mrbin

Matthias S. Klein

Version 1.9.5

Contents

1 Introduction **2**

2 mrbin: Magnetic Resonance Binning, Integration and Normalization **2**
2.1 Prerequisites and Considerations . . . 2
2.2 Running mrbin in Interactive Mode . . . 2
2.3 Setting Parameters at the Command Line . . . 5
2.4 Processing of mrbin Objects . . . 6
2.5 Adding Metadata to mrbin Objects . . . 7
2.6 Editing mrbin Objects . . . 8
2.7 Example: 2D Data . . . 9
2.8 Example: Lipid Data Analysis . . . 13
2.9 Recreating Data and Parameters . . . 14
2.10 mrbin Workflow . . . 14

3 atnv: Affine Transformation of Negative Values **15**

4 mrplot: NMR Plotting **15**

5 Heatmap Plots **17**

6 Basic Plotting Commands **18**

7 fia: Feature Impact Assessment of Artificial Neural Networks **19**

8 Known Issues **20**
8.1 Linux tcl Issues . . . 20
8.2 Pop-up Windows Missing on Apple Computers, RStudio . . . 20
8.3 Pop-Up Windows . . . 20
8.4 Firewall Warnings . . . 20

9 License **20**

10 Citation **21**

11 References **21**

1 Introduction

This package is a collection of functions for processing and analyzing metabolomics data.
The namesake function mrbin() uses spectral binning to convert 1D or 2D Nuclear Magnetic Resonance (NMR) data into a matrix of values suitable for further data analysis and performs basic processing steps in a reproducible way. Negative values, a common issue in NMR data, are replaced by positive values. All used parameters are stored in a readable text file and can be restored from that file to enable exact reproduction of the data at a later time.
The atnv algorithm for replacing negative values in NMR data sets can be employed using atnv().
NMR plotting functions are found in mrplot().
Artificial Neural Network features can be analyzed using Feature Impact Assessment (FIA) using the function fia().

2 mrbin: Magnetic Resonance Binning, Integration and Normalization

The main NMR binning functions of this package are controlled via the mrbin() function. Results returned include the final bin list and a set of used parameters.

2.1 Prerequisites and Considerations

To use this package, you will need NMR data in the Bruker file format accessible on your computer. Please make sure your data is Fourier transformed, phase corrected, baseline corrected, and correctly referenced. The data has to be stored in folders according to standard Bruker folders, that means foldername/1/pdata/1 etc. Experiment numbers and processing numbers can be freely chosen.

This package has been tested for 1D ^{1}H NOESY and 2D ^{1}H-^{13}C HSQC spectra.

Before starting mrbin, take a look at your NMR data, for example in Bruker Topspin, and decide on the following parameters (you will be able to set the values of these parameters for your data during running mrbin):

- Bin area: Area where signals are observed in your data set
- Bin width: Should match roughly the width of a singlet peak in your data set. Given in ppm.
- Bin height (only 2D): Should match roughly the height of a singlet peak in your data set. Given in ppm.
- Solvent area: Area to exclude to remove solvent artifacts
- Additional areas to be removed: Any other area containing artifacts, such as streaks surrounding strong peaks.

mrbin will also show you preview plots for these parameters during the run.

2.2 Running mrbin in Interactive Mode

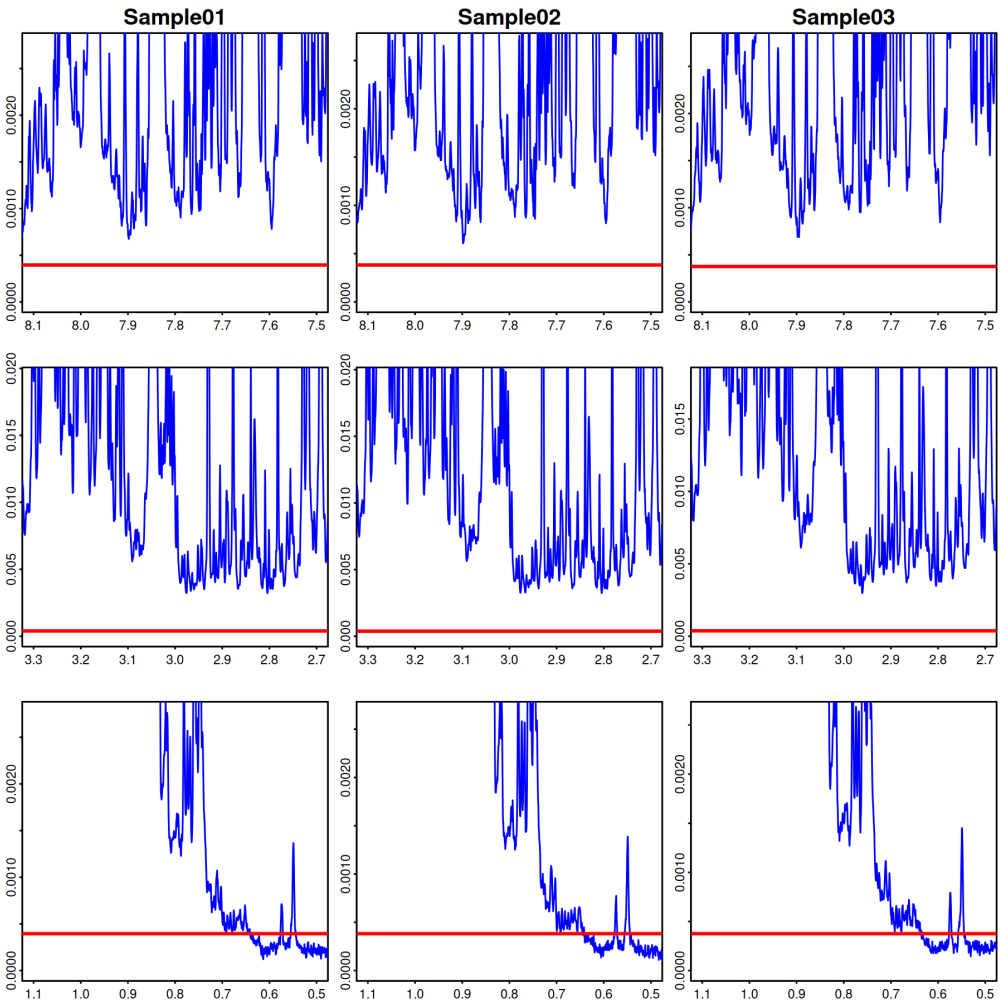
You can start mrbin using the following code:

```
> library(mrbin)
```

```
> results<-mrbin()
```

