t)$ restricted to $[0.2\ s, 0.23\ s]$. Plot $x_2(t)$. Is this part of $x(t)$ voiced or unvoiced?

Question 3. Plot the signal $x_H(t)$ corresponding to "Hello". You can play a sound using the command `sound`. For example `sound(x,Fs)` will play the sound x ("Hello world"). On the same figure, show were the phonemes are located.

3.3 Frequency analysis

Question 4. Use the `fft` function to plot the magnitude spectrum of the signal of $x_1(t)$ and $x_2(t)$. The frequency axis should be labeled with frequencies expressed in Hz with $f \in \left[-\frac{F_s}{2}, \frac{F_s}{2}\right[$. What are the differences between the magnitude spectra of x_1 and x_2?

3.4 Short Term Fourier Transform and Spectrogram

Speech signal varies slowly, indeed it is stationary for periods of 20 to 30 milliseconds. The Fourier transform provide the frequency content of the speech signal; however it does not capture the localization in time. Consequently, we will analyze $x(t)$ signal using a sliding window of duration 20 to 30 milliseconds. In particular, computing the Discrete Fourier Transform (DFT) of each window captures the frequency content of the windowed signal and how it changes with time. This operation is called **Short Term Fourier Transform** (STFT). Mathematically, the STFT is expressed as

$$X(m, \nu) = \sum_{n=0}^{L-1} x_n w_{n-m} e^{-j2\pi\nu n}. \tag{1}$$

One can show that the computation of the STFT for $m = 0, d, 2d, \ldots$ can be performed following these three steps:

1. decompose the vector x (of length L) into $M = \lfloor \frac{L}{d} \rfloor$ frames of N samples. N has to be chosen so that $\frac{N}{F_s} \in [20, 30]$ milliseconds. Each frame will be stored as columns. The first frame contains elements $(x_0, \dots x_{N-1})^T$. The second frame contains elements $(x_d, \dots x_{d+N-1})^T$ so that it overlaps the first frame on the $N - d$ last elements. Figure 7 illustrates this processing.

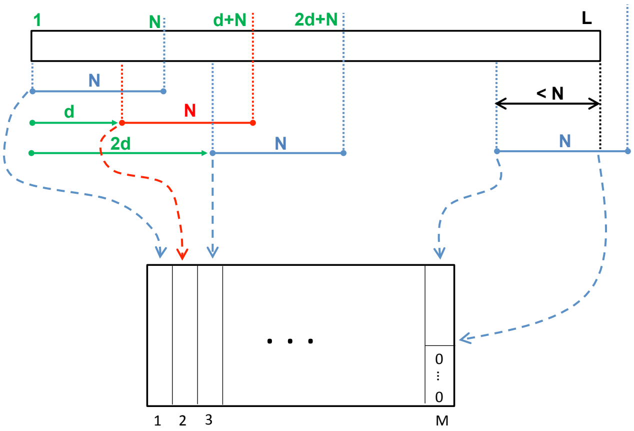


Figure 7: Framing of x_n.

2. multiply each column of the matrix created in step 1 by a window $w = (w_0, w_1, \dots w_{N-1})$. The window signal w_n can be rectangular, triangular, Hamming, or Hanning, however in this work, we will only consider rectangular or Hamming window. Figure 8 illustrates this processing.

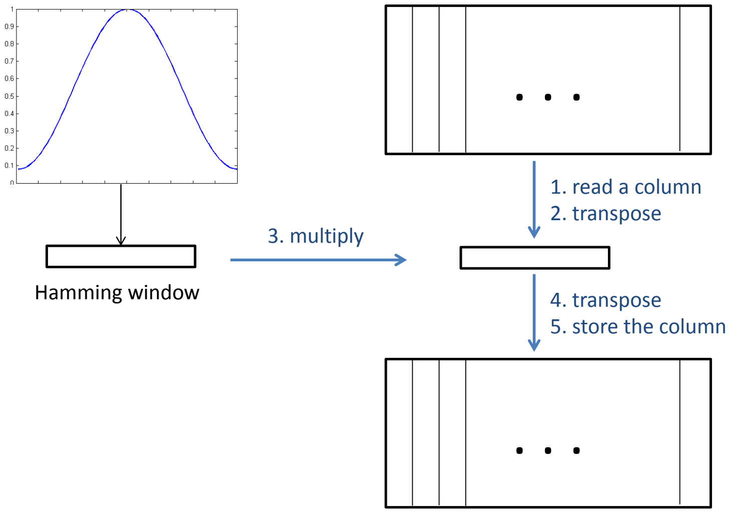


Figure 8: Windowing of x_n.

