Inferring Proteolytic Processes from Mass Spectrometry Time Series Data

vorgelegt von

Stephan Aiche

Dissertation zur Erlangung des Grades

Doktor der Naturwissenschaften (Dr. rer. nat.)

am

Fachbereich Mathematik und Informatik

der

Freien Universität Berlin

May 2013

Datum der Disputation: 30.09.2013

Betreuer:

Professor Dr. Christof Schütte, *Freie Universität Berlin*

Gutachter:

Professor Dr. Christof Schütte, *Freie Universität Berlin*
Professor Dr. Oliver Kohlbacher, *Eberhard Karls Universität Tübingen*

Abstract

Proteolysis, the catalyzed hydrolysis of peptide bonds, is an important post-translational modification, having a significant influence on the life cycle of protein and peptides. It is involved in numerous biological processes, like apoptosis, cell cycle progression, or blood coagulation. More then 500 genes were annotated as proteases, the enzymes catalyzing proteolytic cleavage of proteins and peptides, but many of them are still insufficiently characterized. Hence a profound understanding of proteolytic processes is essential for a detailed analysis of many biological processes. Furthermore proteolysis is associated with multiple complex diseases like cancer and Alzheimer's disease and is known to be involved in the infection with the HI-virus. Beyond its implication in biological processes, proteolysis can also be utilized for diagnostic and treatment purposes. Proteases, the enzymes catalyzing proteolytic cleavage, are established drug targets and their potential as biomarkers has been postulated in 2006 by Villanueva et al.

In this thesis we present a novel approach to the characterization of proteolytic processes using mass spectrometry data. We utilize the qualitative and quantitative information of the mass spectra to construct a model, the degradation graph, containing all involved peptides as well as the individual proteolytic reactions that connect them. We further propose a transformation of the degradation graph into a mathematical model that can be utilized in combination with the mass spectrometry data to estimate the rate constants of the individual reactions inside the degradation graph. Additionally we developed a score that can be used to rate different degradation graphs with respect to their ability to explain the observed mass spectrometry data. We use this score to iteratively improve the structure of an initially constructed degradation graph so as to account for errors during the construction of the degradation graph.

While more and more mass spectrometry data is produced and is publicly available, there is a lack of well annotated, so called gold standard or ground truth datasets. Those datasets are required for a thorough benchmarking of novel algorithms and newly developed software. This problem is increasing as the experimental setups and scientific questions in computational mass spectrometry get more and more complex.

We therefore present MSSimulator, a comprehensive simulator for mass spectrometry data. Although using simulated data does not remove the need for testing on real datasets, it eases algorithm benchmarking and development, due to the availability of ground truth data which enables us to compare and validate the results more effectively. MSSimulator is the currently most comprehensive simulator for mass spectrometry data. It provides different types of experimental setups (e.g. labeled and label-free setups), simulation of tandem mass spectra, as well as numerous options to reflect different experimental conditions like noise, chromatographic conditions, or instrument type. It produces different levels of ground truth starting with the simulated raw data, to feature and peak locations, and relational information (e.g. grouping of charge states or labeled pairs). With the data generated by MSSimulator we benchmarked different existing applications for the analysis of mass spectrometry data as well as our own approach for the analysis of proteolytic processes.

Zusammenfassung

Proteolyse, die Hydrolyse von Peptidbindungen, ist eine wichtige post-translationale Modifikation, die maßgeblich den Lebenszyklus von Proteinen und Peptiden beeinflusst. Sie ist in zahlreichen biologischen Prozessen, wie z.B. der Regulation des Zellzyklus, der Apoptose oder der Blutgerinnung regulatorisch aktiv. Mehr als 500 Gene im menschlichen Genom wurden als Proteasen, Enzyme die den proteolytischen Verdau von Proteinen und Peptiden katalysieren, annotiert. Trotzdem sind viele bis heute nur unzureichend untersucht. Ein besseres Verständnis proteolytischer Prozesse, der komplexen Kaskaden von interagierenden Proteasen, ist folglich eine grundlegende Voraussetzung für eine detaillierte Analyse biologischer Prozesse. Bei der Entwicklung von komplexen Krankheiten wie Krebs und Alzheimer und der Infektion mit dem HI-Virus spielt die Proteolyse ebenfalls eine bedeutende Rolle und beeinflusst folglich sowohl deren Diagnose als auch die Behandlung. Proteasen sind etablierte Zielproteine für Arzneimittel. Ihr Potential als Biomarker wurde 2006 von Villanueva et al. beschrieben.

In dieser Arbeit beschreiben wir einen neuen Ansatz zur Charakterisierung von proteolytischen Prozessen. Wir präsentieren eine Methode, die unter Ausnutzung der qualitativen und quantitativen Informationen in Massenspetrometriedaten, ein Modell - den Degradation Graph - konstruiert. Dieses Modell enthält sowohl alle involvierten Peptide als auch die proteolytischen Reaktionen, die diese mit einander verbinden. Zusätzlich beschreiben wir eine Transformation des degradation graphs in ein mathematisches Modell, welches zusammen mit den Massenspektrometriedaten dazu verwendet werden kann die Reaktionskonstanten der einzelnen proteolytischen Reaktionen zu schätzen. Darüber hinaus haben wir ein Bewertungsschema für den degradation graph entwickelt. Es dient dazu, verschiedene degradation graphs miteinander, im Bezug auf ihrer Fähigkeit die beobachteten Daten zu erklären, zu vergleichen. Dieses Bewertungsschema haben wir dazu verwendet die anfänglich konstruierten degradation graphs schrittweise zu verbessern um mögliche Fehler bei der Konstruktion auszugleichen.

In den letzten Jahren ist die Menge an öffentlich verfügbaren Massenspetrometriedaten stetig angestiegen. Dennoch herrscht weiterhin ein Mangel an gut annotierten Datensätzen, so genannter Goldstandards. Die Goldstandards sind notwendig um neu entwickelte Programme und Algorithmen intensiv testen und mit bestehenden Ansätzen vergleichen zu können. Die zunehmende Komplexität der wissenschaftlichen Fragestellungen und experimentellen Techniken vergrößert den Bedarf an Goldstandards zusätzlich.

Zur Lösung des Problems haben wir MSSimulator entwickelt, einen umfangreichen Simulator für Massenspetrometriedaten. Obwohl die Verwendung von simulierten Daten die Notwendigkeit der Validierung auf realen Daten nicht obsolet macht, so erleichtert es doch die Entwicklung und das Testen von neuen Methoden. Ein Vergleich mit bereits existierenden Methodiken wird ebenfalls stark vereinfacht. MSSimulator ermöglicht die Simulation von unterschiedlichen experimentellen Ansätzen sowie die Simulation von Tandem-Massenspektrometriedaten. Es bietet vielfältige Einstellmöglichkeiten um die generierten Daten unter anderem im Hinblick auf Rauschen, chromatographischen Bedingungen oder Auflösung, dem eigenen experimentellen Aufbau anzupassen. MSSimulator erzeugt mehrere Ebenen des Goldstandards, angefangen bei den simulierten Rohdaten über die exakten Peptide- und Peakpositionen bis hin zu Gruppierungsinformationen, z.B. unterschiedlicher Ladungsvarianten. Die simulierten Daten nutzen wir in dieser Arbeit zum Vergleich verschiedener existierender Applikationen, zur Analyse von Massenspektrometriedaten und zur Entwicklung und Validierung unseres Ansatzes zur Analyse von proteolytischen Prozessen.

Acknowledgments

First and foremost I want to thank Prof. Christof Schütte and Prof. Knut Reinert who made this thesis project possible. Their constant support, patience, advice, and motivation helped me to carry out this project over the past years. I also thank Tim Conrad who supervised my work during the last years and was a constant source of comments, discussions, and valuable advices. My thanks also go to Oliver Kohlbacher for reviewing this thesis and for the numerous, interesting discussions we shared. Additionally I want to thank Prof. Hartmut Schlüter, Maria Trusch, Diana Hildebrand, and Susanna Röblitz who provided important data, fruitful discussions, and invaluable comments.

Further I want to thank all the current and previous members of the biocomputing group and the Algorithmic Bioinformatics group for the great working environment they provided. Especially I would like to thank my office mates, Chris, Sandro, and Dave who made the numerous working hours a very pleasant time. I'm also very grateful to Sabine and Axel for all the coffees we drank together. Many thanks are also given to the OpenMS development team for giving me the opportunity to collaborate with such an inspiring group of people on an incredible peace of software.

Last but not least I want to thank my family, my friends, and especially Sabrina for their constant and unconditional support over the last years.

Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Proteases – Essential part of nature's toolkit

Protease catalyzed cleavage of peptide bonds is one of the most important irreversible, post-translational modifications of proteins and peptides. Proteases (also known as peptidases or proteolytic enzymes) operate in two distinct modes, either they remove single amino acids from the N- or C-terminus of the protein (exoproteases) or they hydrolyze a peptide bond in the protein (endoproteases). Currently MEROPS [1], a comprehensive database for peptidases, lists 816 putative, human peptidases and 1,670 peptidase inhibitors (as of December 17, 2012). Beside their prominent involvement in food catabolism, proteases are involved in a variety of biological processes where they serve as regulatory enzymes by activating or deactivating the targeted peptides and proteins. A vast number of complex biological processes and signal cascades involve proteolysis, e.g., blood coagulation [2], cell cycle progression [3], and apoptosis [4]. There is also an active interaction between proteases and other regulatory enzymes like kinases [5].

Beyond their importance under normal biological conditions, peptidases play an important role in complex diseases, such as cancer [6], inflammatory bowel diseases [7], Alzheimer's disease [8, 9], and HIV or HCV infection [10]. It has been reported that there exist correlations between protease activity and tumor invasion, metastasis establishment and early stage development of cancer [11, 12, 13, 14]. With their involvement in many disease related processes they also became an interesting target for drug development [15, 10]. For instance inhibitors of the HIV-I protease are prominent targets for HIV treatment [16, 17]. In Alzheimer's disease the β-secretase is a promising therapeutic target [18].

Proteolyitc enzymes are not only involved in different complex biological diseases, but can also help in detecting such diseases. The low molecular weight peptide fragments measurable in the blood serum that were generated by proteolytic activity were recognized already in 2006 by Liotta and coworkers as potential biomarkers [19, 20]. This was supported by the results of Villanueva et al. [21] in 2006 where the authors suggested that exoprotease activity and the resulting serum peptide patterns could be used not only to discriminate healthy

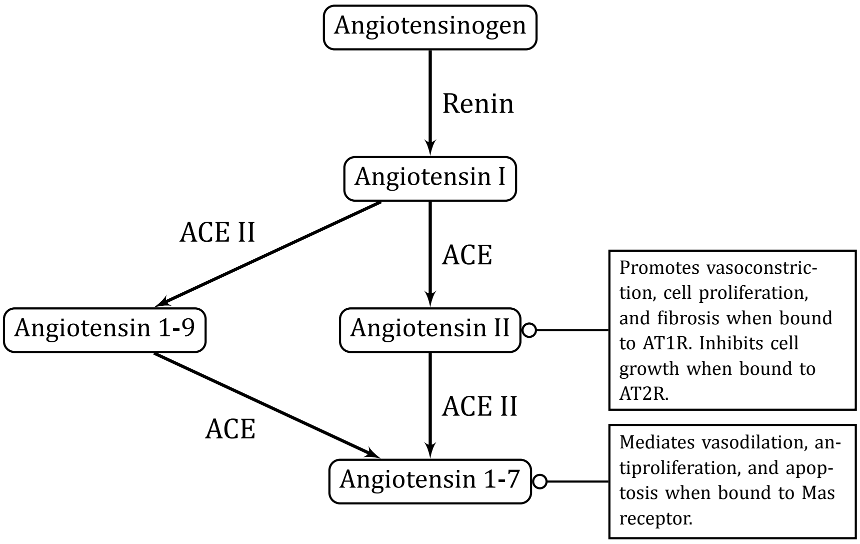

Angiotensinogen
Renin
Angiotensin I
ACE II
ACE
Angiotensin 1-9
Angiotensin II
Promotes vasoconstriction, cell proliferation, and fibrosis when bound to AT1R. Inhibits cell growth when bound to AT2R.
ACE II
ACE
Angiotensin 1-7
Mediates vasodilation, antiproliferation, and apoptosis when bound to Mas receptor.

Figure 1.1: The renin-angiotensin system. The precursor protein Angiotensinogen is targeted by the protease renin which produces Angiotensin I. Angiotensin I itself is degraded by two different proteases, Angiotensin Converting Enzyme and Angiotensin Converting Enzyme II, producing different fragments with opposing effects on the cardiovascular system. **ACE** Angiotensin Converting Enzyme; **ACE II** Angiotensin Converting Enzyme II; **AT1R** Angiotensin II type 1 receptor; **AT2R** Angiotensin II type 2 receptor.

from cancer patients, but even to differentiate the cancer type. Recent studies like the one presented by Peccerella et al. [22] also support this hypothesis.

An example of a regulatory system that is tightly controlled by proteolytic enzymes is the renin-angiotensin system that is shown in Figure 1.1. The precursor protein angiotensinogen is targeted by the protease renin which produces angiotensin I. Angiotensin I itself is degraded by two different proteases, angiotensin converting enzyme (ACE) and angiotensin converting enzyme II (ACE II), producing different fragments with opposing effects on the cardiovascular system [23]. Dysregulation of this system of proteases was associated with cardiovascular diseases. This led to the development of renin inhibiting drugs such as aliskiren [24].

In this rather small example we can already see that for a complete understanding of the renin-angiotensin system we need to know which protease targets which substrate, at which specific position the protein or peptide gets cleaved (the so called cleavage site), and at which rate the different substrates are produced due to their different regulatory effects. The notion of cleavage site and the nomenclature to describe the surrounding amino acids was initially proposed by Schechter and Berger [25] and is shown in Figure 1.2. The two N- and C-terminal amino acids that are directly located at the hydrolyzed peptide bond are named $P1$ and $P1'$

Pn - - P4 — P3 — P2 — P1 ╫ P1’ – P2’ – P3’ – P4’ - Pm’

Figure 1.2: Schechter Berger notation of the protease cleavage site. The peptide bond between the amino acids $P1$ and $P1'$ is cleaved.

respectively. With increasing distance to the hydrolyzed bond the numbers rise (e.g., $P3$ for the third amino acid on the N-terminal side of the hydrolyzed bond). In most cases four amino acids on the N-terminal and three amino acids on the C-terminal side are used to characterize a cleavage site.

Mass Spectrometry as quantitative technique to analyze proteolytic processes

As we can see, the complete characterization of proteolytic reactions and their interactions requires both qualitative information in order to identify the substrate, the products, and the exact cleavage site of the proteolytic enzyme, as well as quantitative information to describe the dynamic evolution of the process. Mass spectrometry (MS) can provide both types of information and therefore is a suitable measurement technique for the study of proteolytic reactions.

Mass spectrometry of biomolecules like peptides and proteins was made possible through the introduction of soft ionization techniques in the late 1980s allowing the analysis of intact, non-fragmented biomolecules in a mass spectrometer. The two most widely used techniques are Matrix-assisted laser desorption/ionization (MALDI) [26, 27] and Electrospray ionization (ESI) [28]. Since their introduction the role of mass spectrometry in the analysis of peptides and proteins increased drastically [29]. Combined with other techniques like gel electrophoresis or chromatographic separation it finally enabled the study of the complete ensemble of proteins in a cell or sample, the proteome [30]. With this, mass spectrometry became also a suitable tool for the analysis of complex biological processes like proteolysis. Different techniques were developed that focus on different aspects of the proteolytic reactions, e.g., identification of the cleavage site using ”Proteomic Identification of protease Cleavage Sites” (PICS) [31]. Elaborate overviews of available mass spectrometry based techniques were given by Schlüter, Hildebrand, et al. [32], Impens et al. [33], and van den Berg and Tholey [34].

In this thesis we will focus only on a small subset of these techniques, where peptide probes are incubated over time with a mixture of proteolytic enzymes. A prominent example is the mass-spectrometry-assisted enzyme-screening (MES) system proposed by Schlüter, Jankowski, et al. [35]. Here complex protein mixtures with unknown proteolytic activity are immobilized and incubated with possible target peptides. By sampling from the incubated peptides at different time points and generating mass spectra of these samples, one can gain insight not only in the generated peptide fragments but, due to the quantitative information obtainable from the mass spectra, also in the kinetics of the process.

A new approach to model and analyze proteolytic reactions

Only few approaches were made to analyze the data resulting from those incubation experiments in a systematic manner. For instance Yi et al. [36] analyzed the sequential degradation of fibrinopeptide A and used time series data to estimate the associated kinetic parameters of their proposed reaction model. However they only focused on a single peptide (fibrinopeptide A) and the associated reactions and did not extend their approach further. So far no general approach is known to the authors that is able to analyze such data sets.

In this thesis we will present a systematic approach to analyze such experiments. We will introduce a data structure to describe and visualize the complete degradation process as well as the involved proteolytic enzymes, the *degradation graph*. It is comparable to the cleavage graph as it was presented by Kluge, Gambin, and Niemiro [37], but also includes endoproteolytic reactions. A similar extension to the cleavage graph was proposed by Dittwald et al. [38]. We will further describe how to translate the degradation graph into a mathematical model that describes the reaction rates of the individual proteolytic reactions inside the degradation graph. Based on this mathematical formulation and the quantitative information extracted from the mass spectra we formulate an optimization problem to estimate the reaction rates for the individual proteolytic reactions. Finally, to compensate for the noise in mass spectrometry data and the uncertainty in the identification of peptides that are part of the analyzed proteolytic reaction, we present an approach to optimize the initially constructed degradation graph with a focus on its ability to explain the observed data. As measure for the optimization we use a novel score that captures the ability of the model to reconstruct the dynamical behavior as well as its ability to explain the signals observed in the mass spectra.

Simulation as a novel ground truth

A complex measurement technique, such as mass spectrometry, and the constantly increasing amount of data generated by modern mass spectrometers require novel algorithms and software implementing those, since manual analysis becomes infeasible [39, 40]. The development of such algorithms and the corresponding software implementation requires thorough benchmarking and comparison to alternative or existing solutions in order to prove its competitiveness. For such a comparison benchmark data is required of which the desired outcome is known, e.g., the contained peptides and their quantities. In most cases such data sets are produced by manual annotation. However, manual annotation greatly relies on the expertise of the person annotating the data set and is prone to errors due to the complexity of the data sets. This task gets even more involved if complex processes, such as proteolytic process, are analyzed. An alternative is the simulation of benchmark data. Especially in the field of mass spectrometry based proteomics this approach has long been ignored. Only few attempts were made and those were in most cases tailored exactly to the problem that was studied.

In this thesis we present MSSimulator [41], a comprehensive simulator for mass spectrometry data. To ensure that MSSimulator is applicable to a wide range of problems in computational mass spectrometry, we carefully designed MSSimulator to be as flexible as possible in the data that can be generated. It provides multiple levels of ground truth, such as exact peak locations, feature positions, and associations between signals (e.g., between multiple charge states). It is the first approach to provide the ability to simulate labeled experiments and fragment spectra. MSSimulator is highly customizable, allowing the simulation of various experimental techniques and conditions. We will show how MSSimulator can be used to benchmark existing and newly developed software solutions. Additionally we will use MSSimulator to benchmark our approach for the analysis of proteolytic processes.

1.2 Guide to this thesis

This thesis focuses on the analysis of proteolytic processes using computational proteomics approaches. It is structured as follows.

Chapter 2 gives an introduction into the field of mass spectrometry based proteomics and some of the associated computational problems. This will serve as foundation for the following chapters as they will partly rely on a fundamental understanding of mass spectrometry based proteomics.

Chapter 3 introduces MSSimulator, a versatile simulation software for mass spectrometry data. The chapter will describe the structure of MSSimulator as well as the reasoning behind the individual simulation steps. Chapter 3 further presents in an exemplary manner possible applications of MSSimulator to test and benchmark existing applications for feature detection and quantification. Parts of this chapter have been published in Bielow et al. [41].

In Chapter 4 we will introduce our approach to model and analyze proteolytic processes based on mass spectrometry data. We will also introduce the notion of degradation graphs and a method to construct them based on mass spectrometry time series data. The chapter will further describe how the degradation graph can be used to construct a mathematical model that describes the dynamical evolution of the proteolytic process. We conclude Chapter 4 with an approach to optimize the structure of the degradation graph in case it was constructed from noisy data.

In Chapter 5 we will use MSSimulator to validate the degradation graph approach from Chapter 4 on simulated data. We show the influence of noise on the estimated reaction rates as well as the ability of our approach to optimize the structure of an initially constructed degradation graph. Furthermore we show the applicability of the approach to real data by testing it on an incubation time series of beta-2-microglobulin with immobilized urine proteins. The Chapters 4 and 5 have been published in Aiche et al. [42].

In Chapter 6 we will focus on some of the computational challenges associated with the methods described earlier. The approaches we present throughout this thesis are, if carried out on large data sets, computationally very demanding. We therefore integrated them into the existing grid platform *proteomics.net* [43]. We will describe the proteomics.net platform, its concepts, and the extensions implemented by the author.

Chapter 7 concludes this thesis by summarizing the contributions described in the previous chapters. We will also give some ideas for future extensions of the presented approaches.

Chapter

2 Mass Spectrometry based Proteomics – An Introduction

In the last decades mass spectrometry became an essential part of proteomic research [29, 44, 45]. The instruments as well as the experimental techniques improve every year leading to an ever increasing output of mass spectrometry data [39]. Currently hundreds to thousands of peptide signals can be identified and quantified in a single LC-MS experiment. With the rapidly growing amount of data and increasingly complex experimental questions also the computational challenges in handling and analyzing the data are getting more and more important.

In the following chapters we will give an overview of this rapidly evolving field. Therefore we will use this chapter to introduce the related concepts, ideas, and terminology. We will start with an introduction of a very generic LC-MS workflow which is depicted in Figure 2.1. Afterwards we will introduce the terminology of this field as it is used in this thesis. We will conclude this introduction into mass spectrometry based proteomics by describing two of the most common analysis tasks, identification and quantification. Since parts of this thesis rely on the OpenMS framework [46] and are implemented in this framework we will use the last part of this chapter to give a short overview of OpenMS.

2.1 Sample preparation

LC-MS experiments involve numerous sample preparation steps. The correct handling of samples and the strict adherence to *Standard Operating Procedures* (SOPs) are essential to get reproducible and valid experimental results.

Here we will only highlight one part of the sample preparation, digestion, due to its connection to the topic of this thesis. Digesting proteins prior to mass spectrometry is a widely used experimental technique, often referred to as shotgun- or bottom-up-approach. The most widely used enzyme for the digestion is trypsin, due to its favorable properties: it cuts protein and peptides after lysine and arginine residues, produces at least two positive charges, one at the N-terminus and one at the C-terminal lysin or arginine, and the generated peptides

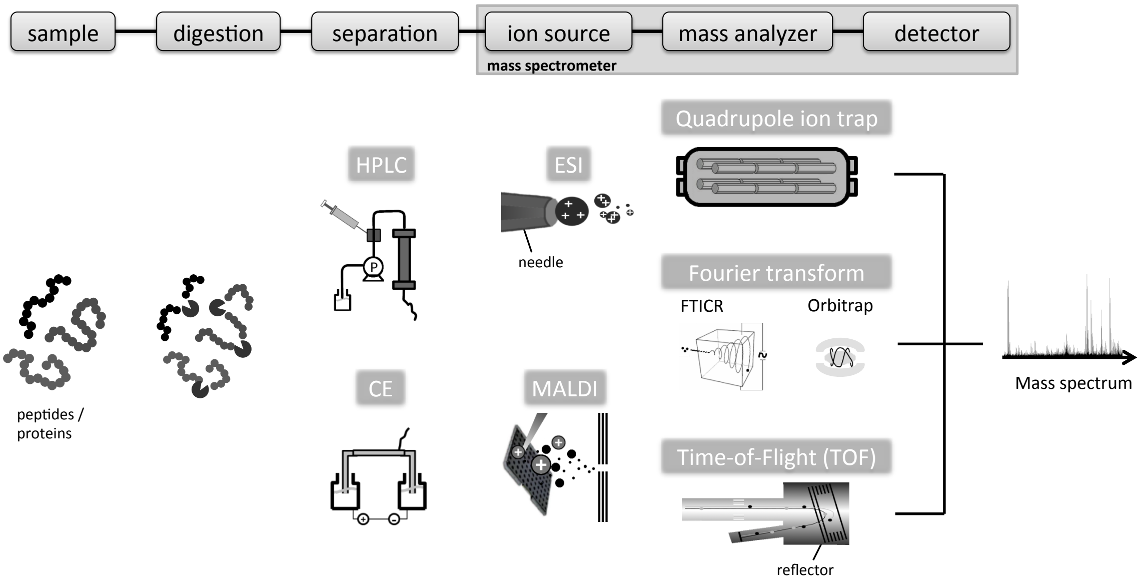


Figure 2.1: Common setup of LC-MS experiments. Shown are the different stages of a typical LC-MS experiment and the most commonly used techniques for separation, ionization, and the mass analyzer. Figure adapted from Bielow [47].

have an average length of 14 amino acids [48].

In contrast to the described bottom-up approach there also exists a top-down approach, where complete proteins are analyzed. The advantage over the bottom-up approach is obviously the possibility to infer the mass of the intact protein. In combination with a list of possible protein sequences it can further be used to infer also the amino acid sequence of the observed protein, including the position and identity of post-translational modifications (PTM).

2.2 Separation

Complex samples like cell lysates or blood serum contain thousands of proteins and after digestion even more peptides. For these samples to be able to be analyzed with a mass spectrometer they need to be separated. Originally this was done using two- or multi-dimensional gel-electrophoresis (2-DE). But lack of automatability in the context of high-throughput experiments led to the replacement by techniques like high-performance liquid-chromatography (HPLC) or capillary electrophoresis (CE).

2.2.1 High-performance Liquid-Chromatography

High-performance liquid-chromatography is the most common separation technique in mass spectrometry based proteomics. Briefly explained, the sample (the peptides) is in solution and the liquid (mobile phase) is pumped under high pressure through a column, which is packed with small particles (stationary phase). The type of particles in the stationary phase determines the type of separation, e.g., by using a charged stationary phase the sample is separated by charge.

The time a peptide needs to travel through the HPLC column, the retention time, is determined by its physicochemical properties. This relation can be utilized in the analysis of MS data sets, e.g., to improve MS/MS identifications by removing false positive hits [49]. Consequently different groups focused their research on predicting these retention times using different machine learning techniques [50, 51]. Alternatively, the predictions can be used to design targeted proteomics experiments [52] or to simulate mass spectrometry data for algorithm development and benchmarking [41] (see Chapter 3).

2.2.2 Capillary Electrophoresis

Alternatively to HPLC, capillary electrophoresis (CE) has become a prominent separation technique in proteomics. The basic principle of CE is based on the different speed of charged particles (in case of proteomics peptide and proteins) in fluid under the influence of an electric field. The peptides or proteins travel through a narrow fused-silica capillary. Each peptide species has an individual speed which depends on its charge, size, shape, and other physicochemical properties. In contrast to HPLC the time needed to travel through the capillary is called migration time.

Both technologies, HPLC as well as CE, are highly automatable due to their direct coupling to mass spectrometry, i.e., the eluting analytes are directly injected into the mass spectrometer. Each technique has individual strengths and weaknesses, CE for instance is very robust and has a high reproducibility, while HPLC has a greater loading capacity [53, 54].

2.3 Mass Spectrometry

Mass spectrometry is an analytical technique that is used to measure the mass or more specifically the mass over charge ratio of analytes. The basic principle of mass spectrometry relies on the manipulation of the trajectories of charged particles using an electromagnetic field. This requires that the analyte is in gas-phase and charged. This can either be a positive (by protonation) or a negative (by deprotonation) charge. Hence mass spectrometers do not measure the mass directly, but the mass-to-charge ratio, often also denoted as mass over charge, or m/z. Here m is the atomic/molecular mass of the analyte in u, also often referred to as *Dalton* (Da) and z the number of elementary charges. As an alternative to the m/z notation the unit *Thomson* was proposed [55], which is defined as $1\mathrm{Th} = 1\frac{\mathrm{u}}{\mathrm{e}}$.

An integral property of any mass spectrometer is the resolution R [56] as it describes the ability to separate two adjacent signals. It is defined as

$$R = \frac{M}{\Delta M}, \tag{2.1}$$

where M is the mass of a singly charged ion and ΔM is the width of the peak at the ”50% of the maximum” peak height. Since the resolution is not constant over the complete mass range of the instrument, it is usually given at a m/z value of 400.0.

The second important property is *accuracy*, the difference between the measured and theoretical mass over charge value of an ion. The accuracy is in general given in *parts per million* (ppm) and is computed as follows

$$accuracy = \frac{m_{obs} - m_{theo}}{m_{theo}} \times 10^{-6}, \tag{2.2}$$

where m_{obs} is the observed m/z value and m_{theo} the theoretical m/z value. The term accuracy should not be confused with the term *precision*. Precision describes the ability of the measurement device to reproduce the measurement, hence if the same measurement is repeated, it describes the deviation between the per run reported mass over charge for the same analyte.

A mass spectrometer basically consists of three subunits, namely ion source, mass analyzer, and detector. In the following sections we will explain the role of these subunits for the mass spectrometer and present the most common available techniques in mass spectrometry based proteomics.

2.3.1 Ion sources

The ion source, as the name implies, ionizes the analyte and transfers it into the gas phase, so that the analyte can be analyzed in the mass spectrometer. In mass spectrometry based proteomics the two most common techniques are Matrix-assisted laser desorption/ionization (MALDI) [26, 27] and Electrospray ionization (ESI) [28].

Matrix-assisted laser desorption/ionization (MALDI)

To ionize a sample using MALDI it first needs to be mixed with a solution of matrix molecules. The resulting solution of sample and matrix molecules is then spotted in small amounts on a surface, usually a metal plate specifically designed for this purpose. Over time the solvent will vaporize and the matrix molecules recrystalize. The analyte molecules will be embedded in those crystals, they are co-crystalized. For the actual ionization a laser is shot in short pulses at the crystalized, spotted sample. This leads to a desorption of material from the spotted sample, containing neutral and ionized particles.

Surface-enhanced laser desorption/ionization (SELDI)

A prominent variation of MALDI is Surface-enhanced laser desorption/ionization (SELDI) [57]. The process is very similar to MALDI with the difference that the sample is first spotted on the surface. The surface has a certain chemical affinity to some of sample molecules (e.g., hydrophobic surface) leading to a selective binding of some of the sample molecules. This step serves as an additional separation step. The unbound molecules are subsequently washed from the surface. The remaining sample is then mixed with a matrix solution directly on the surface.

Electrospray ionization (ESI)

To ionize a sample using ESI it must be already solved in a liquid. This liquid is directed through a capillary. A high voltage is applied to the tip of the capillary leading to an evaporation of the liquid into a fine aerosol and to the formation of ions. The exact physical process is not completely understood. The two most prominent models explaining the process are the ion evaporation model [58] and the charge residue model [59]. A recent overview on electrospray ionization can be found in Wilm [60].

In comparison both techniques and their variations have advantages and disadvantages. For instance ESI can be coupled to a chromatographic column, by directly injecting the eluting sample into the mass spectrometer. For MALDI the eluting sample has to be collected in fractions which are later analyzed, which can also be an advantage: if correctly stored, the collected samples stay stable for several months or even years and can be reanalyzed if necessary [61].

2.3.2 Mass analyzers

The mass analyzer is the integral part of the mass spectrometer. Here the ionized molecules are separated by their mass to charge ratio. Different techniques for the separation exist that are constantly improving in terms of speed, resolution, and accuracy. Common to all techniques is the use of electro-magnetic fields to achieve the separation.

One example of a mass analyzer is the linear time of flight (TOF) analyzer, initially described by Stephens [62] in 1946. The ions are accelerated by an electric field and then introduced into a field-free drift chamber. Subsequently the time is measured till the ions hit the detector which is located opposite to the acceleration area and normal to the acceleration vector. The velocity of the individual ions and with this the time the ions need to travel to the detector is proportional to the square root of the mass-to-charge ratio of the individual ions. A more in depth description of time of flight instruments coupled to MALDI ionization, including an overview of recent developments, can be found in Vestal [63]. Time of flight instruments can achieve high resolutions $> 30{,}000$ [63]. They are often combined with quadrupole analyzers to so called QTOF instruments [64]. In QTOF instruments the quadrupol analyzers serve as mass filter and collision cell for the fragmentation of ions (see Section 2.4.1) which makes them especially suitable for LC-MS/MS studies.

Even higher resolutions ($\gg 100{,}000$) can be achieved with Fourier transform ion cyclotron instruments (FT-ICR) [65, 66]. As an alternative, hybrid instruments combining a linear ion trap with an Orbitrap [67] like the LTQ Orbitrap [68] achieve resolutions up to 100,000.

