Practical: Population-Adjusted Indirect Comparisons with `outstandR`

University College London (UCL)

2025-05-19

Introduction

This practical session investigates the use of population-adjusted indirect treatment comparisons (ITCs).

When we want to compare two treatments, say A and B, we ideally use a head-to-head randomized controlled trial (RCT). However, such trials are not always available. Instead, we might have:

1. An RCT comparing A to a common comparator C (the AC trial), for which we have **Individual Patient Data (IPD)**.
2. An RCT comparing B to the same common comparator C (the BC trial), for which we only have **Aggregate Level Data (ALD)**, like summary statistics from a publication.

If the patient populations in the AC and BC trials differ in characteristics that modify the treatment effect (effect modifiers), a simple indirect comparison (A vs C minus B vs C) can be misleading. **Population adjustment methods** aim to correct for these differences, providing a more valid comparison of A vs B in a chosen target population (often the population of the BC trial).

We will use our `{outstandR}` R package which provides a suite of tools to perform these adjustments. In this practical, we will:

1. Simulate IPD and ALD for both binary and continuous outcomes.
2. Use `{outstandR}` to apply methods like Matching-Adjusted Indirect Comparison (MAIC) and G-computation.
3. Explore how to change the outcome scale for reporting.
4. Interpret the basic output from `{outstandR}`.

Learning Objectives: By the end of this practical, you will be able to:

- Understand the scenario requiring population adjustment.
- Prepare IPD and ALD in the format required by `{outstandR}`.
- Apply MAIC and G-computation methods for binary and continuous outcomes.
- Interpret and report results on different scales.

Part 0: Setup and Package Loading

First, we need to load the necessary R packages. If you haven't installed them, you'll need to do so. We have created the `simcovariates` package to use here for data generation which you'll need to install from GitHub. The `outstandR` package will also need to be installed.

```
# Ensure packages are installed:
#
# install.packages(c("dplyr", "tidyr", "boot", "copula", "rstanarm", "remotes"))
#
# remotes::install_github("n8thangreen/simcovariates") # For gen_data
# remotes::install_github("StatisticsHealthEconomics/outstandR")

library(outstandR)
library(simcovariates) # For gen_data
library(dplyr)
library(tidyr)
# library(rstanarm) # Loaded by outstandR if/when needed for Bayesian G-comp
# library(boot)     # Loaded by outstandR if/when needed for MAIC

# For reproducibility of simulated data
set.seed(123)
```

Part 1: Data Simulation & Preparation - Binary Outcomes

We'll start with a scenario involving a binary outcome (e.g., treatment response: yes/no).

1.1 Simulation Parameters

We use parameters similar to the `{outstandR}` vignette to define our simulation. These control sample size, treatment effects, covariate effects, and population characteristics.

```r
N <- 200              # Sample size per trial

# Active treatment vs. placebo allocation ratio (2:1 implies ~2/3 on active)
allocation <- 2/3

# Conditional log-OR for active treatment vs. common comparator C
b_trt <- log(0.17)

# Conditional log-OR for each unit increase in prognostic variables (X3, X4)
b_X <- -log(0.5)

# Conditional log-OR for interaction term (treatment * effect modifier) for X1, X2
b_EM <- -log(0.67)

# Mean of prognostic factors (X3, X4) in AC trial
meanX_AC <- c(0.45, 0.45)

# Mean of prognostic factors (X3, X4) in BC trial (DIFFERENT from AC)
meanX_BC <- c(0.6, 0.6)
meanX_EM_AC <- c(0.45, 0.45) # Mean of effect modifiers (X1, X2) in AC trial

# Mean of effect modifiers (X1, X2) in BC trial (DIFFERENT from AC)
meanX_EM_BC <- c(0.6, 0.6)
sdX <- c(0.4, 0.4)           # Standard deviation of prognostic factors
sdX_EM <- c(0.4, 0.4)        # Standard deviation of effect modifiers
corX <- 0.2                  # Covariate correlation coefficient
b_0 <- -0.6                  # Baseline intercept coefficient on logit scale
```

> **Note**
>
> **Effect Modifiers vs. Prognostic Variables:**
> - **Prognostic variables** (`X3`, `X4` here) predict the outcome regardless of treatment. - **Effect modifiers** (`X1`, `X2` here) change the magnitude or direction of the treatment effect. Differences in the distribution of effect modifiers between trials are a key reason for population adjustment.

