

OPENMS


The OpenMS Developers

Mathias Walzer, Timo Sachsenberg, Fabian Aicheler,
Marc Rurik, Johannes Veit,
Bludau Isabell, Patrick Pedrioli,
Julianus Pfeuffer, Xiao Liang,
Knut Reinert, and Oliver Kohlbacher

Contents

1	General remarks	4
2	Getting started	5
2.1	Data conversion	5
2.2	Data visualization using TOPPView	5
2.3	Introduction to KNIME / OpenMS	8
2.3.1	KNIME concepts	8
2.3.2	Overview of the graphical user interface	11
2.3.3	Creating workflows	11
2.3.4	Sharing workflows	12
2.3.5	Duplicating workflows	12
2.3.6	A minimal workflow	13
2.3.7	Advanced topic: Meta nodes	15
2.3.8	Advanced topic: R integration	16
3	Metabolomics	18
3.1	Introduction	18
3.2	Quantifying metabolites across several experiments	18
3.3	Identifying metabolites in LC-MS/MS samples	21
3.4	Convert your data into a KNIME table	22
3.4.1	Bonus task: Visualizing data	23
3.5	Downstream data analysis and reporting	24
3.5.1	Data preparation ID	24
3.5.2	Data preparation Quant	24
3.5.3	Statistical analysis	25
3.5.4	Interactive visualization	26
3.5.5	Advanced visualization	27
3.5.6	Data preparation for Reporting	28

1 General remarks

- This handout will guide you through an introductory tutorial for the OpenMS/TOPP software package [1].
- OpenMS [2] is a versatile open-source library for mass spectrometry data analysis. Based on this library, we offer a collection of command-line tools ready to be used by end users. These so-called TOPP tools (short for “The OpenMS Proteomics Pipeline”) [3] can be understood as small building blocks of arbitrary complex data analysis workflows.
- In order to facilitate workflow construction, OpenMS was integrated into KNIME [4], the Konstanz Information Miner, an open-source integration platform providing a powerful and flexible workflow system combined with advanced data analytics, visualization, and report capabilities. Raw MS data as well as the results of data processing using TOPP can be visualized using TOPPView [5].
- In this hands-on tutorial session, you will become familiar with some of the basic functionalities of OpenMS/TOPP, TOPPView, and KNIME and learn how to use a selection of TOPP tools used in the tutorial workflows.
- All data referenced in this tutorial can be found in the  **Example_Data** folder that came with this tutorial.

2 Getting started

Before we get started we will install OpenMS and KNIME using the installers provided on the USB stick. Please choose the directory that matches your operating system and execute the installer. Note that these steps are not necessary if you use one of our laptops.

For example for Windows you call

- the OpenMS installer: `Windows / OpenMS-2.0_Win64_setup.exe`
- the KNIME installer: `Windows / OpenMS-2.0-prerequisites-installer.exe`
and `Windows / KNIME Full 3.1.1 Installer (64bit).exe`

on Mac you call

- the OpenMS installer: `Mac / OpenMS-2.0.0_setup.dmg`
- the KNIME installer: `Mac / knime-full_3.1.1.macosx.cocoa.x86_64.dmg`

and follow the instructions.

2.1 Data conversion

Each MS instrument vendor has one or more formats for storing the acquired data. Converting these data into an open format (preferably mzML) is the very first step when you want to work with open-source mass spectrometry software. A freely available conversion tool is ProteoWizard. The OpenMS installation package for Windows automatically installs ProteoWizard, so you do not need to download and install it separately.

Please note that due to restrictions from the instrument vendors, file format conversion for most formats is only possible on Windows systems, so exporting from the acquisition PC connected to the instrument is usually the most convenient option. All files used in this tutorial have already been converted to mzML by us, so you do not need to do it yourself.

2.2 Data visualization using TOPPView

Visualizing the data is the first step in quality control, an essential tool in understanding the data, and of course an essential step in pipeline development. OpenMS provides a