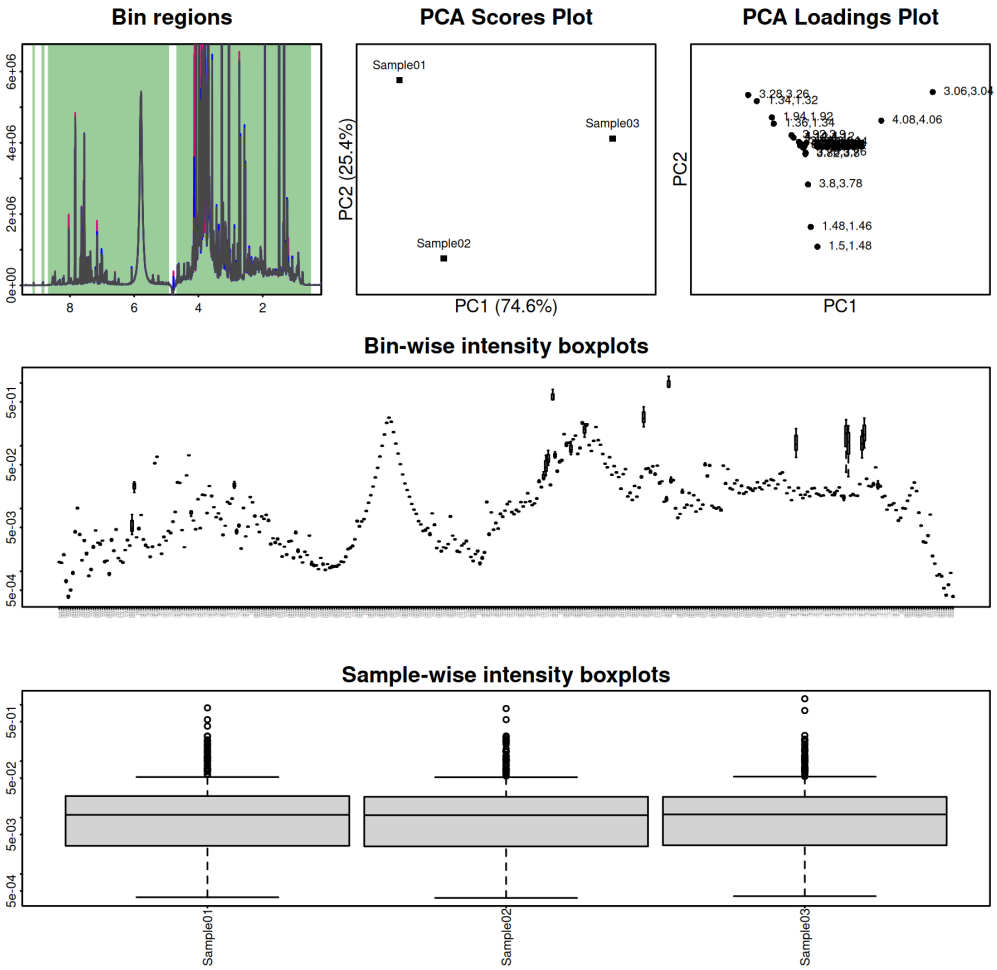
This will start a series of questions that will guide you through the parameters to be used. Preview plots will help adjusting parameters to the current dataset and allow for spotting any inconsistencies in the spectral data.

If noise removal is used, preview plots of the signal-to-noise ratios (SNR) of selected spectra are provided to help chose an SNR level matching the used data. Please note that the visual representation of SNR levels is approximate, as noise removal is performed on the binned data, but not on the raw spectra. To achieve a satisfactory preview of SNR levels, each spectrum is scaled in a way so its peak heights are equivalent to the respective bin value. The median of these ratios is used as a scaling factor for the whole spectrum.



Please be aware that the calculated bin values can be viewed as pseudo-integrals of the contained signals. This is because the generated bin values are calculated as the average value of all data points within each bin, multiplied by the area of the bin. This avoids the issue of different numbers of data points within a bin in different spectra, and corrects for bins that are different in their size.

After finishing, a summary is displayed in the console. In addition, several plots are shown. The output plots allow for reviewing data quality. Please note that any negative values are not shown in the boxplots for clarity. Any negative values still remain in the dataset, though.
:



mrbin() returns an (invisible) object of type mrbin, containing the following variables:

- bins: A matrix containing bin data for all samples, Depending on the option you chose, the data will be cleaned up and scaled.
- parameters: A list containing all parameters used to create the bin matrix.
- metadata: A list containing metadata, if provided.
- transformations: A character vector containing information on the data transformation and scaling that has been performed, for example reference scaling, PQN, atnv, log transform, etc.
- changeLog: A data.frame containing information on documented changes that were made to the data, including time stamps.
- changeValues: A list containing control values, enabling verifying changes by checkmrbin(results)

If saving to disk is selected, several files will be written to the chosen directory:

- A .Rdata file containing the generated mrbin data object
- A .pdf file containing quality control plots of the raw binned data
- A .pdf file containing quality control plots of the processed binned data
- A .pdf file containing preview plots of the chosen signal-to-noise ratios (SNR) of selected spectra

- A .txt file containing all parameters and potential warning messages from the mrbin run. This file can be reloaded to mrbin to recreate a dataset for reproducibility

The generated quality control plots show the following panels:

- Previews of a few selected spectra, overlaid with the areas of the bins generated (in green) to make sure all peaks of interest are covered
- PCA scores and loadings plots of the generated bin data to aid in spotting data quality issues
- Boxplots of bin intensities per bin (feature) to aid in spotting data quality issues. Please note that for the purpose of these boxplots only, any negative values are not shown
- Boxplots of bin intensities per sample to aid in spotting data quality issues. Please note that for the purpose of these boxplots only, any negative values are not shown

2.3 Setting Parameters at the Command Line

Parameters can also be submitted at the command line. With the default setting of silent=FALSE, the user will be guided through the user input questionnaire to view, and make adjustments to, the parameters submitted at the command line.

When using silent=TRUE, this will set up all parameters and run all steps without asking for further user input. This should only be used after carefully reviewing all parameters in interactive mode, as data issues might be missed.

The following example provides parameters for analyzing a 1D data set.

```
> results<-mrbin(parameters=list(dimension="1D",
+              binwidth1D=.02,
+              referenceScaling="Yes",
+              removeSolvent="Yes",
+              solventRegion=c(4.95,4.65),
+              noiseRemoval="Yes",
+              signal_to_noise1D=35,
+              noiseThreshold=0.75,
+              PQNScaling="No",
+              fixNegatives="No",
+              logTrafo="No",NMRvendor="mrbin",
+              useAsNames="Spectrum titles",
+              NMRfolders=c("C:/Bruker/TopSpin/data/guest/nmr/1/10/pdata/10",
+                           "C:/Bruker/TopSpin/data/guest/nmr/2/10/pdata/10",
+                           "C:/Bruker/TopSpin/data/guest/nmr/3/10/pdata/10")
+      ))
```

In the example shown above, spectra from the mrbin package are used. To use your own data, change the folder names to the full names of the folders containing the files "1r" (or "2rr" for 2D data) as follows (Please note that on Windows computers, backward slashes need to be replaced by forward slashes):

```
>    NMRfolders=c("C:/Bruker/TopSpin/data/guest/nmr/1/10/pdata/10",
+              "C:/Bruker/TopSpin/data/guest/nmr/2/10/pdata/10",
+              "C:/Bruker/TopSpin/data/guest/nmr/3/10/pdata/10")
```

