Chapter 23

Data Conversion with ProteoWizard msConvert

Ravali Adusumilli and Parag Mallick

Abstract

Recent advances in proteome informatics have led to an explosion in tools to analyze mass spectrometry data. These tools operate across the analysis pipeline doing everything from assessing quality control to matching peptides to spectra to quantification. Unfortunately, the vast majority of these tools are not able to operate directly on the proprietary formats generated by the diverse mass spectrometers. Consequently, the first step in many protocols is the conversion of data from vendor-specific binary files to open-format files. This protocol details the use of ProteoWizard's msConvert and msConvertGUI software for this conversion, taking format features, coding options, and vendor particularities into account. We specifically describe the various options available when doing conversions and the implications of each option.

Key words Proteomics, Proteome informatics, Data conversion, Open formats, mzML, mzXML, ProteoWizard

1 Introduction

Mass-spectrometry-based proteomics has become an important component of biological research. Numerous proteomics methods have been developed to identify and quantify the proteins in biological and clinical samples, identify pathways affected by endogenous and exogenous perturbations, and characterize protein complexes. However, before it is possible to explore any of these important biological questions, a battery of computational tools is required to convert mass spectrometry data into knowledge. Each vendor has developed its own ecosystem of tools, such as ProteomeDiscoverer (Thermo) and Analyst (SCIEX). to help interpret the complex data produced by their instruments. However, the proteome informatics research community has developed thousands of alternate tools with a wide range of capabilities. Unfortunately, the vast majority of these tools are not able to directly read vendors' binary formats and instead are dependent upon "open" formats such as mzML and mzXML. Translating from closed binary formats to open formats requires specialized

Lucio Comai et al. (eds.), *Proteomics: Methods and Protocols*, Methods in Molecular Biology, vol. 1550,
DOI 10.1007/978-1-4939-6747-6_23,

software. Initially a set of tools that specifically converted data from one vendor format into an open format (e.g., ReAdW) emerged. More recently, the ProteoWizard toolkit [1, 2] has provided a swiss-army knife tool in msConvert that is able to convert all popular vendor formats into diverse open formats.

Beyond performing a direct translation of file contents, from a closed binary format to an openly readable format, conversion is also a data processing step. In this step, researchers may perform lossy data compressions (such as centroiding) or fundamentally alter the values in files through recalibration (such as in mzRefinery). Here we describe a variety of protocols for data conversion and particularly detail all the diverse options available to users during conversion.

The msConvert tool described below is available as user-friendly graphical user interface (GUI) program (msConvertGUI) for Windows and as cross-platform (Windows, Mac, Linux) command line tool (msConvert). We note that there are more options available in the command line tool than in the GUI. MS Convert offers the flexibility of processing multiple input file formats with options for several filters and transformations. The details of the filters and transformations available for msConvertGUI and msconvert command line will be further discussed below. In addition to converting from vendor formats to open formats, msConvert and msConvertGUI can also convert among open formats, such as from mzXML to mzML. Conversion from vendor formats is only achievable on Windows. However, conversion among open formats can be performed on any platform.

2 Materials

2.1 Installation of ProteoWizard (Includes Both msConvert, msConvertGUI and msPicture):

1. Make sure you have installed Microsoft .NET Framework **3.5** SP1 and **4.0**.
   (a) MS .NET Framework 3.5 SP1 can be found at: http://www.microsoft.com/en-us/download/details.aspx?id=22
   (b) MS .NET Framework 4 can be found at: http://www.microsoft.com/en-us/download/details.aspx?id=17851
2. Download the latest version of proteoWizard from: http://proteowizard.sourceforge.net/
   (a) We typically recommend the Windows 64-bit installer for most users.
   (b) Upon download, double-click the .msi file to install.
   (c) proteoWizard will typically be installed into a folder 'C:\Program Files\ProteoWizard\ProteoWizard 3.0.DDDD'

3. (For Command Line Use) Add "proteoWizard" folder to the path variable:
   (a) Go to control panel > System and security > System > Advanced settings > Environment Variables
   (b) You should see a table/list of variables and values. Double click on "Path".
   (c) You should see a bunch of values separated by ";" in the Variable value field.
   (d) Add your proteowizard location to the list separating with a semicolon.
   For example: "C:\Windows; C:\Program Files\ProteoWizard 3.0.8708\"
   (e) Restart Windows.

3 Methods

3.1 Using msConvertGUI to Convert the File Format of Data Files

1. Navigate to the proteowizard directory (Fig. 1) (For ex: C:\Program Files\ProteoWizard\ProteoWizard 3.0.8708)
2. Start the msConvertGUI application
   Double-click on "msConvertGUI" application to open. This should open the msconvert GUI application. Visually, msconvertGUI is divided into four boxes (Fig. 2), which are described in greater detail below.
   (a) Select input files and output files directory location (**Top left**).

