Final Degree Thesis

Batch Processing of Electrical Bioimpedance Spectroscopy Measurements. Implementation and Validation

By

VICTOR SANCHEZ ORTIZ

UNIVERSITY OF BORÅS

SCHOOL OF ENGINEERING

FINAL DEGREE THESIS 30 ECTS, ERASMUS 2009-10 SWEDEN
THESIS Nº 2/2010, COMUNICATION AND SIGNAL PROCESSING

Batch Processing of Electrical Bioimpedance Spectroscopy Measurements. Implementation and Validation

Víctor Sánchez Ortiz

Master Degree Thesis
Subject Category: Medical Technology, Signal Processing

University of Borås
School of Engineering
SE- 501 90 BORAS
Phone number: +46 33 435 4640

Examiner: Fernando Seoane Martínez
Supervisor: Fernando Seoane Martínez
Date: 2010 May 26th

Keywords: Electrical Bioimpedance Spectroscopy, Cole Model, Hook Effect, Signal Analysis.

ABSTRACT

Nowadays, Electrical Bioimpedance (EBI) measurements have become a common practice as they are useful for different clinical applications for non-invasive monitoring.

In recent years new applications of EBI measurements based in spectral analysis have risen and been validated. Due to this fact, the use of spectral analysis on Electrical Bioimpedance measurements is going to open the door for new indicators for health assessment.

One of the goals of this thesis is to provide functions for the development of a Software tool for Electrical Bioimpedance Spectroscopy analysis, the other is to design and implement functions to perform a batch analysis of EBI measurements of different subjects for comparison.

Once these objectives have been implemented, spectral analysis and validation of characterization features will be checked easily, accelerating the process of test and analysis of experimental data analysis.

ACKNOWLEDGEMENTS

M’agradaria començar per donar les gracies a tota la meva família, els meus pares, avis, ties i oncles, que han estat sempre al meu costat, ajudant-me i animant-me en els moments en que ha durat aquesta travessia, en alguns moments molt dura, que ha sigut la carrera. I especialment, vull recordar la meva mare que sempre ha estat al meu costat estimant-me i que ara mateix esta passant un mal moment i només li puc dir: “ Mama cúrate que te necesito a mi lado”.

Als meus amics de Figueres, que sempre que torno a l’Empordà em fan recordar que estic a casa.

Als diferents amics que he anat fent durant la meva estada a Barcelona i que ser que són amics per tota la vida.

Als diferents companys de feina, de la UPC i professors, que han compartit els seus coneixements i temps amb mi.

Als meus amics del laboratori de Högskolan i d’Erasmus que han fet que aquesta estada a Borås hagi sigut una de les millors experiències de la meva vida i que els portaré al cor tota la vida, especialment a Sabrina.

I per suposat al meu supervisor, Dr. Fernando Seoane Martínez, ja que sense la seva ajuda hauria sigut impossible realitzar aquest treball. “Fernando, gracias por guiarme y por todo el conocimiento que me has transmitido durante estos meses en Suecia”.

TABLE OF CONTENTS

LIST OF ACRONYMS

BIVA	Bioelectrical Impedance Vector Analysis
BT	BioImpedance Tomography
CPE	Constant Phase Element
ECF	Extracellular fluid
EIP	Electrical Impedance Plethysmography
EIT	Electrical Impedance Tomography
GUI	Graphical User Interface
GUIDE	Matlab Graphical User Interface Design Tool
V/I	Voltage / Current
ICF	IntraCellular Fluid
NLLS	Non-Linear Least Squares Method

CHAPTER 1.

INTRODUCTION

1.1. Introduction

Nowadays there are several applications of Electrical Bioimpedance (EBI), these have appeared to respond the clinical needs as methods for non-invasive monitoring and the detection of changes in the structure and composition of body tissues produced by pathophysiological processes. Several applications based on EBI have the common step of fitting the EBI measured data to a model described by the Cole equation by estimating the Cole parameters.

1.2. Motivation

Humanity requires progress to diagnose and cure diseases. Within that progress, investigations using methods for signal analysis of EBI measurements could be used in healthcare applications like cancer detection, heart or brain monitoring and other sport and leisure applications for improve athletic performance.

The implementation of a software tool to implement EBI Spectroscopy Analysis that includes all kind of analysis methods is extremely useful for researchers to assess the information contained in EBI measurements.

1.3. Goal

The main goal of this thesis is to support the implementation of a Software Tool for Electrical Bioimpedance Spectroscopy Analysis. This tool should provide temporal and spectral analysis and Cole fitting.

A secondary aim is to test and assess the proper functionality of the implemented function analysis toolbox to perform EBI signal analysis and make comparisons between different spectra.

1.4. Work done

One of the tasks of this thesis work has been to produce analysis functions to support in the development of a Software tool for Electrical Bioimpedance Spectroscopy analysis. The other task has been to produce several functions and their respective assessments to perform a batch analysis of the EBI measurements of different subjects with the aim to select only the subjects, which present a Gaussian distribution in the of their respectively characteristic frequency.

1.5. Structure of the Thesis Report

This thesis report is organized in six chapters and the references. This chapter is the introduction to the performed thesis work. Chapter 2 gives a brief background to EBI, focusing in its frequency dependency, its electrical model and common artifacts in EBI measurements. Chapter 3 explains the developed software, including all the functions related with temporal and spectral analysis as well as Cole fitting, Cole rejection and special signal

analysis functions to specific purposes. Chapter 4 shows the results obtained with the special signal analysis functions. Chapter 5 discusses problematic aspects found along the process of design, implementation and the performance of implemented solution. Then at the end Chapter 6 presents the general conclusion and proposes future work to be done.

1.6. Out of Scope

Although spectroscopy analysis has been done with transcephalic measurements on adults and newborns, the main goal of this thesis work was to validate the implemented functions with such measurements. Therefore, a thorough spectroscopy analysis to obtain spectral reference values of EBI and Cole parameters from the set of transcephalic EBI spectroscopy measurements is completely out of the scope of this final degree work.

CHAPTER 2.

ELECTRICAL BIOIMPEDANCE

2.1 Introduction

Electrical Bioimpedance (EBI) is a measurement of the opposition to the flow of electric current the living tissues or biological material present. Thus, it is a common technology applied within medicine, with more than 60 years of successful applications in clinical investigations, physiological research and medical diagnosis (Schwan, 1999)

More recently, impedance measurements have been used in a number of applications such as the first application of bioimpedance techniques for monitoring applications like impedance cardiography or impedance plethysmography, where the electrical impedance is measured and used to determine the amount of fluid in the pleural cavities to detect deep vein thrombosis (Songer, 2001).

Since the first application bioimpedance measurements have been used in several medical applications; examples from a long list are lung function monitoring (Olsson et al., 1970), body composition (Kushner and Schoeller, 1986) and several kinds of cancer detection.

In the last 20 years other techniques like imaging method tissue, Bioimpedance Tomography also known as electrical impedance tomography, has been developed based on a method in which a series of electrodes are attached in a single plane to the chest or breast of the patient. An image of the tissue is then constructed based on the impedance information (Seoane, 2007).

2.2. Electrical Properties of Living Tissue

2.2.1 Electrical properties

The electrical properties of biological tissue are determined by its constituents. Any tissue is formed by extracellular fluid and cells containing the intracellular fluid inside the cell membrane. The extracellular fluid (ECF) is the medium surrounding the cells, hence denotes all body fluid outside of cells, also denominated the extracellular space.

The pericellular medium, cell and the extracellular space, contains water, and electrolytes that provides tissue with ionic conductance capabilities. On the other hand, the cell membrane constituted by a thin lipid bilayer plasma membrane, has capacitive properties that provides it with polarization capabilities (Zou and Guo, 2003).

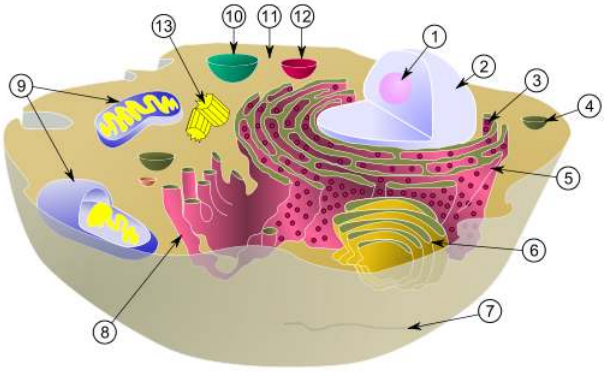

Fig 2.1 A living cell and its constituents: 1.Nucleous, 2 Nucleus, 3 Ribosome, 4 Vesicle, 5 Rough endoplasmic reticulum, 6 Golgi apparatus, 7 Cytoskeleton, 8 Smooth endoplasmatic reticulum, 9 Mitochondrion, 10 Vacuole, 12 Cytosol, 13 Lysosome, 14 Centriole

Due to free ions contained into intracellular and extracellular fluid, which are free to migrate and transport the electrical charge, we can consider almost any biological tissue as an electrolyte. Therefore, we can also consider tissue as ionic conductor, where K^+, $Na^+$and Ca^+ are the most important ions contributing to the ionic current. Table 2.1 contains the approximated concentration of the most common ions present in biological tissue.

Important cellular ionic concentrations

	Intracellular	Extracellular
$\boldsymbol{Na^+}$	10-20 mM	150 mM
$\boldsymbol{K^+}$	100 mM	5 mM
$\boldsymbol{Ca^+}$	10(-4) mM	1 mM

Table 2.1 Aproximate concentration of ions in living tissue [Guyton and Hall (2001)].

One of the most important constituents of the cell, as was mentioned before, is the plasmatic membrane, Fig 2.2. The intrinsic electrical conductance of this structure is very poor, in the order of 10^{-6}S/m and it is considered as a dielectric material. An important property of a dielectric is its ability to support an electrostatic field and therefore storage energy. The total structure formed by the intracellular fluid, plasma membrane and extracellular fluid forms a conductor-dielectric-conductor like a structure behaving as a capacitor, with an approximate capacitance of 1 μF/cm2.

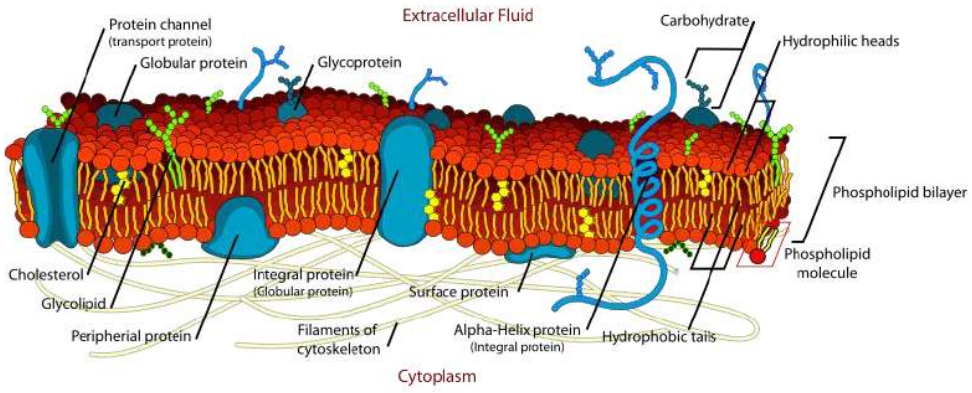

Fig 2.2 Schematic diagram of typical membrane proteins in a biological membrane (Hill, 2008)

2.2.2 Frequency dispersion

Biological tissue presents certain frequency behavior due to frequencial dependence of the permittivity and conductivity. Thus the frequency spectrum is not constant, presenting four transition regions, which are known as dispersion windows. The classification of the dispersion windows is based on the electrical examination of biomaterials as a function of frequency that is known as dielectric spectroscopy. H.P. Schwan divided the relaxation mechanisms initially in 3 groups, α-, β-, and γ-dispersion (Schwan, 1957) and later in 4 groups (Schwan, 1994) named δ-dispersion.

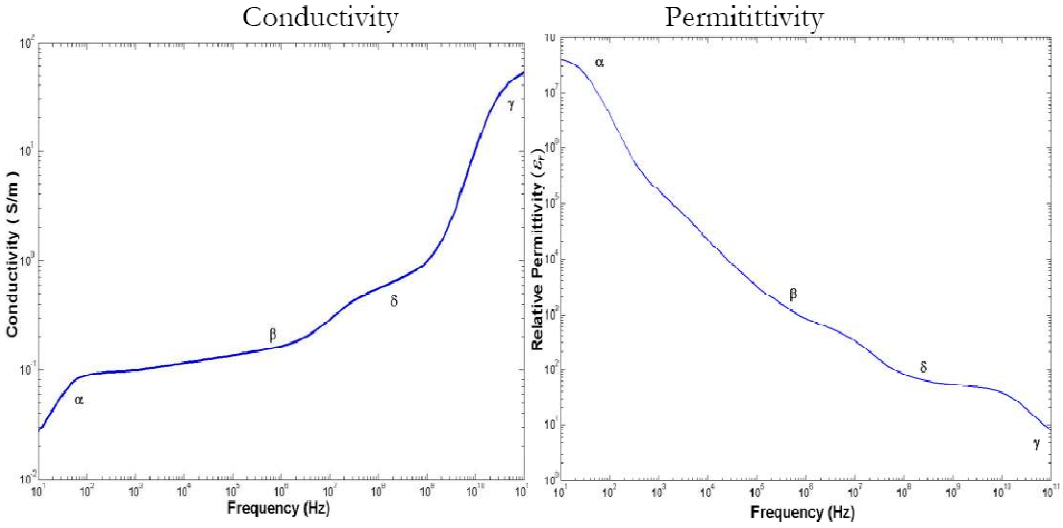

Fig 2.3 Frequency dependence of the conductivity and permittivity in the brain grey matter (Seoane, 2007)

2.2.2.1. α-dispersion

The α-dispersion appears at low frequencies, between 10 Hz –10 kHz. Although the elements that contribute to this frequency dependency are not clearly identified yet, (Schwan and takashima, 1993) established three main causes. First, the effect of the endoplasmic reticulum contributes to this frequency dependence. Second, the channel proteins present in the plasma membrane causes also the frequency-dependent conductance. Finally, the relaxation of counter-ions on the charged cellular surface is another mechanism that produces this frequency dependence.

2.2.2.2. β-dispersion

This dispersion is mainly due to the low conductivity and capacitive properties of the plasma membrane and other internal membrane structures and their interactions with the extra and intra-cellular electrolytes. It ranges from approximately 10 kHz to 100 kHz (Ivorra, 2003).

2.2.2.3 γ-dispersion

This frequency dependence is caused by the large content of water in cell and tissue. Tissue water is identical to normal water, which relaxes at 20 GHz, except for the presence of proteins and amino acids, etc. Tissue water displays a broad spectrum of dispersion from hundreds of MHz to some GHz.

2.2.2.4 δ-dispersion

The δ-dispersion is a minor additional relaxation between β and γ, it is caused in part by rotation of amino acids, partial rotation of charge side groups of proteins, and relaxation of protein-bound water that occurs between 300 and 2000 MHz (Schwan, 1957).

The following table shows the elements that contribute to the different kind of dispersions that have been mentioned above.

Contributing Biomaterial Element		Dispersion			
		α	β	γ	δ
Water and Electrolytes				●	
Biological Macromolecules	Amino acids		●	●	●
	Proteins		●	●	●
	Nucleic acids	●	●	●	●
Vesicles	Surface Charged	●	●		
	Non-Surface Charged		●		
Cells with Membrane	+ Fluids free of protein		●		
	+ Tubular system	●	●		
	+ Surface charge	●	●		
	+ Membrane relaxation	●	●		
	+ Organelles		●	●	●
	+ Protein		●	●	●

Table 2.3 Electrical Dispersions of Biological Matter (Schwan, 1994)

2.3. Electrical Model

2.3.1 Electrical Impedance

The electrical impedance *Z*, is a complex number with magnitude equal to the relation of magnitudes and phase equal to the difference of phases.

$$Z = V/I \Rightarrow \begin{matrix} |Z| = |V|/|I| \\ Z = V - I \end{matrix} \quad \textbf{(2.1)}$$

The real part of the impedance is the Resistance while the imaginary part is the Reactance. The resistive part causes the power loss ($R = \{Z\}$ whereas the reactance causes the delay between voltage and current ($X = \mathbb{I}\{Z\}$)

$$Z = R + jX\ [\Omega] \quad \textbf{(2.2)}$$

2.3.1.1 Impedance of a resistance

A resistance obeys the Ohm´s law per definition. Thus, the only relation between voltage and current can be a relation of magnitudes.

$$Z = \{Z\} = R = V/I \quad \textbf{(2.3)}$$

2.3.1.2 Impedance of a capacitance

For a capacitance, the current is proportional to the time derivate of voltage. This means that the Ohm´s law as we expressed before is no longer valid.

$$Z = -j\frac{1}{2\pi fC} \quad \textbf{(2.4)}$$

Hence, the capacitance impedance depends on frequency (f) and is purely reactive (phase angle = $-90°$).

According to the last expressions, we can say that a capacitance behaves as a resistance, with value $\frac{1}{2\pi fC}$: an open-circuit (no conductance) for very low frequencies and a short-circuit for high frequencies. Another way to say this is:

"In a capacitance, high frequency currents are free to flow and low frequency currents are blocked." (Ivorra, 2003).

The impedance values are not only determined by the electrical properties of the material, conductivity and permittivity, but are also by the geometrical constrains. In general, the values of interest will be the electrical properties of the materials since they will be not dependent on the geometry used in each study.

$$Y = G + jB = G + jwC \equiv K(\sigma + j\omega\varepsilon) \quad \textbf{(2.5)}$$

- Admitance ,Y=1/Z is the inverse of the admittance.
- Conductance, G is the real part of the admittance (Siemens (S)=(1/Ω)).
- Susceptance, B is the imaginary part of the admittance (Siemens (S)=(1/Ω)).
- K is the scaling factor of the measurement cell = area/length ($\frac{cm^2}{cm} = cm$).
- Conductivity of the material σ (S/cm).
- Permittivity of the material ε (F/cm).

2.3.2 Electrical Circuit of the cell

As previously mentioned, electrical properties of tissue are given by its constituents. Therefore considering these constituents, applying theory of electrical circuits and simplifying, an electrical equivalent model for the cell can be elaborated (Fricke, 1924). The model is depicted in figure, Fig 2.4 c).

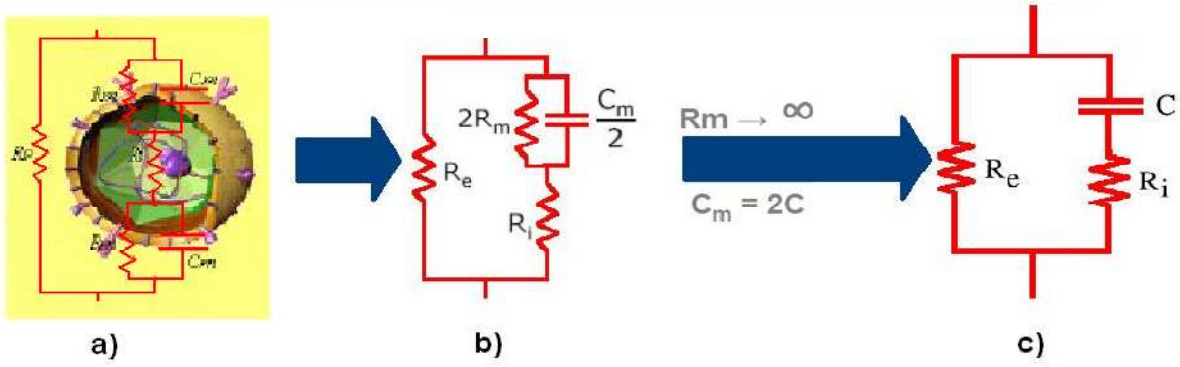

Fig 2.4 Equivalent circuit of a cell where *Re* is the extracellular fluid Resistance, *Ri* the intracellular fluid Resistance, *Rm* the trans-membrane ionic channel Resistance and *Cm* represents the cell membrane Capacitance

The capacitor *Cm* represents the membrane, *Rm* represents the resistance of the ionic channels (high value due to their low conductivity) and *Re* and *Ri* represent the extra cellular and intracellular fluids respectively. In this model, the resistance of the membrane has been neglected due to its extremely large value.

The impedance spectrum of a cell according to Fricke's model is given by the following equation:

$$Z = \frac{R_e(1+jR_iC\omega)}{1+jC\omega(R_i+R_e)} \quad \textbf{(2.6)}$$

According to this simplified model, the electrical behavior at high and low frequencies can be explained as follows:

- At low frequencies current does not flow through the cells, because the cell membrane acts as a capacitor. In this case, the impedance is reduced to R_e. Such blocking effect decreases with increasing frequency.

$$\omega \to 0 \Rightarrow Z_0 = R_e \quad \textbf{(2.7)}$$

- At high frequencies, the capacitance decreases and current flows through the cell. At very high frequencies, the impedance becomes the parallel connection of R_e and R_i.

$$\omega \to \infty \Rightarrow Z_0 = \frac{R_eR_i}{R_e+R_i} \quad \textbf{(2.8)}$$

On the other hand, this equivalent circuit proposed by Fricke is not adequate to simulate tissue in a general way. It was checked by (Kanai H, 1983) but but it was just correct for blood because it contains one dominant cell species (Jaffrin MY, 1997). In fact, human tissue contains different types of cells and in this case, the Cole model (Cole, 1940) is more general. It generalizes Fricke model being valid for tissue containing different types of cell species, but lacks an electrical representation.

2.3.3 Cole model

The Cole model is a function that shows the behavior of electrical impedance of biological tissue. This model consists of three parts: an equation, an equivalent circuit, and a complex impedance circular arc (Grimnes and Martinsen, 2005).

