Tutorial on R package **mtrank**

Theodoros Evrenoglou, Guido Schwarzer

This vignette serves as a tutorial for R package **mtrank**. This package allows R users to produce treatment hierarchies in network meta-analysis using the ranking method proposed by Evrenoglou et al. (2024).

You could either install R package **mtrank** from CRAN (not yet available)

```
install.packages("mtrank")
```

or the development version from GitHub

```
remotes::install_github("TEvrenoglou/mtrank")
```

Next, we make the package available.

```
library("mtrank")
## Loading required package: meta
## Loading required package: metadat
## Loading 'meta' package (version 8.0-2).
## Type 'help(meta)' for a brief overview.
## Loading required package: netmeta
## Loading 'netmeta' package (version 3.0-2).
## Type 'help("netmeta-package")' for a brief overview.
```

We see that loading **mtrank** will automatically load the R packages **meta**, **metadat** and **netmeta**.

Next, we load the 'antidepressants' dataset which is part of R package **mtrank**. You can get information about this dataset using the command *help(antidepressants)*.

```
data("antidepressants")
```

Define treatment-choice criterion

In the next step, we set the treatment-choice criterion and transform the data using the function tcc(). More details about this function can be obtained using the command *help(tcc)*.

The treatment-choice criterion is determined by the minimal clinically important difference (MCID), which represents the smallest relative effect between two treatments that can influence the selection of the preferable treatment. Users can set this value using argument 'mcid' in tcc(), which then defines the range of equivalence (ROE) based on the MCID and its reciprocal. For binary outcomes, the MCID must be specified on its natural scale. The ROE constructed using the MCID is always symmetrical, but for users who require a non-symmetrical ROE, the arguments 'lower.equi' and 'upper.equi' allow for explicit specification of preferred bounds. However, if argument 'mcid' is specified, arguments 'lower.equi' and 'upper.equi' are ignored.

```
ranks <- tcc(treat = drug_name, event = responders, n = ntotal,
  studlab = studyid, data = antidepressants,
  mcid = 1.25,
  sm = "OR", small.values = "undesirable")
```

We could print the preferences with the following command (result not shown).

```r
ranks$grouped.preferences
```

As argument ‘mcid = 1.25’ the ROE is defined as [1 / mcid, mcid] = [0.8, 1.25]. The ROE is stored as the lower and upper bound.

```r
c(ranks$lower.equi, ranks$upper.equi)
## [1] 0.80 1.25
```

Alternatively, we could define a ROE by specifying its lower and upper bound.

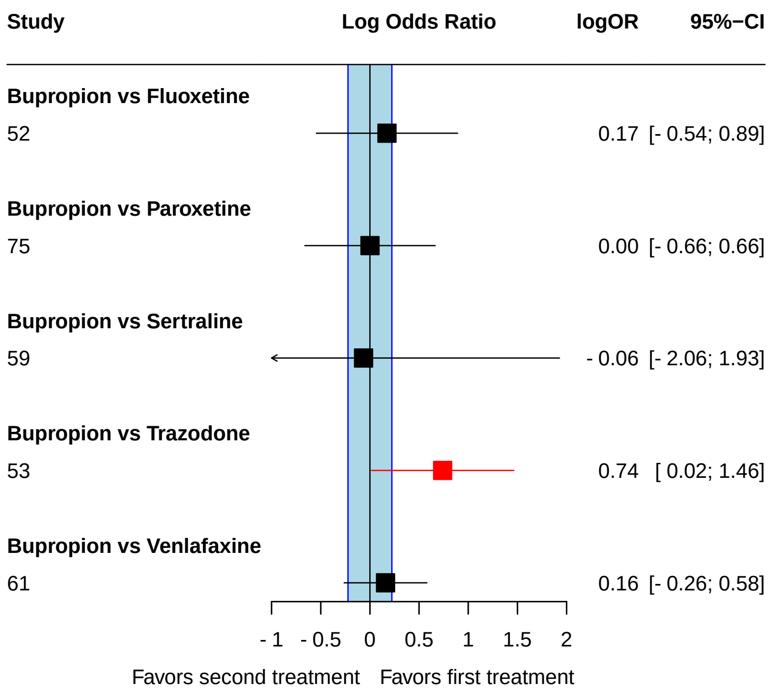
```r
ranks <- tcc(treat = drug_name, event = responders, n = ntotal,
  studlab = studyid, data = antidepressants,
  lower.equi = 0.80, upper.equi = 1.25,
  sm = "OR", small.values = "undesirable")
```

Note, an asymmetric ROE could be defined in this way.

