MetCirc: Navigating mass spectral similarity in high-resolution MS/MS metabolomics data

Thomas Naake[1] and Emmanuel Gaquerel[2]

[1] Max Planck Institute of Molecular Plant Physiology
14476 Potsdam-Golm, Germany
[2] Plant Defense Metabolism
Centre for Organismal Studies
69120 Heidelberg, Germany

May 2, 2019

1 Introduction

The MetCirc package comprises functionalities to display, (interactively) explore similarities and annotate features of MS/MS metabolomics data. It is especially designed to improve the interactive exploration of metabolomics data obtained from cross-species/cross-tissues comparative experiments. Notably, MetCirc includes functions to calculate the similarity between individual MS/MS spectra based on a normalised dot product (NDP, see Li et al. (2015) for further details) calculation taking into account shared fragments or main neutral losses.

Visualisation of molecular networks was pioneered by the Dorrestein lab (e.g. Watrous et al., 2012) to efficiently organize MS/MS spectra in such a way that clusters of MS/MS spectra sharing multiple common fragments are rapidly inferred.
In contrary to the analysis there, the MetCirc framework offers two ways for the computation of mass spectrum similarity: one way deploys common (shared) fragments and the other shared neutral losses that are deduced from the spectra. Especially the latter approach allows to extract clusters of spectra corresponding to compound families facilitating the rapid dereplication of hitherto unknown metabolites. Small molecules, as produced by plants, return during fragmentation few fragments. Sometimes only a few fragments among them are shared and diagnostic across the compound family. Compared to the MS/MS similarity scoring based on shared fragments, the second similarity measure incorporates, in a non-supervised manner, neutral losses, which definitely helps obtaining biochemically-meaningful MS/MS groupings.

The interpretation drawn from network reconstruction highly depends from topological parameters applied during the network construction steps. MetCirc circumvents this confinement. Furthermore, it uses instead smaller MS/MS datasets obtained from experiments involving *a priori*

defined biological groups (organisms, tissues, etc.) to visualise within and between MS/MS feature similarities on a circular layout - inspired by the Circos framework (Krzywinski et al., 2009). The visualisation is adjustable (MS/MS ordering and similarity thresholds) via the shiny interface and does not uniquely emphasize on large clusters of MS/MS features (a frequent caveat of network visualisation). Furthermore, the implemented functionality for annotation may improve dereplication of known and unknown molecules.

This vignette uses as a case study indiscriminant MS/MS (idMS/MS) data from Li et al. (2015), unpublished idMS/MS data collected from different organs of tobacco flowers and data from the Global Natural Product Social Molecular Networking (GNPS) library to navigate through the analysis pipeline. The pipeline includes creation of an MSP-object, binning of fragment ions, calculation of a similarity measure (NDP), assignment to a similarity matrix and visualisation of similarity based on interactive and non-interactive graphical tools using the circlize framework (Gu et al., 2014).

MetCirc is currently under active development. If you discover any bugs, typos or develop ideas of improving MetCirc feel free to raise an issue via GitHub or send a mail to the developers.

2 Prepare the environment

To install MetCirc enter the following to the R console

```
if (!requireNamespace("BiocManager", quietly=TRUE))
    install.packages("BiocManager")
BiocManager::install("MetCirc")
```

Before starting, load the MetCirc package. This will also load the required packages circlize, amap, scales and shiny:

```
library(MetCirc)

## Loading required package:  amap
## Loading required package:  circlize
## ========================================
## circlize version 0.4.6
## CRAN page:  https://cran.r-project.org/package=circlize
## Github page:  https://github.com/jokergoo/circlize
## Documentation:  http://jokergoo.github.io/circlize_book/book/
##
## If you use it in published research, please cite:
## Gu, Z. circlize implements and enhances circular visualization
##  in R. Bioinformatics 2014.
## ========================================
```

```
## Loading required package:  scales
## Loading required package:  shiny
```

Two options exist to load data for the MetCirc workflow: (1) loading a data frame with a minimum requirement of a column "id" (comprising unique identifiers for MS/MS features) and of the columns "mz" and "intensity" comprising the fragment ions and their intensities, respectively, or (2) loading a data frame resembling a .MSP file object. In the following we will exemplify the usage of `convert2MSP` and `convertMSP2MSP` to load data into the workflow.

