TOPP and TOPPView Tutorial
Version: 2.4.0-GIT-NOTFOUND-GIT-NOTFOUND

Contents

1 Concepts

TOPP, The OpenMS Proteomics Pipeline provides a set of computational tools which can be easily combined into analysis pipelines even by non-experts and then be used in proteomics workflows. These applications range from useful utilities file format conversion, peak picking) over wrapper applications for known applications (e.g. Mascot) to completely new algorithmic techniques for data reduction and data analysis. TOPP is based on the OpenMS library and as more functionality is added to new OpenMS releases, TOPP will naturally contain new or updated tools.

In addition to offering a toolbox, TOPP contains TOPPView - the central graphical user interface - which allows the user to view, inspect and manipulate proteomics data. TOPPView reads standard formats and allows the user not only to view the data, but also to interactively call TOPP tools and display the results. As such it is a powerful tool for interactive data manipulation.

In this tutorial we will now in detail explain the capabilities of TOPPView and the TOPP tools using real data sets and standard analysis tasks.

2 TOPPView main interface

TOPPView is a viewer for MS and HPLC-MS data. It can be used to inspect files in mzML, mzData, mzXML and several other text-based file formats.

In each view, several datasets can be displayed using the layer concept. This allows visual comparison of several datasets as well as displaying input data and output data of an algorithm together.

TOPPView is intended for visual inspection of the data by experimentalists as well as for analysis software by developers.
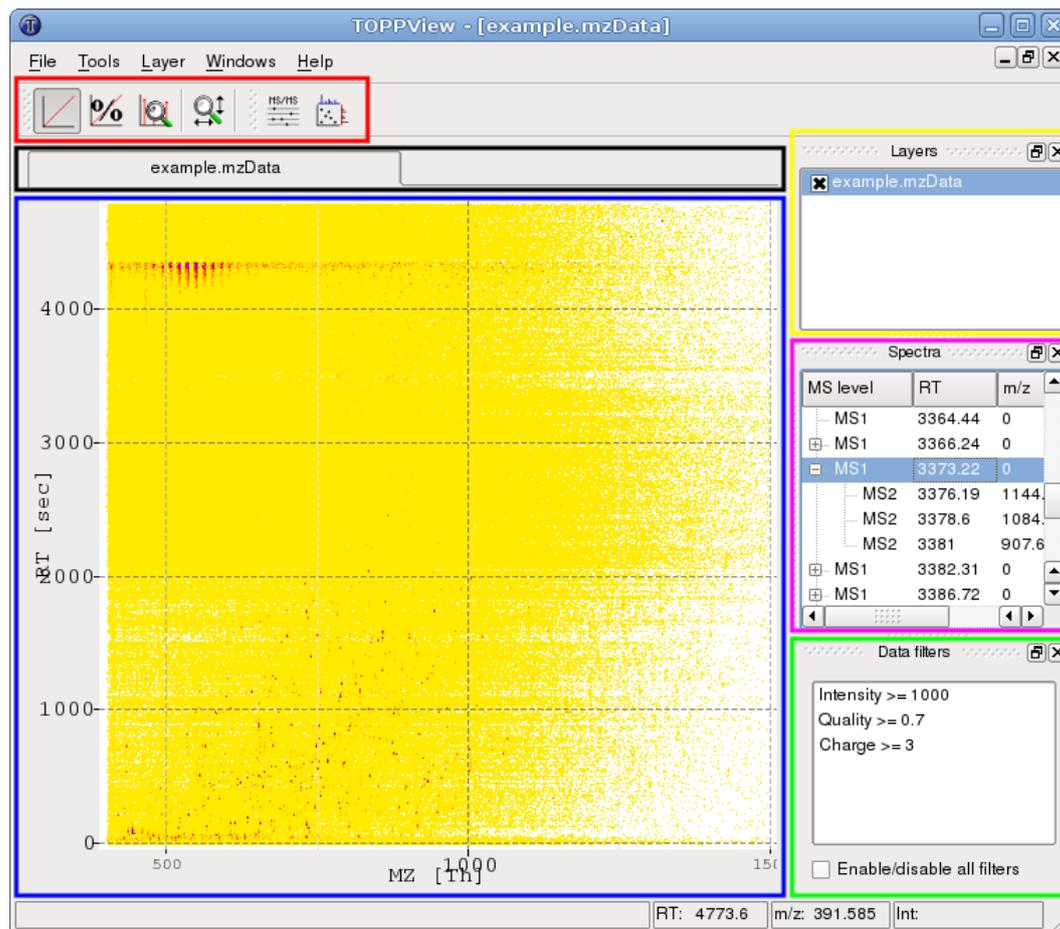
The above example image shows a 2D view of TOPPView (blue rectangle) and the corresponding Display modes and view options (red rectangle). In the right dock area, you can see the Layers (yellow rectangle), the Spectrum browser (magenta rectangle) and the Data filtering tool (green rectangle). Switching between open windows can be done using the tab bar (black rectangle).

2.1 Layers

Each view of TOPPView supports several datasets, called layers. In the *layer manager* (dock window in the upper right corner), layers can be hidden and shown using the check box in front of each layer name.

By clicking on a layer, this layer is selected, which is indicated by a blue background. The selected layer can be manipulated using the *Tools* menu.

Layers can be copied by dragging them to the tab bar. If the layer is dropped on a tab, it is added to the corresponding window. If the layer is dropped on the tab bar but not on a tab, a new window with that layer is added.

2.2 Spectrum browser

The spectra contained in a peak map can be browsed in the *spectrum browser*. It displays a tree view of all spectra in the current layer, where a spectrum with MS level *n* is a child of the spectrum with MS level *n - 1* that contains its corresponding precursor peak. For each spectrum in the list, the retention time and (for spectra with MS level > 1) the m/z value of the precursor peak are also shown.

If a 2D or 3D window is active, double-clicking a spectrum will open it in a new 1D window. If the active window is a 1D view, the spectra can be browsed and the currently selected spectrum will be shown.

2.3 Data filtering

TOPPView allows filtering of the displayed peak data and feature data. Peak data can be filtered according to intensity and meta data. Meta data is arbitrary data the peak is annotated with. Feature data can be filtered according to intensity, charge, quality and meta data.

Data filters are managed by a dock window. Filters can be added, removed and edited through the context menu (right button mouse click) of the data filters window. For convenience, filters can also be edited by double-clicking them.

2.4 Command line options

Several files can be opened in one layer from the command line by putting a '+' character between the file names. The following command opens three files in three layers of the same window:

```
TOPPView file1.mzML + file2.mzML + file3.mzML
```

Without the '+' the files would be opened in three different windows.

The color gradient used to display a file can be changed by adding one of several '@' commands. The following command opens the file with a white-to-black gradient:

```
TOPPView file1.mzML @bw
```

For more information on command line parameters call:

```
TOPPView --help
```

2.5 Looking for help?

You can display a short help text for each button and dock window of TOPPView by clicking on it in *What's this*? mode. *What's this*? mode can be entered using the *Help* menu.
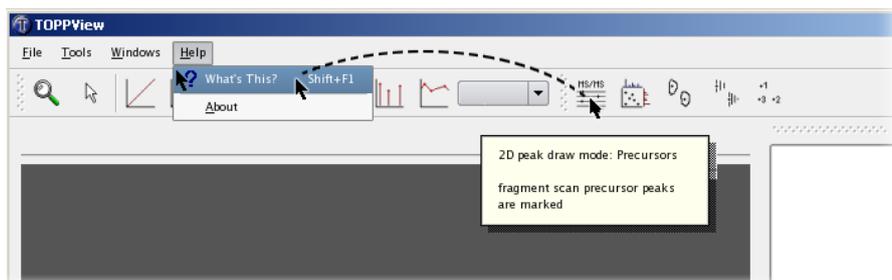
TOPPView offers three types of views – a 1D view for spectra, a 2D view for peak maps and feature maps, and a 3D view for peak maps. All three views can be freely configured to suit the individual needs of the user.

2.6 Basic use

All three views share a similar interface. Three action modes are supported – one for translation, one for zooming and one for measuring:

- Translate mode
  - It is activated by default
  - Move the mouse while holding the mouse button down to translate the current view
  - Arrow keys can be used to translate the view without entering translate mode (in 1D-View you can additionally use Shift-key to jump to the next peak)
- Zoom mode
  - All previous zoom levels are stored in a zoom history. The zoom history can be traversed using CTRL+/CTRL- or the mouse wheel (scroll up and down)
  - Zooming into the data:
    * either mark an area in the current view with your mouse, while holding the left mouse button plus the *CTRL* key to zoom to this area.
    * or use your mouse wheel to traverse the zoom history
    * if you have reached the end of the history and keep on pressing CTRL+ or scroll up, the current area will be enlarged by a factor of 1.25
  - Pressing the *Backspace* key resets the zoom and zoom history
- Measure mode
  - It is activated using the *SHIFT* key
  - Press the left mouse button down while a peak is selected and drag the mouse to another peak to measure the distance between peaks
  - This mode is implemented in the 1D and 2D mode only

2.7 1D view

The 1D view is used to display raw spectra or peak spectra. Raw data is displayed using a continuous line. Peak data is displayed using one stick per peak. The color used for drawing the lines can be set for each layer individually. The 1D view offers a mirror mode, where the window is vertically divided in halves and individual layers can be displayed either above or below the "mirror" axis in order to facilitate quick visual comparison of spectra. When a mirror view is active, it is possible to perform a spectrum alignment of a spectrum in the upper and one in the lower half, respectively. Moreover, spectra can be annotated manually. Currently, distance annotations between peaks, peak annotations and simple text labels are provided.

The following example image shows a 1D view in mirror mode. A theoretical spectrum (lower half) has been generated using the theoretical spectrum generator (*Tools > Generate theoretical spectrum*). The mirror mode has been activated by right-clicking the layer containing the theoretical spectrum and selecting *Flip downward* from the layer context menu. A spectrum alignment between the two spectra has been performed (*Tools > Align spectra*). It is visualized by the red lines connecting aligned peaks and can be reset through the context menu. Moreover, in the example, several distances between abundant peaks have been measured and subsequently replaced by their corresponding amino acid residue code. This is done by right-clicking a distance annotation and selecting *Edit* from the context menu. Additionally, peak annotations and text labels have been added by right-clicking peaks and selecting *Add peak annotation* or by right-clicking anywhere and selecting *Add label*, respectively. Multiple annotations can be selected by holding down the CTRL key while clicking them. They can be moved around by dragging the mouse and deleted by pressing DEL.
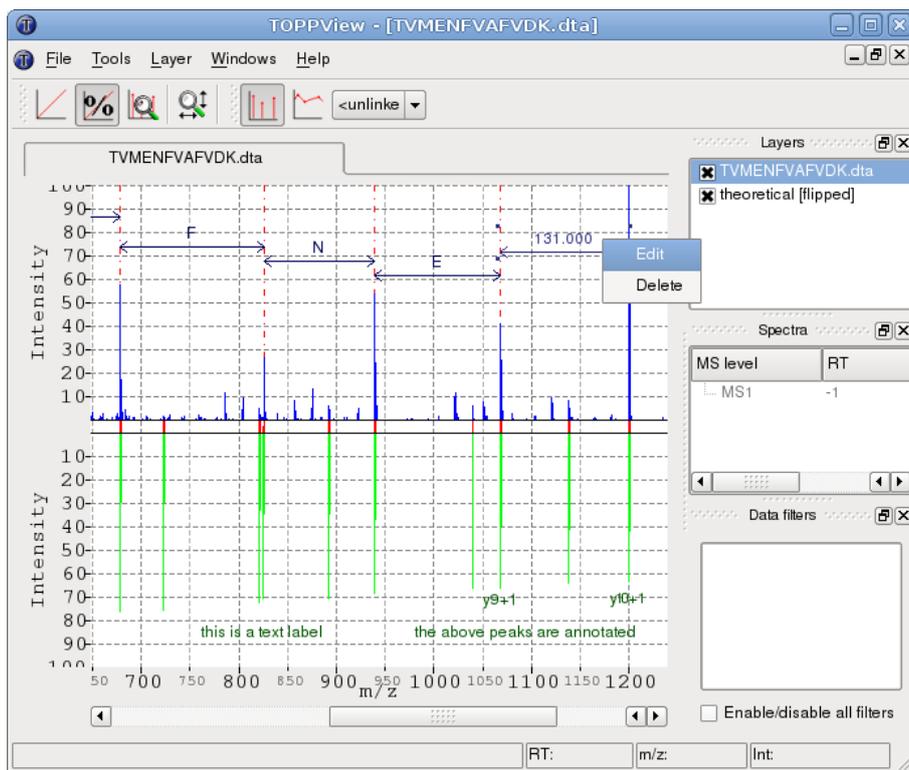
Through the **context menu:** of the 1D view you can:

- View/edit meta data

- Save the current layer data

- Change display settings

- Add peak annotations or arbitrary text labels

- Reset a performed alignment

2.8 2D view

The 2D view is used to display peak maps and feature maps in a top-down view with color-coded intensities. Peaks and feature centroids are displayed as dots. For features, also the overall convex hull and the convex hulls of individual mass traces can be displayed. The color gradient used to encode for peak and feature intensities can be set for each layer individually.

