Comparative Effectiveness and Personalized Medicine Research Using Real-World Data

Thomas P. A. Debray, Tri-Long Nguyen, and Robert W. Platt

5/3/23

Table of contents

1 Preface

Thomas Debray (Smart Data Analysis and Statistics B.V.)

This book provides practical guidance for estimating the effectiveness of treatments in real-world populations. It explains how real-world data can directly be used or combined with other data sources to derive overall and individualized estimates of treatment effect. The book explains statistical methods for implementing bias adjustments, conducting evidence synthesis and individualizing treatment effect, whilst also providing illustrative examples and supporting software. The chapters and contents of the book are written by leading experts, with a track record in the generation and/or evaluation of real-world evidence.

This book is intended as a pivotal textbook for statisticians, epidemiologists, methodologists, regulators and/or regulatory scientists considering, undertaking or appraising the real-world evidence of treatment effectiveness. It covers key concepts and stages to derive and evaluate treatment effect estimates for entire populations and specific individuals. The book offers a conceptual framework towards estimating treatment effects at both the population and individualized level, where modelling methods may include traditional regression-based and machine learning methods.

Motivation

Although randomized clinical trials traditionally form the cornerstone of comparative effectiveness research, there is a growing demand to consider evidence from "real-world data" (RWD) in clinical decision-making. These data are often available from observational cohort studies, administrative databases, and patient registries, and may offer additional insights into the comparative effectiveness and safety of treatments. Yet, the analysis of RWD and the evaluation of real-world evidence face many operational and methodological challenges.

In this book, we aim to address three current needs. First, this book will offer the guidance that is currently lacking on assessing the quality of RWD and on implementing appropriate statistical methods to reduce bias of single study estimates of treatment effects. Second, this book will provide researchers with advanced approaches to pooling estimates from multiple non-randomized studies for which traditional evidence synthesis methods are not suitable.

Finally, to answer the growing need to translate average estimates of treatment effects to individualized clinical decision-making, this book will present recent methods for more tailored approaches where patient characteristics are used to derive their individualized prognosis and treatment benefit.

This book aims to explain key principles and state-of-the-art methods for deriving treatment effects in entire populations and specific individuals using RWD. It will not only discuss statistical theory by key experts in the field; it will also provide illustrative examples and practical guidance for implementation in R. In short, the book aims to prepare a new generation of researchers who wish to generate and integrate evidence from both randomized and non-randomized data sources to investigate the real-world effectiveness of treatments in populations and individual patients.

Contents

The book is divided into six sections:

1. **Introduction**. This section introduces the relevance of real-world data for conducting comparative effectiveness research, and discusses various concerns regarding their use.
2. **Principles of treatment effect estimation using real-world data**. In this section, we discuss key principles of treatment effect estimation in non-randomized data sources. We explain methods to adjust for confounding (including propensity score analysis and disease risk score analysis) and missing data when estimating the treatment effect for a specific (sub)population.
3. **Principles of evidence synthesis**. In this section, we discuss statistical methods for estimating the treatment effect using (individual participant and/or aggregate) data from multiple studies. To this purpose, key principles of meta-analysis are introduced and explained, including the standard fixed effect and random effects meta-analysis models, methods for individual patient data (IPD) meta-analysis, methods for network meta-analysis, and methods for data-driven and tailored bias adjustment.
4. **Advanced modelling issues for dealing with additional bias in both randomized and non-randomized data sources**. In this section, we discuss advanced statistical and machine learning methods for dealing with time-varying confounding, informative visit schedules, and measurement error.
5. **Individualizing treatment effects for personalized medicine**. In this section, we discuss statistical methods to estimate and evaluate individualized treatment effects.
6. Closing

2 Validity control and quality assessment of real-world data and real-world evidence

Christina ReadThomas Debray (Smart Data Analysis and Statistics B.V.)

```
library(readxl)
library(robvis)
```

The quality of real-world data is often suboptimal and can therefore lead to bias when generating real-world evidence (RWE). In this chapter, we will introduce key quality concerns of RWD, including their accuracy, completeness, and timeliness. Subsequently, we will discuss which steps can be taken to assess the quality of RWD, and determine their fitness for use. The chapter will also introduce directed acyclic graphs to explain how the analysis of RWD may be affected by different types of bias. We will put particular focus on confounding bias, selection bias, and information bias, and explain how these biases can be addressed by referring to specific chapters from the book. Finally, the chapter presents common quality appraisal tools that can be used to assess the quality of real-world evidence (for instance when conducting a systematic review).

A risk of bias assessment was conducted in the COVID-NMA review. We can create a summary table of risk of bias assessment and produce a traffic light plot as follows:

```
Risk_of_Bias <- read_excel("resources/RoB-covid.xlsx")

#creation of traffic light plot
trafficlight_rob <- rob_traffic_light(data = Risk_of_Bias, tool = "ROB2")
trafficlight_rob
```

Risk of bias domains

Study	D1	D2	D3	D4	D5	Overall
Abd−Elsalam et al	+	X	+	+	−	X
Ader	+	+	+	+	+	+
Beigel (ACTT−1)	+	+	+	+	+	+
Cao et al	+	+	+	−	+	−
Cavalcanti et al	+	−	+	−	+	−
Criner*	−	−	+	+	−	−
Davoudi−Monfared	+	+	+	X	+	X
Horby (RECOVERY; DEXA)	+	+	+	+	+	+
Horby (RECOVERY; LOPI)	+	+	+	+	+	+
Horby (RECOVERY; TOCI)	+	+	+	+	+	+
Islam^	+	−	+	+	+	−
Lyngbakken et al	+	+	+	X	+	X
Mahajan	−	−	−	+	−	−
Pan (WHO Solidarity; HCQ)	+	−	+	+	+	−
Pan (WHO Solidarity; INTB)	+	−	+	+	+	−
Pan (WHO Solidarity; LOPI)	+	−	+	+	+	−
Pan (WHO Solidarity; REM)	+	+	+	+	+	+
Spinner	+	+	+	+	−	−
Wang	+	+	+	−	+	+
Broman	+	+	+	+	−	−
Declerq	+	+	+	+	+	+
Gordon (REMAP−CAP)	+	−	+	+	+	+
Hermine	+	+	−	+	+	+
Rohmani	+	X	+	−	+	X
Rosas (COVACTA)	+	+	+	+	+	+
Rutgers	+	+	+	+	+	+
Salama(EMPACTA)	+	+	+	+	+	+
Salvarani*	+	X	+	+	+	+
Soin (COVINTOC)	+	−	+	+	+	+
Stone*	−	+	+	+	+	+
Tomazini	+	+	+	+	+	+
Talaschian	−	−	−	+	−	−
Ulrich et al	+	+	X	+	−	X
Veiga	+	−	+	+	+	+

Domains:
D1: Bias arising from the randomization process.
D2: Bias due to deviations from intended intervention.
D3: Bias due to missing outcome data.
D4: Bias in measurement of the outcome.
D5: Bias in selection of the reported result.

Judgement
X High
− Some concerns
\+ Low

Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)

Matrix products: default

locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] robvis_0.3.0.900 readxl_1.4.2

loaded via a namespace (and not attached):
 [1] digest_0.6.31    cellranger_1.1.0 jsonlite_1.8.4   magrittr_2.0.3
 [5] evaluate_0.21    rlang_1.1.0      cli_3.6.1        rstudioapi_0.14
 [9] rmarkdown_2.21   tools_4.2.3      xfun_0.39        yaml_2.3.7
[13] fastmap_1.1.1    compiler_4.2.3   htmltools_0.5.5  knitr_1.42
```

3 Confounding adjustment using propensity score methods

Tammy Jiang (Biogen)
Thomas Debray (Smart Data Analysis and Statistics B.V.)

The purpose of this document is to provide example R code that demonstrates how to estimate the propensity score and implement matching, stratification, weighting, and regression adjustment for the continuous propensity score. In this example using simulated data, we have two disease modifying therapies (DMT1 and DMT0) and the outcome is the number of post-treatment multiple sclerosis relapses during follow-up. We will estimate the average treatment effect in the treated (ATT) using propensity score matching, stratification, and weighting. We will estimate the average treatment effect in the population (ATE) using regression adjustment for the continuous propensity score. The treatment effects can be interpreted as annualized relapse rate ratios (ARR).

We consider an example dataset with the following characteristics:

```r
head(dat)
```

```
   age female prevDMTefficacy premedicalcost numSymptoms prerelapse_num
1:  50      1            None        3899.61           1              1
2:  51      0            None        9580.51           1              0
3:  56      0            None        4785.89           1              0
4:  44      1            None        8696.80           1              1
5:  63      0            None        2588.03           1              0
6:  28      1            None        5435.57           1              0
   treatment y      years      Iscore
1:      DMT1 0 1.78507871 Moderate A1
2:      DMT1 0 0.01368925     High A1
3:      DMT1 2 3.25530459     High A1
4:      DMT1 2 5.73853525     Neutral
5:      DMT1 0 1.31143053     High A1
6:      DMT1 0 0.59137577 Moderate A0
```

3.1 Comparing baseline characteristics

- `DMT1` is the treatment group and `DMT0` is the control group
- `prevDMTefficacy` is previous DMT efficacy (none, low efficacy, and medium/high efficacy)
- `prerelapse_num` is the number of previous MS relapses

	DMT0	DMT1
n	2300	7700
age (mean (SD))	51.39 (8.32)	44.25 (9.79)
female = 1 (%)	1671 (72.65)	5915 (76.82)
prevDMTefficacy (%)		
None	1247 (54.22)	3171 (41.18)
Low_efficacy	261 (11.35)	858 (11.14)
Medium_high_efficacy	792 (34.43)	3671 (47.68)
prerelapse_num (mean (SD))	0.39 (0.62)	0.46 (0.68)

3.2 Estimating the propensity score

3.2.1 Logistic regression

We sought to restore balance in the distribution of baseline covariates in patients treated with DMT1 (index treatment) and DMT0 (control tratment). We fit a multivariable logistic regression model in which treatment was regressed on baseline characteristics including age, sex, previous DMT efficacy, and previous number of relapses.