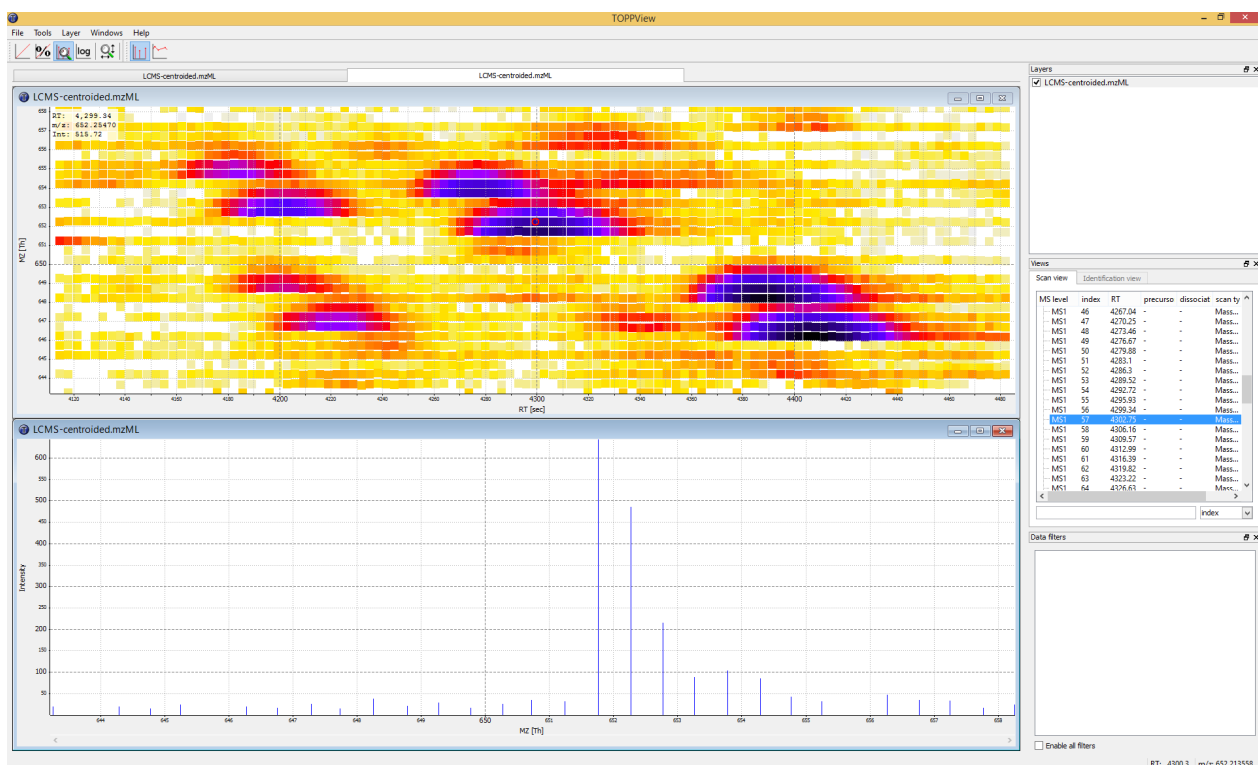


Figure 1: TOPPView, the graphical application for viewing mass spectra and analysis results. Top window shows a small region of a peak map. In this 2D representation of the measured spectra, signals of eluting peptides are colored according to the raw peak intensities. The lower window displays an extracted spectrum (=scan) from the peak map. On the right side, the list of spectra can be browsed.

convenient viewer for some of the data: **TOPPView**.

We will guide you through some of the basic features of **TOPPView**. Please familiarize yourself with the key controls and visualization methods. We will make use of these later throughout the tutorial. Let's start with a first look at one of the files of our tutorial data set:


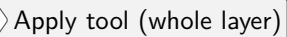
- Start **TOPPView** (see Start-Menu or Applications on MacOS)
- Go to **File** > **Open File**, navigate to the directory where you copied the contents of the USB stick to, and select **Example_Data** > **Introduction** > **datasets** > **small** > **velos005614.mzML**. This file contains a reduced LC-MS map (only a selected RT and m/z range was extracted using the TOPP tool **FileFilter**) of a label-free measurement of the human

platelet proteome recorded on an Orbitrap velos. The other two mzML files contain technical replicates of this experiment. First, we want to obtain a global view on the whole LC-MS map - the default option Map view 2D is the correct one and we can click the button.

- Play around.
- Three basic modes allow you to interact with the displayed data: scrolling, zooming and measuring:
 - Scroll mode
 - * Is activated by default (though each loaded spectra file is displayed zoomed out first, so you do not need to scroll).
 - * Allows you to browse your data by moving around in RT and m/z range.
 - * When zoomed in, to scroll the spectra map, click-drag on the current view.
 - * Arrow keys can be used to scroll the view as well.
 - Zoom mode
 - * Zooming into the data: either mark an area in the current view with your mouse while holding the left mouse button plus the key to zoom to this area or use your mouse wheel to zoom in and out.
 - * All previous zoom levels are stored in a zoom history. The zoom history can be traversed using + or + or the mouse wheel (scroll up and down).
 - * Pressing the Backspace key zooms out to show the full LC-MS map (and also resets the zoom history).
 - Measure mode
 - * It is activated using the 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.
- Right click on your 2D map and select and examine your data in 3D mode

- Go back to the 2D view. In 2D mode, visualize your data in different normalization modes, use linear, percentage and log-view (icons on the upper left tool bar).

Note: On Apple OS X, due to a bug in one of the external libraries used by OpenMS, you will see a small window of the 3D mode when switching to 2D. Close the 3D tab in order to get rid of it.

- In **TOPPView** you can also execute TOPP tools. Go to   and choose a TOPP tool (e.g., FileInfo) and inspect the results.

2.3 Introduction to KNIME / OpenMS

Using OpenMS in combination with KNIME you can create, edit, open, save, and run workflows combining TOPP tools with the powerful data analysis capabilities of KNIME. Workflows can be created conveniently in a graphical user interface. The parameters of all involved tools can be edited within the application and are also saved as part of the workflow. Furthermore, KNIME interactively performs validity checks during the workflow editing process, in order to make it more difficult to create an invalid workflow.

Throughout most of the parts of this tutorial you will use KNIME to create and execute workflows. This first step is to make yourself familiar with KNIME.

2.3.1 KNIME concepts

A workflow is a sequence of computational steps applied to a single or multiple input data sets to process and analyze the data. In KNIME such workflows are implemented graphically by combining so-called nodes. A node represents a single analysis step in a workflow. Nodes have input and output ports where the data enters the node or the results are provided for other nodes after processing, respectively. KNIME distinguishes between different port types, representing different types of data. The most common representation of data in KNIME are tables (similar to an excel sheet). Ports that accept tables are marked with a small triangle. For OpenMS we use a different port type, so called file ports, representing complete files. Those ports are marked by a small blue box. Filled blue boxes represent mandatory inputs and empty boxes optional inputs.

A typical OpenMS workflow in KNIME can be divided in two conceptually different parts:

- Nodes for signal and data processing, filtering and data reduction. Here, files are passed between nodes. Execution times of the individual steps are longer as the main computational steps are performed.
- Downstream statistical analysis and visualization. Here, tables are passed between nodes.

Between file-based processing and table-based analysis a conversion node typically performs the conversion from OpenMS results into KNIME tables.

Nodes can have three different states, indicated by the small traffic light below the node.

- Inactive, failed, and not yet fully configured nodes are marked red.
- Configured but not yet executed nodes are marked yellow.
- Successfully executed nodes are marked green.

If the node execution failed the node will switch to the red state.

Most nodes will be configured as soon as all input ports are connected. For some nodes additional parameters have to be provided that cannot be either guessed from the data or filled with sensible defaults. In this case, if you want to customize the default configuration, you can open the configuration dialog of a node with a double-click on the node. For OpenMS you will see a configuration dialog like the one shown in Figure 2.

Note: OpenMS distinguishes between normal parameters and advanced parameters. Advanced parameters are by default hidden from the users since they should only rarely be customized. In case you want to have a look at the parameters or need to customize them in one of the tutorials you can show them by clicking on the checkbox Show advanced parameter in the lower part of the dialog.

The dialog shows the individual parameters, their current value and type, and, in the lower part of the dialog, the documentation for the currently selected parameter.

