Rnmr1D package

Daniel Jacob

2018-08-06

Contents

Preamble

A development version of the `Rnmr1D` package can be downloaded from GitHub. To install it, we can follow the indications in the `README.md` file.

Features illustration of the Rnmr1D package

Rnmr1D is the main module in the NMRProcFlow web application (nmrprocflow.org)[1] concerning the NMR spectra processing.

Description

Rnmr1D R package is aimed to performs the complete processing of a set of 1D NMR spectra from the FID (raw data) and based on a processing sequence (macro-command file). An additional file specifies all the

spectra to be considered by associating their sample code as well as the levels of experimental factors to which they belong.

NMRProcFlow allows experts to build their own spectra processing workflow, in order to become re-applicable to similar NMR spectra sets, i.e. stated as use-cases. By extension, the implementation of NMR spectra processing workflows executed in batch mode can be considered as relevant provided that we want to process in this way very well-mastered and very reproducible use cases, i.e. by applying the same Standard Operating Procedures (SOP). A subset of NMR spectra is firstly processed in interactive mode in order to build a well-suited workflow. This mode can be considered as the 'expert mode'. Then, other subsets that are regarded as either similar or being included in the same case study, can be processed in batch mode, operating directly within a R session.

See the NMRProcFlow online documentation https://nmrprocflow.org/ for further information.

Test with the extra data provided within the package

To illustrate the possibilities of Rnmr1D 1.0, we will use the dataset provided within the package. This is a very small set of 1H NMR spectra (6 samples) acquired on a Bruker Advanced III 500Mz instrument (ZG sequence, solvent D20, pH 6), derived from plant leaves. The experimental design of the study focused on a treatment (stress vs. control) with 3 replicates for each samples (unpublished).

```
library(Rnmr1D)
data_dir <- system.file("extra", package = "Rnmr1D")
RAWDIR <- file.path(data_dir, "CD_BBI_16P02")
CMDFILE <- file.path(data_dir, "NP_macro_cmd.txt")
SAMPLEFILE <- file.path(data_dir, "Samples.txt")
```

The samples matrix with the correspondence of the raw spectra, as well as the levels of the experimental factors

```
samples <- read.table(SAMPLEFILE, sep="\t", header=T,stringsAsFactors=FALSE)
samples
```

```
##          Spectrum Samplecode EXPNO PROCNO Treatment
## 1 CD_BBI_16P02-R1         R1    10      1   control
## 2 CD_BBI_16P02-R2         R2    10      1   control
## 3 CD_BBI_16P02-R3         R3    10      1   control
## 4 CD_BBI_16P02-R7         R7    10      1    stress
## 5 CD_BBI_16P02-R8         R8    10      1    stress
## 6 CD_BBI_16P02-R9         R9    10      1    stress
```

The Macro-commands list for processing

```
CMDTEXT <- readLines(CMDFILE)
CMDTEXT[grep("^#$", CMDTEXT, invert=TRUE)]
```

```
##  [1] "#%% Vendor=bruker; Type=fid; LB=0.3; GB=0; ZF=2; BLPHC=FALSE; PHC1=TRUE; FP=0; TSP=TRUE"
##  [2] ""
##  [3] "# Baseline Correction: PPM Range = (  4.966  ,  9.348  )"
##  [4] "airpls 4.966 9.348 3 "
##  [5] ""
##  [6] "# Baseline Correction: PPM Range = (  0.396  ,  4.712  )"
##  [7] "airpls 0.396 4.712 4 "
##  [8] ""
##  [9] "# Baseline Correction: PPM Range = (  0.621  ,  1.522  )"
## [10] "airpls 0.621 1.522 5 "
## [11] ""
```

```
## [12] "# Baseline Correction: PPM Range = (  1.227  ,  1.353  )"
## [13] "airpls 1.227 1.353 6"
## [14] ""
## [15] "# Normalisation ( CSN ) of the Intensities based on the selected PPM ranges..."
## [16] "normalisation CSN"
## [17] "0.98 1.085"
## [18] "5.024 9.282"
## [19] "1.451 4.696"
## [20] "EOL"
## [21] ""
## [22] "# Zeroing the selected zones ..."
## [23] "zero"
## [24] "4.683 5.015"
## [25] "EOL"
## [26] ""
## [27] "# Alignment of the selected zones ( 5.024 , 9.611 )"
## [28] "clupa 10.2 10.5 5.024 9.611 0.01 5 0"
## [29] ""
## [30] "# Alignment of the selected zones ( 2.652 , 2.742 )"
## [31] "align 2.652 2.742 0.05 0"
## [32] ""
## [33] "# Alignment of the selected zones ( 2.651 , 2.677 )"
## [34] "align 2.651 2.677 0.05 0"
## [35] ""
## [36] "# Alignment of the selected zones ( 0.642 , 4.699 )"
## [37] "clupa 10.2 10.5 0.642 4.699 0.01 5 0"
## [38] ""
## [39] "# Bucketing - UNIFORM"
## [40] "bucket unif 10.2 10.5 0.01 3 0"
## [41] "4.696 0.98"
## [42] "9.389 5.008"
## [43] "EOL"
```

Do the processing

doProcessing is the main function of this package. Indeed, this function performs the complete processing of a set of 1D NMR spectra from the FID (raw data) and based on a processing sequence (macro-command file). An additional file specifies all the spectra to be considered by associating their sample code as well as the levels of experimental factors to which they belong. In this way it is possible to select only a subset of spectra instead of the whole set.

```
out <- Rnmr1D::doProcessing(RAWDIR, cmdfile=CMDFILE, samplefile=SAMPLEFILE, ncpu=2)
```

```
## Rnmr1D:  --- READING and CONVERTING ---
## Rnmr1D:  Vendor=bruker, Type=fid, LB=0.3, GB=0, ZF=2, BLPHC=FALSE, PHC1=TRUE, FP=0, TSP=TRUE
## Rnmr1D:  Generate the 'samples' & 'factors' files from the list of raw spectra
## Rnmr1D:  -- Nb Spectra = 6 -- Nb Cores = 2
##
## Rnmr1D:  Generate the final matrix of spectra...
## Rnmr1D: ------------------------------------
## Rnmr1D: Process the Macro-commands file
## Rnmr1D: ------------------------------------
## Rnmr1D:
## Rnmr1D:  Baseline Correction: PPM Range = ( 4.966 , 9.348 )
```

```
## Rnmr1D:     Type=airPLS, lambda= 3
## Rnmr1D:   Baseline Correction: PPM Range = ( 0.396 , 4.712 )
## Rnmr1D:     Type=airPLS, lambda= 4
## Rnmr1D:   Baseline Correction: PPM Range = ( 0.621 , 1.522 )
## Rnmr1D:     Type=airPLS, lambda= 5
## Rnmr1D:   Baseline Correction: PPM Range = ( 1.227 , 1.353 )
## Rnmr1D:     Type=airPLS, lambda= 6
## Rnmr1D:   Normalisation of the Intensities based on the selected PPM ranges...
## Rnmr1D:     Method =CSN
## Rnmr1D:   Zeroing the selected PPM ranges ...
## Rnmr1D:     Zone 1 = ( 4.683 , 5.015 )
## Rnmr1D:   Alignment: PPM Range = ( 5.024 , 9.611 )
## Rnmr1D:     CluPA - Resolution =0.01 - SNR threshold=5 - Reference=0
## Rnmr1D:     --- Peak detection : nDivRange = 64
## Rnmr1D:     --- Peak detection time:  5.64  sec
## Rnmr1D:     --- The reference spectrum is:  3
## Rnmr1D:     --- Spectra alignment to the reference: maxShift = 16
## Rnmr1D:     --- Spectra alignment time:  0.939999999999998  sec
## Rnmr1D:   Alignment: PPM Range = ( 2.652 , 2.742 )
## Rnmr1D:     Rel. Shift Max.=0.05 - Reference=0
## Rnmr1D:   Alignment: PPM Range = ( 2.651 , 2.677 )
## Rnmr1D:     Rel. Shift Max.=0.05 - Reference=0
## Rnmr1D:   Alignment: PPM Range = ( 0.642 , 4.699 )
## Rnmr1D:     CluPA - Resolution =0.01 - SNR threshold=5 - Reference=0
## Rnmr1D:     --- Peak detection : nDivRange = 64
## Rnmr1D:     --- Peak detection time:  4.48  sec
## Rnmr1D:     --- The reference spectrum is:  2
## Rnmr1D:     --- Spectra alignment to the reference: maxShift = 16
## Rnmr1D:     --- Spectra alignment time:  1.05  sec
## Rnmr1D:   Bucketing the selected PPM ranges ...
## Rnmr1D:     UNIF - Resolution =0.01 - SNR threshold=3 - Append=0
## Rnmr1D:     Zone 1 = ( 0.98 , 4.696 ), Nb Buckets = 343
## Rnmr1D:     Zone 2 = ( 5.008 , 9.389 ), Nb Buckets = 265
## Rnmr1D:     Total Buckets = 608
```