```r
# Fit logistic regression model
ps.model <- glm(treatment ~ age + female + prevDMTefficacy + prerelapse_num,
                data = dat, family = binomial())

# Summary of logistic regression model
summary(ps.model)
```

```
Call:
glm(formula = treatment ~ age + female + prevDMTefficacy + prerelapse_num,
    family = binomial(), data = dat)

Deviance Residuals:
```

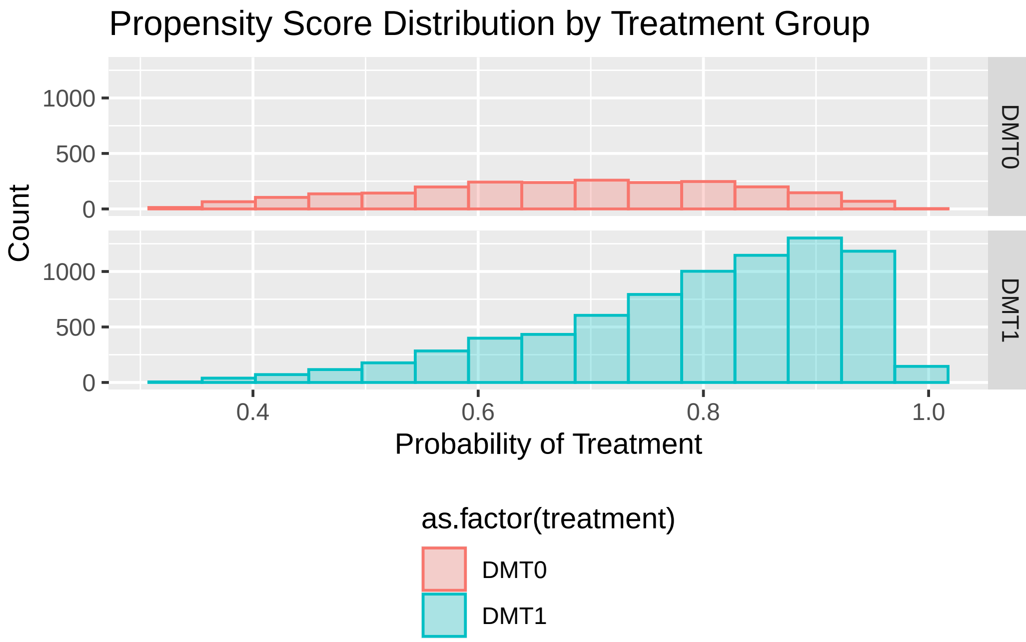
```
    Min       1Q   Median       3Q      Max
-2.7949   0.2585   0.5220   0.7478   1.5033

Coefficients:
                                     Estimate Std. Error z value Pr(>|z|)
(Intercept)                          4.809473   0.157127  30.609  < 2e-16 ***
age                                 -0.086708   0.002996 -28.939  < 2e-16 ***
female1                              0.253611   0.057664   4.398 1.09e-05 ***
prevDMTefficacyLow_efficacy          0.310394   0.083022   3.739 0.000185 ***
prevDMTefficacyMedium_high_efficacy  0.660266   0.054393  12.139  < 2e-16 ***
prerelapse_num                       0.156318   0.039288   3.979 6.93e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 10786  on 9999  degrees of freedom
Residual deviance:  9597  on 9994  degrees of freedom
AIC: 9609

Number of Fisher Scoring iterations: 5
```

```r
# Extract propensity scores
dat$ps <- predict(ps.model, data = dat, type = "response")
```

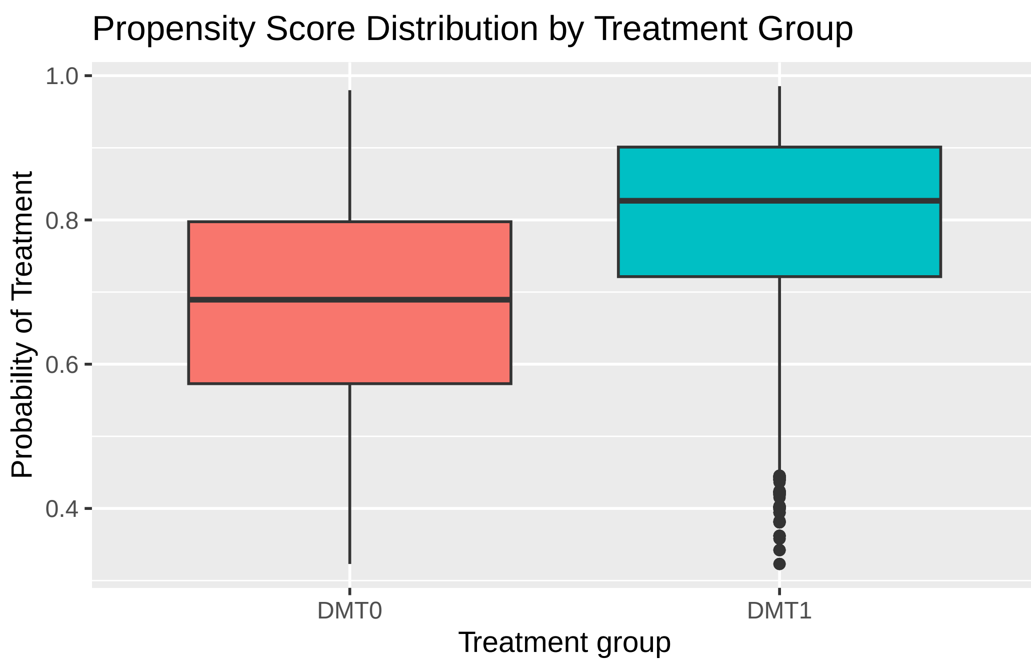
3.2.2 Assessing overlap

We examined the degree of overlap in the distribution of propensity scores across treatment groups using histograms and side-by-side box plots.

```r
# Histogram
ggplot(dat, aes(x = ps, fill = as.factor(treatment), color = as.factor(treatment))) +
  geom_histogram(alpha = 0.3, position='identity', bins = 15) +
  facet_grid(as.factor(treatment) ~ .) +
  xlab("Probability of Treatment") +
  ylab("Count") +
  ggtitle("Propensity Score Distribution by Treatment Group") +
  theme(legend.position = "bottom", legend.direction = "vertical")
```




```
# Side-by-side box plots
ggplot(dat, aes(x=as.factor(treatment), y=ps, fill=as.factor(treatment))) +
  geom_boxplot() +
  ggtitle("Propensity Score Distribution by Treatment Group") +
  ylab("Probability of Treatment") +
  xlab("Treatment group") +
  theme(legend.position = "none")
```




```r
# Distribution of propensity scores by treatment groups
summary(dat$ps[dat$treatment == "DMT1"])
```

```
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.3230  0.7214  0.8265  0.7970  0.9010  0.9854
```

```r
summary(dat$ps[dat$treatment == "DMT0"])
```

```
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.3230  0.5730  0.6894  0.6795  0.7975  0.9799
```

3.3 Propensity score matching

3.3.1 1:1 Optimal full matching without replacement

```r
library(MatchIt)

# Use MatchIt package for PS matching
opt <- matchit(treatment ~ age + female + prevDMTefficacy + prerelapse_num,
               data = dat,
               method = "full",
               estimand = "ATT")

opt
```

```
A matchit object
 - method: Optimal full matching
 - distance: Propensity score
             - estimated with logistic regression
 - number of obs.: 10000 (original), 10000 (matched)
 - target estimand: ATT
 - covariates: age, female, prevDMTefficacy, prerelapse_num
```

3.3.2 Assess balance after matching

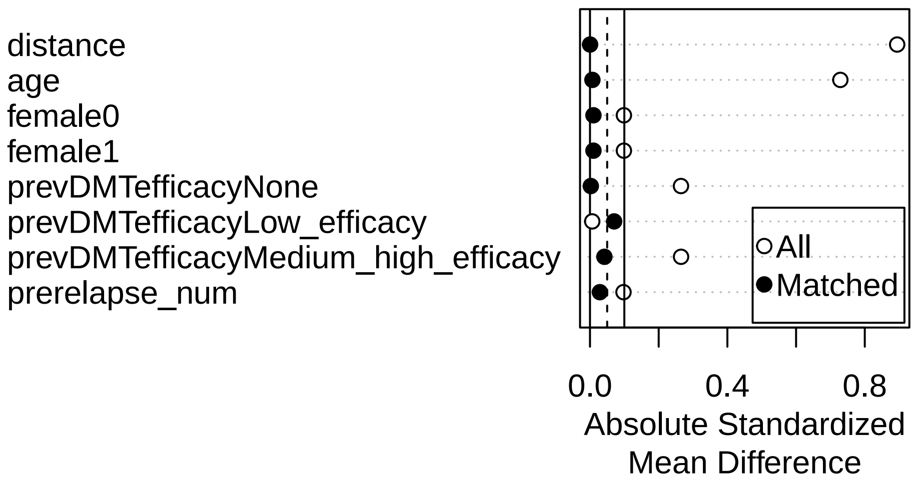
```r
summary(opt)
```

```
Call:
matchit(formula = treatment ~ age + female + prevDMTefficacy +
    prerelapse_num, data = dat, method = "full", estimand = "ATT")