For the different instruments the resolution may vary over the measured m/z range. In

FT-ICR instruments the resolution degrades with increasing m/z values. In Orbitrap instruments the behavior is slightly better since the resolution degrades only with the square root of the m/z value. Only TOF instruments show a constant resolution over the complete m/z range.

2.3.3 Detectors

The detector of the mass spectrometer, as the name implies, finally detects the separated, ionized molecules. Basically two modes of detection exist. The first one records the ions as they hit the detector plate. These detectors are typically used in Time-of-flight instruments. The second type is contact-free and records ions as they pass near the detector plate, e.g., in FT-ICR instruments.

2.3.4 Data nomenclature

To ease the understanding of the following chapters we will now introduce the basic terminology for mass spectrometry data. Further terms will follow later in their specific context, but we will here introduce the more general aspects. While introducing the different terms we will also highlight associated computational challenges.

Most chemical elements occur in different variants, so called *isotopes*. Isotopes have identical numbers of protons and electrons, but differ in the number of neutrons, leading to a different atomic mass. The individual isotopes of an element occur with different abundances in nature. For instance the two stable isotopes of hydrogen, ^{1}H and ^{2}H occur with the abundances 99.9885 % and 0.0115 %. The third isotope ^{3}H occurs only extremely rarely in nature. The so called *isotope distribution* summarizes these relative abundances in a probability distribution. For isotopes we have to distinguish between multiple types of masses. The *mono isotopic* mass, that is the mass of the isotope carrying the smallest amount of neutrons; and the *average mass*, that is the the sum of all individual isotope masses weighted by their natural abundance. The same notion is used for molecules. Here the mono isotopic mass is the sum of all mono isotopic masses of all atoms forming the molecule, and the average mass the sum of all average atom masses.

For single atoms the probability of carrying an additional neutron is quite low as we can see for example in hydrogen. For peptides and proteins, due to the large amount of atoms, the probability of carrying additional neutrons increases. In a mass spectrometer with a high enough resolution the isotope distribution of peptides can easily be seen. For instance Figure 2.2 shows such isotope distributions for peptides of different masses.

Computing isotope distributions given an empirical formula is straight forward and fast

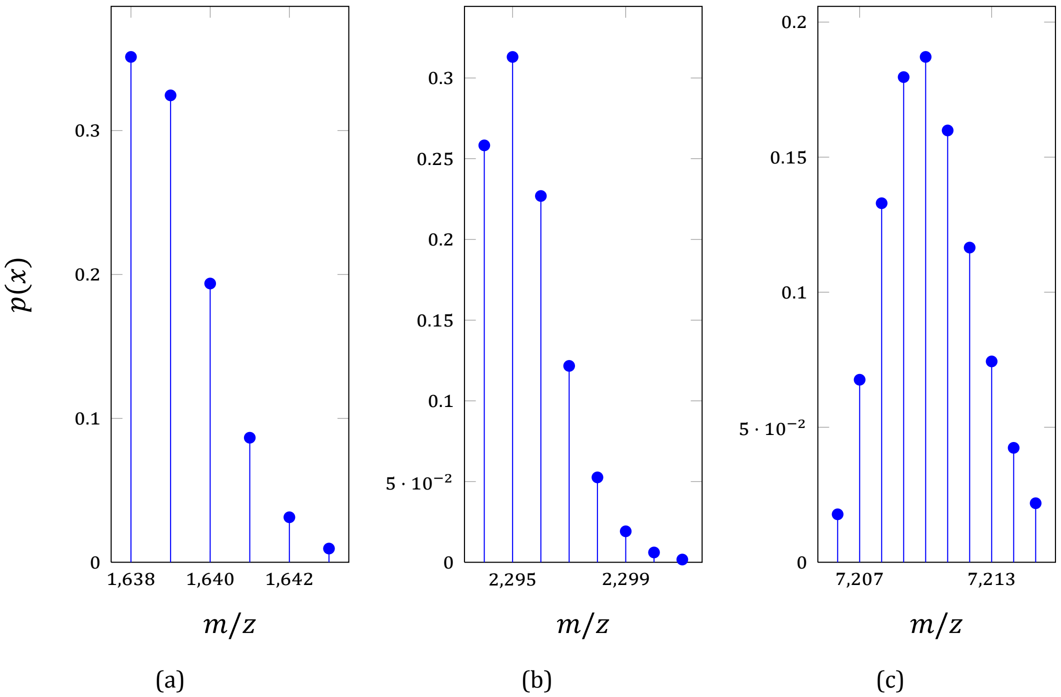


Figure 2.2: Isotope distributions for different peptide fragments of Somatostatin (P61278): (a) Somatostatin-14, (b) Somatostatin-28, and (c) Somatostatin-Propeptide (P61278:25-88).

algorithms exist [69]. If only the mass is known, it is still possible to compute an approximate isotopic distribution. The idea behind the approximate isotope distribution is based on the observation that the number of carbon, nitrogen, oxygen, and hydrogen atoms in peptides grows linearly with the mono isotopic weight of the peptide. Senko, Beu, and McLafferty used this observation to construct an average amino acid, named *averagine*. It has an average mass of 111.1254 Da and its empirical formula is $C_{4.9385}H_{7.7583}N_{1.3577}O_{1.4773}S_{0.0417}$. Given a peptide mass m one now can simply obtain an average isotopic distribution as follows. First we compute the number of averagines needed for the given mass by computing $m/111.1254$. The number of averagine units is now multiplied with the empirical formula of the averagine. For C, N, O, and S this number of atoms is rounded to the nearest integer value. The difference to the original mass m is corrected by adjusting the number H atoms. Using the resulting empirical formula we can again compute the isotopic distribution for the mass m using algorithms as proposed by Kubinyi [69].

We will call the data produced by the mass spectrometer *raw data*. For each m/z value the detector response, i.e., the number of of observed ions for the m/z, is reported. The reported detector response is also called *intensity*. The set of all pairs of m/z and intensity value is called *raw spectrum*. Every charged peptide will form one or multiple so called *peaks* in the raw spectrum. A peak is a local amplitude in the mass spectrum generated by multiple molecules of the same analyte hitting (or passing the detector). While being centered at the theoretical m/z value the amplitude will be spread around the theoretical position with a gaussian like curve shape. The exact shape of this curve is discussed in the community and clearly depends on the used instrument. The most common functions used to describe the shape of the peak are the truncated Gaussian function

$$I\left(x_i\right) = H \exp^{-\frac{\left(x_i - x_0\right)^2}{s^2}}, \tag{2.3}$$

or the Lorentzian function

$$I\left(x_i\right) = H \frac{1}{1 + \frac{\left(x_i - x_0\right)^2}{s^2}}, \tag{2.4}$$

where H is the maximal intensity, x_0 the m/z value of the apex of the peak, and s determines the peak width. But also combinations of the two functions leading to asymmetric peaks are possible. A comparison of the two shapes and their combinations is shown in Figure 2.3.

Besides of common signal processing tasks like noise reduction or baseline filtering the first data analysis step is called centroiding or peak picking. Here the collection of raw data

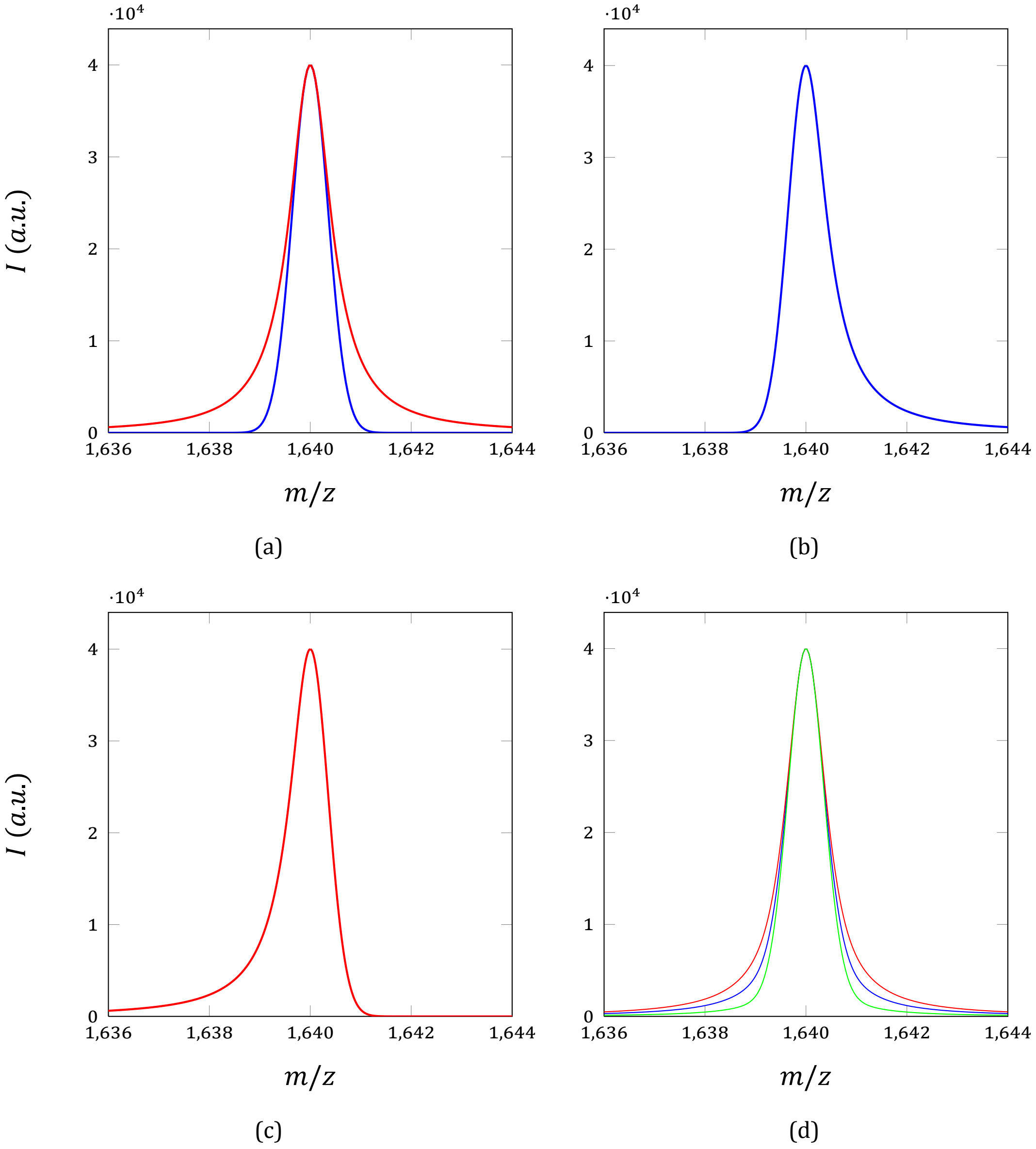


Figure 2.3: Common peak shape functions in mass spectrometry. (a) shows both the Gaussian (blue) and Lorentzian (red) function. In (b) a mixture of Gaussian and Lorentzian is shown, where the function up to the apex is a Gaussian and from there on a Lorentzian. (c) shows the opposite of (b). (d) gives mixtures of Gaussian and Lorentzian with 0.5 Gaussian and 0.5 Lorentzian (blue), 0.2 Gaussian and 0.8 Lorentzian (red), and 0.8 Gaussian and 0.2 Lorentzian (green).

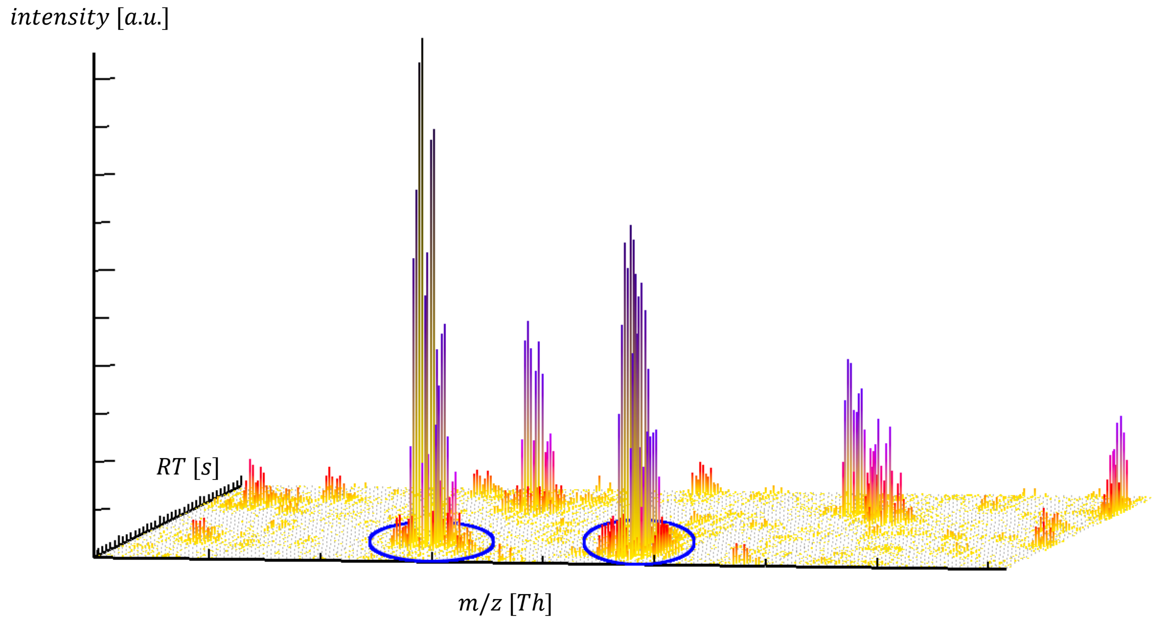


Figure 2.4: Exemplary LC-MS map from a bird's-eye perspective. Shown is a small subset of the map from m/z 824 to 865 and between $3,540$ and $3,900$ $seconds$. Marked in blue are two features.

points forming the peak are condensed into a single pair of m/z and intensity. The centroided m/z corresponds to the m/z of the apex of the peak and the intensity to the area under the curve of the peak. Many modern mass spectrometry instruments are able to directly report centroided data instead of raw data. Depending on the instrument and the confidence in the centroided data provided by the instrument it may be favorable to perform the centroiding by oneself. Here different algorithms have been proposed to solve this problem. An overview and comparison of different available approaches was given by Yang, He, and Yu [71]. In some situations the term peak may also be used for the centroided data point.

In case of LC-MS setups not only one mass spectra is generated, but multiple spectra are collected over time while the sample elutes from the LC column. The collection of spectra or scans annotated with their specific retention time is called a *LC-MS map*. An example of such an LC-MS map is shown in Figure 2.4.

Peptides, or biomolecules in general, do not elute at a single time point, but with varying amounts over multiple distinct RT scans. The graph describing how much of the analyte is eluting from the column over time is called elution profile. The shape of this elution profile is a topic of discussion in the literature. J. Li gives an overview of different available shape functions and their ability to reproduce real elution profiles [72].

The signal generated by a single eluting peptide consisting of its isotope distribution in the m/z dimension and the elution profile in the retention time dimension form a 3-dimensional signal which we call a *feature*. Finding these features automatically is consequently a major challenge in the computational analysis of LC-MS data.

In special cases features can be grouped, e.g., because they originate from the same peptide but have different charge states or when two different LC-MS maps are compared in which case they likely represent the same peptide. We call such grouped features *consensus features*.

2.4 Analysis of LC-MS data

In the previous sections we described the basic principles of mass spectrometry based proteomics and introduced the basic terms used in this field. In the following section we will now describe the two most common analysis tasks, identification and quantification. We will conclude this section with a short summary and, for the interested reader, references to additional literature.

2.4.1 Identification

One major goal in LC-MS experiments is the identification of the peptides and proteins in the sample. In the following sections we will describe two common approaches, peptide mass fingerprinting and tandem mass spectrometry.

Peptide mass fingerprinting

The basic idea behind *Peptide mass fingerprinting* (PMF) is based on the accurate measurement of the analyte's mass. The acquired mass is compared to a database of theoretical peptide masses and the peptide that matches best to the observed mass is selected. The approach was proposed by different groups independently in the 1990s [73, 74, 75, 76, 77]. As long as the sample is known and the mass accuracy is high enough the PMF approach can lead to reasonable results. But the task of finding a unique identification gets difficult as soon as the search space increases for instance due to the addition of post translational modifications to the database or an increased database size (e.g., the complete human proteome).

The accurate mass and time tag (AMT) approach [78] is an extension of the peptide mass fingerprinting idea. Here the retention time of the peptide is incorporated into the identification, resulting in a more specific identification. In addition to the sequence database required

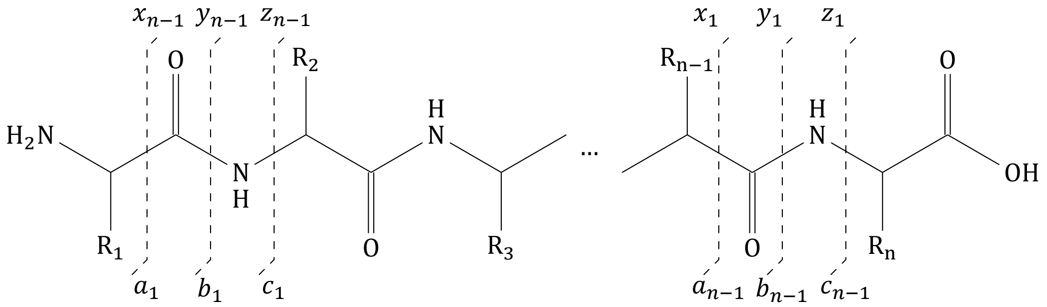


Figure 2.5: Shown is the peptide backbone with annotated break points and the corresponding naming convention. The a, b, and c fragments denote the N-terminal fragments of the peptide and the x, y, and z fragments the C-terminal ones. The indices indicate how many N- or C-terminal amino acids of the peptide are contained in the fragment, e.g., the a_1 fragment contains only the first amino acid of the chain whereas the y_{n-1} contains all but the first amino acid.

to match the observed to the theoretical mass, a database of retention times is required. The article by Zimmer et al. [79] gives an excellent overview of the AMT approach.

Tandem Mass Spectrometry

Tandem mass spectrometry extends the previously described mass spectrometry analyses by selecting a certain m/z range for further analysis. All ions in the selected m/z range are fragmented and the resulting ions are again analyzed with the mass spectrometer. The fragmentation is achieved for example by colliding the ions with neutral gas molecules like helium, the so called *Collision-induced dissociation*. Common alternatives are *Higher-energy C-trap dissociation* (HCD) [80] and *Electron-transfer dissociation* (ETD) [81]. The resulting mass spectrum is called tandem MS spectrum, MS/MS spectrum, or MS^2 spectrum. Ideally the selected mass range is so small that it contains only one peptide species. This peptide is called the *precursor ion* or *precursor peptide*.

What makes tandem mass spectrometry interesting is that the fragmentation occurs mostly at the backbone of the peptide, producing fragment ions with defined mass differences, so called *ion ladders*. The fragmentation sites for a classical peptide are depicted in Figure 2.5. The mass differences in between the peaks of the ion ladder correspond to the mass of the additional or respectively missing amino acid. Hence the mass difference can be used to infer part or even the complete amino acid sequence of the precursor peptide.

The problem of sequencing a peptide based on a tandem MS spectrum is computationally very challenging. Two different approaches exist to solve this problem. The first one, using solely the observed ion ladder and the mass of the precursor peptide as input, is the *de novo* approach. This approach is especially useful if no prior information of the peptides contained in the sample exist, e.g., because no database of protein sequences for the analyzed organism exists. The *de facto* standard for *de novo* peptide identification is PepNovo [82], but alternatives like Antilope [83] and PEAKS [84] exist.

If enough information on the analyzed organism exists to compile a list of candidate peptide sequences an alternative solution is the so called *database search*. Here the acquired tandem MS spectrum is matched against theoretical tandem MS spectra generated from the amino acid sequences in the database. Based on a scoring schema the best matching theoretical spectrum is selected as identification. Widely used algorithms for database search are Mascot [85], Sequest [86], OMSSA [87], and X!Tandem [88]. The results obtained by the individual search engines can also be combined to improve the overall results in the so called consensus scoring or ConsensusID approach [89].

Another important aspect of tandem mass spectrometry is the selection of precursor ions. With increasing complexity of the sample a tandem MS spectrum cannot be generated for every signal of interest. Most mass spectrometers mainly select the most intense signals for fragmentation. The selection happens in most cases based on a single survey scan, a full mass spectrum. Based on this survey scan one or more precursor ions are selected for fragmentation. But also more sophisticated selection approaches that try to maximize the number of identified peptides exist [90].

2.4.2 Quantification

The second important goal in mass spectrometry is the quantification of the peptides and proteins in the sample. In the last years different approaches for absolute and relative quantification have been proposed. We will give here a short overview of the general approaches by considering the comparison of two biological states, e.g., healthy and diseased. However, most of the techniques can readily be extended to more then two samples.

Two general classes of quantification approaches exist, label-free and labeled quantification. For each of the classes different subclasses exist, operating either directly on MS data or on tandem MS data.

Labeled quantification

The term labeled-quantification bundles a multitude of different experimental approaches used to quantify different samples in single MS measurements. All of them are based on the idea that stable-isotope-labeled peptides behave identically or nearly identically to their natural counterparts. By introducing stable-isotope labels the peptide signals are shifted in the mass spectrum and allow a direct comparison of the individual signals by combining the labeled and unlabeled peptides in a single mass spectrometry experiment. These stable-isotopes can either be introduced chemically or metabolically. Both approaches have different advantages and disadvantages.

Metabolic labeling, where stable-isotope labeled versions of amino acids are added to the growth medium of cells, is the earliest point at which modifications can be introduced. This allows an early combination of the different samples and reduces variations due to individual biochemical treatment or mass spectrometry measurements. The most prominent technique is stable isotope labeling with amino acids in cell culture (SILAC) [91]. Here stable-isotope versions of lysine and arginine are added to the growth medium, ensuring that most of the tryptic peptides carry a modification.

In *chemical labeling* the labels are introduced either on protein or on peptide level by attaching chemical modifications containing stable isotopes to the proteins or peptides in the sample. Prominent techniques are ICAT [92], ICPL [93], and ^{18}O stable isotope labeling [94]. A special subgroup of chemical labeling consists of techniques using so called *isobaric tags*. Two approaches are widely used that are based on isobaric tags, namely TMT (tandem mass tags) [95] and iTRAQ (isobaric tags for relative and absolute quantification) [96]. For both approaches the individual labels consist of three parts: the *reactive group*, which creates the chemical connection between the peptide and the modification, the *reporter group*, which has specific mass for every modification, and the *balancer group* which ensures that the complete modifications have an equal mass. Due to the equal mass the modified peptides are indistinguishable in the MS scan, but when fragmented the reporter groups are detached from the peptide and are visible in the tandem MS spectrum. Since the reporter mass differs for all labels, they form individual signals in the tandem MS spectrum that can be used for quantification. The masses of the reporter ions are between 113 and 121 Da for iTRAQ and 126 and 131 Da for TMT and therefore do not interfere with the MS/MS identification. TMT and iTRAQ are available in two different versions, TMT duplex and 6-plex [97] and iTRAQ 4-plex and 8-plex [98].

In summary, most of the chemical labeling approaches are easier to realize, while the metabolic labeling approach reduces the technical variability of the measurement due to the earlier combination of the samples.

Label-free quantification

The term label-free quantification also bundles multiple approaches for quantification of peptides and proteins. What all of these approaches have in common is that the samples, in contrast to labeled quantification, are measured individually and only the final datasets are combined for analysis and quantification. The difference between the approaches is in the way the peptide and protein intensities or expression values are obtained. The three main approaches are (i) MS intensity based approaches, (ii) spectral counting, and (iii) selected reaction monitoring.

In MS intensity based approaches all individual samples are measured with the classical LC-MS/MS workflow as we described it earlier in this chapter. The resulting LC/MS maps are combined in two steps, feature finding and alignment. Feature finding, as we explained earlier, is the task of finding peaks that belong to the same peptide. To solve this problem different algorithmic approaches have been proposed in the last years, like Superhirn [99] or msInspect [100]. In the alignment step the features, identified in the individual maps, are linked together by either using their specific m/z value and retention time or by utilizing existing MS/MS identifications. A review of existing methods can be found in [101]. Following the alignment the grouped features can be, after careful normalization [102], compared based on their signal intensities.

In contrast the spectral counting based approaches do not use the detector response (i.e. signal intensity) for quantification but the number of identifications obtained for a peptide or protein. The concept is based on the observation that the number of identifications for a given peptide are correlated with the abundance of the corresponding protein in the sample. Prominent methods based on spectral counting are Spectral Counting [103], the exponentially modified protein abundance index emPAI [104], or RIBAR and xRIBAR [105]. While these approaches have a certain acceptance in the community there still are problems, like the effect of the chosen identification method [106] that should be kept in mind when choosing a spectral counting approach. The interested reader is referred to Colaert, Vandekerckhove, et al. [107] and Colaert, Gevaert, and Martens [105] for an overview and comparison of available techniques.

Selected reaction monitoring (SRM) or multiple reaction monitoring (MRM) is a so called targeted approach. In contrast to the previously described approaches no survey scan is generated nor a full MS/MS spectrum. Only specific combinations of precursor mass and fragment ion mass are measured, so called transitions. The quantification is obtained from the chromatogram of the fragment ion intensity. The transitions are chosen in a way that they are specific for a peptide and ideally also for a protein. This allows a highly accurate quantification of a limited number of proteins in the measured sample. Selecting a set of transitions (a so called SRM assay) for a specific experiment in a complex mixture of proteins that are still unique is a complex task and different solutions are available to this problem. An overview of available solutions can be found in [108].

2.4.3 Summary

This section has introduced two of the most common tasks in the analysis of mass spectrometry data. As one can see, different solutions for both (identification and quantification) exist and choosing the correct one heavily depends on the experimental setup, the available resources, and the research objective of the particular study. Domon and Aebersold give a very good overview on available techniques and try to answer the question which solution to choose when [109].

After careful selection of the experimental strategy the problem remains how to analyze the data. Different solutions exist and one of them we will present in the following section. OpenMS provides solutions for both labeled and label free quantification as well as advanced tools for the identification of peptides in LC-MS experiments. A recent overview of available tools can be found in the review by Müller, Brusniak, et al. [110]. Cappadona et al. give a recent overview of open challenges in quantitative mass spectrometry which should be considered when designing an experiment [111].

2.5 OpenMS – An open-source framework for mass spectrometry data analysis

OpenMS [46] is a C++ library for the analysis of mass spectrometry data. It was designed with the aim to provide the necessary data structures and algorithms to handle mass spectrometry data in an efficient and easy-to-use manner. The core data structures and algorithms were designed with the goals efficiency, robustness, extensibility, portability, and ease-of-use.

The OpenMS library is divided into two parts, the core library (OpenMS), that provides all the data structures for data handling and loading as well as the analysis algorithms, and the GUI library (OpenMS_GUI) that bundles all the graphical user interface aspects of OpenMS needed to visualize mass spectrometry data. To implement all this, OpenMS relies on several third party libraries that are, except Qt [112], shipped as the OpenMS contrib. OpenMS provides a separate build system for the contrib that can be used to easily build the contrib with a single command on Windows, Linux, and Mac OS X. The contrib contains Xerces-C++ [113], used for XML parsing, Boost used for different mathematical operations and, used together with bzip2 [114] and zlib [115], to read and write compressed files. Further the contrib contains the GNU Scientific Library (GSL) [116] for different statistical operations, the GNU Linear Programming Kit (GLPK) [117] to solve linear programming problems, and libSVM to solve machine learning problems. SeqAn [118] is used for different sequence related tasks (e.g., suffix arrays of peptide sequences). An overview of the structure and dependencies of

OpenMS is shown in figure Figure 2.6.

OpenMS and OpenMS_GUI are C++ libraries that can be used by programmers to easily and fast implement new tools and algorithms. To ease the access to the data provided by the mass spectrometry instrument, OpenMS implements the most common mass spectrometry raw data formats like mzML [119], mzXML [120], and mzData [121]. For the exchange of identification and quantification information OpenMS provides specific formats like featureXML, consensusXML, and idXML, as well as the new PSI standards mzIdentML [122], and mzQuantML [123].

For those users not capable of programming their own applications, OpenMS provides TOPP – the OpenMS proteomics pipeline [124]. TOPP provides most of the algorithms implemented in OpenMS as small, easily combinable applications.

To ease the generation of complex workflows using the TOPP tools OpenMS provides additionally TOPPAS [125], a graphical workflow editor. Furthermore OpenMS can be integrated into the comprehensive data analysis platform KNIME [126]. To inspect the resulting data OpenMS provides TOPPView [127], a powerful visualization tool for mass spectrometry data.

The reader is referred to Sturm and Kohlbacher [127], Junker et al. [125], and Kohlbacher et al. [124], and Sturm [128] for excellent overviews as well as in-depth descriptions of the capabilities of the individual graphical tools, TOPP, and OpenMS respectively.

Contributions to OpenMS

As parts of this thesis were implemented in OpenMS, the author contributed to several parts of OpenMS. Next to numerous bug fixes and code improvements, the new simulation tool MSSimulator (which will be introduced in the following chapter) was added to OpenMS. Additionally the authors contributed TMTAnalyzer, an adaption of the existing ITRAQAnalzyer [47] for TMT data sets, and the existing FeatureFinderCentroided was extended to work also with asymmetric elution profile shapes.

Further the contrib build system was completely rewritten based on CMake [129]. Large parts of the build and test system of OpenMS were extended. The author ported OpenMS to work on Mac OS X and wrote the native installer for Mac OS X.

The author also implemented the integration of OpenMS into KNIME. The implemented approach, used to integrate OpenMS into KNIME, can also be utilized to integrate any other command line tool into KNIME.

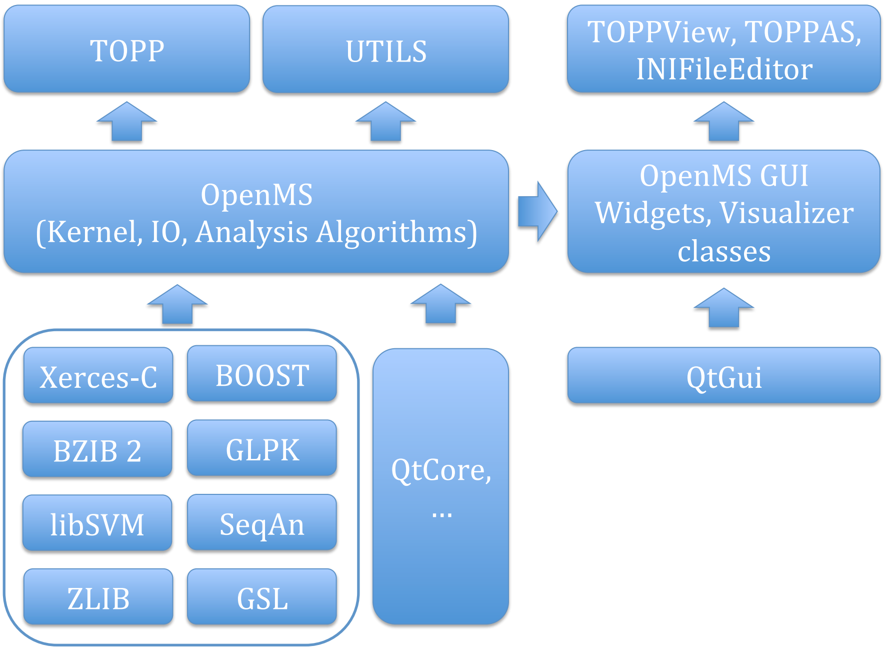

TOPP
UTILS
TOPPView, TOPPAS, INIFileEditor
OpenMS (Kernel, IO, Analysis Algorithms)
OpenMS GUI Widgets, Visualizer classes
Xerces-C
BOOST
BZIB 2
GLPK
libSVM
SeqAn
ZLIB
GSL
QtCore, ...
QtGui

Figure 2.6: The architecture and dependencies of the OpenMS library. On the bottom of the picture the dependencies of OpenMS are shown. In the center the two OpenMS libraries are shown. The top shows the different types of applications provided by the OpenMS library.

Chapter 3 A new Ground Truth for Computational Mass Spectrometry

3.1 Why we need a ground truth

Analyzing dynamic processes like proteolysis using mass spectrometry based techniques is a complex task, since the experimental setup can be very involved and time consuming. In contrast, the development of methods to analyze the generated data sets requires the availability of a sufficient amount of benchmark data. Ideally these datasets would reflect different experimental conditions (e.g., varying timeframes or different noise levels) to assess the performance and limitations of the developed methods. Another important requirement is that all information about the underlying process is available, meaning that the degraded peptides, the reactions, the products, and the reactions rates of the proteolytic process are known. But these datasets are only available in a very limited number.