The following example image shows a small section of a peak map and the detected features in a second layer.
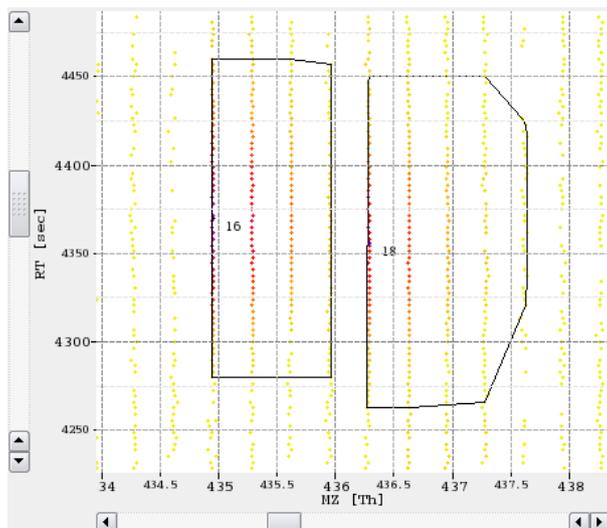
In addition to the normal top-down view, the 2D view can display the projections of the data to the m/z and RT axis. This feature is mainly used to assess the quality of a feature without opening the data region in 3D view. Through the **context menu:** of the 2D view you can:

- View/edit meta data

- View survey/fragment scans in 1D view

- View survey/fragment scans meta data

- View the currently selected area in 3D view

- Save the current layer data

- Change display settings

2.9   3D view

The 3D view can display peak maps only. Its primary use is the closer inspection of a small region of the map, e.g. a single feature. In the 3D view slight intensity differences are easier to recognize than in the 2D view. The color gradient used to encode peak intensities, the width of the lines and the coloring mode of the peaks can be set for each layer individually.
The following example image shows a small region of a peak map:
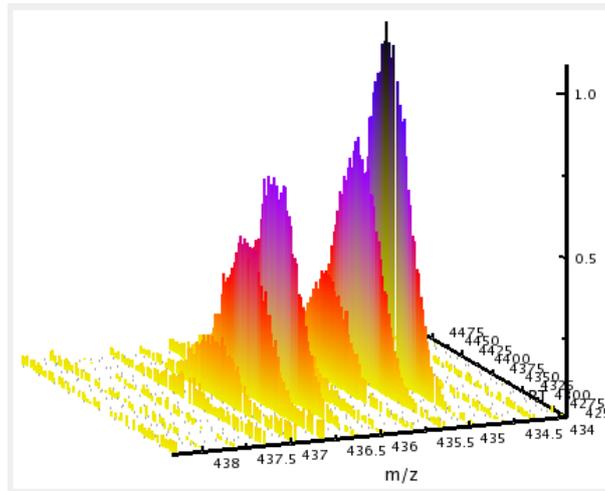
Through the **context menu:** of the 3D view you can:

- View/edit meta data

- Save the current layer data

- Change display settings

All of the views support several display modes and view options. Display modes determine how intensities are displayed. View options configure the view.
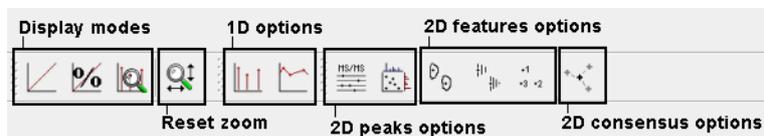
2.10   Display modes

Intensity display modes determine the way peak intensities are displayed.

- **Linear:**
  Normal display mode.

- **Percentage:**
  In this display mode the intensities of each dataset are normalized with the maximum intensity of the dataset. This is especially useful in order to visualize several datasets that have large intensity differences. If only one dataset is opened it corresponds to the normal mode.

- **Snap to maximum intensity:**
  In this mode the maximum currently displayed intensity is treated as if it was the maximum overall intensity.

2.11   View options

View options configure the view of the current layer.

- **1D:**
  Switching between raw data and peak mode.

- **2D (peaks):**
  MS/MS precursor peaks can be highlighted.
  Projections to m/z and RT axis can be shown.

- **2D (features):**
  Overall convex hull of the features can be displayed.
  Convex hulls of mass traces can be displayed (if available).
  A feature identifier, consisting of the feature index and an)optional label can be displayed.

- **2D (consensus):**
  The elements of a consensus feature can be displayed.

- **3D:**
  Currently there are no options for 3D view.

TOPPView also offers limited data analysis capabilities for single layers, which will be illustrated in the following sections. The functionality presented here can be found in the *Tools* menu:
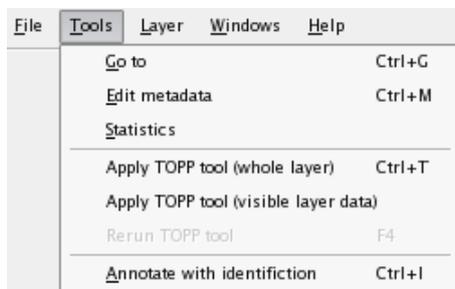
2.12 TOPP tools

Single TOPP tools can be applied to the data of the currently selected layer or to the visible data of the current layer. The following example image shows the TOPP tool dialog:
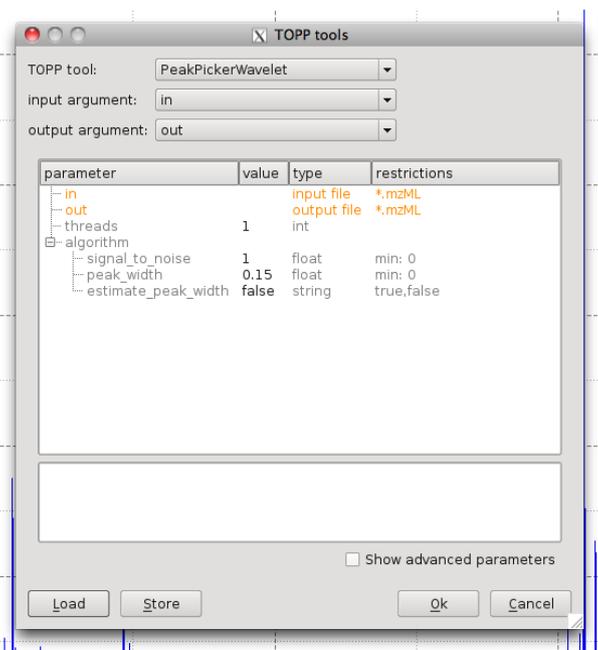
To apply a TOPP tool, do the following:

- Select a TOPP tool and if necessary a type.

- Specify the command line option of the tool, that takes the input file name.

- Specify the command line option of the tool, that takes the output file name.

- Set the algorithm parameters manually or load them from an INI file.

2.13 Metadata

One can access the meta data the layer is annotated with. This data comprises e.g. contact person, instrument description and sample description.
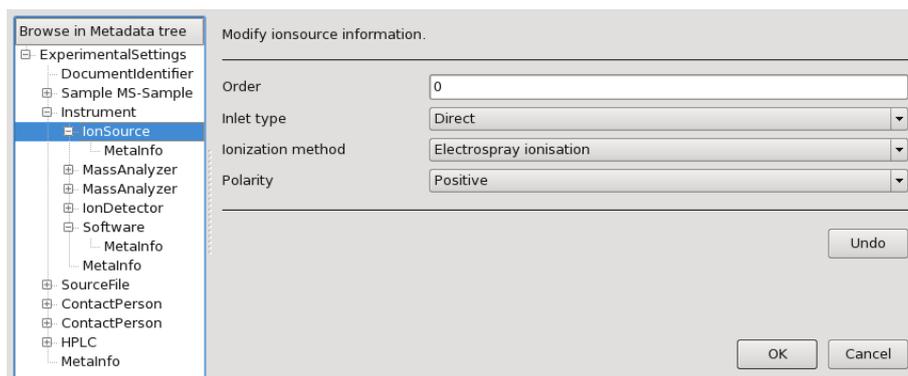
Note

Identification data, e.g. from a Mascot run, can be annotated to the spectra or features, too. After annotation, this data is listed in the meta data as well.

2.14 Statistics

Statistics about peak/feature intensities and peak meta information can be displayed. For intensities, it is possible to display an additional histogram view.

	Count	Min	Max	Average	Distribution
Intensity	–	10.00	7284.19	37.27	Show
area	1041134	0.40	1210.53	7.24	
charge	1041134	0.00	0.00	0.00	
fwhm	1041134	0.10	1.92	0.19	
leftWidth	1041134	0.48	84.78	10.32	
peakShape	1041134	1.00	1.00	1.00	
rValue	1041134	0.51	1.00	0.97	
rightWidth	1041134	0.70	84.89	10.31	
signalToNoise	1041134	2.00	1147.68	10.27	

TOPPView also offers editing functionality for feature layers.

After enabling the feature editing mode in the context menu of the feature layer, the following actions can be performed:

- Features can be dragged with the mouse in order to change the m/z and RT position.

- The position, intensity and charge of a feature can be edited by double-clicking a feature.

- Features can be created by double-clicking the layer background.

- Features can be removed by selecting them and pressing the DEL key.

File handling	
CTRL+O	Open file
CTRL+W	Close current window
CTRL+S	Save current layer
CTRL+SHIFT+S	Save visible data of current layer

Navigation in the data	
CTRL	Activate zoom mode
SHIFT	Activate measurement mode
Arrow keys	Translate currently shown data (1D View: Shift + Left/Right moves to the next peak in sparse data)
CTRL+'+',CTRL+'-'	Move up and down in zoom history
mouse wheel	Move up and down in zoom history
CTRL+G	Goto dialog
Backspace	Reset zoom
PageUp	Select previous layer
PageDown	Select next layer

Visualization options	
CTRL+R	Show/hide grid lines
CTRL+L	Show/hide axis legends
N	Intensity mode: Normal
P	Intensity mode: Percentage
S	Intensity mode: Snap-to-maximum
I	1D draw mode: peaks
R	1D draw mode: raw data
CTRL+ALT+Home	2D draw mode: increase minimum canvas coverage threshold (for raw peak scaling). 'Home' on MacOSX keyboards is also 'Fn+ArrowLeft'
CTRL+ALT+End	2D draw mode: decrease minimum canvas coverage threshold (for raw peak scaling). 'End' on MacOSX keyboards is also 'Fn+ArrowRight'
CTRL+ALT+'+'	2D draw mode: increase maximum point size (for raw peak scaling)
CTRL+ALT+'-'	2D draw mode: decrease maximum point size (for raw peak scaling)

Annotations in 1D view	
CTRL+A	Select all annotations of the current layer
DEL	Delete all currently selected annotations

Advanced	
CTRL+T	Apply TOPP tool to the current layer
CTRL+SHIFT+T	Apply TOPP tool to the visible data of the current layer
F4	Rerun TOPP tool
CTRL+M	Show layer meta information
CTRL+I	Annotate with identification results
1	Show precursor peaks (2D peak layer)
2	Show projections (2D peak layer)
5	Show overall convex hull (2D feature layer)

6	Show all convex hulls (2D feature layer)
7	Show numbers and labels (2D feature layer)
9	Show consensus elements (2D consensus layer)
Help	
F1	Show TOPPView online tutorial
SHIFT+F1	Activate *What's this?* mode

