Data Analysis in Respiratory Physiology. Model Detection, Parameter Extraction and Prediction Methods for Lung Protective Ventilation.

Steven Ganzert

Dissertation
zur Erlangung des Grades
„Doktor der Naturwissenschaften“

im Fachbereich Physik, Mathematik und Informatik
der Johannes Gutenberg-Universität in Mainz

vorgelegt von
Steven Ganzert
geboren in Freiburg im Breisgau

Mainz, Januar 2016

Name des Erstgutachters:	Name in der Online-Version gelöscht
Name des Zweitgutachters:	Name in der Online-Version gelöscht
Prüfungstermin:	18.01.2016

Contents

Zusammenfassung **ix**

Abstract **xi**

Danksagung **xiii**

1 Introduction **1**
1.1 Motivation 2
1.2 Medical Background 5
1.3 Medical Data 16
1.4 Aim of Work 21

2 Analysis of Stress Relaxation in the Human Lung **25**
2.1 Materials and Methods 26
2.2 Results 30
2.3 Discussion 31
2.4 Conclusions 37

3 Modeling of Respiratory Mechanics **39**
3.1 Equation Discovery in a Medical Problem Domain 40
3.2 Methodological Background: the LAGRAMGE System 42
3.3 Methodological Background: the GSAT Algorithm 68
3.4 Materials and Methods 70
3.5 Results 79
3.6 Discussion 85
3.7 Related Work 88
3.8 Conclusions 89

4 Prediction of Mechanical Lung Parameters **91**
4.1 Medical Background and Clinical Purpose 92
4.2 Materials and Methods 92
4.3 Results 105
4.4 Discussion 106
4.5 Conclusions 111

5 Summary **113**

A Derivation of Laplace Transform **117**

B RMSE for Two-Point and Multiple-Point Analysis **119**

C Ordered Refinement of Beam Elements **121**

D Calculation of Tracheal Pressure **125**

List of Symbols **139**

Acronyms **143**

Glossary and Index **145**

Curriculum Vitae **151**

List of Figures

1.1 Pressure-volume curves 7
1.2 Stress relaxation processes 11
1.3 Spring-and-dashpot model 13
1.4 Scheme of super syringe maneuver 14
1.5 Scheme of PEEP wave maneuver 16
1.6 Sample of super syringe maneuver (control group) 22
1.7 Sample of PEEP wave maneuver (control group) 23

2.1 Super syringe maneuver for ARDS and control group 26
2.2 Data preprocessing for two-point/multiple-point analysis 27
2.3 Results of parameter estimation 32
2.4 Frequency analysis 33

3.1 Universal grammar 44
3.2 Parse trees of derivation example 53
3.3 PEEP wave maneuver 76
3.4 Sample time series data 77
3.5 Input sequences of production rules for universal grammar 78
3.6 Experimental results relating to equation of motion (EOM) identifications: benchmark test(i) 80

3.7 Experimental results relating to the error variance: benchmark test (ii) . . 82
3.8 Root mean squared error during the course of iterations 84

4.1 Feature extraction from super syringe maneuver data 100
4.2 Sample results for one patient in experimental setting 2 109

B.1 RMSE for two-point and multiple-point analysis 120

List of Tables

1.1 Related work with respect to the analysis of viscoelastic effects 12
1.2 Patient data: ARDS group 18
1.3 Patient data: control group 20

3.1 Heights of production rules in universal grammar G_{U} 46
3.2 Refinements of parse trees $\mathcal{T}$ from ordered sets of production rules 52
3.3 Example for single parse tree refinement 59
3.4 Refinements of parse trees $\mathcal{T}$ biased by sequence of derivation steps 60
3.5 Modeling respiratory mechanics: results benchmark test (i)/(iii) 81
3.6 Modeling respiratory mechanics: results benchmark test (ii)/(iii) 83
3.7 Modeling respiratory mechanics: general performance 85

4.1 Sample data of one patient for model learning 101
4.2 Dataset statistics 102
4.3 Results for experimental setting 1 107
4.4 Results for experimental setting 2 108

D.1 Rohrer-coefficients for calculation of tracheal pressure 126

List of Algorithms

3.1 General algorithm for deriving parse trees 48
3.2 Procedure *RefinementOperator* 48

3.3 Top-level LAGRAMGE algorithm 49
3.4 Implementation of general algorithm for deriving parse trees 58
3.5 Implementation of procedure *RefinementOperator* 62
3.6 Procedure *NextDerivationStep* 62
3.7 Procedure *NextProductionRule* 62
3.8 Implementation of top-level LAGRAMGE algorithm 63
3.9 GSAT algorithm 69
3.10 Procedure *RandomRefinementOperator* 71
3.11 Procedure *RandomNextDerivationStep* 71
3.12 Procedure *ImproveRefinement* 72
3.13 Procedure *RandomLAGRAMGE* 73

List of Equations

1.1 Compliance: C 5
1.2 Pressure difference between external atmosphere and airways: $\Delta P(t)$ . . . 9
1.3 Equation of motion: $\Delta P(t)$ 9
1.4 Viscoelastic time constant: τ_{ve} 12
1.5 Slow pressure decrease: $P(t)$ 12
2.1 Adjustment of respiratory pressure of i-th volume step: $P^i(t)$ 28
2.2 Laplace transform of electrical circuit: $\frac{P(s)}{\dot{V}(s)}$ 29
2.3 Compliance C at volume step i: C^i 29
2.4 Resistance R at volume step i: R^i 29
2.5 Effective additional resistance at volume step i: ΔR^i_{add} 29
2.6 Viscoelastic resistance at volume step i: R^i_{ve} 30
2.7 Viscoelastic compliance at volume step i: C^i_{ve} 30
3.1 Root mean squared error as heuristic function for estimation of model quality: $F_{\mathrm{h}}(\mathcal{T})$ 49
3.2 Identified model equation: P 77
4.1 Linear combination of basis functions: $f(\mathrm{x};\mathrm{w})$ 93
4.2 Matrix of values of basis functions: $\mathbf{\Phi}_{NH}$ 93
4.3 Calculation of function values: $f(\mathrm{x}_n$ 93
4.4 Prior distribution of parameters w of dimension H: $P(\mathrm{w})$ 93
4.5 Mean m of f_N: $m(f_N)$ 94
4.6 Covariance matrix C of f_N: $C(f_N)$ 94
4.7 Prior distribution of f_N: $P(f_N|\mathrm{X}_N)$ 94
4.8 Prior distribution P of target values y_N: $P(\mathrm{y}_N|\mathrm{X}_N)$ 94
4.9 Specification of a Gaussian process (i): $f(\mathrm{x})$ 95

4.10 Specification of a Gaussian process (ii): $m(\mathrm{x})$ 95
4.11 Specification of a Gaussian process (iii): $k(\mathrm{x}$ 95
4.12 Example of covariance function: $k(\mathrm{x}, \mathrm{x}')$ 95
4.13 Example of kernel: $k(\mathrm{x}, \mathrm{x}'$ 95
4.14 Matrix for calculation of covariance function (i): K_N 96
4.15 Matrix for calculation of covariance function (ii): K_* 96
4.16 Matrix for calculation of covariance function (iii): K_{**} 96
4.17 Matrix for calculation of covariance function (iv): $[\mathrm{y}_N\ y_{N+1}]^T$ 96
4.18 Conditional distribution of the output y_{N+1} (i): $P(y_{N+1}|\mathrm{x}_{N+1}, \mathcal{D})$ 96
4.19 Conditional distribution of the output y_{N+1} (ii): $m(\mathrm{x}_{N+1})$ 96
4.20 Conditional distribution of the output y_{N+1} (iii): $C(\mathrm{x}_{N+1})$ 96
4.21 Pearson VII Kernel: $K(x_i, x_j; (\sigma, \omega))$ 103
4.22 Pearson VII Kernel with $\sigma = \omega = 1$: $K(x_i, x_j; (1, 1))$ 103
A.1 Derivation of Laplace transform (i): P 118
A.2 Derivation of Laplace transform (ii): $R_{ve}(\dot{V} - \dot{V}_{ve,C})$ 118
A.3 Derivation of Laplace transform (iii): $\dot{V}_{ve,C}$ 118
A.4 Derivation of Laplace transform (iv): $R_{ve}\frac{\partial \dot{V}}{\partial t} - R_{ve}\frac{\dot{V}_{ve,C}}{\partial t}$ 118

Zusammenfassung

Eine der wichtigsten lebenserhaltenden Maßnahmen in der Intensivmedizin ist die mechanische Beatmung der Lunge. Bei inadäquater Einstellung des Beatmungsgerätes können jedoch Lungenschäden induziert oder verstärkt werden. Daher ist es ist das Ziel lungenprotektiver Beatmungsstrategien, den mechanischen Stress, dem die Lunge während der Beatmung ausgesetzt ist, zu reduzieren. Um solche Beatmungsstrategien entwickeln zu können, ist das Wissen über die physiologischen und mechanischen Zusammenhänge und Wechselwirkungen in der Lunge eine wesentliche Voraussetzung. In klassischen Ansätzen aus der Biomedizintechnik wird organspezifisches Vorwissen über mechanische und physiologische Zusammenhänge zur Modellierung des Organs verarbeitet. Mit Hilfe dieser Modelle können die Werte einzelner Modellparameter in Abhängigkeit bestehender Randbedingungen berechnet werden. Während diese Art der Modellbildung vorwiegend auf numerischen Methoden basiert, ermöglichen Ansätze aus dem „Machine Learning“ und „Knowledge Discovery in Databases“ die Vorhersage von Parametern mittels statistischer Methoden. Hier können die Vorhersagen auf Basis von Merkmalen getroffen werden, die nicht unmittelbar ersichtlich in Wechselwirkung zueinander stehen und über die es unter Umständen auch noch keine gesicherten Erkenntnisse gibt.

Die vorliegende Arbeit untersucht die Atemmechanik des Menschen unter mechanischer Beatmung mit Methoden aus der Biomedizintechnik, der „Knowledge Discovery in Databases“ und dem „Machine Learning“. In allen vorgestellten Analysen werden Zeitreihendaten verwendet, die am beatmeten Patienten akquiriert wurden. Mittels klassischer Methoden aus der Biomedizintechnik wird die Relaxation der Lunge nach zuvor induziertem mechanischem Stress untersucht, um Rückschlüsse auf den Energietransfer zwischen dem Beatmungsgerät und der Lunge zu ziehen. In einem „Equation Discovery“ Ansatz werden Lungenmodelle auf Basis von vorher definierten Bedingungen für die Modellstruktur erstellt und mittels einer Heuristik qualitativ beurteilt. Dieser Ansatz ermöglicht die Konstruktion von Lungenmodellen mit und ohne Hintergrundwissen. In einem dritten Ansatz werden statistische Methoden aus dem maschinellen Lernen zur Vorhersage der druckabhängigen, nicht-linearen Volumendehnbarkeit der Lunge („Compliance“) verwendet. Zur Vorhersage werden Gauß'sche Prozesse verwendet, ein nicht-parametrischer Modellbildungsansatz für nicht-lineare Funktionen. In dem hier vorgestellten medizinischen Problemfeld bietet dieser Ansatz eine Methodik, die die Prädiktion beatmungsrelevanter Lungenparameter für eine auf den individuellen Patienten abgestimmte Beatmungsstrategie ermöglichen soll.

Abstract

In intensive care medicine, mechanical ventilation is one of the most important life-sustaining-therapies. However, an inadequate adjustment of the ventilator can induce or aggravate lung damages. Therefore, it is the aim of lung protective ventilation strategies to reduce the mechanical stress which the lung is exposed to during mechanical ventilation. To develop such strategies, the knowledge of physiological and mechanical interrelations and interactions is an essential prerequisite. In classical approaches from the field of biomedical engineering, organospecific prior knowledge about mechanical and physiological interrelations is used to model the organ. By means of such models, the values of particular model parameters can be calculated subject to existing boundary conditions. While this type of modeling is primarily based on numerical methods, approaches from the field of machine learning and knowledge discovery in databases allow to predict such parameters by statistical methods. Here, the predictions can be made on the basis of features, which do not correlate in an obvious way and for which possibly no verified knowledge exists.

The presented work investigates respiratory mechanics of the human lung under the condition of mechanical ventilation with methodologies from biomedical engineering, knowledge discovery in databases and machine learning. In all presented analyses, time series data acquired from mechanically ventilated patients are used. Relaxation processes within the lung after priorly induced mechanical stress are investigated by classical methods from the field of biomedical engineering. Such analyses are performed to draw conclusions with respect to the energy transfer between the ventilator and the lung. In the context of an equation discovery approach, lung models are generated on the basis of preliminarily defined constraints with respect to the model structure and are qualitatively evaluated by use of a heuristic. This approach enables the construction of lung models with or without prior knowledge. In a third approach, statistical methods from the field of machine learning are used to predict the pressure dependent, nonlinear compliance (volume distensibility) of the lung. The prediction task is solved by Gaussian processes, a nonparametric modeling approach for nonlinear functions. In the presented medical problem domain this approach provides a methodology that should allow for the prediction of parameters relevant for a patient specific adjustment of the ventilation strategy.

Danksagung

Eine Dissertation zu verfassen ist zu guter Letzt immer auch ein ordentliches Stück weit Teamarbeit. Ich möchte mich deshalb bei allen meinen Kolleginnen und Kollegen, die mich auf meinen verschiedenen wissenschaftlichen Stationen[1] begleitet haben, bedanken. Für ihre Bereitschaft, im besten wissenschaftlichen Sinne immer wieder zu diskutieren, konstruktiv zu sein und – wann immer nötig – zu unterstützen.

Mein besonderer Dank gilt meinem Betreuer und Doktorvater[1] für das Vertrauen in meine Arbeit. Sein fachkundiger Rat und die gemeinsamen Diskussionen haben meine Arbeit stets vorangebracht. Seine umfassende und dennoch pragmatische Herangehensweise an alle auftauchenden Fragestellungen schätze ich sehr.

Genauso sehr möchte ich mich bei meinem Betreuer und Zweitgutachter[1] bedanken. Seine Begeisterung für die Wissenschaft war mir stets Inspiration und sein leidenschaftliches Engagement für unsere Arbeitsgruppe hat mich nachhaltig beeindruckt. Ich bedanke mich sehr für seine fortwährende Unterstützung.

Insbesondere bedanke ich mich auch bei meiner Familie, die mir in schwierigen Phasen immer den Rücken gestärkt und mir gezeigt hat, die Dinge gelassen zu sehen. Und dabei ganz besonders herzlich bei meiner Frau[1], die mich ermutigt hat, diese Arbeit tatsächlich zu erstellen und die mir all die dazu nötige Freiheit gegeben hat.

[1] Personenbezogene Angaben in der Online-Version gelöscht.

Chapter 1

Introduction

In intensive care medicine, mechanical ventilation often is the only option to keep patients alive. State of the art ventilator devices offer diverse breathing modes and patterns for the controlled ventilation of patients suffering from respiratory insufficiency [46]. A special issue in the field of mechanical ventilation is to apply lung protective ventilation strategies. Although it is well-known that the lung may suffer severe damage from inadequate ventilator settings by excessive mechanical stress, the responsible mechanical characteristics of the human lung are only little known and in actual medical research great efforts are made to gain more insight into lung mechanics under the condition of mechanical ventilation (Table 1.1).

With different techniques from machine learning (ML), knowledge discovery in databases (KDD) and biomedical engineering, this work approaches this current topic from intensive care medicine. Time series data, recorded from the mechanically ventilated human lung, are explored to gain insight into the mechanical behavior of the lung under passive respiration. Especially the fields of ML and KDD offer methods to predict features with uncertain constraints, a capability which is not given in classical biomedical engineering. Thus, the capability of such predictive approaches concerning the interaction between lung parameters are explored. In medical practice, the observation of these parameters are most important for decision making in terms of lung protective ventilation strategies.

This work is organized as follows. In the introduction section, the motivation of this work, the medical problem domain and the data acquisition are presented. Part of this thesis has been previously published [36, 38, 39, 40] and we refer to these publications in the chapters 2, 3 and 4. Chapter 2 presents the investigation of stress relaxation processes by means of classical biomedical engineering. Chapter 3 addresses the modeling of the ventilated human lung by an equation discovery (ED) approach. Chapter 4 deals with the prediction of lung parameters by Gaussian process (GP) modeling. As the applied techniques differ substantially, each section elaborates on the underlying technical aspects.

Additional information concerning technical aspects of chapters 2, 3 and 4 are presented in the appendices A, B, C and D, respectively. The appendix sections are followed by the bibliography, a section listing the used symbols, a section providing the used acronyms and finally a glossary and index.

1.1 Motivation

Mechanical ventilation is the life-saving therapy in intensive care by all means. However, inadequate ventilator settings can induce or aggravate lung injury during mechanical ventilation [109]. To prevent such ventilator associated lung injury (VALI), lung protective ventilation strategies are essential. Such strategies have been shown to considerably improve the outcome of critically ill patients [109]. A basic factor with respect to lung protective mechanical ventilation is the level of applied pressure. Appropriately adjusted, it reduces the mechanical stress of the lung tissue and hence its irreversible damage: a continuous minimum pressure level prevents the collapse of the alveoli. If – on the other hand – pressure becomes exceedingly high, the patient might suffer severe lung damage. The manual customization of this pressure level is very sensitive and has to be set individually for each patient. The initial decision for a given ventilatory approach determines the overall direction of treatment. Furthermore, the effects of restructuring processes of the human lung with its seemingly viscous structures delay the measurable response of mechanical stress. Hence, a wrong initial approach may become visible delayed in time for hours or even days and the causality of the observed alterations might even remain undiscovered. Moreover, the measurement of appropriate mechanical parameters is related to potentially harmful respiratory maneuvers, intermitting the regular mechanical breathing pattern. To determine the pressure-volume interval of an optimized mechanical ventilation, as much as possible of the complete respiratory pressure-volume capacity has to be observed. Thus, data acquisition itself requires the application of exceedingly high pressure and volume ranges.

"From a mechanistic point of view, all living systems are composed of processes. These processes act, or interact, through manipulation of molecular mechanisms, chemical concentrations, ionic electrical current, and/or mechanical forces and displacements. A physiological process performs some operation(s) or manipulation(s) in response to a specific input (or inputs), which gives rise to a specific output (or outputs). In this regard, a process is the same as a system [...] To study and quantify complex processes, we often impose rather severe simplifying constraints. The most common assumption is that the process and its components or sub-processes behave in a linear manner, and that their basic characteristics do not change over time. [...] Of course, most living systems change over time, are adaptive, and are often nonlinear." (John Semmlow [102], 2005, p. 9). Such assumptions are also made when modeling the respiratory system. However, the lung is a most dynamic system – not only in terms of the continuous work of breathing. As men-

tioned above, mechanical ventilation is assumed to cause restructuring processes within the lung tissue and therefore the mechanical behavior of the lung changes over time. Observable lung parameters are showing a nonlinear behavior not only according to the applied pressure level but also dependent on time. It is an important task for the physician to anticipate such changes in order to prevent lung damage. Unfortunately, only few parameters are known to estimate the mechanical stress of the lung and not many of the complex mechanical interactions within the pulmonary system are known. Consequently, three questions can be identified: how can we get more insight into lung mechanics? How can we improve the corresponding techniques? How can we predict mechanical lung parameters relevant for lung protective ventilation?

"This assumption [that basic characteristics of a process do not change over time] is referred to as the 'linear time-invariant' model. Such an assumption allows us to apply a powerful array of mathematical tools that are known collectively as linear system analysis. [...] There are several ways to study a linear system [...]: analog analysis using analog models, and systems analysis using system models. [...] The primary difference between analog and systems analysis is the way the underlying physiological processes are represented. In analog analysis, individual components are represented by analogous elements. Often these elements show detailed structures and provide some insight into the way in which a given process is implemented, although they may also represent processes more globally. In systems analysis, a whole process can be represented by a single mathematical equation. The advantage of using analog analysis is that the model is often closer to the underlying physiological processes. Conversely, analyzing at the process level, as in systems analysis, provides a more succinct description and offers a better overall view of the system under study. In addition, the more abstract representation provided by systems models emphasizes behavioral characteristics and may aid in identifying behavioral similarities between processes that contain quite different elements." (John Semmlow [102], 2005, pp. 10, 11). In this work, both approaches are used for modeling the human lung.

In our first study, the method of choice is a classical approach from biomedical engineering when we use an analog model in a piecewise linear approach. We put the focus on viscoelastic tissue properties. Viscoelastic effects of the human lung have been examined in detail [1, 3, 11, 21, 22, 29, 30, 53, 84, 85, 86, 103] and should be considered with respect to lung protective ventilation. However, aspects of nonlinear behavior have rarely been taken into account and if so, only in a limited way [22, 29, 103]. Therefore, our first study analyzes stress relaxation processes with a focus on nonlinear behavior. The model of choice is the well accepted spring-and-dashpot model [6], which includes the representation of viscoelastic and stress relaxation effects (chapter 2, *Analysis of Stress Relaxation in the Human Lung*).

The spring-and-dashpot model is very well suited to describe the focused effects, but it does not represent other aspects of lung mechanics and it is in particular not a model of the complete respiratory system. As mentioned above, in biomedical engineering the analog

system approach represents biological systems in more detail whereas system models apply a more abstract, "black box" like approach. Sometimes such black boxes are either helpful or unavoidable, if for example a more general approach is preferable or there is a lack of detailed knowledge of a specific model component, respectively. The overall aim of course is to gain new insight into a system and to model it in more and more detail. But this is generally mutually dependent. To get more insight one needs more detailed models to represent specific characteristics in order to investigate new ones. Vice versa, one needs knowledge of specific characteristic aspects when constructing a model. In the second study, we present a semiautomatic ED approach to create lung models "from scratch", i.e. without – or with only little – background knowledge. This could be most helpful to gain more insight into different aspects of lung mechanics when required background knowledge is missing (chapter 3, *Modeling of Respiratory Mechanics*).

For lung protective ventilation, the objective must be an individualized estimation of the lung status. Ideally, the physician should be able to anticipate changes of the lung status, represented by specific parameters, when adjusting the ventilator. Unfortunately, given a lung model, individualization can be achieved only by changing the boundary conditions. The model itself is a general presentation of the lung. In the best case, a model would represent a specific class of the lung damage or the lung status, respectively, corresponding to the specific patient. However, in many cases such a general model might not describe the patient specific lung status sufficiently to allow approximately correct parameter predictions and thus support the physician when making his or her decision concerning ventilatory settings. In the ED approach, a semiautomatic modeling technique is provided. With the resulting models, the changes of lung parameters in dependence of changing boundary conditions can be simulated and thus predicted. One could think of this approach to be frequently applied to patient data recorded at runtime and then frequently generating an updated model. However, the model finding process is very time-consuming. Fortunately, the fields of ML and KDD provide techniques to model feature dependencies without any background knowledge, but only on previous observations of parameters. Depending on the complexity of the problem and the size of the dataset, such models can even be generated in adequate time. Therefore, in our third study, we go one step further: we predict lung parameters without any knowledge of the underlying processes and dependencies by application of GP modeling. In medical practice, the benefit of this approach would be the capability to frequently generate actualized, patient specific models which could be used for an individual adjustment of the ventilator (chapter 4, *Prediction of Mechanical Lung Parameters*).

Summarizing, this work is strongly motivated by the aim to gain insight into lung mechanics and to show the potential of the applied techniques. The investigated questions are of actual interest and include most practical aspects in terms of lung protective ventilation. Finally, a considerable amount of medical real-world data should allow for showing the practical benefit of the results.

1.2 Medical Background

In the following section, we will introduce the reader to the most relevant concepts and the terminology of this study from a medical problem domain.

1.2.1 The Acute Respiratory Distress Syndrome (ARDS)

The acute respiratory distress syndrome (ARDS) [109] was mentioned for the first time in 1967 by Ashbaugh *et al.* [2]. ARDS is not a specific disease but a response to an acute injury of the lungs. It is defined as a complex of symptoms including acute dyspnea, hypoxemia and reduced lung compliance. Diverse types of trauma may lead to ARDS, including breathing in vomited stomach contents (aspiration), inhalation of toxic fumes, widespread infection of the lungs as in bilateral pneumonia, sepsis (bloodstream infection), near drowning, a major blood loss, shock, direct trauma to the chest, and some drug overdoses. A distinction is made between pulmonary and extra-pulmonary ARDS. While in pulmonary ARDS the alveolar epithelium is directly injured, the extra-pulmonary ARDS is characterized by an indirect injury of vascular endothelium by inflammatory mediators such as abdominal peritonitis or occlusion, acute pancreatitis or abdominal trauma. ARDS is a life-threatening condition. Mortality rate estimates range from 30 to 70 percent [83]. Many people surviving ARDS make a full recovery. However, some survivors have lasting damage to their lungs.

1.2.2 Mechanical Lung Parameters

In the 1990s, low tidal volume and pressure-limited ventilation were supposed to lower mortality in patients mechanically ventilated for ARDS [47]. In a way, this was the beginning of lung protective ventilation strategies [41]. Since then, a variety of such strategies targeting the reduction of VALI has been proposed [26, 106, 109].

A basic concept for lung protective ventilation is the optimization of the respirator settings with respect to the volume distensibility or compliance of the mechanically ventilated lung. During mechanical ventilation basically two indicators can be observed: Firstly, the compliance C of the respiratory system, which is determined by the change of respiratory volume V divided by the change of pressure P in the alveoli (Figure 1.1.a), i.e.

$$C = \frac{\Delta V}{\Delta P} \tag{1.1}$$

Secondly, the Newtonian airway resistance R of the respiratory system. Ideally, the lung

shows a high compliance and a low resistance. The critically ill ARDS lung is characterized by the contrary, a low compliance and a high resistance.

The compliance depends on the strain of the lung tissue. This strain again is highly influenced by the stress [19], respectively the pressure applied to the respiratory system. Under the condition of mechanical ventilation a high compliance of the lung is assumed to be associated with a minimization of the mechanical stress to the lung tissue and hence a reduction of the risk of irreversible damage of the respiratory system [75]. For each patient, the maximum compliance C_{max} is found within an individual pressure range.

1.2.3 Lung Protective Ventilation

For a lung protective ventilation, the pressure range associated to the maximum compliance C_{max} has to be determined and the ventilator settings appropriately adjusted. A well accepted procedure is to inflate the lung over large pressure-volume ranges, covering almost the complete inspiratory capacity [49] (Figure 1.1.a). Volume is inflated with almost zero flow (so-called "static" conditions). Thus, as the flow induced pressure fraction hardly exists, the curve refers to alveolar pressure, i.e. depicts the pressure conditions at the small lung structures under static conditions. The inspiratory limb typically shows a sigmoid curve shape. The maximum compliance at the inspiratory limb is detected at the curve interval with the steepest slope. This is supposed to be the optimal pressure-volume range for a lung protective ventilation [70].

To ventilate the lung within this pressure-volume range, the ventilator is adjusted to a basic pressure offset onto which the regular mechanical breaths are applied. This basic pressure level is called the positive end-expiratory pressure or PEEP. With respect to the compliance-pressure curve, which is the first derivative of the sigmoid pressure-volume curve, in principle three physiological situations can be distinguished (Figure 1.1.b):

(i) alveolar underdistension (atelectasis)

(ii) alveolar overdistension

(iii) normal alveolar distension

Situation (i) is characterized by increasing compliance, situation (ii) is characterized by decreasing compliance, whereas situation (iii) is characterized by an almost constant maximal compliance. Situation (i) requires an increase of PEEP, situation (ii) requires a decrease of PEEP whereas situation (iii) characterizes an adequate PEEP setting [74].

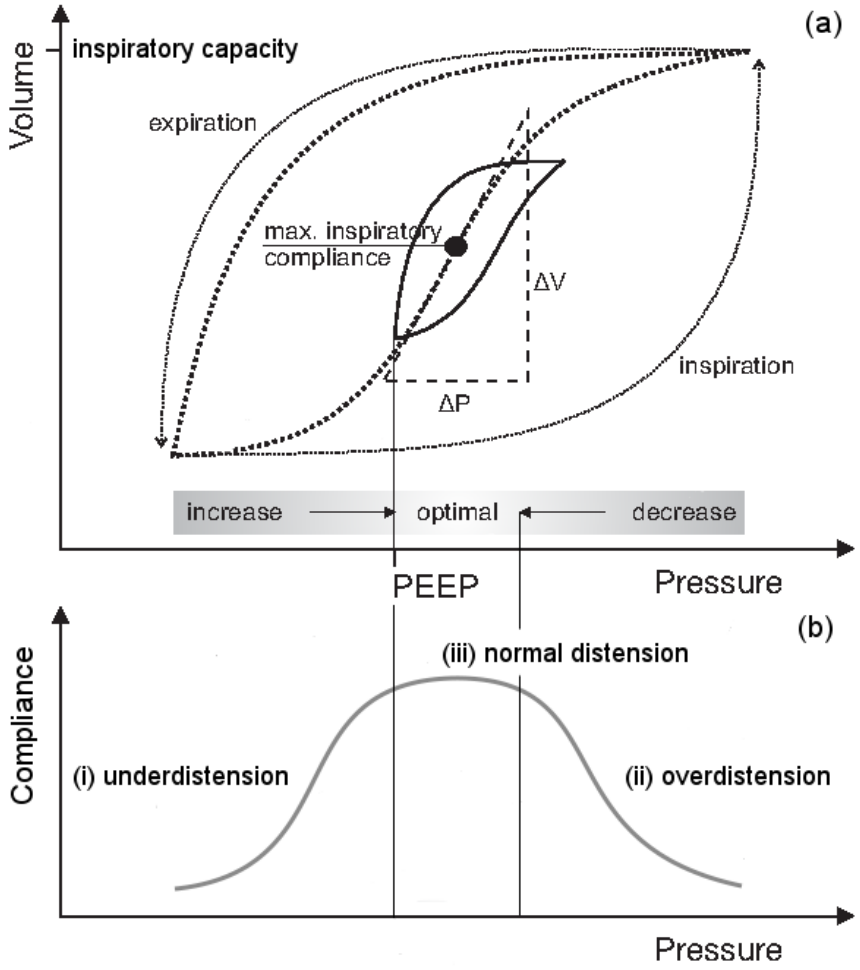

Figure 1.1: (a) Schematized pressure-volume loops measured under static (dotted large loop: inspiratory flow ≈ 0 mL/sec) and dynamic (straight small loop: inspiratory flow > 0 mL/sec) conditions. Under static conditions, the loop covers the complete inspiratory capacity. The inspiratory limb typically shows a sigmoid shape. After reaching a maximum pressure (determined beforehand to avoid lung injury), expiration starts. The expiratory limb exhibits a fast decrease in pressure at the beginning, which is decreasing in speed at the end of expiration. In the center of part (a) a dynamic loop is depicted. Such a dynamic loop accords to a regular breath within the tidal volume. To optimize the ventilation, the positive end-expiratory pressure (PEEP) should be adjusted to a level where a maximal compliance can be expected. The pressure gap between the static and the dynamic loop results from the flow induced pressure fraction under dynamic conditions. (b) Schematized compliance-pressure curve (Equation (1.1)) as the derivative of the sigmoid shaped inspiratory limb of the pressure-volume loop. (i) The range of alveolar underdistension (atelectasis), (ii) alveolar overdistension and (iii) normal alveolar distension.

1.2.4 Techniques for the Estimation of Lung Parameters

Different techniques can be used to obtain a numerical description of the lung in terms of mechanical lung parameters. While in two-point methods data are obtained from lung inflation with almost zero airflow, in multiple-point methods parameter estimation is based on data obtained during regular breathing. Besides for general investigations of the human lung, both techniques are also used for acute survey at bedside. Another technique to get insight into lung mechanics in general is the investigation of stress relaxation processes. Stress relaxation processes are related to viscoelastic effects and equilibration effects of inhomogeneously distributed air volumina within the lung ("Pendelluft" phenomenon). A well accepted model to represent such effects is the so-called spring-and-dashpot model [6]. All these techniques and methods are described in this section.

Two-Point Methods

While in medical practice the resistance of the respiratory system is generally not evaluated (Figure 1.2), typically, the compliance C is determined by so-called two-point methods. As mentioned above, the compliance is determined as quotient of the volume change and the change of alveolar pressure, i.e. $C = \Delta V/\Delta P$ (Equation (1.1)). Under static conditions, i.e. when the airflow equals 0 mL/sec, the alveolar pressure is measurable as the externally recordable airway pressure which equals the alveolar pressure. Under these zero flow conditions, an additional resistive pressure component induced by the airflow is omitted.

To obtain such static conditions, a commonly applied technique is to set an end-inspiratory pause (EIP) by interrupting the airflow upon the end of inspiration for at least 1 second. In this way, the compliance can be calculated from the applied tidal volume and the difference between the end-inspiratory and the end-expiratory pressure after complete expiration following the EIP. In case the expiration has been complete, the end-expiratory pressure corresponds to the PEEP. In case of an incomplete expiration, the end-expiratory alveolar pressure is higher than the airway pressure and is called intrinsic PEEP. An intrinsic PEEP has to be considered when calculating the compliance, as otherwise the compliance can be underestimated by 35% [94].

Multiple-Point Methods: The Equation of Motion (EOM)

In contrast to the two-point methods, where two defined zero flow points (EIP and end of expiration) are related for compliance estimation, the so-called multiple-point methods relate continuously measured data points. Here, a model equation is fitted to the data via least-squares-fitting [17]. Over two-point methods, multiple-point methods have the advantage that the analysis can be done breathwise under continuous ventilation [45]. No

respiratory maneuvers are needed, especially no airflow interruptions. Moreover, compliance and resistance are evaluated simultaneously.

The model on which the multiple-point methods are based on is the *linear equation of motion* for the respiratory system [46]. The pressure difference between the external atmosphere and the airways ΔP is in balance with three pressure components of the respiratory system and can be described by a first-order differential equation:

$$\Delta P(t) = f_1(V(t)) + f_2(\dot{V}(t)) + f_3(\ddot{V}(t)) \tag{1.2}$$

"The function f_1 represents the elastic pressure fraction [P_{el}] of the respiratory system which is resulting from the sum of all elastic properties of the respiratory system. In the simplest case the elastic pressure fraction is proportional to the volume displacement: $P_{el} = E \times V(t)$. The proportionality factor is the elastance E of the lung, the reciprocal of which is called volume distensibility or compliance [i.e. $P_{el} = 1/C \times V(t)$]. The function f_2 represents the resistive pressure fraction [P_{res}], which in the simplest case is proportional to the volume change $\dot{V}$: $P_{el} = R \times (\dot{V}(t))$. Its proportionality factor is the flow resistance [R]. The third function f_3 represents the pressure fraction which is caused by the inertia of the respiratory system [P_{in}]. In the simplest case the pressure fraction caused by inertia is proportional to the volume acceleration $\ddot{V}$: $P_{in} = I \times \ddot{V}(t)$. The proportionality factor is the inertance I." (Citation translated from Josef Guttmann [43], 1993, p. 2). As P_{in} is negligibly small, it is not used in practice. On the other hand, under the condition of mechanical ventilation, the PEEP is used as constant pressure offset to "keep the lungs open". Finally, the equation of motion for the respiratory system is defined by

$$\Delta P(t) = 1/C \times V(t) + R \times \dot{V}(t) + PEEP \tag{1.3}$$

where C denotes the lung compliance, R denotes the resistance of the respiratory system and PEEP the offset of the airway pressure at the end of expiration. By fitting the EOM to flow, pressure and volume time series data (Figures 1.2.b, 3.3.b-d), the compliance and resistance can be calculated from continuously sampled data [45]. However, as in the two-point methods, nonlinear aspects like turbulent flow and viscoelastic tissue properties, the latter observable as stress relaxation processes, are not included in this model.

Stress Relaxation Processes

Exposing the lung tissue to an abrupt change in volume causes a stress relaxation response, which is a power function of time. The corresponding stress relaxation curve represents the measurable process of viscoelastic energy dissipation under zero flow condition (Figure 1.2). The latter can be obtained by rapid flow interruption, a method based on the

interrupter technique introduced in 1927 by von Neergaard and Wirz [114, 115]: By the sudden interruption of (inspiratory) airflow due to rapid airway occlusion, the respiratory pressure instantaneously drops by the amount of the product airflow rate (immediately preceding flow interruption) times Newtonian airway resistance of the respiratory system. Vuilleumier [117] was the first to observe that the initial rapid pressure drop is followed by a slow pressure decrease. This second pressure decrease is due to stress relaxation processes of the viscoelastic lung tissue, but can also be interpreted as effect of volume equilibration between lung compartments with different time constants ("Pendelluft" phenomenon of the inhomogeneous lung) [5, 8, 95]. From the mere phenomenon of a slow pressure change both effects cannot be distinguished.

During the past few decades, the effects of stress relaxation caused by the viscoelastic properties of lung tissue have been intensively investigated by model-based analysis techniques [1, 3, 11, 21, 22, 29, 30, 53, 84, 85, 86, 103] (Table 1.1). In these studies, viscoelastic parameters were usually assumed to be constant. However, Eissa *et al.* [29] found that this assumption holds true only for the baseline tidal volume range on zero end-expiratory pressure (ZEEP) and up to applied volumes of 0.7 L. It was speculated that this might reflect nonlinear viscoelastic behavior for higher pulmonary volumes. In addition, Sharp and colleagues [103] reported that when inflating normal lungs with successive steps of equal volume (0.5 L), up to a final volume of 3.0 L, the amplitude of the slow pressure drop owing to stress adaptation increases nonlinearly with inflation volume. However, the approaches applied in these studies were not specifically designed to quantify such nonlinear effects or their progression over wide ranges of pressure and volume. Moreover, the dynamic loading process during volume inflation has not been taken into account because parameter estimation has been exclusively based on the stress relaxation curves under static zero flow conditions.

The Spring-and-Dashpot Model

Different mathematical model concepts have been introduced [8] for the interpretation of stress relaxation processes of the lung. Already in 1955, Mount [76] introduced a mechanical model considering relaxation processes. Accordingly, stress relaxation due to viscoelasticity is conventionally modeled using Maxwell bodies, which are elastic springs connected to resistive dashpots [33]. Bates *et al.* [6] showed that this so-called spring-and-dashpot model appropriately explains viscoelastic properties of the lung tissue. It has been validated with data of isolated lung strips [68, 78, 108] and respiratory data obtained from animals [9, 55, 104]. In clinical investigations [21, 30, 85], within the tidal volume range, the impact of flow, volume and frequency on elastance and resistance of the respiratory system was analyzed.

For the present work, an electrical analog of a spring-and-dashpot model [11, 53] was used for parameter estimation (Figure 1.3). The model consists of two components: (1) A

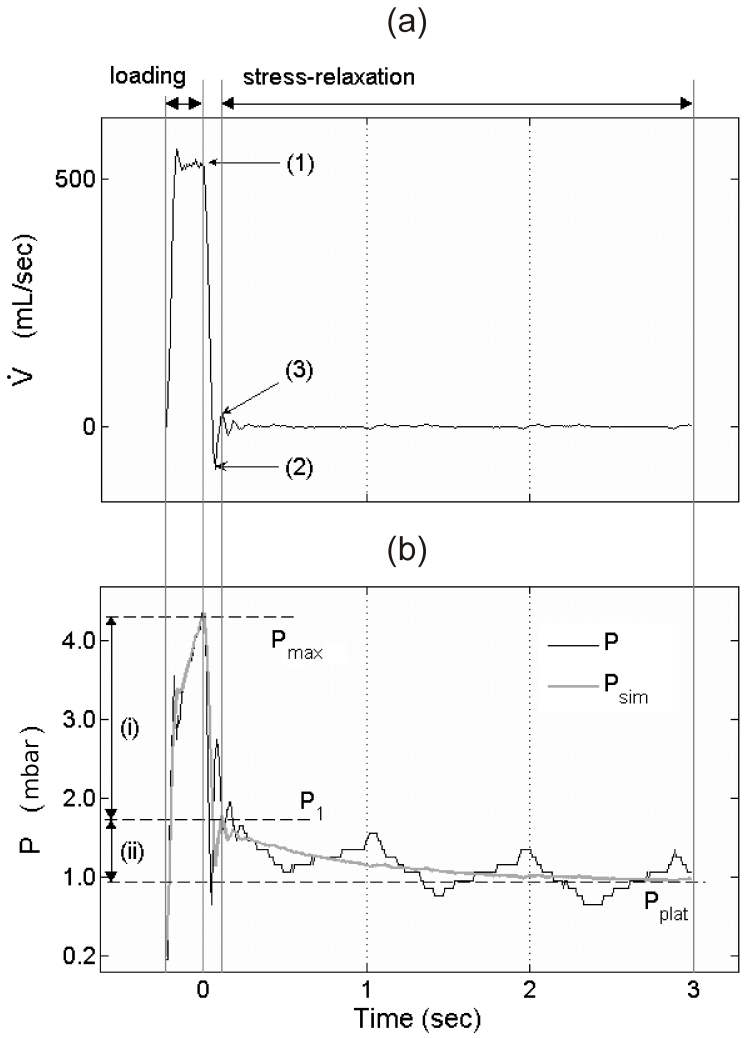

Figure 1.2: Stress relaxation processes. (a) Respiratory flow $\dot{V}$ and (b) pressure P time series of one 100 mL volume step including the phases of volume loading ($\dot{V} > 0$ mL/sec) and stress relaxation ($\dot{V} = 0$ mL/sec during occlusion interval). (a) Labeled points indicate: (1) start of valve closure, (2) flow falling below zero due to valve characteristics, (3) estimated end of valve closure. The data between (1) and (3) show typical oscillations, induced by the closure of the occlusion valve. Therefore, this interval was excluded when fitting the spring-and-dashpot model to the data. (b) P with maximum pressure (P_{max}) and approximated plateau pressure (P_{plat}). P_{sim} depicts the model-simulated respiratory pressure by use of the fitted parameter values from the electrical analog of the spring-and-dashpot model. (i) denotes the initial resistive pressure drop (P_{max} down to P_1). To determine the resistance of the respiratory system, the quick resistive pressure drop is divided by the airflow rate which is met immediately before airway occlusion. (ii) denotes the succeeding slow pressure change indicating stress relaxation between level P_1 and P_{plat}. The regular oscillations within this interval are induced by the heart beat (cardiogenic oscillations).

Table 1.1: Related work investigating pulmonary stress relaxation processes in the lungs, based on viscoelastic models.

Study	*Year*	*Lung Tissue*		*Analysis of*	*Analysis of*	*Analysis of*	
		Normal	*Injured*	*P/V Range**	*Nonlinearity*	*Loading*	*Relaxation*
Sharp *et al.* [103]	1967	x	x†	inspiratory capacity‡	x§	-	x
D'Angelo *et al.* [21]	1989	x	-	partial	-	-	x
Eissa *et al.* [30]	1991	-	x	partial	-	-	x
Pesenti *et al.* [86]	1991	x	x	partial	-	-	x
D'Angelo *et al.* [22]	1992	x	-	partial	x$^{\|}$	-	x
Eissa *et al.* [29]	1992	-	x	partial	x$^{\sharp}$	-	x
Jonson *et al.* [53]	1993	x	-	partial	-	-	x
Pelosi *et al.* [84]	1995	x	x	partial	-	-	x
Beydon *et al.* [11]	1996	-	x	partial	-	-	x
Pelosi *et al.* [85]	1997	-	x¶	partial	-	-	x
Antonaglia *et al.* [1]	2000	x	x	partial	-	-	x
Bates *et al.* [3]	2007	-	x$^{\&}$	partial$^{\pounds}$	-	x	x
Ganzert *et al.* [38]	2009	x	x	inspiratory capacity**	x$^{\$}$	x	x

$*$ partial ranges defined by particular PEEP levels and/or tidal volumes; † restrictive thoracic abnormalities; ‡ up to 3 L; § analysis of stress-adaptation quantified by the amount of slow pressure drop after successively applied 0.5 L volume steps; ‖ viscoelastic model parameters R_{ve} and τ_{ve} (Equation (1.4)) determined for ZEEP and one level of PEEP; ♯ viscoelastic parameters R_{ve}, E_{ve} ($= 1/C_{ve}$) and τ_{ve} for PEEP levels of 0, 5, 10, 15 mbar; ¶ obese patients; & degassed, washed canine lung strips; £ unidirectional 10% length changes starting from three initial lengths; $**$ up to a maximum plateau pressure of 45 mbar; $ analysis of viscoelastic parameters R_{ve}, C_{ve} and τ_{ve} for successively applied 0.1 L volume steps.

Newtonian airway resistance R and a (static) compliance of the respiratory system C and (2) the electrical analog of a resistive dashpot R_{ve} and an elastic spring C_{ve} as resistance and compliance of the component which is modeling viscoelastic behavior. The viscoelastic time constant τ_{ve} of the viscoelastic model component quantifies the stress relaxation dynamics of the system and is determined by the product of viscoelastic resistance R_{ve} and viscoelastic compliance C_{ve} as

$$\tau_{ve} = R_{ve} \times C_{ve} \tag{1.4}$$

Referring to Figure 1.2, the course of the slow pressure decrease can be calculated by

$$P(t) = P_1 \times e^{-t/\tau_{ve}} \tag{1.5}$$

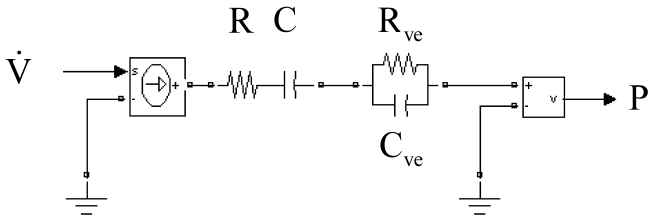

Figure 1.3: Electrical circuit analog to the spring-and-dashpot model. R denotes the Newtonian airway resistance and C the (static) compliance. R_{ve} and C_{ve} are the resistance and the compliance of the viscoelastic component, respectively. The respiratory airflow $\dot{V}$ represents the input and the respiratory pressure P the output of the model.

1.2.5 Techniques for Measuring Respiratory Data

In general, the lung compliance is calculated from pressure-volume data recorded under quasi-static conditions. This means that the resistive components being induced by the airflow are almost eliminated, i.e. airflow $\dot{V} \approx 0mL/sec$. A diversity of standardized measuring techniques has been introduced to allow data recording under quasi-static conditions, being appropriate for the analysis of respiratory mechanics.

During low-flow inflation maneuvers [49], large volumes are applied to the lung using extremely low airflow rates. As a consequence, the lung compliance can be calculated directly from the continuously recorded pressure-volume relationship. In the so-called SCASS (static compliance by automated single steps) method [107] different amounts of volume are applied breathwise. These tidal volumes differ gradually. For each breath, the volume to inflate is chosen at random. This procedure is iterated until each of the predefined volumes has been applied. After the inspiratory part of each maneuver breath, the airflow is interrupted for several seconds. At the end of such an occlusion maneuver, static conditions are reached and pressure and volume are recorded for the subsequent analysis. To minimize hysteresis effects[1], several breaths of normal tidal volume are applied between the maneuver breaths. In this way, measurements of the whole volume of the lung are taken, and the compliance can be computed under quasi-static conditions for predefined, gradual volume steps.

