# MzidLibrary user guide

Version 1.6.11

Date: 1/12/2014

Contents

[MzidLibrary user guide 1](#_Toc405203346)

[1 Overview 2](#_Toc405203347)

[1.1 FalseDiscoveryRate 3](#_Toc405203348)

[1.2 FalseDiscoveryRateGlobal 4](#_Toc405203349)

[1.3 Omssa2mzid 4](#_Toc405203350)

[1.4 Tandem2mzid 4](#_Toc405203351)

[1.5 Csv2mzid 5](#_Toc405203352)

[1.6 Mzid2Csv 6](#_Toc405203353)

[1.7 Threshold 7](#_Toc405203354)

[1.8 ProteoGrouper 7](#_Toc405203355)

[1.9 InsertMetaDataFromFasta 8](#_Toc405203356)

[1.10 AddEmpaiToMzid 8](#_Toc405203357)

[1.11 CombineSearchEngines 8](#_Toc405203358)

[1.12 CreateRestrictedFASTADatabase 9](#_Toc405203359)

[1.13 MzIdentMLToMzTab 9](#_Toc405203360)

[1.14 RescoreMods 10](#_Toc405203361)

[1.15 CombinePSMMzidFiles 10](#_Toc405203362)

[1.16 GenericFasta 11](#_Toc405203363)

[1.17 AddGenomeCoordinatesForPeptides 11](#_Toc405203364)

[1.18 AddRetentionTimeToMzid 11](#_Toc405203365)

[1.19 CombineFastaFiles 12](#_Toc405203366)

[1.20 XtandemPercolator 12](#_Toc405203367)

[1.21 OmssaPercolator 13](#_Toc405203368)

[1.22 MsgfPercolator 13](#_Toc405203369)

[2 Common errors when using the library 14](#_Toc405203370)

# Overview

The library can be used either in command line mode or in GUI (visualized) mode; both are available to download from (<https://code.google.com/p/mzidentml-lib/>).

In the following section, we describe the purpose of each tool in the library, the input and output file types and the mandatory and optional parameters. Optional parameters are denoted by square brackets.

To run the mzidLibrary from the command line, the following command structure and ordering **must** be used:

java -jar "path\_to\_jar\mzidentml-lib.jar" FUNCTION INPUT\_FILE OUTPUT\_FILE PARAMS

The function name, input file path, output file path and each parameter may be enclosed with double quotes. Double quotes are essential around the file paths if they contain a space, such as:

“C:\Windows\Documents and settings\jonesar\myfilename.mzid”

***Please note – the input file name and output file name MUST be different otherwise an error will result.***

To run the mzidLibrary in the GUI interface, the TOOL\_NAME, INPUT\_FILE and OUTPUT\_FILE are selected as shown in Figure 1, and the parameters need to be entered manually in the “Options” text box.

