Workflow metabolite set enrichment analysis with bootstrapping with `bmetenrichr`

This R-package aims to perform metabolite set enrichment analysis (MSEA) on single-cell metabolomics datasets. In contrast to bulk-metabolomics, metabolite annotation is often more ambiguous with fully resolved molecular structures. That means, annotations are vectors of isomeric (and/or isobaric) molecules, complicating downstream MSEA. This package uses a boostrapping approach by performing enrichment analyses many times with random sampling of the isomers/isobars.

```
options(stringsAsFactors = FALSE, warn = -1)

## install devtools if not installed
if(!("devtools" %in% rownames(installed.packages()))){
  install.packages("devtools",  repos = c(CRAN = "http://cran.rstudio.com"))
  }

## install bmetenrichr if not installed
if(!("bmetenrichr" %in% rownames(installed.packages()))){
   devtools::install_github(repo = "martijnmolenaar/bmetenrichr", build_vignettes = TRUE)
}

library(bmetenrichr)
#> Loading required package: ggplot2
#> Loading required package: dplyr
#>
#> Attaching package: 'dplyr'
#> The following objects are masked from 'package:stats':
#>
#>     filter, lag
#> The following objects are masked from 'package:base':
#>
#>     intersect, setdiff, setequal, union
```

The package contains example data from Rappez et al., 2021 (https://doi.org/10.1038/s41592-021-01198-0).

```
data("Rappez_et_al")

## the main input is a single-cell metabolomics matrix with molecules as rows and cells as columns
Rappez_et_al$sc_matrix[1:10,1:10]
#>                          8        15         18         25         27
#> C10H10N4O.K     0.01145815 0.0000000 0.00000000 0.00000000 0.00000000
#> C10H10O6.K      0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
#> C10H11N4O7P.Na  0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
#> C10H11N5O3.H    0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
#> C10H12ClN5O3.K  0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
#> C10H12ClN5O3.Na 0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
#> C10H12ClNO4.H   0.01707534 0.0620677 0.05203464 0.01376432 0.02417577
#> C10H12ClNO4.K   0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
```

```
#> C10H12ClNO4.Na  0.17496077 0.1702377 0.17311070 0.16992632 0.01635700
#> C10H12FN5O4.H   0.00000000 0.0000000 0.00000000 0.00000000 0.00000000
#>                         29          35         38          41          47
#> C10H10N4O.K     0.008558847 0.000000000 0.00000000 0.000000000 0.000000000
#> C10H10O6.K      0.000000000 0.000000000 0.00000000 0.000000000 0.006726224
#> C10H11N4O7P.Na  0.000000000 0.005782876 0.02968958 0.000000000 0.000000000
#> C10H11N5O3.H    0.000000000 0.000000000 0.00000000 0.000000000 0.000000000
#> C10H12ClN5O3.K  0.000000000 0.000000000 0.00000000 0.000000000 0.000000000
#> C10H12ClN5O3.Na 0.000000000 0.000000000 0.00000000 0.000000000 0.000000000
#> C10H12ClNO4.H   0.028755533 0.030977567 0.03611660 0.009692678 0.031959898
#> C10H12ClNO4.K   0.000000000 0.000000000 0.00000000 0.000000000 0.000000000
#> C10H12ClNO4.Na  0.174631645 0.147417684 0.12427280 0.160955912 0.120047020
#> C10H12FN5O4.H   0.000000000 0.000000000 0.00000000 0.000000000 0.000000000

## for the molecules, a vector with molecular formulas plus adduct is required
rownames(Rappez_et_al$sc_matrix)[1:10]
#>  [1] "C10H10N4O.K"     "C10H10O6.K"      "C10H11N4O7P.Na"  "C10H11N5O3.H"
#>  [5] "C10H12ClN5O3.K"  "C10H12ClN5O3.Na" "C10H12ClNO4.H"   "C10H12ClNO4.K"
#>  [9] "C10H12ClNO4.Na"  "C10H12FN5O4.H"

## a conditions vector is required to define to which condition a given cell belongs
Rappez_et_al$conditions[1:10]
#>  [1] "F" "F" "F" "F" "F" "F" "F" "F" "F" "F"

## in this analysis, only specific annotations should be included as others are extracellular
Rappez_et_al$cellular[1:10]
#>     C10H10N4O.K      C10H10O6.K  C10H11N4O7P.Na    C10H11N5O3.H  C10H12ClN5O3.K
#>           FALSE           FALSE           FALSE            TRUE           FALSE
#> C10H12ClN5O3.Na   C10H12ClNO4.H   C10H12ClNO4.K  C10H12ClNO4.Na   C10H12FN5O4.H
#>           FALSE           FALSE           FALSE           FALSE           FALSE
```