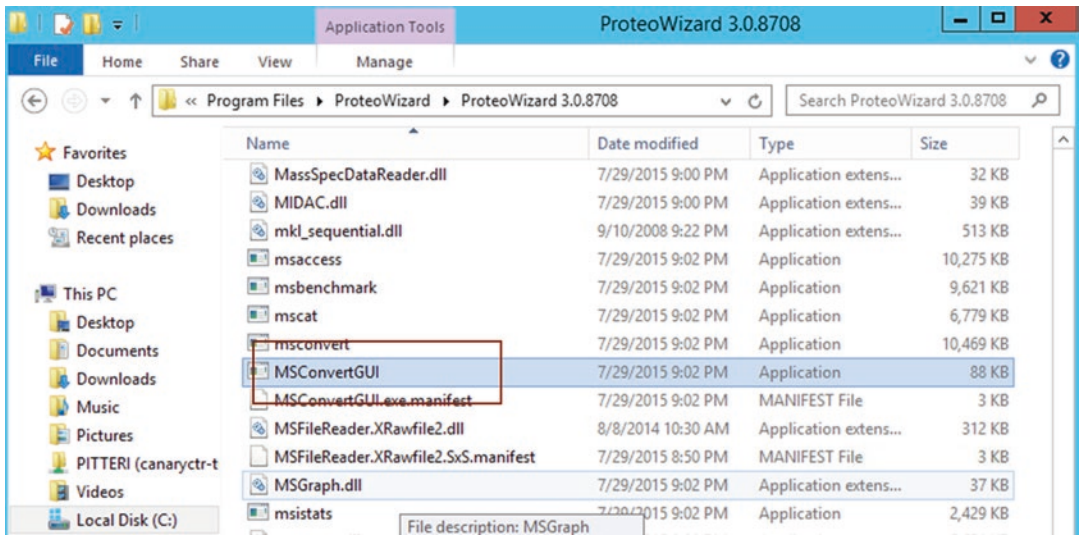

Fig. 1 Open from the folder

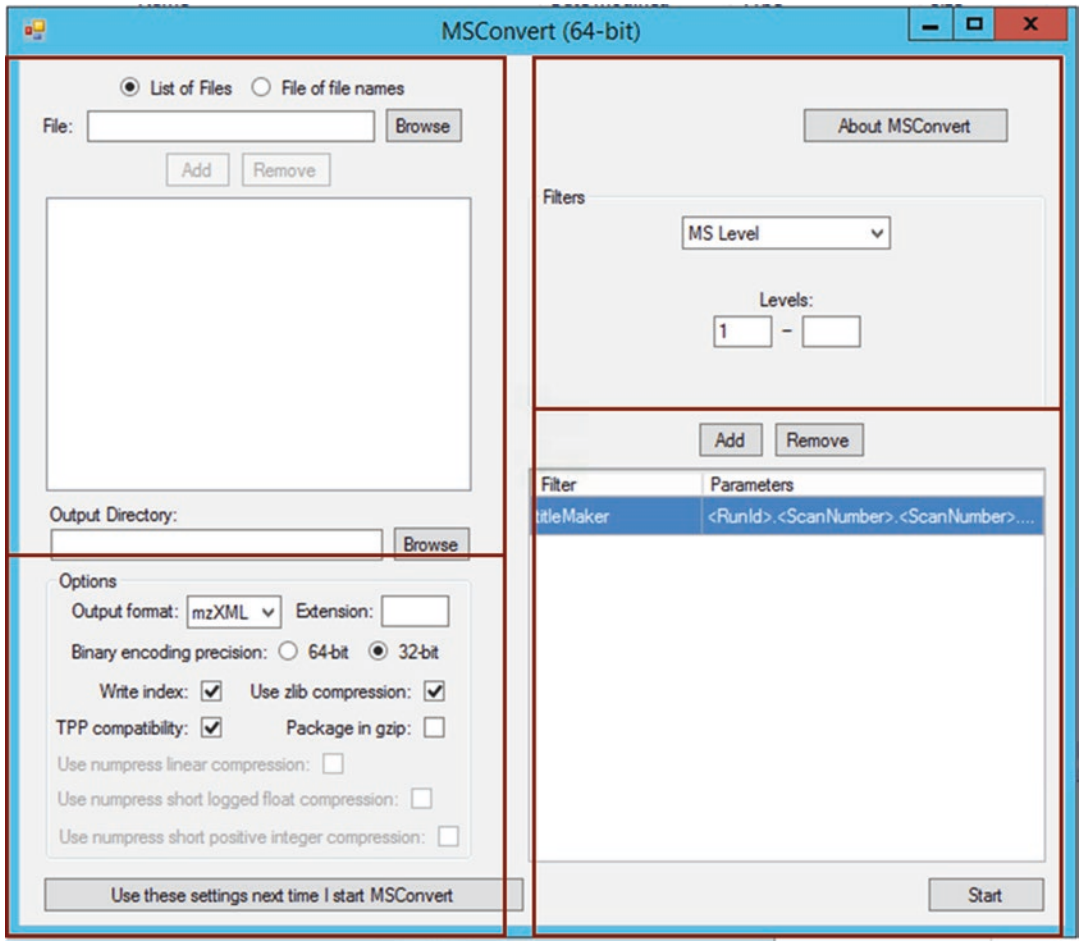

Fig. 2 MSConvertGUI

(b) **Bottom left**: Options—to select the output format, extension, binary encoding precision, write index, use zlib compression, TPP compatibility, package in gzip, numpress compression types (linear, short logged float, short positive integer).

(c) **Top Right**: Filters—Type of filters and filtering levels/options to add or remove.

(d) **Bottom Right**: Box to view the selected filters in **step c**.

3. Choosing files to convert (top left, Figs. 3 and 4):

Type: radio button

Default: List of Files

Select the file import type (list of files or file of filenames)—further explained below.

(a) *List of Files:*

Type: Button ("Browse")

Additional types: Button ("Add" and "Remove")

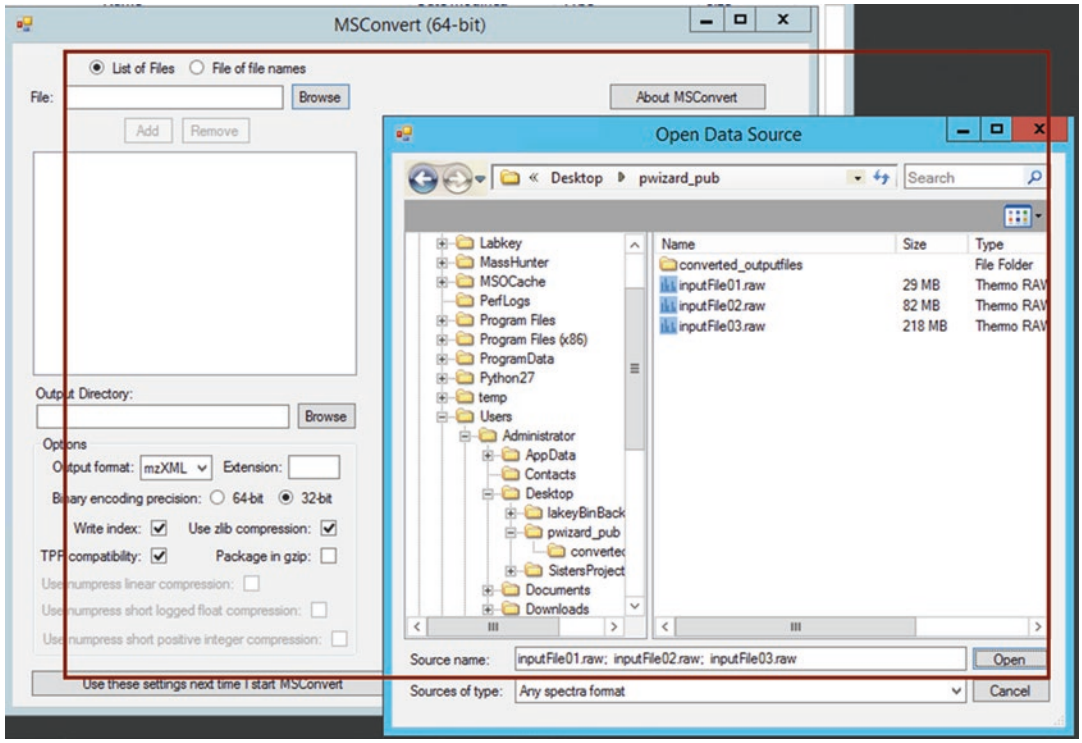

Fig. 3 Click on Browse and select input files from MSConvertGUI

i. Select the "List of Files" option on top-left of the msconvert GUI.

ii. Click on the "Browse" button-this should open a desktop browser window to select the input files.

iii. Go to the directory with the input formatted (*.RAW files) from the browser window in **step ii**.

iv. Select one input file or multiple files by holding down the shift/control button on the keyboard.

v. Click "open"—the selected files should now be visible in the msconvert GUI (in the box right below the "List of Files" and "File of filenames" options.

vi. To remove some files after adding them to the GUI, select the file/files to remove in the box from **step v**. Click "Remove" right above the box with the list of files.

(b) *File of file names:*

Type: Button ("Browse")

Additional types: Button ("Add" and "Remove")

Input files to be converted can be imported from a text file with input file names and locations.

