KniMet – A pipeline for the processing of chromatography-mass spectrometry metabolomics data

User Guide

Table of Contents

Introduction

KniMet is a workflow for the post-processing of LC-MS, GC-MS and LC-IM-MS metabolomics data. It is based on the KNIME analytical platform (Berthold *et al.*, 2007) with integrated R (R Core Team, 2014) nodes, and allows the user to perform missing values imputation (MVI), feature filtering, normalisation, batch correction and feature annotation.

In the following, you will find instructions on how to install KniMet and all the dependencies (Installation), a detailed description on how to configure and perform each processing step (What does it do?), and finally a series of Guided Examples to perform using the example data provided in the `testfiles` folder of this repository.

Installation

1. Install R

Download and install R from your preferred CRAN mirror, where you will find both the precompiled binary distributions for different operative systems, and the instructions for installation. Please refer to the R user guide, which can be found here.

Once the R installation has completed, the package 'Rserve' (Urbanek, 2013) needs to be installed to allow the interaction between R and KNIME. Please install it in R using

```
install.packages('Rserve')
```

2. Install OpenMS

OpenMS is a library providing a set of command-line tools as well as KNIME nodes to analyse MS data. OpenMS nodes have been included into KniMet to perform annotation based on accurate mass.

To install it, download the installer for the platform you are working on from https://www.openms.de/download/openms-binaries/

3. Install KNIME

Download and install the KNIME version comprising all free extensions from here; please refer to the getting started page where instructions regarding the download, installation and usage are provided.

4. Download KniMet and import it in KNIME

Once the `KniMet.knwf` file has been downloaded, it needs to be imported into KNIME: click on `File → Import KNIME workflow…`, in the window that opens tick `Select File` and browse to the directory where you saved the `KniMet.knwf`, select it, and click `Finish`.
The KniMet workflow will now be visible on the `KINME Explorer` pane (*Figure 1*). Apart from the KniMet workflow, you should also download the `testfiles` and `LipidMaps_Annotation` folders, containing respectively some input files to test the pipeline (more on this on the Guided Examples part) and the files to perform annotation based on accurate mass obtained with an in-house script from the LIPID MAPS Structure Database (LMSD) (Fahy *et al.*, 2007).

By double clicking on the KniMet workflow, you might be prompted to an error window asking for installation of the missing and required extensions: click on `Yes` and then in the `Install` window that opens select `Next >`, `I accept the terms of the licence agreements` and `Finish`. You will need to restart KNIME to apply the changes. When you open again KniMet after

the restart, you might get a window saying `Warning during load Reason: KniMet loaded with warnings`. This is caused by previous configurations; it can be ignored by clicking on `OK`.

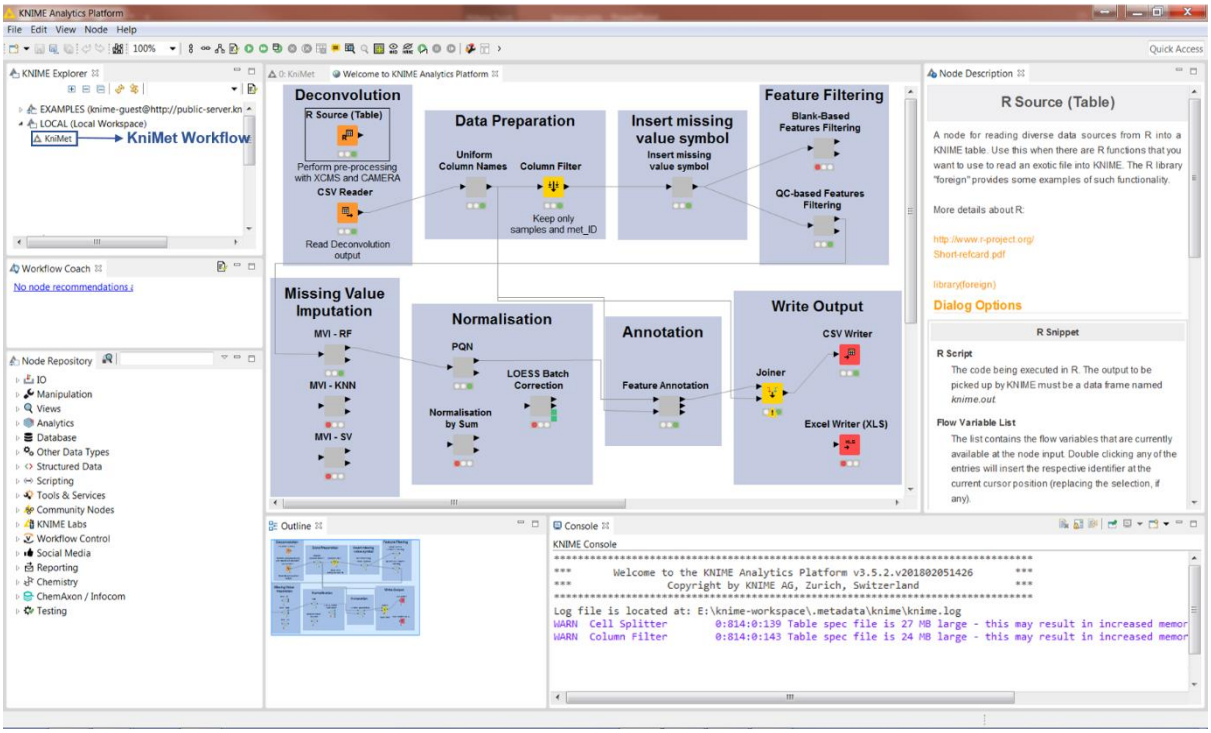

Figure 1: The KniMet workflow is highlighted on the KNIME Explorer pane, while on the central pane the KniMet pipeline is shown

5. Point KNIME to use the local version of R:

Click on `File→ Preferences` and then `KNIME → R` on the left hand side of the open window, and point to your local installation of R (usually `C:\Program Files\R\R-x.x.x` where x.x.x stands for the version number of your R installation) by clicking on the browse button on the right (*Figure 2*). Congratulations, you succesfull installed all the requirements to run the KniMet workflow!

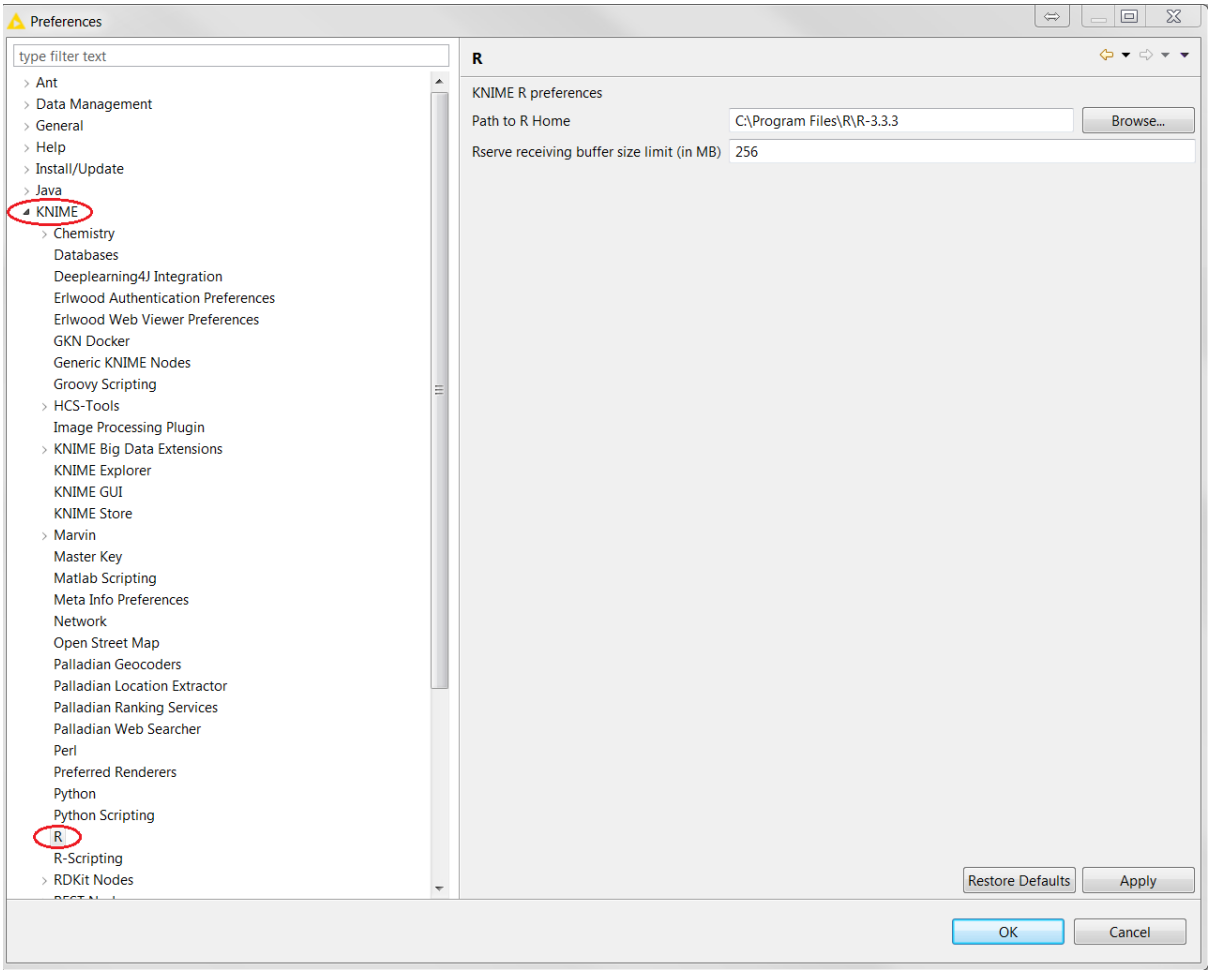

Figure 2: The KNIME Preferences window with circled in red the items to click to be able to point KNIME to the locally installed version of R.

What does it do?

1. Perform data deconvolution/Import deconvoluted data.

You can decide whether to pre-process the data directly in the KNIME platform by using the connection with the locally installed R version provided by the **R Source (Table) - Perform pre-processing with XCMS and CAMERA** node, or instead read the data matrix obtained from deconvolution performed externally with the **CSV Reader – Read Deconvolution output** node (*Figure 3*).

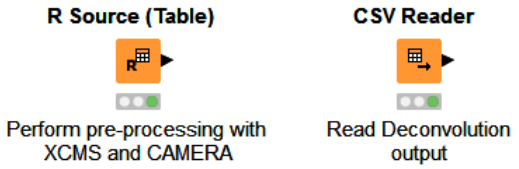

Figure 3: The R Source node on the left hand side allows data deconvolution within the KniMet platform, while the CSV Reader on the right imports a data matrix into the platform.

