TECHNICAL UNIVERSITY OF CRETE
SCHOOL OF MINERAL RESOURCES ENGINEERING
Postgraduate Program “Geotechnology and Environment”
MSc Thesis

“Development of a Novel Mass-Spectral Library for the Identification of Biomarker Compounds in Oils”

Makridis Minas

Mechanical Engineer

Examination Committee:

Prof. N. Pasadakis (Supervisor)

Prof. N. Varotsis

Prof. N. Kalogerakis

Chania, 2017

Contents

Abstract

Biomarkers in oils are organic substances that draw their origin from biological molecules that can be correlated with living organisms' constituents. The presence of biomarkers in petroleum suggests that the chemical structure of the original biological molecules has remained unaltered throughout the processes of petroleum generation. This is the key factor for researchers to use biomarkers in order to specify several features of petroleum including the maturation level and the type of the original organic matter and as a tool for the correlation with source rocks.

The identification of biomarkers in petroleum samples is carried out by means of the combined gas chromatography - mass spectrometry technique which is considered a state of the art analytical method for this purpose. Through the chromatography section, samples are subjected to separation of their pure substances which they eventually enter the mass spectrometry section where the quantitative and qualitative detection occurs. The detection comprises the extraction of complete mass spectra for the molecules of interest, whereas the final identification is accomplished with the comparison of the produced spectra with existing mass spectral libraries. However, the identification is not often successful since petroleum biomarkers mass spectra are quite rare in these libraries.

The objective of this study is the development of a novel mass spectral library, for the identification of biomarkers, based on mass spectra presented in various sources (e.g. books, journal articles, web pages). The basic tool for the realization of this effort is the MatLab environment. The library development includes three stages; the data digitization, the dataset creation and the matching process. All the operations are user-friendly regardless the individual's experience level in MatLab. However, the algorithmic background is provided to support the comprehension of these operations.

Keywords: Petroleum, biomarker, mass spectrum, library, algorithm, matching.

Περίληψη

Ως βιοδείκτες του πετρελαίου ορίζονται οι οργανικές ενώσεις που αντλούν την προέλευσή τους από βιολογικά μόρια συστατικών των ζώντων οργανισμών. Η ύπαρξη των βιοδεικτών στο πετρέλαιο φανερώνει ότι η χημική δομή των πρόδρομων βιολογικών μορίων έχει παραμείνει αναλλοίωτη καθ' όλη τη διάρκεια των διεργασιών της δημιουργίας του. Το γεγονός αυτό αποτελεί καθοριστικό παράγοντα για τους ερευνητές που δύνανται να χρησιμοποιούν τους βιοδείκτες για τον καθορισμό αρκετών χαρακτηριστικών του πετρελαίου όπως το επίπεδο ωριμότητας και την προέλευση της οργανικής ύλης αλλά και ως εργαλείο για τη συσχέτισή του με το εκάστοτε μητρικό πέτρωμα.

Η ανίχνευση βιοδεικτών στα πετρελαϊκά δείγματα πραγματοποιείται μέσω της συνδυασμένης τεχνικής αέριας χρωματογραφίας - φασματοσκοπίας μάζας η οποία αποτελεί την πλέον καθιερωμένη αναλυτική μέθοδο για αυτόν τον σκοπό. Κατά το στάδιο της αέριας χρωματογραφίας τα δείγματα διαχωρίζονται στις καθαρές ουσίες που τα συνθέτουν και στη συνέχεια αυτές εισέρχονται στο τμήμα της φασματοσκοπίας μάζας όπου πραγματοποιείται η τελική ποσοτική και ποιοτική αναγνώριση. Η αναγνώριση περιλαμβάνει τη λήψη ολοκληρωμένων φασμάτων μάζας για τα μόρια των ουσιών ενδιαφέροντος, ενώ η τελική ταυτοποίηση επιτυγχάνεται μέσω της σύγκρισης των παραγομένων φασμάτων με ήδη υπάρχουσες βιβλιοθήκες φασμάτων μάζας. Ωστόσο, η ταυτοποίηση συχνά καθίσταται ανεπιτυχής λόγω της σπανιότητας των φασμάτων βιοδεικτών στις εν λόγω βιβλιοθήκες.

Ο αντικειμενικός σκοπός της παρούσας μελέτης είναι η ανάπτυξη μιας καινοτόμου βιβλιοθήκης φασμάτων μάζας, για την ταυτοποίηση βιοδεικτών, η οποία βασίζεται σε φάσματα μάζας που έχουν παρουσιαστεί σε διάφορες πηγές (βιβλία, άρθρα, ιστοσελίδες). Βασικό εργαλείο για την επίτευξη αυτού του πονήματος αποτελεί το προγραμματιστικό περιβάλλον MatLab. Η ανάπτυξη της βιβλιοθήκης περιλαμβάνει τρία επί μέρους στάδια: την ψηφιοποίηση των δεδομένων, τη δημιουργία του ενιαίου συνόλου δεδομένων και τη διαδικασία ταυτοποίησης. Όλες οι λειτουργίες είναι φιλικές προς το χρήστη ανεξάρτητα από το επίπεδο εξοικείωσής του με το MatLab. Ωστόσο, παρέχεται και το αλγοριθμικό υπόβαθρο για την κατανόηση αυτών των λειτουργιών.

Λέξεις Κλειδιά: Πετρέλαιο, βιοδείκτης, φάσμα μάζας, βιβλιοθήκη, αλγόριθμος, ταυτοποίηση.

To my wife Georgia and my son Antonios.

This page has been intentionally left blank.

1. Biomarkers

Hydrocarbon formations are generated only under specific geological conditions involving the occurrence of source rocks created throughout the process of sedimentation. Gradual deposition of both organic matter and mineral particles is the main causative factor for the creation and the evolution of a possible hydrocarbon source rock. Living organisms are the precursors of the organic matter bearing the four basic groups of biopolymeric constituents i.e. lipids, carbohydrates, proteins and lignin, as regards their molecular structure.

After the organisms' death, these biopolymers degrade into their bio-monomers. Their structure is rearranged due to reactions controlled by bacterial enzymes, into substances called 'geo-monomers'. These are eventually converted into the 'geo-polymer' kerogen within a few hundred meters of burial [1]. This process which involves the described transformations and leads to the creation of kerogen is called diagenesis.

After the process of diagenesis, oil is generated by the thermal degradation of kerogen in the source beds. With the continuous process of the sedimentation the depth of burial increases, the temperature in these rocks rises and, above a certain threshold temperature, the chemically labile portion of the kerogen begins to transform into petroleum compounds [2]. In other words, pressure and heat comprise the driving forces for the both the ignition of the generation reactions and the maturation of petroleum. The process that leads to the creation of petroleum is called catagenesis.

The temperature interval where petroleum generation is in progress is referred to as the 'oil window'. It extends over the temperature interval of about 80-150°C. For petroleum exploration, the determination of the precise stage at which hydrocarbon generation reactions have progressed in a particular source rock is very important because it can make the exploitation economically advisable. On the other hand, bad timing or lack of knowledge of the maturation level of petroleum can lead to a withdrawal of the exploitation procedures. It can be said that the oil window is characterized by 'flexibility', in other words the presence of it may vary from one source rock to another.

The determination of the oil window can be realized through the use of biomarkers as geochemical tools. Biomarkers are organic substances that can be correlated with living organisms' constituents. As regards petroleum, biomarkers draw their origin from biological molecules that had been altered through the sedimentation [3]. However, their chemical structure has remained relatively unaltered and this is the reason why biomarkers have managed to ensure their presence inside the 'oil window' where petroleum generation is in progress. Petroleum generation (green area) along with the presence of biomarkers (blue

area) is shown in the schematic representation of petroleum formation during burial (Figure 1.1).

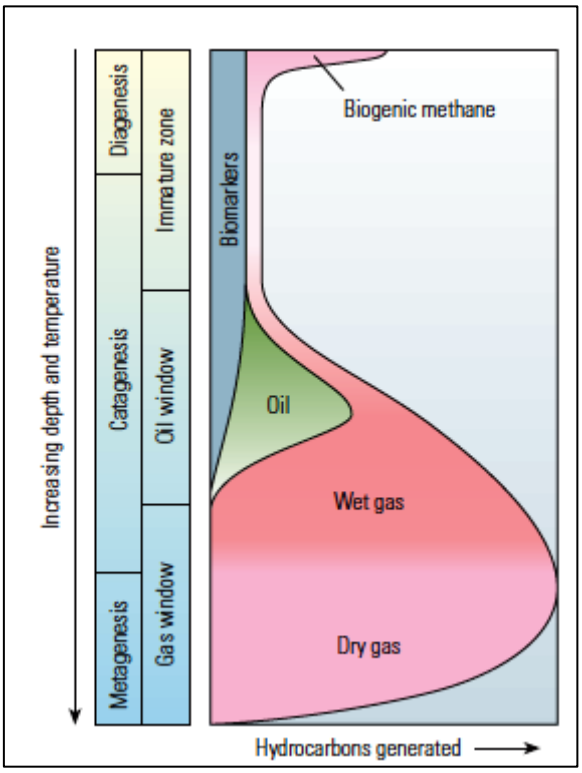

Figure 1.1: Schematic representation of petroleum formation during burial (modified from [4]).

The presence of petroleum biomarkers enables the researchers to use them as unmistakable 'witnesses' in order to specify several characteristics of the original organic matter. Apart from the maturation level of petroleum which has been already discussed, there are more features that can be specified, including the type of the organic matter, the migration paths of petroleum, the correlation of one kind of petroleum with another or even the correlation of different types of petroleum with their source rock [3].

1.1. Types of Biomarkers

The maturation level of petroleum can be determined through the study of specific biomarker ratios and their differentiations. During the process of maturation, petroleum constituents undergo a great deal of thermal stress which leads to alternations of individual biomarker molecules stereochemistry The spatial arrangement of specific methyl groups ($-CH_3$) or hydrogen atoms as parts of the hydrocarbon ring system or side chains, changes systematically as a function of increasing temperature (configurational isomerization). In this way, the original biological form of the molecule with respect to the 3D-arrangement of these particular groups is gradually changed into a thermally more stable form [5].

There are several biomarkers categories that are used for the discrimination of different types of petroleum. These categories include normal alkanes, isoprenoids, hopanes and steranes. The understanding of biomarkers as geochemical characterization criteria requires a brief description of this classification which is provided to the reader in the following paragraphs.

1.1.1. Normal Alkanes

Normal alkanes with carbon atoms ranging from C_{15} to C_{35} are considered to be quite strong evidence of the initial biological matter because their molecular structure is met inside both terrestrial and marine organisms. This specific molecular structure consists of simple hydrocarbons chains. Consequently, it can be easily correlated with the molecular structure of lipids which are included among the fundamental constituents of living organisms. The great importance of lipids is based on their contribution during the cellular creation procedures of living organisms.

An indicator of whether the organic matter originates from terrestrial organisms or marine ones is the terrigenous to aquatic ratio (TAR). Ratios of certain n-alkanes can be used to identify changes in the relative amounts of terrigenous versus aquatic hydrocarbons in sediment or rock extracts [6]. High TARs indicate more terrigenous organic matter from the surrounding watershed relative to aquatic sources [7]. TAR is given by the following equation:

$$\mathrm{TAR} = \frac{\mathrm{nC}_{27} + \mathrm{nC}_{29} + \mathrm{nC}_{31}}{\mathrm{nC}_{15} + \mathrm{nC}_{17} + \mathrm{nC}_{19}} \tag{1.1}$$

Both the numerator and the denominator of TAR ratio include the concentrations of normal alkanes measured as peak height or areas in gas chromatography. TAR must be used with caution because it is sensitive to thermal maturation and biodegradation. Another interesting aspect using the TAR index is the fact that certain non-marine algae like Botryococcus braunii contribute to the C_{27} – C_{31} normal alkanes [8, 9]. Therefore, the researcher must be very cautious using TAR in order to avoid erroneous estimations.

It has been proved by [10] that the ratio of the odd number carbon molecule paraffines to the even carbon molecule paraffines, between a range of carbon chain length, is reduced as the age of sediments increases. For the quantification of this attribute the Carbon Preference Index (CPI) has been introduced. This index is given by the following equation:

$$\mathrm{CPI}_{C_{24}-C_{34}} = \left[\frac{(C_{25} + C_{27} + C_{29} + C_{31} + C_{33})}{(C_{24} + C_{26} + C_{28} + C_{30} + C_{32})} + \frac{(C_{25} + C_{27} + C_{29} + C_{31} + C_{33})}{(C_{26} + C_{28} + C_{30} + C_{32} + C_{34})} \right] \times \frac{1}{2} \tag{1.2}$$

Alternatively, the Odd to Even Predominance (OEP) index introduced in [11] can be used too. It is defined as the ratio of odd number molecule n-alkanes over even number molecule n-alkanes. In order to calculate this index, five continuous members of the n-alkanes series are used and then they are applied into the following equation:

$$OEP = \left[\frac{\left(C_i + 6C_{i+2} + C_{i+4}\right)}{\left(4C_{i+1} + 4C_{i+3}\right)} \right]^{(-1)i+1} \quad (1.3)$$

Where C_i: n-alkane concentration and (i+2) the n-alkane in the middle of the studied chain [3].

Like TAR, these two described indices are strongly depending on the maturity level of the organic matter, but also on the biodegradation that the organic matter has been subjected to. The latter has to do with the fact that the paraffins are more susceptible to biodegradation than other organic molecules like aromatics, which tend to repel bacteria from introducing them to their metabolic actions.

1.1.2. Isoprenoids

The isoprenoid (or terpenoid or isopentenoid) biomarkers are composed of two or more five-carbon units of isoprene (2- methyl-1, 3-butadiene) [12]. The applicability of isoprenoids to be used as biomarkers is based on the resistance they present against the conditions during petroleum generation. This remarkable resistance is explained by the chemical structure of isoprene which includes covalent carbon-carbon bonds [6]. The chemical structure of isoprene is shown in Figure 1.2.

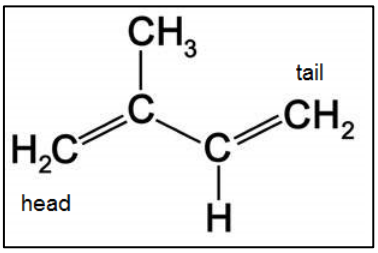

Figure 1.2: Chemical structure of isoprene.

Isoprenoids can be classified according to the number of isoprene units from which they are biogenetically derived, even though some carbons may have been added or lost [13]. Moreover, isoprenoid biomarker families can be found as saturated (acyclic or cyclic) forms and they can be classified according to the number of their isoprene subunits [6]. In general, the basic isoprenoid families are provided in Table 1.1.

Table 1.1: Basic isoprenoid families (modified from [14]).

Name	Isoprene Subunits
Hemiterpane (C_5)	1
Monoterpanes (C_{10})	2
Sesquiterpanes (C_{15})	3
Diterpanes (C_{20})	4
Sesterterpanes (C_{25})	5
Triterpanes & steranes (C_{30})	6
Tetraterpanes (C_{40})	8
Polyterpanes ($C_{5n\ (n>8)}$)	9 or more

The acyclic isoprenoids can also be further categorized according to the way that the isoprene units are connected to each other. Head to tail linkage gives the group of normal isoprenoids like pristane (C_{19}) and phytane (C_{20}), [3]. These acyclic, isoprenoid biomarkers originate from the hydrolysis of porphyrin during the sedimentation. Porphyrin is the basic part of chlorophyll molecules, while chlorophyll is the fundamental substance for the photosynthetic process of plants. The molecular structures of pristane and phytane are given in Figure 1.3.

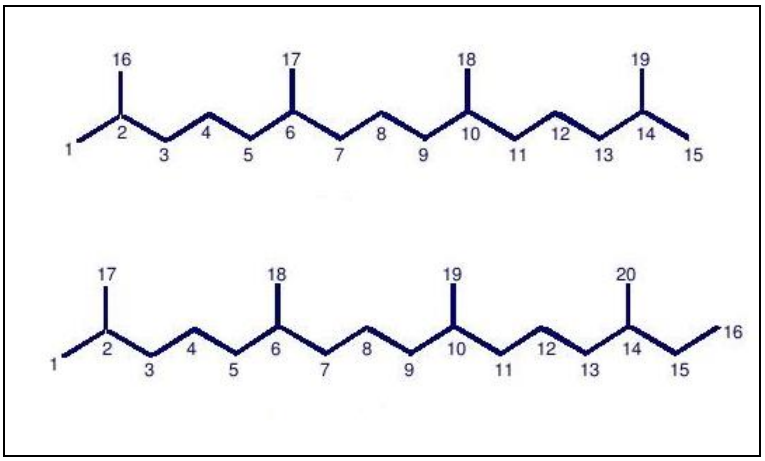

Figure 1.3: Pristane and phytane chemical structures.

The presence of oxygen leads to oxidizing sedimentation conditions, whereas the lack of oxygen is correlated with reducing conditions. The degradation of chlorophyll under these two different cases leads to the occurrence of pristane and phytane, respectively. Therefore, the ratio of pristane over phytane (Pr/Ph) is used as a measure for the characterization of the original sedimentation environment [15]. The quantification of the discussed indices is given in Table 1.2.

Table 1.2: CPI, OEP indices and Pr/Ph ratio numerical interpretation.

Biological matter / Index	CPI	OEP	Pr / Ph
Tree leaves	4		
Sponges	1,2		
Corals	1,1		
Marine plangton	1,1		
Surface sediments	2,5 - 5,5		
Deep sediments	1		
Terrestrial plants		>>1	
Shore sediments		2-5	
Marine organisms		~1	
Anoxic or hypersaline			<0,8
Marine			0,8-2,5
Terrestrial			≥3

Geochemical research in petroleum involves the use of cyclic isoprenoids also. The saturate forms of these substances are called terpanes. The most widely used group of terpanes in petroleum geochemistry is the hopanes. The molecular structure of hopanes consists of four rings of six carbon atoms and one ring of five carbon atoms. Their number of carbon atoms is between the range of 27-35 and the most traceable hopanes are C_{29} and C_{30}, named as nor - hopane and hopane respectively. The geochemical interest of hopanes is based upon their usage for the characterization of the sedimentation environment. Moreover, they are used as a measure of correlation of petroleum with the different source rocks and also as maturity indicators [3]. The chemical structure of hopane is provided in Figure 1.4.

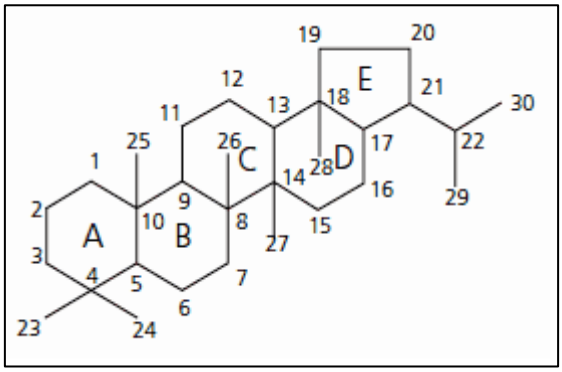

Figure 1.4: Hopane chemical structure.

Apart from hopanes there is another group of cyclic isoprenoid biomarkers, i.e. the steranes. Steranes are organic substances originating from the chemical reduction of sterols, which are present in both eukaryotic and prokaryotic organisms, during the phase of diagenesis.

The most common steranes have a number of carbon atoms ranges between 26-30, however their most standard forms are considered to be C_{27}, C_{28}, and C_{29} [3]. The chemical structure of the basic type of steranes (C_{27} - cholestane) is given in Figure 1.5.

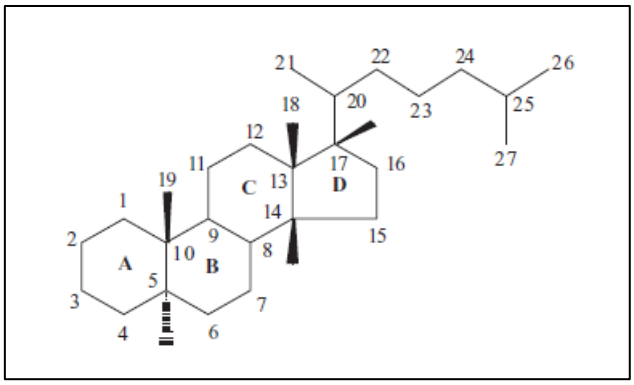

Figure 1.5: Cholestane chemical structure.

The fact that steranes originate from the chemical reduction of sterols during the diagenetic phase, justifies their applicability for the evaluation of the prevailing conditions during these reductive reactions. In other words they can be used to provide information about the depositional environment [3]. The characterization of the original environment according to the basic types of steranes is illustrated in Figure 1.6.