Summary of Balance for All Data:
                           Means Treated Means Control Std. Mean Diff.
distance                          0.7970        0.6795          0.8943
age                              44.2496       51.3883         -0.7289
female0                           0.2318        0.2735         -0.0987
female1                           0.7682        0.7265          0.0987
prevDMTefficacyNone               0.4118        0.5422         -0.2649
prevDMTefficacyLow_efficacy       0.1114        0.1135         -0.0065
```

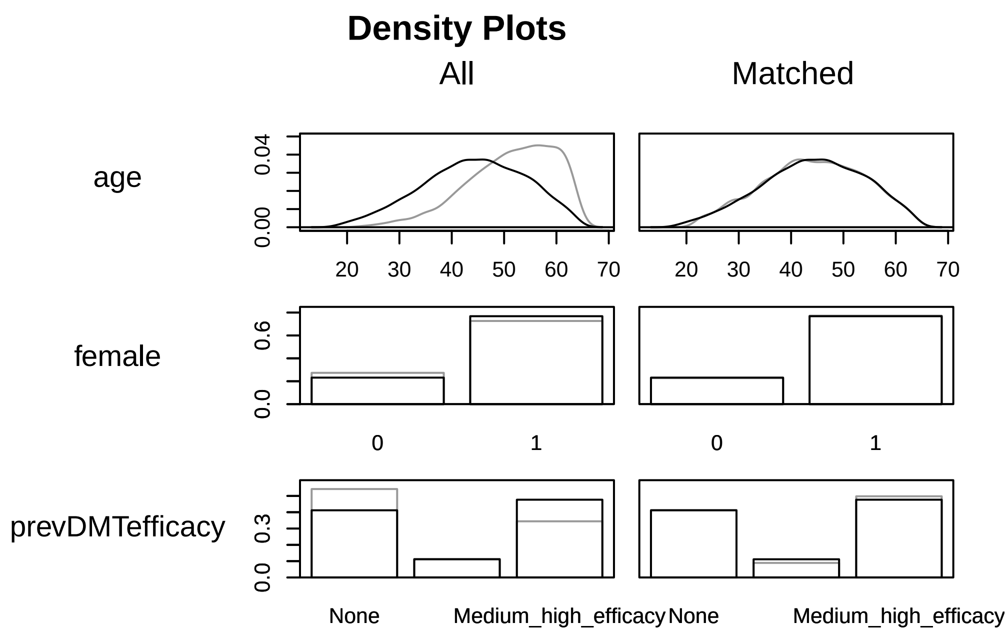
```
prevDMTefficacyMedium_high_efficacy        0.4768        0.3443          0.2651
prerelapse_num                             0.4595        0.3930          0.0976
                                    Var. Ratio eCDF Mean eCDF Max
distance                                0.7873    0.1917   0.3379
age                                     1.3868    0.1519   0.3085
female0                                      .    0.0417   0.0417
female1                                      .    0.0417   0.0417
prevDMTefficacyNone                          .    0.1304   0.1304
prevDMTefficacyLow_efficacy                  .    0.0020   0.0020
prevDMTefficacyMedium_high_efficacy          .    0.1324   0.1324
prerelapse_num                          1.1990    0.0133   0.0383

Summary of Balance for Matched Data:
                                    Means Treated Means Control Std. Mean Diff.
distance                                   0.7970        0.7970          0.0003
age                                       44.2496       44.3185         -0.0070
female0                                    0.2318        0.2275          0.0101
female1                                    0.7682        0.7725         -0.0101
prevDMTefficacyNone                        0.4118        0.4130         -0.0024
prevDMTefficacyLow_efficacy                0.1114        0.0893          0.0703
prevDMTefficacyMedium_high_efficacy        0.4768        0.4977         -0.0419
prerelapse_num                             0.4595        0.4399          0.0288
                                    Var. Ratio eCDF Mean eCDF Max
distance                                0.9976    0.0005   0.0075
age                                     1.0392    0.0038   0.0153
female0                                      .    0.0043   0.0043
female1                                      .    0.0043   0.0043
prevDMTefficacyNone                          .    0.0012   0.0012
prevDMTefficacyLow_efficacy                  .    0.0221   0.0221
prevDMTefficacyMedium_high_efficacy          .    0.0209   0.0209
prerelapse_num                          1.1319    0.0060   0.0229
                                    Std. Pair Dist.
distance                                     0.0008
age                                          0.0667
female0                                      0.1775
female1                                      0.1775
prevDMTefficacyNone                          0.1100
prevDMTefficacyLow_efficacy                  0.1846
prevDMTefficacyMedium_high_efficacy          0.1614
prerelapse_num                               0.2170

Sample Sizes:
              Control Treated
```

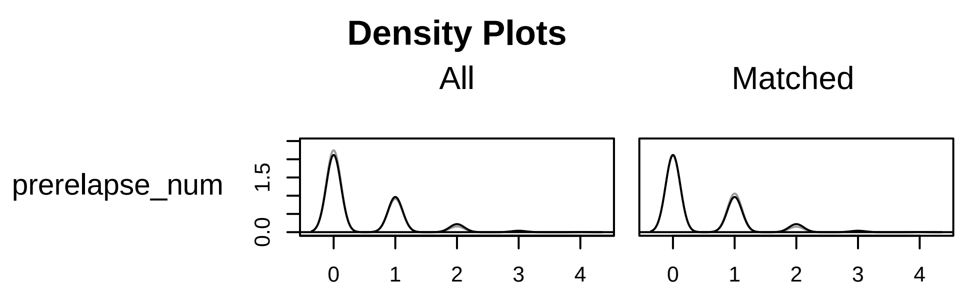
```
All            2300.      7700
Matched (ESS)   307.06    7700
Matched        2300.      7700
Unmatched         0.         0
Discarded         0.         0
```

```r
plot(summary(opt))
```




```r
# black line is treated group, grey line is control group
plot(opt, type = "density", which.xs = vars)
```


Density Plots
All
Matched
age
0.00
0.04
20 30 40 50 60 70
female
0.0
0.6
0 1
prevDMTefficacy
0.0
0.3
None
Medium_high_efficacy


Density Plots
All
Matched
prerelapse_num
0.0
1.5
0 1 2 3 4

3.3.3 Estimating the ATT

We can estimate the ATT in the matched sample using Poisson regression in which the number of post-treatment relapses is regressed on treatment status and follow-up time for each patient (captured by the variable `years`). More details are provided at https://cran.r-project.org/web/packages/MatchIt/vignettes/estimating-effects.html.

```
# Matched data
matched.data <- match.data(opt)

# Poisson regression model
opt.fit <- glm(y ~ treatment + offset(log(years)),
            family = poisson(link = "log"),
            data = matched.data,
            weights = weights)

# Treatment effect estimation
opt.comp <- comparisons(opt.fit,
                        variables = "treatment",
                        vcov = ~subclass,
                        newdata = subset(matched.data, treatment == "DMT1"),
                        wts = "weights",
                        transform_pre = "ratio")

opt.comp |> tidy()
```

```
# A tibble: 1 x 8
  term      contrast    estimate std.error statistic  p.value conf.low conf.high
  <chr>     <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 treatment mean(DMT1)~    0.804     0.102      7.88 3.25e-15    0.604      1.00
```

As indicated in the summary output above, the annualized relapse rate ratio for DMT1 vs DMT0 among patients treated with DMT0 (ATT) is given as 0.8 with a 95% confidence interval ranging from 0.6 to 1.

3.4 Propensity score stratification

3.4.1 Divide sample into quintiles of propensity scores

We will form five mutually exclusive groups of the estimated propensity score.

```
# Create five strata
dat <- dat %>% mutate(ps.strata = cut(ps,
                                      breaks = c(quantile(ps, probs=seq(0,1,0.2))),
                                      labels = seq(1:5),
                                      include.lowest = TRUE))

# Number of patients in each stratum
table(dat$ps.strata)
```

```
   1    2    3    4    5
2002 2015 1991 1997 1995
```

3.4.2 Assess balance within each propensity score stratum

Within each propensity score stratum, treated and control patients should have similar values of the propensity score and the distribution of baseline covariates should be approximately balanced between treatment groups.

3.4.2.1 Propensity Score Stratum #1

```
tab1.strata1 <- CreateTableOne(vars, data = dat %>% filter(ps.strata == 1),
                               factorVars = c("female", "prevDMTefficacy"),
                               strata = "treatment", test = FALSE)

tab1.strata1.print <- print(tab1.strata1, catDigits = 2, contDigits = 2,
                            smd = TRUE)
```

	DMT0	DMT1	SMD
n	901	1101	
age (mean (SD))	58.38 (3.67)	57.45 (3.73)	0.251
female = 1 (%)	605 (67.15)	775 (70.39)	0.070
prevDMTefficacy (%)			0.056
None	650 (72.14)	771 (70.03)	
Low_efficacy	106 (11.76)	130 (11.81)	
Medium_high_efficacy	145 (16.09)	200 (18.17)	
prerelapse_num (mean (SD))	0.29 (0.53)	0.33 (0.56)	0.074

3.4.2.2 Propensity Score Stratum #2

```
tab1.strata2 <- CreateTableOne(vars, data = dat %>% filter(ps.strata == 2),
                               factorVars = c("female", "prevDMTefficacy"),
                               strata = "treatment", test = FALSE)

tab1.strata2.print <- print(tab1.strata2, catDigits = 2, contDigits = 2,
                            smd = TRUE)
```

	DMT0	DMT1	SMD
n	617	1398	
age (mean (SD))	52.18 (4.35)	51.97 (4.22)	0.049
female = 1 (%)	458 (74.23)	1048 (74.96)	0.017
prevDMTefficacy (%)			0.054
None	292 (47.33)	624 (44.64)	
Low_efficacy	69 (11.18)	162 (11.59)	
Medium_high_efficacy	256 (41.49)	612 (43.78)	
prerelapse_num (mean (SD))	0.40 (0.64)	0.41 (0.66)	0.004

3.4.2.3 Propensity Score Stratum #3

```
tab1.strata3 <- CreateTableOne(vars, data = dat %>% filter(ps.strata == 3),
                               factorVars = c("female", "prevDMTefficacy"),
                               strata = "treatment", test = FALSE)

tab1.strata3.print <- print(tab1.strata3, catDigits = 2, contDigits = 2,
                            smd = TRUE)
```

	DMT0	DMT1	SMD
n	392	1599	
age (mean (SD))	46.73 (4.06)	46.36 (4.08)	0.092
female = 1 (%)	305 (77.81)	1193 (74.61)	0.075
prevDMTefficacy (%)			0.041
None	168 (42.86)	687 (42.96)	
Low_efficacy	52 (13.27)	191 (11.94)	
Medium_high_efficacy	172 (43.88)	721 (45.09)	
prerelapse_num (mean (SD))	0.49 (0.68)	0.47 (0.66)	0.031

3.4.2.4 Propensity Score Stratum #4

```
tab1.strata4 <- CreateTableOne(vars, data = dat %>% filter(ps.strata == 4),
                               factorVars = c("female", "prevDMTefficacy"),
                               strata = "treatment", test = FALSE)

tab1.strata4.print <- print(tab1.strata4, catDigits = 2, contDigits = 2,
                            smd = TRUE)
```

	DMT0	DMT1	SMD
n	269	1728	
age (mean (SD))	41.07 (4.11)	40.88 (4.29)	0.046
female = 1 (%)	203 (75.46)	1356 (78.47)	0.071
prevDMTefficacy (%)			0.084
None	105 (39.03)	634 (36.69)	
Low_efficacy	22 (8.18)	181 (10.47)	
Medium_high_efficacy	142 (52.79)	913 (52.84)	
prerelapse_num (mean (SD))	0.50 (0.69)	0.51 (0.71)	0.012

3.4.2.5 Propensity Score Stratum #5

