MassAnalyzer,

a program to help find labeled peptides and compare them to their unlabeled counterparts in a SILAC experiment

By

Tommy Narrevik

27.11.2008

Master Thesis

Department of Bioinformatics

University of Bergen

Preface

Proteins are the work horses of every living organism, and they do most of the important work that keeps the organism alive. These proteins have their own shape and other characteristics mostly defined by their sequence and the individual amino acids in them. The wonder about these proteins are that they often cooperate in complex networks in order to be able to fulfill their work assignments.

It is easy to understand that scientists have taken such an interest in proteins and how these works. They do this by identification and quantification. Not long ago identification was used to understand one protein at a time. Now the complex networks of proteins are in focus, where up to hundreds of proteins are identified at once.

One method that helps identifying such massive amount of proteins at once are called mass spectrometry. This method measures the mass of small things that can't be seen by the naked eye, as in proteins. These measurements of masses can then be used to identify the sequence of the protein, and used to search large knowledge databases to identify their work place and assignment.

A scientist might want to investigate the difference between a healthy and an unhealthy organism. This can be done by adding a label to the proteins of the unhealthy organism, effectively increasing the mass of every protein. By analyzing both organisms' proteins it is possible to compare the results against each other, with a difference in numbers indicating a protein as a possible cause for the disease.

The MassAnalyzer is a program that helps scientists with mass spectrometry, by providing ways to analyze the measurements of masses. It also contains an important method, called Pair Finder, that lets the user find the labeled proteins and compare them to their normal counterparts.

I will take this opportunity to thank my guidance councilor Ingvar Eidhammer for his help making this thesis possible, and Gustavo Souza for letting me have it in the first place. Another thanks to both for their feedback and Gustavo Souza for supplying the control sets.

A thank you also goes to my good friend Tormod Haugen for helping me out by answering any technical question I threw his way, and to Harald Bartnes for helping out with programming questions. Another thanks goes to my friends and family for supporting me.

Table of Contents

1 Goal of this Thesis

A common method to find differentiation in protein expression is called stable-isotope labeling. The method is mostly used in mass spectrometry(MS), but as with most methods in chemistry it might, with a little creativeness, be transfered to other fields. SILAC(Stable Isotope Labeling by Amino Acids in Cells) is one such method. This method incorporates an isotope of leucine(label) into the protein of the cell while it grows. The labeled proteins(marker) can then be compared to the unlabeled molecules. Both of these are taken from a culture that has grown on normal leucine(chapter 4.1). Peaks in a mass spectrum matching a mass distance(with same charge and number of markers) that is the same as the mass difference between the marker and normal leucine, taking the charge into account, are called pairs. Usually the full proteins are not compared with each other, but the digestion product(peptides) of these two samples are. These pairs can then be identified with MS/MS.

1.1 Problems with Existing Procedures

Existing procedures for MS/MS usually lets the program pick the 3-8 most intense peaks from a MS spectrum(chapter 3.5). The sequences of these will be generated and checked with a search program(chapter 3.4). This procedure might result in some pairs if they are close to equal in intensities. On the other hand if one of the samples contain a heavily down-regulated protein this will most likely not be chosen for MS/MS. The pair will therefore be incomplete after MS/MS. Even though the pair can be found post hoc(in the mass spectrum) there is no way of being sure that it is a true pair without sequencing both of the peaks in the pair.

There exist other procedures for choosing peaks for an MS/MS. One of these are inclusion list with or without specified retention times. This method can only be used if the wanted peaks/pairs are known before the experiment is run.

1.2 Selecting Peaks for MS/MS

The ideal situation is that pairs can be run in MS/MS. The challenge will then be to find these pairs in the MS-run. More than one type of pair can be imagined to exist:

- Pairs that have the same number of markers and charge.
- Pairs that have different charge but the same number of markers.

Pairs that have different numbers of markers should not exist, since SILAC fully exchange leucine for the marker in the experiment, unless the experiment is not fully run out. The equation will be the same for both types of pairs:

$$\frac{M_{unmarked}+Z_u}{Z_u}=\frac{M_{marked}+Z_m}{Z_m}-\frac{n*m}{Z_m} \quad ,$$

where M_{marked} is the mass of the marked peptide, $M_{unmarked}$ is the mass of the peptide with normal leucine, Z_u is the charge for the unmarked, Z_m is the charge for the marked, n is the number of leucine in the marked molecule and m is the mass difference between the marked and normal leucine.

True pairs will be the pairs that satisfy the equation and also have the same sequence, while the

ones not satisfying the latter will only be potential pairs.

The charge and numbers of markers are unknown in this experiment, so all the possible combinations of n and Z have to be tested out. This will result in a lot of possible pairs and the second challenge will be to identify the true pairs.

For this thesis the goal is to identify the first type of pairs, since theoretically SILAC will produce both peaks for every pair(chapter 4.1). The last part of the equation, the difference between marked and unmarked, will therefore be used to find pairs, since Z_m and Z_u will be the same. This thesis should at least solve the challenge with a single mass spectrum in mind.

1.3 Implementing the Solution in a Running Environment

The best way would be to directly feed the found pairs into the program that records and control the MS/MS peak selection. Since this program, MassLynx, has its own proprietary data format and is hard to manipulate through modules(program that changes behavior of another program) the second best way will have to be taken. This entails making a program that do a post analysis(runs off-line) and creates an inclusion list that can be used in a later MS/MS run.

1.4 Other Goals

The program should be as easy as possible to interact with, while also being relative easy to maintain.

2 Background: Bioinformatics and Biology

To understand the need for bioinformatics one must understand the need for knowledge of the biological diversity on earth.

Since the dawn of civilization some people have been interested in how humans functions, first and foremost to find cures or to explain some disease(small pox being the most prominent). In early times diseases were often seen as punishments from the gods/god and later on as instabilities in the 4 humours, as explained by the Greek philosopher Hippocratus in the 5th century BC. This belief continued into late 15th century. The new postulation after this became that diseases could come from outside the body. In the 18th century variolization, vaccination of small pox, was documented as used for the first time, but it wasn't before the 1870s when Louis Pasture and collaborates discovered that a victim would become immune to a lethal bacterial infection if they first were attacked by a mild strain of the bacteria and survived[1]. The knowledge of diseases have increased since then and the survivability of a before lethal disease is quite high. Some diseases are also believed to be eradicated.

In the last century a lot of advances were made to make bioinformatics needed. Discovering DNA, or as it was called at that time "nucleic", was done by Miescher in 1869 and strengthened by Altman in 1889 [2]. The discovery that DNA is the source of genes, and not proteins as most scientist thought at that time, wasn't done before 1944 by Avery and even he was shocked by the result [3]. Although this discovery wasn't taken to heart by most scientists before the most prominent discovery for biochemistry was made, by Watson and Crick in 1953, of how DNA forms a double helical structure [4, 5]. In the period 1950-1963 many scientists helped elucidate how the cellular machinery goes from DNA to protein, which is well explained by Watson in his article "Involvement of RNA in the Synthesis of Proteins" in 1963 [6]. It would take another 14 years before Sanger invented his method to sequence DNA [7]. As advancement in technology progressed automatic sequencing of DNA became possible in 1986. This and the development of better method for sequencing resulted in that the first bacteria was sequenced(*Haemophilus influenzae*) in 1995, the first insect (*Drosophila melanogaster*, fruit fly) in 2000 and the first "draft" of the human genome in 2001 [8].

The discovery of protein, the other side of bioinformatics, was done in the 19th century [9]. Pauling and Corey discovered the secondary structure alpha-helix of the protein alpha-keratin in 1951[10], while Sanger invented the first way to sequence proteins (Insulin) in 1945-55. The method for sequencing was not improved until the late 1960s when Edman published the method now called Edman degradation reaction [11].

X-ray(Rontgen in 1895) crystallography has been a crucial factor to the discovery of the properties of DNA and proteins[12].

Even if computers allowed some scientists to classify their proteins on computers as early as the 1960 [11]. It's not before the late 1990s these methods became widespread, as supercomputer became more common, internet became a source of knowledge, computers became more common among people and schools, and of course automatic gene/genome sequencing became more common. All of the points mentioned above made programs like BLASTx (Basic Local Alignment Search Tool), FASTx, CLUSTAL(global alignment by using guide a tree (computer constructed

phylogenetic tree)), DALI(structure alignment) etc. possible. The common ground for these programs are that they are used in conjunction with huge databases containing protein/DNA data, making it possible to compare vast amount of data [13]. Bioinformatics are the production of these kind of programs, or more to the point the usage of programming, mathematics(for algorithms) and statistics(for probabilities that the results are correct) to solve biological problems. Bioinformatics work closely with its two related fields called genomics where the DNA is in focus and proteomics where the protein is in focus. Both fields collect data faster than it is possible to analyse, since analysis (characterize, identify function etc.) is still done in the laboratories. Although a coarse analysis can be done through computer programs.

2.1 Proteomics

Proteomics are the study of the proteome, which is defined as the proteins contained in an organism. The proteome are often divided further into two groups, being proteins at a specific location like tissues and specific condition like cell division[14]. The first part is especially true for eukaryotes, while procaryotes usually do not have any compartments or any form of specialization within a "colony" of cells[15]. The overall goal to study proteomes is to understand the function of all the proteins in an organism. In these days this is done by registering proteins to a database. GO(Gene Ontology) has created a controlled vocabulary and a database that uses it. They characterize proteins using the protein's molecular function, biological process and cellular component. GO does not include metabolic pathways, regulatory networks etc. , or super-cellular structures like the skin, organs or any other body parts. Other ontologies complement GO here and might include one or more of these characteristics[14]. Today proteomics are said to have four cornerstones, which are:

- ***Protein identification***, aims to identify the proteins in a sample. This is usually done in one of two ways. One being to identify by sequence and the other one is to measure many different properties of the proteins in the sample so that their true identity can be evaluated with a low margin of error.
- ***Protein characterization***, aims to characterize the proteins by their biophysical and/or biochemical properties with no regards to if the protein has been identified or not.
- ***Protein quantification***, aims to set a quantity to each of the proteins in a sample, either as relative or absolute values.
- ***Protein sample comparison***, aims to compare the proteins in a sample with one or more other samples. There are more than one aspect to this, and they are:
    - relative occurrence, the presence of a protein in one of the sample while not in the other samples.
    - relative abundance, the presence of a protein in abundance but with differing values in the varying samples.
    - differential modification, the presence of different modified forms of the same protein in different samples.

Proteomics got an edge on transcriptomics(gene transcription in a cell) since the amount of mRNA does not necessarily show the real level of the proteins in the cell, because of down regulation, degradation, different splice variants (eukaryotes) etc. [16]. Proteomics therefore lends itself to medicine through expression profiles, the amount of different proteins in a cell compared to healthy cells, to solve the enigma of some diseases. Only proteins that are linked to the disease should be in the profile. These proteins are often called biomarkers. Other molecules and phenomena(like elevated blood pressure etc.) linked to a disease are also called biomarkers. To generate these profiles MS(mass spectrometry), protein microarray, 2D SDS-PAGE analysis are common. Although the latter has bad resolution(hard to find the right proteins in the gel)[17, 18]. Cancer is

one example of a disease there have been gained insight into through proteomics.

Other application for proteomics are within neuropathology, cardiovascular disease and microbiology(*E.Coli's* proteome has been studied) [18].

2.1.1 What is a Protein

Most of the contents of this chapter is taken from reference 19, where it is not a new reference is given.

Proteins are translated from mRNA, with the help of tRNA, which are transcribed from the DNA of the cells(Fig. 2.1). They are the main work horses of the cell, functioning as enzymes(catalyst for specific reactions), building material (keratin), signal molecules, membrane transporters, DNA replication(DNA polymerase) etc. Both its structure and characteristics are important for it to function properly.

Figure 2.1: Simple drawing of the way from DNA to protein.

2.1.1.1 Protein Structure

A protein are built up of 20 different amino acids. These have a common structure with the only difference appearing in their side chains(Fig. 2.2, full list in appendix A). The amino acids have names like Alanine, Proline etc, but also go by a three letter abbreviate (Ala, Pro etc.), or a one letter abbreviation (A, P etc.).

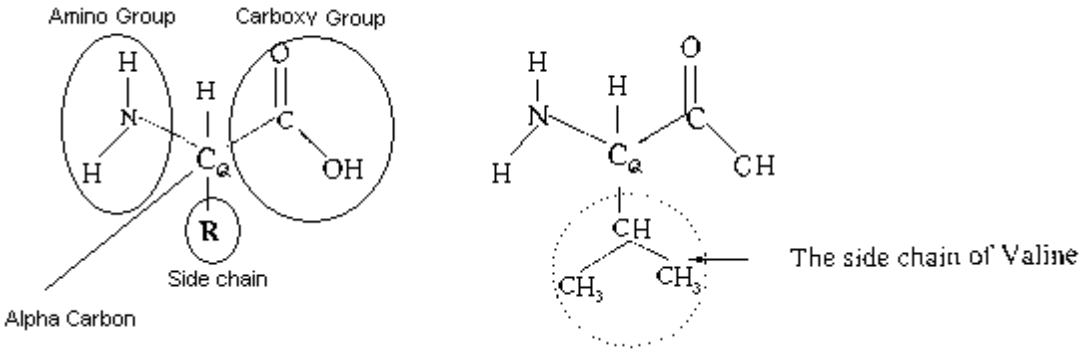


Figure 2.2: Left is a sketch of an amino acid showing its amino group, carboxy group and side chain(R) circled in. To the right the side chain for valine is shown instead of the generic R for side chain. Taken from [14].

The primary structure of the protein is that its amino acids are linked together by peptide bonds forming a polypeptide chain(Fig.2.3). The ends of the polypetide are called C-terminal (carboxyl group) and N-terminal(free amino group). The amino acids on the polypeptide chain can rotate around two internal bonds, called Φ and Ψ, allowing the whole polypeptide chain some flexibility(Fig. 2.4). Because of steric hindrances the rotation of the two bonds are not free. This is shown in the Ramachardian plot in the right part of figure 2.4.

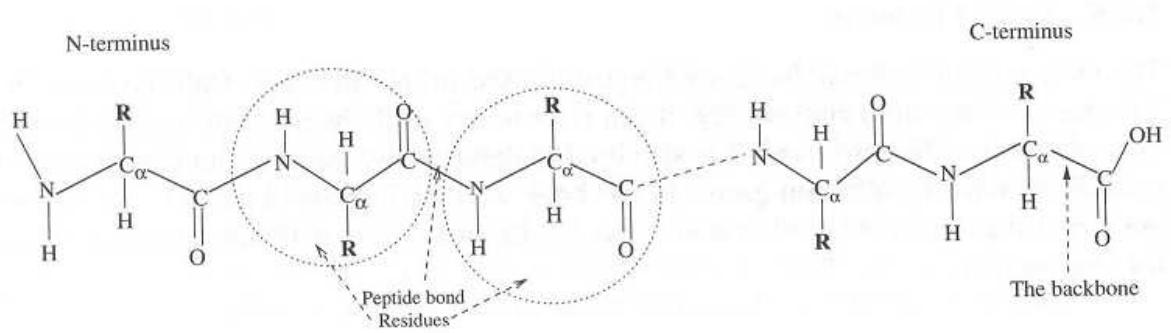


Figure 2.3: This shows a short polypepetide chain, where the N-terminus and C-terminus are marked. As well as the amion acids, circled in, forming a peptide bond which constitute a whole polypeptide when they are many. The whole chain, from residue to residue, from the N-terminus through every amino acids alpha Carbon forms to the C-terminus forms what is known as a polypeptides backbone. Taken from [14].

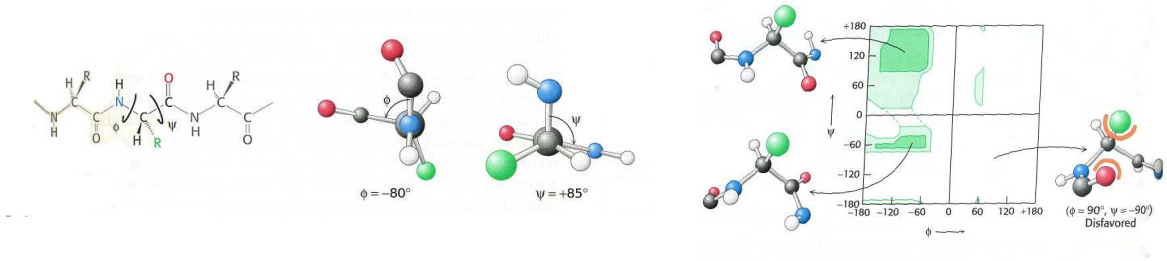


Figure 2.4: To the left is a polypeptidechain showing the torison angles Φ an d Ψ. In the middle this is shown with stick models. On the right side a ramachandran plot over the most likely angles are shown in green. The stick figures color coding are as follow, green=side chain, red = oxygen, grey=carbon and white=hydrogen. Taken from [19].

The secondary structure of the protein is the ability of the polypeptide chain to fold itself. This is a result of the intra-bond rotation in an amino acid and the amino acid's different characteristics/steric clashes(as a consequence of their side chain). These abilities makes the amino acid sequence the main factor to how a protein will fold, since different amino acids favors different conformational structures(because of the abilities mentioned above)[appendix A]. Although the sequence is the main factor, the main driving force for the actual folding is the hydrogen bonds between the backbones of one or more parts of the polypeptide chain. α-helix, β-sheets, turns(often called Ω-turns) and loops(Fig. 2.5) are all names for common secondary structures.

The tertiary structure of a protein is the ability of one protein to fold into a more complex structure with different amount of the different secondary structures. This folding is cooperative and depends mostly on the chemical characteristics of the amino acid side chains present in the protein compared to the natural milieu of the protein. This driving force is also called free energy. An example of protein folding are membrane(lipid(fatty acid) bilayer surrounding a cell) proteins that tends to have a hydrophobe outer layer while the core is hydrophobic. Proteins in water solution, called globular proteins, tends to follow the opposite formula. Steric hindrances also makes its role by making some tertiary structures impossible to attain for certain proteins, since logic implies that two side chains can not occupy the same space. On top of all this, proteins called chaperons helps some proteins to fold into the right structures, which often is the most favored energy wise(lowest free energy).

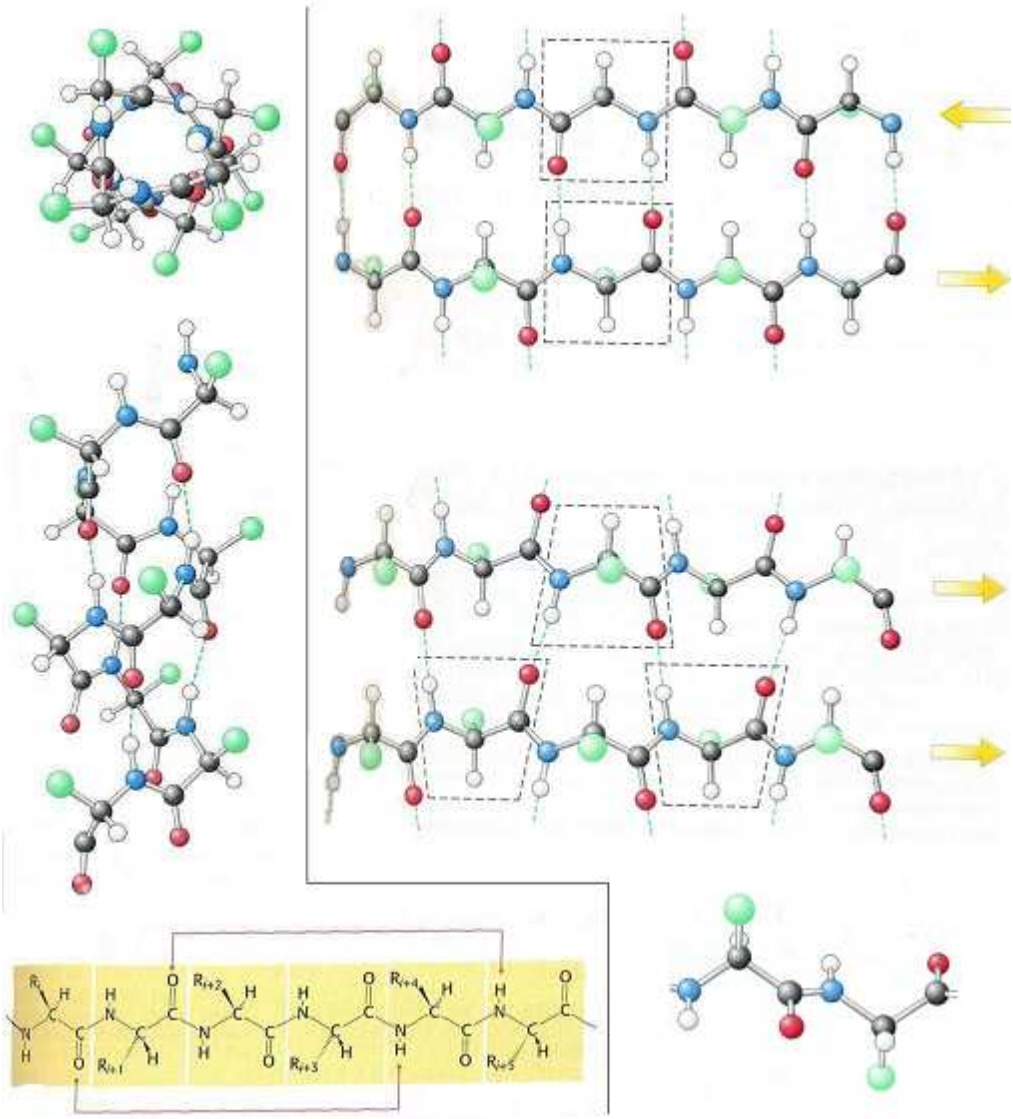


Figure 2.5: On the top and middle left side an α -helix are shown from above and from the side. On the bottom left side the hydrogen bonding between the amino acids in the α-helix are shown. The right side shows β-sheet in anti-paralel strands on the top, paralel strands in the middle and a sideview showing the side chains of the amino acids alternating between being in the plane and out of the plane on the bottom. The stick figures color coding are as follow, green=side chain, red = oxygen, grey=carbon and white=hydrogen. The green stippled lines are hydrogen bonds. Taken from [19].

The quaternary structure of a protein is the ability of different smaller proteins to form protein complexes, where each protein function as a subunit of this complex. An example of this is Hemoglobin with its two α units/protein and two β units/protein forming what is called a tetramer(4unit complex). Another example is the Cro protein of bacteriophage λ(virus that infects bacteria) that is a dimer(2unit complex).

2.1.1.2 Protein Characteristics

Most of the contents in this chapter is taken from reference 14, where it is not a new reference is given.

Every amino acid in a protein contribute to the characteristics of a protein, even the sequence they have can contribute to some of the characteristics. These characteristics as in all chemistry are used to either separate, identify or quantify the chemical in question.

Mass

The definition of the mass(m) of an atom/molecule is 1/12 the mass of carbon-12 isotope* and is called unified atomic mass unit(u), or the more common term Dalton(Da). Another term often used for «mass» is molecular weight which also uses carbon-12 as a template but adds gravity to it (gravity is dependent on location on earth/moon etc). IUPAC, 3rd draft, has the following definitions for measuring mass:

- Exact mass, the calculated mass of an ion or molecule containing a single type of isotope(usually the lightest one).
- Monoisotopic mass, the exact mass of an ion or molecule using the most abundant isotope.
- Average mass, the calculated mass of an ion or molecule using the average mass of the isotopes.
- Nominal mass, same as monoisotopic mass but rounded down to the nearest integer. Also the same as the atom numbers of each constituent added together.
- Mass defect, the difference between the mass number and the monoisotopic mass.
- Mass excess, is the negative of mass defect.
- Accurate mass, experimentally determined mass of an ion that is used to determine the elemental formula.
- Mole, a common measurement in chemistry meaning the mass of 6.022*10^22 (Avogadro's number) molecules or atoms.

To measure the mass SDS-PAGE are most commonly used, but mass spectrometry(MS) can also be used.

Isoelectric point(pI)

The isoelectric point(pI) is closely related to pH which is defined as the negative of the logarithm of the molar hydronium-ion concentration[20]. Water itself is often used as a defining point for pH. Clean water have a pH of 7, which means it has free hydronium-ions in the solution. The reason for this is its ability to auto-ionize, given by the formula $H_20(l) \leftrightarrow H^+(aq) + OH^-(aq)$. The equilibrium constant for this formula is $1*10^{-14}$, which gives $H^+(H_3O^+$, hydronium-ion) the constant of $1.0*10^{-7}$ which equals to a pH of 7. The pI uses the same scale and same principle as pH with the addendum that the molecule has to be neutral compared to the amount of negative and positive charges in the molecule. An example is a molecule of pI 4.0 which will be neutral, compared to charges, at pH 4.0. The negative and positive charges in a protein is mainly due to the side chains of its amino acids. Aspartic acid(acid), Histidine(acid), Cysteine(base), Arginine(base) etc. are good example of this. The pI can be calculated(see [14] page 11 for a deeper understanding of this), or it can be found experimentally by using SDS-PAGE with pH gradient, or liquid chromatography(LC).

In mass spectrometry pI is mainly used to separate peptides/proteins. This is usually done by 2D SDS-PAGE or LC.

*** Every element in nature has isotopes which means they have a mixture of forms which they can naturally occur as. These different forms usually mean an element have more neutrons(part of the nuclear in an atom) than the «ordinary» amount, while retaining all of the element's properties except the mass and radioactivity.**

Hydrophobicity

In the world of chemistry water is a polar molecule, meaning it has an end with a positive charge and one with a negative charge(Fig. 2.6). This gives it the ability to surround charged molecules to stabilize them(Fig. 2.6). Molecules without a charge can be soluble in water if they can support hydrogen bonding, like methanol (OH group is good at hydrogen bonding). If a molecule can not support hydrogen bonding the molecule wont be 'attracted' to water and the usually weak intra molecular bonds will be stronger. This kind of compounds are therefore coined hydro-phobic[20], «shy» of water. Crude oil is a good example, when put into the water it will lay as a thin film on top of it. Shaking the solution will mix it but with time the oil will again be on top of the water as a film. Compounds can have different degrees of hydrophobicity, which is often seen in longer organic molecules like proteins and oils. Temperature and pressure, to a lesser extent, can and will affect the solubility of a compound in a positive or negative way depending on characteristics of this very compound.

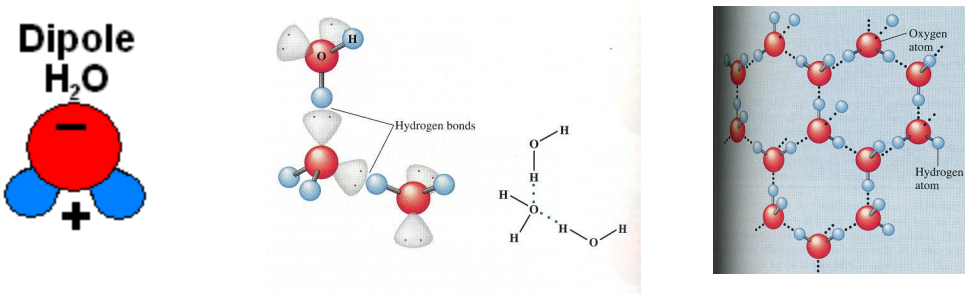


Figure 2.6: The left side shows the dipole of the water molecule. Even if water itself is neutral the two ends of it has a positive and negative attractive force to it. The middle shows a stick figure with hydrogen bonds between two water molecules. Here the electrons in the O-H bond is attracted to the oxygen atom leaving its positively charged proton partially exposed. On the right side this hydrogen bonding is shown on a larger scale, and the water matrix can be seen clearly. In the middle the same hydrogen bonding are shown as a formula with dotted lines as the hydrogen bonds. The figure's color coding are as follow, red=oxygen and blue=hydrogen. Middle and right taken from [20].

Hydrophobicity can both be measured, and calculated using a scale for hydrophobicity, like Kyte-Doolittle(appendix B), and then use GRAVY(GRand Averages of hYdrophobicity) or a sliding window.

In mass spectrometry hydrophobicity is most often used as a separating factor. An example is reverse column chromatography in HPLC(high pressure liquid chromatography).

2.1.2 Post Translational Modification

If proteins did not get modified after translation proteomics would be less important as a science, and genomics would be the easy way out. This is not the case and genomics can not give all the answers to how proteins and cells works. The post-translational modification(PTM) is multiple and varied depending on different conditions of the cell, as well as different tissues in a multiple cell organism which most eukaryotes are(not counting protezans). To mention some of the PTMs that occurs in most organisms, be it prokaryote or eukaryote:

- Phosphorylation (adds a phosphor(PO_3^{2-}) to a threonine or serine residue), often used to make a protein fold into a more active form or deactivate a protein (by having it fold into a less active form). The process uses a group of enzymes called kinases which uses ATP(the cells energy currency to do its work). In prokaryotes it is used for signal transduction by phosphorylating histidine or aspartate[21], this ancient signal pathway has been found in eukaryotes to[22].

- Dephosporylation (removes a phosphor(PO_3^{2-}) from a threonine or serine residue), regulates like phosphorylation does, but uses the enzyme group phosphatase to do the work.
- Acetylation, a means of regulation of, at least, the histones which are the proteins assisting with DNA packaging[19].
- Glycosylation, is the most enigmatic of the PTM, but some uses are as antigens(human blood type A, B and O)[19].
- Ligand binding, are components that binds to a protein to make it active, like Na^+ to some membrane transporters[19].
- Cleavage of protein, are used to cleave away sorting signals or to activate a protein by removing an inhibitory sequence from the protein.

Slicing is not exactly a PTM but needs to be mentioned, and has only been discovered to occur in eukaryotes. Slicing happens before translation and after transcription and is a way for eukaryotes to diversify their proteins by modifying/cleaving their mRNA. Meaning that the same mRNA can be cleaved in different ways so that it can be used for different but fairly similar protein processes.

In eukaryote organisms you also have the occurrence of alleles, similar DNA sequence but with some modification compared to each other. This is also called polymorphism. Single nucleotide polymorphism(SNP) if its a single base that has been changed. Transcribed genes, from alleles, usually behave the same in the end, but they can also behave differently and even lead to disease.

In mass spectrometry PTMs makes it harder to identify proteins/peptides since the added mass of the usually unknown PTM can mask the protein/peptide as another of its kind.

2.1.3 Methods used within Proteomics

The following sub chapters will give a short introduction to how proteins characteristics can be found, as well as how this data is stored.

2.1.3.1 Databases

The most central part to proteomics are the database, these contains thousands of thousands of protein sequences with data about different characteristics depending on the database.

UniProt

UniProt is one such database that base itself on three well established databases called Swiss-Prot, TrEMBL(Translated EMBL) and PIR(protein information resource). Swiss-Prot and PIR are manually curated whileTrEMBL on the other hand are automatically curated. Taking a better look at Swiss-Prot reveals that it contains annotations that are trustworthy, meaning that during the curation they are checked against well established data and linked to this data. This result in a stable and a well cross-linked database. TrEMBL on the other hand contains automatic annotations and predicted sequences which complements Swiss-Prot well.

Swiss-Prot's data contains accession number, sequence, predicted modifications and a feature table. The sequences contained within Swiss-Prot are after translation but before modification. The feature table contains the differences between, possible, multiple reports of the same protein.

These features are:

- Conflict, different amino acids for same positions are reported in the bibliography.
- Variants, where the sources reports that alleles exists.
- Mutagen, positions where alterations of amino acids has been performed experimentally.
- VarSplic, sequences that results of variant splicing.
- PTM, post translational modification with its position and type of modification.

Apart from the UniProt-knowledge base the systems also contains UniParc(Uniprot archive) and UniRef(a reference structure for different sequence identity levels).

Other Databases

Other protein databases are NCBI non-redundant database contains sequence of proteins that does not share a sequence. This database collects its data from amongst Swiss-Prot, TrEMBL and RefSeq etc. Another database is IPI(the international protein index) which is a non-redundant database born during the human genome project. It now also includes other organisms like mouse, rats and *Arabidopsis*. Since the database collects its data from other databases including UniProt, Ensembl and RefSeq it uses an algorithm to «choose» what gets into the database. This algorithm goes one step beyond non-redundancy and clusters sequences with more than 95% similarities, where it uses the longest sequence as the master sequence.

Other databases like CATH (Class, Architecture, Topology, Homologous superfamily), SCOP (Structural Classification of Proteins Database) and FSSP-DaliDD are all databases used for protein structure comparison. The data is here amongst other sorted by the total secondary structures in a protein. So a protein containing only α-helixes will be sorted under α-helixes. Appendix C Shows a table how these databases classify the proteins.

Time Instability of Sequence Databases

Many of the protein databases takes their data from multiple other protein databases. These databases are said to be time instable. This happens when one database updates its contents by deleting, adding or changing some of their sequences, which results in that databases using data from this one will display their data «incorrectly». The only way to remedy this is to partly or fully rebuild the database. If only links are used into another database the problem becomes less problematic, since these can easily be updated.

2.1.3.2 Tools of Proteomic

Experimentally Finding the Correct Mass of a Protein

For a long time SDS-PAGE (sodiumdodecylsulphate-polyacrylamidegelelectrophorese), which is explained better in chapter 3.3, was used to experimentally determine the mass of a protein. This is a limited method where the mass is easily inaccurate and the quantity of the spots on a gel are hard to read, but programs that do a good analysis from a gel exists. SDS-PAGE can also be used to determine the pI of a protein.

In 1984 mass spectrometry(MS) came into play with ESI, although with its limits(like having a hard time to identify to small or to large proteins, difficult to assess the quantity of a protein and its low reproducability of an experiment) it is a fast throughput device (explained better in chapter 3.3) and will most probably take over a lot of the jobs that SDS-PAGE do.

Protein Sequencing

Quite some time ago Edman found a technique to sequence proteins which got the name Edman's degradation. This technique sequences proteins up to a length of 60 amino acids by removing them one by one from the N-Terminal end of the polypeptide, and then identifying them. This is time consuming and requires about a days work.

In the later years an instrument called MS/MS(two mass spectrometry experiments after each other) has gotten into the field. It is able to sequence/partly sequence a protein that has been digested to relative small peptide(10-20 average length). These peptides can, with high probability, be used to identify the whole protein from a protein database. MS/MS suffers from the same as MS.

Identifying Proteins in a Solution

Methods like Western Blotting has been used for protein identification, of proteins of small abundance in solution, for a long time. The results from an SDS-PAGE is transferred to a polymer sheet by electroblotting. Here the proteins are more accessible for reaction. An antigen specific to the wanted protein is added to the sheet and binds to the protein if it is present.

MS/MS is also very sensitive to proteins of small abundance and can prove an excellent instrument for this kind of identification. For proteins in larger abundance ordinary SDS-PAGE can be used. Another method gaining a lot of ground within identification are microarrays[23]. These are small chips containing antibodies(for protein identification) and cDNA or DNA oligo-nucleotides(for DNA). Each chip can identify the presence of 100, if not 1000s of proteins in one go.

Characterization of Proteins

Characterizations of proteins are very individual, but usually contains methods to test out catalytic activity, strength of ligand binding etc. for enzymes. Another more indirect method is to manipulate the gene that the protein is coded from. One such method is called knock-out, which basically mutates a gene so that it either becomes unusable or do not get expressed at all. These are actually two different things, since an expressed gene that is not used for anything can still clog up the fine machinery of a cell in many ways. Knock-out study are often used to find out if an organism can live without the gene and what effects the knocked-out gene has on the organism.

The last to be mentioned method is cloning, which means copying a gene from one organism and put it into another. Hopefully the gene is related to the one copied into or this one may lack some of the needed cell machinery for the gene to work properly.

For cloning the laboratory workhorse *E.Coli* is often used, if not for more than a middle stage. As always in chemistry there are a multitude of more methods, some good some bad, and even more is invented as time pass.

3 Background: Mass Spectrometry(MS) Analysis

MS stands for mass spectrometry. The history of MS starts as early as 1886 when Eugen Goldstein discovers canal rays. Wilhelm Wien demonstrated that canal rays can be deflected with magnetic fields in 1898 and found out that the positive rays could be measured in a charge to mass ratio. J.J. Thomson improved Wien's work and measured the mass to charge ratio of electrons in 1898. In 1913 he was able to separate particles with different mass to charge ratio. On the way to the modern mass-spectrometer, TOF(time of flight) got invented around 1946 by W. Stephens and M. Dole developed ESI (Electron spray ionization) in 1968. J. Fenn and coworkers used ESI to ionize biomolecules in 1984. In 1985 MALDI(matrix-assisted laser desorption ionization) were described by F. Hillenkamp, M. Karas and coworkers [24, 25].

The basis for modern MS is ionization of molecules, in gas phase, followed by an analysis of the molecules where each M/Z value is detected. Here M means mass and Z means charge of the molecule. These three parts are often divided into different «instruments» coupled together, more on this in chapter 3.2. An important note about MS is that the molecules that is ionized compete to be ionized, meaning that two consecutive experiments on the same sample might yield different results. In other words the reproducibility is low[14].

Most of the following sub chapters are taken from the book Computational Methods for Mass Spectrometry Proteomics[14], while where it is not there is a reference to the article in question.

3.1 Analysis Set-up

MS experiments within proteomics are usually based on two approaches, the top-down and the bottom-up approach. The difference between these two are that the top-down is MS on intact proteins while the bottom-up digest the proteins first, into peptides, and does an MS on the peptides. The latter is the most used approach for more than one reason which is mentioned below:

- The absolute error in the measurement increases with increased M/Z value
- Due to a proteins atomic building blocks having isotopes, see 3.2.2.2, a protein wont have one mass but a range of masses. Peptides are small molecules and therefore they will contain fewer building blocks, ergo the mass range will be smaller for peptides.
- The sensitivity is higher for peptides than intact proteins.
- Some proteins are impossible to measure as in very large proteins, very hydrophobic proteins etc.
- Intact proteins may contain many post-translational modifications(PTM) which complicates the analysis. Peptides on the other hand often contains none or up to a few PTMs since they are relative short molecules.

In bottom-up protein analysis mass-fingerprinting(MFP) and MS/MS are two methods. In MFP proteins are separated before digested into peptides. The masses of these peptides are then used to search against a protein database(chapter 3.5). More than one protein can share a secondary characteristic like pI (isoelectric point), which means the analysis is often done on more than one protein. The set-up for mass fingerprinting is shown below, in figure 3.1.

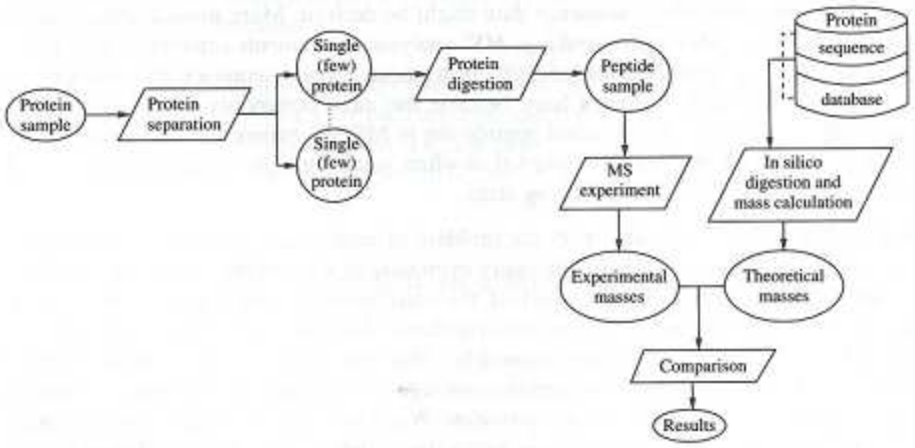


Figure 3.1: This shows the step by step setup for a mass-fingerprinting pipeline. Taken from [14].

In MS-MS the proteins are digested into peptides before separated(based on a characteristic), followed by an MS and another MS on a specific M/Z interval. Between the first and second MS some of the stronger peaks, one peak represent one or more peptides, are chosen and fragmented into even smaller parts. These fragments produces a result that can be used to find part or the whole sequence of the selected peptides(chapter 3.6). These sequences can then be used in a database(chapter 3.5) search and the coverage, how many sequences matches a specific protein, of the database proteins can be found. The higher the coverage the better the chance that the chosen peptides belongs to that protein. MS/MS follows the set-up shown in figure 3.2.

