Forestplot Tutorial

Visualizations for Epidemiological Analysis

Maria Kalimeri, maria.kalimeri@nightingalehealth.com, Nightingale Health Ltd.
Qin Wang, qin.wang@computationalmedicine.fi, Computational Medicine, Faculty of Medicine, University of Oulu

Mar. 24, 2018

Installation instructions

Forestplot is an under-development R package to plot associations of diseases and phenotypes in the form a forest plot (a.k.a. blobbogram). Specifically, it offers built-in naming and grouping features to assist visualizations of epidimiological analysis with NMR metabolomics quantified biomarkers (see section "Overview of NMR data"). The associations may be linear (linear regression), odds ratios (logistic regression) or hazard ratios (e.g. Cox Proportional hazards).

To install, first open an R session in the parent directory of the `forestplot/` folder. You will need the package `devtools`, if not already installed, install it by typing:

```
# Install
install.packages("devtools")
```

Then install and load `forestplot` by typing:

```
# (Make sure you are in the parent directory of
# forestplot) Install
devtools::install(pkg = "forestplot")
# Load in session
library(forestplot)
```

Hopefully, dependencies will be installed automatically. If not let me know. (See also pdf manual for list of dependencies.)

Overview of NMR data

The biomarker concentration data are often available in both tsv and xlsx formats. R can read both but in this tutorial we will focus on tsv format. We will be using the collection of the `tidyverse` packages which you need to have installed. Naturally, the previous step installed also `tidyverse` since it is a dependency for `forestplot`. If `tidyverse` is not installed, install it by typing `>install.packages("tidyverse")` and then load it by typing `>library(tidyverse)`.