Load example data sets from Li et al. (2015), tissue idMS/MS data and GNPS data. sd02_deconvoluted comprises 360 idMS/MS deconvoluted spectra with fragment ions (m/z, chromatographic retention time, relative intensity in %) and a column with unique identifiers for MS/MS features (here, the corresponding principal component group with the precursor ion). The data set is derived from the study of Li et al. (2015).

```
## load data from Li et al., 2015
data("sd02_deconvoluted", package = "MetCirc")
```

The second data set comes from the data-independent MS/MS collection of different floral organs from tobacco plants. Using our pipeline, this data set will be used to visualise shared metabolites between tissues as well as structural relationships among within- and between-organ MS/MS spectra. MS/MS data are merged across floral organs in one unique data file `idMSMStissueproject.Rdata` as `tissue`. Information on the organ-localisation of each MS/MS spectrum is stored in `compartmentTissue`.

```
## load idMSMStissueproject
data("idMSMStissueproject", package = "MetCirc")
```

The third data set comes from the GNPS data base (downloaded at February 11, 2017 from: http://prime.psc.riken.jp/Metabolomics_Software/MS-DIAL/MSMS-GNPS-Curated-Neg.msp). It contains 22 MS/MS features (a truncated file of the original one) and is formatted in the .MSP file format, a typical format for MS/MS libraries. The MS/MS spectra are (normally) separated by blank lines and give information on the metabolite (its name, precursor m/z value, retention time, class, adduct ion) and contain additional information.

```
## load data from GNPS
data("convertMSP2MSP", package = "MetCirc")
```

3 Prepare data for mass spectral similarity calculations

The `MSP`-class in MetCirc mimicks the typical MS spectral library file in .MSP format in ASCII text format used by mass spectral libraries (http://www.nist.gov/srd/upload/NIST1a11Ver2-0Man.pdf). The functions `convert2MSP` and `convertMSP2MSP` create an entry for each MS/MS

feature and their corresponding spectra including fragment ions and their intensities. Each entry of the MSP-object has the following accessors: retention time, m/z values, metabolite name and class, adduct ion name and further information. The MSP-object contains information on the number of peaks and all fragment ions together with their relative intensity in %. The retention time calculated by the function `convert2MSP` is the average value of the retention time values of all fragment ions belonging to the specific precursor or the retention time value that is stored in the column "id". The data frame containing information on the number of peaks, fragments and their intensities can be accessed by `peaks(MSP-object)`.

3.1 Preparing the sd02_deconvoluted data set for analysis

Here, we convert the examplatory data from Li et al. (2015) into an MSP-object that is later used as the input for mass spectral similarity calculations. The MetCirc workflow requires a data frame in the form of sd02_deconvoluted to run the pipeline. Required columns in such data frames are "mz" (containing the m/z values of fragments), "intensity" (intensity values in % for the respective fragments) and "id". The column "id" has to contain information about the precursor ion and should be a unique descriptor for the MS/MS features. For instance, this column may be the derived from the output of the xcms/CAMERA processing creating unique identifiers for each metabolite decoded in the column "pcgroup". A pcgroup is defined as a peak correlation group obtained by this workflow. It corresponds to a MS or MS/MS spectrum deconvoluted by CAMERA and contains information about the precursor identity and its isotope cluster. The user needs to combine the pcgroup value with the m/z value of the precursor ion of each parent molecule. The column "id" may additionally contain information about the precursor retention time.