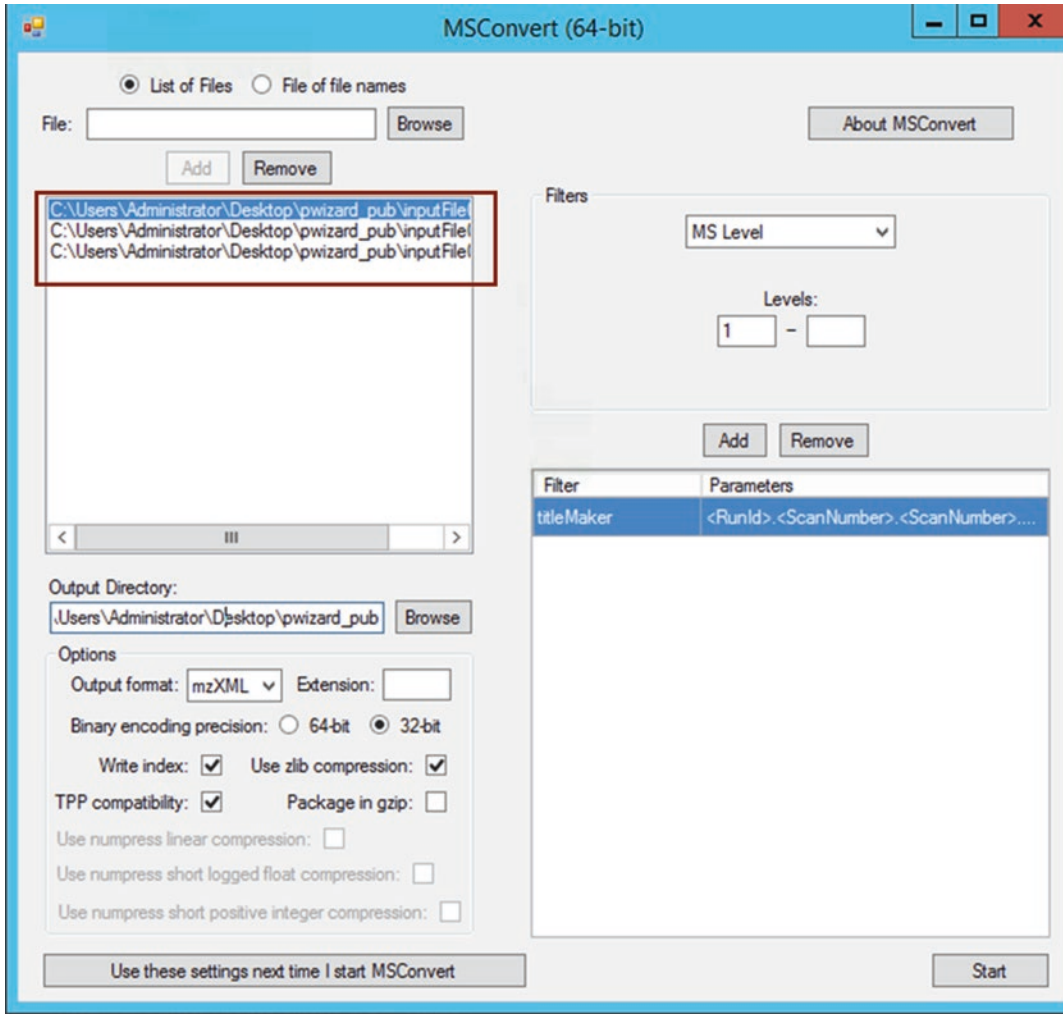

Fig. 4 MSConvertGUI list of selected files

i. Select the "File of File names" option on top-left of the msconvert GUI.

ii. Click on the "Browse" button-this should open a desktop browser window to select the input files.

iii. Go to the folder containing the text file with "input filename and paths" to be converted.

iv. Select the text file/files with "input filename and paths" to be converted.

v. Click "open".

The files to be imported (selected in either **step a** or **b**) can be viewed in **step 2**.

4. Select an Output Directory *(*top left*):*

   Type: Button ("Browse")

   Default: Folder path with the input files

   (a) The default output location is set to the directory with the input formatted files selected. If input files are selected across multiple directories, the directory containing the first selected input file is selected.

   (b) To choose an output location other than the default, click on "Browse". This should open a browser window.

   (c) Select an already existing folder from the desktop or create a new folder to save the output files.

   (d) To create new output folder, click on "Make New Folder"—this will create a new folder. Type in the output name for the folder (For Example: "Project1_output")

   (e) Click "OK" to select the folder or "Cancel" to cancel the folder selection.

5. **Select Conversion Options (Fig. 5):**

   (a) Output format:

   Type: Drop down list

   Default: mzXML

   Select the format of the converted output file. The following output formats can be selected:

   i. mzXML (default)
   ii. mz5
   iii. mgf (Mascot generic format)
   iv. text (Proteowizard internal text format)
   v. ms1
   vi. cms1
   vii. ms2
   viii. cms2

   (b) Extension:

   Type: Text box

   Default: Blank

   To add an extension to the output formatted file other than the option chosen above. For example: mzXML can be saved as .fred.mzXML if needed

   (c) Binary encoding precision:

   Type: radio button

   Default: 32-bit

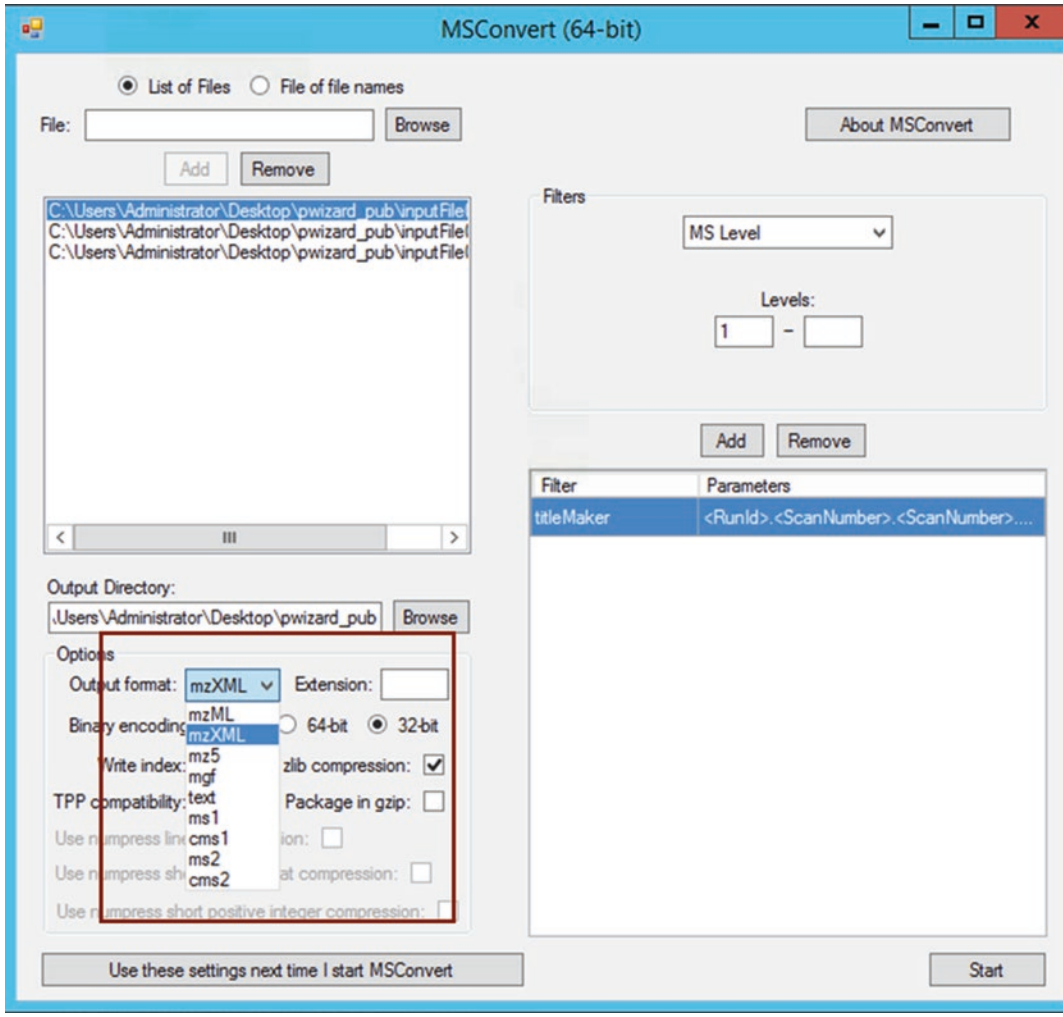

Fig. 5 MSConvertGUI output format options

Select the output file encoding type. The output file can be encoded as 32-bit (default—smaller file size) or 64-bit.

(d) Write index:

Type: Checkbox

Default: Checked

Check this box to write index to the output file (*see* **Note 1**).

(e) Use zlib compression:

Type: Checkbox

Default: Checked

Check this box to "zlib" compress the output file (*see* **Note 2**).

(f) TPP compatibility:

Type: Checkbox

Default: Checked

Check this box to make the output file "TPP compatible."

(g) Package in gzip:

Type: Checkbox

Default: NOT Checked

Check this box to "gzip" the output file (*see* **Note 3**).

