



**POLITECNICO**  
MILANO 1863

POLITECNICO DI MILANO  
SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE  
M. SC IN APPLIED STATISTICS

# Breathing patterns recognition: a Functional Data Analysis approach

*Supervisor:* Prof. Andrea Aliverti  
*Co-Supervisor:* Dr. Maria Antonella Lo Mauro

*Candidates:* Alessandra Colli  
matr. 892296  
Luca Colombo  
matr. 905175

Academic Year 2018-2019



# Acknowledgements

We would like to thank all the people who made this possible.

Foremost, we would like to thank our supervisors, professor Andrea Aliverti for his support and guidance throughout this thesis work, and Maria Antonella Lo Mauro for her precious assistance and her inspiring commitment.

We would like to thank professor Laura Maria Sangalli for her valuable advices, and professors Anna Maria Paganoni and Simone Vantini for the time they dedicated to us.

Thanks to Jack and Max, who were part of the Applied Statistics team were this project began. We gave our bodies to research together, now all our data are part of this work. We would also like to thank all the other friends who supported us throughout these years: Edo, Giulio, Teo and Toni.

Finally, a special thank to our families for their encouragement and their help, both moral and material. None of this could have been possible without your support.

# Sommario

Lo scopo di questo lavoro è applicare l'analisi di dati funzionali ai dati ottenuti mediante Pletismo-  
grafia Optoelettronica (OEP), una tecnica non invasiva per la misurazione della variazione di volume  
toracico nel tempo associata alla respirazione, sviluppata all'interno del Dipartimento di Elettronica  
Informazione e Bioingegneria del Politecnico di Milano. I dati acquisiti mediante questa tecnica sono  
caratterizzati da una elevata variabilità intra-soggetto e inter-soggetto, dipendente sia da fattori  
fisiologici che da fattori esogeni quali cambiamento di postura o artefatti di misura. La loro analisi  
attualmente richiede scelte arbitrarie da parte dell'operatore, come il posizionamento dei minimi  
del tracciato, l'eliminazione di respiri outlier e l'individuazione di gruppi all'interno dell'attività  
respiratoria. Viene dunque proposta una innovativa procedura semi-automatica e riproducibile in  
R basata su tecniche di analisi di dati funzionali, robusta rispetto a dataset non banali, in grado  
di caratterizzare in senso funzionale il breathing pattern di un soggetto. La procedura sviluppata  
consente di ottenere risultati inediti, quali l'estrazione su base statistica di un rappresentante fun-  
zionale del respiro del soggetto e di indagare l'eventuale presenza di cluster, sia dal punto di vista  
dei valori assoluti, sia dal punto di vista della forma. L'efficacia e la flessibilità della metodolo-  
gia proposta vengono mostrate attraverso l'applicazione a una serie di casi clinici. Infine, viene  
mostrato come l'estrazione della curva caratteristica permetta un'analisi comparativa del breathing  
pattern tra gruppi di soggetti con diverse caratteristiche, attraverso una applicazione al confronto  
di soggetti sani a diverse età e posture, e pazienti affetti da Distrofia Muscolare di Duchenne in  
diverse fasi della malattia.



# Abstract

The aim of this work is to apply Functional Data Analysis to data acquired by means of Optoelectronic Plethysmography (OEP), a technique allowing for a non-invasive chest wall volume variation measurement during respiration, developed within the Dipartimento di Elettronica Informazione e Bioingegneria at Politecnico di Milano. The OEP data are characterized by a high intra-subject and inter-subject variability, due to both physiological factors and exogenous factors like postural changes and/or measurement noise. The state-of-art analysis requires operator-dependent choices such as minima positioning in the data track, elimination of outlier breaths and the individuation of breath clusters in the data. An innovative semi-automatic and reproducible procedure in R is proposed, based on Functional Data Analysis techniques, which allows for the characterization of a subject's breathing pattern in a functional sense, and is robust with respect to non-trivial datasets. This procedure allows to achieve new results such as the extraction on a statistical basis of a functional representative of a subject's breath, or the statistical individuation of clusters in the breathing pattern with respect to absolute values or shape. Flexibility and effectiveness of the procedure are validated through its application to real case studies. Finally it is shown how the extraction of a representative breath curve allows for an inter-subject breathing pattern comparison, with an application to the comparison of healthy subjects in different postures and age classes, and patients suffering from Duchenne Muscular Dystrophy in different disease stages.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>The respiratory system and breathing mechanics</b>	<b>3</b>
2.1	The Muscles of Respiration and the Chest Wall . . . . .	4
2.1.1	Inspiratory muscles . . . . .	5
2.1.2	Expiratory muscles . . . . .	7
2.2	Optoelectronic plethysmography . . . . .	7
2.3	State of the art . . . . .	10
<b>3</b>	<b>Functional Data Analysis</b>	<b>12</b>
3.1	Smoothing . . . . .	12
3.2	Functional Outlier Detection . . . . .	16
3.2.1	Depth measures for functional data . . . . .	16
3.2.2	Magnitude functional outlier detection . . . . .	18
3.2.3	Shape functional outlier detection . . . . .	18
3.3	Curve registration and clustering . . . . .	19
3.3.1	Mathematical framework . . . . .	20
3.3.2	K-means with alignment . . . . .	22
<b>4</b>	<b>Plethysmography data preprocessing</b>	<b>24</b>
4.1	Breaths separation . . . . .	24
4.1.1	Algorithm steps . . . . .	24
4.2	Outlier detection . . . . .	31
4.2.1	Algorithm steps . . . . .	31
4.3	Validation of preprocessing techniques . . . . .	37
4.3.1	Minima Examples . . . . .	37
4.3.2	Outlier detection - performance . . . . .	37
4.3.3	Outlier detection - Quiet Breathing with maneuvers . . . . .	44
4.3.4	Outlier detection - Quiet Breathing with noise . . . . .	47
<b>5</b>	<b>Breathing patterns identification</b>	<b>49</b>
5.1	Quiet breathing . . . . .	49
5.2	Exercise . . . . .	52
5.3	Mechanical ventilation . . . . .	54
5.4	Shape patterns during quiet breathing . . . . .	55

5.5	Case studies and applications . . . . .	57
5.5.1	Quiet breathing example . . . . .	57
5.5.2	Case report - Cystic Fibrosis . . . . .	59
5.5.3	Case report - Exercise . . . . .	65
5.5.4	Case report - Mechanical ventilation . . . . .	70
5.5.5	Case report - Duchenne . . . . .	75
<b>6</b>	<b>Population analyses</b>	<b>82</b>
6.1	Methodology . . . . .	82
6.2	Comparing different postures . . . . .	84
6.3	Comparing different ages - healthy sample . . . . .	87
6.3.1	Volume variations comparison . . . . .	87
6.3.2	Compartments relative contribution . . . . .	91
6.4	Comparing different ages - DMD sample . . . . .	95
6.4.1	Volume variations comparison . . . . .	95
6.4.2	Compartments relative contribution . . . . .	99
6.5	Comparing different clinical conditions . . . . .	103
6.5.1	Healthy children and DMD children . . . . .	103
6.5.2	Healthy adults and DMD adults . . . . .	110
6.6	Discussion . . . . .	117
<b>7</b>	<b>Conclusion and further developments</b>	<b>122</b>
<b>A</b>	<b>Code</b>	<b>127</b>
A.1	Prerequisites . . . . .	127
A.2	Subject analysis - procedure . . . . .	127
A.2.1	File <code>smooth_breath.R</code> . . . . .	135
A.2.2	File <code>localmin.R</code> . . . . .	137
A.2.3	File <code>outlier_detection.R</code> . . . . .	140
A.2.4	Auxiliary functions . . . . .	145
A.3	Inter-subjects comparison . . . . .	149
A.3.1	File <code>plot_compartments.R</code> . . . . .	152

# List of Figures

2.1	The major respiratory organs. . . . .	4
2.2	Three-compartment chest wall model of the respiratory system, where different pressures of interest are shown: $P_{ao}$ pressure at the airway opening, $P_{alv}$ alveolar pressure, $P_{pl}$ pleural pressure, $P_{ab}$ abdominal pressure, $P_{bs}$ pressure at the body surface. . . . .	5
2.3	Inspiration and expiration sequences. . . . .	6
2.4	Optoelectronic plethysmography (OEP): principle of measurement . . . . .	9
2.5	Geometrical models of the three chest wall compartments: pulmonary rib cage (RCp), abdominal rib cage (RCa), and abdomen (AB) (left, three views) and their volume changes during quiet spontaneous breathing, respectively, $V_{rcp}$ ; $V_{rca}$ ; and $V_{ab}$ . Chest wall volume ( $V_{cw}$ ) is equal to $V_{rcp} + V_{rca} + V_{ab}$ . . . . .	9
2.6	Acquired OEP data for a healthy subject in sitting position. . . . .	10
2.7	On the left, a normal spirogram. On the right, schematic diagram of the variations of VT during incremental exercise in healthy subjects. . . . .	11
3.1	The thirteen basis functions defining an order four spline with nine interior knots, shown as vertical dashed lines. . . . .	14
3.2	On the left panel an example of amplitude-only variation is shown; on the right an example of phase-only variation. . . . .	20
3.3	Possible choices of dissimilarity index and warping function class. . . . .	21
4.1	A real signal with problematic minima. . . . .	25
4.2	Quiet breathing acquisition with two vital capacities . . . . .	31
4.3	Minima examples. . . . .	38
4.5	Identified outliers over compartments. . . . .	40
4.6	Time outlier detection iterations. . . . .	41
4.7	Magnitude outlier detection iterations. . . . .	42
4.8	Shape outlier detection iterations. . . . .	42
4.9	Chest wall volume with minima. . . . .	44
4.10	Time and magnitude outlier detection. . . . .	44
4.11	Shape outlier detection. . . . .	45
4.12	Identified outliers over chest wall and compartments. . . . .	46
4.13	Quiet breathing example with measurement artifacts . . . . .	47
4.14	Identified outliers over compartments. . . . .	48
5.1	The main features of respiratory muscle action during exercise. . . . .	53
5.2	Quiet breathing example. . . . .	57

5.3	Outliers on total volume. . . . .	57
5.4	Output of kmap function. . . . .	58
5.5	Median breath components in comparison. . . . .	58
5.6	Breathing tracks - Chest wall and compartments . . . . .	60
5.7	Outlier detection results. . . . .	61
5.8	Multivariate kma results - $L^2$ distance . . . . .	61
5.9	Clustering on data volumes. . . . .	62
5.10	Clustered breaths . . . . .	63
5.11	Median breaths of clusters - $L^2$ distance . . . . .	63
5.12	Multivariate kma results - Pearson distance . . . . .	64
5.13	Clustering on total volume - Pearson distance . . . . .	64
5.14	Data acquisition during exercise. . . . .	66
5.15	Outlier detection results. . . . .	67
5.16	Multivariate kma results - $L^2$ distance . . . . .	67
5.17	Clustered breaths. . . . .	68
5.18	Cluster centroids in comparison. . . . .	68
5.19	Clustering on data volumes. . . . .	69
5.20	Breathing tracks - Chest wall and compartments . . . . .	71
5.21	Outlier detection results. . . . .	72
5.22	Multivariate kma results - $L^2$ distance . . . . .	72
5.23	Clustering on data volumes. . . . .	73
5.24	Clustered breaths . . . . .	74
5.25	Median breaths of clusters . . . . .	74
5.26	Breathing tracks - Chest wall and compartments . . . . .	76
5.27	Outlier detection results in the chest wall. . . . .	77
5.28	Multivariate kma results - $L^2$ distance . . . . .	77
5.29	Clustering on data volumes. . . . .	78
5.30	Clustered breaths . . . . .	79
5.31	Median breaths of clusters . . . . .	79
5.32	Multivariate kma results - Pearson distance . . . . .	80
5.33	Clustering on data volumes. . . . .	81
6.1	Normalized median breaths . . . . .	84
6.2	Inter-subject clustering . . . . .	85
6.3	Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified. . . . .	86
6.4	Median breaths - healthy sample . . . . .	88
6.5	Classificaton results . . . . .	89
6.6	Classification results in terms of volume variation: the left column shows the true labels, the right column shows how breaths are classified. . . . .	90
6.7	Normalized median breaths - healthy sample . . . . .	91
6.8	Classification result - compartment contribution . . . . .	92
6.9	Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified. . . . .	94
6.10	Median breaths - DMD sample . . . . .	96
6.11	Classification results - DMD sample . . . . .	97

6.12	Classification results without RCa - DMD sample . . . . .	97
6.13	Classified median breaths - DMD sample . . . . .	98
6.14	Normalized median breaths - DMD sample . . . . .	100
6.15	Classification results - Compartment contribution in DMD sample . . . . .	101
6.16	Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified. . . . .	102
6.17	Median volume variations - DMD and CTR young samples . . . . .	104
6.18	Classification results - DMD young and CTR young. . . . .	105
6.19	Classification results in terms of volume variation: the left column shows the true labels, the right column shows how breaths are classified. . . . .	106
6.20	Normalized median breaths - DMD and CTR young samples. . . . .	107
6.21	Classification results - DMD young and CTR young. . . . .	108
6.22	Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified. . . . .	109
6.23	Median volume variation - DMD and CTR adult samples. . . . .	111
6.24	Classification results - DMD adult and CTR adult. . . . .	112
6.25	Classification results in terms of volume variation: the left column shows the true labels, the right column shows how breaths are classified. . . . .	113
6.26	Normalized median breaths - DMD and CTR adult samples. . . . .	114
6.27	Classification results - DMD adult and CTR adult. . . . .	115
6.28	Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified. . . . .	116
6.29	Position comparison - supine and seated healthy subjects . . . . .	118
6.30	Age comparison - supine healthy subjects . . . . .	119
6.31	Age comparison - DMD subjects . . . . .	120
6.32	Clinic condition comparison - Healthy and DMD adult . . . . .	121

# List of Tables

4.1	Outlier detection results - perturbed track . . . . .	43
4.2	Outlier error rates in different configurations. . . . .	43
6.1	Confusion matrix for subject posture . . . . .	85
6.2	Confusion matrix for subject age . . . . .	87
6.3	Confusion matrix for subject age - compartment contribution . . . . .	92
6.4	New confusion matrix for subject age - compartment contribution . . . . .	93
6.5	Confusion matrix for subject age - DMD sample . . . . .	95
6.6	New confusion matrix for subject age - DMD sample . . . . .	96
6.7	Confusion matrix for subject age - Compartment contribution in DMD sample . . .	99
6.8	Confusion matrix for DMD and CTR adult samples - volume variation . . . . .	110
6.9	Confusion matrix for DMD and CTR adult samples - compartment contribution . .	115



# Chapter 1

## Introduction

Far from self-sustaining, the body depends on the external environment, both as a source of substances the body needs to survive and as a catch basin for its wastes. Cells making up the body require a continuous supply of oxygen to carry out their vital functions, and give off carbon dioxide, a waste product the body must get rid of. The major function of the respiratory system is to supply the body with oxygen and dispose of carbon dioxide. Pulmonary ventilation, better known as breathing, is the movement of air into and out of the lungs so that the gases there are continuously changed and refreshed [16].

Several methods to measure pulmonary ventilation are available. Optoelectronic plethysmography (OEP) is a now-established technique, based on typical methods for optical motion analysis, that allows to measure the variations of the volume of the chest wall and its compartments during breathing. This method, differently from traditional spirometry, provides a non-invasive measurement of a subject's pulmonary ventilation, and allows to study the contribution of the different muscles of respiration during breathing [3]. The muscles of respiration are essential parts of the respiratory system, since lungs are not capable of inflating themselves, and will expand only when there is an increase in the volume of the thoracic cavity. In humans, as in the other mammals, this is achieved primarily through the contraction of the diaphragm, but also by the contraction of the intercostal muscles which pull the rib cage upwards and outwards [22]. Thanks to OEP it is possible to study chest wall kinematics in the majority of conditions, including exercise [2] [4]. OEP data can be used to evaluate the breathing pattern of the subject under consideration, which can be summarized with a number of parameters such as the respiratory rate, the tidal volume, the end-expiratory lung volume. These quantities are then used in diagnostics, for example to assess the evolution of respiratory system diseases [4].

The OEP data consist in temporal series of volume measurements of the chest wall and its compartments. Single volume oscillations, taken with respect to the total chest wall volume, constitute the subject's breaths. Given a set of curves, each representing a breath of the subject, computation of breathing pattern parameters has been standardized and can be done almost automatically. However, a rigorous, automatic and reproducible procedure to analyse such data is lacking: state-of-art analysis of OEP data still involves operator-dependent choices, especially for what concerns cutting the data vector in single breaths, which is usually done by hand by the operator. Also, outlying curves are removed based on visual inspection, and the breathing pattern parameters are computed on a set of curves which is selected manually by the analyst among the whole dataset

[13] [14]. Moreover, in some applications (such as exercise) different breathing patterns may coexist in the same acquisition, that is, it is required to distinguish clusters among data [2].

Functional Data Analysis (FDA) provides statistical tools able to tackle these problems. In fact, OEP data can be interpreted as a sequence of realisations of a smooth underlying function, that is the breath function of the patient under consideration. A statistical procedure to analyse OEP data, from breaths separation to pattern recognition, is provided. B-spline smoothing [18], functional outlier detection based on depth measures [9] [10] and K-medoids with Alignment [19] techniques are combined and adapted to work with Optoelectronic Plethysmography data, in order to deal with a variety of clinical situations.

A definition of breath is given as the oscillation between two consecutive minima of the total chest wall volume. An algorithm, based on a pre-smoothing step, to find total volume minima is provided. The concept of outlying breath is formalized and an algorithm for outlier detection is given, which combines different outlier detection techniques. This procedure is semi-automatic and reproducible, and allows to work on the original data avoiding manual curves selection. Moreover, it is possible to provide a functional characterization of the breathing pattern of a subject. K-medoids with Alignment is employed to find clusters in the breath curves: the  $L^2$  distance is used in contexts where the amplitude or the mean volume trend in the data acquisition are significant, while the Pearson correlation coefficient is proposed when shape patterns in the data are investigated. Cluster centroids can be used for a synthetic description of the groups. Functional medians can also be extracted as representatives of the quiet breath of a subject, and used for population analysis. The proposed outlier detection algorithm can be used to study outliers in the medians, while K-medoids with Alignment is applied to study group structures among the patients. A number of examples and case studies show the effectiveness and the robustness of the developed procedure in analyzing the OEP data, which by their nature are characterized by a high variability, both intra-subject and inter-subject.

The thesis is organized as follows. In Chapter 2, basic concepts of respiratory system physiology and breathing mechanics are discussed, and a description of Optoelectronic Plethysmography is provided. Chapter 3 presents a Functional Data Analysis theoretical background, with a focus on the techniques that are employed to study OEP data. Chapter 4 describes a statistical methodology for OEP data preprocessing, from breaths separation to outlier detection. Examples of application are provided, with an experimental assessment of the outlier detection method. In Chapter 5 a method to find the representative breath function of a subject is discussed. K-medoids with Alignment algorithm is applied either to find the functional median breath of a subject, or to find possible cluster structures in his breathing pattern. Case studies in different clinical and experimental settings are discussed. In Chapter 6 methods and results of population analysis are presented. Functional medians are used to compare healthy subjects in sitting and supine position, and supine at different ages. A comparison between healthy subjects and patients affected by Duchenne Muscular Dystrophy at different ages is discussed. Finally, in the Appendix the complete R code for the analysis is reported in detail, with indications on the R packages needed to run the scripts.

## Chapter 2

# The respiratory system and breathing mechanics

In this chapter, we provide a brief description of the respiratory system and its functions.

The **respiratory system** includes the nose, nasal cavity, and paranasal sinuses; the pharynx; the larynx; the trachea; the bronchi and their smaller branches; the lungs, which contain the terminal air sacs, or alveoli; the parts of the central nervous system concerned with the control of the muscles of respiration, and the chest wall. The chest wall consists of the muscles of respiration — such as the diaphragm, the intercostal muscles, and the abdominal muscles — and the rib cage. In Figure 2.1 a representation of the main respiratory organs is provided [16].

The major function of the respiratory system is to supply the body with oxygen and dispose of carbon dioxide. Most of the tissues of the body require oxygen to produce energy, so a continuous supply of oxygen must be available for their normal functioning. Carbon dioxide is a by-product of this aerobic metabolism, and it must be removed from the vicinity of the metabolizing cells.

To accomplish this function, at least four processes, collectively called **respiration**, must happen [16]:

1. **Pulmonary ventilation:** movement of air into and out of the lungs so that the gases there are continuously changed and refreshed (commonly called breathing).
2. **External respiration :** movement of oxygen from the lungs to the blood and of carbon dioxide from the blood to the lungs.
3. **Transport of respiratory gases:** transport of oxygen from the lungs to the tissue cells of the body, and of carbon dioxide from the tissue cells to the lungs. This transport is accomplished by the cardiovascular system using blood as the transporting fluid.
4. **Internal respiration:** movement of oxygen from blood to the tissue cells and of carbon dioxide from tissue cells to blood.

The breathing of all vertebrates with lungs consists of repetitive cycles of inhalation and exhalation. **Inspiration** is the period when air flows into the lungs, and **expiration** is the period when gases exit the lungs. The number of respiratory cycles per minute is the breathing or respiratory rate, and is one of the four primary vital signs of life [22].

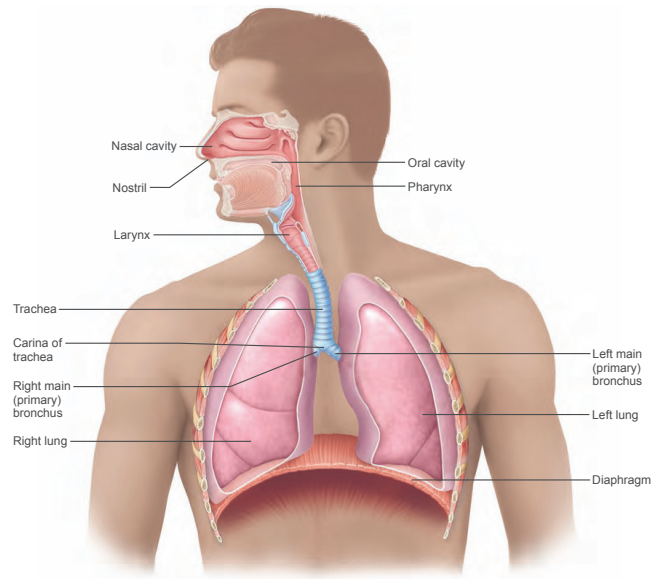


Figure 2.1: The major respiratory organs.

## 2.1 The Muscles of Respiration and the Chest Wall

The muscles of respiration and the chest wall are essential components of the respiratory system. The lungs are not capable of inflating themselves, and will expand only when there is an increase in the volume of the thoracic cavity. In humans, as in the other mammals, this is achieved primarily through the contraction of the diaphragm, but also by the contraction of the intercostal muscles which pull the rib cage upwards and outwards [22].

The primary components of the chest wall are shown schematically in Figure 2.2 [4]. These include the rib cage; the external and internal intercostal muscles and the diaphragm, which are the main muscles of respiration; and the lining of the chest wall, the visceral and parietal pleura.

The **intrapulmonary** (intra-alveolar) **pressure** ( $P_{alv}$ ) is the pressure in the alveoli. Intrapulmonary pressure rises and falls with the phases of breathing, but it always eventually equalizes with the atmospheric pressure [16] [12].

The pressure in the pleural cavity, the **intrapleural pressure** ( $P_{pl}$ ), also fluctuates with breathing phases, but is always about 4 mm Hg less than  $P_{alv}$ . That is,  $P_{pl}$  is always negative relative to  $P_{alv}$ .

Any condition that equalizes  $P_{pl}$  with the intrapulmonary (or atmospheric) pressure causes immediate lung collapse. It is the **transpulmonary pressure**—the difference between the intrapulmonary and intrapleural pressures ( $P_{alv} - P_{pl}$ )—that keeps the air spaces of the lungs open or, phrased another way, keeps the lungs from collapsing. Moreover, the size of the transpulmonary pressure determines the size of the lungs at any time [16].

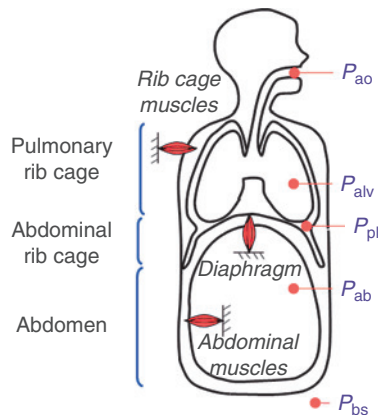


Figure 2.2: Three-compartment chest wall model of the respiratory system, where different pressures of interest are shown:  $P_{ao}$  pressure at the airway opening,  $P_{alv}$  alveolar pressure,  $P_{pl}$  pleural pressure,  $P_{ab}$  abdominal pressure,  $P_{bs}$  pressure at the body surface.

### 2.1.1 Inspiratory muscles

The muscles of inspiration include the diaphragm, the external intercostals, and the accessory muscles of inspiration [12]. As these muscles are activated, the thoracic dimensions increase, the lungs are stretched and the intrapulmonary volume increases, thereby decreasing the gas pressure inside it. This drop in pressure causes air to rush into the box from the atmosphere, because gases always flow down their pressure gradients [16].

#### The Diaphragm

The diaphragm is the primary muscle of inspiration. It is a large dome-shaped muscle (about  $250 \text{ cm}^2$  in surface area) that separates the thorax from the abdominal cavity. The diaphragm is considered to be an integral part of the chest wall and must always be considered in the analysis of chest wall mechanics. When a person is in the supine position, the diaphragm is responsible for about two thirds of the air that enters the lungs during normal quiet breathing [22]. When a person is standing or seated in an upright posture, the diaphragm is responsible for only about one third to one half of the tidal volume. It is innervated by the two phrenic nerves, which leave the spinal cord at the third through the fifth cervical segments [12].

During normal quiet breathing, contraction of the diaphragm causes its dome to descend 1 to 2 cm into the abdominal cavity, with little change in its shape. This elongates the thorax and increases its volume. During a deep inspiration, the diaphragm can descend as much as 10 cm. With such a deep inspiration, the limit of the compliance of the abdominal wall is reached. After this point, contraction of the diaphragm elevates the lower ribs. If one of the leaflets of the diaphragm is paralyzed (for example, because of transection of one of the phrenic nerves), it will “paradoxically” move up into the thorax as intrapleural pressure becomes more negative during a rapid inspiratory effort [12].

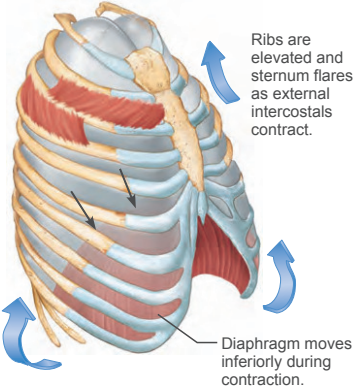
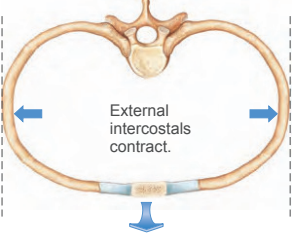
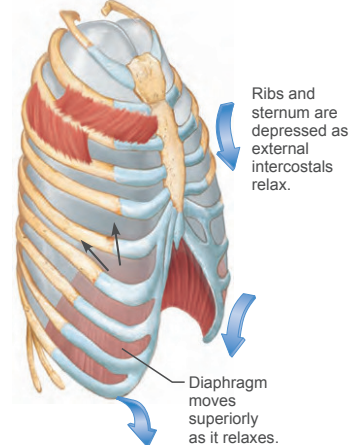
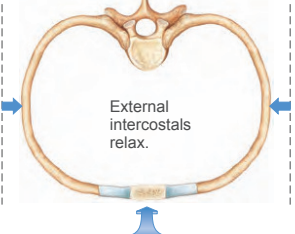
	Sequence of events	Changes in anterior-posterior and superior-inferior dimensions	Changes in lateral dimensions (superior view)
<b>Inspiration</b>	<ol style="list-style-type: none"> <li>① Inspiratory muscles contract (diaphragm descends; rib cage rises).</li> <li>↓</li> <li>② Thoracic cavity volume increases.</li> <li>↓</li> <li>③ Lungs are stretched; intrapulmonary volume increases.</li> <li>↓</li> <li>④ Intrapulmonary pressure drops (to <math>-1</math> mm Hg).</li> <li>↓</li> <li>⑤ Air (gases) flows into lungs down its pressure gradient until intrapulmonary pressure is 0 (equal to atmospheric pressure).</li> </ol>	 <p>Ribs are elevated and sternum flares as external intercostals contract.</p> <p>Diaphragm moves inferiorly during contraction.</p>	 <p>External intercostals contract.</p>
<b>Expiration</b>	<ol style="list-style-type: none"> <li>① Inspiratory muscles relax (diaphragm rises; rib cage descends due to recoil of costal cartilages).</li> <li>↓</li> <li>② Thoracic cavity volume decreases.</li> <li>↓</li> <li>③ Elastic lungs recoil passively; intrapulmonary volume decreases.</li> <li>↓</li> <li>④ Intrapulmonary pressure rises (to <math>+1</math> mm Hg).</li> <li>↓</li> <li>⑤ Air (gases) flows out of lungs down its pressure gradient until intrapulmonary pressure is 0.</li> </ol>	 <p>Ribs and sternum are depressed as external intercostals relax.</p> <p>Diaphragm moves superiorly as it relaxes.</p>	 <p>External intercostals relax.</p>

Figure 2.3: Inspiration and expiration sequences.

### **The External Intercostals**

When they are stimulated to contract, the external intercostal, parasternal intercostal, and scalene muscles raise and enlarge the rib cage. The parasternal muscles, which are usually considered part of the internal intercostals, are inspiratory muscles and may be partly responsible for raising the lower ribs. The scalene muscles appear to contract in normal quiet breathing and are therefore not accessory muscles [12].

These muscles are innervated by nerves leaving the spinal cord at the first through the eleventh thoracic segments. During inspiration, the diaphragm and inspiratory rib cage muscles contract simultaneously. If the diaphragm contracted alone, the rib cage muscles would be pulled inward (this is called retraction). If the inspiratory muscles of the rib cage contracted alone, the diaphragm would be pulled upward into the thorax [12].

### **2.1.2 Expiratory muscles**

Expiration is passive during normal quiet breathing, and no respiratory muscles contract. During exhalation (breathing out), at rest, all the muscles of inhalation relax, returning the chest and abdomen to a position called the “resting position”, which is determined by their anatomical elasticity [12]. At this point the lungs contain the functional residual capacity of air, which, in the adult human, has a volume of about 2.5–3.0 liters. Although the diaphragm is usually considered to be completely relaxed during expiration, it is likely that some diaphragmatic muscle tone is maintained, especially when one is in the horizontal position. The inspiratory muscles may also continue to contract actively during the early part of expiration, especially in obese people. This so-called braking action may help maintain a smooth transition between inspiration and expiration. It may also be important during speech production [12].

Active expiration occurs during exercise, speech, singing, the expiratory phase of coughing or sneezing, and in pathologic states such as chronic bronchitis.

The main muscles of expiration are the muscles of the abdominal wall and the internal intercostal muscles.

### **The Abdominal Muscles**

When the abdominal muscles contract, they increase abdominal pressure and push the abdominal contents against the relaxed diaphragm, forcing it upward into the thoracic cavity. They also help depress the lower ribs and pull down the anterior part of the lower chest [12].

### **The Internal Intercostal Muscles**

Contraction of the internal intercostal muscles depresses the rib cage downward in a manner opposite to the actions of the external intercostals [12].

## **2.2 Optoelectronic plethysmography**

The measurement of pulmonary ventilation is usually done with a spirometer or a pneumotachograph. However, this procedure is more complex than it may seem, because temperature, humidity, pressure, viscosity, and density of gas influence the recording of its volume. Mouthpieces, face masks, and noseclips may introduce leaks and therefore cause losses, are impractical for prolonged

measurement, limit the subject's mobility, introduce additional dead space, and thereby increase tidal volume. They also make the subject aware that his breathing is being measured and therefore interfere with the natural pattern of breathing and its neural control [4].

All these problems have induced investigators to attempt to measure ventilation indirectly by external measurement of chest wall surface motion. Optoelectronic plethysmography (OEP) is a now-established technique, based on typical methods for optical motion analysis, that allows to measure the variations of the volume of the chest wall and its compartments during breathing. A number of reflective markers are positioned by a hypoallergenic tape on the trunk of the subject in selected anatomical reference sites of the rib cage and the abdomen, as in Figure 2.4. A set of cameras is placed nearby the subject under analysis. Each camera is equipped with an illuminator (infrared light-emitting diodes) that determines a high contrast between the reflective marker and the rest of the scene on the recorded image, thus allowing the fully automatic recognition of the markers. When a single marker is seen by two or more cameras, its position (defined by the three-dimensional coordinates in the reference system of the laboratory) can be calculated by stereophotogrammetry, being known the position, orientation, and the internal parameters of each camera. Once the 3D coordinates (X, Y, Z) of the points belonging to the chest wall surface are acquired with reference to an arbitrary coordinate system, a closed surface is defined by connecting the points to form triangles (mesh of triangles). For each triangle, the area and the direction of the normal of the plane defined by that triangle are determined. Successively, the internal volume of the shape is computed using Gauss' theorem (or divergence theorem, or Green's theorem in space) [4].

In the last years, different protocols of OEP have been developed for different experimental and clinical situations. The validation of these measurement protocols was always performed by comparing the chest wall volume variation, measured by OEP, with lung volume variations measured by a spirometer or integrating a flow measurement at the airway opening. In the first studies, volume changes were compared in healthy subjects while sitting or standing and wearing 89 markers, during quiet breathing, slow vital capacity maneuvers, and incremental exercise on a cycle ergometer. In these conditions, the coefficient of variation of the two signals was always lower than 4%. Successively, OEP was validated in constrained postures, like the supine and prone position [4].

Another advantage of OEP with respect to the traditional Spirometry is that it allows to study the contribution of the different muscles of respiration during breathing. The chest wall can be modeled as being composed of three different compartments: pulmonary rib cage (RCp), abdominal rib cage (RCa), and the abdomen (AB), as depicted in Figure 2.5. This model is the most appropriate for the study of chest wall kinematics in the majority of conditions, including exercise. It takes into consideration the fact that the lung- and diaphragm-apposed parts of the rib cage (RCp and RCa, respectively) are exposed to substantially different pressures on their inner surface during inspiration, that the diaphragm acts directly only on RCa, and that non-diaphragmatic inspiratory muscles act largely on RCp. Abdominal volume change is defined as the volume swept by the abdominal wall [4].

In our work we will analyse data collected by means of OEP. A data acquisition from a subject consists in:

- Time (s) : the vector of times at which the markers positions were measured (and volume was computed). Frequency of acquisition is 60Hz;
- RCp volume (L);
- RCa volume (L);



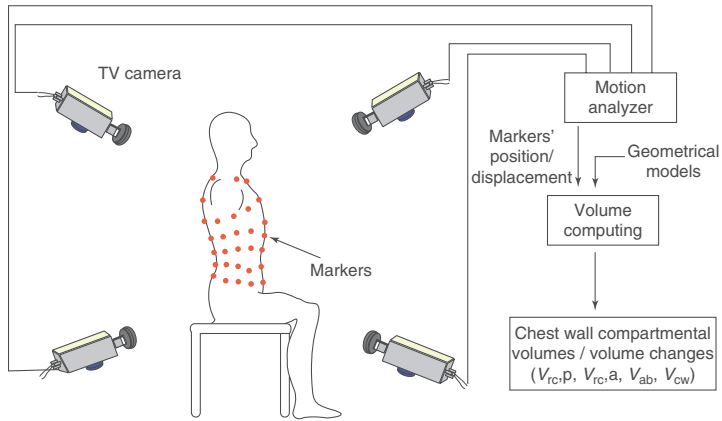


Figure 2.4: Optoelectronic plethysmography (OEP): principle of measurement

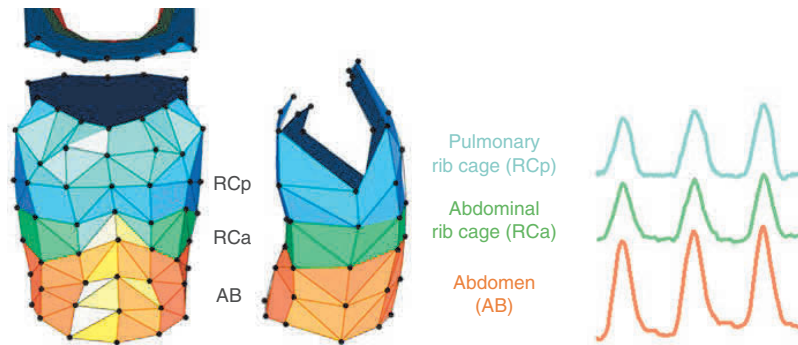


Figure 2.5: Geometrical models of the three chest wall compartments: pulmonary rib cage (RCp), abdominal rib cage (RCa), and abdomen (AB) (left, three views) and their volume changes during quiet spontaneous breathing, respectively,  $V_{rcp}$ ;  $V_{rca}$ ; and  $V_{ab}$ . Chest wall volume ( $V_{cw}$ ) is equal to  $V_{rcp} + V_{rca} + V_{ab}$ .

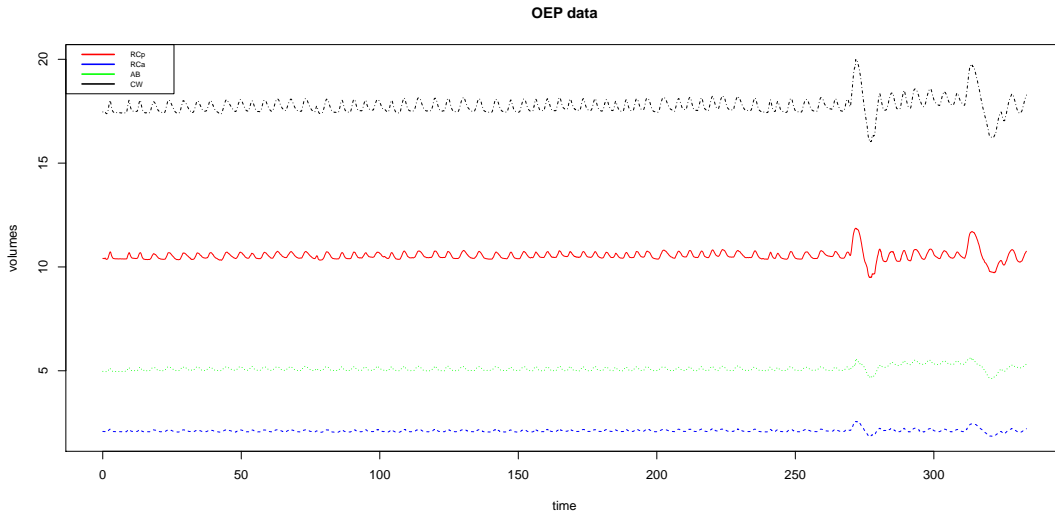


Figure 2.6: Acquired OEP data for a healthy subject in sitting position.

- AB volume (L);
- Chest Wall volume or Total volume (L), that is the sum of the volumes of the three compartments.

A visual representation of raw data is given in Figure 2.6.

OEP data can be used to evaluate the so-called *breathing pattern*. Some breathing patterns are for example the Quiet Breathing (QB) or eupnea, which is the ordinary breathing at rest of a subject, forced breathing or hyperpnea, which is the breath modality during exercise, paradoxical breathing, that is inward abdominal or rib cage movement with inspiration and outward movement with expiration, apnea, tachipnea and many more. Each pattern is characterized by specific quantities which can be measured based on OEP data or a spirogram. Figure 2.7 shows some of them: total lung capacity (TLC), functional residual capacity (FRC), expiratory reserve volume (ERV), residual volume (RV), vital capacity (VC), tidal volume ( $V_T$ ). Other important quantities are also the end-expiratory lung volume (EELV) and end-inspiratory lung volume (EILV), the respiratory rate (number of breaths per minute) and ventilation (the product of breathing frequency and tidal volume).

## 2.3 State of the art

Despite the computation of breathing pattern parameters has been standardized and can be done almost automatically, state-of-art analysis of OEP data still involves arbitrary choices of the analyst, especially for what concerns positioning the local minima of the signal (to get single breath curves) which is usually done by hand by the operator. The selection of a sample of so-obtained curves where to compute the breathing pattern parameters is also arbitrary: for example, outliers are removed based on visual inspection [14]. Moreover, in some applications (such as exercise) different

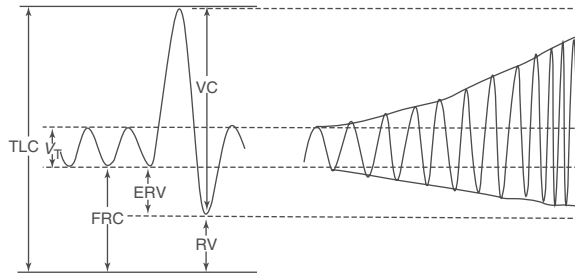


Figure 2.7: On the left, a normal spirogram. On the right, schematic diagram of the variations of VT during incremental exercise in healthy subjects.

breathing patterns may coexist in the same acquisition [2], while no automatized methodology to analyse cluster structures in this kind of data is available. For an example of a recent study in the field of breathing pattern analysis, one can see for instance Lo Mauro et al., 2016 [13].

The aforementioned issues can be tackled statistically by means of Functional Data Analysis. In fact, OEP data can be interpreted as a sequence of realisations of a smooth underlying function, that is the breath function of the patient under consideration. Then, it is reasonable to study the OEP data in this framework. In addition, the functional interpretation of breaths allows for a richer analysis, since it is possible to characterize the breathing pattern of a subject by means of its representative breathing function other than traditional scalar parameters.

This work represents the first attempt of application of Functional Data Analysis to Optoelectronic Plethysmography data. Data analysed in the thesis have been provided by LaRes - Laboratorio di Analisi della Respirazione at the Dipartimento di Elettronica Informazione e Bioingegneria of Politecnico di Milano. LaRes mission is to create innovation in respiratory medicine developing new methodologies and technologies for respiratory diseases analysis, functional evaluation and treatment. Part of these data have been collected by LaRes during experimental campaigns, and part through scientific collaborations between LaRes and other hospitals: Ospedale Maggiore Policlinico di Milano, Istituto Medea Bosisio Parini and Uppsala University Hospital.

## Chapter 3

# Functional Data Analysis

Functional Data Analysis is a branch of statistics which is able to study complex data, seen as observations of functions varying over a continuum. This kind of data arises in many real-life applications of statistics, especially in the biomedical field. Indeed, in many biomedical and health studies, individual observations are signals or temporal curves observed at discrete time points, and each curve provides the evolution over time of a certain process of interest for a given individual. This is also the case of the OEP data, that we can view as an ensemble of smooth curves, each one representing a breath. In this chapter, we will describe the Functional Data Analysis tools that we employed to study such data.

### 3.1 Smoothing

In Functional Data Analysis, *smoothing* means finding a functional representation of data, given noisy measurements. In this section, we will refer to the treatment of linear smoother of Ramsay and Silverman, 2005 [18].

Consider a sample of  $m$  curves  $\{t_{ij}, y_{ij}\}_{j=1 \dots n_i}$ ,  $i = 1 \dots m$ . We suppose that there exists a smooth function  $x_i$  such that a data vector  $\mathbf{y}_i$  can be expressed as a sum of this function plus a noise term  $\varepsilon$ :

$$y_{ij} = x(t_{ij}) + \varepsilon_{ij} \quad j = 1 \dots n_i, \quad i = 1 \dots m, \quad (3.1)$$

where  $\varepsilon_{ij}$  *i.i.d.*,  $\mathbb{E}[\varepsilon_{ij}] = 0$ ,  $\text{Var}(\varepsilon_{ij}) = \sigma^2$ .

Normally, the construction of the functional observations  $x_i$  using the discrete data  $y_{ij}$  takes place separately or independently for each record  $i$ , therefore we simplify notation assuming that a single function  $x$  for a data vector  $\mathbf{y}$  has to be estimated, that is (in vector notation)

$$\mathbf{y} = x(\mathbf{t}) + \varepsilon. \quad (3.2)$$

To obtain a functional representation of our data, we can express  $x$  as a combination of *basis functions*. A basis function system is a set of known functions  $\phi_k$  that are mathematically independent of each other and such that we can approximate arbitrarily well any function by taking a weighted sum or linear combination of a sufficiently large number  $K$  of these functions. Basis function procedures represent a function  $x$  by a linear expansion:

$$\hat{x}(t) = \sum_{k=1}^K \hat{c}_k \phi_k(t) \quad (3.3)$$

in terms of the  $K$  known basis functions. The  $\hat{c}_k$  are called the *coefficients* of the basis expansion. Let us express this relation in matrix terms:

$$\mathbf{y} = \mathbf{\Phi} \hat{\mathbf{c}} + \boldsymbol{\varepsilon}, \quad (3.4)$$

where

$$\mathbf{\Phi} = \begin{bmatrix} \phi_1(t_1) & \phi_2(t_1) & \dots & \phi_K(t_1) \\ \phi_1(t_2) & \phi_2(t_2) & \dots & \phi_K(t_2) \\ \vdots & \vdots & & \vdots \\ \phi_1(t_n) & \phi_2(t_n) & \dots & \phi_K(t_n) \end{bmatrix}$$

$$\mathbf{y} = (y_1, \dots, y_n)^T, \quad \hat{\mathbf{c}} = (\hat{c}_1, \dots, \hat{c}_K)^T.$$

Notice that, thanks to basis functions, we can also express the derivative of our signal since:

$$\hat{x}(t) = \sum_{k=1}^K \hat{c}_k D\phi_k(t). \quad (3.5)$$

A linear smoother can be obtained minimizing the least squares criterion:

$$\text{SSE} = \sum_{j=1}^n (y_j - \sum_{k=1}^K \hat{c}_k \phi_k(t_j)). \quad (3.6)$$

In matrix terms:

$$\text{SSE} = (\mathbf{y} - \mathbf{\Phi} \hat{\mathbf{c}})^T (\mathbf{y} - \mathbf{\Phi} \hat{\mathbf{c}}). \quad (3.7)$$

The least squares solution is therefore

$$\hat{\mathbf{c}} = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{y}$$

$$\hat{\mathbf{y}} = \mathbf{\Phi} \hat{\mathbf{c}} = \mathbf{\Phi} (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^T \mathbf{y} = \mathbf{S} \mathbf{y} \quad (3.8)$$

where matrix  $\mathbf{S}$  is called the *smoothing matrix*.

There are many possible choices for the basis functions  $\phi_k$ : in our application, we chose to smooth our data with B-spline basis system. This choice is motivated by B-splines flexibility and efficiency in computations.

### Curve smoothing using B-splines

Spline functions are the most common choice of approximation system for non-periodic functional data or parameters, as they combine fast computation with great flexibility, often achieved with a small number of basis functions.

The first step in defining a spline is to divide the interval over which a function is to be approximated into  $L$  subintervals separated by values  $\tau_i$ ,  $i = 1, \dots, L - 1$  that are called breakpoints or knots. Over each interval, a spline is a polynomial of specified order  $m$ . The order of a polynomial is the number of constants required to define it, and is one more than its degree, its highest power. Adjacent polynomials join up smoothly at the breakpoint which separates them for splines of order greater than one, so that the function values are constrained to be equal at their junction. Moreover, derivatives up to order  $m - 2$  must also match up at these junctions. There is the possibility of reducing these smoothness constraints by using multiple knots at junction points. Thus, the term *breakpoint*, strictly speaking, refers to the number of unique knot values, while the term *knot* refers to the sequence of values at breakpoints, where some breakpoints can be associated with multiple knots. The knots are all distinct in most applications, and consequently breakpoints and knots are then the same thing. The total number of degrees of freedom in the fit (the number of parameters required to define a spline function) equals the order of the polynomials plus the the number of interior breakpoints:  $m + L - 1$ . If there are no interior knots, the spline reverts to being a simple polynomial.

Any spline system fulfills the following properties:

- Each basis function  $\phi_k(t)$  is itself a spline function as defined by an order  $m$  and a knot sequence  $\tau$ ;
- Any linear combination of these basis functions is a spline function;
- Any spline function defined by  $m$  and  $\tau$  can be expressed as a linear combination of these basis functions.

The most popular spline system is the B-spline one, which we are using in this work.

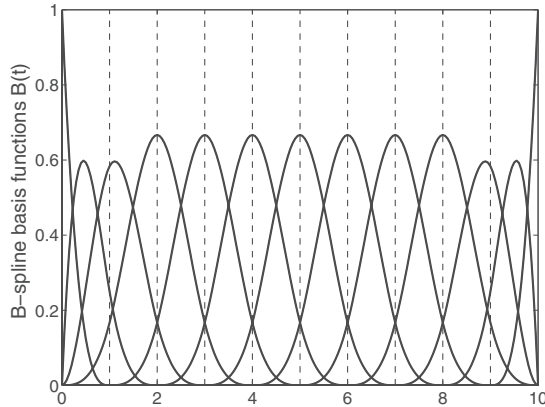


Figure 3.1: The thirteen basis functions defining an order four spline with nine interior knots, shown as vertical dashed lines.

Figure 3.1 shows the thirteen B-spline basis functions for an order four spline defined by nine equally spaced interior breakpoints. Because cubic splines have two continuous derivatives, each basis function makes a smooth transition to the regions over which it is zero.

The property that an order  $m$  B-spline basis function is positive over no more than  $m$  intervals, and that these are adjacent, is called the *compact support* property, and enables efficient computations.

### Penalized smoothing

A crucial point of spline smoothing consists in selecting the right number of basis  $K$ : we wish to ensure that the estimated curve gives a good fit to the data, for example in terms of the residual sum of squares, but we do not wish the fit to be too dependent on our noisy data if this results in a curve that is excessively locally variable, that is, we wish to keep the variance of the estimate under control. Penalization allows to use a rich functional space ( $K \sim n$ ) without the risk of overfitting.

