

## 13 Proteomics and Mass Spectrometry (Knut Reinert)

The exposition is based on the following sources, which are all recommended reading:

1. de Hofmann, Stroobant: Mass Spectrometry, Principles and Applications, Wiley
2. Perkins et al.: Probability-based protein identification by searching sequence databases using mass spectrometry data, Electrophoresis, 1999, 20, 3551-3567 and MASCOT webpage at [www.matrixscience.com](http://www.matrixscience.com).
3. Bafna, Edwards: SCOPE, a probabilistic model for scoring tandem mass spectra against a peptide database, Bioinformatics, 2001, 17, 1, 13-21

### 13.1 Technology of mass spectrometers

Mass spectrometers are devices that measure the mass of ions relative to their units of charge that is the *mass over charge*  $m/z$ .

They usually contain the following elements:

1. A device to introduce the compound (e.g. a chromatograph) into the analyzer
2. A source to produce ions from the compound
3. One or more analyzers to separate the various ions according to their  $m/z$
4. a detector to count the ions emerging from the last analyzer
5. a computer to process the detector data.

### 13.2 Technology of mass spectrometers



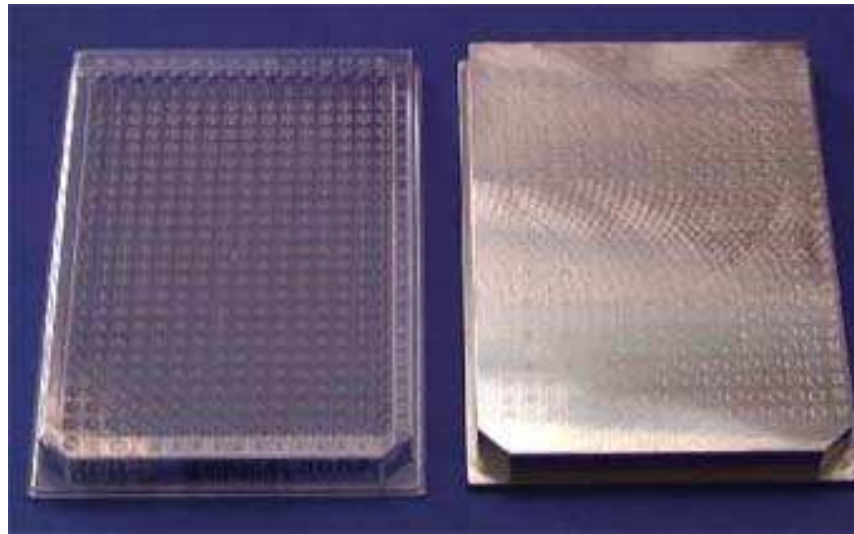
### 13.3 Introducing the analyte

The most common device (for protein MS) to introduce the compound into the analyzer is *HPLC* (high performance liquid chromatography). The peptides are separated while traveling through the chromatography *column* depending e.g. on their hydrophobicity.

The analytes can then directly be subjected to the ionization phase or they can be *spotted* on a *MALDI target* (Matrix Assisted Laser Desorption Ionization).

Then the compound is ionized before entering the analyzer. The two most common devices (for protein MS) to ionize the compounds are *MALDI* and *ESI* (Electrospray Ionization).

### 13.4 Introducing the analyte

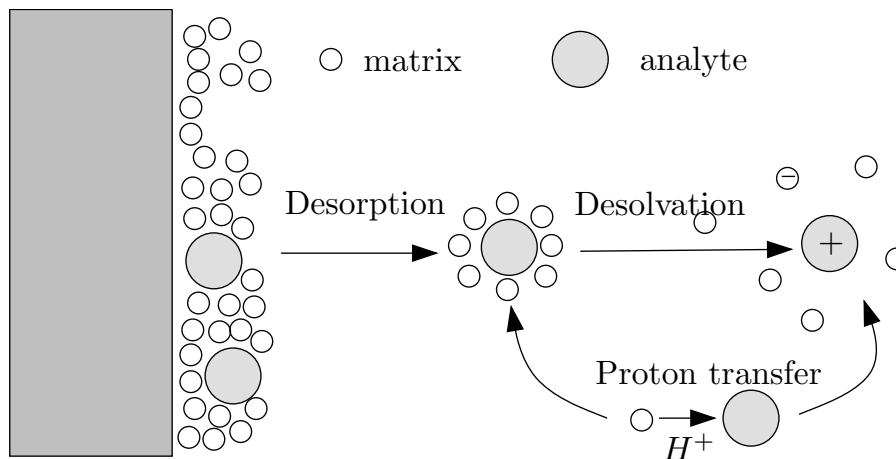


### 13.5 Ionization

In MALDI the compound is mixed in a solvent containing small organic molecules in solution called *matrix*, which has a strong absorption at the laser wavelength.

The mixture is then dried and results in the compound molecules being completely isolated in the matrix. Then laser pulses are shot at the mixture which results in rapid heating of the matrix and its expansion into the gas phase. The analyte molecules are set free (and stay intact) and are ionized.

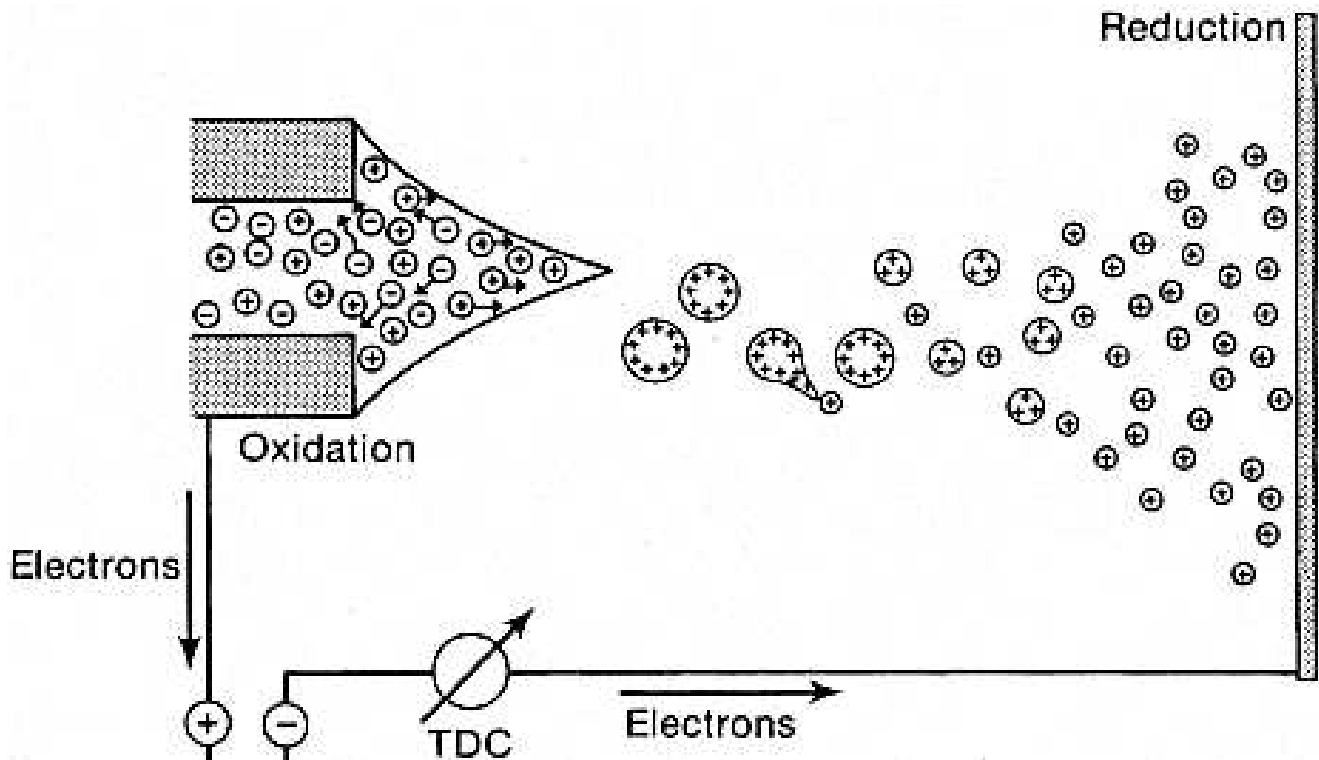
The process is not completely understood, but involves probably gas phase proton transfer. Usually the charge is 1 in MALDI.



In ESI, the compound is ionized at atmospheric pressure which increases the ionization efficiency greatly but poses problems at coupling the ion source to the low pressure analyzer compartment.

An electrospray is produced by applying a strong potential difference (3 – 6kV) to a capillary through which the liquid containing the analyte flows. This results in an electric field of the order of  $10^6 V/m$ . The field causes charge accumulation at the surface of the liquid which will break to form highly charged droplets.

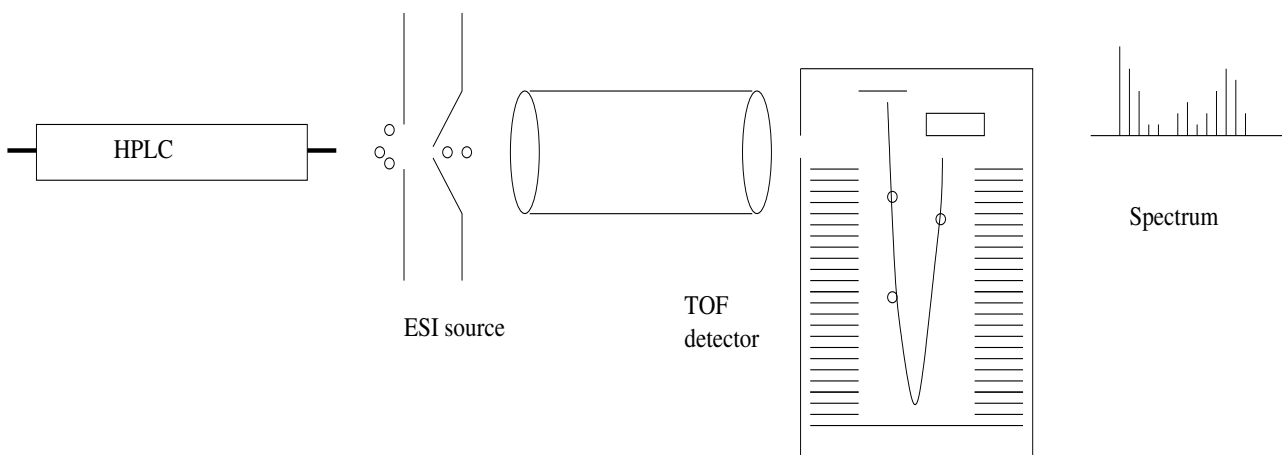
The droplets are then dried leaving one or more (usually less than six) charge units on the analytes.



If MS is used for identifying peptides, both MALDI and ESI are used. If a quantitative measurement is required, ESI has some advantages.

There are various types of mass analyzers like *quadrupole* analyzers, *quadrupole ion trap* analyzers, *time-of-flight* analyzers, *ion cyclotron resonance* and *fourier transform* analyzers etc.

We shortly describe the time-of-flight (TOF) analyzer. Here the ionized analyte with charge  $ze$  ( $e = 1.6 \cdot 10^{-19}$  coulomb) is accelerated by a potential  $V_s$  and flies a distance  $d$  through the chamber (field-free) until it hits the detector. The analyzer measures the time until the ion hits. Then it holds that  $t^2 = \frac{m}{z} \cdot \left(\frac{d^2}{2eV_s}\right)$ . Hence the lower the  $m/z$  of an ion, the faster it reaches the detector.

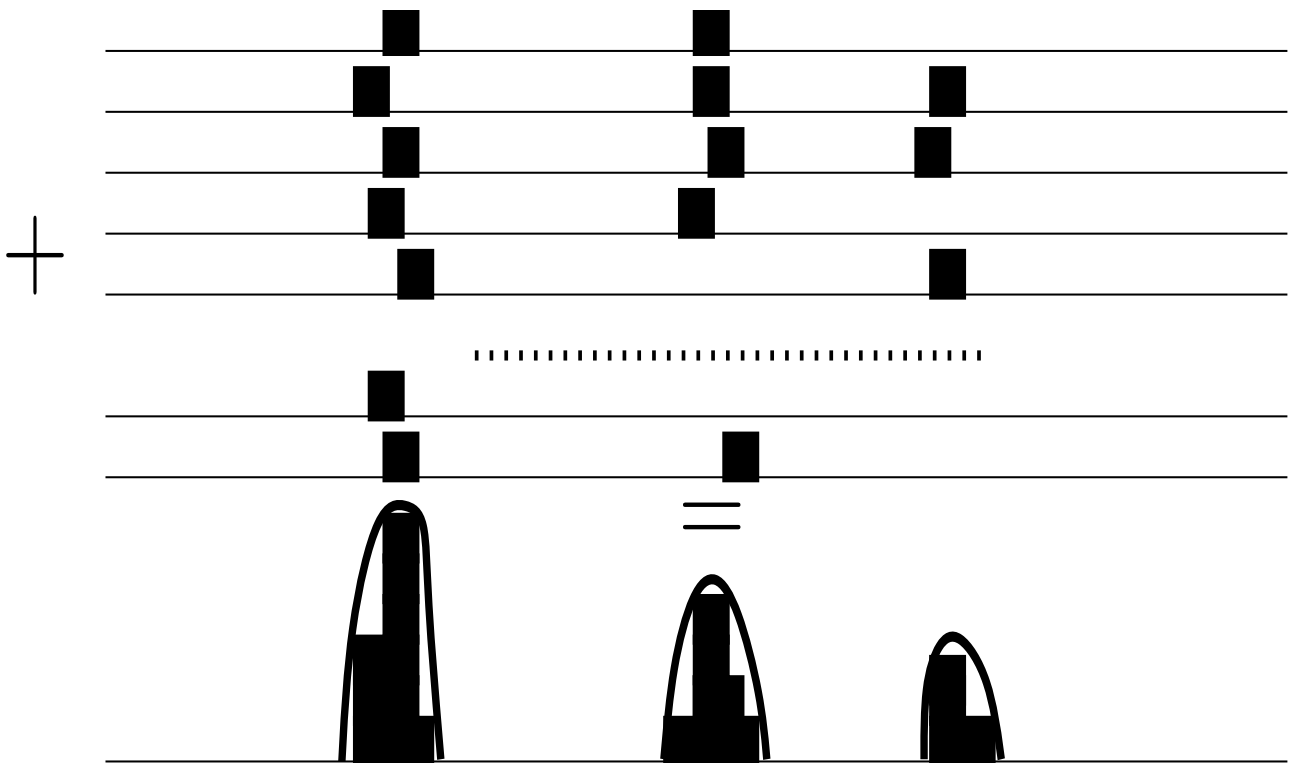


## 13.6 Detectors

As detectors usually *photographic plates*, *faraday cylinders*, or *array detectors* are used in conjunction with electron or photon multipliers to increase the intensity of the signal.

