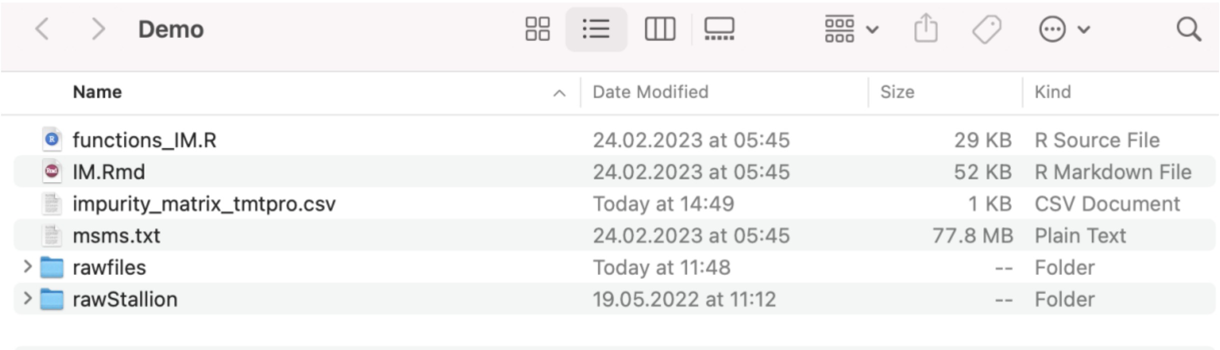
User Guide to:
Interference Modeling in Multiplex Proteomics

Overview

This document serves as a guide to the demo on our GitHub (https://github.com/maxperutzlabs-ms/InterferenceModeling_in_MultiplexProteomics) and for the interference modeling workflow in general, initially described in our publication (https://doi.org/10.1016/j.mcpro.2023.100694). The demo comes with its own dataset on which basis the workflow can be experienced from start to finish. We recommend running the demo first before applying the workflow to your own data – this will get the user familiar with the required data input, the script's adjustable parameters as well as the intermediate and final data output. In this user guide, we will look at the required setup for running the workflow and then how to run it on the basis of the available demo dataset. Finally, there is a description of the generated output.

Required Setup

To apply the workflow, we recommend creating a new directory (here: "Demo") containing all the necessary software (i.e. scripts and tools) as well as the required data input to run the script. For this demo, the required setup looks like this:



Name	Date Modified	Size	Kind
functions_IM.R	24.02.2023 at 05:45	29 KB	R Source File
IM.Rmd	24.02.2023 at 05:45	52 KB	R Markdown File
impurity_matrix_tmtpro.csv	Today at 14:49	1 KB	CSV Document
msms.txt	24.02.2023 at 05:45	77.8 MB	Plain Text
rawfiles	Today at 11:48	--	Folder
rawStallion	19.05.2022 at 11:12	--	Folder

Figure 1 - *Overview of required setup of files and directories.*

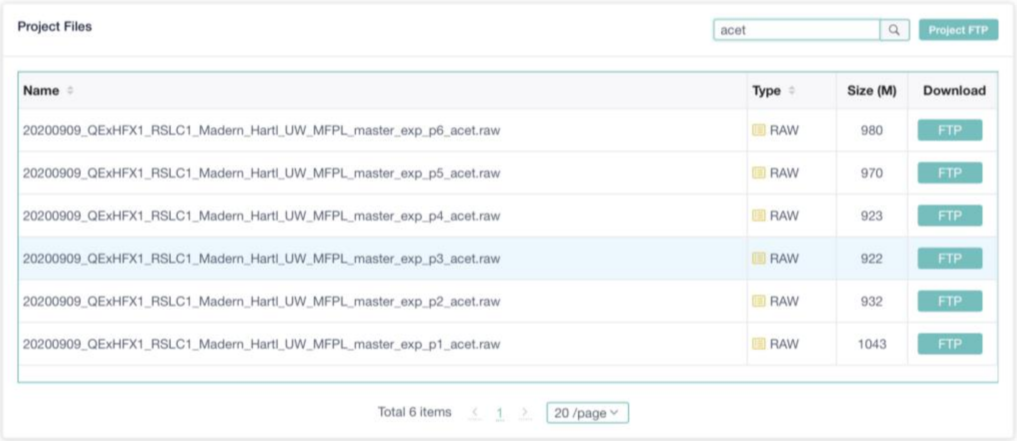
We will now give a detailed description of all the individual files shown in Figure 1:

