Università degli Studi di Padova
Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale

Protein Identification from Top-Down Mass Spectra, a Fast Filtering Algorithm

Laureando
Alessandro Mammana

Relatore
Alberto Apostolico

Anno Accademico 2010-2011

Contents

Chapter 1

Introduction

Proteomics deals with the large-scale determination of gene and cellular function directly at the protein level. Scientists working in the field often need to know which proteins are present in an organism given point and sometimes in what amounts. Identifying proteins contained in a sample is therefore a vital need for proteomics. Mass spectrometry has increasingly become the method of choice for the analysis of complex protein samples. It has been made possible by the availability of gene and genome sequence databases and technical and conceptual advances in many areas, most notably the discovery and development of protein ionization methods.

Mass spectrometry (MS) is an analytical technique that measures the mass-to-charge ratio ($\frac{m}{z}$) of charged particles. More precisely, the output of a mass spectrometer is a set of peaks, each having an intensity and a position: the intensity depends on the number of detected ions sharing the same $\frac{m}{z}$ value, the position is the shared $\frac{m}{z}$ value. This set of peaks is the raw mass spectrum of the sample.

In proteomics studies, the sample is often derived from a mixture of proteins after a process that may include separation, ionization, proteolytic digestion and fragmentation, and it consists of many different fragments of identical proteins. The obtained spectra serve as fingerprints, that can be more or less specific, depending on factors like instrumentation accuracy and mixture composition. Protein identification generally consists in the determination of the peptide sequence the spectrum belongs to, but the goal can be more or less ambitious. Sometimes the goal is to find just one of the protein isoforms present in a database and sometimes not only the protein exact sequence, but also its post-translational modifications (PTM) have to be determined. There are two main approaches to protein identification through mass spectrometry: the bottom-up and and top-down approach.

1.1 The different approaches

1.1.1 Bottom-up proteomics

In bottom-up mass spectrometry, the proteins in the mixture undergo proteolytic digestion before the introduction into the mass spectrometer. Proteolytic digestion is carried out by proteolytic enzymes, such as trypsin, and consists in the cleavage of the whole protein at specific sites known in advance (specificity of the enzyme). Then the sample is ionized, sometimes after a separation stage, and finally analyzed by the spectrometer. This is what characterizes bottom-up proteomics: that the analytes are cleaved peptides. There are two commonly used methodologies: peptide mass fingerprinting and tandem mass spectrometry (MS/MS or MS^2).

In peptide mass fingerprinting, peptide masses obtained from the MS scan are compared to calculated peptide masses. These masses are obtained by an *in silico* cleavage of proteins or gene sequences in a database using the appropriate specificity.

In tandem mass spectrometry, a peptide ion is isolated in the mass analyzer and subjected to dissociation to produce product ion fragments. These fragments are then analyzed by a second mass spectrometric measurement. With the product ion spectra, different methods can be applied: a *de novo* analysis of the whole protein, a *de novo* reconstruction of small peptide sequences that serve as tags for a database search, or a database search based on a cross-correlation analysis.

1.1.2 Top-down proteomics

In top-down proteomics, intact protein molecular ions generated by electrospray ionization are introduced into the mass analyzer and are subjected to gas-phase fragmentation for MS analysis. Therefore its difference from the bottom-up approach resides in that charged fragments of the whole protein are analyzed instead of proteolytic digests, and they are usually much larger (10^4 Daltons is a typical parent mass for a top-down spectrum).

As in the bottom-up approach, different methods can be applied to attempt to identify proteins from spectra. These methods will be discussed later.

1.1.3 Advantages and disadvantages

Bottom-up proteomics is the most mature and most widely used approach for protein identification, but it still suffers from some problems. Most importantly, only a fraction of the total peptide population of a given protein is identified. Therefore, information on only a portion of the protein sequence is obtained. This limited sequence coverage might be sufficient for the identification of one of the protein isoforms from a database, but it is deficient when it comes to characterize the exact protein form and to identify possible post-translational modifications.

The two major advantages of the top-down strategy are the potential access to the complete protein sequence and the ability to locate and characterize PTMs. In addition, the time-consuming protein digestion required for bottom-up methods is eliminated. Despite these advantages, top-down proteomics is a relatively young field compared to bottom-up proteomics, and currently suffers from several limitations. One of them is that bioinformatic tools for top-down proteomics are still primitive.

A third approach known as middle-down is becoming more and more important. In middle-down mass spectrometry proteins are digested with enzymes that have a low specificity, i.e., that split proteins into large fragments. This last method has characteristics in between top-down and bottom-up mass spectrometry, but it is closer to the latter. For a thorough discussion of these topics the reader is referenced to some reviews [5].

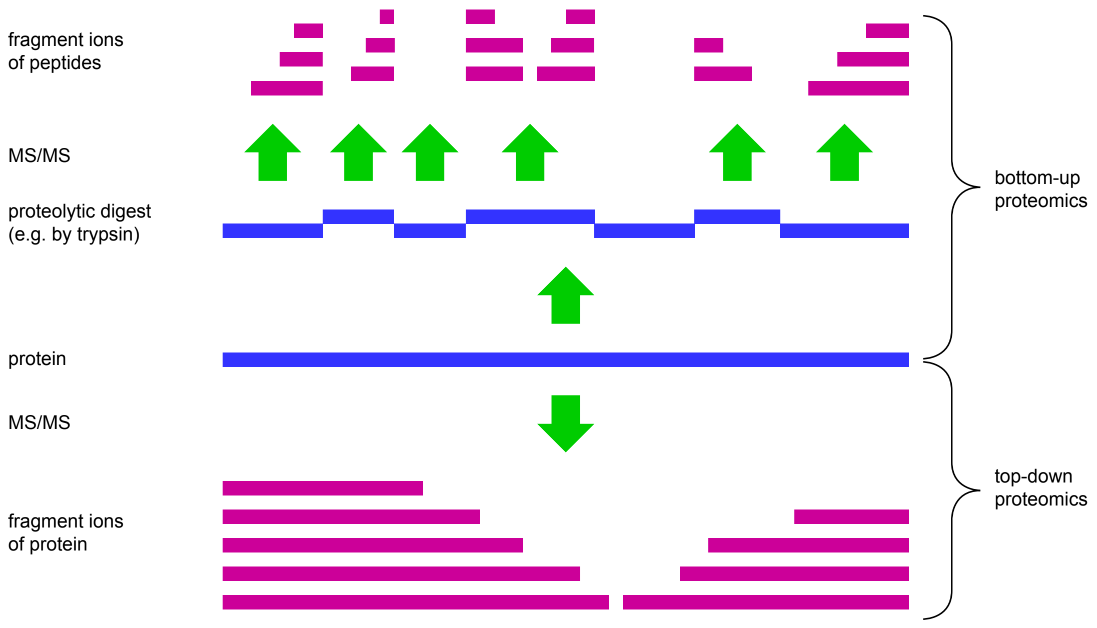


Figure 1.1: The two main approaches to mass spectrometry.

1.2 Top-down mass spectrometry

Top-down mass spectrometry is a rapidly evolving field. In the last two years, due to advances in protein separation and top-down instrumentation, top-down mass spectrometry moved from analyzing single proteins to analyzing complex samples containing hundreds or even thousands of proteins. Such experiments produce a big amount of spectra with a high degree of complexity. To deal with this amount of complex data, fast and accurate bioinformatic tools are necessary. Since algorithms for interpreting top-down spectra are still in infancy, many recent developments include computational innovations in protein identification. Some available tools for protein identification will be presented. The most basic notions about top-down mass spectrometry, such as b-ions, y-ions, raw spectra and deconvoluted spectra, are presented in the Appendix.

1.2.1 Tools for protein identification through database search

Algorithmic tools can attempt to interpret a spectrum in two ways, respectively database search and *de novo* sequencing[1]. The latter has the advantage that completely unknown proteins can be identified, but, on the other hand, it is usually slower and less powerful than database search. Moreover, because top-down spectra are more complex than bottom-up spectra, *de novo* sequencing is not as effective as in the bottom-up approach. Database search tools are in general faster and more powerful, when the protein is in the database. This last requirement might be not so stringent for two reason. First, proteins that spectra come from sometimes belong to organisms of which the complete proteome, or most of it, is already known. Second, spectra can be matched with proteomes of very similar organisms, because it is very likely that the unknown protein, or at least a close variant, is shared by the two organisms.

The following is a list of the most important tools for protein identification based on database search.

- **ProSightPTM** ProSightPTM is the most commonly used tool for protein identification from top-down spectra (see [9] for more details). The algorithm searches spectra against a "shotgun annotated" protein database: an extended database that contains not only a given set of proteins, but also every expected modified version of the same protein. The extended database is much larger than the original one. Being a restrictive PTM search, that takes into account a fixed set of PTMs and ignore all the others, it may

[1]Also hybrid approaches are possible. In fact tools that build tags from the spectrum and use these tags to filter a database can be considered a hybrid approach.

fail to identify protein forms which are not included in the shotgun annotated database due to unexpected modifications. This tool also provides an estimation of the statistical significance of the results.

- **PIITA** PIITA is a recently developed algorithm [16] that performs a restrictive PTM search. In contrast with ProSightPTM, PIITA is a precursor independent method: it compares all theoretical fragmented ions of a database protein (b-, y-, c-,z- ions) with the peaks in the spectrum. It is thus capable of identifying protein species with unexpected modifications at the N-terminus or C-terminus, but not species modified on both N- and C-termini.

- **US-Tags** The authors [15] developed a tag-based approach. First, long sequence tags (6 amino acids or longer) are generated from the spectrum, then these tags are matched against the database with very fast pattern matching algorithms and finally, when a match occurs, the entire protein is annotated. This approach, while powerful, relies on long sequence tags that may be difficult to obtain for some spectra.

- **MSTopDown** In [4] an algorithm based on spectral alignment is proposed. This algorithm allows researchers to match top-down spectra to proteins with unexpected modifications, i.e., without knowing which modifications are present in the sample. The disadvantage of this approach is that it is rather slow in practice, and it does not provide a statistical analysis of the results.

- **Mascot and SEQUEST** These tools commonly known for bottom-up protein identification have been adapted to top-down mass spectrometry.

1.2.2 MS-Align+, a brief overview

MS-Align+ is a tool that is being developed in these years at UC San Diego and that will be shortly publicly available [10]. It shares the spectral alignment approach with MSTopDown, but greatly improves on speed and statistical analysis of the results. A particularly powerful and unique feature of this algorithm is that it can run in **blind mode**. In blind mode the algorithm can interpret spectra that come from proteins with post-translational modifications, ranging from amino-acid substitutions, to insertions and deletions. This is a challenging task for the program, because in blind mode proteins with a mass different from the spectrum parent mass cannot be excluded as in the other approaches. The algorithm performs the following steps.

1. **Spectrum deconvolution** This step is in common with the vast majority of the protein identification tools. MS-Deconv [11], is the method of choice for this step.

2. **Database filtering** As mentioned above the candidate proteins in the database cannot be filtered out based solely on their masses, or the masses of prefixes, suffixes and internal fragments of the protein. But at the same time running the whole algorithm for every protein in the database would make the overall execution prohibitively slow, as it will be shown below. Therefore MS-Align+ needs a sophisticated and fast filtering algorithm that reduces the database size.

3. **Spectral alignment algorithm** This algorithm is at the heart of MS-Align+ power in recognizing post-translational modifications. It is the same algorithm used in MS-TopDown, with some improvements. It will be presented in the next chapter.

4. **Analysis of the statistical significance of the results** Another important routine calculates an E-Value and a P-Value for the protein-spectrum match. This is a big improvement relative to the previous approaches, where this step is absent or the analysis is not accurate. Accurate analysis can alternatively be obtained with the use of decoy database, but it is a very time-consuming procedure. The interest reader is pointed to the article by Kim [7].

1.2.3 The need for a filtering algorithm

The spectral alignment algorithm, while powerful, is rather slow. It can be tested running MS-TopDown, that uses a basic version of it and it is freely downloadable [6]. With typical parameters it takes about 1.5 seconds to align a single spectrum with 150 peaks against a single small protein of 70 amino acids (on a dual core laptop computer with two 2.5 Ghz CPUs). Aligning a small data set with 1000 spectra against a small proteome with 1000 proteins would then take about $1.5 \cdot 10^6$ seconds, i.e. 17 days. Such a time is, of course, too long. A filtering algorithm that reduces the number of candidate proteins is essential for MS-Align+ to be practical. It will be shown that the full power of the spectral alignment algorithm is not needed for every protein. A fast filtering algorithm that filters out proteins safely and quickly will be presented.

Chapter 2

Improving database filtering

The purpose of this chapter is to present MS-Filter: the algorithm designed as a filtering stage for MS-Align+. The structure of this presentation follows the steps done by this author in the development of the final program. After some preliminaries, the first topic is the spectral alignment algorithm. This is the core algorithm in MS-Align+ and understanding how it works provides useful insights for the design of a filtering stage. Next, a simple but powerful filtering approach, called from some authors the spectral convolution [14], is introduced, along with some simple algorithms based on it and some theoretical considerations. This is the most important part of this chapter, because the current implementation of MS-Align+ is based on the spectral convolution and because MS-Filter is a variant of it. Finally MS-Filter is presented in detail, but the presentation is not self-contained, it deeply relies on the material introduced in the preceding section. The performance of the new algorithm, instead, is the subject of the next chapter.

