Tutorial 1: Library Generation from DDA data

1. Introduction

Before a targeted, peptide-centric DIA analysis can be performed, a spectral library containing peptide-query parameters needs to be generated. Such a spectral library can be created either from DDA runs of the same or related samples, from the DIA runs directly (e.g. by using DIA-Umpire, Tutorial 3), or you can download an existing spectral library from a public repository such as SWATH Atlas (www.swathatlas.org).

The most common procedure to date is the generation of a spectral library from parallel DDA runs and this is what we will also do in today's tutorial. To ensure maximal compatibility between the spectral library and the DIA runs, it is beneficial to obtain DDA data from the same MS instrument and on the same LC column as the DIA data. It is important that both the DDA and the DIA runs contain the same RT reference peptides for iRT alignment between runs. Our samples contain the widely used iRT standard peptides from Biognosys (https://biognosys.com/).

Spectral library and peptide-query parameter generation consists of several steps:

1) **DDA data acquisition**: Since we will further analyze DIA data from hybrid proteome samples of three species (E.coli, human and yeast), one DDA run was performed for both hybrid proteome samples (A and B) on a QqTOF instrument (6600 TripleTOF, AB Sciex) and QqOrbitrap instrument (Q-Exactive Plus).
   Note! In order to get a deeper spectral library, more DDA runs should be used. It is further advisable to inject the proteomes of different species separately and search them against organism specific protein databases for real-world applications. However, for the sake of simplicity and run time, we will only use one DDA run per hybrid proteome sample and search them against a combined proteome database for the library generation in today's tutorial.
2) **Data conversion and peak picking**: In order to save time in this tutorial we will not convert the raw DDA files into mzXML format. This is a routinely performed process and the information on this conversion can be found in the Tutorial Overview section. In this tutorial, we will start with the mzXML files that are the product of conversion and centroiding the raw DDA data. Additionally the data was trimmed by applying an absolute intensity threshold of 2 on the centroided data (for more detailed description, see Appendix 2 of the tutorial overview). This filtering leads to a much smaller file size, and as a consequence the software tools described in this tutorial will run faster. If the library generation workflow is performed using a powerful computing infrastructure, this filtering step can be omitted.
3) **Sequence database search**: We will use the Comet search engine for database search of the acquired DDA data. Here, the first step is to define a FASTA protein sequence database to search against. In our case this we will use a combined FASTA file containing sequences for *E.coli*, human and yeast, plus the iRT reference peptide sequences. Importantly, the FASTA database contains forward and reverse sequences, while the reverse sequences serve as decoys. After the database search, the results are scored with peptideProphet and combined using iProphet.
   Note: We only use one search engine (Comet) here. In cases where a very deep library is required, multiple database search engine results can be combined.
4) **FDR control**: The next crucial step is FDR control of the combined search results. The results from several search engines when combined increase the number of protein and peptide identifications with a high confidence level. We will do this using MAYU at an FDR cutoff of 1% on the protein level.
5) **Combination of search results to a spectral library**: SpectraST is used for spectral library generation. First, a redundant spectral library from all combined search results

is generated with aligned retention times across runs. Second, redundant entries in the spectral library are merged to generate a consensus spectrum.

6) **Spectral library with peptide-query parameters**: The spectral library is finally filtered to contain only selected transitions for each peptide query and decoy peptide-query parameters are generated. The file is in a last step converted to a DIA analysis compatible format (e.g. pqp for OpenSWATH [older versions used TraML or csv]). This then results in a final spectral library ready for use with OpenSWATH.

2. Preparing your working environment

Setup of an analysis bash script

In some of the tutorials, we are working with software tools that we execute directly from the command line (terminal). It is good practice for such analyses to keep a record of the software tools that you used sequentially and specifically the parameter options you selected. This functions like an electronic "lab-book" for computational analyses that is extremely helpful in case you need to be repeated some steps later on or if you want to adapt the commands and parameters for the analysis of a different dataset.

To make the scripting and execution of different commands very easy, text editors with embedded terminals are very convenient. Here we will use RStudio (Version 1.1.383) with an embedded R console (later used for Tutorial 5) as well as an embedded terminal (used in Tutorials 1,3,4 & 6).

- Open RStudio on your VM.
- Open the file "`run_DDA_LibraryGeneration.sh`" in the DIA_Course/Tutorial1_Library directory.
  Note: The "`.sh`" specifies that you are about to write a bash script, meaning a script with commands that can be directly interpreted by the terminal.
- What you see now is a scaffold bash script for the library generation tutorial. **Note:** lines starting with a "#" are comments that will not be interpreted by the computer. Using the "#" you can document your code to make it readable and for you and others to easily repeat analyses in the future.
- You can execute different lines in the script by pressing the "Run" button on the upper write corner of your script panel. The line where your cursor is placed will be executed. If you press the "Run_script" button, all lines in your script will be sequentially executed on the Terminal (bottom left panel).