However, it is discussed in the literature, in which way these quasi-static measurements in fact reflect the status of the lung or if respiratory data should rather be obtained under dynamic conditions [105]. In Figure 1.1 the difference of pressure-volume loops recorded under static and dynamic conditions is schematized. A method providing dynamic measurements has been introduced by Guttmann *et al.* [45]. Here, data are recorded during

[1]Characterized by differing pressure-volume curves during in- and expiration due to different elastic retraction forces.

continuous mechanical ventilation. The applied volumes do not exceed regular ventilatory settings according to the tidal volume of the individual lung. From this data, the compliance and resistance of the respiratory system are calculated by fitting the EOM model via regression analysis. For the studies presented within this thesis, data obtained by the two measuring methods "super syringe maneuver" and "PEEP wave maneuver" as described in the following sections were used.

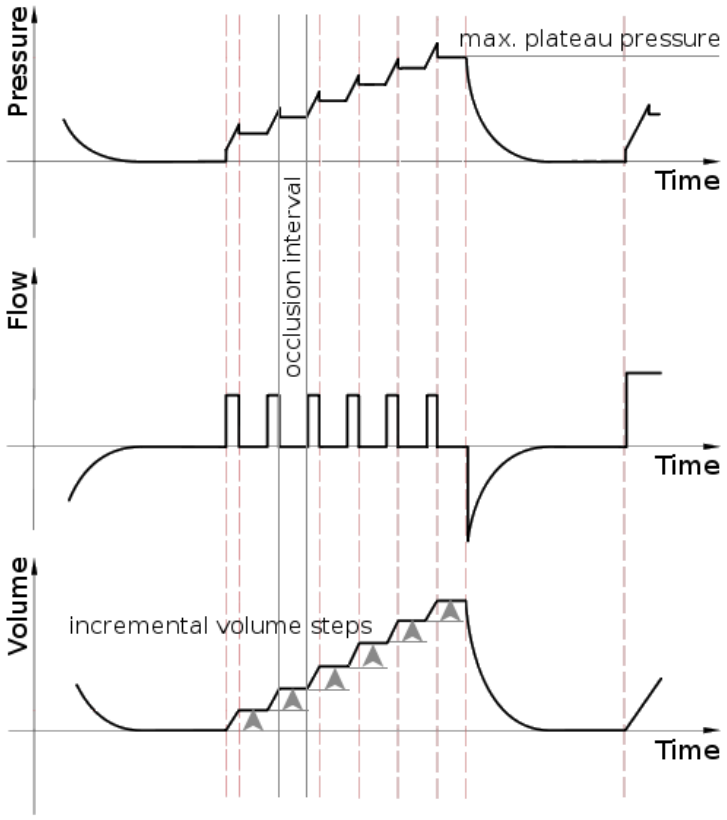

Figure 1.4: Scheme of super syringe maneuver. The three measured data time series volume, flow and pressure are depicted against time. Volume is increased with constant inspiratory airflow rates. At the end of each such volume step, the airway is occluded for several seconds. This is a prerequisite for the calculation of the static compliance, i.e. the pressure-volume relation. With each volume increase, the plateau pressure level increases as well. Volume is stepwise increased until a predefined maximum plateau pressure level is reached. See Figure 1.2 for details of the occlusion interval. Graphics based on [25].

The Super Syringe Maneuver

The super syringe maneuver [69] enables the repetitive calculation of the static compliance at multiple pressure-volume levels over a wide range of the inspiratory capacity. This allows to relate the compliance to the distension of the lung. Figure 1.4 schematically depicts the maneuver: performing super syringe maneuvers, constant volume portions are repetitively applied. In this way, the volume is stepwise incremented. All volume portions are inflated with a constant inspiratory airflow rate. With the volume increasing, the airway pressure increases as well. Volume portions are applied until a predefined maximum plateau pressure level is reached. Ideally, the maneuver starts from zero PEEP after complete expiration. At the end of each volume step, the airflow is interrupted for a predefined time-interval of several seconds (Figure 1.2). At the end of each such inspiratory break, a static pressure-volume relation is reached and is used to calculate the static compliance (Equation (1.1)).

The maneuver is conducted in volume-controlled mode. In its original form, the maneuver implies a volume increase as well as a volume decrease. In our study solely the increasing limb of the maneuver is conducted. A complete super syringe maneuver is depicted in Figure 1.6.

The PEEP Wave Maneuver

Alike the super syringe maneuver, the PEEP wave maneuver [88] enables the calculation of the lung compliance at several pressure-volume levels over a wide range of the inspiratory capacity. Figure 1.5 schematically depicts the maneuver: during PEEP wave maneuvers, continuous mechanical ventilation is applied with short end-inspiratory ($< 1\,\text{sec}$) and end-expiratory zero-flow phases. In the maneuver, the PEEP level is successively increased until a maximum plateau pressure level is reached. Starting from 0 mbar the PEEP level is increased in predefined equally sized pressure steps. Between two PEEP level increases, a predefined amount (minimum of three [88]) of breaths is applied. This is due to the fact that after a pressure increase, the lung needs more than one breath to stabilize at the new PEEP level. This stabilization is necessary for gaining appropriate data for compliance calculations. While for each breath identical volume portions are inflated at constant inspiratory airflow rates, the exhaled volume differs. Compliance calculations are based on the discrepancy of the inspiratory and the expiratory tidal volume. Based on this data, the static compliance as well as the so-called “incremental compliance” can be calculated.

The maneuver is conducted in volume-controlled mode. In its original form, the maneuver implies a PEEP level increase as well as a PEEP level decrease until the zero PEEP level is reached again. In our study, PEEP wave maneuvers solely implied the increase of the pressure level. A complete PEEP wave maneuver is depicted in Figure 1.7.

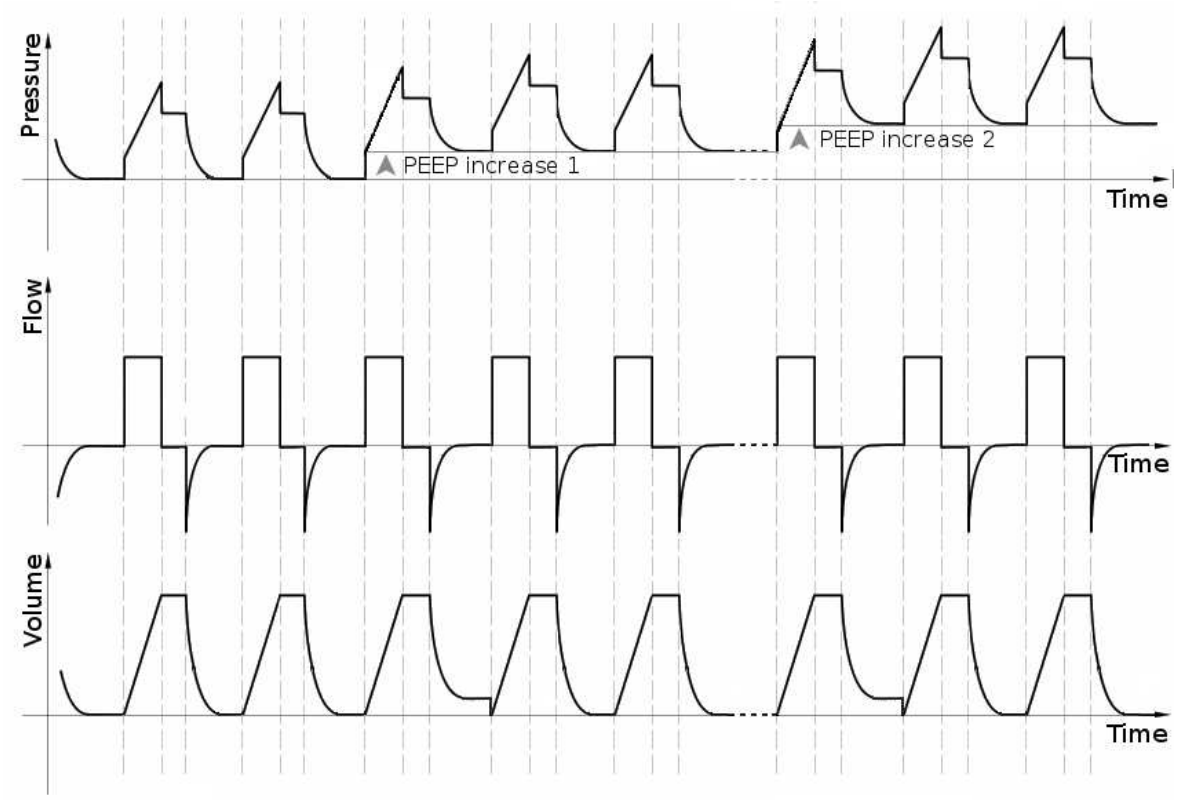

Figure 1.5: Scheme of PEEP wave maneuver. The three measured data time series volume, flow and pressure are depicted against time. During a PEEP wave maneuver regular breaths are applied to the patient. Starting from 0 mbar the PEEP level is increased in predefined pressure steps. Within the maneuver merely the inspiratory and the exhaled volume is recorded, but not the absolute intrapulmonary volume. Therefore, after each exhalation, the volume is reset to 0 mL. This can be seen most clearly after a pressure increase step. At this point, the volume is not exhaled completely, which increases the intrapulmonary gas volume. For the calculation of the static compliance based on a PEEP wave maneuver the absolute intrapulmonary gas volume is not relevant as the method is based on the difference between the inspiratory and expiratory volume. Graphics based on [25].

1.3 Medical Data

Data were obtained from two patient studies: (i) a multicenter study including 28 mechanically ventilated patients suffering from ARDS [35, 38, 105] (ARDS group); and (ii) a study including 13 mechanically ventilated patients under conditions of preoperative anesthesia [38, 122] (control group). For the first study (chapter 2), data of both groups were analyzed in order to gain insight into characteristics of the damaged lung suffering from

ARDS in comparison to the healthy lung. In the second study (chapter 3), data of the control group representing the healthy human lung were analyzed in terms of a modeling task. In the third study (chapter 4), data of the ARDS group representing the damaged human lung were analyzed in order to predict lung parameters as essential indicators in context of lung protective ventilation strategies.

Data from super syringe maneuvers were available from 20 of 28 patients of the ARDS group and from 11 of 13 patients of the control group. Both patient studies were approved by the local ethics committees. Written informed consent was obtained from patients, next of kin or a legal representative. Automated respiratory maneuvers were applied using identical equipment (Evita4Lab-system, Dräger Medical, Lübeck, Germany). Gas flow was measured using a pneumotachograph (Fleisch No. 2, F+G GmbH, Hechingen, Germany). Volume was determined by integration of the flow signal. Airway pressure was measured using a differential pressure transducer (PC100SDSF, Hoffrichter, Schwerin, Germany). Flow and pressure data were measured proximally to the endotracheal tube at a sampling rate of 125 Hz. Patients were ventilated in the volume-controlled mode and at a constant inspiratory flow rate.

1.3.1 Subjects and Medication of ARDS Group

The multicenter study from which the data of the ARDS group were collected was carried out in intensive care units across eight German university hospitals. See Table 1.2 for details.

Patients

Patients suffering from pulmonary ARDS (n = 5) or extra-pulmonary ARDS (n = 15) were included in the study. The patients had to be mechanically ventilated for 24 hours or longer before entering the study. Exclusion criteria were: patients considered ready to be weaned by the attending physician; in the terminal stage of disease; the presence of an obstructive lung disease, a bronchopleural fistula or known air leakage; hemodynamic instability or intolerance to a five minute ZEEP phase; age below 16 years; or pregnancy.

Medication

Neuromuscular blocking drugs were applied as required. Sedatives were administered to achieve a Ramsay sedation score of 4 to 5.

Table 1.2: Characteristics of ARDS group.

No.	*Age (years)*	*Sex (m/f)*	*Height (cm)*	*Weight (kg)*	*Primary Diagnosis*	*ARDS (p/ep)*
1	51	m	180	125	pancreatitis	ep
2	38	m	180	100	severe thorax trauma	p
3	50	f	168	65	pancreatitis	ep
4	66	m	175	85	peritonitis	ep
5	30	f	160	61	peritonitis	ep
6	49	f	160	100	pneumonia	p
*7	34	m	185	90	traumatic open brain injury	ep
8	67	m	183	104	postresuscitation after heart failure	ep
9	39	m	170	60	peritonitis	ep
10	62	m	165	70	subarachnoid hemorrhage	ep
11	60	m	175	85	peritonitis	ep
12	50	m	175	85	traumatic brain injury	ep
13	45	m	175	80	carcinoma of the floor of the mouth	ep
*14	77	m	180	95	pneumonia	p
15	38	m	175	75	traumatic brain injury	ep
16	73	m	154	63	pneumonia	p
17	59	m	170	66	abdominal aortic aneurysm	ep
18	37	m	183	90	pancreatitis	ep
19	45	m	178	90	pneumonia after blunt abdominal trauma	p
20	42	m	175	70	liver cirrhosis	ep
mean	50.6		168.6	87.7		
SD	13.5		20.1	28.5		

m: male; *f*: female; *p*: pulmonary ARDS; *ep*: extra-pulmonary ARDS; *SD*: standard deviation; *: not used in experiments based on super syringe maneuvers in chapter 4.

Ventilation

Patients were ventilated in a volume-controlled mode with a constant inspiratory flow rate in the supine position. The tidal volume was targeted at $8.0 \pm 2.0\,\mathrm{mL/kg}$. Inspiratory time and flow rate were set to obtain an EIP of 0.2 seconds or longer. Before the measurements, respiratory rate was adjusted to keep the partial pressure of arterial carbon dioxide below 55.0 mmHg. Between respiratory maneuvers, the fraction of inspired oxygen (FiO_2) was chosen to maintain arterial oxygen saturation above 90%.

Respiratory Maneuvers

During the protocol, ventilator settings remained unchanged. During respiratory maneuvers, the FiO_2 was set to 1.0. Five different maneuvers (low-flow inflation [49], incremental

positive end-expiratory pressure trial (PEEP wave [88]), enlarged tidal volume breath for dynamic pressure-volume analysis (SLICE method [45]), static compliance by automated single steps [107] and super syringe [69]) were performed in random sequence. To obtain standard volume history, patients were ventilated with ZEEP for five minutes before each maneuver. Pressure, flow and volume time series data were recorded. Pressure and flow data were measured proximally to the endotracheal tube at a sample rate of 125 Hz. The applied volume was calculated by integration of the flow data.

Super syringe maneuvers: In the ARDS group automatized super syringe maneuvers (section 1.2.5) were performed completely for 20 patients. During a single maneuver, the ventilatory system repetitively applied volume steps of 100 mL with constant inspiratory airflow rates up to a maximum plateau pressure of 45 mbar. At the end of each volume step, airflow was interrupted for 3 seconds (Figure 2.1). Data obtained from super syringe maneuvers conducted in the ARDS group were used in chapters 2 and 4.

1.3.2 Subjects and Medication of Control Group

Data were measured under conditions of preoperative anesthesia for orthopedic surgery at the University Hospital of Freiburg. See Table 1.3 for details.

Patients

Patients in American Society of Anesthesiologists (ASA) physical status I and II undergoing general anesthesia and tracheal intubation were included in the study. Exclusion criteria were: patients with indications of lung disease; age below 18 years; as electrical impedance tomography was also performed in these patients (data not used in the present studies), the presence of any condition precluding the implementation of electrical impedance tomography such as a pacemaker, an implanted automatic cardioverter defibrillator, implantable pumps, pregnancy, lactation period, or iontophoresis.

Medication

Anesthesia was induced with fentanyl and propofol. Propofol was applied continuously to maintain anesthesia. Vecuronium bromide was applied for neuromuscular blocking.

Ventilation

Patients were ventilated in the volume-controlled mode (applied volume: 10 mL/kg, respiratory rate: 12 breaths/minute, inspiratory to expiratory ratio: 1:1.5, FiO_2: 1, PEEP:

Table 1.3: Characteristics of control group.

No.	*Age (years)*	*Sex (m/f)*	*Height (cm)*	*Weight (kg)*	*Primary Diagnosis*
1	56	f	158	64	lesion of the right anterior meniscus
2	19	m	190	83	rupture of right anterior cruciate ligament
3	22	f	176	68	state after fracture of left upper arm
4	19	m	195	85	bimalleolar ancle joint fracture
5	25	m	185	85	compartment syndrome left lower leg
6	46	m	185	87	cartilage damage medial condyle of femur
7	35	m	180	89	fracture of left lateral tibial plateau
8	36	f	170	72	state after plating of fractured olecranon
9	25	m	180	77	bilateral fracture of lower leg, fractured left ancle joint
10	27	m	179	70	4-part fracture of head of humerus
11	36	f	164	63	lesion of ventral capsule-labrum-complex of right shoulder
*12	44	m	178	77	open fracture of the left lower leg
*13	33	m	164	60	tibial plateau fracture
mean	32.5		177.2	75.4	
SD	11.2		10.8	10.0	

m: male; *f*: female; *SD*: standard deviation; *: not used in experiments based on super syringe maneuvers in chapter 2.

0 mbar) while in the supine position. To prevent potential atelectasis, a recruitment maneuver was performed by increasing PEEP up to a plateau pressure of 45 mbar. Ventilation at the corresponding PEEP was maintained for six breaths and then reduced to ZEEP.

Respiratory Maneuvers

During the protocol, ventilator settings remained unchanged. An incremental PEEP trial [88] followed by a super syringe maneuver [69] was performed. To standardize volume history, both maneuvers were preceded by ventilation with ZEEP for five minutes. Pressure, flow and volume time series data were recorded. Pressure and flow data were measured proximally to the endotracheal tube at a sample rate of 125 Hz. The applied volume was calculated by integration of the flow data.

Super syringe maneuvers: Automatized super syringe maneuvers (section 1.2.5) were performed for 13 patients of the control group. The maneuvers were performed in the same standardized mode as for the ARDS group: during a single maneuver, the ventilatory system repetitively applied volume steps of 100 mL with constant inspiratory airflow rates up to a maximum plateau pressure of 45 mbar. At the end of each volume step, airflow was interrupted for 3 seconds (Figure 1.6). Data obtained from super syringe maneuvers

conducted in the control group were used in chapter 2.

PEEP wave maneuvers: Automated PEEP wave maneuvers (section 1.2.5) were performed for 13 patients of the control group. During such maneuvers, continuous mechanical ventilation was applied at successively increasing PEEP levels. Starting from 0 mbar, PEEP was increased in steps of 2 mbar up to a maximum plateau pressure of 45 mbar (Figure 1.7). Data obtained from PEEP wave maneuvers conducted in the control group were used in chapter 3.

1.4 Aim of Work

For three different tasks, techniques from classical biomedical engineering, from KDD and ML are used. Thus, within three different studies, the analysis of mechanical lung parameters and their nonlinear behavior, a method for the semiautomatic detection of lung models as basis for refined analysis techniques and a methodological approach to predict the lung compliance as alternative of respiratory maneuvers are presented.

1.4.1 Analysis of Stress Relaxation Processes

During mechanical ventilation, energy is transferred from the ventilator to the patient's respiratory system. As in volutrauma and barotrauma, the amount of transferred energy is directly related to VALI. However, volutrauma and barotrauma are both restricted to the particular physical quantities volume and pressure. Other parameters also directly influencing the transferred energy as the respiratory rate [50] are disregarded in these concepts.

One part of the transferred energy is required to overcome respiratory system resistance and compliance, another part is stored or dissipates in the viscoelastic components of the respiratory system while following the respiratory cycle.

One could subsume all those different factors under an energy-related concept of lung injury. Hence, minimizing this "energotrauma" would be equivalent to the minimization of energy transfer by simultaneously adapting pressure, volume and frequency. This could be helpful in the development of lung protective ventilation strategies. In this thesis, nonlinear effects of viscoelastic tissue properties are investigated with respect to lung protective ventilation [38] (chapter 2).

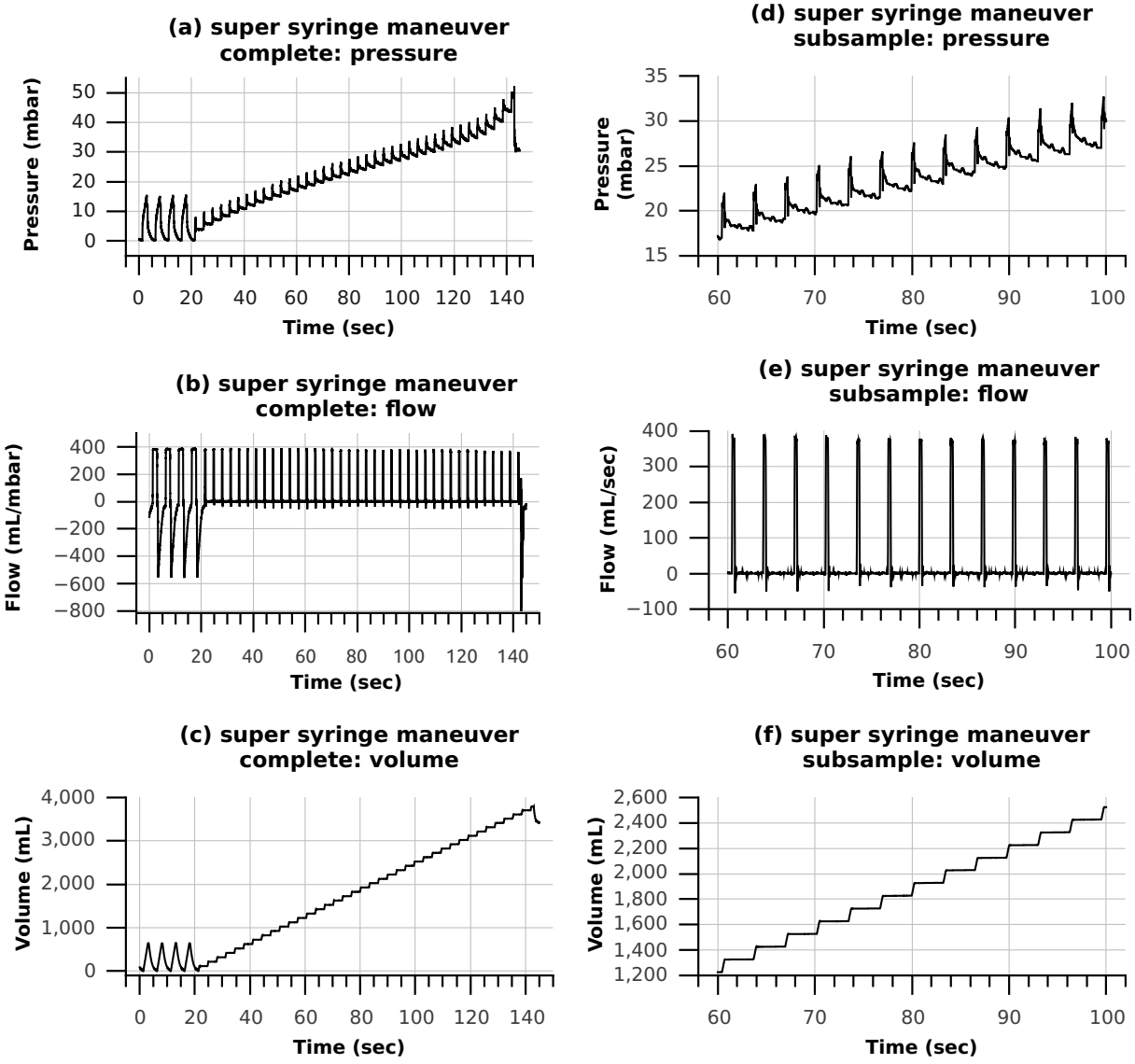

Figure 1.6: Sample of super syringe maneuver (Figure 1.4) recorded in the control group. The recordings were obtained from the same patient as the PEEP wave maneuver sample data in Figure 1.7. Complete recording of the (a) pressure, (b) flow and (c) volume time series. The maneuver consisted of 37 volume steps of 100 mL which were applied at an airflow rate of 400 mL/sec. The maneuver started after a regular inflation at an airflow rate of 400 mL/sec with an applied volume of 600 mL. Subsamples of pressure (d), flow (e) and volume (f) time series. After each inflation the airflow was interrupted for 3 seconds (Figure 1.2). The pressure time series in subgraphics (a) can be roughly split in 3 parts: in its lower and the upper third the curve shows a steeper slope than in the middle third. As in each inflation step the volume is increased for the same amount of 100 mL, this indicates that at low and high levels of the lung capacity the compliance $C = \Delta V/\Delta P$ (Equation (1.1)) is lower than in intermediate inflation levels of the lung (Figure 1.1).

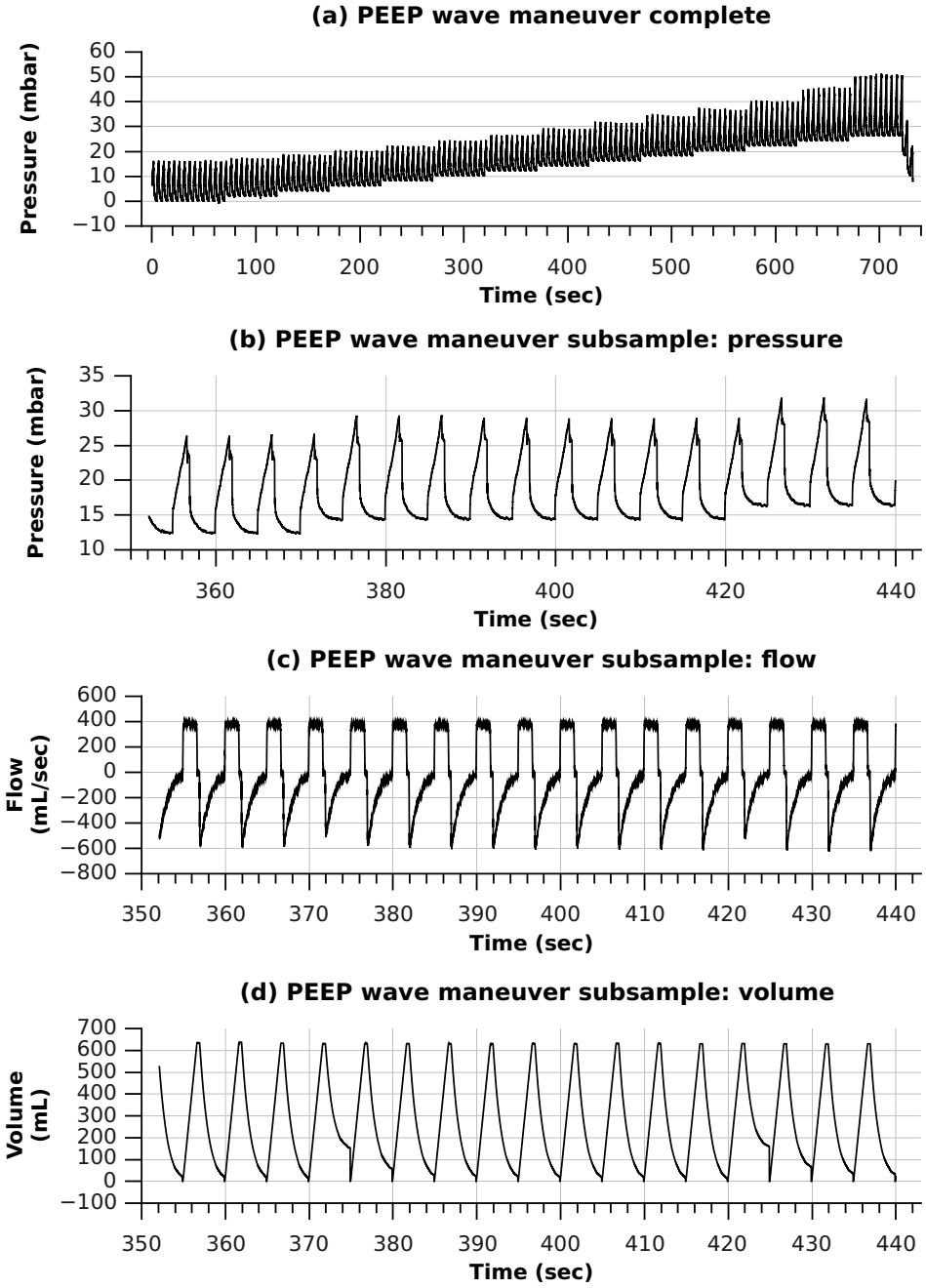

Figure 1.7: Sample of PEEP wave maneuver recorded in the control group. The recordings were obtained from the same patient as the super syringe maneuver sample data in Figure 1.6. (a) Complete recording of the pressure time series. One can clearly distinguish the 14 different PEEP levels. (b) Subsample of pressure time series. At second 370 and 420 the PEEP level was increased. In between these time points, the PEEP level was set from 12 to 14 mbar. (c) Subsample of flow time series. The inspiratory airflow rate was set to 400 mL/sec (over the complete maneuver run). (d) Subsample of volume time series. The inspiratory volume was set to 600 mL (over the complete maneuver run). At second 375 and 425, after an incomplete exhalation succeeding a PEEP level increase, one can clearly identify the volume reset to 0 mL (Figure 1.5).

1.4.2 Equation Discovery for Model Detection

The analysis of the respiratory mechanics under mechanical ventilation is generally based on mathematical models to describe and interpret the associated mechanisms [3, 6, 11, 33, 48]. The equation of motion [46] as commonly accepted mathematical model of the respiratory system provides the basis for most clinically applied methods of respiratory mechanics analysis [66, 110]. However, this model has restrictions since it does not include aspects of nonlinearity or inhomogeneity of the human lung. Consequently, refined methods and models are required to analyze respiratory mechanics. Especially monitoring techniques under the dynamic condition of continuous mechanical ventilation at bedside are of interest [42, 74]. In this thesis, an equation discovery approach as basis for the semiautomatic detection of refined lung models is presented [37, 39] (chapter 3).

1.4.3 Gaussian Process Modeling for Compliance Prediction

Mechanical interactions and time-dependent properties of the respiratory system have been content of medical research for a long time. Actually, the modeling of the respiratory system generally covers particular properties like lung recruitment [4, 48] and viscoelastic effects [12, 53]. However, especially nonlinear effects are hardly taken into account. Besides the lack of appropriate models, it is a difficult task to choose adequate boundary conditions for model based parameter estimation. Therefore, under mechanical ventilation, a reliable prediction of either the parameters describing the boundary conditions or the queried meaningful parameters itself could help to estimate changes in the mechanical behavior of the lung without the necessity of possibly harmful maneuvers. In this thesis, the lung compliance as essential indicator for an optimized lung protective ventilation is predicted based on Gaussian processes [36] (chapter 4).

Summarizing, the overall aim of this thesis is the analysis of respiratory mechanics under the condition of mechanical ventilation. The data consists of pressure, flow and volume time series data recorded from patients suffering from ARDS (ARDS group) and patients without lung damage (control group). The analyses where performed with respect to lung protective ventilation strategies. A variety of techniques including classical biomedical engineering, equation discovery, machine learning and knowledge discovery in databases should be applied in order to allow different degrees of abstraction.

Chapter 2

Analysis of Stress Relaxation Processes in the Human Lung

Limiting the energy transfer between ventilator and lung is crucial for the ventilatory strategy in ARDS. Part of the energy is transmitted to the viscoelastic tissue components where it is stored or dissipates. In mechanically ventilated patients, viscoelasticity can be investigated by analyzing pulmonary stress relaxation. While stress relaxation processes of the lung have been intensively investigated, nonlinear interrelations have not been systematically analyzed, and such analyses have been limited to small volume or pressure ranges (Table 1.1).

Fung introduced the concept of quasi-linear viscoelasticity [33]. Accordingly, the stress relaxation dynamics quantified by the viscoelastic time constant τ_{ve} (Equation (1.4)) is independent of the strain of the lung parenchyma, which corresponds to intropulmonary volume. On the other hand, the quasi-static stress-strain relationship is nonlinear [3, 33, 34]. We hypothesized that according to the quasi-static viscoelastic concept, the stress relaxation process quantified by the viscoelastic time constant is independent of pressure. In addition we hypothesized that – other than the viscoelastic time constant – the viscoelastic resistance and the viscoelastic compliance depend on pulmonary pressure.

The purpose of the present study is to investigate nonlinear pressure-dependent viscoelastic properties of the respiratory system with focus on differences in energy distribution between healthy and ARDS lungs. The total range of inspiratory capacity is analyzed up to a plateau pressure of 45 mbar. The analysis includes both the processes of dynamic loading and static stress relaxation of the tissue. For data acquisition, standardized super syringe maneuvers are automatically performed. Data analysis is based on a viscoelastic lumped parameter model and compared to classical two-point analysis techniques. Frequency related characteristics are investigated by impedance analysis.

2.1 Materials and Methods

2.1.1 Datasets

Data were obtained from standardized super syringe maneuvers [69] (section 1.2.5) applied to patients suffering from ARDS (ARDS group, section 1.3.1) and patients without lung-injury (control group, section 1.3.2). During the automatically operated maneuvers, the ventilator repetitively applied volume steps of 100 mL, with an inspiratory airflow rate of 558 ± 93 mL/sec for the ARDS group and 470 ± 95 mL/sec for the control group up to a maximum plateau pressure of 45 mbar. At the end of each volume application, airflow was interrupted for three seconds. Figure 2.1 depicts a representative pressure time series for a super syringe maneuver obtained from one patient of the ARDS group and one patient of the control group. For each group, the curves show typical progressions indicating the different health status of the lung.

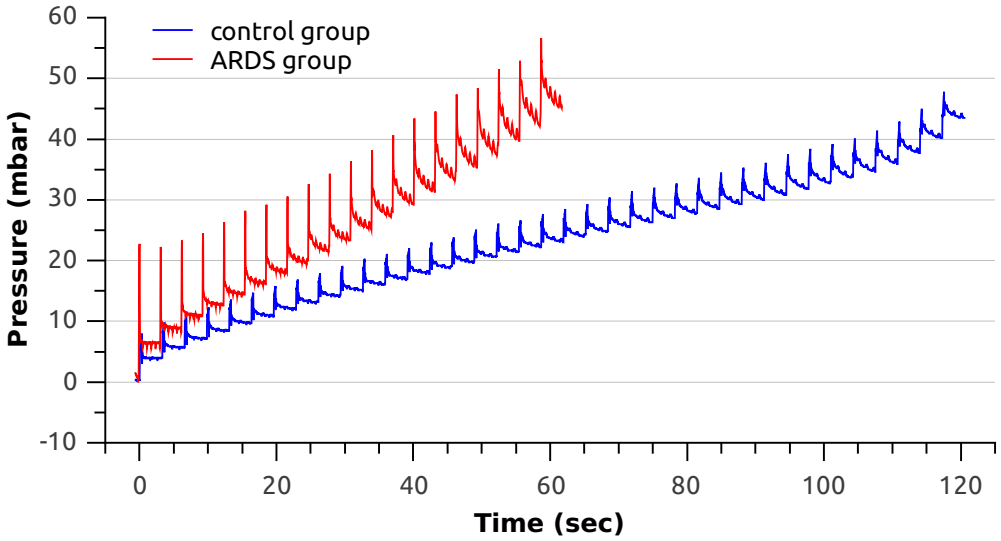

Figure 2.1: Super syringe maneuver for ARDS and control group. Representative time series for standardized super syringe maneuvers obtained from one patient suffering from the acute respiratory distress syndrome (ARDS group) and one patient with healthy lungs (control group). Volume steps of 100 mL were repetitively applied up to a maximum plateau pressure of 45 mbar. After each volume step, airflow was interrupted for three seconds. Note that in ARDS, the maximum plateau pressure is reached after approximately only half of the volume steps applied to the healthy lung and that at the beginning of each inspiratory step the pressure peaks are more pronounced. In the literature it is discussed whether this is due to a reduced lung volume and/or to stiffer lung tissue in ARDS compared to healthy lungs [41].

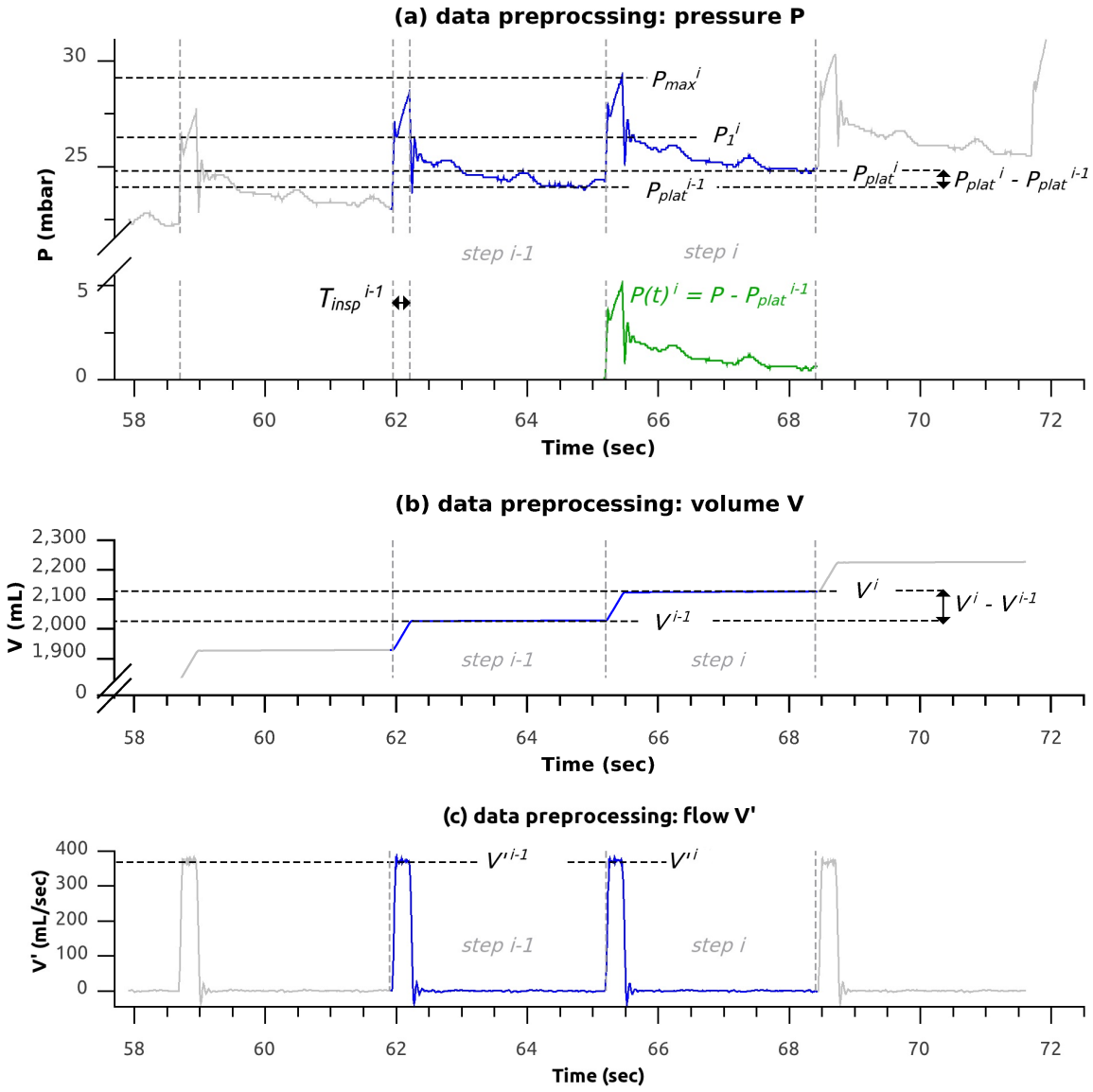

Figure 2.2: Data preprocessing for two-point and multiple-point analysis. *Two-point analysis*: for each volume inflation step i, the static compliance C^i (Equation (2.3)), the corresponding Newtonian airway resistance R^i (Equation (2.4)), the effective additional resistance ΔR^i_{add} (Equation (2.5)) which is used to calculate the resistance related to viscoelastic effects R^i_{ve} (Equation (2.6)) and finally the corresponding compliance C^i_{ve} (Equation (2.7)) was calculated. Note that in the data records each data point was marked as inspiratory or expiratory and therefore the inspiratory time T^i_{insp} as used in Equation (2.6) was provided most precisely by the respirator. (a) Stepwise pressure calculations related to Equations (2.3), (2.4) and (2.5). (b) Stepwise volume calculations related to Equation (2.3). (c) Corresponding flow values used in Equations (2.4) and (2.5). *Multiple-point analysis*: (a) For each volume inflation step i, the pressure P^i was corrected by an offset determined by the plateau pressure of the previous step *i-1* (Equation (2.1)). (c) For the flow as second input for the spring-and-dashpot model, no preprocessing had to be made.

2.1.2 Data Preprocessing

To prevent data analysis from being affected by the pressure oscillations caused by the closure of the interruption valve, the referring data interval was excluded from the data fitting (Figure 1.2). The time period corresponding to the valve-induced pressure oscillations was defined as the interval between the onset of flow interruption and the end of the successive first positive slope of the flow curve (Figure 1.2.a). It was assumed that the stress relaxation was completed after a 3 sec occlusion interval, and that by then, pressure had reached the plateau pressure level P_{plat} and the system was stabilized by this time. Thus, the respiratory pressure of the i-th volume step P^i was adjusted with an offset determined by the plateau pressure P_{plat}^{i-1} of the preceding maneuver volume step i-1 according to Equation (2.1) (Figure 2.2).

$$P^i(t) = P - P_{plat}^{i-1} \tag{2.1}$$

P_{plat} was calculated as the mean of the pressure values obtained from a 0.5 seconds interval at the end of occlusion. For step i = 1, P_{plat}^0 was set to the mean of the last 0.5 seconds of the preceding expiration at ZEEP.

2.1.3 Data Analysis

All analyses and model simulations were carried out using the Matlab® software package Version R2006b (The MathWorks®, Natick, MA, USA). For details about two-point and multiple-point methods, see section 1.2.4 also.

Estimation: Parameter Extraction via Multiple-Point Method

We used the electrical analog of a spring-and-dashpot model as described in section 1.2.4 consisting of two components: (1) A Newtonian airway resistance R and a (static) compliance of the respiratory system C and (2) the electrical analog of a resistive dashpot R_{ve} and an elastic spring C_{ve} as resistance and compliance of the component which is modeling viscoelastic behavior. The viscoelastic time constant τ_{ve} of the viscoelastic component quantifies the stress relaxation dynamics of the system and is determined by the product of the viscoelastic resistance R_{ve} and the viscoelastic compliance C_{ve} (Equation (1.4)).

For each volume step i within each super syringe maneuver, the parameters R^i, C^i, R_{ve}^i and C_{ve}^i, were estimated by fitting the model via a multiple regression analysis to the time series data (Figure 1.2). Representation of the electrical circuit (Figure 1.3) in terms of a Laplace transform leads to Equation (2.2). (See Appendix A for details.)

$$\frac{P(s)}{\dot{V}(s)} = \frac{RCR_{ve}C_{ve}s^2 + (RC + CR_{ve} + R_{ve}C_{ve})s + 1}{CR_{ve}C_{ve}s^2 + Cs} \tag{2.2}$$

The pressure of the respiratory system P is represented by the numerator and the respiratory flow $\dot{V}$ by the denominator. The sum of squared errors was used as the scalar function of R, C, R_{ve} and C_{ve} to be optimized. For optimization the simplex search method of Lagarias *et al.* [60] was used.

Validation: Parameter Estimation via Two-Point Method

To verify the plausibility of the parameters estimated by the multiple-point method, results were compared to the classical two-point method as described in [71]: for data from a single flow interruption, the static compliance is calculated as the volume change divided by the pressure change, which is measured as the end-inspiratory plateau pressure minus the pressure level at the end of expiration (Figure 3.3.b). As our measurings are based on a super syringe maneuver, the expiratory phase is omitted (Figure 2.2). Therefore, each stepwise pressure increase is based on the plateau pressure at the end of the previous step and the static compliance C at volume step i was calculated as

$$C^i = \frac{V^i - V^{i-1}}{P^i_{plat} - P^{i-1}_{plat}} \tag{2.3}$$

where V^i represents the accumulated volume at step i (Figure 2.2.a,b). Note that P^i_{plat} represents the inspiratory plateau pressure and P^{i-1}_{plat} substitutes the end-expiratory plateau pressure (see above). The corresponding Newtonian airway resistance was estimated as (Figures 1.2, 2.2.a,c)

$$R^i = \frac{P^i_{max} - P^i_1}{\dot{V}^i} \tag{2.4}$$

The pressure value at start of the valve closure P^i_1 and the relaxation time τ^i_{ve} were approximated by backward extrapolation of an exponential data fit of the pressure interval during flow interruption (Figure 1.2.b). The inspiratory time T^i_{insp} at step i was extracted from each interval of volume inflation (Figure 2.2.a,c). The effective additional resistance ΔR^i_{add} was calculated as

$$\Delta R^i_{add} = \frac{P^i_1 - P^i_{plat}}{\dot{V}^i} \tag{2.5}$$

and thus

$$R^i_{ve} = \frac{\Delta R^i_{add}}{1 - e^{-T^i_{insp}/\tau^i_{ve}}}. \tag{2.6}$$

Finally, the viscoelastic compliance C^i_{ve} was estimated as

$$C^i_{ve} = \frac{\tau^i_{ve}}{R^i_{ve}}. \tag{2.7}$$

Impedance Analysis

Impedance analysis was performed with respect to dependence on respiratory frequency for four categories: ARDS group at low (7.5 mbar) and high (42.5 mbar) plateau pressure, and control group at the same low and high plateau pressure. For each category, the parameters R, C, R_{ve} and C_{ve} were determined and inserted into the model. For each parameterized model, a Bode magnitude plot was drawn.

Data Presentation and Statistical Evaluation

For data presentation, the estimated values of the model parameters were linearly interpolated in steps of 2.5 mbar within a pressure range between 7.5 and 42.5 mbar. For each resulting pressure level, interpolated parameter values beyond the 1.5 fold of the interquartile range were eliminated as outliers. Normal distribution of the determined parameter values could not be proved. Therefore, statistical evaluation was based on the Wilcoxon rank-sum test. The significance level was set to $P \leq 0.05$. Data are presented as median (lower to upper quartile), unless otherwise indicated.

2.2 Results

The super syringe maneuvers consisted of 5 to 38 occlusions in the ARDS group, and 37 to 39 occlusions in the control group. The total inflated volumes were 1965 ± 929 mL for the ARDS group, and 4064 ± 67 mL for the control group (Figure 2.1).

