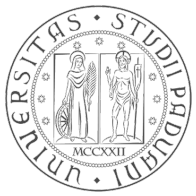


Università degli Studi di Padova

Dipartimento di Ingegneria

Corso di Laurea in Ingegneria dell'Informazione

A Strategy for Noise Reduction in Speech Recordings from Smartphones and Tablets

Laureando **Marco Ancona**

Relatore **Leonardo Badia**

Anno Accademico 2012/2013

Contents

Abstract

The aim of this work is to analyse the performance of Apple's iPhone and iPad as voice recorders, while at the same time finding algorithms to enhance speech recordings and reduce the noise introduced by the low quality built-in microphone. We perform spectral analysis of silent recordings to acquire the noise print from different device models. Comparing these results, we assess whether or not different iPhone devices can be modelled with the same noise source. We also propose a speech enhancement algorithm to reduce additive noise introduced during the recording. Finally, we comment on the results and make a few considerations for further developments.

Abstract

Lo scopo di questo lavoro è quello di analizzare le prestazioni di Apple iPhone e iPad come registratori vocali e di cercare un algoritmo per il miglioramento delle registrazioni vocali e la riduzione del rumore introdotto dai microfoni di limitate prestazioni intergrati in essi. Analizziamo alcune registrazioni effettuate in ambiente silenzioso per acquisire l'impronta di rumore dei diversi modelli di smartphones e tablets. Comparando i risultati proviamo che differenti esamplari di uno stesso modello di dispositivo possono essere modellati con la stessa sorgente di rumore. Proponiamo inoltre un algoritmo per la riduzione del rumore additivo introdotto durante la registrazione. Infine, commentiamo i risultati e facciamo alcune considerazioni per sviluppi futuri.

Chapter 1

Introduction

Nowadays smartphones are becoming more and more common in everyday life. Until recently, dedicated devices have been used for taking photos, recording lectures and conferences, listening to music or finding a route. Provided that cameras, voice recorders, music players and GPS navigation devices are still essential tools for those who need professional results and the best efficiency, smarthphones are gradually replacing these devices for everyday needs [1]. According to [2], the majority of smartphone and tablet users say their mobile device has replaced a traditional alarm clock (61.1%), a GPS device (52.3%) and a digital camera (44.3%). Personal planners have been replaced by smartphones (41.6%) as well as landline phones (40.3%). More than a third no longer need a separate MP3 player (37.6%) or a video camera (34.2%). It seems clear that people are more and more willing to compromise on quality just to have all these functionalities merged into a single, portable device.

Nowadays digital voice recorders are at risk of extinction because smartphone apps can do many of the same tasks, provided that high recording quality can be achieved by implementing advanced and efficient noise reduction filters. While hardware should be optimized to reduce thermal noise and provide a wide range frequency response, recording apps should be optimized to reduce residual thermal noise, compensate ambient noise and enhance speech quality (echo and reverb reduction, equalization). Moreover, when dealing with smartphones other factors must be considered. Among these [3, 4]:

- Computational complexity.
- Power consumption.
- Storage limits.

- Interference from other tasks.

For these reasons, we strongly believe that improving secondary features of a smartphone, focusing on both hardware and software, is necessary in order to obtain a good alternative to dedicated devices.

The aim of this work is analysing the performances of Apple's iPhone [5] and iPad [6] as voice recorders, in order to find algorithms to enhance speech recordings and reduce the thermal noise introduced by the low quality built-in microphones. We therefore perform spectral analysis of silent recordings on different device models to acquire noise prints, which are necessary to perform noise reduction through a Wiener filter [7]. We also compare different thermal noise prints between iPhone and iPad models to understand whether or not we can assume no significant difference between devices of the same model. Finally, we evaluate the enhanced speech recordings both through objective and subjective listening tests and we adjust the filter's parameters according to these results.

The rest of this thesis is organized as follows. In Chapter 2 we recall the basics of Fourier Transform, Energy and Power Spectral Density and we introduce the Wiener filter and the the decision-directed method for the *a priori SNR* estimation. In Chapter 3 we provide a brief description of the MEMS microphones commonly embedded into the smartphones, we define the concept of noise print and show our filtering technique based on it. To conclude the chapter, we list the devices and the recordings we used for the tests of our filter. In Chapter 4, we provide the results of the filtering process, discuss the trade off between noise suppression and speech distortion and compare the noise prints of different copies of the same device model. Finally, Chapter 5 contains our conclusions and plans for further work.

Chapter 2

Background

Speech enhancement aims to improve speech quality by using various algorithms [8], e.g. spectral subtraction, spectral enhancement based on hidden Markov processes (HMPs) and subspace methods. The objective of enhancement is improvement in intelligibility and/or overall perceptual quality of degraded speech signal using audio signal processing techniques. The problem of enhancing speech signal degradated by uncorrelated additive noise, when the noisy signal alone is available, has recently received much attention [8, 9, 10] since it has many potential applications. In particular, the great development of mobile communications or hearing devices has made single-channel speech enhancement a very important field of research.

Section 2.1 gives an introduction to Fourier transform and the basic principles of spectral estimation. In Section 2.2 we introduce the basics of noise reduction and define the Wiener filter [7] which will be used in our noise reduction algorithm combined with a model based spectral estimation method.

2.1 Fourier Transform, Power Spectrum and Periodograms

Power spectrum of a signal shows the distribution of the signal power along frequency. It also reveals important information on the correlation structure of the signal.

The Fourier transform of a continuous-time signal $x(t)$ is defined as [7, 11]

$$X(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft}dt \tag{2.1}$$

where $X(f)$ is a complex number, whose amplitude and phase represents

amplitude and phase of the signal at frequency f. The inverse Fourier transform is given by

$$x(t) = \int_{-\infty}^{+\infty} X(f) e^{j2\pi ft} dt \tag{2.2}$$

Since we are dealing with digital signal processing, we can only manage a sampled version $x[n]$ of the original signal $x(t)$. The Discrete-Time Fourier Transform (DTFT) of a sampled signal $x[n]$ can be obtained from (2.1)

$$X(f) = \sum_{-\infty}^{+\infty} x[n] e^{-j2\pi fn} \tag{2.3}$$

It is important to highlight that the spectrum of a sampled signal is periodic with a period $f = 1$

$$\begin{aligned} X(f+1) &= \sum_{-\infty}^{+\infty} x[n] e^{-j2\pi(f+1)n} \\ &= \sum_{-\infty}^{+\infty} x[n] e^{-j2\pi fn} \underbrace{e^{-j2\pi n}}_{=1} = \sum_{-\infty}^{+\infty} x[n] e^{-j2\pi fn} = X(f) \end{aligned} \tag{2.4}$$

The inverse Fourier transform of a sampled signal is defined as

$$x[n] = \int_{-1/2}^{1/2} X(f) e^{j2\pi fn} df \tag{2.5}$$

2.1.1 Discrete Fourier Transform and Fast Fourier Transform

One of the most important reasons behind the success of discrete-time method for the analysis and synthesis of signals was the development of increasingly efficient tools to perform Fourier analysis on digital devices. Actually, the processing of a signal on digital computers requires that both the time-domain signal and its Fourier transform are discrete. This result can be achieved by the *Discrete Fourier Transform (DFT)* [11].

Let $x[n]$ be a signal of finite duration; that is, there is an integer N_1 so that

$$x[n] = 0, \qquad 0 \leq n \leq N_1 - 1 \tag{2.6}$$

Furthermore, consider $X(f)$ the discrete time Fourier transform of x[n] according to (2.3). We can construct a periodic signal $\tilde{x}[n]$, with an integer period $N \geq N_1$, such that

$$\tilde{x}[n] = x[n] \qquad 0 \leq n \leq N - 1 \tag{2.7}$$

and

$$\tilde{x}[n+N] = \tilde{x}[n] \qquad n \geq N. \tag{2.8}$$

The Fourier series coefficients for $\tilde{x}[n]$ are given by

$$a_k = \frac{1}{N} \sum_{<N>} \tilde{x}[n] e^{-j\frac{2\pi}{N}kn} \tag{2.9}$$

Choosing the interval of summation to be that over which $\tilde{x}[n] = x[n]$, we obtain

$$a_k = \frac{1}{N} \sum_{n=0}^{N-1} \tilde{x}[n] e^{-j\frac{2\pi}{N}kn} \tag{2.10}$$

Eq.(2.10) defines the coefficients that comprise the DFT of $x[n]$, defined as

$$\tilde{X}(k) = \sum_{n=0}^{N-1} x[n] e^{-j\frac{2\pi}{N}kn} \qquad k = 0, ..., N-1 \tag{2.11}$$

Comparing (2.3) and (2.11) we see that the DFT differs from the DTFT in that its input and output sequences are both finite. The inverse Fourier transform (IDFT) is given by

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{j\frac{2\pi}{N}kn} \qquad m = 0, ..., N-1 \tag{2.12}$$

A periodic signal has a discrete spectrum and conversely any discrete frequency spectrum corresponds to a periodic signal. Hence, the implicit assumption in DFT is that the signal $x[n]$ is periodic with a period of N samples.

The importance of the DFT stems from the fact that the original finite duration signal can be recovered from its Discrete Fourier Transform. Moreover, a second important feature of the DFT is that there are extremely fast algorithms whose set has collectively come to be known as the *Fast Fourier Transform (FFT)*, an efficient method for the calculation of the DFT of finite-duration sequences [11, 12, 13].

Let us consider the direct evaluation of the DFT expression in (2.11). Since $x[n]$ may be complex, N complex multiplications and $(N-1)$ complex additions are required to compute each value of the DFT directly. Direct computation of all N values has therefore complexity which is $O(N^2)$; thus the number of arithmetic operations required to compute the DFT by direct method becomes very large for large values of N.

Most approaches to improve the efficiency of the computation of the DFT rely on the symmetry and periodicity properties of the complex coefficient $W_N = e^{-j2\pi/N}$ such as

$$\begin{cases} {W_N}^{k(N-n)} = {W_N}^{-kn} = ({W_N}^{kn})^* & \text{(complex coniugate symmetry);} \\ {W_N}^{kn} = {W_N}^{k(n+N)} = ({W_N}^{(n+N)n}) & \text{(periodicity in n and k);} \end{cases}$$

Efficient algorithms for the FFT computation, such as Cooley–Tukey algorithm [14], can return the same result of a direct DFT computation with an overall improvent from $O(N^2)$ to $O(N \log N)$.