2.1 Preliminaries

It is worth to present here some notions that occur often throughout this work. The notions presented in the Appendix are required to understand this material.

2.1.1 Protein and spectrum representation

A deconvoluted mass spectrum S is characterized by a parent mass M_a and set of peaks $\tilde{A}$, each having an intensity and a position. In this work, when there is no ambiguity, a deconvoluted mass spectrum will be called simply spectrum. Every peak can be originated by either a b-ion or a y-ion, but for protein identification it is useful to work with only one ion type. Unfortunately, there is no easy way to tell apart the different ion types in the mass spectrum. However, if a peak was originated by a b-ion and its position is p, then a peak originated by the corresponding y-ion must have position $M_a - p$ and, consequently, if a peak with position p is originated by a y-ion the peak generated by the corresponding b-ion must have position $M_a - p$. To make sure that every possible peak coming from a b-ion is present in the considered set of peaks, a new set of peaks A is built from $\tilde{A}$ adding, for each peak at an arbitrary position p, a new peak with the same intensity and position $M_a - p$. The **representation of the spectrum** S is the sequence $[a_i]_{i=1}^{n}$ of the peak positions in A sorted from the smallest to the largest value. In a computer program a sequence of masses is stored as an array of floating point numbers, therefore sometimes the sequence is denoted with the array $a[1:n]$.

A protein is usually represented by an amino acid sequence, but, in the algorithms that are going to be presented, what really matters is just the monoisotopic mass of each residue[1]. Let the protein P be a sequence of m amino acids and let $[P]_j$ denote the set composed by the first j elements. The **theoretical spectrum** of the protein P, also called the **representation of the protein**, is a sequence of m masses $[b_j]_{j=1}^{m}$ such that b_j is the sum of the monoisotopic residue masses of every amino acid in $[P]_j$. It follows that b_1 is the monoisotopic residue mass of the first amino acid, that b_m is the mass of the whole protein minus about 18 Daltons, and that the sequence b is sorted from the smallest to the largest value. As for the spectrum, sometimes the sequence is denoted with the array $b[1:m]$.

2.1.2 Database search of mass spectra

The goal of a computer program for the database search of mass spectra is to find from a spectrum S and a database D the protein in the database that might have originated that spectrum. When the protein is known we say that the protein has been **identified** and that the protein is the **interpretation** of the spectrum. However, even when the protein is known, there are several ways to explain how each peak in S derive from a protein fragment, especially when there are post-translational modifications. A **protein-spectrum alignment** between a protein P and a spectrum S is a specific way of interpreting each peak in S in terms of protein fragments or noise. The goal of a database search tool for top-down mass spectra is, therefore, to determine, from a given spectrum and a database, the protein most likely to be the interpretation of the spectrum, to specify the protein-spectrum alignment and, optionally, to give a measure of the likelihood that the interpretation is correct.

[1]However, other algorithms try to derive from the spectrum a subsequence of amino acids that with high probability belong to the correct protein, called tags, then use pattern matching algorithms to match tags with protein sequences. See [15] for such an approach.

2.1.3 Database filtering

A **database filtering algorithm** for database search of top-down spectra is a tool that, given a spectrum S and a database D, returns a restricted database D'. The proteins in D' should be the most likely to be the spectrum correct interpretation. This tool, if used to preprocess the database before running a database search tool, should reduce the running time and never filter out the correct protein.

2.2 Spectral alignment algorithm

The algorithm that MS-Align+ is based on was developed by Pevzner and it is called the spectral alignment algorithm. The interested reader can find a detailed treatment in the article by Pevzner [14]. Here we will show a simplified version of it.

Let $a[1:n]$ and $b[1:m]$ be the representation respectively of a spectrum S and a protein P. We call **spectral grid** the set of points in the euclidean plane with coordinates $(b_j, -a_i)$ for $0 \leq i \leq n$ and $0 \leq j \leq m$. A **spectral alignment** is a particular path in the spectral grid, i.e., a path composed of segments with both ends in the grid's points and with some other properties. Formally, a spectral alignment is a sequence of segments $L_1, L_2, \ldots L_r$ where $L_k = ((b_{j_{k-1}}, -a_{i_{k-1}}), (b_{j_k}, -a_{i_k}))$ and the indexes i_k and j_k are strictly increasing with k, starting at zero and less than, respectively n and m. A segment L_k of a spectral alignment can be of three kinds.

1. If $b_{j_{k-1}} - a_{i_{k-1}} = b_{j_k} - a_{i_k}$, then L_k is called a diagonal segment. The biological interpretation is that, if this segment belongs to the correct alignment, the ions with masses $a_{i_{k-1}}$ and a_{i_k} are the prefixes of the protein P formed by respectively the first j_{k-1} and the first j_k residues. In practice the exact equality is not required, but the distance between the two values must be very small. In that case we say that (i_{k-1}, j_{k-1}) and (i_k, j_k) are **codiagonal**.

2. If $b_{j_{k-1}} - a_{i_{k-1}} < b_{j_k} - a_{i_k}$, then L_k represents a modification. In fact it suggests that there has been a modification between $[P]_{j_{k-1}}$ and $[P]_{j_k}$ causing a mass reduction of the peptide sequence between the first prefix and the second, such as a deletion.

3. If $b_{j_{k-1}} - a_{i_{k-1}} > b_{j_k} - a_{i_k}$, then L_k represents a modification as well. In fact it gives evidence of a modification occurring between $[P]_{j_{k-1}}$ and $[P]_{j_k}$ causing an increase of the protein mass, such as a phosphorylation.

The diagonal score of a spectral alignment is the number of segments it is made up of. The number of modifications in the path is the number of segments that are not diagonal. The goal of the algorithm is to find the maximum score of a spectral alignment with up to F modifications, where F is a fixed number.

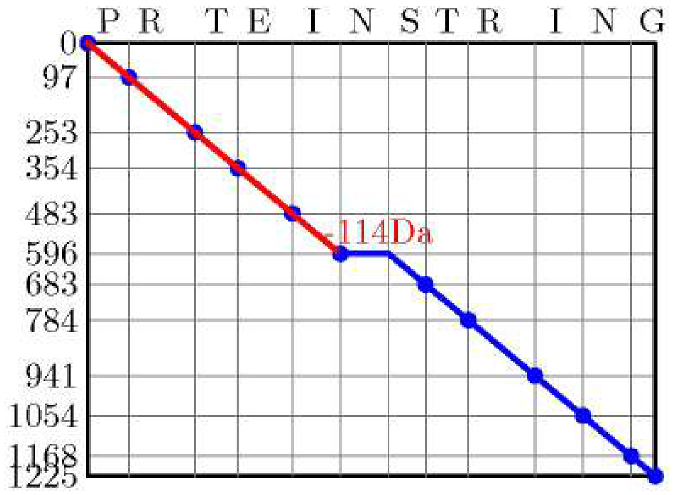


Figure 2.1: Example of a spectral alignment. The spectral alignment is made up of only diagonal segments excepted the line in the middle, that, according to the terminology just introduced, is a horizontal segment. It is drawn as two segments and it suggests the deletion of the residue N.

This problem can be solved in time $\Theta(nmF)$ with the following dynamic programming algorithm, called the spectral alignment algorithm. Let $D_{ij}(k)$ denote the highest score of a spectral alignment that ends in $(b_j, -a_i)$ with exactly k modifications, and assign to every position $D_{i0}(k)$ and $D_{0j}(k)$ the score 0. If such a score is computed for every i, j, k where $0 \leq i \leq n$, $0 \leq j \leq m$ and $0 \leq k \leq F$, the score of the optimal spectral alignment can be determined. Finally, as it is customary with dynamic programming algorithms, the exact alignment can be obtained by maintaining pointers associated to every $D_{ij}(k)$ score and by backtracking from the highest score.

The recurrence relation for $D_{ij}(k)$ exploited by the algorithm is

$$D_{ij}(k) = \max_{(i',j')<(i,j)} \begin{cases} D_{i'j'}(k) + 1 & \text{if } (i',j') \text{ and } (i,j) \text{ are codiagonal} \\ D_{i'j'}(k-1) + 1 & \text{otherwise.} \end{cases}$$

Here $(i', j') < (i, j)$ means that $i' < i$ and $j' < j$. This formula, although correct, is not practical because too many (i', j') values have to be considered for every update. Therefore the algorithm uses and updates some auxiliary data structures that make the update possible in constant time. These details will not be presented here. The resulting algorithm has a $\Theta(nmF)$ time and space complexity and solves the spectral alignment problem efficiently.

There are some details that have been omitted for the sake of simplicity. For instance, the spectrum does not only contain masses corresponding to prefixes of the protein (b-ions), but also masses corresponding to suffixes (y-ions). Peaks intensities can play a role, and the best scoring path is not free to end anywhere in the spectral grid. Moreover, modifications are not all identical. There are some very frequent modifications, such as phosphorylation or adenylation, that produce a known mass shift and that deserve a different treatment from a random mass shift. These factors are taken into account with simple modifications of the above algorithm and are not important for the work that will be shown here.

2.3 The spectral convolution approach

As before, let a protein P be represented by its theoretical spectrum (i.e., the spectrum with all the protein b-ions) $b_1 < \ldots < b_m$, and a spectrum S by the ordered sequence of its peaks $a_1 < a_2 \ldots < a_n$. The notion of a **diagonal** will be presented, along with its importance as a filtering criterion. Thereafter we will present a simple algorithm that detects diagonals through spectral convolution.

2.3.1 The diagonals

Definition A diagonal of length l with offset o is a pair of subsequences $[i_k]_{k=1}^{l}$ and $[j_k]_{k=1}^{l}$ such that $b_{j_k} - a_{i_k} = o$ for $k = 1, 2 \ldots l$[2].

Diagonals have a straightforward biological interpretation. In fact if there is a diagonal of length l at offset o, then, after shifting every mass in the spectrum by o, l peaks in the spectrum will coincide with l prefix masses in the protein. If S is the spectrum of an unmodified peptide P and if S and contains l true peaks (i.e. not noise peaks) there will be a diagonal of length l with offset 0 between S and P. Instead, if a spectrum comes from a peptide that has lost some initial residues with total mass o, there is a long diagonal with offset o. In general, if the protein the spectrum comes from has k internal post-translational modifications that differentiate it from protein P and if n spectrum peaks are true, there is a diagonal of length at least $\frac{n}{k}$ between P and S. Moreover, the cleavage of a peptide at the N-terminus or C-terminus, that are very frequent, do not count as internal modifications. Since we look for proteins that have no more than 2 or 3 modifications, and since the most frequent modifications consist in the cleavage of a N-terminal or C-terminal peptide sequence, a candidate protein must have

[2]The name *diagonal* derives from the terminology introduced when the spectral alignment algorithm was presented. It is also a natural choice if the relation is thought of in terms of a path in the spectral grid.

a long diagonal with a spectrum in order for it to be considered a significant protein-spectrum match. Diagonal detection, therefore, is a simple and effective strategy for database filtering in blind search mode. This is the principle that all the algorithms presented here build on.

2.3.2 Diagonals with errors

In practice, the exact equality will never be reached because of measurement errors[3], hence diagonals should be considered with a certain error tolerance. One way of dealing with errors is to use relative errors, measured in part per millions (ppm). In this case a diagonal of length l with offset o and with a maximum allowed error of five ppm (a reasonable value for the currently available technology) is a pair of subsequences $[i_k]_{k=1}^{l}$ and $[j_k]_{k=1}^{l}$ such that $|b_{j_k} - a_{i_k} - o| < 5\frac{a_{i_k}}{10^6}$ for $k = 1, 2 \ldots l$. Relative errors are important because mass spectrometers can determine peaks positions with an accuracy such that, the heavier is the ion, the bigger is the error. In fact the mass accuracy of a spectrometer, one of its most important parameters, is normally given in part per millions. However, when the mass accuracy is not too small, say five ppm, and when spectra parent masses are not too big, say 10^4 Daltons, we can use an absolute error that is half of the maximum allowed error, hence $0.5 \cdot 10^4 \cdot 5 \cdot 10^{-6} = 0.025$ Daltons. Such small absolute errors should not reduce the algorithm sensitivity too much and they are a lot easier to deal with. Throughout this work only absolute errors will been used. The definition of a diagonal has to be adapted accordingly.

Definition A diagonal of length l with offset o and with error tolerance e is a pair of subsequences $[i_k]_{k=1}^{l}$ and $[j_k]_{k=1}^{l}$ such that $|b_{j_k} - a_{i_k} - o| < e/2$ for $k = 1, 2 \ldots l$.

2.3.3 The spectral convolution

We are now in a position to give a rigorous formulation to the spectral convolution problem.

Spectral convolution problem. Given a spectrum S, a protein P and an error tolerance e find the longest diagonal between S and P with error tolerance e.

An algorithm that solves this problem can be easily turned into an algorithm for database filtering, in fact when the longest diagonal between P and S is known, its length is also known, and it can be used as a score for the protein-spectrum alignment. Assuming that this a good scoring function (assumption that has been already motivated in subsection 2.3.1 and that will be extensively pondered

[3]In order to be accurate enough, an algorithm needs to use finely discretized mass values, therefore the mass values must be represented as big integers, or floating point numbers.

later), the database can be filtered out scoring a spectrum against every protein and returning the best ones. Next, a simple way of addressing this problem is presented.

