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

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')
```

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/

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.

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 will be prompted to an error window asking for installation of the missing and required extensions: click on `Yes` and then in the `Install` window

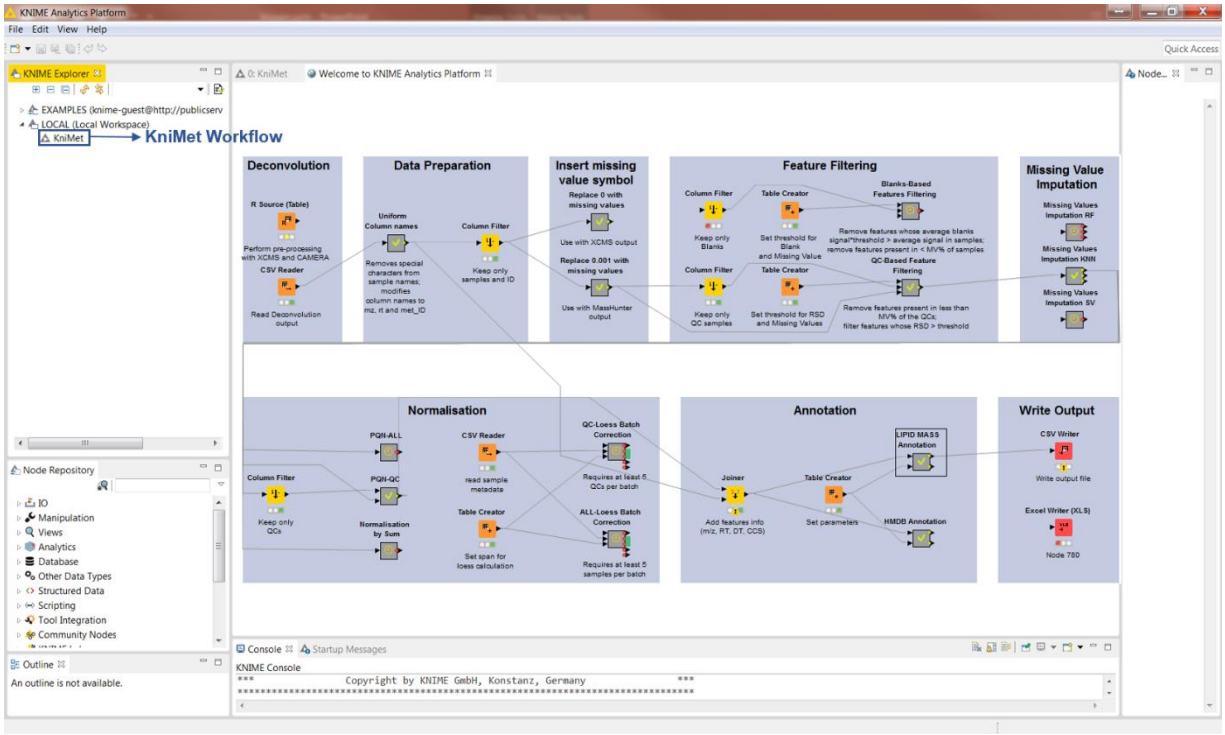

Figure 1: The KniMet workflow is highlighted on the KNIME Explorer pane, while on the central pane the KniMet pipeline is shown

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`. It derives from previous configurations; it can be ignored by clicking on `OK`.

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!

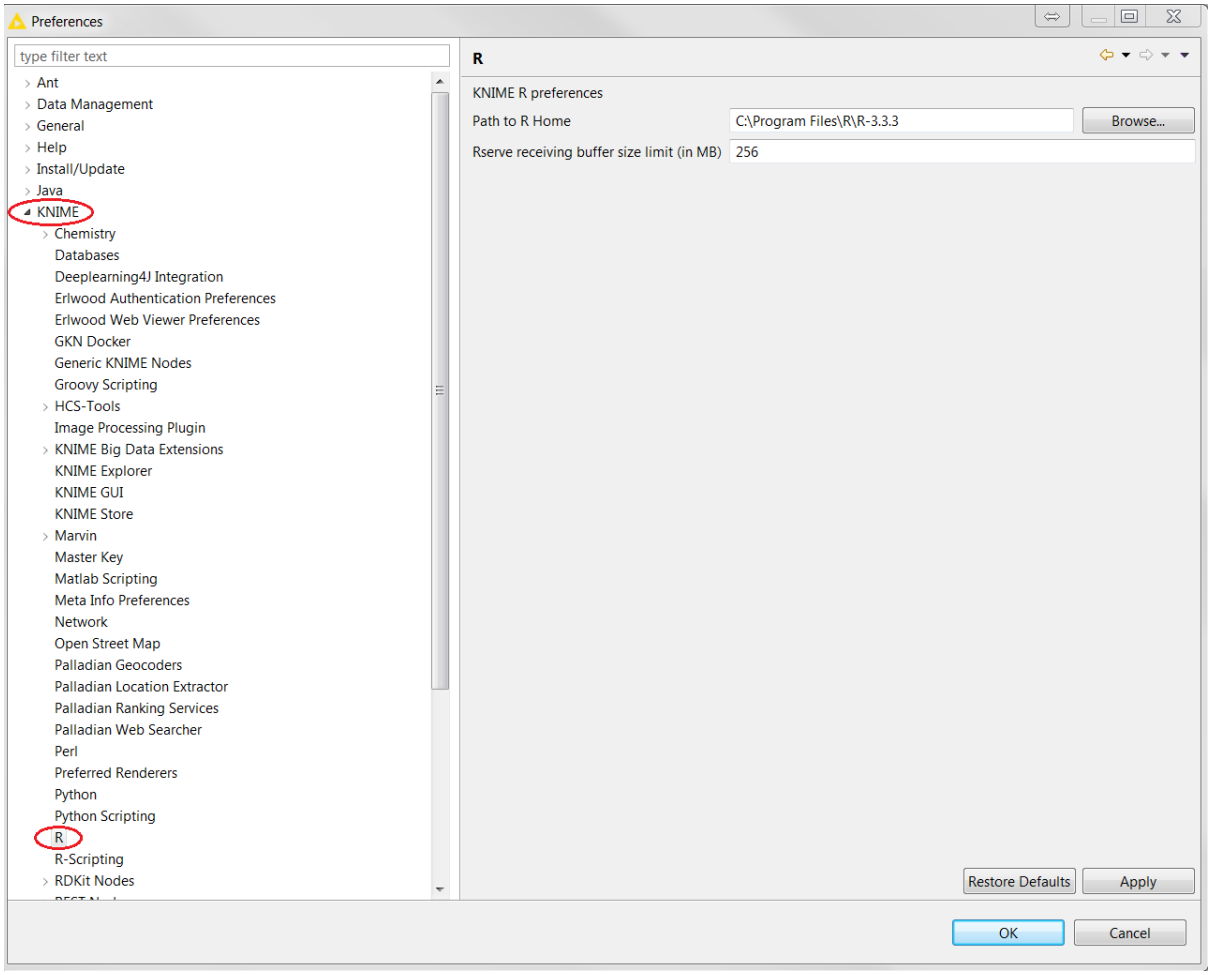

Figure 2: The KNIME Preferences window with circled in red the items to click to be able to point KNIME to the locally installed version of R.

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*).

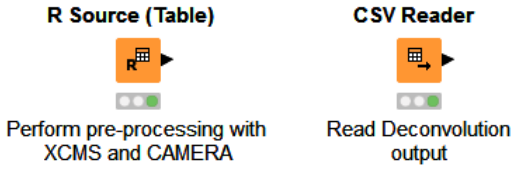

Figure 3: The R Source node on the left hand side allows data deconvolution within the KniMet platform, while the CSV Reader on the right imports a data matrix into the platform.

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 the scope. Double clicking on it, or right clicking and then clicking on `Configure…` will open the configuration window (shown in *Figure 4*), where 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.*, 2004) 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.