3 Calling TOPP tools from TOPPView

To smooth your raw data call one of the available NoiseFilters via the Tools-menu, (*Tools > Apply TOPP tool*), then select NoiseFilterSGolay or NoiseFilterGaussian as TOPPtool (green rectangle). The parameters for the filter type can be adapted (blue rectangle). For the Savitzky-Golay filter you can set the frame length and the order of the polynomial that is fitted. For the Gaussian filter the gaussian width and the ppm tolerance for a flexible gaussian width depending on the m/z value can be adapted. Press Ok to run the selected NoiseFilter.
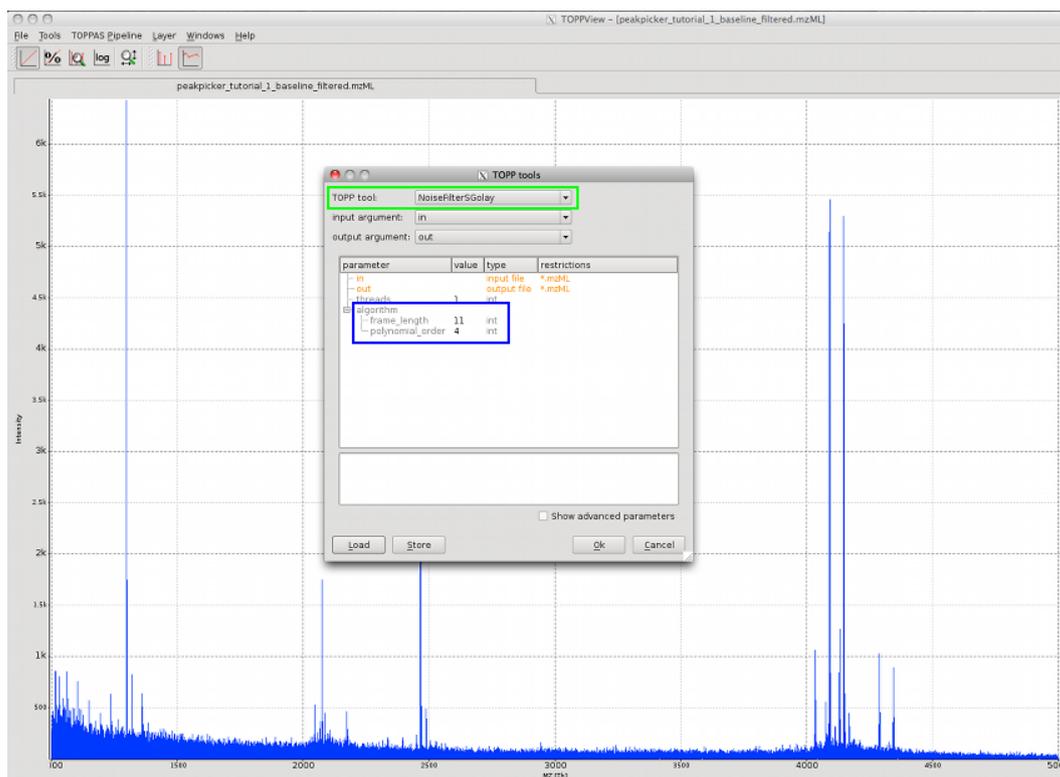
The following image shows a part of the spectrum after smoothing as red line with the un-smoothed data in green.
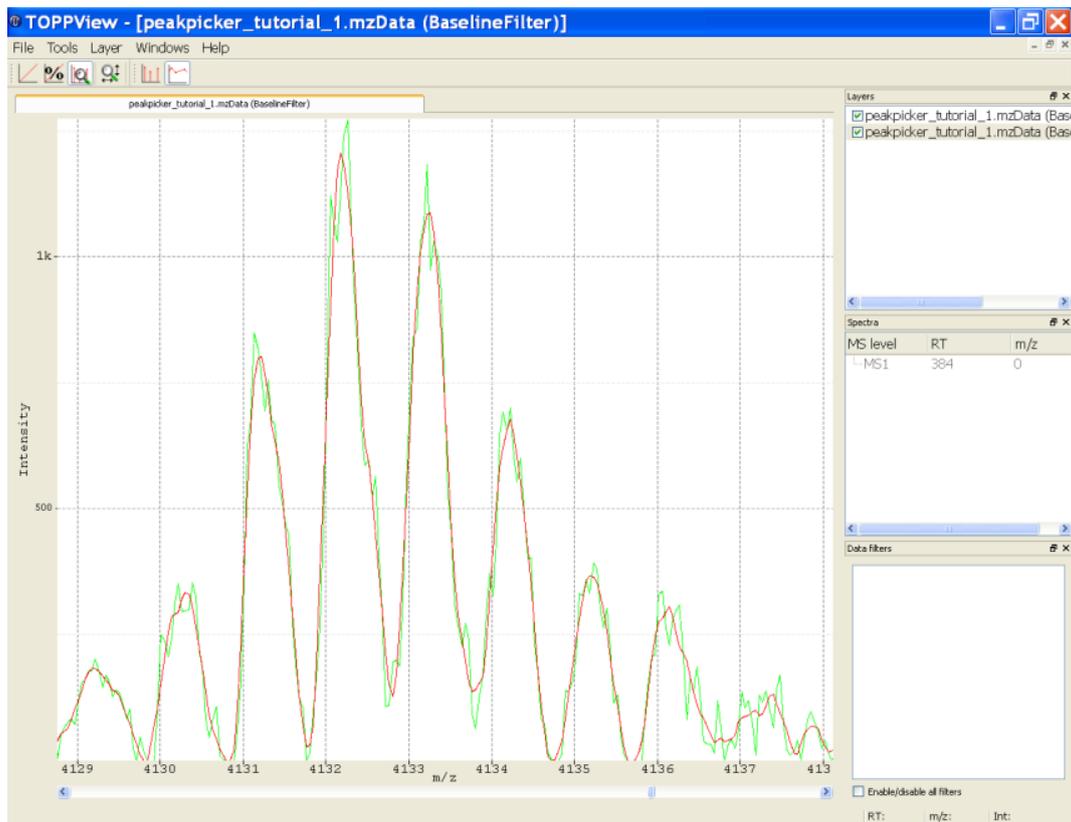
First we load the spectrum to be analyzed in TOPPView. If you want to test the described tools, you can open the tutorial data via the File-menu (*File > Open example file*, then select *peakpicker_tutorial_1.mzML*). The BaselineFilter can be called via the Tools-menu (*Tools > Apply TOPP tool*), then select BaselineFilter as TOPPtool (red rectangle). One can choose between different types of filters (green rectangle), the one mainly used is TopHat. The other important parameter is the length of the structuring element (blue rectangle). The default value is 3 Thomson. Press OK to start the baseline subtraction.
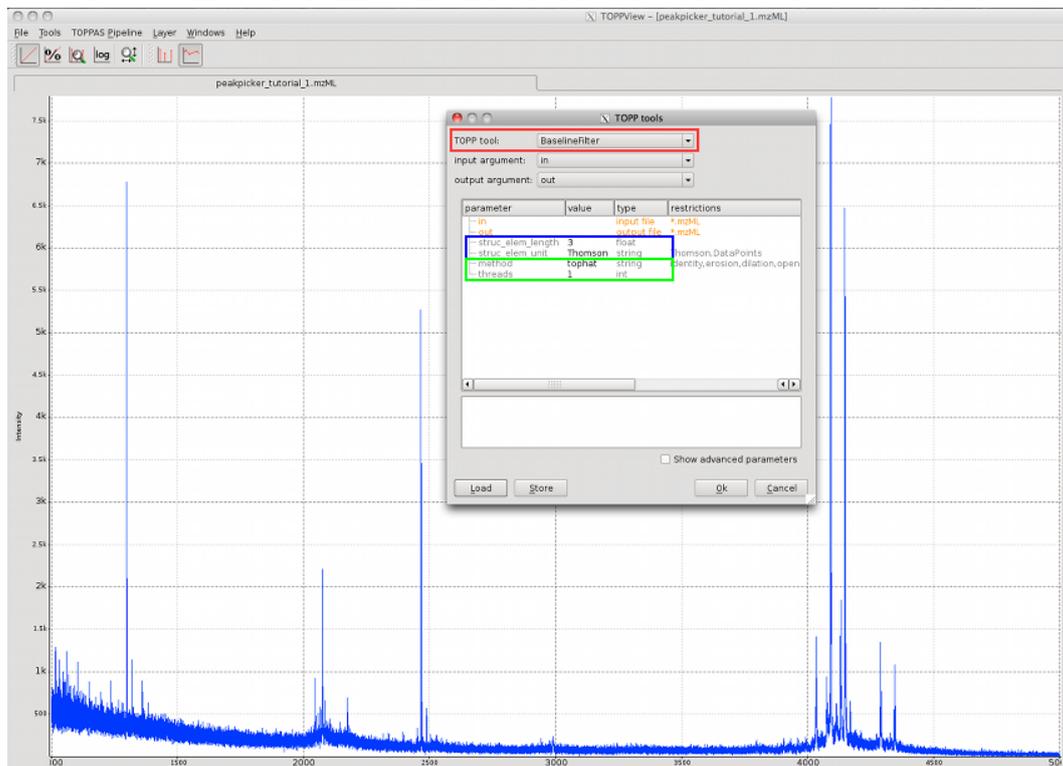
The following image shows a part of the spectrum after baseline filtering as green line, the original raw data is shown by the blue line.
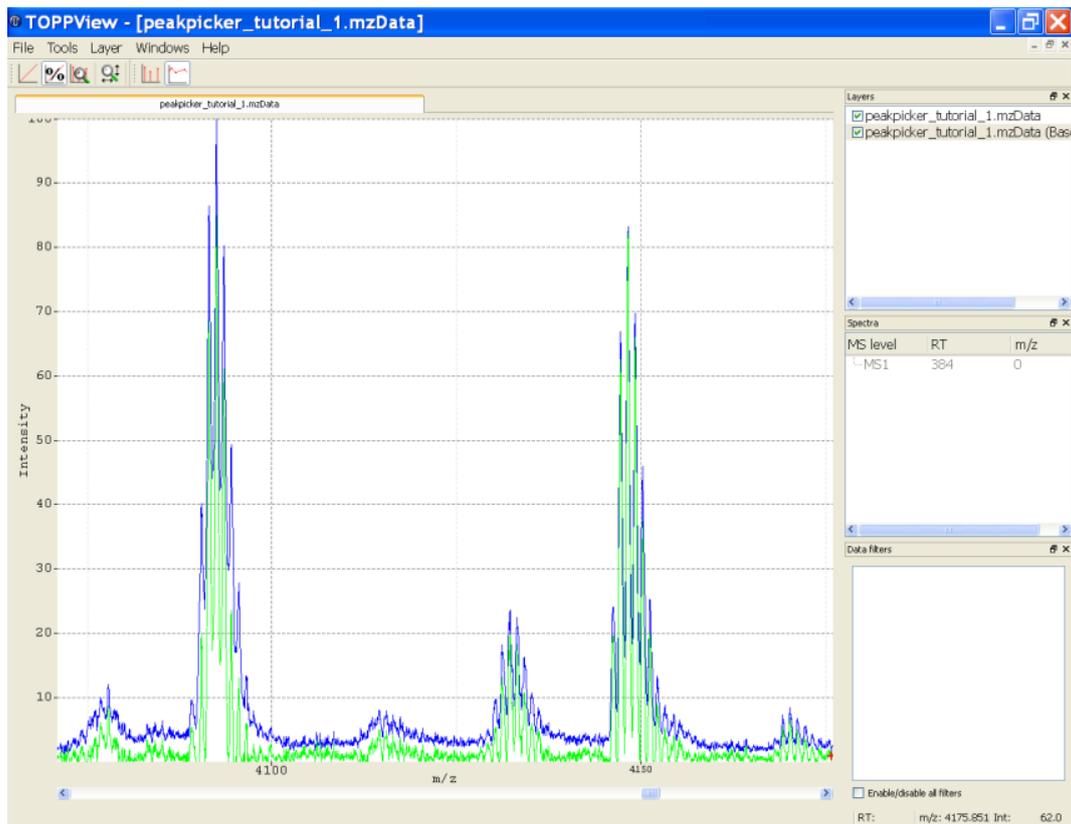
If you have low resolution data you may consider to smooth your data first (Smoothing raw data) and subtract the baseline (Subtracting a baseline from a spectrum) before peak picking.

We have two types of PeakPickers, the PeakPickerWavelet and one especially suited for high resolution data (Peak↩ PickerHiRes). In this tutorial we consider the PeakPickerWavelet. We use the file *peakpicker_tutorial_2.mzML* from the examples data (*File > Open example data*).

The main parameters are the peak width and the minimal signal to noise ratio for a peak to be picked. If you don't know the approximate fwhm of your peaks you can use the estimation included in the PeakPickerWavelet. Here you need to set the flag estimate_peak_width to true.After applying the PeakPickerWavelet you can see which peak width was estimated and used for peak picking in the log window.

If you want to estimate the peak width on your own, you may use the measuring tool (Basic use) to determine the fwhm of one or several representative peaks.

If the peak picker delivers only a few peaks even though the the peak_with and signal_to_noise parameters are set to good values, you may consider changing the advanced parameter fwhm_lower_bound_factor to a lower value. All peaks with a lower fwhm than fwhm_lower_bound_factor ∗ peak_width are discarded.