1.2 Generate IPD for AC Trial (Binary Outcome)

We simulate Individual Patient Data (IPD) for a trial comparing treatments A and C.

```r
ipd_trial_bin <- gen_data(N,
                          b_trt,
                          b_X,
                          b_EM,
                          b_0 = b_0,
                          meanX_AC,
                          sdX,
                          meanX_EM_AC,
                          sdX_EM,
                          corX,
                          allocation,
                          family = binomial("logit"))

# Treatment 'trt' is 0 or 1
# We map 0 to 'C' (comparator) and 1 to 'A' (new treatment)
ipd_trial_bin$trt <- factor(ipd_trial_bin$trt, labels = c("C", "A"))
```

Lets look at the generated data.

```r
head(ipd_trial_bin)
```

```
            X1         X2         X3         X4 trt y
1  0.420906647  0.6501898  0.6817174 0.61434770   A 0
2  0.009771062  1.0476893  0.9321016 0.04336125   A 0
3  0.086942077 -0.4289788  0.3807218 0.18401299   A 0
4 -0.039661515  0.7256527  0.4987618 0.54389751   A 1
5  0.585786267  0.2143042  0.2207665 0.64831303   A 0
6  0.600816955 -0.3921163 -0.3156147 0.17139023   A 0
```

```r
summary(ipd_trial_bin)
```

```
       X1                X2                X3                X4
 Min.   :-0.7022   Min.   :-0.7504   Min.   :-0.7311   Min.   :-0.6038
 1st Qu.: 0.1864   1st Qu.: 0.2158   1st Qu.: 0.1819   1st Qu.: 0.1820
 Median : 0.4603   Median : 0.5231   Median : 0.4557   Median : 0.4032
 Mean   : 0.4636   Mean   : 0.4723   Mean   : 0.4390   Mean   : 0.4289
 3rd Qu.: 0.7043   3rd Qu.: 0.7245   3rd Qu.: 0.7171   3rd Qu.: 0.6886
 Max.   : 1.5292   Max.   : 1.5804   Max.   : 1.6463   Max.   : 1.3328
 trt           y
 C: 67   Min.   :0.00
```

```
 A:133   1st Qu.:0.00
         Median :0.00
         Mean   :0.32
         3rd Qu.:1.00
         Max.   :1.00
```

The `ipd_trial_bin` dataframe contains patient-level data: covariates (`X1-X4`), treatment assignment (`trt`), and outcome (`y`).

