POLITECNICO DI MILANO
Facoltà di Ingegneria
Corso di Laurea in Ingegneria Biomedica

PROGETTO IN ITINERE

Implementazione di un modello morfometrico dell'albero tracheo-bronchiale durante flusso-limitazione.

Contenuti:
Lo scopo del progetto è quello di sviluppare un modello morfometrico delle vie aeree e del polmone umano per l'interpretazione di dati sperimentali acquisiti su pazienti di patologie ostruttive.

Relazione di: Irene Acerbi
Matr. 647780

Tutor Universitario: Prof.Andrea Aliverti

Tutor: Ing. Raffaele Dellacà

Anno Accademico 2002-2003

POLITECNICO DI MILANO
College of Engineering
First Level Course of Biomedical Engineering

CURRICULAR PROJECT:
Implementation of a morphometric model of the tracheo-bronchial tree during flow-limitation.

Contents:
The aim of the project is to develop a morphometric model of the airways and human lung in order to give an interpretation of experimental data from patients suffering diseases causing bronchoconstriction.

Essay by: Irene Acerbi
Student ID: 647780

University Tutor: Prof.Andrea Aliverti

Tutor: Dr. Raffaele Dellacà

Academic Year 2002-2003

INDEX

INTRODUCTION

The aim of this project is to implement a morphomertic model of the human airway tree for computing the total respiratory system input impedance (Zin).

The software shall mimic the mechanical impedance of the airways tree and the human lung in order to calculate Zin and simulate the effect of different patterns of bronchoconstriction as experienced by patients suffering from obstructive diseases such as chronic bronchitis, enphysema, chronic bronchial asthma, COPD (chronic obstructive pulmonary disease), etc.

Of all diagnostic procedures used for investigating the respiratory system, the Forced Oscillations Technique (FOT) provides an easy way to extract meaningful parameters about the mechanical behaviour of the tracheo-bronchial tree. FOT is a non-invasive method that requires only passive cooperation from the patient and it can be performed during quiet breathing. This method can provide specific and reliable insights on the mechanical properties and mechanisms that contribute to breathing. Specifically, an external pressure forcing signal is applied to the respiratory system and the resulting flow is measured. As a result, FOT can provide the estimation of several transfer functions of the respiratory system [1] the value of Zin versus frequency. (3,8)

The impedances are defined as the transfer function between flow and pressure at certain frequencies. In particular, the lung input impedance, $Z_{in}(j\omega)$ is obtained when both flow and pressure are measured at the airway opening and it is defined as the ratio of the Fast Fourier Transform (FFT) of pressure $P_{ao}(j\omega)$ and the FFT of flow, $V_{ao}(j\omega)$ both measured at the airways opening:

$$Z_{in}(j\omega) = \frac{P_{ao}(j\omega)}{V_{ao}(j\omega)}$$

where ω = angular frequency.

In order to give an interpretation of the value of impedance obtained with FOT, a simple model of the airway system is usually adopted. It is a R-C series model (Figure 1) that consists in two lumped elements, representing the airways frictional resistance opposed to the flow (R) and the compliance of the respiratory system (C), which is the mathematic reciprocal of the elastance (C=1/E) and represents the elastic properties of tissues and the compressibility of the gas contained into the lung.

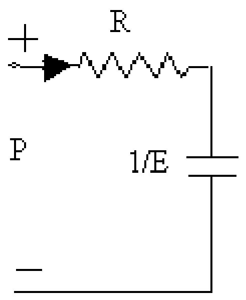


Figure 1

One-compartment model of the respiratory system.

According to this model, the real part of the impedance is a measure of lung resistance:

$$R=Re(Zin)$$

and elastance may be related to the imaginary part by

$$E=-\omega Im(Zin)$$

This first-order linear model is very simple, as it doesn't account for the heterogeneous distribution of the mechanical properties throughout the respiratory system. Moreover it considers constant parameters, while flow resistance and lung elastance are functions of lung volume, flow and frequency. Frequency dependence of the real and imaginary part of the input impedance increases especially in case of inhomogeneous constriction throughout the tracheo-bronchial tree (as common in obstructive disease), and in this condition the simple model of Figure 1 can not describe experimental data.

The description of experimental Zin obtained from patients with obstructive airway disease requires to adopt different and more accurate models. These models shall account for uneven constriction patterns throughout the bronchial tree, airway walls compliance and tissue viscoelasticity. For these reasons, the most appropriate model to use for the study and interpretation of Zin in patients with obstructive disease is a morphometric model of the airway tree that describes the structure/function relationship in the lung.

In this models the geometric structure of the bronchial tree is combined with the airway wall and alveolar tissue mechanical properties to describe the whole system behavior.

THE MORPHOMETRIC MODEL

This work is based on an advanced tree model that uses a morphometrically accurate branching system, invokes gas compliance and has the option of floppy airway tubes. The model allows for an input constriction of the diameters according to the tree geometry (all the tree/ certain generations only). The branching geometry is based on the Horsfield morphometric model.

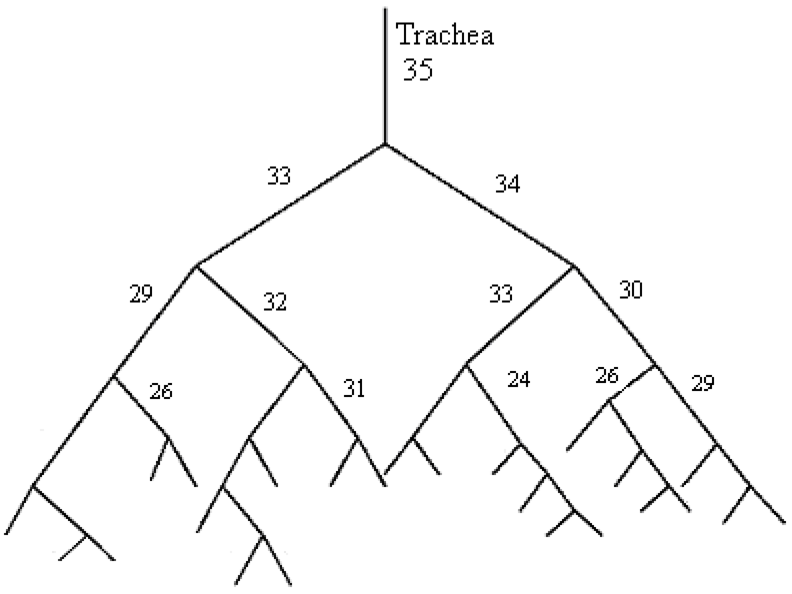


Figure 2 Example airway tree with branches numbered according to the pattern of the airway orders by Horsfield.

The human asymmetrical airway branching model as defined by Horsfield has the characteristic that the ordering by Horsfield begins in the periphery of the lung (order 1) and terminates at the trachea (order 35), as shown in figure 2. At each bifurcation the parent branch divides into two daughter branches.

To provide a tree for this project it was used an existing software, written in C++ language, that creates the airway tree. It is a 50401 branches tree, saved in a mat file, every branch is described by 5 values: the generation number, the flow, the diameter, the length and the node, the 3-D coordinate.

The generation number describes the branching pattern, beginning at the trachea (generation 0). The numbering rises till the ending branches leading to the alveoli (generation 28), as shown in Figure 3:

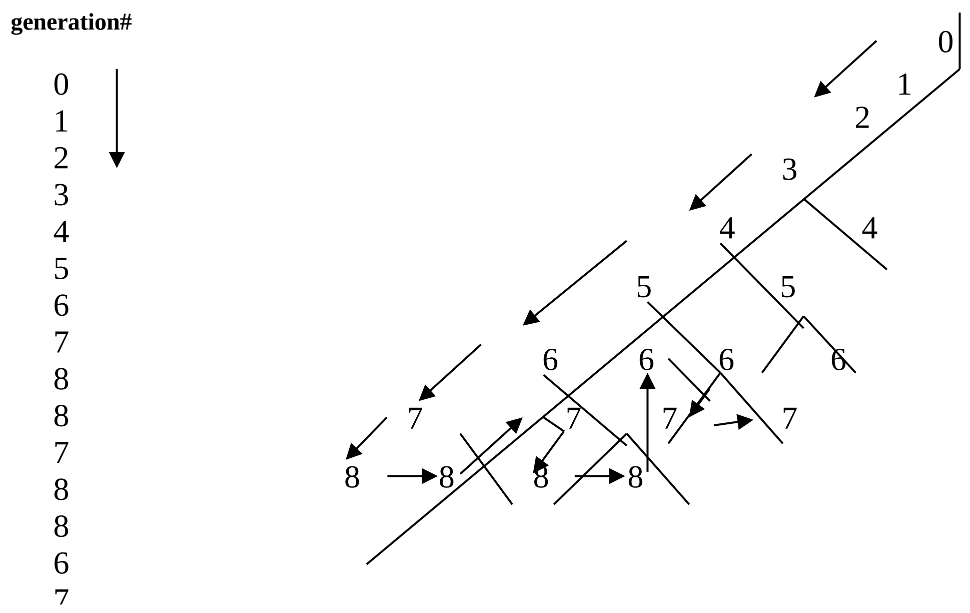


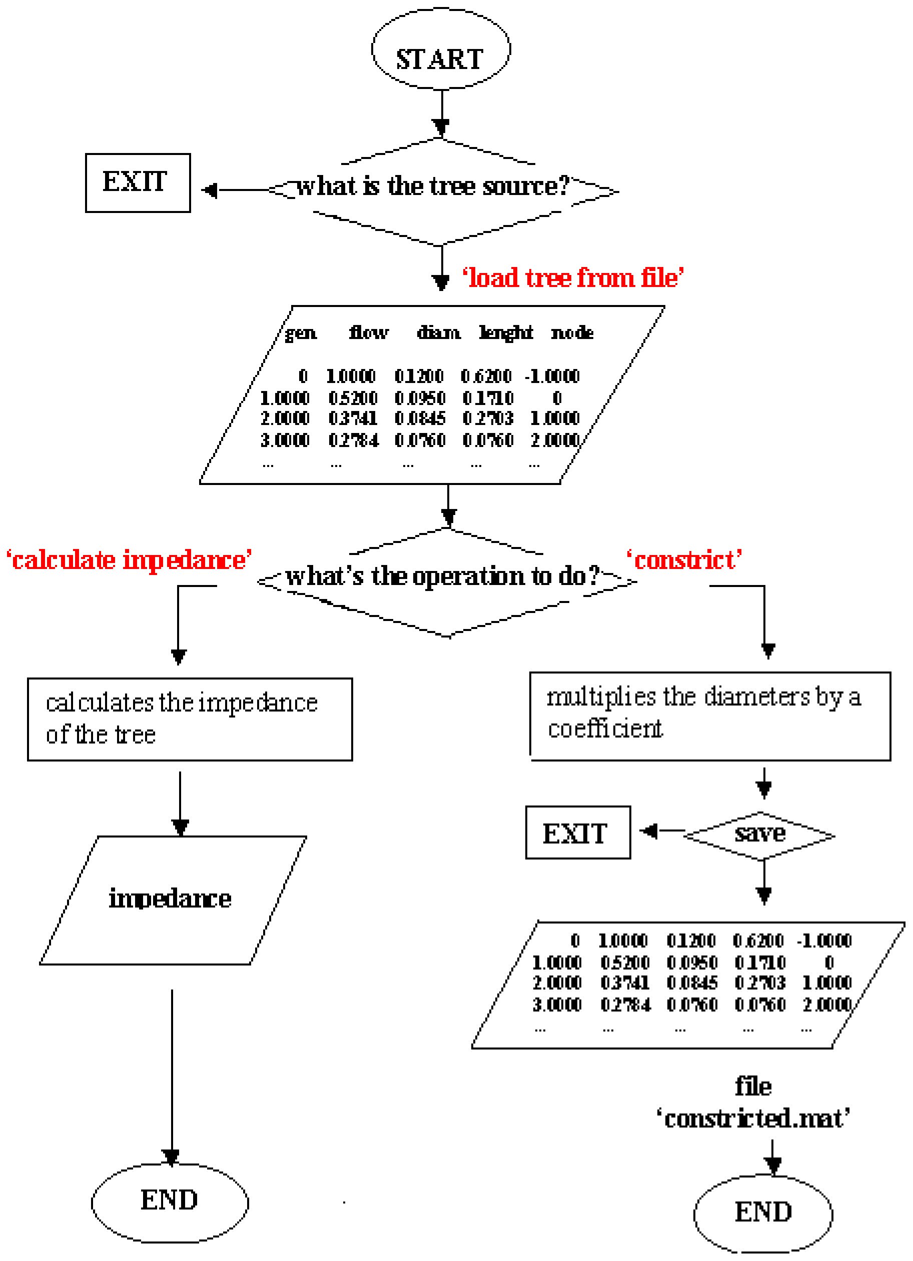
Figure 3. Reconstructing of the geometry of the tree from the sequence of the generation numbers.

Di ers and lengths are order dependent according to the algorithm which generates the tree.

In 1997, Lutchen and Gillis published a method to allow heterogeneous constriction of airways(4). Then in 1999 (11), Gillis and Lutchen explicitly incorporated airway smooth muscle wall areas distinct for healthy, mild and severe asthma to permit heterogeneous smooth muscle constriction in the tree. This model allows prediction of dynamic lung resistance and elastance for an asymmetrical system. Gas compliance and floppy wall tubes are also included in this model.(2)

MAIN PROCEDURE

The software main procedure is composed of three parts which are the loading of the source tree from a file, and two principal functions to work on it, the calculation of the tree impedance, and the airways constriction.



gen	flow	diam	lenght	node
0	1.0000	0.1200	0.6200	-1.0000
1.0000	0.5200	0.0950	0.1710	0
2.0000	0.3741	0.0845	0.2703	1.0000
3.0000	0.2784	0.0760	0.0760	2.0000
...	...	...	...	...

0	1.0000	0.1200	0.6200	-1.0000
1.0000	0.5200	0.0950	0.1710	0
2.0000	0.3741	0.0845	0.2703	1.0000
3.0000	0.2784	0.0760	0.0760	2.0000
...	...	...	...	...

Figure 4 . Schematic representation of the main procedure.

In order to run the program all the functions shall be opened in the 'current directory' of the Matlab page. Then it is possible to run the program entering ‘program’ in the command window, or pressing F5 on the keyboard once the **program.m** file is open.

The program starts with loading the tree that will be used in the simulations. The tree must be formerly saved as ‘mtree’ in a file.mat. The variable ‘mtree’ shall be a 5 columns matrix with n rows. Each row will be interpreted as a branch of the airway tree, the five columns representing respectively the generation number, the flow, the diameter, the length, and the node, a 3-D coordinate. The functions involved in these operations are **source.m** and **load_tree.m**. The operator is asked to type in the name of the file to open. In case of error the program asks for the file name to be typed again, or eventually prints to the screen a message, and the program ends.

The variable ‘mtree’ is created in the ‘current workspace’ and the next step is to choose between two options: calculate the impedance of the tree or apply a constriction; the menu is provided by the function file **what_to_do.m**.

CALCULATING THE AIRWAY TREE IMPEDANCE

If the chosen option is ‘calculate the impedance’ the sub-function **impedanceR.m** receives as parameters the tree and a value of frequency. The routine will be repeated for every given value of the frequency array fixed by the programmer, in order to allow the calculation of the complex value of impedance through the spectrum.

The first step is to evaluate the number of terminal branches, NTR, and this is possible checking the generation number and eventually the node value. BranchA is final when the generation number of the following branch, branchB is equal to the generation of branchA-1. Otherwise it is possible to identify terminal branches considering the value in the 5th column, the spatial indicator: if nodeB is smaller than nodeA it means that A is terminal. These are the controls on which the function **terminals.mat** is based.

In order to obtain proper values of impedance all diameters and lengths must be scaled. With **SCALE.m** all lengths (4th column) are scaled so that the trachea is normalized to the length of 10 cm. All diameters (3rd column) are multiplied by the same factor. Then **FRC.m** finds the values at Functional Residual Capacity - that is the volume of the lungs at the end of a quiet expiration. Now the data describing the airway tree is ready to be used to calculate the value of impedance.

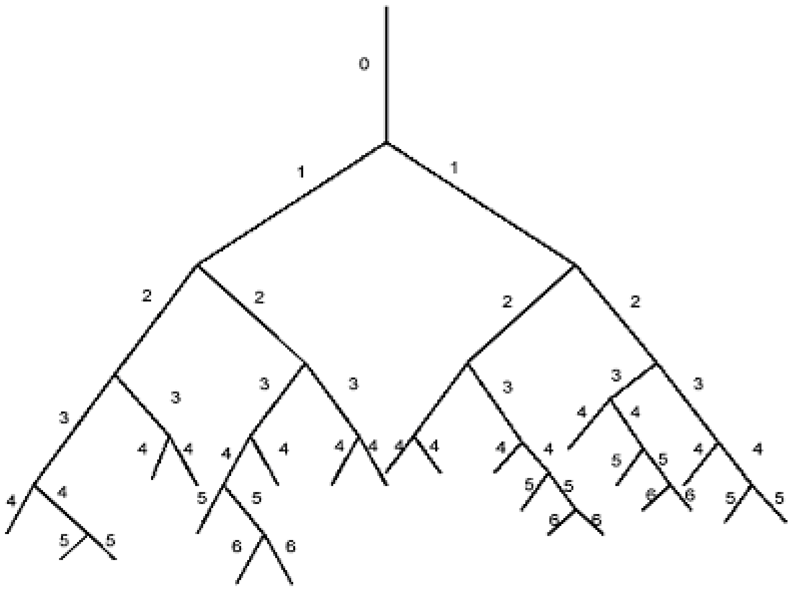


Figure 5 Example airway tree with branches numbered with the generation number, used for calculation of lung impedance.