The following image shows a part of the spectrum with the picked peaks shown in green, the estimated peak width in the log window and the measured peak width.
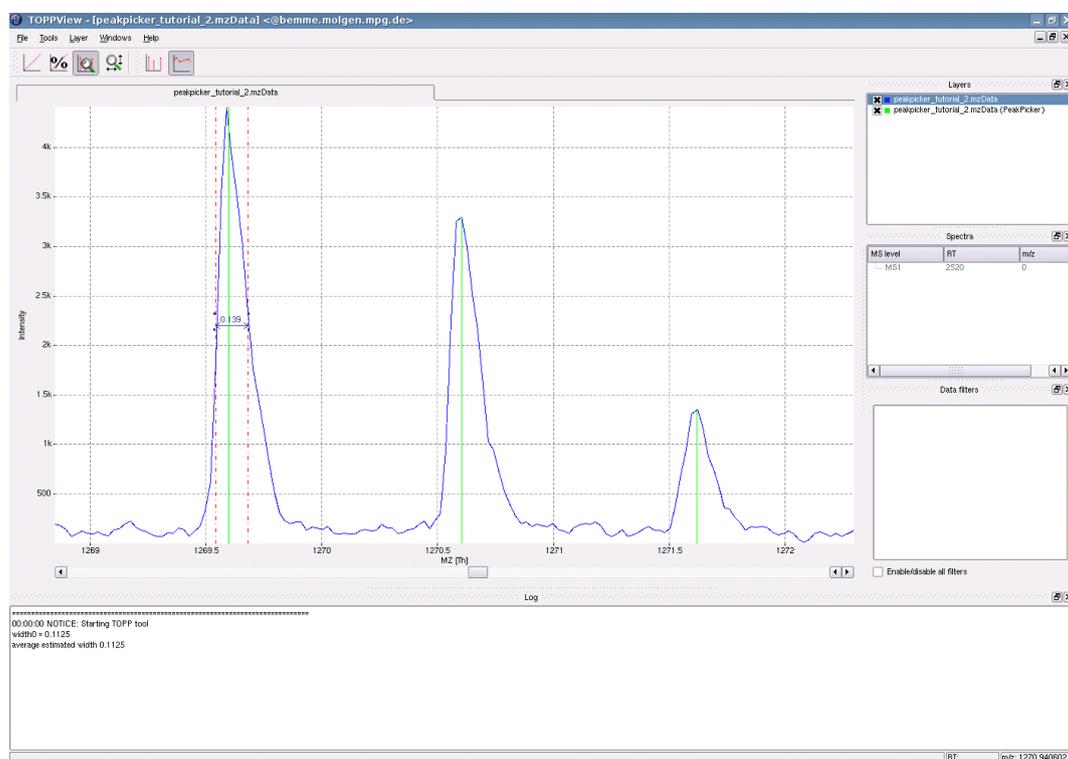
To quantify peptide features, TOPP offers the **FeatureFinder** tools. In this section the **FeatureFinderCentroided** is used, which works on centroided data only. There are other FeatureFinders available that also work on profile data.

For this example the file *LCMS-centroided.mzML* from the examples data is used (*File > Open example data*). In order to adapt the algorithm to our data, some parameters have to be set.

Intensity

The algorithm estimates the significance of peak intensities in a local environment. Therefore, the HPLC-MS map is divided into n times n regions. Set the *intensity:bins* parameter to 10 if you are working on a whole map. If you are working on a small region only, you should set it to 1.

Mass trace

For the mass traces, you have to define the number of adjacent spectra in which a mass has to occur (*mass_trace↩ :min_spectra*). In order to compensate for peak picking errors, missing peaks can be allowed (*mass_trace:max↩ _missing*) and a tolerated mass deviation must be set (*mass_trace:mz_tolerance*).

Isotope pattern

The expected isotopic intensity pattern is estimated from an averagene amino acid composition. The algorithm searches all charge states in a defined range (*isotopic_pattern:change_min* to *isotopic_pattern:change_max*). Just as for mass traces, a tolerated mass deviation between isotopic peaks has to be set (*isotopic_pattern:mz_tolerance*). The image shows the centroided peak data and the found peptide features. The used parameters can be found in the TOPP tools dialog.
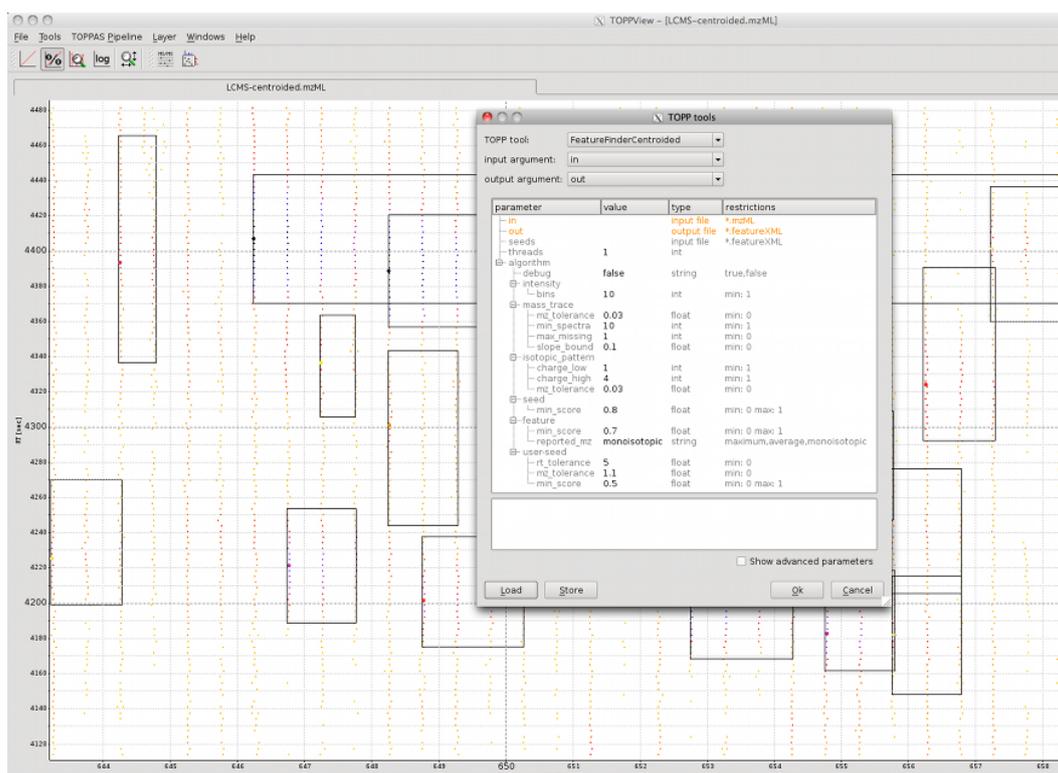
4   Using TOPPAS to create and run TOPP pipelines

TOPPAS allows you to create, edit, open, save, and run TOPP workflows. Pipelines can be created conveniently in a GUI. The parameters of all involved tools can be edited within TOPPAS and are also saved as part of the pipeline definition in the .toppas file. Furthermore, **TOPPAS** interactively performs validity checks during the pipeline editing process and before execution (i.e., a dry run of the entire pipeline), in order to prevent the creation of invalid workflows. Once set up and saved, a workflow can also be run without the GUI using *ExecutePipeline -in <file>*.

The following figure shows a simple example pipeline that has just been created and executed successfully:
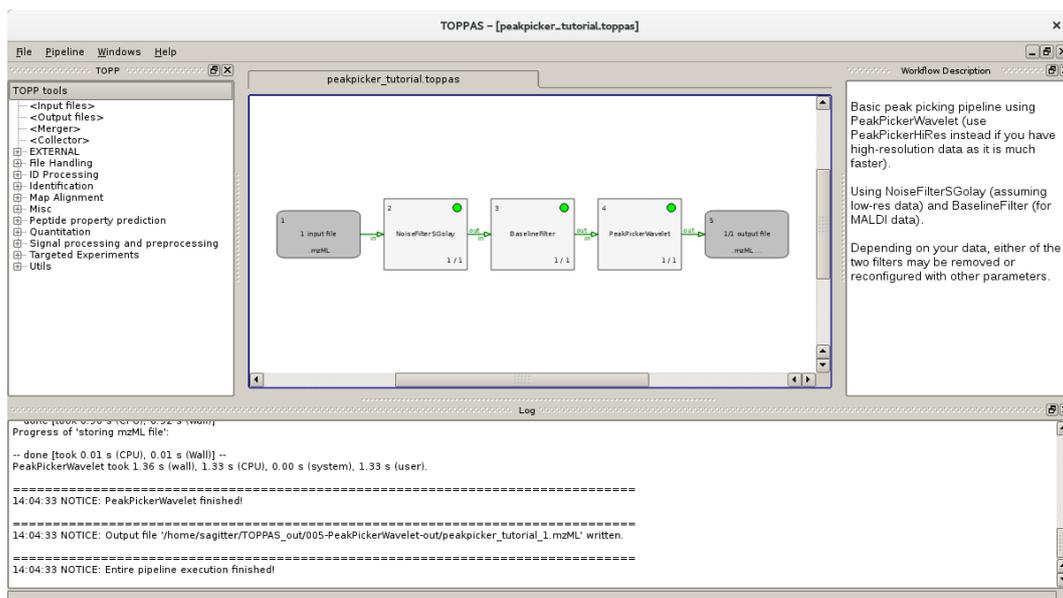
To create a new TOPPAS file, you can either:

- open TOPPAS without providing any existing workflow - an empty workflow will be opened automatically

- in a running TOPPAS program choose: *File > New*

- create an empty file in your file browser (explorer) with the suffix *.toppas* and double-click it (on Windows systems all *.toppas* files are associated with TOPPAS automatically during installation of OpenMS, on Linux and MacOS you might need to manually associate the extension)

4.1 Introduction

The following figure shows the **TOPPAS** main window and a pipeline which is just being created. The user has added some tools by drag&dropping them from the TOPP tool list on the left onto the central window. Additionally, the user has added nodes for input and output files.

Next, the user has drawn some connections between the tools which determine the data flow of the pipeline. Connections can be drawn by first *deselecting* the desired source node (by clicking anywhere but not on another node) and then dragging the mouse from the source to the target node (if a *selected* node is dragged, it is moved on the canvas instead). When a connection is created and the source (the target) has more than one output (input) parameter, an input/output parameter mapping dialog shows up and lets the user select the output parameter of the source node and the input parameter of the target node for this data flow - shown here for the connection between PeptideIndexer and FalseDiscoveryRate. If the file types of the selected input and output parameters are not compatible with each other, **TOPPAS** will refuse to add the connection. It will also refuse to add a connection if it would create a cycle in the workflow, or if it just would not make sense, e.g., if its target is an input file node. The connection between the input file node (#1) and the OMSSAAdapter (#5) tool is painted yellow which indicates it is not ready yet, because no input files have been specified.
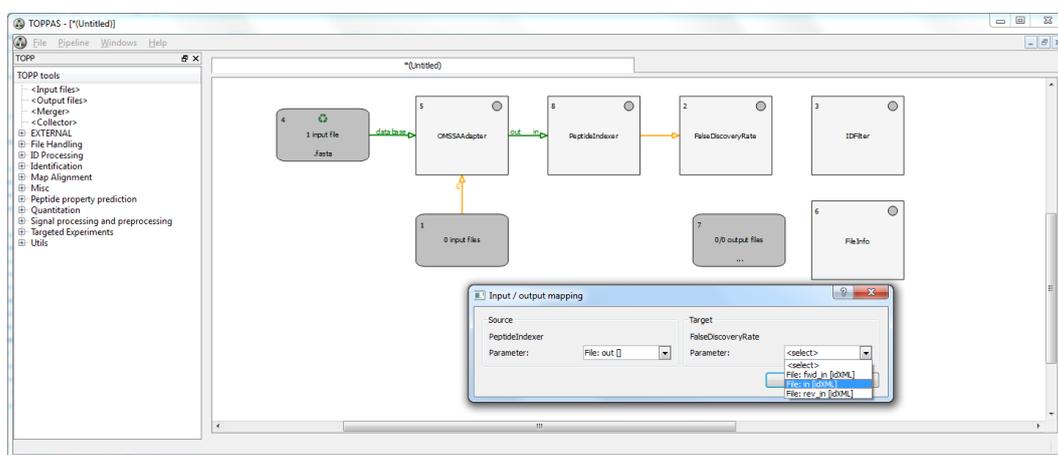
The input/output mapping of connections can be changed at any time during the editing process by double-clicking an connections or by selecting *Edit I/O mapping* from the context menu which appears when a connection is right-clicked. All visible items (i.e. connections and the different kinds of nodes) have such a context menu. For a detailed list of the different menus and their entries, see Menus .