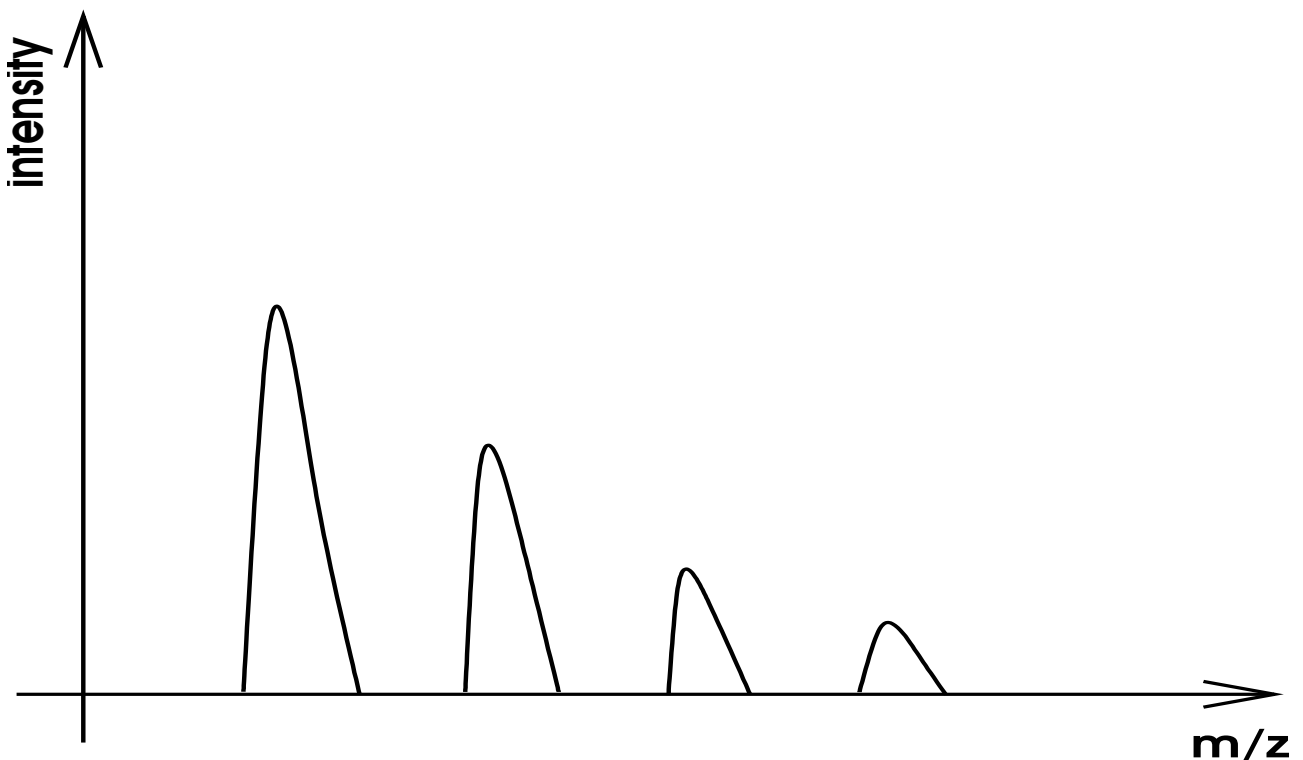
What is interesting for us is, that certain detectors, need after an ion hit some time to recover (called *dead time*) until it can measure the next ion. Hence there is always a *supression effect* that needs to be considered.

The machine software takes with a certain frequency measurements memorizing the time an ion hit (i.e. its  $m/z$  measured in Thomson or Dalton,  $1Da = 1.665402 \cdot 10^{-27} kg$ ) and combines these measurements into a so called *scan* or *spectrum*. These raw spectra are for all practical purposes the "rawest" data we see.

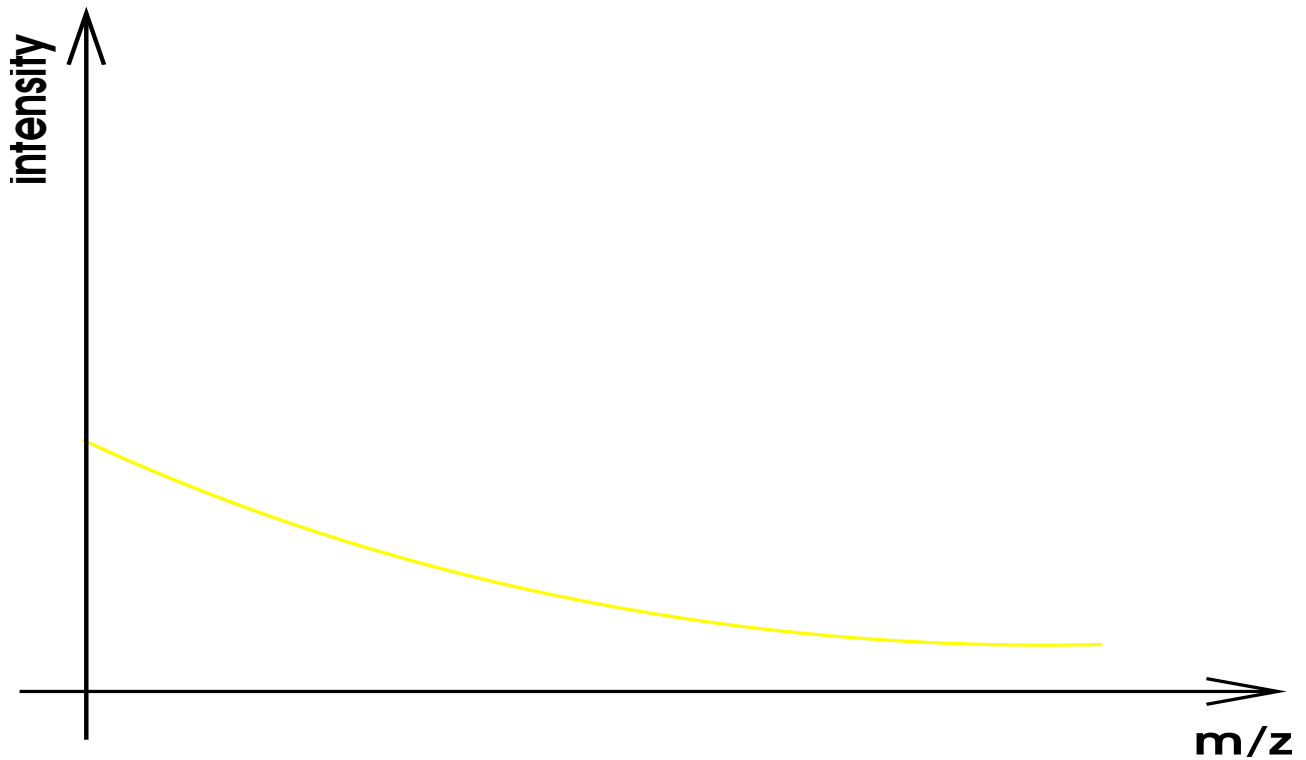


### 13.7 Preprocessing data

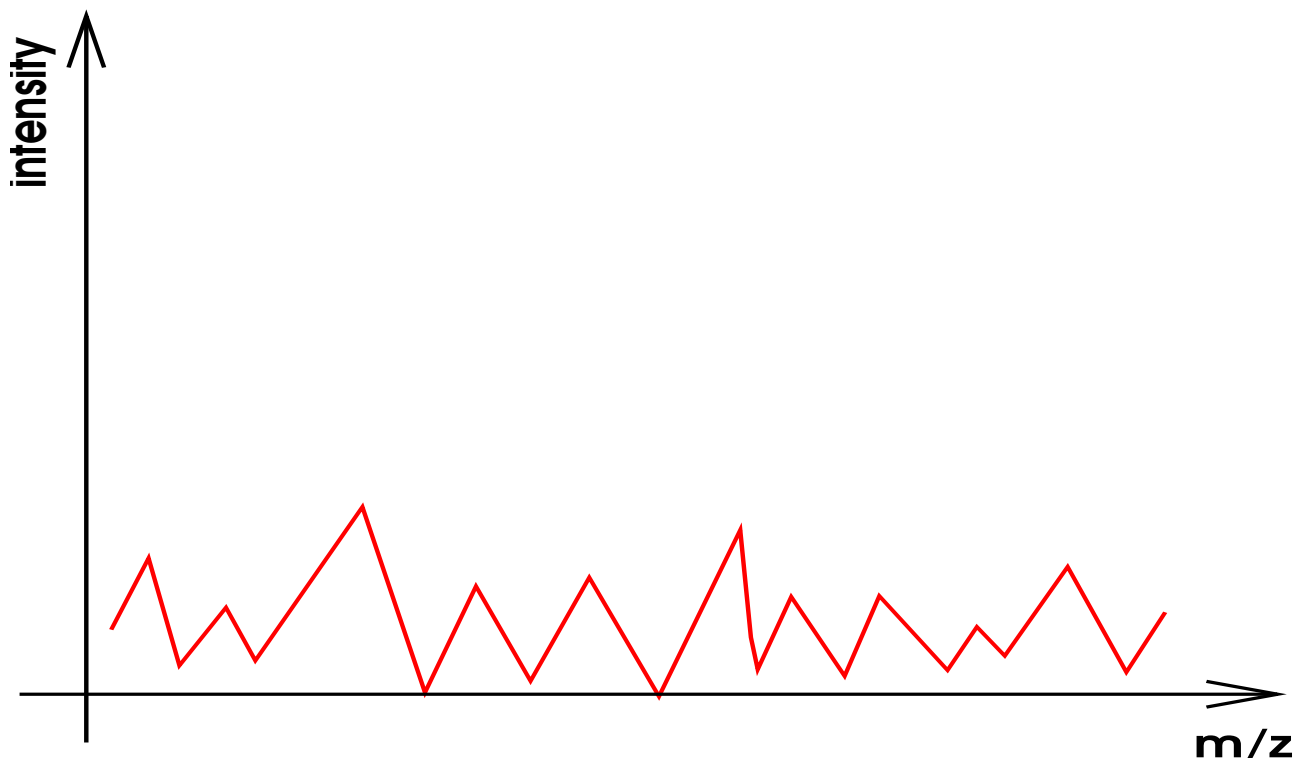
But what does the raw spectrum contain? As we argued it contains the *signal*



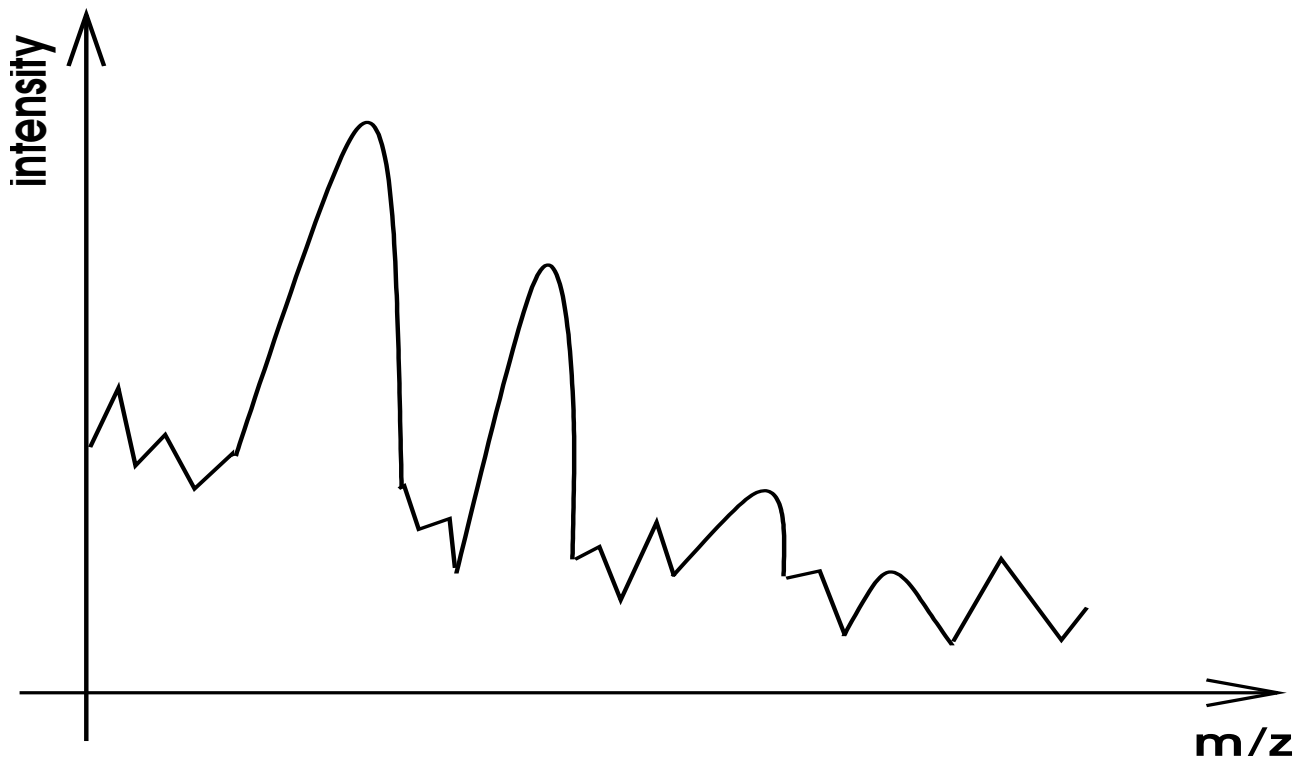
It contains a *baseline* of the spectrum which represents the level of 0 intensity caused by the instrument.