The structure of the tree is represented in the source file by the generation numbers of the branches. The first column of the matrix is a sequence of numbers that corresponds to the tree branching as in the example in Fig.3.

The calculation of Zin consists in a for loop that will run as many times as there are branches in the tree. It starts at the trachea, which is the first branch in the airway tree file. Initially, i=1, hence branch [1] is the trachea (generation 0) and branch [i+1] is the left daughter branch (generation 1). The first step is to check if the generations of the two branches are equivalent. Since this is not true, the airway properties of the trachea (branch [1]) are pushed onto the stack “airway”. The index is incremented at the second “for loop”(i=i+1) and the program compares again the generations of branch [i] and branch [i+1], in this case branch [2] and branch [3] respectively. The process of checking to see if the generation of the current branch [i] is equivalent to the generation of branch [i+1] and pushing branch [i] onto the airway stack, continues until the two are equal, in which case we may take one of two paths. Let’s say branch[4] is found to be a terminal branch, this indicates that branching will continue along branch [5]. The impedance of branch [4], which is a terminal branch, is calculated by **CalcImp2.m**, and added to the alveoli component given by **Alveoli.m**. The value of impedance just found is stored and pushed onto “fstack”. The airway properties of branch [4] and branch [5] are pushed onto the airway stack, i is then incremented such that the branch[6] is the current branch and branch [7] is the next branch. The algorithm then proceeds as before to check if the generation of branch [i] is equivalent to branch [i+1]. The routine will stop again when it reaches two terminal airways. (3)

A different path is taken when the program finds that the generation of branch [i] is equal to branch [i+1]. In this case it combines the impedances of the two branches in parallel with the file **parallemImp.m**, as they represent a pair of terminal branches with a common parent branch. The airway stack is popped to retrieve the airway properties of the common parent branch, branch$_a$. The impedance of the parent branch is computed, **CalcImp.m**, and added in series, **seriesImp.m,** with the parallel combination of the terminal branches. Airway stack is then popped again, getting branch$_b$, and the generation is compared with that of the parent branch just used, branch$_a$. If comparing branch$_a$ and branch$_b$, it was found their generations were the same, the program pops "fstack" to retrieve the equivalent impedance of looking into branch$_b$. **RparallelImp.m** does the parallel of the two value of impedance finding Zeq. Then the program pops the airway stack to retrieve their common parent branch, calculates its impedance with **CalcImp.m**, and adds in series to Zeq with **seriesImp.m**. Airway stack is popped again and the new branch generation is compared to the generation of branch$_b$. If the generation of branch$_a$ is greater than that of branch$_b$, so branch$_b$ and branch$_a$ are both pushed back onto the airway stack. The current Zeq is also pushed onto "fstack", i is incremented and the process continues down the tree until it reaches two consecutive branches with the same generation.

This stack-based algorithm is repeated for a number of cycles equal to the number of branches in the airway tree. At the end **Calc_Zup.m** adds to the total impedance the value of impedance of the upper airways, set by the programmer, and finally Zeq=Zin.

By traversing the tree in this manner, it is possible to compute the impedance of each branch individually and thus can incorporate anatomic changes such as heterogeneous broncho-constriction and airway smooth muscle shortening (ASM) into simulation studies. .

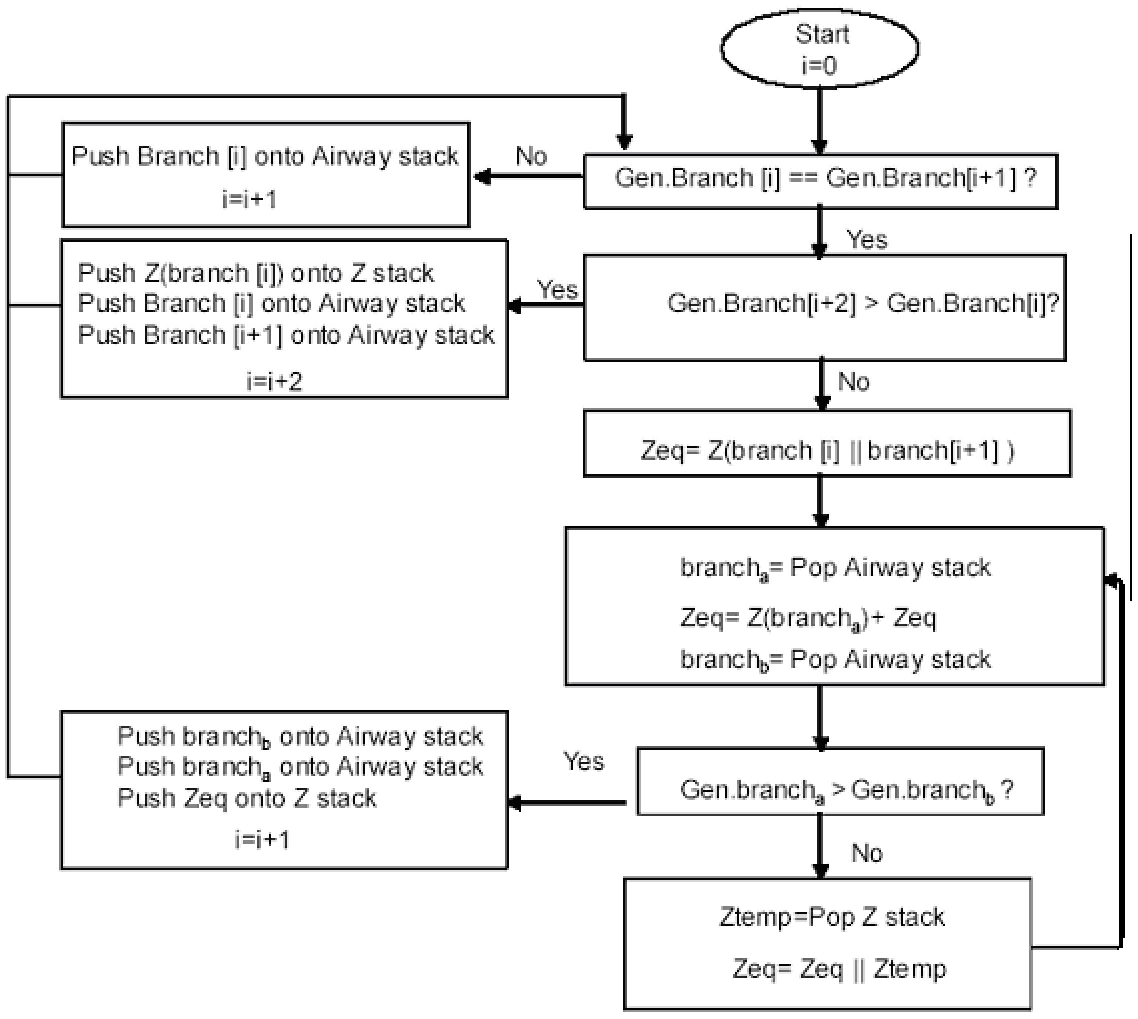

Start
i=0
Gen.Branch [i] == Gen.Branch[i+1] ?
No
Push Branch [i] onto Airway stack
i=i+1
Yes
Gen.Branch[i+2] > Gen.Branch[i]?
Yes
Push Z(branch [i]) onto Z stack
Push Branch [i] onto Airway stack
Push Branch [i+1] onto Airway stack
i=i+2
No
Zeq= Z(branch [i] || branch[i+1])
branch_a= Pop Airway stack
Zeq= Z(branch_a)+ Zeq
branch_b= Pop Airway stack
Gen.branch_a > Gen.branch_b ?
Yes
Push branch_b onto Airway stack
Push branch_a onto Airway stack
Push Zeq onto Z stack
i=i+1
No
Ztemp=Pop Z stack
Zeq= Zeq || Ztemp

Figure 6 Flow chart for traversing through the airway tree.(3)
Algorithm terminates when i=size of airway tree and Zin=Zeq.

In the algorithm there are two different functions which calculate the impedance of the tubes in different ways according to the location of the branch: **CalcImp.m** for all branches inside the tree, **CalcImp2.m** for terminal branches only.

CalcImp.m finds a value of the complex impedance of the tube $Z=R+j\omega I$, where ω is equal to $2\pi f$. R is the resistance in $cmH2O \cdot l^{-1} \cdot s$, while I is the fluid inertance in $cmH2O \cdot l^{-1} \cdot s^2$. The real part (R) is obtained according to Poiseuille's Law, as $R=(8\mu L)/(\pi r^4)$, μ is the fluid viscosity, L is the length (4th column of the tree matrix), r is the tube radius, found thanks to the diameter in the 3rd column of the tree matrix. The imaginary part is ωI, $I=\rho_{gas}L/(\pi r^2)$, where ρ_{gas} is the gas density. All constants are assigned by the user directly in the text of the function. Then the value of the impedance of the tube is combined to the gas compliance and with the wall tissue component. The equations used in **CalcImp.m** are referred to the single generation model in Figure 7.