2.1.2 Frequency resolution

Assume a signal of length T_0 seconds, sampled by at least the Nyquist rate, producing N samples. Then the sampling interval is $T = \frac{T_0}{N}$, the sampling frequency is $f_s = \frac{1}{T}$ and the highest frequency of the signal is, at most

$$f_{max} = \frac{1}{2T} = \frac{N}{2T_0}. \tag{2.13}$$

The frequency resolution of the DFT spectrum is proportional to the signal length N, and is [7]

$$\Delta f = \frac{f_s}{N} = \frac{1}{T_0} = \frac{1}{NT} \tag{2.14}$$

The discrete Fourier transform gives the values of the amplitude spectrum at frequencies $1/T_0, 2/T_0, ..., N - 1/T_0, N/T_0$. Actually, given the symmetry of the transform, values related to the indices $0 \leq k \leq N/2 - 1$ are for the positive frequencies, while $N/2 \leq k \leq N - 1$ are for the negative ones. In particular, $k = N - 1$ corresponds the frequency $f = -\frac{f_s}{N} = -\frac{1}{NT}$. Nyquist frequency corresponds to index $N/2$, when N is even.

For short length record the spectral resolution is low. However, the spectrum of a short signal can be interpolated to obtain a smoother spectrum. This is normally achieved by *zero-padding* the time domain signal $x[n]$. Consider a signal of N samples $(x[0], ..., x[N - 1])$. Increase the signal length from N to $2N$ samples by adding N zeros to obtain the sequence

$$x[0], ..., x[N-1], \underbrace{0, ..., 0}_{\text{N zeros}}$$

The spectrum of the zero-padded signal consists of $2N$ spectral samples, N of which, $X[0], X[2], X[4], ..., X[2N - 2]$ are the same of those that would

be obtained from the DFT of the original N time domain samples, and the other N samples are the interpolated spectral lines that result from zero-padding. Note that this method does not increase the spectral resolution; it merely has an interpolating, or smoothing, effect in the frequency domain.

2.1.3 Signal energy, Energy Spectral Density and Power Spectral Density

As expressed in the Parseval's theorem [11], the energy of a discrete time, real signal $x[n]$ can be computed either in the time or in the frequency domain as

$$E_x = \sum_{m=-\infty}^{\infty} x^2[m] = \int_{-1/2}^{1/2} |X(f)|^2 \tag{2.15}$$

provided the sum exists and is finite. If the total energy of a signal is a finite non-zero value, then that signal is classified as an *energy signal* [15]. The function

$$\varepsilon_x(f) = |X(f)|^2 \tag{2.16}$$

is called the *Energy Spectral Density (ESD)* [16]. We can also define the *cross energy spectral density* [16] of two signals $x[n]$ and $y[n]$ as $\varepsilon_{xy}(f) = X(f)Y^*(f)$.

Most of the signals encountered in the applications are such that their variation in the future cannot be known exactly. It is only possible to make probabilistic statements about that variation. The mathematical device to describe such a signal is that of a *random process* [16], which consists of an ensemble of possible realizations, each of which has some associated probability of occurrence. Of course, from the whole ensemble of realizations, the experimenter can usually observe only one realization of the signal. However, the realizations of a random signal, viewed as infinite-length discrete-time sequences, are not absolutely summable, and hence, not possess DTFTs. A random signal usually has finite average power and, therefore, it makes more sense to define a *power spectral density (PSD)* [16], which describes how the power of a signal or time series is distributed over the different frequencies. A signal with a finite non-zero average power P_x is classified as a *power signal* [15].

Let us have a single realization of a stochastic process in the time domain, $x[n]$, and consider a finite number N of its samples, $\tilde{x}[n]$. Defining $\tilde{X}(f)$ as the DTFT of $\tilde{x}[n]$, the ESD is found from $\tilde{X}(f)$ by computing the expectation of the squared amplitude spectrum:

$$\varepsilon_x(f) = \mathbb{E}\left[\left|\tilde{X}(f)\right|^2\right] \tag{2.17}$$

As N grows to infinity, so does $\varepsilon_x(f)$. We divide it by the interval length N to curb this growth, which leads to the expression for the PSD [17]

$$S_x(f) = \lim_{N\to\infty} \frac{\varepsilon(f)}{2N+1} = \lim_{N\to\infty} \mathbb{E}\left\{\frac{1}{2N+1}\left|\sum_{n=-N}^{N} x[n]e^{-j2\pi fn}\right|^2\right\} \tag{2.18}$$

For the sake of completeness, we also highlight that many authors define the PSD as the Fourier transform of the signal autocorrelation function (e.g. in [7, 16]). Actually, this definition is a consequence of the important result of the Wiener-Khinchin theorem and its equivalence with (2.18) can be proved under weak conditions [18, p. 7].

2.1.4 Power spectrum estimation using Periodograms

In real-world application, the PSD can only be estimated from an N sample record. A number of methods have been proposed for the spectrum estimation; here we focus on *non-parametric methods*, where the PSD is estimated directly from the signal itself. The simplest of such methods is the *periodogram*, introduced by Sir Arthur Schuster in 1898 [7, 19]. The periodogram can be defined as [7]

$$\hat{S}_x(f) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j2\pi fn}\right|^2 = \frac{1}{N}|X(f)|^2 \tag{2.19}$$

Note that the periodogram definition is very similar to (2.18), except for the facts that we are now dealing with a finite-length signal and we had to drop the expectation operator since we have only one realization of the process. Due to finite length and random nature of most signals, the spectra obtained from different records of a signal vary randomly over the average spectrum. As the record length N increases the expectation of the periodogram converges to the power spectrum $S_x(f)$ and the variance of $\hat{S}_x(f)$ converges to $[S_x(f)]^2$. Hence the spectrogram is unbiased but not a consistent estimate.

A number of methods have been developed to reduce the variance of the spectrogram. One such technique to solve the variance problems is also known as the method of averaged periodograms or *Bartlett's method* [7]. The idea is to divide the set of N samples into L sets of $N_0 = N/L$ samples,

compute the DFT of each set, square it to get the power spectral density and compute the average of all of them.

Another important method, which we use in the development of our noise reduction filter, is *Welch's method* [20]. This is an improvement of the standard periodogram spectrum estimating and the Bartlett's methods, in that it reduces noise in the estimated power spectra in exchange for reducing the frequency resolution. Due to the noise caused by imperfect and finite data, the noise reduction from Welch's method is often desired. As with Barlett's method, a signal $x[n]$, of length N samples, is divided into K sets of length M. However, the idea behind Welch's method is that the segments are partially *overlapping* and each segment is *windowed* prior to computing the periodogram. Then, the Welch power spectrum is computed as the average of the K periodograms. The window function alleviates the discontinuities and reduces the spread of the spectral energy into the side lobes of the spectrum.

2.2 Noise reduction

Noise is inevitable in all applications that are related to voice and speech, thus the signal of interest that is picked up by a microphone is generally contamined by noise and has to be cleaned up with digital processing tools before it is stored, analyzed, transmitted, or played out. The observed microphone signal can be modeled as a superposition of the clean speech and additive noise. The objective of noise reduction, then, becomes to restore the original clean speech when only the mixed signal is available. By and large, the developed techniques for noise reduction can be classified into three categories [8]: 1) filtering technique, 2) spectral restoration and 3) model-based methods. The basic idea behind the filtering technique is to pass the noisy speech through a linear filter. This filter can be designed to significantly attenuate the noise level while leaving the clean speech relatively unchanged. The most important algorithms is this category include Wiener filters and subspace methods [8]. Comparatevely, the spectral restoration technique treats noise reduction as a robust spectral estimation problem, estimating the spectrum of the clean speech from that of the noisy signal. Among this category, the minimum-mean-square-error (MMSE) estimator, the maximum-likelihood (ML) estimator and the maximum a posteriori (MAP) estimator, to name a few [8]. Finally, in the model-based methods, a mathematical model is used to represent human speech production and parameters estimation is carried out in the model space. This category includes harmonic-model-based Kalman filtering approaches and hidden-Markov-model-based statistical methods [8].

Unfortunately, an optimal estimate from signal processing perspective does not necessarily correspond to the best quality according to human ear. The objective of the problem has subsequently been broadened, which can be summarized to achieve one or more of the following primary goals:

1. to improve objective performance criteria such as intelligibility, signal-to-noise ratio (SNR) [16], noise-reduction factor [8], etc,;

2. to improve the perceptual quality of the degraded speech;

3. to increase the robustness of other speech processing (speech coding, echo cancellation, automatic speech recognition, etc.) to noise.

We will focus on the Wiener filter techniques for the development of our noise reduction technique.

2.2.1 Signal model

In many speech applications, a system with a number of inputs and outputs needs to be indentified. For the purpose of this work, we will now consider a single-input single-output (SISO) system since most of modern smartphones have just one microphone that can be used to capture sounds. Actually, a few high-end modern smartphones do have a secondary microphone to perform in-call noise cancellation (e.g. Apple's iPhone 5) but captured audio data from this secondary microphone is usually not accessible for further analysis and noise reduction on third-party applications.

The model used for SISO system is shown in Fig. 2.1. The noise-reduction problem considered in this work is to recover a speech signal of interest $x[n]$ from the noisy observation

$$y[n] = x[n] + b[n] \tag{2.20}$$

where $x[n]$ is the original signal at time n, $b[n]$ is the unwanted additive noise, assumed to be a zero-mean random process (white or colored) and uncorrelated with $x[n]$. In this case, the noise reduction problem is formulated as estimating a cleaned speech signal $\hat{x}[n]$ from the observation $y[n]$.

Applying an N-point DFT at both sides of (2.20), we have the following relationship in the frequency domain

$$Y(m, f_k) = X(m, f_k) + B(m, f_k) \tag{2.21}$$

where

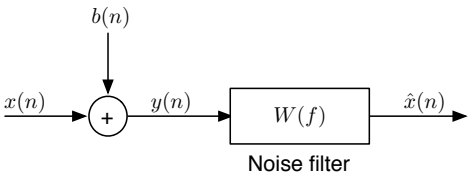


Figure 2.1: Single-input single-output (SISO) system for additive noise reduction

$$Y(m, f_k) = \sum_{n=0}^{N-1} w[n]y[m - N + n + 1]e^{-j\frac{2\pi}{N}kn}, \tag{2.22}$$

is the short-time DFT of the noisy speech at frame m, f_k represents the k^{th} spectral component, $k = 0, 1, ..., N - 1$, $w[n]$ is a window function (e.g. Hamming window) and $X(m, f_k)$ and $B(m, f_k)$ are the short-time DFTs of the clean speech and noise signal, defined similarly to $Y(m, f_k)$.