In our case, instead of finding the classical least squares solution by minimizing the SSE as we have seen before, we minimize the penalized sum of squares  $\text{PENSSE} = \text{SSE} + \lambda \text{PEN}(x)$ , where  $\lambda$  can be chosen for example with Generalized Cross Validation (GCV). Penalization introduces some bias in the estimate, but allows to significantly reduce variance when we use a large  $K$ , therefore it is a convenient choice to reduce the overall Mean Squared Error (MSE), which is a widely used goodness-of-fit indicator consisting of a variance term plus a bias term:

$$\text{MSE}[\hat{y}(t)] = \text{Bias}^2[\hat{y}(t)] + \text{Var}(\hat{y}(t)). \quad (3.9)$$

The MSE of a fit can often be reduced by introducing some bias in order to reduce sampling variance, and this is a key reason for imposing smoothness on the estimated curve. In fact,  $\text{PEN}(x)$  can be defined as a roughness penalty, making use of the function derivatives. Indeed, many functional data analyses require the estimation of derivatives, either because these are of direct interest, or because they play a role in some other part of the analysis. A popular way to quantify the notion of “roughness” of a function is the integrated squared second derivative of the function. In fact the square of the second derivative  $[D^2x(t)]^2$  of a function at  $t$  is often called its curvature at  $t$ , since a straight line, which has no curvature, also has a zero second derivative. In this context  $\text{PEN}_2(x) = \int [D^2x(s)]^2 ds$ .

This penalty may not be suitable, since it controls curvature in  $x$  itself, and therefore only the slope in the derivative  $Dx$ . It does not require the second derivative  $D^2x$  even to be continuous, let alone smooth in any sense. In general, if the derivative of order  $m$  is the highest required by a problem, one should actually penalize the derivatives of order  $m+2$  in order to control the curvature of the highest order derivative.

In this work, we are going to use the first derivative of our signal. Therefore, we introduce a penalisation over the breaths third derivative by minimizing this loss function:

$$\text{PENSSE}_\lambda = \text{SSE} + \lambda \text{PEN}_3(x), \quad (3.10)$$

with  $\text{PEN}_3(x) = \int [D^3x(s)]^2 ds$ .

This can be expressed in matrix form as

$$\text{PENSSE}_\lambda = \text{SSE} + \lambda \mathbf{c}^T \mathbf{R}_\phi \mathbf{c}, \quad (3.11)$$

where  $R_{\phi(k,l)} = \int D^3\phi_k(s)D^3\phi_l(s)ds$ .

The least squares solution thus becomes:

$$\begin{aligned}\hat{\mathbf{c}}_\lambda &= (\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \mathbf{R}_\phi)^{-1} \mathbf{\Phi}^T \mathbf{y} \\ \hat{\mathbf{y}} &= \mathbf{\Phi} (\mathbf{\Phi}^T \mathbf{\Phi} + \lambda \mathbf{R}_\phi)^{-1} \mathbf{\Phi}^T \mathbf{y} = \mathbf{S} \mathbf{y}\end{aligned}\quad (3.12)$$

## 3.2 Functional Outlier Detection

### 3.2.1 Depth measures for functional data

In univariate statistics, boxplots are used to display descriptive statistics of data: the median, the first and third quartiles and the extreme observations. In particular, the median is often referred to as the most "central" observation of the sample, since it is morally the central observation with respect to the ordered sample [21].

To generalize order statistics or ranks to the functional setting, different versions of data depth have been introduced to measure how deep (central) or outlying an observation is.

In this work we will adopt depth measures for multivariate functional data as described in Ieva et al., 2013 [9] and Ieva and Paganoni, 2017 [10].

First, let us describe the notion of Band Depth (BD), as introduced in López-Pintado and Romo, 2009 [15]. We suppose to have a sample of  $n$  observations of the form  $y_i = y_i(t)$ ,  $i = 1, \dots, n$ ,  $t \in I$ , where  $I$  is an interval in  $\mathbb{R}$ . The idea behind band depth is to provide a method to order functional data according to decreasing depth values:  $y_{[1]}(t)$ , that is the curve with highest depth, will be the deepest (most central) curve or simply the median curve, and  $y_{[n]}(t)$  will be the most outlying curve.

López-Pintado and Romo introduced the band depth concept through a graph-based approach. The *graph* of a function  $y(t)$  is the subset of the plane

$$G(y) = \{(t, y(t)) : t \in I\}.$$

The *band* in  $\mathbb{R}^2$  delimited by the curves  $y_{i_1}, \dots, y_{i_k}$  is

$$B(y_{i_1}, \dots, y_{i_k}) = \{(t, x(t)) : t \in I, \min_{r=1, \dots, k} y_{i_r}(t) \leq x(t) \leq \max_{r=1, \dots, k} y_{i_r}(t)\}. \quad (3.13)$$

Let  $J$  be the number of curves determining a band, where  $J$  is a fixed value with  $2 \leq J \leq n$ . If  $Y_1(t), \dots, Y_n(t)$  are independent copies of the stochastic process  $Y(t)$  generating the observations  $y_1, \dots, y_n$ , the population version of the band depth for a given curve  $y(t)$  with respect to the probability measure  $P$  is defined as

$$BD_J(y, P) = \sum_{j=2}^J BD^{(j)}(y, P) = \sum_{j=2}^J P\{G(y) \subset B(Y_1, \dots, Y_j)\}, \quad (3.14)$$

where  $B(Y_1, \dots, Y_j)$  is a band delimited by  $j$  random curves. The sample version of  $BD^{(j)}(y, P)$  is obtained by computing the fraction of the bands determined by  $j$  different sample curves containing the whole graph of the curve  $y(t)$ :

$$BD_n^{(j)} = \binom{n}{j}^{-1} \sum_{1 \leq i_1 < i_2 < \dots < i_j \leq n} \mathbb{I}\{G(y) \subseteq B(y_{i_1}, \dots, y_{i_j})\}, \quad (3.15)$$



where  $\mathbb{I}\{\cdot\}$  denotes the indicator function. Computing the fraction of the bands containing the curve  $y(t)$ , the bigger the value of band depth, the more central position the curve has.

Then, the sample band depth of a curve  $y(t)$  is

$$BD_{n,J}(y) = \sum_{j=2}^J BD_n^{(j)}(y). \quad (3.16)$$

A sample median function is a curve from the sample with largest depth value, defined by  $\operatorname{argmax}_{y \in \{y_1, \dots, y_n\}} BD_{n,J}(y)$ . If there are ties, the median will be the average of the curves maximizing depth.

López-Pintado and Romo also proposed a more flexible definition, which does not involve an indicator function: the modified band depth (MBD). MBD measures the proportion of time that a curve  $y(t)$  is in the band:

$$MBD_n^{(j)}(y) = \binom{n}{j}^{-1} \sum_{1 \leq i_1 < i_2 < \dots < i_j \leq n} \frac{\lambda\{A_j(y)\}}{\lambda\{I\}}, \quad (3.17)$$

where  $A_j(y) \equiv A(y; y_{i_1}, \dots, y_{i_j}) \equiv \{t \in I : \min_{r=i_1, \dots, i_j} y_r(t) \leq y(t) \leq \max_{r=i_1, \dots, i_j} y_r(t)\}$ , and  $\lambda$  is the Lebesgue measure on  $I$ .

From now on we will consider bands defined by two curves, i.e.  $J = 2$ . Let us consider for each pair of curves  $y_i$  and  $y_j$  the band that they define in  $I \times \mathbb{R}$ :

$$B(y_i, \dots, y_j) = \{(t, y(t)) : t \in I, \min(y_i(t), y_j(t)) \leq y(t) \leq \max(y_i(t), y_j(t))\}. \quad (3.18)$$

Then,  $MBD_n(y)$  is the mean over all possible bands of the time that  $y(t)$  spends inside a band, while the original band depth accounts for the proportion of bands in which a curve is entirely contained. If  $y(t)$  is always inside the band, the modified band depth degenerates to the band depth.

Because the modified band depth takes the proportion of times that a curve is in the band into account, it avoids having too many depth ties and is more convenient to obtain the most representative curves in terms of magnitude.

Another interesting depth measure is the Modified Epigraph Index (MEI). As in the case of the MBD, the MEI is a generalization of the Epigraph Index (EI), that accounts for the proportion of curves that lie entirely above  $y$  (Lopez-Pintado and Romo, 2011 [15]).

The Epigraph Index is defined as:

$$EI(y) = \frac{1}{n} \sum_{i=1}^n \mathbb{I}\{y_i(t) \geq y(t), \forall t \in I\}. \quad (3.19)$$

The modified epigraph index of  $y \in \{y_1, \dots, y_n\}$  is defined as

$$MEI_{\{y_1, \dots, y_n\}}(y) = \frac{1}{n} \sum_{i=1}^n \frac{\lambda(\{t \in I | y_i(t) \geq y(t)\})}{\lambda(I)}, \quad (3.20)$$

and it stands for the mean proportion of time that  $y$  lies below the curves of the sample.

In Ieva et al., 2013 [9] and Ieva and Paganoni, 2017 [10] these statistics have been generalized to the multivariate functional framework. Let  $Y$  be a stochastic process taking values in the

space  $C(I; \mathbb{R}^L)$  of continuous functions  $\mathbf{y} = (y_1, \dots, y_L) : I \rightarrow \mathbb{R}^L$ , with  $I$  as before. We have a dataset constituted of  $N$  sample observations of this process, which we indicate by  $\mathbf{y}_1, \dots, \mathbf{y}_n$ , where  $\mathbf{y}_j = (y_{j1}, \dots, y_{jL})$ . The MBD and MEI of  $\mathbf{y}$  become:

$$\begin{aligned} MBD_{\mathbf{y}_1, \dots, \mathbf{y}_n}(\mathbf{y}) &= \sum_{k=1}^L p_k MBD_{y_{1k}, \dots, y_{nk}}^J(y_k) \\ MEI_{\mathbf{y}_1, \dots, \mathbf{y}_n}(\mathbf{y}) &= \sum_{k=1}^L p_k MEI_{y_{1k}, \dots, y_{nk}}(y_k) \end{aligned} \tag{3.21}$$

with  $p_k > 0$ ,  $\forall k = 1, \dots, L$ ,  $\sum_{k=1}^L p_k = 1$ , where the choice of the weights  $p_k$  is usually problem-driven.

### 3.2.2 Magnitude functional outlier detection

An useful tool to detect functional outliers is the functional boxplot developed by Sun and Genton, 2010 [21]. They used MBD to provide an ordering of the observations according to their depth measures, and built a functional version of the well-known Inter Quartile Range (IQR).

Let us indicate the most central curve by  $y_{[1]}(t)$ , and the most outlying one with  $y_{[n]}(t)$ . In the classical boxplot, the box itself represents the middle 50% of the data. The band delimited by the  $\alpha$  proportion ( $0 < \alpha < 1$ ) of deepest curves from the sample is used to estimate the  $\alpha$  central region. In particular, the sample 50% central region is

$$C_{0.5} = \{(t, y(t)) : \min_{r=1, \dots, [n/2]} y_{[r]}(t) \leq y(t) \leq \max_{r=1, \dots, [n/2]} y_{[r]}(t)\}, \tag{3.22}$$

where  $[n/2]$  is the smallest integer not less than  $n/2$ . The border of the 50% central region is defined as the envelope representing the box in a classical boxplot. Thus, this 50% central region is the analog to the IQR and gives a useful indication of the spread of the central 50% of the curves. This is a robust range for interpretation because the 50% central region is not affected by outliers or extreme values, and gives a less biased visualization of the curves' spread. The median curve  $y_{[1]}(t)$  is also a robust statistic to measure centrality. The “whiskers” of the boxplot are the vertical lines of the plot extending from the box and indicating the maximum envelope of the dataset except the outliers. Sun and Genton proposed to extend the 1.5 times IQR empirical outlier criterion to the functional boxplot. The fences are obtained by inflating the envelope of the 50% central region by 1.5 times the range of the 50% central region. Any curves outside the fences are flagged as potential outliers and can be isolated from the original dataset.

Ieva et al., 2013 [9] generalized this procedure to construct the multivariate functional boxplot, by using the multivariate definition of MBD to rank observations. This implies that the envelope of the central region is composed of the same  $\alpha\%$  most central curves, with respect the multivariate index of depth, in each component.

### 3.2.3 Shape functional outlier detection

A method that can be used to detect shape outliers and covariance outliers is the outliergram (see Arribas-Gil and Romo, 2014 [6]), based on the computation of MBD and MEI of univariate functional data. Shape outliers are curves that present a different pattern with respect to the rest

of the data in terms of their derivatives. Covariance outliers are curves generated by a model that is different from the model of the majority of data just in terms of the variance and covariance operator that affects the second order moments of data.

Given a set of data  $y_1, \dots, y_n$  in the space  $C(I; \mathbb{R})$  of the continuous functions the following inequality holds [6]:

$$MBD_{y_1, \dots, y_n}(y_j) \leq a_0 + a_1 MEI_{y_1, \dots, y_n}(y_j) + a_2 n^2 (MEI_{y_1, \dots, y_n}(y_j))^2, \quad j = 1, \dots, n, \quad (3.23)$$

where  $a_0 = a_2 = -2/(n(n-1))$  and  $a_1 = 2(n+1)/(n-1)$ . So considering a scatterplot of MBD against multivariate MEI of data, the points lying far from the quadratic boundary correspond to shape outliers, and data with very low values of MBD are potential magnitude outliers.

If in a sample of perfectly aligned curves with common shape one introduces a curve with a different pattern, then the  $\mathbb{R}^2$  point corresponding to the pair (MEI, MBD) for this new curve will lie far away from the parabola defined by the points corresponding to the rest of the curves. In general trajectories of a random process will cross many times even if they all exhibit the same trend pattern. Then, the (MEI, MBD) points will not define a perfect parabola and identifying outlying trajectories will not be straightforward. Arribas-Gil and Romo proposed the use of the univariate boxplot rule for outlier detection on the vertical distances to the parabola: given a sample of curves  $y_1, \dots, y_n$  with  $mb_j = MBD_{y_1, \dots, y_n}(y_j)$  and  $me_j = MEI_{y_1, \dots, y_n}(y_j)$  for  $j = 1, \dots, n$  we consider the distances  $d_j = a_0 + a_1 me_j + n^2 a_2 me_j^2 - mb_j$  and define as shape outliers those curves with  $d_j \geq Q_{d_3} + 1.5IQR_d$ , where  $Q_{d_3}$  and  $IQR_d$  are the third quartile and inter-quartile range of  $d_1, \dots, d_n$ .

To jointly visualize the observations in terms of shape and the boundary between the outlying and non-outlying curves they propose to represent in  $\mathbb{R}^2$  the (MEI, MBD) points together with the parabola shifted downwards by  $Q_{d_3} + 1.5IQR_d$ . This graphical representation is referred to as the outliergram.

In Ieva and Paganoni, 2017 [10] the following inequality is proved:

$$MBD_{y_1, \dots, y_n}^J(y) \leq a_0 + a_1 MEI_{y_1, \dots, y_n}(y_j) + a_2 n^2 (MEI_{y_1, \dots, y_n}(y_j))^2, \quad j = 1, \dots, n, \quad (3.24)$$

where  $a_0 = a_2 = -2/(n(n-1))$  and  $a_1 = 2(n+1)/(n-1)$ . Therefore, it is possible to generalize the outliergram to multivariate functional setting, with a construction that is analogue to the univariate case.

### 3.3 Curve registration and clustering

A common problem encountered in functional data analysis is the misalignment of data (Ramsay and Silverman, 2005 [18], Sangalli et al., 2010 [19]). Indeed, variation in functional observations involves both the *amplitude* and the *phase* of the data. Amplitude variability pertains to the sizes of particular features such as peaks or valleys of our functional data, ignoring their timings. Phase variability is variation in the timings of the features without considering their sizes (see Figure 3.2).

Let us consider two breaths  $b_1$  and  $b_2$  pertaining to the same breathing track. One of our goals will be to cluster breaths of the same subject, in order to see whether there are groups in his quiet breathing. The breath curves  $b_1$  and  $b_2$  can in principle differ because of two types of variation. The first is vertical variation, or amplitude variation, due to the fact that  $b_1(t)$  and  $b_2(t)$  may simply differ at points of time  $t$  at which they are compared, but otherwise exhibit the same shape features at that time. But they may also exhibit phase variation in the sense that functions  $b_1$  and  $b_2$  should

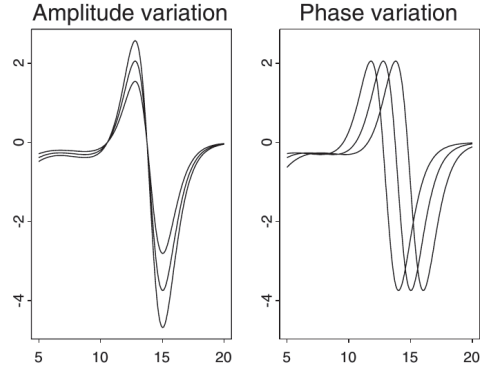


Figure 3.2: On the left panel an example of amplitude-only variation is shown; on the right an example of phase-only variation.

not be compared at the same time  $t$  because they are not exhibiting the same behavior. For this reason, before comparing shape and amplitude of two breaths we need to *align* them, that is to transform their time scales in order to make them compliant.

### 3.3.1 Mathematical framework

In this work we will adopt the methodology developed by Sangalli et al, 2010 [19], which is able to efficiently cluster and align in  $K$  groups a set of curves.

As we have seen, variability among two or more curves can be thought of as having two components: phase variability and amplitude variability. Heuristically, phase variability is the one that can be eliminated by suitably aligning the curves, and amplitude variability is the one that remains among the curves once they have been aligned .

Consider a set  $C$  of (possibly multidimensional) curves  $\mathbf{c}(s) : \mathbb{R} \rightarrow \mathbb{R}^d$ . Aligning  $\mathbf{c}_1 \in C$  to  $\mathbf{c}_2 \in C$  means finding a warping function  $h(s) : \mathbb{R} \rightarrow \mathbb{R}$ , of the abscissa parameter  $s$ , such that the two curves  $\mathbf{c}_1 \circ h$  and  $\mathbf{c}_2$  are the most similar (with  $(\mathbf{c}_1 \circ h)(s) := \mathbf{c}(h(s))$ ).

It is thus necessary to specify a similarity index  $\rho(\mathbf{c}_1, \mathbf{c}_2) : C \times C \rightarrow \mathbb{R}$  that measures the similarity between two curves, and a class  $W$  of warping functions  $h$  (such that  $\mathbf{c}_1 \circ h \in C$ , for all  $\mathbf{c} \in C$  and  $h \in W$ ) indicating the allowed transformations for the abscissa. Aligning  $\mathbf{c}_1$  to  $\mathbf{c}_2$  means finding  $h^* \in W$  that maximizes  $\rho(\mathbf{c}_1 \circ h; \mathbf{c}_2)$ . This procedure decouples phase and amplitude variability without loss of information.

The choice of the couple  $(\rho, W)$  (which defines phase and amplitude variability) cannot be arbitrary, but has to satisfy some coherence properties:

- The similarity index  $\rho$  is bounded from above, with maximum value 1. Moreover,  $\rho$  is *reflexive*, *symmetric* and *transitive*;
- The class of warping functions  $W$  is a convex vector space and has a group structure with respect to function composition  $\circ$ .
- The couple  $(\rho, W)$  functions is consistent in the sense that , if two functions  $c_1$  and  $c_2$  are simultaneously warped along the same warping function  $h \in W$ , their similarity does not

dissimilarity $d$	warpings $W$
$\ c_1 - c_2\ $	$W_{shift}$
$\ c'_1 - c'_2\ $	$W_{shift}$
$\ (c_1 - \bar{c}_1) - (c_2 - \bar{c}_2)\ $	$W_{shift}$
$\ (c'_1 - \bar{c}'_1) - (c'_2 - \bar{c}'_2)\ $	$W_{shift}$
$\left\  \frac{c_1}{\ c_1\ } - \frac{c_2}{\ c_2\ } \right\ $	$W_{affinity}$
$\left\  \frac{c'_1}{\ c'_1\ } - \frac{c'_2}{\ c'_2\ } \right\ $	$W_{affinity}$
$\left\  \text{sign}(c'_1)\sqrt{ c'_1 } - \text{sign}(c'_2)\sqrt{ c'_2 } \right\ $	$W_{diffeomorphism}$

Figure 3.3: Possible choices of dissimilarity index and warping function class.

change:

$$\rho(c_1, c_2) = \rho(c_1 \circ h, c_2 \circ h) \quad \forall h \in W$$

This guarantees that is not possible to obtain a fictitious increment of similarity between two curves  $c_1$  and  $c_2$  by simply warping them simultaneously to  $c_1 \circ h, c_2 \circ h$ . This propriety is called *W-invariance*.

From the previous follows that for all  $h_1$  and  $h_2 \in W$ ,

$$\rho(c_1 \circ h_1, c_2 \circ h_2) = \rho(c_1 \circ h_1 \circ h_2^{-1}, c_2) = \rho(c_1, c_2 \circ h_2 \circ h_1^{-1})$$

This means that a change in similarity between  $c_1$  and  $c_2$  obtained by warping simultaneously  $c_1$  and  $c_2$  can be obtained by warping only  $c_1$  or  $c_2$ .

An equivalent formulation of this framework can be provided using couple  $(\mathcal{E}, W)$  where  $\mathcal{E}$  is a *dissimilarity* measure between two curves;  $\mathcal{E}$  has to satisfy analogous properties as  $\rho$ , except it has to be bounded from *below*, with minimum value 0. For multidimensional functions, the similarity/dissimilarity measure is computed via the average of the indexes in all directions.

Consistent  $(\mathcal{E}, W)$  couples are shown in Figure 3.3. The class of warping functions in the table are:

$$\begin{aligned} W_{shift} &= \{h : h(t) = t + q, q \in \mathbb{R}\} \\ W_{dilation} &= \{h : h(t) = mt, m \in \mathbb{R}^+\} \\ W_{affine} &= \{h : h(t) = mt + q, m \in \mathbb{R}^+, q \in \mathbb{R}\} \end{aligned}$$

and the more general class of diffeomorphisms  $W_{diffeomorphism}$ , composed by increasing functions that are smooth and have a smooth inverse. The case where no alignment is performed corresponds to the special case  $W_{identity} = \{h : h(t) = t\}$ .

### 3.3.2 K-means with alignment

Consider the problem of clustering and aligning a set of  $N$  curves  $\{\mathbf{c}_1, \dots, \mathbf{c}_N\}$  with respect to a set of  $k$  template curves  $\{\varphi_1, \dots, \varphi_N\}$  (with  $\{\mathbf{c}_1, \dots, \mathbf{c}_N\} \subset C$  and  $\underline{\varphi} \subset C$ ). For each template curve  $\varphi_j$ , define the domain of attraction  $\Delta_j(\underline{\varphi}) = \{\mathbf{c} \in C : \sup_{h \in W} \rho(\varphi_j, \mathbf{c} \circ h) \geq \sup_{h \in W} \rho(\varphi_r, \mathbf{c} \circ h), r \neq j\}$ ,  $j = 1, \dots, k$ , which is the ensembles of curves  $\mathbf{c}$  that are more similar to  $\varphi_j$  than to the other templates. Moreover, define the labelling function  $\lambda(\underline{\varphi}, \mathbf{c}) = \min\{r : \mathbf{c} \in \Delta_r(\underline{\varphi})\}$ , which indicates the cluster curve  $\mathbf{c}$  should be assigned to.

In order to cluster and align the set of  $N$  curves  $\{c_1, \dots, c_N\}$  with respect to  $k$  unknown templates we should first solve the following optimization problem:

Find  $\underline{\varphi} = \{\varphi_1, \dots, \varphi_k\} \subset C$  and  $\underline{\mathbf{h}} = \{h_1, \dots, h_N\} \subset W$  such that

$$\frac{1}{N} \sum_{i=1}^N \rho(\varphi_{\lambda(\underline{\varphi}, c_i)}, c_i \circ h_i) \geq \frac{1}{N} \sum_{i=1}^N \rho(\psi_{\lambda(\underline{\psi}, c_i)}, c_i \circ g_i) \quad , \quad (3.25)$$

for any other set of  $k$  templates  $\underline{\psi} = \{\psi_1, \dots, \psi_k\} \subset C$  and any other set of  $N$  warping functions  $\underline{\mathbf{g}} = \{g_1, \dots, g_N\} \subset W$ ; and then, for  $i = 1, \dots, N$ , assign  $c_i$  to the cluster  $\lambda(\underline{\varphi}; c_i)$ , and align it to the corresponding template,  $\varphi_{\lambda(\underline{\varphi}; c_i)}$ , using the warping function  $h_i$ .

This maximization problem is not analytically solvable; then, Sangalli et al [19] propose an iterative procedure to find an approximate solution. The *K-means with Alignment* algorithm proceeds as follows:

Let  $\underline{\varphi}_{[q-1]}$  be the set of templates after iteration  $q-1$ , and  $\{c_{1[q-1]}, \dots, c_{N[q-1]}\}$  be the  $N$  curves aligned and clustered to  $\underline{\varphi}_{[q-1]}$ . At the  $q$ -th iteration the algorithm performs the following steps.

**Template identification step.** For  $j = 1, \dots, k$ , the template of the  $j$ -th cluster,  $\varphi_{j[q]}$ , is estimated using all curves assigned to cluster  $j$  at iteration  $q-1$ , i.e. all curves  $c_{i[q-1]}$  such that  $\lambda(\underline{\varphi}_{[q-1]}; c_{i[q-1]}) = j$ .

Ideally, the template  $\varphi_{j[q]}$  should be estimated as the curve  $\varphi \in C$  that maximizes the total similarity:

$$\sum_{i: \lambda(\underline{\varphi}_{[q-1]}; c_{i[q-1]})=j} \rho(\varphi, c_{i[q-1]}) \quad (3.26)$$

The set  $M$  where the template functions are searched has to be fixed. Two choices are natural:

- $M$  may coincide with the entire functional space  $C$ . In this case, the representative functions are called Frechet templates (i.e. the *means*). The resulting algorithm is called K-mean Alignment algorithm;
- $M$  may coincide with the set of functions  $\{c_1, \dots, c_N\}$ , in which case the representative functions are called Karcher templates (that is, the *medians* or *medoids*) and the algorithm is called K-medoid Alignment.

**Assignment and alignment step.** The set of curves  $\{c_{1[q-1]}, \dots, c_{N[q-1]}\}$  are clustered and aligned to the set of templates  $\underline{\varphi}_{[q]}$  for  $i = 1, \dots, N$ ; the  $i$ -th curve  $c_{i[q-1]}$  is aligned to  $\varphi_{\lambda(\underline{\varphi}_{[q-1]}; c_{i[q-1]})}$  and the aligned curve  $\tilde{c}_{i[q]} = c_{i[q-1]} \circ h_{i[q]}$  is assigned to cluster  $\lambda(\underline{\varphi}_{[q-1]}; c_{i[q-1]}) = \lambda(\underline{\varphi}_{[q-1]}; \tilde{c}_{i[q-1]})$

**Normalization step.** After each assignment and alignment step, a normalization step is performed. For  $j = 1, \dots, k$ , all the  $N_{j[q]}$  curves  $\tilde{c}_{i[q]}$  assigned to cluster  $j$  are warped along the warping function  $(\bar{h}_{j[q]})^{-1}$  where

$$\bar{h}_{j[q]} = \frac{1}{N_{j[q]}} \sum_{i: \lambda(\underline{\varphi}_{[q-1]}; c_{i[q-1]})=j} h_{j[q]}. \quad (3.27)$$

In this way, at each iteration, the average warping undergone by curves assigned to cluster  $j$  is the identity transformation  $h(s) = s$ . The normalization step is thus used to select, among all candidate solutions to the optimization problem, the one that leaves the average locations of the clusters unchanged, thus avoiding the drifting apart of clusters or the global drifting of the overall set of curves. Note that the normalization step preserves the clustering structure chosen in the assignment and alignment step, i.e., for all  $i$ ,  $\lambda(\underline{\varphi}_{[q]}; c_{i[q]}) = \lambda(\underline{\varphi}_{[q]}; \tilde{c}_{i[q]})$

The algorithm is initialized with a set of initial templates  $\underline{\varphi}_{[0]} = \{\varphi_{1[0]}, \dots, \varphi_{k[0]}\} \subset C$ , and with  $\{c_{1[0]}, \dots, c_{N[0]}\} = \{c_1, \dots, c_N\}$ , and stopped when, in the assignment and alignment step, the increments of the similarity indexes are all lower than a fixed threshold.

If the problem is given in terms of dissimilarity measures, the optimization problem becomes a minimization one:

$$\frac{1}{N} \sum_{i=1}^N \mathcal{E}(\varphi_{\lambda(\underline{\varphi}, c_i)}, c_i \circ h_i) \leq \frac{1}{N} \sum_{i=1}^N \mathcal{E}(\psi_{\lambda(\underline{\psi}, c_i)}, c_i \circ g_i). \quad (3.28)$$

Thus, the template  $\varphi_{j[q]}$  should be estimated as the curve  $\varphi \in C$  that *minimizes* the total within-cluster dissimilarity:

$$\sum_{i: \lambda(\underline{\varphi}_{[q-1]}; c_{i[q-1]})=j} \mathcal{E}(\varphi, c_{i[q-1]}). \quad (3.29)$$

## Chapter 4

# Plethysmography data preprocessing

In this chapter a semi-automatic OEP data analysis procedure in R is proposed to prepare a functional dataset of breaths from raw data . The aim is to form a set of 4-dimensional smooth curves, each representing a single breath in its four components: total volume, RCp volume, RCa volume, AB volume. There are three main issues to face in this phase:

- **Breaths separation.** This step involves finding the local peaks of a noisy signal, and an opportune definition of breath;
- **Breaths smoothing.** This point is strictly linked with breaths separation;
- **Outlier detection.** Identifying and removing outliers is a fundamental step to make further analysis more robust.

### 4.1 Breaths separation

Breaths separation is done cutting the data vectors in pieces, each one representing a breath, where the cutting points are to be found with respect to the total volume. Indeed, referring to the Spirometry test, a breath is usually defined as the interval between the points of zero air flux coming out of the mouth during expiration [24]. Hence, we need to translate this concept in a functional framework in order to separate breaths correctly.

#### 4.1.1 Algorithm steps

Ideally, the cutting points we are looking for should be nothing more than the local minima of the chest wall volume. In real cases, however, the lower part of the signal can be very noisy and subject to small perturbations (Figure 4.1), due for example to heartbeat.

For this reason, positioning the cutting points is usually done by hand by experts, which is arbitrary and rather time-consuming; instead, we provide a robust method to do this almost automatically.



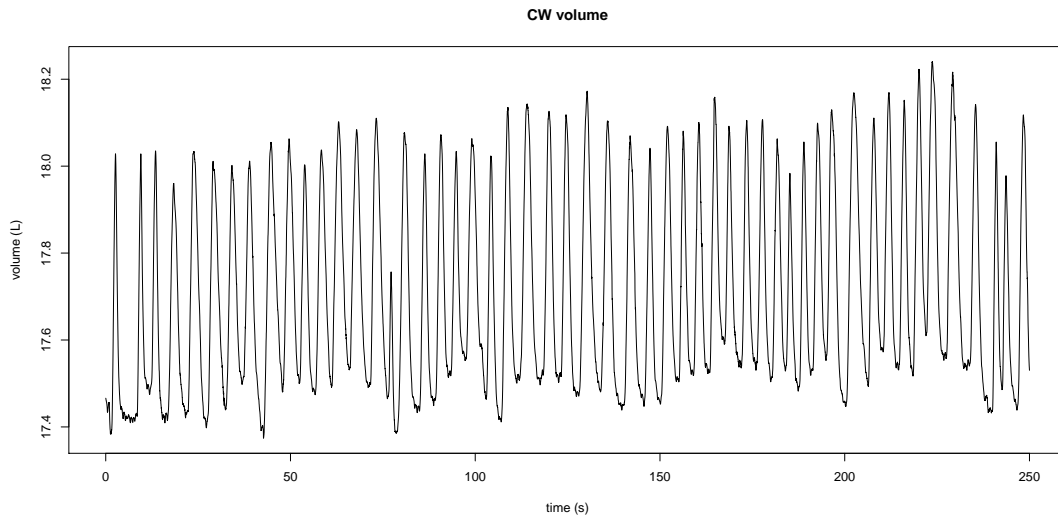


Figure 4.1: A real signal with problematic minima.

1. Find local maxima of the total volume;
2. Smooth the pieces of volume between two consecutive maxima;
3. Divide the smoothed curves in subintervals;
4. Find the subinterval where the mean derivative starts to grow;
5. Pick the local minimum of that subinterval as cutting point;
6. Project the cutting points over compartments;
7. Smooth the so-obtained breaths.

The reasoning behind this algorithm is the following: we say that a breath starts (and the previous breath ends) in the point where the total volume signal derivative starts to grow.

In order to obtain a smooth estimate of the signal derivative, we want to smooth our signal using B-splines with a penalization on the third derivative (see section 3.1). However, due to the high sampling frequency, the number of data points in a track is high and smoothing the whole total volume can be very time-consuming even when the data acquisition is rather short (acquisitions usually last more than 2 minutes).

A more efficient solution can be found taking advantage of a characteristic of breathing tracks: while the lower part of the signal is noisy and contains a lot of oscillations, we can instead locate local maxima very precisely. Therefore, we first take the local maxima of the total volume, and smooth the parts of signal between two consecutive maxima; then, we divide each smoothed curve in subintervals (5-10 points) where we compute the mean signal derivative. We take the cutting point as the local minimum of the volume in the interval formed by the union of the two subintervals

such that the mean derivative in the second subinterval is more than 3 times the mean derivative in the previous subinterval.

Once the cutting points have been found, the four signals (CW, RCp, RCa, AB) are cut in correspondence with these points, and the resulting breath components are smoothed like before using penalized B-splines.

## Code

Let us consider first the problem of smoothing. In order to smooth our breaths, we wrote a R function called `smooth_breath`. Implementation made use of functions from the R package `fda`, developed to support functional data analysis as described in Ramsay, J. O. and Silverman, B. W. (2005) [18], which is available on CRAN.

In the function `smooth_breath` we create a b-spline basis over an equidistant grid of knots. Then, we create an object of type *functional parameter*, to enclose penalization over the third derivative, and we evaluate the Generalized Cross Validation index for values of lambda on a log scale. Once found the best lambda, we evaluate the smoothed signal and its derivatives.

Function `smooth_breath` takes as input:

- `time`: the time vector, that is the abscissa of the signal;
- `amplitude`: the volume vector;
- `order`: the order of the b-spline basis to use. It defaults to 5;
- `grid_coef`: an integer. It defines the refinement of the knots grid. Defaults to 10 (1 knot each 10 datapoints).
- `lambda`: a vector with penalisation parameters, the best one is selected via Generalized Cross Validation (GCV). It is advisable to look for the optimal lambda on a log scale;
- `plot`: boolean. If 1 two plots are produced, one with the computed GCV for each lambda and one with the smoothed curve overlapped to the original signal.

Output quantities (contained in a list) are:

- `smoothed_curve`: the evaluation of the smooth signal;
- `der1`: the evaluation of the signal first derivative;
- `der2`: the evaluation of the signal second derivative;
- `lambda`: the optimal penalisation parameter;
- `GCV`: optimal GCV statistic;
- `df`: degrees of freedom in the smoothed curve.

```

smooth_breath= function(time, amplitude, order=5, grid_coef=10,
                        lambda=c(1e-4,1e-5,1e-6,1e-7,1e-8,1e-9,1e-10),
                        plot=0)
{
  # create breakpoints vector (uniform grid)
  breakst=c()
  for(i in seq(1,length(time),grid_coef))
    breakst=c(breakst,time[i])

  # create the bspline basis
  base=create.bspline.basis(c(time[0],time[length(time)]),
                           breaks=breakst,norder=order)

  abscissa=time;
  Xobs0=amplitude;

  # evaluate the best penalization lambda using GCV
  gcv=numeric(length(lambda))
  for (i in 1:length(lambda))
  {
    functionalPar = fdPar(fdobj=base, Lfdobj=3, lambda=lambda[i])
    gcv[i] = smooth.basis(abscissa, Xobs0, functionalPar)$gcv
  }

  lam=lambda[which.min(gcv)]

  # functional parameter, having arguments: basis, order of the
  # derivative to be penalized, smoothing parameter
  functionalPar = fdPar(fdobj=base, Lfdobj=3, lambda=lam)

  Xss=smooth.basis(abscissa, Xobs0, functionalPar)

  # evaluation of the smooth function
  Xss0 = eval.fd(abscissa, Xss$fd, Lfd=0)

  # evaluation of the first derivative
  Xss1 = eval.fd(abscissa, Xss$fd, Lfd=1)

  # evaluation of the second derivative
  Xss2 = eval.fd(abscissa, Xss$fd, Lfd=2)

  df = Xss$df # the degrees of freedom in the smoothing curve
  GCV = Xss$gcv # the value of the GCV statistic

  if(plot){
    x11()
    plot(log10(lambda),gcv)

    x11()
  }
}

```

```

    plot( abscissa , Xobs0 , xlab="t" , ylab="observed data" )
    points( abscissa , Xss0 , type="l" , col="blue" , lwd=2)
  }

  result=list( smoothed_curve=Xss0 , der1= Xss1 , der2= Xss2 ,
              lambda=lam , GCV=GCV , df=df );
  result
}

```

Now let us move to the minima function. In this case we exploit the function `peaks` from the R package `splus2R` (available on CRAN) in order to find the local maxima of the signal, then we use the previously defined `smooth_breath` to get an estimate of the pieces first derivatives.

Input to `find_local_min` function are:

- `time`: time vector;
- `voltot`: volume vector;
- `peak_span`: parameter `span` to be passed to the `peaks` function. A peak is defined as an element in a sequence which is greater than all other elements within a window of width `span` centered at that element;
- `grid_coef`: parameter to be passed to `smooth_breath`;
- `step`: length of the subintervals on which the mean derivative is computed;
- `slope_coef`: we state that the mean derivative starts to grow if the mean derivative in a subinterval is `slope_coef` times the mean derivative in the previous subinterval. Default value is 3;
- `plot`: boolean. If 1 two plots are produced, one showing the detected local maxima and the other showing the local minima;
- `spiky_min`: logical. If TRUE then `peaks` is directly used to compute local minima. The default is FALSE.

The choice of parameter `peak_span` can be done looking at the (approximate) breathing frequency of the subject: for example, supposing a sampling frequency of 60Hz, if looking at the volume plot we see that the breaths duration is about 3s, then a good value for `peak_span` could be  $3 \times 60 = 180 \rightarrow 181$ . This means that, for each data point, the algorithm will check if it is higher than the 90 points on the left and the 90 on the right, where 90 data points correspond to 1.5s. The +1 is for the central point of the window, that is, the point being evaluated. For this reason, `peak_span` should always be an odd integer.

The output of the function is a list containing the following:

- `minima`: vector time points corresponding to minima locations;
- `minidx`: vector of positions of minima in the data vector;
- `maxidx`: vector of positions of maxima in the data vector;
- `deltas`: vector of breath lengths.

```

find_local_min=function(time, voltot, peak_span=201, grid_coef=10,
                        step=10, slope_coef=3, plot=0, spiky_min=F)
{
  # find local maxima
  localmax=which(peaks(voltot, span=peak_span)==TRUE)

  if(spiky_min=F){ # normal/low frequency breaths, with tails

    br=table_breaths(time, voltot, localmax)

    min=NULL
    minidx=NULL

    # smoothing to find derivatives between two maxima
    for(j in 1:dim(br$breaths)[2]){
      vec=which(!is.na(br$breaths[,j]))

      #smooth pieces one by one
      sm=smooth_breath(br$times[vec,j], br$breaths[vec,j],
                      grid_coef=grid_coef, plot = 0)

      timem=br$times[vec,j]
      volm=br$breaths[vec,j]
      der1m=sm$der1

      if(length(timem)>step){

        # divide the interval between two maxima in subintervals
        # and compute the mean subintervals derivatives

        amp=length(timem)%step

        breaks=seq(1, (length(timem)-amp), by=step)

        minloc=NULL

        if(length(breaks)>2){

          ms=c(mean(der1m[breaks[1]:breaks[2]]))

          for(j in 2:(length(breaks)-1)){
            med=mean(der1m[breaks[j]:breaks[j+1]])
            ms=c(ms, med)
            # look in which subinterval the mean first derivative
            starts to grow
            if(med>=0 & med>slope_coef*ms[j-1]){
              id=which.min(volm[breaks[j-1]:breaks[j+1]])
              minloc=timem[breaks[j-1]+id-1]
            }
          }
        }
      }
    }
  }
}

```

```

    }
    } else if (length(breaks)==2){
        id=which.min(volm[breaks[1]:breaks[2]])
        minloc=timem[breaks[1]+id-1]
    }

    min=c(min, minloc)
    minidx=c(minidx, which(time==minloc))
}

}

} else{ # for high-frequency breaths, without tails
# use 'peaks' directly to find the min
minidx=which(peaks(-voltot, span=peak_span)==TRUE)
min=voltot[minidx]
}

# breath lengths computation
deltaT=c();

last=time[minidx[1]];

for (i in 2:(length(min))){
    t=time[minidx[i]];
    deltaT=c(deltaT, abs(t-last));
    last=t;
}

if(plot){

x11()
plot(time, voltot, type='l', main='Local maxima')
points(time[localmax], voltot[localmax], col='red', pch=20)
x11()
plot(time, voltot, type='l', main='Local minima')
points(time[minidx], voltot[minidx], pch=20, col='red')
}

result=list(minima=min, minidx=minidx, maxidx=localmax,
            deltas=deltaT)

result
}

```

The auxiliary function `table_breaths`, given the signal and the cutting points, returns a list of two matrices, one of breaths times, and one with breaths amplitudes; then, curves are smoothed in all their four components using `smooth_breath`. Full code can be found in the Appendix.

## 4.2 Outlier detection

In a data acquisition it is frequent that “outlier breaths” may occur during the subject’s respiration. The sources of outlyingness can be several, for example:

- Coughs, yawns, talk or similar;
- Respiratory maneuvers, such as “deep breaths” or “vital capacity”, which the subject was asked to perform during the acquisition;
- Movements of the subject, causing irregularities in the track;
- Subject falling asleep during acquisition;
- Software measurement errors, like “jumps” or “spikes”;
- And many other (unknown) causes.

Outlier breaths need to be identified and removed not to spoil subsequent analysis. We will tackle this problem under different perspectives, combining state-of-the-art univariate and functional outlier detection techniques.

Although the concept of “outlier breath” may be pretty intuitive, as the two vital capacities of Figure 4.2, a rigorous definition is lacking. We formalize the outlier detection problem as follows: a breath can be outlying with respect to three characteristics (plus their combinations): time-duration, magnitude and shape. Given a curve, we state that it is an outlier if is outlying in at least one of these features.

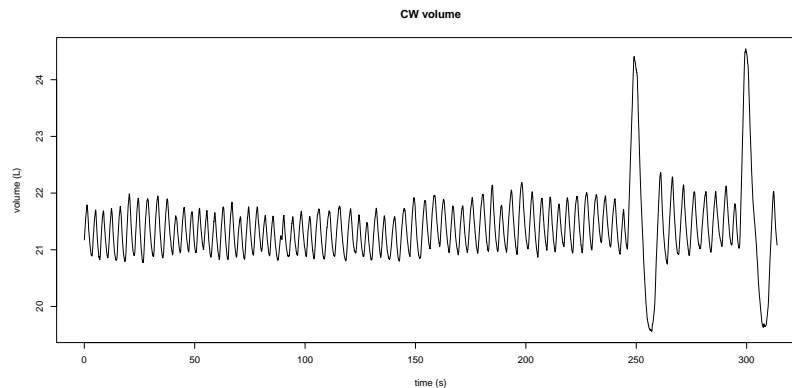


Figure 4.2: Quiet breathing acquisition with two vital capacities

### 4.2.1 Algorithm steps

Our outlier detection algorithm finds and removes outliers in sequence: first time outliers have to be removed, then magnitude ones, finally shape ones. Inverting the steps may result in less accurate output.

1. Identify time outliers with a boxplot of breath lengths;
2. Remove them and repeat until no more time outliers are found;
3. Resample each curve on 100 points;
4. Identify magnitude outliers using functional boxplot;
5. Remove them and repeat until no more magnitude outliers are found;
6. Identify shape outliers using the outliergram;
7. Remove them and repeat until no more shape outliers are found.

The resampling step is needed because functional outlier detection techniques require functional data of the same length. Removing time outliers before this step can guarantee that all the breaths have homogeneous duration, therefore improving the results.

In the multivariate functional outlier detection we weighted the 4 dimensions equally (0.25, 0.25, 0.25, 0.25). Other choices could have been, for example, giving less weight to the RCa which is usually noisier (0.3, 0.3, 0.1, 0.3) or giving more weight to the total volume (0.4, 0.2, 0.2, 0.2). However, these weights choices proved to give either the same results as using uniform weights, or they individuated less outliers than expected (subsection 4.3.2).

Notice that, due to the peculiar characteristics of our data, we need to repeat each outlier detection step (e.g. boxplot on breath lengths) until no more outliers are found before passing to the subsequent step. This is due to the fact that very often data tracks present some outliers which are so extreme that they can “mask” the presence of others. A common example is the vital capacity (VC): the maximum amount of air a person can expel from the lungs after a maximum inhalation. The VC length and amplitude are usually so big that, for example, smaller time outliers will not be detected if we just compute the boxplot once.

## Code

Implementation of functional multivariate boxplot and outliergram is provided by the R package `roahd` (Ieva, Paganoni, Romo, Tarabelloni, 2019 [11]), available on CRAN. We wrote a R function called `outlier_detection`, which combines together the different outlier detection techniques and performs algorithm iterations.

The input of the function is:

- `times`: matrix with breaths times. Times have to start from 0 for each breath;
- `smoothed_tot`: matrix of the smooth total volume breaths;
- `smoothed_rcp`: matrix of the smooth RCp breaths;
- `smoothed_rca`: matrix of the smooth RCa breaths;
- `smoothed_ab`: matrix of the smooth AB breaths;
- `plot_option`: boolean. If 1, plots of the algorithm iterations are produced;
- `weights`: either “uniform” or a vector of weights to be passed to `fbplot` and `multivariate_outliergram`.



- **range**: scalar. The inflating factor for time boxplots whiskers. Defaults to 1.5.
- **no\_iter**: boolean. If 1, outlier detection at each phase is not repeated, but just the first iteration is performed. defaults to 0.

