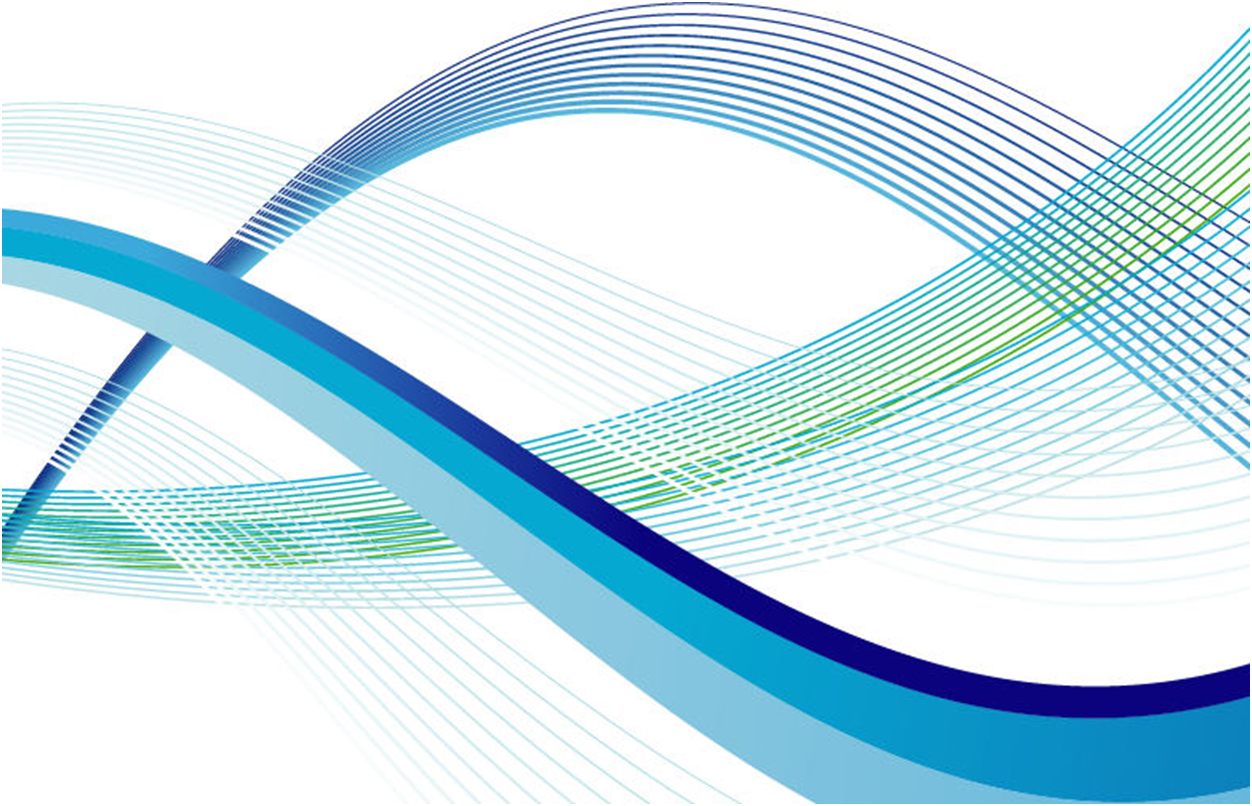
|  |
| --- |
| Polytech’ Nice Sophia Antipolis |
| Word recognition |
| Guénon Marie, Achard Jean-Paul, Favreau Jean-Dominique |
|  |
| **Guénon Marie, Achard Jean-Paul, Favreau Jean-Dominique** |
| **24/1/2013** |

|  |
| --- |
|  |



Contents

[Introduction 3](#_Toc345579926)

[Intended plan: 4](#_Toc345579927)

[Sound treatment 5](#_Toc345579928)

[1. Sound recording 5](#_Toc345579929)

[2. Time split 5](#_Toc345579935)

[3. Spectrogram 6](#_Toc345579936)

[Hamming window 6](#_Toc345579937)

[Fast Fourier transformation 6](#_Toc345579938)