The ouput list includes severals metadata, data and other information.

```
ls(out)
```

```
## [1] "factors" "infos"   "nuc"     "origin"  "rawids"  "samples" "specMat"
```

```
out$infos
```

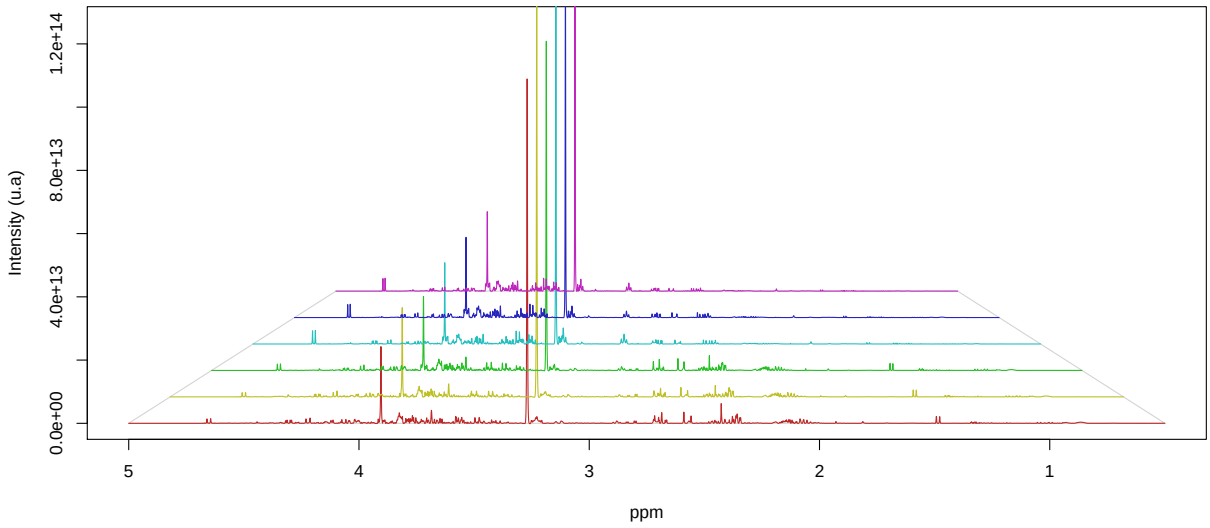
```
##      Spectrum          Samplecode EXPNO PROCNO PULSE NUC  SOLVENT   GRPDLY
## [1,] "CD_BBI_16P02-R1" "R1"       "10"  "0"    "zg"  "1H" "H2O+D2O" "76"
## [2,] "CD_BBI_16P02-R2" "R2"       "10"  "0"    "zg"  "1H" "H2O+D2O" "76"
## [3,] "CD_BBI_16P02-R3" "R3"       "10"  "0"    "zg"  "1H" "H2O+D2O" "76"
## [4,] "CD_BBI_16P02-R7" "R7"       "10"  "0"    "zg"  "1H" "H2O+D2O" "76"
## [5,] "CD_BBI_16P02-R8" "R8"       "10"  "0"    "zg"  "1H" "H2O+D2O" "76"
## [6,] "CD_BBI_16P02-R9" "R9"       "10"  "0"    "zg"  "1H" "H2O+D2O" "76"
##      PHC0               PHC1                    SF            SI
## [1,] "5.83856503237649" "-0.0339284268138111"  "500.1625008" "65536"
## [2,] "5.86627284508418" "-0.0445959039482346"  "500.1625008" "65536"
## [3,] "5.77833759576259" "0.0105653710605261"   "500.1625008" "65536"
## [4,] "5.81872078864614" "-0.00636448174775571" "500.1625008" "65536"
## [5,] "5.83483933104464" "-0.0299512822062665"  "500.1625008" "65536"
```

```
## [6,] "5.78002134991747" "0.0156080974316673"   "500.1625008" "65536"
##      SW                 SWH                 RELAXDELAY O1
## [1,] "12.0009016085441" "6002.40096038415" "25"       "2500.8"
## [2,] "12.0009016085441" "6002.40096038415" "25"       "2500.8"
## [3,] "12.0009016085441" "6002.40096038415" "25"       "2500.8"
## [4,] "12.0009016085441" "6002.40096038415" "25"       "2500.8"
## [5,] "12.0009016085441" "6002.40096038415" "25"       "2500.8"
## [6,] "12.0009016085441" "6002.40096038415" "25"       "2500.8"
```

Spectra plots

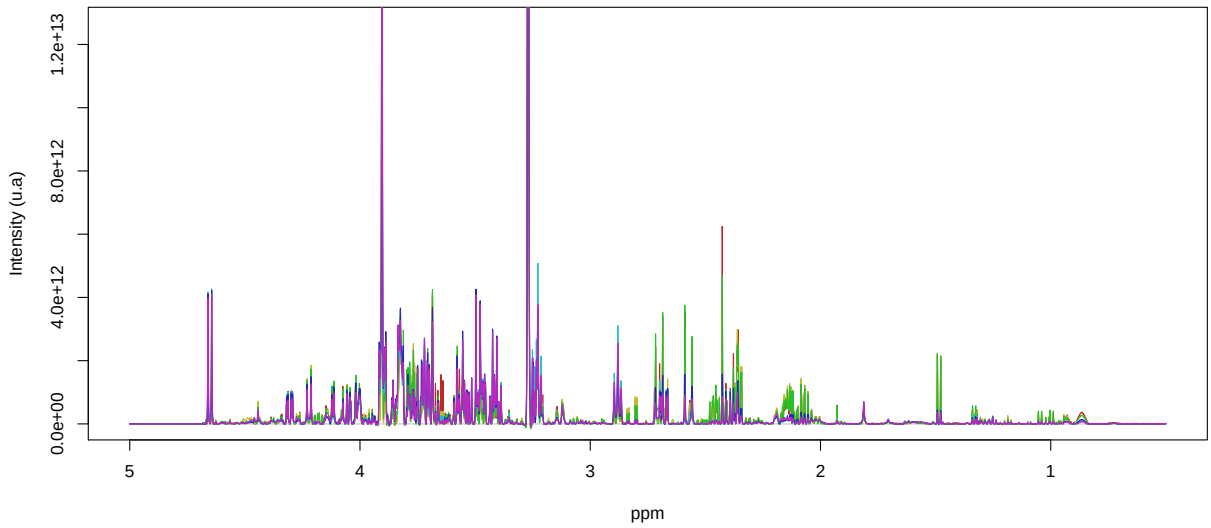
Stacked Plot with a perspective effect

```
plotSpecMat(out$specMat, ppm_lim=c(0.5,5), K=0.33)
```