enrichment analysis with isomers

The main enrichment object can be generated as follows. By default, `bmetenrichr` uses LION (Molenaar et al., 2019, GigaScience, https://doi.org/10.1093/gigascience/giz061) as metabolite sets.

```
myTestRun <-
  initEnrichment(scmatrix = Rappez_et_al$sc_matrix,
                 annotations = rownames(Rappez_et_al$sc_matrix),
                 conditions = Rappez_et_al$conditions,
                 include = Rappez_et_al$cellular,
                 condition.x = "U",
                 condition.y = "F"                    )
#>
#> Parsing isomers...
#> single-cell metabolomics matrix of 3385 metabolites and 8807 cells
#> active pathway: LION
#>
#> conditions: F, FI, FIT, U
#>
#> condition.x: U
#> condition.y: F
```

First, metabolites are ranked by `rankScore()`

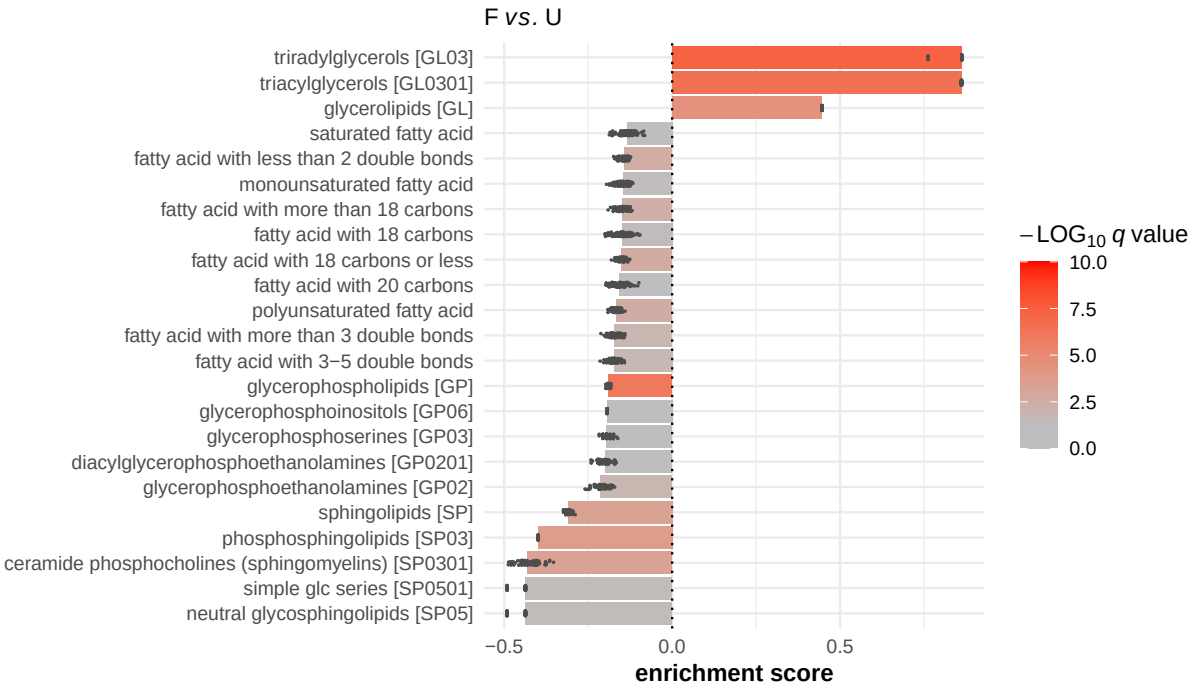
```
## rank metabolites, in this case by t.test statistic

myTestRun <- rankScore(myTestRun, ranking.by = 't.test')
#> number of ties: 19 (1.73%)
```