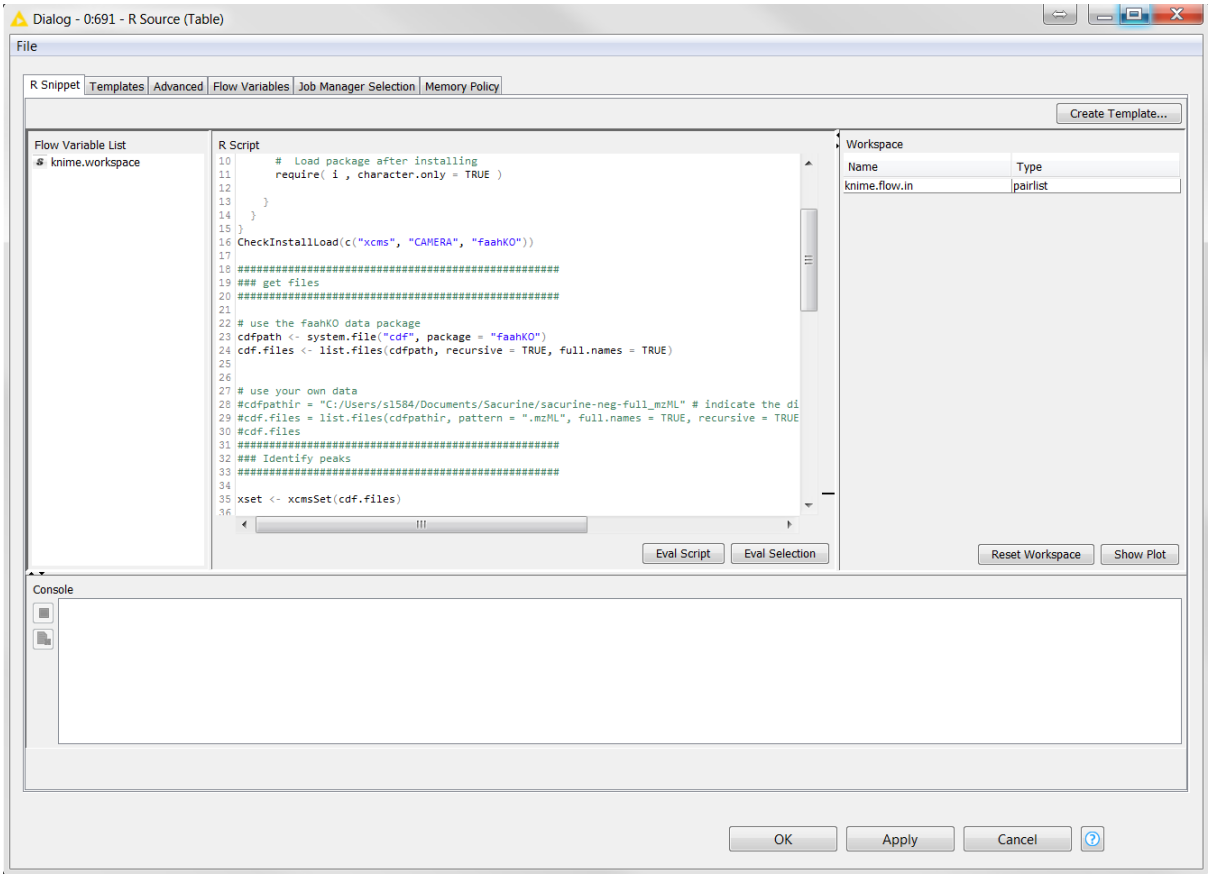

Figure 4: The R Source (Table) configuration pane

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 voice from the drop down menu, or by clicking `Ctrl+Enter` from the configuration window.

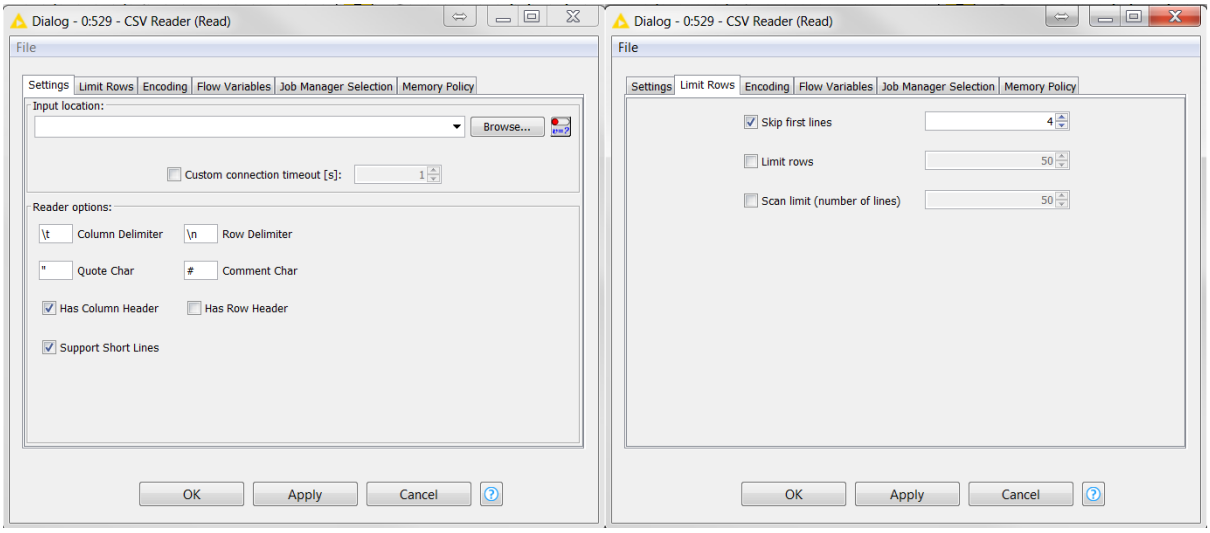

Figure 5: The Setting tab (left) and the Limit Rows tab (right) of the CSV Reader node are shown

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.

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 (more on the definition of metanode later) **Uniform column names - remove spaces and special characters from sample names, add met_ID feature identifier** (*Figure 6*), which will modify sample names by replacing spaces with underscores and deleting any character different from letters, numbers and underscores. Moreover, it will uniform the naming of the columns containing mass-to-charge, retention time and feature identifiers by converting any column named m/z or mzmed to `mz`, RT or rtmed to `rt` and ID and featureidx `met_ID`. Hence, the output will be 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 (possibly) samples.

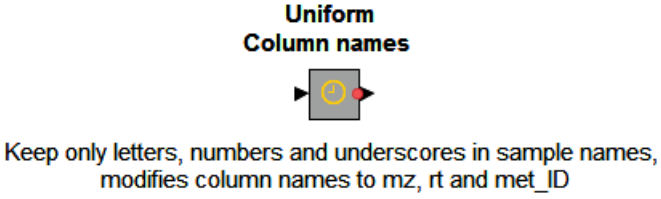

Figure 6: The Uniform Column Names metanode

As commented above, this is not a node but a metanode, i.e. a node containing a sub-workflow inside. Although the sub-workflows can be visualised and modified by double-clicking on the metanodes, this as well as all the following metanodes in KniMet were designed so that they do not need any kind of configuration, and can be run simply by right-clicking on them and clicking on `Execute`.

3. Keep only samples and ID.

At this point it is important to keep only the columns regarding met_ID and samples, 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*).

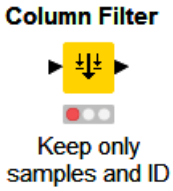

Figure 7: The column Filter node used to keep only column containing samples intensities and features identifiers

Like all the other column filter nodes that will follow, the configuration window 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 (*Figure 8*). Columns can be moved from one side to the other by selecting them and using the buttons in between the two panes `add >>`, `add all >>`, `<< remove` and `<< remove all`. 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. You might find that in the `Exclude` panel there are some entries enclosed in a red square which do not correspond to any of the columns in your dataset. You do not need to worry and can just ignore them: they are derived from the previous selection, as the ticked `Enforce exclusion` option under the red panel keeps memory of the columns excluded the last time that the node was run. 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`.