2.2.2 Wiener filter

The Wiener filter, first proposed by Norbert Wiener during the 1940s and published in 1949 [21], forms the foundation of data-dependent linear least squared error filters. The coefficients of a Wiener filter are calculated to minimize the average squared error distance between the filter output and the desired signal. In its basic form, the Wiener filter theory assumes that signals are stationary and ergodic processes. However, since the filter coefficients can be periodically recalculated, for every block of N samples, then the filter adapts itself to the characteristics of the signal within the blocks and becomes block-adaptive. In particular, in our noise reduction problem, the noise is considered stationary and ergodic, thus it fulfills the assumptions of the theory. On the other hand, the speech signal is not stationary but can ben considered quasi-stationary on frames of length 20-40 ms; so the filter coefficients must be recomputed on each frame [7, 22].

The Wiener filter can be written in both time and frequency domains. We will now focus on the latter, in which each subband filter is independent of the filters corresponding to other frequency bands. The Wiener filter is obtained by minimizing the mean-square error (MSE) between the signal of interest and the spectrum.

Let us consider the signal model in (2.21). The Wiener filter output $\hat{X}(m, f_k)$ is the product of the input signal $Y(m, f_k)$, and the filter frequency response $W(m, f_k)$

$$\hat{X}(m, f_k) = W(m, f_k)Y(m, f_k) \tag{2.23}$$

The estimation error signal $E(m, f_k)$ is defined as the difference between the desired signal $X(m, f_k)$ and the filter output $\hat{X}(m, f_k)$ as

$$E(m, f_k) = X(m, f_k) - \hat{X}(m, f_k) = X(m, f_k) - W(m, f_k)Y(m, f_k) \tag{2.24}$$

The MSE criterion is then written as

$$\begin{aligned} J_x[W(m, f_k)] &= \mathbb{E}\left[|E(m, f_k)|^2\right] \\ &= \mathbb{E}\left[|X(m, f_k) - W(m, f_k)Y(m, f_k)|^2\right] \end{aligned} \tag{2.25}$$

where $\mathbb{E}[]$ is the expectation operator. The frequency-domain subband Wiener filter is derived by the criterion

$$W_o(m, f_k) = \underset{W(m,f_k)}{\arg\min} \quad J_x[W(m, f_k)] \tag{2.26}$$

To obtain the least mean squared error filter, we set the derivative of (2.25) with respect to filter $W(m, f_k)$ to zero

$$\frac{\partial \mathbb{E}\left[|E(m, f_k)|^2\right]}{\partial W(m, f_k)} = 0 \tag{2.27}$$

From this equation we can derive the frequency response of the Wiener filter

$$\begin{aligned} W_o(m, f_k) &= \frac{\mathbb{E}\left[(X(m, f_k))^*(Y(m, f_k))\right]}{\mathbb{E}\left[(Y(m, f_k))^*(Y(m, f_k))\right]} \\ &= \frac{\mathbb{E}\left[|X(m, f_k)|^2\right]}{\mathbb{E}\left[|Y(m, f_k)|^2\right]} \end{aligned} \tag{2.28}$$

Where the last equality follows from the fact that the speech signal $X(m, f_k)$ and the noise $B(m, f_k)$ are uncorrelated, and thus

$$\mathbb{E}\left[(X(m, f_k))^*(Y(m, f_k))\right] = \mathbb{E}\left[(X(m, f_k))^*(X(m, f_k))\right]. \tag{2.29}$$

.

According to (2.19), we can also write

$$W_o(m, f_k) = \frac{S_x^{(m)}(f_k)}{S_y^{(m)}(f_k)} \tag{2.30}$$

where $S_x^{(m)}(f_k)$ is the power spectral density of the m^{th} frame of $x[n]$ and $S_y^{(m)}(f_k)$ is defined in the same way for $y[n]$. It can be seen that the frequency-domain Wiener filter $W(m, f_k)$ is nonnegative and real-valued, therefore it only modifies the amplitude of the noisy speech spectra, while leaving the phase unchanged. We see from (2.30) that, in order to obtain the Wiener filter, we need the PSDs of both the noisy and the original speech signals. The former can be directly estimated from the noisy observation $y[n]$ but $x[n]$ is not accessible. However, exploiting the fact that speech and noise are assumed to be uncorrelated, we have

$$S_y^{(m)}(f_k) = S_x^{(m)}(f_k) + S_b^{(m)}(f_k) \tag{2.31}$$

and hence the Wiener filter can be written as

$$W_o(m, f_k) = \frac{S_x^{(m)}(f_k)}{S_x^{(m)}(f_k) + S_n^{(m)}(f_k)} = \frac{S_y^{(m)}(f_k) - S_b^{(m)}(f_k)}{S_y^{(m)}(f_k)} \tag{2.32}$$

Now we see that the filter depends on the PSDs of both the noisy speech and the noise signals, where the latter can be estimated during the absence of speech. Most of the classical speech enhancement techniques require the evaluation of two parameters, the so-called *a posteriori SNR* and the *a priori SNR*, first proposed by Ephrain and Malah [22] and defined by

$$SNR_{post}(m, f_k) = \frac{|Y(m, f_k)|^2}{\mathrm{E}\left[|B(m, f_k)|^2\right]} \tag{2.33}$$

and

$$SNR_{prio}(m, f_k) = \frac{\mathrm{E}\left[|X(m, f_k)|^2\right]}{\mathrm{E}\left[|B(m, f_k)|^2\right]} \tag{2.34}$$

Dividing numerator and denominator of (2.32) by the noise power spectra $S_b^{(m)}(f_k)$, considering the definition of power spectral density of (2.18) and the definition of SNR_{prio} of (2.34), the Wiener filter can be written as

$$W_0(m, f_k) = \frac{SNR_{prio}(m, f_k))}{SNR_{prio}(m, f_k) + 1} \tag{2.35}$$

From (2.35), we can deduce that, for additive noise, the Wiener filter frequency response is a real positive number in the range $0 \leq W_0(m, f_k) \leq 1$. We can consider two limit cases: at very high SNR ($SNR_{prio}(m, f_k) \to +\infty$), the filter applies little or no attenuation to the noise-free frequency component; on the other extreme, when $SNR_{prio}(m, f_k) = 0$, $W_0(m, f_k) = 0$. Therefore, for additive noise, the Wiener filter attenuates each frequency component f_k in proportion to an estimate of the signal to noise ratio.

In pratical implementations, both the *a priori SNR* and the *a posteriori SNR* have to be estimated, and the quality of the restored speech signal is strongly related to the choice of the estimators. According to (2.33) and (2.34), an estimate of the noise power spectra is necessary to evaluate the *a posteriori SNR*. Moreover, an estimate of the clean speech signal is also necessary for the *a priori SNR*. While the noise power spectra can be estimated from silent frames of the noisy signal $y[n]$, the clean speech signal $x[n]$ is not available at any time. In the simplest solution, by expoiting the fact that $x[n]$ and $b[n]$ are supposed uncorrelated, an estimate of the desired signal power spectra is obtained by subtracting an estimate of the noise spectra from that of the noisy signal, that is $S_x(f_k) = S_y(f_k) - S_b(f_k)$. This leads to the following estimate for the *a priori SNR*

$$\hat{SNR}_{prio}(m, f_k) = \hat{SNR}_{post}(m, f_k) - 1 \tag{2.36}$$

The main drawback of this approach is that the resulting cleaned signal suffers from noise-related fluctuactions in low SNR conditions that lead to a very annoying *musical noise* [9].

Another well-known approach is the *decision-directed* method [22] by Ephraim and Malah according to which the two estimated SNRs are computed as follows

$$\hat{SNR}_{post}(m, f_k) = \frac{|Y(m, f_k)|^2}{\hat{S}_b^{(m)}(f_k)} \tag{2.37}$$

$$\hat{SNR}_{prio}(m, f_k) = \alpha \frac{\left|\hat{X}(m-1, f_k)\right|^2}{\hat{S}_b^{(m)}(f_k)} + (1-\alpha)\max(\hat{SNR}_{post}(m, f_k) - 1, 0) \tag{2.38}$$

where $\left|\hat{X}(m-1, f_k)\right|$ is the estimate of the clean speech spectral amplitude from the preceeding segment $m-1$. In case that in a spectral bin f_k the SNR is very high, (2.38) yields $\hat{SNR}_{prio}(m, f_k) \approx \hat{SNR}_{post}(m-1, f_k)$ after several segments of speech activity. This is generally sufficient to prevent distortion of the speech coefficients when $\hat{SNR}_{prio}(m, f_k)$ is used in a noise reduction algorithm. Typical values of the parameter α are in the range 0.92 to 0.98 [8]. A higher value of α better suppresses musical noise, but this also leads to an undesired clipping of low energy speech components so that the cleaned speech sounds muffled. We can also notice that $\alpha = 0$ leads again to (2.36).

Chapter 3

A noise reduction technique

For the development of our noise reduction technique, we focus on the last Apple's devices [23]. In particular, we analyze the performance of iPhone 4, iPhone 5, iPad and iPad 2 for speech recordings and we measure the amount of noise introduced by these devices. Then we show the MATLAB implementation of our noise reduction filter and make some comments on its parameters. Finally, we define the *noise print*, capture a silent recording for each device and make comparisons between the noise power distribution over the spectrum on different devices.

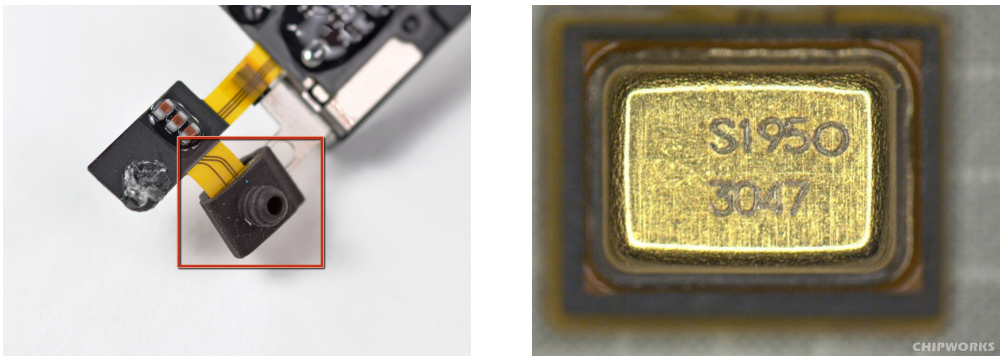
3.1 iPhone and iPad microphones

Nowadays a variety of different microphone types exist. Most microphones today use electromagnetic induction (dynamic microphone) or capacitance change (condenser microphone) [24] to produce an electrical voltage signal from mechanical vibration. For those applications in which a compact and efficient microphone solution is required, such as on smartphones, MicroElectrical-Mechanical System (MEMS) microphones are used [25]. This kind of microphone provides a pressure-sensitive diaphragm which is etched directly into a silicon chip by MEMS techniques [26], and is usually accompanied with an integrated preamplifier. Most MEMS microphones are variants of the condenser microphone design and offer plenty of advantages, including tiny size, low power usage and consistent performance over time and temperature [27]. Often MEMS microphones have built in analog-to-digital converter (ADC) circuits on the same CMOS chip, making the chip a digital microphone, readily integrable on modern digital products. Major manufacturers producing MEMS silicon microphones are Wolfson Microelectronics [28], Analog Devices [29], Akustica [30], Infineon, Knowles Electronics [31]

and STMicroelectronics [32].

