Development of data processing pipeline for Multidimensional Liquid Chromatography coupled to Mass Spectrometry

By

Valentin Gabarov

Graduation report

Submitted to

Hanze University of Applied Science Groningen

in partial fulfillment of the requirements

for the degree of

Fulltime Honours Master Sensor System Engineering

2016

Abstract

Development of data processing pipeline for Multidimensional Liquid Chromatography coupled to Mass Spectrometry

by

Valentin Gabarov

Biomarker discovery is essential to the pharmaceutical industry for the diagnosis, monitoring and potentially prevention of disease. The Threshold Avoidance Proteomics Pipeline (TAPP) is a tool currently being developed for the discovery of low abundance protein biomarkers, using liquid chromatography coupled with mass spectrometry (1D-LC-MS/MS). The complexity of proteomic samples exhausts the separation power of any single separation technique, which prevents it for distinguishing peptides with similar chemical properties from each other. The use of two liquid chromatography separation phases orthogonal to each other (2D-LC-MS/MS) improves the separation power of the pipeline, but results in data split into fractions. This introduced the need for additional data processing, compared to the 1D-LC-MS/MS algorithm currently used by TAPP. This project aimed to develop a new algorithm using already existing components of TAPP that aligns the fractions in each sample in order to correct for shifts in retention time and links data between the fractions based on proximity. The same fractions of each sample are processed using the one dimensional workflow and combined with the new fraction link information into data format that mimics its predecessor. The additional dimension allows to analyze proteome with larger dynamic concentration range, identifying potential lower abundance biomarkers.

Declaration

I hereby certify that this report constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given when I have used the language, ideas, expressions or writing of another.

I declare that the report describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Valentin Gabarov

Acknowledgements

First I would like to thank Peter Horvatovich for the opportunity to work on this fascinating and innovative project as part of the Analytical Biochemistry research group in the Department of Pharmacy at the University of Groningen. I'm also grateful for the extraordinary amount of assistance provided by him in the form of discussion and feedback during the entirety of the project.

Additionally, I would like to thank Victor Guryev for assistance as a secondary supervisor within the research group, as well as to all members of the group for the discussions during the weekly meetings.

Last but not least, I would like Esther Vertelman for being both my graduation coordinator and supervisors, and providing excellent feedback during all staged of my project.

1 CONTENTS

2 LISTS

2.1 ABBREVIATIONS

MS	Mass Spectrometry
LC	Liquid Chromatography
HPLC	High Performance Liquid Chromatography
TAPP	Threshold Avoidance Proteomics Pipeline
SCX	Strong-cation-exchange Chromatography
MS/MS	Tandem Mass Spectrometry
LC-MS	Liquid Chromatography coupled to Mass Spectrometry (Assumed One Dimensional)
2D-LC-MS	Two-Dimensional Liquid Chromatography coupled to Mass Spectrometry
LC^n-MS	Multi-Dimensional Liquid Chromatography coupled to Mass Spectrometry
RT(rt)	Retention Time
Mz	Mass over Charge
COPD	Chronic Obstructive Pulmonary Disease
FSM	Finite State Machine

2.2 LIST OF DEFINITIONS

Grid	C++ program which converts mzXML Mass Spectrometer Data onto a uniform grid
Centroid	C++ program which is used to extract peaks, without the use of thresholding
Warp2d	C++ program used to align two sets of LC-MS peaks, with respect to the retention time
MetaMatch	C++ program which performs a clustering algorithm between peak sets of different samples
Fraction	When using more than one LC, the sample is divided into fractions using another type of LC
2D-LC-MS/MS Algorithm	This refers to all steps required to achieve data processing from mzXML files to a single list of Meta Peaks using all fractions in all samples provided
TAPP 2D	Module for identifying peaks caused by the same chemical compound spread across multiple fractions
Workflow (1D or 2D)	Automated execution of all steps required in order to processes either 1D-LC-MS or 2D-LC-MS data, using only a couple of input files and a single command from a user
mzXML	Open data format for storage and exchange of mass spectrometer data
.pks	Data format used by TAPP to store peaks extracted by Centroid
.wpks	Data format used by TAPP to store peaks warped to a .pks file using Warp2d
.mpks	Data format used by TAPP to store Meta peaks generated by MetaMatch
.pid	Data format used by TAPP to store the identification numbers of the peaks used by MetaMatch to form Meta peaks
JSON	Lightweight data-interchange format, originating from Java, but used by multiple programing languages
Connection (Peaks)	Two or more peaks that have been identified to have originated from the same compound by ether MetaMatch, TAPP 2D or MS/MS identification.

2.3 List of Figures

2.4 List of Tables

3 RATIONAL

Biological markers (biomarkers) have been defined as “cellular, biochemical or molecular alterations that are measurable in biological media such as human tissues, cells, or fluids”. Disease biomarkers are used as an indicator of a biological factor that represents either a subclinical manifestation, stage of the disorder, or a surrogate manifestation of a disease. They can represent direct steps in the causal pathway of a disease or related to the molecular cause of the disease. Biomarkers are used in the diagnosis and management of many disease such as cardiovascular disease, infections, immunological and genetic disorders, and cancer [1].

There are two main elements of Biomarker research: the discovery and the validation phases. By screening a wide range of compounds using profiling methodologies, the discovery phase aims to find potential biomarker candidates. Although these methodologies can applied to the fields of genomics and metabolomics, within the scope of this project the focus is on proteomics [2]. The discovery phase consists of analysis of a limited number of samples from well-classified patients and controls. The result is a large number of quantified and often also identified molecules relative to a limited number of analyzes. The goal is to identify compounds that are statistically significant between the pre-classified samples, and which have the potential to be generalized to new set of samples. Due to the low number of samples the obtained statistical models need to be validated, in order to exclude observed differences caused by chance or by any other reason for false positive identification [3].

3.1 LIQUID CHROMATOGRAPHY

Combining Liquid Chromatography (LC) with Mass Spectrometry (MS) is a power separation technology used for biomarker discovery. The Frank Suits at IBM Watson center in collaboration with Analytical Biochemistry research group at the Groningen Research Institute of Pharmacy is developing the Threshold Avoidance Proteomics Pipeline (TAPP), which aims to maximize the dynamic range of quantification in single stage (non-fragmented) LC-MS data [4].

LC is an analytical technique used to separate protein derived peptides based on their ability to propagate through a solvent in a stationary phase included in a column and subsequently enter the separated peptides into the mass spectrometer, which quantifies and fragments the eluting peptides. In LC the water/organic solvent composition is changing in time and allow different compounds to displace at different speeds based on their size, shape and physicochemical properties of their chemical surface. The time that is required for a compound to exit the LC column is called retention time(rt) [5].

3.2 THE NEED OF AN ADDITIONAL LC

Biological samples are highly complex in term of protein composition. It is estimated that a living organism contains 100 thousand to millions of protein forms. Proteins are further cleaved to smaller peptides (1 protein results in 10-80 peptides depending from the size of the protein and the cleavage rule of the protease enzyme) resulting in highly complex peptide mixtures, which need to be analysed comprehensively by an analytical system [2].

Proteomics analysis requires as much separation of samples as possible and within LC that separation capability is known as peak capacity. Proteomic samples are so complex that they exhaust the

separation power of any single dimension separation system presently available. By using different physio-chemical properties of proteins it is possible to create two "orthogonal" separation dimensions, which is known as Multi-Dimensional Liquid Chromatography (LC^n) [2].

A widely used approach in proteomics combines strong-cation-exchange chromatography (SCX) and reversed-phase LC. In the first dimension the stationary phase binds tightly with any strongly basic analytes and in the latter it has a stronger affinity for hydrophobic compounds [6]. Two Dimensional Liquid Chromatography coupled with Mass Spectrometry (2D-LC-MS) combines these two separation techniques and provide suitable orthogonality and large peak capacity that enable efficient analysis of complex proteomics samples [7].

3.3 Mass Spectrometry

Mass spectrometry (MS) is a technique that separates a compounds based on its mass to charge ration (m/z). Mass spectrometers shoot a stream of ions with the same velocity through a magnetic and electric fields. The fields deflect each individual ion based on their mass to charge ratio, which results in ions hitting different sections of a detector or reaching the detector at different time. The first step in a mass spectrometer is to ionize the compounds providing a charge. Depending on the mass spectrometer used it is possible to analyze samples in solid, liquid and gas phases, each requiring a different ionization process and allow to fragment them with different types result of fragmentation approaches [8].

The output of a mass spectrometer is a large number of peaks across a m/z spectrum, with each peak representing the amount of ions. With those peaks it is possible to construct a quantitative list of compounds (peptides, metabolites) that were present in the sample, some of which may be potential biomarkers for a specific disease [3]. Figure 1 shows single-stage LC-MS image of ion distribution obtained from trypsin digested human serum sample depleted from the 6 most abundant proteins.

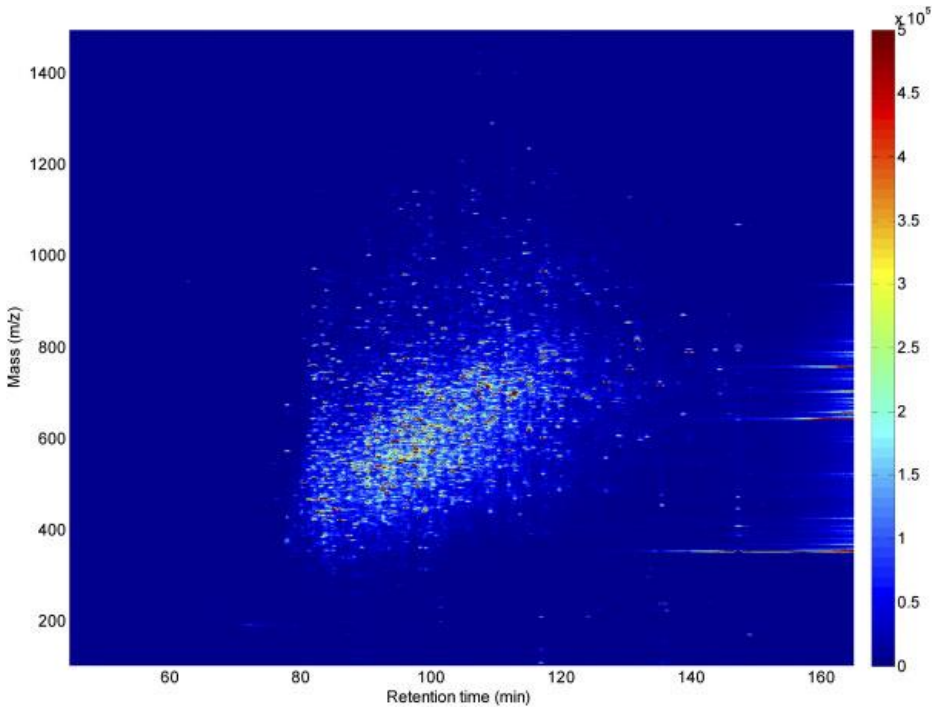


Figure 1 Example of a trypsin-digested human blood serum after depleting it of the 6 most abundant proteins. [7]

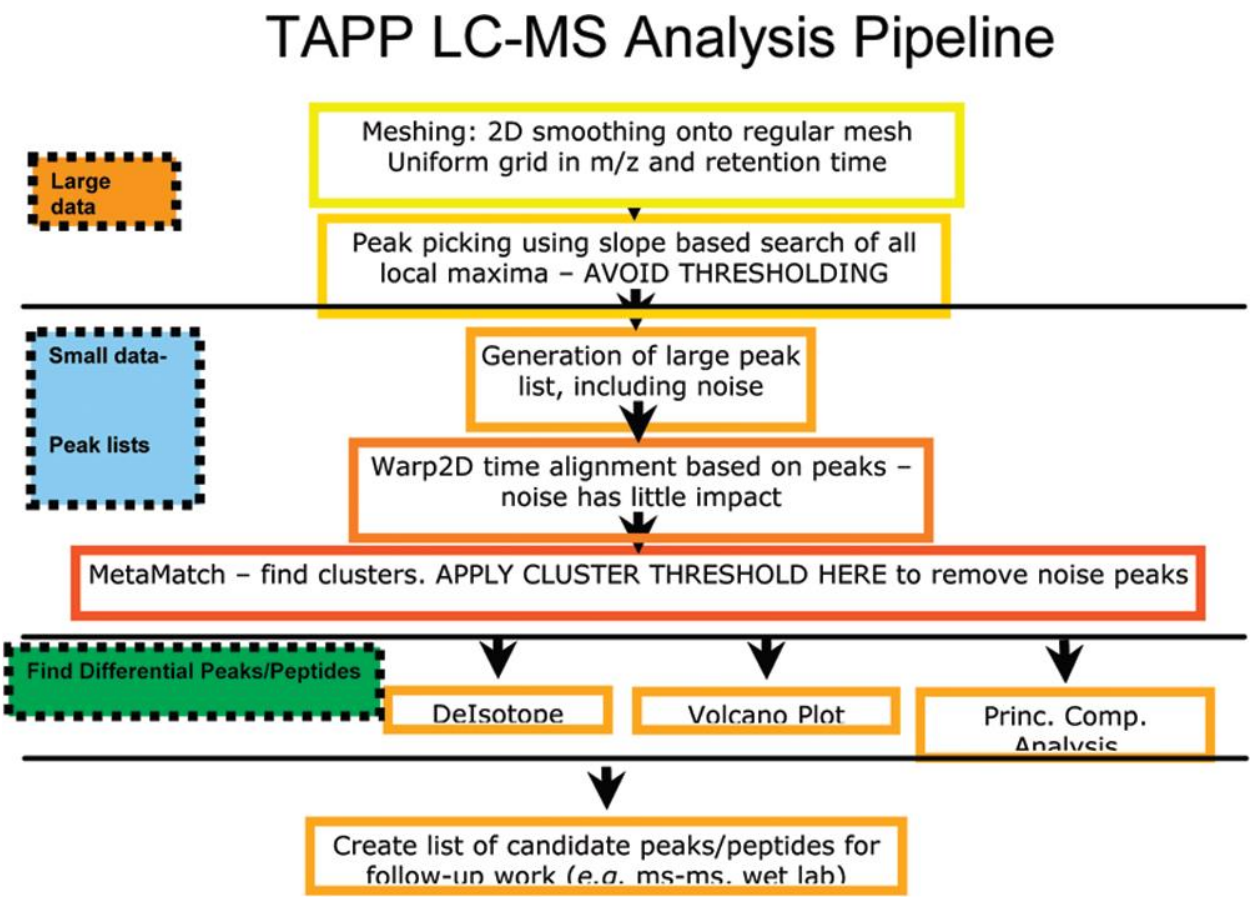
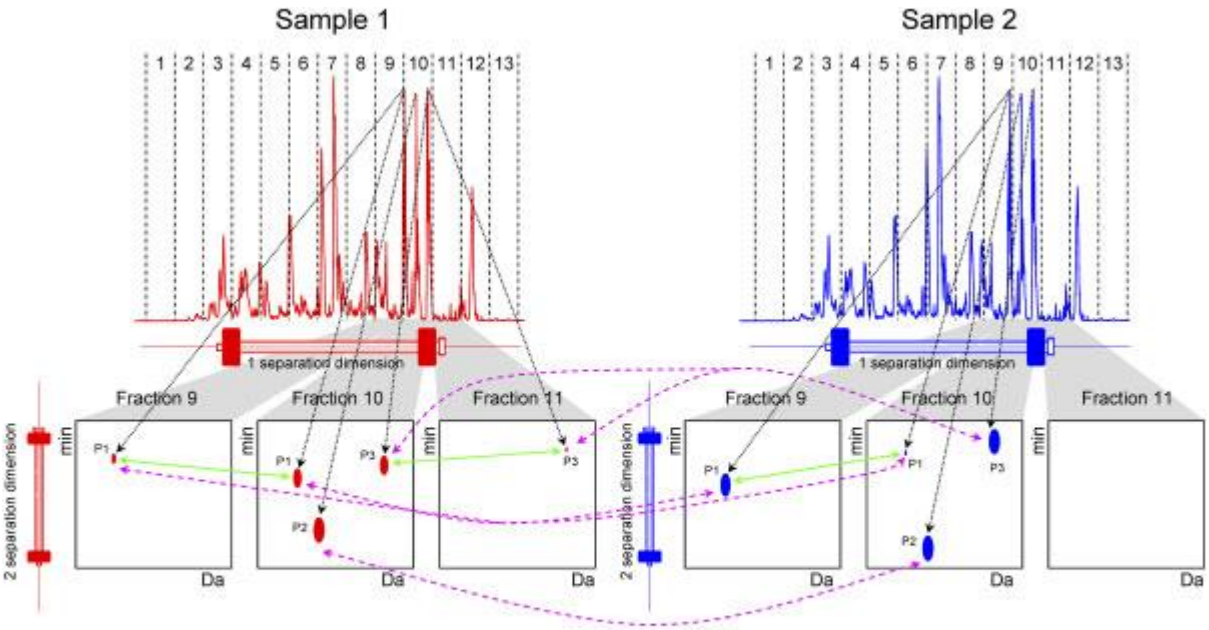


Figure 2 Outline of the Threshold Avoidance Proteomics Pipeline, showing the individual steps required to convert the raw 1D-LC-MS/MS data into a table that contains quantity of compounds in different samples amenable for statistical analysis [4].

Figure 2 shows the steps needed to convert the data raw 1D-LC-MS data into information that can be used for statistical analysis that has the aim of discovering biomarkers. The first steps are used to produce list of peaks present in the raw data (Figure 1) that are obtained without the use of thresholds, which would remove the low abundant peaks hidden in the background noise. In order to compare peaks from different chromatography they need to be aligned in LC dimension, since LC is prone to shifts in its retention time. Afterwards the aligned peak lists are analyzed using a clustering algorithm at which point noise is finally filtered out by identifying the same peaks in multiple LC-MS chromatograms (samples) that cluster closely in retention time and m/z coordinates. In the last steps of TAPP the peaks are converted into peptide information and the outcome is a table that contains quantitative information on peptides and non-identified compounds in different samples[4].