Then, perform enrichment analysis with n = 100 bootstraps

```
myTestRun <- calcEnrichment(myTestRun, n = 100)
#>
#> Bootstrapping...
#>
#> Match to pathway...
#> 35.43% of annotations were matched to pathway
#>
#> Perform enrichment analysis...
```

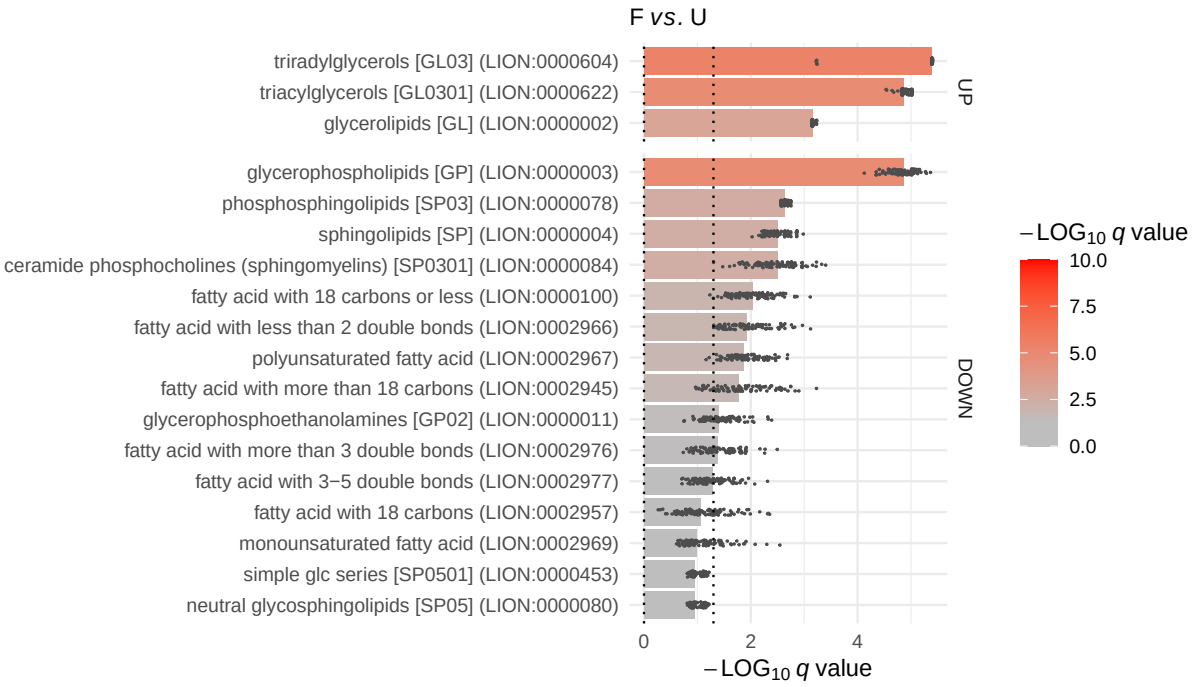
The enrichment analysis can be visualized with `plotEnrichment()`, here with enrichment score (ES) on x-axis

```
plotEnrichment(myTestRun, min.annotations = 5, q.value.cutoff = .1, by.statistic = "ES")
```



Plots can also be arranged with q.values on x-axis, and with LION IDs

```
plotEnrichment(myTestRun, min.annotations = 5, q.value.cutoff = .05, plotIDs = T,
               by.statistic = "q.value")
```