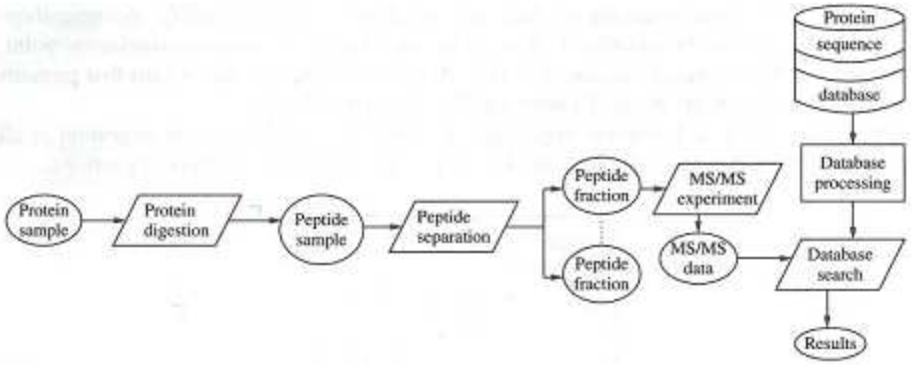


Figure 3.2: This shows the step by step setup for a MS-MS pipeline. Taken from [14].

3.1.1 Pre MS Methods

In the previous sub-chapter two different approaches for bottom-up MS analysis were mentioned and figure 3.1 and 3.2 shows the set-up for these. Both of these contains the steps separation and fragmentation/digestion. These two will be explained better below.

3.1.1.1 Fragmentation/Digestion

Fragmentation/digestion means breaking/cutting a molecule into smaller parts. Two ways to do this will be mentioned here. First one is fragmentation which could be imagined using a sledgehammer on glass, while the second one is digestion which can be imagined using a scissor(that could cut glass) on the glass.

Fragmentation is used if where the breakage occurs does not matter, or if the breakage of the bond have to happen fast (like in the second step of MS/MS). Even fragmentation can be partly 'controlled' by using a fragmentation source that favours one or more type(s) of fragments over others.

Digestion on the other hand is done by enzymes called proteases and takes more time than fragmentation, but the scientist is rewarded, to a degree, with a more specific cut/break point. These proteases occurs naturally in cells where they have chores like cleaving proteins for consumption/ rebuilding, removal of miss-folded proteins, deactivation/activation of proteins(ADAM-17(tumour necrosis factor-α conertase[26])), cell apoptosis(cell death, Caspases[27]) etc. Every protease has its own specific cleavage site that consists of one or more sub sites. The way the protease recognise these sub sites is by the shape, size and/or charge of the amino acid that goes into the specific sub site. Between two of these sub sites the cleavage point, also called scissile bond, is situated. This is were the protein is cut in two halves. The specificity of these sub sites are called cleavage activator and preventor, which means that an amino acid has to be or not be there to facilitate the cleavage. Trypsin, a well know protease cleaves after an arginine(R) or lysin(K) not followed by a proline(P). Here R and K are activators while P is a preventor, also written as [RK].<P> where . is the cleavage point [14]. Even if every protease has a specific cleavage point they are prone to miss some of them and can also cleave at the wrong site(random cleavage).

Trypsin is an often used protease, since it does few missed cleavages, little random cleavage, produces peptides with an average amino acids of 11(based on average occurrences of amino acids), easily obtainable and can be used in many situations like cleavage in gel etc. Chymotrypsin is another often used protease, but it is not as selective as trypsin.

Mass spectrometry has the weakness of having low sensitivity at low or high M/Z which means peptides can not be to short or to long. In practice this means choosing the protease with care. Foreknowledge of the sequence can help choose a protease that yields peptides of an appropriate length.

3.1.1.2 Separation

The most used method for separation within MS is SDS-PAGE(Sodiumdodecylphosphate -polyacrylamide gel electrophoresis) at least for protein mass-fingerprinting. While MS-MS experiments where digestion comes first often uses HPLC(high pressure liquid chromatography).

Gel electrophoresis

Gel electrophoresis is based on using an electric current to move molecules through a porous gel. Before the experiments starts a visible dye is added to the sample. This dye is a light molecule and acts an indication on when to stop the experiment by, most likely, being the fastest molecule in the sample. When the dye front(the dyes from all the wells) gets close to the end of the gel the electrophoresis should be stopped or the experiment might be ruined, because components in the sample might move of the gel. The length the molecule travels through the gel compared to the dye front is an indicator of how large/heavy the molecule is. The formula for this is:

$$\mathrm{Log}_{10} M = aRf + b,$$

where M is the mass, Rf is the migrated distance of the protein/migrated distance of dye front, and a and b are constants that are found by migrating a standard through the gel. Proteins have different overall charge and can be folded, which can result in partially shielding of the overall charge. SDS, a detergent with a hydrophilic head and a long hydrophobic tail is used to denature(unfold) the proteins. When it denatures a protein it adds a charge to the protein roughly proportional to the length of the protein. This has the added effect to prevent aggregation of proteins that are not properly folded.

Gel electrophoresis can be modified to take into account more than mass, pI (isoelectric point) is an example of this. For pI separation an, immobilized, pH gradient is used instead of current, making the proteins travel to the spot where the pH equals to their pI. The combination of mass and pI are called 2D PAGE and with the use of SDS, 2D SDS-PAGE(Fig. 3.4).

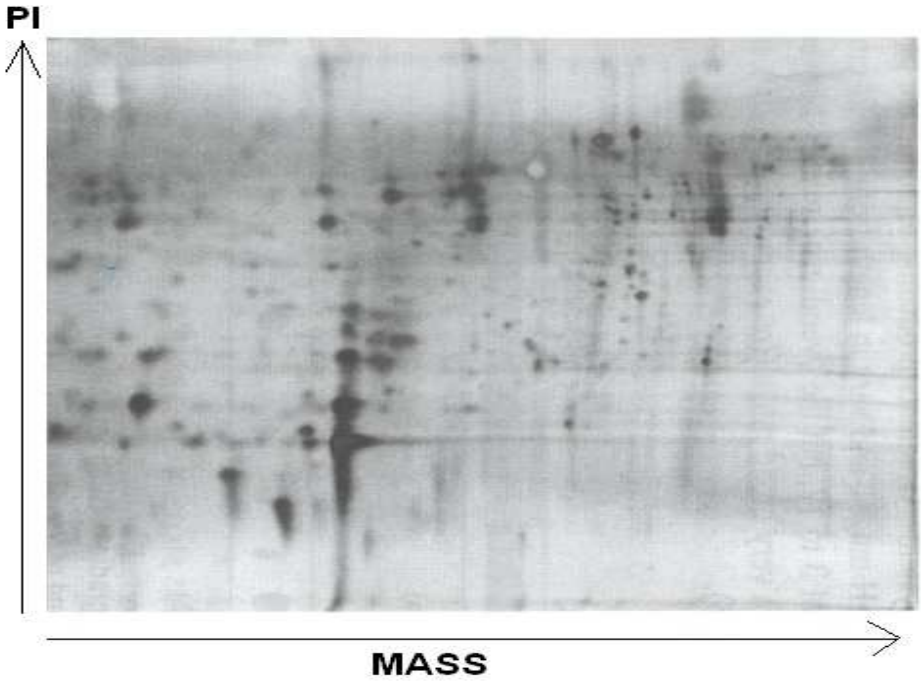


Figure 3.4: 2D SDS-PAGE showing mass in the horizontal and PI in the vertical direction. Recreated from [14].

Here one dimension of the gel is used for mass separation(current) and the other one is used for pI separation.

Visualization of the gel is done by staining the proteins, before the sample with them is added to the wells in the gel, and detecting them with an instrument. The most used staining methods are silver and fluorescent dyes. When the stains are visualized with an instrument, appropriate to the staining method used, the samples will show up as bands(spots with the shape of the well where the proteins are). These bands contains stained protein(s), and the intensity of a band compared to a standard can tell how much of that protein(s) is in that band. The problems with SDS-PAGE is listed underneath:

- Not all proteins will appear on the gel. This especially means hydrophobic proteins (GRAVY under 0.4), small proteins(mass under 8kDa), large proteins(mass over 150kDA), proteins with PI under 3 or above 10 and proteins of low abundance(because most dyes have a threshold for when it will stain a protein).
- Contamination of the gel
- Protein aggregation, not denatured by SDS (to low concentration of SDS). To high might result in micelles that can trap some of the dye.
- Not sufficient accuracy and precision makes it difficult to identify proteins by PAGE.

- The same band can contain more than one protein if two or more protein share common properties(like PI and mass for 2D PAGE).
- A protein can be divided between many bands if it has more than one form(truncated or modified).
- Low reproducibility of 2D SDS-PAGE.

Experiments where the differences between two cultures of the same bacteria are important, like up-regulation of proteins at a specific growth stadium, the two cultures can be stained with different dyes/metals. A reference sample can be stained by a third dye/metal. Followed by adding all of them to the same gel. The different dyes can then be detected and the intensity for the same bands, but for different samples, can be compared to see if there are changes in the transcription of the proteins in those band.

HPLC

HPLC(high pressure liquid chromatography) is based on column chromatography and separates the components of a solution based on its affinity to a liquid compared to a solid. The liquid containing the sample(eluent) is forced through a column by using a pump, instead of gravity. When the liquid with the sample enters the column the components in the sample will start interacting with the solid phase of the column. The stronger the affinity these components have to the solid phase, the longer they will take to pass through the column. The liquid phase is often varied so the affinity/solubility of the components to the liquid phase increases over time. Eluate is the name given to the liquid that has passed through the column. If the HPLC is equipped with a detector, often UV-spectrometer, the eluate's peak appear in a chromatogram (Fig. 3.5) based on the time it takes for it to travel through the column, also called retention time(t_r, max of the peak). For mass-spectrometry where the eluate enters the MS-instrument directly ion count can be used instead of the normal detection method.

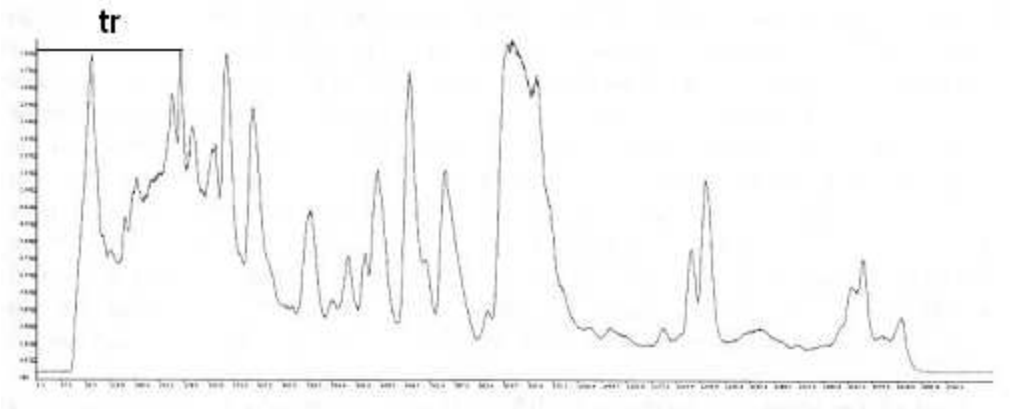


Figure 3.5: Chromatogram from HPLC showing the t_r to the third real visible peak. This chromatogram is a little noisy, since some peaks might be two peaks, meaning that the resolution of the chromatogram is not to high. Taken from [14].

Chromatograms can be used for identification and quantification. Identification is done by using the retention time of a component, while quantification is done by calculating the area under the peak. Resolution is how well two neighbouring peaks are separated, higher means better. Both resolution and the fact that peaks become wider at higher retention times are a factor that can make identification and quantification more difficult. Figure 3.6 shows two ideal peaks while figure 3.5 shows a typical chromatogram.

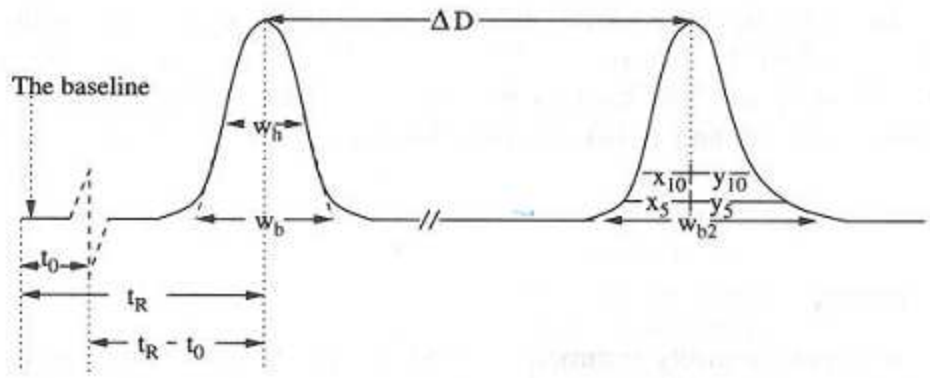


Figure 3.6: Shows two peaks that are perfectly separated with no asymetrically tails, ideal peaks. t_0 is the deadtime(the time it takes substance to eluate from the HPLC), t_r is the retention time, ΔD is the distance between two equally high peaks, w_h is the peak's half width, w_b is the baseline width of the peak. X_{10} is the width on the left side at 10% height etc. Taken from [14].

Different forms of HPLC exist. One of them are reverse phase(RP) chromatography which uses hydrophobicity as separating factor. Another is strong cation exchange chromatography (SCX) which separates using the charge of the molecules as a factor. Variation on SCX is weak cation exchange, weak anion exchange and strong anion exchange. Basically the difference is what type of charge is used as the separating factor and what pH is needed to neutralize the charge. Others also exist but RP and SCX is the most common used in proteomics.

3.2 MS Instrumentation

As said earlier MS consist of three steps, ionization, mass analysis and detection. These three steps often use different instruments with their own strength and weaknesses. Following will be a short introduction to the most common instruments for mass spectrometry on peptides and proteins.

3.2.1 Ionization

Ionization sources are often categorized as hard and soft ionization. Large molecules like proteins often have unwanted fragmentation with hard ionization sources, while soft ionization keeps the molecules intact. Soft ionization can even under specific conditions keep non-covalent interactions intact[28]. This is why soft ionization sources like MALDI and ESI made MS so popular for proteomics. Ionization sources differs in the amount it ionizes each molecule, some gives molecules with one charge(wanted in MS) others more(wanted in MS/MS, since molecules are fragmented between the two MS). Common for all of them are that they should follow as many of these simple rules as possible:

- All components should be ionized in a detectable amount
- The ionized amount should be proportional to the amount of the component in the sample.
- There should be no fragmentation unless fragments are wanted.
- There should be no unwanted adduct ions, where adduct ions are a component in the sample modified by one or more atoms.
- No contamination should occur, this is easiest to remove in the sample preparation.

3.2.1.1 MALDI(matrix assisted laser desorption ionization)

MALDI have the property of ionizing molecules in such a way that they get an average charge of one. This makes it dominant in single mass-spectrometry, but it can also be used in MS/MS although ionization sources that gives molecules of a higher charge is preferred here. The matrix in MALDI consist of small organic molecules(α-cyano-4-hydroxycinnamic acid is very commonly used in peptide MS) that absorbs light at specific wavelengths(usually UV).

An organic solvent, in acidic conditions, is used to dissolve the matrix before it is mixed with the sample. The mixed solution is spotted on a sample plate as small droplet(micro liters). Here the organic solvent is allowed to evaporate. During this time the matrix goes through a process called crystallisation where the matrix forms crystals. These newly formed crystals contains the sample within them acting as a shell that protects the sample from the disruptive laser light. The plate is then fired upon by a laser in short pulses and the matrix molecules absorbs the energy/gets ionized and in the process is released from the plate. Optimal conditions is when the laser light has the same wavelength as the strong absorption peak of the matrix. When the matrix is in the gas-phase the sample is able to receive a proton from the matrix/get ionized. The ionized sample is then transferred by an electric field to the mass analyzer.

3.2.1.2 ESI (electron spray ionization)

ESI is the choice for MS/MS since it has the ability to strongly ionize molecules, giving them more than one charge. The samples is brought into the ionization source in a liquid flow, example the eluate from HPLC. In the ionization source the liquid is forced through a heated needle which sprays the sample, as a mist of small droplets, into a strong electromagnetic field. The droplets evaporates slowly and as they become smaller the electric field gets stronger on their surface. When the field is strong enough the sample desorbs from the surface mostly carrying two or more in charge. While under the influence of the electric field the sample is transported to the mass analyzer. Most ESI instruments operates with an atmospheric pressure and mass analyzers usually operates with low pressure, which means the sample must be transported from atmospheric to low pressure. Figure 3.7 shows the principle of how ESI works.

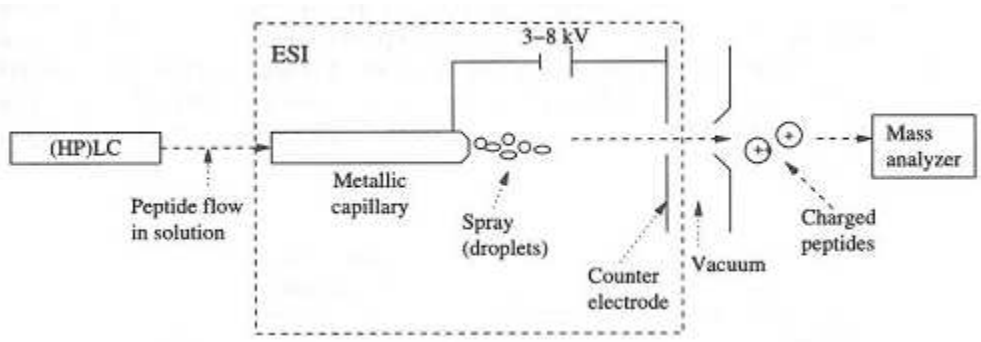


Figure 3.7: The principle of ESI. Taken from [14].

3.2.2 Mass Analyzer

This is the part where MS and MS/MS differs. MS is usually coupled with a TOF(time of flight) analyzer, while MS/MS have more options. MS/MS have a wide array of mass analyzers like TOF/TOF, Ion Trap(IT), Triple Quadrupole(triple quad), Fourier Transform Ion Cyclotron Resonance(FT-ICR) etc. or a combination of these like the TOF-IT.

One important ability of mass analyzers is their ability to discriminate between two components with close to equal masses, example isotopes. This ability has the name resolution or revolving power, which are two ways to measure the discriminating power of a mass analyzer. Resolution can be, in league with early IUPAC, described as:

- *p percent valley definition:* the resolution is defined as m/Δm if two peaks of equal heights are separated by a valley where p is the percentage of the lowest point between these two peaks. p is usually set to 10.
- *Peak width definition:* the resolution is defined as m/Δm where m is the mass and Δm is the peak width at the height specified as the fraction of the peak height. The fraction 50%, 5% or 0.5% of the height is recommended. FWHM is often used as abbreviation for the 50% height fraction.

3.2.2.1 TOF(time of flight)

The time of flight mass analyzer is simply a tube where the inside is kept at vacuum conditions, or as close as possible. The time it takes a molecule to travel through the tube based on its start/end velocity(same since it is operated under vacuum) can be used to calculate the mass of the molecule by using simple physics. The kinetic energy for the molecule is produced by a electric field resulting in the formula:

$$\frac{mv^2}{2} = zeP,$$

where m is the mass, v is the velocity, z is the charge, e is the charge of an electron and P is the potential of the electric field. Adding the relation between the time and the distance of travel, t = d/v (v since we have vacuum), and rearranging the formula we get:

$$t = \frac{d}{\sqrt{2eP}}\sqrt{\frac{m}{z}} = C\sqrt{\frac{m}{z}}$$

From the above formula the difference between TOF for two molecules can be calculated as:

$$\Delta t = \frac{d}{\sqrt{2eP}}\sqrt{\frac{m+\Delta m+z}{z}} - \sqrt{\frac{m+z}{z}}$$

From this we can see that the resolution between two components with one mass unit difference decreases with increasing mass. This means that the higher the mass becomes the harder it is to observe the difference between the two components. Another important factor, for flight time, is the spreading of the same peptide when it reaches the detector. This happens because when the matrix leaves the sample plate it gets spread out in 3D space, meaning the same peptide in the sample will start at different points and get different velocity. Other factors that influence the spreading is the length of the laser pulse and the size of the matrix/sample. This spreading can partly be rectified by increasing the d(length of the tube) or decreasing the voltage. Both has its limitations, since it will be difficult to keep vacuum in a long tube and lower voltage has a negative impact on the sensitivity. Other methods are delayed pulse extraction and reflectron.

Delayed pulse extraction basically gives the ions a free drift time into the extraction field before the voltage is turned on. The extraction voltage is turned on a few tens to a few hundred nanoseconds after the laser pulse. This results in fast ions getting less extra velocity since they are further into the field.

Reflectron is an electrostatic mirror placed at the end of the linear flight tube effectively doing two jobs. The first is to increase the flight length by turning the peptides back in the same direction as the ion source, and the second is to gather the peptides of the same mass (Fig. 3.8). The reason this happens is because ionized peptides that enter the electrostatic field will be turned back depending on their velocity. Ions with higher velocity penetrates deeper into the field before turning back.

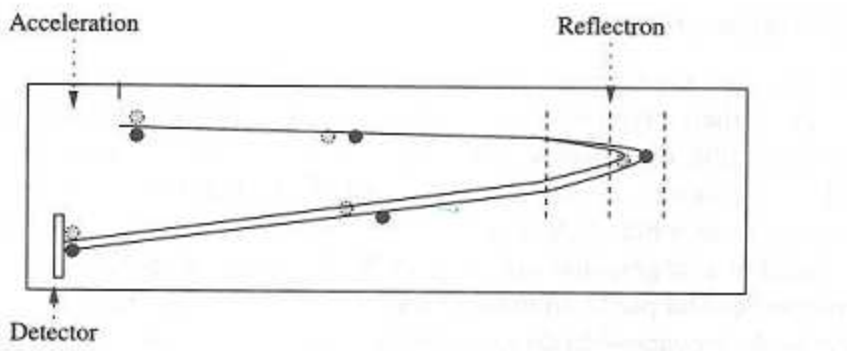


Figure 3.8: Shows a reflectron that are fitted to a TOF analyzer. The white and black spot in the upper right symbolizes ions with different velocity but have the same mass. When they hit the reflectron(stippled lines) the one with the higher velocity penetrates deeper and therefore are turned around later which results in both ions hitting the detector in the lower left at the same time. Taken from [14].

Every instrument needs to be calibrated before use, since most produce random electrical noise when used. This is done with a standard, either added into the sample(internal) or in a run done just before the sample(external). Internal calibration might make the MS spectra more clustered but is more accurate since MS spectra have low reproducibility. Some proteases also have autolytic peaks, by self-digestion, that is already known and can therefore be used as a standard. External calibration on the other hand suffers from the low reproducibility of mass spectrometry on peptides.. For TOF the calibration follows the equation:

$$\frac{m}{z} = a\,t^2 + bt + c\ ,$$

where the constants a, b and c have to be found for each individual instrument.

3.2.3 Detectors

A detector measures the ions that hits it at the each time interval. The electric signal produced by the detector is sent to a computer that produces a mass spectrum from it.

3.3 MS/MS Instruments

MS/MS mass analyzers needs to fragment the peptides between their two runs, which means the build of these are often a little different(chapter 3.6). Peptide MS/MS usually uses post source fragmentation (PSD), instead of the normal in source ionization(ISD).

3.3.1 TOF/TOF

These have two TOF tubes. The first one gets the initial MS analyses done, while the second one analyses the fragments from a specific M/Z range. Figure 3.9 shows how a TOF/TOF works with fragmentation after selection, and figure 3.10 shows one with fragmentation before selection.

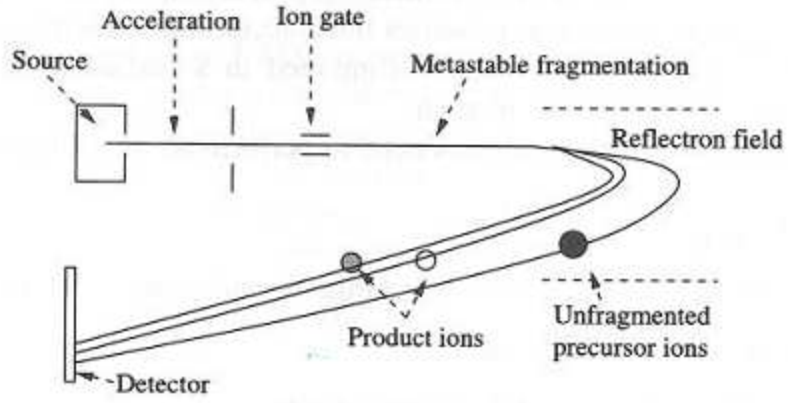


Fig 3.9: Shows the principle for a TOF/TOF analyzer with PSD(post source decay) fragmentation. The ion gate only lets through ions in a specific M/Z range. Taken from [14].

Fragmentation(Fig. 3.9) is done by laser induced dissociation(LID). This is were the laser produces metastable ions that will fragment by themselves in the mass analyzer. MALDI process can provide enough energy for this to happen and produces mainly a, b and y product ions(Appendix D).

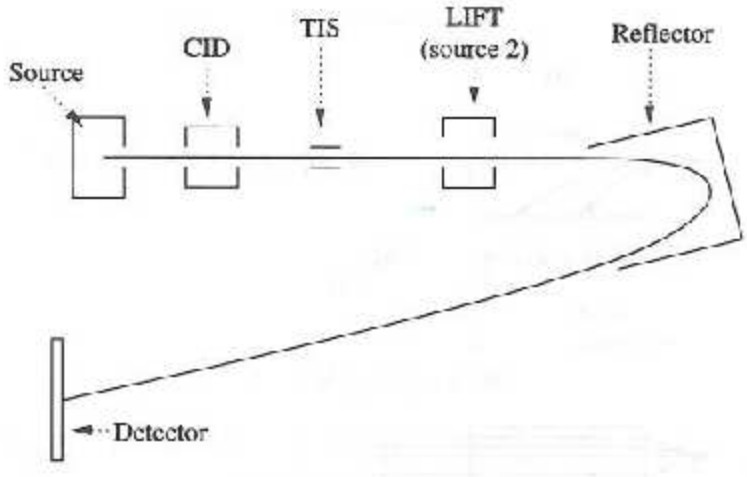


Fig 3.10: Illustrates fragmentation of the ions before they have been selected for the second MS. CID means collision induced dissociation which takes place in a collision cell. In this cell you have an inert collision gas(example argon). The peptides collides with this gas and builds up potential energy resulting in fragmentation when the peptide reaches its fragmentation threshold. This results in product ions and neural loss fragments(fragments without charge). TIS is an ion gate that only lets ion with a specific velocity through. Ions from the same peptide have the same velocity. LIFT is a unit that provides a lift in potential energy to a molecule. Taken from [14].

For information about how quadrupole, ion trap and fourier transform ion cyclotron resonance works see reference 14, chapter 8.4. Table 3.1 shows the characteristics of the different mass analyzers for MS/MS. Note that the data changes since all the analyzers are improved continually by their manufacturers.

Table 8.1 Characteristics and performances of commonly used types of mass spectrometers. v indicate available, (v) indicate optional. +, ++, +++ indicate possible or moderate, good or high, and excellent or very high, respectively. LOD = Limit of Detection. Reproduced from Domon and Aebersold (2006) by permission of SCIENCE AAAS

	IT/LIT	Q-TOF	TOF/TOF	FT-ICR	Triple quad	Q-TRAP
Mass accuracy	Low	Good	Good	Excellent	Medium	Medium
Resolving power	Low	Good	High	Very high	Low	Low
Sensitivity (LOD)	Good		High	Medium	High	High
Dynamic range	Low	Medium	Medium	Medium	High	High
ESI	v	v		v	v	v
MALDI	(v)	(v)	v			
Identification	++	++	++	+++	+	+
Quantification	+	+++	++	++	+++	+++
Throughput	+++	++	+++	++	++	++
Detection of modifications	+	+	+	+		+++

Table 3.1: Note: Copy and cited from article 14, where the top contains the table text.

3.4 Mass Spectrum

The mass spectrum is a graph where the x-axis is M/Z and the y-axis is intensity. Here M/Z means the mass to charge ratio and is dimensionless even though it is sometimes falsely reported as Da, Th or u. The charge of a peak needs to be known to calculate the real mass of a peptide. The intensity on the other hand is the strength of the signal at a specific M/Z value. A mass spectrum produced by a MS instrument are often called a raw spectrum. These raw spectra can contain anything from a little to a lot of noise, which mainly comes in the form of electronic noise and chemical noise. Electronic noise is random fluctuation in the instrument, while chemical noise is contamination from sample preparation. Contaminations can be detergents, polymers leaked from sample tubes, human keratin etc. The raw spectrum has to be 'cleaned' up before any analysis can be used to pry out the information it contains. A lot of methods have been invented for exactly this purpose and can be used as stand alone or combined as needed. Some of them being baseline correction, noise reduction, deisotoping and peak detection.

3.4.1 Baseline Correction

The baseline is mostly chemical noise that offsets the intensities. Figure 3.11 shows a mass spectrum with a high baseline. This kind of noise often shows a high dependency on M/Z values, the lower the M/Z the higher the noise. It also varies from spectrum to spectrum, meaning that each spectrum has to be treated separately.

Noise can be corrected by decreasing the intensities. The simplest form is to find the lowest point and move it to 0% intensity while keeping the highest point at 100% intensity. This means 'extending' the peaks vertically. More advanced methods uses locale baseline corrections to take into account that the baseline varies with M/Z values. These methods uses a sliding window(looks at values in intervals along the M/Z-axis) and fits the baseline to polynomial or exponential functions. The window for this sort of baseline correction should preferably contain no real peaks, or the function should at least not fit real peaks into the baseline.

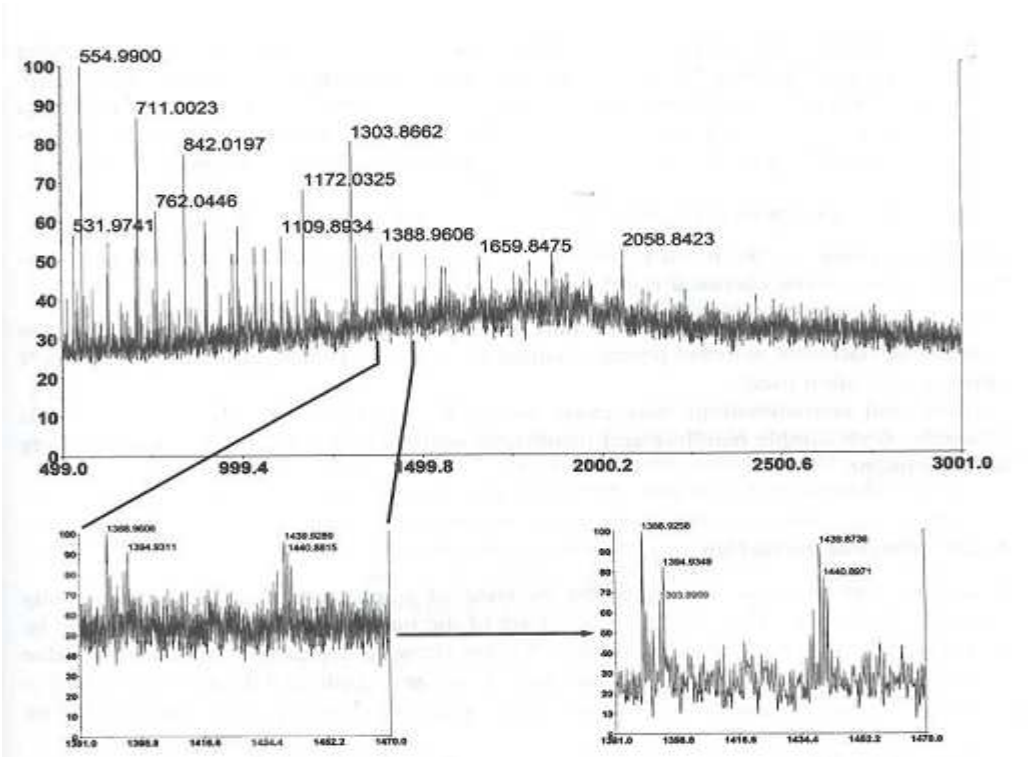


Figure 3.11: Upper part of the figure shows a raw mass spectrum with a high baseline, much noise and low intensity noise. The lower left shows a small part of this mass spectrum and it shows that the peaks are hard to separate. The lower right shows the same area after noise filtering and baseline correction. One note here is that all this noise most probably will affect the peak intensities(from the peak detection algorithm) as well as the M/Z values. Taken from [14].

Another method for baseline correction is the non-linear filter called top-hat, taken from the mathematical morphology literature. In mathematics top-hat means to subtract its morphological opening, which is the dilation of the erosion of a set A by a structuring element B. In plain English this means that you remove objects larger than the structuring object(whose size must be chosen carefully). This typically results in a contrast enhancement because the slow trends are removed. The top-hat operator is mainly used for enhancing the contrast in images, but can be used for any numerical function[29, 30]. Figure 3.12 shows the top-hat used on a sample(could have been a mass spectrum).

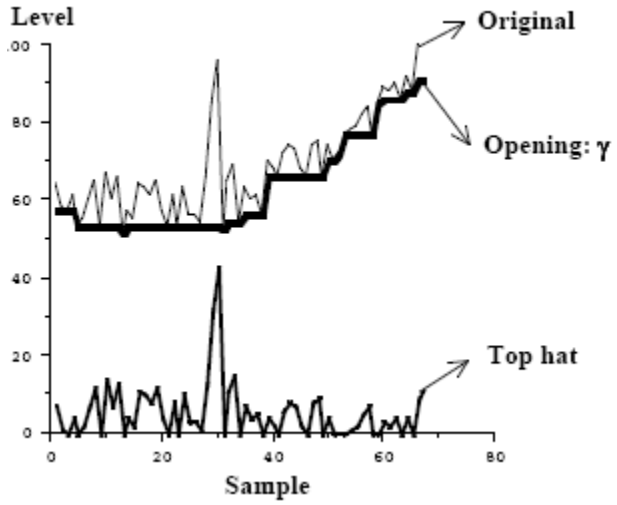


Figure 3.12: Top-hat operator used on a sample. Taken from [29].

3.4.2 Noise Reduction and Smoothing

A mass spectrum can be vary jagged, figure 11, and therefore hard to read. Smoothing can help making it more readable. Smoothing can be looked upon as a filter, in that a sliding window is moved over the data while 'correcting' the middle value based on a function. The windows length and weight are based upon known noise, where the resolution often is taken into account. The function used is usually based upon a weighted average of the values in the window. A common used smoothing is Gaussian filter/function, shown below:

$$G(x)=\frac{1}{\sqrt{2\pi\rho}}e^{\frac{-x^2}{2\rho^2}},$$

where ρ determines the length of the window and the weights are calculated by a discretization of the function values. The varying noise level along the M/Z-axis makes small windows, usually no larger than 1Da, commonly used for the function.

Fourier transform was used by Baggerly et.al[31] when they found a systematic noise in their mass spectra data set. They used a fourier transform, for several spectra, in the regions that did not contain any of the larger peaks to establish their suspicions. The noise was consistent but not in the same phase, and was removed by subtracting out a sinusoid out of the tails of each of the spectra. The noise was suspected to come from their alternating current power source.

Savitsky-Golay is often used in removing noise in mass spectrum. it is simply a simplified least square method using a sliding window to calculate the middle point of the window. A 5 point Savitsky-Golay is using a window 5 points wide. The article, reference 31, has a real good appendix for weights for the different points in the window for different types of filters[32].

3.4.3 Peak Detection

The main object of peak detection algorithms is to create a peak list where each peak represent only true peptides, and their isotopes, from the sample.

Centroid peak detection is a popular way to detect peaks[33]. Gentzel et.al. made one that also uses the change of the peak width with different M/Z values[34]. The algorithm starts of by doing a windowed search along the M/Z axis. The width of the window is based on the M/Z value and is shown in the equation underneath:

$$\text{peak width}(x) = pw(x) = 0.08+0.0004x,$$

where x is the M/Z value where the peak is centroided and the coefficients are based on the instrument used. This equation can be seen upon as varying resolution as M/Z varies, in an increasing manner with increasing M/Z, which compares well to how the resolution works for most MS instruments. In each window the M/Z value with the highest value(x_m) is found and the total intensity for all the M/Z values in the window is summed using the equation:

$$\sum_j y_j \geq t ,$$

where the sum is over j such that $x_i \le x_j \le x_i + pw(x)$ and t is a predefined threshold. If the sum equals or exceeds the threshold a check is done to see if the maximum is on the flank of the peak. If it is then the window is moved until it is not. Then a centroid is placed at the highest M/Z value in the window and M/Z(x_c) and intensity(y_c) is calculated for the peak using the equations:

$$x_c = \frac{\sum_j x_j * y_j}{\sum_j y_j} \quad \text{and} \quad y_c = \sum_j y_j \,,$$

where the sum is over j such that $x_m - pw(x_m)/2 \le x_m \le x_m + pw(x_m)/2$. If a peak has been detected the window is slided to the end of the detected peak(x_m+pw(x_m)/2) and started anew to try to find the next peak in the spectra. They also mentioned using a test to check if the two slopes of the centroid peak is shared with another peak and dividing the intensity between them if it is. A test to check the spacing of the data points to see if the peaks already had been centroided was also mentioned. Figure 3.12 shows a run of the method on a MS/MS spectra.

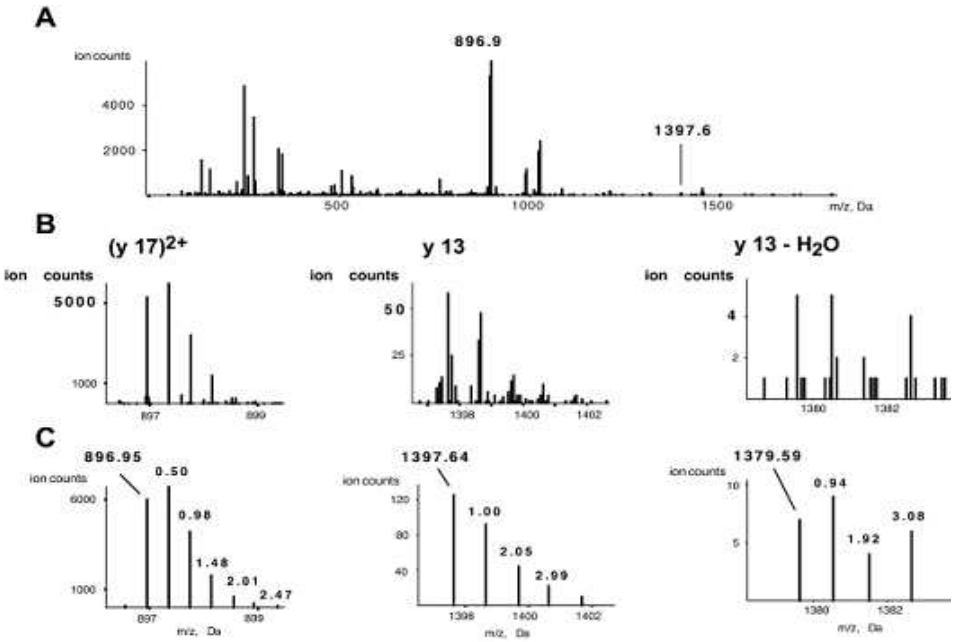


Figure 3.12: Panel A shows a complete MS-MS spectrum. Panel B shows selected ions from the spectrum in panel A. While panel C shows the same ions as B but after the peak detection algorithm, centroiding, has been run. Taken from [34].

Continuous wavelets transformation(CWT) have been used to detects peaks in a spectrum[35] using the mexican hat wavelet as the mother-wavelet. CWT is used to divide continuous-time functions into wavelets and consist of a mother wavelet and daughter wavelets. The main purpose of the mother-wavelet is to provide a source function to generate the daughter wavelets which are simply the translated and scaled versions of the mother-wavelet. Mathematically the CWT can be represented as:

$$C(a,b) = \int s(t)\psi_{a,b}(t)\,dt \,, \psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\frac{t-b}{a} 0, a \in R^{\pm\{0\}}, b \in R \,,$$

where s(t) is the signal, a is the scale, b is the translation, ψ(t) is the mother wavelet, ψ(a,b) is the scaled translated wavelet and C is the 2D matrix of wavelet coefficients. These coefficients reflects the pattern between the signal s and $\psi_{a,b}$(t). The higher they are the better the matching.

In the peak detection algorithm they used CWT to examine the changes in height and width of the peaks in the mass spectrum. The coefficient for each scale of the CWT has its local maxima around the peak's center. Figure 3.13 shows a mass spectrum, 2D CWT matrix and a ridge line graph. Looking at the 2D CWT matrix and the MS you can clearly see that the lighter areas match up with peaks in the mass spectrum and that the top scales(upper part of the CWT) matches the peaks widths. The ridge line graph is created by first finding the local maximas for each scale in the 2D CWT by using a sliding window. The maximas are then linked together by using another sliding window to create the ridges in the ridge line graph. This is done by searching every level of the scale until the gap between the scale with no found maxima exceeds the threshold for the gap size. The rest of the maximas on scales lower than the last found maxima is removed from that m/z value. The procedure is repeated until the end of the 2D CWT is reached. This transformation is less susceptible to local minima and more robust to changes to the coefficients in the search space.