This problem is not limited to the analysis of proteolytic processes using MS data. Recent advancements in the development of high-throughput mass spectrometry enable the research community to generate thousands of spectra in a short period of time. With the amount of generated data the need for sophisticated algorithms that can analyze these data sets increases. Developing such algorithms and tools is a laborious task. The differences between individual data sets in mass spectrometry based proteomics due to differing measurement techniques (e.g., mass analyzer or LC column) can be drastic. Benchmarking on different data sets is therefore essential. In contrast to other disciplines (e.g., multiple sequence alignments [130]), carefully compiled databases with annotated test data sets are scarce in mass spectrometry based proteomics. Hence the availability of sufficient test data is a major problem in the process of developing algorithms for the analysis of mass spectrometry data [131].

We therefore propose MSSimulator [41], a versatile simulator for LC-MS/MS experiments. The developed solution can be used in many different scenarios to ease the development and benchmarking of algorithms, as we will show at the end of this chapter.

3.2 Existing approaches

The idea of using simulated mass spectrometry measurements is not new. In 2005 Coombes et al. [132] presented a simulation approach for MALDI TOF spectra based on their Cromwell software. Morris et al. [133] used this software to generate simulated benchmark datasets to assess the performance of their new feature extraction and quantification approach. Schulz-Trieglaff et al. [134] presented the first comprehensive approach to simulate LC-MS data and used it to benchmark different feature detection approaches [134]. Renard et al. [135] used a quite simple simulation strategy to validate the NITPICK feature finding algorithm [135]. In 2009 Yang, He, and Yu [71] used the simulated datasets from Morris et al. [133] to benchmark different peak picking algorithms.

However, all of these approaches were focused only on the simulation of a subset of the complete LC-MS/MS experimental procedure. Therefore we designed MSSimulator with the aim to include most of the steps of a classical LC-MS/MS experiment.

3.3 Structure of MSSimulator

MSSimulator was implemented using C++ and is based on the OpenMS [46] library (see Section 2.5). MSSimulator is also available in TOPP, The OpenMS Proteomics Pipeline [124]. The simulator can be configured using a xml file, which can be edited either manually or using INIFileEditor, a dedicated GUI shipped with OpenMS. MSSimulator uses FASTA files as input, in addition to the configuration file. The FASTA file contains the protein or peptide sequences in single-letter amino acid code. The amino acid sequences can also contain modifications[1]. The FASTA header can further contain protein/peptide specific information like the abundance or a specific retention time.

MSSimulator also supports the addition of contaminations to the simulated experiments. All that is required is the elemental composition of the contaminant, i.e., its empirical formula (e.g., CH_3OH for Methanol). A detailed description of the file format for the contaminants is given in [41, Supporting Information].

MSSimulator consists of several submodules, accounting for the different phases of a LC-MS/MS experiment. Each individual step and the underlying model will be explained in the following sections.

[1] All modifications contained in UNIMOD [136] are supported.

3.3.1 Digestion

In many LC-MS workflows, digesting the sample with a protease like Trypsin is the first step. MSSimulator is able to simulate the digestion based on two different strategies. Furthermore, if no digestion is wanted, e.g., when simulating a top-down experiment, it can also be disabled using the configuration file.

The first strategy performs a complete *in-silico* digestion, i.e., every potential protease cleavage site is targeted. It further simulates missed cleavages up to a user defined threshold. If missed cleavages are simulated, the completely cleaved peptides will still be contained in the sample.

The second strategy is based on a model from Siepen et al. [137]. The described procedure was reimplemented in OpenMS to predict missed cleavages. The underlying model included in MSSimulator is based on trypsin data, but can easily be adapted. The user simply needs to substitute the text file containing the model parameters. For an extension to other enzymes the user needs to compute the log likelihood ratio data matrix described in [137].

3.3.2 Peptide separation

Due to the constantly increasing complexity of the samples in modern mass spectrometry based proteomics experiments, prefractioning of the digested peptides is an inevitable step. MSSimulator therefore supports two widely used approaches for peptide separation: High Performance Liquid Chromatography (HPLC) and Capillary Electrophoresis (CE). The two techniques use different physicochemical properties of the peptides to separate the peptides and therefore complement each other. For the HPLC simulation MSSimulator uses a machine learning based approach. It utilizes support vector regression to predict a retention time for each peptide in the sample. In contrast, the simulation of the capillary electrophoresis is based on a theoretical, linear model which we will later explain in more detail.

Liquid Chromatography

High Performance Liquid Chromatography (HPLC) is a widely used pre-fractioning technique in mass spectrometry based proteomics. A mobile phase containing the analyte is pumped through a column containing the stationary phase. Depending on the used stationary phase and the peptides the time for passing the column will differ.

Schulz-Trieglaff et al. already applied the Paired Oligo-Border Kernel (POBK) [50] to accurately predict the retention times for peptides in their simulation. MSSimulator uses the same approach combined with a more flexible noise model for the predicted retention time. The noise model consists of two components: a gaussian noise with a user defined standard deviation, that shifts the predicted retention time, and an affine transformation (as proposed in [138]) with user defined scale and offset to model inter-experimental variations of the predicted retention times. A trained model for the Paired Oligo-Border Kernel is provided with MSSimulator. Using tandem MS identifications one can also train a custom model using the RTModel tool which is part of TOPP.

Capillary Electrophoresis

In capillary electrophoresis molecules are separated in a strong electric field. Dependent on different physicochemical properties peptides show different migration times in the electric field. The migration time is further determined by the background electrolyte and its properties, e.g., type of ions, pH, or ionic strength.

The migration time model used by MSSimulator mainly focuses on the correct simulation of the electrophoretic mobility (μ_{ep}) of the peptides. In contrast, the electroosmotic flow (μ_{eo}), which is mainly governed by the viscosity of the buffer and the capillary itself, is a parameter that can be customized by the user.

The electrophoretic mobility is predicted based on two physicochemical properties of the peptide, namely mass and net charge. A well known mathematical model for the electrophoretic mobility is

$$\mu_{ep} = q/MW^{\alpha}, \tag{3.1}$$

where q is the net charge of the ion, MW is its molecular weight and α is a constant. If an electric field is applied in a vacuum the speed of the peptide is proportional to its net charge. However, in a medium (i.e., the electrolyte) one needs to correct for frictional drag. This is done by the MW^{α} term. Choosing an optimal α is an extensively discussed topic. The most common values are $\frac{1}{3}$, $\frac{1}{2}$, $\frac{2}{3}$, which all relate to different theoretical models. For details on the choices of α and charge determination see [41, Supporting Information].

To finally determine the migration time of a single peptide with known weight and net charge we compute

$$t = \frac{L_d L_t}{(\mu_{ep} + \mu_{eo})V}, \tag{3.2}$$

where L_d is the distance between injection site and detector, L_t is the total capillary length and V is the applied voltage (see [139]). The above presented model can also predict negative migration times. These peptides are discarded but will be mentioned in a summary statistic.

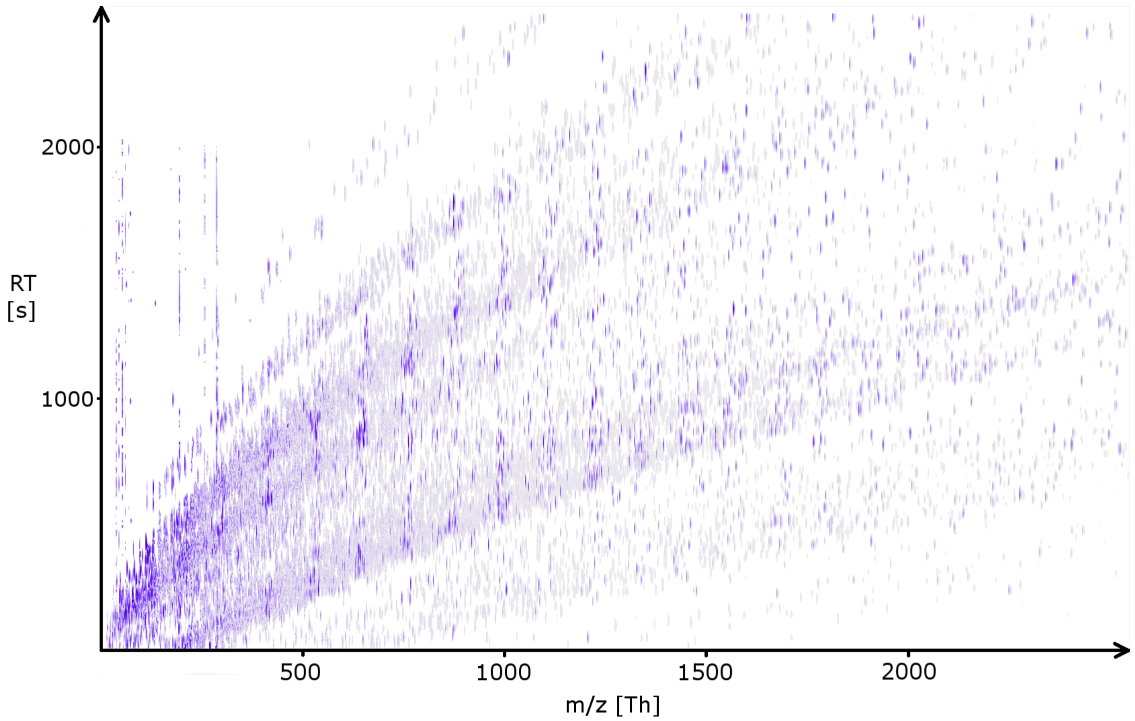


Figure 3.1: Simulated raw CE/MS map of 100 proteins using the default CE settings of MSSimulator.

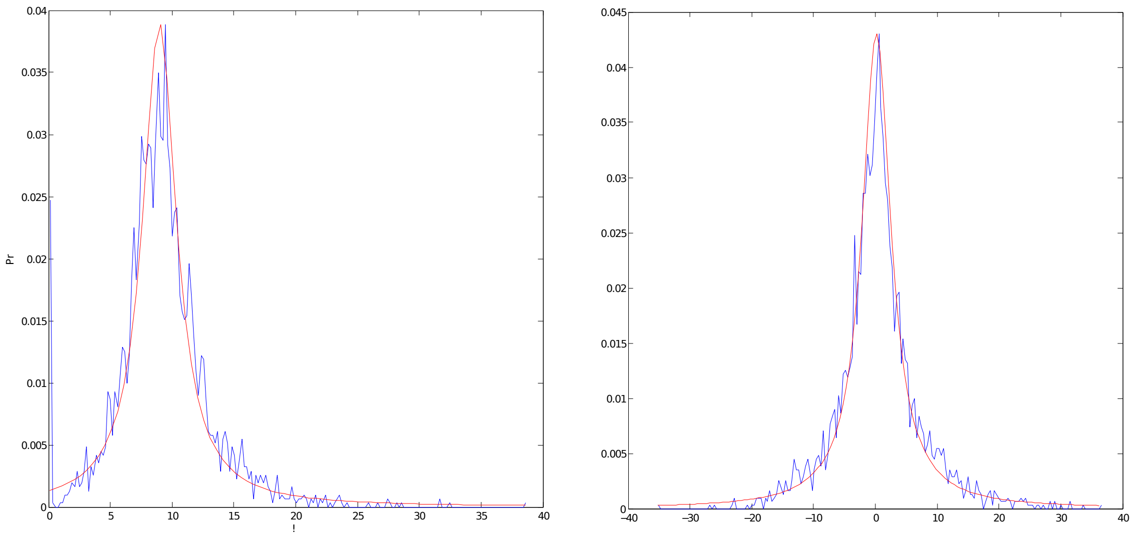
A major difference between capillary electrophoresis and HPLC is the different behavior of the peak width. In a typical HPLC the peak width stays constant, while in CE the peak width increases with the migration time. This is due to dispersion factors and decreased mobility. MSSimulator uses a linear model to include this effect in the simulation.

Figure 3.1 shows a CE/MS experiment that was simulated using MSSimulator with default CE settings. The typical CE charge bands can easily be observed in the simulated data.

3.3.3 Elution profile shapes

Regardless of whether CE or HPLC is used as pre-fractioning technique, peptides will not elute at a single time point but over a period of time in varying amounts. This generates a so called elution profile for each peptide. The shape of this elution profile in the retention time dimension needs to be modeled as a part of the final signal.

In many cases the Gaussian function is used to model the elution of peptides from a chromatographic column. While it is easy to use, it is unfortunately not able to model asymmetric elution profiles which can often be observed in experimental data. To account for such asymmetric elution profiles MSSimulator uses the exponential-Gaussian hybrid function (EGH) as



(a) (b)

Figure 3.2: The distribution of the observed parameter values (blue) and the fitted Lorentzian distribution (red) for the two main parameters of the EGH function that is used to model the shape of elution profiles: (a) σ_g (b) τ

it was presented in [140] to model elution profiles:

$$f_{egh}(t) = \begin{cases} H \exp\left(\frac{-(t-t_R)^2}{2\sigma_g^2+\tau(t-t_R)}\right), & 2\sigma_g^2 + \tau(t-t_R) > 0 \\ 0, & 2\sigma_g^2 + \tau(t-t_R) \leq 0 \end{cases}, \tag{3.3}$$

where t is the retention time, t_R the center of the chromatographic peak, H the peak height, σ_g the standard deviation of the peak and τ the time constant of the exponential decay. The sign of τ will determine if it is an exponential decay or growth.

Using the EGH to simulate elution profiles requires choosing adequate values for the parameters H, t_R, τ, and σ_g. H and t_R simply reflect the intensity and the predicted retention time of the peptide, but for τ and σ_g there is no specific relation to the peptide or any of its features known. To still choose realistic values for the EGH we carried out a series of experiments on real LC-MS datasets in order to find a realistic distribution of values for τ and σ_g. To achieve this we modified the TOPP FeatureFinderCentroided to use the EGH function instead of the Gaussion function, to estimate the shape of the elution profile. We then extracted for the found features the estimated values for τ and σ_g. Figure 3.2 shows the normalized counts for the different observed τ and σ_g values. As one can see in Figure 3.2 the distribution of the values seems to follow a Lorentzian distribution

$$f(x) = \frac{1}{\pi} \frac{\gamma}{(x - x_0)^2 + \gamma^2}, \tag{3.4}$$

where x_0 is the location parameter and γ the scale of the Lorentzian distribution.

Based on this observation MSSimulator provides the ability to sample for each peptide signal, both τ and σ_g, from two separate Lorentzian distributions. The default values were estimated from the above shown data using the curve fit functionality of Matlab®.

To reflect poor chromatographic conditions the user can also customize the quality of the generated elution profiles. In this process MSSimulator adds uniformly distributed noise to the elution profile followed by a moving average filter to reduce the effect of outliers.

3.3.4 Filtering peptides by their detectability

There are multiple reasons why a peptide could not appear in an LC-MS experiment. Some of them are coupled to ionization (e.g., poor ionization) or to chemical properties like solubility in the used mobile-phase of the LC column. To model this effect also into the data generated by MSSimulator, the peptide detectability filter introduced in Schulz-Trieglaff et al. [134] was included in MSSimulator. It is based on a support vector machine combined with a paired oligo-border kernel. It computes the likelihood of each peptide to create a signal in a mass spectrum. All peptides below a certain user defined threshold will then be discarded. MSSimulator is shipped with a trained model, based on the date presented by Mallick et al. [141]. Customized models, based on own datasets, can easily be generated using TOPP's PTModel tool. The detectability filter, if not applicable, can also be disabled.

3.3.5 Ionization

In modern mass spectrometry based proteomics two ionization techniques are prevalent: Electrospray Ionization (ESI) and Matrix-assisted laser desorption/ionization (MALDI). Both techniques are supported by MSSimulator.

For Electrospray Ionization the charge of peptide heavily depends on the number of basic residues. MSSimulator therefore models the charge states as a Binomial distribution

$$p(k) = \binom{n}{k} p^k (1-p)^{n-k}, \tag{3.5}$$

where $p(k)$ is the probability of a peptide to have a charge k, given it has n basic residues.

p is the probability of each basic residue to carry a charge. By default p is set to 0.8. MSSimulator further supports custom charge adducts like Na^+ or K^+.

For Matrix-assisted laser desorption/ionization MSSimulator samples the probability for each charge state from a discrete distribution. The default probabilities are $P(q = 1) = 0.9$ for charge 1 and $P(q = 2) = 0.1$ for charge 2, but can also be customized in the configuration file.

3.3.6 Modeling peptide signals in the Mass Spectrum

In the previous sections we collected for each peptide information about its charge state, the retention time and the shape of the elution profile. Using this information MSSimulator computes the final LC-MS signal for each peptide.

Each peptide signal consists of two basic components, i.e., the shape of the elution profile and the isotopic signal in m/z dimension. The shape of the elution profile was already computed as shown in Section 3.3.3. To compute the signal in the m/z dimension a fast algorithm by Kubinyi [69] is used. It computes the isotopic envelope of the peptide based on its amino acid composition. The algorithm was already implemented in OpenMS and hence could be used easily in MSSimulator. The result of the algorithm is a set of isotopic masses and the corresponding intensities. Again the problem arises that a mass spectrometry detector does not generate a single isotopic peak at a specified m/z value, but distributes the intensity over a defined range. Thus we need to model a specific peak shape for each of the computed pairs of isotopic mass and intensity. The optimal model of this peak shape is topic of extensive discussions in the mass spectrometry community as well as in the literature [142]. MSSimulator provides the two most common models, the truncated Gaussian function and the Lorentzian function. Both shapes have been shown in Chapter 2, Figure 2.3. The actual width of the peaks can be controlled by the user in terms of the resolution R [56]:

$$R = \frac{M}{\Delta M}, \tag{3.6}$$

where M is the mass of a singly charged ion and ΔM is width of the peak at 50% of the maximum peak height. For different mass spectrometry instruments the resolution may vary with m/z. TOF instruments have a constant resolution, while FTICR instruments show a linearly degrading resolution with increasing m/z. In Orbitrap instruments the resolution degrades with the square root of m/z. To account for these differences, MSSimulator provides all three models. The user only specifies the resolution R_{400} at $m/z = 400.0$ Th and then MSSimulator computes at each sampling point the specific resolution for the chosen resolution behavior.

To now generate a final signal for a given peptide, MSSimulator computes a product model of the generated signals in the m/z and RT dimension.

3.3.7 Tandem MS sampling

Independently of the previously generated mass spectrum, MSSimulator provides the capability to generate tandem MS signals for selected peptides. To generate a realistic tandem mass spectra special emphasis has to be put on an accurate prediction of the fragment intensities. The fragment intensities depend heavily on the used fragmentation technique. Since collision induced dissociation (CID) is one of the most commonly used fragmentation techniques, several methods have been proposed to predict the fragment intensities in CID spectra. Most of these approaches are based on machine learning techniques like probabilistic decision trees [143], neural networks [144], Bayesian neural networks [145], or RankBoosting [146]. Alternatively Zhang [147] proposed a kinetic model to predict fragmentation for low energy CID spectra.

MSSimulator provides three different modes to simulate tandem MS spectra.

The first, naïve mode gives the user the possibility to select the ion types (including neutral loss ions and charge variants) to simulate. The user can also specify the intensity for the individual ion types.

The second mode is based on a support vector machine (SVM) classifier. The classifier is trained to predict the absence, presence, and abundance of the primary ion types (b- and y-ions for CID spectra). It uses the 35 descriptors introduced by Zhou, Bowler, and Feng [145], to encode each individual peptide bond in the simulated peptides. See [41, Supporting Information] for more details on the descriptors and their usage in MSSimulator. During the training phase the training spectra are searched for peaks within an interval around the expected m/z value. For the training a class balanced set is used which contains spectra with abundant as well as missing samples. Suitable values for the SVM are obtained via grid search.

The third mode is based on support vector regression (SVR). The regression is used to predict the intensity for the individual fragment ion peaks. To speed up the final prediction, only the intensities of the b- and y-ions are predicted using the SVR. The intensity of the neutral loss ions is predicted using a Baysian approach, where the probability of observing a certain loss ion with a certain intensity is learned based on the predicted intensity of the corresponding primary ion. Since the approach predicts only discrete intensity levels, we apply intensity binning.

The SVM and SVR approach are currently only supported for a maximum charge of three. MSSimulator is shipped with a trained model for both approaches but custom models can also be trained based on a user provided dataset.

3.3.8 Labeled experiments

Chemical and metabolic labeling are important analysis and quantification techniques in modern mass spectrometry based proteomics. Therefore MSSimulator provides a framework that allows the fast and easy incorporation of any labeling technique into the simulation. Currently MSSimulator provides four widely used techniques, namely ICPL (isotope-coded protein label) [93], iTRAQ (isobaric tag for relative and absolute quantitation) [96], SILAC (stable isotope labeling by amino acids in cell culture) [91] and ^{18}O labeling [94]. For each labeling channel the user needs to provide a separate FASTA input file. It allows the user to model different protein and peptide compositions for the individual channels, as well as differences in abundance and modification state.

ICPL labeling

MSSimulator comes with a predefined labeling mechanism for ICPL labeling [93]. The original two- and three-channel ICPL labeling is based on the modification of the free amino groups of denatured proteins with either the deuterium free (light channel) or the deuterium containing (heavy channels) version of the ICPL reagent. MSSimulator further supports the post-digest ICPL workflow as it was proposed by Fleron et al. [148]. Extending the ICPL labeling mechanism to further support the fourth channel (ICPL_10) is straight forward.

iTRAQ labeling

MSSimulator also supports the simulation of iTRAQ experiments. Both iTRAQ modes, 4plex and 8plex, are supported. The user can arbitrarily allocate the different channels to the tandem MS reporter ions. MSSimulator further supports custom isotope correction matrices and is shipped with the default matrix provided by Applied Biosystems.

Further the efficiency of the labeling of the tyrosine residues can be modified by the user. The default value is set to 30 %. Peptides containing tyrosine residues are split into two sibling peptides, having different masses. The abundance of the sibling peptides reflects the labeling efficiency (e.g., 30 % vs. 70 % for the default labeling efficiency). For lysine residues and the N-terminus MSSimulator assumes a labeling efficiency of 100 %. The MS/MS spectra generated for iTRAQ labeled peptides contain the reporter ions in the m/z range from $113 - 121$ Th. The fragment ions have a mass shift of $+145$ Da for each modified amino acid contained in the peptide.

Stable Isotope Labeling by Amino Acids in Cell Culture

SILAC is a prominent approach in quantitative proteomics based on the incubation of cell lines with an isotopically labeled form of an amino acid (e.g., deuterated leucine). MSSimulator currently supports experiments with three different SILAC channels, each having a defined label for lysine and arginine. Except the light channel as this one carries no modification. The default labels used by MSSimulator introduce a mass shift of 4.0 and 6.0 Da for the medium channel and 8.0 and 10.0 Da for the heavy channel lysine and arginine respectively. MSSimulator assumes a 100 % incorporation of the modification, but implementing incomplete incorporation is straight forward.

^{18}O Stable Isotope labeling

Another widely used chemical labeling approach in quantitative proteomics is ^{18}O Stable Isotope Labeling. Labeling peptides with ^{18}O tags is achieved by digesting the proteins with an endoprotease (usually trypsin) in the presence of $H_2^{18}O$. During the digestion the two C-terminal oxygen atoms are exchanged by the heavier ^{18}O atoms thereby introducing a mass shift of +4.0 Da. Unfortunately the labeling reaction is not always complete and also monolabeled peptides carrying only one ^{18}O atom (resulting in a mass shift of +2.0 Da) and unlabeled peptides are generated. Therefore MSSimulator provides the ability to modify the labeling efficiency. Based on this value the abundance B of each labeled peptide is split up on the three different states: $i)$ $\mathbf{B_0}$, the unlabeled state (the abundance will be merged with the abundance of the unlabeled channel), $ii)$ $\mathbf{B_1}$, the monolabeled state, and $iii)$ $\mathbf{B_2}$ the di- or completely labeled state. The abundances of the different states are distributed based on the labeling efficiency f according to the kinetic model described by Ramos-Fernández, López-Ferrer, and Vázquez [149]:

$$B_0 = B\,(1-f)^2 \tag{3.7}$$

$$B_1 = B2f\,(1-f) \tag{3.8}$$

$$B_2 = Bf^2. \tag{3.9}$$

3.4 Output of MSSimulator

MSSimulator generates multiple output files that provide multiple layers of information. The first, most important, and the only mandatory output file is the raw MS data in mzML [119] format. Alternative output formats (e.g., mzData [121] or mzXML [120]) can also be generated by transforming the mzML to the desired target format using TOPP's FileConverter tool.

The second layer of information provided by MSSimulator is the feature data as feature map (in OpenMS' featureXML format). It contains all information about the simulated peptides, annotated with their position (m/z and retention time), their charge, the used charge adducts, and the specific peptide sequence (containing possible modifications). The data can be converted to a textual representation like a csv (comma-separated values) file using TOPP's TextExporter. From there on the data can easily be opened in spreadsheet applications like Microsoft® Excel.

A second featureXML file contains the same information for all contaminants simulated by MSSimulator.

As third output MSSimulator provides an additional mzML file containing the exact positions of each simulated peak. The data can be used to readily benchmark peak picking algorithms.

The fourth group of results generated are consensus maps (in the OpenMS specific consensusXML format) containing information about the associations between the generated signals. The first consensus map contains the association information between all charge states of each peptide, e.g., to ease the benchmarking quantification approaches or decharging of feature data. The second consensus map holds all information about the associations of labeled and unlabeled (or differentially labeled) peptides. Again consensusXML can easily be converted to csv and then be viewed, edited, and analyzed.

3.5 Benchmarking the data generated with MSSimulator

One of the remarkable features of MSSimulator is its high configurability, due to which MSSimulator can easily be adapted to mimic certain instrument types. MSSimulator is shipped for instance with example configuration files for QTOF and FT instruments. Other configurations can easily be generated if key parameters like the instrument resolution and certain noise parameters are known.

To show that the data produced by MSSimulator is consistent with real data, MSSimulator was configured to use the same instrument setup and protein mix as it was used in two datasets (Mix 3, low-resolution QTOF and high-resolution Fourier Transform (FT) data) of the Standard Protein Mix Database [150]. To now assess how well the simulated data resembles real data the exact same analysis pipeline (centroiding, feature finding) was applied to both datasets. Subsequently the number of identified features, the charge distribution, and the intensity range of both datasets were compared and showed a high agreement (see

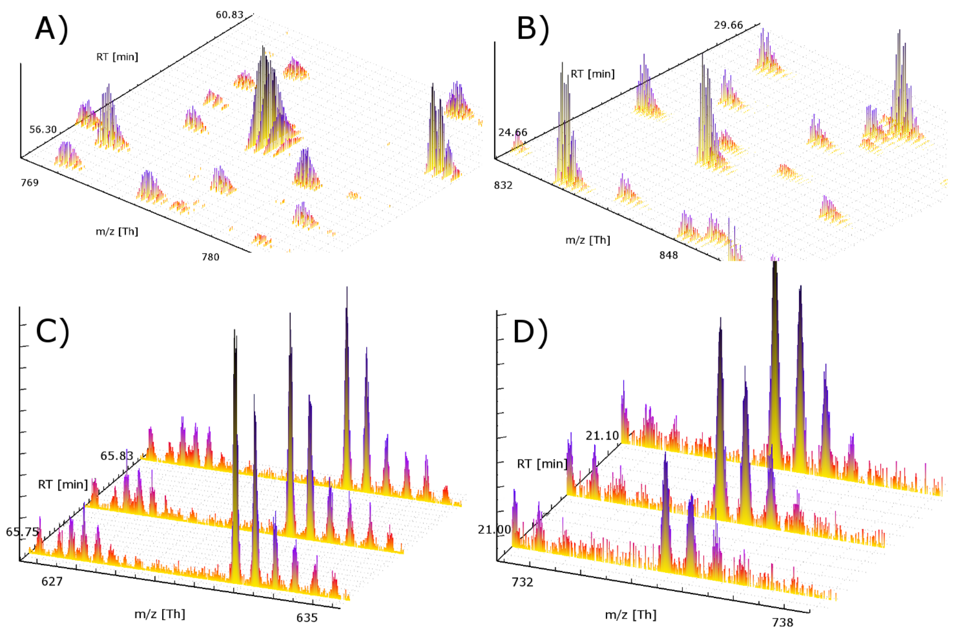


Figure 3.3: Comparison of real vs. simulated data for FT and QTOF instruments. For clarity, data is shown on zoomed regions of an LC/MS map. A) real FT data, B) simulated FT data, C) real QTOF data, D) simulated QTOF data.

	FT real	FT simulated
# Scans	2,715	2,546
# Features	3,011	3,649
Intensity range [a.u.]	1.07×10^4 - 1.84×10^8	1.08×10^4 - 4.35×10^7

Table 3.1: Comparison of key parameters for simulated and real datasets based on the B06-11071 dataset from Mix3, Standard Protein Mix Database [150].

Table 3.1 and Figure 3.4). For a visual comparison see Figure 3.3.

The initial motivation for the development of MSSimulator was to ease the development and benchmarking of algorithms for mass spectrometry data. Therefore MSSimulator was used to benchmark different algorithmic problems in mass spectrometry and the results are summarized in the following subsections.

3.5.1 Comparison of two established approaches for SILAC quantification

Developing efficient algorithms for the quantitative analysis of labeled or unlabeld mass spectrometry datasets is a very laborious task. Accurate benchmarking requires the manual annotation of data sets as gold standard for a later comparison. Given different instrument types and settings this can become a very time consuming and error-prone task.

To prove the value of MSSimulator in such a setup, a comparison of two known approaches

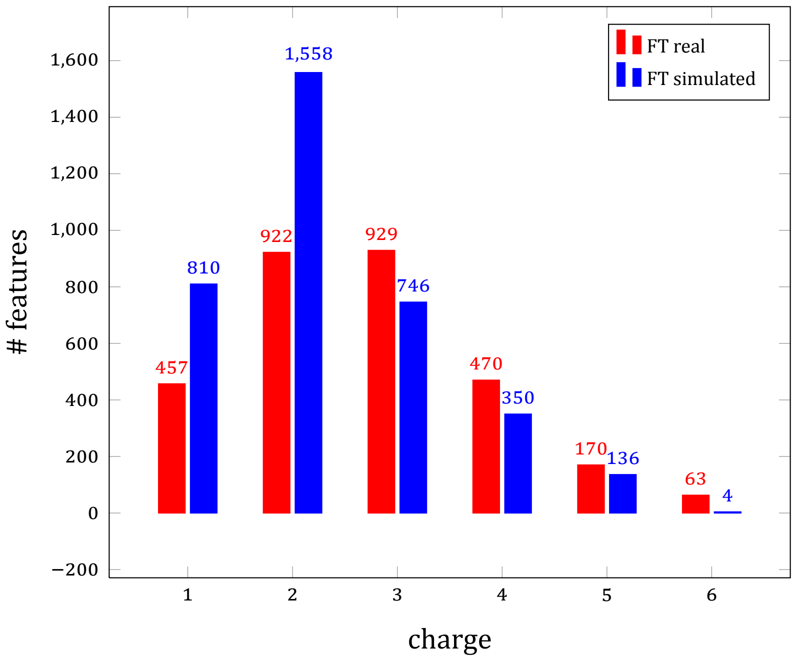


Figure 3.4: Comparison of the charge distribution for simulated and real datasets based on the B06-11071 dataset from Mix3, Standard Protein Mix Database [150].

for quantification of SILAC datasets, namely XPRESS [151] and ASAPRatio [152], was carried out on simulated data. The Trans-Proteomic Pipeline (TPP)[2] [153] implementation of both approaches was used for the experiments.

The XPRESS software was originally designed to work on data labeled with isotope-coded affinity tags (ICAT) [92], but can also handle arbitrary modifications. XPRESS identifies the coeluting profiles of the labeled pairs using tandem MS identifications and determines the abundance based on the area of each chromatographic peak for each channel.

ASAPRatio also works on the chromatographic peaks, but employs a more elaborate processing of the extracted chromatograms and a more sophisticated error analysis then XPRESS. ASAPRatio starts by extracting multiple chromatograms for the first three theoretical isotopic peaks of the peptide identified by tandem MS. Afterwards ASAPRatio smooths the chromatograms by applying a Savitzky-Golay filter, removes background noise, and calculates the area under the individual chromatographic peaks. Subsequently the ratios of all peptides with the same sequence but different charge states are combined into a single peptide ratio.

