

FRAGARIYO v2.1

# Contents

[Overview 2](#_bookmark0)

[References 2](#_bookmark1)

1. [Choose where to save results 3](#_bookmark2)
2. [PreRun Items 3](#_bookmark3)
3. [Match terminal fragments first 5](#_bookmark4)

[Terminal Fragment Parameter File 6](#_bookmark5)

[Terminal Fragment Results 7](#_bookmark6)

1. [Internal Fragmentation 8](#_bookmark7)
2. [Input Generator 8](#_bookmark8)

[Input ready for Internal Fragment matching: 8](#_bookmark9)

1. [Running Internal Fragment Search 8](#_bookmark10)
   1. [Mass Resolution 8](#_bookmark11)
   2. [Tolerance Error 8](#_bookmark12)
   3. [Internal Run 8](#_bookmark13)

[Internal Fragment Results 11](#_bookmark14)

1. [Data Analysis 13](#_bookmark15)

[Sequence Coverage Calculations 13](#_bookmark16)

[ClipsMS Plotter 13](#_bookmark17)

[Internal Fragments Outputs 13](#_bookmark18)

# Overview

Post-translational modifications are important to understand the connection between protein structure and function. Mass spectrometry (MS)-based proteomics allowed for the high-throughput sequencing and quantitation for proteins in complex samples.1 The typical bottom-up (BU) workflow requires digestion of proteins into peptides, which are analytes amenable to analysis by the instrumentation available in the early stages of MS. While BU proteomics is well established technique, the required digestion step precludes identification of all the protein forms (proteoforms)2 that exist in the sample of interest. In top- down (TD) proteomics3, intact proteins are analyzed by LC-MS/MS optimized for large ions. Analyzing intact ions allows for the detection of proteoforms, however in typical TD proteomics denaturing conditions are used. Therefore, if there are structural differences between proteoforms that information is lost under denaturing conditions. Protein stoichiometry and sequence information can both be obtained from proteoforms of interest when using native TD (nTD) proteomics.4 While improved instrumentation has been improving the accuracy and resolution of nTD data, software for the analysis of these datasets is lagging behind (plus a lack of standardization in file types and workflows to use for analysis).

Fragariyo is written in Python 3.7 using PyCharm IDE. It is a set of scripts capable of database search and multi-pass matching for terminals and internal fragments (taking into account modifications and disulfide bonds). Currently, Fragariyo accepts IMTBX\Grppr .isotopes files (or Mobilatron), and CSV files (m/z, charge, and intensity) Let’s hunt some fragments with Fragariyo!

## References

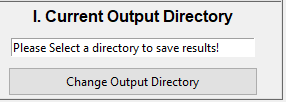
* 1. Aebersold, R.; Mann, M. Mass Spectrometry-Based Proteomics. *Nature* **2003**, *422*, 198–207.
  2. Aebersold, R.; Agar, J. N.; Amster, I. J.; Baker, M. S.; Bertozzi, C. R.; Boja, E. S.; Costello, C. E.; Cravatt, B. F.; Fenselau, C.; Garcia, B. A.; Ge, Y.; Gunawardena, J.; Hendrickson, R. C.; Hergenrother, P. J.; Huber, C. G.; Ivanov, A. R.; Jensen, O. N.; Jewett, M. C.; Kelleher, N. L.; Kiessling, L. L.; Krogan, N. J.; Larsen, M. R.; Loo, J. A.; Ogorzalek Loo, R. R.; Lundberg, E.; Maccoss, M. J.; Mallick, P.; Mootha, V. K.; Mrksich, M.; Muir, T. W.; Patrie, S. M.; Pesavento, J. J.; Pitteri, S. J.; Rodriguez, H.; Saghatelian, A.; Sandoval, W.; Schlüter, H.; Sechi, S.; Slavoff, S. A.; Smith, L. M.; Snyder, M. P.; Thomas, P. M.; Uhlén, M.; Van Eyk, J. E.; Vidal, M.; Walt, D. R.; White, F. M.; Williams, E. R.; Wohlschlager, T.; Wysocki, V. H.; Yates,