Below, we load two example tsv files in R and view the available biomarkers (the files can be found in the `demos` folder):

```
# Read the biomarker concentration file
metabol_data_1 <- read_tsv(file = "example_nmr_1.tsv",
                     na = c("NA", "NDEF", "TAG"),
                     col_types = cols(.default = col_double(), sampleid = col_character()))
metabol_data_2 <- read_tsv(file = "example_nmr_2.tsv",
```

```
                    na = c("NA", "NDEF", "TAG"),
                    col_types = cols(.default = col_double(), sampleid = col_character()))
class(metabol_data_1)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
print(head(metabol_data_1))
```

```
## # A tibble: 6 x 229
##   sampleid    `XXL-VLDL-P` `XXL-VLDL-L` `XXL-VLDL-PL` `XXL-VLDL-C`
##   <chr>              <dbl>        <dbl>         <dbl>        <dbl>
## 1 id4878   0.0000000000893      0.0190      0.00281       0.00215
## 2 id0129   0.000000000291       0.0629      0.00694       0.0133
## 3 id8068   0.0000000000184      0.00391     0.0000568     0.000986
## 4 id9348   0.000000000276       0.0601      0.00660       0.0146
## 5 id5818   0                    0           0             0
## 6 id4003   0.0000000000506      0.0108      0.00120       0.00213
## # ... with 224 more variables: `XXL-VLDL-CE` <dbl>, `XXL-VLDL-FC` <dbl>,
## #   `XXL-VLDL-TG` <dbl>, `XL-VLDL-P` <dbl>, `XL-VLDL-L` <dbl>,
## #   `XL-VLDL-PL` <dbl>, `XL-VLDL-C` <dbl>, `XL-VLDL-CE` <dbl>,
## #   `XL-VLDL-FC` <dbl>, `XL-VLDL-TG` <dbl>, `L-VLDL-P` <dbl>,
## #   `L-VLDL-L` <dbl>, `L-VLDL-PL` <dbl>, `L-VLDL-C` <dbl>,
## #   `L-VLDL-CE` <dbl>, `L-VLDL-FC` <dbl>, `L-VLDL-TG` <dbl>,
## #   `M-VLDL-P` <dbl>, `M-VLDL-L` <dbl>, `M-VLDL-PL` <dbl>,
## #   `M-VLDL-C` <dbl>, `M-VLDL-CE` <dbl>, `M-VLDL-FC` <dbl>,
## #   `M-VLDL-TG` <dbl>, `S-VLDL-P` <dbl>, `S-VLDL-L` <dbl>,
## #   `S-VLDL-PL` <dbl>, `S-VLDL-C` <dbl>, `S-VLDL-CE` <dbl>,
## #   `S-VLDL-FC` <dbl>, `S-VLDL-TG` <dbl>, `XS-VLDL-P` <dbl>,
## #   `XS-VLDL-L` <dbl>, `XS-VLDL-PL` <dbl>, `XS-VLDL-C` <dbl>,
## #   `XS-VLDL-CE` <dbl>, `XS-VLDL-FC` <dbl>, `XS-VLDL-TG` <dbl>,
## #   `IDL-P` <dbl>, `IDL-L` <dbl>, `IDL-PL` <dbl>, `IDL-C` <dbl>,
## #   `IDL-CE` <dbl>, `IDL-FC` <dbl>, `IDL-TG` <dbl>, `L-LDL-P` <dbl>,
## #   `L-LDL-L` <dbl>, `L-LDL-PL` <dbl>, `L-LDL-C` <dbl>, `L-LDL-CE` <dbl>,
## #   `L-LDL-FC` <dbl>, `L-LDL-TG` <dbl>, `M-LDL-P` <dbl>, `M-LDL-L` <dbl>,
## #   `M-LDL-PL` <dbl>, `M-LDL-C` <dbl>, `M-LDL-CE` <dbl>, `M-LDL-FC` <dbl>,
## #   `M-LDL-TG` <dbl>, `S-LDL-P` <dbl>, `S-LDL-L` <dbl>, `S-LDL-PL` <dbl>,
## #   `S-LDL-C` <dbl>, `S-LDL-CE` <dbl>, `S-LDL-FC` <dbl>, `S-LDL-TG` <dbl>,
## #   `XL-HDL-P` <dbl>, `XL-HDL-L` <dbl>, `XL-HDL-PL` <dbl>,
## #   `XL-HDL-C` <dbl>, `XL-HDL-CE` <dbl>, `XL-HDL-FC` <dbl>,
## #   `XL-HDL-TG` <dbl>, `L-HDL-P` <dbl>, `L-HDL-L` <dbl>, `L-HDL-PL` <dbl>,
## #   `L-HDL-C` <dbl>, `L-HDL-CE` <dbl>, `L-HDL-FC` <dbl>, `L-HDL-TG` <dbl>,
## #   `M-HDL-P` <dbl>, `M-HDL-L` <dbl>, `M-HDL-PL` <dbl>, `M-HDL-C` <dbl>,
## #   `M-HDL-CE` <dbl>, `M-HDL-FC` <dbl>, `M-HDL-TG` <dbl>, `S-HDL-P` <dbl>,
## #   `S-HDL-L` <dbl>, `S-HDL-PL` <dbl>, `S-HDL-C` <dbl>, `S-HDL-CE` <dbl>,
## #   `S-HDL-FC` <dbl>, `S-HDL-TG` <dbl>, `XXL-VLDL-PL_%` <dbl>,
## #   `XXL-VLDL-C_%` <dbl>, `XXL-VLDL-CE_%` <dbl>, `XXL-VLDL-FC_%` <dbl>,
## #   `XXL-VLDL-TG_%` <dbl>, `XL-VLDL-PL_%` <dbl>, ...
```