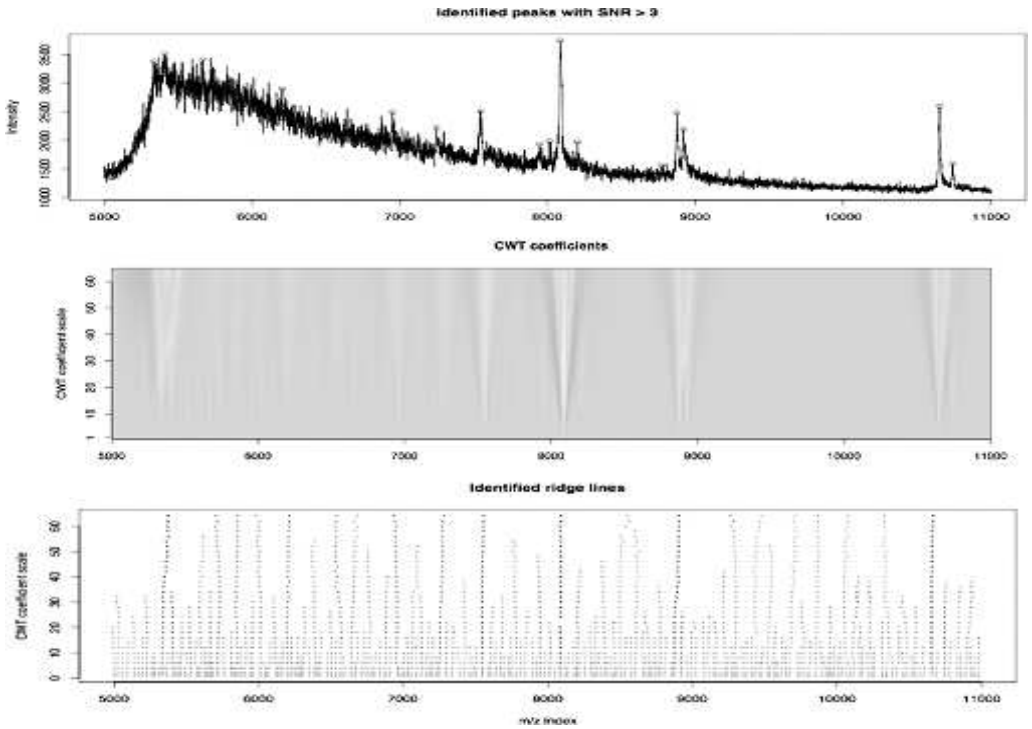


Figure 3.13: Upper part shows a mass spectrum. The middle part shows the mass spectrum after wavelet transformation, called 2D CWT. The peaks can here be seen as the lighter parts in the 2D CWT. The lower part shows the ridge line graph for the 2D CWT. Taken from [35].

Baseline reduction is automatically done in their wavelet function. They also define the signal to noise ratio(SNR) as the ratio of the estimated peak signal strength to the locale noise around that peak. This local noise are looked upon as small positive and negative peaks, and uses the lowest CWT scales coefficients (a=1) to represent the noise. They also define the local noise around a peak to be the 95% quantile of the absolute CWT coefficient in a window around the peak.

To identify the peaks from the ridge line graph they use three rules:

- The scale corresponding to the maximum amplitude on the ridge line, which is proportional to the width of the peak, should be within a certain range
- The SNR should be larger than a certain threshold.
- The length of ridge lines should be larger than a certain threshold.

They then uses the rules to estimate the peak parameters and refines it afterwards. This estimate can be refined again if needed by doing another CWT over a defined segments in the mass spectrum.

A template model has also been used to detect peaks, where the template is how a peak should be identified, with properties like width, height etc., in the mass spectrum. This is then used to find peaks satisfying part of the template based on a threshold. The match needs to fulfil three more criteria, which is high signal to noise ratio, be precise and be unique. They used a template that is able to adapt to variable shapes by adapting it to the parameters set in the used database[36].

3.4.4 Intensity Normalization

Peak intensities varies from spectrum to spectrum, which means a comparison of these to each other can be difficult. To rectify this the peaks can be normalized to the highest intensity in the spectrum. This changes intensities to percentages ranging from 0-100%. The spectrum can also be normalized with TIC(total ion count of the spectrum) resulting in the same relations, as the previous, between the peaks. A threshold can then be used to remove the noise from the spectrum, removing all the values underneath it by considering them as noise. Both methods goes under the term relative intensity normalization(RIN). The weakness of these two methods are that under the quite common conditions when one peak has much higher intensity than the rest, the smaller peaks, even if they are real, have a great chance to be considered as noise [37].

A more robust method, across different spectra, is to use rank based normalization. The highest intensity peak gets rank one, second highest rank two etc.

Since normal rank normalization does not take the magnitude of the intensities into account a method called cumulative intensity normalization(CIN) can be used[37]. The equation is shown below:

Cumulative normalized intensity of the nth highest peak = $\frac{\sum_{rank(x)\geq n} \boldsymbol{I(x)}}{TIC}$,

where I(x) is the raw intensity of the peak, TIC is the total ion count and rank(x) is the rank of peak x when sorted by descending order of intensity magnitude.

Figure 3.14 shows an example where there is one strong peak. The two graphs to the right shows the intensity normalized with RIN(using TIC) and CIN. The x-axis is the normalized values sorted on ascending order, while the y-axis is the normalized intensities. The line in both graphs are the rank based normalization. As the graph in figure 3.14(right side) shows the normalized intensities are strengthened in the CIN compared to RIN.

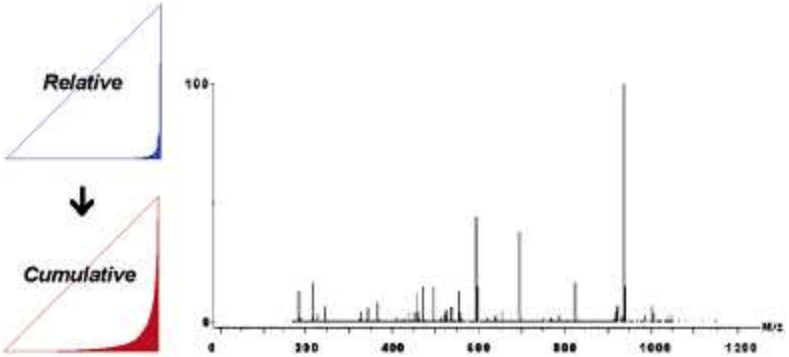


Figure 3.14: The mass spectrum to the right is the one that the normalization is done to. To the upper left is the relative normalization and to the lower left is the cumulative normalization. Taken from [37].

Figure 3.15 shows another MS spectrum and it shows almost the same as the previous figure. But here the intensities for the CIN are weakened at the low end of the x-axis which is often noise. The CIN also has a smoother curve than the RIN and are more in line with the rank based[37].

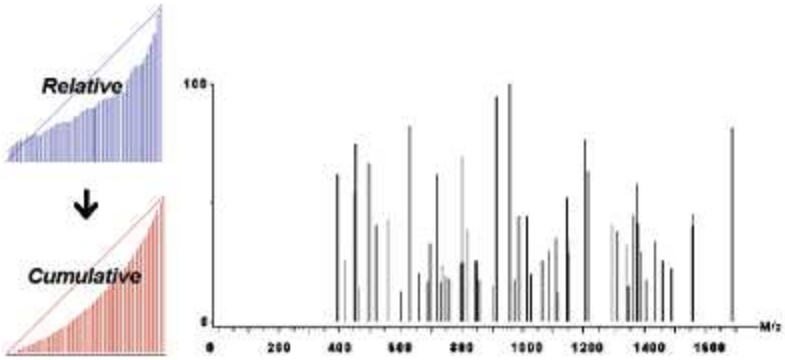


Figure 3.15: The mass spectrum to the right is the one that the normalization is done to. To the upper left is the relative normalization and to the lower left is the cumulative normalization. Taken from [37].

If a whole set of mass spectra is looked upon at once the normalization coefficient(dividing coefficient) can be set to be the total TIC of a spectrum divided by the total TIC of all the spectra in the set[30].

3.4.5 Deisotoping

Since peptides consist mainly of many C and H atoms the isotopes of these will result in an isotopic envelope in the mass spectrum(shown in Fig. 3.16). Deisotoping is the process of reducing these isotopic envelopes into a single peak representing it. There are more than one way of doing this.

Monoisotoping is the reduction of the isotopic envelope down to the lowest M/Z value in that envelope. Usually used in MALDI where reflectron and delayed extraction has been used.

Intensityisotoping is the process of reducing the isotopic envelope down to the most intense peak in the envelope.

Deisotoping reduces the isotopic envelope down to the centroid peak, where the centroids M/Z value is determined from the intensities of the individual isotopes in the envelope. This corresponds to the calculation of the average mass of the peptide in the envelope.

Beware that isotopic envelopes for two different peptides might overlap in the mass spectrum and separating them, as two peaks, might be a difficult task. Study of isotopic behavior for the instrument(for MALDI the matrix) in question might help[14].

There have been done a small unpublished comparison studies, where charge overlapping was seen upon as a problem, of the deisotoping programs Mascot Distiller, MaxEnt3 (Waters) and ReSpect (PPL) [38].

A more accurate method for deisotoping is described by Olson et.al. called probability grouping[39], which is used to find probable isotopic clusters for an analyte by calculating a theoretical mass spectrum. They also mention dynamic programing and fourier transform as emerging methods for deisotoping.

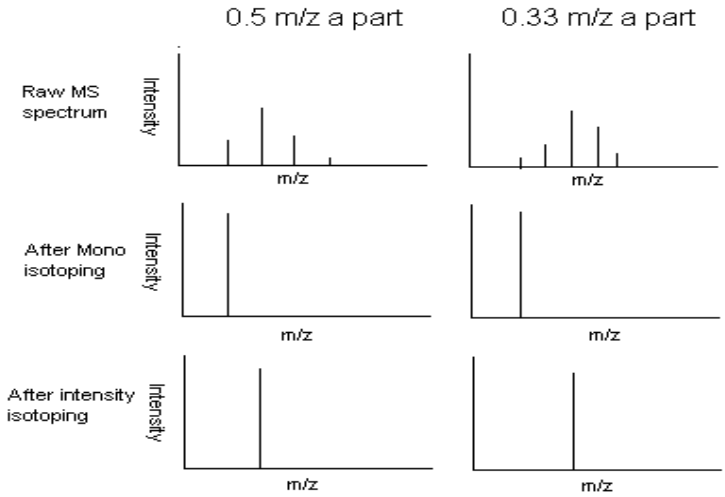


Figure 3.16: The upper part shows a cut from a mass spectrum. The one to the left has a cluster of peaks with charge 2, ergo each peak is 0.5 m/z a part from the previous one. To the right has a cluster of peaks with charge 3, ergo 0.33m/z difference between the peaks. The middle part show a mono isotoping while the lower part shows a intensity isotoping of the cuts from the spectrum in the upper part.

3.4.6 Deconvolution

Mass spectrum produces from ionization sources that produces peptides with more than one charge, like ESI, can also produce peaks for the same peptide more than once in a mass spectrum. When these peaks are gathered into one it is called deconvolution/charge state deconvolution. The charge of the resulting peak is usually one(Fig. 3.17).

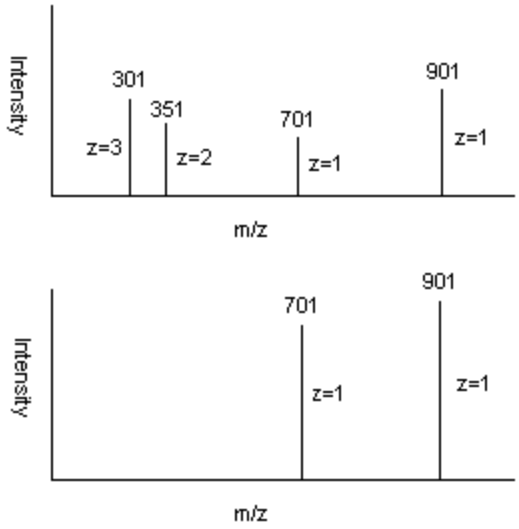


Figure 3.17: The upper part shows a mass spectrum with peaks at 301, 351, 701 and 901. The lower part deconvulution is done on the spectrum above and two peaks of charge one is left, mainly the peak at 701 and 901.

3.4.7 Spurious Peak Removal

Spurious peaks are non-peptide peaks. These peptides can be removed by using the fractional masses of peptides(Fig. 3.18 shows a distribution of these). Fractional masses is the part of the mass after the decimal point. Every amino acid are built from 4 building blocks which have close to an integer mass. This means every amino acid will have close to an integer mass. For one charged peptides digested by trypsin the graph in figure 3.19 can be used to calculate two equations to be used to identify peaks that are non-peptides.

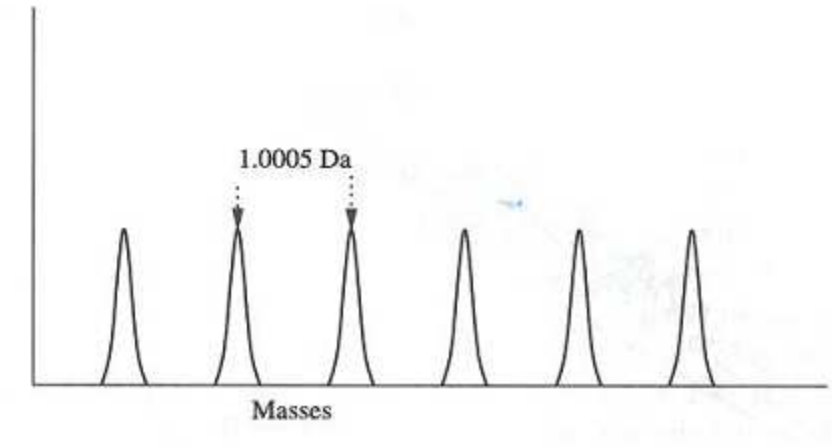


Figure 3.18: Depicted in this figure is the 1.0005Da distance between the two apexes of neighbouring peptide clusters taken from a peptide distribution. Taken from [14].

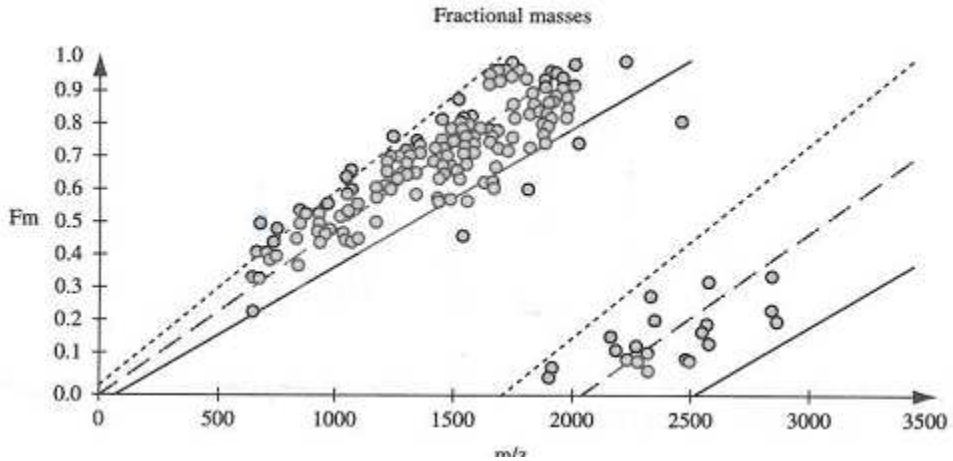


Figure 3.19: The graph shows the distribution of the peptide masses, cleaved with trypsin, compared to m/z values. The peptide masses is single charged. The middle line is the average regression line for the fractional masses and the lower and upper is one standard deviation from the average. Taken from [14].

Peaks with low, sometimes all under 500, M/Z can usually be removed since the background noise is high at that range. If single charge is used the M/Z might go as high as 700-800 since peptides cleaved, with trypsin, has a specific minimum M/Z. Peaks with M/Z values above 3-4000 are also removed since sensitivity and accuracy are usually lower after this. Also peptides are seldom that heavy.

Spurious peaks can also be removed by more direct means, like an exclusion list. This list can as an example contain peaks that is typical for keratin cleavage with the used protease etc. But beware that removing these kind of peaks might be bad for the result, since there might be overlapping isotopic envelopes or a peptide from the list can have the same mass as peptide in the sample(or the same M/Z, since the peptide in the sample can have more than one charge).

For large sets of mass spectra each mass spectrum can be scanned and peaks appearing in many mass spectra can be removed as it is likely a contamination. This of course depends on how the experiment is set-up. Theoretically you can produce a lot of similar peaks from a sample in a lot of spectra.

3.5 Database Search

As mentioned in chapter 3.1 proteins gets stored in databases. These databases can be used in MS to identify the peaks in the mass spectra.

In top-down MS this means calculating the mass of a peak and then compare it to the mass of the proteins in the database.

Bottom-up MS on the other hand operates on peptides, meaning the database has to be preprocessed before the search is done. The preprocessing consist of at least in silico cleavage, which is the theoretical cleavage of proteins in the database using the protease used in the experiment. One problem present itself with cleavage of proteins and that is missed cleavages and random cleavages. If a more precise, or the chance to get a higher coverage of the protein, is wanted then these two should be taken into account.

Missed cleavages is rather simple too take into account by just missing cleavage sites. The result on the other hand is a much huger search space. To be more precise the search space increases by 1/2(2n+2nk-k^2+k+2) where k is the number of possible missed cleavages for each peptide.

Random cleavage on the other hand presents more trouble than it is worth. 1. it increases the search space with a vast amount, even for small proteins. 2. the fragments will often make no sense since it will be covered by most of the proteins. Number 1 and 2 can be rectified a little by restricting the random fragments to a specific minimum size.

On top of this another phenomena called PTM is important to consider. PTM as explained in the chapter 2.1.2 comes in varied sizes and varied amino acids they modify. Taking a lot of these into account also increases the search space considerably. On the bright side the search space can be reduced considerably by having some foreknowledge of the organism or protein group the proteins comes from. Even limiting the search only to a family of organisms limits the search space a lot.

The right part of figure 3.20 shows a simplified view of the preprocessing steps, while the left shows the same for the MS experiment.

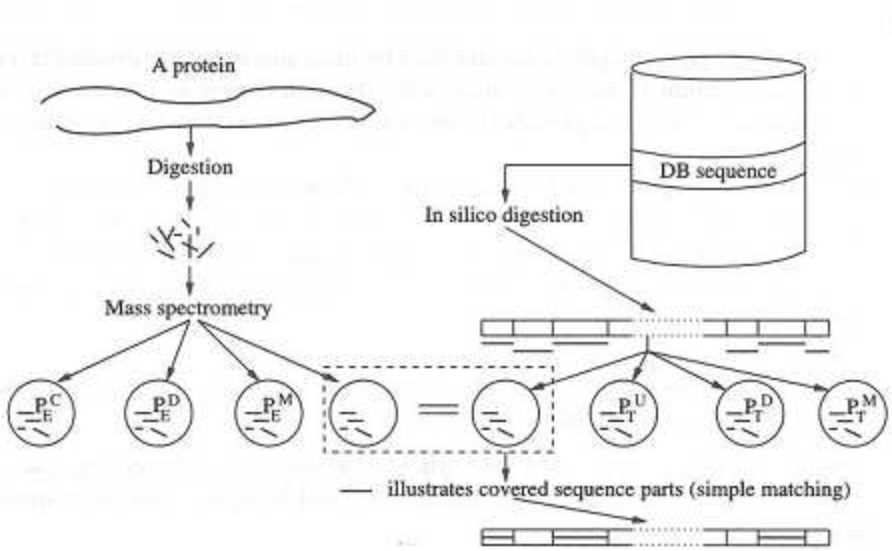


Figure 3.20: To the left the a protein is digested and a MS is done on it. On the left side the same is done with in silico digestion. A search with the experimental peptides are done in the database and the coverage of a match is shown in the lower part of the figure. Taken from [14].

When the database has been preprocessed the peptides can be compared to the database's peptides. The result is most often a coverage for the proteins found in the database. Where the coverage means how many peptides from the sample matches the in silico peptides from the said protein in the database(bottom Fig. 3.20). An example is that three peptides from the sample matches three of the in silico peptides for a protein found in the database, giving it a coverage of 3. Coverage is also often given as a percentage, where a 20-30% coverage of a protein is ordinary for an MS experiment. The coverage is so low in MS experiments that random matching can give a high coverage by chance. This is why scoring schemes have been invented to score the different matches in a database search. Also a P-value, probability that a match is by chance, is used to make sure that the right protein is the most likely to be the one shown as the top one in the result list. Even these three indication might not be enough to find the most likely protein(s) from the database that is in the sample. A simple solution to this might be that the protein has never been sequenced before, ergo not appearing in the database. Another solution might be that to many proteins from the database gets close to the same score, coverage and P-value, but as with all chemistry more experiments can be done to get to the solution. All the extra information such experiments will give will most likely help to deduce what protein is the right one.

PMF searches on masses and this from a protein that is selected before digestion. In theory this makes PMF database search a one protein search. In practice the protein chosen for digestion can as well be two or more proteins. This means the database search should consider this. The easiest way is to remove the peptides(peaks in MS) that identified the first protein and then do another search on the remaining peptides(peaks in MS). Even with this search a random protein can get a higher coverage than the true proteins, since the second, third etc. protein gives peptides that can match the peptides in the matched protein by chance. This can be remedied by limiting the search to a specific group of proteins.

For MS search program like MSA and Aldente exist. While scoring algorithms like Mascot-Mowse and OLAV-PMF is used after the search. Search algorithm for MS-MS can be as simple as a sequence search with FASTA. But MS-MS requires more preparation before getting to the sequence, like de-novo sequencing using spectrum graphs[14]. Some scoring schemes for MS-MS are spectral contrast angle, cross-correlation, rank-based, SEQUEST, SCOPE and OLAV.

3.6 MS/MS

MS/MS instrumentation has been mentioned in many occasions earlier in this chapter, and a better introduction for MS/MS itself is in its place. In the first step of MS/MS an ordinary MS is done. This results in a mass spectrum. The next steep before the second MS is to choose a range of M/Z value/peaks(precursor ions) to fragment from the first MS. These are usually the 3-8 most intense peaks in the mass spectrum. Each of these are in turn picked out and fragmented(into product ions) before the second MS is done. The precursor ions each produces a mass spectrum, called an MS/MS spectrum(Fig. 3.21). The peaks in these spectra represents charged fragments from the peptide in the chosen M/Z range(the product ions). These spectra can then be analyzed and part or the whole of the sequence can be found, since usually there is only one peptide for each M/Z range. Instead of using the most intense M/Z values/peaks some instruments give the choice of an inclusion lists specifying the M/Z values and for LC-MS the retention times they can be found at. Two LC-MS, or MS for that matter, is never the same which means some freedom in the retention time is needed.

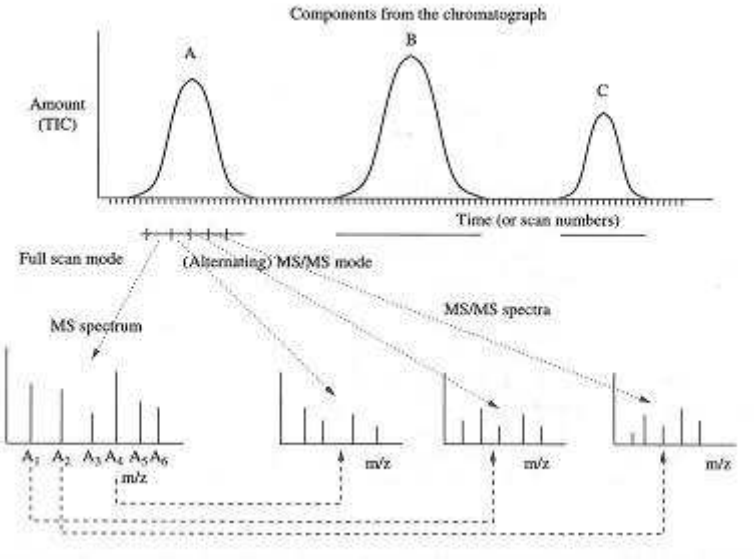


Figure 3.21: The upper part shows a mass spectrum where the three most intense peaks are chosen for another MS. The lower part shows the MS/MS spectrum for each of the three chosen peaks. Taken from [14].

4 Background: Quantification in Mass Spectrometry

Quantification in itself means to quantify the amount of one or more substances in a solution. This can be done in many ways and are usually only stopped by the ingenuity of the researcher/chemist. To mention a few that necessarily do not have anything to do with MS are immunologi/antibodies, titration, chromatography (gas and liquid), ultraviolet spectrometry etc. The use of a method depends on what you try to quantify. For proteomics it is proteins, these can be detected with 2D-PAGE, mass spectrometry, antibodies (if prepped for it), to an extent PCR(polynucleotide chain reaction), mass spectrometry(MS) etc. Of these MS is one of the least expensive methods to use and it is relative fast compared to many of the others. The two most used methods are 2D-PAGE-MS and MS-MS[33]. See chapter 3 for an explanation on these two. These two techniques are often used with something called labeled or unlabeled analysis. Labeled analysis is when a chemical is used to either modify(chemical labeling) the protein or is incorporated(metabolic labeling) into the protein. Unlabeled is when the analysis is done directly on the protein. For MS the quantification is done using the intensity of the M/Z values.

4.1 Metabolic Labeling

In metabolic labeling the technique of stable-isotope dilution is used. This makes use of the fact that different isotopes have different mass and can therefore be separated on their peaks in the mass spectrum. For polypeptide chains like proteins this means radioactive or heavy isotope of one or more amino acids, their building blocks C, H, N or O, or heavy salts[33]. These are put into the growth media and the cells in questions are grown on it.

One such method for stable-isotope labeling is called SILAC(Stable Isotope Labeling by Amino Acids in Cells). In this method leu-d3 (leucine modified with deuterium) is used in the growth media as free amino acids. The growth media is dialyzed so no other leucine than leu-d3 is present. The cell culture uses about 5 cycles to fully incorporate leu-d3 into their proteins, 5 days for the experiment in reference 38. The actual analysis is done on cell cultures being in the same or different growth state where one has been grown on leu-d3 and the other one has been grown normally. An example would be to analyze a tumor cell grown on leu-d3 against a healthy cell of the same species. This experiment could be used to gain insight into what proteins are expressed, up- and down-regulated in tumor cells. The next step is to mix the lysate of these two and digest the proteins with a protease. The cell cultures can also be purified before mixed if needed. Following this digestion is an MS analysis. The mass spectrum produced by the MS will have some peaks with a mass(M/Z at same charge) that is equal to the mass differences between leu-d0 and leu-d3. These can/will represent leu-d0(ordinary leucine) and leu-d3 proteins. Figure 4.1 shows the course of events for a SILAC experiment.

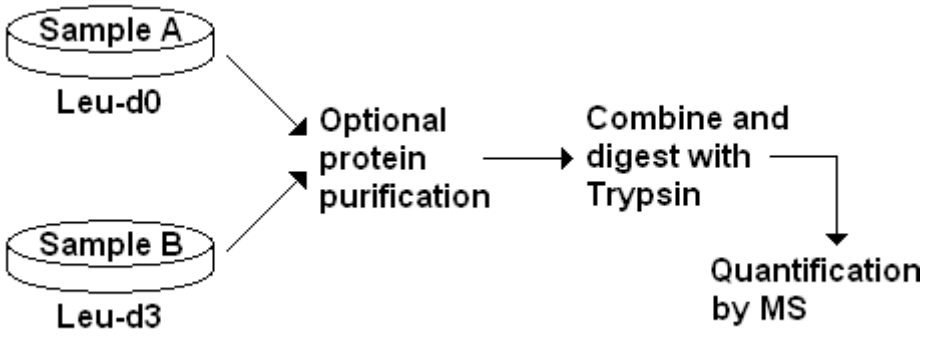


Figure 4.1: Shows the course of events for a normal SILAC experiment. Recreated from [40].

Cells grown on leu-d3 acts normally for the growth state they are in.

Mixing leu-d0 and leu-d3 with a difference of 1:3 shows up in the mass spectrum as 1:3(figure 4.2). The main drawback of this method is that the cells have to be grown so it can not be used on dead cell's proteins[40]. Another drawback is that it is time consuming, needs 5 cell cycles(here 5 days).

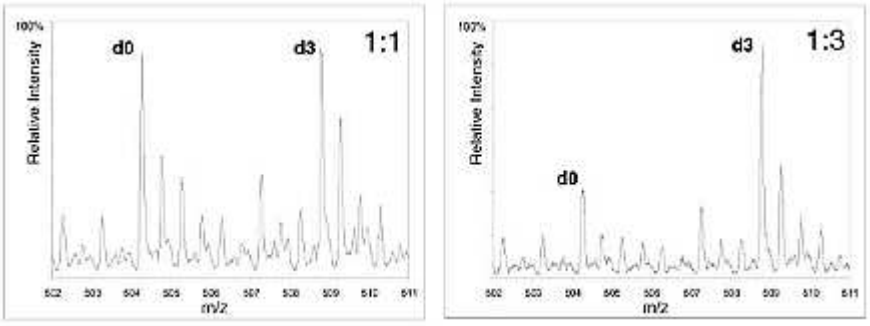


Figure 4.2: Both mass spectrum are from MS-MS and the marked peptide in question is SCNCLLLK. The mixture on the left was prepared with equal amount of d0 and d3, while the mixture to the right was prepared with 1:3. Both shows up with the right quanta in the mass spectra. Taken from [40].

Other ways for metabolic labeling is to use a growth media with ^{15}N instead of the ordinary ^{14}N(nitrogen). This method suffers from the drawback that the nitrogen ^{15}N is not 100% incorporated into all protein making it hard to analyze the produced mass spectrum. On top of that the growth media is expensive and hard to make for mammalian systems. This mostly reduces the method to microorganisms[40,41].

4.2 Chemical Labeling

Chemical labeling is to covalently bind a chemical to the substance that is to be quantified. These chemicals can then be seen in a mass spectrum by looking at mass differences between a modified and a ordinary peak of the substance. One such method is to enzymatically incorporate ^{18}O from water to peptides[33]. Two methods that needs to mentioned are ICAT(isotope coded affinity tags) and iTRAQ(which builds on ICAT).

ICAT consist of three functional elements, a specific chemical reactivity group, an isotopically coded linker and an affinity tag. The affinity tag has a specificity towards sulfhydril groups(as in cysteine) attached to a eightfold deuterated linker with a biotin tag on the end(Fig. 4.3).

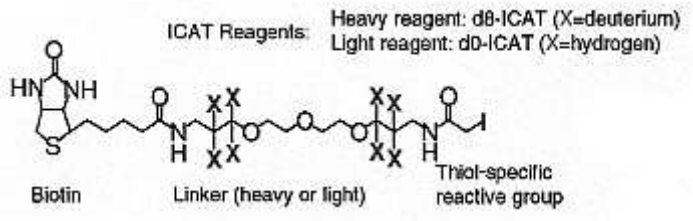


Figure 4.3: Shows a picture of the ICAT molecule. Biotin marker to the left, linker in the middle and the reactive group to the right. Taken from [42].

Figure 4.4 shows the course of events for ICAT. Two samples from one cell state are prepared. One with the light form of ICAT reagent and the other with the heavy form of the ICAT reagent. The mixture are mixed then cleaved with a protease to create small peptide fragments. The fragments with cysteine/reagent on them are separated from the rest of the mixture by using avidin affinity chromatoghraphy. This effectively reduces the complexity of the solution. The last step is then to use μLC-MS/MS to find both the quantity and the identity of the fragments with cysteine on. This information can then be used to find the parent proteins of the fragments. The first MS step finds the quantity, and the second MS step finds the identity of the labeled fragments. The whole process can be automated[42].

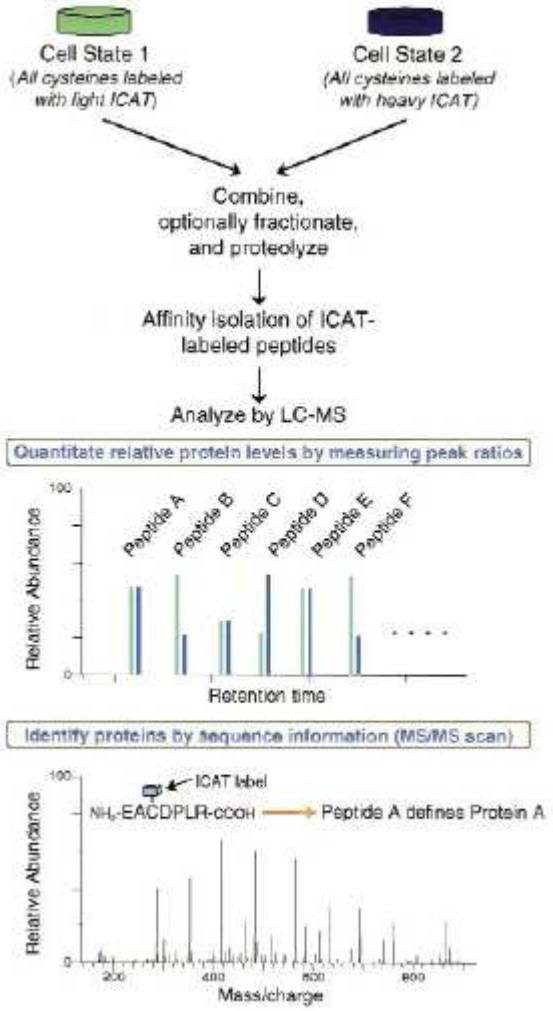


Figure 4.4: Shows the course of events for a normal ICAT experiment. Taken from [42].

iTRAQ is based on ICAT but differs in that it labels primary amines* instead of cysteine. Since primary amines are more abundant the method also produces a more complex result. This can in many situation be to complex if not coupled to some good separation methods beforehand. iTRAQ consist of four different affinity tags, meaning four different samples can be tried out in every experiment. These affinity tags have the same mass in a mass spectrum, but when exposed to CID(collision-induced dissociation) part of the sequence is cleaved of. This part is called the reporter and each tag have its own reporter tag on 114, 115, 116 and 117 M/Z in a MS/MS spectrum. This also means that the rest of the tag, called the balancer, for the four possible tags have different masses/compositions[43, 44]. Figure 4.5 shows an example MS/MS spectrum for iTRAQ.

*Primary amines are ammonia(NH_3) with a organic group bound to it instead of one of the hydrogens. This means that amino acids in the n-terminal have one, as well as arginine, histidine, lysine, aspargine and glutamine.

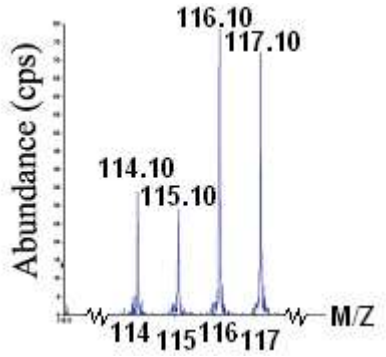


Figure 4.5: Shows the 4 peaks produced by the iTRAQ tags after they have been cleaved with CID. Recreated from [43].

4.3 Unlabeled

As the name says unlabeled analysis uses no stable-isotopes to quantify the substances in a sample, instead it uses algorithms, knowledge etc. to deduce the substances from the mass spectrum. Unlabeled analysis can at least be divided into two fields which is single MS and LC-MS(also called comparative LC-MS).

The procedure for single MS is to divide the sample into subsamples. Do a single MS on every subsample. Followed by normalization within the samples so they become comparable. Then the analysis starts by finding peaks with equal MS within the samples, taking M/Z distortion into account(imperfect resolution of the mass spectrometer). The interesting peaks can be put into a inclusion list and used in an MS/MS run.

Comparative LC-MS on the other hand requires a lot of the instruments upstream of the mass spectrometer as well as the mass spectrometer itself. A typical work-flow for such a method is to have a data collection step first, by running LC-MS alone. From the mass spectra the intensities are used as a measure for the quantities of the different proteins/peptides. After this the analysis software does its job creating an include list(peaks that needs further investigation), which can be used to pick out the peaks in the sample in an MS/MS run.

Statistics like multivariate analysis can be used in comparative LC-MS, but then the LC-MS has to be done many times under as close to equal conditions to produce enough data. After these runs the data needs to be normalized in the different runs so they are comparable. Other methods includes peak alignment which consist in finding peaks that are in more than one dataset(LC-MS runs), where varying M/Z due to technical drifts and varying retention times can complicate the analysis[45]. All of these methods needs a lot of preprocessing to get good results. Preprocessing consist of peak detection, feature removal, normalization, mass spectrum evaluation, retention dewarping[46] etc (see MS chapter 3.4 for details on these, except last two). Some software also uses visualization as an aid[45].

Most unlabeled methods have only been tested on their own dataset and there has now been done an effort to compare these against the same dataset. The dataset to be compared against has been chosen because features are already known, by using MS/MS, and the data set has gone through a lot of data preprocessing before fed to the unlabeled methods. The datasets themselves have already been used on the algorithm OBI-Warp[46]. Figure 4.6 shows the results for 6 algorithms which are OpenMS, msInspect, SpecArray, Xalign, XCMS and Mzmine[chapter 2.3 in 46].

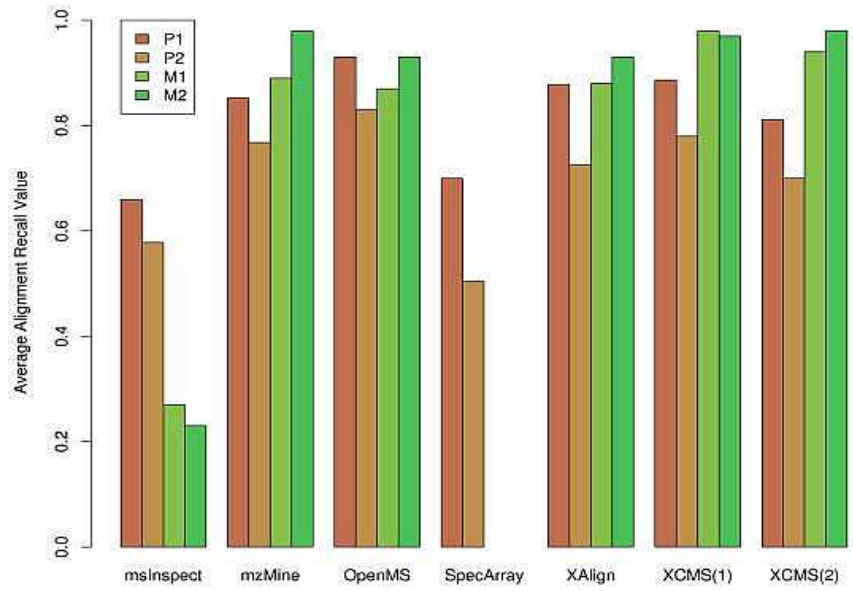


Figure 4.6: Taken from [46]. P1 and P2 are the two proteomics datasets, while M1 and M2 are metabolomics sets. The picture shows that OpenMS do well on both P1 and P2. The other programs except msInspect and SpecArray do very well to. A note is that only the alignment part is tested. Taken from [46].

As shown in the figure 6 these results are based on something called recall and precision. The first one is the probability that a feature(protein) is found while the second one is the probability that a found feature(protein) is relevant. The formulas for these can be seen below[46]:

$$\text{Recall}_{\text{Align}} = \frac{1}{N}\sum_{i=1}^{N}\frac{|\text{gt}_i \cap \tilde{\text{tool}}_i|}{|M_i| \cdot |\text{gt}_i|}. \qquad \text{Precision}_{\text{Align}} = \frac{1}{N}\sum_{i=1}^{N}\frac{|\text{gt}_i \cap \tilde{\text{tool}}_i|}{|\tilde{\text{tool}}_i|}$$

, where gt_i is the consensus features(from base set), $tool_i$ is the features from the used software and M_i is a penalty to break up a feature in the consensus set.

4.3.1 OpenMS

OpenMS is an open source framework made in ANSI C++ and should be easy to port to different platforms. This framework aims for ease of use by using an object oriented programming language to code the classes, and a software to generate class documentation. OpenMS are also trying to make the framework perform well while still being robust. Figure 4.7 shows the overall design of OpenMS[47].