6. **Select Filter Options (Figs. 5, 6 and 7)** (*see* **Note 4**)

Filters:

Type: Dropdown list

Default: MS Level

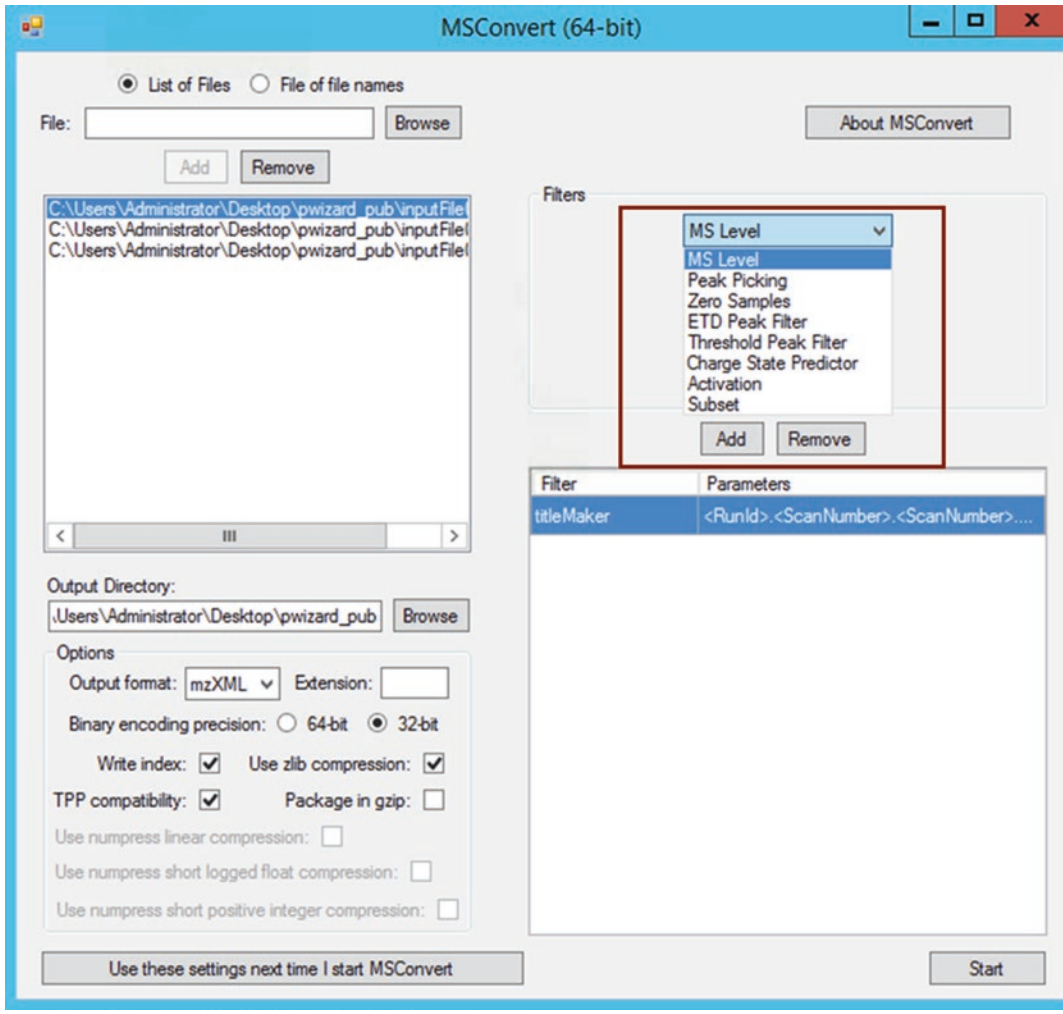

Fig. 6 MSConvertGUI filtering options

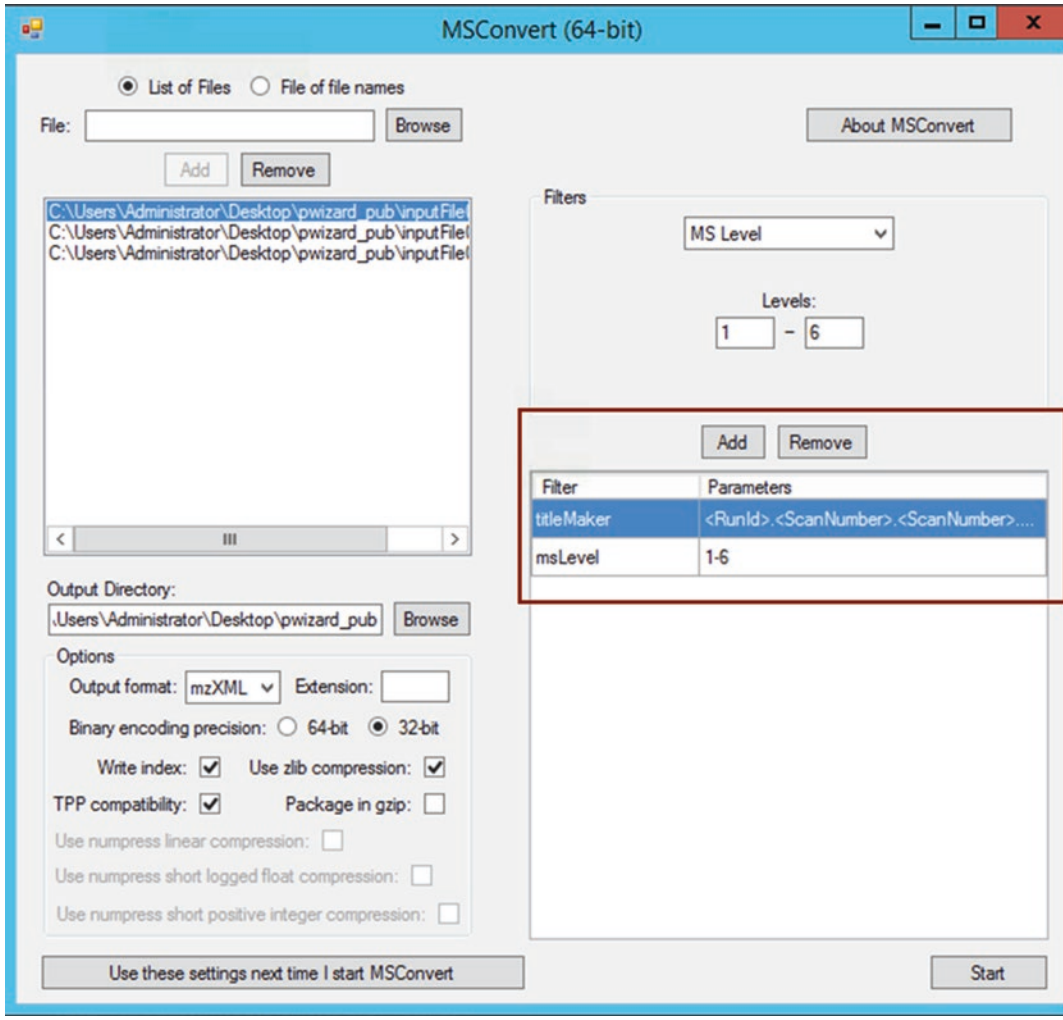

Fig. 7 MSConvertGUI list of selected filters

Select the filter to apply to the output file. The following filters can be selected:

(a) MS Level (Default)

(b) Peak Picking (*see* **Note 5**)

(c) Zero Samples

(d) ETD Peak Filter

(e) Threshold Peak Filter

(f) Charge State Predictor

(g) Activation

(h) Subset

Click "Add" to add the settings selected from **step a** and/or **b**. Alternately, remove the settings added by clicking "Remove". View the selected filters and parameters in the box.

7. **Save Settings.** Click "Use these settings next time I start MSConvert" (bottom left) to set the settings from steps 1 to 10 as defaults so they do not need to be set the next time you run the program.
8. Double-check all the settings. Once all options have been filled click "Start." A new window will appear with a list of files to be converted and current progress. Details on the conversion of the currently selected file will be shown in the text box at the bottom of the new menu. Once all files have finished converting it is safe to close the progress window and use the resulting files

3.2 Using the Command Line Version of msConvert to Convert the File Format of Data Files

1. Make sure the latest version of proteowizard is downloaded and setup as instructed above.
2. Go to command prompt/windows power shell (for windows 8+). If the path variable is set under "**SETUP PROTEOWIZARD"** instructions, all the proteowizard tools can be called without typing the absolute path. For example: *msconvert.exe* can be called directly if path variable is set. If not, *C:\Program Files\ProteoWizard 3.0.8708\msconvert.exe* will be required to call the tool.

   For this protocol, the assumption is that path has been set and so all the tools will be called directly without the absolute path.
3. To view all options under the msConvert commandline, type *msconvert.exe* (or *C:\Program Files\ProteoWizard 3.0.8708\msconvert.exe*) and hit ENTER/return.
   (a) This should show all the options available under msConvert commandline (screenshot attached).
   (b) All Command line usage terms for msconvert can be summarized into four sections:

1. **Usage:** Generic command format to run the msconvert tool
2. **Options:** List of all available options to run msconvert (
   - For example: -f for "text file with input filenames" (like File of file names *in msconvertGUI)*
     1. -o for "output directory "
     2. -c for configuration file
3. **FILTER OPTIONS:** This section lists all the options to filter the data in the output file
   - For example: index (similar to write index on the GUI);
     1. msLevel to select the level of ms data (same as MS Levels in the GUI)
     4. **Examples:** This section list few examples with explanations to run msconvert commandline options.