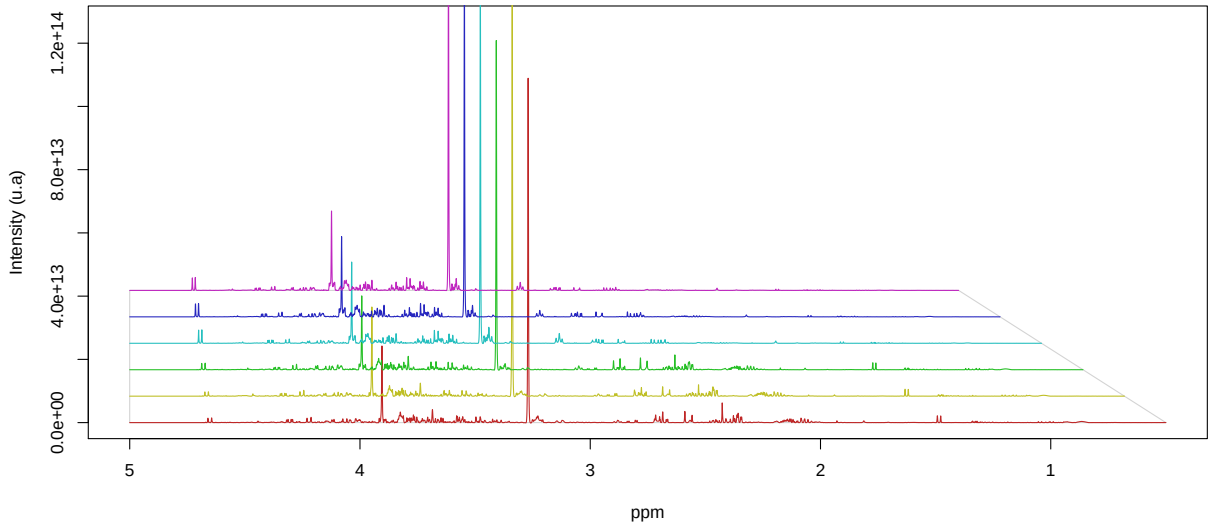
Overlaid Plot

```
plotSpecMat(out$specMat, ppm_lim=c(0.5,5), K=0, pY=0.1)
```

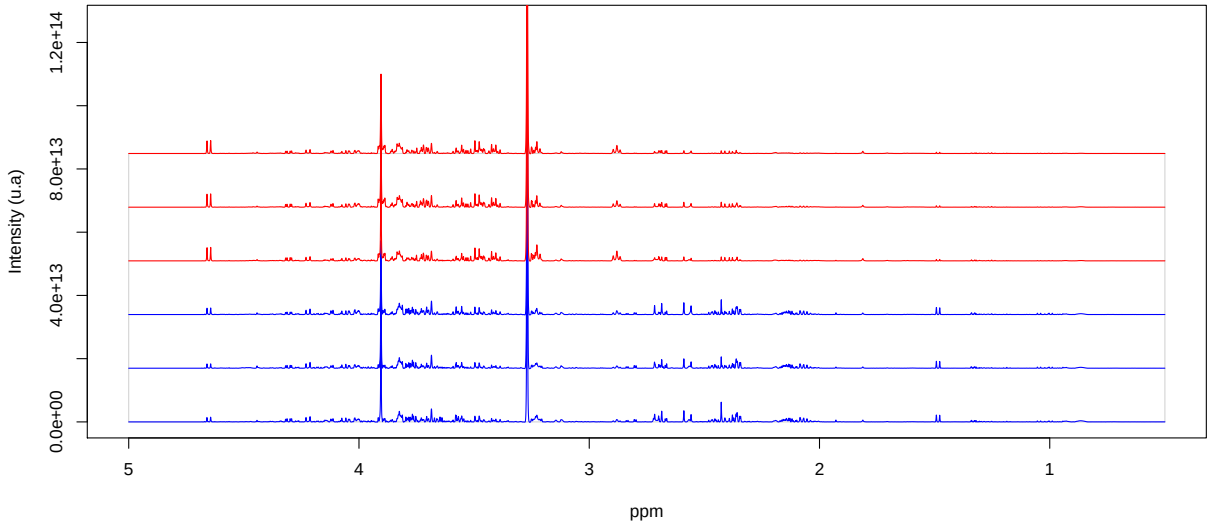


Some other plots to illustrate the possibilities

```
plotSpecMat(out$specMat, ppm_lim=c(0.5,5), K=0.33, asym=0)
```




```
cols<- c( rep("blue",length(out$samples$Treatment)));
cols[out$samples$Treatment=="stress"] <- "red"
plotSpecMat(out$specMat, ppm_lim=c(0.5,5), K=0.67, dppm_max=0, cols=cols)
```



Apply additional processing

It is possible to apply additionnal processing after the main processing. For that the doProcCmd function can process macro-commands included in a string array to be applied on the spectra set previously generated. In the previous processing, the bucketing has been done with a uniform approach. We are going to change this by an intelligent bucketing [1], more efficient to generate relevant variables as we will further see .

```
specMat.new <- Rnmr1D::doProcCmd(out,
     c( "bucket aibin 10.2 10.5 0.3 3 0", "9.5 4.9", "4.8 0.5", "EOL" ), ncpu=2, debug=TRUE)
```

```
## Rnmr1D:  Bucketing the selected PPM ranges ...
## Rnmr1D:     AIBIN - Resolution =0.3 - SNR threshold=3 - Append=0
## Rnmr1D:     Zone 1 = ( 4.9 , 9.5 ), Nb Buckets = 174
## Rnmr1D:     Zone 2 = ( 0.5 , 4.8 ), Nb Buckets = 359
## Rnmr1D:     Total Buckets = 533
```

```
out$specMat <- specMat.new
```

Get the data matrix

Before exporting, in order to make all spectra comparable each other, we have to account for variations of the overall concentrations of samples. In NMR metabolomics, the total intensity normalization (called the Constant Sum Normalization) is often used so that all spectra correspond to the same overall concentration. It simply consists to normalize the total intensity of each individual spectrum to a same value.

```
outMat <- Rnmr1D::getBucketsDataset(out, norm_meth='CSN')
outMat[, 1:10]
```

```
##      B0_8433  B0_8942   B0_9032  B0_9110   B0_9319   B0_9396  B0_9486
## R1 26.629385 790.7420 13.781555 4.880644 199.96601 101.13499 75.13242
## R2 19.703300 587.7658  8.004962 1.704992 145.51685  85.59131 82.86045
## R3 17.568801 573.3763  8.006654 1.069050 117.69809  70.83956 66.31759
## R7  2.281647 191.6645  5.302362 1.095003  62.92991  38.99682 22.90221
## R8  5.138960 277.1071  5.143850 1.760685  58.42666  36.56559 25.19425
```

```
## R9  1.330692 124.4668  5.744872 2.177681  47.73960  28.26060 20.89296
##      B0_9550   B0_9616   B0_9731
## R1  8.963269 15.014303 23.939061
## R2 12.520129 20.780660 31.644640
## R3 13.475888 21.166506 33.199414
## R7  5.762101  7.243036  7.102918
## R8  5.226604  7.324907  6.729933
## R9  4.780748  8.180084  8.111968
```

Bucket Clustering

Note: The following features are integrated into the BioStatFlow web application the biostatistical analysis companion of NMRProcFlow (the 'Clustering of Variables' analysis in the default workflow).