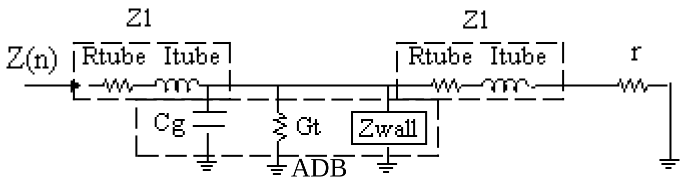


Figure 7 The impedance of a single airway generation considered in the simulation is based on the model by K.R. Lutchen and H.Gillis(4), but it has one more resistive element in parallel to the gas compliance.

$$\begin{gathered}
Rtube=(8*\mu_{gas}*L)/(\pi*(Radius^{\wedge}4)) \\
Itube=(\rho_{gas}*L)/(\ \pi*(Radius^{\wedge}2)) \\
Z1=RTube+j*\ \omega*ITube \\
ADB=Gt+j*\ \omega*Cg \\
ADB=1./ADB \quad \text{(it's a compliance)} \\
ADB=(ADB.*Zwall)/(ADB+Zwall) \\
Z(n)=Z1+((Z1+r).*ADB)/(Z1+ADB+r)
\end{gathered}$$

In order to take into account the tissue characteristics, the constants used in CalcImp.m are given by different functions. The airway wall is divided in the outer wall area (adventitia), the layer of smooth muscle and the submucosa. **WallType.m** is a function in which the values describing the airway walls can be switched to different conditions: healthy, COPD, asthmatic, fatal asthmatic. This function returns a value for the wall component, referring to the subdivision represented in Fig.7. **Cartilage.m** is the function which provides a result for the fractional cartilage content, according to constants found in literature (11). **Hwalls.m** computes the values of resistance, inertance and compliance of the airway walls for human lung using diameter, wall thickness and wall tissue material properties using the equations provided by literature (10).

For all terminal airways, the function **CalcImp2.m** finds the complex value of impedance, as for non-rigid tubes:

$$Rtube=(8*\mu_{gas}*L)/(\pi*(Radius\hat{}4))$$
$$Itube=(\rho_{gas}*L)/(\ \pi*(Radius\hat{}2))$$
$$Ztube=Rtube+j\ \omega Itube$$

The process needs only the geometric dimensions of the branch, diameter and length, and the gas density and viscosity.

CalcAlveoli.m is the function that computes the value of the alveoli impedance. The viscoelastic contribute of the alveolus must be added to the impedance of all the terminal branches.

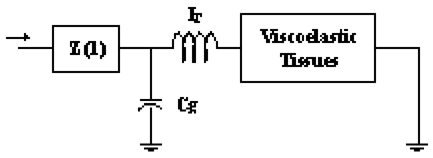


Figure 8. Alveolar Tissue Element. (4)

The equations used in **CalcAlveoli.m,** according to the model in Figure 8, are

G_tiss=G_tiss*NTR
H_tiss=H_tiss*NTR
I_tiss=I_tiss*NTR

Ztissue=(G_tiss-H_tiss)/(w^(alpha))

$$Zalveoli = \frac{(j\omega *It+Ztissue /(j\omega*Cg))}{(j\omega*It+Ztissue + (1/(j\omega*Cg)}$$

The alveoli are in parallel with each other. The airway tree can be considered as a complex network of impedances. At the end of the network (or attached to the terminal branches) are alveolar tissue impedances (Zissue). Each of these Ztissues are at one end connected to the terminal branch and at the other “connected to ground”. This ground can be considered a common ground shared by all of the Ztissue impedances. As mentioned earlier, the other end of the alveoli is connected to the terminal airway branches. Combining all the airways together into one whole network, one end of each of the alveoli elements would be connected together and the other ends will be connected together to the common ground. This is the reason the alveoli are considered connected in parallel with each other. Each of the pairs of terminal branches are parallel with each other and then there is the parallel combination of all of the combined terminal branches and their impedance. Because of this peculiar geometry of the network, in the function **CalcAlveoli.m** the number NTR is actually multiplying the values of tissue damping(G), tissue resistance(H) and tissue inertance (I).

CONSTRICTING THE AIRWAY TREE

Choosing the option “constrict” from the menu, the function **constrict.m** lets the operator impose the parameters for a uniform constriction, which can be applied to all diameters or to the diameters of certain generations only.

The program multiplies the diameters of the original tree by the coefficient given by the operator. There are three different options: the constriction might be applied to all the generations, to some specified generations only, or to all peripheral branches, which means to all branches with a diameter less than 2 mm. The diameters of the chosen generations are multiplied by the desired coefficient, which is constant for each generation.

It is possible to save the constricted tree or to cancel it. **Savetree.m** allows to save the modified tree in the new file "constricted.mat". Such file will appear in the current directory.

EXEMPLES

Running the program with the parameters of a healthy subject, the results are:

frequency	**impedance healthy**
2	2.386891 - 0.506142i
4	2.329703 - 0.120315i
6	2.311309 + 0.088986i
8	2.305246 + 0.250475i
10	2.304704 + 0.390999i
12	2.306594 + 0.520563i
14	2.309679 + 0.644327i
16	2.313688 + 0.765279i
18	2.318636 + 0.885400i
20	2.325298 + 1.007196i

In order to simulate the airway impedance in COPD patients the operator shall set the wall properties to COPD conditions. For simulating the typical constriction in COPD subjects, the generations to choose shall correspond to peripheral branches, i.e. those branches with diameters less than 2 mm (9). The tree considered in this project has to 28 generations, and therefore the constriction will be applied to generation #10 and up.

To perform this simulation the program was run twice. First, the source standard tree was loaded and constricted, applying 35% of diameter constriction, -which means that it was selected the option of constricting the peripheral branches, and the given coefficient was 0.65- and saved. Then, the program was run again using the newly created “constricted.mat” file as source file, and choosing the option for computing the impedance:

frequency	**impedance COPD with 35% constriction in gen.10-28**
2	4.1292 - 0.5399i
4	4.0139 - 0.0297i
6	3.9762 + 0.3012i
8	3.9687 + 0.5823i
10	3.9792 + 0.8423i
12	4.0032 + 1.0919i
14	4.0386 + 1.3366i
16	4.0845 + 1.5811i
18	4.1430 + 1.8334i
20	4.2354 + 2.1005i

Here follows the representation of the results of these simulations(Figure 8 and Figure 9): in the two graphs it is possible to compare the real and the imaginary part of the impedance at each value of frequency from 2 to 20 Hz.

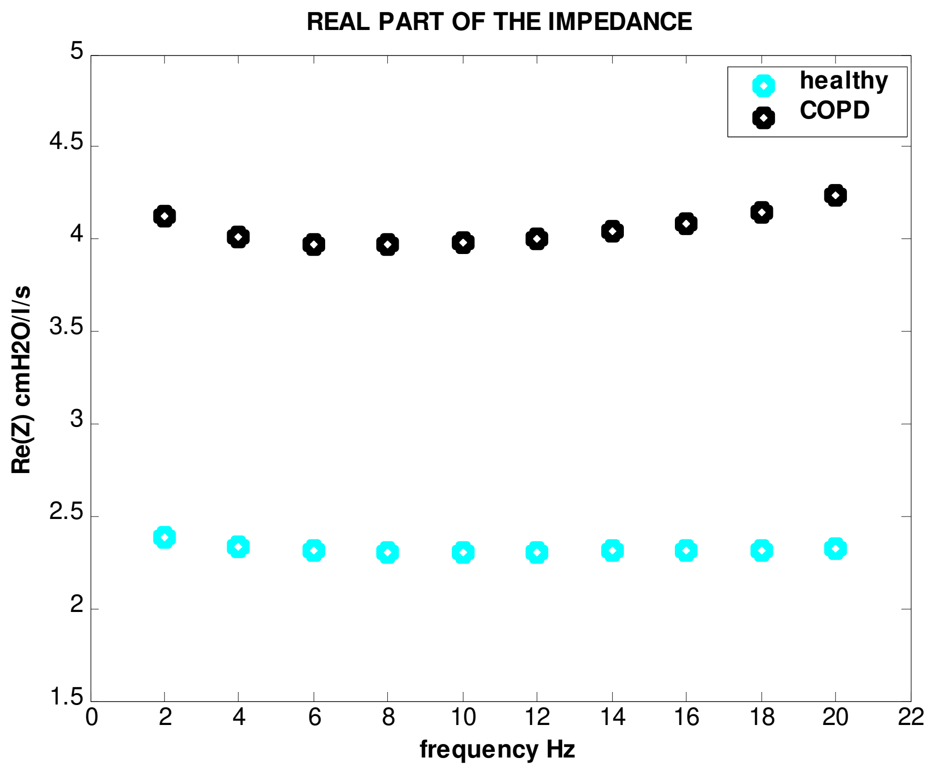


Figure 9
Simulation spectra of the real part of the impedance for different conditions: healthy and COPD constricted airways.