Create MSP-object from the data of (Li et al., 2015):

```
## identify precursor mz
finalMSPLi2015 <- convert2MSP(sd02_deconvoluted, splitPattern = " _ ",
                                                  rt = TRUE, splitIndMZ = 2)
```

The `splitPattern` argument takes a character string specifying how elements in the column "id" are separated, `splitIndMZ` and `splitIndRT` (here not used) take numerical values specifying the position of the m/z and retention time value of the precursor ion in the column "mz". Please note that specification of splitIndRT is optional and the retention time values will only be retrieved when rt is set to `TRUE`. Furthermore, `convert2MSP` can take further arguments to retrieve annotation data (metabolite name and classes, adduct ion names, further information; see `?convert2MSP` for further information).

3.2 Preparing the floral organ data set for analysis

We would like to restrict the proof-of-function analysis to four tissues (sepal, SPL; limb, LIM; anther, ANT; style, STY). We will truncate `tissue` in order that it contains only these instances

belonging to these types of tissue. In a next step, we will create a MSP-object, `finalMSP`, comprising the features found in the tissues SPL, LIM, ANT and STY.

```
## create vectors with precursor names present in tissue
tissueSPL <- compartmentTissue[compartmentTissue[,"SPL"] == TRUE, 1]
tissueLIM <- compartmentTissue[compartmentTissue[,"LIM"] == TRUE, 1]
tissueANT <- compartmentTissue[compartmentTissue[,"ANT"] == TRUE, 1]
tissueSTY <- compartmentTissue[compartmentTissue[,"STY"] == TRUE, 1]

## truncate tissue
tissueSPL <- tissue[tissue[,4] %in% tissueSPL,]
tissueLIM <- tissue[tissue[,4] %in% tissueLIM,]
tissueANT <- tissue[tissue[,4] %in% tissueANT,]
tissueSTY <- tissue[tissue[,4] %in% tissueSTY,]

## create msp and combine msp objects of different tissues
## here we set splitIndRT = 2 since column "id" is formatted in the
## following way: m/z value_retention time_pcgroup
finalMSPtissue <- convert2MSP(tissueSPL, rt = TRUE, splitIndRT = 2)
finalMSPtissue <- combine(finalMSPtissue, convert2MSP(tissueLIM,
                                                      rt = TRUE, splitIndRT = 2))
finalMSPtissue <- combine(finalMSPtissue, convert2MSP(tissueANT,
                                                      rt = TRUE, splitIndRT = 2))
## finalMSPtissue <- combine(finalMSPtissue, convert2MSP(tissueSTY,
##                                                       rt = TRUE, splitIndRT = 2))
```

For the `binning` function used later, we need to pass a vector containing the groups (compartments, times, species, etc.) of the metabolites. Here, we will derive a vector, `compartment`, which gives the compartment for each entry of `finalMSPtissue`. `compartment` refers here to floral organs, but could be species names, experimental conditions, etc., too; i.e. the object can be any biological identifier relevant to the comparative experiment conducted.

```
## create vector with compartments
compSPL <- rep("SPL", length(convert2MSP(tissueSPL)))
compLIM <- rep("LIM", length(convert2MSP(tissueLIM)))
compANT <- rep("ANT", length(convert2MSP(tissueANT)))
compSTY <- rep("STY", length(convert2MSP(tissueSTY)))

compartment <- c(compSPL, compLIM, compANT, compSTY)
```

3.3 Preparing the GNPS data set for analysis

Alternatively as mentioned above, an MSP-object can also created from .MSP objects, that are typical data formats for storing MS/MS libraries. Required properties of such a data frame are

the name of the metabolite (row entry "NAME:"), the m/z value of the precursor ion ("PRECURSORMZ" or "EXACTMASS:"), the number of peaks of the feature ("Num Peaks:") and information on fragments and peak areas (see `?convertMSP2MSP` for further information and retrieve `data("convertMSP2MSP", package = "MetCirc")` for a typical msp data frame).