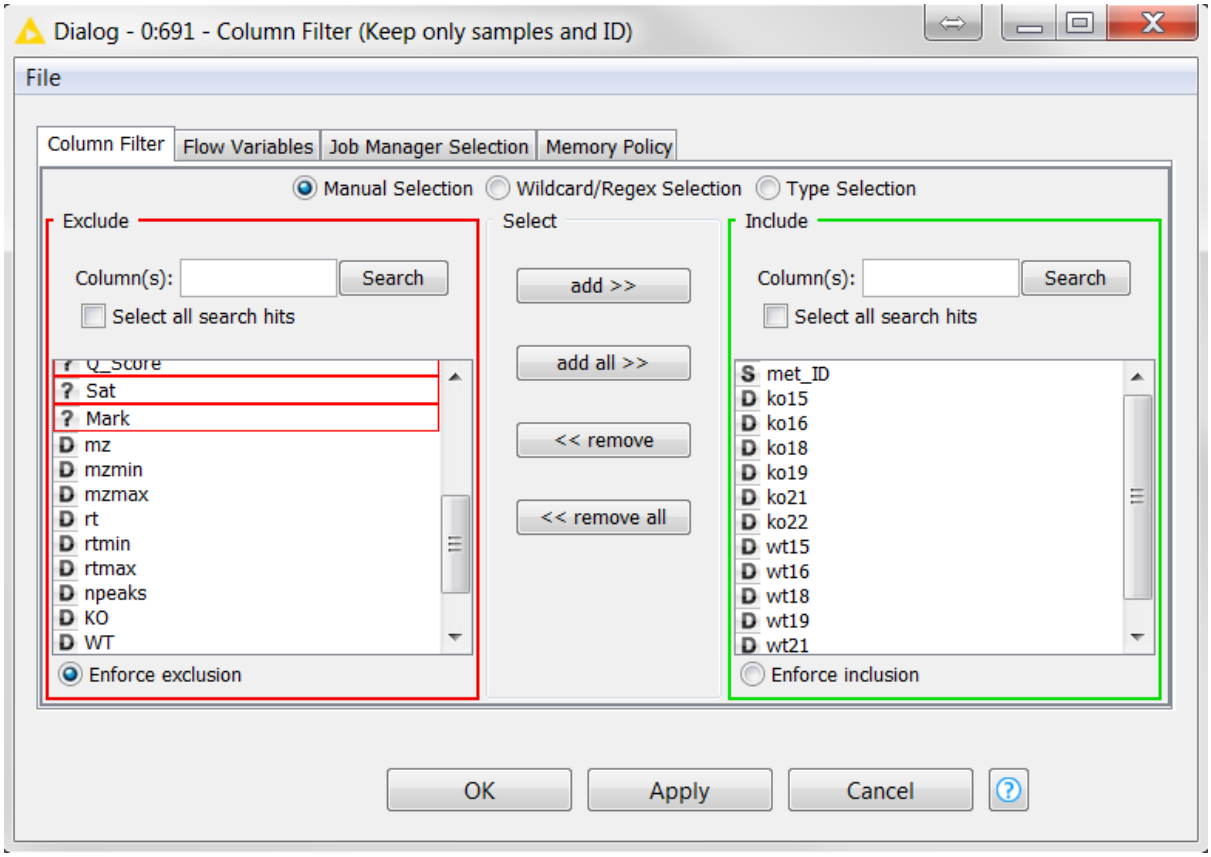

Figure 8: The configuration window of the Column filter node

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 5 and 6, respectively), hence they need to be replaced with the missing value symbol. Two metanodes are available depending on the source of the input file, **Replace 0.001 with missing values - Use with MassHunter output** and **Replace 0 with missing values - Use with XCMS output** (Figure 9).

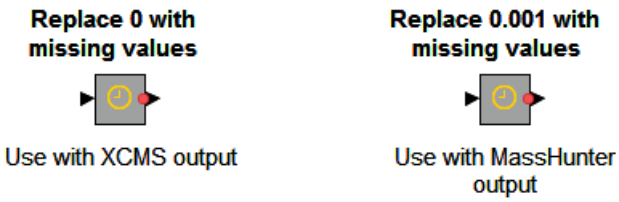

Figure 9: The metanodes to insert the correct missing value symbol in the data matrix obtained from MassHunter (right) and XCMS (left) are shown.

Once a table containing only the appropriate columns, i.e. samples and `met_ID`, is connected to the metanode, it can be run with no configuration needed. The output will be a table differing from the input only on the zeroes (or 0.001) which are replaced by the missing value symbol (*Figure 10*).

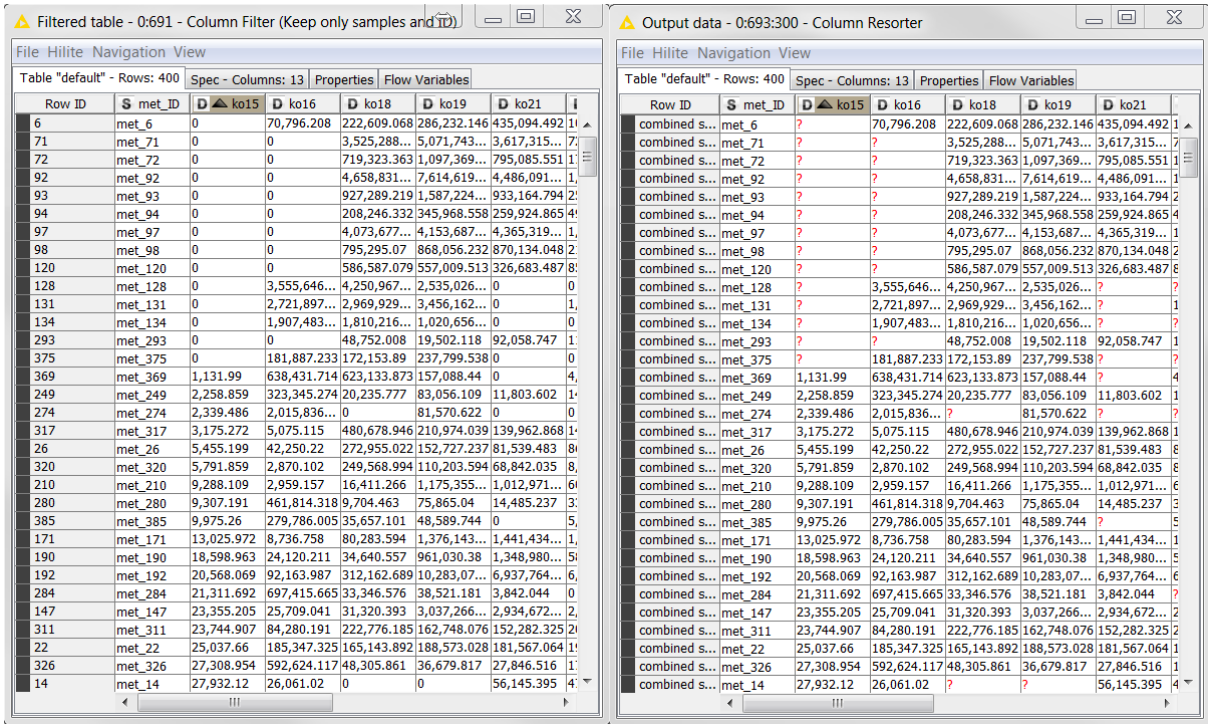

Figure 10: On the left is shown the data matrix obtained from the deconvolution of the faahKO data, containing several zeroes, while on the left table is shown the same data presenting the missing value symbol

5. Feature Filtering

Feature filtering can be performed based on either the QCs or the blank samples.

5.1. If pooled samples (from now on named 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, first we need to select the columns containing the QC samples with the **Column Filter - Keep only QC samples** node (Figure 11, bottom left). Configuration of the node consists in moving only the QC samples to the `Include` panel, while all the other columns should be listed under `Exclude` (please refer to point 3 for details on the configuration). Once configuration has finished, the node can be run and it will yield a table containing only the columns relative to the QCs. A second preliminary step consists in setting the threshold and a missing value percentage values, and this can be done with the **Table Creator - Set threshold for RSD and Missing Values** node (Figure 11, top left). Default values are set to RSD = 0.2 and MV = 50%, but they can be modified by double clicking on the relative cell of the table and editing the values.
The output of both nodes can be fed to the input ports of the **QC-based Features Filtering - Remove features present in less than MV% of the QCs; filter features whose RSD > threshold** metanode (Figure 11, right), which will delete all the features not consistently detectable across the QC runs. Make sure that the top input of the metanode is connected to the **Table Creator**, the central input port to the column filter, and the bottom input port should be connected to the output from the node inserting missing values (step 4). Once the right inputs are fed to the metanode, executing it will yield a table containing `met_ID` and all samples columns, but missing the rows corresponding to the features not repeatable across the run.

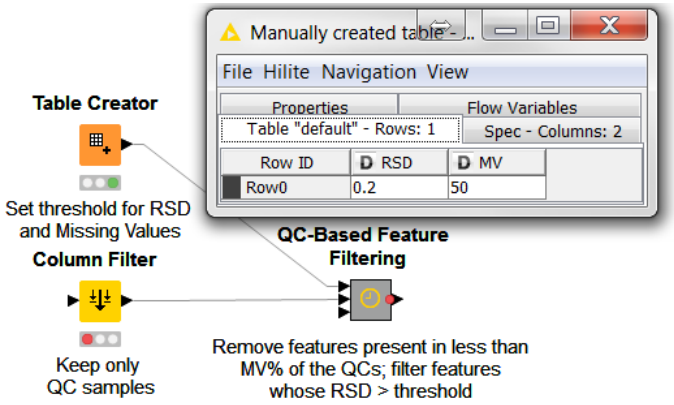

Figure 11: The Table creator node to set the relative standard deviation and missing value threshold along with its output are shown on top, while the column filter to select only the pooled samples and the metanode to perform features filtering are in the bottom part of the figure.