At the bucketing step (see above), we have chosen the intelligent bucketing [2], it means that each bucket exact matches with one resonance peak. Thanks to this, the buckets now have a strong chemical meaning, since the resonance peaks are the fingerprints of chemical compounds. However, to assign a chemical compound, several resonance peaks are generally required in 1D 1 H-NMR metabolic profiling. To generate relevant clusters (i.e. clusters possibly matching to chemical compounds), we take advantage of the concentration variability of each compound in a series of samples and based on significant correlations that link these buckets together into clusters. Two approaches have been implemented:

Bucket Clustering based on a lower threshold applied on correlations

In this approach an appropriate correlation threshold is applied on the correlation matrix before its cluster decomposition [3]. Moreover, an improvement can be done by searching for a trade-off on a tolerance interval of the correlation threshold : from a fixed threshold of the correlation (cval), the clustering is calculated for the three values (cval-dC, cval, cval+dC), where dC is the tolerance interval of the correlation threshold. From these three sets of clusters, we establish a merger according to the following rules: 1) if a large cluster is broken, we keep the two resulting clusters. 2) If a small cluster disappears, the initial cluster is conserved. Generally, an interval of the correlation threshold included between 0.002 and 0.01 gives good trade-off.

cval=0 => threshold automatically estimated

```
options(warn=-1)
clustcor <- Rnmr1D::getClusters(outMat, method='corr', cval=0, dC=0.003, ncpu=2)
```

```
## #-- Clustering --
## #  Correlation Method: pearson
## #  Correlation Threshold : 0.997
## #  Correlation Tolerance: 0.003
## #  Nb Clusters: 44
## #
```

Bucket Clustering based on a hierarchical tree of the variables (buckets) generated by an Hierarchical Clustering Analysis (HCA)

In this approach a Hierachical Classification Analysis (HCA, hclust) is applied on the data after calculating a matrix distance ("euclidian" by default). Then, a cut is applied on the tree (cutree) resulting from hclust, into several groups by specifying the cut height(s). For finding best cut value, the cut height is chosen i) by testing several values equally spaced in a given range of the cut height, then, 2) by keeping the one that gives the more cluster and by including most bucket variables. Otherwise, a cut value has to be specified by the user (vcutusr)

vcutusr=0 => cut value automatically estimated

```r
options(warn=-1)
clusthca <- Rnmr1D::getClusters(outMat, method='hca', vcutusr=0)
```

```
## #-- Clustering --
## #  Distance Method: euclidean
## #  Agglomeration Method: complete
## #  Cutting Tree threshold: 0.18
## #  Nb Clusters: 109
## #
```

Clustering results

The getClusters function returns a list containing the several components, in particular:

- **clusters** List of the ppm value corresponding to each cluster. the length of the list equal to number of clusters
- **clustertab** the associations matrix that gives for each cluster (column 2) the corresponding buckets (column 1)

```r
clusthca$clustertab[1:20, ]
```

```
##       [,1]      [,2] [,3]
##  [1,] "B0_8433" "C1" "0.8433"
##  [2,] "B0_8942" "C1" "0.8942"
##  [3,] "B1_2236" "C1" "1.2236"
##  [4,] "B2_1158" "C1" "2.1158"
##  [5,] "B2_2969" "C1" "2.2969"
##  [6,] "B2_3154" "C1" "2.3154"
##  [7,] "B2_3599" "C1" "2.3599"
##  [8,] "B2_3843" "C1" "2.3843"
##  [9,] "B2_8459" "C1" "2.8459"
## [10,] "B3_1309" "C1" "3.1309"
## [11,] "B3_1581" "C1" "3.1581"
## [12,] "B3_7802" "C1" "3.7802"
## [13,] "B4_2638" "C1" "4.2638"
## [14,] "B4_4067" "C1" "4.4067"
## [15,] "B4_4112" "C1" "4.4112"
## [16,] "B4_6735" "C1" "4.6735"
## [17,] "B5_6905" "C1" "5.6905"
## [18,] "B5_9795" "C1" "5.9795"
## [19,] "B5_9938" "C1" "5.9938"
## [20,] "B5_9985" "C1" "5.9985"
```

```r
clusthca$clusters$C5      # same as outclust$clusters[['C5']]
```

```
##  [1] 0.9550 0.9616 0.9731 0.9810 1.0116 1.0279 1.3319 2.1403 2.1550 2.2643
## [11] 2.4449 2.4513 2.4599 2.4685 2.4734 2.4850 2.5150 2.5333 3.0781 4.1719
## [21] 4.1796 4.1913 4.1977 7.2172 7.5567
```

Based on these clusters, it is possible to find candidate by querying online databases - See for example:

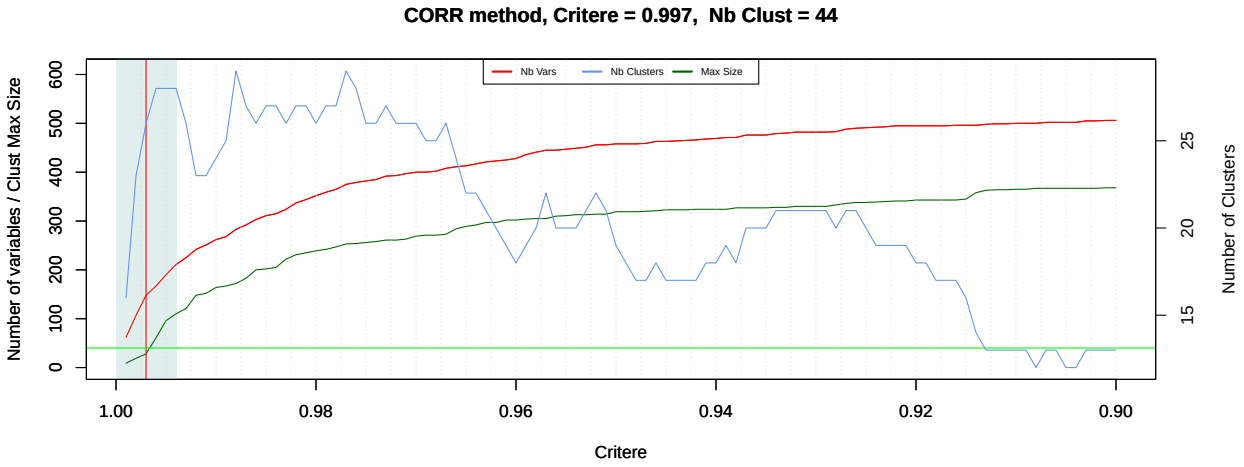
- NMR Peak Matching (DBREF6): http://pmb-bordeaux.fr/PM/webapp