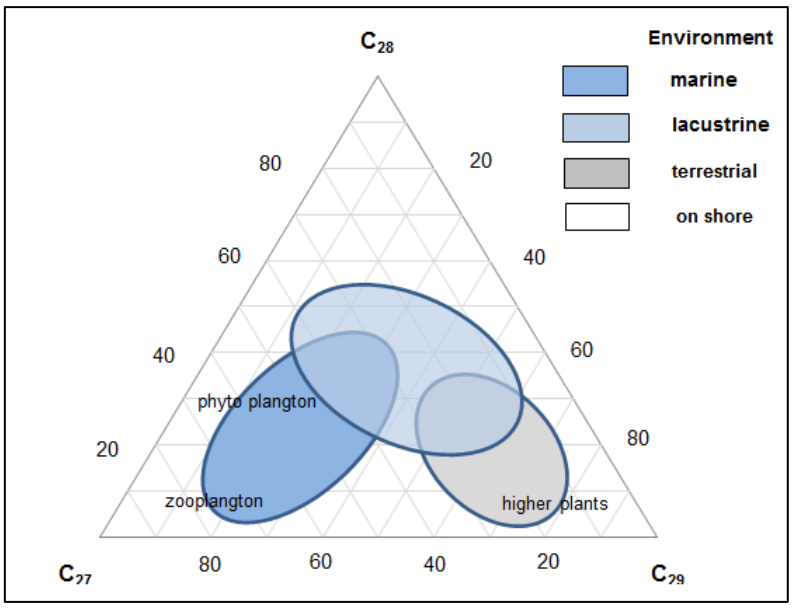

Figure 1.6: Original environment evaluation according steranes' concentrations (modified from [3]).

This page has been intentionally left blank.

2. Gas Chromatography - Mass Spectrometry (GC - MS)

This Chapter provides a description of the combined GC - MS method. The basic tools for the characterization of biomarkers are the mass spectral databases obtained by this procedure. The unknown samples are subjected to this procedure which provides the mass spectrum as a final product. Gas chromatography - mass spectrometry (GC - MS) is a combination of two discrete micro-analytical techniques. Gas chromatography (GC) is a separation technique, while mass spectrometry (MS), can be used as an identification technique.

This combined method can become useful both for the separation of the components of a complex mixture (qualitative information) and for the quantitative determination from a few femtomoles of an analyte [16]. The basic steps of the whole procedure are given in Figure 2.1 in a form of a schematic diagram, followed by separate description of each step instrumentation, are provided in the following paragraphs.

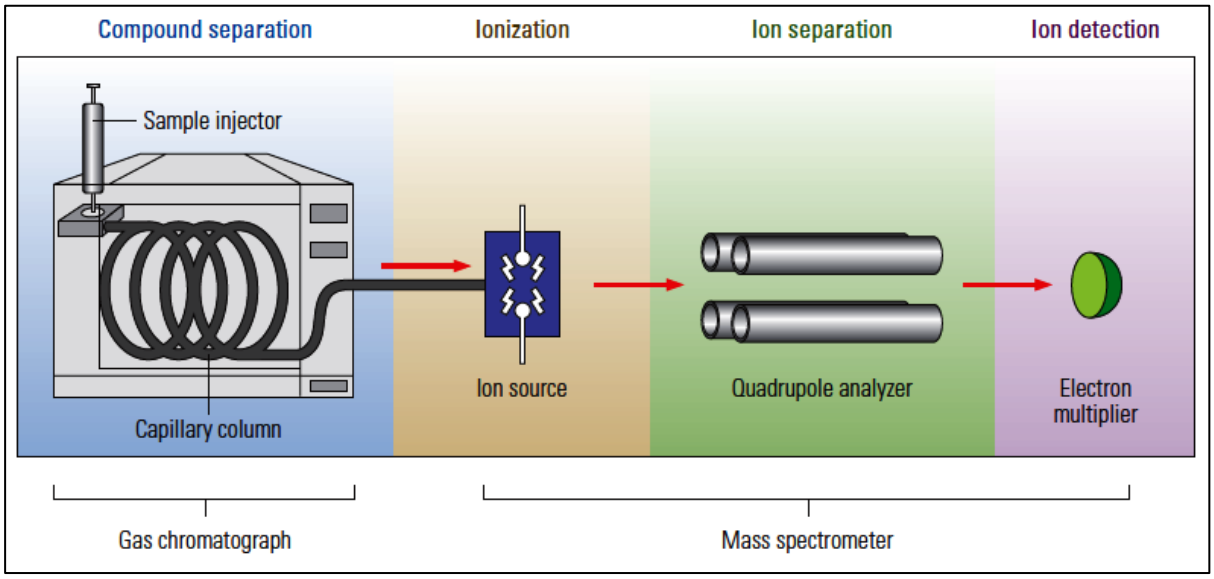

Figure 2.1: Schematic diagram of GC-MS devices (modified from [4]).

2.1. Gas Chromatography (CG)

The operating principle of GC holds that the sample which is a mixture of different compounds undergoes a gradual separation when it passes through the column of the chromatograph [17, 18]. The sample is vaporized by increasing its temperature when entering the chromatograph device. Along with the sample, a carrier gas is also introduced into the chromatograph which drives the sample vapors through the column.

The inner walls of the chromatograph capillary column are coated with organic materials which comprise the stationary phase. The separation is achieved by the different degree of

affinity of the mixture compounds to the stationary state. The more a compound is held by the stationary phase the greater degree of affinity it has to it. Consequently, the time that the compound spends inside the column (retention time) is greater. Eventually, compounds with great retention times elute later in time from the column. On the other hand, compounds with short retention times leave the column earlier. It must be mentioned that the carrier gas is characterized by the lowest degree of affinity with the stationary phase and this is the reason why it exits the column first. As soon as they exit the chromatograph, all the compounds continue their path to the mass spectrometer for the final detection and quantification.

2.2. Ionization (Electron Ion Source)

Electron Ionization (EI) is widely used in organic mass spectrometry. It involves the interaction of a low-pressure (~0,1 Pa) analyte gas cloud which enters the ionization stage after the CG stage, with electrons that have been accelerated through a 70 eV electric field. Most organic molecules have an ionization potential of about 10 eV while maximum fragmentation results around 30 eV. For reproducibility reasons, the implementation of a 70 eV electric field is necessary for the optimal fragmentation when different devices are used [19]. The schematic illustration of a typical EI source is provided in Figure 2.2.

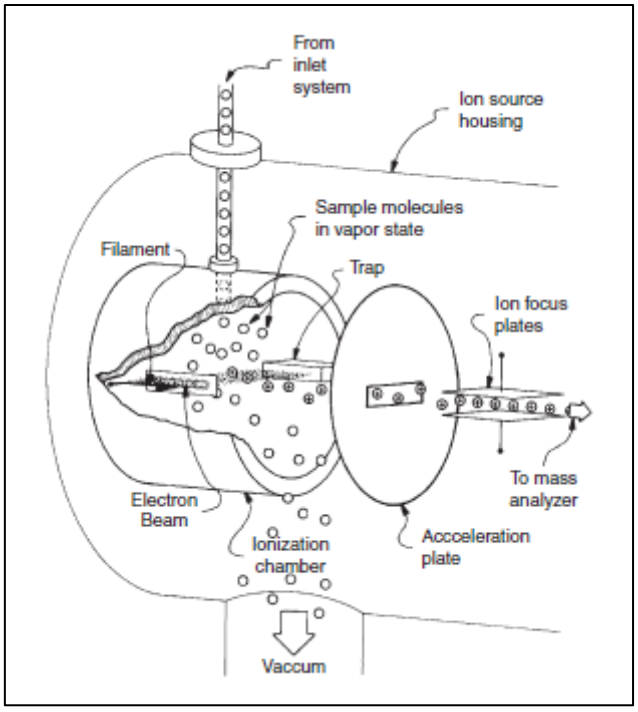

Figure 2.2: Schematic illustration of an EI source (modified from [19]).

The ionization chamber occupies a volume of about 1 cm^3 which is enclosed inside the ion source housing. The chamber incorporates a heated filament, which gives off electrons, and an electron trap at the opposite side of it [20]. A permanent magnet applies a potential

difference between the filament and the trap. As a result, the electrons are accelerated and they become able enough to travel from the one side to the other. Consequently, an electron beam is established between the two sides. The electron beam has a helical form which increases the probability of an ionizing electron to come in proximity with the energy cloud that comprises most of the molecular volume of an analyte [19].

The ionizing electrons energize the analyte molecules by being in proximity with them and not necessarily by coming in contact with a nucleus of an atom or one of the molecules' electrons [21]. The energized molecules wanting to achieve a lower energy state expel an electron providing the positive-charge molecular ion. The electrons expelled by the analyte undergo the same fate as the ionizing electrons, as they are attracted to and be neutralized by the positively charged trap. It must be mentioned that the ionizing electron may interact with more than a single analyte molecule. A series of lenses is used to extract, focus and accelerate the ions to the mass analyzer [19, 21].

2.3. Ion Separation (Quadrupole Analyzer)

The basic principle of quadrupole's operation is based upon the existence of an electric field which accelerates the ions coming out of the source region and into the analyzer. With the implementation of AC and DC voltages the quadrupole filters out ions which their masses are lower or higher than certain critical values. The critical masses are determined by the settings of the apparatus. Different settings allow ions of different masses to pass through the quadrupole section and reach the detector providing the mass spectrum of the analyzed substance.

The analyzer [22] consists of four rod shaped electrodes arranged across from each other. The electrodes are constructed in a sense that the cross section of the free space between the rods forms a hyperbola. Under this consideration the created electric field inside the electrodes will have a hyperbolic form too. The electric field is formed with the superposition of DC and AC voltage. This voltage is applied upon every two electrodes alternately (x and y sets) with a phase shift of 180° producing an oscillating electric field (Figure 2.3).

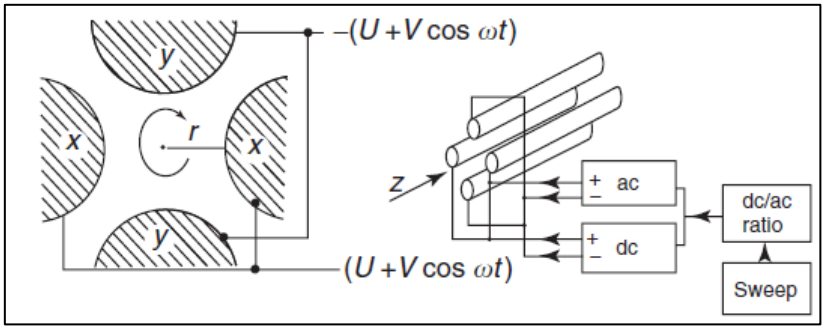

Figure 2.3: Quadrupole cross section and applied voltages (modified from [23]).

During the periods of positive AC voltage, the positive ions are forced to gather near the center of the free space between the electrodes. On the other hand, when the AC voltage is negative, the ions are attracted to the electrodes. The presence of positive or negative super positioned DC voltage can act in favor of the higher or the lower mass ions respectively so that they can reach the detector.

When the DC voltage is positive, the ions tend to keep their motion centered. The lower mass ions are influenced by the AC voltage oscillations more, because they accelerate more easily, so it is more possible for them to collide with the electrodes. Higher mass ions retain more stable trajectories because they do not accelerate so easily and DC voltage keeps them focused. It can be concluded that the presence of the positive DC voltage favors the higher mass ions to retain their trajectories centered while lower than a critical mass ions collide with the electrodes (Figure 2.4, blue line).

When the DC voltage is negative the positive ions tend to be destabilized from the center of the motion and they tend to be attracted to the electrodes. In this case the AC voltage is the voltage that can provide some stabilization to their motion, when it takes positive values. So, the DC voltage favors the lower mass ions to maintain their trajectories. Higher than a critical mass ions collide with the electrodes (Figure 2.4, red line).

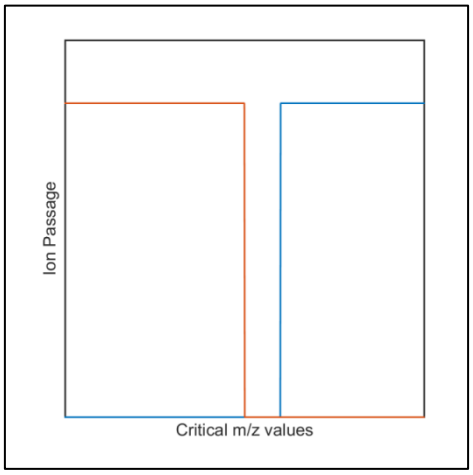

Figure 2.4: Double mass filtered peak creation.

The co-presence of AC and DC potentials acts as a combined low and high band filter for the ions. This double filter creates a window which is permeable to the ions of the selected m/z value. The ions reach the detector and produce a peak, only if their m/z values lie inside the narrow range that the double filter produces, whereas the ions above or below the set m/z value are rejected by the quadrupole. The AC and DC fields are regulated by potential or frequency, so different m/z ions can reach the detector. By scanning all the m/z range

successively, different peaks are created and consequently, a complete mass spectrum is acquired.

The locus of the stable operation of the quadrupole lies within the shaded triangle of Figure 2.5. Inside this area the ions follow stable orbits and they are transmitted through the analyzer, whereas outside this triangle the orbits become infinitely large and the ions are lost [23]. The gradient of the scan line determines the proportion of ions transmitted. The optimum gradient is selected to cut the shaded triangle near its apex and give the optimum combination of transmission and mass resolution.

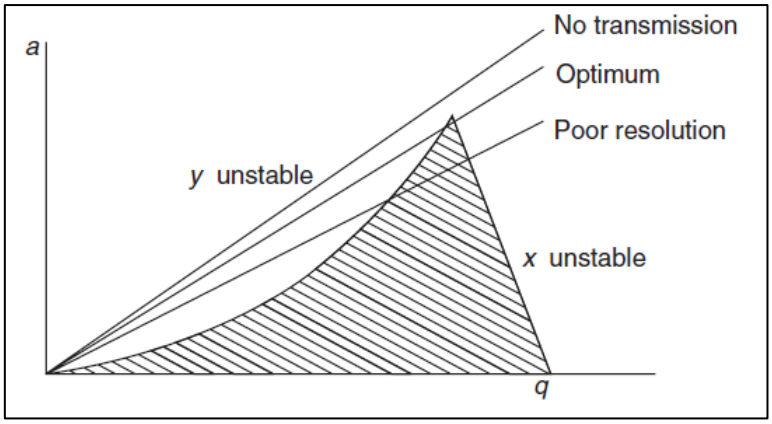

Figure 2.5: Stable operation locus (modified from [23]).

If there is no DC voltage, all ions will pass through and there will be no mass separation. As the gradient of the scan line is increased, i.e., as U is increased relative to V, the proportion of ions following stable orbits decreases and the mass resolution increases. If the scan line is too steep it will not intersect with the triangle at all and no ions will be transmitted.

2.4. Ion Detection (Electron Multiplier)

One of the most widely used ion detector in mass spectrometry is the electron multiplier (EM). The basic principle of its operation holds that the ions coming out from the analyzer are accelerated to a high level of velocity, enhancing the efficiency of detection. The acceleration is achieved by maintaining high potential (±3 to ±30 kV) via an electrode called “conversion dynode”, which is located at the opposite side of the detected ions [20].

When an ion strikes the conversion dynode it causes the emission of several secondary particles. These particles can be positive ions, negative ions, electrons or neutrals. When positive ions strike the negative high-voltage conversion dynode, the secondary particles of interest are negative ions and electrons. On the contrary, when negative ions strike the positive high-voltage conversion dynode, the secondary particles are positive ions [20].

The electron multipliers may consist of several discrete dynodes or alternatively their configuration may comprise a continuous dynode called channeltron. As regards the first case, the electron multiplier is made up of a series of 12 to 20 dynodes. These dynodes are held at decreasing negative potentials by a chain of resistors. The first dynode is held at a high negative potential from -1 to -5 kV, whereas the output of the multiplier remains at a reference ground potential.

The secondary particles generated from the conversion dynode strike the first dynode surface causing an emission of secondary electrons. These electrons are accelerated to the next dynode because this is held at a lower potential, causing the emission of more electrons. This process continues as the secondary electrons travel towards the ground potential. Thus a cascade of electrons is created and the final flow of electrons provides an electric current at the end of the electron multiplier which is eventually increased by conventional electronic amplification [20]. The entire procedure is provided in Figure 2.6.

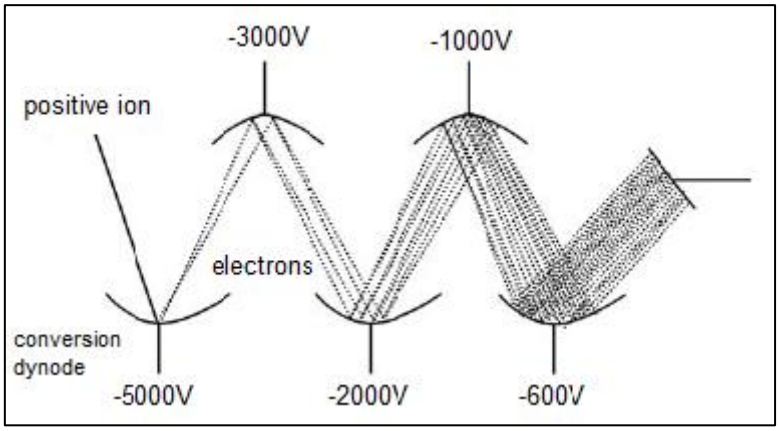

Figure 2.6: Schematic diagram of an electron multiplier (modified from [20]).

2.5. Hydrocarbons Mass Spectra Interpretation

The final product of the mass spectrometer is a complete mass spectrum which provides the necessary information for the molecular identification. The main idea is that the mass spectrum shows the mass of the molecule and the masses of the different ions that can be possibly formed during the ionization procedure.

The axis of abscissas expresses the mass to z (m/z) ratio, i.e. the fraction of mass to the number of charges of the ions employed. The axis of ordinates expresses the relative intensity, i.e. frequency of occurrence of an ion under the chosen ionization conditions. The most frequent ion is characterized as the base peak. The base peak reflects the propensity of a molecule to break at a certain point of its structure during the ionization. The scale is usually given in percentages relative to the base peak which have 100% intensity. The last

visible peak of the spectrum range usually represents the molecular ion which is equal to the molar mass of the substance.

The observation of a mass spectrum reveals that every ion peak is usually accompanied by several secondary peaks giving the impression that ion peaks appear in the form of clusters. This is explained by the fact that carbon atoms can be found at their different stable isotopic forms e.g. ^{12}C, ^{13}C. Therefore, the fragmentation during the ionization process may produce the respective isotopic ions. Eventually, the peaks that correspond to these isotopic ions appear in the spectra as isotopic abundances [24].

The way that the peaks are distributed in a hydrocarbon spectrum follows certain rules. In normal alkanes most of the peaks tend to appear in the beginning of the spectrum having low m/z values. The reason is that simple bonds are more susceptible in breaking during the ionization. Usually, the first fragment ion peak represents an M^{+29} ion due to the loss of an ethyl radical. The base peak of normal alkanes with carbon chain ≥4, show a base peak at m/z 43 or 57. Alkanes yield a series of ions represented by peaks differing by 14 m/z units (e.g., 43, 57, 71, 85, etc.) [19], because of the successive losses of CH_2 groups. The molecular ion of normal alkanes is quite remote in terms of its m/z value and it appears at the end of the spectrum. This is explained by the limited possibility for the acquisition of a whole molecule after the ionization. This is the reason why the molecular ion of normal alkanes has a very small relative abundance value.

As regards the mass spectra of branched alkanes, they are governed by the tendency for fragmentation at the branch points. Consequently, the location of the branch point can be determined based on the m/z values of the three peaks representing the more stable secondary carbenium ions [16]. In cycloalkanes the cleavage is favored at the bond which connects the naphthenic ring to the rest aliphatic side chain of the molecule [19]. Saturated rings have a more solid structure and, therefore, they increase their probability of passing through the ionization and the mass analyzer intact.

Aromatic rings are even more stable in terms of their tendency to give ions during the ionization. A very typical example is the molecule of benzene. The biggest probability for this molecule is to pass through the ionization intact. As a result, the base peak of the mass spectrum coincides with the molecular ion peak, and it is equal to the molecular mass, since the rest ‘ion possibilities’ are more insignificant.