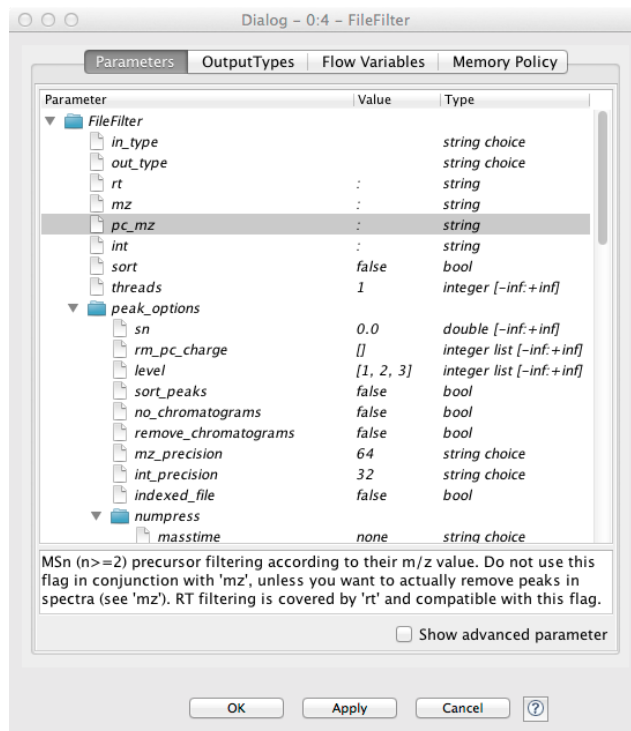


Figure 2: Node configuration dialog of an OpenMS node.

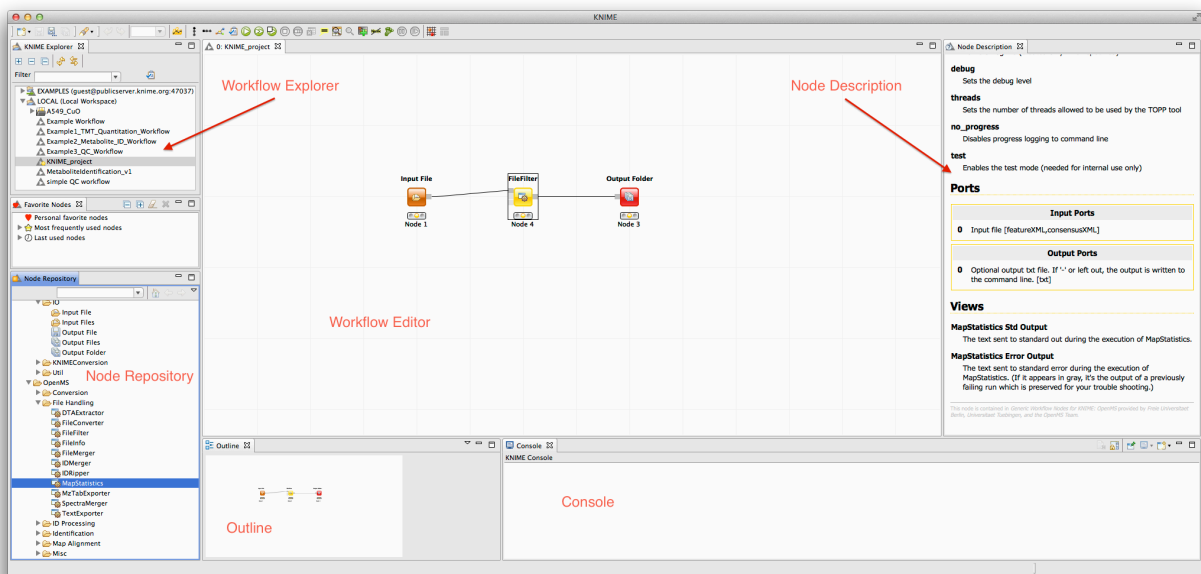


Figure 3: The KNIME workbench.

2.3.2 Overview of the graphical user interface

The graphical user interface (GUI) of KNIME consists of different components or so called panels that are shown in Figure 3. We will shortly introduce the individual panels and their purposes below.

Workflow Editor: The workflow editor is the central part of the KNIME GUI. Here you assemble the workflow by adding nodes from the Node Repository via "drag & drop". Nodes can be connected by clicking on the output port of one node and releasing the mouse at the desired input port of the next node.

Workflow Explorer: Shows a list of available workflows (also called workflow projects). You can open a workflow by double clicking it. A new workflow can be created with a right-click in the Workflow Explorer followed by selecting `New KNIME Workflow...`.

Node Repository: Shows all nodes that are available in your KNIME installation. Every plugin you install will provide new nodes that can be found here. The OpenMS nodes can be found in `Community Nodes >> OpenMS`. Nodes for managing files (e.g., Input Files or Output Folders) can be found in `Community Nodes >> GenericKnimeNodes`. You can search the node repository by typing the node name into the small text box in the upper part of the node repository.

Outline: The Outline panel contains a small overview of the complete workflow. While of limited use when working on a small workflow, this feature is very helpful as soon as the workflows get bigger.

Console: In the console panel warning and error messages are shown. This panel will provide helpful information if one of the nodes failed or shows a warning sign.

Node Description: As soon as a node is selected, the Node Description window will show the documentation of the node including documentation for all its parameters. For OpenMS nodes you will also find a link to the tool page in the online documentation.

2.3.3 Creating workflows

Workflows can easily be created by a right click in the Workflow Explorer followed by clicking on `New KNIME Workflow...`.

2.3.4 Sharing workflows

To be able to share a workflow with others, KNIME supports the import and export of complete workflows. To export a workflow, select it in the Workflow Explorer and select **File** > **Export KNIME Workflow...**. KNIME will export workflows as a zip file containing all the information on nodes, their connections, and their configuration. Those zip files can again be imported by selecting **File** > **Import KNIME Workflow...**.

Note: For your convenience we added all workflows discussed in this tutorial to the **Workflows** folder. If you want to check your own workflow by comparing it to the solution or got stuck, simply import the full workflow from the corresponding zip file.

2.3.5 Duplicating workflows

During the tutorial a lot of the workflows will be created based on the workflow from a previous task. To keep the intermediate workflows we suggest you create copies of your workflows so you can see the progress. To create a copy of your workflow follow the next steps.

- Right click on the workflow you want to create a copy of in the Workflow Explorer and select **Copy**.
- Right click again somewhere on the workflow explorer and select **Paste**.
- This will create a workflow with same name as the one you copied with a (2) appended.
- To distinguish them later on you can easily rename the workflows in the Workflow Explorer by right clicking on the workflow and selecting **Rename**.