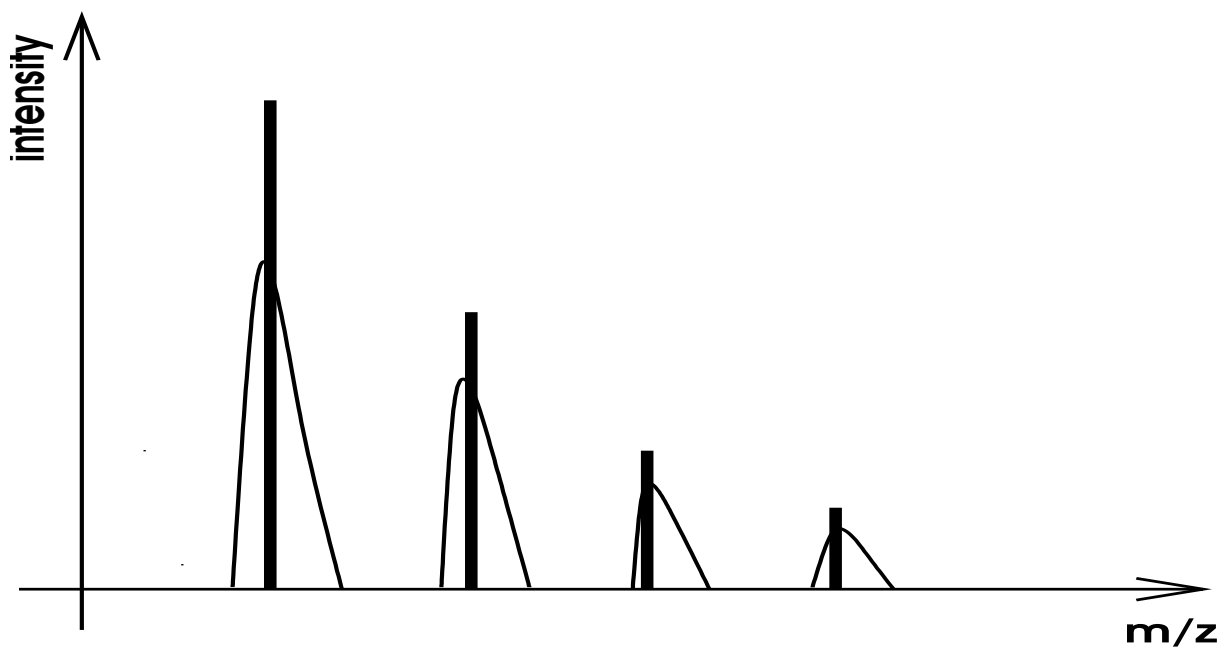
In addition we have some *random* and *chemical* noise.



Adding all these signals together we have the raw spectrum.



The goal is to convert this spectrum into a *stick spectrum* where each stick corresponds to a signal peak, and the baseline and noise are removed.



The conversion of the raw spectrum into a stick spectrum is usually done by the machine software.

This conversion includes the following steps:

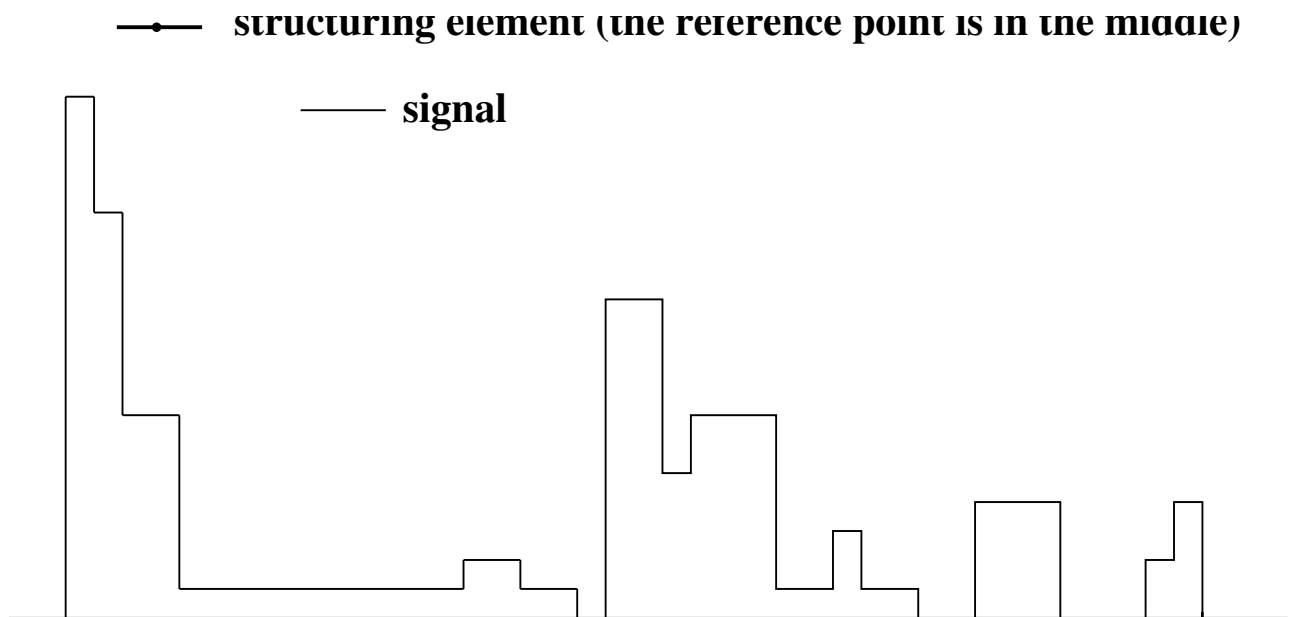
1. Baseline reduction
2. Calibration

3. Peak Picking
4. Stick conversion

We give now a few examples of these conversion steps.

### 13.8 Baseline reduction

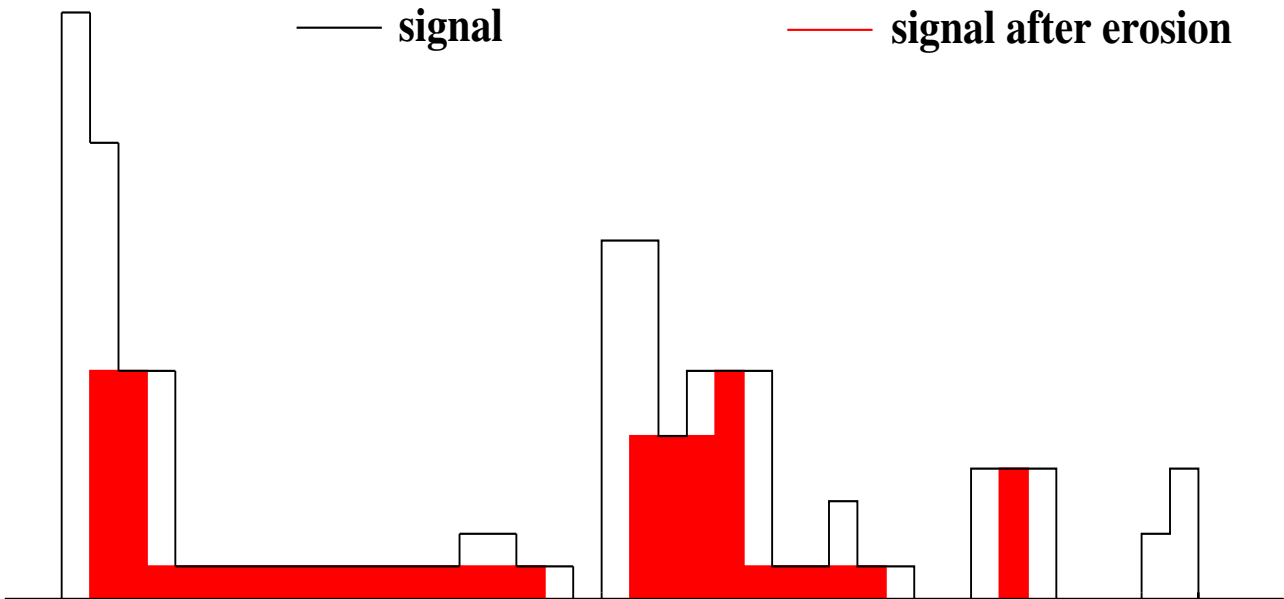
One possible way for eliminating unwanted structures in a signal is to use mathematical morphology, which is the analysis of spatial structures. The basic idea of a morphological filter is to inhibit selected signal structures. Such structures could be noise or some irrelevant signal structures like the baseline.



One basic operation for morphological filter is the *erosion*. Given a set  $X$ , a structuring Element  $B$  with reference point  $y$ , the erosion determines the set of all points  $x$  such that  $B$  centered at  $x$  is completely contained in  $X$ . The erosion is denoted as:

$$[\epsilon_B(X)](x) = \{x \mid B_x \subseteq X\}$$

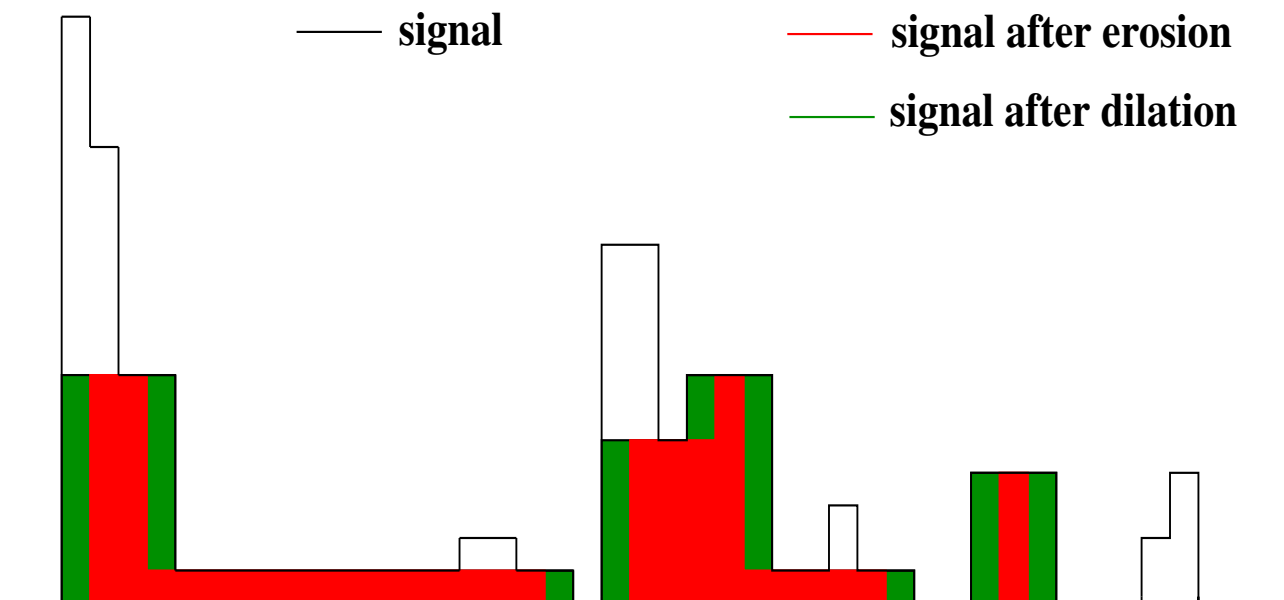
—●— structuring element (the reference point is in the middle)