```
tab1.strata5 <- CreateTableOne(vars, data = dat %>% filter(ps.strata == 5),
                               factorVars = c("female", "prevDMTefficacy"),
                               strata = "treatment", test = FALSE)

tab1.strata5.print <- print(tab1.strata5, catDigits = 2, contDigits = 2,
                            smd = TRUE)
```

	DMT0	DMT1	SMD
n	121	1874	
age (mean (SD))	33.26 (4.95)	32.04 (5.58)	0.233
female = 1 (%)	100 (82.64)	1543 (82.34)	0.008
prevDMTefficacy (%)			0.050
None	32 (26.45)	455 (24.28)	
Low_efficacy	12 (9.92)	194 (10.35)	
Medium_high_efficacy	77 (63.64)	1225 (65.37)	
prerelapse_num (mean (SD))	0.52 (0.66)	0.52 (0.73)	0.004

3.4.3 Estimating and pooling of stratum-specific treatment effects

The overall ATT across strata can be estimated by weighting stratum-specific estimates by the proportion of treated patients in each stratum over all treated patients in the sample.

We first define a function `att.strata.function()` to calculate stratum-specific estimates of the treatment effect:

```
att.strata.function <- function(data, stratum, confint = TRUE) {

  fit <- glm("y ~ treatment + offset(log(years))",
      family = poisson(link = "log"),
      data = data %>% filter(ps.strata == stratum))

  arr <- round(as.numeric(exp(coef(fit)["treatmentDMT1"])), digits = 3)
  ll <- ul <- NA

  if (confint) {
    ll <- round(exp(confint(fit))["treatmentDMT1",1], digits = 3)
    ul <- round(exp(confint(fit))["treatmentDMT1",2], digits = 3)
  }

  return(c("stratum" = stratum,
           "arr" = arr,
           "ci_lower"  = ll,
           "ci_upper"  = ul))
}

arr.strata <- as.data.frame(t(sapply(1:5, att.strata.function, data = dat)))
arr.strata
```

```
  stratum   arr ci_lower ci_upper
1       1 0.904    0.760    1.076
2       2 0.822    0.696    0.975
3       3 0.798    0.666    0.961
4       4 0.716    0.587    0.881
5       5 0.589    0.463    0.761
```

Subsequently, we define a function `weights.strata.function()` to calculate the weights for each stratum. The weight is the proportion of treated patients in each stratum over all treated patients in the sample:

```
weights.strata.function <- function(data, stratum) {
  n_DMT1_stratum <- nrow(data %>% filter(ps.strata == stratum & treatment == "DMT1"))
  n_DMT1_all <- nrow(data %>% filter(treatment == "DMT1"))
  weight <- n_DMT1_stratum/n_DMT1_all
  return(c("stratum" = stratum, "weight" = weight))
}

weights.strata <- as.data.frame(t(sapply(1:5, weights.strata.function, data = dat)))
weights.strata
```

```
  stratum    weight
1       1 0.1429870
2       2 0.1815584
3       3 0.2076623
4       4 0.2244156
5       5 0.2433766
```

```
# Create table with ARRs and weights for each PS stratum
arr.weights.merged <- merge(arr.strata, weights.strata, by = "stratum")

# Calculate the weighted ARR for each stratum
arr.weights.merged <- arr.weights.merged %>%
  mutate(weighted.arr = as.numeric(arr) * weight)

# Sum the weighted ARRs across strata to get the overall ATT
sum(arr.weights.merged$weighted.arr)
```

```
[1] 0.7482462
```

We now define a new function `ps.stratification.bootstrap()` that integrates estimation of the ATT and the PS weights for bootstrapping purposes:

```
ps.stratification.bootstrap <- function(data, inds) {
  d <- data[inds,]

  d$ps.strata <- cut(d$ps,
                     breaks = c(quantile(dat$ps, probs = seq(0, 1, by = 0.2))),
                     labels = seq(5),
                     include.lowest = TRUE)
```

```r
  arr.strata <- as.data.frame(t(sapply(1:5, att.strata.function,
                                       data = d, confint = FALSE)))

  weights.strata <- as.data.frame(t(sapply(1:5, weights.strata.function, data = d)))

  return(arr.strata$arr[1] * weights.strata$weight[1] +
           arr.strata$arr[2] * weights.strata$weight[2] +
           arr.strata$arr[3] * weights.strata$weight[3] +
           arr.strata$arr[4] * weights.strata$weight[4] +
           arr.strata$arr[5] * weights.strata$weight[5])
}
```

We can now estimate the treatment effect and its confidence interval using the bootstrap procedure:

```r
library(boot)
```

```
Attaching package: 'boot'

The following object is masked from 'package:survival':

    aml
```

```r
set.seed(1854)
arr.stratification.boot <- boot(data = dat,
                                 statistic = ps.stratification.bootstrap,
                                 R = 1000)

# Bootstrapped ARR
median(arr.stratification.boot$t)
```

```
[1] 0.7558609
```

```r
# Bootstrapped ARR 95% CI
quantile(arr.stratification.boot$t[,1], c(0.025, 0.975))
```

```
     2.5%     97.5%
0.6835885 0.8362947
```

3.5 Propensity score weighting

3.5.1 Calculate propensity score weights for ATT

Propensity score weighting reweights the study sample to generate an artificial population (i.e., pseudo-population) in which the covariates are no longer associated with treatment, thereby removing confounding by measured covariates. For the ATT, the weight for all treated patients is set to one. Conversely, the weight for patients in the control group is set to the propensity score divided by one minus the propensity score, that is, $(\mathrm{PS}/(1-\mathrm{PS}))$. We estimated stabilized weights to address extreme weights.

```r
library(WeightIt)

w.out <- weightit(treatment ~ age + female + prevDMTefficacy + prerelapse_num,
                  data = dat,
                  method = "ps",
                  estimand = "ATT")
                  #stabilize = TRUE)

w.out
```

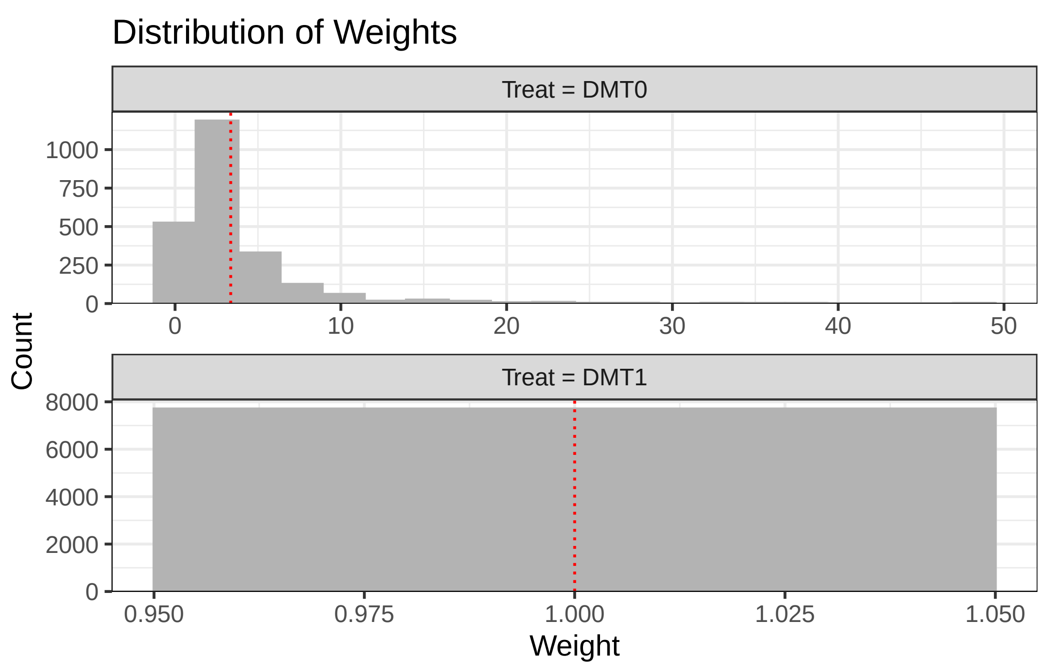
```
A weightit object
 - method: "glm" (propensity score weighting with GLM)
 - number of obs.: 10000
 - sampling weights: none
 - treatment: 2-category
 - estimand: ATT (focal: DMT1)
 - covariates: age, female, prevDMTefficacy, prerelapse_num
```

```r
summary(w.out)
```

```
                 Summary of weights

- Weight ranges:

        Min                                 Max
DMT0 0.4772 |---------------------------| 48.6856
DMT1 1.0000  ||                            1.0000

- Units with the 5 most extreme weights by group:
```

```
        9492    8836    6544    9610    4729
 DMT0 32.1027 32.1027 34.3126 38.1817 48.6856
           8       7       4       2       1
 DMT1      1       1       1       1       1

- Weight statistics:

     Coef of Var   MAD Entropy # Zeros
DMT0       1.098 0.673   0.383       0
DMT1       0.000 0.000  -0.000       0

- Effective Sample Sizes:

              DMT0 DMT1
Unweighted 2300.   7700
Weighted   1043.16 7700
```

```
plot(summary(w.out))
```



3.5.2 Assess balance in the weighted sample

```r
bal.tab(w.out, stats = c("m", "v"), thresholds = c(m = .05))
```

```
Balance Measures
                                         Type Diff.Adj     M.Threshold
prop.score                           Distance  -0.0045 Balanced, <0.05
age                                   Contin.   0.0054 Balanced, <0.05
female                                 Binary   0.0005 Balanced, <0.05
prevDMTefficacy_None                   Binary  -0.0003 Balanced, <0.05
prevDMTefficacy_Low_efficacy           Binary   0.0023 Balanced, <0.05
prevDMTefficacy_Medium_high_efficacy   Binary  -0.0020 Balanced, <0.05
prerelapse_num                        Contin.  -0.0034 Balanced, <0.05
                                     V.Ratio.Adj
prop.score                                0.9926
age                                       1.0102
female                                         .
prevDMTefficacy_None                           .
prevDMTefficacy_Low_efficacy                   .
prevDMTefficacy_Medium_high_efficacy           .
prerelapse_num                            1.0941

Balance tally for mean differences
                    count
Balanced, <0.05         7
Not Balanced, >0.05     0

Variable with the greatest mean difference
 Variable Diff.Adj     M.Threshold
      age   0.0054 Balanced, <0.05

Effective sample sizes
              DMT0 DMT1
Unadjusted 2300.   7700
Adjusted   1043.16 7700
```

3.5.3 Estimate the ATT

One way to estimate the ATT is to use the survey package. The function `svyglm()` generates model-robust (Horvitz-Thompson-type) standard errors by default, and thus does not require additional adjustments.