The fragmentation of substances containing aromatic rings and alkyl groups is characterized by the benzylic cleavage where a bond on the benzylic carbon atom will cleave due to the pairing of an electron in that bond with the ring radical site resulting to the creation of a benzyl ion with more stable form, the tropylium ion [19]. Tropylium is a charged aromatic ring

cation with seven carbon and seven hydrogen atoms. The base peak of substances like toluene and ethylbenzene has the same value as the m/z value of tropylium.

Polynuclear aromatic hydrocarbons (PAH's), like chrysene, are conjugated forms of the benzene molecule. Their mass spectra prove that their structure is very stable and their base peaks coincide with their molecular ion. The mass spectra for the discussed hydrocarbons substances are provided in Figure 2.7.

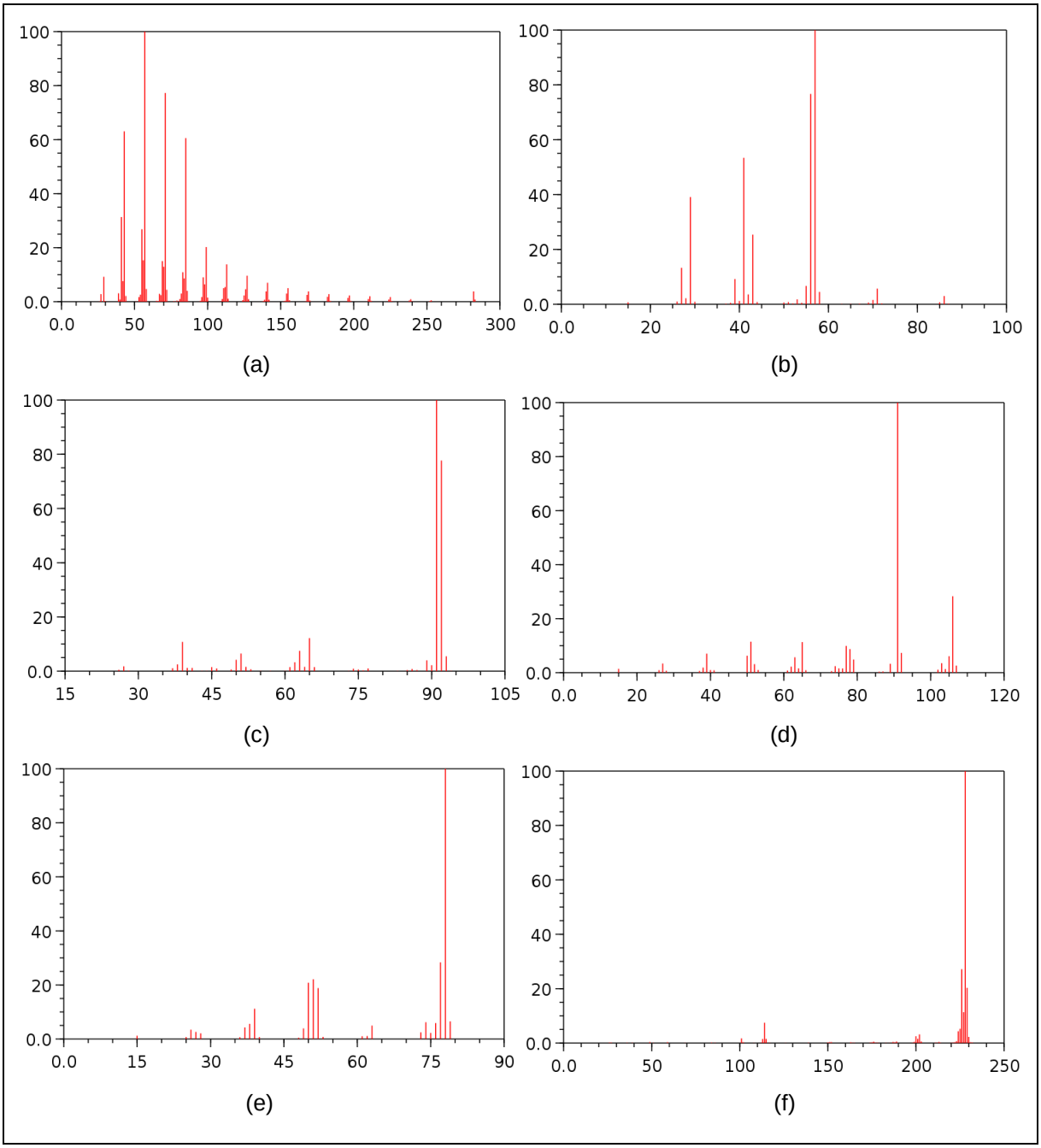

Figure 2.7: Mass spectra for n-eicosane (a), 3-methyl-pentane (b), toluene (c), ethyl-benzene (d), benzene (e), chrysene (f) [25].

The described categories of hydrocarbon molecules are some of the most distinct ones and provide a satisfactory comprehension of the way that the mass spectra are produced. Of course, there are numerous hydrocarbon molecules with much more complex molecular structure. Since the present study is over molecules used as biomarkers the collection of information about their mass spectra base peak ions for some of them was quite necessary and it is provided in Table 2.1.

Table 2.1: Base peaks of hydrocarbon biomarkers molecules (data derived from [6]).

Biomarker	Base Peak
Pristane	57
Phytane	57
Alkylcyclohexanes	83
Methylalkylcyclohexanes	97
Terpanes	123
Gammacerane	191
Hopanes	191
14α - Sterane	217
14β - Sterane	218
14a - methyl - Sterane	231
14b - methyl - Sterane	232
dia – Sterane	259

This page has been intentionally left blank.

3. Hardcopy Data Digitization

The present Chapter incorporates a full description of all the necessary procedures for hardcopy data interpretation and capturing from spectrum images. These images can be derived from various sources (e.g. books or journal articles). A great number of biomarkers mass spectra is included in [26]. In particular, there are 373 mass spectra of such compounds in this book, so the main core of the developed library comes directly from this specific bibliographic source. The names' list of these 373 compounds is also provided in Appendix A of the present text.

The transition from a hardcopy format to a digital one requires several preparative actions, so that the captured data of interest can be properly stored. Once these data are stored, they require extra handling in order to become realistic. The effective coordination of the whole set of operations is carried out by a developed application in MatLab R2014b environment dedicated for this purpose. This application is a graphical user interface (GUI) which has been constructed in a simple way ready to be handled by users just by following the described instructions.

3.1. Image Preparation

Every spectrum image that is intended to be translated by the developed source code must be subjected to a series of preparative actions so as to meet the functionality requirements of the code. This set of actions has to be performed by the user with a great deal of attention for the achievement of the optimal quality. An image of a bad quality is often characterized by the presence of excessive and unwanted information, i.e. flaws or imperfections. Moreover, a bad quality image may lose useful information, i.e. spectral peaks. In any of these cases a bad quality image could possibly lead to an erroneous data translation.

The first action includes the digitization of the hardcopy image through a common scanning device. There are not specific requirements as regards the scanning device as long as the user is complied with the following rules. One of the basic scanning attributes that the user has to be very careful with is the alignment of the hard copy format i.e. the book, when it is placed on the device's scanning surface. The more aligned with the frames of the scanning surface the hard copy is, the less corrective actions the produced image requires after the scanning procedure is complete.

Apart from the alignment, the scanning resolution has to be regulated to 600 dpi, and the image must be saved as a colored .png file. The digitized format of the picture will look like the example of the Figure 3.1 witch shows the mass spectrum of the C_{20} - tricyclic terpane.

This substance can be found with an index number of 170 in both [26] and Appendix A of the present text.

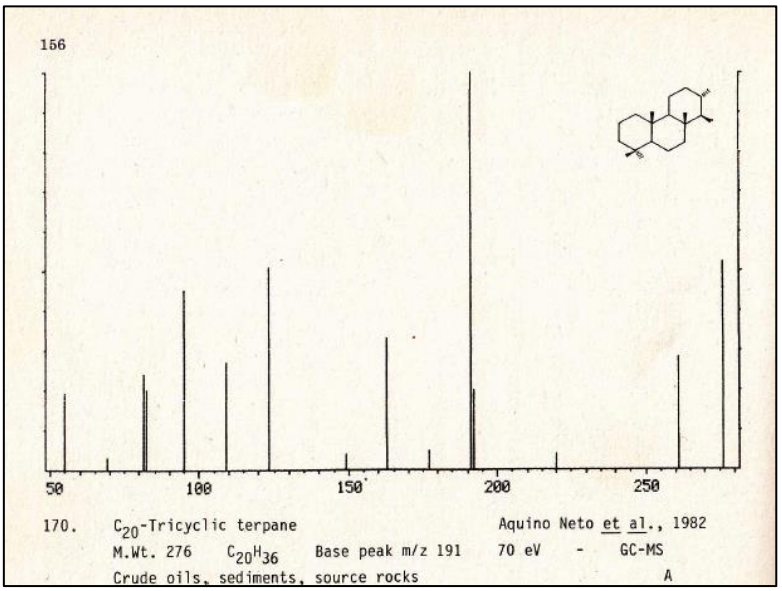

Figure 3.1: Scanned image of C_{20} - tricyclic terpane mass spectrum (modified from [26]).

3.2. Scanned Image Modifications

After the scanning procedure, the user has to perform some additional actions, including cropping, alignment and deleting unnecessary parts from the original image. These actions are described below and they can be properly executed with the help of any common program of image processing.

The first action involves the cropping of the image omitting all the excessive information, and setting new margins. The left hand and the right hand margins of the image must be set exactly on the first and the last ticks of the m/z axis respectively, leaving the vertical axes of the image outside of the new margins. This rule must be followed very carefully especially if the spectrum image consists of two segments. In this case, the segments must be joined in a sense that the second segment will be the sequel of the first one. The compliance with this rule guarantees that the developed algorithm which places the two segments together will work properly. The new left and right margins are shown in Figure 3.2 with red upward arrows.

The new bottom margin must be set up to the point that the m/z axis is left out of the image. When this margin is about to be set the user must notice the alignment of the cropping line with the image m/z axis. Moreover, the angle between these lines must be zero, or close to zero otherwise the user will have to correct this by performing alignment to the image as many degrees as the deviation from zero angle is. The zero degrees angle requirement is also illustrated in Figure 3.2.

The top margin must be set in a way that a small void will exist above the maximum peak of the spectrum. This is a requirement of the MatLab code which will read the image, as it will be explained in the next chapters. The top margin when set, will define the new image height measured in pixels. If the spectrum consists of two segments the height of the second segment must be exactly the same as the height of the first one. This dimensional limitation is very important because the code will not be able to concatenate the two segments if their heights, i.e. the number of rows, are different. The top margin is illustrated with a red downward arrow in Figure 3.2.

After the cropping procedure, the image is defined by its new margins. The only usable information at this point is the number of peaks, so any other information will be translated as noise by the code. In order to avoid this, the user has to remove the redundant areas that provide excessive information. These areas are the chemical type of the substance and any flaws the image could potentially contain due to its original printing or imperfections created during scanning.

Apart from these kinds of flaws there is another type of unnecessary information that must be removed. As it has been already discussed, the angle between the m/z axis of the image and the bottom crop line must be zero or near to zero. Any deviation leaves parts from the m/z axis inside the margins of the cropped image producing an unwanted situation, so these parts must be removed also. The removal of all the described redundant areas is also illustrated with red frames in Figure 3.2. There is a red downward arrow just above the m/z axis which shows the area that must be removed too.

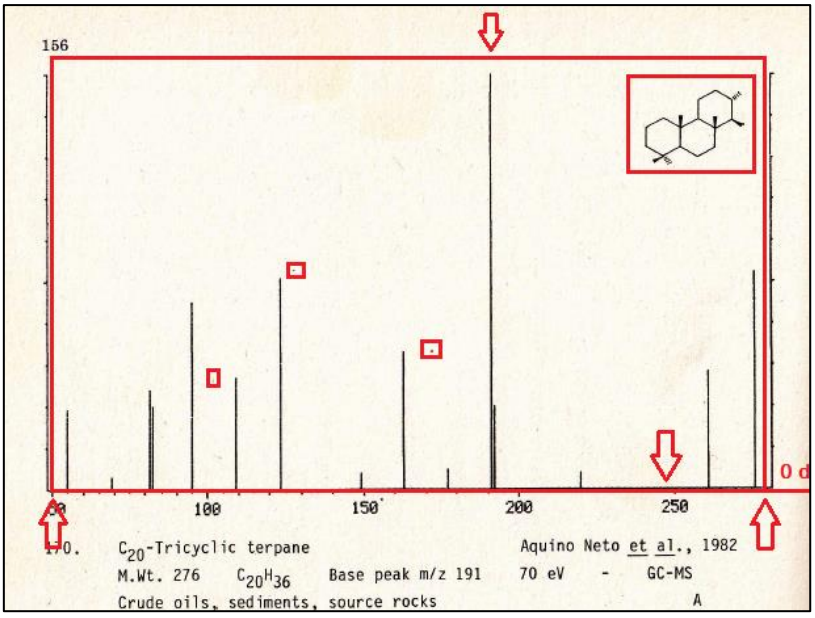

Figure 3.2: Modifications on the scanned image of C_{20} - tricyclic terpane mass spectrum.

After these modifications the processed image maintains only the information which is referred to the peaks. The final appearance of the image is reflected in Figure 3.3. As it can be noticed, the baseline and the vertical axes are not included in the image, making the peaks the only visible elements. This format is the appropriate one for the code to access, so all the available input images must be formatted like this example.

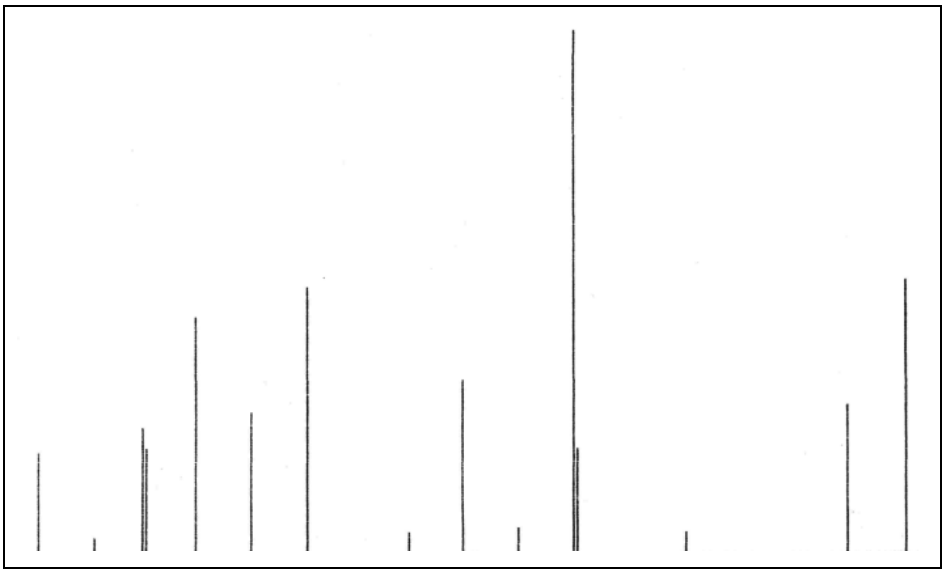

Figure 3.3: Final appearance of the scanned image C_{20} - tricyclic terpane mass spectrum.

3.3. Graphical User Interface (GUI) `bio_x_app`

During the initial design of this study, the development of a user friendly code format has been considered necessary in order that the input information and the extracted data could be more convenient for the users to handle. The format for the application can become very useful to any individuals regardless their experience level in MatLab. The user has not necessarily to be aware of the underlying algorithmic details of the application. However, the code for this application is provided for extra study in Appendix B because it occupies a great deal of space. A detailed step-by-step description of the `bio_x_app` principles of operation is given in the present paragraph.

3.3.1. Running the App

The first step for the user is to run MatLab through the shortcut icon. After the MatLab initializing procedure is complete, the user has to make sure that the `bio_x_app.m` file is located inside the window of the current folder along with the accompanying files that are needed for the application in order to be functional. These necessary files that the current folder must include are given in Table 3.1.

Table 3.1: Necessary files for the application functionality.

File	Use
`bio_x_app.m`	main function
`bio_x_app.fig`	application interface format
`image_to_coords.m`	internal function
`Title.mat`	intro for the .txt output
`index_number``.png`	input .png files

The next step is to double click on the `bio_x_app.m` and the MatLab Editor will appear. The next move is to click on the Run button of the editor. This action will allow the code to produce the visual output of the graphical user interface. Once the application becomes visible, all the underlying internal functions of the code are ready to accept their input arguments.

The user must always have an effective way of monitoring all the current actions so a good display arrangement is very helpful. As it was previously discussed, it is very important for the current folder to be displayed so the user can attach the files of the Table 3.1 inside. Apart from the presence of the current folder, the workspace window and the command line are also quite important for an optimal control of the whole set of program actions. A suggested display arrangement, where the noticeable parts are illustrated with red frames, is shown in Figure. 3.4.

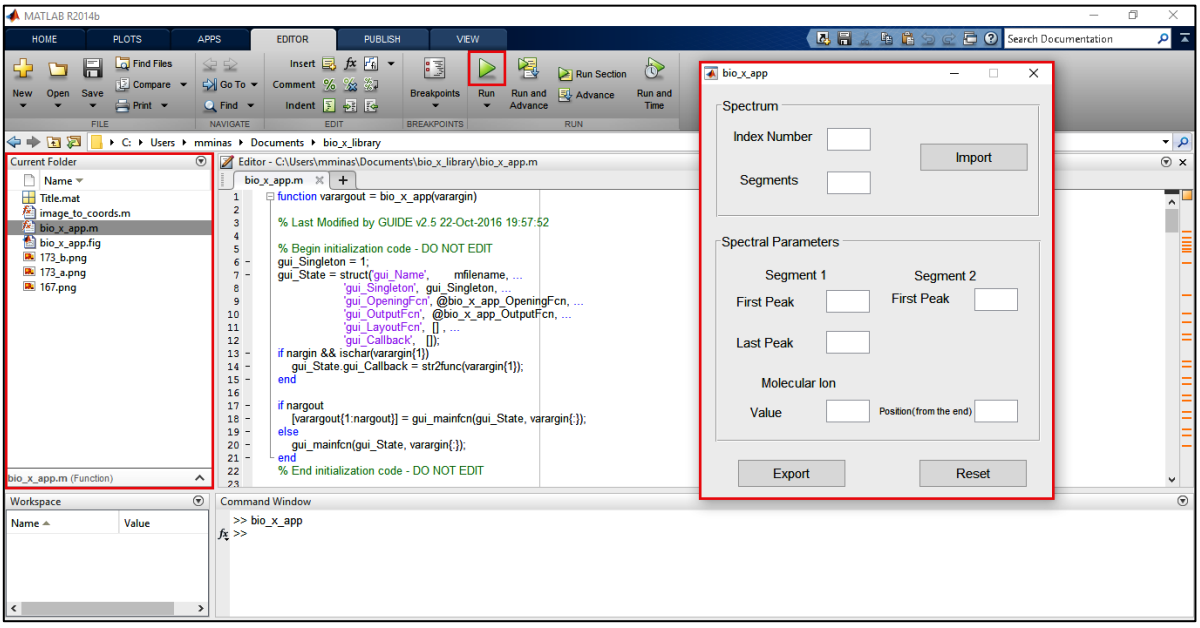

Figure 3.4: General arrangement of MatLab windows.

3.3.2. Application General Description

The visual configuration of the application has been designed to provide a quick and simple way of data insertion avoiding the time consuming procedure of repeated interactive questions. The application incorporates two separate panels and each one of them contains a certain number of fields, so the input information can be stored. Moreover, there are two push buttons; the “Export” button which stores the data into a .txt file format and the “Reset” button which clears out the whole configuration. The ‘Reset’ button can be pressed by the user anytime there is a need for corrections, regardless the progress of data insertion at that time.

The tabbing order is preset to provide the user with a convenient way of filling the fields with the right order and the push buttons execute their functions by using the keyboard space bar (instead of enter button). Alternatively, the input information can be inserted by browsing with the mouse over the interface and clicking inside every field of interest. The fields can only accept numerical characters, so any other format of typing will not be recognized since the program is designed to produce an error dialog box. If such a dialog box occurs the user has to close it and reset the application to allow re-initiation. These described attributes are shown in Figure 3.5.