2.4 Processing of mrbin Objects

All processing algorithms, such as noise removal, scaling, etc., can be performed within mrbin(). However, it is also possible to run these (optional) steps later. For this, you should select "No" in mrbin() when asked about noise removal, PQN scaling, atnv, and log transform. Please note that scaling to the reference substance happens at the raw spectrum level and needs to be done during running mrbin(); it cannot be performed later. Please consider that, apart from noise removal, most of the possible processing methods might not be appropriate for your specific dataset and/or data analysis plan. Therefore, you will need to make a decision based on your specific situation. Here is a short overview of implemented commands, which should be performed in the order shown below:

Set signal-to-noise ratios (SNR) interactively:

```
> results<-setNoiseLevels(results)
```

To remove noise signals using the newly defined SNR levels, use removeNoise(). Noise levels are individually checked for each bin of each spectrum. Expected noise levels are calculated for each bin of each spectrum, based on the estimated noise level of the respective spectrum, corrected for the area of each bin and the number of data points within this bin (inverse square root relationship). Noise removal needs to be performed before any other scaling or transformation that changes bin values.

```
> results<-removeNoise(results)
```

Remove negative values by atnv scaling (see below for details):

```
> results<-atnv(results)
```

Correction for individual dilutions. Use if sample volumes or sample weights were not identical for all samples. The bin value of a sample will be multiplied by the respective dilution factor:

```
> results<-setDilutionFactors(results)
> results<-dilutionCorrection(results)
```

Probabilistic Quotient Normalization (PQN) scaling, this is similar, but better, than scaling to the total sum. Use for urine or tissue extracts:

```
> results<-PQNScaling(results)
```

Logarithm transform to make data more normal. Data will no longer be linearly correlated to molecular concentration but rather represent fold changes. This might only work after atnv transform due to negative values:

```
> results<-logTrafo(results)
```

Scaling to unit variance. This is rarely necessary, but might be advantageous, e.g. for use in artificial neural networks. Data will no longer be linearly correlated to molecular concentration:

```
> results<-unitVarianceScaling(results)
```

Plot results:

```
> plotResults(results)
```

Removing a spectrum from the dataset. This should not be done after noise removal, PQN scaling, unit variance scaling, or atnv, as these methods employ data from all samples in the dataset. Only removing a sample before invoking any of these methods will ensure reproducible data transformations.

```
> results<-removeSpectrum(results)
```

2.5 Adding Metadata to mrbin Objects

You can add or edit metadata, including metabolite identities, for an mrbin object as follows:

```
> results<-metadatamrbin(results)
```

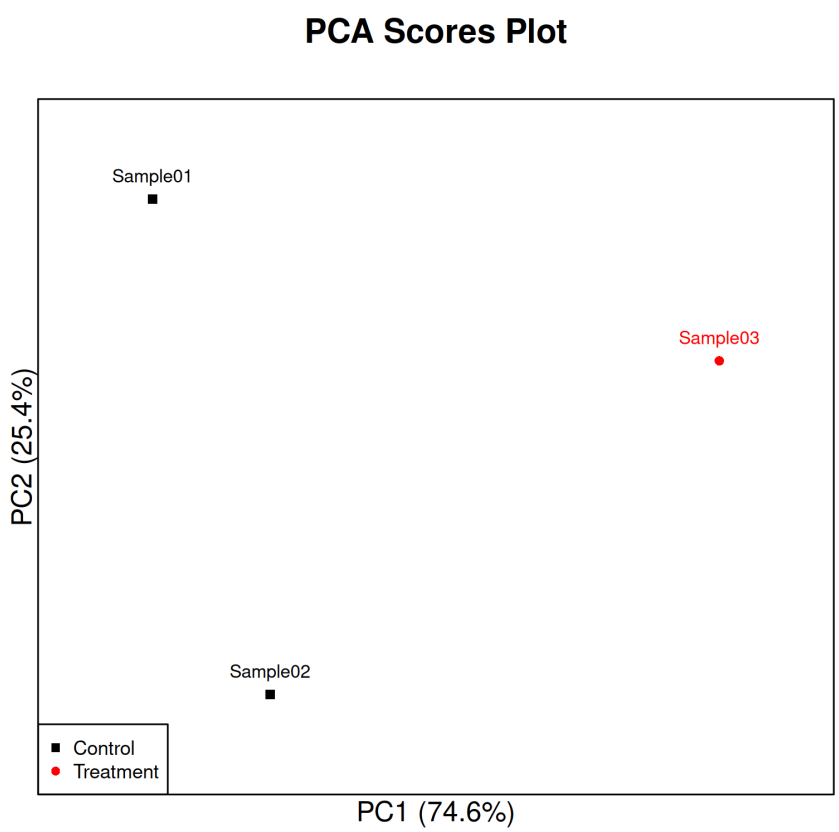
If you choose "edit" for any of the categories, an editor window may open. After editing text, close this window and select "Save changes" to save your edits. Make sure to keep the opening and closing quotation marks.

Alternatively, you can edit metadata fields in the command line:

```
> results<-metadatamrbin(results,metadata=list(
+   projectTitle="Test project",
+   factors=factor(c("Control","Control","Treatment"))))
```

If you choose to define treatment groups in this step, you can plot a PCA with color coded group information as follows:

```
> plotPCA(results)
```