1.3 Generate ALD for BC Trial (Binary Outcome)

For the BC trial (comparing B vs C), we only have Aggregate Level Data (ALD). We first simulate IPD for BC and then summarize it. The key here is that `meanX_BC` and `meanX_EM_BC` are different from the AC trial, creating a population imbalance.

```r
# Simulate IPD for BC trial (using BC trial's covariate means)
BC_IPD_bin <- gen_data(N,
                       b_trt,
                       b_X,
                       b_EM,
                       b_0,
                       meanX_BC, # Using BC means
                       sdX,
                       meanX_EM_BC, # Using BC means
                       sdX_EM,
                       corX,
                       allocation,
                       family = binomial("logit"))

BC_IPD_bin$trt <- factor(BC_IPD_bin$trt, labels = c("C", "B")) # 0=C, 1=B

# Now, aggregate BC_IPD_bin to create ald_trial_bin
# This mimics having only published summary statistics.

# Covariate summaries (mean, sd for X1-X4,
# assumed same across arms in BC trial for simplicity)
cov_summary_bin <- BC_IPD_bin %>%
  select(X1, X2, X3, X4) %>% # Select covariate columns
  summarise(across(everything(), list(mean = mean, sd = sd))) %>%
  pivot_longer(everything(), names_to = "stat_var", values_to = "value") %>%
  # 'stat_var' will be like "X1_mean", "X1_sd". We need to separate these.
```

```
  separate(stat_var, into = c("variable", "statistic"), sep = "_") %>%
  # Covariate summaries are often reported for the overall trial population
  mutate(trt = NA_character_)

# Outcome summaries (number of events 'sum',
# mean proportion 'mean', sample size 'N' for y by trt)
outcome_summary_bin <- BC_IPD_bin %>%
  group_by(trt) %>%
  summarise(
    sum_y = sum(y),      # Number of events
    mean_y = mean(y),    # Proportion of events
    N = n()              # Sample size in this arm
  ) %>%
  ungroup() %>%
  pivot_longer(cols = -trt, names_to = "stat_var", values_to = "value") %>%
  # 'stat_var' will be "sum_y", "mean_y", "N". We need to parse this.
  mutate(
    variable = case_when(
      grepl("_y$", stat_var) ~ "y", # If it ends with _y, variable is y
      stat_var == "N" ~ NA_character_, # For N, variable can be NA
      TRUE ~ stat_var # Default
    ),
    statistic = case_when(
      grepl("sum_", stat_var) ~ "sum",
      grepl("mean_", stat_var) ~ "mean",
      stat_var == "N" ~ "N",
      TRUE ~ stat_var # Default
    )
  ) %>%
  select(variable, statistic, value, trt)


# Combine covariate and outcome summaries for the final ALD structure
ald_trial_bin <- bind_rows(cov_summary_bin, outcome_summary_bin) %>%
  select(variable, statistic, value, trt)
```

Viewing the data,

```
print(as.data.frame(ald_trial_bin))
```

```
   variable statistic       value  trt
```

```
1         X1      mean   0.5961081 <NA>
2         X1        sd   0.4015645 <NA>
3         X2      mean   0.5779233 <NA>
4         X2        sd   0.3895705 <NA>
5         X3      mean   0.5799632 <NA>
6         X3        sd   0.3981054 <NA>
7         X4      mean   0.5944841 <NA>
8         X4        sd   0.4316603 <NA>
9          y       sum  28.0000000    C
10         y      mean   0.4179104    C
11      <NA>         N  67.0000000    C
12         y       sum  32.0000000    B
13         y      mean   0.2406015    B
14      <NA>         N 133.0000000    B
```

The `ald_trial_bin` is in a 'long' format with columns: `variable` (e.g., "X1", "y"), `statistic` (e.g., "mean", "sd", "sum", "N"), `value`, and `trt` (treatment arm, or NA if overall). This is the format `{outstandR}` expects.

Part 2: Model Fitting - Binary Outcomes

Now we use `{outstandR}` to perform population adjustments. We'll compare treatment A (from AC trial IPD) with treatment B (from BC trial ALD), using C as the common anchor. The target population for comparison will be the BC trial population.

2.1 Define the Model Formula

The model formula specifies the relationship between the outcome (`y`), prognostic variables (`X3`, `X4`), treatment (`trt`), and effect modifiers (`X1`, `X2`). For a binary outcome with a logit link, the model is:

$$\text{logit}(p_t) = \beta_0 + \beta_X(X_3 + X_4) + [\beta_t + \beta_{EM}(X_1 + X_2)]\ \text{I}(t \neq C)$$

This translates to the R formula: `y ~ X3 + X4 + trt + trt:X1 + trt:X2` (The intercept β_0 is implicit).

```r
lin_form_bin <- as.formula("y ~ X3 + X4 + trt + trt:X1 + trt:X2")
```

2.2 Matching-Adjusted Indirect Comparison (MAIC)

MAIC reweights the IPD from the AC trial so that the mean covariate values of the effect modifiers match those of the BC trial population.

```
# MAIC involves bootstrapping, which can take a moment.
# The number of bootstrap replicates can sometimes be
# controlled in strategy_maic() for speed,
# e.g. n_boot = 100 for a quick check, but higher
# (e.g., 1000) is better for stable results.
# We'll use the default for now.

out_maic_bin <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_maic(
    formula = lin_form_bin,
    family = binomial(link = "logit")
    # If your package allows, you might add:
    # , n_boot = 200 # for faster demo
  )
)
```