Viscoelastic compliance, as well as viscoelastic resistance, depended on plateau pressure, and they differed between the control and the ARDS group. Viscoelastic resistance (Figure 2.3.a) increased with pressure for both the control and the ARDS group (control: 8.4 (7.4 to 11.9) up to 35.2 (25.6 to 39.5) mbar $\cdot$ sec/L; ARDS: 11.9 (9.2 to 22.1) up to 73.5 (56.8 to 98.7) mbar $\cdot$ sec/L). In contrast, viscoelastic compliance (Figure 2.3.b) decreased with pressure for both groups (control: 130.1 (116.9 to 151.3) down to 37.4 (34.7 to 46.3) mL/mbar; ARDS: 125.8 (80.0 to 211.0) down to 17.1 (13.8 to 24.7) mL/mbar). Both

interrelations presented a nonlinear progression. At plateau pressures below 17.5 mbar, R_{ve} remained almost constant with no significant differences between the control (10.1 (8.0 to 13.2) mbar · sec/L) and the ARDS group (12.8 (9.9 to 22.0) mbar · sec/L). At plateau pressures of 17.5 mbar and above, statistically significant differences were observed and increased with plateau pressure (control: 15.6 (10.7 to 26.6) mbar · sec/L; ARDS: 34.7 (22.1 to 48.0) mbar · sec/L). In the ARDS group, the overall viscoelastic resistance was significantly larger (ARDS: 28.2 (15.4 to 42.9) mbar · sec/L; control: 13.2 (9.4 to 23.2) mbar · sec/L), and viscoelastic compliance was significantly smaller (ARDS: 41.4 (27.5 to 62.8) mL/mbar; control: 88.0 (64.0 to 111.8) mL/mbar) than in the control group. In contrast, the viscoelastic time constant (Figure 2.3.c) remained almost unchanged and did not significantly differ between groups (ARDS: 1.07 (0.88 to 1.31) seconds, control: 1.20 (0.92 to 1.58) seconds).

With increasing respiratory frequency, the impedance of the respiratory system converged to a small value (Figure 2.4). At frequencies between 5 and 20 breaths/min, the respiratory system exhibited smaller impedances in control subjects compared with ARDS patients (Figure 2.4, insert). High plateau pressures induced higher impedance values than low pressures.

2.3 Discussion

2.3.1 Main Findings

The main results of this study are:

(i) The viscoelastic resistance R_{ve} and the viscoelastic compliance C_{ve} depended nonlinearly on increasing plateau pressure. In both groups, R_{ve} increased and C_{ve} decreased with increasing pressure, but these changes were different in ARDS and normal lungs.

(ii) Stress relaxation dynamics represented by the viscoelastic time constant τ_{ve} were independent of pressure and disease state (healthy vs. ARDS).

(iii) The pulmonary mechanical impedance increased with plateau pressure and decreased with respiratory frequency.

2.3.2 Mechanical Properties

During each inflation step of a super syringe maneuver, mechanical stress is applied to the lungs and part of the applied energy is loaded to the viscoelastic lung tissue components represented by the viscoelastic compliance, while part of this energy dissipates via the

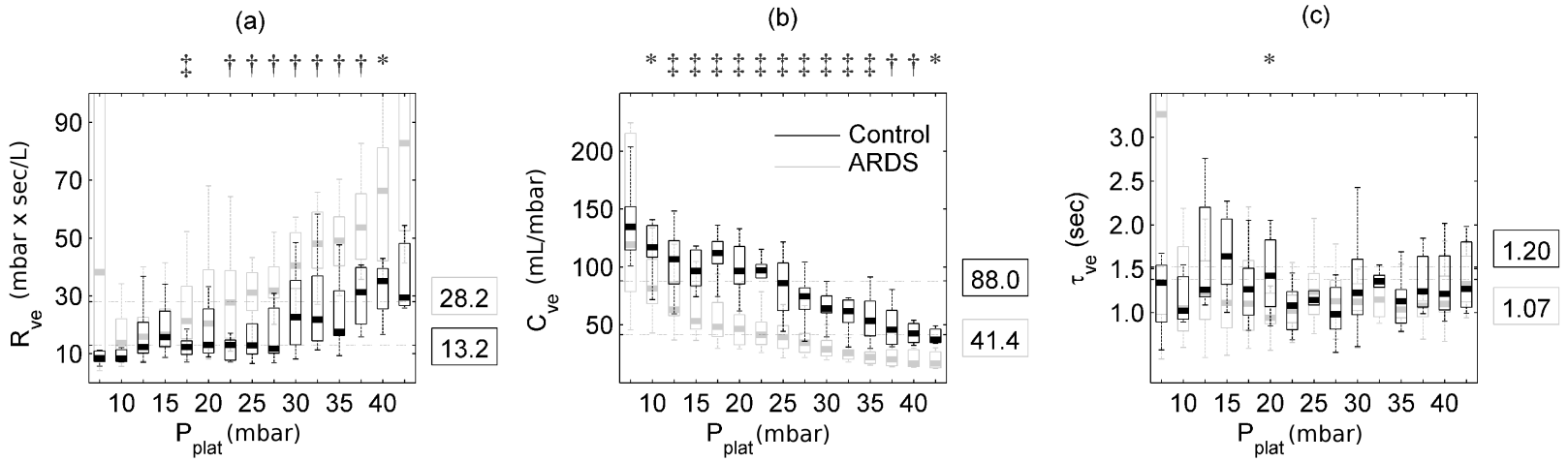

Figure 2.3: Results of parameter estimation. Estimated parameters of viscoelasticity for the ARDS group and the control group in terms of lower quartiles, medians and upper quartiles plotted against plateau pressure (P_{plat}). The estimated values of the model parameters were linearly interpolated in steps of 2.5 mbar within a pressure range between 7.5 and 42.5 mbar. For each resulting pressure level, interpolated parameter values beyond the 1.5 fold of the interquartile range were eliminated as outliers. Values on the right side of the diagrams indicate the overall medians. Statistically significant levels are indicated by $*P \leq 0.05$, $\dagger P \leq 0.01$ and $\ddagger P \leq 0.001$. (a) Viscoelastic resistance R_{ve} as well as (b) viscoelastic compliance C_{ve} differ significantly between both groups. For both parameters, a notably nonlinear progression with increasing pressure was observed. (c) The viscoelastic time constant τ_{ve} of the viscoelastic model component does not differ between the two patient groups, and it does not depend on P_{plat}.

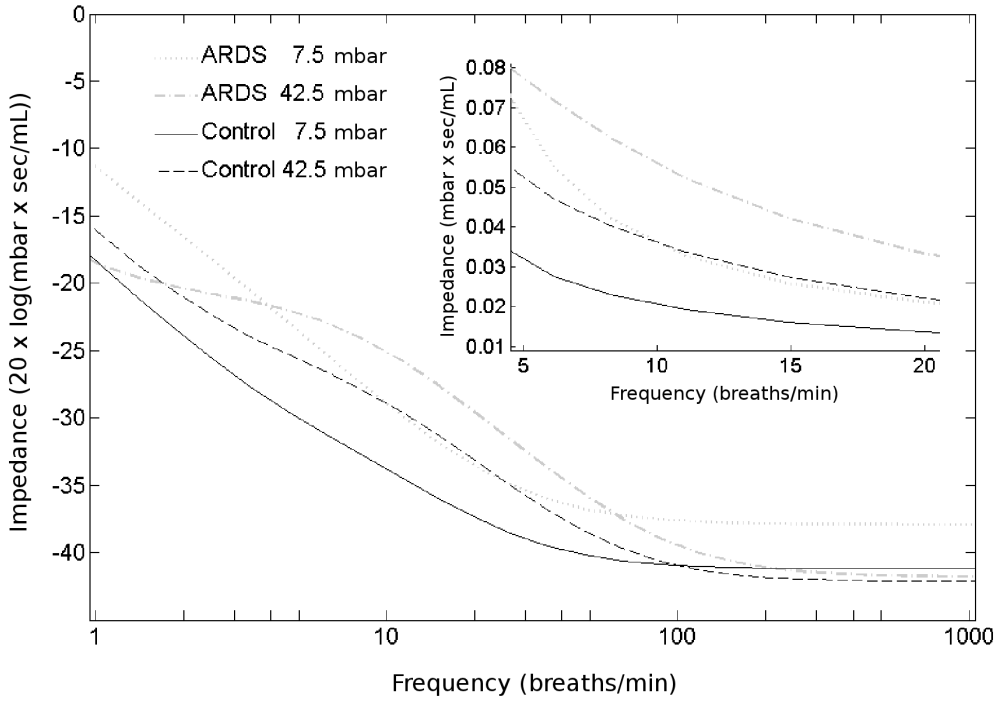

Figure 2.4: Frequency analysis. Frequency dependence of the respiratory systems mechanical impedance. The four curves were extracted from the magnitude diagram of a Bode plot, which was obtained from the Laplace transform representing the electrical circuit model. The curves represent the impedance of the model for plateau pressures of 7.5 mbar (low) and 42.5 mbar (high) for both patient groups. Note that the y-axis of the diagram is scaled by $20 \cdot log$, i.e. dB, while the insert is linearly scaled.

viscoelastic resistance. In the subsequent zero flow phase the pulmonary tissue elements approximate a relaxation state at the new plateau pressure level. Therefore, each new inflation step starts from an increased baseline strain, which is quantified by the corresponding plateau pressure. Furthermore, each step starts from a particular relaxation state of the viscoelastic elements. Based on the fact that the pressure increase per 100 mL step of volume inflation was larger in ARDS, the super syringe maneuver data revealed a discrepancy between both groups concerning the number of volume steps until the maximum pressure level was reached (Figure 2.1). Despite these considerably different pressure-volume relations, the time constant of viscoelasticity was independent of both, the pulmonary plateau pressure and also the disease state.

Fung's [33] concept of quasi-linear viscoelasticity may provide a theoretical explanation: the experimental results showed that the quasi-static stress-strain relation is non-

linear [33, 3]. On the other hand, stress relaxation dynamics are independent of strain. Implying quasi-linear viscoelasticity, Ingenito and colleagues [52] analyzed parenchymal tissue strips obtained from guinea pigs. They stated that in acute lung injury, changes in the elastic and dissipative properties of lung parenchyma can occur. Bates [3] transferred Fung's general mathematical concept to lung tissue mechanics by proposing a refined spring-and-dashpot model. This model is able to predict the stress relaxation power law in a strain independent manner using a sequential recruitment of Maxwell bodies. The validation of the model was based on experiments with tissue strips taken from canine lung parenchyma [10]. The particular arrangement and interaction of the spring-and-dashpot elements of this model are well suited to describe viscoelastic tissue properties. Although using a basic lumped parameter model, our findings are consistent with Bates' observation [3] that his modeling approach exhibits quasi-linear viscoelastic behavior, in both qualitative and quantitative agreement with experimental data. Hence, the concept of quasi-linear viscoelasticity seems to apply to the human lung under mechanical ventilation: because the viscoelastic time constant τ_{ve} was independent of the plateau pressure, stress relaxation was similar for all pressure levels. Referring to the nonlinearity of the quasi-static stress-strain relation, C_{ve} and R_{ve} showed distinct nonlinear dependencies on increasing plateau pressure (Figure 2.3). Compared with the normal lung, the increase in R_{ve} in ARDS seemed to start at lower plateau pressures, and it had a steeper slope for pressures above 17.5 mbar. These findings are in accordance with previous studies [84, 86] investigating the effect of PEEP on respiratory resistance. These studies found that the resistance is abnormally elevated in ARDS and that it increases with PEEP, particularly at 10 mbar and higher. This was assumed to be caused by stress adaptation phenomena and/or to be due to time constant inhomogeneities. Investigating data from patients with normal lungs, D'Angelo and colleagues [22] stated that the viscoelastic behavior of the lung is independent of volume and found no significant differences between ZEEP and PEEP for the viscoelastic parameters. Investigating data from ARDS patients, Eissa and colleagues [29] indeed found an increase of R_{ve} and an increase of the viscoelastic elastance E_{ve} (i.e. a decrease of $C_{ve} = 1/E_{ve}$) with an increasing PEEP level albeit a statistical significance could hardly be shown. The latter might stem from the rather small number of nine investigated subjects.

Concerning the pressure dependence of viscoelasticity, two situations can be distinguished: (i) in low pressure ranges, C_{ve} is large and R_{ve} is small; (ii) in high pressure ranges, the situation is reversed, with small C_{ve} and large R_{ve}. Therefore, in low pressure ranges, the viscoelastic compartment is characterized by a large loading capacity C_{ve} for viscoelastic energy, which can easily dissipate via a small R_{ve}. In contrast, at high pressure ranges, the situation is characterized by a small loading capacity C_{ve}, and an impaired energy dissipation caused by a large R_{ve}. During mechanical ventilation, low plateau pressure is therefore associated with small resistance imposed by the viscoelastic element, whereas at high plateau pressure, ventilation is impaired due to a large resistance generated by the viscoelasticity. Our results indicate that at low plateau pressures, that is below 20 mbar, viscoelastic resistance is not affected, whereas at high plateau pressures,

it is. Therefore, within the context of the viscoelastic properties of lung tissue, ARDS patients might benefit from low alveolar pressure.

The difference between low plateau pressure (small R_{ve}, large C_{ve}) and high plateau pressure (large R_{ve}, small C_{ve}), as well as between ARDS and normal lungs, is responsible for the different sensitivity to respiratory frequency (Figure 2.4). The impedance values in the ARDS lungs at high plateau pressures exceeded impedance values in the normal lung at low plateau pressures by up to 270%. At higher frequencies, e.g. 60 to 900 breaths/min, as used for high frequency oscillatory ventilation [18], the curves converged towards an identical small value. Hotchkiss and colleagues [50] observed in isolated rabbit lungs that ventilation at low respiratory frequencies caused less edema formation and histologic alterations than ventilation at high frequencies and identical tidal volume, airway plateau pressure, PEEP and peak pulmonary artery pressure. Interpreting these results from a mechanico-energetical point of view as underlying the present study, Hotchkiss and colleagues provided evidence that the amount of energy transfer indeed seems to be crucial for the induction of lung damage under mechanical ventilation: physically, the amount of mechanical energy is equivalent to the amount of mechanical work. This again is defined by the product of (volume-dependent) pressure and volume change. By keeping the applied pressure level and tidal volume constant, the transferred energy (energy per time) increases with increasing frequency, because over time the energy multiplies with the respiratory rate.

Thus, with respect to a clinical interpretation, from the mechano-energetical point of view our results might show evidence that ARDS patients would benefit from low alveolar pressures and high frequencies combined with reduced tidal volumes. If tidal volume is reduced under preservation of minute ventilation – in the context of lung protective ventilation – the same ventilatory effect can be achieved with a smaller energy transfer by a reduction of the frequency dependent impedance.

2.3.3 Validity of Method

For the present study, the data collection had to satisfy three main prerequisites: (1) to investigate stress relaxation dynamics, rapid flow interruptions were required; (2) the measured pressure-volume range had to be as wide as possible; and (3) a compromise between tolerable maneuver duration and desired high pressure-volume resolution had to be achieved. An appropriate measurement technique fulfilling these requirements was a standardized super syringe maneuver [69]. Specifically, with respect to each of the prerequisites this method implied the application of flow interruptions, it allowed for plateau pressures of up to 45 mbar in our experimental setting and achieved a compromise by application of small 100 mL volume steps. In addition, due to the degree of automation and standardization of the technique, the super syringe maneuvers were highly reproducible for all patients in both groups.

Pressure oscillations following the closure of the valve [99] are known to affect parameter estimation in the conventional two-point analysis [55, 71, 104]. Particularly the estimation of the point P_1 which marks the end of the fast pressure drop and the beginning of the slow pressure decrease is aggravated (Figure 1.2). Therefore, we excluded data corresponding to that time interval from the fitting and included, instead, the loading interval during volume inflation for the multiple regression analysis (Figure 1.2, section 2.1.2). This improved parameter estimation compared with the analysis exclusively based on the stress relaxation data as performed with the two-point method. Specifically, the root mean squared error (RMSE) as rate for the deviation between measured and re-calculated pressure curves was reduced by 31% in the control group and by 55% in the ARDS group. (The re-calculated pressure curve was obtained by substituting C, R, C_{ve} and R_{ve} derived from both the two-point and the multiple-point method into the spring-and-dashpot model. See Appendix B for detailed information). A sensitivity analysis showed this approach to be very stable with respect to noise in the flow and pressure time series data.

In the literature [7, 54], the side effects of prolonged closure of the occlusion valve on parameter estimation have been discussed. Corrections such as that of the maximal pressure at the end of the inspiratory flow phase have been suggested. However, these limitations do not apply to our experimental settings because the closure time of the ventilator valve we used was extremely short (1 ms, according to the manufacturer specifications).

Edibam and colleagues [28] found that during continuous tidal ventilation nonlinear characteristics of the lung elastance depend on the flow pattern (pressure- vs. volume-control) and inspiratory to expiratory ratio. The differences in nonlinear behavior were supposed to be most likely caused by the viscoelastic behavior of the respiratory system. Although the applied volume-dependent single-compartment model was well capable of describing the contribution of a nonlinear elastance fraction to the total elastance of the respiratory system, it was not designed to quantify the potentially underlying viscoelastic effects. For this purpose a more expressive two-compartment model was applied in the present study. This model has been shown to adequately describe non-Newtonian behavior in normal lungs [53] and in ARDS at ZEEP [11]. In ARDS at PEEP, volume-dependent modeling of the viscoelastic compliance C_{ve} improved the accuracy of the model [11]. Therefore, instead of direct nonlinearity in the model itself, nonlinearity was approximated rather by a sequence of linear models parameterized on different pressure levels.

The question remains, if the data obtained with the static super syringe maneuver fits to respiratory mechanics during dynamic continuous tidal ventilation. In contrast to classical two-point analysis of static super syringe maneuvers, we did not restrict our multiple-point analysis to the equilibrated respiratory system (Figure 1.2.b, P_{plat}). Instead, we included the dynamic part during volume inflation to our model fit and focused on the dynamic stress relaxation response of the respiratory system during the zero flow phase. Furthermore, with appropriate parameter settings the viscoelastic model has been shown to be independent of the flow pattern [53]. Taken together, we conclude that the estimated

viscoelastic parameters are also valid during continuous tidal ventilation.

Gattinoni and Pesenti [41] showed that the ARDS lung is small rather than stiff and that at the same tidal volume, mechanical strain is larger in the small ARDS lung compared with the normal lung. Thus, given a small lung volume, even a low energy transfer may cause or aggravate ventilator-associated lung injury as the energy impacts on a smaller inner surface. Indeed, volumes applied within the super syringe maneuvers were considerably smaller in ARDS (Figure 2.1). To prevent this bias, the investigated mechanical parameters were interpreted in relation to respiratory pressure and not to lung volume.

2.3.4 Limitations

The effects of viscoelasticity and "Pendelluft" (volume equilibration between compartments of the inhomogeneous lung) are hard to distinguish. Therefore, although we used a model that has been proven to be appropriate to study viscoelastic behavior in the homogeneous, healthy lung [6], we are very careful in our interpretations, following the example of preceding studies in this field.

Due to the mechanical inhomogeneity of the ARDS lung there is a particular dilemma when healthy lungs are compared with ARDS lungs. Because of alveolar consolidation and atelectasis, inflation with constant volume steps of 100 mL is in all likelihood delivered to a smaller amount of lung tissue in ARDS than in normal controls. This aggravates the interpretation of our findings from a physiological point of view. However, this does not impact the clinical implications of limiting the energy transfer to the lung.

Low plateau pressure values were hard to observe in ARDS. This is likely to be due to the high opening pressures of the ARDS lung. In addition, in the control group, high pressure ranges were not frequently measured because this was prevented by a built-in safety feature in the device. Therefore, parameters could rarely be estimated for low (ARDS) and high (control) pressures, which resulted in high variances for the parameters estimated within these ranges.

2.4 Conclusions

In this study the viscoelastic resistance, the viscoelastic compliance and the viscoelastic time constant were investigated on data covering the whole range of inspiratory capacity. Nonlinear pressure-dependencies of the lung viscoelasticity differ between patients with healthy and ARDS lungs. In contrast, the time constants of stress relaxation processes are independent of pressure and respiratory disease. These findings confirm Fung's concept of quasi-linear viscoelasticity. Finally, the impedance of the respiratory system interacts

with its viscoelastic properties. With regard to clinical evidence, we cautiously conclude that by application of low inspiratory pressures and high respiratory frequencies combined with low tidal volumes, the energy transfer from the respirator to the lung can be reduced. This in turn is potentially lung protective.

Chapter 3

Equation Discovery for Modeling Respiratory Mechanics

Equation discovery (ED) has been introduced [65] to identify functional relations from parameters observed over the course of time. However, while on artificial data such systems have been shown to perform well, the performance on real-world data have to be viewed with a critical eye: "Unfortunately, while a great deal of effort has been expended in designing function-finding systems, little has been done to test them. Researchers have nearly always relied on anecdotal evidence, reporting the successes of their programs on a few hand-selected cases, most of which have consisted of artificial data generated to conform exactly to a functional relationship. Also, although performance clearly depends on the environment in which a function-finding system is deployed, researchers have omitted specification of such an environment in their reporting. What we would really like to know about a function-finding program is not its record of successes on artificial problems chosen by the programmer, but its likelihood of success on a new problem generated in a prespecified environment and involving real scientific data." (Cullen Schaffer [96], 1990, p. 828).

The purpose of this study is to use ED in a real-world problem domain on real-world data. An ED system is applied to model respiratory mechanics. An important system requirement is the capability to process background knowledge given by previously developed mechanical lung models and to present such background knowledge in a flexible manner. This would be the methodological prerequisite to identify even more elaborated models within respiratory data. Respiratory data from 13 mechanically ventilated patients with normal lungs are analyzed. A previously developed system [111] is modified. The original system traverses the hypothesis space in a specific sequential order which shows to have a biased impact on the performance of the system. The benchmark test for the modified system consists of the re-identification of a well-known model of respiratory mechanics.

3.1 Equation Discovery in a Medical Problem Domain

ED systems intend to detect mathematical models for the description and analysis of time series data. Two general approaches are used to achieve this goal. Firstly the data-driven approach, secondly the knowledge-driven approach. In the data-driven approach a vast effort has to be made in order to extract a model from given datasets. Depending on the size of the dataset and the complexity of the hypothesis space this trial-and-error strategy to fit classes of functions to the observed data in general is time-consuming. Alternatively, ED systems try to reduce the hypothesis space by describing possible solutions or parts of a possible solution beforehand. In this way, the system has a certain amount of background knowledge at hand, which reduces the search space and might also be used to guide a heuristic search through the hypothesis space. In general the question remains how much background knowledge should be used. If the hypothesis space is bound by tight constraints represented by the background knowledge one probably looses part of the discovery aspect of finding new systematical relations in the observed data. If the hypothesis space is not reduced appropriately one might not be able to handle time constraints. For this reason it is sound to use classical search strategies for traversing the hypothesis space as it decouples dependencies between the guidance through the search space and the amount of background knowledge given to the system. Moreover, in most of the problem domains it is exactly the discovery part which motivates the application of theses systems. And finally it is often simply the case that there is no particular background knowledge at hand. It rather is the purpose of analyzing data to extract such knowledge.

The first ED system BACON [65] was introduced in 1977 by Pat Langley. BACON.1 [61] identifies two-term numeric laws in measured data. After several versions, BACON.6 was finally capable to discover complex relations involving multiple variables. A hill-climbing technique was capable of handling noisy data [63].

In 1990, Falkenhainer and Michalski introduced the ABACUS system [31]. It is the first system being able to describe a model by more than one equation. Moreover, ABACUS reduced the necessity of user supplied information, which in BACON for example was used to handle noise by a hill-climbing technique. It also performs a units analysis to reduce search space complexity. However, the system is limited in the kind of equations to be discovered and it suffers from the computational cost of the applied search strategies.

The IDS system [82], introduced in 1990 by Nordhausen and Langley, uses a beam search strategy. A novelty is the kind of presenting the results: a history is generated, consisting of a temporal sequence of qualitative states, i.e data samples. This history represents states, numeric conditions for state transitions or qualitative relations between the states. While the description of a physical system is very detailed, the interpretation of such results might be complex.

Zembowicz and Żytkow introduced the EF (equation finder) algorithm [121], improving the handling of noisy data. Equation finder (EF) calculates the deviation of each data point, which is part of the input of the system then. According to its deviation, the impact of a data point with higher deviation is less than one with a small deviation when evaluating the fitting quality of a generated equation. Unfortunately, EF can only handle bivariate data points.

EF is part of the FAHRENHEIT [123] system, developed in 1987 by Żytkow. FAHRENHEIT is a successor of BACON. One difference is the additional output if conditions, i.e. intervals of parameter values for which each law holds. These conditions are partially expressed as functions of other terms themselves. Moreover FAHRENHEIT handles irrelevant terms more efficiently than BACON. However, FAHRENHEIT is not as powerful as the IDS system in its capability to describe qualitative structures for which laws hold.

The system LAGRANGE [27] by Džeroski and Todorovski was the first system to generate higher order differential equations for the description of dynamic systems. It is based on multidimensional linear regression. The introduction of new terms is done by multiplication, which is following the style of inductive logic programming [77, 79]. The generated differential equations could be of a very complex structure and the task of solving such equation system remains as post-processing task. GOLDHORN by Krizman *et al.* [59] extends LAGRANGE in terms of handling noisy data.

Schmidt and Lipson [98] coped with the problem of finding irrelevant relations in the data. A new type of search criteria was defined: to measure how well an expression represents a nontrivial invariance over the experimental data, the partial-derivative of the measured variables were paired as the equations should predict connections between dynamics of subcomponents of the system. As symbolic regression was used, the search could be biased by providing specific combinations of initial mathematical building blocks. However, in some experimental settings the system faced performance deficiencies. And also the ambiguity in terms of units could be promisingly addressed, but not finally solved.

ED systems have to face a variety of problems. Fitting time series is eminently affected by the initial parameter settings within the fitting process and also by noise in the data. Various attempts have been made to address this issue [59, 82, 96, 121]. Another problem is the representation of the results. LAGRANGE [27], for example, is capable of generating models consisting of higher order differential equations, however, the task of solving them remains part of post-processing the results. Another general problem in KDD and ML also affecting ED systems is the handling of vast search spaces. Several approaches have been introduced such as coping with irrelevant variables [31] and detecting inconsistencies of units [31, 57].

Introducing the E* algorithm [96], Schaffer stated that "What we would really like to know about a function-finding program is not its record of successes on artificial problems chosen by the programmer, but its likelihood of success on a new problem generated in a

prespecified environment and involving real scientific data. To date, function-finding research has provided no information on which an estimate of this likelihood might be based. In view of this, my recent research has concentrated on the problem of evaluating function-finding systems [...], and, in the process, I have amassed quite a large collection of real scientific data for use in testing.[...] While previous researchers have concentrated mainly on constructing one of an infinite number of possible functional forms or, equivalently, searching an infinite space of formulas, I believe it is both more accurate and more productive to view function-finding as a classification problem - one of deciding reliably between a fixed, finite set of potential relationships." (Cullen Schaffer [96], 1990, p. 828). Consequently, Schaffer set great value upon testing his system on real-world data. Therefore, E* was developed and tested on no less than 352 real-world datasets. Each set involved two variables for which the reporting scientist has hypothesized a single functional relationship. The task of E* in this environment was to classify one of eight possible functions. Although this testbed of bivariate data and predefined relationships seemed to be a very simple one, Schaffer finally noted: "Previous researchers have attempted to provide methods for more complex function-finding problems in addition to this simple one. My bivariate data gives such strong evidence of the difficulty of the most basic problem, however, that it casts serious doubt on most of these attempts" (Cullen Schaffer [96], 1990, p. 829). This statement indicates that the application of ED in a medical problem domain with real-world data might be a difficult undertaking and some initial experiments confirmed this assumption.

Concerning the proposal of viewing function-finding as a classification problem of deciding reliability between a fixed, finite set of potential relationships, the LAGRAMGE system by Todorovski and S. Džeroski [111] provides an appropriate solution and even goes one step further. Instead of providing a definite choice of possible solutions, LAGRAMGE uses a declarative bias, which is well-known from the field of inductive logic programming [79], to generate such functional relationships inherently. This feature allows to provide a certain amount of background knowledge to the system. Moreover – and this was the original purpose of this declarative bias approach in ED – it reduces the hypothesis space and thus alleviates the problem of efficiency in complex problem domains.

It was our task to apply ED on real-world pressure, flow and volume time series data recorded from the mechanically ventilated human lung. The aim was to extract models from these data. Such models would potentially provide new insight into lung mechanics.

3.2 Methodological Background: the LAGRAMGE Equation Discovery System

Declarative bias was introduced to the field of ED by Todorovski and Džeroski with the LAGRAMGE system [111]. The system uses context free grammars to reduce the hypothesis space. The grammars restrict the choice of mathematical models by defining applicable

model fragments. Parse trees — representing mathematical equations — are derived from the given grammar and span the hypothesis space. In this section, the operation mode of the system and its drawbacks are presented.

3.2.1 System Input

The system operates on a two-part input. The first part consists of a context free grammar, the second of the data, containing one or more time series datasets.

Input Part I: The Context Free Grammar

The first part of the input consists of a context free grammar $G = (N, T, P, S)$ with

- N as set of nonterminal symbols,
- T as set of terminal symbols,
- P as set of productions $\{p_1, \ldots, p_l\}$ and
- $S \in N$ as nonterminal start symbol.

An input grammar for the system would be presented in a grammar file. Such an input is shown as an example in Figure 3.1: this universal grammar allows the combination of the four basic mathematical operations $+$, $-$, $\times$ and $\div$ with the capability to change the order of priority for the operands by using brackets. Formally, the universal grammar G_{U} is given by:

- $G_{\mathrm{U}} = \{N, T, P, S\}$ with
- $N = \{E, F, I, v\}$,
- $T = \{+, -, \times, \div, const, (,)\}$, where *const* represents a constant, its value being determined during the data fit,
- $P = \left\{ \begin{array}{l} E \to E + F, E \to E - F, E \to F, \\ F \to F \times I, F \to F \div I, F \to I, \\ I \to const, I \to v, I \to (E) \end{array} \right\}$

  v represents a placeholder for the domain variables V and is derived to a variable $v_i \in V$, i.e. $v \to v_i \in P$ (the latter implicitly included in P and not to be specified in the input grammar).
- $S = E$ as nonterminal start symbol.

$$\begin{array}{rcl|l|l} E & \rightarrow & E+F & E-F & F \\ F & \rightarrow & F \times I & F \div I & I \\ I & \rightarrow & const & v & (E) \end{array}$$

Figure 3.1: Universal grammar $G_{\mathrm{U}} = (N, T, P, S)$ allowing to apply the four basic mathematical operands $+$, $-$, $\times$ and $\div$ with $N = \{E, F, I, v\}$, $T = \{+, -, \times, \div, const, (,)\}$, $S = E$ and $P = \{p_{E,1}, p_{E,2}, \ldots, p_{I,3}\}$ (e.g., $p_{E,1} = E \rightarrow E + F$). Note the system specific interpretation of the "|" symbol: in the LAGRAMGE system, production rules having the same height are ordered by their listing sequence in the grammar (section 3.2.2, Table 3.1). Therefore, "|" can be interpreted as "before"-relation. In the modified system, "|" is used in its original meaning of "OR".

Each production $p_i \in P$ is of the form $A \rightarrow \alpha$. $A \in N$ is called the left-hand side (or head) and α the right-hand side of the production p_i. In the input grammar file, α is given as $a_1|a_2|\ldots|a_m$, i.e. $A \rightarrow a_1|a_2|\ldots|a_m$, where $a_i \in (N \cup T)^*$. In the following, P_A denotes the set of production rules $\{p_{A,1}, p_{A,2}, \ldots, p_{A,m}\}$ having A as left-hand side. The set of all productions P consists of subsets of productions $P = \{P_A, P_B, P_C, \ldots\}$ which are disjunct by the nonterminal symbols $\{A, B, C, \ldots\}$ on their left-hand sides, e.g. $P_A = \{p_{A,1}, p_{A,2} \ldots, p_{A,m}\} = \{A \rightarrow a_1, A \rightarrow a_2, \ldots, A \rightarrow a_m\}$. Therefore, one has to distinguish between a *production* and a *production rule.* A production $p_i = P_A$ is defined as a set of production rules $\{p_{A,1}, p_{A,2}, \ldots, p_{A,m}\} = \{A \rightarrow a_1, A \rightarrow a_2, \ldots, A \rightarrow a_m\}$ all having the same nonterminal symbol A on the left-hand side. As it turns out, the "|" relation as used in the example input in Figure 3.1 cannot be interpreted as an "OR" relation. In fact, the production rules within each production are specifically ordered before applying them for successively generating parse trees as will be described in section 3.2.2.

Input Part II: The Data

The second part of the input consists of the data $D = (M, V, v_d)$ with

- $M = \{\{t_{0..n}, v_1(t_{0..n}), \ldots, v_k(t_{0..n})\}_{1..m}\}$ as set of one ore more (1 to m) time series datasets (Figures 3.3, 3.4), where each measurement is a table of recorded values of domain variables $v_1 ... v_k$ at successive time points $t_0 ... t_n$.

$time$	v_1	v_2	$\ldots$	v_k
t_0	$v_{1,0}$	$v_{2,0}$	$\ldots$	$v_{k,0}$
t_1	$v_{1,1}$	$v_{2,1}$	$\ldots$	$v_{k,1}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
t_n	$v_{1,n}$	$v_{2,n}$	$\ldots$	$v_{k,n}$

– $V = \{v_1, v_2, \ldots, v_k\}$ as set of domain variables,

– $v_d \in V$ as dependent variable.

3.2.2 Inductive Bias for Model Refinement

A classical approach when using inductive bias in ML [72] is to assume that the simplest consistent hypothesis about the target function potentially is the best hypothesis. This principle is based on Occam's razor (William of Ockham, c. 1285 – c. 1349) which could be informally cited as "the simplest explanation that covers all the facts is usually the best". Amongst others, one benefit of this policy is to avoid overfitting[1] the data. According to this, the LAGRAMGE system also prefers short hypothesis or models, i.e. parse trees with a minimum depth. To realize this inductive bias, the system applies the production rules in a specific sequential order. This ordering is implemented in three ways, which will be described in the following.

Order of Production Rules (I) by Heights of Derivable Parse Trees

The production rules with the same nonterminal A on the left-hand side, i.e. all production rules in the set of production rules P_A, are ordered ascending by the minimal heights h of the parse trees derivable from the rules in P_A with A as root node. The minimal height h of production rule p_i, $h(p_i)$, where $p_i = p_A = A \rightarrow A_1 A_2 \ldots A_n$ and $A_j \in N \cup T$, is calculated as[2]

Definition 1.

$$h(p_i) := 1 + max_{i=1}^{n}\{h(A_i)\},\ where$$

$$h(A) := \left\{ \begin{array}{ll} min_{q \in p_A}\{h(q)\} & ; A \in N \\ 0 & ; A \in T \end{array} \right\}.$$

As a consequence, $\{p_{A,1}, p_{A,2}, \ldots, p_{A,m}\}$ is an ordered set (P_A, R_h) where the relation R_h is defined as

Definition 2. $R_h \subseteq P_A \times P_A : p_{A,i} R_h p_{A,j} :\Leftrightarrow h(p_{A,i}) < h(p_{A,j})$.

When deriving a parse tree, production rules with a minimum height are applied first. The heights of the universal grammar G_{U} in Figure 3.1 are given in Table 3.1.

[1] A model m is said to "overfit" if it performs well on the training data, but shows bad performance over the entire distribution of data samples, i.e. there exists an alternative model m' , which possibly performs worse than model m on the training data, but better over the entire dataset [73].

[2] Cited from [111]

Table 3.1: Heights of production rules in universal grammar G_{U}. Note that the production rule $v \rightarrow \ v_i$, which assigns a domain variable v_i to its placeholder v in the grammar is automatically implied in the grammar definition and is not indicated in the grammar input file.

h	*production rules with equal height h*
1	$I \rightarrow const$, $v \rightarrow v_i$
2	$F \rightarrow I$, $I \rightarrow v$
3	$E \rightarrow F$, $F \rightarrow F \times I$, $F \rightarrow F \div I$
4	$E \rightarrow E + F$, $E \rightarrow E - F$, $I \rightarrow (E)$

Order of Production Rules (II) by Sequential Appearance in the Grammar

The production rules $p_{A,i} \in P_A$ having the same height $h(p_{A,i})$ are applied by the sequence of their appearance in the grammar input file. In such a file, right-hand sides α of the productions $p_A \rightarrow \ \alpha$ having the same left-hand side A are given as $\alpha = a_1|a_2|\ldots|a_m$. The separator "|" splits the production p_A into a set of production rules $P_A = \{p_{A,1}, p_{A,2}, \ldots, p_{A,m}\} = \{A \rightarrow \ a_1, A \rightarrow \ a_2, \ldots, A \rightarrow \ a_m\}$ (with $a_i = A_1 A_2 ... A_n$ and $A_j \in N \cup T$, i.e. $a_i \in (N \cup T)^*$). When talking about P_A as a "set of production rules", the separator "|" would intuitively be interpreted as an "OR" operator. But in contrast, the ordering of the production rules by their sequential appearance implies that the production $E \rightarrow \ E + F|E - F|F$ differs from $E \rightarrow \ F|E - F|E + F$ when defining them in the grammar input file. (Note that this characteristic has been taken into account in Table 3.1.) In general, this means that

$$p_i \rightarrow \ a_1|a_2|\ldots|a_m \neq p_i \rightarrow \ a_m|a_{m-1}|\ldots|a_1$$

which leads to the following definition of the ordering relation R_s. Let $s(p_{A,i})$: *numbered position of* a_i *within the sequential appearance on the right-hand side of* $p_A \rightarrow a_{i-k}|\ldots|a_i|\ldots|a_{i+l}$. Then

Definition 3. $R_s \subseteq P_A \times P_A : p_{A,i} R_s p_{A,j} :\Leftrightarrow h(p_{A,i}) = h(p_{A,j})\ AND\ s(p_{A,i}) < s(p_{A,j})$

Order of Production Rules (III) by Sequential Appearance of Domain Variables

When assigning a domain variable $v_i \in V$ to a nonterminal leaf node, the domain variables are applied in the sequential appearance in the data input files, i.e. left columns before right columns. Therefore, similar to the ordered sets of production rules with the same left-hand sides, the set of domain variables is also ordered by their "*left to right*" appearance in the data input files. Accordingly, for the data files it holds that

$$\begin{array}{c|cccc} time & v_1 & v_2 & \dots & v_k \\ \hline t_0 & v_{1,0} & v_{2,0} & \dots & v_{k,0} \\ t_1 & v_{1,1} & v_{2,1} & \dots & v_{k,1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_n & v_{1,n} & v_{2,n} & \dots & v_{k,n} \end{array} \neq \begin{array}{c|cccc} time & v_k & v_{k-1} & \dots & v_1 \\ \hline t_0 & v_{k,0} & v_{k-1,0} & \dots & v_{1,0} \\ t_1 & v_{k,1} & v_{k-1,1} & \dots & v_{1,1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_n & v_{k,n} & v_{k-1,n} & \dots & v_{1,n} \end{array} .$$

This finally leads to a third ordering relation R_d for the production rules in a set P_A. Let $c(p_{A,i}) :=$ *column number of A counted from left to right in the data input file with* $A \in V$. Then

Definition 4. $R_d \subseteq P_A \times P_A : p_{A,i} R_d p_{A,j} :\Leftrightarrow A \in V$ *AND* $c(p_{A,i}) < c(p_{A,j})$.

Order of Production Rules: Compound and Example

Putting these orderings, i.e. the relations R_h, R_s and R_d (Definitions 2 to 4) together, one obtains a compound ordering relation R_{hsd} given by

Definition 5. $R_{hsd} \subseteq P_A \times P_A : p_{A,i} R_{hsd} p_{A,j} :\Leftrightarrow p_{A,i} R_h p_{A,j}$ *OR* $p_{A,i} R_s p_{A,j}$ *OR* $p_{A,i} R_d p_{A,j}$.

Ordering the sets of production rules P_E, P_F, P_I and P_v of the universal grammar G_{U} (Figure 3.1) by the relation R_{hsd} leads to the four ordered sets[1]

$$\begin{array}{lcll} (P_E, R_{hsd}) & = & \{E \to F, E \to E + F, E \to E - F\} & = \{p_{E,1}, p_{E,2}, p_{E,3}\} \\ (P_F, R_{hsd}) & = & \{F \to I, F \to F \times I, F \to F \div I\} & = \{p_{F,1}, p_{F,2}, p_{F,3}\} \\ (P_I, R_{hsd}) & = & \{I \to const, I \to v, I \to (E)\} & = \{p_{I,1}, p_{I,2}, p_{I,3}\} \\ (P_v, R_{hsd}) & = & \{v \to v_1, v \to v_2, \dots, v \to v_k\} & = \{p_{v,1}, p_{v,2}, \dots, p_{v,k}\} \end{array}$$

Order of Production Rules: Effects

The ordered application of the production rules and the sequential assignment of the domain variables when generating a parse tree implies an additional, characteristical effect to the model finding process. To begin with, Algorithm 3.1 presents the general algorithm for deriving a parse tree from a context free grammar: beginning with the nonterminal start symbol S as root node, a parse tree $\mathcal{T}$ is derived by iteratively applying the valid production rules of the grammar G to each nonterminal node of $\mathcal{T}$ until all leaves of $\mathcal{T}$ consist of terminal symbols.

There are two crucial steps when the sequential order of production rules and domain variables has an effect as an additional inductive bias. Firstly, when the production rule

[1] The order relation of the elements within the sets are represented by the sequence of the entries.

Algorithm 3.1 General algorithm for deriving a parse tree $\mathcal{T}$ from a context free grammar G. (This algorithm refers to the work of L. Todorovski and S. Džeroski ([111], table 2).)

1: begin with parse tree $\mathcal{T}$ consisting of start node S
2: **repeat**
3: choose a nonterminal leaf node A in $\mathcal{T}$
4: choose a production rule $p_{A,i} = A \rightarrow a_i \in P_A$
5: expand node A with all nonterminals and terminals on the right side of $p_{A,i}$
6: **until** all leaf nodes in $\mathcal{T}$ are terminals

Algorithm 3.2 Procedure *RefinementOperator* refines a parse tree by sequential application of ordered production rules. (This algorithm refers to the work of L. Todorovski and S. Džeroski ([111], table 4).)

procedure *RefinementOperator*($\mathcal{T}$, A, h_{max})
1: let $p_{A,i} \in P_A$ be the production rule actually applied at A
2: $l :=$ length of path from A to the root node of $\mathcal{T}$
3: delete the subtrees of node A
4: **if** $i+1 \leq |P_A|$ **and** $l + h(p_{A,i+1}) \leq h_{max}$ **then** // if not all rules $p_{A,i} \in P_A$ were applied yet and overall derivation length will not exceed h_{max} by application of next rule
5: let $p_{A,i+1} = A \rightarrow a_{i+1}$// get next production rule
6: replace $p_{A,i}$ by $p_{A,i+1}$// and replace previously applied rule with new rule
7: expand A with all terminals and nonterminals on the right side of $p_{A,i+1}$
8: **repeat** // iteratively expand the resulting subtree having A as root node
9: choose nonterminal leaf node B in $\mathcal{T}$// by choosing a nonterminal B and
10: let $p_{B,1} = B \rightarrow b_1$// applying first production rule with B on the left side
11: expand node B with all terminals and nonterminals on the right side of $p_{B,1}$
12: **until** all leaf nodes in $\mathcal{T}$ are terminals
13: **end if**

$\mathcal{T}$	parse tree with nonterminal start node S as root
P_A	set of production rules with A on the left-hand side
A, B	nonterminal node in parse tree $\mathcal{T}$
h_{max}	maximum height of refined parse tree $\mathcal{T}$

$p_{A,i}$ is chosen from the ordered set P_A in line 4 of Algorithm 3.1. Instead of choosing any of all possible production rules $p_{A,i} \in P_A$, $p_{A,i}$ is the successor of the previously applied production rule $p_{A,i-1}$ from the ordered set $P_A = \{p_{A,1}, p_{A,2}, \ldots p_{A,m}\}$. This also holds for the sequential order of the domain variables when terminating a leaf node by assigning a domain variable $v_i \in V$ to it. Again, v_i is chosen as the successor of v_{i-1} which has been previously assigned to the nonterminal placeholder v.

The second step showing additional effects to the search is found in line 5 of Algorithm 3.1. When choosing the nonterminal leaf node A the subtree $\mathcal{T}_A$ with root node A

Algorithm 3.3 Top-level LAGRAMGE algorithm performing a beam search. (This algorithm refers to the work of L. Todorovski and S. Džeroski ([111], table 6).)

procedure LAGRAMGE($D, G, h_{max}, w_b, \mathcal{F}, F_\mathrm{h}, \mathcal{S}$)

1: let $\mathcal{T}_0$ be the shallowest derivation tree with no nonterminal leaf nodes
2: $Q := \{\mathcal{T}_0\}$
3: **repeat**
4: $Q' := Q$
5: $R(Q) :=$ set of all possible refinements of parse trees $\mathcal{T}$ in Q as $\bigcup r(\mathcal{T}_\mathrm{i}) : \mathcal{T}_\mathrm{i} \in Q$ with $r(\mathcal{T}_\mathrm{i}) :=$ set of all possible refinements of $\mathcal{T}_\mathrm{i}$
6: fit the constant parameter values of expressions in $R(Q)$ with $\mathcal{F}$
7: evaluate expressions in R according to F_h
8: $Q = Q' \cup R(Q)$
9: retain the w_b best expressions in Q in terms of F_h
10: **until** $Q == Q'$ or $\mathcal{S}$

D	dataset, i.e. the definition of the domain variables V
G	context free grammar
h_{max}	maximum height of the parse trees
w_b	beam width
$\mathcal{F}$	fitting method
F_h	heuristic function
$\mathcal{S}$	stopping criterion (none, timelimit [sec])

has to be refined. It is known which production rule ($p_{A,i-1}$) has been applied to A in the previous iteration. But the resulting subtree $\mathcal{T}'_A$ has to be refined completely from scratch. Consequently, for each nonterminal B in $\mathcal{T}'_A$ the first production rule $p_{B,1} \in P_B$ is applied. The complete algorithm for the refinement of a parse tree $\mathcal{T}$ is shown in Algorithm 3.2. A simple example shown in the section 3.2.4, *Biased Beam Search*, on page 50 demonstrates the mode of operation of the algorithm in detail and shows the drawbacks of the procedure.

3.2.3 Model Evaluation

Each parse tree $\mathcal{T}$ derivable from the grammar G with the nonterminal start symbol S as root node represents a model, which is fit to the given data by the fitting method $\mathcal{F}$ (see line 6 in Algorithm 3.3). In our setting, $\mathcal{F}$ is represented by the implementation of the Nelder-Mead algorithm [81] as described in [87]. The quality of the model is estimated by the heuristic function F_h which in our setting is the RMSE of the model fits (see line 7 in Algorithm 3.3):

$$F_\mathrm{h}(\mathcal{T}) = \sqrt{\frac{\sum_{t=0}^{n}(v_{d,t} - E_t(\mathcal{T}))^2}{n}} \tag{3.1}$$

Here, E_t denotes the expression represented by the derived parse tree $\mathcal{T}$ having the domain variables $v_i \in V$ assigned to their value at time step t. $v_{d,t}$ represents the dependent variable $v_d \in V$ assigned to its value at time step t.