Research has also been performed in order to improve the peak capacity by performing two orthogonal LC before the MS [7]. However, TAPP in current stage does not include the ability to handle data that has additional LC dimension. The first LC will produce a number of fractions for one particular sample, where each fraction corresponds to a 1D-LC-MS map (Figure 1). The same compound of a sample may be present in multiple fractions, which will result in the same peaks appearing in neighboring chromatography. The goal is to compare the fractions between each other to find the same peaks present in multiple fractions in a sample and finally to match the same peaks between all samples as shown in Figure 3.



	Sample 1				Sample 2			
	Fraction 9	Fraction 10	Fraction 11	Sum	Fraction 9	Fraction 10	Fraction 11	Sum
Peak 1 (P1)	0.25	0.75	–	1	0.85	0.15	–	1
Peak 2 (P2)	–	1	–	1	–	1	–	1
Peak 3 (P3)	–	0.9	0.1	1	–	1	–	1

Figure 3 Example of comparing the same peaks in m/z and retention time (2nd LC dimension) between fractions (1st LC dimension) and samples [7].

The implementation of an additional LC dimension and the steps required to accomplish this task are defined by the following research question and sub questions:

- **What new algorithm can combine 1D-LC-MS/MS data of fractions for a sample taken in orthogonal liquid chromatography separation, using modules from TAPP?**
    - *Where is the optimal location for the implementation of the new algorithm in the current pipeline?*
    - *How can peaks generated by the same compound, but present in different fractions be identified?*
    - *What steps are required to create a solution in a way that its performance is limited only by the computational hardware?*

4 SITUATIONAL & THEORETICAL ANALYSIS

4.1 1D TAPP

In order to augment TAPP with the capabilities to process 2D-LC-MS data the following three questions must be answered: where, what, and how, must be implemented. For the scope of this project the TAPP will be considered to read raw LC-MS data in a mzXML format (standard format for MS data) up until preparing peak list. The resulting peaks represent isotopes of peptides generated by the MS fragmentation.

TAPP is composed primarily of two types of functional files. The first are binary executable files written in C++ that are designed to handle large amounts of data efficiently. These are the Grid, Centroid, Warp2d and MetaMatch [4]. The second are scripts written in Perl designed to pass data between executables, combining the individual steps into one coherent work flow. Additionally there are the variety of input/output data files like mzXML and dat, but most notably the peak list files. The peak list files are human readable ascii text files and come in both .pks and .wpks the internal format of which are exactly the same and contains information of peaks such as a unique peak identifier, position in m/z and rt, height, volume and more. Figure 4 shows the files involved in the first two steps of TAPP as well as the significant reduction in file size.

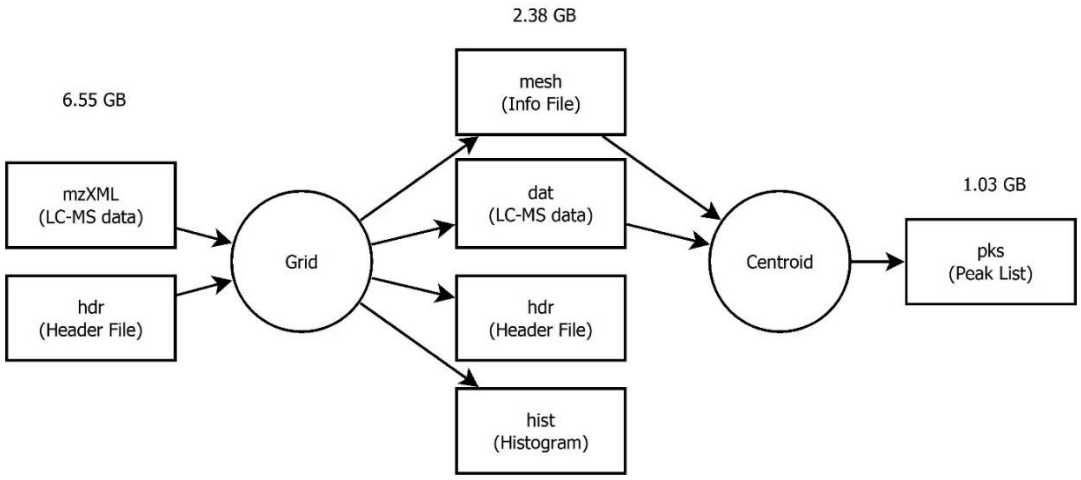


Figure 4 Data conversion of the first two steps of TAPP. The numbers above show the size of the data for one 1D-LC-MS/MS file typically processed by TAPP. Those have extensions of mzXML, dat and pks.

4.2 GRID

Grid takes the mzXML file that includes the complete data obtained by the mass spectrometer and uses information from the header file that contains list of parameters such as the type of spectrometer, resolution, range and etc. to arrange the data on a grid. The resolution of a mass spectrometer is dependent on the m/z and if mapped to a simple uniform grid will result in under sampling at one end and over sampling in another end of the m/z range. Grid uses a specific formula for each mass spectrometer in order to calculate the variable distance in m/z between points, resulting in a uniform sampling. Additionally Grid uses a 2 dimensional Gaussian function which smooths out the data, but it is soft enough to not affect the quantitative information of the peaks [4].

4.3 CENTROID

Centroid takes an input the mesh file created by the Grid module, which points to the .dat file, and looks for peaks in the data. Many algorithms base peak finding on a combination of local maximums with a threshold. When using a threshold, a critical factor is choosing one that separates the noise from the signal, however parts of the signal that are below the noise level are directly discarded. TAPP on the other hand takes all local maximum or apply very low threshold and extracts a mixture of noise and signal. Each peak is identified by its m/z and rt of its highest points and peak column and height is registered as quantitative information [4].

4.4 WARP2D

Warp2d is the next step in TAPP and it takes a reference and a sample peak list and has the goal to correct for shifts in retention time if any are present. It then proceeds to align the sample to the reference by warping the information [9]. The warp2d is based on the Correlation Optimized Warping (COW) algorithm, which is designed to compensate for the natural variability observed when performing LC and allow for separate chromatography data to be compared between each other that include retention time coordinate [10]. COW performs this by first split the sample chromatograph into a predefined number of equally spaced segments, which are stretched or shrunk and the position of the segment with the highest similarity is retained. In the original COW implementation correlation between the chromatography was used as similarity metric and the method works with single-stage LC-MS data[10]. Warp2d developed further this concept by using information from the m/z scale to calculate the metric by using the sum of overlap of the most abundant peaks between two chromatograms (Figure 5). Warp2D performs correction only in the rt dimension. The metric itself is the overlap between the reference and sample peaks in the segment being warped. The shape of the peaks approximates a Gaussian function, and since this is two dimensional Gaussian (three dimensional if counting the intensity as a dimension) the overlapping volume is calculated and used as the similarity metric in COW [9].

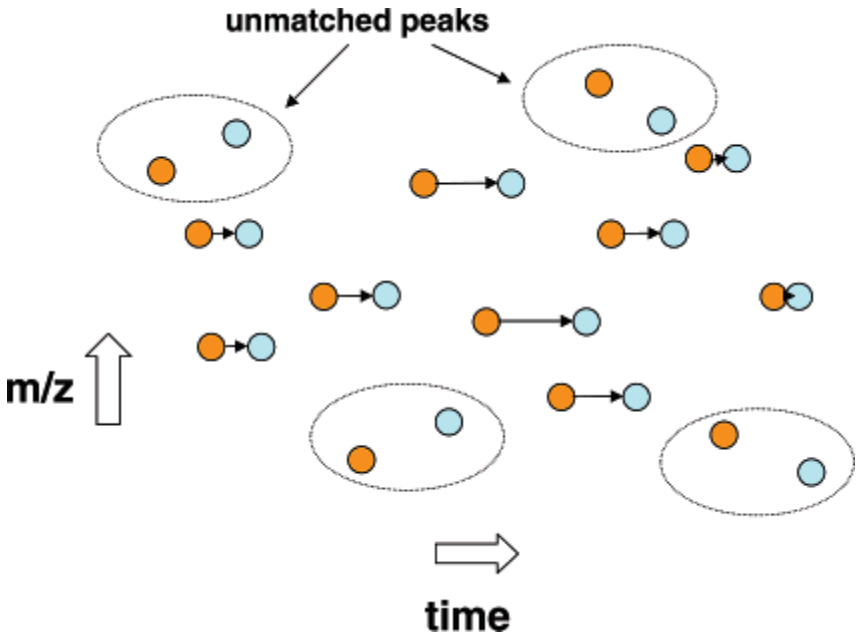


Figure 5 Principle of retention time correction applied by Warp2d aligning a sample chromatography to a reference chromatogram. Warping is performed only in retention time, by maximizing the overlap of peaks in two chromatograms [9].

4.5 MetaMatch

MetaMatch takes as an input a multitude of sample peak lists that have been aligned to each other with Warp2d and potentially grouped in sample groups (ex. Control and Disease samples). MetaMatch is a centroid clustering algorithm. MetaMatch does not match all peaks, but has the aim to find LC-MS signals that cluster together tightly in rt and m/z from different chromatograms at least from one sample group. During the execution of this algorithm a maximum cluster size in rt and m/z is defined and a threshold is defined to found peaks within this cluster. Peaks that do not end up in a cluster become "orphans" and are considered as noise or peaks from compounds that are too spare in the sample group. Peaks that are close enough to each other and fall in the selection criteria are extracted as Meta Peaks. Figure 6 shows clusters and orphan peaks after MetaMatch clustering. This process rejects a large number of peaks across multiple files that are considered irrelevant so it further reduces the size of the data files (ex. 150 000 Meta Peaks and 2.6M orphans from 35 peak lists was obtained in an example data pre-processing analysis) [4].

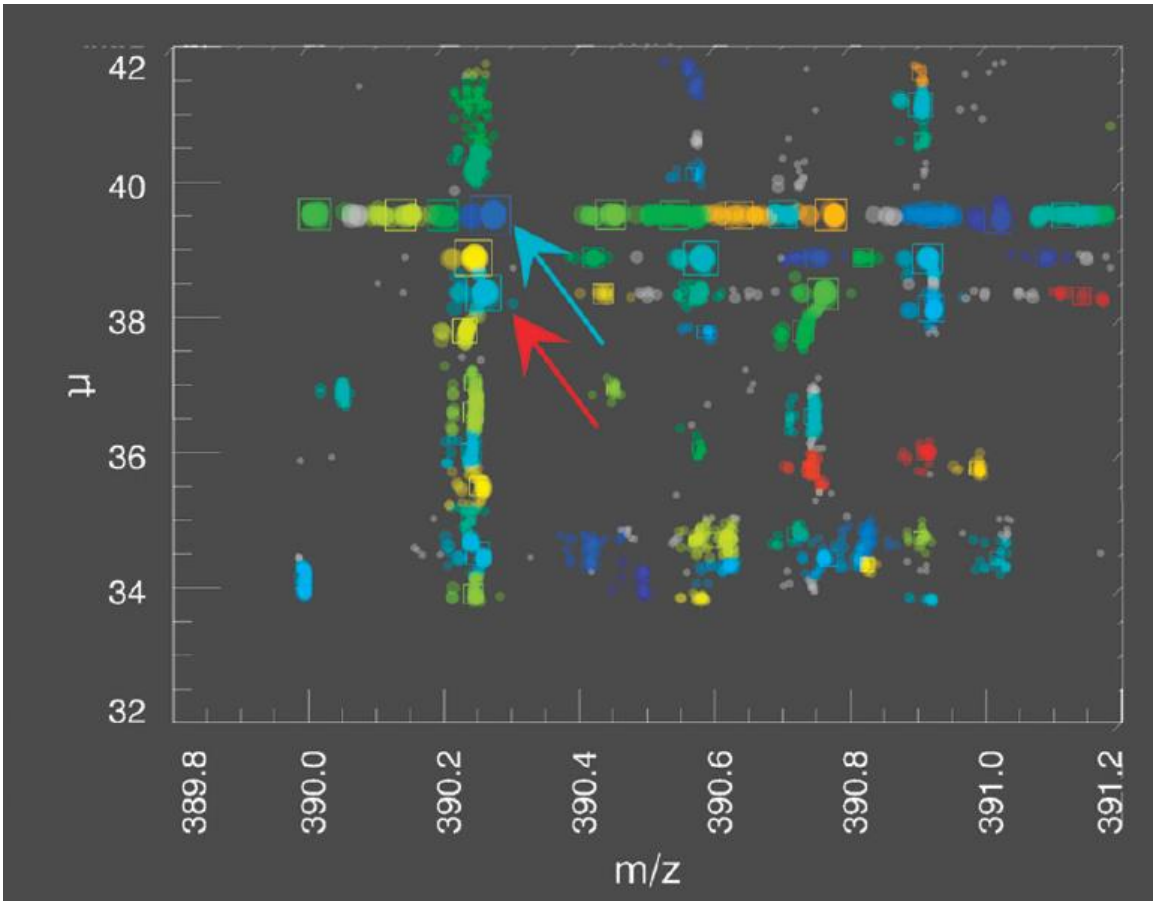


Figure 6 Data selection showing the clusters and orphan peaks created by MetaMatch. Peaks are represented by circles, which are colored based on the clusters they have been assigned to. The cluster colors are chosen randomly except for grey which represents orphan peaks. Squares indicate the location and size of Meta Peaks clusters. The arrows point to peptides which have been identified further as critical information for the specific experiment [4]

4.6 PERL WORKFLOW

All four of these C++ binary executable files are fully functional and can be executed individually on a set of samples, however this would take a significant amount of effort and time to execute all binary modules as CLI commands manually. Furthermore, when executing these commands multiple times on files with similar names the chances of human error are quite high, resulting in incorrect or inconsistent results. To avoid all these issues a number of scripts have been developed in Perl have been implemented [11]. These include functionally such as:

- Storing the variables needed by the individual steps in the one configuration file, making it easy for a user to set all variables in one location.
- Smooth execution of a large amount of repetitive operations and error free transfer of data files between each step of TAPP.
- Allow TAPP to use multiple threads and processor cores for better utilization of computing resources.
- Present the user with a Graphical User Interface (GUI) for the workflow with a small learning curve, instead of relying on Command Line Interface (CLI). E.g. in implementing the workflow in Galaxy.

Additionally, with the Perl workflow implementation comes with a number of recommendations with respect to follow up work on the pre-process data [11]. The first one is with respect to General coding conventions like naming of functions, methods, variables, constants and etc. during programming [12]. Although not mandatory, such conventions may overlap between different programming languages. They are considered good etiquette during programming, since it allows for easier understanding of the code by other individuals working on the same project, at this or future time point. The second recommendation is for future algorithms to include the option for parallel computing and follow Data Oriented Design (DOD) [11].

4.7 Parallel Computing

Parallel computing is a wide definition but in this case the focus would be to use multiple nodes working in parallel to produce one result. Due to physical limitations, a single core on a processor can only perform up to a certain speed (CPUs rarely surpass speeds of 4GHz), so modern CPUs have multiple cores. In many computer programs are not designed in a such a way as to make use of multiple cores at once, or can use up to a pre-set number of cores, resulting in wasted processing resources. Most programming languages support multi-core functionality using libraries/modules, which makes parallel computing independent of the underlying hardware and software. This makes coding algorithms easier, however the algorithm itself still must be designed in a way that it can be split in multiple data and/or instruction streams [13].

4.8 Data Orientated Design

The last recommendation is for new developers to use DOD when working on TAPP. The currently more widely spread method of programming is called Object Oriented Programming (OOP), which is a tree like structuring of code around the concept of classes inheriting properties from upper level ones. In OOP the data is spread across a number of interconnected objects and results are split in multiple objects. An object can be for example a vehicle and stored information on the object are e.g. velocity and weight. When performing calculations just for one object that is perfectly fine. However, if there are thousands of cars and if the goal is to calculate where they will end up after a few seconds, each car needs to be called, and extract individually their corresponding velocity. When calculations need to be performed on a large set of the same type of data DOD, recommends grouping it directly together instead of spreading it throughout a number of objects, which allows for faster access to the data e.g. by using efficient indexing. This concept is shown in Figure 7[14].

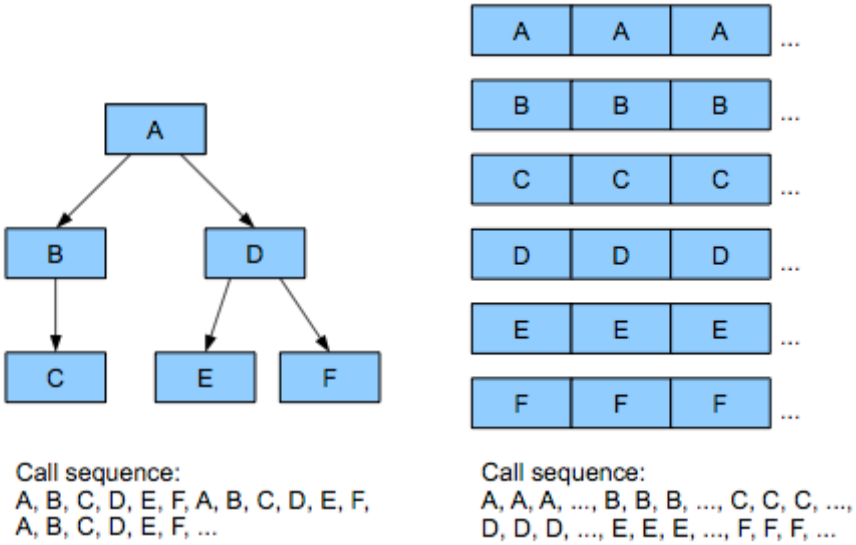


Figure 7 Comparison between Object Oriented Programming (left) and Data Oriented Design (right). Each letter represents a different type of data, which needs to be accessed. In the left case, the program needs to get each type of data from a single object and only then proceed to the next. In the right data is grouped based on type and not to which object it belongs to [14]

4.9 MAX PLANCK WORK ON 2D-LC-MS

So far only the LC^n-MS work within the Groningen Research Institute of Pharmacy has been reviewed, however the Max Planck Institute of Biochemistry is also performing research within this area. They have developed a quantitative proteomics software package designed for analyzing large MS data sets, called MaxQuant [15]. This software is capable of label-free quantification of peptides spread across multiple LC fractions. In their paper from 2014 they have described many of the algorithms they are using. With respect to the matching of peaks between LC fractions they only state that they consider matching peaks in adjacent fractions to be the same peak and [16]:

> *"Further details on the alignment and matching algorithms, including how to control the FDR of matching, will be described in a future manuscript."* - [16]

MaxQuant is free, but only packaged binaries based on the .NET 4.5 framework are available, and it is not possible to analyze and reuse their program code.