- **msms.txt** is a MaxQuant PSM table, i.e. a database search result that lists row-wise PSMs. Note that a FragPipe PSM table is also supported by the workflow. For this demo, the input PSM- table "msms.txt" is available on GitHub in the Demo folder. Note that this table was already filtered for certain raw files of interest to decrease the file size, however this will not change the analysis. The table lists PSMs from measurements of acetyl-peptide enriched samples.

- **rawfiles** is a required subdirectory containing the raw files relevant to your data. In more detail, this subdirectory needs to contain all the Thermo raw files (*.raw) that were used in the database search to obtain the PSM table (e.g., msms.txt). The raw files are needed in order to extract scan-specific information like noise values etc. via the rawStallion tool (a Windows command line tool, see below for more information). The extracted information is subsequently stored as tsv files which can be read into R. For this demo, we need the following six raw files that were obtained from measuring acetyl (K)-peptide enriched samples:

  20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p1_acet.raw
  20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p2_acet.raw
  20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p3_acet.raw
  20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p4_acet.raw
  20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p5_acet.raw
  20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p6_acet.raw

  The six raw files are available for download on PRIDE (identifier PXD040449, https://www.ebi.ac.uk/pride/archive/projects/PXD040449). Please download them and put them into a subdirectory called "rawfiles". On PRIDE, these six files raw files can easily be found by searching for "acet" in the search bar:

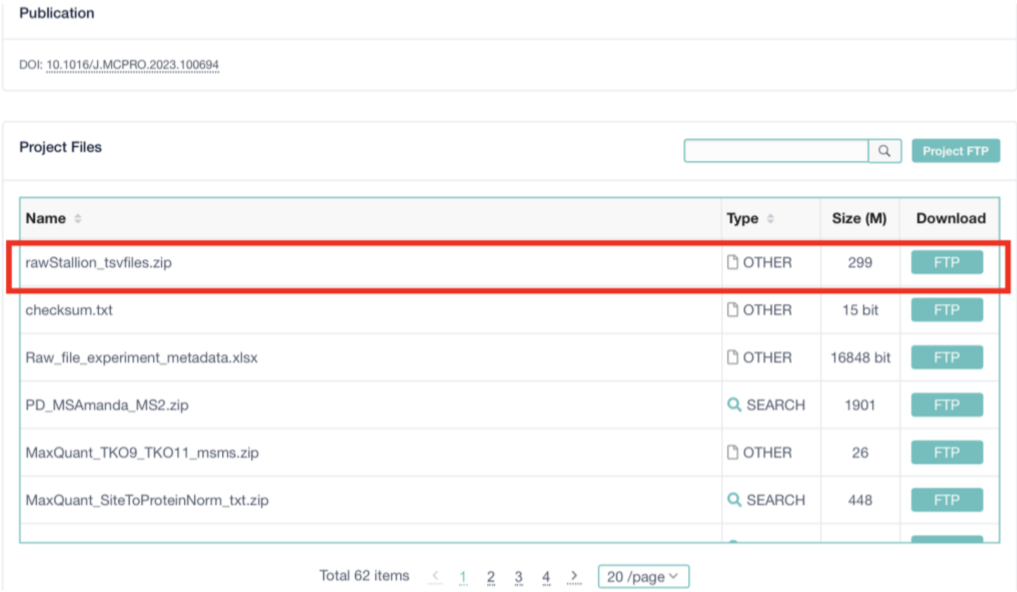


Name	Type	Size (M)	Download
20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p6_acet.raw	RAW	980	FTP
20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p5_acet.raw	RAW	970	FTP
20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p4_acet.raw	RAW	923	FTP
20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p3_acet.raw	RAW	922	FTP
20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p2_acet.raw	RAW	932	FTP
20200909_QExHFX1_RSLC1_Madern_Hartl_UW_MFPL_master_exp_p1_acet.raw	RAW	1043	FTP

Figure 2 *- The Thermo raw files needed in this demo are stored on PRIDE.*

When running the workflow, the R code will call on rawStallion to extract scan-specific information from all Thermo raw files located in the “rawfiles” folder and write this information to corresponding tsv files. After this point, the raw files are not needed anymore.