3.2.4 Biased Beam Search

The intention to introduce a declarative bias to ED was to reduce the complexity of the search space. It has been shown that indeed the search space can be strongly affected by providing the system an appropriate input grammar [111]. However, to define such a grammar, an appropriate amount of background knowledge about the system to identify might be necessary. Unfortunately, many tasks probably exhibit a more explorative characteristic where only little background knowledge is available and thus, an explorative search which is implemented in the system most likely still is very time-consuming. The application of a beam search provides a second approach to reduce time consumption at unchanged search space complexity by guiding the traversal of the search space. Applying a beam search [13] of width w_b, in each iteration successively for all w_b beam elements, i.e. parse trees $Q = \{\mathcal{T}_1, \ldots, \mathcal{T}_{w_b}\}$, each possible refinement $\mathcal{T}_i'$ is generated by separately refining each nonterminal node A in $\mathcal{T}_i$. From the set of all possible refinements $R = \{\mathcal{T}_1'^1, \ldots, \mathcal{T}_1'^k, \mathcal{T}_2'^1, \ldots, \mathcal{T}_2'^l, \mathcal{T}_{w_b}'^{\,1}, \ldots, \mathcal{T}_{w_b}'^{\,m}\}$ of all beam elements unified with the set of the beam elements Q of the previous iteration, the w_b elements with the lowest error in terms of the heuristic function F_h are stored in the beam for the next iteration. We will show that the choice of the elements with the lowest error is essential for the modeling process if there are several models of the same quality. The algorithm stops, if within an iteration none of the beam elements was substituted or the stopping criterion holds (Algorithm 3.3). The application of ordered production rules and domain variables within the beam search induces an additional bias to the top-level algorithm.We will show that this bias implies some crucial drawbacks.

Revealing Drawbacks (I): A Theoretical Application Flow

When performing a beam search of width w_b according to Algorithm 3.3, the set of beam elements Q contains the w_b best performing beam elements in terms of the heuristic function F_h. The following simple *example grammar* shows that this can easily lead to hit local minima. We define an input grammar G_S and the data input D_S as:

- Grammar $G_\mathrm{S} = \{N, T, P, S\}$ with
  - $N = \{E, F, v\}$
  - $T = \{\times\}$
  - $P = \{E \to F, F \to F \times F, F \to v\}$ defined in the input grammar file by

$$E \rightarrow F$$
$$F \rightarrow F \times F \mid v$$

- $S = E$

– Data $D_{\mathrm{S}} = \{M, V, v_d\}$ consisting of a single dataset with a single data sample

- $M = \{\{t_0, v_{1,0}, v_{2,0}, v_{3,0}\}\}$ which is provided by the input data file as one time series dataset consisting of a single data sample at point in time t_0

time	v_1	v_2	v_3
t_0	0	2	4

- where the domain variables are presented by $V = \{v_1, v_2, v_3\}$
- and $v_d = v_3$ as dependent variable.

The obvious and simple solution for this task would be the model equation $v_d = v_2 \times v_2$, i.e. $4 = 2 \times 2$. Unfortunately, this equation cannot be found running through Algorithm 3.3 when setting the beam width $w_b = 2$. To begin with, the production rules p_i are ordered by the minimal height of the derivable parse trees with the nonterminal on the left-hand side of the rules as root node. The minimal height h of the production rules is given by:

h	*production rules with equal height h*
1	$v \rightarrow v_i$
2	$F \rightarrow v$
3	$E \rightarrow F$, $F \rightarrow F \times F$

Ordering P by the relation R_{hsd} results in three ordered sets of production rules from which the parse trees $\mathcal{T}_i$ as shown in Table 3.2 and Figure 3.2 with the error according to the heuristic function F_{h} are derived:

$$\begin{aligned}(P_E, R_{hsd}) &= \{E \rightarrow F\} &&= \{p_{E,1}\}\\ (P_F, R_{hsd}) &= \{F \rightarrow v, F \rightarrow F \times F\} &&= \{p_{F,1}, p_{F,2}\}\\ (P_v, R_{hsd}) &= \{v \rightarrow v_1, v \rightarrow v_2\} &&= \{p_{v,1}, p_{v,2}\}\end{aligned}$$

In the following we will show two systematic drawbacks implied in the ordering of production rules as it (i) boosts the hitting of local minima and (ii) leads to the cutting of the hypothesis space.

Drawbacks (i) hitting local minima: In this section we refer to Algorithm 3.3. The sequence of the parse tree refinements is given in Table 3.2. The beam width w_b is set to 2 and no restrictions are made to the maximum height of the parse trees, i.e. $h_{max} = \infty$.

Table 3.2: Refinements of parse trees from ordered sets of production rules (P_E, R_{hsd}), (P_F, R_{hsd}) and (P_v, R_{hsd}) of grammar G_{S} (page 50) according to Algorithm 3.8. The parse trees are listed in ascending order of their height. The basis parse tree $\mathcal{T}_{basis}$ and the refined tree $\mathcal{T}_{refined}$ are given in columns 1 and 2. Column 3 provides all steps to derive $\mathcal{T}_{refined}$. The labeled arrows within the derivation steps indicate which production rules are applied to the associated nonterminals on the left side of the arrow. Production rules and nonterminals are associated in a left to right order. Column 3 also indicates which production rule is newly applied to the associated nonterminal to refine $\mathcal{T}_{basis}$ to $\mathcal{T}_{refined}$: such rules and nonterminals are marked by an overline. The expression $E(\mathcal{T}_i)$ represented by the refined parse tree $\mathcal{T}_i$ is given in column 4. The value of the heuristic function F_{h}, i.e. the RMSE, is given in column 5. The calculation of F_{h} is based on the variable values $v_1 = 0$,$v_2 = 2$ and a value of 4 for the dependent variable v_d (page 51).

$\mathcal{T}_{basis}$	$\mathcal{T}_{refined}$	*derivation and refinement steps*	$E(\mathcal{T}_i)$	$F_{\mathrm{h}}(\mathcal{T}_i)$
S	$\mathcal{T}_0$	$\overline{E} \xrightarrow{\overline{p_{E,1}}} F \xrightarrow{p_{F,1}} v \xrightarrow{p_{v,1}}$	v_1	4
$\mathcal{T}_0$	$\mathcal{T}_1$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,1}} \overline{v} \xrightarrow{\overline{p_{v,2}}}$	v_2	2
$\mathcal{T}_0, \mathcal{T}_1$	$\mathcal{T}_2$	$E \xrightarrow{p_{E,1}} \overline{F} \xrightarrow{\overline{p_{F,2}}} F \times F \xrightarrow{p_{F,1},p_{F,1}} v \times v \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1$	4
$\mathcal{T}_2$	$\mathcal{T}_3$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,1},p_{F,1}} \overline{v} \times v \xrightarrow{\overline{p_{v,2}},p_{v,1}}$	$v_2 \times v_1$	4
$\mathcal{T}_2$	$\mathcal{T}_4$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,1},p_{F,1}} v \times \overline{v} \xrightarrow{p_{v,1},\overline{p_{v,2}}}$	$v_1 \times v_2$	4
$\mathcal{T}_3, \mathcal{T}_4$	$\mathcal{T}_5$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,1},p_{F,1}} v \times \overline{v} \xrightarrow{p_{v,2},\overline{p_{v,2}}}$	$v_2 \times v_2$	0
$\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5$	$\mathcal{T}_6$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} \overline{F} \times F \xrightarrow{\overline{p_{F,2}},p_{F,1}} F \times F \times v \xrightarrow{p_{F,1},p_{F,1},p_{v,1}} v \times v \times v_1 \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_1$	4
$\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5$	$\mathcal{T}_7$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times \overline{F} \xrightarrow{p_{F,1},\overline{p_{F,2}}} v \times F \times F \xrightarrow{p_{v,1},p_{F,1},p_{F,1}} v_1 \times v \times \underset{2}{v} \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_1$	4
$\mathcal{T}_6$	$\mathcal{T}_8$	$\underset{9}{E} \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,2},p_{F,1}} F \times F \times v \xrightarrow{p_{F,1},p_{F,1},p_{v,1}} \overline{v} \times v \times v_1 \xrightarrow{\overline{p_{v,2}},p_{v,1}}$	$v_2 \times v_1 \times v_1$	4
$\mathcal{T}_6$	$\mathcal{T}_9$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,2},p_{F,1}} F \times F \times v \xrightarrow{p_{F,1},p_{F,1},p_{v,1}} v \times \overline{v} \times v_1 \xrightarrow{p_{v,1},\overline{p_{v,2}}}$	$v_1 \times v_2 \times v_1$	4
$\mathcal{T}_6$	$\mathcal{T}_{10}$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,2},p_{F,1}} F \times F \times \overline{v} \xrightarrow{p_{F,1},p_{F,1},\overline{p_{v,2}}} v \times v \times v_2 \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_2$	4
$\mathcal{T}_7$	$\mathcal{T}_{11}$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,1},p_{F,2}} v \times F \times F \xrightarrow{p_{v,1},p_{F,1},p_{F,1}} v_1 \times \overline{v} \times v \xrightarrow{\overline{p_{v,2}},p_{v,1}}$	$v_1 \times v_2 \times v_1$	4
$\mathcal{T}_7$	$\mathcal{T}_{12}$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,1},p_{F,2}} v \times F \times F \xrightarrow{p_{v,1},p_{F,1},p_{F,1}} v_1 \times v \times \overline{v} \xrightarrow{p_{v,1},\overline{p_{v,2}}}$	$v_1 \times v_1 \times v_2$	4
$\mathcal{T}_7$	$\mathcal{T}_{13}$	$E \xrightarrow{p_{E,1}} F \xrightarrow{p_{F,2}} F \times F \xrightarrow{p_{F,1},p_{F,2}} \overline{v} \times F \times F \xrightarrow{\overline{p_{v,2}},p_{F,1},p_{F,1}} v_2 \times v \times v \xrightarrow{p_{v,1},p_{v,1}}$	$v_2 \times v_1 \times v_1$	4
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

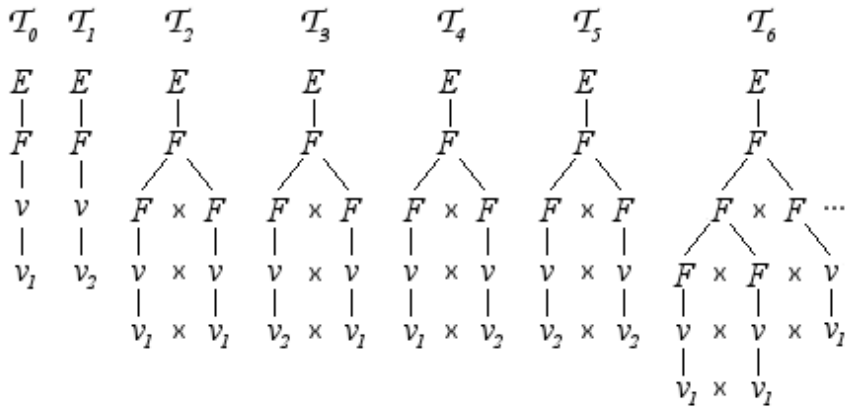

Figure 3.2: Parse trees derived from example grammar G_{S}.

The algorithm starts with the shallowest parse tree $\mathcal{T}_0$ and initializes Q with $Q = \{\mathcal{T}_0\} = \{E \to F \to v \to v_1\}$ representing the model $v_0 = v_1$ (Table 3.2, Figure 3.2).

$$\text{Start: } Q = \{\mathcal{T}_0\}.$$

The evaluation of $\mathcal{T}_0$ according to the heuristic function F_{h} (Equation (3.1)) yields $F_{\mathrm{h}}(\mathcal{T}_0) = \sqrt{1/n \times (v_d - E(\mathcal{T}_0))^2} = \sqrt{1/n \times (v_d - v_1)^2} = \sqrt{1/1 \times (4-0)^2} = 4$. Generating all possible refinements of Q results in set $R(Q) = R(\{\mathcal{T}_0\}) = r(\mathcal{T}_0) = \{\mathcal{T}_1, \mathcal{T}_2\}$ (i.e. $r(\mathcal{T}_0)$ generating all possible refinements of $\mathcal{T}_0$) with $F_{\mathrm{h}}(\mathcal{T}_1) = 2$ and $F_{\mathrm{h}}(\mathcal{T}_2) = 4$. With Q' as copy of Q, Q itself is set to $Q = Q \cup R(Q) = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2\}$. The w_b best expressions in Q according to F_{h} are kept. What if there are several expressions with the same error as given by $F_{\mathrm{h}}(\mathcal{T}_0) = F_{\mathrm{h}}(\mathcal{T}_2) = 4$? We make the following assumption:

Assumption 1. Beam elements are substituted only if a potential substitute has a lower error according to F_{h} than the actually worst performing beam element.

Under this prerequisite we have to discard derivation tree $\mathcal{T}_2$ with the same error as $\mathcal{T}_0$ and get for Q as result of the first iteration

$$\text{End of iteration 1: } Q = \{\mathcal{T}_0, \mathcal{T}_1\}.$$

For the next iteration, deriving all refinements of $\mathcal{T}_0$ we get again $r(\mathcal{T}_0) = \{\mathcal{T}_1, \mathcal{T}_2\}$. Refining $\mathcal{T}_1$ leads to $r(\mathcal{T}_1) = \{\mathcal{T}_2\}$ and thus $Q = Q' \cup R(Q) = \{\mathcal{T}_0, \mathcal{T}_1\} \cup r(\mathcal{T}_0) \cup r(\mathcal{T}_1) = \{\mathcal{T}_0, \mathcal{T}_1\} \cup \{\mathcal{T}_1, \mathcal{T}_2\} \cup \{\mathcal{T}_2\} = \{\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2\}$ with $F_{\mathrm{h}}(\mathcal{T}_0) = 4$, $F_{\mathrm{h}}(\mathcal{T}_1) = 2$ and $F_{\mathrm{h}}(\mathcal{T}_2) = 4$. With reference to Assumption 1, keeping the best performing 2 elements in Q leads to $Q = \{\mathcal{T}_0, \mathcal{T}_1\} = Q'$ and the algorithm stops at end of iteration two with Q set to

$$\text{End of iteration 2: } Q = \{\mathcal{T}_0, \mathcal{T}_1\} \ .$$

Though Assumption 1 is not really arbitrary in the sense of a greedy search strategy and according to Occam's razor, this simple example clearly illustrates the effect which the strict preference of short hypothesis, i.e. shallow parse trees, and what the application of the production rules in the previously described order can cause. To ease this preference we alter Assumption 1 to

Assumption 2. Beam elements are substituted if a potential substitute has a lower or equal error according to F_h than the actually worst performing beam element.

As we will see, for our example this change improves the result. The algorithm again starts with

$$\text{Start: } Q = \{\mathcal{T}_0\}.$$

$\mathcal{T}_0$ can be refined to $\mathcal{T}_1$ and $\mathcal{T}_2$, i.e. $R(Q) = \{\mathcal{T}_1, \mathcal{T}_2\}$. But now, instead of keeping $\mathcal{T}_0$ performing equally to $\mathcal{T}_2$ with a RMSE of 4, $\mathcal{T}_0$ is substituted by $\mathcal{T}_2$ according to Assumption 2 and we get

$$\text{End of iteration 1: } Q = \{\mathcal{T}_1, \mathcal{T}_2\}.$$

Deriving all refinements of $\mathcal{T}_1$ yields $r(\mathcal{T}_1) = \{\mathcal{T}_2\}$ again. $\mathcal{T}_2$ refines to $r(\mathcal{T}_2) = \{\mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_6, \mathcal{T}_7\}$ with $F_\mathrm{h}(\mathcal{T}_3) = F_\mathrm{h}(\mathcal{T}_4) = F_\mathrm{h}(\mathcal{T}_6) = F_\mathrm{h}(\mathcal{T}_7) = 4$ (Table 3.2). The best choice to get close to the model expression $\mathcal{T}_5$ with the lowest error $F_\mathrm{h}(\mathcal{T}_5) = 0$ would be to choose $\mathcal{T}_3$ or $\mathcal{T}_4$. Both trees would have $\mathcal{T}_5$ amongst their refinements, i.e. $\mathcal{T}_5 \in r(\mathcal{T}_3)$ and $\mathcal{T}_5 \in r(\mathcal{T}_4)$. Choosing ($\mathcal{T}_6$) or $\mathcal{T}_7$) would lead to parse trees representing multiplications with 3 multiplicands and more. Such equations produce a RMSE of 4 or larger. Therefore, we arbitrarily choose to substitute $\mathcal{T}_2$ by $\mathcal{T}_3$ and get

$$\text{End of iteration 2: } Q = \{\mathcal{T}_1, \mathcal{T}_3\}.$$

During the next iteration $\mathcal{T}_1$ again refines to $r(\mathcal{T}_1) = \{\mathcal{T}_2\}$ while $\mathcal{T}_3$ refines to $r(\mathcal{T}_3) = \{\mathcal{T}_5, \mathcal{T}_6, \mathcal{T}_7\}$. As $F_\mathrm{h}(\mathcal{T}_1) = 2$, $F_\mathrm{h}(\mathcal{T}_2) = F_\mathrm{h}(\mathcal{T}_3) = F_\mathrm{h}(\mathcal{T}_6) = F_\mathrm{h}(\mathcal{T}_7) = 4$ and $F_\mathrm{h}(\mathcal{T}_5) = 0$, $\mathcal{T}_3$ is substituted by $\mathcal{T}_5$ to

$$\text{End of iteration 3: } Q = \{\mathcal{T}_1, \mathcal{T}_5\}.$$

Next, $\mathcal{T}_1$ refines to $r(\mathcal{T}_1) = \{\mathcal{T}_2\}$ again and $\mathcal{T}_5$ refines to $r(\mathcal{T}_5) = \{\mathcal{T}_6, \mathcal{T}_7\}$ with $F_\mathrm{h}(\mathcal{T}_2) = F_\mathrm{h}(\mathcal{T}_6) = F_\mathrm{h}(\mathcal{T}_7) = 4$. As $F_\mathrm{h}(\mathcal{T}_1) = 2$ and $F_\mathrm{h}(\mathcal{T}_5) = 0$ none of the two elements are substituted and the algorithm ends with

End of iteration 4: $Q = \{\mathcal{T}_1, \mathcal{T}_5\}$.

The algorithm finally identifies the two expressions with the lowest errors which can be produced by the grammar G_{S}. This would not have been possible if in iteration 2 we did not choose $\mathcal{T}_3$ (or $\mathcal{T}_4$) but $\mathcal{T}_6$ or $\mathcal{T}_7$. Of course, in our simple example this problem could have been solved easily by setting the beam width to at least six elements. In this case, iteration two would have ended with $Q = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_5, \mathcal{T}_6, \mathcal{T}_7\}$ and the two best performing elements $\mathcal{T}_1$ and $\mathcal{T}_5$ would have been identified as well. (On the other hand, under Assumption 2 the beam elements would be replaced in a (probably) infinite number of iterations by parse trees with an equal error of 4 and the only way to stop the procedure would be a time limit.) However, for the practical application, the estimation of a reasonable beam width implies another requirement to a reasonable experimental setup – which of course is a common problem in this field. But moreover, an increase of the beam width also leads to an increase of required computational memory. For complex problems this might lead to massive time consumption or even the loss of practicability.

Drawbacks (ii) cutting the hypothesis space: In this section we again refer to Algorithm 3.3 invoked with a beam width w_b of 2 and the maximum height of the parse trees set to $h_{max} = \infty$.

With just a slight change in our setting of the dataset D_{S} we will see that the hypothesis space is possibly cut with the consequence that the derivation trace to the best solution in terms of F_{h} is lost. We simply change the value of the domain variable $v_2 = 2$ to the value of $v_2 = -2$. Remember that the dependent variable v_d equals v_3.

time	v_1	v_2	v_3
t_0	0	-2	4

This changes $F_{\mathrm{h}}(\mathcal{T}_1) = \sqrt{1/n \times (v_d - v_2)^2} = \sqrt{1/1 \times (4 - (-2))^2} = 6$ from value 2 to value 6. Referring to Table 3.2, the values of F_{h} for all other expressions stay the same, especially $F_{\mathrm{h}}(\mathcal{T}_5) = 0$ still shows the best performance with respect to F_{h}. Following Assumption 2, we then get the following iterations:

Start: $Q = \{\mathcal{T}_0\}$.

$\mathcal{T}_0$ is refined to $r(\mathcal{T}_0) = \{\mathcal{T}_1, \mathcal{T}_2\}$. With $F_{\mathrm{h}}(\mathcal{T}_0) = 4$, $F_{\mathrm{h}}(\mathcal{T}_1) = 6$ and $F_{\mathrm{h}}(\mathcal{T}_2) = 4$ we keep $\mathcal{T}_0$ and $\mathcal{T}_2$ (cf. End of iteration 1, page 54) so that iteration one ends with

End of iteration 1: $Q = \{\mathcal{T}_0, \mathcal{T}_2\}$.

Deriving all refinements of $\mathcal{T}_0$ and $\mathcal{T}_2$ yields $R(Q) = r(\mathcal{T}_0) \cup r(\mathcal{T}_2) = \{\mathcal{T}_1, \mathcal{T}_2\} \cup \{\mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_6, \mathcal{T}_7\}$. As $F_\mathrm{h}(\mathcal{T}_0) = F_\mathrm{h}(\mathcal{T}_2) = F_\mathrm{h}(\mathcal{T}_3) = F_\mathrm{h}(\mathcal{T}_4) = F_\mathrm{h}(\mathcal{T}_6) = F_\mathrm{h}(\mathcal{T}_7) = 4$ and $F_\mathrm{h}(\mathcal{T}_1) = 6$ according to Assumption 2 we have to substitute $\mathcal{T}_0$ and $\mathcal{T}_2$ in $Q = Q' \cup R(Q) = \{\mathcal{T}_0, \mathcal{T}_2\} \cup \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_6, \mathcal{T}_7\}$. By arbitrarily choosing $\mathcal{T}_6$ and $\mathcal{T}_7$ we get

$$\text{End of iteration 2: } Q = \{\mathcal{T}_6, \mathcal{T}_7\}.$$

Now it is impossible to derive the best performing expression from $\mathcal{T}_5$ neither from $\mathcal{T}_6$ nor from $\mathcal{T}_7$ (Table 3.2). But moreover, we lost the possible basis trees $\mathcal{T}_3$ and $\mathcal{T}_4$ for $\mathcal{T}_5$ as well as $\mathcal{T}_2$ as basis for $\mathcal{T}_3$ and $\mathcal{T}_4$, $\mathcal{T}_0$ and $\mathcal{T}_1$ as basis for $\mathcal{T}_2$ and finally $\mathcal{T}_0$ as basis for $\mathcal{T}_1$.

We conclude that the traversal of the hypothesis space, which is biased by the height of the parse trees and consequently by the length of the generated numerical expressions implies some crucial drawbacks, which are related to the problem of hitting local minima and cutting the hypothesis space by the sequential application of ordered production rules. Besides these drawbacks related to theoretical considerations, also the practical implementation of the traversal of the hypothesis space implies some pitfalls. In the following section insight is given into these practical aspects.

Revealing Drawbacks (II): A Practical Application Flow

In Algorithm 3.3 the substitution of beam elements implies a nondeterministic behavior (Algorithm 3.3, line 9). Whether following Assumption 1 or Assumption 2, we have two cases in which a random decision has to be made:

- **Nondeterminism case (i):** Firstly, if within one iteration more than one parse tree is generated which has a lower error (Assumption 1) / a lower or equal error (Assumption 2) than the worst performing beam element, but a higher error than the second worst performing element. Then exactly one of the newly generated parse trees has to be arbitrarily chosen as substitute.
- **Nondeterminism case (ii):** Secondly, if the number of refinements is larger than the beam width w_b and all refinements have lower errors (Assumption 1) / lower or equal errors (Assumption 2) than the best performing beam element. Then w_b parse trees have to be arbitrarily chosen as substitutes.

In the implementation of the algorithmic procedures, the nondeterministic choice of a substitute (Algorithm 3.3, line 9) is changed into a deterministic one as given in Algorithm 3.8, line 7 ff.: the refinements $r(\mathcal{T})$ of the basis parse tree $\mathcal{T}$ are generated one after another. (The manner of functioning of Algorithm 3.8 and its subroutines will be described in detail below.) Each of these refinements is evaluated according to F_h (Algorithm 3.8,

line 15), is then compared to the beam element with the highest error (Algorithm 3.8, line 19) and finally substitutes for the worst performing element or not (Algorithm 3.8, line 20). Referring to the nondeterministic behavior in cases (i) and (ii), the algorithmic processing changes as follows:

- **Determinism under Assumption 1:** In case that Assumption 1 holds, this means that if within one iteration $\mathcal{T}_i \in r(\mathcal{T})$ is generated, evaluated and inserted into the beam before $\mathcal{T}_j \in r(\mathcal{T})$ is generated and $\mathcal{T}_j$ has the same error, then $\mathcal{T}_j$ is not represented in the beam. The inverse holds if $\mathcal{T}_j$ is processed before $\mathcal{T}_i$.
- **Determinism under Assumption 2:** In case that Assumption 2 holds, this means that if within one iteration $\mathcal{T}_i \in r(\mathcal{T})$ is generated, evaluated and inserted into the beam before $\mathcal{T}_j \in r(\mathcal{T})$ is generated and evaluated and $\mathcal{T}_j$ has the same error as $\mathcal{T}_i$, then $\mathcal{T}_j$ substitutes for $\mathcal{T}_i$ and the latter is not represented in the beam. The inverse holds if $\mathcal{T}_j$ is processed before $\mathcal{T}_i$.

All refinements $r(\mathcal{T})$ of a substituted parse tree $\mathcal{T}$ and recursively all refinements of these refinements will not be generated and evaluated in future iterations for the lack of $\mathcal{T}$ (unless they can be generated from a different basis tree, cf. Table 3.2). As a consequence, it is essential for the final results in which sequence the parse trees are generated. This is partially regulated by the sequential application of the ordered production rules. But before choosing a production rule for the refinement of a parse tree, the nonterminal node of the basis parse tree which provides the root for the refinement (i.e. the head/left-hand side of the production rule) has to be chosen. In which sequence should the nonterminal nodes of a parse tree be refined (Algorithm 3.3, line 5)? To understand this, it is important to know, in which way a single parse tree is derived.

Deterministic parse tree refinement by sequential derivation steps: Before going through Algorithm 3.8 and its subroutines given in Algorithms 3.5 to 3.7, the concept of a derivation step has to be explained in more detail. Remembering section 3.2.1 we find that a production rule p_i is given as $p_i = p_A = A \to A_1 A_2 \dots A_n$ with $A_j \in N \cup T$. One derivation step to derive a parse tree $\mathcal{T}$ is given as follows. Let *expression* $A_1 \dots A_k \dots A_n \in (N \cup T)^*$ *be the expression before application of* $p_{A,k}$ *and* $A_1 \dots A_{k-1} A'_1 \dots A'_m A_{k+1} \dots A_n \in (N \cup T)^*$ *be the expression after application of* $p_{A,k}$. Then

Definition 6. $dstep(\mathcal{T}) : A_1 \dots A_k \dots A_n \overset{p_{A,k}}{\to} A_1 \dots A_{k-1} A'_1 \dots A'_m A_{k+1} \dots A_n$

The expression specified by a derivation step $dstep(\mathcal{T})$ is given as follows. Let $A_1 \dots A_k \dots A_n \in (N \cup T)^*$ *be the expression on the left-hand side of* $dstep(\mathcal{T})$ *and expression* $A_1 \dots A_{k-1} A'_1 \dots A'_m A_{k+1} \dots A_n \in (N \cup T)^*$ *the expression on the right-hand side of* $dstep(\mathcal{T})$.

Definition 7. $e(dstep(\mathcal{T})) : A_1 \dots A_{k-1} A'_1 \dots A'_m A_{k+1} \dots A_n \in (N \cup T)^*$ *on the right-hand side of* $dstep(\mathcal{T})$.

Algorithm 3.4 Implementation of General algorithm for deriving a parse tree $\mathcal{T}$ from a context free grammar with a maximum height of h_{max}.

1: begin with parse tree $\mathcal{T}$ consisting of start node S
2: $i := 1$
3: apply $p_{S,1}$ as $dstep_i(\mathcal{T})$
4: apply $RefinementOperator_{Impl}(\mathcal{T}, dstep_i(\mathcal{T}), h_{max})$// see Algorithm 3.5

$\mathcal{T}$	parse tree with nonterminal start node S as root
$p_{S,1}$	first production rule with S on the left-hand side
$dstep_i(\mathcal{T})$	i-th derivation step applied to derive $\mathcal{T}$ (Definition 6)
h_{max}	maximum height of derived parse tree $\mathcal{T}$

An example of a parse tree refinement (and the application of subsequent derivation steps) is given in Table 3.3. Referring to $dstep_3(\mathcal{T}_6)$, we can see that the expression $F \times F \times F$ can be derived in two ways. Either the production rule $F \rightarrow F \times F$ is applied to the first nonterminal node F in the preceding expression $F \times F$ or to the second. In both cases the final expression $v_1 \times v_1 \times v_1$ can be derived. This leads to the following observations (known in context of ideal refinement operators in inductive logic programming [116]) which are important in the following:

Statement 1. *A parse tree $\mathcal{T}$ is uniquely defined by the sequence of the applied derivation steps $dstep_0(\mathcal{T})$ to $dstep_n(\mathcal{T})$ which are applied to generate a final expression $e(dstep_n(\mathcal{T}))$ merely consisting of nonterminals.*

Statement 2. *The identical expression can be derived by different sequences of derivation steps. (In contrast to the unique sequence of derivation steps for deriving a specific parse tree.) This especially holds for the expression $e(dstep_n(\mathcal{T}))$ at the end of the parse tree generation which is representing the model.*

In the system implementation (Algorithm 3.8), when refining a parse tree, in each derivation step deterministically always the most left nonterminal symbol of the expression derived by the previous derivation step is refined. (Consequently, A_1 to A_{k-1} are all terminal symbols and on the other hand, A_{k+1} to A_n as well as the inserted symbols A'_1 to A'_m might be terminals or nonterminals.) The implementation of the parse tree derivation is explained in Algorithm 3.4 and its subroutines in Algorithms 3.5 to 3.7. The crucial step is found in lines 1 and 2 in Algorithm 3.6. With each derivation step, a new expression consisting of terminals and nonterminals is completed. After the final derivation step, the tree is completely refined and no more nonterminals are found in the leaf nodes.

Within the top-level beam search (Algorithm 3.8), parse trees are not derived from scratch. This only happens once, right at the beginning of the search. Running the search, new parse trees are derived (refined) from previously generated parse trees. For each beam element $b_i = \mathcal{T}_j$, all its refinement successors have to be generated and evaluated

Table 3.3: Example for a single parse tree refinement. *Upper table*: sequence of steps to derive parse tree $\mathcal{T}_6$ from the example grammar G_S (see page 50 and Tables 3.2, 3.4). $dstep_i(\mathcal{T}_6)$ denotes the i-th derivation step to derive $\mathcal{T}_6$ resulting in the expression $e(dstep_i(\mathcal{T}_6))$ given in column three by application of the production rule given in column two. *Lower table*: sequence of steps to derive parse tree $\mathcal{T}_6'$ by substituting production rule $p_{F,2} = F \to F \times F$ for $p_{F,1} = F \to v$ applied in $dstep_6(\mathcal{T})$.

$dstep_i(\mathcal{T}_6)$	*applied rule*	$e(dstep_i(\mathcal{T}_6))$
$dstep_0(\mathcal{T}_6)$	$S = E$	E
$dstep_1(\mathcal{T}_6)$	$E \to F$	F
$dstep_2(\mathcal{T}_6)$	$F \to F \times F$	$F \times F$
$dstep_3(\mathcal{T}_6)$	$F \to F \times F$	$F \times F \times F$
$dstep_4(\mathcal{T}_6)$	$F \to v$	$v \times F \times F$
$dstep_5(\mathcal{T}_6)$	$v \to v_1$	$v_1 \times F \times F$
$dstep_6(\mathcal{T}_6)$	$F \to v$	$v_1 \times v \times F$
$dstep_7(\mathcal{T}_6)$	$v \to v_1$	$v_1 \times v_1 \times F$
$dstep_8(\mathcal{T}_6)$	$F \to v$	$v_1 \times v_1 \times v$
$dstep_9(\mathcal{T}_6)$	$v \to v_1$	$v_1 \times v_1 \times v_1$

$\longrightarrow$ substitution of production rule $p_{F,2} = F \to F \times F$ for $p_{F,1} = F \to v$ in $dstep_6(\mathcal{T})$ and in the following applying $p_{F,1} = F \to v$, $p_{v,1} = v \to v_1$, $p_{F,1} = F \to v$, $p_{v,1} = v \to v_1$ in derivation steps 7 to 10 in a left to right sequence of the appearance of nonterminal symbols of each expression. Note that the new derivation steps 11 and 12 are identical to previous steps 8 and 9, respectively, but are renumbered as 2 steps have been deleted and 5 new ones inserted.$\longrightarrow$

$dstep_i(\mathcal{T}'_6)$	*applied rule*	$e(dstep_i(\mathcal{T}'_6))$
$dstep_0(\mathcal{T}_6')$	$S = E$	E
$dstep_1(\mathcal{T}_6')$	$E \to F$	F
$dstep_2(\mathcal{T}_6')$	$F \to F \times F$	$F \times F$
$dstep_3(\mathcal{T}_6')$	$F \to F \times F$	$F \times F \times F$
$dstep_4(\mathcal{T}_6')$	$F \to v$	$v \times F \times F$
$dstep_5(\mathcal{T}_6')$	$v \to v_1$	$v_1 \times F \times F$
$dstep_6(\mathcal{T}_6')$	$F \to F \times F$	$v_1 \times F \times F \times F$
$dstep_7(\mathcal{T}_6')$	$F \to v$	$v_1 \times v \times F \times F$
$dstep_8(\mathcal{T}_6')$	$v \to v_1$	$v_1 \times v_1 \times F \times F$
$dstep_9(\mathcal{T}_6')$	$F \to v$	$v_1 \times v_1 \times v \times F$
$dstep_{10}(\mathcal{T}_6')$	$v \to v_1$	$v_1 \times v_1 \times v_1 \times F$
$dstep_{11}(\mathcal{T}_6')$	$F \to v$	$v_1 \times v_1 \times v_1 \times v$
$dstep_{12}(\mathcal{T}_6')$	$F \to v$	$v_1 \times v_1 \times v_1 \times v_1$

Table 3.4: Refinements of parse trees $\mathcal{T}$ biased by sequence of derivation steps and by the ordered sets of production rules (P_E, R_{hsd}), (P_F, R_{hsd}) and (P_v, R_{hsd}) of grammar G_{S} (page 50) according to Algorithm 3.8. The basis parse tree $\mathcal{T}_{basis}$ and the refined tree $\mathcal{T}_{refined}$ are given in columns 1 and 2. Column 3 provides all steps to derive $\mathcal{T}_{refined}$. The labeled arrows within the derivation steps indicate which production rules $p_{A,i}$ are applied to the associated nonterminals on the left side of the arrow. Production rules and nonterminals are associated in a left to right order. Column 3 also indicates which production rule is newly applied to the associated nonterminal to refine $\mathcal{T}_{basis}$ to $\mathcal{T}_{refined}$: such rules and nonterminals are marked by an overline. The numbers below the nonterminal symbols indicate the sequence of the derivation steps. If this number is given in round brackets, then the last production rule $p_{A,i}$ in the sequence of rules applicable to A is applied. The expression $E(\mathcal{T}_i)$ represented by the refined parse tree $\mathcal{T}_i$ is given in column 4. The value of the heuristic function F_{h}, i.e. the RMSE, is given in column 5. The calculation of F_{h} is based on the variable values $v_1 = 0$,$v_2 = 2$ and a value of 4 for the dependent variable v_d (page 51). The sequential refinements for each basis tree are grouped between horizontal lines. Square brackets following the name of a basis tree denote which nonterminal was processed to refine the tree. If an expression E can be derived from different basis trees $\mathcal{T}$, those are listed in round brackets. Note that such expressions do not necessarily have identical derivation step sequences.

$\mathcal{T}_{basis}$	$\mathcal{T}_{refined}$	*derivation and refinement steps*	$E(\mathcal{T}_i)$	$F_{\mathrm{h}}(\mathcal{T}_i)$
S	$\mathcal{T}_0$	$\overset{(1)}{\overline{E}} \xrightarrow{\overline{p_{E,1}}} \underset{2}{F} \xrightarrow{p_{F,1}} \underset{3}{v} \xrightarrow{p_{v,1}}$	v_1	4
$\mathcal{T}_0[3]$	$\mathcal{T}_1$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{2}{F} \xrightarrow{p_{F,1}} \overset{(3)}{\overline{v}} \xrightarrow{\overline{p_{v,2}}}$	v_2	2
$\mathcal{T}_0[2], (\mathcal{T}_1[2])$	$\mathcal{T}_2$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \overset{(2)}{\overline{F}} \xrightarrow{\overline{p_{F,2}}} \underset{3}{F} \times \underset{5}{F} \xrightarrow{p_{F,1},p_{F,1}} \underset{4}{v} \times \underset{6}{v} \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1$	4
$\mathcal{T}_1[2]$	$\mathcal{T}_2$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \overset{(2)}{\overline{F}} \xrightarrow{\overline{p_{F,2}}} \underset{3}{F} \times \underset{5}{F} \xrightarrow{p_{F,1},p_{F,1}} \underset{4}{v} \times \underset{6}{v} \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1$	4
$\mathcal{T}_2[6]$	$\mathcal{T}_3$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \underset{3}{F} \times \underset{5}{F} \xrightarrow{p_{F,1},p_{F,1}} \underset{4}{v} \times \overset{(6)}{\overline{v}} \xrightarrow{p_{v,1},\overline{p_{v,2}}}$	$v_1 \times v_2$	4
$\mathcal{T}_2[5], (\mathcal{T}_3[5])$	$\mathcal{T}_4$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \underset{3}{F} \times \overset{(5)}{\overline{F}} \xrightarrow{p_{F,1},\overline{p_{F,2}}} \underset{4}{v} \times \underset{6}{F} \times \underset{8}{F} \xrightarrow{p_{v,1},p_{F,1},p_{F,1}} v_1 \times \underset{7}{v} \times \underset{9}{v} \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_1$	4
$\mathcal{T}_2[4]$	$\mathcal{T}_5$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \underset{3}{F} \times \underset{5}{F} \xrightarrow{p_{F,1},p_{F,1}} \overset{(4)}{\overline{v}} \times \underset{6}{v} \xrightarrow{\overline{p_{v,2}},p_{v,1}}$	$v_2 \times v_1$	4
$\mathcal{T}_2[3], (\mathcal{T}_5[3])$	$\mathcal{T}_6$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \overset{(3)}{\overline{F}} \times \underset{8}{F} \xrightarrow{\overline{p_{F,2}},p_{F,1}} \underset{4}{F} \times \underset{6}{F} \times \underset{9}{v} \xrightarrow{p_{F,1},p_{F,1},p_{v,1}} \underset{5}{v} \times \underset{7}{v} \times v_1 \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_1$	4
$\mathcal{T}_3[5]$	$\mathcal{T}_4$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \underset{3}{F} \times \overset{(5)}{\overline{F}} \xrightarrow{p_{F,1},\overline{p_{F,2}}} \underset{4}{v} \times \underset{6}{F} \times \underset{8}{F} \xrightarrow{p_{v,1},p_{F,1},p_{F,1}} v_1 \times \underset{7}{v} \times \underset{9}{v} \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_1$	4
$\mathcal{T}_3[4]$	$\mathcal{T}_7$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \underset{3}{F} \times \underset{5}{F} \xrightarrow{p_{F,1},p_{F,1}} \overset{(4)}{\overline{v}} \times \overset{(6)}{v} \xrightarrow{\overline{p_{v,2}},p_{v,2}}$	$v_2 \times v_2$	0
$\mathcal{T}_3[3]$	$\mathcal{T}_8$	$\overset{(1)}{E} \xrightarrow{p_{E,1}} \underset{(2)}{F} \xrightarrow{p_{F,2}} \overset{(3)}{\overline{F}} \times \underset{8}{F} \xrightarrow{\overline{p_{F,2}},p_{F,1}} \underset{4}{F} \times \underset{6}{F} \times \underset{9}{v} \xrightarrow{p_{F,1},p_{F,1},p_{v,2}} \underset{5}{v} \times \underset{7}{v} \times v_2 \xrightarrow{p_{v,1},p_{v,1}}$	$v_1 \times v_1 \times v_2$	4
⋮	⋮	⋮	⋮	⋮

according to the heuristic function F_h. Therefore, starting from all nonterminal nodes of $\mathcal{T}_j$, refinements are generated (Algorithm 3.1). As mentioned above, this nondeterministic step is altered into a deterministic one in the system implementation. The crucial point here again is that this is done in a specific sequence.

Deterministic derivation of refinement sets by sequential parse tree nodes: Referring to Statement 1, a parse tree $\mathcal{T}$ is uniquely defined by the sequence of the applied derivation steps $dstep_0(\mathcal{T})$ to $dstep_n(\mathcal{T})$. Each derivation step represents an expression $E = e(dstep_i(\mathcal{T}))$ derived from derivation step $dstep_i(\mathcal{T})$. Such expressions consist of terminals and nonterminals $E := (N \cup T)^*$ (besides the expression after the last derivation step, solely consisting of terminal symbols). I.e., all such expressions contain at least one nonterminal symbol (Table 3.3). To process derivation step $dstep_{i+1}(\mathcal{T})$, for each nonterminal A in $e(dstep_i(\mathcal{T}))$ a new subtree is generated by application of the successor production rule $p_{A,j+1}$ of the in $e(dstep_i(\mathcal{T}))$ previously applied $p_{A,j}$. This is done in a left to right sequence of nonterminals appearing in $e(dstep_i(\mathcal{T}))$. Based on this, the following statement is made concerning the implementation of the parse tree refinement:

Statement 3. *To derive all possible refinements of a parse tree $\mathcal{T}$, all derivation steps* 1 *to* n *to derive $\mathcal{T}$ are traversed in the reversed order $n \to 1$. For each derivation step i, the applied production rule p_A is substituted by the successor production rule $p_{A,j+1}$. This implies that for each nonterminal symbol A within an expression $e(dstep_{1..n}(\mathcal{T}))$ solely the successor production rule $p_{A,j+1}$ of the previously applied rule $p_{A,j}$ is used.*

(In this way, the "basis" parse tree ($\mathcal{T}$) is preserved to process all refinements without destroying the tree structure for further derivations; if one would start building refinements, e.g. from the first derivation step, the tree structure most probably would completely change. Starting the building of refinements from the last derivation step, the preceding derivation steps are kept and can be traversed step by step. From a practical perspective, in this way, the duplication of the basis tree can be omitted. See Table 3.3 and Algorithm 3.8 to reproduce.)

Table 3.4 depicts the sequential derivation of refinements biased by the sequence of the derivation steps (and by the ordered set of the production rules) according to Algorithm 3.4. What can be clearly seen at first view is that the same model expressions are represented several times, e.g. $v_1 \times v_1 \times v_1$. The reason is that such expressions can be derived by different derivation sequences (Statement 2) and these, in fact, represent different parse trees, which might refine to diverse parse trees representing different model expressions. Therefore, when the algorithm checks if a parse tree already exists in the beam to avoid redundancies, it is actually checked if the sequence of derivation steps differs and not if the finally derived expressions are different (Statement 1; Algorithm 3.8, line 11).