Forest plot showing impact of treatment choice criterion

R function forest.tcc() can be used to create a forest plot. This function gets as first argument the object created from tcc() and as second the argument ‘treat’ which specifies the treatment for which the treatment choice criterion needs to be inspected. If users do not specify the argument ‘treat’ then forest plots are generated for every direct comparison in the network.

```r
forest(ranks, treat = "bupropion", xlim = c(-1, 2),
  label.left = "Favors second treatment",
  label.right = "Favors first treatment",
  fill.equi = "lightblue", spacing = 1.5)
```



Probabilistic ranking model

The probabilistic ranking model can be fitted using mtrank() from R package **mtrank**. The ability estimates obtained from mtrank() can then be visualised in a forest plot.

```
fit <- mtrank(ranks)
```

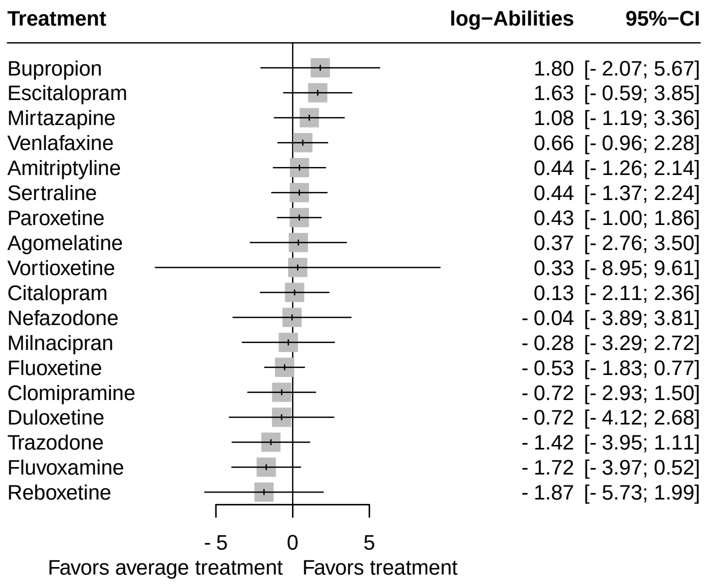
We can print the ability estimates and the probabilities of each treatment to rank first.

```
fit
##       treatment log_ability   lower  upper probability
##       Bupropion      1.7991 -2.0718 5.6700      0.2121
##    Escitalopram      1.6308 -0.5913 3.8529      0.1792
##     Mirtazapine      1.0843 -1.1948 3.3634      0.1038
##     Venlafaxine      0.6578 -0.9650 2.2806      0.0677
##   Amitriptyline      0.4431 -1.2585 2.1448      0.0546
##      Sertraline      0.4356 -1.3677 2.2389      0.0542
##      Paroxetine      0.4318 -0.9959 1.8595      0.0540
##     Agomelatine      0.3687 -2.7642 3.5016      0.0507
##    Vortioxetine      0.3271 -8.9513 9.6056      0.0487
##      Citalopram      0.1253 -2.1103 2.3609      0.0398
##      Nefazodone     -0.0406 -3.8877 3.8065      0.0337
##     Milnacipran     -0.2837 -3.2869 2.7196      0.0264
##      Fluoxetine     -0.5265 -1.8255 0.7725      0.0207
##    Clomipramine     -0.7167 -2.9293 1.4960      0.0171
##      Duloxetine     -0.7236 -4.1247 2.6776      0.0170
##       Trazodone     -1.4199 -3.9462 1.1064      0.0085
##     Fluvoxamine     -1.7235 -3.9664 0.5194      0.0063
##      Reboxetine     -1.8692 -5.7309 1.9924      0.0054
```

Forest plot of log-ability estimates

R function forest.mtrank() can be used to create a forest plot of ability estimates. By default, log-ability estimates are shown in the forest plot.

```
forest(fit)
```



Alternatively, we could plot the abilities using argument 'backtransf' (figure not shown).

```
forest(fit, backtransf = TRUE)
```

Pairwise preferences

Finally, R package **mtrank** allows the calculation of pairwise preferences through the function paired_pref(). This function expects the following arguments:

- x: an mtrank object
- treat1: the first treatment considered in the treatment comparison,
- treat2: the second treatment considered in the treatment comparison,
- type: the probability of interest.

For more details about this function please use *help(paired_pref)*.

```
# Get probability that bupropion is better or worse than trazodone
paired_pref(fit, treat1 = "bupropion", treat2 = "trazodone",
  type = c("better", "worse"))
## The probability that Bupropion is better than Trazodone is equal to: 0.6637
## The probability that Bupropion is worse than Trazodone is equal to: 0.0265

# Get probability that bupropion is tied with trazodone
paired_pref(fit, treat1 = "bupropion", treat2 = "trazodone",
  type = "tie")
## The probability that Bupropion is tied to Trazodone is equal to: 0.3097

# Get all three probabilities
paired_pref(fit, treat1 = "bupropion", treat2 = "trazodone",
  type = "all")
## The probability that Bupropion is better than Trazodone is equal to: 0.6637
## The probability that Bupropion is tied to Trazodone is equal to: 0.3097
## The probability that Bupropion is worse than Trazodone is equal to: 0.0265
```

It is also possible to contrast one drug with several others (and to provide abbreviated but unambiguous drug names).

```
# Get probability that bupropion is better than other drugs
paired_pref(fit, treat1 = "bupr",
  treat2 = c("fluo", "paro", "sert", "traz", "venl"), type = "better")
##      treat1      treat2 p_better
##   Bupropion  Fluoxetine   0.5473
##   Bupropion  Paroxetine   0.4111
##   Bupropion  Sertraline   0.4106
##   Bupropion   Trazodone   0.6637
##   Bupropion Venlafaxine   0.3791
```

References

Evrenoglou, Theodoros, Adriani Nikolakopoulou, Guido Schwarzer, Gerta Rücker, and Anna Chaimani. 2024. "Producing Treatment Hierarchies in Network Meta-Analysis Using Probabilistic Models and Treatment-Choice Criteria." https://arxiv.org/abs/2406.10612.