The existence of a diagonal of length l implies that for l distinct pairs of indexes (i, j) the l differences $b_j - a_i$ share approximately the same value, while all the other pairs should give $b_j - a_i$ values that are randomly spread out. The opposite is also true: l distinct pairs (i, j) such that the differences $b_j - a_i$ share approximately the same value are the distinctive footprint of a diagonal of length l. Therefore a straightforward way of detecting such a diagonal is to generate all nm differences $b_j - a_i$, store them into an array and check for l values close together. This strategy is at the heart of all the filtering algorithms that will be presented and it motivates the following definitions.

Definition The multiset $P \ominus S = \{b_j - a_i \text{ for } 0 < i \leq n,\ 0 < j \leq m\}$ is called the **spectral convolution** of P and S. The elements of this multiset will be called **convolution peaks**, or simply peaks when they are not likely to be confounded with the peaks of a spectrum. Finally $(P \ominus S)^e(x)$, called **the diagonal score function**, will denote the number of elements p in $P \ominus S$ such that $x \leq p < x + e$.

Using this new notation we can reformulate the spectral convolution problem as follows.

Spectral convolution problem. given a protein P, a spectrum S and a maximum tolerable error e, determine $\max\{(P \ominus S)^e(x)\}$.
It turns out that the spectral convolution problem can be addressed in several ways:

1. The naïve, exact algorithm,
2. the approximate algorithm,
3. Fourier transforms.

2.3.4 The naïve, exact algorithm

The first thing one might think of, is to produce all nm differences (convolution), store the floating point mass values in an array, sort the array and check for high concentration of peaks. This is the strategy of the following algorithm.

Claim 2.3.1 *Let $a[1:n]$ and $b[1:m]$ represent respectively a spectrum S and a protein P. The following pseudocode solves the spectral convolution problem and it can be implemented in time $O(nm \log(\min\{m, n\}))$.*

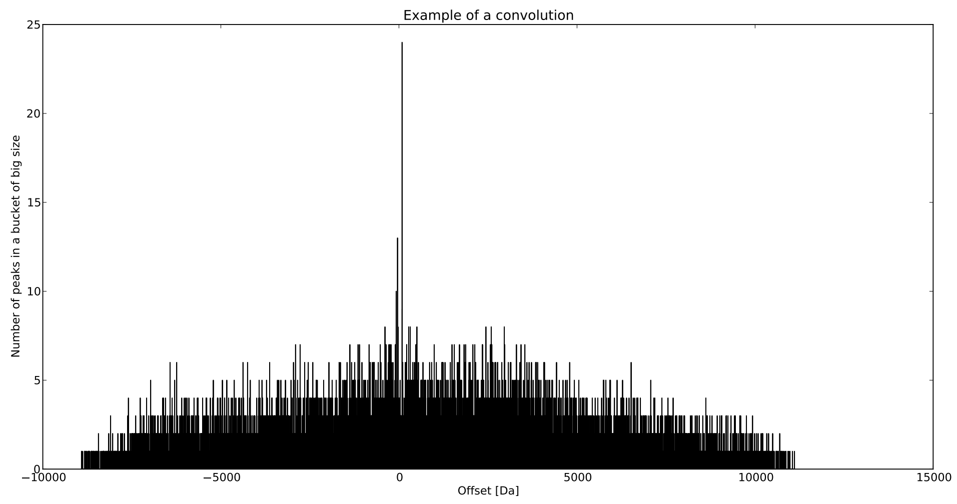


Figure 2.2: Spectral convolution between a protein and a spectrum. Here the diagonal score function is sampled and the maximum tolerable error is big. The high peak around 0 Da suggests that there is a significant alignment.

```
ExactConvolution($S, P, e$)
 1  $c \leftarrow \emptyset$
 2  for each $j \in [1 : m]$
 3      do for each $i \in [1 : n]$
 4             do ADD($c, b[j] - a[i]$)
 5  SORT($c$)
 6  $score \leftarrow 0$
 7  for each $p \in c$
 8      do $k \leftarrow$ number of peaks in c within the range $[p, p + e)$
 9         if $k > score$
10           then $score \leftarrow k$
11  return $score$
```

The correctness of this code is considered trivial and will not be proven. We will only do some considerations about time complexity.

Unfortunately the sorting phase is too time consuming: it requires $O(nm\log(nm))$ asymptotic time, if done naively. However, it can be improved. In fact, prior to convolution, the spectrum and the protein are represented by sorted arrays, therefore the nm convolution peaks computed in the inner loop of line 3 can be grouped into m groups of n already sorted peaks, and the time to merge the m groups is only $O(nm\log(m))$. Similarly, the nm convolution peaks can be grouped into n groups of m sorted peaks, and the time to merge them is $O(nm\log(n))$.

However the running time is still too long, as the statistics in Figure 2.3 show. On the other hand this algorithm has the unique feature that it can solve the spectral convolution problem without any approximation.

2.3.5 The approximate algorithm

It is customary to sort large amounts of data using algorithms that exploit the memory addressing capabilities of a computer (bucket sort, radix sort, hashing techiques) rather than comparison based algorithms (merge sort, quick sort, heap sort). This, of course, can be done also in this case. However we can do slightly better at the cost of some approximation error. In fact, in these sorting strategies, some overhead is due to the collision management: when two elements fall in the same bin something must be done in order to keep them separate[4]. But for the purposes of the spectral convolution problem, peaks falling within the same bin can be simply counted and their exact values disregarded, because the algorithm can base the detection of high concentration of peaks on the detection of bins containing a big number of peaks. This strategy introduces some errors that will be discussed later. Next, a simple implementation.

Claim 2.3.2 *Let $a[1:n]$ and $b[1:m]$ represent respectively a spectrum S and a protein P and let M_a and M_b denote the parent masses respectively of the spectrum and the protein. The following algorithm returns a score k such that*

$$max_{p\in R}\{(P\ominus S)^e(p)\}\leq k\leq max_{p\in R}\{(P\ominus S)^{2e}(p)\}$$

and it runs in time $O(mn+\frac{M_a+M_b}{e})$.

```
ApproxConvolution($a, b, e$)
1   $M_a \leftarrow a[n]$, $M_b \leftarrow b[m]$,
2   $c \leftarrow$ new array; $i_c(p) \equiv \lfloor\frac{p+M_a}{e}\rfloor$ for each $p$
3   for each $i \in [1:n]$
4         do for each $j \in [1:m]$
5                do $d \leftarrow i_c(b[j]-a[i])$
6                   $c[d] \leftarrow c[d]+1$
7
8   $score \leftarrow 0$
9   for each $i$ from 0 to $i_c(M_b)$
10        do $k \leftarrow c[i]+c[i+1]$
11           if $k > score$
12              then $score \leftarrow k$
13  return $score$
```

[4]Linked lists or open addressing are two commonly used techiques for this purpose

Before proving the claim a name is given to an array such as array c defined at line 1, because it will occur again in this work.

Definition The **convolution array** for protein P and spectrum S at resolution e and range (x, y) is an array c such that $c[i] = (P \ominus S)^e(ie+x)$, for $0 \leq i \leq \frac{y-x}{e}$. The function from real numbers to natural numbers $i_c(p) \equiv \lfloor \frac{p-x}{e} \rfloor$ is called **translation function** of the convolution array c.

The convolution array can be viewed as the diagonal score function with error tolerance e sampled each e Daltons. This is why the computation of the convolution array will be referred to as **convolution at resolution** e.

Proof Let c be the convolution array for protein P and spectrum S at resolution e and range (x, y) and let $i_c(p)$ denote its translation function. It is straightforward to notice that the following property holds:

$$|\,\{p \in P \ominus S \cap [x, y) : i_c(p) = k\}\,| = c[k].$$

Therefore, the array c computed in the pseudocode at line 7 is indeed a convolution array for protein P and spectrum S at resolution e with range $(-M_a, M_b)$. To prove the first part of the claim it suffices to notice that, for each p,

$$(P \ominus S)^e(p) \leq (P \ominus S)^{2e}(ei_c(p) + M_a) = c[i_c(p)] + c[i_c(p) + 1],$$

and that

$$c[i] + c[i + 1] \leq (P \ominus S)^{2e}(ei + M_a).$$

As for the time performance, the convolution stage takes time $\Theta(nm)$ while the iteration through the convolution array takes time $\Theta(\frac{M_a+M_b}{e})$. $\blacksquare$

In practical situations, when e is small (about 0.025 Daltons), the score returned by the approximate algorithm is seldom different from the score returned by the exact one, and when they differ the difference is small.

The time performance of this algorithm has an evident shortcoming: when e becomes very small the time for the detection of high diagonal scores increases dramatically and becomes more important than the time for the convolution stage. However, a simple variant of this algorithm can be devised that does not suffer from very small values of e, therefore it the most desirable choice between the two when high precision is required.

```
ApproxConvolution2($a, b, e$)
 1  $M_a \leftarrow a[n]$, $M_b \leftarrow b[m]$,
 2  $score \leftarrow 0$
 3  $c \leftarrow$ new array; $i_c(p) \equiv \lfloor \frac{p+M_a}{e} \rfloor$ for each $p$
 4  for each $i \in [1 : n]$
 5        do for each $j \in [1 : m]$
 6                 do $d \leftarrow i_c(b[j] - a[i])$)
 7                    $c[d] \leftarrow c[d] + 1$
 8                    if $c[d] > score$
 9                       then $score \leftarrow k$
10                    $d \leftarrow d + 1$
11                    $c[d] \leftarrow c[d] + 1$
12                    if $c[d] > score$
13                       then $score \leftarrow k$
14  return $score$
```

The differences between the old and the new version of the algorithm are the following.

- The array c is no more a convolution array at resolution e as in the old version: each position contains the sum of two consecutive positions of the convolution array.

- The detection occurs simultaneously with the convolution: every time the array is modified a check is done.

Besides these differences the two algorithms remain very similar, in fact they always return the same score, but the running time of the new version is $\Theta(nm)$. In Figure 2.3 the running time of the two algorithms is compared for different values of the error tolerance e.

The space requirements of both algorithms depend on e, in fact the size of the convolution array is $\frac{M_a+M_b}{e}$, therefore the smaller e, the bigger the space required. When e is about 0.5 Daltons, if the proteins and the spectra have reasonable parent masses, any current laptop computer should be able to load the entire array into the RAM[5], however when e goes under 0.025 Daltons, older computers might run out of RAM. Almost always the array does not fit into the cache memory.

[5]If the protein has a mass of 10^6 Daltons, which is very big, if the spectrum has the same parent mass, and if the convolution array is an array of 4 bytes long integers, the memory required to store the entire array is 16 MB, which is small enough to fit in any current laptop computer RAM.

2.3.6 Fourier transforms

Fourier transforms are a powerful tool when dealing with convolutions between sequences. Also in the present context Fourier transforms can be applied because the computation of the convolution array is almost a convolution between to sequences[6], as the present subsection aims to show.

In the previous algorithms we have implicitly used a sparse representation for the protein and the spectrum, i.e. if the spectrum S contains n peaks the array a that represents S contains n floating point values. The same is true for the protein representation. Another option, however, is to use a dense representation for the spectrum and the protein. We can assume that the spectrum peaks occur only at positions multiple of a fixed weight e and represent them through an array that contains only zeros or ones. A peak at position p is then represented by a one at position $\lfloor \frac{p}{e} \rfloor$, and the other positions of the array contain only zeros. Let $a_d[1 : n_d]$ and $b_d[1 : m_d]$ be the dense representations of respectively the spectrum and the protein. In the following manipulations the two arrays have to be thought of as sequences defined for every possible integer i: when i is negative or greater than the array maximum size the ith value of the sequence is 0. Suppose we want to compute the value of $(P \ominus S)^e(ek)$, where k is an integer.

Claim 2.3.3 *The sequence* $[(P \ominus S)^e(ek)]_{k \in Z}$ *is a convolution between two sequences.*

Proof

$$\begin{aligned}(P \ominus S)^e(ek) &= |\{(i,j) : b[j] - a[i] = ek\}| = |\{j : b_d[j] = 1 \wedge a_d[j-k] = 1\}| \\ &= \textstyle\sum_{j=k}^{m_d} 1 \cdot \{b_d[j] = 1 \wedge a_d[j-k] = 1\} = \sum_{j=k}^{m_d} b_d[j] \cdot a_d[j-k]\end{aligned}$$

Let a_d^{-1} denote the **mirror sequence** of a_d: $a_d^{-1}[i] = a_d[-i]$.

$$\begin{aligned}(P \ominus S)^e(ek) &= \textstyle\sum_{j=k}^{m_d} b_d[j] \cdot a_d[j-k] = \sum_{j=k}^{m_d} b_d[j] \cdot a_d[-(-j+k)] \\ &= \textstyle\sum_{j=k}^{m_d} b_d[j] \cdot a_d^{-1}[k-j] = (b_d \oplus a_d^{-1})[k]\end{aligned}$$

The last symbol denotes the convolution between sequences, therefore the diagonal score sampled every e Dalton is approximately the convolution between the protein dense representation and the mirror sequence of the spectrum dense representation. ∎

[6] The relation between Fourier transforms and sequence convolution, or polynomial multiplication, is assumed as known. The interested reader can find a comprehensive treatment in a famous textbook on algorithms [3].