- Place your cursor on the 2nd line and press "Run".

Now we have changed the working directory to the tutorial folder.

3. Sequence database search

Spectral libraries are built from fragment ion spectra that are assigned with high confidence to a peptide sequence (peptide spectrum matches, PSMs). To establish this match, centroided fragment ion spectra are subjected to sequence database searching. At this stage, it is important that the protein sequence database (typically in FASTA format) contains the sequences of the retention time reference peptides (iRT peptides in our case) to allow for retention time normalization at a later step. To control the FDR of the PSMs, the protein sequence database also needs to contain a decoy entry for every protein. Even though protein sequence reversal is the most commonly used method to generate decoy peptides, decoys most precisely reflecting target peptides are generated by pseudo-reversal of target peptide sequences (in a protein sequence the tryptic peptide sequences are reversed but keeping the last amino acid intact (K or R)). This pseudo-reversal method is used in this tutorial. To

maximize the number of PSMs and the discrimination between true and false assignments, the search output of multiple search engines may be combined. In general, we recommend using search engines that are maximally orthogonal in their search algorithms, as this results in highest numbers of identifications. The optimal parameters for search engines, such as the number of tolerated tryptic termini, missed cleavages, precursor mass tolerance and variable modifications depend on the specific biological sample, experimental setup and purpose of the library. In this protocol, searches were done for fully tryptic peptides. The PSMs from each search engine are scored using PeptideProphet and subsequently combined and rescored using iProphet, a tool that integrates evidence from multiple identifications of the same peptide across different experiments and search engines and thus improves the discriminating power between correctly and incorrectly assigned PSMs.
Like in a usual DDA workflow, we will perform a sequence database search to identify all possible peptides in our samples. For reasons of simplicity and run time, we will only use one search engine, Comet, to perform the sequence database search. Comet is a single command line binary that is freely available for download at http://comet-ms.sourceforge.net and is also bundled with Trans Proteomic Pipeline.

Now, the first step is to perform a Comet database search. To do this we first have to generate a parameter file for Comet and adjust the parameters. You can also learn more about the search parameters at http://comet-ms.sourceforge.net/parameters.

- To open and adjust the comet parameter file go to “File” → “Open File” and open the “`comet.params.high-high_TTOF`” or “`comet.params.high-high_QE`” from the directory `C://DIA_Course/Tutorial1_Library/parameter_files`.

- Now go through the parameters with the description below and adjust the path of the sequence database.
  - **Attention!** Please be aware that we have two datasets you can choose from **(TTOF or QE)**. A few parameters differ for these data.

Database

- Define the path to the FASTA protein sequence database:
  `database_name = C:\DIA_Course\Data\napedro_3mixed_human_yeast_ecoli_20140403_iRT_reverse.fasta`
- Define whether the algorithm should generate internal decoys. We have previously generated decoys in the FASTA file and do not use Comet internal decoys.
  `decoy_search = 0`

CPU threads

- Define the number of CPU threads to be used. We will leave this number at 0, which means that Comet uses all available CPU threads (maximum 64).
  `num_threads = 0`

Masses

- Define the peptide mass tolerance and the units of this value. We will use **30 ppm for TTOF data and 10 for QE data**.
  ```
  peptide_mass_tolerance = 30
  peptide_mass_units = 2
  ```
- Define the type of experimental mass value (monoisotopic or average). We will use monoisotopic mass for both MS1 and MS2 level.
  ```
  mass_type_parent = 1
  mass_type_fragment = 1
  ```
- Specify the precursor tolerance type. This controls how the peptide mass tolerance parameter is applied. It can be applied to the singly charged peptide mass or it can be applied to the precursor m/z. We will apply mass tolerance to the precursor m/z.
  `precursor_tolerance_type = 1`

- Specify whether the peptide mass tolerance should take into account possible isotope errors in the precursor mass measurement. We will not perform isotope error searches.
  `isotope_error = 0`

Search enzyme

- Define the search enzyme number. We will use Trypsin
  `search_enzyme_number = 1`
- Define the number of enzyme termini a peptide must have. For example, if trypsin is specified as the search enzyme, only fully tryptic peptides would be analyzed if "num_enzyme_termini = 2" whereas semi-tryptic peptides would be analyzed if "num_enzyme_termini = 1". We will search for fully tryptic peptides only.
  `num_enzyme_termini = 2`
- Specify whether missed cleavage sites are allowed. We will not allow any missed cleavage sites.
  `allowed_missed_cleavage = 0`