Drawbacks (iii) repetitive identical parse tree refinement: To show the impact of the deterministic parse tree refinement we take up again the example of the theoretical application flow (page 50) with the beam width w_b set to 2 and process the data following

Algorithm 3.5 Implementation of procedure *RefinementOperator* refining a parse tree by sequential application of production rules.

procedure $RefinementOperator_{Impl}(\mathcal{T}, dstep_i(\mathcal{T}), h_{max})$

1: **repeat**
2: apply $NextDerivationStep(\mathcal{T}, dstep_i(\mathcal{T}), h_{max})$// see Algorithm 3.6
3: **until** $\mathcal{T}$ contains no more nonterminal leaf nodes

$\mathcal{T}$	parse tree with nonterminal start node S as root
$dstep_i(\mathcal{T})$	i-th derivation step applied to derive $\mathcal{T}$ (Definition 6)
h_{max}	maximum height of derived parse tree $\mathcal{T}$

Algorithm 3.6 Procedure for application of the successor of the i-th derivation step $dstep_i(\mathcal{T})$, which has been applied to derive a parse tree $\mathcal{T}$.

procedure $NextDerivationStep(\mathcal{T}, dstep_i(\mathcal{T}), h_{max})$

1: $E := e(dstep_i(\mathcal{T}))$
2: let A be the most left nonterminal in E
3: apply $NextProductionRule(\mathcal{T}, A, h_{max})$// see Algorithm 3.7

$\mathcal{T}$	parse tree with nonterminal start node S as root
$dstep_i(\mathcal{T})$	i-th derivation step applied to derive $\mathcal{T}$ (Definition 6)
h_{max}	maximum height of derived parse tree $\mathcal{T}$
E	expression consisting of nonterminal and terminal symbols, $E := (N \cup T)^*$
$e(dstep_i(\mathcal{T}))$	expression E derived after derivation step $dstep_i(\mathcal{T})$ (Definition 7)

Algorithm 3.7 Procedure *NextProductionRule* applies successor $p_{A,i+1}$ of production rule $p_{A,i}$ which has been applied to nonterminal node A in parse tree $\mathcal{T}$.

procedure *NextProductionRule*($\mathcal{T}$, A, h_{max})

1: let $p_{A,i} \in P_A$ be the production rule actually applied at A
2: $l :=$ length of path from A to the root node of $\mathcal{T}$
3: delete the subtrees of node A
4: **if** $i+1 \leq |P_A|$ **and** $l + h(p_{A,i+1}) \leq h_{max}$ **then** // if not all rules $p_{A,i} \in P_A$ were applied yet and overall derivation length will not exceed h_{max} by application of next rule
5: let $p_{A,i+1} = A \rightarrow a_{i+1}$// get next production rule
6: replace $p_{A,i}$ by $p_{A,i+1}$// and replace previously applied rule with new rule
7: expand A with all terminals and nonterminals on the right side of $p_{A,i+1}$
8: **end if**

$\mathcal{T}$	parse tree with nonterminal start node S as root
A	nonterminal node in parse tree $\mathcal{T}$
h_{max}	maximum height of refined parse tree $\mathcal{T}$
P_A	set of production rules with A on the left-hand side

Algorithm 3.8 Implementation of top-level LAGRAMGE algorithm performing a beam search.

procedure LAGRAMGE$_{Impl}(D, G, h_{max}, w_b, (B, <), \mathcal{F}, F_\mathrm{h}, \mathcal{S})$

1: let $\mathcal{T}_0$ be the shallowest derivation tree with no nonterminal leaf nodes
2: fit the constant parameter values of $\mathcal{T}_0$ with fitting method $\mathcal{F}$
3: evaluate error for expression represented by $\mathcal{T}_0$ according to heuristic function F_h
4: $(B, <) \ni b_1 := \mathcal{T}_0$// initiate beam with $\mathcal{T}_0$ as first beam element
5: **repeat**
6: $(B, <)' := (B, <)$// copy actual beam
7: **for** $i := 1 \rightarrow w_b$ (see footnote[1]) **do** // for all beam elements
8: $\mathcal{T} := b_i \in (B, <)'$// copy beam element as derivation tree $\mathcal{T}$ for actual evaluation
9: **for** $dstep_n(\mathcal{T})$ to $dstep_1(\mathcal{T})$ **do** // for all derivation steps of $\mathcal{T}$ do in reversed order
10: apply $RefinementOperator_{impl}(\mathcal{T}, dstep_i(\mathcal{T}), h_{max})$// refine $\mathcal{T}$ until it contains no more nonterminal leaf nodes, see Algorithm 3.5
11: **if** $\exists b_j \in (B, <)$, so that $\mathcal{T}$ is derived by the identical sequence of derivation steps as b_j **then**
12: continue
13: **end if**
14: fit the constant parameter values of $\mathcal{T}$ with $\mathcal{F}$
15: evaluate expression represented by $\mathcal{T}$ according to F_h
16: **if** b_{w_b} is empty beam element **then** // while beam contains empty elements
17: $b_{w_b} := \mathcal{T}$// insert new element
18: sort $(B, <)$// and sort elements in descending order according to F_h
19: **else if** $F_\mathrm{h}(\mathcal{T}) < F_\mathrm{h}(b_1)$ AND $\mathcal{T} \notin (B, <)$ **then** // if beam is completely filled and $\mathcal{T}$ has lower error than first beam element and $\mathcal{T}$ is not element of beam
20: $b_1 := \mathcal{T}$// replace first beam element, i.e. element with highest error
21: sort $(B, <)$// and sort elements in descending order according to F_h
22: **end if**
23: **end for**
24: **end for**
25: **until** $(B, <) == (B, <)'$ **or** $\mathcal{S}$ // until no more beam elements have changed or stopping criterion is fulfilled

D	dataset, i.e. the definition of the domain variables V
G	context free grammar
h_{max}	maximum height of the parse trees
w_b	beam width
$(B, <)$	ordered set of beam elements $B = \{b_1, b_2, \ldots, b_{w_b}\}$: $b_i < b_j :\Leftrightarrow F_\mathrm{h}(b_i) > F_\mathrm{h}(b_j))$ (Definition 8)
$\mathcal{F}$	fitting method
F_h	heuristic function
$\mathcal{S}$	stopping criterion (none, timelimit [sec])
$dstep_i(\mathcal{T})$	i-th applied derivation step to derive parse tree $\mathcal{T}$ (Definition 6)

[1] In the system implementation, the elements of the sorted beam $(B, >)$ are refined in a left to right sequence. See Appendix C for the analysis of the impact to the system performance.

the practical implementation given in Algorithm 3.8, which follows Assumption 1 (page 53) and Statement 3. See Table 3.4 for the sequentially derived parse trees. In the following, $(B, <)$ denotes an ordered set of beam elements: Let $B = \{b_1, \ldots, b_{w_b}\}$ *be a beam of width* w_b *and* F_h *the error function to measure the performance of the single beam elements.*

Definition 8. $(B, <) : b_i \in B < b_j \in B :\Leftrightarrow F_\mathrm{h}(b_i) > F_\mathrm{h}(b_j)$

The algorithm starts with the shallowest parse tree $\mathcal{T}_0$ and the beam $(B, <)$ is initialized with (Algorithm 3.8, line 4):

$$\text{Start: } (B, <) = \{\mathcal{T}_0\}.$$

The first refinement of $\mathcal{T}_0$ according to the derivation sequence (Algorithm 3.8, line 9, Table 3.4) is $\mathcal{T}_1$ with an error $F_\mathrm{h}(\mathcal{T}_1) = 2$. As the beam element $b_2 \in B$ is still empty, $\mathcal{T}_1$ is collected as b_2 (Algorithm 3.8, line 16). The next refinement of $\mathcal{T}_0$ is $\mathcal{T}_2$ with $F_\mathrm{h}(\mathcal{T}_2) = 4$. As its error is not lower than that of the worst performing element $\mathcal{T}_0 = b_1 \in (B, <)$ it is rejected. (Note that after each replacement of the worst performing beam element b_1, $(B, <)$ is rearranged in descending order of the errors of the beam elements (Algorithm 3.8, line 18). At end of the first iteration we get

$$\text{End of iteration 1: } (B, <) = \{b_1, b_2\} = \{\mathcal{T}_0, \mathcal{T}_1\}.$$

Starting the next iteration with element $b_1 = \mathcal{T}_0$ (Algorithm 3.8, line 7), we begin the refinement sequence with $\mathcal{T}_1$ again (Algorithm 3.8, line 9, Table 3.4). As $\mathcal{T}_1$ is already represented in the beam, it is rejected (Algorithm 3.8, line 11). The next processed refinement of $\mathcal{T}_0$ consists of $\mathcal{T}_2$ again, which is rejected as it has the same error as b_1 (Algorithm 3.8, line 19). Now $b_2 = \mathcal{T}_1$ is refined and its first and only refinement consists of $\mathcal{T}_2$. Note that there are the three nonterminals E, F and v found in $\mathcal{T}_1$. But production rule $p_{F,2}$ is the only rule which can be applied according to the sequential application of production rules (page 51). As $\mathcal{T}_2$ does not perform better than b_1 it is rejected. Finally, iteration 2 ends with

$$\text{End of iteration 2: } (B, <) = \{b_1, b_2\} = \{\mathcal{T}_0, \mathcal{T}_1\}.$$

As all beam elements are identical to those at the end if iteration one, i.e. $(B, <) = (B, <)'$ (Algorithm 3.8, line 25), the algorithms stops after only two iterations and the expression with the best performance has not been detected ($\mathcal{T}_7$ in Table 3.4). In conclusion, this result does not differ from the result when using the nondeterministic procedure (Algorithm 3.3) as shown in the example when hitting local minima (page 51).

Nevertheless, the deterministic application flow can negatively impact the outcome. Let us repeat the above example, but this time the beam width w_b is set to 4. The algorithm starts again with

$$\text{Start: } (B, <) = \{\mathcal{T}_0\}.$$

$(B, <)$ is copied to $(B, <)'$ and all $b_i \in (B, <)'$ are refined. Refining $\mathcal{T}_0$ first leads to $\mathcal{T}_1$ with $F_\text{h}(\mathcal{T}_1) = 2$. As the beam is not completely filled (Algorithm 3.8, line 16), $\mathcal{T}_1$ is inserted and after sorting the beam elements (Algorithm 3.8, line 18) finally $(B, <) = \{\mathcal{T}_0, \mathcal{T}_1\}$. The next refinement of $\mathcal{T}_0$ is given by $\mathcal{T}_2$. Again, as the beam is not filled, $\mathcal{T}_2$ is inserted, the beam is sorted and at the end of iteration 1 we get

$$\text{End of iteration 1: } (B, <) = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_1\}.$$

In the next iteration $(B, <)$ is copied again to $(B, <)'$. First $\mathcal{T}_0 \in (B, <)'$ is again refined to $\mathcal{T}_1$ and $\mathcal{T}_2$. As both trees are already elements of $(B, <)$, no insertion or substitution, respectively, is made. The first refinement of the second beam element $\mathcal{T}_2$ is given by $\mathcal{T}_3$ with $F_\text{h}(\mathcal{T}_3) = 4$. The beam still is not completely filled and $\mathcal{T}_3$ is inserted. After sorting we get $(B, <) = (\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_1)$. The next refinements of $\mathcal{T}_2$ are derived in the sequence $\mathcal{T}_4$, $\mathcal{T}_5$ and finally $\mathcal{T}_6$ with $F_\text{h}(\mathcal{T}_4) = F_\text{h}(\mathcal{T}_5) = F_\text{h}(\mathcal{T}_6) = 4$. As none of these parse trees performs better than $b_1 = \mathcal{T}_0$, no substitution is made. Finally the last element $\mathcal{T}_1$ of $(B, <)'$ is refined to $\mathcal{T}_2$, which is already element of $(B, <)$ and therefore not inserted. Iteration 2 ends with

$$\text{End of iteration 2: } (B, <) = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_1\}$$

At the beginning of iteration 3 $(B, <)$ is copied again to $(B, <)'$. $\mathcal{T}_0 \in (B, <)'$ is again refined to $\mathcal{T}_1$ and $\mathcal{T}_2$. Again both trees are already elements of $(B, <)$ and no insertion or substitution, respectively, is made. Next, $\mathcal{T}_2$ is again refined to $\mathcal{T}_3$, $\mathcal{T}_4$, $\mathcal{T}_5$ and finally to $\mathcal{T}_6$ with $F_\text{h}(\mathcal{T}_3) = F_\text{h}(\mathcal{T}_4) = F_\text{h}(\mathcal{T}_5) = F_\text{h}(\mathcal{T}_6) = 4$. As $\mathcal{T}_3$ is already element of $(B, <)$ and $\mathcal{T}_4$, $\mathcal{T}_5$ and $\mathcal{T}_6$ do not perform better than $b_1 = \mathcal{T}_0$, no insertion or substitution, respectively, is made. Now $b_3 = \mathcal{T}_3$ is refined to $\mathcal{T}_4$, $\mathcal{T}_7$ and finally to $\mathcal{T}_8$. $\mathcal{T}_4$ with $F_\text{h}(\mathcal{T}_4) = 4$ is rejected. $F_\text{h}(\mathcal{T}_7) = 0$ substitutes $b_1 = \mathcal{T}_0$ and afterwards $(B, <)$ is sorted to $\{\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_1, \mathcal{T}_7\}$. $\mathcal{T}_8$ with $F_\text{h}(\mathcal{T}_8) = 4$ is rejected again. Finally $\mathcal{T}_2$ as the only refinement of $\mathcal{T}_1$ is rejected, as $b_1 = \mathcal{T}_2$. Iteration 3 ends with

$$\text{End of iteration 3: } (B, <) = \{\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_1, \mathcal{T}_7\}$$

Iteration 4 starts with setting $(B, <)' = (B, <)$. The refinement sequence of $b_1 = \mathcal{T}_2$ is given by $\mathcal{T}_3$, $\mathcal{T}_4$, $\mathcal{T}_5$ and $\mathcal{T}_6$ again, all having an error of 4. As $\mathcal{T}_3 \in (B, <)$ and $\mathcal{T}_4$, $\mathcal{T}_5$ and $\mathcal{T}_6$ do not perform better than $b_0 = \mathcal{T}_2$, the beam elements do not change. $b_2 = \mathcal{T}_3$ is refined to $\mathcal{T}_4$, $\mathcal{T}_7$ and $\mathcal{T}_8$. As $\mathcal{T}_4$ and $\mathcal{T}_8$ have an error of 4 and $\mathcal{T}_7$ is already element of $(B, <)$, the beam is not changed. Now $b_3 = \mathcal{T}_1$ is refined to $\mathcal{T}_2$, which is already element of the beam. Finally, $\mathcal{T}_7$ is refined. As can be deduced from the derivation sequence for $\mathcal{T}_7$ given in Table 3.4, the two possible refinements of $\mathcal{T}_7$ lead to the expressions $v_1 \times v_1 \times v_2$ and

$v_2 \times v_1 \times v_1$, respectively. As both expressions perform with an error of 4, none of the refinements of $\mathcal{T}_7$ is inserted into the beam and iteration 4 ends with

$$\text{End of iteration 4: } (B, <) = \{\mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_1, \mathcal{T}_7\}$$

Now it holds that $(B, <) = (B, <)'$ (Algorithm 3.8, line 25) and the algorithm stops. Presenting $\mathcal{T}_7$ as best performing beam element, the algorithm indeed detected the best solution in terms of a minimal error. However, this successful run was highly influenced by the sequence in which the refinements for each single beam element were generated, which will be explained in more detail below.

In the following, $(r(T), <^r)$ denotes an ordered set of parse tree refinements: Let $\mathcal{T}$ *be a parse tree and* $r(\mathcal{T}) = \{r_1, \ldots, r_n\}$ *be the set of all possible refinements of* $\mathcal{T}$. Then

Definition 9. $(r(T), <^r) : r_i \in r(T) <^r r_j \in r(T) :\Leftrightarrow r_i$ *is derived before* r_j *according to Statement 3.*

The critical step is found in the refinement of $\mathcal{T}_2$. According to Statement 3, $\mathcal{T}_2$ is refined to $(r(\mathcal{T}_2), <^r) = \{\mathcal{T}_3, \mathcal{T}_4, \mathcal{T}_5, \mathcal{T}_6)$. Let us arbitrarily assume that the refinement sequence was the other way around, i.e. $(r(\mathcal{T}_2), <^r) = \{\mathcal{T}_6, \mathcal{T}_5, \mathcal{T}_4, \mathcal{T}_3)$. Then we get the following processing sequence of the algorithm (only the important steps are listed):

$$\text{Start: } (B, <) = \{\mathcal{T}_0\}.$$

- Copy $(B, <)$ to $(B, <)'$
- Refine all $b_i \in (B, <)'$ to $(r(\mathcal{T}_0), <^r) = \{\mathcal{T}_1, \mathcal{T}_2\}$
  - Insert $\mathcal{T}_1$ into $(B, <)$ as beam is not filled and sort $(B, <)$
  - Insert $\mathcal{T}_2$ into $(B, <)$ as beam is not filled and sort $(B, <)$
- Check if $(B, <)$ equals $(B, <)'$: as it does not, start next iteration

$$\text{End of iteration 1: } (B, <) = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_1\}.$$

- Copy $(B, <)$ to $(B, <)'$
- Refine all $b_i \in (B, <)' = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_1\}$ and get
  - $(r(\mathcal{T}_0), <^r) = \{\mathcal{T}_1, \mathcal{T}_2\}$
  - Reject $\mathcal{T}_1$ and $\mathcal{T}_2$ as both are elements of $(B, <)$
  - $(r(\mathcal{T}_2), <^r) = \{\mathcal{T}_6, \mathcal{T}_5, \mathcal{T}_4, \mathcal{T}_3\}$
  - Insert $\mathcal{T}_6$ into $(B, <)$ as beam is not filled and sort $(B, <)$

- Reject $\mathcal{T}_5$, $\mathcal{T}_4$ and $\mathcal{T}_3$ as none performs better than $b_1 = \mathcal{T}_0$
- $(r(\mathcal{T}_1), <^r) = \{\mathcal{T}_2\}$
- Reject $\mathcal{T}_2$ as it is element of $(B, <)$

– Check if $(B, <)$ equals $(B, <)'$: as it does not, start next iteration

End of iteration 2: $(B, <) = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_6, \mathcal{T}_1\}$.

– Copy $(B, <)$ to $(B, <)'$

– Refine all $b_i \in (B, <)' = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_6, \mathcal{T}_1\}$ and get

- $(r(\mathcal{T}_0), <^r) = \{\mathcal{T}_1, \mathcal{T}_2\}$
- Reject $\mathcal{T}_1$ and $\mathcal{T}_2$ as both are elements of $(B, <)$
- $(r(\mathcal{T}_2), <^r) = \{\mathcal{T}_6, \mathcal{T}_5, \mathcal{T}_4, \mathcal{T}_3\}$
- Reject $\mathcal{T}_6$ as it is element of $(B, <)$
- Reject $\mathcal{T}_5$, $\mathcal{T}_4$ and $\mathcal{T}_3$ as none performs better than $b_1 = \mathcal{T}_0$
- $(r(\mathcal{T}_6), <^r)$: as can be deduced from the derivation sequence for $\mathcal{T}_6$ given in Table 3.4, 6 different refinements can be generated leading to the 4 different expressions $v_1 \times v_1 \times v_2$, $v_1 \times v_2 \times v_1$ and $v_2 \times v_1 \times v_1$ and, $v_1 \times v_1 \times v_1 \times v_1$. As all expressions perform with an error of 4, none of the refinements of $\mathcal{T}_6$ is inserted into the beam.
- $(r(\mathcal{T}_1), <^r) = \{\mathcal{T}_2\}$
- Reject $\mathcal{T}_2$ as it is element of $(B, <)$

– Check if $(B, <)$ equals $(B, <)'$: as $(B, <) = (B, <)'$ holds, the algorithm stops.

End of iteration 3: $(B, <) = \{\mathcal{T}_0, \mathcal{T}_2, \mathcal{T}_6, \mathcal{T}_1\}$.

What the two examples clearly show, is that the deterministic sequence of refinement generation can systematically lead to a good result – but also can systematically lead to a bad result. In subsequent iterations, the same refinements of a parse tree $\mathcal{T} \in (B, <)$ are generated repetitively in exactly the same sequence – just as in our example for $\mathcal{T}_3$. If a parse tree refinement $r_i \in (r(\mathcal{T}), <^r)$ is not inserted into the beam after evaluation and comparison of its error to the worst performing element of the actual valid beam, it cannot be included in future iterations. This is due to the fact that the error of this worst performing element can only be decreased but never increased. Consequently, if none of the refinements of a beam element b_i performs better than the worst performing of all existing beam elements, $\mathcal{T}$ will be kept as beam element until itself becomes the worst performing element and is finally substituted. Therefore, only the trace of beam elements producing

refinements of better performance in each iteration will be followed. As soon as no such refinement exists, the beam element is "lost" for further investigation. This even augments the effects of hitting local minima (drawback (i) on page 51) and cutting the hypothesis space (drawback (ii) on page 55): as long as this one beam element remains in the beam, the same refinements are evaluated over and over again and the derivation trace starting from $\mathcal{T}$ is at least temporarily cut.

In our simple example this systematic behavior is aggravated by the artificial situation that several refinements have identical errors. However, this setting is not the source of the problem as it remains even if the errors of the refinements of the beam elements differ, but do not perform better than the worst performing beam element. Another point to be mentioned is that in the system implementation, the elements of the sorted beam $(B, <)$ are refined in a left to right sequence (Algorithm 3.8, line 7). But in contrast to the sequential application of ordered production rules (Definition 5) or the sequential refinement of a parse tree (Statement 3), this has minor impact on the system performance as shown in Appendix C.

We conclude that it is necessary to decouple the search heuristic for traversing the hypothesis space from the presentation of the background knowledge and therefore the aspect of randomization should be considered.

3.3 Methodological Background: the GSAT Algorithm

The GSAT algorithm [100, 101] was developed to solve propositional satisfiability (SAT) problems of propositional formulas in conjunctive normal form (CNF). SAT had been shown to be a NP-hard task [20]. A propositional formula in CNF consists of a conjunction ($\wedge$) of disjunctions ($\vee$) of literals, i.e. a conjunction of clauses. A literal is a propositional variable (or atom) A or its negation $\neg A$. A clause is defined as a disjunction of literals which can be multi-digit, e.g. $(A \vee \neg B \vee C)$, single-digit, e.g. A, or an empty clause $\square$. Examples for propositional formulas in CNF are $(A \vee \neg B) \wedge (B \vee \neg C \vee \neg D)$, $A \vee B$, $A \wedge B$, $A \wedge (B \vee C)$, 1, 0. GSAT performs as a local greedy search. It iteratively processes two steps (see Algorithm 3.9):

1. A truth assignment for the clauses in the formula is randomly generated. This step is repeated up to a maximum number of tries (input variable $MAX\text{-}TRIES$).
2. After each randomly generated truth assignment, the assignment of each literal is changed ("flipped"). The algorithm then determines which variable flip leads to the largest increase in the total number of satisfied clauses in the formula and keeps this assignment. Such flips are repetitively applied until either a satisfying truth assignment of the formula is found or a maximum number of flips ($MAX\text{-}FLIPS$) is reached.

Algorithm 3.9 GSAT algorithm for solving propositional satisfiability problems. (This algorithm refers to the work of Selman *et al.* ([100], figure 1).)

procedure GSAT(α,$MAX\text{-}TRIES$,$MAX\text{-}FLIPS$)

```
1: for i := 1 to MAX-TRIES do
2:   T := a randomly generated truth assignment
3:   for j := 1 to MAX-FLIPS do
4:     if T satisfies α then
5:       return T
6:     else
7:       p := a propositional variable such that a change (flip) in its truth assignment
         gives the largest increase in the total number of clauses of α that are satisfied
         by T
8:       T := T with the truth assignment of p changed
9:     end if
10:  end for
11: end for
12: return "no satisfying truth assignment found"
```

α	set of clauses
$MAX\text{-}TRIES$	maximum number of iterations of randomly generating truth assignments for α. After each such random assignment the assignment of all literals is flipped.
$MAX\text{-}FLIPS$	maximum number of flips of the assignments of all literals in α

The performance of GSAT has been compared to the Davis-Putnam algorithm (DP) [23]: "DP is in essence a resolution procedure [...]. It performs a backtracking search in the space of all truth assignments, incrementally assigning values to variables and simplifying the formula. If no new variable can be assigned a value without producing an empty clause, it backtracks. The performance of the basic DP procedure is greatly improved by using *unit propagation* whenever unit clauses arise (a unit clause is a clause that contains a single literal): variables occurring in unit clauses are immediately assigned the truth value that satisfies the clause, and the formula is simplified, which may lead to new unit clauses, etc. This propagation process can be executed quite efficiently (in time linear in the total number of literals). DP combined with unit propagation is one of the most widely used methods for propositional satisfiability testing." (Bart Selman *et al.* [100], 1992, p. 441). The authors state that for all satisfiable formulas in the test set which was found by DP, GSAT succeeded as well. It had been expected that GSAT would frequently hit some local minima where some clauses remain unsatisfied, which apparently was not the case. Moreover, GSAT has been found to perform substantially faster that DP. The algorithm handles much larger formulas (up to 500 variables) than DP (up to around 40 variables). GSAT has also been found to perform effectively in applications such as graph coloring, N-queens encoding and Boolean induction problems.

Revealing the above mentioned drawbacks in terms of hitting local minima (section 3.2.4, *Drawbacks (i)*, page 51), cutting the hypothesis space (section 3.2.4, *Drawbacks (ii)*, page 55) and the repetitive identical parse tree refinements (section 3.2.4, *Drawbacks (iii)*, page 61) in context of the representation of the background knowledge when traversing the hypothesis space, we found that using the randomized search logic of GSAT might essentially improve the performance of the system. Not in the sense of optimizing the system by accelerating the search progress, but by enabling the system to avoid local optima and to approach global optima by injecting randomness.

3.4 Materials and Methods

With respect to the drawbacks as described previously, the ED system LAGRAMGE was modified. In this section, the modifications are described. Experiments performed for the validation of the modifications, the used data and the results are presented. Finally, experiments to re-identify the equation of motion model (section 1.2.4) on real-world data are performed and presented.

3.4.1 System Modifications

The aim was to avoid the additional bias induced by the sequential application of production rules and the specific refinement of parse trees (based on the sequential traversal of the nonterminals, i.e. parse tree nodes). Therefore, the refinement of the parse trees as well as the processing of the top-level beam search was modified. We found that the method of choice was randomization. Following GSAT, the improvement of a model is processed in two steps, firstly by modification of the parse tree refinement and secondly by the iterative evaluation of the heuristic function. In a third step, the superordinate beam search is re-organized. Finally, redundant parse tree evaluations are avoided.

Modification 1: Randomized Parse Tree Refinement

For the following paragraph please see Algorithms 3.10 and 3.11. Referring to GSAT, the initial randomly generated truth assignment of the variables of a Boolean formula is mimicked as followed: instead of choosing the production rules for the refinement of a parse tree $\mathcal{T}$ and the termination of its subtrees in the predefined sequential order (Algorithm 3.2 and Algorithms 3.5 to 3.7), these rules are selected randomly. Instead of terminating the subtrees $\mathcal{T}_\mathrm{A}$ for all nonterminals A in the parse tree $\mathcal{T}$ by application of the first successor rule $p_{A,i+1}$ of the previously applied rule $p_{A,i}$ (Algorithm 3.10, line 1), the succeeding rule $p_{A,j} \in P_A$ is randomly chosen (Algorithm 3.10, line 4). The in this way derived subtree

Algorithm 3.10 Procedure *RandomRefinementOperator* refines parse tree by random application of production rules.

procedure *RandomRefinementOperator* ($\mathcal{T}$, A, h_{max})

1: let $p_{A,i} \in P_A$ be the production rule actually applied at A
2: let $dstep_j(\mathcal{T})$ be the derivation step which generates the expression in which A is the most left nonterminal
3: delete the subtrees of node A
4: randomly choose production rule $p_{A,j} \neq p_{A,i} \in P_A$ with $l(A) + h(p_{A,j}) \leq h_{max}$// overall derivation length must not exceed h_{max} by application of $p_{A,j}$
5: expand node A with all nonterminals and terminals on the right side of $p_{A,j}$ to $\mathcal{T}_\mathrm{A}$ as $dstep_{j+1}(\mathcal{T})$
6: **repeat**
7: apply $RandomNextDerivationStep(\mathcal{T}, dstep_{j+1}(\mathcal{T}), h_{max})$// randomly choose derivation step and apply, see Algorithm 3.11
8: **until** all leaf nodes in $\mathcal{T}_\mathrm{A}$ are terminals

$\mathcal{T}$	parse tree with nonterminal start node S as root
A	nonterminal node in parse tree $\mathcal{T}$
h_{max}	maximum depth of derived parse tree
P_A	set of production rules with A on the left-hand side
$dstep_j(\mathcal{T})$	j-th derivation step applied to derive $\mathcal{T}$ (Definition 6)
$l(A)$	length of path from A to the root node of $\mathcal{T}$
$h(p)$	minimal height of parse tree with production p applied to its root node (Definition 1)
$\mathcal{T}_\mathrm{A}$	parse tree with nonterminal A as root node

Algorithm 3.11 Procedure for application of the successor of the i-th derivation step $dstep_i(\mathcal{T})$, which has been applied to derive a parse tree $\mathcal{T}$ with random choice of the production rule.

procedure $RandomNextDerivationStep(\mathcal{T}, dstep_i(\mathcal{T}), h_{max})$

1: $E := e(dstep_i(\mathcal{T}))$
2: let A_i be the most left nonterminal in E
3: randomly choose production rule $p_{A,j} \neq p_{A,i}$ with $l(A) + h(p_{A,j}) \leq h_{max}$// overall derivation length must not exceed h_{max} by application of $p_{A,j}$
4: expand node A with all nonterminals and terminals on the right side of $p_{A,j}$ as $dstep_{i+1}(\mathcal{T})$

$\mathcal{T}$	parse tree with nonterminal start node S as root
$dstep_i(\mathcal{T})$	i-th derivation step applied to derive $\mathcal{T}$ (Definition 6)
h_{max}	maximum height of derived parse tree $\mathcal{T}$
E	expression consisting of nonterminal and terminal symbols, $E := (N \cup T)^*$
$e(dstep_i(\mathcal{T}))$	expression E derived after derivation step $dstep_i(\mathcal{T})$ (Definition 7)
$l(A)$	length of path from A to the root node of $\mathcal{T}$

Algorithm 3.12 Procedure *ImproveRefinement* tries to improve a parse tree $\mathcal{T}$ with respect to the heuristic function $\mathcal{F}_\mathrm{h}$. All valid production rules are applied to all nonterminal nodes in $\mathcal{T}$. After each such application of a production rule, the resulting tree is terminated (no more nonterminals in leaf nodes) by random choice of production rules. The parse tree with the lowest error according to $\mathcal{F}_\mathrm{h}$ substitutes $\mathcal{T}$.

procedure $ImproveRefinement(\mathcal{T},\, h_{max},\, \mathcal{F}_\mathrm{h})$

1: $\mathcal{T}' := \mathcal{T}, \mathcal{T}'' := \mathcal{T}$// create working copy $\mathcal{T}'$ and template $\mathcal{T}''$ for re-use
2: **for all** nonterminal nodes A in $\mathcal{T}'$ **do** // to all nonterminal nodes
3: **for all** production rules $p_{A,i} \in P_A$ **do** // apply all possible production rules
4: apply $p_{A,i}$ to A as $dstep_j(\mathcal{T}')$
5: **repeat**
6: apply $RandomNextDerivationStep(\mathcal{T}', dstep_j(\mathcal{T}'), h_{max})$// see Algorithm 3.11
7: **until** all leaf nodes in $\mathcal{T}'$ are terminals
8: **if** $\mathcal{T}' \notin \mathcal{T}_{history}$ **then** // if tree was not evaluated previously
9: **if** $\mathcal{F}_\mathrm{h}(\mathcal{T}') < \mathcal{F}_\mathrm{h}(\mathcal{T})$ **then** // if error of $\mathcal{T}'$ is lower than error of origin $\mathcal{T}$
10: $\mathcal{T} := \mathcal{T}'$// replace origin $\mathcal{T}$ by $\mathcal{T}'$ in beam
11: **end if**
12: insert $\mathcal{T}'$ into $\mathcal{T}_{history}$// historicize evaluated tree to avoid re-evaluation
13: **end if**
14: **for all** nonterminal nodes B in $\mathcal{T}'$ **do**
15: apply $RandomRefinementOperator(\mathcal{T}',\, B,\, h_{max})$// see Algorithm 3.10
16: **if** $\mathcal{T}' \in \mathcal{T}_{history}$ **then** // check if refined tree has been evaluated previously
17: continue// evaluate next nonterminal node B
18: **else if** $\mathcal{F}_\mathrm{h}(\mathcal{T}') < \mathcal{F}_\mathrm{h}(\mathcal{T})$ **then** // if refined tree has lower error rate than origin
19: $\mathcal{T} := \mathcal{T}'$// replace origin $\mathcal{T}$ by $\mathcal{T}'$ in beam
20: **end if**
21: Insert $\mathcal{T}'$ into $\mathcal{T}_{history}$// historicize newly evaluated tree to avoid re-evaluation
22: **end for**
23: **end for**
24: $\mathcal{T}' := \mathcal{T}''$// reset $\mathcal{T}'$ to origin for next iteration
25: **end for**

$\mathcal{T}$	parse tree with nonterminal start node S as root
h_{max}	maximum depth of derived parse tree
$\mathcal{F}_\mathrm{h}$	heuristic function
P_A	set of production rules with A on the left-hand side
$p_{A,i}$	production rule with A on the left-hand side
$\mathcal{T}_{history}$	(limited) set of previously derived parse trees

Algorithm 3.13 Procedure *RandomLAGRAMGE* processes a beam search of width w_b with randomized choice of production rules for refining the parse trees.

procedure *RandomLAGRAMGE* $(w_b, B, h_{max}, \mathcal{F}_\mathrm{h}, \mathcal{S})$

1: **for** $i := 1$ to w_b **do** // initialize beam
2: $\mathcal{T} := S$// set start node as initial tree
3: apply *RandomRefinementOperator*($\mathcal{T}$, S, h_{max})// randomly refine initial tree, see Algorithm 3.10
4: **while** $\mathcal{T} \in \mathcal{T}_{history}$ **do** // do not insert identical parse trees
5: apply *RandomRefinementOperator*($\mathcal{T}$, S, h_{max})
6: **end while**
7: insert $\mathcal{T}$ into B
8: insert $\mathcal{T}$ into $\mathcal{T}_{history}$// historicize tree to avoid re-evaluation
9: **end for**
10: **repeat**
11: $B' := B$// copy beam for check of termination condition
12: **for all** $\mathcal{T} \in B$ **do**
13: $\mathcal{T}' := \mathcal{T}$// copy tree for error comparison
14: randomly choose nonterminal A in $\mathcal{T}'$
15: apply *RandomRefinementOperator*($\mathcal{T}'$, A, h_{max})
16: **if** $\mathcal{T}' \notin \mathcal{T}_{history}$ and $F_\mathrm{h}(\mathcal{T}') < F_\mathrm{h}(\mathcal{T})$ **then**
17: $\mathcal{T} := \mathcal{T}'$
18: insert $\mathcal{T}$ into $\mathcal{T}_{history}$
19: **end if**
20: apply *ImproveRefinement* ($\mathcal{T}$, h_{max}, $\mathcal{F}_\mathrm{h}$)// try to improve $\mathcal{T}$ by random refinements, see Algorithm 3.12
21: **end for**
22: **until** $B == B'$ or $\mathcal{S}$// until no more beam elements have changed or stopping criterion is fulfilled

w_b	beam width
B	set of beam elements $B = \{b_1, b_2, \ldots, b_{w_b}\}$
h_{max}	maximum depth of derived parse tree
$\mathcal{F}_\mathrm{h}$	heuristic function
$\mathcal{S}$	stopping criterion (none, timelimit[sec])
S	nonterminal start symbol
$\mathcal{T}$	parse tree with nonterminal start node S as root
$\mathcal{T}_{history}$	(limited) set of previously derived parse trees

$\mathcal{T}_\mathrm{A}$ is terminated by iteratively applying a randomly chosen production rule $p_{B,i} \in P_B$ to all nonterminal leaf nodes B in $\mathcal{T}_\mathrm{A}$ until no more nonterminal leaf nodes are left and $\mathcal{T}$ is completely refined to $\mathcal{T}'$ (Algorithm 3.10, lines 6 to 8).

Modification 2: Optimized Successor Selection

For the following paragraph please see Algorithm 3.12. Again referring to GSAT, the step of flipping the variable assignment with the objective of finding the variable with the largest increase in the total number of satisfied clauses is mimicked. In the first step a parse tree $\mathcal{T}$ has been randomly refined to $\mathcal{T}'$ (see above). Now, in a second step, for this refined parse tree $\mathcal{T}'$, the system tries to minimize $\mathcal{F}_\mathrm{h}(\mathcal{T}')$ (Algorithm 3.13, line 20): successively for all nonterminals A in $\mathcal{T}'$ (Algorithm 3.12, line 2), all $p_{A,i} \in P_A$ are applied (Algorithm 3.12, line 3). Now, whenever a subtree $\mathcal{T}_\mathrm{A}$ is derived by application of a production rule $p_{A,i}$, this subtree $\mathcal{T}_\mathrm{A}$ is randomly refined again (Algorithm 3.12, lines 5 to 7) until all leaf nodes consist of terminal symbols. After each of such a random refinement, $\mathcal{F}_\mathrm{h}(\mathcal{T}')$ is evaluated (Algorithm 3.12, line 9). If $\mathcal{F}_\mathrm{h}(\mathcal{T}') < \mathcal{F}_\mathrm{h}(\mathcal{T})$, then $\mathcal{T}$ is substituted by $\mathcal{T}'$. Whether $\mathcal{T}'$ performs better than $\mathcal{T}$ or not, for all nonterminals B in $\mathcal{T}'$ a random refinement is performed with B as root node (Algorithm 3.12, lines 14-15). Each such refined tree $\mathcal{T}'$ is tested against $\mathcal{T}$ in terms of $\mathcal{F}_\mathrm{h}$. If $\mathcal{T}'$ performs better than $\mathcal{T}$, $\mathcal{T}$ is substituted by $\mathcal{T}'$ (Algorithm 3.12, lines 18-19). Note that the evaluation of $\mathcal{F}_\mathrm{h}$ is not performed if the refined tree has been evaluated before (Algorithm 3.12, lines 8, 16).

Modification 3: Reorganized Beam Search

For the following paragraph please see Algorithm 3.13. In the original system implementation each single refinement of a beam element $b_i = \mathcal{T}$ is compared to all beam elements (as the beam is sorted and the refinement is compared to the worst performing beam element b_1) and substitutes the worst performing element b_1 if required (Algorithm 3.8, lines 19 to 21). (In an extreme case all beam elements are replaced by the successor refinements of one single element and the traces of all other elements are cut.) In the modified version each single beam element $b_i = \mathcal{T}$ is randomly refined to $\mathcal{T}'$ (Algorithm 3.13, lines 12 to 15) and the heuristic function is evaluated as $F_\mathrm{h}(\mathcal{T}')$ (if $\mathcal{T}'$ was not evaluated before, Algorithm 3.13, line 16). Now the performance in terms of F_h of the refinement $\mathcal{T}'$ is compared solely to the actually evaluated beam element $\mathcal{T}$ as origin of $\mathcal{T}'$ and $\mathcal{T}$ is replaced by $\mathcal{T}'$ if its error could be reduced (Algorithm 3.13, lines 16 to 17). As the algorithm does not stop before none of the beam elements $\mathcal{T}_{1..w_b}$ could be improved (or none of the elements can be refined anymore or the stopping criterion $\mathcal{S}$ is met) a beam element is tried to be improved in each new iteration $i+1$ even if it was not improved in the previous iteration i (Algorithm 3.13, line 22).

Modification 4: Minimized Redundancies

For the following paragraph please see Algorithm 3.13 and 3.12. Derived parse trees, i.e. all derivation steps to generate a model expression (Statement 1, page 58) are historicized (Algorithm 3.13, lines 8, 18; Algorithm 3.12, lines 12, 21). (As this might cause a high demand on memory, the number of stored parse trees is limited.) To avoid redundant model evaluations, the system checks if a parse tree $\mathcal{T}$ has been derived previously by checking the entries in this history (Algorithm 3.13, lines 4, 16; Algorithm 3.12, lines 8, 16). In this way, not only the elements of the actual beam are checked for redundancies as in the original algorithm (Algorithm 3.8, line 11), but also the history of previously derived parse trees of all beam elements.

Strictly speaking, this is not a beam search in its pure specification any more. E.g. it is no longer a breadth-first search used for generating search trees, beam elements are no longer sorted by the costs according to the heuristic function and a refined beam element solely does replace its (random) successor and is not compared to all other beam elements in terms of the error. However, as still a predefined number of candidates is kept for improvement and the modified successor selection still identifies the "most promising" amongst them, we stick to this wording.

3.4.2 System Validation: Patients and Datasets

The data for this study were obtained by automated PEEP wave maneuvers as described in section 1.2.5 (Figure 1.5). The study included data of thirteen patients of the control group (section 1.3.2, Figure 1.7). For analysis, the last breathing cycle before increasing the PEEP to the next pressure level was extracted from the data (Figure 3.3.a and b). Thus, each patient dataset consisted of 11 to 14 (13.7 $\pm$ 0.95 [mean $\pm$ SD]) breathing cycles at different PEEP levels. The input datasets consisted of time series data for the four domain variables P_{aw}, $\dot{V}$, V and $PEEP$ with P_{aw} as dependent variable. The PEEP level was given as constant value during a single breathing cycle (Figure 3.4).

3.4.3 System Validation: Benchmark Tests

The benchmark test to validate the modified system was to identify the EOM model (section 1.2.4) in the respiratory data. The performance of the modified system was tested against the original algorithm (section 3.2). Special attention was paid to the effect of different input sequences of the domain variables and production rules (section 3.2.4).

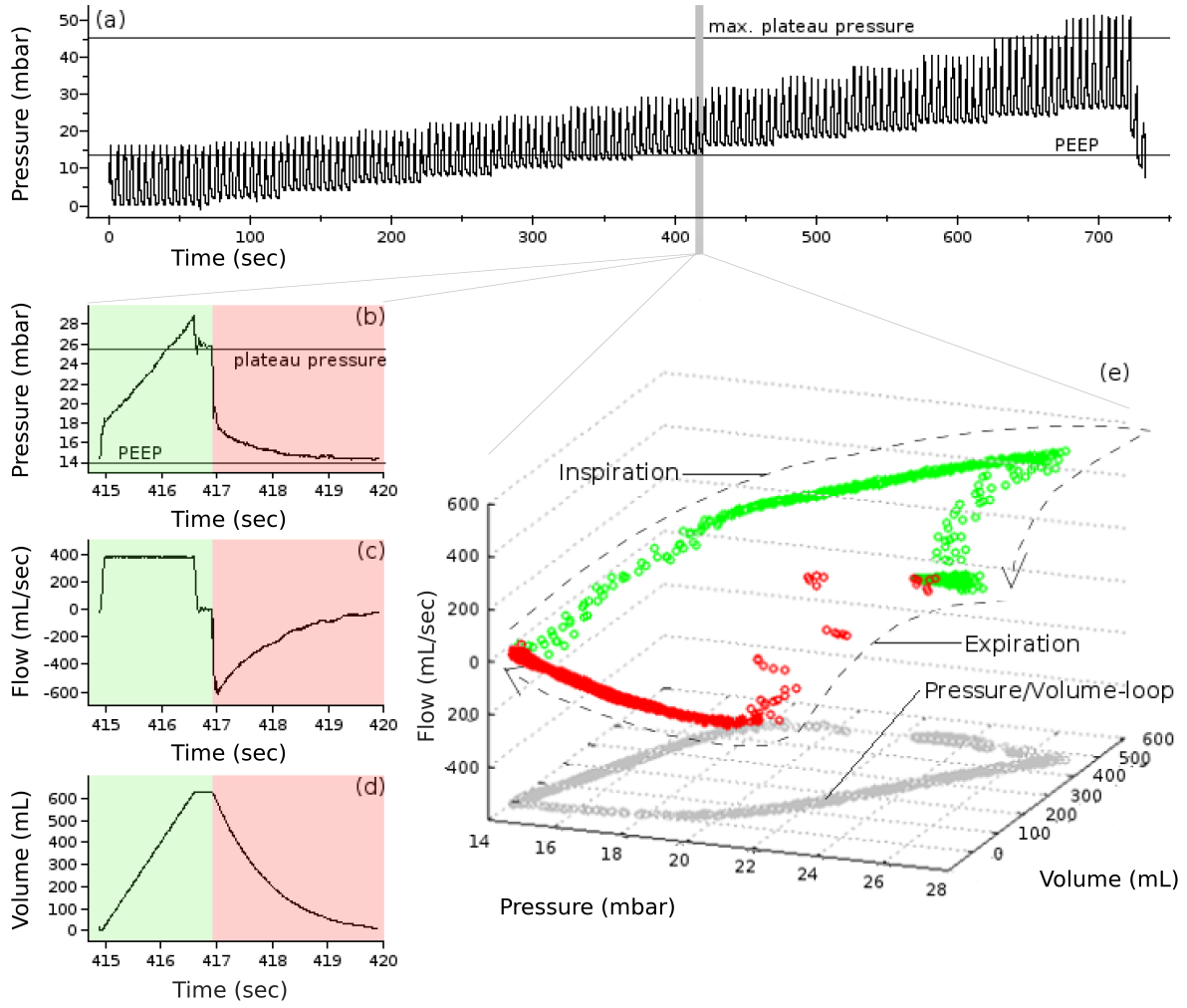

Figure 3.3: PEEP wave maneuver data sample of one patient. (a) PEEP wave maneuver with successively increasing PEEP level in steps of 2 mbar up to a maximum plateau pressure of 45 mbar. (b) Sample pressure curve of one breath at PEEP level of 14 mbar. As it is the last breath before the next change of PEEP, this breath was used for analysis. The plateau pressure level is approximated at the end of the zero flow phase after inflation. (c) and (d) depict the corresponding flow and volume curve, respectively. (e) illustrates the correspondence of pressure (b), flow (c) and volume (d) in a 3D-plot. The green limb of the curve represents the inspiratory part, the red limb the expiratory part. The grey-colored loop on the 2D pressure-volume plane depicts the pressure-volume loop as presented schematically in Figure 1.1.

Figure 3.4: Sample time series data consisting of pressure, volume, flow and PEEP-offset of 14 mbar.

Paw	Vol	Flow	PEEP	
⋮	⋮	⋮	⋮	
14.4	21	-0.0416667	14	;
14.4	21	-0.0416667	14	;
14.4	21	-0.0416667	14	;
14.4	20	-0.0416667	14	;
14.4	20	-0.0383333	14	;
14.4	20	-0.0416667	14	;
14.4	19	-0.0366667	14	;
14.4	19	-0.0400000	14	;
14.4	0	-0.0350000	14	;
15.4	0	0.0316667	14	;
16.1	0	0.0000000	14	;
16.3	1	0.1300000	14	;
16.3	2	0.1250000	14	;
16.3	2	0.1250000	14	;
⋮	⋮	⋮	⋮	

Benchmark Test (i): Identification of the EOM

A valid model equation should represent the EOM model, i.e re-identify the EOM from the data. A model was assumed to represent the EOM model, if after simplification the resulting equation consisted of the three additive terms $const \times V/C$, $const \times R \times \dot{V}$ and $const \times PEEP$ (Equation (1.3)) with optional constant multipliers *const*. Additional constant additive terms were allowed to be included and thus a positively identified model was of the form

$$P = [const \times]\ 1/C\ \times\ V\ +\ [const \times]\ R\ \times\ \dot{V}\ +\ [const \times]\ PEEP\ [+\ const] \quad (3.2)$$

Note that *const*, C and R could take both positive or negative values, but had to lead to pure positive additive terms aside from the additional optional constant additive terms [+ const].

Benchmark Test (ii): RMSE of the Model Fits

Besides the structure of a model equation represented by a parse tree $\mathcal{T}$, the second benchmark was determined by the value of the heuristic function $F_{\mathrm{h}}(\mathcal{T})$ when fitting a model

initial input sequence	*reverse input sequence*
$E \rightarrow E+F \mid E-F \mid F$	$E \rightarrow F \mid E-F \mid E+F$
$F \rightarrow F \times I \mid F \div I \mid I$	$F \rightarrow I \mid F \div I \mid F \times I$
$I \rightarrow const \mid v \mid (E)$	$I \rightarrow (E) \mid v \mid const$

Figure 3.5: Input sequences of production rules for the universal grammar G_{U} (Figure 3.1) with the original sequence on the left and the reversed sequence on the right.

to the data. Here, F_{h} was given by the RMSE of the model fits (Equation (3.1)). Consequently, the RMSE of the model fit to the raw data should be minimized.

Benchmark Test (iii): Correlation between Input Sequences and Tests (i)/(ii)

To identify the statistical correlation between the input sequence and the quality of the analysis results, a two-factorial analysis of variance (ANOVA) was performed. For both systems (original and modified) it was tested if (i) the identification of the EOM and (ii) if the RMSE of the provided model solutions depended on the input sequence of the variables and production rules. The median values of the RMSE calculated over all experimental runs were compared by a Wilcoxon rank-sum test. The significance level was set to $p < 0.05$ for "significant" and $p < 0.003$ for "highly significant".

3.4.4 System Validation: Experimental Setup of the Input

The input sequence of the independent variables (section 3.2.1), i.e. their listing in the data files (section 3.2.2), was permuted and thus, for each patient the six sequences

$$(\dot{V}, V, Peep),\ (\dot{V}, Peep, V),\ (V, \dot{V}, Peep),\ (V, Peep, \dot{V}),\ (Peep, \dot{V}, V),\ (Peep, V, \dot{V})$$

were provided. The input grammar consisted of a universal grammar, allowing to apply the four basic mathematical operands $+$, $-$, $\times$ and $\div$ (Figure 3.1). While a parse tree depth of five would have been sufficient to derive the EOM model, resulting in a search space size of 7300, we allowed a maximum parse tree depth of six, increasing the search space complexity to 14,674,005 different parse trees [111]. The production rules were provided in an initial sequence and its reverse $[initial, reverse]$ (Figure 3.5). Consequently, twelve different input combinations of variable sequences in the data files and production rule sequences in the grammar were analyzed for each of the 13 patient datasets.

To avoid a deterioration of the system's performance of the original system caused by a small beam width (section 3.2.4, page 51), the beam width was set to 20. This was the

largest possible in terms of the memory requirements.