Starting from the last relation, it is not hard to devise an algorithm that computes the convolution between the two sequences in time $O((m_d+n_d)\log(m_d+n_d))$ using the fast Fourier transform algorithm. This strategy would have two disadvantages however. First, the simplifying assumption that peaks occur at positions multiple of e is restrictive, and every discretization technique will introduce small errors in the convolution array. Unfortunately there is a much bigger problem with this algorithm, that is the value of m_d and n_d. The former is the mass of a protein made up of m amino acids, divided by e. Assuming an average amino acid mass of 110 Daltons and a maximum tolerable error of 0.025 Daltons, m_d is approximately 4400 times bigger than m. Such a constant is, of course, enormous in practical situations, therefore even if the algorithm based on fast Fourier transforms might be asymptotically faster than every other approach, the constants hidden in the O notation make it an unviable option.

2.3.7 Comparison of the different algorithms running time

The purpose of this subsection is to make the previous considerations about speed performance more quantitative. All the previous algorithms but the last one have been implemented and tested before developing MS-Filter. The development of these tools provided deep insights for the design of more sophisticated ones, and it is the author belief that these results might be useful for anyone trying to develop a fast filtering tool or a database search tool. The plot in figure 2.3 provides a visual representation of the algorithms performance. The source code of these algorithms, that are written in Java, is publicly available [12].

2.3.8 Statistics of the convolution array

The purpose of this subsection is to determine a statistical description of the convolution array. This is important for a number of reasons:

1. from a probabilistic model of the diagonal score, it is possible to address the problem of the analysis of significance of a given alignment,
2. it provides the designer of algorithms with a sense of the typical values its algorithm has to work with,
3. it provides a sound motivation for heuristic arguments.

First, the function $(P \ominus S)^e(p)$ will be characterized as a random variable, where the randomness is originated from the spectrum peak distribution and the protein sequence. Finally, a formula for the expected number of diagonals of a certain

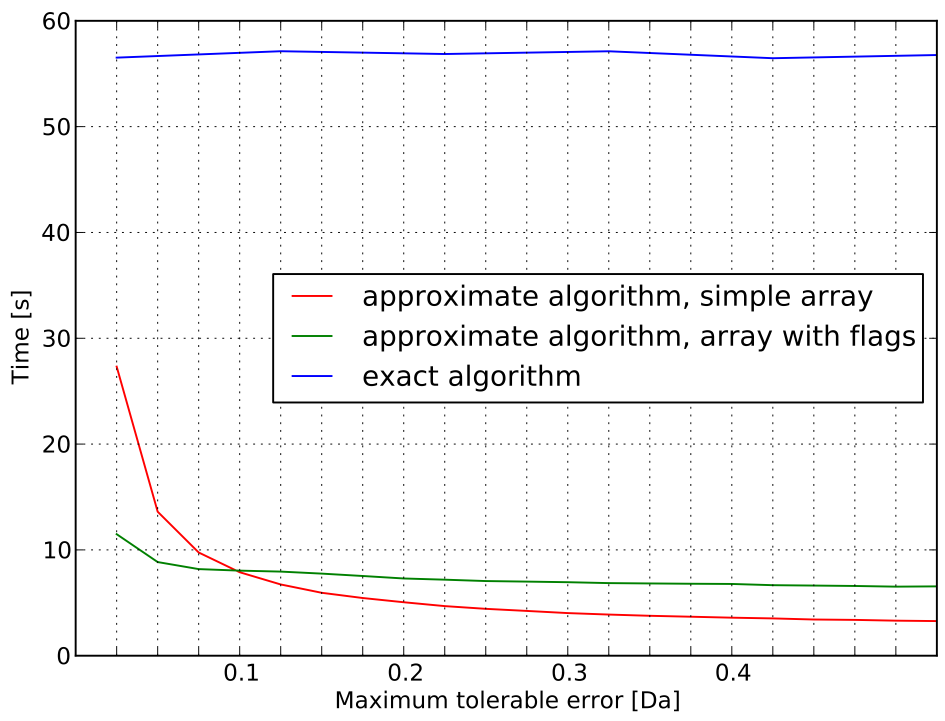


Figure 2.3: Comparison of the three presented algorithms running time as a function of the tolerable error. While the exact algorithm running time does not depend on the tolerable error, the two other algorithms heavily depend on it. In fact the tolerable error is inversely proportional to the size of the convolution array. When the convolution array is big the running time increases both because a smaller portion of it fits into the cache memory and because, for the simple array implementation, the whole array needs to be scanned. The data were obtained matching a spectrum with a parent mass of about $8 \cdot 10^3$ Da and 164 peaks against a database of about 4500 proteins, the *Salmonella Typhimurium* proteome.

length will be derived. This is a common approach for evaluating the statistical significance of an alignment, used also in other contexts (genome and proteome sequence alignment). In fact an estimate of the randomness of a certain alignment (p-value or e-value) determines how likely that alignment is to occur by chance, and therefore how likely it is not to be significant. For a sensitive choice of the scoring function, therefore, the bigger the e-value or p-value, the smaller the score. The following short treatment does not pretend to give an accurate answer to this problem, but only to describe qualitatively the role played by factors such as the number of peaks in the spectrum, the error tolerance and the protein mass.

However, the formulas derive from a rigorous probabilistic model.

The following assumptions will be made:

1. the spectrum has a certain parent mass M_a and contains a certain number n of peaks,
2. the spectrum peaks are uniformly distributed from 0 to M_a,
3. the protein sequence is random and infinitely long. The amino acids are chosen independently with probabilities equal to their frequencies in the dataset of interest.

The third assumption may seem strong, but it actually is not, in fact the results will be generalized to peptides of arbitrary length.

Claim 2.3.4 *Given the above model, the random variable $(P \ominus S)^e(p)$ is approximately a binomial variable with parameters n and $\frac{e}{\mu_{aa}}$, where μ_{aa} is the average residue monoisotopic weight.*

Proof The key observation is that once we choose p (assume p positive for now) there is a certain subset A of the interval $[0, M_a]$ such that every peak of S falling in A contributes to the value of $(P \ominus S)^e(p)$. The set A is precisely the union of $n(p)$ intervals of length e, where $n(p) = |\{j : p \leq b[j] < p + M_a\}|$, therefore the area[7] of A is $e \cdot n(p)$. Then the number of peaks falling into A is a binomial variable with parameters n and $\frac{|A|}{M_a} = \frac{en(p)}{M_a}$, where the first parameter is the integer parameter and the last is the probability parameter that define a binomial variable. It is a know fact that for a binomial variable of parameters n and P (the symbol P in this paragraph does not denote a protein) the mean and the variance are respectively nP and $nP(1-P)$. In this case, however, P is a random variable, therefore mean and variance are respectively $E[nP]$ and $E[nP(1-P)]$. The final step for computing $(P \ominus S)^e(p)$ first two moments is the determination of a statistical description for $n(p)$.

The random variable $n(p)$ depends on the protein sequence. Given that the amino acids in the protein sequence are chosen independently with probabilities equals to their frequencies, the statistical description of $n(p)$ depends on the amino acid mass probability distribution. The random variable $n(p)$ is the increase of a renewal process from time p to time $p + M_a$, where the interoccurrence times are given by the amino acid mass probability distribution. However, even without knowing this fact, it is intuitive that, denoting with μ_{aa} the average amino acid

[7]The case where one of the $n(p)$ intervals is not entirely contained into $[0, M_a]$ is overlooked, both because it is very unlikely with a sensible value for e and because even if it happens the adjustment to the area of A is very little

mass, $n(p)$ mean is about $\frac{M_a}{\mu_{aa}}$ for sufficiently large M_a. Moreover, denoting with σ_{aa}^2 the variance of the amino acid masses, $n(p)$ variance is about $\frac{M_a \sigma_{aa}^2}{\mu_{aa}^3}$. Derivations of these formulas will not be provided here, but the reader is pointed to appropriate references on renewal processes [8].

Finally, based on these considerations, we can conclude that

1. $(P \ominus S)^e(p)$ mean is $E[n \cdot \frac{en(p)}{M_a}] = n\frac{eE[n(p)]}{M_a} = n\frac{e}{\mu_{aa}}$,
2. $(P \ominus S)^e(p)$ variance is $E[n\frac{en(p)}{M_a}(1 - \frac{en(p)}{M_a})] = n(\frac{eE[n(p)]}{M_a} - \frac{e^2 E[n(p)^2]}{M_a^2}) = n(\frac{e}{\mu_{aa}} - \frac{e^2}{\mu_{aa}^2}\frac{\mu_{aa} M_a + \sigma_{aa}^2}{\mu_{aa} M_a})$.

In practical situations M_a is much bigger than μ_{aa} and from the graph in figure 2.4 it is easy to see that the standard deviation of the amino acid monoisotopic masses is relatively small compared to its mean (i.e. the monoisotopic masses are well concentrated around their mean). Therefore the expression for the variance can be safely reduced to $n\frac{e}{\mu_{aa}}(1 - \frac{e}{\mu_{aa}})$. In other words, $(P \ominus S)^e(p)$ behaves nearly identically to a binomial variable with parameters n and $\frac{e}{\mu_{aa}}$ (see figure 2.5). ∎

It is interesting to notice that a similar formula for the diagonal score distribution has already been devised and used by the developers of ProsightPTM [13]. In their approach, the score of a protein-spectrum alignment is a Poisson random variable of parameter $n\frac{e}{\mu_{aa}}$, which is therefore very similar to the formula just derived[8].

Now the analysis can be easily extended to proteins of finite length.

Claim 2.3.5 *Let P be the longest prefix of a random infinite amino acid sequence such that its parent mass is less than M_b, let S be a random spectrum with parent mass M_a and let $I(M_a, M_b, p)$ denote* $\min\{p + M_a, M_b\} - \max\{0, p\}$. *Then $(P \ominus S)^e(p)$ is approximately a binomial random variable with parameters n and $\frac{e}{\mu_{aa}}\frac{I(M_a,M_b,p)}{M_a}$.*

The first assumption states that P is a random protein with a mass slightly smaller than M_b. A strict constraint on the protein parent mass would, in fact, limit the number of possible proteins and make the analysis harder. The function $I(M_a, M_b, p)$ defined in the claim has a simple interpretation: it is the spectrum area overlapping with the protein in an alignment starting at p (see Figure 2.6 for a graphical representation).

[8]It is a known fact that a binomial random variable of parameters n and p, when n is large and p small, is very close to a Poisson random variable of parameter np. The precise formulation of this statement is sometimes referred to as *Law of rare events* and can be found in any textbook on random variables.

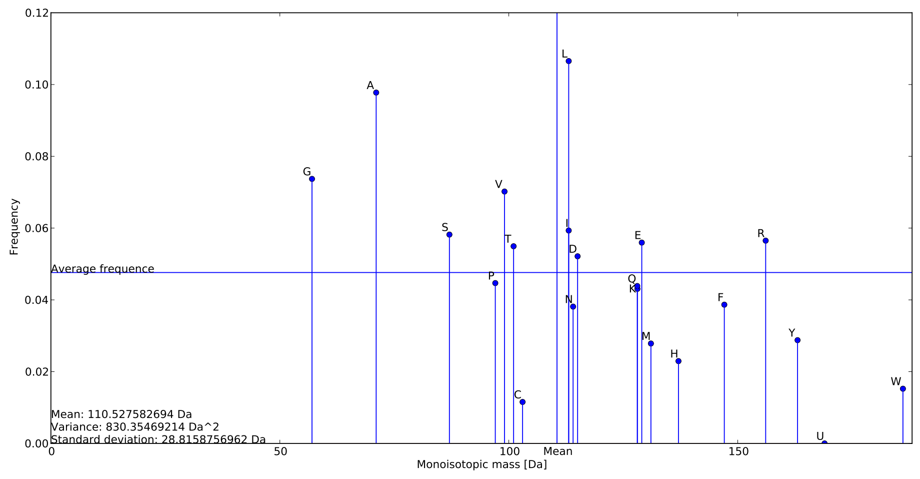


Figure 2.4: Probability mass function of the amino acid monoisotopic masses. The frequency values were obtained from the *Salmonella Typhimurium* proteome.