To now benchmark the performance of XPRESS and ASAPRatio a dataset was generated using MSSimulator containing an unlabeled and a labeled channel. For the labeled channel arginine and lysine were modified introducing a mass shift of ≈ 6.02 Da. The proteins had

[2] TPP v4.4.1 (VUVUZELA)

four different ratios: 1 : 1, 1 : 2, 1 : 4 and 1 : 10. After applying the naïve trypsin model for digestion and the HPLC simulation on the column provided with MSSimulator, the dataset contained 782 different peptide features. Following the simulation of the raw mass spectrometry data we generated exact identification for all peptide features. This was done to ensure that side effects like inaccurate tandem MS identifications do not influence the final results. These identification results were converted into the pepXML[3] format using TOPP and analyzed by XPRESS and ASAPRatio. Both tools produce again pepXML annotated with computed peptide ratios. In Figure 3.5 the computed peptide ratios are plotted against the original simulated ones. XPRESS as well as ASAPRatio could reconstruct most of the SILAC pairs. But, as one would have expected, both tools have, to a different extend, problems with overlapping signals. Finally one can state that ASAPRatio showed its superiority over XPRESS due to its more robust error analysis.

The presented concept for the comparison and benchmarking of different quantification approaches can easily be extended to other labeling techniques. It further is easy to automate for the evaluation of different parameter sets or under varying conditions (e.g., instrument parameters like noise or resolution). Using the presented workflow one can easily assess the influence of all those parameters on a newly developed or existing tool by simply utilizing the availability of the ground truth (i.e., feature positions, simulated ratios, etc.).

3.5.2 Benchmarking feature detection in High-Resolution data

A second and also very time consuming task is the development of feature detection algorithms for mass spectrometry data. To overcome the problem of exact location and charge state for peptide signals being unknown in real datasets, manually annotated ones are often used.

In this section we will show a complementary approach using MSSimulator as it was previously described in works of Morris et al. [133] and Schulz-Trieglaff et al. [134]. Using simulated data eases the evaluation of the computed features in terms of false discovery rate (FDR) and true positive rate (TPR), since the exact feature location, its charge, and intensity are known. Further it is very easy to test the robustness of algorithms to mass spectrometry specific factors like noise or resolution.

As an example scenario we will here use simulated date to compare the performance of Hardklör [154] (v1.34), an established feature detection tool for high resolution data, and the FeatureFinderCentroided (FFC) that is shipped with TOPP. The test datasets were generated based on 18 different proteins including multiple contaminations as described in Klimek et al. [150]. As is instrument setting the earlier described FT instrument preset was used.

[3] http://tools.proteomecenter.org/wiki/index.php?title=Formats:pepXML

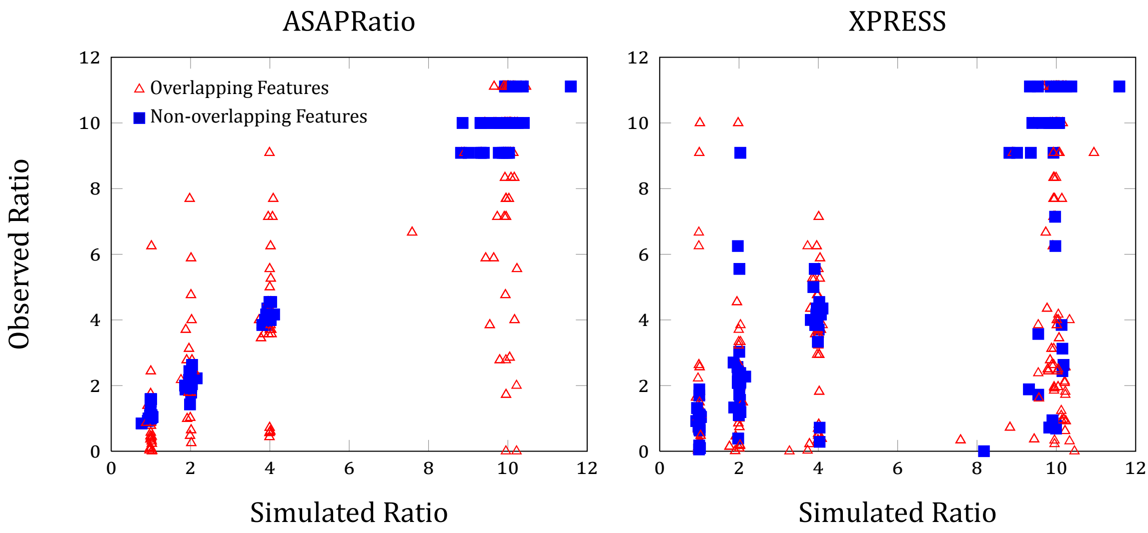


Figure 3.5: Ratios computed by ASAPRatio (left) and XPRESS (right) plotted against the ratios simulated by MSSimulator. Peptide features that overlap with at least one other feature (which is not the labeled partner) are marked as red triangles, non-overlapping features are marked as blue squares.

Hardklör was run with slightly modified parameters according to the ”Sample Config Files” section on the Hardklör website[4]. Hardklör reports features spectrum-wise, therefore the results of Hardklör were post-processed with Krönik (v1.3)[5], to combine features that persist over multiple LC-MS scans.

The OpenMS quantification pipeline consists of the PeakPickerHiRes to generate centroided data. Subsequently the FeatureFinderCentroided was applied using customized parameters (see [41, Supporting Information]).

The configuration files for both tools and a description of the analysis steps can be found in [41, Supporting Information]. The FDR and TPR values were computed based on the features simulated by MSSimulator.

Hardklör as well as the FeatureFinderCentroided showed a good performance on simulated data. The FDR and TPR values are shown in Table 3.1(a).

An important question here is the influence of the chromatographic conditions. To assess the influence of these conditions a second dataset with an increased distortion of the elution profiles was generated. For both tools the performance slightly dropped. The effect on the performance (FDR and TPR values) is shown in Table 3.1(b).

The presented method to assess the performance of two feature detection approaches is

[4] http://proteome.gs.washington.edu/software/hardklor/config.html
[5] http://proteome.gs.washington.edu/software/hardklor/programs.html

(a) $distortion = 4.0$

	FDR	TPR
Hardklör	0.07	0.855
FFC	0.196	0.814

(b) $distortion = 8.0$

	FDR	TPR
Hardklör	0.115	0.798
FFC	0.203	0.642

Table 3.2: False discovery rate (FDR) and true positive rate (TPR) for Hardklör and FeatureFinderCentroided under two different chromatographic conditions. Chromatographic distortion was set to (a) 4.0 and (b) 8.0.

easy to implement and requires only a small effort in data preparation (compared to the usually required manual annotation of real datasets). It could also be easily applied to benchmark existing or a self developed software and assess the influence of data specific properties like chromatographic conditions on the performance of these tools.

3.6 Contributions

As MSSimulator is a joint work with Chris Bielow and Sandro Andreotti, we will shortly summarize the parts of MSSimulator that were explicitly developed by the author. The author reimplemented and restructured the code of the predecessor tool LC-MSSim and integrated it into OpenMS. The previously monolithic tool was modularized into classes corresponding to the individual simulation steps (e.g., digestion, retention time prediction). The exponential gaussian hybrid shape for the retention time was implemented as a realistic and easily customizable function for the elution profile. Additionally the OpenMS FeatureFinderCentroided was customized to also use this shape, so as to easily generate default parameter values based on the parameter values extracted from real datasets. We further implemented parts of the noise models (e.g., detector noise, distortion of elution profiles) and the 1D (no LC column) simulation. Also the generic labeling framework and the SILAC, ICPL, and ^{18}O labeling were implemented. Moreover the most time consuming parts of MSSimulator were parallelized using OpenMP [155].

Chapter 4 Modeling Proteolytic Processes: The degradation graph

Proteolytic enzymes are one of the largest enzyme families in human and are involved in numerous biological processes. In recent years the importance of proteolytic enzymes in the context of complex diseases has been recognized and a lot of effort has been made to study individual proteolytic enzymes, reactions, or even enzyme families. What is still missing is a complete understanding of the interactions between different proteolytic enzymes and their targeted proteins and peptides, as well as the dynamic behavior of the reactions, i.e., the reaction rates of the individual proteolytic reactions.

This chapter will introduce our approach to get a systematic view on proteolytic processes. The method is based on mass spectrometry time series data and attempts to combine all the information contained in the data to improve our understanding of the observed proteolytic process. All the information is combined in a single model, the *degradation graph*. The degradation graph, as it is described in this thesis, is similar to the *cleavage graph* introduced by Kluge, Gambin, and Niemiro [37]. In contrast to the original *cleavage graph*, the degradation graph is able to model also endoproteolytic reactions, that were not included in the approach of Kluge, Gambin, and Niemiro [37].

In the following chapter we will describe the degradation graph. We will start with an overview of existing approaches to analyze proteolytic processes. Afterwards we will introduce the degradation graph formally and describe how the degradation graph can be constructed from mass spectrometry data. Subsequently we will show how the degradation graph can be used to estimate the reaction rates of the underlying proteolyitc process by translating it into a system of ordinary differential equations (ODE) combined with a short introduction to mathematical modeling of biological reactions. In the last part of this chapter we will present a strategy to optimize the structure of the initially constructed degradation graph, when it was constructed from noisy data.

4.1 Existing approaches

Different approaches were presented in the last years to analyze proteolytic processes using mass spectrometry. Most of them are focused on the identification of the cleavage site on either peptide or protein level. This can be done for instance by comparing two samples, one treated with a specific proteolytic enzyme and one untreated, by chemical labeling, like it was done in the study by Enoksson et al. [156]. Other approaches rely on the identification of novel C- or N-termini, formed by proteolytic reactions [157, 158] or even both termini using the COFRADIC approach [159]. A recent review article by van den Berg and Tholey [34] gives an excellent overview of the available techniques.

Only few approaches were made to not only identify the cleavage sites but also to model the dynamics and interactions of the individual degradation processes. Yi et al. [36] modeled the degradation of fibrinopeptide A as a sequential multi-step reaction (SMSR). They used AQUA [160, 161], a method for absolute quantification, to acquire intensity values that they subsequently used to estimate the reaction parameters of the SMSR model.

Kluge, Gambin, and Niemiro [37] proposed the *cleavage graph*, a data structure to model proteolytic processes in LC-MS data sets. Each node represents a peptide and two nodes a and b are connected if the sequence of b can be obtained by either removing the N- or C-terminal amino acid from a. In this construction approach only exoproteolytic reactions could be modeled. The nodes inside the graph were generated based on tandem MS identifications. The activity of the proteolytic process was modeled by means of the *Chemical Master Equation* and included a stationarity assumption, i.e., the model had a constant influx of peptides. That assumption is debatable, especially in the context of mass spectrometry measurements. In parallel to our efforts, Gambin and Kluge [162] extended their model to also include endoproteolytic reactions. They also removed the stationarity assumption introduced the original article. In Dittwald et al. [38] the model was again extended by improving the estimation procedure of the model parameters.

4.2 Definition of the degradation graph

A series of interacting, proteolytic reactions can be modeled as a graph $G = (V, E)$. The different peptides that are degraded and generated correspond to the nodes V and the proteolytic reactions converting the peptides are represented by the edges E. Under normal, physiological conditions proteolysis is irreversible, therefore we can model the proteolytic reactions as directed edges, that are directed from the peptide that is targeted by the proteolytic enzyme to the one generated by the proteolytic reaction.

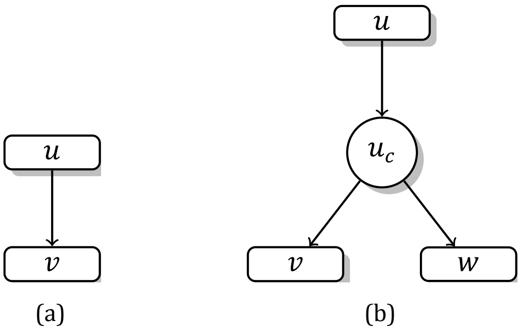


Figure 4.1: Representation of different proteolytic reactions inside the degradation graph. (a) Exoprotease reaction, (b) Endoprotease reaction. See Figure 3 for an example of a degradation graph that contains both reaction types.

The different proteolytic reactions are represented as follows. Exoproteolytic reactions, where a single amino acid is removed from one of the peptide termini, are modeled by a directed edge from a node u to v if we can obtain the amino acid sequence of v by removing a single amino acid from the N- or C-terminus of the amino acid sequence of u. Due to the small mass the removed amino acid is not modeled directly in the graph. In case of endoproteolytic reactions we need to consider both products of the reaction. The goal is to have a single edge in the graph that represents the reaction of cutting a peptide u at position c into two smaller fragments v, w. Since an edge only connects two nodes we need to break the idea of one reaction equals one edge. Since we still want to associate all relevant information on a reaction to a single edge, we add a pseudo node u_c to the graph. The pseudo node is labeled with the sequence of u and the cutting position c, which makes the node unique inside the graph. We then connect u to u_c and u_c with v and w. The later two edges are called pseudo edges, since they do not carry any information.

Figure 4.1 shows both types of reactions separately. A more complex example with real peptide sequences is shown in Figure 4.2.

4.3 Constructing a degradation graph from Mass Spectrometry Data

In the previous section we defined the degradation graph and explained its relation to proteolytic processes. In the following section we give an algorithm to construct the degradation graph based on a series of N mass spectra collected at different time points $t_1 \dots t_N$ and a seed sequence S that is assumed to be processed by unknown proteases. The seed sequence could either be known based on the conducted experiment (i.e., a known peptide was incu-

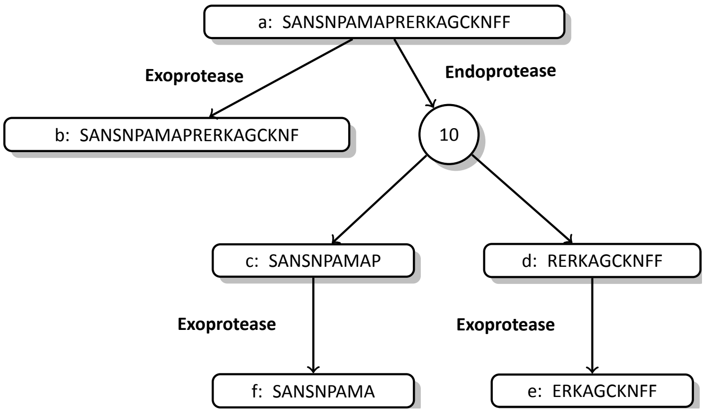


Figure 4.2: Artificial protease system acting on a single peptide (SANSNPAMAPRERKAGCKNFF) and the resulting degradation products.

bated with unknown proteases) or based on tandem MS identifications. Given this input we search in the mass spectra for signals that originate from fragments of S, generated by the unknown proteolytic process.

The approach to construct the degradation graph is divided into two main steps, verification and extension. Both steps rely on the identification of signals in the mass spectra that correspond to the searched peptides. We will start by explaining the approach used in this thesis to identify peptide signals, followed by a description of the verification and extension steps. Both, verification and extension, are executed successively on all mass spectra. Before the first mass spectrum can be processed the degradation graph needs to be initialized. A node for the seed sequence S is added to the degradation graph, the root node. With the initialized degradation graph the verification step is executed on the first mass spectrum. This is followed by the extension step. The two steps are then repeatedly executed on the remaining mass spectra. The pseudo code for both parts is shown in Figure 4.3.

To ease the understanding of the details of the two algorithmic steps we shortly introduce some notation. Given a node v in the degradation graph, $s(v)$ denotes the amino acid sequence of the peptide associated with the node v. The length of the amino acid sequence is given by $|s(v)|$. $s(v)[a,b]$ with $1 \leq a < b \leq |s(v)|$ is the subsequence of the amino acid sequence from position a to position b. $m(v)$ denotes the mass of the peptide associated with the node v. If we identify a signal that corresponds to the peptide associated with v, we denote its intensity with $I_{m(v)}(t_i)$. The association between mass and intensity takes into account that mass spectrometers cannot distinguish peptides with equal mass to charge ratios. Hence a signal could be associated with different peptides with the same mass to charge ratio. By introducing the mapping we avoid counting the signal twice in the later analysis. The set of all peptide masses in the graph is denoted by M. We further introduce a queue of nodes

L, which is empty at the beginning of the construction. This queue will store all nodes of the degradation graph that need to be verified and extended.

4.3.1 Identification of peptide signals

As already stated, the identification and correct association of peptides to mass spectrometry signals is crucial for the following analysis steps. As described in Section 2.4.1, identification of peptides can be done in two ways, using

1. Tandem MS spectra [163] combined with *de-novo* identification algorithms or database search approaches,
2. peptide mass fingerprinting [73], where peptides are identified solely based on the mass-to-charge ratio of the peaks in the mass spectrum.

In the remainder of this thesis we use a simple peptide mass fingerprinting approach but an incorporation of tandem MS identifications is also possible. The applied peptide mass fingerprinting approach works as follows.

In general we assume that all mass spectra were centroided prior to the analysis and that the area under the curve of the original peak was assigned as intensity to the centroided peak. We then distinguish between low and high resolution mass spectra.

For low resolution mass spectra, i.e., the isotopic peaks are not individually recognizable but are merged into a single peak, we search for a peak with the average mass (divided by the charge) in the mass spectra. We allow a user defined maximal difference between the theoretical mass and the peak mass. If more then one peak is found, the one with the smallest distance to the theoretical average mass is chosen. The reported intensity is the intensity of the peak (i.e., the area under the curve of the original peak).

For high resolution mass spectra, i.e., the individual isotopic peaks are recognizable, we start by searching for a peak with the monoisotopic mass of the peptide. Again if more than one peak is found, we choose the one with the smallest distance. We then try to extend the isotopic pattern by searching for peaks with a distance of $+1/c$, where c is the assumed charge of the peptide. For each new peak we find we require an intensity ratio below a certain threshold ($t = 0.9$), to avoid collecting equally spaced noise peaks instead of real peptide signals.

4.3.2 Verification of the degradation graph

Each iteration of the algorithm on a new mass spectrum i is started with the verification of the degradation graph with respect to the new spectrum. For the given mass spectrum we need

```
function VERIFICATION(degradation draph $G$, Spectrum $P$, Time $t$)
    $L \leftarrow \{\}$
    for each node $v$ in $G$ do
        if $P$ contains a signal $p$ for peptide $s(v)$ then
            $L \leftarrow \{L, v\}$
            $I_{m(v)}(t) \leftarrow$ intensity of $p$
        end if
    end for
    return $L$
end function

function EXTENSION(degradation graph $G$, Spectrum $P$, Time $t$, Node List $L$)
    for each node $u$ in $L$ do
        if $P$ contains a signal $p$ for peptide $s(u)[2, |s(u)|]$ then
            create node $v$, with $s(v) \leftarrow s(u)[2, |s(u)|]$ and $I_{m(v)}(t) \leftarrow$ intensity of $p$
            add edge $u \rightarrow v$
            $L \leftarrow \{L, v\}$
        end if
        if $P$ contains a signal $p$ for peptide $s(u)[1, |s(u)| - 1]$ then
            create node $v$, with $s(v) \leftarrow s(u)[1, |s(u)| - 1]$ and $I_{m(v)}(t) \leftarrow$ intensity of $p$
            add edge $u \rightarrow v$
            $L \leftarrow \{L, v\}$
        end if
        for each $c$, $2 < c < |s(v)| - 1$ do
            if $P$ contains signals $p_v, p_w$ for peptides $s(u)[1, c]$ and $s(u)[c + 1, |s(u)| - 1]$ then
                create nodes $u_c$
                add edge $u \rightarrow u_c$
                create node $v$, with $s(v) \leftarrow s(u)[1, c]$ and $I_{m(v)}(t) \leftarrow$ intensity of $p_v$
                create node $w$, with $s(w) \leftarrow s(u)[c + 1, |s(u)|]$ and $I_{m(w)}(t) \leftarrow$ intensity of $p_w$
                add edge $u_c \rightarrow v$
                add edge $u_c \rightarrow w$
                $L \leftarrow \{L, v, w\}$
            end if
        end for
    end for
end function
```

Figure 4.3: Pseudo code for the degradation graph construction algorithm. The notation is defined in the text.

to check which nodes of the degradation graph have a corresponding signal in the spectrum by applying the above descried identification approach given the sequence attached to the node $s\left(v\right)$. Each node that could be identified in the mass spectrum is appended to the queue L. We further annotate it with the observed intensity $I_{m(v)}\left(t_i\right)$.

4.3.3 Extension of the degradation graph

The second step on each spectrum is the extension of the degradation graph. The extension procedure is repeated as long as the queue L is not empty. In each iteration a node u is removed from the beginning of L and processed as follows.

The procedure starts by emulating the exoproteolytic degradation of the peptide. It removes the N- and C-terminal amino acid separately from $s\left(u\right)$ and searches for the corresponding signals. If a signal can be identified we add a node v to the graph, annotate it with the signal intensity $I_{m(v)}\left(t_i\right)$ and set its sequence $s\left(v\right)$ to either $s\left(u\right)\left[2, \left|s\left(u\right)\right|\right]$ or $s\left(u\right)\left[1, \left|s\left(u\right)\right| - 1\right]$. Afterwards we connect the nodes u and v by an edge pointing from u to v. The newly generated node v is then appended to the queue L.

Subsequently the endoproteolytic reactions are emulated. The amino acid sequence $s\left(u\right)$ is divided into two parts for each position c with $2 < c < \left|s\left(u\right)\right| - 1$. If both fragments can be identified in the mass spectrum a pseudo-node u_c is added to the degradation graph, annotated with the amino acid sequence $s\left(u\right)$ and the cutting position c and connected to u. Afterwards a node for both fragments v and w is added to the graph, annotated with the corresponding signal intensities ($I_{m(v)}\left(t_i\right)$, $I_{m(w)}\left(t_i\right)$), the amino acid sequences ($s\left(u\right)\left[1, c\right]$ and $s\left(u\right)\left[c+1, \left|s\left(u\right)\right|\right]$), and connected to the pseudo-node u_c. Both newly generated nodes v, w are appended to the queue L.

The constraint that both fragments need to be identified is relaxed to at least one if the other fragment is out of the mass range of the mass spectrum. The unobserved fragment in this case is assigned the same intensity as the observed one. Additionally the unobserved is not added to the queue L to avoid extension.

4.3.4 Handling more then one seed sequence

In the presented approach we assume that the degradation graph has a single seed sequence, i.e., only one peptide is processed by the proteolytic enzymes. In most cases this will be sufficient but under certain conditions two or more seed peptides could be necessary. The presented construction algorithm can be easily modified to incorporate more then one seed, by simply adding the additional seed sequences during the initialization to the queue L. Given that there is no overlap between the amino acid sequences of the seeds this will lead to multiple, non-connected degradation graphs. Only if two or more of the seed sequences share a common sub-sequence, the peptide corresponding to the shared sub-sequence is observed, and if there exists a path from the two seeds to this node, the graphs will be connected.

In the remainder of this chapter we will focus only on the case that we have a single seed but all the approaches can be easily be extended to handle also the more general case with more seed sequences.

4.4 Modeling the reaction kinetics

In the previous sections we used mass spectrometry data to generate the degradation graph. The degradation graph reflects the sequence of the individual proteolytic reactions of the overall proteolytic process. Based on the degradation graph we now want to construct a mathematical model that describes accurately the reaction rates of the individual proteolytic reactions inside the degradation graph. The parameters of this mathematical model will later on be estimated based on the intensity data collected during the construction of the degradation graph.

Classical reaction kinetics are based on the *law of mass action* which relates the rate of a reaction to the concentration of the reactants. Given the generic reaction

$$\mathrm{A} \xrightarrow{k} \mathrm{B} + \mathrm{C}, \tag{4.1}$$

the reaction rates are defined as the following system of coupled, ordinary differential equations (ODEs)

$$\frac{dA\left(t\right)}{dt} = -kA\left(t\right) \tag{4.2}$$

$$\frac{dB\left(t\right)}{dt} = kA\left(t\right) \tag{4.3}$$

$$\frac{dC\left(t\right)}{dt} = kA\left(t\right), \tag{4.4}$$

where $A\left(t\right)$, $B\left(t\right)$, and $C\left(t\right)$ are the concentrations of the reactants A, B, and C at time point t and k_a is the rate constant of the reaction. Given that multiple reactions occur in parallel the reaction rates can be combined for the individual reactants. For instance, given the following generic reactions

$$\mathrm{A} \xrightarrow{k_1} \mathrm{B} + \mathrm{C} \tag{4.5}$$

$$\mathrm{A} \xrightarrow{k_2} \mathrm{D} \tag{4.6}$$

$$\mathrm{B} \xrightarrow{k_3} \mathrm{D} \tag{4.7}$$

the reaction rates are defined as follows

$$\frac{dA\left(t\right)}{dt} = -k_1 A\left(t\right) - k_2 A\left(t\right) \tag{4.8}$$

$$\frac{dB\left(t\right)}{dt} = k_1 A\left(t\right) - k_3 B\left(t\right) \tag{4.9}$$

$$\frac{dC\left(t\right)}{dt} = k_1 A\left(t\right) \tag{4.10}$$

$$\frac{dD\left(t\right)}{dt} = k_3 B\left(t\right). \tag{4.11}$$

The law of mass action is based on the assumption that the reaction system is well stirred and all the reactants are available in a sufficient amount. This assumption does not always hold, especially in biological systems. Here alternative approaches like stochastic chemical kinetics as described by Gillespie [164] can be used. A thorough overview of modeling approaches in computational biology can be found in the reviews by Materi and Wishart [165] and Machado et al. [166].

4.4.1 Generating a kinetic model for the degradation graph

As described above we will derive the kinetic model for the degradation graph based on the law of mass action, as it was previously done by Yi et al. [36]. Each proteolytic reaction, or each edge in the degradation graph, is modeled as a first–order reaction, i.e., the speed of the reaction depends only on the concentration of one reactant. In case of proteolytic reactions, this reactant is the protein or peptide that was degraded. Side effects like saturation of degradation products are neglected, but can be incorporated by extending the ODE model accordingly. The rate equations for an exoproteolytoc reaction, where u is degraded to v, are written as follows

$$\frac{dC_u(t)}{dt} = -k_{uv}C_u(t) \tag{4.12}$$
$$\frac{dC_v(t)}{dt} = k_{uv}C_u(t), \tag{4.13}$$

where $C_u(t)$ and $C_v(t)$ denote the concentration of peptide u and v at time t. k_{uv} is the kinetic rate constant for the reaction. Endoproteolytic reactions (u degraded to v and w) are represented in the same manner. The only difference is that both degraded products need to be modeled:

$$\frac{dC_u(t)}{dt} = -k_{uvw}C_u(t) \tag{4.14}$$
$$\frac{dC_v(t)}{dt} = k_{uvw}C_u(t) \tag{4.15}$$
$$\frac{dC_w(t)}{dt} = k_{uvw}C_u(t). \tag{4.16}$$

Each reaction and reactant in the degradation graph is transformed as described above. As an example the degradation graph shown in Figure 4.2 was transformed into the following system of ordinary differential equations

$$\frac{dC_a(t)}{dt} = -k_{ab}C_a(t) - k_{acd}C_a(t) \tag{4.17}$$
$$\frac{dC_c(t)}{dt} = k_{acd}C_a(t) - k_{cf}C_c(t) \tag{4.18}$$
$$\frac{dC_e(t)}{dt} = k_{de}C_d(t) \tag{4.19}$$
$$\frac{dC_b(t)}{dt} = k_{ab}C_a(t) \tag{4.20}$$
$$\frac{dC_d(t)}{dt} = k_{acd}C_a(t) - k_{de}C_d(t) \tag{4.21}$$
$$\frac{dC_f(t)}{dt} = k_{cf}C_c(t). \tag{4.22}$$

In the here considered experiments the proteolytic process as well as the mass spectrometry measurements happen *ex vivo*. Therefore the base–peptide is assumed to have a fixed starting concentration ($C_a(0)$ in the above example) and it is not further produced. There may exist settings where this assumption does not hold. In such a situation the ODE could be extended to also model such a behavior.

4.4.2 Transforming peptide concentrations to signal intensities

The rate equations used to model the dynamics of the proteolytic processed are based on concentration values for each chemical species that participates in the reactions. The presented approach uses mass spectrometry data where only intensities associated with a specific mass-to-charge ratio are observed. Assuming that the correct charge for a given signal can always be determined, there are still several problems that need to be addressed if one wants to use the intensities to estimate the parameters of the rate equations.

The first problem is the relationship between intensity and concentration. Different studies [167, 168] have shown that a linear relationship between the concentration of a peptide and the observed signal intensity exists. To model this relationship the peptide concentrations of the ODE model are translated into intensities associated with a mass using a linear transformation.

$$\hat{I}_m(t) = f_i C_i(t), \tag{4.23}$$

where $\hat{I}_m(t)$ is the intensity associated with the mass m at time point t, f_i is a peptide specific factor, and $C_i(t)$ the concentration, computed by the model, for peptide i at time point t, with $m = m(i)$. Yi et al. [36] already applied a similar transformation successfully in their study.

This transformation implicitly solves also the second problem of comparability between two observed intensities in the same spectrum. Since each observed signal intensity will be transformed individually to the common concentration domain, the resulting concentrations can be compared afterwards. This transformation can further be used to compensate for systematic effects that occur in each measurement, e.g., quantification errors or incomplete ionization. To compensate for other, non-linear effects also alternative transformations could be used.

Another problem arises from the ambiguity of the mass, i.e., there could be several peptides with the same or a nearly identical mass. These peptides cannot be distinguished by a mass spectrometer. For instance in the degradation graph shown in Figure 4.4 the two highlighted nodes cannot be distinguished by their mass, since their amino acid composition is identical. Given a mass spectrum with a corresponding peak the construction algorithm would add both nodes to the graph. To prevent that the signal is counted twice we modify the initially proposed intensity transformation such that the intensity is the sum of the peptide concentrations with equal mass,

$$\hat{I}_m(t) = \sum_{i \in P(m)} f_i C_i(t), \tag{4.24}$$

where $P(m)$ is the set of all peptides i which have the mass m.

The last problem is the inter spectra variability of the intensities. To ensure that we can

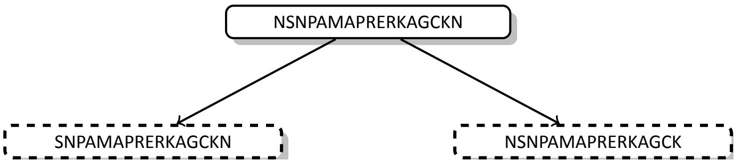


Figure 4.4: An artificial degradation graph where the two highlighted fragments cannot be distinguished by their mass, since they have the same amino acid composition.

compare the intensities between different spectra we need to normalize the intensities. Our normalization approach is based on the assumption that the sum of all intensities belonging to the proteolytic process should stay constant. Based on this we fix the total intensity to a value N[1] and distribute it over the different peaks based on their relative intensities.

$$I'_m(t) = N \frac{I_m(t)}{\sum_{j \in M} I_j(t)}, \tag{4.25}$$

where $I'_m(t)$ is the normalized intensity of mass m at time point t, $I_m(t)$ the observed intensity of mass m at time point t, and M the set of all peptide masses belonging to the proteolytic process.

4.4.3 Estimating kinetic rate constants

After generating a dynamical system for the degradation graph and a transformation to the intensity domain, the next task is to find suitable values for the rate constants and the transformation parameters such that the dynamical evolution of the intensity values predicted by the model agrees with the observed intensities. This task is in general known as parameter estimation or parameter identification. Following standard practice in the field of parameter estimation we want to solve the weighted least squares minimization problem,

$$\min \sum_{m \in M} \left(\sum_{i=0}^{N} \left(\frac{\left(\hat{I}_m(t_i) - I_m(t_i)\right)^2}{w(m,i)} \right) \right), \tag{4.26}$$

where M is the set of all observed masses, $I_m(t_i)$ is the intensity observed for mass m at time point t_i, $\hat{I}_m(t_i)$ is intensity predicted by ODE system for the mass m at time point t_i, and w is a weighting function. The weighting function can for instance be used to use relative instead of absolute deviations, i.e.,

[1] In the later parts of this thesis we used $N = 10.000$.

$$w\left(m,i\right)=I_m\left(t_i\right). \tag{4.27}$$

By using such a weighting function the effect of different intensities being on different orders of magnitude can be reduced.

In theory the given minimization problem can be solved by many available optimization techniques. After testing different freely available approaches we decided to use POEM to estimate the model parameters as well as the transformation parameters and the initial concentration of the base–peptide. POEM is a Matlab®-based version of BioPARKIN [169, 170] and is based on the damped Gauss-Newton method.

While POEM performs well in our experiments (see Chapter 5) one need to keep in mind that the applied method is a local optimization approach, hence it will only return a local minimum and it cannot be guaranteed or even determined if the given result corresponds to the global minimum. For small instances of the degradation graph this problem could be circumvented by brute-force or simulated annealing based approaches. But with increasing size of the degradation graph, and with this an increasing amount of reactions connecting the nodes inside the graph, the number of parameters can easily increase to 40 or more. For such instances brute-force or simulated annealing approaches are infeasible with respect to required computation time.