5.2. In case QCs are not available, or you would rather perform feature filtering based on other kinds 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 the samples is higher than their average values in the blanks multiplied by a factor (default 2) and if they are present in a given percentage of samples (default 80%). Similarly to what described in point 5.1, preliminary steps consist in defining both the aforementioned threshold and percentage with the **Table Creator - Set threshold for Blank and Missing Value**, while blanks samples need to be selected with the **Column Filter - Keep only Blanks** node (*Figure 12*).
Make sure that the top input of the metanode is connected to the **Table Creator** output, the central input port to the **Column filter**, and the bottom input port should be connected to the output from the node inserting missing values (step 4). Once the right inputs are fed to the metanode, executing it will yield a table containing `met_ID` and all samples columns but blanks, and with only the rows corresponding to the features respecting the intensity and missing values criteria defined above.

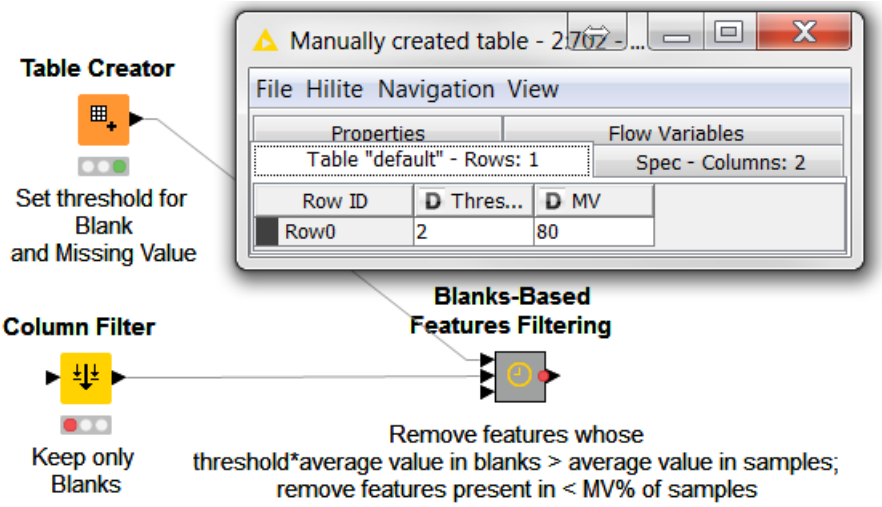

Figure 12: The Table creator node to set the threshold and missing value percentage along with its output are shown on top, while the column filter to select only the pooled samples and the metanode to perform features filtering are in the bottom part of the figure.

6. Missing Values Imputation

Several methods have been implemented to impute missing values, including Random Forest (**Missing Values Imputation RF**), Key-nearest neighbour (**Missing Values Imputation KNN**) or Small Value (**Missing Values Imputation SV**) (Figure 13).

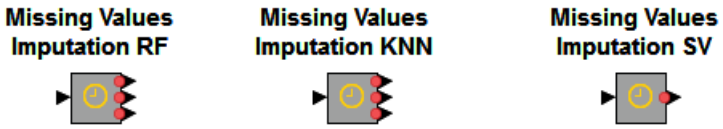

Figure 13: The metanodes to perform missing value imputation using Random Forest (left), Key-nearest neighbours (centre) and small value replacement (right).

These metanodes take as input the result of either **Replace 0/0.001 with missing values** or **QC-based Features Filtering**, and give as result the data matrix differing from the input only in the imputed missing values (*Figure 14*). In particular, the SV imputation metanode gives only this output, while KNN and RF present three output ports corresponding to the data matrix (top), R stderr (centre) and R stdout (bottom). 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.

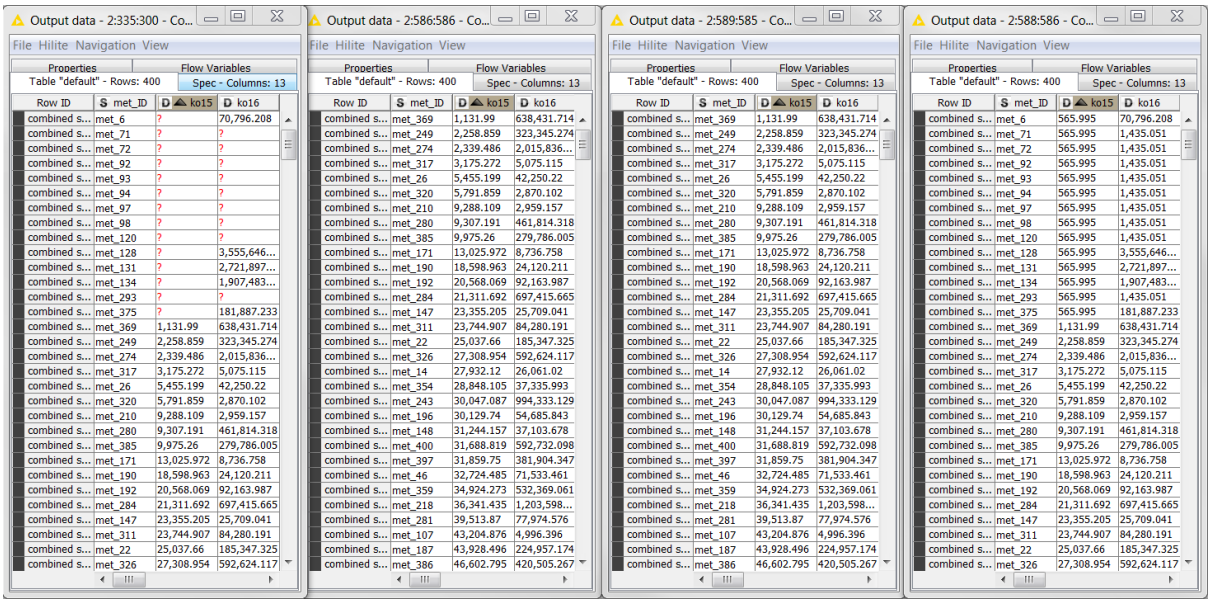

Figure 14: Comparison between the results of the Replace 0 with missing value symbol node (first table starting from left), missing value imputation RF (second), Missing value imputation KNN (third), Missing value imputation SV (last).

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 column (sample) has more than 80 % missing values. In this case, the missing values could be handled with another method such as RF imputation. 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 15, left*). This can be done by searching the node in the Node Repository (bottom left of your KNIME window) and double clicking on it to add it to the workflow. The node should take as input the outcome of the feature filtering metanode and should be configured to filter (include) all sample columns with a missing value threshold = 80 (*Figure 15*,

right). 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 KNN` metanode.