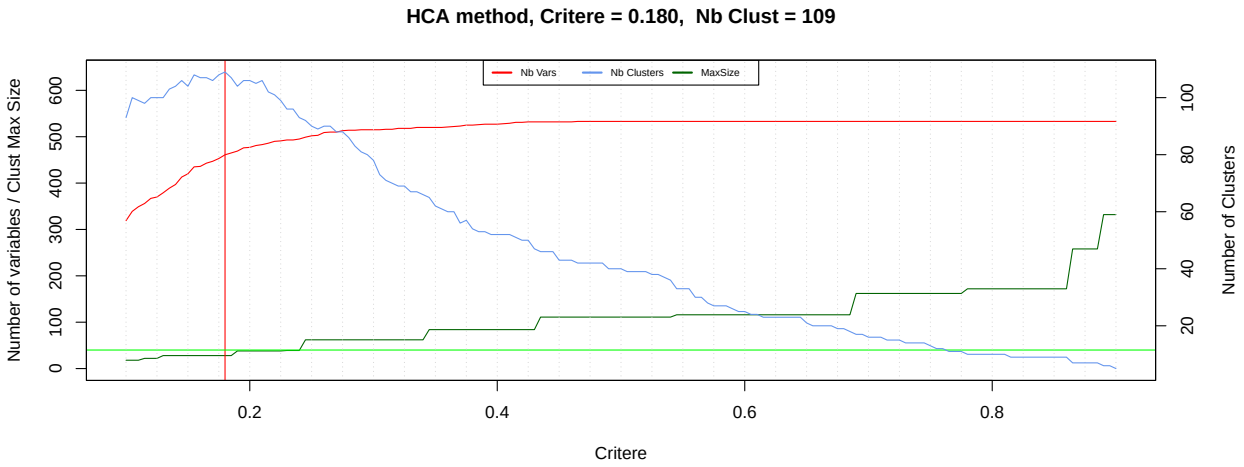
Clustering Comparison resulting of the both 'hca' & 'corr' methods

The'Corr' approach produces fewer clusters but correlations between all buckets are guaranteed to be greater than or equal to the set threshold; unlike the'hca' approach which produces many more clusters and especially small sizes (2 or 3) but which may result from aggregates having a wider range around the threshold of values of correlations between buckets.

Criterion curves

```
layout(matrix(1:2, 2, 1,byrow = TRUE))
plotCriterion(clustcor, reverse=TRUE)
plotCriterion(clusthca)
```



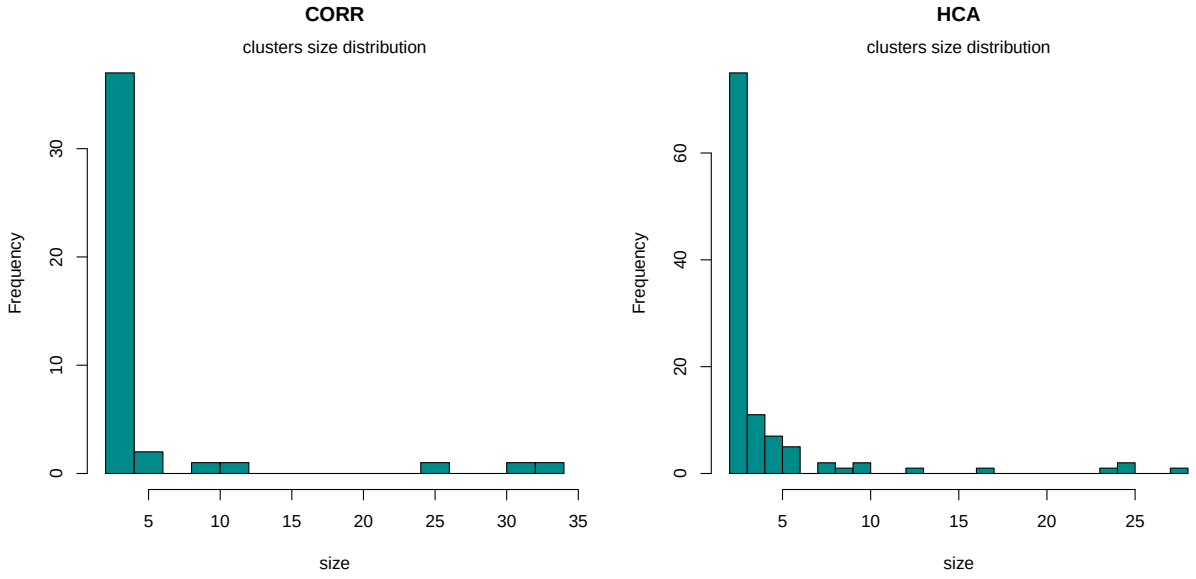


Clusters distribution

These plots allow us to have an insight on the clusters distribution

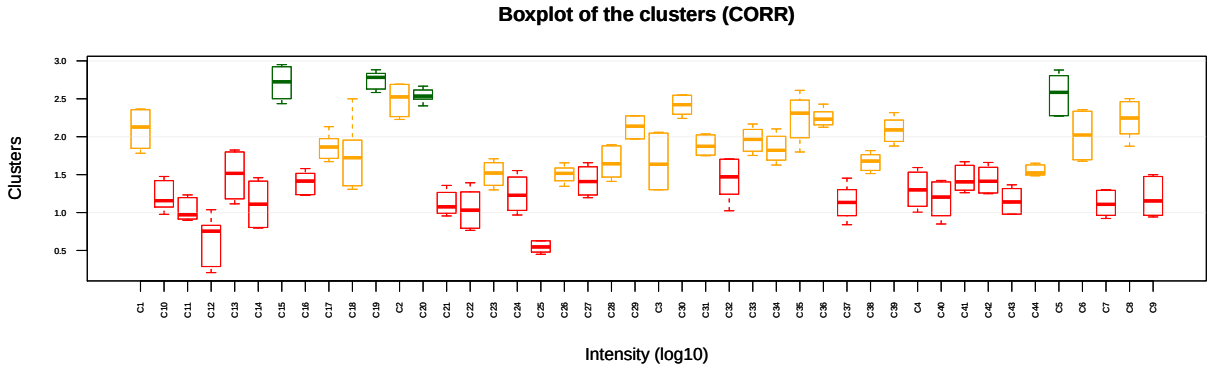
```
layout(matrix(1:2, 1, 2,byrow = TRUE))

hist(simplify2array(lapply(clustcor$clusters, length)),
```

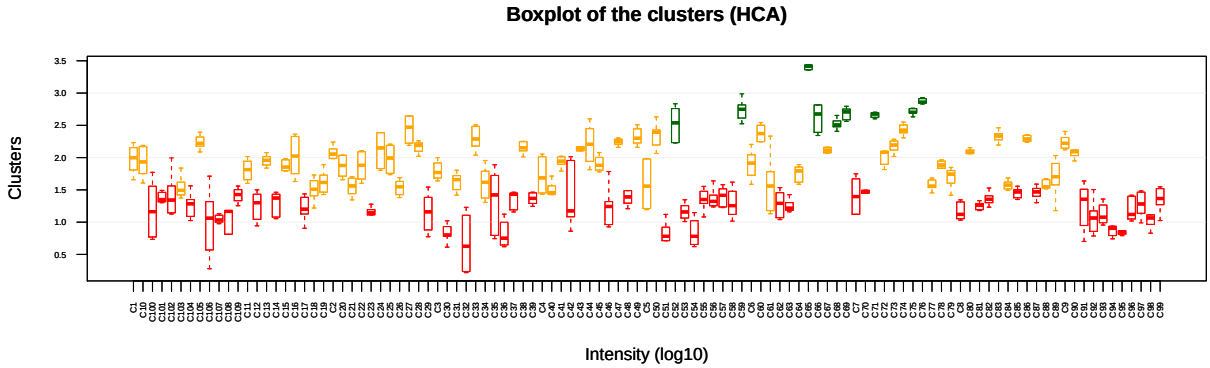
```
     breaks=20, main="CORR", xlab="size", col="darkcyan")
mtext("clusters size distribution", side = 3)

hist(simplify2array(lapply(clusthca$clusters, length)),
     breaks=20, main="HCA", xlab="size", col="darkcyan")
mtext("clusters size distribution", side = 3)
```