The following figure shows a possible next step: the user has double-clicked one of the tool nodes in order to configure its parameters. By default, the standard parameters are used for each tool. Again, this can also be done by selecting *Edit parameters* from the context menu of the tool.
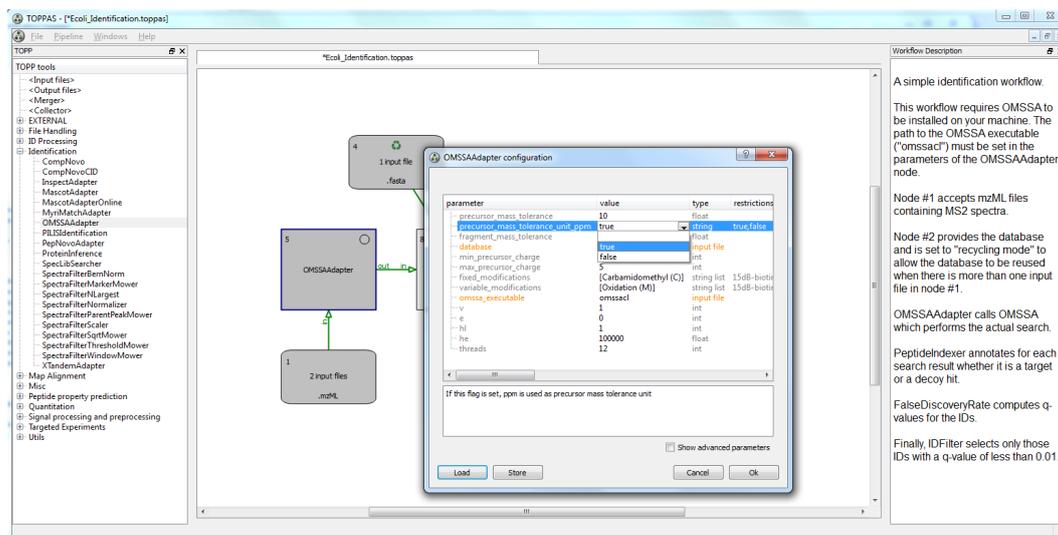
Once the pipeline has been set up, the input files have to be specified before the pipeline can be executed. This is done by double-clicking an input node and selecting the desired files in the dialog that appears. Input nodes have a special mode named "recycling mode", i.e., if the input node has fewer files than the following node has rounds (as it might have two incoming connections) then the files are recycled until all rounds are satisfied. This might be useful if one input node specifies a single database file (for a Search-Adapter like Mascot) and another input node has the actual MS2 experiments (which is usually more than one). Then the database input node would be set to "recycle" the database file, i.e. use it for every run of the MascotAdapter node. The input list can be recycled an arbitrary number of times, but the recycling has to be *complete*, i.e. the number of rounds of the downstream node have to be a multiple of the number of input files. Recycling mode can be activated by right-clicking the input node and selecting the according entry from the context menu. Finally, if you have input and output nodes at every end of your pipeline and all connections are green, you can select *Pipeline > Run* in the menu bar or just press *F5*.
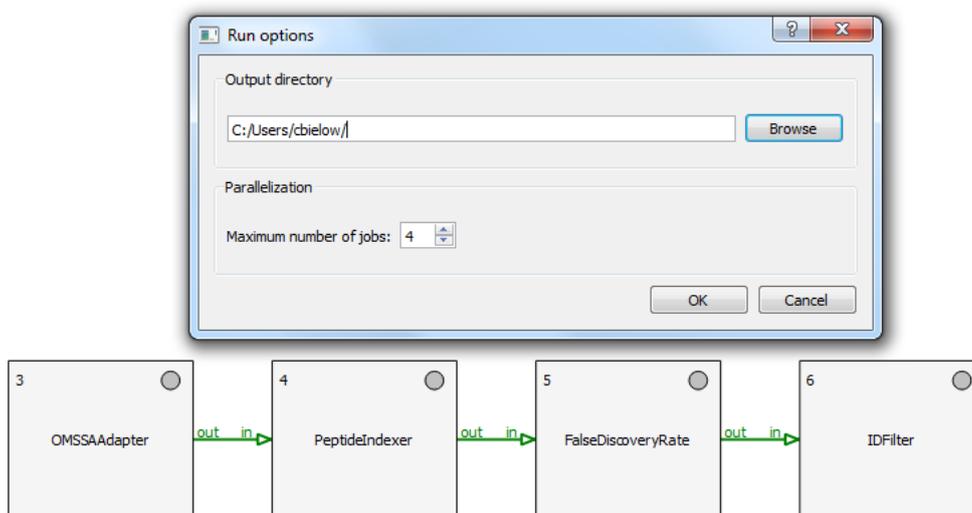
You will be asked for an output file directory where a sub-directory, *TOPPAS_out*, will be created. This directory will contain your output files. You can also specify the number of jobs TOPPAS is allowed to run in parallel. If a number greater than 1 is selected, TOPPAS will parallelize the pipeline execution in the following scenarios:

- A tool has to process more than one file. In this case, multiple files can be processed in parallel.

24

- The pipeline contains multiple branches that are independent of each other. In this case, multiple tools can run in parallel.

Be careful with this setting, however, as some of the TOPP tools require large amounts of RAM (depending on the size of your dataset). Running too many parallel jobs on a machine with not enough memory will cause problems. Also, do not confuse this setting with the *threads* parameter of the individual TOPP tools: every TOPP tool has this parameter specifying the maximum number of threads the tool is allowed to use (although only a subset of the TOPP tools make use of this parameter, since there are tasks that cannot be computed in parallel). Be especially careful with combinations of both parameters! If you have a pipeline containing the *FeatureFinderCentroided*, for example, and set its *threads* parameter to 8, and you additionally set the number of parallel jobs in **TOPPAS** to 8, then you end up using 64 threads in parallel, which might not be what you intended to do.

In addition to *TOPPAS_out*, a *TOPPAS_tmp* directory will be created in the OpenMS temp path (call the *Open↩ MSInfo* tool to see where exactly). It will contain all temporary files that are passed from tool to tool within the pipeline. Both folders contain further sub-directories which are named after the number in the top-left corner of the node they belong to (plus the name of the tool for temporary files). During pipeline execution, the status lights in the top-right corner of the tools indicate if the tool has finished successfully (green), is currently running (yellow), has not done anything so far (gray), is scheduled to run next (blue), or has crashed (red). The numbers in the bottom-right corner of every tool show how many files have already been processed and the overall number of files to be processed by this tool. When the execution has finished, you can check the generated output files of every node quickly by selecting "@a Open @a files @a in @a TOPPView" or "@a Open @a containing @a folder" from the context menu (right click on the node).

4.2 Mouse and keyboard

Using the mouse, you can

- drag&drop tools from the TOPP tool list onto the workflow window (you can also double-click them instead)

- select items (by clicking)

- select multiple items (by holding down *CTRL* while clicking)

- select multiple items (by holding down *CTRL* and dragging the mouse in order to "catch" items with a selection rectangle)

- move all selected items (by dragging one of them)

- draw a new connection from one node to another (by dragging; source must be deselected first)

- specify input files (by double-clicking an input node)

- configure parameters of tools (by double-clicking a tool node)

- specify the input/output mapping of connections (by double-clicking a connection)

- translate the view (by dragging anywhere but on an item)

- zoom in and out (using the mouse wheel)

- make the context menu of an item appear (by right-clicking it)

Using the keyboard, you can

- delete all selected items (*DEL* or *BACKSPACE*)

- zoom in and out (+ / -)

- run the pipeline (*F5*)

- open this tutorial (*F1*)

Using the mouse+keyboard, you can

- copy a node's parameters to another node (only parameters with identical names will be copied, e.g., 'fixed_modifications') (*CTRL* while creating an edge) The edge will be colored as dark magenta to indicate parameter copying.

4.3   Menus

Menu bar:

In the *File* menu, you can

- create a new, empty workflow (*New*)

- open an existing one (*Open*)

- open an example file (*Open example file*)

- include an existing workflow to the current workflow (*Include*)

- visit the online workflow repository (*Online repository*)

- save a workflow (*Save / Save as*)

- export the workflow as image (*Export as image*)

- refresh the parameter definitions of all tools contained in the workflow (*Refresh parameters*)

- close the current window (*Close*)

- load and save TOPPAS resource files (.trf) (*Load / Save TOPPAS resource file*)

In the *Pipeline* menu, you can

- run a pipeline (*Run*)

- abort a currently running pipeline (*Abort*)

In the *Windows* menu, you can

- make the TOPP tool list window on the left, the description window on the right, and the log message at the bottom (in)visible.

In the *Help* menu, you can

- go to the OpenMS website (*OpenMS website*)

- open this tutorial (*TOPPAS tutorial*)

Context menus:

In the context menu of an *input node*, you can

- specify the input files

- open the specified files in TOPPView

- open the input files' folder in the window manager (explorer)

- toggle the "recycling" mode

- copy, cut, and remove the node

In the context menu of a *tool*, you can

- configure the parameters of the tool

- resume the pipeline at this node

- open its temporary output files in TOPPView

- open the temporary output folder in the file manager (explorer)

- toggle the "recycling" mode

- copy, cut, and remove the node

In the context menu of a *Merger* or *Collector*, you can

- toggle the "recycling" mode

- copy, cut, and remove the node

In the context menu of an *output node*, you can

- open the output files in TOPPView

- open the output files' folder in the window manager (explorer)

- copy, cut, and remove the node

The following sections explain the example pipelines TOPPAS comes with. You can open them by selecting *File > Open example file*. All input files and parameters are already specified, so you can just hit *Pipeline > Run* (or press *F5*) and see what happens.

4.4   Profile data processing

The file *peakpicker_tutorial.toppas* contains a simple pipeline representing a common use case: starting with profile data, the noise is eliminated and the baseline is subtracted. Then, PeakPickerWavelet is used to find all peaks in the noise-filtered and baseline-reduced profile data. This workflow is also described in the section Profile data processing. The individual steps are explained in more detail in the TOPPView tutorial↩ : Smoothing raw data, Subtracting a baseline from a spectrum, and Picking peaks.
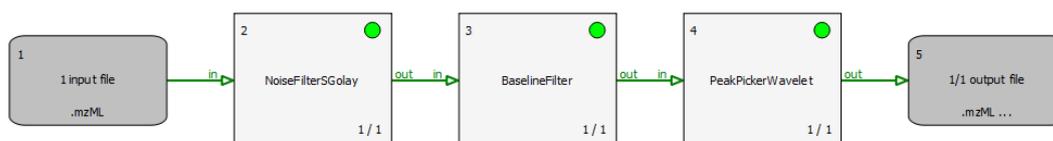
4.5   Identification of E. coli peptides

This section describes an example identification pipeline contained in the example directory, *Ecoli_Identification.↩ toppas*. It is shipped together with a reduced example mzML file containing 139 MS2 spectra from an E. coli run on an Orbitrap instrument as well as an E. coli target-decoy database.

We use the search engine OMSSA (Geer et al., 2004) for peptide identification. Therefore, OMSSA must be installed and the path to the OMSSA executable (omssacl) must be set in the parameters of the OMSSAAdapter node.

- Node #1 accepts mzML files containing MS2 spectra.

- Node #2 provides the database and is set to "recycling mode" to allow the database to be reused when there is more than one input file in node #1.

- OMSSAAdapter calls OMSSA which performs the actual search.

- PeptideIndexer annotates for each search result whether it is a target or a decoy hit.

- FalseDiscoveryRate computes q-values for the IDs.

- Finally, IDFilter selects only those IDs with a q-value of less than 1%.
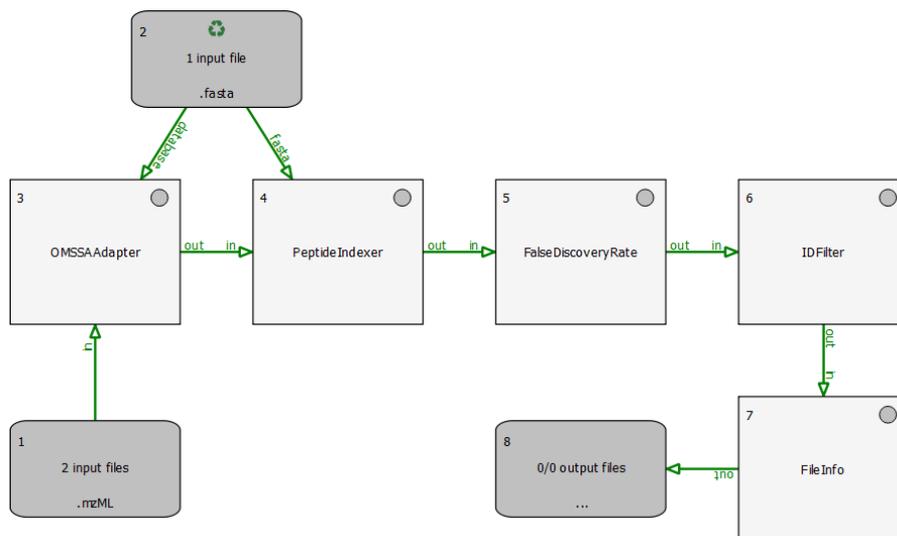
Extensions to this pipeline would be to do the annotation of the spectra with multiple search engines and combine the results afterwards, using the ConsensusID TOPP tool.

The results may be exported using the TextExporter tool, for further analysis with different tools.