```r
library(survey)

weighted.data <- svydesign(ids = ~1, data = dat, weights = ~w.out$weights)

weighted.fit <- svyglm(y ~ treatment + offset(log(years)),
                       family = poisson(link = "log"),
                       design = weighted.data)

exp(coef(weighted.fit)["treatmentDMT1"])
```

```
treatmentDMT1
    0.7083381
```

```r
exp(confint(weighted.fit))["treatmentDMT1",]
```

```
    2.5 %    97.5 %
0.6245507 0.8033662
```

As indicated above, propensity score weighting yielded an ATT estimate of 0.71 (95% CI: 0.66; 0.76).

An alternative approach is to use `glm()` to estimate the treatment effect and calculate robust standard errors.

```r
# Alternative way to estimate treatment effect
weighted.fit2 <- glm(y ~ treatment + offset(log(years)),
              family = poisson(link = "log"),
              data = dat,
              weights = w.out$weights)

# Extract the estimated ARR
exp(coef(weighted.fit2))["treatmentDMT1"]
```

```
treatmentDMT1
    0.7083381
```

```r
# Calculate robust standard error and p-value of the log ARR
coeftest(weighted.fit2, vcov. = vcovHC)["treatmentDMT1",]
```

```
     Estimate    Std. Error       z value      Pr(>|z|)
-3.448337e-01  6.442745e-02 -5.352280e+00  8.685284e-08
```

```r
# Derive 95% confidence interval of the ARR
exp(lmtest::coefci(weighted.fit2,
       level = 0.95, # 95% confidence interval
       vcov. = vcovHC)["treatmentDMT1",])
```

```
    2.5 %    97.5 %
0.6243094 0.8036767
```

Using this approach, the ATT estimate was 0.71 (95% CI: 0.62; 0.8).

3.6 Regression adjustment for the propensity score for the ATE

In this approach, a regression model is fitted to describe the observed outcome as a function of the received treatment and the estimated propensity score:

```r
ps.reg.fit <- glm(y ~ treatment + ps + offset(log(years)),
                  family = poisson(link = "log"),
                  data = dat)

summary(ps.reg.fit)
```

```

Call:
glm(formula = y ~ treatment + ps + offset(log(years)), family = poisson(link = "log"),
    data = dat)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0160  -0.7336  -0.4441  -0.1352   4.2634

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   -1.99585    0.10359 -19.266  < 2e-16 ***
treatmentDMT1 -0.25598    0.04431  -5.777 7.60e-09 ***
ps             1.07521    0.13878   7.748 9.36e-15 ***
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 7514.7  on 9999  degrees of freedom
Residual deviance: 7443.0  on 9997  degrees of freedom
AIC: 12378

Number of Fisher Scoring iterations: 6
```

```r
# ATE
exp(coef(ps.reg.fit))["treatmentDMT1"]
```

```
treatmentDMT1
    0.7741606

Waiting for profiling to be done...
Waiting for profiling to be done...
```

Bootstrapped confidence intervals can be obtained as follows:

```r
# Function to bootstrap for 95% CIs
ps.reg.bootstrap <- function(data, inds) {
  d <- data[inds,]

  fit <- glm(y ~ treatment + ps + offset(log(years)),
             family = poisson(link = "log"),
             data = d)

  return(exp(coef(fit))["treatmentDMT1"])
}

set.seed(1854)

# Generate 1000 bootstrap replicates
arr.boot <- boot(dat, statistic = ps.reg.bootstrap, R = 1000)

# Extract the median annualized relapse rate across 1000 bootstrap replicates
median(arr.boot$t)
```

```
[1] 0.7750426
```

```r
# Take 2.5th and 97.5th percentiles to be 95% CI
quantile(arr.boot$t[,1], c(0.025, 0.975))
```

```
     2.5%     97.5%
0.7010540 0.8545169
```

3.7 Overview

Method	Estimand	Estimate	95% CI (lower)	95% CI (upper)
Optimal full matching	ATT	0.8039901	0.6040414	1.0039388
Propensity score stratification	ATT	0.7482462	NA	NA
Propensity score stratification (with bootstrapping)	ATT	0.7558609	0.6835885	0.8362947
Propensity score weighting	ATT	0.7083381	0.6245507	0.8033662
Propensity score weighting (robust SE)	ATT	0.7083381	0.6243094	0.8036767
PS regression adjustment	ATE	0.7741606	0.7101080	0.8448218
PS regression adjustment (bootstrapping)	ATE	0.7750426	0.7010540	0.8545169

Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19044)

Matrix products: default

locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
[1] grid      stats     graphics  grDevices utils     datasets  methods
[8] base
```

```
other attached packages:
 [1] WeightIt_0.14.1       boot_1.3-28.1         MatchIt_4.5.3
 [4] sandwich_3.0-2        truncnorm_1.0-9       tableone_0.13.2
 [7] survey_4.2-1          survival_3.5-5        Matrix_1.5-4
[10] MASS_7.3-58.3         marginaleffects_0.12.0 lmtest_0.9-40
[13] zoo_1.8-12            knitr_1.42            ggplot2_3.4.2
[16] data.table_1.14.8     cobalt_4.5.1          dplyr_1.1.1

loaded via a namespace (and not attached):
 [1] tidyselect_1.2.0 xfun_0.39        mitools_2.4      splines_4.2.3
 [5] haven_2.5.2      lattice_0.21-8   labelled_2.11.0  colorspace_2.1-0
 [9] vctrs_0.6.1      generics_0.1.3   htmltools_0.5.5  yaml_2.3.7
[13] utf8_1.2.3       rlang_1.1.0      e1071_1.7-13     pillar_1.9.0
[17] glue_1.6.2       withr_2.5.0      DBI_1.1.3        lifecycle_1.0.3
[21] munsell_0.5.0    gtable_0.3.3     codetools_0.2-19 evaluate_0.21
[25] labeling_0.4.2   forcats_1.0.0    fastmap_1.1.1    class_7.3-22
[29] fansi_1.0.4      optmatch_0.10.6  Rcpp_1.0.10      checkmate_2.2.0
[33] backports_1.4.1  scales_1.2.1     jsonlite_1.8.4   farver_2.1.1
[37] chk_0.8.1        hms_1.1.3        digest_0.6.31    insight_0.19.1
[41] cli_3.6.1        tools_4.2.3      magrittr_2.0.3   proxy_0.4-27
[45] tibble_3.2.1     crayon_1.5.2     pkgconfig_2.0.3  rlemon_0.2.1
[49] rmarkdown_2.21   rstudioapi_0.14  R6_2.5.1         compiler_4.2.3
```

References

4 Effect Modification Analysis within the Propensity score Framework

Mohammad Ehsanul Karim (University of British Columbia)

Observational comparative effectiveness studies often adopt propensity score analysis to adjust for confounding. Although this approach is relatively straightforward to implement, careful thought is needed when treatment effect heterogeneity is present. This chapter illustrates the estimation of subgroup-specific treatment effects using (traditional) covariate adjustment methods, propensity score matching, propensity score weighting, propensity score stratification, and covariate adjustment using propensity scores.

First, we need to install the R package `simcausal`, which can be obtained from GitHub:

```
devtools::install_github('osofr/simcausal', build_vignettes = FALSE)
```

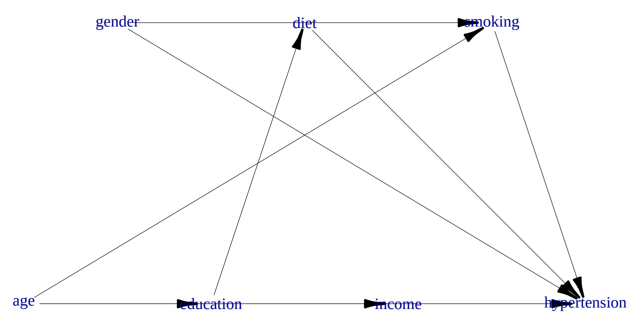
We will use the following data-generation model:

```
require(simcausal)
D <- DAG.empty()
D <- D +
  node("age", distr = "rnorm",
       mean = 2, sd = 4) +
  node("gender", distr = "rbern",
       prob = plogis(4)) +
  node("education", distr = "rbern",
       prob = plogis(3 + 5 * age)) +
  node("diet", distr = "rbern",
       prob = plogis(1 - 3 * education)) +
  node("income", distr = "rbern",
       prob = plogis(2 - 5 * education - 4 * age)) +
  node("smoking", distr = "rbern",
       prob = plogis(1 + 1.2 * gender + 2 * age)) +
  node("hypertension", distr = "rbern",
```

```
        prob = plogis(1 + log(3) * diet +
                      log(1.3) * age +
                      log(3.5) * smoking +
                      log(0.5) * gender))
Dset <- set.DAG(D)
```

Below is the diagram, with pink lines representing open backdoor path.

```
using the following vertex attributes:

NAdarkbluenone100.50

using the following edge attributes:

black0.210.60.5
```



We can now generate an example dataset:

```
Obs.Data <- sim(DAG = Dset, n = 50000, rndseed = 123)
Obs.Data$smoking <- as.character(Obs.Data$smoking)
Obs.Data$income <- as.factor(Obs.Data$income)
```

```
Obs.Data$income <- relevel(Obs.Data$income, ref = "1")
```

Sample data from the hypothetical example of association between hypertension and smoking, where other variables such as income, age [centered], gender, education and diet also plays a role in the data generation process.

	age	gender	education	diet	income	smoking	hypertension
34901	12.29	1	1	1	0	1	1
149	10.40	1	1	0	0	1	1
10060	2.99	1	1	0	0	1	0
22220	-4.31	0	0	0	1	0	1
9979	-6.44	0	0	0	1	0	1

4.1 Covariate adjustment

4.1.1 Interaction approach

Below, we estimate a logistic regression model to assess whether the effect of smoking (the exposure) on hypertension is modified by income levels. This model considers the following variables:

- Outcome: `hypertension`
- Exposure variables: `smoking` and `income`
- Confounders: `age` and `gender`

```
require(jtools)

fit.w.em <- glm(hypertension ~ smoking * income + age + gender,
            family = binomial(link = "logit"), data = Obs.Data)

results.model <- summ(fit.w.em, exp = TRUE)
```

Results indicate that the interaction between smoking status and income level is statistically significant (p = 0.02).