1.1. In case you would like to perform deconvolution using XCMS (Smith *et al.*, 2006) and CAMERA (Kuhl *et al.*, 2012) inside KniMet, the **R Source (Table) - Perform pre-processing with XCMS and CAMERA** node could be configured for this purpose. Double clicking on it, or right clicking and then clicking on `Configure…` will open the configuration window (shown in *Figure 4*), and in the central pane will be shown an R script to perform deconvolution with XCMS and peak annotation with CAMERA. In the example R script provided, the data package faahKO (Saghatelian *et al.*, 2004a) is used, and the results obtained from it can be used to test the pipeline. Otherwise, you could edit the script to perform deconvolution on your data. Once the editing is completed, the node can be run by clicking `Ctrl+Enter` from the configuration window, or by first saving the changes clicking on `OK` in the configuration window and then right clicking on the node and selecting `Execute`. Please note that on the first run of this node, as well as in several other steps of this pipeline involving R nodes, you will be prompted to a window asking for permission to install some R libraries which are missing in your local R installation. Accept, select the CRAN mirror that you would like to use and wait for the installation to finish. Once the execution of the node has terminated, the deconvoluted data matrix, the R stderr and stdout can be accessed by right clicking on the node and selecting `Data from R`, `View: R Std Output` or `View: R Error Output` from the drop-down menu that opens.

1.2. In case the deconvolution step has been performed externally, the matrix containing the analysed samples and other information (such as m/z, RT, etc.) as columns, and the detected features as rows can be imported using the **CSV Reader – Read Deconvolution output** node. Double click on the node, or right click on the node and click on `Configure…`; this will open the configuration window (*Figure 5*). From the `Settings` tab click on `Browse` and select the input data that you wish to analyse. Make sure that the right parameters are selected, such as column delimiter = \t for tab separated files, and tick `Has Column Header`. Usually MassProfiler, which is the program from the MassHunter Workstation Software suite used to deconvolute Ion Mobility – MS (IM-MS) data acquired with the Agilent 6560 Instrument, inserts in the file 4 initial lines that do not need to be imported. The number of rows to be read can be modified by moving to the `Limit Rows` tab, ticking on `Skip first lines` and selecting the number of lines to avoid (4 in this case). When the configuration process is finished, the settings can be saved by clicking on `OK` and the node can be run by right clicking on it and

selecting `Execute`, the second element from the drop down menu, or by clicking `Ctrl+Enter` from the configuration window.

Once the chosen initial node has been executed, the imported table can be visualised by right clicking on it and selecting `File Table` at the bottom of the drop down menu. We suggest to always check the output of each step and make sure that the result is what was expected, i.e. rows corresponds to features and columns to samples and their relative m/z, RT, etc. The column names of the output table need to be modified in order to have a standardised table to process, and this step can be done by connecting the output port of the last executed node to the input port of the following one described in point 2.

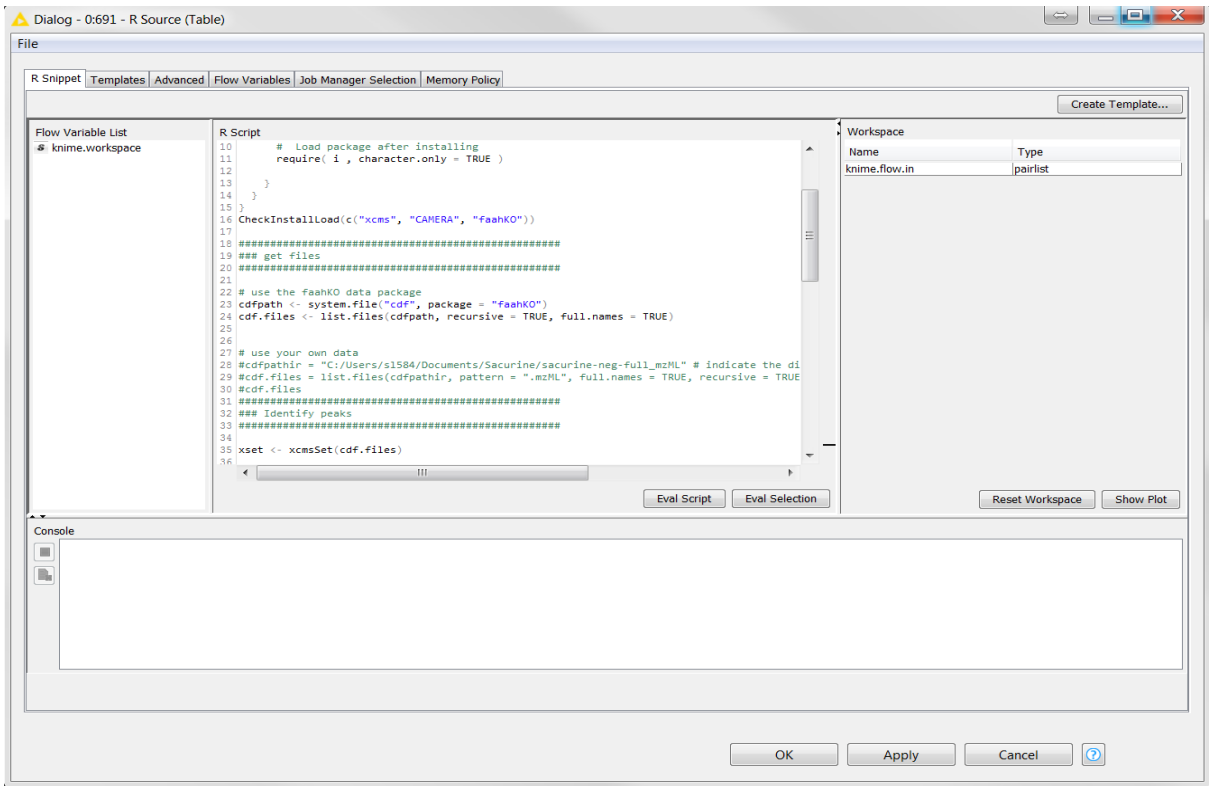

Figure 4: The R Source (Table) configuration pane

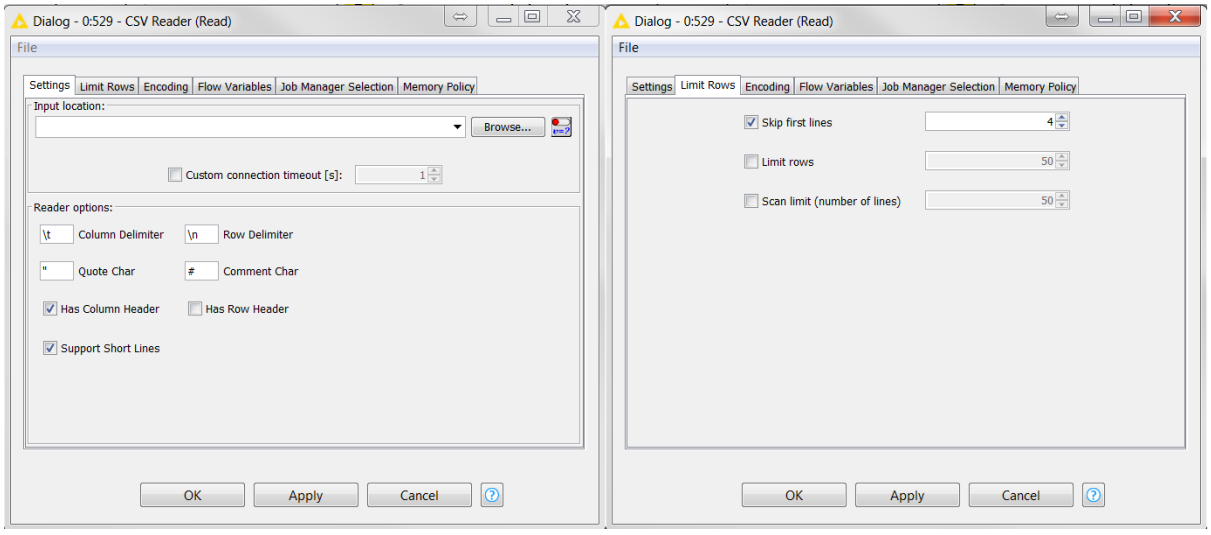

Figure 5: The Setting tab (left) and the Limit Rows tab (right) of the CSV Reader node are shown

2. Uniform Column Names.

The column names of the deconvoluted data matrix need to be standardised in order to proceed smoothly with the following steps. This step is performed with the metanode (i.e. subworkflow wrapped into a node) **Uniform column names** (*Figure 6*), which will standardise the column names of the deconvoluted data matrix, replace spaces with underscores and delete any character different from letters, numbers and underscores. The configuration panel of this metanode can be divided in 3 very similar sections: "Select column containing mass-to-charge", "Select column containing feature identifiers" and "Select column containing retention time". Each one of these sections, like the configuration window of all the other column filter nodes that will follow, is characterised by a left pane bordered in red (`Exclude`) where the columns to be discarded should be listed, and a right pane bordered in green (`Include`) where only the columns to be kept should be listed. Columns can be moved from one side to the other by double clicking on them, or selecting them and using the buttons in between the two panes `add >>`, `add all >>`, `<< remove` and `<< remove all`. You might find that in the `Include` panel there are some entries enclosed in a red square which do not correspond to any of the columns in your dataset. You can just ignore them: they derive from a previous execution, as the ticked `Enforce inclusion` option under the red panel keeps memory of the columns selected the last time that the node was run. As the titles of each section suggest, the columns containing mass-to-charge, retention time and feature identifiers need to be selected so that their names will be properly converted to `mz`, `rt` and `met_ID`. The node will give two outputs: a table where the only differences with its input will be in the name of the columns regarding mass-to-charge, retention time, feature identifiers and (if needed) samples (top output port), and a second table summarising the settings of the metanode (bottom output port).

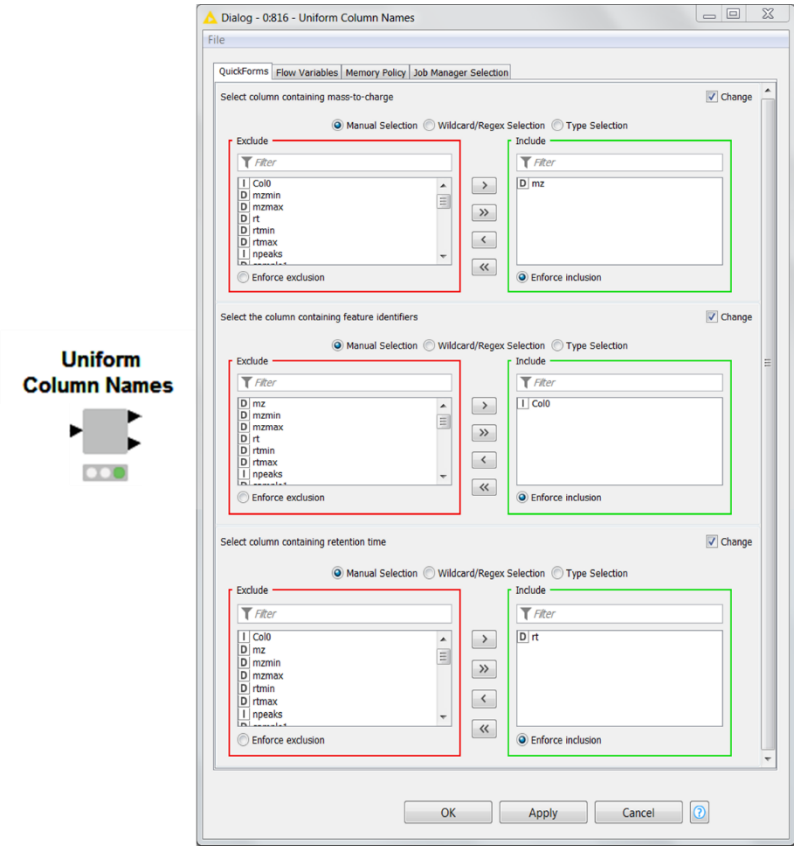

Figure 6: The Uniform Column Names metanode (left) and its configuration panel (right).