Proof The proof is similar to the proof of the previous claim. Now $n(p)$ is $|\{j : \max\{0,p\} \leq b[j] < \min\{p+M_a, M_b\}\}|$. In other words, $n(p)$ is the increase of a renewal process after a period of length $I(M_a, M_b, p)$, therefore its mean and variance are, respectively, $\frac{I(M_a,M_b,p)}{\mu_{aa}}$ and $\frac{I(M,N,p)\sigma^2_{aa}}{\mu^3_{aa}}$. Finally, $(P \ominus S)^e(p)$ mean and variance are, respectively, about $n\frac{e}{\mu_{aa}}\frac{I(M,N,p)}{M}$ and $n\frac{e}{\mu_{aa}}\frac{I(M,N,p)}{M_a}(1 - \frac{e}{\mu_{aa}}\frac{I(M_a,M_b,p)}{M_a})$. Again, the diagonal score behaves like a binomial variable with parameters n and $\frac{e}{\mu_{aa}}\frac{I(M_a,M_b,p)}{M_a}$ (see Figures 2.5 and 2.7). ▮

From the statistical description of $(P \ominus S)^e(p)$ we can easily derive a formula for the expected number of alignments with score greater than k. The probability that at a given position p the diagonal score with error e is greater than a certain threshold is:

$$Prob[(P \ominus S)^e(p) \geq k] = \sum_{i=k}^{n} \binom{n}{i} \left(\frac{e}{\mu_{aa}}\frac{I(M_a, M_b, p)}{M_a}\right)^i \left(1 - \frac{e}{\mu_{aa}}\frac{I(M_a, M_b, p)}{M_a}\right)^{n-i}.$$

Remember that the approximate algorithms described earlier sample the the diagonal score $(P \ominus S)^{2e}(p)$ every e Dalton. This is an approximation useful also for the computation of the expected number of good alignments. Let $Eval(M_a, M_b, n, k, e)$ denote the expected number of integer indexes i such that $(P \ominus S)^{2e}(ie) \geq k$.

$$Eval(M_a, M_b, n, k, e) = \sum_{i=\lfloor \frac{-M_a}{e} \rfloor}^{\lceil \frac{M_b}{e} \rceil} Prob[(P \ominus S)^{2e}(ie) \geq k].$$

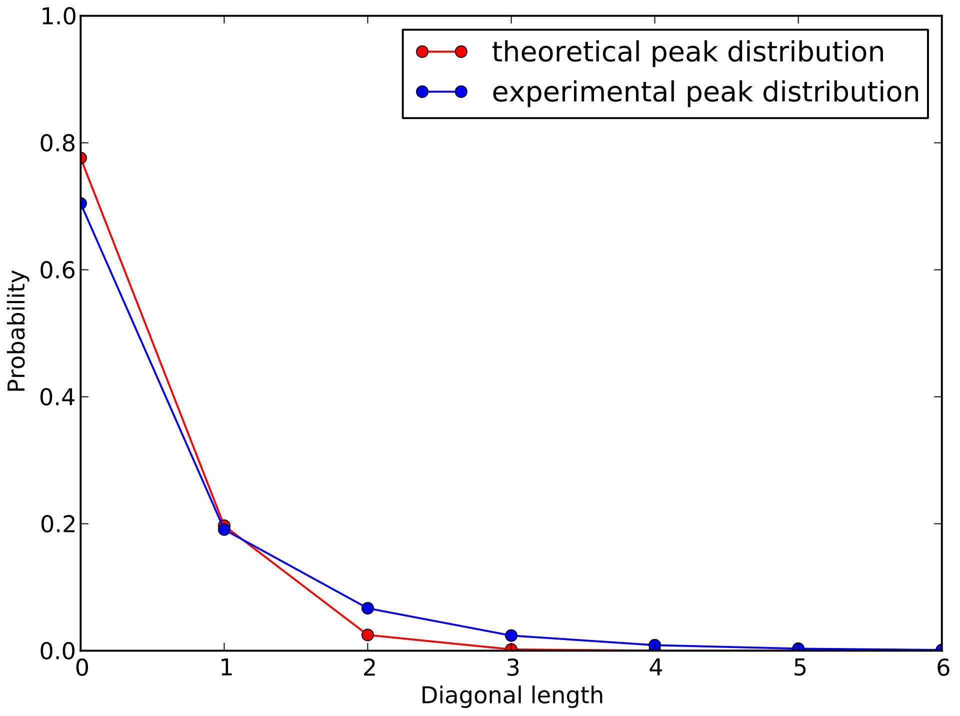


Figure 2.5: Probability mass function of the diagonal score between a random infinite protein and a random spectrum with 140 peaks at a specific offset and an error tolerance of 0.2 Daltons. The theoretical distribution is compared to experimental data from $3 \cdot 10^6$ protein-spectrum pairs.

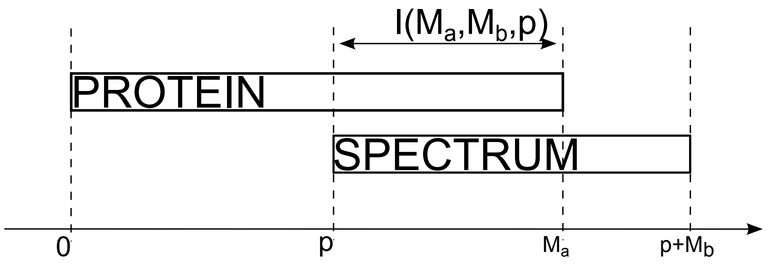


Figure 2.6: Graphical representation of the $I(M_a, M_b, p)$ function, which represents the portion of the spectrum that overlaps with the protein.

This formula estimates the randomness of a protein-spectrum alignment, therefore it can be used as a score.

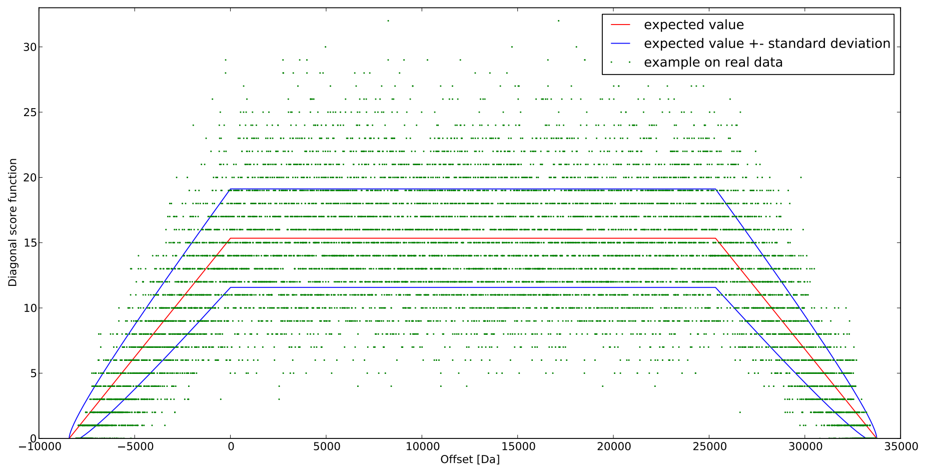


Figure 2.7: Visual representation of mean and variance of the diagonal score as a function of the offset with a protein of finite length. The central segment that caracterizes the mean curve corresponds to the mean of an alignment with a random infinite length protein and a random spectrum. The experimental data are obtained from a protein with mass of 33754.45 Daltons, a spectrum with a parent mass of 8420.75 Daltons and 212 peaks, and an error tolerance of 8 Daltons. Such a huge error tolerance has been chosen for the sake of clarity.

However caution must be used for the applicability of this result. Treating S as a random spectrum with a given parent mass and number of peaks, in fact, simplifies the calculations greatly, but in reality the expected number of alignments with score greater than k heavily depends on S exact nature. For instance, if S has n peaks all very close together, then with every protein P of mass M_b there will be about $2\frac{M_b}{\mu_{aa}}$ offsets (of type ie with i integer) such that the diagonal score is equal to n. In other words this e-value formula estimates the randomness of a spectrum-protein alignment given the spectrum mass, the spectrum number of peaks and the protein mass, but it does not estimate the randomness of a spectrum-protein match given S exact distribution. In database search of mass spectra, the spectrum used for the search is, of course, known, therefore an e-value that depends on S exact description is more appropriate. This is the approach followed in [7].

However, the formulas above can give a rough idea of how the significance of an alignment depends on k, n and the parent masses. These are the conclusions that can be drawn:

1. the probability of having a diagonal of length k with error e and offset o decreases almost exponentially with k,

2. the probability of a high diagonal score increases linearly with the mass of

the protein,

3. the number of diagonals with a score bigger than k is almost proportional to e^k, therefore it decreases polynomially as e decreases.

These considerations suggest that it is reasonable to score diagonals from the same spectrum according to their lengths. Moreover, in case two diagonals of a spectrum with two different proteins have the same score, it looks reasonable to consider more meaningful the alignment with the lightest protein. This is the scoring function used in all the implemented algorithms.

2.3.9 From a single protein to an entire database

Until now the discussion focused on algorithms that match a single spectrum against a single protein. This is obviously the most important subroutine that a filtering tool for database search must have. However the final algorithm has to address the additional problem of efficiently collecting the results from an entire database. The final output of a filtering tool is a reduced subset of proteins that are the most likely to be the interpretation of the spectrum. The number of proteins belonging to this set is called **capacity of the filter** and it can be chosen in several ways. The choice is strictly related to the choice for the scoring function for the protein-spectrum alignments. In all the approaches presented so far and in the following, the scoring function is approximately the diagonal score, without any normalization. It is evident that this scoring function is not based on the computation of an e-value for a protein-spectrum alignment, which is a delicate problem and which might be addressed after filtering, but it only provides a simple spectrum dependent score: such scores can be compared when they refer to alignments of the same spectrum with different proteins, but they do not say anything about alignments of different spectra with different proteins. This is why it is reasonable that a filtering algorithm based on such a scoring function has a fixed capacity rather than a score dependent capacity, because scores have a relative rather than absolute meaning. Tools for database search of top-down spectra, in contrast, usually try to provide an absolute score (non spectrum dependent) because they should measure how likely the output proteins are to be the correct interpretation of the spectrum. Some approaches score protein-spectrum alignments assuming a certain distribution of relative scores (this is the approach used in PIITA [16], for instance), others, like MS-Align+ and ProsightPTM, try to compute the e-value of protein-spectrum alignments and others use other scoring procedures.

The most efficient way of collecting the k top-scoring proteins is to store them into a data structure which keeps a partial ordering between the collected matches. It allows a fast inspection of the lowest scoring protein and at the same time fast insertion methods, which is all the algorithm needs. Such a data structure is called priority queue, it is usually implemented through a binary heap, and it is very common, therefore its details will not be discussed here (for details consult [3]).

Since the goal of the algorithm is to output the best C proteins, it is clear that, if at any time the lowest scoring alignment in the priority queue has score s and if there are already C alignments, every other alignment with score less than s can be safely disregarded, therefore s acts as a threshold score. Such a threshold is very useful because it dramatically reduces the search space: if the algorithm knows in advance that only diagonal scores greater that s deserve consideration, it runs faster. This is the reason why every implemented algorithm in reality solves the **thresholded version** of the spectral convolution problem:

Given a spectrum S, a protein P, an error tolerance e and a threshold t find the longest diagonal with error e between S and P greater than t.

2.4 The improvement: MS-Filter

MS-Filter algorithm is based on the spectral convolution approach and it can be seen as a variant of the approximate algorithm presented earlier. It improves, mainly, in two important features over the previous implementations: speed and sensitivity. While the speed improvement is technical, the sensitivity improvement is based on considerations about mass spectrometry instrumentation. They will be discussed in the next subsections. The results, instead, will be presented in the next chapter.

2.4.1 Two-stage convolution

As mentioned above, MS-Filter is an improvement of the approximate algorithm already presented. From figure 2.3 it clearly emerges that the running time of the approximate algorithms is heavily dependent on the tolerable error, especially if the latter goes below 0.025 Daltons, which is the trend imposed by newest mass spectrometric technologies. In other words there is a speed-sensitivity trade-off for spectral convolution based algorithms: the smaller the error tolerance, the slower the execution time. The two-stage convolution approach allows for a high precision diagonal detection while keeping the running time short.

The first idea behind this technique is that the threshold score can be used to reduce the number of calculations done by the algorithm. At any point in the program execution there is a threshold score t such that any alignment with score less than t can be safely disregarded. The threshold score t is due to three possible reasons.

1. The filtering algorithm is used by a tool that aligns a spectrum S with all the proteins from a database and collects the best C protein-spectrum alignments. It usually happens that, at a certain point in the program, C alignments with score greater than t have already been found. In that case any other alignment with score lower than t can be neglected, and t acts as a threshold score.

2. Even when the database consists of a single protein a threshold score t naturally arises. In fact only the best alignment from each protein is considered, therefore whenever an alignment with score t is detected, all the following alignments with score less than t can be ignored.

3. The user might want to set a threshold score to speed up the program or to ignore non significant alignments.

The second idea is that, given an error tolerance e and a threshold t, there are parts of the convolution array with resolution e that can be ignored without being computed. Let c' denote the convolution array at resolution e', where $e' > e$, and let $i_{c'}(p)$ be its translation function. From section 2.3.3 recall that $c'[i]$ is a sampling of the function $(P \ominus S)^{e'}(p)$ every e' Daltons and $c'[i]+c'[i+1]$ is a sampling of the function $(P \ominus S)^{2e'}(p)$ every e' Daltons. Now since $(P \ominus S)^{2e'}(p) \geq (P \ominus S)^{2e}(q)$, where $p \leq q < p + 2e' - 2e$, we can infer that:

1. If $t > c'[i_{c'}(p)] + c'[i_{c'}(p) + 1]$ the diagonals with error tolerance e and offset close to p can be safely ignored, because no diagonal at those offsets can have length greater than t.

2. If $t \leq c'[i_{c'}(p)] + c'[i_{c'}(p) + 1]$ there might be a diagonal with error tolerance e, offset close to p and a length greater than the threshold. Those offsets need to be further inspected.

Briefly, the computation of the convolution array c' at resolution $e' > e$, for a sensitive choice of e', is fast and greatly reduces the positions of the convolution array c at resolution e to be inspected[9].

The last key idea is that when only a small portion of the convolution array at high resolution is needed, it is not necessary to compute the whole array, that is by far the most expensive operation in the basic approximate algorithm, but just the portion of interest, and this can be done in an efficient way.