If we expand previous model to adjust for an additional confounder `education`, we have:

```
fit.w.int <- glm(hypertension ~ smoking * income + age + gender + education,
                 family = binomial(link = "logit"),
                 data = Obs.Data)
```

	exp(Est.)	2.5%	97.5%	z val.	p
(Intercept)	5.46	4.37	6.82	14.97	0.00
smoking1	2.93	2.60	3.30	17.69	0.00
income0	0.48	0.41	0.57	-8.28	0.00
age	1.29	1.27	1.31	36.77	0.00
gender	0.54	0.43	0.67	-5.55	0.00
smoking1:income0	1.27	1.04	1.56	2.33	0.02

```
results.int.model <- summ(fit.w.int, exp = TRUE)
```

	exp(Est.)	2.5%	97.5%	z val.	p
(Intercept)	5.69	4.56	7.11	15.31	0.00
smoking1	3.35	2.95	3.79	18.85	0.00
income0	1.09	0.85	1.40	0.68	0.49
age	1.30	1.28	1.32	37.32	0.00
gender	0.54	0.43	0.67	-5.58	0.00
education	0.42	0.35	0.51	-8.87	0.00
smoking1:income0	1.10	0.90	1.35	0.93	0.35

The interaction term between income and smoking is no longer statistically significant (p = 0.35).

We can generate a summary report from aforementioned effect modification analysis.

```
require(interactionR)

em.object <- interactionR(fit.w.em,
                          exposure_names = c("income0", "smoking1"),
                          ci.type = "mover", ci.level = 0.95,
                          em = TRUE, recode = FALSE)
```

The table below depicts the adjusted odds ratios for income levels (`high = 0`, and `low = 1`). The variables `CI.ll` and `CI.ul` depict the lower and upper limits of the 95 percent confidence intervals, `OR11` $= OR_{A=1,M=1}$, `OR10` $= OR_{A=1}$, `OR01` $= OR_{M=1}$ and `OR00` captures the reference.

Similarly, for the analysis adjusting for an additional confounder `education`, we have:

Table 4.1: Summary report from an interaction analysis when investigating association between two exposure variables (smoking and income) and hypertension.

Measures	Estimates	CI.ll	CI.ul
OR00	1.00	NA	NA
OR01	2.93	2.60	3.30
OR10	0.48	0.41	0.57
OR11	1.80	1.63	1.98
OR(smoking1 on outcome [income0==0]	2.93	2.60	3.30
OR(smoking1 on outcome [income0==1]	3.72	3.14	4.41
Multiplicative scale	1.27	1.04	1.56
RERI	-0.61	-0.98	-0.29

Table 4.2: Summary report from an interaction analysis when investigating association between two exposure variables (smoking and income) and hypertension.

Measures	Estimates	CI.ll	CI.ul
OR00	1.00	NA	NA
OR01	1.09	0.85	1.40
OR10	3.35	2.95	3.79
OR11	4.02	3.29	4.92
OR(income0 on outcome [smoking1==0]	1.09	0.85	1.40
OR(income0 on outcome [smoking1==1]	1.20	1.00	1.45
OR(smoking1 on outcome [income0==0]	3.35	2.95	3.79
OR(smoking1 on outcome [income0==1]	3.69	3.11	4.37
Multiplicative scale	1.10	0.90	1.35
RERI	0.59	0.03	1.27
AP	0.15	0.00	0.26
SI	1.24	1.01	1.53

```
# test run with additive model
Obs.Data$smoking <- as.numeric(as.character(Obs.Data$smoking))
Obs.Data$income <- as.numeric(as.character(Obs.Data$income))
fit.w.int.add <- glm(hypertension ~ smoking * income + age + gender + education,
                     family = gaussian(link = "identity"), data = Obs.Data)
sim_slopes(fit.w.int.add, pred = smoking, modx = income,
           exp = TRUE, robust = TRUE,
           confint = TRUE, data = Obs.Dat)
```

```
JOHNSON-NEYMAN INTERVAL

When income is INSIDE the interval [-3.27, 16.87], the slope of smoking is
p < .05.

Note: The range of observed values of income is [0.00, 1.00]

SIMPLE SLOPES ANALYSIS

Slope of smoking when income = 0.00 (0):

  Est.   S.E.   2.5%   97.5%   t val.      p
------ ------ ------ ------- -------- ------
  0.25   0.02   1.24    1.34    12.76   0.00

Slope of smoking when income = 1.00 (1):

  Est.   S.E.   2.5%   97.5%   t val.      p
------ ------ ------ ------- -------- ------
  0.28   0.01   1.30    1.34    34.53   0.00
```

4.1.2 Stratification

This approach involves estimating a regression model in different strata of the discrete effect modifier `income`:

```r
# Estimate the prognostic effect of smoking in low income individuals
fit.income1 <- glm(hypertension ~ smoking + age + gender,
            family = binomial(link = "logit"),
            data = subset(Obs.Data, income == 1))

# Estimate the prognostic effect of smoking in high income individuals
fit.income0 <- glm(hypertension ~ smoking + age + gender,
            family = binomial(link = "logit"),
            data = subset(Obs.Data, income == 0))
```

The table below summarizes the adjusted odds ratios for smoking across the different income levels (`low = 1`, and `high = 0`) as obtained using the stratified approach.

Note that we can obtain the same results by estimating a regression model with an interaction term between the modifier and all covariates:

Value of income	Estimate	2.5 %	97.5 %	z value	p value
1	3.07	2.71	3.47	17.65	0
0	3.59	3.02	4.26	14.57	0

```
fit.all.int <- glm(hypertension ~ income * (smoking + age + gender),
                   family = binomial(link = "logit"), data = Obs.Data)

# Odds ratio for smoking in individuals with low income
exp(coef(fit.all.int)["smoking"])
```

```
smoking
3.59026
```

```
# Odds ratio for smoking in individuals with high income
exp(coef(fit.all.int)["smoking"] + coef(fit.all.int)["income:smoking"])
```

```
 smoking
3.066878
```

4.2 Propensity score matching

4.2.1 Stratification with exact matching within subgroups

We simulate another example dataset using aforementioned DAG, but restrict the sample size to 5000 individuals to reduce computational burden.

```
set.seed(123)
Obs.Data <- sim(DAG = Dset, n = 5000, rndseed = 123)
```

We first estimate the propensity of smoking in the high-income group (`income == 0`):

```
require(MatchIt)

match.income.0 <- matchit(smoking ~ age + gender,
                          data = subset(Obs.Data, income == 0),
                          method = "full", distance = "glm", link = "logit")
data.income.0 <- match.data(match.income.0)
```

Below, we draw a sample from the high-income group based on the hypothetical example of an association between hypertension and smoking. Here age [centered], gender, education, and diet are covariates.

```
             age gender education diet income smoking hypertension  distance
657    6.0810120      0         1    1      0       1            1 0.9999874
4932   1.6109860      1         1    0      0       1            0 0.9943155
252   -0.2475055      1         1    1      0       0            1 0.8525107
2693  -0.2511048      1         1    0      0       1            1 0.8516785
1646  -0.2836155      1         0    1      0       1            1 0.8439843
         weights subclass
657   1.00000000       36
4932 1.00000000       50
252   0.03296089       25
2693 1.00000000       25
1646 1.00000000        4
```

Now, we do the same for the low-income group (`income == 1`):

```
match.income.1 <- matchit(smoking ~ age + gender,
                          data = subset(Obs.Data, income == 1),
                          method = "full", distance = "glm", link = "logit")
data.income.1 <- match.data(match.income.1)
```

We estimated the exposure effect from a weighted outcome model for the matched data. While the **weights** are essential for estimating the point estimate from the outcome model, the **subclass** variable assists in calculating the robust variance of the exposure effect estimate.

```
# Treatment effect estimation
fit.income.0 <- glm(hypertension ~ smoking + age + gender,
                    data = data.income.0, weights = weights,
                    family = binomial("logit"))
fit.income.1 <- glm(hypertension ~ smoking + age + gender,
                    data = data.income.1, weights = weights,
                    family = binomial("logit"))
# Robust variance calculation
fit.nexp.adj.res1 <- summ(fit.income.1,
                          robust = TRUE,
                          cluster = "subclass",
                          confint = TRUE)
fit.nexp.adj.res0 <- summ(fit.income.0,
                          robust = TRUE,
```

```
                         cluster = "subclass",
                         confint = TRUE)
```

Table 4.3: Subgroup-specific treatment effect estimates (expressed in log-OR) from the hypothetical example using the stratified approach.

Value of income	Est.	2.5%	97.5%	z val.	p
0	3.74	-37.58	45.06	0.18	0.86
1	1.39	0.94	1.85	6.04	0.00

4.2.2 Joint approach without exact matching within subgroups

Here, entire cohort data is used to estimate the propensity scores, and the effect modifier `income` is considered as a covariate in the propensity score model:

```
ps.formula <- as.formula("smoking ~ age + gender + income")
match.obj.j <- matchit(ps.formula, data = Obs.Data,
                      method = "full",
                      distance = "glm",
                      link = "logit")
match.data.j <- match.data(match.obj.j)
```

```
fit.joint.no.exact <- glm(hypertension ~ smoking*income + age + gender,
                          data = match.data.j,
                          weights = weights,
                          family = binomial("logit"))
require(interactions)
nem.nexp.adj.res <- sim_slopes(fit.joint.no.exact,
                               pred = smoking,
                               modx = income,
                               robust = "HC1",
                               cluster = "subclass",
                               johnson_neyman = TRUE,
                               confint = TRUE,
                               data = match.data.j)
```

4.2.3 Joint approach with exact matching within subgroups

We specify the moderator variable's name in the `exact` argument of the `matchit` function.

Table 4.4: Subgroup-specific treatment effect estimates (expressed in log-OR) from the hypothetical example using the joint approach.

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	3.85	1.00	1.89	5.82	3.84	0
1	1.40	0.28	0.85	1.95	4.99	0

```
ps.formula.no.mod <- as.formula("smoking ~ age + gender")
match.obj.js <- matchit(ps.formula.no.mod, data = Obs.Data,
                        method = "full", distance = "glm",link = "logit",
                        exact = "income")
match.data.js <- match.data(match.obj.js)
fit.joint.exact <- glm(hypertension ~ smoking*income + age + gender,
                       data = match.data.js, weights = weights,
                       family = binomial("logit"))
js.nexp.adj.res <- sim_slopes(fit.joint.exact,
                              pred = smoking, modx = income,
                              robust = "HC1", cluster = "subclass",
                              johnson_neyman = FALSE, confint = TRUE,
                              data = match.data.js)
```

Table 4.5: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using the Joint model, separate matching approach.