N. A.; Young, N. L.; Zhang, B. How Many Human Proteoforms Are There? *Nat Chem Biol* **2018**, *14*, 206–214.

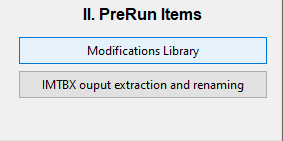
* 1. Toby, T. K.; Fornelli, L.; Kelleher, N. L. Progress in Top-Down Proteomics and the Analysis of Proteoforms. *Annual Review of Analytical Chemistry* **2016**, *9*.
  2. Zhou, M.; Lantz, C.; Brown, K. A.; Ge, Y.; Paša-Tolić, L.; Loo, J. A.; Lermyte, F. Higher-Order Structural Characterisation of Native Proteins and Complexes by Top-down Mass Spectrometry. *Chem Sci* **2020**, *11*, 12918–12936.

# Choose where to save results.

Simply press the “Change Output Directory button”. It can be changed at anytime except while Fragariyo is running processes.

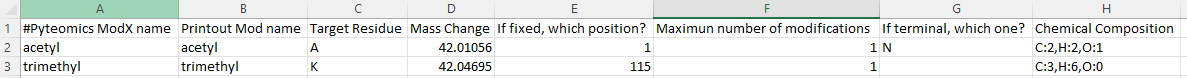


# PreRun Items

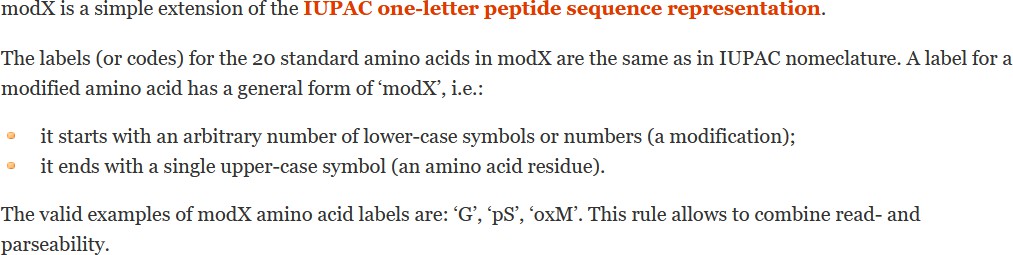


* 1. Modifications Library. Once clicked, user can upload a .txt file containing modification information (see ModificationsRepository.txt for a template).

ModificationsRepository.txt



* + 1. Pyteomics ModX name: String with the modification name compliance with ModX
       1. ModX = <https://pyteomics.readthedocs.io/en/latest/api/parser.html>



* + 1. Printout Mod name: String with the modification name the user would like the results to print out as.
    2. Target Residue (*Multiple residues support incoming*)
    3. Mass Change (decimal number that can be either positive or negative).
    4. If modifications is fixed, which position
    5. Maximum number of Modifications per peptide
    6. If modifications is a terminal one, which terminus to place it at?
    7. Chemical Composition for scoring (only implemented in internal fragments searches, *terminal fragment support incoming)*

Element#1[colon]number of atoms of Element#1[comma] Element#2[colon]number of atoms of Element#2[comma]…etc…

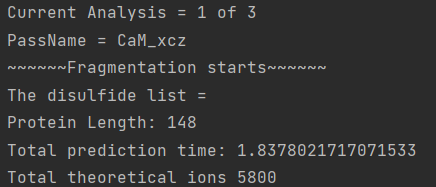
DON’T ADD SPACES!

* 1. IMTBX Output Extraction and Renaming

Script to rename files output from IMTBX/Grppr. It takes .isotopes files from their original .raw file and renames them according to the .raw folder.

# Match terminal fragments first

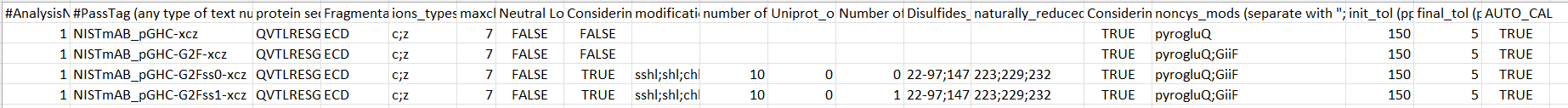
1. Choose Peaklist Files (File types acceptable are):
   1. .isotopes files from IMTBX/Grppr for Waters and from Mobilitron for Agilent files
   2. mMass files (<http://www.mmass.org/)>
   3. csv files (mass-to-charge ratio, charge, intensity with headers m/z,z,int or sample name)
   4. unmatched files (produced from running terminal fragments. If ions are not matched, the unmatched ones will be placed in that file)
2. Upload .ions (a previously produced theoretical ion database) files when prompted to, if a theoretical database has been created.
3. Load the Parameter file (see TerminalFragments\_template.csv to learn how to fill a parameter file).
4. Program will run (progress can be followed thru the pop-up terminal window).