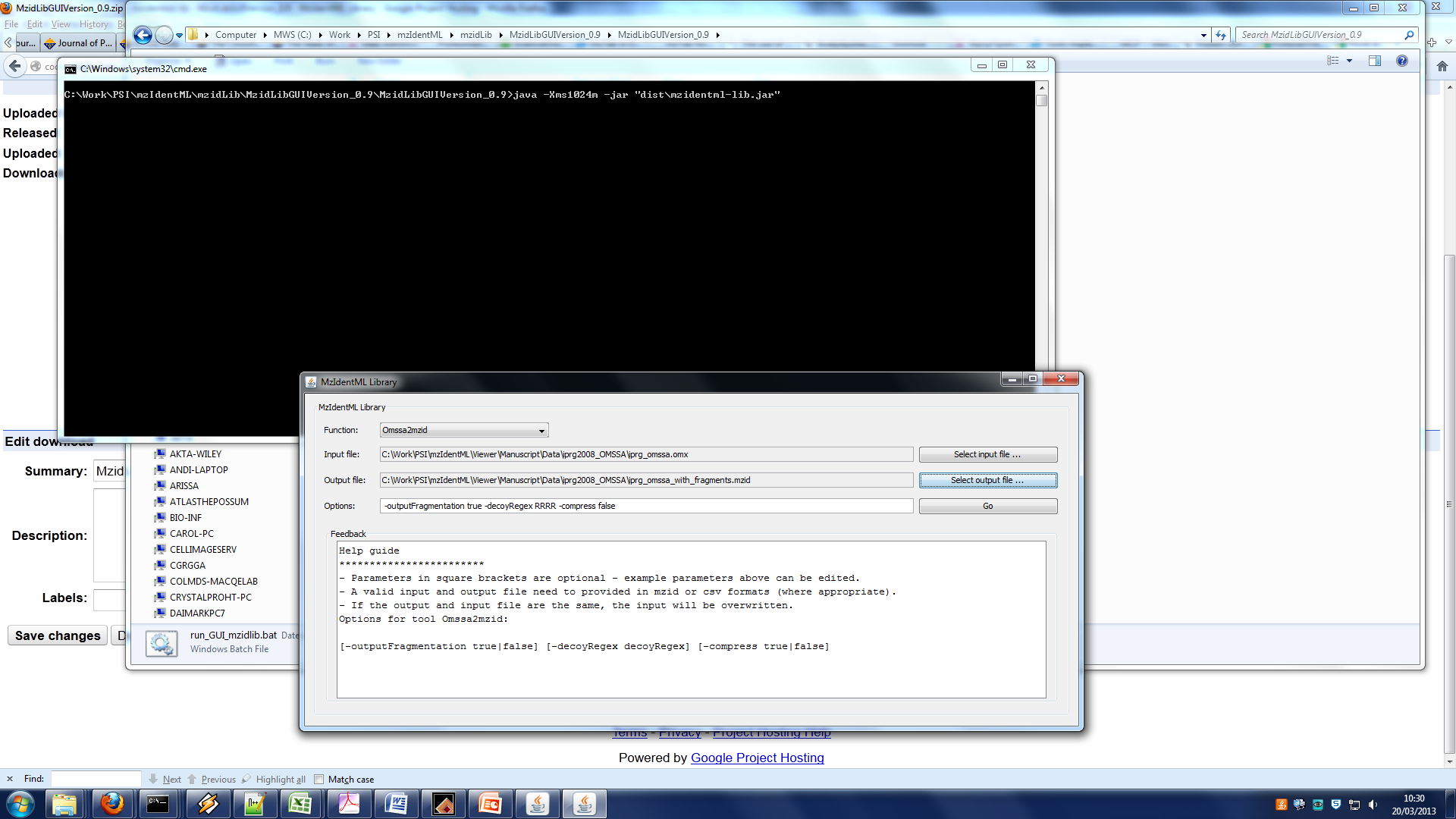


Figure 1 The graphical interface for the mzidLibrary. The user should select the tool (Function), the input file and output file as shown. The structure of the parameters entered under “Options” is the same as for the command line mode.

To run the MzidLib GUI use the following command:

java -Xms1024m -cp "E:\Lib\trunk\dist\mzidentml-lib.jar" uk.ac.liv.mzidlib.gui.LibGui

# 

## FalseDiscoveryRate

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -decoyRegex decoyRegex -decoyValue decoyToTargetRatio -cvTerm cvTerm -betterScoresAreLower true|false [-compress true|false] |
| **Description** | This tool can be used for mzid files in which a decoy database search has been performed to calculate three new data types for each PSM: Local FDR, Q-value and FDRScore [PMID: 19253293] and assign these to every PSM with new CV terms. You must specify the score you wish to order by, using -cvTerm [MS:XXXX] sourced from the PSI-MS CV and whether the scores are ordered low to high or vice versa. |

Example options for GUI:

-decoyRegex Rev\_ -decoyValue 1 -cvTerm MS:1001172 -betterScoresAreLower true -compress true

Example command line:

java -jar "mzidentml-lib.jar" FalseDiscoveryRate mydata.mzid mydata\_fdr.mzid.gz -decoyRegex Rev\_ -decoyValue 1 -cvTerm MS:1001172 -betterScoresAreLower true -compress true