Output is a list containing:

- **filtered.times**: matrix of breaths time vectors starting from 0, after outlier removal;
- **filtered.Vtot**: matrix of total volume vectors, after outlier removal;
- **filtered.Vrcp**: matrix of RCp volume vectors, after outlier removal;
- **filtered.Vrca**: matrix of RCa volume vectors, after outlier removal;
- **filtered.Vab**: matrix of AB volume vectors, after outlier removal;
- **outliers.idx**: vector of outlier breaths indexes;
- **time.outliers.idx**: vector of indexes of time outliers;
- **magnitude.outliers.idx**: vector of indexes of magnitude outliers;
- **shape.outliers.idx**: vector of indexes of shape outliers.

```

outlier_detection=function(times ,smoothed_tot ,smoothed_rcp ,smoothed_rca ,
                           smoothed_ab,plot_option=1, weights='uniform',
                           range=1.5, no_iter=0)
{
  filtered.times=times
  filtered.Vtot=smoothed_tot
  filtered.Vrcp=smoothed_rcp
  filtered.Vrca=smoothed_rca
  filtered.Vab=smoothed_ab

  # Check if the number of breaths is high enough to safely apply
  # the entire procedure
  breaths.are.too.few=max(else(dim(filtered.Vtot)[2]<30, 1, 0),
                          no_iter)

  if(breaths.are.too.few)
  warning('Number of breaths is too low to apply outlier detection
iteratively. Outlier checks have been performed only once')

  # Auxiliary vector to store the original indices of the outliers
  aux.idx=1:dim(smoothed_tot)[2]

  ##### outlier detection: TIME
  time.outliers.idx=NULL # vector of indices for time outliers
  found=1

```

```

deltas=c()
for(i in 1:dim(filtered.times)[2])
  deltas=c(deltas,
           filtered.times[length(which(!is.na(filtered.times[,i]))),i])

# iterate detection until no more outliers are found
while(found){
  x11()
  bp <- boxplot(deltas, range=range, plot = plot_option)
  timesout <- bp$out
  timesout <- unique(timesout)

  out <- NULL
  for(k in 1:length(deltas)){
    for(h in timesout)
      if(deltas[k]==h){
        out <- c(out,k)
        break
      }
  }

# remove outliers in time (if any)
if(length(out)>0){
  deltas=deltas[-out]

  filtered.times=filtered.times[,-out]
  filtered.Vtot=filtered.Vtot[,-out]
  filtered.Vrcp=filtered.Vrcp[,-out]
  filtered.Vrca=filtered.Vrca[,-out]
  filtered.Vab=filtered.Vab[,-out]

  time.outliers.idx=c(time.outliers.idx, aux.idx[out])
  aux.idx=aux.idx[-out]

  # Stop to 1 iteration if there are not many breaths
  if(breaths.are.too.few)
    found=0
}
else
  found=0
}

# Rescale breaths on [1:100] to apply functional outlier detection
# on magnitude and shape
scaled.Vtot=rescale_all(filtered.Vtot, filtered.times)
scaled.Vrcp=rescale_all(filtered.Vrcp, filtered.times)
scaled.Vrca=rescale_all(filtered.Vrca, filtered.times)
scaled.Vab=rescale_all(filtered.Vab, filtered.times)

```

```

##### outlier detection: MAGNITUDE
magnitude.outliers.idx=NULL # vector of indices for magnitude outliers
found=1

# iterate detection until no outliers are found

while(found){
  out1=fbplot(mfData(grid=1:100,list(t(scaled.Vtot),t(scaled.Vrcp),
    t(scaled.Vrca),t(scaled.Vab))),
    Depths=list(def='MBD',weights=weights),
    main=list('Magnitude outliers','Magnitude outliers',
    'Magnitude outliers','Magnitude outliers'),
    display=plot_option)
  idx1=out1$ID_outliers;

# remove outliers in magnitude (if any)
if(length(as.vector(idx1))>0){
  scaled.Vtot=scaled.Vtot[,-idx1]
  scaled.Vrcp=scaled.Vrcp[,-idx1]
  scaled.Vrca=scaled.Vrca[,-idx1]
  scaled.Vab=scaled.Vab[,-idx1]

  filtered.times=filtered.times[,-idx1]
  filtered.Vtot=filtered.Vtot[,-idx1]
  filtered.Vrcp=filtered.Vrcp[,-idx1]
  filtered.Vrca=filtered.Vrca[,-idx1]
  filtered.Vab=filtered.Vab[,-idx1]

  magnitude.outliers.idx=c(magnitude.outliers.idx, aux.idx[idx1])
  aux.idx=aux.idx[-idx1]

# Stop to 1 iteration if there are not many breaths
if(breaths.are.too.few)
  found=0
}
else
  found=0
}

##### outlier detection: SHAPE
shape.outliers.idx=NULL # vector of indices for shape outliers
found=1

while(found){ # iterate detection until no outliers are found
  x11()
  out2=multivariate_outliergram(mfData(grid=1:100,list(t(scaled.Vtot),
  t(scaled.Vrcp), t(scaled.Vrca),t(scaled.Vab))),
  weights=weights,
  display = plot_option)

```

```

idx2=out2$ID_outliers;
if(length(as.vector(idx2))>0){

  scaled.Vtot=scaled.Vtot[,-idx2]
  scaled.Vrcp=scaled.Vrcp[,-idx2]
  scaled.Vrca=scaled.Vrca[,-idx2]
  scaled.Vab=scaled.Vab[,-idx2]

  filtered.times=filtered.times[,-idx2]
  filtered.Vtot=filtered.Vtot[,-idx2]
  filtered.Vrcp=filtered.Vrcp[,-idx2]
  filtered.Vrca=filtered.Vrca[,-idx2]
  filtered.Vab=filtered.Vab[,-idx2]

  shape.outliers.idx=c(shape.outliers.idx, aux.idx[idx2])
                    aux.idx=aux.idx[-idx2]

  # Stop to 1 iteration if there are not many breaths
  if(breaths.are.too.few)
    found=0
}
else
  found=0
}

# Cut breaths to the longest one
max.len=0
for(j in 1:dim(filtered.times)[2]){
  len=length(which(!is.na(filtered.times[,j])))
  if(len>max.len)
    max.len=len
}

filtered.times=filtered.times[1:max.len,]
filtered.Vtot=filtered.Vtot[1:max.len,]
filtered.Vrcp=filtered.Vrcp[1:max.len,]
filtered.Vrca=filtered.Vrca[1:max.len,]
filtered.Vab=filtered.Vab[1:max.len,]

result=list( filtered.times=filtered.times,
              filtered.Vtot=filtered.Vtot,
              filtered.Vrcp=filtered.Vrcp,
              filtered.Vrca=filtered.Vrca,
              filtered.Vab=filtered.Vab,
              outliers.idx=c(time.outliers.idx, magnitude.outliers.idx,
                              shape.outliers.idx),
              time.outliers.idx=time.outliers.idx,
              magnitude.outliers.idx=magnitude.outliers.idx,
              shape.outliers.idx=shape.outliers.idx)
}

```

The auxiliary function `rescale_all` is used to resample breaths over the same number of points. Full code is available in the Appendix.

## 4.3 Validation of preprocessing techniques

### 4.3.1 Minima Examples

Let us see a couple of examples of application of the algorithm on real data, shown in Figure 4.3. In the upper panel we can see a regular quiet breathing of an healthy patient, where minima were found keeping default values.

In the middle panel instead we can see an example of “long-tail” breathing. Provided that each of the breaths lasts more than 5s, setting `peak_span=301` was sufficient to correctly position the cutting points at the end of breath tails. Other parameters were left as default.

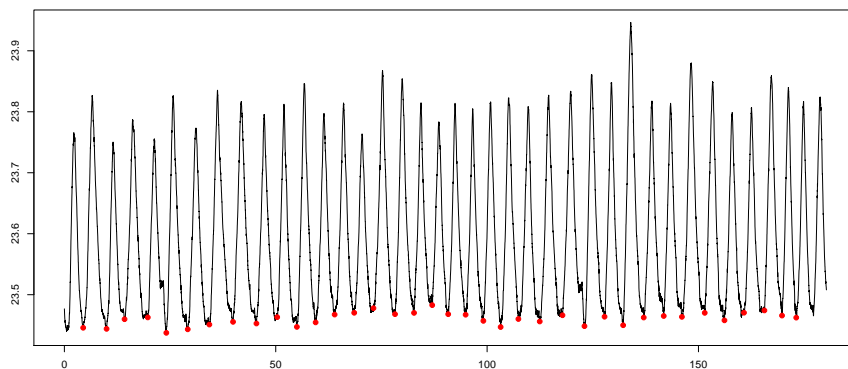
Bottom panel shows an extreme example with pauses between one breath and another. Using the same parameters as before, we were able to get a very good result.

### 4.3.2 Outlier detection - performance

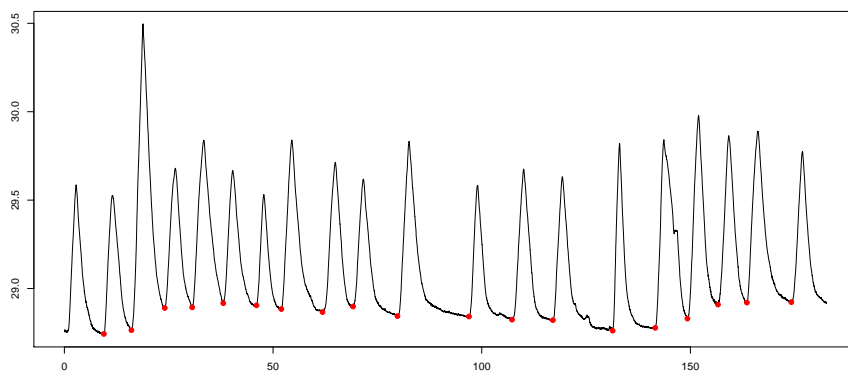
In order to validate the outlier detection procedure, we designed an experiment, consisting in a protocol for data acquisition. Four healthy subjects aged 20-25 years old were involved in two data acquisitions (one in seated and one in supine position) in which they were asked to perform 30s of some maneuvers every 30s of quiet breathing, in the following order:

- 30 s of *quiet breathing* (QB)
- 30 s *talking*
- 30 s QB
- 30 s *yawning*
- 30 s QB
- 30 s *long inspiration*
- 30 s QB
- 30 s *chaos* (a free time slot, where a subject can breathe in any unusual way)
- 30 s QB
- 30 s *long expiration*
- 30 s QB.

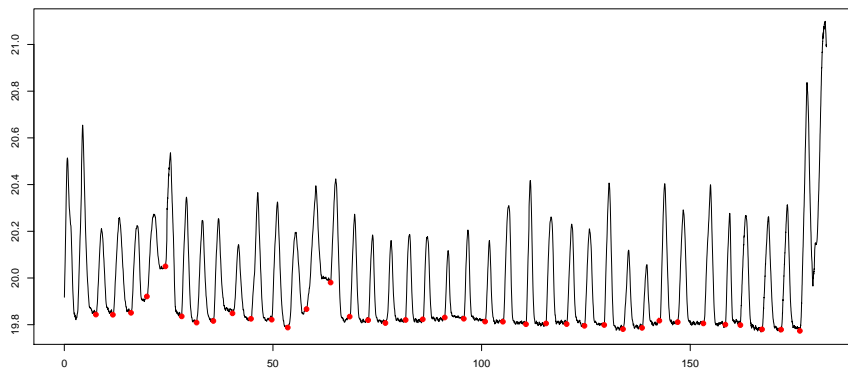
The result of an acquisition that follows this protocol is a track where perturbations are specifically localized. This allows to understand (even visually) whether the outlier detection steps is doing well. We see the chest wall volume and the detected minima of one of these acquisitions in Figure 4.4a. This data is really extreme and can work as a stress test for our outlier detection algorithm.



(a) Minima in a regular signal

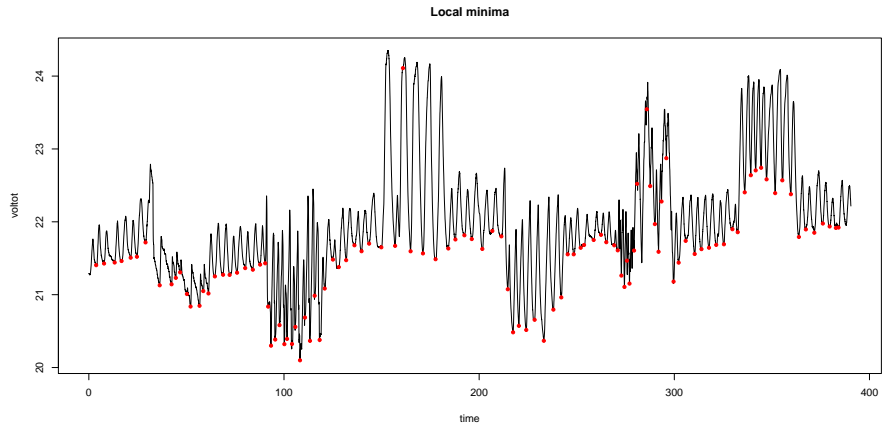


(b) Minima in a long-tail signal.

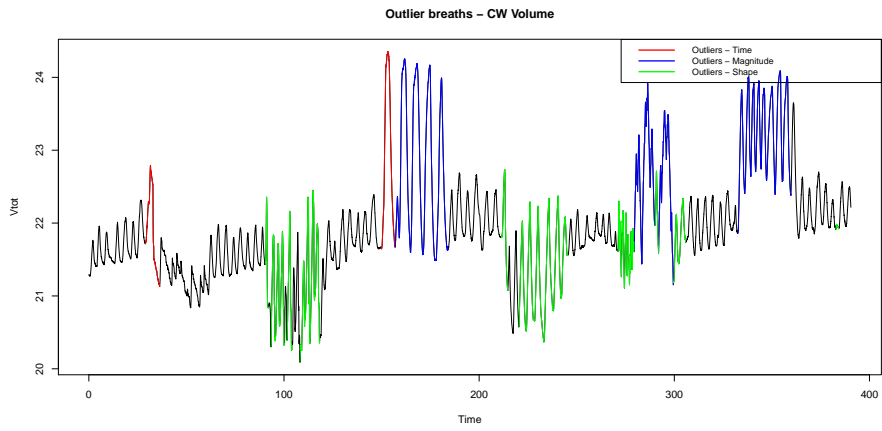


(c) Minima in a signal with pauses.

Figure 4.3: Minima examples.



(a) Chest wall volume and minima.



(b) Outliers on the CW volume.

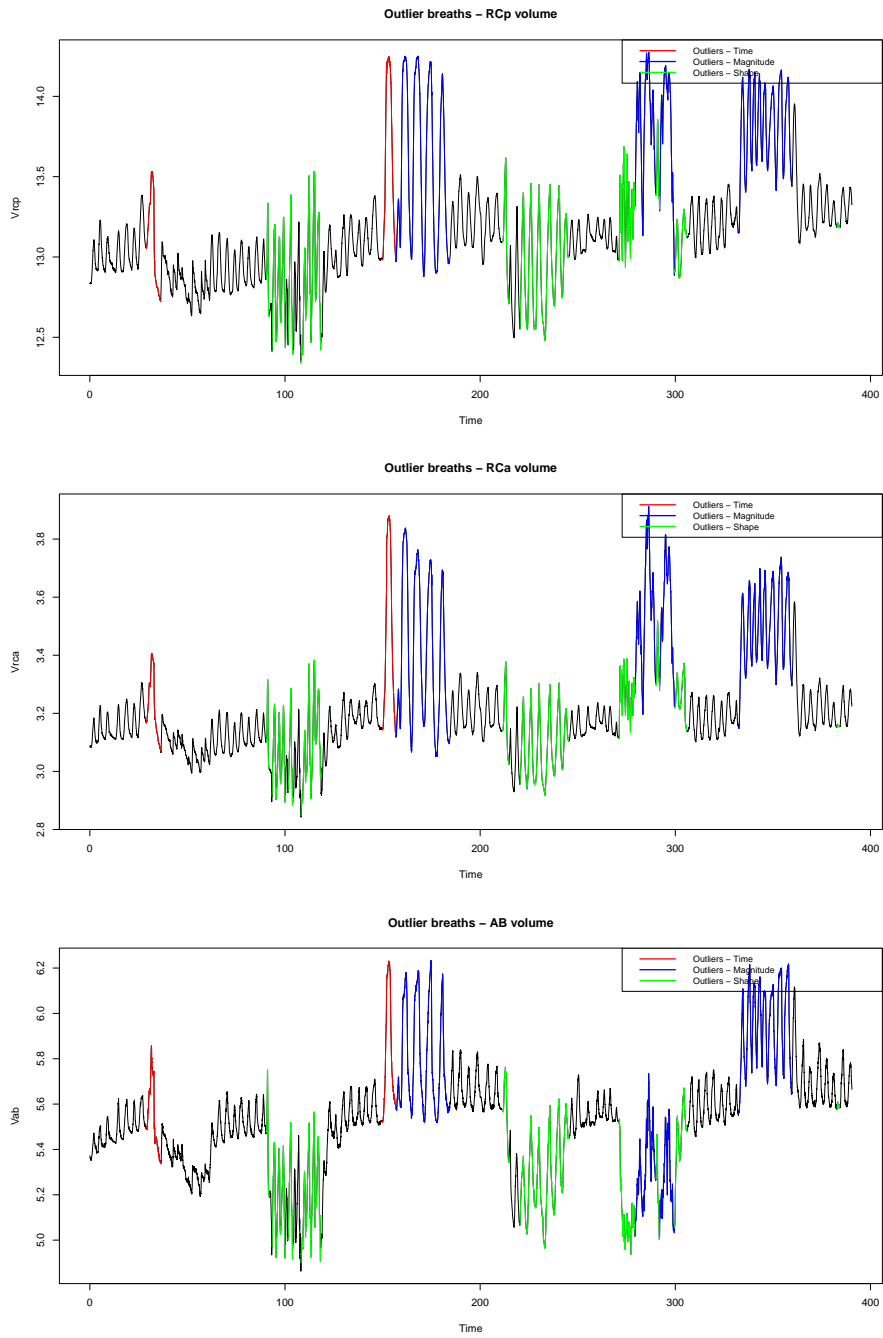


Figure 4.5: Identified outliers over compartments.



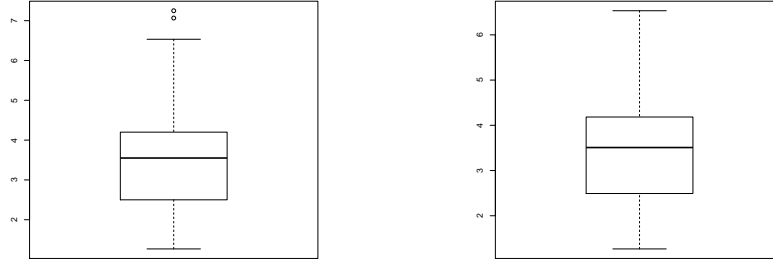


Figure 4.6: Time outlier detection iterations.

In order to understand what breaths were removed and in which phase, let us consider Figure 4.4b. As we can see, outliers are evident from the track. At first glance, we can observe that the majority of outliers were correctly removed. Only the talking part remained untouched: this is explained by the fact that talk is the framework which is most similar to quiet breathing with respect to other types of outlyingness. In Figures 4.6, 4.7 and 4.8 we can see the results of the procedure iterations in sequence (first time, then magnitude and shape).

Figure 4.6 contains the three performed iterations of magnitude outlier detection and is to be read as follows: on the left, a block of 4 functional boxplots, corresponding to the 4 signals (CW + compartments) in the first iteration; on the right, the set of 4 boxplots of the second iteration; at the bottom, the 4 boxplots of the last iteration where no more magnitude outliers were found. Each block is organized in this way: top-left, the chest wall, top-right, the RCp, bottom-left, RCa, and bottom-right the AB volume. Blue line in the middle corresponds to the median curve, the blue band corresponds to the traditional box of the boxplot (from the first to the third functional quartile), while the solid dark blue lines are the boxplot's whiskers. The functional boxplot (see subsection 3.2.2) is obtained by ranking functions from the center of the distribution outwards thanks to a suitable depth definition (multivariate MBD in this case), computing the region of 50% most central functions. The fences are obtained by inflating such region by a factor  $F = 1.5$ . Given the envelope of the functions entirely contained inside the inflated region, the data crossing these fences even for one time instant are considered outliers (they are the coloured curves).

Figure 4.8 displays the five iterations of shape outlier detection. As regards each outliergram, values of MEI and MBD on its axes are the multivariate depth measures, and each point in the graph corresponds to one breath (seen as the ensemble of its 4 components). The dark boundary on the top corresponds to the quadratic boundary in equation 3.24, while the lower boundary corresponds to the upper one inflated by a 1.5 factor (subsection 3.2.3).

We also performed some sensitivity analysis of our procedure with respect to the weights choice using as weights for multivariate functional depths the vectors  $(0.3, 0.3, 0.1, 0.3)$ , giving less weight to RCa, and  $(0.4, 0.2, 0.2, 0.2)$  giving more weight to the total volume. In both cases, the result was absolutely identical to the one obtained with uniform weights.

In order to evaluate the performance of the outlier detection algorithm, it is necessary to consider two aspects. First, true outliers should be removed by the algorithm, then, no quiet breaths should be recognized as outliers. Therefore, we define the following quantities: OUT = true outliers in the track, QB= true quiet breaths, AOUT=detected outliers, AQB=breaths who were not removed by

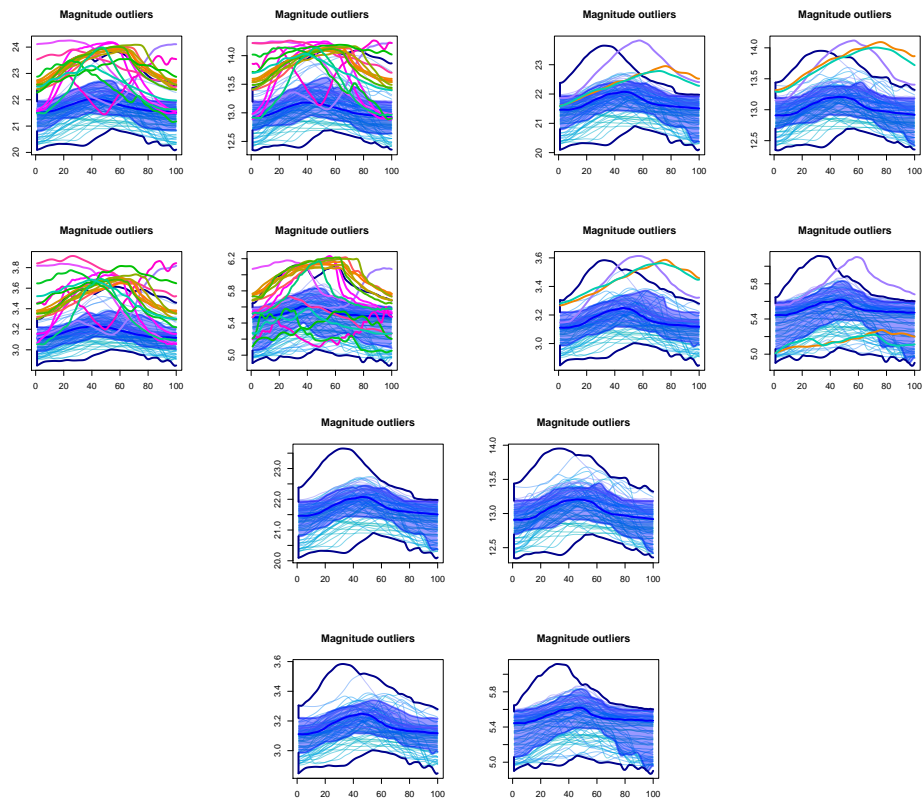


Figure 4.7: Magnitude outlier detection iterations.

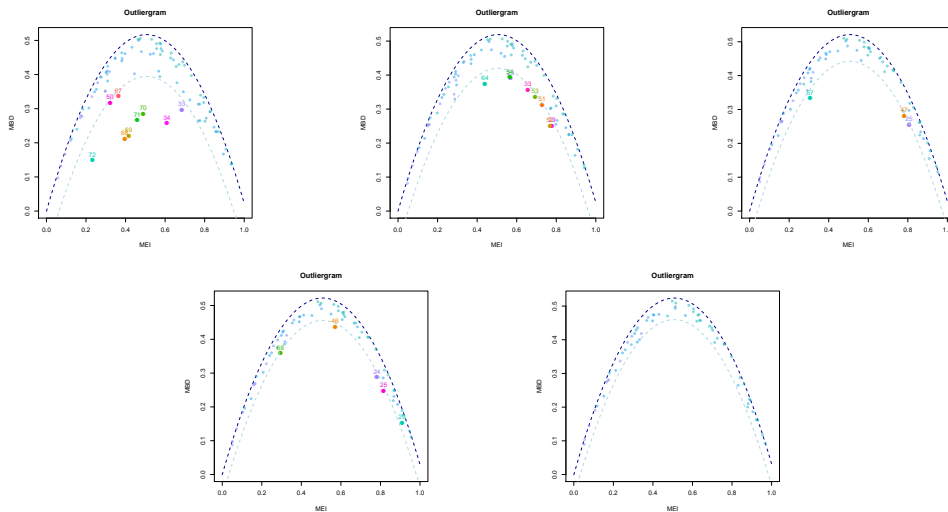


Figure 4.8: Shape outlier detection iterations.

the algorithm (leaved as quiet breathing). Results of outlier detection are summarized in Table 4.1 in terms of these quantities.

	OUT	QB
AOUT	47	0
AQB	18	44

Table 4.1: Outlier detection results - perturbed track

Among 109 breaths individuated in the minima detection, 47 were marked as outliers and 62 were leaved. 18 outliers were not detected (25% of the total number of outliers), 8 in the talking part. Nevertheless, we are quite satisfied with this result, since the most important outlier blocks were removed almost entirely. In fact, 85% of non-talk outliers were detected. Moreover, no breaths in the QB sections were removed.

Let us define a performance index: we define the *outlier error index* as  $OUT-EI = 100 \times \frac{\text{number of errors}}{\text{total number of breaths}}$ , where the errors are the number of QB recognized as outliers plus the number of outliers not detected by the algorithm. In this case, the value of OUT-EI is 16.5% for all the weight choices. We computed the OUT-EI for each of the 8 acquisitions and the three possible weight choices. We define *outlier error rate* OUT-ER as the mean of the OUT-EI in the 8 acquisitions. In Table 4.2 the values of OUT-ER for the different weight choices are shown. Uniform weights over the 4 breath components had the better performance. OUT-ER was also computed separately for the supine and seated positions, resulting in a better performance of uniform weights in both configurations. Moreover, it seems that errors in supine position are fewer. This can be explained by the fact that breath in sitting position is generally noisier (the subject can move) and has a lower abdominal contribution (abdomen is actively involved in outliers such as coughing), therefore outliers are less evident than in supine position.

	OUT-ER	OUT-ER seated	OUT-ER supine
w=(0.25, 0.25, 0.25, 0.25)	22.8%	24.2%	21.35%
w=(0.3, 0.3, 0.1, 0.3)	24.85%	25%	24.7%
w=(0.4, 0.2, 0.2, 0.2)	24.6%	25%	24.2%

Table 4.2: Outlier error rates in different configurations.

The outcome of our analysis shows that, as expected, the choice of uniform dimension weights to compute multivariate functional depths is reasonable and provides stable results. In particular, giving more weight to the total chest wall volume seems to select the same outliers; lighting the weight on the abdominal rib cage instead seems not to be an appropriate choice, since evident outliers may not be detected. This result suggests that, although the abdominal rib cage signal is the smaller in absolute value (RCa is the smaller compartment, see Figure 2.5) and in general noisier than the others, it plays an important role when we want to study the breathing patterns of a subject.

### 4.3.3 Outlier detection - Quiet Breathing with maneuvers

In this example (Figure 4.9) the subject was asked to perform some maneuvers during the data acquisition, specifically a vital capacity and deep breaths. Maneuvers are a frequent occurrence during the process of data collection, our goal is to successfully separate the quiet breathing from the rest. Referring to the results in subsection 4.3.2, uniform weights are employed.

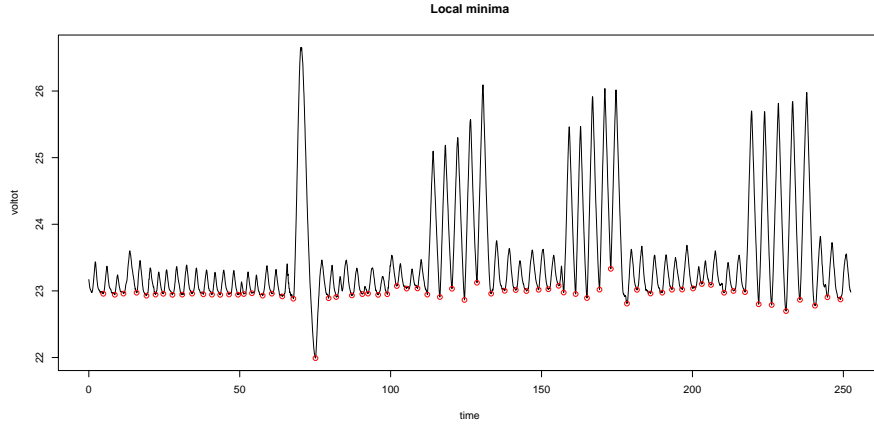
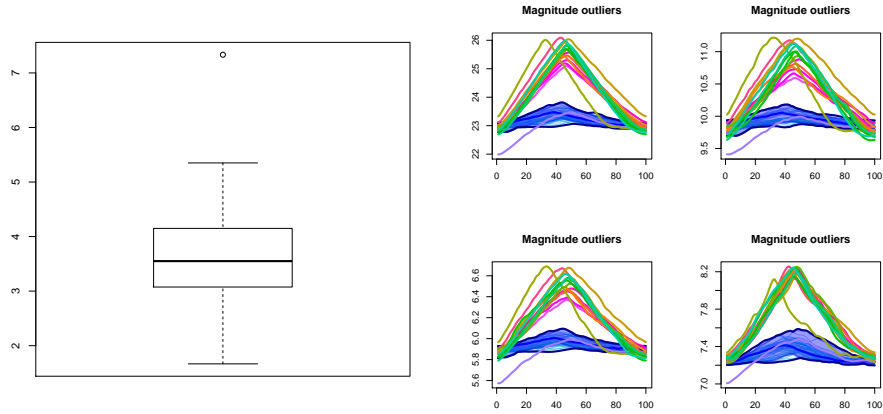


Figure 4.9: Chest wall volume with minima.

In Figure 4.12 we can see that the vital capacity is removed (correctly) as a time outlier (Figure 4.10a), the deeper breaths are removed as a result of the functional boxplot (Figure 4.10b) and finally four breaths are removed by the outliergram as shape outliers (Figure 4.11). All curves corresponding to non-quiet breathing are successfully identified and removed.



(a) Boxplot on breaths durations. (b) Multivariate functional boxplot output.

Figure 4.10: Time and magnitude outlier detection.

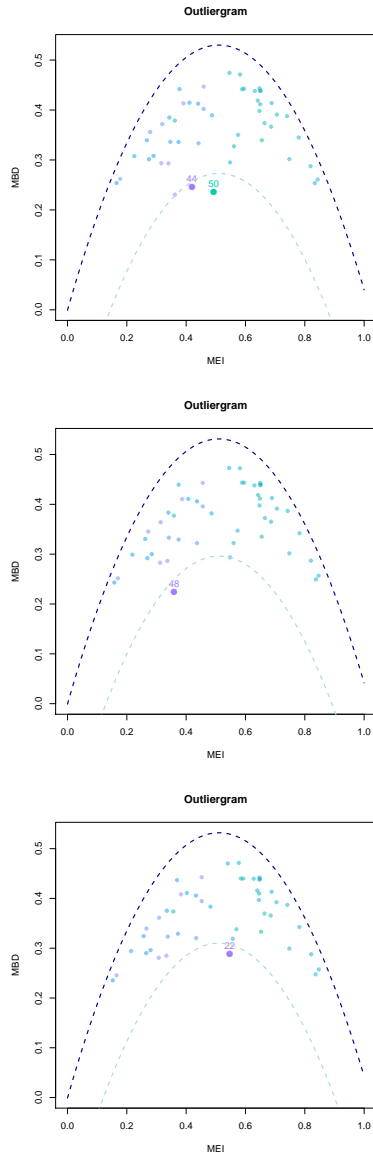


Figure 4.11: Shape outlier detection.

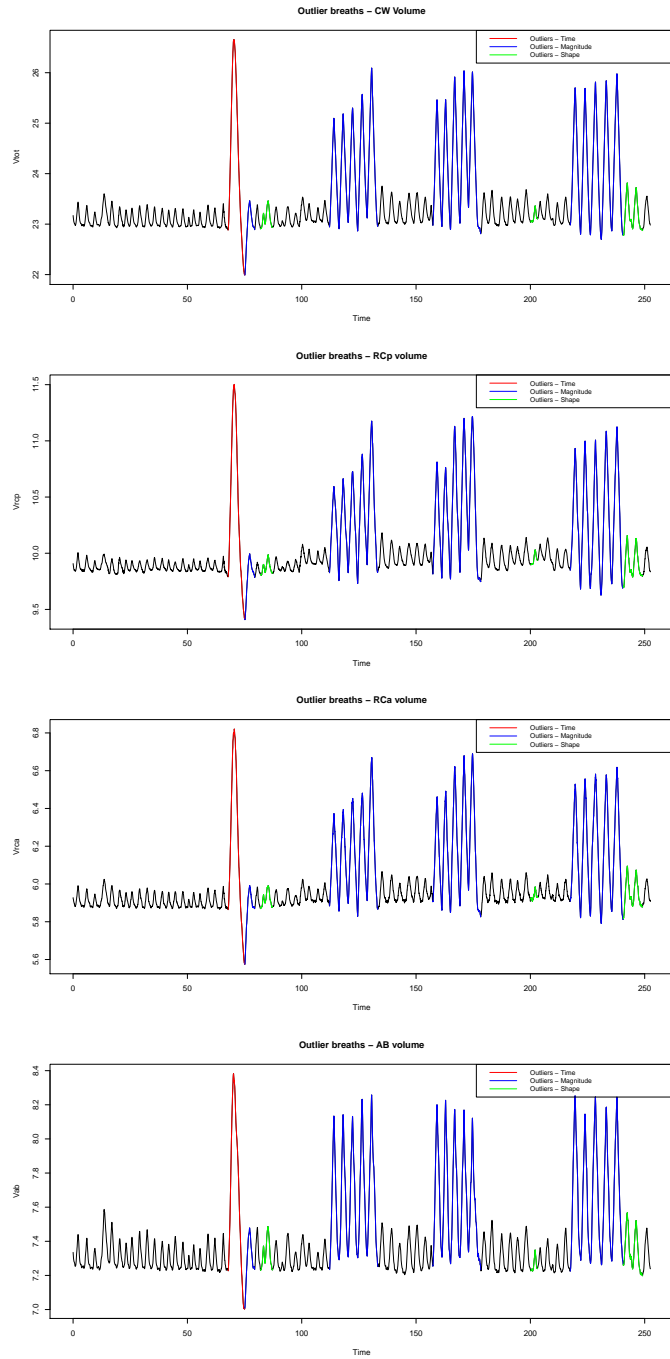
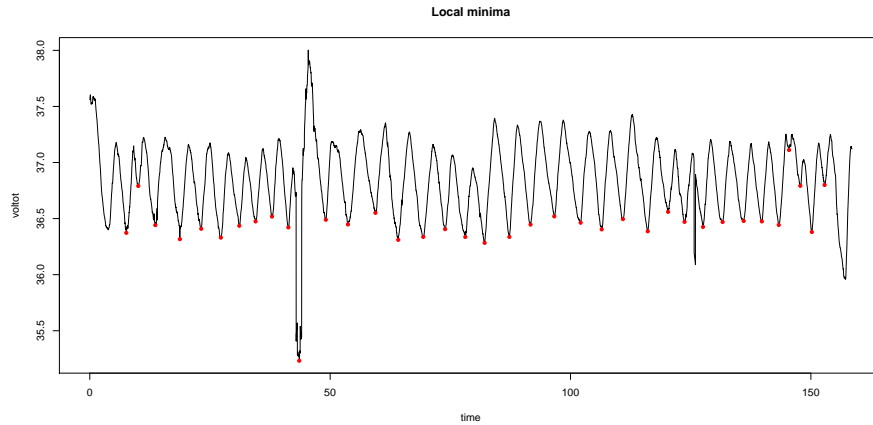


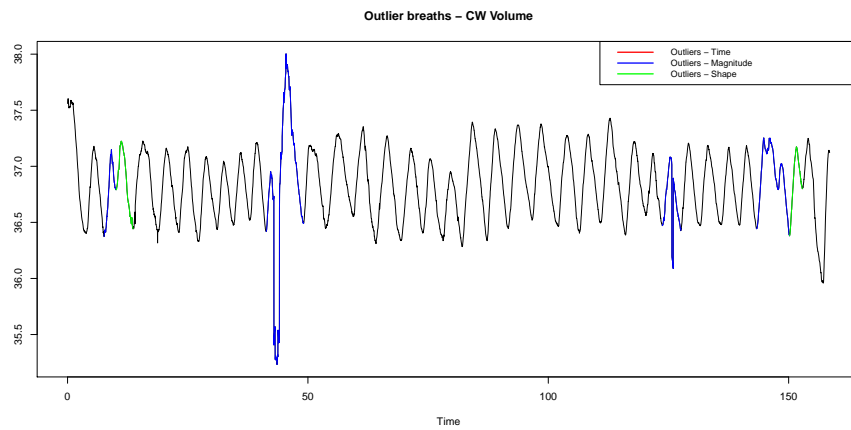
Figure 4.12: Identified outliers over chest wall and compartments.

### 4.3.4 Outlier detection - Quiet Breathing with noise

In this example we will perform outlier detection over a quiet breathing acquisition of a healthy patient aged 60 in seated position. Referring to the results in subsection 4.3.2, uniform weights are employed. Detected outliers are shown in Figures 4.13b (total volume) and 4.14 (compartments). Some evident outliers from the plots, due to errors in the measurement software, are removed as expected. Also outliers due to problematic minima are detected, while quiet breathing is untouched at it should be.



(a) Chest wall volume with minima.



(b) Outliers on the CW volume.

Figure 4.13: Quiet breathing example with measurement artifacts

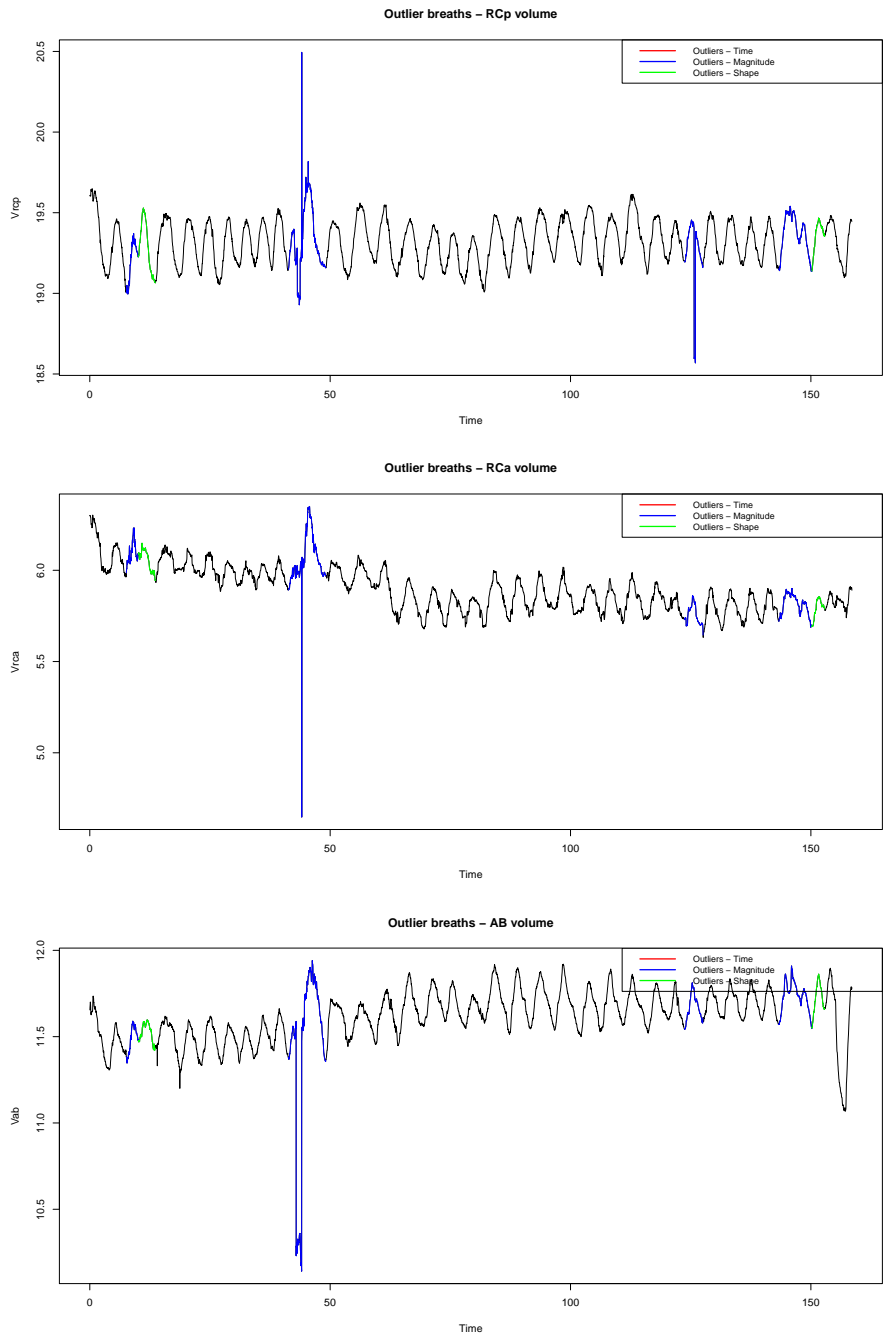


Figure 4.14: Identified outliers over compartments.



## Chapter 5

# Breathing patterns identification

Optoelectronic plethysmography has been adopted for evaluating breathing under a wide variety of circumstances both in health and disease. Each of these applications involves different variables of interest, which require different methods of analysis. In this section we propose some methodologies to analyze patterns in the OEP data in specific frameworks. Considering breaths as functions, we are able to find the *functional breathing patterns*, that is, breathing patterns of a subject can be summarized by means of representative breath curves, other than the traditional breathing pattern parameters. The functional representation is richer: curves enclose in itself the fundamental scalar quantities such as breath duration and amplitude, but also breath shape, which could not be studied otherwise.

### 5.1 Quiet breathing

Quiet breathing (QB), also known as eupnea, is a mode of breathing that occurs at rest and does not require the cognitive thought of the individual [22]. During quiet breathing, the diaphragm and external intercostals must contract. The quiet breathing of an healthy person has usually very small variations in terms of duration, amplitude and shape; thus, it makes sense to ask whether it is possible to extract a single curve as a representative of the quiet breathing of the subject.

This representative breath has to be intended in terms of all the main quantities of interest during breathing: mean volume, duration and amplitude; in other words, it should be a *real* breath. Then, we propose to apply *K-medoids with alignment* algorithm on the smoothed breath curves (after preprocessing) using affine warping functions with  $K = 1$ ; in other words, we compute the functional *median* of the curves. We will consider the median computed with respect to the  $L^2$  distance between curves normalized by the square root of the length of the common curves domain  $D$ :

$$d(c_1, c_2) = \frac{1}{\sqrt{D}} \left( \int_D (c_1(t) - c_2(t))^2 dt \right)^{1/2}, \quad (5.1)$$

where  $D = D_1 \cap D_2$ ,  $D_1$  and  $D_2$  being the time domains of  $c_1$  and  $c_2$ , respectively. In case of multidimensional data, the distance is computed as the mean of the distances between homologous dimensions.

The use of the normalizing constant is justified by the following invariance property of this measure with respect to affine warping functions : for  $c_1(t)$ ,  $t \in D_1$  and  $c_2(t)$ ,  $t \in D_2$  and

$h(t) = at + b$ ,  $a \in \mathbb{R}$ ,  $b \in \mathbb{R}^+$ , we have  $d(c_1, c_2) = d(c_1 \circ h, c_2 \circ h)$  [1]. In other words, introducing this normalization allows to use affine warping functions with  $L^2$  distance.

An example of representative breath identification is shown in subsection 5.5.1. Once localized the median breath in the dataset, it can be extracted and used, for example, in comparison with the medians coming from other subjects (chapter 6).

## Code

In our work we used the optimized K-Mean Alignment algorithm implementation by Alessandro Zito [25], which is available on GitHub in the form of an R package called `fdakmapp`. Code can be found at <https://github.com/zitale/fdakmapp> in the branch *pacs*.

Function `kmap` of `fdakmapp` package performs K-mean or K-medoid with Alignment algorithms. We use it with the following input:

- `timet`: numeric matrix [n.func X grid.size]. The abscissa values where each function is evaluated. `n.func`: number of functions in the dataset. `grid.size`: maximal number of abscissa values where each function is evaluated.
- `arr`: array [n.func X grid.size X d]. Evaluations of the set of original functions on the abscissa grid `x`. `n.func`: number of functions in the dataset. `grid.size`: maximal number of abscissa values where each function is evaluated. `d`: number of function components, i.e. each function is a d-dimensional curve.
- `n_clust`: scalar. Required number of clusters. Default value is 1. Note that if `n_clust=1` `kma` performs only alignment without clustering.
- `warping_method`: character. Type of alignment required. The implemented options are: “affine”, “dilation”, “shift” and “noalign”. In our code, we use “affine”.
- `center_method`: character. Type of clustering method to be used. Possible choices are: “mean”, “medoid” and “pseudomedoid”. Default value is “mean”. In our code, we use “medoid”.
- `similarity_method`: character. Required similarity measure. Possible choices are: “pearson”, “l2”. Default value is “pearson”. In order to compute the median, we will use “l2”.
- `comp_original_center`: boolean. If `comp_original_center=TRUE` the initial center with relative dissimilarities is computed otherwise this step is skipped.

The output of `kmap` is:

- `x.center.orig`: numeric vector of length `n_out`. Abscissa of the center computed if `comp_original_center=TRUE`.
- `y.center.orig`: numeric vector `n_out` or matrix `n_out X n_dim`. Value of the center computed if `comp_original_center = TRUE`.
- `similarity.origin`: numeric vector `n_obs` dissimilarity, similarity or distance of the original center respect the observations computed if `comp_original_center=TRUE`.
- `x.final`: matrix [n.func X grid.size]. Aligned abscissas.

- `n.clust.final`: scalar. Final number of clusters. Note that it may differ from initial number of clusters (i.e., from `n.clust`) if some clusters are found to be empty.
- `x.centers.final`: matrix [`n.clust.final` X `grid.size`]. abscissas of the final function centers.
- `y.centers.final`: matrix [`n.clust.final` X `n.out`] or array [`n.clust.final` X `n.out` x `n.dim`], contain the evaluations of the final functions centers.
- `labels`: vector of length `n.obs`. Cluster assignments.
- `similarity.final`: vector [`n.obs`]: similarities, dissimilarities or distance between each function and the center of the cluster the function is assigned to.

Further details about `kmap` can be found in the package manual.

The following code describes the steps to take after outlier detection in order to find the median breath:

```
# Outlier detection #

filtered=outlier_detection(data,min,times ,smoothed ,smoothed_rcp ,
                          smoothed_rca ,smoothed_ab, plot_option=1)

filtered.times=filtered$filtered.times
filtered.Vtot=filtered$filtered.Vtot
filtered.Vrcp=filtered$filtered.Vrcp
filtered.Vrca=filtered$filtered.Vrca
filtered.Vab=filtered$filtered.Vab
outliers=filtered$outliers.idx

# preparation of a data array to be passed to kmap

datavec = c(c(t(filtered.Vtot)), c(t(filtered.Vrcp)),
            c(t(filtered.Vrca)), c(t(filtered.Vab)))
arr=array(datavec, c(dim(filtered.Vtot)[2],dim(filtered.Vtot)[1], 4))

timet=t(filtered.times)

# compute the median using k-medoids with 1 group

med=kmap(timet, arr, n_clust=1, warping_method='affine',
         center_method='medoid', similarity_method='l2',
         comp_original_center = T)

kmap_show_results(med) # plots the result

# individuate the median index with respect to L2 distance
indm=which(med$similarity.origin==0)

# pick the median
```

```

median=cbind( filtered.times[,indm], filtered.Vtot[,indm],
               filtered.Vrcp[,indm], filtered.Vrca[, indm],
               filtered.Vab[,indm])

```

## 5.2 Exercise

Forced breathing, also known as hyperpnea, is a mode of breathing that can occur during exercise or actions that require the active manipulation of breathing, such as singing [2]. During forced breathing, inspiration and expiration both occur due to muscle contractions. In addition to the contraction of the diaphragm and intercostal muscles, other accessory muscles must also contract. During forced inspiration, muscles of the neck, including the scalenes, contract and lift the thoracic wall, increasing lung volume. During forced expiration, accessory muscles of the abdomen, including the obliques, contract, forcing abdominal organs upward against the diaphragm. This helps to push the diaphragm further into the thorax, pushing more air out. In addition, accessory muscles (primarily the internal intercostals) help to compress the rib cage, which also reduces the volume of the thoracic cavity [2]. In Figure 5.1, the evolution of breathing patterns during exercise is represented (TLC: total lung capacity; EILV: end-inspiratory lung volume; EELV: end-expiratory lung volume; RV: residual volume).

During exercise first inspiratory and expiratory flows increase (i.e., the breath amplitude); contemporarily, also the breathing frequency increases. Then, when the maximal effort is reached, there is an increase also in the mean value of the volume, that is an upward shift of the curves. However, this is only true for non-athlete healthy subjects: In fact, in highly fit endurance athletes the pressure produced by inspiratory muscles can approach the maximum and expiratory pressures are increased to levels at which dynamic compression of the airways determines expiratory flow limitation. This phenomenon also frequently occurs in elderly subjects due to the age-related loss of lung elastic recoil, and is a common feature of patients with chronic obstructive pulmonary disease (COPD), not only during exercise but also at rest in the most severe cases. When expiratory flow is limited, end-expiratory lung volume has to be increased to allow for further increases of flow (Incomplete expiration prior to the initiation of the next breath causes progressive air trapping). In other words, expiratory flow limitation causes the so called “dynamic hyperinflation”. This means that the increase in mean volume can occur at the beginning of the effort [2]. In other pathologies, different trends in the mean volume (and in amplitude) can also be observed during exercise.

For these reasons, OEP data involving exercise contain different breathing patterns, according to the level of effort (Figure 5.1). Therefore, studying these data were are interested in finding breath clusters. Since in this case the mean value of the volume is an important quantity of interest, we propose to apply K-medoids with Alignment algorithm using affine warping functions and the normalized  $L^2$  distance in formula 5.1 as dissimilarity measure. We use K-medoids instead of K-means since the group centroids, which we consider the representatives of the patient’s breathing patterns, have to be real breaths, taken with their duration and amplitude. Results of a real case study are shown in subsection 5.5.3.

### Code

The code for clustering is very similar to the one used to compute the median; indeed, we can use `kmap` imposing `n_clust` equal to  $K$ ,  $K$  being the number of clusters.

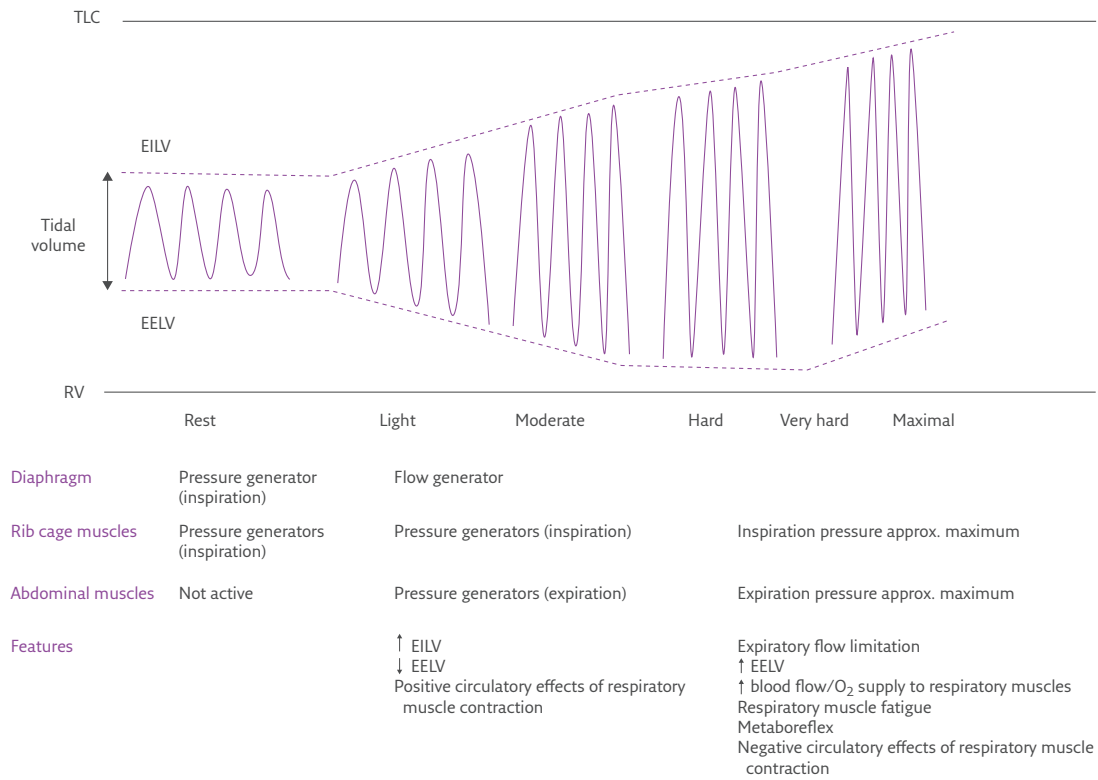


Figure 5.1: The main features of respiratory muscle action during exercise.