A second problem to consider carefully is the choice of the initial parameters for the estimation. The initial parameters can heavily influence the outcome of the Gauss-Newton method, thus given an inappropriate choice of initial values the returned local minimum can be far away from the global minimum.

Finally, convergence of the Gauss-Newton method can not be guaranteed. Again, if the initial parameters are poorly chosen or due to insufficient sampling or an incorrect model the Gauss-Newton method is not guaranteed to converge to a local minimum. For such scenarios the implementation included in POEM applies sophisticated termination criteria (see Dierkes et al. [169] for details).

In summary, even if the Gauss-Newton method converges and returns a reasonable estimate for the parameters, a certain degree of uncertainty remains due to the fact that the result is in many cases only a local minimum. The problem, that only local minima are found, is especially relevant in the context of model discrimination. In Section 4.5 we will rank different degradation graphs and the corresponding models against each other and thus it should be kept in mind that it can occur that given two models A and B the returned local minimum for model A is superior to the one of model B despite the global minimum of model B is superior to the one of model A. To mitigate such problems we extended the criteria for the ranking of the individual degradation graphs (see Section 4.5.1).

A more in-depth analysis of the problems associated to the Gauss-Newton method and possible approaches to overcome or at least mitigate them, as they are applied for instance in POEM [169], is beyond the scope of this thesis. The interested reader is referred to Deuflhard [170] for a detailed description of the Gauss-Newton method and Dierkes et al. [169] for more information on POEM respectively its successor BioPARKIN. Bock et al. [171] intensively discusses the robustness of parameter estimation methods which also should be considered in this context.

How to choose initial values

As prior knowledge on the modeled system is very limited, good initial values for the estimation of the model parameters are hard to find. We therefore choose the initial values based on the following scheme: For each node the edge (i.e., proteolytic reaction) is selected, which leads on the shortest path to the root node. For the corresponding rate constant (k_i) we assign an initial value of 1.0. For all other incoming reactions the initial value is set to a value of 10^{-6}. All transformation parameters (f_i) are set to 1.0.

4.5 Evaluation and optimization of the degradation graph structure

Based on the above presented construction algorithm the degradation graph is maximal, i.e., it contains every possible signal that could originate from a fragment of the base–peptide and every possible reaction connecting two fragments. This assumption is not necessarily true in all cases. Given the set of nodes in Figure 4.5, the two dashed edges represent reactions that could have happened but are not necessary to explain all observed fragments. It can also happen that a node is added to the graph based on a signal with a mass equal or nearly equal to a possible fragment of the base–peptide but with no relation to the process. We will call such peptides *decoy* peptides. In such a case the degradation graph contains a node and with it connected edges that do not participate in the proteolytic process which we want to model. To remove such nodes and edges the following section describes a method to rank different subgraphs of the initially constructed degradation graph with respect to their ability to explain the observed data. Further a heuristic is presented to construct a reasonable subset of subgraphs based on the initial degradation graph and the described ranking approach.

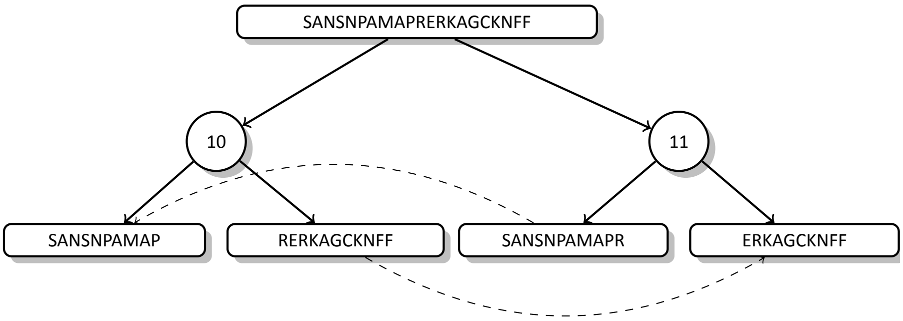


Figure 4.5: An artificial degradation graph with two edges (dashed) that could be possible but not necessary to explain the two resulting peptides.

4.5.1 Evaluating different models

To compute the degradation graph that explains the observed data best, it is necessary to first define a measure to rank the different possible solutions.

To ease the following explanations we will introduce some further notation. Given a degradation graph G, a subgraph G' is defined as $G' = (V', E')$, where $V' \subseteq V$ and $E' \subseteq E$. We also require that G' is connected, i.e. for all pairs of nodes $u, v \in V'$ exists a path of length n in E'^n that connects u and v. The subgraph G' also defines $M' \subseteq M$ as the subset of all masses m and their associated intensities that are explained by the subgraph $M' = \{m(v), v \in V'\}$.

The proposed ranking method is based on two individual components. The first component, S_C, is the average Pearson correlation of the intensities predicted by the model (with estimated reaction parameters) and the actual observed data. This component is used to represent the goodness of fit between the computed model including the estimated parameters and the measured data. For each mass $m \in M'$ that is still explained by the subgraph the Pearson correlation r_m between the predicted intensity values $\hat{I}_m$ and the observed ones I_m is computed.

$$r_m = \frac{\frac{1}{N}\sum_{i=1}^{N}\left(\left(\hat{I}_m(t_i) - \bar{\hat{I}}\right)\left(I_m(t_i) - \bar{I}\right)\right)}{\sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{I}_m(t_i) - \bar{\hat{I}}\right)^2 \frac{1}{N}\sum_{i=1}^{N}\left(I_m(t_i) - \bar{I}\right)^2}} \tag{4.28}$$

$$\bar{\hat{I}} = \frac{1}{N}\sum_{i=1}^{N}\hat{I}_m(t_i) \tag{4.29}$$

$$\bar{I} = \frac{1}{N}\sum_{i=1}^{N}I_m(t_i). \tag{4.30}$$

Afterwards the mean of all Pearson correlations is used as the measure S_C of the goodness of fit.

$$S_C = \frac{1}{|M'|} \sum_{m \in M'} r_m. \tag{4.31}$$

The second score component, S_V, is the conserved part of the standard deviation of the measured signal intensities of the original degradation graph that is conserved in the subgraph.

$$S_V = \frac{\sum_{m \in M'} s_m}{\sum_{m \in M} s_m}, \tag{4.32}$$

where s_m is the standard deviation of the signal corresponding to the mass m. Under the assumption that the modeled proteolyic process changes the concentrations of all involved peptides, S_V models the ability of the subgraph to explain the variability of the initially collected signal intensities.

To compute a single score S from the two components S_C and S_V the weighted sum of both scores is computed:

$$S = w_C S_C + w_V S_V \tag{4.33}$$

Choosing appropriate weights w_C and w_V is always dependent on the proteolytic process that should be modeled and the quality of the measured data. For instance in cases where the observed time series is not well suited to estimate reliable rate constants and transformation parameters more emphasis should be put on the variability component and vice versa. In several simulation studies the combination of $w_C = 0.9$ and $w_V = 0.1$ showed the best separation of the correctly and wrongly identified models.

An alternative, non-weighted scoring approach

The performance of the above presented scoring scheme is highly determined by the choice of w_C and w_V. To overcome this problem we present an alternative scoring approach, that is independent of the choice of any weighting factors. This approach combines both components of the first score in a single value. It is the sum of squared residuals between the estimated model and the observed values, weighted by the controlled variability of the signals,

$$S_R = \sum_{m \in M'} \left(\frac{1}{CV_m} \sum_{i=0}^{N} \left(\frac{\left(\hat{I}_m(t_i) - I_m(t_i)\right)^2}{\max\left(I_m(t_i), T\right)} \right) \right), \tag{4.34}$$

where CV_m is the controlled variability of the signals observed for mass m

$$CV_m = \frac{\sigma_m}{|\mu_m|}. \tag{4.35}$$

While having the advantage that we do not need to optimize the weighting parameters w_C and w_V, we obviously lose the advantage that S_C and S_V are both normalized ($0 \leq S \leq 1$), which eases the interpretation and aids in comparing different degradation graphs.

4.5.2 Heuristic search for the optimal degradation graph

The task to generate all possible subgraphs, construct the associated dynamic system, and estimate the corresponding rate constants and transformation parameters is feasible for small degradation graphs, but with increasing size in terms of nodes and reactions, especially the parameter estimation gets more and more computationally intensive. Generating all possible combinations of reactions would result in $2^{|E|}$ subgraphs. Even if only the feasible combinations would be considered (i.e., those that contain the base–peptide and are connected) the number would still grow exponentially with the amount of initially identified reactions. For each of these subgraphs we would need to generate the corresponding ODE system, estimate the rate constants and transformation parameters, and compute the rank.

To accelerate the search for the optimal degradation graph we designed a heuristic approach that speeds up the search. Preliminary experiments on simulated data have shown that the above presented graph score improves if the structure of the degradation graph gets closer to the original one. This can also be explained based on the composition of the score. The first component, which reflects the goodness of fit between observation and prediction, should improve constantly if components that do not participate in the observed proteolytic process are removed. The second component, the conserved signal variability, will decrease only drastically if we remove nodes that vary more over time then the regular signal variation due to the measurement. These nodes should be the ones that participate in the proteolytic process. Nodes that do not participate should have a much lower variability and therefore will not decrease the score that much if they are removed from the degradation graph.

Due to the method used for the construction of the degradation graph it can be assumed that the graph is maximal in the sense that it contains all signals originating from the proteolytic process that should be modeled. It further contains all possible combinations of reactions connecting these signals. And, as stated earlier, it will maybe contain peptides and reactions that did not happen or do not belong to the process. To now find the optimal subgraph we start by removing all terminal reactions of the graph (i.e., reactions that produce at least one leaf) separately. We call this *leaf pruning*. For each individual subgraph we estimate the rate constants and transformation parameters and subsequently rate the subgraph based on the above presented criteria. We now take the best N subgraphs that were already *leaf pruned*, remove again the terminal reactions, and again compute parameters and score. We repeat this step as long as we can find at least one subgraph that was not pruned in a previous iteration and is under the top N scoring graphs. The pseudo code for the above presented approach is shown in Figure 4.6.

Using this heuristic it was possible to drastically reduce the amount of necessary parameter estimations to identify the optimal subgraph.

Preliminary tests on simulated data have shown that setting N to either 2 or 3 is sufficient. Using these settings allows us to effectively bound the number of parameter estimation runs while still finding the original degradation graph.

4.6 Running time considerations

In cases where the initially constructed degradation graph is large the above presented approach of degradation graph construction, parameter estimation, and structure optimization (summarized in Figure 4.7) is computationally quite intensive. Therefore an estimate of the running time in the worst case is given in this section.

The complexity of the initial degradation graph construction is determined by the number of required identification look ups in the mass spectrum. In the worst case every possible degradation product would need to be verified. The resulting degradation graph would contain every possible substring of the base–peptide. So the number of verifications is equal to the number of possible substrings which is, for a base–peptide of length n, $\frac{n(n-1)}{2}$. Given N time points, the maximum number of verifications is bound by $N\left(\frac{n(n-1)}{2}\right)$.

The complexity of the parameter estimation is dependent on the number of unknown parameters and the number of measurement points and can be approximated by $2N\,|E|^3$. N is the number of time points, i.e., the number of acquired mass spectra, and $|E|$ the number of proteolytic reactions, i.e., the number of edges in the graph that connect non-pseudo nodes. So the time required for the parameter estimation will decrease with the complexity of the degradation graph but will still require a considerable amount of time. The total number of

```
function Optimize(degradation graph $G, N$)
    Estimate(G)
    $S \leftarrow \{G\}$                         → $S$ is a List, sorted in descending order by the score
    $T \leftarrow \{\}$                                    → $T$ is the list of already trimmed graphs
    while $BestN \leftarrow$ FindBestNGraphs(S,T,N), $size\ (BestN) \neq 0$ do
        for each degradation graph $H \in BestN$ do
            $trimmed \leftarrow$ Trim($H$)                            → create list of subgraphs
            $T \leftarrow \{T, H\}$                                    → mark $H$ as processed
            for each degradation graph $K \in trimmed$ do
                Estimate(K)
                $S \leftarrow \{S, K\}$               → append estimated graph $K$ to the sorted list
            end for
        end for
    end while
end function

function FindBestNGraphs(Sorted List $S$, List $T$, $N$)
    $BestN \leftarrow \{\}$
    for each degradation graph $G \in S$ do
        if $G \notin T$ then                  → use only graphs that were not already trimmed
            $BestN \leftarrow \{BestN, g\}$
        end if
        if $size\ (BestN) = N$ then                               → select at most $N$ graphs
            break
        end if
    end for
    return $BestN$
end function

function Trim(degradation graph G)
    $trimmed \leftarrow \{\}$
    for each Leaf $l$ of $G$ do
        $trimmed \leftarrow \{trimmed, G \setminus l\}$                → remove only leaf $l$ from the graph
    end for
    return $trimmed$
end function
```

Figure 4.6: Pseudo code for the degradation graph optimization heuristic. The score used to sort the list S is the score described in Section 4.5.1. The function Estimate represents the parameter estimation procedure for the associated ODE model (see Section 4.4).

these optimizations is still $\approx 2^{|E|}$ in the worst case, since the heuristic can not guarantee an earlier termination in all cases.

4.7 Implementation

The complete approach, as it is summarized in Figure 4.7, was implemented as a small Java library. The degradation graph construction and the structure optimization were additionally integrated into the proteomics.net platform [172]. This allows an easy distribution of the independent parameter estimation steps, during the structure optimization, on different machines inside a local network. Chapter 6 gives an introduction into the proteomics.net platform and describes the extensions, implemented for this thesis, that were necessary to support our approach.

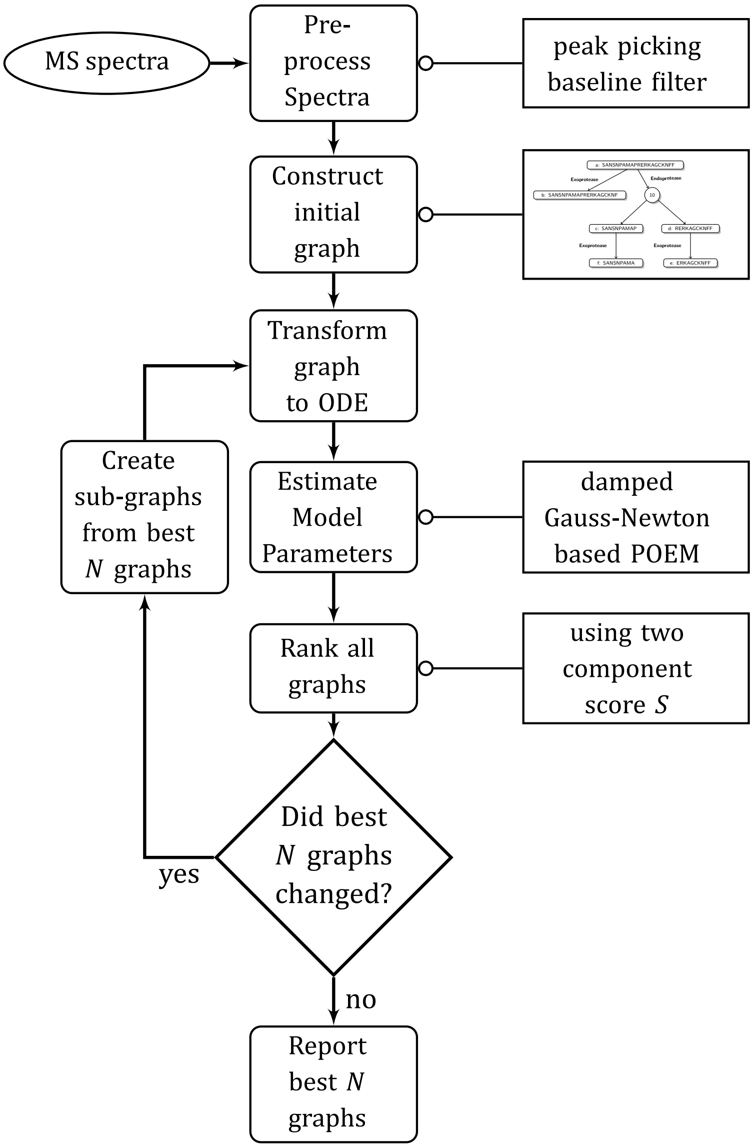


Figure 4.7: Flowchart of the here presented approach to construct and optimize degradation graphs based on mass spectrometry time series data. Each of the operations shown will be explained individually in the following sections.

Chapter 5 Validation of degradation graphs using simulated and real data

The aim of the following chapter is to benchmark the approach presented in Chapter 4. To assess the performance of the approach we need to validate two different aspects:

Structure Does the reconstructed degradation graph reflect the real series of proteolytic reactions? Is the heuristic able to remove all nodes and edges that were added due to misleading signals to the degradation graph?

Dynamics Are the estimated rate constants correct? Can the approach handle the noise in mass spectrometry measurements and still estimate reasonable parameters of the ODE?

All this has to be tested under varying conditions like the complexity of the degradation graph or increasing noise in the mass spectrometry data. Since data with such a variety of parameters like noise and varying complexity of the degradation graph is not available, a part of the validation will be carried out on data simulated with MSSimulator (see Chapter 3), followed by the analysis of an incubation experiment where a fragment of beta-2-microglobulin was incubated with a mixture of immobilized urine proteins.

5.1 Simulation study

As stated earlier, data availability is a general problem, especially for monitoring complex reactions like proteolysis. Therefore four different simulated mass spectrometry datasets were generated using MSSimulator (Chapter 3 and [41]). We will first explain the general approach for the simulation as well as the analysis of the simulated data. Subsequently the four datasets and the analysis results will be explained in detail. The first one is the degradation of fibrinopeptide A as presented in [36]. The proteolytic process was simulated several times with varying noise settings to show the effect of noise on the results of the degradation graph approach. The last three datasets are artificially constructed proteolytic systems, that we use in order to show the performance of the method on more complex proteolytic systems.

5.1.1 Experimental setup

As stated earlier (see Section 3.3) MSSimulator needs at least two input files to generate the desired mass spectra.

The first one is the general configuration file containing all the information on noise, resolution, ionization technique, etc. For these experiments we chose MALDI ionization, disabled the liquid chromatography mode, and disabled the digestion, since we are already dealing with peptides. MSSimulator provides different types of noise (e.g., detector noise, shot noise). For the analysis one type of noise is especially relevant, the general signal variability. It is an intensity dependent deviation of the signal intensity of a ion species, e.g., if we set the intensity noise value to 10 %, the total signal intensity (area under the curve) of the simulated peak will vary with a standard deviation of 10 % of the original signal intensity. We will refer to this kind of noise as *signal variability*.

The second input file is the FASTA file containing the peptide sequences annotated with their individual abundances. For each time point a FASTA file including the abundance values was generated based on the ODE system to be simulated. For the ODE system the initial value of the abundance of the base peptide was set to 10,000.0, for all other peptides the abundance was set to 0.0. All the decoy peptides that were used to generate wrong identifications have constant, randomly assigned abundances that are in the same magnitude as the other peptides.

Following the simulation all generated mass spectra were post-processed. The OpenMS PeakPickerWavelet [173] was applied to all spectra to convert the raw into centroided data. The identification of peaks is done using the peptide mass fingerprinting approach that is directly implemented into the degradation graph construction tool (see Section 4.3.1).

5.1.2 Simulation study 1: Validation using the *ex vivo* degradation of fibrinopeptide A (FPA)

The first simulation study was carried out to show that the approach presented in Chapter 4 is able to fully recover the structure of the degradation graph and the corresponding reaction rates for the proteolytic reactions. To achieve this a data set based on the fibrinopeptide A[1] (FPA) degradation (as it was described by Yi et al. [36]) was simulated and analyzed. The proteolytic process consists of a series of exoproteolytic cleavages at the N-terminus of FPA. The corresponding degradation graph is shown in Figure 5.1. The reaction rates were also, with minor modifications, taken from Yi et al. [36]. The modified parameters are also shown in Figure 5.1.

[1] Swiss-Prot:P02671[20-35]

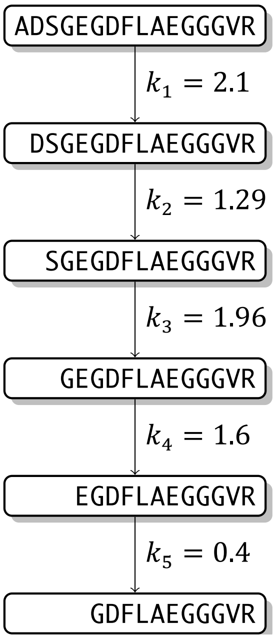


Figure 5.1: Degradation graph of the sequential degradation of fibrinopeptide A (FPA) as reported in [36]. The rate constants of the individual reactions, used in the experiments, are shown as annotations beside the corresponding edges.

The whole proteolyitc system, given the above described initial values, was simulated over a time span of 5 hours. 10 sampling points were generated for the time series, five during the first hour of the incubation and five distributed equally over the remaining 4 hours. For each of the time points five different mass spectra were generated with increasing signal variability. The signal variability values were set to 5, 10, 20, 30, and 40 % of the original signal intensity. The impact of the signal variability on the time course of the peptide intensities is shown in Figure 5.2. Subsequently the generated mass spectra were preprocessed (as described above) and analyzed using the presented method to estimate the degradation graph structure as well as the the rate constants.

Our approach succeeded in reconstructing the originally simulated degradation graph (see Figure 5.1) for all simulated noise levels. Figure 5.3 shows the effect of the signal variability on the score presented in Section 4.5.1. One can clearly see that the score decreases with increasing noise. Figure 5.4 shows the effect of the signal variability on the quality of the parameter estimation. Again one can see that with increasing signal variability the quality of the estimated parameters decreases but stays in a reasonable range if the signal variability is below 30 %.

The simulations have shown that the presented approach is able to recover the structure as well as the reaction rates even in the presence of extensive noise. The estimated reaction

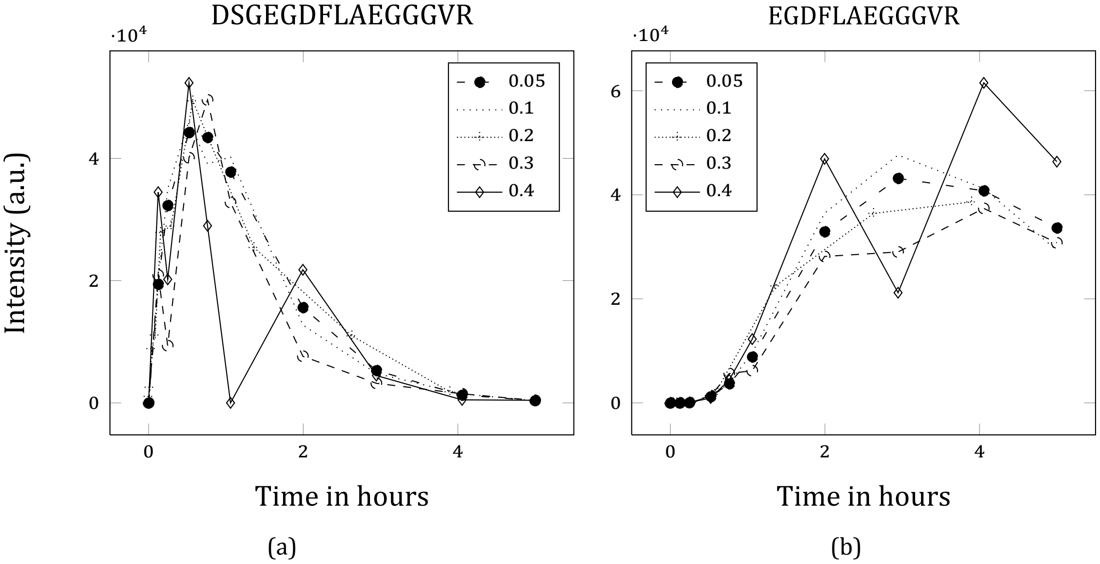


Figure 5.2: Effect of the different signal variability settings on the simulated signal intensities. Shown are the extracted signal intensities for two peptides (a) DSGEGDFLAEGGGVR (left) and (b) EGDFLAEGGGVR (right) of the fibrinopeptide A system shown in Figure 5.1 with increasing signal variability values.

rates have an acceptable agreement with the originally simulated parameters. The problem of decreasing performance of the parameter estimation for signal variability values ≥ 30 % could possibly be mitigated by increasing the number of sampling points, especially in time spans, where the system changes the most.

5.1.3 Simulation study 2: Complex degradation of human plasma peptides

The first simulation study has shown the performance of the degradation graph approach under varying measurement conditions. In the second simulation study we will investigate the ability of the approach to handle complex proteolytic reactions where also endoproteolytic reactions occur. Therefore a test set of three different human plasma peptides (and peptide fragments) degraded by multiple artificial endo- and exoproteases was generated. The targeted peptides were fragments of endothelin 1[2], angiotensin[3], and somatostatin-28[4]. All reactions and peptide fragments are shown in Figures 5.5 to 5.7. Since all the reactions in the degradation graph were artificial, we needed to define also the reactions rates by hand. All reaction rates were chosen in the same range as the experimentally confirmed parameters

[2] Swiss-Prot:P05305[53-73]

[3] Swiss-Prot:P01019[34-43]

[4] Swiss-Prot:P61278[89-116]

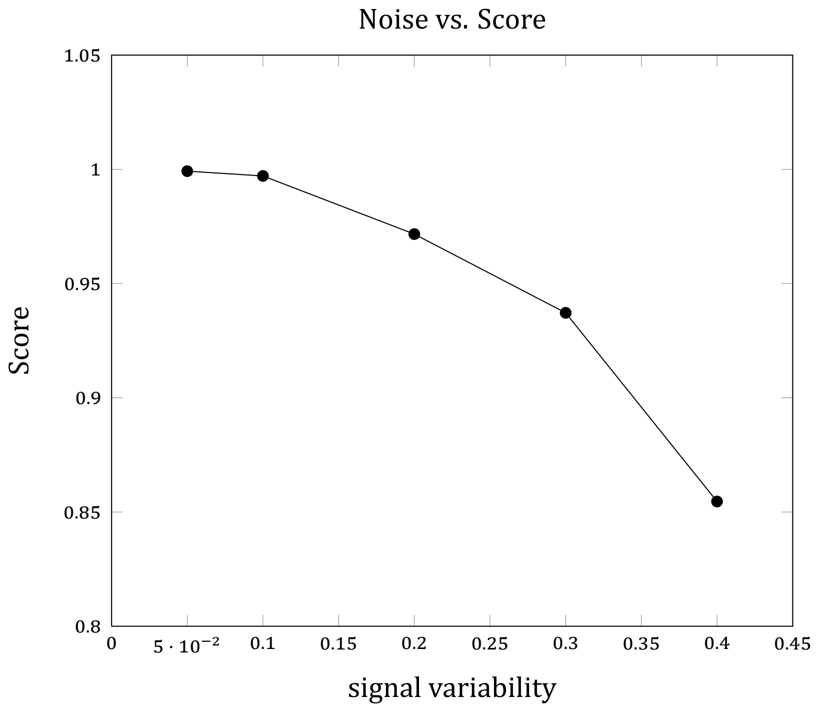


Figure 5.3: Effect of the variability of the signal with respect to the intensity on the score S computed by our method. Data was generated based on the fibrinopeptide A system shown in Figure 5.1.

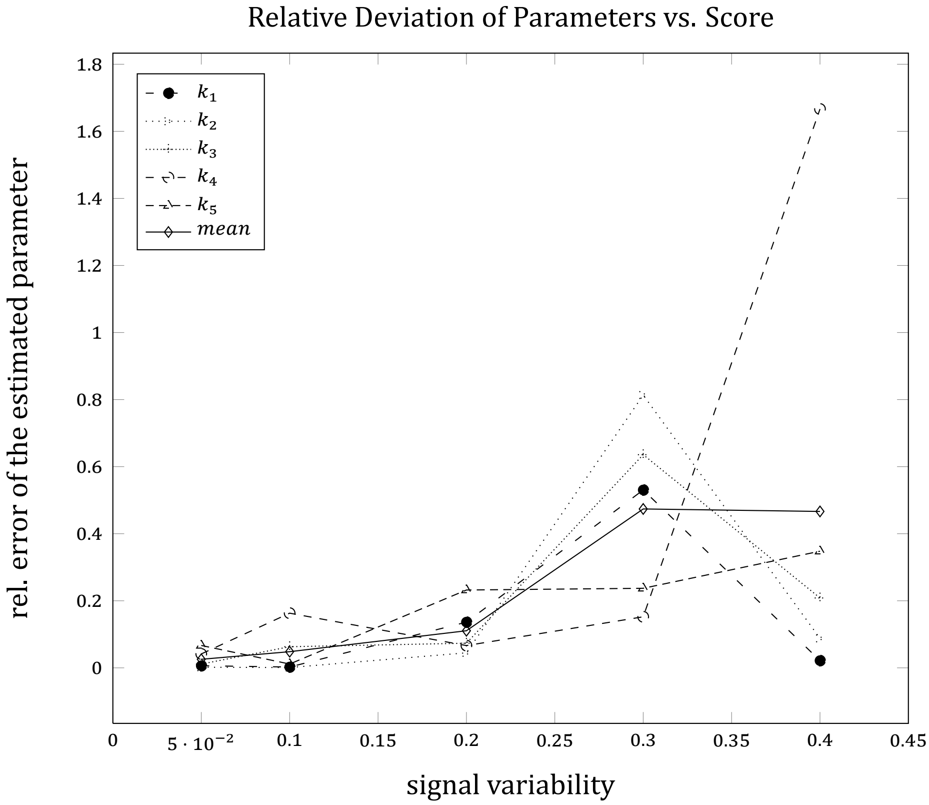


Figure 5.4: Effect of the variability of the signal with respect to the intensity on the quality of the estimated reaction parameters. The quality is given in terms of the relative deviation of the estimated from the real parameter ($\frac{|p_{real}-p_{est}|}{p_{real}}$). Data was generated based on the fibrinopeptide A system shown in Figure 5.1. The reaction parameters are numbered in the order of degradation (e.g., FPA → FPA-1 = k_1) shown in Figure 5.1. The parameters are numbered in the order of degradation (e.g., FPA → FPA-1 = k_1).

of the FPA degradation and are shown in Tables 5.1 to 5.3 (column p_{real}).

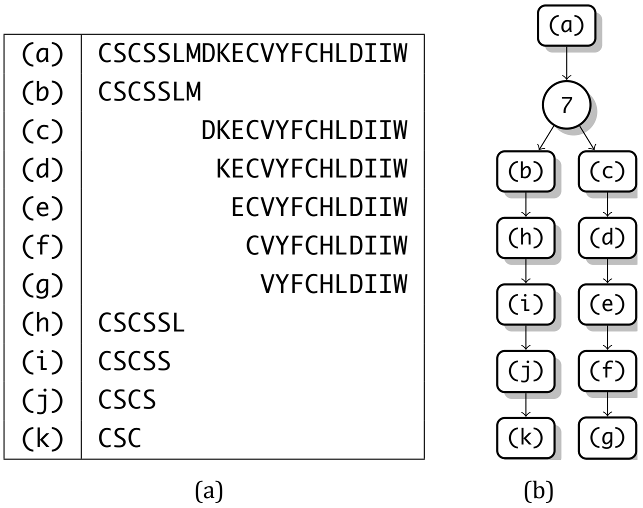
The three different systems were simulated again over a time span of 5 hours. 15 mass spectra were simulated for each of the three systems. 6 of these mass spectra were generated during the first hour of the three time series, since during this time the systems changed the most. The remaining mass spectra were simulated at equally distributed time points over the remaining 4 hours. All mass spectra were generated with a fixed signal variability of 20 %. During the mass spectrometry simulation decoy peptides were added to the mass spectra. Each of them had a mass similar to a possible fragment of one of the nodes in the degradation graph. To account for this during the analysis, we applied the method proposed in Section 4.5.2 to iteratively optimize the structure of the degradation graph.

For all three systems the degradation graph approach was able to reconstruct the original system based on the simulated data. The degradation graph with the highest rank based on the proposed score was the originally simulated one. In case of the angiotensin degradation graph the original fragment (e) was misinterpreted as IHPFH. The N- and C-terminal amino acids of its predecessor IHPFHL (Leucin and Isoleucin) have equal masses and therefore cannot be distinguished in the mass spectrometer, hence both solutions are equally good.