4.6 Quantitation of BSA runs

The simple pipeline described in this section (*BSA_Quantitation.toppas*) can be used to quantify peptides that occur on different runs. The example dataset contains three different bovine serum albumin (BSA) runs. First, FeatureFinderCentroided is called since the dataset is centroided. The results of the feature finding are then annotated with (existing) identification results. For convenience, we provide these search results from OMSSA with peptides with an FDR of 5% in the BSA directory.
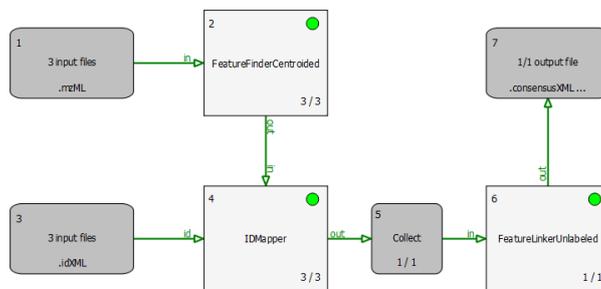
Identifications are mapped to features by the IDMapper. The last step is performed by FeatureLinkerUnlabeled which links corresponding features. The results can be used to calculate ratios, for example. The data could also be exported to a text based format using the TextExporter for further processing (e.g., in Microsoft Excel).

The results can be opened in TOPPView. The next figures show the results in 2D and 3D view, together with the feature intermediate results. One can see that the intensities and retention times are slightly different between the runs. To correct for retention times shift, a map alignment could be done, either on the spectral data or on the feature data.

YIC(Carbamidomethyl)DNQDTISSK

YIC(Carbamidomethyl)DNQDTISSK

1810
1800
1790
1780
1770
1760
1750
1740
1730
RT [se
0

722.5

725

30
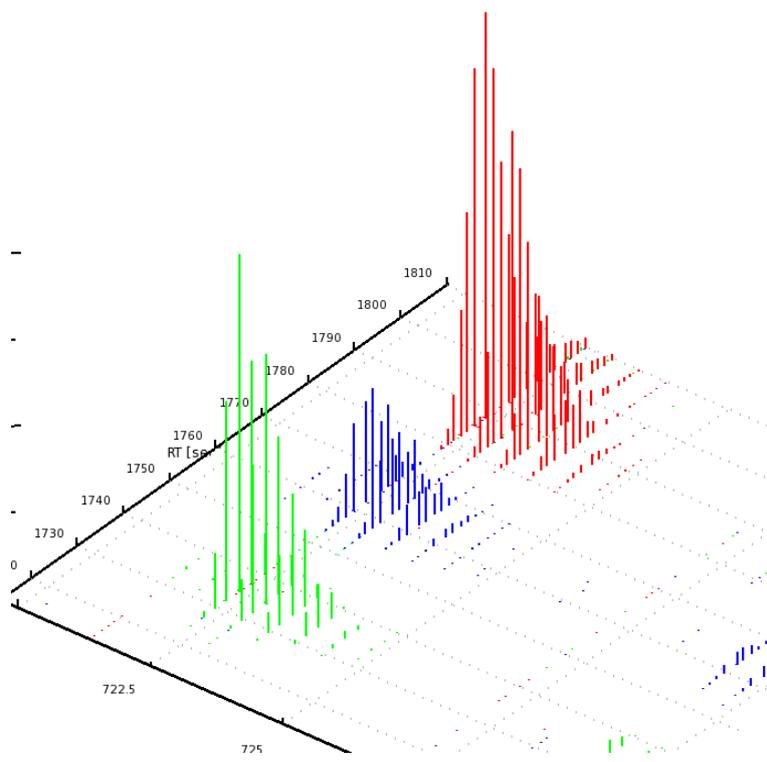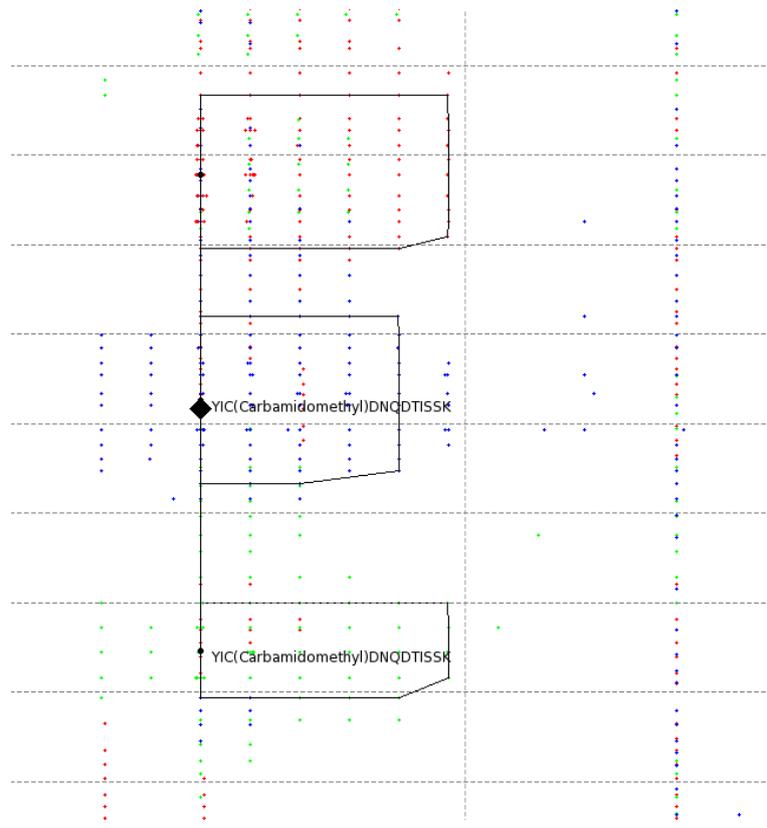
4.7 Merger and Collect nodes

The following example is actually not a useful workflow but is supposed to demonstrate how merger and collector nodes can be used in a pipeline. Have a look at *merger_tutorial.toppas:*
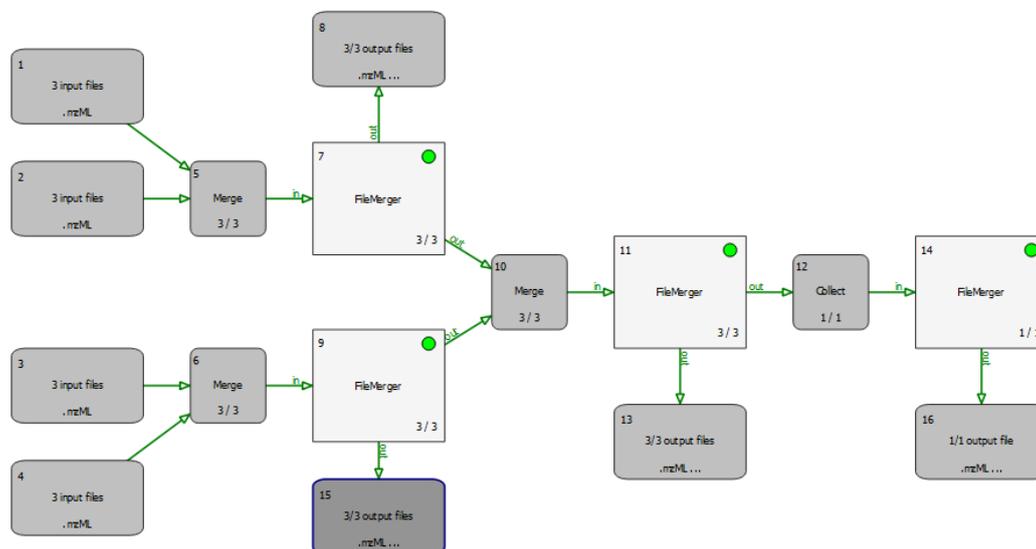
As its name suggests, a merger merges its incoming file lists, i.e., files of all incoming edges are appended into new lists (which have as many elements as the merger has incoming connections). All tools this merger has outgoing connections to are called with these merged lists as input files. All incoming connections should pass the same number of files (unless the corresponding preceding tool is in recycling mode).

A collector node, on the other hand, waits for all rounds to finish before concatenating all files from all incoming connections into one single list. It then calls the next tool with this list of files as input. This will happen exactly once during the entire pipeline run.

In order to track what is happening, you can just open the example file and run it. When the pipeline execution has finished, have a look at all input and output files (e.g., select *Open in TOPPView* in the context menu of the input/output nodes). The input files are named rt_1.mzML, rt_2.mzML, ... and each contains a single spectrum with RT as indicated by the filename, so you can easily see which files have been merged together.

5 Advanced Users: Tips & Tricks

We will not introduce you to some advanced concepts of TOPP, which will increase your productivity and easy usage.

Global database for search engine adapters

In your $HOME directory you will find an OpenMS.ini file in the .OpenMS subfolder. This INI file contains global parameters which are honored by many/all TOPP tools - depending on what the parameters refer to. The *id_db_dir* parameter allows you to specify one or more directories where you have placed FASTA files (or related, e.g., .psq files). If you now specify just a filename (without path) in an ID engine adapter, the filename will be looked up in the current working directory. If its not found, the directories specified in *id_db_dir* will be searched. This allows you to build scripts and/or TOPPAS pipelines which are portable across several computers - you just have to adapt the OpenMS.ini once on each machine.

Note when using TOPPAS: You can use an "input file" node to specify the FASTA file for several engines simultaneously. However, when selecting the file, TOPPAS will use the absolute pathname and the dialog will not allow you to name a non-existing file. After you've selected the file, you can however edit the filename and remove the path (this will issue a warning which you can ignore).

Note

> This approach does not work for our MascotAdapters, as each Mascot instance has its own database managed internally. You can however, make sure that the database is present on all mascot servers you are going to use, thus making the INI settings portable.

Using External tools in your workflows

OpenMS supports the wrapping of external tools (like msconvert from ProteoWizard), thus allowing you to build scripts and/or TOPPAS pipelines containing external tools.

6 Scripting with TOPP

This tutorial will give you a brief overview of the most important TOPP tools. First, we explain some basics that you will need for every TOPP tool, then we show several example pipelines.

6.1 File formats

The TOPP tools use the HUPO-PSI standard format mzML 1.1.0 as input format. In order to convert other open formats (mzData, mzXML, DTA, ANDI/MS) to mzML, a file converter is provided by TOPP.
Proprietary MS machine formats are not supported. If you need to convert these formats to mzML, mzData or mzXML, please have a look at the SASHIMI project page or contact your MS machine vendor.
mzML covers only the output of a mass spectrometry experiment. For further analysis of this data several other file formats are needed. The main file formats used by TOPP are:

- **mzML** The HUPO-PSI standard format for mass spectrometry data.

- **featureXML** The OpenMS format for quantitation results.

- **consensusXML** The OpenMS format for grouping features in one map or across several maps.

- **idXML** The OpenMS format for protein and peptide identification.

Documented schemas of the OpenMS formats can be found at http://www.openms.de/schemas/ .
idXML files and *consensusXML* files created by OpenMS can be visualized in a web browser directly. XSLT stylesheets are used to transform the XML to HTML code. The stylesheets are contained in the *OpenMS/share/↩ OpenMS/XSLT/* folder of your OpenMS installation.
If you want to view the file on the computer with the OpenMS installation, you can just open it in your browser.
If you copy the file to another computer, you have must copy the XSLT stylesheet to that computer and change the second line in the XML file. The following example shows how to change the stylesheet location for an idXML file. You simply have to change the *PATH* in the line
```
<?xml-stylesheet type="text/xsl" href="file:///*PATH*idXML.xsl"?>
```
to the folder where the stylesheet resides.

6.2 Common arguments of the TOPP tools

The command line and INI file parameters of the TOPP tools vary due to the different tasks of the TOPP tools. However, all TOPP tools share this common interface:

- **-ini** <file> Use the given TOPP INI file

- **-log** <file> Location of the log file (default: 'TOPP.log')

- **-instance** <n> Instance number in the TOPP INI file (default: '1')

- **-debug** <n> Sets the debug level (default: '0')

- **-write_ini** <file> Writes an example INI file

- **-no_progress** Disables progress logging to command line

- **--help** Shows a help page for the command line and INI file options

6.3 TOPP INI files

Each TOPP tool has its own set of parameters which can be specified at the command line. However, a more convenient (and persistent) way to handle larger sets of parameters is to use TOPP INI files. TOPP INI files are XML-based and can contain the configuration of one or several TOPP tools.

The following examples will give an overview of how TOPP tools can be chained in order to create analysis pipelines. INI files are the recommended way to store all settings of such a pipeline in a single place.

Note that the issue of finding suitable parameters for the tools is not addressed here. If you encounter problems during the execution of the example pipelines on your data, you probably have to adapt the parameters. Have a look at the documentation of the corresponding TOPP tool in that case.

6.3.1 Parameter documentation

General documentation of a TOPP tool and documentation for the command line parameters, can be displayed using the command line flag *–help*.

Some TOPP tools also have subsections of parameters that are internally handed to an algorithm. The documentation of these subsections is not displayed with *–help*. It is however displayed in **INIFileEditor** (see next section), or when using *–helphelp* (which also shows advanced parameters).