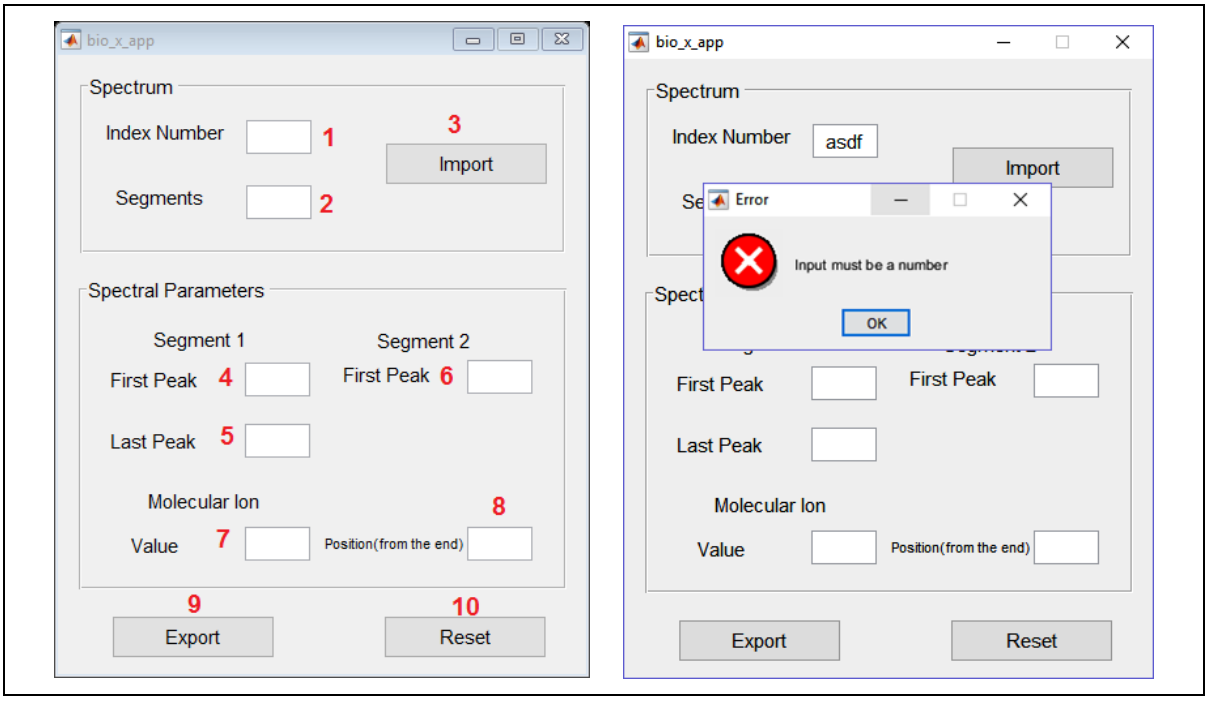

Figure 3.5: Tabbing order and error message using the application.

3.3.3. Filling up the Fields

The first panel is entitled “Spectrum” and requires the general information about the spectrum which is about to be accessed by the code. In the first field with the title “Index

Number", the user has to assign the numerical index of the spectrum. This index must be unique because the data will occupy a defined location in the library when extracted during the construction of the database.

The second field with the title "Segments" receives as input the number of parts that the spectrum is split to. The application in its current form can accept only the numbers 1 or 2. Usually, the spectra are consisted of one segment although two parts spectra can be found in the literature. This justifies the fact that the filing options for this field are limited only to these certain two ones (1 or 2).

The next step for the user is to press the "Import" push button. Once this button is activated, this operation will perform a browsing procedure inside the directory where the `bio_x_app` is located. At this point the user has to select the .png file which corresponds to the index number that has been declared according to the previous steps. In case of the number of segments has been declared as "2", the browsing procedure is programmed to perform two separate iterations automatically. During these iterations the user has to select one segment for each run.

The second panel is entitled "Spectral Parameters" and requires more detailed information about the spectra. Each field of this panel i.e. "First Peak" and "Last Peak" for "Segment 1" and "First Peak" for "Segment 2" must be completed with the actual values of the first and last peaks of the spectra. These values have to be derived from the bibliographic source that every spectrum of interest has been taken from, so the user has to refer to the respective sources of the spectra during the input of these values.

The area with the title "Molecular Ion" includes two fields. The first field i.e. "Value" requires the input of the actual value (m/z) of the molecular ion while the second field with the title "Position (from the end)" requires information about the position of the molecular ion from the end of the spectrum. This field can accept only the numbers 1, 2, 3 declaring the first, second or third position from the end of the spectrum respectively. The algorithm will ignore any peak beyond the molecular ion, so it is very important that the declaration of the "Position from the end", to be done correctly.

As is has been already mentioned, there are two cases regarding the number of the spectral segments. If the user decides that the number of segments will be "1", then the options "Segment 1 - Last Peak" and "Segment 2 - First Peak" will become inoperative and their visual appearance will be grayed out. These two fields have practical use only in the case of two segments, so this attribute has been developed to prevent users from possible wrong entries of data. A properly filled up interface referring to the example of C_{20} - tricyclic terpane is provided in Figure 3.6.

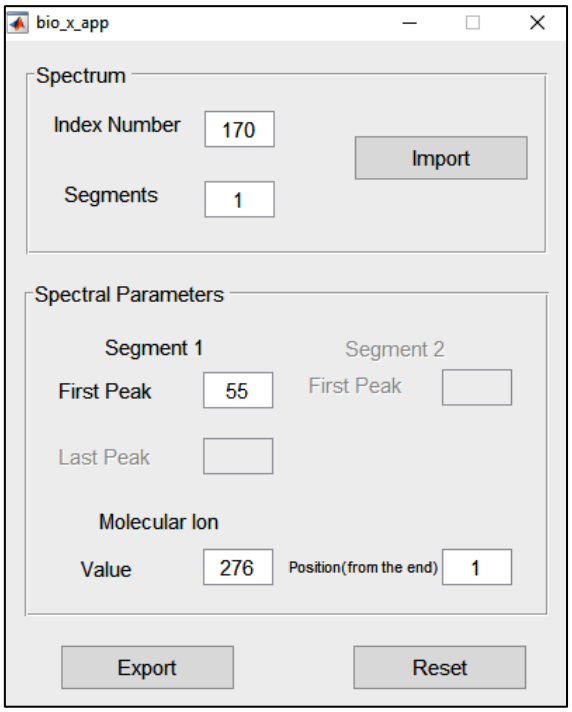

Figure 3.6: Properly filled interface for C_{20} - tricyclic terpane.

3.3.4. Storing the Output and Resetting the App

Once the fields are filled with all the necessary input values, the next step is to click on the "Export" button in order to acquire the stored information in a .txt file format. As was previously explained, this format is suitable for the computer which operates the biomarkers database. As it is obvious in Figure 3.7 the peaks coordinates are placed exactly after the introductory information and the relative abundance coordinates are normalized to 999000. The last step is to click on the "Reset" button to clear out the application interface in order that the user can repeat the whole procedure for another spectrum.

Although the algorithm that determines the coordinates of the peaks is able to produce them with great precision, i.e. four decimal digits, the respective information in each .txt file are set to be stored in the form of integer numbers. Consequently, there is a possibility that there will be some identical X (m/z) values in the output. So, after the output is produced, the user has to access the .txt file manually, inspect the coordinates and change to the next integer any identical m/z values that are possibly present.

The last thing that the user has to notice is the blank row at the end of the .txt files. Manual changes can potentially create more than one blank row at the end of the data. The user has to ensure that the blank row is only one by moving the cursor right after the last relative abundance value and pressing the "Delete" key several times. After that, the user must press the "Enter" key once. Finally, all the changes in the .txt file must be saved.

```
##TITLE=Library Entry 1 in C:\MassHunter\Library\TEST.L
##JCAMPDX=Revision 4.10
##DATA TYPE=MASS SPECTRUM
##SAMPLE DESCRIPTION=Performance evaluation sample
##NAMES=170
##CAS NAME=C20 tricyclic terpane (20tri, 22.916)
##MOLFORM=C20H36
##CAS REGISTRY NO=170
##MP= 0
##BP= 0
##MW= 276
##$RETENTION INDEX=0
##Library_RI= 0.00000e+000
##Library_RT= 22.916
##$CONDENSED SPECTRUM=NO
##NPOINTS= 15
##XYDATA=(XY..XY)
    55      187131
    69       23246
    82      235366
    83      195267
    95      447487
   109      265005
   123      505602
   149       34869
   163      327770
   177       44168
   191      999000
   192      197592
   220       37194
   261      281859
   276      522455
```

Figure 3.7: Output file of C_{20} - tricyclic terpane.

This page has been intentionally left blank.

4. Digitization Program Structural Background

The objective of the present Chapter is to describe the underlying operations of the graphical user interface presented in Chapter 3. These operations involve image analysis and processing but the entire coding is based on simple thoughts. The Chapter describes mainly the developed internal function which transforms the hardcopy data into precise mass spectral two dimensional coordinates.

4.1. Function `image_to_coords (I,FIRST,LAST,key)`

This function is an internal process of the `bio_x_app` which executes several operations. The first input argument, denoted as `I`, stands for the input image. The objective of the function is the extraction of the peaks coordinates (m/z values to relative abundances) by accessing any available spectrum image. It incorporates dedicated algorithms for each operation so it can be studied separately. Every algorithm segment is described in a different paragraph including the respective MatLab code abstract.

4.1.1. Image Enhancement

According to the RGB color model, every image consists of three components; red, green and blue and its visual appearance is a superposition of these components. Moreover, every image consists of the very basic elements; the pixels. Using the MatLab function `imread` [27] the image is translated to a three dimensional matrix. The first two dimensions represent the number of pixel rows and the number of pixel columns. The third dimension represents the number of the image RGB components. Thus, this dimension is always expressed by the number three, one for every RGB component.

The three dimensional matrix which is created after the analysis of colored images to a pixel level provides abundant information. This can be avoided through the use of MatLab function `rgb2gray` [28] which turns the image to a grayscale version of the original colored one. The new image is a two dimensional matrix where the first dimension is the number of pixel rows and the second is the number of pixel columns. The number of rows multiplied by the number of columns gives the image's total number of pixels.

The produced gray image consists of pixels with values inside the range of [0 - 255]. This particular range stands for the intensity of black color in a sense that the number zero expresses the total absence of color and the number 255 expresses the highest intensity of color resulting in a white appearance. In other words, this range includes 256 shades of gray color.

An effective image enhancement implies the reduction of the fuzziness of the image. This can be achieved by turning high and low intensity gray pixels into absolute white and absolute black ones respectively. A threshold of 220 intensity value has been selected in this study, which has been proved the optimal one after a lot of tests. Imposing this threshold, every pixel with an intensity value over 220, turns to white i.e. 255, while below 255 turns to 0 i.e. black. The outcome is a new version of the initial image in which all the pixels are white or black and there are not any intermediate values. The image enhancement algorithm is provided in Figure 4.1.

```
I = rgb2gray(I);                          % turning the image to gray scale

[n,m] = size(I);                          % dimensions of the image

[N_row,baseline] = min(sum(I,2));         % determining the baseline

y0 = baseline;
ymax = 1;
x0 = 1;
xmax = m;
I = I(ymax:y0,x0:xmax);                   % image cropped

I(baseline,:) = 0;                        % baseline zeroize

[n,m] = size(I);                          % dimensions of the cropped image

for i = 1:n;
  for j = 1:m;
       if I(i,j) > 220;
          I(i,j) = 255;          % replacement of gray pixels by white ones
       else I(i,j) = 0;
       end
  end
end
```

Figure 4.1: Image enhancement algorithm.

4.1.2. Peaks Coordinates Detection and De-noising

Once the image is analyzed to a pixel level, the next step is the detection of the coordinates where the spectrum peaks are located. In most of the cases, the images containing spectra are quite distorted because they do not originate from high quality printing devices. This causes lack of sharpness as regards the way the peaks appear. In such images the peak bars consist of three or four pixel columns with zero intensity i.e. black, instead of one bar which is the ideal situation. Consequently, the information fails to remain focused and it is significantly dispersed among several pixel columns, making the coordinates difficult to locate.

The imperfections of the image are found as randomly distributed accumulations of black colored spots. The pixel analysis of these spots shows that they usually occupy 5 or 6 rows and columns crossed together creating a square - like formation of zero intensity pixels. These imperfections are totally unwanted information during the process of coordinates tracing and they could be characterized as “noise”.

The appearance of the top most point of a maximum peak with a relative abundance value of 100, and a nearby image imperfection are provided in Figure 4.2. It can be clearly observed that the peak bar consists of 12 black pixel columns, instead of one that would be the ideal situation. Moreover, the imperfection consists of 27 black pixels. The ideal situation for this imperfection could be the total absence of it.

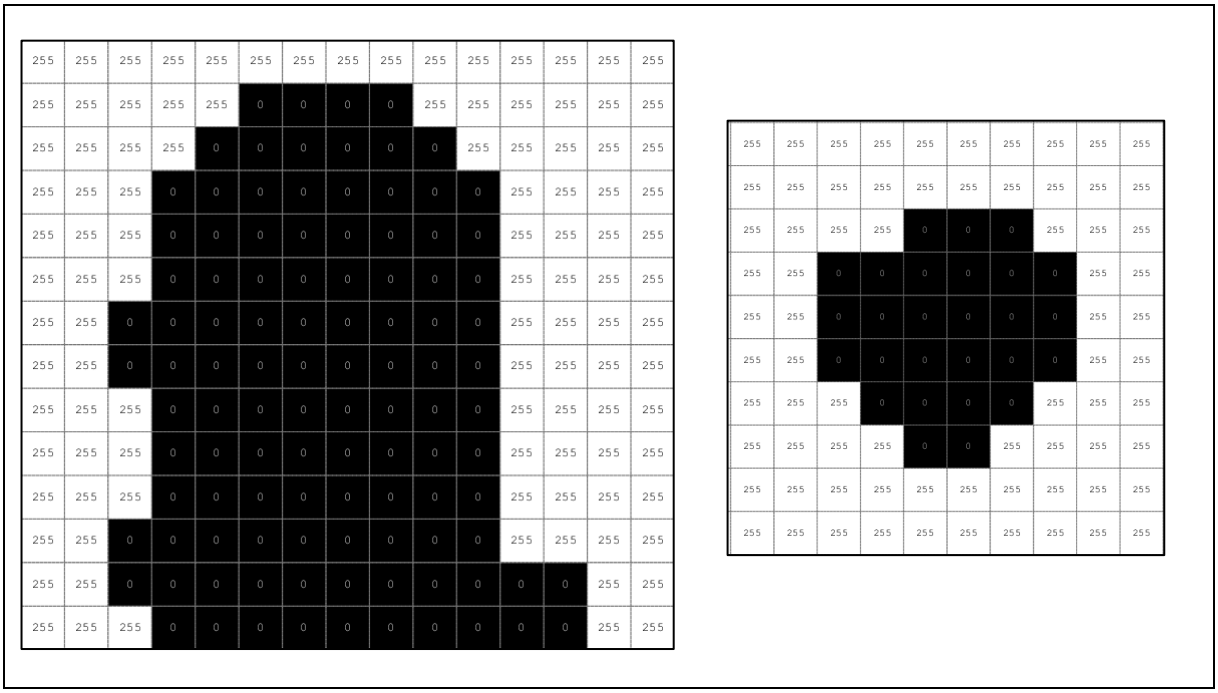

Figure 4.2: Pixel analysis of peak highest tip and imperfection appearance.

As for the peak bars, an effective methodology by which the most predominant black columns can prevail among their neighboring weaker ones is the only solution for the problem of information dispersal. The best criterion for the determination of the prevailing columns includes the summation of pixel intensities of every image column. After the summation, the columns with very small sum values can be considered potentially peak bars. On the other hand, columns with greater sum values suggest that white pixels participate in the formation of these columns reducing their possibility of being characterized as spectral peak bars.

The described procedure is achieved through the MatLab function `sum` [29]. This function performs summation among the first dimension of an array i.e. the rows of the matrix. As a

result of this action, a new matrix is produced. This matrix is a row vector which contains the results of the summations for every single column. Apparently, the number of columns remains the same as the number of columns of the original matrix.

The final screening can be achieved through the implementation of the MatLab function `findpeaks` [30]. This function spots the local maxima of the data that is applied to and stores both the maximum values and their locations. As a result, when the input argument for this function is the row vector of the sums of rows, the produced output arguments are two variables; one which reflects the values of maximum sums and one which reflects their respective locations, i.e. the column numbers of where they are positioned.

If the function is used as a simple command inside the code script, without any output arguments, it produces a visual result. Such an output is given in Figure 4.3 along with the original image in order to provide the reader with a comprehensive overview of described operation.

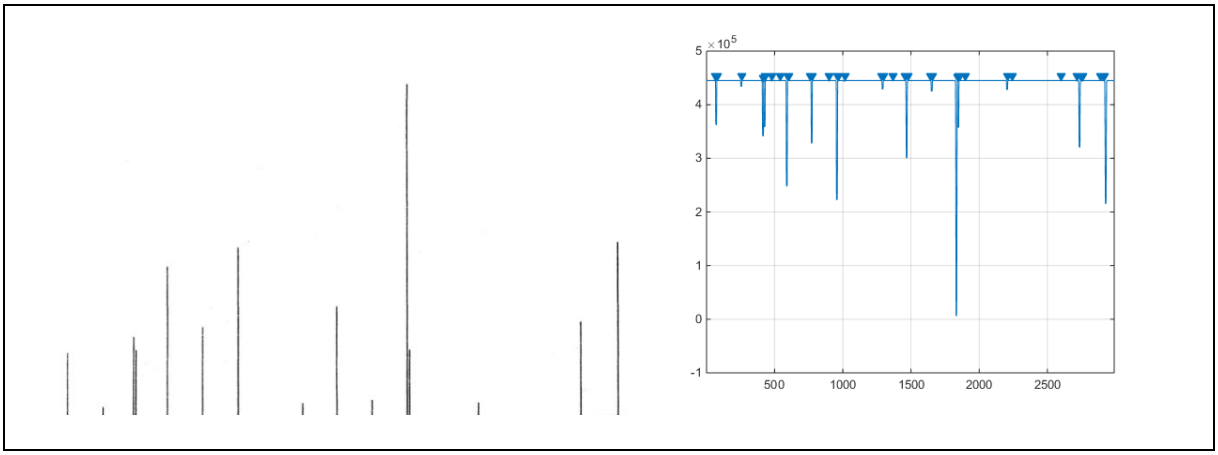

Figure 4.3: Local maxima of the sum of rows.

According to the previous analysis the `findpeaks` function traces the locations of high intensities where the white pixel predominance exists. This is the reason why the second section of Figure 4.3 looks like a mirror image of the original one. The implementation of the minus sign right before the `sum` function is considered necessary in order to force the `findpeaks` function to behave in the opposite way and to locate the local minima of the sums of rows.

The local minima of the sum of rows express the spectral peak bars because they have the lowest sums of pixels of their neighboring columns. Once the local minima are located, the algorithm returns both their values and their locations to an array. The produced output from this action is provided in Figure 4.4 along with the original image for a quick comparison. It can be clearly observed that the mirror image is inverted and now it resembles the original image.

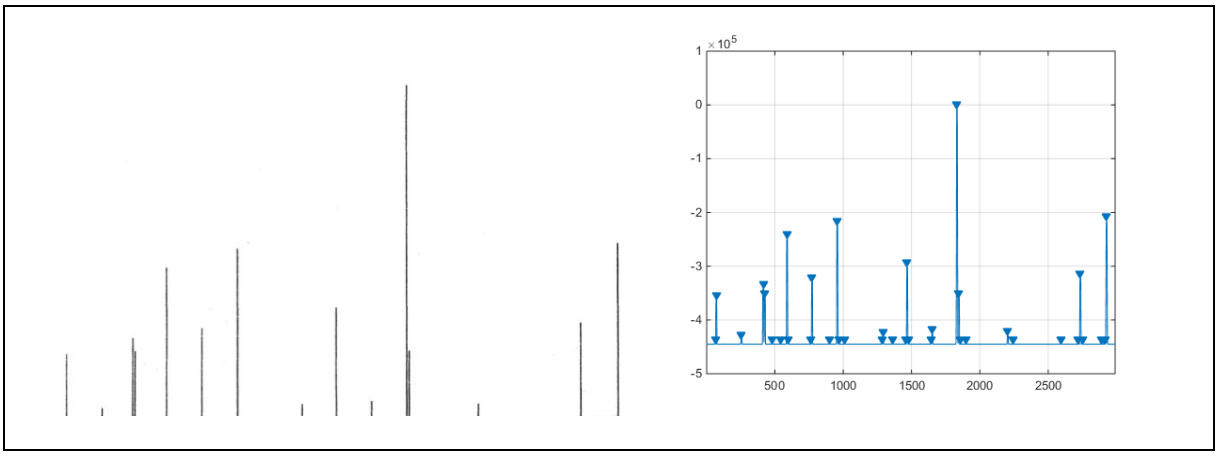

Figure 4.4: Local minima of the sum of rows.