Please note that this demo can also be run by directly using the tsv files that the rawStallion tool would extract from the raw files. This is especially relevant for users without direct access to a windows OS, which is a requirement for rawStallion. The tsv files are available on PRIDE (identifier PXD040449, https://www.ebi.ac.uk/pride/archive/projects/PXD040449) and stored as “rawStallion_tsvfiles.zip”:



Publication

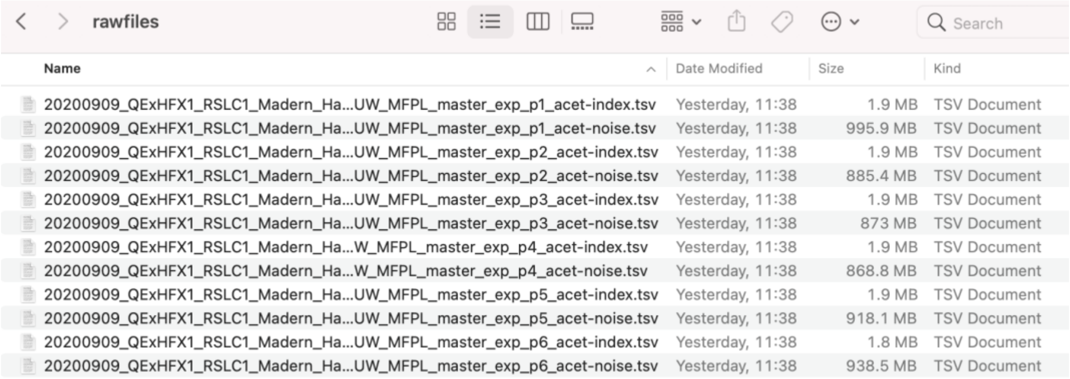
DOI: 10.1016/J.MCPRO.2023.100694

Project Files

Name	Type	Size (M)	Download
rawStallion_tsvfiles.zip	OTHER	299	FTP
checksum.txt	OTHER	15 bit	FTP
Raw_file_experiment_metadata.xlsx	OTHER	16848 bit	FTP
PD_MSAmanda_MS2.zip	SEARCH	1901	FTP
MaxQuant_TKO9_TKO11_msms.zip	OTHER	26	FTP
MaxQuant_SiteToProteinNorm_txt.zip	SEARCH	448	FTP

Figure 3 *- The optional tsv files for this Demo are also stored on PRIDE. By using the tsv files directly, a user can run the demo without the rawStallion tool.*

Please unzip “rawStallion_tsvfiles.zip”, and put the tsv files into the “rawfiles” subdirectory (see Figure 4). Note that there two tsv files per Thermo raw file – this is intended.



rawfiles

Name	Date Modified	Size	Kind
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p1_acet-index.tsv	Yesterday, 11:38	1.9 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p1_acet-noise.tsv	Yesterday, 11:38	995.9 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p2_acet-index.tsv	Yesterday, 11:38	1.9 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p2_acet-noise.tsv	Yesterday, 11:38	885.4 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p3_acet-index.tsv	Yesterday, 11:38	1.9 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p3_acet-noise.tsv	Yesterday, 11:38	873 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...W_MFPL_master_exp_p4_acet-index.tsv	Yesterday, 11:38	1.9 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...W_MFPL_master_exp_p4_acet-noise.tsv	Yesterday, 11:38	868.8 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p5_acet-index.tsv	Yesterday, 11:38	1.9 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p5_acet-noise.tsv	Yesterday, 11:38	918.1 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p6_acet-index.tsv	Yesterday, 11:38	1.8 MB	TSV Document
20200909_QExHFX1_RSLC1_Madern_Ha...UW_MFPL_master_exp_p6_acet-noise.tsv	Yesterday, 11:38	938.5 MB	TSV Document

Figure 4 *- The downloaded tsv files for the Demo are stored in the “rawfiles” subdirectory. Running the rawStallion tool on the Thermo raw files will lead to the same file setup.*

Regardless of whether you run the demo with the Thermo raw files (and rawStallion), or instead use the shortcut with the tsv files only, the workflow will use the same R code and the results will be identical.