Any microphone produces some level of noise through its electronics, its transducer and its housing. This inherent noise is known as self noise [27]. An high signal-to-noise ratio (SNR) indicates a quiet noise, while a lower SNR is related to microphones with greater self noise. When the audio source is very close to the microphone, the SNR is usually high since the source power is high too and thus the useful signal power is enough for near-field applications, such as during calls. On the contrary, in far-field applications, where the microphone is not positioned next to the sound source, a noisy mic with low SNR can only generate a poor signal. This is what normally happens using a smartphone for speech recording purposes because the main audio source (the speaker) is usually far from the device.

As far as iPhone 4 is concerned, Apple included two MEMS microphones in the device handset [33]. The main microphone, placed on the bottom side of the device, is used both for making calls and for general audio recording. It is manufactured by by Knowles Electronics, model S1950, and it is shown in Fig.3.1. The Knowles S1950, like the other microphone in the iPhone 4, consists of two main parts: the MEMS to capture sounds and the ASIC to interpret the analog signals given off by the MEMS die (Fig.3.2).

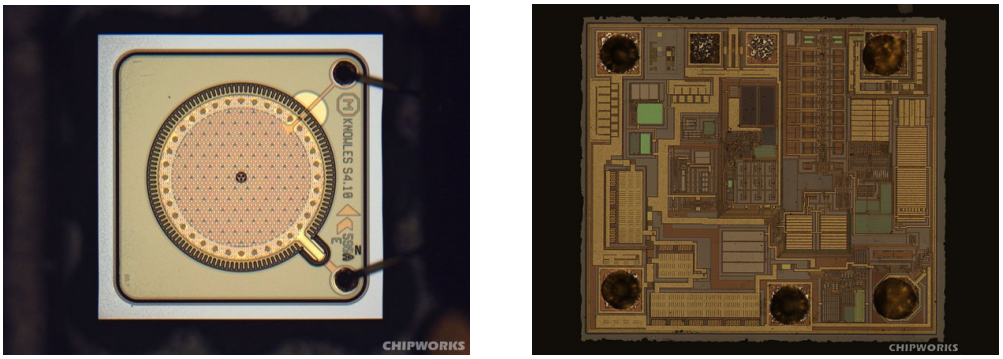


(a) Microphone in its package (b) S1950 chip

Figure 3.1: iPhone 4 main microphone (Knowles S1950)

The MEMS works like a microscopic vesion of a condenser microphone. The microphone itself has a simple design, comprising of two parallel polysilicon plates (very thin plates made of multiple small silicon crystals) that act as plates in a capacitor. The upper plate is perforated with an array of small holes, and is separated from the bottom plate by a small air gap. As the sound waves from someone's voice hit this top plate, the upper plate is deflected very slightly. Because these two plates hold electric charge, these

deflections cause minute changes in the electric field between the two plates. The fixed bottom capacitor plate senses and relays these changes as an analog signal. The ASIC portion of the microphone decodes and processes the analog signal sent to it from the MEMS and sends the result to the iPhone 4 processor [33].



(a) MEMS (b) ASIC

Figure 3.2: Knowles S1950 components

The second microphone in the handset is the Infineon 1014, which is only used for noise reduction during calls but whose data seem not to be accessible from the developer end. This microphone, like all the built-in MEMS mics on modern smartphones, works in a similar way of Knowles S1950.

3.2 Filter implementation

We propose a MATLAB implementation of the Wiener filter based on decision-directed method by Ehpraim and Malah [22]. As explained in Section 2.2.2 this tecnique requires both the noisy speech signal and a sample of the noise signal in order to estimate the original clean speech. A widespread technique to extract a pure noise sample is to apply a silence detector to the original recording to recognize segments in which speech is absent and then merge these segments to the desired sample length. We want to highlight that longer noise samples provide higher frequency resolution on the noise PSD estimation. Provided that a recording has at least one silent segment longer than one second, an appropriate settings of the silence detector would make it possible to reveal that silent sample and, therefore, apply the Wiener filter. Notice that the noise must be quasi-stationary, otherwise the noise sample used to filter the recording must be recomputed more than once, possibly at high rate.

Actually, the quasi-stationarity is a reasonable assumption for many noise enviroments such as the noise inside a car emanating from the engine, aircraft noise, office noise from computer machines, etc. Since the noise is assumed to be quasi-stationary, the knowledge of the noise PSD would be sufficient to apply the Wiener filter, that is to say that the noise sample can be replaced by the noise PSD that carries the same information.

Moreover, if our aim is just to eliminate self noise introduced by the built-in microphone, we can estimate the noise PSD from a silent recording and store it for further usage, without performing a new noise print estimation every time we apply the filter.

Finally, is we find out that the self noise print is almost the same on all the pieces of the same smartphone model, we can compute the noise PSD only once for each model and hard code it into the filtering software in order to perform speech enhancement without the need for a silent detector. The filter can also be applied to recordings that do not have silent segments long enough.

3.2.1 Noise prints

We call *noise print* an estimate of the PSD of a quasi-stationary noise signal, based on the Welch's method (2.1.3). The algorithm for the noise print estimation starting from a noise sample is reported in Listing 3.1. Firstly, given the window length W and the overlapping factor S, the noise sample is split into N overlapping segments. Then, the FFT of each segment is computed. Because the process is wide-sense stationary and Welch's method uses PSD estimates of different segments of the time series, the modified periodograms approximately represent uncorrelated estimates of the true PSD and averaging reduces the variability [20]. The final result is an estimated noise print with W frequency bins.

In our analysis, all the noise signals are segmented into frames of 1024 samples, 50% overlapping and windowed with the Hamming function. This leads to PSDs with 512 frequency bins and 43 Hz resolution.

Listing 3.1: Matlab algorithm to compute a *noise print* from noise sample

```
1 % Segmentation. Given a noise sample, the window length W ...
    and the overlapping fator S, we get a W x N matrix, ...
    where N is the number of frames.
2 seg = segmentHamming(noiseSample, W, S);
3 % Compute FFT
4 N = fft(seg);
5 % Extract amplitude
```

```
6 NAbs = abs(N);
7 % Compute periodogram of noise sample. The result is a W x ...
      1 matrix.
8 noisePrint = 1/W * mean((NAbs.^2), 2);
```

3.2.2 Decision directed method implementation

The implementation of the main filter is shown in Listing 3.2. It requires both the noisy signal spectrogram [34] *ySpectr* and the noise print of the undesired signal. Notice that we only modify the amplitude of the noisy speech, with different gain values for each frequency bin and each frame. In order to exploit the quasi-stationarity of the speech signal, the frame length cannot be too large. Its typical value is 20-40 ms, so the segmentation of a 10 seconds recording leads to $250 - 500$ non-overlapping frames. Actually, our segmentation algorithm is based on the Hamming window [35] at 50% overlap and 1024 samples segments. Considering a sample rate of 44.1 kHz, 1024 samples corresponds to segments of 23.3 ms length. Hence 10 seconds leads to ≈ 860 segments that is also the number of times the filter frequency response is evaluated. Given that the FFT size equals the segments length, every PSD estimation consists of 513 values correponding to the frequencies bins between 0 and the Nyquist frequency.

The filter gain is based on the decision-directed method of the *a priori SNR* [22].

Listing 3.2: Matlab implementation of Wiener filter based on decision-directed method

```
1  % Filter implementation
2  XPsd = 0;
3  for k=1:numFrames
4      SNR_Post = ySpectr(:,k)./noisePrint;
5      SNR_Pri = alpha*XPsd./noisePrint + ...
           (1-alpha).*max(SNR_Post-1,0);
6      % Wiener filter
7      G = SNR_Pri./(SNR_Pri + 1);
8      YAbs(:,k) = G.*YAbs(:,k);
9      % Power Spectral Density estimation of last cleaned ...
           frame (this will be used in the next iteration)
10     XPsd = 1/W * YAbs(:,k).^2;
11 end
12
13 % Segments merging
14 % Back to complex number
15 Y = YAbs.*exp(1i*YPhase);
```

```
16 % Inverse Fourier transform
17 seg = real(ifft(Y));
18 % Merge segments into the final signal
19 newSignal = mergeSegments(seg, S);
```

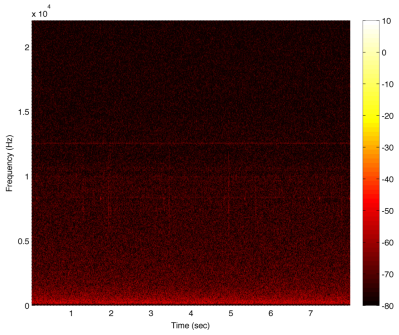
The main parameter `alpha` represents a trade off between musical tones suppression and speech distortion. In particular `alpha` ≈ 0 leads to an excellent speech quality but weak noise reduction and annoying musical tones. On the contrary, a value of `alpha` ≈ 1 better suppresses musical noise, but this also leads to an undesired clipping of low energy speech components and, as a consequence, the cleaned speech sounds muffled. A number of subjective and objective tests with the purpose of determining the optimal value of α are available in the literature [8, 9, 22] where a value of 0.98 was determined as a good compromise.