Print and inspect all the biomarker abbreviations

```
print(names(metabol_data_1))
```

```
##   [1] "sampleid"      "XXL-VLDL-P"    "XXL-VLDL-L"    "XXL-VLDL-PL"
##   [5] "XXL-VLDL-C"    "XXL-VLDL-CE"   "XXL-VLDL-FC"   "XXL-VLDL-TG"
##   [9] "XL-VLDL-P"     "XL-VLDL-L"     "XL-VLDL-PL"    "XL-VLDL-C"
##  [13] "XL-VLDL-CE"    "XL-VLDL-FC"    "XL-VLDL-TG"    "L-VLDL-P"
```

```
##  [17] "L-VLDL-L"      "L-VLDL-PL"     "L-VLDL-C"      "L-VLDL-CE"
##  [21] "L-VLDL-FC"     "L-VLDL-TG"     "M-VLDL-P"      "M-VLDL-L"
##  [25] "M-VLDL-PL"     "M-VLDL-C"      "M-VLDL-CE"     "M-VLDL-FC"
##  [29] "M-VLDL-TG"     "S-VLDL-P"      "S-VLDL-L"      "S-VLDL-PL"
##  [33] "S-VLDL-C"      "S-VLDL-CE"     "S-VLDL-FC"     "S-VLDL-TG"
##  [37] "XS-VLDL-P"     "XS-VLDL-L"     "XS-VLDL-PL"    "XS-VLDL-C"
##  [41] "XS-VLDL-CE"    "XS-VLDL-FC"    "XS-VLDL-TG"    "IDL-P"
##  [45] "IDL-L"         "IDL-PL"        "IDL-C"         "IDL-CE"
##  [49] "IDL-FC"        "IDL-TG"        "L-LDL-P"       "L-LDL-L"
##  [53] "L-LDL-PL"      "L-LDL-C"       "L-LDL-CE"      "L-LDL-FC"
##  [57] "L-LDL-TG"      "M-LDL-P"       "M-LDL-L"       "M-LDL-PL"
##  [61] "M-LDL-C"       "M-LDL-CE"      "M-LDL-FC"      "M-LDL-TG"
##  [65] "S-LDL-P"       "S-LDL-L"       "S-LDL-PL"      "S-LDL-C"
##  [69] "S-LDL-CE"      "S-LDL-FC"      "S-LDL-TG"      "XL-HDL-P"
##  [73] "XL-HDL-L"      "XL-HDL-PL"     "XL-HDL-C"      "XL-HDL-CE"
##  [77] "XL-HDL-FC"     "XL-HDL-TG"     "L-HDL-P"       "L-HDL-L"
##  [81] "L-HDL-PL"      "L-HDL-C"       "L-HDL-CE"      "L-HDL-FC"
##  [85] "L-HDL-TG"      "M-HDL-P"       "M-HDL-L"       "M-HDL-PL"
##  [89] "M-HDL-C"       "M-HDL-CE"      "M-HDL-FC"      "M-HDL-TG"
##  [93] "S-HDL-P"       "S-HDL-L"       "S-HDL-PL"      "S-HDL-C"
##  [97] "S-HDL-CE"      "S-HDL-FC"      "S-HDL-TG"      "XXL-VLDL-PL_%"
## [101] "XXL-VLDL-C_%"  "XXL-VLDL-CE_%" "XXL-VLDL-FC_%" "XXL-VLDL-TG_%"
## [105] "XL-VLDL-PL_%"  "XL-VLDL-C_%"   "XL-VLDL-CE_%"  "XL-VLDL-FC_%"
## [109] "XL-VLDL-TG_%"  "L-VLDL-PL_%"   "L-VLDL-C_%"    "L-VLDL-CE_%"
## [113] "L-VLDL-FC_%"   "L-VLDL-TG_%"   "M-VLDL-PL_%"   "M-VLDL-C_%"
## [117] "M-VLDL-CE_%"   "M-VLDL-FC_%"   "M-VLDL-TG_%"   "S-VLDL-PL_%"
## [121] "S-VLDL-C_%"    "S-VLDL-CE_%"   "S-VLDL-FC_%"   "S-VLDL-TG_%"
## [125] "XS-VLDL-PL_%"  "XS-VLDL-C_%"   "XS-VLDL-CE_%"  "XS-VLDL-FC_%"
## [129] "XS-VLDL-TG_%"  "IDL-PL_%"      "IDL-C_%"       "IDL-CE_%"
## [133] "IDL-FC_%"      "IDL-TG_%"      "L-LDL-PL_%"    "L-LDL-C_%"
## [137] "L-LDL-CE_%"    "L-LDL-FC_%"    "L-LDL-TG_%"    "M-LDL-PL_%"
## [141] "M-LDL-C_%"     "M-LDL-CE_%"    "M-LDL-FC_%"    "M-LDL-TG_%"
## [145] "S-LDL-PL_%"    "S-LDL-C_%"     "S-LDL-CE_%"    "S-LDL-FC_%"
## [149] "S-LDL-TG_%"    "XL-HDL-PL_%"   "XL-HDL-C_%"    "XL-HDL-CE_%"
## [153] "XL-HDL-FC_%"   "XL-HDL-TG_%"   "L-HDL-PL_%"    "L-HDL-C_%"
## [157] "L-HDL-CE_%"    "L-HDL-FC_%"    "L-HDL-TG_%"    "M-HDL-PL_%"
## [161] "M-HDL-C_%"     "M-HDL-CE_%"    "M-HDL-FC_%"    "M-HDL-TG_%"
## [165] "S-HDL-PL_%"    "S-HDL-C_%"     "S-HDL-CE_%"    "S-HDL-FC_%"
## [169] "S-HDL-TG_%"    "VLDL-D"        "LDL-D"         "HDL-D"
## [173] "Serum-C"       "VLDL-C"        "Remnant-C"     "LDL-C"
## [177] "HDL-C"         "HDL2-C"        "HDL3-C"        "EstC"
## [181] "FreeC"         "Serum-TG"      "VLDL-TG"       "LDL-TG"
## [185] "HDL-TG"        "TotPG"         "TG/PG"         "PC"
## [189] "SM"            "TotCho"        "ApoA1"         "ApoB"
## [193] "ApoB/ApoA1"    "TotFA"         "UnSat"         "DHA"
## [197] "LA"            "FAw3"          "FAw6"          "PUFA"
## [201] "MUFA"          "SFA"           "DHA/FA"        "LA/FA"
## [205] "FAw3/FA"       "FAw6/FA"       "PUFA/FA"       "MUFA/FA"
## [209] "SFA/FA"        "Glc"           "Lac"           "Pyr"
## [213] "Cit"           "Glol"          "Ala"           "Gln"
## [217] "Gly"           "His"           "Ile"           "Leu"
## [221] "Val"           "Phe"           "Tyr"           "Ace"
## [225] "AcAce"         "bOHBut"        "Crea"          "Alb"
## [229] "Gp"
```

```r
# You may also see their full names by printing the built-in dataset
head(forestplot::biomarkers)
```

```
## # A tibble: 6 x 4
##   abbrev fullname      forest_plot_disp_name forest_plot_categories
##   <chr>  <chr>         <chr>                 <chr>
## 1 Ile    Isoleucine    Isoleucine            Branched-chain amino acids
## 2 Leu    Leucine       Leucine               Branched-chain amino acids
## 3 Val    Valine        Valine                Branched-chain amino acids
## 4 Phe    Phenylalanine Phenylalanine         Aromatic amino acids
## 5 Tyr    Tyrosine      Tyrosine              Aromatic amino acids
## 6 His    Histidine     Histidine             Aromatic amino acids
```