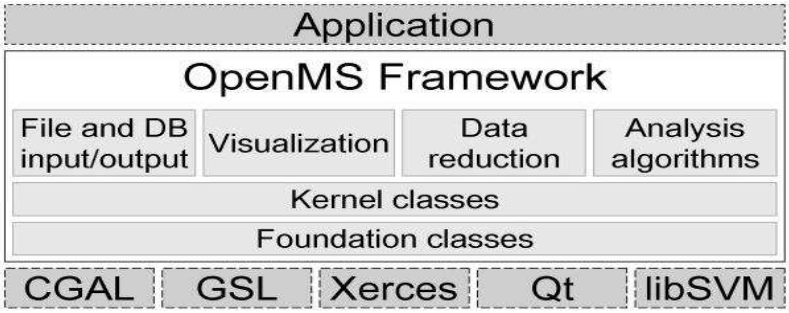


Figure 4.7: The overall design of OpenMS. The lower part containing CGAL etc are external libraries used in OpenMS

.

OpenMS supports most non-proprietary formats, like mzData and mzXML. It also supports a database but does not use it at the moment. Visualization of the mass spectrum is supported both in 2D and 3D. It supports signal processing with different noise reduction filtering, like Savitzky-Golay and Gaussian low pass filter, and baseline correction. For peak detection they uses their own algorithm based on wavelet as a first step and then an asymmetrical peak function is fitted to the raw data to extract important peak information(centroid, area signal to noise ratio etc) as a second step. OpenMS feature detection and quantification, for LC-MC, gives each feature characteristics. These are mass-to-charge ratio, the centroid of its elution curve and the signal area. The framework consist of many algorithms suited for different instruments and mass resolutions. Their concept are based on 2D data and uses what they call an averagine, which is a representation of an average amino acid, to approximate the amino acid composition for a given mass. Followed by an estimate of the the atomic composition are calculated, and used to find the isotopes in the mass spectrum. Eluation is approximated by a Gaussian or an exponentially Gaussian distribution. These computations are expensive so candidate regions has to be selected carefully. The raw data is stored in a RawMap and the extracted features in a FeatureMap[47].

The multiple alignment algorithm in OpenMS uses two phases, called superposition and consensus phase. This architecture allows different algorithms to be implemented for each step, allowing more flexibility. The superposition phase, superimpose RawMaps or FeatureMaps, uses an algorithm called pose clustering, which is an algorithm based on the general paradigm for point pattern matching. The algorithm considers intensities as well as the different measuring accuracies of the m/z and retention dimensions. This is then used to find the optimal transformation. Where the definition is to try to map as many features in one map as close to features in the other map by using a voting schema to determine the result. This is followed by finding landmarks in the the superimposed maps and using these to refine the results with a linear regression step.

The consensus phase uses the data in the superposition phase to generate a consensus map using a nearest neighbor search. The result is a star-alignment map, where the most complete map acts as the reference map[46, 47].

4.4 MassLynx

MassLynx is a software suite containing programs like MassLynx MS, TargetLynx, ChromaLynx etc. Here we will take a deeper look at MassLynx MS[48].

4.4.1 MassLynx MS

MassLynx MS is a program created to help researchers to record and interpret mass spectra. Quantification is among the interpretation part. It contains methods like smoothing-, subtraction of a polynomial-, centering of peaks-in a mass spectrum etc. As well as a peak detection method. It lets you record mass spectra from MALDI, ESI and a lot of other often used instruments. This recording can be done in batch mode, meaning you can queue the jobs you want done. The files from a recording is called .raw files. The .raw file contains these types of files:

- _Header.txt Data file header information
- _Funcs.inf Information on functions acquired
- _history.inf Information on how data has been processed
- _expment.inf Experimental record information.
- _Func001.dat Data file for first function (one for each function)
- _Func001.idx Data file index for first function
- _FuncXXX.dat Data file for the XXX'th function
- _FuncXXX.idx Data file index for XXX'th function
- _proc001.dat First processed data file (one for each process)
- _proc001.idx Index for first processed data file
- _procXXX.dat XXX'th processed data file
- _procXXX.idx Index for XXX'th processed data file

The largest of the function files are usually the file containing the first MS part of the MS/MS experiment. One important note is that the .dat files themselves are in a proprietary format and this is in binary. Fortunately MassLynx MS have a tool called Databridge bundled with it. This handy little program can convert from and too MassLynx's raw format from a couple of different formats, but not mzXML(an often used XML format for MS).

Opening a raw files in MassLynx MS, set of mass spectra, takes you to the chromatography window. It also opens the first mass spectrum in the raw file(Fig. 4.8). The chromatogram is based on TIC(total ion count) for each spectra, where the highest TIC among the spectra are used to normalize the TIC for every spectra between 0 and 100%(upper part of figure 1). For the mass spectrum the highest intensity of all the m/z values are used to normalize the other values between 0-100%. These two windows give you access to methods to manipulate its respective content, as well as a feature to zoom(by marking the area), although this one is implemented in such a way that it is hard to use.

If you see a chromatogram peak that is interesting, you can double click on it and the mass spectrum for the corresponding retention time, is opened in the mass spectrum window just above the previous spectrum. The mass spectrum window can become very crowded if you do this many times, and every spectra in it will share the same axis and current zoom. In the mass spectrum windows you can search for a mass spectrum on its retention time. If the retention time does not exist it takes you as close as it can. You can also show the peak list, but for some reason MassLynx have decided that it is important to show m/z values as integer values and the intensities as decimal values. The list also contains the TIC and the BPI(base peak intensity, same as TIC but around the most intense peak in the area) value for every peak. This list can be saved easily. The program have a lot of other options that is more for the advanced user.

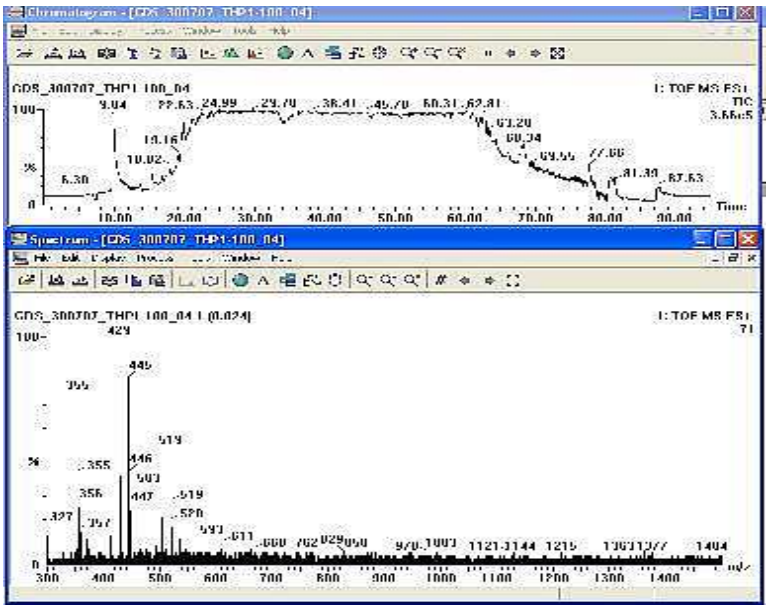


Figure 4.8: This figure shows the chromatogram on the top, using TIC to make it. While on the bottom a mass spectrum is shown with filename to upper right. Retention time is show in parenthis after the filename and the number before the parenthesis is the mass spectrum number.

A typical work flow for MassLynx MS(Fig. 4.9) could be to record the mass spectrum, and MS/MS spectra via MassLynx. Then you could start to look at the chromatogram, picking out interesting peaks. The mass spectra from the picked peaks could then be further analyzed with some smoothing/noise reduction and a peak detection. From here the real analysis could start by picking out peaks that is interesting, analyze isotopic envelopes and so on. MassLynx is very handy if you like to analyze everything yourself, but if you do not have time for this MassLynx does not offer much in global analysis.

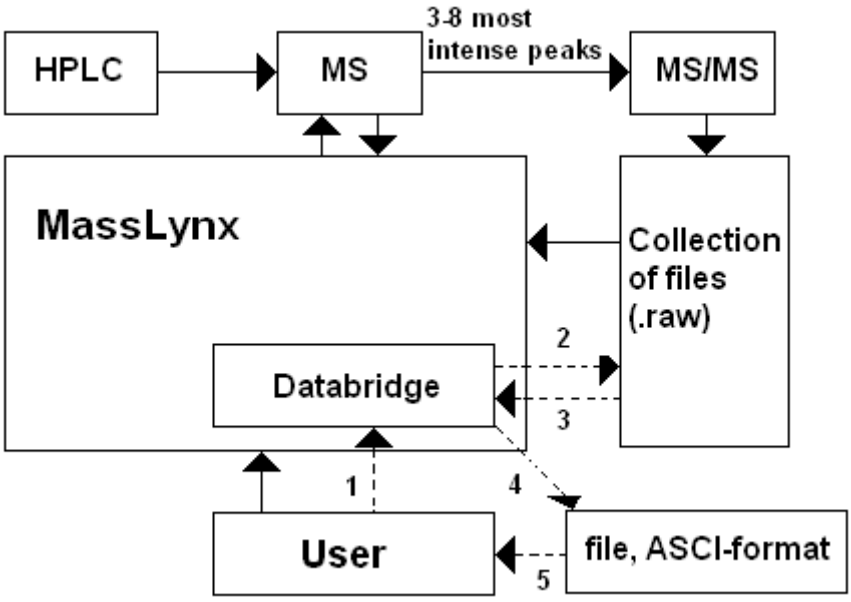


Figure 4.9: Shows a work flow for a user from MS to .raw files. The same user also wants to convert his data to ASCI so that he/she can be able to read it an ordinary text editor(this is showed with stippled lines). The numbers 1-5 shows the order of the work flow where the user converts a .raw file to ASCI.

MassLynx MS as a program is at least partly written in Java, where Java 2D is an essential component. As a side note MassLynx also supports macroing via visual basic.

5 Work Flow of the MassAnalyzer

The Pair Finder analysis method in the MassAnalyzer program(MAP) will analyze mass spectra taken from SILAC experiments. The input parameters will therefore be the mass spectra and the parameters for the program. The way from raw spectra to marker pairs are shown underneath:

1. Create a project, containing the information needed to load a set of mass spectra.
2. Fetch and convert the spectra from the data file.
3. Set the input parameters for analysis methods in the program
4. Pre-process the mass spectra
5. Find the marker pairs in each mass spectra
6. Present the results

Before we can start to discuss these points in detail we have to take a look at the main window of the GUI, how the the data structure for mass spectra is built and what a marker pair is.

5.1 Introduction to the GUI

All the user interaction is done in the program's GUI, main window is shown in figure 5.1. The main window can be seen as divided into three mayor parts.

1. The result list(RSL) and result panel, shows the results when an analysis have been done, shown on the right and middle in figure 1
2. The ms list(MSL), list of mass spectra currently in the project, shown on the right of figure 1.
3. The template, containing the analysis methods used for an analysis, shown on the bottom in figure 1. With the name of the current used template in its border.

Here a project means a collection containing a specific MSL, template and RSL saved in a specific location on the computer's hard drive.

The main window also have a menu at the top. This menu contains a file menu, where the user can save and load any of the mayor parts of the project. The edit menu contains the options that can be done by buttons inside the GUI. The help menu only contains the version of the program. Both the file and edit menu contains sub menus for the different mayor parts. A note about the menu is that it is easily updated if new options are implemented. The other parts of the GUI will be discussed as they are used by the user.

There are hundreds of way to build a GUI, maybe one of these are better than all of these, but usually it is more up to the taste of the user. It is beyond the scope of this thesis to create a GUI that the user can move around and do as he/she wants with. This means this part of the GUI wont be discussed in detail, and was put together on the basis how the author would want it to look and feel like.

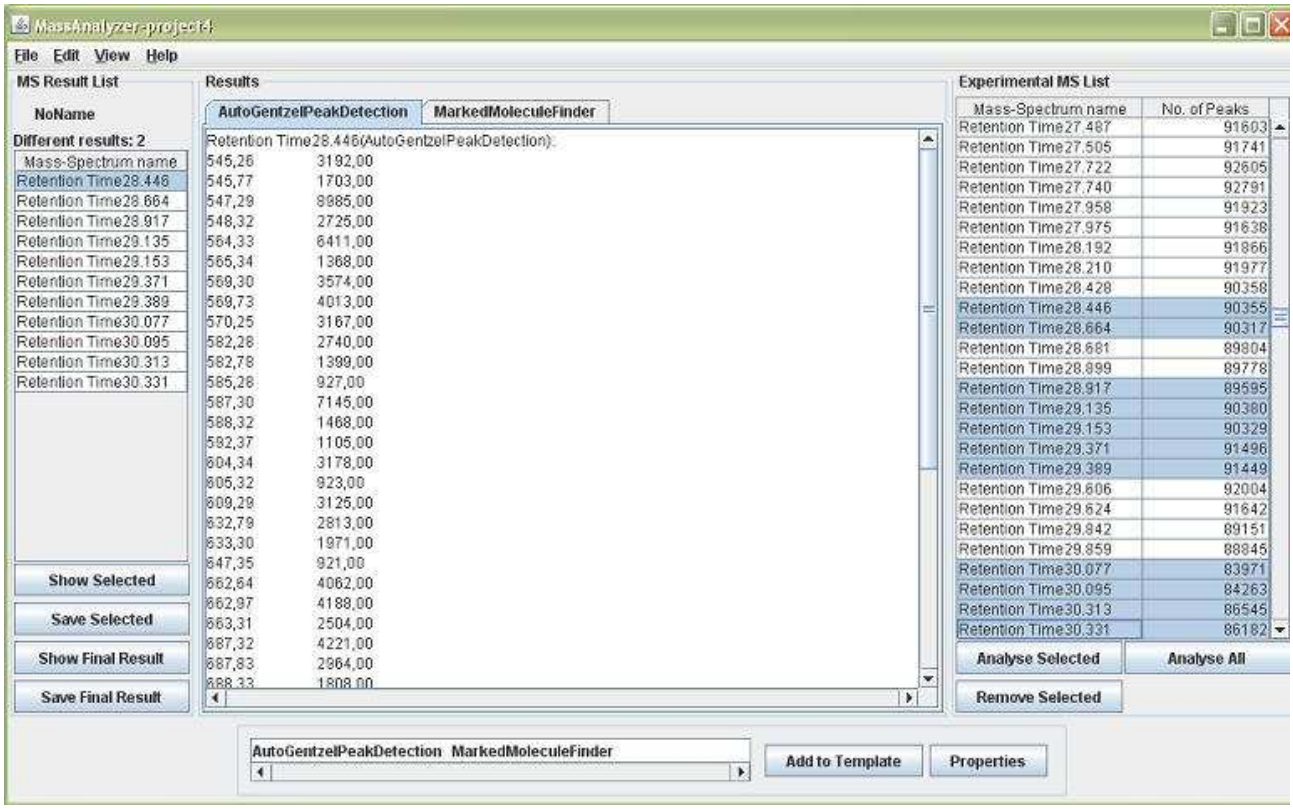


Figure 5.1: GUI to the main window of the project called 'project4'. The result list is to the right, results in the middle, mass spectra list from an experiment on the right and the analysis method template in the bottom.

5.2 Mass Spectrum Data Structure

A mass spectrum usually comes as a list of pairs, where a pair contains a M/Z and intensity value. There are more than one way to build a data structure for a list of pairs. The most memory effective would be to make an object linking up two lists of primitive type float or double. This would most probably be the fastest implementation to. Only one problem though, the mass spectra that was provided, and is fairly usual these days, have 56000+ pairs. This means it can be a problem to fit it into primary memory(memory on cpu) when working on it, and might result in methods for partially loading the spectrum would have to be made. Another way to make the list is to make a list of objects that will contain the pairs. This will take some more memory to make as well as some more memory to store, but it will be more easy to fit into memory at run time. It will also be more easy to send the pairs to other methods since the object for them are already made.

MAP stores the mass spectrum as an object structure, and its called 'MassSpectrum'. The structure that MAP uses for the pairs is also of the type object, and is called 'MZI'(M/Z, intensity). There are more than one variation of the 'MassSpectrum' structure depending on where it is used in MAP. The MSL uses a 'ProxyMassSpectrum'(based on proxy pattern[49]), and it only stores the name, number of elements and where to load the mass spectrum from. The RSL uses a 'ResultMassSpectrum' which contains another object with variable amount of information that would not be sensible to store in different types of 'MassSpectrum'. This could be information about the previous used analysis method, such as the most intense peak etc. The analysis methods in the template uses the ordinary 'MassSpectrum' structure, which the raw mass spectra also uses after they have been loaded.

5.3 Marker Pair

A marker pair is a pair of peptides/peaks with a set mass/(M/Z) distance from each other and sharing the same charge of their isotopic envelopes. To explain this consider a mass spectrum that has been through peak detection but not deisotoping. This mass spectrum will contain areas with isotopic envelopes(Fig. 5.2). If two such areas contains peaks within a set distance of each other and the isotopic envelopes share the same charge these can be gathered into one marker pair as shown in figure 5.2. One peak from each isotopic envelope with the set distance in this marker pair is called an isopair, also shown in figure 5.2. This means that a marker pair can be seen as many isopairs.

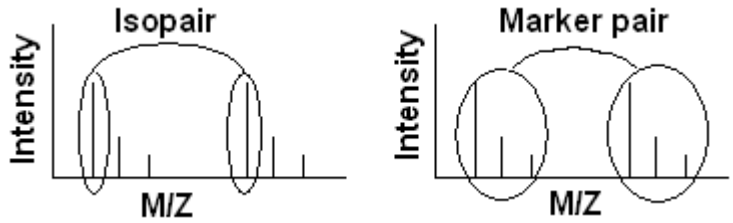


Figure 5.2: The left part shows an isopair, while the right part shows a marker pair.

5.4 Create a Project

MAP lets the user create a new project by using the project tab and choosing new under the file menu in the program's main window. When creating a new project the user gets a screen(Fig. 5. 3) with the following choices, where all of them have to be filled in to be able to create the project.

Figure 5.3: Shows the window for a new project and the the information that needs to be filled in to create a new project.

The first step is for the user to name the project and set the current folder where the project should be saved. This ensures that the save command will work for the different part of the program that can be saved. All of these parts can of course also be saved another place by using their respective 'save as' command.

The second step is for the user to give a path(folder) where the MassAnalyzer's mass spectra files should be stored. This folder will also contain an .ml file which is a list over the mass spectra for this project. The .ml file can be loaded directly into the program letting the user get a list over the mass spectra for that project, and use it in any other projects.

The third step is for the user to specify what file contains the mass spectra to be loaded. This file should be an ASCI file from the program Databridge(chapter 4.4), or at least follow the same format as such an ASCI file. How to use the Databridge program is given in appendix E.

When the user presses the 'Make Project' button the program will collect the parameters and use them in the program before proceeding to the next step.

5.5 Fetch and Convert Mass Spectra from Data File

After the user pressed the 'Make Project' button the program loads the file and starts converting one and one spectra. The files are converted from ASCI to binary, which later will seem to be a predicament since the files was in binary in the first place(chapter 5.5.2). This conversion is done to ease the load times of single spectrum or a batch of spectra within the program, since the whole file can not be loaded at once(ASCI file is close to 2Gb). As the mass spectra are loaded the data needed is added to the 'ProxyMassSpectrum' data structure. These objects are then stored in a list, which is used to create the MSL in the main window. When the conversion is finished the list is displayed in the main window and the user can start to interact with the program again. This procedure will make the creation of the project take some more time, but will make the analysis run a lot smoother.

5.5.1 Algorithm(s)

The program can load many types of mass spectra, which are as objects(created in this program using Java's serialization, chapter 7.5.2), text taken from within MassLynx, ASCI files from Databridge and two binary formats. Where one of the binary formats are only for raw spectra. The ASCI and the two binary formats have the same algorithms but different classes and/or procedures to decipher the contest of the files(Algorithm 1).

Algorithm 1. ASCI and binary loading of a mass spectrum from a file

```
var
   MassSpectrum MS
begin
   for each pair in the file          //the difference in the algorithms is what it
                                                considers as a pair in the file
      store the pair in an MZI pair
      add MZI pair to MS
   end
end
```

Loading an object file on the other hand is a Java implemented class(Serializable interface, see chapter 7.5.2), but in the end this one to is an iterative process like the other two.

The program contains more than one way to save a mass spectrum. It can save it as an object, text and two different binary format, where one of them are used for raw spectra only. They all follow the same iterative process but ends up saving them as different files(Algorithm 2).

Algorithm 2. Binary and ASCI saving of a mass spectrum to a file

```
var
   MassSpectrum MS
begin
   MS = mass spectrum created with the program
   for each MZI pair in MS
      save pair to file          //the difference in the algorithms is how they save the pair
   end
end
```

Both algorithms runs in a single loop with no special implementations in them, and therefore runs in O(n) time.

A note on MAP in general is that it uses the bridge pattern[49] to make it possible to switch load/save methods at real time if wanted.

When the program fetches the spectra it uses the ASCI loading algorithm and when it converts the spectra it uses the binary saving algorithm for raw spectra.

5.5.2 Discussion

As with all programming there are more than one solution. The one chosen here is not necessarily the best one. Lets take a closer look at the problems compared to the current solution.

As explained in chapter 4.4 MassLynx has its own proprietary format in binary. The files for an LC-MS experiments are often around the size of 1Gb or higher, depending on the experiment. Binary format means that every character, decimal, integer and so on are saved as their bit representations. This means that two character, 2bytes each, could also represent one decimal, double=4bytes. As you can see for a 1Gb file there could be ~2.56*108 decimals, the double amount of characters or a mixture of these two. If integers and float, two other primetive data types are used then the possibilities for a mixture of all these becomes vast.

From here there are really only two ways to go, get the data format syntax from the vendor or have some insight into the data structure and go from there. The latter might be fine, but for all we know the insight we got in one file might not work for another, depending on if all the syntax is used in the file(s) the insight was gained from. This means that the insight might work 90% of the time, but since we are talking about massive files often containing as much as 4300 mass spectra some random errors could be quite hard to find since a program that reads the file will always be able to read the decimal value even if there should have been two characters there. This complicates the whole procedure of reading a MassLynx .raw file directly.

MAP wont be able to load the whole file into memory, unless some Java parameters is tweaked which by itself is easy enough. Tweaking these to high might as well make the computer MAP is run on get into memory problems, and as a fact it would use a lot more memory than necessary. On the positive the analysis in MAP would probably run faster, but at the expense of 1Gb+ memory. The only reason such a road should be taken would be if all the mass spectra should be analyzed in such a way that all the data needed to be available at all times.

MAP could in theory run on as little as 10mb or less, ~1Mb if the GUI was excluded and the result was saved directly as text, and in the authors eyes it is not worth having it steal all the ram just because it can.

For MAP the first step that has to be taken, unfortunately, by the user is to convert the file from MassLynx to ASCI format with a program called Databridge(chapter 4.4).

The reason that ASCI format was chosen, over mzXML(a free XML, extended markup language, format) was that in the early incarnations of MAP the file was read directly, creating new mass spectra just before they got analysed. This of course is a bad solution if you want some freedom in what spectra you want to analyse. An example would be reading from ASCI and choosing only the last spectra for analysis. This would result in reading through the whole file, but only use the last spectra for analysis. Another reason for ASCI in the early stages is that it was easier to interpret and takes less space than the mzXML format. Also the program MassWolf[50], that converts MassLynx files to mzXML, is harder to understand for the ordinary user than Databridge.

The next step for MAP is to save the mass spectra as binary files. This sounds redundant, but it saves time in the long run. How, by making the spectra easy to load one by one at run time. If the mass spectra was saved as ASCI every MZI pair in them would have to be converted from text to decimal values. For one spectra this would mean about 112000+ conversions. This is quite a lot of work even if only a few spectra was chosen for analysis. The binary format in this program follows quite simple rules(chapter 8.3.5). Each spectrum is also saved as its own file in the specified folder. This is to make the removal and later on adding of a single spectrum to a project more easy. In one big file removing a spectrum would, 1. let it stay in the file but not get used or 2. remove it from the file resulting that the index would have to be updated for all indexes after the removed one in the file.

This back and forth is actually a little harder than it could have been. As mentioned earlier there is a format called mzXML. This format might have been better to choose from the start as the main format, and if time had permitted it the format would have been tried out. The program MassWolf might even be possible to run automatic in MAP. If it is better only depends on one thing, can XML(extended markup language)[51] files be indexed in an easy way so that the program can easily read from the middle of the file. The answer is yes, XML can use its inherent tag structures for an automatic index, or if the class parsing the XML allows it, it can be indexed on byte level just as binary. The same goes for ASCI if it has an internal structure, and if the class that reads the ASCI file allows for byte indexing. XML though needs to be parsed just like ASCI and this might give some overhead, but XML allows for binary blocks meaning that the overhead for reading a mass spectrum will only be the reading of its tags and some few properties. All in all this should be enough to consider implementing mzXML as the standard input in the future.

5.6 Choose Analysis Method and Set its Parameters

At this step all the files have already been 'preloaded' into the MSL feature of the program. The next step is to choose analysis methods and set the parameters for these. To do this the user have to press the 'Add to Template' button. This will bring up a new window(Fig. 5.4).

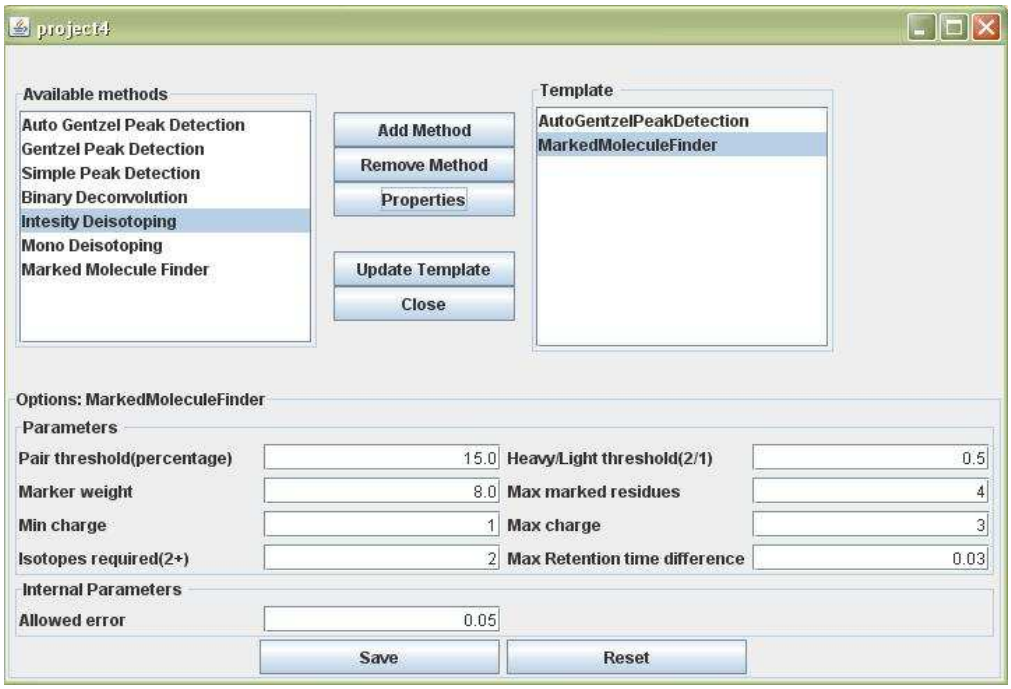


Figure 5.4: Shows the add to template window. To the left is a list of the available analysis methods, to the right is the currently chosen analysis methods and at the bottom the parameters for

To the left in figure 5.4 all the possible analysis methods, to manipulate a mass spectrum, are listed. Selecting one and pressing the 'Add Method' button puts it into the template list on the right side of the window. It also updates the lower part of the window to show the parameters for the added analysis method.

The parameters are divided into two categories, internal and external(on figure 4 just called parameters). The internal parameters are those that are modifying the inner workings of the analysis method, but in such a way that the user most often do not want to change them. The external parameters on the other hand lets the user manipulate the analysis method but in ways that should more or less be known to them. An example would be for peak detection; the internal parameters might adjust parameters based on the mass spectrometer the user recorded the original mass spectrum in, while the external parameters could be the threshold to consider a peak a true peak and not noise. Pressing the 'Reset' button will reset the parameters to what they previously were, and 'Save' will save the new parameters.

The order the analysis methods are chosen in will be the order they will be executed in. In other words the one of the top of the list to the right will be the one executed first, while the bottom one will be the last executed. To really use the template for a future analysis the user have to press the 'Update Template' button or the template in the main window wont update.

Pressing the 'Close' button or the x in the upper right corner closes the window. Now the template panel in the main window will show the chosen analysis methods. Selecting one and pressing the 'Properties' button will open a window that lets the user adjust the parameters, pressing the 'Save' button in this window updates the parameters for the chosen analysis method.

5.6.1 The Template System

A template based system was chosen for how the analysis methods was sown together. This means a list is used to hold the analysis methods, which themselves are object with parameters as the only variables. With the exception of analysis methods that allows a global result. These often holds an extra data structure to keep track of the results as the mass spectra gets analysed. All the parameters of the analysis methods are themselves objects that takes text as their way of setting their value, and they can return the value as a decimal, integer, string(text in Java) or boolean. The returned value depends on the object class used. This makes the parameters relative flexible when used by an outsider to create an algorithm(which should be fairly easy to do). The template approach was chosen since the algorithm themselves can be nested in different ways. Sometime the user will need a noise reduction before the peak detection, then the user can just add them together in that order. Other times the user might go for peak detection, normalization and some specific higher algorithm like finding pairs in mass spectra created by SILAC. Polymorphism is used to create the algorithms, meaning inheritance is used. All the algorithms inherits one interface, meaning they all need to have some specific methods implemented. This makes it possible for the program to store them all in the same list and run the same methods on them without any casting from one type of object to another.

5.6.2 Discussion

The template panel and template window was made in such a fashion that it should be fairly easy to get a hang of what can be done in it. As mentioned for the main window there are always a lot of ways to implement a GUI, and usually some users will agree it is a good implementation and others wont. What is worth mentioning here is the way the program handles updates to the parameters and

why it does so. As mentioned in 5.6 the user have to press the 'Update template' button to update the template in the main window. The decision for this was based on that the user might want to fiddle with the parameters while the program was running an analysis. If this was possible without a copy of the template this would have dire consequences for the analysis in question, since spectra might be analysed based on different parameters or even with different algorithms. A better method was implemented in the end, but since the work on the GUI itself and sowing the program together got less than a months work, it was not finished. The program works fine but in a suboptimal way when it comes to updating the template. The small but important step added in the end was a copy of the template for the analysis part of the program. This means that the template now should be available for direct modification without having to press the 'Update Template' button, but it is not fully implemented.

The basic underlying structure could have been implemented with the user only able to change the parameters for a peak detection and a SILAC pair finder algorithm. This would result in a less complex program but the future for it would be grim. An example would be, the user finds out that the peak detection method alone is not good enough to remove noise and unwanted peaks. So the user tells some programmer the problem, or even start to fiddle with the code himself(if possible). The user will then get the dilemma, do we add an algorithm before the peak detection or do we further modify the peak detection method to take care of the new problems. The first solution surely looks like a less flexible way than the one already implemented in MAP, and the second one makes it harder to use the new algorithm for other types of analysis. All in all the current implementation is flexible, but might in some cases require a little bit more of the user.

5.7 Pre-process the Mass Spectra

After the user has created his/her template it is time to press one of the analysis buttons. There are basically two ways to analyse the mass spectra. One, the user can analyse them all, by using the current template, by pressing the 'Analyse All' button. Two, the user can analyse a set of selected mass spectra from the 'ms list' by pressing the 'Analyse Selected'. The rules for selecting are standard for the operative system. For windows that means shift to select many in a row, ctrl to select one and one as you click on them and both to combine. When the mass spectra gets analysed they go through analysis method by analysis method, as explained in chapter 5.6.1. The analysis methods that have been implemented at this point is:

- Two deisotoping analysis methods, one for mono isotoping and one for intensity isotoping.
- A deconvolution analysis method.
- Gentzel's peak detection[34] in two versions. One with a set threshold independent of the current spectra and one with a threshold dependent on the current spectra.
- A simple peak detection analysis method.
- Pair Finder aka Marked molecule finder analysis method, discussed in the next 5.8.

5.7.1 Deisotoping

The two deisotoping methods are based on the find them all principle. This means that no rules or such are used to divide the intensity for peaks that has been registered among more than one peak, and no peak take precedence over another. Both analysis methods starts by running through the mass spectrum(preferably one after a peak detection method) once for every possible charge of the isotopic envelopes. For every possible charge all the possible envelopes are found and stored in a simple data structure. This structure contains a list of lists, where the inner list contains the peaks in

the envelope as well as the charge of it, and the outer list contains all the envelopes. The only real difference is what is registered as the final M/Z value. For mono isotoping this is the first peak in the envelope and for intensity isotoping it is the most intense peak. Both are shown in algorithm 3, with a / dividing the differences between them.

Algorithm 3 mono isotoping/intensity isotoping

```
var
    MassSpectrum isotopeList
    MZIIsotopeList tempEnvelope                 //the isotopic envelope structure
begin
    for each possible charge
        for each pair(v) in the original mass spectrum
            tempEnvelope = find envelope for the charge with v as start, ignoring values already found
                           to be in an envelope
            register the first/most intense peak as the main peak in tempEnvelope
            add tempEnvelope to the isotopeList
        end
    end
    add all the peaks that has not occurred in an envelope to isotopeList
    sort isotopeList on M/Z
end
```

5.7.2 Deconvolution

There are only one deconvolution method, and it is based on searching over different charges for peaks that can belong to the same peptide. It starts at the highest M/Z value and says that this one is considered to be of a charge. Then it searches the mass spectrum to see if it can find its counterparts with higher charge(ergo lower M/Z). This is done for all the M/Z values. The deconvolution analysis method is shown as algorithm 4.

Algorithm 4, deconvolution

```
var
    lowMZ = the lowest MZ in the current mass spectrum
    currentMZ
    MassSpectrum result                 //returned by the algorithm
begin
    for each pair(v) in the current mass spectrum, starting at the highest M/Z
        for each possible charge
            currentMZ = v divided by charge
            if currentMZ is higher than lowMZ
                do a binary search to see if currentMZ is in the current mass spectrum,
                    if such a value exist and the charge of it is the same as charge add it to result
                    else discard it
                    end if
                end
            end
        end
    end
    return result
end
```

5.7.3 Peak Detection

To both the analysis methods in the chapter 5.7.1 and 5.7.2 a good peak detection method is needed. Three such analysis methods have been made for MAP.

The first one is an analysis method based on Gentzel's peak detection method, and is fairly well explained in chapter 3.4.3. MAP's analysis method differs in some ways though. In that it uses a threshold based on the highest peak in the current mass spectrum, and also does some heuristics instead of 'saving' the results into structures followed with a processing of these. MAP's peak detection goes through the mass spectrum twice. First time is to find the M/Z value with the highest intensity. Then make a centroid peak around it. This is followed by taking the percentage, given in the parameters, of the centroided peak and call it an intensity threshold.

The second time it uses this intensity threshold to find the peaks, by using a sliding window equal to pw(x) (chapter 3.4.3) and accumulate the intensities. If the analysis method does not find a peak, it removes the values of the M/Z in the low end and adds the extra in the high end. This ensures that an area is not counted twice if a peak is not found. On the other hand if a peak is found the iteration, over the mass spectrum, continues at the end of the found peak. Both these ensures that the mass spectrum is only traversed once for the peak detection procedure if no peaks are found, and up to twice if the whole mass spectrum contains peaks following each other. Once for the detection and once for the centroiding. See algorithm 5 for the process.

The second one is also a Gentzel peak detection method but skip the searching for the most intense peak and let the user set the threshold(minimum intensity) directly. This analysis method is basically the same as Algorithm 5. minus the two first step after **begin**.

Algorithm 5, modified Gentzel's peak detection

```
var
    threshold                   //a percentage of the most intense peak, set in the parameters
    accumulatedIntensity        //total intensity in a sliding window
    highPeak                    //highest intensity in a sliding window, also M/Z value
    MassSpectrum result         //returned by the algorithm
begin
    find the highest intensity value MZ in the mass spectrum
    threshold  = a percentage of the intensity of a centroid peak around MZ
    for each pair(v) in the current mass spectrum
        accumulatedIntensity = accumulate all the intensities in a sliding window based on the peak
                               width pw(x) (see chapter 3.4.3)
        highPeak = the peak with the highest intensity in the same window as above
        if accumulatedIntensity >= threshold
            make a centroided peak around highPeak and add it to the result
            set v to be at the end of the added peaks width
        end if
    end
    return result
end
```

The third peak detection method is a little bit more naïve. It does much of the same as the Gentzel method. It iterates over the mass spectrum, but instead of using pw(x) as a sliding window it uses a fixed value. This analysis method also have a threshold as the second Gentzel and uses it in the same way, but when a window passes the threshold this peak detection skips the centroiding and uses the highest M/Z as the apex of the peak. The intensity on the other hand is calculated by following the peak, on both sides, from the apex until it reaches the valley between two peaks or the baseline. On the way down it ignores small increases in the intensities, while adding the intensities as it goes. It will also subtract the next peaks contribution to the current looked at peak. After this it will start to find the next peak using the same procedure again, but with less intensity for the part shared with the previous peak. Algorithm 6 sketches out the process.

Algorithm 6, simple peak detection

```
var
   accumulatedIntensity          //total intensity in a sliding window
   highPeak                      //highest intensity in a sliding window, also M/Z value
   MassSpectrum result           //returned by the algorithm
   nextPeak                      //the intensity the next peak will be reduced by
begin
   for each value v in the current mass spectrum
      accumulatedIntensity = accumulate all the intensities as the peaks goes upward and then
                             goes downwards until it reaches a valley or zero intensity
      highPeak = the apex of the peak found when accumulatedIntensity was found
      if valley is found
         add to accumulatedIntensity  a part(nextPeak) of the next peak by using linear regression
         move v to the end value of the linear regression
         subtract from  accumulatedIntensity a part of the next peak by using linear regression, and
                add this part to nextPeak
      end
      add the peak to the result by using the highPeak's M/Z as the peak apex and
            accumulatedIntensity as the intensity
       accumulatedIntensity  = nextPeak
   end
   return result
end
```

5.7.4 Discussion

Before the discussion starts it is in its place to say that both deisotoping and the deconvolution analysis methods are all tests to see how the Pair Finder (aka Marked Molecule Finder) can be implemented properly. The simple peak detection is also a test of sorts to see how a peak detection method can be implemented and how the intensities of peaks can be shared.

Both the deisotoping methods are a little naively implemented, and they both need a preprocessing step behind them if they are going to be fully working. They both also remembers the peaks for each envelope so that they can be further processed after they have been run. One of the problems with allowing multi charged envelopes is that some paradoxes are bound to appear. One such is: take a part of a mass spectrum, here the peaks have 0.25 spacing between them. This would mean that an analysis method would find those peaks fitting up to three times. First as 0.25(charge 4), then as 0.5(charge 2) and lastly as 1(charge 1). Whom of these are the correct one ? The problem can be solved by knowing a little about the ionization source. How likely is it that it gives charge 4 at a specific M/Z range, or charge two or one. To my understanding the protease used can also have

an effect here, since some matrixes(for MALDI) seems to ionize specific amino acids[14]. The step after sorting out ambiguities of envelopes would be to share the intensity among the peaks, but how much should each isotopic envelope get of the total. This would depend on an isotopic template, or general rules how isotopic envelopes looks like.

Another weakness of the two deisotoping are that they do not split long isotopic envelopes, this can really be problematic if a charge of one is taken into account for a multi charged spectra. Remember chapter 3.4.7 where the masses between two peptides is about 1.005 from apex to apex(for charge 1). Depending on how reliable the peak detection method really is the 1.005 between charged one peptides can become a nuisance. This can be partly rectified in less complex mass spectra by looking at the changes of the intensities in the envelope in question. If the intensity goes up then down while iterating over the peaks then the result is fine. On the other hand if the intensities goes up again then there is a real reason to believe the envelope actually contain two envelopes and not one. The problem for complex spectra is that the intensities can go up again just because you have two or more overlapping envelopes.

Some general rules could have been devised for the purpose of getting a more compact result. One rule is to consider the number of peaks in an envelope and use a threshold based on this to remove unlikely envelopes from the result. Many such rules can be used after the deisotoping methods themselves to ensure that the user get the right one for his experiment, or they can be built into the analysis method in the first place.

The good part for the two deisotoping methods is that finding all the possible envelopes in a mass spectrum is good enough for a less complex spectrum(at least if you look away from charge one in a multiple charged spectrum), while for complex spectrum another method should be added after this one, or used instead of this one.