3.3 Measures

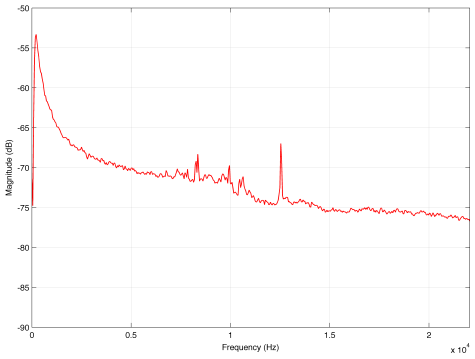
Our test recordings have been performed in a room lined with sound assorbing material to reduce the environmental noise as much as possible. Two different recordings have been taken: 1) a silent recording, 8 seconds long, at 44100 sample rate and 16 bit depth; 2) a recording with a spoken voice, at 44100 sample rate and 16 bit depth.

3.3.1 Noise measure

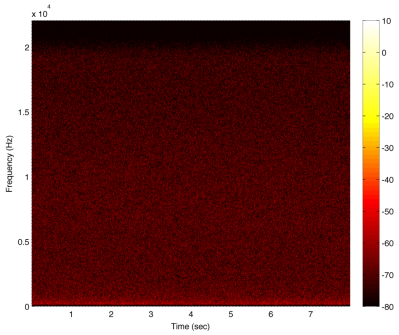
The first 44.1 kHz recording has been used to evaluate the self noise of each analyzed device and the results are shown in Fig. 3.3 both as periodograms and PSDs. The PSD is estimated between 0 and the *Nyquist frequency* 22050 Hz. Notice that the self noise is not a white noise [16] because the power is not equally distributed along the spectrum. The iPad 1st generation introduces a noise that rapidly decay at low frequencies while has a slow, linear gradient for medium and high frequencies. Two spectral peaks are clearly visible at $f \approx 8300$ Hz and $f \approx 16600$ Hz. The iPad 2 shows the same fast decay at low frequencies but there are no peaks and the power distribution between 2000 Hz and 18000 Hz resemble that of white noise. Above 18000 Hz the PSD shows a sudden decrease. The noise recorded by the iPhone 4 has two peaks at 15000 Hz and a 12 dB fall in the noise power around 17000 Hz. Finally, the new iPhone 5 shows a quite irregular distribution of the noise power for high frequencies, with a peak at 15000 Hz that matches the one of iPhone 4.



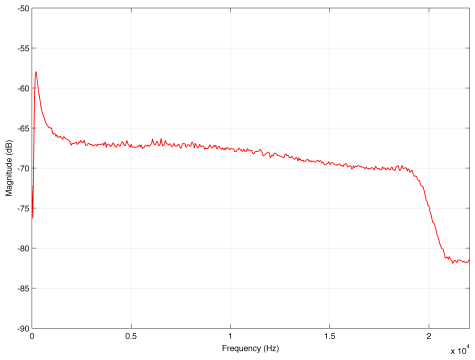
(a) iPad 1 - Spectrogram



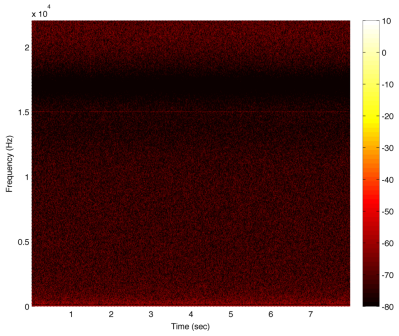
(b) iPad 1 - Noise PSD



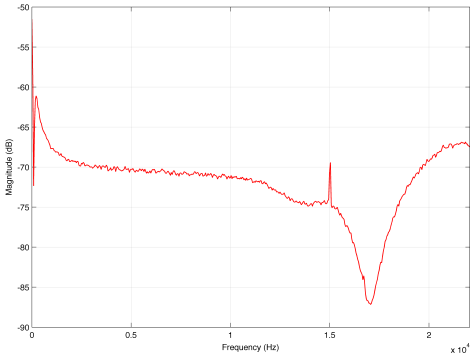
(c) iPad 2 - Spectrogram



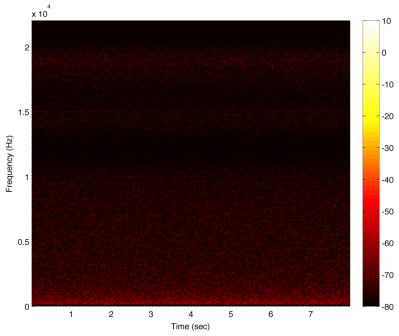
(d) iPad 2 - Noise PSD



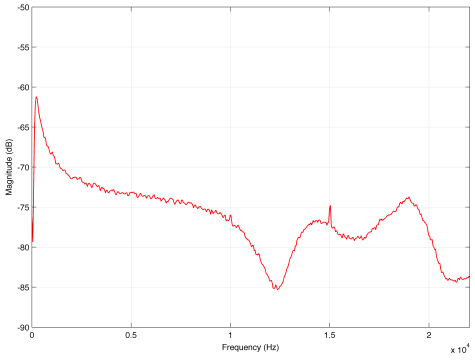
(e) iPhone 4 - Spectrogram



(f) iPhone 4 - Noise PSD



(g) iPhone 5 - Spectrogram



(h) iPhone 5 - Noise PSD

Figure 3.3: Noise power spectral density of different devices

Table 3.1: RMS Amplitude of Noise signals

Device	Maximum	Minimum	Average
iPad 1st Gen	-63.31 dB	-65.52 dB	-64.48 dB
iPad 2	-63.47dB	-64.35dB	-63.89dB
iPhone 4	-65.52dB	-66.49dB	-66.04dB
iPhone 5	-68.98dB	-70.31dB	-69.72dB

We highlight that a number of different factors are involved in the noise generation, e.g. the microphone's self-noise, the thermal noise in the conductors and the noise introduced by the amplifier. Moreover, we do not have access to Apple's filtering algorithms that are applied to the raw data from MEMS sensors and therefore we can only adopt a black box approach. Nevertheless, given that the analyzed noise signals are stationary, the final noise data is sufficient to generate accurate noise prints for each device and then apply our noise reduction strategy to the noisy recordings.

A time-domain analysis reveals that the RMS amplitudes of the noise signals are similar for all the considered devices. The analysis is based on 50 ms non-overlapping windows and the results are reported in Table 3.1. The gap between the devices with the highest noise RMS amplitude (iPad 2) and the lowest RMS amplitude (iPhone 5) is 5.8 dB which means that the power of the self noise introduced on the iPad 2 is about 4 times greater than on the iPhone 5.

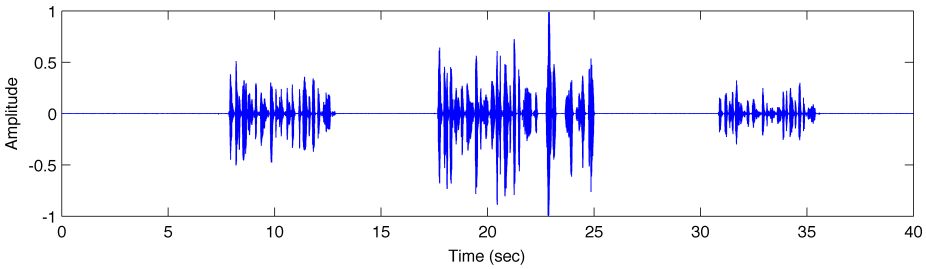
To conclude, we want to point out that the storage space required for a single noise print is about 2 kilobytes. In fact we can suppose a 32-bit floating point number for each value and a total of 512 frequency bins to cover the entire spectrum with a sufficient resolution. Therefore, storing a hundred noise prints for as many devices constitutes a negligible use of resources for the majority of mobile applications. We propose this approach to develop recording apps with noise reduction algorithms that rely on noise prints previously evaluated. This approach can be exploited to reduce the noise generated by the electronics, while the reduction of enviromental noise still requires a real-time analysis of the noise PSD.

3.3.2 Speech recordings

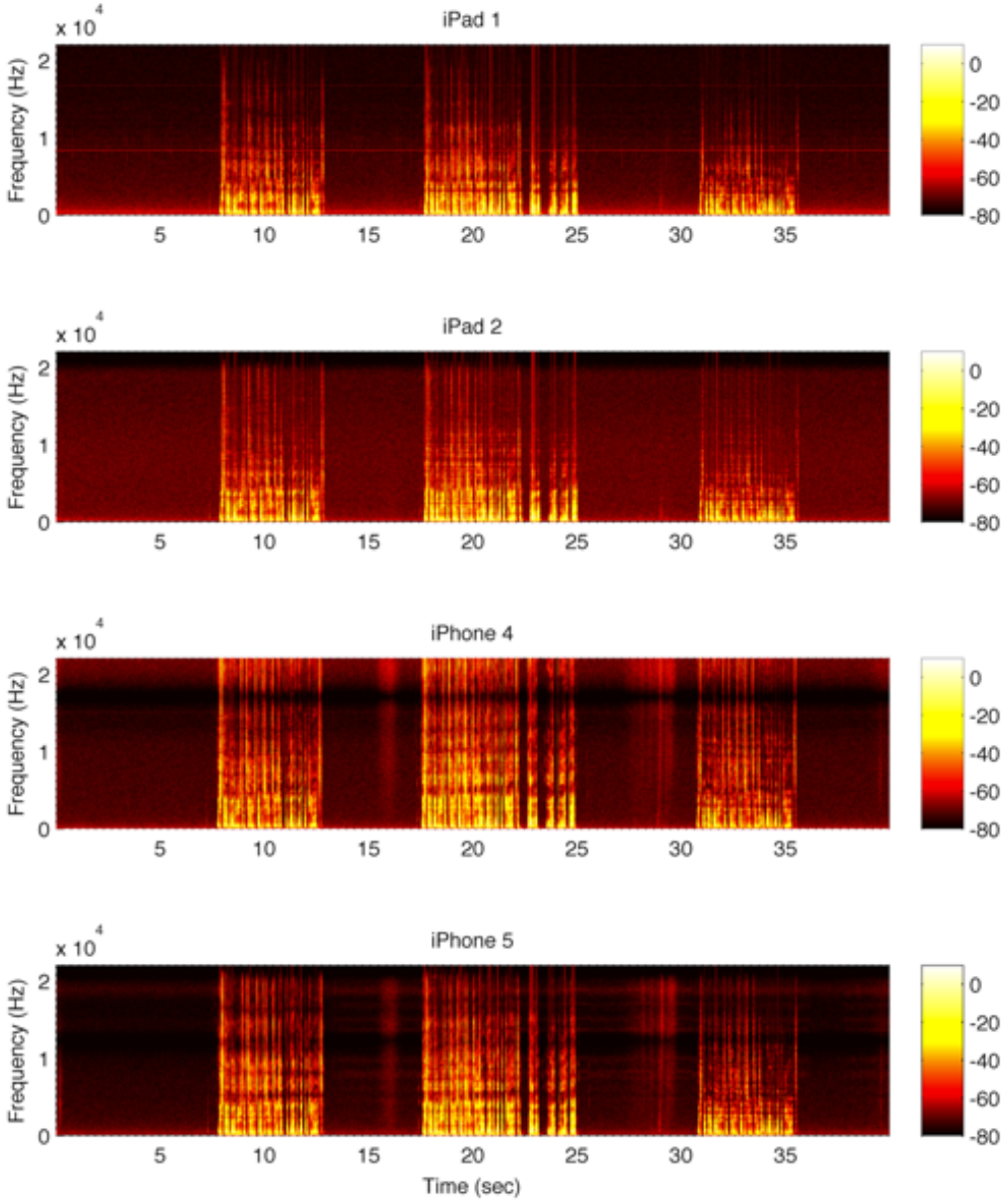
The four devices analyzed have been recording the same source at the same time to ensure results consistency. Fig. 3.4 shows the signal waveform and the

related spectrograms for different devices. The recording are 40 seconds long and the speech signal is clearly divided into three blocks. The discontinuities on the speech signal are useful to highlight the background noise and to give us the possibility of testing our filter response to sudden changes in the signal amplitude. Furthermore, compared to the first speech frame, the second one is louder while the third one is quieter in order to profile the filter gain for different SNR values.