We see that they cover three orders (log scaled)

```
layout(matrix(1:2, 2, 1,byrow = TRUE))
plotClusters(outMat,clustcor, horiz=FALSE, main="Boxplot of the clusters (CORR)")
plotClusters(outMat,clusthca, horiz=FALSE, main="Boxplot of the clusters (HCA)")
```





PCA

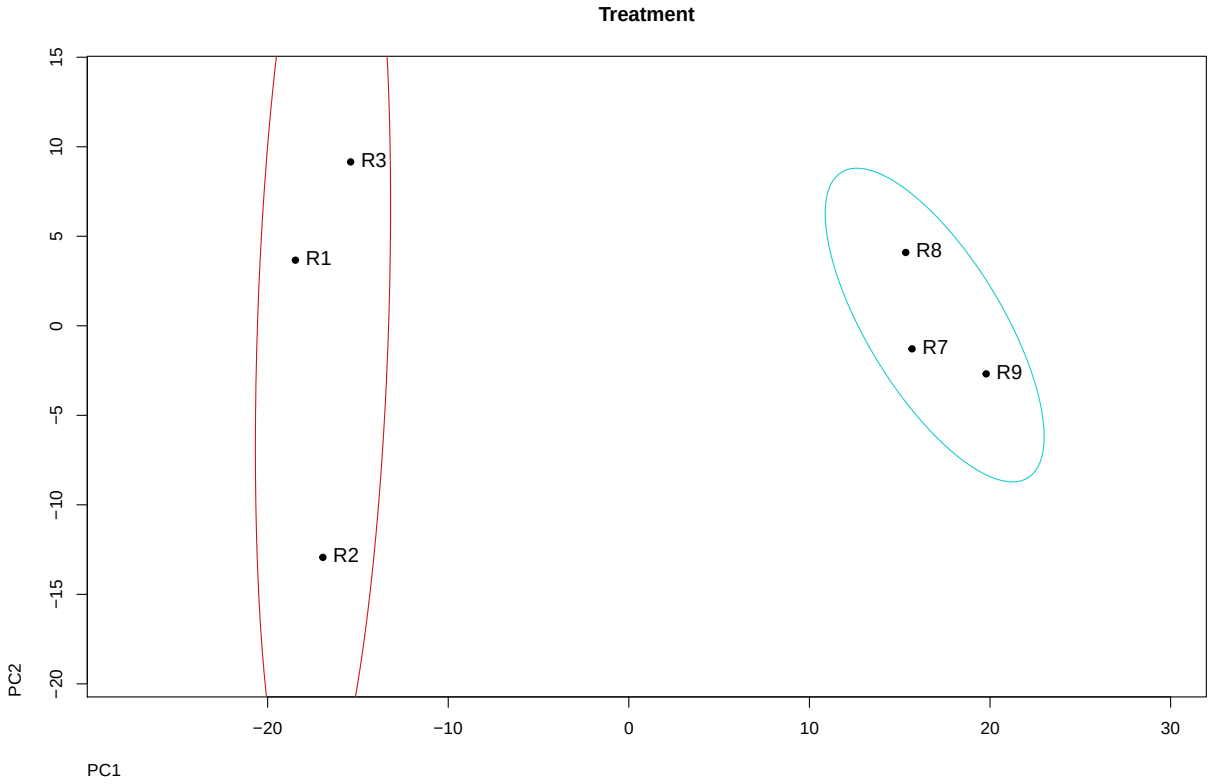
PCA is a multivariate analysis without prior knowledge on the experiment design. In this way, it allows us to see the dataset as it is, projected in the two main components where data variance is maximal.

```
pca <- prcomp(outMat,retx=TRUE,scale=T, rank=2)
```

Plot PCA Scores

Ellipses corresponding to the factors levels are simply added on graph. It is possible to specify their confidence level (95% by default).

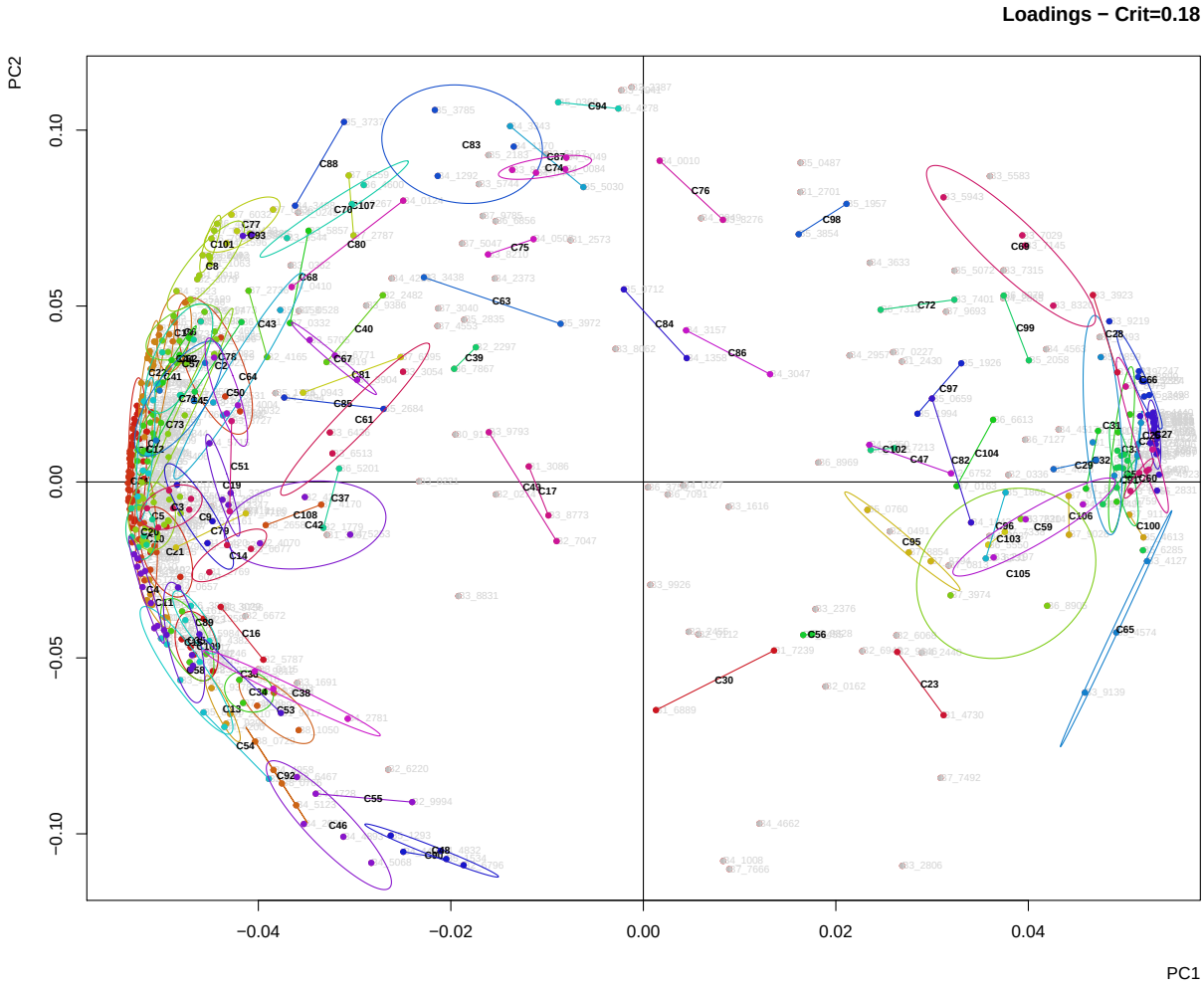
```
# Choose 'Treatment' as factor, confidence level = 95%
plotScores(pca$x, 1, 2, out$samples, factor='Treatment', level=0.95)
```