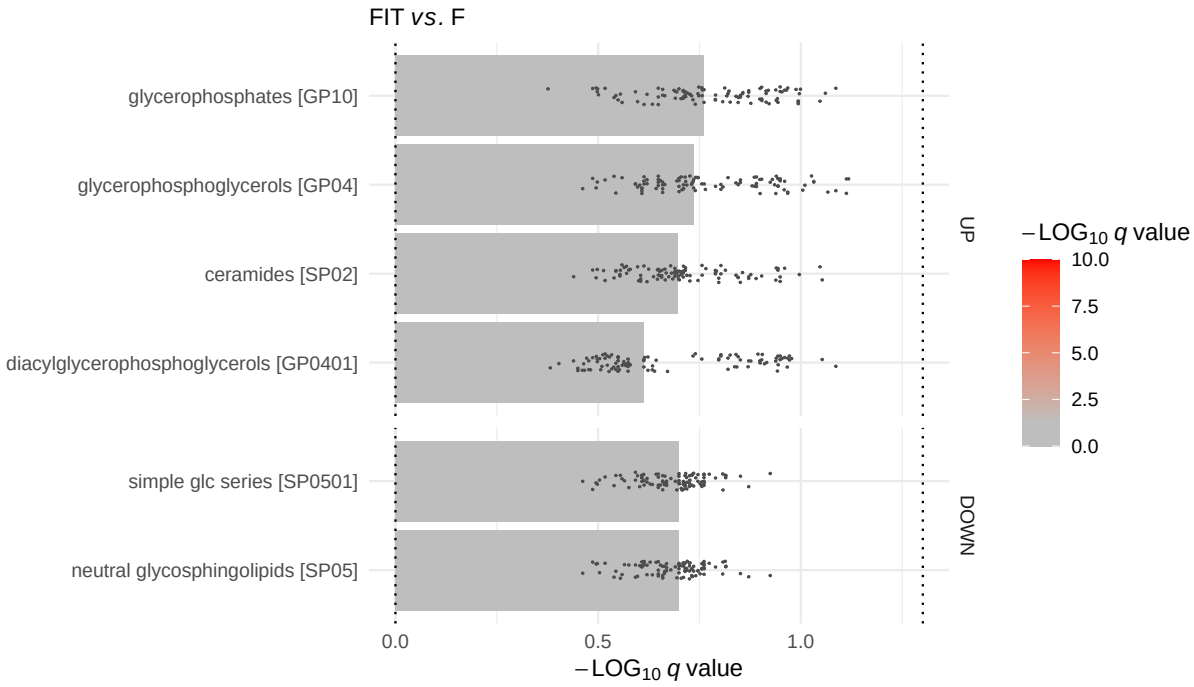
The enrichment analysis can also be exported as `data.frame`:

```
enrichmentTable(myTestRun)[1:10,]
#>         LION_ID                                                LION_name     n
#> 1  LION:0000604                                  triradylglycerols [GL03]   9.0
#> 2  LION:0000003                                glycerophospholipids [GP] 290.0
#> 3  LION:0000622                                triacylglycerols [GL0301]   8.0
#> 4  LION:0000002                                       glycerolipids [GL]  28.0
#> 5  LION:0000078                              phosphosphingolipids [SP03]  30.0
#> 6  LION:0000004                                     sphingolipids [SP]  48.5
#> 7  LION:0000084 ceramide phosphocholines (sphingomyelins) [SP0301]  24.5
#> 8  LION:0000100                      fatty acid with 18 carbons or less 206.0
#> 9  LION:0002966                fatty acid with less than 2 double bonds 222.0
#> 10 LION:0002967                                polyunsaturated fatty acid 150.0
#>     ES_median       ES_sd p.value_median   p.value_sd q.value_median
#> 1   0.8622590 0.021932022   4.596620e-08 4.378734e-06   4.045026e-06
#> 2  -0.1889701 0.003716503   4.675098e-07 3.509875e-07   1.401890e-05
#> 3   0.8614679 0.000000000   3.317287e-07 0.000000e+00   1.401890e-05
#> 4   0.4463952 0.000000000   3.787071e-05 0.000000e+00   7.046016e-04
#> 5  -0.3992509 0.000000000   1.823436e-04 0.000000e+00   2.292319e-03
#> 6  -0.3102381 0.007905573   2.908494e-04 1.696354e-04   3.195926e-03
#> 7  -0.4316684 0.028083167   2.533914e-04 9.772138e-04   3.197399e-03
#> 8  -0.1499332 0.009538610   1.055901e-03 1.663671e-03   9.216216e-03
#> 9  -0.1422782 0.010873956   1.470311e-03 2.037405e-03   1.192274e-02
#> 10 -0.1658310 0.009725300   1.601349e-03 1.932453e-03   1.343570e-02
#>      q.value_sd fraction.bootstrap.presence
#> 1  1.269854e-04                           1
#> 2  1.084090e-05                           1
#> 3  3.452169e-06                           1
#> 4  2.847325e-05                           1
#> 5  2.704334e-04                           1
```

```
#> 6  1.623456e-03                          1
#> 7  7.009310e-03                          1
#> 8  1.010707e-02                          1
#> 9  1.300131e-02                          1
#> 10 1.195363e-02                          1
```