- **rawStallion** is a Windows command-line application utilizing the Thermo RawFileReader software. rawStallion reads relevant information (e.g. noise values, m/z values, intensity values, etc.) from Thermo raw files and writes them to two tsv files per raw file. The tool can be downloaded here: https://github.com/fstanek/rawStallion.

  Note that when running the R program, the rawStallion tool will be called automatically by invoking OS commands from within R. In other words, the user will only have to run R code and make sure that a) rawStallion is functioning and located in the directory as shown in Figure 1, and b) the relevant Thermo raw files are located in the "rawfiles" subdirectory.

- **IM.Rmd** is the R Markdown script that performs the entire workflow in R. This file is located in the main folder of the repository on GitHub.

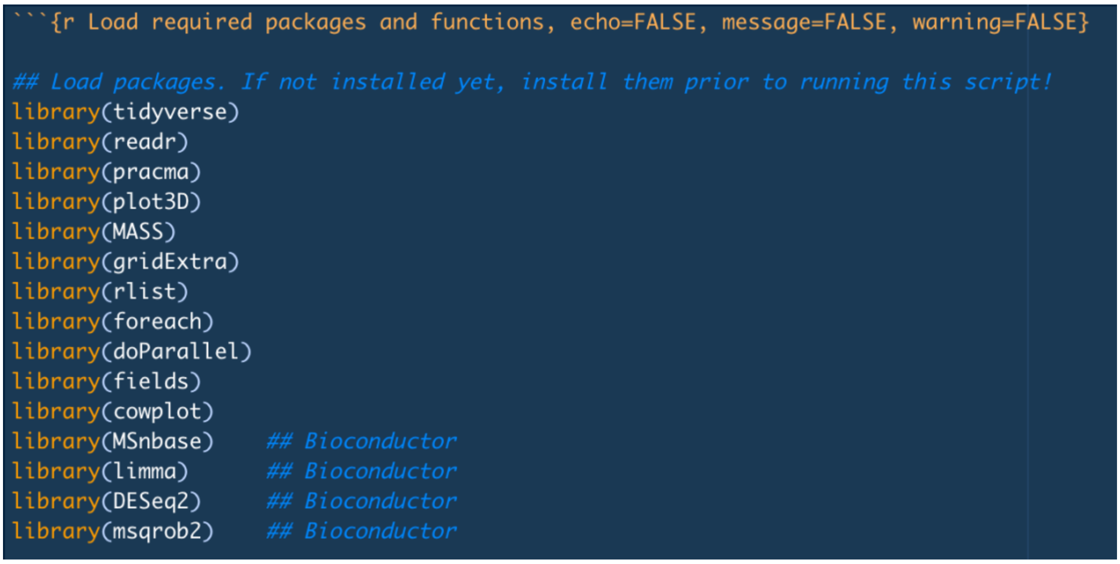
- **functions_IM.R** contains functions automatically sourced by the main script "IM.Rmd". This file is located in the main folder of the repository on GitHub.

- **impurity_matrix_tmtpro.csv** is a csv file that contains an isotopic impurity matrix specific to the multiplexing label reagents used in the experiment. This file is required because the workflow extracts reporter ion intensities from MS2 spectra and impurity-correct them for isotopic impurities using this matrix. Rows in the matrix reflect relative contribution of individual reagents to reporter ion channels ordered along the columns. Please find the corresponding file "impurity_matrix_tmtpro.csv" required for this Demo on GitHub in the Demo folder. Note that for any other dataset, impurity matrix is ideally manually curated using the isotopic impurity information on the product sheet that comes with the labeling kit.

Running the Program

Now that we have all the required files assembled as shown in Figure 1, open the script IM.Rmd in R studio and ensure that the working directory is set to this directory (here "Demo"). We can then proceed to go through the script IM.Rmd which will carry out the whole workflow.

