Filter W4M data by sample class

Purpose of the `w4mclassfilter` package

The purpose of the w4mclassfilter R package is to provide the computational back-end of a Galaxy tool for inclusion in Workflow4Metabolomics (W4M).

Galaxy tools are file-oriented; because of this, the `w4mclassfilter::w4m_filter_by_sample_class` method reads from and writes to files. General-purpose R packages usually use data structures in memory for their input and output, which may mean that this R package is not generally useful outside of the context of Galaxy.

The `w4mclassfilter::w4m_filter_imputation` function is the default imputation method used by `w4m_filter_by_sample_class`; if other methods are to be used in the Galaxy tool, they might best be incorporated into the `w4mclassfilter` R package, although they could be implemented in another R package to be used by the Galaxy tool.

How the `w4m_filter_by_sample_class` function is used

A Galaxy tool wrapper invokes `w4m_filter_by_sample_class`. For exploratory or debugging purposes, the package may be installed loaded into R and help may then be obtained with the following command:

```
?w4mclassfilter::w4m_filter_by_sample_class
```

W4M uses the XCMS and CAMERA packages to preprocess GC-MS or LC-MS data, producing three files (which are documented in detail on the Workflow4Metabolomics (W4M) web site).
In summary:

1. `sampleMetadata.tsv`: a tab-separated file with metadata for the samples, one line per sample

- One column of this file indicates the *class* of the sample.
- It is the class that is used by this function to determine whether to include the sample in, or exclude the sample from, further analysis.

2. `variableMetadata.tsv`: a tab-separated file with metadata for the features detected, one line per feature

- A feature is a location in the two dimensional space defined by the GC-MS or LC-MS data set, which corresponds to a compound or a group of compounds.
- One dimension is the mass-to-charge ratio, `m/z`.
- The other dimension is the `retention time`, i.e., how long until the solvent gradient eluted the compound(s) from the column.

3. `dataMatrix.tsv`: a tab separated file with the MS intensities for each sample for each feature:

- There is one column per sample.
- There is one row per feature.
- If a feature is missing for a sample, the intensity value is `NA`.
- For numerical reasons, intenisities may be negative, but this has no meaning in the real world.

By default, the `w4m_filter_by_sample_class` function imputes negative and `NA` intensity values as zero using the `w4m_filter_imputation` function.

When `w4m_filter_by_sample_class` is invoked, an array of class names is supplied in the `classes` argument. If the `include` argument is true, then only samples whose *class* column in `sampleMetadata.tsv` will be *included in* the output; by contrast, if the `include` argument is false, then only samples whose *class* column in `sampleMetadata.tsv` will be *excluded from* the output.

Even if no rows or columns of the `dataMatrix.tsv` input have zero variance, there is the possibility that eliminating samples may result in some rows or columns having zero variance, adversely impacting downstream statistical analysis. Consequently, `w4m_filter_by_sample_class` eliminates these rows or columns and the corresponding rows from `sampleMetadata.tsv` and `variableMetadata.tsv`.

When `w4m_filter_by_sample_class` completes running, it writes out updated `sampleMetadata.tsv`, `variableMetadata.tsv`, and `dataMatrix.tsv` files. The paths to the output files *must* be different from the paths to the input files.

As of v0.98.3, `w4m_filter_by_sample_class` can interface not only with files but also with data structures; see release note below.

New in release v0.98.2 - support regular expressions

Beginning with v0.98.2, w4mclassfilter supports use of R regular expression patterns to select class-names.

The R `base::grepl` function (at the core of this functionality) uses POSIX 1003.2 standard regular expressions, which allow precise pattern-matching and are exhaustively defined at: http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap09.html

However, only a few basic building blocks of regular expressions need to be mastered for most cases:

- `^` matches the beginning of a class-name
- `$` matches the end of a class-name
- `.` outside of square brackets matches a single character
- `*` matches character specified immediately before zero or more times
- square brackets specify a set of characters to be matched.

Within square brackets

- `^` as the first character specifies that the list of characters are those that should * not * be matched.
- `-` is used to specify ranges of characters

Caveat: The tool wrapper uses the comma (`,`) to split a list of sample-class names, so *commas may not be used within regular expressions for this tool*

First Example: Consider a field of class-names consisting of `marq3,marq6,marq9,marq12,front3,front6,front9,front12`

- The regular expression `^front[0-9][0-9]*$` will match the same sample-classes as `front3,front6,front9,front12`
- The regular expression `^[a-z][a-z]3$` will match the same sample-classes as `front3,marq3`
- The regular expression `^[a-z][a-z]12$` will match the same sample-classes as `front12,marq12`
- The regular expression `^[a-z][a-z][0-9]$` will match the same sample-classes as `front3,front6,front9,marq3,marq6`

Second Example: Consider these regular expression patterns as possible matches to a sample-class name 'AB0123':

- `^[A-Z][A-Z][0-9][0-9]*$` - MATCHES `**^AB0123$**`
- `^[A-Z][A-Z]*[0-9][0-9]*$` - MATCHES `**^AB0123$**`
- `^[A-Z][0-9]*` - MATCHES `**^A** B0123$` (The first character is a letter, `*` can specify zero characters, and end of line did not need to be matched.)
- `^[A-Z][A-Z][0-9]` - MATCHES `**^AB0** 123$` (The first two characters are letters aind the third is a digit.)
- `^[A-Z][A-Z]*[0-9][0-9]$` - NO MATCH (The name does not end with the pattern `[A-Z][0-9][0-9]$`, i.e., it ends with four digits, not two.)
- `^[A-Z][0-9]*$` - NO MATCH (The pattern specifies that second character and all those that follow, if present, must be digits.)

New in release v0.98.3 - more flexible input and output

R package v0.98.3 - add support for env, list, data.frame, and matrix I/O * To support XCMS outside the context of Galaxy, `w4m_filter_by_sample_class` now supports input from and output to data structures as follows:

Inputs:

1. `dataMatrix_in`

- if a string, treat as a file path (i.e., same as in previous releases)
- else if an env or list, read `dataMatrix_in$dataMatrix`
- else if a data.frame, use `as.matrix(dataMatrix_in)`
- else if a matrix, use directly
- else error

2. `sampleMetadata_in`

- if a string, treat as a file path (i.e., same as in previous releases)
- else if an env or list, read `sampleMetadata_in$sampleMetadata`
- else if a data.frame, use directly
- else error

3. `variableMetadata_in`

- if a string, treat as a file path (i.e., same as in previous releases)
- else if an env or list, read `sampleMetadata_in$sampleMetadata`
- else if a data.frame, use directly
- else error

Outputs:

1. `dataMatrix_out`

- if a string, treat as a file path (i.e., same as in previous releases)
- else if an env or list, write `dataMatrix_out$dataMatrix`
- else error

2. `sampleMetadata_out`

- if a string, treat as a file path (i.e., same as in previous releases)
- else if an env or list, write `sampleMetadata_out$sampleMetadata`
- else error

3. `variableMetadata_out`

- if a string, treat as a file path (i.e., same as in previous releases)
- else if an env or list, write `sampleMetadata_out$sampleMetadata`
- else error