Plot PCA Loadings

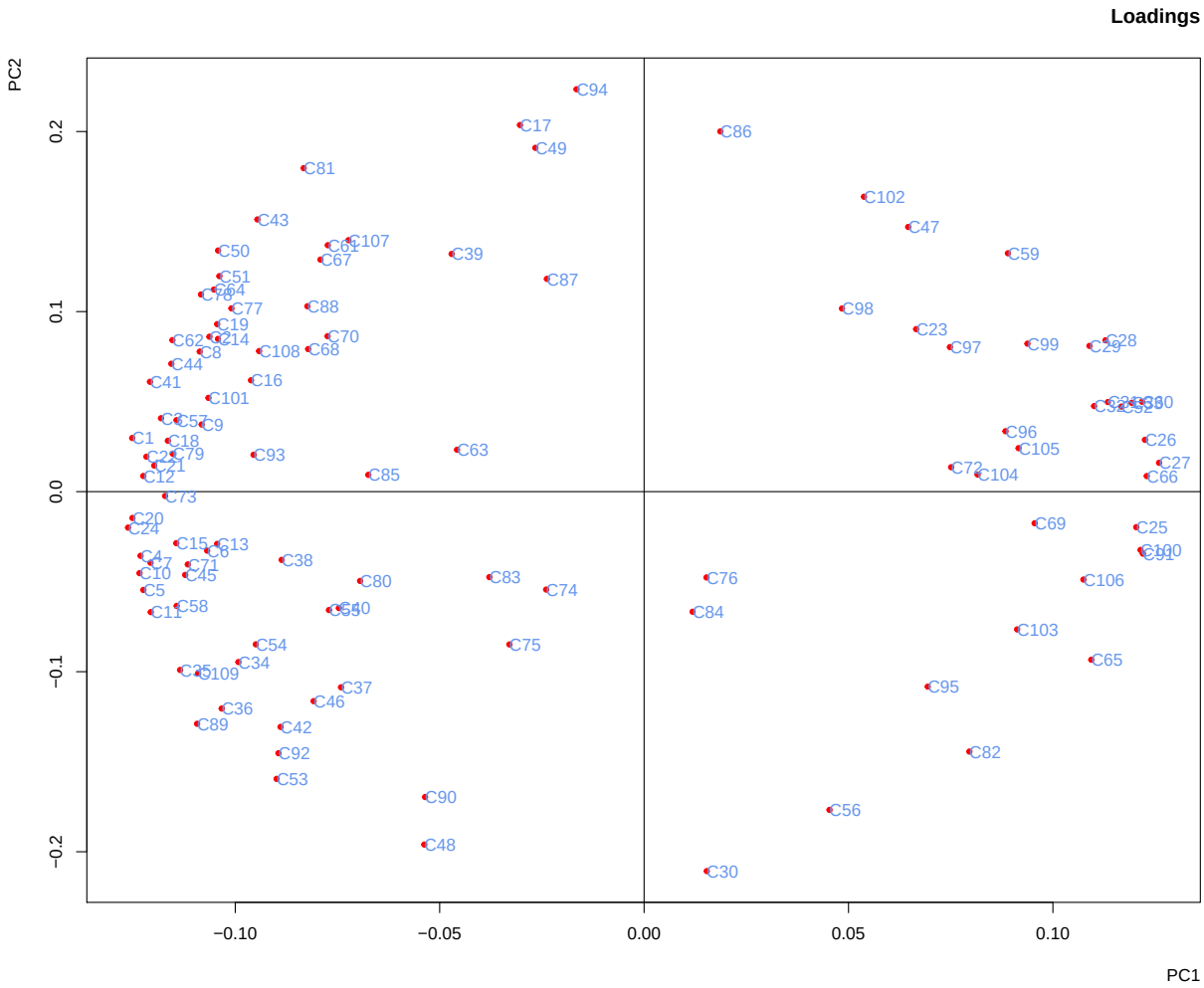
Having calculated the clustering in the previous step (see above), it can be superimposed on the loadings plot in order to view its distribution and thus its efficiency. Here we choose those based on the HCA method. We can see that where the clusters are located, this corresponds to the maximum variance of the buckets, i.e. mainly at the ends of the first principal component (PC1).

```
plotLoadings(pca$rotation, 1, 2, associations=clusthca$clustertab,
             cexlabel=0.6, level=0.8, main=sprintf("Loadings - Crit=%s",clusthca$vcrit) )
```



Plot loadings with the only merged variables for each cluster (based on their average)

```
outMat.merged <- Rnmr1D::getMergedDataset(outMat, clusthca, onlycluster=TRUE)
pca.merged <- prcomp(outMat.merged,retx=TRUE,scale=T, rank=2)
plotLoadings(pca.merged$rotation, 1, 2, associations=NULL, cexlabel=1 )
```



Quod erat demonstrandum !

Low-level functionnalities

Rnmr1D package also provides a set of low-level functions allowing to process spectra one by one. It accepts raw data come from three major vendors namely Bruker GmbH, Agilent Technologies (Varian) and Jeol Ltd. Moreover, we also support the nmrML format [4] (See nmrml.org for further information on this format) Here is an example : Preprocessing one of the raw spectrum taken from the provided data example set. The term pre-processing designates here the transformation of the NMR spectrum from time domain to frequency domain, including the Fast Fourrier Tranform (FFT) and the phase correction.

```
data_dir <- system.file("extra", package = "Rnmr1D")
RAWDIR <- file.path(data_dir, "CD_BBI_16P02")
```

We first initialize the list of preprocessing parameters. The raw spectrum is a FID (Free Induction Decay in the time domain) and was acquired on a Bruker instrument. The processing consistes by applying a line broadening (LB=0.3), then a zerofilling (ZFFAC=4) before applying the FFT (Fast Fourrier Transform). Finally, a phasage has to be applied (Order 0, i.e. OPTPHC0 & Order 1 i.e. OPTPHC1 both equal to TRUE) then a calibration of the ppm scale (TSP=TRUE).

```
procParams <- Spec1rProcpar
procParams$LOGFILE <- ""
procParams$VENDOR <- 'bruker'
procParams$INPUT_SIGNAL <- 'fid'
procParams$LB <- 0.3
procParams$ZEROFILLING <- TRUE
procParams$ZFFAC <- 4
procParams$OPTPHC1 <- TRUE
procParams$TSP <- TRUE
```