*Terminal Matching starts. Fragariyo GUI will freeze, but progress can be followed in the terminal window running with Fragariyo. Fragmentation will start. The analysis number will appear as well as the current pass (PassName). An analysis is a group of passes.*

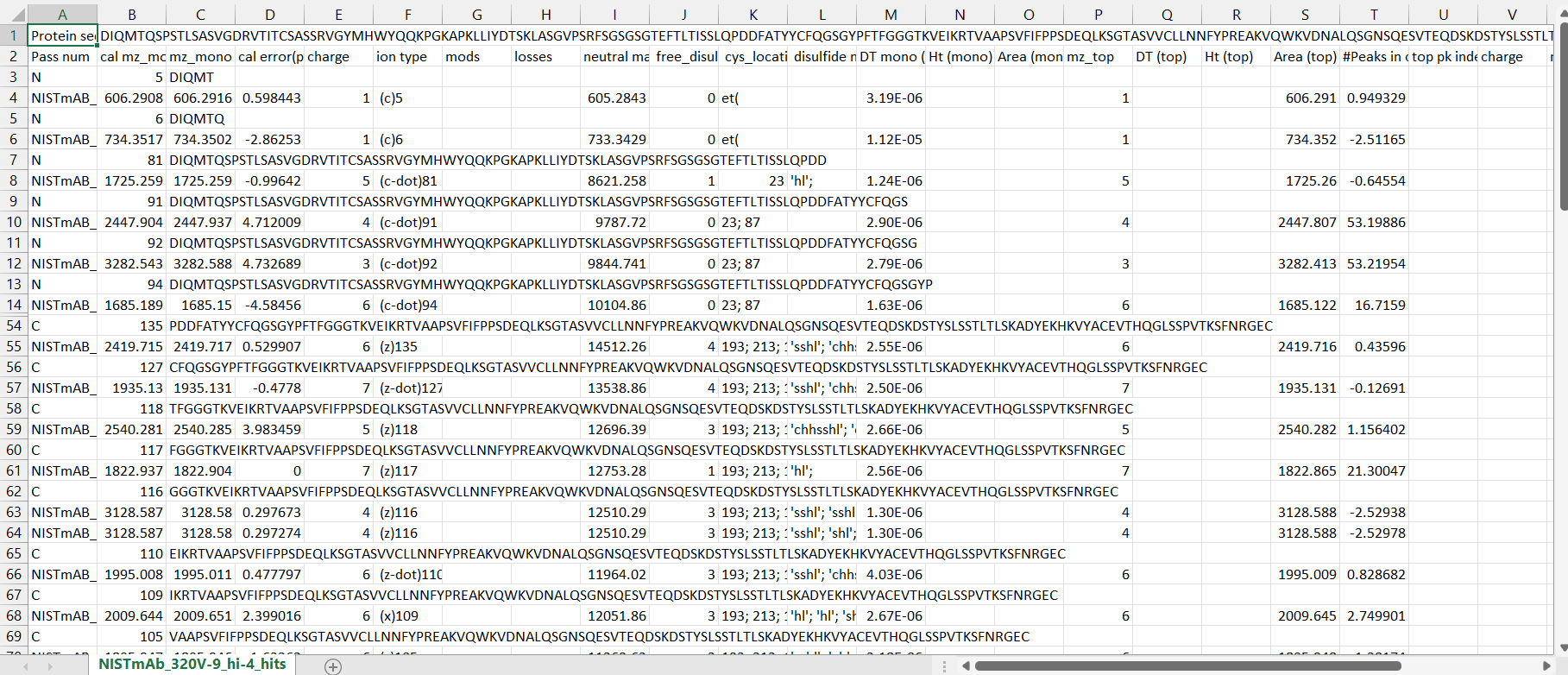
## Terminal Fragment Parameter File

DON’T USE SPACES WHEN INPUTING ITEMS SEPARATED BY SEMICOLONS “;”