Apart from MaxQuant all other information found for 2D-LC-MS was either based on tandem Mass Spectrometry (MS/MS) or labeled. LC-MS/MS data consists of non-fragmented mass spectra and fragmented spectra. During the elution of peptides from the column the most abundant non-fragmented peaks are submitted to fragmentation allowing identification of compounds (peptide in proteomics sample) [17]. The assignment of compound identity to non-fragment peaks allow, to match an identified peak across multiple chromatograms. However, many peptides co-elute in the second LC and these peptides render precursor ion selection and MS/MS-based identification incomplete, but accurate. Furthermore, only a small number of peaks can be selected by the spectrometer for further fragmentation and the selected peaks are the ones with higher amplitude. This moves away from the goal of TAPP to be able to analyze all compounds including low abundant ones [7]. The labeled MS is also not a choice, since it limits the number of analyzed samples and has high cost [18].

5 CONCEPTUAL MODEL

Using the information reviewed in the previous chapter it is now possible to answer the questions needed to allow for the TAPP workflow to function with 2D-LC-MS.

5.1 WHERE

Taking into consideration the significant reduction of data size between each of the TAPP steps it is best to implement the solution as further down the flow as possible. Although, this would require performing the steps for every LC fraction, the modules used in those steps are optimized for efficiency even if they do not support parallel processing within them. Parallel processing is desired from the scope of the entire work flow so it is possible to execute the same step at the same time on different fractions. Furthermore, it is possible in the future for these modules to be further optimized (ex. Using GPU processing) and they can simply be swapped with their predecessors without the need to modify the workflow.

The first two modules offer the most significant data reduction, thus analysing the relation between fractions is best performed after them. However, MetaMatch is used to remove the noise using different samples, which means that in order to work with 2D-LC-MS it ether has to be implemented on all fractions at the same time or be implemented for large set of fractions. The first option would require significant modifications to the C++ code, which can be complex for implementation. It is preferable to modify the Perl workflow rather than its individual components. On the other hand, the latter option introduces its own challenges.

It is well within reason that compounds may have experienced different amounts of shifts in their retention time from the first LC, resulting in a varying distribution in LC fractions between samples. Thus a real peak can be present in the 6^{th} fraction of sample 4, but not in the same fraction for any of the other samples, resulting in it being orphaned and considered as noise. So identifying peaks across samples needs to be performed before the noise removal, but it also needs to be considered after performing MetaMatch between samples. The result could be that MetaMatch removes a peak, but the 2D-LC-MS algorithm returns it afterwards because the same peak was passed by MetaMatch for another fraction.

5.2 WHAT

With the optimal location being around the Warp2d method it is important to considered the requirements for peak comparison. As is handled in the 1D-LC-MS, peak lists need to be aligned before compared. Warp2d is already capable of efficiently performing this operation between samples it can also be used to align fractions if they show important overlap in constituting compound. One of the outputs of Warp2D is a quality file, which includes the mean overlap ratio before and after warping the peaks. This metric is effective, since it includes intensity, position, and width of the peaks in 2D space, and the geometrical and arithmetical mean of the sum of overlapping peaks in the reference and the sample chromatograms. Taking into consideration the spread of peaks between fractions it's possible to limit the number of fractions that need to be compared based on result of Warp2d. Figure 8 shows the result of testing this concept.

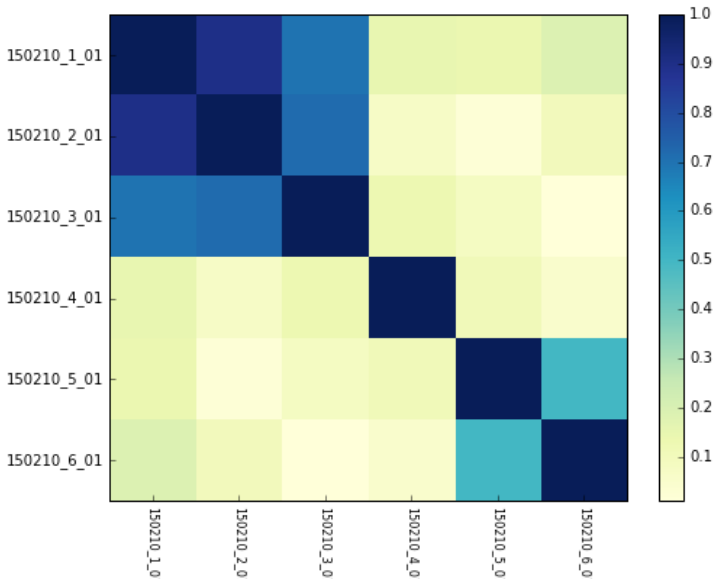


Figure 8 Heat map of the mean of the overlap of 2d Gaussian peak ratio after applying Warp2d on 6 fractions from the same sample. The y axis shows samples used by for reference and the x axis the samples that were aligned (i.e. rt of them corrected for monotonic shifts) . This shows that the first three fractions and the last two have a significant amount of overlap.

Two things can be drawn from this preliminary test. The first is that the result is symmetrical to the order of selecting sample and reference chromatograms, so aligning one fraction to another or with the opposite order results in the same mean warped ratio. The second is that peaks are not equally spread between fractions. This results in the possibility of the same peaks present in only a single fraction or multiple fractions within the same sample. This information is useful to identify groups of fractions that should be analysed further in more detail and exclude fractions that do not overlap with their neighbors.

After identifying fractions that share peaks, it is necessary to compare the individual peaks in them. On first glance this can be potentially performed using MetaMatch, however some problems become apparent after considering its functionality. MetaMatch is designed to take a large number of samples and cluster them together to filter out noise. In cases of fractions, the amount of input peak lists for each group will be lower and the algorithm needs for a peak to be present in a pre-set number of them. With this method having more similarity produces higher quality results, however if the LC of the first dimension has more separation power the similarity between adjacent fractions should be lower. The fraction comparison in Figure 8 is obtained by using a spin-column chromatography, which does not separate the sample well, but has high sample throughput (a large number of samples can be separated at the same time) [19]. So having almost full overlap like between the first three fractions in the heat map is not the representative of better quality separation approach such as High Performance Liquid Chromatography (HPLC).

In order to compare peaks between fractions it is possible to use a variety of methods. Almost all comparison methods use a metric, which can be complex like the volume overlap of Warp2d or simpler like the distance between the centres of the peaks like MetaMatch. The distance between peaks is already small due to Warp2d, so using another complex metric will most likely not produce better results compared to a simpler one.

An additional benefit of working with LC fractions is that the shared peak intensities become a significant factor. Since peaks are already assumed to have a Gaussian distribution for the 1D-LC-MS, the same can be assumed for their relation between fractions. If three fractions share the same peak, then its logical that the peak in the middle fraction should not have a lower intensity (excluding quantification error) compared to the two others. Furthermore, if a peak is several orders of magnitude higher in one fraction than its neighbour it can be excluded. Within this step it's possible to include additional criteria that matching peaks must pass in order to consider them caused by the same peptide.

Finally, the results of linking peaks between fractions of the same sample need to be combined with the information from MetaMatch, so it can be used by the peptide labelling algorithms that normally follow. Since the goal is to quantify the peptides, this procedure needs to complete two tasks. First one is linking each peak between the samples and their individual fractions. Some samples may have a common peak that may be spread between a different number of fractions and it is assumed that the peak will have at least one fraction overlapping between samples. The second task is sequential to the first and it is used to obtain the total intensity of a peak between fractions, allowing analysis of the quantitative information of the same compound between samples. This procedure is graphically represented in Figure 9.

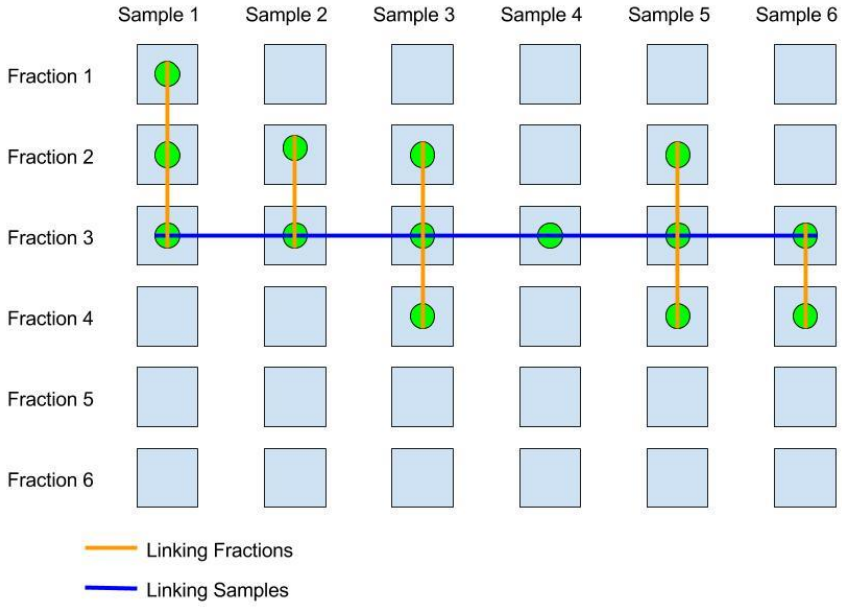


Figure 9 Example of linking the same peak between samples and fractions as part of the 2D-LC-MS algorithm.

In the presented scenario, MetaMatch would most likely assume that the peak in fraction one of sample one is just noise and reject it. Using the linking between fractions it is discovered that during the analysis of fraction 3 that the first and second peaks is the same peak as in fraction in sample 1.

5.3 How

As stated earlier modifying the existing modules written in C++ requires considerable efforts, thus the 2D-LC-MS workflow should be implemented using a Scripting language. Although the current flow uses Perl for this purpose, it also has a high learning curve compared to an alternative language such as

Python. The 2D-LC-MS algorithm can be implemented in Python and then with small modifications in the Perl scripts be included in the TAPP workflow.

Although the language itself will not be the same as the one used by previous individuals working on the pipeline, the recommendation for using DOD is to be followed. The advantages with respect in handling large amounts of data of DOD make it an obvious style for working when analyzing large peak lists. However, the discovered fraction groups during the algorithm will be treated individually, more in the style of OOP.

Generating a full Warp2D heat map will prove a significant challenge for the algorithm, since the computational time will rise exponentially with the number a fractions. For this reason, a step by step approach is preferable. Since a threshold will need to be used to decide which fractions share considerable amount of peaks, it is possible to start analyzing fractions next to each and checking their value. Only if they pass the threshold should the further fractions be considered. Furthermore, using this approach it is possible to start analyzing the peaks in a fraction group before all fractions are analyzed, if the conclusion that no other fraction will join the group is made.

In order to obtain optimal results during peak comparison, it is important in which direction peak lists are aligned using Warp2D. If only two fractions are compared it is irrelevant, which is the reference and which is the sample. In the case of three or more however, for best results is preferred to choose the middle fraction as reference. In the case of more than three, chain warping may need to be implemented. Chain warping means that fractions not directly next to the reference fraction need to be aligned to the next fraction that is closer to the reference fraction and already aligned to the next closer and so on. During the generation of the fraction groups warped peak list files are created, some of which can be reused, saving computational time. In order to have a reference in the middle for more than three fractions some additional alignment is required that would correspond to parts of the lower diagonal.

When comparing peaks in a fraction group all the peak files in that group will need to be loaded into memory. However, since some peaks in one list will not match to peaks in another, it is only necessary to save linked peaks in a file for further use. Additionally, since the peak list format is the same throughout the workflow it is undesirable to modify it, thus the better option is to create a new file including the link information between peaks in fractions that references the peak lists. This information can only include information like the peak IDs for the same peak linked across fractions and the total intensity of the peak.

The opportunities for using parallel processing come from three locations in this concept. The first is parallel alignment with Warp2D procedures for finding the fraction groups. Multiple Warp2D operations can be performed at the same time as long as they do not need the result of each other. This concept can be explained using again Figure 8, only a single Warp2D can run on a row, but multiple rows can be analyzed at the same time. The second option is when analyzing fraction groups, since they do not share peaks. A method can be called to analyze a fraction group for each group that is considered complete. The third parallel processing option can be performed during the linking between fractions and samples. Each set of the same fraction across samples can be processed individually as long as all the required links from the previous step are completed.

6 RESEARCH DESIGN

The conceptual model can be split into the following three steps:

- Generating fraction groups
- Linking peaks between fractions
- Analysing peaks between fractions and samples

The steps obviously needed to be developed sequentially, but tested individually by using pre-generated files from previous successful tests. The development of each step followed a similar procedure. Additionally, the full 2D-LC-MS algorithm was developed using approaches described below.

6.1 FINITE STATE MACHINE

First steps in designing the algorithms was performed by using a simple logic and creating a Finite State Machine (FSM). FMS is an abstract machine that can be in one of a finite number of states at any given point in time. The machine can change its state based on conditional transitions. This step is beneficial to the overall development process for a number of reasons. Primarily it simplifies the problem to a point where new ideas related to the execution flow of the algorithm can be quickly implemented and tested. Since it does not deal directly with actual data it is prone to errors in that respect, which are best solved after the FSM is implemented into code. An additional benefit of working with FSM is that it can be used to better and quicker understand the final code. This included the developer when working on the following steps as well as future individuals that need to work on the project.

6.2 FIRST IMPLEMENTATION USING IPYTHON

After the structure of each process was refined using FSM it was tested. iPython offered a good environment for quick prototyping of code, with the benefit of block structure and integrated HTML capabilities. The result was that each individual component of the FSM can be neatly worked upon. Furthermore, the iPython implementation is a good environment for visualisation of limited amounts of data. The code was tested with a small data set and the performance of the algorithm was assessed. (the heat map in Figure 8 was generated using iPython). Combining visual output with HTML results in yet another good resource to assess the quality of the processing.

6.3 OPTIMIZATION ON WORKSTATION

One of the down sides of iPython is that its ease of use comes at the price of added overhead. In order to optimize the code for better performance it was converted from the iPython .ipynb files to simpler .py files. The iPython interface is capable of performing this operation however the added overhead needed to be manually removed. Furthermore, the code was restructured by converting the iPython blocks into methods. This is the point at which multi-process support was implemented and loading of constants through the help of configuration files. At this stage the functionality of the script was tested using larger slices or a full data set. The data size used was dependent on the processing power of the workstation used.

Making modifications to the script was still straightforward thanks to using PyCharm as programing environment, which offered great programing support. Once the .py version of the script was completed

its performance was assess with respect of processing power and I/O limitation. Workstations generally do not have the computing power to execute the full script at what could be considered a reasonable time and can hinder the usability of the station for other purposes when the code is running. For these reasons the number of tests were kept to a minimum and in many occasions successful test results were reused from previous steps.

6.4 DEBUGGING ON COMPUTATIONAL CLUSTER

The final step is transferring the script to the high performance machine (HPC) that has been designed for the purposes of performing intensive processing tasks. In this specific case that would be an Open SUSE (Linux Distribution) remotely accessed cluster. The changes required for the workstation version of the script to run in HPC environment are primarily based on assessing if all modules used are installed on the cluster. Many versions of Python exist such as the Python distribution called Anaconda, which resulted in a mismatch between the used modules and the modules present on a system. Once all used modules were successfully imported the script needed final edits in order to run successfully. The modifications at this point were kept to a minimum, since they were performed using simple text based interfaces, which did not provide any programing support.

Finally, this is the stage at which the individual components of the 2D-LC-MS algorithm were tested, since the cluster provides large amounts of RAM and large number of CPU cores. When running the workflow components, the files generated by previous tests can be used as well to decrease the processing and testing time, however full testing of the final algorithm and assessment of its performance was performed as well.

6.5 TEST DATA

To test the functionality of the linking of peaks between fractions using TAPP 2D, samples from a Chronic Obstructive Pulmonary Disease (COPD) study were used. The full data set is composed of 10 control samples and 10 COPD samples each consisting of 6 SCX fractions. However in order to test the functionality of algorithm only 3 of each were used. Samples 1, 2 and 8 are control and 11, 12 and 16 are COPD. With respect to the linking between fractions, since data is processed only per sample, the sample group in which the sample belongs at this stage is not relevant.

During testing the test dataset was extended with two additional sample sets in each class resulting in a total of 10 input samples. This was performed to improve the performance of MetaMatch when calculating Meta Peak using samples for each fraction. The noise filtering performed by MetaMatch is based around the presence of the same peak in a preset number of samples in at least one sample group. In order to test the algorithm for merging MetaMatch results per fraction using the linking information generated by the previous step, its required that a peak be present in more than half the samples of one class. The final data set used for testing was:

- Fractions used in all samples: 20mM, 40mM, 60mM, 100mM, 500mM and 1000mM
- Samples of Class Control (0): 1, 2, 3, 5 and 8
- Samples of Class COPD (1): 11, 12, 13, 14 and 16
- Total of 60 input files

6.6 ALGORITHM VALIDATION

In order to confirm that the output of the individual modules is correct the following methods were used.