As with the deisotoping the deconvolution has its problems when it comes to dividing the intensity among more than one charge state of different peptides. it is not to hard to imagine that a peak could by chance belong to both a charge 2 peptide and another charge 3 peptide. This can be easily rectified if the charge state of each peak is given. Usually this is done in an deisotoping method and the deconvolution should not need to consider this.

The version of Gentzel's method implemented in MAP has its weaknesses, and as often happens some of these can easily enough be rectified but was discovered to late. The fact that it can only divide an M/Z interval between three peaks is one(Fig. 5.5), although for most spectrum this is enough. Also it can have problem to find all the peaks in very dense M/Z intervals, more precisely in intervals where the distance between the apex of two peaks are less than pw(x)/2. When peaks are centroided they are considered to be close to symmetric, most peaks on the other hand is not. This fact can easily miss interpret the intensities of two or more overlapping peaks, since the overlapping peaks intensities will be divided equally among the peaks. This can partly be rectified by using another scheme for dividing the intensities. As a note to miss interpreting intensities the Gentzel method can also place the apex of the peak on the wrong M/Z value since the apex of the centroid is based on the intensity of the whole peak. This can partly be rectified by finding all the peaks first without centroiding(but setting the apex to be the highest intensity), check for overlapping peaks, divide intensities among the peaks using a good scheme for it and then centroid the peaks. Even this can give errors, but they will likely be real small.

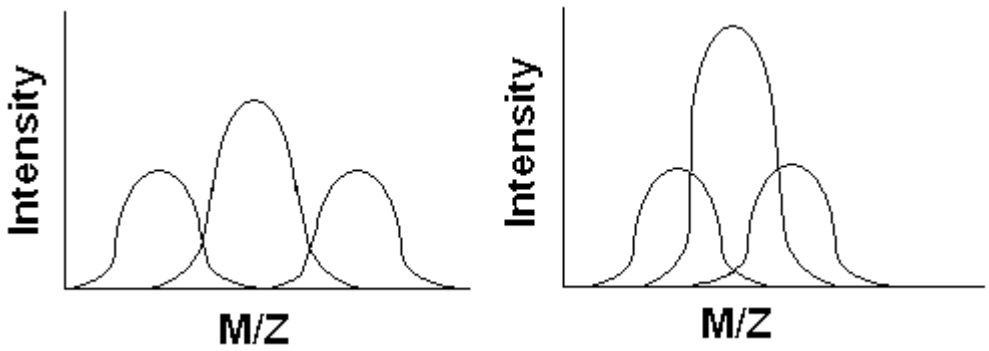


Figure 5.5: To the left the MAPs Gentzel analysis method is shown, and it is clearly that only two and two peaks overlap at once. To the right three overlapping peaks are shown.

The simple peak detection should be able to pick up asymmetric peaks, although both the M/Z values for the peak's apex and the intensity might be wrong. The M/Z value that is the real peak apex is not necessarily in the raw spectrum, and the intensity of asymmetric peaks do not usually follow a linear line from the valley to zero intensity. The peak apex can be 'guessed' at by centroiding, and the intensities between overlapping peaks can be found by using something else than linear regression. Maybe fourier could be used.

5.8 Find the Pairs in each Spectra

To find the marker pairs in a spectrum the distance between two M/Z values must equal the mass of the marker times the number of markers in the molecule, and the isotopic envelope's charge have to match for both the isotopic envelopes.

When both of these match there are a great possibility that the two M/Z values does not match by chance. MAP's analysis method for finding pairs for a known marker weight(marker pair) does both of these. The program lets the user set two crucial parameters, which are the maximum number of markers and the maximum number of charges MAP will look for. These parameters are used to systematically search the mass spectrum for pairs matching these. It starts by searching for pair with charge one and one marker, proceeds to search for pairs with one charge and two markers etc. until both the parameters reaches the maximum number. The way this is done is:

1. Find every possible pair with the correct distance in the mass spectrum, also called isopairs(figure 2), and put them in a pair list.
2. Search the list from 1. to see if any of the isopairs is in an isotopic envelope matching the current looked at charge in the analysis method. Only the lowest M/Z values of the pairs need to be checked out, since if two isopairs matches the envelope for these they will also match it for the highest M/Z in the isopair(see figure 2 to see the point).
3. Accept all the set of isopairs that also exceed the other parameters for the analysis method(see figure 4, options in the bottom part), where the number of isopairs needed is usually the most crucial one. All the accepted set of isopairs are called marker pairs.

In the result one M/Z value will show up for the peak, which is the first M/Z value for the marker pair. This is the same as monoisotoping. The whole analysis method is shown in algorithm 7.

Algorithm 7, Pair Finder aka Marked Molecule Finder

```
var
    MassSpectrum markerList          //The result is stored here
    MassSpectrum pairList            //stores a temporary list of pairs
    MassSpectrum globalList          //stores the global result
    iso-pair                         //stores a temporary isopair to be validated
    mw =  the weight of the marker
begin
    for each possible number of markers
        for each possible number of charges
            for each value(v) in original mass spectrum
                find the pairs satisfying the difference shown in the equation (mw*markers)/charge and
                    add them to the pairList
            end
            for each pair in the pairList
                iso-pair = find the pairs satisfying the isotopic envelope for the current pair, meaning
                           that the lowest/highest M/Z values in the pairs have a distance satisfying
                           the charges
                add the pairs exceeding the threshold and the number of iso-pairs to the markerList and
                    to the globalList
            end
        end
    end
    return markerList
end
```

There is also a global part to the analysis method, by having it save all the marker pairs into the same data structure. This is done in such a way that the same marker pair over different spectra is saved in the same place. If a marker pair is present in more than one spectra it is a clear indication that it should be investigated closer. The data structure itself is both used for single and global analysis, but the global part utilizes more of it. The data structure contains a list of pair objects, which contains both the M/Z and intensity for the two peaks in the pair, as well as the retention times these two were found at. The data structure also contains the charge and number of markers of the marker pair.

5.8.1 Discussion

To start of the discussion lets take a look at the opposite approach, which would be to deisotope first then use this to find the pairs. This might result in more exact intensity values if a good deisotoping method was used. On the other hand it would make the second part of the Pair Finder analysis method more complex. If one of the peaks in the isotopic envelope did not exceed the minimum threshold it would be lost to the second part of the analysis method. This one is easy to rectify by setting the threshold as low as possible, but this will result in much of the noise to be able to pair with real peaks. The worst that could happen would actually be if the deisotoping set the wrong peak to be the main peak, meaning a match would not occur in the second part of the analysis method. This can easily happen by chance in a spectrum that have some overlapping peaks, by having one of the overlapping peaks boosting the intensity of the other peak resulting in this being picked as the most intense peak instead of the right one. This would of course be at least partly rectified by a good deisotoping analysis method, but as mentioned in chapter 5.7.4 these can be real hard to make.

The next in line is the intensities of the isopairs. The Pair Finder itself does not check for overlapping isotopic envelopes when it matches up pairs. This means the intensity might be wrong. In the end the right pair should be more worth than an intensity that is not 100% correct. A solution to this problem could be to check the mass spectrum around the chosen pairs after each mass spectrum and correct these appropriately.

One thing that would really make an analysis method like this shine was if there existed a good way to score the pairs found before the MS/MS was run. Such a scheme could take into account how many isopairs each marker pair had, how many spectra in a row contained the same marker pair and some probability that the marker pair was a real peptide. These could then be combined to a score or even a probability telling what the chance would be for this one to be a true marker pair. As with all proteomics these scores could only be looked upon as guidelines, in the end the MS/MS would be the decider.

6 Results with Discussion

In this chapter the analysis methods in the program called MassAnalyzer is tested, and the results from these are shown followed by a discussion. The test data is taken from one of the three .raw files given to us early in the project. Where each of these files contains about 4300 mass spectrum where about 2300 of these are from MS experiments and the rest is from MS/MS experiments. The file chosen is the GDS_300707_THP1 -100_07.raw (Raw_007). As a solution/control, where this is possible, mass spectrum graphs taken from MassLynx is used. Regrettably it is difficult to find good solution/control mass spectra on the world wide web.

The program OpenOffice was used to draw the mass spectrum graphs for the MassAnalyzer program, since this only produces peak lists.

6.1 Peak Detection

The MassAnalyzer program contains three analysis methods for peak detection(PD). These are described in chapter 8.3.3 and goes by the name 'SimplePeakDetection'(SPD), 'Gentzel Peak detection'(GPD) and 'AutoGentzelPeakDetection'(AGPD).

6.1.1 Noisy Mass Spectrum

From Raw_007 the mass spectrum at retention time 10.036 is analysed first, mostly because it is a noisy mass spectrum with a varying baseline. Figure 6.1 shows the raw mass spectrum graph from MassLynx. The raw mass spectrum itself contains 50000+ peaks, so the M/Z values for the 'peaks' here should not be confused with real peaks but rather be looked upon as potential peaks.

For the next results the inbuilt peak detection in MassLynx is used with two settings, low and high, and are shown in figure 6.2 and 6.3 respectively. The difference between these settings is that the high setting contains methods for background subtraction etc. The low setting is to show of all the PD methods on equal footing.

What can be seen in these three figures and in every mass spectrum from MassLynx that is presented is that the y-axis is in percentages.

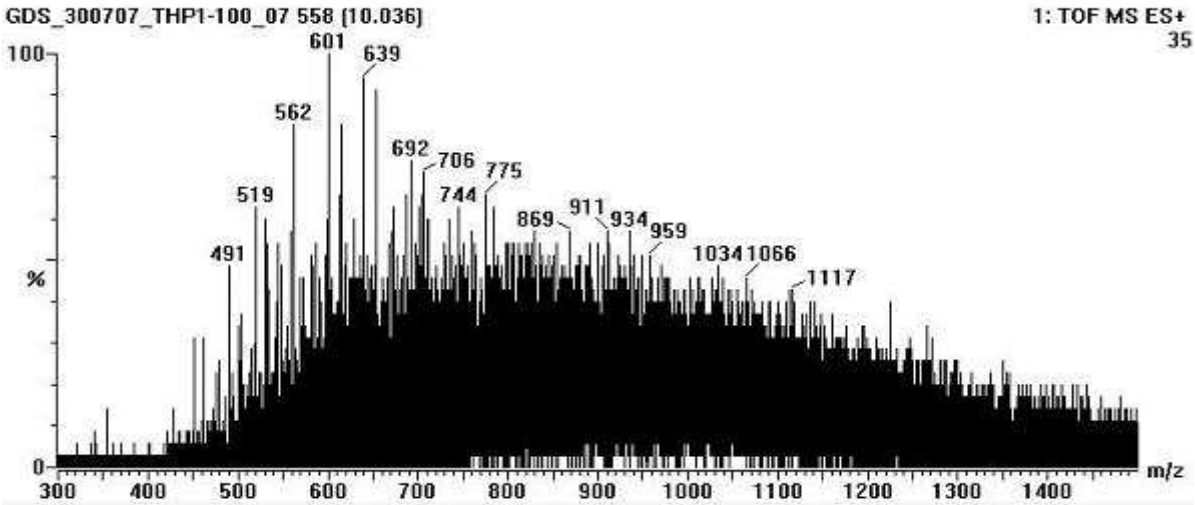


Figure 6.1: Raw mass spectrum graph from retention time 10.036. As can be seen this mass spectrum contains a lot of noise and varying baseline.

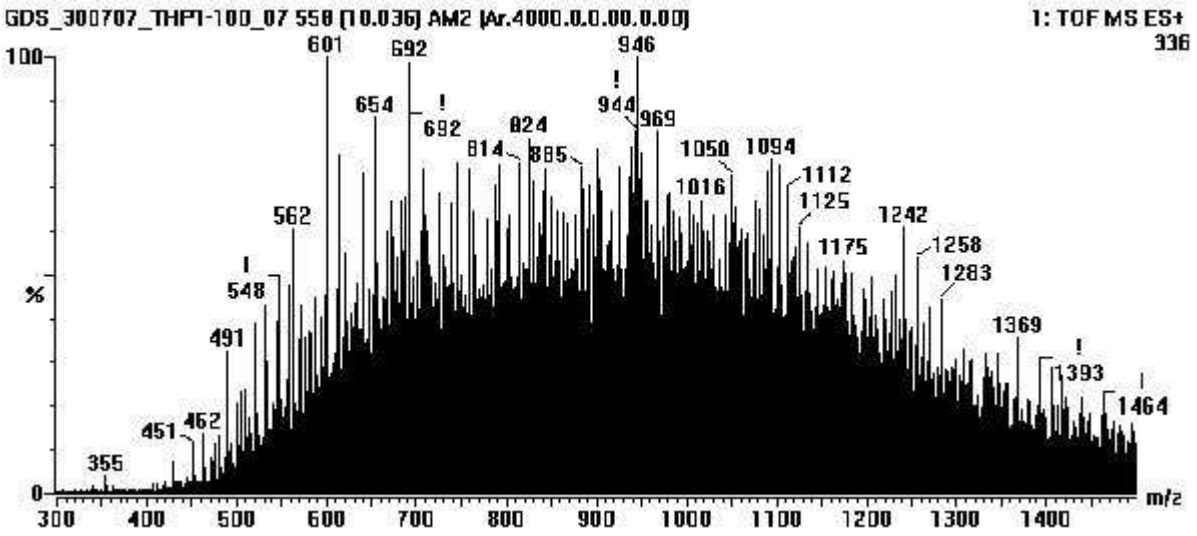


Figure 6.2: MassLynx's peak detection on low settings on the mass spectrum at retention time 10.036.

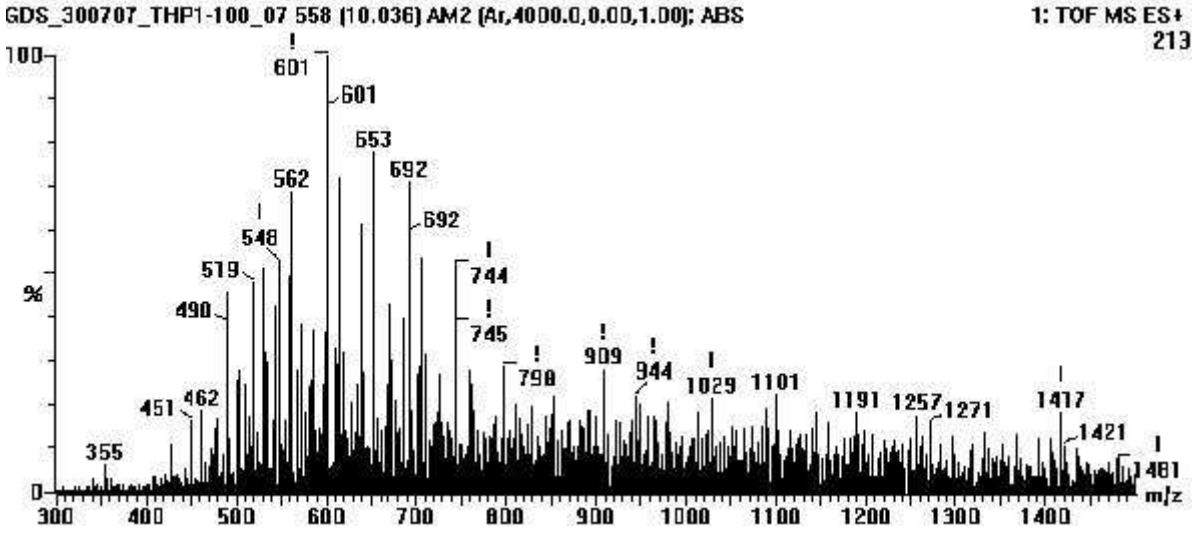


Figure 6.3: MassLynx's peak detection on high settings on the mass spectrum at retention time 10.036.

The first peak detection method out is the AGPD. Figure 6.4 shows a mass spectrum graph from this one, note that the scale on the y-axis is not in percentages but in intensities. The threshold used to produce this graph is set to 10% of the most intense peak, in the AGPD.

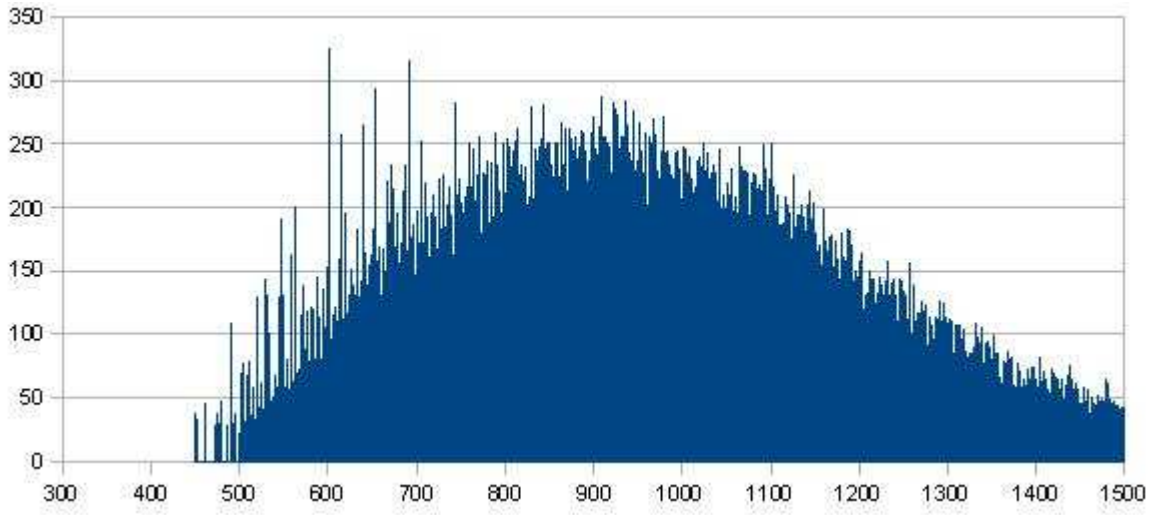


Figure 6.4: Mass spectrum graph from 'AutoGentzelPeakDetection' analysis method. Here the threshold is 10% of the most intense peak.

The next peak detection method is GPD. The threshold is harder to set for this one. A threshold of 200 in intensity is used to produce the graph, since this is a rather low on the large scale. Figure 6.5 shows the mass spectrum graph.

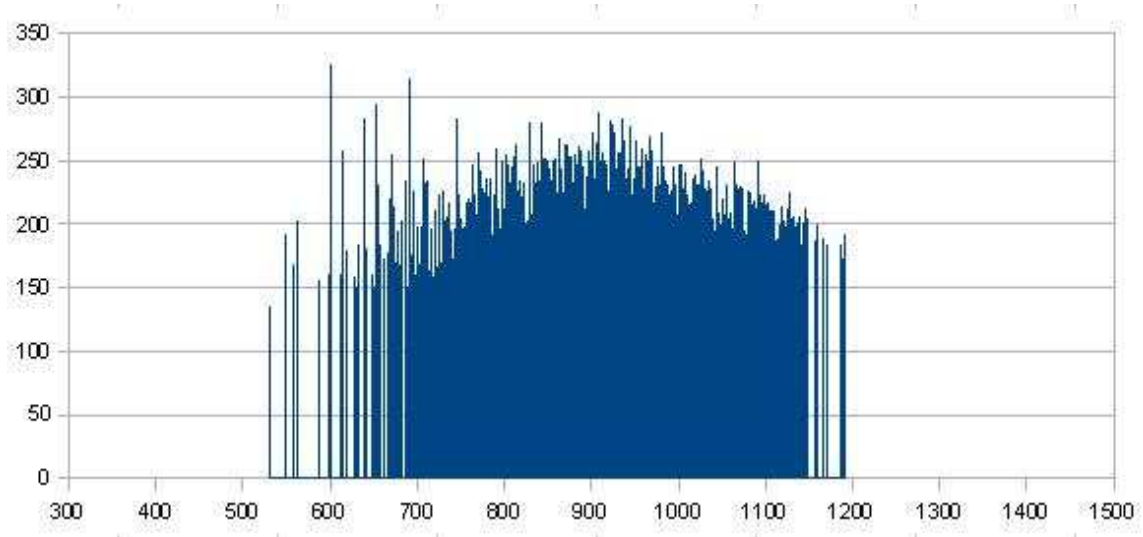


Figure 6.5: Mass spectrum graph from 'GentzelPeakDetection' analysis method. Here the threshold is set to 200 in intensity.

The last peak detection for this mass spectrum is SPD and the settings was set to a threshold of 200 and a sliding window of 0.35M/Z. The mass spectrum graph is shown in figure 6.6.

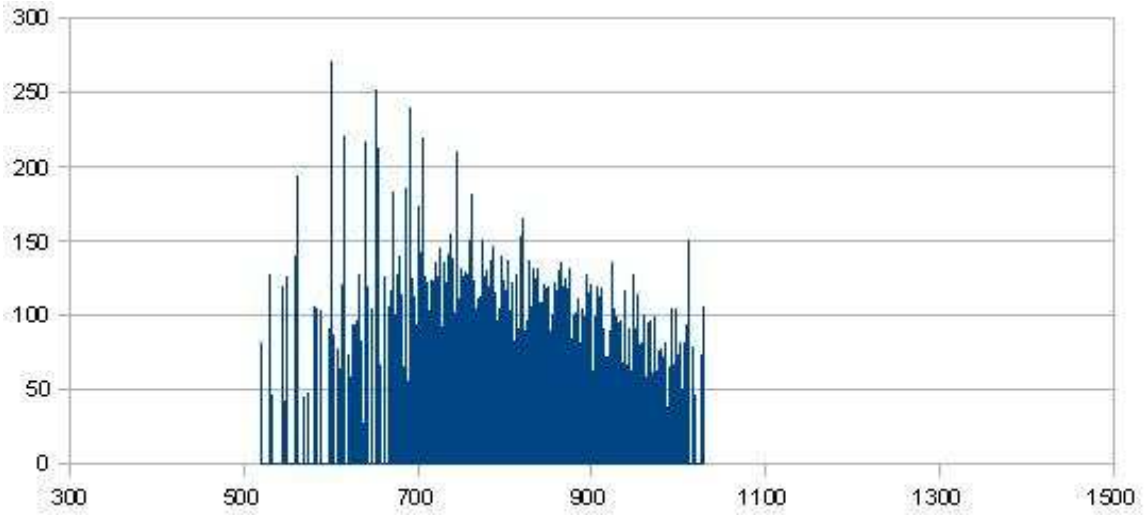


Figure 6.6: Mass spectrum graph from 'SimplePeakDetection' analysis method. Here the threshold is set to 200 in intensity and the sliding window to 0.35 M/Z which is about the same size as for 'GentzelPeakDetection' when it looks at peaks around 750 in M/Z

6.1.1.1 Discussion

As can be seen both AGPD and GPD have the same general shape as the MassLynx peak detection on low setting. By increasing the setting of the MassLynx PD the mass spectrum becomes a lot more readable. All the analysis methods in the MassAnalyzer program have no noise reduction or baseline correction at this moment. These were analysis methods meant to be implemented at a point and for the user to decide what he/she would use at run-time. This means that the results on this mass spectrum was not unsuspecting at all.

Comparing the mass spectrum graphs from the MassAnalyzer program one can see that both GPD and AGPD is quite similar(Fig. 6.4 and 6.5). The only big difference is that it seems GPD gets rid of more noise. What can be said about this is that AGPD's strength is also its weakness. This is that

AGPD normalize the peak threshold based on a percentage of the most intense M/Z value in the raw mass spectrum AGPD is currently analyzing. This percentage is set in the analysis methods parameters. This leads to the intensity of the threshold varying from mass spectrum to mass spectrum, which was intended as a form of global normalization for analysis on more than one mass spectra. The weakness is that a noisy spectrum like the one the test was run on will result in a low intensity threshold which again results in a lot of noise being looked upon as real peaks. The solution to this is quite simple, add a minimum threshold for the current threshold. This threshold can be implemented in the same way as for GPD, which is that all peaks under this threshold will be considered noise. This will of course have the weakness that for a mass spectrum with little noise but low intensities the real peaks wont be found. As a defence to this method, it is difficult to know if such a mass spectrum is not noise.

The overall result of the SPD(Fig. 6.6) is that the peaks are weaker in intensities as well as there are peaks missing in some areas. Here this is good since most of it is noise, but overall it is bad since that is usually not the case.

6.1.2 Strong Peaks Mass Spectrum

From Raw_007 the mass spectrum at retention time 30.077 is analysed next, mostly because it is a mass spectrum containing many strong peaks. Figure 6.7 shows the (raw data)mass spectrum graph from MassLynx. After using peak detection from MassLynx on a low setting is shown in figure 6.8, which is done to show of the results at equal footing. High setting of the same peak detection method from MassLynx is shown in figure 6.9.

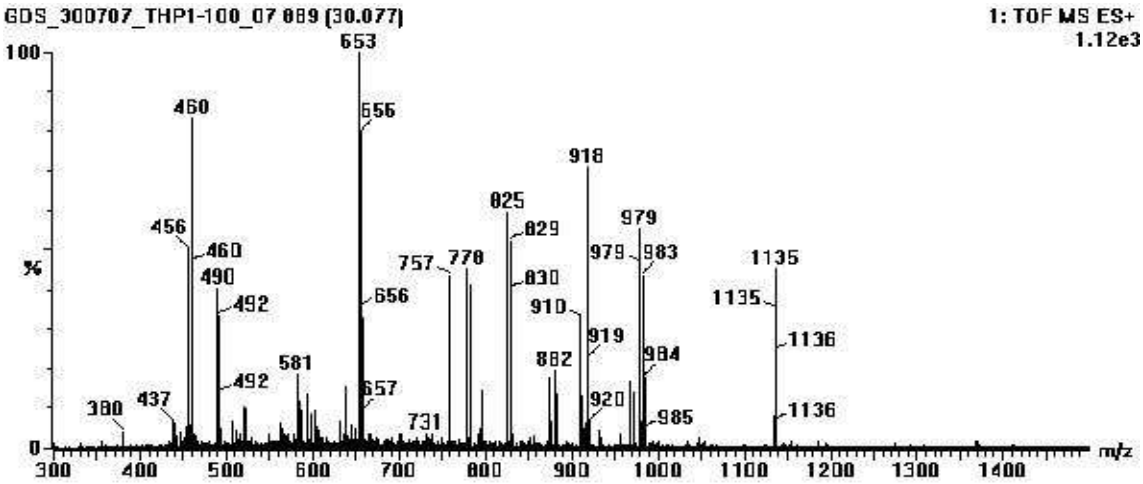


Figure 6.8: Raw mass spectrum graph from retention time 30.077. As can be seen this mass spectrum contains a lot of noise and varying baseline.

Figure 6.9: MassLynx's peak detection on low settings on the mass spectrum at retention time 30.077.

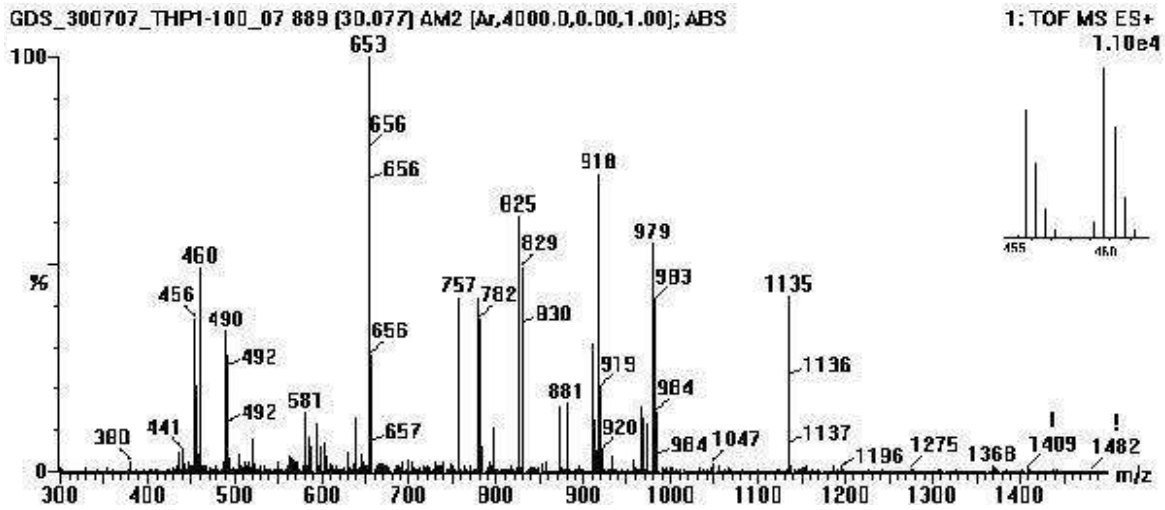


Figure 6.10: MassLynx's peak detection on high settings on the mass spectrum at retention time 30.077.

First out of the analysis methods from the MassAnalyzer program is the ADGP(Fig. 6.11). This was run with a threshold at 10% of the most intense peak. The next one is GDP(Fig. 6.12), and it was run at threshold of 200 in intensity. The last one is SPD(Fig. 6.13), and the parameters were set to 200 in threshold and 0.35 M/Z sliding window.

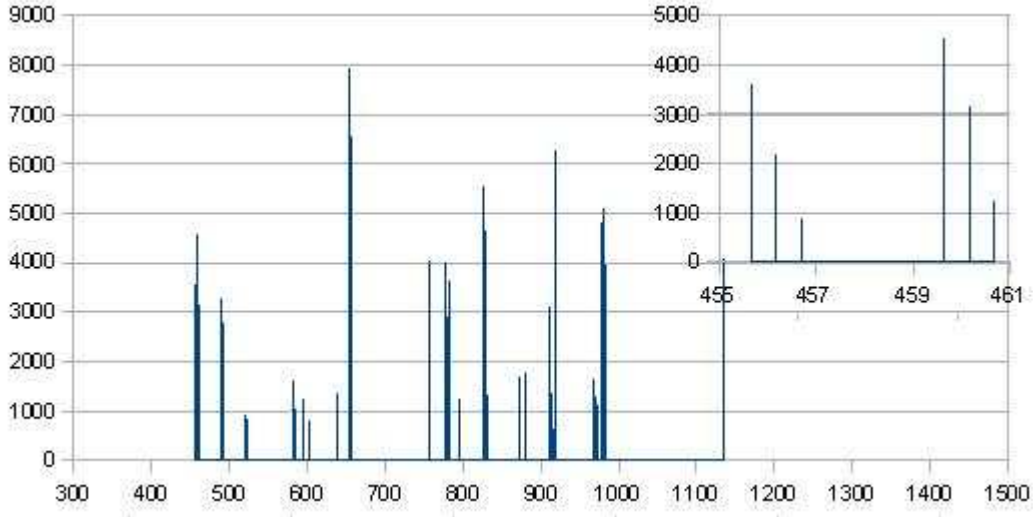


Figure 6.11: Mass spectrum graph from 'AutoGentzelPeakDetection' analysis method. Here the threshold is 10% of the most intense peak.

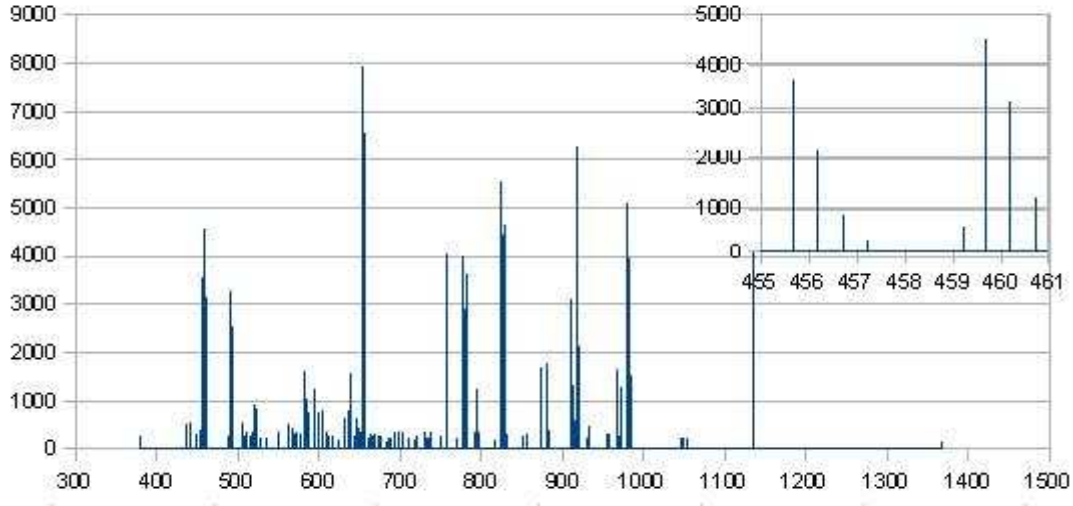


Figure 6.12: Mass spectrum graph from 'GentzelPeakDetection' analysis method. Here the threshold is set to 200 in intensity.

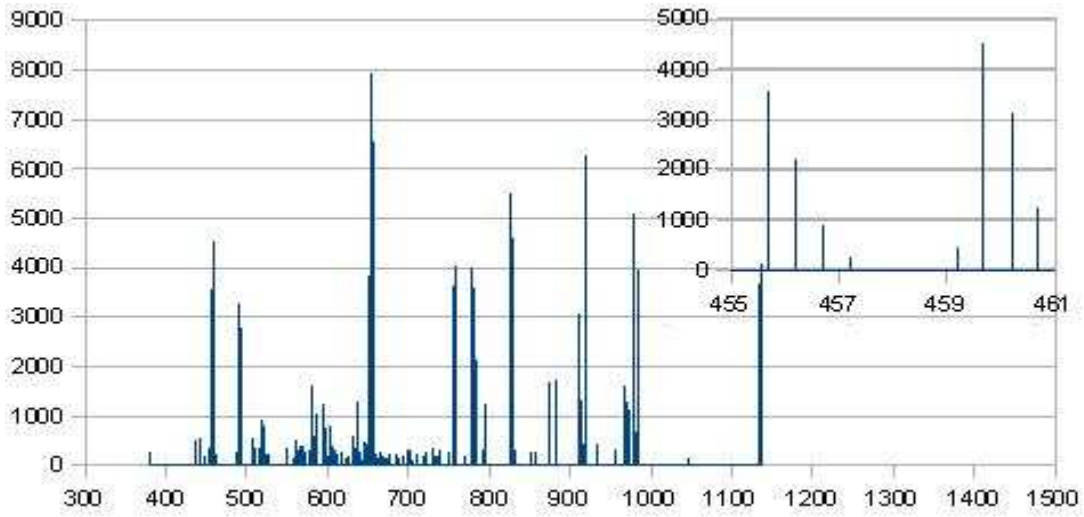


Figure 6.13:Mass spectrum graph from 'SimplePeakDetection' analysis method. Here the threshold is set to 200 in intensity and the sliding window to 0.35 M/Z which is about the same size as for 'GentzelPeakDetection' when it looks at peaks around 750 in M/Z

6.1.2.1 Discussion

Here all the peak detection methods in the MassAnalyzer program do good. All the important peaks has been found and they are all strong in all of the PD methods. SPD(Fig. 6.13) looks like it might miss or misplace some peaks since the graph looks a little different in some areas(strong peaks at around 970 and 650 M/Z are examples). On top of this SPD seems to produce a bit noise in its spectrum, but it is hard to set a threshold based on an intensity value for a larger analysis. A higher setting would have removed much of this noise, but on a global scale a higher setting might easily remove real peaks. GPD(Fig. 6.12) also have the same kind of noise in its graph as SPD, while AGPD(figure 11) seems to have been able to ignore the noise because of the strong peaks, even if the threshold for this one is relative low.

The zoomed in areas shows that all of the methods manages to have a close to similar peak distribution in the areas 455 to 461 which is one of the sets of isopairs from the control set(appendix F). The difference is in AGPD this time, since it has removed some of the weak peaks, that might or might not have been noise. Most likely this was part of the isotopic envelope since it seems to be consistent with the rest of the peaks in the envelope.

6.2 Deisotoping

There are two deisotoping methods in the MassAnalyzer program, 'MonoDeisotoping'(MD) and 'IntensityDeisotoping'(ID), both of these will be used on the raw mass spectrum shown in figure 6.8. The peak detection method used is AGPD(Fig. 6.11). Table 6.1 shows the data after MD has been used to the left and after ID has been used on the right.

Monoisotoping						Intensityisotoping					
M/Z	Intensity	Charge	M/Z	Intensity	Charge	M/Z	Intensity	Charge	M/Z	Intensity	Charge
455,69	4425	1	455,69	6606	2	455,69	4425	1	455,69	6606	2
459,68	5765	1	459,68	8892	2	459,68	5765	1	459,68	8892	2
489,97	5120	2	489,97	8278	4	489,97	5120	2	489,97	8278	4
491,98	3977,5	2	491,98	6517	4	491,98	3977,5	2	491,98	6517	4
637,86	2664	2				637,86	2664	2			
652,95	9312	1	**652,95**	**21805**	**3**	652,95	9312	1	**653,29**	**21805**	**3**
655,63	15546	3				655,63	15546	3			
756,98	4527,5	1	**756,98**	**10904**	**3**	756,98	4527,5	1	**757,32**	**10904**	**3**
778,37	5045	1	778,37	7920	2	778,37	5045	1	778,37	7920	2
782,38	5763	2				782,38	5763	2			
825,37	7499	1	825,37	11944	2	825,37	7499	1	825,37	11944	2
829,38	5927	1	829,38	9361	2	829,38	5927	1	829,38	9361	2
910,4	5669	1				910,4	5669	1			
918,4	8364	1				918,4	8364	1			
967,39	3000	2				967,39	3000	2			
971,4	2381	2				971,4	2381	2			
978,95	7150	1	**978,95**	**12222**	**2**	978,95	7150	1	**979,45**	**12222**	**2**
982,96	5467	1	982,96	9273	2	982,96	9273	2	982,96	5467	1
1134,98	5973	1	**1134,98**	**10043**	**2**	1134,98	5973	1	**1135,48**	**10043**	**2**

Table 6.1: This table shows all the isotopic envelopes found using monoisotoping(MD) and intensityisotoping(ID). The bold (M/Z, intensity, charge) on the same line on both sides are where these two deisotoping methods have chosen differently. Each of the lines in the envelope have the same M/Z except for the bold one on the right side of the table.

6.2.1.1 Discussion

Table 1 clearly shows that the two analysis methods MD and ID do the choices differently, and that they find all the isotopic envelopes and do not make any choice about whom of those are the right one. Table 1 also shows that the differences between MD and ID is for the peaks 653,29 and 757,32 and 979,45 and 1135,48. Figure 6.14 shows these peaks taken from the mass spectrum in MassLynx, but just as an illustration since the peaks have more different intensities in the real mass spectrum. All of these shows that the differences is really there.

Table 1 illustrates another point to, that the envelopes of charge 2 and 4 can be mistaken for the other, as well as those of charge 1 and 2. All the double finds show this(on the same line in table 6.1). The mass spectrum shows that peaks can vary in isotopic envelopes even within the same mass spectrum. Even for peaks that might be a marker pair, like 652,95 and 655,63 with the charge 3.

A researcher can more easily go into the mass spectrum and make judgement calls of what charge is right compared to what a computer is able to do. Some good rules, isotopic templates etc. can make a good start for the computer to do the right choices also, but if these are a little of so will the result be.

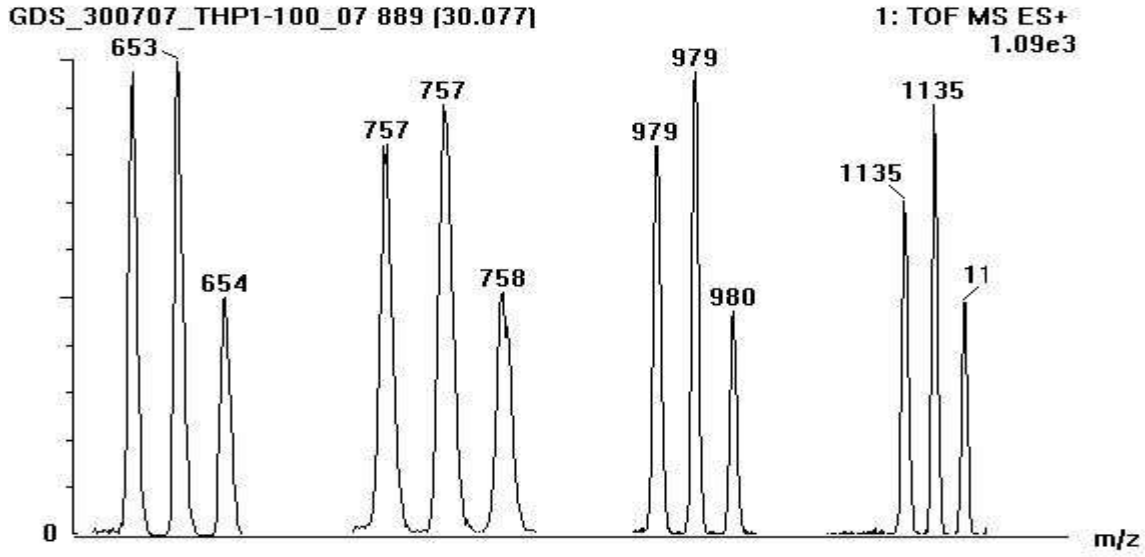


Figure 6.14: The peaks where the ID and MD does differently as taken from MassLynx, but modified to be seen in the same window. Also the intensities are not as a like between these peaks as it is shown here.