The equation (2.9) is the Cole empirical equation for the frequency dependence of tissue or cell suspension complex impedance. This equation is not only commonly used to represent EBI data, but also it is often used to analyze the obtained EBI measurements. The analysis is based in the four parameters contained in the Cole equation *R0*, *R*∞, α and τ that is the inverse of characteristic frequency, ω_c.

$$Z = R_\infty + \frac{R_0-R_\infty}{1+(j\omega\tau)^\alpha} \quad \textbf{(2.9)}$$

Where Z is the complex impedance expressed in Ohms [Ω], the resistance [Ω] at very high and very low frequencies are respectively R_∞ and R_0, j is the imaginary unit, ω is the

angular frequency [1/s], τ is the characteristic relaxation time constant of the system [s] and α is an exponent [dimensionless] with values between 0 and 1, being this last the typical for a single dispersion.

Furthermore the equivalent electrical model is based on the replacement of an ideal capacitor in the Debye model, shown in the figure Fig 2.5, with a more general constant phase element (CPE).

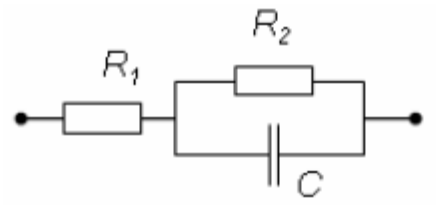

Fig 2.5 Debye single dispersion electrical model for human tissue**.**

The relation between the empirical equation (2.9) and the last model, Fig 2.5, are: $R_0 = R_1 + R_2$, $R_\infty = R_1$ and $\tau = R_2 C$.

The most used parametric plot to represent the impedance is the Cole Plot. The obtained curve is not the original measured data but it is a curve fitted to the mathematical Cole equation and the angular frequency as independent variable, as in (2.9).

In this plot, the resistive part (R) (in the horizontal axis) is plotted against the conjugate part of the reactance (X*), (in vertical axis). This plot is a semicircle with approximated radius $(R_0 - R_\infty)/2$ which crosses the real axis R_0 and R_∞. Moreover this semicircle is depressed in its center having the imaginary center below the resistance axes. The grade of this depression is determined by the parameter α, obtaining a perfect semicircle when α equals 1. See figure, Fig 2.6.

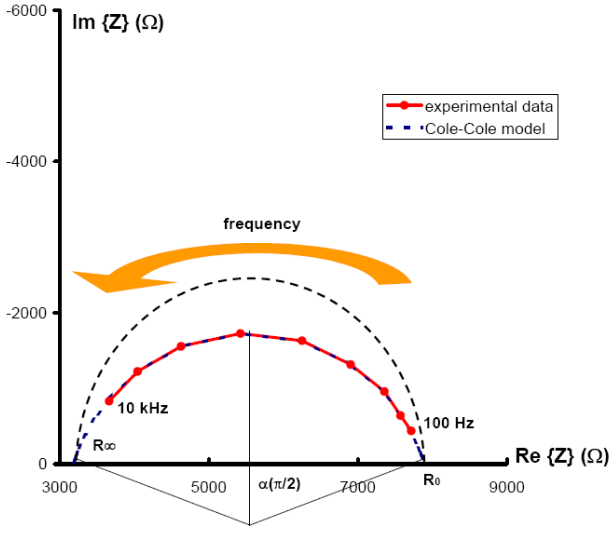

Fig 2.6 Cole plot (Ivorra, 2003)

2.4 Common artifacts in the measurements

2.4.1 Capacitive Leakage

The Hook or Tail Effect, in regards to the field of Electrical Bioimpedance (EBI) is the deviation suffered mostly by the reactance and the phase. The effect is most noticeable at high frequencies by the increasing reactance in the impedance plot.

As it is shown in figure Fig 2.7, the impedance plot of the impedance presents a 'hook-a-like' deviation at the lowest values of the impedance, R_{∞}. These values correspond with the high frequencies in an EBI system.

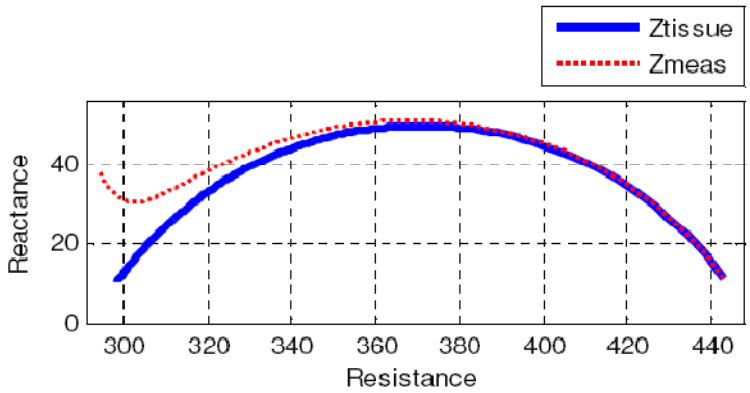

Fig 2.7 Cole plot. In Blue the Cole system presents ideal behavior and in Red the plot presents a deviation from the ideal, that is the denominated Hook Effect at the lowest values of impedance. (Buendia, 2009)

The presence of a Hook Effect in the EBI data influences the Cole fitting algorithms that estimate the Cole equation (2.9) from the EBI measured data. Cole fitting is a process that intends to fit the measured EBI into the Cole equation, which is a system with a single dominant dispersion, while EBI data containing Hook Effect contains two dominant dispersions. Ideally, the effect of the capacitive leakage should be removed, corrected or compensated prior to any attempt of Cole fitting process. The design and performance of the correction function are further explained in the next chapters.

CHAPTER 3.

TOOLBOX

3.1 General Overview

One of the aims of this thesis work is to produce analysis functions to support the development of a Software tool for EBI Spectroscopy analysis. Such application has been developed in Matlab and its respective graphical user interface has been developed by Alex Rodríguez in the final thesis project "*Development of a Software Application Suite for Electrical Bioimpedance Data Analysis*" using the Graphical User Interface tool GUIDE supplied by Matlab.

The analysis tool provides spectral and temporal analysis, Cole fitting as well as a rejection function and visualization of different kinds of plots such as the Cole plot and the following immitance variables against frequency: Resistance, Reactance, Susceptance, Module and Phase. The application also enables doing: analysis of spectral features for signal classification, calculation of impedance index, BIVA analysis and histograms of the characteristic frequency.

Since another goal of this thesis was to test the suitability of the implemented function analysis toolbox, several functions for specific purposes have been implemented and they are explained in the following subsection of special functions.

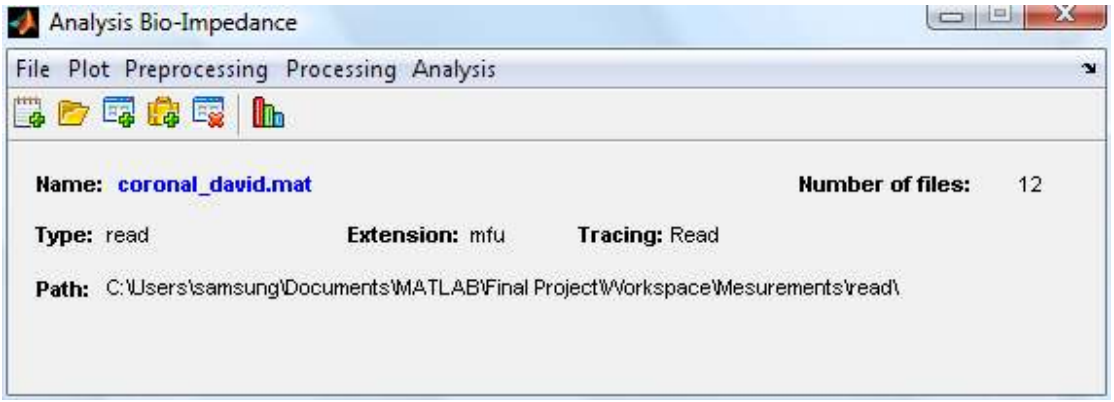

Fig 3.1 Screen shot of the application

3.2 Functions for the Software Tool

This section presents the functions that have been designed for the Matlab analysis tool. The functions that are designed in the framework of this thesis are extensively explained. The following scheme in Fig 3.2 shows the names and the relationship between such functions. The goal of this report is not to explain how to use the application. The user manual is fully explained in "*Development of a Software Application Suite for Electrical Bioimpedance Data Analysis*".

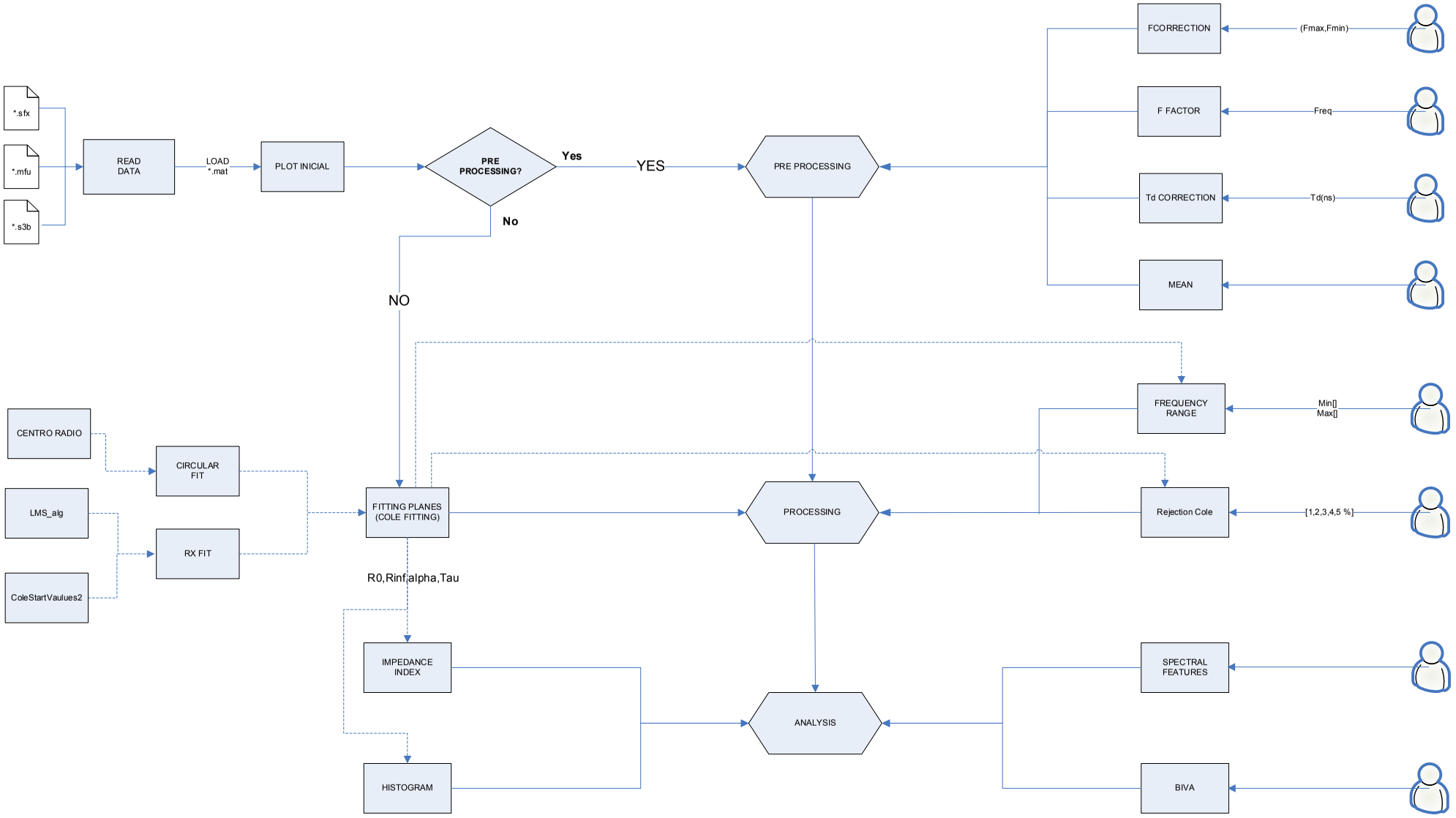

Fig 3.2 Flux Diagram of the Software Tool

3.2.1 Correction Function

As introduced in the thesis *"Hook Effect on Electrical Bioimpedance Spectroscopy Measurements. Analysis, Compensation and Correction"* (Buendia, 2009); in order to obtain a total correction of the capacitive leakage present on the EBI measurement Zmeas(ω), we have to consider the correction factor as a complex function of frequency FCorr(ω). This way the imaginary part will modify the module and the real will modify the phase.

$$F_{Corr}(\omega) = \frac{Log\left|1 - j\omega C_{stray} Z_{meas}(\omega)\right|}{j\omega} \quad \textbf{(3.1)}$$

By multiplying the obtained EBI measurement Zmeas(ω) by a complex exponential function defined by the Fcorr(ω) function times -jω, the EBI corrected Zcorr(ω) can be obtained as indicated in (3.2).

$$Z_{Corr}(\omega) = Z_{meas}(\omega) * e^{-j\omega F_{Corr}(\omega)} \quad \textbf{(3.2)}$$

Substituting (3.1) in (3.2) it is possible to obtain the final expression of Zcorr(ω) in function of the EBI measurement Zmeas(ω) and the parasitic capacitance Cstray

$$Z_{Corr}(\omega) = Z_{meas}(\omega) * e^{-Log\left|1 - j\omega C_{stray} Z_{meas}(\omega)\right|} \quad \textbf{(3.3)}$$

The equations above are used in the matlab code as shown on the Code box 3.1, as we can see in lines (5) and (6) colored in gray. Furthermore, another important aspect to take into account in this function, it is the way to find the Cstray from the Susceptance in the range of frequencies where the effect is especially noticeable. Therefore we can chose that range thanks to the function *frecrow*, lines (1) and (2), this function finds the number of row that is closer to the input value of the frequency and we can see the implementation in Code 3.2.

As it is observed in the following Figure, from already 200 KHz we can notice the effect of the parasite capacitance in parallel with the measurement load. Linearizing the red line through a polynomial curve fitting, it is possible to obtain the value of Cstray as the slope of the line. The estimated value of Cstray is the value of A in equation (3.4). This method is implemented in the code of the functions in the lines (3) and (4).

$$y = A\omega + B \quad \textbf{(3.4)}$$

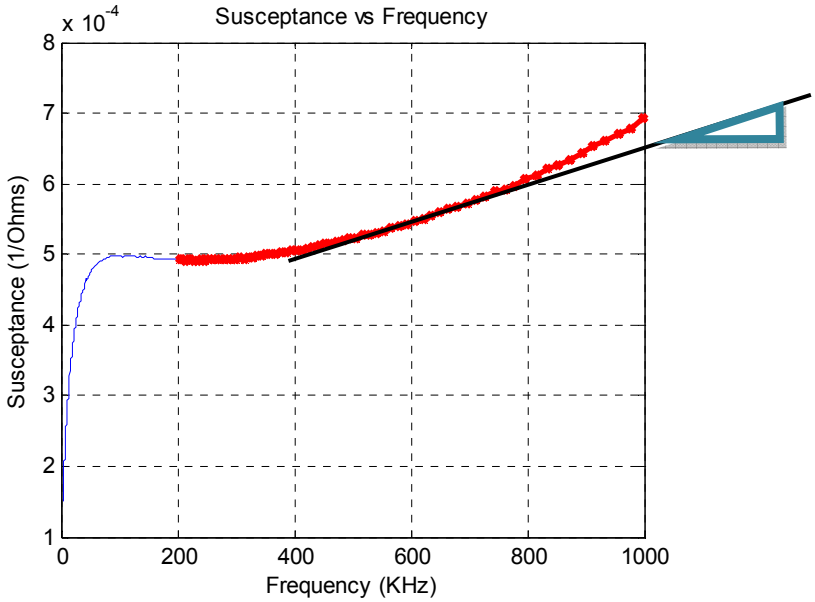

Fig 3.3 Susceptance with capacitive leakage effect and its slope.

```
function[] = Fcorrection(DataArray,Fmin,Fmax)

numfiles = size(DataArray,3); % To know the max value of files
[rows,columns] = size(DataArray); % To know the max value of rows and columns

for i=1:numfiles %Number of files
    for j=1:rows  %Number of rows
        Rmeas(j,i) = DataArray(j,2,i); % Safe the value of resistance
        Xmeas(j,i) = DataArray(j,3,i); % Safe the value of reactance
        Zmeas(j,i) = Rmeas(j,i)+(sqrt(-1)*Xmeas(j,i)); % Calculate the impedance
        Ymeas(j,i) = 1./Zmeas(j,i); % Calculate the admitance
        Smeas(j,i) = imag(Ymeas(j,i)); % Calculate the susceptance
        Wmeas(j,i) = 2*pi*(DataArray(j,1,i))*1000; % Calculate the frequency in radians
    end
end

for i=1:numfiles %Number of files
    %Choose start and final samples for linealize
    N = frecrow(Fmin,rows,DataArray); (1)
    M = frecrow(Fmax,rows,DataArray); (2)

    for j=N:M
        Smeascorr(j-N+1,i) = Smeas(j,i);
        Wmeascorr(j-N+1,i) = Wmeas(j,i);
        if j==M
            y = polyfit(Wmeascorr(:,i),Smeascorr(:,i),1); (3)
            Cstray(i) = y(1); (4)
        end
    end

    for j=1:rows
        Fcorr(j,i) = (log(1-(sqrt(-1)*Wmeas(j,i)*Cstray(i)*Zmeas(j,i)))./(sqrt(-1)*Wmeas(j,i)); (5)
        Zcorr(j,i) = Zmeas(j,i)*exp(-(sqrt(-1)*Wmeas(j,i)*Fcorr(j,i))); (6)


    end
    end
end
```

Code 3.1 Matlab Code of *Correction Function*

```
function[out]=frecrow(Frec,rows,d)


continu=1;

for cont=1:rows
    valor=d(cont,1,1);
    if((valor>=Frec) && (continu==1))
        out=cont;
        continu=0;
    end
end
end
```

Code 3.2 frecrow function finds the number of the row corresponding to the frequency.

3.2.2 Factor Correction

The following function in Code 3.3 and 3.4 makes corrections of the capacitive lekeage using the Correction Function method for a specific value of frequency, *fcorr*, introduced by the user, line (1). The value of Cstray is calculated in the same way that was explained before, only with the difference that we take all the range of frequencies, lines (2) and (3).

As shown the line (1), we use the function *frecrow* to know exactly the number of row corresponding to the input frequency. This value is going to be our *fcorr*. The resulting Zmeas(ω) is multiplied by the complex exponential, $e^{-j\omega fcorr(\omega)}$ $e^{-j\omega fcorr}$ $e^{-j\omega fcorr}$. The obtained result is made equal to the original expression of Zcorr(ω) to solve for the factor *fcorr*, line (4).

Notice that in order to solve for a value of *fcorr* a specific value of frequency has been required. This means that such specific value is expected to correct perfectly only at the specific frequency. The results are shown in figures, Fig.3.4 and 3.5.

```
function[] = Ffactor(DataArray, Freq)

numfiles = size(DataArray,3);
[rows,columns] = size(DataArray);


N = frecrow(Freq,rows,DataArray);                                          (1)


    N = frecrow(Fmeas(1,i),rows,DataArray);     % The value of N is the first row
    M = frecrow(Fmeas(rows,i),rows,DataArray); % The value of M is the last row
    for j=N:M
        Smeascorr(j-N+1,i) = Smeas(j,i);
        Wmeascorr(j-N+1,i) = Wmeas(j,i);
        if j==M
            y = polyfit(Wmeascorr(:,i),Smeascorr(:,i),1);                  (2)
            Cstray(i) = y(1);(3)
        end
    end
```

Code 3.3 Matlab code of *Factor Correction* function part 1

```
    for j=1:rows
        Fcorr(j,i) = (log(1-(sqrt(-1)*Wmeas(j,i)*Cstray(i)*Zmeas(j,i)))./
(sqrt(-1)*Wmeas(j,i));
    end
    for j=1:rows
        …
        Zcorr(j,i) = Zmeas(j,i)*exp(-(sqrt(-1)*Wmeas(j,i)*Fcorr(N,i)));          (4)



    end
end
```

Code 3.4 Matlab code of *Factor Correction* function part 2

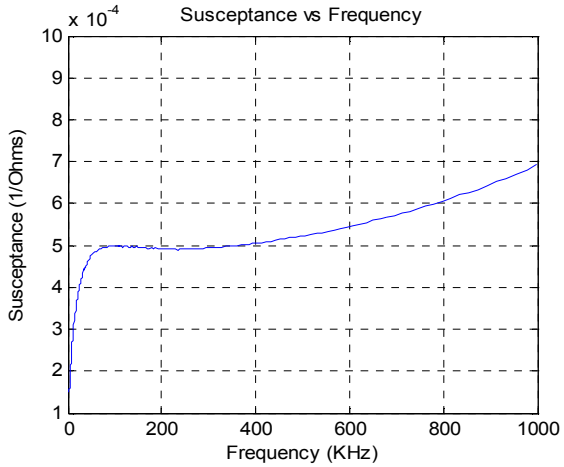

Fig 3.4 Susceptance without correction

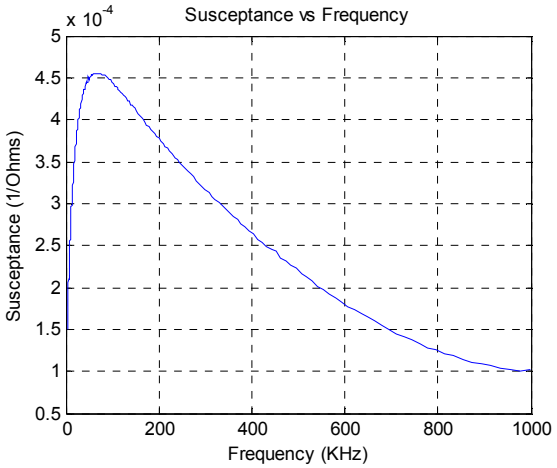

Fig 3.5 Susceptance with correction at 300 Khz

3.2.3 Td Compensation

This function performs a Time Delay (*Td*) compensation with a *Td* value introduced by the user or with an assessment of multiple values of *Td* that can be chosen automatic mode. As shown in line (1) of the following code box, the exponent is imaginary. Thus it is deduced that a scalar *Td* will modify only the phase of Zmeas(ω), leaving the module unchanged. This is proven on figures, Fig.3.6 and 3.7.