Notice that the recording presents a significant amount of background noise on all the devices but the noise power distribution over the spectrum is sligthly different on different device models, which is in agreement with the noise prints shown in Fig. 3.3.



(a)



(b)

Figure 3.4: Speech recording waveform (a) and spectrogram on different devices (b)

Chapter 4

Results

In this chapter, we firstly discuss the trade off between noise reduction and speech distortion. Then we show the results of our noise reduction filter applied to the noisy speech recordings. The noise filter, that requires the noise prints of each device in order to perfom speech enhancement properly, will only rely on the noise prints previously computed and shown in 3.3.1. Finally, we compare the noise prints of three copies of the same device model, an iPhone 4, to validate the hypothesis according to which different copies of the same device model have similar noise prints. This will confirm that every noise print can be evaluated once only and then applied to different copies of the same device model.

4.1 Noise suppression and speech distortion

As highlighthed in Section 3.2.2, the results our Wiener filter implementation based on the decision-directed method are strongly influenced by the value of the parameter *alpha*. Low values of this parameter reduce the distortion on the enhanced speech, but reduce the filter noise attenuation as well, introducing annoying musical tones. On the contrary, high values of *alpha* lead to an undesired clipping of low energy speech components with the undesired consequence of a distorted speech signal.

In Fig. 4.1, the average filter gain over the spectrum for different values of `alpha` is shown and compared to the original noisy signal power. On the one hand, we can see that the filter attenuation on every frame is strongly related to the original signal power on the same frame. This means that the attenuation is highest when speech is absent and weaker when the useful signal have to be preserved. On the other hand, the attenuation depends on the value of `alpha`. Low values of this parameter lead to weak noise

attenuation (10 dB or less) while values `alpha` ≈ 1 can raise the attenuation indefinitely.

The main drawback of high values of `alpha` is speech distortion. As you can see in Fig.4.1(c, d), as long as `alpha` increases the variance of the filter gain increases as well. This means that the filter becomes more intrusive even during speech presence, with a high attenuation during the short intervals of silence between the words of a sentence. As a consequence, the beginning of a new word, which consists of a low-power signal, tends to be clipped by the filter and this is the main cause of speech distortion.

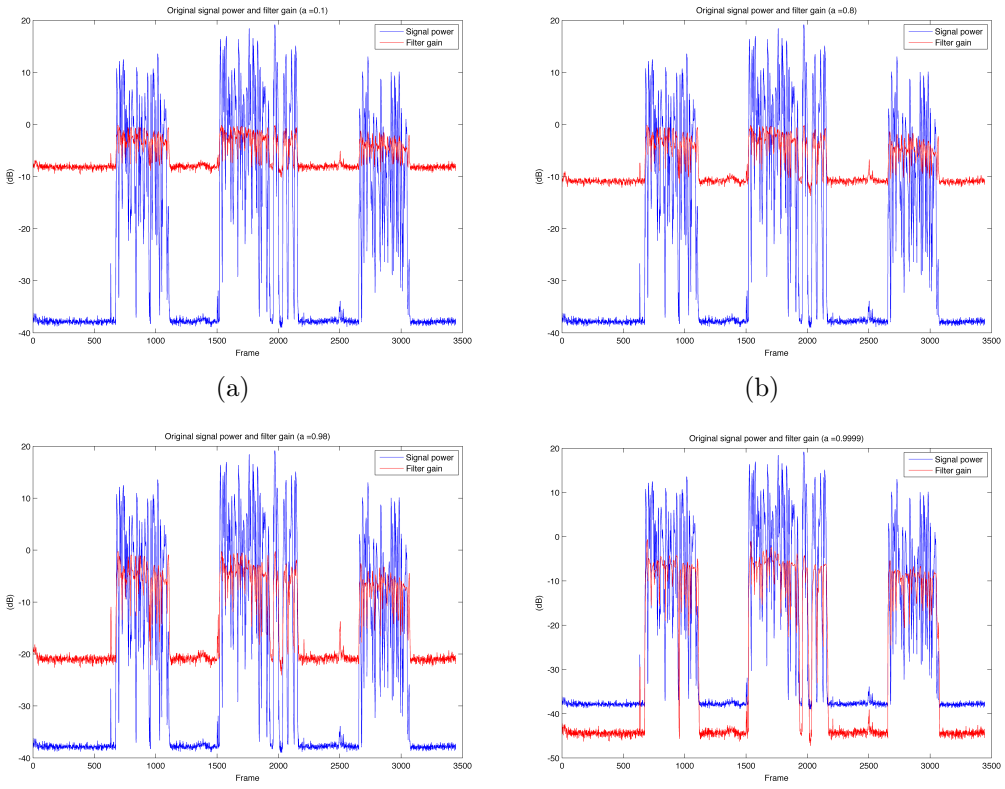


Figure 4.1: Original signal signal power compared to filter gain for different values of *alpha*

Although the original signal power determines the average noise filter gain, remember that the gain for each frequency bin is evaluated independently. In Fig.4.2, we can see that even in those frames where speech power is predominant over background noise, the filter attenuation is still strong for those high frequencies outside the speech signal band. Notice that, during

speech presence, the filter gain is ≈ 1 at lower frequencies, which means that these frequencies are not significantly affected by the filter itself.

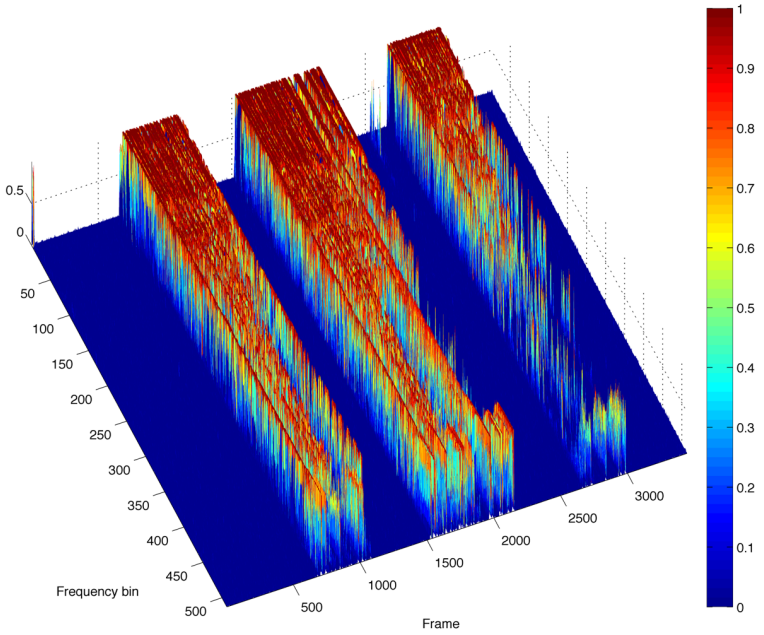


Figure 4.2: Filter gain over different frequency bins and time frames

Finally, Fig.4.3 shows the relation between noise-reduction factor [8] and and the value of the parameter `alpha`. One of the primary issues we must determine when dealing with a noise reduction filter is how much noise is actually attenuated. The noise-reduction factor is a measure of this, and is defined as the ratio between the original noise intensity and the intensity of the residual noise remaining in the noise-reduced speech. This value is greater than one when noise is reduced.

The graph shows the attenuation of the original recording during absence of speech. The noise-reduction factor tends to infinity (so the filter gain tends to 0) for `alpha` tending to 1, so we can always choose a value of `alpha` that guarantee the desired noise suppression. Unfortunately, as discussed before, too high a noise attenuation leads to an enhanced speech that sound rather muffled.

To evaluate the performances of a noise filter in keeping the desired speech signal unchanged there are two categories of measures, i.e. subjective and objective ones. Subjective measures rely on human listeners' judgments and,

as far as speech quality is concerned, this method should be the most appropriate performance criterion because it is the listener's judgment that ultimately counts. Unfortunately, subjective evaluation is labor intensive, time consuming and the results are expensive to obtain. A lot of tests with the decision-directed method for the *a priori* signal-to-noise ratio estimation can be found in literature [8, 22]. Appropriate and widely used values of `alpha` that leads to both good speech quality and noise suppression are $0.96 - 0.98$.