[Redundancies elimination 7](#_Toc345579939)

[Amplitudes weighting 7](#_Toc345579940)

[Mel scale and filter bank 7](#_Toc345579941)

[Spectrogram reconstruction 8](#_Toc345579942)

[Learning and comparison 10](#_Toc345579943)

[1. Existing methods 10](#_Toc345579944)

[Learning 10](#_Toc345579945)

[Conclusion 11](#_Toc345579946)

# Introduction

# Intended plan:

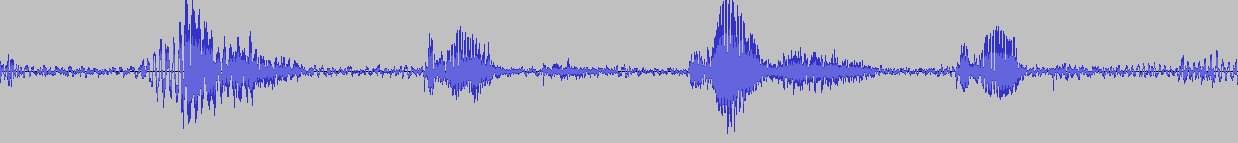
* Sound treatment
  + Sound recording (complete word)
  + Auto-correlation
  + Time split in slices of 20/30 ms, for all times
  + Fourier transformation for all slices 🡺 put together in a spectrogram (matrix)
* Learning (problem of Bakis / Hidden Markov/ Dynamic Time Warping)
  + Frequencies scaling: 20 significant points (Mel scale 🡺 assign importance to some frequencies)  
    Not compare the 1st with the 1st, it could exist some needed translations or insignificant isolated elements.
  + Comparison: « compare most similar words », eventually if we have a great vocabulary, begin with a leak sort.  
    ex : in French, 6/10  
    S-I-S 🡨 delete the similar parts  
    D-I-S  
    ↑compare relevant part
* HMI
  + Graphic  Interface: display syllable (and the spectrogram)
  + « Purchase » Learning : buttons « ok »/ « not ok » and we put the syllable in the database
  + Labyrinth  « game »: we move with our voice  
    / !\ learning the words the gamer will use during the « game »

# Sound treatment

## Sound recording

Choice of the driver to use, choice of the mike

When we are waiting for a response, we start the sound recording and it keeps going in continues, we remove the irrelevant parts (without speech or noise)



## Time split

Time is split in slices of approximately 30 ms:   
For all time t, we take a signal’s slice which extends from to.

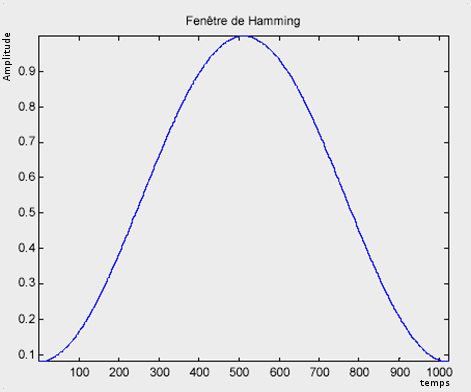
However, the fast Fourier transformation needs a number of points which are a power of 2 in order to perform faster. We chose to take signal’s slices of size n (with n samples), where n is a power of 2 and so that “n\*sampling time” is the closest to 30ms.

We can easily see here that the last signal’s slice will not necessarily has the same size as the others, that’s why we complete the end of the signal with 0 and so have a signal that has a size which is a multiple of the power of 2 considered.

## Spectrogram

### Hamming window

Once the signal is split in slices, we apply to it a Hamming window in order to avoid big discontinuities on the borders, and so avoid inconsistent results with the Fourier transformation.

Formula of the Hamming window:

Where T is the signal’s duration of the segment studied

### Fast Fourier transformation

We are now searching to apply to each signal’s slice the Fourier transformation, which is written:

Where

However, in a goal of speed, we seek to use the fast Fourier transformation, which is written:

For where

For

### Redundancies elimination

Any Fourier transformation is periodic a symmetrical in 0. Since it is not interesting to study the same thing twice (loose of time), we can keep only a signal’s half on which will be made the comparisons.