3. Keep only samples and ID.

At this point it is important to keep only the columns regarding met_ID and samples (including blanks and QCs if present), and exclude the others which can be recovered at later stages. This step is performed with the **Column Filter - Keep only samples and ID** node (*Figure 7*). Similarly to the **Uniform column names**, the configuration window is characterised by a left `Exclude` pane and a right `Include` pane. Since only the columns containing the samples and `met_ID` should be kept, they should be moved to the right pane, whereas all the other columns containing other types of information should be directed to the left hand side. Once the configuration has terminated, the node can be executed. The output of this node will be a table containing only the columns relative to the samples and `met_ID`.

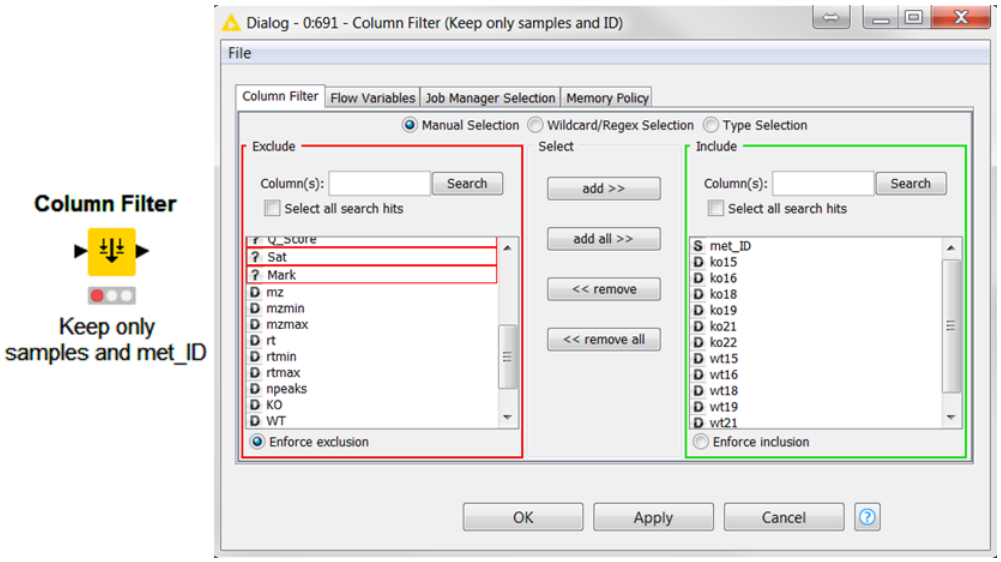

Figure 7: The column Filter node used to keep only column containing samples intensities and features identifiers (left) and its configuration panel (right).

4. Insert missing value symbol.

When a feature is not found in a given sample, XCMS inserts a zero whereas MassProfiler inserts 0.001. Either way, these are interpreted as values from both the feature filtering and the missing value imputation steps (points 0 and 6, respectively), hence they need to be modified with the **Insert missing value symbol** metanode (*Figure 8*).

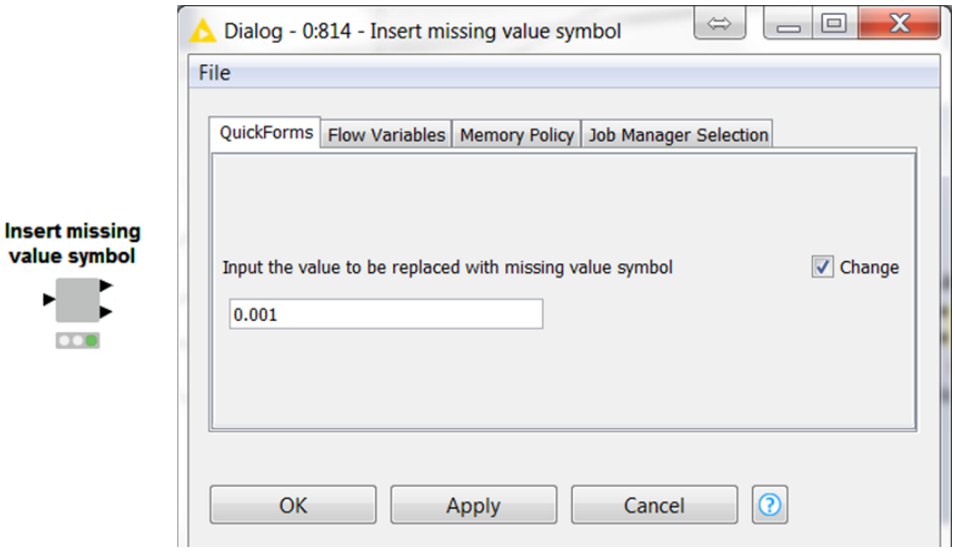

Figure 8: The metanodes to insert the correct missing value symbol in the data (left) and its configuration pane (right).

The metanode takes as input a table containing only the appropriate columns, i.e. samples and `met_ID`, and its configuration consists in inputting the value inserted by the deconvolution program for missing features. The **Insert missing value symbol** metanode gives two outputs: a table differing from the input only for the presence of missing value symbol (top output port, *Figure 9*), as well as a table containing the settings of the metanode (bottom output port).

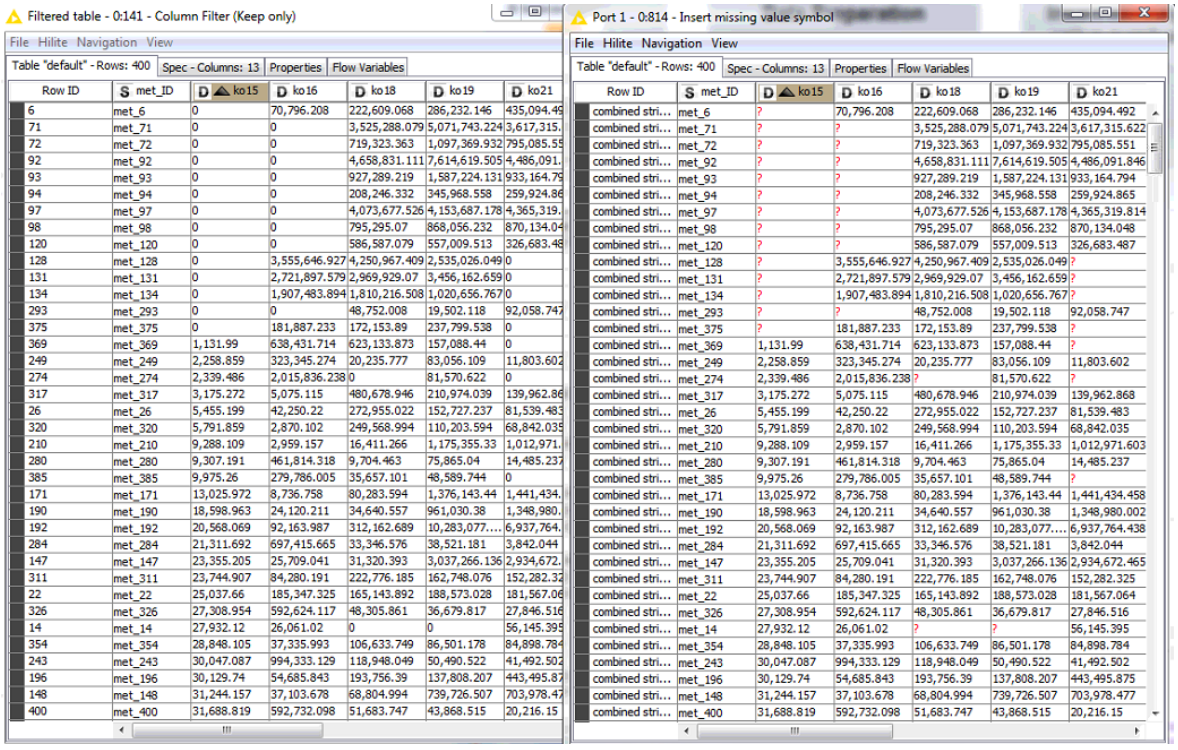

Figure 9: On the left is shown the data matrix obtained from the deconvolution of the faahKO data, containing several zeroes, while on the left table is shown the same data presenting the missing value symbol.

5. Feature Filtering

Feature filtering can be performed based on either on the blank samples, on a dilution series of pooled samples (from now on named QCs), or on identical QCs.

5.1. If identical QCs have been run along with the study samples, they can be used to assess the quality of the data by removing features with poor repeatability, i.e. features missing in a given percentage of the QCs (MV) and whose Relative Standard Deviation (RSD) across the QCs is above a given threshold. To do this, the metanode **QC-based Features Filtering** can be used (*Figure 10*). Configuration of the metanode consists of moving only the QC samples to the `Include` panel, while all the other columns should be listed under `Exclude` (similar to the **Column filter** node described in point 3). Moreover, the RSD and MV thresholds need to be inputted at the bottom part of the configuration pane (default values: RSD = 0.2, MV = 50%). The metanode takes as input a table containing `met_ID`, samples and QCs columns previously processed with the **Insert missing value symbol**. It outputs a table missing the rows corresponding to the features not repeatable across the run (top output port) as well as a table containing the setting used for the processing, the original and the final number of features (bottom output port).

5.2. Similarly, if a dilution series of QCs has been run along with the study samples, they can be used to assess the quality of the data by removing features whose intensity does not variate with variation in concentration, i.e. whose RSD is below a given threshold. Apart from the QCs at

different dilution, this method requires also at least 2 identical QCs at maximum concentration to be used to remove features missing in a given percentage of them. The **QC dilution series-based Features Filtering** metanode can be configured by selecting the diluted QCs, the RSD threshold, the identical QCs and the MV threshold (*Figure 11*).

5.3. In case QCs are not available, or you would rather perform feature filtering based on other kind of samples, the **Blank-Based Features Filtering** metanode could be used to remove background/noise. Please note that we use the term blank to refer to the injection of a sample containing all the solvents, buffers, culture media, etc. except the actual biological matrix analysed (cells, tissue, fluid, etc.). Features will be retained only if their average value in samples is higher than their average values in blanks multiplied by a factor (default 2) and if they are present in a given percentage of samples (default 80%). Configuration, input and outputs of the metanode are similar to what is described in point 5.1, with definition of both the aforementioned threshold and percentage and selection of the columns containing blanks.