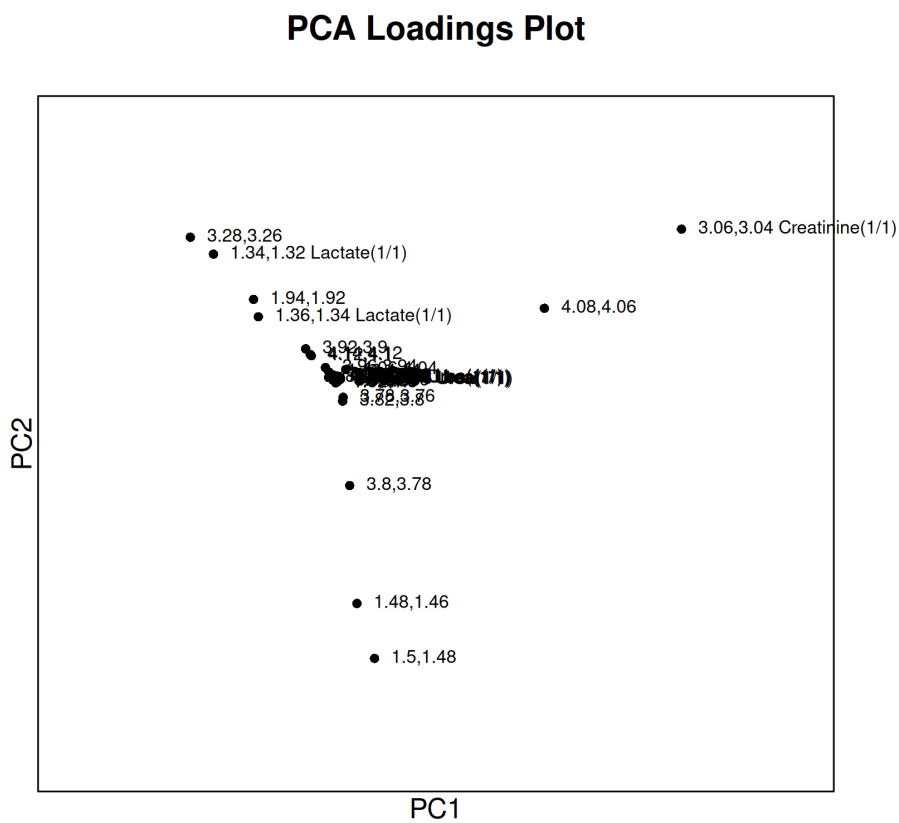


The results can be annotated with potential metabolite identities using the command metadatamrbin(), or in the command line as follows. Left, right, top, and bottom borders of the metabolite signal need to provided as a matrix, along with metabolite names. The provided data will then be used to annotate the bins of the dataset. Bins might be assigned to multiple metabolites if they cover the areas of several potential signals.

```
> #Annotate the dataset with signal identities
> metaboliteIdentities<-matrix(c(1.346,1.324,21,23,
+                                3.052,3.043,30.5,33.5,
+                                5.7,6.0,0,150),
+                     ncol=4,byrow=TRUE)
> rownames(metaboliteIdentities)<-c("Lactate","Creatinine","Urea")
> results<-annotatemrbin(results,metaboliteIdentities=metaboliteIdentities)
```

The annotated loadings plot of the PCA can be displayed as follows:

```
> plotPCA(results,loadings=TRUE,annotate=TRUE)
```



2.6 Editing mrbin Objects

As mrbin objects use a documented change log, ideally you would use only functions from the mrbin package to change them (e.g. log transform), as these are documenting changes. If you wish to perform other edits, please do so using the command editmrbin(), as in the following example:

```
> results<-editmrbin(
+   results,
+   bins=results$bins,#omit this line if no changes are made to bins
```

```
+   parameters=list(noiseThreshold=0.75),
+   metadata=list(projectTitle="Test project"),
+   comment="Changed title and noise parameters",
+   transformations="Scaling"#If you change bins, provide a short explanation
+   )
```

To see the change log, use:

```
> results$changeLog
```

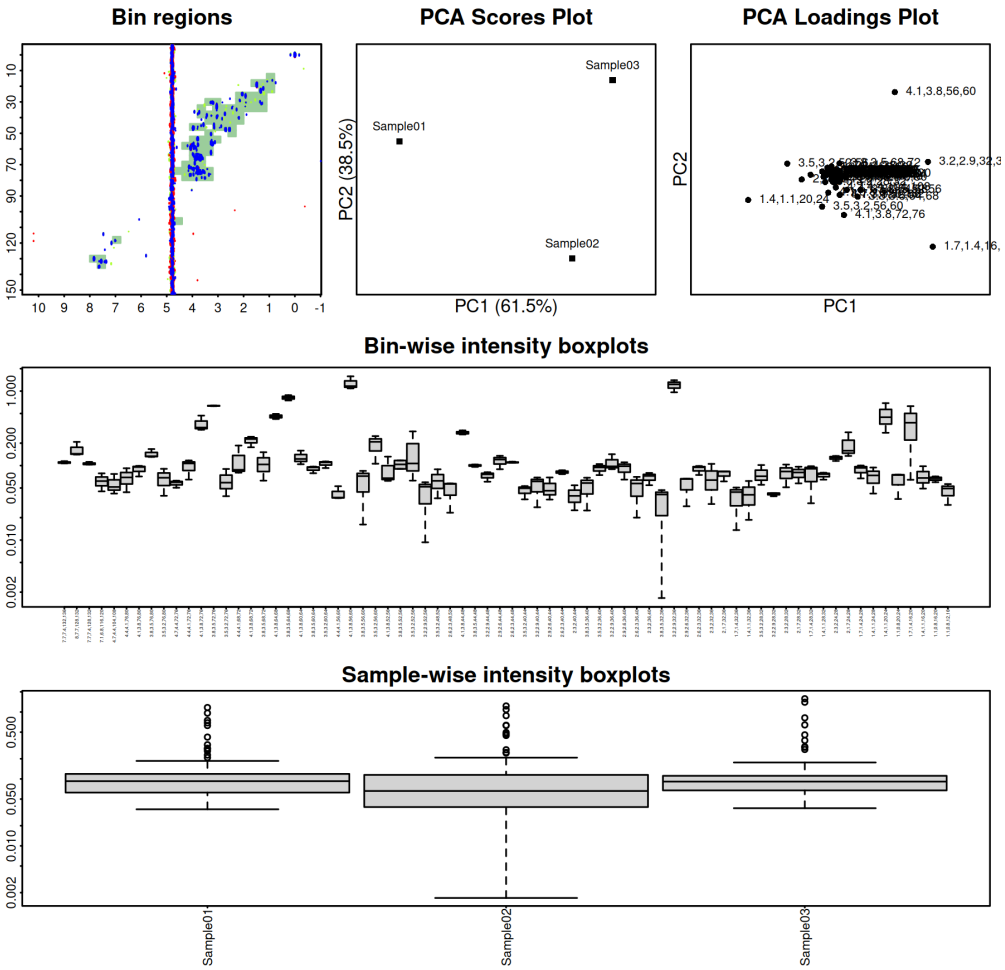
To see a summary of scalings and transformations, and check for undocumented edits to the data, use:

```
> checkmrbin(results)
```