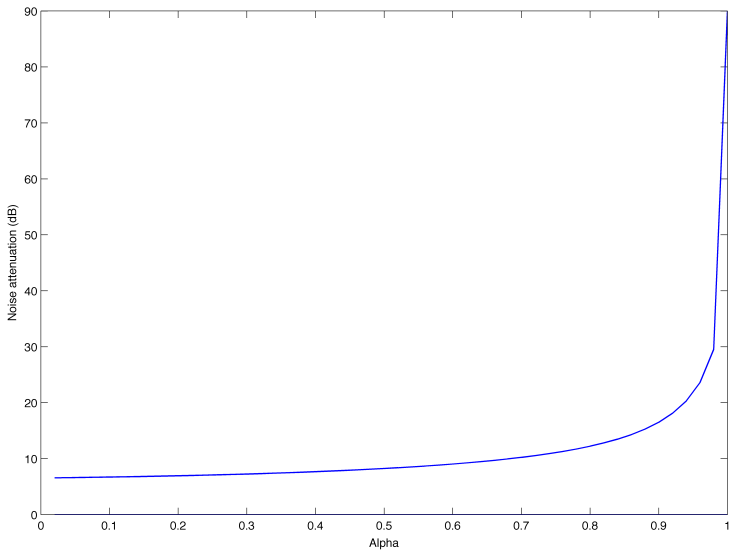
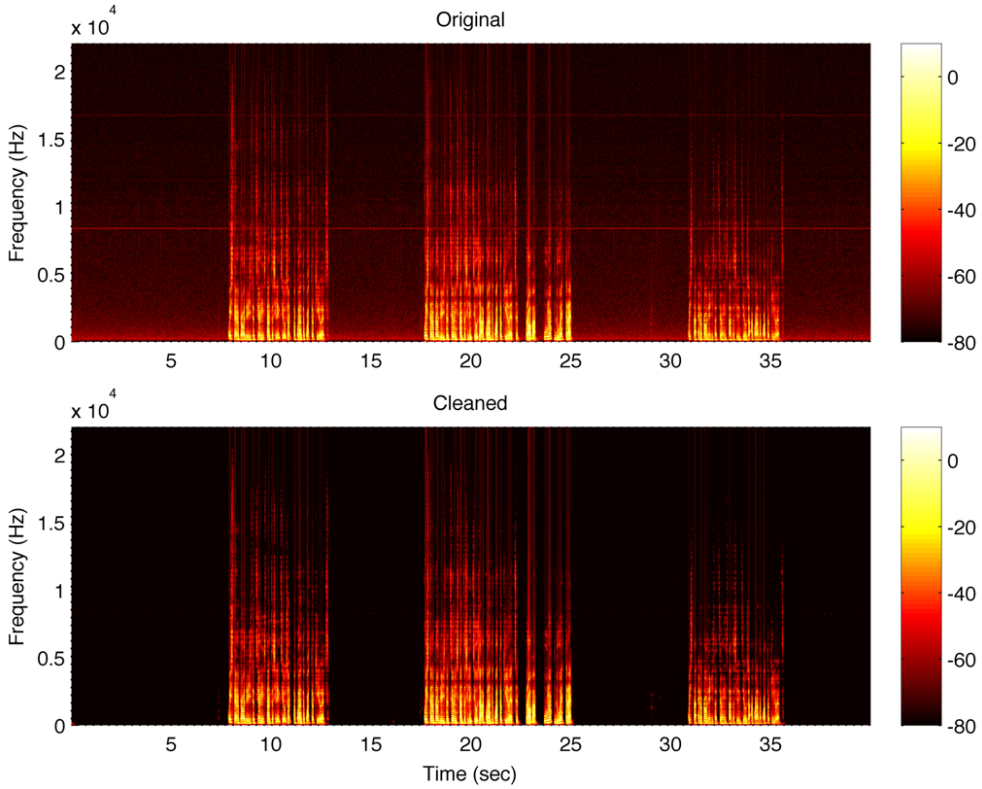


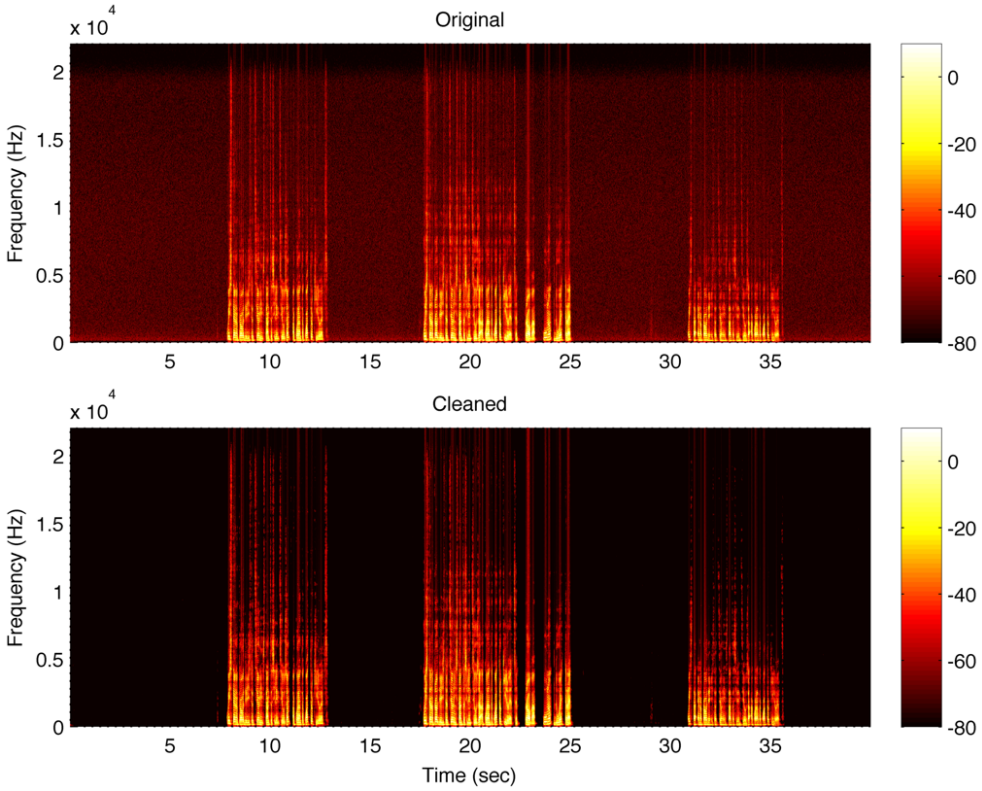
Figure 4.3: Noise-Reduction factor (in dB) over different values of `alpha`

4.2 Enhanced recordings

We now show the results of speech enhancement with the proposed algorithm, for different device models and with the main parameter `alpha` $=$ 0.98. The periodograms in Fig.4.4 and Fig.4.5 clearly show that attenuation of background noise is almost 30 dB while the power of the useful signal is preserved. The listening quality is very good. The speech signals present very low distortion while the power of the background noise has been reduced to a level that can be heard only by turning up the volume significantly. When the noise is audible, it presents some musical tones but the power of this residual noise is not high enough to make the listening annoying.

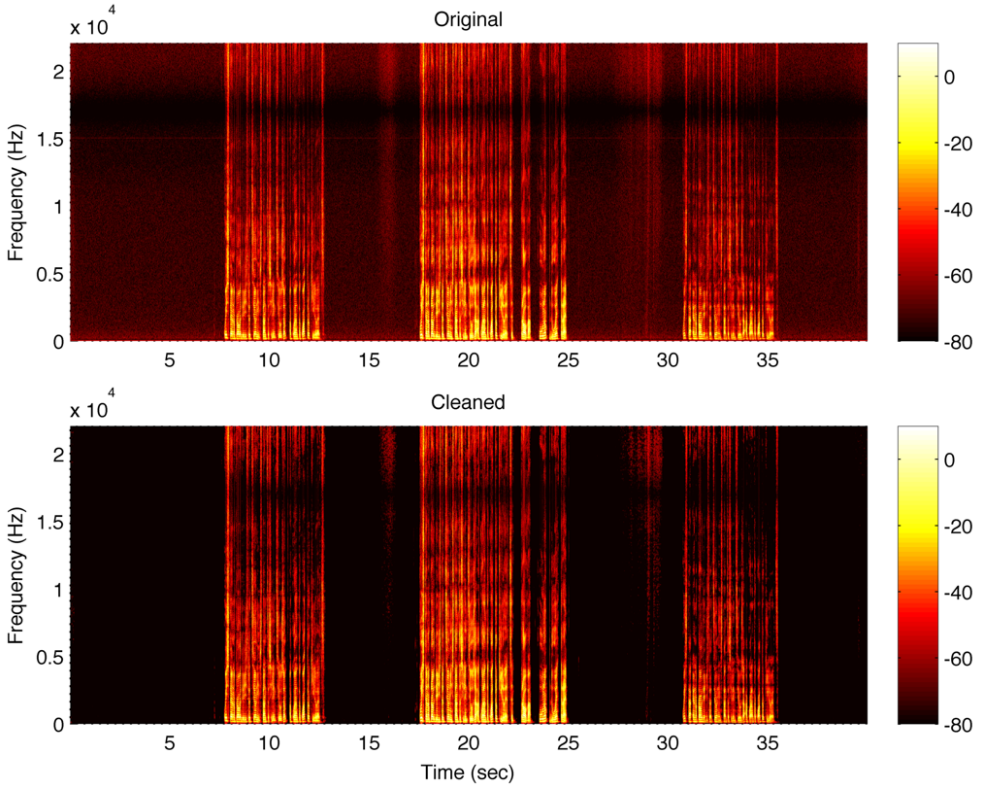


(a) iPad 1

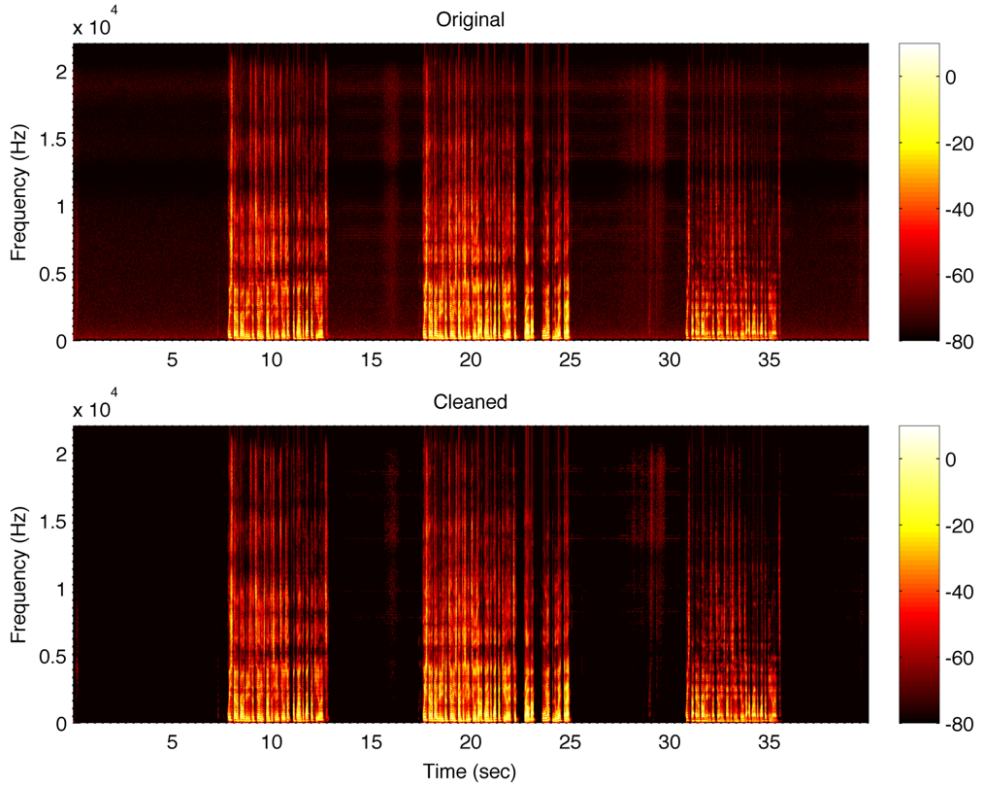


(b) iPad 2

Figure 4.4: Comparison between original and cleaned signals spectrograms (iPad)