6.3.2 Creating an INI file for a TOPP tool

The easiest way of creating an INI file is to advise the corresponding TOPP tool to write its default configuration file using the argument '-write_ini' on the command line. Now the INI file can be adapted to your needs using **INIFileEditor**.
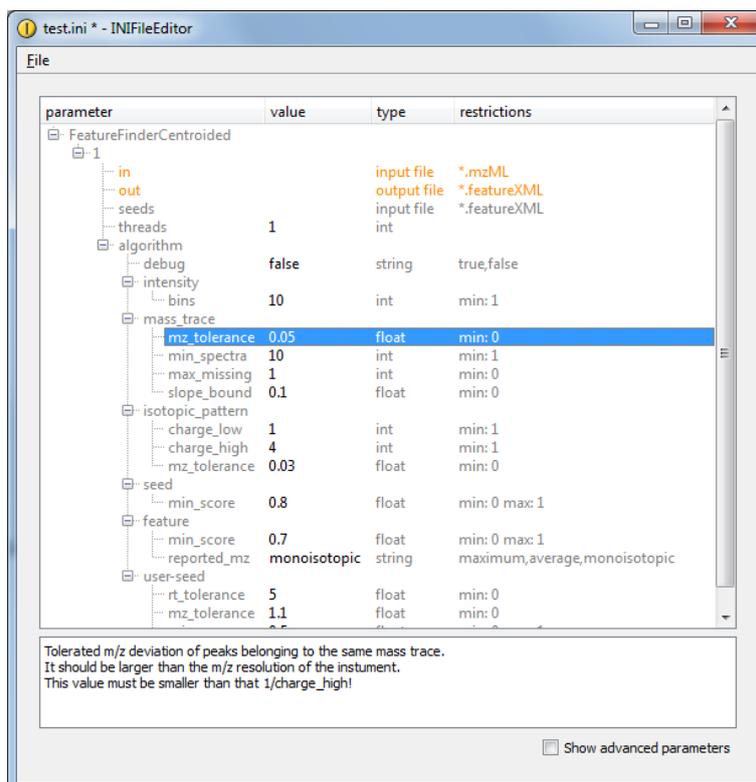
In the TOPP_INIFileEditor, the documentation of the parameters is displayed in the window at the bottom, once you click on the respective parameter.

6.3.3 Updating an INI file for a TOPP tool or a whole TOPPAS pipeline

If you have an old INI file which does not work for a newer OpenMS version (due to renamed/removed or new) parameters, you can rescue parameters whose name did not change into the new version by using our UTILS_IN↩ IUpdater tool by calling it with (a list of) outdated INI and/or TOPPAS files. See the INIUpdater tool description for details. This will remove invalid parameters and add new parameters (if available) while retaining values for unchanged parameters.

6.3.4 General structure of an INI file

An INI file is always enclosed by the *<PARAMETERS>* tag. Inside this tag, a tree-like hierarchy is created with *<NODE>* tags that represent sections and *<ITEM>* tags, each of which stores one of the parameters. The first two level of the hierarchy have a special meaning.
Example: Below is the content of an INI file for **FileFilter**.
Several parameter sets for a TOPP tool can be specified in a *tool section*. The tool section is always named after the program itself, in this case "FileFilter".

- In order to make storing several parameter sets for the same tool in one INI file possible, the tool section contains one or several *numbered instance subsections* ('1', '2', ...). These numbers are the instance numbers which can be specified using the '-instance' command line argument. (Remember the default is '1'.)

- Within each instance section, the actual parameters of the TOPP tool are given. INI files for complex tools can contain nested subsections in order to group related parameters.

- If a parameter is not found in the instance section, the *tool-specific common section* is considered.

- Finally, we look if the *general common section* contains a value for the parameter.

Imagine we call the **FileFilter** tool with the INI file given below and instance number '2'. The FileFilter parameters *rt* and *mz* are looked up by the tool. *mz* can be found in section **FileFilter** - *2*. *rt* is not specified in this section, thus the *common* - **FileFilter** section is checked first, where it is found in our example. When looking up the *debug* parameter, the tool would search the instance section and tool-specific common section without finding a value. Finally, the general *common* section would be checked, where the debug level is specified.

```
<PARAMETERS>
  <NODE name="FileFilter">
    <NODE name="1">
      <ITEM name="rt" value="0:1200" type="string"/>
    </NODE>
    <NODE name="2">
      <ITEM name="mz" value="700:1000" type="string"/>
    </NODE>
  </NODE>
  <NODE name="common">
    <NODE name="FileFilter">
      <ITEM name="rt" value=":" type="string"/>
      <ITEM name="mz" value=":" type="string"/>
    </NODE>
    <ITEM name="debug" value="2" type="int"/>
  </NODE>
</PARAMETERS>
```

6.4 General information about peak and feature maps

If you want some general information about a peak or feature map, use the **FileInfo** tool.

- It can print RT, m/z and intensity ranges, the overall number of peaks, and the distribution of MS levels

- It can print a statistical summary of intensities

- It can print some meta information

- It can validate XML files against their schema

- It can check for corrupt data in peak files See the 'FileInfo –help' for details.

6.5 Problems with input files

If you are experiencing problems while processing an XML file you can check if the file does validate against the XML schema:
```
FileInfo -v -in infile.mzML
```
Validation is available for several file formats including mzML, mzData, mzXML, featureXML and idXML.
Another frequently-occurring problem is corrupt data. You can check for corrupt data in peak files with **FileInfo** as well:
```
FileInfo -c -in infile.mzML
```

6.6 Converting your files to mzML

The TOPP tools work only on the HUPO-PSI *mzML* format. If you need to convert *mzData*, *mzXML* or *ANDI/MS* data to *mzML*, you can do that using the **FileConverter**, e.g.
```
FileConverter -in infile.mzXML -out outfile.mzML
```
If you use the format names as file extension, the tool derives the format from the extension. For other extensions, the file formats of the input and output file can be given explicitly.

6.7 Converting between DTA and mzML

Sequest DTA files can be extracted from a mzML file using the **DTAExtractor:**
```
DTAExtractor -in infile.mzML -out outfile
```
The retention time of a scan, the precursor mass-to-charge ratio (for MS/MS scans) and the file extension are appended to the output file name.
To combine several files (e.g. DTA files) to an mzML file use the **FileMerger:**
```
FileMerger -in infile_list.txt -out outfile.mzML
```
The retention times of the scans can be generated, taken from the *infile_list.txt* or can be extracted from the DTA file names. See the FileMerger documentation for details.

6.8 Extracting part of the data from a file

If you want to extract part of the data from an mzML file, you can use the **FileFilter** tool. It allows filtering for RT, m/z and intensity range or for MS level. To extract the MS/MS scans between retention time 100 and 1500, you would use the following command:
```
FileFilter -in infile.mzML -levels 2 -rt 100:1500 -out outfile.mzML
```

6.9 Profile data processing

Goal: You want to find all peaks in your profile data.

The first step shown here is the elimination of noise using a **NoiseFilter**. The now smoothed profile data can be further processed by subtracting the baseline with the **BaselineFilter**. Then use one of the **PeakPickers** to find all peaks in the baseline-reduced profile data.
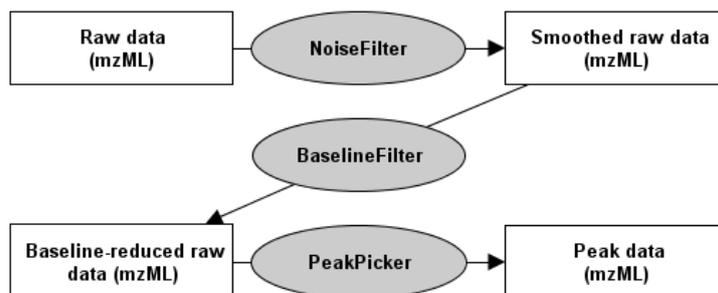
We offer two different smoothing filters: NoiseFilterGaussian and NoiseFilterSGolay. If you want to use the Savitzky Golay filter, or our **BaselineFilter** with non equally spaced profile data, e.g. TOF data, you have to generate equally spaced data using the **Resampler** tool.

6.10 Picking peaks with a PeakPicker

The **PeakPicker** tools allow for picking peaks in profile data. Currently, there are two different TOPP tools available, PeakPickerWavelet and PeakPickerHiRes.

PeakPickerWavelet   **Input data:** profile data (low/medium resolution)
Description: This peak picking algorithm uses the continuous wavelet transform of a raw data signal to detect mass peaks. Afterwards a given asymmetric peak function is fitted to the raw data and important peak parameters (e.g. fwhm) are extracted. In an optional step these parameters can be optimized using a non-linear optimization method.   The algorithm is described in detail in Lange et al. (2006) Proc. PSB-06.   **Application:**   This algorithm was designed for low and medium resolution data. It can also be applied to high-resolution data, but can be slow on large datasets.   See the PeakPickerCWT class documentation for a parameter list.

PeakPickerHiRes   **Input data:** profile data (high resolution)

Description:
This peak-picking algorithm detects ion signals in raw data and reconstructs the corresponding peak shape by cubic spline interpolation. Signal detection depends on the signal-to-noise ratio which is adjustable by the user (see parameter *signal_to_noise*). A picked peak's m/z and intensity value is given by the maximum of the underlying peak spline. Please notice that this method is still **experimental** since it has not been tested thoroughly yet.

Application:
The algorithm is best suited for high-resolution MS data (FT-ICR-MS, Orbitrap). In high-resolution data, the signals of ions with similar mass-to-charge ratios (m/z) exhibit little or no overlapping and therefore allow for a clear separation. Furthermore, ion signals tend to show well-defined peak shapes with narrow peak width. These properties facilitate a fast computation of picked peaks so that even large data sets can be processed very quickly.
See the PeakPickerHiRes class documentation for a parameter list.

6.11 Finding the right parameters for the

NoiseFilters, the BaselineFilter and the PeakPickers
Finding the right parameters is not trivial. The default parameters will not work on most datasets. In order to find good parameters, we propose the following procedure:

1. Load the data in TOPPView

2. Extract a single scan from the middle of the HPLC gradient (Right click on scan)

3. Experiment with the parameters until you have found the proper settings

- You can find the **NoiseFilters**, the **BaselineFilter**, and the **PeakPickers** in **TOPPView** in the menu 'Layer' - 'Apply TOPP tool'

6.12 Calibration

We offer two calibration methods: an internal and an external calibration. Both can handle peak data as well as profile data. If you want to calibrate profile data, a peak picking step is necessary, the important parameters can be set via the ini-file. If you have already picked data, don't forget the '`-peak_data`' flag.

The external calibration (**TOFCalibration**) is used to convert flight times into m/z- values with the help of external calibrant spectra containing e.g. a polymer like polylysine. For the calibrant spectra, the calibration constants the machine uses need to be known as well as the expected masses. Then a quadratic function is fitted to convert the flight times into m/z-values.

The internal calibration (**InternalCalibration**) uses reference masses in the spectra to correct the m/z-values using a linear function.

In a typical setting one would first pick the TOF-data, then perform the TOFCalibration and then the Internal↩
Calibration:

```
PeakPickerWavelet -in raw_tof.mzML -out picked_tof.mzML -ini pp.ini
TOFCalibration -in picked_tof.mzML -out picked.mzML -ext_calibrants ext_cal.mzML
              -ref_masses ext_cal_masses
              -tof_const tof_conv_consts -peak_data
InternalCalibration -in picked.mzML -out picked_calibrated.mzML
                    -ref_masses internal_calibrant_masses -peak_data
```

6.13 Map alignment

The goal of map alignment is to transform different HPLC-MS maps (or derived maps) to a common retention time axis. It corrects for shifted and scaled retention times, which may result from changes of the chromatography. The different **MapAligner** tools take n input maps, de-warp them and store the n de-warped maps. The following image shows the general procedure:
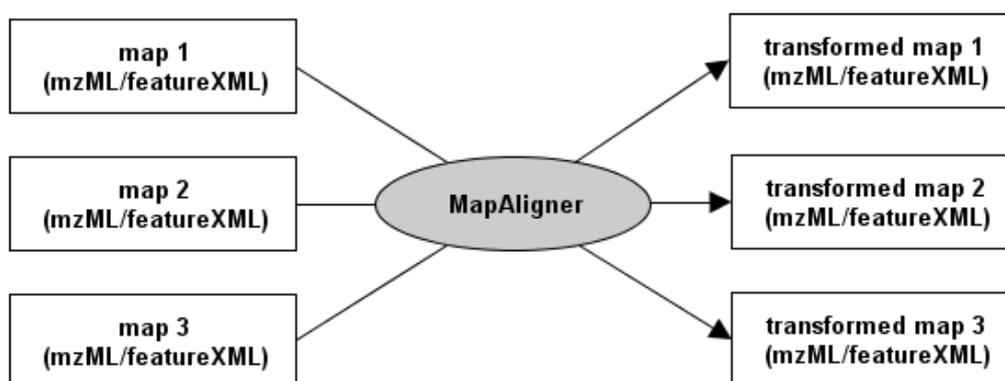
There are different map alignment tools available. The following table gives a rough overview of them:

Application: MapAlignerPoseClustering   **Applicable to:** feature maps, peak maps
Description:
This algorithm does a star-wise alignment of the input data. The center of the star is the map with most data points. All other maps are then aligned to the center map by estimating a linear transformation (shift and scaling) of retention times. The transformation is estimated using a pose clustering approach as described in doi:10.1093/bioinformatics/btm209

Application: MapAlignerIdentification   **Applicable to:** feature maps, consensus maps, identifications
Description:
This algorithm utilizes peptide identifications, and is thus applicable to files containing peptide IDs (idXML, annotated featureXML/consensusXML). It finds peptide sequences that different input files have in common and uses them as points of correspondence. From the retention times of these peptides, transformations are computed that convert each file to a consensus time scale.