Data preprocessing and statistical analysis were performed by application of the Matlab® software package version R2006b (The MathWorks, Natick, MA). The two versions of the applied ED systems were implemented in the C programming language. Experiments were run on a standard laptop computer (1001.7 MiB memory, 1.83 GHz processor) running under the operating system Ubuntu (version 8.04).

3.5 Results

3.5.1 Benchmark Test (i)/(iii): Identification of the EOM

The original system identified the EOM in 37.2%, the modified system in 96.8% of all runs. For the original system, the two-factorial ANOVA revealed a highly significant dependence of the identification rate from the input sequence of the domain variables as well as from the input sequence of the production rules. The interaction between these two factors was found to be statistically significant. In contrast, the modified system did not show such dependencies. The original system showed a higher variance of the results (Figure 3.6, Table 3.5).

3.5.2 Benchmark Test (ii)/(iii): RMSE of the Model Fits

A statistically highly significant dependence of the RMSE on the input sequence of the production rules was observed for the original system. The dependence of the RMSE on the input sequence of the domain variables as well as the interaction between the two factors was found to be statistically significant. The modified system did not show such dependencies. The original system showed a higher variance of the RMSE (Figure 3.7, Table 3.6). The modified system generated more precise models with respect to the RMSE in the early as well as in the final state of the iterative model derivation process (Figure 3.8). The maximum number of iteration steps amounts to 27 for the original and to 23 for the modified system. The medians of the RMSE are statistically significantly lower in the modified system (Figure 3.8, insert).

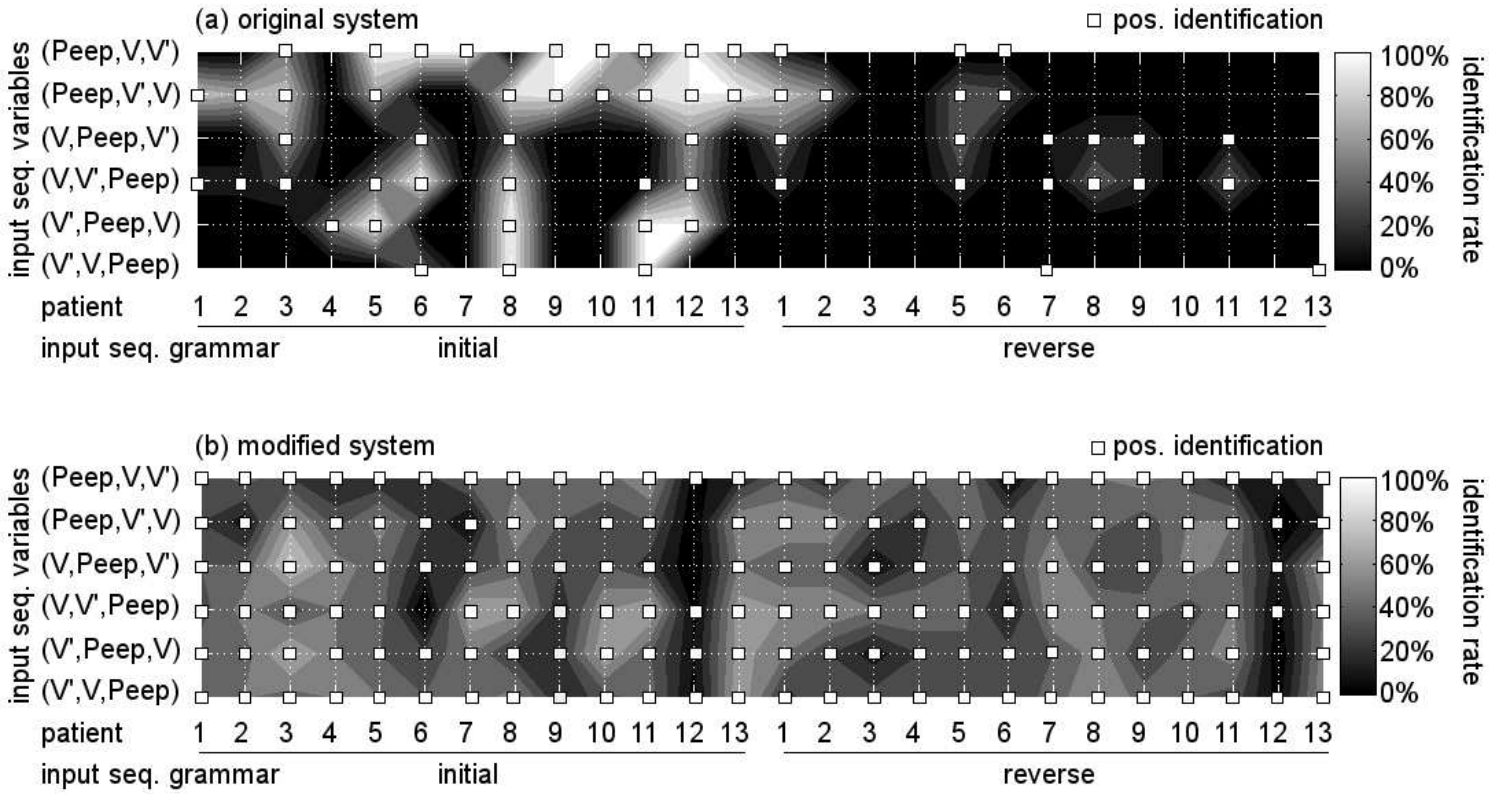

Figure 3.6: Benchmark test (i): experimental results for (a) original and (b) modified system with respect to identification of the EOM in the data. The *identification rate* indicates the percentage of beam elements (i.e. models) representing the EOM upon termination of the algorithm. The contour plot (interpolation for visualization purposes) represents the *identification rate* for all thirteen patients with respect to the input sequences of the domain variables and the production rules. The square markers indicate that at least one beam element provides a *positive identification* of the EOM. The dependence of the original system on the input sequences is indicated by the far more inhomogeneous gray scale coloration. Some peaks can be clearly identified within the range of the initial input sequence of the production rules. Note that 1. a darker color indicates a lower *identification rate*, 2. the interpolation is exclusively for better visualization, 3. adjacency of entries on the x- and y-axis is arbitrary as they merely represent the datasets and sequential orders, respectively.

Table 3.5: Results benchmark test (i)/(iii): two-factorial ANOVA with respect to the identification rate for original and modified system both run with beam width twenty. *factor 1* represents the sequential input of the independent domain variables in the data file, *factor 2* represents the input sequence of the production rules within the grammar. For each of the twelve input sequence combinations and each of the thirteen patients it was evaluated if at least one of the twenty models given in the result set (i.e. beam) represented the EOM upon termination of the algorithm. These positive detections were tested against *factor 1* and *factor 2*.

source	*sse*	*df*	*mse*	*F*	*p*
original system					
$factor1$	3.13	5	0.63	3.20	0.0090
$factor2$	2.56	1	2.56	13.11	0.0004
interaction	2.59	5	0.52	2.65	0.0253
error	28.15	144	0.20		
total	36.44	155			
modified system					
$factor1$	0.11	5	0.02	0.68	0.6393
$factor2$	0.01	1	0.01	0.20	0.6554
interaction	0.11	5	0.02	0.68	0.6393
error	4.62	144	0.03		
total	4.84	155			

source: source of variability; *sse*: sum of squared errors; *df*: degree of freedom associated with source; *mse*: mean squared error, i.e *sse*/*df*; *F*: F-value; *p*: p-value.

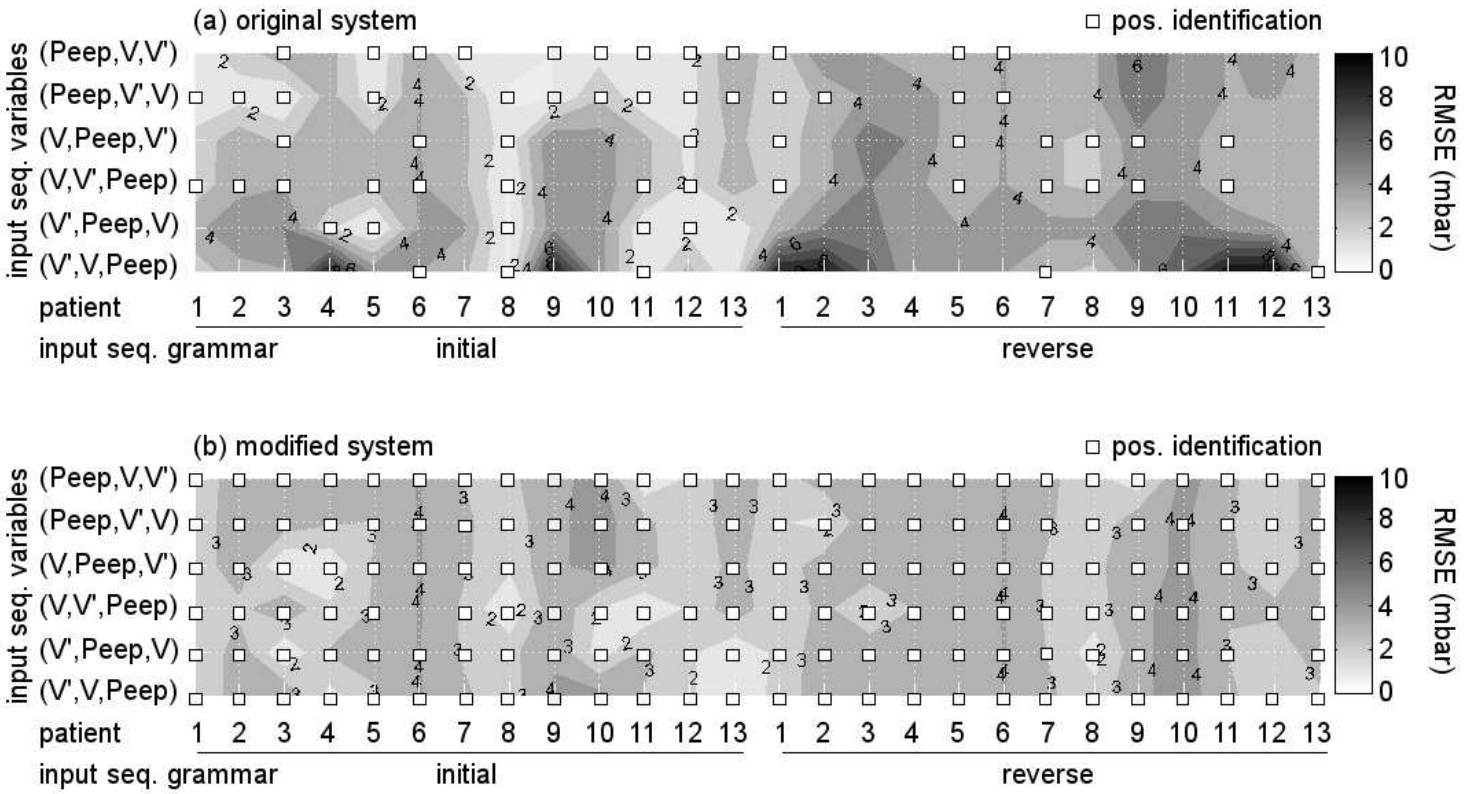

Figure 3.7: Benchmark test (ii): experimental results for (a) original system and (b) modified system with respect to the error variance of the generated models. The error variance is represented by the median RMSE of all beam elements (i.e. models) upon termination of the algorithm. The contour plot (interpolation for visualization purposes) represents this median RMSE for all thirteen patients with respect to the input sequences of the domain variables and the production rules. The square markers indicate that at least one beam element provides a *positive identification* of the EOM. The independence of the modified system on the input sequences is indicated by the more homogeneous gray scale coloration. Note that 1. a brighter color indicates a lower RMSE, 2. the interpolation is exclusively for better visualization, 3. adjacency of entries on the x- and y-axis is arbitrary as they merely represent the datasets and sequential orders, respectively.

Table 3.6: Results benchmark test (ii)/(iii): two-factorial ANOVA with respect to the RMSE of the models for original and modified system. Both runs were performed with beam width twenty: *factor 1* represents the sequential input of the independent domain variables in the data file, *factor 2* represents the input sequence of the production rules within the grammar. For each of the twelve input sequence combinations and each of the thirteen patients the median was calculated over the RMSE of all twenty models given in the result set (i.e. beam) upon termination of the algorithm. The medians were tested against *factor 1* and *factor 2*.

source	*sse*	*df*	*mse*	*F*	*p*
original system					
$factor1$	1.63	5	0.33	3.78	0.0030
$factor2$	3.22	1	3.22	37.36	0.0000
interaction	1.30	5	0.26	3.02	0.0126
error	12.40	144	0.09		
total	18.54	155			
modified system					
$factor1$	0.14	5	0.03	0.92	0.4702
$factor2$	0.01	1	0.01	0.40	0.5277
interaction	0.05	5	0.01	0.32	0.8997
error	4.52	144	0.03		
total	4.72	155			

source: source of variability; *sse*: sum of squared errors; *df*: degree of freedom associated with source; *mse*: mean squared error, i.e sse/df; *F*: F-value; *p*: p-value.

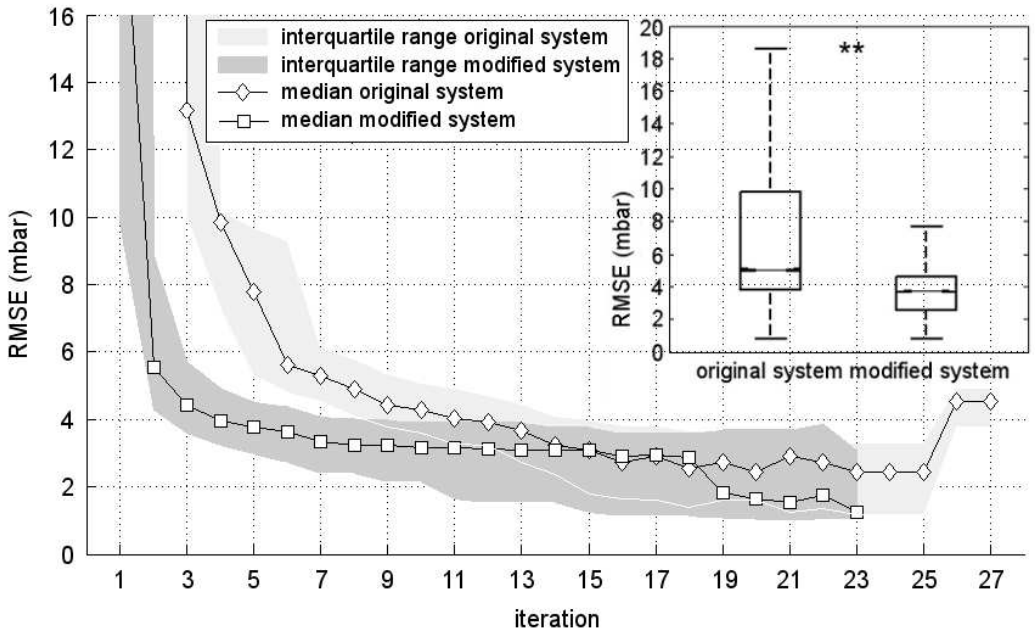

Figure 3.8: RMSE during the course of iterations: the RMSE is calculated as the median RMSE over all beam elements after each iteration. It is based on the results for all patients and all twelve input sequence combinations (grammar and domain variables). Note that the increase of the median RMSE for the original system at the end of the presented curve is caused by the modality of calculating the median over the results for all patients and input sequences, only separated by the iteration step. (During a single run, the algorithm would stop before an increase of the RMSE, i.e. a worsening of the model fit.) *insert*: Median of the RMSE for the original and the modified system calculated over all results for all patients and independent of the input sequences. As a Wilcoxon rank-sum test showed, the medians differ with high statistical significance (** $p < 0.003$).

3.5.3 General Performance

In about 97% of all test runs (i.e. different input combinations with respect to the input sequence of the variables and the grammar) the modified system identified the equation of motion in at least one of the 20 beam elements whereas the original system managed to do so in about 37% of all runs. Within a test run, on average the original system performed about 20% more iterations than the modified system. The modified system evaluated 9 times as many model equations as the original system. About 24% of the model equations generated by the modified system were redundant derivations and thus not re-evaluated. For each run, the modified system generated about 12 times as many model equations and needed about 26 times as much time as the original system (Table 3.7).

Table 3.7: General performance results of original and modified system. The table presents the percentage of positive detections of the equation of motion (EOM) over all test runs. Each test run is characterized by the specific input sequence of the variables and the grammar. The number of iterations as well as the number of evaluated, of redundant (generated trees are not redundantly evaluated) and of generated parse trees are given as average over all test runs. Note that the number of redundant parse trees for the original system is indeterminable. The running times are also provided as average over all test runs and are given in absolute numbers of seconds. Naturally the absolute running times depend on the underlying hardware. But the provided numbers allow the relative comparison of the time consumption between the two systems as all experiments were performed on the same hardware. (However, the analysis of the system did not focus on aspects of time consumption.)

system	*pos. detections*	*iterations*	*trees*			*seconds*
			evaluated	*redundant*	*generated*	
	(%)	(*mean*)	(*mean*)	(*mean*)	(*mean*)	(*mean*)
original	37.2	15	959	–	959	1588
modified	96.8	12	8549	2730	11279	42047

pos. detections: positive detections, i. e. at least one of the beam elements contains the EOM; *iterations*: iterations performed; *evaluated*: evaluated, i.e. fitted parse trees; *redundant*: redundant parse trees identified in backup table and not re-evaluated; *generated*: generated parse trees in terms of grammar derivations; *seconds*: running time of the system in seconds.

3.6 Discussion

The main results of this study are:

(i) The modified ED system was able to identify the well-known EOM model in real-world data obtained from mechanically ventilated patients.

(ii) The modified system's performance was independent of the input sequences of the model variables and the production rules of the context free grammar representing domain-dependent background knowledge.

Originally, ED approaches were data-driven [14, 56, 62, 82]. Depending on the size of the dataset and the hypothesis space, the trial-and-error strategy of this domain-independent [96] policy to fit functional formulas is generally time-consuming. Besides this computational problem – inherent in most ML and KDD tasks – Schaffer [97] highlighted that it is not the mere identification of relationships in data which ED has to deal

with. In fact, ED systems should exclusively identify relationships of real scientific relevance, a problem which has not been considered for a long time. He also raised concerns about systems being exclusively demonstrated on few hand-selected cases often involving artificial data [96]. Consequently, the E* algorithm he introduced was tested on a vast set of measured data. In spite of searching an infinite space of possible functional formulas, the search space was restricted to a fixed, finite set of potential relationships. In a way, this was a first step to a knowledge-driven, domain-dependent approach in ED.

Focusing on the reduction of search space complexity, declarative bias was introduced to ED [111]. Besides the benefit concerning computational complexity, this approach provides the capability to process domain-dependent background knowledge. Moreover, declarative bias represents such background knowledge in a very flexible way and therefore provides the capability to describe a problem task as a trade-off between exploration and exploitation: if the hypothesis space is bounded by tight constraints in terms of detailed background knowledge, the presented results might be scientifically more relevant while part of the exploratory discovery aspect of finding new systematic relations in the data might be lost. On the other hand, if the constraints are not tight enough, one has not only to face time constraints caused by a vast hypothesis space, but potentially also a loss of relevance in the provided results.

For the domain-specific task to identify a physiological lung model in measured respiratory time series data, an ED system had to: (i) be able to handle background knowledge, (ii) derive reasonable results in the least possible time and (iii) be able to handle noisy data in a robust way. According to (i), we found the LAGRAMGE system being well applicable to our task. However, first experiments on simulated data revealed that the ordering of the production rules according to the height of derivable parse trees implied a particular bias when traversing the hypothesis space, which consequently affected the identification of suitable models. We found two technical reasons responsible for these effects.

Firstly, in addition to the requested reduction of the hypothesis space by the declarative bias, the hypothesis space was kind of "cut". The sequential application of the production rules implies the uniqueness of the derivation paths. The refinement of a parse tree $\mathcal{T}$ (with start symbol S as root node) requires the termination of its subtree $\mathcal{T}_{\mathrm{A}}$ (nonterminal symbol A as root node), to which the production rule $p_{A,1}$ has been applied, by applying the first production rule $p_{B,1}$ to all nonterminals B in $\mathcal{T}_{\mathrm{A}}$. Therefore, if $\mathcal{T}$ is not kept as one of the beam elements for the next refinement iteration, none of the successor trees of T_A applying rule $p_{B,i\neq 1}$ instead of $p_{B,1}$ to any of the nonterminals B can be generated and thus are lost for evaluation (section 3.2.4, *Drawbacks (ii)*, page 55). This effect is even aggravated by the implementation of the beam search (section 3.2.4, *Drawbacks (iii)*, page 61): within an iteration, the element with the highest error with respect to $\mathcal{F}_{\mathrm{h}}$ of the actual beam is immediately substituted with a refinement having a lower error. A refinement $\mathcal{T}'$ is not exclusively compared to its origin $\mathcal{T}$, but always to the set of all beam elements. This implies the possibility that even if a refinement could lead to improvements of its origin

in terms of $\mathcal{F}_\mathrm{h}$ in successive iterations (i.e. refinements), it might not be stored in the beam for further refinements. This property could be compensated to a certain degree by a larger beam width, however, resulting in an increasing memory demand. On the other hand, if a beam element was not improved but also not substituted by the refinement of another element, the same production rules are applied in the next iteration step again and thus the same refinements are re-evaluated.

Secondly, the sequential transition of the hypothesis space might occasionally cause the algorithm to meet the stopping criterion in a local minimum. Refining subtree $\mathcal{T}_\mathrm{A}$ by applying production rule $p_{A,i+1}$ might impair the model in terms of $\mathcal{F}_\mathrm{h}$ compared to the application of $p_{A,i}$ and thus $\mathcal{T}_\mathrm{A}$ might be excluded from further refinements — depending on the beam width. However, refining $\mathcal{T}_\mathrm{A}$ by $p_{A,i+2}$ in turn might improve the model, but in this case is not processed anymore (section 3.2.4, *Drawbacks (i)*, page 51 and *Drawbacks (ii)*, page 55).

Thus, the task was to adequately modify the search heuristic of the algorithm. Although being simple, the local greedy search performed by the GSAT algorithm has been shown to perform effectively [100]. Yet, the performance of GSAT was hard to explain. It has been speculated that the crucial factor is to have an approximate solution which can be refined iteratively. We found this approach perfectly matching our task: starting with a randomly generated model, i.e parse tree, this was attempted to be improved by systematically applied random refinements of the iteratively derived parse trees. According to our results, this strategy to traverse the hypothesis space was found to be effective. The previously mentioned "cutting" of the hypothesis space – biased by the sequential application of production rules and thus ordered parse tree refinements – could obviously be avoided by the strategy of randomized parse tree refinements. Applied in context of the modified beam search, the system includes aspects of a momentum and a lookahead: each single beam element is evaluated and replaced as necessary. On the other hand, the stopping criterion is not met before none of all beam elements can be improved with respect to the heuristic function. In this way, individual elements are refined again (lookahead) although possibly not having been improved in a preceding refinement iteration (momentum).

Testing our system on real-world data obtained from mechanically ventilated patients without any preprocessing of the data in terms of noise reduction, the results indicate a robust performance. The additional bias affecting the search strategy was eliminated. The data noise as usually found in measured data of physiological systems did not show any effect on the performance within our experimental settings. However, caused by the momentum and the lookahaed, the modified system evaluated a much higher number of models compared to the original system (Table 3.7). Resulting in an increased evaluation time, this is a hard constraint concerning the potential application at bedside under the aspect of individualized model inference for the patient as diagnostic support. Nevertheless, an average of about 8550 evaluated models is still a small fraction of the overall number of more than 1.4 million models spanning the hypothesis space.

3.7 Related Work

To the best of our knowledge, this is the first application of an ED system to measured respiratory data from intensive care medicine. However, efforts have been made to improve ED systems which are capable to process domain-dependent background knowledge since they have been introduced. Promoting the view of such "domain-dependent" ED, Langley *et al.* introduced the approach of inductive process modeling (IPM) [16, 64]. IPM combines the objectives of model inference from time series data, knowledge representation in established scientific formalisms and incorporation of domain knowledge embedded in a simulation environment. The system incorporates LAGRAMGE to take advantage of declarative bias though not using the full scope of LAGRAMGE's abilities concerning the representation of background knowledge in terms of context free grammars. IPM combines processes into a model, and background knowledge consists of the provision of generic processes. Models are induced under the assumption that the combination of any set of generic processes produces valid model structures. Consequently, the model space might contain candidates noncompliant to the expectations of a domain expert. Thus, Todorovski *et al.* [112] further developed the IPM approach to hierarchical inductive process modeling (HIPM), where processes are represented in a hierarchical order.

A general problem in ED – as well as in the entire field of ML – is that of overfitting the data. An approach addressing this point explicitly is the FUSE (forming unified scientific explanations) system [15]. A main aspect of FUSE is a data preprocessing step related to bagging, which enables the system to infer several different models by application of HIPM. Amongst those, the subprocesses are ranked by their frequency and combined into a final model. In another approach discussing overfitting in ED, De Pauw *et al.* [24] introduced a model identifiability measure to generate models with an optimized complexity with respect to the given data. Background knowledge is represented in the form of an initial model being either general or overly complex.

A main requirement on an ED system applied to the specific domain of modeling lung mechanics was a flexible representation of background knowledge. The system should be capable of inferring models from rather low-level prior knowledge as well as from more detailed knowledge. This is important to the domain expert, who might start modeling from scratch (no background knowledge) or who might want to improve existing models (background knowledge given). Although having proven to perform well in multiple applications, we assume the aforementioned systems demand rather detailed prior insight into the processes to be modeled. Again, the representation of background knowledge by context free grammars is highly flexible and furthermore can be used for reducing the search space. Depending on the amount of background knowledge provided, the analysis of the data can be designed to have more exploratory or more exploitative characteristics. It is arguable to which degree model complexity could be influenced by an appropriate structure of the applied grammar, which in turn could help to reduce overfitting but potentially would require again specific prior knowledge. It is also arguable if for some applications

the order dependent approach of the original LAGRAMGE system might even be helpful, as it could be used as a specific feature to bring in background knowledge.

As our study was designed in the context of a specific real-world problem domain, our approach was of a rather pragmatic character. We focused on improving the robustness of the applied system concerning reproducibility of model inference in the context of sparse prior knowledge, accepting the previously mentioned drawbacks of a comparatively basic system. However, the optimization of model complexity and the integration of a model identifiability measure is an important aspect to be considered in future work, and further experiments could be conducted with such a system incorporating the search strategy presented in this study.

3.8 Conclusions

We presented an ED approach based on the LAGRAMGE system [111], being able to handle domain-dependent background knowledge. To decouple the search heuristic for traversing the hypothesis space from the presentation of the background knowledge, we implemented a randomized hill-climbing search heuristic resembling the GSAT algorithm. This novel system was shown to be well applicable in the domain of real-world time series data, as it was robust concerning the mode of the data input, the representation of background knowledge and noise inherent in the measured data. This is a prerequisite for further applications in respiratory physiology.

Chapter 4

Gaussian Process Modeling for the Prediction of Mechanical Lung Parameters

To avoid ventilator associated lung injury (VALI) during mechanical ventilation, the ventilator is adjusted with reference to the compliance of the lung. For lung protective ventilation, the lung should be inflated at its maximum compliance, i.e. when during inspiration a maximal intrapulmonary volume change is achieved by a minimal change of pressure (Figure 1.1). To accomplish this, one of the main parameters is the adjusted PEEP. As changing the ventilator settings usually produces an effect on the patient's lung mechanics with a considerable time delay, the prediction of the compliance change associated with a planned change of PEEP could assist the physician at the bedside.

The purpose of this study is to statistically model the pressure (or volume) dependent behavior of the compliance, taking into account nonlinear properties across the entire range of the inspiratory lung capacity. The detailed prediction of this mechanical lung parameter could assist the physician when individually adjusting the applied pressure level in order to reduce the risk of VALI. To address our prediction task, we apply Gaussian processes (GPs) [91], a probabilistic, nonparametric modeling approach for nonlinear functions. GPs are applicable for supervised regression and classification tasks in high-dimensional spaces where the input as well as the output space can be multidimensional. Although GPs were introduced as the technique of GP regression or Kriging [58] decades ago, they have been gaining attractiveness over the past few years, as they were shown to be applicable to a wide range of datasets [90, 91, 118].

4.1 Medical Background and Clinical Purpose

When applying a super syringe maneuver [69], rapid flow interruptions (section 1.2.4) are iteratively performed after consecutive inflations of equally sized volume portions (section 1.2.5). These flow interruptions reveal characteristic stress relaxation curves, exponentially approximating the plateau pressure level P_{plat} (Figures 1.2, 4.1). The spring-and-dashpot model [6] (section 1.2.4) is assumed to simulate the viscoelastic behavior of the lung tissue. Fitting this model to flow interruption data provides the four parameters respiratory compliance C, resistance R, viscoelastic compliance C_{ve} and viscoelastic resistance R_{ve}, which are nonlinearly related to P_{plat} [38] (chapter 2). In its entirety the numerical values of these parameters reflect the mechanical status of the respiratory system. Although such model approaches provide a detailed quantitative description of the lung and consider dynamic effects, in medical practice, mechanical lung parameters are determined from data obtained under static conditions (sections 1.2.4, 1.2.5).

In clinical practice, the PEEP is adjusted to a level where the corresponding compliance is assumed to be maximal (section 1.2.3). The pressure-volume range which is associated with maximal compliance (Figure 1.1) is highly individual for each patient and influenced by the actual status of the lung. A change of the ventilator settings usually affects the lung mechanics with a considerable time delay. Therefore, the prediction of the effects related to the change of the PEEP-settings with respect to the compliance could assist the physician at the bedside. Firstly, possibly harmful maneuvers could be at least partially avoided. Secondly, as the initial ventilatory approach might already induce a subsequent lung damage and in case of ARDS essentially influences the further progress of the disease, an objective a priori estimation of optimized ventilatory settings could provide a helpful advice in terms of lung protective ventilation (section 1.2.3).

4.2 Materials and Methods

4.2.1 Gaussian Processes for Regression

The core of this study faces a classical regression task: given a dataset $\mathcal{D}$ consisting of N observations of X_N, y_N, where X_N denotes a set of N input vectors $\{\mathrm{x}_n\}_{n=1}^N = \{[x_1, \ldots, x_D]_n\}_{n=1}^N$ of parameters x with a fixed dimension D and y_N the set of corresponding target values $\{y_n\}_{n=1}^N$, what is the target value y_{N+1} for a new input data point x_{N+1}? Gaussian regression [58, 67, 80, 91, 119] introduces a probabilistic, nonparametric approach to this problem.

Gaussian Processes

Nonlinear regression problems are generally modeled by parametrizing a function $f(\mathrm{x})$ with parameters w to $f(\mathrm{x};\mathrm{w})$. Using a set of (nonlinear) basis functions $\{\phi_h(x)\}_{h=1}^{H}$, the parameterized function $f(\mathrm{x};\mathrm{w})$ can be rewritten as a linear combination of these basis functions $\phi_h(x)$ as

$$f(\mathrm{x};\mathrm{w}) = \Phi(\mathrm{x})\mathrm{w} = \sum_{h=1}^{H} \phi_h(\mathrm{x})\omega_h \tag{4.1}$$

with the vector $\Phi(\mathrm{x}) = [\phi_1(\mathrm{x}), \ldots, \phi_H(\mathrm{x})]$ containing the H basis functions and the vector $\mathrm{w} = [\omega_1, \ldots, \omega_H]$ containing the H parameters.

With the set of input observations X_N an $N \times H$ matrix $\mathbf{\Phi}_{NH}$ can be defined as the matrix of values of the basis functions $\{\phi_h(x)\}_{h=1}^{H}$ at the input points $\{\mathrm{x}_n\}_{n=1}^{N}$:

$$\mathbf{\Phi}_{NH} = \begin{pmatrix} \phi_1(\mathrm{x}_1) & \phi_2(\mathrm{x}_1) & \cdots & \phi_H(\mathrm{x}_1) \\ \phi_1(\mathrm{x}_2) & \phi_2(\mathrm{x}_2) & \cdots & \phi_H(\mathrm{x}_2) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_1(\mathrm{x}_\mathrm{N}) & \phi_2(\mathrm{x}_\mathrm{N}) & \cdots & \phi_H(\mathrm{x}_\mathrm{N}) \end{pmatrix} \tag{4.2}$$

Referring to this matrix, the function values $f(\mathrm{x}_n;\mathrm{w})$ for input point $\mathrm{x}_n \in \mathrm{X}_N$ with parameter vector w can be calculated as

$$f(\mathrm{x}_n;\mathrm{w}) = \sum_{h=1}^{H} \mathbf{\Phi}_{nh}\omega_h \tag{4.3}$$

Under the assumption that the prior distribution P of the parameters w of dimension H is Gaussian with mean $\vec{\mu} = E(\mathrm{w}) = \vec{0}$ and the covariance matrix $\Sigma_\mathrm{w} = E((\mathrm{w} - \vec{\mu})(\mathrm{w} - \vec{\mu})^\mathrm{T})$, i.e.

$$P(\mathrm{w}) = \mathcal{N}_H(\vec{0}, \Sigma_\mathrm{w}), \tag{4.4}$$

the function f as a linear combination of the basis functions $\phi_{1\ldots H}(\mathrm{x_n})$ with the Gaussian distributed (with zero mean) coefficients w is also Gaussian distributed with zero mean: with f_N as the vector of all values of $f(\mathrm{x}_{1\ldots N})$ at all N input points $\{\mathrm{x}_1, \ldots, \mathrm{x}_N\} = \mathrm{X}_N$,

the mean m of f_N and the covariance matrix C of f_N can be built as

$$\begin{aligned} m(f_N) &= E(f_N) \\ &= E(\mathbf{\Phi}_{NH}\mathrm{w}) \\ &= \mathbf{\Phi}_{NH}E(\mathrm{w}) \\ &= \mathbf{\Phi}_{NH}\cdot\vec{0} \\ &= \vec{0} \end{aligned} \tag{4.5}$$

$$\begin{aligned} C(f_N) &= E((f_N-\vec{\mu})(f_N-\vec{\mu})^{\mathrm{T}}) \\ &= E(f_N f_N^{\mathrm{T}}) \\ &= E((\mathbf{\Phi}_{NH}\mathrm{w})(\mathbf{\Phi}_{NH}\mathrm{w})^{\mathrm{T}}) \\ &= E(\mathbf{\Phi}_{NH}(\mathrm{w}\mathrm{w}^{\mathrm{T}})\mathbf{\Phi}_{NH}^{\mathrm{T}}) \\ &= \mathbf{\Phi}_{NH}E(\mathrm{w}\mathrm{w}^{\mathrm{T}})\mathbf{\Phi}_{NH}^{\mathrm{T}} \\ &= \mathbf{\Phi}_{NH}E((\mathrm{w}-\vec{\mu})(\mathrm{w}-\vec{\mu})^{\mathrm{T}})\mathbf{\Phi}_{NH}^{\mathrm{T}} \\ &= \mathbf{\Phi}_{NH}\Sigma_{\mathrm{w}}\mathbf{\Phi}_{NH}^{\mathrm{T}} \end{aligned} \tag{4.6}$$

With Equations (4.5) and (4.6) we get the prior distribution of f_N as

$$P(f_N|\mathrm{X}_N) = (\vec{0}, \mathbf{\Phi}_{NH}\Sigma_{\mathrm{w}}\mathbf{\Phi}_{NH}^{\mathrm{T}}) \tag{4.7}$$

with the result that the vector f_N of all N function values has a Gaussian distribution with zero mean for the input data X_N. Assuming that the target values differ by additive Gaussian noise from the function values, the prior distribution P of the target values y_N is also Gaussian:

$$\begin{aligned} P(\mathrm{y}_N|\mathrm{X}_N) &= \mathcal{N}(\vec{0}, C(f_N)+\Sigma_{\mathrm{y}}) \\ &= \mathcal{N}(\vec{0}, \mathbf{\Phi}_{NH}\Sigma_{\mathrm{w}}\mathbf{\Phi}_{NH}^{\mathrm{T}}+\Sigma_{\mathrm{y}}) \end{aligned} \tag{4.8}$$

Now we meet the defining property of a GP [91]:

Definition 10. *A Gaussian process (GP) is a collection of random variables, any finite number of which have a joint Gaussian distribution.*

As we can see, the covariance of the outputs y is expressed as a function $f(\mathrm{x};\mathrm{w})$ of the inputs x. In GP modeling, the covariance of the outputs is described by a covariance or

kernel function $k(\mathrm{x}, \mathrm{x}')$. This, together with the mean function $m(\mathrm{x})$ completely specifies a GP as

$$f(\mathrm{x}) \sim \mathcal{GP}(m(\mathrm{x}), k(\mathrm{x}, \mathrm{x}')) \tag{4.9}$$

with

$$\begin{aligned} m(\mathrm{x}) &= E[f(\mathrm{x})] & (4.10)\\ k(\mathrm{x}, \mathrm{x}') &= E[(f(\mathrm{x}) - m(\mathrm{x}))(f(\mathrm{x}') - m(\mathrm{x}'))] & (4.11) \end{aligned}$$

Prediction Using Gaussian Processes

According to Mercer's theorem [91], any covariance function $k(\cdot, \cdot)$ may be chosen as long as it is positive definite. A popular example for such a covariance function is given by the RBF-kernel:

$$k(\mathrm{x}, \mathrm{x}') = \exp\left(-\frac{||\mathrm{x} - \mathrm{x}'||^2}{2\sigma^2}\right). \tag{4.12}$$

A radial basis function (RBF) centers the function values radial-symmetrically around a center and can be used as a basis function for approximation purposes. Other kernels include additional parameters, which is the reason that GP modeling is not completely nonparametric. Such a kernel is given by e.g.

$$k(\mathrm{x}, \mathrm{x}'; \Theta) = \theta_1 \exp\left[-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d - x_d')^2}{r_d^2} + \theta_2\right], \tag{4.13}$$

where x_d is the d^{th} component of a d dimensional vector x. $\Theta = (\theta_1, \theta_2, \{r_d\})$ defines the parameters of the covariance function k for a D-dimensional parameter space: θ_1 defines the vertical scale of variations of the function. θ_2 defines the offset for the function away from zero. Finally, r_d defines a length scale related to each direction d of the D-dimensional parameter space for which a significant variance of the target y is expected.

Having N observations $\mathrm{X}_N, \mathrm{y}_N$ of the input vector x of dimension D and the corresponding target value y (here, the target space is one dimensional, however, GPs are capable to handle multidimensional target spaces as well) and having chosen a covariance function $k(\cdot, \cdot)$ the prediction of a target y_{N+1} at a new observation of input point x_{N+1} is straightforward: first, the covariance function is calculated for all possible combinations of the N

input points resulting in the following matrix K_N:

$$K_N = \begin{bmatrix} k(\mathrm{x}_1, \mathrm{x}_1, \Theta) & k(\mathrm{x}_1, \mathrm{x}_2, \Theta) & \cdots & k(\mathrm{x}_1, \mathrm{x}_N, \Theta) \\ k(\mathrm{x}_2, \mathrm{x}_1, \Theta) & k(\mathrm{x}_2, \mathrm{x}_2, \Theta) & \cdots & k(\mathrm{x}_2, \mathrm{x}_N, \Theta) \\ \vdots & \vdots & \vdots & \vdots \\ k(\mathrm{x}_N, \mathrm{x}_1, \Theta) & k(\mathrm{x}_N, \mathrm{x}_2, \Theta) & \cdots & k(\mathrm{x}_N, \mathrm{x}_N, \Theta) \end{bmatrix} \tag{4.14}$$

Second, the covariance function is calculated for the combination of the new observation x_{N+1} with the previously observed points $\mathrm{x}_n \in \mathrm{X}_N$ as matrix K_*:

$$K_* = \begin{bmatrix} k(\mathrm{x}_{N+1}, \mathrm{x}_1, \Theta) & k(\mathrm{x}_{N+1}, \mathrm{x}_2, \Theta) & \cdots & k(\mathrm{x}_{N+1}, \mathrm{x}_N, \Theta) \end{bmatrix} \tag{4.15}$$

And finally, the covariance function is calculated for the combination of the new observation with itself resulting in the matrix K_{**}:

$$K_{**} = \begin{bmatrix} k(\mathrm{x}_{N+1}, \mathrm{x}_{N+1}, \Theta) \end{bmatrix} \tag{4.16}$$

According to the definition of a GP in Definition 10 we have

$$\begin{bmatrix} \mathrm{y}_N \\ y_{N+1} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K_N & K_*^T \\ K_* & K_{**} \end{bmatrix}\right) \tag{4.17}$$

Now, given the original observations in dataset $\mathcal{D}$ consisting of the inputs X_N and the corresponding outputs y_N and given the new observation of the input variables x_{N+1} the conditional distribution of the output y_{N+1} is Gaussian and can be derived by

$$P(y_{N+1}|\mathrm{x}_{N+1}, \mathcal{D}) \sim \mathcal{N}(m(\mathrm{x}_{N+1}), C(\mathrm{x}_{N+1})) \tag{4.18}$$

with

$$m(\mathrm{x}_{N+1}) = K_* K_N^{-1} \mathrm{y}_N \tag{4.19}$$

$$C(\mathrm{x}_{N+1}) = K_{**} - K_* K_N^{-1} K_*^T \tag{4.20}$$

where $m(\mathrm{x}_{N+1})$ represents the estimate for y_{N+1} as the mean of the distribution.

4.2.2 Prediction Task

To determine the lung compliance C, the lung-volume is related to the alveolar pressure (Equation (1.1)). Different pressure-volume levels are related to differing compliance values. Expecting lower compliance values within the lower and upper part of a pressure-volume curve and higher compliance values in the center part of such a curve (section 1.2.3, Figure 1.1), a prior assumption on our modeling task is that the hypothesis space consists of the derivatives of sigmoid-like shaped functions. As all sets of respiratory data obtained from the patients imply general as well as individual characteristics of the ARDS lung, the shape of these functions ought to be distributed according to a prior probability. Therefore, we hypothesized that our modeling task would benefit from the probabilistic modeling of (possibly) nonlinear functions as provided by GP modeling. In this study inferences were made from the status of the respiratory system to the compliance value. The inference procedure is subdivided into two steps:

Inference step (i): First, after the feature (i.e. parameter) extraction from the measured raw data, a (statistical) GP-model is generated.

Inference step (ii): Second, based on such a GP-model, predictions are made for the lung compliance.

The feature extraction (section 4.2.4) basically relates distinct pressure levels to the estimated compliance level. As a statistical modeling technique is used, the prediction of the compliance is not constricted to such originally related pressure-compliance levels. In contrast, the lung compliance is predicted for pressure levels far out of bound (higher and lower) of the originally related pressure range. In this study inferences were made from the status of the respiratory system at a distinct pressure level range to the compliance value at a different pressure level. The status of the respiratory system is quantified by the parameters compliance C, Newtonian airway resistance R (section 1.2.2) as well as the compliance C_{ve} and resistance R_{ve}, which are supposed to be related to viscoelastic effects of the lung tissue (section 1.2.4, *Stress Relaxation Processes* and *The Spring-and-Dashpot Model*).

The modeling task is based on three requirements. Firstly, the models should reliably predict the compliance for all measured pressure levels. Secondly, especially the pressure level at which a maximum compliance can be expected should be predictable. And finally, related to the demands of the physician at bedside, it should be predicted if the basic pressure level, i.e. the PEEP, could be optimized in terms of a lung protective ventilation, i.e. a ventilation at a maximized compliance level. As a consequence, the following tasks can be defined:

Prediction task (i): The modeling and prediction of the compliance-pressure curve covering the range of the inspiratory lung capacity.

Prediction task (ii): The prediction of the maximum compliance value C_{max} (Figure 1.1) and its corresponding plateau pressure value $P_{plat}(C_{max})$ (Figure 1.2).

Prediction task (iii): The prediction whether the instantaneous pressure level should be increased, decreased or retained in order to achieve C_{max} (which we refer to as "trend" in the following).

4.2.3 Raw Data

The data for this retrospective study were obtained from a multicenter study including patients mechanically ventilated due to severe ARDS [35, 105] (section 1.3.1). Standardized super syringe maneuvers [69] (sections 1.2.5, *The Super Syringe Maneuver*; 1.3.1, *Respiratory Maneuvers*) were applied completely for 20 patients. As for two patients the tube-types were not recorded and thus the tracheal pressure could not be calculated (section 4.2.4, *Preprocessing*), 18 patients were included for this study (Table 1.2). During the automatically operated super syringe maneuvers, the ventilator repetitively applied volume steps of 100 mL, with an inspiratory airflow rate of 558 ± 93 mL/sec up to a maximum plateau pressure of 45 mbar. At the end of each volume application, airflow was interrupted for three seconds.

4.2.4 Data Preprocessing and Feature Extraction

Prediction task (i) *modeling of compliance-pressure curve* is based on the features compliance C, resistance R, viscoelastic compliance C_{ve}, viscoelastic resistance R_{ve} and plateau pressure P_{plat} which are extracted from the preprocessed raw data. Prediction tasks (ii) *prediction of maximum compliance and corresponding plateau pressure* and (iii) *prediction of trend* are based on the features maximum compliance C_{max} and P_{plat} which are extracted from the modeling result of task (i) and the preprocessed raw data.

Preprocessing

To determine the tracheal pressure P_{trach} as improved approximation of the alveolar pressure P (section 1.2.2), the pressure drop at the endotracheal tube ΔP_{ETT} was calculated by the Rohrer equation and subtracted from the measured airway pressure P_{aw}. All further computations are based on the alveolar pressure P, i.e. its approximation P_{trach}. For detailed information cf. Appendix D.

For each flow interruption step i, the plateau pressure level P^i_{plat} was approximated by the mean pressure of the last 0.5 s of airflow interruption (Figure 4.1, insert). The

airway pressure data of each interruption step i were corrected by the offset of the preceding plateau pressure value at step $i-1$, i.e. from each pressure value P which was measured during the interval of interruption step i, the pressure offset P_{plat}^{i-1} was subtracted (section 2.1.2, Equation (2.1)).

Feature Extraction

For prediction task (i): for each flow interruption step i, the attributes C^i, R^i, C_{ve}^i and R_{ve}^i were estimated by fitting the electrical analog of a spring-and-dashpot model [53] to the data (section 1.2.4, *The Spring-and-Dashpot Model*) corresponding to P_{plat}^i. The spring-and-dashpot model was represented by the Laplace transform with the respiratory flow as system input and the respiratory pressure as output (Equation (2.2), Appendix A). For optimization the simplex search method of Lagarias *et al.* [60] was used with the sum of squared errors as the scalar function of R, C, R_{ve} and C_{ve} to be optimized. Feature samples including compliances > 300 ml/mbar were excluded as unphysiologically high values. For model fitting, we used the MatLab® software package version R2006b (The MathWorks™, Natick, MA).

For prediction tasks (ii) and (iii): compliance-pressure curves (compliance C related to plateau pressure level P_{plat}) were predicted over the entire measured pressure range for each maneuver. To do so, the compliance values were predicted for all measured plateau pressures (section 4.2.6). Note that the compliance prediction of compliance corresponding to plateau pressure is implied in prediction task (i).