Data analysis and visualization

Linear regression

We will now read the phenotype data that correspond to the blood biomarker data and join nmr with phenotype datasets.

```r
# Read the biomarker concentration file
pheno_data_1 <- read_tsv(file = "example_pheno_1.tsv",
                         col_types = cols(.default = col_double(),
                                          sampleid = col_character(),
                                          VMALE=col_factor(levels = c(0,1))))
pheno_data_2 <- read_tsv(file = "example_pheno_2.tsv",
                         col_types = cols(.default = col_double(),
                                          sampleid = col_character(),
                                          VMALE=col_factor(levels = c(0,1))))
metabol_n_pheno_1 <- left_join(x = metabol_data_1,
                               y = pheno_data_1,
                               by = "sampleid")
metabol_n_pheno_2 <- left_join(x = metabol_data_2,
                               y = pheno_data_2,
                               by = "sampleid")
```

We will now look at the association of BMI with each metabolite via linear regression:

$$y = \beta * x + \alpha$$

Where metabolite is the outcome , y, and BMI is the exposure x. The association of x with y refers to the beta coefficient (β).

```r
# We will use the built-in dataframe `biomarkers`, which contains all possible
# biomarkers of the NMR platform under discussion. (The dataframe contains
# standard abbreviation, spelled-out name and forestplot display name and one
# suggestive categorization of the biomarkers.)

# The biomarkers we will estimate the associations for are the intersection
# between the column biomarkers$abbrev and the column names of the metabol_n_pheno
# and the column
bmrs <-
```

```
  biomarkers$abbrev %>%
  intersect(., colnames(metabol_n_pheno_1)) %>%
  intersect(., colnames(metabol_n_pheno_2))

# No of bmr
nobmr <- length(bmrs)
```

```
# FIRST COHORT ANALYSIS
# Log transform the metabolites
metabol_n_pheno_1[,bmrs] <-
  metabol_n_pheno_1[,bmrs] %>%
  apply(2, log1p)

# Initialize data_frames that will store beta, SE and p-values
beta_cohort1 <-
  se_cohort1 <-
  pval_cohort1 <-
  data_frame(abbrev=character(nobmr),
             BMI=double(nobmr))

## Sex- and age-adjusted associations of cohort to BMI
cohort1 <-
  metabol_n_pheno_1 %>%
  gather(.,
         key = bmr,
         value = bmr_value,
         bmrs) %>%
  split(.$bmr) %>%
  map(~ lm(scale(bmr_value)  ~ BMI+VMALE+AGE, data = .)) %>%
  map(summary) %>%
  map("coefficients")

# Assign values to beta, se and pval dataframe
beta_cohort1$BMI <-
  cohort1 %>%
  sapply(., function(x) x["BMI","Estimate"])
beta_cohort1$abbrev <-
  cohort1 %>%
  names

se_cohort1$BMI <-
  cohort1 %>%
  sapply(., function(x) x["BMI","Std. Error"])
se_cohort1$abbrev <-
  cohort1 %>%
  names

pval_cohort1$BMI <-
  cohort1 %>%
  sapply(., function(x) x["BMI","Pr(>|t|)"])
pval_cohort1$abbrev <-
  cohort1 %>%
  names
```

```r
# SECOND COHORT ANALYSIS
# Log transform the metabolites
metabol_n_pheno_2[,bmrs] <-
  metabol_n_pheno_2[,bmrs] %>%
  apply(2, log1p)

# Initialize data_frames that will store beta, SE and p-values
beta_cohort2 <-
  se_cohort2 <-
  pval_cohort2 <-
  data_frame(abbrev=character(nobmr),
             BMI=double(nobmr))

## Sex- and age-adjusted associations of cohort to BMI
cohort2 <-
  metabol_n_pheno_2 %>%
  gather(.,
         key = bmr,
         value = bmr_value,
         bmrs) %>%
  split(.$bmr) %>%
  map(~ lm(scale(bmr_value)  ~ BMI+VMALE+AGE, data = .)) %>%
  map(summary) %>%
  map("coefficients")

# Assign values to beta, se and pval dataframe
beta_cohort2$BMI <-
  cohort2 %>%
  sapply(., function(x) x["BMI","Estimate"])
beta_cohort2$abbrev <-
  cohort2 %>%
  names

se_cohort2$BMI <-
  cohort2 %>%
  sapply(., function(x) x["BMI","Std. Error"])
se_cohort2$abbrev <-
  cohort2 %>%
  names

pval_cohort2$BMI <-
  cohort2 %>%
  sapply(., function(x) x["BMI","Pr(>|t|)"])
pval_cohort2$abbrev <-
  cohort2 %>%
  names
```

Forest plotting (linear)