1. Analysis number
2. Name for the pass (it can be something informative about the pass e.g. ion type, mutant, etc)
3. One-letter-code protein sequence
4. Fragmentation type, it selects the correct c and z ion types to use. Currently CID and ECD are allowed. If another type is needed, fell free to request it.
5. Ion types (do you need both c and z ions or only one?)
6. Maximum charge state to be used
7. Neutral losses bool. TRUE will considered neutral losses.
8. Disulfide bond bool. TRUE makes the script to determine the amount of disulfide bonds and where they are and to separate them if the two cysteine partners end up in different fragments
9. What modifications to use for the reduced cysteines
   1. sshl;shl;chhsshl;hl;sh;h
10. What is the maximum number of modifications to be used for the cysteine modifications (maximum number of cysteines that can be reduced)
11. Uniprot\_offset, int. Number that the sequence index must be offset to match the disulfide bond positions , if these positions were obtain from the uniport database. Put “0” is no offset needed
12. Number of disulfide bonds to be broken even if their cysteines are in the same fragment
13. Disulfide bond list
    1. Cys1\_index-Cys1partner\_index; Cys2\_index-Cys2partner\_index (e.g NISTmAb disulfide bonds: 22-97;147-203;223-1000;229- 1000;232-1000;264-324;370-428).
14. Index for cysteines that do not participate in any disulfide bonds.
15. Considering modifications (not involved in disulfide bonds)
16. Modifications list (separate modifications with semi colons (e.g. NISTmAb modification for the n-term and glycan: pyrogluQ;GiiF)
    1. See Modifications.py section for more information
17. Initial error tolerance
18. Final error tolerance
19. Auto Calibration bool.If TRUE, Auto calibration will be performed using

## Terminal Fragment Results



The results are organized by termini and fragment site. The results are saved as.csv files and in binary .hits files (created with Pickle package).

# Internal Fragmentation

# Input Generator

Internal fragment matches are scored by comparing experimental with theoretical isotope envelopes. Therefore, experimental isotope information is extracted from files containing the xy coordinates of the spectrum of interest. First a .csv file with experimental ions (mass-to-charge ratio, charge, intensity with headers m/z,z,int) is requested. Next Fragariyo will ask for the xy coordinates (currently only accepting .csv files from Agilent and xy files from Breuker. (select filetype in the lower right part of the pop-up window).

.cvs format example:

A screenshot of a table

Description automatically generated

For IM-MS data obtained in Waters instrument use IMTBX-Grppr (https://pubs.acs.org/doi/10.1021/acs.analchem.7b04999) to produced .isotopes files and use the following columns in the .csv format:

.isotopes files > .csv

mz\_mono > m/z

charge > z(charge)

ab\_top\_total > int (intensity )

### Table Description automatically generatedInput ready for Internal Fragment matching:

## \_interexpion.csv file

* 1. Neutral mass of experimental ion obtained from charge (z) and m/z values
  2. Charge
  3. Mass-to-charge ratio
  4. Intensity
  5. Values *m/z* corresponding to the peak isotope envelope
  6. Values *intensity* corresponding to the peak isotope envelope

# Running Internal Fragment Search

## Mass Resolution:

Approximate the mass resolution of you mass spectra (used to generate theoretical isotope envelopes)

## Tolerance Error:

What error to use (for internal fragments is recommended to internally calibrate data first to obtain the most accurate results). Typical, error used for internal fragments = 1 – 2- ppm.