Create `MSP`-object from the GNPS .MSP file:

```
finalMSPGNSP <- convertMSP2MSP(msp2msp)
```

4 Binning and calculation of similarity matrix

4.1 Workflow for `tissue` data set using fragment ions

Binning. Due to slight differences in m/z values over measurements, fragments might have m/z values which differ from other fragments even though they are in theory identical. `binning` will bin together fragment ions which are similar (set by the parameter `tol` for tolerance). In the following this will allow for comparison between m/z values. The functions `binning` bins fragments together based on minimal distance to bins which were calculated either by the mean or the median of fragments which were put in intervals according to the `tol` parameter.

`binning` expects a vector (`group`) which comprises membership of the entries in the `msp` object, to a compartment, species, individual, etc. If `group` is not specified `binning` will create an internal dummy variable group ("a" with the length of the `msp` object). We use here the `tissue` data set.

```
## create data frame with binned fragments
binnedMSP <- binning(msp = finalMSPtissue, tol = 0.01,
                     group = compartment, method = "median")
```

Calculation of the similarity matrix. The normalised dot product (NDP) calculates the similarity coefficient between two MS/MS features. For a considered MS/MS pair, peak intensities of shared m/z values for precursor/fragment ions and neutral losses are employed as weights $W_{S1,i}$ and $W_{S1,i}$ within the following formula:

$$NDP = \frac{\sum(W_{S1,i} \cdot W_{S2,i})^2}{\sum(W_{S1,i}^2) * \sum(W_{S2,i}^2)},$$

with S1 and S2 the spectra 1 and 2, respectively, of the ith of j common peaks differing by the tolerance parameter specified in `binning`. Weights are calculated according to $W = [peak\ intensity]^m \cdot [m/z]^n$, with m = 0.5 and n = 2 as default values as suggested by MassBank. For further information see Li et al. (2015).

`createSimilarityMatrix` calculates all pair-wise NDP coefficients between MS/MS features. `createSimilarityMatrix` needs a matrix as an argument which has the fragment ions as rows

(m/z / retention time) and all fragment ions as column names. Entries of such a matrix are intensities for a specific fragment ion (intensity will be zero if fragment ion does not occur for the respective precursor) for a given MS/MS feature. The function `binning` will return a matrix with such properties.

```
## similarity Matrix
similarityMat <- createSimilarityMatrix(binnedMSP)
```

`createSimilarityMatrix` returns a square matrix with precursor m/z / retention time as column and row names. The entries of the returned matrix are NDP scores ranging between 0 and 1 that are the pair-wise similarities between the MS/MS features.

Clustering/visualisation. At this stage, we would like to visualise the pair-wise similarities of MS/MS features after clustering them. Many functions are available to cluster features such as `hclust` from `stats`, `Mclust` from `mclust` or `hcluster` from `amap`. We would like to use the latter to cluster similar features. To cluster features and visualise the result (see figure 1) we enter:

```
## load package amap
hClustMSP <- hcluster(similarityMat, method = "spearman")
## visualise clusters
plot(hClustMSP, labels = FALSE, xlab="", sub="")
```