Let's now attempt to plot all the biomarkers with the forestplot function.

```r
# We first merge the betas, se and pval dataframes
beta <- full_join(x = beta_cohort1,
```

```
                  y = beta_cohort2,
                  by = "abbrev")

se <- full_join(x = se_cohort1,
                y = se_cohort2,
                by = "abbrev")

pval <- full_join(x = pval_cohort1,
                  y = pval_cohort2,
                  by = "abbrev")

names(beta) <-
  names(se) <-
  names(pval) <-
  c("abbrev", "cohort1", "cohort2")

# Create a grouping (in this case we just use one of the built in options)
# Function bmr_selected_grouping() takes one argument "bmr_grouping_choice",
# which must be one of the following four strings: "onepage_forestplot",
# "serum_all", "edta_plasma_all", "short").
bmr_all_grouped <- bmr_selected_grouping(bmr_grouping_choice = "serum_all")

forestplot(beta=beta,
           se=se,
           pval=pval,
           biomarker_groups_as_list=bmr_all_grouped,
           filename='plot_linear_comparison.pdf',
           plot_title="Linear associations to BMI",
           is_log_odds_ratio=F,
           xlabel="SD difference (95% CI)",
           signif_cutoff=0.05,
           legend_vars=names(beta)[2:3],
           height = 12,
           width = 9)
```

```
## pdf
##   2
```

Notice that when only one cohort (or study) is plotted, the beta values are displayed on the right y-axis. For one-study plots the default color is black, but this is customizable, see `?forestplot` option `plotcolors`.

Notice also below that we are free to input other parameter specifications that will be fed into the `pdf` device used for the plot. In this case below I use "a4", which I highly recommend for the cases where all biomarkers are plotted.

```
forestplot(beta=beta[,1:2],
           se=se[,1:2],
           pval=pval[,1:2],
           biomarker_groups_as_list=bmr_all_grouped,
           filename='plot_linear_cohort1.pdf',
           plot_title="Linear associations to BMI",
           is_log_odds_ratio=F,
           xlabel="SD difference (95% CI)",
           signif_cutoff=0.05,
           paper="a4",
```

```
           height = 12,
           width = 9)
```

```
## pdf
##   2
```

Let's now plot a smalled group of biomarkers and see how we can customize the looks of the output. We can build our own grouping but we will start by using the existing example in `bmr_selected_grouping()` function.

```
# A grouping with smaller number of biomarkers
bmr_all_grouped <- bmr_selected_grouping(bmr_grouping_choice = "example_short")

forestplot(beta=beta,
           se=se,
           pval=pval,
           biomarker_groups_as_list=bmr_all_grouped,
           filename='plot_linear_cohort1_short.pdf',
           plot_title="Linear associations to BMI",
           is_log_odds_ratio=F,
           xlabel="SD difference (95% CI)",
           signif_cutoff=0.05,
           legend_vars=names(beta)[2:3],
           paper="a4",
           height = 12,
           width = 9)
```

```
## pdf
##   2
```

If you inspect the last pdf file, you'll notice that the layout needs some further customizing. Let's start by moving the y-axis labels closer to the

Logistic regression

We will perform a logistic regression with a dummy event in order to plot also with the `is_log_odds_ratio` flag on.

```
# Create a random event
metabol_n_pheno_1 <-
  metabol_n_pheno_1 %>%
  mutate(., event=BMI>30)

# Initialize data_frames that will store beta, SE and p-values
beta_cohort1 <-
  se_cohort1 <-
  pval_cohort1 <-
  data_frame(abbrev=character(nobmr),
             event=double(nobmr))

## Sex- and age-adjusted associations of cohort to event
cohort1 <-
  metabol_n_pheno_1 %>%
  gather(.,
         key = bmr,
```

```
          value = bmr_value,
          bmrs) %>%
  split(.$bmr) %>%
  map(~ glm(event ~ scale(bmr_value) + VMALE+AGE, data = ., family = "binomial")) %>%
  map(summary) %>%
  map("coefficients")

# Assign values to beta, se and pval dataframe
beta_cohort1$event <-
  cohort1 %>%
  sapply(., function(x) x["scale(bmr_value)","Estimate"])
beta_cohort1$abbrev <-
  cohort1 %>%
  names

se_cohort1$event <-
  cohort1 %>%
  sapply(., function(x) x["scale(bmr_value)","Std. Error"])
se_cohort1$abbrev <-
  cohort1 %>%
  names

pval_cohort1$event <-
  cohort1 %>%
  sapply(., function(x) x["scale(bmr_value)","Pr(>|z|)"])
pval_cohort1$abbrev <-
  cohort1 %>%
  names
```

