# Anchored binary analysis

2024-09-30

## Loading R packages

```r
# install.packages("maicplus")
library(maicplus)
```

Additional R packages for this vignette:

```r
library(dplyr)
```

## Illustration using example data

This example reads in `centered_ipd_twt` data that was created in `calculating_weights` vignette and uses `adrs_twt` dataset to run binary outcome analysis using the `maic_anchored` function by specifying `endpoint_type = "binary"`.

```r
data(centered_ipd_twt)
data(adrs_twt)

centered_colnames <- c("AGE", "AGE_SQUARED", "SEX_MALE", "ECOG0", "SMOKE", "N_PR_THER_MEDIAN")
centered_colnames <- paste0(centered_colnames, "_CENTERED")

weighted_data <- estimate_weights(
  data = centered_ipd_twt,
  centered_colnames = centered_colnames
)

# get dummy binary IPD
pseudo_adrs <- get_pseudo_ipd_binary(
  binary_agd = data.frame(
    ARM = c("B", "C", "B", "C"),
    RESPONSE = c("YES", "YES", "NO", "NO"),
    COUNT = c(280, 120, 200, 200)
  ),
  format = "stacked"
)

result <- maic_anchored(
  weights_object = weighted_data,
  ipd = adrs_twt,
  pseudo_ipd = pseudo_adrs,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  normalize_weight = FALSE,
  endpoint_type = "binary",
  endpoint_name = "Binary Endpoint",
  eff_measure = "OR",
  # binary specific args
  binary_robust_cov_type = "HC3"
)
```

There are two summaries available in the result: descriptive and inferential. In the descriptive section, we have summaries of events.

```
result$descriptive
```

```
## $summary
##   trt_ind treatment                  type   n   events events_pct
## 1       C         C IPD, before matching 500 338.0000   67.60000
## 2       A         A IPD, before matching 500 390.0000   78.00000
## 3       C         C  IPD, after matching 500 131.2892   26.25784
## 4       A         A  IPD, after matching 500 142.8968   28.57935
## 5       C         C        AgD, external 320 120.0000   37.50000
## 6       B         B        AgD, external 480 280.0000   58.33333
```

In the inferential section, we have the fitted models stored (i.e. logistic regression) and the results from the `glm` models (i.e. odds ratios and CI).

```
result$inferential$summary
```

```
##          case        OR       LCL       UCL         pval
## 1          AC 1.3119021 0.8210000 2.0963303 2.562849e-01
## 2 adjusted_AC 1.6993007 1.2809976 2.2541985 2.354448e-04
## 3          BC 2.3333333 1.7458092 3.1185794 1.035032e-08
## 4          AB 0.5622438 0.3239933 0.9756933 4.061296e-02
## 5 adjusted_AB 0.7282717 0.4857575 1.0918611 1.248769e-01
```

Here are model and results before adjustment.

```
result$inferential$fit$model_before
```

```
## 
## Call:  glm(formula = RESPONSE ~ ARM, family = glm_link, data = ipd)
## 
## Coefficients:
## (Intercept)         ARMA  
##      0.7354       0.5302  
## 
## Degrees of Freedom: 999 Total (i.e. Null);  998 Residual
## Null Deviance:       1170 
## Residual Deviance: 1157  AIC: 1161
```

```
result$inferential$fit$res_AC_unadj
```

```
## $est
## [1] 1.699301
##
## $se
## [1] 0.2488482
##
## $ci_l
## [1] 1.280998
##
## $ci_u
## [1] 2.254199
##
## $pval
## [1] 0.0002354448
```

```
result$inferential$fit$res_AB_unadj
```

```
##              result             pvalue
## "0.73[0.49; 1.09]"            "0.125"
```

Here are model and results after adjustment.

```
result$inferential$fit$model_after
```

```
##
## Call:  glm(formula = RESPONSE ~ ARM, family = glm_link, data = ipd,
##     weights = weights)
##
## Coefficients:
## (Intercept)         ARMA
##      0.6559       0.2715
##
## Degrees of Freedom: 999 Total (i.e. Null);  998 Residual
## Null Deviance:       495.5
## Residual Deviance: 493.9     AIC: 454.5
```

```
result$inferential$fit$res_AC
```

```
## $est
## [1] 1.311902
##
## $se
## [1] 0.3275028
##
## $ci_l
## [1] 0.821
##
## $ci_u
## [1] 2.09633
```

```
##
## $pval
## [1] 0.2562849
```

```
result$inferential$fit$res_AB
```

```
##             result             pvalue
## "0.56[0.32; 0.98]"            "0.041"
```

## Using bootstrap to calculate standard errors

If bootstrap standard errors are preferred, we need to specify the number of bootstrap iteration (`n_boot_iteration`) in `estimate_weights` function and proceed fitting `maic_anchored` function. Then, the outputs include bootstrapped CI. Different types of bootstrap CI can be found by using parameter `boot_ci_type`.

```
weighted_data2 <- estimate_weights(
  data = centered_ipd_twt,
  centered_colnames = centered_colnames,
  n_boot_iteration = 100,
  set_seed_boot = 1234
)

result_boot <- maic_anchored(
  weights_object = weighted_data2,
  ipd = adrs_twt,
  pseudo_ipd = pseudo_adrs,
  trt_ipd = "A",
  trt_agd = "B",
  trt_common = "C",
  normalize_weight = FALSE,
  endpoint_type = "binary",
  endpoint_name = "Binary Endpoint",
  eff_measure = "OR",
  boot_ci_type = "perc",
  # binary specific args
  binary_robust_cov_type = "HC3"
)

result_boot$inferential$fit$boot_res_AB
```

```
## $est
## [1] 0.1272177
##
## $se
## [1] NA
##
## $ci_l
## [1] 0.1315433
##
## $ci_u
## [1] 0.3386533
##
## $pval
## [1] NA
```