## Internal Run:

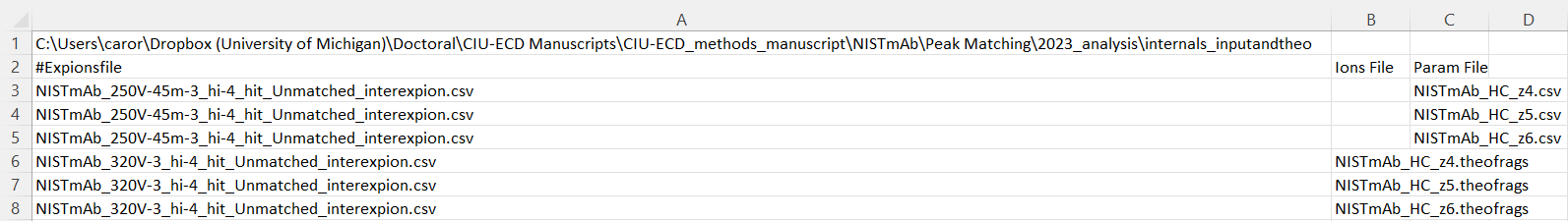
Once you click run, the program will request a batch file that contains the locations of other files. In order to set the internal run three files will be used: \_interexpion.csv (product of IV.I Input Generator), a parameter file to create the database (or a .theofrags file if the database has been created and saved previously), and the batch file. Below you will find a description of each file and what their columns mean. DON’T USE SPACES WHEN INPUTING ITEMS SEPARATED BY SEMICOLONS “;”

## A screenshot of a computer Description automatically generated parameter file (for internal fragments, not the one use for terminals):

1. Analysis number
2. Pass name (something descriptive about the pass (e.g ions searched for, mods searched for, etc)
3. Sequence = Protein sequence one-letter code
4. Fragment chemistry (it selects internal fragments allowed based on the chemistry selected). Currently CID and ECD are allowed. If another type is needed, fell free to request it.
5. Maximum charge to be considered (right now max and min are the same to keep the passes somehow smaller. In a future update hopefully this will be simplified).
6. Minimum fragment length in amount of residues
7. Maximun fragment length in amount of residues
8. Modifications for non-cysteiens residues separate by semicolons (examples shoes an analysis with several passes where a different NISTmAb glycan is searched for). LEAVE EMPTY IF NOT LOOKING FOR MODS!
9. What modifications to use for the reduced cysteines
10. What is the maximum number of modifications to be used for the cysteine modifications (maximum number of cysteines that can be reduced)

## Batch file:

1. Disulfide bond bool. TRUE makes the script to determine the amount of disulfide bonds and where they are and to separate them if the two cysteine partners end up in different fragments
2. Uniprot\_offset, int. Number that the sequence index must be offset to match the disulfide bond positions , if these positions were obtain from the uniport database. Put “0” is no offset needed
3. Number of disulfide bonds to be broken even if their cysteines are in the same fragment
4. Disulfide bond list
   1. Cys1\_index-Cys1partner\_index; Cys2\_index-Cys2partner\_index (e.g NISTmAb disulfide bonds: 22- 97;147-203;223-1000;229-1000;232-1000;264-324;370-428).
5. Index for cysteines that do not participate in any disulfide bonds.

You will be prompted to upload a batch file to run several internal fragment searches automatically or to just run one search (it helps to keep track if the right database file got run with the right ions).

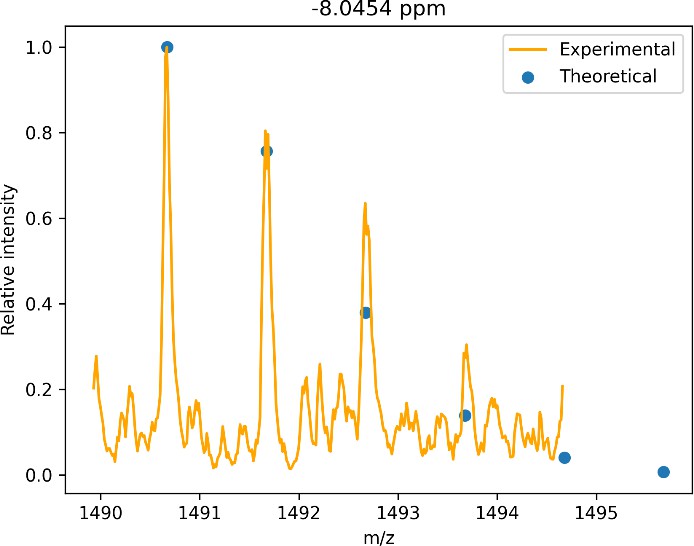
* A1: Path to the folder where all the files o be used are found.
* Row 2: Headers
* Column A: Place \_interexpion.csv file name, including extension as seen in image above.
* Column B: Name of the .theofrags file (if exists or it it was created to a previous search in the same batch).
* Column C: Name of the Parameter file to be used for the \_interexpion.csv file in the same row.