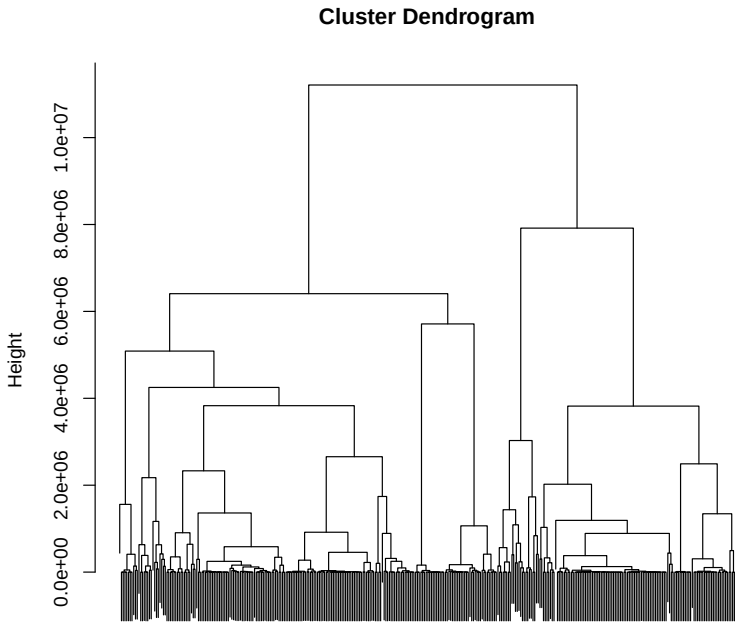

Figure 1: Cluster dendrogramm for similarity matrix based on fragment ion NDP calculation

To promote readibility we will not show labels. These can be printed to the R console by `colnames(similarityMat)[hClustMSP$order]`.

Extraction of highly-related features using clustering. Within the R session the similarity matrix can be truncated by using the above-mentioned functions to retrieve a similarity matrix that contains highly-related MS/MS features. For instance, this might be needed when we want to analyse modules of high similarity, representing e.g. a certain metabolite class. By way of example, we extract in the following all features from module 1 when defining three modules in total and define a new similarity matrix of highly-related features. This kind of matrix can be used in later analysis steps, e.g. in the analysis with **shinyCircos**.

```
## define three clusters
cutTreeMSP <- cutree(hClustMSP, k = 3)
## extract feature identifiers that belong to module 1
module1 <- names(cutTreeMSP)[as.vector(cutTreeMSP) == "1"]
## create a new similarity matrix that contains only highly related features
similarityMat_module1 <- similarityMat[module1, module1]
```

4.2 Workflow for `tissue` data set using neutral losses

Another way to compare the similarity of metabolites is based on neutral losses (cf. table 1 in the appendix for a selection of common neutral losses): common neutral losses are shared among structurally-related metabolites. MetCirc contains functionality to calculate neutral losses from MSP-objects. To convert a MSP-object with fragments into a MSP-object with neutral losses enter:

```
nlMSP <- msp2FunctionalLossesMSP(finalMSPtissue)
```

Binning and calculation of the similarity matrix. Analogously to the MSP-object with fragments we can bin the `nlMSP` MSP-object with neutral losses and create a similarity matrix:

```
## bin msp file with functional losses, create table with same fragments
## (binning)
nlBinnedMSP <- binning(nlMSP, tol = 0.01, group = compartment, method = "median")
## similarity Matrix
nlSimilarityMat <- createSimilarityMatrix(nlBinnedMSP)
```

The NDP calculation uses here neutral losses corresponding to mass differences between the precursor and fragment ions within a spectrum. The intensities are the respective intensities of the fragments.

Clustering/visualisation. Analogously to the MSP-object with fragment ions, we are able to cluster the similarity matrix based on neutral losses.

```
## Clustering
nlHClustMSP <- hcluster(nlSimilarityMat, method = "spearman")
```

To visualise the clustering enter the following line of code (labels will not be displayed due to readibility):

```
plot(nlHClustMSP, labels = FALSE)

## labels can be reproduced by entering in the console
colnames(nlSimilarityMat)[nlHClustMSP$order]
```

5 Visualisation using the shiny/circlize framework

MetCirc comprises functionality to visualise metabolomics data, exploring it interactively and annotate unknown features based on similarity to other (known) features. One of the key features of the implemented interactive framework is, that groups can be compared. A group specifies the affiliation of the sample: it can be any biological identifier relevant to the comparitive experiment conducted, e.g. it can be a specific tissue, different times, different species, etc. The visualisation tools implemented in MetCirc allow then to display similarity between precursor ions between and/or within groups.