Variable modifications

- Define the variable modifications to be used in the search. Up to 9 variable modifications can be specified. There are 4 entry settings that are associated with this parameter: The first entry is a decimal value specifying the modification mass difference. The second entry is the residue(s) that the modifications are possibly applied to. If more than a single residue is modified by the same mass difference, list them all as a string. The third entry is an integer 0 or 1 to specify whether the modification is a variable modification (0) or a binary modification (1). A variable modification analyzes all permutations of modified and unmodified residues. A binary modification analyzes peptides where all residues are either modified or all residues are not modified. The fourth entry is an integer specifying the maximum number of modified residues possible in a peptide for this modification entry. The fifth entry specifies the distance the modification is applied to from the respective terminus:
  - -2 = apply anywhere except c-terminal residue of peptide
  - -1 = no distance constraint
  - 0 = only applies to terminal residue
  - 1 = only applies to terminal residue and next residue
  - 2 = only applies to terminal residue through next 2 residues
  - N = only applies to terminal residue through next N residues where N is a positive integer

  The sixth entry specifies which terminus the distance constraint is applied to:
  - 0 = protein N-terminus
  - 1 = protein C-terminus
  - 2 = peptide N-terminus
  - 3 = peptide C-terminus.

  The seventh entry specifies whether peptides must contain this modification. If set to 1, only peptides that contain this modification will be searched for.
  - 0 = not forced to be present
  - 1 = modification is required.

  The default value is "0.0 X 0 3 -1 0 0" if this parameter is missing. We will not search any variable modifications and leave this part as it is.
- Specify the total/maximum number of residues that can be modified in a peptide.
  `max_variable_mods_in_peptide = 5`
- Specify `require_variable_mod.` This parameter takes in one input value that controls whether the analyzed peptides should contain at least one variable modification i.e. force all reported peptides to have a variable modification. 0 = consider both modified and unmodified peptides (default). 1 = analyze only peptides that contain a variable modification
  `require_variable_mod = 0`

Fragment ions

- Specify `fragment_bin_tol`. This parameter controls the bin size associated with fragment ions. It defines the mass width associated with a single MS/MS peak as it is stored internally in an array element. Although it's not explicitly a fragment ion tolerance, it's probably easiest to think of it as such. **For TTOF we choose 0.05 and for QE 0.02**
```
fragment_bin_tol = 0.05
```
- Specify `fragment_bin_offset`. This parameter controls how each fragment bin of size fragment_bin_tol is defined in terms of where each bin starts. We will not apply any offset.
```
fragment_bin_offset = 0.0
```
- Specify `theoretical_fragment_ions`. This parameter specifies how theoretical fragment ion peaks are represented. Even though Comet does not generate/store a theoretical spectrum, it does calculate fragment ion masses and this parameter controls how the acquired spectrum intensities at these theoretical mass locations contribute to the correlation score. We will set the fast correlation score to be a sum of the intensities at each theoretical fragment mass bin and half the intensity of each flanking bin.
```
theoretical_fragment_ions = 0
```
- Specify which fragment ions to be considered in the search. We will use B-ions and Y-ions.
```
use_A_ions = 0
use_B_ions = 1
use_C_ions = 0
use_X_ions = 0
use_Y_ions = 1
use_Z_ions = 0
use_NL_ions = 0
```

Output

- Specify how to output the search results. We will generate an output as a pepXML file with 3 search result hits (peptides) reported for each spectrum query, and with sample enzyme set to Trypsin.
```
output_sqtstream = 0
output_sqtfile = 0
output_txtfile = 0
output_pepxmlfile = 1
output_percolatorfile = 0
output_outfiles = 0
print_expect_score = 1
num_output_lines = 5
show_fragment_ions = 0
sample_enzyme_number = 1
```

mzXML parameters

- Specify the scan range to search. Only spectra within (and inclusive) of the specified scan range are searched. We will search all scans.
```
scan_range = 0 0
```
- Specify the precursor charge range to search. We will search all charge states.
```
precursor_charge = 2 5
```
- Specify whether or not to override existing precursor charge state information when present in the files with the charge range specified by the "`precursor_charge`" parameter. We will keep any known precursor charge state values in the input files.
```
override_charge = 0
```
- Specify which scans are searched (MS2 scans or MS3 scans). We will search MS2 scans.
```
ms_level = 2
```

- Specify which scan types are to be searched. We will search all scan types.
  ```
  activation_method = ALL
  ```

Misc. parameters

```
digest_mass_range = 600.0 5000.0
num_results = 100
skip_researching = 1
max_fragment_charge = 3
max_precursor_charge = 6
nucleotide_reading_frame = 0
clip_nterm_methionine = 1
spectrum_batch_size = 10000
decoy_prefix = reverse_
```

Spectral processing

- Specify the minimum number of m/z-intensity pairs that must be present in a spectrum before it is searched. This parameter can be used to avoid searching nearly sparse spectra that will not likely yield an identification. We will use a minimum of 10 peaks.
  ```
  minimum_peaks = 10
  minimum_intensity = 0
  remove_precursor_peak = 1
  remove_precursor_tolerance = 1.0
  clear_mz_range = 0.0 0.0
  ```

Static modifications

- Specify any static modification to any residue. We will use cysteine carbamidomethylation as static modification. This leads to mass shift of 57.021464.
  ```
  add_C_cysteine = 57.021464
  ```

- Now the file is ready! Save the file.