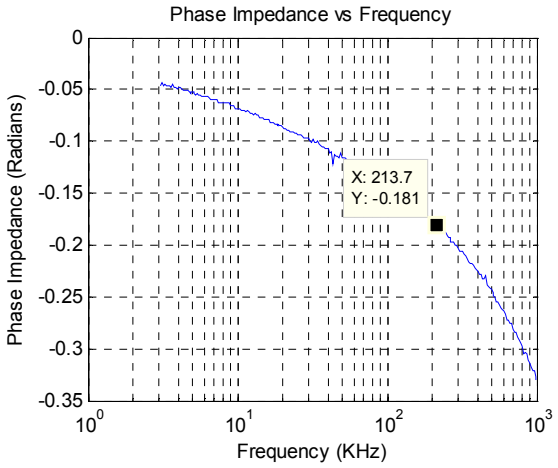

Fig 3.6 Phase without correction

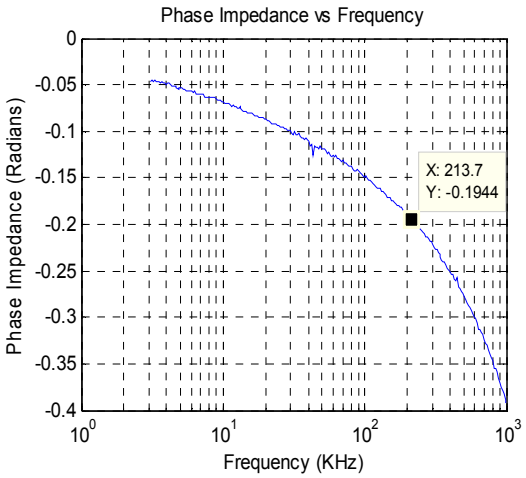

Fig 3.7 Phase with a correction of 10 ns

```
function[] = Tdfactor(DataArray ,Td)

 [rows,columns] = size(DataArray);

switch Td
    case '1ns'
        td = 1E-9;
    case '5ns'
        td = 5E-9;
    case '10ns'
        td = 10E-9;
    case '15ns'
        td = 15E-9;
    case '20ns'
        td = 20E-9;
end



for i=1:numfiles %Number of files
    for j=1:rows
        Zfactor(j,i) = Zmeas(j,i)*exp(-(sqrt(-1)*Wmeas(j,i)*td)); (1)


    end
end

end
```

Code 3.5 Matlab code of the *Td Compensation* function

3.2.4 Mean

The following function calculates the mean of all the files containing a *.mat extension file. Such mean is calculated row by row. Consequently, if we have the struct *DataArray(256,3,20)* with 20 files, then we have to pass each position of the row of each file as a parameter of the mean function, for instance of the resistance (line (1), we have the mean of all the rows and all the files.

Finally, we save all the arrays of frequency, resistance and reactance in the struct DataArray, lines (2) to (3).

```
function[namefile,pathtemp] =
DataMean(DataArray,ext,namefile,namesource,pathsourcefile,type,tracing)


 [rows,columns] = size(DataArray);

meanfrec = zeros(1,length(DataArray(:,1,1)))';
meanresi = zeros(1,length(DataArray(:,1,1)))';
meanreact = zeros(1,length(DataArray(:,1,1)))';

for i=1:1:rows
    Mean.meanfrec(i) = mean(DataArray(i,1,:));
    Mean.meanresi(i) = mean(DataArray(i,2,:));                    (1)
    Mean.meanreact(i) = mean(DataArray(i,3,:));
end

clear('DataArray');

DataArray(:,1) = Mean.meanfrec(:);                                (2)
DataArray(:,2) = Mean.meanresi(:);
DataArray(:,3) = Mean.meanreact(:);                               (3)


end
```

Code 3.6 Matlab code of *Mean* function

3.2.5 Cole Fitting

The function *Fitting_planes* calculates a fitted theoretical curve in four different models, this function was developed in the thesis "*Methods for Cole Parameter Estimation from Bioimpedance Spectroscopy Measurements. A comparative Study"*, (Ayllón *et al.*, 2009). Three of the models are implemented in the *RXfit* function and the last one in the function *Circularfit*.

RXfit function

The first three models are *R(ω), jX(ω) and R(ω) + jX(ω)* and we can see their implementation in the function *RXfit* in lines (5) to (7) in Code 3.7, which is used by the function *Fitting_planes*. These models use the decomposition of the complex Z from the Cole equation in (2.9) into the real part, the resistance R and imaginary part, the reactance X, in (3.6) and (3.7) respectively and applying the NLLS method, the fit function has been evaluated with those three different models according to the following equations.

Moreover the natural frequency ω is the independent variable, for the curve fitting and the estimation of the Cole parameters R_0, R_∞, α, and τ as the model coefficients.

$$j^{\propto} = cos(\alpha\,\pi/2) + jsin(\alpha\,\pi/2) \tag{3.5}$$

$$R(\omega) = R_\infty + \frac{(R_0 - R_\infty)\left(1+(\omega\tau)^\alpha cos\left(\alpha\frac{\pi}{2}\right)\right)}{1+2(\omega\tau)^\alpha cos\left(\alpha\frac{\pi}{2}\right)+(\omega\tau)^{2\alpha}} \tag{3.6}$$

$$X(\omega) = -j + \frac{(R_0 - R_\infty)(\omega\tau)^\alpha sin\,(\alpha\frac{\pi}{2})}{1+2(\omega\tau)^\alpha cos\left(\alpha\frac{\pi}{2}\right)+(\omega\tau)^{2\alpha}} \tag{3.7}$$

The initial values are calculated in the function *ColeStartValues2*. Such function and its code for the model coefficients are explained according to section *5.1.2 Cole Start Values*.

The other values are shown from the lines (1) to (4) in Code 3.7 and there are respectively: the stop tolerance involving the model value and the stop tolerance indicating the coefficients that have been set to its default value of 10^{-6}. The maximum number of model evaluations is set as 6000 and the maximum number of fit iterations as 10^7 instead of the old value that was 400. Lower bounds for R_0, R_∞ and τ are 0, and for α is 0.5. Upper bounds for R_0, R_∞ and τ are 1.5 times their estimated initial value, and for α is 1.

NLLS methods are used to implement the curve fitting in the function *RXfit* calling to function *fit*, lines (8) to (10) in Code 3.8.

The explanation of the performance of such method and the convergence criterion are found in the Master degree thesis of (Ayllón *et al.*, 2009). Although the values for the convergence criterion are correct in most of the cases, a change for the maximum number of fit iterations is necessary. For instance, performing consecutive fittings with a great deal of EBI measurement data files contained in the *DataArray(:,:,:)* structure, more iterations are needed. Therefore the maximum number of fit iterations is set as 10^7.

As we can see in the code, the input variable of the functions are:

- w: input frequency in Hz.
- Rdata: input resistance.
- Xdata: input reactance.
- st_: starting values from coleStartValues2 function.
- displayON:if this value is equal 1, the fittings are plotted.

- Zest: Estimation of the Z that belongs to the Circular Fit method.

The output variable in the code is:

- cfR/cfX/cfRX: all the four coefficients R_0, R_∞, τ and α for each method.

To conclude with this function, only to remark that the code relative to the plotting for each model is not included in the following figure.

```
function [cfR,cfX,cfRX] = RXfit(w,Rdata,Xdata, st_, displayON,Zest)


%Data for R+X fit
RXdata = Rdata + Xdata;

if displayON==1
    % Set up figure to receive datasets and fits

    % Plot data originally in dataset "Rdata vs. w"

end

%_____________________________________________________________________%
% FITTING
%_____________________________________________________________________%

%Fit options

    taumax=1.5*st_(3);          (1)
    romax=1.5*st_(1);           (2)
    rimax=1.5*st_(2);           (3)
    fo  = fitoptions('method','NonlinearLeastSquares','MaxFunEvals', 6000,'MaxIter',
1000e3,'Lower',[0 0 0 0.5 ], 'Upper', [romax rimax taumax 1]); %limit values
    ok_ = ~(isnan(w) | isnan(Rdata));                                          (4)
    set(fo_,'Startpoint',st_);

%Fit model R
     fitModelR = fittype('b + ((a-b)+(a-
b)*cos(d*pi/2)*(w*c)^d)/(1+2*cos(d*pi/2)*(w*c)^d+(w*c)^(2*d))',...
     'dependent',{'Rfit'},'independent',{'w'},...
     'coefficients',{'a', 'b', 'c', 'd'});                                     (5)

 %Fit model X
     fitModelX= fittype('((a-b)*sin(d*pi/2)*(w*c)^d)/(1+2*cos(d*pi/2)*(w*c)^d+(w*c)^(2*d))',...
     'dependent',{'Xfit'},'independent',{'w'},...
     'coefficients',{'a', 'b', 'c', 'd'});                                     (6)

 %Fit model R+X
     fitModelRX= fittype('b + ((a-b)+(a-
b)*cos(d*pi/2)*(w*c)^d)/(1+2*cos(d*pi/2)*(w*c)^d+(w*c)^(2*d))+((a-
b)*sin(d*pi/2)*(w*c)^d)/(1+2*cos(d*pi/2)*(w*c)^d+(w*c)^(2*d))',...
     'dependent',{'RXfit'},'independent',{'w'},...
     'coefficients',{'a', 'b', 'c', 'd'});                                     (7)
```

Code 3.7 Matlab code of the use of the *fit* functions in *RXfit* function part 1.

```
% R Fitting
    [cfR,goodnessR,outputR]=fit(w(ok_),Rdata(ok_),fitModelR,fo_);          (8)

    if displayON==1
    % Plot R fit
    end

% X Fitting
    ok_ = ~(isnan(w) | isnan(Xdata));
    [cfX,goodnessX,outputX] = fit(w(ok_),Xdata(ok_),fitModelX,fo_);        (9)

     if displayON==1
     % Plot X fit
     end

%R+X Fitting
    ok_ = ~(isnan(w) | isnan(RXdata));
    [cfRX,goodnessRX,outputRX] = fit(w(ok_),RXdata(ok_),fitModelRX,fo_);   (10)

     if displayON==1
     % Plot R+X fit

     end


%Z Fitting


     if displayON==1
     % Plot Z fit
     end
end
```

Code 3.8 Matlab code of the use of the fit functions in RXfit function part 2.

3.2.5.1 Circularfit function

On the other hand, there is another possibility to estimate the Cole parameters, this fourth approach consist on fitting the EBI measured data to the a semicircular plane in the impedance plane.

This model takes the consideration that the Cole function creates in the impedance plane a perfect semicircle with the centre depressed below the resistance axis. In this way it estimates the complex centre and radius by an approach based in obtaining a complex centre C and radius R that produces a set of semicircular points which its variance of its squared distance to each point from the measurement set is minimum.

The whole analysis of this novel method is extensively tackled in "*Bioimpedance Spectroscopy Measurements.A Comparative Study"* by (Ayllón *et al.*, 2009). Nevertheless we are going to present the equations that are useful to explain the Matlab code for the functions *CentroRadio* and *Circularfit*.

The equations concerning the function *CentroRadio* that calculates the centre C and radius R of the semicircle from the input D impedance are explained in the next paragraphs, while the respective implementation of the function is shown in Code 3.9.

The distance R_n from the centre C=x+jy to the impedance $Z_n = X_n + jY_n$ is expressed as in equation (3.8) and then with consecutive transformations expressing the mean squared distance and then finding *x* and *y* through the variance from (3.9) that is minimum.

$$R_n{}^2 = |C - Z_n|^2 = (x - X_n)^2 + (y - Y_n)^2 = x^2 + y^2 + X_n{}^2 + Y_n{}^2 - 2xX_n - 2yY_n \quad \textbf{(3.8)}$$

$$Var\left\{\check{R}^2\right\} = \frac{1}{N}\sum_{n=1}^{N}(R_n{}^2 - \check{R}^2)^2 \tag{3.9}$$

For this purpose, the variance is derived in respect to x and y and both expressions are equaled to zero and solved by the following system of equations:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2\sum_{n=1}^{N} XC_n{}^2 & 2\sum_{n=1}^{N} XC_n Y\,C_n \\ 2\sum_{n=1}^{N} XC_n Y\,C_n & 2\sum_{n=1}^{N} YC_n{}^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_{n=1}^{N}(X2\,C_n + Y2C_n)XC_n \\ \sum_{n=1}^{N}(X2\,C_n + Y2C_n)YC_n \end{pmatrix} \tag{3.10}$$

Once we have the expression (3.10), which corresponds in the line (1) in Code 3.9 is a regression function of a circle in the complex plane. Moreover, the function returns the complex center (C), that we can find in line (2), and the real radius (R) The return value minimizes the variance of the distances from the center to all the points, line (3).

```
function [C,R]=CentroRadio(D)

X=real(D);
Y=imag(D);
Xm=X-mean(X);
Ym=Y-mean(Y);
X2m=X.^2-mean(X.^2);
                    Y2m=Y.^2-mean(Y.^2);

M=2*[sum(Xm.^2)  sum(Xm.*Ym);sum(Xm.*Ym)  sum(Ym.^2)];
V=[sum((X2m+Y2m).*Xm);sum((X2m+Y2m).*Ym)];

y=inv(M)*V;                                          (1)

C=y(1)+i*y(2);                                       (2)
R=sqrt(mean((X-y(1)).^2+(Y-y(2)).^2));               (3)
```

Code 3.9 Matlab code of *CentroRadio* function

Finally, we are going to present the equations regarding to the function *Circularfit* in Code 3.10.

The following equations, which are referred to lines (2) and (3) of the code, take the complex centre (C) and the real radius (R) from the *CentroRadio* function, line (1), to calculate the estimations of R_∞ and R_0.

$$R_\infty = R\{C\} - \sqrt{R^2 - \mathbb{I}\{C\}^2} \tag{3.11}$$

$$R_0 = R\{C\} + \sqrt{R^2 - \mathbb{I}\{C\}^2} \tag{3.12}$$

The coefficient α, can be obtained from the slope that forms *C*- R_∞, equation (3.13) and lines (3) and (4). Once α is obtained, the value of f_c can be solved from the Cole equation, equation (3.14) and lines (5) to (7). These operations take the result (C) and (R) from the *CentroRadio* function like the previous ones.

$$\alpha = 1 \pm \frac{2}{\pi} arctan\left(\frac{\mathbb{I}\{C\}}{\sqrt{R^2 - \mathbb{I}\{C\}^2}}\right) = 1 \pm \frac{2}{\pi} arctan\left(\frac{1}{\sqrt{\left(\frac{R}{\mathbb{I}\{C\}}\right)^2 - 1}}\right) \quad \textbf{(3.13)}$$

$$f_c = jf\left(\frac{R_0 - R_\infty}{Z - R_\infty} - 1\right)^{\frac{1}{\alpha}} \quad \textbf{(3.14)}$$

In addition, with the result obtained for f_c we can easily find the value for τ in line (8) of the code. Finally, with all the coefficients found, the value for Z_{est} in this method can be taken out with line (9).

```
function [cfZ,Zest] = Circularfit(fmeas,Zmeas)

% to calculate fc
W=fmeas/2/pi;
Q=[ones(size(W));W;W.^2];
M=Q'/(Q*Q');

%center and radius of the circunference
[C,R]=CentroRadio(Z);                                          (1)

% Rinf, R0 y alfa
EstRinf=real(C)-sqrt(R^2-imag(C)^2);                           (2)
EstR0=real(C)+sqrt(R^2-imag(C)^2);                             (3)
EstAlfa=1-atan(imag(C)/sqrt(R^2-imag(C)^2))/pi*2;
if EstAlfa>1
 EstAlfa=1+atan(imag(C)/sqrt(R^2-imag(C)^2))/pi*2;             (4)
end


% fc
aux=(abs(sqrt(-1)*W.*((EstR0-EstRinf)./(Zmeas-EstRinf)-1).^(-1/EstAlfa)));  (5)

%Regression curve of minimun squares
V=aux*M;                                                       (6)
Estfc=abs(V(1));                                               (7)

EstTau=1/Estfc/2/pi;                                           (8)

cfZ=[EstR0 EstRinf EstTau EstAlfa];

%error
Zest=EstRinf+(EstR0-EstRinf)./(1+(sqrt(-1)*W/Estfc).^EstAlfa);  (9)


end
```

Code3.10 Matlab code of *Circularfit* function

3.2.5.2 Fitting_planes function

This function performs the four fitting planes [*R(ω), jX(ω), R(ω) + jX(ω) and Zplane*] previously presented and shows the results. Furthermore, we can choose the frequency range that we want to perform the Cole fitting, thus we only need to put the variable *If_freq_range=1* equal one and indicate the vector of minimum and maximum frequencies *Arrayfreq=[fmin,fmax]* in the header. The implementation in the code is possible to follow from line (1) to (3) in Code 3.11.

As we can see in the code, the input variables of the functions are:

- finput: input frequency in Hz.
- Rmeas: input resistance.
- Xmeas: input reactance.
- displayON: if this value is equal 1, the fittings are plotted.
- If_freq_range: previously commented.
- Arrayfreq: previously commented.

The output variable in the code is:

DataArray: an output structure containing all the coefficients and estimated impedances.

The sign of the input reactance Xdata, line (2) and (4), has to be changed in *RXfit* function, line (5), to make the Cole plots concave

Another important feature to comment, is the relationship between the variables coef in each plane, line (7) to (11), and the characteristic coefficients. In the following table we can see the correspondence with the characteristic coefficients.

coefZ(1)	**coefZ(2)**	**coefZ(3)**	**coefZ(4)**
$\boldsymbol{R_0}$	R_∞	τ	α

Table 3.1 Table of correspondence between variables coef of the code and coefficients in Zplane

The impedance in each plane is extracted from the Cole equation, for example line (6) in *R(ω)* plane.

As explained before, the output structure *DataArray* contain all the important variables that we need to save. Among these variables there are the characteristic coefficients, the impedance estimation in each plane, line (12), the central frequencies in each plane, line (13) and the same original values of frequency, resistance and reactance that were in the header, line (14) to (15).

Finally, when the input argument *display_ON=1* is stated, the function plots all the different obtained fitting models.

```
function [DataArray]= Fitting_planes(finput,Rmeas,Xmeas,displayON,,If_freq_range,Arrayfreq)

if(If_freq_range==1)                                                    (1)

    A=Arrayfreq';

    % selecting frequency range to use in the fitting.
    low_f_limit=min(A);
    high_f_limit=max(A);

    for i=1:length(finput) % eliminates the low frequencies
        if finput(i) >= low_f_limit
            flow_Index = i;
            break
        else
            flow_Index = 1;
        end
    end

    for j=1:length(finput)% eliminates the high frequencies
        if finput(j) >= high_f_limit
            fhigh_Index = j-1;
            break
        else
            fhigh_Index = j;
        end
    end

 %  Readjust arrays with the frequency limits

    zn=Rmeas+sqrt(-1)*Xmeas;
    Rmeas_lim=Rmeas(flow_Index:fhigh_Index);
    Xmeas_lim=Xmeas(flow_Index:fhigh_Index);
    zn=Rmeas_lim+sqrt(-1)*Xmeas_lim;
    Rdata=real(zn);
    Xdata=-imag(zn);                                                    (2)
    f=finput(flow_Index:fhigh_Index);  %new frequency array

else                                                                    (3)
    zn = Rmeas + sqrt(-1)*Xmeas;
    Rdata = real(zn);
    Xdata = -imag(zn);                                                  (4)

    f=finput;

end

%Estimate starting values
     startingVals = coleStartValues2(zn,f);

%calculate fittings (in 4 different planes)
     [coefZ ,Zest] = Circularfit(f,zn);

     [cfR,cfX,cfRX]=RXfit(f,Rdata,Xdata, startingVals, displayON,Zest);  (5)
```

Code 3.11 Matlab code of *Fitting_planes* function part 1.

```
%coeficients estimated
     coefR=[cfR.a, cfR.b, cfR.c, cfR.d];     %Coefficients of fitting in R-w plane
     coefX=[cfX.a, cfX.b, cfX.c, cfX.d];     %Coefficients of fitting in X-w plane
     coefRX=[cfRX.a, cfRX.b, cfRX.c, cfRX.d];%Coefficients of fitting in R+X-w plane

     coef=[coefR;coefX;coefRX;coefZ];
     coefM=mean(coef);

     zcoleR= coefR(2) + (coefR(1)-coefR(2))./(1+(sqrt(-1)*f*coefR(3)).^coefR(4));          (6)
     zcoleX= coefX(2) + (coefX(1)-coefX(2))./(1+(sqrt(-1)*f*coefX(3)).^coefX(4));
     zcoleRX= coefRX(2) + (coefRX(1)-coefRX(2))./(1+(sqrt(-1)*f*coefRX(3)).^coefRX(4));
     zcoleZ= coefZ(2) + (coefZ(1)-coefZ(2))./(1+(sqrt(-1)*f*coefZ(3)).^coefZ(4));

     fcR=1/coefR(3)/2/pi;
     fcX=1/coefX(3)/2/pi;
     fcRX=1/coefRX(3)/2/pi;
     fcZ=1/coefZ(3)/2/pi;

 if displayON==1

% Plot estimated cole functions
    figure;
    hold on;

    plot(Rdata, Xdata, '*') %cole with real data
    plot(real(zcoleR), -imag(zcoleR), 'r') %cole with parameter estimated from R
    plot(real(zcoleX), -imag(zcoleX), 'og') %cole with parameter estimated from X
    plot(real(zcoleRX), -imag(zcoleRX), '*y') %cole with parameter estimated from R+X
    plot(real(zcoleZ), -imag(zcoleZ), 'm') %cole with parameter estimated from R+X
    hold off
    legend('Real Data','Estimated Cole from R','Estimated Cole from X','Estimated
Cole from R+X', 'Estimated Cole from Z')
    title('Cole function estimation')
    xlabel('R')
    ylabel('-X')


end

DataArray.Coefs(1,:) = [coefR(1), coefR(2), coefR(3), coefR(4)];                       (7)
DataArray.Coefs(2,:) = [coefX(1), coefX(2), coefX(3), coefX(4)];                       (8)
DataArray.Coefs(3,:) = [coefRX(1), coefRX(2), coefRX(3), coefRX(4)];                   (9)
DataArray.Coefs(4,:) = [coefZ(1), coefZ(2), coefZ(3),coefZ(4)];                        (10)
DataArray.Coefs(5,:) = [coefM(1), coefM(2), coefM(3), coefM(4)];                       (11)

DataArray.Impedance = [zcoleR,zcoleX,zcoleRX,zcoleZ];                                  (12)

DataArray.CentralFreqs = [fcR,fcX,fcRX,fcZ];                                           (13)

DataArray.Data_in(:,1) = finput;                                                       (14)
DataArray.Data_in(:,2) = Rmeas;                                                        (15)
DataArray.Data_in(:,3) = Xmeas;                                                        (16)

end
```

Code 3.12 Matlab code of ***Fitting_planes*** function part 2.