`shinyCircos` uses the function `createLinkMatrix` which selects these precursor ions that have a normalised dot product within the range defined by `threshold_low` and `threshold_high` to a precursor ion. Internally, in `shinyCircos`, `createLinkMatrix` will be called to produce link matrices based on the given thresholds.

```
linkMat <- createLinkMatrix(similarityMatrix = similarityMat,
                            threshold_low=0.5, threshold_high=1)
```

As we have calculated similarity coefficients between precursors, we would like to visualise these connections interactively and explore the data. The MetCirc package implements `shinyCircos` that allows for such kind of exploration. It is based on the shiny and on the circlize (Gu et al., 2014) framework. Inside of `shinyCircos` information of precursor ions are displayed by (single-) clicking over precursors. Precursors can also be permanently selected by double-clicking on them. The similarity coefficients can be thresholded by changing the slider input. Also, on the sidebar panel, the type of link to be displayed can be selected: should only links between groups be displayed, should only links within groups be displayed or should all links be displayed? Ordering inside of groups can be changed by selecting the respective option in the sidebar panel. Momentarily, there are options to reorder features based on clustering, the m/z or the retention time of the precursor ion. In the "Appearance" tab, the size of the plot can be changed, as well

as the precision of the displayed values for the m/z and retention time values. Please note, that the precision will only be changed in the text output (not in the graphical output). Also, the user can decide if the legend is displayed or not.
On exiting the shiny application via the exit button in the sidebar panel, selected precursors will be returned which are allocated here to `selectedFeatures`. `selectedFeatures` is a vector of the precursor names.

To start the shiny app, run

```
selectedFeatures <- shinyCircos(similarityMat)
```

To show and modify annotations enter

```
selectedFeatures <- shinyCircos(similarityMat, msp = finalMSPtissue)
```

to the console. This will load information stored in the `MSP`-object to the `shinyCircos` interface. Information (metabolite names and class, adduct ion name and further information) will be printed to the interface when (single-)clicking on MS/MS features. Note, that on the sidebar on the right side text fields will appear, that allow for changing the annotation of the selected feature. Click "update annotation" to save the changes to the `MSP`-object. On exiting `shinyCircos` via the exit button, selected precursors and the (updated) `MSP`-object will be returned to the console. In the example above, enter `selectedFeatures$msp` to retrieve the `MSP`-object.

MetCirc allows also to create such figures outside of an interactive context, which might be helpful to create figures and export them e.g. in .pdf or .jpeg format. `shinyCircos` does currently not support to export figures as they can be easily rebuilt outside of `shinyCircos`; building figures outside of the interactive context also promotes reproducibility of such figures.

To rebuild the figure in a non-interactive environment, run

```
## order similarity matrix according to retention time
simM <- createOrderedSimMat(similarityMat, order = "retentionTime")
groupname <- rownames(simM)
## create link matrix
linkMat <- createLinkMatrix(similarityMatrix = simM,
                    threshold_low = 0.99, threshold_high = 1)
## cut link matrix (here: only display links between groups)
linkMat_cut <- cutLinkMatrix(linkMat, type = "inter")

## set circlize paramters
circos.par(gap.degree = 0, cell.padding = c(0, 0, 0, 0),
            track.margin = c(0, 0))

## here set indSelected arbitrarily
```

```
indSelected <- 1
selectedFeatures <- groupname[1]

## actual plotting
plotCircos(groupname, linkMat_cut, initialize = TRUE, featureNames = TRUE,
    cexFeatureNames = 0.2, groupSector = TRUE, groupName = FALSE,
    links = FALSE, highlight = TRUE)

highlight(groupname = groupname, ind = indSelected, LinkMatrix =
            linkMat_cut)
## plot without highlighting
plotCircos(groupname, linkMat_cut, initialize = TRUE, featureNames = TRUE,
    cexFeatureNames = 0.2, groupSector = TRUE, groupName = FALSE, links = TRUE,
    highlight = FALSE)
```