The MAIC results (default: Log-Odds Ratio scale):

```
print(out_maic_bin)
```

```
Object of class 'outstandR'
Model: binomial
Scale: log_odds
Common treatment: C
Individual patient data study: AC
Aggregate level data study: BC
Confidence interval level: 0.95

Contrasts:

# A tibble: 3 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl>      <dbl>      <dbl>
1 AB          -0.0495     0.226     -0.981      0.882
2 AC          -0.868      0.123     -1.56      -0.179
```

```
3 BC          -0.818      0.103     -1.45      -0.191

Absolute:

# A tibble: 2 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl> <lgl>      <lgl>
1 A             0.266   0.00183 NA         NA
2 C             0.461   0.00431 NA         NA
```

The output provides `contrasts` (e.g., A vs B) and `absolute_effects` in the target (BC) population. By default, for `binomial(link="logit")`, the effect measure is the log-odds ratio.

2.3 Changing the Outcome Scale (MAIC Example)

Often, we want results on a different scale, like log-relative risk or risk difference. The `scale` argument in `outstandR()` allows this.

```r
out_maic_bin_lrr <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_maic(
    formula = lin_form_bin,
    family = binomial(link = "logit")
  ),
  scale = "log_relative_risk" # Key change!
)
```

The MAIC results on the log-relative risk scale,

```r
print(out_maic_bin_lrr)
```

```
Object of class 'outstandR'
Model: binomial
Scale: log_relative_risk
Common treatment: C
Individual patient data study: AC
Aggregate level data study: BC
Confidence interval level: 0.95
```

```
Contrasts:

# A tibble: 3 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl>      <dbl>      <dbl>
1 AB         -0.00612    0.0951     -0.610      0.598
2 AC         -0.558      0.0505     -0.999     -0.118
3 BC         -0.552      0.0445     -0.966     -0.139

Absolute:

# A tibble: 2 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl> <lgl>      <lgl>
1 A             0.264   0.00166 NA         NA
2 C             0.460   0.00456 NA         NA
```

> **Tip**
>
> **Your Turn!** Try getting MAIC results on the **risk difference** scale.
> Hint: `scale = "risk_difference"`.

```r
out_maic_bin_rd <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_maic(
    formula = lin_form_bin,
    family = binomial(link = "logit")
  ),
  scale = "risk_difference" # Key change!
)
```

The MAIC results on the risk difference scale,

```r
print(out_maic_bin_rd)
```

```
Object of class 'outstandR'
Model: binomial
Scale: risk_difference
Common treatment: C
Individual patient data study: AC
```

```
Aggregate level data study: BC
Confidence interval level: 0.95

Contrasts:

# A tibble: 3 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl>      <dbl>      <dbl>
1 AB          -0.0165   0.433       -1.31      1.27
2 AC          -0.194    0.00684     -0.356    -0.0317
3 BC          -0.177    0.426       -1.46      1.10

Absolute:

# A tibble: 2 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl> <lgl>      <lgl>
1 A             0.266   0.00192 NA         NA
2 C             0.460   0.00486 NA         NA
```

2.4 Parametric G-computation with Maximum Likelihood (G-comp ML)

G-computation fits an outcome regression model to the IPD (AC trial) and then uses this model to predict outcomes for each patient *as if* they had received treatment A and *as if* they had received treatment C, but standardized to the covariate distribution of the target (BC) population.

```r
out_gcomp_ml_bin <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_gcomp_ml(
    formula = lin_form_bin,
    family = binomial(link = "logit")
  )
)
```

```r
print(out_gcomp_ml_bin)
```

```
Object of class 'outstandR'
Model: binomial
Scale: log_odds
```

```
Common treatment: C
Individual patient data study: AC
Aggregate level data study: BC
Confidence interval level: 0.95

Contrasts:

# A tibble: 3 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl>      <dbl>      <dbl>
1 AB          -0.0774     0.230      -1.02      0.863
2 AC          -0.895      0.128      -1.60     -0.195
3 BC          -0.818      0.103      -1.45     -0.191

Absolute:

# A tibble: 2 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl> <lgl>      <lgl>
1 A             0.273   0.00224 NA         NA
2 C             0.477   0.00449 NA         NA
```

Part 3: Adapting for Continuous Outcomes

What if our outcome is continuous, like change in blood pressure or a quality-of-life score? The principles are similar, but we need to adjust the data generation and model specification.