Now that the parameters for the sequence database search are specified we can run Comet from the command line ☺

- Go back to the "`run_DDA_LibraryGeneration.sh`" file.
- Insert the second command in line 12 (place holder for comet search command). The first comet search is already added as an example. Run now both comet commands
- **Note**! In order to revise if a program was run successfully we can write a log file. Therefore we will add "`&>> Tutorial1_log.txt`" on all our commands.
- **Note!** The places where you should insert the commands described in this tutorial are marked with "`## insert ##`".
- **Note!** The "\" at the end of a line means that the command continues in the next line - this is equivalent to having only one very long line.
- **Note!** If you use the **QE dataset please adjust the path and the filenames** accordingly. The next steps following are exactly the same for both datasets.

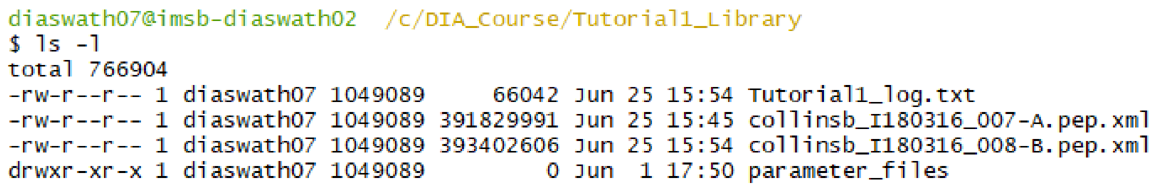
```
comet -
P/c/DIA_Course/Tutorial1_Library/parameter_files/comet.params.high-
high_TTOF \
-Ncollinsb_I180316_007-A \
/c/DIA_Course/Data/DDA_data/TTOF/collinsb_I180316_007-A.mzXML \
&>> Tutorial1_log.txt

comet -
P/c/DIA_Course/Tutorial1_Library/parameter_files/comet.params.high-
high_TTOF \
-Ncollinsb_I180316_008-B \
/c/DIA_Course/Data/DDA_data/TTOF/collinsb_I180316_008-B.mzXML \
&>> Tutorial1_log.txt
```

> **Note:** `-P` specifies the parameter file and `–N` points to the output directory and file name base. The last parameter is the input mzXML file.

The Comet search will result in the generation of one “pep.xml” output file for each mzXML file searched.

- Check if all output files were successfully generated in your directory:
  Type “ls –l” in the command line terminal.

```
ls –l
```



```
diaswath07@imsb-diaswath02  /c/DIA_Course/Tutorial1_Library
$ ls -l
total 766904
-rw-r--r-- 1 diaswath07 1049089     66042 Jun 25 15:54 Tutorial1_log.txt
-rw-r--r-- 1 diaswath07 1049089 391829991 Jun 25 15:45 collinsb_I180316_007-A.pep.xml
-rw-r--r-- 1 diaswath07 1049089 393402606 Jun 25 15:54 collinsb_I180316_008-B.pep.xml
drwxr-xr-x 1 diaswath07 1049089         0 Jun  1 17:50 parameter_files
```

If you see the two new pep.xml files in your directory you can move to the next step of scoring the search results with PeptideProphet (file size is about 30-37 MB).

As dozens to hundreds of DDA runs might be combined to generate a comprehensive peptide-query-parameter library, it is important to thoroughly control the FDR of the final data set both at the level of PSMs and at the level of inferred proteins. PeptideProphet automatically validates the peptide assignments to a certain MS/MS spectra made by a sequence database search algorithm. From each dataset, it creates distributions of search scores and peptide properties among correct and incorrect peptide assignments, and uses those distributions to compute for each result a probability that it is correct. PeptideProphet can be used to validate several different sequence database search algorithms outputs.

- Run PeptideProphet on all of the Comet results by first including the following command to your script and then running it:

```
xinteract -dreverse_ \
-OARPd \
-Ninteract.comet.pep.xml \
collinsb_*.pep.xml \
&>> Tutorial1_log.txt
```

Options:
`-d`: specifies the recognition pattern for the reverse decoy sequences as used in the FASTA database, in our case all decoy sequences start with “`reverse_`”
`-OARPd`: The `–O` option specifies that the following letters are parameters for PeptideProphet. `A` stands for using accurate mass binning, `R` for using retention time information, `P` for using a non-parametric statistical model, and d for reporting decoy hits with a computed probability based on the model learned.
`-N`: name of the basic output pep.xml file that is created from peptideProphet

Peptide prophet finally generates 4 output files from which we will only use the `interact.comet.pep.xml` file for further processing. The next step is to combine all search results with iProphet. iProphet combines the evidence from multiple identifications of the same peptide sequences across different spectra, experiments, precursor ion charge states, and modified states. When applied in tandem with PeptideProphet, it provides more accurate representation of the multilevel nature of DDA data. It allows accurate and effective integration of the results from multiple database search engines applied to the same data. The use of iProphet in the TPP increases the number of correctly identified peptides at a constant false discovery rate. As the main outcome, iProphet permits the calculation of accurate posterior probabilities and false discovery rate estimates at the level of sequence identical peptide identifications, which in turn leads to more accurate probability estimates at the protein level.