We generate the metadata from the list of raw spectra namely the samples and the list of selected raw spectra

```
metadata <- generateMetadata(RAWDIR, procParams)
metadata
```

```
## $samples
##      V11                V11
## [1,] "CD_BBI_16P02-R1" "CD_BBI_16P02-R1"
## [2,] "CD_BBI_16P02-R2" "CD_BBI_16P02-R2"
## [3,] "CD_BBI_16P02-R3" "CD_BBI_16P02-R3"
## [4,] "CD_BBI_16P02-R7" "CD_BBI_16P02-R7"
## [5,] "CD_BBI_16P02-R8" "CD_BBI_16P02-R8"
## [6,] "CD_BBI_16P02-R9" "CD_BBI_16P02-R9"
##
## $rawids
##      [,1]
## [1,] "C:/Users/djaco/Documents/R/win-library/3.5/Rnmr1D/extra/CD_BBI_16P02/CD_BBI_16P02-R1/10"
## [2,] "C:/Users/djaco/Documents/R/win-library/3.5/Rnmr1D/extra/CD_BBI_16P02/CD_BBI_16P02-R2/10"
## [3,] "C:/Users/djaco/Documents/R/win-library/3.5/Rnmr1D/extra/CD_BBI_16P02/CD_BBI_16P02-R3/10"
## [4,] "C:/Users/djaco/Documents/R/win-library/3.5/Rnmr1D/extra/CD_BBI_16P02/CD_BBI_16P02-R7/10"
## [5,] "C:/Users/djaco/Documents/R/win-library/3.5/Rnmr1D/extra/CD_BBI_16P02/CD_BBI_16P02-R8/10"
## [6,] "C:/Users/djaco/Documents/R/win-library/3.5/Rnmr1D/extra/CD_BBI_16P02/CD_BBI_16P02-R9/10"
##      [,2] [,3]
## [1,] "10" "0"
## [2,] "10" "0"
## [3,] "10" "0"
## [4,] "10" "0"
## [5,] "10" "0"
## [6,] "10" "0"
##
## $factors
##      [,1] [,2]
## [1,] "1"  "Samplecode"
```

Spec1rDoProc is the function that can preprocess only one raw spectrum at time. We take the first raw spectrum (FID)

```
ID <- 1
ACQDIR <- metadata$rawids[ID,1]
spec <- Spec1rDoProc(Input=ACQDIR,param=procParams)
```

```
## Read the FID ...OK
## Preprocessing ...
##  Exp. Line Broadening (LB=0.300000)
##  TD = 16384
##  Zero Filling (x4)
```

```
##  SI = 65536
##  Applied GRPDLY ...OK
##  FFT ...OK
## OK
## Optimizing the zero order phase ...OK
## Optimizing the first order phase ...OK
## PPM calibration based on TSP  ... PPM min =-0.923470
## OK
```

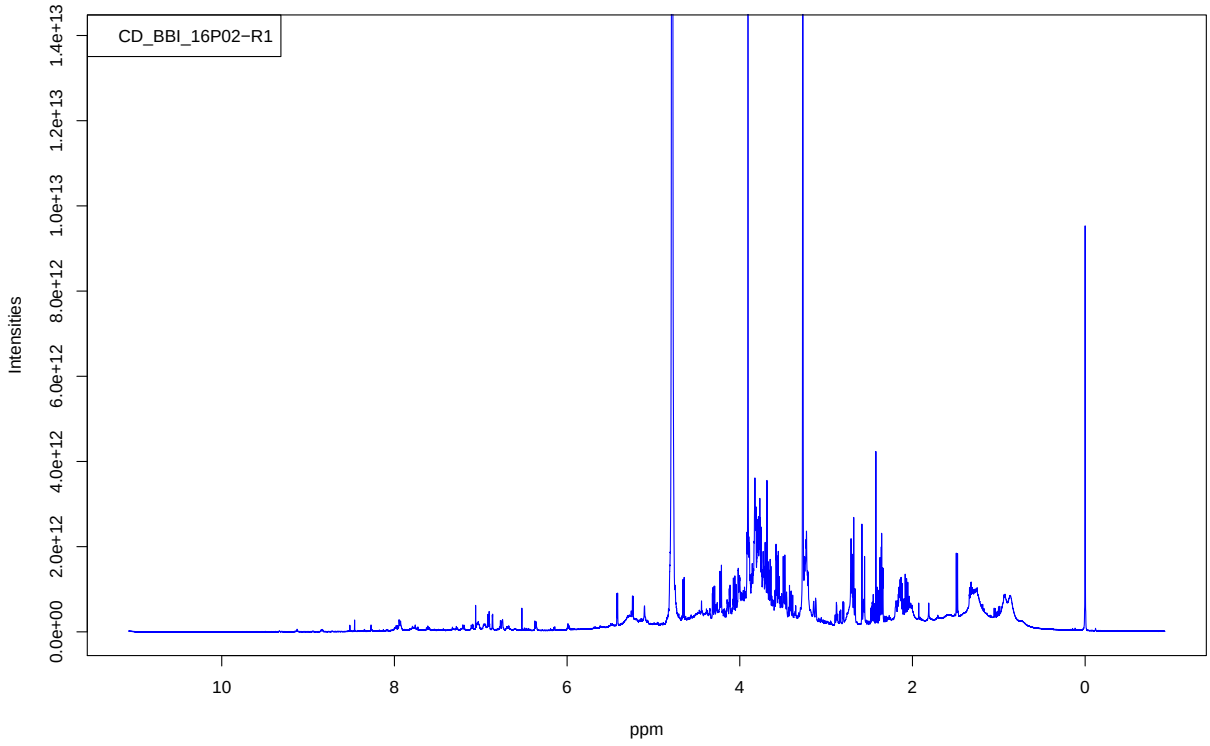
It returns a list containing the following components:

```
ls(spec)
```

```
##  [1] "acq"    "data"   "dppm"   "fid"    "int"    "param"  "path"
##  [8] "pmax"   "pmin"   "ppm"    "proc"   "rawfid"
```

Then, we plot the spectrum in the frequency domain (ppm)

```
plot( spec$ppm, spec$int, type="l", col="blue",
                xlab="ppm", ylab="Intensities",
                xlim=c( spec$pmax, spec$pmin ), ylim=c(0, max(spec$int/100)) )
legend("topleft", legend=metadata$samples[ID,1])
```



You can also see the small application online at the URL https://pmb-bordeaux.fr/nmrspec/.

References

[1] Jacob, D., Deborde, C., Lefebvre, M., Maucourt, M. and Moing, A. (2017) NMRProcFlow: A graphical

and interactive tool dedicated to 1D spectra processing for NMR-based metabolomics, Metabolomics 13:36. https://www.ncbi.nlm.nih.gov/pubmed/28261014

[2] De Meyer, T., Sinnaeve, D., Van Gasse, B., Tsiporkova, E., Rietzschel, E. R., De Buyzere, M. L., et al. (2008). NMR-based characterization of metabolic alterations in hypertension using an adaptive, intelligent binning algorithm. Analytical Chemistry, 80(10), https://www.ncbi.nlm.nih.gov/pubmed/18419139

[3] Jacob D., Deborde C. and Moing A. (2013). An efficient spectra processing method for metabolite identification from 1H-NMR metabolomics data. Analytical and Bioanalytical Chemistry 405(15), https://www.ncbi.nlm.nih.gov/pubmed/23525538

[4] Schober D, Jacob D, Wilson M, Cruz JA, Marcu A, Grant JR, Moing A, Deborde C, de Figueiredo LF, Haug K, Rocca-Serra P, Easton J, Ebbels TMD, Hao J, Ludwig C, Günther UL, Rosato A, Klein MS, Lewis IA, Luchinat C, Jones AR, Grauslys A, Larralde M, Yokochi M, Kobayashi N, Porzel A, Griffin JL, Viant MR, Wishart DS, Steinbeck C, Salek RM, Neumann S. (2018) Anal Chem https://www.ncbi.nlm.nih.gov/pubmed/29035042