6.6.1 Linking between fractions

The goal of the linking module of the 2D-LC-MS algorithm is to connect peaks that are close to each other between two or more fraction files of the same sample. The output was assessed by visualising the connected peaks between adjacent fractions, and assessing the distance between them. The linking modules output includes identification numbers of the linked peak in each fraction. With this information the peak list files involved were opened and plotted. Taking into consideration the small drift between data in the m/z dimension, observed in the current 1D-LC-MS pipeline, the distance between two linked peaks in this dimension needs to be 0.1 m/z (rough estimation). With respect to rt, the drift observed was considerably higher so reliably defining a maximum distance is not possible. For these tests the max rt width used by Warp2d of 0.5 rt is used here as well.

The correct performance of this algorithm can be assessed by confirming that the closest peaks have indeed been linked together. However, since the module is relying on a cut off distance between intensity ratio of the two peaks in the two fractions, the later parameter needs to be explored in greater detail to obtain an optimum value. This should follow the techniques used during the previous development stages of TAPP like the ones outlined in parameter tuning of the TAPP paper [4].

6.6.2 Fusion of Meta Peaks using Linking

Once there is information describing the Meta Peaks on a per-fraction basis and links between fraction on a per-sample basis, this information is merged together in a new set of output files. This final output includes meta peaks and the quantitative value of their member peaks have been corrected using the link information (i.e. split peaks in different fraction combined to one quantitative value). In order to confirm this result without the use of steps further down the proteomics pipeline (peptide identification), the meta peaks can be assessed based on the peaks they include. All peaks that are a part of the same meta peak need to be in a small m/z range (for example the once stated earlier of 0.1m/z), and within a reasonable range in rt. Again retention time is hard to define. However, when considering the properties of LC, it is expected that when using a large sample set, the peak locations in rt will follow normal distribution.

6.6.3 Overall validation

As mentioned previously the most reliable way for validating the result of the 2D-LC-MS pipeline would be on a peptide basis. This requires that peptide identification be performed on the final meta peaks and be compared with results using other analysis methods. The most reliable of those would be based on MS/MS identifications on the same dataset (available in the original mzXML files). Peptide identification however is outside the scope of this project so this is left to follow up project.

7 RESULTS

7.1 ALGORITHM OVERVIEW

To begin with Figure 10 represents an overview of the current 1D TAPP workflow. The individual modules are being executed using a Perl script. All the configuration is taken from the JSON file, in which each module's variables are categorized for convenient modification. From the conceptual module it was established that the modifications to this workflow have to be after the generation of peak list files (.pks). However, during work on the algorithm it was established that additional modifications would be required before performing any data processing, in order to achieve correct functionality.

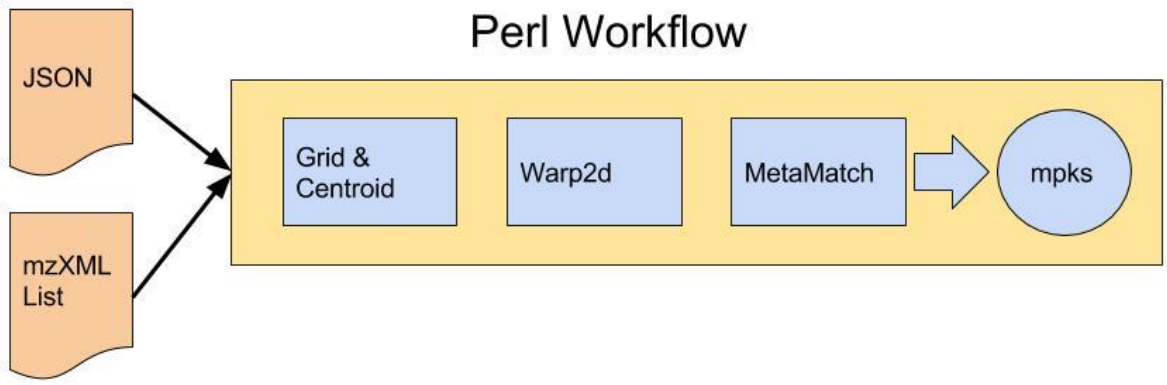


Figure 10 Overview of 1D TAPP workflow showing data flow from the left to right. The JSON file stores all parameters that are required to configure the workflow. The mzXML list includes the names of the initial data as well as classification of samples. Each sample belongs to a category (for example Disease and Control samples), which is used by Meta Match during clustering.

The JSON file can be easily augmented with additional variable fields, which are to be used by the new sections of the workflow. The mzXML list on the other hands requires two additional variables for each entry (input mzXML file). For the 1D TAPP it is known that each entry is a single sample, however for the 2D TAPP an entry is a fraction of a sample. Additionally, due to the behavior of the LC fractions, the order of the fractions is of great importance as well. This means that the new solution should have a new format where mzXML list includes sample and fraction identifiers for each entry as well as the class identifier.

The output of the 1D workflow are a set of files which include a list of the clustered Meta Peaks (.mpks), a list of peaks that were left unused (Orphaned by MetaMatch)(.orph), a list of Meta Peak identifiers and the original peak identifiers that are combined in a cluster (.pid) and a file including statistics on the clustering performance of the MetaMatch (.stats). These files combine information from all mzXML files and are used further on in other operations like identifying peptides. The 2D version of TAPP needs to be able to present the same outputs to the user and additional information like the fractions to which each peak belongs.

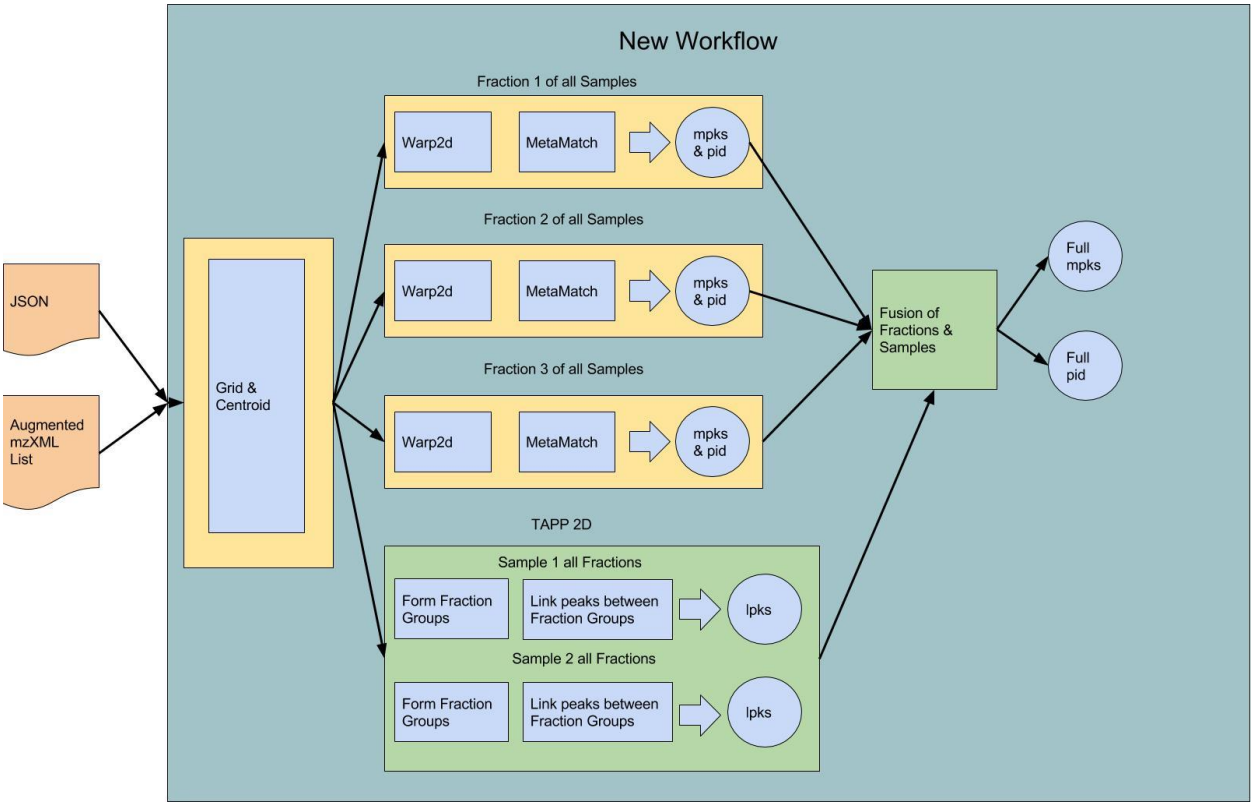


Figure 11 Overview of the 2D-LC-MS workflow. The yellow fields represent use of parts of the current Perl scripts and green ones that have been developed during this project using Python.

Figure 11 shows the new 2D TAPP workflow and again covers the flow of information from mzXML files to the same output files as the 1D TAPP combining fractions and samples. This solution uses as much as possible from the original workflow in the yellow fields. Following the conceptual module all files are processed from mzXML files to peak lists using the existing Perl scripts. Within the JSON file its possible to specify which operations to be executed so it is straightforward to perform only Grid and Centroid on all input files. Once that is performed the complexity of the solution escalates, but on data which is much smaller in size.

Using the example in Figure 9 it is possible to view the required operations on the files by columns and rows. When several samples from the same category are split into fractions, the same compounds would be present in the same fraction of each sample. The reason that a peak is split based on the same physicochemical property and the similarity between the same fraction of two samples is assumed to be greater than that of two fractions of the same sample. This assumption of similarity is of course highly dependent on the samples in question and the LC used for fractionation and the number of fractions taken, but in general can be considered true.

What results from this is that, it is possible to follow the current 1D processing in order to obtain a comparison between the samples, but only on a per fraction basis after correction with peaks split between fractions. Thus if there are 6 fractions for each sample and the solution is to perform 6 1D TAPP workflows. The result is pid and mpks files for each fraction, which now need to be combined. When viewing Figure 9 this would represent the horizontal operations.

A large part of the new solution involves the processing of the data between fractions of the samples, which is a vertical operation. This is performed by the TAPP 2D algorithm implemented in python, and discussed in greater detail in the sections that follow. The importance with respect to the overview is that the new TAPP 2D module generates a new type of files named Link Peak Lists (.lpks), which provide identifier information and height of the peaks that were combined using the algorithm. Keeping in mind the possibility of large number of fractions (more than 20) for each sample multiple .lpks files may have been generated each describing one fraction group. This concept was discussed earlier in the conceptual module for the new algorithm.

The result is a set of mpks and pid files for each fraction, and a set of .lpks files for each sample. Since both the .pid and .lpks files were generated using the same original peak lists, the peaks within them can be directly related to each other using their set of identifiers. In the final step of the new workflow, all peaks that are essentially to summarize result of the same compound in different samples and fractions are grouped together and the mpks files can be combined into a single mpks file and single pid file. These files contain quantitative information of the same compounds in different samples for the complete dataset and which information in table format can be used for subsequent statistical analysis.

7.2 TAPP 2D

The module called TAPP 2D handles the processing of data from peaks list to lpks files. Within the fractions of each sample are processed in parallel. This module is capable of running multiple sub processes allowing for the use of multiple processors at the same time. The first responsibility of the module is to load configuration from the input JSON file and pass the appropriate parameters to the modules that are required to fulfill their function. TAPP 2D generates a Fraction Set for each sample, and a Fraction Group for each fraction. Populates the location of input pks files in the File Manager.

Another main functionality of the module is controlling the use of the multiple processors. Each fraction group for every fraction sets processes information on its own. However, when a heavy computational operation is required like Warp2D and fraction linking, it passes a request to TAPP 2D. TAPP 2D places these operations in a list and starts processing them using as many CPU cores as specified in the JSON file using a queue implemented in the file manager. It also tracks the status of each operation and when one finishes it starts executing the next one in the list.

Once the list is finished TAPP 2D initiates another cycle of operations to all fraction groups that are still require more processing. Each group performs its calculations using the newly generated data and either resolves or creates a new request. Finally, at the end of each cycle TAPP 2D checks the status of each group, and goes on until all groups have been fully processed.

7.2.1 Fraction Sets

A fraction set is a class that contains all the fraction groups for a single sample. Its main functionality keeping track of all the groups contained within it and pass information back and forward between the individual groups and the TAPP 2D module. During its initialization each set generates a fraction group for every fraction. Additionally, a fraction set is capable of creating a summary of information for every group within it that can be written to a file for use further on.

7.2.2 Fraction Groups

Initially each fraction is considered a fraction group containing only itself. Each group has a status important keeping track of what operations need to be performed on it. The following 6 states have been defined with the module:

- Active - Group has the potential to expand on its own or be merged with another group.
- Unresolved – Group has no potential to expand, but may be still merged with another group.
- Resolved – Group has been defined to include a number of fractions.
- Centered – The central fraction to which all others should be referenced has been identified.
- Warped – All peak lists and warped peak lists required by the group exist.
- Linked – Linking has been performed on the fractions of this group.

All groups start in an **active** state and need to reach a **linked** state or be merged with another group. Groups have a "do" method that behaves differently based on each state, thus the TAPP 2D method can instruct each group to do its own operations until all groups reach a **linked** state.

Following the behavior of fractions discussed in the conceptual model using Figure 8, each group by itself my change it status or request a Warp2d operation based on the mean sum of peak volume overlap between the fraction and its neighboring fractions. Each group begins from the row and column of the index of its original fraction and during each step it may move one column to the right or change its state. This behavior is shown in Figure 22 in Appendix A by using a Finite State Machine.

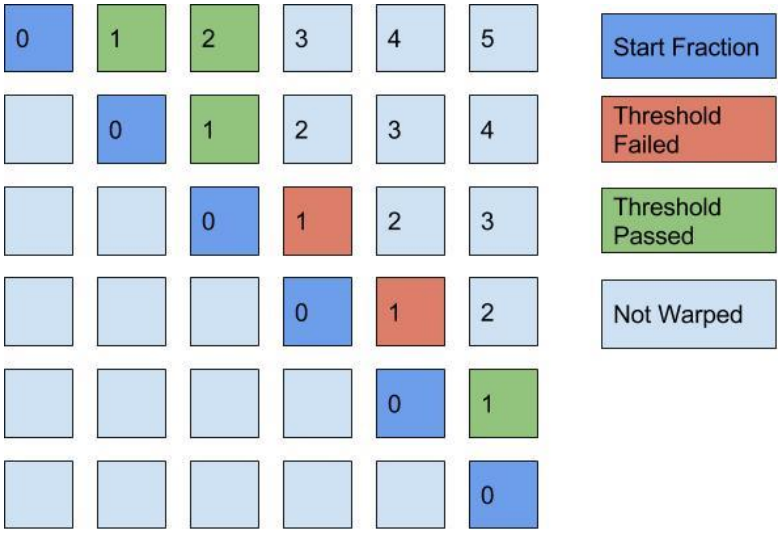


Figure 12 Example operation status to calculate Fraction Groups. Each fraction group initially only includes its starting fraction and starts moving to the right measuring its overlap with the next using warp2d.

An example of the functioning of the fraction group can be presented using the results shown in Figure 8 in Figure 12. During the first cycle of operations groups 1, 2 and 5 pass their thresholds, 3 and 4 do not and 6 is discovered to have reached the end. At this point it is already known that:

- 5 and 6 are a group 4 did not pass its threshold. Furthermore, since it's the end of the row that group is resolved.
- 4 is in a group with only one fraction.

- 1 and 2, and 2 and 3 are grouped together, but need an additional step to see if they will merge into a single group.

Since each group processes information on its own, the bottom group that includes fractions 5 and 6 can already proceed to the next steps. States may be skipped by a group depending on its size, using the following logic:

- A group that includes only a single fraction can be directly considered **linked**
- A group that includes two fractions can be jump to the **warped** state and requires only linking between the first fraction in the group and the already warped second fraction (to the first one)
- Groups with three or more fractions need to pass through all states

These conditional steps allow for smaller fractions to finish processing earlier, and avoids a large fraction group which is slowing down the overall data processing. Furthermore, during testing of code it is possible to stop the processing up to a certain state.

Groups with more than three fractions require all steps, which perform the chain warping mentioned in the conceptual model. In the case of three fractions in a group the chain warp requires only a single additional Warp2D to be executed between the second and the first fraction in the group. Each additional fraction in the group requires one more warp towards the central fractions.

If a group has more than three fractions and has an odd number of fractions the middle fraction is considered as the center of the group. When there is an even number of fractions than the middle two are compared, using the already existing Warp2d results, based on their mean overlap ratio with their respective neighbors away from the center. At this point the group is considered **centered** and chain warping can be performed on it.

To give an example of the transition between a **centered** and **warped** state, assume that a group has 6 fractions. Thus the following steps are performed:

- Fraction 3 is considered the center since the overlap between fractions 2 and 3 is larger than that between fractions 4 and 5.
- While resolving the group a warp between fractions 3 and 4 already exists (3←4).
- Warping first requests the following two Warps: 3←2 and (3←4) ←5.
- Afterwards it requests: (3←2) ←1 and (3←4←5) ←6

With this the resulting files are:

- Fraction 1: 3_2_1.wpks
- Fraction 2: 3_2.wpks
- Fraction 3: 3.pks
- Fraction 4: 3_4.wpks
- Fraction 5: 3_4_5.wpks
- Fraction 6: 3_4_5_6.wpks

This files now can be provided to the link module and the individual overlapping peaks identified. Linking is requested in the same way as Warp2D by passing it through the TAPP 2D's parallel processing list, since it is a CPU intensive operation.

7.2.3 Link

The link module first loads the peak identifier, mz, rt and height of all peaks from each provided file in the correct order. Once all the required information is obtained a k dimensional-tree (KD-tree) algorithm is performed between each pair of fractions. This algorithm was chosen since it has an efficient C implementation in Python which allows for finding the closest neighbors between two sets of data points.