Note: To rename a workflow it has to be closed.

2.3.6 A minimal workflow

Let us now start with the creation of our very first, very simple workflow. As a first step, we will gather some basic information about the data set before starting the actual development of a data analysis workflow.

- Create a new workflow.
- Add an **Input File** node and an **Output Folder** node (to be found in Community Nodes GenericKnimeNodes IO) and a **FileInfo** node (to be found in the category Community Nodes OpenMS File Handling) to the workflow.
- Connect the **Input File** node to the **FileInfo** node, and the first output port of the **FileInfo** node to the **Output Folder** node.

Note: In case you are unsure about which node port to use, hovering the cursor over the port in question will display the port name and what kind of input it expects.

The complete workflow is shown in Figure 4. FileInfo can produce two different kinds of output files.

- All nodes are still marked red, since we are missing an actual input file. Double-click the Input File node and select Browse. In the file system browser select Example_Data Introduction datasets tiny velos005614.mzML and click Open. Afterwards close the dialog by clicking Ok.

Note: Make sure to use the “tiny” version this time, not “small”, for the sake of faster workflow execution.

- The **Input File** node and the **FileInfo** node should now have switched to yellow, but the **Output Folder** node is still red. Double-click on the **Output Folder** node and click on Browse to select an output directory for the generated data.
- Great! Your first workflow is now ready to be run. Press ↑ + F7 to execute the complete workflow. You can also right click on any node of your workflow and select Execute from the context menu.



Figure 4: A minimal workflow calling FileInfo on a single file.

- The traffic lights tell you about the current status of all nodes in your workflow. Currently running tools show either a progress in percent or a moving blue bar, nodes waiting for data show the small word “queued”, and successfully executed ones become green. If something goes wrong (e.g., a tool crashes), the light will become red.
- In order to inspect the results, you can just right-click the **Output Folder** node and select `View: Open the output folder`. You can then open the text file and inspect its contents. You will find some basic information of the data contained in the mzML file, e.g., the total number of spectra and peaks, the RT and m/z range, and how many MS1 and MS2 spectra the file contains.

Workflows are typically constructed to process a large number of files automatically. As a simple example, consider you would like to gather this information for more than one file. We will now modify the workflow to compute the same information on three different files and then write the output files to a folder.

- We start from the previous workflow.
- First we need to replace our single input file with multiple files. Therefore we add the **Input Files** node from the category `Community Nodes >> GenericKnimeNodes >> IO`.
- To select the files we double-click on the **Input Files** node and click on `Add`. In the filesystem browser we select all three files from the directory `Example_Data > Introduction > datasets > tiny`. And close the dialog with `Ok`.
- We now add two more nodes: the **ZipLoopStart** and the **ZipLoopEnd** node from the category `Community Nodes >> GenericKnimeNodes >> Flow`.
- Afterwards we connect the **Input Files** node to the first port of the **ZipLoopStart** node, the first port of the **ZipLoopStart** node to the **FileInfo** node, the first output



Figure 5: A minimal workflow calling FileInfo on multiple files in a loop.

port of the **FileInfo** node to the first input port of the **ZipLoopEnd** node, and the first output port of the **ZipLoopEnd** node to the **Output Folder** node (NOT to the **Output File**). The complete workflow is shown in Figure 5

- The workflow is already complete. Simply execute the workflow and inspect the output as before.

In case you had trouble to understand what ZipLoopStart and ZipLoopEnd do - here is a brief explanation:

- The **Input Files** node passes a list of files to the **ZipLoopStart** node.
- The **ZipLoopStart** node takes the files as input, but passes the single files sequentially (that is: one after the other) to the next node.
- The **ZipLoopEnd** collects the single files that arrive at its input port. After all files have been processed, the collected files are passed again as file list to the next node that follows.

2.3.7 Advanced topic: Meta nodes

Workflows can get rather complex and may contain dozens or even hundreds of nodes. KNIME provides a simple way to improve handling and clarity of large workflows:

Meta Nodes allow to bundle several nodes into a single **Meta Node**.

Task



Select multiple nodes (e.g. all nodes of the ZipLoop including the start and end node). To select a set of nodes, draw a rectangle around them with the left mouse button or hold **Ctrl** to add/remove single nodes from the selection. Open the context menu (right-click on a node in the selection)

and select **Collapse into Meta Node**. Enter a caption for the **Meta Node**. The previously selected nodes are now contained in the **Meta Node**. Double clicking on the **Meta Node** will display the contained nodes in a new tab window.

Task



Undo the packaging. First select the **Meta Node**, open the context menu (right-click) and select **Expand Meta Node**.

2.3.8 Advanced topic: R integration

KNIME provides a large number of nodes for a wide range of statistical analysis, machine learning, data processing and visualization. Still, more recent statistical analysis methods, specialized visualizations or cutting edge algorithms may not be covered in KNIME. In order to expand its capabilities beyond the readily available nodes, external scripting languages can be integrated. In this tutorial, we primarily use scripts of the powerful statistical computing language R. Note that this part is considered advanced and might be difficult to follow if you are not familiar with R. In this case you might skip this part.

R View (Table) allows to seamlessly include R scripts into KNIME. We will demonstrate on a minimal example how such a script is integrated.

Task



First we need some example data in KNIME, which we will generate using the **Data Generator** node. You can keep the default settings and execute the node. The table contains 4 columns, each containing random coordinates and one column containing a cluster number (Cluster_0 to Cluster_3). Now place a **R View (Table)** node into the workflow and connect the upper output port of the **Data Generator** node to the input of the **R View (Table)** node. Right-click and configure the node.

If you get an error message like "Execute failed: R_HOME does not contain a folder with name 'bin'.": please change the R settings in the preferences. To do so open **File > Preferences > KNIME > R** and enter the path to your

R installation (the folder that contains the bin directory).

If R is correctly recognized we can start writing an R script. Consider that we are interested in plotting the first and second coordinates and color them according to their cluster number. In R this can be done in a single line.