A common problem is how to fix  $K$ . In fact, if  $K$  is too small, we are missing some clusters; if instead it is too big, we are overfitting our data. In our code we chose to look for the elbow of the mean similarities plot: we run K-medoids with Alignment for different values of  $K$  and we compute the mean of the final within-cluster similarities (dissimilarities in our case), then we plot these values for each  $K$ . We take  $K$  as the number of clusters for which the mean (dis)similarity does not improve much taking higher values, that is, an elbow in the plot.

```
## ... ..

# K-mean alignment settings
distance='l2'
warp='affine'
center='medoid'

# choose number of clusters —> elbow
K=4
checksim=numeric(K)
```

```

for(j in 1:K){
    check=kmap(timet, arr, n_clust=j, warping_method=warp,
               center_method=center, similarity_method=distance,
               comp_original_center = T)

    checksim[j]=mean(check$similarity.final)
}

x11()
plot(checksim, type='l', xlab='Cluster labels',
      ylab='Mean dissimilarities',
      main=paste0('Mean Dissimilarity - ',warp,sep=''))
points(checksim, pch=1, col='blue',lwd=2)

# Selected number of clusters
nclust=2 # if the elbow is at K=2

res=kmap(timet, arr, n_clust=nclust, warping_method=warp,
         center_method=center, similarity_method=distance,
         comp_original_center = T)

kmap_show_results(res)

```

### 5.3 Mechanical ventilation

Mechanical ventilation is a life support treatment. A mechanical ventilator (ventilator, respirator, or breathing machine) is a machine that helps people breathe when they are not able to breathe enough on their own, blowing gas (air plus oxygen as needed) into a person's lungs. It can help a person by doing all of the breathing or just assisting the person's breathing. The ventilator can also provide what is called positive end expiratory pressure (PEEP), air pressure in the lungs which is above the atmospheric pressure at the end of expiration. Pressure support ventilation (PSV) applies PEEP to hold the lungs open so that the air sacs do not collapse [5].

Most patients who need support from a ventilator because of a severe illness are cared for in a hospital's intensive care unit (ICU). Patients suffering from neuromuscular diseases such as Duchenne Muscular Dystrophy (DMD) undergo a progressive weakening of muscles, included the ones of respiration, therefore need a continuous ventilation support [4] [14].

OEP data can be used to understand the effects of mechanical ventilation on a patient's breath. For example, it is interesting to see how much the mean lung volume changes after the application of external PEEP. Moreover, the action of the ventilator (which replaces the work of the patient's respiratory muscles) has also an action over the breathing dynamics. DMD patients in advanced stages have a compromised diaphragm, which is not able to act coordinately with the other (less compromised) respiratory muscles. As a result, the AB volume measured with OEP presents a phase shift with respect to the rib cage. The abdominal behaviour is often paradoxical, that is, with a phase shift of 180 degrees. Among the effects of mechanical ventilation is also the removal of these impairments, which results in an "healthy" breathing pattern.

In order to study these phenomena, we have to take into account both breath mean volume and breath shape, which are influenced by the action of the ventilator. We will proceed as in the exercise case, applying K-medoids with Alignment algorithm with the  $L^2$  distance as dissimilarity measure and affine warping functions.

Code is the same as in the exercise case. Results of a real case study are shown in subsection 5.5.4.

## 5.4 Shape patterns during quiet breathing

In some applications, one is more interested in shape patterns variation during breathing rather than changes in the mean volume. During the quiet breathing of unhealthy patients, alternating patterns can occur. In other cases, data show a pattern transition during time that we would like to capture.

In this setting, we would like to find clusters which depend on the breaths shape rather than their mean volume or their amplitude. We will thus apply the K-medoids with Alignment algorithm on the breaths derivatives, using affine warping functions. The chosen dissimilarity (semi-metric) is then

$$d(c_1, c_2) = \left\| \frac{c'_1}{\|c'_1\|} - \frac{c'_2}{\|c'_2\|} \right\|, \quad (5.2)$$

which induces the similarity index

$$\rho(c_1, c_2) = \frac{\langle c'_1(t), c'_2(t) \rangle}{\|c'_1\|^2 \|c'_2\|^2}, \quad (5.3)$$

where  $\langle \cdot, \cdot \rangle$  is the scalar product in  $L^2$  and  $\|\cdot\|$  the induced norm. The similarity index  $\rho$  is called *normalized Pearson correlation* and corresponds to the cosine of the angle formed by  $c'_1$  and  $c'_2$ . For multidimensional curves (4-dimensional in our case),  $\rho$  is computed as the average of the 1-dimensional similarity indexes, that is, the average of the cosines of the angles between the derivatives of homologous components of  $c_1$  and  $c_2$  [20]. An example of shape classification is shown in subsection 5.5.5.

### Code

Code is analogous to the  $L^2$  version, but in this case we have to use `kmap` on the breaths *derivatives* with `similarity_method = 'pearson'`. Moreover, the function finds clusters by minimizing the *similarity* index with changed sign, therefore we have to change the sign back if we want to plot the similarity indexes to find the elbow.

```
filtered=outlier_detection(times,smoothed,smoothed_rcp,smoothed_rca,
                           smoothed_ab,plot_option=1)

outliers=filtered$outliers.idx

filtered.times=filtered$filtered.times
filtered.Vtot1=smoothed1[,-outliers]
filtered.Vrcp1=smoothed_rcp1[,-outliers]
```

```

filtered.Vrcal=smoothed_rca1[,-outliers]
filtered.Vabl=smoothed_ab1[,-outliers]

filtered.Vtot1=filtered.Vtot1[1:dim(filtered.times)[1],]
filtered.Vrcp1=filtered.Vrcp1[1:dim(filtered.times)[1],]
filtered.Vrcal=filtered.Vrcal[1:dim(filtered.times)[1],]
filtered.Vabl=filtered.Vabl[1:dim(filtered.times)[1],]

datavec = c(c(t(filtered.Vtot1)), c(t(filtered.Vrcp1)),
            c(t(filtered.Vrcal)),c(t(filtered.Vabl)))
arr=array(datavec, c(dim(filtered.Vtot1)[2],dim(filtered.Vtot1)[1], 4))

timet=t(filtered.times)
original.times=volumes.tot$times[,-outliers] # for graphics
original.times=original.times[1:dim(filtered.times)[1],]

# K-mean alignment settings
distance='pearson'
warp='affine'
center='medoid'

# look for similarity elbow
K=4
checksim=numeric(K)

for(j in 1:K){

    check=kmap(timet, arr, n_clust=j, warping_method=warp,
              center_method=center, similarity_method=distance,
              comp_original_center = T)

    checksim[j]=mean(check$similarity.final)
}

x11()
plot(-checksim, type='l', xlab='Cluster labels', ylab='Mean similarities',
     main=paste0('Mean Similarity - ',warp,sep=''))
points(-checksim, pch=1, col='blue',lwd=2)

# Selected number of clusters
nclust=2

res=kmap(timet, arr, n_clust=nclust, warping_method=warp,
        center_method=center, similarity_method=distance,
        comp_original_center = T)

kmap_show_results(res)

```



## 5.5 Case studies and applications

### 5.5.1 Quiet breathing example

In Figure 5.2 we can observe the quiet breathing of a healthy patient in supine position, with also two vital capacities. After outlier removal ( Figure 5.3) breaths were aligned and the medoid was computed, as shown in Figure 5.4, and the corresponding breath was picked. Figure 5.5 displays the four components of the median breath in comparison (each curve has been translated to start from 0). This kind of plot is very common in the literature of breathing patterns studies, since it synthetically describes the contribution of each compartment to the total [13]. However, the displayed breath is one chosen by the analyst, therefore it has got mainly a *qualitative* value. Our method provides a better information instead: we are using the functional median as a summary statistic of the respiration mode, which has also a *quantitative* value in terms of duration and amplitude.

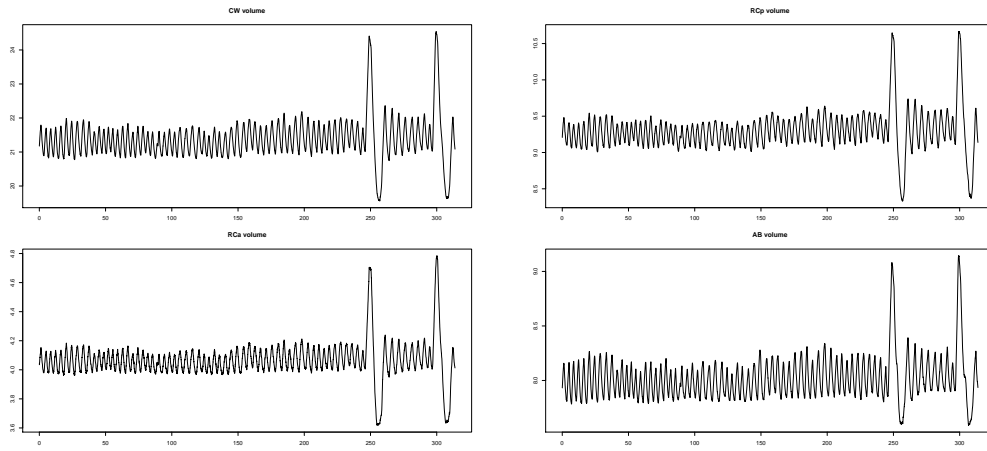


Figure 5.2: Quiet breathing example.

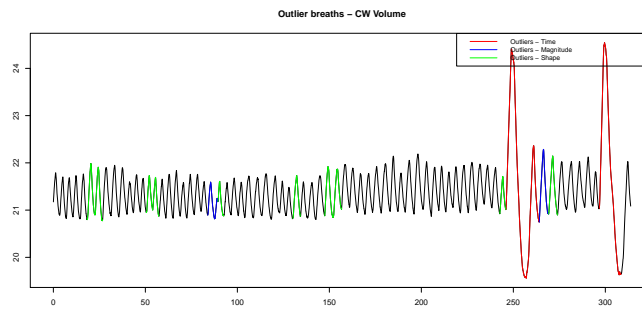


Figure 5.3: Outliers on total volume.

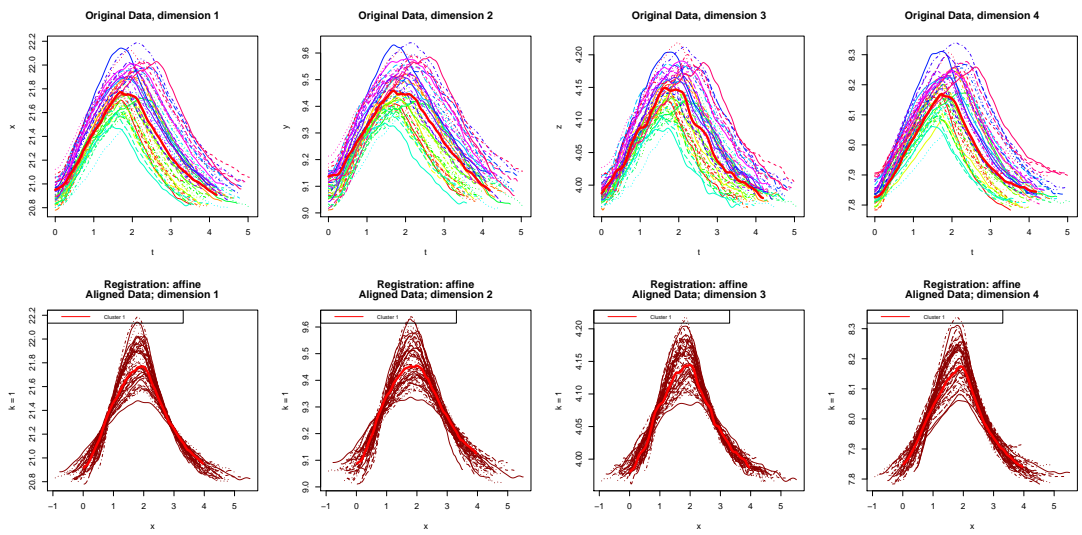


Figure 5.4: Output of kmap function.

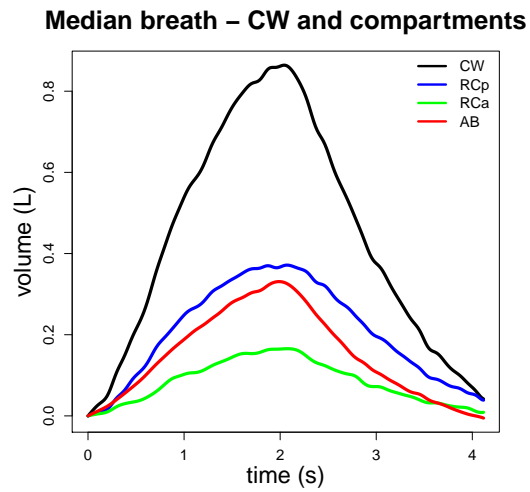


Figure 5.5: Median breath components in comparison.

## 5.5.2 Case report - Cystic Fibrosis

This case study involves data acquisition from a woman suffering from Cystic Fibrosis (CF). CF is a genetic disorder that affects mostly the lungs, but also the pancreas, liver, kidneys, and intestine. In people with CF, mutations in the cystic fibrosis transmembrane conductance regulator (CFTR) gene cause the CFTR protein to become dysfunctional. When the protein is not working correctly, it's unable to help move chloride (a component of salt) to the cell surface. Without the chloride to attract water to the cell surface, the mucus in various organs becomes thick and sticky.

In the lungs, the mucus clogs the airways and traps germs, like bacteria, leading to infections, inflammation, respiratory failure, and other complications [7].

From an inspection of the breathing tracks in Figure 5.6, we can notice spikes in the signal, corresponding to coughs. In particular, after the second cough there seem to be a sudden decrease in the total volume. We expect this behavior to be influential in the clustering step.

### Outlier detection

The results of the full outlier detection is shown in Figure 5.7. As expected, the coughs are all removed as magnitude outliers. The first breaths after the change of the absolute value of the volume are identified as time outliers.

### Patterns

Since there seem to be a step in amplitude, the chosen functional distance in this case has been the  $L^2$  one. The elbow in the mean similarities plot indicated 2 clusters.

The algorithm identified two clusters, collecting all the breaths before and after the second cough respectively (Figure 5.8 and 5.9). The two groups are different in terms of absolute value of the volume in the chest wall, in particular in the RCa compartment. Interestingly, they also differ in terms of shape in the RCp compartment, with cluster 1 presenting a paradox (the volume decreases during the inspiration phase). In Figure 5.10 we can observe the the significant difference in absolute volume of the two clusters and the shift in phase in the pulmonary rib cage.

### Discussion

Cluster 1 represents the breathing pattern before the cough, which is characterized by a higher mean value and a significantly paradoxical behaviour in terms of RCp volume (see Figure 5.11). Cluster 2 encloses the breathing pattern afterwards, which is characterized by a decrease in the mean value of RCa, while the paradox disappears. It is likely that at the beginning of the acquisition the patient had an obstruction in the lungs, which was removed with the cough, resulting in a change of her breathing activity. This case is a borderline example about the selection of the distance on top of the grouping structure. We used the  $L^2$  distance for the clustering, which puts more weight on the absolute value. However, except for RCp compartment, the shape between the groups seems quite similar, so are the two breathing patterns actually different?

To answer this question, we repeated the clustering step using the Pearson distance on the first derivatives of the signal. Again, the similarity index suggested 2 clusters. The results are show in Figure 5.12 and Figure 5.13 and are substantially equal to those obtained previously.

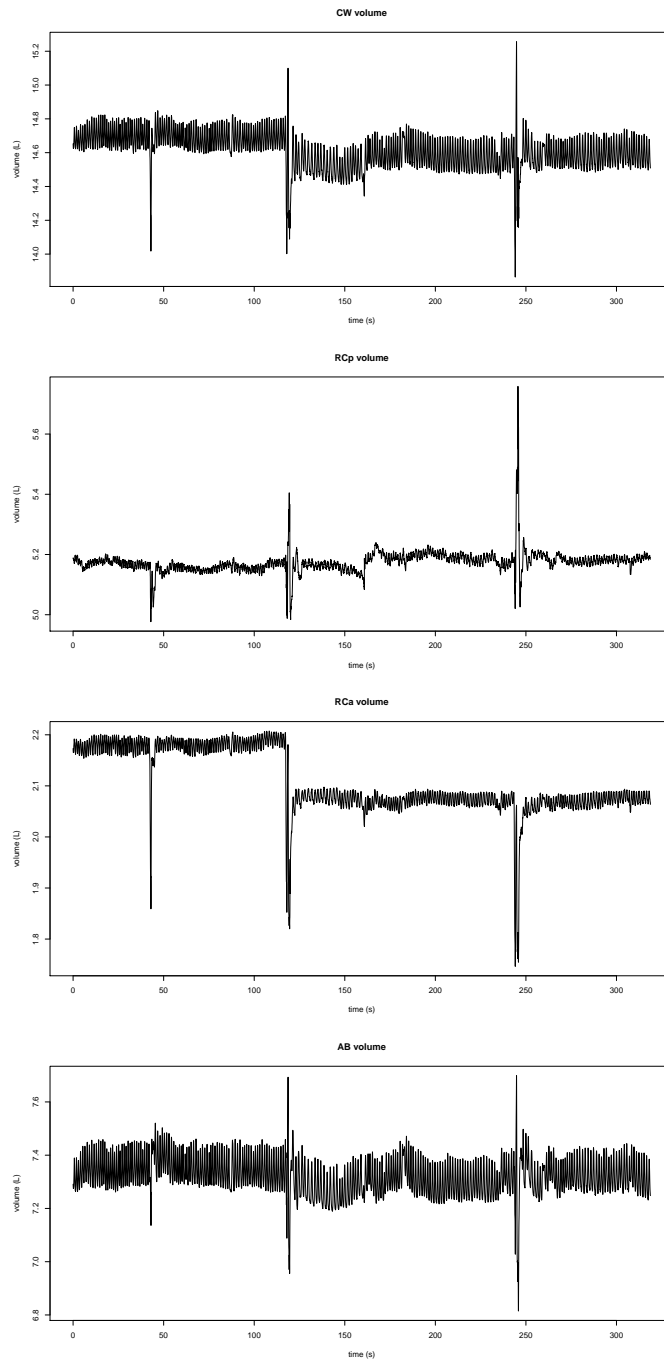


Figure 5.6: Breathing tracks - Chest wall and compartments

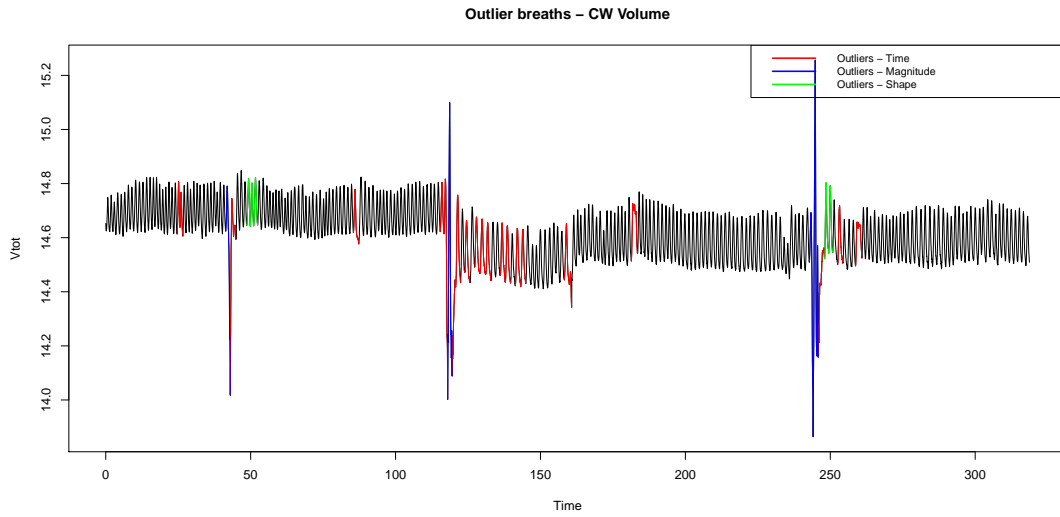


Figure 5.7: Outlier detection results.

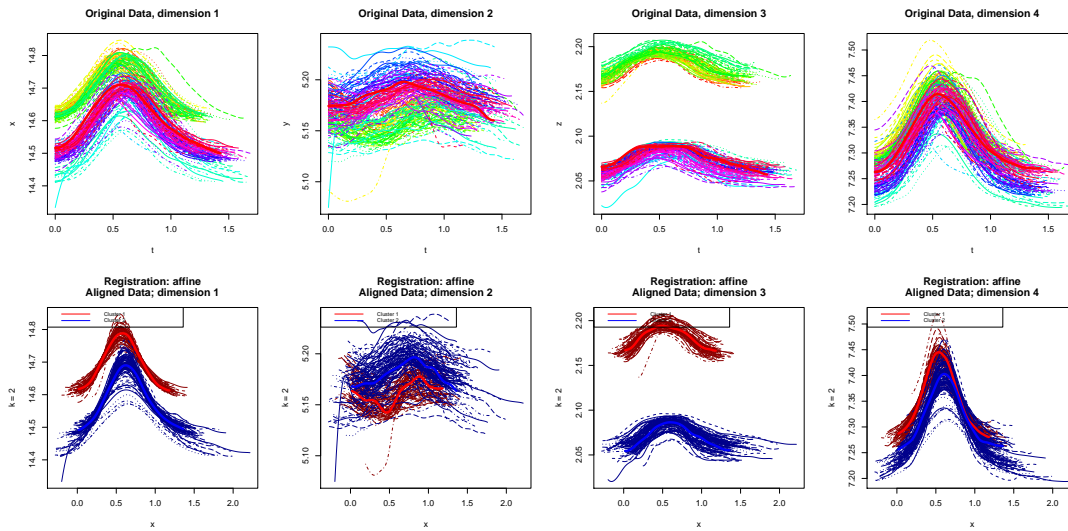


Figure 5.8: Multivariate kma results -  $L^2$  distance

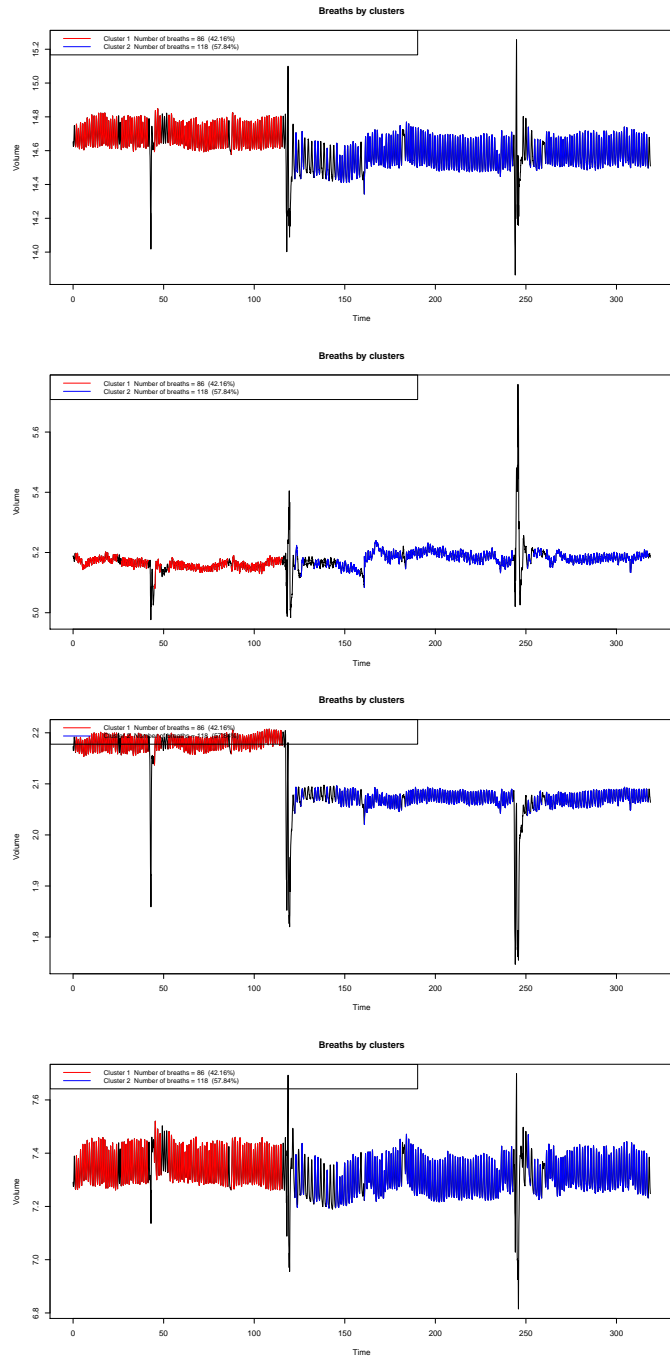


Figure 5.9: Clustering on data volumes.

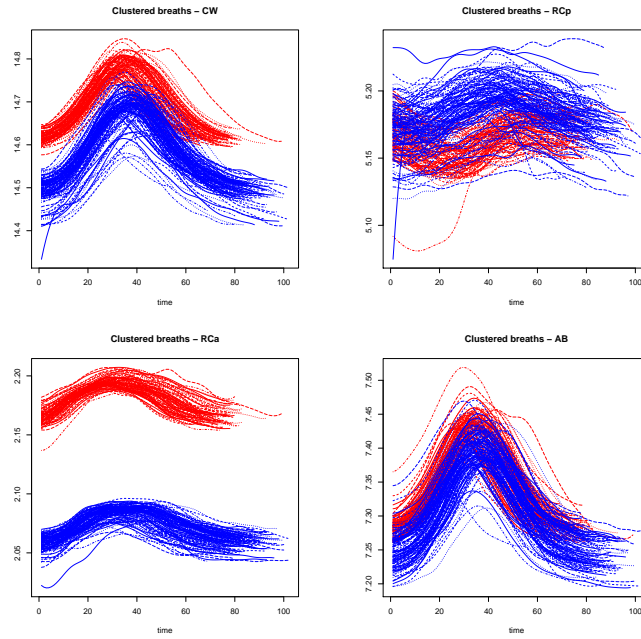


Figure 5.10: Clustered breaths

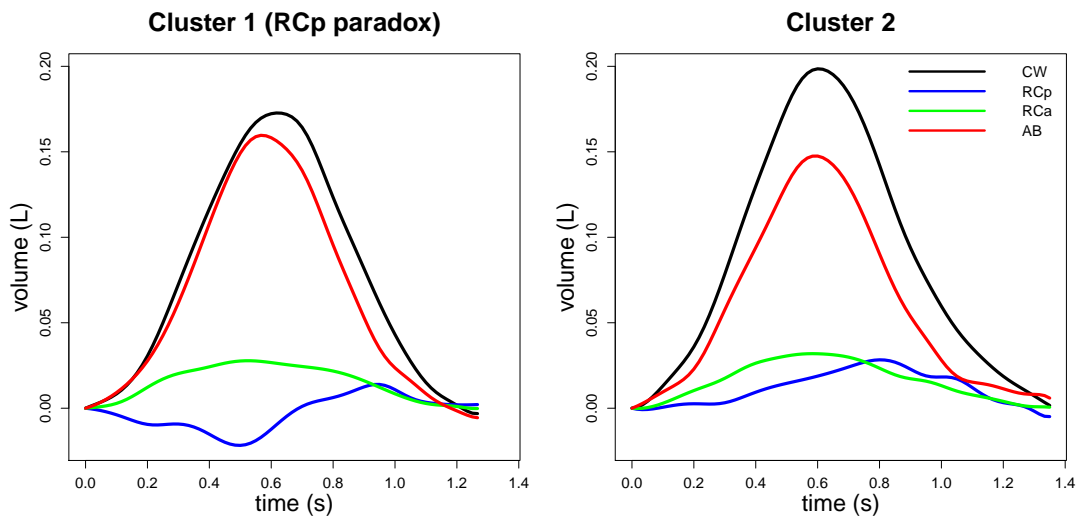


Figure 5.11: Median breaths of clusters -  $L^2$  distance

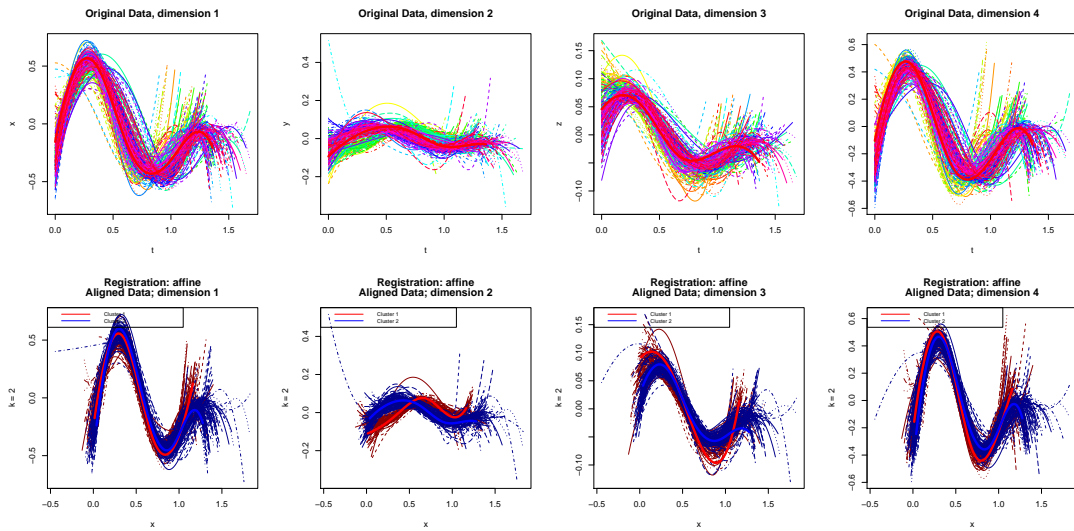


Figure 5.12: Multivariate kma results - Pearson distance

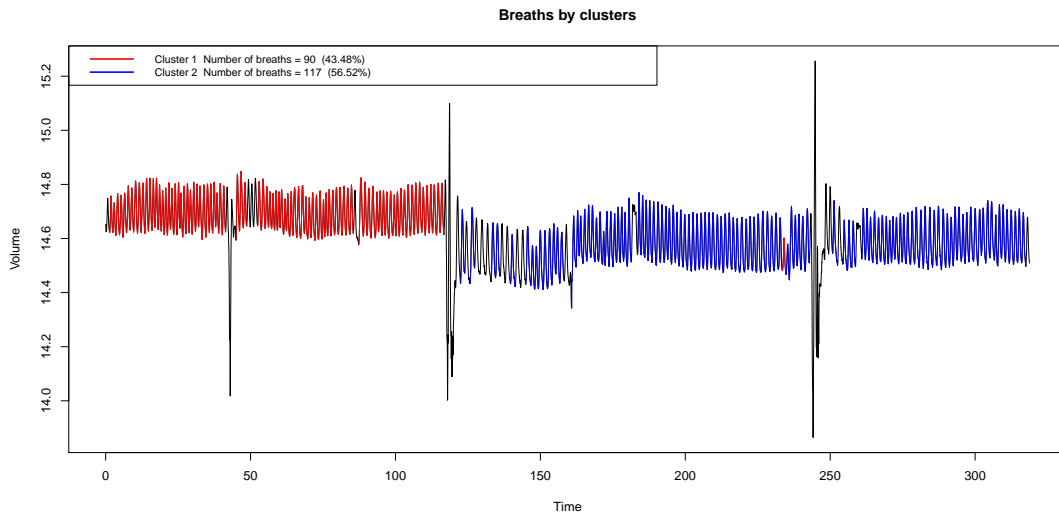


Figure 5.13: Clustering on total volume - Pearson distance



### 5.5.3 Case report - Exercise

In this case report we will analyse data from a healthy subject aged 16 sitting on a cycle ergometer. The data acquisition comprehends 180s of quiet breathing with a vital capacity, followed by exercise at growing effort levels. The bike resistance was increased in a stepwise manner for 5 times, and removed in the end for subject recovery. In Figure 5.14 it is possible to see the variations in breathing pattern as the required effort changes. As expected, the behaviour is similar to the theoretical one shown in Figure 5.1. Flat parts in the plot correspond to points in which the data recording was stopped to save the partial result and have to be considered as missing data.

#### Outlier detection

Since quiet breathing contains few curves with respect to the rest of the track, and those breaths have a higher duration, it was decided to be slightly more permissive with respect to time outliers and the inflating factor for time boxplots was set to 2 instead of 1.5. The vital capacity has been marked as outlier and most quiet breathing has been preserved, flat parts were removed together with the spike due to a measurement error (see Figure 5.15).

#### Patterns

Variables of interest in forced breathing are linked with absolute values: breath amplitude, End-Expiratory/End Inspiratory Lung Volume (EELV/EILV) and the trend in mean volume, therefore the  $L^2$  distance was chosen. The elbow in mean dissimilarity plot indicated 5 clusters. Results are shown in Figures 5.16, 5.17, 5.18 and 5.19.

Cluster 4 encloses the quiet breathing and presents a higher rib cage contribution with respect to the abdominal one, which is typical of the sitting position. Cluster 5 encloses the initial effort and is characterised by an increase in breath amplitude in all compartments, but above all has a greater abdominal contribution with respect to quiet breathing. The increase in effort corresponds to higher amplitude and a slight grow in the mean volume, and all compartments are actively involved in ventilation. Finally, in the recovery phase, the mean volume goes back to its original value, while amplitude gradually decreases.

#### Discussion

Classification proved to be able to capture the main variations of breathing pattern during exercise. It is well-known that breathing pattern changes during exercise in reaction to increasing effort levels, and different subjects react in their own way to increase ventilation: some increase amplitude first, and then breathing frequency; others start by increasing frequency and then amplitude; others proceed gradually augmenting both amplitude and frequency [2]. Clusters were correctly placed in sequence over the data acquisition, and separate different effort stages as expected (Figure 5.19). The observation of group centroids in comparison (Figure 5.18) allows for a better comprehension of breathing dynamics: in particular, three breathing modes during exercise were found, independently of the bike resistance level: the subject started increasing amplitude (cluster 5), then breathing frequency, finally frequency and EELV (cluster 3 and 1). This is coherent with the theoretical behaviour of breathing pattern during maximal effort (Figure 5.1). Finally, recovery (cluster 2) was not immediate, although the mean value of chest wall volume dropped immediately to the original one. It is also possible to observe a gradual transition in cluster 2 towards quiet breathing (Figure 5.17).

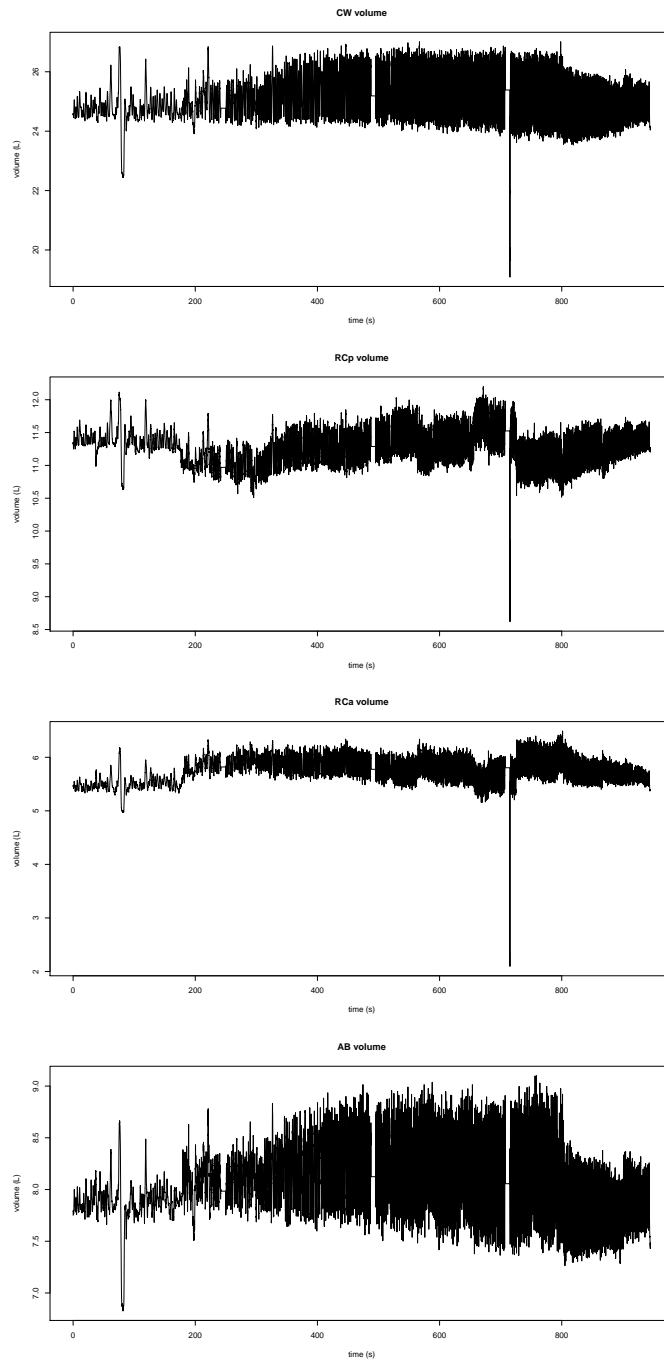


Figure 5.14: Data acquisition during exercise.

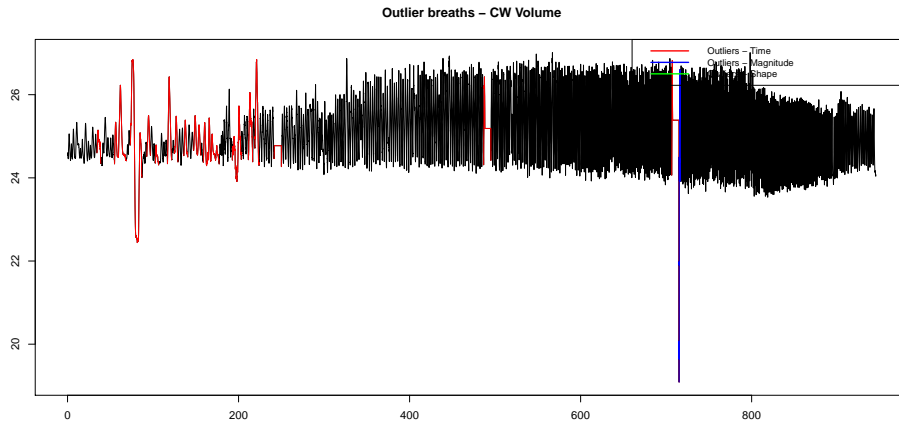


Figure 5.15: Outlier detection results.

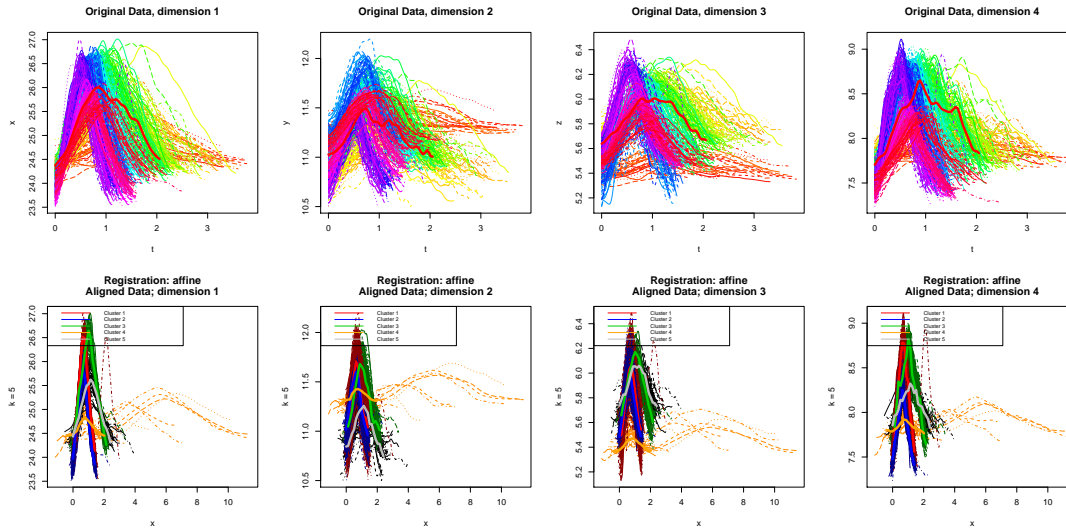


Figure 5.16: Multivariate kma results -  $L^2$  distance

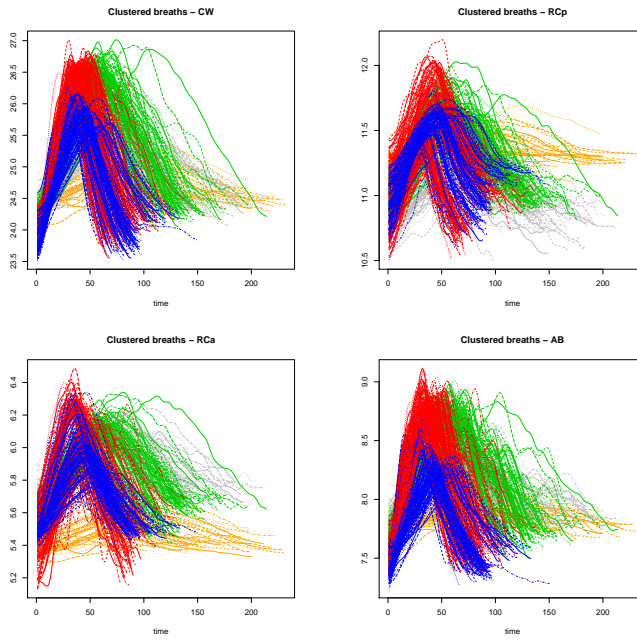


Figure 5.17: Clustered breaths.

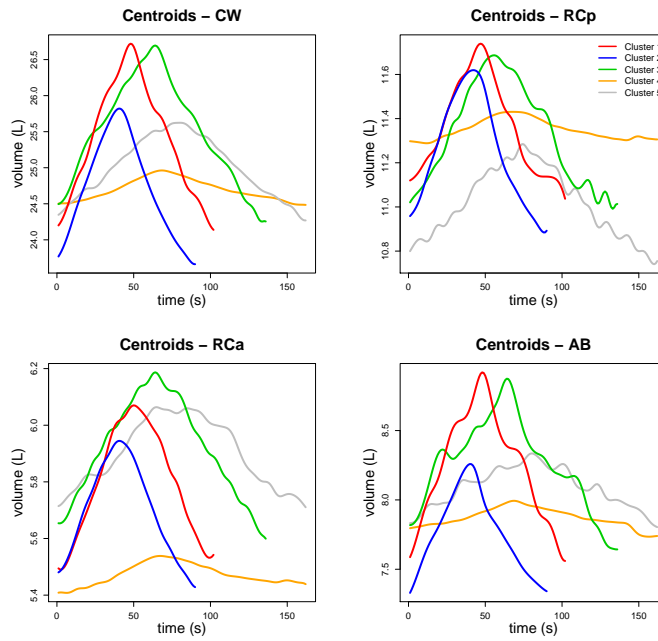


Figure 5.18: Cluster centroids in comparison.

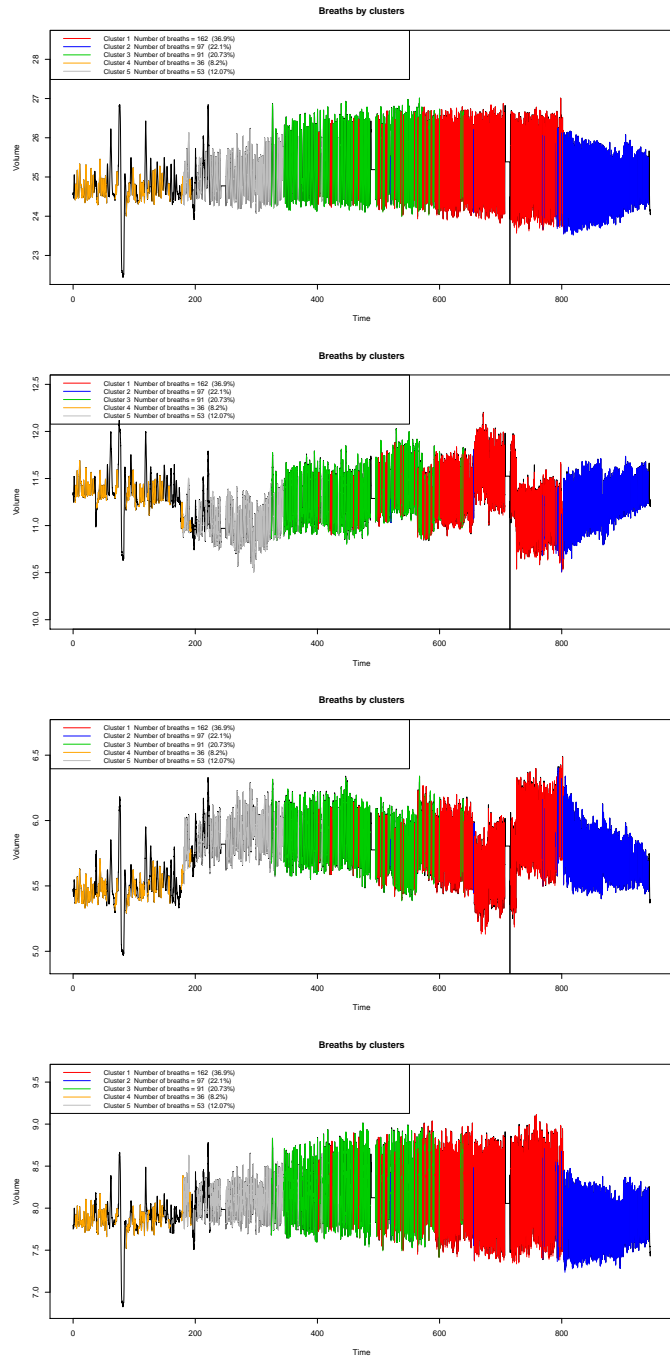


Figure 5.19: Clustering on data volumes.

### 5.5.4 Case report - Mechanical ventilation

This case study involves data acquisition from patient suffering from glycogenosis, a metabolic disorder caused by enzyme deficiencies affecting either glycogen synthesis, glycogen breakdown or glycolysis (glucose breakdown), typically in muscles and/or liver cells. Symptoms include enlargement of the liver, heart, or striated muscle, including the tongue, with progressive muscular weakness [23]. During the acquisition, mechanical ventilation of the patient was activated and stopped. It is possible to observe the transition from the patient's quiet breathing to the breath induced by mechanical ventilation and back, as shown in Figure 5.20. Ventilation replaces the action of the patient's respiratory muscles, having an effect both on the mean value of the chest wall volume and on the breathing dynamics. Therefore, in our analysis we expect to identify clusters characterized by differences in both volume and shape.

#### Outlier detection

The results of the full outlier detection is shown in Figure 5.21. The moment corresponding to the end of ventilation is marked as outlier, as well as some noise at the end of the track.

#### Patterns

Since with ventilation the absolute value of the volume is relevant, the chosen functional distance in this case has been the  $L^2$  one. The elbow in the mean similarities plot indicated 2 clusters: Cluster 1 covers the non-ventilated breathing, Cluster 2 covers the ventilated breathing (see Figure 5.22 and Figure 5.23).

#### Discussion

It is possible to notice in cluster 2 how the abdominal paradox is progressively reduced after ventilation is activated (Figure 5.24). When ventilation ends, the breathing activity immediately returns to its original pattern. Figure 5.25 shows the median breaths for all compartments in both clusters: non-ventilated breathing presents a paradoxical pattern in the abdomen. With ventilation, the phase shift of the abdomen is significantly reduced, thus the chest wall volume increases and we can see a slight decrease of the rib cage volume, which no longer has to compensate the full abdominal phase shift.