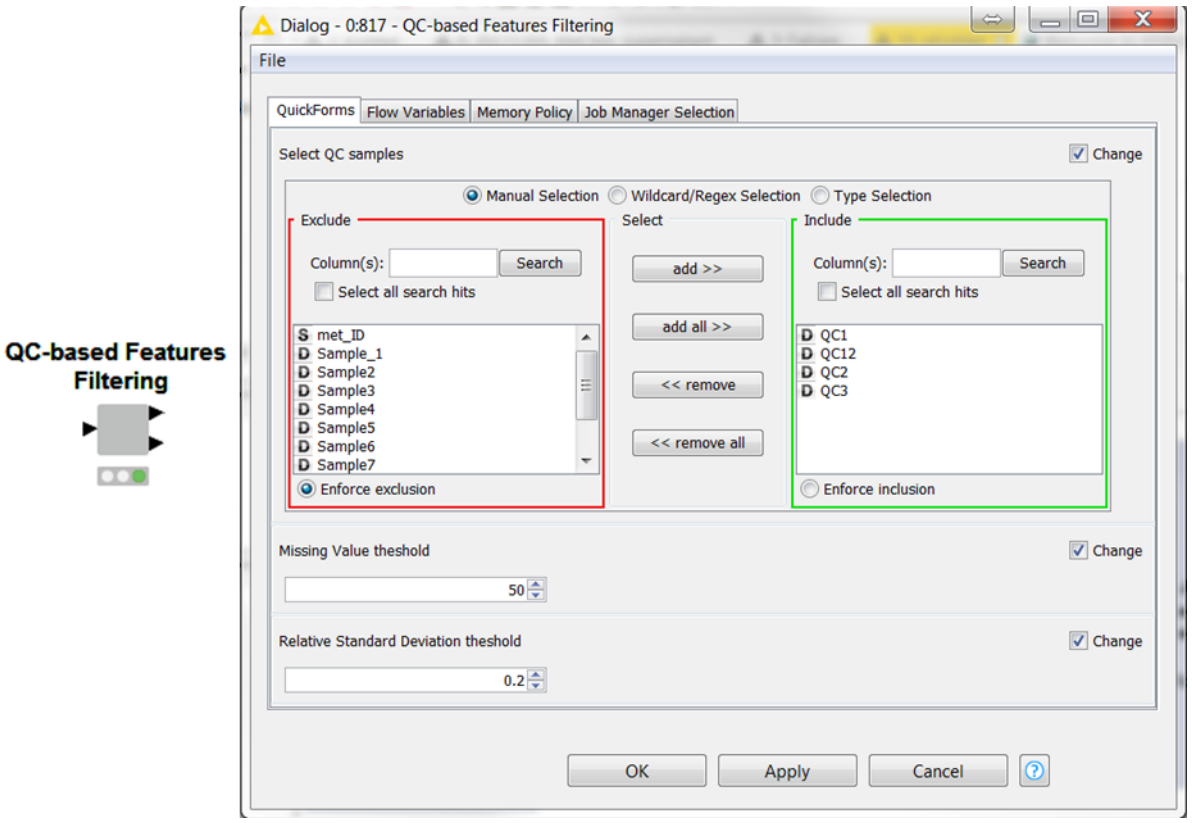

Figure 10: The QC-based Feature Filtering metanode (left) and its configuration pane (right).

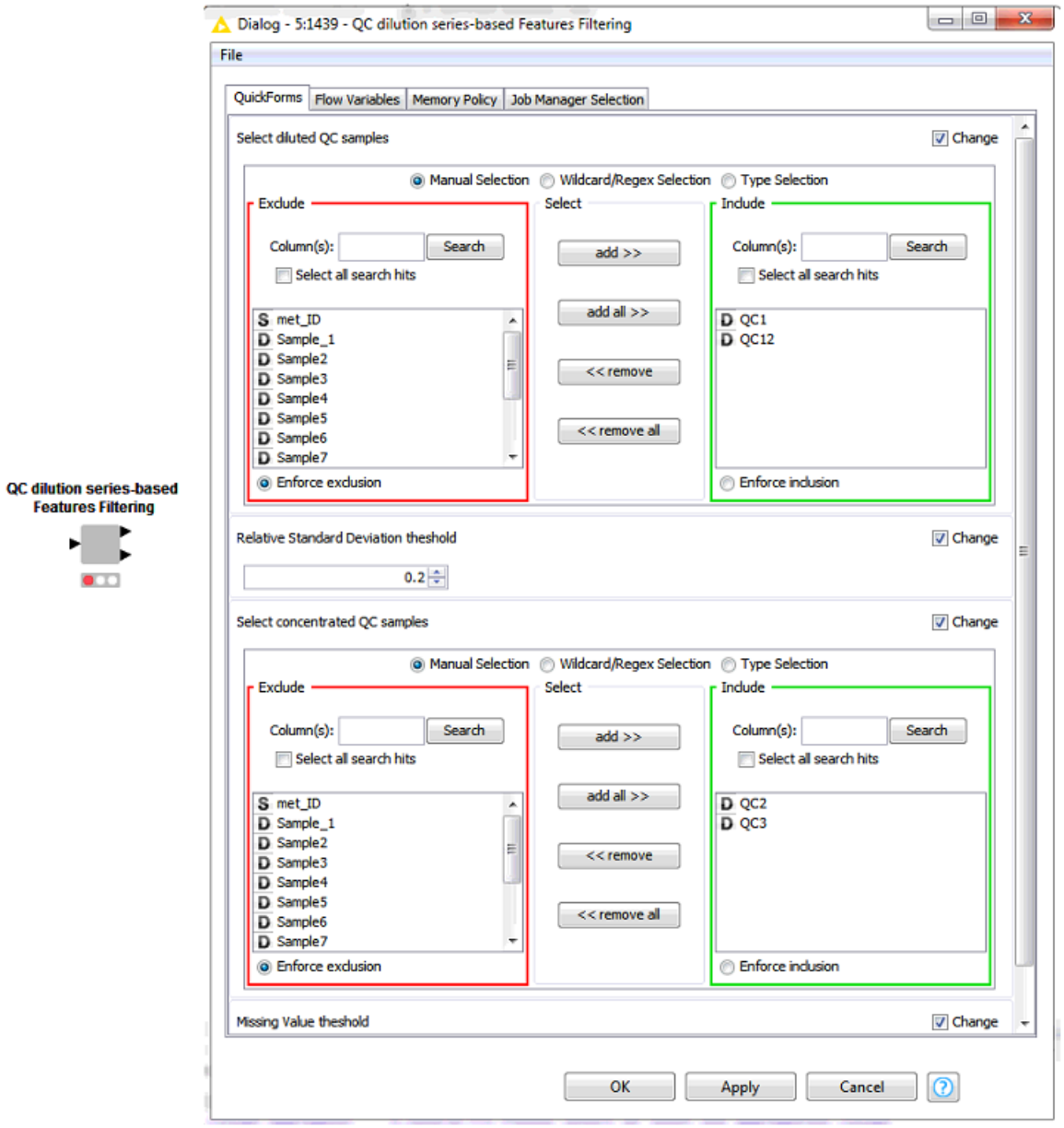

Figure 11: The Qc dilution-series based Feature Filtering metanode (left) and its configuration pane (right)

6. Missing Values Imputation

Several methods have been implemented to perform missing values imputation (MVI), including Random Forest (**MVI RF**), Key-nearest neighbour (**MVI KNN**) or Small Value (**MVI SV**) (*Figure 12*). These metanodes do not need any specific configuration. They take as input a data table containing missing values, and give as results a matrix differing from the input only in the imputed missing values (top output port, *Figure 13*). Moreover, they output a second table containing details of the processing (bottom output port). In particular, the second output of the SV imputation metanode gives only a column containing the name of the algorithm used, while

KNN and RF present three additional columns containing the R standard error, R standard output, and information regarding the R session.
On the first run of the KNN and RF metanodes you will be prompted to the installation of the R packages required by them and not already installed in your local R version.
Please note that RF can be relatively time consuming compared to KNN and SV methods. Moreover, bear in mind that the KNN method will fail if a sample has more than 80 % missing values. In this case, RF could be used instead of KNN. Alternatively, the samples containing more than 80% missing values could be deleted from further analyses by using a `Missing value column filter` before the KNN metanode (*Figure 14*), which should take as input the outcome of the feature filtering metanode and filter (include) all sample columns with a missing value threshold = 80. Once the node has been executed and the samples with high percentage of missing values deleted, its output can be connected to the `Missing Values Imputation KNN` metanode.

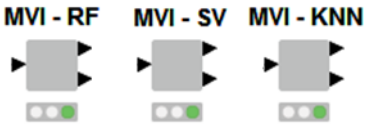

Figure 12: The metanodes to perform missing value imputation using Random Forest (left), small value replacement (centre) and Key-nearest neighbours (right).

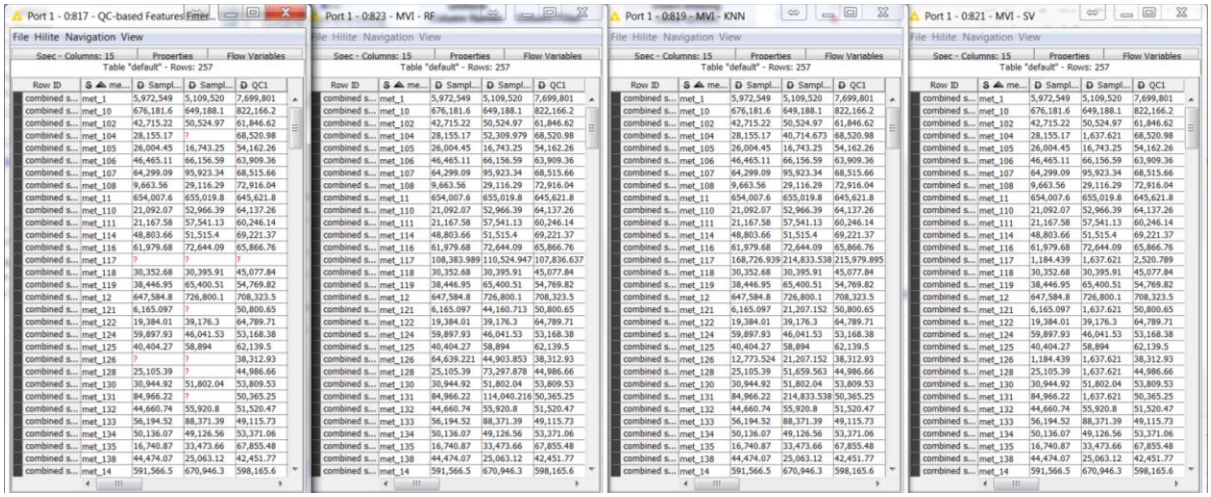

Figure 13: Comparison between the results of the QC-based Feature Filtering metanode (first table starting from left), MVI RF (second), MVI KNN (third) and MVI SV (last).

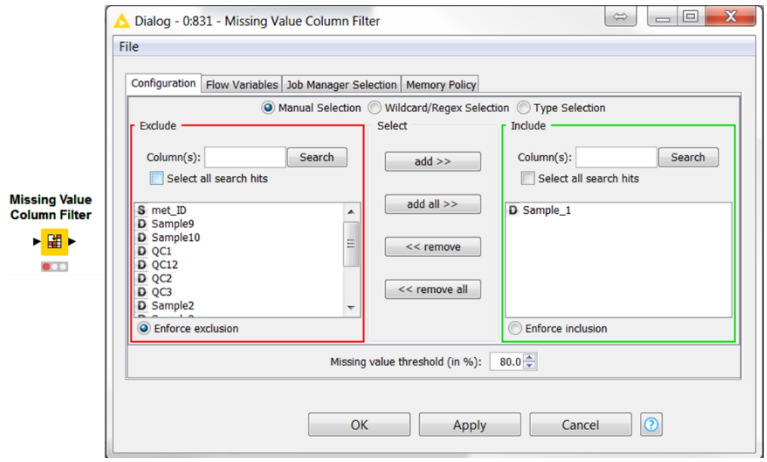

Figure 14: The Missing Value Column Filter can be added before MVI KNN in case the latter fails because one or more columns presenting 80% or more missing values.