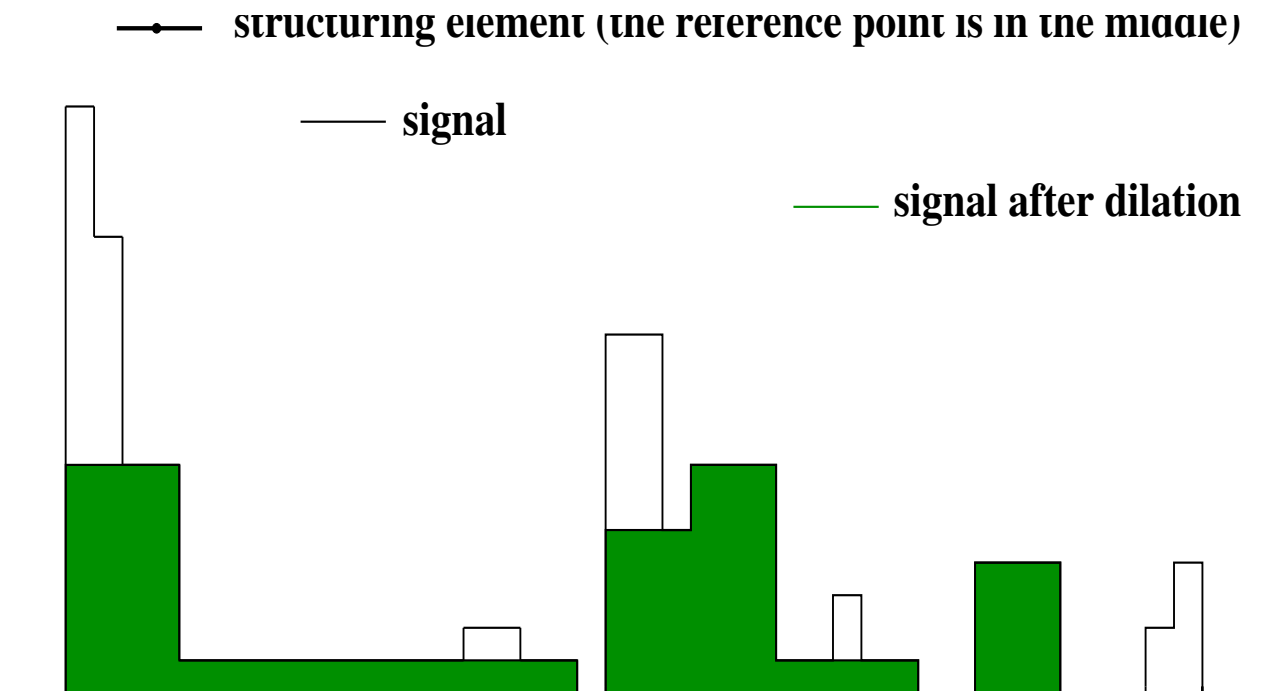
The other basic operation for morphological filter is the *dilatation*. Given a set  $X$ , a structuring Element  $B$  with reference point  $y$ , the dilatation determines the set of all points  $x$  such that  $B$  with reference point  $x$  touches  $X$ . The dilatation is denoted as:

$$[\delta_B(X)](x) = \{x \mid B_x \cap X \neq \emptyset\}$$

—●— structuring element (the reference point is in the middle)

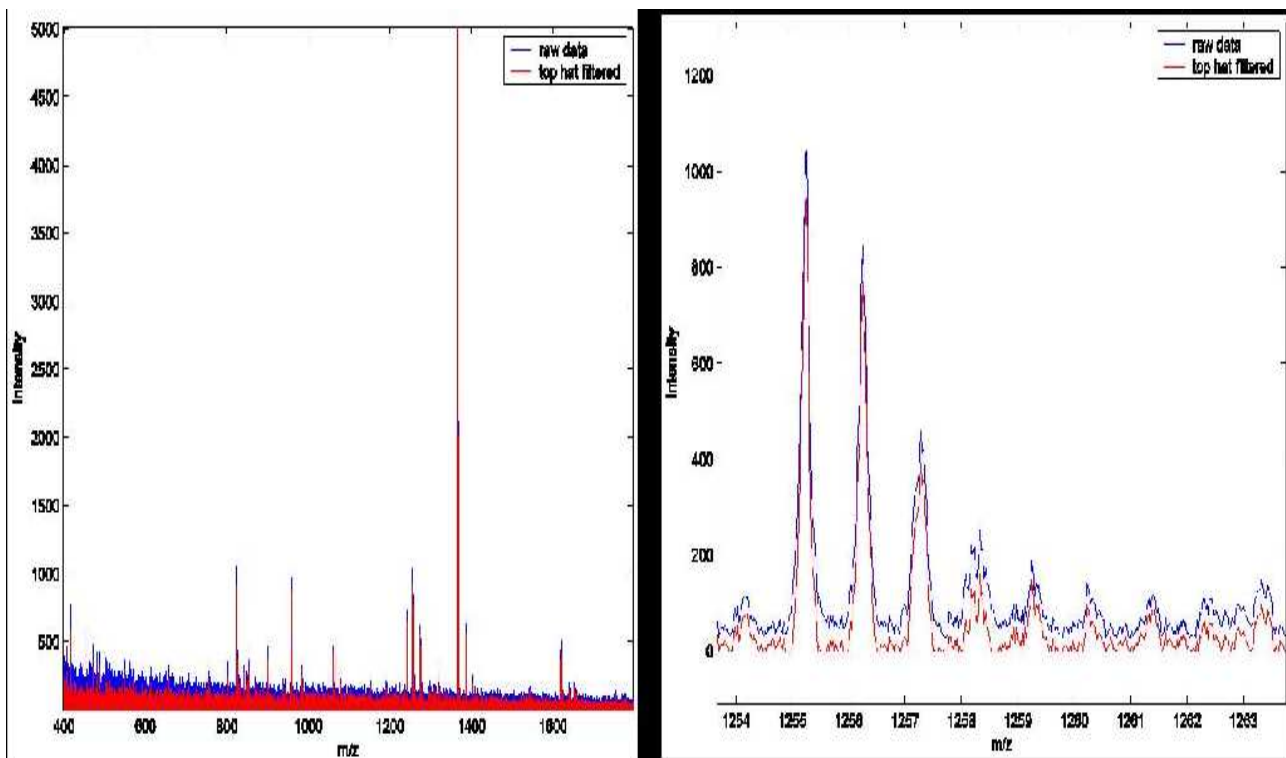


The successive application of erosion and then dilatation is called *opening*. The opening is used by the *top hat filter*  $t$  for removing the baseline of a signal. It is simply the subtraction of a signal  $s$  from its opening. Hence in our example we would subtract the green signal.



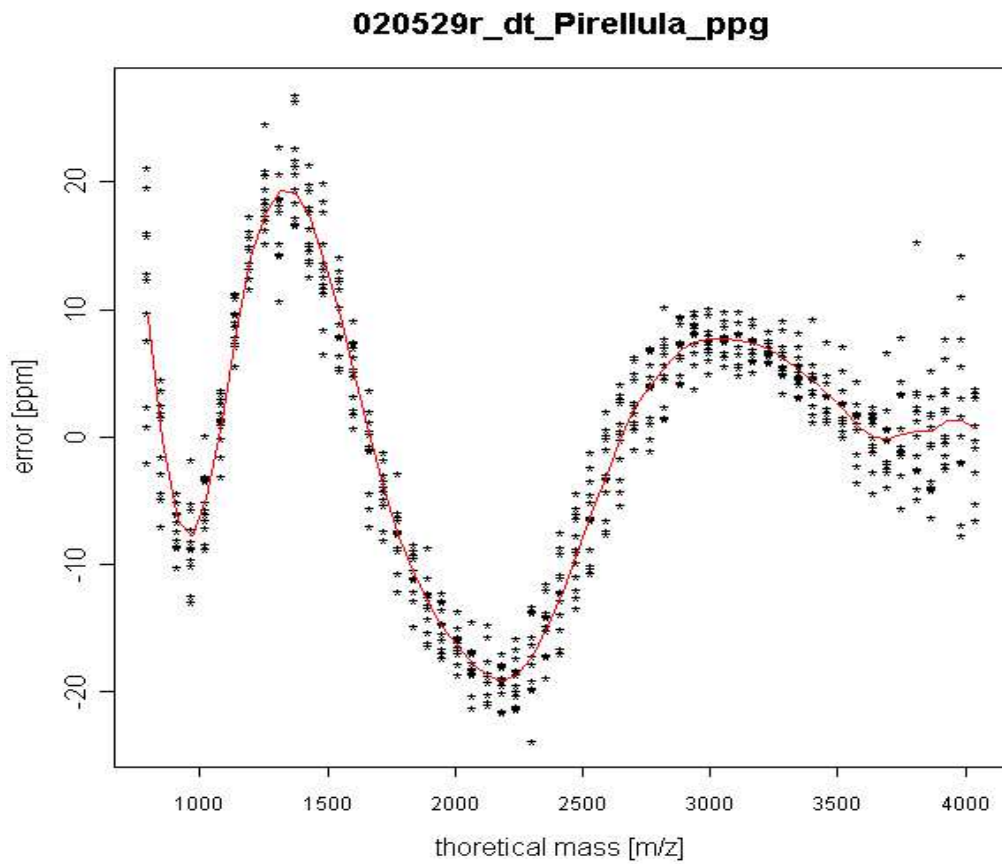
Note that there are many other possible ways to do a baseline reduction.

Here is a picture of real data with a top hat filter applied.



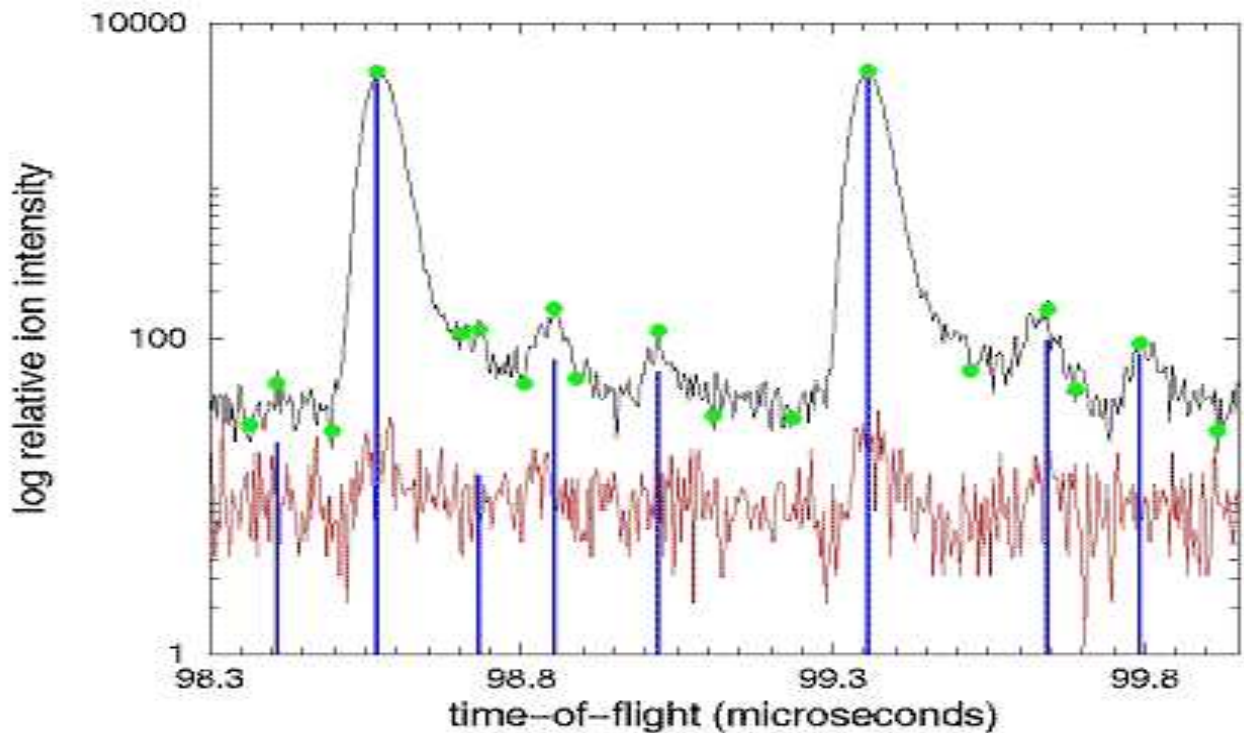
## 13.9 Calibration

Another problem that occurs in the measurement is that the data might be *uncalibrated*, i.e. the peaks have a systematic shift that can often be described as an affine function, but also nonlinear calibration errors are possible. (picture shows PMF data from *Piruella*, taken from Eryk Wolskis website).



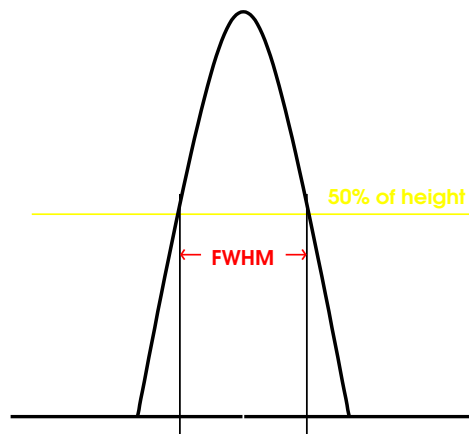
### 13.10 Peak picking

The peak picking routine has now to determine the centroid of the a signal peak and subsequently integrate the area beneath it, since this an approximation of the total ion count.



### 13.11 Stick conversion

Finally once we have chosen a peak we convert it into a stick. The stick represents the most important features of the measurement, the centroid  $m/z$  and the intensity (=area under the peak). Usually the software computes some more characteristic values of the peak and stores them along with the stick, like the height of the peak, its full width at half max (FWHM), etc.



This stick data, is what many algorithms books assume as their input (since this is what the instrument software yields). However, it is of utmost importance to know how this data is derived.

### 13.12 Mass Spectrometry of Proteins

After this short introduction into the many preprocessing steps involved, we now have a closer look at the actual compounds we want to analyze, namely peptides.

Peptides are almost exclusively made out of the elements nitrogen (N), oxygen (O), carbon (C), and hydrogen (H) (the amino acid cystein also contains sulfur (S)).