(a) iPhone 4



(b) iPhone 5

Figure 4.5: Comparison between original and cleaned signals spectrograms (iPhone)

4.3 Noise prints comparison

We suggested to use a pre-evaluated noise print for each device model and apply that noise print to the noise reduction filter on all the device copies of the same model. This assumption is based on the consideration that copies of the same device are results of mass production and the same components are embedded in them. The usage of the same model of MEMS microphone is an important factor that leads to similar self noise PSDs. Yet, we should consider parameters fluctuations on all the componets, which differentiate each unique device from the others, but we believe that the noise prints are similar enough to obtain the desired results ignoring this slight difference.

For our tests, we used three copies of an iPhone 4. The noise prints for these devices have been plotted in Fig.4.6.

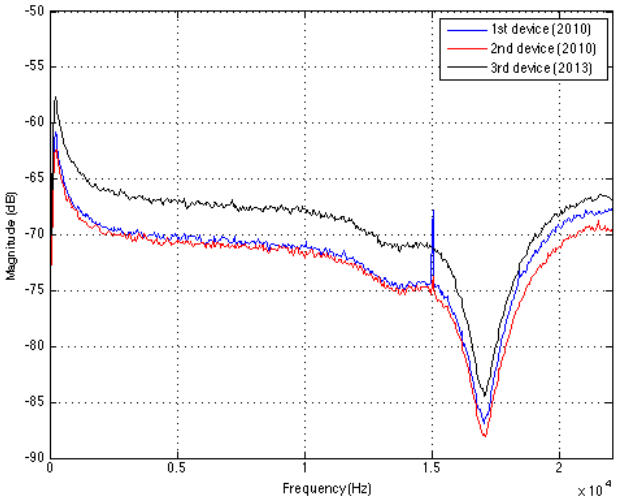


Figure 4.6: Comparison of noise prints for three different iPhone 4 copies

We can clearly notice that the two devices produced during the same year present very similar noise prints. The third device, a copy of the same model but produced two years later, introduces a noise with a slightly higher power; yet, the power distribution of this copy still follows the same shape over the spectrum. We believe that averaging the noise prints of a few devices can generate a valid noise print to be applyed on all the copies of the same model.

Chapter 5

Conclusions and future work

5.1 Conclusions

The first and foremost conclusion of this work is that a convenient noise reduction filter can be implemented on smartphones and mobile devices. This filter represents a good approach to compensate the noise introduced by the low quality electronic components with a solution that does not require any additional hardware. Our tests proved that the speech signal enhanced by our filter contains far less background noise than the original recording and that, at the same time, the speech signal quality is preserved.

This filter could be implemented either in the device OS by the device manufacturer or in dedicated recording applications. Relying only on pre-evaluated noise prints, the algorithm can be used on each supported device without requiring the user a further calibration. A recording application that implements this algorithm should contain a database with all the noise prints of the device models in which the application is expected to work and, therefore, this requires an intensive and time consuming labor for the developers who will have to simulate a silent recording on all the devices they want to support and extract the necessary noise PSDs from them. On the other hand, this solution does not create problems in terms of memory usage and is ready-to-work when the app is installed on the device. We underline the importance of ease of use and absence of configurationis which are foundamental for the success of the application, considering that nowadays users are not willing to spend time to set up their smartphone and expect every app to work without complicated setting procedures.

Besides dedicated recording applications, the filter could be implemented on the OS by the device manufacturer. The cooperation of the manufacturers to generate and publish the noise prints for the devices they produce would

be extremely important to speed up the process and to add support for new devices as soon they become available on the market. Since the main difficulty in the implementation of the proposed technique is the creation of a database with all the noise prints required, it would be convenient to have access to the noise prints data directly from the manufacturers. Yet, a standard format for this kind of data should be created in order to facilitate the integration of the noise prints in the application that requires them.

5.2 Future work

The next step in the development of a complete solution for speech enhancement on mobile devices will be the conversion of the MATLAB alghoritm in C++ and then its porting to different devices. This will require a particular attention on the performance, in particular for the FFT algorithm that must be highly efficient. The alghoritm will be first ported on iOS since this OS is used on a limited number of different devices and therefore the creation of all the necessary noise prints would be rather straight forward. On iOS, the Accelerate framework [36] would provide all the highly efficient mathematical functions that we need to convert the MATLAB code and obtain good performances. A number of test for resources usage will be necessary to understand whether or not the power of older devices is enought to achieve the desired results is a resonable amount of time.

Moreover, further improvements of the filter alghoritm are possible. The actual filter is designed to only reduce the self noise introduced by the microphone and the other electronic components but it would be possible to modify the code to dynamically adapt to the environmental noise and reduce it as well.

Finally, the algorithm should be modified to be applicable in real-time. We would like to apply the filtering during the recording playback in order to make the elaboration invisible to the final user and preserve the original file too.

Bibliography

[1] Mintel Oxygen Reports, "Sales of digital cameras decline as consumers snap up smartphones." http://www.mintel.com/press-centre/press-releases/890/sales-of-digital-cameras-decline-as-consumers-snap-up-smartphones, 2012.

[2] Prosper Mobile Insights, "Smartphones and Tablets Replacing Alarm Clocks, GPS Devices and Digital Cameras, According to Mobile Survey." http://www.prweb.com/releases/2011/7/prweb8620690.htm, June 2011.

[3] "iOS Developer Documentation - Performance Tuning." http://developer.apple.com/library/ios/#documentation/iphone/conceptual/iphoneosprogrammingguide/PerformanceTuning/PerformanceTuning.html#//apple_ref/doc/uid/TP40007072-CH8-SW1.

[4] "Microsoft MSDN: Designing Mobile Applications." http://msdn.microsoft.com/en-us/library/ee658108.aspx.

[5] "iPhone Official Website." http://www.apple.com/iphone/.

[6] "iPad Official Website." http://www.apple.com/ipad/.

[7] S. V. Vaseghi, *Advanced Signal Processing and Digital Noise Reduction*. Wiley - Teubner, 1996.

[8] Benesty, Sondhi, and Huang, *Springer Handbook of Speech Processing*. Springer, 2008.

[9] C. Breithaupt and R. Martin, "Analysis of the Decision-Directed SNR Estimator for Speech Enhancement With Respect to Low-SNR and Transient Conditions," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 19, no. 2, pp. 277 – 289, 2010.

[10] C. Beaugeant and P. Scalart, "Speech Enhancement using a Minimum Least Square Amplitude Estimator," *Proceeding of IWAENC*, 2011.

[11] A. V. Oppenheim, A. S. Willsky, and H. Nawab, *Signals and Systems*. Prentice-Hall, 1997.

[12] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*. Prentice-Hall, 1999.

[13] F. Rocca, *Elaborazione numerica dei segnali*. CUSL Milano, 2005.

[14] M. F. Steven G. Johnson, "Implementing FFTs in practice." `http://cnx.org/content/m16336/latest/`.

[15] A. B. Melissa Selik, Richard Baraniuk, "Signal energy vs. signal power." `http://cnx.org/content/m10055/latest/`.

[16] N. Benvenuto and M. Zorzi, *Principles of Communications Networks and Systems*. Wiley, 2011.

[17] S. Tubaro, "Some notes on the power spectral density of random processes." `http://home.deib.polimi.it/tubaro/ENS2/psd.pdf`, 2011. Notes.

[18] P. Stoica and R. Moses, *Spectral Analysis of Signals*. Prentice-Hall, 2005.

[19] MathWorks, "Spectral analysis - matlab documentation," 2013.

[20] P. D. Welch, "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms," *IEEE Trans. Audio Electroacoustics*, vol. AU-15, pp. 70–73, June 1967.

[21] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. M.I.T. Press.

[22] Y. Ephraim and D. Malah, "Speech Enhancement Using a Minimum Mean-Square Error Short-Time Spectral Amplitude Estimator," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-32, no. 6, pp. 1109–1121, 1984.

[23] "Apple Official Website." `http://www.apple.com/`.

[24] G. M. Ballou, ed., *Handbook for Sound Engineers*. SAMS, 1991.

[25] AAC Technologies, "Microphones." `http://www.aactechnologies.com/category/10`.

[26] "MEMS Exchange." https://www.mems-exchange.org/MEMS/what-is.html.

[27] "Technical Article MS-2348 - Low Sel Noise: The first step to High Performance MEMS Microphone Application," tech. rep., Analog Devices, 2012.

[28] "Wolfson Microelectronics MEMS Microphones." http://www.wolfsonmicro.com/products/mems_microphones/.

[29] "Analog devices mems microphones." http://www.analog.com/en/mems-sensors/mems-microphones/products/index.html.

[30] "Akustica MEMS Microphones." http://akustica.com/Microphones.asp.

[31] "Knowles MEMS Microphones." http://www.knowles.com/search/products/m_surface_mount.jsp.

[32] "STMicroelectronics MEMS Microphones." http://www.st.com/web/en/catalog/sense_power/FM89/SC1564.

[33] "iPhone 4 Microphone Teardown." http://www.ifixit.com/Teardown/iPhone+4+Microphone+Teardown/3473/1.

[34] "Spectrogram using short-time fourier transform." http://www.mathworks.it/it/help/signal/ref/spectrogram.html.

[35] "Hamming window." http://www.mathworks.it/it/help/signal/ref/hamming.html.

[36] "iOS Developer Documentation - Accelerate framework." http://developer.apple.com/library/ios/#documentation/Accelerate/Reference/AccelerateFWRef/.