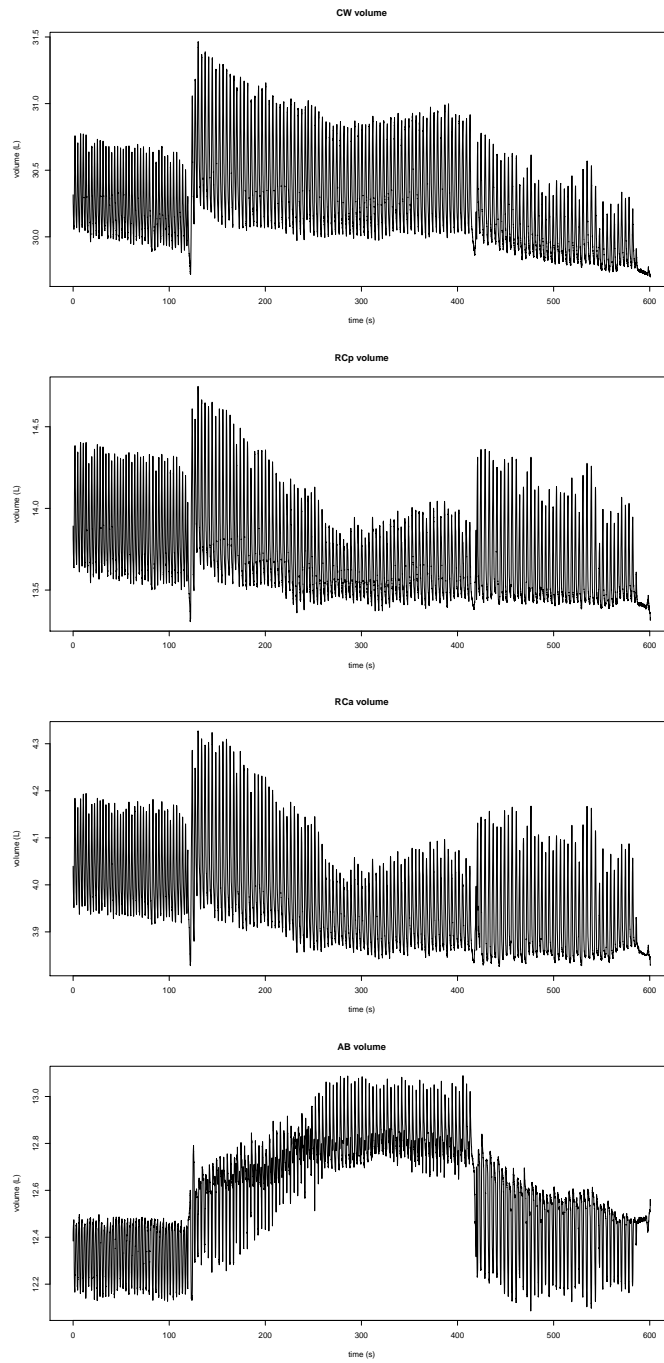


Figure 5.20: Breathing tracks - Chest wall and compartments

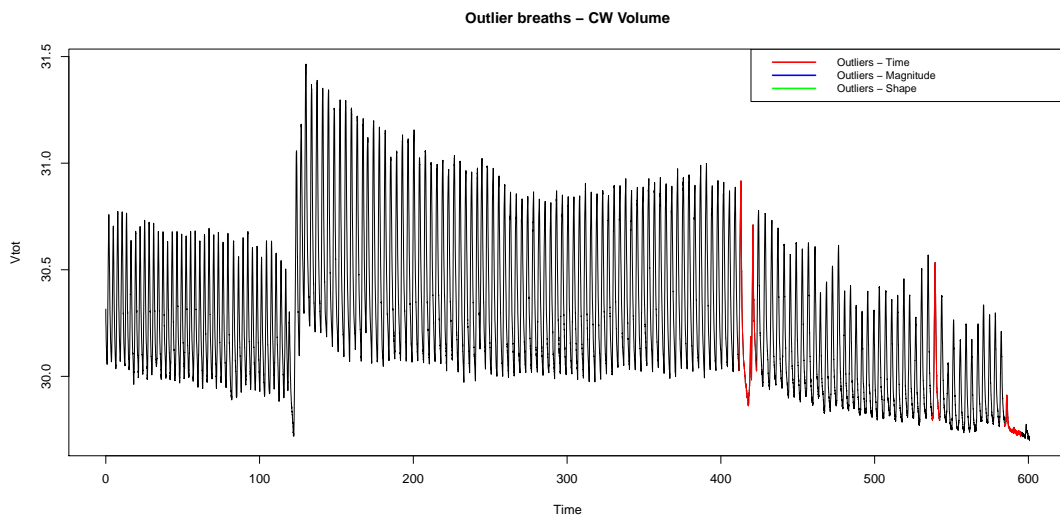


Figure 5.21: Outlier detection results.

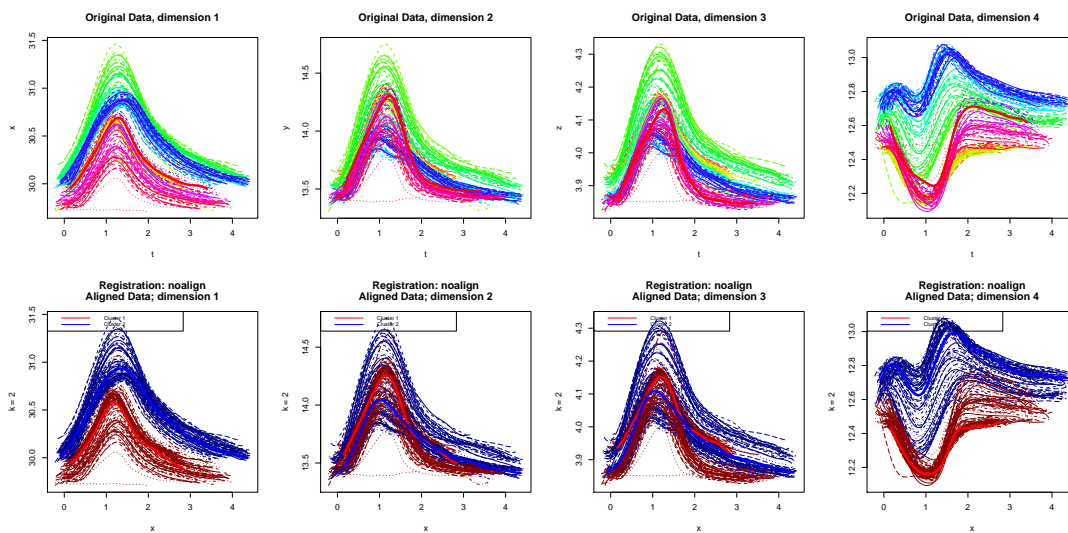


Figure 5.22: Multivariate kma results -  $L^2$  distance



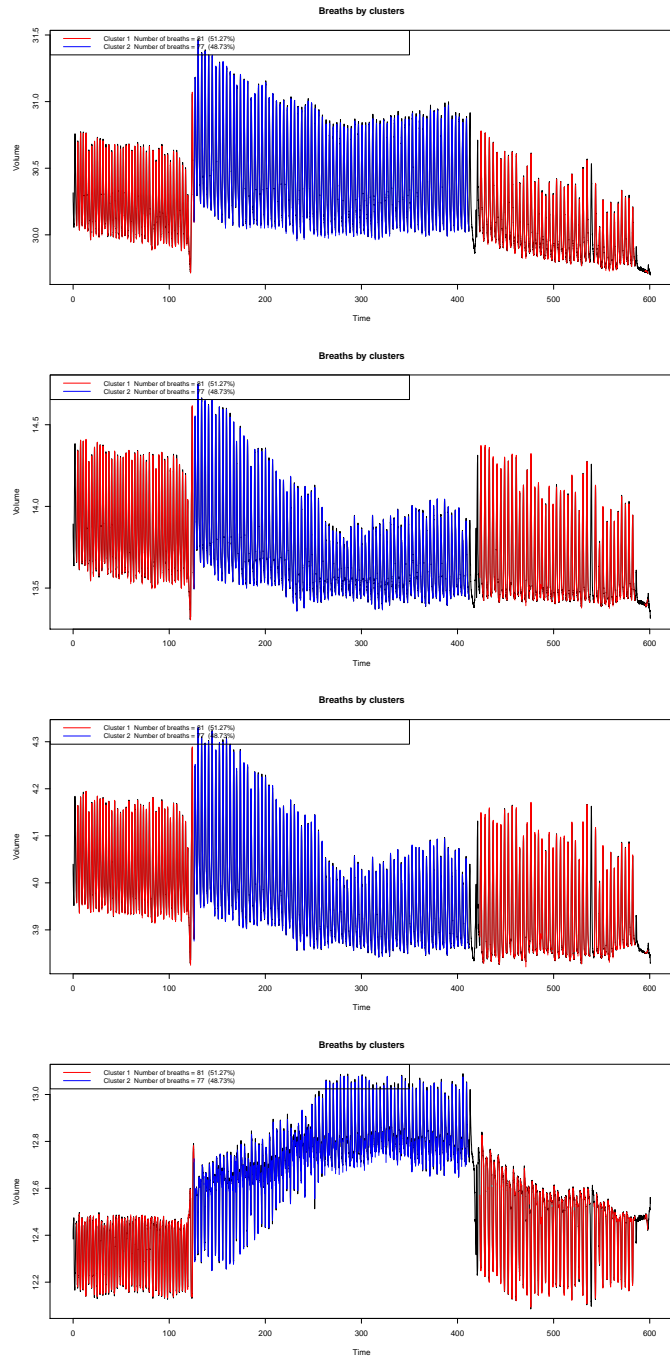


Figure 5.23: Clustering on data volumes.

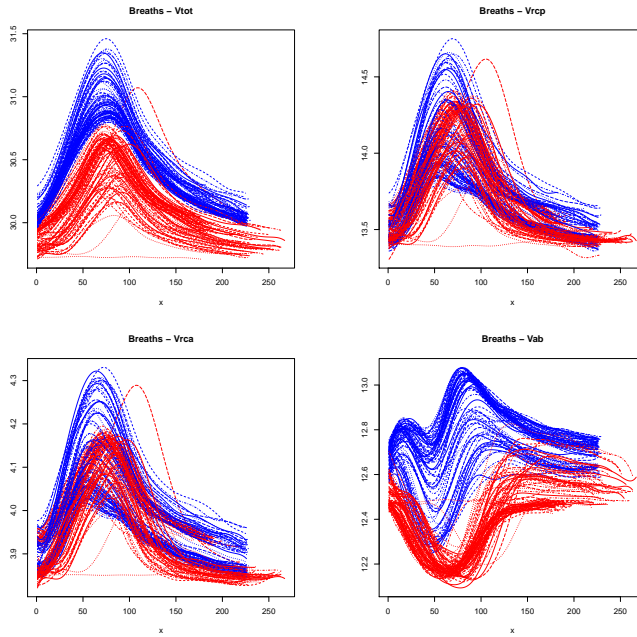


Figure 5.24: Clustered breaths

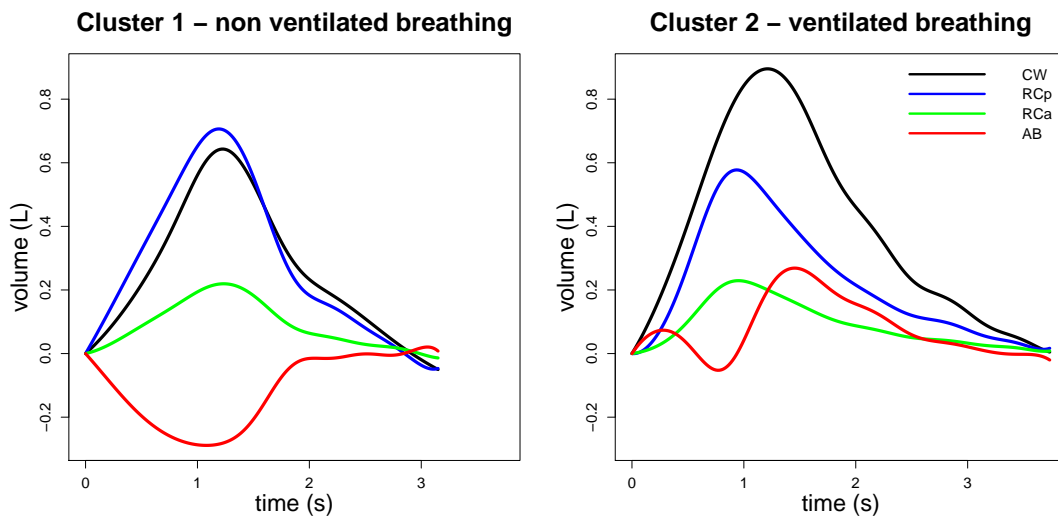


Figure 5.25: Median breaths of clusters

### 5.5.5 Case report - Duchenne

This case study involves data acquisition from a patient affected by Duchenne muscular dystrophy (DMD). DMD is an X-linked myopathy resulting in progressive wasting of locomotor and respiratory muscles, with consequent chronic ventilatory failure that is the main cause of death. In these patients it is extremely important, therefore, to measure lung function and respiratory muscle action in order to monitor the progression of the disease, to identify early signs of ventilatory insufficiency, to plan optimal interventions for improving the quality of life and to quantify the effects of novel gene-modifying strategies and pharmacological therapies [14]. The symptoms of muscle weakness usually begins around the age of four in boys and worsens quickly. Typically muscle loss occurs first in the thighs and pelvis followed by those of the arms. This can result in trouble standing up.

As we can see from Figure 5.26, although the total volume in this subject is regular, there is an alternation in all the chest wall compartments between abdominal breathing and rib cage breathing.

#### Outlier detection

The results of the outlier detection are shown in Figure 5.27. Almost no outliers are found, which is reasonable since the two breathing patterns are exactly alternated and equal in terms of number of breaths.

#### Patterns

The chosen functional distance in this case has been the  $L^2$  one, since we can clearly see a difference in absolute volume between the two breathing patterns. The elbow in the mean similarities plot indicated 2 clusters. Result is shown in Figures 5.28, 5.29 and 5.30 :

Cluster 2 covers the abdominal breathing, with a phase shift in the RCp volume. Cluster 1 covers the rib cage breathing. Figure 5.31 shows the median breaths of the chest wall and compartments for each cluster.

#### Discussion

This subject has two distinct, equally frequent breathing patterns. At the current stage, considering the overall median of the track is not advised because it would not be representative of the real way of breathing. As the dystrophy will worsen, the abdominal muscles will weaken more and more, so we expect cluster 1 to become dominant and cluster 2 to disappear progressively. In that case, it will be safer to associate the overall median to the patient breathing activity.

The  $L^2$  distance was selected in this case because the alternating segments differ significantly by absolute value. Pearson distance over breaths derivatives is also a valid choice because the shape of the two patterns is very different. Indeed, by grouping the first derivatives of the curves we basically obtain the same result, except for the extreme breath of each sub-group (see Figure 5.32 and 5.33). This may indicate that the transition between abdominal breathing and rib-cage breathing is not immediate.

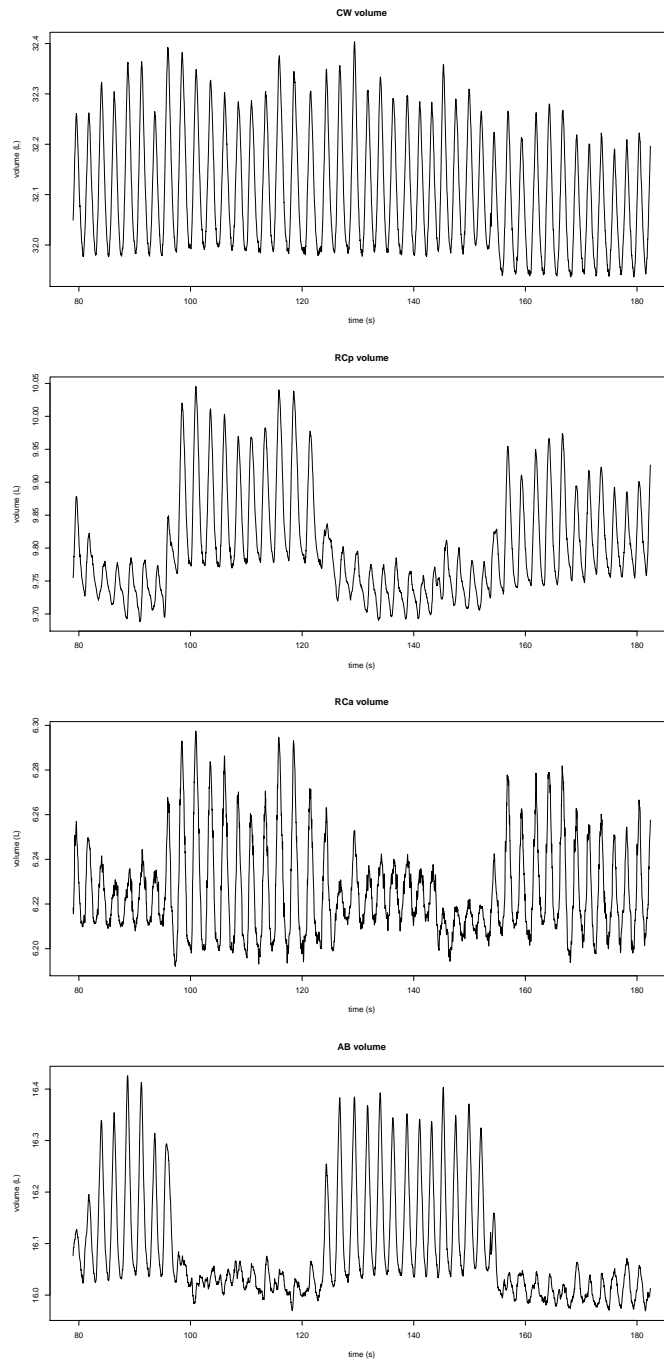


Figure 5.26: Breathing tracks - Chest wall and compartments

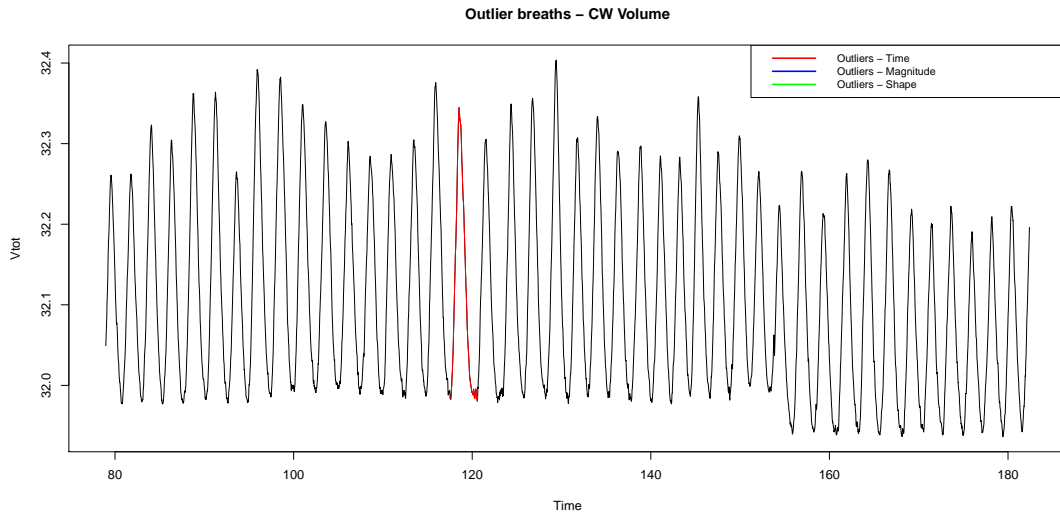


Figure 5.27: Outlier detection results in the chest wall.

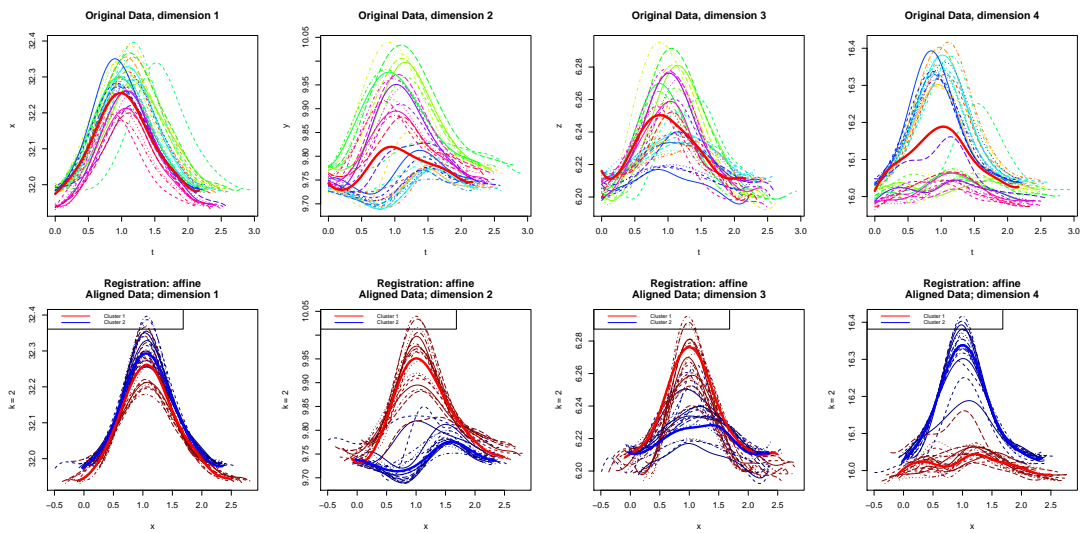


Figure 5.28: Multivariate kma results -  $L^2$  distance

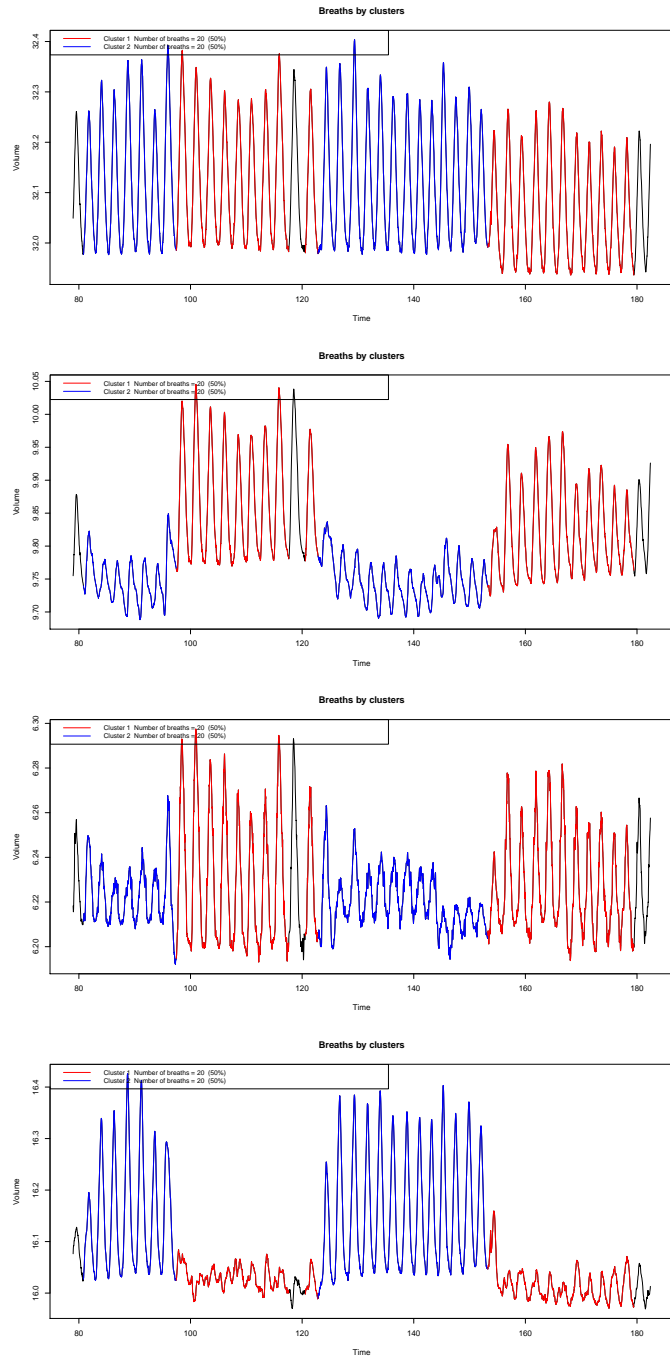


Figure 5.29: Clustering on data volumes.

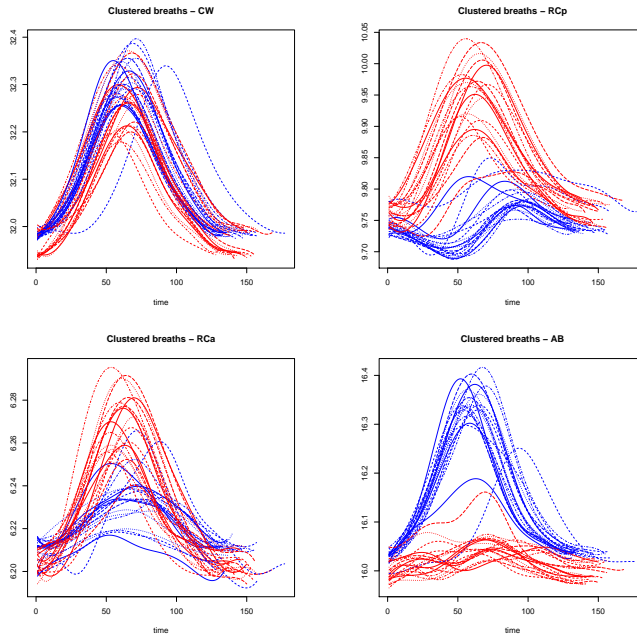


Figure 5.30: Clustered breaths

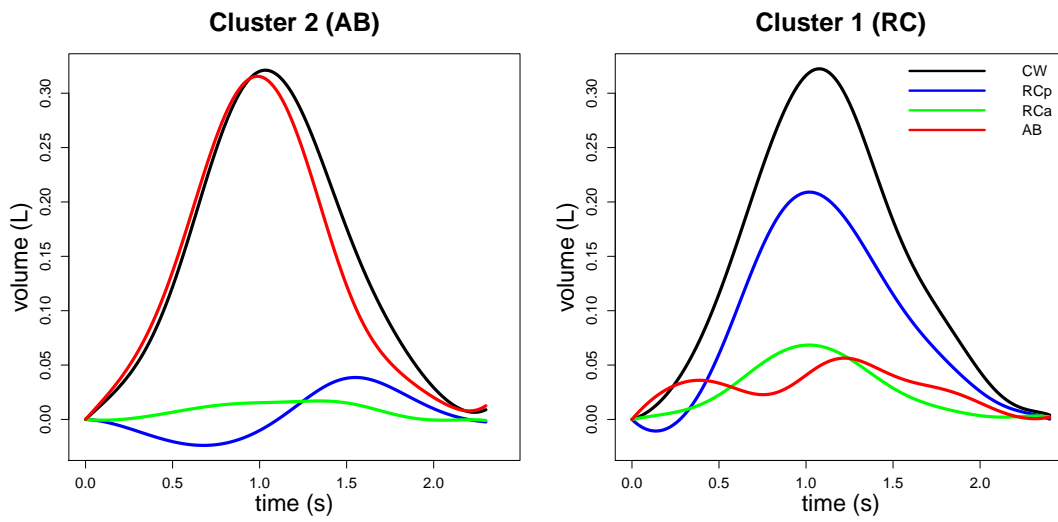


Figure 5.31: Median breaths of clusters

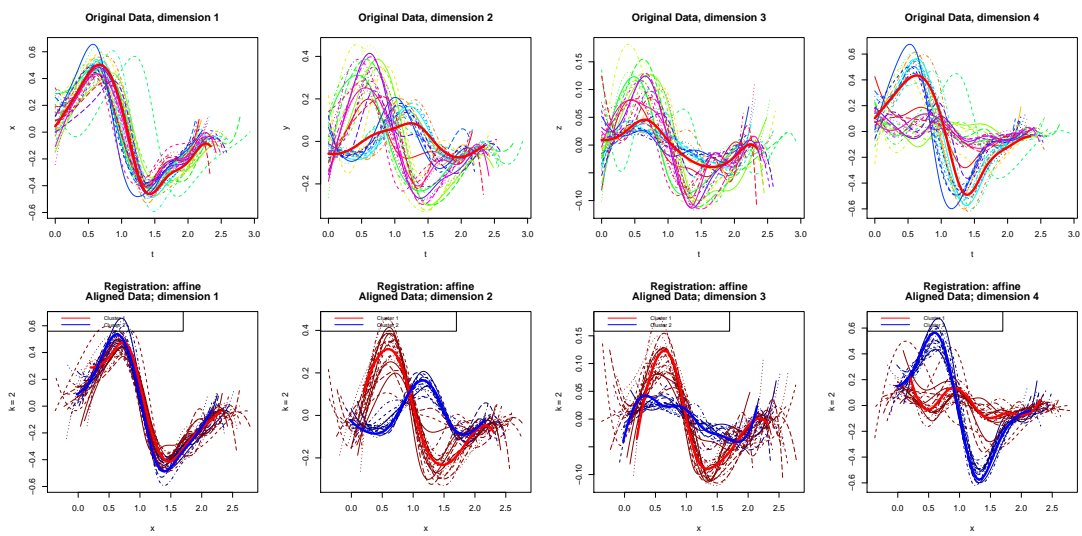


Figure 5.32: Multivariate kma results - Pearson distance



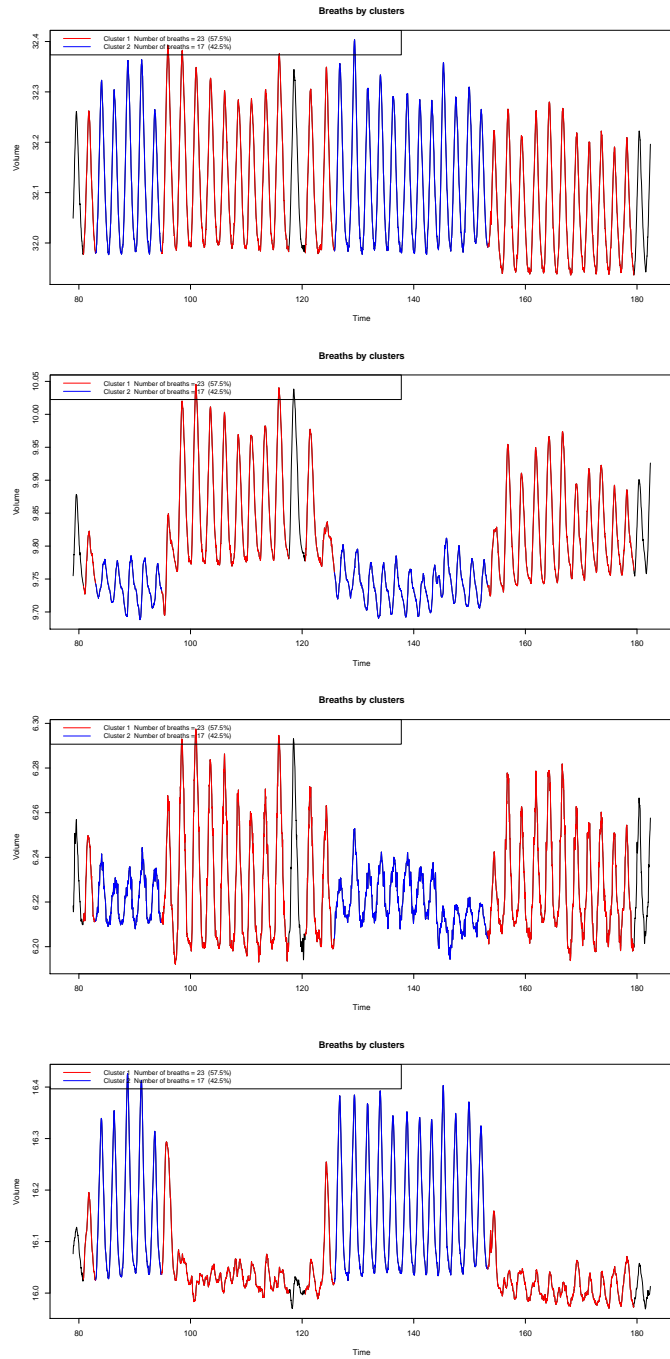


Figure 5.33: Clustering on data volumes.

## Chapter 6

# Population analyses

Breathing pattern comparison between different subjects is a complex task, which is usually done qualitatively or by means of the breathing pattern parameters. Here, a method to compare breathing functions from different subjects is described, taking advantage of the procedure defined in the previous chapter. In fact, the median breath as defined in section 5.1 can be picked (when it makes sense to do so) as a representative of the breathing mode of a person. Therefore, we can compare the representative breaths of different subjects applying a very similar procedure to the one that was defined for intra-patient analysis. Analyses over populations which differ on posture, age and clinical condition are presented.

### 6.1 Methodology

Let us describe the methodology that has been developed to compare breath curves of different subjects. First of all, it is important to notice that median breaths cannot be taken as they are, but some transformation is needed.

First, the absolute value of the volume has to be removed, since it mostly depends on the person's weight. What matters in this context is the volume variation, that is the breath amplitude. Therefore, all the representative breaths have to be made start from 0.

Then, it has to be considered that the value of breath amplitudes (the volume variation) depends on the lungs volume, which in turn depends on the chest wall height. This differs from subject to subject and does not influence respiration, thus it is also necessary to normalize amplitudes dividing by a suitable value. The candidate is of course the chest wall height, but this measure is problematic for several reasons. First of all, it has never been measured explicitly in any of previous studies, therefore data are not available; then, a rigorous definition of this measure has not been given. In fact, the chest wall height is influenced by posture and could even change during an acquisition if, for example, the patient moves.

Finally, in some situations one may suspect a systematic difference in the value of amplitudes between subjects groups, for example between healthy and unhealthy patients. This is for example the case of patients suffering from Duchenne Muscular Dystrophy (DMD), who undergo a progressive reduction of the tidal volume  $V_T$  due to muscle wasting [14]. In this setting it can be reasonable not to introduce any amplitude normalization, but to compare median breaths directly.

All these considered, we individuated three alternative ways to normalize the representative breath amplitude:

1. Divide the amplitude of all compartments by the patient's height. In fact height can be easily measured and is in general proportional to the chest wall height.
2. Divide by the amplitude of all compartments by the amplitude of the total volume, a.k.a. tidal volume ( $V_T$ ). In this way, compartment curves get an amplitude which represents the relative contribution of that compartment to the total (amplitudes like percentages of  $V_T$ ). If this alternative is chosen, the total volume curve becomes less informative and can be removed from the following analysis.
3. Leave amplitude as it is, i.e. no normalization. This alternative should be chosen when tidal volume differences between subjects is under interest (real amplitudes, in Litres).

Although the first alternative seems the most natural, we prefer a combination of both the second and third one, which produce more interpretable results in terms of measurement units ( $\%V_T$ , and L). In fact, no normalization can be used first to investigate differences in groups comprehensive of the tidal volume. Then, a second analysis with total volume normalization can be done, since it allows to compare the relative contribution of compartments to respiration in different subjects.

Once fixed normalization, outlier detection can be applied as in section 4.2. In this setting, outliers correspond to different patients, therefore their breathing pattern may be analysed later to understand the (possibly clinical) reasons of their outlyingness.

Finally, clustering techniques as in section 5.3 (absolute value) or section 5.4 (shape) can be applied to assess the presence of group structure within the sample, and if group structures are linked with some variables of interest, as the presence of some pathological condition.

## 6.2 Comparing different postures

This case study involves 54 healthy subjects measured in supine position and 28 healthy subjects measured in seated position. Each subject is represented by his multivariate median breath (chest wall and compartments). Each median is then normalized by its respective chest wall volume. Figure 6.1 shows the so obtained normalized breaths. We can observe that supine subjects are characterized by a lower contribution of the rib cage in the breathing activity and a higher contribution of the abdomen, while for seated subjects it is the opposite. This observation is supported by previous studies, e.g. in Aliverti et al., 2001 [3].

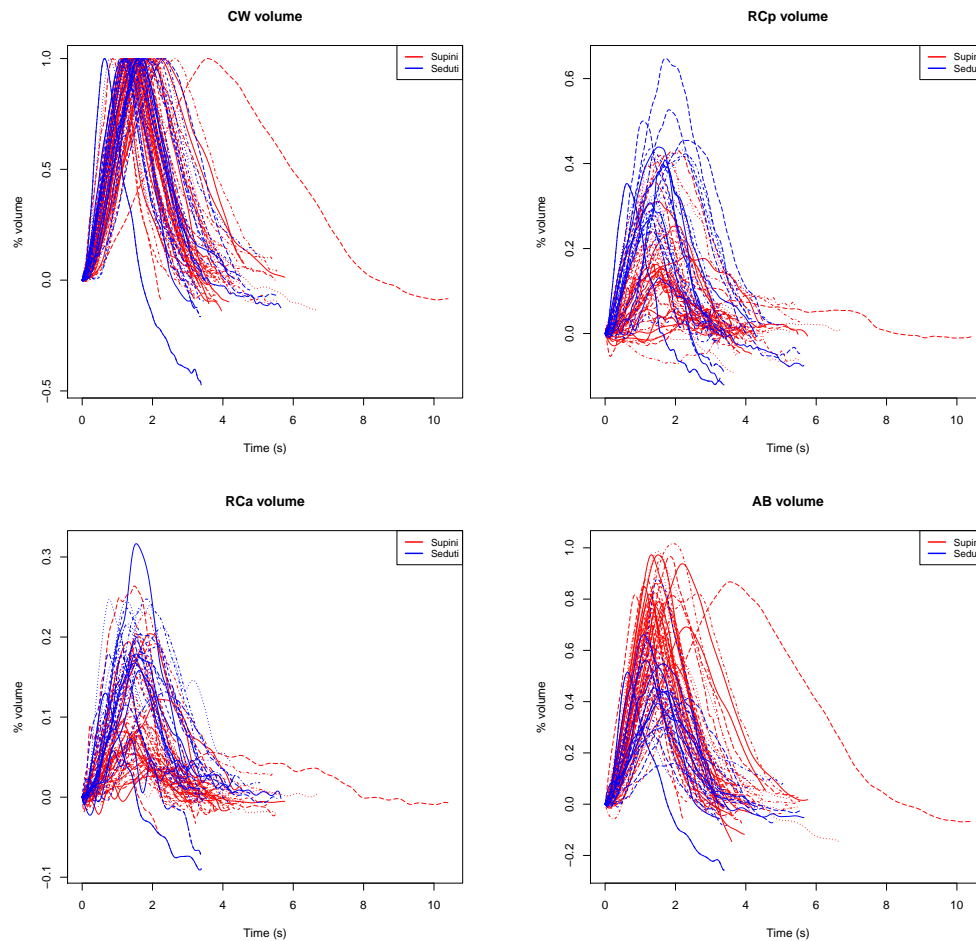


Figure 6.1: Normalized median breaths

We then proceed with outlier detection and clustering via K-medoids with Alignment.  $L^2$  distance is chosen since we expect different groups to be characterized by different percent contribution of the segments in the chest wall. The values of the similarity index suggests two clusters, the result

is shown in Figure 6.2. RCa volume was not considered since it has the lowest relative contribution to breathing activity, and it is usually noisier.

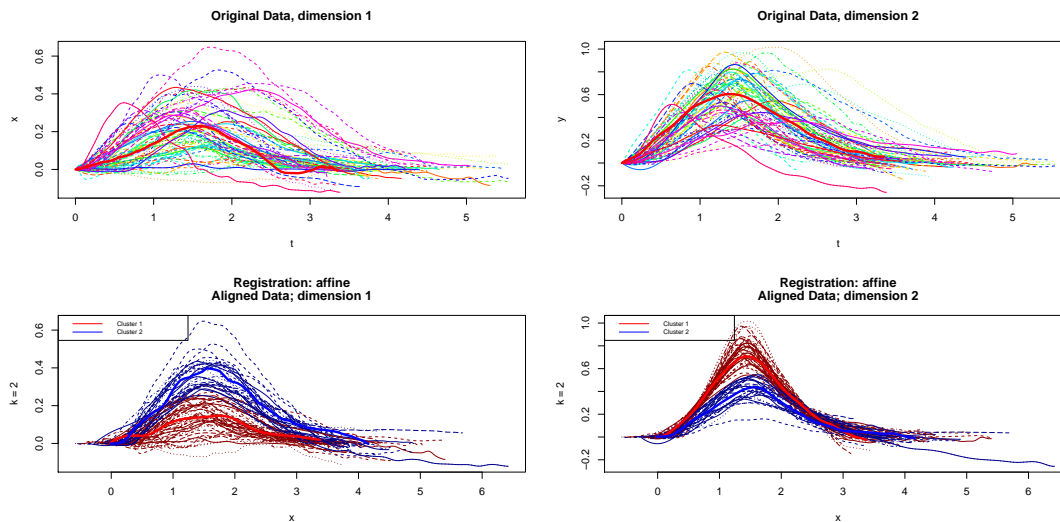


Figure 6.2: Inter-subject clustering

Cluster 1 groups breaths with a lower rib cage volume and a higher abdominal volume, while cluster 2 is characterized by a higher rib cage volume and a lower abdominal volume.

Checking the labels of each curve, we find that cluster 1 is mainly composed by supine and cluster 2 by seated. The misclassification table (Table 6.1) reports that the majority of subjects is correctly classified (outliers are not in the count).

		Classified	
		Supine	Seated
True	Supine	42	9
	Seated	5	20

Table 6.1: Confusion matrix for subject posture

Figure 6.3 shows how the breaths are classified by kma.

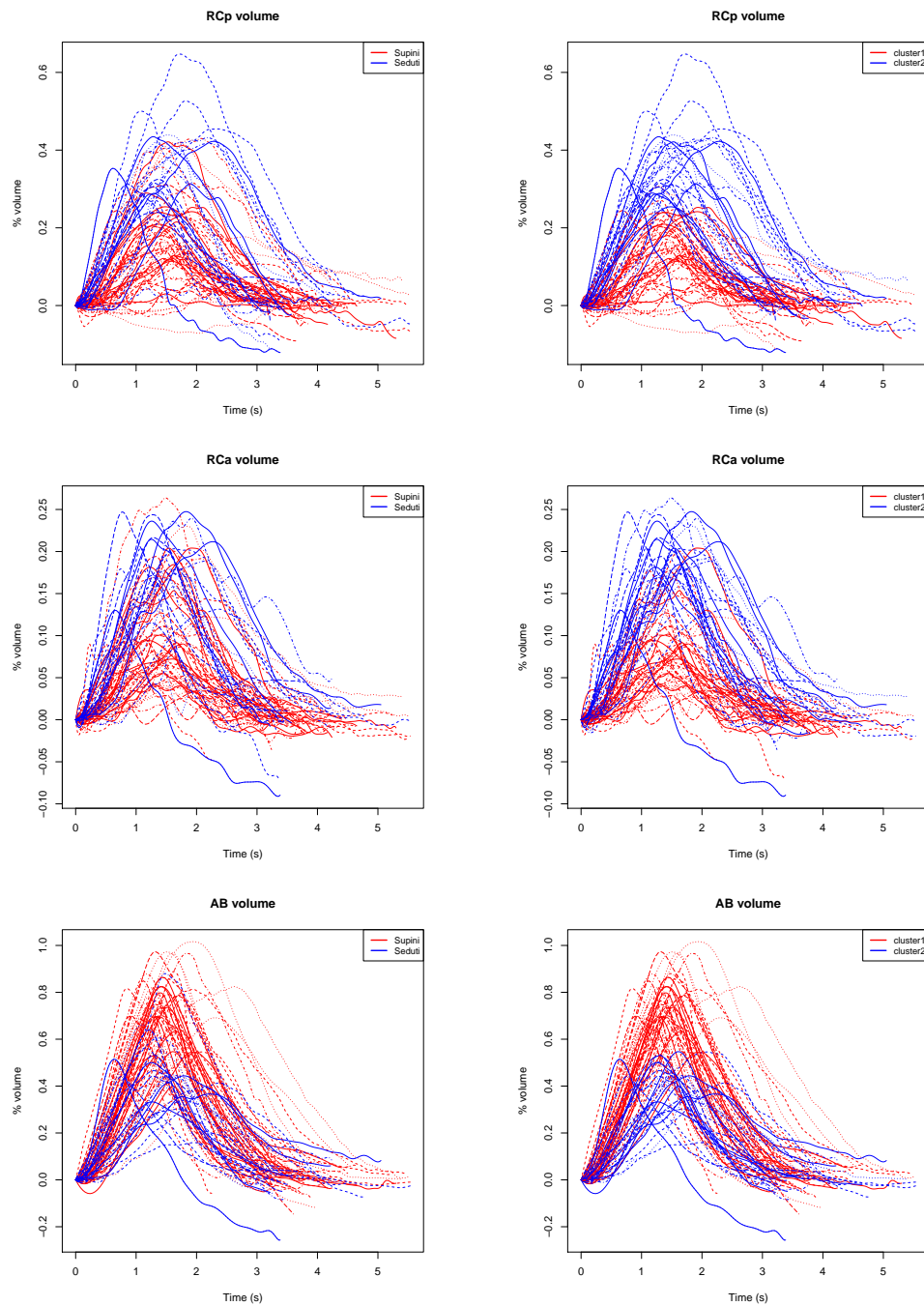


Figure 6.3: Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified.

## 6.3 Comparing different ages - healthy sample

This case study involves 65 healthy subjects of different ages measured in supine position. People in the sample belongs to the following age ranges:

- 21 subjects between 5 and 10 years old;
- 21 subjects between 18 and 25 years old;
- 23 over 60 subjects.

We would like to investigate whether there is any difference in the breathing pattern of children, young adults and elder people, both in terms of volume variations, and relative contribution of compartments. Each subject is represented by his multivariate median breath (chest wall and compartments).

### 6.3.1 Volume variations comparison

Median breaths are made to start from volume 0, and no other normalization is introduced. Figure 6.4 shows the ensemble of so obtained curves, where different colours correspond to different age classes. The children group is characterized by a smaller amplitude and a higher breathing frequency, while young adults seem to have the highest tidal volume. This difference is easily explained by the fact that children have smaller lungs, therefore have a smaller volume variation compared to adults.

We proceed with outlier detection and clustering. Outlier detection removes 2 children, 4 young and 9 over 60 among the noisiest. For classification the  $L^2$  distance is chosen since we expect different groups to be characterized by volume variations of the segments in the chest wall. The values of the similarity index suggests two clusters, the result is shown in Figure 6.5. Cluster 1 groups breaths with a higher volume variation in all compartments. Cluster 2 is characterized by smaller volume variations and duration. Not surprisingly, Cluster 2 encloses most of the children, while Cluster 1 contains adults. Table 6.2 is the confusion matrix for this case. Based on this result, we can say that there is no evidence of a significant difference between young adults and over 60 in terms of volume variations. Figure 6.6 shows how the breaths are classified by kma.

		Classified	
		[5-10]	[18-25] and Over 60
True	[5-10]	16	3
	[18-25] and Over 60	3	28

Table 6.2: Confusion matrix for subject age

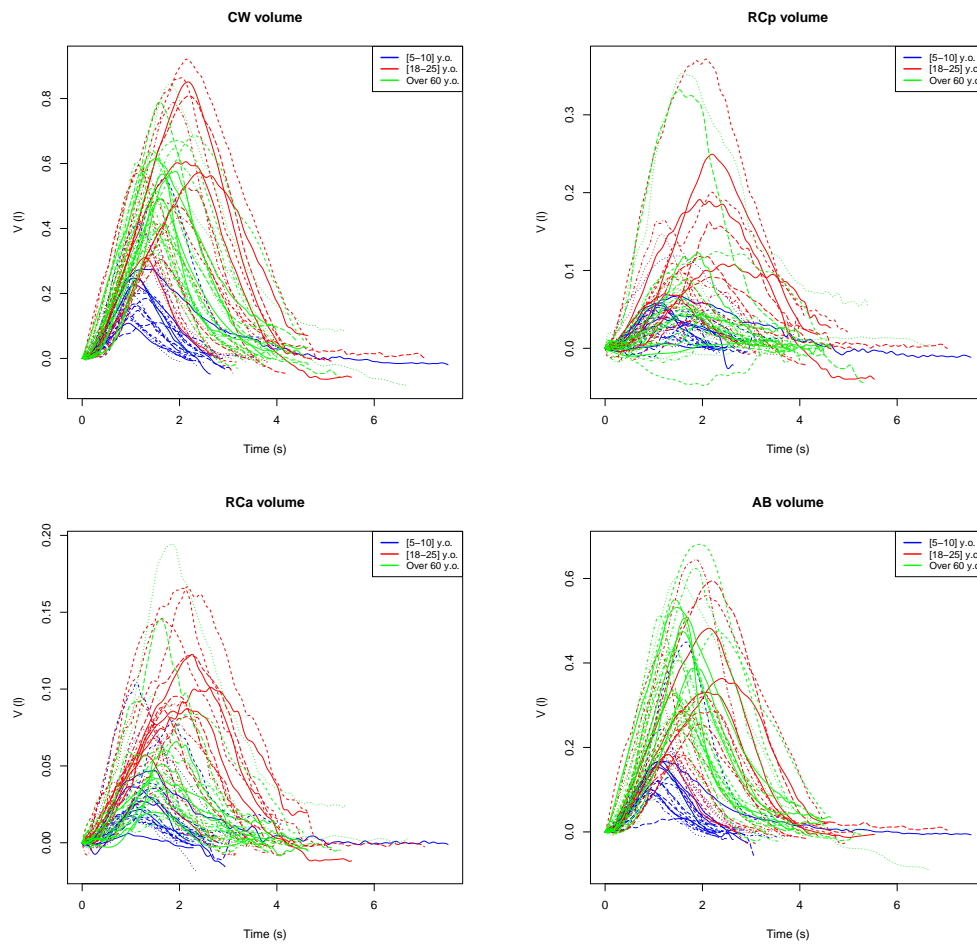


Figure 6.4: Median breaths - healthy sample



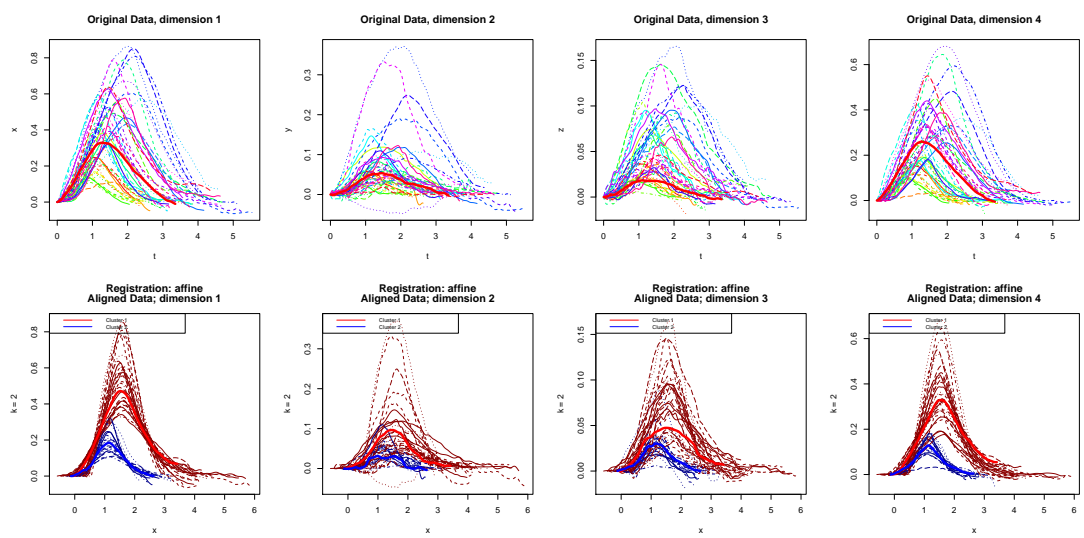


Figure 6.5: Classification results

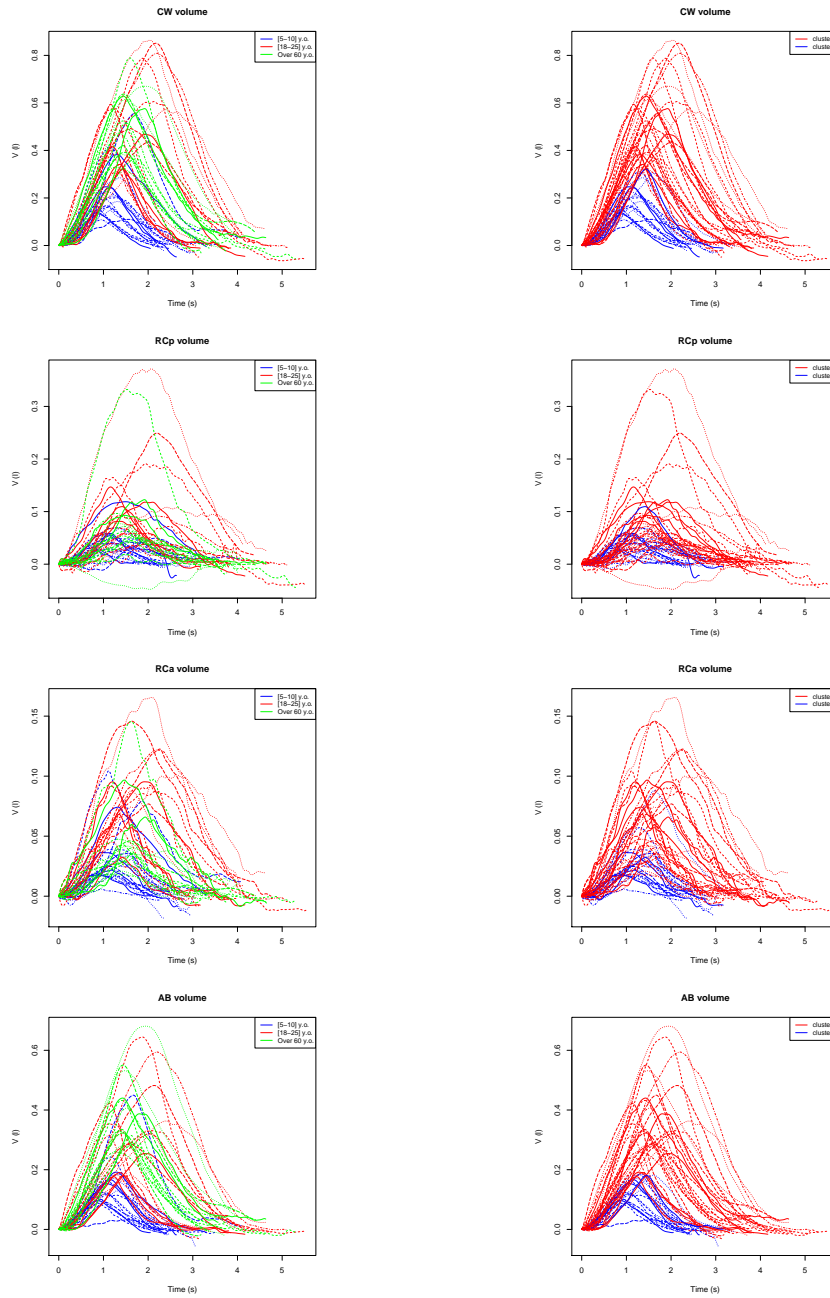


Figure 6.6: Classification results in terms of volume variation: the left column shows the true labels, the right column shows how breaths are classified.