In the **R View (Table)** text editor, enter the following code:

```
plot(x=knime.in$Universe_0_0, y=knime.in$Universe_0_1, main="Plotting column ↔  
Universe_0_0 vs. Universe_0_1", col=knime.in$"Cluster Membership")
```

Explanation: The table provided as input to the **R View (Table)** node is available as R **data.frame** with name **knime.in**. Columns (also listed on the left side of the R View window) can be accessed in the usual R way by first specifying the **data.frame** name and then the column name (e.g. **knime.in\$Universe_0_0**). **plot** is the plotting function we use to generate the image. We tell it to use the data in column **Universe_0_0** of the dataframe object **knime.in** (denoted as **knime.in\$Universe_0_1**) as x-coordinate and the other column **knime.in\$Universe_0_1** as y-coordinate in the plot. **main** is simply the main title of the plot and **col** the column that is used to determine the color (in this case it is the **Cluster Membership** column).

Now press the and buttons.

Note: Note that we needed to put some extra quotes around **Cluster Membership**. If we omit those, R would interpret the column name only up to the first space (**knime.in\$Cluster**) which is not present in the table and leads to an error. Quotes are regularly needed if column names contain spaces, tabs or other special characters like \$ itself.





3 Metabolomics

3.1 Introduction

Quantitation and identification of chemical compounds are basic tasks in metabolomic studies. In this tutorial session we construct a UPLC-MS based, label-free quantitation and identification workflow. Following quantitation and identification we then perform statistical downstream analysis to detect quantitation values that differ significantly between two conditions. This approach can, for example, be used to detect biomarkers. Here, we use two spike-in conditions of a dilution series (0.5 mg/l and 10.0 mg/l, male blood background, measured in triplicates) comprising seven isotopically labeled compounds. The goal of this tutorial is to detect and quantify these differential spike-in compounds against the complex background.

3.2 Quantifying metabolites across several experiments

For the metabolite quantification we choose an approach similar to the one used for peptides, but this time based on the OpenMS **FeatureFinderMetabo** method. This feature finder again collects peak picked data into individual mass traces. The reason why we need a different feature finder for metabolites lies in the step after trace detection: the aggregation of isotopic traces belonging to the same compound ion into the same feature. Compared to peptides with their average model, small molecules have very different isotopic distributions. To group small molecule mass traces correctly, an aggregation model tailored to small molecules is thus needed.

- Create a new workflow called for instance "Metabolomics".
- Add a **Input Files** node and configure it with all mzML files from  **Example_Data** ▶ **Metabolomics** ▶ **datasets**.
- Add a **ZipLoopStart** node and connect the **Input Files** node to the first port of the **ZipLoopStart** node.
- Add a **FeatureFinderMetabo** node (from  **Community Nodes** >>  **OpenMS** >>  **Quantitation**) and connect the first output port of the **ZipLoopStart** to the **FeatureFinderMetabo**.

- For an optimal result adjust the following settings. Please note that some of these are advanced parameters.

parameter	value
algorithm → common → chrom_fwhm	8.0
algorithm → mtd → trace_termination_criterion	sample_rate
algorithm → mtd → min_trace_length	3.0
algorithm → mtd → max_trace_length	600.0
algorithm → epd → width_filtering	off

- Add a **ZipLoopEnd** node and connect the output of the **FeatureFinderMetabo** to the first port of the **ZipLoopEnd** node.

To facilitate the collection of features corresponding to the same compound ion across different samples, an alignment of the samples' feature maps along retention time is often helpful. In addition to local, small-scale elution differences, one can often see constant retention time shifts across large sections between samples. We can use linear transformations to correct for these large scale retention differences. This brings the majority of corresponding compound ions close to each other. Finding the correct corresponding ions is then faster and easier, as we don't have to search as far around individual features.

- After the **ZipLoopEnd** node add a **MapAlignerPoseClustering** node (Community Nodes > OpenMS > Map Alignment), set its Output Type to featureXML, and adjust the following settings

parameter	value
algorithm → max_num_peaks_considered	-1
algorithm → superimposer → mz_pair_max_distance	0.005
algorithm → superimposer → num_used_points	10000
algorithm → pairfinder → distance_RT → max_difference	20.0
algorithm → pairfinder → distance_MZ → max_difference	20.0
algorithm → pairfinder → distance_MZ → unit	ppm

The next step after retention time correction is the grouping of corresponding features in multiple samples. In contrast to the previous alignment, we assume no linear relations of features across samples. The used method is tolerant against local swaps in elution order.

- After the **MapAlignerPoseClustering** add a **FeatureLinkerUnlabeledQT** (Community Nodes >> OpenMS >> Map Alignment) and adjust the following settings

parameter	value
algorithm → distance_RT → max_difference	40.0
algorithm → distance_MZ → max_difference	20.0
algorithm → distance_MZ → unit	ppm

- After the **FeatureLinkerUnlabeledQT** add a **TextExporter** node (Community Nodes >> OpenMS >> File Handling).
- Add an **Output Folder** node and configure it with an output directory where you want to store the resulting files.
- Run the pipeline and inspect the output.

You should find a single, tab-separated file containing the information on where metabolites were found and with which intensities. You can also add **Output Folder** nodes at different stages of the workflow and inspect the intermediate results (e.g., identified metabolite features for each input map). The complete workflow can be seen in Figure 6. In the following section we will try to identify those metabolites.

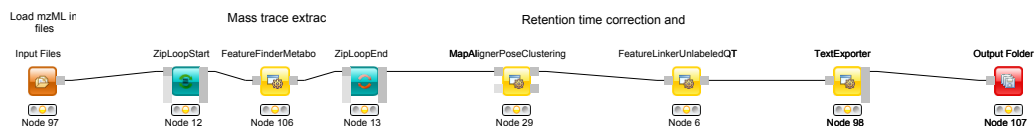


Figure 6: Label-free quantification workflow for metabolites

3.3 Identifying metabolites in LC-MS/MS samples

At the current state we found several metabolites in the individual maps but so far don't know what they are. To identify metabolites OpenMS provides multiple tools, including search by mass: the **AccurateMassSearch** node searches observed masses against the Human Metabolome Database (HMDB)[6, 7, 8]. We start with the workflow from the previous section (see Figure 6).