Application: MapAlignerSpectrum   **Applicable to:** peak maps
Description:
This *experimental* algorithm uses a dynamic-programming approach based on spectrum similarity for the alignment. The resulting retention time mapping of dynamic-programming is then smoothed by fitting a spline to the retention time pairs.

Application: MapRTTransformer   **Applicable to:** peak maps, feature maps, consensus maps, identifications

Description:
This algorithm merely *applies* a set of transformations that are read from files (in TransformationXML format). These transformations might have been generated by a previous invocation of a MapAligner tool. For example, you might compute a transformation based on identifications and then apply it to the features or raw data. The transformation file format is not very complicated, so it is relatively easy to write (or generate) your own transformation files.

6.14 Feature detection

For quantitation, the **FeatureFinder** tools are used. They extract the features from profile data or centroided data. TOPP offers different types of **FeatureFinders:**

FeatureFinderIsotopeWavelet   **Input data:** profile data
Description: The algorithm has been designed to detect features in raw MS data sets. The current implementation is only able to handle MS1 data. An extension handling also tandem MS spectra is under development. The method is based on the *isotope wavelet*, which has been tailored to the detection of isotopic patterns following the averagine model. For more information about the theory behind this technique, please refer to Hussong et al.: "Efficient Analysis of Mass Spectrometry Data Using the Isotope Wavelet" (2007). Please note that this algorithm features no "modelling stage", since the structure of the isotopic pattern is explicitly coded by the wavelet itself. The algorithm also works for 2D maps (in combination with the so-called *sweep-line* technique (Schulz-↩ Trieglaff et al.: "A Fast and Accurate Algorithm for the Quantification of Peptides from Mass Spectrometry Data" (2007))). The algorithm can be executed on (several) high-speed CUDA graphics cards. Tests on real-world data sets revealed potential speedups beyond factors of 200 (using 2 NVIDIA Tesla cards in parallel). Please refer to Hussong et al.: "Highly accelerated feature detection in proteomics data sets using modern graphics processing units" (2009) for more details on the implementation. **Seeding:** Identification of regions of interest by convolving the signal with the wavelet function. A score, measuring the closeness of the transform to a theoretically determined output function, finally distinguishes potential features from noise. **Extension:** The extension is based on the sweep-line paradigm and is done on the fly after the wavelet transform. **Modelling:** None (explicitly done by the wavelet). See the FeatureFinderAlgorithmIsotopeWavelet class documentation for a parameter list.

FeatureFinderCentroided   **Input data:** peak data

Description:

This is an algorithm for feature detection based on peak data. In contrast to the other algorithms, it is based on peak/stick data, which makes it applicable even if no profile data is available. Another advantage is its speed due to the reduced amount of data after peak picking.

Seeding:

It identifies interesting regions by calculating a score for each peak based on

- the significance of the intensity in the local environment

- RT dimension: the quality of the mass trace in a local RT window

- m/z dimension: the quality of fit to an averagine isotope model

Extension:

The extension is based on a heuristics – the average slope of the mass trace for RT dimension, the best fit to averagine model in m/z dimension.

Modelling:

In model fitting, the retention time profile (Gaussian) of all mass traces is fitted to the data at the same time. After fitting, the data is truncated in RT and m/z dimension. The reported feature intensity is based on the fitted model, rather than on the (noisy) data.

See the FeatureFinderAlgorithmPicked class documentation for a parameter list.

6.15 Feature grouping

In order to quantify differences across maps (label-free) or within a map (isotope-labeled), groups of corresponding features have to be found. The **FeatureLinker** TOPP tools support both approaches. These groups are represented by consensus features, which contain information about the constituting features in the maps as well as average position, intensity, and charge.

6.16 Isotope-labeled quantitation

Goal: You want to differentially quantify the features of an isotope-labeled HPLC-MS map.
The first step in this pipeline is to find the features of the HPLC-MS map. The FeatureFinder applications calculate the features from profile data or centroided data.
In the second step, the labeled pairs (e.g. light/heavy labels of ICAT) are determined by the **FeatureLinker↩ Labeled** application. **FeatureLinkerLabeled** first determines all possible pairs according to a given optimal shift and deviations in RT and m/z. Then it resolves ambiguous pairs using a greedy-algorithm that prefers pairs with a higher score. The score of a pair is the product of:

- feature quality of feature 1

- feature quality of feature 2

- quality measure for the shift (how near is it to the optimal shift)

6.17 Label-free quantitation

Goal: You want to differentially quantify the features of two or more label-free HPLC-MS map.
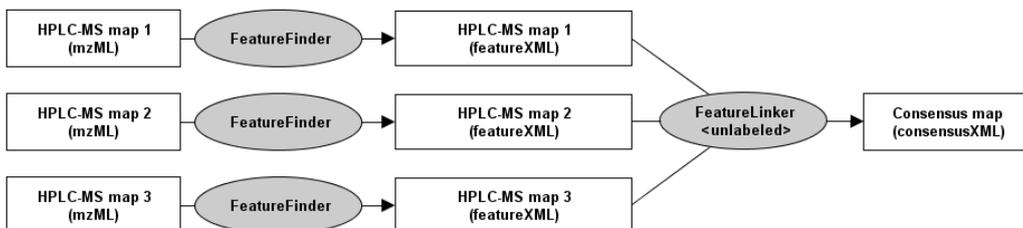
Note

This algorithm assumes that the retention time axes of all input maps are very similar. If you need to correct for retention time distortions, please have a look at Map alignment .

6.18   Consensus peptide identification

Goal: Use several identification engines in order to compute a consensus identification for a HPLC-MS\MS experiment.

OpenMS offers adapters for the following commercial and free peptide identification engines: Sequest, Mascot, OMSSA, PepNovo, XTandem and Inspect.

The adapters allow setting the input parameters and data for the identification engine and return the result in the OpenMS idXML format.

In order to improve the identification accuracy, several identification engines can be used and a consensus identification can be calculated from the results. The image below shows an example where Mascot and OMSSA results are fed to the **ConsensusID** tool (ConsensusID is currently usable for Mascot, OMSSA and XTandem).
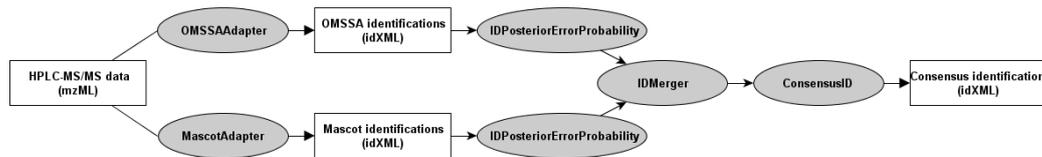
Goal: Combine quantitation and identification results.

Protein/peptide identifications can be annotated to quantitation results (featureXML, consensusXML) by the **ID↩ Mapper** tool. The combined results can then be exported by the **TextExporter** tool: Conversion between OpenMS XML formats and
.

6.19 Peptide property prediction

You can train a model for retention time prediction as well as for the prediction of proteotypic peptides.

Two applications has been described in the following publications: Nico Pfeifer, Andreas Leinenbach, Christian G. Huber and Oliver Kohlbacher Statistical learning of peptide retention behavior in chromatographic separations: A new kernel-based approach for computational proteomics. BMC Bioinformatics 2007, 8:468 Nico Pfeifer, Andreas Leinenbach, Christian G. Huber and Oliver Kohlbacher Improving Peptide Identification in Proteome Analysis by a Two-Dimensional Retention Time Filtering Approach J. Proteome Res. 2009, 8(8):4109-15

The predicted retention time can be used in IDFilter to filter out false identifications. Assume you have data from several identification runs. You should first align the data using MapAligner. Then you can use the various identification wrappers like MascotAdapter, OMSSAAdapter, ... to get the identifications. To train a model using RTModel you can now use IDFilter for one of the runs to get the high scoring identifications (40 to 200 distinct peptides should be enough). Then you use RTModel as described in the documentation to train a model for these spectra. With this model you can use RTPredict to predict the retention times for the remaining runs. The predicted retention times are stored in the idXML files. These predicted retention times can then be used to filter out false identifications using the IDFilter tool.

A typical sequence of TOPP tools would look like this:

```
MapAligner -in Run1.mzML,...,Run4.mzML -out Run1_aligned.mzML,...,Run4_aligned.mzML
MascotAdapter -in Run1_aligned.mzML -out Run1_aligned.idXML -ini Mascot.ini
MascotAdapter -in Run2_aligned.mzML -out Run2_aligned.idXML -ini Mascot.ini
MascotAdapter -in Run3_aligned.mzML -out Run3_aligned.idXML -ini Mascot.ini
MascotAdapter -in Run4_aligned.mzML -out Run4_aligned.idXML -ini Mascot.ini
IDFilter -in Run1_aligned.idXML -out Run1_best_hits.idXML -pep_fraction 1 -best_hits
RTModel -in Run1_best_hits.idXML -out Run1.model -ini RT.ini
RTPredict -in Run2_aligned.idXML -out Run2_predicted.idXML -svm_model Run1.model
RTPredict -in Run3_aligned.idXML -out Run3_predicted.idXML -svm_model Run1.model
RTPredict -in Run4_aligned.idXML -out Run4_predicted.idXML -svm_model Run1.model
IDFilter -in Run2_predicted.mzML -out Run2_filtered.mzML -rt_filtering
IDFilter -in Run3_predicted.mzML -out Run3_filtered.mzML -rt_filtering
IDFilter -in Run4_predicted.mzML -out Run4_filtered.mzML -rt_filtering
```

If you have a file with certainly identified peptides and want to train a model for RT prediction, you can also directly use the IDs. Therefore, the file has to have one peptide sequence together with the RT per line (separated by one tab or space). This can then be loaded by RTModel using the -textfile_input flag:

```
RTModel -in IDs_with_RTs.txt -out IDs_with_RTs.model -ini RT.ini -textfile_input
```

The likelihood of a peptide to be proteotypic can be predicted using PTModel and PTPredict. Assume we have a file PT.idXML which contains all proteotypic peptides of a set of proteins. Lets also assume, we have a fasta file containing the amino acid sequences of these proteins called mixture.fasta. To be able to train PTPredict, we need negative peptides (peptides, which are not proteotypic). Therefore, one can use the Digestor, which is located in the APPLICATIONS/UTILS/ folder together with the IDFilter:

```
Digestor -in mixture.fasta -out all.idXML
IDFilter -in all.idXML -out NonPT.idXML -exclusion_peptides_file PT.idXML
```

In this example the proteins are digested in silico and the non proteotypic peptides set is created by subtracting all proteotypic peptides from the set of all possible peptides. Then, one can train PTModel:

```
PTModel -in_positive PT.idXML -in_negative NonPT.idXML -out PT.model -ini PT.ini
```

6.20 Export of OpenMS XML formats

As TOPP offers no functionality for statistical analysis, this step is normally done using external statistics packages. In order to export the OpenMS XML formats into an appropriate format for these packages the TOPP **TextExporter** can be used.

It converts the the following OpenMS XML formats to text files:

- featureXML

- idXML

- consensusXML

The use of the **TextExporter** is very simple:
```
TextExporter -in infile.idXML -out outfile.txt
```

```
@section TOPP_example_convert_import_feature Import of feature data to OpenMS
```

OpenMS offers a lot of visualization and analysis functionality for feature data.
Feature data in text format, e.g. from other analysis tools, can be imported using the **TextImporter**. The default mode accepts comma separated values containing the following columns: RT, m/z, intensity. Additionally meta data columns may follow. If meta data is used, meta data column names have to be specified in a header line. Without headers:

```
1201  503.123 1435000
1201  1006.246  1235200
```

Or with headers:

```
RT  m/z Int isHeavy myMeta
1201  503.123 1435000 true  2
1201  1006.246  1235200 maybe 1
```

Example invocation:
```
TextImporter -in infile.txt -out outfile.featureXML
```
The tool also supports data from msInspect,SpecArray and Kroenik(Hardkloer sibling), just specify the -mode option accordingly.

6.21 Import of protein/peptide identification data to OpenMS

Peptide/protein identification data from several identification engines can be converted to idXML format using the **IDFileConverter** tool.

It can currently read the following formats:

- Sequest output folder

- pepXML file

- idXML file

It can currently write the following formats:

- pepXML

- idXML

This example shows how to convert pepXML to idXML:
```
IDFileConverter -in infile.pepXML -out outfile.idXML
```