As it is obvious in the second section of Figure 4.4 there is an accumulation of local minima near the baseline. These minima are produced due to the imperfections of the original image near the baseline. These imperfections must be regarded as noise i.e. excessive information. The developed algorithm has the ability to remove the unwanted information by normalizing the minimum peaks to 100 and subtracting the normalized values from the number 100. Consequently, the resulting values which are very small are reduced further to zero and then all the zeros are totally excluded. The algorithm of the described processes is given in Figure 4.5.

```
[pks,locs] = findpeaks(-sum(I)); % local maxima (minimum sum of pixels)
                                 and their locations

X = locs;                        % the peaks coordinates
Y = pks;

Y = Y*(100 / min(pks));          % normalization of minimum peaks to 100

Y = 100 - Y;                     % reduction of the small values

[q,r] = size(X);
[k,l] = size(Y);

x = reshape(X,[r,q]);            %reshaping the vectors
y = reshape(Y,[l,k]);

y(y<1)=0;                        % zeroize the very small values ( < 1)

coords = [x y];                  % joining coordinates into one vector

coords = coords(all(coords,2),:);   % removal of unwanted zeros

x = coords(:,1);

y = coords(:,2);
```

Figure 4.5: Peaks detection and de-noising algorithm.

4.1.3. Removal of Peaks beyond the Molecular Ion

The fourth input argument of the `image_to_coords` function denoted as “key”, defines the position of the molecular ion peak inside the spectrum considering that the peaks counting starts from the end to the beginning. This means that the molecular ion peak could be located on the first position or greater. In most cases the molecular ion is located on the first position from the end. There are also some cases where the molecular ion is located at the second or the third position from the end of the spectrum. Every other peak beyond the molecular ion represents its isotopic abundance.

The removal of the molecular ion isotopic abundance is considered necessary because it is abundant information for the code. The algorithm of the function has the ability of ignoring these unnecessary peaks for a given “key” value, e.g. if the position of the molecular ion peak from the end is two the algorithm will exclude every peak to the right of the second last peak. This is achieved through the reduction of their grayscale values to zero and then through the exclusion of the entire zero values which remain present. The actions referred to the determination of the molecular ion and the removal of peaks beyond the molecular ion algorithms are provided in Figure 4.6.

```
[rows_y,colls_y] = size(y);

                                % zeroize values after the molecular ion,
if key == 2;                      on condition on its spectral position
    y(rows_y,1) = 0;
else if key == 3;                 % second or third position from the end
     y(rows_y,1) = 0;
     y(rows_y - 1,1) = 0;
    end
end

coords = [x y];                   % joining coordinates into one vector

coords = coords(all(coords,2),:); % exclude sets of coordinates after the
                                    molecular ion

x = coords(:,1);
y = coords(:,2);
```

Figure 4.6: Definition of the molecular ion peak as the last peak of the spectrum algorithm.

4.1.4. X and Y- Axes Modification

With the previous analysis, the methodology by which the locations of the peaks are determined has been explained. These locations are expressed in pixel values because they represent the pixel columns where the local minima are located. On the contrary, the actual x - axis must represent the actual range expressed in m/z values, in order to be realistic. Thus,

one of the function's operations is the transformation of these pixel values to actual m/z values.

The dedicated algorithm segment for the realization of this requirement, calculates the relative distance between every matrix element and the first one in a scale from 0 to 100 in a sense that the first element will have a relative distance of zero and the last element will have a relative distance of 100. This action is performed two times; one for the pixel values and one for the actual values. Since the relative distance values are common, there can be equalization of the two datasets through the equation:

$$\left[\frac{X_i - X_{FIRST}}{X_{LAST} - X_{FIRST}}\right]_{actual} = \left[\frac{X_i - X_1}{X_m - X_1}\right]_{pixel} \tag{4.1}$$

where:

X_{FIRST}, X_{LAST} : input arguments representing the actual values.

X_m : m^{th} element corresponding to the number of matrix rows.

The only unknown variable of the Equation 4.1 is the term $[X_i]_{actual}$ so the equation can be rearranged to provide the solution. The new form of the equation is given by the following expression:

$$[X_i]_{actual} = \left[\frac{X_i - X_1}{X_m - X_1}\right]_{pixel} \cdot [X_{LAST} - X_{FIRST}]_{actual} + [X_{FIRST}]_{actual} \tag{4.2}$$

The last action is the normalization of the relative abundance values from pixel values to 100 in order to acquire the realistic relative abundance values. This can be achieved through the implementation of the following equation:

$$Y_{normalized} = \frac{Y}{\max\{Y\}} \cdot 100 \tag{4.3}$$

The algorithm which describes these actions is given in Figure 4.7.

```
actual_difference = LAST - FIRST;
[last,first] = size(x);
pseudo_difference = x(last) - x(first);

for pace = 1:last;
    distances(pace,:) = (x(pace,1) - x(1,1)) / pseudo_difference;
end
x = (distances*actual_difference) + FIRST;
coords = [x y];
x = coords(:,1);
y = coords(:,2);
y_correction_coefficient = 100/max(y);  % normalize y coordinates to 100
y = y_correction_coefficient*y;
coords = [x y];
```

Figure 4.7: Final acquisition of peaks coordinates.

As soon as the normalized version of the peaks coordinates is produced, one can clearly notice that the arrangement of this dataset resembles a discrete distribution of numbers. Although the examination of the visual appearance of the produced data is not necessary for the algorithm, the observation of it could be quite interesting. The optimal way for the produced coordinates data to be visualized is the implementation of the MatLab function `stem` [31].

By plotting the data with this function, one can realize that the peaks distribution has exactly the same shape with the original image, which proves that the developed algorithms work effectively. This comparison between the original scanned image and the produced one is provided in Figure 4.8.

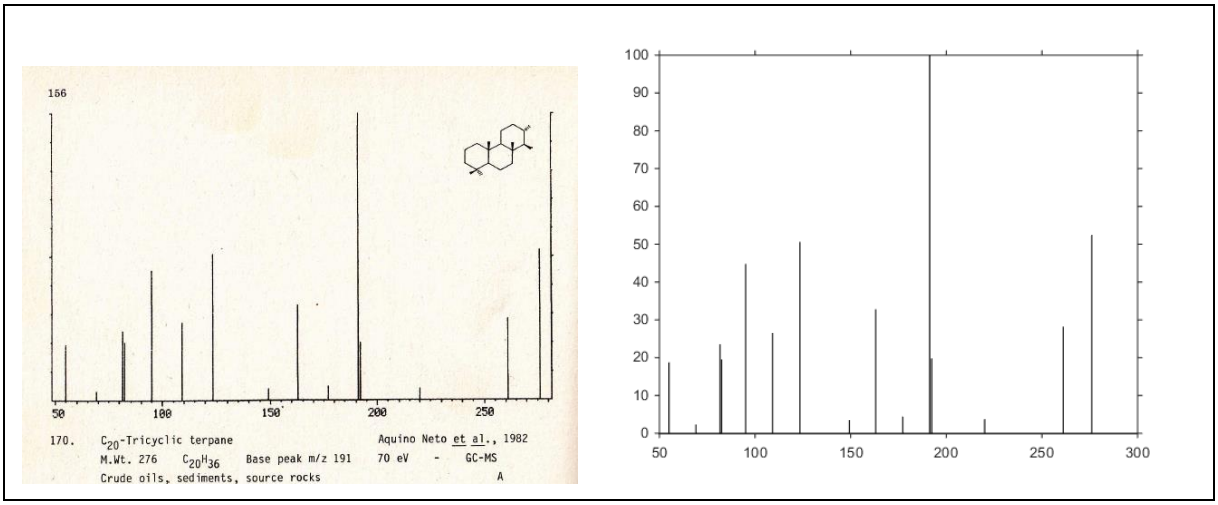

Figure 4.8: Comparison of the original image with the produced one.

4.1.5. Output File

The central computer which handles the mass spectra requires that the information will be in an ASCII format. Consequently, all the information must be stored as a .txt file. Moreover, the general arrangement of the characters inside the .txt file must follow a certain pattern which includes an introductory field printed right before the peaks' coordinates are displayed and the positioning of the coordinates at predetermined locations.

The creation of the introductory field requires manual typing of this introductory information. This text segment consists of 17 rows of information and each row starts with two hashtag symbols. The information regarding the peaks coordinates is stored right after the header text automatically. The header text has to be saved as a MatLab .mat file, in order that it can be accessed and operated by the code in an efficient way. The name of the file for the needs of this study has been arbitrarily selected to be "Title.mat" and the respective text is provided in Figure 4.9.

```
##TITLE=Library Entry 1 in C:\MassHunter\Library\TEST.L
##JCAMPDX=Revision 4.10
##DATA TYPE=MASS SPECTRUM
##SAMPLE DESCRIPTION=Performance evaluation sample
##NAMES=HP
##CAS NAME=Insert Component Name
##MOLFORM=C27H46
##CAS REGISTRY NO=000000-00-0
##MP= 0
##BP= 0
##MW= 0
##$RETENTION INDEX=0
##Library_RI= 0.00000e+000
##Library_RT= 00000.00
##$CONDENSED SPECTRUM=NO
##NPOINTS= 34
##XYDATA=(XY..XY)
```

Figure 4.9: Header text for the Title.mat file.

The peaks coordinates must be located right below the header text. The algorithm accesses the contents of the Title.mat file and the peaks coordinates are stored automatically exactly below the header text. The only restriction regarding this arrangement is that the X coordinates must project 6 points from the beginning of the text and the Y coordinates 12 points from the end of the last digit of every X coordinate. Moreover, the Y coordinates must be normalized to 999000 through the equation:

$$Y_{normalized} = \frac{Y}{\max\{Y\}} \cdot 999000 \qquad (4.4)$$

The realization of these actions is provided in Figure 4.10.

```
dlmwrite('index.txt', coords);                  % storing the coordinates
load('Title.mat','TITLE');                      % loading the header text

fileID = fopen('index.txt','w');
formatSpec = '%s \n';
[nrows,ncols] = size(TITLE);

for row = 1:nrows
    fprintf(fileID,formatSpec,TITLE{row,:});
end

norm_x = coords(:,1);
norm_y = ((coords(:,2))*999000)/max(coords(:,2)); % normalizing to 999000
norm_coords = [norm_x norm_y];

fprintf(fileID,'%6.0f%12.0f\n', norm_coords');  % determination of spaces
fclose(fileID);

newName = [int2str(index_number),'.txt'];         % renaming the file
movefile('index.txt',newName)
```

Figure 4.10: Storing of coordinates information.

This page has been intentionally left blank.

5. Library Construction

The objective of the present Chapter is to provide the reader with a description of the dedicated code developed for the generation of a biomarkers dataset in the form of library. This specific code requires as input information, all the .txt files created by using the methodology described in Chapter 3, along with a utility function, in order to operate. The operation of the utility function (`stretch.m`) is explained in a separate paragraph. The necessary files for the code are given in Table 5.1.

Table 5.1: Necessary files for the matching algorithm.

File	Use
`library_biomarkers.m`	`main script`
`stretch.m`	`expansion function`
`all the created txts`	`database input`

5.1. Creation of Labeling Rows and Ion Column

The first operation of the code is to clear out the workspace memory. All the variables occupy assigned workspace sectors once they are produced, so the workspace must be cleared out in order that the program can run more than one times. So, the presence of `clear all` command in the beginning of the script is considered mandatory.

The first operation of the code is the construction of a separate set of rows of data which includes all the necessary titles for the mass spectral properties. So, the challenge of the initial part of the script is to extract specific information of the header text of every .txt file. This information includes the index number, the name, the molecular formula, the melting point, the boiling point, the molecular weight and the retention time of the GC procedure, of the biomarkers.

The code performs browsing through the directory, it accesses the whole set of .txt files alphanumerically and finally acquires all the desired information. It must be mentioned that all the available biomarkers data have been derived from [26], and they are provided to the reader through the Appendix A.

The code is programmed to acquire the information from certain rows of the .txt files. The rows of interest are the 5^{th}, 6^{th}, 7^{th}, 9^{th}, 10^{th}, 11^{th} and the 14^{th} and they reflect the mentioned properties. The code avoids every property label by skipping the characters before the desired data. So, the starting point of reading for these rows is the 9^{th}, 12^{th}, 11^{th}, 6^{th}, 6^{th}, 6^{th}, 14^{th} character, respectively. An overview of this action is provided in Figure 5.1 illustrating the

.txt file for the C_{20}-Tricyclic terpane. The information of interest of the header text is marked with yellow color.

```
##TITLE=Library Entry 1 in C:\MassHunter\Library\TEST.L
##JCAMPDX=Revision 4.10
##DATA TYPE=MASS SPECTRUM
##SAMPLE DESCRIPTION=Performance evaluation sample
##NAMES=170
##CAS NAME=C20 tricyclic terpane (20tri, 22.916)
##MOLFORM=C20H36
##CAS REGISTRY NO=170
##MP= 0
##BP= 0
##MW= 276
##$RETENTION INDEX=0
##Library_RI= 0.00000e+000
##Library_RT= 22.916
##$CONDENSED SPECTRUM=NO
##NPOINTS= 15
##XYDATA=(XY..XY)
    55      187131
    69       23246
    82      235366
    83      195267
    95      447487
   109      265005
   123      505602
   149       34869
   163      327770
   177       44168
   191      999000
   192      197592
   220       37194
   261      281859
   276      522455
```

Figure 5.1: Biomarkers general information acquisition.

The ion labeling column is predefined to range within [41-700], so the code includes the insertion of a matrix with the numbers 41-700 which is placed in the first column of the library. The conclusion is that the first 7 rows of the library are informative data, and the first column represents the labeling column for the molecular ions.

5.2. Creation of Dataset

Apart from the informative data, the desired form of a uniform dataset requires the extraction of the respective coordinates' values (m/z - relative abundance) and the placement of the relative abundance values next to each other as columns, in a way that each column reflects the numerical data of a specific .txt file. Moreover, the rows of the dataset represent the m/z values. The range of interest is within [41 - 700] so the first row of the dataset stands for the m/z value "41" and the last one stands for the value "700".

Because of the fact that the dataset requires standard dimensions in order to behave as an array, all the intermediate values between the successive coordinates of the .txt files must be zero. In order to overcome this situation the code calls the `stretch` function which is explained in §5.2.1.

For the acquisition of the numerical data, several characteristics of the .txt files created with the described procedure of Chapters 3 and 4 must be taken into account for the construction of a uniform dataset. Every .txt file contains the header text which is followed by the a set of values arranged in a way that the first column is the set of the m/z values and the second column is the set of the respective relative abundances normalized to 999000.

The header text occupies 17 rows, so the sets of values start from the 18th row of the file. Thus, the access of the files must take place with an offset of 17 rows. The column offset is set to zero because the recorded information begins exactly at the top left corner of the .txt file. The developed algorithm accesses all the available .txt files, skips the header by using the offset setting, and finally reads and extracts the coordinates.

Then, it uses the `stretch` function to bring all the coordinate sets to the same size (1000x1). The m/z values of interest ranges within [41-700] so the values outside this range are programmed to be removed. Once the numerical data of the library are organized, the code assigns to each column the respective properties in the form of labeling information, which were stored according to the discussion in §5.1. A segment of the produced output of this code is given in Figure 5.2.

'Index Num...	'168 '	'170'	'173'	'175 '	'177 '	'178 '	'179 '	'180 '	'181 '	'182 '	'183 '	'186 '	'187 '
'Name'	'C19 tricycli...	'C20 tricycli...	'C21 tricycli...	'C22 tricycli...	'C23 tricycli...	'C24 tricycli...	'C25 tricycli...	'C26 tricycli...	'C28 tricycli...	'C29 tricycli...	'C30 tricycli...	'17β(H)-22,...	'17b 21b 30...
'Molecular ...	'C19H34'	'C20H36 '	'C21H38 '	'C22H40 '	'C23H42 '	'C24H44 '	'C25H46 '	'C26H48 '	'C28H50 '	'C29H52 '	'C30H54 '	'C27H46 '	'C29H50 '
'MP'	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '
'BP'	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '
'Molecular ...	' 262 '	' 276 '	'290 '	' 304 '	' 318 '	' 332'	' 346 '	' 360 '	' 388 '	' 402 '	' 416 '	' 370 '	' 398 '
'Library Ret...	' 21.011 '	' 22.916'	' 25.153'	' 27.414'	' 30.269 '	' 31.904'	' 35.452 '	' 38.173 '	' 44.632'	' 47.075'	' 50.147 '	' 51.565 '	' 00000.00 '
41	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0
53	0	0	0	73133	0	39801	0	49441	0	62257	27589	76735	0
54	0	0	0	0	0	19900	0	15996	39091	30404	0	13030	0

Figure 5.2: Constructed library structure.

The final action of this code is to save the result as a .mat file. This is very important because the library will be used for the matching procedures or any other kind of study. The code for the library construction is provided in Figure 5.3.

```
clc;clear all
row_labels = cellstr([ 'Index Number          ';
                       'Name                  ';
                       'Molecular Formula     ';
                       'MP                    ';
                       'BP                    ';
                       'Molecular Weight      ';
                       'Library Retention Time'

mz_values = transpose([41:700]);

LIBRARY = dir('*.txt');   % access of .txt files and acquire the length

for j = 1:length(LIBRARY);

       c = textread(LIBRARY(j,:).name, '%s','delimiter', '\n');

      info(j,:) =
[{c{5}(9:end)};{c{6}(12:end)};{c{7}(11:end)};{c{9}(6:end)};...
                {c{10}(6:end)};{c{11}(6:end)};{c{14}(14:end)}];

     M = dlmread(LIBRARY(j,:).name,' ',17,0);% access after the header
    [m,n] = size(M);               % coordinates information acquisition

        for i = 1:m;

        locations(i,:) = find(M(i,:));

        end

        for i = 1:m;

        M(i,1) = M(i,locations(i,1));

        M(i,n) = M(i,locations(i,2));

        end

  M = [M(:,1) M(:,n)];

  LIB(j,:) = stretch(M);                          % expanding to [1000,1]

end

info = (transpose(info));

labels = [row_labels info];

LIBRARY = transpose(LIB);

LIBRARY(701:1000,:) = [];              % cropping below 40 and above 700

LIBRARY(1:40,:) = [];

LIBRARY = [mz_values LIBRARY];

LIBRARY = [labels;num2cell(LIBRARY)];

save('LIBRARY.mat')
```

Figure 5.3: Code for biomarkers' library construction.

5.2.1. Function stretch (array)

The use of this function is very important for the dataset construction. As it was previously mentioned, the column vectors must be of the same size. So, there is a need of filling up with zeros the intermediate spaces between the successive coordinates. The `stretch` function, receives as an input argument the array of the coordinates set (m/z and relative abundance values) and expands it, returning as an output argument a new array with 1000 rows and 1 column.

The function involves the pre-allocation method to define a zero matrix with a size of 1000 rows and 2 columns. Then, it accesses the input array and captures the m/z value of every row. Every row is assigned in the zero matrix according to the respective m/z value, replacing the zero values. After, this action the m/z values column is omitted and the final output is a 1000x1 array which contains only relative abundance values. The algorithm concept and the code are provided in the following equation and the function code is provided in Figure 5.4.

$$\begin{bmatrix} m/z & \sim 100 \\ m/z & \sim 100 \\ m/z & \sim 100 \\ . & . \\ . & . \\ . & . \\ m/z & \sim 100 \end{bmatrix} \xrightarrow{?m/z} \begin{matrix} 1 \\ 2 \\ 3 \\ . \\ . \\ . \\ . \\ . \\ . \\ 1000 \end{matrix} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ . & . \\ . & . \\ . & . \\ . & . \\ . & . \\ . & . \\ 0 & 0 \end{bmatrix} \tag{5.1}$$

```
function [ expanded_matrix ] = stretch( array )

tank = zeros(1000,2);                          % preallocation matrix

tank(array(:,1)) = array(:,1);

tank(find(tank),2) = array(:,2);

expanded1 = tank(:,2);

expanded_matrix = expanded1;

end
```

Figure 5.4: Stretch function code.

This page has been intentionally left blank.