- Run iProphet on the `interact.comet.pep.xml` file:

```
InterProphetParser DECOY=reverse_ \
interact.comet.pep.xml \
iProphet.pep.xml \
&>> Tutorial1_log.txt
```

Options:
`DECOY`: specifies the recognition pattern for the reverse decoy sequences as used in the FASTA database, in our case all decoy sequences start with "`reverse_`"
Second-last argument: input file from peptideProphet
Last argument: output filename that will be generated by iProphet

4. FDR control

In order to ensure the FDR control on the protein level across all measured DDA runs, we use MAYU. MAYU is an algorithm that implements a robust method to estimate the FDR of large-scale DDA data sets at PSM, peptide and protein levels, and it can be applied to the iProphet output. The iProphet probability is used as a ranking, and the FDR estimated by MAYU is controlled on the basis of decoys.

- To process the iProphet results with MAYU for FDR estimation, run the following command:

```
perl /c/TPP/bin/Mayu.pl \
-A iProphet.pep.xml \
-C
/c/DIA_Course/Data/napedro_3mixed_human_yeast_ecoli_20140403_iRT_rever
se.fasta \
-E reverse_ \
-G 0.01 \
-H 101 \
-I 0 \
&>> Tutorial1_log.txt
```

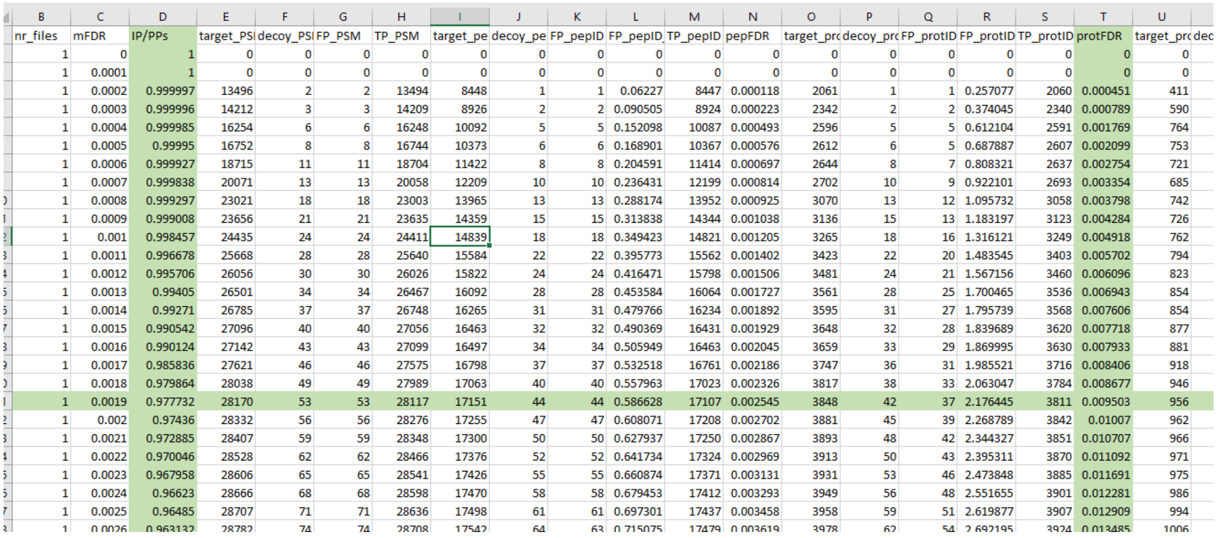
Options:
`-A`: file name of the iProphet output pep.xml file
`-C`: path to FASTA protein sequence database
`-E`: specifies the recognition pattern for the reverse decoy sequences as used in the FASTA database, in our case all decoy sequences start with "`reverse_`"
`-G`: defines upper limit of error analysis (maximal mFDR)
`-H`: defines the resolution of error analysis (mFDR steps)
`-I`: number of missed cleavages used for database search. Use the same number of missed cleavages for tryptic peptides as for the database search. We didn't allow any missed cleavage sites in the Comet search.

MAYU generates two output files containing information on FDR estimations on PSM, peptide and protein level. In our next step, we will look up the minimum iProphet probability (IP/PPs) at which the protein FDR is <1% in the file ending with "`_main_1.07.csv`".