3. Compute the DFT on N_{fft} points of each row using `fft`. We recall that if x has a length N, `fft(x)` computes the DFT at the frequencies $\nu = (0, \frac{1}{N}, \dots \frac{N+1}{N})$, where $\nu = \frac{f}{F_s}$. On the other hand, if x has a length N, `fft(x,Nfft)` computes the DFT at the frequencies $\nu = (0, \frac{1}{N_{fft}}, \dots \frac{N_{fft}-1}{N_{fft}})$, where $\nu = \frac{f}{F_s}$

Question 5. Create a function `stft` that performs the STFT of the signal x_n. This function will have the following header

stft.m

```
function [X, f, t] = stft(x,w,d,N_fft,Fs)
% This function computes the stft for m = [0, d, 2d, 3d...]
% This function outputs are:
% -> X, which is  a matrix of n_fft lines and M columns
%    M is the number of elements of m
%    X(i,j) is the value of the spectrogram for time t(i) and frequency f(j)
% -> f, is a column vector of the frequencies (in Hz)
% -> t, is a row vector containing the times of the beginning of the windows
```

The modulus squared of the STFT is called the spectrogram

$$S_x(m,\nu) = \frac{1}{N} \left| X(m,\nu) \right|^2 . \tag{2}$$

Question 6. Create a function `spectro` that computes the spectrogram of x_n.

spectro.m

```
function [Sx, f, t] = spectro(x,w,d,N_fft,Fs)
% This function computes the spectrogram for m = [0, d, 2d, 3d...]
% This function outputs are:
% -> Sx, which is  a matrix of n_fft lines and
%                               M (number of elements of m) columns
%    Sx(i,j) is the value of the spectrogram for time t(i) and frequency f(j)
% -> f, is a column vector of the frequencies (in Hz)
% -> t, is a row vector containing the times of the beginning of the windows
```

Question 7. Plot the following spectrograms for the "hello world":

- $N = 441$, $d = 441$, $N_{fft} = 1024$, $w = hamming(N)$,
- $N = 441$, $d = 441$, $N_{fft} = 1024$, $w = ones(1, N)$,
- $N = 882$, $d = 441$, $N_{fft} = 1024$, $w = hamming(N)$.

Analyze the differences between these three spectrograms.

4 Feature extraction

In this section we focus on the extraction of features from a signal $x_i = (x_i[0], x_i[1] \dots x_i[N-1])$, of $30ms$ of the speech signal x. These features are computed to be used by the classifier to determine which word was spoken. We will consider the following features

- the voiced/unvoiced flag,
- the pitch,
- the Mel Frequency Cepstrum Coefficients (MFCC),

These features will be presented and computed in the rest of this section. Create a new main script named `part4.m` to answer these questions.

4.1 Voiced/Voiceless flag and pitch

The *voiced/unvoiced flag* indicates whether the signal is voiced or unvoiced. This flag is equal to '1' if the signal considered is voiced, and equal to '0' otherwise. When the signal is voiced, this signal is periodic with period T; the *pitch* is defined as $p_i = \frac{1}{T}$.