### 6.3.2 Compartments relative contribution

Here each median is normalized by its respective chest wall volume. Figure 6.7 shows the so obtained normalized breaths. We can observe that younger subjects seem to be characterized on average by a higher rib cage and a lower abdomen contribution, while for older subjects it is the opposite.

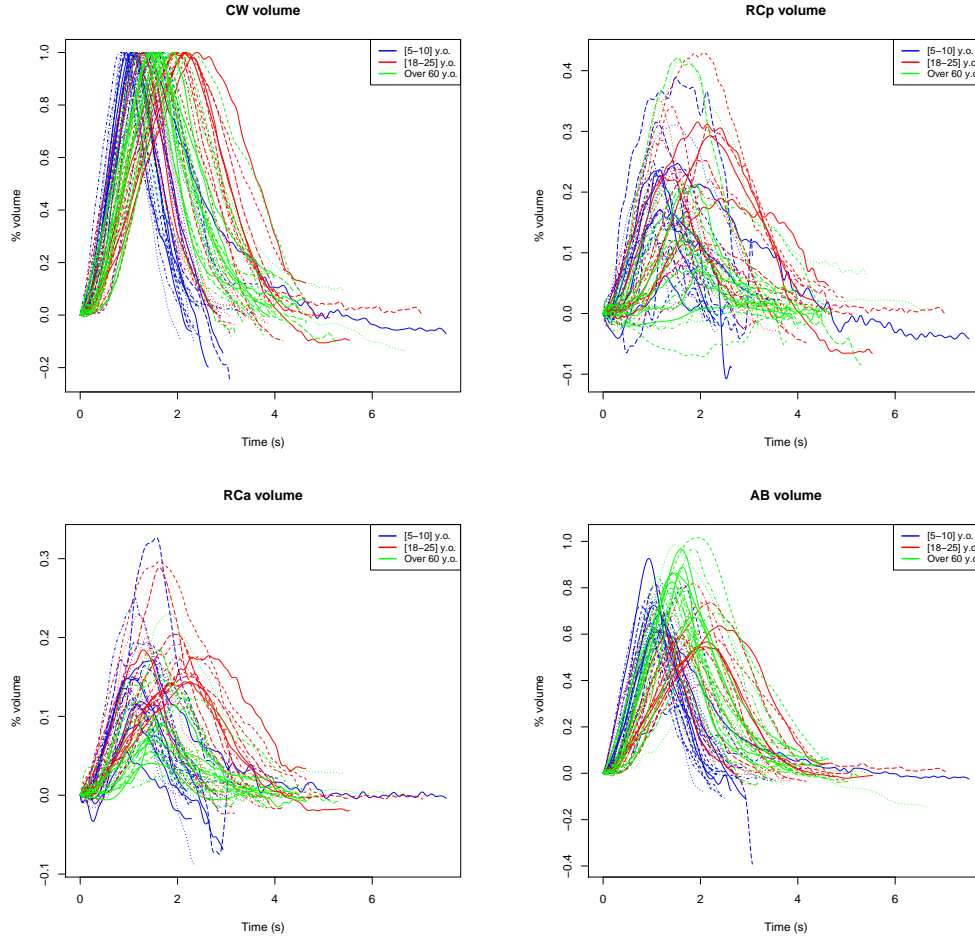


Figure 6.7: Normalized median breaths - healthy sample

Outlier detection spotted 4 children, 3 young and 5 elders. The  $L^2$  distance is chosen for classification since we expect different groups to be characterized by different relative contribution of the segments in the chest wall. The values of the similarity index suggests three clusters, the result is shown in Figure 6.8. As before, RCa volume was not considered in classification, since it has the lowest contribution to breathing activity.

Cluster 1 groups breaths with a higher rib cage and lower abdominal contribution. Cluster 2 is characterized by a lower rib cage contribution, while abdomen is higher and breaths duration is shorter. Cluster 3 has the lowest rib cage contribution, sometimes paradoxical, and high abdominal

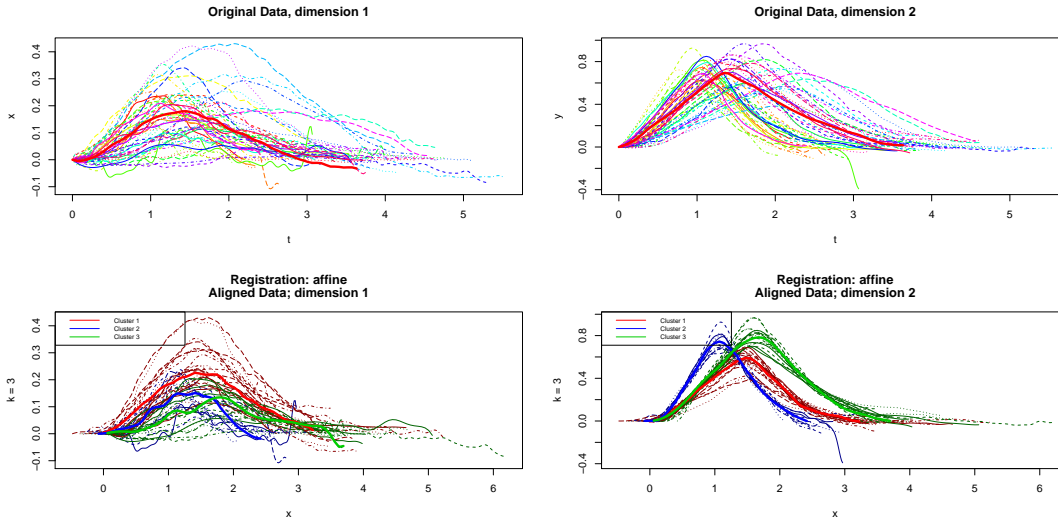


Figure 6.8: Classification result - compartment contribution

volume, while breathing frequency is similar to the one in Cluster 1. We check whether these clusters are linked with age. Cluster labels give the following division:

- Cluster 1 is composed by: 7 subjects in [5-10] age range, 12 in [18-25] range, 4 over 60;
- Cluster 2 is composed by: 9 subjects in [5-10] age range, 2 in [18-25] range, 2 over 60;
- Cluster 3 is composed by: 1 subject in [5-10] age range, 4 in [18-25] range, 12 over 60.

We can then assign Cluster 1 to young adults, Cluster 2 to children and Cluster 3 to over 60. Table 6.3 shows the age confusion matrix. We observe that some children are classified as young adults, while some young adults are put together with elders.

		Classified		
		[5-10]	[18-25]	Over 60
True	[5-10]	9	7	1
	[18-25]	2	12	4
	over 60	2	4	12

Table 6.3: Confusion matrix for subject age - compartment contribution

On the other hand, we can merge the two groups of [5-10] and [18-25] age range into a single population of under 25 years old. In this case, we have that Cluster 1 and 2 together provide a good covering of this newly defined group, as shown by the Table 6.4.

These results suggest that, although there might be some difference between children and young adults, it seems to be linked most with breathing frequency rather than compartmental contribution, and it is not neat. On the contrary, over 60 are well-distinguished from the younger subjects. In

		Classified	
		[5-10] and [18-25]	Over 60
True	[5-10] and [18-25]	30	5
	Over 60	6	12

Table 6.4: New confusion matrix for subject age - compartment contribution

fact, RCp contribution in that group is lower than the abdominal one, while the rib cage may also show a paradoxical behaviour. This may be explained by a greater rigidity of the rib cage in seniors, due to a lower muscular tone. Figure 6.9 shows how the breaths are classified by kma, considering cluster 1 and 2 as representative of the under 25 population.

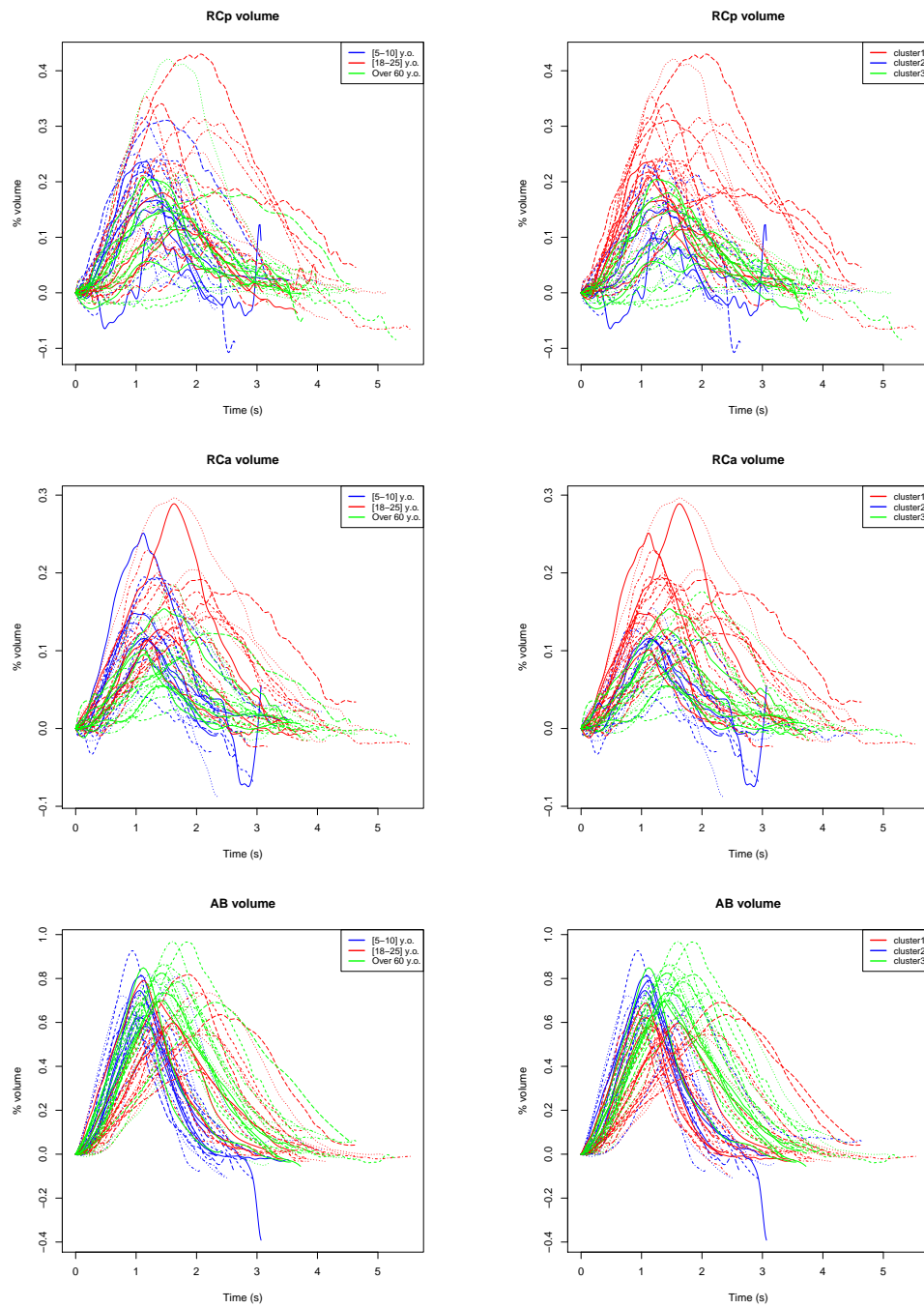


Figure 6.9: Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified.

## 6.4 Comparing different ages - DMD sample

The following case studies compare a group of 49 children between 5 and 10 years old affected by Duchenne Muscular Dystrophy (DMD) with a group of 40 elder patients aged from 18 to 25 years old. We recall that DMD is a severe genetic disease which results in progressive locomotor and respiratory muscle wasting, whose symptoms arise during childhood [14]. All acquisitions are in supine position. Each subject is represented by his multivariate median breath (chest wall and compartments). We would like to investigate whether there are differences in breathing pattern in the two age classes in terms of volume variation and relative contribution of compartments.

### 6.4.1 Volume variations comparison

Median breaths are made to start from volume 0, and no other normalization is introduced. Figure 6.10 shows the ensemble of so obtained curves, where different colours correspond to different age classes. Children seem to have a lower pulmonary rib cage volume with respect to adults. Notice that, differently from what we have seen for healthy subjects, tidal volume for elder DMD seems not higher than the children's one. In fact, reduction of the tidal volume in adult DMD patients is a symptom of respiratory muscles weakening [14].

We proceed with outlier detection and clustering. Some DMD children are clearly outlying, both in terms of duration and shape, and are removed as expected. In total, during outlier detection 8 children and 1 adult are spotted.

For classification the  $L^2$  distance is chosen as before. The values of the similarity index suggests two clusters. Result is shown in Figure 6.11 while the confusion matrix is in Table 6.5. Based on this, we do not see an evident age-related subdivision between groups (which is opposite from what we observed in healthy subjects).

		Classified	
		[5-10]	[18-25]
True	[5-10]	32	9
	[18-25]	15	24

Table 6.5: Confusion matrix for subject age - DMD sample

We then remove the RCa compartment, which gives the lowest contribution to tidal volume, in order to verify whether age-related differences arise in the behaviour of the pulmonary rib cage compared with abdomen. Result of classification is shown in Figure 6.12.

Cluster 1 groups breaths with a higher volume variation in abdomen rather than rib cage. Cluster 2 is characterized by smaller abdominal variations and a higher tidal volume. Now Cluster 2 encloses most of adult patients, while Cluster 1 contains children. Table 6.6 is the confusion matrix for this case. Classified breaths are shown in Figure 6.13. Based on this result, we can say that adults are characterized by a higher tidal volume and rib cage volume, while children have got a higher abdominal volume variation. We can compare these insights with the ones obtained with a healthy sample, where adults had a significantly higher volume variation in the chest wall and all of its compartments with respect to children.

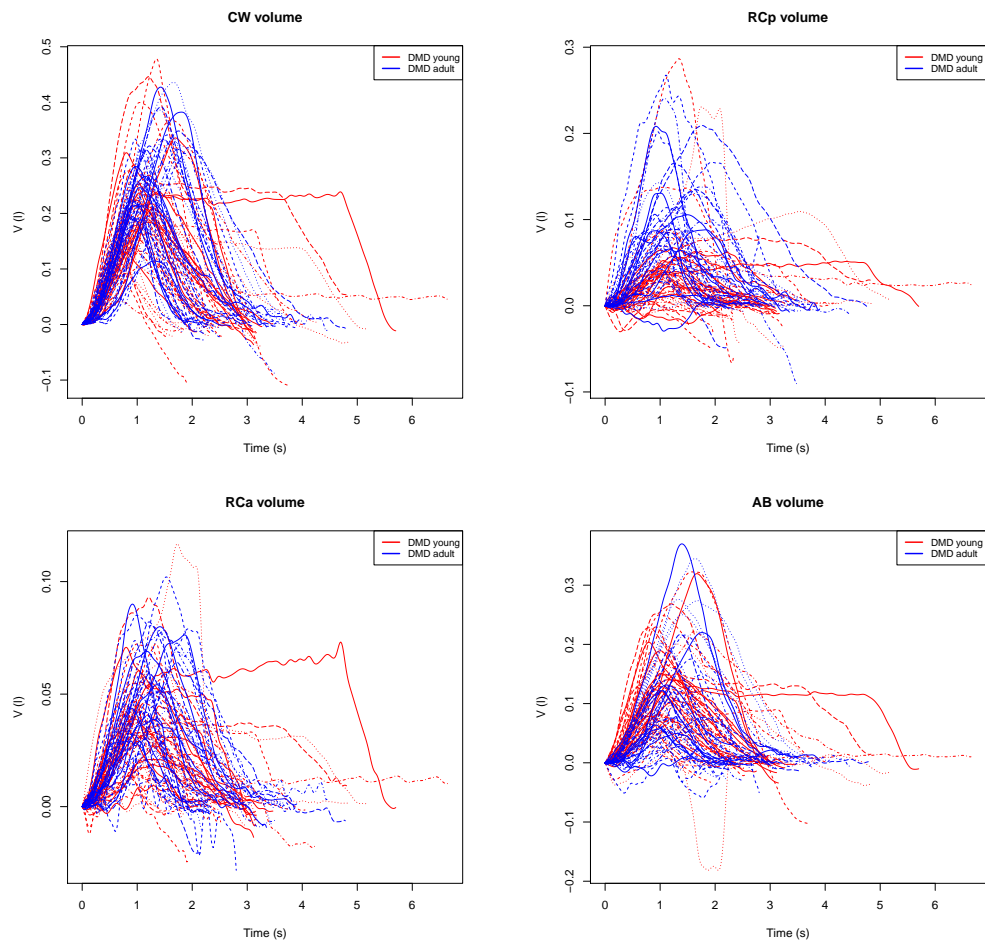


Figure 6.10: Median breaths - DMD sample

		Classified	
		[5-10]	[18-25]
True	[5-10]	31	10
	[18-25]	5	34

Table 6.6: New confusion matrix for subject age - DMD sample



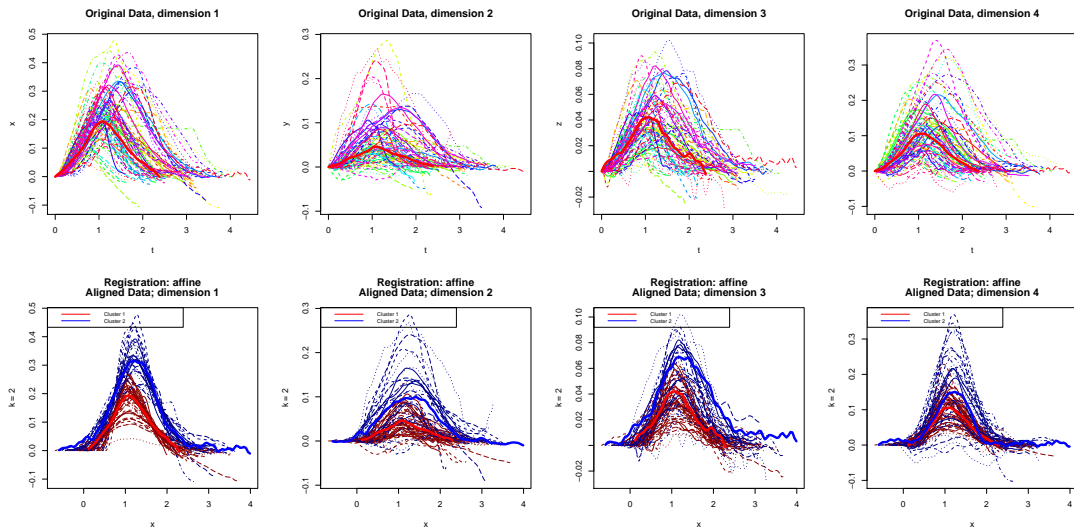


Figure 6.11: Classification results - DMD sample

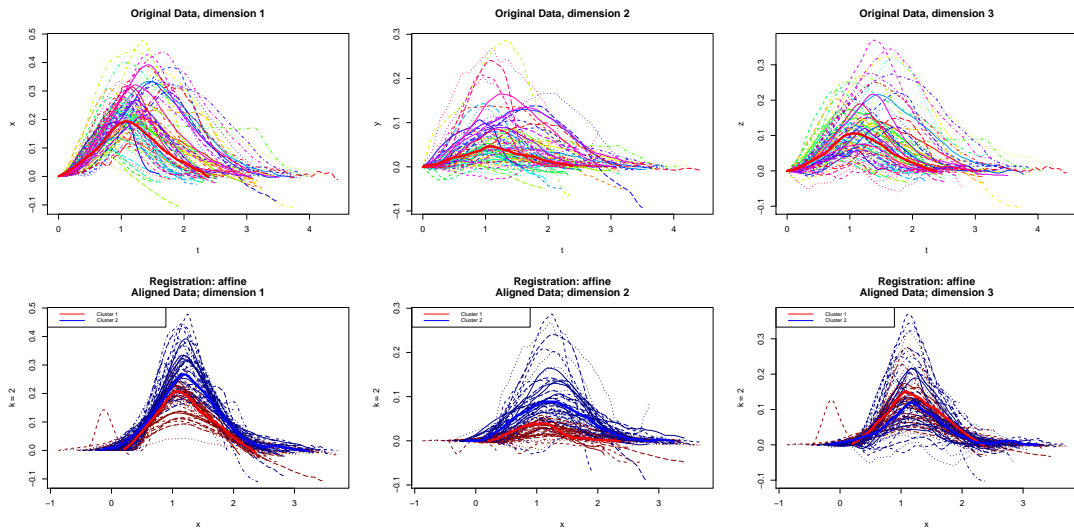


Figure 6.12: Classification results without RCa - DMD sample

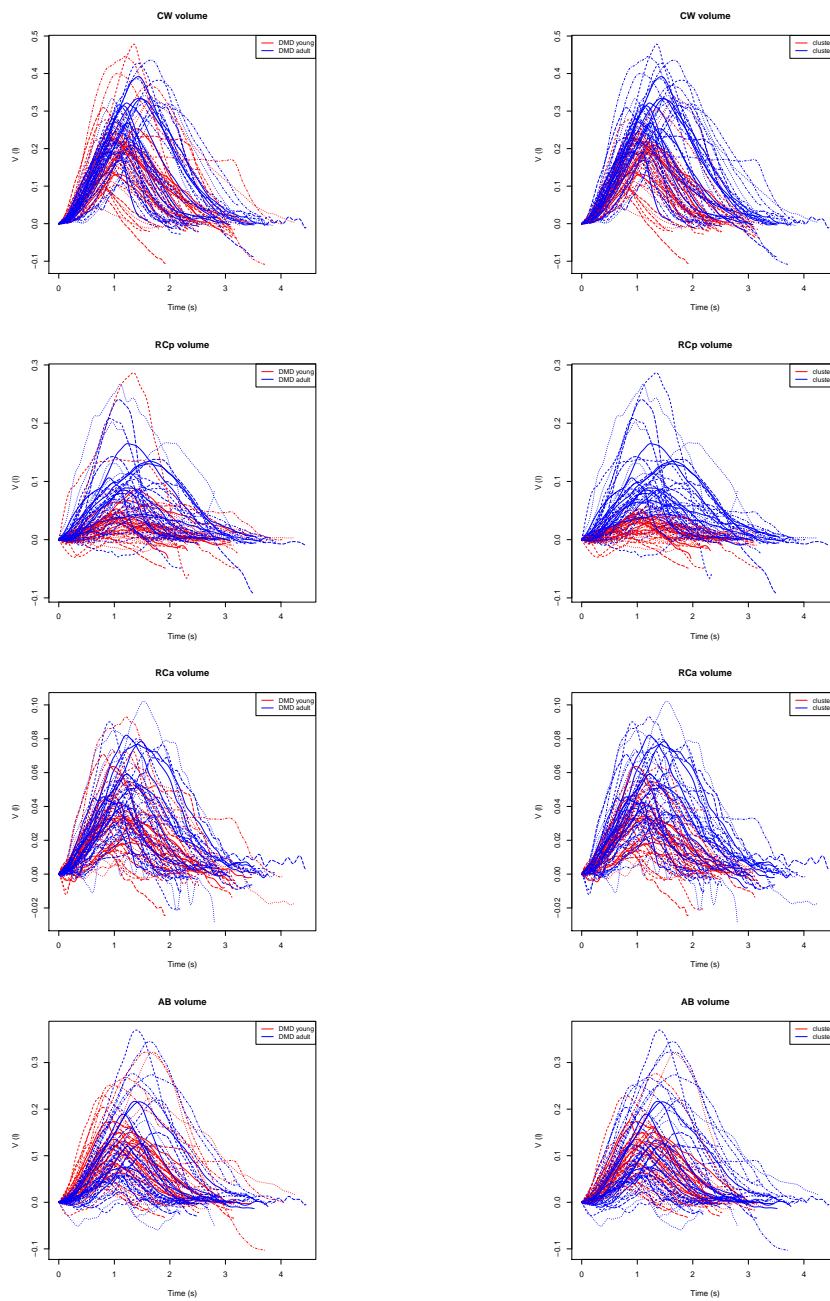


Figure 6.13: Classified median breaths - DMD sample

## 6.4.2 Compartments relative contribution

As before, here medians are normalized dividing by their corresponding tidal volume, while only RCp and AB are considered for classification. Figure 6.14 shows the normalized curves. We can observe that adults seem to be characterized by a higher rib cage and a lower abdomen contribution, while for children the opposite seems to hold.

Outlier detection removes 10 subjects in the 5-10 age range, which we recall to be the noisiest group, and one in 18-25. Figures 6.15 and 6.16 show the results of classification. Confusion matrix is provided by Table 6.7. Cluster 1 groups breaths which have a lower rib cage contribution with respect to the abdominal one, and covers most of the children, while the opposite holds for Cluster 2 which holds most of the adults. This is coherent with the fact that the diaphragm, which is responsible for the abdominal volume variation, is weakened first by the disease, and needs compensation by the rib cage. Therefore, while in children supine respiration is prevalently abdominal (as in healthy subjects), elder patients have a major rib cage contribution. Nevertheless, as we have seen in the volume variations comparison, this compensation is not sufficient to guarantee a level of ventilation comparable with the one of healthy adults.

		Classified	
		[5-10]	[18-25]
True	[5-10]	29	10
	[18-25]	10	29

Table 6.7: Confusion matrix for subject age - Compartment contribution in DMD sample

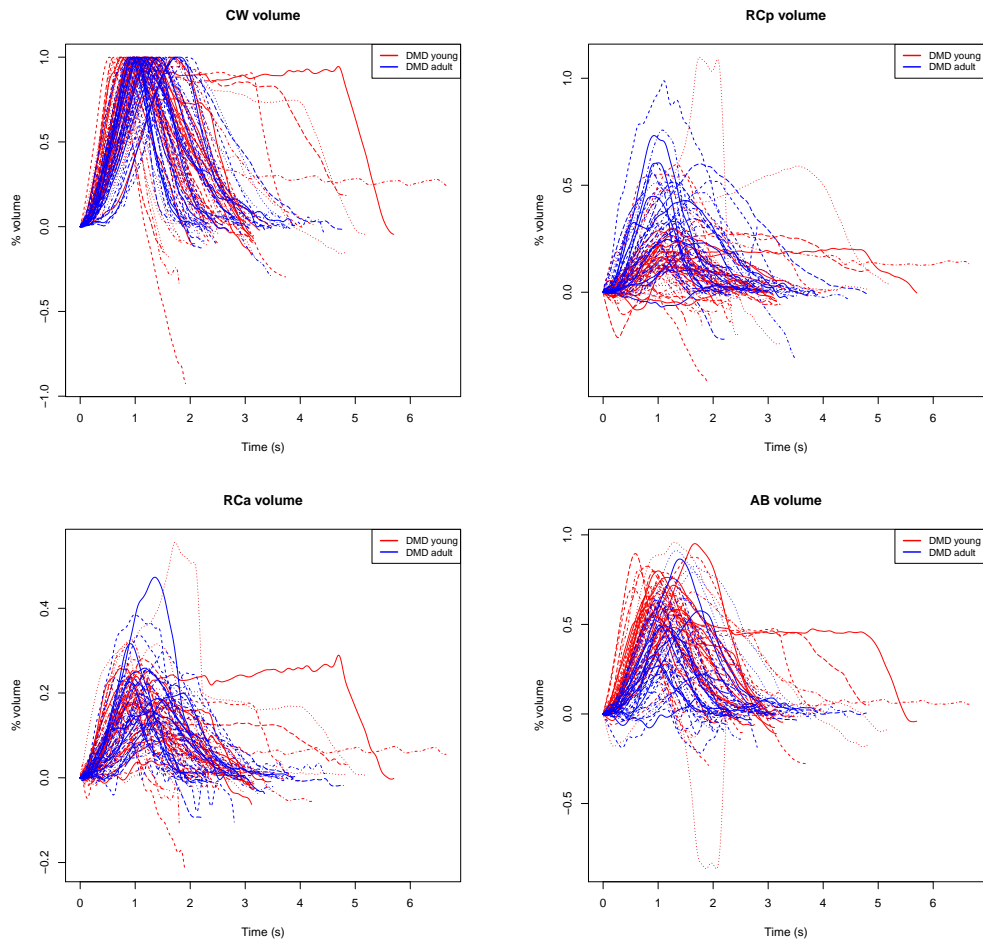


Figure 6.14: Normalized median breaths - DMD sample

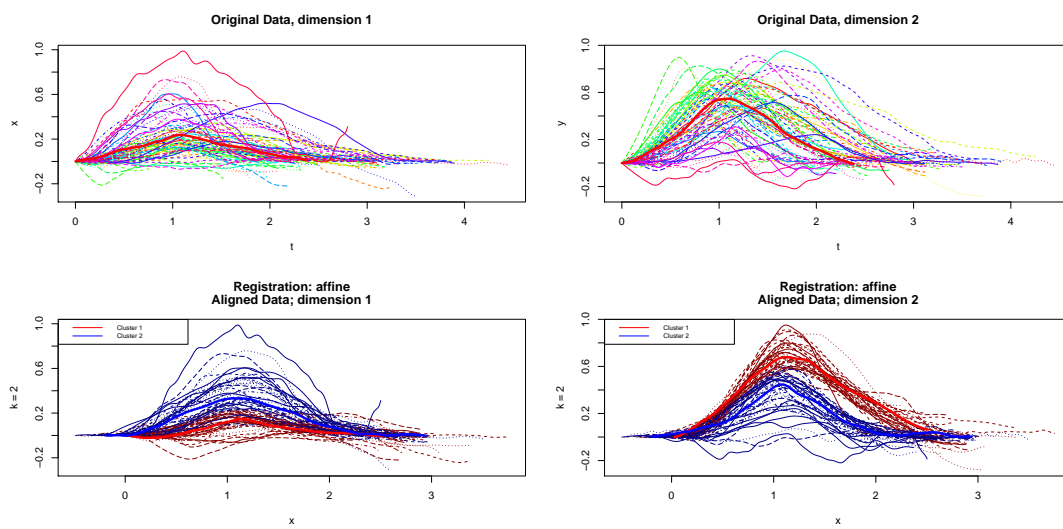


Figure 6.15: Classification results - Compartment contribution in DMD sample

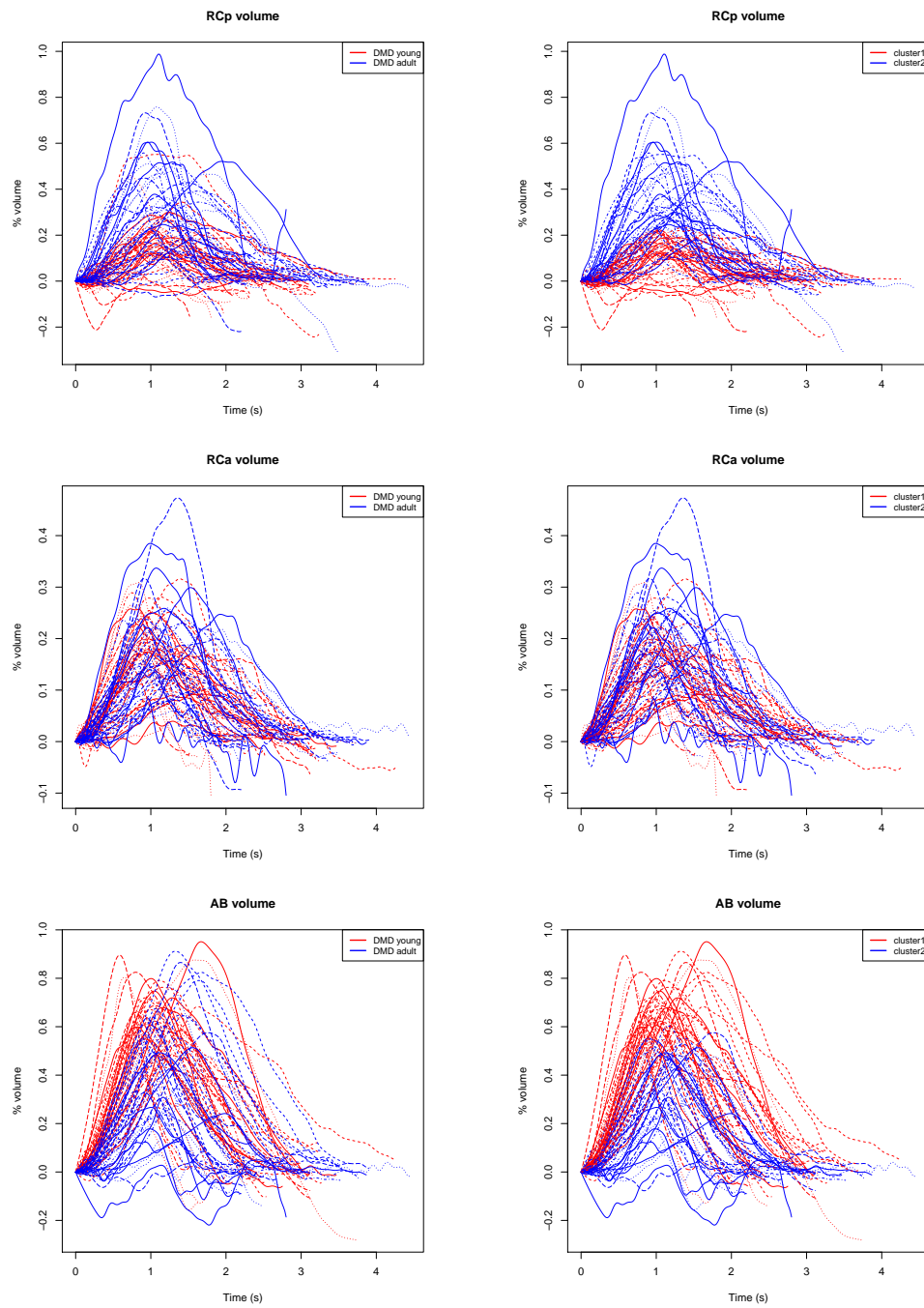


Figure 6.16: Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified.

## 6.5 Comparing different clinical conditions

The following case studies compare a group of patients affected by Duchenne Muscular Dystrophy (DMD) with a control group of healthy subjects (CTR). All acquisitions are in supine position. Each subject is represented by his multivariate median breath (chest wall and compartments). We would like to investigate whether there is any difference in the breathing pattern of healthy and not healthy subjects in same age range in terms of volume variation and relative contribution of compartments.

### 6.5.1 Healthy children and DMD children

In this analysis we consider the following samples:

- CTR young: 21 healthy subjects between 5 and 10 years;
- DMD young: 49 DMD patients between 5 and 10 years.

#### Volume variations comparison

Median breaths are made to start from volume 0, and no other normalization is introduced. Figure 6.17 shows the so obtained curves, where different colours correspond to different age classes. There seem to be no clear distinction between healthy and DMD subjects.

Some DMD children are clearly outlying, both in terms of duration and shape, and are removed as expected. In total, during outlier detection 5 CTR and 21 DMD are removed. The reason why so many curves are removed is that data acquisition for children is typically short and very noisy (children move a lot and movement is detected by the software) so the median for each subject is computed on a much lower number of breaths with respect to an adult. This implies that medians are more irregular and thus the outlier detection is more sensitive.

Figures 6.18 and 6.19 show the results of classification. Cluster 1 groups breaths which have an overall larger volume, cluster 2 groups those with a lower volume. However, none of the two clusters is representative of CTR or DMD sample so there is no point in constructing a misclassification table. For an age range of [5-10] years old, it seem that the difference in volume variation is not linked to the clinic condition.

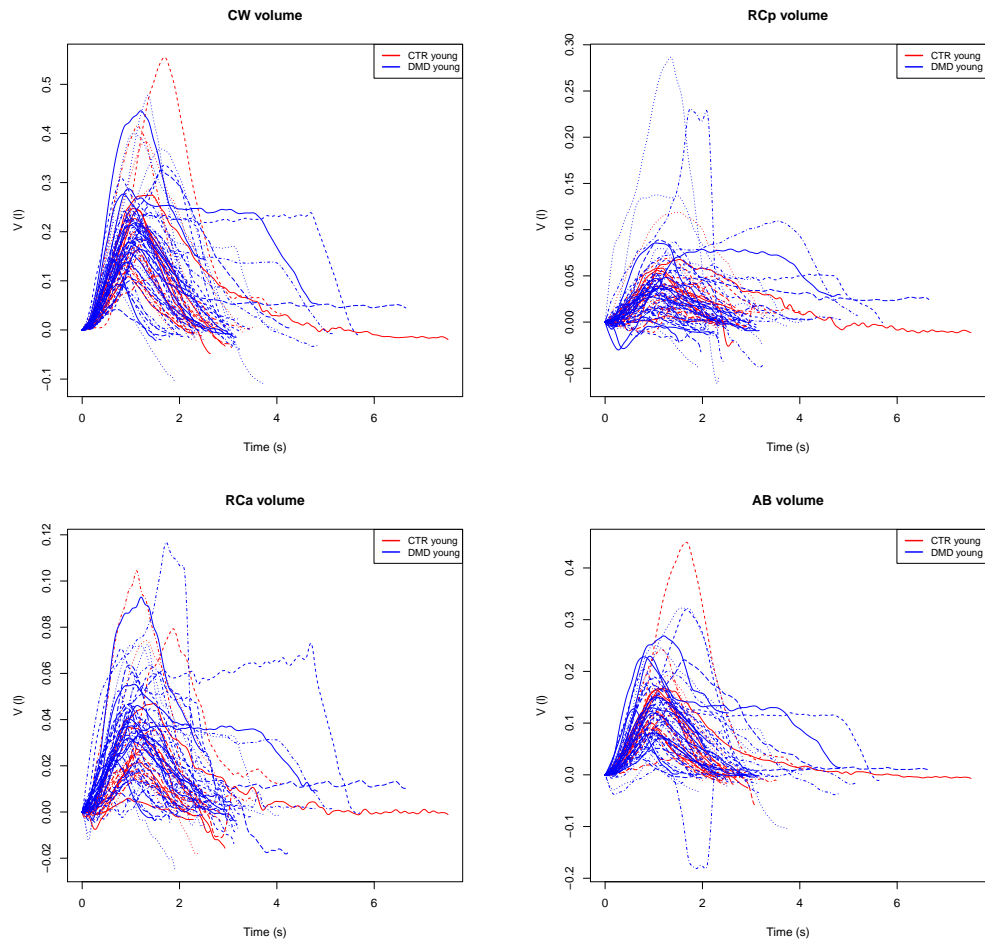


Figure 6.17: Median volume variations - DMD and CTR young samples



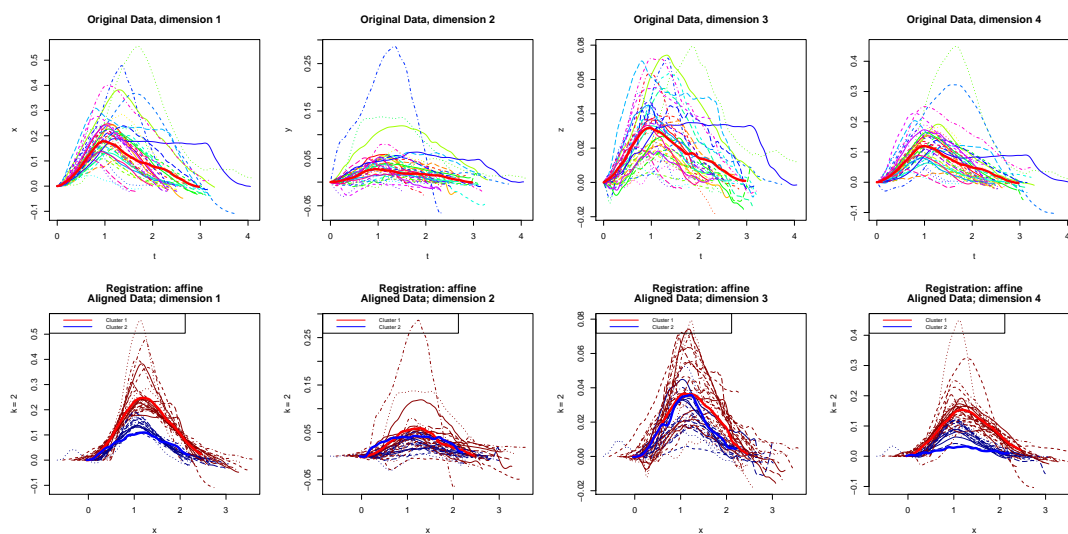


Figure 6.18: Classification results - DMD young and CTR young.

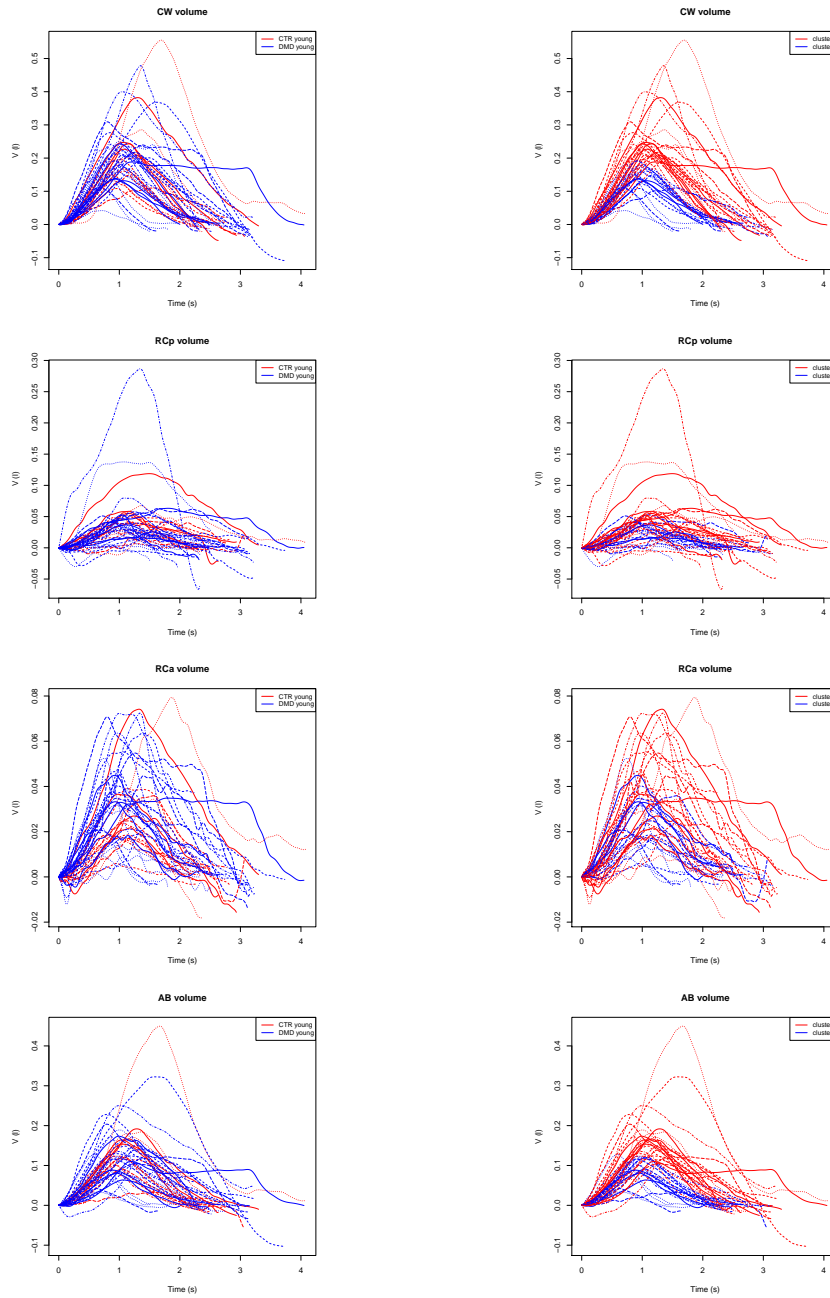


Figure 6.19: Classification results in terms of volume variation: the left column shows the true labels, the right column shows how breaths are classified.

## Compartments relative contribution

Medians are normalized dividing by their corresponding tidal volume, while only RCp and AB are considered for classification. Figure 6.20 shows the normalized curves. As before, there seem to be no clear distinction between healthy and DMD subjects. In total, during outlier detection 5 CTR and 21 DMD are removed.

Figures 6.21 and 6.22 show the results of classification. Again, the median breaths of the two groups are scattered in the two clusters, so it is not advisable to pick them as representative of the samples. As with volume variations, the difference in terms of relative contribution of the compartments in the breathing activity seems to be unrelated to the pathology. Based on these results, we can state that in the age range from 5 to 10 years old children affected by DMD still have breathing patterns which are substantially analogous to the ones of healthy homologous.