## FalseDiscoveryRateGlobal

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -decoyValue decoyToTargetRatio -decoyRegex decoyRegex -cvTerm cvTerm -betterScoresAreLower true|false -fdrLevel fdrLevel -proteinLevel proteinLevel [-compress true|false] |
| **Description** | The Global FDR module calculates the FDR on one of the three levels. 1) PSM, 2) Peptide, 3) ProteinGroup. If ProteinGroup is chosen, there are two options for protein level PAG or PDH. |

Example options for GUI:

-decoyValue 1 -decoyRegex REVERSED -cvTerm MS:1002125 -betterScoresAreLower true -fdrLevel Peptide -proteinLevel PAG -compress false

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" FalseDiscoveryRateGlobal E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined.mzid E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide.mzid -decoyValue 1 -decoyRegex REVERSED -cvTerm MS:1002125 -betterScoresAreLower true -fdrLevel Peptide -proteinLevel PAG -compress false

## Omssa2mzid

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input]. omx or [input]. omx.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | [-outputFragmentation true|false] -decoyRegex decoyRegex [-omssaModsFile pathToLocalOmssaModsFile] [-userModsFile pathToLocalUserModsFile] [-compress true|false] |
| **Description** | This tool converts OMSSA omx (XML) files into mzid. It has optional parameters for inserting fragment ions into mzid (much larger files). If a decoy Regex is specified, the mzid attribute isDecoy will be set correctly for peptides. No protein inference is done by this tool (no protein list produced). To make valid mzid output, OMSSA must have been run with the option “-w” include spectra and search params in search results. Without this option, search parameters cannot be extracted from OMSSA. In this case, the OMSSA CSV converter should be used. |

Example options for GUI:

-outputFragmentation false -decoyRegex Rev\_ -compress true

Example command line:

java -jar "mzidentml-lib.jar" Omssa2mzid mydata.omx mydata\_omssa.mzid.gz -outputFragmentation false -decoyRegex Rev\_ -compress true

## Tandem2mzid

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input]. xml or [input]. xml.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | [-outputFragmentation (true|false)] [-decoyRegex decoyRegex] [-databaseFileFormatID (e.g. MS:1001348 is FASTA format) "MS:100XXX"] [-massSpecFileFormatID (e.g. MS:1001062 is MGF) "MS:100XXX"] [-idsStartAtZero (true for mzML searched, false otherwise) [true|false]] [-compress true|false] |
| **Description** | This tool converts X!Tandem XML results files into mzid. There are several optional parameters: whether to export fragment ions (makes bigger files), and include a decoy regular expression to set the isDecoy attribute in mzid. Valid mzid files require several pieces of metadata that are difficult to extract from mzid files, the format of the database searched and the file format of the input spectra. If these parameters are not set, the converter attempts to guess these based on the file extension. In X!Tandem, the numbering of spectra differs dependent upon the input spectra type - the IDs start at zero for mzML files, the IDs start at one for other spectra types e.g. MGF. This is a command line parameter which should be set to make sure that the mzid file references the correct spectrum in the source spectrum file. |

Example options for GUI:

-outputFragmentation false -decoyRegex Rev\_ -databaseFileFormatID MS:1001348 -massSpecFileFormatID MS:1001062 -idsStartAtZero false -compress true

Example command line:

java -jar "mzidentml-lib.jar" Tandem2mzid mydata.xml mydata\_tandem.mzid.gz -outputFragmentation false -decoyRegex Rev\_ -databaseFileFormatID MS:1001348 -massSpecFileFormatID MS:1001062 -idsStartAtZero false -compress true

## Csv2mzid

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].csv or [input].csv.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -paramsFile paramsFileLocation -cvAccessionForPSMOrdering (e.g. "MS:1001328" is OMSSA:evalue) [-applyFixedMods true|false] [-decoyRegex decoyRegex] [-compress true|false] |
| **Description** | This tool is intended for converting OMSSA CSV output in mzid. Since the CSV format does not contain any search metadata, this tool requires a separate parameters file containing the search metadata (see example under example\_files and Figure 2). Developers can easily adapt this tool to process other types of CSV file into mzid, by altering the file csv\_config\_file.csv under src/resources and re-building. |