All these elements occur in nature in different isotopes that differ in their mass. The following table gives all masses of atoms in amino acids and their isotope proportions:

sym	isotopic mass	Rel. %	avg. mass
C	12.000000	98.9	12.011
	13.003355	1.112	
H	1.007825	99.985	1.00794
	2.014	0.015	
N	14.003074	99.63	14.00674
	15.000108	0.37	
O	15.994915	99.76	15.9994
	16.999133	0.04	
	17.999169	0.20	
S	31.972970	95.03	32.066
	32.971456	0.75	
	33.967866	4.22	
	35.967080	0.02	

Given the *atomic composition* of a molecule we know its numbers  $n_c, n_h, n_n, n_o$  and  $n_s$  of C, H, N, O, and S atoms.

### 13.13 Isotope Patterns

The rank  $i$  peak of an isotope-cluster (peak  $r_i$ ) is the peak with  $i$  Da of additional mass compared to the monoisotopic peak. Given the number of C, N, O and S atoms in a peptide, the relative height of  $r_i$  can be computed using a *binomial convolution*.

Consider a list  $L$  of isotope contributors (i. e.  $^{13}\text{C}$ ,  $^{15}\text{N}$ ,  $^{17}\text{O}$ ,  $^{18}\text{O}$ , the influence of S and H isotopes is negligible). For each element  $l \in L$ , let  $p_l$  be the frequency of occurrence of an isotope,  $w_l$  be the integer offset weight of the isotope (1 for  $^{13}\text{C}$ , 2 for  $^{18}\text{O}$ , etc.), and  $n_l$  be the number of atoms.

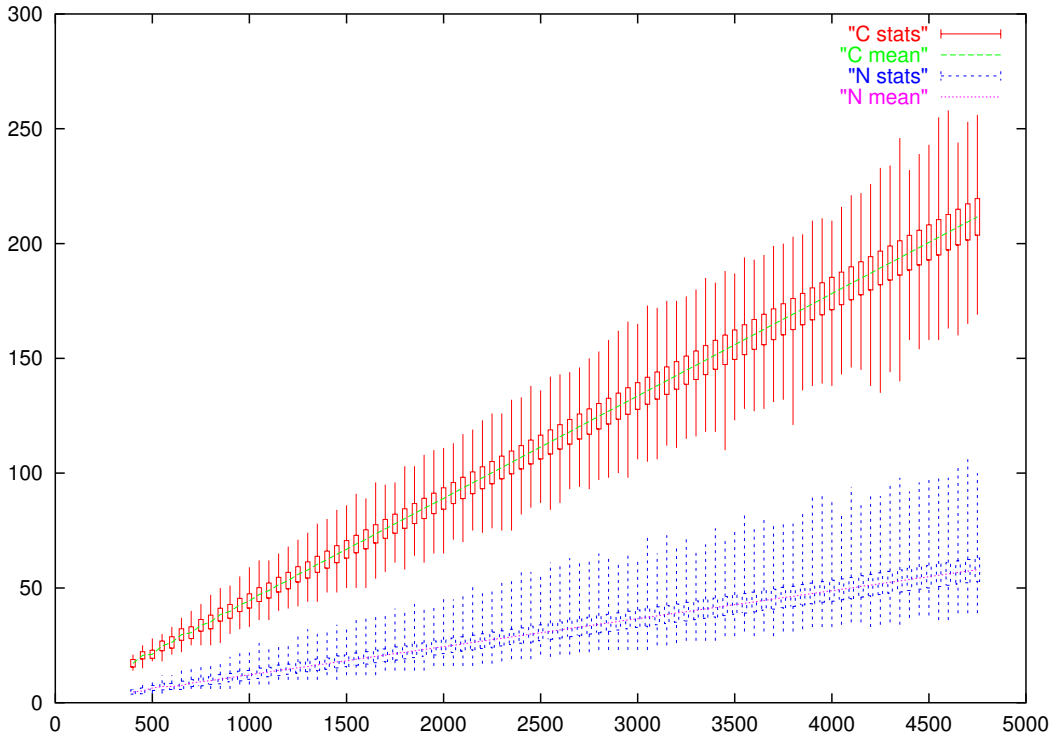
### 13.14 Isotope Patterns

Let  $P$  denote the joint isotope distribution, where  $P(L, k)$  is the probability of seeing peak  $r_k$ , given a list  $L$  of isotope contributors. Then for some  $l \in L$

$$P(L, k) = \sum_{\substack{i=0, \\ i \% w_l = 0}}^k b(n_l, \frac{i}{w_l}, p_l) P(L - \{l\}, k - i). \quad (13.1)$$

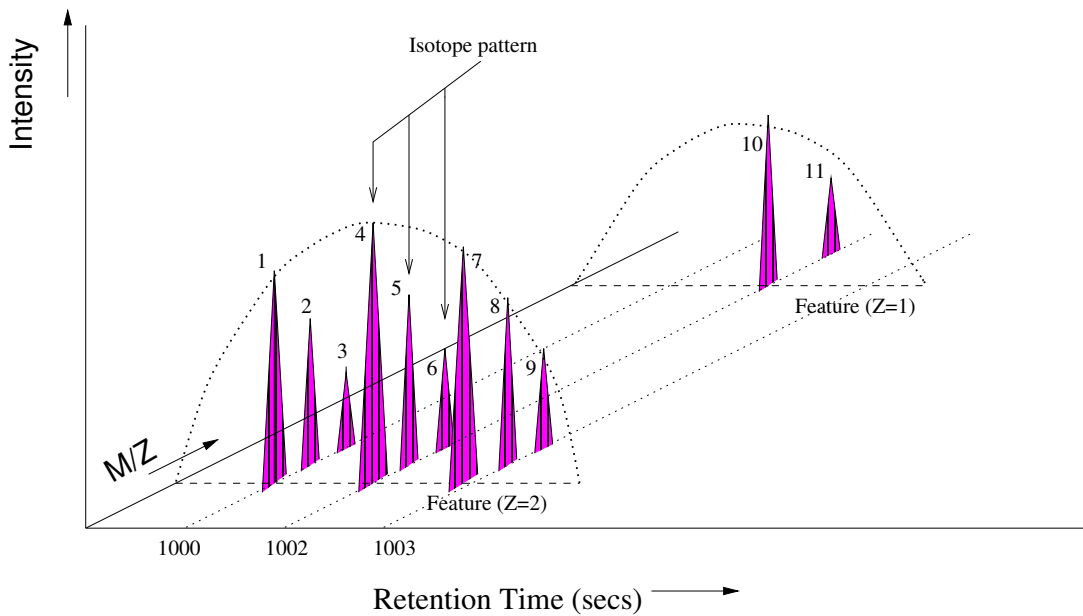
Note that we are making the approximation that the differences in masses between different isotopes are integers. This is not quite true, but in practice we compensate by looking for the peak at difference  $k$  over a range  $(k - \delta, k + \delta)$ .

If one plots the atomic content of proteins in some protein database (e.g. SwissProt) it becomes evident, that the number of atoms for each type grows roughly linearly. The picture show on the x-axis the molecular weight and on the y-axis the number of atoms of a type.

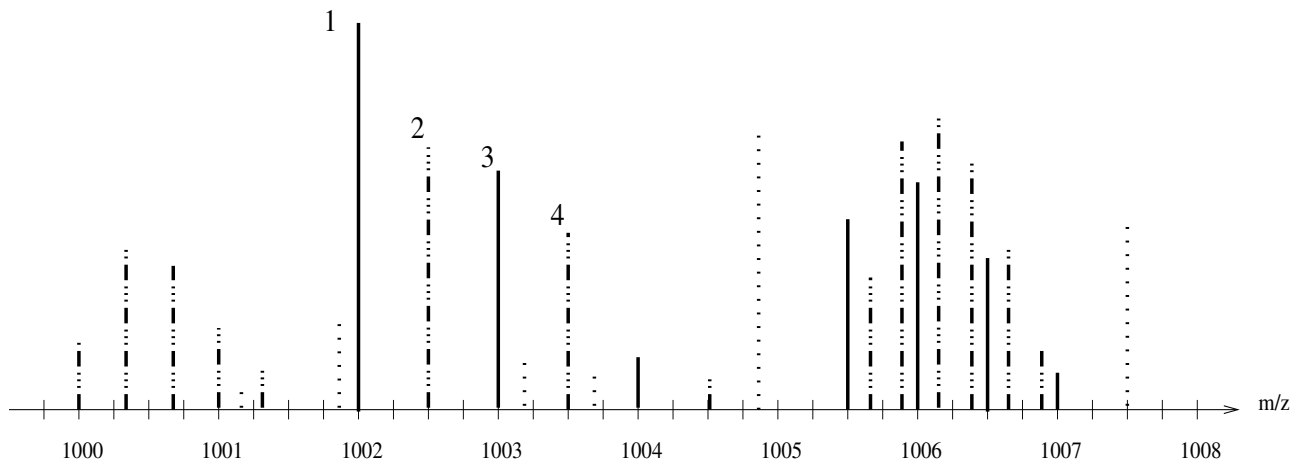


Since the number of C,N, and O atoms grows about linearly with the mass of the molecule it is clear that the isotope pattern changes with mass.

mass	P(k=0)	P(k=1)	p(k=2)	p(k=3)	p(k=4)
1000	0.55	0.30	0.10	0.02	0.00
2000	0.30	0.33	0.21	0.09	0.03
3000	0.17	0.28	0.25	0.15	0.08
4000	0.09	0.20	0.24	0.19	0.12



Note that the isotope pattern is dependent on the mass *not* on the  $m/z$ . It can be used to distinguish peptide content from non-peptide content and to deconvolve the signals.



### 13.15 Analysis of Peptides

There are two dominant analyses of peptides using mass spectrometers:

1. Peptide mass fingerprinting (PMF)
2. MS/MS (also called MS<sup>2</sup>)

In PMF the proteins are *digested* using a restriction enzyme like trypsin (it cuts preferentially after lysin and argenine unless the next amino acid is a proline). The digestion is so specific that it is often possible to identify the protein from this information alone.

In MS/MS a peptide is further fragmented using for example CID (collision induced dissociation).

Although the data derived from PMF and MS/MS experiments has slightly different characteristics, the general approach for using it is similar.

The experimental data are compared with calculated peptide mass or fragment ion mass values, obtained by applying appropriate cleavage rules in the sequence database. Corresponding mass values (and sometimes intensities) are counted or scored in a way that allows the peptide which matches the data best to be identified.

Algorithmically there are two interesting problems:

1. How do we score the data against the theoretical spectrum and how significant is the score?
2. How do we quickly generate theoretical candidate spectra from large protein or transcript databases?

We will give two different answers to the first question. First we describe the MOWSE score, used in the popular MASCOT package ([www.matrixscience.com](http://www.matrixscience.com)), a program package for PMF and MS/MS searches, then we will give a description of the SCOPE algorithm, developed at Celera Genomics.

### 13.16 Peptide Mass Fingerprinting

The input is here a list of masses of the tryptic peptides. For example for human albumin the list of masses contains 49 masses:

```
2917.322 2593.242 2433.263 2404.170 2203.001 2045.095 1915.773
1853.910 1742.894 1623.787 1600.731 1511.842 1386.620 1384.535
1381.533 1342.634 1320.490 1311.741 1257.523 1191.574 1149.615
1024.455 1018.477 2917.322 2593.242 2433.263 2404.170 2203.001
2045.095 1915.773 1853.910 1742.894 1623.787 1600.731 1511.842
```

```

1386.620 1384.535 1381.533 1342.634 1320.490 1311.741 1257.523
1191.574 1149.615 1024.455 1018.477 2917.322 2593.242 2433.263
2404.170 2203.001 2045.095 1915.773 1853.910 1742.894 1623.787
1600.731 1511.842 1386.620 1384.535 1381.533 1342.634 1320.490
1311.741 1257.523 1191.574 1149.615 1024.455 1018.477 1017.536
1013.598 1013.424 1000.603 984.488 960.562 951.441 940.448
etc....

```

Of course proteolysis is not always complete. Steric hindrance, the local context of the cleavage site, or simply insufficient enzyme concentration might lead to one or more miscleavages, that means two peptides that should be digested are still together.

For example, if we allow for two miscleavages in the albumin example, the mass list contains 204 masses.

How would MASCOT score the peaks list? It first computes a MOWSE score and then the probability, that this MOWSE score was achieved by chance. Then the probability  $p$  is converted into the MASCOT score as  $-10 \log p$ . So the lower the probability, the higher the MASCOT score.

MOWSE compares the calculated peptide masses for each entry in the sequence database with the set of experimental data. Each calculated value which falls within a given mass tolerance of an experimental value counts as a match.