Tables 5.1 to 5.3 show the results of the parameter estimation for the model with the highest rank in comparison to the parameters used for the simulation. Just like in the first study, the estimated parameters show a reasonable agreement with the one used for the simulation. In average the relative deviation of the estimated from the real parameters is between 10 and 20 % in all three experiments. Figure 5.8 shows the extracted intensities of two characteristic somatostatin-28 fragments compared with predicted model intensities. As one can see, the predicted model intensities and the simulated intensities have a good agreement in their dynamic behavior. It can be observed that the largest errors occur towards the end of the degradation process (e.g., k_{ef} for the somatostatin 28 system). This could be explained with the lower amount of generated mass spectra for the corresponding reactants which gives us less information for an accurate parameter estimation. The problem could possibly be solved by an extension of the experiment beyond the 5 hours or an increased amount of samples during the last hours.

5.2 Evaluation on real data

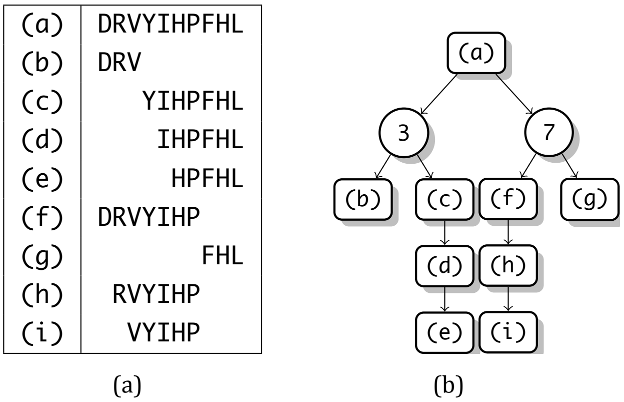
In the previous sections we have shown, using simulated data, that the degradation graph approach works. In the following section we will use a real dataset to show also its applicability to experimental data measured on a real mass spectrometer. The dataset consists of



(a)	CSCSSLMDKECVYFCHLDIIW
(b)	CSCSSLM
(c)	DKECVYFCHLDIIW
(d)	KECVYFCHLDIIW
(e)	ECVYFCHLDIIW
(f)	CVYFCHLDIIW
(g)	VYFCHLDIIW
(h)	CSCSSL
(i)	CSCSS
(j)	CSCS
(k)	CSC

(a) (b)

Figure 5.5: Degradation of endothelin-1 by multiple artificial endo- and exoproteases. (a) The mapping of indices to sequences. (b) The degradation graph.



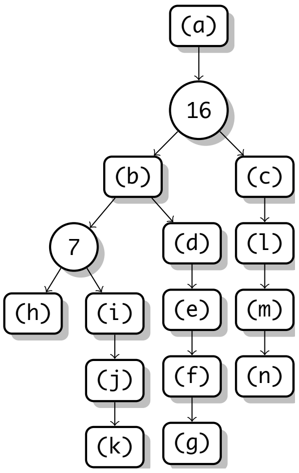
(a)	DRVYIHPFHL
(b)	DRV
(c)	YIHPFHL
(d)	IHPFHL
(e)	HPFHL
(f)	DRVYIHP
(g)	FHL
(h)	RVYIHP
(i)	VYIHP

(a) (b)

Figure 5.6: Degradation of angiotensin by multiple artificial endo- and exoproteases. (a) The mapping of indices to sequences. (b) The degradation graph.

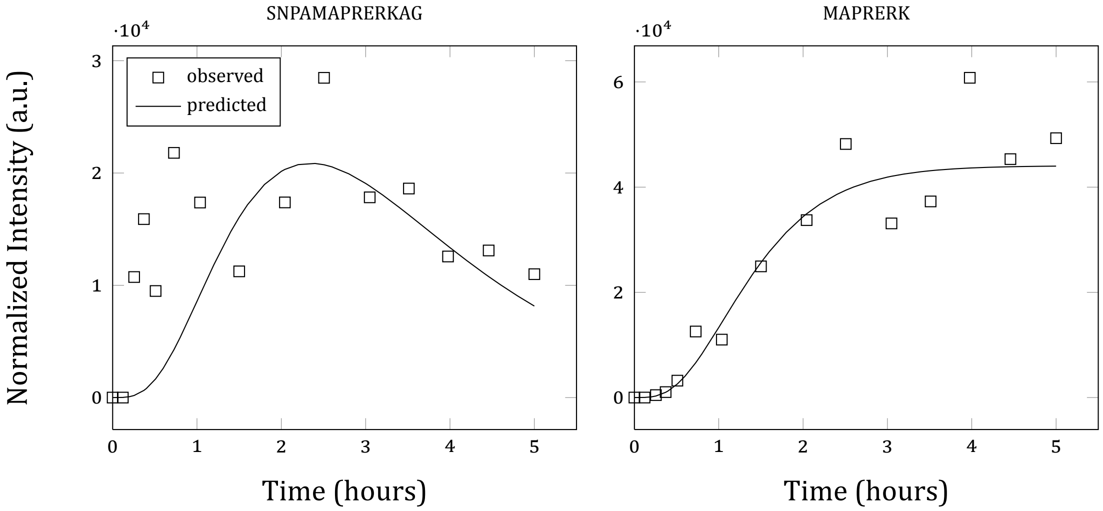
(a)	SANSNPAMAPRERKAGCKNFFWKTFTSC
(b)	SANSNPAMAPRERKAG
(c)	CKNFFWKTFTSC
(d)	ANSNPAMAPRERKAG
(e)	NSNPAMAPRERKAG
(f)	SNPAMAPRERKAG
(g)	NPAMAPRERKAG
(h)	SANSNPA
(i)	MAPRERKAG
(j)	MAPRERKA
(k)	MAPRERK
(l)	CKNFFWKTFTS
(m)	CKNFFWKTFT
(n)	CKNFFWKTF

(a)



(b)

Figure 5.7: Degradation of somatostatin-28 by multiple artificial endo- and exoproteases. (a) The mapping of indices to sequences. (b) The degradation graph.



(a) (b)

Figure 5.8: Shown is the intensity course of two peptide fragments compared with the predicted model intensities for the best somatostatin-28 degradation graph.

Parameter	p_{real}	p_{est}	$\lvert p_{real} - p_{est} \rvert$	$\frac{\lvert p_{real} - p_{est} \rvert}{p_{real}}$
k_{jk}	1.30	0.949	0.351	0.270
k_{ij}	1.90	2.496	0.596	0.314
k_{hi}	2.10	2.369	0.269	0.128
k_{bh}	1.05	0.955	0.095	0.091
k_{abc}	3.50	5.025	1.525	0.436
k_{fg}	2.30	1.351	0.949	0.414
k_{cd}	4.30	4.284	0.016	0.004
k_{ef}	0.30	0.380	0.080	0.265
k_{de}	2.10	2.015	0.085	0.040

Table 5.1: Relative and absolute deviations of the estimated parameter values for the endothelin 1 system. The indices for the parameter names are taken from Figure 5.5. p_{real} denotes the parameter values used for the initial simulation and p_{est} the value estimated by the presented approach. The last two columns contain the absolute and the relative deviation of the estimated from the real parameter value.

Parameter	p_{real}	p_{est}	$\lvert p_{real} - p_{est} \rvert$	$\frac{\lvert p_{real} - p_{est} \rvert}{p_{real}}$
k_{fh}	0.50	0.498	0.002	0.004
k_{abc}	3.20	3.733	0.533	0.167
k_{afg}	1.80	2.226	0.426	0.236
k_{de}	1.05	1.111	0.061	0.058
k_{hi}	1.30	1.225	0.076	0.058
k_{cd}	1.50	1.320	0.180	0.120

Table 5.2: Relative and absolute deviations of the estimated parameter values for the angiotensin system. The indices for the parameter names are taken from Figure 5.6. p_{real} denotes the parameter values used for the initial simulation and p_{est} the value estimated by the presented approach. The last two columns contain the absolute and the relative deviation of the estimated from the real parameter value.

Parameter	p_{real}	p_{est}	$\lvert p_{real} - p_{est} \rvert$	$\frac{\lvert p_{real} - p_{est} \rvert}{p_{real}}$
k_{de}	0.70	0.719	0.019	0.027
k_{mn}	2.80	3.125	0.325	0.116
k_{lm}	1.20	1.145	0.055	0.046
k_{cl}	3.10	3.312	0.212	0.068
k_{jk}	2.40	1.998	0.402	0.167
k_{ij}	1.60	1.951	0.351	0.219
k_{ef}	1.24	2.032	0.792	0.639
k_{bd}	3.20	2.648	0.552	0.172
k_{bhi}	3.40	2.158	1.242	0.365
k_{abc}	4.3	3.760	0.540	0.126
k_{fg}	2.54	0.940	1.600	0.630

Table 5.3: Relative and absolute deviations of the estimated parameter values for the somatostatin 28 system. The indices for the parameter names are taken from Figure 5.7. p_{real} denotes the parameter values used for the initial simulation and p_{est} the value estimated by the presented approach. The last two columns contain the absolute and the relative deviation of the estimated from the real parameter value.

a time series measurement where a fragment of beta-2-microglobulin[5] was incubated with immobilized urine proteins.

5.2.1 Experimental setup

For the immobilization of urine proteins from haemolytic urine of renal transplantation patients CNBr-activated Sepharosebeads® 6MB were used. The Sepharosebeads® were incubated in 0.1 M hydrochloric acid (HCl) on a mixer (Horizontal Shaker, Rotator Drive STR4 Stuart Scientific, Redhill, England) for 30 minutes and washed with HPLC-grade water. The immobilization of urine proteins onto the Sepharosebeads® was done in coupling-buffer (100 mM $NaHCO_3$, 500 mM $NaCl$, pH 8.3) during an incubation period of 2 hours on a mixer. Per preparation 50 µl urine and 30 µl Sepharosebeads® were used. After immobilization the Sepharosebeads® were washed with HPCL-grade water. Free binding capacities were saturated by over night incubation at 4 °C in blocking-buffer (100 mM $NaHCO_3$, 500 mM $NaCl$, 0.2 M Glycin, pH 8.3). Afterwards the blocking-buffer was removed by washing with HPLC-grade water repeatedly.

The incubation of immobilized urine proteins took place in sodium acetate buffer at pH 4.9 and was started by addition of the beta-2-microglobulin fragment to the immobilized proteins with a final concentration of 10^{-4}M in a reaction volume of 50 µl. At nine distinct time

[5] Swiss-Prot:P61769[77-97]

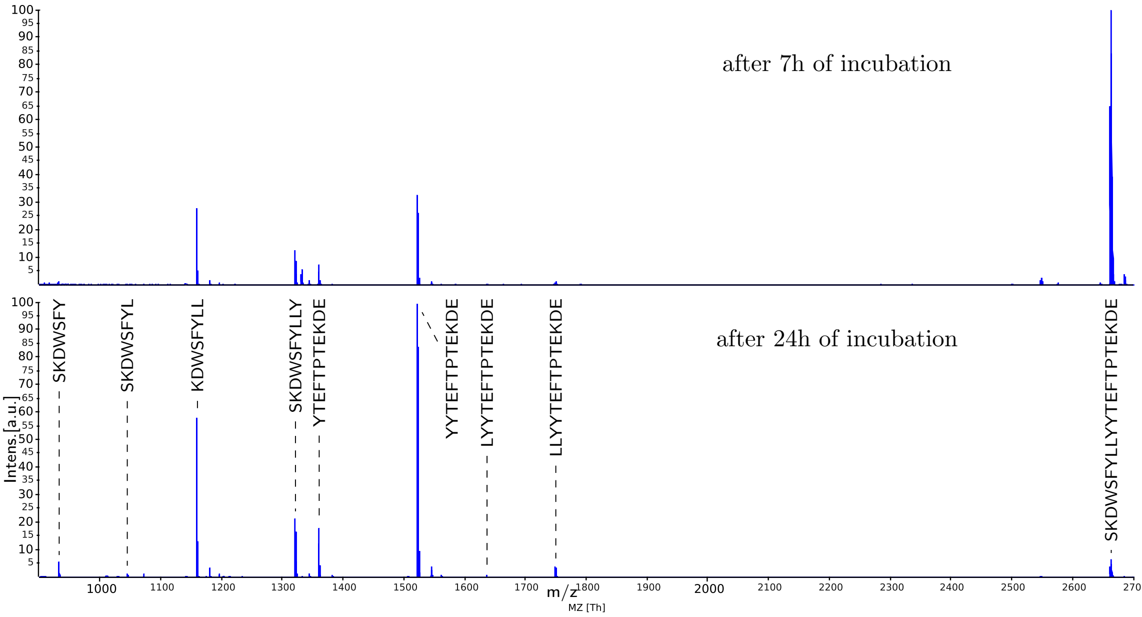


Figure 5.9: Mass spectra generated during the degradation of a fragment of beta-2-microglobulin by a mixture of urine proteins at time point $t_8 = 7$ h (upper) and $t_9 = 24$ h (lower) of incubation. Intensity is given in percent of maximal peak intensity. Horizontal axis is given in Th. In the lower spectra all fragments that could be verified by tandem MS identifications were annotated.

points aliquots were taken from the reaction mixture and diluted in a ratio of 1:10 in 0.2 % (v/v) formic acid (Fluka/ Sigma-Aldrich, Steinheim, Germany) for MALDI-TOF/TOF analysis on a 4700 Proteomics Analyzer (Applied Biosystems).

The distinct time points were $t_1 = 0$ min, $t_2 = 10$ min, $t_3 = 20$ min, and $t_4 = 30$ min, and $t_5 = 1$ h, $t_6 = 2$ h, $t_7 = 4$ h, $t_8 = 7$ h, and $t_9 = 24$ h. Manual inspection of the data lead to the assumption that at least four different endoproteolytic cuts at positions 7—10 occurred during the incubation. These cuts respectively the resulting fragments could be validated using tandem MS spectra. The main reaction of the proteolytic process happened between the time points $t_8 = 7$ h and $t_9 = 24$ h. Figure 5.9 shows the mass spectra for both time points. In the mass spectrum for $t_9 = 24$ h the peaks for the eight validated fragments generated by the four assumed endoproteolytic reactions as well as the base peptide are annotated.

5.2.2 Analysis using the degradation graph approach

All mass spectra were again preprocessed by the OpenMS PeakPicker. The preprocessed mass spectra were subsequently analyzed using the degradation graph approach. The acquired mass spectra had a high enough resolution to resolve single isotopic peaks. To account for this we applied the high resolution version of the peptide mass fingerprinting approach, as it was described in Section 4.3.1. We further applied the inter-spectrum normalization described in Section 4.4.2.

5.2.3 Results

Figure 5.10 shows the initially constructed degradation graph. It contains all four manually validated cuts and the associated fragments. Additionally it contains four unvalidated exoproteolytic reactions and one additional endoproteolytic reaction. For a better visual differentiation the unvalidated reactions and fragments are represented as dashed edges and nodes.

The newly identified reactions target different peptide fragments (nodes) of the originally expected degradation graph. Two of the exoproteolytic reactions directly target the N- and C-terminus of the base peptide and produce the one amino acid shorter fragments b and c. The two other exoproteolytic reactions digest the N-terminal fragment (fragment d) produced by the endorproteolytic cut at position 7 of the base peptide (generating fragment e) and the C-terminal fragment (fragment j) produced by the endoproteolytic cut at position 9 of the base peptide (generating fragment k). The unvalidated endoproteolytic reaction cuts the fragment b, which was produced by an unvalidated cut at the N-terminus of the base peptide, at position 9, and generates the fragments i and k. Fragment k is also unvalidated.

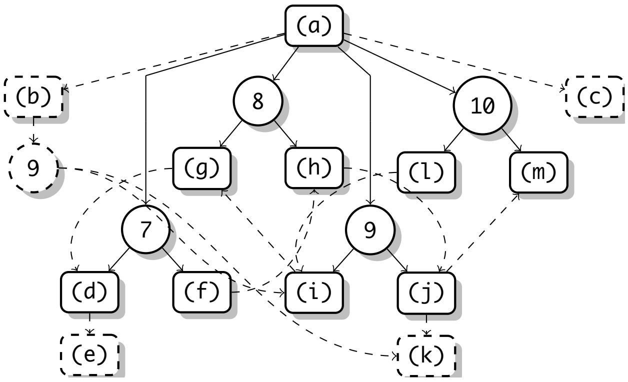
Additionally to the previously described unvalidated reactions and fragments one can see that the degradation graph construction method added several exoproteolytic reactions interconnecting the validated fragments. This can be explained by the idea of the construction method, as it always adds every possible explanation to the graph. In this particular case it leads to all the interconnecting edges. Although all of these reactions are possible, they are very unlikely and hence should be removed during the optimization. To model this also during the parameter optimization step, the initial values were chosen according to the heuristic described in Section 4.4.3.

We also adjusted the weighting factors of the ranking score described in Section 4.5.1 to $w_C = 0.8$ and $w_V = 0.2$. The reasoning for this decision was the lack of observations for the actual reaction, i.e., sampling points between time point 8 and 9. The insufficient amount of sampling points may affect the quality of the fit of the time series such that the estimated rate constants may not be that reliable. Hence a decreased weight for the quality of fit w_C was justified.

Combined with the above presented customization, the optimization of the degradation graph structure was carried out as described in Section 4.5.2. Since for the investigated combination of urine proteins and beta-2-microglobulin the correct exo- and endoproteolytic

(a)	SKDWSFYLLYYTEFTPTEKDE
(b)	SKDWSFYLLYYTEFTPTEKD
(c)	KDWSFYLLYYTEFTPTEKDE
(d)	SKDWSFY
(e)	KDWSFY
(f)	LLYYTEFTPTEKDE
(g)	SKDWSFYL
(h)	LYYTEFTPTEKDE
(i)	SKDWSFYLL
(j)	YYTEFTPTEKDE
(k)	YYTEFTPTEKD
(l)	SKDWSFYLLY
(m)	YTEFTPTEKDE

(a)



(b)

Figure 5.10: Initial degradation graph for beta-2-microglobulin estimated from real data. Shown is the degradation graph for beta-2-microglobulin which was initially estimated from a MALDI time series. (a) The mapping of indices to sequences. (b) The degradation graph. The dashed edges and nodes represent the reactions that were not validated manually.

reactions are unknown, the resulting list of ranked subgraphs needs to be inspected. Figure 5.11 shows the progression of the score with respect to rank.

Further inspection of the progression of scores gives us additional insights into the created subgraphs. Figure 5.11 clearly shows multiple areas where subgraphs were ranked with a nearly identical score (e.g., between rank 1 and 19). By inspection of the corresponding degradation graphs, one can see that they all have a common structure in terms of nodes and edges but vary in the number of side reactions like the interconnecting edges. Nearly all of the non-shared edges have an estimated reaction rate of 10^{-6}. The larger drops in the score mostly correspond to additions or removals of nodes in the corresponding degradation graphs.

Figure 5.12 shows the degradation graph with the highest score. The resulting degradation graph contains all validated fragments and the expected endoproteolytic reactions cutting the beta-2-microglobulin fragment at positions 7-10. All unvalidated fragments and reactions were removed during the optimization. Not all of the reactions interconnecting the fragments produced by the validated reactions were removed during the optimization (see dashed edges in Figure 5.12). Two reactions, converting fragment i to g and g to d, are still included in the optimized degradation graph but have an estimated reaction rate of 10^{-6}. Although the reactions are still included in the degradation graph they have effectively no

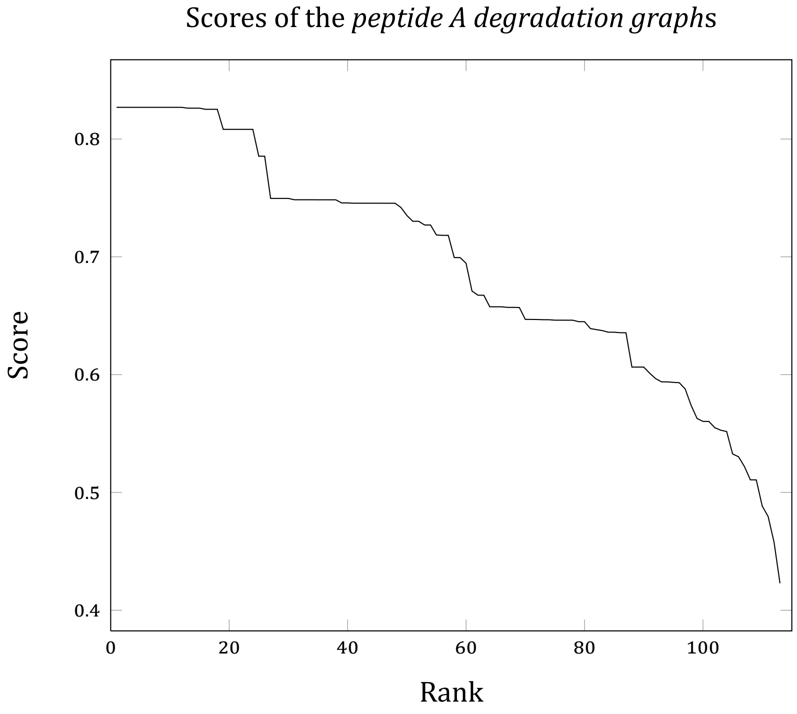


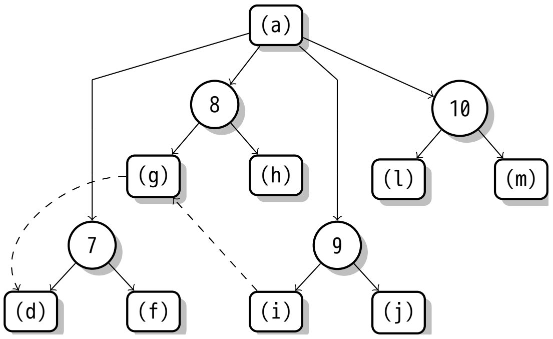
Figure 5.11: Plot of degradation graph rank vs score. Shown is the progression of the score S computed for all created sub graphs of the initial degradation graph shown in Figure 5.10.

influence on the dynamics of the system and can therefore be neglected.

Figure 5.13 shows the time course of the observed and predicted intensities for a subset of peptide fragments of the highest scored degradation graph (see Figure 5.12). The measured data points fluctuate around the predicted intensity values, but the estimated values still seem to reproduce the general behavior of the system. To improve the estimation one could conduct more mass spectrometry measurements, especially in the time from 7 h to 24 h. The approach would also benefit from a more robust quantification, e.g., via spiked in control samples. This would also ease the inter-spectra normalization.

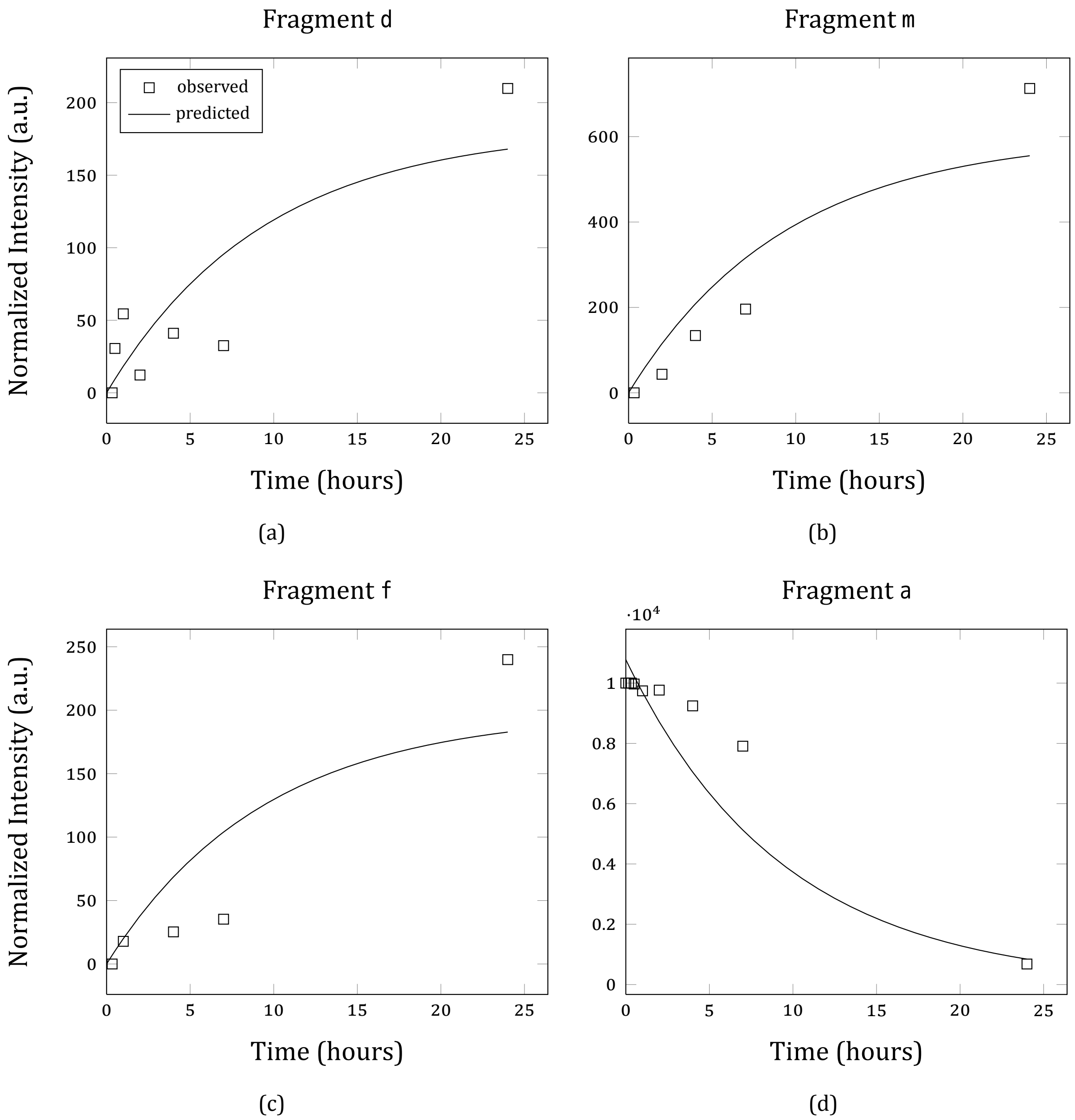
(a)	SKDWSFYLLYYTEFTPTEKDE
(d)	SKDWSFY
(f)	LLYYTEFTPTEKDE
(g)	SKDWSFYL
(h)	LYYTEFTPTEKDE
(i)	SKDWSFYLL
(j)	YYTEFTPTEKDE
(l)	SKDWSFYLLY
(m)	YTEFTPTEKDE

(a)



(b)

Figure 5.12: Optimized degradation graph for the beta-2-microglobulin fragment estimated from real data. Shown is (a) the mapping of indices to sequences, (b) the optimized degradation graph for the beta-2-microglobulin fragment. The dashed edges and nodes represent those reactions, that were not validated manually and are still present after optimization. Note that the reaction rates for the dashed reaction were estimated to values of 10^{-6} and with this have no practical influence on the dynamics of the system.



(a)

(b)

(c)

(d)

Figure 5.13: Intensity course for different fragments of the manually validated degradation graph. See text for more details.

Chapter 6 Computational aspects

In the past decades the computational requirements in research have changed drastically. More and more institutions need large computational resources to analyze the permanently increasing amounts of data. Especially in the field of computational biology the amount of data has increased massively in the last years due to the improvements in measurement techniques in both genomics and proteomics. Not only data intensive disciplines but also simulation based approaches, e.g., in molecular dynamics, require serious amounts of computation time.

In this thesis we introduced such computationally demanding approaches. For instance the parameter estimation step (see Section 4.4) or the degradation graph optimization (see Section 4.5.2) require a considerable amount of time when carried out on complex data sets. Also the simulation of mass spectrometry data, as described in Chapter 3, is time consuming if the sample is large or multiple parameter settings should be evaluated.

The most common solution to the problem is the establishment of compute clusters or grids. In both cases the computationally expensive task is delegated to a specialized compute infrastructure. Both clusters as well as grids require a considerable amount of money in terms of buying and maintaining the infrastructure. Trained personnel is also required to operate the cluster or grid. But both money and trained personnel is not always available in the required amounts. On the other hand many workstations in research institutes have idle periods where they are not or only slightly used. Conrad [43] described an approach to utilize these idle resources in an easy and effective way. The approach was termed the *quasi ad-hoc Grid* (QAD Grid).

For the experiments and the corresponding computations described in this thesis we utilized the QAD Grid. In the following sections we will introduce the QAD Grid and the extensions we made for to carry out these experiments. We will conclude this by describing the integration of the methods and tools for simulation and the analysis of proteolytic processes in the QAD Grid.

6.1 The quasi ad-hoc grid

Dividing problems into smaller tasks and distributing them onto multiple computers or processors is one of the classical paradigms in computer science. Beside the parallelization of the process on a single computer utilizing multiple processors or even graphic cards, two main approaches exist: *compute clusters* and *compute grids*.

Clusters usually consist of a group of homogenous[1] computers, so called nodes, and a master or host system, which distributes individual tasks to the cluster nodes. In most cases all the cluster nodes will be located at the same institute or compute facility. From a user's perspective the cluster will be represented as a single system, the host, to which the user submits tasks, sometimes in combination with requirements regarding the resources needed by the task (e.g., the amount of memory). The host then decides to which of the cluster nodes the task gets assigned and, after completion, reports the task as finished.

In comparison, grid systems can be much more heterogenous and the number of computers inside a grid is not fixed. Also the nodes of the grid do not need to be installed at the same location, but can communicate via the internet. Generally speaking grids are more loosely coupled then clusters. They bundle existing compute resources on a bigger scale to tackle large computational tasks in a joint effort. A detailed introduction into the field of grid computing is beyond the scope of this thesis. We therefore refer the interested reader to Foster and Kesselman [174].

Cluster and grid approaches have their clear benefits due to the large resources they make available to the users, but they also are hard to operate and, especially for small institutions with no dedicated funding, unaffordable. As an alternative to full featured grid setups so called *peer-to-peer* grids or *free-to-join* grids have been proposed, where institutions can join a specialized grid infrastructure based on easy to maintain programs like OurGrid [175]. Each institution contributes its idle resources and in exchange can submit their own jobs to the grid. The idea of utilizing idle resources is not new and was already proposed by Litzkow, Livny, and Mutka [176] in the late 1980s.

A similar approach is followed by the QAD Grid [43]. It maintains a list of tasks and client machines (so called worker). The individual workers are running a specialized program and assign themselves to a task as soon as they are idle. If no worker is available the central grid server can also attempt to start the worker program on registered clients.

[1] With respect to their hard- and software configuration.

6.1.1 Building blocks of the quasi ad-hoc grid

This section will describe the relevant components of the QAD Grid. A full list of all components and additional details on the exact realization can be found in Conrad [43, Chapter 5]. Here we will give a short overview to give the reader a broad understanding of the capabilities of the QAD Grid.

(1) Grid Platform Server: The Grid Platform Server constitutes the central controlling entity of the QAD Grid. It also serves as a portal where users can access the Grid system, e.g., via web-based front-end.

It provides the following core features:

Data-Management It maintains storage and distribution of the individual datasets inside the Grid, controls the accessibility of each individual dataset, and delivers data on request to the workers.

Job-Management It controls the creation, scheduling, and distribution of jobs inside the QAD Grid.

Worker-Management It maintains a list of available workers and starts and stops them upon request.

Security-Management As security is a central issue in distributed systems, the Grid Platform Server controls all security related issues, e.g., data accessibility on a per user and per worker basis, data and communication encryption, and user login rights.

The Grid Platform Server was developed with two additional requirements: performance and reliability. To ensure that both requirements are always met, the Grid Platform Server can be executed in parallel on multiple systems. The job and worker details are periodically synchronized between all instances using a central database system.

(2) Data: All datasets are stored on the Grid Platform Server. Further the data is replicated to the worker's host system to ensure efficient access during the computations.

(3) Workers: The Workers perform the computational work and mirror parts of the data. They request jobs from the Grid Platform Server, process them, and transfer the results back to the server.

6.1.2 QAD Grid design concepts

In this section we will shortly describe the design concepts that played an important role while developing the QAD Grid and that distinguish the approach from other grid systems.

Database Centered Communication Based on the idea that communication with modern database systems is fully platform independent, the communication in the QAD Grid is based on the underlying database system. The communication between the grid nodes and the Grid Platform Server as well as between grid nodes is based on the modification and creation of defined database entries. Those entries are read and interpreted by both the grid nodes (the workers) and the Grid Platform Server.

Security A central requirement for distributed systems with heterogeneous clients is a centralized security. In the QAD Grid this is achieved by utilizing the existing database security features. All connections to the database require a valid account and are encrypted and transferred over a SSL[2] connection.

Data Access As stated earlier, all datasets are centrally hosted on the Grid Platform Server. From there the data is distributed to the workers on an *on-demand* basis or replicated automatically in the background. Using these two data deployment strategies the QAD Grid can operate in *data-follows-client* as well as *client-follows-data* mode.