6.3 Deconvolution

The deconvolution method function best if a deisotoping analysis method is used before it. This way it will have the extra information of what charge a peak has.

6.3.1.1 Discussion

None of the deisotoping analysis methods are fair to use for this analysis method, since they both give multiple 'MZICharge' with the same M/Z value. The modified binary search of the deconvolution method would have to happen on the right index, of equal M/Z values, by chance. The mass spectrum in figure 8 has one triplet with the M/Z 978,95 and 655,63 and 489,97. These can all be found with the deconvolution method with some luck involved, and adds up to M/Z 978,95 and intensity 42305,00. Looking at table 1 these are the correct values. There are also other M/Z values that have been deconvoluted, but the above mentioned one was the one in the control set. Switching the place between deconvolution and monoisotoping in the template yields a little different results, here the M/Z is the same but the intensity is only 20500. When browsing through the long list it seems like with this order it does not manage to find the charge 3 peak at M/Z 655,63. No more will be said about deconvolution, since there are currently no good deisotoping method in the MassAnalyzer program that can be put in front of it. The solution for this analysis method could be to change the search method to something more accurate where there are more than one M/Z with the same value, and check them all out.

6.4 Pair Finder aka Marked Molecule Finder

The Pair Finder method will be tried against some of the more interesting peaks in the control set. The control set itself is taken from a SILAC experiment where the three most intense peaks were chosen for MS/MS. The sequences of these peaks were checked for charge, number of markers and so on, and a centroided peak was created by the MSQuant program. This means the control set will not contain the full solution, but rather a small part of it.

The mass spectrum chosen for the test was from retention time 10.036, 22.562, 30.077, 36.220 and 44.702 from the Raw_007. The first one was noisy mass spectrum. The second and third one contains one unknown isotope in the control set (appendix F), be it heavy or light. The Fourth one contains both isotopes in the solution. There are also results from two intervals of mass spectra. The first one is from retention time 51.514 to 52.456, while the second one is from retention time 29.624 to 30.567. The last result will be a global one, seeing how many of the marker pairs in the control set can be found with Pair Finder method.

In the tables for the following test mass spectra the headings TLI means total light isotope intensity, 2/1 means heavy- divided by light-isotope intensity, Pair No. means how many isopairs there are in the marker pair and MS-Count means how many mass spectra the marker pair is in. The 2/1 can vary much between the result and the control set, since it is very hard to analyse the exact same mass spectrum as in the control set(because the retention times does not match well with the ones in the raw file).

The parameters will be the same through all of the analysis, except where noted otherwise, and are shown in figure 6.15.

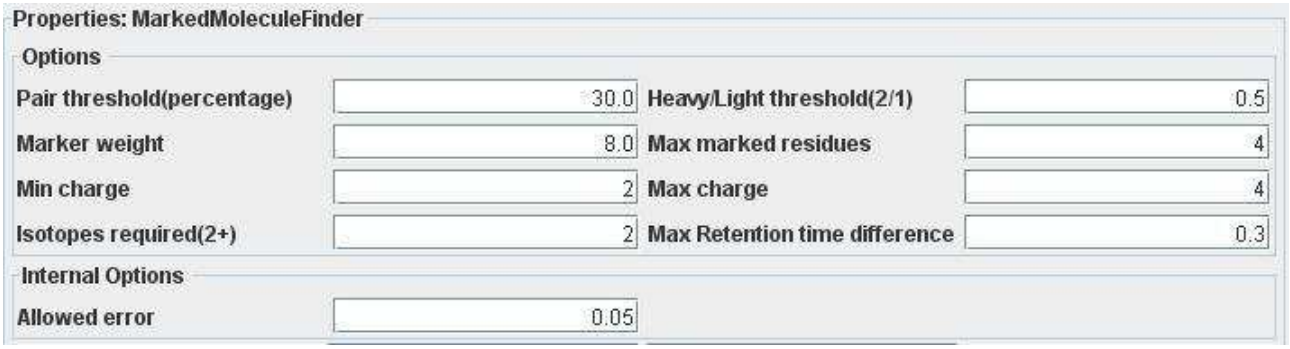


Figure 6.15: The parameters used in all the results with the Marked Molecule Finder

6.4.1 Noisy Mass Spectra

The mass spectrum at retention time 10.036(Fig. 6.1) is presented in table 6.2, but only as a short version(none of the marker pair is listed) since 675 marker pairs were found.

Retention Time10.036

N/Z	1	2	3	4	
1	0	50	3	73	
2	0	48	41	78	Total peaks in parent spectrum: 2303
3	0	64	60	74	Total of parent peaks in possible pairs: 2268
4	0	48	32	73	Total marker pairs: 675

Table 6.2: Shows the results of the Pair Finder used on the mass spectrum at retention time 10.036 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For two markers and three charge there have been found 41 marker pairs.

6.4.2 Heavy Isotope Missing in the Control Set

The next mass spectrum is from retention time 22.562. The result is presented in table 6.3.

Retention Time22.562

N/Z	1	2	3	4	
1	0	2	0	0	
2	0	0	0	0	Total of peaks in parent spectrum: 35
3	0	0	0	0	Total of parent peaks in possible pairs: 22
4	0	0	0	0	Total marker pairs: 2

MZLow-MZHigh	Charge	Markers	TLI	2/1
524,75-528,76	2	1	9516	0,8
581,27-585,28	2	1	22283	0,82

Control Set M/Z	Charge	Markers	Sequence	
524,75	2	1	NGQDLGVAFK	0,91

Table 6.3: Shows the results of the Pair Finder used on the mass spectrum at the retention time 22.562 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For one marker and two charge there have been found 2 marker pairs.

6.4.3 Four Peaks from the Same Peptide in the Control Set

The mass spectrum at rertention time 30.077(Fig. 6.8) is the next one to be tried with the Pair Finder. Table 6.4 shows the result.

Retention Time30.077

N/Z	1	2	3	4	
1	0	4	1	1	
2	0	0	0	0	Total of peaks in parent spectrum: 65
3	0	0	0	0	Total of parent peaks in possible pairs: 53
4	0	0	0	0	Total marker pairs: 6

MZLow-MZHigh	Charge	Markers	TLI	2/1
455,69-459,68	2	1	6606	1,35
778,37-782,38	2	1	6877	0,84
825,37-829,38	2	1	11944	0,78
979,45-983,47	2	1	12222	0,76
653,29-655,97	3	1	20253,5	0,77
489,97-491,98	4	1	8278	0,79

Control Set M/Z	Charge	Markers	Sequence	2/1
978,95	2	1	HGYIGEFEIIDDHRAGK	0,83
655,63	3	1	HGYIGEFEIIDDHRAGK	0,83
652,97	3	1	HGYIGEFEIIDDHRAGK	0,83
489,97	4	1	HGYIGEFEIIDDHRAGK	0,83

Table 6.4: Shows the results of the Pair Finder used on the mass spectrum at the retention time 30.077 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For one marker and two charge there have been found 4 marker pairs.

6.4.4 Light Isotope Missing in the Control Set

Looking for one unknown isotope in the mass spectrum at retention time 36.220. The Pair Finder result can be found in table 6.5.

Retention Time36.220

N/Z	1	2	3	4	
1	0	1	0	0	
2	0	0	0	0	Total of peaks in parent spectrum: 49
3	0	0	0	0	Total of parent peaks in possible pairs: 14
4	0	0	0	0	Total marker pairs: 1

MZLow-MZHigh	Charge	Markers	TLI	2/1
913,47-917,48	2	1	6082	0,81

Control Set M/Z	Charge	Markers	Sequence	2/1
916,98	2	1	VGINYQPPTVVPGGDLAK	0,92

Table 6.5: Shows the results of the Pair Finder used on the mass spectrum at the retention time 36.220 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For one marker and two charge there have been found 1 marker pairs.

6.4.5 Both Isotopes in Control Set

Last single mass spectrum result for the Pair Finder is from retention time 44.702 and is shown in table 6.6.

Retention Time44.702

N/Z	1	2	3	4	
1	0	0	1	0	
2	0	0	0	0	Total of peaks in parent spectrum: 49
3	0	0	0	0	Total of parent peaks in possible pairs: 31
4	0	0	0	0	Total marker pairs: 1

MZLow-MZHigh	Charge	Markers	TLI	2/1
868,39-871,07	3	1	20594	0,75

Control Set M/Z	Charge	Markers	Sequence	2/1
870,74	3	1	DLYANTVLSGGTTMYPGIADRMQK	0,86
868,06	3	1	DLYANTVLSGGTTMYPGIADRMQK	0,86

Table 6.6: Shows the results of the Pair Finder used on the mass spectrum at the retention time 44.702 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For one marker and three charge there have been found 1 marker pairs.

6.4.6 Interval of Mass Spectra with No Peaks in the Control Set

Table 6.7 are an interval from retention time 51.514 to 52.456. This interval was chosen since the control set had no pairs in this area.

Retention Time51.514-52.456

N/Z	1	2	3	4	
1	0	0	0	0	
2	0	0	0	0	Number of MZ spectra: 9
3	0	0	0	0	Total of parent peaks in possible pairs: 131
4	0	0	0	0	Total marker pairs: 0

Table 6.7: Shows the results of the Pair Finder used on the mass spectra at the retention times 51.514 to 52.456 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For one marker and two charge there have been found 0 marker pairs.

6.4.7 Interval of Mass Spectra with Four Peaks in the Control Set

The interval from retention time 29.624 to 30.567 contains the mass spectrum at retention time 30.077(Fig. 6.8) and was chosen to see how this marker pair behaves over more than one mass spectra. The results are shown in table 6.8.

Retention Time29.624-30-567

N/Z	1	2	3	4	
1	0	15	10	4	
2	0	0	0	0	Number of MZ spectra: 8
3	0	0	0	0	Total of parent peaks in possible pairs: 370
4	0	0	0	0	Total marker pairs: 8

MZLow-MZHigh	Charge	Markers	TLI	2/1	Ret.Time	Pair.No	MS-Count
685,33-689,34	2	1	18798	0,71	29,62	3	1
455,68-459,67	2	1	26391	1,42	29,84	3	4
778,37-782,37	2	1	13908	0,84	30,08	2	2
825,35-829,37	2	1	67796	0,79	30,08	3	4
979,45-983,47	2	1	43596	0,72	30,08	3	4
653,29-655,97	3	1	131566	0,69	29,84	4	8
912,40-915,07	3	1	27255	0,65	30,31	4	2
490,22-492,23	4	1	33099	0,78	30,08	3	4

Table 6.8: Shows the results of the Pair Finder used on the mass spectra at the retention times 29.624 to 30.567 from Raw_007. The table to the left shows the number of marker pairs found in the different categories of charge and number of markers. Example: For one marker and two charge there have been found 15 marker pairs. The control set for these are the same as for table 4.

6.4.8 Global Results

A simplified version of the global results are shown in table 6.9. Here the MZ/MZ mass spectra is also included(this result is from an earlier version of the MassAnalyzer program).

Global

N/Z	1	2	3	4	
1	10669	2765	2498	3169	
2	12794	2115	1758	2868	
3	10389	1856	2060	2683	Number of MZ spectra: 4328

isopair =3+	TP	FP	FN	TotList	TotSpec
Pair	38	5728	7	45	5792
Single	26	0	15	41	

isopair = 2	TP	FP	FN	TotList	TotSpec
Pair	7	18313	38	45	18335
Single	15	0	26	41	

Table 6.9: Global result for the Pair Finder run on a little older version of the Pair Finder analysis method. The allowed error here is set to 0.06 instead of 0.05 some pairs from the solution did not want to go with lower. The threshold for the peaks in AGPD was set to 5%. A note to this table is that all TP, FP and FN is compared to the used control set.

6.4.9 Discussion

The Pair Finder seems to do real well on finding pairs, even some that are not in the control set. Although these are not necessarily real pairs. To be looked upon as a real pair both the strongest peaks in an isopair, from the marker pair, needs to be investigated by MS/MS.

Starting at table 6.2 the Pair Finder like the PDs is also confused by noise, but only because the AGPD produced a peak list with to much noise. This only shows that the peak detection before the Pair Finder is very important.

Table 6.4, 6.5 and 6.6 shows that the isopairs in the marker pair is of a peak compared to the solution. There can be more than one reason for this. They all share one common ground which is:

- The peaks in the isotopic envelope varies in strength over the retention times the envelope appears in. They usually start of weak and increases before decreasing again.

This means that early mass spectrum of an isotopic envelope might miss out on low intensity peaks, since earlier used analysis methods might have removed it as noise. For marker pair this is doubly true since there are two isotopic envelopes that needs to be considered. If one of them is missing a peak an isopair wont be formed and that will make the Pair Finder miss it. Usually the low intensity peaks is in front or back of the isotopic envelopes so only fluctuation that shows this should be considered as missed peaks.

In table 6.4, 6.5 and 6.6 all the marker pairs miss one in the start, and compared to the solution the rest of the isotopic envelopes fits if the missed isopair was inserted into the marker pair.

The control set for table 6.3 and 6.5 shows only one of the light and heavy isotopes in the isopair. The Pair Finder finds the missing isotopes and makes a marker pair out of them.

Not surprisingly is that where both the isotopes of the isopair shows in the control set it also was found by the Pair Finder(Table 6.6).

In table 6.4 the control set have 4 peaks to be found in 3 pairs, all were found. These were also found in table 6.8 where the interval containing the single mass spectrum used to produce table 6.4 were analysed.

There is no real surprise that some more marker pairs were found by the Pair Finder, since as mentioned before the usual method for MS/MS(which the control set is based upon) is to take out some of the strongest peaks and analyse them. The mass spectrum at 26.562 contained one extra marker pair(table 3). The mass spectrum at 30.077 contained three extra marker pairs(Table 6.4). These pairs also appears in the interval(table 7) containing this particular mass spectrum. This interval also shows that two of these (455.68, 459.67) and (825.35, 829.37) have real potential to be real ones since they appear in 4 of the 8 mass spectra in the interval. The last pair found in 30.077 only appeared in that mass spectrum. The interval have more to show of though, 2 more pairs on top of the before mentioned was also found. One of these were at the start of the interval, but only in one mass spectrum. This one could be investigated closer by taking a look at the mass spectra before the chosen interval and see if it appears in more than only one. The second one of these was in the middle of the interval and only appeared in two mass spectra. This might be worth investigating but not as a top priority, unless this marker pair has a specific M/Z value or have an intriguing 2/1.

The second chosen interval showed the same as the control set, which is nothing found(Table 6.7)

One thing to consider when looking at table 9 is that this is only against the control set, which means that many of the FP in this case will actually be possible marker pairs that might need further investigation with MS/MS. This the result shows that all the pairs(Table 6.9) in the control set can be found by adding the the numbers from the isopairs containing two and three+. Also the numbers under the false negative in this table shows that they compare to other ones true positive, which means the 7 false negative for three+ isopairs shows up as the true positives for two isopairs.

What can be said about the potential marker pairs from this run is that they are based upon an earlier incarnation of the Pair Finder analysis method, which means it miss out on some of the parameters to sort out unwanted marker pairs(like to low charge etc). Upgrading this in the test method might reduce the numbers of found false positives, but it might also increase it since the new version of the Pair Finder also have the extra criteria of retention time difference which the old one did not have. What is clear is that the PairFinder will find all the marker pairs from the control set as well as some extra. Both situation of course depends on the parameters set in the Pair Finder and the analysis methods used before it.

The reason for using this early result is that they show all the pairs will be found, even with the new incarnation of the Pair Finder. The reason for not using the new Pair Finder is that the Result Structure has not been upgraded and the new version of the Pair Finder does not integrate it.

Back to the huge number of potential marker pairs(FP). The experiment this collection of mass spectra is taken from is a SILAC experiment on two different cell cultures. As is known is that a cell contains a lot of different active proteins at once and digesting these to peptides with about size 10 results in thousands of fragments. This means that these high numbers of potential marker pairs are not just false positives(still control set) but are as the word says potential marker pairs. What can be said, even though many are potential marker pairs, is that the Pair Finder method is very sensitive to noise. If the mass spectra analysed are noisy and this noise is not removed there might happen on some occasions that some peaks are paired by chance, but one likes to think that pairing up peaks on the marker distance first and then using this to further pair them up on the isotopic envelopes will remove most of this. Also as seen in with the two deisotoping analysis methods, that two isotopic envelopes can overlap completely if they have charge 1, 2, 4, 8 and so on. An easy calculation shows this, 1/charge is the distance between two isotopes in an envelope. If the envelope is big enough and the peaks are spread with distances of 0.25 an envelope for charge 4 will be found as well as charge 2 and 1. The Pair Finder also suffer from this in its current condition, since it does not try to deduce what of the isotopes are the right one. The reason this is mentioned is that these are bound to have appeared among the potential marker pairs.

Deconvolution can help decrease the number of possible pairs for the Pair Finder by gathering the same peptide with different charge under one pair.

7 Java

Java is a free object oriented programing language created by Sun Microsystems. The difference between Java and other programing languages is that Java is a multi-platform language, through its virtual machine, with a lot of pre-made classes. The multi-platform part can be broken if classes are written especially for one platform or hardware. All the classes delivered by Sun Microsystem are multi-platform though.

7.1 Object Oriented Language

Object oriented programing uses classes to define their objects. These classes contains variables and references describing the object as well as methods that uses or have an effect on the variables. The program can then instantiate these classes directly as objects or references in other objects. Each class can instantiated more than once, so it is best to look upon a class as a template representing the goal of the object in question. An example of a class could be Animal. This class could describe features like feeding habits, number of legs etc. through variables, and the methods could for example calculate how many days an animal could live on a food source and such. An instance of this class could be an elephant, dog, mouse etc. If further specification of the animal is needed, like grouping them into distinct groups with their own behavior inheritance can be used. Inheritance is simply that the object(child) inherits the methods and variables from their mother class(also called super class). An example could be a Dog class inheriting from the Animal class. Now types like spaniel and collie could be instantiated by the Dog object. The good thing about inheritance is that you get polymorphism, which means that you can call on methods on the Dog object as if it was an Animal object. A good example of this would be to keep a list of Animal(s) with Dog objects in it. This is possible since the Dog object inherits from the Animal object. Then you could iterate through the list, for example sorting on number of legs, and use all the object in the list as Animal objects. That is what they are, even the dogs, and use this to ask for number of legs. Other uses for objects are as events, messages etc. The sky is the limit.

If one should compare object oriented programing to another language one could use C. In C you have a data structure called struct which simply is a gathering of variables and/or pointers. This data structure inherently lacks the methods and the ease of access changing objects have, but other than that they are quite similar.[52]

7.2 Java Virtual Machine

To make a multi-platform programing language Sun Microsystem decided to go for a virtual machine(JVM). This can be looked upon as a software machine, emulation, on top of the real machine. Before the code can be run on the JVM it has to be compiled. Instead of compiling it to a specific environment like windows, the compiler creates what Sun has called bytecode. This bytecode is then interpreted by the JVM which 'communicates' with the specific environment. This literally rules out the need of compiling the code for different environments, as long as the code does not use any libraries/packages that requires a specific environment. Figure 7.1 shows a rough sketch of the way from code to a running a program.

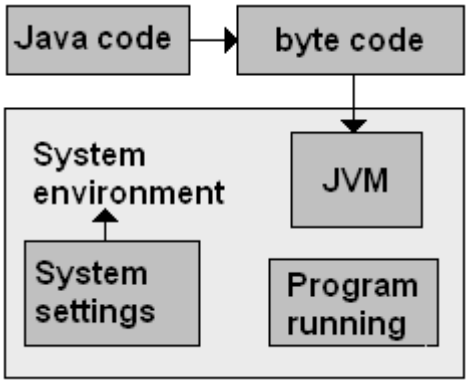


Figure 7.1: Shows a sketch over the way from Java code to running on a machine.

The JVM also does something called inlining, which basically is a form to 'correct' the code to make it more effective. An example would be a loop which goes for 2000 steps checking an array for its length every run with no code inside the loop that could alter the length of the array. Then the JVM would take out this length check and replace it with a variable before the loop, saving a lot of checkup time every time the loop is started. The JVM also does garbage collection for the programmer, unlike C where the programmer have to do it him-/herself. On top of this the JVM also allows for thread synchronization.[53]

7.3 Inheritance in Java

Inheritance is implemented in three ways in Java, through:

- abstract class, a class that can not be instansiated as an object but can be used as a reference. This class can contain half finished methods, which have to be created by its children.
- interface, a class that can not be instantiated, can not have variables except final static(variable is the same for each instansiation of the class and can not be changed) and methods only contains the head/signature. Interfaces can be looked upon as a contract the implementing object needs to fulfill.
- ordinary class.

An object can only inherit from a class or an abstract class, while it can inherit as many interfaces as it wants to.[52]

7.4 Data Structures in Java

Unlike many other languages Java comes with the most typical data structures already implemented. To mention a few; array, vector, list, graph, map, tree etc.

One structure that is often used for data that changes size is the arraylist. This structure is a list of objects backed up by an array, literally removing the one weakness a list has over an array. This is being able to get an element using an index directly instead of having to iterate over the list. Although it also gets one weakness from the array, and that is the resizing of an array when the list goes over the capacity, and moving of elements in an array when an element is not put in at the end of the list. These weaknesses is not so big since an arraylist contains references to objects and not the objects themselves.

One more data structure has to be mentioned, and that is the multi dimensional arrays and how these are implemented. In Java only the first dimension have its elements neatly stored after each other in the computer's memory. Any added dimension after the first makes the extra dimensions added as arrays containing arrays. Since array is an object in Java the second dimension actually adds references to the arrays containing the elements(Fig. 7.2).

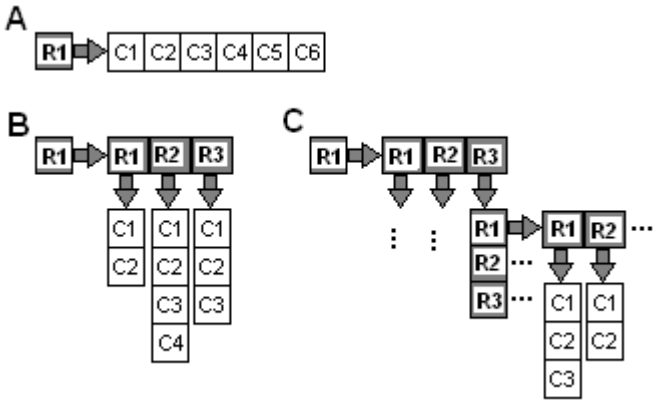


Figure 7.2: Upper left shows a single dimension array. Lower left shows a 2D array in Java. To the right a multidimensional array is shown.

7.5 Classes in Java

Classes in Java can both be static, meaning no objects can be created from them, or classes that object can be created from. Java itself contains most of the classes a novice programmer and/or even an advanced programmer would like to use.

7.5.1 Graphical Unit Interface in Java (GUI)

Java contains its own package of classes that helps build a GUI. These are called awt, swing and the newer Java beans. Java beans are actually reusable software components and are mentioned since GUI components in themselves often are reusable. Swing contains classes for constructing tables, lists, windows, areas to write texts, buttons etc. awt contains much of the same components, but as heavy weight components(harder to put together). The swing model is shown underneath(Fig. 7.3):

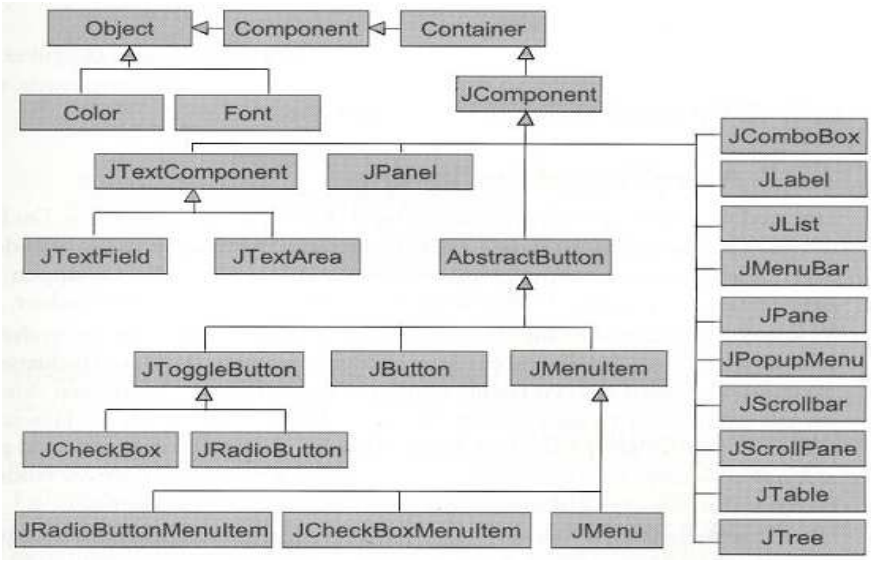


Figure 7.3: Shows a small selection of the class diagram over the swing model. Where component are the highest one except for object, since all classes has a object as upper parent. Classes with J in front of it is from swing, while the rest is awt.

This model's strength lies in that you can combine things. You start with a window, add a panel to it containing a layout manager which helps you place out the visual elements like buttons, lists and so on. The good part though is that you can add another panel to one of the places in the layout manager and this panel contains its own layout manager. This can continue forever. This results in a flexible way to place components on the screen and your imagination is more or less the limit, when looking away from screen space. Although screen place can be gained by using a scroller, which can be added to any swing component, except windows(awt component called Frame).

List and tables uses their own data models to keep track of the data[52], although many like a decoupling of data and GUI and therefore uses strings as their data model elements. Swing runs on what sun has called the event dispatching thread which executes all drawing on the screen and events.

7.5.2 Other Classes in Java

Other classes that needs to be mentioned is classes that helps with the input and output(IO) of a program. The IO classes in Java are linked together by using one class to open files for writing and one for reading, and then use classes that encapsulate this to do more advanced reading or writing, like writing a String (character array) to the file at once. The same IO classes can also be used to write/read to/from the console or read from the keyboard. There exist IO classes to read/write binary, ASCI and Java objects. The last one is in Java called serialization and is meant as a smart way to send objects over a network[52].

8 Implementation

In this chapter some of the implementation of the MassAnalyzer(MAP) will be shown and discussed. For starters there will be an overview of the program. Then continuing with an insight into the program's packages, and last some of the classes within these packages will be discussed more closely.

8.1 Overview

The program can be looked upon as consisting of two parts, which are the GUI layer and the program layer(Fig. 8.1). The left side of figure 8.1 shows miscellaneous packages, the middle shows the main packages and the right shows the utility packages. The main difference is that the utility packages should be stand alone packages. This division into two layers makes it easy to update the GUI without it having any effect on the program itself. At least when coupled to an implementation where the data structures used in the GUI has no real connection to the program layer's data structures. The only connection between the two layers are through the 'massanalyser' package.

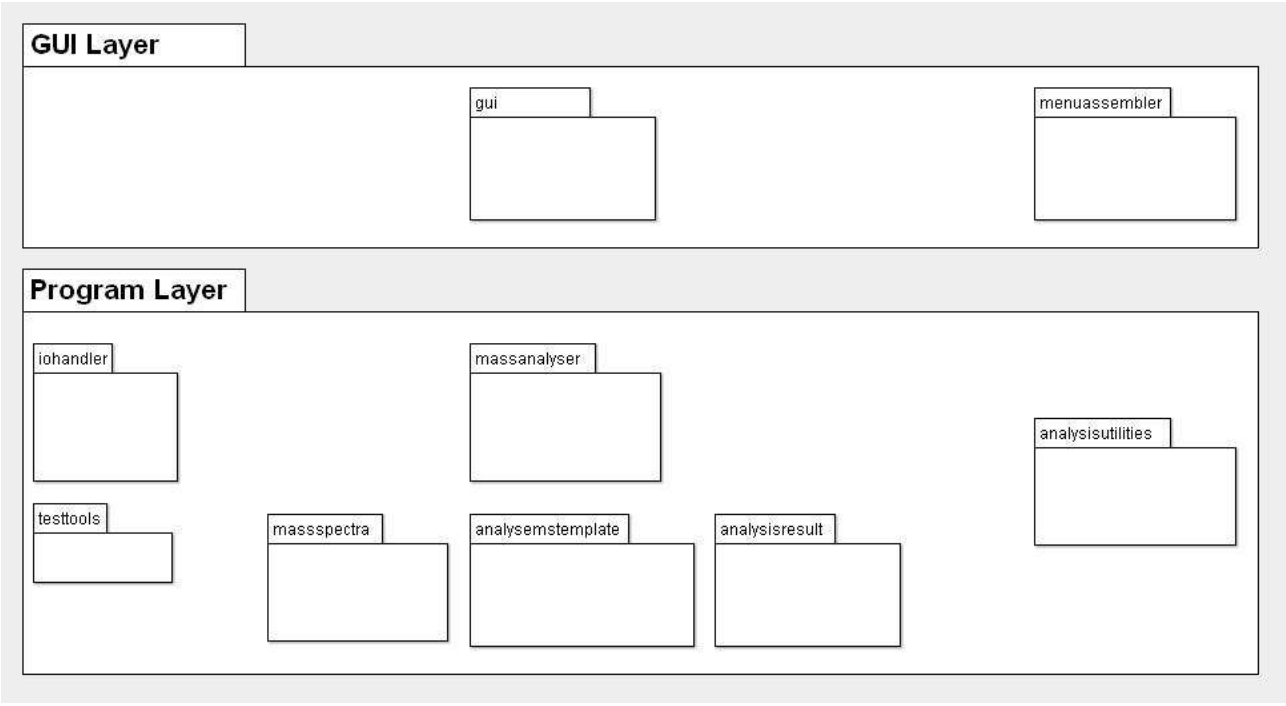


Figure 8.1: The program is divided into two layers, program and GUI. The packages within in each layer is also ordered, to the left misc packages resides, to the middle the main packages resides and to the right the utility packages resides.

8.2 GUI Layer

The graphical unit interface(GUI) is divided into two packages. The main package, called 'gui', contains all the user interfaces, while the utility package called 'menuassembler', helps the program to make a main menu in an easy way.

8.2.1 gui Package

A user needs to be able to interact(user interface) with the data, and this package contains the classes to do so. The two ways a user can interact with the program is to make it display some of the data and modify the data. Two ways to do this are:

- The data structures can be linked directly to the GUI
- The data structure can be indirectly linked to the GUI

The main difference is that in directly linked data structures a change in the GUI will automatically mean a change in the underlying data structures, while the other one means the GUI has to call certain methods in the underlying data structure before the change can occur. The latter one is preferred since it lets the data structures in the program layer to be decoupled from the GUI, which again result in it being easier to update/change the GUI without any major programming. The classes in this package indirectly links the data structures to the GUI by storing a reference to the linking class called 'MassAnalyser' in the 'massanalyser' package.

Figure 8.2 shows the class diagram for the 'gui' package.

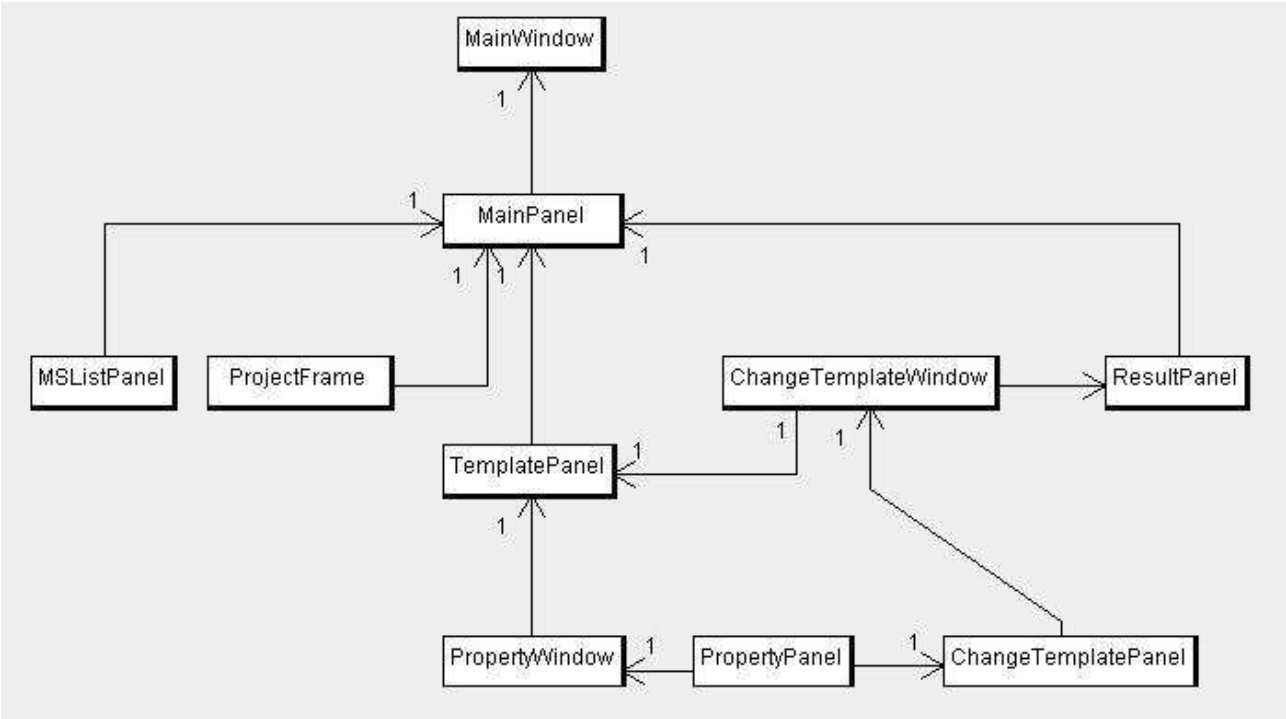


Figure 8.2: Shows a class diagram over all the classes in the 'gui' package.

All of the classes in this package follows a standardized set-up. The constructor calls an 'init' method which places the elements and creates the listeners(which are Java's answer to user interaction). The listeners are classes that monitors changes from the keyboard, pressing of buttons, mouse movement etc. When the user modifies something the classes needs to update their own display as well as sometimes other classes display. This are done through the 'renew' method these classes contains.

The roles of the different classes is explained shortly underneath:

- The main window class takes the 'massanalyser' from the program layer as a parameter and distributes this to the other underlying classes that needs data from the program layer. It also contains the main panel.
- The main panel keeps tab on the other panels in the main window as well as distributes these to the 'menuassembler' class in the GUI layer to get the elements for the main menu.
- The MSListPanel shows the collection of mass spectra currently ready for analysis by using a table.
- The ProjectFrame lets a user create new projects and, load and save them.
- The Template Panel shows the user the current used template as well as allowing them to access the 'ChangeTemplateWindow' to change the analysis methods and the parameters of these. It also contains a direct linkage to the 'PropertyWindow' so the parameters of the selected analysis method can be changed.
- The ChangeTemplateWindow is a window for the 'ChangeTemplatePanel'.
- The ResultPanel keeps a list(table) over the mass spectra which there have been an analysis on as well as ways to show these results.
- The PropertyWindow is a window for the 'ChangeTemplatePanel' when it is shown alone.
- The PropertyPanel shows the properties of an analysis method.
- The ChangeTemplatePanel lets the user see what analysis methods that are chosen as well as change the parameters of the selected ones, which are changed in the 'PropertyPanel'(lower part of the window).

8.2.2 menuassembler Package

Most programs have a main menu, either as a mouse menu or as a bar menu. This package contains a method for the latter. As seen in the class diagram for the 'gui' package(Fig. 8.2) there are many panels that requires their own listeners(Java) that lets the user interact with the program. For a menu bar this means collecting a lot of information from all the classes/panels including the name of the menu items in question. Hard coding it into the main panel is the easy way to go, but it makes the main panel code bulky as well as hard to keep track of for a programmer.

This package gives the responsibility to the classes/panels in question to implement their own menu items. A class diagram of the two classes in this package is shown in figure 8.3. As can be seen there are no links between them, except that the 'MenuAssembler's only method only works on classes that implements the 'AutoMenyAssembly' interface.

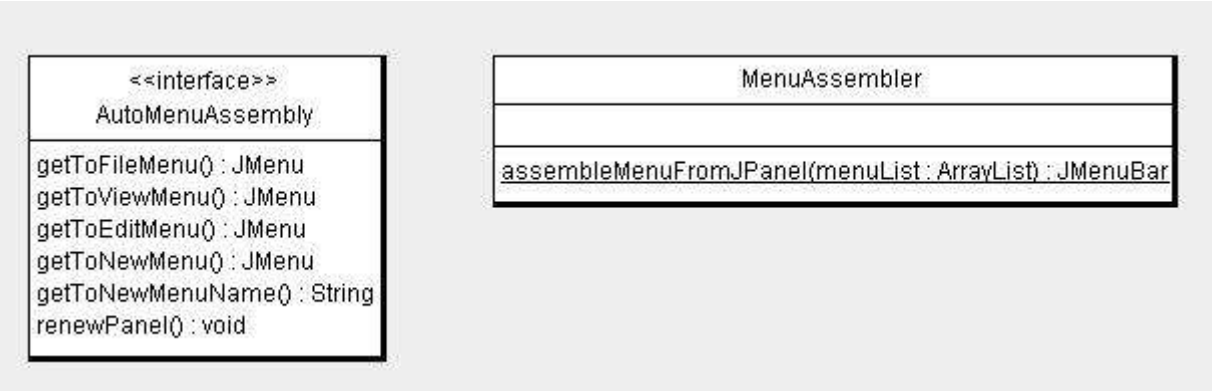


Figure 8.3: Shows a class diagram of the classes in the 'menuassembler' package.

This is how it works:

1. The classes/panels that wants an item in a menu in the menu bar have to implement the interface called 'AutoMenuAssembly'.
2. The class/panel that wants the menu has to implement a list over the panels that wants to be part of the menu. Usually this is done anyway since the underlying panels needs to be linked to somewhere.
3. When building the menu the class/panel calls the method in the 'MenuAssembler' class with the panel list as an argument. Any class/panel that does not implement the 'AutoMenyAssembly' interface gets ignored.
4. The menu bar can be displayed and used.

The 'MenuAssembler' allows the ordinary 'File', 'Edit' and 'View' menus to be displayed, as well as one custom made menu. It also checks if the custom made menu has been created before and add the new menu options to it if it is already there.

8.3 Program Layer

The program layer contains more classes than the GUI layer, and on top of that the classes are more diverse. There are no class that are more central than the other, but to solve the problem in this thesis straightforward the class 'massSpectrum', 'AutoGentzelPeakDetection', 'MarkedMoleculeFinder' and one or two of the IO classes would be needed. The rest makes the program more flexible. As mentioned in the overview the program layer is divided into three parts(figure 8.1). The main part consist of the classes that do the analytic work, the utility part consist of the IO functions as well as some output classes, while the last part is for testing.

8.3.1 massanalyser Package

This package binds the other main classes together as well as providing an interface to the GUI. Figure 8.4 shows the class diagram of the only class in the package.

A more in depth view of the class is in order. The class contains a lot of different data structures:

- Experimental mass spectra list(ESL)
- The currently used template

- A cloned template that are a deep copy of the current working template when the cloning was asked for
- Result list(RL)

On top of all these structures it contains the glue for all the methods that allows flexibility in the choice of analysis methods to be run on a list of mass spectra. These methods takes a single mass spectrum, all mass spectra in the ESL or an index list to mass spectra in the ESL and sends the mass spectra/spectrum to the 'AnalyseTemplate' class in the 'analysetemplate' package for processing. The 'AnalyseTemplate' returns the results in the form of a 'ResultList'(is in the 'analysisresult' package), which again is stored in the RL.

On top of this it contains methods to load and save different parts of the program layer, like the template, ESL and RL. It also contains a file path to where the current project is saved, as well as where the mass spectra in the ESL should be read from.

Figure 8.4: Shows the class diagram of the only class in the 'massanalyser' package.