- Open the file ending with "`_main_1.07.csv`" in Excel
- Identify the column with the name "protFDR"
- Go down until you reach the row with the last value that is smaller than 0.01
- Mark the row and find the column "IP/PP", which is the iProphet probability score
- Write down the number in your marked row – this is the score cutoff you use for the further analysis (result should be in the range of: 0.97-0.98 in our example).

Caution! Depending on the computer you ran the above software tools, the value might be slightly different.). **Also the QE data will give a different value here**.

Here you can see a screenshot of the MAYU output csv file and the corresponding columns of interest to retrieve the IP/PPs value corresponding to <1% protein FDR:



B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	
nr_files	mFDR	IP/PPs	target_PSI	decoy_PSI	FP_PSM	TP_PSM	target_pe	decoy_pe	FP_pepID	FP_pepID_	TP_pepID	pepFDR	target_pro	decoy_pro	FP_protID	FP_protID	TP_protID	protFDR	target_pro	dec
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	0.0001	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
1	0.0002	0.999997	13496	2	2	13494	8448	1	1	0.06227	8447	0.000118	2061	1	1	0.257077	2060	0.000451	411	
1	0.0003	0.999996	14212	3	3	14209	8926	2	2	0.090505	8924	0.000223	2342	2	2	0.374045	2340	0.000789	590	
1	0.0004	0.999985	16254	6	6	16248	10092	5	5	0.152098	10087	0.000493	2596	5	5	0.612104	2591	0.001769	764	
1	0.0005	0.99995	16752	8	8	16744	10373	6	6	0.168901	10367	0.000576	2612	6	5	0.687887	2607	0.002099	753	
1	0.0006	0.999927	18715	11	11	18704	11422	8	8	0.204591	11414	0.000697	2644	8	7	0.808321	2637	0.002754	721	
1	0.0007	0.999838	20071	13	13	20058	12209	10	10	0.236431	12199	0.000814	2702	10	9	0.922101	2693	0.003354	685	
1	0.0008	0.999297	23021	18	18	23003	13965	13	13	0.288174	13952	0.000925	3070	13	12	1.095732	3058	0.003798	742	
1	0.0009	0.999008	23656	21	21	23635	14359	15	15	0.313838	14344	0.001038	3136	15	13	1.183197	3123	0.004284	726	
1	0.001	0.998457	24435	24	24	24411	14839	18	18	0.349423	14821	0.001205	3265	18	16	1.316121	3249	0.004918	762	
1	0.0011	0.996678	25668	28	28	25640	15584	22	22	0.395773	15562	0.001402	3423	22	20	1.483545	3403	0.005702	794	
1	0.0012	0.995706	26056	30	30	26026	15822	24	24	0.416471	15798	0.001506	3481	24	21	1.567156	3460	0.006096	823	
1	0.0013	0.99405	26501	34	34	26467	16092	28	28	0.453584	16064	0.001727	3561	28	25	1.700465	3536	0.006943	854	
1	0.0014	0.99271	26785	37	37	26748	16265	31	31	0.479766	16234	0.001892	3595	31	27	1.795739	3568	0.007606	854	
1	0.0015	0.990542	27096	40	40	27056	16463	32	32	0.490369	16431	0.001929	3648	32	28	1.839689	3620	0.007718	877	
1	0.0016	0.990124	27142	43	43	27099	16497	34	34	0.505949	16463	0.002045	3659	33	29	1.869995	3630	0.007933	881	
1	0.0017	0.985836	27621	46	46	27575	16798	37	37	0.532518	16761	0.002186	3747	36	31	1.985521	3716	0.008406	918	
1	0.0018	0.979864	28038	49	49	27989	17063	40	40	0.557963	17023	0.002326	3817	38	33	2.063047	3784	0.008677	946	
1	0.0019	0.977732	28170	53	53	28117	17151	44	44	0.586628	17107	0.002545	3848	42	37	2.176445	3811	0.009503	956	
1	0.002	0.97436	28332	56	56	28276	17255	47	47	0.608071	17208	0.002702	3881	45	39	2.268789	3842	0.01007	962	
1	0.0021	0.972885	28407	59	59	28348	17300	50	50	0.627937	17250	0.002867	3893	48	42	2.344327	3851	0.010707	966	
1	0.0022	0.970046	28528	62	62	28466	17376	52	52	0.641734	17324	0.002969	3913	50	43	2.395311	3870	0.011092	971	
1	0.0023	0.967958	28606	65	65	28541	17426	55	55	0.660874	17371	0.003131	3931	53	46	2.473848	3885	0.011691	975	
1	0.0024	0.96623	28666	68	68	28598	17470	58	58	0.679453	17412	0.003293	3949	56	48	2.551655	3901	0.012281	986	
1	0.0025	0.96485	28707	71	71	28636	17498	61	61	0.697301	17437	0.003458	3958	59	51	2.619877	3907	0.012909	994	
1	0.0026	0.963132	28782	74	74	28708	17542	64	63	0.715075	17479	0.003619	3978	62	54	2.692195	3924	0.013485	1006	

5. Combination of search results to a spectral library

The next step is the generation of a spectral library using SpectraST. SpectraST is a software tool that compiles all fragment ion spectra assigned to a specific peptide sequence above a certain quality threshold (e.g., iProphet probability) into a spectral library format.