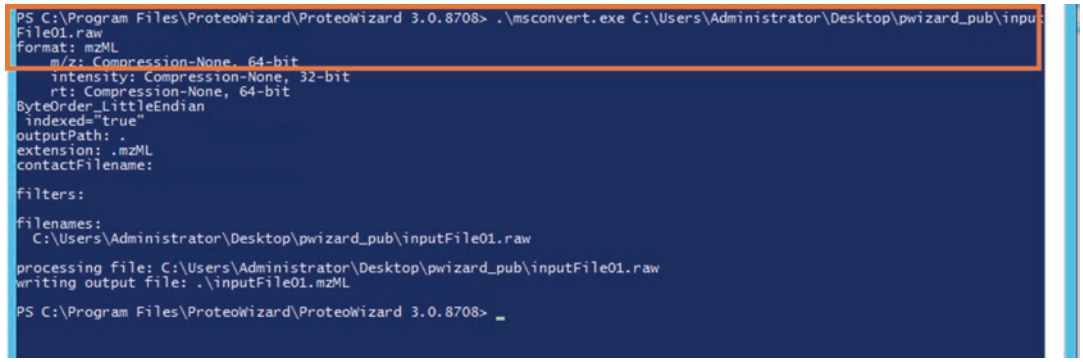

Fig. 8 msconvert—default RAW (to mzML) file conversion from commandline

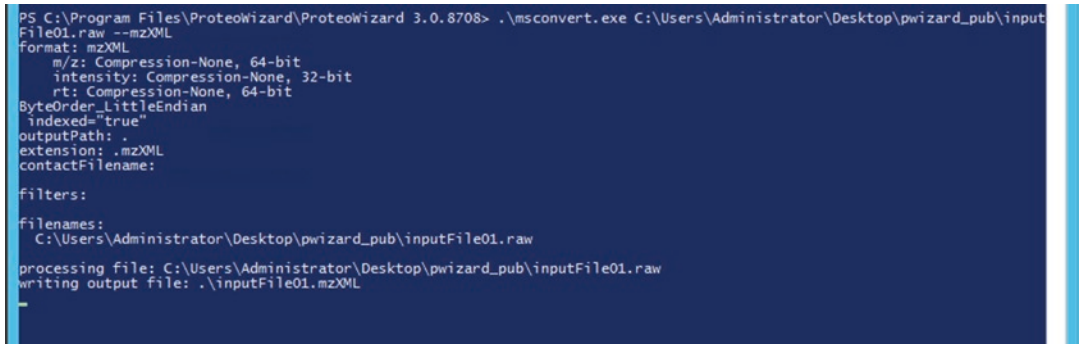

Fig. 9 msconvert—RAW to mzXML file conversion from commandline

4. Execute msConvert (Figs. 8, 9 and 10)

   The simplest way to run msConvert is as:

   msconvert C:\Users\Administrator\Desktop\pwizard_pub\inputFile01.raw

   The above is the most basic command to run a single file on one input *.RAW file. The "*inputFile01.RAW*" in the above command is converted to *"inputFile01.mzML"* formatted file by default. The *"inputFile01.mzML"* file will be located in the current working directory (i.e., the directory from which the command is executed).

   However, msconvert provides the user with flexible options to choose the output conversion format, set the output directory location, set file format extensions, provide configuration file choice to set all the conversion properties together. These options are elaborated in **Note 6**.

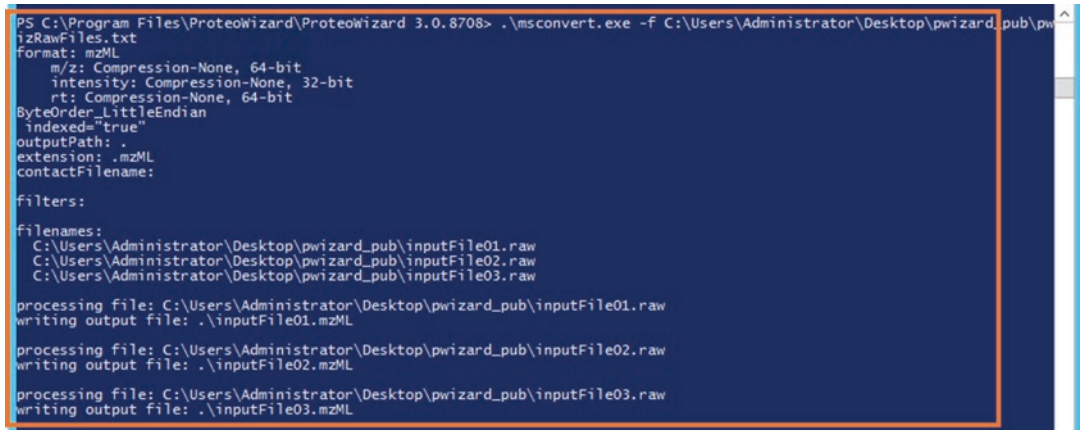

Fig. 10 msconvert commandline using -f (or --filelist) to convert multiple files together

5. Execute msConvert with variations in parameters and filters.
   (a) For each filter to be included (*see* **Note** 7), add text structured like the following: --filter "peakPicking true 1-". The first piece signals that a new data filter is being added. The section in double quotes supplies the name of the filter (see below) and the configuration options for it.
   (b) Specify the output format options, such as specifying the overall format by one of the following flags: "--mzML", "--mzXML", "--mz5", "--mgf". If subsequent software allows it, specify "zip" encoding for data within the files through use of the "-z" option. The full list of these options can be found by running the msConvert executable without any parameters.
   (c) Specify which files are to be converted. Both absolute and relative paths are permitted, and wildcard characters such as '*' and '?' can be used to process multiple files in a single pass.
   (d) Combine these elements into a single command line, such as the following:

```
msconvert.exe --filter "peakPicking
true 1-" --mz5 -z *.raw
```

4 Notes

1. A file index allows for more rapid scan level access of the file. The index itself makes files slightly larger. However, instead of having to read a file from beginning to end, it can now be read from random positions.
2. zlib compression is a seemingly invisible change to the file. Essentially, the ms spectra themselves are zlib compressed prior

to writing the file. This greatly compresses the file, but also can slow down its reading. Furthermore, some readers are unable to handle zlib compressed files.

3. Unlike zlib compression, gzip actually fully gzips the entire file. This shrink files by about ~20–30%. Once gzipped, the files are no longer directly human readable without being ungzipped first. ProteWizard and some other tools are able to read files, even when gzipped. However, they are no longer able to provide random access. The vast majority of open source tools are not able to read gzipped files directly.
4. Filters are operations that are performed on spectra to alter them prior to writing out a converted file. While some filters are simple (e.g., only writing out MS/MS data), others can be quite complex and substantially alter your data. The commandLine msConvert has access to many more filters than are available in the GUI. Please note, that as filters can fundamentally change your data, they may substantially impact downstream operations, such as quantification.
5. Peak Picking, is the msConvert nomenclature for "centroiding" which is a very common procedure for shrinking file sizes. However, this is an irreversible operation that discards a substantial amount of data. It should be used with caution.
6. msConvert has a large number of available options (Fig. 8):
   All the available options are listed on running the command "msconvert" from the command prompt.