7. Normalisation

Several normalisation methods are available, namely Probabilistic Quotients Normalisation (PQN), Loess Batch Corrections, both based on either all samples or the QCs, and Sum Normalisation.

7.1. The **PQN** metanode (*Figure 15*) is designed to perform Probabilistic Quotient Normalisation either on all samples or only on the QCs (if available). Its configuration consists in selecting the columns to be used for normalisation: only the QCs should be moved to the "Include" pane in cases of QC-based PQN, while all samples should be included if PQN has to be calculated on all samples. This metanode takes as input a table with `met_ID` and samples/QCs columns, and outputs a table containing the same rows and columns as the input, but normalised values in the cells (top input port). Moreover, the bottom input port will contain a table indicating the number of samples used to perform PQN normalisation and, in cases where only a portion of the samples was used, their names.

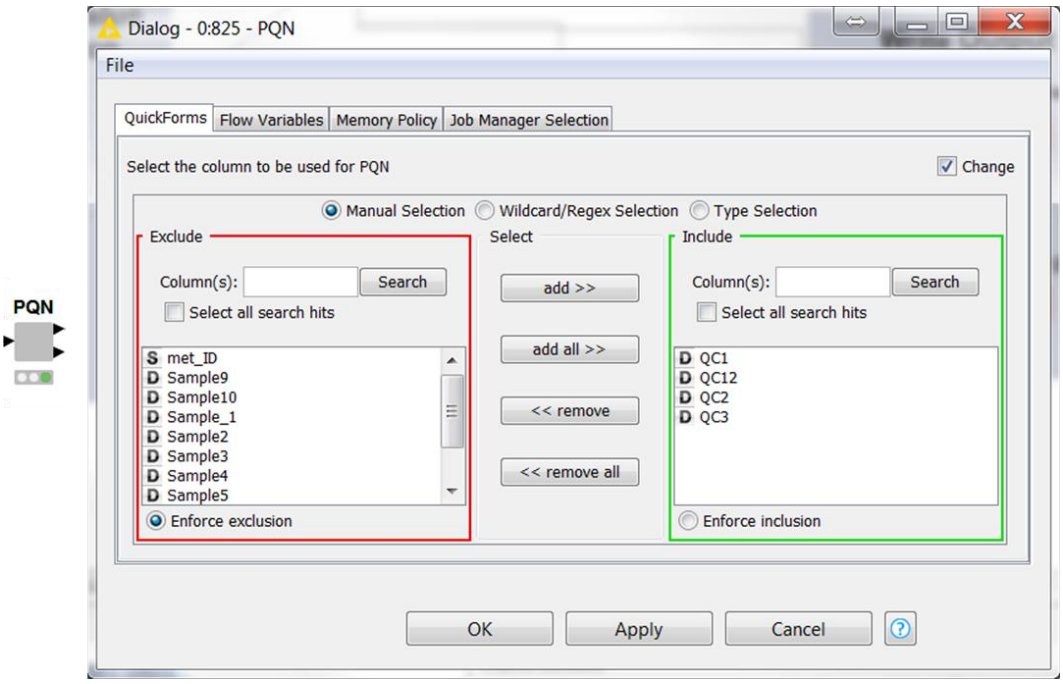

Figure 15: The PQN metanode (left) and its configuration panel (right).

7.2. The metanode **Normalisation by Sum** (*Figure 16*) normalises the data by dividing each feature in a given sample for the sum of intensities of all features in the same sample and multiplying by 100. This metanode does not require any configuration, it can be run once the output from the previously run step containing the `met_ID` and the columns for all samples is fed to its input port.

Figure 16: The Normalisation by Sum metanode

7.3. In case there is a batch-effect deriving from the acquisition of the samples in multiple analytical blocks, a more specific normalisation method should be utilised, such as **LOESS Batch Correction** (*Figure 17*). The method is based on the R script and R wrapper implemented by the Workflow4Metabolomics team to perform the robust locally estimated scatterplot smoothing

(LOESS) signal correction (Dunn *et al.*, 2011; Thévenot *et al.*, 2015; Giacomoni *et al.*, 2015). Configuration consists in selecting a file containing the samples metadata (as per example in *Table 1*), setting the desired span for LOESS calculation (between 0 and 1, default = 1; higher values correspond to higher smoothing and prevent overfitting; Cleveland *et al.*, 1992), and specifying it the correction should be performed on all samples or on the QCs.
The Loess Batch Correction metanodes provide four outputs: the corrected data matrix, a table containing details about the settings and the R session, and plots depicting both the sum of intensities for each sample and the PCA score plots of components 1-4 before and after signal correction *(Figure 18*). For more details on this specific step, please refer to the "How to" section in the Workflow4metabolomics website (http://workflow4metabolomics.org/howto).

Table 1: Example of a sample metadata table.

SampleName	injectionOrder	sampleType	Batch
QC1	1	pool	1
Sample1	2	sample	1
Sample2	3	sample	2
blank1	4	blank	2
…	…	…	…

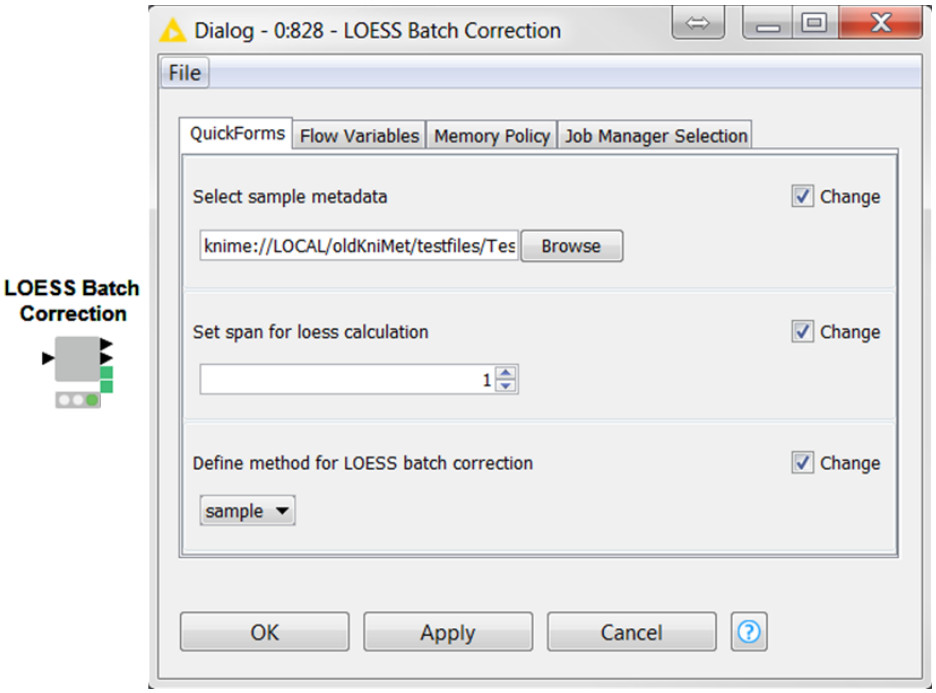

Figure 17: The configuration panel of the LOESS Batch Correction metanode

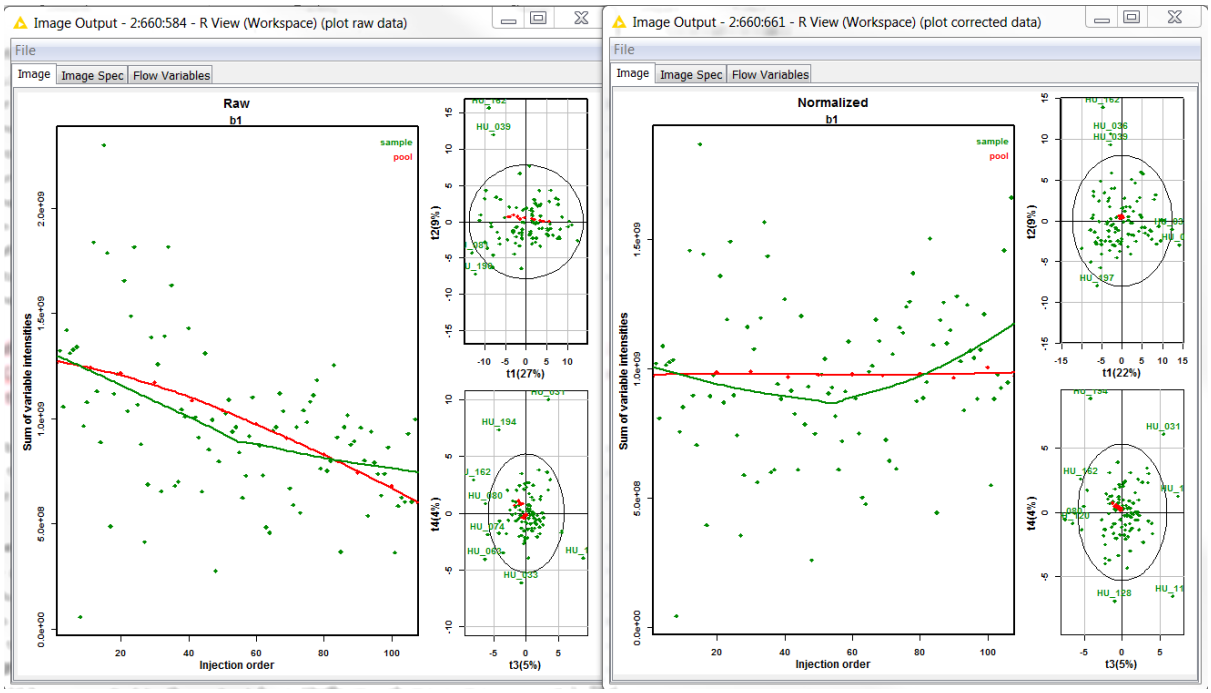

Figure 18: Plots showing the distribution of average intensities across the injection order and the PCA for the components 1-4 before (left) and after (right) QC-LOESS normalisation

8. Annotation

Annotation can be performed based solely on mass match with an external database using the **Feature Annotation** metanode, or based on mass, retention time and CCS match with a library of internally acquired compounds with the **Feature Annotation from internal IMS DB** metanode. Moreover, annotation based on the internal IMS library can be coupled with an external library thanks to the **Feature Annotation - Internal2external** metanode.
In all cases, annotation was implemented by integrating the AccurateMassSearch node of the OpenMS library (Pfeuffer *et al.*, 2017), and can be performed on either the LIPID MAPS (Fahy et al., 2007) files obtained from manipulation of the LIPID MAPS Structure Database (LMSD) with an in-house script and provided in the `LipidMaps_Annotation` folder of this repository, the HMDB (Wishart *et al.*, 2013, 2009, 2007) files provided in the share/OpenMS/CHEMISTRY folder of your OpenMS installation (in Windows usually in C:/Program Files/OpenMS – x.x.x where x.x.x stands for the version installed), or any other database files properly formatted.