3.2.6 Cole Rejection

The *Cole Rejection* function calculates the Cole fitting by using rejecting values as inputs. The values that differ less than the selected percentage, respect to the Cole function values are used to recalculate a new *Cole function* fitting. There are four possible percentages of rejection: 1, 2, 3, 4 and 5%. After each fitting, a dialog window appears giving the possibility of continue with a new fitting.

The function receives the resistance *Rmeas,* the reactance *Xmeas* and the desired rejection limit (Value) as input parameters. First of all, we have to execute the Cole fitting and hence these variables have to be passed as a parameter to *Fitting planes* function, line (1). The results of the Cole fitting are saved with the resistance in *RCole* and reactance *XCole*.

As it was aforementioned and as we can see in the *switch(case)* of the code, there are four rejection ratios possible, 1, 2, 3, 4 or 5%. If the value that we introduce is incorrect, a variable called *NothingHappens* turns to 1 and the function is not executed. Otherwise, for instance a value of 2 %, every value would be multiplied by 1.02, that is *Valuemax*, and by 0.98, the *Valuemin*. Thus the upper and lower limits are created.

When *FirstTime=1*, it means it is the first time that we make the comparison, the upper and lower limits are multiplied by *RCole* and *XCole* and then stored in arrays called *RValueMax,* corresponding to upper limit values, *RValueMin* corresponding to lower limit values and exactly the same with reactance values.

From line (2) to line (3) we find the core of the algorithm, that is a *while* loop that will not finish until we go through all the values of the array *Rvaluemax*. Inside of the *while* loop, there are *if-else* conditions. The first condition to enter is that resistance values are in the range established by the limits (*RValueMax* and *RValueMin*). While those conditions are satisfied the same process is done with the reactance in another *if-else* condition. Otherwise, the index *j=j+1* must be increased to process the next elements of resistance and reactance. Once the whole conditions are fulfilled, values are saved in two arrays called *Rrejection, Xrejection* and *FreqRejection.* Moreover indexes *j* and *w* are incremented to next value in the array.

When it arrives to the last value of the arrays, a question dialog window appears asking to the user if he/she wants to continue. If the user wants to continue, the variable *Continue* turns to 1 and the remaining values that accomplish with the limits (*RpreColefit, XpreColefit* and *freq)* are passed as inputs parameters to *Fitting_planes* function, otherwise the values are saved in the struct *DataArray,* lines (3) to (4).

The function makes it possible to continue with the results of *Fitting_planes* (*r2* and *i2*) and with *FirstTime=0* the *RValueMax* and *RValueMin values* are the result of multipling respectively *r2* by *ValueMax* and *ValueMin*. And it is exactly the same performing with the reactance (*i2*). Then we perform the algorithm again, doing the comparison with these new arrays and the original arrays that we had in the header of the function.

In conclusion, the rejection algorithm will be executed until the user decides to not continue (choosing No in the question dialog) or until there are no remaining values of the arrays of rejection (*Rrejection, Xrejection* and *FreqRejection*). The last statement could carry problems that are analyzed in the section Cole Start Values of the chapter 5.

```
function[DataArray,num]= ColeRejection(f,Rmeas,Xmeas,Value,namesource,loop)

IfRejec=1;
numrejections = 1;
num=1;
freq = f;

[DataArraycole]= Fitting_planes(f,Rmeas,Xmeas,0,IfRejec,0,0,namesource);    (1)

RCole=real(DataArraycole.Impedance(:,1)); %RCole=real(zcoleZ);
XCole=imag(DataArraycole.Impedance(:,1)); %XCole=imag(zcoleZ);
r2 = RCole;
i2 = XCole;

Length=length(Rmeas);
NothingHappens=0;

switch (Value)
    case '1'              %  Rejection 1%
        Valuemax=1.01;
        Valuemin=0.99;

    case '2'              %  Rejection 2%
        Valuemax=1.02;
        Valuemin=0.98;

    case '3'              %  Rejection 3%
        Valuemax=1.03;
        Valuemin=0.97;

    case '4'              %  Rejection 4%
        Valuemax=1.04;
        Valuemin=0.96;

    case '5'              %  Rejection 5%
        Valuemax=1.05;
        Valuemin=0.95;

    otherwise
        NothingHappens = 1;
end

Continue = 1;
FirstTime = 1;

if (NothingHappens == 0)
   while (Continue)
        if (FirstTime)

            RValueMax = RCole*Valuemax;
            RValueMin = RCole*Valuemin;
            XValueMax = XCole*Valuemax;
            XValueMin = XCole*Valuemin;
            ROriginal = Rmeas;
            XOriginal = Xmeas;
            fOriginal = f;
            DataArray.Rrejection = Rmeas;
            DataArray.Xrejection = Xmeas;
        else

            RValueMax = r2*Valuemax;
            RValueMin = r2*Valuemin;
```

Cod 3.13 Matlab code of *Cole Rejection* function part 1

```
            ROriginal = Rmeas;
            XOriginal = Xmeas;
            XValueMax = i2*Valuemax;
            XValueMin = i2*Valuemin;


            fOriginal=f;
            Length = length (ROriginal);
            DataArray.Rrejection = Rrejection;
            DataArray.Xrejection = Xrejection;

         end

         w = 1;
         j = 1;
         while ((j<=length(RValueMax)))                                          (2)
            if((RValueMax(j)>ROriginal(j))&&(RValueMin(j)<ROriginal(j)))
                if((XValueMax(j)<XOriginal(j))&&(XValueMin(j)>XOriginal(j)))
                     Rrejection(w)=ROriginal(j);
                     Xrejection(w)=XOriginal(j);
                     FreqRejection(w)= fOriginal(j);
                     w=w+1;
                     j=j+1;
                else                    j=j+1;
                end
            else
            j = j+1;            end
         end                                                                     (3)

         % We have to remove the cell values of R,X and Freq rejection that has 0 value.

         % Update arrays

              choice = questdlg('There are remains of Impedance values. Do you want to
continue with the rejection?','Continue with Rejection?,'Yes','No','Yes');
              if(strcmp(choice,'Yes'))
                  RpreColefit=Rrejection;
                  XpreColefit=Xrejection;
                  freq = FreqRejection;
          [DataArraycole]=
Fitting_planes(freq,RpreColefit,XpreColefit,0,IfRejec,0,0,namesource);
                  num = num + 1;
                  FirstTime = 0;
                  r2=real(DataArraycole(1).Impedance(:,1)); % Resistance
                  i2=imag(DataArraycole(1).Impedance(:,1)); % Reactance
              else
                  Continue = 0;
              end
         end
    end

    Zrejection = Rrejection + sqrt(-1)*Xrejection;

end

DataArray.freq = freq;                                                           (4)
DataArray.r2 = r2;
DataArray.i2 = i2;
DataArray.num = num;
DataArray.Zrejection=Zrejection;
DataArray.DataArraycole = DataArraycole;

DataArray.RValueMax = RValueMax;
DataArray.RvalueMin = RValueMin;
DataArray.XValueMax = XValueMax;
DataArray.XvalueMin = XValueMin;                                                 (5)

end
```

Code 3.14 Matlab code of *ColeRejection* function part 2

3.3 Special Functions

The aim of this section is to present several functions implemented to perform a batch analysis of the EBI measurements from different subjects with the goal to select only the subjects which present a Gaussian distribution in the distribution of their characteristic frequency. Such analysis consists in two phases: the first one is performed by the function *Final* and give as a result if the characteristic frequencies have a Gaussian behavior or not. However, the measurements have to be passed with a previous correction made by the *Fcorrection* function.

The second phase performed by function *Final_B*, only selects the subjects from the previous phase that presents a Gaussian distribution in the characteristic frequencies and perform the same filter with the remaining subjects. The general diagram is shown in the following Fig.3.8.

Although the analysis is possible in each fitting plane, the *R(ω)* plane is the chosen due to it being the best fitting in relation to the characteristic frequencies, the causes of this performance are explained in section *5.1.1 Estimated Cole from R(ω)of* chapter 5.

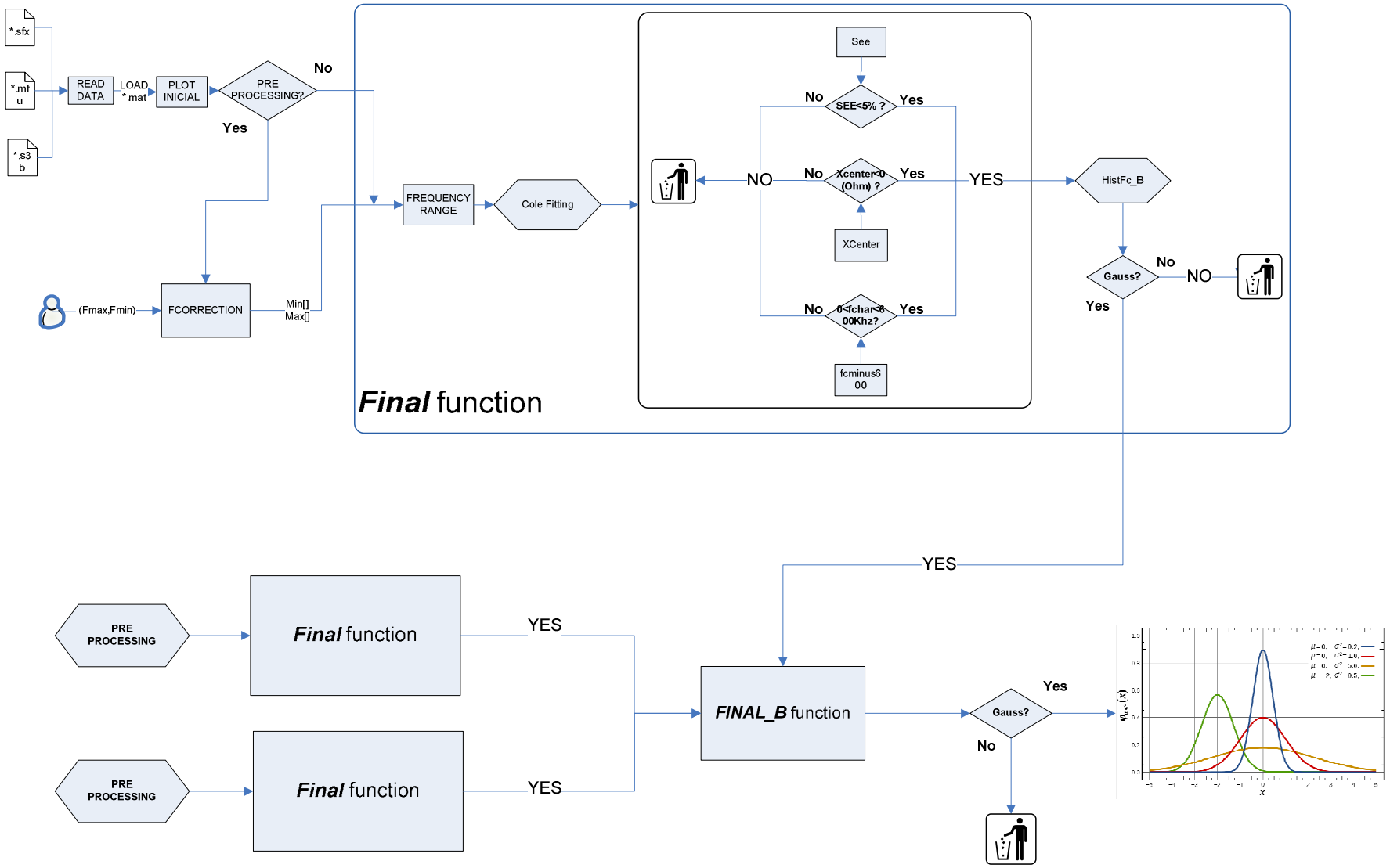

Fig 3.8 Flux diagram of the whole analysis performed by Special Functions

3.3.1 Final

As was mentioned before, the *Final* function is fed by the corrections in the measurements made previously by the user, *DataArray(:,:,numfiles*), with the *Fcorrection* function. The function also permits to specify the first (index *first*) and the last (index *last*) number of the file of the struct *DataArray* that the user wants to process. Moreover, it is possible to know the name of the file, line (3) in Code 3.16, which is processed in the analysis. This last task it is possible to perform, passing the array of names *namesource* as a parameter in the header. Ending with the header just mentioned, that the user can also choose the name in which he/she wants to save the file (*.mat) along with the number of the file that is contained in the structure, line (9).

There are important parameters that may influence the result of the analysis of a complex EBI spectrum. Within these parameters the frequency range, considered when making the Cole Fitting is very important thus the resulting characteristic frequencies, line (2) in Code 3.15, are frequencialy distributed and a possible Gaussian spectrum in the outcome of the analysis can be depicted. In this design of the function, it was decided to select several frequency ranges. The selected frequency ranges have produced a total of 110 different frequency ranges and they are listed in the following table:

Min (Khz)	Max (Khz)
3	450
6	465
9	480
12	495
15	510
18	525
21	540
24	555
27	570
30	585
	600

Table 3.2 Frequency ranges for doing each fitting

The initial idea was that for each range, the measurements were analyzed for a rejection limit of 0% and 5%. In the case of a Rejection Limit set to 0%, no measurement points will be discarded from the analysis data set and this is exactly the Rejection Limit that was chosen, hence, is the only option that allow the feasibility in most of the performance of the function *Fitting_planes*, this aspect is further explained in section *5.1.2 Cole Start Values* of chapter 5.

In addition, it was also the idea of using 20 values of *Td*, from 0 to 20 ns, but the number of Cole Fittings is increased from the 110 iterations that there are with the actual performance to 2200 (110*20) iterations. This high number of iterations causes the necessity of a computer cluster to provide results of the analysis in an acceptable time.

Once the loops for the frequency range are defined with the values *min* and *max*, these values along with the arrays of frequency, resistance and reactance are passed as parameters to *Fitting_planes_B* function. The results of the fitting are collected in the *DataCol(num,min,max)* struct, line (1) in Code 3.15. Finally, the Cole parameters of the fitting R_0, R_∞, τ ,α and other interesting values to save as well as *zcoleR* and the original values of the measurements (*fmeas*, *Rmeas* and *Xmeas)* are saved in the struct *data*.

Prior to find the characteristic frequencies histogram, the fitting results are applied to three filters, lines (4), (5) and (6) to remove possible damaged data introduced by measurement artifacts. The applied basic filters are the following:

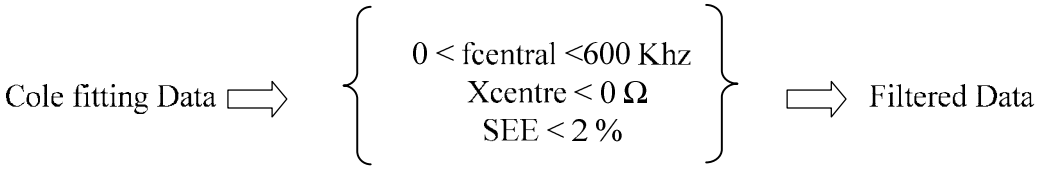

Fig 3.9 Filtering applied

When the results from the filters are obtained (*OkS*, *OkX* and *Okfc*) a loop that cover all the positions of these last arrays make a comparison of each position. If the three values are equal "1" in the same position, it means that the measurements are correct and hence their respective central frequencies, *fcRout*, are used to calculate the histogram with the function *Histfc_B*, line (7).

The results provided by *Histfc_B*, *ArrayR*, are used to find out if remaining central frequencies have a Gaussian distribution. Such comparison is made by *CompareGraph* function that provides the result variables as well as: *GaussOK*, *centroid*, *centroidMax*, *centroidMin* and *freq,* line (8). Such result variables are saved along with the struct data, the arrays: *seeR*, *Xc* and *fcminus*. Finally, variable *file* is the name the user wants to save the file, line (10).

```
function[GaussOK]=Final(DataArray,namesource,first,last,namefile)

 for num=first:last
 i=1;
        for min=3:3:30
               for max=450:15:600

                 [DataCol1(num,min,max)] =  Fitting_planes_B(DataArray(:,1,num),    (1)
                 DataArray(:,2,num),DataArray(:,3,num),0,0,1,[min,max]);
                 FCR(i)=DataCol1(num,min,max).fcR;                                  (2)
                 zcoleR(i).z=DataCol1(num,min,max).zcoleR;
                 R0(i)=DataCol1(num,min,max).R0;
                 Ri(i)=DataCol1(num,min,max).Ri;
                 tau(i)=DataCol1(num,min,max).tau;
                 alpha(i)=DataCol1(num,min,max).alpha;
                 Rmeas(i).r=DataCol1(num,min,max).Rdata;
                 Xmeas(i).x=DataCol1(num,min,max).Xdata;
                 fmeas(i).f=DataCol1(num,min,max).fdata;
                 i=i+1;
               end
        end
      %%%%%%%%%%%%%% values for each measurement %%%%%%%%%%%%%%%%%%%%%%%%%%%%

      name=namesource(num);                                                         (3)
      data.fcR=FCR;
      data.zcoleR=zcoleR;
      data.R0=R0;
      data.Ri=Ri;
      data.tau=tau;
      data.alpha=alpha;
      data.name=name;
      data.Rmeas=Rmeas;
      data.Xmeas=Xmeas;
      data.fmeas=fmeas;

      %%%%%%%%%%%%%%%% Filters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

      [OkS,seeR]= See_B(data);                                                      (4)
      [OkX,Xc]= Xcenter_B(data);                                                    (5)
      [Okfc,fcminus]= fcminus600_B(data);                                           (6)

      tmpf=zeros(1,length(data.Rmeas(1,:)));

      j=1;

      for i=1:length(data.Rmeas(1,:))

          if(OkS(i)== OkX(i)==Okfc(i)==1)
              tmpf(j)=data.fcR(i);
              j=j+1;
          end

      end

     j=1;
     i=1;

      for k=1:length(tmpf)

        if(tmpf(j)~=0)
          fcRout(i)=tmpf(j);
          i=i+1;
          j=j+1;
        else
          j=j+1;
        end

      end
```

Code 3.15 Matlab code of *Final* function part 1

```
    %%%%%%%%%%%%%%% Histogram %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    ArrayR=HistFc_B(fcRout,num);                                          (7)

    %%%%%%%%%%%%%%% Compare with a Gaussian %%%%%%%%%%%%%%%%%%%%%%%%%%%%

    [GaussOK,centroid,centroidMax,centroidMin,freq]=
    CompareGraph(ArrayR(1,:),ArrayR(2,:));                                (8)

    %%%%%%%%%%%%%%% Save each file %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    file = sprintf('%s_%d.mat',namefile,num);                             (9)
    save(file,'data','fcRout','ArrayR','GaussOK','centroid','centroidMax',  (10)
    'centroidMin','freq','file','seeR','Xc','fcminus');

 end
end
```

Code 3.16 Matlab code of *Final* function part 2

All the functions mentioned before, are explained in the following paragraphs:

3.3.1.1 Xcentre_B

This function discards the reactive center of semicircle, *Xc,* that are positive. Biological tissue fitted to a Cole model cannot exhibit a positive value for *Xc* in a Cole plot. The design consists in a loop that allows going over all the positions of the variables (*R0* ,*Ri* and *alpha*) that are involved in the algorithm to find *Xc*. If the *Xc* value satisfies the condition, the array *OkR* turns to "1" otherwise turns to "0".

According to the following representation of the depressed Cole-Cole plot semicircle, it is possible to find the reactive centre value from the drawn variables.

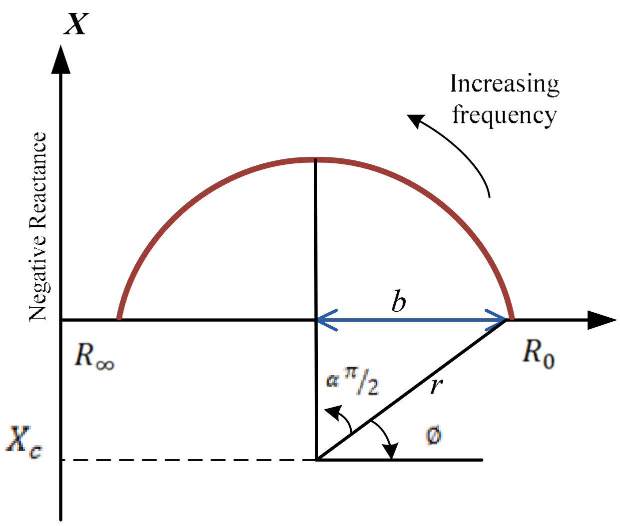

Fig 3.10 Depressed Cole-Cole plot semicircle

$$b = \frac{R_0 - R_\infty}{2} \quad \textbf{(3.15)}$$

$$+\alpha\frac{\pi}{2}=\frac{\pi}{2}\rightarrow\quad=\frac{\pi}{2}(1-\alpha) \tag{3.16}$$

$$\left.\begin{matrix} rcos=b \\ rsen=-X_c \end{matrix}\right\}\rightarrow\begin{matrix} r=\frac{b}{cos} \\ r=\frac{X_c}{sen} \end{matrix}\rightarrow\ X_c=-b\frac{sen}{cos}=-btan \tag{3.17}$$

```
function[OkR,Xc]= Xcenter_B(data)

 Rmeas=data.Rmeas;
 R0=data.R0;
 Ri=data.Ri;
 alpha=data.alpha;

for j=1:length(Rmeas(1,:))

        b(j)=(R0(j)-Ri(j))/2;
        Phi(j)=(pi/2)*(1-alpha(j));
        Xc(j)=-b(j)*tan(Phi(j));



        if((Xc(j)<0))

           OkR(j)=1;

        else
           OkR(j)=0;

        end

 end

end
```

Code 3.17 Matlab code of *Xcenter_B* function

3.3.1.2 See_B

This function calculates the Standard Error of Estimate of the Cole impedance, in percentage representation. Consequently, only fitting with relative low percentage, 2%, of SEE are not discarded.