8.3.2 massspectra Package

This package contains almost all of the data structures in MAP. The view of this package is that every analysis method might requires its own type of structure, which means that the main class of all the data structures(shown in figure 8.5 to the left) are the 'AbsMassSpectrum' classes. Here polymorphism is used again. The most needed methods for every mass spectrum is in the 'AbsMassSpectrum' class, which is an abstract class. The other classes inherits from this class but have different purposes. The 'MassSpectrum' class is the most used one, and consist of a list of 'MZI' values which is another class with polymorphism in it(shown to the lower right of figure 8.5). Another class is 'ResultMassSpectrum' which is the same class as the 'MassSpectrum' but with the added ability to contain an object('ResultHolder' interface) containing information(like max peak intensity and so on) which usually would be wasted to store directly in the data structure since the values would always be the same. The last one is the 'ProxyMassSpectrum' class which contains some information about a mass spectrum but lacks the data values for it, but these can be loaded and unloaded at command. This last one is used to save memory when the mass spectrum does not need to be loaded, usually at any time when an analysis is not done.

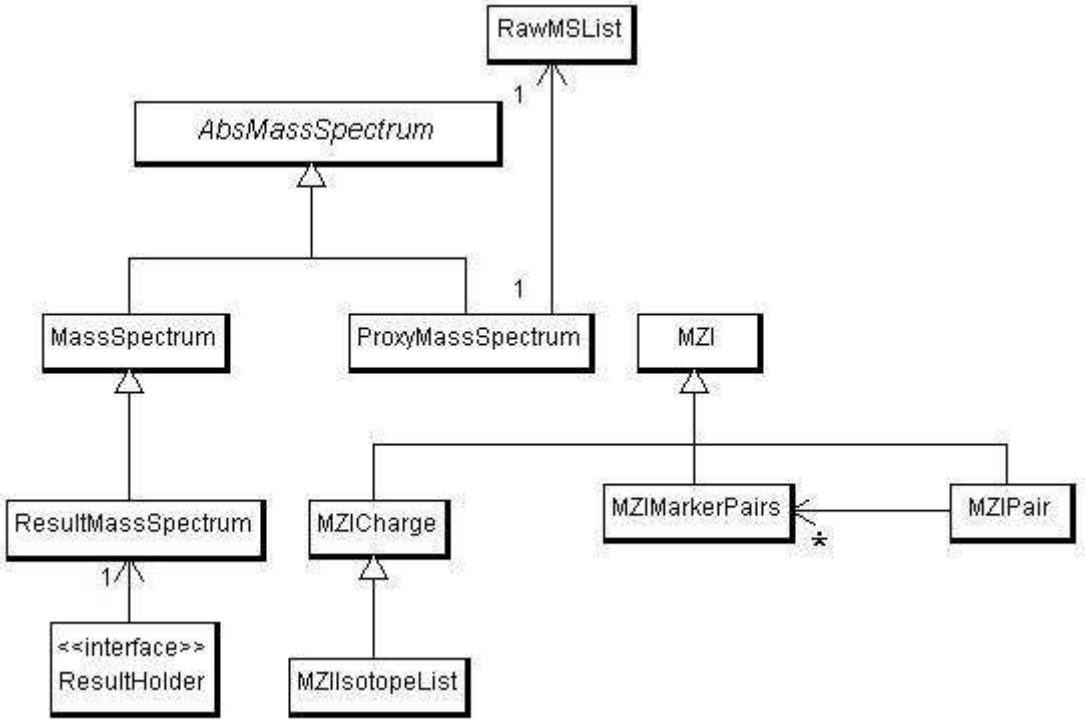


Figure 8.5: Shows a class diagram of the 'massspectra' classes.

The other classes in this package is the 'RawMSList' which contains a list of 'ProxyMassSpectrum' and a 'IOHandler' to load and save these mass spectra. A note to make here is that this is the same 'IOHandler's bound to the individual 'ProxyMassSpectrum's . There is also a method to ensure they stay this way. The reason for this implementation is that all the mass spectra is saved in the same folder but 'ProxyMassSpectrum' is a generalisation using the proxy pattern[49]. Most of the method in the class is to add, remove and so on from the mass spectra list. There is one more thing to be said about the 'RawMSList' though and it is that it uses the interface called 'AnalyseList' in the 'AnalysisUtilities' package. This interface's main goal is to make sure the class can easily make a 'copy' of the contents it wants to show in a GUI table.

The last part of the package, that was mentioned above, is the 'mzi' sub package that contains the data object for the 'MassSpectrum' class. This package contains the class 'MZI' that the other classes inherits from. This class is simply an object saving the two variables M/Z and intensity. The classes that inherits from this class stores more variables used for different analysis methods. A note here is that the polymorphism gives you the variables stored in the super class 'MZI' so the most important variables should be saved here. Although there are no problem if they do not, since Java lets its classes to be easily known through the command 'instanceof', and the programmer can check for the right subclass before analysing a value. The sub classes are:

- MZICharge which contains the extra variable charge
- MZIIsotopeList which contains a list of MZICharge
- MZIMarkerPair which is used in the 'MarkedMoleculeFinder' class, in the 'AnalyseAlgorithm' package, contains a deep structure.
- MZIPair that contains another 'MZI' value making the two values be a pair. As well as a variable for when they were found

The 'MZI' class and its subclasses all contains methods that put their data into a string. These are inherited so that every class gets to write their data when an object stored as an 'MZI' object is asked for a printout of its data.

The 'MZIMarkerPair' needs a closer look(figure 8.6).

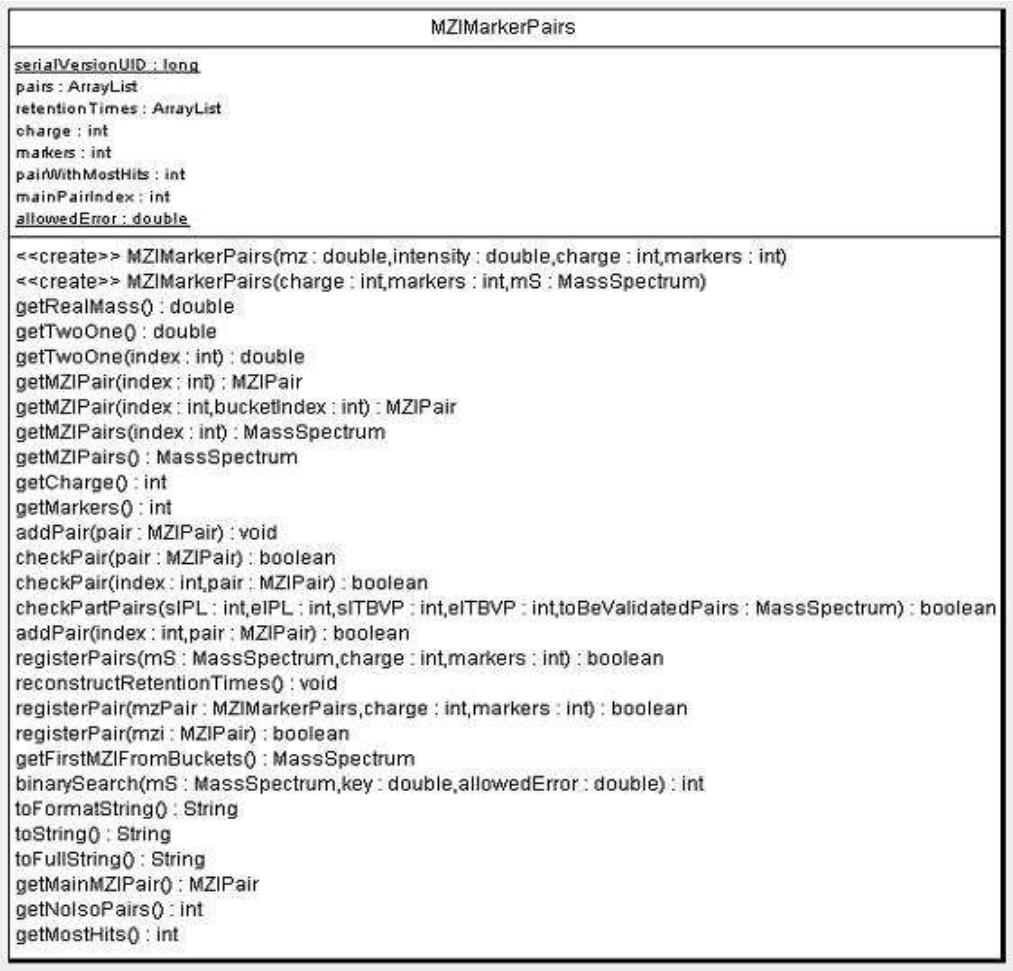


Figure 8.6: Shows the class diagram of the 'MZIMarkerPair' class.

This class is used in the 'MarkedMoleculeFinder' analysis method in the 'AnalyseAlgorithm' package. The structure itself contains a two dimensional list, where the second dimension contains 'MZIPair's(Fig. 8.7). The R1, C1 and C2 linked to it is the same 'MZIPair' but found at different retention times. The same goes for R2, R3 etc. A mass spectrum generally has a lot of these 'MZIMarkerPairs' in it, making the final structure a three dimensional one. The class also stores the most abundant of the pairs as the main pair('MZI's initial (M/Z, intensity) pair). It also contains another two dimensional structure that contains the same objects as the previous array(meaning no more memory is used to store them), but sorted on the retention times each of the isopairs were found.

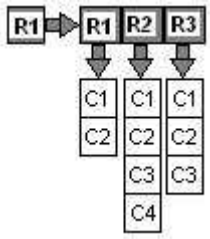


Figure 8.7: Two dimensional list, also shown in chapter 7.4 figure 7.2 B. Here R? is the reference to the list of 'MZIPair' where every element in the list(referenced by an R?) contains the same M/Z value but with different intensities and retention time.

To be able to do all its chores it also needs to store the charge and number of markers(see SILAC chapter 4.1 for explanation) for this particular 'MZIMarkerPairs'.

The interesting methods in this class is the methods that checks if a pair matches the ones already stored in this particular structure. The 'CheckPair' methods take different arguments but they end up doing the same which is to check if a set of isopairs satisfy the criteria to be added to the list. The criteria is that there are at least two values matching the ends of the current values or all of the isopairs matches all the ones already in the structure, and they matches the charge and number of markers for this 'MZIMarkerPairs'. The 'RegisterPair' does the same but also takes a 'MassSpectrum' as an argument, meaning that it goes through this mass spectrum to see if every isopair in it matches with this particular marker pair. The check methods only check if the set of isopairs matches the stored ones and returns the boolean value true if this is true.

This class also have methods to calculate the total intensity of the heavy isotope divided by the total intensity of the light isotope(also called 2/1). It can also calculate the the real mass of the main pair.

The last method that should be mentioned is that it uses a modified binary search to search the pair list. Here modified means that it searches on values, of the primitive type double, in a sorted list. It uses a set interval around the wanted value to find the right one. This is all good as long as the values are not closer than the interval(which is relative small) to each other.

8.3.3 analysemstemplate Package

This package contains two sub packages, which are:

- 'analyseproperities' which contains generalized parameter classes
- 'analysealgorithms' which contains all the analysis methods

This can be considered one of the most important packages since it contains all the analysis methods, and it is the most likely to be updated at a later stage by adding new analysis methods.

To set and get the parameters in the analysis methods in the 'analysealgorithms' package the package 'analyseproperties' implements a set of general classes. The class diagram for 'analyseproperties' can be seen in figure 8.8. As can be seen in the figure an interface is the highest class, 'AnalysisPropertyInterface' in the hierarchy. This interface act as a template that the rest of the classes in the class tree have to follow. The next in the tree is the abstract class 'AbsAnalysis-Property' and it implements some of the methods of the interface as well as adds some constructors. All the classes branching from this class implements a set of methods to be able to get their data out as boolean, decimal, integer or String, as well as a method to convert a string into the class' stored value. Figure 8.9 shows an expanded class diagram view of the 'AnalysisPropertyInterface' to the left and the 'DecimalProperty' to the right. As can be seen from this figure the interface forces the class to implement all the necessary methods for the parameters to be set and used, while the class itself adds the extra methods for it to be initiated/set properly. The class 'DecimalProperty' has the ability to set if the value should be rounded up or down(in an object variable) when it is returned as an integer with the method 'getInteger'.

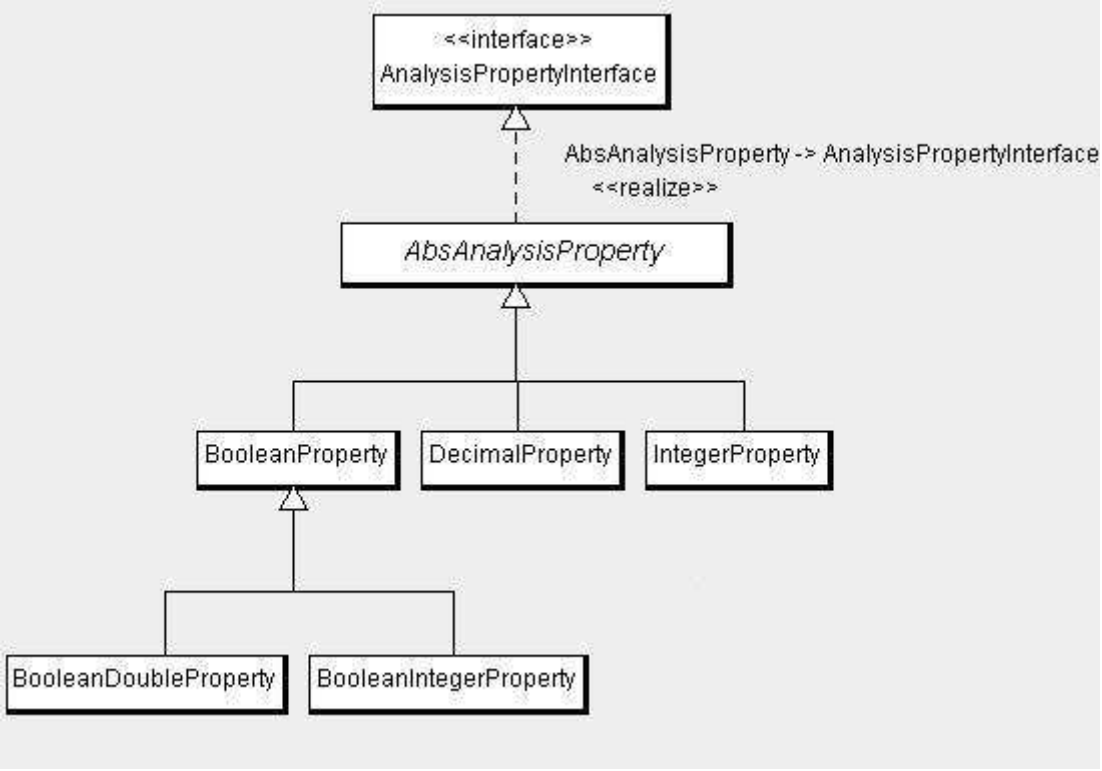


Figure 8.8: The class diagram for the 'analyseproperties' package.

The two classes 'BoleanDoubleProperty'(BDP) and 'BooleanIntegerProperty'(BIP) is subclasses to the 'BooleanProperty' class. BDP adds a decimal variable to be returned by the 'getDouble' method, one decimal variable can be set for when BDP's main variable is true and one can be set for when it is false. BIP does the same for the 'getInteger' method. The returned value for the super class('BooleanProperty') for these two methods are 0 for false and 1 for true.

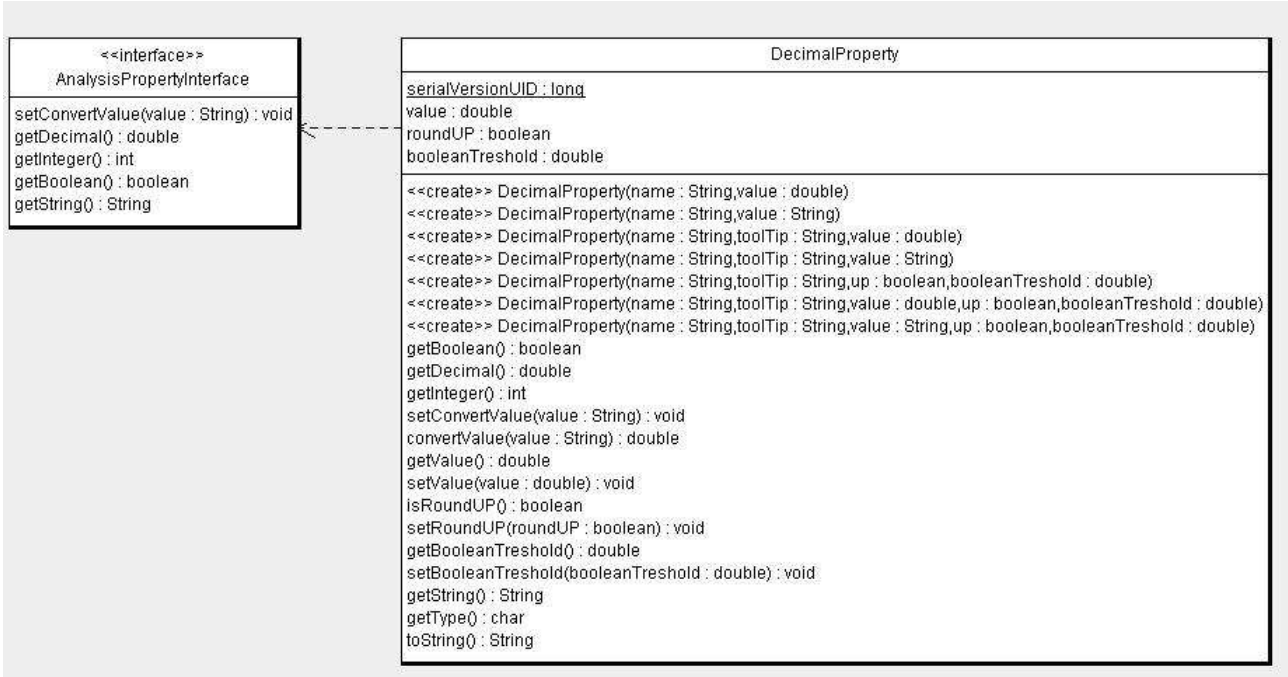


Figure 8.9: The 'AnalysisPropertyInterface' to the left and the 'DecimalProperty' class to the right.

The next step is to use these parameter classes in the 'analysealgorihm' package, which is shown in figure 8.10. In this figure the 'AbsAnalyseAlgorithm'(Fig. 8.11) clearly shows that it is paramount to the analysis methods. This class has it sole purpose to implement methods to get and set the parameters of the analysis method as well as provide some unfinished classes that its children needs to implement to work properly and to be able to produce results as text. The text results can be recreated from the 'ResultList'.

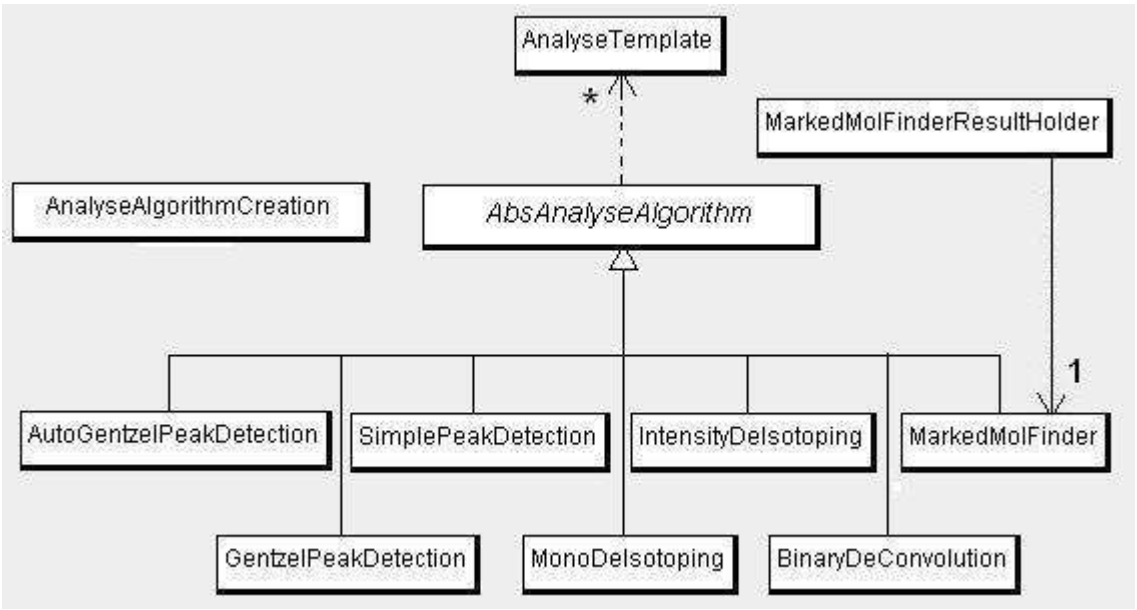


Figure 8.10: The class diagram for the 'analysealgorithms' package, also showing 'AnalyseTemplate' class and 'ResultHolder' interface from the main package.

Figure 8.10 also shows that the 'AbsAnalyseAlgorithm' class is tied to the 'AnalyseTemplate' class. This class contains methods to add, remove and get analysis methods from a list of 'AbsAnalyseAlgorithm' objects that it holds. It also contains the methods that do the analysis on a mass spectrum or a set of mass spectra. The last mentioned methods uses the polymorphism in the 'AbsAnalyseAlgorithm' to manage the analysis. They also uses the polymorphism in the mass spectra themselves. The 'AnalyseAlgorithmCreation' helps the class to create new analysis methods to put in its list.

The bottom part of figure 8.10 shows all the currently implemented analysis methods. These are built on different algorithms which are shown in chapter 5.7 and 5.8. For this thesis the two most important classes among the analysis methods are the 'AutoGentzelPeakDetection' and the 'MarkedMolFinder'(fig 8.11). Both will get a much closer look.

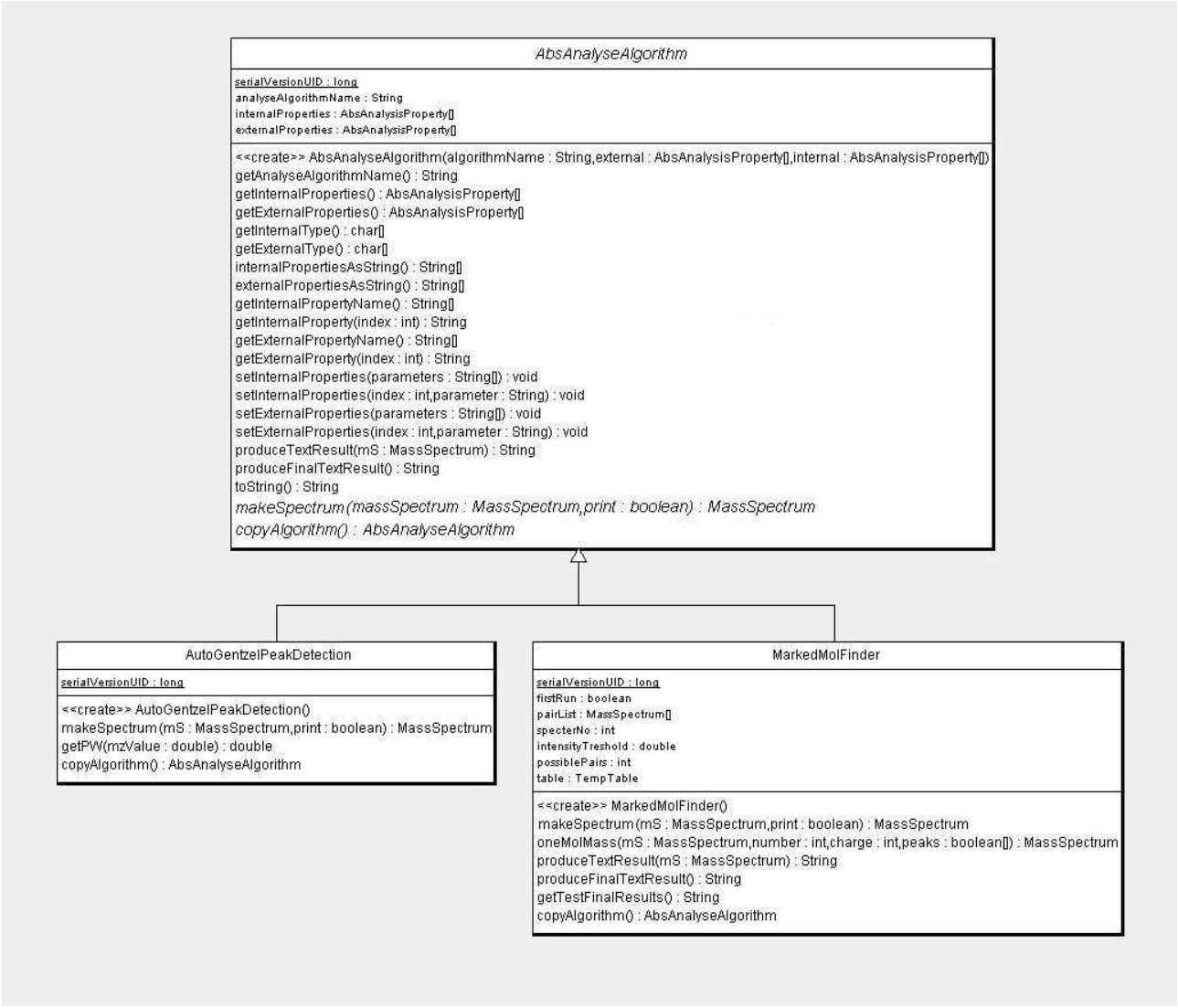


Figure 8.11: Shows the two classes 'AutoGentzelPeakDetection' and 'MarkedMolFinder' with their super class 'AbsAnalyseAlgorithm'. The rest of the classes that inherit from the super class is hidden to simplify the view.

The 'AutoGentzelPeakDetection' class contains an analysis method for peak detection based on Gentzel's peak detection algorithm[34]. As can be seen there are not many methods implemented for this one, the 'getPW(x)' is to get the peak width for the M/Z value x(chapter 3.4.3). The analysis method itself is in the 'makeSpectrum' method. This analysis method exploits the fact that if the

threshold is not exceeded the area does not need to be traversed again. Just remove some peaks at the front and add some at the end. The amount of peaks added and removed depends on the new calculated pw(x) for the current M/Z value looked at. Since this analysis method does not add the found peaks to secondary structures and search them afterwards it can only find double overlaps, not triples or higher(Fig. 8.12). If a peak is found on the other hand it sets the new search to start at the end of this peak. This means that any new peak with the apex before this value will not be found, or the peak M/Z for the peak apex for this peak might be a little of.

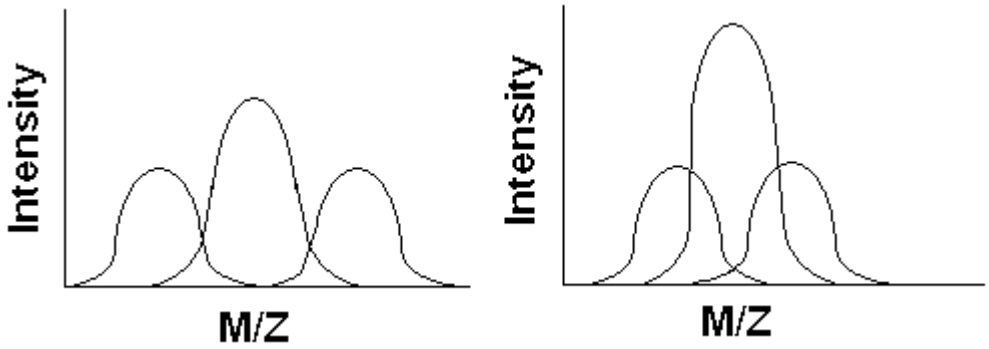


Figure 8.12: To the left there are two double overlapping peaks. The middle with the left one and the middle with the right one. To the right all the three peaks are overlapping.

The 'MarkedMolFinder' implements the SILAC pair finder analysis method, but it can also be used for any markers as long as the mass difference between the marked peptides and the same unmarked peptides is known. This class is the only class to make use of the 'ResultMassSpectrum' in the 'massspectra' package at this time. This class has a lot of parameters and they are:

- Pair threshold(percentage), the intensity threshold for the most intense peak in the potential pair
- Heavy/Light threshold(2/1), heavy isotope intensity divided by light isotope intensity
- Marker weight, the molecular weight of the molecular marker. Can also be looked upon as the mass difference between an unmarked peptide and a marked peptide(if there is only one marker in the marked pepetide)
- Max marked residues, the maximum allowed number of markers for any pair
- Min charge, the minimum allowed charge for any pair
- Max charge, the maximum allowed charge for any pair
- Isotopes required(2+), the threshold of isopairs a pair needs to exceed
- Max retention time difference, the largest retention time difference between the two closest pairs in a marker pair
- Allowed error, is the decimal error allowed in the analysis method

As algorithm 7 in chapter 5.8 shows it is divided into two parts. In the first part all the possible pairs up to the maximum charge(z) and number of markers(n) are found by the use of a greedy algorithm. Remember the difference between marked and unmarked peptides are:

$\frac{n * m_{weight}}{z}$, where m_{weight} is the marker weight.

The greedy algorithm works on the principle that if it is to low for me it is also to low for you. This

principle works for a list that is sorted and where the z and m are kept constant during the search. Figure 8.13 shows the principle in action.

In words this means:

1. Start with the first M/Z value in the mass spectrum as the start M/Z(SMZ) and the second one as the next M/Z(NMZ).
2. The difference between the SMZ and NMZ is checked. If it is lower than the equation above NMZ is set to the next M/Z value proceeding it in the mass spectrum.
3. Repeat step two until the difference between them are equal or higher than the value from the equation above.

    1. If it is equal an isopair is added to a new 'MassSpectrum' and SMZ is set to the next M/Z value proceeding it in the mass spectrum.

    2. If on the other hand the difference is higher then only SMZ is moved.

4. Go to step 2 until NMZ goes over the end of the mass spectrum, since every difference between SMZ and NMZ after this will always be lower than the equation.

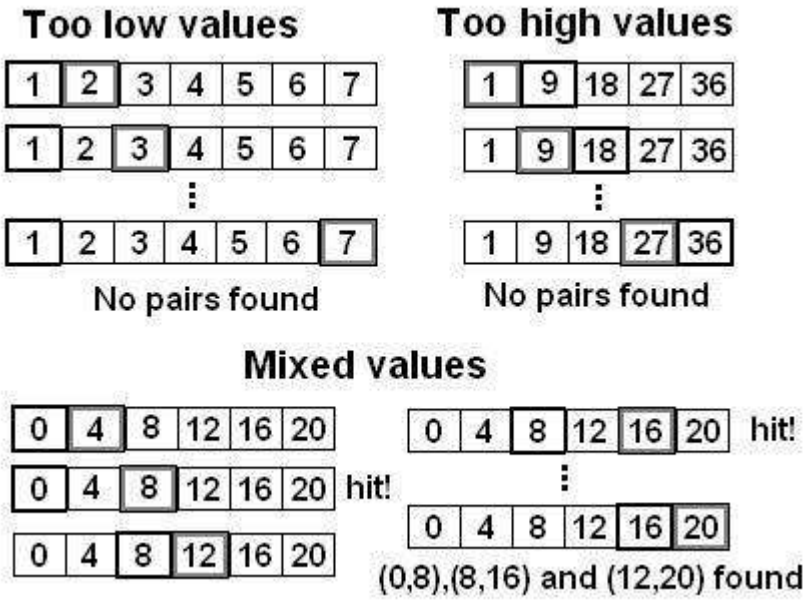


Figure 8.13: Shows three runs of the greedy algorithm where the distances should be 8. Top left shows one where all the values are to low. Top right shows one where all the values are to high. The bottom one shows one where all the values are mixed(usual occurrence) and the pairs (0,8), (8,16) and (12,20) was found. The Black border square is SMZ(start M/Z) while the grey squared is the NMZ(next M/Z).

The second step is to go through all the possible isopairs and see if any of them satisfies the criteria for being a marker pair, which are that at least two or more isopairs shares the same isotopic envelope (set in the parameters, but minimum 2) and that the most intense peak in the pair is above the threshold. The second step also to some extent uses the same principle as the greedy algorithm above, but only as a way to end the algorithm early. What is the main greedy approach here is that no pair from the possible pair list(PPL) can be in more than one marker pair, as long as z and m is kept constant. This holds true as long as the difference between two M/Z values is lower than the allowed error, which in all practical cases holds true. Example: a charge of 10, which is incredible high, would give an isotopic difference of 0.1 which is quite high compared to any sensible allowed error. This greedy implementation means that every isopair found to be in a marker pair is removed from the PPL, decreasing the list by each iteration.

The process is shown in figurer 8.14. In words this means:

1. Start with the first M/Z value in the mass spectrum as the start M/Z(SMZ) and the second one as the next M/Z(NMZ). Remove the SMZ from the mass spectrum, but remember its value.
2. The difference between the SMZ and NMZ is checked. If it is lower than the currents charge isotopic differences(1/Z).
3. Repeat step two until the difference between them are equal or higher than the value from the equation above.

   1. If the distance is higher check if the current set of isopairs satisfy the criteria to be turned into a marker pair. Do it if they do, else discard it. Reset the current set of isopairs. Go to step 1.

   2. If the difference is equal then add the isopair to the current set of isopairs and remove it from the PPL. This can be done since the distance between two peaks need to be lower than the allowed error to match up to the same possible pair. Move the SMZ to be in the same position as the just removed possible pair. Move the NMZ to be in the position just after SMZ.
4. Go to step 1 until NMZ goes over the end of the mass spectrum. Check if SMZ is at the start and NMZ is at the end of the possible pair list. If this is so check 3.1 before ending the algorithm, since this means that every difference after this will be to low for an isopair to form. If not check as in 3.1and go to step 1 until the list is empty.

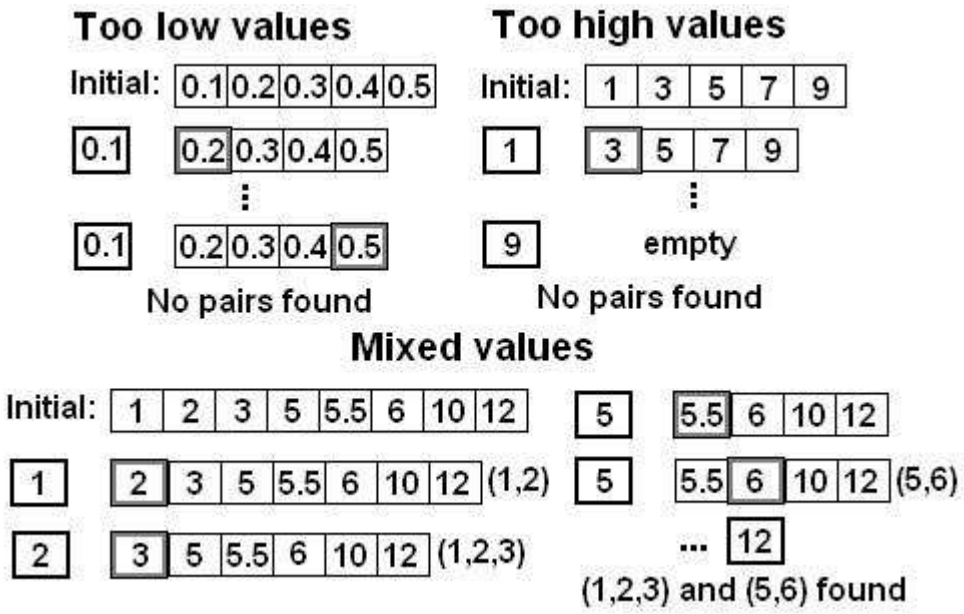


Figure 8.14: Shows three runs of the greedy algorithm where the distances should be 8. Top left shows one where all the values are to low. Top right shows one where all the values are too high. The bottom one shows one where all the values are mixed(usual occurrence) and the pairs (1,2,3) and (5,6) was found. The Black border square is SMZ(start M/Z) while the grey squared is the NMZ(next M/Z).

The marker pair list contains all the marker pairs found for the current z and m. The whole procedure will be repeated until all the possible combinations of z and m is searched.

For every mass spectrum checked the results will be added to the global result list in the 'MarkedMolFinder' class. This is a two dimensional 'MassSpectrum' list containing objects of the

'MZIMarkerPairs' found in the 'massspectra' package. Where the dimensions are the max z and m values. To ease the strain of the program only the part of the global list that has the same charge and number of markers will be checked as explained in the 'MZIMarkerPairs' class description.

8.3.4 analysisresult Package

This package contains two classes that stores the results produced by the 'AnalyseTemplate' class in the 'analysetemplate' package. The data structure itself is stored in the 'MassAnalyser' class in the 'massnalyser' package. Figure 8.15 shows an expanded class diagram of this package.

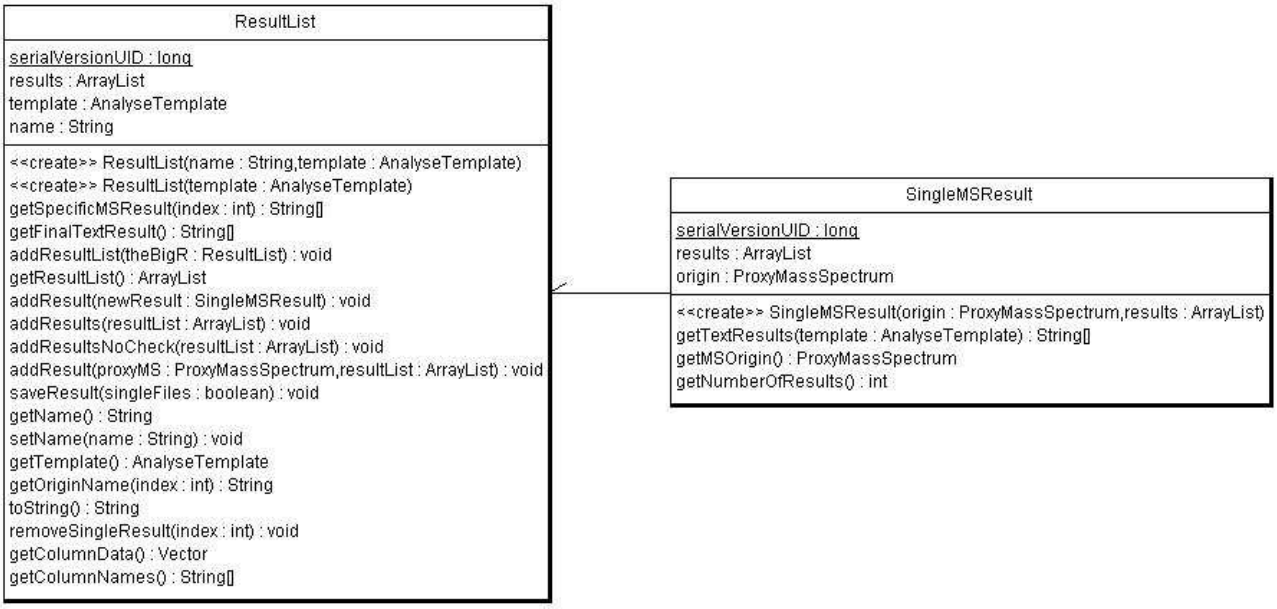


Figure 8.15: Shows the class diagram of the two classes in the 'analysisResult' package.

The 'SingleMSResult' stores all the results as 'MassSpectrum' from the analysis methods for a specific experimental mass spectrum(in ESL) which is referenced to as a 'ProxyMassSpectrum'. An example 'AnalyseTemplate' containing a method for peak detection and one for deisotoping run on four different mass spectrum. This would produce four 'SingleMSResult' objects each linking to one of the four mass spectra. These would also contain a list of two mass spectra containing the results from each of the two analysis methods. The 'getTextResults' returns a String array containing the results.

The 'ResultList' on the other hand contains a list over all the 'SingleMSResult's as well as the 'AnalyseTemplate' that produced these. This is to ensure that the result can be recalled easily. It also contains a method called 'getFinalTextResult' that returns the String representation, as an array, of the global result for each analysis method in the 'AnalyseTemplate' stored in this object. This class also allows to remove, add and get a specific result within its list. It also implements the 'AnalyseListData' from the 'analysisutilities' package allowing it easy access to a copy of its data(used in the GUI to build a table).

8.3.5 iohandler Package

This package has two sub packages:

- 'massspectra' which contains methods to save and load mass spectra.
- 'project' which contains methods to save the 'MassAnalyser' class in the 'massanalyser' package as well as the parts in this class, which are the ESL, RL and 'AnalyseTemplate'.

Figure 8.16 shows the class diagram for both sub packages.

The 'massspectra' sub package contains a hierarchy that follows the bridge pattern[49]. This pattern lets the program have many implementation of the same type of algorithm, based on the operative system, user choices etc. This means that the program can easily exchange its current used algorithm in runtime to another one if need be. To be able to do this the pattern contains a class the program uses directly, a policy class that gives the rules when the different implementation should be used and an interface that connects the algorithm classes together under one type.