To compare other conditions, use `setConditions()`:

```
## now, let's test FIT vs F

myTestRun <-  setConditions(object = myTestRun, condition.x = 'F', condition.y = 'FIT')
myTestRun
#> single-cell metabolomics matrix of 3385 metabolites and 8807 cells
#> active pathway: LION
#>
#> conditions: F, FI, FIT, U
#>
#> condition.x: F
#> condition.y: FIT

## rank metabolites, in this case by t.test statistic

myTestRun <- rankScore(myTestRun, ranking.by = 't.test')
#> number of ties: 24 (2.19%)

## and perform enrichment analysis

myTestRun <- calcEnrichment(myTestRun, n = 100)
#>
#> Bootstrapping...
#>
#> Match to pathway...
#> 15.73% of annotations were matched to pathway
#>
#> Perform enrichment analysis...

plotEnrichment(myTestRun, min.annotations = 5, q.value.cutoff = .05)
```



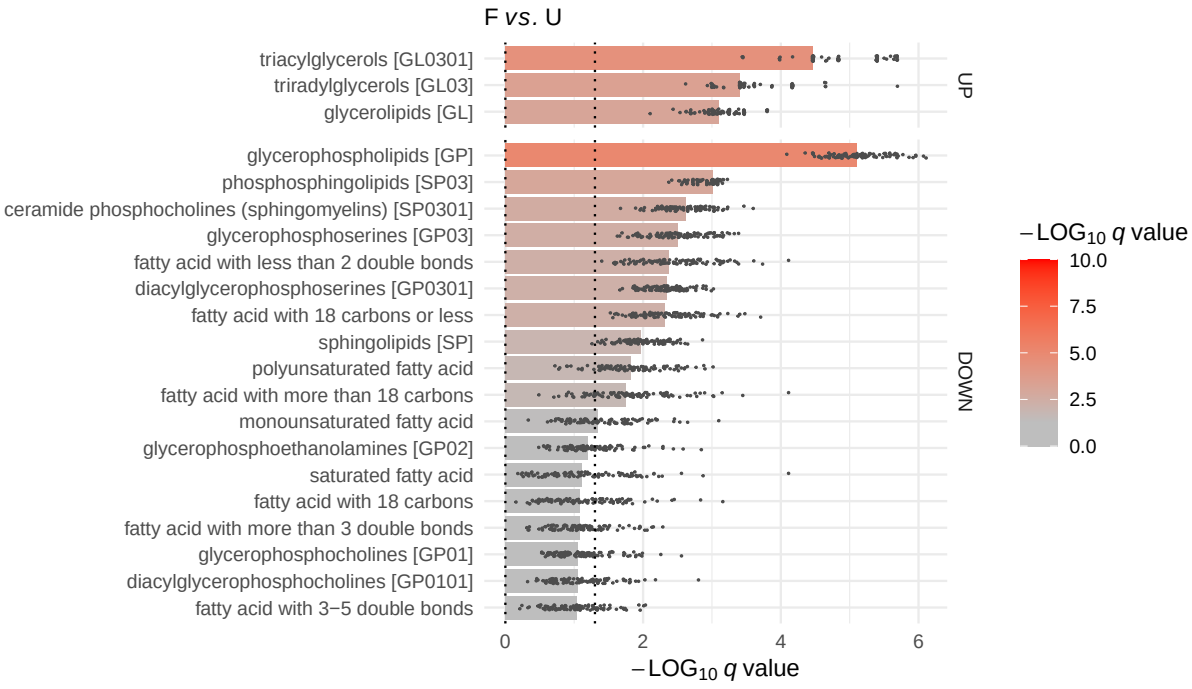
enrichment analysis with isomers and isobars