8.1. The **Feature Annotation** metanode (*Figure 19*), can perform annotation based on accurate mass match with an external library. Configuration of the metanode consists in defining a number of settings:

8.1.1. **Define Mass Error Value (ppm)**: Define the mass error value in ppm to be used for feature annotation
8.1.2. **Define Ionization Mode**: Indicate polarity of the data
8.1.3. **Positive adducts** and **Negative Adducts**: Select the file containing the list of Positive adducts to consider for annotation. As the name suggests, these files contain the list of adducts to be considered for feature annotation as formula of the adduct and charge separated by a semicolon (for instance M+H;+1). An example can be found in the share/OpenMS/CHEMISTRY folder of your OpenMS installation. In case you would like to remove/add adducts from these lists, they can be modified and saved somewhere else, and the new user-defined adduct lists can be used instead.

8.1.4. **Mapping File**: Select the mapping file containing mass, formula and Database ID for each specie in the database. This is a tab separated file containing two header lines referring to database name and version along with exact mass, molecular formula and database ID(s) for the entry. This file could be **LMSDFMappingFile.tsv** provided in the `LipidMaps_Annotation` folder of this repository, **HMDBMappingFile.tsv** provided with OpenMS, or any other database in the same format as these.

8.1.5. **DB2Struct File**: Select the structure file (containing Database ID, name, SMILES and InChI for each specie in the database). This is a tab separated file containing database ID, name of the molecule, InChI and SMILES for each entry in the database. This file could be **LMSDF2StructMapping.tsv** provided in the `LipidMaps_Annotation` folder of this repository, **HMDB2StructMapping.tsv** provided with OpenMS, or a local database in the same format as these.

8.1.6. **Database portion**: Insert a string that will be used to filter the database and perform annotation only for compounds presenting this string in their identifier. For instance, if annotation needs to be performed only on the fatty acyl category of the LIPID MAPS database write "LMFA", while if annotation has to be performed only on the Eicosanoids class write "LMFA03" (please refer to the LIPID MAPS Lipid Classification System for the list of categories and classes). Leave this field blank if annotation on the whole database needs to be performed.

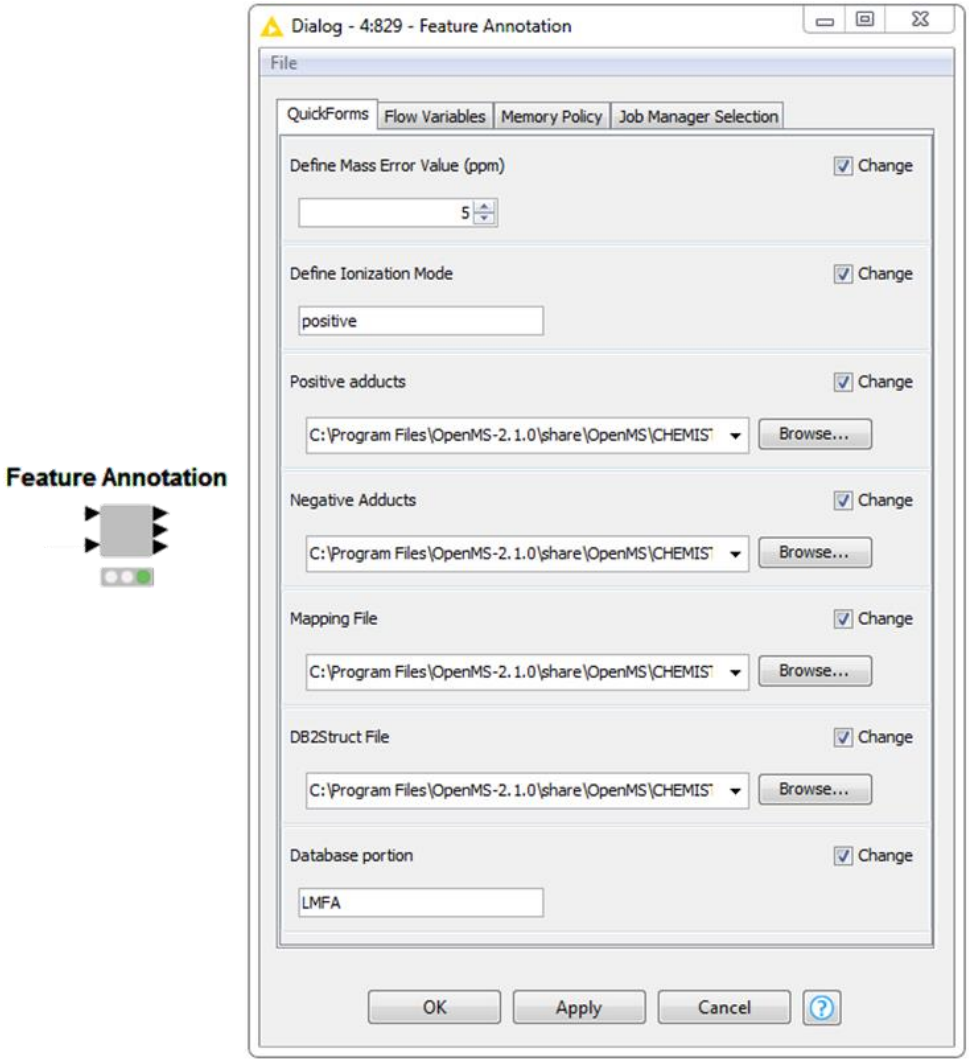

Figure 19: The Feature Annotation metanode (left) and its configuration panel (right).

The metanode takes as top input the data table to be subjected to annotation, containing only `met_ID` and sample columns, and as bottom input the table outputted by the **Uniform column names** metanode from which it will retrieve other information needed for this step. It will produce three outputs: a data table, identical to that fed to the metanode with the addition of a column containing the database identifier(s) for the annotated compounds, a table containing details about the molecules annotated (identifier, name, adduct type, Δppm, etc.; both tables shown in *Figure 20*), and a table summarising the user-defined settings.

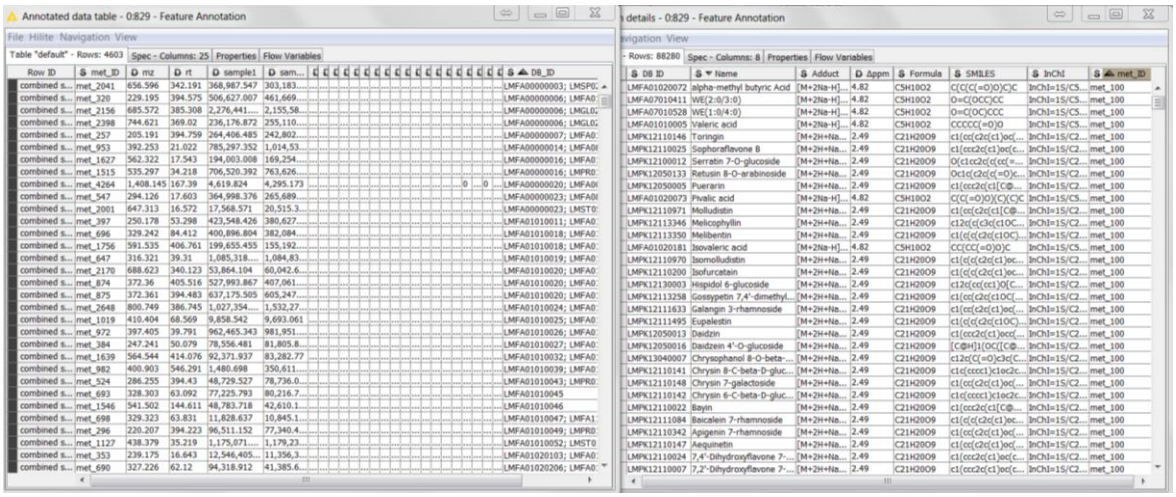

*Figure 20: The annotated data table outputted by the **Feature Annotation** metanode is shown on the left hand side, while on the right the table containing the annotation details is shown.*

8.2. The **Feature Annotation from internal IMS DB** metanode (*Figure 21*) can perform annotation based on accurate mass, retention time and CCS match with a library of compounds internally acquired. Apart from annotating compounds present in the internal library, this step also adds lipid-like flags to features presenting identical accurate mass but differing on either RT or $^{DT}CCS_{N2}$ values from the compounds in the database. More specifically, lipid-like features can be category-like (e.g. fatty acid-like) if their accurate mass and CCS values fall within the annotation thresholds but their RT does not, or *vice versa* specific lipid-like (e.g. arachidonic acid-like) for features with accurate mass and RT identical to those in the database but different CCS value (Hinz, Liggi, *et al*., data not published).
The **Eicosanoids_DTIMS_library.csv** provided with this repository was obtained from 48 lipid standards acquired individually in 3 different days. It is a tab-separated document containing the following columns: CCS mean, DT mean, RT mean, CCS RSD, Name, Ion type (adduct written in OpenMS format, for instance M+H;1+), Enzymatic Group (LOX, COX, CYP, etc.), Molecular group (FA, PC, HETE, etc.), SMILES, formula, ExactMW (theoretical mass of the molecule), InChI, and LIPID MAPS ID.
Configuration of the metanode is similar to the one described above regarding definition of mass error value and ionization mode, while has the additional settings:

8.2.1. **RT difference:** Define the retention time difference calculated between the library and the dataset at hand to be used as cutoff for feature annotation. Default value is 1.2 minutes.
8.2.2. **CCS Error**: Define the Relative Standard Deviation calculated between the CCS in the library and the CCS in the dataset at hand to be used as cutoff for feature annotation. Default value is 2%.
8.2.3. **Internal library file:** Load the file containing the internal library.
8.2.4. **RT difference – conformers:** Define the RT difference calculated between several conformers to consider them deriving from the same molecule. Default value 0.01 minutes. This step was implemented to remove dimers or other adducts that do not have a corresponding proton-monomer in the same chromatographic peak.

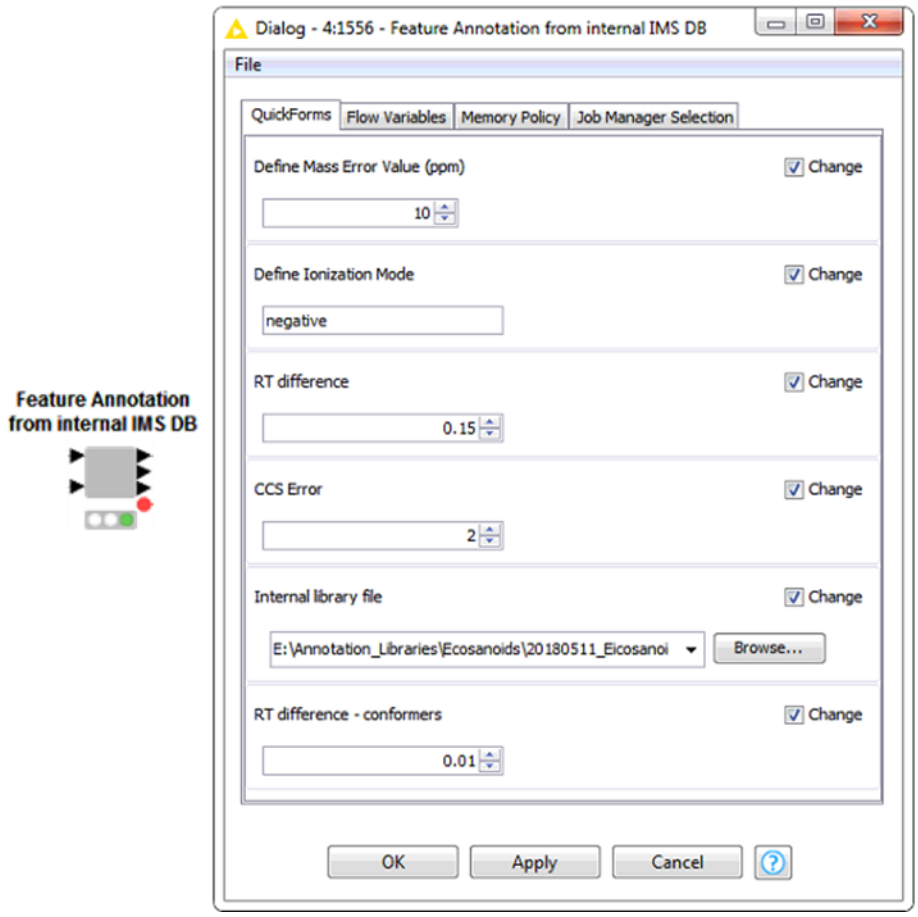

*Figure 21: The **Feature Annotation from internal IMS DB** metanode (left) and its configuration panel (right)*

The metanode takes as top input the data table to be subjected to annotation, containing only `met_ID` and sample columns, and as bottom input the table outputted by the **Uniform column names** metanode from which it will retrieve other information needed for this step, such as retention time and CCS value. It will produce four outputs: a data table, identical to that fed to the metanode with the addition of a column named "Internal DB" containing the names and adducts of the compounds from the internal library annotated with our features, a table containing details about the molecules annotated, a table summarising the user-defined settings, and a list of variables to be passed to the **Feature Annotation - Internal2external** metanode (red circle below output ports).
Please note: adduct lists do not need to be specified as the list of adducts will be taken from the internal library file.