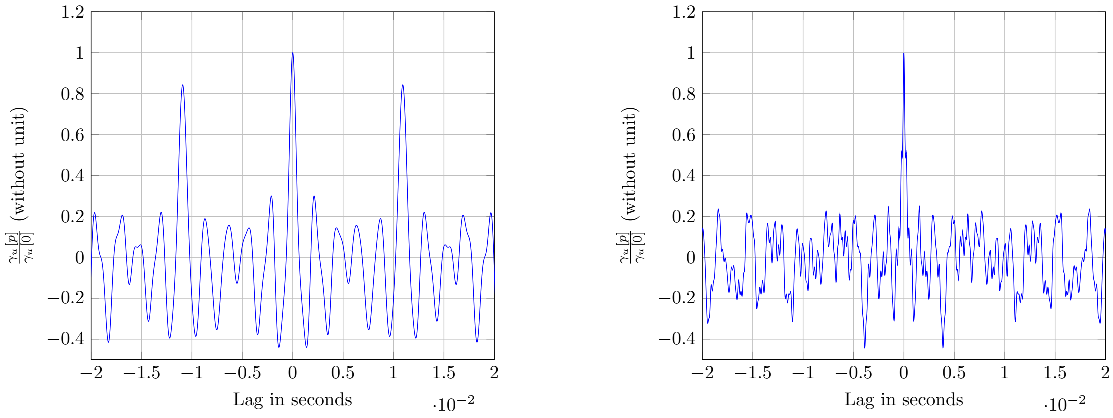
To evaluate whether $x_i[m]$ is voiced or unvoiced, we compute its auto-correlation using the following unbiased estimator

$$\gamma_u[p] = \begin{cases} \frac{1}{N-p} \sum_{k=p}^{N-1} x_i[k]x_i^*[k-p] \text{ for } p = 0, 1, \dots N-1, \\ \gamma_u[-p]^* \text{ for } p = -1, \dots -N+1. \end{cases} \tag{3}$$

$\gamma_u[p]$ has a local maximum in $p = 0$ furthermore, if $x_i[m]$ exhibits some periodic properties with period P samples, the $\gamma_u[P]$ will also be a local maximum; whereas if $x_i[m]$ behave as a noise, it will not be the case (see Figure 9a and Figure 9b). Let P be defined as

$$P = \underset{p \in [\frac{F_s}{500}, \frac{F_s}{50}]}{\text{argmax}} \; \gamma_u[p], \tag{4}$$

where argmax stands for the position of the maximum. We will consider that x_i is voiced if $\gamma_u[P] \geq 0.6\gamma_u[0]$ and that x_i is unvoiced if $\gamma_u[P] < 0.6\gamma_u[0]$. If x_i is voiced, the pitch is defined as $p_i = \frac{F_s}{P}$, if x_i is unvoiced, $p_i = 0$.



(a) Auto-correlation of the signal given in Figure 6a for lags $\frac{p}{F_s} \in [-20, 20]$ ms.

(b) Auto-correlation of the signal given in Figure 6b for lags $\frac{p}{F_s} \in [-20, 20]$ ms.

Figure 9: Auto-correlation functions of signals of Figure 6.

Question 8. Create a function `autocorr` that computes the autocorrelation of $x_i[m]$ from equation (3).

autocorr.m

```
function [gam_x, p] = autocorr(xi,maxP)
% This function computes the autocorrelation estimator of equation (3)
% for lags p = [-maxP+1, ..., 0, 1, ..., maxP-1]
%
% This function returns:
% -> gam_x, which is  a vector of lenght 2*maxP-1 elements
%     containing the autocorrelation values
% -> p is a vector containing the different lags at which
%     the autocorrelation is computed
```

Question 9. Create a function isvoiced that computes the voiced/unvoiced flag and the pitch.

isvoiced.m

```
function [voicedFlag, pitch] = isvoiced(x,Fs)
% This function the voiced/unvoiced flag and the pitch
% This function returns:
% -> voicedFlag = 1 if x is voiced
%    voicedFlag = 0 if x is unvoiced
%
% -> pitch contains the pitch in Hz
```

Question 10. Compute the pitch features for each frame of the signals one1, one2, two1, and two2. Can these coefficients be used for the application proposed in this work.

4.2 MFCC

Mel-Frequency Cepstrum Coeficients (MFCC) are the cepstral parameters mostly employed for speech classification and recognition. They are based on a logarithmic scaling of the frequency abscissa called the *mel* scale. The mel scale is given by

$$m_{mel} = 2595 \log(\frac{f_{Hz}}{700} + 1). \tag{5}$$

This mel scale can be interpreted as a model of perception of sound by the cochlea.