The C_{max} value of a compliance-pressure curve was determined by fitting a degree three polynomial to the raw data and to the modeled compliance-pressure curve, respectively. Then the extreme values of the polynomial were calculated. If no local maximum was located within the pressure range $(min(P_{plat}^1, \ldots, P_{plat}^n), max(P_{plat}^1, \ldots, P_{plat}^n))$ covered by the individual super syringe maneuver, C_{max} was set to the C value at the minimum or maximum pressure value, depending on the slope of the curve (Figure 4.2).

4.2.5 Feature Data

The initial system status was represented by the fitted parameter and measured plateau pressure values of four consecutive steps i to $i+3$, resulting in the feature sample given by the tuple $(P_{plat}^{i\ldots i+3}, C^{i\ldots i+3}, R^{i\ldots i+3}, C_{ve}^{i\ldots i+3}, R_{ve}^{i\ldots i+3})$. Based on the feature tuple the target value to predict was the compliance value C for the corresponding plateau pressure P_{plat} at step k, i.e. C^k at level P_{plat}^k . For the learning task, the input feature samples consisted of 22-tuples $(P_{plat}^{i\ldots i+3}, C^{i\ldots i+3}, R^{i\ldots i+3}, C_{ve}^{i\ldots i+3}, R_{ve}^{i\ldots i+3}, P_{plat}^k, C^k)$: for each patient, i.e. super syringe maneuver, the complete input dataset for the model learning task consisted of all possible tuples $(P_{plat}^{i\ldots i+3}, C^{i\ldots i+3}, R^{i\ldots i+3}, C_{ve}^{i\ldots i+3}, R_{ve}^{i\ldots i+3})$ combined

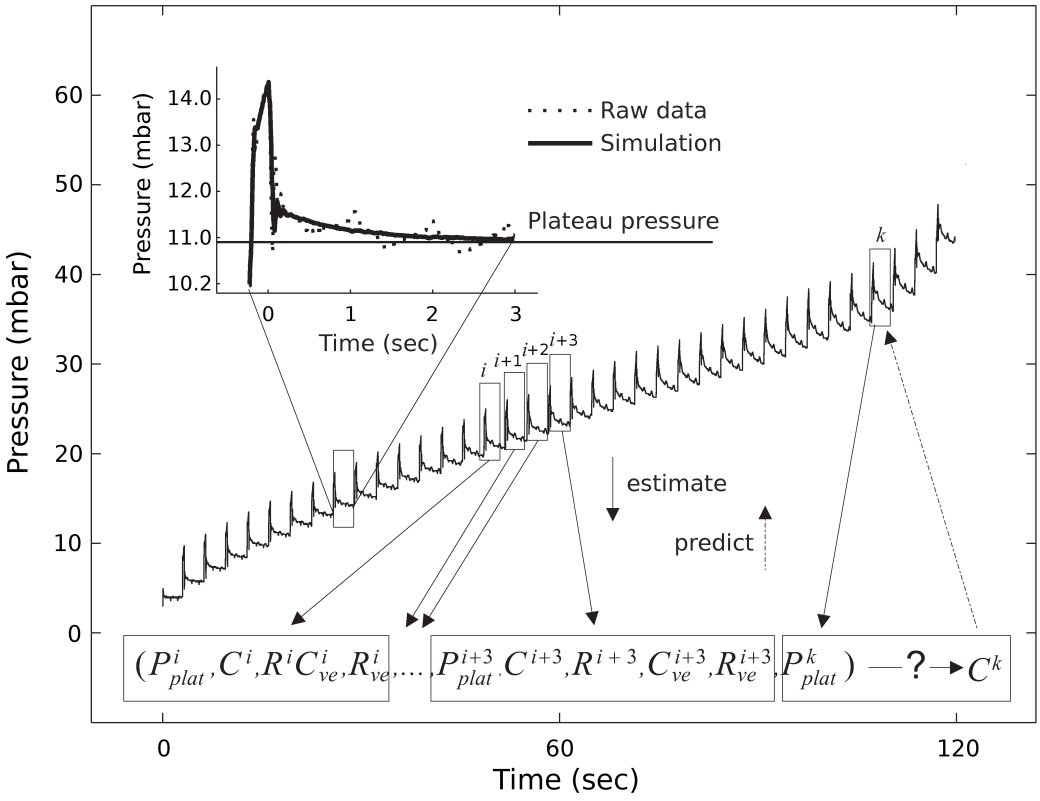

Figure 4.1: Feature extraction from super syringe maneuver data. The main curve shows a representative data sample recorded during a super syringe maneuver. The insert shows a single occlusion step resulting in plateau pressure level P_{plat} of approximately 11.0 mbar after an airway occlusion time of 3 sec. Raw data and simulated model fit of the spring-and-dashpot model are shown. For the estimation of the features C, R, C_{ve} and R_{ve} the spring-and-dashpot model was fitted to the time series data recorded at each volume inflation step. Each such step corresponds to a distinct plateau pressure level. The extracted feature of in each case four consecutive plateau pressure levels i to $i+3$ were combined with all possible pairs of C^k and P^k_{plat} (i.e. C^k related to the plateau pressure P^k_{plat} which it corresponds to) (Table 4.1). For the model learning task, the input data samples consisted of 22-tuples $(P^{i\ldots i+3}_{plat}, C^{i\ldots i+3}, R^{i\ldots i+3}, C^{i\ldots i+3}_{ve}, R^{i\ldots i+3}_{ve}, P^k_{plat}, C^k)$. For the prediction task following, the input consisted of 21-tuples $(P^{i\ldots i+3}_{plat}, C^{i\ldots i+3}, R^{i\ldots i+3}, C^{i\ldots i+3}_{ve}, R^{i\ldots i+3}_{ve}, P^k_{plat})$ with C^k as the target to predict.

Table 4.1: Sample data of one patient for model learning. For this patient, data for in total nine consecutive plateau pressure levels were recorded. The lung status is described by the 20-tuple ($P_{plat}^{i...i+3}$, $C^{i...i+3}$, $R^{i...i+3}$, $C_{ve}^{i...i+3}$, $R_{ve}^{i...i+3}$) including P_{plat}, C, R, C_{ve} and R_{ve} of four consecutive pressure levels $i...i+3$. Consequently, for this patient six lung status could be described. Each such status is combined with all of the nine plateau pressure levels P_{plat}^{k} and the corresponding compliance C^{k}. In total this results in 54 22-tuples of the form ($P_{plat}^{i...i+3}$, $C^{i...i+3}$, $R^{i...i+3}$, $C_{ve}^{i...i+3}$, $R_{ve}^{i...i+3}$, P_{plat}^{k}, C^{k}). Three consecutive pressure levels are depicted in the table where $P_{plat}^{i...i+3}$ is given as $Pplat\_[a,\ldots,d]$, $C^{i...i+3}$ as $C1\_[a,\ldots,d]$, $R^{i...i+3}$ as $R1\_[a,\ldots,d]$, $C_{ve}^{i...i+3}$ as $C2\_[a,\ldots,d]$, $R_{ve}^{i...i+3}$ as $R2\_[a,\ldots,d]$, P_{plat}^{k} as $Pplat\_r$ and C^{k} as $C1\_r$. Note that no bootstrapping (random sampling with replacement) was applied; each sample is unique.

Pplat_a	R1_a	R2_a	C1_a	C2_a	Pplat_b	R1_b	R2_b	C1_b	C2_b
⋮					⋮				⋮
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
9.2444	0.007806	0.008113	32.5412	78.9177	11.6841	0.006668	0.013343	39.8745	63.592
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
11.6841	0.006668	0.013343	39.8745	63.592	13.8443	0.007108	0.020782	47.5262	58.9178
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713
⋮					⋮				⋮

⟶

Pplat_c	R1_c	R2_c	C1_c	C2_c	Pplat_d	R1_d	R2_d	C1_d	C2_d	Pplat_r	C1_r
⋮					⋮						⋮
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	9.2444	32.5412
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	11.6841	39.8745
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	13.8443	47.5262
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	15.9443	53.1127
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	54.0266
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	19.1635	65.6289
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	20.4952	104.8464
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	21.9126	87.7620
13.8443	0.007108	0.020782	47.5262	58.9178	15.9443	0.007228	0.028462	53.1127	50.2713	23.1589	67.4900
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	9.2444	32.5412
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	11.6841	39.8745
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	13.8443	47.5262
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	15.9443	53.1127
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	17.7572	54.0266
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	65.6289
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	20.4952	104.8464
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	21.9126	87.7620
15.9443	0.007228	0.028462	53.1127	50.2713	17.7572	0.006385	0.020368	54.0266	39.7203	23.1589	67.4900
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	9.2444	32.5412
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	11.6841	39.8745
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	13.8443	47.5262
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	15.9443	53.1127
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	17.7572	54.0266
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	19.1635	65.6289
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	20.4952	104.8464
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	21.9126	87.7620
17.7572	0.006385	0.020368	54.0266	39.7203	19.1635	0.005523	0.026624	65.6289	32.1067	23.1589	67.4900
⋮					⋮						⋮

Table 4.2: Dataset statistics. For each patient (Table 1.2) the number of data samples, the calculated maximum compliance C_{max} and the corresponding plateau pressure $P_{plat}(C_{max})$ based on the raw data are given. The *Testing* samples correspond to all data samples (22-tuples) which were generated for each single patient measurement (super syringe maneuver) by combination of the extracted features C, R, C_{ve}, R_{ve} and P_{plat}. For setting (i) these samples were used for both, learning the model and predicting the compliance. For setting (ii), the model was learned on the *Training* samples. For each patient these *Training* samples consisted of all *Testing* samples for all patients excluding the *Testing* data for this very patient. After learning the model from the *Training* data, in setting (ii) the model then was applied on the *Testing* data to predict the compliance. Consequently, for performance measuring the model prediction based on the *Testing* data was evaluated. The size of the datasets obtained from each patient, i.e. the *Testing* data, depended on the number of occlusions performed during the super syringe maneuver (and thus on the number of extracted feature values) as well as the number of excluded unphysiological values in the preprocessing step. In total, for each patient the *Training* and the *Testing* samples summed up to 4526.

Patient	*#Training*	*#Testing*	C_{max} mL/mbar	$P_{plat}(C_{max})$ mbar
1	4514	12	78.2	44.3
2	3446	1080	121.4	30.9
3	4256	270	74.2	30.3
4	4066	460	141.3	36.8
5	4508	18	28.0	36.7
6	4418	108	37.8	20.9
7	3954	572	96.8	31.1
8	4438	88	82.1	36.1
9	3976	550	81.4	23.9
10	4222	304	51.2	23.7
11	4256	270	153.1	29.9
12	4438	88	80.5	33.9
13	4148	378	62.2	20.1
14	4472	54	82.1	21.3
15	4522	4	40.3	11.0
16	4418	108	115.8	33.6
17	4418	108	120.0	32.6
18	4472	54	29.5	9.1

#Training, #Testing: number of samples for training and testing; C_{max}: maximum compliance estimated from the Testing samples; $P_{plat}(C_{max})$: plateau pressure at which maximum compliance was measured.

with each single compliance to pressure relation (C^k, P^k_{plat}) for all occlusion steps. For the prediction of the target C^k based on such models, the input consisted of 21-tuples $(P^{i...i+3}_{plat}, C^{i...i+3}, R^{i...i+3}, C^{i...i+3}_{ve}, R^{i...i+3}_{ve}, P^k_{plat})$.

In Figure 4.1 the feature extraction is explained in detail. Table 4.1 shows an extract of an input sample data file for the modeling task.

4.2.6 Data Modeling and Experimental Settings

While the prediction system was implemented via GP modeling, the M5P algorithm [89] was applied as reference method for the validation of the system (see section *System Validation: M5P* in this chapter). All algorithms for parameter prediction were applied as implemented in the data mining software WEKA [120].

System Estimation: Gaussian Process Modeling

For GP modeling a Pearson VII function-based universal kernel [113]

$$K(x_i, x_j; (\sigma, \omega)) = \frac{1}{\left[1 + \left(\frac{2\sqrt{\|x_i - x_j\|^2}\sqrt{2^{(1/w)}-1}}{\sigma}\right)^2\right]^{\omega}} \tag{4.21}$$

with $\sigma = 1$ and $\omega = 1$ (σ conrols the half-width or Pearson width and ω the tailing factor of the peak) was applied. The simplified equation is given below.

$$K(x_i, x_j; (1, 1)) = \frac{1}{1 + \left(2\sqrt{\| x_i - x_j \|^2}\right)^2} \tag{4.22}$$

For prediction task (i) the models were learned based on the 22-tuples $(P^{i...i+3}_{plat}, C^{i...i+3}, R^{i...i+3}, C^{i...i+3}_{ve}, R^{i...i+3}_{ve}, P^k_{plat}, C^k)$ as input (section 4.2.4). For the prediction of the entire compliance-pressure curve (i.e. C related to P_{plat}), for all pressure ranges, i.e. 21-tuples $(P^{i...i+3}_{plat}, C^{i...i+3}, R^{i...i+3}, C^{i...i+3}_{ve}, R^{i...i+3}_{ve}, P^k_{plat})$, the target C^k was predicted (Table 4.1) as the posterior mean of the GP model. (Whereas the posterior variance was of no importance for analysis purposes.)

For the prediction tasks (ii) and (iii) the measured and the predicted C_{max} and P_{plat} values were compared. For task (iii) it was determined if both the measured and the predicted levels of the plateau pressure $P_{plat}(C_{max})$ were estimated within the initial

pressure range $[P^i_{plat}, P^{i+3}_{plat}]$ or above or below. Accordingly the trends for measured and modeled data were evaluated and compared.

To investigate the capabilities and the potential of the approach, two experimental settings were evaluated based on the data as given in Table 4.2:

Setting 1: Separately for each single patient dataset (i.e. measuring based on a single super syringe maneuver, column "#Testing" in Table 4.2) the three prediction tasks were performed. This experimental setting should confirm the suitability of GP models for the given data.

Setting 2: For each patient, the training set consisted of all patient datasets excluding the dataset of that specific patient, which was used as the test set (columns "#Training" and "#Testing" in Table 4.2). Based on this combination of data, again the prediction tasks (i)-(iii) were performed. This experimental setting should investigate the ability of the approach to detect appropriate models for a specific patient from a general data pool.

System Validation: M5P

As the reference method for prediction of the compliance-pressure curves the M5P algorithm [89] was evaluated. M5P generates a combination of conventional decision trees with linear regression functions as model trees. Such trees contain multivariate linear models in the leaves and represent piecewise linear functions. The training cases are specified by a fixed set of attributes with an associated target value (as for GP modeling). The quality of a model is measured by the accuracy with which it predicts the target values of unseen cases. Given a set of training cases, this set is either associated with a leaf node of the model tree or some test is chosen that splits the set into subsets corresponding to the outcome of the test. The same process is applied recursively to the subsets. Over-elaborated model trees might have to be pruned back. The M5P algorithm used the same input as was used for the GP modeling.

4.2.7 Performance Measures

Prediction task (i): for prediction task (i), the performance of the two statistical modeling techniques GP modeling and the M5P algorithm were measured by the Pearson product-moment correlation coefficient [92]. The coefficient was calculated for the model predicted compliance value C^k and the compliance value C^k which has been estimated by fitting the electrical analog of the spring-and-dashpot model to the measured flow interruption raw data. Other performance measures such as error estimations were supposed to be inadequate as the raw data showed high variability. As two different statistical modeling

techniques were applied for the experimental setups, the correlation coefficient for each of the models was calculated to allow for a comparison of the performance of both techniques.

Prediction task (ii): for prediction task (ii), the percentage difference between the maximum compliance (respectively its corresponding P_{plat}) determined from the raw data and the predicted maximum compliance (respectively P_{plat}) was calculated.

Prediction task (iii): task (iii) was evaluated by the percentage of correct predictions of the trend.

4.3 Results

In the following, the results for the two experimental settings are provided. For both settings the prediction tasks to model and predict compliance-pressure curves (prediction task (i)), to predict the maximum compliance and its corresponding plateau pressure (prediction task (ii)) and to predict whether the instantaneous pressure level should be increased, decreased or retained in order to achieve the maximum compliance (prediction task (iii)) were performed.

The results for the experimental setting (1) are given in Table 4.3. The results for experimental setting (2) are given in Table 4.4. For setting (2), see also Figure 4.2. Results are given as $mean \pm SD$.

4.3.1 Experimental Setting (1)

To remind, in the first experimental setting it should be shown, that GP models are generally suitable to describe the given data. Models were detected and evaluated on the super-syringe measurings for each patient, i.e. model detection and compliance prediction was based on the data of the same patient.

Prediction task (i): the prediction of the compliance-pressure curve by GP modeling reached an averaged correlation coefficient of 0.78 ± 0.16, and the reference method M5P a higher averaged correlation coefficient of 0.92 ± 0.23.

Prediction task (ii): while the predicted maximum compliance C_{max} differed by $9.7\% \pm 6.5\%$ from C_{max} estimated from the raw data, $P_{plat}(C_{max})$ differed by an average of $3.2\% \pm 3.8\%$.

Prediction task (iii): the prediction, if the PEEP should be increased, decreased or retained, was correctly answered in $93.2\% \pm 11.1\%$.

4.3.2 Experimental Setting (2)

Again to remind, in the second experimental setting it should be investigated if the approach is capable to detect appropriate models for a specific patient from a general data pool. Models were detected on a training set and a testing set, the latter obtained from a single patient and not included in the training set.

Prediction task (i): the correlation coefficient of the learned model had an average of 0.34 ± 0.24 for GP modeling and a lower averaged correlation coefficient of 0.18 ± 0.22 for the M5P algorithm as reference method.

Prediction task (ii): the predicted C_{max} differed with an average of $34.3\% \pm 34.3\%$ and predicted $P_{plat}(C_{max})$ with $40.7\% \pm 70.1\%$ from the maximum compliance and corresponding pressure values derived from the raw data.

Prediction task (iii): prediction of the trend for the pressure correction was in 2/3 of the cases ($66.3\% \pm 30.3\%$) correct.

4.4 Discussion

The main results of this study are:

(i) Based on the parametric description of the lung status at bounded pressure-volume ranges the statistical modeling approach of GP is capable, to some extend, of predicting the nonlinear lung compliance over the entire pressure-volume range of the lung capacity.

(ii) The predictive precision has to be improved for a possible clinical application of the approach.

Mechanical ventilation is the life-saving key therapy in intensive care medicine. However, this therapy is in a dilemma since it incorporates both life-saving power and the risk of inducing lung injury as in VALI [32]. Since in 2000 it was for the first time clinically proven that ventilatory settings directly influence patient mortality [109], the development of lung protective ventilation strategies came into the focus of interest in intensive care medicine. VALI is primarily induced by the transfer of high mechanical energy from the ventilator to the patient's lungs. Therefore, the individual adaptation of the energy transfer to the nonlinear volume distensibility (compliance) came into focus when developing new strategies of lung protective ventilation. In this context, the early prediction of effects induced by changing the ventilator settings with respect to respiratory mechanics is a promising approach. This work presents a new approach to predict the lung compliance in mechanically ventilated patients.

Table 4.3: Results for experimental setting 1. Prediction tasks (i)(cf. r_{GP}, r_{M5P}), (ii) (cf. $\Delta C_{max}, \Delta P_{plat}(C_{max})$) and (iii)(cf. $Trend$) separately for each single patient dataset. Note that the poor results for the $Trend$ prediction for patients 2, 10 and 11 are related to poor results in the prediction of the compliance-pressure curve represented by r_{GP}. However, a contrary situation is found for patients 7 and 18.

Patient	*Reference values*		*Task(i)*		*Task(ii)*		*Task(iii)*
	C_{max}	$P_{plat}(C_{max})$	r_{GP}	r_{M5P}	ΔC_{max}	$\Delta P_{plat}(C_{max})$	*Trend*
	mL/mbar mean±SD	mbar mean±SD			%	%	%
1	62.9 ± 0.0	44.0 ± 0.00	0.99	0.98	19.6	0.0	100
2	141.6 ± 1.8	44.8 ± 0.53	0.59	0.99	16.7	0.3	67
3	72.4 ± 0.5	29.2 ± 0.04	0.83	0.98	2.4	3.7	100
4	130.9 ± 1.5	34.0 ± 0.35	0.85	0.98	7.4	7.5	100
5	25.8 ± 0.0	36.7 ± 0.00	0.91	0.94	7.6	0.0	100
6	34.3 ± 0.0	19.7 ± 0.01	0.89	0.96	9.1	5.9	100
7	102.1 ± 0.5	31.5 ± 0.14	0.49	0.98	3.4	0.8	91
8	66.6 ± 0.3	36.1 ± 0.00	0.93	0.99	18.9	0.0	100
9	80.2 ± 0.2	24.8 ± 0.04	0.71	0.99	1.5	3.8	100
10	49.9 ± 0.1	20.4 ± 0.33	0.57	0.97	2.5	14.1	75
11	139.3 ± 0.3	28.3 ± 0.04	0.57	0.97	9.0	5.1	73
12	66.9 ± 0.3	33.9 ± 0.00	0.93	0.98	16.9	0.0	100
13	60.2 ± 0.4	18.9 ± 0.20	0.81	0.98	3.2	5.8	89
14	76.3 ± 0.2	22.7 ± 0.03	0.86	0.94	7.0	6.0	83
15	36.8 ± 0.0	11.4 ± 0.00	0.88	0.00	8.8	3.9	100
16	101.2 ± 0.8	33.6 ± 0.00	0.94	0.99	12.7	0.0	100
17	94.4 ± 0.8	32.6 ± 0.00	0.83	0.96	21.4	0.0	100
18	27.5 ± 0.0	9.1 ± 0.00	0.55	0.96	6.9	0.0	100
mean±SD			0.78±0.16	0.92±0.23	9.7±6.5	3.2±3.8	93.2±11.1

C_{max}: maximum compliance estimated by a polynomial fit to the predicted compliance values for all plateau pressure levels; $P_{plat}(C_{max})$: plateau pressure at which maximum compliance was estimated; r_{GP}, r_{M5P}: Pearson product-moment correlation coefficient of model-predicted and measured compliance values at given plateau pressure levels where the models were learned by GP and M5P, respectively; ΔC_{max}, $\Delta P_{plat}(C_{max})$: percentage difference of predicted and originally measured value of C_{max} and $P_{plat}(C_{max})$, respectively; $Trend$: percentage of correctly predicted position of $P_{plat}(C_{max})$, i.e. for each measured plateau pressure level it was determined if the predicted maximum compliance would be found at a higher, lower or equal pressure level and the percentage of correctly given estimates calculated.

Table 4.4: Results for experimental setting 2. training datasets were built by alternatively leaving out one patient dataset. The left out dataset was used for testing. The table presents the results of the runs for prediction tasks (i)(cf. r_{GP}, r_{M5P}), (ii) (cf. $\Delta C_{max}, \Delta P_{plat}(C_{max})$) and (iii)(cf. $Trend$). Note that the poor results for the $Trend$ prediction for patients 1 and 18 are related to poor results in the prediction of the compliance-pressure curve represented by r_{GP}. However, a contrary situation is found for patients 10 and 13.

Patient	*Reference values*		*Task(i)*		*Task(ii)*		*Task(iii)*
	C_{max}	$P_{plat}(C_{max})$	r_{GP}	r_{M5P}	ΔC_{max}	$\Delta P_{plat}(C_{max})$	*Trend*
	mL/mbar mean±SD	mbar mean±SD			%	%	%
1	48.9 ± 5.0	31.5 ± 0.2	0.02	0.50	37.5	28.6	0
2	92.8 ± 5.0	34.7 ± 2.8	0.47	0.41	23.6	22.2	90
3	73.0 ± 17.1	29.8 ± 2.9	0.23	0.19	1.7	1.9	80
4	98.7 ± 17.8	27.1 ± 1.7	0.25	0.24	30.2	26.4	60
5	40.8 ± 2.8	29.7 ± 12.0	0.32	-0.29	46.2	18.8	67
6	46.2 ± 2.8	26.7 ± 2.9	0.55	-0.12	22.4	27.8	44
7	105.6 ± 11.2	33.9 ± 2.2	0.34	0.16	0.1	6.9	91
8	59.3 ± 5.8	31.2 ± 1.3	0.52	0.32	27.9	13.7	63
9	99.5 ± 9.5	32.8 ± 1.4	0.40	0.29	22.2	37.7	55
10	66.7 ± 5.1	33.3 ± 6.8	-0.04	-0.14	30.3	40.5	56
11	83.5 ± 5.4	32.2 ± 1.9	0.28	0.33	45.4	19.7	67
12	60.8 ± 5.3	33.7 ± 0.6	0.63	0.22	24.4	0.6	100
13	60.5 ± 7.3	28.1 ± 2.3	0.04	-0.07	2.7	39.8	56
14	99.8 ± 20.1	38.3 ± 4.5	0.52	0.06	21.7	78.8	67
15	70.5 ± 0.0	17.7 ± 0.0	0.35	0.35	74.8	61.0	100
16	92.7 ± 8.6	33.4 ± 0.4	0.60	0.43	20.0	0.4	100
17	79.5 ± 3.0	32.6 ± 0.0	0.77	0.34	33.8	0.0	100
18	74.5 ± 12.0	37.1 ± 3.4	-0.09	0.07	152.1	308.4	0
mean±SD			0.34±0.24	0.18±0.22	34.3±34.3	40.7±70.1	66.3±30.3

C_{max}: maximum compliance estimated by a polynomial fit to the predicted compliance values for all plateau pressure levels; $P_{plat}(C_{max})$: plateau pressure at which maximum compliance was estimated; r_{GP}, r_{M5P}: Pearson product-moment correlation coefficient of model-predicted and measured compliance values at given plateau pressure levels where the models were learned by GP and M5P, respectively. Note that a negative correlation coefficient is achieved if the model systematically predicts the opposite to the true target; ΔC_{max}, $\Delta P_{plat}(C_{max})$: percentage difference of predicted and originally measured value of C_{max} and $P_{plat}(C_{max})$, respectively; $Trend$: percentage of correctly predicted position of $P_{plat}(C_{max})$, i.e. for each measured plateau pressure level it was determined if the predicted maximum compliance would be found at a higher, lower or equal pressure level and the percentage of correctly given estimates calculated.

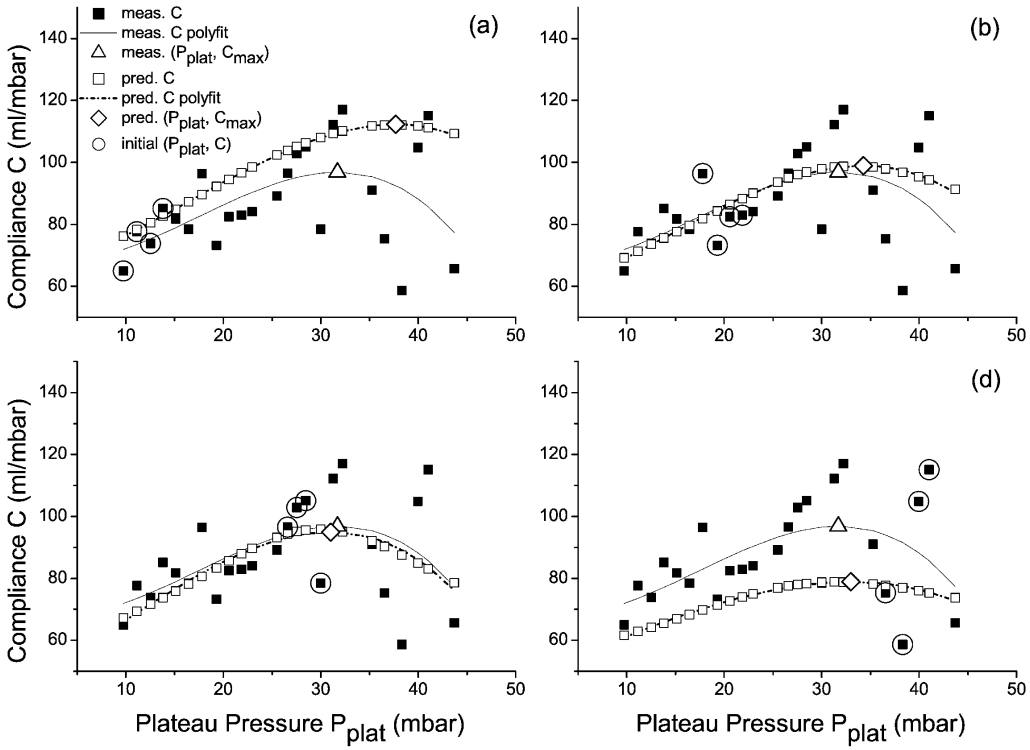

Figure 4.2: Sample results for one patient in experimental setting 2. Predictions of C ($pred.C$) are based on the respiratory status at low (a), intermediate (b), (c) and high (d) levels of P_{plat} (initial (P_{plat}, C)). Each status is quantified by the feature samples of $C^{i..i+3}$, $R^{i..i+3}$, $C_{ve}^{i..i+3}$ and $R_{ve}^{i..i+3}$ corresponding to 4 consecutive plateau pressure levels $P_{plat}^{i..i+3}$, which are tagged by a circle ($initial(P_{plat}, C)$. The variability of the measured (i.e., fitted) C values ($meas.C$) is clearly exhibited. Measured and predicted C_{max} and the corresponding P_{plat} ($meas.$ (P_{plat}, C_{max}), $pred.$ (P_{plat}, C_{max})) are determined by polynomial fits ($meas.$ C $polyfit$, $pred.$ C $polyfit$). Note that for each status the compliance is predicted for pressure values of P_{plat} lower/higher than or within the initial pressure range of the lung status and thus the entire inspiratory pressure-volume range is covered.

Performance within experimental setting (1) confirmed that GP modeling is basically suitable for the present task and problem representation. As hypothesized, the compliance-pressure curves were adequately modeled, having slopes of the first derivative of a sigmoid-like function (Figures 4.2, 1.1). Differentiated characteristics for the individual patient datasets were expressed in differing curve slopes. Comparing the results of GP modeling and M5P for prediction task (i) within settings (1) and (2) leads to the assumption that M5P tends more to overfitting than GP modeling. This was perhaps due to the fact that for the present problem domain, the modeling of functions might provide a higher degree of abstraction and reduce the impact of noise.

Nevertheless, individual compliance-pressure curves for new observations according to setting (2) showed rather poor results. While the prediction of C_{max} and $P_{plat}(C_{max})$ (prediction task (ii)) as well as the prediction of the correct trend for the pressure correction (prediction task (iii)) showed failure rates below 10% in setting (1), which might be sufficiently precise for an indication in medical practice, the results again were impaired within setting (2). Predictions with divergences of more than 30% for C_{max} and $P_{plat}(C_{max})$ and failure rates in a similar range for trend prediction provide at most rough estimates. This implies that learning an individualized model might require an individualized feature selection. Aside from this, further investigation could include the optimization of the kernel parameters and also whether the application of an alternative kernel itself could improve the predictive quality of the models.

The choice of exactly four consecutive levels in airway plateau pressure and their corresponding compliance values as a basis for the prediction of the compliance C^k at plateau pressure level P^k_{plat} was arbitrary. In contrast to numerical models, statistical modeling allows the usage of probabilities, which allow again a high degree of abstraction. For example, the mere distance between the pressure level on which the prediction is based and the level for which the compliance shall be predicted is of no importance. By using statistical modeling techniques, it is possible to relate compliance values to unphysiological pressure levels and vice versa. The combination of four compliance-pressure pairs as abstract descriptor for the lung status allows statistical prediction of compliance values at diverse pressure levels, answering the following question: given a 4-tuple of compliance-pressure relations, what is the most probable compliance value for the specific pressure value P^k_{plat}? An increasing number of these compliance-plateau pressure pairs could improve the predictive accuracy, as the lung status itself would be described in a more differentiable way. However, such an increase implies that an expanded pressure-volume range has to be measured before a prediction can be performed. This again means a longer interruption of the regular breathing and possibly higher pressures to be applied, which is potentially contradictory to the concept of lung protective ventilation. We assume that it therefore might be necessary to include other features for the prediction task. This could be additional numerical features such as the arterial oxygen saturation, which in clinical practice is often used by the physician when adjusting the ventilator (regardless of a lung protective ventilation strategy). Moreover, demographic data of the individual patient or the medical history of

the patient might be helpful for an individualized optimization.

4.5 Conclusions

We introduced a statistical modeling approach based on GP modeling to predict the non-linear compliance of the human lung. The results indicate that the combination of classical model fitting and statistical modeling is generally capable of solving this task. However, the results of the experiments which were run on data obtained from patients suffering from ARDS show that the precision of the prediction needs to be improved. We assume that individualized feature selection as a preprocessing step could be helpful and should be brought into focus in future efforts.

Chapter 5

Summary

"What we would really like to know about a function-finding program is not its record of successes on artificial problems chosen by the programmer, but its likelihood of success on a new problem generated in a prespecified environment and involving real scientific data." (Cullen Schaffer [96], 1990, p. 828). This statement implies a considerable part of the motivation to this work. It is the opinion of the author that a large variety of brilliant algorithms have been developed to model a wide range of problems. But that not very many of them have been applied and validated on real-world data in a real-world problem domain. Therefore, it was the intention of this work to apply approaches from KDD and ML in a real-world problem domain on measured scientific data from a medical problem domain.

"The most common assumption is that the process [as part of a living system to be modeled] and its components or sub-processes behave in a linear manner, and that their basic characteristics do not change over time. Of course, most living systems change over time, are adaptive, and are often nonlinear." (John Semmlow [102], 2005, p. 9). This simple statement describes very concisely one of the main challenges when modeling the mechanically ventilated human lung. The lung has to be considered as a most dynamical system which by no means shows a linear and time-invariant behavior. Applying mechanical stress to the lung tissue – as under the condition of mechanical ventilation – induces restructuring processes of the lung with its seemingly viscous structures. While the mechanical behavior of the lung changes over time, the specific reasons for these changes are very difficult to determine, if known at all. In medical practice, only few mechanical parameters are observed. However, even under the risk of inducing a lung damage, mechanical ventilation is the live-saving therapy in intensive care medicine and the improvement of lung protective ventilation strategies is of actual interest in medical research.

In the introduction of this thesis, three questions were formulated: How can we get more insight into lung mechanics? How can we improve the corresponding techniques? How can

we predict mechanical lung parameters relevant for lung protective ventilation? Assuming that the basic characteristics of the processes to model do not change over time, linear time-invariant models allow to apply the powerful mathematical tools of linear system analysis. However, as mentioned above, the human lung is a highly dynamic, nonlinear system. The idea to work on the above questions was to combine classical system analysis from the field of biomedical engineering with statistical modeling techniques known from KDD and ML. Modeling techniques providing a high degree of abstract representation were chosen as, related to the statements above, it was our intention to apply techniques deviating from the classical approaches.

This thesis presents three studies to investigate real-world time series data obtained from mechanically ventilated patients. The super syringe maneuver and the PEEP wave maneuver were applied as automated and standardized methods to record pressure-, flow- and volume-data. Such maneuvers were operated on two groups of patients, such suffering from ARDS and such with healthy lungs.

In the first study, the focus was on the nonlinear behavior of the lung. To investigate the nonlinearity of the lung compliance, a classical approach from biomedical engineering was applied. The method used the analog spring-and-dashpot model, itself describing a linear behavior. This model is assumed to depict viscoelastic effects. Using this model piecewise on separated pressure levels to analyze stress relaxation processes of the lung, it was possible to quantitatively describe a nonlinear compliance-pressure relation. Moreover, it could be shown that stress relaxation processes are independent of the applied pressure level.

The first study focused on the very special aspect of nonlinear mechanical properties of the lung. Such properties could be very well approached with techniques from biomedical engineering. The spring-and-dashpot model is very well capable to model stress relaxation processes in the lung. However, it does not represent other aspects of lung mechanics and it is in particular not a model of the complete respiratory system. To get such a general model, sub-models as represented by the spring-and-dashpot model have to be combined to better understand the overall picture.

In the second study, we presented a semiautomatic ED approach to create lung models "from scratch". The focus here was set on a technique being able to handle background knowledge. This property could be most helpful to gain more insight into different aspects of lung mechanics. The applied system was derived from an established ED system and was capable to extract the well-known equation of motion model for the respiratory system directly from the measured real-world time series data.

Being aware of the nonlinear compliance-pressure relation, the third study used GP modeling as technique to statistically model the compliance-pressure relation of the lung. Based on features obtained from stress relaxation analyses, the compliance-pressure curve spanning the complete inspiratory capacity, i.e. the complete pressure-volume range of the

lung, was predicted. In medical practice, an essential procedure in terms of lung protective ventilation is to optimize the PEEP setting. At an optimized PEEP, the lung is ventilated with a maximized compliance, meaning that the volume is inflated with a minimum of pressure applied. In turn this leads to a reduction of the mechanical stress affecting the lung. The prediction of the compliance for specific pressure values could support the physician at the bedside when making decisions with respect to the PEEP setting. As results show, the approach basically is able to perform such predictions. However, for a bedside assistance the procedure has to be improved. We assume that the approach might benefit from an individualized feature selection.

This thesis is application-driven in the sense that different techniques have been utilized to analyze real-world data from a medical problem domain. Concerning the questions posed at the beginning, techniques from knowledge discovery in databases and machine learning in combination with classical modeling techniques from biomedical engineering were capable to gain insight into lung mechanics. In order to model lung mechanics we modified an existing algorithm and were able to successfully apply it on real-world data. And finally, even if this approach has to be improved, we were able to apply a statistical modeling technique to predict mechanical lung parameters.

This work was strongly motivated by the aim to gain insight into lung mechanics and to show the potential of the applied techniques. From the medical point of view the common theme was the lung protective ventilation of the patient. From the technological side the common theme was to investigate methods with focus on their practical applicability. While precise feature extraction is very well supported by biomedical engineering, the task to detect new models with little prior knowledge as a highly explorative process is strongly supported by techniques from knowledge discovery in databases. From the resulting new insights, again classical models could be established. Always focusing on the practical usability of the applied methods, we see a great potential in the combination of rather descriptive techniques from biomedical engineering and more explorative methods from knowledge discovery in databases for future work in this medical problem domain.

Appendix A

Chapter 2, Equation 2.2: Derivation of Laplace Transform

The representation of the electrical circuit (Figure 1.3) as Laplace transform leads to Equation (2.2), where the pressure P of the respiratory system is presented by the numerator and the respiratory flow $\dot{V}$ by the denominator. This equation is derived as follows.

The model can be described by two differential equations, where $\dot{V}_{ve,C}$ denotes the flow conducted through the capacitor C_{ve}:

$$P = R\dot{V} + \frac{1}{C}\int \dot{V}dt + \frac{1}{C_{ve}}\int \dot{V}_{ve,C}dt \tag{A.1}$$

$$R_{ve}(\dot{V} - \dot{V}_{ve,C}) = \frac{1}{C}_{ve}\int \dot{V}_{ve,C}dt \tag{A.2}$$

Differentiation of Equation (A.1) by time and resolving for $\dot{V}_{ve,C}$ leads to

$$\begin{aligned} \frac{\partial P}{\partial t} &= R\frac{\partial \dot{V}}{\partial t} + \frac{1}{C}\dot{V} + \frac{1}{C_{ve}}\dot{V}_{ve,C} \\ \Leftrightarrow \dot{V}_{ve,C} &= C_{ve}(\frac{\partial P}{\partial t} - R\frac{\partial \dot{V}}{\partial t} - \frac{1}{C}\dot{V}) \end{aligned} \tag{A.3}$$

Differentiation of Equation (A.2) by time leads to

$$R_{ve}\frac{\partial \dot{V}}{\partial t} - R_{ve}\frac{\dot{V}_{ve,C}}{\partial t} = \frac{1}{C_{ve}}\dot{V}_{ve,C} \tag{A.4}$$

Inserting Equation (A.3) into Equation (A.4) we get

$$\begin{aligned} R_{ve}\frac{\partial \dot{V}}{\partial t} - R_{ve}(C_{ve}\frac{\partial^2 P}{\partial t^2} - RC_{ve}\frac{\partial^2 \dot{V}}{\partial t^2} - \frac{C_{ve}}{C}\frac{\partial \dot{V}}{\partial t}) &= \frac{\partial P}{\partial t} - R\frac{\partial \dot{V}}{\partial t} - \frac{1}{C}\dot{V} \\ \Leftrightarrow RR_{ve}C_{ve}\frac{\partial^2 \dot{V}}{\partial t^2} + (R + R_{ve} - \frac{R_{ve}C_{ve}}{C})\frac{\partial \dot{V}}{\partial t} + \frac{1}{C}\dot{V} &= R_{ve}Cve\frac{\partial^2 P}{\partial t^2} + \frac{\partial P}{\partial t} \end{aligned}$$

And finally, Laplace transformation leads to

$$RR_{ve}C_{ve}s^2\dot{V}(s) + (R + R_{ve} - \frac{R_{ve}C_{ve}}{C})s\dot{V}(s) + \frac{1}{C}\dot{V}(s) = R_{ve}Cves^2P(s) + sP(s)$$

$$\Leftrightarrow \frac{P(s)}{\dot{V}(s)} = \frac{RCR_{ve}C_{ve}s^2 + (RC + CR_{ve} + R_{ve}C_{ve})s + 1}{CR_{ve}C_{ve}s^2 + Cs}$$

Appendix B

Chapter 2, Section 2.3.3: RMSE for Two-Point and Multiple-Point Analysis

Figure B.1 shows the benefit of multiple-point methods in comparison to conventional two-point methods. The root mean squared error (RMSE) as rate for the deviation between measured and re-calculated data was lower when using the multiple-point methods. This holds for the data of both patient groups: for the control group, the RMSE was reduced by 31% and for the ARDS group by even 55% when applying the multiple-point method instead of the two-point method.

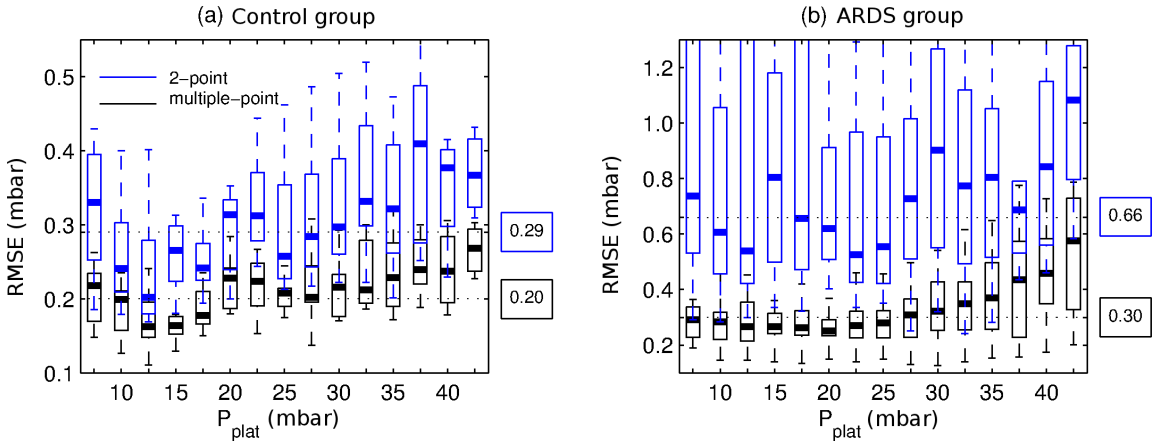

Figure B.1: The root mean squared error (RMSE) for two-point and multiple-point analysis. The measured pressure curves were compared to re-calculated pressure curves. The RMSE as rate for the deviation between measured and re-calculated curves was computed. The re-calculated pressure curve was obtained by substituting the parameters C, R, C_{ve} and R_{ve} derived from both two-point and multiple-point method into the spring-and-dashpot model. C, R, C_{ve} and R_{ve} were linearly interpolated in steps of 2.5 mbar within a pressure range between 7.5 and 42.5 mbar (Figure 2.3). Interpolated parameter values beyond the 1.5 fold of the interquartile range were eliminated as outliers. For each pressure level, the four parameters were set as input for the spring-and-dashpot model. The curve was computed with the inspiratory airflow-rate as set for the patient-measuring during a super syringe maneuver (constant inspiratory flow-rates and zero flow rates during occlusion for all inflation steps within each maneuver). Both boxplots (a) and (b) present the data for each pressure level as median with the lower and upper quartile. (a) RMSE of pressure curves for interpolated pressure levels for the control group. (b) RMSE of pressure curves for interpolated pressure levels for the ARDS group. Values on the right side of both diagrams indicate the overall medians.

Appendix C

Chapter 3, Algorithm 3.8: Impact of Ordered Refinement of Beam Elements to the System Performance

The ordered refinement of beam elements in Algorithm 3.8, line 7, indeed can impact the system performance. However, this occurs only under artificial premises as the following example shows.

We refer to the examples on pages 64 and 66. The beam width is set to $w_b = 2$. Recall that beam $(B, <)$ is ordered as for $b_i \in B < b_j \in B :\Leftrightarrow F_{\mathrm{h}}(b_i) > F_{\mathrm{h}}(b_j)$. We arbitrarily assume that $F_{\mathrm{h}}(\mathcal{T}_0) = F_{\mathrm{h}}(\mathcal{T}_1) = 4$, $r(\mathcal{T}_0) = \{\mathcal{T}_2, \mathcal{T}_3\}$ with $F_{\mathrm{h}}(\mathcal{T}_2) = F_{\mathrm{h}}(\mathcal{T}_3) = 2$ and $r(\mathcal{T}_1) = \{\mathcal{T}_4, \mathcal{T}_5\}$ with $F_{\mathrm{h}}(\mathcal{T}_4) = F_{\mathrm{h}}(\mathcal{T}_5) = 2$. We jump into a notional run of the search at start of iteration n with $(B, <) = \{\mathcal{T}_0, \mathcal{T}_1\}$.

In the first run, the beam elements $b_i \in (B, <)$ are refined in a left to right sequence. We get the following iteration:

Start of iteration n: $(B, <) = \{\mathcal{T}_0, \mathcal{T}_1\}$.

- Copy $(B, <)$ to $(B, <)'$
- Refine all $b_i \in (B, <)'$ and get
  - $r(\mathcal{T}_0, <^r) = \{\mathcal{T}_2, \mathcal{T}_3\}$
  - Replace $\mathcal{T}_0$ by $\mathcal{T}_2$, sort $(B, <)$ to $\{\mathcal{T}_1, \mathcal{T}_2\}$
  - Replace $\mathcal{T}_1$ by $\mathcal{T}_3$, sort $(B, <)$ to $\{\mathcal{T}_2, \mathcal{T}_3\}$
  - $r(\mathcal{T}_1, <^r) = \{\mathcal{T}_4, \mathcal{T}_5\}$
  - Reject $\mathcal{T}_4$ and $\mathcal{T}_5$ as both do not perform better than $b_1 = \mathcal{T}_2$

End of iteration n: $(B, <) = \{\mathcal{T}_2, \mathcal{T}_3\}$.