Example options for GUI:

-paramsFile example\_files/toxo\_omssa\_params.csv -cvAccessionForPSMOrdering "MS:1001328" -decoyRegex Rev\_ -compress true

Example command line:

java -jar "mzidentml-lib.jar" Csv2mzid mydata.csv mydata\_omssa.mzid.gz -paramsFile example\_files/toxo\_omssa\_params.csv -decoyRegex Rev\_ -decoyValue 1 -cvTerm MS:1001172 -betterScoresAreLower true -compress true



Figure 2 – The format of the search metadata parameters file required by the Csv2mzid function.

## Mzid2Csv

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].csv or [output].csv.gz |
| **PARAMS** | -exportType exportProteinGroups|exportPSMs|exportProteinsOnly|exportRepProteinPerPAGOnly|exportProteoAnnotator [-verboseOutput true|false] [-compress true|false] |
| **Description** | This tool can export from an mzid file into CSV, according to one of the four types of export specified as parameters. |

Example options for GUI:

-exportType exportPSMs -verboseOutput false -compress true

Example command line:

java -jar "mzidentml-lib.jar" Mzid2Csv mydata\_fdr.mzid.gz mydata.csv -verboseOutput false -compress false

## Threshold

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -isPSMThreshold true|false -cvAccessionForScoreThreshold "MS:100XXX" -threshValue doubleValue -betterScoresAreLower true|false -deleteUnderThreshold true|false [-compress true|false] |
| **Description** | This tool can be used to set the passThreshold parameter for PSMs or proteins in an mzid file, to indicate high-quality identifications that will be used by another tool. It can handle any type of score (sourced from the PSI-MS CV) and scores can be ordered low to high or vice versa. If deleteUnderThreshold is specified, PSMs and referenced proteins under the threshold will be removed from the file. |

Example options for GUI:

-isPSMThreshold true -cvAccessionForScoreThreshold "MS:1002125" -threshValue 0.01 -betterScoresAreLower false -deleteUnderThreshold true -compress true

Example command line:

java -jar "mzidentml-lib.jar" Threshold mydata\_fdr.mzid.gz mydata\_fdr\_threshold.mzid.gz -isPSMThreshold true -cvAccessionForScoreThreshold "MS:1002125" -threshValue 0.01 -betterScoresAreLower false -deleteUnderThreshold true -compress true

## ProteoGrouper

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -requireSIIsToPassThreshold true|false -cvAccForSIIScore cvAccForSIIScore -logTransScore true|false -verboseOutput true|false [-version1\_1 true|false] [-useProteoAnnotator true|false] [-compress true|false] |
| **Description** | This tool can perform sequence-based protein inference, based on a set of PSMs. It should be parameterized with the CV accession for the PSM score used to create a protein score. The tool also needs to know whether the score should be log transformed (true for e/p-values etc) to create a positive protein score. |

Example options for GUI:

-requireSIIsToPassThreshold true -verboseOutput false -cvAccForSIIScore \"MS:1001171\" -logTransScore false -version1\_1 true -useProteoAnnotator false -compress true

Example command line:

java -jar "mzidentml-lib.jar" ProteoGrouper mydata\_fdr\_threshold.mzid.gz mydata\_fdr\_threshold \_groups.mzid.gz -requireSIIsToPassThreshold true -verboseOutput false -cvAccForSIIScore \"MS:1001171\" -logTransScore false -version1\_1 true -compress true

## InsertMetaDataFromFasta

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -fastaFile fastaFilelocation -accessionSplitRegex (regularExpressionTosplitAccSurroundByForwardSlashes e.g. "/ /") -compress true|false |
| **Description** | This tool can be used to extract the description line from a given FASTA file and insert into an mzid file. The tool needs a regular expression to split the accession from the description line in the FASTA file. |