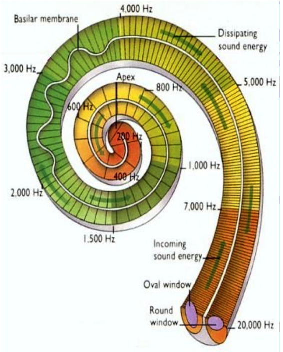


Figure 10: Physical model of the cochlea.

The cochlea is a portion of the inner ear that informs the brain about the presence of certain frequencies. Vibrations occur in different spatial locations (along its length) depending on the frequencies of the incoming sound; a vibration is proportional to the energy localized around the respective frequency. Because of the structure of the cochlea (see Figure 10), the distinction between two closed frequencies is difficult, especially for high frequencies. Consequently, we will consider that the cochlea acts as a filter bank. As it is shown in Figure 11, each filter of this filter bank has a triangular frequency response.

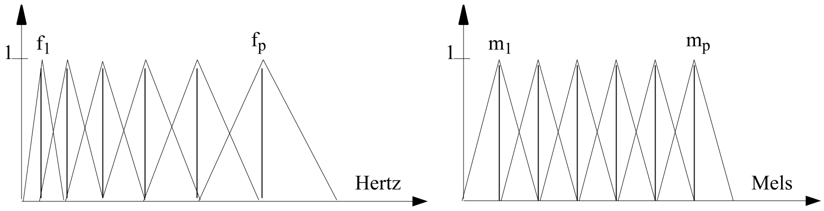


Figure 11: Mel scale and filter bank

The computation of the MFCC features is obtained using the signal processing chain given in Figure 12. $MF(m,r)$ values are computed according to Figure 13. MFCC coefficients are computed as the output of the following Discrete Cosine Transform (DCT)

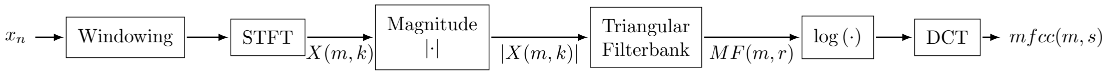


Figure 12: Signal processing for computing MFCC coefficients

$$mfcc_i = \sqrt{\frac{2}{P}} \sum_{j=1}^{P} \log(E_j) \cos(\frac{\pi}{P} i (j - 0.5))$$

where i represents the ith MFCC coefficient with $i \in [1, N_{mfcc}]$.

The function computing the coefficients of the filter bank is called `compute_filter_bank`.

Question 11. Create a function `mfcc` that computes the MFCC coefficients. This function will have the following header.

mfcc.m

```
function [mfcc] = mfcc(x,w,d,Nfft,N_{mfcc})
% This function returns the mfcc features
```

At this point, you should obtain a matrix $N_{mfcc} \times M$ containing N_{mfcc} MFCC coefficients for the M frames. The length of spoken words changes with the speaker and with the pronounced word, hence M can change. Consequently, we will not keep all MFCC coefficients but averages the N_{mfcc} coefficients on the M frames.

Question 12. Create a function `mfcc_features` that computes the average MFCC. This function will have the following header.

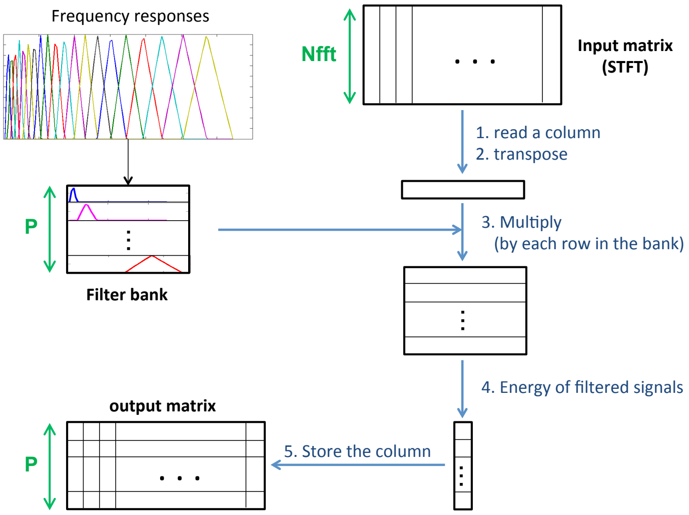


Figure 13: Computation of MFCC coefficients

```
                              mfcc_features.m
function [mfcc_feat] = mfcc_features(x,w,d,Nfft,N_{mfcc})
% This function returns a vector of the averaged P mfcc features
```