6. Matching Process

This Chapter aims to help the reader to understand the way that the final identification of a biomarker molecule can be conducted. The concept of the identification procedure is based on the comparison of an unknown spectrum with the spectra that are included in a dataset in the form of a library. The comparison can be done with the use of several mathematical approaches like the similarity indices or pairwise distances. The basic condition for the comparison applicability is that the compared elements must be of the same format. This requirement is met by several preparative actions of the unknown spectrum. The chapter begins with the description of these actions, continues with the matching algorithms and ends with a report of the produced results.

6.1. Pretreatment of the Unknown Data

The unknown data must be in the form of a .txt file in order to meet the requirements of the matching algorithm. The matching algorithm requires that both the library components and the unknown data must have the same structure in a sense that the unknown data resembles a library component. So, first of all these data must be prepared for the comparison with the library. In this work spectral data files created by the Agilent Chemstation software are employed. Files from other MS packages can also be used after a similar treatment. The initial form of the unknown spectrum is shown in Figure 6.1.

```
Scan 5398 (55.748 min): RIGAKIS_2015_SAT_TIC.D\data.ms (-5389)
RIGAKIS_2015_SAT_TIC
Modified:subtracted
m/z Abundance
50.90   244.0
53.00   56.0
54.00   67.0
55.10   1915.0
57.10   267.0
59.00   39.0
67.05   684.0
68.00   412.0
69.10   1514.0
70.10   627.0
72.10   29.0
73.00   217.0
75.00   42.0
77.00   282.0
80.05   375.0
81.10   2035.0
82.05   597.0
83.10   230.0
84.10   152.0
```

Figure 6.1: Initial form of the unknown spectrum data file.

There are three issues that the code must deal with before the spectrum is compared with the library. As it can be observed the first four rows of the .txt file do not contain numerical data. These rows are unnecessary information for the matching process, so they must be removed. The other two issues have to do with the m/z values and the relative abundance values. The m/z values must not contain decimal digits while the relative abundance values must be normalized to 999000. These three issues are covered by a dedicated script with the name `plug_in.m` which is provided in Figure 6.2.

```
clc;clear all

[filename, filepath]=uigetfile({'*.txt'},'Select  File'); % selection

coords = dlmread(filename,' ',4,0); % reading offset of 4 rows

norm_x = coords(:,1);

norm_y = ((coords(:,2))*999000)/max(coords(:,2)); % normalizing to 999000

norm_coords = [norm_x norm_y]; % coordinates aquisition

dlmwrite('temp.txt', norm_coords); % storing coordinates to a temp file

fileID = fopen('temp.txt','w');

fprintf(fileID,'%6.0f%12.0f\n', norm_coords'); % determination of spaces

fclose(fileID);

movefile('temp.txt',filename)       % file rename
```

Figure 6.2: Code segment for the preparative procedures (`plug_in.m`).

The script browses the directory and prompts the user to open the file that contains the data of the unknown spectrum. So, the both the script and the unknown spectrum file must be located in the same directory. Once the file has been specified the code performs the three discussed preparative actions.

The removal of the unnecessary four informative rows is achieved through the determination of row offset during the file access. The offset is set to number 4, so the reading takes place from the fifth row. The column offset is set to zero because there is no need for avoiding any information as regards the files columns. The decimal digits of the m/z values are removed automatically as the code rewrites the values to a new file. The normalization of the relative abundance values is achieved through the implementation of the Equation 4.4.

After these preparative actions, the output file must be checked visually by the user for double entries. If there are any double entries they must be corrected so the file will contain

only unique m/z values. Once again the user must be very careful with the very final row of the file which has to be blank. More than one blank row at the end of the file is not permitted, so a good practice for the user is to press several times the “Delete” key, and then the “Enter” key once.

6.2. Final Matching

The code that has been developed for the data matching requires the simultaneous presence of several files into the same directory in order to run. These files are given in Table 6.1.

Table 6.1: Necessary files for the matching algorithm.

File	Use
`final_match.m`	matching code
`stretch.m`	expansion function
`LIBRARY.mat`	biomarkers library
`unknown``.txt`	spectrum to be compared

The file `final_match.m` is the main script for the final matching process. The function `stretch` expands the acquired coordinates to a vector with dimensions 1000x1. The complete description of this function has been already described in §5.2.1. The file LIBRARY.mat is the generated biomarkers database, and the unknown.txt is the unknown spectrum after the preparative actions that it has been subjected to, according the discussion in §6.1.

The first part of the matching process contains the selection of the unknown spectrum file, so the initiation of the code includes the browsing through the directory and the prompt to the user in order to select the specific unknown spectrum file. Once this action is made, the code calls the stretch function which expands the captured coordinates to a vector with dimensions 1000x1. The next step includes cropping the first 40 and the last 300 rows in order to change the unknown spectrum vector to a similar and comparable form with the library.

Once the unknown spectrum is pretreated, the code continues to the next step loading the constructed biomarkers library. Then the code determines the frame of the numerical field which is necessary for the calculations. This numerical field, denoted as Lib_num_core includes all the data of the library except the first 7 labeling rows and the first labeling column and all the matching calculations are conducted by accessing this numerical field. The

numerical field and the respective code segment for these actions are given in Figure 6.3 and Figure 6.4.

Index Num...	'168 '	'170'	'173'	'175 '	'177 '	'178 '	'179 '	'180 '	'181 '	'182 '	'183 '	'186 '	'187 '
'Name'	'C19 tricycli...	'C20 tricycli...	'C21 tricycli...	'C22 tricycli...	'C23 tricycli...	'C24 tricycli...	'C25 tricycli...	'C26 tricycli...	'C28 tricycli...	'C29 tricycli...	'C30 tricycli...	'17β(H)-22,...	'17b 21b 30...
'Molecular ...	'C19H34'	'C20H36 '	'C21H38 '	'C22H40 '	'C23H42 '	'C24H44 '	'C25H46 '	'C26H48 '	'C28H50 '	'C29H52 '	'C30H54 '	'C27H46 '	'C29H50 '
'MP'	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '
'BP'	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '	' 0 '
'Molecular ...	' 262 '	' 276 '	'290 '	' 304 '	' 318 '	' 332'	' 346 '	' 360 '	' 388 '	' 402 '	' 416 '	' 370 '	' 398 '
'Library Ret...	' 21.011 '	' 22.916'	' 25.153'	' 27.414'	' 30.269 '	' 31.904'	' 35.452 '	' 38.173 '	' 44.632'	' 47.075'	' 50.147 '	' 51.565 '	' 00000.00 '
41	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0
53	0	0	0	73133	0	39801	0	49441	0	62257	27589	76735	0
54	0	0	0	0	0	19900	0	15996	39091	30404	0	13030	0

Figure 6.3: Numerical field determination.

```
clc;clear all
[filename, filepath]=uigetfile({'*.txt'},'Select  File'); % selection

unknown_coords = dlmread(filename);
unknown_coords = stretch(unknown_coords); % expansion
unknown_coords(701:1000,:) = [];          % tail cropping
unknown_coords(1:40,:) = [];              % head cropping

spectrum = unknown_coords;      % unknown spectrum coordinates acquisition

load LIBRARY.mat                % library loading

[lib_r,lib_c] = size(LIBRARY);
LIB_num_core = cell2mat(LIBRARY(8:lib_r,2:lib_c));% library num core
[r,c] = size(LIB_num_core);
```

Figure 6.4: Initial code segment for the matching procedure.

6.3. Matching Calculations

The matching calculations are based on five mathematical approaches i.e. the Pearson correlation coefficient, the vector cosine, the Euclidean distance, the city-block distance and the Chebychev distance. The first two approaches are called similarity indices whereas the three latter ones are pairwise distances. There are no any certain restrictions for the application of these mathematical approaches apart from the fact that the three mentioned distance functions involve the need of horizontally arranged vectors to conduct the pairwise distance calculation, so all the compared vectors must be horizontally orientated. Other than this restriction the algorithm does not have any requirements. The code segment for the matching calculations is given in Figure 6.5.

```
for z = 1:c;
  L = LIB_num_core(:,z);

  S = spectrum;

  R  = corrcoef(S,L);

  correlation_coefficient(z,:) = R(1,2);

  Dist_euclidean(z,:) = pdist2(transpose(S),transpose(L));

  Dist_chebychev(z,:) = pdist2(transpose(S),transpose(L),'chebychev');

  Dist_cityblock(z,:) = pdist2(transpose(S),transpose(L),'cityblock');

  inner_product(z,:) = dot(S,L);

  magnitude_product(z,:) = norm(S)*norm(L);
 end
```

Figure 6.5: Matching calculations loop.

6.3.1. Pearson Correlation Coefficient

The correlation coefficient is a measure of linear dependence between two random variables (X, Y) and is expressed by the fraction of their covariance over the product of their standard deviations. The value of the correlation coefficient moves inside the range of [-1, 1], considering -1 and 1 the total negative and total positive linear dependence respectively, and 0 the value where there is no linear dependence at all [32]. The general form of the Pearson correlation coefficient is given by:

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \tag{6.1}$$

Replacing the covariance notation with its similar expression in terms of the mathematical expectation and the means of the random variables, the correlation coefficient can be written as the fraction of the expectation of the product of the random variables minus their mean values, over the product of the standard deviations of the two random variables.

$$\rho_{X,Y} = \frac{\mathrm{E}\left[(X-\mu_X)(Y-\mu_Y)\right]}{\sigma_X \sigma_Y} \tag{6.2}$$

6.3.2. Vector Cosine

With the use of the definition of variance in the denominator, the Equation 6.2 can take a more explicit form:

$$\rho_{X,Y} = \frac{E\left[(X-\mu_X)\cdot(Y-\mu_Y)\right]}{\sqrt{Var[X]}\cdot\sqrt{Var[Y]}} = \frac{E\left[(X-\mu_X)\cdot(Y-\mu_Y)\right]}{\sqrt{E\left[(X-\mu_X)^2\right]}\cdot\sqrt{E\left[(Y-\mu_Y)^2\right]}} \tag{6.3}$$

Given that the majority of the observations are equal to zero, the terms μ_x and μ_y can be considered approximately zero. This convention leads to the following formula:

$$\rho_{X,Y} = \frac{E[X\cdot Y]}{\sqrt{E\left[X^2\right]}\cdot\sqrt{E\left[Y^2\right]}} \tag{6.4}$$

Considering that X and Y are vectors, their product will be a scalar measure. This scalar measure is equal to its expectation. Consequently, the expectation of the product of the two variables can be written as their dot product. Moreover, the magnitudes of the two vector values can be determined by calculating the square roots of their squared components by using the expressions:

$$\|X\| = \sqrt{X_i^2 + X_j^2 + \ldots + X_n^2} \quad \text{and} \quad \|Y\| = \sqrt{Y_i^2 + Y_j^2 + \ldots + Y_n^2} \tag{6.5}$$

Since the results of these calculations will also be scalar, their expected values will be constant numbers. In conclusion, the product of the standard deviations can be expressed as the product of the vector magnitudes.

Finally, taking into account that the dot product of two vectors is equal to the product of their magnitude multiplied by the cosine of the angle θ between them, we can obtain the relationship between the correlation coefficient and the cosine of the angle θ, which will be:

$$\rho_{X,Y} = \frac{E[X\cdot Y]}{\sqrt{E\left[X^2\right]}\cdot\sqrt{E\left[Y^2\right]}} = \frac{X\bullet Y}{\|X\|\cdot\|Y\|} = \cos\theta_{X,Y} \tag{6.6}$$

6.3.3. Distances

The use of distances is another practice for matching data. The realization of this approach involves the need of the mathematical background of several expressions that can be obtained from the Minkowski distance. Considering two vectors X = (x_1,x_2,…,x_n) and Y = (y_1,y_2,…,y_n) the Minkowski distance $d_{1,2}$ between them, is given by the formula:

$$d_{1,2} = \left(\sum_{i=1}^{n}\left|x_i - y_i\right|^p\right)^{\frac{1}{p}} \tag{6.7}$$

For p=1, the Equation 6.7 provides the Manhattan or city block distance between two vectors X and Y in an n-dimensional real vector space with fixed Cartesian coordinate system,

expressing the sum of lengths of the projections of the line segment between the points onto the coordinate axes, and it is given by the following formula [33]:

$$d_{1,2} = \sum_{i=1}^{n} |x_i - y_i| \tag{6.8}$$

For p=2, the Equation 6.7 provides the Euclidean distance which is the intrinsic metric of the Euclidean space. In other words the Euclidean distance provides the length line segment which connects X and Y [33] and it is given by the following formula:

$$d_{1,2} = \left(\sum_{i=1}^{n} |x_i - y_i|^2 \right)^{\frac{1}{2}} \tag{6.9}$$

For p=∞, the Equation 6.7 provides the Chebyshev distance [34]. The infinity symbol means that the distance is equal to the maximum absolute difference of all the differences between the two examined vectors. The Chebychev distance can be described by the formula:

$$d_{1,2} = \max_i \left(|x_i - y_i| \right) \tag{6.10}$$

6.4. Matching Calculations Output

The final step of the code produces the output of the matching calculations. The output is programmed to give sorted lists of biomarkers for the first five predominant ones. The lists begin with the most correlated to the least correlated biomarker. The concept of this code segment is that there is a gathering of all the calculated values of matching which takes place between the unknown spectrum and the library biomarkers. So, there is a matching value for every comparison since all the matching values are equal to the population of biomarkers inside the library. The variable “cutter” determines the length of the output and it is equal to the number of correlated biomarkers as it has been already mentioned. The code for these last actions is provided in Figure 6.6.

```
cosine_theta = inner_product ./ magnitude_product;

cutter = 5;

combo_labels = transpose(LIBRARY(1:2,2:lib_c));

correlation_coefficient = sortrows([combo_labels
num2cell(correlation_coefficient)],-3);
correlation_coefficient =
cell2table(correlation_coefficient(1:cutter,:),'VariableNames',{'Index','B
iomarker','Criterion'})

cosine_theta = sortrows([combo_labels num2cell(cosine_theta)],-3);
cosine_theta =
```

```
cell2table(cosine_theta(1:cutter,:),'VariableNames',{'Index','Biomarker','
Criterion'})

Dist_euclidean = sortrows([combo_labels num2cell(Dist_euclidean)],3);
Dist_euclidean =
cell2table(Dist_euclidean(1:cutter,:),'VariableNames',{'Index','Biomarker'
,'Criterion'})

Dist_chebychev = sortrows([combo_labels num2cell(Dist_chebychev)],3);
Dist_chebychev =
cell2table(Dist_chebychev(1:cutter,:),'VariableNames',{'Index','Biomarker'
,'Criterion'})

Dist_cityblock = sortrows([combo_labels num2cell(Dist_cityblock)],3);
Dist_cityblock =
cell2table(Dist_cityblock(1:cutter,:),'VariableNames',{'Index','Biomarker'
,'Criterion'})
```

Figure 6.6: Code segment for the final output of the matching procedure.

7. Results

The objective of the last Chapter is the presentation of the constructed library performance which is examined both through the unknown matching algorithm of the Agilent Chemstation software and the algorithms that have been already described in the previous Chapter. The identification results refer to a specific actual sample. At the end of the Chapter a concluding paragraph is provided which gives the opportunity for a brief discussion over the value of this whole effort.

7.1. Matching Calculations Report

The main issue of the existing libraries is the lack of mass spectral data regarding the compounds that can be used for biomarkers identification. Although these libraries include a plenty of substances, the solution for this problem still remains. For example, one of the most popular libraries i.e. NIST Chemistry WeBook [25] which contains over 40.000 species, does not provide a dedicated set of data for biomarkers identification.

For the time being, the developed library of the present work contains 114 entries referring to biomarkers compounds that have been obtained only from the literature source of [26]. According to this source these compounds are classified to separate groups. Despite the fact that during the construction of the dataset all the available compounds have been considered one group, the understanding of the library contents requires a classification of its entries which is provided in Table 7.1.

Table 7.1: Library compounds classification.

Group	Compounds	Index Numbers
Aromatic Hopane Derivatives	4	226 - 229
Aromatic Steroid Hydrocarbons	18	356 - 373
Hopanes	27	168, 170, 173, 175, 177 - 183, 186, 190, 191 - 195, 198, 199, 206 - 209, 235, 236, 242
Hopenes	10	216 - 225
Lupanes	5	230 - 234
Oleananes	7	237 - 241, 243, 244
Pentacyclic Triterpanes	10	187 - 189, 197, 200 - 205
Rearranged Steranes	9	346 - 348, 350 - 355
Regular Steranes	17	303 - 305, 308 - 310, 327 - 329, 331, 332, 336 - 341
Steranes	6	306, 307, 326, 335, 343, 349
Tetra-Penta Cyclic Aromatics	1	275

After the comparison of the unknown mass spectra with the developed library, the produced matching results prove the validity of the library. The unknown matching algorithm of the Agilent Chemstation software has managed to identify the compounds of the actual sample with a great deal of accuracy. The majority of the matching results have the best ranking which is the first position, expressing a perfect identification. However, there are four species that do not receive the first place (two second places, one third and one ninth).

The other algorithms (similarity indices and distances) perform less accurately compared to the unknown algorithm of the Agilent Chemstation software. This has to do with the fact that these algorithms are extremely sensitive in comparing information from different data. Small differentiations through the process of normalization can lead to misjudgment of the proper mass spectrum of reference. However, these algorithms cannot be totally rejected because they perform well in many cases. The results are given in Table 7.2.

Table 7.2: Matching results of the six algorithms.

	Name	Pear.	Cos.	Eucl.	Cheb.	Cityblk	Ag.Chem.
1	C_{29} normoretane	1	1	1	1	1	1
2	C_{30} tricyclic terpane	>15	>15	15	>15	15	1
3	C_{30} tricyclic terpane	>15	>15	>15	>15	>15	1
4	C_{28} 17a18a21b(H)-bisnorhopane	7	7	3	14	2	1
5	C_{29} Tm 17a(H)21b(H)-norhopane	1	1	3	9	4	1
6	C_{30} 17a(H)-hopane	4	4	4	4	2	2
7	C_{30} moretane	3	3	3	3	2	1
8	C_{21} tricyclic terpane	1	1	9	1	11	1
9	C_{24} tricyclic terpane	2	2	2	6	1	1
10	C_{25} tricyclic terpane-R	4	6	5	1	4	1
11	C_{25} tricyclic terpane-S	13	13	10	7	4	9
12	C_{24} tetracyclic terpane	>15	>15	>15	>15	>15	1
13	C_{26} tricyclic terpane-R	8	6	10	14	10	1
14	C_{25} tetracyclic terpane	>15	>10	>10	>10	>10	1
15	C_{28} tricyclic terpane-S	2	2	1	3	3	1
16	C_{26} tetracyclic terpane	>15	>15	>15	14	>15	1
17	C_{28} tricyclic terpane-R	2	2	9	9	>15	1
18	C_{27} tetracyclic terpane	>15	>15	>15	>15	>15	1
19	Tm 17a(H)-trisnorhopane	3	2	8	13	5	1
20	oleanane	4	4	3	3	12	1
21	gammacerane	4	4	4	8	>15	3
22	C_{31} 22R 17a(H) hopane	>15	>15	>15	>15	>15	1
23	C_{31} 22S 17a(H) hopane	>15	>15	14	>15	14	1
24	C_{33} 22R 17a(H) hopane	1	1	1	1	8	1
25	C_{33} 22S 17a(H) hopane	1	1	1	1	8	1
26	C_{21}- 5a (H) sterane	1	1	1	1	>15	2
27	C_{21}- 5b (H) sterane	1	1	2	2	3	1
28	C_{27} aa 20S sterane	1	1	1	1	1	1
29	C_{28} aa 20R sterane	4	4	2	2	9	1
30	C_{29} aa 20S sterane	7	7	4	10	3	1
31	C_{27} ba 20R diasterane	1	1	1	1	1	1

Another issue which causes a relative loss of matching accuracy when the first five algorithms are used is the fact that the mass spectra could potentially contain isotopic abundances that could possibly translated erroneously during the process of digitization. There has been an effort to overcome this obstacle by the implementation of the Single Ion Monitoring (SIM) method instead of Total Ion Current Ion (TIC) method, during the extraction of the examined species mass spectral data, which resulted into the improvement of six matching results of the five algorithms. The implementation of the SIM method has removed the secondary abundances and it has maintained only the main peaks. The species that have been evaluated with this method are highlighted in Table 7.2.

7.2. Conclusions

In conclusion, the value of the present study is based upon the fact that it introduces a practical method of combining different mass spectral data from literature sources to an integrated dataset which forms a uniform library. The algorithmic background allows the expansion of the library, thus the number of compounds that potentially can be included is considered infinite. Consequently, the library is a very flexible tool since it can easily incorporate new compounds.