A KD-tree is a data structure used for sorting a finite set of points, first created Jon Bentley in the 1970s. This algorithm splits the data sets into multiple nodes each having only two children, generating a tree of values[20]. Although the KD-tree algorithm is designed to handle multidimensional data, for the purposes of this project only a 2D version is used. Figure 13 shows an example of how a 2D tree is constructed using this algorithm. It is important to note that each level of the tree represents a split in only a single dimension, which alternates with each split.

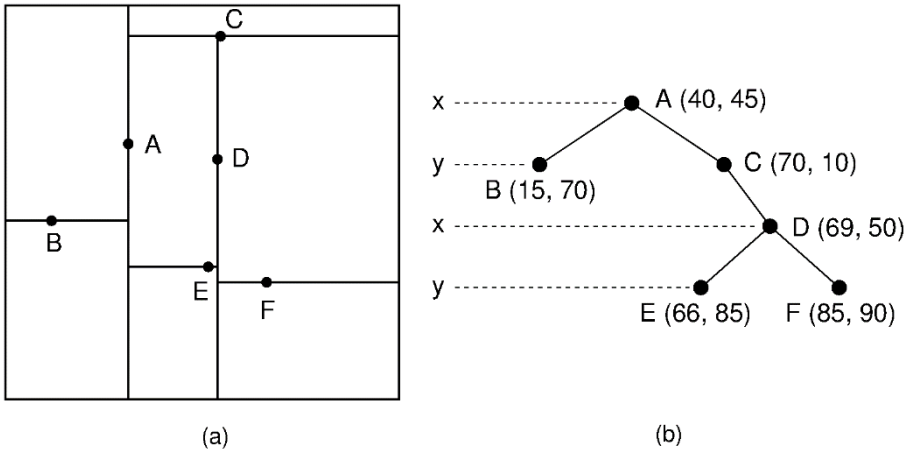


Figure 13 Scheme of KD-tree decomposition of a two dimensional data set containing 7 points [21].

A common operation using a generated kd-tree is referred to as Nearest Neighbor Queries. Given a point with the same dimensions its start going through the tree and at each level only needs to be compared with the value of the split point in the dimension it was split. Eventually the query is placed with one of the blocks defined by the algorithm. Queries have a cutoff distance, which is used to significantly lower the computation time, since only the nodes within this distance are checked as shown in Figure 13 [20].

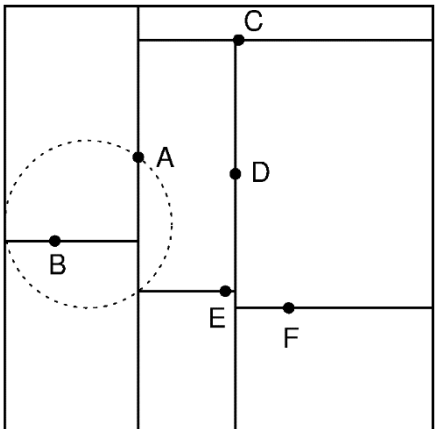


Figure 14 Example of a Query using the generated KD-tree[21].

When using the KD-tree the m/z and the rt are used as the x and y dimensions respectively. Within the Python implementation the Query cutoff distance is defined just by a radius around the point. Since the shifts between peaks in rt are significantly larger than those in m/z, the cutoff distance is defined by the user as two separate variables. The data set is scaled using them and the default cutoff distance used is 1, and in essence the cutoff provided to the query becomes a multiplier.

The Query returns the index of the closest neighbor, within the radius, or if not presents returns a value one higher than the index range, which are filtered out. Using these index pairs, the peaks are further filtered based on their height difference. If one of the peaks is larger than the other by a degree specified by the user, the peak pair is removed from the list. All peaks that have been linked and pass the threshold are placed within a new list that includes: the peak identifier in the first fraction, the peak identifier in the second fraction, and the combined height of the peaks.

When a group has a size of N, this operation results in N-1 link lists. If the group has more than two fractions these lists need to be merged together. With each merger an additional identifier column should be added in the list. Regardless of the size the merger is always performed using the last identifier column of the left list and the first of the right and both lists are ordered based on those identifiers. The lists are made the same size and based on the merging column, the values in the right list are inserted into the left one. When an identifier on the right is the same as that on the left these rows (peaks) are merged together into a single row.

The end result is a list of a column identifier for each fraction in the given group and a final column with the combined peak height. The list is extracted to previously mentioned lpks file, which describes the linked peaks within each fraction group.

7.2.4 File Manager

The file manager module is used by TAPP 2D to store the file locations and names of all files required during the complete pre-processing. During the initialization the file manager stores references to the original peak lists as well as variables like the parameters for Warp2D and required for linking, which parameters are set in the JSON configuration file. Additionally, it creates a work directory for each individual sample where the resulting wpks and lpks files are stored.

This module is also responsible for generating the arguments list for the execution of the for mentioned operations. TAPP 2D uses only the sample and fraction ids, and file manager translates those to specific files, adds the variables from the configuration JSON file and directs the output of the operation to one of the work folders. All this information is returned to TAPP 2D as a string of arguments that can be understood by Bash (on Linux) or CMD (on Windows), with the first argument the full path to the requested executable (also included in the configuration file).

When TAPP 2D requests a Warp2D, it sends a request with two values, the reference fraction index and the sample fraction index. If the reference fraction index has a value larger than 1, File Manager automatically knows that this is referring to an already warped file so it looks for it in the work directory instead of the original pks one.

The advantage of all this is that no other module within TAPP 2D has to work with file paths and details of execution of Warp2D. This concepts leads to the decrease of the amount of memory used and simplifying the reading and debugging of the Python scripts.

7.2.5 Testing of TAPP 2D

Figure 15 shows the resulting grouping between fractions of the example dataset.

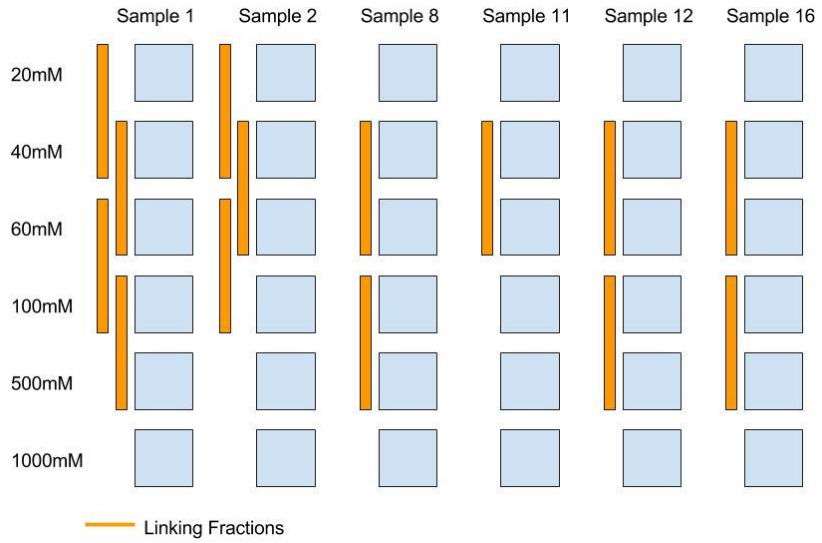


Figure 15 Resulting fraction groups from testing TAPP 2D on 6 samples.

As seen on Figure 15, created fraction groups using the same configuration vary between samples. Middle fractions show more tendency to be grouped compared to the first and last ones. This result can be verified using overlap tables created for each sample. Table 1 and Table 2 show the overlap generated during the processing of samples 1 and 2 respectively.

	1	2	3	4	5	6
1	1	0.4498	0.0325			
2		1	0.2678	0.0108		
3			1	0.2293	0.0734	
4				1	0.2617	0.
5					1	0.0092
6						1

Table 1 Overlap table of sample 1 of COPD data set, obtained using TAPP 2D fraction linking. Green fields have passed the threshold of 0.2, red have not, and orange were generated based on adjacent results.

	1	2	3	4	5	6
1	1	0.5487	0.0573			
2		1	0.4218	0.0152		
3			1	0.3199	0.	
4				1	0.0619	
5					1	0.0243
6						1

Table 2 Overlap table of sample 1 of COPD data set, obtained using TAPP 2D fraction linking. Green fields have passed the threshold of 0.2, red have not, and orange were generated based on adjacent results.

Both tables were processed using only two cycles and a total of 8 Warp2d executions for each sample. An important concept is presented here with respect to decreasing the processing time for the algorithm. The orange fields have been auto generated because the values below them did not pass the threshold. In the first case if fraction 5 does not overlap with 6 than fraction 4 won't overlap with 6

either. Using the same logic if two fractions are tested and do not have sufficient overlap no fraction before them (with respect to the overlap table this would be all fields above) would have an overlap.

From this data the largest overlap was observed between fractions 2 and 3 (40mM and 60mM) of sample 16, so it was selected to verify the performance of the linking method. The original peak list of sample 16 fraction 2 was plotted with the warped peak list of fraction 3 to fraction 2. Each peak is represented using an ellipse using the m/z and rt of the peak as the center point and the peak width in those dimensions as the width of the ellipse.

Additionally, in the same plot data was also added from the MS/MS identifications for the same sample. This data includes points where the mass spectrometer has performed a fragmentation of a peak and the peak was identified using UniProt protein sequence database with PEAKS software.PEAKS is a software that identifies fragment MS/MS spectra using database search approach and provides a list of identified peptides and proteins. This information is useful because it is an alternative method for verifying if two peaks that are indeed caused by the same compound in different fractions from the same sample. The overview of this is presented in Figure 16 and a closer view in Figure 17.

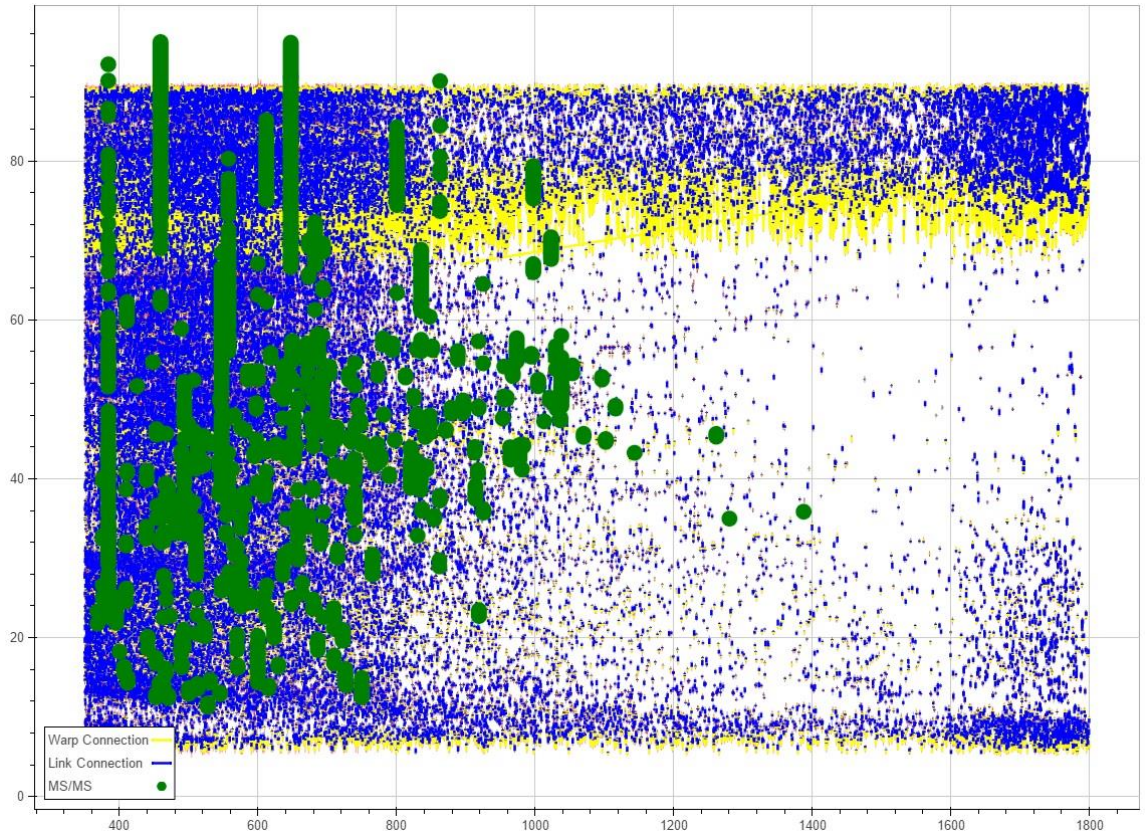


Figure 16 Overview of Sample 16 Fractions 2 (40mM) and 3 (60mM) Peak lists, and MS/MS identified peptides using UniProt protein sequence database. Additionally, the unwrapped peak list of fraction 3 has been added in order to observe the shift corrected by Warp2d.

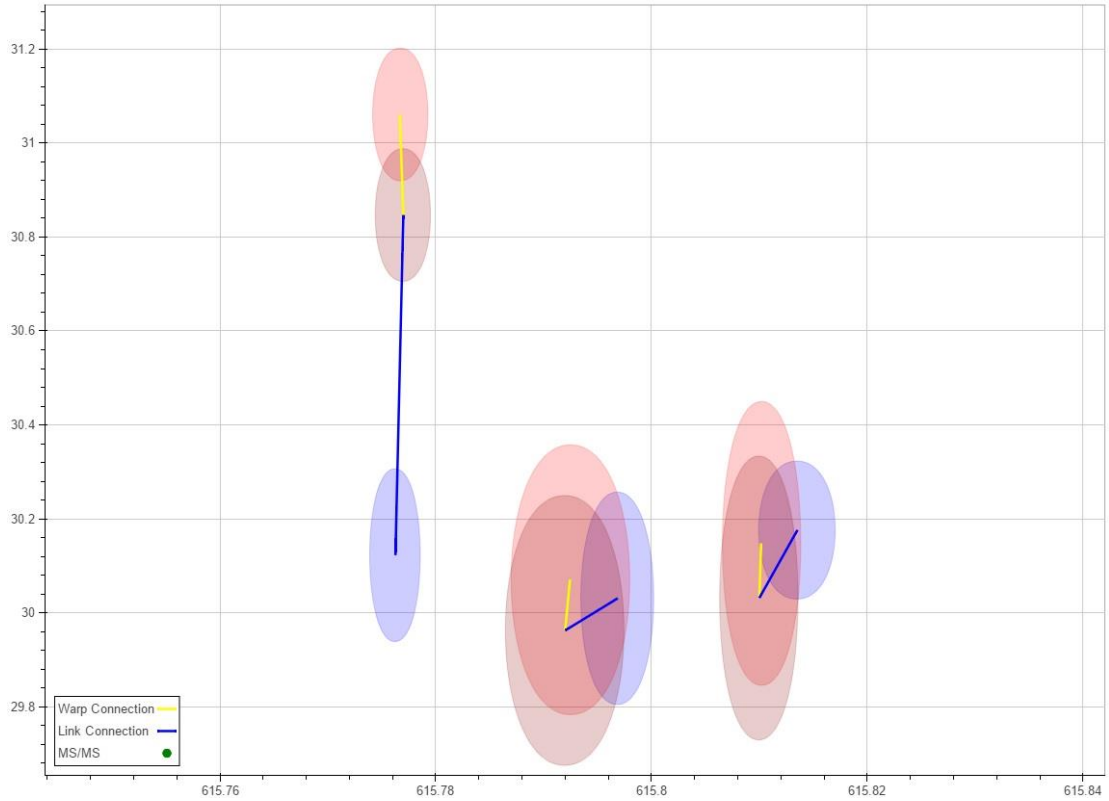


Figure 17 Example of two correct and one incorrect link between fractions 2 (40mM) and 3 (60mM) of sample 16. Blue ellipses are peaks from fraction 2, light red are from fraction 3 and dark red are from fraction 3 warped to fraction 2. Yellow lines show the connection between warped and non-warped fraction 3 (generated by Warp2d) and blue lines show the connection between fraction 2 and warped fraction 3 (generated by the Link Module).

Figure 17 presents a good example of the situations observed by the results of the Link module. The peaks on the left have been linked together despite the two fractions obviously being too far away, even after performing Warp2d. This is considered an incorrect link. The two other groups in the figure, however, are considered correct links. In this case Warp2d has not brought the peaks from the different fractions closer, however as explained earlier the retention time alignment is not performing on per peak basis, but on segments of the entire rt range.

When looking closer at Figure 17 something unexpected can be observed. Behind the linking of peaks between fractions it is assumed that there would be a small drift in the mz dimension, however Warp2D corrects shifts peaks only in rt. Thus all yellow lines should be perfectly vertical, and yet they are not due to mass shifts in high resolution Orbitrap data. This is shown in greater detail in Figure 18, which sheds light on the cause of this error. During the shift the peak was rounded to a precision of 2 digits after the decimal point. This is confirmed by directly viewing the original peak list generated by centroid and the one by Warp2D in a text editor.