Question 13. For this question and for the rest of this work, we consider the following set of parameters

- N is such that the duration of the window is 30ms,
- d is such that the frame shift is 15ms,
- $P = 20$,
- $N_{mfcc} = 20$,
- we will not consider the first MFCC coefficient.

Compute MFCC features for the signals `one1`, `one2`, `two1`, and `two2`. Can these coefficients be used for the application proposed in this work.

5 Classification

Create a new main script named `part5.m` to answer questions of this part. Now that we can extract features from the classifier, we will use these features to recognize isolated spoken words. The role of the classifier is to perform this word recognition. **In this section features will only consider the last 11 MFCC coefficients when computing $N_{mfcc} = 12$ MFCC coefficients**.

We consider a K-Nearest Neighbors (KNN) method for classifying signals. This method is divided into two steps:

- **the train step**: we consider a set of known data named *train set*. The train set is used to learn the model parameters. The class of each signal of the train set is known. In this step we have to extract the features from data in the train set. These features will be used by the test step.
- **the test step**: we consider an other set of data named *test set*. In this step, we compare features extracted from a signal of the test set $\mathbf{f}$ to every feature extracted from the train set $\mathbf{f}'$. To measure similarity between $\mathbf{f}$ and $\mathbf{f}'$ the distance $d(\mathbf{f}, \mathbf{f}') = ||\mathbf{f} - \mathbf{f}'||_2$ is proposed; where the norm $||\mathbf{f}||_2$ is defined as the Euclidian norm: $||\mathbf{f}||_2 = \sqrt{\sum_k |f_k|^2}$.

The KNN algorithm is summarized in Algorithm 1, with notation

- x is a signal among the test set signals,
- x' is a signal among the train set signals,
- $\mathbf{f}$ is the feature extracted from x,
- $\mathbf{f}'$ is the feature extracted from x',
- the set $KNN(\mathbf{f})$ is the set of $\mathbf{f}'$ containing the K nearest neighbors of $\mathbf{f}$.

Algorithm 1 KNN algorithm

```
Require: $\mathbf{f}'$ for all elements in the train set, $\mathbf{f}$, and $K$
  for all $x'$ in the train set do
    Compute $d(\mathbf{f}, \mathbf{f}')$
  end for
  for all $x' \in KNN(\mathbf{f})$ do
    Count the number of elements of $KNN(\mathbf{f})$ belonging to the same class as $x'$.
  end for
Output: $x$ belongs to the most represented class within $KNN(\mathbf{f})$.
```

For testing the KNN algorithm, we provide two data files `data_1.mat` and `data_2.mat`, both containing speech signals of two people saying numbers from 0 to 30. We also provide four others files containing the train and test sets `train_1.mat`, `test_1.mat`, `train_2.mat`, and `test_2.mat`. The `data_1.mat` file contains a unique speech signal called `data_1` and the sampling rate of this signal F_{s1}. The file `train_1.mat` provides a matrix called `train_1` that will be used to extract the training data from `data_1`. `train_1` is a $3 \times N_{train1}$ matrix. Each column $\mathbf{c}$ of this matrix describes an element of the train set as follows

1. the first element of $\mathbf{c}$ contains the class of the element,
2. the second element of $\mathbf{c}$ contains the the start index for the corresponding x in `data_1`,
3. the third element of $\mathbf{c}$ contains the stop index for the corresponding x in `data_1`.

The file `test_1.mat` provides `test_1` is a $3 \times N_{test1}$ matrix, that is structured exactly as `train_1`. The file `train_2.mat` and `test_2.mat` are build similarly.