Example options for GUI:

-fastaFile example\_files/TgondiiME49\_ToxoDB-6\_2.fasta -accessionSplitRegex "/ /" -compress true

Example command line:

java -jar "mzidentml-lib.jar" InsertMetaDataFromFasta mydata.mzid mydata\_annotated.mzid -fastaFile example\_files/TgondiiME49\_ToxoDB-6\_2.fasta -accessionSplitRegex "/ /" -compress false

## AddEmpaiToMzid

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -fastaFile fastaFilelocation -accessionSplitRegex (regularExpressionTosplitAccSurroundByForwardSlashes e.g. "/ /") [-enzymeRegex (enzymeRegex defaults to trypsin: "(?<=[KR])(?!P)" )] [-verboseOutput true|false] [-compress true|false] |
| **Description** | This tool applies the emPAI protocol to mzid files [PMID 15958392]. The tool will only work if protein inference has already been performed and there is a protein list in the file. It requires the location of the fasta file searches, a regular expression (such as a space) to split the accessions from the description lines and optionally a regular expression of the enzyme used in the search (if not trypsin). |

Example options for GUI:

-fastaFile example\_files/TgondiiME49\_ToxoDB-6\_2.fasta -accessionSplitRegex "/ /" -verboseOutput false -compress true

Example command line:

java -jar "mzidentml-lib.jar" AddEmpaiToMzid mydata\_fdr\_threshold\_groups.mzid.gz mydata\_fdr\_threshold\_groups\_empai.mzid.gz -fastaFile example\_files/TgondiiME49\_ToxoDB-6\_2.fasta -accessionSplitRegex "/ /" -verboseOutput false -compress true

## CombineSearchEngines

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output].mzid or [output].mzid.gz |
| **PARAMS** | -firstFile firstFile -firstcvTerm firstcvTerm -firstbetterScoresAreLower firstbetterScoresAreLower -secondFile secondFile -secondcvTerm secondcvTerm -secondbetterScoresAreLower secondbetterScoresAreLower -thirdFile thirdFile -thirdcvTerm thirdcvTerm -thirdbetterScoresAreLower thirdbetterScoresAreLower -rank rank -decoyRatio decoyRatio -outputFile outputFile -debugFile debugFile -decoyRegex decoyRegex -compress false |
| **Description** | This tool can be used for combining multiple search engines and can output csv debugging information as well as mzid files. This example for three search engines but can be used for two search engines as well. |

Example options for GUI:

-firstFile iprg\_omssa.mzid -firstcvTerm MS:1001328 -firstbetterScoresAreLower true -secondFile iprg\_mascot.mzid -secondcvTerm MS:1001172 -secondbetterScoresAreLower true -thirdFile iprg\_tandem.mzid -thirdcvTerm MS:1001330 -thirdbetterScoresAreLower true -rank 1 -decoyRatio 3 -outputFile iprg\_combined.mzid -debugFile debug.txt -decoyRegex RRRR -compress false

Example command line:

java -jar "mzidentml-lib.jar" CombineSearchEngines -firstFile iprg\_omssa.mzid -firstcvTerm MS:1001328 -firstbetterScoresAreLower true -secondFile iprg\_mascot.mzid -secondcvTerm MS:1001172 -secondbetterScoresAreLower true -thirdFile iprg\_tandem.mzid -thirdcvTerm MS:1001330 -thirdbetterScoresAreLower true -rank 1 -decoyRatio 3 -outputFile iprg\_combined.mzid -debugFile debug.txt -decoyRegex RRRR -compress false

## CreateRestrictedFASTADatabase

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. fasta or [output]. fasta.gz |
| **PARAMS** | Parameters: [-compress true|false] |
| **Description** | It reads all PDHs with passthreshold=true and create a new FASTA file from these, assuming that InsertMetaDataFromFasta has already been run to insert sequences and descriptions into the file. This routine is intended for use with a two-stage search procedure, creating a new database to be searched in stage 2, based on the confidently identified proteins. |