We will use SpectraST in two consecutive iterations. First, we will transform all retention times into a normalized retention time scale. This is accomplished by establishing a linear correlation between experimental retention times and unit-less absolute retention time values (iRTs) for retention time reference peptides identified in each DDA run. For this purpose, we have spiked in the iRT reference peptides from Biognosys. The resulting correlation curves are then used to convert the retention time for all the other peptides identified in the corresponding DDA runs into iRT scale.

- Run SpecrtaST to generate a redundant spectral library with all spectra above the MAYU-selected iProphet probability score and align the retention times
  Caution! Replace the number following `-cP` with the cutoff you read out from the MAYU output in the step above.

```
spectrast -cNSpecLib \
-cICID-QTOF \
-cf "Protein! ~ reverse_" \
-cP0.977732 \
-c_IRT../Data/irtkit.txt \
-c_IRR iProphet.pep.xml \
&>> Tutorial1_log.txt
```

Options:
`-cN`: output file name
`-cI`: Set the instrument and acquisition settings of the spectra (in case not specified in data files). Examples: -cICID, -cIETD, -cICID-QTOF, -cIHCD. The latter two are treated as high-mass accuracy spectra. **If you use QE data please use -cIHCD**
`-cf`: Specifies filtering criteria. In our case proteins that start with the decoy tag "reverse_" are removed and not used for spectral library generation
`-cP`: minimum iProphet probability to include a PSM for spectral library generation. This is the MAYU output IP/PPs value corresponding to <1% protein FDR (in our example the value is 0.977732).
`-c_IRT`: File specifying landmark peptides to use for normalizing retention times to iRTs
`-c_IRR`: normalize retention times by linear regression
last argument: iProphet output file

- **Caution!** The irtkit.txt file contains the peptide sequences to be used as iRT retention time reference peptides. This file needs to be adjusted in case reference peptides different from the ones suggested have been used.
- **Critical step!** The correlation coefficient R^2 of the linear regression should be >0.95. Open the spectrast.log file in a text editor and scroll to the end to see the linear regression equation and the R^2.

  - Open the spectrast.log file in Notepad++ and check the correlation coefficient R^2 of the linear regression under "Final fitted equation" to make sure it is good ☺.

In the second step, SpectraST is used to generate consensus spectrum for each peptide sequence based on all runs where a spectrum was matched to the specific peptide sequence. Even though modern mass spectrometers display a reasonably high level of reproducibility in repeat recordings of fragment ion spectra of the same peptide across replicates, the consolidation of multiple fragmentation observations of the same peptide precursor ion into a single consensus spectrum provides a more accurate fragmentation pattern than any single spectrum (best replicate) for that precursor. The consensus spectrum, therefore, is the optimal representation of the fragment ion spectrum of a targeted peptide.

- Generate a consensus library by running the following command:

```
spectrast -cNSpecLib_cons \
-cICID-QTOF \
-cAC SpecLib.splib \
&>> Tutorial1_log.txt
```

Options:
`-cN`: output file name
`-cI:` Set the instrument and acquisition settings of the spectra (in case not specified in data files). Examples: -cICID, -cIETD, -cICID-QTOF, -cIHCD. The latter two are treated as high-mass accuracy spectra. **If you use QE data please use -cIHCD**
`-cAC`: create the consensus spectrum of all replicate spectra of each peptide ion.

- Generate a SpectraST MRM transition list:

```
spectrast -cNSpecLib_pqp \
-cICID-QTOF \
-cM \
SpecLib_cons.splib \
&>> Tutorial1_log.txt
```

Options:
`-cN`: output file name
`-cI:` Set the instrument and acquisition settings of the spectra (in case not specified in data files). Examples: -cICID, -cIETD, -cICID-QTOF, -cIHCD. The latter two are treated as high-mass accuracy spectra. **If you use QE data please use -cIHCD**
`-cM`:

6. Peptide-query parameter library generation

The final step during library generation for targeted DIA analysis is the definition of specific peptide-query parameters within the spectral library. This includes filtering of the spectral library and selection of the most representative spectra per peptide ion. The number of fragment ions (spectra) per peptide ion should be high enough to ensure specificity of identification within a SWATH MS map, but not too high, as less-intense transitions introduce noise into the extracted data. As investigated in previous analysis, a set of the 6 most intense fragment ions for each precursor is a good trade-off. We further only use the most frequently occurring fragment ions (b and y) and charge states (1 and 2).