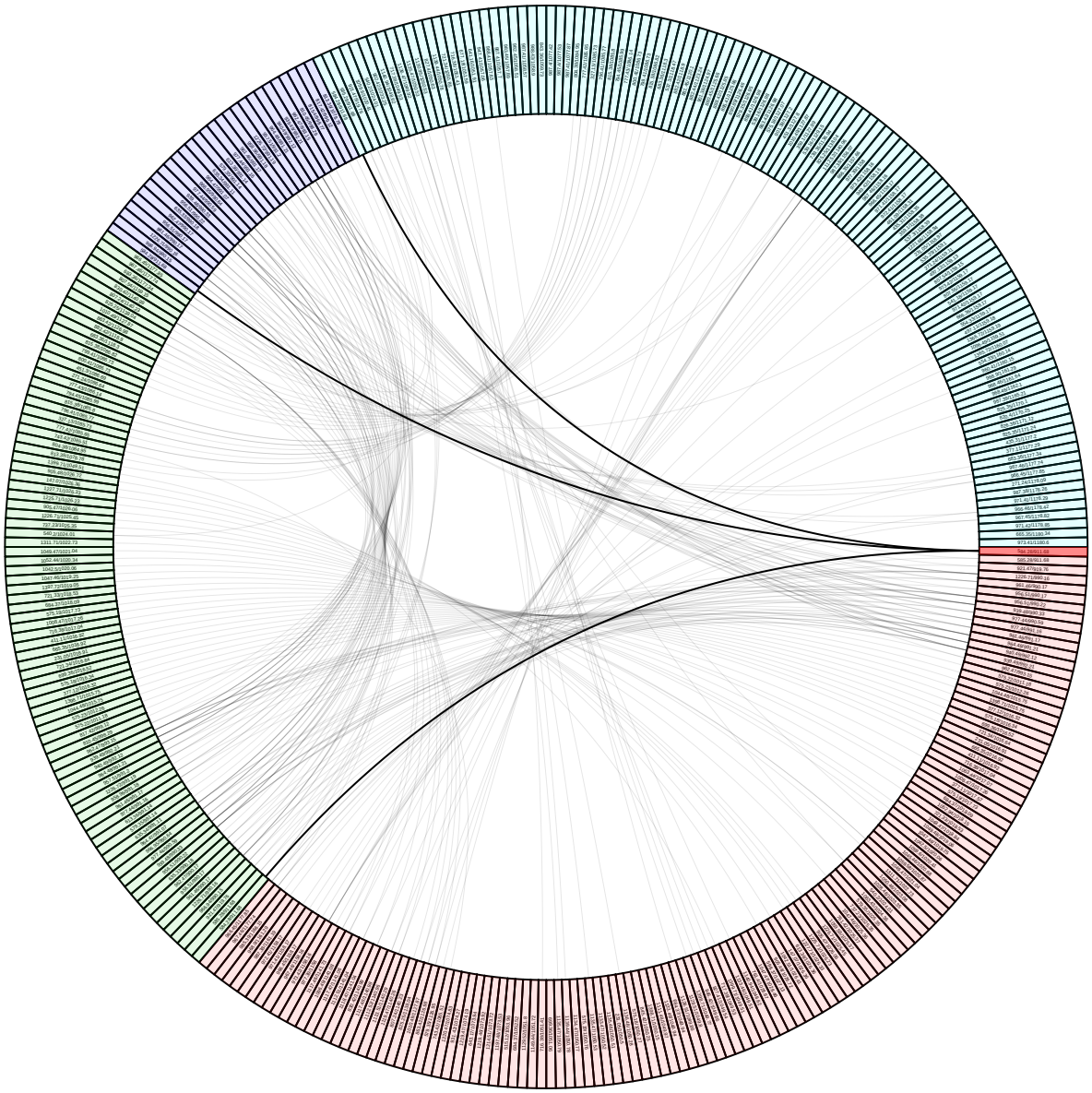

Figure 2: Examplary plot where an arbitrary feature is highlighted. Upon highlighting all links will be plotted in grey (expect links to and from highlighted features). The intensity of the background colour of features will be reduced as well. Features belonging to a group (species, individual, organ, different time) will be indicated by the same background colour.