The classes under the interface 'MSLoader' and 'MSSaver' are the algorithm classes. The 'MSPolicyLoad' and 'MSPolicySave' are the policy classes, while the 'IOHandler' presents methods to easily create instances of the implementation classes without calling them directly. In this implementation the 'IOHandler' acts for both the 'MSLoader' and 'MSSaver' bridge pattern.

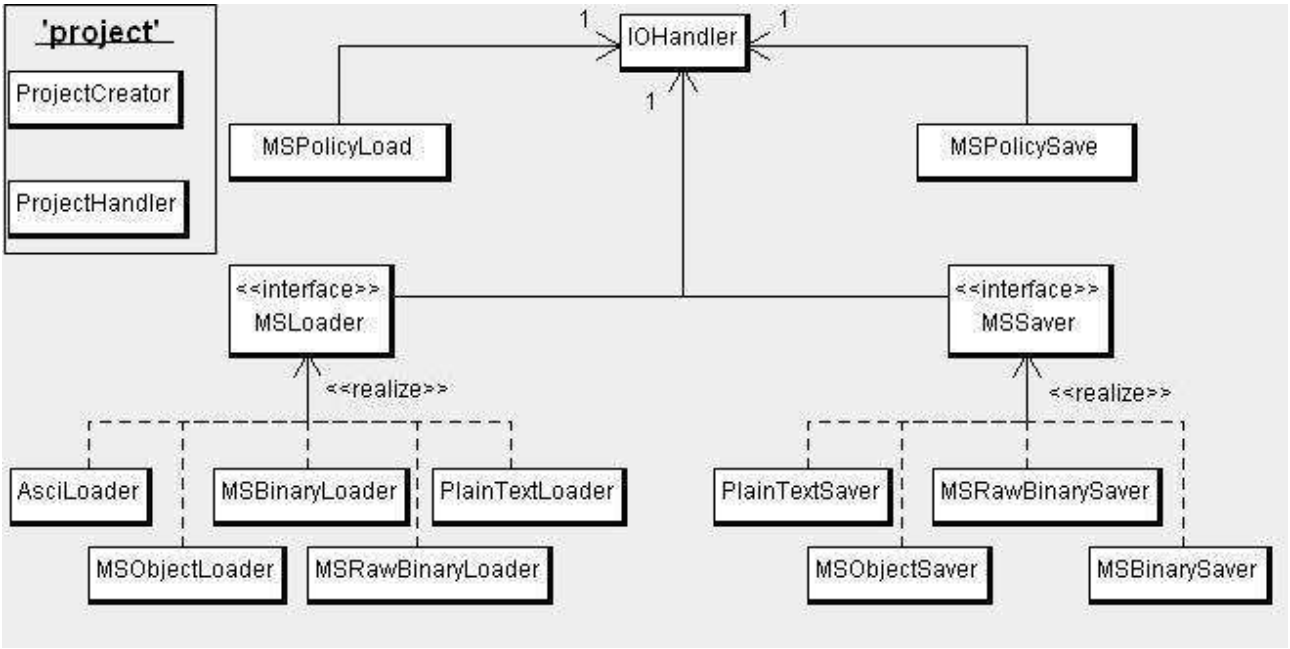


Figure 8.16: Shows the class diagram of the package 'iohandler'. The sub package 'project' is the two classes to the upper left. The rest is from the sub package called 'massspectra'.

The classes inheriting the interface 'MSLoader' is:

- 'AsciLoader', a class that loads mass spectrum from the program Databridge.
- 'MSObjectLoader, a class that loads a mass spectrum saved by the serialization classes in Java. This is literally an object dump.

- 'MSBinaryLoader, a class that load a mass spectrum based on this program's own binary format. The set-up for the format is as follow:
    - Original name;
    - Current name;
    - Two character code followed by the data in the 'MZI' type it codes for
- 'MSRawBinaryLoader', a class that loads the raw mass spectrum themselves based on this program's own format. The set-up for the format is as follow:
    - Original name;
    - Current name;
    - The data for an 'MZI' object, meaning the pair (M/Z, intensity) of the primitive type double
- 'PlaingTextLoader', a class that loads a mass spectrum list taken from within MassLynx.

The classes inheriting the interface 'MSSaver' is:

- 'PlainTextSaver', a class that dumps the String from a 'MassSpectrum's output method.
- 'MSObjectsaver', a class that dumps the object to file using the serialization classes of Java.
- 'MSRawBinarySaver', a class that saves the format listed in the 'MSLoader' class of the same name.
- 'MSBinarySaver', a class that saves the format listed in the 'MSLoader' class of the same name.

8.3.6 analysisutilities Package

This package contains some small support classes. Figure 8.17 shows a class diagram of these. The 'AnalyseListTableModel' is a class that helps the GUI create a table. The interface 'AnalyseListData' is used in conjunction with the previous class by making sure the class that wants to show some or all of its data in a table(in the GUI) have ways to name the columns of the table as well as fill it with data. The 'LineOutputFormater' is a class containing methods to format text output, like making a table in a String by using a set size for the table elements and the room between the table elements. The last class called 'TempTable' is used in some of the analysis methods to store two dimensional tables, currently only the primitive type integer is allowed.

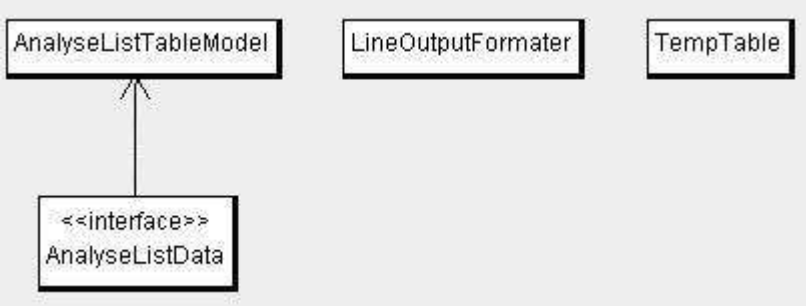


Figure 8.17: Shows the class diagram of the 'analysisutilities' package.

8.3.7 testtools Package

This package contains classes to test the other classes. Figure 8.18 shows a class diagram for this package. The 'MassAnalyserCreater' class was used to test the GUI and its elements. The only thing it does is to create a fully loaded 'MassAnalyser' with a RL, ESL and 'AnalyseTemplate'.

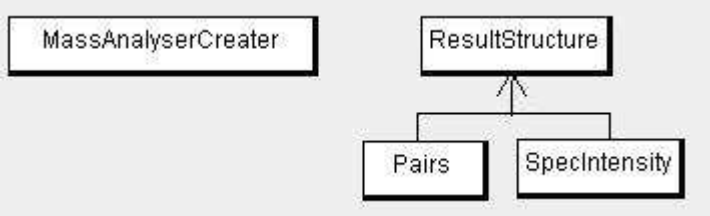


Figure 8.18: Shows the class diagram of the 'testtools' package.

The 'ResultSpecter' needs a deeper look into though. This class is used in conjunction with a modified 'MarkedMolFinder' (only contains ways to send data to this class). When an instance of this class is created it takes a text file containing the 'solution' to a 'MarkedMolFinder' run. This 'solution' is stored into different lists of the inner class called 'Pairs'. The pair inner class contains a light and heavy isotope of the 'solution' pair, number of charges, markers and what the name of the marker is(example 'Lys'). There are three different lists:

- Pairs in list, this list all the pairs contain all the data that need to be known.
- Pairs not in list, this list contains all the pairs where one of the isotopes are missing.
- Unknown pairs, this list is empty at the start but will in the end contain all the pairs found by the 'MarkedMolFinder' analysis method but not in the 'solution' list. These can be false positives (FP) which means that they are not really pairs, or they can be false negatives(FN) which means they should be there but the 'solution' did not pick it up(can be the case if the solution is based upon another algorithm or is incomplete).
-

All these lists are two dimensional where the charge and number of markers are used to retrieve/set the content in it. This effectively means that all the pairs in the list are sorted on these two variables. This implementation makes it easier on the program since it needs sorted lists to function properly, and this leaves the program to sort on one data value(M/Z) instead of three(M/Z, charge and number of markers).

The 'Pairs' inner class contains three lists within it. These three registers different amount of isopairs in the marker pair. One list for no isopairs(means single pair), one for two marker pairs(means two pairs) and one for three or more marker pairs. This class have methods which tells what list a pair should be assigned to. The main class('ResultStructure') also have a method that delegates it on to the 'Pairs' class. The 'Pairs' class also contain methods to calculate the real mass and the 2/1 for their own pair.

The 'SpecIntensity' is an inner class like 'Pairs' and are used to save the intensity for every occurrence the 'MarkedMolFinder' finds during a run. Example: a marker pair in the 'solution' is found three times by the 'MarkedMolFinder', since all the marker pairs have the same M/Z values it is redundant to save these. So instead each of the three marker pairs save their intensities in one 'SpecIntensity' object each, which is kept within the 'Pairs' itself. The 'SpecIntensity' also contains the data for when the marker pair was found.

The 'ResultStructure' class itself contains a lot of methods to register the pairs, not unlike the 'MZIMarkerPairs' class in the 'massspectra' package. These methods check if the the main peak of the 'MZIMarkerPairs' matches an M/Z value with one in the solution, with the same charge and number of markers. The possible results are:

- If it does not match any it goes into the unknown pairs in list.
- If it matches perfectly it goes into the pairs in list.
- If it matches the heavy or light isotopes to one of the pairs in the pairs not in list. Then the opposite peak is calculated and the new pair is put into the pairs in list. It also tags the new pair as a false true pair since it was not in the list from the start.

All of these searches uses the same kind of modified binary search as the 'MZIMarkerrPairs' does.

When the whole 'MarkedMolFinder' analysis method is done running the results within the 'ResultStructure' can be displayed with all the information, only pairs in list, pairs in unknown list with more then three iso pairs or as a simplified view of how many of the 'solution' pairs where found.

9 Conclusion

The Pair Finder analysis method works well on the mass spectra it was tested on, it also found new marker pairs that have a potential to be the same peptide but with different isotopes of Leucine. What is also clear is that the preprocessing done before running the Pair Finder is very important, since this can remove false positives among the multitudes of potential marker pairs. At the moment the analysis method produces a big list of potential pairs to be examined by the user. This list can be reduced drastically by adjusting the parameters to fit the mass spectra data. Also; adjusting the parameters will result in fewer false positives among the potential marker pairs.

What would really help to make the list more readable, would be to add a scoring scheme after the Pair Finder method. This could take into account:

- How many mass spectra in a row the marker pair appeared in
- How many isopairs are in the marker pair
- If any of the peptide/marker pair appears with different charge
- Other information produced by the Pair Finder.

The fact that the Pair Finder analysis method can be used with other labeled methods than SILAC, lends it flexibility. The only criteria is that the label is visible in a single MS experiment.

The Pair Finder, like most analysis methods, has its weaknesses. Most of these should be easy to catch by a researcher with some knowledge about MS.

The peak detection analysis methods performs well, although the AGPD has a problem removing noise from mass spectra when there no high peak. This can easily be rectified by adding a baseline correction and noise reduction methods before the peak detection.

The deisotoping methods do not try to discern between overlapping isotopic envelopes and list them all. Most likely these should not be used except for analysis methods that needs this kind of data. Both analysis methods uses the same principles, and can be made better by considering isotope envelope templates to differ between the isotopic envelopes.

The deconvolution analysis method performs badly, because there are no good deisotoping method to insert before it.

The program is flexible since the user can decide the order and what analysis methods to be used. On the down side this might require the user to know more to get the right results. This can easily be circumvented by loading a set of analysis methods with a preset order, which has been created by another user.

Another good side of the program is that it is user friendly, and gives a good overview of the experimental data and results. It also let the user load and save all of the parts, like the experimental data list, result list and the template containing the analysis methods.

The downside to the program is that it needs time to convert the mass spectra to the right data format. This is mostly due to the fact that MassLynx has its own proprietary data format.

All over the program is easy to maintain at a later point, like adding new analysis methods, new GUI and so on.

9.1 Possible Future Upgrades

Most programs are not finished when they are done, there are always room for improvement in some way. This sub chapter will mention some of the thoughts for improvement of this program.

- Create a good baseline correction analysis method
- Create a good noise reduction method
- Upgrade the program to be able to show the results as more than just text. Graphs for mass spectra and tables for insight into data structures are two ways to improve the readability of the results.
- Scoring Scheme for the Pair Finder. This can be made as an analysis method and to be run just after the Pair Finder method.
- Be able to add new analysis methods to the end of the result template, avoiding a complete rerun of the already created results.
- Add filter parameters to the analysis methods. These parameters would not modify the analysis method itself, only the results shown.
- Add a way to load the mass spectra as mzXML, an open XML format for mass spectrum.
- Get the 'View' menu to work in the main menu bar in the GUI, or remove it.

[1] **History of Immunology, introductory article to the book with the same name**
A. M. Silverstein, Johns Hopkins University School of Medicine, Baltimore, Maryland, USA
(http://mrw.interscience.wiley.com/emrw/9780470015902/els/article/a0003078/current/pdf)

[2] **The "Undiscovered" Discovery**
W. M. Stanley
Archives of Environental Health, Vol. 3, (Sept. 21, 1970), p. 256-62.
(http://profiles.nlm.nih.gov/CC/G/M/G/V/_/ccgmgv.pdf)

[3] **Induction of transformation by a desoxyribonucleic acid fraction isolated from pneumococcus type III**
Journal of Experimental Medicine, Vol. 149, Issue 2 (Feb. 1, 1979), p. 297-326
OT Avery, CM MacLeod and M McCarty

[4] **The complementary structure of deoxyribonucleic acid**
F.HC. Crick and J.D. Watson
Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, Vol. 223, No. 1152 (Apr. 7, 1954), p. 80-96
(http://www.jstor.org/stable/99239?seq=1)

[5] **Discovering genes are made of DNA**
M. McCarty
Nature , Vol. 421, (Jan. 23, 2003)
(http://cmbi.bjmu.edu.cn/news/report/2003/DNA50/source/406.pdf)

[6] **Involvement of RNA in the Synthesis of Proteins**
J. D. Watson
Science, New Series, Vol. 140, No. 3562 (Apr. 5, 1963), p. 17-26
(http://www.jstor.org/stable/1710716?seq=9)

[7] **DNA Sequencing with Chain-Terminating Inhibitors**
F. Sanger, S. Nicklen and A. R. Coulson
Proceedings of the National Academy of Sciences of the United States of America, Vol. 74, No. 12 (Dec., 1977), p. 5463-5467

[8] **DNA diagnostics in the fifty-year retrospect**
V. V. Demidov
Expert Reviews, Vol. 3, No. 2 (Mar., 2003), p. 121-124

[9] **Discovery of the Secrets of Life Timeline**
www.historyofscience.com
P Lesk and N Reynolds
(http://www.historyofscience.com/pdf/discovery-secrets-life-timeline.pdf)

[10] **Stable configuration of polypeptide chains**
L.Pauling, R.B. Corey
Proceedings of the Royal Society of London. Series B, Biological Sciences, Vol. 141, No. 902 (Mar. 11, 1953), p. 21-33

[11] **The Introduction of Computers into Systematic Research in the United States during the 1960s**
J. B. Hagen
Stud. Hist. Phil. Biol. & Biomed. Sci., Vol. 32, No. 2(May. 23, 2001), p. 291–314

[12] **The Specular Reflection of X-rays**
W. L. Bragg
Nature, Volume 90, Issue 2250 (Dec. 12, 1912), p. 410

[13] **Protein bioinformatics an algorithmic approach to sequence and structure analysis**
I. Eidhammer, I. Johanssen and W. R. Taylor
John Wiley & Sons Ltd, 2004

[14] **Computational Methods for Mass Spectrometry Proteomics**
I. Eidhammer, K. Flikka, L. Martens and S.O. Mikalsen
John Wiley & Sons Ltd, 2007

[15] **Microbiology, fifth edition**
Prescott, Harley and Klein
McGraw-Hill Science/Engineering/Math; 5 edition (October 25, 2002)

[16] **Introduction to Proteomics: Tools for the New Biology**
D. C. Liebler
Human Press, 2002

[17] **Disease proteomics**
S. Hanash, Department of Pediatrics, University of Michigan
Nature, Vol. 422, Issue 6928(Mar. 13, 2003), p. 226-232

[18] **Proteomics: a new approach to the study of disease**
G. Chambers, L. Lawrie, P. Cash and G. I. Murray
Journal of Pathology, Vol. 192, Issue 3 (Jun., 2000), p. 280-288.

[19] **Biochemistry, fifth edition**
J. M. Berg, J.L. Tymoczko and L. Stryer
W. H. Freeman; Fifth Edition edition (February 15, 2002)

[20] **General Chemistry**
D. D. Ebbing and S. E. Gammon
Houghton Miffline Company, 1999

[21] **Eukaryotic signal transduction via histidine-aspartate phosphorelay**
P. Thomason and R. Kay
Journal of Cell Science, Vol 113, Issue 18(2000), p. 3141-3150

[22] **Signal transduction schemes of bacteria.**
J. S. Parkinson
Cell., Vol. 73, No. 5(1993), p. 857-871

[23] **Microarrays: biotechnology's discovery platform for functional genomics**
M. Schena, R. A. Heller, T. P. Theriault, K. Konrad, E. Lachenmeier and R. W. Davis
Trends Biotechnology,Vol. 16, Issue 7 (Jul. 1, 1998), p. 301-306

[24] **Measuring Mass: From Positive Rays to Proteins.**
M. A Grayson, ed. Philadelphia
Chemical Heritage Foundation (June 30, 2005)

[25] **From large analogical instruments to small digital black boxes: 40 years of progress in mass spectrometry and its role in proteomics. Part I 1965–1984.**
E. Gelp´
JOURNAL OF MASS SPECTROMETRY, J. Mass Spectrom. 2008; 43: 419–435

[26] **The ADAM metalloproteinases**
D. R. Edwards, M. M. Handsley, C. J. Pennington
Molecular Aspects of Medicine, Vol. 29, Issue 5(Aug. 7, 2008), p. 258–289

[27] **Mammalian Caspases: Structure, Activation, Substrates, and Functions During Apoptosis**
W. C. Earnshaw, L. M. Martins and S. H. Kaufmann
Annual Review of Biochemistry, Vol. 68, (Jul., 1999), p. 383-424

[28] **Mass Spectrometry in Proteomics**
R. Aebersold and D. Goodlett
Chemical Review, Vol. 101, (2001), p. 269-295

[29] **Jean Serra's publicly available course**
(http://cmm.ensmp.fr/~serra/cours/pdf/en/ch3en.pdf)

[30] **Normalization, Baseline correction and alignment of high-throughput data**
A.C. Sauve, T.P. Speed
Proceedings Gensips, 2004

[31] **A comprehensive approach to the analysis of matrix-assisted laser desorption/ionization-time of flight proteomics spectra from serum samples**
K.A. Baggerly, J.S. Morris, J. Wang, D. Gold, LC. Xiao and K.R. Coombes.
Proteomics, Vol. 3, Issue 9 (Sep. 9, 2003), p. 1667-72.

[32] **Smoothing and Differentiation of Data by Simplified Least Squares Procedures,**
A. Savitzky and M.J.E. Golay
(http://pubs.acs.org/cgi-bin/archive.cgi/ancham/1964/36/i08/pdf/ac60214a047.pdf)

[33] **Mass spectrometry-based proteomics**
R. Aebersold and M. Mann
Nature, Vol. 422, Issue 6928 (Mar. 13, 2003), p. 198-207.

[34] **Preprocessing of tandem mass spectrometric data to support automatic protein identification**
M. Gentzel, T. Köcher, S. Ponnusamy and M. Wilm
Proteomics, Vol. 3, Issue 8 (Aug. 12, 2003), p. 1597–1610

[35] **Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching.**
P. Du , W.A. Kibbe and S.M. Lin
Bioinformatics, Vol. 22, Issue 17(Jul. 4, 2006), p. 2059-2065

[36] **Improving protein identification from peptide mass fingerprinting through a parameterized multi-level scoring algorithm and an optimized peak detection**
R. Gras, M. Müller, E. Gasteiger, S. Gay, P.A. Binz, W. Bienvenu, C. Hoogland, J.C. Sanchez , A. Bairoch, D.F. Hochstrasser, R.D. Appel
Electrophoresis, Vol. 20, Issue 18 (Dec. 8, 1999), p. 3535

[37] **Quality Assessment of Tandem Mass Spectra Based on Cumulative Intensity Normalization**
S. Na and E. Paek
Journal of proteome research, Vol. 5, No,12(2006), p. 3241-3248

[38] **Comparison of Methods for Detecting and Deisotoping Weak, High Charge Signals in Data**
M.R.Alecio, A.G.Ferrige, L.Pannell and R.S.Ray
(http://positiveprobability.com/POSTERS/2006Deiso.pdf)

[39] **Calculation of the Isotope Cluster for Polypeptides by Probability Grouping**
M. T Olson and A. L. Yergey
Article in press.

[40] **Stable isotope labeling by amino acids in cell culture, SILAC, as a simple and accurate approach to expression proteomics**
S. Ong, B. Blagoev, I. Kratchmarova, D.B. Kristensen, H. Steen, A. Pandey and M. Mann
Molecular Cell Proteomics, Vol. 1, Issue 5(May. 20,2002), p. 376-86.

[41] **Proteome analysis using selective incorporation of isotopically labeled amino acids**
T. D. Veenstra, S. Martinovi, G. A. Anderson, L. Paa-Toli and R. D. Smith
Journal of American Society for Mass Spectrometry, Vol. 11, Issue 1 (Sep. 24, 1999), p. 78-82.

[42] **Quantitative analysis of complex protein mixtures using isotope-coded affinity tags**
S. P. Gygi, B. Rist, S. A. Gerber, F. Turecek, M. H. Gelb and R. Aebersold
Nature Biotechnology, Vol. 17, Issue 10 (Aug. 2, 1999), p. 994-999

[43] **Search for Cancer Markers from Endometrial Tissues Using Differentially Labeled Tags iTRAQ and cICAT with Multidimensional Liquid Chromatography and Tandem Mass Spectrometry**
L. DeSouza, G. Diehl, M.J. Rodrigues, J. Guo, A.D. Romaschin, T.J. Colgan and K.W. M. Siu
Journal of Proteomic Research, Vol. 4, Issue 2 (Mar. 12, 2005), p. 377-86

[44] **Stable isotope methods for high-precision proteomics**
L. V. Schneider and M.P. Hall
Drug Discovery Today, Volume 10, Issue 5 (Mar. 1, 2005), p. 353-363

[45] **Comparative LC-MS: A landscape of peaks and valleys**
A.H. P. America and J.H.G. Cordewener
Proteomics, Vol. 8, Issue 4 (Feb. 22, 2008), p. 731-49

[46] **Critical assessment of alignment procedures for LC-MS proteomics and metabolomics measurements**
E. Lange, R. Tautenhahn,S. Neumann and C. Gröpl
BMC Bioinformatics, (Sep. 15, 2008)
(http://www.biomedcentral.com/1471-2105/9/375)

[47] **OpenMS – An open-source software framework for mass spectrometry**
Reviewed by M. Sturm, A. Bertsch, C. Gröpl, A. Hildebrandt, R. Hussong, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, A. Zerck, K. Reinert and O. Kohlbacher
BMC Bioinformatics, (Mar. 26, 2008)
(http://www.biomedcentral.com/1471-2105/9/163)

[48] **MassLynx**
(http://www.waters.com/waters/nav.htm?cid=513164&locale=en_US)

[49] **Object Oriented Software Engineering, using UML, Patterns and Java**
B. Bruegge and A.H. Dutoit
Pearson Education Inc., Pearson Hall, 2004

[50] Very little known, does not have its own homepage. Are updated regulary on sourceforge(free program site) though.
(http://tools.proteomecenter.org/wiki/index.php?title=Software:massWolf)

[51] **Online XML tutorial**
(http://www.w3schools.com/xml/)

[52] **Programmering i Java**
E. Lervik and V.B. Havdal
Fofatterene, Stiftelsen TISIP og Gyldendal Norsk Forlag 2003

[53] **JVM:**
(http://java.sun.com/products/hotspot/docs/whitepaper/Java_HSpot_WP_v1.4_802_3.html)
Thread synchronization:
(http://java.sun.com/javase/technologies/hotspot/vmoptions.jsp)

Appendix

Appendix A: Amino Acids with Side Chains and Characteristics

Name	Symbol	Side Chain	Residue Mass		pI values	pK values	Side chain polarity	Occurrence (%)
			Monoisotopic	Average				
Alanine	A, Ala	CH3-	71.04	71.08	6.01		nonpolar	7.87%
Arginine	R, Arg	HN=C(NH2)-NH-(CH2)3-	156.10	156.19	10.76	12.0	polar	5.42%
Asparagine	N, Asn	H2N-CO-CH2-	114.04	114.10	5.41		polar	4.13%
Aspartic acid	D, Asp	HOOC-CH2-	115.03	115.09	2.77	4.4	polar	5.34%
Cysteine	C, Cys	HS-CH2-	103.01	103.15	5.07	8.5	nonpolar	1.50%
Glutamine	Q, Gln	H2N-CO-(CH2)2-	128.06	128.13	5.65		polar	3.96%
Glutamic acid	E, Glu	HOOC-(CH2)2-	129.04	129.12	3.22	4.4	polar	6.66%
Glycine	G, Gly	H-	57.02	57.05	5.97		nonpolar	6.95%
Histidine	H, His	N=CH-NH-CH=C-CH2-	137.06	137.14	7.59	6.5	polar	2.29%
Isoleucine	I, Ile	CH3-CH2-CH(CH3)-	113.08	113.16	6.02		nonpolar	5.91%
Leucine	L, Leu	(CH3)2-CH-CH2-	113.08	113.16	5.98		nonpolar	9.65%
Lysine	K, Lys	H2N-(CH2)4-	128.10	128.17	9.74	10.0	polar	5.92%
Methionine	M, Met	CH3-S-(CH2)2-	131.40	131.20	5.74		nonpolar	2.39%
Phenylalanine	F, Phe	Phenyl-CH2-	147.07	147.18	5.48		nonpolar	3.95%
Proline	P, Pro	-N-(CH2)3-CH-CH2-	97.05	97.12	6.48		nonpolar	4.82%
Serine	S, Ser	HO-CH2-	87.03	87.08	5.68		polar	6.84%
Threonine	T, Thr	CH3-CH(OH)-	101.05	101.11	5.87		polar	5.41%
Tryptophan	W, Trp	Phenyl-NH-CH=C-CH2-	186.08	186.21	5.89		nonpolar	1.13%
Tyrosine	Y, Tyr	4-OH-Phenyl-CH2-	163.06	163.18	5.66	10.0	polar	3.02%
Valine	V, Val	CH3-CH(CH2)-	99.07	99.13	5.97		nonpolar	6.73%

Appendix B: Kyte and Doolittle Hydrophobicity Index

A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V
1.8	-4.5	-3.5	-3.5	2.5	-3.5	-3.5	-0.4	-3.2	4.5	3.8	-3.9	1.9	2.8	-1.6	-0.8	-0.7	-0.9	-1.3	4.2

The larger the number is, the more hydrophobic the amino acid is.

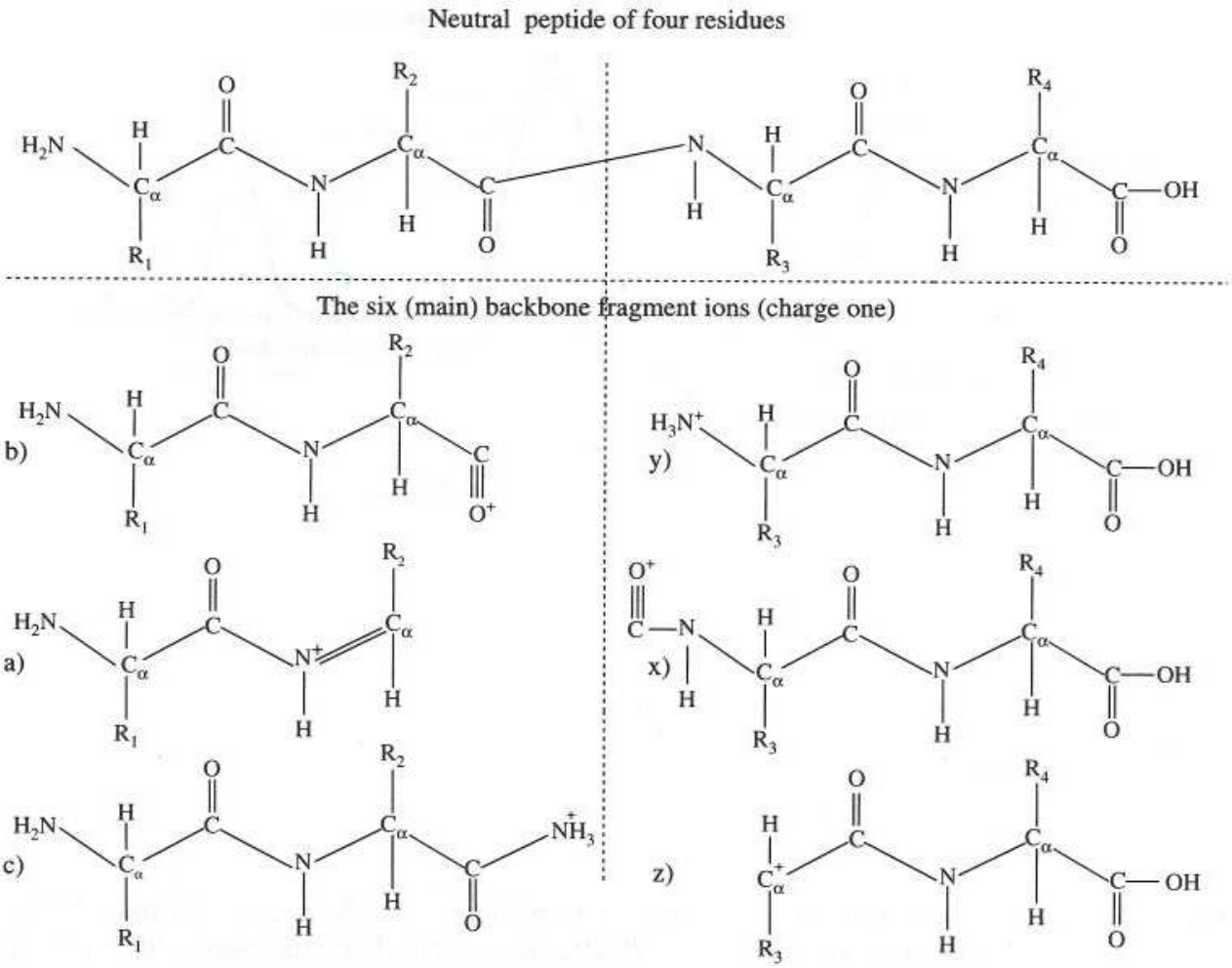
Appendix C: Database Classification of Secondary Structures

The table and it's text is taken from reference [13] . It clearly shows that the four different programs classify secondary structures differently(Table 14.2).

Table 14.2 A comparison between CATH, SCOP and DaliDD at class level of 3 October 2002. The numbers of different folds (topologies) inside different classes are shown. The numbers of different architectures are also shown for CATH.

CATH			SCOP		DaliDD	
Mainly α	(5)	219	All α	151	All α	257
Mainly β	(119)	133	All β	110	All β	69
α β	(13)	245	α and β (a/b)	113	α/β	97
Few SSs	(1)	77	α and β (a+b)	208	α-β meander	19
			Multi-dom. (α and β)	34	Antipar. β	28
			Membr. and cell surf.	12	Ambiguous	208
			Small	50	Unclassified	405
			Coiled coil	4		
			Low resolution	8		
			Peptides	63		
			Designed proteins	17		

Appendix D: Fragmentation Profile in MS/MS



This figure shows the six main backbone fragment types a neutral peptide when fragmented with CID and have a charge of 1. Fragment type a, b and c on the left and its complimentary on the right(x, y and z).

Appendix E: How to Use Databridge

Databridge is a small program bundled with MassLynx. This program lets you convert other formats into MassLynx format, as well as converting MassLynx format into ASCI or NetCDF.

To start Databridge go into the MassLynx program folder on the start menu. Here the icon of the program will be showed(it's a black exclamation mark in a yellow hexagon). This take you to the selection screen, shown in figure A.1.

Figure A.1: The selection screen in Databridge.

On the left side(Source) of figure A.1 a 'Select' button is shown. Here you can select what file you want to convert. The right side lets you choose where to place and what the name of the converted file should be. The 'Options' button lets you choose the type the source file is, and what type the target file should be. The 'Options' panel is shown in figurer A.2.

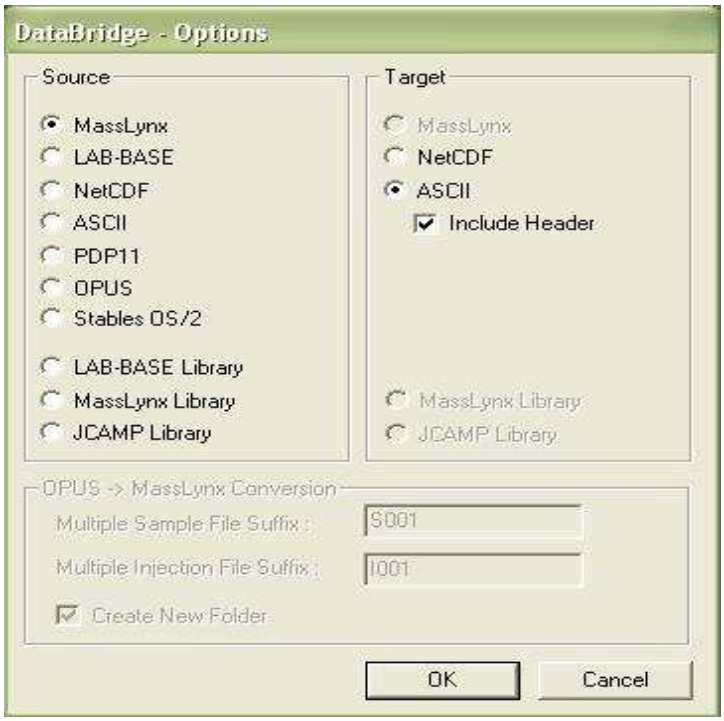


Figure A.2: Shows the format to be converted from(source) and the format converted to(target).

The convert button starts the conversion of the file, and will show a progression bar of how far it has gotten.

For the 'MassAnalyzer' program follow the recipe underneath:

1. Start Databridge
2. Press 'Options' button
3. Choose MassLynx as source
4. Choose ASCI as target
5. Check 'include header' check box
6. Press 'OK' button
7. Press 'Select' button and choose the .raw file that you want to converting
8. Back in the select window, write the target name and select directory it will be placed by either typing it or press 'Directory' button
9. Press 'Convert'
10. Wait circa 10min for conversion.

Appendix F: Control Set, GDS_300707_THP1 -100_07

Sequence is the peptide sequence from Mascot, M/Z is the mass to charge ratio, Charge is the chrage of the isotopic envelope of the peptide, Modifications is what kinf of modification the peptide have, Retention time is the first retention time this peptide was found at and 2/1 is the ratio between the heavy and light isotope.

Sequence	M/Z	Charge	Modifications	Retention time	2/1
DLYANTVLSGGTTMYPGIADRMQK	870,74	3	1Lys8	44,79	0,86
DLYANTVLSGGTTMYPGIADRMQK	868,06	3	1Lys8	44,73	0,86
EITALAPSTMK	581,27	2		26,59	0,86
EITALAPSTMK	585,28	2	1Lys8	26,62	0,86
AIADTGANVVVTGGK	690,87	2	1Lys8	22,61	0,83
AIADTGANVVVTGGK	686,86	2		22,58	0,83
NFINNPLAQADWAAK	840,91	2	1Lys8	46,78	0,91
NFINNPLAQADWAAK	836,89	2		46,85	0,91
NQVAMNPTNTVFDAK	825,37	2		30,24	0,85
NQVAMNPTNTVFDAK	829,37	2	1Lys8	30,35	0,85
DTNGSQFFITTVK	733,35	2	1Lys8	38,19	0,6
DTNGSQFFITTVK	729,34	2		38,25	0,6
SQIFSTASDNQPTVTIK	918,95	2		30,86	0,61
SQIFSTASDNQPTVTIK	922,95	2	1Lys8	30,89	0,61
ADLINNLGTIAK	625,83	2	1Lys8	37,22	0,85
ADLINNLGTIAK	621,82	2		37,28	0,85
EDVFVHQTAIK	647,82	2	1Lys8	23,79	0,86
EDVFVHQTAIK	643,8	2		23,76	0,86
HGFLEEFITPIVK	769,41	2	1Lys8	54,91	0,85
DVNAAIATIK	508,28	2		27,35	0,92
VGINYQPPTVVPGGDLAK	916,98	2	1Lys8	36,31	0,92
HGYIGEFEIIDDHRAGK	978,95	2		30,18	0,83
HGYIGEFEIIDDHRAGK	655,63	3	1Lys8	30,12	0,83
HGYIGEFEIIDDHRAGK	652,97	3		30,05	0,83
HGYIGEFEIIDDHRAGK	489,97	4		30,42	0,83
SEHPGLSIGDTAK	656,29	2		20,29	0,88
SEHPGLSIGDTAK	660,3	2	1Lys8	20,32	0,88
SEHPGLSIGDTAK	437,88	3		20,35	0,88
GEHPGLSIGDVAK	640,31	2		24,56	0,91
LHNMIVDLDNVVK	759,39	2	1Lys8	41,43	0,74
LHNMIVDLDNVVK	755,39	2		41,31	0,74
AVQQPDGLAVLGIFLK	838,97	2	1Lys8	65,61	0,84
AVQQPDGLAVLGIFLK	834,97	2		65,67	0,84
NGIHDLDNISFPK	735,36	2		35,42	0,79
NGIHDLDNISFPK	739,35	2	1Lys8	35,54	0,79
SDHPGISITDLSK	685,33	2		28,23	0,8
SDHPGISITDLSK	689,34	2	1Lys8	28,26	0,8
DHENIVIAK	523,78	2	1Lys8	18,26	0,85
DHENIVIAK	519,77	2		18,2	0,85
DSPSVWAAVPGK	607,29	2		31,42	0,95
DSPSVWAAVPGK	611,3	2	1Lys8	31,45	0,95
MFIGGLSWDTTK	682,33	2	1Lys8	47,85	0,64
AALTRDPQFQK	662,84	2	1N-ac 1Lys8	26,38	0,78
AALTRDPQFQK	658,83	2	1N-ac	26,35	0,78
HRDFVAEPMGEK	708,32	2		19,58	0,82
HRDFVAEPMGEK	712,33	2	1Lys8	19,61	0,82
HRDFVAEPMGEK	472,54	3		19,65	0,82

Sequence	M/Z	Charge	Modifications	Retention time	2/1
YQLDPTASISAK	647,32	2		27,59	0,82
YQLDPTASISAK	651,33	2	1Lys8	27,76	0,82
NPILWNVADVVIK	740,9	2		59,15	0,68
RKRELDVEEAHAASTEEK	700,01	3		15,15	0,97
RKRELDVEEAHAASTEEK	705,35	3	2Lys8	15,18	0,97
RKRELDVEEAHAASTEEK	525,26	4		15,21	0,97
RAGELTEDEVERVITIMQNPRQYK	721,6	4	1Lys8	49,94	0,75
RAGELTEDEVERVITIMQNPRQYK	719,6	4		50	0,75
TLHPDLGTDK	548,77	2		18,06	0,78
TLHPDLGTDK	552,77	2	1Lys8	18	0,78
HGGPKDEERHVGDLGNVTADK	749,69	3	2Lys8	17,76	0,88
HGGPKDEERHVGDLGNVTADK	744,34	3		17,73	0,88
HGGPKDEERHVGDLGNVTADK	558,51	4		17,79	0,88
HVTVIGGGLMGAGIAQVAAATGHTVVLVDQ	903,97	4		67,01	0,88
HVTVIGGGLMGAGIAQVAAATGHTVVLVDQ	905,98	4	1Lys8	67,04	0,88
HQNVQLPREGQEDQGLTK	1039,01	2		19,94	0,94
HQNVQLPREGQEDQGLTK	695,65	3	1Lys8	20,05	0,94
HQNVQLPREGQEDQGLTK	692,98	3		19,91	0,94
HQNVQLPREGQEDQGLTK	521,99	4	1Lys8	20,08	0,94
NGQDLGVAFK	524,75	2		26,65	0,91
EINNAHAILTDATK	755,88	2		25,2	0,86
EINNAHAILTDATK	759,89	2	1Lys8	25,24	0,86