The **Feature Annotation - Internal2external** metanode allows annotation based on an external database, such as LIPID MAPS or HMDB, following annotation with an internal library. Its configuration window is a simplified version of the **Feature Annotation** metanode (*Figure 22*), with only Mapping File, and DB2Struct File and eventually database portion to be set. All the other parameters (polarity, mass accuracy for annotation, etc.) are taken from the previously run **Feature Annotation from internal IMS DB** metanode.
The **Feature Annotation - Internal2external** metanode takes as inputs the results from the top output port as well as the variable port of the **Feature Annotation from internal IMS DB** metanode, and gives three outputs: a data table identical to that fed to the metanode with the addition of a column named "Internal DB", containing the names and adducts of the compounds from the

external database annotated with our features, a table containing details about the molecules annotated, and a table summarising the user-defined settings.

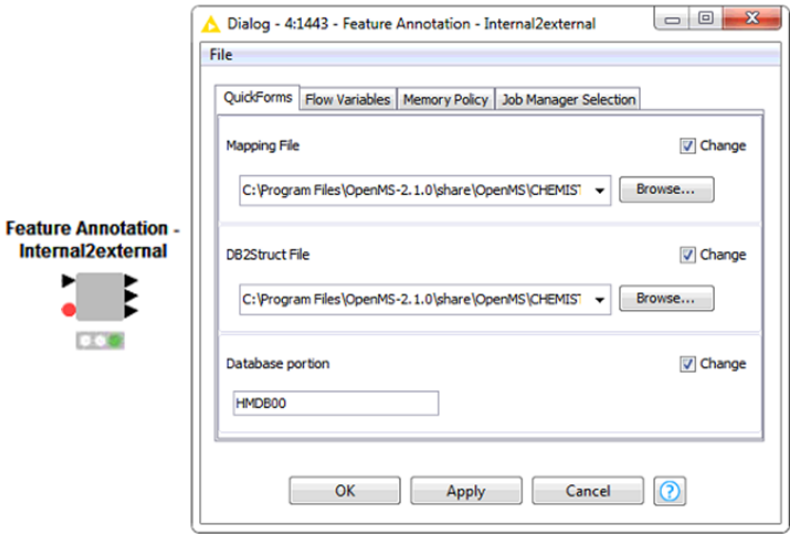

Figure 22: The Feature Annotation - Internal2external metanode (left) and its configuration panel (right)

9. Output

The processed data can be written to an output file in the desired format. Prior to output writing, you might want to include in the final output some of the columns filtered out with the **Column Filter - Keep only samples and ID**. To do this, you can connect to the top input port of the **Joiner** node the output from the **Feature annotation** metanode, while on its bottom output port you should connect the output from the **Uniform column names** metanode. Configuration of this node consists in selecting the column in common between the two tables, which can be done in the "Joiner settings" tab, as well as selecting the columns from each table that should be retained in the output from the "Column selection" tab (*Figure 23*).

Once the desired columns have been added to the processed data matrix, either the **CSV Writer** or the **Excel Writer (XLS)** node can be used to write the output in the desired format. Configuration of these nodes is very similar and requires indication of location and name of the output file to be written, plus a series of other settings such as selecting whether the file should be overridden in case of name already assigned, or if the node execution should fail. Once either one of the two writing nodes has been configured, it can be run and the output is ready for any other analysis that you wish to perform outside KNIME.

10. Report

The last output port of each metanode contains a summary of the user-defined settings and some details regarding the computations performed within the metanode. The information contained in these tables could be used to build a report containing an overview of the processing steps performed and the settings utilised. This could be done using the KNIME built-in reporting functionality as described in https://www.knime.com/reporting.

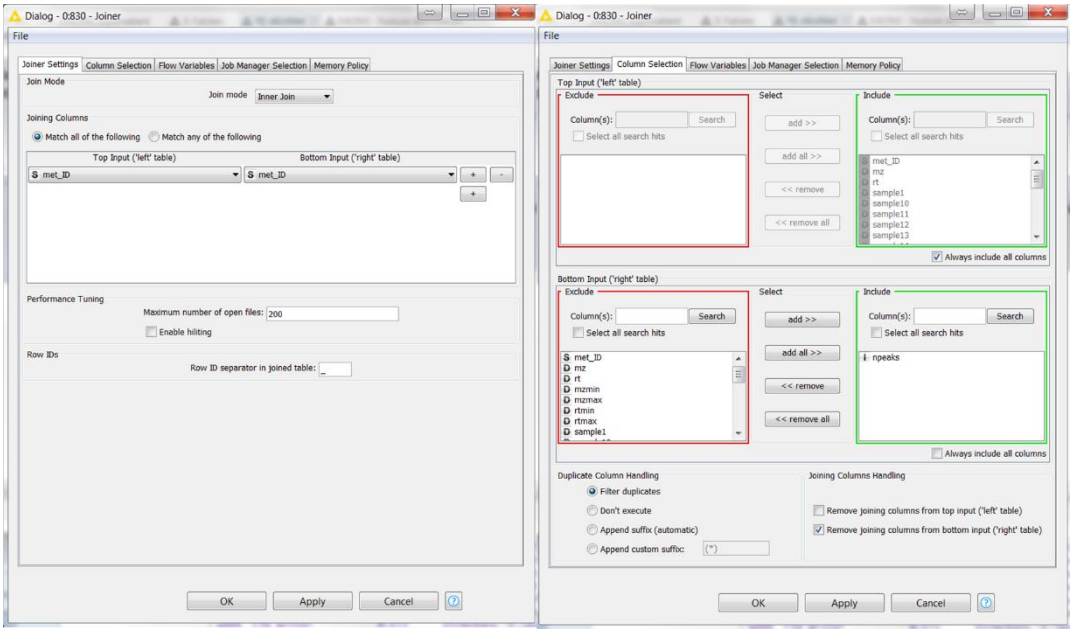

Figure 23: The Joiner Settings tab (left) and the Column Selection tab (right) of the configuration window of the Joiner node. In the shown example, only the "npeaks" column from the original data matrix (bottom input) is added to all columns from the processed data matrix.

Guided Examples

These examples are based on the tab-separated files provided into the `testfiles` folder of this repository, which can be downloaded to your local machine.

1. Perform deconvolution with XCMS in KniMet.

In this example we will use the R data package faahKO (Saghatelian *et al.*, 2004b), containing LC/MS data files from the spinal cord of 12 wild-type and knockout mice collected in positive ionization mode, to perform deconvolution within the KniMet workflow by using the R source node and proceed with data processing. This dataset contains neither the injection of QC samples nor blanks, hence feature filtering cannot be performed and data can be normalised only on all samples. The workflow, shown in *Figure 24*, comprises connection between nodes and metanodes to perform the following steps:

1.1. Data Deconvolution, performed with the **R Source (Table)** node;
1.2. **Uniform column names** metanode to eliminate special characters from names;
1.3. **Column Filter** to remove all the columns but samples and features identifiers;
1.4. **Insert missing value symbol** and **MVI KNN** to replace zeroes with missing values and impute their values;
1.5. **Normalisation by Sum**;
1.6. **Feature Annotation;**
1.7. **CSV Writer** preceded by a joiner to write to an external file the output of the processing.

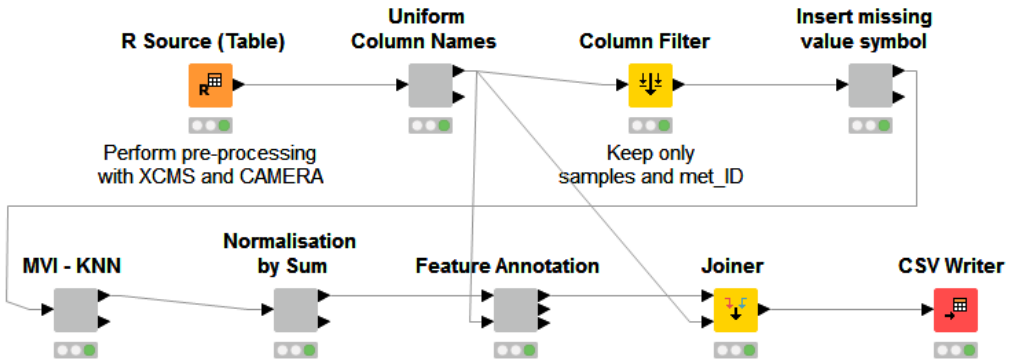

Figure 24: KniMet workflow used to process the data deconvoluted within the pipeline with the R Source node

2. Perform Processing of Drift tube Ion Mobility data.

The data acquired with the Agilent 6560 Ion Mobility Q-TOF LC-MS cannot be pre-processed with open access software, but the result from the deconvolution performed with the vendor's software can be imported into KniMet and subjected to post-processing. The file "Test_LCIMMS_Eicosanoids_Features.tsv" in the `testfiles` folder of this repository was obtained performing feature extraction with MassProfiler on blanks and platelets extracted from human plasma and run on the Agilent 6560 Ion Mobility Q-TOF LC-MS in negative ion mode. This file can be downloaded and used to follow this example, in which annotation with an internal library of eicosanoids standards is also performed (Hinz, Liggi, *et al.*, data not published). In *Figure 25* the connection between nodes and metanodes to process the matrix containing blanks are shown, namely:

2.1. Import the deconvoluted data matrix using the **CVS Reader** node – make sure to skip the first 4 lines as the actual data matrix starts from the fifth row;
2.2. **Uniform column names** metanode to eliminate special characters from names;
2.3. **Column Filter** to remove all the columns but samples and features identifiers;

2.4. **Insert missing value symbol**;
2.5. **Blank-based Feature Filtering** metanode;
2.6. **MVI KNN** to impute missing values;
2.7. **PQN** based on all samples;
2.8. **Feature Annotation from internal IMS DB** followed by **Feature Annotation - Internal2external** to perform annotation based on the internal IMS library provided with the workflow, and integrate it with the fatty acyl section of the LIPID MAPS database;
2.9. **CSV Writer** preceded by the **Joiner** to write to an external file the output of the processing.