By default, only isomers are included. With isobars = TRUE, it's also possible to include isobars within a set m/z range:

```
## create object

myTestRun <-
  initEnrichment(scmatrix = Rappez_et_al$sc_matrix,
                 isobars = TRUE,                      ## to include isobars (default is FALSE)
                 mass_range_ppm = 3,                  ## mass range to define isobars
                 polarization_mode = "positive",      ## mode is important to include the right adducts
                 annotations = rownames(Rappez_et_al$sc_matrix),
                 conditions = Rappez_et_al$conditions,
                 include = Rappez_et_al$cellular,
                 condition.x = "U",
                 condition.y = "F"                    )
#> polarization_mode is: positive
#>
#> Parsing isomers...
#> Parsing potential isobars...
#>
#> single-cell metabolomics matrix of 3385 metabolites and 8807 cells
#> active pathway: LION
#>
#> conditions: F, FI, FIT, U
#>
#> condition.x: U
#> condition.y: F
```

Downstream, the same workflow can be used:

```
## rank metabolites, in this case by t.test statistic

myTestRun <- rankScore(myTestRun, ranking.by = 't.test')
#> number of ties: 19 (1.73%)

## perform enrichment analysis with n = 100 bootstraps

myTestRun <- calcEnrichment(myTestRun, n = 100)
#>
#> Bootstrapping...
#>
#> Match to pathway...
#> 35.34% of annotations were matched to pathway
#>
#> Perform enrichment analysis...

## example of the annotations, that now also include isobars

myTestRun$annotations[[
  sample(which(sapply(myTestRun$isobars_list, length) > 1), size = 1)]][1:10]
#>                                isomer                                  isobar1
#>                            "PI(48:0)"  "PS(28:5(10Z,13Z,16Z,19Z,22Z)/26:0)"
#>                              isobar2                                  isobar3
#> "PS(18:1(11E)/36:4(21Z,24Z,27Z,30Z))"  "PS(26:5(11Z,14Z,17Z,20Z,23Z)/28:0)"
#>                              isobar4                                  isobar5
#> "PS(20:1(11Z)/34:4(19Z,22Z,25Z,28Z))" "PS(18:1(11Z)/36:4(21Z,24Z,27Z,30Z))"
#>                              isobar6                                  isobar7
#>  "PS(30:5(12Z,15Z,18Z,21Z,24Z)/24:0)"  "PS(28:5(13Z,16Z,19Z,22Z,25Z)/26:0)"
#>                              isobar8                                  isobar9
#> "PS(28:4(13Z,16Z,19Z,22Z)/26:1(17Z))" "PS(22:1(13Z)/32:4(17Z,20Z,23Z,26Z))"

## plot enrichment analysis, with q.values on x-axis, and with LION IDs

plotEnrichment(myTestRun, min.annotations = 5, q.value.cutoff = .05,
               by.statistic = "q.value")
```


F vs. U
triacylglycerols [GL0301]
triradylglycerols [GL03]
glycerolipids [GL]
glycerophospholipids [GP]
phosphosphingolipids [SP03]
ceramide phosphocholines (sphingomyelins) [SP0301]
glycerophosphoserines [GP03]
fatty acid with less than 2 double bonds
diacylglycerophosphoserines [GP0301]
fatty acid with 18 carbons or less
sphingolipids [SP]
polyunsaturated fatty acid
fatty acid with more than 18 carbons
monounsaturated fatty acid
glycerophosphoethanolamines [GP02]
saturated fatty acid
fatty acid with 18 carbons
fatty acid with more than 3 double bonds
glycerophosphocholines [GP01]
diacylglycerophosphocholines [GP0101]
fatty acid with 3−5 double bonds
UP
DOWN
0
2
4
6
−LOG10 q value
−LOG10 q value
10.0
7.5
5.0
2.5
0.0