3.1 Simulate Continuous Data

We'll use `family = gaussian("identity")` for the `gen_data` function. We might also adjust some coefficients to be more sensible for a continuous scale.

```r
# Adjust some parameters for a continuous outcome
b_0_cont <- 5      # Intercept on the continuous scale
b_trt_cont <- -1.5 # Mean difference for treatment A vs C
b_X_cont <- 0.5    # Effect of prognostic vars on continuous outcome

# Effect of effect modifiers on treatment effect (continuous)
b_EM_cont <- 0.3
```

3.1.1 IPD for AC Trial (Continuous)

```
ipd_trial_cont <- gen_data(N,
                           b_trt_cont,
                           b_X_cont,
                           b_EM_cont,
                           b_0_cont,
                           meanX_AC,
                           sdX,
                           meanX_EM_AC,
                           sdX_EM,
                           corX,
                           allocation,
                           family = gaussian("identity")) # Key change!

ipd_trial_cont$trt <- factor(ipd_trial_cont$trt, labels = c("C", "A"))

head(ipd_trial_cont)
```

```
          X1          X2         X3         X4 trt        y
1  0.3347920  1.25173997  0.8251443  0.3626829   A 5.287769
2  1.3518916  0.34102429  0.7105816  0.2199213   A 4.530242
3  0.8390412  0.15676647  0.6917419  0.3092831   A 3.491232
4  0.8418207 -0.17637220 -0.2343619 -0.5699550   A 5.199625
5  0.5758347  0.06594836  0.2838801  0.1641963   A 4.366868
6 -0.3415192  0.58383236  0.4612317  0.1143282   A 3.391340
```

```
summary(ipd_trial_cont$y)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.827   3.717   4.536   4.503   5.267   8.399
```

3.1.2 ALD for BC Trial (Continuous)

```
BC_IPD_cont <- gen_data(N,
                        b_trt_cont,
                        b_X_cont,
                        b_EM_cont,
                        b_0_cont,
                        meanX_BC, # Using BC means
```

```
                        sdX,
                        meanX_EM_BC, # Using BC means
                        sdX_EM,
                        corX,
                        allocation,
                        family = gaussian("identity")) # Key change!

BC_IPD_cont$trt <- factor(BC_IPD_cont$trt, labels = c("C", "B"))

# Aggregate BC_IPD_cont for ALD
# Covariate summaries structure remains the same
cov_summary_cont <- BC_IPD_cont %>%
  select(X1, X2, X3, X4) %>%
  summarise(across(everything(), list(mean = mean, sd = sd))) %>%
  pivot_longer(everything(), names_to = "stat_var", values_to = "value") %>%
  separate(stat_var, into = c("variable", "statistic"), sep = "_") %>%
  mutate(trt = NA_character_)

# Outcome summaries for continuous data: mean, sd, N for y by trt
outcome_summary_cont <- BC_IPD_cont %>%
  group_by(trt) %>%
  summarise(
    mean_y = mean(y),    # Mean outcome
    sd_y = sd(y),        # Standard deviation of outcome
    N = n()              # Sample size
  ) %>%
  ungroup() %>%
  pivot_longer(cols = -trt, names_to = "stat_var", values_to = "value") %>%
  mutate(
    variable = case_when(
      grepl("_y$", stat_var) ~ "y",
      stat_var == "N" ~ NA_character_,
      TRUE ~ stat_var
    ),
    statistic = case_when(
      grepl("mean_", stat_var) ~ "mean",
      grepl("sd_", stat_var) ~ "sd", # Changed from sum to sd
      stat_var == "N" ~ "N",
      TRUE ~ stat_var
    )
  ) %>%
  select(variable, statistic, value, trt)
```

```r
ald_trial_cont <- bind_rows(cov_summary_cont, outcome_summary_cont) %>%
  select(variable, statistic, value, trt)

print(as.data.frame(ald_trial_cont))
```

```
   variable statistic       value  trt
1        X1      mean   0.5941535 <NA>
2        X1        sd   0.3856699 <NA>
3        X2      mean   0.5695096 <NA>
4        X2        sd   0.4234775 <NA>
5        X3      mean   0.5642288 <NA>
6        X3        sd   0.3971081 <NA>
7        X4      mean   0.5739154 <NA>
8        X4        sd   0.3957993 <NA>
9         y      mean   5.3318057    C
10        y        sd   0.8647773    C
11     <NA>         N  67.0000000    C
12        y      mean   4.5914634    B
13        y        sd   0.9994386    B
14     <NA>         N 133.0000000    B
```