The SEE is calculated with the following equations, where R is the real part of the original impedance Zmeas and $\bar{R}$ is the real part of ZCole that is the impedance obtained with the fitting method in R(ω).

$$SEE_M=\sqrt{\frac{\sum(M-\bar{M})^2}{N}} \tag{3.18}$$

Where N is the number of estimations and M is the magnitude under study, and this particular case the resistance from R(ω) plane.

$$SEE_R=\sqrt{\frac{\sum(R-\bar{R})^2}{N}} \tag{3.19}$$

```
Function[OkR,seeR]= See_B(data)

      Rmeas=data.Rmeas;
      Xmeas=data.Xmeas;
      fmeas=data.fmeas;
      zcoleR=data.zcoleR;

for j=1:length(Rmeas(1,:))

  zmeas=Rmeas(1,j).r(:,1)+sqrt(-1)*Xmeas(1,j).x(:,1);
  zn(j).zm=zmeas;

  zplane=zcoleR(1,j).z(:,1);
  zR(j).zr=zplane;

%%%%%% SEE in R plane

seeRres(j)=sqrt(sum((real(zn(j).zm(:,1))-
real(zR(j).zr)).^2./(real(zn(j).zm(:,1))).^2)/length(zn(j).zm(:,1)));
seeR(j)=seeRres(j);

  if(seeR(j) < 0.02)      %x<2%

       OkR(j)=1;

   else
       OkR(j)=0;

  end

 end

end
```

Code 3.18 Matlab code of *See B* function

3.3.1.3 Fcminus600_B

The last function of the filters discards the fittings with characteristic frequencies, which are bigger than 600 KHz or negative. The design consists in a loop that allows going over all the positions of the array of frequencies and compares each *fc* with 0 and 600 KHz. If values satisfy the condition, the array *OkR* turns to "1" otherwise turns to "0". The code ins shown in Code box 3.19

```
function[OkR,FCR]= fcminus600_B(data)

 Rmeas=data.Rmeas;
 FCR=data.fcR;

 for j=1:length(Rmeas(1,:))


 if((FCR(j)>0)&&(FCR(j)<600))
 OkR(j)=1;
 else
 OkR(j)=0;
 end
```

Code 3.19 Matlab code of *fcminus600_B* function

3.3.1.4 HistFc_B

The following function calculates an histogram of an array of frequencies, *FCR*, passed as parameter and provides a result which is saved in an Array of two dimensions, *ArrayR=[nR,fR]*, where the first row, *nR,* is a vector with the probability (%) of being in a specific frequency and the second row, *fR*, is the vector of frequencies in which they are represented in the last probabilities.

```
function[ArrayR]=HistFc_B(FCR,num)


    [nR,fR]=hist(FCR,length(FCR));

    ArrayR=[nR;fR];

end
```

Code 3.20 Matlab code of *HistFc_B* function

3.3.1.5 CompareGraph

This function compares the histogram graph with a Gaussian distribution. The way to do this is based in an algorithm that calculates the centroid of frequencies to find where the probability is concentrated, vector of probabilities *Array(1,:)* in line (1), along all the frequencies of the of the vector *Array(2,:)*, line (2).

$$centroid = \frac{\sum_{i=1}^{N} |nR|^2 fR}{\sum_{i=1}^{N} |nR|^2} \quad \textbf{(3.20)}$$

Where *nR* is the vector of probabilities *Array(1,:)* and *fR* is the vector of frequencies *Array(2,:).*

Once the centroid is calculated, the function finds the five maximum peaks of the vector of probabilities and their respective frequencies, line (3) and (4). Moreover, if at least three of the maximum frequencies, *ok>=3*, are contained inside the interval formed by *centroidMin* and *centroidMax* bounds, the variable *GaussOk* turns "1" and hence a histogram with Gaussian distribution is considered. In fact, the last bounds represent the value the centroid plus and minus 15% respectively, lines (5) to (7) in Code 3.22.

Finishing with this function, only remark that the result variables such as *GaussOk, centroid*, *centroidMax*, *centroidMin*, the array with five maximum frequencies, *freq*, and *ok* are return as parameters by the function.

```
function[GaussOK,centroid,centroidMax,centroidMin,freq,ok]= CompareGraph(mag1,mag2)

Array(1,:)=mag1;                                                    (1)
Array(2,:)=mag2;                                                    (2)


%%% Calculation of the centroid %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    for i=1:length(Array(1,:))

        Absmag(i)=abs(Array(1,i));

    end
        if(sum(Absmag)~=0)

        A=sum((Absmag.^2).*Array(2,:));
        B=sum(Absmag.^2);

        centroid=A/B;

        else
        centroid=0;
        end

if(centroid~=0)

%%%%%% Finding the 5 maximum peaks %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Array2(1,:)=mag1;
Array2(2,:)=mag2;
for m=1:5

   [peak(m),pos]=max(Array2(1,:));                                  (3)

   freq(m)=Array2(2,pos);                                           (4)


%%%Delete the last value and his frequency of the Array of values%%%

   i=1;
   j=1;
   tmp=zeros(1,length(Array2));
   tmpf=zeros(1,length(Array2));

   for k=1:length(Array2(1,:))

       if(Array2(2,j)~=freq(m))

        tmp(i)=Array2(1,j);
        tmpf(i)=Array2(2,j);
        i=i+1;
        j=j+1;

       else
        j=j+1;
       end
   end


  Array2(1,:)=tmp;
  Array2(2,:)=tmpf;

end
```

Code 3.21 Matlab code of *CompareGraph* function part1

```
%%%we have to compare the values of the frequency of the peaks with the
% value of the centroid %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ok=0;

Percentage=centroid*0.15;                                                    (5)
centroidMax=centroid+Percentage;                                             (6)
centroidMin=centroid-Percentage;                                             (7)

  for i=1:5

    if((centroidMax>freq(i))&&(freq(i)>centroidMin))

        ok=ok+1;

    end

  end

  if(ok>=3)
    GaussOK=1;
  else
    GaussOK=0;
  end

end

end
```

Code 3.22 Matlab code of *CompareGraph* function part2

3.3.2 Final_B

The last function to perform the analysis is fed with the subjects files (*.mat) coming from *Final* function, the task for retrieving such files is done with *uigetfile*, line (1), which opens a standard dialog box to enable the user to select the file to treat.

The useful variables of each file are: *fcRout*, *GaussOk* and data; that are extracted from the files with *load* function, line (2), and deposited in the struct *d*. Using a loop to cover all the files, the variable *GaOk* is checked, line (3), and if the Gaussian distribution is satisfactory, the algorithm saves the name, *nameOK* in line (4) and the array of all the valid frequencies, *FCR* in line (5), of the correct file.

Once the central frequencies array, *FCR*, of all the files is completely filled, the histogram is calculated passing that variable as a parameter to *Histfc_B*, line (6). The utility of the return variable, *ArrayR*, and subsequent comparison with *CompareGraph* follows the same performance and returns the same variables as was mentioned in the previous subsection, line (7). Therefore, some returned variables such as *GaussOk*, *centroid*, *centroidMin* and *centroidMax* are displayed in the histogram plot, thanks to the function *text*, lines (8) to (11).

Finally, the user can open a standard dialog box for saving the involved workspace variables with function *uisave*, line (12).

```
function[]=Final_B()

 [x, PATHNAME]=uigetfile('C:\Users\samsung\Documents\MATLAB\My project\*.mat',          (1)
 file','MultiSelect', 'on');

 y = cellstr(x);
 cd(PATHNAME); %put PATHNAME as current directory

 numfiles = size(y,2);
 namesource = x;

  for i=1:numfiles
    namesource(i) = y(1,i);
    temp = char(y(1,i));
    d= load (temp,'fcRout','GaussOK','data');                                        (2)
    fcRout(1,:,i)=d.fcRout;
    GaOk(i)=d.GaussOK;
    names(i)=d.data.name;
  end

k=1;m=1;

 for i=1:numfiles

    if(GaOk(i)==1)                                                                   (3)

        nameOK(m)=names(i);                                                          (4)
        m=m+1;
        for j=1:length(fcRout(1,:,i)
            FCR(k)= fcRout(1,j,i); % j number of colums, i number of file            (5)
            k=k+1;
        end
```

Code 3.23 Matlab code of *Final_B* function part 1

```
[GaussOK,centroid,centroidMax,centroidMin,freq,ok]=                                    (7)
CompareGraph(ArrayR(1,:),ArrayR(2,:));

nametemp = char(names(1));

figure ('name', nametemp);

   hist(FCR,length(FCR));    title('Histogram plot of fcR (Khz) ');

   xlabel('Khz')
   ylabel('Number of frequencies')

   text(GaussOK,GaussOK,...                                                            (8)
       ['GaussOK = ',num2str(GaussOK)],'Position',[123 7.708 17.32],...
       'HorizontalAlignment','center',...
       'BackgroundColor',[1 1 0]);

   text(centroidMax,centroidMax,                                                       (9)
       ['centroidMax = ',num2str(centroidMax)],'Position',[118.3 7.146 17.32],...
       'HorizontalAlignment','center',...
       'BackgroundColor',[1 1 0]);

   text(centroidMin,centroidMin,                                                       (10)
       ['centroidMin = ',num2str(centroidMin)],'Position',[119.4 6.023 17.32],...
       'HorizontalAlignment','center',...
       'BackgroundColor',[1 1 0]);

   text(centroid,centroid,                                                             (11)
       ['centroid = ',num2str(centroid)],'Position',[120.9 6.608 17.32],...
       'HorizontalAlignment','center',...
       'BackgroundColor',[1 1 0]);
uisave();                                                                              (12)
end
```

Code 3.24 Matlab code of *Final_B* function part 2

CHAPTER 4.

RESULTS

4.1. General Overview

The contents of this chapter show the results obtained from testing the different software tools implemented in this thesis work and also in "*Development of a Software Application Suite for Electrical Bioimpedance Data Analysis*", with EBI data obtained from transcephalic measurements on adults and newborns. When applied successfully, it is possible obtain proper results from frequency analysis or Cole plot representation and its relative values like the characteristic frequencies,f_c, τ or α. All these results are compared with the results provided by BioImp, the software tool used with the impedance spectrometer SFB7 manufactured by Impedimed.

4.2. EBI Data Analysis

The input data used for all the following examples are source files with the same format as described in preceding chapters. The files containing the EBI data are any of the following formats: *.mfu, *.s3b or *.sfx.

The performed analysis consists in 5 phases:

1. Corrections in the measurements.
2. Cole fitting of EBI measurements in different intervals of each subject.
3. Filtering of the Cole fitting results and histogram.
4. Filtering of the histogram results.

4.2.1. Adults

In this section, 20 files from healthy adults have been analyzed with the purpose of obtaining reference values for the spectrum of complex EBI.

4.2.1.1 Corrections in the measurements

The different results presented below have a deviation suffered mostly by the reactance and the phase and it is easy to notice at low and high frequencies. This deviation is caused mostly by a capacitive leakage and creates a hook-alike deviation in the data, which usually strongly affects measurements at frequencies above 500 kHz.

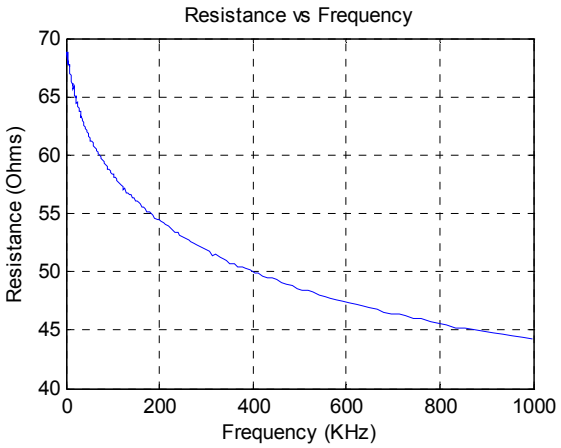

Fig 4.1. *Resistance VS Frequency*

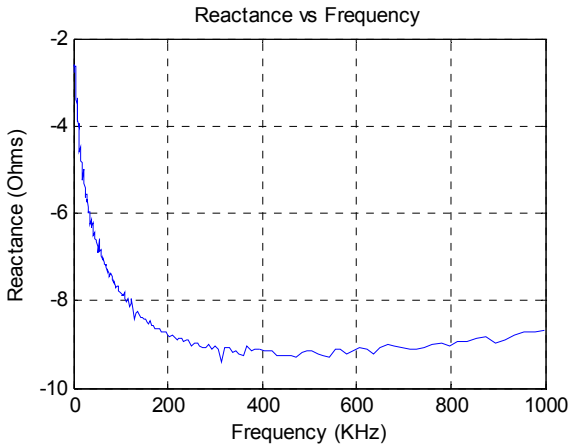

Fig 4.2. *Reactance VS* Frequency

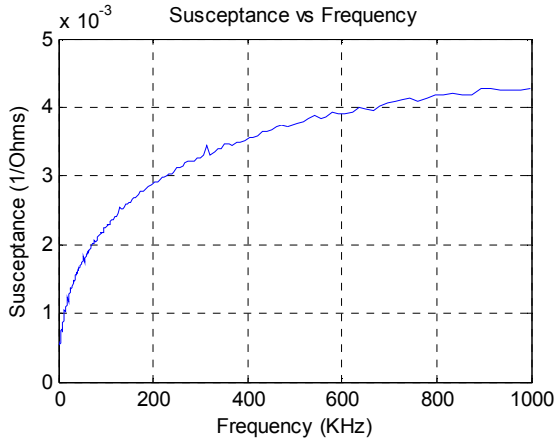

Fig 4.3 *Susceptance VS Frequency*

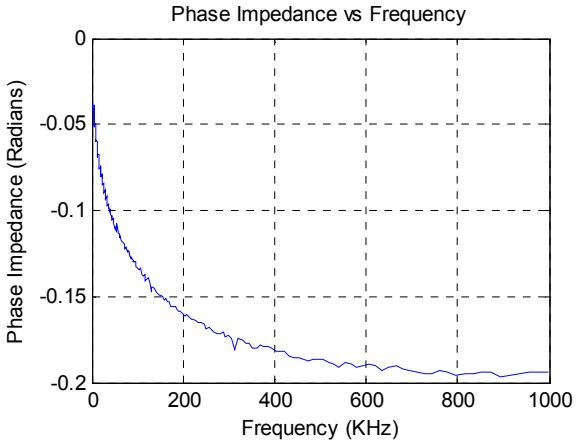

Fig 4.4 *Impedance Phase VS Frequency*

Despite the observed hook-alike deviation present in the reactance measurements at high frequencies in Fig.4.2, and the Phase in Fig.4.4 it is possible to observe that the curve of the Resistance in Fig. 4.1, have a normal behavior due to the curve approaches at a specific value. If we observe the figure of the susceptance in Fig.3.3, it is also possible to notice the deviation caused by the capacitive leakage, like in the Reactance, which is starting from approximately 300 KHz.

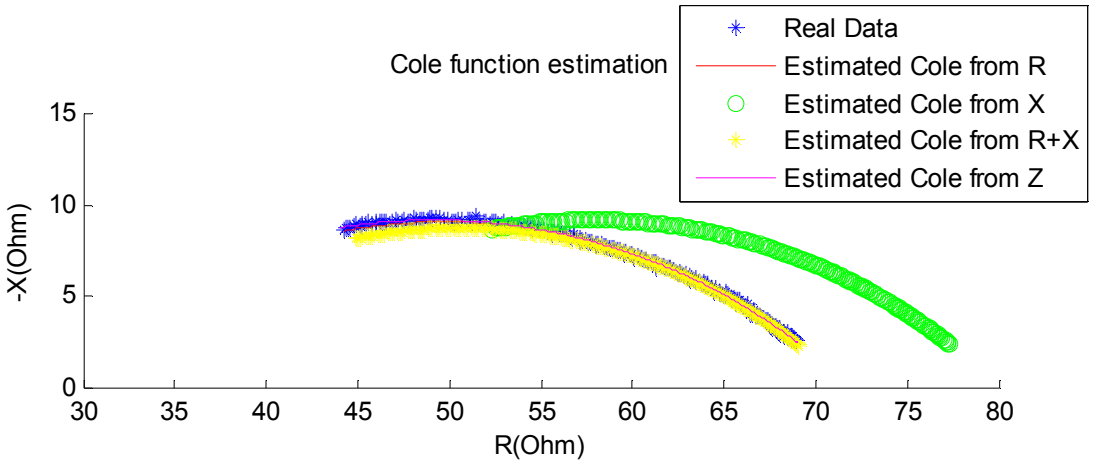

Fig 4.5 Cole Plot fitted in the four different planes**.**

The Figure above, Fig 4.5, shows the Cole plot representation without any correction applied. The blue line corresponds to the experimental Data and the other colored lines correspond to the fitted curves each of them obtained with a different fitting method or approach. The best fitting, according to the original data, are the Estimated Cole from the resistance spectra and the Estimated Cole from the impedance plane, but the performance for this last fitting is not always good and the reason of that is explained in chapter xx in section of discussion.

Therefore, in the whole analysis, the R(ω) fitting is used to estimate the different characteristic coefficients. To compare the obtained results of each file, the best way to check if the application is working properly or not is with an analysis with BioImp.

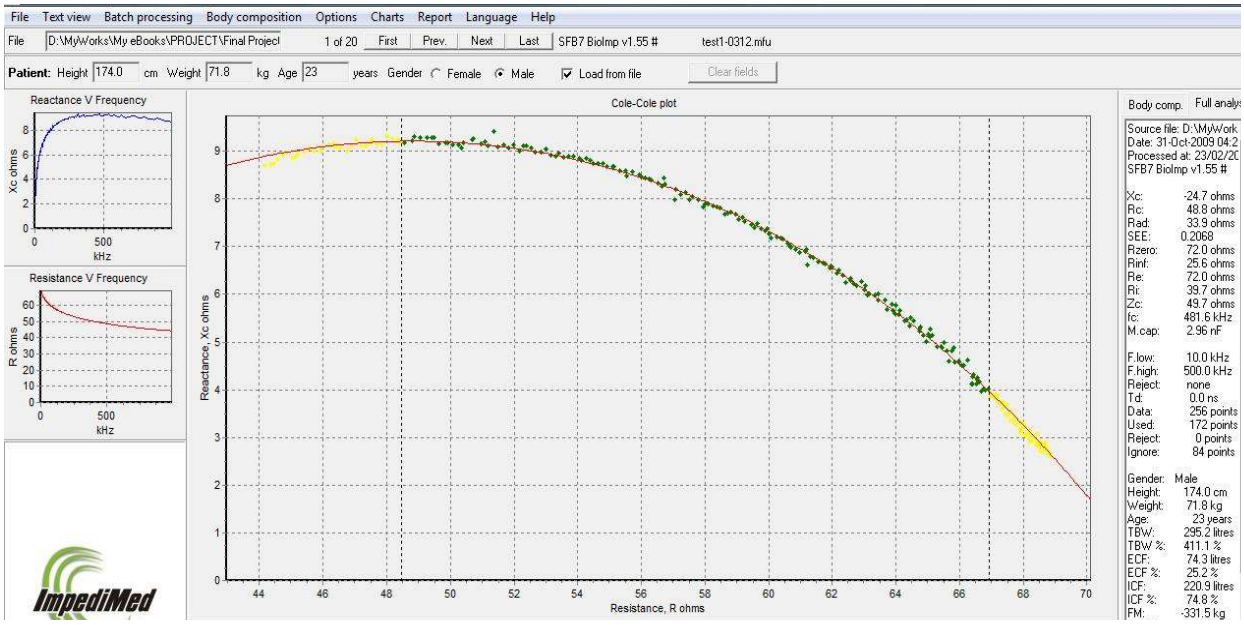

Fig 4.6. Bioimp representation of the file test1-0312.mfu

In Fig 4.6, a Cole plot fitted by BioImp is shown without any correction. On the left side, resistance and reactance against frequency are also displayed. If the plots are compared with figures 4.1 and 4.2, it is possible to notice that the values of both the spectra are correct and the values are similar.

After detecting the presence of capacitive leakage artifacts, the next step is to proceed to correct the Hook effect present in the EBI data. In this way, a correction between 3 - 1000 KHz is made using the *FCorrection* function presented in Chapter 3.