Parameter Flag	Explanation
-f [--filelist] arg	Specify text file containing filenames
-o [--outdir] arg (=.)	Set output directory ('-' for stdout) [.]
-c [--config] arg	Configuration file (optionName=value)
--outfile arg	Override the name of output file
-e [--ext] arg	Set extension for output files [mzML|mzXML|mgf|txt|mz5]
--mzML	Write mzML format [default]
--mzXML	Write mzXML format
--mz5	Write mz5 format
--mgf	Write Mascot generic format
--text	Write ProteoWizard internal text format
--ms1	Write MS1 format
--cms1	Write CMS1 format
--ms2	Write MS2 format
--cms2	Write CMS2 format

(continued)

Parameter Flag	Explanation
-v [--verbose]	Display detailed progress information
--64	Set default binary encoding to 64-bit precision [default]
--32	Set default binary encoding to 32-bit precision
--mz64	Encode *m/z* values in 64-bit precision [default]
--mz32	Encode *m/z* values in 32-bit precision
--inten64	Encode intensity values in 64-bit precision
--inten32	Encode intensity values in 32-bit precision [default]
--noindex	Do not write index
-i [--contactInfo] arg	Filename for contact info
-z [--zlib]	Use zlib compression for binary data
--numpressLinear [=arg(=2e-009)]	Use numpress linear prediction compression for binary mz and rt data (relative accuracy loss will not exceed given tolerance arg, unless set to 0)
--numpressPic intensities	Use numpress positive integer compression for binary intensities (absolute accuracy loss will not exceed 0.5)
--numpressSlof [=arg(=0.0002)]	Use numpress short logged float compression for binary intensities (relative accuracy loss will not exceed given tolerance arg, unless set to 0)
-n [--numpressAll]	Same as --numpressLinear --numpressSlof (see https://github.com/fickludd/ms-numpress for more info)
-g [--gzip]	gzip entire output file (adds .gz to filename)
--filter arg	Add a spectrum list filter
--merge level	Create a single output file from multiple input files by merging file-level metadata and concatenating spectrum lists
--simAsSpectra	Write selected ion monitoring as spectra, not chromatograms
--srmAsSpectra chromatograms	Write selected reaction monitoring as spectra, not chromatograms
--combineIonMobilitySpectra	Write all drift bins/scans in a frame/block as one spectrum instead of individual spectra
--acceptZeroLengthSpectra	Some vendor readers have an efficient way of filtering out empty spectra, but ittakes more time to open the file
--ignoreUnknownInstrumentError	If true, if an instrument cannot be determined from a vendor file, it will not be an error
--help	Show this message, with extra detail on filter options

-f or --filelist (Fig. 10)

- The -f or --filelist flag is used in case the user would like to convert multiple input files together.
- For instance, consider a text file containing the filenames of three *.RAW files which need to be converted to *.mzML format.
- Sample text file (pwizRawFiles.txt) with list of filenames and locations:

  C:\Users\Administrator\Desktop\pwizard_pub\inputFile01.raw

  C:\Users\Administrator\Desktop\pwizard_pub\inputFile02.raw

  C:\Users\Administrator\Desktop\pwizard_pub\inputFile03.raw
- Using the *pwizRawFiles.txt* file above with the flag -f, the command should look like:
- msconvert -f C:\Users\Administrator\Desktop\pwizard_pub\pwizRawFiles.txt
- Additional flags can be added to convert the files to different formats as discussed further.
- NOTE: One line inside the *pwizRawFiles.txt* file should have only one input filename. Each input filename in the list should be present in a new line

-o or --outdir

- The flags "-o" or "--outdir" can be used to specify the directory to write the output file
- For example:
- mconvert -o C:\Users\Administrator\Desktop\pwizard_pub\
- Default: Current working directory

-c or --config

- This option provides the user to enter all file conversion properties together on one or more files.
- Sample configuration (msconvert.config) file:
- mzXML=true
- zlib=true
- filter="index [3,7]"
- filter="precursorRecalculation"
- IMPORTANT: DO NOT use spaces before and after "=" inside the configuration file

--outfile

- The --outfile argument lets the user set the name of the output file. If missing, the output filename will be based on the given input filename.

-e or --ext

- This argument can be used to select the extension of the output filename.
- Options available: mzML, mzXML, mgf, txt, mz5
- If none selected, the extension will be based on the output file conversion format (For example: using argument --mzXML with msconvert will result in the <output_filename> .mzXML).
- If no output file conversion is selected, then the default as mentioned earlier will be "mzML"

--mzML

- The output converted file will be ".mzML" format.
- This is the default if no other format is selected

--mzXML

- Very similar to the above "--mzML" format
- The output file will be created as <output_filename> . mzXML

--mz5

- Similar to --mzML and --mzXML options
- Writes the output file to "mz5" format

--64

- The "--64" option is used to set the binary encoding to 64-bit precision for the converted file (same as 64bit precision option on the msconvertGUI)
- Default if no other encoding flags are selected

--32

- Like above, this is used to set the binary encoding.
- As the flag suggests, this option lets us set the output file conversion to 32-bit encoding precision
- The *m/z* and intensity values can also be encoded independently using the --mz23 and --inten64 flags.

--mz64

- The "mz" prefix to the flag "--mz64" refers to *m/z* values This flag along with the following "--mz32" flag below

offers the flexibility of setting the *m/z* values only encoding to 64-bit precision in case the user would like to specify the "intensity" coding separately.

- --mz64 is the default for the *m/z* encoding.

--mz32

- Similar to the "--mz64" option above.
- In case of mz and intensity independent encoding, We can select "--mz32" for 32-bit encoding of "*m/z*" values

--inten64

- The "inten" prefix to the flag "--inten64" refers to intensity values.
- This and the below given "--inten32" flags can be used for encoding intensity values independently.
- --inten64 is the default for encoding the intensity values.

--noindex

- Setting this "--noindex" flag will tell the msconvert command line program NOT to write index to the output file.

-i or --contactInfo

- Provide the msconvert program with contact information in a filename and path by using this flag.

-z or --zlib

- Use zlib compression for binary data by setting this flag. This option is used for data compression.

--numpressLinear

- Use numpress linear prediction lossy compression for binary mz and rt data (relative error guaranteed less than given tolerance)

--numpressPic

- Use numpress positive integer lossy compression for binary intensities (maximum 0.5 absolute error guaranteed)

--numpressSlof

- Use numpress short logged float lossy compression for binary intensities (relative error guaranteed less than given tolerance)

-n or --numpressAll

- Same as --numpressLinear --numpressSlof
- See https://github.com/fickludd/ms-numpress for more info

-g or --gzip

- Use "-g" or "--gzip" option to gzip the entire output file.
- Selecting this option reduces the file size and adds ".gz" to the output filename

--filter

- IMPORTANT: Adding this argument provides the option to add/set filters before writing the data to output file.
- Several options are available for filtering the data. All the available filter types/options are explained in greater detail under the next section C (FILTER OPTIONS).
- Each type of filter mentioned in section C should be preceded by the "--filter" flag.
- For Example:
  - msconvert C:\Users\Administrator\Desktop\pwizard_pub\inputFile01.raw
  - --filter "peakPicking true 1-" --filter "threshold bpi-relative .5 most-intense"

--merge

- IMPORTANT: This argument/flag provides us the option to merge the output of multiple input files into a single output files. This option can be very useful if needed to merge fractions with each fraction having an individual *. RAW file.

--simAsSpectra

- "sim" refers to selected ion-monitoring. This is a useful option available in case the selected ion-monitoring is required as spectra and NOT as chromatograms. This is often used to represent MRM data in mzXML files, which do not natively support chromatogram data.

--srmAsSpectra

- "srm" refers to selected reaction monitoring. Like the "--simAsSpectra" option, the "--srmAsSpectra" flag can be used to write the selected reaction monitoring as sepctra and NOT chromatograms. This is often used to represent MRM data in mzXML files, which do not natively support chromatogram data.

--combineIonMobilitySpectra

- Usually, ion mobility spectra are written individually for each scan. This option can be used to write all drift bins/scans in a frame/block as one spectrum instead of individual spectra.

--acceptZeroLengthSpectra

- This option lets you accept zero length spectra in case the file contains empty spectra.
- Some (but not all) vendor readers have an efficient way of filtering out empty spectra.
- NOTE: Takes more time to open the file.

--ignoreUnknownInstrumentError

- The instrument information should be present in the vendor file usually but in case it is missing, this option can be useful.
- If true, if an instrument cannot be determined from a vendor file, it will not be an error

--help

- Use this option to display all filtering options in detailed

7. The msconvert commandline has additional filtering options when compared to the msConvertGUI.

   For example, msLevel is present in both commandline and msConvertGUI (as MS Levels on GUI). This filter selects only spectra with indicated mslevels (ms level 1). Similarly, the analyzer option in the commandline allows the user to retain the spectra with the specified mass analyzer ("FTMS" or "ITMS") type only.

   Note: Filters are applied sequentially in the order that you list them, and the sequence order can make a large difference in your output. In particular, the peakPicking filter must be first in line if you wish to use the vendor-supplied centroiding algorithms since these use the vendor DLLs, which only operate on raw untransformed data.