The real part of the impedance is rather constant through the frequency spectrum in the simulation of an healthy subject. In the simulation of a COPD patient, with pathologic wall conditions and constriction in the entire lung periphery, the real part of the impedance is higher at the considered frequencies, 2-20 Hz: these values mimic the phenomenon of the increase in the airways resistance. Moreover, the values representing COPD present frequency dependence.

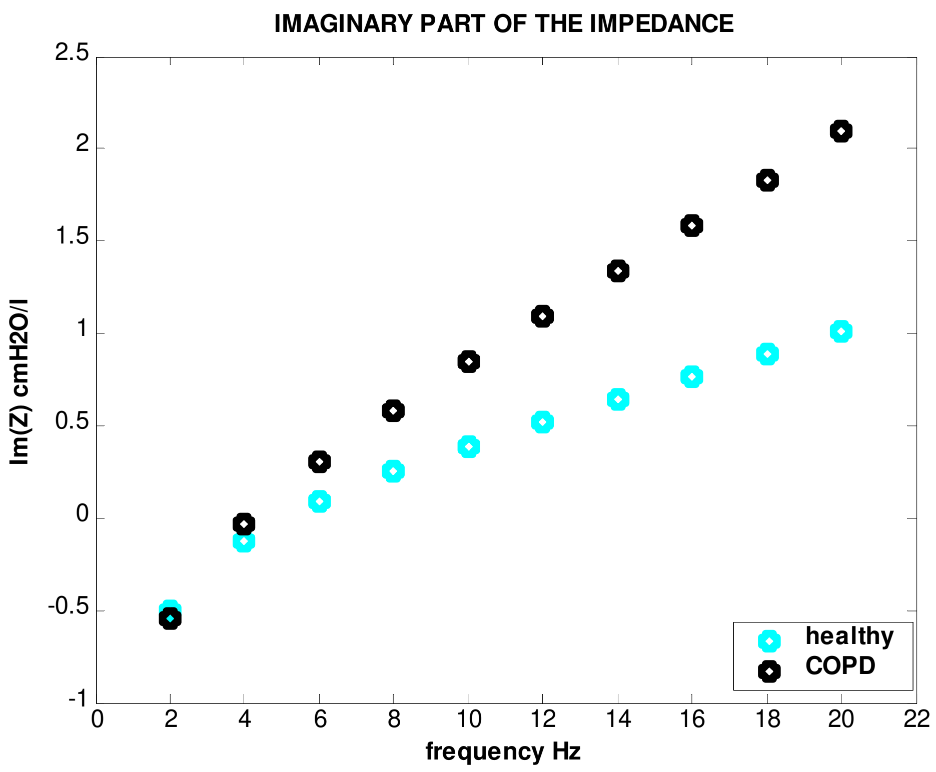


Figure 10 Simulation spectra of the imaginary part of the impedance for different conditions: healthy and COPD constricted airways.

The imaginary part of the impedance is frequency dependent and its value increase with frequency, both in the healthy and the COPD subjects in these simulations. At the breathing frequencies (<4Hz) the values are very similar. The difference between normal and COPD appears only at higher frequencies, as the disease conditions result in greater values. These simulations are consistent with the thesis in(5).

LIST OF THE IMPLEMENTED PROCEDURES

[ans]=program( )
offers a menu of options to apply to the matrix contained in a source file:
a function allows is to calculate the impedance of a tree at different frequencies /here the frequency array is set/;another function applies a constriction pattern to the tree
INPUT: none
OUTPUT: "ans", the result of the chosen function, or a message of error
FUNCTIONS CALLED: source.m; what_to_do.m; impedanceR.m; CONSTRICT.m.

[tree]=source( )
asks the operator to specify the source file to use; the source file shall contain a tree matrix
INPUT: none
OUTPUT: "tree", the tree matrix provided from load_tree.m, or the message of error of the called function
FUNCTIONS CALLED: load_tree.m

[mtree]=load_tree( )
asks to insert the name of the tree file.mat through a box dialog
opens the given file
if it is impossible it asks again for the file name or lets the operator leave the program prints the given file name with a message in the workspace
loads the matrix "mtree" of 5 columns (gen flow diam len node)
INPUT: none
OUTPUT: "mtree", the matrix contained in the file, or –1
FUNCTIONS CALLED: none

[operation]=what_to_do( )
allows the operator to pick from a menu what to do with the airway tree
INPUT: none
OUTPUT: none
FUNCTIONS CALLED: none

[tree1]=CONSTRICT (tree)
constricts the tree multiplying by a constant ratio -set by the operator- the chosen generations: the whole tree, the specified generation only, or all peripheral branches (diameter<2mm)
INPUT: "tree", a five columns matrix representing a branching tree
OUTPUT: "tree1", a five columns matrix representing a branching tree
FUNCTIONS CALLED: savetree.m

[]=savetree(tree1)
save the matrix "tree1" in a new file.mat, called "constricted.m", which will appear in the current directory
INPUT: the matrix "tree1"
OUTPUT: none
FUNCTIONS CALLED: none

[tot]=impedanceR(tree,freq)
calculates the overall impedance of the airway tree "tree" for the value of frequency "freq"
INPUT: the matrix "tree", a five columns matrix representing an airway tree
OUTPUT: "tot", a complex value which is the computed impedance
FUNCTIONS CALLED:terminals.m;SCALE.m;CalcImp2.m;CalcAlveoli.m;parallelImp.m; seriesImp.m; RparallelImp.m;Calc_Zup.m.

[NTR]=terminals(tree)
finds the number of terminal airways in "tree"

INPUT: the matrix 5 column matrix "tree"
OUTPUT: "NTR", an integer
FUNCTIONS CALLED: none

[tree]=SCALE(mtree)
scales diameters and lengths of the airway tree: scales lengths to 10cm(trachea) and uses the same factor to scale diameters
INPUT: 5 column matrix "mtree", 3rd and 4th column must be strictly positive real numbers
OUTPUT: "tree", 5 column matrix
FUNCTIONS CALLED: FRC.m

[tree]=FRC(tree)
finds the dimension values at FRC(Functional Residual Capacity): diameters and lengths
INPUT: 5 column matrix "tree", 3rd and 4th columns must be strictly positive real numbers
OUTPUT: "tree", 5 column matrix
FUNCTIONS CALLED: none

[r]=CalcImp2(branch,freq)
calculates the complex value of the impedance for a single branch of the airway tree
INPUT: 5 elements horizontal array "branch", 3rd and 4th element must be strictly positive real numbers; "freq", frequency value
OUTPUT: "r", complex impedance
FUNCTIONS CALLED: none

[r]=CalcAlveoli(freq,NTR)
calculates the impedance of the alvoelar tissue;
INPUT: "freq", frequency value; "NTR", number of terminal airways
OUTPUT: "r", complex impedance
FUNCTIONS CALLED: none

[rp]=parallelImp(branchA,branchB,freq,NTR)
calculates the complex value of the impedance of a double ending terminal branches in parallel plus adds the alveoli tissue element
INPUT: 5 elements horizontal arrays: "branchA", "branchB", 3rd and 4th element must be strictly positive real numbers; "freq", frequency value; "NTR", number of terminal airways

OUTPUT: "rp", complex impedance
FUNCTIONS CALLED: CalcAlveoli.m

[rs]=seriesImp(r, branch, freq)
calculates the impedance of the given branch and adds it in series to the value r
INPUT: "r", complex impedance; 5 elements horizontal array "branch", 3rd and 4th element must be strictly positive real numbers; "freq", frequency value;
OUTPUT: "rs", complex impedance
FUNCTIONS CALLED: CalcImp.m

[Z]=Calc_Zup(freq,rs)
assigns a value to the upper airways resistance and inertance and returns the total value of the impedance: airway tree plus upper airways
INPUT: "freq", frequency value; "rs", complex impedance .
OUTPUT: "Z", complex impedance of the whole system
FUNCTIONS CALLED: none

[rp]=RparalleImp(r,q)
this function does the parallel of two given complex numbers and returns the result
INPUT: "r", complex impedance, "q", complex impedance.
OUTPUT: "rp", complex impedance

FUNCTIONS CALLED: none

[Ztemp]=CalcImp(branch,freq,r)
for all branches but terminal branches; the equations used refer to Fig.1 "Impedance of a single airway generation" (Kenneth R. Lutchen and Heater Gillis. Relationship between heterogeneous changes in airway morphometry and lung resistance and elastance. J.Appl.Phisiol. 161:1192-1201,1997.)
INPUT: 5 elements horizontal array "branch", 3rd and 4th element must be strictly positive real numbers; "freq", frequency value; "r", complex impedance.
OUTPUT: "Ztemp", complex impedance
FUNCTIONS CALLED: Walltype.m; Cartilage.m; Hwalls.m.