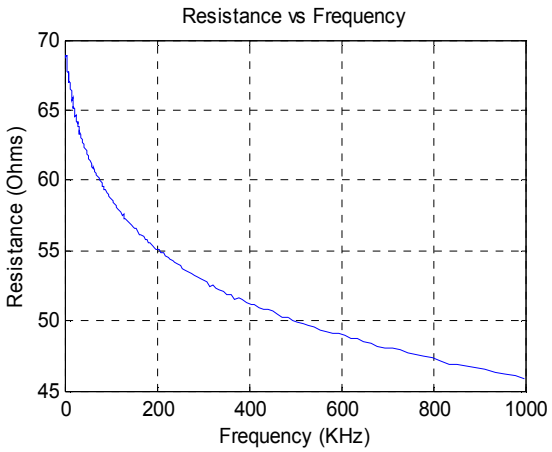

Fig 4.7 Resistance vs Frequency

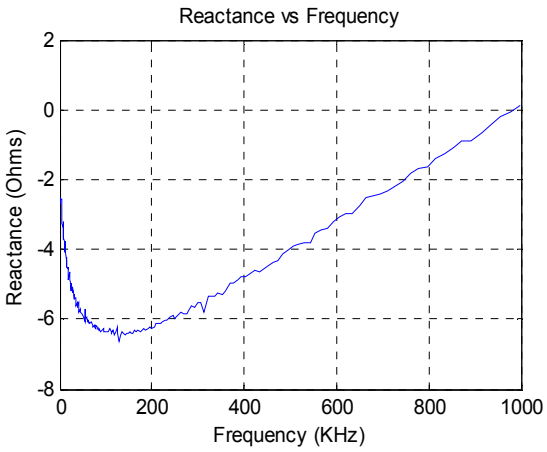

Fig 4.8 Reactance vs Frequency

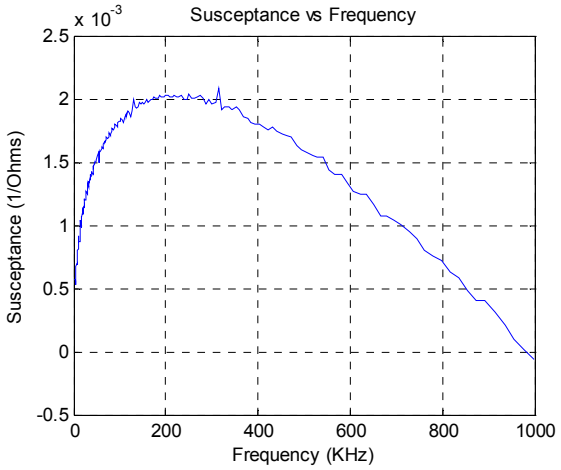

Fig 4.9 Susceptance vs Frequency

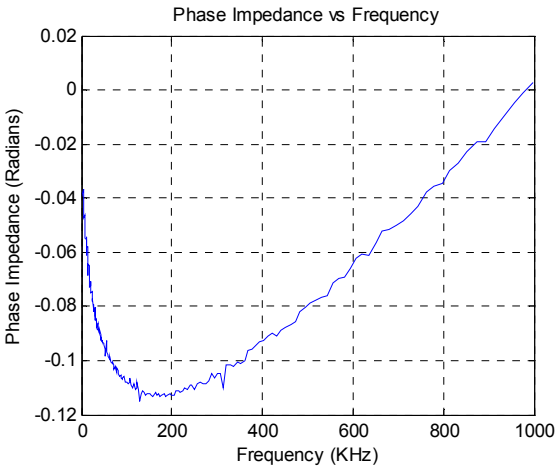

Fig 4.10 Impedance Phase vs Frequency

Looking at the Figures above, that where EBI data is plotted from 3 - 1000 kHz and comparing it with the previous figures, Fig.4.1-4.4, it is possible to observe significant differences in the plot of the reactance, phase and susceptance, which now tend towards zero for increasing frequencies. This is the expected behavior from EBI data. The resistance remains with the same aspect.

The figure, Fig.4.11, shows the Original data in blue and the Correction Data in black. As it was presented before, the best fitting for the spectrum of the Cole Plot in the impedance plane, is performed with Zplane.

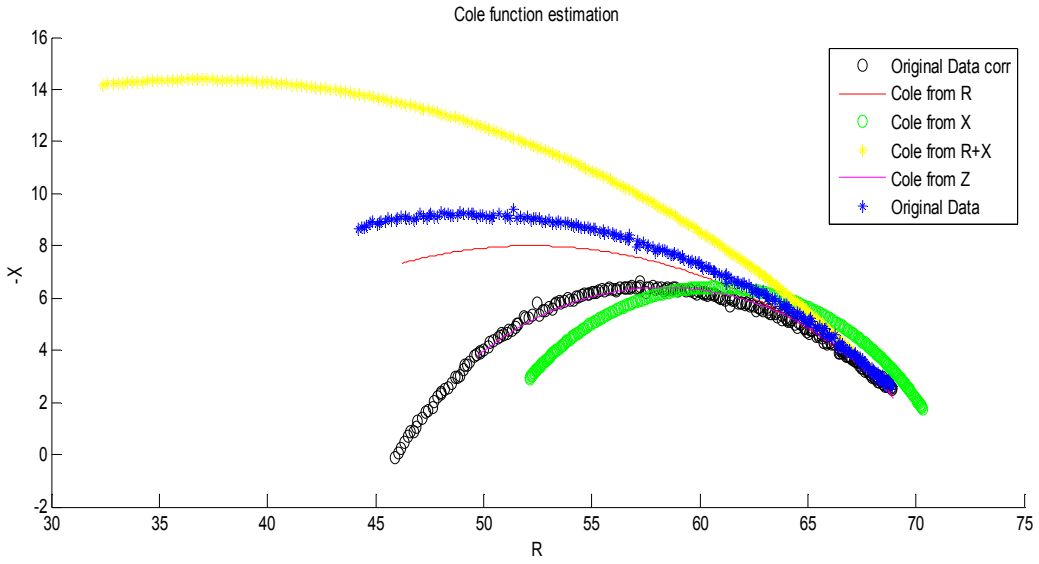

Fig 4.11 Cole Estimation in the four planes with corrections

Fig 4.12 represents the file with corrections. By comparing it with Figures 4.7 and 4.8, it is possible to observe that the maximum and minimum values of the resistance are near 69 Ω and 45 Ω respectively and are approximately the same with the representation of Bioimp and the obtained with the Toolbox. If you also compare the reactance, it is possible to see that the maximum and minimum values of the reactance in Bioimp and the Toolbox are approximately 6.3 Ω and 0 Ω respectively.

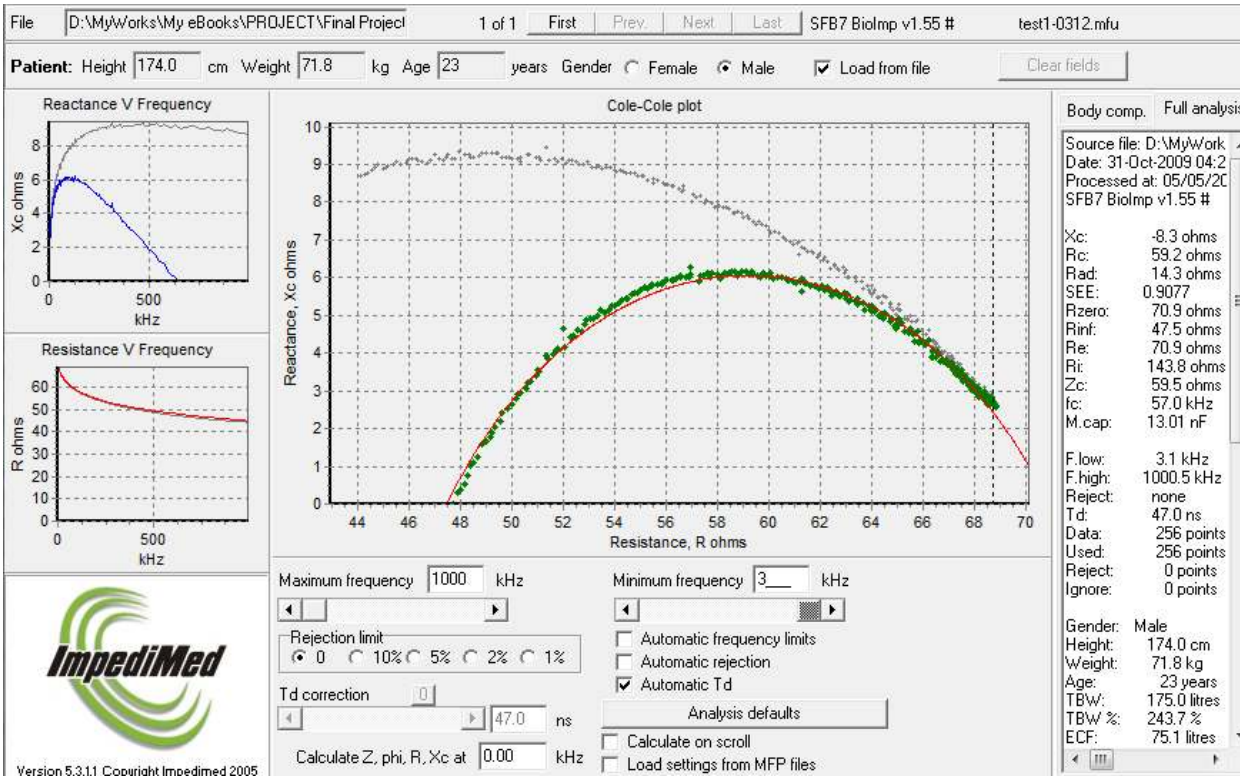

Fig 4.12 Bioimp representation of the file test1-0312.mfu with the correction.

The following tables show other important coefficients that we can find using the function *Fitting Planes*. As it is commented before, the whole analysis will be done with R(ω) plane. In Fig. 4.12 there are different values such as: R_0=70.9 Ω, R_∞=47.5 Ω and *fc*=57 Khz that compare with the results of the Table 4.1 and Table 4.2 are in the same order of magnitude: R_0=71.5309 Ω, R_∞=32.8842 Ω. Besides, in Table 4.2 the only near magnitudes for *fc*=57 Khz are: *fcR*=52.9730 Khz in the plane R(ω) and *fcRX*=102.9301 in R(ω)+X(ω), but this last result is discarded by the results of previous result in the same plane.

	R0	**R∞**	**τ**	**α**
R(ω)	71.5309 Ω	32.8842 Ω	0.0030 s	0.5000
X(ω)	71.4729 Ω	49.9126 Ω	0.0116 s	0.6839
R(ω)+X(ω)	71.5262 Ω	2.0862 Ω	0.0015 s	0.5000
Z plane	70.8322 Ω	45.8304 Ω	0.0098 s	0.6029

Table 4.1 Coeficients in each plane

	fcR	**fcX**	**fcRX**	**fcZ**
Value (Khz)	52.9730	13.7643	102.9301	16.1606

Table 4.2 Central Frequencies in each plane

4.2.1.2 Cole fitting of EBI measurements in different intervals of each subject.

The next step in the analysis is the application of the function *Final* and *Final_B*. The design and the way that these functions work are explained in chapter 3 in special functions section.

Once the original files from the Coronal brain measurements are corrected, those 20 files are introduced in the function *Final*, which takes each file and performs a Cole fitting in the R(ω) plane in a 110 intervals of frequencies.

For each range, the measurements would be analyzed for a rejection limit of 0% and 5%, but the performance of the *Cole Rejection* function is not suitable for a narrow range of values. This problem in the function is duly explained in chapter 5 in section Cole Start Values.

4.2.1.3 Filtering of the Cole fitting results and histogram.

Prior to execute the histogram with the characteristic frequencies the fitting results have to be passed through three different filters to remove possible damaged data affected by measurement artifacts.

The applied basic filters are the following, as to be presented in the chapter 3 in section special functions:

$0 < fcharacterisitc < 600$ Khz

Xcentre < 0 Ω

SEE < 2%

The first filter is performed with the function *fcminus600* and as expected all values satisfy the condition. The values of *fcR* have maximum values of 200 KHz.

The second filter provides results of Xcentre, which are between [-18.6138 Ω, - 16.2682 Ω], thus these values are completely correct.

The last filter consists in removing values of Standard Error of Estimate less than 2%. The values are in the range of [0.1%, 1%].

Therefore, all the coefficients of the fitting are part of the useful data to perform the histograms. The fittings done in the different files of one subject, produce several different distributions of characteristic frequency values taken in the R(ω) plane.

The following figures .4.13-4.16 contain histograms with different types of distributions. In most of them, the value of *GaussOK*, presented in the chapter 3 in special functions section is equal one, therefore indicates that the distribution of all the different *fcR* has a Gaussian behavior. The subjects with not Gaussian distributions are removed from further analysis.

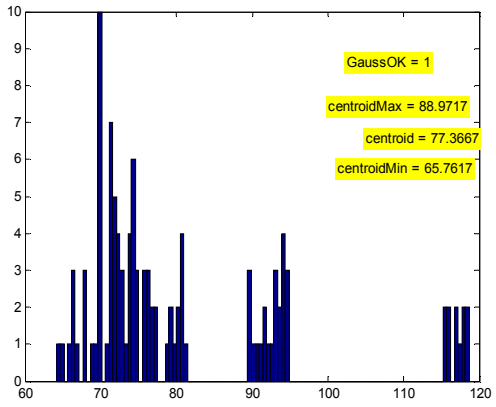

Fig 4.13 Correct Gaussian distribution

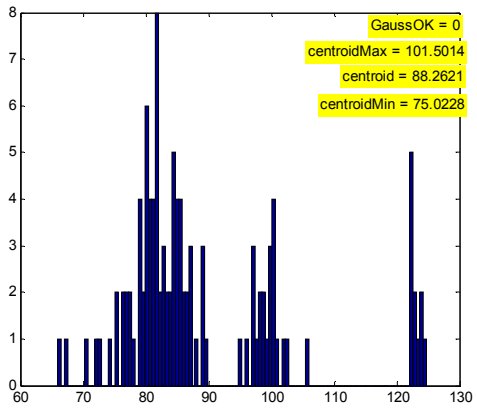

Fig 4.14 Incorrect Gaussian distribution

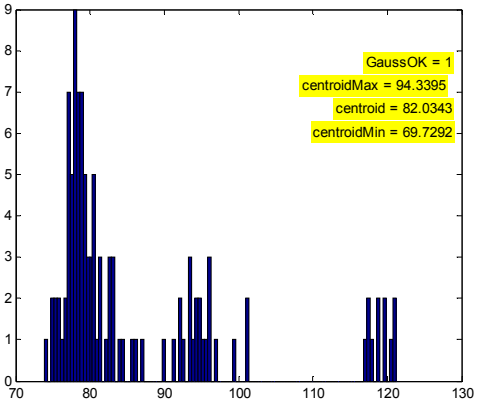

Fig 4.15 Correct Gaussian distribution

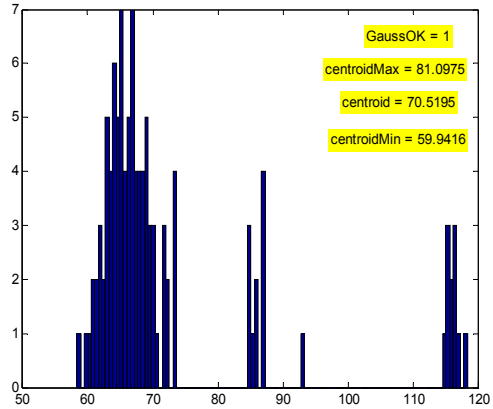

Fig 4.16 Correct Gaussian distribution

4.2.1.4 Filtering of the histogram results.

In the last step, the function *Final_B* is used for the adult's analysis. The design and the way that these functions work are explained in chapter in special functions section. The idea consists of taking the 20 results from the performance of the function *Final* and keeping the ones that has *GaussOk=1*.

As it is shown in Fig. 4.17, the distribution of the results is nearly a perfect Gaussian, inasmuch as at least three of the maximum frequencies are contained inside the interval formed by *centroidMin* and *centroidMax* bounds and hence the variable *GaussOk* turns "1".

The Gaussian resemblance of the histograms is the expected when the results are correct or are properly corrected.

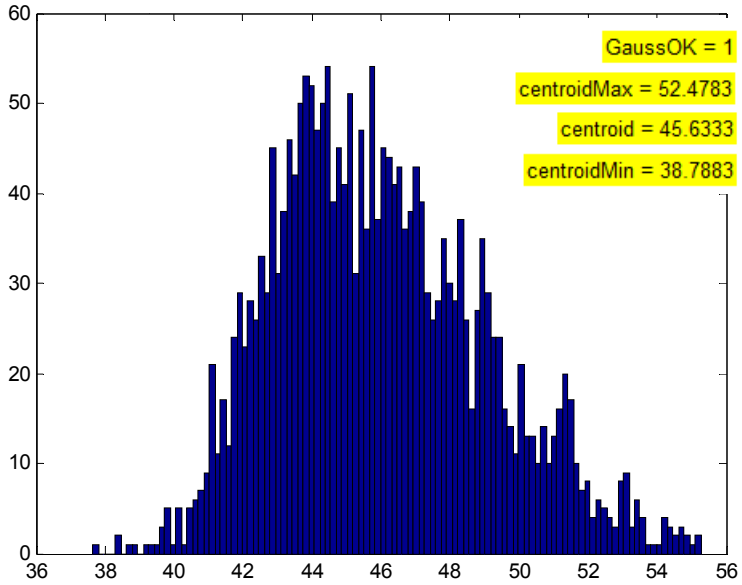

Fig 4.17 Histogram of the Final_B results

4.2.2 Newborns

In this section, 12 files from healthy newborns have been analyzed with the purpose of obtaining the same reference values for the spectrum of complex EBI that was achieved with the adults.

4.2.2.1 Corrections in the measurements

The Figures presented below have a deviation and artifacts suffered mostly at low and high frequencies. In Figures Fig.4.19 and 4.21 these anomalies are observed at low frequencies and in Fig.4.20 there is a deviation caused by capacitive leakage and an artifact, which strongly affects the frequencies from 900 Khz to 1000 Khz. Therefore, a correction of these measurements is necessary.

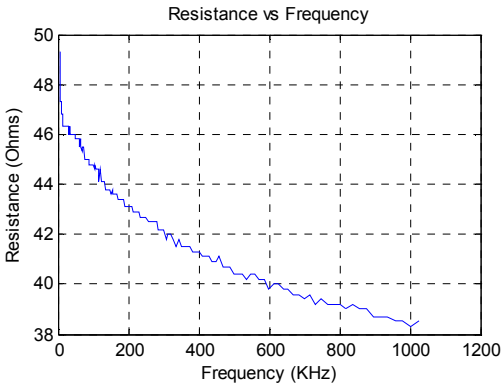

Fig 4.18 Resistance VS Frequency

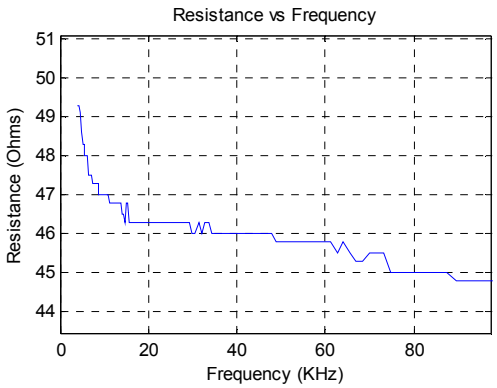

Fig 4.19 Artifacts in Resistance at low frequencies

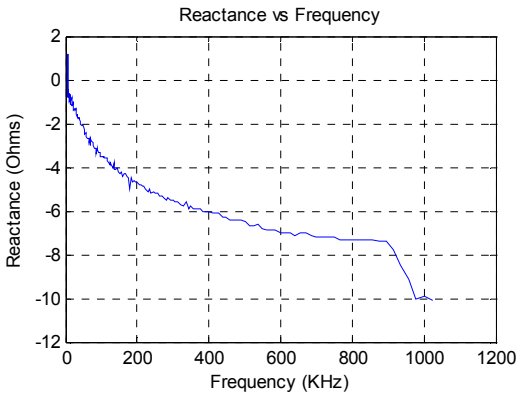

Fig 4.20 Reactance VS Frequency

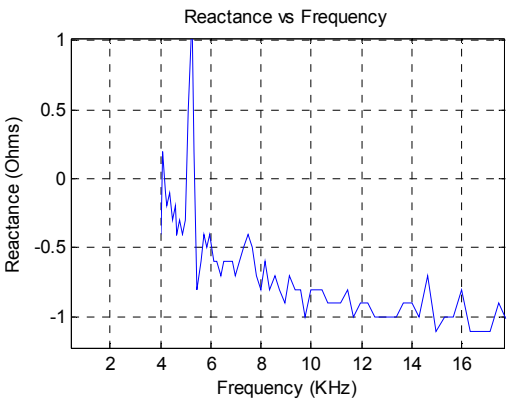

Fig 4.21 Artifacts in Reactance at low frequencies

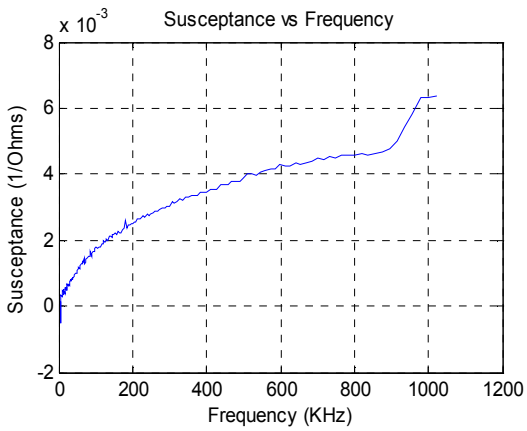

Fig 4.22 Susceptance VS Frequency

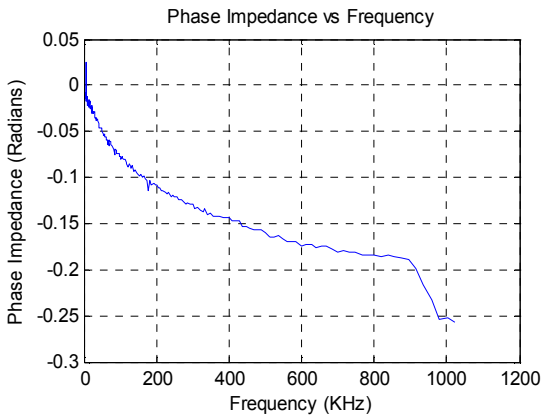

Fig 4.23 Phase Impedance VS Frequency

Fig. 4.24 shows a very bad fitting of Zfit and as a consequence of doing the fittings without correction and due the performance of Circular fit. It is also seen a correct behavior of the other fittings.