- Add a **FileConverter** node and connect the output of the **FeatureLinkerUnlabeledQT** to the incoming port.
- Open the Configure dialog of the **FileConverter** and select the tab "OutputTypes". In the drop down list for FileConverter.1.out select "featureXML".
- Add an **AccurateMassSearch** node and connect the output of the **FileConverter** to the first port of the **AccurateMassSearch**.
- Add four **Input File** nodes and configure them with the following files
 - **Example_Data** ▶ **Metabolomics** ▶ **databases** ▶ **PositiveAdducts.tsv**
This file specifies the list of adducts that are considered in the positive mode. Each line contains the formula and charge of an adduct separated by a semi-colon (e.g. M+H;1+). The mass of the adduct is calculated automatically.
 - **Example_Data** ▶ **Metabolomics** ▶ **databases** ▶ **NegativeAdducts.tsv**
This file specifies the list of adducts that are considered in the negative mode analogous to the positive mode.
 - **Example_Data** ▶ **Metabolomics** ▶ **databases** ▶ **HMDBMappingFile.tsv**
This file contains information from a metabolite database in this case from HMDB. It has three (or more) tab-separated columns: mass, formula, and identifier(s). This allows for an efficient search by mass.
 - **Example_Data** ▶ **Metabolomics** ▶ **databases** ▶ **HMDB2StructMapping.tsv**
This file contains additional information about the identifiers in the mapping file. It has four tab-separated columns that contain the identifier, name, SMILES, and INCHI. These will be included in the result file. The identifiers in this file must match the identifiers in the HMDBMappingFile.tsv.

- In the same order as they are given above connect them to the remaining input ports of the **AccurateMassSearch** node.
- Add an **Output Folder** node and connect the first output port of the **AccurateMassSearch** node to the **Output Folder**.

The result of the **AccurateMassSearch** node is in the mzTab format [9] so you can easily open it in a text editor or import it into Excel or KNIME, which we will do in the next section. The complete workflow from this section is shown in Figure 7.

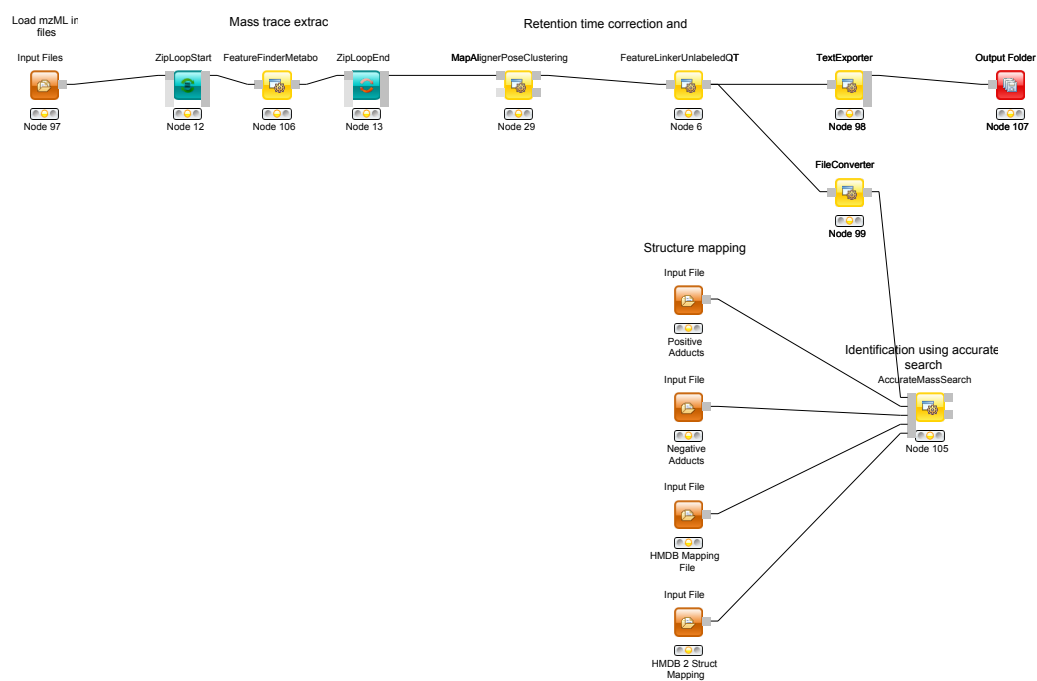


Figure 7: Label-free quantification and identification workflow for metabolites

3.4 Convert your data into a KNIME table

The result from the **TextExporter** node as well as the result from the **AccurateMassSearch** node are files while standard KNIME nodes display and processes only KNIME tables. To convert these files into KNIME tables we need two different nodes. For the **AccurateMassSearch** results we use the **MzTabReader** node (Community Nodes >> OpenMS >> Conversion >> mzTab), for

the result of the **TextExporter** we use the **ConsensusTextReader** (Community Nodes >> OpenMS >> Conversion).

When executed, both nodes will import the OpenMS files and provide access to the data as KNIME tables. You can now easily combine both tables using the **Joiner** node (Data Manipulation >> Column >> Split & Combine) and configuring it to match the m/z and retention time values of the respective tables. The full workflow is shown in Figure 8.

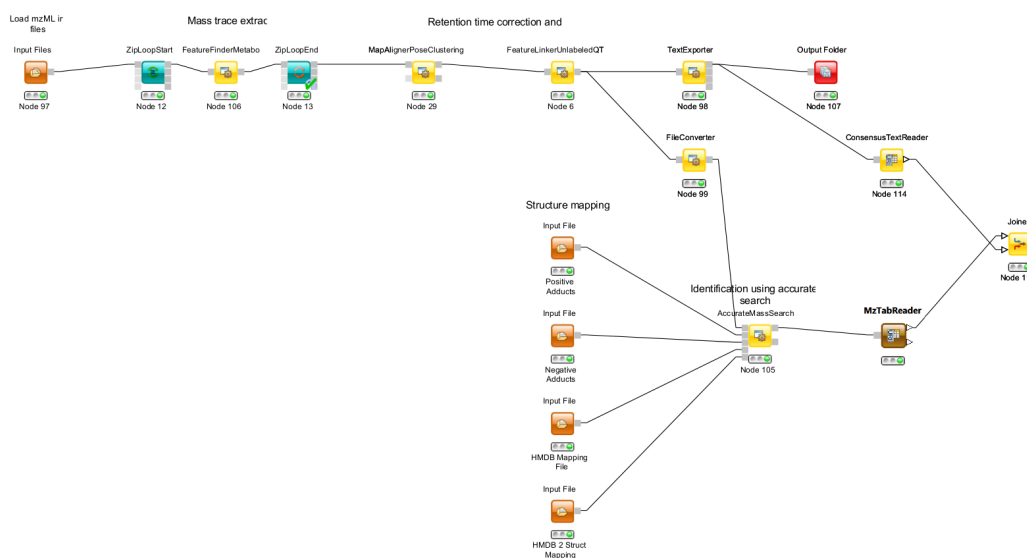


Figure 8: Label-free quantification and identification workflow for metabolites that loads the results into KNIME and joins the tables.