Filter Flag	Explanation
index	<index_value_set>
msLevel	<mslevels>
chargeState	<charge_states>
precursorRecalculation	
mzRefiner	mzRefinerinput1
precursorRefine	
peakPicking	[<PickerType>[snr=<minimumsignal-to-noiseratio>] [peakSpace=<minimumpeakspacing>][msLevel=<ms_levels>]]
scanNumber	<scan_numbers>

(continued)

(continued)

Filter Flag	Explanation
scanEvent	<scan_event_set>
scanTime	<scan_time_range>
sortByScanTime	
stripIT	
metadataFixer	
titleMaker	<format_string>
threshold	<type><threshold><orientation>[<mslevels>]
mzWindow	<mzrange>
mzPrecursors	<precursor_mz_list>
defaultArrayLength	<peak_count_range>
zeroSamples	<mode>[<MS_levels>]
mzPresent	<tolerance><type><threshold><orientation> <mz_list>[<include_or_exclude>]
scanSumming	[precursorTol=<precursortolerance>][scanTimeTol=<scantimetolerance>]
MS2Denoise	[<peaks_in_window>[<window_width_Da>[multicharge_fragment_ relaxation]]]
MS2Deisotope	[hi_res[mzTol=<mzTol>]][Poisson[minCharge=<minCharge>] [maxCharge=<maxCharge>]]
ETDFilter	[<removePrecursor>[<removeChargeReduced>[<removeNeutralLoss>[<bl anketRemoval>[<matchingTolerance>]]]]]
chargeStatePredictor	[overrideExistingCharge=<true|false(false)>] [maxMultipleCharge=<int(3)>][minMultipleCharge=<int(2)>][singleC hargeFractionTIC=<real(0
turbocharger	[minCharge=<minCharge>][maxCharge=<maxCharge>] [precursorsBefore=<before>][precursorsAfter=<after>] [halfIsoWidth=<half-widthofisolationwindow>][defaultMinCharge=<def aultMinCharge>][defaultMaxCharge=<defaultMaxCharge>][useVendor Peaks=<useVendorPeaks>]
activation	<precursor_activation_type>
analyzer	<analyzer>
analyzerType	<analyzer>
polarity	<polarity>

- *msLevel*

  This filter selects only spectra with the indicated <mslevels>.

  – The value set to this argument needs to be an integer.

  – For Example:

  ```
  # extract MS1 scans only
  msconvert C:\Users\Administrator\Desktop\pwizard_pub\
      inputFile01.raw
  --filter "msLevel 1"
  # Alternatively, inside the configuration file from the
      msconvert.config file (input as in section 3.2, step 2),
      add the filter for "msLevel 1" as:
  mzXML=true
  zlib=true
  filter="index [3,7]"
  filter="msLevel 1"        #For ms1 scans only
  ```

- *chargeState*

  – This filter keeps spectra that match the listed charge state(s).

  – The value set to this argument needs to be an integer.

  – Both known/single and possible/multiple charge states are tested. Use 0 to include spectra with no charge state at all.

- *precursorRecalculation*

  – This filter recalculates the precursor *m/z* and charge for MS2 spectra.

  – Looks at the prior MS1 scan to better infer the parent mass.

  – Works only on orbitrap and FT data and does not use any third party (vendor DLL) code.

  – ONLY need to use as a flag. For example:

  ```
  # Add this line if using config file
  filter="precursorRecalculation"
  # If using as a flag to the command line parameter, add the
      following to the command:
  --filter "precursorRecalculation"
  ```

- *mzRefiner*

  – This filter recalculates the *m/z* and charges, adjusting precursors for MS2 spectra and spectra masses for MS1 spectra.

  – It uses an ident file with a threshold field and value to calculate the error and will then choose a shifting mechanism to correct masses throughout the file. It only works on Orbitrap, FT, and TOF data. It is designed to work on mzML files.

- *precursorRefine*
  - Similar to *precursorRecalculation*
  - This filter refines the precursor *m/z* and charge for MS2 spectra. It looks at the prior MS1 scan to better infer the parent mass. It only works on orbitrap, FT, and TOF data. It does not use any third party (vendor DLL) code.
- *peakPicking*
  - This filter performs centroiding on spectrawith the selected ms_level argument

    peakPicking [<PickerType>[snr=<minimum signal-to-noise ratio>] [peakSpace=<minimum peak spacing>] [msLevel=<ms_levels>]]
- *scanNumber*
  - This filter takes input as an integer and lets the user select spectra by scan number
  - Scan numbers are 1-based and not contiguous
- *scanEvent*
  - scanEvent filters spectra by scan event and takes input in the form of a set.
  - This filter offers the flexibility of excluding the selected scan events as well.
  - For Example:

    # All scan event EXCEPT 5 can be included using-

    --filter “scanEvent 1-4 6- ”
- *scanTime*
  - Spectra can be selected using a specific time range using this filter.
- *sortByScanTime*
  - As the name suggests, the spectra can be a sorted in ascending order by scan start time.
- *stripIT*
  - “IT” in the filter name refers to “ion trap”.
  - This filter removes out the ion trap data spectra with MS level 1.
- *metadataFixer*
  - As the name suggests, this filter can be used to “fix” metadata—meaning, it adds/replaces a spectra’s TIC/BPI metadata.
  - This filter traverses the *m/z* intensity arrays to find the sum and max.

  - Usually used after peak-picking, for Example:

    --filter "peakPicking true 1-" --filter metadataFixer

- *titleMaker*
  - Uses a string format as the input
  - For Example:

    --filter "titleMaker<RunId>.<ScanNumber>.<ScanNumber>.<ChargeState>"

- *Threshold*
  - This filter can be used to set several thresholds.
  - Data in the input file meeting all the thresholds is retained and everything else is filtered out.
  - Threshold filter can be set to "*most or least*" (i.e., either upper or lower limit can be set) for all give threshold types. This can be set using the orientation in the threshold format as given below.
  - Format for threshold-

    threshold<type><threshold><orientation>[<mslevels>]
  - Various threshold types with their input format and explanation are given below:

Threshold type	Threshold format	Explanation
count	Integer (*n*)	Retains intensity values for the selected number of "*n*" data points. Data points where intensity is equal to *n*th intense data point are removed
count-after-ties	Integer (*n*)	Same as the count but retains data points where intensity = *n*th data point
absolute	Flag (no argument)	Keep data whose absolute intensity matches threshold
bpi-relative	Flag (no argument)	Data is retained if the intensity matches the percentage of base-peak intensity. 0.75 = 75 %
tic-relative	Flag (no argument)	Similar to above but for retains data for individual intensities greater than or less than the percentage of total ion current for the scan. 1 = 100 %
tic-cutoff	Flag (no argument)	As the name suggests, this is a cutoff. Data is retained UPTO the percentage of the total ion current for the scan
ms_levels	Integer (*n*)	OPTIONAL. If selected, only scans with MS level = *n* will be selected

- *mzWindow*
  - As the name suggest, this filter is used to provide a range to select the *m/z* within the selected window.
  - The *m/z* values falling ONLY within the selected range will be retained.
  - Input type: range [mzLow, mzHigh]
  - For Example:

    --filter "mzWindow [100.1,307.5]"
- *mzPrecursors*
  - This filter allows us the flexibility of selecting spectra within a given list of precursor *m/z* values.
  - The input type is a list of *m/z* precursor value
  - For Example:

    --filter "mzPrecursors [123.4,567.8]"
- *defaultArrayLength*
  - For this filter, the default array length refers to the range of peak counts.
  - The name is derived from mzML format file where "defaultArrayLength" refers to peak list.
  - This can be specified as range of integers
  - For Example:

    # Retain only peakcounts >=100:

    --filter "defaultArrayLength 100-"
- *zeroSamples*
  - Usage: zeroSamples <mode> [<MS_levels>]
  - This is a useful filter to deal with zero values in the spectrum. This can be used one of two ways—either remove spectra with zero values or add the sero value incase the spectra is missing.
  - Mode options-
    i. *removeExtra: consecutive zero intensity peaks are removed from spectra*

    "100.1,1000 100.2,0 100.3,0 100.4,0 100.5,0 100.6,1030"

    would become

    "100.1,1000 100.2,0 100.5,0 100.6,1030"

    and a peak list

    "100.1,0 100.2,0 100.3,0 100.4,0 100.5,0 100.6,1030 100.7,0 100.8,1020 100.9,0 101.0,0"

    would become

    "100.5,0 100.6,1030 100.7,0 100.8,1020 100.9,0"

ii. *addMissing: When <mode> is "addMissing", each spectrum's sample rate is automatically determined and flanking zeros are inserted around non-zero data points. The optional [=<flankingZeroCount>] value can be used to limit the number of flanking zeros, otherwise the spectrum is completely populated between nonzero points.*

For example, to make sure spectra have at least five flanking zeros around runs on nonzero points, use filter-

"addMissing = 5".