5.1 Train the classifier

Question 14. Create a function `train_classifier` which takes as inputs `data`, `train` that trains the classifier.

train_classifier.m

```
function [matF] = train_classifier(data,train)
% This function returns matF, a matrix contaning features
% for every element of the train set.
```

Use this function to identify the model corresponding to the elements of `train_1` and `train_2`.

5.2 Test the classifier

Question 15. Use `test_1`, and `test_2` to test the classifier. For each element x of the test set, use the KNN algorithm to estimate the class of x. Compute the probability of success for the classification as follows

$$P(success) = \frac{\text{Number of correctly classified } x \text{ from the test set}}{\text{Total number of elements in the test set}}. \tag{6}$$

Question 16. Use `test_1`, and `test_2` to compute the confusion matrix **Cm**. Elements $Cm(i,j)$ of this matrix are defined as

$$Cm(i,j) = P(x \text{ is classified in the class } j|x \text{ belongs to the class } i) \tag{7}$$

Plot the confusion matrix you obtain as an image (see the example given in Figure 14).

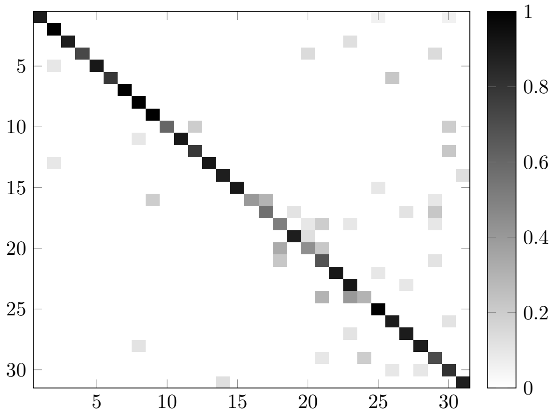


Figure 14: An example of confusion matrix with success probability of 0.79.

Question 17. How to compute $P(success)$ from the confusion matrix

6 Improved versions of the classifier

Create a new main script named `part6.m` to answer questions of this part. In this section we adapt the classifier proposed in the previous section so that it works in real-time.

Question 18. Add your voice to the classifier. To do so, look at the structure of the `data`, `test` and `train` files to create your own files.

Question 19. Train and test this new classifier.

Question 20. Why did you have to add your voice to the classifier first?

Question 21. Write a Matlab program called `real_time_classification` that performs the following tasks

1. record 2 seconds of signal
2. find if there are spoken words using an energy threshold
3. extract these words
4. classify these words
5. display the sign image corresponding to the classified word
6. go back to step 1