Rather than just counting the number of matching peptides, MOWSE uses empirically determined factors to assign a statistical weight to each individual peptide match. The matrix of weighting factors is calculated during the database build stage, as follows:

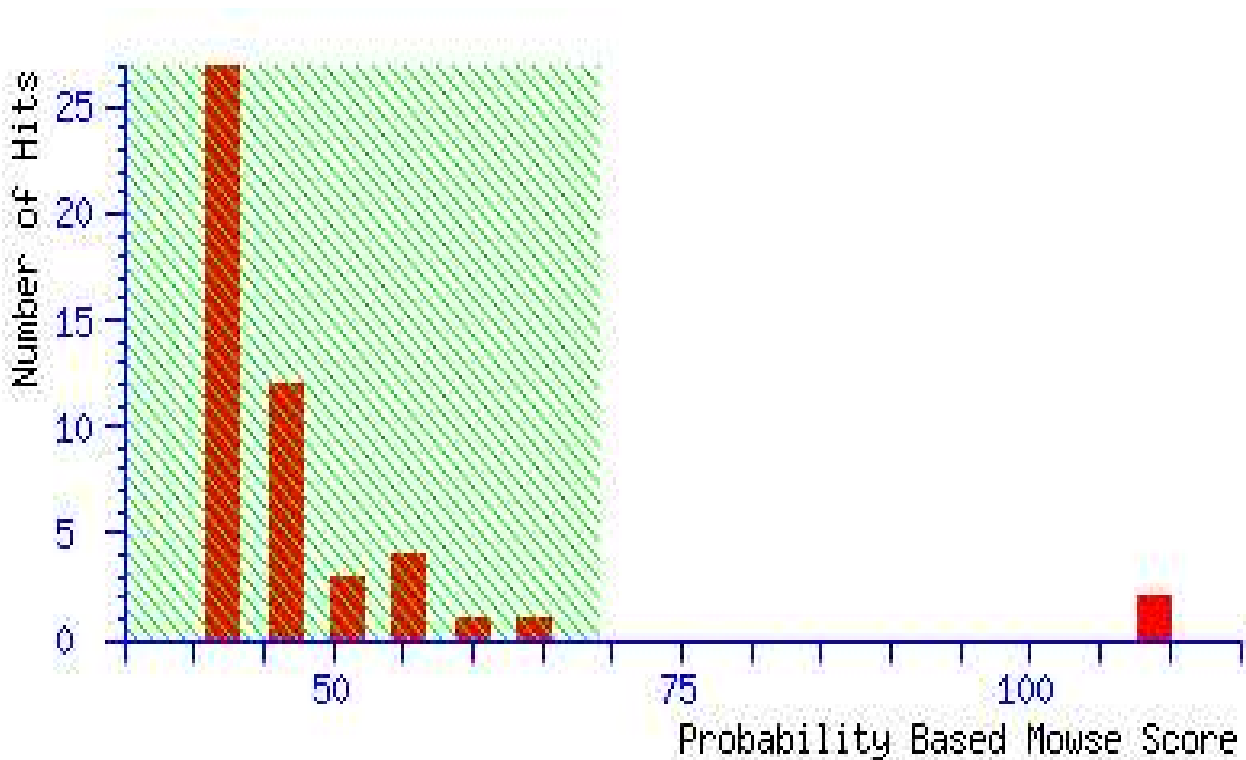
A frequency factor matrix,  $F$ , is created, in which each row represents an interval of 100 Da in peptide mass, and each column an interval of 10 kDa in intact protein mass. As each sequence entry is processed, the appropriate matrix elements  $f_{i,j}$  are incremented so as to accumulate statistics on the size distribution of peptide masses as a function of protein mass. The elements of  $F$  are then normalised by dividing the elements of each 10 kDa column by the largest value in that column to give the MOWSE factor matrix

$$M = (m_{i,j}) = \frac{f_{i,j}}{\max_i f_{i,j}}$$

After searching the experimental mass values against a calculated peptide mass database, the score for each entry is calculated according to:

$$score = \frac{50000}{M_{prot} \prod_n m_{i,j}}$$

Where  $M_{prot}$  is the molecular weight of the entry and the product term is calculated from the MOWSE factor elements for each of the  $n$  matches between the experimental data and peptide masses calculated from the entry. Finally this score is turned into a probability as mentioned before, and its significance is computed.



Scores in the green, shaded region are not significant (at a level of  $p = 5\%$ ).

As mentioned before, the measured stick spectrum is compared against the theoretical spectrum derived from the sequence database.

Unfortunately it is usually not sufficient (both in PMF and MS/MS) only to consider the theoretical digest, even when considering miscleavages. What is generally worse is that the amino acids can occur in *modified form*.

There are the *natural* post translational modifications, such as phosphorylation and glycosylation. There are the accidental modifications which are artefacts of sample handling, such as oxidation. Finally, there are the modifications deliberately introduced during sample work-up, such as cysteine derivatisation.

Hence, it is usually not known beforehand which modifications occur. MASCOT models two modifications, *fixed* and *variable* modifications.

Fixed modifications come at no cost, since the molecular weight of an amino acid is just replaced by its modified weight.

Variable modifications are those which may or may not be present. MASCOT tests all possible arrangements of variable modifications to find the best match. For example, if Oxidation (M) is selected, and a peptide contains 3 methionines, Mascot will test for a match with the experimental data for that peptide containing 0, 1, 2, or 3 oxidised methionine residues. This greatly increases the complexity of a search, resulting in longer search times and reduced specificity, so variable modifications should be used sparingly. (A database of such modifications is *Delta Mass*).

Finally, noise peaks coming from non perfect data processing and chemical noise (contaminants), make the identification difficult. The most common contaminants are keratins (Hair, skin, dandruff).

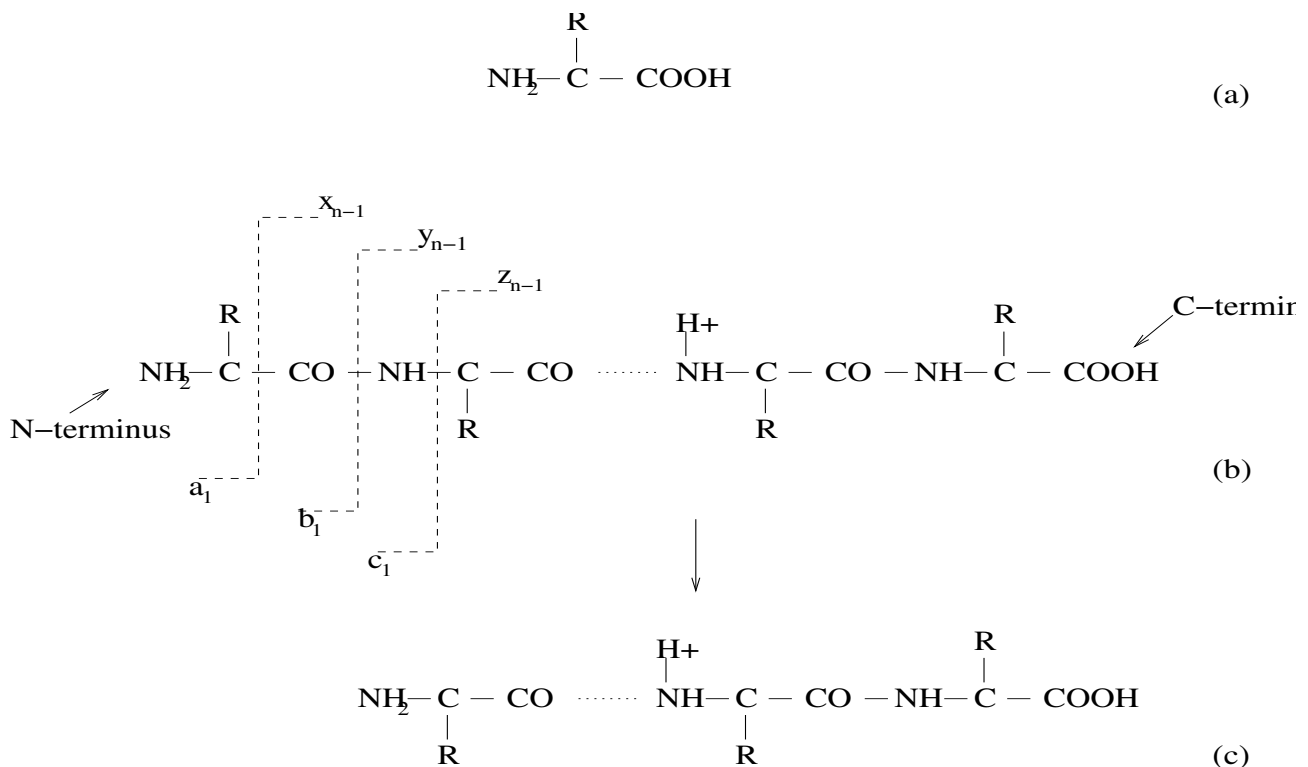
## 14.2 SCOPE

Recall again the structure of a peptide chain, consisting of different amino acids joint by *peptide bonds*. Amino-acids are distinguished from each other by the secondary structure of the side chain R.

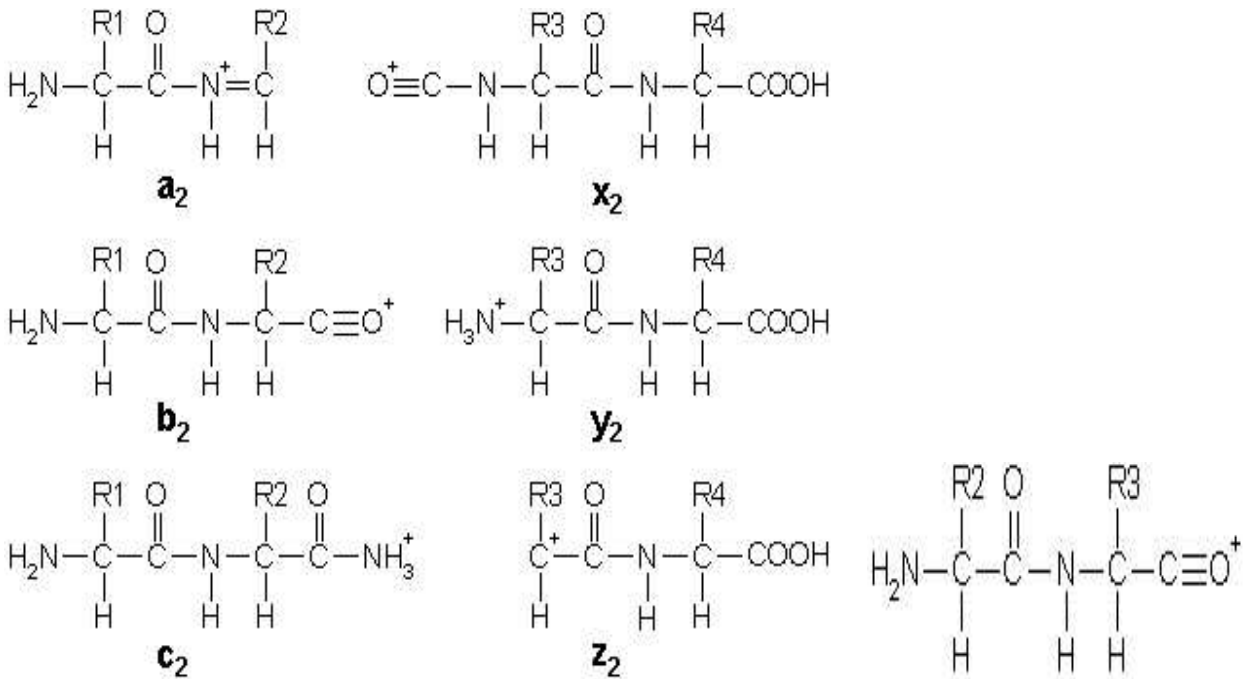
In tandem mass spectrometry (MS/MS) ionized peptides are fragmented by *collision-induced dissociation* (CID). Fragments retaining the ionizing charge after CID have their mass-to-charge ratio measured. Since peptides typically break a peptide-bond when they fragment by CID, the resulting spectrum contains information about the constituent amino-acids of the peptide.

## 14.3 Ion types

The fragmentation of the peptide in CID is a stochastic process governed by the physiochemical properties of the peptide and the energy of collision. The charged fragment can be inferred by the position of the broken bond and the side retaining the charge. In the below figure, the N-terminal  $a_1, b_1, c_1$  fragments, and the C-terminal  $x_{n-1}, y_{n-1}, z_{n-1}$  fragments are shown.



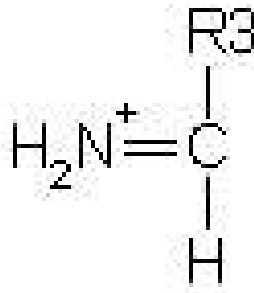
While  $a, b, x$  and  $y$  represent the commonly occurring fragments, a high energy collision often results in other fragments, including *internal* fragments formed by breakage at two points, and fragments formed by breaks in side-chains.



source:www.matrixscience.com

The above pictures show the *a*, *b*, *c*, *x*, *y*, *z* ions as well as an internal ion.

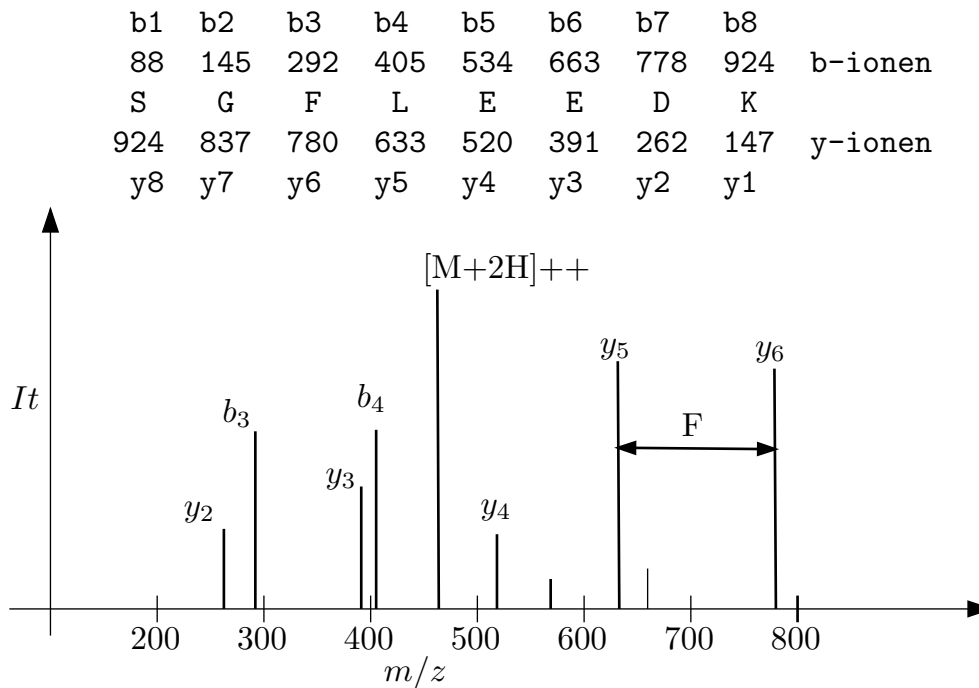
In high energy collision side chain cleavages can occur which helps distinguishing isomers like Leucin and Isoleucin. Another help can be the presence of *immonium* ions that represent a single amino acid (minus the loss of CO and the addition of H).