3.2 Model Fitting for Continuous Outcomes

The model formula structure can remain the same if we assume linear relationships. The key change is in the `family` argument of the strategy function.

```r
lin_form_cont <- as.formula("y ~ X3 + X4 + trt + trt:X1 + trt:X2")
```

Let's use G-computation ML as an example.

```r
out_gcomp_ml_cont <- outstandR(
  ipd_trial = ipd_trial_cont,
  ald_trial = ald_trial_cont,
  strategy = strategy_gcomp_ml(
    formula = lin_form_cont,
    family = gaussian(link = "identity") # Key change!
  )
  # For Gaussian family, the default scale is typically
  # "mean_difference", # which is often what we want.
  # We could explicitly state: scale = "mean_difference"
)
```

```r
print(out_gcomp_ml_cont)
```

```
Object of class 'outstandR'
Model: gaussian
Scale: mean_difference
Common treatment: C
Individual patient data study: AC
Aggregate level data study: BC
Confidence interval level: 0.95

Contrasts:

# A tibble: 3 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl>      <dbl>      <dbl>
1 AB           -0.557    0.0472     -0.982     -0.131
2 AC           -1.30     0.0285     -1.63      -0.966
3 BC           -0.740    0.0187     -1.01      -0.473

Absolute:

# A tibble: 2 x 5
  Treatments Estimate Std.Error lower.0.95 upper.0.95
  <chr>         <dbl>     <dbl> <lgl>      <lgl>
1 A              4.28   0.00703 NA         NA
2 C              5.58   0.0215  NA         NA
```

Tip

Your Turn! Try applying **MAIC** to the continuous outcome data. 1. Use `family = gaussian(link = "identity")` within `strategy_maic()`. 2. What `scale` would be appropriate if not the default? (e.g., `"mean_difference"`)

```r
# Solution for MAIC with continuous data:
out_maic_cont <- outstandR(
  ipd_trial = ipd_trial_cont,
  ald_trial = ald_trial_cont,
  strategy = strategy_maic(
    formula = lin_form_cont,
    family = gaussian(link = "identity")
  ),
  scale = "mean_difference"
)
print(out_maic_cont)
```

2.5 Other Methods

`{outstandR}` supports other methods. Here's how you might call them. These are set to `eval=FALSE` to save time in this practical.

- **Simulated Treatment Comparison (STC):** A conventional outcome regression approach.

```r
out_stc_bin <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_stc(
    formula = lin_form_bin,
    family = binomial(link = "logit")
  )
)
print(out_stc_bin)
```

- **Bayesian G-computation (G-comp Bayes):** Similar to G-comp ML but uses Bayesian methods (e.g., MCMC via `rstanarm`), which can better propagate uncertainty but is computationally more intensive.

```r
# This would require rstanarm and can be slow.
out_gcomp_stan_bin <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_gcomp_stan(
    formula = lin_form_bin,
    family = binomial(link = "logit")
```

```
    # For a faster demo if options are passed through:
    # stan_args = list(iter = 500, chains = 2, refresh = 0)
  )
)
print(out_gcomp_stan_bin)
```