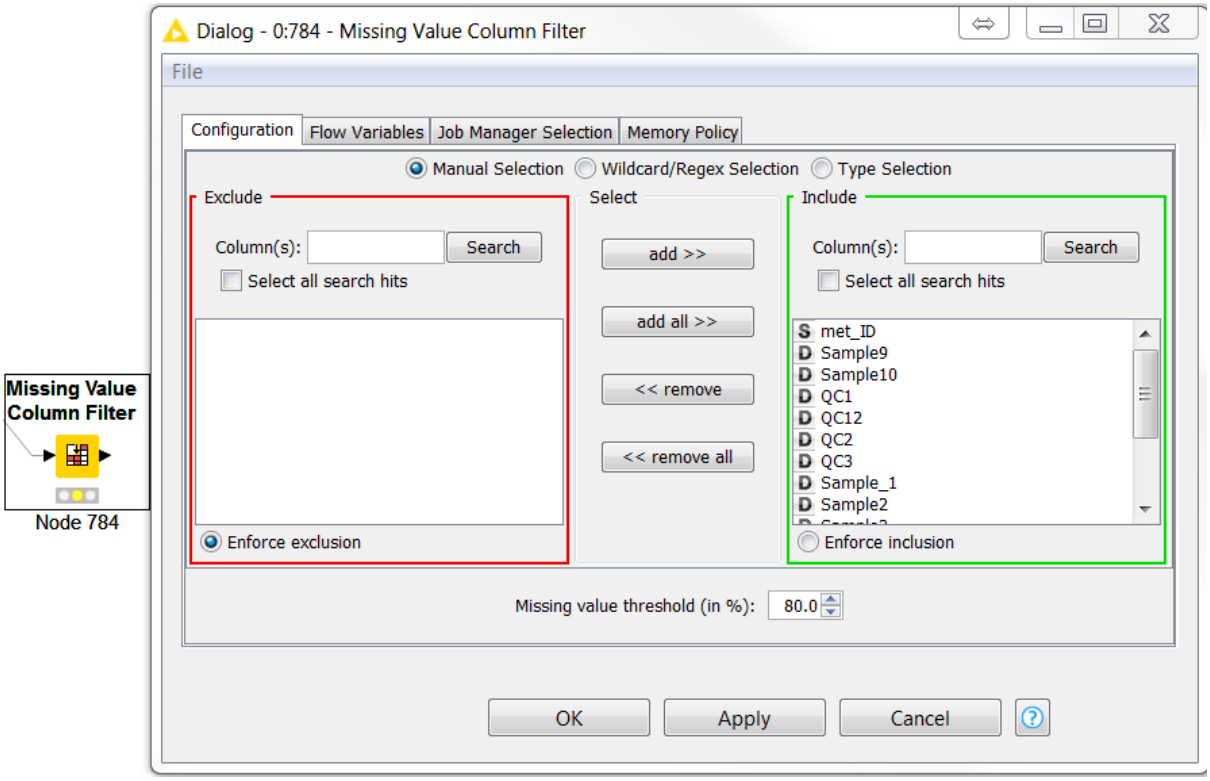

Figure 15: The Missing Value Column Filter can be added before Missing Values Imputation KNN in case the latter fails because one or more columns present 80% or more missing values.

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 metanodes **PQN-QC** and **PQN-ALL** (Figure 16) are designed to perform Probabilistic Quotient Normalisation either on all samples or only on the QCs (if available). In case the **PQN-QC** method is chosen, the **Column filter - Keep only QCs** node should be executed to select only the QC samples (configuration of this node follows the same direction provided above in point 3). Once the column filter has been executed, make sure that its output is connected to the top input port of the **PQN-QC** node, whereas a table with all samples and `met_ID` as columns (such as the output of missing value imputation) should be connected to the bottom input port. In case **PQN-ALL** is chosen, no preliminary nodes need to be run, and its only input port should be connected with a table with all samples and `met_ID` as columns. The output of these metanodes will be a table containing the same rows and columns as the input, but normalised values in the cells.

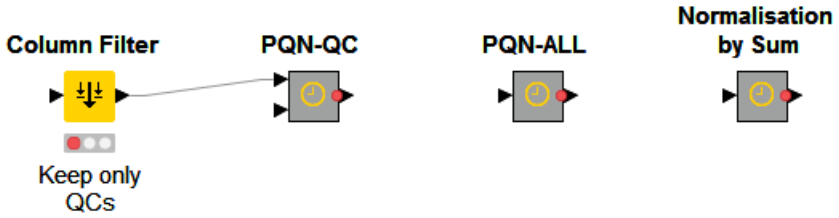

Figure 16: The metanode to perform PQN based on QCs along with the Column Filter node, the metanode to PQN based on all samples, and the metanode for Sum Normalisation are shown.

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.

7.3. In case of a batch-effect deriving from the acquisition of the samples in multiple analytical blocks, a more specific normalisation method should be utilised. Both methods available, **QC-Loess Batch Correction** and **ALL-Loess Batch Correction**, need the preliminary import of sample metadata through the **CSV reader - read file containing sample metadata** node (*Figure 17*).

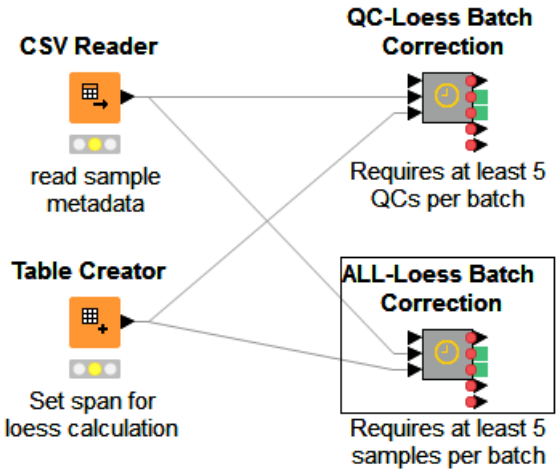

Figure 17: The CSV Reader node imports the sample metadata into the QC-Loess and the ALL-Loess Batch correction metanodes

Please note that a correct functioning of the Loess batch correction methods is possible only if the file contains exactly the column names showed in the example (*Table 1*), and the sample names are the same as those present in the data matrix fed to the CSV reader in point 1. If extra columns with other meta-information (for instance gender) are present, they will simply be ignored by this step without influencing the results. Once this node has been executed, its output should be connected to the central input port of the Batch Correction method of choice.

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
…	…	…	…

A second preliminary step before running the batch correction metanode of choice consists in setting the desired span for loess calculation (default = 1.0) by using the **Table Creator - Set span for loess calculation** (please refer to point 5.1 for instructions on configuration) node,

whose output should be connected to the bottom input port of the Batch correction metanode of choice.
Both **Loess Batch Correction** methods are based on the R script and R wrapper implemented by the Workflow4Metabolomics team to perform the robust locally estimated scatterplot smoothing (LOESS) signal correction (RLSC) (Dunn *et al.*, 2011; Thévenot *et al.*, 2015; Giacomoni *et al.*, 2015). If the **QC -Loess Batch Correction** metanode is selected, correction is performed based on the QC samples, which should be properly indicated in the metadata file imported with the previous **CVS Reader**. Both Batch Correction metanodes will perform Loess correction only if there are at least five QC/samples runs per batch, otherwise linear regression will be used to normalise the data. The Loess Batch Correction metanodes provide five outputs: the corrected data matrix (first), a plot depicting both the sum of intensities for each sample and the PCA score plots of components 1-4 before and after signal correction (second and third), R stdout (fourth) and R stderr (fifth). For more details on this specific step, please refer to the "How to" section in the Workflow4metabolomics website (http://workflow4metabolomics.org/howto).

None of the batch correction metanodes needs to be configured, just make sure that the output from the **PQN-QC/PQN-ALL** node is connected to the top input port, the output from the **CSV reader - read file containing sample metadata** to the central and finally the output from the **Table Creator - Set span for loess calculation** to the bottom input port.

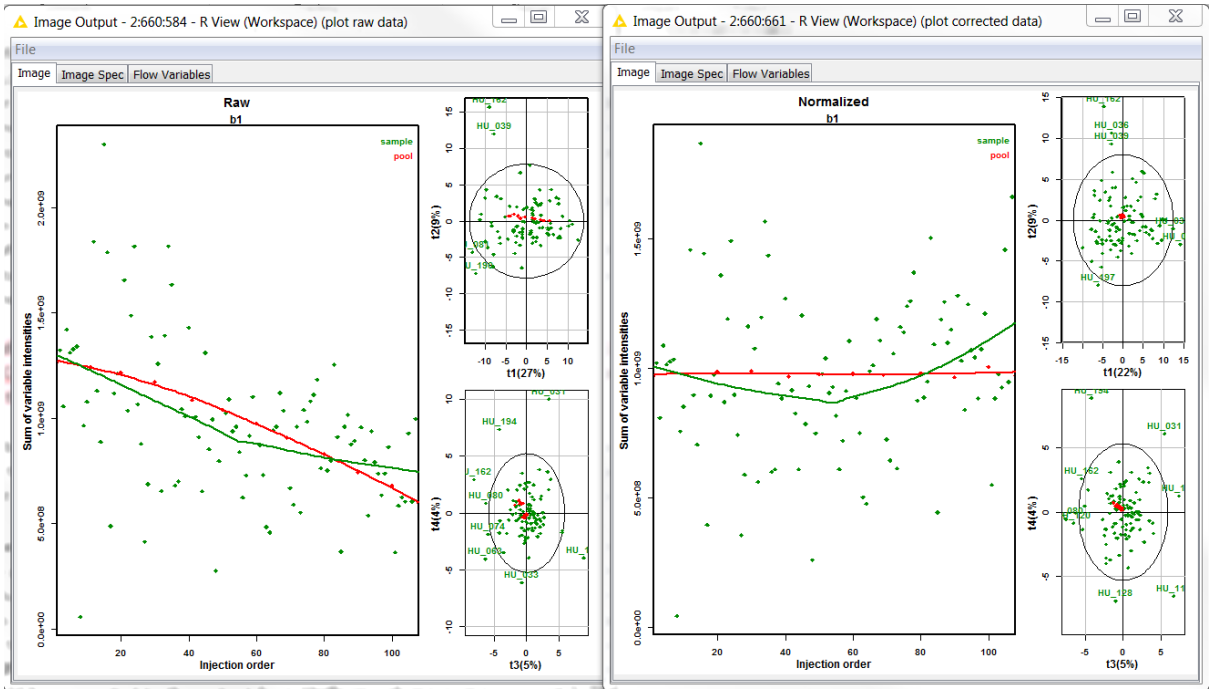

Figure 18: Plots showing the distribution of average intensities across the injection order and the PCA for the components 1-4 before (left) and after (right) QC-LOESS normalisation

8. Annotation

Feature annotation based on accurate mass was implemented by integrating the AccurateMassSearch node of the OpenMS library (Pfeuffer *et al.*, 2017) into the **LIPID MAPS Annotation** and **HMDB Annotation** metanodes, based respectively on LIPID MAPS (Fahy et al., 2007) and HMDB (Wishart *et al.*, 2013, 2009, 2007). The steps required to perform feature annotation are shown in *Figure 19*.