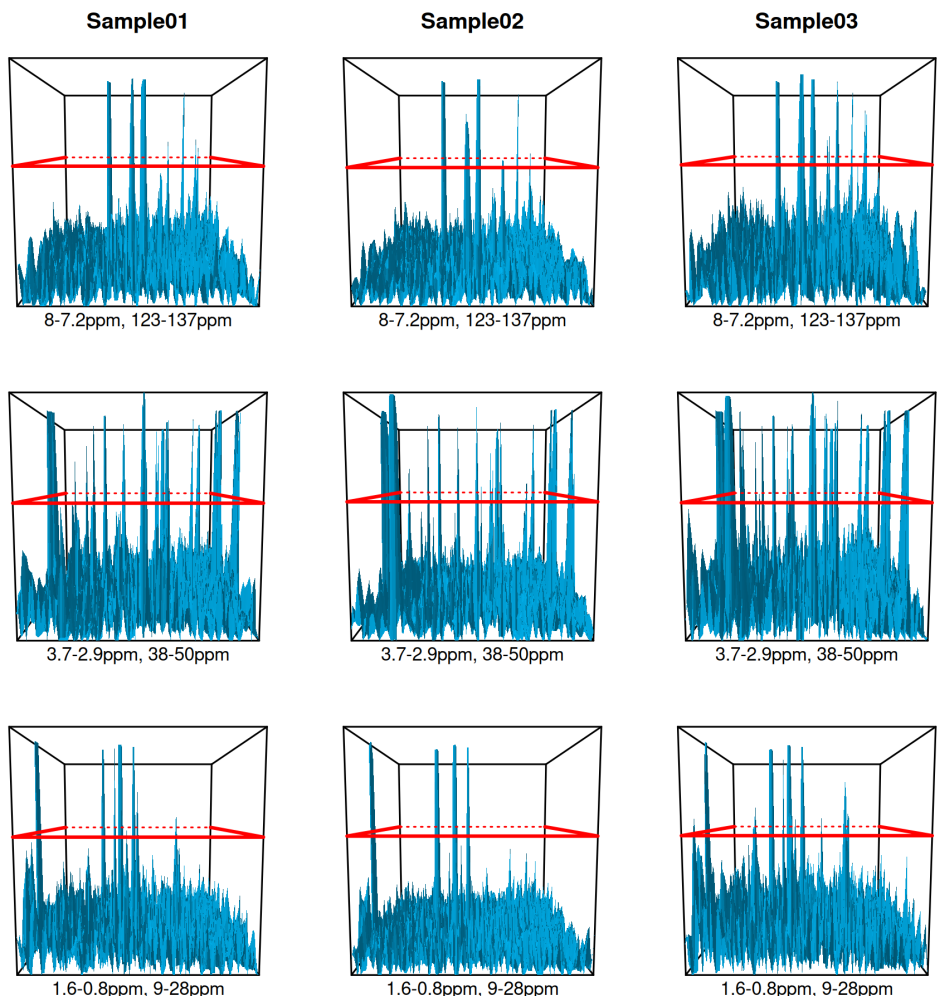
2.7 Example: 2D Data

The following example provides parameters for analyzing a 2D data set.

```
> results2D<-mrbin(setDefault=TRUE,parameters=list(dimension="2D",
+                binwidth2D=0.3,
+                binheight=4,
+                removeSolvent="Yes",
+                solventRegion=c(4.95,4.65),
+                noiseRemoval="Yes",
+                signal_to_noise2D=5,
+                noiseThreshold=0.75,
+                PQNScaling="No",
+                fixNegatives="No",
+                logTrafo="No",
+                useAsNames="Spectrum titles",
+                NMRfolders=c("C:/Bruker/TopSpin/data/guest/nmr/1/10/pdata/10",
+                 "C:/Bruker/TopSpin/data/guest/nmr/2/10/pdata/10",
+                 "C:/Bruker/TopSpin/data/guest/nmr/3/10/pdata/10"),
+                NMRvendor="mrbin"))
```



The approximate signal-to-noise levels are plotted for up to three selected spectra. This plot will also be saved as a pdf file.



The binned NMR data can be found in results$bins. This numeric matrix contains bins in columns and samples in rows, and can be directly used for further data analysis.

The binned data can also be loaded from the hard drive later (if an output file was specified):

```
> load("C:/Users/User/Documents/mrbin.Rdata")
```

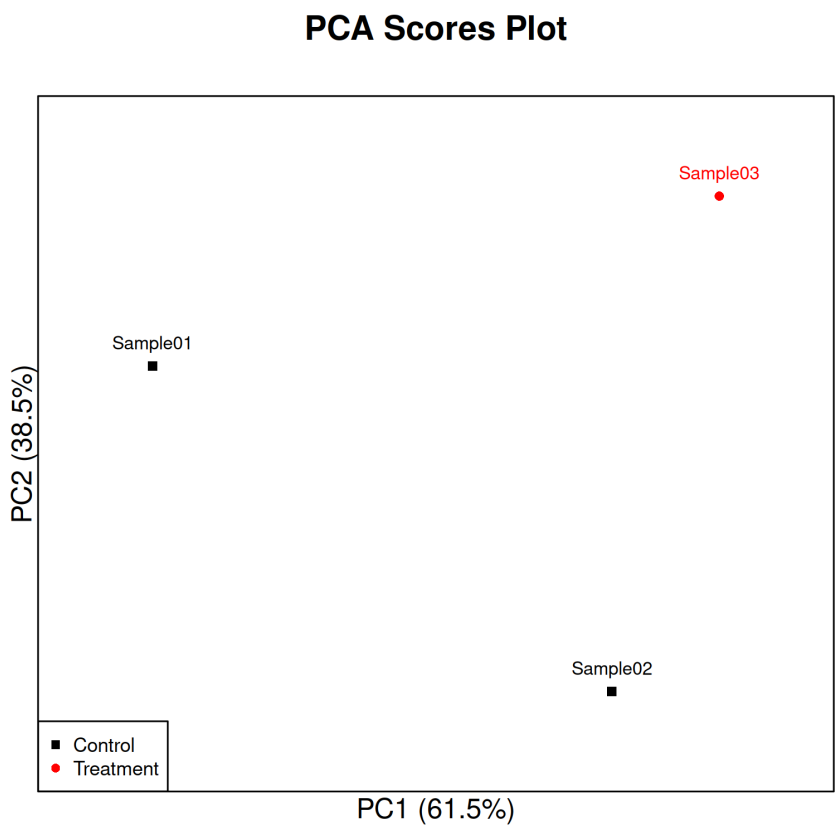
Column names include left and right borders or each bin separated by commas. For 2D data, column names include left, right, top, and bottom border of each bin.

You can edit metadata fields in the command line:

```
> results2D<-metadatamrbin(results2D,metadata=list(
+   projectTitle="Test project",
+   factors=factor(c("Control","Control","Treatment"))))
```

If you define treatment groups in this step, you can plot a PCA with color coded group information as follows:

```
> plotPCA(results2D)
```

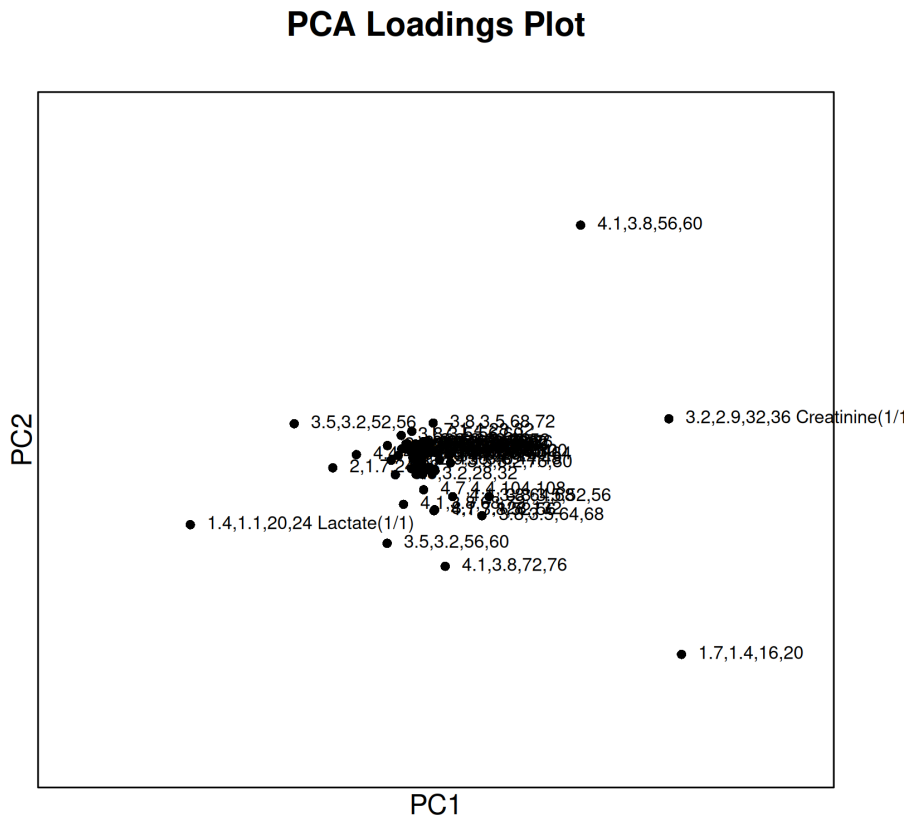


The results can be annotated with potential metabolite identities using the command metadatamrbin(), or in the command line as follows:

```
> #Annotate the dataset with signal identities
> metaboliteIdentities<-matrix(c(1.346,1.324,21,23,
+                                3.052,3.043,30.5,33.5,
+                                5.7,6.0,0,150),
+                     ncol=4,byrow=TRUE)
> rownames(metaboliteIdentities)<-c("Lactate","Creatinine","Urea")
> results2D<-annotatemrbin(results2D,metaboliteIdentities=metaboliteIdentities)
```