Forest plotting (logarithmic)

```
# We first merge the betas, se and pval dataframes
beta <- beta_cohort1
se <- se_cohort1
pval <- pval_cohort1

names(beta) <-
  names(se) <-
  names(pval) <-
  c("abbrev", "event")

# Create a grouping (in this case we just use one of the built in options)
# Function bmr_selected_grouping() takes one argument "bmr_grouping_choice",
# which must be one of the following four strings: "onepage_forestplot",
# "serum_all", "edta_plasma_all", "short").
bmr_all_grouped <- bmr_selected_grouping(bmr_grouping_choice = "serum_all")

forestplot(beta=beta,
           se=se,
           pval=pval,
           biomarker_groups_as_list=bmr_all_grouped,
           filename='plot_logistic_cohort1.pdf',
```

```
            plot_title="Odds ratios",
            is_log_odds_ratio=T,
            xlabel="SD-scaled odds ratios (95% CI)",
            signif_cutoff=0.05,
            plotpointshape = 23,
            legend_vars=names(beta)[2],
            paper="a4",
            height = 12,
            width = 9)
```

```
## pdf
##   2
```

Below there is also an example with a small grouping that is built from scratch by the user. Here we will also make use of the `indices` parameter, a list of numeric vectors, that has either 1, 2 or 4 components corresponding to 1, 2 or 4 columns respectively and each containing the indices of the `biomarker_groups_as_list` to be plotted in each column.

```
# A grouping with smaller number of biomarkers
metabo_groups_custom <- list(`Branched-chain amino acids` = c("Ile",
    "Leu", "Val"), `Aromatic amino acids` = c("Phe",
    "Tyr", "His"), `Other amino acids` = c("Ala", "Gly",
    "Gln", NA), `Glycolysis-related metabolites` = c("Lac",
    "Pyr", "Glol"), `Ketone bodies` = c("AcAce", "bOHBut"),
    Inflammation = c("Gp"), Miscellaneous = c("Crea",
        "Alb", "Ace", "Cit", NA), `Fatty acid ratios` = c("TotFA",
        "SFA/FA", "MUFA/FA", "PUFA/FA", "FAw6/FA",
        "LA/FA", "FAw3/FA", "DHA/FA", "UnSat"), Phospholipids = c("SM",
        "TotCho", "PC", "TotPG", "TG/PG"))



forestplot(beta = beta, se = se, pval = pval, biomarker_groups_as_list = metabo_groups_custom,
    indices = list(c(1:22), c(23:44)), filename = "plot_logistic_cohort1_small.pdf",
    plot_title = "Odds ratios", is_log_odds_ratio = T,
    xlabel = "SD-scaled odds ratios (95% CI)", signif_cutoff = 0.05,
    width = 10, height = 5)
```

```
## pdf
##   2
```

So for example, in the `indices` parameter, indices 1 to 4 correspond to "Branched-chain amino acids", "Ile", "Leu", and "Val" from `metabo_groups_custom`. Mind that you would naturally prefer to equally sepqrate the biomarkers over the two columns, which is why here we plot the first 22 in the left column and the second 22 in the second column. However, equal number of biomarkers per column is not a requirement. Finally, notice that by adding NA values you can add empty space wherever you want in order to visually separate the groups or to better align the two columns.