[Watot]=Walltype(Diam)
wall contribute to impedance in every non-terminal airway; the operator might switch the properties according to the state of the airways
INPUT: "Diam", strictly positive real number accounting for an airway transversal diameter;
OUTPUT: "Watot", complex impedance (wall contribute in the airway)
FUNCTIONS CALLED: none

[Cartil]=Cartilage(Diam)
calculates the fractional cartilage content, as in JAP 86(6) 2000:20012,1999
INPUT: "Diam", strictly positive real number accounting for an airway transversal diameter;
OUTPUT: "Cartil", complex impedance (cartilage contribute in the airway)
FUNCTIONS CALLED: none
[Zwall]=Hwalls(freq,Diam,Len,Watot,Cartil)
computes the total impedance of the wall of a given airway branch
INPUT: "freq", frequency value; "Diam", "Len", strictly positive real numbers accounting for an airway diameter and length; "Watot", complex impedance (wall contribute in the airway), "Cartil", complex impedance (cartilage contribute in the airway)
OUTPUT: "Zwall", complex impedance of the wall of the airway
FUNCTIONS CALLED: none

BIBLIOGRAPHY

1. H. Kitaoka and B.Suki branching design of the bronchial tree on a diameter-flow relationship. J.Appl.Phisiol. 82(3):968-976,1997.
2. N. T. Tgavalekos. Image Assisted Modeling of Lung Asthmatics. Master of Science Boston University 2003
3. N. T. Tgavalekos. Senior Project, Boston University, 2000
4. K.R. Lutchen and H. Gillis Relationship between heterogeneous changes in airway morphometry and lung resistance and elastance. J.Appl.Phisiol. 161:1192-1201,1997.
5. K.R. Lutchen, J.L. Greenstein and B.Suki How inhomogeneities and airway walls affect frequency dependence and separation of airway and tissue properties. J.Appl.Phisiol.80(5):1696-1707,1996.
6. K.R. Lutchen and H. Gillis J.Appl.Phisiol.,1999.
7. G. Dubini Fluidodinamica nelle vie aeree, Meccanica respiratoria. 85-95.
8. R. Dellacà Assessment of Respiratory Mechanics –tesi dottorato, chapter 2
9. M. Setta and G. Turato Alterazioni strutturali nelle diverse forme cliniche di BPCO (bronchite cronica, enfisema, asma bronchiale cronico). I Quaderni della BPCO. Anonymous. Anonymous. 3-23, 2003.
10. Habib, Suki, Bates and Jackson J.Appl.Phisiol. 1994
11. JAP 86(6) 2000:20012, 1999
12. R. Peslin and J. J. Fredberg Oscillation mechanics of the respiratory system. in Section 3: P. T. Macklem, The Respiratory System American Physiological Society, Bethesda, Maryland, USA. 145-177. 1986.
13. G. Ciaburro Manuale Matlab. http://ciaburro.it