References

Gu, Z. et al. (2014). "circlize implements and enhances circular visualization in R". In: *Bioinformatics* 30, pp. 2811–2812. DOI: 10.1093/bioinformatics/btu393.

Krzywinski, M.I. et al. (2009). "Circos: An information aesthetic for comparative genomics". In: *Genome Research* 19, pp. 1639–1645. DOI: 10.1101/gr.092759.109.

Li, D., I.T. Baldwin, and E. Gaquerel (2015). "Navigating natural variation in herbivory-induced secondary metabolism in coyote tobacco populations using MS/MS structural analysis". In: *PNAS* 112, E4147–E4155. DOI: 10.1073/pnas.1503106112.

Watrous, J. et al. (2012). "Mass spectral molecular networking of living microbial colonies". In: *PNAS* 109, E1743–E1752. DOI: 10.1073/pnas.1203689109.

Appendix

Session information

All software and respective versions to build this vignette are listed here:

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: OS X El Capitan 10.11.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] MetCirc_1.14.0 shiny_1.3.2    scales_1.0.0   circlize_0.4.6
## [5] amap_0.8-16    knitr_1.22
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.1          later_0.8.0         digest_0.6.18
##  [4] mime_0.6            R6_2.4.0            grid_3.6.0
##  [7] xtable_1.8-4        magrittr_1.5        evaluate_0.13
## [10] highr_0.8           stringi_1.4.3       GlobalOptions_0.1.0
## [13] promises_1.0.1      tools_3.6.0         stringr_1.4.0
## [16] munsell_0.5.0       httpuv_1.5.1        xfun_0.6
## [19] compiler_3.6.0      colorspace_1.4-1    shape_1.4.4
## [22] htmltools_0.3.6
```

Neutral losses

Table 1: The table gives examplatory fractionation of precursors into neutral losses (given their m/z and the corresponding atoms):

CH_2	14.0157
CH_4	16.0313
NH_3	17.0265
H_2O	18.0106
K^+ to $NH_4{}^+$"	20.9293
Na^+ to H^+	21.9819
C_2H_2	26.0157
CO	27.9949
C_2H_4	28.0313
CH_3N	29.0266
CH_2O	30.0106
CH_5N	31.0422
S	31.9721
H_2S	33.9877
$K^+$to H^+	37.9559
C_2H_2O	42.0106
C_3H_6	42.0470
CHNO	43.0058
CO_2	43.9898
CH_2 O_2	46.0055
C_4H_8	56.0626
C_3H_9N	59.0735
C_2H_4 O_2	60.0211
CH_4N_2O	60.0324
SO_2	63.9619
C_5H_8	68.0626
C_3H_6 O_2	74.0368
C_6H_6	78.0470
SO_3	79.9568
$C_3H_2O_3$	86.0004
$C_4H_8O_2$	88.0517
$C_4H_{12}N_2$	88.1000
$H_2(SO)_4$	97.9674
$H_3(PO)_4$	97.9769
$C_5H_{10}O_2$	102.0618
$C_3H_4O_4$	104.0110
$C_6H_{12}O_2$	116.0861
$C_2H_5O_4P$	123.9926

$C_5H_8O_4$	132.0423
$C_7H_{19}N_3$	145.1579
$C_6H_{10}O_4$	146.0579
$C_6H_{10}O_5$	162.0528
$C_6H_{12}O_5$	164.0685
$C_6H_8O_6$	176.0321
$C_6H_{12}O_6$	180.0634
$C_6H_{10}O_7$	194.0427
$C_8H_{12}O_6$	204.0655
$C_{11}H_{10}O_4$	206.0579
$C_{10}H_{15}N_3\ O_6\ S$	305.0682
$C_{10}H_{17}N_3\ O_6\ S$	307.0838
$C_{12}H_{20}O_{10}$	324.1057
$C_{12}H_{22}O_{11}$	342.1162