This is a quite important aspect because as the number of the included compounds of the library increases, the identification uncertainties can be reduced. This brings out the usefulness of the developed library especially as regards the compounds with no information on their retention times. These compounds can be identified more easily as the size of the library becomes greater.

The fact that the present work has been entirely carried out in the MatLab environment is also an advantage. The involvement of different kinds of programs has been avoided throughout all the stages of the library development giving the opportunity for exclusive study of the algorithms without shifting from one program to another. The exclusive study can discover code segments that could need future modifications for the improvement of the computing power of the code. Moreover, future work upon this study could include an effort of decoding the matching algorithm of the Agilent Chemstation software.

This page has been intentionally left blank.

Appendix A - Biomarkers Index

Ind.	Name	Formula	M.wt
1	2-Methylpentadecane	$C_{16}H_{34}$	226
2	3-Methylpentadecane	$C_{16}H_{34}$	226
3	2-Methylhexadecane	$C_{17}H_{36}$	240
4	3-Methylhexadecane	$C_{17}H_{36}$	240
5	3-Methylheptadecane	$C_{18}H_{38}$	254
6	3-Methyleicosane	$C_{21}H_{44}$	296
7	7-Methylheptadecane	$C_{18}H_{38}$	254
8	8-Methylheptadecane	$C_{18}H_{38}$	254
9	6- and 7-Methylhexadecane	$C_{17}H_{36}$	240
10	2,6-Dimethylheptane	$C_{9}H_{20}$	128
11	13,16-Dimethyloctacosane	$C_{30}H_{36}$	422
12	2,6,10-Trimethylhexadecane	$C_{19}H_{40}$	268
13	2,6,10,13-Tetramethylpentadecane	$C_{19}H_{40}$	268
14	2,6,10-Trimethylundecane	$C_{14}H_{30}$	298
15	2,6,10-Trimethyldodecane	$C_{15}H_{32}$	212
16	2,6,10-Trimethyltridecane	$C_{16}H_{34}$	226
17	2,6,10-Trimethyltetradecane	$C_{17}H_{36}$	240
18	2,6,10-Trimethylpentadecane	$C_{18}H_{38}$	254
19	2,6,10,14-Tetramethylpentadecane	$C_{19}H_{40}$	268
20	2,6,10,14-Tetramethylhexadecane	$C_{20}H_{42}$	282
21	2,6,10,14-Tetramethylheptadecane	$C_{21}H_{44}$	296
22	2,6,10,14-Tetramethyloctadecane	$C_{22}H_{46}$	310
23	2,6,10,14-Tetramethylnonadecane	$C_{23}H_{48}$	324
24	2,6,10,14,18-Pentamethylnonadecane	$C_{24}H_{50}$	338
25	2,6,10,14,18-Pentamethyleicosane	$C_{25}H_{52}$	352
26	2,6,10,14,18-Pentamethylheneicosane	$C_{26}H_{54}$	366
27	2,6,10,14,18-Pentamethyldocosane	$C_{27}H_{56}$	380
28	2,6,10,14,18-Pentamethyltricosane	$C_{28}H_{58}$	394
29	2,6,10,14,18,22-Hexamethyltricosane	$C_{29}H_{60}$	408
30	2,6,10,14,18,22-Hexamethyltetracosane	$C_{30}H_{62}$	422
31	6,10,14,18,22-Pentamethylpentacosane	$C_{30}H_{62}$	422
32	2,6,10,14,18,22-Hexamethylpentacosane	$C_{31}H_{64}$	436
33	2,6,10,14,18,22-Hexamethylheptacosane	$C_{33}H_{68}$	464
34	2,6,10,14,18,22,26-Heptamethylheptacosane	$C_{34}H_{70}$	478
35	2,6,10,14,18,22,26-Heptamethyloctacosane	$C_{35}H_{72}$	492
36	2,6,10,14,18,22,26-Heptamethylnonacosane	$C_{36}H_{74}$	506
37	2,6,10,14,18,22,26,30-Octamethylhentriacontane	$C_{39}H_{80}$	548
38	Prist-1-ene	$C_{19}H_{38}$	266
39	Prist-2-ene	$C_{19}H_{38}$	266
40	3,7,11-Trimethyltetradecane	$C_{17}H_{36}$	240
41	3,7,11-Trimethylhexadecane	$C_{19}H_{40}$	268

42	3,7,11,15-Ttetramethylheptadecane	$C_{21}H_{44}$	296
43	3,7,11,15-Tetramethylheptadecane	$C_{22}H_{46}$	310
44	3,7,11,15-Tetramethyleicosane	$C_{24}H_{50}$	338
45	3,7,11,15,19-Pentamethylheneicosane	$C_{26}H_{54}$	366
46	3,7,11,15,19-Pentamethyldocosane	$C_{27}H_{56}$	380
47	3,7,11,15,19-Pentamethyltricosane	$C_{28}H_{58}$	394
48	3,7,11,15,19-Pentamethyltetracosane	$C_{29}H_{60}$	408
49	3,7,11,15,19,23-Hexamethylpentacosane	$C_{31}H_{64}$	436
50	3,7,11,15,19,23-Hexamethylhexacosane	$C_{32}H_{66}$	450
51	3,7,11,15,19,23-Heptamethylnonacosane	$C_{36}H_{74}$	506
52	2,6,10,15-Tertramethylheptadecane	$C_{21}H_{44}$	296
53	2,6,10,15,19-Pentamethyleicosane	$C_{25}H_{52}$	352
54	2,6,10,14,19-Pentamethyleicosane	$C_{25}H_{52}$	352
55	Squalene	$C_{30}H_{50}$	410
56	Squalane	$C_{30}H_{62}$	422
57	Lycopane	$C_{40}H_{82}$	562
58	β-Carotene	$C_{40}H_{56}$	536
59	Perhydro-β-carotene	$C_{40}H_{78}$	558
60	2,6,10,14,17,21-Hexamethylhexacosane	$C_{32}H_{66}$	450
61	2,6,10,14,17,21,25-Heptamethylhexacosane	$C_{33}H_{68}$	464
62	2,6,10,14,17,21,25-Heptamethyloctacosane	$C_{35}H_{72}$	492
63	2,6,10,14,17,21,25-Heptamethyltriacontane	$C_{37}H_{76}$	520
64	3,7,11,15,18,22,26-Heptamethylhentriacontane	$C_{38}H_{78}$	534
65	1,6,10,14,17,21,25,29-Heptamethylhentriacontane	$C_{38}H_{78}$	534
66	2,6,10,14,17,21,25,29-Octamethylhentriacontane	$C_{39}H_{80}$	548
67	3,7,11,15,18,22,26,30-Octamethyldotriacontane	$C_{40}H_{82}$	562
68	Botryococcane	$C_{34}H_{70}$	478
69	2,6,10-Trimethyl-7-(3-methylbutyl)-dodecane	$C_{20}H_{42}$	282
70	1-(1,5,8,12,16,20-Hexamethyldocosyl)-3-(4-methyl)-cyclopentane	$C_{40}H_{80}$	560
71	1-(1,5,9,13-Tetramethyltetradecyl)-4-(4,8,12-tri-methyltridecyl)-benzene	$C_{40}H_{74}$	554
72	1,5-Dimethylhexylbenzene	$C_{14}H_{22}$	190
73	1-Methyl-3-(4,8-dimethylnonyl)-benzene	$C_{18}H_{30}$	246
74	1-Methyl-?-ethyl-5-(4,8-dimethylnonyl)-benzene	$C_{20}H_{34}$	274
75	4,8,12-Trimethyltridecylbenzene	$C_{22}H_{38}$	302
76	1-Methyl-3,3-dimethyl-2-(3,7-dimethylundecyl)-cyclohexane	$C_{22}H_{44}$	308
77	Decylcyclohexane	$C_{16}H_{32}$	224
78	Undecylcyclohexane	$C_{17}H_{34}$	238
79	Dodecylcyclohexane	$C_{18}H_{36}$	252
80	Tridecylcyclohexane	$C_{19}H_{38}$	266
81	Docosylbenzene	$C_{28}H_{50}$	386
82	1-Methyl-2-heneicosylbenzene	$C_{28}H_{50}$	386
83	1-Methyl-3-heneicosylbenzene	$C_{28}H_{50}$	386
84	1,3-Dimethyl-2-eicosylbenzene	$C_{28}H_{50}$	386
85	1,3,4-Trimethyl-2-nonadecylbenzene	$C_{28}H_{50}$	386

86	1-Methyl-4-heneicosylbenzene	$C_{28}H_{50}$	386
87	1,2 or 1,4-Dimethyl-3-eicosylbenzene	$C_{28}H_{50}$	386
88	Monoaromatic (Green River Shale)	$C_{13}H_{18}$	174
89	Monoaromatic (Green River Shale)	$C_{15}H_{24}$	204
90	Monoaromatic (Green River Shale)	$C_{16}H_{24}$	216
91	Monoaromatic (Green River Shale)	$C_{14}H_{24}$	228
92	Monoaromatic (Green River Shale)	$C_{17}H_{26}$	230
93	Monoaromatic (Green River Shale)	$C_{17}H_{26}$	230
94	Monoaromatic (Green River Shale)	$C_{18}H_{26}$	242
95	Monoaromatic (Green River Shale)	$C_{18}H_{30}$	246
96	Monoaromatic (Green River Shale)	$C_{18}H_{28}$	244
97	Monoaromatic (Green River Shale)	$C_{18}H_{28}$	244
98	Monoaromatic (Green River Shale)	$C_{18}H_{26}$	242
99	Monoaromatic (Green River Shale)	$C_{19}H_{30}$	258
100	Monoaromatic (Green River Shale)	$C_{18}H_{30}$	246
101	Monoaromatic (Green River Shale)	$C_{18}H_{28}$	244
102	Monoaromatic (Green River Shale)	$C_{20}H_{30}$	270
103	Monoaromatic (Green River Shale)	$C_{20}H_{32}$	272
104	Monoaromatic (Green River Shale)	$C_{20}H_{34}$	274
105	Monoaromatic (Green River Shale)	$C_{21}H_{34}$	286
106	Monoaromatic (Green River Shale)	$C_{21}H_{32}$	284
107	Monoaromatic (Green River Shale)	$C_{22}H_{36}$	300
108	Monoaromatic (Green River Shale)	$C_{22}H_{32}$	296
109	Monoaromatic (Green River Shale)	$C_{21}H_{28}$	280
110	Monoaromatic (Green River Shale)	$C_{25}H_{44}$	344
111	Monoaromatic (Green River Shale)	$C_{26}H_{44}$	356
112	4β(H)-Eudesmane	$C_{15}H_{28}$	208
113	8β(H)-Drimane	$C_{15}H_{28}$	208
114	C_{14}-Bicyclic sesquiterpane	$C_{14}H_{26}$	194
115	C_{15}-Bicyclic sesquiterpane	$C_{15}H_{28}$	208
116	C_{15}-Bicyclic sesquiterpane	$C_{15}H_{28}$	208
117	C_{15}-Bicyclic sesquiterpane	$C_{15}H_{28}$	208
118	C_{15}-Bicyclic sesquiterpane	$C_{15}H_{28}$	208
119	C_{16}-Bicyclic sesquiterpane	$C_{16}H_{30}$	222
120	C_{16}-Bicyclic sesquiterpane	$C_{16}H_{30}$	222
121	Cuparene	$C_{15}H_{22}$	202
122	Cedrane	$C_{15}H_{26}$	206
123	trans-Guainane	$C_{15}H_{28}$	208
124	trans-Gadinane	$C_{15}H_{28}$	208
125	Cadalene	$C_{15}H_{18}$	198
126	Ionene	$C_{13}H_{18}$	174
127	Longifolene	$C_{15}H_{24}$	204
128	Bisnorsimonellite	$C_{17}H_{20}$	224
129	Simonellite	$C_{19}H_{24}$	252

130	Retene	$C_{18}H_{18}$	234
131	1,2,3,4-Tetrahydroretene	$C_{18}H_{22}$	238
132	19-Norabieta-3,8,11,13-tetraene	$C_{19}H_{26}$	254
133	19-Norabieta-4(18)8,11,13-tetraene	$C_{19}H_{26}$	254
134	19-Norabieta-4,8,11,13-tetraene	$C_{19}H_{26}$	254
135	18-Norabieta-8,11,13-triene	$C_{19}H_{28}$	256
136	19-Norabieta-8,11,13-triene	$C_{19}H_{28}$	256
137	Norabietane	$C_{19}H_{34}$	262
138	Dehydroabietane	$C_{20}H_{30}$	270
139	Abietane	$C_{20}H_{36}$	276
140	Fichelite	$C_{19}H_{34}$	262
141	Norpimarane	$C_{19}H_{34}$	262
142	Isopimara-7,15-diene	$C_{20}H_{32}$	272
143	13-Isopimaradiene	$C_{20}H_{32}$	272
144	$\Delta^{8,9}$-Sandaracopimaradiene	$C_{20}H_{32}$	272
145	Sandaracopimarane	$C_{20}H_{36}$	276
146	Sclarene	$C_{20}H_{32}$	272
147	Isophyllocladene (13β-Kaur-15-ene)	$C_{20}H_{32}$	272
148	Phyllocledene (13β-Kaur-16-ene)	$C_{20}H_{32}$	272
149	α-Phyllocladane	$C_{20}H_{34}$	274
150	Kaur-16-ene	$C_{20}H_{32}$	272
151	$\Delta^{5,10}$-Rimuene	$C_{20}H_{32}$	272
152	Dihydrorimuene	$C_{20}H_{34}$	274
153	Tetrahydrorimuene	$C_{20}H_{36}$	276
154	Hibaene	$C_{20}H_{32}$	272
155	Isohibaene	$C_{20}H_{32}$	272
156	Cembrene	$C_{20}H_{32}$	272
157	Rosa-5,15-diene	$C_{20}H_{32}$	272
158	Lauren-1-ene	$C_{20}H_{32}$	272
159	C_{20}-Diterpane	$C_{20}H_{34}$	274
160	C_{20}-Diterpane	$C_{20}H_{34}$	274
161	C_{17}-Diterpane	$C_{17}H_{30}$	234
162	C_{18}-Diterpane	$C_{18}H_{32}$	248
163	C_{18}-Diterpane	$C_{18}H_{32}$	248
164	C_{18}-Diterpane	$C_{18}H_{32}$	248
165	C_{19}-Diterpane	$C_{19}H_{34}$	262
166	C_{20}-Diterpane	$C_{20}H_{36}$	276
167	C_{19}-Tricyclic terpane	$C_{19}H_{34}$	262
168	C_{19}-Tricyclic terpane	$C_{19}H_{34}$	262
169	C_{19}-Tricyclic terpane	$C_{19}H_{34}$	262
170	C_{20}-Tricyclic terpane	$C_{20}H_{36}$	276
171	Isocopalane	$C_{20}H_{36}$	276
172	Isocopalene	$C_{20}H_{34}$	274
173	C_{21}-Tricyclic terpane	$C_{21}H_{38}$	290

174	C_{21}-Tricyclic terpane	$C_{21}H_{38}$	290
175	C_{22}-Tricyclic terpane	$C_{22}H_{40}$	304
176	C_{23}-Tricyclic terpane	$C_{23}H_{42}$	318
177	C_{23}-Tricyclic terpane	$C_{23}H_{42}$	318
178	C_{24}-Tricyclic terpane	$C_{24}H_{44}$	332
179	C_{25}-Tricyclic terpane	$C_{25}H_{46}$	346
180	C_{26}-Tricyclic terpane	$C_{26}H_{48}$	360
181	C_{28}-Tricyclic terpane	$C_{28}H_{50}$	388
182	C_{29}-Tricyclic terpane	$C_{29}H_{52}$	402
183	C_{30}-Tricyclic terpane	$C_{30}H_{54}$	416
184	C_{44}-Tricyclic terpane	$C_{44}H_{84}$	612
185	C_{25}-Tricyclic terpane	$C_{25}H_{46}$	346
186	17β(H)-22,29,30-Trisnorhopane	$C_{27}H_{46}$	370
187	17β(H),21β(H)-30-Norhopane	$C_{29}H_{50}$	398
188	17β(H),21β(H)-Hopane	$C_{30}H_{52}$	412
189	17β(H),21β(H)-Homohopane	$C_{31}H_{54}$	426
190	17α(H),22,29,30-Trisnorhopane	$C_{27}H_{46}$	370
191	17α(H),18α(H),21β(H)-28,30-Bisnorhopane	$C_{28}H_{48}$	384
192	17β(H),21β(H)-30-Norhopane	$C_{29}H_{50}$	398
193	17α(H),21β(H)-Hopane	$C_{30}H_{52}$	412
194	17α(H),21β(H)-Homohopane	$C_{31}H_{54}$	426
195	17α(H),21β(H)-Trishomohopane	$C_{33}H_{58}$	454
196	17α(H),21β(H)-C_{40}-Hopane	$C_{40}H_{72}$	552
197	17β(H),18α(H),21α(H)-28,30-Bisnormoretane	$C_{28}H_{48}$	384
198	17β(H),21α(H)-30-Normoretane	$C_{29}H_{50}$	398
199	17β(H),21α(H)-Moretane	$C_{30}H_{52}$	412
200	17β(H),21α(H)-Homomoretane	$C_{31}H_{54}$	426
201	17α(H),21α(H)-30-Norhopane	$C_{29}H_{50}$	398
202	17α(H),21α(H)-Hopane	$C_{30}H_{52}$	412
203	17β(H),18α(H),21β(H)-25,28,30-Trisnorhopane	$C_{27}H_{46}$	370
204	17α(H),21β(H)-25-Norhopane	$C_{29}H_{50}$	398
205	17β(H),18α(H),21α(H)-25,28,30-Trisnormoretane	$C_{27}H_{46}$	370
206	17,21-Secohopane	$C_{24}H_{42}$	330
207	17,21-Secohopane	$C_{25}H_{44}$	344
208	17,21-Secohopane	$C_{26}H_{46}$	358
209	17,21-Secohopane	$C_{27}H_{48}$	372
210	8,14-Secohopane	$C_{27}H_{48}$	372
211	8,14-Secohopane	$C_{29}H_{52}$	400
212	8,14-Secohopane	$C_{30}H_{54}$	414
213	8,14-Secohopane	$C_{30}H_{54}$	414
214	8,14-Secohopane	$C_{30}H_{54}$	414
215	8,14-Secohopane	$C_{27}H_{48}$	372
216	22,29,30-Trisnorhop-13(18)-ene	$C_{27}H_{44}$	368
217	22,29,30-Trisnorhop-17(21)-ene	$C_{27}H_{44}$	368

218	30-Norneohop-13(18)-ene	$C_{29}H_{48}$	396
219	30-Norhop-17(21)-ene	$C_{29}H_{48}$	396
220	Hop-22(29)-ene	$C_{30}H_{50}$	410
221	Hop-17(21)-ene	$C_{30}H_{50}$	410
222	Neohop-13(18)-ene	$C_{30}H_{50}$	410
223	Hop-21-ene	$C_{30}H_{50}$	410
224	Homohop-17(21)-ene	$C_{31}H_{52}$	424
225	Homohop-30-ene	$C_{31}H_{52}$	424
226	D-Ring monoaromatic hopane	$C_{27}H_{40}$	364
227	C,D-Ring diaromatic hopane	$C_{26}H_{34}$	346
228	B,C,D-Ring-triaromatic hopane	$C_{25}H_{28}$	328
229	A,B,C,D-Ring-tetraaromatic hopane	$C_{24}H_{22}$	310
230	Lupane	$C_{30}H_{52}$	412
231	17β(H)-23,28-Bisnorlupane	$C_{28}H_{48}$	384
232	17α(H)-23,28-Bisnorlupane	$C_{28}H_{48}$	384
233	17α(H)-28-Norlupane	$C_{29}H_{50}$	398
234	17β(H)-28-Norlupane	$C_{29}H_{50}$	398
235	De-A-lupane	$C_{29}H_{42}$	330
236	18α(H)-Oleane	$C_{30}H_{52}$	412
237	Olean-18-ene	$C_{30}H_{50}$	410
238	Olean-12-ene	$C_{30}H_{50}$	410
239	Olean-13(18)-ene	$C_{30}H_{50}$	410
240	Friedelane	$C_{30}H_{52}$	412
241	Friedel-18-ene	$C_{30}H_{50}$	410
242	Gammacerane	$C_{30}H_{52}$	412
243	Filicane	$C_{30}H_{52}$	412
244	Glutane	$C_{30}H_{52}$	412
245	Cycloartane	$C_{30}H_{52}$	412
246	Cycloeuclane	$C_{30}H_{52}$	412
247	Cyclolaudane	$C_{31}H_{54}$	426
248	Serratane I (14β(H))	$C_{30}H_{52}$	412
249	Serratane II (14α(H))	$C_{30}H_{52}$	412
250	Aborane	$C_{30}H_{52}$	412
251	Adianane	$C_{30}H_{52}$	412
252	Urs-12-ene	$C_{30}H_{50}$	410
253	Fernane	$C_{30}H_{52}$	412
254	Fernene (?) (Fern-8-ene)	$C_{30}H_{50}$	410
255	Fernene (?) (Fern-9(11)-ene)	$C_{30}H_{50}$	410
256	Unknown (Fern-7-ene)	$C_{30}H_{50}$	410
257	Shionane	$C_{30}H_{54}$	414
258	Onocerane I (8β(H),14α(H))	$C_{30}H_{54}$	414
259	Onocerane II (8β(H),14β(H))	$C_{30}H_{54}$	414
260	Onocerane III (8α(H),14β(H))	$C_{30}H_{54}$	414
261	Resin compound T	$C_{30}H_{52}$	412