### Amplitudes weighting

Here we seek to minimize the importance of low frequencies. In fact, the auto-correlation has not removed the whole noise present in the recording and this noise parasitizes the low frequencies.

logarithmic scale to weighting to minimize the   
reduce the dynamic importance of low frequencies

Where

### Mel scale and filter bank

#### Mel scale

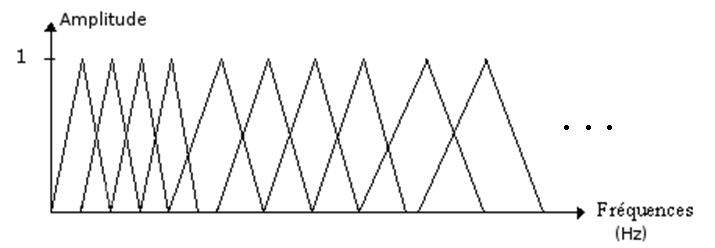
The Mel scale is intended to reduce the importance of high frequencies that are less finely perceived by the human ear. In fact, the perception intensity of a stimulus does not linearly increase as a function of its power, but exponentially.  
The Mel scale allows moving from the frequency of the input signal (in Hz) to a frequency (in Mel) more representative of the human hearing with the following formula:

Where F is the input signal frequency at the point considered.

This gives the following curve:

#### Filter bank

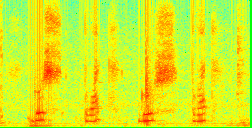
The filter bank serves to reduce the number of frequencies considered in 20 possible values while respecting the Mel scale.  
Yet the use of this unit is not enough. In order to have a relative bandwidth which remains constant, the filter is built with triangular filters uniformly positioned on the Mel scale and so, non uniformly on the frequency scale.  
This gives us the following curve:



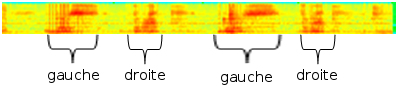
### Spectrogram reconstruction

The reconstitution of the spectrogram consists of grouping together in a matrix the whole Fourier transformations obtained and transformed for each signal’s instant. The rows of the matrix represent the changes in frequency, the columns the changes in time, and the value of each box represents the amplitude of the Fourier transform on the time and frequency given.

Spectrogram obtained before filtering with the Mel scale:



The same spectrogram obtained after filtering with the Mel scale:



We can easily see here the correspondence between the two spectrograms. Moreover we can see that two identical words have much the same form on the spectrogram.

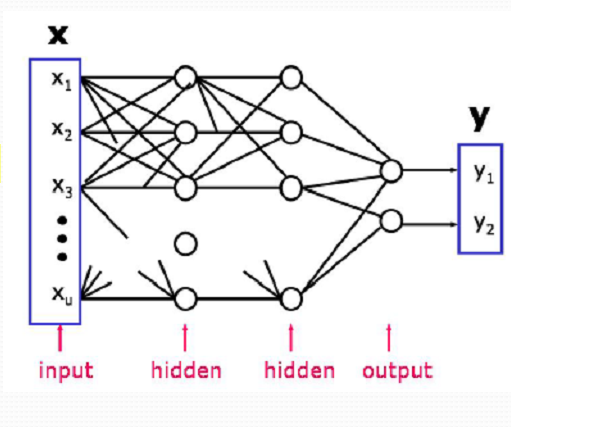
# Learning and comparison

## Existing methods

### Learning

#### Artificial neuronal networks

A neuronal network is generally composed of a succession of layers. Each layer is composed of Ni neurons, having their inputs in the Ni-1 neurons of the previous layer.  
Each synapse (connection between two neurons) is associated with a synaptic weight, so that the  
Ni-1 are multiplied by this weight, and then summed by the neurons of level i.   
Put the layers of a neuronal network one behind the other tantamount to cascade several transformation matrices and could be reduced to a single matrix, produced by the others.