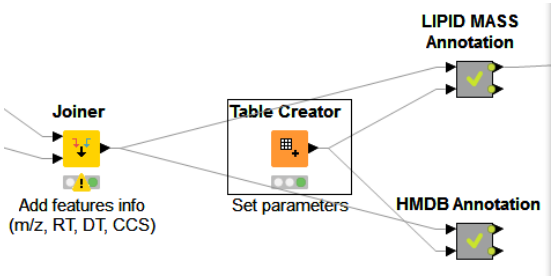

Figure 19: The set of nodes and metanodes required to perform feature annotation based on accurate mass: a joiner to add to the processed data information regarding the m/z of the features, a Table Creator to set several parameters for annotation, and finally the two metanodes to perform feature annotation with either Lipid Maps or HMDB libraries.

8.1. Add the `mz` column to the data matrix, which is missing if any of the steps from point 3 onwards was performed. **The Joiner - Add features info (m/z, RT, DT, CCS)** node can accomplish this task by connecting its top input port to the output of the previously run node (for instance **PQN-QC** if normalisation on the pooled samples was performed), while the bottom input table should be connected to the output from **Uniform column names** in point 2. The configuration pane of this node consists of several tabs. In the `Joiner Setting` tab, make sure that `Inner join` is selected under the `Join Mode` section, while under the `Joining columns` section of the same tab the `met_ID` column has to be selected for both `Top input ('left table')` and `Bottom input ('right table')`. In this way, the values in the ID column will be used to match the two tables and join the normalised signals with the other information relative to them. The second tab `Column Selection` is the one where the columns to be joined from the input tables have to be selected. Its layout is similar to the column filter described above (point 3), with the difference that there will be a section for the top input table and another for the bottom input table. Move the columns that need to be kept in the output in the `Include` box of both sections, and make sure that `Filter duplicates` is selected under `Duplicate columns handling` and `Remove joining columns from bottom input` is ticked under `Joining columns handling` (*Figure 20*). In this way, if by accident a column relative to a given samples is included from both top (processed) and bottom (original) inputs, only the transformed data is kept. This will result in a warning message in the console (`Possible problem in configuration found. The "Duplicate Column Handling" is configured to filter duplicates, but the duplicate columns are not equal since the bottom table has more elements than the top table`) and a triangle symbol on the traffic light below the Joiner node, which can be ignored – they refer to the fact that the top table has more columns than the bottom table, which is exactly the reason why we are joining them! Once the node has been run, its output port can be connected to the first input port of the annotation metanode of choice.

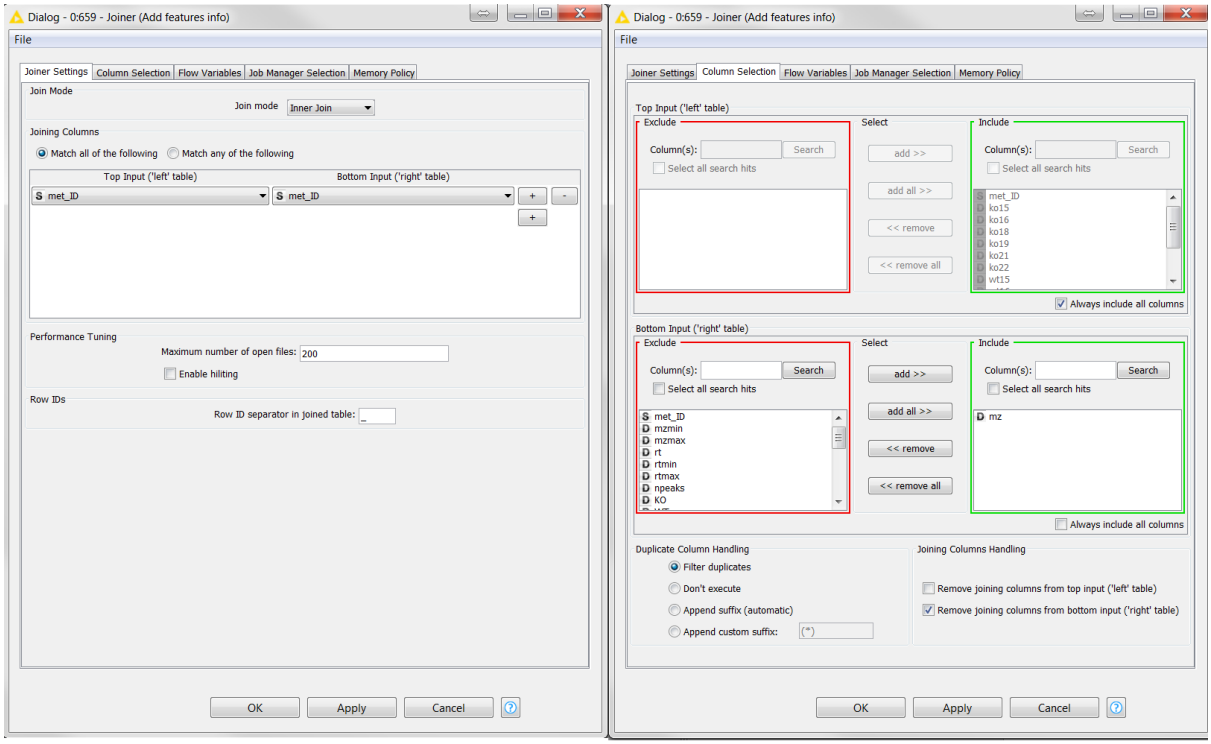

Figure 20: The Joiner setting tab (left) ad the Column Selection tab (right) of the Joiner node are shown

8.2. The node **Table Creator – Set parameters** is needed to set up the ppm error window (default = 20), the polarity of your data and the library to use for feature annotation. These parameters can be inputted by double clicking on the node to open its configuration window (*Figure 21*), and double clicking on the relative cells.
Apart from ppm error and polarity of the data, you will need to define path and file name of several files needed for the annotation metanode to work properly. This is due to the fact that the annotation process as implemented by OpenMS AccurateMassSearch handles files as opposed to KNIME tables. For this reason, the annotation metanode needs to write a csv file containing retention time, m/z and feature intensity, which can be deleted after execution. The name and path of this file need to be set here under the column named temp file and the extension of the file must be .edta. Similarly, path and file name of the files containing the list of positive and negative adducts to import and the library files need to be imputed in their respective columns. These files are:

8.2.1. **Database Mapping File**: 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. The HMDB mapping file (**HMDBMappingFile.tsv**) is provided with the OpenMS installation and can be found 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). The Mapping file for LIPID MAPS (**LMSDFMappingFile.tsv**) was obtained from manipulation of the LIPID MAPS Structure Database (LMSD) with an in-house script and is provided in the `LipidMaps_Annotation` folder of this repository. The name and path of this file should be provided on the fourth column (Database Mapping File) of the **Table Creator – Set parameters** node.

8.2.2. **Database2StructMapping File**: is a tab separated file containing database ID, name of the molecule, InChi and SMILES for each entry in the database to feed as second file input to the annotation metanode. Similarly to the mapping file,

HMDB2StructMapping.tsv is located in the share/OpenMS/CHEMISTRY folder of your OpenMS installation, while the **LMSDF2StructMapping.tsv** is provided in the `LipidMaps_Annotation` folder of this repository. The name and path of this file should be provided on the fifth column (Database2Struct Mapping) of the **Table Creator – Set parameters** node.