TEXT OF THE IMPLEMENTED PROCEDURES

```
        function [Zup]=Calc_Zup(freq,r);
        %this function assigns a value to the upper airways resistance and
inertance
        %and returns the total value of the impedance: airway tree plus upper
airways
        w=[];
        Z1=[];
        w=2*pi*freq;
        R=2.000;%cmH2O/l
        I=0.000;%cmH2O/l*s
        Z1=R+j.*w.*I;
        Zup=Z1+r;
        %end of function


        function [Z_tiss]=CalcAlveoli(freq, NTR);
        %calculates the impedence of the alvoelar tissue; returns a complex number
        VFRC=3.0;%liters
        w=2*pi*freq;%
        s=j*w;
        G=1.22;%cmH2O*l^(-1)*s
        H=5.8;%cmH2O*l^(-1)*s
        I=0.00069;%cmH2O*l^(-1)*s^2
        alpha=(2.0/pi)*atan(H/G);% the argument of the trig function has only the
real part
        G_tiss=G.*NTR;
        H_tiss=j.*H.*NTR;
        I_tiss=I.*NTR;
        Comp=VFRC./(1033*NTR); %1 atm= 10.33 mH20 = 1033cmH2O
        Z=(G_tiss-H_tiss)./(w^(alpha));
        Z=s.*I_tiss+Z;
        Z_tiss=(Z/(s.*Comp))/(Z + (1/(s.*Comp)));
        %end of function


        function [Ztemp]=CalcImp(branch,freq,r);
        %for all branches but terminal branches
        %the eqations used refer to Fig.1 "Impedance of a single airway generation"
        %(Kenneth R. Lutchen and Heater Gillis.J.Appl.Phisiol. 161:1192-1201,1997.)
        %constants
        Mu=(1.739*(10^(-4))); %Nora's value
        GasRho=(1.1316*(10^(-3))); %Nora's value
        x=2.0;
        %variables
        Diam=[];
        Len=[];
        Radius=[];
        w=[];
        RTube=[];
        ITube=[];
        ZTube=[];
        Cg=[];
        Gt=[];
        ADB=[];
        Watot=[];
        Cartil=[];
        Zwall=[];
        Diam=branch(1,3);
        Len=branch(1,4);
        w=2*pi*freq;
        Radius=Diam/2;
        RTube=(8*Mu.*Len)./(pi.*(Radius^4));
        ITube=(GasRho.*Len)/(pi.*(Radius^2));
        RTube=RTube./x;
        ITube=ITube./x;
        Z1=RTube+j.*w.*ITube;
        Cg=pi.*(Radius^2).*Len/1033000;
        Gt=4.6768387*(10^(-15));
        ADB=Gt+j.*w.*Cg;
        ADB=1./ADB;
        Watot=WallType(Diam);%properties from literature
        Cartil=Cartilage(Diam);%properties from literature
        Zwall=Hwalls(freq,Diam,Len,Watot,Cartil);
        ADB=(ADB.*Zwall)/(ADB+Zwall);
        Ztemp=Z1+((Z1+r).*ADB)/(Z1+ADB+r);
        %end of function
        function [Z1]=CalcImp2(branch,freq);
        %calculates the complex value of the impedance
        %for a single branch of the airway tree
        %Mu=1.90*(10^(-5));%value of the air dinamic viscosity at 1 atm,
        %310 K(body temperature) expressed in kg/(m*s)
        %kg/(m*s) => N/(s*m^2) => Pa*s => ((10.33*100)/101325)cmH2O*s
        %Mu=0.018791679*(10^(-5)); %Irene's value %cmH2O*s
        Mu=(1.739*(10^(-4))); %Nora's value
        %GasRho=1.139;%value of the air density at 1 atm,
        %310 K(body temperature) expressed in kg/(m^3)
        %GasRho=1.139; %or 1.16 % g/l
        GasRho=(1.1316*(10^(-3))); %Nora's value
        RTube=[];
        ITube=[];
        Cg=[];
        x=[];
        %each branch is a row of the tree matrix and the data in the 5 columns are:
        %(1)generation, (2)flow, (3)diameter, (4)lenght, (5)node
        Diam=[];
        Len =[];
        Radius=[];
        Diam = branch(1,3);
        Len = branch(1,4);
        Radius=Diam/2;% all length are in cm, scaled for a trachea of 10 cm
        RTube=(8*Mu.*Len)/(pi.*(Radius^4));%value of the tube resistance
        %by Poiseuille's law  R=(8*(Mu)*L)/(pi*((R)^4))
        ITube=(GasRho.*Len)/(pi.*(Radius^2));%cmH2O*(l^(-1))*(s^2)
        x=1;
        RTube=RTube./1;
        ITube=ITube./1;
        Z1=RTube+j.*2*pi.*freq.*ITube;
        %end of function


        function [cartil]=Cartilage(Diam);
        % Fractional Cartilage content, reference JAP 86(6)2000:20012,1999
           yo=-0.0314;
           a=0.2556;
           b=0.1368;
         cartil=yo+a.*Diam+b.*(Diam^2);
        %end of function


        function [tree1]=CONSTRICT(tree,freq);
        %constricts the tree multiplying by a constant ratio the chosen generations
or the whole tree
        s=length(tree);
        tree1=tree;%copy the tree
        c=[];%constriction rate
        %c=0.6; %fixed constriction rate
        choice=questdlg('operations','choose   what   to   do',   'constrict   all
tree','constrict generation','constrict peripheral','constrict all tree');
        if isequal(choice,'constrict all tree');
            p_info=inputdlg({'COSTRICTION RATE'});
            c=str2num(p_info{1});
            size(c)
            %tree1
             tree1(:,3)=tree1(:,3).*c;%constriction applied to all the diameters
            elseif isequal(choice,'constrict generation');
            p_info=inputdlg({'GENERATION','COSTRICTION RATE',});
            GEN=str2num(p_info{1});
            c=str2num(p_info{2});
            size(GEN)
            size(c)
               for i=1:s;
                if tree(i,1)==GEN;
```

```
                    tree1(i,3)=tree1(i,3).*c;
                     %constriction applied to the diameters where the gen# is the
specified one
               end
           end
           c
         elseif isequal(choice,'constrict peripheral');
           tree1=tree;
           p_info=inputdlg({'COSTRICTION RATE'});
           c=str2num(p_info{1});
             for i=1:s;
                 if (tree(i,3)<=0.2);%I consider 2 mm as the maximum diameter for
peripheral branches/The choice is done on FRC values
                         tree1(i,3)=tree1(i,3).*c;% same constriction applied the
diameters
               end
           end
       else
           'did not constrict'
       end
       choice=questdlg('operations','SAVE?','save   constricted   tree','do   not
save','save constricted tree' );
       if isequal(choice,'save constricted tree');
            savetree(tree1); %call to function that saves in the current directory
a new file containing tree1 as 'mtree'
       end
       %  end of function


       function [tree]=FRC(tree);
       %finds the values at FRC(Functional Residual Capacity)
       DALPHA=1.56;
       DBETA=0.354;
       DP=0.217;
       LFAC=1.13;
       DFAC=[];
       Diam=[];
       Len=[];
       for(i=1:length(tree));
           Diam=tree(i,3);
           Len=tree(i,4);
           DFAC=DALPHA/( 1+((Diam./DP)^(-1.*DBETA)) );
           tree(i,3)=tree(i,3).*DFAC; %scale all diameters
           tree(i,4)=tree(i,4).*LFAC; %scale all lenghts
           Diam=[];%
           Len=[];
       end
       %end of function


       function [EFFZWALL]=HWalls( freq,Diam,Len,Watot,Cartil);
       %returns the total impedance
       %of the wall of a given branch
       w=[];
       s=[];
       HWALL=[];
       SOFTRWALL=[];%soft tissue resistance
       CARTRWALL=[];%cartilage resistance
       SOFTIWALL=[];%soft tissue inertance
       CARTIWALL=[];%cartilage inertance
       SOFTCWALL=[];%soft tissue compliance
       CARTCWALL=[];%cartilage compliance
       SOFTZWALL=[];%soft tissue impedance
       CARTZWALL=[];%cartilage impedance
       w=2.0*pi*freq;
       s=j.*w;
       HWALL=(((Diam^2)/4.00+Watot./pi)^(.5))-Diam/2;
             %Compute Rw,Iw,Cw values for the human lung using diameter, wall
thicknesses
            %and wall tissue material properties using the equations provided by
Habib,
           %Suki,Bates adn Jackson (1994)
```

```
            %Serial Distribution of airway mechanical properties in dogs:effects of
histamine
        SOFTRHO= 1.06;
        CARTRHO= 1.14;
        SOFTVIS= .88;
        CARTVIS= 58.6;
        SOFTYMOD= 600.0;
        CARTYMOD= 40000.0;
                  SOFTRWALL=(4.0.*HWALL*SOFTVIS*1000.0)/(pi*(Diam^3)*Len);
                  CARTRWALL=(4.0.*HWALL*CARTVIS*1000.0)/(pi*(Diam^3)*Len);
                  SOFTIWALL=(HWALL.*SOFTRHO)/(pi*Diam*Len) ;
                  CARTIWALL=(HWALL.*CARTRHO)/(pi*Diam*Len);
                  SOFTCWALL=(pi.*(Diam^3).*Len)/(4.0.*HWALL.*SOFTYMOD*1000);
                  CARTCWALL=(pi.*(Diam^3).*Len)/(4.0.*HWALL.*CARTYMOD*1000);
               SOFTZWALL=SOFTRWALL+s.*SOFTIWALL+1/(s.*SOFTCWALL);
               CARTZWALL=CARTRWALL+s.*CARTIWALL+1/(s.*CARTCWALL);
                              EFFZWALL=(CARTZWALL*SOFTZWALL)/(Cartil*SOFTZWALL+(1.0-
Cartil)*CARTZWALL);
        %end of function


        function [rtot]=impedanceR(mtree, freq);
                              %calculates the ovrall impedance of the airways
                                           %
                                           %accessory functions :
                                           %
                                           %terminals.m
                                           %SCALE.m >> FRC.m
                                           %CalcImp2.m
                                           %CalcAlveoli.m
                                           %parallelImp.m  >> CalcAlveoli.m
                                           %               >> CalcImp2.m
                                           %seriesImp.m >> CalcImp.m >> WallType.m
                                                  %                           >>
Cartilage.m
                                           %                         >> Hwalls.m
                                           %RparallelImp.m
                                           %Calc_Zup.
        s=[];%number_of_airways
        awstack=[];
        fstack=[];
        h=0; %h is the index of the awstack
        %(this is a dynamic stack in which are recorded the data of the branches
before finding an ending point)
        f=0; %f is the index of the fstack(this is a dynamic stack in which are
recorded values of impedence)
        %each row of  matrix "tree" represents a branch;  in the first column there
is the gen# of the branch
        branchk=[];
        branchk_1=[];
        branch_temp=[];
        rp=[];%variable for the solution of each parallel
        rs=[];%variable for the solution of each series
        %temporary values
        rs_temp=[];
        Z_temp=[];
        r_temp=[];
        s=length(mtree);
        i=[]; %i is the index used to get the branches of the airway tree from the
tree matrix;
        NTR= terminals(mtree);% finds the number of terminal airways
        tree=SCALE(mtree);%scales lengths and diameters
        branchk_1=tree(1,:);
        %the trachea is the first branch of the airway tree and here becomes the
first elem. of branchk_1
        i=1;
        debug=[];
        while(i<=s+1);%if(i<=s); %this condition shall always be true during the
program
        %it means that the index must be less or equal to "treesize-1"
        %...moved up because the cycle starts at i=1 instead of i=0(Nora's)
        %outer for cycle, reads the source matrix (whos size is s)and stores the
branches in awstack(dyanamic stack)
```

```
       %the first branch(trachea) is already taken
               i=i+1;
               i
               branchk=tree(i,:);%getting the next segment
           if (branchk(1,1) > branchk_1(1,1));%if this is not an ending branch
       %push branchk_1 in awstack
                awstack = [awstack; branchk_1(1,:)]; %save the previuos branch in
the stack
               h=length(awstack(:,1));
               branchk_1(1,:)=branchk(1,:);
       %the current branch becomes parental, ready check the next in row from the
tree source list
                   elseif( (i<s-1) & (tree((i+1),1) > branchk(1,1)));
       %if the tree is asymmetric and stops at branch (k-1)
       % if the gen# of the branch next to branchk is less than the gen# of
branchk
       %it means that branchk IS AN ENDING BRANCH
               rs_temp=CalcImp2(branchk_1 , freq);
       %gets the impedence of the branch according to the given frequency value
               debug=[debug;rs];
               Ztemp=CalcAlveoli(freq, NTR);
               debug=[debug;Ztemp];
               rs=rs_temp+Ztemp;
               %push rs in fstack
                fstack=[fstack; rs]; %saving the current value of impedence in the
fstack
               f=length(fstack(:,1)); %one more element in fstack
               %saving branch k-1
               awstack = [awstack; branchk_1(1,:)];
               h=length(awstack(:,1));
               %saving branch k
               awstack = [awstack; branchk(1,:)];
               h=length(awstack(:,1));
               branchk_1(1,:)=tree((i+1),:);
               %i
               i=i+1;
               i
               %'strapippo'
            else
               %calculate parallel
               rp=parallelImp(branchk_1, branchk, freq, NTR);
               whileflag=0;
                   while(h>=0 & wileflag==0);% if(h>=0);while awstack has some
element in it or none
                   %pop awstack
                   h=length(awstack(:,1));
                    branchk_1=awstack(h,:); %take the last branch from the top of
the awstack
                   h=h-1;
                   if(h>0)
                       awstack = awstack(1:h, :);%take off the last row
                   else