The loadings plot of the PCA can be displayed as follows:

```
> plotPCA(results2D,loadings=TRUE,annotate=TRUE)
```



2.8 Example: Lipid Data Analysis

Analyzing lipid NMR signals can be accomplished using custom bin lists rather than a regular grid of bins. This can be done as in the following code:

```
> results <- mrbin(parameters=list(dimension="1D",binMethod="Custom bin list",
+  specialBinList=matrix(c(5.45,5.2,0,160,
+                          2.9,2.74,0,160,
+                          2.14,1.93,0,160,
+                          1.41,1.2,0,160,
+                          0.94,0.8,0,160,
+                          2.44,2.2,0,160,
+                          4.325,4.26,0,160
+                          ),ncol=4,byrow=TRUE,dimnames=list(c(
+                          "-CH=CH- Methene",
+                          "=CH-CH2-CH= Diallylic",
+                          "-CH2-CH=CH- Allylic",
+                          "-CH2- Methylene",
+                          "-CH3 Methyl",
+                          "COO-CH2-CH2- Methylene_to_carboxyl",
+                          "Glycerol"
+                          ),NULL)),
```

```
+  referenceScaling="Yes",reference1D=c(0.03,-0.03),removeSolvent="No",
+  noiseRemoval="No",PQNScaling="No",fixNegatives="Yes",logTrafo="No",
+  NMRfolders=c("C:/Bruker/TopSpin/data/guest/nmr/1/10/pdata/10",
+               "C:/Bruker/TopSpin/data/guest/nmr/2/10/pdata/10",
+               "C:/Bruker/TopSpin/data/guest/nmr/3/10/pdata/10")
+  ))
```

When using custom bin lists, each spectral data point may be part of multiple bins. For rectangular bin lists, each data point will only be counted once. The lipid signal areas used in this example are based on Klein et al, 2011. Please note that the data used in this example does not include lipid signals but signals of small molecules.

2.9 Recreating Data and Parameters

In order to create reproducible results, mrbin will save the used parameters to a text file. Please keep this file. You may want to share this file in a data repository when publishing your findings.

While it is fine to view the parameter text file in a text editor, please do not change its contents, as this may break its formatting.

In order to recreate a previous data set, or to reload previously used parameters, use:

```
> results<-mrbin()
```

and select ""Reload from file" when asked "Set parameters or use existing parameters?". This will restore all parameters that were previously used. If the file was created using an older version of mrbin, this may cause inconsistencies. Missing parameters will be added using standard parameters. Ideally, download the older mrbin version at CRAN and use the old version to recreate the data in an exact way.

Please be aware that bins will have to be recalculated in this case, so the original NMR spectra will have to be present to do this.

2.10 mrbin Workflow

The sequence of data processing is as follows:

- Gathering initial parameters from user
- Creating a set with coordinates of each bin
- Optional: Removing solvent region
- Optional: Removing additional regions
- Optional: Cropping of HSQC spectra to the region along the diagonal
- Optional: Merging regions containing peaks with unstable positions such as citric acid
- Reading NMR spectral data
- Optional: Scaling to reference region
- Binning
- Optional: Removal of bins containing only zeros due to solvent removal (zero trimming)

- Quality control plot, including PCA
- Optional: Removal of bins containing mostly noise
- Optional: Correction for dilution due to differing sample volume or weight
- Optional: PQN transform
- Optional: Replacement of negative values (atnv transform)
- Optional: Log transform
- Optional: Scaling to unit variance
- Quality control plot, including PCA
- Returning final bin data and parameters as an mrbin object

3 atnv: Affine Transformation of Negative Values

The function atnv replaces (column-wise) negative values by a small positive number. The number is calculated as an affine transformation to the range of the lowest positive number to 0,01*the lowest positive number (of this column). Ranks stay unchanged. Positive numbers are not altered.

If sample-wise noise levels are available, the median noise level of samples with negative values is calculated and replaces the lowest positive number in case it is smaller. If no noise data is available, the 1% percentile of all positive values in the data set is used as an estimate.

It is recommended to use this function AFTER noise removal and other data clean-up methods, as it may alter (reduce) the noise level of the binned data. If no NMR data and noise levels are provided as arguments, the function will use NMR data and noise levels from the global variables mrbin.env$bins and mrbin.env$mrbinTMP.

To use own (user provided) data:

```
> atnv(NMRdataMatrix,noiseLevelVector)
```

To use current mrbin data from the internal memory, use atnv()) without parameters. This requires data loaded using mrbin(). This is usually not necessary as it is included in the mrbin work flow.

4 mrplot: NMR Plotting

The mrbin package contains basic plotting commands for 1D and 2D NMR data.

Most convenient is using the mrplot function, which is menu-based:

```
> mrplot()
```

In the mrplot menu, spectra can be added or removed, the view can be zoomed in or out, and intensities can be changed. If multiple spectra are added to a plot, they are shown overlaid. If both 1D and 2D spectra are added, both are shown in region-matched plots, e.g. for metabolite identification purposes.

Other options are as follows:
It is possible and convenient to highlight peaks of interest, this will show a gray line including the chemical shift value in ppm. If two peaks are highlighted, the distance is shown in Hz. This can be used to determine

coupling constants.

You can also provide a list of mrbin bin names for highlighting. For example, this could be the names of significant bins from a statistical analysis. The areas of these bins will be shown in green. This can be used to aid in identifying unknown signals.

The areas of known metabolites can be shown using a peak identification file, or a matrix containing the same information.