262	Resin compound W	$C_{30}H_{52}$	412
263	Resin compound R	$C_{30}H_{52}$	412
264	Dammarane	$C_{30}H_{54}$	412
265	Dammara-18(28),21-diene	$C_{30}H_{50}$	410
266	Dammara-13(17),24-diene	$C_{30}H_{50}$	410
267	Eupha-7,24-diene	$C_{30}H_{50}$	410
268	Hydrocarbon analogue of Colysanoxide	$C_{30}H_{50}$	410
269	γ-Polypodatetraene	$C_{30}H_{50}$	410
270	α-Polypodatetraene	$C_{30}H_{50}$	410
271	Bacchara-12,21-diene	$C_{30}H_{50}$	410
272	Lemmaphylla-7,21-diene	$C_{30}H_{50}$	410
273	Shiona-3,21-diene	$C_{30}H_{50}$	410
274	Unknown triterpenoid (Tarax-14-ene)	$C_{30}H_{50}$	410
275	Aromatized triterpane	$C_{26}H_{30}$	342
276	Aromatized triterpane	$C_{26}H_{30}$	342
277	Aromatized triterpane	$C_{25}H_{26}$	326
278	Aromatized triterpane	$C_{24}H_{22}$	310
279	Aromatized triterpane	$C_{24}H_{22}$	310
280	Aromatized triterpane	$C_{25}H_{24}$	324
281	Aromatized triterpane	$C_{21}H_{22}$	274
282	Aromatized triterpane	$C_{22}H_{28}$	292
283	Compound C	$C_{23}H_{20}$	296
284	Compound M	$C_{26}H_{26}$	296
285	Compound G	$C_{25}H_{22}$	322
286	Compound N	$C_{26}H_{26}$	338
287	Compound O	$C_{25}H_{24}$	324
288	Compound L	$C_{25}H_{24}$	324
289	Compound A	$C_{19}H_{14}$	242
290	Compound C'	$C_{23}H_{24}$	300
291	Compound D'	$C_{26}H_{28}$	340
292	Compound J'	$C_{27}H_{32}$	356
293	Compound H'	$C_{26}H_{30}$	342
294	Compound E'	$C_{26}H_{32}$	344
295	Compound A'	$C_{17}H_{14}$	218
296	Compound F'	$C_{26}H_{30}$	342
297	Compound I'	$C_{27}H_{32}$	356
298	Tetramethyloctahydrochrysene	$C_{22}H_{28}$	292
299	Dimethyloctahydrochrysene	$C_{20}H_{24}$	264
300	3,3,7-Trimethyl-1,2,3,4-tetrahydrochrysene	$C_{21}H_{22}$	274
301	1,2,3,4-Tetrahydro-1,2,9-trimethylpicene	$C_{25}H_{24}$	324
302	1,2,3,4-Tetrahydro-1,2-dimethylpicene	$C_{24}H_{22}$	310
303	5α(H)-Androstane	$C_{19}H_{32}$	260
304	5β(H)-Androstane	$C_{19}H_{32}$	260
305	D-Homo-5α(H)-Androstane	$C_{20}H_{34}$	274

306	5α(H)-Pregnane	$C_{21}H_{36}$	288
307	5β(H)-Pregnane	$C_{21}H_{36}$	288
308	C_{23}-sterane	$C_{23}H_{40}$	316
309	Norcholest-2-ene	$C_{26}H_{44}$	356
310	Norcholest-4-ene	$C_{26}H_{44}$	356
311	21-Nor-5α(H)-cholest-24-ene	$C_{26}H_{44}$	356
312	Cholestatriene	$C_{27}H_{42}$	366
313	Cholestadiene	$C_{27}H_{44}$	368
314	5α(H)-Cholest-1-ene	$C_{27}H_{46}$	370
315	5α(H)-Cholest-2-ene	$C_{27}H_{46}$	370
316	5α(H)-Cholest-3-ene	$C_{27}H_{46}$	370
317	Cholest-4-ene	$C_{27}H_{46}$	370
318	Cholest-5-ene	$C_{27}H_{46}$	370
319	5α(H)-Cholest-6-ene	$C_{27}H_{46}$	370
320	5α(H)-Cholest-7-ene	$C_{27}H_{46}$	370
321	5α(H)-Cholest-8(14)-ene	$C_{27}H_{46}$	370
322	5α(H)-Cholest-22-ene	$C_{27}H_{46}$	370
323	5β(H)-Cholest-23-ene	$C_{27}H_{46}$	370
324	5β(H)-Cholest-24-ene	$C_{27}H_{46}$	370
325	3α(H),5α(H)-Cyclocholestane	$C_{27}H_{46}$	370
326	5α(H)-Cholestane	$C_{27}H_{48}$	372
327	5β(H)-Cholestane	$C_{27}H_{48}$	372
328	5α(H),8α(H),14β(H)-Cholestane	$C_{27}H_{48}$	372
329	5α(H),14β(H)-Cholestane	$C_{27}H_{48}$	372
330	24-Methylcholestadiene	$C_{27}H_{46}$	382
331	24-Methyl-5α(H)-cholest-2-ene	$C_{27}H_{48}$	384
332	24R-Methyl-5α(H)-cholest-3-ene	$C_{27}H_{48}$	384
333	24R-Methylcholest-5-ene	$C_{27}H_{48}$	384
334	24-Methyl-5β(H)-cholest-24-ene	$C_{27}H_{48}$	384
335	24-Methyl-5α(H)-cholestane	$C_{28}H_{50}$	386
336	24-Methyl-5β(H)-cholestane	$C_{28}H_{50}$	386
337	24R-Methyl-5α(H),14β(H),17β(H)-cholestane	$C_{28}H_{50}$	386
338	4α(H)-Methylcholestane	$C_{28}H_{50}$	386
339	14α-Methylcholestane	$C_{28}H_{50}$	386
340	24-Ethylcholestadiene	$C_{29}H_{48}$	396
341	24-Ethyl-5α(H)-cholest-2-ene	$C_{29}H_{50}$	398
342	24R-Ethylcholest-4-ene	$C_{29}H_{50}$	398
343	24-Ethyl-5α(H)-cholestane	$C_{29}H_{52}$	400
344	24-Ethyl-5β(H)-cholestane	$C_{29}H_{52}$	400
345	Lanostane	$C_{30}H_{54}$	414
346	20R-Diacholestane	$C_{27}H_{46}$	370
347	20S-Diacholestane	$C_{27}H_{46}$	370
348	20R-13α(H),17β(H)-Diacholestane	$C_{27}H_{48}$	372
349	20R-13α(H),17α(H)-Diacholestane	$C_{27}H_{48}$	372

350	4-Methyl-13α(H),17β(H)-diacholestane (20R or S)	$C_{28}H_{50}$	386
351	4-Methyl-13β(H),17α(H)-diacholestane (20R or S)	$C_{28}H_{50}$	386
352	24-Ethyldiacholest-13(17)-ene	$C_{29}H_{50}$	398
353	19-Norholestane	$C_{26}H_{46}$	358
354	5β(H)-A-Norcholestane	$C_{26}H_{46}$	358
355	5α(H)-A-Norcholestane	$C_{26}H_{46}$	358
356	1-Methyl-19-norholest-1,3,5(10)-triene	$C_{27}H_{42}$	366
357	4-Methyl-19-norholest-1,3,5(10)-triene	$C_{27}H_{42}$	366
358	4-Methyl-24-ethyl-19-norholest-1,3,5(10)-triene	$C_{29}H_{46}$	396
359	C_{27}-Monoaromatic (C-ring) sterane	$C_{27}H_{42}$	366
360	C_{27}-Monoaromatic (C-ring) sterane	$C_{27}H_{42}$	366
361	C_{27}-Monoaromatic (C-ring) sterane	$C_{27}H_{42}$	366
362	C_{27}-Monoaromatic (C-ring) sterane	$C_{27}H_{42}$	366
363	C_{27}-Monoaromatic (C-ring) sterane	$C_{27}H_{42}$	366
364	C_{20}-Triaromatic sterane	$C_{20}H_{20}$	260
365	C_{26}-Triaromatic sterane	$C_{26}H_{32}$	344
366	C_{27}-Triaromatic sterane	$C_{27}H_{34}$	358
367	1-Methyl-C_{27}-triaromatic sterane	$C_{27}H_{34}$	358
368	4-Methyl-C_{27}-triaromatic sterane	$C_{27}H_{34}$	358
369	4-Methyl-C_{29}-triaromatic sterane	$C_{29}H_{38}$	386
370	C_{25}-Monoaromatic de-A-sterane	$C_{25}H_{40}$	348
371	14α(H)-1(10-6)-abeo-cholesta-5,7,9(10)-triene	$C_{27}H_{42}$	366
372	14β(H)-1(10-6)-abeo-cholesta-5,7,9(10)-triene	$C_{27}H_{42}$	366
373	24R-14β(H)-1(10-6)-abeo-cholesta-5,7,9(10)-triene	$C_{29}H_{46}$	386

This page has been intentionally left blank.

Appendix B - Graphical User Interface

```
function varargout = bio_x_app(varargin)

 % Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @bio_x_app_OpeningFcn, ...
                   'gui_OutputFcn',  @bio_x_app_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before bio_x_app is made visible.
function bio_x_app_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);


% --- Outputs from this function are returned to the command line.
function varargout = bio_x_app_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;


% --- Executes on button press in Import.
function Import_Callback(hObject, eventdata, handles)

if handles.segs == 1;
     [filename, filepath1]=uigetfile({'*.png'},'Select  File');      %
read the file on condition on the number of segments
     handles.I = imread(filename);

 else if handles.segs == 2;
        [filename1, filepath1]=uigetfile({'*.png'},'Select  File');
        handles.I1 = imread(filename1);

        [filename2, filepath2]=uigetfile({'*.png'},'Select  File');
        handles.I2 = imread(filename2);

     end

end

guidata(hObject,handles)
```

```
% --- Executes on button press in Export.
function Export_Callback(hObject, eventdata, handles)

if handles.segs == 1;
    I = handles.I;
    FIRST = handles.fp;
    LAST = handles.mi;
    key = handles.key;
    coords = image_to_coords(I,FIRST,LAST,key);

else if handles.segs == 2;
        I1 = handles.I1;
        FIRST1 = handles.fp;
        LAST1 = handles.seg1_last_peak;
        key1= 1;
        coords1 = image_to_coords(I1,FIRST1,LAST1,key1);

        I2 = handles.I2;
        FIRST2 = handles.seg2_first_peak;
        LAST2 = handles.mi;
        key2= handles.key;
        coords2 = image_to_coords(I2,FIRST2,LAST2,key2);

        coords = [coords1;coords2];

    end

end

dlmwrite('index.txt', coords);

load('Title.mat','TITLE');

fileID = fopen('index.txt','w');
formatSpec = '%s \n';
[nrows,ncols] = size(TITLE);

for row = 1:nrows
    fprintf(fileID,formatSpec,TITLE{row,:});
end

norm_x = coords(:,1);
norm_y = ((coords(:,2))*999000)/max(coords(:,2));     % normalize to
999000
norm_coords = [norm_x norm_y];

fprintf(fileID,'%6.0f%12.0f\n', norm_coords');

fclose(fileID);

newName = [int2str(handles.index_number),'.txt'];
movefile('index.txt',newName)
```

```
% --- Executes on button press in Reset.
function Reset_Callback(hObject, eventdata, handles)
set(findobj(0,'style','edit'),'string','')
set(handles.text8,'Enable','on')
set(handles.text9,'Enable','on')
set(handles.text12,'Enable','on')
set(handles.segment_1_last_peak,'Enable','on')
set(handles.segment_2_first_peak,'Enable','on')


function index_number_Callback(hObject, eventdata, handles)
index_number = str2double(get(hObject, 'String'));
if isnan(index_number)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
% Save the component_name
handles.index_number = index_number;
guidata(hObject,handles)


% --- Executes during object creation, after setting all properties.
function index_number_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function Segments_Callback(hObject, eventdata, handles)
segs = str2double(get(hObject, 'String'));
if isnan(segs)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end

if segs == 1
    set(handles.text8,'Enable','off')
    set(handles.text9,'Enable','off')
    set(handles.text12,'Enable','off')
    set(handles.segment_1_last_peak,'Enable','off')
    set(handles.segment_2_first_peak,'Enable','off')

end

% Save the number of segments
handles.segs = segs;
guidata(hObject,handles)


% --- Executes during object creation, after setting all properties.
function Segments_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function first_peak_Callback(hObject, eventdata, handles)
fp = str2double(get(hObject, 'String'));
if isnan(fp)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
% Save the first peak
handles.fp = fp;
guidata(hObject,handles)


% --- Executes during object creation, after setting all properties.
function first_peak_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function molecular_ion_Callback(hObject, eventdata, handles)
mi = str2double(get(hObject, 'String'));
if isnan(mi)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
% Save the molecular ion
handles.mi = mi;
guidata(hObject,handles)


% --- Executes during object creation, after setting all properties.
function molecular_ion_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function molecular_ion_position_Callback(hObject, eventdata, handles)
key = str2double(get(hObject, 'String'));
if isnan(key)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
% Save the position of the molecular ion
handles.key = key;
guidata(hObject,handles)


% --- Executes during object creation, after setting all properties.
function molecular_ion_position_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function segment_1_last_peak_Callback(hObject, eventdata, handles)
seg1_last_peak = str2double(get(hObject, 'String'));
```

```
if isnan(seg1_last_peak)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
% Save the position of the segment 1 last peak
handles.seg1_last_peak = seg1_last_peak;
guidata(hObject,handles)


% --- Executes during object creation, after setting all properties.
function segment_1_last_peak_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


function segment_2_first_peak_Callback(hObject, eventdata, handles)
seg2_first_peak = str2double(get(hObject, 'String'));
if isnan(seg2_first_peak)
    set(hObject, 'String', 0);
    errordlg('Input must be a number','Error');
end
% Save the position of the segment 2 first peak
handles.seg2_first_peak = seg2_first_peak;
guidata(hObject,handles)

% --- Executes during object creation, after setting all properties.
function segment_2_first_peak_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

This page has been intentionally left blank.

References

[1] B. P. Tissot and D. H. Welte, *Petroleum formation and occurrence*. Berlin: Springer, 1984.

[2] A. S. McKenzie and T. M. Quigley, "Principles of geochemical prospect appraisal," *American Association of Petroleum Geologists Bulletin,* pp. 72, 300-415, 1998.

[3] N. Pasadakis, *Petroleum Geochemistry*. Thessaloniki: Tziolas, 2015.

[4] K. McCarthy, K. Rojas, M. Niemann, D. Palmowski, K. Peters, and A. Stankiewicz, "Basic petroleum geochemistry for source rock evaluation," *Oilfield Review,* vol. 23, pp. 41-42, 2011.

[5] D. W. Waples and T. Machihara, *Biomarkers for geologists. A practical guide to the application of steranes and triterpanes in petroleum geology*. Tulsa: American Association of Petroleum Geologists, 1991.

[6] K. E. Peters, C. C. Walters, and J. W. Moldowan, *The Biomarker Guide (2nd ed)*. Cambridge, UK: Cambridge University Press, 2005.

[7] R. A. Bourbonniere and P. A. Meyers, "Sedimentary Geolipid Records of Historical Changes in the Watersheds and Productivities of Lakes Ontario and Erie," *Limnology and Oceanography,* vol. 41, pp. 352-3599, 1996.

[8] J. M. Moldowan, W. K. Seifert, and E. J. Gallegos, "Relationship Between Petroleum Composition and Depositional Environment of Petroleum Source Rocks," *American Association of Petroleum Geologists Bulletin,* vol. 69, pp. 1255-1268, 1985.

[9] S. Derenne, C. Largeau, E. Casadevall, and J. Connan, "Comparison of Torbanites of Various Origins and Evolutionary Stages. Bacterial Contribution to their Formation. Cause of Lack of Botryococcane in Bitumens," *Organic Geochemistry,* vol. 12, pp. 43-59, 1988.

[10] E. E. Bray and E. D. Evans, "Distribution of n-paraffins as a clue to recognition of source beds " *Geochimica et Cosmochimica Acta,* vol. 22, pp. 2-15, 1961.

[11] R. S. Scalan and J. E. Smith, "An improved measure of odd-even predominance in the normal alkanes of sediment extracts and petroleum," *Geochimica et Cosmochimica Acta,* vol. 34, pp. 611-620, 1970.

[12] W. R. Nes and M. L. McKean, *Biochemistry of Steroids and Other Isopentenoids*. Baltimore: University Park Press, 1977.

[13] J. D. Connolly and R. A. Hill, *Dictionary of Terpenoids*. London: Chapman and Hall, 1991.

[14] Z. Wang, S. A. Stout, and M. Fingas, "Forensic Fingerprinting of Biomarkers for Oil Spill Characterization and Source Identification," *Environmental Forensics,* vol. 7, pp. 105-146, 2006.

[15] T. G. Powell, "Pristane/Phytane ratio as environmental indicator," *Nature,* vol. 333, p. 604, 1998.

[16] J. T. Watson and O. D. Sparkman, *Introduction to Mass Spectrometry: Instrumentation, Applications and Strategies for Data Interpretation*. Chichester, U.K.: Wiley, 2007.

[17] R. L. Grob, *Modern Practice of Gas Chromatography*, Third ed., 1995.

[18] I. A. Fowlis, *Gas Chromatography*, Second ed., 1995.

[19] O. D. Sparkman, E. P. Zelda, and G. K. Fulton, *Gas Chromatography and Mass Spectrometry : A Practical Guide*, 2nd ed. U.K: Academic Press, 2011.

[20] E. de Hoffmann and V. Stroobant, *Mass Spectrometry Principles and Applications*, Third ed. England: Wiley, 2007.

[21] K. L. Busch, "Electron ionization, up close and personal," *Spectroscopy (Eugene, OR),* vol. 10, pp. 39-42, 1995.

[22] C. Steel and M. Henchman, "Understanding the Quadrupole Mass Filter through Computer Simulation," *Journal of Chemical Education,* vol. 75, pp. 1049-1054, 1998.

[23] K. J. Welham, *Mass Separation*. Hull, UK, 2005.

[24] H. J. Hübschmann, *Handbook of GC-MS, Fundamentals and Applications*: Wiley-VCH, 2015.

[25] *NIST Chemistry WebBook*. Available: http://webbook.nist.gov/chemistry

[26] R. P. Philp, *Fossil fuel biomarkers applications and spectra*. New York: Elsevier Science, 1985.

[27] MathWorks. *MatLab function "imread"*. Available: https://www.mathworks.com/help/matlab/ref/imread.html

[28] MathWorks. *MatLab function "rgb2gray"*. Available: https://www.mathworks.com/help/matlab/ref/rgb2gray.html

[29] MathWorks. *MatLab function "sum"*. Available: https://www.mathworks.com/help/matlab/ref/sum.html

[30] MathWorks. *Signal Processing Toolbox - MatLab function "findpeaks"*. Available: https://www.mathworks.com/help/signal/ref/findpeaks.html

[31] MathWorks. *MatLab function "stem"*. Available: https://www.mathworks.com/help/matlab/ref/stem.html

[32] K. Pearson, "Notes on regression and inheritance in the case of two parents," in *Proceedings of the Royal Society of London*, 1895, pp. 240-242.

[33] M. Deza and E. Deza, *Encyclopedia of Distances*: Spinger, 2009.

[34] J. M. Abello and P. M. Pardalos, *Handbook of Massive Data Sets*: Springer, 2002.