– The <MS levels> is optional but when used, retains spectra with only the ms levels specified using the option. Format: integer set (For example: 1–5)

- *mzPresent*
  - Usage: mzPresent <tolerance> <type> <threshold> <orientation> <mz_list> [<include_or_exclude>]
  - This filter is very similar to threshold filter
    i. tolerance: Specified as a number and units (PPM or MZ).

    For example, "5 PPM" or "2.1 MZ".

    ii. <type>, <threshold>, and <orientation> operate as in the "threshold" filter (refer to *step 14*).

    iii. <mz_list>: List of mz - [mz1,mz2, … mzn]

    #Data points within <tolerance> of any of these values will be kept. For example,

    "[100, 300, 405.6]"

    iv. <include_or_exclude>: as the terms suggest, using *include* (default) retains all the values that match the criteria where as *exclude* drops data points matching the criteria.

- *scanSumming*
  - Usage: scanSumming [precursorTol=<precursor tolerance>] [scanTimeTol=<scan time tolerance>]
  - Used to sum "*MS2201D* sub-scans whose precursors are within the given <precursor tolerance> and <scan time tolerance>
  - Defaults:
    i. Precursor tolerance: 0.05
    ii. Scan time tolerance: 10 s
  - Used where sub-scans need to be summed to increase the signal-to-noise ratio.
  - IMPORTANT: Only tested on waters data so far.

- *MS2Denoise*
  - Usage: MS2Denoise [<peaks_in_window> [<window_width_Da> [multicharge_fragment_relaxation]]]
  - Use to denoise, i.e., remove peaks with noise for spectra with precursor ions.
    i. <peaks_in_window>: the number peaks to select in moving window, default is 6.
    ii. <window_width_Da>: the width of the window in Da, default is 30.
    iii. <multicharge_fragment_relaxation>—if "true" (the default), allows more data below multiply charged precursors.
- *MS2Deisotope*
  - MS2Deisotope [hi_res [mzTol=<mzTol>]] [Poisson [minCharge=<minCharge>][maxCharge=<maxCharge>]]
  - Uses the Markey method or a Poisson model to deisotope the ms2 spectra.
    i. hi_res: set to "false" (default) or "true"
       - *mzTol: Input format is a decimal value. Sets the m/z tolerances .Regular default is 0.5, high resolution default is 0.01*
    ii. Poisson: Used to define the search range for the charge. Default is 1
       1. minCharge: Default is 1.
       2. maxCharge: Default is 3.
  - For Example:
    --filter "MS2Deisotope true mzTol=0.4 1 minCharge=1 maxCharge=3"
- *ETDFilter*
  - Usage: ETDFilter [<removePrecursor> [<removeChargeReduced> [<removeNeutralLoss> [<blanketRemoval> [<matchingTolerance>]]]]]
    i. <removePrecursor>—specify "true" to remove unreacted precursor (default is "false")
    ii. <removeChargeReduced>—specify "true" to remove charge reduced precursor (default is "false")
    iii. <removeNeutralLoss>—specify "true" to remove neutral loss species from charge reduced precursor (default is "false")
    iv. <matchingTolerance>—specify matching tolerance in MZ or PPM (examples: "3.1 MZ" (the default) or "2.2 PPM")

- *chargeStatePredictor*
  - Usage: chargeStatePredictor [overrideExistingCharge=<true|false (false)>] [maxMultipleCharge=<int (3)>] [minMultipleCharge=<int(2)>][singleChargeFractionTIC=<real (0.9)>] [maxKnownCharge=<int (0)>] [makeMS2=<true|false (false)>]
    i. <overrideExistingCharge> : always override existing charge information (default:"false")
    ii. <maxMultipleCharge> (default 3) and <minMultipleCharge> (default 2): range of values to add to the spectrum's existing "MS_possible_charge_state" values.If these are the same values, the spectrum's MS_possible_charge_state values are removed and replaced with this single value.
    iii. <singleChargeFractionTIC>: is a percentage expressed as a value between 0 and 1 (the default is 0.9, or 90%). This is the value used as the previously mentioned ratio of intensity above and below the precursor *m/z*.
    iv. <maxKnownCharge> (default is 0, meaning no maximum): the maximum charge allowed for "known" charges even if override existing charge is false. This allows overriding junk charge calls like +15 peptides.
    v. <algorithmMakeMS2>: default is "false", when set to "true" the "makeMS2" algorithm is used instead of the one described above.
- *turbocharger*
  - turbocharger [minCharge=<minCharge>] [maxCharge=<maxCharge>] [precursorsBefore=<before>] [precursorsAfter=<after>] [halfIsoWidth=<half-width of isolation window>] [defaultMinCharge=<defaultMinCharge>] [defaultMaxCharge=<defaultMaxCharge>] [useVendorPeaks=<useVendorPeaks>]
  - <maxCharge> (default: 8) and <minCharge> (default 1): defines range of possible precursor charge states.
  - <before> (default: 2) and <after> (default 0): number of survey (MS1) scans to check for precursor isotopes, before and after a MS/MS in retention time.
  - <half-width of isolation window> (default: 1.25): half-width of the isolation window from which precursor is derived. Window is centered at target *m/z* with a total size of ± the value entered.
  - <defaultMinCharge> (default: 0) and <defaultMaxCharge> (default: 0): in the event that no isotope is found in the iso-

lation window, a range of charges between these two values will be assigned to the spectrum. If both values are left at zero, no charge will be assigned to the spectrum.

- *activation*
  - Usage: activation <precursor_activation_type>
  - Filter to retain only spectra of a specified type of activation. DOES NOT effect non-MS spectra.
  - Input options:
    i. ETD
    ii. CID
    iii. SA
    iv. HCD
    v. HECID
    vi. BIRD
    vii. ECD
    viii. IRMPD
    ix. PD
    x. PSD
    xi. PQD
    xii. SID
    xiii. SORI
- *analyzer*
  - Usage: analyzer <analyzer>
  - Filter to retain only spectra of a specified type of mass analyzer
  - Four options for input:
    i. "quad"
    ii. "orbi"
    iii. "FT"
    iv. "IT"
    v. "TOF"
- *analyzerType*
  - Currently deprecated but accepted two options are: "FTMS" or "ITMS"
- *Polarity*
  - This filter allows the user to retain spectra with scan of the selected polarity.

- Available Options:
  i. “negative”
  ii. “positive”
  iii. “+”
  iv. “–”

References

1. Chambers MC, Maclean B, Burke R, Amodei D, Ruderman DL, Neumann S, Gatto L, Fischer B, Pratt B, Egertson J, Hoff K, Kessner D, Tasman N, Shulman N, Frewen B, Baker TA, Brusniak MY, Paulse C, Creasy D, Flashner L, Kani K, Moulding C, Seymour SL, Nuwaysir LM, Lefebvre B, Kuhlmann F, Roark J, Rainer P, Detlev S, Hemenway T, Huhmer A, Langridge J, Connolly B, Chadick T, Holly K, Eckels J, Deutsch EW, Moritz RL, Katz JE, Agus DB, MacCoss M, Tabb DL, Mallick P (2012) A cross-platform toolkit for mass spectrometry and proteomics. Nat Biotechnol 30(10):918–920. doi:10.1038/nbt.2377
2. Kessner D, Chambers M, Burke R, Agus D, Mallick P (2008) ProteoWizard: open source software for rapid proteomics tools development. Bioinformatics 24(21):2534–2536. doi:10.1093/bioinformatics/btn323