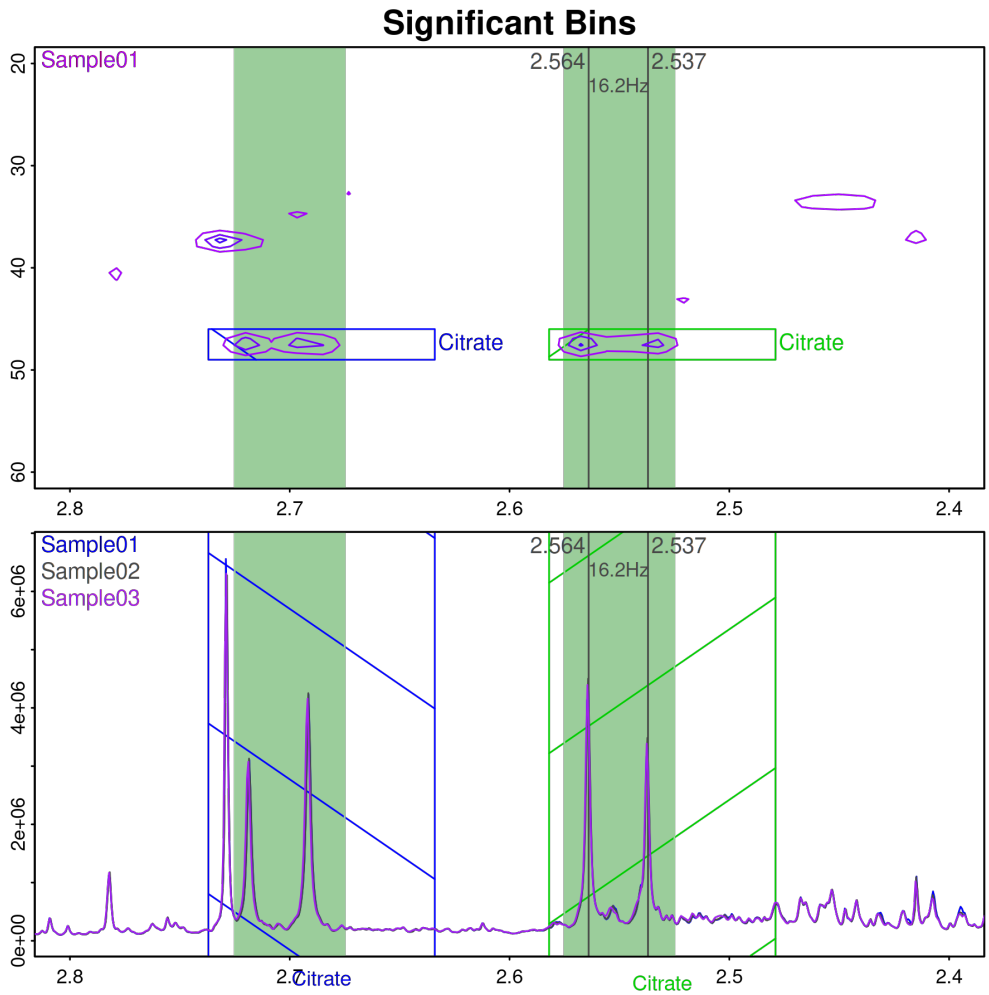
In case of using a file, provide the full file path for a .csv file containing such a matrix, the first columns containing metabolite names, and the first row being a header. Each row belongs to one unique metabolite signal (left, right, top, bottom borders). Row names are metabolite names.

In case of a matrix, this should be a 4-column matrix where each row belongs to one unique metabolite signal (left, right, top, bottom borders). Row names are metabolite names.

Areas of known metabolites are shown in the plot as red shaded boxes. This is useful for identification of unknown peaks.

Parameters and folder names can passed to mrplot on the command line, as follows:

```
> metaboliteIdentities<-matrix(c(1.346,1.324,21,23,
+                               4.12,4.1,70.8578,71.653,
+                               3.052,3.043,30.5,33.5,
+                               4.066,4.059,57,59.5,
+                               2.582,2.479,46,49,
+                               2.737,2.634,46,49),
+                     ncol=4,byrow=TRUE)
> rownames(metaboliteIdentities)<-c("Lactate","Lactate","Creatinine","Creatinine",
+         "Citrate","Citrate")
> mrplot(folders=c(system.file("extdata/1.mr2",package="mrbin"),
+                  system.file("extdata/1.mr1",package="mrbin"),
+                  system.file("extdata/2.mr1",package="mrbin"),
+                  system.file("extdata/3.mr1",package="mrbin")),
+        NMRvendor="mrbin",#default is "Bruker"
+        dimensions=c("2D","1D","1D","1D"),
+            zoom=c(2.8,2.4,20,60),
+        highlight=c(2.564,2.537),
+        binlist=c("2.725,2.675","2.575,2.525"),
+        annotate=TRUE,metaboliteIdentities=metaboliteIdentities,
+        plotTitle="Significant Bins",intensity1D=24,hideMenu=TRUE)
```