The first code block loads multiple required packages:



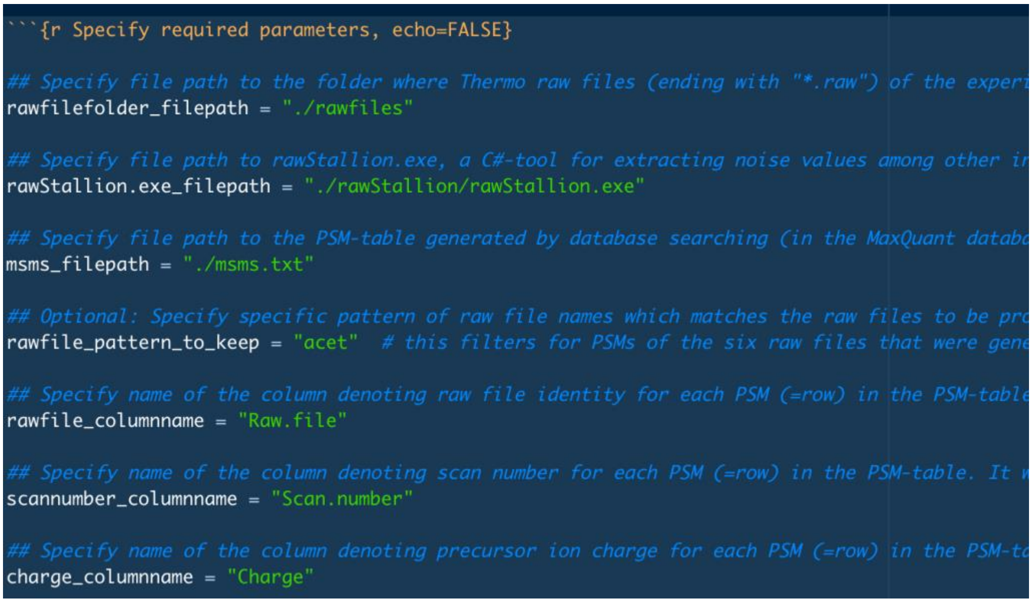
~~~
```{r Load required packages and functions, echo=FALSE, message=FALSE, warning=FALSE}

## Load packages. If not installed yet, install them prior to running this script!
library(tidyverse)
library(readr)
library(pracma)
library(plot3D)
library(MASS)
library(gridExtra)
library(rlist)
library(foreach)
library(doParallel)
library(fields)
library(cowplot)
library(MSnbase)    ## Bioconductor
library(limma)      ## Bioconductor
library(DESeq2)     ## Bioconductor
library(msqrob2)    ## Bioconductor
~~~

Figure 5 *- Code block for loading required R and Bioconductor packages.*

These packages need to be installed prior to running the script. Regular R packages can be installed within R studio. To install Bioconductor packages, please visit the respective Bioconductor websites (e.g. https://bioconductor.org/packages/release/bioc/html/MSnbase.html) and follow the instructions in the respective “Installation” section.

The second code block is where the user is required to specify the input parameters needed to successfully run the script. Here is a screenshot of the top few parameters:



~~~
```{r Specify required parameters, echo=FALSE}

## Specify file path to the folder where Thermo raw files (ending with "*.raw") of the experi
rawfilefolder_filepath = "./rawfiles"

## Specify file path to rawStallion.exe, a C#-tool for extracting noise values among other i
rawStallion.exe_filepath = "./rawStallion/rawStallion.exe"

## Specify file path to the PSM-table generated by database searching (in the MaxQuant datab
msms_filepath = "./msms.txt"

## Optional: Specify specific pattern of raw file names which matches the raw files to be pr
rawfile_pattern_to_keep = "acet"  # this filters for PSMs of the six raw files that were gene

## Specify name of the column denoting raw file identity for each PSM (=row) in the PSM-table
rawfile_columnname = "Raw.file"

## Specify name of the column denoting scan number for each PSM (=row) in the PSM-table. It
scannumber_columnname = "Scan.number"

## Specify name of the column denoting precursor ion charge for each PSM (=row) in the PSM-t
charge_columnname = "Charge"
~~~

Figure 6 *- Code block listing adjustable parameters for the workflow. These parameters can be adjusted by the user to accommodate any new data set.*

These parameters aim to configure the program to your specific input data. Please note that anything outside of this code block does not require input from the user ☺. In its current form on GitHub, the script's parameter settings are configured to run the demo.