7 Contacts

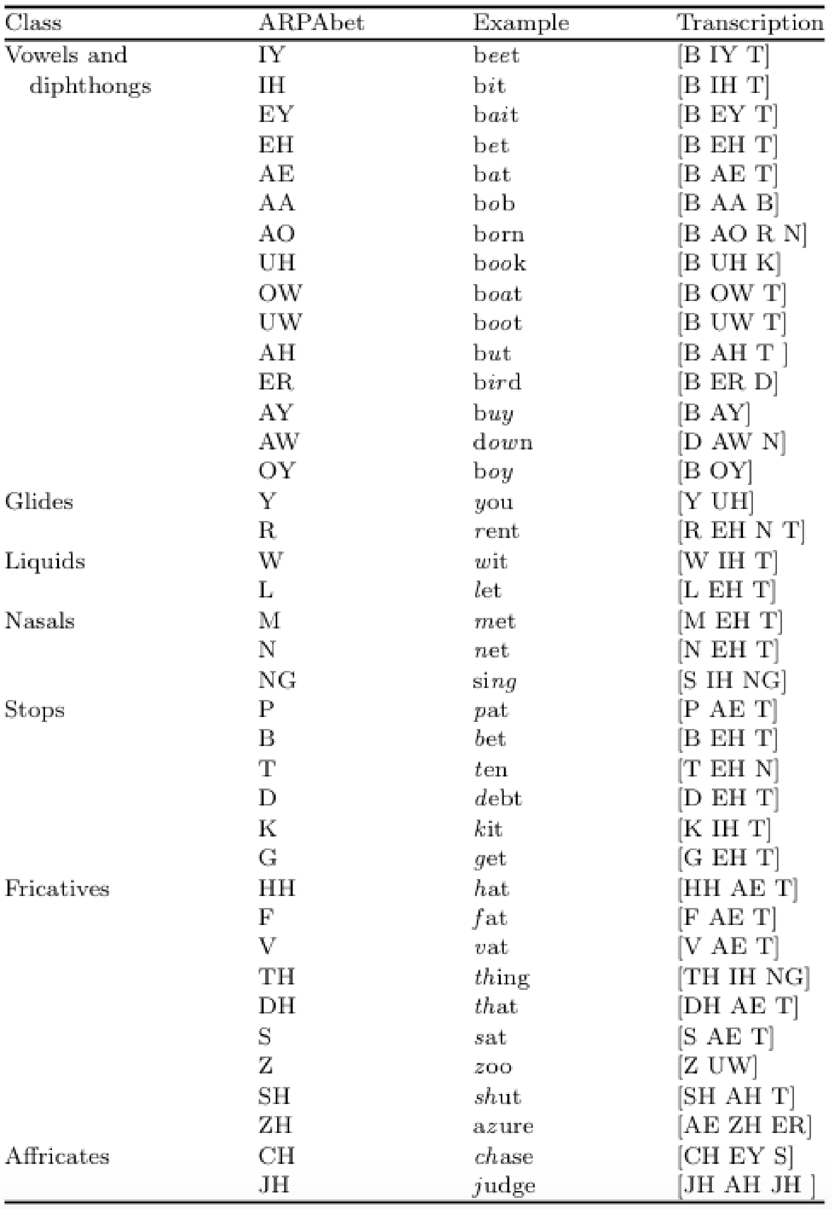
- Pedro Caio Castro Cortes C Coutinho - pedro.castro_cortes_c_coutinho@bordeaux-inp.fr
- Thomas Lorblanchès - thomas.lorblanches@wanadoo.fr
- Romain Tajan - romain.tajan@ims-bordeaux.fr

References

[1] Dennis H. Klatt. Review of the ARPA Speech Understanding Project. *The Journal of the Acoustical Society of America*, 62(6):1345–1366, 1977.

[2] Lawrence R. Rabiner and Ronald W. Schafer. Introduction to digital speech processing. *Foundations and Trends® in Signal Processing*, 1(1–2):1–194, 2007.

Appendix A Table of phonemes from the ARPAbet



Class	ARPAbet	Example	Transcription
Vowels and diphthongs	IY	b*ee*t	[B IY T]
	IH	b*i*t	[B IH T]
	EY	b*ai*t	[B EY T]
	EH	b*e*t	[B EH T]
	AE	b*a*t	[B AE T]
	AA	b*o*b	[B AA B]
	AO	b*o*rn	[B AO R N]
	UH	b*oo*k	[B UH K]
	OW	b*oa*t	[B OW T]
	UW	b*oo*t	[B UW T]
	AH	b*u*t	[B AH T]
	ER	b*ir*d	[B ER D]
	AY	b*uy*	[B AY]
	AW	d*ow*n	[D AW N]
	OY	b*oy*	[B OY]
Glides	Y	*y*ou	[Y UH]
	R	*r*ent	[R EH N T]
Liquids	W	*w*it	[W IH T]
	L	*l*et	[L EH T]
Nasals	M	*m*et	[M EH T]
	N	*n*et	[N EH T]
	NG	si*ng*	[S IH NG]
Stops	P	*p*at	[P AE T]
	B	*b*et	[B EH T]
	T	*t*en	[T EH N]
	D	*d*ebt	[D EH T]
	K	*k*it	[K IH T]
	G	*g*et	[G EH T]
Fricatives	HH	*h*at	[HH AE T]
	F	*f*at	[F AE T]
	V	*v*at	[V AE T]
	TH	*th*ing	[TH IH NG]
	DH	*th*at	[DH AE T]
	S	*s*at	[S AE T]
	Z	*z*oo	[Z UW]
	SH	*sh*ut	[SH AH T]
	ZH	a*z*ure	[AE ZH ER]
Affricates	CH	*ch*ase	[CH EY S]
	JH	*j*udge	[JH AH JH]