8.2.3. **PositiveAdducts.tsv** and **NegativeAdducts.tsv** contain, as the name suggest, 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). These files are provided in the share/OpenMS/CHEMISTRY folder of your OpenMS installation, but in case you would like to remove/add adducts from these list they can be modified and saved somewhere else, and the new user-defined adduct list can be used instead. The name and path of this files should be provided on the sixth (Positive Adducts) and seventh (Negative Adducts) columns of the **Table Creator – Set parameters** node.

Do not change the column names and write the polarity parameter in lower case (i.e. "positive" or "negative"), otherwise the following annotation will fail.
Once the parameters have been defined and the node executed, the output port of this node can be connected to the second input port of the annotation metanode of choice.

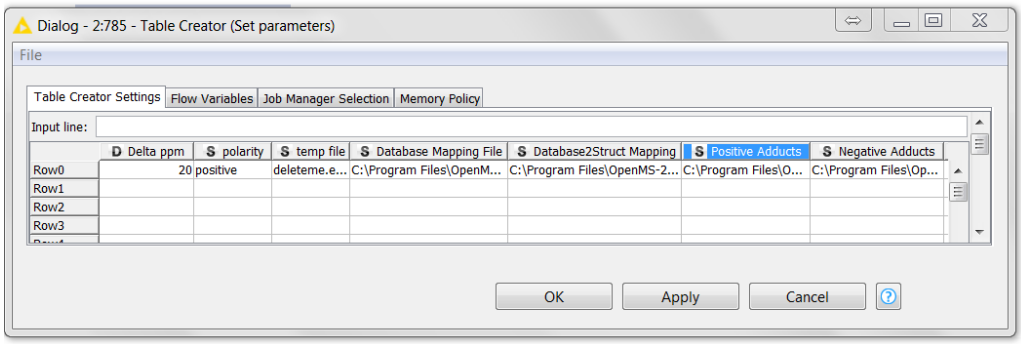

Figure 21: The configuration window of the Table Creator - Set parameters metanode

8.3. Once all the parameters have been set, we can proceed with the feature annotation by using either the **LIPID MAPS Annotation** or the **HMDB Annotation**. These metanodes will produce two output files (*Figure 22*). The first one will look identical to that fed to the annotation metanode, with the only addition of one column named either LIPID MAPS ID or HMDB ID containing the database identifier(s) annotated which each feature. The second output will instead contain the database ID, name, adduct type, Δppm, formula, SMILES, InChI and the met_ID to which it was annotated.

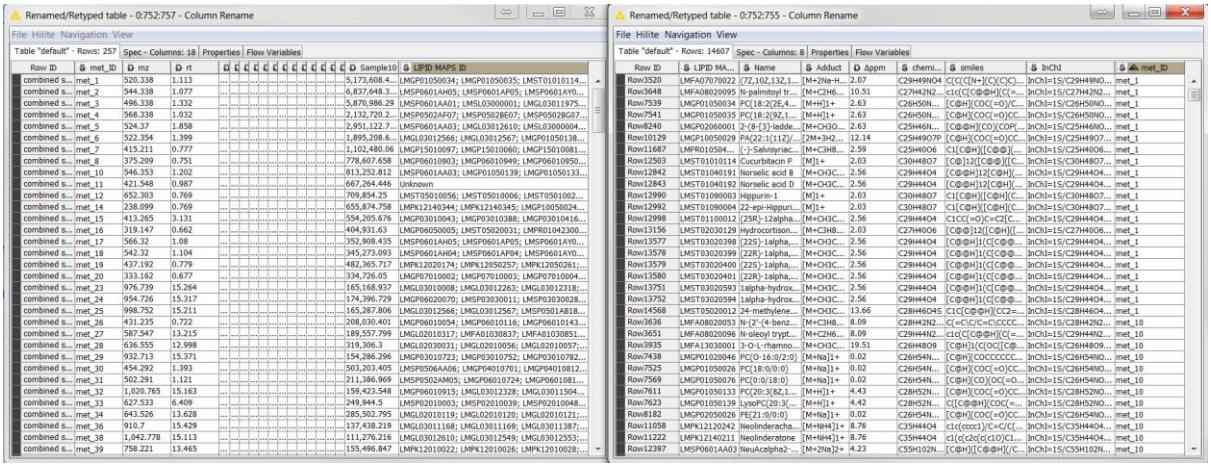

Figure 22: The first output of the annotation metanodes is shown to the left, while on the right hand side the bottom output is shown.

9. Output

The processed data can be written to an output file in the desired format. Configuration of the **CSV Writer** node includes indicating location and filename of the output file to be written by clicking on `Browse`, ticking `Write column header` and selecting whether the file should be overridden in case of name already assigned, or if the node execution should fail. Once the node has been configured, it can be run and the output is ready for any other analysis that you wish to perform outside KNIME.

In case you would like to save the output in the xls format, the **Excel Writer (XLS)** node can be configured in a way similar to that described for the CSV reader.

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.*, 2004), 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 does not contain injection of neither QC samples nor blanks, hence feature filtering could be not be performed and data can be normalised only on all samples. The workflow, shown in *Figure 23*, 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. **Replace 0 with missing values** and **Missing Values imputation KNN** to insert the missing value symbol and replace its value;
1.5. **Normalisation by Sum**;
1.6. **HMDB Annotation** preceded by the **Joiner** and the **Table Creator** nodes;
1.7. **CSV Writer** to write to an external file the output of the processing.

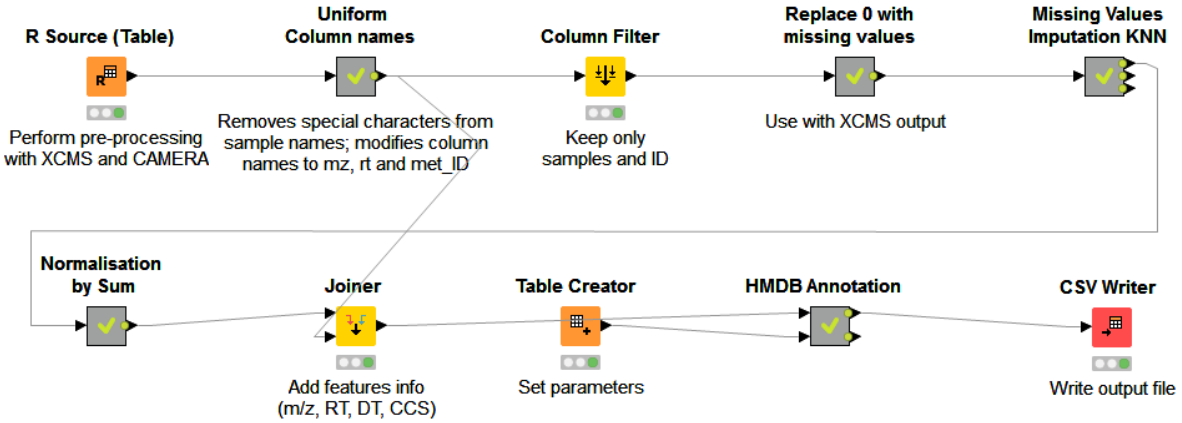

Figure 23: KniMet workflow used to process the data deconvoluted within the pipeline with the R Source node

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 files "Test_LCIMMS_QC_features.tsv" and "Test_LCIMMS_Blanks_features.tsv"in the `testfiles` folder of this repository were obtained performing feature extraction with MassProfiler on mouse liver samples and either QC or blanks run on the Agilent 6560 Ion Mobility Q-TOF LC-MS in positive ion mode. These files can be downloaded and used to follow this example. In *Figure 24* the connection between nodes and metanodes to process the matrix containing QCs are shown, namely:

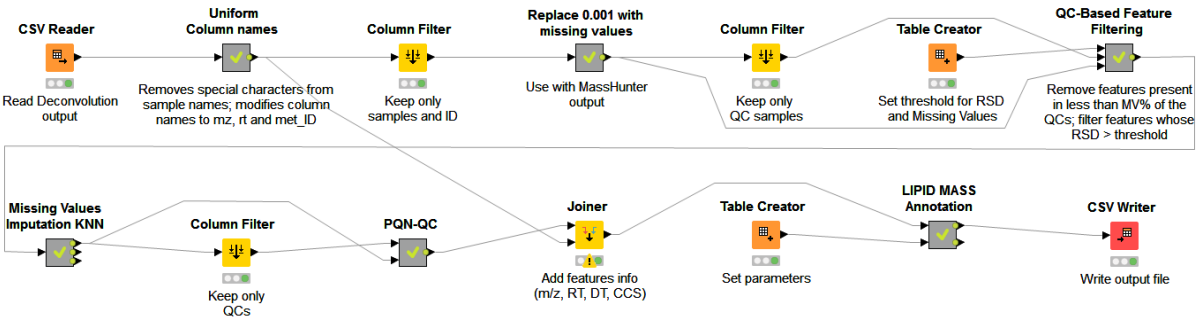

Figure 24: The complete workflow for the analysis of LC-IM-MS data containing pooled samples and pre-processed outside the KniMet workflow

2.1. Import of 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. **Replace 0.001 with missing values** to insert the missing value symbol;
2.5. Selection of QC samples, threshold for missing values and RSD and feature filtering based on them with the **Column Filter**, the **Table Creator** nodes and the **QC-based feature filtering** metanode.