- Peak (index, m/z, rt and height) in the pks file: 278705 1233.53598 54.75 15651.44875
- Peak (index, m/z, rt and height) in the wpks file: 278705 1233.54 55.0324 15651.4

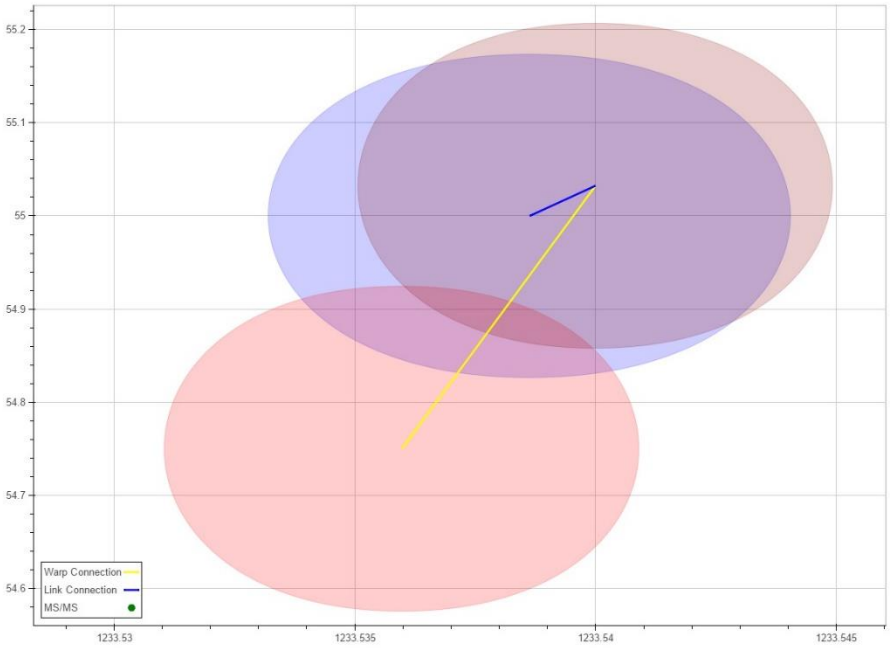


Figure 18 Unexpected shift in the m/z dimension of a peak caused by the low precision of the Warp2D module. In fractions 2 and 3 of sample 16. All shifts of Warp2d should be only in the rt dimension, resulting in perfectly vertical lines.

Overall link module does perform correctly with respect to connecting the two closest peaks in a pair of neighboring fractions. However, the original idea to confirm that at least several links are correct by comparing peptide sequences identified for MS/MS spectra to a single-stage (non-fragmented) peak. Possibly the closets match is shown in Figure 19.

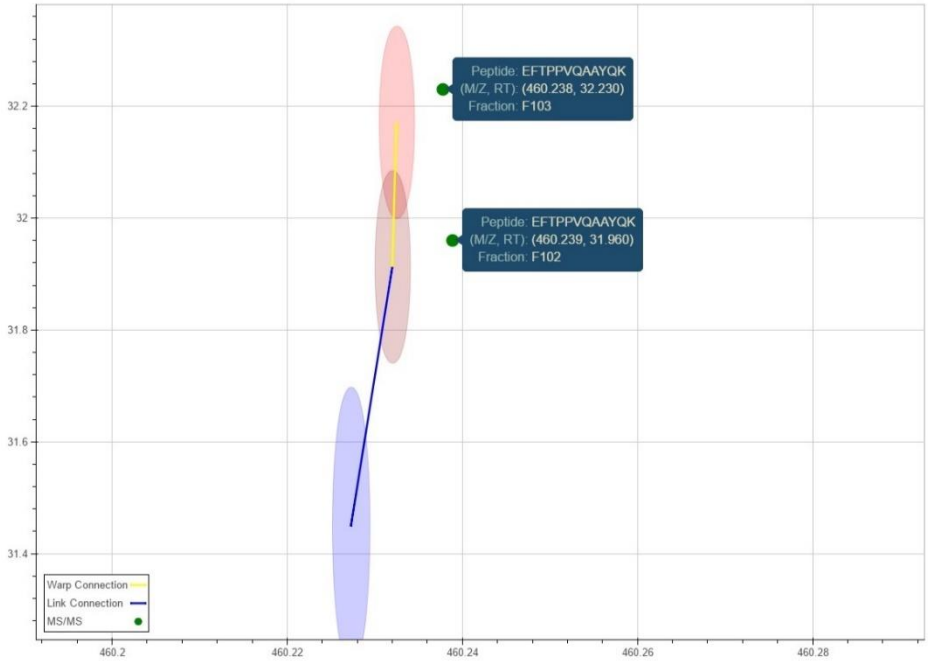


Figure 19 Comparison of peak identification between TAPP 2D and peptide identification using UniProt. UniProt has identified the same peptide in the two neighboring fractions (in peptide data F102 and F103) represented by the green circles. The peptide in fraction 3 (60mM) can be considered a match with the TAPP peak (Light red ellipse), however, the MS/MS identified peptide in the other fraction (lower one) is too far away from the peak in the same fraction extracted using TAPP (Blue ellipse).

Most of the MS/MS identified peptides were observed to have a significantly wider spread in rt, compared to peaks extracted using Grid & Centroid in the same region. Additionally, the occasions where peaks of the two different types of data processing overlap are extremely rare:

- 1 Million peaks originally extracted by centroid (TAPP)
- 68 Thousand links of individual peaks between the two fractions
- 15 Thousand identified MS/MS peaks (For all fractions in the sample)
- 1447 identified MS/MS peaks both present in Fraction 2 and 3 (40mM and 60mM)
- 3 overlaps between the any peaks in the MS/MS and TAPP data sets (Observed)
- No occasion found where MS/MS information confirms TAPP 2D link (Observed)

7.3 Fraction Fusion

7.3.1 Obtaining Connection Information

In order to combine all the Meta Peaks from the separate fractions, all previous steps within the workflow need to have been completed. Although a couple of solutions can be used for performing this task, the more straightforward was selected, because of time restrains.

A Python script loads information of peak identifiers from the fraction pid files and sample lpks files. During the loading process all peaks are saved into a single list with the following information:

- **Connection identifier**, which is the same for peaks that were either grouped into a Meta Peak or linked using TAPP 2D. Furthermore, its incremental between the input file so if a lpks ends with identifier N, the next file will start at N+1.
- **Origin Sample**, identifier for the sample in which the peak belongs.
- **Origin Fraction**, identifier for the fraction in which the peak belongs.
- **Peak Identifier**, the original peak id in its peak list, which is used by both linking of fraction and creating Meta Peaks of samples.
- **Source Identifier**, a value used later on to decide it the entry originated from a lpks or pid file.

Once all the information is loaded the peak entries are converted into a string using the sample, fraction and peak identifiers and single characters describing each field (**S** {sample id} **F** {fraction id} **P** {peak id}, example **S2F4P21301**).

This new list is sorted and is searched for occurrences where there are multiples of the same peak. Since it is possible that two fraction groups share a fraction, it is possible that the same peak has a link record in to different fraction groups. This is where the processing flag comes into play. It allows for these occasions to be treated differently from the mainstream case in which peak is present in a lpks and in a pid files.

Keeping in mind that in the 1D-LC-MS workflow MetaMatch is the final step, which is used to filter out noise, a peak that is not present in a pid file can be excluded. However, a single peak in a pid file may be spread through linking files to multiple fractions, where it is not part of a meta peak. In order to include those situation in the fusion algorithm all information per sample needs to be processed together.

Combining connection information is performed by changing the **connection identifier** of one entry to that of the other entry, which has the same peak. After performing this operation all possible times, the

end result is that each peak generated by the same compound will have a unique identifier which is shared across samples and fractions. Connection entries are grouped by their connection identifier, which at this point is no longer required, resulting in an output in the following form:

Source Identifier	Sample Identifier	Fraction Identifier	Peak Identifier
5	0	1	2
-143983	0	2	6735
-143983	0	3	500084
-435	0	1	765
-85855	0	1	765
5	0	1	765
-1	0	1	2
-435	0	0	825
-85537	0	2	123
8	0	0	825
-85537	0	1	2
-1	0	0	2
-85855	0	2	6735
8	0	0	2

Table 3 Example of connection map generated during 2D-LC-MS(/MS) data fusion.

7.3.2 Resolve Peaks

The result of the previous section is multiple entries for each sample such as the one shown in Table 3. This information needs to be first converted into a set of peaks and then merged into one or more peaks depending on the link information. In this data format the connection entries have positive values for the mpid of the meta peaks created on a per-fraction basis, and negative values represent entries obtained from the link files. This information needs to be resolved down to a set of peaks that are to be merged together.

Using the connection identifiers and the peak identifiers the entry shown in Table 3 generates the connection maps shown in Figure 20 and Figure 21. The connection entry resolves into two maps because peaks 2 and 825 share the same meta peak in fraction 0. They are separated and individually evaluated.

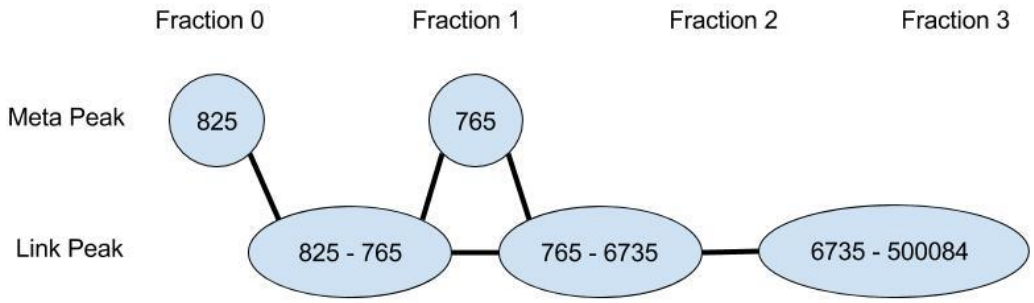


Figure 20 Connection map obtained during the fusion of meta peaks. Figure shows that a meta peak entry with peak identifier (pid) of 825 in fraction 0 is connected to another meta peak, through peak index (pid) 765, in fraction 1. Additionally, it has been connected to peaks in fractions 2 and 3, but those peaks are not present in meta peaks. This map is generated using the entry depicted in Table 3.

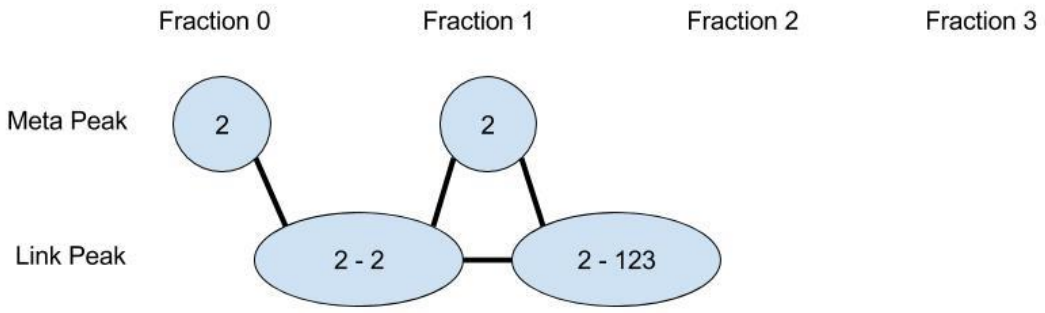


Figure 21 Connection map obtained during the fusion of meta peaks. This shows that peak 2 of fraction 0 is linked to peak 2 of fraction 1, with an addtional peak in fraction 2 that is an orphan peak. Note: The multiple peaks with identifier 2 are seperated based on the fraction they are in. This map is generated using the entry depicted in Table 3.

Once the maps are generated a list of peaks that need to be connected is created. Using the map from Figure 21 the following peaks are extracted:

	m/z	rt	Height
Fraction 0 – Peak 825	690.3494062	33.47787342	31423861.0
Fraction 1 – Peak 765	690.3488531	34.14770596	17012698.08
Fraction 2 – Peak 6735	690.3503636	34.97690614	723055.0682
Fraction 3 – Peak 500084	690.3531131	28.23569507	90111.6538

Table 4 Peaks obtained from a connection map that need to be merged together.

Judging by the m/z values in Table 4 it can be concluded that these peaks are likely caused by the same compound and need to be merged. Furthermore, the height of the peaks decreases with fractions and complies with the assumption of how the LC would split a compound between fractions.

These peaks are merged in the following way:

- The m/z and rt of the resulting peak is calculated using a weighted average of the input peaks using their respective Heights.
- The Height and volume of the resulting peak is the sum of the corresponding parameters of the input peaks.
- The resulting peak's meta peak identifier, peak identifier and fraction identifier become that of the highest input peak that is present in a meta peak from the Meta Match per fraction lists.

When ordered by fraction, if at some point the height of the peaks drops below a certain threshold and then rises again it is assumed that these are no longer a part of the same compound. In this case multiple resulting peaks may be created by splitting the connection map at the minimum. In order, to obtain information about the peak volume and avoid the previously mentioned issues with the drop in precision by Warp2d (and consequentially MetaMatch), the peak information used is directly taken from the original peak lists generated by Centroid. This results in the wide range observed in the rt dimension between peaks, since no warping has been performed to move them closer together. This is not an issue however since the important information is knowing, the height of the same peak in various samples.

The output is extracted to a new pid file (.expid) with each entry including all information required to generate a new meta peak list and corresponding pid file in the final step.

7.3.3 Create Final Meta Peaks

Meta Peak List files contain the following information for each peak described within them:

> *" mpid mz rt NPeaks mzSigma rtSigma Height Volume ExtremeClass ClassH0 ClassH1 FileH0 FileH1 FileH2 FileH3 FileH4 FileH5 " – Header information of the mpks file format with 6 input samples and 2 classes for the example test data set.*

This information needs to be generated by combining the extended pid (expid) files that are generated for each sample. The information is split in this way to allow for each sample to be processed in parallel, since all operation required for fusion of data up to this point involves other fractions of the same sample. Merging the data back together using the mpid values of each entry from all files, each unique mpid is given to a function that recalculated the meta peak (with higher precision).

Should be noted that in this file format "samples" are referred to as "files", since during preparation some samples may be processed in a different manner before passing through the pipeline. With respect to data flow this does not change the any functionality of the algorithm and even the same sample may have multiple entries they are treated as unique samples.

7.3.3.1 Number of peaks and standard deviation of m/z and rt

During the meta peak creation first the number of peaks (NPeaks), the standard deviations of the m/z and rt values (mzSigma and rtSigma) are calculated. This is performed since they are not height or volume dependent.

7.3.3.2 Height per File, m/z and rt

In one sample multiple peaks may be presnt in a slucter that fall within the rt and m/z threshold of MetaMAtch module. Each of the FileH# (Height per file) takes the value of the highest peak part of this meta peak that is present in the corresponding file. This check is performed since it is possible that there are multiple peaks in the same meta peak and same file. The volume of the highest peak is also considered for calculations later on. If a file does not have a corresponding peak its height value is set to 0.

The m/z and rt values of the corresponding heights peaks per file are also taken into a calculation for the meta peak's position. The mean of all highest peaks per file present is taken as the position of the meta peak.

7.3.3.3 Height per Class and ExtremeClass

Once all FileH# values are filled, the Class heights (ClassH#) are calculated, based on the mean of the highest peaks present in files. If a file does not have a peak its excluded from this mean. The ExtremeClass value is an integer identifying which class is higher.

7.3.3.4 Height and Volume

The height and volume of the meta peak are the height and volume of the higher class. All thought the volume per class or file is not present in the final output it is still calculated.

7.3.3.5 New Meta Peak Identifier

The MetaMatch algorithm sorts its entries based on their height (Highest meta peak has an index of 0). Since multiple modifications have been performed on the meta peaks the original index is no longer valid. Once all meta peaks have been generated, they are sorted based on height and given a new mpid corresponding to their new position in the full meta peak list.

7.3.4 Test Results

The above described solution was tested on the extended dataset including 10 samples and 6 fractions described in 6.5 Test Data. The first step in the meta peak fusion algorithm took between 2 and 4 minutes to calculate per sample. For this data set it took roughly 26 minutes to calculate all connections. At the current point time there is no parallel processing implemented, however due to the design of the algorithm it is possible to perform all steps up to the meta peak creation on a per-sample basis on separate CPU processes.

The meta peak creation algorithm took a total of 2 615 226 merged peaks which were grouped into 448 834 meta peaks. Calculation of these meta peaks took roughly 2 hours, which is possible to decrease. Since each individual meta peak is calculated on its own using previously calculated set of peaks, the meta peak calculation can be processed in parallel without any maximum core restrictions.

An example of the archived output is shown in Table 5 and Table 6. Both these tables correspond to a single meta peak entry. The total output includes this time of information for each of the meta peaks stated above.

Mpid	**m/z**	**rt**	**NPeaks**	
6	383.89685769	20.68568200	7	
mzSigma	**rtSigma**	**Height**	**Volume**	**ExtremeClass**
0.00040303	0.82220957	1.63947219e+09	3.65900372e+10	0
	ClassH0	**ClassH0**	**FileH0**	**FileH1**
	1.63947219e+09	9.35465742e+08	0.00000000e+00	7.68453099e+07
FileH2	**FileH3**	**FileH4**	**FileH5**	**FileH6**
2.12774263e+09	2.71382862e+09	0.00000000e+00	4.38919549e+07	1.34629413e+09
FileH7	**FileH8**	**FileH9**		
1.65135573e+09	7.00321154e+08	0.00000000e+00		

Table 5 Single line entry of final mpks file, output of the 2D-LC-MS algorithm.

Mpid	**m/z**	**Rt**	**Height**	**Volume**	**FileID**	**FractID**	**PeakID**
6	383.89592750	22.50527939	7.68453099e+07	1.40587765e+09	1	3	65
6	383.89714200	20.12047465	2.12774263e+09	4.52538682e+10	2	3	0
6	383.89715730	19.72689181	2.71382862e+09	6.31103658e+10	3	3	0
6	383.89674550	20.36524231	4.38919549e+07	8.14987776e+08	5	3	22
6	383.89695310	20.91371366	1.34629413e+09	3.03074056e+10	6	3	0
6	383.89712160	20.52406961	1.65135573e+09	3.27990785e+10	7	3	2
6	383.89695680	20.64410255	7.00321154e+08	1.35248044e+10	8	3	0

Table 6 Entries for meta peak 6 of the final pid file, output of the 2D-LC-MS algorithm.

A larger partial output of the mpks and pid files can be found in Appendix B.