Job/Worker matching An essential task in every grid system is the correct allocation of tasks to workers. In the QAD Grid approach this is archived based on the so called *service ID*. Jobs are tagged with a service ID which specifies the particular operation or computation to be performed. The worker then requests available jobs from the Grid Platform Server tagged with its own service ID, i.e., jobs that it is able to process.

Job Pull Compared to many other cluster and grid systems, the QAD Grid system does not distribute (or push) jobs to the workers, instead the workers pull jobs from the Grid Platform Service. This eases the problem of availability of matching workers. The central Grid Platform Server maintains a list of active tasks and every client machine can decide independently (e.g., based on its current workload) whether it pulls a job from the central system or not.

Worker Injection In contrast to many other systems, the QAD does not require that the client software is already installed on the target/client machine. On demand the Grid Platform Server deploys the necessary client software to the target system and starts the

[2] SSL, *Secure Sockets Layer*, is a cryptographic protocol that provides secure communication over the Internet [177].

worker. The transfer is managed via SSH[3] for Linux/Unix based systems and WMI[4] for Microsoft® Windows® clients. This solely requires that the Grid Platform Server has access to the respective machine.

(Hot) service deployment In conventional grid or cluster systems a new service (in this case we mean a new application or available computational service) has to be distributed throughout the grid system by installing the new application on every node. In the QAD Grid each worker deploys its own services to the cluster by registering its service ID in the Grid Platform Server and with this, enabling job submissions with this particular service ID. In this specific context hot service deployment means that the service is available as soon as the worker has started. No restart of any client or server systems is required.

Workflows / -items The QAD Grid is not only able to handle single job submissions, it can also model dependencies between tasks. Based on this workflows can easily be modeled and executed on the QAD Grid.

6.1.3 The worker concept

The worker is an abstraction of an analysis step, which can reach from a simple, atomic operation (e.g., file format conversion) up to complex analysis (e.g., peak picking). The worker is a program that runs on client system and communicates with the Grid Platform Server via database entries. Accordingly a QAD Grid Worker can be implemented in any modern programming language that is able to connect to a central database. This further ensures platform independence. Implementations for Java and Matlab already exist [43] and perform a multitude of different tasks. A worker needs to provide the following features:

- Ability to connect to and register at the Grid Platform Server. Registration at the Grid Platform Server includes the different information like IP address, operating system, and its current workload. But most importantly it needs to provide a *service ID*. The service ID defines what kind of job the worker is able to handle. Every job in the database is also tagged with a service ID and the Grid Platform Server matches jobs to workers based on this service ID.

[3] SSH, or Secure Shell, is a network protocol that allows data exchange between two systems (a client and a server) over a secured channel.

[4] The Windows Management Instrumentation (WMI) provides the infrastructure for the management of Windows-based operating systems [178].

- Request a job including the job's parameters. This is executed periodically to ensure that as soon as a new task, with a matching *service ID*, is registered in the database it will be executed by one of the available workers.

- Loading data from either the Grid Platform Server (i.e., the database) or from a specified file system via a defined protocol (e.g., FTP or S-FTP).

- Execute its defined task on the loaded data.

- Transfer the results of the computation back to the Grid Platform Server. This can be again realized in different ways (e.g., database or file based).

- Send regular messages to the Grid Platform Server to indicate its current status. The status is used as a kind of visual feedback for the web-based front-end. Using this the worker can for instance indicate the current phase of the analysis or other types of progress information. An additional information sent in the status message is the workload of the machine it is running on, caused by *foreign processes*. This information is used to distribute tasks based on the available resources, e.g., if two workers with the same *service ID* are available, the one with the lower workload will be chosen.

All this functionality is provided by the *Base Worker*, a Java based reference implementation. Starting from the Base Worker one can easily derive one's own Java based worker. Alternatively one can implement the above mentioned requirements on one's own.

6.1.4 Extending the worker implementation

The original implementation of the base worker indeed provided all the given functionality but had two mayor downsides:

1. Setting up a running worker required a certain knowledge of the underlying process and the methods provided by the base worker. This included the knowledge of the correct sequence of initialization steps as well as an individual implementation of the pull request.

2. The base worker had several dependencies to different third-party libraries that needed to be accessible on the client machine prior to the execution.

3. The problem of transporting and accessing data during the computation that is not stored in the database was not solved completely.

To ease the implementation of our own and any future workers we redesigned the base worker and its build system with these shortcomings in mind. In the following sections we will present the different approaches we took to solve the above named problems. We will start by describing the restructuring of the base worker. Afterwards we will describe the changes made to the build system. Using the new build system we implemented a new remote startup mechanism which will also be described. Our solution to the data transportation problems will be described in Section 6.2.

Restructuring the base worker

The aim of this restructuring was to reduce the amount of code needed to implement a worker. We started with the original implementation, where the implementation of a custom worker started with deriving from the base worker class.

The derived worker should contain only the task specific code, e.g. the peak picking algorithm or the degradation graph construction code. In contrast, the original implementation required additional code, to pull the job from the Grid Platform Server, initialize the worker, and handle occurring errors.

This is a classical violation of one of the fundamental principles of Object-Oriented software design, the open-closed principle [179, 180]. The open-closed principle states that a module (in our case the base worker) should be closed for modifications and open for extensions. In our case this means that the base worker should be open for extensions in form of new workers being implemented based on the existing functionality, but it should be closed to modifications in the general handling of jobs. But in this case, as the functionality to handle a task was not hidden from the deriving class, but explicitly needed to be reimplemented each time a new worker was implemented, it was not closed at all.

Hence we decided to restructure the base worker to follow the open-closed principle. A suitable structure for the described process of waiting for jobs and handling them as soon as they are entered into the Grid Platform Server is the *Observer Pattern* [181]. It defers the process of waiting for the task and handling the state transitions to a second process, which repeatedly queries the database for a new task. The worker itself serves as an observer. The base worker defines the abstract method `onTask()` which needs to be implemented by every derived class. As soon as the worker is started it will also start the background task which notifies the worker as soon as a task arrives.

By simply applying the well known observer pattern we could drastically ease the implementation of new workers based on the already existing base worker implementation.

Redesigning the worker build system

The original worker implementation was based on an ant[5] based build system. Since the requirements regarding the portability of the build system changed with the increased amount of people working on the project, we decided to switch to Apache Maven[6] as new build system. The central part of the Maven build system is the *Project Object Model* (POM). The POM is an XML file (named pom.xml) containing all necessary information about the project to build and execute it. Especially it is used to define all necessary dependencies to external libraries needed to build and execute the project. Maven itself will then take care of satisfying those requirements by downloading them from different, configurable sources.

The restructured build system was also used to ease the remote startup mechanism described earlier (see Section 6.1.3). By handing the dependency management over to Maven, a minimalistic POM can be defined which has a single dependency to the worker that should be started. It further uses the *exec:java* goal[7] to tell Maven to execute the main class of the desired worker. By using this the complete remote startup procedure could be reduced to the following steps:

1. Generate startup pom containing a dependency entry for the worker which should be started and exec:java goal for its main class.

2. Transfer the startup pom to an empty directory on the target machine.

3. Execute Maven in the directory where the startup pom was copied to.

The size of the startup pom is $\approx$ 1kB, thus transferring it is extremely fast. The time for the startup depends on the libraries the worker depends on and the speed of the connection between the remote machine and the server providing the dependencies. Since dependencies in Maven are versioned this also solves also distribution of new versions of workers. The startup POM simply needs to be adapted to the new version and Maven will take care of downloading the corresponding version before starting the worker.

6.2 Enabling transportation of mass data in the QAD Grid

Handling data inside any grid system is always a problematic task. For the QAD Grid this was originally solved by storing all information on the QAD Platform Server more precisely in the

[5] The Apache Ant project [182]

[6] Apache Maven Project [183]

[7] `http://mojo.codehaus.org/exec-maven-plugin/java-mojo.html`

underlying database. The base worker provided the necessary functionality to retrieve and store datasets from the database.

While integrating the presented approaches it became obvious that the database centric approach has its limits, especially when we need to integrate existing tools that operate on files. Therefore we decided to add a file based transportation mechanism to the base worker. The file based transportation between the grid nodes imposed several requirements on the used transfer protocol:

Speed The transfer protocol should allow high throughput especially when transferring huge datasets.

Stability The transfer should be reliable and if possible a resume functionality should be available, in case the connection was lost.

Security It should be possible to restrict access to different resources.

Given the dynamic nature of the QAD Grid approach the choice of technologies and transport protocols is limited, especially if one would need to fulfill all requirements at once. We therefore extended the existing QAD Grid and worker implementation by adding functionality to transport files via two different protocols, namely WebDAV [184] and BitTorrent [185]. Both protocols have their specific advantages depending on the actual requirements, e.g., size of the datasets, available bandwidth, or number of workers requesting resources in parallel.

The WebDAV protocol was implemented for transport in a closed network, e.g., the local network of a research institute, where we assume a fast connection between the server and the worker. It also provides authentication capabilities ensuring that only specific workers can access specific datasets.

The BitTorrent protocol was implemented for the transfer of data in slower networks and for situations where multiple workers need to access or store huge datasets simultaneously. In such scenarios a single WebDAV store would not be sufficient to deliver or accept the data at a high enough speed. By the ability of the BitTorrent protocol to utilize the up- and download capacities of all workers in the grid, the overall transfer speed can be increased.

In the following sections we will describe in more detail the motivation for both protocols and present some of the implementation-specific details.

WebDAV based data transport

WebDAV (Web Distributed Authoring and Versioning) is an extension of the HTTP [186] protocol for collaborative editing and management of files via the world wide web. In the worker scenario it requires an existing WebDAV share that can be accessed by the worker. This share can be accessed by the workers to either download the input data for a specific operation or task or store the results of a task.

Implementation

We implemented a class *WebDAVClient* which provides all necessary methods to retrieve, store, delete, and list files stored on a WebDAV share. Both authenticated and unauthenticated connections are supported. The implementation uses the webdavclient4j project[8] as transportation backend. Due to the abstraction of the operations the backend can also be easily substituted by other implementations.

BitTorrent based data transport

While the classical data transfer approach (like WebDAV) relies on a single server, providing a specific resource (e.g., a file), and a client, retrieving this resource, the BitTorrent protocol is based on the idea that every client, retrieving a resource can also be used to distribute the already transfered parts of the resource to other clients. This drastically reduces the traffic generated on the server compared to the classical data transfer approach. It can also increase the transfer speed, since not only a single server is uploading data, but many clients provide parts of the data and thus increase the overall transfer speed.

BitTorrent basics

The BitTorrent protocol defines two basic entities, the *Tracker* and the *Peer*. The tracker coordinates the distribution of the files between the peers. The file transfer is initialized with a *torrent* file, which provides all necessary informations on the file that should be shared (e.g., name and size), the address of the tracker, which coordinates the transfer, and a list of *Pieces*. Pieces are small segments of the original file that are protected by a hash code. As soon as a peer connects to the tracker, the tracker will tell him which other peers can provide which pieces. The peer will then start to non-sequentially download the available pieces from the different available peers. After a piece is downloaded successfully it is verified by the peer using the hash given in the torrent file. The hash is computed using the SHA-1 hash function [187]. A peer which already downloaded all pieces is called a *Seeder*.

[8] `https://sourceforge.net/projects/webdavclient4j/`

Implementation

The complete implementation is based on the bitext Java Bittorrent API[9]. A major effort was made to add an additional abstraction layer so that the underlying API can also be replaced in future versions.

Our Bittorrent API implementation consists of two parts, the client and the tracker. In the following paragraphs we will shortly highlight the differences between a classical client or peer respectively tracker and the one implemented in the QAD Grid.

Tracker The classical tracker serves mainly as a coordinator between the peers that have already downloaded a torrent file. In the the QAD Grid, in contrast to the classical BitTorrent scenario, we need to provide a way to distribute the torrent files to the different available grid nodes. Therefore the tracker was extended with the functionality to list all managed torrents and their properties (list of peers, date of creation) and to distribute the corresponding torrent files on demand.

The tracker also serves as a client for each newly announced torrent file. This ensures that each announced data set has at least two peers.

Client The client was extended with the functionality to automatically discover new torrents by requesting a list of torrents from the tracker. It can then decide if it wants to start downloading one of the available torrents based on different decision strategies:

Newest It will automatically start to download the newest of all available torrents.

Least-Seeders It will choose the torrent with the least seeders to increase the amount of peers that can serve the corresponding file.

Each worker can further generate a new torrent based on an existing file and announce the generated torrent to the tracker. As soon as it announced a new file it will also start to serve as a peer. Once the complete file is available the worker will automatically be a seeder.

All this happens only if a worker is idle, i.e., does not process a worker specific task. Each worker stops sharing as soon as a new task arrives and restarts as soon as the task is finished.

Adding data to the QAD-Grid Uploading data into the QAD-Grid is realized using a simple upload client. The user can select a file from the local hard drive and add it to the QAD-Grid. The client tool will then announce the selected file to the tracker and start uploading it. Since the tracker was implemented in such a way that it automatically also downloads announced

[9] `http://code.google.com/p/bitext/`

files there will be at least one client in the grid that downloads the file. As soon as the upload is finished the client can be closed. The file is then stored on the tracker and will be replicated to workers inside the grid.

Requesting specific files from the tracker If now a worker wants to access a specific file stored as torrent it can easily request the corresponding torrent file from the tracker and start to download the corresponding file. Since the tracker is automatically a seeder there will be at least one seeding client. In most cases the file will already be distributed to other workers resulting in more seeding clients and with this a higher transfer speed.

6.3 Embedding the degradation graph into the QAD Grid

In the previous sections we described the QAD Grid and the extensions that we developed. Based on these extensions we integrated the degradation graph construction as well as the structure optimization, described in Section 4.3 and Section 4.5.2, as a single worker into the QAD Grid. Additionally we integrated the parameter estimation as a separate worker into the QAD Grid.

To utilize the existing base worker implementation we chose Java as programming language for the development. The graph data structure was realized using the open-source graph library JUNG–Java Universal Network/Graph Framework[10]. The data structures required for the mass spectrometry data handling were provided by the jmstoolbox, a small Java based library for rapid development and prototyping of proteomics algorithms, developed in cooperation with Axel Rack, that we will shortly introduce here.

jmstoolbox – a minimalistic java library to handle mass spectrometry data

The jmstoolbox was developed to provide common data structures and algorithms for the development of mass spectrometry data analysis software. It consists of five parts, each bundled in a separate module.

base The *base* module provides the basic proteomics data types like amino acids, proteins, and peptides, as well as mathematical concepts and algorithms (e.g., clustering).

processor The *processor* module provides interfaces and abstract implementations of algorithms or processes, which can be used to effectively parametrize algorithms and run them concurrently in separate threads.

[10] `http://jung.sourceforge.net/`

proteomics The *proteomics* module provides data types specific for mass spectrometry based proteomics, like spectrum and peak. It further provides utility methods to handle those data types (e.g., efficient filtering, transformation, or alignment).

io The *io* module provides support for different mass spectrometry data formats like DTA[11] or pepXML[12].

visualization The *visualization* module provides basic plotting functionality for mass spectra based on the JFreeChart library[13].

Implementing the degradation graph in the jmstoolbox

Using the jmstoolbox and JUNG we implemented the complete degradation graph approach as a separate library. It provides the degradation graph data structure and a method to construct it based on mass spectrometry time series data. The construction algorithm utilizes the processor abstraction of the jmstoolbox. It further provides a visualization of the degradation graph based on the Java Swing API[14] and an export to the dot format to visualize the degradation graph using Graphviz [188, 189]. The degradation graph module is further able to utilize the MEROPS database [1] to check if the identified proteolytic cleavages are already annotated in MEROPS.

Embedding POEM in the QAD Grid

The last step to carry out the complete analysis described in Chapter 4 is the parameter estimation. As described earlier it is based on POEM, a Matlab based parameter estimation tool.

To again utilize the benefits of the QAD Grid we also wrapped POEM into a separate worker. To achieve this we extended the already existing Matlab worker to specifically handle the needs of POEM and to store the estimated parameters in the central database.

By implementing the parameter estimation as separate worker we could easily distribute the most time consuming step of the approach on different machines, by parallelizing the estimation steps for the different sub-graphs (see Section 4.5 for details). By separating these two steps we could also easily substitute the optimization technique by a different tool.

[11] http://www.matrixscience.com/help/data_file_help.html#DTA

[12] http://tools.proteomecenter.org/wiki/index.php?title=Formats:pepXML

[13] http://www.jfree.org/jfreechart/

[14] http://docs.oracle.com/javase/tutorial/uiswing/

6.4 Embedding MSSimulator in the QAD Grid

The previous chapters presented the QAD Grid as well as the integration of the degradation graph approach into the QAD Grid. Parts of the validation of the degradation graph approach (see Chapter 5) were carried out on simulated data. This and the computational requirements of extensive simulations lead to the decision to also integrate MSSimulator into the QAD Grid. Therefore we implemented a new worker that manages the call of MSSimulator on the worker machine. Although the execution of MSSimulator given an executable and a configuration and input file is straightforward from a Java program, finding the correct executable and the required input files can be problematic in the context of the QAD Grid. Therefore we will shortly summarize below how we solved these two issues.

Finding the correct MSSimulator executable

Since MSSimulator is a platform specific executable we needed to guarantee that the worker finds the correct executable on its target machine. To ensure that, we utilized the central configuration store, a list of configuration files stored on the grid platform server. The configuration files are line oriented properties files[15]. The central configuration store contains machine specific properties, that are accessed during the startup of the worker. There we placed for each worker that can execute MSSimulator the necessary information (path to executable, library path) to run MSSimulator on its specific machine. The user can further add a `local.properties` file to the directory where the worker is executed to overwrite the existing configuration, e.g., to use a newly build version of MSSimulator. If a MSSimulator worker is started and cannot find a valid MSSimulator executable it will automatically shutdown.

Configuring MSSimulator

For the configuration of the simulation we use the standard worker parameters. Encoded are the directory containing the input files, an OpenMS ini-file containing the simulation parameters and a target directory to store the simulated files. It further accepts an optional TOPPAS [125] pipeline which is applied to each of the simulated mass spectra as optional post-processing step.

[15] `http://docs.oracle.com/javase/7/docs/api/java/util/Properties.html`

Chapter

7 Conclusion and Future Directions

With this thesis we made contributions to two different fields of computational biology. The first one is a comprehensive approach to simulation of mass spectrometry experiments; the second one is a novel approach to modeling and analysis of proteolytic processes based on mass spectrometry data. In this last chapter we will summarize these contributions and discuss possible extensions.

7.1 Conclusion

In Chapter 3 we presented MSSimulator, a simulator for mass spectrometry experiments. MSSimulator provides the so far most complete collection of models and algorithms to simulate LC-MS/MS experiments. It simulates the complete LC-MS/MS workflow, starting with enzymatic digestion and peptide separation (HPLC and CE), as well as different ionization techniques (MALDI and ESI). Subsequently MSSimulator generates raw signals using different peak shapes and variable elution profiles, as well as a customizable resolution and different types of noise. Further it can simulate label-free as well as labeled experiments. MSSimulator already includes different, widely used labeling techniques (SILAC, iTRAQ, ICPL, and ^{18}O). But the carefully designed labeling framework implemented in MSSimulator can be easily extended to support almost any labeling technique. The different levels of ground truth generated during the simulation allow a wide range of applications for MSSimulator, like algorithm benchmarking and validation scenarios as we have shown in Chapter 3. MSSimulator is available in OpenMS/TOPP and was published in Bielow et al. [41].

The second major contribution of this thesis is the *degradation graph* approach which we presented in Chapter 4. The degradation graph is used to model proteolytic processes, consisting of multiple, interacting proteolytic reactions. We further proposed an algorithm to construct the degradation graph from a time series of mass spectrometry measurements and showed how the graph can be translated into a system of ordinary differential equations. This system of ordinary differential equations can be utilized, in combination with the mass spectrometry data used for the construction, to estimate the rate constants of the individual reactions in the modeled proteolytic process. Additionally we proposed an approach to optimize the initially constructed structure of the degradation graph in the presence of noise and misleading signals. The optimization approach is based on a score that ranks sub-graphs of the initial graph with respect to their ability to explain the observed mass spectra as well as the dynamic behavior of the proteolytic reactions. We have shown in Chapter 5 on simulated and real data that the approach is able to reconstruct the structure of the degradation graph and the rate constants of the ODE system, even in the presence of noise and decoy signals. The optimization approach could remove all falsely added nodes and edges from the degradation graph, leaving only the original proteolytic reactions as final result. The complete method was published in Aiche et al. [42].

In Chapter 6 we described the implementation of the degradation graph approach and how we integrated it into the proteomics.net platform [172]. The integration into the proteomics.net platform allowed us to perform a large scale testing on simulated data by distributing the different steps of the analysis (e.g., construction, parameter estimation) on multiple compute nodes. We also extended the proteomics.net platform by adding transportation methods for large data sets based on the BitTorrent and WebDAV protocols.

7.2 Improving the ground truth

As we described earlier, MSSimulator is currently the most comprehensive simulation framework for mass spectrometry data. But to improve its usefulness for algorithm development and benchmarking as well as its acceptance in the community, it can be further extended in different ways. Here we want to summarize some of these extensions.

MSSimulator would benefit from an automated parameter selection based on an existing datasets. Currently all simulation parameters (e.g., resolution, noise, peak shape) have to be chosen by hand, which can be time-consuming and, if not carried out carefully, prone to error. We provide reasonable presets for different machine types but this does not cover all settings. Automating this step based on existing datasets would make the benchmarking and testing more convenient, would increase the usability of MSSimulator, and could help in avoiding errors.

Additionally, more instrument specific properties could be integrated into the simulation to make the data and the resulting algorithmic problems when analyzing the data more realistic. For instance, the type of noise in real data can differ with the used instrument [190] which is currently not reflected by MSSimulator.

The already started benchmarking of available tools could be extended as well as it would on one hand provide a solid overview of the performance of available techniques and would, on the other hand, show the limits and possible extensions of MSSimulator.

7.3 Validation and extension of the degradation graph concept

The degradation graph approach requires an intensive validation on more real data sets to benchmark its performance and limits. In the modeling and construction approach one could consider the inclusion of unobserved peptides into the graph, i.e., peptides that participate in the reactions but are not observable in the mass spectra, maybe due to poor ionization properties or a too fast degradation in between two sampling points. An adoption of the construction algorithm is possible and would require an additional searching step. If a fragment was not found one would need to search for all possible sub-fragments of the unobserved fragment to ensure that it was not further degraded.

The scoring could also be improved, as it currently requires that the user chooses the weighting parameters for the two components of the score. A first approach towards a new, non-weighted score was already proposed in Chapter 4 but requires further testing.

It could further be tested if labeling based quantification strategies improve the estimated reaction rates. Combining multiple time points in a labeled experiment, maybe in combination with a pooled reference to ensure the comparability between different labeling runs, could improve the quantification results and with it the estimated rate constants.

An extension towards LC/MS data would also be interesting. For example, a low identification rate in LC/MS map can in some cases be explained by proteolytic activity. The peptides generated by the proteases are not tryptic and hence will not be identified by most identification approaches. Here one could start from identified peptides and try to infer the identity of the unidentified ones using our construction approach. The peptide mass fingerprinting based identification could be further improved in this context by comparing the retention times with predicted ones [50].

Finally, the generated ODE system could be extended by including the different proteolytic enzymes and the associated binding reactions, e.g.,

$$\mathrm{A} + \mathrm{P} \xrightarrow{k_1} \mathrm{AP} \xrightarrow{k_2} \mathrm{B} + \mathrm{P}, \tag{7.1}$$

for the degradation of peptide A to peptide B catalyzed by the protease P. While this would further increase the complexity of the parameter estimation, since the number of reactions rates would double and neither the complex AP nor the protease P will be observable in the mass spectra, it would also allow an even more accurate description of the actual proteolytic reaction.

7.4 Extending our view on biomarkers

The presented degradation graph approach could be the foundation for an extension of our understanding of biomarkers. Currently most studies focus on the identification of single or multiple differentially expressed proteins. With the results of our approach one could not only compare the differentially expressed proteins but also parts of the underlying process that lead to the differential expression. Using again the renin-angiotensin system from the introduction as an example, we would not only analyze the differential expression of the Angiotensin I and its fragments (see Figure 1.1) but we could be able to analyze the different paths in the system and compare them between different states (e.g., healthy versus a specific cardiovascular disease). Combined with a reasonable approach to assess the differences between the structure of two degradation graphs and the estimated rate constants, this would drastically improve our ability to classify different biological conditions.

Curriculum Vitae

For privacy reasons, the curriculum vitae is not contained in the online version of this thesis.

For privacy reasons, the curriculum vitae is not contained in the online version of this thesis.

Eigenständigkeitserklärung

Ich versichere, dass ich die hier vorgelegte Dissertation mit dem Titel „*Inferring Proteolytic Processes from Mass Spectrometry Time Series Data*“ selbstständig angefertigt und die benutzten Quellen und Hilfsmittel vollständig angegeben sind.

Ein Promotionsverfahren wurde zu keinem früheren Zeitpunkt an einer anderen in- oder ausländischen Hochschule oder bei einem anderen Fachbereich beantragt. Die Bestimmungen der Promotionsordnung sind mir bekannt.

Berlin, 03.05.2013

Stephan Aiche

Bibliography

[1] N. D. Rawlings, A. J. Barrett, and A. Bateman. „MEROPS: the database of proteolytic enzymes, their substrates and inhibitors.“ *Nucleic Acids Res.* **40**(Database issue) (Jan. 2012), pp. D343–D350.

[2] P. N. Walsh and S. S. Ahmad. „Proteases in blood clotting.“ *Essays Biochem.* **38** (Jan. 2002), pp. 95–111.

[3] J. P. Alao. „The regulation of cyclin D1 degradation: roles in cancer development and the potential for therapeutic invention.“ *Mol Cancer.* **6**(1) (Jan. 2007), p. 24.

[4] N. A. Thornberry and Y. Lazebnik. „Caspases: Enemies Within“. *Science* **281**(5381) (Aug. 1998), pp. 1312–1316.

[5] C. López-Otín and T. Hunter. „The regulatory crosstalk between kinases and proteases in cancer“. *Nat Rev Cancer* **10**(4) (2010), pp. 278–292.

[6] T. Ludwig. „Local proteolytic activity in tumor cell invasion and metastasis.“ *Bioessays* **27**(11) (Nov. 2005), pp. 1181–91.

[7] I. Cleynen, P. Jüni, G. E. Bekkering, et al. „Genetic evidence supporting the association of protease and protease inhibitor genes with inflammatory bowel disease: a systematic review.“ *PLoS ONE* **6**(9) (2011), e24106.

[8] J. T. Huse and R. W. Doms. „Closing in on the amyloid cascade: recent insights into the cell biology of Alzheimer’s disease.“ *Mol Neurobiol* **22**(1-3) (2000), pp. 81–98.

[9] J. Hardy and D. J. Selkoe. „The amyloid hypothesis of Alzheimer’s disease: progress and problems on the road to therapeutics“. *Science* **297**(5580) (July 2002), pp. 353–6.

[10] E. De Clercq. „The design of drugs for HIV and HCV.“ *Nat Rev Drug Discov* **6**(12) (Dec. 2007), pp. 1001–18.

[11] T. Masaki, H. Matsuoka, M. Sugiyama, et al. „Matrilysin (MMP-7) as a significant determinant of malignant potential of early invasive colorectal carcinomas.“ *Br. J. Cancer* **84**(10) (May 2001), pp. 1317–21.

[12] M. F. Leeman, S. Curran, and G. I. Murray. „New insights into the roles of matrix metalloproteinases in colorectal cancer development and progression.“ *J Pathol.* **201**(4) (Dec. 2003), pp. 528–34.

[13] K. Kessenbrock, V. Plaks, and Z. Werb. „Matrix metalloproteinases: regulators of the tumor microenvironment“. *Cell* **141**(1) (Apr. 2010), pp. 52–67.

[14] C. López-Otín and T. Hunter. „The regulatory crosstalk between kinases and proteases in cancer“. *Nat Rev Cancer* **10**(4) (Apr. 2010), pp. 278–92.

[15] C. Seife. „Blunting Nature's Swiss Army Knife“. *Science* **277**(5332) (1997), pp. 1602–1603.

[16] P. Ashorn, T. J. McQuade, S. Thaisrivongs, et al. „An inhibitor of the protease blocks maturation of human and simian immunodeficiency viruses and spread of infection“. *Proc. Natl. Acad. Sci. USA* **87**(19) (Oct. 1990), pp. 7472–6.

[17] I. T. Weber and J. Agniswamy. „HIV-1 Protease: Structural Perspectives on Drug Resistance“. *Viruses* **1**(3) (Dec. 2009), pp. 1110–36.

[18] A. K. Ghosh, M. Brindisi, and J. Tang. „Developing β-secretase inhibitors for treatment of Alzheimer's disease“. *J. Neurochem.* **120 Suppl 1** (Jan. 2012), pp. 71–83.

[19] L. A. Liotta and E. F. Petricoin. „Serum peptidome for cancer detection : spinning biologic trash into diagnostic gold“. *J. Clin. Invest.* **116**(1) (2006), pp. 26–30.

[20] E. F. Petricoin, C. Belluco, R. P. Araujo, et al. „The blood peptidome: a higher dimension of information content for cancer biomarker discovery.“ *Nat Rev Cancer* **6**(12) (Dec. 2006), pp. 961–967.

[21] J. Villanueva, D. R. Shaffer, J. Philip, et al. „Differential exoprotease activities confer tumor-specific serum peptidome patterns.“ *J. Clin. Invest.* **116**(1) (Jan. 2006), pp. 271–284.

[22] T. Peccerella, N. Lukan, R. Hofheinz, et al. „Endoprotease profiling with double-tagged peptide substrates: a new diagnostic approach in oncology.“ *Clin. Chem.* **56**(2) (Feb. 2010), pp. 272–80.

[23] H. Xia and E. Lazartigues. „Angiotensin-converting enzyme 2: central regulator for cardiovascular function“. *Curr. Hypertens. Rep.* **12**(3) (2010), pp. 170–175.

[24] P. Verdecchia, F. Angeli, G. Mazzotta, et al. „The renin angiotensin system in the development of cardiovascular disease: role of aliskiren in risk reduction“. *Vasc Health Risk Manag* **4**(5) (2008), p. 971.

[25] I. Schechter and A. Berger. „On the size of the active site in proteases. I. Papain“. *Biochem. Biophys. Res. Commun.* **27**(2) (Apr. 1967), pp. 157–62.

[26] M. Karas and F. Hillenkamp. „Laser desorption ionization of proteins with molecular masses exceeding 10,000 daltons“. *Anal. Chem.* **60**(20) (Oct. 1988), pp. 2299–301.

[27] K. Tanaka, H. Waki, Y. Ido, et al. „Protein and polymer analyses up to m/z 100 000 by laser ionization time-of-flight mass spectrometry“. *Rapid Commun. Mass Spectrom.* **2**(8) (1988), pp. 151–153.

[28] J. B. Fenn, M. Mann, C. K. Meng, et al. „Electrospray ionization for mass spectrometry of large biomolecules.“ *Science* **246**(4926) (Oct. 1989), pp. 64–71.

[29] R. Aebersold and M. Mann. „Mass spectrometry-based proteomics.“ *Nature* **422**(6928) (2003), pp. 198–207.

[30] M. R. Wilkins, C. Pasquali, R. D. Appel, et al. „From proteins to proteomes: large scale protein identification by two-dimensional electrophoresis and amino acid analysis“. *Biotechnology (N.Y.)* **14**(1) (Jan. 1996), pp. 61–5.

[31] O. Schilling and C. M. Overall. „Proteome-derived, database-searchable peptide libraries for identifying protease cleavage sites“. *Nat. Biotechnol.* **26**(6) (June 2008), pp. 685–94.

[32] H. Schlüter, D. Hildebrand, C. Gallin, et al. „Mass spectrometry for monitoring protease reactions.“ *Anal Bioanal Chem* **392**(5) (2008), pp. 783–792.

[33] F. Impens, N. Colaert, K. Helsens, et al. „MS-driven protease substrate degradomics“. *Proteomics* **10**(6) (Mar. 2010), pp. 1284–96.

[34] B. H. J. van den Berg and A. Tholey. „Mass spectrometry-based proteomics strategies for protease cleavage site identification.“ *Proteomics* **12**(4-5) (Feb. 2012), pp. 516–529.

[35] H. Schlüter, J. Jankowski, J. Rykl, et al. „Detection of protease activities with the mass-spectrometry-assisted enzyme-screening (MES) system“. *Anal Bioanal Chem* **377**(7-8) (Dec. 2003), pp. 1102–7.

[36] J. Yi, Z. Liu, D. Craft, et al. „Intrinsic peptidase activity causes a sequential multi-step reaction (SMSR) in digestion of human plasma peptides.“ *J. Proteome Res.* **7**(12) (Dec. 2008), pp. 5112–5118.