Claim 2.4.1 *Let $a[1:n]$ represent a spectrum S, $b[1:m]$ the theoretical spectrum of a protein P, and aa_{min} the monoisotopic mass of the lightest amino acid. If*

[9]The determination of c exact positions has been voluntarily omitted because it would make this treatment heavier to read and because it is straightforward.

$\delta \leq aa_{min}$, then the following algorithm computes correctly the convolution array at resolution e in the restricted range $[p, p+\delta)$ in asymptotic time $O(n+m)$.

```
RestrictedConvolution($a, b, p, \delta, e$)
 1  $c \leftarrow$ new array; $i_c(p) \equiv \lfloor \frac{p+M_a}{e} \rfloor$ for each $p$
 2  $i \leftarrow 0$, $j \leftarrow 0$
 3  while $(i \leq n) \wedge (j \leq m)$
 4        do $d \leftarrow b[j] - a[i]$
 5           if $d < p$
 6                 do $j \leftarrow j + 1$
 7              else
 8                    $i \leftarrow i + 1$
 9                    if $d < p + \delta$
10                          do $c[i_c(d)] \leftarrow c[i_c(d)] + 1$
11
12  return $c$
```

Proof We need to prove that every convolution peak in $[p, p+\delta)$ is counted into array c. Let $p(j, i)$ denote the convolution peak $b[j] - a[i]$, and let the ordered pair (y, x) denote an iteration of the while loop at line 3, where x and y are the values assumed respectively by the variables i and j at the beginning of each iteration. In the following, $(a, b) < (c, d)$ is equivalent to $(a < c) \wedge (b < d)$. We will prove by induction that, at the beginning of each iteration (y, x) done by the algorithm, all the peaks $p(m, l)$ such that $(m, l) < (y, x)$ and $p(y, x) \in [p, p+\delta)$ have been counted into the convolution array, and that if $p(y, x) < p$ then $x = 0$ or $p(y, x-1) \geq p$.

At iteration $(0, 0)$ this is obviously true. Suppose now that the induction hypothesis is true for iteration (y, x), we need to prove that it remains true at the next iteration. There are two possibilities.

1. $p(y, x) < p$. There are no peaks $p(m, l)$ with $(y, x) \leq (m, l) < (y+1, x)$ such that $p(m, l) \in [p, p+\delta)$. In fact if $m = y$ then $p(y, l) < p(y, x) < p$ and if $m = y+1$, then, unless $x = 0$, $p(y+1, x-1) > p+\delta$, because:

$$p(y+1, x-1) = p(y, x-1) + (b[y+1] - b[y]) \geq$$
$$p(y, x-1) + aa_{min} > p(y, x-1) + \delta \geq p + \delta$$

   The last step is motivated by the induction hypothesis. Since the next iteration has index $(y+1, x)$ and since the algorithm does not count any peak, at the following iteration all the convolution peaks $p(m, l)$ with $(m, l) < (y+1, x)$ have already been counted. Moreover if $p(y+1, x) < p$ then, unless $x = 0$, $p(y+1, x-1) = (b[y+1] - b[y]) + p(y, x-1) > p$. Again the last step follows from the induction hypothesis.

2. $p(y,x) > p$. Then the only pair (m,l) such that $(y,x) \leq (m,l) < (y,x+1)$ is (y,x). Consistently, if $p(y,x) < \delta$ then it is inserted in the convolution array, otherwise not. Finally, if at the following iteration it occurs that $p(y,x+1) < p$ then $p(y,x) > p$ according to the induction hypothesis.

This concludes the correctness proof.

As for the time complexity, in every iteration of the while loop at line 3 no more than a constant number of operations is performed and either i or j are increased. From these two observations it follows that the time complexity is $O(n+m)$. ▮

After this result, understanding what MS-Filter does is straightforward (see Figure 2.8 for a schematic representation of the workflow).

1. It computes the convolution array at resolution $e' > e$.

2. It detects candidate intervals $[p, p+\delta)$ where the diagonal score with error e could be greater than t. Close intervals are grouped together into a buffer, called interval buffer. In fact for two intervals $[p_1, p_1+\delta_1)$ and $[p_2, p_2+\delta_2)$ such that $p_2+\delta_2-p_1 < aa_{min}$, a single restricted convolution at high resolution is enough to detect possible high diagonal scores. Therefore the interval buffer contains one or more candidate intervals and identifies a larger interval of length less than aa_{min} where a restricted convolution needs to be computed.

3. The restricted convolution at high resolution is computed for the region of interest, high diagonal scores are detected and the interval buffer is emptied. Whenever an alignment with score t' greater than t is found, the threshold at both convolution stages increases to t'.

2.4.2 Accounting for errors at the deconvolution stage

As mentioned in the introduction, MS-filter works with deconvoluted spectra. Spectrum deconvolution is an error prone process, because it attempts to determine what the monoisotopic mass of a protein fragment is from its isotopomer envelope and because it tries to tell apart envelopes from noise. Because an isotopomer envelope of a protein fragment usually appears as a sequence of peaks spaced approximately one Dalton one from the next, an error in the determination of the beginning of the sequence causes an error in the determination of the monoisotopic mass of the fragment that is a approximately a multiple of one Dalton. Figure 2.9 illustrates this phenomenon. The most common errors of this kind are +1/-1 Dalton shift errors. In the following they will be called simply **shift errors**. For the dataset used in this work shift errors have been estimated to occur approximately 10-20% of the times.

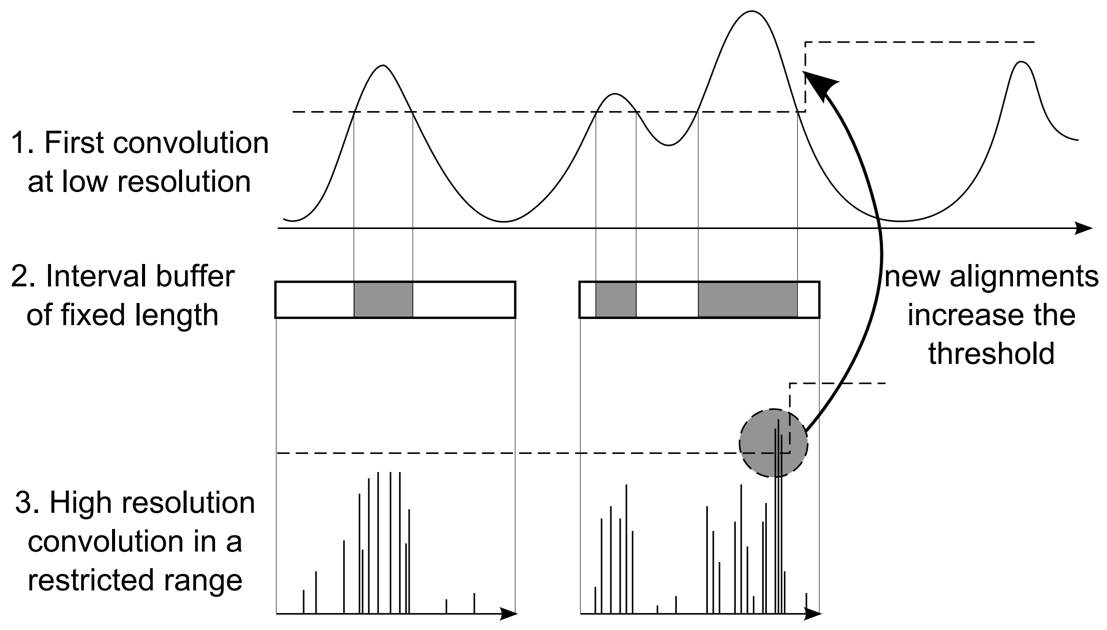

1. First convolution at low resolution
2. Interval buffer of fixed length
3. High resolution convolution in a restricted range
new alignments increase the threshold

Figure 2.8: Illustration of the two-stage convolution approach. The first operation is a convolution at low resolution, i.e. the computation of a convolution array at low resolution. When an interval is detected that contains a number of peaks greater than a given threshold, it is collected into a buffer, until no more intervals can be collected such that the range between the beginning of the first interval and the end of the last is smaller than a fixed quantity. Finally, in a last high resolution convolution stage, high diagonal scores are detected with precision in the interval of interest. When a score higher than the threshold is detected, the latter increases.

In the spectral convolution $P \ominus S$ these errors cause a diagonal of length l at position p to be split into three smaller diagonals at positions $p-1$, p and $p+1$, of which the diagonal at position p should be roughly 90% longer than the other two on average. One way of accounting for this phenomenon is to consider a modified diagonal score function $\overline{(P \ominus S)^e}(p) = (P{\ominus}S)^e(p) + (P{\ominus}S)^e(p{+}1) + (P{\ominus}S)^e(p{-}1)$. This is equivalent to consider a modified spectral convolution $\overline{P \ominus S}$ that contains, for each convolution peak p from the multiset $P \ominus S$, the peaks p, $p+1$ and $p-1$. But this solution results in the following problems:

1. the running time required for a whole array convolution would be approximately three times bigger,
2. the noise increases significantly,
3. the new diagonal score does not take into account that the diagonal at position p should be longer than those at position $p-1$ and $p+1$.

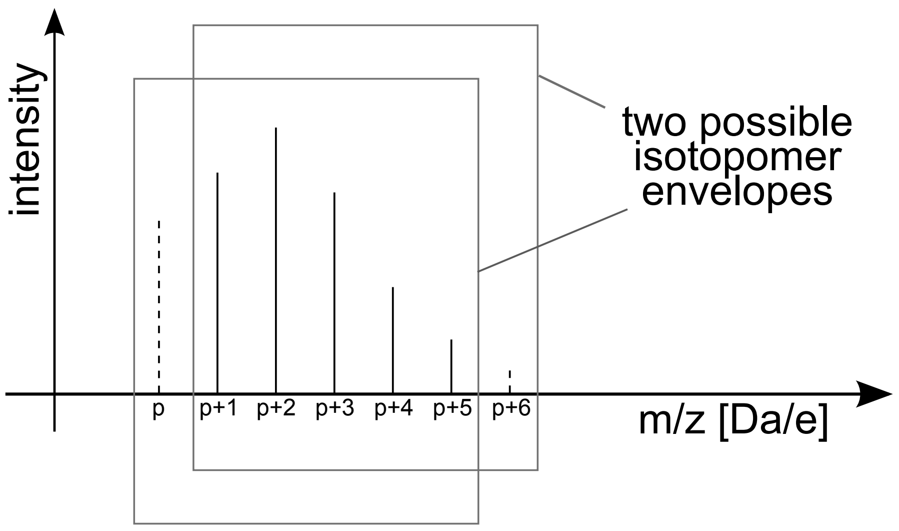


Figure 2.9: Isotopomer envelopes in the raw spectrum. In this example the ions are singly charged and the peaks are spaced by approximately one Dalton. An error in the determination of the first and the last peak of the isotopomer envelope can cause an error in the correct determination of the monoisotopic residue mass of the ions. Peak intensities help to prevent this error, because the ratios between intensities can be guessed from the envelope position, but errors are still frequent.

Therefore the algorithm would not only be significantly slower, but also more likely to output false positive alignments. For these problems, instead of using the new diagonal score function, it seems more appropriate to use the old one and to take into consideration possible shifts as a final correction. This is what MS-Filter does. The rest of this subsection describes how this correction is integrated into the two-stage convolution technique.

These are the modifications to the algorithm presented earlier.

1. At the first low resolution convolution stage, a threshold $t' < t$ is used. The idea behind this choice is that ignoring the mass shifts, a diagonal should still be visible in the spectral convolution, but its length diminished proportionally to the number of shifted peaks. The choice for t' should depend on the expected number of shift errors. For a 10-20% percentage of shift errors, a sensitive value for t' is $0.7t$ (rounded to the closest integer).

2. At the second high resolution convolution stage, the shift errors can be accounted for using a slightly modified restricted convolution algorithm. If the convolution array from p to $p + \delta$ has to be computed, for each peak d the peaks $d + 1$ and $d - 1$ are added to the convolution array as well.

The first correction increases the number of candidate offsets to examine, but it is robust to equally long random triples of diagonals distributed according to the $p-1, p, p+1$ pattern. The second correction makes the restricted convolution algorithm slightly slower but less than three times slower.

As it will be shown in the result section, this correction makes MS-Filter more sensitive at the cost of an increase in the running time. In some cases, especially with good quality spectra, this feature is not desirable, therefore it can be turned off.

2.4.3 Parameters of the algorithm

There are only two important parameters that need to be chosen for this algorithm.

The first one is the final error tolerance e used in the high resolution convolution stage. This parameter, which needs to be chosen for every other spectral convolution based approach, should depend on the spectrometer mass accuracy and on the typical parent masses of the spectra that need to be interpreted. For the spectrometer and the spectra in use, an error tolerance of 0.025 Daltons proved appropriate.

The second parameter is e', the resolution used in the first convolution stage. This parameter is peculiar to MS-Filter and its value has a fundamental impact on the algorithm speed. Choosing e' too large would cause a big number of candidates to be produced and the second high resolution convolution stage to be executed too often. On the other hand, choosing e' too small would cause the first low resolution stage to run slower. A theoretical justification for e' is not easy to devise, intuitively, e' should increase as e increases and decrease as the number of peaks in the spectrum increases. The strategy used in this work for the choice of e' is completely experimental, i.e., the parameter that yielded the best results was chosen ($e' = 0.4$ Daltons). While in general such a strategy might lead to a learning effect, i.e. an algorithm that performs well only in the dataset in use, in this case the dataset was big and diverse enough that the choice for e' should be appropriate for a wide variety of spectra.