Likewise, a matrix containing QCs and not containing blanks can be processed with a very similar workflow, which differs from the one described above only in the feature filtering and in the normalisation method, both of which are now QCs.

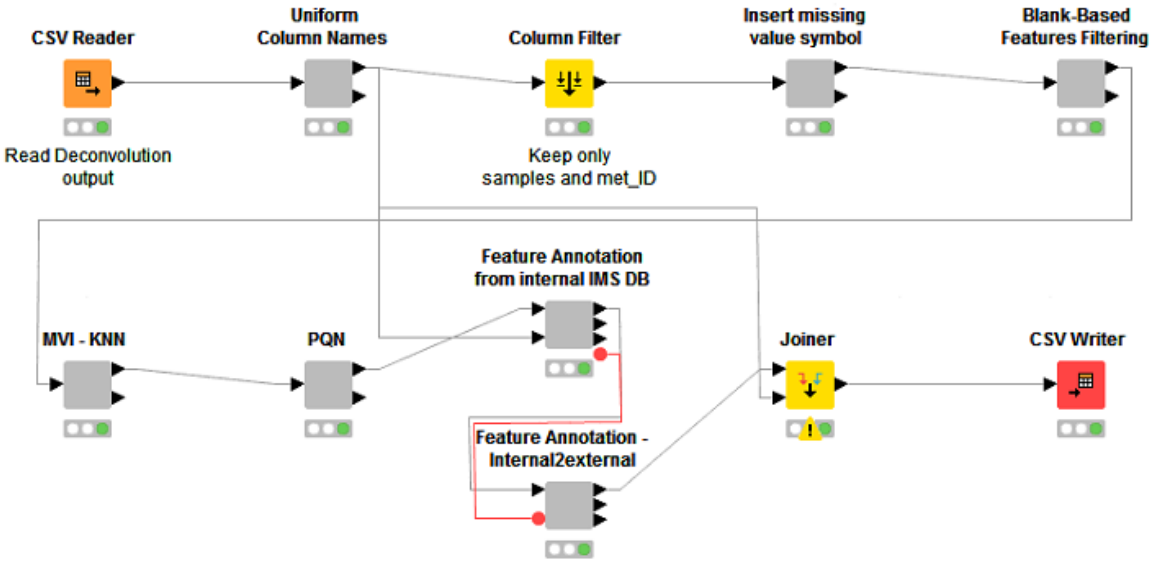

Figure 25: The complete workflow for the analysis of LC-IM-MS data containing pooled samples and pre-processed outside the KniMet workflow

3. Perform Batch Correction

The `testfiles` folder contains the tab-separated file Test_Batchfeatures.tsv, which was obtained by processing human plasma acquired on a LC-MS instrument on positive ionisation mode in two different analytical blocks. In order to correct for batch effect, information regarding order of injection and batch of the samples is needed and can be found in the "Test_Batch_samplemetadata.tsv" file in the `testfiles` folder. Please note that this dataset does not contain QCs injection, hence the Loess method based on all samples will be used. The workflow to process this dataset is shown in *Figure 26* and contains:

3.1. **CVS Reader** node to import the deconvoluted data matrix;
3.2. **Uniform column names** metanode to eliminate special characters from names;
3.3. **Column Filter** node to remove all the columns but samples and features identifiers;
3.4. **Loess Batch Correction** metanode to perform batch correction on all samples;
3.5. **Insert missing value symbol** followed by **MVI KNN** to impute missing values;
3.6. **Feature Annotation**;

3.7. **CSV Writer** preceded by the **Joiner** to write to an external file the output of the processing.

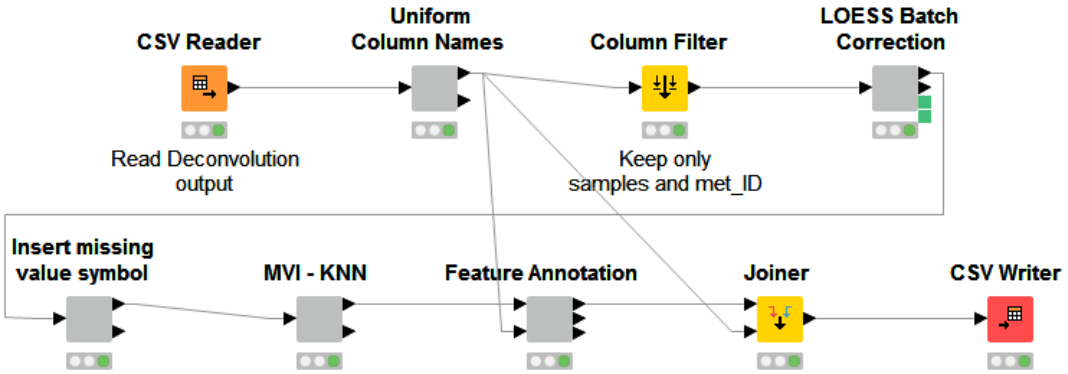

Figure 26: The KniMet workflow adapted to perform Loess batch correction based on all samples

The effective results of batch correction can be evaluated by comparing variable intensities as well as the separation of samples in PCA space before and after normalisation by visualising the two plots outputted by the **All-Loess Batch Correction** metanode (*Figure 27*).

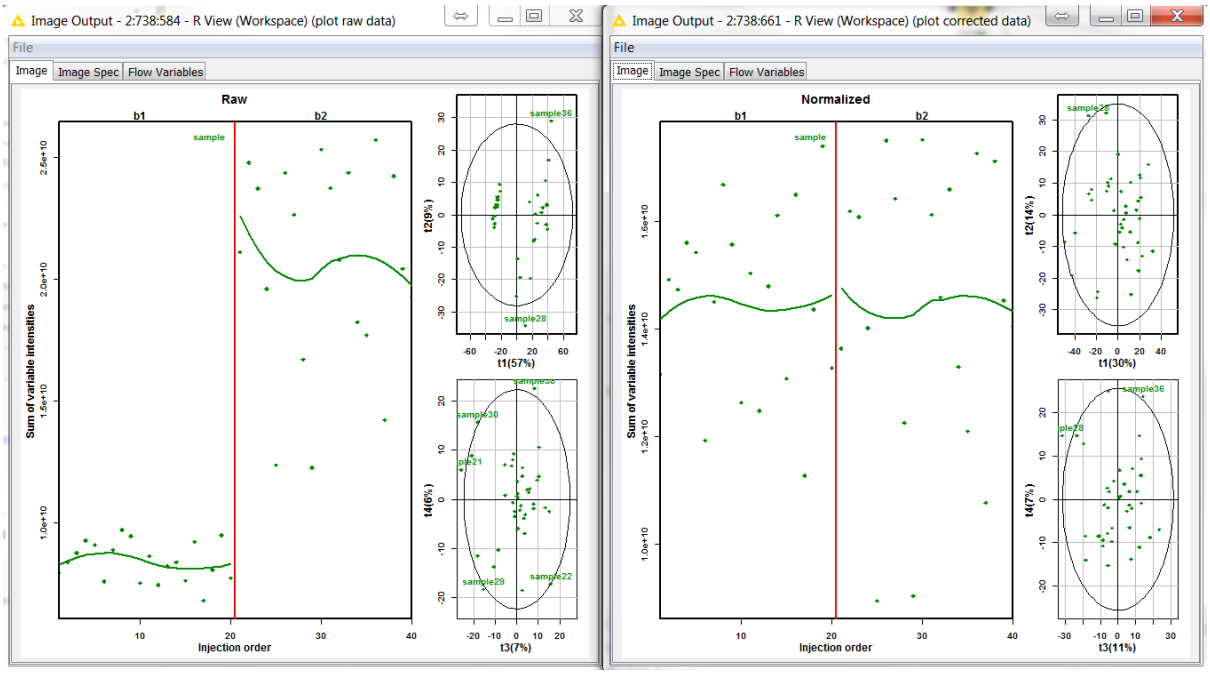

Figure 27: The image outputs of the Loess Batch Correction performed on all samples for raw (left) and normalised (right) data present on the left hand side the sum of intensity for all variables in each given sample as a function of the order of injection, along with the fitted loess curve. On the right hand side of each output image there are two PCA plots for the first and second components (top) and for the third and fourth (bottom).

Bibliography

Berthold,M.R. *et al.* (2007) KNIME: The Konstanz information miner. In, *Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007)*. Springer, pp. 319–326.

Cleveland,S.W. *et al.* (1992) Local regression models. In, Chambers,J.M. and Hastie,T.J. (eds), *Statistical Models in S*. Wadsworth & Brooks/Cole, Pacific Grove, CA, pp. 309–376.

Dunn,W.B. *et al.* (2011) Integration of metabolomics in heart disease and diabetes research: current achievements and future outlook. *Bioanalysis*, **3**, 2205–2222.

Fahy,E. *et al.* (2007) LIPID MAPS online tools for lipid research. *Nucleic Acids Res.*, **35**, W606–W612.

Giacomoni,F. *et al.* (2015) Workflow4Metabolomics: A collaborative research infrastructure for computational metabolomics. *Bioinformatics*, **31**, 1493–1495.

Kuhl,C. *et al.* (2012) CAMERA: An integrated strategy for compound spectra extraction and annotation of liquid chromatography/mass spectrometry data sets. *Anal. Chem.*, **84**, 283–289.

Pfeuffer,J. *et al.* (2017) OpenMS - A platform for reproducible analysis of mass spectrometry data. *J. Biotechnol.*, **261**, 142–148.

R Core Team (2014) R: A Language and Environment for Statistical Computing R Foundation for Statistical Computing, Vienna, Austria.

Saghatelian,A. *et al.* (2004a) Assignment of Endogenous Substrates to Enzymes by Global Metabolite Profiling †. *Biochemistry*, **43**, 14332–14339.

Saghatelian,A. *et al.* (2004b) Assignment of Endogenous Substrates to Enzymes by Global Metabolite Profiling †. *Biochemistry*, **43**, 14332–14339.

Smith,C.A. *et al.* (2006) XCMS: Processing Mass Spectrometry Data for Metabolite Profiling Using Nonlinear Peak Alignment, Matching, and Identification. *Anal. Chem.*, **78**, 779–787.

Thévenot,E.A. *et al.* (2015) Analysis of the Human Adult Urinary Metabolome Variations with Age, Body Mass Index, and Gender by Implementing a Comprehensive Workflow for Univariate and OPLS Statistical Analyses. *J. Proteome Res.*, **14**, 3322–3335.

Urbanek,S. (2013) Rserve: Binary R server.

Wishart,D.S. *et al.* (2009) HMDB: A knowledgebase for the human metabolome. *Nucleic Acids Res.*, **37**, 603–610.

Wishart,D.S. *et al.* (2007) HMDB: The human metabolome database. *Nucleic Acids Res.*, **35**, 521–526.

Wishart,D.S. *et al.* (2013) HMDB 3.0--The Human Metabolome Database in 2013. *Nucleic Acids Res.*, **41**, D801–D807.