In the second run, the beam elements $b_i \in (B, <)$ are refined in a right to left sequence. We get the following iteration:

Start of iteration n: $(B, <) = \{\mathcal{T}_0, \mathcal{T}_1\}$.

- Copy $(B, <)$ to $(B, <)'$
- Refine all $b_i \in (B, <)'$ and get
  - $r(\mathcal{T}_1, <^r) = \{\mathcal{T}_4, \mathcal{T}_5\}$
  - Replace $\mathcal{T}_0$ by $\mathcal{T}_4$, sort $(B, <)$ to $\{\mathcal{T}_1, \mathcal{T}_4\}$
  - Replace $\mathcal{T}_1$ by $\mathcal{T}_5$, sort $(B, <)$ to $\{\mathcal{T}_4, \mathcal{T}_5\}$
  - $r(\mathcal{T}_0, <^r) = \{\mathcal{T}_2, \mathcal{T}_3\}$

- Reject $\mathcal{T}_2$ and $\mathcal{T}_3$ as both do not perform better than $b_1 = \mathcal{T}_4$

End of iteration n: $(B, <) = \{\mathcal{T}_4, \mathcal{T}_5\}$.

As can be easily seen, the beam differs after iteration n when refining beam elements in a left to right sequence and when refining in a right to left sequence. While in the first run $\mathcal{T}_0$ and $\mathcal{T}_1$ where replaced by the two successors of $\mathcal{T}_0$, i.e. $\mathcal{T}_2$ and $\mathcal{T}_3$ respectively, in run 2 $\mathcal{T}_0$ and $\mathcal{T}_1$ were replaced by the two successors of $\mathcal{T}_1$, i.e. $\mathcal{T}_4$ and $\mathcal{T}_5$. However, this is under the arbitrary assumption that several refinements provide the same error. In general, the impact of this ordering is of minor impact. In the following we will prove that within all refinements of all elements of a beam, always the best performing elements, i.e. those with the lowest error according to F_h are presented in the beam after evaluation of all elements.

Hypothesis. *Applying Algorithm 3.8, within all refinements $r(b_1), \ldots, r(b_n)$ of beam elements $b_1 \in (B, <), \ldots, b_n \in (B, <)$ of a sorted beam $(B, <)$ (Definition 8), always the best performing elements with reference to the error F_h are inserted into the beam.*

Proof by cases

Given	*Beam*:	$\{b_1, \ldots, b_n\} \in B$ of $(B, <)$ before iteration
	A:	refinements $\{a_1, \ldots, a_k\} = r(b_i) \in$ Beam
	C:	refinements $\{c_1, \ldots, c_l\} = r(b_j) \in$ Beam
	Beam′:	$\{b'_1, \ldots, b'_m\} \in B$ of $(B, <)$ after iteration
Assumption	$\exists c \in C, \forall a \in A : F_\mathrm{h}(c) < F_\mathrm{h}(a)$	
	$c \notin Beam'$	
Then	$\forall b' \in Beam' : F_\mathrm{h}(b') \leq F_\mathrm{h}(c)$	
Case 1	$\nexists b' \in Beam' : b' \in A \Leftrightarrow \forall b' \in Beam', \forall a \in A : F_\mathrm{h}(b') < F_\mathrm{h}(a)$	
Case 2	$\exists b' \in Beam' : b' \in A \Leftrightarrow \exists b' \in Beam', \exists a \in A : F_\mathrm{h}(b') = F_\mathrm{h}(a) \not\leq F_\mathrm{h}(c)$	

□

Appendix D

Chapter 4, Section 4.2.4: Calculation of Tracheal Pressure

The mere tracheal pressure P_{trach} is estimated by subtracting the pressure drop ΔP_{ETT}, which occurs at the endotracheal tube ETT [44], from the airway pressure P_{aw}:

$$P_{trach} = P_{aw} - \Delta P_{ETT}$$

With respect to the linearity of the airflow, this pressure drop can be calculated in different ways [44]. The Rohrer-equation [46, 93] which combines a linear and a quadratic pressure-flow relation is a well accepted estimate and was applied in this study:

$$\Delta P_{ETT} = K_1 \times \dot{V} + K_2 \times \dot{V}^2$$

Table D.1: Rohrer-coefficients K_1 and K_2 for calculation of tracheal pressure for each patient. Values are taken from [46] for tube types of original length (i.e. uncut). For patients characteristics cf. Table 1.2.

No.	*Diameter* (mm)	K_1 $(mbar \times s/L)$	K_2 $(mbar \times s^2/L^2)$
1	10.0	0.3	1.6
2	8.0	1.7	5.5
3	8.0	1.7	5.5
4	8.5	0.8	4.4
5	9.0	0.7	3.7
6	7.0	1.4	10.1
*7	–	–	–
8	8.0	1.7	5.5
9	8.5	0.8	4.4
10	9.0	0.7	3.7
11	9.0	0.7	3.7
12	8.0	1.7	5.5
13	10.0	0.3	1.6
*14	–	–	–
†15	9.5	0.7	3.7
16	10.0	0.3	1.6
17	9.0	0.7	3.7
18	8.0	1.7	5.5
19	8.0	1.7	5.5
20	8.0	1.7	5.5

∗: Tube diameter not documented and therefore patient excluded from study; ∗: Rohrer-coefficients for type diameter 9.5 not documented, therefore values from diameter 9 mm used for approximation.

Bibliography

[1] V. Antonaglia, A. Peratoner, L. De Simoni, A. Gullo, J. Milic-Emili, and W. Zin. Bedside assessment of respiratory viscoelastic properties in ventilated patients. *European Respiratory Journal*, 16(2):302–308, 2000.

[2] D. G. Ashbaugh, D. B. Bigelow, T. L. Petty, and B. E. Levine. Acute respiratory distress in adults. *The Lancet*, 2:319–323, 1967.

[3] J. H. T. Bates. A recruitment model of quasi-linear power-law stress adaptation in lung tissue. *Annals of Biomedical Engineering*, 35(7):1165–1174, 2007.

[4] J. H. T. Bates and C. G. Irvin. Time dependence of recruitment and derecruitment in the lung: a theoretical model. *Journal of Applied Physiology*, 93:705–713, 2002.

[5] J. H. T. Bates, A. Rossi, and J. Milic-Emili. Analysis of the behavior of the respiratory system with constant inspiratory flow. *Journal of Applied Physiology*, 58(6): 1840–1848, 1985.

[6] J. H. T. Bates, K. A. Brown, and T. Kochi. Identifying a model of respiratory mechanics using the interrupter technique. In *Proceedings of the Ninth Annual Conference of the IEEE Engineering in Medicine and Biology Society*, pages 1802–1803, 1987.

[7] J. H. T. Bates, I. W. Hunter, P. D. Sly, S. Okubo, S. Filiatrault, and J. Milic-Emili. Effect of valve closure time on the determination of respiratory resistance by flow interruption. *Medical and Biological Engineering and Computing*, 25(2):136–140, 1987.

[8] J. H. T. Bates, P. Baconnier, and J. Milic-Emili. A theoretical analysis of interrupter technique for measuring respiratory mechanics. *Journal of Applied Physiology*, 64(5): 2204–2214, 1988.

[9] J. H. T. Bates, K. A. Brown, and T. Kochi. Respiratory mechanics in the normal dog determined by expiratory flow interruption. *Journal of Applied Physiology*, 67 (6):2276–2285, 1989. journal Article Research Support, Non-U.S. Gov't United states 1985).

[10] J. H. T. Bates, G. N. Maksym, D. Navajas, and B. Suki. Lung tissue rheology and 1/f noise. *Annals of Biomedical Engineering*, 22(6):674–681, 1994.

[11] L. Beydon, C. Svantesson, K. Brauer, F. Lemaire, and B. Jonson. Respiratory mechanics in patients ventilated for critical lung disease. *European Respiratory Journal*, 9(2):262–273, 1996. Journal Article Research Support, Non-U.S. Gov't Denmark official journal of the European Society for Clinical Respiratory Physiology.

[12] L. Beydon, C. Svantesson, K. Brauer, F. Lemaire, and B. Jonson. Respiratory mechanics in patients ventilated for critical lung disease. *European Respiratory Journal*, 9(2):262–273, 1996.

[13] R. Bisiani. Beam search. In S. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 56–58. John Wiley & Sons, 1987.

[14] G. L. Bradshaw, P. Langley, and H. A. Simon. BACON.4: The discovery of intrinsic properties. In *Proceedings of the Third Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, pages 19–25, 1980.

[15] W. Bridewell, N. B. Asadi, P. Langley, and L. Todorovski. Reducing overfitting in process model induction. In *Proceedings of the Twenty-Second International Conference on Machine Learning*, pages 81–88, New York, NY, 2005. ACM.

[16] W. Bridewell, P. Langley, L. Todorovski, and S. Džeroski. Inductive process modeling. *Machine Learning*, 71:1–32, 2008.

[17] J. X. Brunner and G. Wolff. *Pulmonary function indices in critical care patients*. Springer, Berlin, 1988.

[18] K. P. Chan, T. E. Stewart, and S. Mehta. High-frequency oscillatory ventilation for adult patients with ARDS. *Chest*, 131(6):1907–1916, 2007.

[19] D. Chiumello, E. Carlesso, P. Cadringher, P. Caironi, F. Valenza, F. Polli, F. Tallarini, P. Cozzi, M. Cressoni, A. Colombo, J. J. Marini, and L. Gattinoni. Lung stress and strain during mechanical ventilation for acute respiratory distress syndrome. *American Journal of Respiratory and Critical Care Medicine*, 178:346–355, 2008.

[20] S. Cook. The complexity of theorem proving procedures. In *Proceedings of the Third Annual ACM Symposium on Thoery of Computing*, pages 151–158, 1971.

[21] E. D'Angelo, E. Calderini, G. Torri, F. M. Robatto, D. Bono, and J. Milic-Emili. Respiratory mechanics in anesthetized paralyzed humans: effects of flow, volume, and time. *Journal of Applied Physiology*, 67(6):2556–2564, 1989. journal Article, Research Support, Non-U.S. Gov't, United states, 1985).

[22] E. D'Angelo, E. Calderini, M. Tavola, D. Bono, and J. Milic-Emili. Effect of PEEP on respiratory mechanics in anesthetized paralyzed humans. *Journal of Applied Physiology*, 73(5):1736–1742, 1992.

[23] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the Association for Computing Machinery (JACM)*, 7(3):201–215, 1960.

[24] D. J. W. DePauw and B. DeBaets. Incorporating model identifiability into equation discovery of ODE systems. In *Proceedings of the 2008 GECCO conference companion on Genetic and evolutionary computation, Atlanta, GA*, pages 2135–2140, New York, NY, 2008. Association of Computing Machinery.

[25] Dräger Medical AG & Co. KGaA. Evita 4 Lab - PC - Fernsteuerung für Messmanöver und spezielle Beatmungsmuster in Evita 4, Gebrauchsanweisung. 2001.

[26] D. Dreyfuss and G. Saumon. Ventilator-induced lung injury: lessons from experimental studies. *American Journal of Respiratory and Critical Care Medicine*, 157: 294–323, 1998.

[27] S. Džeroski and L. Todorovski. Discovering dynamics: From inductive logic programming to machine discovery. *Journal of Intelligent Information Systems*, 4:89–108, 1994.

[28] C. Edibam, A. J. Rutten, D. V. Collins, and A. D. Bersten. Effect of inspiratory flow pattern and inspiratory to expiratory ratio on nonlinear elastic behavior in patients with acute lung injury. *American Journal of Respiratory and Critical Care Medicine*, 167:702–707, 2003.

[29] N. Eissa, V. Ranieri, C. Corbeil, M. Chasse, J. Braidy, and J. Milic-Emili. Effect of PEEP on the mechanics of the respiratory system in ards patients. *Journal of Applied Physiology*, 73(5):1728–1735, 1992.

[30] N. T. Eissa, V. M. Ranieri, C. Corbeil, M. Chasse, F. M. Robatto, J. Braidy, and J. Milic-Emili. Analysis of behavior of the respiratory system in ARDS patients: effects of flow, volume, and time. *Journal of Applied Physiology*, 70(6):2719–2729, 1991.

[31] B. C. Falkenhainer and R. S. Michalski. Integrating quantitative and qualitative discovery in the ABACUS system. In *Machine Learning: An Artificial Intelligence Approach*, pages 153–190. Morgan Kaufman, San Mateo, CA, 1990.

[32] J. A. Frank and M. A. Matthay. Science review: mechanisms of ventilator-induced injury. *Critical Care*, 7:702–707, 2003. 233-241.

[33] Y. C. Fung. *Biomechanics. Mechanical Properties of Living Tissues*. Springer, New York, 1981.

[34] Y. C. Fung. Structure and stress-strain relationship of soft tissues. *American Zoologist*, 24:13–22, 1984.

[35] S. Ganzert, J. Guttmann, K. Kersting, R. Kuhlen, C. Putensen, M. Sydow, and S. Kramer. Analysis of respiratory pressure-volume curves in intensive care medicine using inductive machine learning. *Artificial Intelligence in Medicine*, 26(1-2):69–86, 2002.

[36] S. Ganzert, S. Kramer, K. Möller, D. Steinmann, and J. Guttmann. Prediction of mechanical lung parameters using Gaussian process models. In C. Combi, Y. Shahar, and A. Abu-Hanna, editors, *Artificial Intelligence in Medicine. Proceedings of the 12th Conference on Artificial Intelligence in Medicine, AIME 2009*, volume 5651 of *Lecture Notes in Artificial Intelligence*, pages 380–384, Berlin/Heidelberg, 2009. Springer.

[37] S. Ganzert, K. Möller, S. Kramer, K. Kersting, and J. Guttmann. Identifying mathematical models of the mechanically ventilated lung using equation discovery. In W. Schlegel and O. Dössel, editors, *IFMBE Proceedings WC 2009 "World Congress on Medical Physics and Biomedical Engineering"*, volume 25 of *International Federation for Medical and Biological Engineering Proceedings*, pages 1524–1527, Berlin/Heidelberg, 2009. Springer.

[38] S. Ganzert, K. Möller, D. Steinmann, S. Schumann, and J. Guttmann. Pressure-dependent stress relaxation in acute respiratory distress syndrome and healthy lungs: an investigation based on a viscoelastic model. *Critical Care*, 13(6):R199, 2009.

[39] S. Ganzert, J. Guttmann, D. Steinmann, and S. Kramer. Equation discovery for model identification in respiratory mechanics of the mechanically ventilated human lung. In B. Pfahringer, G. Holmes, and A. Hoffman, editors, *Proceedings of the 13th Conference on Discovery Science, DS-2010*, volume 6332 of *Lecture Notes in Artificial Intelligence*, Berlin/Heidelberg, 2010. Springer.

[40] S. Ganzert, S. Kramer, and J. Guttmann. Predicting the lung compliance of mechanically ventilated patients via statistical modeling. *Physiological Measurement*, 33:345–359, 2012.

[41] L. Gattinoni and A. Pesenti. The concept of "baby lung". *Intensive Care Medicine*, 31:776–784, 2005.

[42] S. Grasso, P. Terragni, L. Mascia, V. Fanelli, M. Quintel, P. Herrmann, G. Hedenstierna, A. S. Slutsky, and V. M. Ranieri. Airway pressure-time curve profile (stress index) detects tidal recruitment/hyperinflation in experimental acute lung injury. *Critical Care Medicine*, 32(4):1018–1027, 2004.

[43] J. Guttmann. *Nichtlinearität von Compliance und Resistance unter mechanischer Beatmung*. Habilitation treatise, University Medical Center, Basel, June 1993.

[44] J. Guttmann, L. Eberhard, B. Fabry, W. Bertschmann, and G. Wolff. Continuous calculation of intratracheal pressure in tracheally intubated patients. *Anesthesiology*, 79(3):503–513, 1993.

[45] J. Guttmann, L. Eberhard, B. Fabry, D. Zappe, H. Bernhard, Lichtwarck-Aschoff, M. Adolph, and G. Wolff. Determination of volume-dependent respiratory system mechanics in mechanically ventilated patients using the new SLICE method. *Technology and Health Care*, 2:175–191, 1994.

[46] C. Haberthür, J. Guttmann, P. M. Osswald, and M. Schweitzer. *Beatmungskurven - Kursbuch und Atlas*. Springer, Berlin/Heidelberg, 2001.

[47] K. Hickling, S. Henderson, and R. Jackson. Low mortality associated with low volume pressure limited ventilation with permissive hypercapnia in severe adult respiratory distress syndrome. *Intensive Care Medicine*, 16:372–377, 1990.

[48] K. G. Hickling. The pressure-volume curve is greatly modified by recruitment. A mathematical model of ARDS lungs. *American Joural of Respiratory and Critical Care Medicine*, 158:194–202, 1998.

[49] L. Holzapfel, D. Robert, F. Perrin, P. L. Blanc, B. Palmier, and C. Guerin. Static pressure-volume curves and effect of positive end-expiratory pressure on gas exchange in adult respiratory distress syndrome. *Critical Care Medicine*, 11(8):591–597, 1983.

[50] J. J. Hotchkiss, L. Blanch, G. Murias, A. Adams, D. Olson, O. Wangensteen, P. Leo, and J. Marini. Effects of decreased respiratory frequency on ventilator-induced lung injury. *Critical Care Medicine*, 161(2):463–468, 2000.

[51] R. Hughes, A. J. May, and W. J. G. Stress relaxation in rabbits' lungs. *The Journal of Physiology*, 146:85–97, 1959.

[52] E. Ingenito, L. Mark, and B. Davison. Effects of acute lung injury on dynamic tissue properties. *Journal of Applied Physiology*, 77(6):2689–2697, 1994.

[53] B. Jonson, L. Beydon, K. Brauer, C. Mansson, S. Valind, and H. Grytzell. Mechanics of respiratory system in healthy anesthetized humans with emphasis on viscoelastic properties. *Journal of Applied Physiology*, 75(1):132–140, 1993.

[54] V. Kessler, G. Mols, H. Bernhard, C. Haberthur, and J. Guttmann. Interrupter airway and tissue resistance: errors caused by valve properties and respiratory system compliance. *Journal of Applie Physiology*, 87(4):1546–1554, 1999.

[55] T. Kochi, S. Okubo, W. A. Zin, and J. Milic-Emili. Flow and volume dependence of pulmonary mechanics in anesthetized cats. *Journal of Applied Physiology*, 64 (1):441–450, 1988. journal Article Research Support, Non-U.S. Gov't United states 1985).

[56] B. W. Koehn and J. M. Żytkow. Experimenting and theorizing in theory formation. In *Proceedings of the ACM SIGART International Symposium on Methodologies for Intelligent Systems*, pages 296–307, New York, NY, 1986. ACM.

[57] M. M. Kokar. Determining arguments of invariant functional descriptions. *Machine Learning*, 1(4):403–422, 1986.

[58] D. G. Krige. A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Africa*, 52(6):119–139, 1951.

[59] V. Križman, S. Džeroski, and B. Kompare. Discovering dynamics from measured data. In *Working Notes of the MLnet Workshop on Statistics, Machine Learning and Knowledge Discovery in Databases*, pages 191–198. Electrotechnical Review, 1995.

[60] J. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright. Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9(1):112–147, 1998.

[61] P. Langley. BACON.1: A general discovery system. In *Proceedings of the Second National Conference of the Canadian Society for Computational Studies of Intelligence*, pages 174–182, 1978.

[62] P. Langley and J. M. Żytkow. Data-driven approaches to empirical discovery. *Artificial Intelligence*, 40:283–310, 1989.

[63] P. Langley, G. L. Bradshaw, and H. A. Simon. Heuristics for empirical discovery. In L. Bolc, editor, *Computational models of learning*, Springer Series In Symbolic Computation-Artificial Intelligence, pages 21–54. Springer, London, UK, 1987.

[64] P. Langley, J. Sanchez, L. Todorovski, and S. Džeroski. Inducing process models from continuous data. In *Proceedings the Nineteenth International Conference on Machine Learning*, pages 347–354, Sydney, 2002. Morgan Kaufmann.

[65] P. W. Langley. BACON: A production system that discovers empirical laws. In *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, page 344. William Kaufmann, 1977.

[66] N. R. Macintyre. Basic principles and new modes of mechanical ventilation. In *Critical Care Medicine: Perioperative Management*, pages 447–459. Lippincott Williams & Wilkins, Philadelphia, PA, 2002.

[67] D. J. C. Mackay. Introduction to Gaussian processes. In *Neural Networks and Machine Learning*, volume 168 of *NATO ASI Series*, pages 133–165. 1998.

[68] G. N. Maksym and J. H. T. Bates. A distributed nonlinear model of lung tissue elasticity. *Journal of Applied Physiology*, 82(1):32–41, 1997.

[69] D. Matamis, F. Lemaire, A. Harf, C. Brun-Buisson, J. C. Ansquer, and G. Atlan. Total respiratory pressure-volume curves in the adult respiratory distress syndrome. *Chest*, 86(1):58–66, 1984.

[70] D. Matamis, F. Lemaire, A. Harf, C. Brun-Buisson, J. C. Ansquer, and G. Atlan. Total respiratory pressure-volume curves in the adult respiratory distress syndrome. *Chest*, 86(1):58–66, 1984.

[71] J. Milic-Emili, F. M. Robatto, and J. H. T. Bates. Respiratory mechanics in anaesthesia. *British Journal of Anaesthesia*, 65(1):4–12, 1990.

[72] T. M. Mitchell. The need for biases in learning generalizations. Technical Report CBM-TR-117, Rutgers University, New Brunswick, NJ, 1980.

[73] T. M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[74] G. Mols, I. Brandes, V. Kessler, M. Lichtwarck-Aschoff, T. Loop, K. Geiger, and J. Guttmann. Volume-dependent compliance in ARDS: proposal of a new diagnostic concept. *Intensive Care Medicine*, 25(10):1084–91, 1999.

[75] G. Mols, H.-J. Priebe, and J. Guttmann. Alveolar recruitment in acute lung injury. *British Journal of Anaesthesia*, 96(2):156–166, 2006.

[76] L. E. Mount. The ventilation flow-resistance and compliance of rat lungs. *Journal of Physiology*, 127(1):157–167, 1955.

[77] S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19-20:629–679, 1994.

[78] D. Navajas, G. N. Maksym, and J. H. T. Bates. Dynamic viscoelastic nonlinearity of lung parenchymal tissue. *Journal of Applied Physiology*, 79(1):348–356, 1995.

[79] C. Nédellec, C. Rouveirol, H. Adé, F. Bergadano, and B. Tausend. Declarative bias in ILP. In L. DeRaedt, editor, *Advances in Inductive Logic Programming*, pages 82–103. IOS Press, Amsterdam, 1996.

[80] R. M. Neil. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report 9702, University of Toronto, CAN, 1997.

[81] J. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal*, 7:308–313, 1965.

[82] B. Nordhausen and P. Langley. A robust approach to numeric discovery. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 411–418, San Francisco, CA, 1990. Morgan Kaufmann.

[83] M. B. Passos Amato, C. S. Valente Barbas, D. Machato Medeiros, R. Borges Magaldi, G. De Paula Pinto Schettino, G. Lorenzi-Filho, R. A. Kairalla, D. Deheinzelin, C. Munoz, R. Oliveira, T. Y. Takagaki, and C. R. Ribeiro Carvalho. Effect of a protective-ventilation strategy on mortality in the acute respiratory distress syndrome. *The New England Journal of Medicine*, 338:347–354, 1998.

[84] P. Pelosi, M. Cereda, G. Foti, M. Giacomini, and A. Pesenti. Alterations of lung and chest wall mechanics in patients with acute lung injury: effects of positive end-expiratory pressure. *American Journal of Respiratory and Critical Care Medicine*, 152(2):531–537, 1995.

[85] P. Pelosi, M. Croci, I. Ravagnan, M. Cerisara, P. Vicardi, A. Lissoni, and L. Gattinoni. Respiratory system mechanics in sedated, paralyzed, morbidly obese patients. *Journal of Applied Physiology*, 82(3):811–818, 1997.

[86] A. Pesenti, P. Pelosi, N. Rossi, A. Virtuani, L. Brazzi, and A. Rossi. The effects of positive end-expiratory pressure on respiratory resistance in patients with the adult respiratory distress syndrome and in normal anesthetized subjects. *The American Review of Respiratory Disease*, 144(1):101–107, 1991.

[87] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1988.

[88] C. Putensen, M. Baum, W. Koller, and G. Putz. The PEEP wave: an automated technic for bedside determination of the volume/pressure ratio in the lungs of ventilated patients. *Der Anaesthesist*, 38(4):214–219, 1989. Article in German.

[89] J. R. Quinlan. Learning with continuous classes. In *Proceedings of the Australian Joint Conference on Artificial Intelligence*, pages 343–348, Singapore, 1992. World Scientific.

[90] J. Ramon. Predicting evolution of critically ill patients. In *Proceedings of the KDD 2006 workshop on Theory and Practice of Temporal Data Mining*, pages 1–3, 2006.

[91] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Boston, MA, 2006.

[92] J. L. Rogers and W. A. Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[93] F. Rohrer. Der Strömungswiderstand in den menschlichen Atemwegen und der Einfluss der unregelmässigen Verzweigungen des Bronchialsystems auf den Atmungsverlauf in verschiedenen Lungenbezirken. *Pflügers Archiv für die gesamte Physiologie des Menschen und der Tiere*, 162:225–299, 1915.

[94] A. Rossi, S. Gottfried, L. Zocchi, B. Higgs, S. Lennox, P. Calvery, P. Begin, A. Grassino, and J. Milic-Emili. Measurement of static compliance of the total respiratory system in patients with acute respiratory failure during mechanical ventilation: the effect of intrinsic PEEP. *The American Review of Respiratory Disease*, 131:672–677, 1985.

[95] A. Rossi, S. B. Gottfried, B. D. Higgs, L. Zocchi, A. Grassino, and J. Milic-Emili. Respiratory mechanics in mechanically ventilated patients with respiratory failure. *Journal of Applied Physiology*, 58(6):1849–1858, 1985.

[96] C. Schaffer. A proven domain-independent scientific function-finding algorithm. In *Proceedings of the 8th National Conference on Artificial Intelligence*, pages 828–833. AAAI Press/The MIT Press, 1990.

[97] C. Schaffer. Bivariate scientific function finding in a sampled, real-data testbed. *Machine Learning*, 12:167–183, 1991.

[98] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental datastress relaxation in rabbits' lungs. *Science*, 324(81):81–85, 2009.

[99] S. Schumann, M. Lichtwarck-Aschoff, C. Haberthür, C. A. Stahl, K. Möller, and J. Guttmann. Detection of partial endotracheal tube obstruction by forced pressure oscillations. *Respiratory Physiology & Neurobiology*, 155(3):227–233, 2007.

[100] B. Selman, H. J. Levesque, and D. Mitchell. A new method for solving hard satisfiability problems. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 440–446, Menlo Park, CA, 1992. AAAI Press.

[101] B. Selman, H. J. Levesque, and D. Mitchell. GSAT: A new method for solving hard satisfiability problems. Technical report, AT&T Bell Laboratories, Murray Hill, NJ, 1992.

[102] J. Semmlow. *Circuits, Signals, and Systems for Bioengineers.* Elsevier Academic Press, Burlington, MA, 2005.

[103] J. T. Sharp, F. N. Johnson, N. B. Goldberg, and P. V. Lith. Hysteresis and stress adaption in the human respiratory system. *Journal of Applied Physiology*, 23(4): 487–497, 1967.

[104] T. Similowski, P. Levy, C. Corbeil, M. Albala, R. Pariente, J. P. Derenne, J. H. T. Bates, B. Jonson, and J. Milic-Emili. Viscoelastic behavior of lung and chest wall in dogs determined by flow interruption. *Journal of Applied Physiology*, 67(6):2219–2229, 1989.

[105] C. A. Stahl, K. Möller, S. Schumann, R. Kuhlen, M. Sydow, C. Putensen, and J. Guttmann. Dynamic versus static respiratory mechanics in acute lung injury and acute respiratory distress syndrome. *Critical Care Medicine*, 34(8):2090–2098, 2006.

[106] O. Stenqvist, H. Odenstedt, and S. Lundin. Dynamic respiratory mechanics in acute lung injury/acute respiratory distress syndrome: research or clinical tool? *Current Opinion in Critical Care*, 14:87–93, 2008.

[107] M. Sydow, H. Burchardi, J. Zinserling, H. Ische, T. A. Crozier, and W. Weyland. Improved determination of static compliance by automated single volume steps in ventilated patients. *Intensive Care Medicine*, 17(2):108–114, 1991.

[108] R. Tanaka and M. S. Ludwig. Changes in viscoelastic properties of rat lung parenchymal strips with maturation. *Journal of Applied Physiology*, 87(6):2081–2089, 1999.

[109] The Acute Respiratory Distress Syndrome Network. Ventilation with lower tidal volumes as compared with traditional tidal volumes for acute lung injury and the acute respiratory distress syndrome. *The New England Journal of Medicine*, 342(18): 1301–1308, 2000.

[110] M. J. Tobin. Ventilator monitoring, and sharing the data with patients. *American Journal of Respiratory and Critical Care Medicine*, 163(4):810–811, 2001.

[111] L. Todorovski and S. Džeroski. Declarative bias in equation discovery. In *Proceedings of Fourteenth International Conference on Machine Learning*, pages 376–384, San Francisco, CA, 1997. Morgan Kaufmann.

[112] L. Todorovski, W. Bridewell, O. Shiran, and P. Langley. Inducing hierarchical process models in dynamic domains. In *Proceedings of the 20th National Conference on Artificial Intelligence - volume 2*, pages 892–897, Menlo Park, CA, 2005. AAAI Press.

[113] B. Üstün, W. J. Melssen, and L. M. C. Buydens. Facilitating the application of support vector regression by using a universal Pearson VII function based kernel. *Chemometrics and Intelligent Laboratory Systems*, 81(1):29–40, 2006.

[114] K. v. Neergaard and K. Wirz. Über die Methode zur Messung der Lungenelastizität am lebenden Menschen, insbesondere beim Emphysem. *Zeitschrift für klinische Medizin*, 105:35–50, 1927.

[115] K. v. Neergard and K. Wirz. Die Messung der Strömungswiderstände in den Atemwegen des Menschen insbesondere bei Asthma und Emphysem. *Zeitschrift für klinische Medizin*, 105:51–82, 1927.

[116] P. R. J. van der Laag and S.-H. Nienhuys-Cheng. Completeness and properness of refinement operators in inductive logic programming. *The Journal of Logic Programming*, 34(3):201–225, 1998.

[117] P. Vuilleumier. Über eine Methode zur Messung des intraalveolären Druckes und der Strömungswiderstände in den Atemwegen des Menschen. *Zeitschrift für klinische Medizin*, 143:698–717, 1944.

[118] J. M. Wang, D. J. Fleet, and A. Hertzmann. Multifactor Gaussian process models for style-content separation. In *Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, 2007*, pages 975–982, 2007.

[119] C. K. I. Williams. Prediction with Gaussian processes: from linear regression to linear prediction and beyond. Technical Report NCRG/97/012, Aston University Birmingham, UK, 1997.

[120] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, CA, 2 edition, 2005.

[121] R. Zembowicz and J. M. Żytkow. Automated discovery of empirical equations from data. In *Methododologies for Intelligent Systems*, volume 542 of *Lecture Notes in Computer Science*, pages 429–440. Springer, Berlin/Heidelberg, 1991.

[122] Z. Zhao, K. Möller, D. Steinmann, I. Frerichs, and J. Guttmann. Evaluation of an electrical impedance tomography-based global inhomogeneity index for pulmonary ventilation distribution. *Intensive Care Medicine*, 35:1900–1906, 2009.

[123] J. Żytkow. Combining many searches in the FAHRENHEIT discovery system. In *Proceedings of the Fourth Internationa Workshop on Machine Learning*, pages 281–287, Irvine, CA, 1987.

List of Symbols

B set of beam elements $\{b_1, b_2, \ldots, b_{w_b}\}$.

$(B, <)$ ordered set of beam elements $B = \{b_1, b_2, \ldots, b_{w_b}\}$: $b_i < b_j :\Leftrightarrow F_{\mathrm{h}}(b_i) > F_{\mathrm{h}}(b_j))$ (Definition 8).

C compliance of the lung.

C_{max} maximum compliance within the complete inspiratory pressure-volume range.

C_{ve} viscoelastic compliance.

$\mathcal{D}$ dataset consisting of N observations of $\mathrm{X}_N, \mathrm{y}_N$ with a fixed dimension D.

D dimension of fixed size.

ΔP_{ETT} pressure drop at endotracheal tube.

E elastance of the lung.

E_{ve} viscoelastic elastance.

E expectation value.

$\mathcal{F}$ fitting method.

F_{h} heuristic function.

$f(\mathrm{x}; \mathrm{w})$ parameterized function.

G context free grammar.

h height of a production rule or parse tree, respectively. Cf. Definition 1 on page 45.

h_{max} maximum height of a production rule or parse tree, respectively.

$k(\cdot, \cdot)$ kernel function.

$\vec{\mu}$ vector of expectation values $(\mu_1, \ldots, \mu_n) \in \mathbb{R}$ of Gaussian distributed random variables.

$\mathcal{N}(\vec{\mu}, \Sigma)$ multivariate Gaussian distribution with mean values $\vec{\mu}$ and covariance matrix Σ.

N set of nonterminal symbols.

$\phi(x)$ (nonlinear) basis function as used in (nonlinear) regression.

$\Phi(\mathrm{x})$ vector of basis functions $[\phi_1(\mathrm{x}), \ldots, \phi_H(\mathrm{x})]$.

$\mathbf{\Phi}_{NH}$ matrix of values of basis functions $\{\phi_h(x)\}_{h=1}^{H}$ at the input points $\{\mathrm{x}_n\}_{n=1}^{N}$.

P pressure.

P_{plat} plateau pressure.

P set of productions $\{p_1, \ldots, p_l\}$ with $p_i := P_A = A \rightarrow a_1 | \ldots | a_m$. Cf. P_A.

P_A production; set of production rules $\{p_{A,1}, p_{A,2}, \ldots, p_{A,m}\} = \{A \rightarrow a_1, A \rightarrow a_2, \ldots, A \rightarrow a_m\}$ all having the same nonterminal symbol A on the left side. Cf. $p_{A,i}$.

$p_{A,i}$ production rule; $p_{A,i} := A \rightarrow a_i$ with $A \in N$ and $a_i \in (N \cup T)^*$.

R Newtonian airway resistance.

$R(Q)$ set of all possible refinements of a set of parse trees in $Q = \{\mathcal{T}_0, \ldots, \mathcal{T}_n\}$ as $\bigcup r(\mathcal{T}_\mathrm{i}) : \mathcal{T}_\mathrm{i} \in Q$. Cf. $r(\mathcal{T})$.

$r(\mathcal{T})$ set of all possible refinements of parse tree $\mathcal{T}$.

R_{ve} viscoelastic resistance.

Σ_x covariance matrix of Gaussian distribution for set x of random variables.

S start symbol $\in N$.

$\mathcal{S}$ stopping criterion (none, timelimit [sec]) of ED algorithm.

$\mathcal{T}$ parse tree (with start symbol S as root node).

$\mathcal{T}_\mathrm{A}$ parse tree with nonterminal A as root node.

$\tau_{\mathbf{ve}}$ viscoelastic time constant.

T set of terminal symbols.

$\mathcal{T}_{history}$ (limited) set of derived parse trees used to avoid redundant parse tree evaluations in ED algorithm.

V volume.

$\dot{V}$ airflow.

$\ddot{V}$ volume acceleration.

w parameter vector $[\omega_1, \ldots, \omega_H]$.

w_b beam width.

x vector of parameters $[x_1, \ldots, x_D]$ with a fixed dimension D.

X_N set of N input vectors $\{\mathrm{x}_n\}_{n=1}^N = \{[x_1, \ldots, x_D]_n\}_{n=1}^N$ of parameters x with a fixed dimension D.

y_N vector of N target values $\{y_n\}_{n=1}^N$ corresponding to input vector X_N.

Acronyms

ANOVA analysis of variance. i, 78, 79, 81, 83

ARDS cf. *acute respiratory distress syndrome*. i, 5, 6, 16, 17, 24–26, 30, 31, 33–37, 92, 97, 98, 111, 114, 120, 133–135

ASA American Society of Anesthesiologists. i, 19

CNF conjunctive normal form. i, 68

dB decibel. i, 33

DP Davis-Putnam algorithm. i, 69

ED cf. *equation discovery*. i, 1, 4, 39–42, 50, 70, 79, 85, 86, 88, 89, 114, 133–135

EF cf. *equation finder*. i, 41, 133

EIP cf. *end-inspiratory pause*. i, 8, 18, 133

EOM cf. *equation of motion*. i, iv, 9, 14, 75, 77–82, 85, 134

FiO_2 fraction of inspired oxygen. i, 18

GP cf. *Gaussian process*. i, 1, 4, 91, 94–97, 103–106, 110, 111, 114, 134

GSAT cf. *greedy satisfiability problem solving algorithm*. i, 68–70, 74, 87, 89, 134

HIPM cf. *hierarchical inductive process modeling*. i, 88, 134

IPM cf. *inductive process modeling*. i, 88, 134

KDD cf. *knowledge discovery in databases*. i, 1, 4, 21, 41, 85, 113, 114, 135

ML cf. *machine learning*. i, 1, 4, 21, 41, 45, 85, 88, 113, 114, 134, 135

PEEP cf. *positive end-expiratory pressure.* i, 6–9, 12, 15, 16, 19–21, 23, 34–36, 75, 76, 91, 92, 97, 105, 115, 135

RBF cf. *radial basis function.* i, 95, 136

RMSE root mean squared error. i, 36, 49, 52, 54, 60, 78, 79, 82–84, 120

SAT propositional satisfiability. i, 68

SCASS cf. *static compliance by automated single steps.* i, 13, 136

VALI cf. *ventilator associated lung injury.* i, 2, 5, 21, 91, 106, 137

ZEEP cf. *zero end-expiratory pressure.* i, 10, 12, 17, 19, 20, 28, 34, 36, 137

Glossary and Index

ABACUS ED system introduced by Falkenhainer and Michalski in 1990 [31]. i, 40

acute respiratory distress syndrome Complex of symptoms including acute dyspnea, hypoxemia and reduced lung compliance. i, 26, 131

BACON ED system introduced by Pat Langley in 1977 [65]. i, 40, 41

barotrauma Structural damage of the lung due to excessive application of pressure. Cf. *volutrauma*. i, 21, 137

compliance Volume distensibility of the lung as reciprocal of the *elastance*. Determined by the change of respiratory volume V divided by the change of applied respiratory pressure P, i.e. $C = \Delta V/\Delta P$. i, ix, xi, 5–10, 12–16, 19, 21, 22, 24, 27–29, 91, 92, 97–99, 101–111, 114, 115, 133

compliance, viscoelastic *Compliance* of the viscoelastic component of *spring-and-dashpot model* and its electrical analog, respectively. Reciprocal of the *viscoelastic elastance*. i, 12, 25, 28, 30–32, 36, 37, 92, 98, 133

derecruitment Loss of inflated lung areas during deflation. Cf. *recruitment*. i, 136

equation finder ED system introduced by Zembowicz and Żytkow in 1991 [121]. i, 41

elastance Volume distensibility of the lung as pressure change divided by volume change; reciprocal of the *compliance*. i, 9, 10, 36, 133

elastance, viscoelastic Reciprocal of the *viscoelastic compliance*. i, 34, 133

end-inspiratory pause Airflow interruption upon the end of inspiration of at least one second. Used to obtain static conditions to determine the lung compliance. i, 131

equation discovery Research area in *knowledge discovery in databases*, developing methods for the automatized discovery of quantitative laws in measured data. i, 131

equation of motion Well known linear equation to describe the pressure difference between the atmosphere and the airways by addition of three pressure components. i, 131

E* ED system introduced by Schaffer in 1990 [96]. i, 41, 42, 86

extra-pulmonary ARDS Type of ARDS which is characterized by an indirect injury of vascular endothelium by inflammatory mediators such as abdominal peritonitis or occlusion, acute pancreatitis or abdominal trauma. Cf. *pulmonary ARDS*. i, 5, 17, 18, 135

FAHRENHEIT ED system introduced by Żytkow in 1987 [123]. i, 41

FUSE ED system "forming unified scientific explanations" introduced by Bridewell *et al.* in 2005 [15]. i, 88

Gaussian process Collection of random variables, any finite number of which having a joint Gaussian distribution. i, 91, 94, 131

GOLDHORN ED system introduced by Križman *et al.* in 1995 [59]. i, 41

greedy satisfiability problem solving algorithm (GSAT) Randomized local search procedure for solving propositional satisfiability problems (SAT) of propositional formulas in conjunctive normal form. i, 131

hierarchical inductive process modeling (HIPM) ED approach introduced by Todorovski *et al.* in 2005 [112]. i, 131

hysteresis effects Characterized by differing pressure-volume curves during in- and expiration due to different elastic retraction forces. i, 13

IDS ED system introduced by Nordhausen and Langley in 1990 [82]. i, 40, 41

inductive logic programming Technique from the field of ML where background knowledge expressed in first-order logic is used to find inductive hypotheses from a set of observations [77, 79]. i, 41, 42, 58

inductive process modeling (IPM) ED approach introduced by Langley *et al.* in 2002 [64]. i, 131

inhomogeneity Unequal ventilation of functional or structural differing lung areas. i, 24, 37

inspiratory capacity Volume of the human lung consisting of the tidal volume and the (additional) inspiratory reserve volume. i, 6, 7, 12, 15, 25, 37, 114

knowledge discovery in databases (KDD) Research area in computer science focusing on knowledge extraction in terms of relations within (vast) datasets. The results should allow for inferencing.. i, xi, 24, 115, 131, 133

LAGRAMGE ED system introduced by Todorovski and Džeroski in 1997 [111]. i, 42, 44, 45, 49, 63, 70, 86, 88, 89

LAGRANGE ED system introduced by Džeroski and Todorovski in 1994 [27]. i, 41

lumped parameter model Electrical analog of the *spring-and-dashpot model*. i, 25, 34, 136

lung protective ventilation Ventilation strategy which is optimized in terms of the reduction of applied mechanical *stress* to the pulmonary system. i, xi, 1–6, 21, 24, 35, 91, 92, 97, 106, 110, 113–115

machine learning (ML) Research area in computer science focusing on learning from data in order to facilitate prediction and decision making.. i, xi, 24, 115, 131

Pearson product-moment correlation coefficient Measure of linear correlation introduced by Karl Pearson [92]. Provides a value between +1 and -1 (inclusive) as measure of the dependence of two variables X and Y. i, 104, 107, 108

Pearson VII function-based universal kernel Kernel function [113]. i, 103

PEEP wave maneuver Respiratory maneuver applying a stepwise in- and decrease of PEEP while performing regular breathing cycles. i, 14–16, 21–23, 75, 76, 114, 136

"Pendelluft" phenomenon Equalization of inhomogeneously distributed air volumina within the lung. It is assumed that besides viscoelastic effects, the Pendelluft phenomenon contributes to stress relaxation processes of the lung tissue. Cf. *viscoelastic effects*. i, 8, 10, 137

plateau pressure Pressure level reached after stress relaxation succeeding a flow interruption during in- or expiration. i, 11, 12, 14, 15, 19–21, 25–35, 37, 76, 92, 98–103, 107–110

positive end-expiratory pressure Pressure level after complete expiration of the tidal volume. i, 6, 19, 132

production Cf. P_A. i, 43, 44, 46, 71

production rule Cf. $p_{A,i}$. i, 44–52, 54, 56–62, 64, 68, 70–75, 78–83, 85–87

pulmonary ARDS Type of ARDS where the alveolar epithelium is directly injured. Cf. *extra-pulmonary ARDS*. i, 5, 17, 18, 134

radial basis function Real-valued function whose values merely depend on the distance of the input parameters from a center point. i, 132

recruitment Gain of additional lung areas during inflation. Cf. *derecruitment*. i, 133

resistance, Newtonian airway Flow resistance in the airways. i, 5, 10, 13, 27–29, 97

resistance, viscoelastic Resistance of the viscoelastic component of *spring-and-dashpot model* and its electrical analog, respectively. i, 12, 25, 28, 30–34, 37, 92, 98

respiratory maneuver Ventilation maneuver accomplished by a well-defined protocol to measure respiratory data such as pressure, volume and flow. Cf. *PEEP wave maneuver*, *super syringe maneuver*. i, 2, 9, 17, 18, 21

spring-and-dashpot model Lung model which is supposed to imply the representation of viscoelastic behavior of the lung tissue. Cf. *lumped parameter model*. i, 3, 8, 10, 11, 13, 27, 28, 34, 36, 92, 99, 100, 104, 114, 120, 133, 135–137

static compliance by automated single steps (SCASS) Measuring method to obtain static pressure-volume relation over the complete range of inspiratory capacity. i, 13, 132

strain Deformation of tissue associated to stress. Linked to stress by $stress = k \times strain$. Cf. *stress*. i, 6, 25, 33, 34, 37, 136

stress Defined as the internal distribution of counterforce per unit of area reacting to an external load. Linked to strain by $stress = k \times strain$. Cf. *strain*. i, xi, 1–3, 6, 10, 12, 25, 28, 31, 33–37, 113, 115, 135, 136

stress relaxation Process observed after in- or deflation of the lung with a constant airflow followed by a flow interruption. After the interruption, a first quick pressure drop (or rise, respectively) is followed again by slow pressure drop (rise), expressing stress relaxation processes of the lung. "This is similar to stress relaxation seen when an imperfectly elastic structure is subjected to a constant strain or distortion, or when an elastic structure is stressed beyond its yield point." (R. Hughes *et al.* [51], 1959, p. 85). i, 1, 3, 8–12, 92, 114

super syringe maneuver Respiratory maneuver applying volume steps of predefined size. After each such step a flow interruption of several seconds is performed. i, 14, 15, 17–20, 22, 23, 25, 26, 28–31, 33, 35–37, 92, 98–100, 102, 104, 114, 120, 136

tidal volume Regular breathing volume of approximately 500 mL in the healthy human lung. i, 5, 7, 8, 10, 12–15, 18, 19, 35, 37, 38

time constant, viscoelastic Time constant for stress relaxation process. In the *spring-and-dashpot model* holds $\tau_{ve} = C_{ve} \times R_{ve}$. i, 12, 25, 28, 31, 32, 34, 37

ventilator associated lung injury (VALI) Lung injury which is supposed to be caused by mechanical ventilation, e.g. due to inadequate ventilator settings. i, 132

viscoelastic effects Effects observable in context of flow interruptions during in- or deflation which are supposed to be caused by viscoelastic properties of the lung tissue. Cf. *"Pendelluft" phenomenon*. i, 3, 8, 24, 27, 36, 97, 114, 135

Ramsay sedation score Scale from 1 – 6 to classify the degree of sedation. i, 17

volutrauma Structural damage of the lung due to excessive application of volume. Cf. *barotrauma*. i, 21, 133

zero end-expiratory pressure Pressure level after complete expiration of the tidal volume with a pressure level of 0 mbar. i, 132

Curriculum Vitae

Curriculum vitae was removed from online version.