Script will run ( a protein a large as BSA took several days and breaking the analysis down to several pieces)

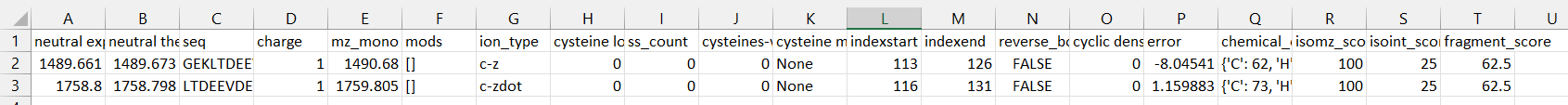
## Internal Fragment Results

The results will be saved in a folder created with the name of the experimental ion .csv file: The folder contains:

1. A .theofrags file (a.ka. .ions file) a binary version of the theoretical database
2. A.txt.frags. A human readable version of the theoretical databased (they can be quite large). To be removed.
3. PNG files showing the matching of the theoretical isotopic envelope and experimental isotopic envelope



1. TSV files (tab-delimited files) containing the details of the matched peaks

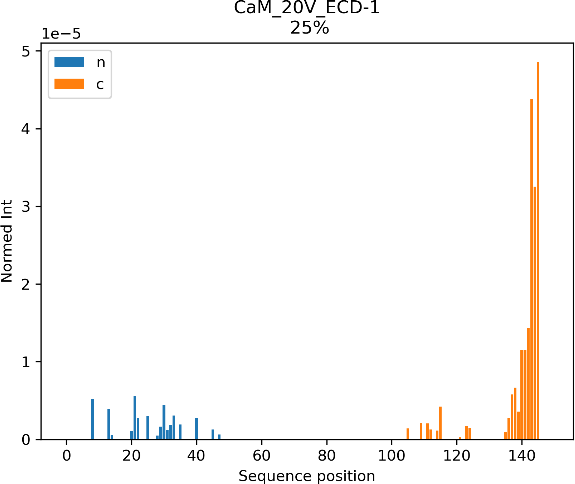


* 1. Neutral mass (experimental ion)
  2. Neutral mass (theoretical ion)
  3. Fragment sequence
  4. Charge
  5. Monoisotopic *m/z* value
  6. Modifications
  7. Ion type
  8. Indices where the cysteines are located
  9. Number of disulfide bonds in internal fragment
  10. How many cysteines are being modified
  11. Cystine modification if disulfide bonds were reduced
  12. Index where the fragment starts
  13. Index where the fragment ends
  14. Bool. If TRUE the sequence is a reverse sequence. To be used as FDR (False Discovery Rate) calculations in the future.
  15. Cyclic density (number of cyclic regions form by disulfide bonds/ total residue number)
  16. Matching error
  17. Chemical compositions (in python dictionary form = {'C': 44, 'H': 72, 'N': 14, 'O': 18, 'S': 0, 'Fe': 0})
  18. Scoring in *m/z*
  19. Scoring in *intensity*
  20. Compound score

# Data Analysis

They are divided into terminal and internal sections

## Sequence Coverage Calculations

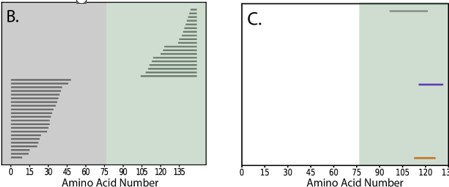


* + Outputs example: A graph showing sequence coverage (coordinate information to replicate this graph is saved as a file: samplename\_analysis.csv) It can be done per file or for a combination of two files.

## ClipsMS Plotter

Based on ClipsMS (Software from the Loo lab) graphing code. It will transform terminal fragment input to be compatible with ClipsMS plotter. There is a terminal and an internal fragments version.

* + - Output:B. Terminal Fragments C. Internal Fragments (color has been added using Illustrator.



## Internal Fragments Outputs

In the Internal fragments analysis tools there is the option of merging the output into one file. The other option is to “average” them: Only retain fragments that appear across all files.