Chapter 3

Experimental results

The aim of this chapter is to present MS-Filter performance: in particular its speed and sensitivity. For a performance analysis to be informative, however, other tools that solve the same computational problem should be used as reference. For the present problem, i.e., database filtering for top-down spectra in blind mode, no tool is known to this author other than the current implementation of the filtering stage of MS-Align+, because the identification of proteins with unexpected modification is a relatively unexplored area. The results presented in this section, therefore, compare MS-Filter performance to the current implementation of MS-Align+ filtering stage. Before that, however, it is necessary to discuss the methods used for the speed and sensitivity analysis, the used dataset, and a quick overview of the algorithm used as reference.

3.1 Speed measurements

Not only MS-Filter running time is compared to that of the reference algorithm on the same dataset, but the running time is analyzed as a function of the input size and the error tolerance. The size of the input is defined in two ways.

Given a database D containing N_d proteins, denoting with M_{d_i} and m_{d_i} respectively the mass and the number of residues of the ith protein in the database, given a spectrum S with mass M_s and with m_s peaks, we define two functions of the input.

- **Matrix size**: $\sum_{i=1}^{N_d}(m_{d_i} \cdot m_s)$,
- **Array size**: $\sum_{i=1}^{N_d}(M_{d_i} + M_s)$.

These two definitions take into account that there can be two bottlenecks in the algorithm execution: the convolution stage, which takes time proportional to the product $m_{d_i} \cdot m_s$, and the iteration through the whole convolution array, which has size proportional to $M_{d_i} + M_s$ and inversely proportional to the error tolerance.

3.2 Sensitivity measurements

The purpose of a filtering algorithm is to select a small number of candidate proteins from a big database in the hope that the correct protein has not been filtered out. The correctness can be assessed by more accurate algorithms or manual inspection. When a dataset of spectra is available such that for each spectrum the correct protein is known, one natural measurement is the percentage of spectra such that the filter outputs the correct protein. This measurement is called **correct filtering percentage (CPG)**, and it is capacity dependent. For an estimate of the goodness of the scoring function another measurement is introduced. The **average rank (AR)** is the average value of the minimum capacity such that the algorithm outputs the correct protein.

3.3 The dataset and the reference algorithm

The dataset consists of 1016 deconvoluted top-down spectra. The proteins that the spectra were obtained from, belong to the bacterium *Salmonella Typhimurium.* These spectra have a parent mass of 9000 Daltons on average and an average number of peaks of 140. For each spectrum the correct protein is known, because it has been previously identified using MS-Align+ and because the e-value associated to the protein-spectrum alignment is very small. The database is the *Salmonella Typhimurium* complete proteome, obtained from SWISSPROT. It consists of about 4500 proteins with an average length of 400 peptides. This dataset is referred to as the *Salmonella Typhimurium* dataset.

The reference algorithm, as mentioned earlier, is the current implementation of the filtering stage in MS-Align+, which is very similar to the approximate algorithm presented in the previous chapter. It has some slight modifications though, which are briefly summarized below.

1. The resolution for the convolution stage is very small: 0.01 Daltons.

2. The score at each position p is estimated summing three contiguous positions of the convolution array instead of two:

$$(P \ominus S)^{2e}(p) \leq c[i_c(p)] + c[i_c(p) + 1] + c[i_c(p) + 2]$$

Despite these differences, the algorithm is very similar to the approximate algorithm presented in Subsection 2.3.5, especially to the second implementation.

3.4 Results

3.4.1 Performance on the *Salmonella Typhimurium* dataset

Table 3.1 shows the overall performance of three different algorithms. The first one is a version of MS-Filter that does not use the shift errors correction, the second one is a version of MS-Filter that uses the shift errors correction, and the third one is the reference algorithm mentioned above. It clearly emerges that the two versions of MS-Filter outperform in speed the reference algorithm by almost an order of magnitude. Also in sensitivity the two algorithms perform better, probably because the reference algorithm uses a too small error tolerance. Considering the two versions of MS-Filter it turns out, as expected, that using the shift errors correction the algorithm is more accurate, as the average filter capacity statistic shows, but it is also slower. Using a capacity of 30 the correct filtering percentage is almost unaffected by the shift errors correction.

Algorithm	**ATS**	**CFP**	**AR**
MS-Filter 2.01	2.66 seconds	98.31%	8.70
MS-Filter 5.0	3.38 seconds	98.41%	4.21
Reference algorithm	23.20 seconds	97.42%	8.71

Table 3.1: Performance of the three algorithms on the *Salmonella Typhimurium* dataset with a fixed capacity of 30 on a dual core 2.5 GHz laptop computer. For display purposes some names have been shortened: ATS stands for average time per spectrum, CFP stands for correct filtering percentage and AR for average rank.

3.4.2 Running time as a function of the input size

Figures 3.1 and 3.2 show how the running time is affected by the size of the input. The input size is measured using the two functions defined earlier. It emerges that, even if MS-Filter performs a sequential scan of the whole convolution array, the running time is not directly influenced by the size of the convolution array, but it seems to be directly proportional to the product of the number of peaks in the spectrum and the number of amino acids in the protein. From this observation we can conclude that the time needed to iterate through the convolution array is small compared to the the time needed for the convolution stage.

The second straightforward observation is that MS-Filter performs almost always a lot better than the reference algorithm. As the input size increases, the running times diverge more and more.

Finally Figure 3.1 also reveals an important characteristic of MS-Filter. Even if the running time is in general proportional to the matrix size of the input, there is a relevant number of spectra for which the algorithm takes considerably longer. This hardly predictable behavior is due to the two-steps convolution approach. The number of candidate offsets selected in the first phase, in fact, strictly depends on the spectrum exact distribution. If the peaks are randomly distributed, then the number of candidates is expected to be small, but particular spectra can generate a very big number of candidates and take a longer than expected running time.

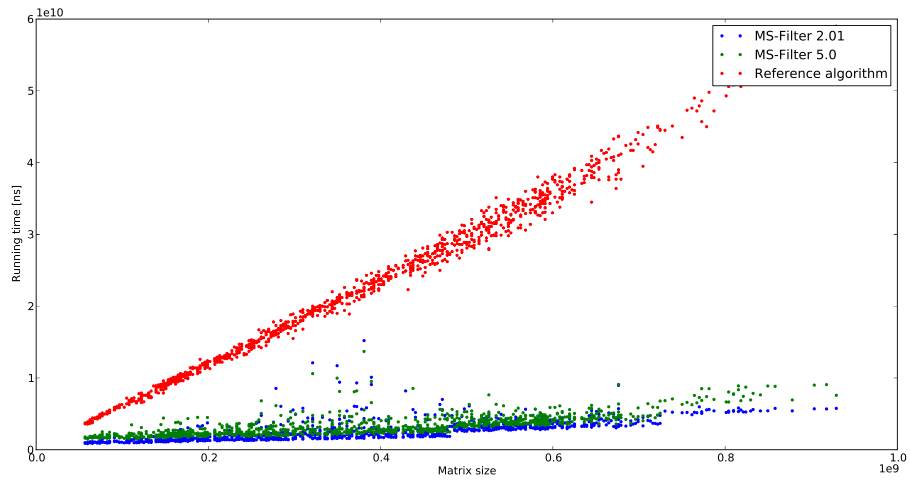


Figure 3.1: Running time of the three algorithms as a function of the matrix size of the input.

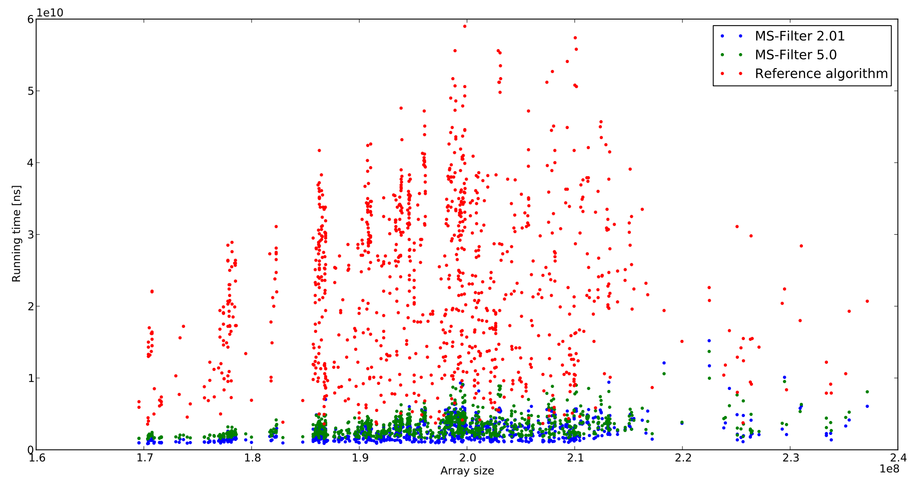
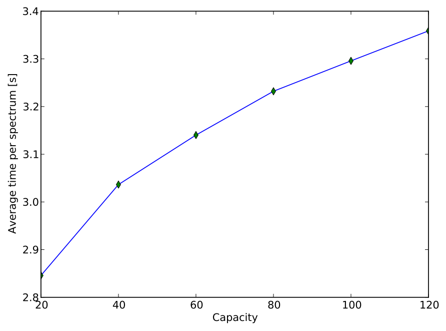


Figure 3.2: Running time of the three algorithms as a function of the array size of the input.



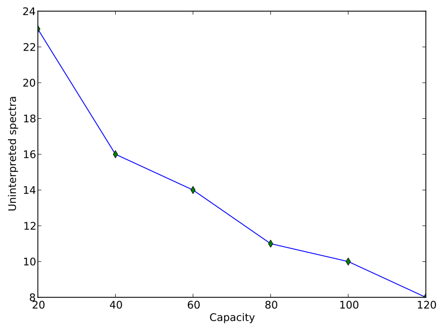


Table 3.2: Average time per spectrum and number of uninterpreted spectra as a function of the capacity. The algorithm used is MS-Filter without the shift errors correction and the dataset is the *Salmonella Typhimurium* dataset.

3.4.3 The impact of the filter capacity

The figures in Table 3.2 show how MS-Filter performance depends on the capacity. The plots were obtained using MS-Filter 2.01, i.e., the version without the shift errors correction, but MS-Filter 5.0 has a very similar performance. It is evident and reasonable that, increasing the capacity, the average time per spectrum increases and the correct filtering percentage increases as well (for display purposes the number of uninterpreted spectra was plotted instead of the correct filtering percentage). However, MS-Filter is particularly sensitive to the capacity. In fact a smaller capacity causes a higher threshold, which in turn causes a faster execution.

Chapter 4

Conclusions

The purpose of this work is to present a novel algorithm for top-down spectra database search. This new tool constitutes a useful addition to proteomics, because, in combination with MS-Align+ and other new tools, it paves the way for the large scale characterization of proteins.

As seen in the Introduction the top-down approach to mass spectrometry has exclusive advantages. Post-translational modifications play a fundamental role in cell to cell communications, signal transduction and regulatory functions and today there is no effective technology for reading out this language at a molecular level. While the traditional bottom-up approach seems more powerful for protein identification, it cannot compete with the top-down approach for protein characterization. However, algorithmic tools for top-down proteomics are still in their infancy. Some of them are unable to recognize proteins with unexpected modifications, others require high quality spectra for the generation of long tags and others are adaptations of algorithms designed for bottom-up spectra. In contrast, MS-Align+ is specifically designed for top-down spectra interpretation, can consider any kind of modification and provides a reliable analysis of statistical significance of the results. However, it lacks an effective algorithm for filtering big databases in blind search mode.

The second chapter presented the algorithm that makes MS-Align+ capable of considering expected and unexpected modifications: the spectral alignment algorithm. Then, the problem of designing a filtering algorithm for MS-Align+ was thoroughly discussed. It has been shown that spectral convolution is a powerful approach for a filtering algorithm and that it can be implemented in several ways. Some simple algorithms based on spectral convolution have been presented and developed. These software tools have been tested on a real dataset and their performance has been analyzed. A basic theoretical model has been developed to study the statistical behavior of the spectral convolution and useful insights have been derived from this analysis. Building on top of the experimental and theoretical work, a new powerful algorithm was presented which contains an original idea for the analysis of the spectral convolution. Moreover, a more accurate version of the same algorithm has been developed that accounts for common errors in the input spectra. This version offers a higher sensitivity at the expense of an increased running time.

The third chapter offered a detailed analysis of the new algorithm performance and compared it with the previously adopted one. It clearly emerges that an order of magnitude speedup has been achieved without affecting sensitivity. It is this author's belief that this tool offers a substantial improvement over the past implementations and that it can make MS-Align+ a even more powerful algorithm for database search in blind mode.

Future work might be done to improve speed or sensitivity of the filtering algorithm. When a dataset is made up of several spectra to match against several proteins, clustering spectra or preprocessing the database are two strategies that are likely to be fruitful. As for sensitivity, moving away from the abstract problem and taking in more and more biological and technological details peculiar to top-down mass spectrometry might prove important. For instance, a filtering tool might account for post-translational modifications more accurately than MS-Filter, use a relative error tolerance rather than absolute or score diagonals based on the intensities of the peaks. However, the purpose of a filtering tool should not be confused with that of a stand-alone database search tool: the former has to be fast and sensitivity is required only to filter out proteins that are unlikely to be the spectrum interpretation, while the latter has to be built on a very accurate scoring function for protein-spectrum alignments.