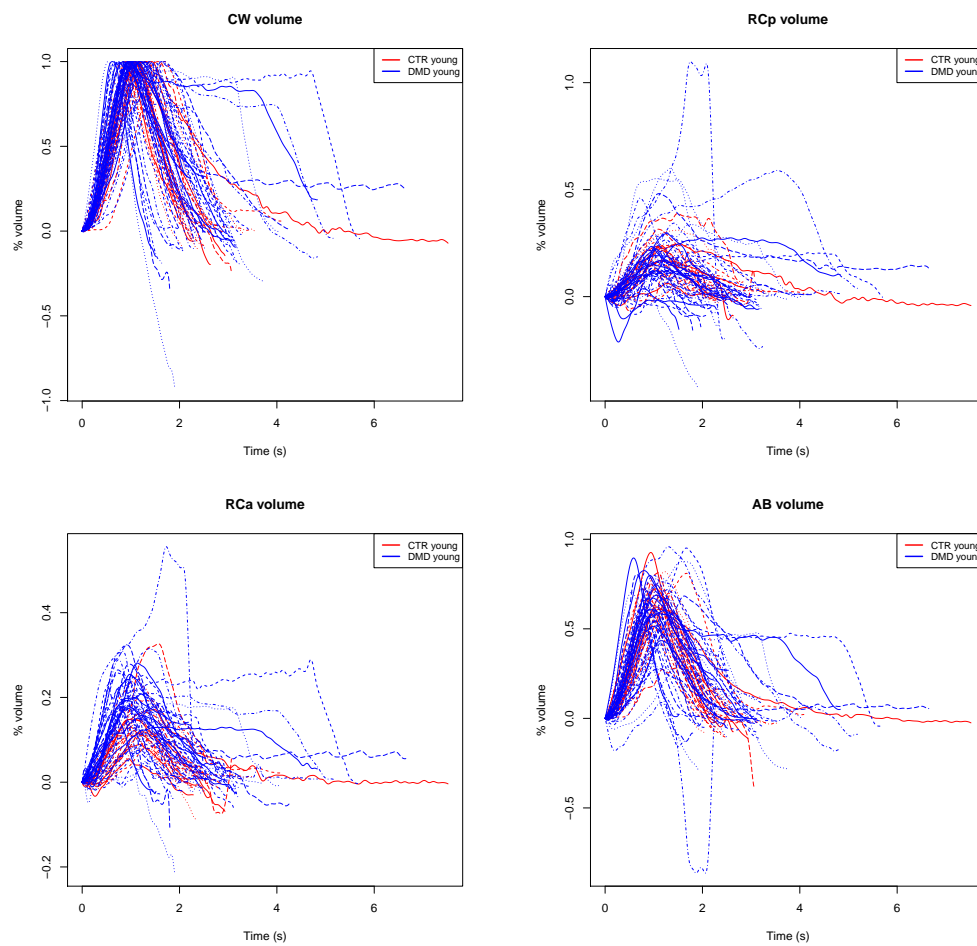


Figure 6.20: Normalized median breaths - DMD and CTR young samples.

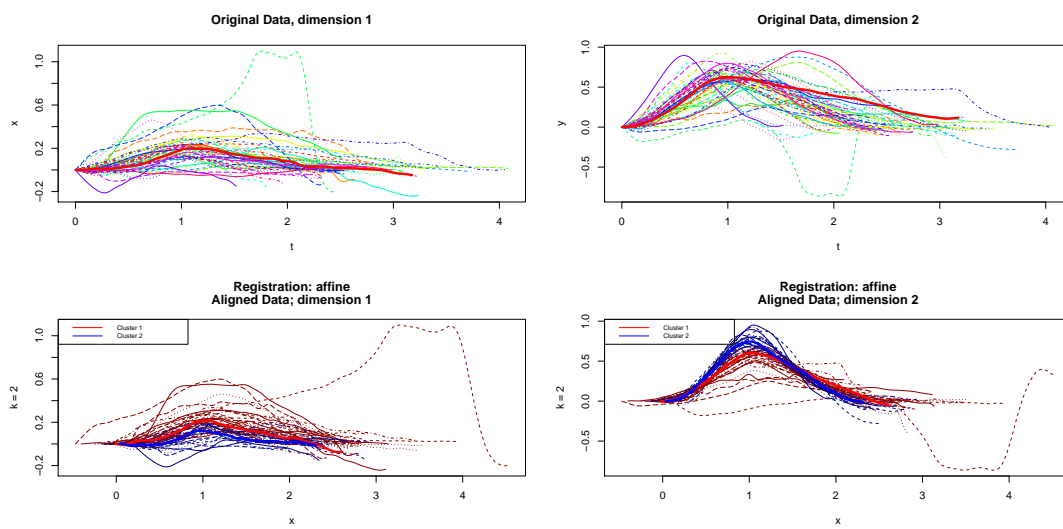


Figure 6.21: Classification results - DMD young and CTR young.

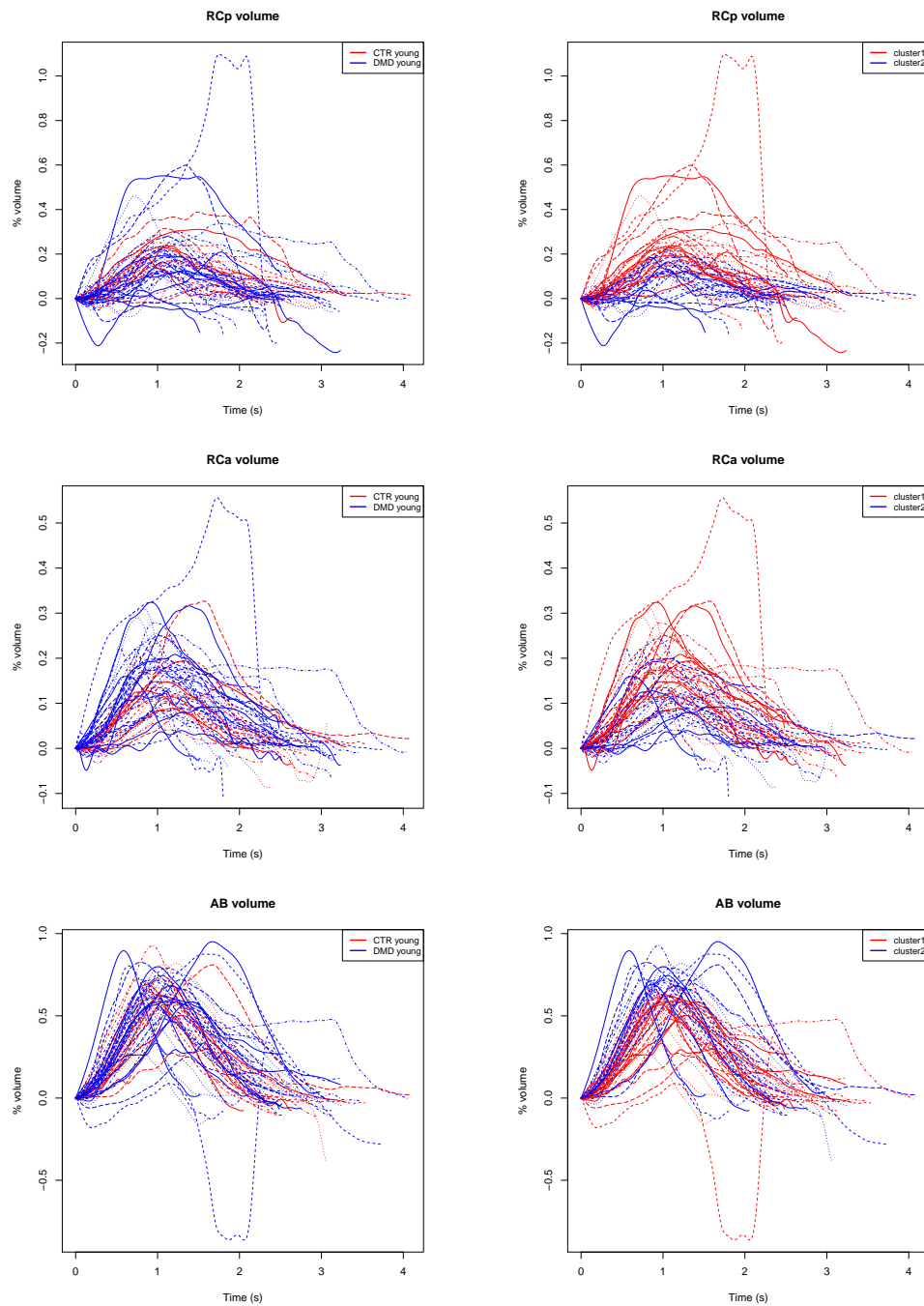


Figure 6.22: Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified.

## 6.5.2 Healthy adults and DMD adults

In this first analysis we consider the following samples:

- 21 healthy subjects between 18 and 25 years
- 40 DMD patients between 18 and 25 years

### Volume variations comparison

Median breaths are made to start from volume 0, and no other normalization is introduced. Figure 6.23 shows the so obtained curves, where different colours correspond to different samples. Healthy subjects have a significantly higher volume variation and a lower breathing frequency with respect to DMD subjects, which is to be expected since DMD weakens the respiratory muscles. Since the number of healthy subjects is much lower than DMD subjects and there are no evident outliers, we skip outlier detection in this case to avoid the removal of too many breaths in the healthy breaths, which would otherwise be considered as magnitude outliers. Figures 6.24 and 6.25 show the results of classification. Cluster 1 groups breaths with the lower volume, which mostly corresponds to the DMD adults, cluster 2 groups breaths with higher volume, which corresponds to the healthy adults. The confusion matrix is in Table 6.8, the two groups are identified with good precision. Healthy adults are characterized by a higher respiratory volume, while DMD patients have a lower volume variation associated with a higher respiratory rate.

		Classified	
		CTR adult	DMD adult
True	CTR adult	15	6
	DMD adult	6	34

Table 6.8: Confusion matrix for DMD and CTR adult samples - volume variation

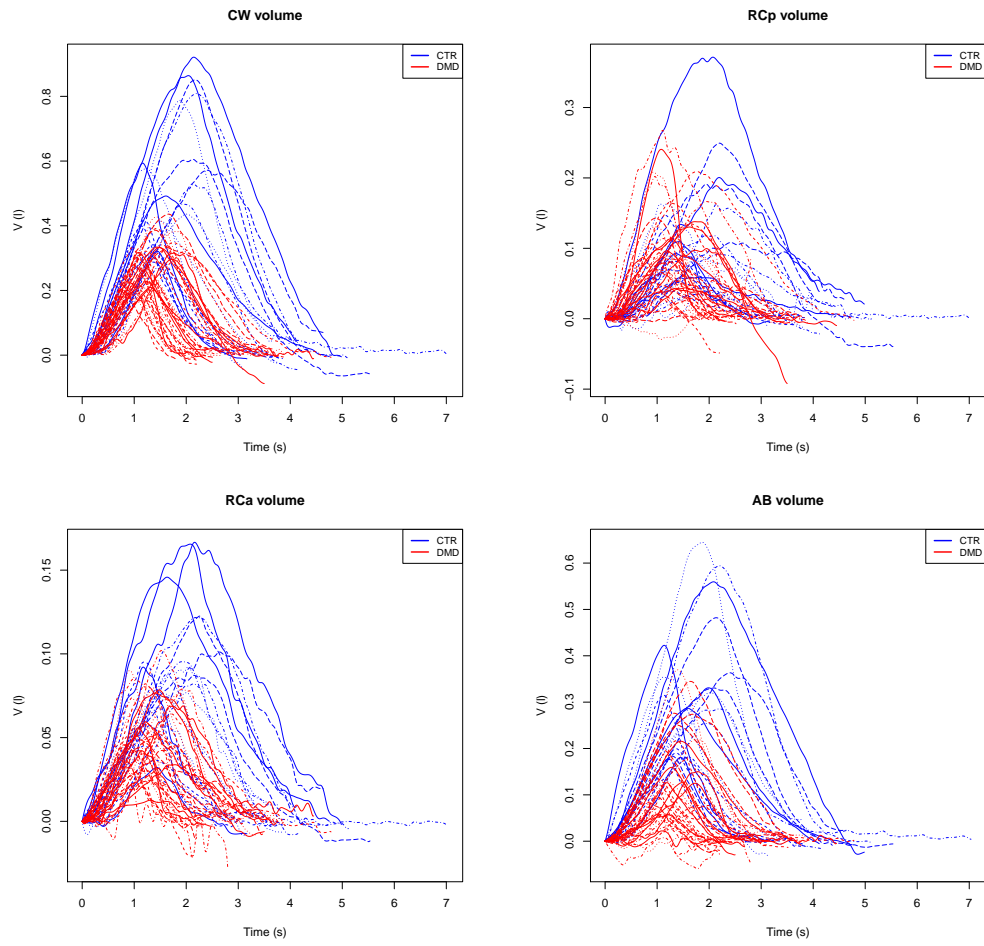


Figure 6.23: Median volume variation - DMD and CTR adult samples.

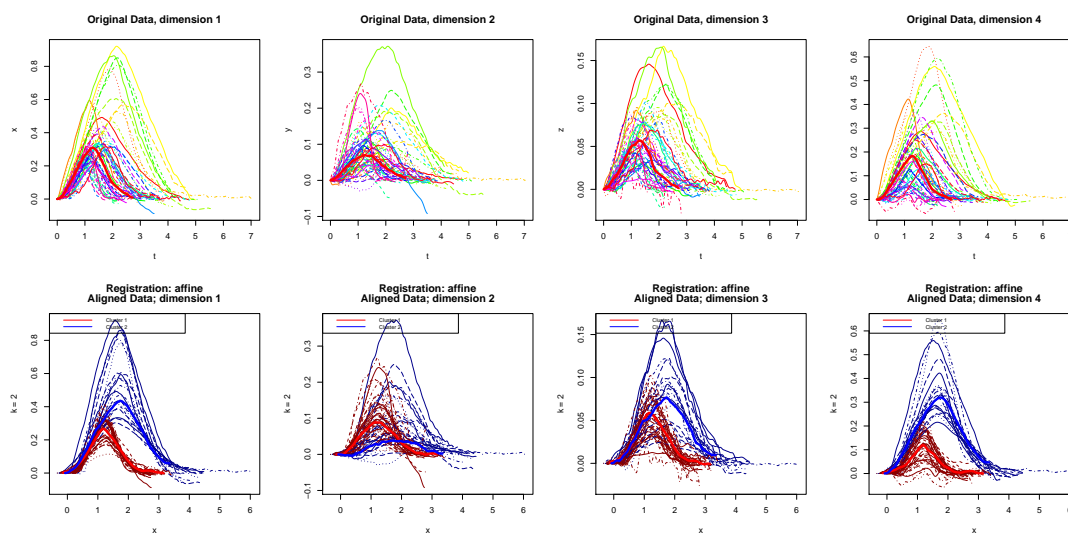


Figure 6.24: Classification results - DMD adult and CTR adult.



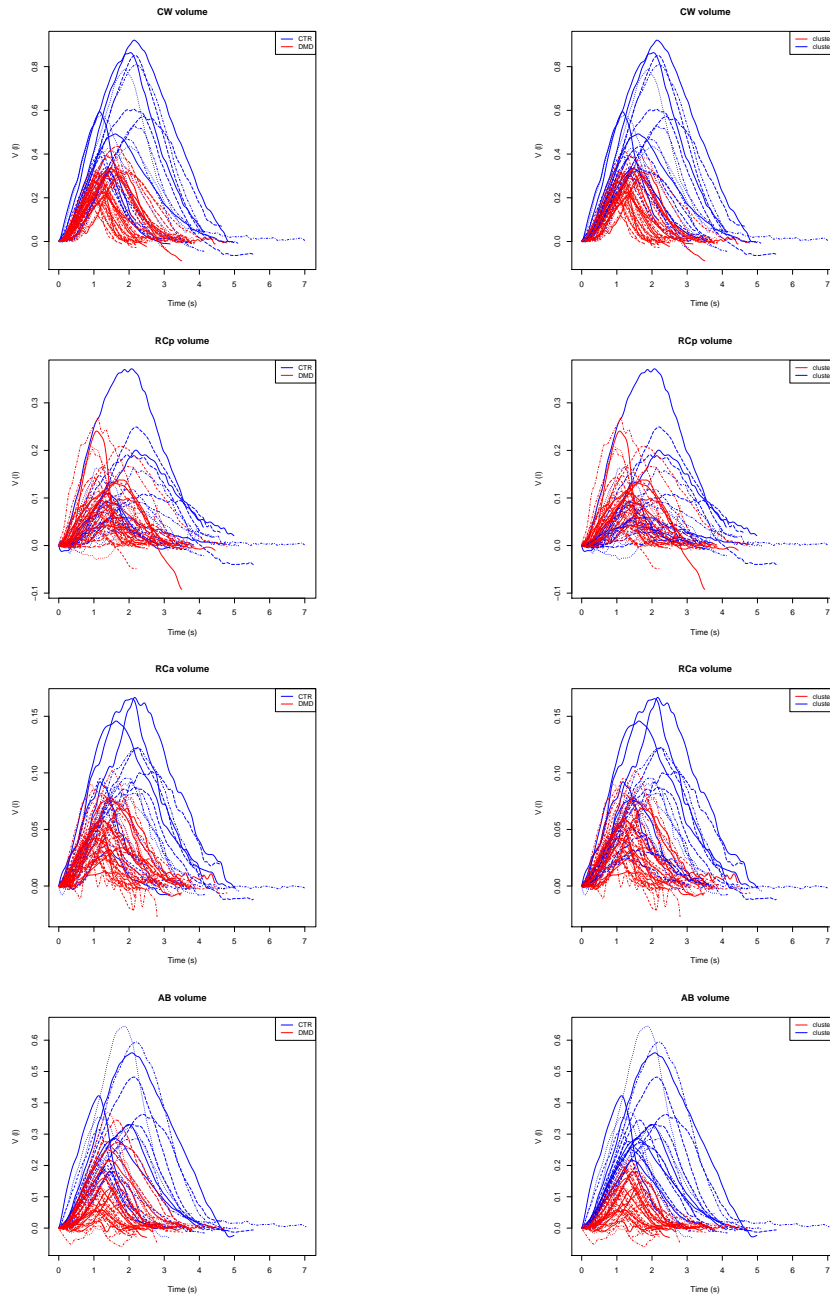


Figure 6.25: Classification results in terms of volume variation: the left column shows the true labels, the right column shows how breaths are classified.

## Compartments relative contribution

Medians are normalized dividing by their corresponding tidal volume, while only RCp and AB are considered for classification. Figure 6.26 shows the so obtained curves, where different colours correspond to different samples. On average the healthy adults have a higher duration of the breaths, which indicates lower breathing frequency. Moreover, they seem to have a smaller RCp contribution, which is reasonable since DMD tends to affect abdominal muscles first, so DMD patients have to resort more on pulmonary breathing (meaning higher RCp contribution for DMD subjects). Outlier detection finds 2 healthy and 5 DMD subjects as outliers. Figures 6.27 and 6.28 show the results of classification. The confusion matrix is provided in Table 6.9. Cluster 1 is characterized by a higher abdominal contribution than the rib cage, and encloses mostly healthy adults, while Cluster 2 presents a higher rib cage contribution and contains DMD patients. Abdominal respiration is coherent with the respiratory pattern described before for healthy supine subjects.

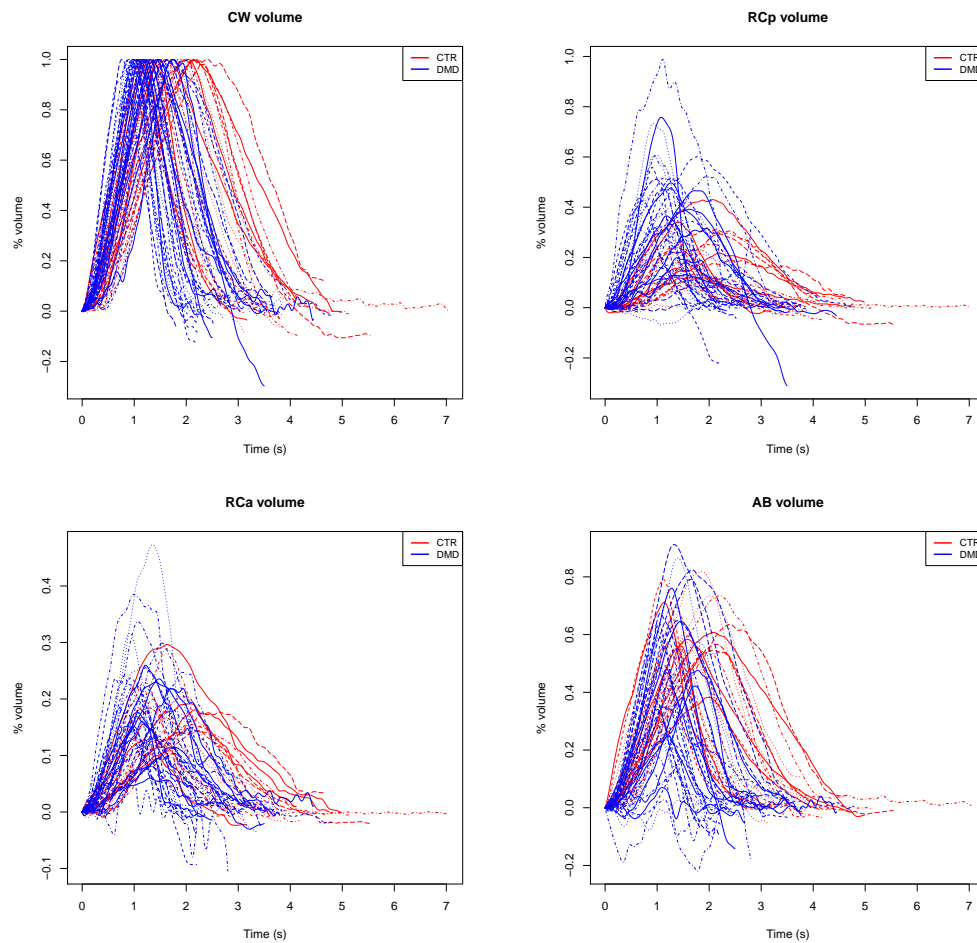


Figure 6.26: Normalized median breaths - DMD and CTR adult samples.

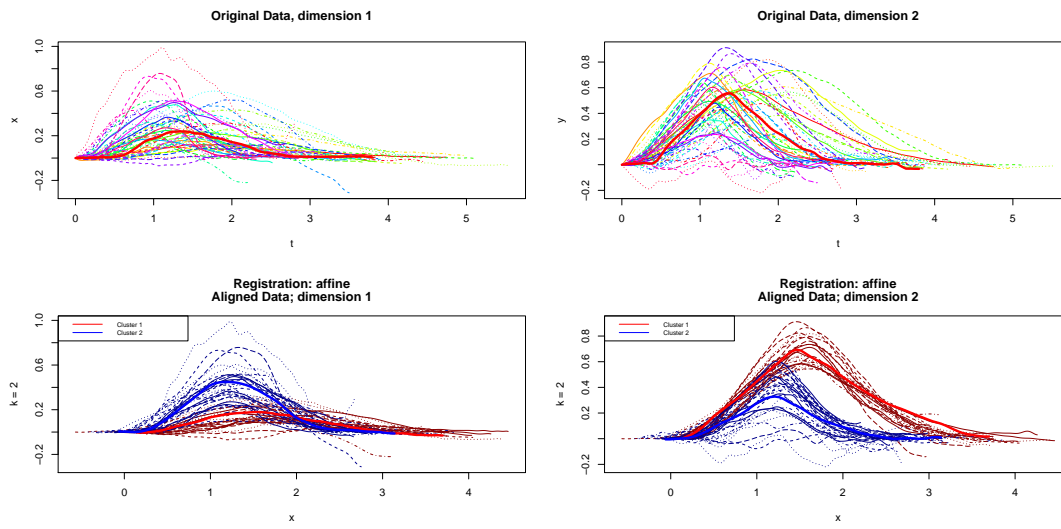


Figure 6.27: Classification results - DMD adult and CTR adult.

		Classified	
		CTR adult	DMD adult
True	CTR adult	14	5
	DMD adult	9	26

Table 6.9: Confusion matrix for DMD and CTR adult samples - compartment contribution

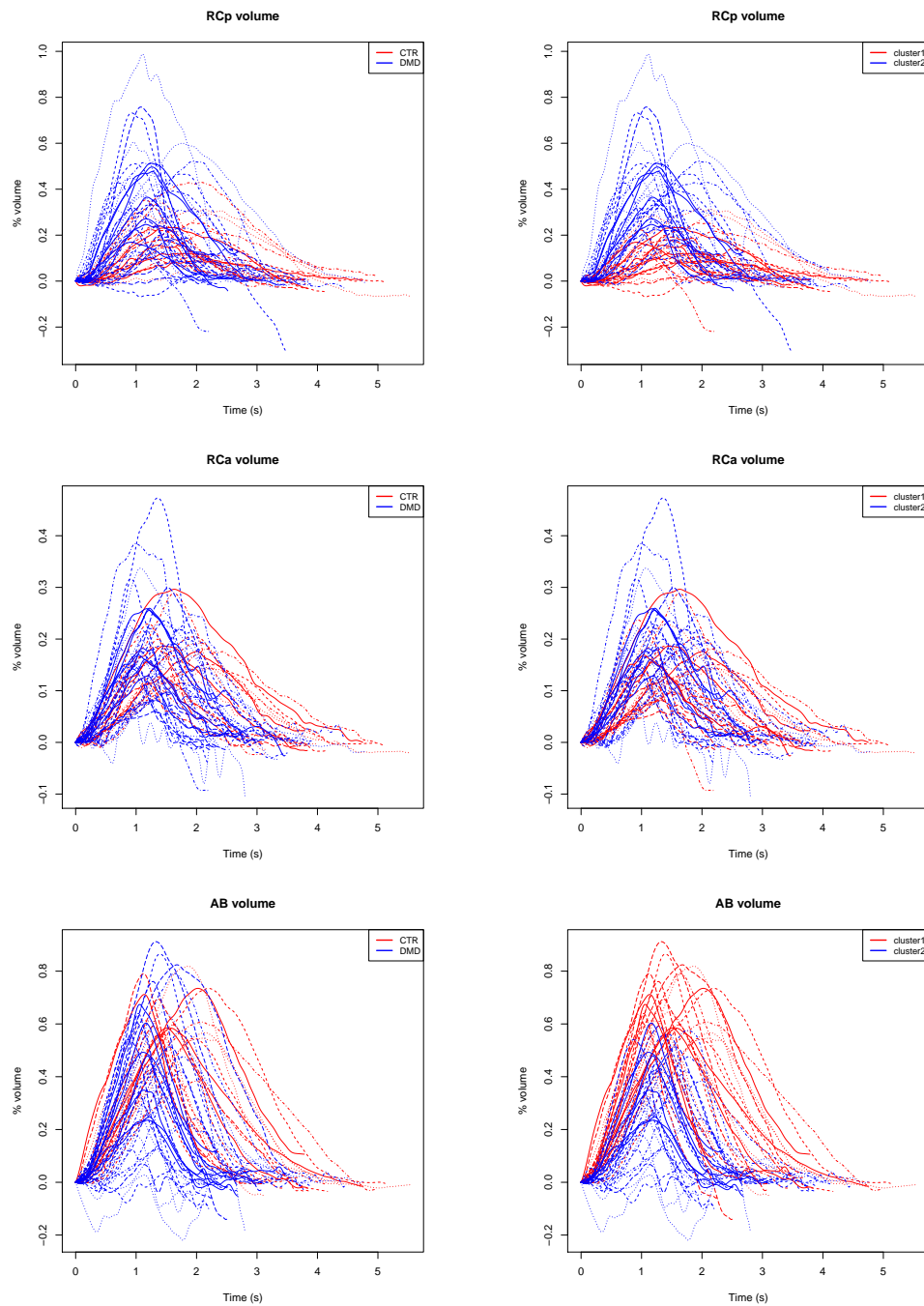


Figure 6.28: Classification results in terms of compartment contribution: the left column shows the true labels, the right column shows how breaths are classified.

## 6.6 Discussion

In this chapter, we declined the procedure described in the previous chapters to compare breathing patterns of healthy subjects in different postures and ages, and patients suffering from DMD. We summarize the main findings that we achieved:

- Respiration in healthy subjects in supine position is prevalently abdominal, while in seated position the higher contribution is given by the rib cage (Figure 6.29);
- Volume variations and breaths duration in healthy subjects are way higher from childhood to adulthood, but there is no significant difference between young adults and elder people (Figure 6.30a);
- Compartments contribution to respiration in healthy subjects changes in time, in particular people aged over 60 have a reduced rib cage contribution which is compensated by the abdomen (Figure 6.30b);
- Children affected by DMD display breathing patterns which are not significantly different from the ones of the healthy, both in terms of volume variations and compartments contribution;
- Adult DMD have a tidal volume which is systematically lower than the one of homologous healthy subjects, and a higher respiratory rate (Figure 6.32a);
- Supine respiration in adult DMD subjects is characterized by a balanced contribution of the rib cage and the abdomen, while in healthy subjects respiration is prevalently abdominal (Figure 6.32b);
- Adult DMD subjects have a tidal volume which is slightly higher than the one of DMD children, moreover adults have a decreased abdominal contribution (Figure 6.31).

These insights are coherent with findings arising from previous clinical studies developed with Optoelectronic Plethysmography, which demonstrated the influence of age and posture over the breathing pattern (see for example Massaroni et al., 2016 [17]). Moreover, results related to DMD patients confirm the outcome of the analysis in Lo Mauro et al., 2018 [14], who showed the significant change from normality of tidal volume, respiratory rate and abdominal contribution from childhood to adulthood.

Thus, the procedure developed in this work is able to reinforce the results obtained with the traditional breathing pattern methodologies, with a more rigorous analysis that requires less arbitrary interventions and allows to study breath curves as a whole instead of a set of scalar parameters.

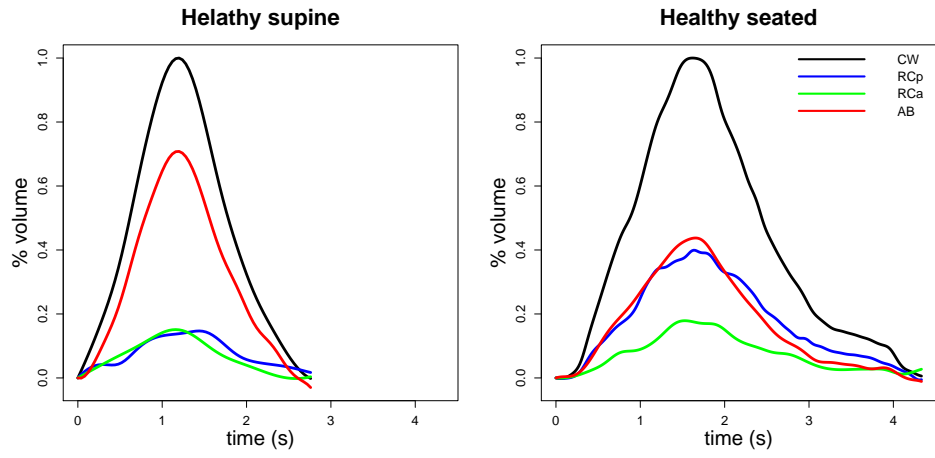
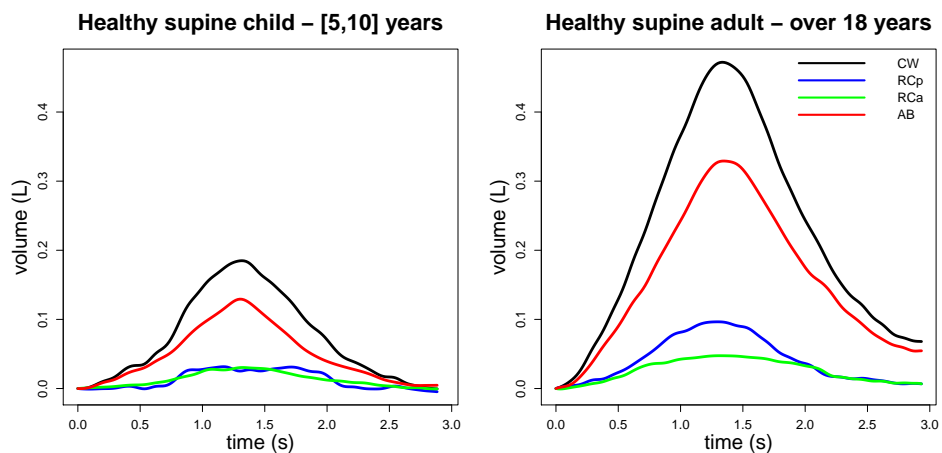
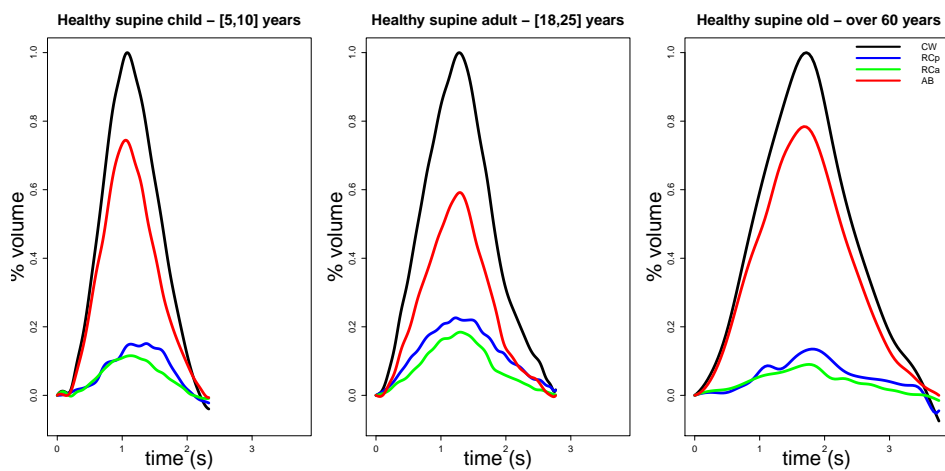


Figure 6.29: Position comparison - supine and seated healthy subjects

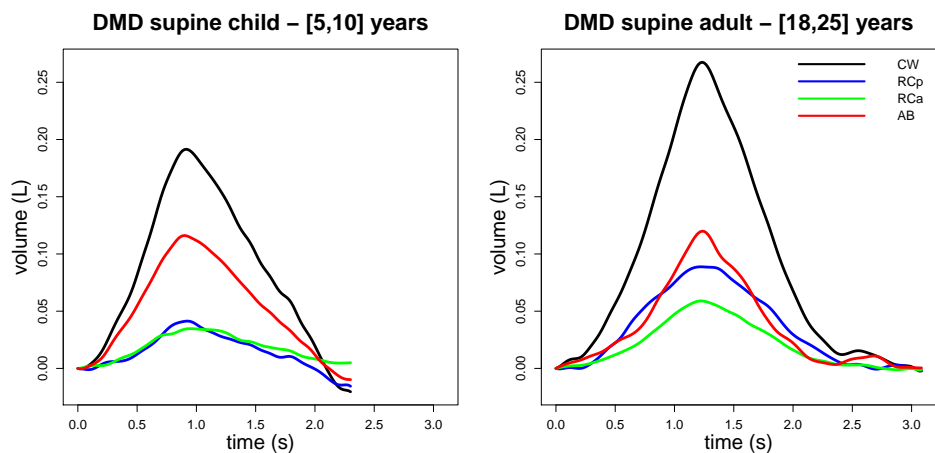


(a) Volume variations.

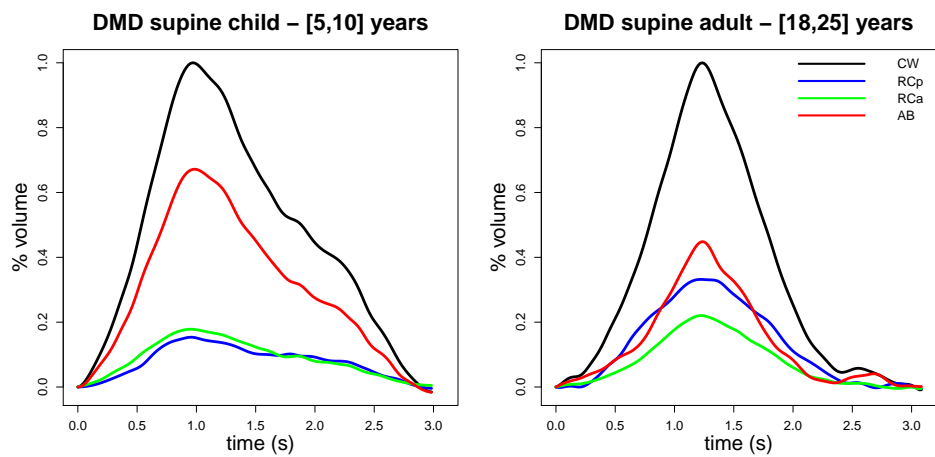


(b) Compartments percentage contribution.

Figure 6.30: Age comparison - supine healthy subjects



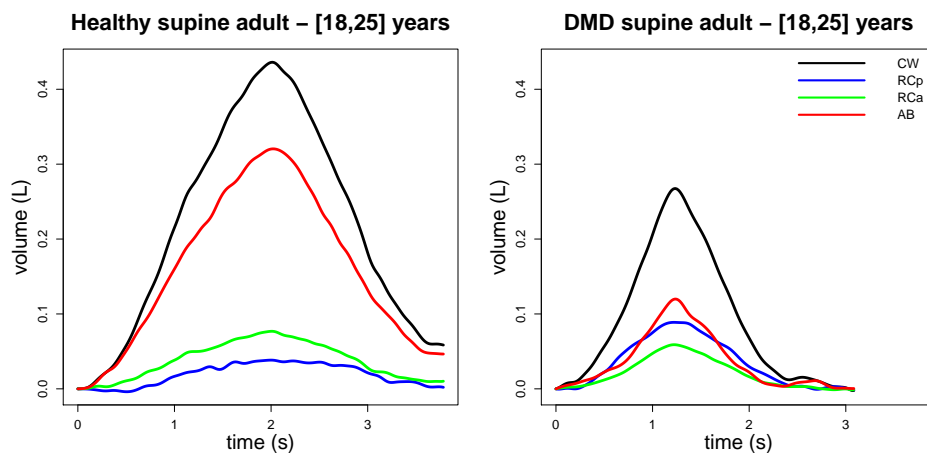
(a) Volume variations.



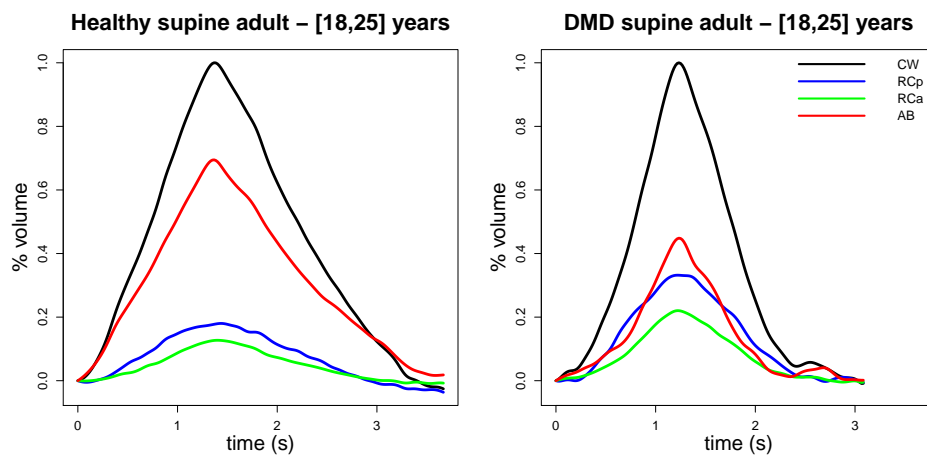
(b) Compartments percentage contribution.

Figure 6.31: Age comparison - DMD subjects





(a) Volume variations.



(b) Compartments percentage contribution.

Figure 6.32: Clinic condition comparison - Healthy and DMD adult

## Chapter 7

# Conclusion and further developments

In this work, we developed a rigorous and reproducible procedure for the statistical analysis of data coming from Optoelectronic Plethysmography, based on Functional Data Analysis techniques.

First, an algorithm was designed to separate breath curves based on the peculiar characteristics of OEP data tracks and B-spline smoothing, accounting for the presence of noisy minima, long-expiration breaths or pauses in the subject respiration. This algorithm allows for a fast minima detection even in long data acquisition, moreover parameters can be tuned based on the data acquisition characteristics, especially the sampling frequency.

Secondly, the concept of outlier breath was formalized in terms of breath duration, magnitude and shape. An algorithm for automatic outlier detection was developed, combining scalar duration boxplot and functional techniques such as multivariate functional boxplot and multivariate outliergram. An experiment was designed to assess the outlier detection algorithm performance in extreme conditions, and to determine the best weight configuration to use in multivariate functional depths. The algorithm proved to be quite robust even in these worst cases, while the best results were achieved through uniform weights.

The two pre-processing algorithms were used to build functional datasets of smooth breath curves from single acquisitions. Multivariate K-Medoids with Alignment algorithm was employed both to extract the median curve of these datasets as a representative curve of the breathing pattern of the subject, or to analyse group structures in breathing patterns, both in terms of absolute value and shape. Different algorithm settings were proposed in relation to different clinical and experimental situations, such as breathing pattern variations during exercise. Application to real case studies shows the effectiveness of the algorithm in capturing cluster structures based on relevant physiological parameters.

Finally, a new way to compare breathing patterns of different subjects was proposed, based on the adaptation of the intra-subject outlier detection and clustering procedure previously defined. Normalized median breaths are used each as representatives of the corresponding subjects. K-Medoids with Alignment over volume variations and relative contribution of compartments has been used to discriminate healthy subjects in different positions and age classes, with a comparison to patients suffering from Duchenne Muscular Dystrophy at different ages. Classification provided physiologically interpretable results, which also reinforced evidence obtained in previous clinical

studies [3] [14].

Many further developments of this work can be possible:

- Using the same methodology developed in this thesis, breathing pattern comparison between healthy subjects and patients with different pathologies other than DMD can be performed;
- This work has been focused mainly over the characterization of quiet breathing, while outliers were discarded. Differences in outliers like vital capacities extracted by different subjects, or in coughs, which are often used to assess a patient's breathing functionality, could be examined [17];
- Plethysmography can be implemented by means of wearable devices, which allow for a continuous monitoring of breathing activity during the daily activities. On-line processing techniques may be developed to analyze data coming from this source, and possibly to identify patterns related to different postures, activities and others;
- Multivariate K-medoids algorithm weights all the curve dimensions equally. An improvement would be to use a weighted K-medoids, giving different importance to the different chest wall compartments;
- K-medoid Alignment for dependent curves could be used to take into account temporal dependence in clustering [1].

# Bibliography

- [1] Abramowicz, K., Arnqvist, P., Secchi, P., Sjøstedt de Luna, S., Vantini, S., Vitelli, V. (2017). Clustering misaligned dependent curves applied to varved lake sediment for climate reconstruction. *Stochastic Environmental Research and Risk Assessment*, 31(1):71–85.
- [2] Aliverti A. (2016). Physiology masterclass: The respiratory muscles during exercise. *Breathe* 2016; 12: 165–168.
- [3] Aliverti, A., Dellacà, R., Pelosi, P., Chiumello, D., Gattinoni, L., Pedotti, A. (2001). Compartmental Analysis of Breathing in the Supine and Prone Positions by Optoelectronic Plethysmography, *Annals of Biomedical Engineering*, Vol. 29, pp. 60–70.
- [4] Aliverti, A. and Pedotti, A. (2016). *Mechanics of Breathing*, 149 DOI 10.1007/978-88-470-5647-3\_11, Springer-Verlag Italia.
- [5] American Thoracic Society website (Patients resources): <https://www.thoracic.org/patients/patient-resources/>
- [6] Arribas-Gil, A. and Romo, J. (2014). Shape outlier detection and visualization for functional data: The outliergram. *Biostatistics*, 15(4):603–619.
- [7] Cystic Fibrosis Foundation website (About Cystic Fibrosis): <https://www.cff.org/What-is-CF/About-Cystic-Fibrosis/>
- [8] Hastie, T. , Tibshirani, R. and Friedman, J.(2008). *The Elements of Statistical Learning*, Springer, 2nd ed.
- [9] Ieva, F. and Paganoni, A. M. (2013). Depth Measures for Multivariate Functional Data, *Communications in Statistics - Theory and Methods*, 42:7, 1265-1276.
- [10] Ieva, F., and Paganoni, A.M. (2017). Component-wise outlier detection methods for robustifying multivariate functional samples. *Stat Papers* 61, 595–614.
- [11] Ieva, F., Paganoni, A. M., Romo, J., Tarabelloni, N. roahd Package: Robust Analysis of High Dimensional Data, *R Journal*.
- [12] Levitzky, Michael G. (2013). *Pulmonary physiology* (Eighth ed.). New York: McGraw-Hill Medical.

- [13] LoMauro, A., Aliverti, A., Mastella, C., Arnoldi, M.T., Banfi, P., Baranello, G. (2016). Spontaneous Breathing Pattern as Respiratory Functional Outcome in Children with Spinal Muscular Atrophy (SMA). PLoS ONE 11(11):e0165818. <https://doi.org/10.1371/journal.pone.0165818>
- [14] LoMauro, A., Romei, M., Gandossini, S., Pascuzzo, R., Vantini, S., D'Angelo, M. G., Aliverti, A. (2018). Evolution of respiratory function in Duchenne muscular dystrophy from childhood to adulthood. Eur Respir J 2018; 51: 1701418 <https://doi.org/10.1183/13993003.01418-2017>.
- [15] López-Pintado, S. and Romo, J. (2009). On the concept of depth for functional data. Journal of the American Statistical Association, 104(486):718 – 734.
- [16] Marieb, E. N. , Hoehn, K., (2015). Human Anatomy and Physiology , Pearson Education, 10th ed.
- [17] Massaroni, C., Carraro, E., Vianello, A., Miccinilli, S., Morrone, M., Levai, I.K., Schena, E., Saccomandi, P., Sterzi, S., Dickinson, J.W., Winter, S., Silvestri, S. (2017). Optoelectronic Plethysmography in Clinical Practice and Research: A Review. Respiration 2017;93:339–354.
- [18] Ramsay, J.O. and Silverman, B.W. (2005). Functional Data Analysis, Springer, 2nd ed.
- [19] Sangalli, L. M. , Secchi, P., Vantini, S., Vitelli, V. (2010). K-mean alignment for curve clustering. Computational Statistics and Data Analysis, 54 (5), 1219-1233.
- [20] Sangalli, L.M., Secchi, P., Vantini, S., 2014. "Analysis of AneuRisk65 data: K-mean Alignment". Electronic Journal of Statistics, Special Section on "Statistics of Time Warpings and Phase Variations", Vol. 8, No. 2, 1891-1904.
- [21] Sun, Ying and Genton, Marc. (2010). Functional Boxplot. Journal of Computational and Graphical Statistics. 20. 10.2307/23110490.
- [22] Wikipedia. Breathing. <https://en.wikipedia.org/wiki/Breathing>
- [23] Wikipedia. Glycogen storage disease. [https://en.wikipedia.org/wiki/Glycogen\\_storage\\_disease](https://en.wikipedia.org/wiki/Glycogen_storage_disease)
- [24] Wikipedia. Spirometry. <https://en.wikipedia.org/wiki/Spirometry>
- [25] Zito, A. (2017). Optimized K-mean Alignment algorithm for clustering functional data: application to brain tractography. PoliTesi <http://hdl.handle.net/10589/137327>.



# Appendix A

## Code

In this appendix the full code used for OEP data analysis is provided.

### A.1 Prerequisites

The results of this thesis were obtained with R. It is advisable to use an R version of at least 3.5.1.