3.4.1 Bonus task: Visualizing data

Now that you have your data in KNIME you should try to get a feeling for the capabilities of KNIME.

Task

- ☑ Check out the **Molecule Type Cast** node (Chemistry >> Translators) together with subsequent cheminformatics nodes (e.g. **RDKit From Molecule** (Community Nodes >> RDKit >> Converters)) to render the structural formula contained in the result table.

Task



Have a look at the **Column Filter** node to reduce the table to the interesting columns, e.g., only the Ids, chemical formula, and intensities.

Task



Try to compute and visualize the m/z and retention time error of the different elements of the consensus features.

3.5 Downstream data analysis and reporting

In this part of the metabolomics session we take a look at more advanced downstream analysis and the use of the statistical programming language R. As laid out in the introduction we try to detect a set of spike-in compounds against a complex blood background. As there are many ways to perform this type of analysis we provide a complete workflow.

Task



Import the workflow from  **Workflows** › **metabolite_ID.zip** in KNIME: 


The section below will guide you in your understanding of the different parts of the workflow. Once you understood the workflow you should play around and be creative. Maybe create a novel visualization in KNIME or R? Do some more elaborate statistical analysis? Feel free to experiment and show us your results if you like. Note that some basic R knowledge is required to fully understand the processing in **R Snippet** nodes.

3.5.1 Data preparation ID

This part is analogous to what you did for the simple metabolomics pipeline.

3.5.2 Data preparation Quant

The first part is identical to what you did for the simple metabolomics pipeline. Additionally, we convert zero intensities into NA values and remove all rows that contain at least

one NA value from the analysis. We do this using a very simple **R Snippet** and subsequent **Missing Value filter** node.

Task



Inspect the **R Snippet** by double-clicking on it. The KNIME table that is passed to an **R Snippet** node is available in R as a data.frame named `knime.in`. The result of this node will be read from the data.frame `knime.out` after the script finishes. Try to understand and evaluate parts of the script (Eval Selection). In this dialog you can also print intermediary results using for example the R command `head()` or `cat()` to the Console pane.

3.5.3 Statistical analysis

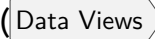
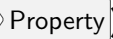
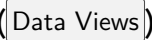

After we linked features across all maps, we want to identify features that are significantly deregulated between the two conditions. We will first scale and normalize the data, then perform a t-test, and finally correct the obtained p-values for multiple testing using Benjamini-Hochberg. All of these steps will be carried out in individual **R Snippet** nodes.

- Double-click on the first **R Snippet** node labeled "log scaling" to open the **R Snippet** dialog. In the middle you will see a short R script that performs the log scaling. To perform the log scaling we use a so-called regular expression (`grep`) to select all columns containing the intensities in the six maps and take the \log_2 logarithm.
- The output of the log scaling node is also used to draw a boxplot that can be used to examine the structure of the data. Since we only want to plot the intensities in the different maps (and not `m/z` or `rt`) we first use a **Column Filter** node to keep only the columns that contain the intensities. We connect the resulting table to a **Box Plot** node which draws one box for every column in the input table. Right-click and select `View: Box Plot`.
- The median normalization is performed in a similar way to the log scaling. First we calculate the median intensity for each intensity column, then we subtract the median from every intensity.

- Open the **Box Plot** connected to the normalization node and compare it to the box plot connected to the log scaling node to examine the effect of the median normalization.
- To perform the t-test we defined the two groups we want to compare. Then we call the t-test for every consensus feature unless it has missing values. Finally we save the p-values and fold-changes in two new columns named p-value and FC.
- The **Numeric Row Splitter** is used to filter less interesting parts of the data. In this case we only keep columns where the fold-change is ≥ 2 .
- We adjust the p-values for multiple testing using Benjamini-Hochberg and keep all consensus features with a q-value ≤ 0.01 (i.e. we target a false-discovery rate of 1%).

3.5.4 Interactive visualization

KNIME supports multiple nodes for interactive visualization with interrelated output. The nodes used in this part of the workflow exemplify this concept. They further demonstrate how figures with data dependent customization can be easily realized using basic KNIME nodes. Several simple operations are concatenated in order to enable an interactive volcano plot.

- We first log-transform fold changes and p-values in the **R Snippet** node. We then append columns noting interesting features (concerning fold change and p-value).
- With this information, we can use various Manager nodes ( ) to emphasize interesting data points. The configuration dialogs allow us to select columns to change color, shape or size of data points dependent on the column values.
- The **Scatter Plot** node () enables interactive visualization of the logarithmized values as a volcano plot: the log-transformed values can be chosen in the 'Column Selection' tab of the plot view. Data points can be selected in the plot and HiLited via the menu option. HiLiteing transfers to all other interactive nodes connected to the same data table. In our case, selection and HiLiteing will also occur in the **Interactive Table** node ()

- Output of the interactive table can then be filtered via the HiLite menu tab. For example, we could restrict shown rows to points HiLited in the volcano plot.

Task



Inspect the nodes of this section. Customize your visualization and possibly try to visualize other aspects of your data.

3.5.5 Advanced visualization

R Dependencies: This section requires that the R packages `ggplot2` and `ggbiplot` are both installed. `ggplot2` is part of the KNIME R Statistics Integration (Windows Binaries) which should already be installed via the full KNIME installer, `ggbiplot` however is not. In case that you use an R installation where one or both of them are not yet installed, add an **R Snippet** node and double-click to configure. In the R Script text editor, enter the following code:

```
#Include the next line if you also have to install ggplot2:  
install.packages("ggplot2")  
#Include the following lines to install ggbiplot:  
install.packages("devtools")  
library(devtools)  
install_github("vqv/ggbiplot")
```

Press to execute the script.