In any case, the mass difference between ions of the same type is the same and characteristic for amino acid.

## 14.4 MS/MS spectrum

Hence, if we measure in a MS/MS experiment a *MS/MS spectrum* of the peptide fragments and if we can identify the correct ions, we can read off the respective amino acids. A cartoon MS/MS spectrum for the peptide SGFLEEDK is shown in the below figure. The N-terminal *b*<sub>1</sub> ion has the mass of serine (87) plus one Dalton for the terminating H. The *y*<sub>1</sub> ion the mass of lysine (128) plus 19 for OH<sub>3</sub>H.



## 14.5 MS/MS spectrum

Other possible fragments, the presence of multiply charged ions, the absence of some ions in a series, and finally noise and measurement error pose a real challenge for the identification algorithms.

Most algorithms for analyzing MS/MS data address the following three modules.

## 14.6 Modules of MS/MS algorithms

**Interpretation:** The *input* is a *MS/MS spectrum*, the *output* is *interpreted-MS/MS-data*. Interpreted-MS/MS-data may include parent peptide mass, partial or complete sequence tags, and combinations of sequence tags and molecular masses.

**Filtering:** The *input* is *interpreted-MS/MS-data* and a peptide sequence database. The *output* is a list of *candidate-peptides* that might have generated the MS/MS spectrum.

**Scoring:** The *input* is a list of *candidate-peptides* and the *MS/MS spectrum*. The *output* is a ranking of the *candidate-peptides* along with a score and possibly a *p*-value (probability that the score was achieved by random chance).

We focus on the scoring of peptides against theoretical spectra. That means we want to answer the question:

Given a peptide *p* how likely is it that it generated the observed spectrum?

SCOPE addresses the following points in a novel way:

1. it models explicitly fragmentation depending on the peptide and experimental setting
2. it models explicitly measurement error
3. it models noise peaks

The SCOPE algorithm models the process of MS/MS spectrum generation by a two-step stochastic process.

1. The first step involves generation of fragments from a peptide, according to a probability distribution estimated from many training samples.
2. The second step involves the generation of a spectrum from the fragments according to the distribution of the instrument measurement error.

## 14.7 Definitions

We introduce now some terminology.

**MS/MS Spectrum:** A MS/MS spectrum  $S \in \mathbf{R}_+^k$  is a vector of positive real numbers specifying the  $k$  observed mass-charge ratios of the spectral peaks.

**Peptide:** A peptide  $p \in \mathcal{A}^n$  is a sequence of  $n$  amino-acid residues from the alphabet of amino-acid symbols,  $\mathcal{A} = A, C, \dots, Y$ .

**Fragment Space:** An enumeration  $\mathcal{F}(p)$  of all fragment mass-charge ratios that a peptide  $p$  might produce. Each element of  $\mathcal{F}(p)$ , then, is a fragment-charge state pair. Thus,

$$\mathcal{F}(p) = \{(a_1, i), (b_1, i), (y_1, i), \dots, \\ (a_n, i), (b_n, i), (y_n, i), i = 1, 2, 3, \dots\}$$

Denote the mass-charge ratio of a fragment  $f \in \mathcal{F}(p)$  by  $(m/z)(f)$ .

**Fragmentation Space:** The fragmentation space  $\phi(p)$  of a peptide  $p$  is the set of all fragmentation patterns of  $p$ . That is,

$$\phi(p) = \{F : F \subseteq \mathcal{F}(p)\}$$

**Noise:** We consider any peak of  $S$  for which  $F(p)$  provides no explanation to be a noise peak.

## 14.8 The two step process

**Fragmentation:** Each of the many copies of a peptide  $p$  that pass into the secondary collision chamber fragments according to the experimental conditions and the nature of the peptide. Depending on the experimental conditions and the physical and chemical properties of the peptide we observe a particular fragment with a certain probability.

In addition to the modeled fragments in  $\mathcal{F}(p)$ , unexpected fragments or contaminant fragments might be observed. A fragmentation pattern  $F$  is the outcome of this random experiment on the fragmentation space  $\phi(p)$ . We will model the noise peaks later.

**Measurement:** Each fragment with a particular mass-charge ratio generates a mass-charge ratio observation close to, but not precisely at its true mass-charge ratio. The observation of many fragments with the same mass-charge ratio leads to the formation of a distinctive peak close to the true mass-charge ratio of these fragments.

The observed peak can then be represented by a single real number, an estimate of the true mass-charge ratio of the fragments that generated it. The deviation of this mass-charge ratio of a peak from its true value is modeled according to a probability distribution, typically the normal distribution.

## 14.9 Scoring spectra

Let  $\psi(S | p)$  denote the probability density function for the random vector  $S$  representing the MS/MS spectrum, given peptide  $p$ . Typically, we are searching a database for the peptide  $p^*$  that satisfies

$$p^* = \arg \max_p \psi(S | p)$$

A formal description of the two-step model of fragmentation followed by measurement is given by:

$$\psi(S | p) = \sum_{F \subseteq \mathcal{F}(p)} \psi(S | F, p) \Pr(F | p)$$

The quantity  $\Pr(F | p)$  represents the probability of a particular fragmentation pattern of a peptide. It is in the computation of  $\Pr(F | p)$  that the complex process of fragmentation can be modeled.

## 14.10 Fragmentation probability estimation

The SCOPE algorithm does not explicitly implement an automatic algorithm to estimate the probabilities  $\Pr(F | p)$  but relies on the judgement of the user.

For example, experienced operators know that the presence of acidic amino-acids in a peptide makes the neutral water loss ion type cleavages much more likely.

In general those probabilities could also be learned from sample spectra of known peptides given a specific experimental setup.

## 14.11 Computing $\psi(S | F, p)$

$\psi(S | F, p)$  describes the probability of observing a collection of spectral peaks, given a particular fragmentation pattern of a peptide  $p$ .

Unfortunately, it is not at all obvious which fragment(s) are responsible for which peak(s), and which peaks should be considered noise. In order to compute  $\psi(S | F, p)$ , we need to either sum over all the possible explanations of each peak which is not feasible or use our understanding of the mass spectrometer to limit the number of terms.

We assume the following:

1. Each unique mass-charge ratio in the fragment space generates at most one spectral peak.
2. Each spectral peak is the observed mass-charge ratio of at most one of the (unique) mass-charge ratios in the fragment space.
3. The assignment of spectral peaks to fragments must be *non-crossing*. For all fragments  $f_1, f_2$  and spectral peaks  $S_1, S_2$ , if  $(m/z)(f_1) < (m/z)(f_2)$  and  $S_1 < S_2$ , then peak  $S_1$  must have been generated by fragment  $f_1$  and peak  $S_2$  must have been generated by fragment  $f_2$ .

In addition, we augment the fragment space  $\mathcal{F}(p)$  with *noise fragments*, one for each spectral peak. Each noise fragment has the same mass-charge ratio as its spectral peak. We denote this augmented fragment space  $\mathcal{F}'(p)$  and the corresponding fragmentation space  $\phi'(p)$ .

Due to the addition of noise fragments all spectral peaks must either be assigned to a unique fragment from our original fragment space  $\mathcal{F}(p)$  or to a noise fragment. Therefore we can make the following

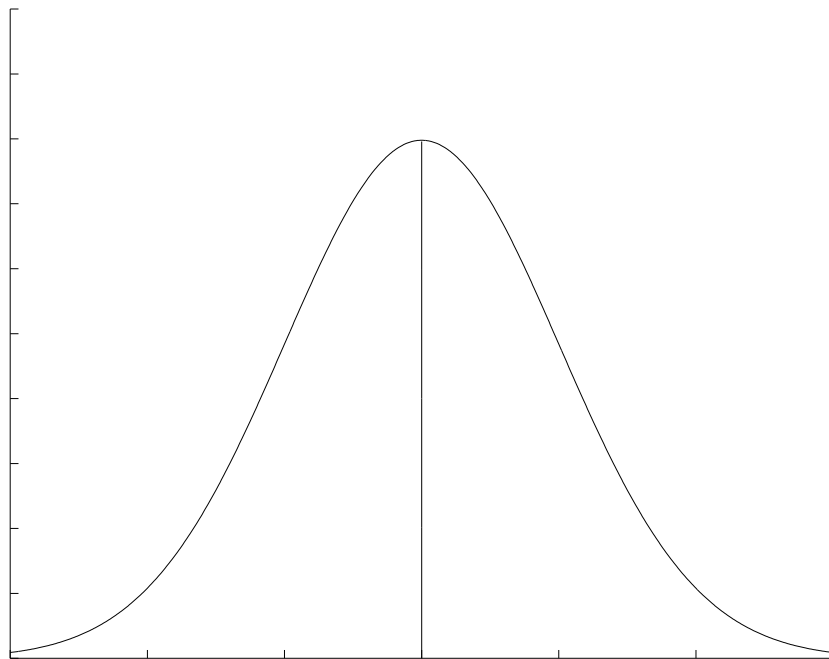
**Observation 1.** Only fragmentation patterns  $F \subseteq \mathcal{F}'(p)$  with  $|F| = k$  have non-zero probability mass.

However, we can say something even stronger. Let  $S_i \stackrel{M}{=} f$  denote the event that peak  $S_i$  is generated by fragment  $f$ , and  $S = (S_1, S_2, \dots, S_k)$  be a tandem MS spectrum ordered by mass-charge ratio. Further, let  $F \subseteq \mathcal{F}(p)$ ,  $|F| = k$  be an arbitrary fragmentation pattern, whose observed fragments  $f_1, f_2, \dots, f_k \in F$  are ordered by mass-charge ratio.

Then only one assignment of spectral peaks to fragments has non-zero probability mass. All of the probability mass for  $\psi(S | F, p)$  is captured by this unique non-crossing assignment. We write:

$$\psi(S | F, p) = \psi(S | \cap_{i=1}^k [S_i \stackrel{M}{=} f_i], F, p)$$

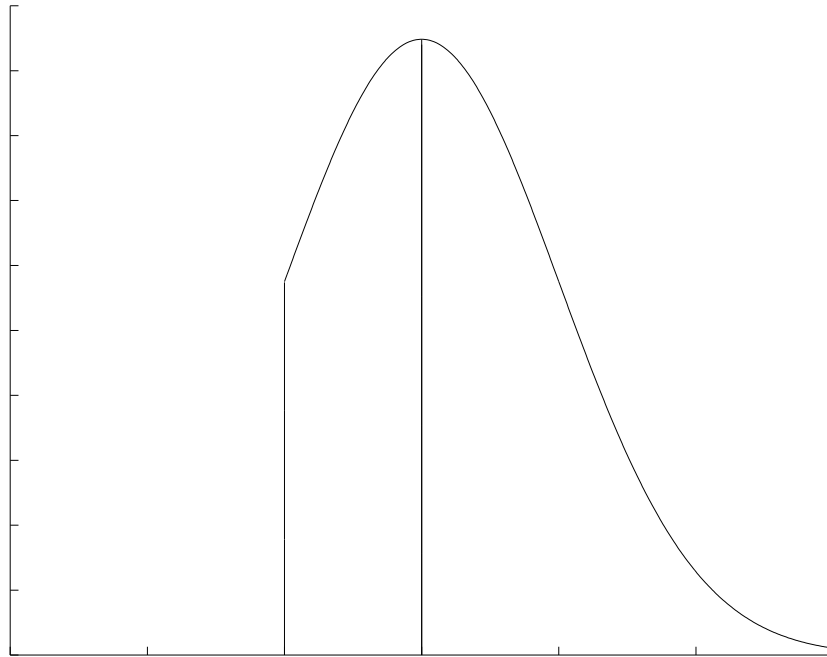
In isolation, the distribution of one measured mass-charge ratio about its true value is independent of any other measured mass-charge ratio about its true value. We model the distribution of the measured mass-charge ratios as normal distributions centered at the fragment mass-charge ratio and the distribution of the measured mass-charge ratio of noise fragments by an impulse function at the mass-charge ratio of its spectral peak.



We expand  $\psi(S | \cap_{i=1}^k [S_i \stackrel{M}{=} f_i], F, p)$  into its components in order to compute it.

$$\begin{aligned} & \psi(S | \cap_{i=1}^k [S_i \stackrel{M}{=} f_i], F, p) \\ &= \psi(S_1 | \cap_{i=1}^k [S_i \stackrel{M}{=} f_i], F, p) \times \\ & \quad \prod_{j=2}^k \psi(S_j | S_1, \dots, S_{j-1}, \cap_{i=1}^k [S_i \stackrel{M}{=} f_i], F, p) \\ &= \psi(S_1 | [S_1 \stackrel{M}{=} f_1], F, p) \times \\ & \quad \prod_{j=2}^k \psi(S_j | S_{j-1}, [S_j \stackrel{M}{=} f_j], F, p). \end{aligned}$$

In the last term  $S_j$  is dependent on the previous  $S_i$  for  $i < j$ . To simplify this we truncate the left-hand tail of the measurement distribution and rescale its total probability density to one.