- **Multiple Imputation Marginalisation (MIM):** Another approach for marginalization.

```
out_mim_bin <- outstandR(
  ipd_trial = ipd_trial_bin,
  ald_trial = ald_trial_bin,
  strategy = strategy_mim(
    formula = lin_form_bin,
    family = binomial(link = "logit")
  )
)
print(out_mim_bin)
```

Part 4: Understanding Output & Wrap-up

Let's briefly revisit one of the binary outcome results to understand the structure of the {outstandR} output.

```
str(out_maic_bin)
```

```
List of 2
 $ contrasts:List of 3
  ..$ means    :List of 3
  .. ..$ AB: num -0.0495
  .. ..$ AC: num -0.868
  .. ..$ BC: num -0.818
  ..$ variances:List of 3
  .. ..$ AB: num 0.226
  .. ..$ AC: num 0.123
  .. ..$ BC: num 0.103
  ..$ CI       :List of 3
  .. ..$ AB: num [1:2] -0.981 0.882
  .. ..$ AC: num [1:2] -1.556 -0.179
  .. ..$ BC: num [1:2] -1.446 -0.191
 $ absolute :List of 2
```

```
  ..$ means    :List of 2
  .. ..$ A: Named num 0.266
  .. .. ..- attr(*, "names")= chr "mean_A"
  .. ..$ C: Named num 0.461
  .. .. ..- attr(*, "names")= chr "mean_C"
  ..$ variances:List of 2
  .. ..$ A: Named num 0.00183
  .. .. ..- attr(*, "names")= chr "mean_A"
  .. ..$ C: Named num 0.00431
  .. .. ..- attr(*, "names")= chr "mean_C"
 - attr(*, "CI")= num 0.95
 - attr(*, "ref_trt")= chr "C"
 - attr(*, "scale")= chr "log_odds"
 - attr(*, "model")= chr "binomial"
 - attr(*, "class")= chr [1:2] "outstandR" "list"
```

The output object (here `out_maic_bin`) is a list containing:

- `$contrasts`: This list provides the estimated treatment effects (e.g., mean difference, log-OR), their variances, and confidence intervals for each pairwise comparison, adjusted to the target population (BC trial).
- `$contrasts$means$AB`: The estimated effect of A versus B. This is often the primary interest.
- `$contrasts$means$AC`: The estimated effect of A versus C.
- `$contrasts$means$BC`: The estimated effect of B versus C (usually derived directly from the ALD).
- `$absolute_effects`: This list provides the estimated mean outcome for each treatment (A, B, C) in the target population. This can be useful for understanding the baseline and treated outcomes.

For example, to extract the estimated log-odds ratio for A vs. B and its variance:

```
log_or_AB <- out_maic_bin$contrasts$means$AB
variance_log_or_AB <- out_maic_bin$contrasts$variances$AB

cat(paste("Estimated Log-OR for A vs. B:", round(log_or_AB, 3), "\n"))
```

```
Estimated Log-OR for A vs. B: -0.05
```

```
cat(paste("Variance of Log-OR for A vs. B:", round(variance_log_or_AB, 3), "\n"))
```

```
Variance of Log-OR for A vs. B: 0.226
```

The vignette for `{outstandR}` (which this practical is based on) shows how to combine results from multiple methods into tables and forest plots for a comprehensive comparison. This is highly recommended for actual analyses.

Key Takeaways

- Population adjustment is crucial when comparing treatments indirectly using IPD and ALD from trials with different patient characteristics (especially different distributions of effect modifiers).
- The `{outstandR}` package provides a unified interface (`outstandR()` function) to apply various adjustment methods.
- You need to:
    1. Prepare your IPD (for the "anchor" trial, e.g., AC) and ALD (for the "comparator" trial, e.g., BC, which also serves as the target population).
    2. Define an appropriate model `formula`.
    3. Choose a `strategy_*()` function corresponding to the desired adjustment method (MAIC, STC, G-comp, etc.).
    4. Specify the outcome `family` (e.g., `binomial()`, `gaussian()`) within the strategy.
    5. Optionally, use the `scale` argument in `outstandR()` to transform results to a desired effect measure scale.
- The methods can be adapted for different outcome types (binary, continuous, count, time-to-event, though we only covered binary and continuous here).