Appendix A

Proteins and top-down mass spectrometry

In this appendix some basic notions about proteins and top-down proteomics are presented. The material has been chosen in order to allow almost any reader to understand this work, rather than to provide an introduction to biology or proteomics. For these purposes the reader is pointed to some appropriate references [2][1].

A.1 Amino acids and proteins

Amino acids are the fundamental building blocks of proteins. Every amino acid has the chemical structure outlined in Figure A.1: it is composed of a central carbon atom, called α-carbon, along with an amino group (NH_2), a carboxyl group ($COOH$), a side-chain (also called R-group) and a hydrogen atom all attached to the α-carbon. The only component that differentiates one amino acid from another is the side-chain, a molecule that can be more or less complex. There are 20 types of amino acids naturally occurring in proteins, each with its mass and chemical properties.

During protein synthesis, the amino acids are chained one after another, forming a linear structure called **polypeptide** (see Figure A.2). The chemical bond that binds the carboxyl group of one amino acid to the amino group of another with the loss of a water molecule is called **peptide bond**. By convention, a protein is oriented from the amino acid with an unbound amino group to the amino acid with an unbound carboxyl group. The two ends are called respectively **N-terminus** and **C-terminus**. When an amino acid is bound at both ends to two other amino acids, it loses some atoms (two hydrogens and an oxygen) and what remains is called **amino acid residue** (see Figure A.1 again).

The most important property of an amino acid for the purposes of top-down mass spectrometry is the amino acid residue mass. However, due to the natural

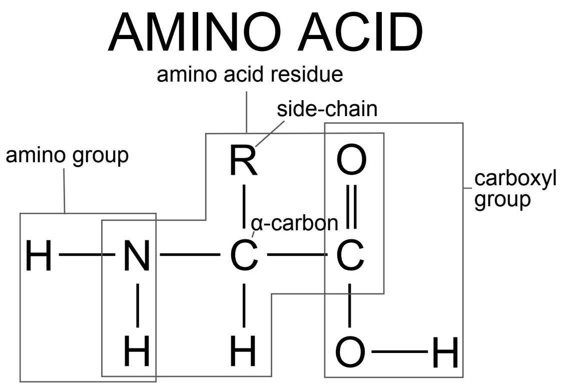


Figure A.1: Structural formula of an amino acid. The amino group, the carboxyl group, the central α-carbon, the side-chains and the residue are outlined.

abundance of isotopes, this mass can be defined in more than one way. The **monoisotopic** mass of a molecule is the sum of the masses of its atoms using the unbound, ground-state, rest mass of the principal (most abundant) isotope for each element instead of the isotopic average mass. Table A.1 lists the 20 natural amino acids, with their names, symbols and monoisotopic residue masses. From the sum of the monoisotopic residue masses of each amino acid in a protein, the monoisotopic mass of the whole protein can be easily obtained adding the weight of a water molecule, which is about 18 Daltons.

A.2 Top-down mass spectrometry

A mass spectrum is essentially a set of peaks coming from charged protein fragments. The process that leads to the final spectrum in a mass spectrometer usually consists of the following steps.

1. From a complex sample, a big number of identical proteins is isolated in a separation step and their mass is determined. This mass is called **parent mass** or **precursor mass** of the spectrum.

2. Each of these identical proteins is fragmented, the fragments are charged and the ions are accelerated towards the detector plate.

3. Each peak in the spectrum comes from a set of ions that impact the detector plate at the same time. Since the impact time is the same, their mass-to-charge ratio is the same, and it is determined based on the impact time.

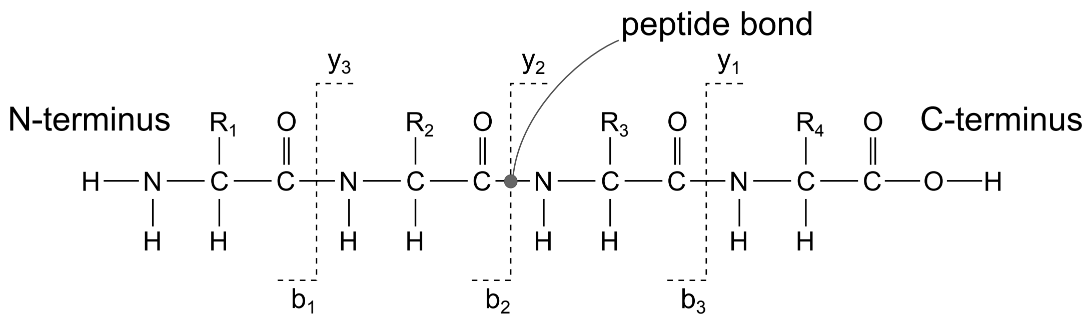


Figure A.2: Structural formula of a simple protein. The N-terminus and C-terminus are the two ends of a protein. At the N-terminus there is an amino acid with an unbound amino group and at the C-terminus an amino acid with an unbound carboxyl group. The peptide bond binds one amino acid with the next. The most important ion types that occur in top-down mass spectrometry derive from fragmentation at specific points. According to the fragmentation point, a specific ion type with a specific index is produced. A *b*-ion is a fragment that includes the N-terminus and a *y*-ion includes the C-terminus. The ion index is the number of residues in the fragment.

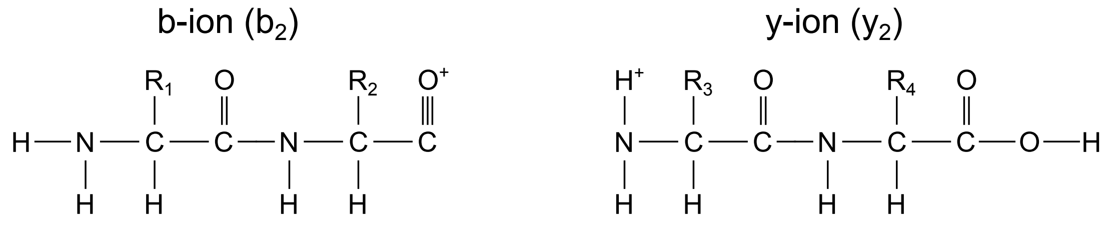


Figure A.3: Structural formula of a *b*-ion and a *y*-ion. From the structural formulas it is easy to derive the relations between monoisotopic masses presented in this section.

In a raw spectrum, each peak has a position, that is the mass-to-charge ratio of a set of fragments that hit the detector plate at the same time, and an intensity, that depends on the number of fragments in this set.

Two kinds of ion are expected to be found most frequently in top-down mass spectrometry: ***y*-ions** and ***b*-ions**. Figures A.2 and A.3 provide a pictorial description of them. Roughly speaking, *b*-ions and *y*-ions are respectively prefixes and suffixes of the parent protein, thinking to a protein as a string in the amino acids alphabet. The index of a *b*-ion is the number of residues in the prefix and, similarly, the index of a *y*-ion is the number of residues in the suffix. The **monoisotopic residue mass of a *b*-ion** is simply the sum of the monoisotopic residue masses of the amino acids in the prefix, while the **monoisotopic residue mass of a y-ion** is, conventionally, the sum of the monoisotopic residue masses of the amino

Name	**Code**	**M.r.m. [Daltons]**
Glycine	G	57.02147
Alanine	A	71.03712
Serine	S	87.03203
Proline	P	97.05277
Valine	V	99.06842
Threonine	T	101.04768
Cysteine	C	103.00919
Isoleucine	I	113.08407
Leucine	L	113.08407
Asparagine	N	114.04293
Aspartic Acid	D	115.02695
Glutamine	Q	128.05858
Lysine	K	128.09497
Glutamic Acid	E	129.04260
Methionine	M	131.04049
Histidine	H	137.05891
Phenylalanine	F	147.06842
Arginine	R	156.10112
Tyrosine	Y	163.06333
Tryptophan	W	186.07932

Table A.1: The 20 natural amino acids. For each amino acid the table shows its name, its one letter code (a three letter code is also common) and its monoisotopic residue mass.

acids in the suffix plus the weight of a water molecule. In a protein of length n, a b-ion of index i and a y-ion of index $n-i$ will be called **corresponding ions** in this work. The sum of the monoisotopic residue masses of two corresponding ions gives the monoisotopic mass of the whole protein. Monoisotopic residue masses of b-ions and y-ions can be obtained from their monoisotopic masses by subtracting the weight of a hydrogen molecule, which is roughly 1 Dalton.

Ions of the same kind and same index may produce many different peaks in the raw spectrum for two reasons, apart from measurement errors.

1. Some ions might be differently charged, so they have different mass-to-charge ratios.

2. Some ions might have the same charge but different isotopes in the molecule. Because of isotopes, if the charge of the fragments is q, the detected peaks are distant a multiple of m_n/q, where m_n denotes the mass of a neutron

(roughly 1 Dalton). Peaks corresponding to ions of the same kind, same index and same charge compose the **isotopomer envelope** of an ion.

In a raw spectrum, therefore, there are several envelopes, sometimes overlapping, along with many noise peaks. Usually protein identification tools cannot directly deal with this complexity, so the spectrum has to be deconvoluted.

In **spectrum deconvolution**, the isotopomer envelopes are detected and replaced with the monoisotopic residue mass of the corresponding ions. Furthermore, a relevant amount of noise is removed. Spectrum deconvolution is a delicate operation and a crucial step in protein identification. Among the most important tools for spectrum deconvolution we mention MS-Deconv, Thrash and XTract [11]. In a deconvoluted top-down spectrum, each peak has a position and an intensity. If the peak really derives from a set of ions of the same kind and same index and if the deconvolution was successful, its position is the monoisotopic residue mass of the ions and its intensity depends on the number of ions in the set.

Bibliography

[1] Ruedi Aebersold and Matthias Mann. Mass spectrometry-based proteomics. *Nature*, 422(6928):198–207, 2003.

[2] Neil A. Campbell and Jane B. Reece. *Biology (8th Edition)*. Benjamin Cummings, 8 edition, December 2007.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. McGraw-Hill, Inc., New York, NY, USA, 2002.

[4] A. M. Frank, J. J. Pesavento, C. A. Mizzen, N. L. Kelleher, and P. A. Pevzner. Interpreting top-down mass spectra using spectral alignment. *Anal. Chem.*, 80:2499–2505, Apr 2008.

[5] Benjamin A. Garcia. What does the future hold for top down mass spectrometry? *Journal of the American Society for Mass Spectrometry*, 21(2):193 – 202, 2010.

[6] UCSD CSE Bioinformatics Group. Computational mass spectrometry homepage. `http://proteomics.ucsd.edu/`.

[7] S. Kim, N. Gupta, and P. A. Pevzner. Spectral probabilities and generating functions of tandem mass spectra: a strike against decoy databases. *J. Proteome Res.*, 7:3354–3363, Aug 2008.

[8] Charles F. Kingman. *Regenerative Phenomena*. Wiley, 1972.

[9] Richard D. LeDuc, Gregory K. Taylor, Yong-Bin Kim, Thomas E. Januszyk, Lee H. Bynum, Joseph V. Sola, John S. Garavelli, and Neil L. Kelleher. Prosight ptm: an integrated environment for protein identification and characterization by top-down mass spectrometry. *Nucleic Acids Research*, 32(Web-Server-Issue):340–345, 2004.

[10] Xiaowen Liu, Yuval Inbar, Pieter C. Dorrestein, Colin Wynne, Nathan Edwards, Puneet Souda, Julian P. Whitelegge, Vineet Bafna, and Pavel A. Pevzner. Deconvolution and database search of complex tandem mass spectra of intact proteins: A combinatorial approach. *Molecular & Cellular Proteomics*, 2010.

[11] Xiaowen Liu, Yufeng Shen, Gordon Anderson, Yihsuan S. Tsai, Ying S. Ting, David R. Goodlett, Richard D. Smith, Vineet Bafna, and Pavel A. Pevzner. Protein identification using top-down spectra. (in preparation).

[12] Alessandro Mammana. Public repository for the implemented filtering algorithms. `https://bitbucket.org/alessandromammana/msfilter`.

[13] Fanyu Meng, Benjamin J. Cargile, Leah M. Miller, Andrew J. Forbes, Jeffrey R. Johnson, and Neil L. Kelleher. Informatics and multiplexing of intact protein identification in bacteria and the archaea. 2001.

[14] P. A. Pevzner, V. Dancik, and C. L. Tang. Mutation-tolerant protein identification by mass spectrometry. *J. Comput. Biol.*, 7:777–787, 2000.

[15] Yufeng Shen, Nikola Tolic, Kim K. Hixson, Samuel O. Purvine, Gordon A. Anderson, and Richard D. Smith. De novo sequencing of unique sequence tags for discovery of post-translational modifications of proteins. *Analytical Chemistry*, 80(20):7742–7754, 2008. PMID: 18783246.

[16] Yihsuan Tsai, Alexander Scherl, Jason Shaw, C. MacKay, Scott Shaffer, Patrick Langridge-Smith, and David Goodlett. Precursor ion independent algorithm for top-down shotgun proteomics. *Journal of The American Society for Mass Spectrometry*, 20:2154–2166, 2009. 10.1016/j.jasms.2009.07.024.