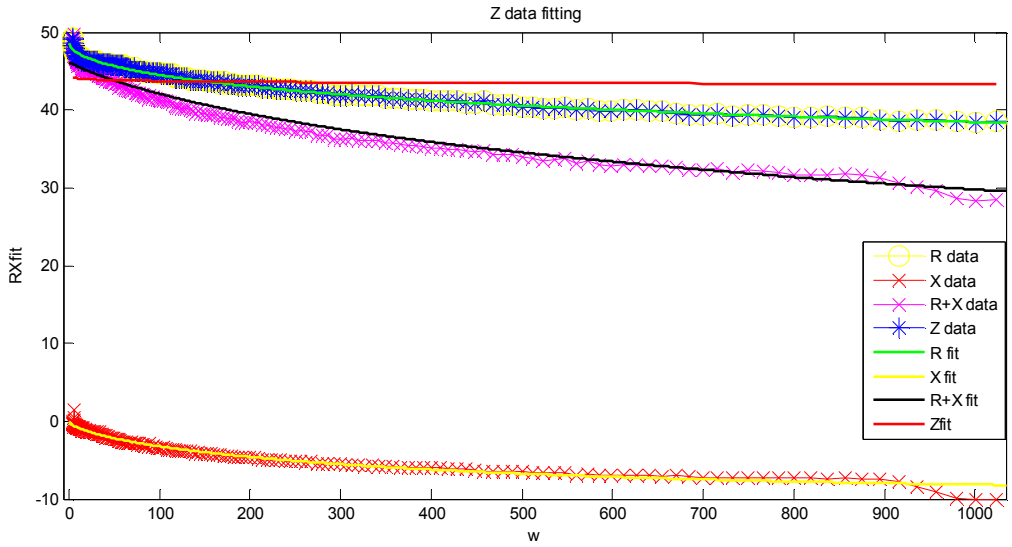

Fig 4.24 Fittings respect the original data without correction

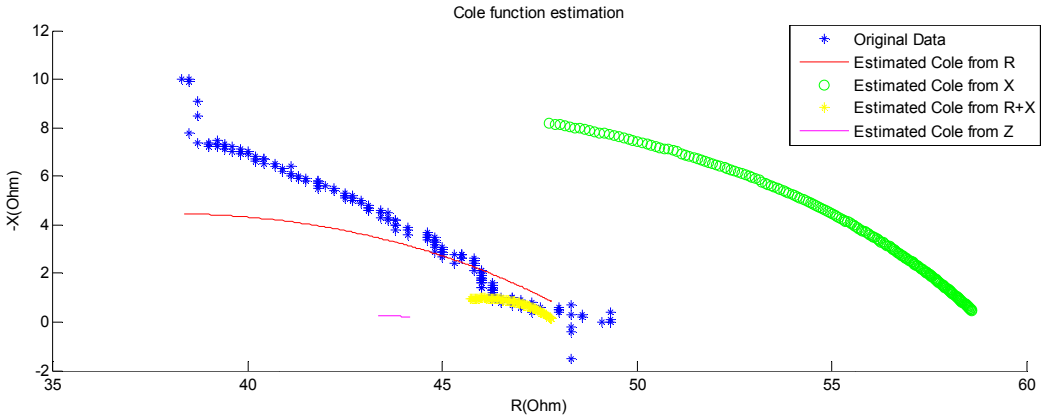

Fig 4.25 Cole Plot in each plane without correction

Although almost all of the fittings in Fig.4.24 seemed to be correct, the figure above shows the Original Data in blue and the other fitted curves in colored lines that are not good measurements. The best fitting according to the original data are the Estimated Cole from R(ω). Thus, theory presented in the previous chapters is fulfilled again. However, the Zfit results are not acceptable in this case.

The representation of the same file in the application BioImp is presented in the Fig. 4.26 and therefore the proper performance of the Toolbox is checked.

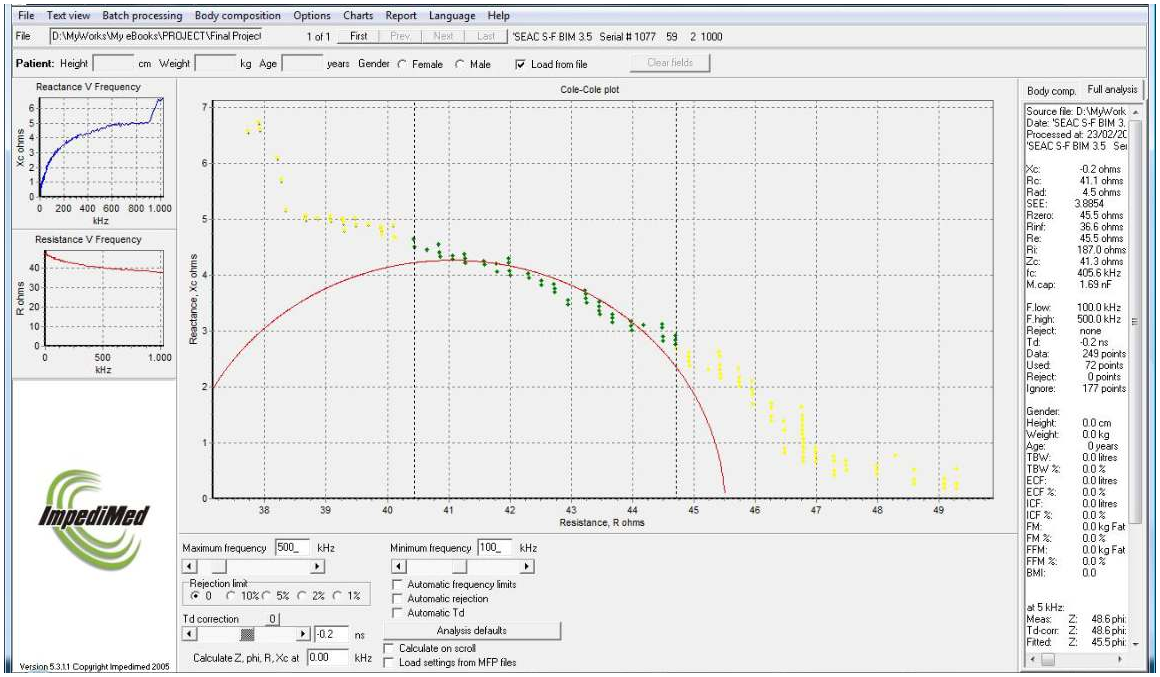

Fig 4.26 Bioimp representation of the file C025b of a newborn.

Once the errors are presented, a correction from 4 - 1024KHz is made using the *FCorrection* function.

The figures below show the resistance and the reactance with the corrections made. Even though the effect of the capacitive leakage is corrected, it is possible to see the same artifacts at low and high frequencies and in the case of resistance it is even worst due to a change near

900 Khz. Consequently, the correction function that were designed, are not useful to correct this kind of errors in the measurements in this fact is presented in chapter 5 of discussion.

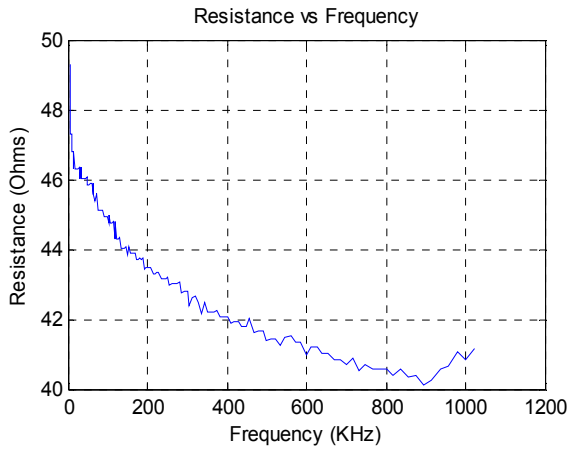

Fig 4.27 Resistance VS Frequency with correction

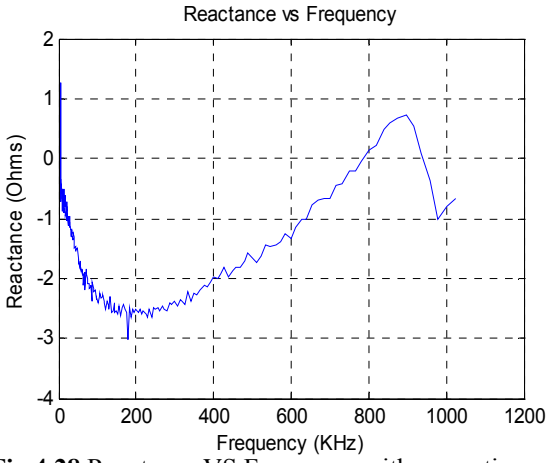

Fig 4.28 Reactance VS Frequency with correction

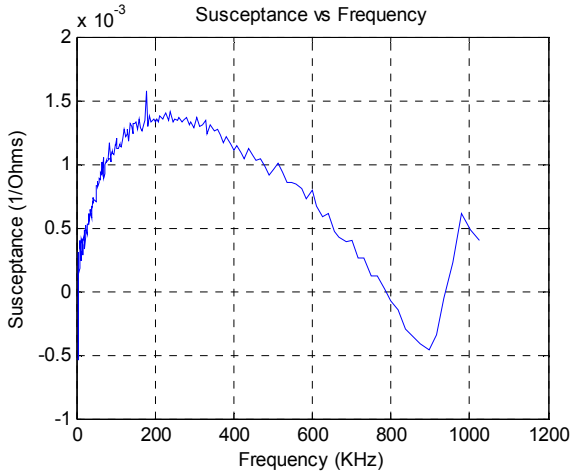

Fig 4.29 Susceptance VS Frequency with correction

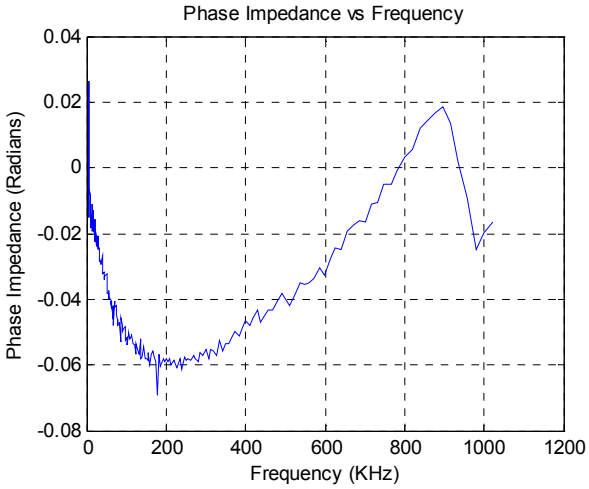

Fig 4.30 Phase Impedance VS Frequency with correction

The fittings in each plane with corrections are presented in Fig.4.31. As noticed, the different fittings follow the original data in an acceptable way, only the Zfit and R+X fit are quite far away and also X fit in some stretch of the graph.

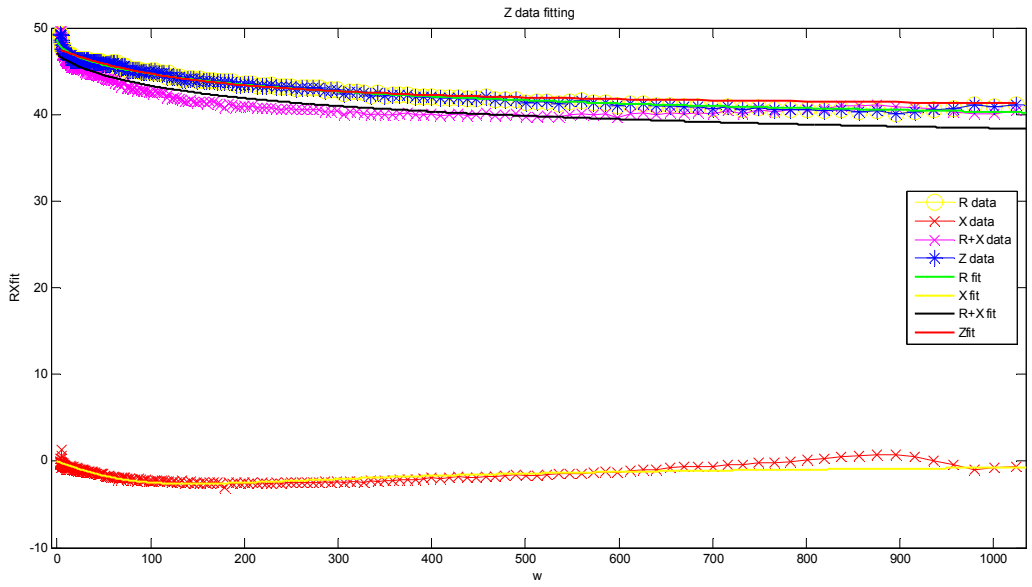

Fig 4.31 Fittings respect the original data with correction

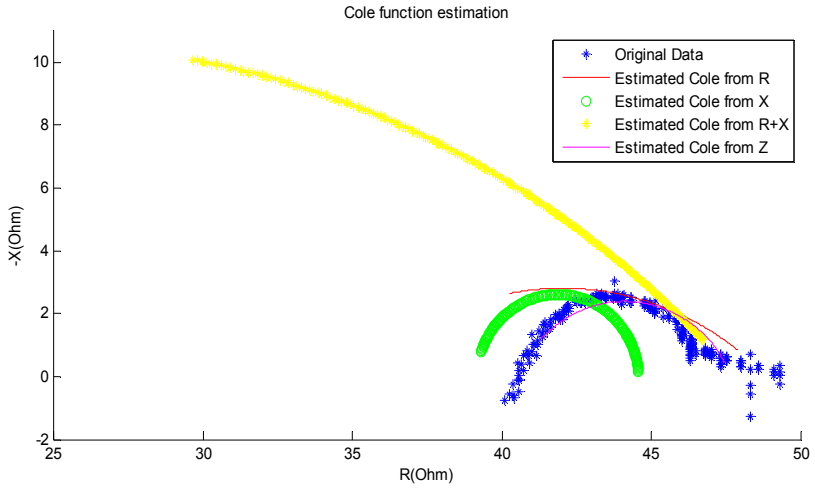

Fig 4.32 Cole Plot in each plane with correction

In the figure above and compared with the Cole Plot without correction of Figure 4.25 a better Cole Plot is simple to identify, however a dispersion at low frequencies of the corrected data, blue line, is impossible to cancel. The error of this dispersion is explained in Chapter 5, the discussion and is intimately linked with the value of the coefficients, such as the central frequency.

As presented before, the best fitting for the spectrum of the Cole Plot is performed with R(ω) and hence the whole analysis is going to be made in reference to this plane.

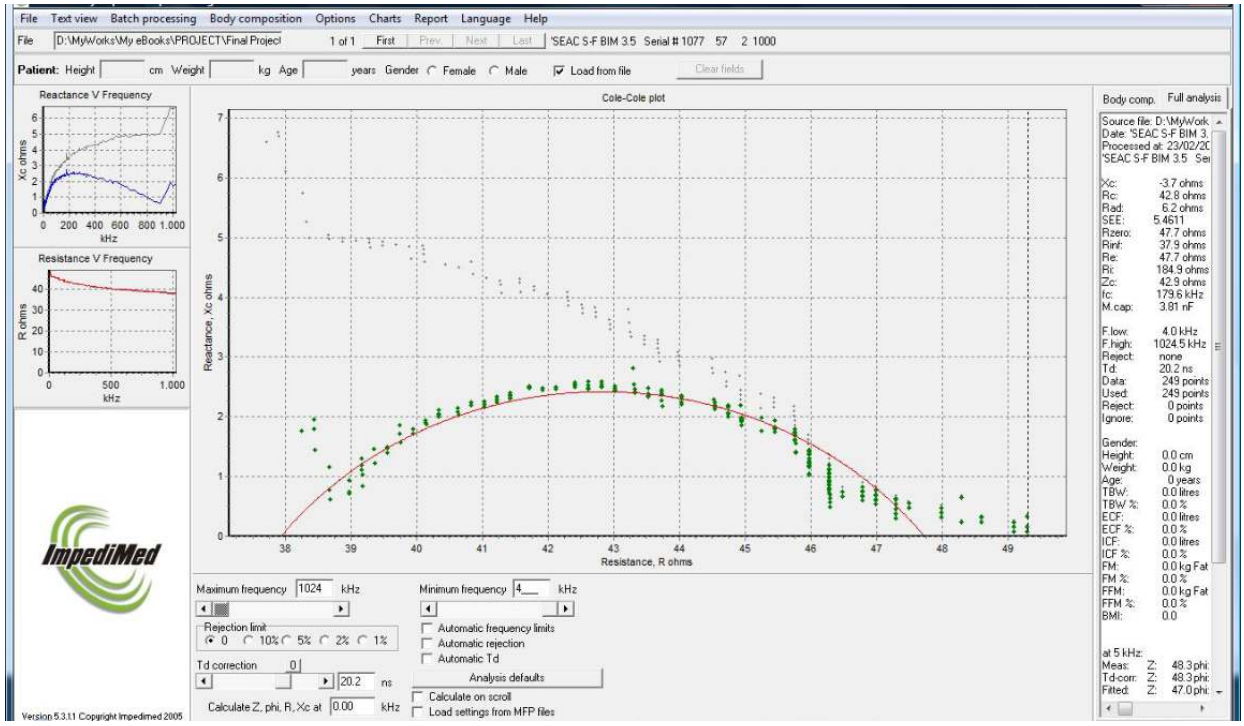

Fig 4.33 Bioimp representation of the file C025b of a newborn with correction

Attending to the figure above, that shows the same file manually corrected with BioImp, a good comparison between the coefficients and the Cole Plots from the ToolBox will be made.

The maximum value of reactance is around -3 ohms and the maximum resistance is around 48 ohms, these are not exactly the same values obtained in our application and thus is induced by the present distortion. Interestingly, the values of R_0 and the R_∞ are in the same order of magnitude comparing with Cole Plot in Fig 4.33 and the values in Table 4.4.

On the other hand, the result of *fcR* in our application is 64.8502 Khz and in BioImp is 179.6 Khz, which confirms the malfunction, due to distortion and hence all the results in the next sections are going to be incorrect.

	fcR	**fcX**	**fcRX**	**fcZ**
Value (Khz)	64.8502	22.7214	352.7401	21.7864

Table 4.3 Central Frequencies in each plane

	R0	R∞	τ	α
R(ω)	48.8310 Ω	32.2546 Ω	0.0025 s	0.5000
X(ω)	44.5746 Ω	39.1519 Ω	0.0070 s	0.9766
R(ω)+X(ω)	47.8400 Ω	1.2936e-08 Ω	4.5120e-04 s	0.5312
Z plane	47.6757 Ω	40.5207 Ω	0.0073 s	0.7492

Table 4.4 Coefficients in each plane of fitting

4.2.2.2 Cole fitting of EBI measurements in different intervals of each subject.

In this section, the same analysis, like in adults, have been made thanks to the functions *Final* and *Final_B*.

Once the original files from the Coronal brain measurements are corrected, those 12 files are introduced in the function Final, which takes each file and performs a Cole fitting in the R(ω) plane in the 110 intervals.

4.2.2.3 Filtering of the Cole fitting results and histogram.

The different files from the *Final* function have to be passed through three different filters to remove possible damaged data affected by measurement artifacts. The applied basic filters are the following, like the ones that were presented in previous sections.

The first filter is performed with the function *fcminus600* and not all values satisfy the condition. The values of *fcR* are between [31.9323 Khz, 1697 Khz].

The second filter provides results of Xcentre, which are between [-22.9789 Ω,-1.0935e-13 Ω], thus these values are completely correct.

The last filter removes values of Standard Error of Estimate less than 2%. The values are in the range of [0.24%, 0.82%].

Figures Fig.4.34 - 4.37 contain histograms with different types of distributions. Most of the results are *GaussOK* equal one. The subjects without Gaussian distributions are removed from further analysis.

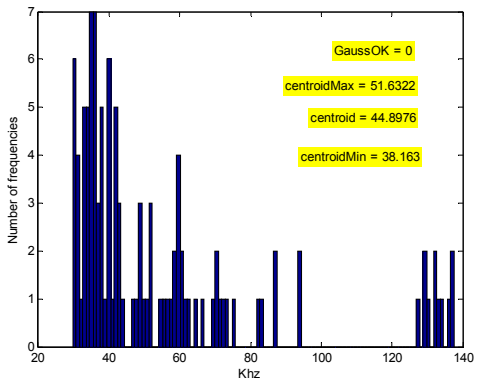

Fig 4.34 Incorrect Gaussian distribution

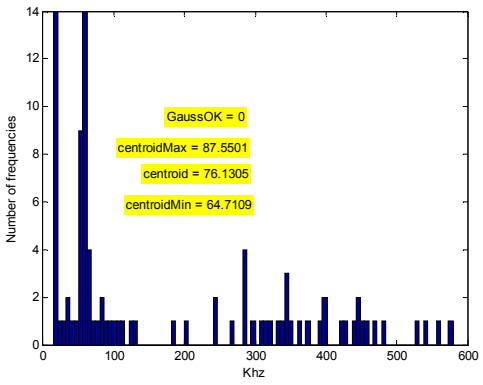

Fig 4.35 Correct Gaussian distribution

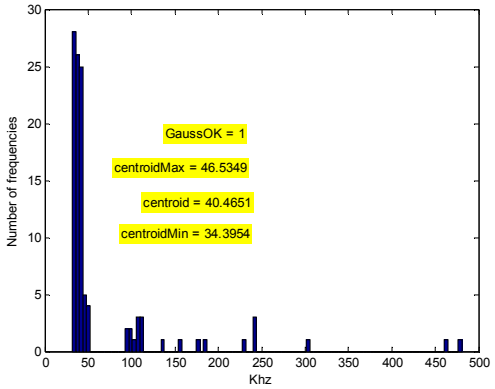

Fig 4.36 Correct Gaussian distribution

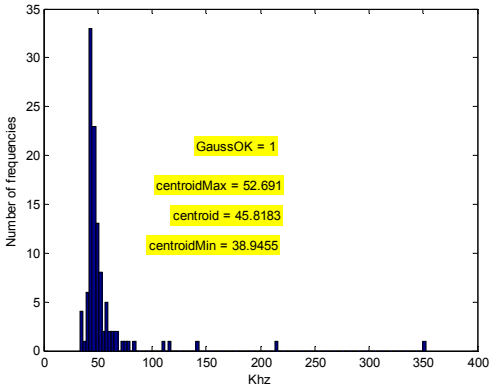

Fig 4.37 Correct Gaussian distribution

4.2.2.4 Filtering of the histogram results.

The function *Final_B* is used in the last stage of analysis. The 12 results from the performance of the function Final have been studied and only the ones that has GaussOk=1 are kept.

According to Fig.4.38, although the result distribution is nearly a Gaussian, the main lobe has shifted to the left side of the plot. This is due to the characteristic frequencies that reach values near 200 Khz, 300 Khz, 400 Khz, 500 Khz and 600 Khz in some stretch of the plot.