                       awstack=[];
                   end;
                   %end pop
                   %add in series to existing
                   rs=seriesImp(rp, branchk_1, freq);
       %add in series to the actual value of the overall impedence
                   debug=[debug;rs];
                   if (length(awstack)==0);
       %if this is the end of the tree awstack, add the impedence of the upper
airways
       %and return the total value rs
                       'THE END'
                       %awstack
                       %fstack
                       %rs
                       %debug
                       rtot=Calc_Zup(freq, rs);
                       return; %end of algorithm
                   end
                   %pop awstack
                   h=length(awstack(:,1));
```

```
                    branchk=awstack(h,:); %take the last branch from the top of the
awstack
                    h=h-1;
                    if(h>0)
                        awstack = awstack(1:h, :);%take off the last row
                    else
                        awstack=[];
                    end;
                    %end pop
                    if( branchk_1(1,1) > branchk(1,1) );
        %if it is not the end of the tree save these branches in awstack
                        %push branchk
                        awstack = [awstack; branchk(1,:)];%saving branch k
                        h=length(awstack);%one more element in  the stack
                        %end push
                        %push branchk_1
                        awstack = [awstack; branchk_1(1,:)];%saving branch k-1
                        h=length(awstack(:,1));%one more element in  the stack
                        %end push
                        %push rs
                        fstack=[fstack; rs]; %save the series combination
                        debug=[debug;rs];
                        f=length(fstack(:,1)); %one more element in fstack
                               if(  tree(i+1,1)==branchk_1(1,1)  &  (tree(i+1,1)+1)~=
tree(i+2,1) );
        %if the right branch terminates before the left
                            %pop
                            h=length(awstack(:,1));
                            branch_temp=awstack(h,:);
                            h=h-1;
                            if(h>0)
                                awstack = awstack(1:h, :);%take off the last row
                            else
                                awstack=[];
                            end;
                            %end pop
                            rs=seriesImp(rp,branch_temp,freq);
        %adding in series the impedence of branchk to the former value of impedence
                              branchk_1=tree(i+1,:);%get the next row from the tree
matrix
                            rs_temp=CalcImp2(branchk_1,freq);
                            Z_temp=CalcAlveoli(freq, NTR);
                             r_temp=rs_temp + Z_temp;%value of the impedence of the
final  branch
                            rp=RparallelImp(rs,r_temp);
        %doing the parallel of the two paths
        %(the right final path to the value of the impedence of the left path up to
here)
                     %'poppo'
                            %pop fstack
                            f=length(fstack(:,1));
                              rs=fstack(f,:); %take the last branch from the top of
the awstack
                            f=f-1;
                            if(f>0)
                                fstack = fstack(1:f, :);%take off the last row
                            else
                                fstack=[];
                            end;
                            %end pop
                            i=i+1;
                            continue;
                        else
                           branchk_1=tree(i+1,:); %take next from array
                           whileflag=1;
                           i=i+1;
                           i
                           continue;
                               %break%If BREAK is executed in an IF statement, it
terminates the statement at that point
                        end
                     else
                        %pop fstack
```

```
                       f=length(fstack(:,1));
                         r=fstack(f,:); %take the last branch from the top of the
awstack
                       f=f-1;
                       if(f>0)
                           fstack = fstack(1:f, :);%take off the last row
                       else
                           fstack=[];
                       end;
                       %end pop
                       rp=RparallelImp(rs,r);
                       %debug=[debug;rp];
                    end%end of if/elseif/else
               end%(h>=0)...while awstack has some element in it or none
           end%if/elseif/else
       end %while(i<=s)...while reading the source tree
       rtot=-1;
       %if the program gets to this point it means an error occurred
       %as the program didn't get till the end of the tree
       %end of function


       function  [mtree]= load_tree; %this function:
                                      %ASKS to insert the name of the tree file.mat
through a box dialog
                                    %OPENS the given file
                                    %        IF IT'S IMPOSSIBLE
                                    %        asks again for the file name
                                    %              or
                                      %         lets the operator leave the program
and
                                       %         PRINTS the given file name with a
message in the workspace
                                       %LOADS the matrix "mtree" of 5 columns (gen
flow diam len node)
                                           %RETURNS the matrix containing the tree
data :-)
       clear all
       p_info=inputdlg({' file.mat containing "mtree": '});
       filename=char(p_info{1});
       filename
       source=fopen(filename,'r');
       if source~=-1;%mtree exists
          load(filename);
          s=length(mtree);
          s
           airway_characteristics=['generation    ' 'flow    ' '   diameter    ' '
lenght   ' '   node'];
       else  %%source==-1...  it's  impossible  to  open  'filename'  given  by  the
operator
           type_again=questdlg('type again the name of the file','the file name is
uncorrect');
           if isequal(type_again,'Yes');
               load_tree
           else
               'CANNOT OPEN THIS FILE'
           end
       end
       %end of function


       function [rp]= parallelImp(branchA, branchB, freq, NTR);
       %calculates the complex value of the impedence of a "double ending"
       %2 terminal branches in parallel plus the alveoli tissue element
       Ztemp=[];
       r=[];%impedance branchA
       q=[];%impedance branchB
       %adds the alveoli tissue element
       Ztemp=CalcAlveoli(freq, NTR);
       r=CalcImp2(branchA, freq)+Ztemp;
       q=CalcImp2(branchB, freq)+Ztemp;
       %does the parallel
       rp=(r*q/(r+q));
```

```
        %end of function


        function [ans]=program;
        %this program offers a menu of options to apply to matrix conatained in a
source file
        %a function allows is to calculate the impedance of a tree at different
frequencies
        %another function applys a constriction pattern to the tree
        %first:  identify the source file
        tree=source;
        frequencies=[2];%; 4; 6; 8; 10; 12; 14; 16; 18; 20 ];
        %frequency array; standard frequencies at which calculate the impedance
        Z=[]; %impedences array in which every value will correspond to a frequency
of the frequency array
        k=0; %index of the impedance array
        operation=what_to_do;%menu
            if isequal(operation,'calculate impedance');
                  m=length( frequencies );%number of repetitions of the impedance
routine
                for i=1:m;%I want to get a value of impedance for each frequency of
the array
                    freq=frequencies(i,1);
                    freq;
                    format long
                    Z(i,1)= impedanceR(tree,freq);
                end
                      ans=[frequencies  Z];%dispaly  the  frequency  array  and  the
corresponding values of impedance
            end %of the case 'calculate impedance'
            if isequal(operation,'constrict');
                %constrict
                tree1=CONSTRICT(tree);
                ans=tree1;
            end
            if isequal(operation,'plot'); %unused
                'PLOT UNUSED'
            end
        %end of function
        %end of program


         function [rp]=RparallelImp(r,q);
        % this function does the parallel of two given complex numbers and returns
the result
        rp=(r*q)/(r+q);
        %end of function


        function []=savetree(tree1);
        %function which allows the operator to save the matrix in a new file.mat
               mtree=[];
               mtree=tree1;
               %name=inputdlg({'new file name'});
        %I cannot make it to give the new file the name given by the operator!
               save constricted mtree;
         %end of function


        function [tree]=SCALE(mtree); %Scales diameters and lengths
        len_factor=10/mtree(1,4); %scales length to 10cm(trachea)
        diam_factor=len_factor; %use same factor to scale diameters
        tree=mtree;
        for(i=1:length(tree));
            tree(i,4)=tree(i,4).*len_factor;%scale all lenghts
            if(tree(i,4)==0.000)  %if length is equal to zero ...does it happen??
                tree(i,4)=.000018;
            end
            tree(i,3)=tree(i,3).*diam_factor;%scale all diameters
        end
        tree=FRC(tree);  %Scale all values at FRC
        tree;
        %end of function
```

```
       function [Ztemp]=seriesImp(r,branch,freq);
       %calculates the impedance of the given branch and adds it in series to the
value r
       Ztemp=CalcImp(branch,freq,r);
       %end of function
       function[sourcefile]=source; %this function asks the operator to specify
the source of the tree to use
       clear all
       choice=questdlg('load  source  file  or  create  a  new  tree','source  file
','load from file','create new tree','load from file');
           if isequal(choice,'load from file');
               sourcefile=load_tree;
                   else
               'create new tree'
           end
       return
       %end of function


       function [NTR]=terminals(tree); %finds the number of terminal airways
       s=length(tree); %tree dimension
       n=0;%number of terminals
       f=0;
       for (i=1:(s-1));
           if (tree(i,5) < tree(i+1,5)) ;
       %condition for no terminal branches, according to the node indicator in the
5th column
               f=f+1; %increase the number of non-ending  branches
           elseif ( tree(i+1,1)==(tree(i,1)+1) );
       %condition for no terminal branches, according to the generation indicator
in the 1st column
              f=f+1; %increase the number of non-ending  branches
           else
               n=n+1; %increase the Number of Terminals branches NTR
           end
       end
       f; %number of non-terminal branches
       NTR=n; %number of terminal branches
       %end of function


       function [WAtot]=WallType(Diam); % wall contribute to impedance in every
non-terminal airway;
       the operator might switch the properties according to the state of the
airways
            %HEALTHY Wall Conditions
            %s1c=.079;
            %s2c=.020;
            %s3c=.052;
            %i1c=.116;
            %i2c=.056;
            %i3c=.067;
           %COPD Wall conditions
           s1c=.086;
           i1c=.084;
           s2c=.035;
           i2c=.016;
           s3c=.056;
           i3c=.054;
          %Asthmatic wall conditions
          %s1c=.137;
          %i1c=.085;
          %s2c=.063;
          %i2c=.001;
          %s3c=.099;
          %i3c=.009;
         %Fatal Asthmatic WALL CONDITIONS
         %s1c=.167;
         %i1c=.070;
         %s2c=.068;
         %i2c=.005;
         %s3c=.106;
         %i3c=.015;
         %variables
         pbm=[]; %basement membrane perimeter in mm
         WAo=[]; %outer wall area(adventia)
         WAsm=[];%ASM
         WAi=[]; %submucosa
         WAtot=[];
               pbm=pi.*Diam*10;   %basement membrane perimeter in mm
               WAo=.01*((s1c*pbm+i1c)^2);     %outer wall area(adventia)
               WAsm=.01*((s2c*pbm+i2c)^2);    %ASM
               WAi=.01*((s3c*pbm+i3c)^2);     %submucosa
               WAtot=WAo+WAsm+WAi;
       %end of function


       function [choice]=what_to_do; %this function allows the operator to pick
from a menu what to do with the airway tree
       clear all
       choice=questdlg('operations','choose what to do', 'calculate
impedance','constrict','plot','calculate impedance');
       %end of function
```