Even though the basic capabilities for (interactive) plots in KNIME are valuable for initial data exploration, professional looking depiction of analysis results often relies on dedicated plotting libraries. The statistics language R supports the addition of a large variety of packages, including packages providing extensive plotting capabilities. This part of the workflow shows how to use R nodes in KNIME to visualize more advanced figures. Specifically, we make use of different plotting packages to realize heatmaps.

- The used **RView (Table)** nodes combine the possibility to write R snippet code with visualization capabilities inside KNIME. Resulting images can be looked at in the output RView, or saved via the **Image Port Writer** node.

- The heatmap nodes make use of the `gplots` library, which is by default part of the R Windows binaries for the KNIME 3.1.1 full installation. We again use regular expressions to extract all measured intensity columns for plotting. For clarity, feature names are only shown in the heatmap after filtering by fold changes.

3.5.6 Data preparation for Reporting

Following the identification, quantification and statistical analysis our data is merged and formatted for reporting. First we want to discard our normalized and logarithmized intensity values in favor of the original ones. To this end we first remove the intensity columns (**Column Filter**) and add the original intensities back (**Joiner**). Note that we use an Inner Join¹. Combining ID and Quantification table into a single table is again achieved using a **Joiner** node.

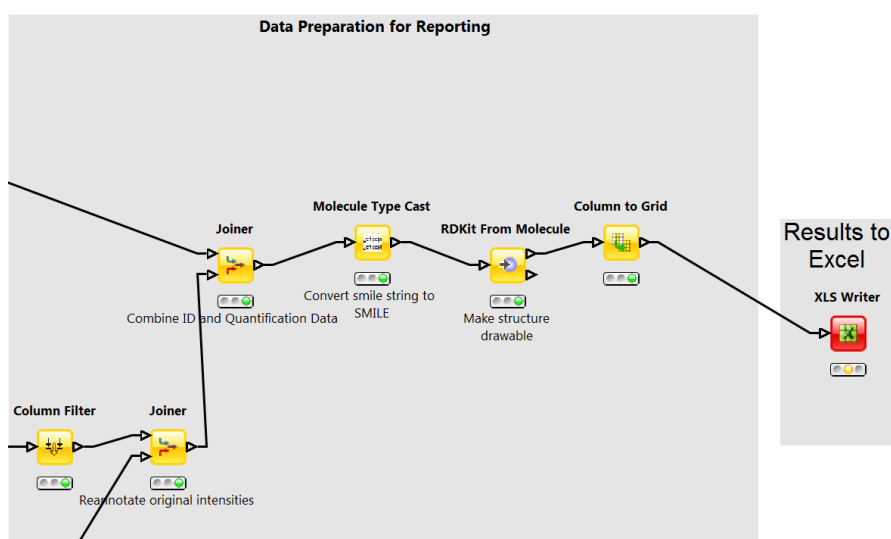


Figure 9: Data preparation for reporting

Question



What happens if we use an Left Outer Join, Right Outer Join or Full Outer Join instead of the Inner Join?

¹Inner Join is a technical term that describes how database tables are merged.

Task



- Inspect the output of the join operation after the Molecule Type Cast and RDKit molecular structure generation.

While all relevant information is now contained in our table the presentation could be improved. Currently, we have several rows corresponding to a single consensus feature (=linked feature) but with different, alternative identifications. It would be more convenient to have only one row for each consensus feature with all accurate mass identifications added as additional columns. To this end, we use the **Column to Grid** node that flattens several rows with the same consensus number into a single one. Note that we have to specify the maximum number of columns in the grid so we set this to a large value (e.g. 100). We finally export the data to an Excel file (**XLS Writer**).

References

- [1] OpenMS, OpenMS home page [online]. 4
- [2] M. Sturm, A. Bertsch, C. Gröpl, A. Hildebrandt, R. Hussong, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, A. Zerck, K. Reinert, and O. Kohlbacher, OpenMS - an open-source software framework for mass spectrometry., *BMC bioinformatics* 9(1) (2008), doi:10.1186/1471-2105-9-163. 4
- [3] O. Kohlbacher, K. Reinert, C. Gröpl, E. Lange, N. Pfeifer, O. Schulz-Trieglaff, and M. Sturm, TOPP—the OpenMS proteomics pipeline., *Bioinformatics* 23(2) (Jan. 2007). 4
- [4] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, C. Sieb, K. Thiel, and B. Wiswedel, KNIME: The Konstanz Information Miner, in *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*, Springer, 2007. 4
- [5] M. Sturm and O. Kohlbacher, TOPPView: An Open-Source Viewer for Mass Spectrometry Data, *Journal of proteome research* 8(7), 3760–3763 (July 2009), doi: 10.1021/pr900171m. 4
- [6] D. S. Wishart, D. Tzur, C. Knox, et al., HMDB: the Human Metabolome Database, *Nucleic Acids Res* 35(Database issue), D521–6 (Jan 2007), doi:10.1093/nar/gkl923. 21
- [7] D. S. Wishart, C. Knox, A. C. Guo, et al., HMDB: a knowledgebase for the human metabolome, *Nucleic Acids Res* 37(Database issue), D603–10 (Jan 2009), doi:10.1093/nar/gkn810. 21
- [8] D. S. Wishart, T. Jewison, A. C. Guo, M. Wilson, C. Knox, et al., HMDB 3.0—The Human Metabolome Database in 2013, *Nucleic Acids Res* 41(Database issue), D801–7 (Jan 2013), doi:10.1093/nar/gks1065. 21
- [9] J. Griss, A. R. Jones, T. Sachsenberg, M. Walzer, L. Gatto, J. Hartler, G. G. Thallinger, R. M. Salek, C. Steinbeck, N. Neuhauser, J. Cox, S. Neumann, J. Fan, F. Reisinger, Q.-W. Xu, N. Del Toro, Y. Perez-Riverol, F. Ghali, N. Bandeira, I. Xenarios, O. Kohlbacher, J. A.

Vizcaino, and H. Hermjakob, The mzTab Data Exchange Format: communicating MS-based proteomics and metabolomics experimental results to a wider audience, Mol Cell Proteomics (Jun 2014), doi:10.1074/mcp.O113.036681. 22