7.4 PIPELINING

In order to encapsulate all the individuals steps of the 2D-LC-MS algorithm into a user friendly shell, all modules need to be executed using only the input files described in the overview of the 2D-LC-MS workflow in Figure 11. No work has been performed towards this goal yet, however, three possible solutions have been theorized:

- Expanding the currently used Perl workflow processing 1D-LC-MS data to include newly developed operations
- Creating a new Python workflow that utilizes the modularity of the existing Perl workflow to perform the yellow blocks described in Figure 11, and run the rest of the operations by itself.
- Reworking the currently used Perl workflow for 1D-LC-MS into Python and augmenting it with the new functionally for 2D-LC-MS

Unfortunately, due to the unexpected complexity in implementing the fusion between meta peaks, work on merging the created modules into a pipeline is left as post project work.

8 DISCUSSION

8.1 PARALLEL PROCESSING

At this point in both the original 1D-LC-MS workflow and the TAPP 2D module the execution of parallel processing is sub-optimal. The reason for this is that in both first a list of operations required is generated and afterwards parallel processing is implemented within that list. Depending on the number of operations in the list and the number of processors available, performance may vary drastically.

An example of this is if 9 Warp2D operations are required and 8 processors are available. On the computational cluster provided a single Warp2D operation takes around 280 seconds to complete. 8 of the 9 operations will be completed around the same time, but afterwards the algorithm will require additional 280 seconds before moving on with the next operation, effectively doubling the required time. If this is the final step in the complete work flow or all next steps required all previous steps to be completed is necessary.

On the other hand, it was discovered that during the operation of both 1D-LC-MS and TAPP 2D CPU intensive operations of different stages can operate at the same time, since many do not depend on each other's output. TAPP 2D's fraction groups have been designed to operate independently and the operations from the 1D-LC-MS Perl workflow also have this possibility.

Furthermore, TAPP 2D and the 1D-LC-MS Perl workflow handle parallel processing independent of each other. This means that if the user wishes to use 8 cores and both TAPP 2D and the 1D-LC-MS workflow run at the same time, the full algorithm will try to use 16 cores. There are workarounds this problem however implementation of a single "task manager" will undoubtedly prove beneficial to the overall usage of the available hardware. The reason for both implementations using this "static" operation list is, because of the complexity in programing introduced when multiple processes need to share information between each other.

8.2 DIFFERENCE IN PRECISION

Although the 1D-LC-MS work flow has been used for a long time, during this project it was discovered that the centroid and warp2d modules output results in a significantly lower precision compared to that of the mass spectrometer used. The reason for this is that the work flow was first developed for older lower resolution mass spectrometer, for which the number of digits after the decimal point was sufficient.

However, with the newer MS instrument centroid was extracting based on the higher precision and was writing to file using the lower one. This situation resulted in pks files where multiple peaks were in the same position in m/z and rt due to rounding, but with different heights and widths. A filter for these duplicates was implemented in the link module of TAPP 2D, which resulted in 80% of the peaks being discarded.

During the project an updated version of centroid was provided, which included height precision, but the Warp2D module exhibited the same issue (at this point an updated version has not been provided). This partially crippled the testing of the TAPP 2D module and slowed down the development processes of the 2D-LC-MS algorithm.

8.3 INCORRECT LINKING OF PEAKS BETWEEN FRACTIONS

With the lower precision of some files and higher precision of others tests were still performed on the data set in order to assess the logic behind linking of peaks between fractions. Overall the results are considered satisfying because the KD-tree algorithm does indeed link together peaks that were brought closer together by Warp2D.

The errors observed are the result of using the same cutoff distance as Warp2D for the KD-tree query, and finding a different optimal value is a matter of testing. Since all the configuration can be changed with the use of the JSON file once the 2D-LC-MS workflow is implemented testing the effect of the cutoff distance can be performed with ease. Furthermore, since there won't be a need to generate new warped peak lists every time, execution time of each test will only be the result of the link module.

8.4 MISMATCH BETWEEN MS/MS AND LINKING INFORMATION

During the situational and theoretical analysis is was theorized that the MS/MS information, although not as extensive as the peak lists, can be used as a base truth when verifying the functionality of the 2D-LC-MS algorithm. The results showed hardly any overlaps between the data sets, and a large variation between the two was observed due to the lower precision of m/z in the precursor ions. Additionally a compound may be spread across a wide range of retention time with the same m/z in the LC-MS/MS data. The centroid peak extraction, however, has shown a tendency of splitting peaks so wide into multiple peaks with lower width.

Solving this can be performed by a better representation of the data sets, by including intensities for the peaks of the identified peptides. This will require more processing power from the visualization platform, and even with the results presented for the TAPP 2D testing the time required was substantial (20-30 seconds for performing a box zoom or moving the visual range). With the low amount of points in the MS/MS data per fractions it was not desired to limit the range of all data sets to decrease the visualization time.

Another approach is to again finish the 2D-LC-MS workflow, process the data with the peptide identification using the full mpks file, and directly compare peptides. Finding the same peptides in the different fractions, even if they are not found exactly next to each other, would prove that linking between fractions is correct.

Annotating peak lists with peptide identification also allows to confirm the results using the chemical properties of a known peptide. Does it make sense to a bioinformatician for this peptide to be present in particular location, with respect to first LC, second LC, and MS?

8.5 PIPELINING THE SOLUTION

As mentioned earlier, the solution was not finalizing into a complete workflow at this point in time. However, since most of the pre-requisites have been completed it is expected that modifying the exiting workflow or creating a new one would be a straightforward process of passing information between the individual algorithm segments.

It would be best if the mentioned dynamic parallel processing is controlled by the outer layer of the workflow. Additionally, expanding the file manager of TAPP 2D to handle files for the MetaMatch

direction of data processing, is considered beneficial to the complete solution, since it will create a more uniform indexing system.

With respect to the three mentioned solution, the option of encapsulating partial Perl workflows in Python is by far the worst of the options. This is because Python will need generate multiple JSON files for each different sub-workflow, or multiple times modify the same one. Both this options will make the operations of the workflow less clear to the user.

The end results of the full Perl or full Python workflows is going to be the same. The question between the two only comes down to their implementations. A potential difference may lay in the fact that Python scripts are capable of being compiled using C++ (Cython). This operation increases the speed of execution of the script, but it will no longer be easily modified. Logic within the workflow itself is not overly complex so this potential improvement will be marginal.

8.6 Overall 2D-LC-MS algorithm performance

The final results obtained by performing the fusion of meta peaks between fractions with the COPD data set, confirm the successful base operation of the algorithm. The newly created Meta Peaks have been augmented with peaks, caused in theory by the same compound, from other than their original fraction. In the case observed in Table 4 the main peak has 63% of it in one fraction and the rest split up in three others. Considering the amount of information contained within each sample this difference may very well be crucial in the discovery of biomarkers.

With the current implementation a meta peak can technically gain a member peak of another meta peak. This can result in meta peaks present in the final result that would not normally have been passed by MetaMatch, since they are no longer present in enough samples. Furthermore, some meta peaks in different fractions may be caused by the same compound, but the algorithm will merge their peak heights only for member peaks that have links. At the current state of the development it is not known how those situations need to be handled. Finding a solution to those situations requires further investigation.

Using only the output of the 2D-LC-MS algorithm (the complete mpks and pid files), a very limited number of conclusions can be made. Using multiple entries in the final meta peak list like the one in Table 5, it is possible to compare the meta peak quality based on the standard deviation of the member peaks. The mzSigma observed in both cases are within the same range, however in the final output shows generally larger deviations in the rt dimension. This is caused by the fact that before MetaMatch is performed all input peak lists are warped to a single one. The new meta peak list uses original meta peak information. This was decided because the files for MetaMatch are warped to a single reference chromatogram that provides the best results. The selection of the reference chromatogram is made by warping every chromatogram to every other chromatogram and assessing the mean of the sum of overlapping peak volume ratio. When using samples and fractions the operations required expand exponentially. Additionally, warping between fractions would further complicate the procedure, because of the need for chain warping files.

9 CONCLUSIONS

The results of the newly developed 2D-LC-MS algorithm confirm that it successfully combines data from neighboring fractions based on proximity of individual peaks. The distance between peaks in the rt dimension of each fraction is successfully decreased by Warp2D, which has been confirmed beforehand to improve the quality of results when comparing peak lists. Connecting peaks between fractions in each sample is performed using all peaks regardless of height, so low abundance peak information is preserved. Currently the augmentation from 1D-LC-MS to 2D-LC-MS requires only three additional variables with respect to linking peaks. The first one defines the minimum of mean of the sum of overlapping peak volume between fractions to be considered as part of one fraction group. The second is the minimum distance between peaks in two neighboring fractions in rt and m/z and the third is the minimum height ratio between peaks split in different fractions of the same peak. The optimal values for these variables may alternate between data sets, so their effect on the final result needs to be explored in greater detail.

Once all peaks within each sample are linked based on proximity this information is used to merge the output of 1D-LC-MS work flows performed on the same fraction of each sample. The result of these steps is two sets of files that described connected peaks between fractions, and connected peaks between samples. Merging this information is performed by a data fusion module, which does not require any additional configuration in its current version. New output format matches the output of the 1D-LC-MS, thus techniques that are normally implemented after TAPP can still be used without the need of any modifications to them.

Only limited assessment of the full 2D-LC-MS algorithms result was possible at this stage, since the main focus of this project the development of the TAPP workflow to process 2D-LC-MS/MS. The newly developed pipeline would successfully combine information on the same compound, if the peak fragmentation base assumptions made regarding the changes caused by the first LC dimension are correct. The modifications implemented allowing TAPP to use 2D-LC-MS data, do not hinder its ability of detecting low abundance peaks, and use all components of the original pipeline.

The new version of TAPP has been designed to be able to run multiple operations in parallel, allowing for efficient use of computational hardware. The only limits in the number of processes running at the same time are the number of fractions or samples in the input data set. The modularity of the algorithm allows for individual sections of the solution to be upgraded in the future. Intermediate results generated by each module provide means for tracking peak information from the end result to the original input data.

In conclusion TAPP has been successfully augmented to handle the additional information obtained by the use of an additional LC dimension, compared to its previous version. The new solution is still usable for the detection of low abundance compounds and run multiple operations in parallel like its predecessor. Although the solution has not been validated using alternative data sets, it has been confirmed that it follows the outlined conceptual model. Thanks to the use of a single configuration file shared by all modules of the algorithm, a future developer can fine-tune the algorithm's performance to the data set they are using.

10 RECOMMENDATIONS

The 2D-LC-MS algorithm was only tested on half the samples of the COPD data set, with one set of configuration parameters. Once the solution is pipelined, expanding the analysis and assessment to the complete COPD dataset and testing with alternative datasets should be a straightforward process. Since all configuration parameters are used only within the linking of peaks between fractions, testing that module on its own would be the first priority for finding optimal values.

As stated the developed solution does successfully follow the outlined conceptual module. In order to confirm the base assumptions of peak fragmentation caused by LC, it is best that the solution's results are compared with those of alternative methods like labeled peptide and MS/MS identification. In order for these results to be directly comparable, the peptide identification of 1D-LC-MS data needs to be implemented with the results of the 2D-LC-MS algorithm.

With respect to the structure of the full TAPP pipeline, several possible improvements can be implemented. First would be the implementation of a unified file management system across all components. Within the workflow, multiple times the scripts needed to pass long paths to files resulting in hard to interpret file names. This becomes an issue in situations like chain warping, where each Warp2D generates file names with increasing length to the describe the output file. The solution to this is using the original fine names only during the first step of the algorithm, and further on referring to data files using sample and fraction identifiers, and file extensions.

The second point of improvement is implementing a shared dynamic parallel processing for all modules. At multiple stages of the algorithm operations are capable of running in parallel and some may finish before the rest. The current system is only capable of handling a predefined set of operations and this creates processing bottlenecks. Implementing dynamic parallel processing, would allow to use the same processing lists, but operations return results independently and new operations can be added to a queued processing list at any point in time. This can drastically decrease computational time for the full pipeline especially when a large number of input files are used.

Finally, the newly developed algorithm does not handle all possible situations that may occur with respect to the same peaks in different fractions. This includes the occasions mentioned earlier where a meta peak is no longer present in enough samples to pass MetaMatch's clustering threshold. Another situation is when a peak from the same compound in meta peaks may be present multiple times in different fractions. Additionally, situations where the height of a linked peaks does not follow the normal distribution expected in LC-MS/MS data. These situations may occur for only a few peaks, however considering the potential importance of all data, a solution needs to be implemented. Removal and merger of meta peaks can be based on a new variable regarding the number of links between the meta peaks. Decomposition of a peak can be based around the expected Gaussian distribution of LC, but will require parameters set based on information currently not used, which is the distance between the fractions.

11 References

[1] R. Mayeux, “Biomarkers: potential uses and limitations.,” *NeuroRx*, vol. 1, no. 2, pp. 182–8, 2004.

[2] J. K. Nicholson and J. C. Lindon, “Systems biology: Metabonomics,” *Nature*, vol. 455, no. 7216, pp. 1054–1056, 2008.

[3] P. L. Horvatovich and R. Bischoff, “Current technological challenges in biomarker discovery and validation,” *Eur J Mass Spectrom (Chichester, Eng)*, vol. 16, no. 1, pp. 101–121, 2010.

[4] F. Suits, B. Hoekman, T. Rosenling, R. Bischoff, and P. Horvatovich, “Threshold-avoiding proteomics pipeline,” *Anal. Chem.*, vol. 83, no. 20, pp. 7786–7794, 2011.

[5] P. W. Carr, D. R. Stoll, and X. Wang, “Liquid Chromatography,” *Anal. Chem.*, vol. 83, no. 6, pp. 1890–1900, 2011.

[6] M. J. Edelmann, “Strong cation exchange chromatography in analysis of posttranslational modifications: Innovations and perspectives,” *Journal of Biomedicine and Biotechnology*, vol. 2011. 2011.

[7] P. Horvatovich, B. Hoekman, N. Govorukhina, and R. Bischoff, “Multidimensional chromatography coupled to mass spectrometry in analysing complex proteomics samples,” *J. Sep. Sci.*, vol. 33, no. 10, pp. 1421–1437, 2010.

[8] A. O. Nier, “A mass spectrometer for isotope and gas analysis,” *Rev. Sci. Instrum.*, vol. 18, no. 6, pp. 398–411, 1947.

[9] F. Suits, J. Lepre, P. Du, R. Bischoff, and P. Horvatovich, “Two-dimensional method for time aligning liquid chromatography-mass spectrometry data,” *Anal. Chem.*, vol. 80, no. 9, pp. 3095–3104, 2008.

[10] N. P. V Nielsen, J. M. Carstensen, and J. Smedsgaard, “Aligning of single and multiple wavelength chromatographic profiles for chemometric data analysis using correlation optimised warping,” *J. Chromatogr. A*, vol. 805, no. 1–2, pp. 17–35, 1998.

[11] K. Gerbrands and P. Horvatovich, “Proteogenomics Data Integration - Thesis for Bachelor Degree,” NHL Hogeschool, 2015.

[12] P. K. Van Der Vlugt, “C Style Guide and Programming Guidelines,” *Style (DeKalb, IL)*, 2003.

[13] J. Dongarra, I. Foster, G. Fox, W. Gropp, K. Kennedy, L. Torczon, and A. White, *Sourcebook of Parallel Computing*. 2003.

[14] N. Llopis, “Data-Oriented Design (Or Why You Might Be Shooting Yourself in The Foot With OOP),” *gamesfromwithin.com*, 2009. [Online]. Available: http://gamesfromwithin.com/DATA-ORIENTED-DESIGN. [Accessed: 19-Mar-2016].

[15] J. Cox and M. Mann, “MaxQuant enables high peptide identification rates, individualized p.p.b.-range mass accuracies and proteome-wide protein quantification.,” *Nat. Biotechnol.*, vol. 26, no. 12, pp. 1367–72, 2008.

[16] J. Cox, M. Y. Hein, C. a Luber, and I. Paron, “Accurate proteome-wide label-free quantification by delayed normalization and maximal peptide ratio extraction, termed MaxLFQ,” *Mol. Cell. …*, vol.

13, no. 9, pp. 2513–2526, 2014.

[17] J. S. Cottrell, “Protein identification using MS/MS data,” *Journal of Proteomics*, vol. 74, no. 10. pp. 1842–1851, 2011.

[18] H. Zhou, W. Li, S.-P. Wang, V. Mendoza, R. Rosa, J. Hubert, K. Herath, T. McLaughlin, R. J. Rohm, M. E. Lassman, K. K. Wong, D. G. Johns, S. F. Previs, B. K. Hubbard, and T. P. Roddy, “Quantifying apoprotein synthesis in rodents: coupling LC-MS/MS analyses with the administration of labeled water.,” *J. Lipid Res.*, vol. 53, no. 6, pp. 1223–31, 2012.

[19] P. L. Horvatovich, “Sample preparation of frozen tissue samples. Digestion with trypsin - Materials and Methods - Lab Manual,” Groningen, 2015.

[20] A. W. Moore, “Efficient Memory-based Learning for Robot Control - Dissertation,” 1990.

[21] OpenDSA, “KD Trees,” 2013. [Online]. Available: http://algoviz.org/OpenDSA/Books/Everything/html/KDtree.html#. [Accessed: 10-May-2016].