- Convert the SpectraST MRM to TraML

```
TargetedFileConverter \
-in SpecLib_pqp.mrm \
-out transitionlist.TraML \
&>> Tutorial1_log.txt
```

Options:
`-in`: input file
`-out`: output file

- Generate target assays by copying this command in your script and run it:
- **Note**: **When you use the QE data, adjust the windows file accordingly.**

```
OpenSwathAssayGenerator \
-in transitionlist.TraML \
-out transitionlist_optimized.TraML \
-swath_windows_file /c/DIA_Course/Data/swath64_w_header.txt \
&>> Tutorial1_log.txt
```

In order to enable decoy-based error-rate control with pyProphet downstream of OpenSWATH, it is important to append the assay library with decoy-query parameters.

- Append decoy transitions to the spectral library:

```
OpenSwathDecoyGenerator \
-in transitionlist_optimized.TraML \
-out transitionlist_optimized_decoys.TraML \
-method shuffle \
&>> Tutorial1_log.txt
```

Options:
`-in`: input TraML library
`-out`: output TraML library with appended decoy transitions
`-method`: Decoy generation method ('shuffle', 'pseudo-reverse', 'reverse', 'shift')
`-append`: Set this flag if decoys should be directly appended to output library

- Convert the library to the pqp format for the further OpenSWATH analysis:

```
TargetedFileConverter \
-in transitionlist_optimized_decoys.TraML \
-out transitionlist_optimized_decoys.pqp \
&>> Tutorial1_log.txt
```

Now you have a final library that can be used for targeted, peptide-centric DIA analysis with OpenSWATH ☺.

Finally, to inspect the library, we will convert it back to the tsv format and take a closer look.

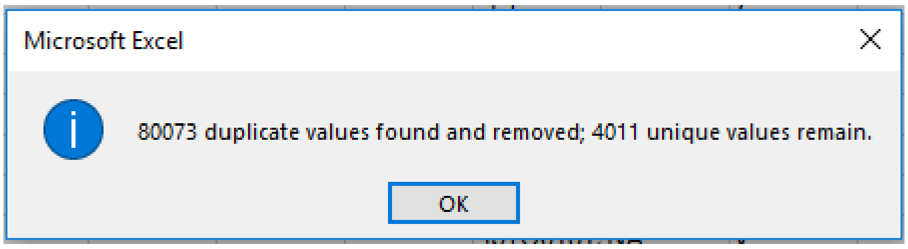
- Run TargetedFileConverter

```
TargetedFileConverter \
-in transitionlist_optimized.TraML \
-out transitionlist_optimized.tsv \
&>> Tutorial1_log.txt
```

You no got through the entire library generation workflow and have a script that you can use as a reference and scaffold for the library generation of other datasets ☺. Save the shell script and exit Rstudio.

Now we would like to visually inspect the library we just build and see how many peptides and proteins are represented per organism.

- Open `transitionlist_optimized.tsv` in Excel.
- **Note!** The best way is to first open Excel, go to "File", select "Open", select "All files (*.*)" in the lower right corner and select the file. Once opened, use the delimiter function to specify that the file is tab delimited.
- Mark column A and go to the DATA tab. Press the button "Text to Columns".
- Select "Delimited", click "Next", select "tab" as delimiter, click "Finish" and accept with "OK". You may get an error that "There's already data in here. Do you want to replace it?". Click "OK"
- Now we want to count the number of proteins in the library. For this we first select the entire data by clicking on the upper left triangle next to the first cell (A1), go to the DATA tab again and click the button "Remove Duplicates".
- In the pop-up window, click "unselect all" and them mark only "ProteinId", click "OK" and "OK".



80073 duplicate values found and removed; 4011 unique values remain.

- In the next step we want to filter for targets of the specific organisms. For this we first select the entire data by clicking on the upper left triangle next to the first cell (A1), select the “Filter” function, and now all the column headers have filter buttons.
- Filter the “ProteinId” column for containing the text “HUMAN”. For this we click on the filter button next to “ProteinId”, select “Text_Filters”, select “Contains…” and specify the value to be “HUMAN”.
- Filter the “ProteinId” column for containing the text “ECOLI”. For this we click on the filter button next to “ProteinId”, select “Text_Filters”, select “Contains…” and specify the value to be “ECOLI”.
- Filter the “ProteinId” column for containing the text “YEAS8”. For this we click on the filter button next to “ProteinId”, select “Text_Filters”, select “Contains…” and specify the value to be “YEAS8”.
- Why do these numbers not add up to the total number of targets?

References:

1. Schubert OT, Gillet LC, Collins BC, Navarro P, Rosenberger G, Wolski WE, Lam H, Amodei D, Mallick P, MacLean B, Aebersold R. Building high-quality assay libraries for targeted analysis of SWATH MS data. Nat Protoc. 2015 Mar;10(3):426-41.

We would like to thank SystemsX for supporting the Zurich DIA / SWATH Course 2018.

SystemsX.ch
The Swiss Initiative in Systems Biology