Let  $\rho_f$  be the distribution of the observed peak about the true mass-charge ratio of fragment  $f$ . Then

$$\begin{aligned} \psi(S_1 \mid [S_1 \stackrel{M}{=} f_1], F, p) &= \rho_{f_1}(S_1) \\ \psi(S_j \mid S_{j-1}, [S_j \stackrel{M}{=} f_j], F, p) &= \begin{cases} \frac{\rho_{f_j}(S_j)}{\int_{S > S_{j-1}} \rho_{f_j}(S)}, & S_j > S_{j-1}; \\ 0, & \text{otherwise.} \end{cases} \end{aligned}$$

## 14.12 Computing $\psi(S \mid p)$

We now show how this choice of  $\psi$  allows an efficient algorithm for computing  $\psi(S \mid p)$ .

We want to avoid the computation of an exponential number of terms in the expression  $\psi(S \mid p) = \sum_{F \subseteq \mathcal{F}(p)} \psi(S \mid F, p) \Pr(F \mid p)$ . To do this we need another assumption, namely that the probability of observing  $f$  must be independent of the observation of other fragments. This is not always true but allows us to compute  $\psi(S \mid p)$  efficiently with dynamic programming.

Given the spectrum  $S = (S_1, \dots, S_k)$  and the fragments  $\mathcal{F}'(p) = \{f_1, \dots, f_m\}$  ordered by mass-charge ratio, we define  $\mathcal{F}'_j(p) = \{f_1, \dots, f_j\}$  to be the first  $j$  fragments of  $\mathcal{F}'(p)$ .

## 14.13 Computing $\psi(S \mid p)$

Then the dynamic programming recurrence function  $\Phi(i, j)$  represents the probability mass associated with the event that the first  $i$  peaks were generated by  $i$  fragments from the first  $j$  fragments of  $\mathcal{F}'(p)$ . Clearly,  $\Phi(k, m) = \psi(S \mid p)$  is the value we are interested in. The following recurrence holds:

$$\Phi(i, j) = \begin{cases} 1, & \text{if } i = 0, \\ 0, & \text{if } i > j, \\ \Phi(i-1, j-1) \\ \quad \times \psi(S_i \mid S_{i-1}, S_i \stackrel{M}{=} f_j) \\ \quad \times \Pr(f_j \mid p) \\ + \Phi(i, j-1) \Pr(\overline{f_j} \mid p), & \text{otherwise.} \end{cases}$$

### 14.14 Computing $\psi(S | p)$

The above recursion corresponds to a special sequence alignment problem. We align the spectrum with all possible fragments in the augmented fragment space. The first term in the sum is for the case that the  $j$ -th fragment is assigned to a spectral peak. The probability of that is the probability of assigning the first  $i - 1$  spectral peaks to  $i - 1$  fragments among the first  $j - 1$  fragments times the probability that  $S_i$  is assigned  $f_j$  times the probability of  $f_j$  given  $p$ .

The second term in the sum describes the fact that  $f_j$  is not assigned to any peak in  $S$ . This might be large if we do not expect a fragment to occur!

### 14.15 Computing $\psi(S | p)$

The most likely assignment  $F^*$  is given by

$$F^* = \arg \max_{F \subseteq \mathcal{F}'(p)} \psi(S | F, p) \Pr(F | p)$$

In practice we need to give each spectral peak a small width, to give it a non-zero probability, and then report the score as  $-\log(p)$ .

### 14.16 Generating peptide candidates from a database

So far we only talked about (a specific) the scoring scheme used to rank a list of candidate peptides. But how do we derive such a list?

Of course we could take an entire protein or transcript database, but this would be very time consuming. We can make use of some information at hand to derive a shorter candidate list. The most useful piece of information is the *parent mass* of the peptide that was subjected to a secondary fragmentation. Or more precisely, a window of about  $2 Da$  around the parent mass.

We define now the general problem more formally:

### 14.17 Peptide Candidate Generation with Integer Weights [PCG(Int)]

Given a string  $\sigma$  of length  $n$  over an alphabet  $\mathcal{A}$  of size  $m$ ; a positive integer mass  $\mu(a)$  for each  $a \in \mathcal{A}$ ; and  $k$  integer mass queries  $M_1, \dots, M_k$ , enumerate all (distinct) pairs  $(i, \omega)$ , where  $1 \leq i \leq k$  and  $\omega$  is a substring of  $\sigma$ , such that

$$\sum_{j=1}^{|\omega|} \mu(\omega_j) = M_i.$$

For convenience, we denote the maximum relevant substring mass by  $M_{\max} = \max_i M_i$  and the minimum relevant substring mass by  $M_{\min} = \min_i M_i$ . Further, since the set of query masses may contain repeated values, we define  $R_{\max}$  to be the maximum number of repeats.

### 14.18 Peptide Candidate Generation with Real Weights [PCG(Real)]

Given a string  $\sigma$  of length  $n$  over an alphabet  $\mathcal{A}$  of size  $m$ ; a positive real mass  $\mu(a)$  for each  $a \in \mathcal{A}$ ; and  $k$  positive real mass queries with positive lower and upper tolerances  $(M_1, l_1, u_1), \dots, (M_k, l_k, u_k)$ , enumerate

all (distinct) pairs  $(i, \omega)$ , where  $1 \leq i \leq k$  and  $\omega$  is a substring of  $\sigma$ , such that

$$M_i - l_i \leq \sum_{j=1}^{|\omega|} \mu(\omega_j) \leq M_i + u_i.$$

We define  $M_{\max}$  and  $M_{\min}$  to be the minimum and maximum relevant substring mass,  $M_{\max} = \max_i(M_i + u_i)$  and  $M_{\min} = \min_i(M_i - l_i)$  and define  $O_{\max}$  to be the maximum number of  $[M_i - l_i, M_i + u_i]$  intervals to overlap any mass.

## 14.19 Issues to take into account

1. Make sure that peptide sequences do not straddle protein boundaries.
2. The search can be refined by requiring sequences with correct digestion patterns (we need the peptide context for this).
3. We need to deal with redundancy in the database.
4. We need to consider posttranslational modifications.

## 14.20 Some numbers

We consider amino-acid sequence databases of a few megabytes to a few gigabytes of sequence, over an alphabet of size 20. The alphabet is small enough that we can look up the mass table for each symbol in the alphabet in constant time.

A typical set of tandem mass spectra from a single high throughput run consists of hundreds to thousands of spectra, so we expect hundreds to tens of thousands of distinct query masses. Furthermore, these query masses are typically constrained between 600-3000 Daltons with a 2 Dalton tolerance on acceptable peptide candidates.

## 14.21 Simple algorithms

Suppose initially that we have a single query mass,  $M$ . Finding all substrings of  $\sigma$  with weight  $M$  involves a simple linear scan. The algorithm maintains indices  $b$  and  $e$  for the beginning and end of the current substring and accumulates the current substring mass in  $\widehat{M}$ .

If the current mass is less than  $M$ ,  $e$  is incremented and  $\widehat{M}$  increased. If the current mass is greater than  $\widehat{M}$ ,  $b$  is incremented and  $\widehat{M}$  decreased. If the current mass equals  $M$ , then  $\sigma_{b\dots e}$  is output. See Algorithm ?? for pseudo-code.

---

### Algorithm 1 Linear Scan

---

```

b  $\leftarrow$  1, e  $\leftarrow$  0,  $\widehat{M}$   $\leftarrow$  0.
while e < n or  $\widehat{M}$   $\leq$  M do
  if  $\widehat{M} = M$  then
    Output  $\sigma_{b\dots e}$ .
  if  $\widehat{M} < M$  and e < n then
    e  $\leftarrow$  e + 1,  $\widehat{M}$   $\leftarrow$   $\widehat{M}$  +  $\mu(\sigma_e)$ .
  else  $\{\widehat{M} \geq M\}$ 
     $\widehat{M}$   $\leftarrow$   $\widehat{M}$  -  $\mu(\sigma_b)$ , b  $\leftarrow$  b + 1.

```

---

## 14.22 Simple algorithms

Obviously the above algorithm needs time  $O(nk)$  to enumerate all peptide candidates that match the  $k$  query masses, if we call it sequentially.

However, the algorithm would output redundant peptide candidates and compute for those their mass from scratch every time it encounters them. On the other hand, the algorithm can easily keep track of context information.

There are now two obvious points that need to be addressed. Firstly, how we can avoid to find redundant peptides, and secondly, how we can speed up the linear scan for  $k$  query masses.

## 14.23 Eliminating redundancy

The obvious way to eliminate redundancy is the use of a string index like a suffix tree or suffix array. A suffix tree can be built in linear time and space and would allow us to conduct the mass computations for redundant substrings only once.

However, as it turns out, the additional overhead for constructing the string index does not pay off.

Hence, we will show now how the  $k$  searches can be conducted simultaneously. To do so, we want to determine at each position of the database, which masses do indeed match the mass of the current database string. We construct a lookup table. For integer masses this is straightforward, however, for real masses we need to discretize the mass space by choosing a *discretization* factor  $\delta$ .

## 14.24 Simultaneous linear scan

---

### Algorithm 2 Simultaneous Linear Scan

---

Sort the query masses  $M_1, \dots, M_k$ .

$b \leftarrow 1, e \leftarrow 0, \widehat{M} \leftarrow 0$ .

**while**  $b \leq n$  **do**

**while**  $\widehat{M} \leq M_{\max}$  **and**  $e < n$  **do**

$e \leftarrow e + 1, \widehat{M} \leftarrow \widehat{M} + \mu(\sigma_e)$ .

$Q \leftarrow \text{QueryLookup}(\widehat{M})$ .

**Output**  $(i, \sigma_{b\dots e})$  for each  $i \in Q$ .

$b \leftarrow b + 1, e \leftarrow b - 1, \widehat{M} \leftarrow 0$ .

---

## 14.25 Query lookup

We choose  $\delta$  such that it matches the smallest query mass tolerance  $\min_i \{u_i + l_i\}$  and allocate a table of the appropriate size. Then we iterate through the query masses to populate the table.

For each query mass, we determine the relevant table entries that represent intervals that intersect with the query mass tolerance interval. The query mass intervals that intersect the interval  $J = [j\delta, (j+1)\delta)$  represented by a table entry  $j$  do so in one of three ways:

the lower endpoint falls inside  $J$ , in which case the query mass index is stored in  $T[j].L$ ;

the upper endpoint falls inside  $J$ , in which case the query mass index is stored in  $T[j].U$ ;

or the query mass interval completely overlaps  $J$ , in which case the query mass index is stored in  $T[j].O$ .

The choice of  $\delta$  ensures that no query mass interval is properly contained in  $J$ . Once all the table entries are populated, then for all table entries  $j$ , the query mass indices  $i \in T[j].L$  are sorted in increasing  $M_i - l_i$  order and similarly, the query mass indices  $i \in T[j].U$  are sorted in decreasing  $M_i + u_i$  order.

The resulting lookup table has size bounded above by  $O(M_{\max}/\delta + k \max_i(l_i + u_i)/\delta)$  and can be constructed in time bounded by  $O(M_{\max}/\delta + k \max_i(l_i + u_i)/\delta + (M_{\max}/\delta)O_{\max} \log O_{\max})$ .

## 14.26 Query lookup

---

### Algorithm 3 Query Mass Lookup Table: Construction: Real Case

---

$\delta \leftarrow \min_i(l_i + u_i)$ .  
**allocate and initialize** a lookup table of size  $N = \lfloor \frac{M_{\max}}{\delta} \rfloor + 1$ .  
**for all** query masses  $i$  **do**  
  **for all**  $j = \lfloor \frac{M_i - l_i}{\delta} \rfloor, \dots, \lfloor \frac{M_i + u_i}{\delta} \rfloor$  **do**  
    **if**  $j\delta \leq M_i - l_i < (j + 1)\delta$  **then**  
       $T[j].L \leftarrow i$   
    **else if**  $j\delta \leq M_i + u_i < (j + 1)\delta$  **then**  
       $T[j].U \leftarrow i$   
    **else**  $\{M_i - l_i < j\delta$  **and**  $M_i + u_i \geq (j + 1)\delta\}$   
       $T[j].O \leftarrow i$   
**for all**  $j = 0, \dots, N - 1$  **do**  
  **sort** the elements  $i$  of  $T[j].L$  in increasing  $M_i - l_i$  order.  
  **sort** the elements  $i$  of  $T[j].U$  in decreasing  $M_i + u_i$  order.

---



---

### Algorithm 4 Query Mass Lookup Table: Lookup: Real Case

---

**input** query mass  $\widehat{M}$ .  
 $j \leftarrow \lfloor \frac{\widehat{M}}{\delta} \rfloor$ .  
**output**  $i$  for each  $i \in T[j].O$ .  
**output**  $i$  in order from  $T[j].L$  until  $M_i - l_i > \widehat{M}$ .  
**output**  $i$  in order from  $T[j].U$  until  $M_i + u_i < \widehat{M}$ .

---

We finish the section with the remark, that the simultaneous linear scan is in practice comparable with an implementation using a suffix array, while it does not get rid of the redundancy in the databases.