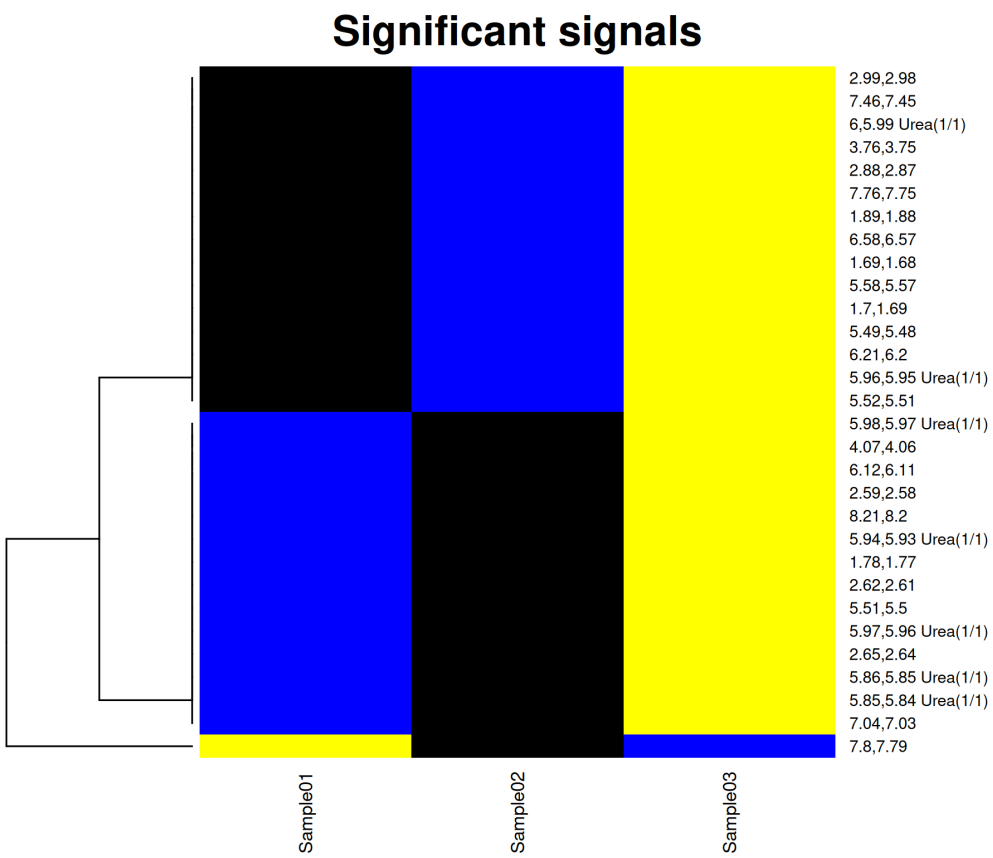


5 Heatmap Plots

Heatmaps are a convenient way to visualize data of significant bins. You can create heatmaps from mrbin objects and a list of significant bins as follows:

```
> # First create NMR bin data, then plot some differential bins.
> results<-mrbin(silent=TRUE,setDefault=TRUE,parameters=list(verbose=FALSE,
+                dimension="1D",binwidth1D=0.01,PCA="No",
+                                showSpectrumPreview="No",
+                signal_to_noise1D=25,noiseThreshold=0.75,
+                                useAsNames="Spectrum titles",
+                NMRvendor="mrbin",#default is "Bruker"
+                NMRfolders=c(
+                system.file("extdata/1.mr1",package="mrbin"),
+                system.file("extdata/2.mr1",package="mrbin"),
+                system.file("extdata/3.mr1",package="mrbin"))
+                ))
> metadata<-c(0,0,1)
> #Find significant signals
> pvalues<-rep(NA,ncol(results$bins))
> names(pvalues)<-colnames(results$bins)
```

```
> for(i in 1:ncol(results$bins)){
+         model<-stats::lm(intensity~treatment,
+      data=data.frame(intensity=results$bins[,i],treatment=metadata))
+         pvalues[i]<-stats::anova(model)$"Pr(>F)"[1]
+ }
> significantBins<-names(sort(pvalues)[1:30])
> #Annotate the dataset with signal identities
> metaboliteIdentities<-matrix(c(1.346,1.324,21,23,
+                                3.052,3.043,30.5,33.5,
+                                5.7,6.0,0,150),
+                     ncol=4,byrow=TRUE)
> rownames(metaboliteIdentities)<-c("Lactate","Creatinine","Urea")
> results<-annotatemrbin(results,metaboliteIdentities=metaboliteIdentities)
> mrheatmap(results=results,
+     binlist=significantBins,annotate=TRUE,
+     main="Significant signals",closeDevice=FALSE)
```



6 Basic Plotting Commands

Instead of using the convenient mrplot() function, basic NMR plotting commands can be accessed from the command line. In most cases, it is preferable to use the mrplot command as detailed above.

To use the basic plotting commands, you need to first load one NMR spectrum:

```
> readBruker(dimension="1D",folder="C:/Bruker/TopSpin/data/guest/nmr/1/10/pdata/10")
> plotNMR()
```

The system.file command loads example data from the mrbin package. To use your own spectra, please adjust as follows:

```
> addToPlot(dimension="1D",folder="C:/Bruker/TopSpin/data/guest/nmr/1/10/pdata/10")
> plotNMR()
```

For 2D data, this code reads as follows:

```
> addToPlot(dimension="2D",NMRvendor="mrbin",
+   folder="C:/Bruker/TopSpin/data/guest/nmr/1/12/pdata/10")
> plotNMR()
```

There are multiple commands for editing the plot:

```
> zoom(left=4.6, right=2, top=10, bottom=150) #Exact zoom
> zoomIn() #Zoom in
> zoomOut() #Zoom out
> intPlus() #Increase intensity
> intMin() #Decrease intensity
> left() #Move spectrum to the left
> right() #Move spectrum to the right
```

For 2D data, you can additionally use the following commands:

```
> contMin() #Decrease minimum contour level (show more small peaks)
> contPlus() #Increase minimum contour level (remove small peaks)
> up() #Move spectrum up
> down() #Move spectrum down
```

7 fia: Feature Impact Assessment of Artificial Neural Networks

The function fia() finds features that can change the outcomes of a model's prediction. For example, fia=1.00 means single compound found in all, or 100 percent of samples. fia=2.45 would indicate that this compound is found in pairs in all but 45 percent of tested samples.
You will need a trained Artificial Neural Network (ANN) model available in R to use fia(), e.g. using packages keras and/or tensorflow. A function named predict needs to be present for fia() to work. If the function name of the prediction function is different, the function name has to be provided in the parameter functionNamePredict. Please make sure to have loaded all required packages before starting fia().

As an example, we will use a logit model instead of ANN, as ANN training would require additional programs to be installed.

```
> #First, define group membership and create the example feature data
> group<-factor(c(rep("Group1",4),rep("Group2",5)))
> names(group)<-paste("Sample",1:9,sep="")
> dataset<-data.frame(
```

```
+     Feature1=c(5.1,5.0,6.0,2.9,4.8,4.6,4.9,3.8,5.1),
+     Feature2=c(2.6,4.0,3.2,1.2,3.1,2.1,4.5,6.1,1.3),
+     Feature3=c(3.1,6.1,5.8,5.1,3.8,6.1,3.4,4.0,4.4),
+     Feature4=c(5.3,5.2,3.1,2.7,3.2,2.8,5.9,5.8,3.1),
+     Feature5=c(3.2,4.4,4.8,4.9,6.0,3.6,6.1,3.9,3.5),
+     Feature6=c(6.8,6.7,7.2,7.0,7.3,7.1,7.2,6.9,6.8)
+ )
> rownames(dataset)<-names(group)
> #train the logit model
> mod<-glm(group~Feature1+Feature2+Feature3+Feature4+Feature5+Feature6,
+    data=data.frame(group=group,dataset),family="binomial")
> fiaresults<-fia(model=mod,dataSet=dataset,factors=group,
+   parameterNameData="newdata",firstLevel=0,type="response")
> fiaresults$scores
```

8 Known Issues

8.1 Linux tcl Issues

In rare cases, the following error (or similar) might occur while navigating pop-up selections in mrbin or mrplot:

tcl grab failed: window not viewable.

This issue seems to be related to tcl handling of select.list pop-up windows. It can potentially be avoided by avoiding double-clicking inside the pop-up windows, instead selecting the desired option with a single click and then clicking OK. To stop the error message from occurring, try to close all plot windows and pop-up windows. It might be helpful to create a new plot window, e.g. by example(mrbin), and then closing the new plot window. To avoid pop-up windows altogether, use mrbin(graphics=FALSE).

8.2 Pop-up Windows Missing on Apple Computers, RStudio

In some cases, running mrbin on Apple computers and/or within RStudio will not generate pop-up windows. To enable pop-up windows, it might be helpful to install the newest version of xquartz from https://www.xquartz.org.

8.3 Pop-Up Windows

mrbin is set up to ask for user input through pop-up windows. This requires graphics support. If graphics support is not enabled on your system, user input will be asked through command line prompts, which offers the full functionality.

8.4 Firewall Warnings

mrbin will try to use the socket approach for parallel computing. This requires establishing network connections to the local cluster, which might trigger the firewall. It is safe to unblock these connections.

9 License

This project is licensed under GPL-3.0.

10 Citation

If you are using mrbin in a publication, please cite the following manuscript:

Klein, M.S. (2021): Affine Transformation of Negative Values for NMR Metabolomics Using the mrbin R Package. J. Proteome Res. 20(2):1397-1404, DOI: 10.1021/acs.jproteome.0c00684

11 References

Klein MS, Dorn C, Saugspier M, Hellerbrand C, Oefner PJ & Gronwald W (2011): Discrimination of Steatosis and NASH in Mice Using Nuclear Magnetic Resonance Spectroscopy. *Metabolomics* 7:237-246