Appendix A

Finite State Machine of Fraction Group Creation

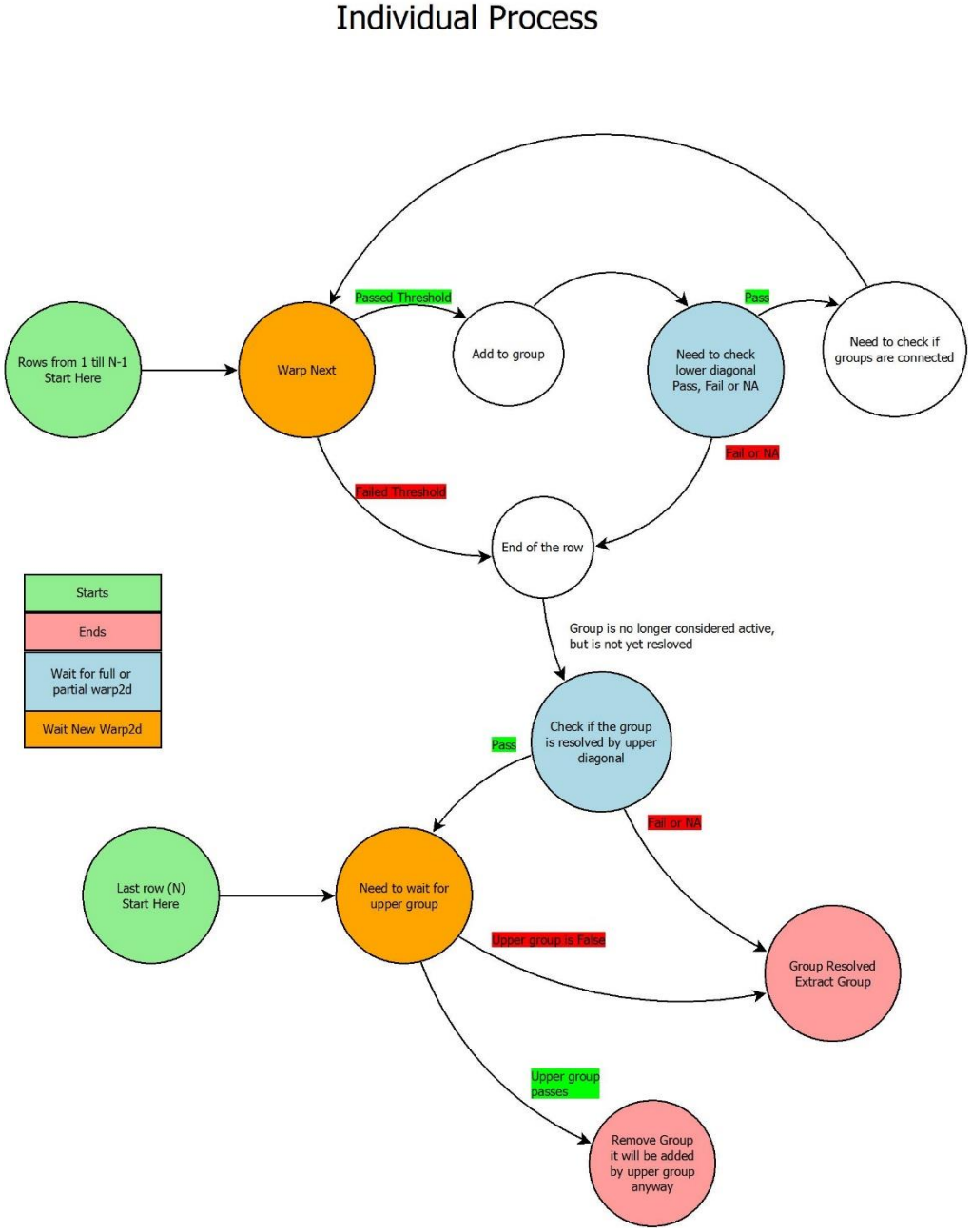


Figure 22 Finite Step Machine of the resolution of a fraction group, during TAPP 2D processing.

Appendix B

2D-LC-MS Algorithm final output example

MPID	MZ	RT	HEIGHT	VOLUME	FILEID	FRACTID	PEAKID
0	657.8373977	34.8288033	9.05E+08	3.84E+10	0	1	1
0	657.8371642	32.35122724	9.66E+08	2.88E+10	1	1	0
0	657.8376819	30.37842759	3.12E+09	9.36E+10	2	1	0
0	657.837655	30.33793874	4.55E+09	1.45E+11	3	1	0
0	657.8376678	32.38892645	3.58E+09	1.34E+11	4	1	0
0	657.8375141	30.7323201	2.11E+09	6.44E+10	5	1	0
0	657.8375088	31.22362266	2.58E+09	7.90E+10	6	1	0
0	657.8378286	30.89699138	3.75E+09	1.24E+11	7	1	0
0	657.8374883	31.10662661	1.28E+09	3.61E+10	8	1	0
0	657.837299	32.53773268	2.12E+09	9.30E+10	9	1	0
1	637.8655651	50.330596	3.17E+06	8.77E+07	1	2	2029
1	637.8682252	47.3808962	1.76E+09	5.34E+10	2	2	0
1	637.8679762	47.18359492	2.57E+09	7.92E+10	3	2	0
1	637.8677604	49.62030628	2.61E+09	9.87E+10	4	2	0
1	637.8678724	48.23545268	2.33E+09	7.08E+10	6	2	1
1	637.8679522	47.51390192	4.27E+09	1.53E+11	7	2	1
1	637.8681915	47.81221901	8.66E+08	2.41E+10	8	2	3
2	657.8459101	30.37241795	1.60E+06	1.18E+07	2	2	12058
2	657.8411317	29.91976558	4.62E+05	2.08E+07	3	2	33462
2	657.8395238	31.92154034	2.53E+06	1.62E+07	4	2	3092
2	657.8373813	30.8576926	3.78E+08	1.08E+10	5	2	0
2	657.8374091	30.96199439	2.80E+09	8.63E+10	6	2	0
2	657.8375495	30.42853259	4.70E+09	1.60E+11	7	2	0
2	657.8373538	30.66312905	1.87E+09	5.66E+10	8	2	0
3	470.7318664	20.62981803	1.16E+06	3.82E+07	0	1	19802
3	470.7269908	16.70423659	4.23E+08	9.25E+09	2	1	39
3	470.7269655	16.77183192	3.54E+08	7.77E+09	3	1	30
3	470.7276282	17.94195792	2.09E+07	4.74E+08	4	2	318
3	470.7255722	17.41697405	1.24E+09	2.75E+10	6	1	4
3	470.725761	17.13503709	2.98E+09	7.08E+10	7	1	1
4	557.3022932	48.20522089	8.49E+04	2.76E+06	1	3	473592
4	557.3020901	47.85495751	3.23E+06	1.53E+08	3	3	4092
4	557.3030634	47.42894744	1.96E+09	6.04E+10	4	3	0
4	557.3042272	45.37654536	2.10E+09	5.73E+10	7	3	0

Table 7 Partial output of 2D-LC-MS algorithm peak id list (pid) file for 10 samples split into 2 classes.

MPID	MZ	RT	NPEAKS	MZSIGMA	RTSIGMA	HEIGHT	VOLUME	EXTREMECLASS
0	657.8375205	31.67826167	10	0.0001879	1.3094738	2.63E+09	8.79E+10	0
1	637.867649	48.29670957	7	0.0008644	1.12234524	2.49E+09	8.28E+10	1
2	657.8394656	30.73215321	7	0.00295237	0.58181511	2.44E+09	7.85E+10	1
3	470.727464	17.7666426	6	0.00209602	1.3458614	2.11E+09	4.91E+10	1
4	557.3029185	47.2164178	4	0.00083829	1.09724245	2.10E+09	5.73E+10	1
5	638.3657879	49.10300559	5	0.0014976	2.27444139	1.87E+09	6.49E+10	0
6	690.3543911	30.95477648	6	0.00380357	1.25644666	1.76E+09	5.67E+10	0
7	658.3359331	31.50990114	7	0.00093671	0.64917637	1.55E+09	5.28E+10	1
8	689.8549994	31.17231062	9	0.00025979	0.96983132	1.55E+09	4.71E+10	0
9	658.3359867	32.19812362	8	0.0008995	2.7812644	1.51E+09	4.73E+10	1
10	395.2388143	20.91796077	8	0.00165925	1.11471686	1.48E+09	2.87E+10	0
11	464.248424	29.77371023	6	0.00052315	0.73057547	1.43E+09	3.12E+10	0
12	466.762098	36.18192075	9	0.00039888	2.75171965	1.40E+09	3.48E+10	1
13	639.8642483	46.8315461	5	0.00061375	3.51866842	1.36E+09	4.05E+10	1
14	536.2818154	45.76193778	6	0.00046937	3.53294101	1.34E+09	3.50E+10	1
15	706.328517	42.03911612	3	0.00068005	0.37588859	1.34E+09	4.39E+10	0
16	510.5838587	21.53901409	8	0.00074131	1.12343082	1.33E+09	3.41E+10	1
17	544.27857	41.25514073	4	0.000666	1.71982921	1.29E+09	3.43E+10	1
18	383.8968093	20.77619448	8	0.00039814	0.80552627	1.23E+09	2.75E+10	0
19	690.352996	31.10194982	6	0.00288713	1.31809701	1.21E+09	3.68E+10	1
20	395.240993	20.18068521	6	0.00238063	0.70037159	1.15E+09	2.18E+10	1
21	689.8544228	31.60370851	9	0.00079827	2.76341638	1.12E+09	3.32E+10	1
22	686.2874131	17.56043516	4	0.00020373	0.82017044	1.11E+09	3.29E+10	0
23	510.9157298	21.55434474	8	0.00086954	1.15013117	1.07E+09	2.77E+10	1
24	575.3399176	20.17685517	3	0.00016401	0.52790787	1.06E+09	3.32E+10	1
25	575.3115672	32.16555567	9	0.00120455	1.6471361	1.03E+09	2.98E+10	1
26	376.195239	24.62346189	7	0.00285786	1.2468038	1.02E+09	2.42E+10	1
27	658.8358798	31.66260842	8	0.00103656	1.42037947	9.84E+08	3.34E+10	1
28	749.7939699	12.01304858	8	0.00257184	1.08054021	9.78E+08	3.25E+10	1
29	409.7244234	22.91429853	6	0.00063652	0.83044758	9.60E+08	2.02E+10	1
30	575.8107448	32.0046416	9	0.00066473	1.15914976	9.56E+08	2.70E+10	0
31	557.6331609	47.14641672	6	0.0019031	0.85472463	9.42E+08	2.57E+10	1
32	383.8967973	20.04656392	8	0.00039378	0.71039008	9.21E+08	2.08E+10	1
33	467.2610021	23.63368084	8	0.00170105	1.0557511	9.19E+08	2.00E+10	0
34	557.9667157	47.08693145	5	0.00185335	0.9275956	9.05E+08	2.59E+10	1
35	547.3191938	32.8492201	8	0.00075438	2.98366066	9.03E+08	2.28E+10	0

Table 8 Partial output of 2D-LC-MS algorithm meta peak list (mpks) file for 10 samples split into 2 classes. (Part 1)

ClassH0	ClassH1	FileH0	FileH1	FileH2	FileH3	FileH4	FileH5	FileH6	FileH7	FileH8	FileH9
2.63E+09	2.37E+09	9.05E+08	9.66E+08	3.12E+09	4.55E+09	3.58E+09	2.11E+09	2.58E+09	3.75E+09	1.28E+09	2.12E+09
1.74E+09	2.49E+09	0.00E+00	3.17E+06	1.76E+09	2.57E+09	2.61E+09	0.00E+00	2.33E+09	4.27E+09	8.66E+08	0.00E+00
1.53E+06	2.44E+09	0.00E+00	0.00E+00	1.60E+06	4.62E+05	2.53E+06	3.78E+08	2.80E+09	4.70E+09	1.87E+09	0.00E+00
2.00E+08	2.11E+09	1.16E+06	0.00E+00	4.23E+08	3.54E+08	2.09E+07	0.00E+00	1.24E+09	2.98E+09	0.00E+00	0.00E+00
6.54E+08	2.10E+09	0.00E+00	8.49E+04	0.00E+00	3.23E+06	1.96E+09	0.00E+00	0.00E+00	2.10E+09	0.00E+00	0.00E+00
1.87E+09	4.55E+07	0.00E+00	0.00E+00	0.00E+00	1.86E+09	1.88E+09	0.00E+00	0.00E+00	9.96E+07	4.32E+05	3.63E+07
1.76E+09	5.78E+08	1.36E+09	0.00E+00	0.00E+00	2.15E+09	0.00E+00	3.93E+05	2.27E+09	1.01E+06	0.00E+00	4.56E+07
1.18E+09	1.55E+09	0.00E+00	9.93E+05	0.00E+00	0.00E+00	2.35E+09	1.18E+09	1.67E+09	2.46E+09	7.93E+08	1.65E+09
1.55E+09	1.33E+09	1.09E+09	1.65E+09	1.60E+09	0.00E+00	1.84E+09	7.48E+08	1.51E+09	1.90E+09	9.03E+08	1.57E+09
8.73E+08	1.51E+09	1.24E+05	3.42E+07	0.00E+00	2.79E+09	6.66E+08	2.19E+08	1.77E+09	2.89E+09	1.14E+09	0.00E+00
1.48E+09	1.84E+08	1.43E+09	8.43E+08	2.17E+09	0.00E+00	0.00E+00	1.45E+05	2.56E+08	1.43E+08	5.19E+08	7.20E+05
1.43E+09	2.41E+08	0.00E+00	0.00E+00	2.55E+09	1.06E+09	6.77E+08	0.00E+00	5.52E+08	2.27E+05	1.72E+08	0.00E+00
1.20E+09	1.40E+09	8.43E+08	4.31E+08	1.42E+09	2.12E+09	0.00E+00	2.52E+08	1.68E+09	3.64E+09	6.74E+08	7.42E+08
1.06E+08	1.36E+09	2.17E+05	0.00E+00	1.50E+06	0.00E+00	3.16E+08	0.00E+00	1.19E+09	1.53E+09	0.00E+00	0.00E+00
8.86E+08	1.34E+09	8.80E+06	0.00E+00	1.30E+09	1.35E+09	0.00E+00	0.00E+00	1.09E+09	2.90E+09	0.00E+00	3.21E+07
1.34E+09	3.57E+08	0.00E+00	0.00E+00	0.00E+00	1.34E+09	0.00E+00	0.00E+00	6.16E+08	9.86E+07	0.00E+00	0.00E+00
1.29E+09	1.33E+09	0.00E+00	4.51E+04	1.35E+09	1.87E+09	1.93E+09	6.95E+08	1.49E+09	2.57E+09	0.00E+00	5.59E+08
5.89E+06	1.29E+09	0.00E+00	0.00E+00	0.00E+00	8.52E+06	3.25E+06	4.01E+05	0.00E+00	2.57E+09	0.00E+00	0.00E+00
1.23E+09	9.35E+08	0.00E+00	7.68E+07	2.13E+09	2.71E+09	6.02E+06	4.39E+07	1.35E+09	1.65E+09	7.00E+08	0.00E+00
1.04E+09	1.21E+09	0.00E+00	1.30E+09	1.58E+05	0.00E+00	1.82E+09	0.00E+00	0.00E+00	2.18E+09	1.45E+09	1.05E+07
8.20E+08	1.15E+09	0.00E+00	3.69E+05	4.43E+08	1.97E+09	8.69E+08	1.60E+08	0.00E+00	2.15E+09	0.00E+00	0.00E+00
8.49E+08	1.12E+09	1.99E+05	0.00E+00	1.70E+05	2.83E+09	5.66E+08	2.53E+08	1.47E+09	2.87E+09	9.85E+08	1.48E+06
1.11E+09	5.55E+08	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.11E+09	0.00E+00	0.00E+00	1.15E+09	3.29E+08	1.83E+08
1.01E+09	1.07E+09	0.00E+00	3.88E+04	1.05E+09	1.38E+09	1.62E+09	5.61E+08	1.19E+09	2.07E+09	0.00E+00	4.48E+08
7.48E+08	1.06E+09	0.00E+00	0.00E+00	7.48E+08	0.00E+00	0.00E+00	0.00E+00	2.78E+08	1.85E+09	0.00E+00	0.00E+00
7.00E+08	1.03E+09	1.68E+06	0.00E+00	2.75E+09	4.66E+07	2.61E+06	2.44E+08	1.53E+09	2.53E+09	8.93E+05	8.46E+08
2.82E+08	1.02E+09	0.00E+00	0.00E+00	2.11E+08	6.27E+08	9.31E+06	1.17E+07	1.43E+09	2.19E+09	0.00E+00	4.52E+08
6.82E+08	9.84E+08	2.32E+08	0.00E+00	6.58E+08	9.30E+08	9.07E+08	0.00E+00	5.92E+08	2.02E+09	6.97E+08	6.28E+08
1.59E+08	9.78E+08	7.68E+07	0.00E+00	2.96E+08	3.44E+06	2.61E+08	7.26E+05	1.09E+09	2.19E+09	6.36E+08	0.00E+00
5.50E+08	9.60E+08	0.00E+00	1.16E+05	0.00E+00	4.78E+08	1.17E+09	0.00E+00	5.82E+07	2.82E+09	6.38E+05	0.00E+00
9.56E+08	1.83E+08	0.00E+00	4.13E+08	1.53E+09	1.27E+09	6.05E+08	1.66E+08	1.55E+08	3.27E+06	7.52E+07	5.16E+08
4.42E+08	9.42E+08	0.00E+00	6.96E+04	1.40E+07	1.65E+07	1.74E+09	0.00E+00	0.00E+00	1.88E+09	1.92E+06	0.00E+00
1.71E+08	9.21E+08	0.00E+00	0.00E+00	4.05E+08	1.07E+08	1.09E+06	2.31E+08	1.44E+09	2.93E+09	3.56E+06	2.20E+05
9.19E+08	2.57E+08	0.00E+00	4.61E+08	1.19E+09	8.75E+08	1.15E+09	2.13E+08	2.40E+05	0.00E+00	2.71E+06	8.11E+08
2.25E+08	9.05E+08	0.00E+00	5.52E+04	1.29E+07	1.41E+07	8.72E+08	0.00E+00	0.00E+00	9.05E+08	0.00E+00	0.00E+00
9.03E+08	2.96E+08	0.00E+00	9.93E+07	1.65E+09	1.15E+09	7.19E+08	6.02E+06	5.96E+08	0.00E+00	3.72E+08	2.08E+08

Table 9 Partial output of 2D-LC-MS algorithm meta peak list (mpks) file for 10 samples split into 2 classes. (Part 2)