[37] B. Kluge, A. Gambin, and W. Niemiro. „Modeling exopeptidase activity from LC-MS data.“ *J. Comput. Biol.* **16**(2) (2009), pp. 395–406.

[38] P. Dittwald, J. Ostrowski, J. Karczmarski, et al. „Inferring serum proteolytic activity from LC-MS/MS data.“ *BMC Bioinformatics* **13 Suppl 5** (2012), S7.

[39] T. Nilsson, M. Mann, R. Aebersold, et al. „Mass spectrometry in high-throughput proteomics: ready for the big time.“ *Nat. Methods* **7**(9) (Sept. 2010), pp. 681–685.

[40] N. Bandeira, A. Nesvizhskii, and M. McIntosh. „Advancing next-generation proteomics through computational research.“ *J. Proteome Res.* **10**(7) (July 2011), p. 2895.

[41] C. Bielow, S. Aiche, S. Andreotti, et al. „MSSimulator: Simulation of Mass Spectrometry Data.“ *J. Proteome Res.* **10**(7) (July 2011), pp. 2922–2929.

[42] S. Aiche, K. Reinert, C. Schütte, et al. „Inferring Proteolytic Processes from Mass Spectrometry Time Series Data Using Degradation Graphs“. *PLoS ONE* **7**(7) (July 2012), e40656.

[43] T. Conrad. „New Statistical Algorithms for the Analysis of Mass Spectrometry Time-Of-Flight Mass Data with Applications in Clinical Diagnostics“. PhD thesis. Berlin, Germany: Freie Universität Berlin, 2008.

[44] B. Domon and R. Aebersold. „Mass spectrometry and protein analysis“. *Science* **312**(5771) (Apr. 2006), pp. 212–7.

[45] B. F. Cravatt, G. M. Simon, and J. R. Yates 3rd. „The biological impact of mass-spectrometry-based proteomics.“ *Nature* **450**(7172) (Dec. 2007), pp. 991–1000.

[46] M. Sturm, A. Bertsch, C. Gröpl, et al. „OpenMS - an open-source software framework for mass spectrometry.“ *BMC Bioinformatics* **9** (2008), p. 163.

[47] C. Bielow. „Quantification and Simulation of Liquid Chromatography-Mass Spectrometry Data“. PhD thesis. Berlin, Germany: Freie Universität Berlin, 2012.

[48] J. M. Burkhart, C. Schumbrutzki, S. Wortelkamp, et al. „Systematic and quantitative comparison of digest efficiency and specificity reveals the impact of trypsin quality on MS-based proteomics“. *J Proteomics* **75**(4) (Feb. 2012), pp. 1454–62.

[49] N. Pfeifer, A. Leinenbach, C. G. Huber, et al. „Improving peptide identification in proteome analysis by a two-dimensional retention time filtering approach“. *J. Proteome Res.* **8**(8) (Aug. 2009), pp. 4109–15.

[50] N. Pfeifer, A. Leinenbach, C. G. Huber, et al. „Statistical learning of peptide retention behavior in chromatographic separations: a new kernel-based approach for computational proteomics.“ *BMC Bioinformatics* **8** (2007), p. 468.

[51] L. Moruz, D. Tomazela, and L. Käll. „Training, selection, and robust calibration of retention time models for targeted proteomics“. *J. Proteome Res.* **9**(10) (Oct. 2010), pp. 5209–16.

[52] A. Bertsch, S. Jung, A. Zerck, et al. „Optimal de novo design of MRM experiments for rapid assay development in targeted proteomics“. *J. Proteome Res.* **9**(5) (May 2010), pp. 2696–704.

[53] R. Bakry, C. W. Huck, M. Najam-ul-Haq, et al. „Recent advances in capillary electrophoresis for biomarker discovery“. *J Sep Sci* **30**(2) (Feb. 2007), pp. 192–201.

[54] H. Mischak, J. J. Coon, J. Novak, et al. „Capillary electrophoresis-mass spectrometry as a powerful tool in biomarker discovery and clinical diagnosis: an update of recent developments“. *Mass Spectrom Rev* **28**(5) (Sept. 2009), pp. 703–24.

[55] R. G. Cooks and A. L. Rockwood. „The ’Thomson’. A Suggested Unit for Mass Spectroscopists“. *Rapid Commun. Mass Spectrom.* **5**(2) (1991), pp. 92–93.

[56] A. D. McNaught and A. Wilkinson. *IUPAC Compendium of Chemical Terminology - the Gold Book*. 2nd ed. Blackwell Science Inc, 1997.

[57] T. W. Hutchens and T.-T. Yip. „New desorption strategies for the mass spectrometric analysis of macromolecules“. *Rapid Commun. Mass Spectrom.* **7**(7) (1993), pp. 576–580.

[58] J. V. Iribarne and B. A. Thomson. „On the evaporation of small ions from charged droplets“. *J. Chem. Phys.* **64**(6) (1976), pp. 2287–2294.

[59] M. Dole, L. L. Mack, R. L. Hines, et al. „Molecular Beams of Macroions“. *J. Chem. Phys.* **49**(5) (1968), pp. 2240–2249.

[60] M. Wilm. „Principles of electrospray ionization“. *Mol. Cell Proteomics* **10**(7) (July 2011), p. M111.009407.

[61] M. A. Kuzyk, L. B. Ohlund, M. H. Elliott, et al. „A comparison of MS/MS-based, stable-isotope-labeled, quantitation performance on ESI-quadrupole TOF and MALDI-TOF/TOF mass spectrometers“. *Proteomics* **9**(12) (June 2009), pp. 3328–3340.

[62] W. E. Stephens. „A Pulsed Mass Spectrometer with Time Dispersion“. *Phys. Rev.* **69** (1946), p. 691.

[63] M. L. Vestal. „Modern MALDI time-of-flight mass spectrometry“. *J Mass Spectrom* **44**(3) (Mar. 2009), pp. 303–17.

[64] I. V. Chernushevich, A. V. Loboda, and B. A. Thomson. „An introduction to quadrupole-time-of-flight mass spectrometry“. *J Mass Spectrom* **36**(8) (Aug. 2001), pp. 849–65.

[65] J. R. Yates, C. I. Ruse, and A. Nakorchevsky. „Proteomics by mass spectrometry: approaches, advances, and applications“. *Annu Rev Biomed Eng* **11** (2009), pp. 49–79.

[66] M. B. Comisarow and A. G. Marshall. „The early development of Fourier transform ion cyclotron resonance (FT-ICR) spectroscopy“. *J Mass Spectrom* **31**(6) (June 1996), pp. 581–5.

[67] Makarov. „Electrostatic axially harmonic orbital trapping: a high-performance technique of mass analysis“. *Anal. Chem.* **72**(6) (Mar. 2000), pp. 1156–62.

[68] A. Makarov, E. Denisov, O. Lange, et al. „Dynamic range of mass accuracy in LTQ Orbitrap hybrid mass spectrometer“. *J. Am. Soc. Mass Spectrom.* **17**(7) (July 2006), pp. 977–82.

[69] H. Kubinyi. „Calculation of isotope distributions in mass spectrometry. A trivial solution for a non-trivial problem“. *Anal. Chim. Acta* **247**(1) (1991), pp. 107–119.

[70] M. W. Senko, S. C. Beu, and F. W. McLafferty. „Determination of monoisotopic masses and ion populations for large biomolecules from resolved isotopic distributions“. *J. Am. Soc. Mass Spectrom.* **6**(4) (1995), pp. 229–233.

[71] C. Yang, Z. He, and W. Yu. „Comparison of public peak detection algorithms for MALDI mass spectrometry data analysis.“ *BMC Bioinformatics* **10** (2009), p. 4.

[72] J. Li. „Comparison of the capability of peak functions in describing real chromatographic peaks“. *J Chromatogr A* **952**(1-2) (Apr. 2002), pp. 63–70.

[73] D. J. Pappin, P. Hojrup, and A. J. Bleasby. „Rapid identification of proteins by peptide-mass fingerprinting.“ *Curr Biol* **3**(6) (June 1993), pp. 327–332.

[74] W. J. Henzel, T. M. Billeci, J. T. Stults, et al. „Identifying proteins from two-dimensional gels by molecular mass searching of peptide fragments in protein sequence databases“. *Proc. Natl. Acad. Sci. USA* **90**(11) (June 1993), pp. 5011–5.

[75] P. James, M. Quadroni, E. Carafoli, et al. „Protein identification by mass profile fingerprinting“. *Biochem. Biophys. Res. Commun.* **195**(1) (Aug. 1993), pp. 58–64.

[76] S. D. Patterson and R. Aebersold. „Mass spectrometric approaches for the identification of gel-separated proteins“. *Electrophoresis* **16**(10) (Oct. 1995), pp. 1791–814.

[77] J. R. Yates 3rd, S. Speicher, P. R. Griffin, et al. „Peptide mass maps: a highly informative approach to protein identification“. *Anal Biochem* **214**(2) (Nov. 1993), pp. 397–408.

[78] L. Pasa-Tolić, C. Masselon, R. C. Barry, et al. „Proteomic analyses using an accurate mass and time tag strategy“. *BioTechniques* **37**(4) (Oct. 2004), 621–4, 626–33, 636 passim.

[79] J. S. D. Zimmer, M. E. Monroe, W.-J. Qian, et al. „Advances in proteomics data analysis and display using an accurate mass and time tag approach". *Mass Spectrom Rev* **25**(3) (May 2006), pp. 450–82.

[80] J. V. Olsen, B. Macek, O. Lange, et al. „Higher-energy C-trap dissociation for peptide modification analysis". *Nat. Methods* **4**(9) (Sept. 2007), pp. 709–12.

[81] J. E. P. Syka, J. J. Coon, M. J. Schroeder, et al. „Peptide and protein sequence analysis by electron transfer dissociation mass spectrometry". *Proc. Natl. Acad. Sci. USA* **101**(26) (June 2004), pp. 9528–33.

[82] A. Frank and P. Pevzner. „PepNovo: de novo peptide sequencing via probabilistic network modeling". *Anal. Chem.* **77**(4) (Feb. 2005), pp. 964–73.

[83] S. Andreotti, G. W. Klau, and K. Reinert. „Antilope – A Lagrangian Relaxation Approach to the de novo Peptide Sequencing Problem". *IEEE/ACM Trans Comput Biol Bioinform* (Mar. 2012).

[84] B. Ma, K. Zhang, C. Hendrie, et al. „PEAKS: powerful software for peptide de novo sequencing by tandem mass spectrometry". *Rapid Commun. Mass Spectrom.* **17**(20) (2003), pp. 2337–42.

[85] D. N. Perkins, D. J. Pappin, D. M. Creasy, et al. „Probability-based protein identification by searching sequence databases using mass spectrometry data". *Electrophoresis* **20**(18) (Dec. 1999), pp. 3551–67.

[86] J. Eng, A. McCormack, and J. Yates. „An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database". *J. Am. Soc. Mass Spectrom.* **5**(11) (1994), pp. 976–989.

[87] L. Y. Geer, S. P. Markey, J. A. Kowalak, et al. „Open mass spectrometry search algorithm". *J. Proteome Res.* **3**(5) (Sept. 2004), pp. 958–64.

[88] R. Craig and R. C. Beavis. „A method for reducing the time required to match protein sequences with tandem mass spectra". *Rapid Commun. Mass Spectrom.* **17**(20) (2003), pp. 2310–6.

[89] S. Nahnsen, A. Bertsch, J. Rahnenführer, et al. „Probabilistic consensus scoring improves tandem mass spectrometry peptide identification". *J. Proteome Res.* **10**(8) (Aug. 2011), pp. 3332–43.

[90] A. Zerck, E. Nordhoff, A. Resemann, et al. „An iterative strategy for precursor ion selection for LC-MS/MS based shotgun proteomics". *J. Proteome Res.* **8**(7) (July 2009), pp. 3239–51.

[91] S.-E. Ong. „Stable Isotope Labeling by Amino Acids in Cell Culture, SILAC, as a Simple and Accurate Approach to Expression Proteomics“. *Mol. Cell Proteomics* **1**(5) (May 2002), pp. 376–386.

[92] S. P. Gygi, B. Rist, S. A. Gerber, et al. „Quantitative analysis of complex protein mixtures using isotope-coded affinity tags.“ *Nat. Biotechnol.* **17**(10) (Oct. 1999), pp. 994–999.

[93] A. Schmidt, J. Kellermann, and F. Lottspeich. „A novel strategy for quantitative proteomics using isotope-coded protein labels.“ *Proteomics* **5**(1) (Jan. 2005), pp. 4–15.

[94] O. A. Mirgorodskaya, Y. P. Kozmin, M. I. Titov, et al. „Quantitation of peptides and proteins by matrix-assisted laser desorption/ionization mass spectrometry using (18)O-labeled internal standards.“ *Rapid Commun. Mass Spectrom.* **14**(14) (Jan. 2000), pp. 1226–32.

[95] A. Thompson, J. Schäfer, K. Kuhn, et al. „Tandem mass tags: a novel quantification strategy for comparative analysis of complex protein mixtures by MS/MS“. *Anal. Chem.* **75**(8) (Apr. 2003), pp. 1895–904.

[96] P. L. Ross, Y. N. Huang, J. N. Marchese, et al. „Multiplexed protein quantitation in Saccharomyces cerevisiae using amine-reactive isobaric tagging reagents“. *Mol. Cell Proteomics* **3**(12) (Dec. 2004), pp. 1154–69.

[97] L. Dayon, A. Hainard, V. Licker, et al. „Relative quantification of proteins in human cerebrospinal fluids by MS/MS using 6-plex isobaric tags“. *Anal. Chem.* **80**(8) (Apr. 2008), pp. 2921–31.

[98] L. Choe, M. D'Ascenzo, N. R. Relkin, et al. „8-plex quantitation of changes in cerebrospinal fluid protein expression in subjects undergoing intravenous immunoglobulin treatment for Alzheimer's disease“. *Proteomics* **7**(20) (Oct. 2007), pp. 3651–60.

[99] L. N. Müller, O. Rinner, A. Schmidt, et al. „SuperHirn - a novel tool for high resolution LC-MS-based peptide/protein profiling“. *Proteomics* **7**(19) (Oct. 2007), pp. 3470–80.

[100] M. Bellew, M. Coram, M. Fitzgibbon, et al. „A suite of algorithms for the comprehensive analysis of complex protein mixtures using high-resolution LC-MS“. *Bioinformatics* **22**(15) (Aug. 2006), pp. 1902–9.

[101] E. Lange, R. Tautenhahn, S. Neumann, et al. „Critical assessment of alignment procedures for LC-MS proteomics and metabolomics measurements“. *BMC Bioinformatics* **9** (2008), p. 375.

[102] K. Kultima, A. Nilsson, B. Scholz, et al. „Development and evaluation of normalization methods for label-free relative quantification of endogenous peptides“. *Mol. Cell Proteomics* **8**(10) (Oct. 2009), pp. 2285–95.

[103] H. Liu, R. G. Sadygov, and J. R. Yates 3rd. „A model for random sampling and estimation of relative protein abundance in shotgun proteomics“. *Anal. Chem.* **76**(14) (July 2004), pp. 4193–201.

[104] Y. Ishihama, Y. Oda, T. Tabata, et al. „Exponentially modified protein abundance index (emPAI) for estimation of absolute protein amount in proteomics by the number of sequenced peptides per protein“. *Mol. Cell Proteomics* **4**(9) (Sept. 2005), pp. 1265–72.

[105] N. Colaert, K. Gevaert, and L. Martens. „RIBAR and xRIBAR: Methods for reproducible relative MS/MS-based label-free protein quantification“. *J. Proteome Res.* **10**(7) (July 2011), pp. 3183–9.

[106] S. Degroeve, A. Staes, P.-J. De Bock, et al. „The Effect of Peptide Identification Search Algorithms on MS2-Based Label-Free Protein Quantification“. *OMICS* **16**(9) (Sept. 2012), pp. 443–8.

[107] N. Colaert, J. Vandekerckhove, K. Gevaert, et al. „A comparison of MS2-based label-free quantitative proteomic techniques with regards to accuracy and precision“. *Proteomics* **11**(6) (Mar. 2011), pp. 1110–3.

[108] J. A. Cham (Mead), L. Bianco, and C. Bessant. „Free computational resources for designing selected reaction monitoring transitions“. *Proteomics* **10**(6) (Mar. 2010), pp. 1106–26.

[109] B. Domon and R. Aebersold. „Options and considerations when selecting a quantitative proteomics strategy“. *Nat. Biotechnol.* **28**(7) (July 2010), pp. 710–21.

[110] L. N. Müller, M.-Y. Brusniak, D. R. Mani, et al. „An assessment of software solutions for the analysis of mass spectrometry based quantitative proteomics data“. *J. Proteome Res.* **7**(1) (Jan. 2008), pp. 51–61.

[111] S. Cappadona, P. R. Baker, P. R. Cutillas, et al. „Current challenges in software solutions for mass spectrometry-based quantitative proteomics“. *Amino Acids* **43**(3) (Sept. 2012), pp. 1087–108.

[112] The Qt Project. *Qt*. `http://qt-project.org`. Oct. 2012.

[113] The Apache XML project. *Xerces-C++*. `http://xerces.apache.org/xerces-c/index.html`. Oct. 2012.

[114] J. Seward. *bzip2*. `http://www.bzip.org`. Oct. 2012.

[115] J.-l. Gailly and M. Adler. *zlib - A Massively Spiffy Yet Delicately Unobtrusive Compression Library*. `http://www.zlib.net`. Oct. 2012.

[116] *GNU Scientific Library Reference Manual - Third Edition*. Network Theory Ltd., 2009.

[117] *GNU Linear Programming Kit*. `http://www.gnu.org/software/glpk/`. Oct. 2012.

[118] A. Döring, D. Weese, T. Rausch, et al. „SeqAn an efficient, generic C++ library for sequence analysis“. *BMC Bioinformatics* **9** (2008), p. 11.

[119] L. Martens, M. Chambers, M. Sturm, et al. „mzML – a community standard for mass spectrometry data.“ *Mol. Cell Proteomics* **10**(1) (Jan. 2011), R110.000133.

[120] P. G. A. Pedrioli, J. K. Eng, R. Hubley, et al. „A common open representation of mass spectrometry data and its application to proteomics research.“ *Nat. Biotechnol.* **22**(11) (Nov. 2004), pp. 1459–1466.

[121] HUPO Proteomics Standards Initiative. *Mass Spectrometry Data Representation (mzData) 1.05*. `http://www.psidev.info/mzdata-1_0_5-docs`. 2005.

[122] HUPO Proteomics Standards Initiative. *mzIdentML 1.1.0 Specification*. `http://www.psidev.info/mzidentml`. Oct. 2012.

[123] M. Walzer, D. Qi, G. Mayer, et al. „The mzQuantML data standard for mass spectrometry-based quantitative studies in proteomics“. *Mol. Cell Proteomics* (Apr. 2013).

[124] O. Kohlbacher, K. Reinert, C. Gröpl, et al. „TOPP–the OpenMS proteomics pipeline.“ *Bioinformatics* **23**(2) (2007), e191–7.

[125] J. Junker, C. Bielow, A. Bertsch, et al. „TOPPAS: A Graphical Workflow Editor for the Analysis of High-Throughput Proteomics Data.“ *J. Proteome Res.* **11**(7) (2012), pp. 3914–3920.

[126] M. R. Berthold, N. Cebron, F. Dill, et al. „KNIME: The Konstanz Information Miner“. In: *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. Springer, 2007.

[127] M. Sturm and O. Kohlbacher. „TOPPView: an open-source viewer for mass spectrometry data“. *J. Proteome Res.* **8**(7) (July 2009), pp. 3760–3.

[128] M. Sturm. „OpenMS - A framework for computational mass spectrometry“. PhD thesis. Wilhelmstr. 32, 72074 Tübingen: Universität Tübingen, 2010.

[129] K. Martin and B. Hoffman. *Mastering CMake 4th Edition*. Kitware, Inc., 2008.

[130] J. D. Thompson, P. Koehl, R. Ripp, et al. „BAliBASE 3.0: latest developments of the multiple sequence alignment benchmark.“ *Proteins* **61**(1) (Oct. 2005), pp. 127–36.

[131] J. Allmer. „A Call for Benchmark Data in Mass Spectrometry-Based Proteomics“. *J Integr OMICS* (2012).

[132] K. R. Coombes, J. M. Koomen, K. A. Baggerly, et al. „Understanding the characteristics of mass spectrometry data through the use of simulation.“ *Cancer Inform* **1** (2005), pp. 41–52.

[133] J. S. Morris, K. R. Coombes, J. Koomen, et al. „Feature extraction and quantification for mass spectrometry in biomedical applications using the mean spectrum.“ *Bioinformatics* **21**(9) (2005), pp. 1764–1775.

[134] O. Schulz-Trieglaff, N. Pfeifer, C. Gröpl, et al. „LC-MSsim – a simulation software for liquid chromatography mass spectrometry data.“ *BMC Bioinformatics* **9** (2008), p. 423.

[135] B. Y. Renard, M. Kirchner, H. Steen, et al. „NITPICK: peak identification for mass spectrometry data.“ *BMC Bioinformatics* **9** (2008), p. 355.

[136] D. M. Creasy and J. S. Cottrell. „Unimod: Protein modifications for mass spectrometry.“ *Proteomics* **4**(6) (June 2004), pp. 1534–6.

[137] J. A. Siepen, E.-J. Keevil, D. Knight, et al. „Prediction of missed cleavage sites in tryptic peptides aids protein identification in proteomics.“ *J. Proteome Res.* **6**(1) (Jan. 2007), pp. 399–408.

[138] E. Lange, C. Gröpl, O. Schulz-Trieglaff, et al. „A geometric approach for the alignment of liquid chromatography-mass spectrometry data“. *Bioinformatics* **23**(13) (July 2007), pp. i273–81.

[139] G. M. M. Laughlin, J. A. Nolan, J. L. Lindahl, et al. „Pharmaceutical Drug Separations by HPCE: Practical Guidelines“. *J. Liq. Chromatogr. Relat. Technol.* **15**(6) (Apr. 1992), pp. 961–1021.

[140] K. Lan and J. W. Jorgenson. „A hybrid of exponential and gaussian functions as a simple model of asymmetric chromatographic peaks“. *J Chromatogr A* **915**(1-2) (2001), pp. 1–13.

[141] P. Mallick, M. Schirle, S. S. Chen, et al. „Computational prediction of proteotypic peptides for quantitative proteomics.“ *Nat. Biotechnol.* **25**(1) (Jan. 2007), pp. 125–131.

[142] R. Matthiesen, ed. *Mass Spectrometry Data Analysis in Proteomics (Methods in Molecular Biology)*. Humana Press, 2007, p. 336.

[143] J. E. Elias, F. D. Gibbons, O. D. King, et al. „Intensity-based protein identification by machine learning from a library of tandem mass spectra“. *Nat. Biotechnol.* **22**(2) (Feb. 2004), pp. 214–9.

[144] R. J. Arnold, N. Jayasankar, D. Aggarwal, et al. „A machine learning approach to predicting peptide fragmentation spectra“. *Pac Symp Biocomput* (2006), pp. 219–30.

[145] C. Zhou, L. D. Bowler, and J. Feng. „A machine learning approach to explore the spectra intensity pattern of peptides using tandem mass spectrometry data“. *BMC Bioinformatics* **9** (2008), p. 325.

[146] A. M. Frank. „Predicting intensity ranks of peptide fragment ions“. *J. Proteome Res.* **8**(5) (May 2009), pp. 2226–40.

[147] Z. Zhang. „Prediction of low-energy collision-induced dissociation spectra of peptides“. *Anal. Chem.* **76**(14) (July 2004), pp. 3908–22.

[148] M. Fleron, Y. Greffe, D. Musmeci, et al. „Novel post-digest isotope coded protein labeling method for phospho- and glycoproteome analysis.“ *J Proteomics* **73**(10) (Sept. 2010), pp. 1986–2005.

[149] A. Ramos-Fernández, D. López-Ferrer, and J. Vázquez. „Improved method for differential expression proteomics using trypsin-catalyzed 18O labeling with a correction for labeling efficiency.“ *Mol. Cell Proteomics* **6**(7) (July 2007), pp. 1274–86.

[150] J. Klimek, J. S. Eddes, L. Hohmann, et al. „The standard protein mix database: a diverse data set to assist in the production of improved Peptide and protein identification software tools.“ *J. Proteome Res.* **7**(1) (2008), pp. 96–103.

[151] D. K. Han, J. Eng, H. Zhou, et al. „Quantitative profiling of differentiation-induced microsomal proteins using isotope-coded affinity tags and mass spectrometry.“ *Nat. Biotechnol.* **19**(10) (Oct. 2001), pp. 946–51.

[152] X.-J. Li, H. Zhang, J. A. Ranish, et al. „Automated statistical analysis of protein abundance ratios from data generated by stable-isotope dilution and tandem mass spectrometry.“ *Anal. Chem.* **75**(23) (Dec. 2003), pp. 6648–57.

[153] E. W. Deutsch, L. Mendoza, D. Shteynberg, et al. „A guided tour of the Trans-Proteomic Pipeline.“ *Proteomics* **10**(6) (Mar. 2010), pp. 1150–9.

[154] M. R. Hoopmann, G. L. Finney, and M. J. MacCoss. „High-speed data reduction, feature detection, and MS/MS spectrum quality assessment of shotgun proteomics data sets using high-resolution mass spectrometry.“ *Anal. Chem.* **79**(15) (Aug. 2007), pp. 5620–32.

[155] O. A. R. Board. *The OpenMP API specification for parallel programming*. `http://openmp.org/wp/openmp-specifications/`. Oct. 2012.

[156] M. Enoksson, J. Li, M. M. Ivancic, et al. „Identification of proteolytic cleavage sites by quantitative proteomics“. *J. Proteome Res.* **6**(7) (July 2007), pp. 2850–8.

[157] T. Nakazawa, M. Yamaguchi, T.-A. Okamura, et al. „Terminal proteomics: N- and C-terminal analyses for high-fidelity identification of proteins using MS“. *Proteomics* **8**(4) (Feb. 2008), pp. 673–85.

[158] G. Xu, S. B. Y. Shin, and S. R. Jaffrey. „Global profiling of protease cleavage sites by chemoselective labeling of protein N-termini“. *Proc. Natl. Acad. Sci. USA* **106**(46) (Nov. 2009), pp. 19310–5.

[159] K. Gevaert, M. Goethals, L. Martens, et al. „Exploring proteomes and analyzing protein processing by mass spectrometric identification of sorted N-terminal peptides“. *Nat. Biotechnol.* **21**(5) (May 2003), pp. 566–9.

[160] S. A. Gerber, J. Rush, O. Stemman, et al. „Absolute quantification of proteins and phosphoproteins from cell lysates by tandem MS“. *Proc. Natl. Acad. Sci. USA* **100**(12) (June 2003), pp. 6940–5.

[161] D. S. Kirkpatrick, S. A. Gerber, and S. P. Gygi. „The absolute quantification strategy: a general procedure for the quantification of proteins and post-translational modifications“. *Methods* **35**(3) (Mar. 2005), pp. 265–73.

[162] A. Gambin and B. Kluge. „Modeling Proteolysis from Mass Spectrometry Proteomic Data“. *Fundamenta Informaticae* **103**(1-4) (2010), pp. 89–104.

[163] A. I. Nesvizhskii. „Protein identification by tandem mass spectrometry and sequence database searching.“ *Methods Mol. Biol.* **367** (2007), pp. 87–119.

[164] D. Gillespie. „Stochastic simulation of chemical kinetics“. *Annu. Rev. Phys. Chem.* **58** (2007), pp. 35–55.

[165] W. Materi and D. Wishart. „Computational systems biology in cancer: modeling methods and applications“. *Gene Regul Syst Bio* **1** (2007), p. 91.

[166] D. Machado, R. Costa, M. Rocha, et al. „Modeling formalisms in systems biology“. *AMB Express* **1**(1) (2011), pp. 1–14.

[167] D. Chelius and P. V. Bondarenko. „Quantitative Profiling of Proteins in Complex Mixtures Using Liquid Chromatography and Mass Spectrometry“. *J. Proteome Res.* **1**(4) (Aug. 2002), pp. 317–323.

[168] W. Wang, H. Zhou, H. Lin, et al. „Quantification of Proteins and Metabolites by Mass Spectrometry without Isotopic Labeling or Spiked Standards“. *Anal. Chem.* **75**(18) (Sept. 2003), pp. 4818–4826.

[169] T. Dierkes, M. Wade, U. Nowak, et al. *BioPARKIN - Biology-related Parameter Identification in Large Kinetic Networks*. Tech. rep. 11-15. Takustr.7, 14195 Berlin: ZIB, 2011.

[170] P. Deuflhard. *Newton Methods for Nonlinear Problems. Affine Invariance and Adaptive Algorithms.* 1st. Vol. 35. Springer Series in Computational Mathematics. Springer, 2004, p. 424.

[171] H. Bock, S. Körkel, E. Kostina, et al. „Robustness Aspects in Parameter Estimation, Optimal Design of Experiments and Optimal Control“. In: *Reactive Flows, Diffusion and Transport.* Ed. by W. Jäger, R. Rannacher, and J. Warnatz. Springer Berlin Heidelberg, 2007, pp. 117–146.

[172] T. Conrad, A. Leichtle, A. Hagehülsmann, et al. „Beating the Noise: New Statistical Methods for Detecting Signals in MALDI-TOF Spectra Below Noise Level“. In: *Computational Life Sciences II.* Ed. by M. R. Berthold, R. Glen, and I. Fischer. Vol. 4216. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2006, pp. 119–128.

[173] E. Lange, C. Gröpl, K. Reinert, et al. „High-accuracy peak picking of proteomics data using wavelet techniques“. *Pac Symp Biocomput* (2006), pp. 243–254.

[174] I. Foster and C. Kesselman, eds. *The Grid: Blueprint for a New Computing Infrastructure.* San Francisco: Morgan Kaufmann Publishers, 1999.

[175] W. Cirne, F. Brasileiro, N. Andrade, et al. „Labs of the World, Unite!!!“ *Journal of Grid Computing* **4** (3 2006), pp. 225–246.

[176] M. Litzkow, M. Livny, and M. Mutka. „Condor-a hunter of idle workstations“. In: *Distributed Computing Systems, 1988., 8th International Conference on.* IEEE. 1988, pp. 104–111.

[177] A. Freier, P. Karlton, and P. Kocher. *The Secure Sockets Layer (SSL) Protocol Version 3.0.* RFC 6101 (Historic). Internet Engineering Task Force, Aug. 2011.

[178] *Windows Management Instrumentation.* `http://msdn.microsoft.com/en-us/library/windows/desktop/aa394582`. Mar. 2012.

[179] R. C. Martin. „The Open-Closed Principle“. *C++ Report* **8** (Jan. 1996).

[180] B. Meyer. *Object-oriented software construction.* New York: Prentice-Hall, 1988.

[181] E. Gamma, R. Helm, R. Johnson, et al. *Design Patterns: Elements of Reusable Object-Oriented Software.* USA: Addison-Wesley Professional, 1994.

[182] The Apache Ant project. *Apache Ant.* `http://ant.apache.org/`. 2012.

[183] Apache Maven Project. *Apache Maven.* `http://maven.apache.org/`. 2012.

[184] L. Dusseault. *HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV).* RFC 4918 (Proposed Standard). Updated by RFC 5689. Internet Engineering Task Force, June 2007.

[185] B. Cohen. *BitTorrent - a new P2P app*. Yahoo eGroups `http://finance.groups.yahoo.com/group/decentralization/message/3160`. July 2001.

[186] R. Fielding, J. Gettys, J. Mogul, et al. *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616 (Draft Standard). Updated by RFCs 2817, 5785, 6266, 6585. Internet Engineering Task Force, June 1999.

[187] D. Eastlake 3rd and P. Jones. *US Secure Hash Algorithm 1 (SHA1)*. RFC 3174 (Informational). Updated by RFCs 4634, 6234. Internet Engineering Task Force, Sept. 2001.

[188] E. R. Gansner and S. C. North. „An open graph visualization system and its applications to software engineering“. *SOFTWARE - PRACTICE AND EXPERIENCE* **30**(11) (2000), pp. 1203–1233.

[189] J. Ellson, E. R. Gansner, E. Koutsofios, et al. „Graphviz and dynagraph – static and dynamic graph drawing tools“. In: *GRAPH DRAWING SOFTWARE*. Springer-Verlag, 2003, pp. 127–148.

[190] P. Du, G. Stolovitzky, P. Horvatovich, et al. „A noise model for mass spectrometry based proteomics“. *Bioinformatics* **24**(8) (Apr. 2008), pp. 1070–7.