The following R packages are needed: `fda`, `roahd`, `splus2R` and `fdakmapp`. The first three are available on CRAN and can be installed with `install.packages("packagename")`. The optimized K-Mean Alignment algorithm implementation by Alessandro Zito [25] is available on GitHub in the form of the R package called `fdakmapp`. Code can be found at <https://github.com/zitale/fdakmapp> in the branch `pac`s. To install it, download the `.zip` folder, unzip it and type on the R console

```
install.packages("/path/to/package/folder", type='source', repos=NULL)
```

### A.2 Subject analysis - procedure

The following code is the one used to find the median or cluster breaths from the data file of a single patient.

```
# load libraries
library(fda)
library(roahd)
library(fdakmapp)
library(splus2R)

# load functions
setwd("/path/to/R/scripts")
source("smooth_breath.R")
source("localmin.R")
source("table_breaths.R")
source("rescale.R")
source("outlier_detection.R")
```

```

source("plot.breaths.byclusters.R")
source("plot_compartments.R")

# load data
setwd("/path/to/datafile")

name='name'

data=read.table(paste(name,".dat",sep=''),header = F)

time=data[,1]
volrcp=data[,2]
volrca=data[,3]
volab=data[,4]
voltot=data[,5]

x11()
plot(time, voltot, col='black', type = 'l', main='CW volume',
      xlab='time (s)', ylab='volume (L)')

x11()
plot(time, volrcp, col='black', type = 'l', main='RCp volume',
      xlab='time (s)', ylab='volume (L)')

x11()
plot(time, volrca, col='black', type = 'l', main='RCa volume',
      xlab='time (s)', ylab='volume (L)')

x11()
plot(time, volab, col='black', type = 'l', main='AB volume',
      xlab='time (s)', ylab='volume (L)')

write_centers=FALSE # TRUE to save results

min=find_local_min(time, voltot, peak_span=101, grid_coef=10, step=10,
                    plot=1,spiky_min = T)

volumes.tot=table_breaths(time, voltot, min$minidx)
volumes.rcp=table_breaths(time, volrcp, min$minidx)
volumes.rca=table_breaths(time, volrca, min$minidx)
volumes.ab=table_breaths(time, volab, min$minidx)

d=min$deltas;
times=volumes.tot$times
for(j in seq(1,length(d)))
times[,j]=times[,j]-times[1,j];

smoothed=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                ncol=dim(volumes.tot$breaths)[2])

```



```

smoothed1=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                 ncol=dim(volumes.tot$breaths)[2])

for(j in 1:dim(volumes.tot$breaths)[2]){
  vec=which(!is.na(volumes.tot$breaths[,j]))
  sm=smooth_breath(volumes.tot$times[vec,j], volumes.tot$breaths[vec,j],
                  grid_coef=5,plot = 0) #smooth breaths one by one
  smoothed[vec,j]=sm$smoothed_curve
  smoothed1[vec,j]=sm$der1
}

smoothed_rcp=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                   ncol=dim(volumes.tot$breaths)[2])
smoothed_rcp1=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                    ncol=dim(volumes.tot$breaths)[2])

for(j in 1:dim(volumes.tot$breaths)[2]){
  vec=which(!is.na(volumes.rcp$breaths[,j]))
  sm=smooth_breath(volumes.tot$times[vec,j], volumes.rcp$breaths[vec,j],
                  grid_coef=5,plot = 0) #smooth breaths one by one
  smoothed_rcp[vec,j]=sm$smoothed_curve
  smoothed_rcp1[vec,j]=sm$der1
}

smoothed_rca=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                   ncol=dim(volumes.tot$breaths)[2])
smoothed_rca1=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                    ncol=dim(volumes.tot$breaths)[2])

for(j in 1:dim(volumes.tot$breaths)[2]){
  vec=which(!is.na(volumes.rca$breaths[,j]))
  sm=smooth_breath(volumes.tot$times[vec,j], volumes.rca$breaths[vec,j],
                  grid_coef=5,plot = 0) #smooth breaths one by one
  smoothed_rca[vec,j]=sm$smoothed_curve
  smoothed_rca1[vec,j]=sm$der1
}

smoothed_ab=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                  ncol=dim(volumes.tot$breaths)[2])
smoothed_ab1=matrix(NA, nrow=dim(volumes.tot$breaths)[1],
                   ncol=dim(volumes.tot$breaths)[2])

for(j in 1:dim(volumes.tot$breaths)[2]){
  vec=which(!is.na(volumes.ab$breaths[,j]))
  sm=smooth_breath(volumes.tot$times[vec,j], volumes.ab$breaths[vec,j],
                  grid_coef=5,plot = 0) #smooth breaths one by one
  smoothed_ab[vec,j]=sm$smoothed_curve
  smoothed_ab1[vec,j]=sm$der1
}

```

```

x11(); matplot(smoothed, type='l', main='Breaths - total volume',
  xlab='time (s)', ylab='volume (L)')
x11(); matplot(smoothed_rcp, type='l', main='Breaths - RCp volume',
  xlab='time (s)', ylab='volume (L)')
x11(); matplot(smoothed_rca, type='l', main='Breaths - RCa volume',
  xlab='time (s)', ylab='volume (L)')
x11(); matplot(smoothed_ab, type='l', main='Breaths - AB volume',
  xlab='time (s)', ylab='volume (L)')

x11(); matplot(smoothed1, type='l', main='First Derivatives - total volume',
  xlab='time (s)', ylab='')
x11(); matplot(smoothed_rcp1, type='l', main='First Derivatives - RCp volume',
  xlab='time (s)', ylab='')
x11(); matplot(smoothed_rca1, type='l', main='First Derivatives - RCa volume',
  xlab='time (s)', ylab='')
x11(); matplot(smoothed_ab1, type='l', main='First Derivatives - AB volume',
  xlab='time (s)', ylab='')

#### Outlier detection ####

filtered=outlier_detection(times, smoothed, smoothed_rcp, smoothed_rca,
  smoothed_ab, plot_option=1)

plot_outliers(time, voltot, volumes.tot, filtered$time.outliers.idx,
  filtered$magnitude.outliers.idx, filtered$shape.outliers.idx,
  main='Outlier breaths - CW Volume')
plot_outliers(time, volrcp, volumes.rcp, filtered$time.outliers.idx,
  filtered$magnitude.outliers.idx, filtered$shape.outliers.idx,
  main='Outlier breaths - RCp Volume')
plot_outliers(time, volrca, volumes.rca, filtered$time.outliers.idx,
  filtered$magnitude.outliers.idx, filtered$shape.outliers.idx,
  main='Outlier breaths - RCa Volume')
plot_outliers(time, volab, volumes.ab,
  filtered$time.outliers.idx,
  filtered$magnitude.outliers.idx, filtered$shape.outliers.idx,
  main='Outlier breaths - AB Volume')

filtered.times=filtered$filtered.times
filtered.Vtot=filtered$filtered.Vtot
filtered.Vrcp=filtered$filtered.Vrcp
filtered.Vrca=filtered$filtered.Vrca
filtered.Vab=filtered$filtered.Vab
outliers=filtered$outliers.idx

#### Clustering ####

# If absolute value is relevant-> groupBy.abs=TRUE, use L2.
# Use Pearson otherwise (groupBy.abs=FALSE)

```

```

cat("Insert a boolean: TRUE if absolute value is relevant (use L2 distance),
    FALSE if only shape is relevant (use d1.pearson). Press ENTER to continue.")
groupBy. abs=scan("", what=logical(), nmax=1)

if(groupBy. abs){

    datavec = c(c(t(filtered.Vtot)), c(t(filtered.Vrcp)), c(t(filtered.Vrca)),
    c(t(filtered.Vab)))
    arr=array(datavec, c(dim(filtered.Vtot)[2], dim(filtered.Vtot)[1], 4))

    }else{

    filtered.Vtot1=smoothed1[, -outliers]
    filtered.Vrcp1=smoothed_rcp1[, -outliers]
    filtered.Vrca1=smoothed_rca1[, -outliers]
    filtered.Vab1=smoothed_ab1[, -outliers]

    filtered.Vtot1=filtered.Vtot1[1:dim(filtered.times)[1],]
    filtered.Vrcp1=filtered.Vrcp1[1:dim(filtered.times)[1],]
    filtered.Vrca1=filtered.Vrca1[1:dim(filtered.times)[1],]
    filtered.Vab1=filtered.Vab1[1:dim(filtered.times)[1],]

    datavec = c(c(t(filtered.Vtot1)), c(t(filtered.Vrcp1)), c(t(filtered.Vrca1)),
    c(t(filtered.Vab1)))
    arr=array(datavec, c(dim(filtered.Vtot1)[2], dim(filtered.Vtot1)[1], 4))

}

timet=t(filtered.times)

original.times=volumes.tot$times[, -outliers] # for graphics
original.times=original.times[1:dim(filtered.times)[1],]

distance=ifelse(groupBy. abs==1, 'l2', 'pearson')
warp= 'affine'

find.median=F # FALSE to find clusters, TRUE to find just the median

if(find.median){

    nclust=1

    # align and compute the median
    res=kmap(timet, arr, n_clust=nclust, warping_method=warp,
    center_method='medoid', similarity_method=distance,
    comp_original_center = T)

    kmap_show_results(res)
}

```

```

# pick and plot the median breath
ind=which(res$similarity.final==0)

centroid1=cbind(filtered.times[,ind], filtered.Vtot[,ind],
                filtered.Vrcp[,ind], filtered.Vrca[, ind],
                filtered.Vab[,ind])

dcentroid1=cbind(filtered.times[,ind], filtered.Vtot1[,ind],
                filtered.Vrcp1[,ind], filtered.Vrcal[, ind],
                filtered.Vab1[,ind])

# plot the centroid
plot_compartments(timet, filtered.Vtot, filtered.Vrcp, filtered.Vrca,
                 filtered.Vab, centroid_idx = ind,
                 labels = res$labels, n_clust=1,
                 titles = c('Median breath - CW and compartments'))

# save the median
if(write_centers)
{
  write.table(centroid1, file=paste0(name, '_median.txt'))
}

}else{

# choose number of clusters —> elbow

cat("Insert the number of clusters to try. Press ENTER to continue.")
K=scan("", what=integer(), nmax=1)
checksim=numeric(K)

for(j in 1:K){

  check=kmap(timet, arr, n_clust=j, warping_method=warp,
            center_method='medoid', similarity_method=distance,
            comp_original_center = T)

  checksim[j]=mean(check$similarity.final)
}

x11()
plot(checksim, type='l', xlab='Cluster labels', ylab='Mean similarities',
     main=paste0('Mean Similarity - ',warp,sep=''))
points(checksim, pch=1, col='blue',lwd=2)

# select K run-time
cat("Insert the selected number of clusters to use. Press ENTER to continue.")
nclust=scan("", what=integer(), nmax=1)

```

```

# cluster
res=kmap(timet, arr, n_clust=nclust, warping_method=warp,
center_method='medoid', similarity_method=distance,
comp_original_center = T)

kmap_show_results(res)

labels=res$labels
table(labels)

# plots
col=plot.breaths.byclusters(time, voltot, original.times, filtered.Vtot, labels)
plot.breaths.byclusters(time, volrcp, original.times, filtered.Vrcp, labels)
plot.breaths.byclusters(time, volrca, original.times, filtered.Vrca, labels)
plot.breaths.byclusters(time, volab, original.times, filtered.Vab, labels)

x11();matplot(filtered.Vtot, col=col$col.groups, type='l',
main='Clustered breaths - CW', xlab = 'time')
x11();matplot(filtered.Vrcp, col=col$col.groups, type='l',
main='Clustered breaths - RCp', xlab = 'time')
x11();matplot(filtered.Vrca, col=col$col.groups, type='l',
main='Clustered breaths - RCa', xlab = 'time')
x11();matplot(filtered.Vab, col=col$col.groups, type='l',
main='Clustered breaths - AB', xlab = 'time')

if(groupBy. abs){
  ind=which(res$similarity.final==0)
 }else{
  ind=which(res$similarity.final >=(-1-10^-15) &
res$similarity.final <=(-1+10^-15))
 }

labels[ind]

# take the 2 most numerous groups and plot their centroids
selected_centroid1=sort(ind)[1]
selected_centroid2=sort(ind)[2]

centroid1=cbind(filtered.times[, selected_centroid1],
filtered.Vtot[, selected_centroid1], filtered.Vrcp[, selected_centroid1],
filtered.Vrca[, selected_centroid1],
filtered.Vab[, selected_centroid1])
centroid2=cbind(filtered.times[, selected_centroid2],
filtered.Vtot[, selected_centroid2], filtered.Vrcp[, selected_centroid2],
filtered.Vrca[, selected_centroid2],
filtered.Vab[, selected_centroid2])

# plot the centroids (2 clusters)
plot_compartments(timet, filtered.Vtot, filtered.Vrcp,

```

```

        filtered.Vrca, filtered.Vab,
        centroid_idx = ind, labels = res$labels,
        n_clust=nclust,
        titles = c('Centroid 1', 'Centroid 2'))

# save the results (2 clusters)
if(write_centers){
  write.table(centroid1, file=paste0(name, '_centroid1.txt'))
  write.table(centroid2, file=paste0(name, '_centroid2.txt'))
}
}

```

Function `kmap` of `fdakmapp` package performs K-mean or K-medoid with Alignment algorithms. We use it with the following input:

- **timet**: numeric matrix [n.func X grid.size]. The abscissa values where each function is evaluated. **n.func**: number of functions in the dataset. **grid.size**: maximal number of abscissa values where each function is evaluated.
- **arr**: array [n.func X grid.size X d]. Evaluations of the set of original functions on the abscissa grid **x**. **n.func**: number of functions in the dataset. **grid.size**: maximal number of abscissa values where each function is evaluated. **d**: number of function components, i.e. each function is a d-dimensional curve.
- **n\_clust**: scalar. Required number of clusters. Default value is 1. Note that if **n\_clust=1** `kma` performs only alignment without clustering.
- **warping\_method**: character. Type of alignment required. The implemented options are: "affine", "dilation", "shift" and "noalign". In our code, we use 'affine'.
- **center\_method**: character. Type of clustering method to be used. Possible choices are: 'mean', 'medoid' and 'pseudomedoid'. Default value is 'mean'. In our code, we use 'medoid'.
- **similarity\_method**: character. Required similarity measure. Possible choices are: 'pearson', 'l2'. Default value is 'pearson'. In order to compute the median, we use 'l2'.
- **comp\_original\_center**: boolean. If **comp\_original\_center=TRUE** the initial center with relative dissimilarities is computed otherwise this step is skipped.

The output of `kmap` is:

- **x.center.orig**: numeric vector of length **n\_out**. Abscissa of the center computed if **comp\_original\_center=TRUE**.
- **y.center.orig**: numeric vector **n\_out** or matrix **n\_out X n\_dim**. Value of the center computed if **comp\_original\_center = TRUE**.
- **similarity.origin**: numeric vector **n\_obs** dissimilarity, similarity or distance of the original center respect the observations computed if **comp\_original\_center=TRUE**.
- **x.final**: matrix [n.func X grid.size]. Aligned abscissas.

- `n.clust.final`: scalar. Final number of clusters. Note that it may differ from initial number of clusters (i.e., from `n.clust`) if some clusters are found to be empty.
- `x.centers.final`: matrix [n.clust.final X grid.size]. abscissas of the final function centers.  
`y.centers.final`: matrix [n.clust.final X n.out] or array [n.clust.final X n.out x n.dim] , contain the evaluations of the final functions centers.
- `labels`: vector of length `n.obs`. Cluster assignments.
- `similarity.final`: vector [n.obs]: similarities, dissimilarities or distance between each function and the center of the cluster the function is assigned to.

Further details about `kmap` can be found in the package manual. Note: if the acquisition contains too few breaths, numerical problems in `kmap` may arise. In this case, one can find the median breath with respect to the non-warped curves by imposing `ind=which(res$similarity.origin==0)`.

### A.2.1 File `smooth_breath.R`

In the function `smooth_breath` we create a b-spline basis over an equidistant grid of knots. Then, we create an object of type *functional parameter*, to enclose penalization over the third derivative, and we evaluate the Generalized Cross Validation index for values of lambda on a log scale. Once found the best lambda, we evaluate the smoothed signal and its derivatives.

Function `smooth_breath` takes as input:

- `time`: the time vector, that is the abscissa of the signal;
- `amplitude`: the volume vector;
- `order`: the order of the b-spline basis to use. It defaults to 5;
- `grid_coef`: an integer. It defines the refinement of the knots grid. Defaults to 10 (1 knot each 10 datapoints).
- `lambda`: a vector with penalisation parameters, the best one is selected via Generalized Cross Validation (GCV). It is advisable to look for the optimal lambda on a log scale;
- `plot`: boolean. If 1 two plots are produced, one with the computed GCV for each lambda and one with the smoothed curve overlapped to the original signal.

Output quantities (contained in a list) are:

- `smoothed_curve`: the evaluation of the smooth signal;
- `der1`: the evaluation of the signal first derivative;
- `der2`: the evaluation of the signal second derivative;
- `lambda`: the optimal penalisation parameter;
- `GCV`: optimal GCV statistic;
- `df`: degrees of freedom in the smoothed curve.

```

smooth_breath= function(time, amplitude, order=5, grid_coef=10,
                        lambda=c(1e-4,1e-5,1e-6,1e-7,1e-8,1e-9,1e-10),
                        plot=0)
{
  # create breakpoints vector (uniform grid)
  breakst=c()
  for(i in seq(1,length(time),grid_coef))
    breakst=c(breakst,time[i])

  # create the bspline basis
  base=create.bspline.basis(c(time[0],time[length(time)]),
    breaks=breakst,norder=order)

  abscissa=time;
  Xobs0=amplitude;

  # evaluate the best penalization lambda using GCV
  gcv=numeric(length(lambda))
  for (i in 1:length(lambda))
  {
    functionalPar = fdPar(fdobj=base, Lfdobj=3, lambda=lambda[i])
    gcv[i] = smooth.basis(abscissa, Xobs0, functionalPar)$gcv
  }

  lam=lambda[which.min(gcv)]

  # functional parameter, having arguments: basis, order of the
  # derivative to be penalized, smoothing parameter
  functionalPar = fdPar(fdobj=base, Lfdobj=3, lambda=lam)

  Xss=smooth.basis(abscissa, Xobs0, functionalPar)

  # evaluation of the smooth function
  Xss0 = eval.fd(abscissa, Xss$fd, Lfd=0)

  # evaluation of the first derivative
  Xss1 = eval.fd(abscissa, Xss$fd, Lfd=1)

  # evaluation of the second derivative
  Xss2 = eval.fd(abscissa, Xss$fd, Lfd=2)

  df = Xss$df # the degrees of freedom in the smoothing curve
  GCV = Xss$gcv # the value of the GCV statistic

  if(plot){

    x11()
    plot(log10(lambda),gcv)
  }
}

```



```

x11()
plot( abscissa , Xobs0 , xlab="t" , ylab="observed data" )
points( abscissa , Xss0 , type="l" , col="blue" , lwd=2)
}

result=list( smoothed_curve=Xss0 , der1= Xss1 , der2= Xss2 ,
lambda=lam , GCV=GCV , df=df );
result
}

```

## A.2.2 File localmin.R

Input to `find_local_min` function are:

- **time**: time vector;
- **voltot**: volume vector;
- **peak\_span**: parameter *span* to be passed to the `peaks` function. A peak is defined as an element in a sequence which is greater than all other elements within a window of width *span* centered at that element;
- **grid\_coef**: parameter to be passed to `smooth_breath`;
- **step**: length of the subintervals on which the mean derivative is computed;
- **slope\_coef**: we state that the mean derivative starts to grow if the mean derivative in a subinterval is `slope_coef` times the mean derivative in the previous subinterval. Default value is 3;
- **plot**: boolean. If 1 two plots are produced, one showing the detected local maxima and the other showing the local minima;
- **spiky\_min**: logical. If TRUE then `peaks` is directly used to compute local minima. The default is FALSE.

The choice of parameter `peak_span` can be done looking at the (approximate) breathing frequency of the subject: for example, supposing a sampling frequency of 60Hz, if looking at the volume plot we see that the breaths duration is about 3s, then a good value for `peak_span` could be  $3 \times 60 = 180 \rightarrow 181$ . This means that, for each data point, the algorithm will check if it is higher than the 90 points on the left and the 90 on the right, where 90 data points correspond to 1.5s. The +1 is for the central point of the window, that is, the point being evaluated. For this reason, `peak_span` should always be an odd integer.

The output of the function is a list containing the following:

- **minima**: vector time points corresponding to minima locations;
- **minidx**: vector of positions of minima in the data vector;
- **maxidx**: vector of positions of maxima in the data vector;

- `deltas`: vector of breath lengths.

```

find_local_min=function(time, voltot, peak_span=201, grid_coef=10, step=10,
                        slope_coef=3, plot=0, spiky_min=F)
{
    localmax=which(peaks(voltot, span=peak_span)==TRUE) # find local maxima

    if(spiky_min=F){ # normal/low frequency breaths, with tails

        br=table_breaths(time, voltot, localmax)

        min=NULL
        minidx=NULL

        # smoothing to find derivatives between two maxima
        for(j in 1:dim(br$breaths)[2]){
            vec=which(! is.na(br$breaths[,j]))

            #smooth pieces one by one
            sm=smooth_breath(br$times[vec,j], br$breaths[vec,j],
                            grid_coef=grid_coef, plot = 0)

            timem=br$times[vec,j]
            volm=br$breaths[vec,j]
            der1m=sm$der1

            if(length(timem)>step){

                # divide the interval between two maxima in subintervals
                # and compute the mean subintervals derivatives

                amp=length(timem)%%step

                breaks=seq(1, (length(timem)-amp), by=step)

                minloc=NULL

                if(length(breaks)>2){

                    ms=c(mean(der1m[breaks[1]:breaks[2]]))

                    for(j in 2:(length(breaks)-1)){
                        med=mean(der1m[breaks[j]:breaks[j+1]])
                        ms=c(ms, med)
                        # look in which subinterval the mean first derivative
                        starts to grow
                        if(med>=0 & med>slope_coef*ms[j-1]){
                            #minloc=timem[breaks[j]]
                    }
                }
            }
        }
    }
}

```

```

id=which.min(volm[breaks[j-1]:breaks[j+1]])
minloc=timem[breaks[j-1]+id-1]
}
}
} else if(length(breaks)==2){
id=which.min(volm[breaks[1]:breaks[2]])
minloc=timem[breaks[1]+id-1]
}

min=c(min, minloc)
minidx=c(minidx, which(time==minloc))
}

}

} else { # for high-frequency breaths, without tails
# use 'peaks' directly to find the min
minidx=which(peaks(-voltot, span=peak_span)==TRUE)
min=voltot[minidx]
}

# breath lengths computation
deltaT=c();

last=time[minidx[1]];

for (i in 2:(length(min))){
t=time[minidx[i]];
deltaT=c(deltaT, abs(t-last));
last=t;
}

if(plot){

x11()
plot(time, voltot, type='l', main='Local maxima')
points(time[localmax], voltot[localmax], col='red', pch=20)
x11()
plot(time, voltot, type='l', main='Local minima')
points(time[minidx], voltot[minidx], pch=20, col='red')
}

result=list(minima=min, minidx=minidx, maxidx=localmax, deltas=deltaT)

result
}

```

### A.2.3 File outlier\_detection.R

Implementation of functional multivariate boxplot and outliergram is provided by the R package `roahd` (Ieva, Paganoni, Romo, Tarabelloni [11]), available on CRAN. We wrote a R function called `outlier_detection`, which combines together the different outlier detection techniques and performs algorithm iterations.

The input of the function is:

- `times`: matrix with breaths times. Times have to start from 0 for each breath;
- `smoothed_tot`: matrix of the smooth total volume breaths;
- `smoothed_rcp`: matrix of the smooth RCp breaths;
- `smoothed_rca`: matrix of the smooth RCa breaths;
- `smoothed_ab`: matrix of the smooth AB breaths;
- `plot_option`: boolean. If 1, plots of the algorithm iterations are produced;
- `weights`: either "uniform" or a vector of weights to be passed to `fbplot` and `multivariate.outliergram`.
- `range`: scalar. The inflating factor for time boxplots whiskers. Defaults to 1.5.
- `no_iter`: boolean. If 1, outlier detection at each phase is not repeated, but just the first iteration is performed. defaults to 0.

Output is a list containing:

- `filtered.times`: matrix of breaths time vectors starting from 0, after outlier removal;
- `filtered.Vtot`: matrix of total volume vectors, after outlier removal;
- `filtered.Vrcp`: matrix of RCp volume vectors, after outlier removal;
- `filtered.Vrca`: matrix of RCa volume vectors, after outlier removal;
- `filtered.Vab`: matrix of AB volume vectors, after outlier removal;
- `outliers.idx`: vector of outlier breaths indexes;
- `time.outliers.idx`: vector of indexes of time outliers;
- `magnitude.outliers.idx`: vector of indexes of magnitude outliers;
- `shape.outliers.idx`: vector of indexes of shape outliers.

```

outlier_detection=function(times ,smoothed_tot ,smoothed_rcp ,smoothed_rca ,
                           smoothed_ab,plot_option=1, weights='uniform',
                           range=1.5, no_iter=0)
{

  filtered.times=times
  filtered.Vtot=smoothed_tot
  filtered.Vrcp=smoothed_rcp
  filtered.Vrca=smoothed_rca
  filtered.Vab=smoothed_ab

  # Check if the number of breaths is high enough to safely apply
  # the entire procedure
  breaths.are.too.few=max(ifelse(dim(filtered.Vtot)[2]<30, 1, 0), no_iter)

  if(breaths.are.too.few)
  warning('Number of breaths is too low to apply outlier detection
  iteratively. Outlier checks have been performed only once')

  # Auxiliary vector to store the original indices of the outliers
  aux.idx=1:dim(smoothed_tot)[2]

  ##### outlier detection: TIME
  time.outliers.idx=NULL # vector of indices for time outliers
  found=1

  deltas=c()
  for(i in 1:dim(filtered.times)[2])
    deltas=c(deltas ,
             filtered.times[which(!is.na(filtered.times[,i]))],i)

  # iterate detection until no more outliers are found
  while(found){
  x11()
  bp <- boxplot(deltas , range=range, plot = plot_option)
  timesout <- bp$out
  timesout <- unique(timesout)

  out <- NULL
  for(k in 1:length(deltas)){
  for(h in timesout)
  if(deltas[k]==h){
  out <- c(out,k)
  break
  }
  }

  # remove outliers in time (if any)
  if(length(out)>0){

```

```

deltas=deltas[-out]

filtered.times=filtered.times[,-out]
filtered.Vtot=filtered.Vtot[,-out]
filtered.Vrcp=filtered.Vrcp[,-out]
filtered.Vrca=filtered.Vrca[,-out]
filtered.Vab=filtered.Vab[,-out]

time.outliers.idx=c(time.outliers.idx, aux.idx[out])
aux.idx=aux.idx[-out]

# Stop to 1 iteration if there are not many breaths
if(breaths.are.too.few)
found=0

}
else
found=0
}

# Rescale breaths on [1:100] to apply functional outlier detection
# on magnitude and shape
scaled.Vtot=rescale_all(filtered.Vtot,filtered.times)
scaled.Vrcp=rescale_all(filtered.Vrcp,filtered.times)
scaled.Vrca=rescale_all(filtered.Vrca,filtered.times)
scaled.Vab=rescale_all(filtered.Vab,filtered.times)

##### outlier detection: MAGNITUDE
magnitude.outliers.idx=NULL # vector of indices for magnitude outliers
found=1

# iterate detection until no outliers are found

while(found){
out1=fbplot(mfData(grid=1:100,list(t(scaled.Vtot),t(scaled.Vrcp),
t(scaled.Vrca),t(scaled.Vab))),
Depths=list(def='MBD',weights=weights),
main=list('Magnitude outliers','Magnitude outliers',
'Magnitude outliers','Magnitude outliers'),
display=plot_option)
idx1=out1$ID_outliers;

# remove outliers in magnitude (if any)
if(length(as.vector(idx1))>0){
scaled.Vtot=scaled.Vtot[,-idx1]
scaled.Vrcp=scaled.Vrcp[,-idx1]
scaled.Vrca=scaled.Vrca[,-idx1]
scaled.Vab=scaled.Vab[,-idx1]
}
}

```

```

filtered.times=filtered.times[,-idx1]
filtered.Vtot=filtered.Vtot[,-idx1]
filtered.Vrcp=filtered.Vrcp[,-idx1]
filtered.Vrca=filtered.Vrca[,-idx1]
filtered.Vab=filtered.Vab[,-idx1]

magnitude.outliers.idx=c(magnitude.outliers.idx, aux.idx[idx1])
aux.idx=aux.idx[-idx1]

# Stop to 1 iteration if there are not many breaths
if(breaths.are.too.few)
found=0
}
else
found=0
}

##### outlier detection: SHAPE
shape.outliers.idx=NULL # vector of indices for shape outliers
found=1

while(found){ # iterate detection until no outliers are found
x11()
out2=multivariate_outliergram(mfData(grid=1:100, list(t(scaled.Vtot),
t(scaled.Vrcp), t(scaled.Vrca), t(scaled.Vab))),
weights=weights,
display = plot_option)
idx2=out2$ID_outliers;
if(length(as.vector(idx2))>0){

scaled.Vtot=scaled.Vtot[,-idx2]
scaled.Vrcp=scaled.Vrcp[,-idx2]
scaled.Vrca=scaled.Vrca[,-idx2]
scaled.Vab=scaled.Vab[,-idx2]

filtered.times=filtered.times[,-idx2]
filtered.Vtot=filtered.Vtot[,-idx2]
filtered.Vrcp=filtered.Vrcp[,-idx2]
filtered.Vrca=filtered.Vrca[,-idx2]
filtered.Vab=filtered.Vab[,-idx2]

shape.outliers.idx=c(shape.outliers.idx, aux.idx[idx2])
aux.idx=aux.idx[-idx2]

# Stop to 1 iteration if there are not many breaths
if(breaths.are.too.few)
found=0
}
else
found=0
}

```

```

}

# Cut breaths to the longest one
max.len=0
for(j in 1:dim(filtered.times)[2]){
len=length(which(!is.na(filtered.times[,j])))
if(len>max.len)
max.len=len
}

filtered.times=filtered.times[1:max.len,]
filtered.Vtot=filtered.Vtot[1:max.len,]
filtered.Vrcp=filtered.Vrcp[1:max.len,]
filtered.Vrca=filtered.Vrca[1:max.len,]
filtered.Vab=filtered.Vab[1:max.len,]

result=list(
filtered.times=filtered.times,
filtered.Vtot=filtered.Vtot,
filtered.Vrcp=filtered.Vrcp,
filtered.Vrca=filtered.Vrca,
filtered.Vab=filtered.Vab,
outliers.idx=c(time.outliers.idx, magnitude.outliers.idx,
shape.outliers.idx),
time.outliers.idx=time.outliers.idx,
magnitude.outliers.idx=magnitude.outliers.idx,
shape.outliers.idx=shape.outliers.idx)
}

```

The function `plot_outliers` can be used to plot the outlier breaths with respect to the volume tracks. Outliers are colored differently according to the phase in which they were removed (red for time outliers, blue for magnitude ones, green for shape). Inputs are the time vector, the volume vector, the matrix of separated breaths and outlier indexes as outputted by `outlier_breaths`.

```

plot_outliers=function(time, vol, volumes,time.outliers.idx,
magnitude.outliers.idx, shape.outliers.idx,
main='Outlier breaths',xlab='', ylab='')
{
col=c('red','blue','green')
x11()
plot(time, vol, type='l', xlab=xlab, ylab=ylabel, main=main, col='black')

for(j in time.outliers.idx){
lines(volumes$times[,j],volumes$breaths[,j], type='l', col=col[1])
}
for(j in magnitude.outliers.idx){
lines(volumes$times[,j],volumes$breaths[,j], type='l', col=col[2])
}
}

```



```

for(j in shape.outliers.idx){
lines(volumes$times[,j],volumes$breaths[,j], type='l', col=col[3])
}

text=c('Outliers - Time','Outliers - Magnitude','Outliers - Shape')
lty <- rep(1, length(text))
lwd <- rep(2, length(text))
legend("topright", legend = text, col = col, lty = lty, lwd=lwd,
cex = 0.8)
}

```

## A.2.4 Auxiliary functions

### File rescale.R

These auxiliary functions were written in order to resample breaths over the same number of points:

```

# rescales all breaths on a 0-100 grid

rescale_all=function(breaths ,times){

  first_br=breaths
  first_time=times
  N=dim(breaths)[2];
  resc_br=matrix(0,100,N)
  grid=seq(0,100,length.out = 100)
  for(i in 1:N){
  # Remove NA
  x=first_time[!is.na(first_time[,i]),i];
  y=first_br[!is.na(first_br[,i]),i];

  resc_br[,i] = rescale_time(x,y,n=100)
  }

  result=data.frame(resc_br)
  result
}

# takes a time series as input and returns the value of the ts in n
# equidistant points over t_min and t_max, using a cubic spline
# for interpolation

rescale_time=function(x,y,n=1000){

  f=x[1];
  t=x[length(x)];

  newx=seq(f,t,length.out=n);

```

```

    res_curve=spline(x,y,xout=newx);
    res_curve$y
}

```

#### File tableBreaths.R

Given the signal and the cutting points, `tableBreaths` creates two matrices (returned inside a list): one of breaths time vectors, and one with breaths amplitude vectors. Missing values (breaths have all different lengths) are filled with NA.

```

tableBreaths=function(time, amplitude, min_idx){

    breaths=amplitude[min_idx[1]:min_idx[2]];
    t=time[min_idx[1]:min_idx[2]];

    PLOT_BREATH = matrix(NA, nrow = length(time), ncol = length(min_idx)-1);
    PLOT_TIME = matrix(NA, nrow = length(time), ncol = length(min_idx)-1);
    #empty matrices for breaths plot, filled with NA
    PLOT_BREATH[1:length(breaths),1]=breaths;
    PLOT_TIME[1:length(breaths),1]=t;
    nrows=0;

    for ( i in seq(3, length(min_idx), by =1) ){

        #newbreath = amplitude[min_idx[i-1]:min_idx[i]];
        PLOT_BREATH[1:length(amplitude[min_idx[i-1]:min_idx[i]]), i-1] =
        amplitude[min_idx[i-1]:min_idx[i]];
        PLOT_TIME[1:length(time[min_idx[i-1]:min_idx[i]]), i-1] =
        time[min_idx[i-1]:min_idx[i]];
        if( length(amplitude[min_idx[i-1]:min_idx[i]]) > nrows ) {
            nrows = length(amplitude[min_idx[i-1]:min_idx[i]])
        }
    }

    PLOT_BREATH = PLOT_BREATH[1:nrows,];
    PLOT_TIME = PLOT_TIME[1:nrows,];

    result=list(breaths=PLOT_BREATH, times=PLOT_TIME);
    result
}

```

#### File plotBreathsByClusters.R

The plotting function `plotBreathsByClusters` displays a graph where the original data track is colored according to the groups. It takes as input the original data (`time` and `amplitude` vectors),

the filtered times and the filtered breaths, and the vector of labels resulting from clustering. A parameter `threshold` can be provided to exclude from the plot clusters whose numerosity is less than `threshold%` of the total number of breaths.

```
plot.breaths.byclusters=function(time, amplitude, plot.times, plot.breaths,
                                labels, threshold=0)
{
  n.breaths=dim(plot.times)[2]
  labels.unique=sort(unique(labels))
  n.clusters=length(labels.unique)

  col.ramp <- c("red", "blue", "green3", "orange", "grey", "yellow")
  col.ramp.after <- rainbow(n.clusters)

  col.ramp <- c(col.ramp, col.ramp.after)
  col.ramp <- col.ramp[1:n.clusters]
  cluster.dimension <- rep(0, n.clusters)

  col.bygroup <- rep(0, n.breaths)
  for(k in labels.unique){
    col.bygroup[which(labels == k)] <- col.ramp[k]
    cluster.dimension[k] <- sum(labels==k)
  }
  cluster.prop <- (cluster.dimension/n.breaths)*100

  ### Plot each breath in the colour of the corresponding cluster

  x11()
  plot(time, amplitude, type='l', xlab='Time', ylab='Volume',
        main='Breaths by clusters', col='black')

  for(j in 1:n.breaths){
    lines(plot.times[,j], plot.breaths[,j], type='l', col=col.bygroup[j])
  }

  text <- rep(0, n.clusters)
  for (i in 1:n.clusters) {
    tt <- c("Cluster ", i, " Number of breaths = ",
           cluster.dimension[i],
           " (", round(cluster.prop[i], digits = 2), "%)")
    tt <- paste(tt, collapse = "")
    text[i] <- tt
  }
  lty <- rep(1, length(text))
  lwd <- rep(2, length(text));
  legend("topleft", legend = text, col = col.ramp, lty = lty, lwd=lwd,
        cex = 0.8)
```

```

#### Plot selected clusters only

# vector of bool, indicate if the cluster is kept or not
selected=rep(1,n.clusters)

# Discarded clusters are in gray
col.bySelectedGroup <- col.bygroup
for(i in 1:n.clusters){
  if(cluster.prop[i] <= threshold){
    selected[i]=0
    col.bySelectedGroup[which(labels == i)] <- "grey"
  }
}

num.discarded=length(which(selected==0))

if(threshold!=0){
# if no threshold is set, show only the first plot

x11()
plot(time, amplitude, type='l', xlab='Time', ylab='Volume',
main=paste(c('Selected clusters - Threshold = ',threshold, "%"),
collapse=""), col='black')

for(j in 1:n.breaths){
  lines(plot.times[,j],plot.breaths[,j], type='l',
col=col.bySelectedGroup[j])
}

text <- NULL
color.leg <-NULL
for (i in 1:n.clusters) {
  if(selected[i]){
    tt <- c("Cluster ", i, " Number of breaths = ",
cluster.dimension[i],
" (",round(cluster.prop[i],digits = 2), "%)")
    tt <- paste(tt, collapse = "")
    text <- c(text, tt)
    color.leg <- c(color.leg, col.ramp[i])
  }
}
text <- c(text, paste("Discarded: ", ifelse(num.discarded>0,
paste("Cluster ", which(selected==0), "None"),
collapse = ""))
color.leg <- c(color.leg, "grey")
lty <- rep(1, length(text))
lwd <- rep(2,length(text));
legend( "topleft", legend = text, col = color.leg, lty = lty, lwd=lwd,

```

```

    cex = 0.8)
  }

  result=list( col.groups=col.bygroup,
              col.selected.groups=col.bySelectedGroup)
}

```

### A.3 Inter-subjects comparison

In this section we report a script example for inter-subjects comparison (seated vs. supine), which can easily be adapted to other applications. It is supposed that the medians for each subject have been saved in files `name_median.txt` and the median derivatives in `name_median_d1.txt`.

```

library(fda)
library(roahd)
library(fdakmapp)
library(splus2R)

setwd("/path/to/Rscripts")

source("smooth_breath.R")
source("localmin.R")
source("table_breaths.R")
source("rescale.R")
source("smooth.R")
source("outlier_detection.R")
source("plot_compartments.R")

setwd("/path/to/datafiles")

supini= # file names vector group 1
seduti= # file names vector group 2

n_supini=length(supini)
n_seduti=length(seduti)

names=c(supini, seduti)

max.len=0
l2=T

if(l2==1){
  for(i in names){
    cent=read.table(paste0(i, '_median.txt', sep=' '), header=T)

```

```

    if (length(cent[,1]) > max.len)
      max.len=length(cent[,1])
  }

  mat.times=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vtot=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vrcp=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vrca=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vab=matrix(NA, nrow = max.len, ncol = length(names))

  for (i in 1:length(names)){
    cent=read.table(paste0(names[i], '_median.txt', sep=''), header=T)
    mat.times[1:length(cent[,1]), i]=cent[,1]
    mat.vtot[1:length(cent[,1]), i]=(cent[,2] - cent[,2][1]) / max(cent[,2] - cent[,2][1],
na.rm = T)
    mat.vrcp[1:length(cent[,1]), i]=(cent[,3] - cent[,3][1]) / max(cent[,2] - cent[,2][1],
na.rm = T)
    mat.vrca[1:length(cent[,1]), i]=(cent[,4] - cent[,4][1]) / max(cent[,2] - cent[,2][1],
na.rm = T)
    mat.vab[1:length(cent[,1]), i]=(cent[,5] - cent[,5][1]) / max(cent[,2] - cent[,2][1],
na.rm = T)
  }
} else {

  for (i in names){
    cent=read.table(paste0(i, '_median_d1.txt', sep=''), header=T)
    if (length(cent[,1]) > max.len)
      max.len=length(cent[,1])
  }

  mat.times=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vtot=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vrcp=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vrca=matrix(NA, nrow = max.len, ncol = length(names))
  mat.vab=matrix(NA, nrow = max.len, ncol = length(names))

  for (i in 1:length(names)){
    cent=read.table(paste0(names[i], '_median_d1.txt', sep=''), header=T)
    mat.times[1:length(cent[,1]), i]=cent[,1]
    mat.vtot[1:length(cent[,1]), i]=cent[,2]
    mat.vrcp[1:length(cent[,1]), i]=cent[,3]
    mat.vrca[1:length(cent[,1]), i]=cent[,4]
    mat.vab[1:length(cent[,1]), i]=cent[,5]
  }
}

max.len=0 # nei centroidi sono salvati anche gli NA alla fine

```

```

for(j in 1:dim(mat.times)[2]){
  len=length(which(!is.na(mat.times[,j])))
  if(len>max.len)
    max.len=len
}

mat.times=mat.times[1:max.len,]
mat.vtot=mat.vtot[1:max.len,]
mat.vrcp=mat.vrcp[1:max.len,]
mat.vrca=mat.vrca[1:max.len,]
mat.vab=mat.vab[1:max.len,]

col=c(rep('red',n_supini),rep('blue',n_seduti))
text=c('Supini','Seduti')
lty <- rep(1, length(text))
lwd <- rep(2, length(text))
col2=c('red','blue')

x11();matplot(mat.times, mat.vtot, type = 'l', main='CW volume',col=col,
xlab = 'Time (s)', ylab = 'V (l)')
legend("topright", legend = text, col = col2, lty = lty, lwd=lwd, cex = 0.8)

x11();matplot(mat.times, mat.vrcp, type = 'l', main='RCp volume',col=col,
xlab = 'Time (s)', ylab = 'V (l)')
legend("topright", legend = text, col = col2, lty = lty, lwd=lwd, cex = 0.8)

x11();matplot(mat.times, mat.vrca, type = 'l', main='RCa volume',col=col,
xlab = 'Time (s)', ylab = 'V (l)')
legend("topright", legend = text, col = col2, lty = lty, lwd=lwd, cex = 0.8)

x11();matplot(mat.times, mat.vab, type = 'l', main='AB volume',col=col,
xlab = 'Time (s)', ylab = 'V (l)')
legend("topright", legend = text, col = col2, lty = lty, lwd=lwd, cex = 0.8)

filtered=outlier_detection(mat.times, mat.vtot,mat.vrcp,mat.vrca,mat.vab,
plot_option = 1)

filtered.times=filtered$filtered.times
filtered.Vtot=filtered$filtered.Vtot
filtered.Vrcp=filtered$filtered.Vrcp
filtered.Vrca=filtered$filtered.Vrca
filtered.Vab=filtered$filtered.Vab

outliers=filtered$outliers.idx

# If absolute value is relevant, use L2. Use Pearson otherwise
groupBy.abs=TRUE

# c(t(filtered.Vtot)), c(t(filtered.Vrca)),

```

```

datavec = c(c(t(filtered.Vrcp)), c(t(filtered.Vab)))
arr=array(datavec, c(dim(filtered.Vrcp)[2], dim(filtered.Vrcp)[1], 2))

timet=t(filtered.times)

distance=ifelse(groupBy.abc==1,'l2', 'pearson')
warp= 'affine'

# choose number of clusters —> elbow
K=4
checksim=numeric(K)

for(j in 1:K){

  check=kmap(timet, arr, n_clust=j, warping_method=warp,
            center_method='medoid', similarity_method=distance,
            comp_original_center = T)

  checksim[j]=mean(check$similarity.final)
}

x11()
plot(checksim, type='l', xlab='Cluster labels', ylab='Mean similarities',
main=paste0('Mean Similarity - ',warp,sep=''))
points(checksim, pch=1, col='blue',lwd=2)

# Selected number of clusters
nclust=2

res=kmap(timet, arr, n_clust=nclust, warping_method=warp,
        center_method='medoid', similarity_method=distance,
        comp_original_center = T)

kmap_show_results(res)

centroid_idx=which(res$similarity.final==0)
labels[centroids_idx]
titles=c('Healty supine', 'Healthy seated')

plot_compartments(time=timet, Vtot=filtered.Vtot, Vrcp=filtered.Vrcp,
                 Vrca=filtered.Vrca, Vab=filtered.Vab,
                 n_clust=length(unique(labels)),
                 centroid_idx=centroid_idx,
                 labels, titles)

```

### A.3.1 File plot\_compartments.R

This function can be used to to make two plots: a plot of the centroid for each cluster (chest wall volume and its compartments), and a plot with the comparison of the centroids compartment by



compartment. The function takes as arguments:

- **time**: matrix of breaths time vectors by rows;
- **Vtot**: matrix of chest wall curves by columns;
- **Vrcp**: matrix of RCp curves by columns;
- **Vrca**: matrix of RCa curves by columns;
- **Vab**: matrix of AB curves by columns;
- **n\_clust**: number of clusters;
- **centroid\_idx**: indexes of clusters centroids;
- **labels**: breaths cluster labels;
- **titles**: plots titles, in the order of centroids indexes.

```

plot_compartments=function(time, Vtot, Vrcp, Vrca, Vab, n_clust, centroid_idx,
                             labels, titles)
{
##### Color scale for clusters
col.ramp <- c("red", "blue", "green3", "orange", "grey", "yellow")
col.ramp.after <- rainbow(n_clust)

col.ramp <- c(col.ramp, col.ramp.after)
col.ramp <- col.ramp[1:n_clust]

##### Plot median vs compartments

# Axis limits
xlim_sup=rep(0,n_clust)
ylim_inf=rep(0,n_clust)
ylim_sup=rep(0,n_clust)
xlim_max=0
ylim_min=0
ylim_max=0
for(i in 1:n_clust){
  xlim_sup[i]=max(time[centroid_idx[i],]-time[centroid_idx[i],1],na.rm=TRUE)
  ylim_inf[i]=min(na.omit(Vtot[,centroid_idx[i]]-Vtot[1,centroid_idx[i]]),
                  na.omit(Vrcp[,centroid_idx[i]]-Vrcp[1,centroid_idx[i]]),
                  na.omit(Vrca[,centroid_idx[i]]-Vrca[1,centroid_idx[i]]),
                  na.omit(Vab[,centroid_idx[i]]-Vab[1,centroid_idx[i]]))
  ylim_sup[i]=max(na.omit(Vtot[,centroid_idx[i]]-Vtot[1,centroid_idx[i]]),
                  na.omit(Vrcp[,centroid_idx[i]]-Vrcp[1,centroid_idx[i]]),
                  na.omit(Vrca[,centroid_idx[i]]-Vrca[1,centroid_idx[i]]),
                  na.omit(Vab[,centroid_idx[i]]-Vab[1,centroid_idx[i]]))
}

```

```

    if(xlim_sup[i]>xlim_max)
      xlim_max=xlim_sup[i]
    if(ylim_sup[i]>ylim_max)
      ylim_max=ylim_sup[i]
    if(ylim_inf[i]<ylim_min)
      ylim_min=ylim_inf[i]
  }

# plot
x11()
par(mfrow=c(1,n_clust))
for(i in 1:n_clust){

  plot(time[centroid_idx[i,]]-time[centroid_idx[i],1],
       Vtot[,centroid_idx[i]]-Vtot[1,centroid_idx[i]],
       col="black", lwd=4, type='l', main=titles[i],
       cex.main=2, xlab='', ylab='', xaxt='n', yaxt='n',
       xlim=c(0,xlim_max), ylim=c(ylim_min,ylim_max))
  axis(1,cex.axis=1.2)
  axis(2,cex.axis=1.2)
  mtext("time (s)", side=1, line=2.5, cex=1.8)
  mtext("volume (L)", side=2, line=2.5, cex=1.8)
  lines(time[centroid_idx[i,]]-time[centroid_idx[i],1],
        Vrcp[,centroid_idx[i]]-Vrcp[1,centroid_idx[i]],
        col="blue", lwd=4)
  lines(time[centroid_idx[i,]]-time[centroid_idx[i],1],
        Vrca[,centroid_idx[i]]-Vrca[1,centroid_idx[i]],
        col="green", lwd=4)
  lines(time[centroid_idx[i,]]-time[centroid_idx[i],1],
        Vab[,centroid_idx[i]]-Vab[1,centroid_idx[i]],
        col="red", lwd=4)
}

leg=c("CW", "RCp", "RCa", "AB");
line.colors=c(1,1);
line.width=c(3,3);
xpos=xlim_max-xlim_max/2.35
ypos=ylim_max+ylim_max/20
legend(xpos,ypos,leg, col=c('black','blue','green','red'),
       lty=line.colors, lwd=line.width, cex=1.2, bty='n')

#### Compare compartments in each group

xlim=c(0,max(time[centroid_idx,]-time[centroid_idx,1], na.rm=TRUE))

ylim_tot=c(min(Vtot[,centroid_idx]-Vtot[1,centroid_idx], na.rm=TRUE),
           max(Vtot[,centroid_idx]-Vtot[1,centroid_idx], na.rm=TRUE))
ylim_rcp=c(min(Vrcp[,centroid_idx]-Vrcp[1,centroid_idx], na.rm=TRUE),
           max(Vrcp[,centroid_idx]-Vrcp[1,centroid_idx], na.rm=TRUE))
ylim_rca=c(min(Vrca[,centroid_idx]-Vrca[1,centroid_idx], na.rm=TRUE),
           max(Vrca[,centroid_idx]-Vrca[1,centroid_idx], na.rm=TRUE))

```

```

      max(Vrca[, centroid_idx] - Vrca[1, centroid_idx], na.rm=TRUE))
ylim_ab=c(min(Vab[, centroid_idx] - Vab[1, centroid_idx], na.rm=TRUE),
          max(Vab[, centroid_idx] - Vab[1, centroid_idx], na.rm=TRUE))

```

```

x11()
par(mfrow=c(1,4))
plot(time[centroid_idx[1],] - time[centroid_idx[1],1],
     Vtot[, centroid_idx[1]] - Vtot[1, centroid_idx[1]],
     col=col.ramp[labels[centroid_idx[1]]], lwd=4, type='l',
     main='Total volume', xlab = 'time (s)', ylab='volume (L)',
     xlim=xlim, ylim=ylim_tot - c(0.025,0))
for(i in 2:n_clust){
  lines(time[centroid_idx[i],] - time[centroid_idx[i],1],
        Vtot[, centroid_idx[i]] - Vtot[1, centroid_idx[i]],
        col=col.ramp[labels[centroid_idx[i]]], lwd=4, type='l')
}
plot(time[centroid_idx[1],] - time[centroid_idx[1],1],
     Vrcp[, centroid_idx[1]] - Vrcp[1, centroid_idx[1]],
     col=col.ramp[labels[centroid_idx[1]]], lwd=4, type='l',
     main='RCP volume', xlab = 'time (s)', ylab='volume (L)',
     xlim=xlim, ylim=ylim_tot - c(0.025,0))
for(i in 2:n_clust){
  lines(time[centroid_idx[i],] - time[centroid_idx[i],1],
        Vrcp[, centroid_idx[i]] - Vrcp[1, centroid_idx[i]],
        col=col.ramp[labels[centroid_idx[i]]], lwd=4, type='l')
}
plot(time[centroid_idx[1],] - time[centroid_idx[1],1],
     Vrca[, centroid_idx[1]] - Vrca[1, centroid_idx[1]],
     col=col.ramp[labels[centroid_idx[1]]], lwd=4, type='l',
     main='RCA volume', xlab = 'time (s)', ylab='volume (L)',
     xlim=xlim, ylim=ylim_tot - c(0.025,0))
for(i in 2:n_clust){
  lines(time[centroid_idx[i],] - time[centroid_idx[i],1],
        Vrca[, centroid_idx[i]] - Vrca[1, centroid_idx[i]],
        col=col.ramp[labels[centroid_idx[i]]], lwd=4, type='l')
}
plot(time[centroid_idx[1],] - time[centroid_idx[1],1],
     Vab[, centroid_idx[1]] - Vab[1, centroid_idx[1]],
     col=col.ramp[labels[centroid_idx[1]]], lwd=4, type='l',
     main='AB volume', xlab = 'time (s)', ylab='volume (L)',
     xlim=xlim, ylim=ylim_tot - c(0.025,0))
for(i in 2:n_clust){
  lines(time[centroid_idx[i],] - time[centroid_idx[i],1],
        Vab[, centroid_idx[i]] - Vab[1, centroid_idx[i]],
        col=col.ramp[labels[centroid_idx[i]]], lwd=4, type='l')
}
text=NULL;
for(i in 1:n_clust){
  text=c(text, paste("Cluster ", i));
}

```

```
}  
line.colors=rep(1,length(text));  
line.width=rep(3,length(text))  
legend("topright",text, col=col.ramp[1:length(text)],  
lty=line.colors, lwd=line.width)  
}
```