To gain insight into each parameter, please refer to the respective comments above each line of code. If specified incorrectly, the program might produce errors down the line or give erroneous results (e.g. when column names or regular expression patterns are defined incorrectly). If a parameter is described as "Optional", specifying this parameter is not required for successfully running the program, as some steps in the workflow can be skipped. Note that optional parameters can be set to their default value (e.g., NULL, or "") to skip the corresponding code sections. The default values of optional parameters are mentioned in the comments.

Once all parameters are correctly specified, the entire script can be executed code block after code block. Each code block performs a specific task and often produces intermediate output (visual and/or textual) of interest. We hope that the intermediate output will be clear in the context of our publication (https://doi.org/10.1016/j.mcpro.2023.100694) and give insight into your data. Additionally, the comments in the code provide additional information on the workflow.

Roughly, the script performs the following steps in order:

1) Reading in of the PSM table and modifying it (e.g. filtering for raw files of interest, filtering out contaminants, etc.).

2) Using rawStallion to extract raw file-specific variables (e.g. noise-values, intensity values, m/z-values, etc.) and save them as tsv files.

3) Calculation of PSM-specific variables using the tsv files (e.g. reporter intensities, PPF, TIW, etc.). Adding those variables to the PSM table.

4) Calculation of other variables for modeling (impurity-corrected reporter intensities, empirical peptide densities, number of labels per peptide, empirical peptide classes, etc.).

5) Modeling via robust multiple linear regression.

6) Using estimated model parameters to calculate EIL values (Estimated Interference Levels).

7) Performing between-sample normalization.

8) Performing interference-correction based on EIL values on normalized intensity data, which generates interference-corrected intensity data.

9) Exporting the results, i.e. the modified PSM table, as txt file. The result table contains extra variables (columns) of interest, including EIL and interference-corrected reporter intensities.

Please note that at several points during the workflow, the script will create session images and save them in your working directory, e.g. "session_including_MS1_features_2023-02-28.RData". These sessions can be reloaded at a later time point via the R function load() to access or recreate part of the workflow without starting from scratch again.

Code blocks described as "Optional" can be skipped, since they are not required to successfully run the program. Further, please note that some code blocks will take some time to run - sometimes several hours - especially if there are many raw files to be processed.

Output

The script produces an output table called "modified_PSM.txt" that is stored in a (newly created) subdirectory named "Results" (see "Demo/Results/modified_PSM.txt" on GitHub as an example). This table is a modified version of the input PSM table and contains multiple additional columns that were generated while running the program. Notable column additions are: Normalized reporter intensity columns (suffix *"_norm");* normalized interference-corrected reporter intensity columns (suffix *"_norm__interference_corrected");* and the columns "*EIL*" (Estimated Interference Level) and "*PPF*" (Precursor Purity Fraction).

Please note that the output table "modified_PSM.txt" of this demo serves as input to the demo for site-to-protein normalization in multiplex proteomics (see GitHub repository: https://github.com/maxperutzlabs-ms/SiteToProteinNormalization_in_MultiplexProteomics). Hence, a demo user can directly continue from here if desired.

Additional Comments

We advise that all the raw files used in the workflow at a time are of similar nature, i.e., that they come from the same experiment and even the same sub-experiment (e.g., acetylome measurements) within the experiment. If there are different kinds of raw files in a PSM table (e.g. measurements of unmodified peptides, acetyl-peptides, and phospho-peptides – all from the same experiment), it is best to run them all separately through this workflow. The reason for that is that they are best normalized and thus interference-corrected independently from each other, since distinct types of peptides (i.e. unmodified, acetyl, phospho, etc.), even within the same experiment, can differ in their relative sample contribution and thus background interference level. Crucially, if the assumption of uniform background interference (after between-sample normalization) is compromised, the interference correction algorithm will most likely introduce bias.