Example options for GUI:

-compress true

Example command line:

java -jar "mzidentml-lib.jar" CreateRestrictedFASTADatabase mydata\_fdr\_threshold\_groups.mzid.gz mydata\_fdr\_threshold\_groups.fasta -compress false

## MzIdentMLToMzTab

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. mztab or [output]. mztab.gz |
| **PARAMS** | Parameters: [-compress true|false] |
| **Description** | Convert mzidentml file to mztab file. |

Example options for GUI:

-compress true

Example command line:

java -jar "mzidentml-lib.jar" MzIdentMLToMzTab 55merge\_mascot\_plus\_fragment\_ions.mzid 55merge\_mascot\_plus\_fragment\_ions.mzTab -compress false

## RescoreMods

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -cvAccForScoreToAdapt \"MS:100XXXX\" -logTransformPSMScore true|false -commonModificationWeight X -mediumModificationWeight X -rareModificationWeight X -generalModificationWeight X -pairedModificationAndUnmodWeight X -multipleVariableModWeight X -compress true|false |
| **Description** | This tool is experimental at this stage, and hasn't been demonstrated to work effectively yet. This tool re-scores modifications identified by the search engine. It creates a new \"PTM score\" for all PSMs identifying modifications, by multiplying a user entered (e-value value) score type (low values are better), by various weighting factors depending on the modification classification. |

Example options for GUI:

-cvAccForScoreToAdapt MS:1002355 -commonModificationWeight 0.5 -mediumModificationWeight 1.0 -rareModificationWeight 2.0 -generalModificationWeight 5.0 -pairedModificationAndUnmodWeight 0.5 -multipleVariableModWeight 4.0 -compress false

Example command line:

java -jar -Xms16G -jar mzid/mzidentml-lib.jar RescoreMods VEG\_long\_and\_short\_combined\_datasets\_fdr.mzid VEG\_long\_and\_short\_combined\_datasets\_fdr\_rescored.mzid -cvAccForScoreToAdapt MS:1002355 -commonModificationWeight 0.5 -mediumModificationWeight 1.0 -rareModificationWeight 2.0 -generalModificationWeight 5.0 -pairedModificationAndUnmodWeight 0.5 -multipleVariableModWeight 4.0 -compress false

## CombinePSMMzidFiles

|  |  |
| --- | --- |
| **INPUT** | A folder of mzid files to be combined |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -combineFractions ture|false [-compress true|false] |
| **Description** | Combine multiple mzid files on PSM level. Protein-level results (if any) will be removed from the final output |

Example options for GUI:

-combineFractions true -compress false

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" CombinePSMMzidFiles C:\Users\fghali\Desktop\combine C:\Users\fghali\Desktop\combine\combine\combine.mzid -combineFractions true -compress false

## GenericFasta

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].fasta or [input]. fasta.gz |
| **OUTPUT\_FILE** | [output]. fasta or [output]. fasta.gz |
| **PARAMS** | -accession\_regex accession\_regex [-inputGff inputGff] [-compress true|false] |
| **Description** | Create a generic Fasta file to be used as an input for SearchGUI. |

Example options for GUI:

-accession\_regex \\S+ -inputGff inputGff.gff -compress true

Example command line:

java -Xms1200m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" GenericFasta E:\proteogalaxy\pipeline\Data\fasta\ToxoDB-9.0\_TgondiiME49\_AnnotatedProteins.fasta E:\proteogalaxy\pipeline\Data\fasta\ToxoDB-9.0\_TgondiiME49\_AnnotatedProteins\_generice.fasta -accession\_regex \\S+ -inputGff inputGff.gff -compress true

## AddGenomeCoordinatesForPeptides

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -inputGff input.gff –outputGff output.gff -compress true |
| **Description** | Add genome coordinates for peptides from the modified fasta file to the mzid file . |

Example options for GUI:

-inputGff input.gff –outputGff output.gff -compress true

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" AddGenomeCoordinatesForPeptides E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide\_threshold.mzid E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide\_threshold\_mappedGff.mzid -inputGff input.gff –outputGff output.gff -compress true