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	3.89	1.01	1.92	5.87	3.87	0
1	1.38	0.28	0.84	1.93	4.95	0

4.2.4 Interaction approach without exact matching within subgroups

Analysts incorporate relevant moderator-covariate interactions into the propensity score model that align with biological plausibility. For instance, in the case study we considered an interaction between age (a covariate) and income (a moderator), but did not include other interactions terms.

```
ps.formula.with.int <- formula("smoking ~ age*income + gender")
match.obj.i <- matchit(ps.formula.with.int, data = Obs.Data,
                       method = "full", distance = "glm",link = "logit")
match.data.i <- match.data(match.obj.i)
```

```r
fit.int.no.exact <- glm(hypertension ~ smoking*income + age + gender,
                        data = match.data.i, weights = weights,
                        family = binomial("logit"))
i.nexp.adj.res <- sim_slopes(fit.int.no.exact,
                             pred = smoking, modx = income,
                             robust = "HC1", cluster = "subclass",
                             johnson_neyman = FALSE, confint = TRUE,
                             data = match.data.i)
```

Table 4.6: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using the interaction approach.

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	3.87	1.00	1.90	5.83	3.86	0
1	1.39	0.28	0.84	1.94	4.95	0

4.2.5 Interaction approach with exact matching within subgroups

This method bears resemblance to the interaction approach for propensity score estimation. However, when it comes to matching, researchers match within each moderator subgroup.

```r
match.obj.is <- matchit(ps.formula.with.int, data = Obs.Data,
                      method = "full", distance = "glm",link = "logit",
                      exact = "income")
match.data.is <- match.data(match.obj.is)
fit.int.exact <- glm(hypertension ~ smoking*income + age + gender,
                     data = match.data.is, weights = weights,
                     family = binomial("logit"))
is.nexp.adj.res <- sim_slopes(fit.int.exact,
                              pred = smoking, modx = income,
                              robust = "HC1", cluster = "subclass",
                              johnson_neyman = FALSE, confint = TRUE,
                              data = match.data.is)
```

4.3 Propensity Score Weighting

4.3.1 Common model

This approach adds confounder-moderator interactions in the common weight model.

Table 4.7: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using the interaction model, separate matching approach.

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	3.86	1.00	1.90	5.83	3.85	0
1	1.40	0.28	0.85	1.95	4.99	0

```
require(WeightIt)
W.out <- weightit(ps.formula.with.int,
                  data = Obs.Data,
                  method = "ps",
                  estimand = "ATT")
require(survey)
d.w <- svydesign(~1, weights = W.out$weights, data = Obs.Data)
fit2w <- svyglm(hypertension ~ smoking*income, design = d.w,
                family = binomial("logit"))
w.nexp.adj.res <- sim_slopes(fit2w, pred = smoking, modx = income,
                             confint = TRUE)
```

Table 4.8: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using the weighting approach.

Value of income	Est.	S.E.	2.5%	97.5%	t val.	p
0	2.66	0.63	1.42	3.89	4.23	0
1	1.32	0.25	0.83	1.82	5.24	0

We can adjust previous analysis model to adopt stabilized weights for the propensity score (`stabilize = TRUE`):

```
W.out.st <- weightit(ps.formula.with.int, data = Obs.Data,
                     method = "ps",
                     estimand = "ATT",
                     stabilize = TRUE)
d.sw <- svydesign(~1, weights = W.out.st$weights, data = Obs.Data)
fit2sw <- svyglm(hypertension ~ smoking*income + age + gender,
                 design = d.sw,
                 family = binomial("logit"))
ws.nexp.adj.res <- sim_slopes(fit2sw,
                              pred = smoking, modx = income,
                              confint = TRUE)
```

Table 4.9: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using stabilized propensity score weights.

Value of income	Est.	S.E.	2.5%	97.5%	t val.	p
0	2.27	0.73	0.84	3.69	3.12	0
1	1.32	0.25	0.83	1.82	5.23	0

4.3.2 Separate models

Propensity score weighting approach with weights estimated separately from each subgroup:

```
ps.formula.with.no.int <- formula("smoking ~ age + gender")
W.out1 <- weightit(ps.formula.with.no.int,
                   data = subset(Obs.Data, income == 1),
                   method = "ps",
                   estimand = "ATT")
trimmed.weight.1.percent1 <- trim(W.out1$weights,
                                  at = 1, lower = TRUE)
```

Table 4.10: Weight summaries before and after truncation.

Weight	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Raw weights	0	0.01	0.11	0.45	1	11.69
1% truncated weights	0	0.01	0.11	0.44	1	7.61

```
# Outcome model for income = 1
d.w1 <- svydesign(~1, weights = trimmed.weight.1.percent1,
                  data = subset(Obs.Data, income == 1))
fit2unadj1 <- svyglm(hypertension ~ smoking, design = d.w1,
                     family = binomial("logit"))

# weight model for income = 0
W.out0 <- weightit(ps.formula, data = subset(Obs.Data, income == 0),
                  method = "ps", estimand = "ATT")
trimmed.weight.1.percent0 <- trim(W.out0$weights, at = 1, lower = TRUE)

# Outcome model for income = 0
d.w0 <- svydesign(~1, weights = trimmed.weight.1.percent0,
                  data = subset(Obs.Data, income == 0))
fit2unadj0 <- svyglm(hypertension ~ smoking, design = d.w0,
```

```
                    family = binomial("logit"))

fit.exp.adj.res1 <- summ(fit2unadj1, confint = TRUE)
fit.exp.adj.res0 <- summ(fit2unadj0, confint = TRUE)
```

Table 4.11: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using the propensity score weighting approach (Separate weight models).

Value of income	Est.	2.5%	97.5%	t val.	p
0	2.21	1.27	3.15	4.60	0
1	1.34	0.85	1.83	5.36	0

4.3.3 Weights from the subgroup balancing propensity scores

Subgroup balancing propensity scores for propensity score weighting:

```
w.out <- weightit(smoking ~ age + gender + income,
               data = Obs.Data,
               method = "ps", estimand = "ATT")
w.out.sb <- sbps(w.out, moderator = "income")
d.w.sb <- svydesign(~1, weights = w.out.sb$weights, data = Obs.Data)
fit2unadj.sb <- svyglm(hypertension ~ smoking*income, design = d.w.sb,
                      family = binomial("logit"))
sb.w.nexp.adj.res <- sim_slopes(fit2unadj.sb,
                              pred = smoking,
                              modx = income,
                              confint = TRUE,
                              johnson_neyman = FALSE,)
```

Table 4.12: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using the subgroup balancing weighting approach.

Value of income	Est.	S.E.	2.5%	97.5%	t val.	p
0	2.68	0.64	1.44	3.92	4.22	0
1	1.32	0.25	0.82	1.82	5.22	0

4.4 Covariate adjustment for the propensity score

4.4.1 As continuous covariate

An implementation of propensity scores as a continuous covariate in the outcome model:

```
# Separate models for each subgroup

# For subgroup income = 1
Obs.Data$ps[Obs.Data$income == 1] <- glm(ps.formula,
                                        data = subset(Obs.Data, income == 1),
                                        family = "binomial")$fitted.values
fit2adj1 <- glm(hypertension ~ smoking + age + gender,
                family = binomial("logit"),
                data = subset(Obs.Data, income == 1))

# For subgroup income = 0
Obs.Data$ps[Obs.Data$income == 0] <- glm(ps.formula,
                                        data = subset(Obs.Data, income == 0),
                                        family = "binomial")$fitted.values
fit2adj0 <- glm(hypertension ~ smoking + age + gender,
                family = binomial("logit"),
                data = subset(Obs.Data, income == 0))

fit.nexp.adj.res1 <- summ(fit2adj1, robust = TRUE, confint = TRUE)
fit.nexp.adj.res0 <- summ(fit2adj0, robust = TRUE, confint = TRUE)
```

Table 4.13: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using Propensity Score as a covariate adjustment approach (considering separate models for each subgroup).

Value of income	Est.	2.5%	97.5%	z val.	p
0	1.16	0.56	1.75	3.83	0
1	1.37	0.96	1.77	6.61	0

```
# Common model
Obs.Data$ps <- glm(ps.formula.with.int, data = Obs.Data,
                   family = "binomial")$fitted.values
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
fit2adjc <- glm(hypertension ~ smoking*income + age + gender + ps,
                family = binomial("logit"),
                data = Obs.Data)
c.nexp.adj.res <- sim_slopes(fit2adjc,
                             pred = smoking, modx = income,
                             confint = TRUE,
                             data = Obs.Data)
```

Table 4.14: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using Propensity Score as a covariate adjustment approach (considering a common model).

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	1.17	0.29	0.61	1.74	4.07	0
1	1.43	0.23	0.98	1.87	6.30	0

4.4.2 As quantiles

The propensity scores as a categorical covariate, broken by quintiles, in the outcome model.

```r
Obs.Data$ps <- glm(ps.formula.with.int,
                   data = Obs.Data,
                   family = "binomial")$fitted.values
quintiles <- quantile(Obs.Data$ps,
                      prob = seq(from = 0, to = 1, by = 0.2),
                      na.rm = T)
Obs.Data$psq <- cut(Obs.Data$ps, breaks = quintiles,
                    labels = seq(1,5), include.lowest = T)
Obs.Data$psq <- as.factor(Obs.Data$psq)

fit2adjq <- glm(hypertension ~ (smoking*psq)*income,
                family = binomial("logit"),
                data = Obs.Data)
cq.nexp.adj.res <- sim_slopes(fit2adjq,
                              pred = smoking,
                              modx = income,
                              confint = TRUE,
                              data = Obs.Data)
```

Table 4.15: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using Propensity Score as a covariate adjustment approach (as quintiles).