In conclusion, all that malfunctions in the designed functions is due to the present dispersion and artifacts of the original files that are impossible to correct.

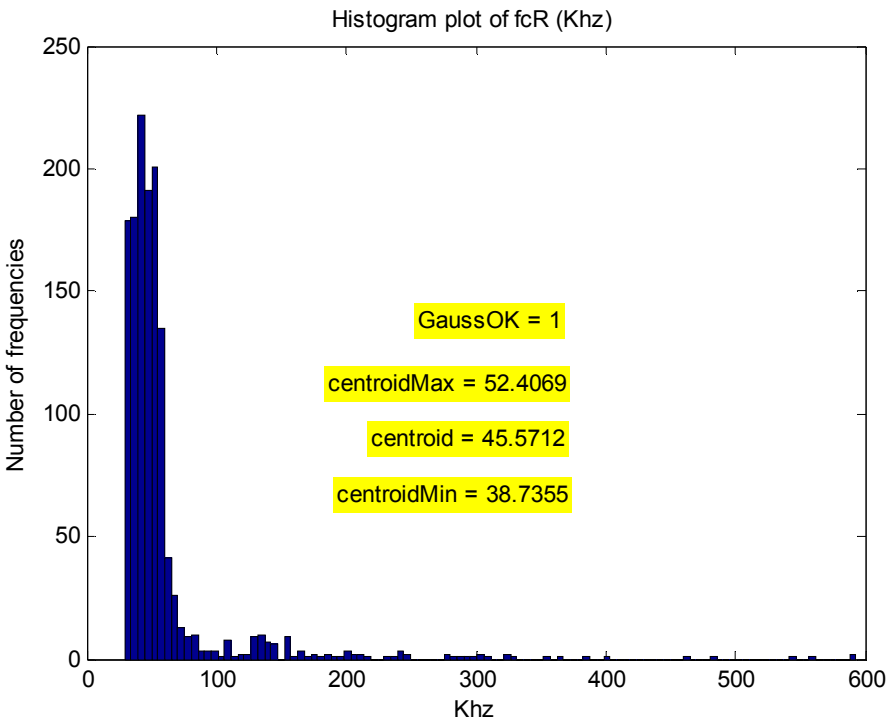

Fig 4.38 Histogram of the *Final_B* result

CHAPTER 5.

DISCUSSION

This thesis work has been done with the goal of supporting the development of a Software Tool Suite for Electrical Bioimpedance Spectroscopy Analysis, by creating functions to make a complete analysis of EBI spectroscopy measurements from a group of subjects and testing the general performance of the Suite analyzing real data. The following sections present the problematic aspects found along the process of design, implementation and the performance of implemented solution.

5.1. Estimated Cole from Zplane

As it was mentioned in Chapter 3, the best fitting according to the original data, is obtained when the Cole parameters are estimated from the resistance spectrum *R(ω)*. Although the *Zplane* method presents a very good fitting according to the original data, is not always working properly and that is the reason to perform the analysis designed in subsection special functions in chapter 3 with the *R(ω)* method.

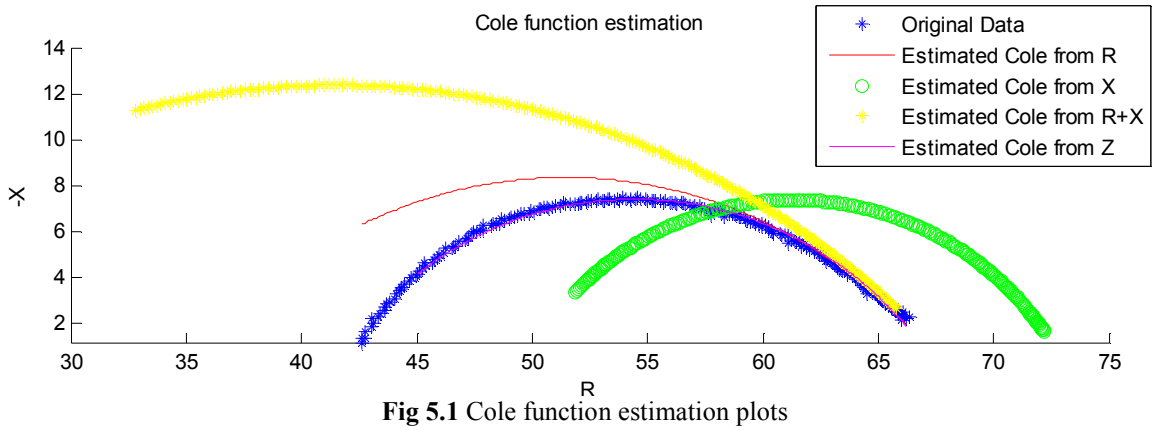

Fig 5.1 Cole function estimation plots

Within the malfunction, we can find Errors in the estimation of the reactance at high frequencies and poor estimations in the central frequency, because this fitting uses a method that estimates the complex centre and radius of the Cole plot and that radius can be influenced by the parasitic capacitor. If the radius and the other coefficients involved in this method were incorrect, the central frequency as the last parameter to calculate, would be calculated incorrectly.

These errors are shown in the next Figures that are based in the Cole Estimation provided by David Ayllón in Figure Fig.5.1 and its corresponding errors.

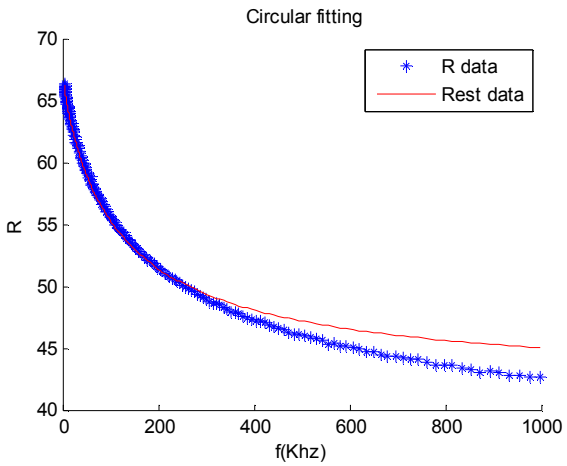

Fig 5.2 Original resistance vs Estimated resistance.

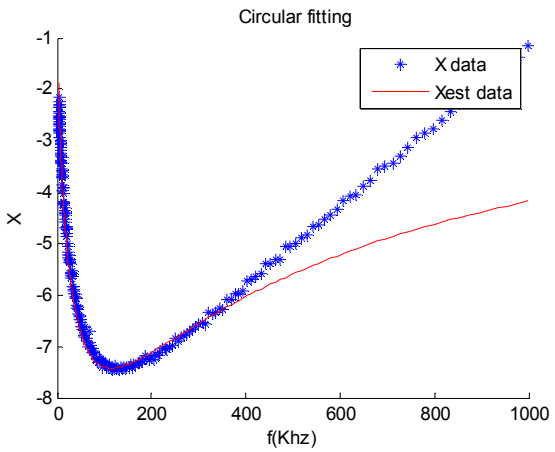

Fig 5.3 Original reactance vs Estimated reactance

Fig. 5.2 and 5.3 show the *Zplane* estimation of the resistance and the reactance is good until around 390 Khz and beyond this point it is going little bit away of the original data. However, the reactance upon the same point is further away from the original data

	R(w)	X(w)	R(w)+jX(w)	Zplane
SeeR	0.0736	2.1059	4.2898	0.7060
SeeX	1.5440	0.4934	4.2893	0.6343
See|Z|	0.1985	2.1026	3.4184	0.7473
SeeColePlot	1.5457	2.1630	6.0664	0.9490

Table 5.1 Standard Error of Estimate (SSE) for the curve fitting of the Resistance, Reactance, Impedance Module and Cole Plot from R(w),X(w), R(w)+jX(w) & Zplane

Finally and according to Table 5.1, the estimation from the *Zplane* performs its best fitting with the reactance. Although the Zplane is not the best fitting method, it is a very good way to fit the ColePlot as it is shown in the Table 5.1 and the figure, Fig 5.1.

5.2. Cole Start Values

There are some functions of the function *RXfit* used in the function *Fitting_planes*, that have a malfunction when there are artifacts on the EBI data and more specifically in the values of resistance and reactance at low and high frequencies. The origin of the malfunction resides in the nature of the function *ColeStartValues*, which calculate initial values for the four Cole parameters from R(ω) and X(ω) planes of original data by following the next method.

The calculation of the regression line as the line that fits the low frequency points of resistance, is depicted in Figure 5.4 thus, we can estimate α from its slope according to equation (5.5). In addition, R_0 can be estimated as the point where the line crosses the R(ω) axis, that is n, and R_∞ can be estimated as the point where the regression line at high frequencies crosses the R(ω) axis, that is q.

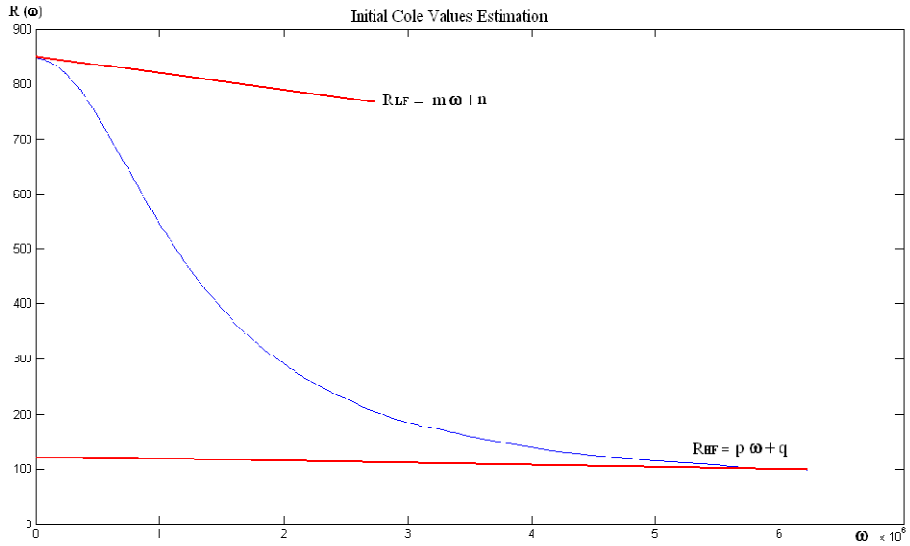

Fig 5.4 Regression lines estimation from R(ω) at low and high frequencies for the initial value estimation Cole parameters (Ayllón *et al.*, 2009)

$$\theta = -arctg(m) \tag{5.1}$$

$$\alpha = \left(\frac{\pi}{2} - \theta\right)\frac{2}{\pi} \tag{5.2}$$

Finally, τ can be estimated from X(ω), taking into account that the maximum reactance is found at the characteristic frequency, and τ =1/ .

Notice, that for all those estimations, it is necessary to have a good linearity in the low and upper regions of the frequency. Otherwise, if the plot depicts abrupt changes in those regions the start values estimation method that use the NLLS method of the function *RXfit* are going to be incorrect and the incorrectness of this estimation will produce the malfunction in the three different curve fittings methods using NLLS. These three evaluated functions are R(ω), jX(ω) and R(ω) + jX(ω), as described in the chapter of the ToolBox.

The following figures show the possible artifacts in the measurements of resistance and reactance that affect in the calculation of the Cole Start Values.

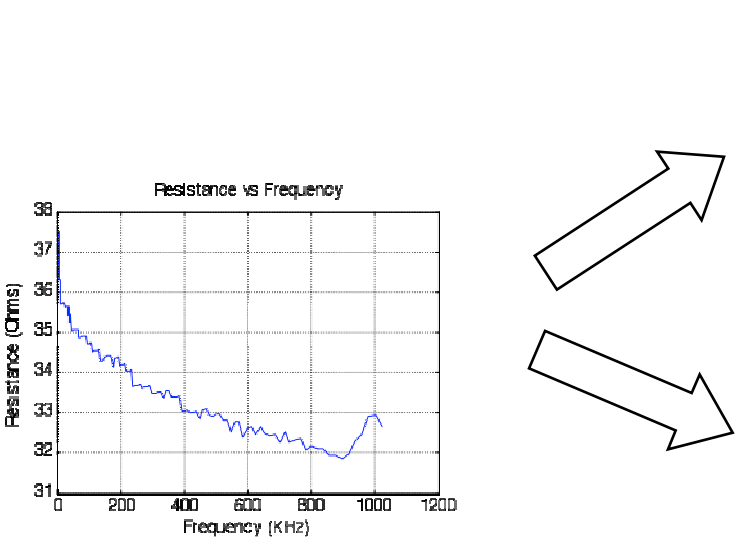

Fig 5.5 Problems in Resistance

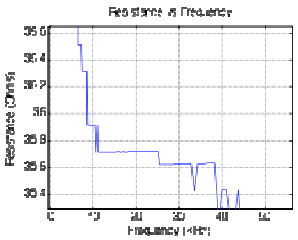

Fig 5.5.1 Zoom in at low frequencies

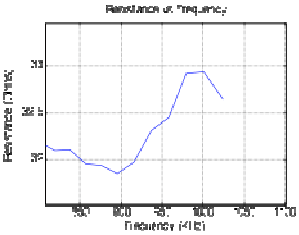

Fig 5.5.2 Zoom in at high frequencies

Another case of errors is that this function is designed to have at least 50 values of low and high frequencies to calculate the slopes and we can see this code box, in lines (1),(2),(3) and (4) of the Code 5.1 thus, in functions like *ColeRejection* is inconvenient due to it being possible that in the second loop of the performance, our remainder values, are less or equal to 100, consequently the *ColeStartValues* function won't be able to calculate such initial values.

```
function [startingVals]=coleStartValues2(z,w)

R=real(z);
X=-imag(z);

% We calculate the regression line at low frequencies
    Rlf=R(1:50);                                    (1)
    wlf=w(1:50);                                    (2)
    [Theta0 Theta1] = LMS_alg(wlf,Rlf, 0.005, 1, 1);


% alpha estimation
    fi=-atan(Theta1);
    alpha=((pi/2)-fi)*2/pi;

%R0 is the point where the line cross Y axis
    R0_init = fsolve(@(x)(x-Theta0)/Theta1,700);

% We calculate the regression line at high frequencies

    Rhf=R(end-10:end);                              (3)
    whf=w(end-10:end);                              (4)
    [Theta0 Theta1] = LMS_alg(whf,Rhf, 0.005, 1, 1);


%Ri is the point where the line cross Y axis
    Ri_init = fsolve(@(x)(x-Theta0)/Theta1,700);
    if Ri_init < 0
        Ri_init=10;
    end

%Stimate Tau from X (maximun value)
    [n,i]=max(X);
    wc=w(i);
    T_init=1/wc;


startingVals=[R0_init, Ri_init T_init alpha];
```

Code 5.1 Code of *coleStartValues2* function

5.3. Correction Function

In the subsection Corrections in the measurements, of the section Newborns contained in the chapter 4, we found artifacts and abrupt changes in the measurements that are impossible to correct. Furthermore, those errors in the EBI measurements are easy to find at low and high values of frequencies and as it was commented in the previous section, cause a non-satisfactory result of the function *RXfit*. In the following figures, which show the susceptance plot, we can see these artifacts after and before the correction.

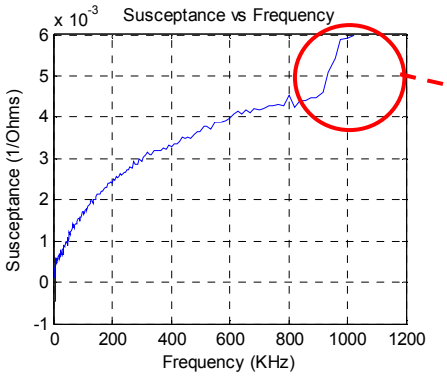

Fig 5.6 Before Correction

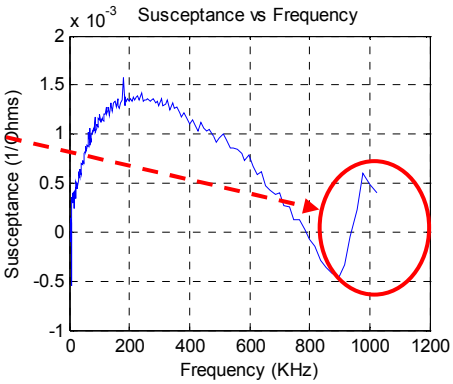

Fig 5.7 After Correction

According to the thesis work "*Hook Effect on Electrical Bioimpedance Spectroscopy Measurements. Analysis, Compensation and Correction"* by Rubén Buendia, we can find a possible reason for that malfunction. In fact, the *Correction* function works with the approach of estimating the stray capacitance from the susceptance and that depends on the ability of the system to perform measurements at high frequencies. Such ability is not available in many of the spectrometers currently used on the market and the accuracy of such estimation depends precisely on the measurements obtained at high frequencies. Therefore, if the measurements at high frequencies are wrong, it will be impossible to correct.

CHAPTER 6.

CONCLUSIONS & FUTURE WORK

6.1. General Conclusions

The performance of the functions presented in the previous sections and their respective success relies on the quality of the EBI measurements. If the measurements exhibit errors like artifacts, no-linearity or deviation, a correction should be made. It is occasionally impossible to correct and thus the performance of the fittings are not suitable, especially in the assessments of the central frequencies.

The Correction factor and the Correction Function, have the limitation related with the estimation of the stray capacitance. The proposed method to estimate the stray capacitance has a frequency dependency and requires performing EBI measurements at high frequencies, which in most of the cases present problems.

The fitting studied methods work relatively well, especially the *R(ω)* and *Zplane*. But the methods that used NLLS algorithm, like the first one, have problems to find the initial values especially when there are few values of EBI measurements or when these measurements at low and high frequencies have not good linearity.

6.2. Future Work

This section includes possible actions to solve the errors found in some functions and the analysis done in this thesis work.

6.2.1 Preliminary Analysis of Measurements

In previous chapters, we found some critical aspects to deal with, such as problems with artifacts, distortion and abrupt changes that depict the measurements of resistance and reactance. An interesting study of that would be to identify which measurements are analyzable or not, that is to implemented an artifact detector for discarding potential outliers. For example: looking up the abrupt changes in the plots at high frequencies, influence of noise in the measurements, in fact, this work had been done on a mathematical model before, although it could be interesting to assess more models of noise to guarantee the correct performance of functions such as *Fitting Planes* or correction functions.

6.2.2 Changes in Cole Start Values

Another interesting action would be to change the design of the function *Cole Start Values* because it causes a lot of problems in the cases that were mentioned before. Hence, a function that can provide another way to calculate the initial values to perform the fittings without any restriction in the first or last values or any restriction in a minimum values can be good.

6.2.3 Correction function

As mention previously, this function has been done and tested with proper measurements at high frequencies. Moreover, in the thesis by (Buendia, 2009) we can find that the equivalent model for the stray capacitance was simplified, neglecting the value of the electrode polarization impedance, which might have had an important effect in the compensation and correction method. This fact should be investigated to achieve a clear understanding of the hook-alike deviation and its correction.

REFERENCES

Ayllón D, Seoane F and Gil-Pita R 2009 Cole Equation and Parameter Estimation from Electrical Bioimpedance Spectroscopy Measurements - A Comparative Study. In: *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE* (Minneapolis: IEEE) pp 3779-82

Buendia R 2009 A Novel Approach for Removing the Hook Effect Artefact from Electrical Bioimpedance Spectroscopy Measurements. In: *School of Engineering,* (Borås: University of BOrås)

Cole K S 1940 Permeability and impermeability of cell membranes for ions. *Quant. Biol.* **8** 110–22

Fricke H 1924 A Mathematical Treatment of the Electrical conductivity of Colloids and Cell Suspensions *Journal of General Physiology* 375-83

Grimnes S and Martinsen O G 2005 Cole electrical impedance Model-a critique and an alternative *Biomedical Engineering, IEEE Transactions on* **52** 132-5

Hill D M 2008 ANAT3231 CELL BIOLOGY. In: *School of Medical Sciences,* ed U o N S Wales-Sidney. (Sidney

Ivorra A 2003 Bioimpedance Monitoring for physicians: an overview

Jaffrin MY M M, Le Gourrier A, Boudailliez B. 1997 Extra- and intracellular volume monitoring by impedance during haemodialysis using Cole-Cole extrapolation. *Med Biol Eng Comput.* **35(3)** 266-70

Kanai H S K, Haeno M. 1983 Kanai, H., Sakamoto, K., andHaeno, M. (1983): 'Electrical measurement of fluid distribution in human legs: estimation of extra and intra-cellular fluid volume,'J. Microw. Power,18, pp. 233–243 *J Microw Power.* **18** pp. 233–43

Kushner R F and Schoeller D A 1986 Estimation of total body water by bioelectrical impedance analysis *Am J Clin Nutr* **44** 417-24

OLSSON T, DAILY W and VICTORIN L 1970 Transthoracic Impedance: I. Theoretical Considerations and Technical Approach *Acta Paediatrica* **59** 15-27

Schwan H P 1957 Electrical properties of tissue and cell suspensions *Adv Biol Med Phys* **5** 147-209

Schwan H P 1994 Electrical properties of tissues and cell suspensions: mechanisms and models. In: *Proceedings of 16th Annual International Conference of the IEEE Engineering in Medicine and Biology Society,* (Baltimore, MD, USA: IEEE) pp 70-1

Schwan H P 1999 The Practical Success of Impedance Techniques from an Historical Perspective *Ann N Y Acad Sci* **873** 1-12

Schwan H P and takashima S 1993 *Encyclopedia of Applied Physics,* ed G L Trigg (New York and Weinheim: VCH publishers)

Seoane F 2007 Electrical Bioimpedance Cerebral Monitoring: Fundamental Steps towards Clinical Application. In: *Signal & Systems,* (Göteborg: Chalmers University of Technology) p 154

Songer J E 2001 Tissue Ischemia Monitoring Using Impedance Spectroscopy: Clinical Evaluation. In: *Biomedical Engineering,* (Worcester: Worcester Polytechnic Institute)

Zou Y and Guo Z 2003 A review of electrical impedance techniques for breast cancer detection *Med Eng Phys* **25** 79-90