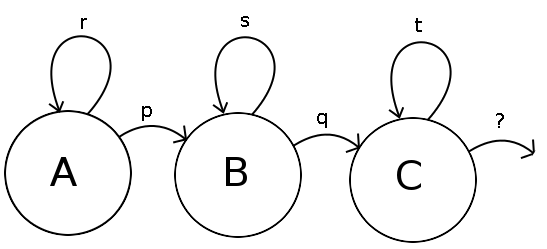


For this learning system, the user does not need to adapt his speech and it is not necessary to train the neural network specifically to the user's voice.  
However, this method requires large learning database with representative cases that will be often encountered. Moreover in the case of extreme elements, they alter the weight given to each possibility, making the division of the possibilities’ space unstable.

In our case, we have a small vocabulary (about twenty words) based on a single-speaker (considering that during a phase of recording a single person has to speak). That’s why it is not necessary here to use a neural network that is heavier to implement (on a computational point) and it suffices to change the comparison database for each new user.

#### Hidden Markov model

A Markov chain is an automaton which contains a number of states, and move at any time from one state to another with a certain probability.  
In our example, we have 3 states: A, B and C. If at time I, the automaton is in state B, it can move to state C with the probability q or remain in state B with the probability s. (Note here that r+q=1, s+q=1 and t+?=1)  
In our case, we removed the possibility of going back because the time is linear and A, B and C are considered as successive states.

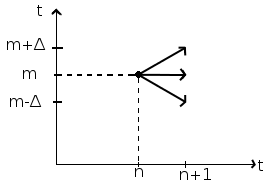


The problem here is to determine the different transition coefficients from one state to another (p, q, r, s, t), and then freely apply this algorithm on word recognition.

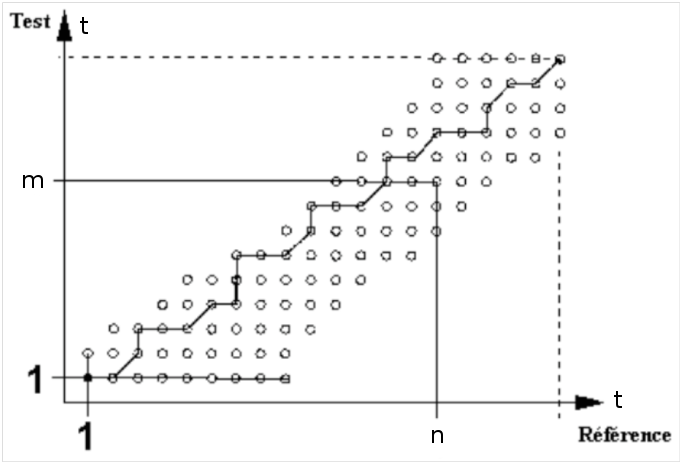
This training can be based on the forward-backward algorithm:  
It starts by calculating the balance of probabilities "forward", which give the probability of obtaining the first k observations in a given sequence, ending in each possible state of the Markov model. Then it calculates the set of probabilities "back", which represents the probability of the other cases when an initial state is given.  
Both sets of probabilities can be combined to obtain the probability of being in each state at a given time during the observation of the sequence.

However, it is not interesting in this case to use the hidden Markov model, because it is too complex for the small vocabulary we are using. So it is sufficient for us to use the Dynamic Time Warping which is a simplification of the hidden Markov model.

Comparison: Dynamic Time Warping (DTW)   
  
The principle of DTW is to determine for each element of a sequence, the best matching element in the other sequence.  
In our case, the point is to compare, using a Euclidean distance, the signal’s spectrum measured with those of different words composing our vocabulary.  
Comparison principle:   
On each iteration, we compare the nth signal’s spectrum measured with the reference spectrums from m-Δ to m+Δ and keep the mth nearest spectrum.  
The displacement is of the form:



Where the arrows represent the possible moves.  
Every successive movement put together, we obtain the minimal path between two sequences.  
Here an example of a representation of the shortest path between a sequence tested and the reference sequence:



Once the distance with every word established, you can now get the smallest of them that correspond to the reference word detected.

# Conclusion