## AddRetentionTimeToMzid

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -compress true|false |
| **Description** | Add Retention Time to Mzid |

Example options for GUI:

-compress true|false

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" AddRetentionTimeToMzid E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide\_threshold.mzid E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide\_threshold\_mappedGff.mzid -compress false

## CombineFastaFiles

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].fasta |
| **OUTPUT\_FILE** | [output]. fasta |
| **PARAMS** | -compress true|false |
| **Description** | Add genome coordinates for peptides from the modified fasta file to the mzid file . |

Example options for GUI:

-inputFasta E:\proteogalaxy\pipeline\Data\fasta\ToxoDB-9.0\_TgondiiME49\_AnnotatedProteins\_generice\_concatenated\_target\_decoy\_gffMapped.fasta -compress false

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" CombineFastaFiles 1.fasta;2.fasta output.fasta -compress false

## XtandemPercolator

|  |  |
| --- | --- |
| **INPUT\_FILE** | Input.xml |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -decoyRegex decoyregex -compress true|false |
| **Description** | Running XtandemPercolator |

Example options for GUI:

-decoyRegex decoyregex -compress true

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" input.xml output.mzid XtandemPercolator -decoyRegex decoyregex -compress true

## OmssaPercolator

|  |  |
| --- | --- |
| **INPUT\_FILE** | Input.omx |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -database database -decoyRegex decoyregex -compress true|false |
| **Description** | Running OmssaPercolator |

Example options for GUI:

-database database -decoyRegex decoyregex -compress true

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" OmssaPercolator input.omx output.mzid -database database -decoyRegex decoyregex -compress true

## MsgfPercolator

|  |  |
| --- | --- |
| **INPUT\_FILE** | [input].mzid or [input].mzid.gz |
| **OUTPUT\_FILE** | [output]. mzid or [output]. mzid.gz |
| **PARAMS** | -decoyRegex decoyregex -compress true|false |
| **Description** | Add genome coordinates for peptides from the modified fasta file to the mzid file . |

Example options for GUI:

-decoyRegex decoyregex -compress true

Example command line:

java -Xms1024m -jar "E:\Lib\trunk\dist\mzidentml-lib.jar" MsgfPercolator E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide\_threshold.mzid E:\proteogalaxy\pipeline\output\Orbitap-WLysate\mzid\CombineSearchEngines\combined\_fdr\_peptide\_threshold\_mappedGff.mzid -decoyRegex decoyregex -compress true|false

# Common errors when using the library

The most common problems result from the following:

* The input file or output file is not in the correct format. All mzIdentML files must be valid with respect to version 1.1. This can be checked using the mzidValidator (<http://code.google.com/p/psi-pi/downloads/list>)
* The parameters are incorrectly specified, please check these carefully against the instructions before submitting a bug report.
* An invalid CV terms is given. Please check that the accession for the CV term(s) you provide are valid (sourced from <http://psidev.cvs.sourceforge.net/viewvc/psidev/psi/psi-ms/mzML/controlledVocabulary/psi-ms.obo>) and they are present in your file.
  + For example, several routines require you to identify the score you wish to use for ordering results, such as an e-value, p-value or score. If you are unsure what is present in your file, open it with a text editor such as Notepad++ and see which CV terms are present within each <SpectrumIdentificationItem> element (representing each peptide-spectrum match).
* Out of memory errors. You may need to increase the memory available to Java by editing the command for the routine or the graphical interface. Example providing a large amount of memory (3072 MB):
  + java –Xms3072m –jar "mzidentml-lib.jar"

For Java developers using the library in other software, we have assembled an online collection of common errors, which will evolve over time:

<http://code.google.com/p/mzidentml-lib/wiki/CommonErrors>

If you believe you have identified a bug in the library, or you cannot solve an error yourself. Please post on the issues list, supplying a full description of your problem, the command you used, and the input file(s).gz (zipped): <http://code.google.com/p/mzidentml-lib/issues/list>