2.6. **Missing Values imputation KNN** to impute missing values;

2.7. Selection of the QC samples with the **Column Filter** node and normalisation **with the PQN-QC** metanode;
2.8. **Lipid Maps Annotation** preceded by the **Joiner** and the **Table Creator** nodes;
2.9. **CSV Writer** to write to an external file the output of the processing.

On the other hand, the matrix containing blanks and not containing QCs can be processed with a very similar workflow (Figure 25) differing from the one described above only in the feature filtering and in the normalisation method, which are now based on Blanks and all samples, respectively.

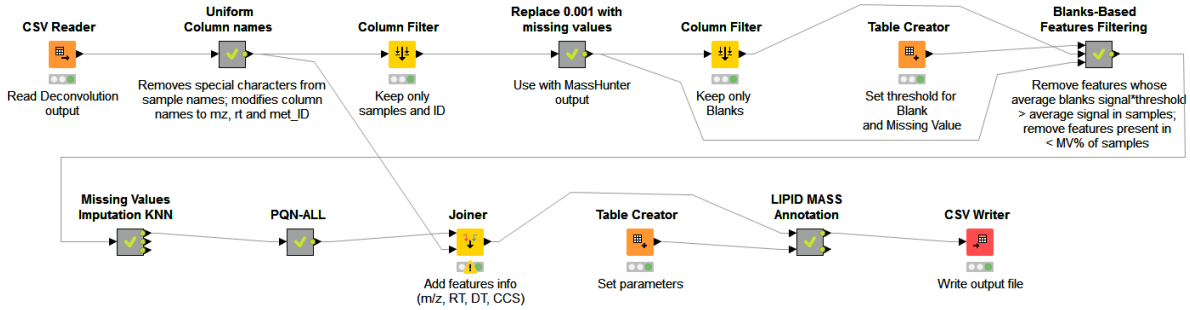

Figure 25: The complete workflow for the analysis of LC-IM-MS data containing blank samples and pre-processed outside the KniMet workflow

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:

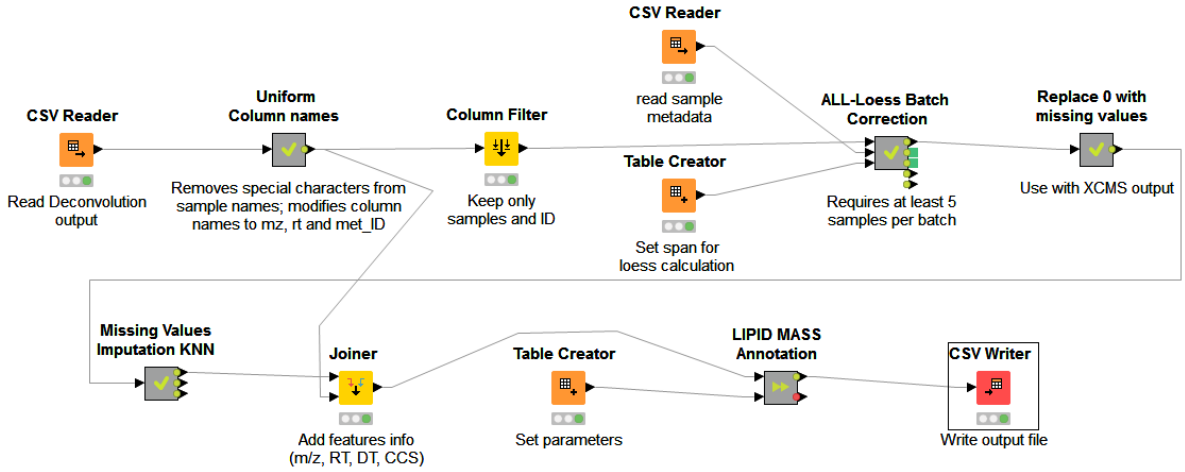

Figure 26: The KniMet workflow adapted to perform Loess batch correction based on all samples

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. **CVS Reader** node to import the sample metadata;

3.5. **ALL-Loess Batch Correction** metanode to perform batch correction on all samples, preceded by the **CSV Reader – read sample metadata** and the **Table Creator – Set span for loess calculation**;
3.6. **Replace 0 with missing values** to insert the missing value symbol followed by **Missing Values imputation KNN** to impute missing values;
3.7. **Lipid Maps Annotation** preceded by the **Joiner** and the **Table Creator** nodes;
3.8. **CSV Writer** to write to an external file the output of the processing.

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

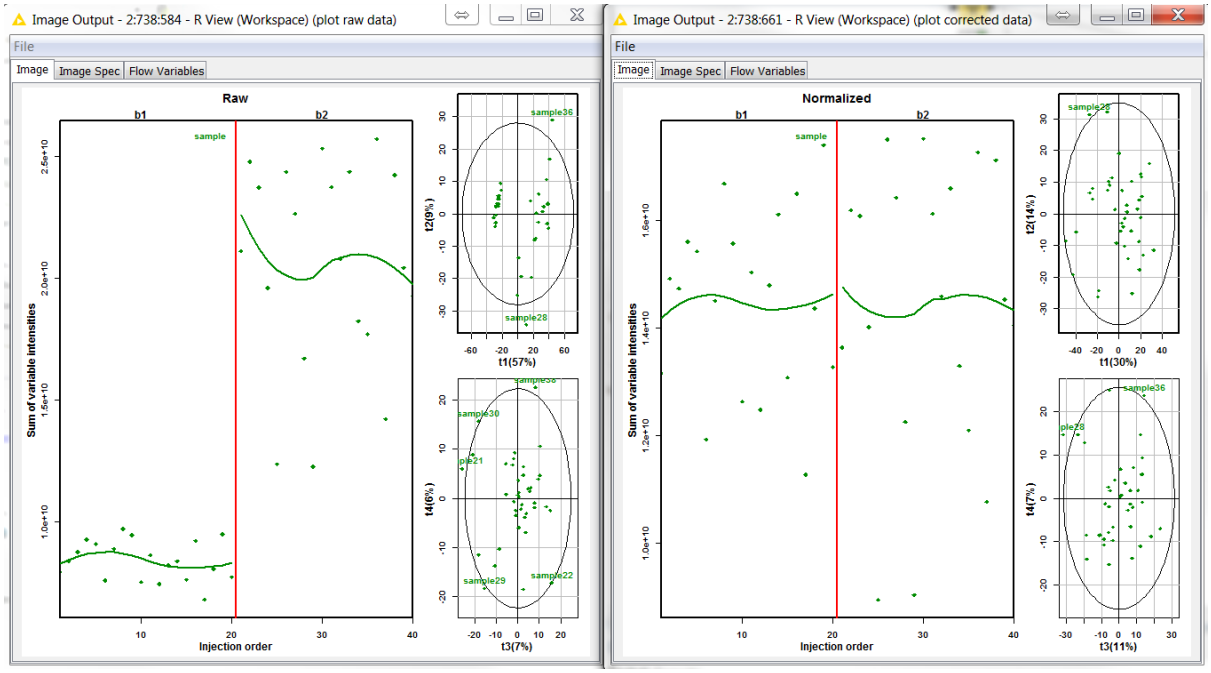

*Figure 27: The image outputs of the **All-Loess Batch Correction** metanode for raw (left) and normalised (right) data present on the left hand side the sum of intensity for all variables in each given sample as a function of the order of injection, along with the fitted loess curve. On the right hand 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.

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.* (2004) 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.