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	3.08	0.63	1.85	4.32	4.91	0
1	2.60	0.47	1.68	3.51	5.56	0

4.5 Propensity Score Stratification

Here is an implementation of propensity score stratification approach by using the marginal mean weighting through stratification (MMWS):

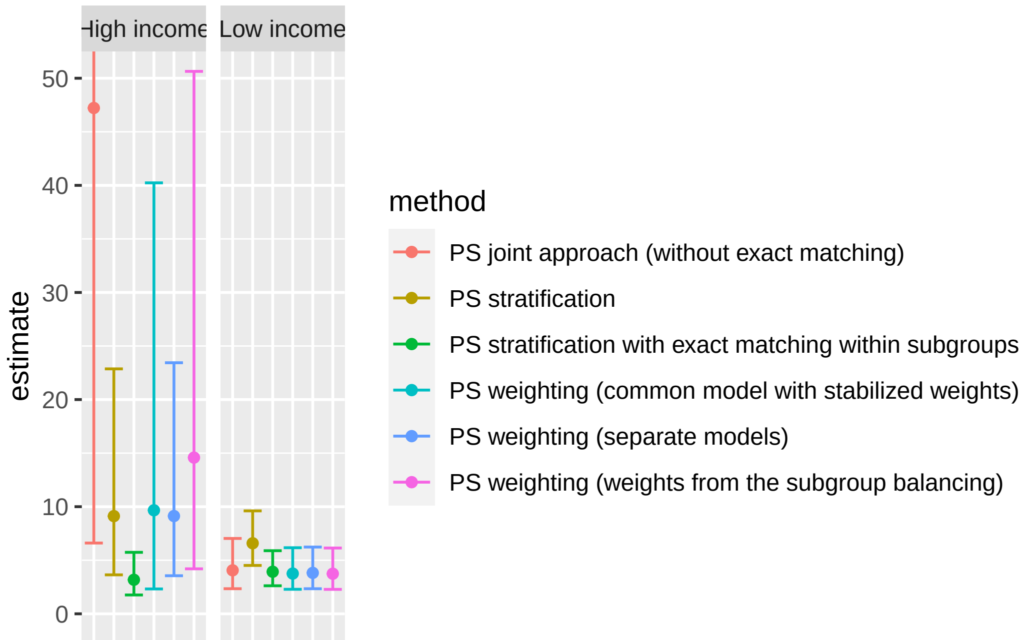
```
match.obj <- matchit(ps.formula, data = Obs.Data,
                      method = "subclass", subclass = 3,
                      estimand = "ATT", min.n = 10)
data.subclass <- match.data(match.obj)
subclass.fit <- glm(hypertension ~ smoking*income, family = binomial("logit"),
              data = data.subclass,
              weights = weights)
subclass.nexp.adj.res <- sim_slopes(subclass.fit,
                                    pred = smoking,
                                    modx = income,
                                    confint = TRUE,
                                    robust = "HC3",
                                    johnson_neyman = FALSE,
                                    data = data.subclass)
```

Table 4.16: Subgroup-specific exposure effect estimates (expressed in log-OR) from the hypothetical example using propensity score stratification approach.

Value of income	Est.	S.E.	2.5%	97.5%	z val.	p
0	2.21	0.47	1.29	3.13	4.71	0
1	1.89	0.19	1.51	2.26	9.78	0

4.6 Summary

The marginal odds ratios for `smoking` are summarized below



Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)

Matrix products: default

locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
[1] grid      stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] scales_1.2.1       ggplot2_3.4.2      readstata13_0.10.1 interactions_1.1.5
```

```
 [5] interactionR_0.1.6 jtools_2.2.1       simcausal_0.5.6    xtable_1.8-4
 [9] dplyr_1.1.1        kableExtra_1.3.4   knitr_1.42         cowplot_1.1.1
[13] survey_4.2-1       survival_3.5-5     Matrix_1.5-4       broom_1.0.4
[17] MatchIt_4.5.3      sandwich_3.0-2     lmtest_0.9-40      zoo_1.8-12
[21] optmatch_0.10.6    WeightIt_0.14.1    cobalt_4.5.1       table1_1.4.3

loaded via a namespace (and not attached):
 [1] fontquiver_0.2.1        webshot_0.5.4           httr_1.4.6
 [4] tools_4.2.3             backports_1.4.1         utf8_1.2.3
 [7] R6_2.5.1                DBI_1.1.3               colorspace_2.1-0
[10] withr_2.5.0             tidyselect_1.2.0        curl_5.0.0
[13] compiler_4.2.3          textshaping_0.3.6       cli_3.6.1
[16] rvest_1.0.3             expm_0.999-7            flextable_0.9.1
[19] xml2_1.3.3              officer_0.6.2           fontBitstreamVera_0.1.1
[22] mvtnorm_1.1-3           askpass_1.1             systemfonts_1.0.4
[25] stringr_1.5.0           digest_0.6.31           rmarkdown_2.21
[28] svglite_2.1.1           gfonts_0.2.0            pkgconfig_2.0.3
[31] htmltools_0.5.5         fastmap_1.1.1           rlang_1.1.0
[34] rstudioapi_0.14         httpcode_0.3.0          shiny_1.7.4
[37] generics_0.1.3          jsonlite_1.8.4          car_3.1-2
[40] zip_2.3.0               magrittr_2.0.3          Formula_1.2-5
[43] Rcpp_1.0.10             munsell_0.5.0           fansi_1.0.4
[46] abind_1.4-5             gdtools_0.3.3           lifecycle_1.0.3
[49] stringi_1.7.12          yaml_2.3.7              carData_3.0-5
[52] promises_1.2.0.1        crayon_1.5.2            lattice_0.21-8
[55] splines_4.2.3           pander_0.6.5            pillar_1.9.0
[58] uuid_1.1-0              igraph_1.4.2            crul_1.4.0
[61] glue_1.6.2              evaluate_0.21           msm_1.7
[64] mitools_2.4             fontLiberation_0.1.0    data.table_1.14.8
[67] vctrs_0.6.1             httpuv_1.6.9            gtable_0.3.3
[70] openssl_2.0.6           purrr_1.0.1             tidyr_1.3.0
[73] assertthat_0.2.1        xfun_0.39               mime_0.12
[76] later_1.3.0             ragg_1.2.5              viridisLite_0.4.2
[79] tibble_3.2.1            ellipsis_0.3.2
```

5 Dealing with missing data

Johanna Munoz (Julius Center for Health Sciences and Primary Care)
Thomas Debray (Smart Data Analysis and Statistics B.V.)

In this example, we consider the estimation of comparative treatment effects in the absence of treatment-effect heterogeneity.

5.0.1 Prepare R environment

```
library(mice)
library(dplyr)
library(ggmice)
library(MatchThem)
```

5.0.2 Generating an observational dataset

We can simulate an observational dataset of $N = 3000$ patients as follows:

```
data_noHTE <- generate_data(n = 3000, seed = 1234)
```

This dataset does not (yet) contain any missing values;

The simulated dataset contains two treatment groups with differences in baseline characteristics. For example, the figure below shows that we have baseline imbalance in age.

5.0.3 Generating missing values

Missing values can be generated using the function `getmissdata()`, which considers the following patterns of missingness for the previous number of relapses (`prerelapse_num`):

1. MAR: missingness depends on `age` and `sex`

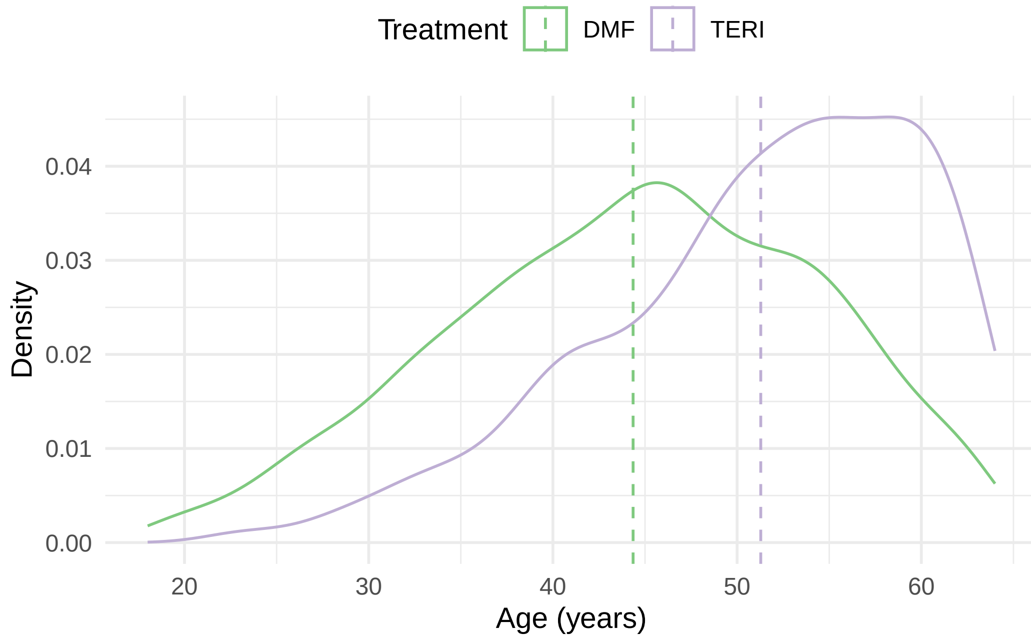


Figure 5.1: Distribution of the EDSS score at each time point

2. MART: missingness depends on `age`, `sex` and the treatment variable `treatment`
3. MARTY: missingness depends on `age`, `sex`, `treatment` and the outcome variable `y`
4. MNAR: missingness depends on `age`, `sex` and `prerelapse_num`

```
mdata_noHTE <- getmissdata(data_noHTE, "MART")
```

After introducing missing values, we only have complete data for $N = 946$ patients.

5.1 Analysis of incomplete data

5.1.1 Complete Case Analysis

Below, we describe how to estimate the ATE using propensity score matching.

```
impdata <- mdata_noHTE[complete.cases(mdata_noHTE), ]

# Apply Matching
mout <- matchit(DMF ~ age + female + prevDMTefficacy + premedicalcost + prerelapse_num,
                data = impdata,
```

Table 5.1: Baseline characteristics of the incomplete dataset.

	DMF	TERI	Overall
	(N=2265)	(N=735)	(N=3000)
Age (years)			
Mean (SD)	44.4 (10.0)	51.3 (8.72)	46.2 (10.1)
Median [Min, Max]	45.0 [18.0, 64.0]	53.0 [23.0, 64.0]	47.0 [18.0, 64.0]
Missing	248 (10.9%)	57 (7.8%)	305 (10.2%)
Female Sex			
Yes	1740 (76.8%)	526 (71.6%)	2266 (75.5%)
No	525 (23.2%)	209 (28.4%)	734 (24.5%)
Efficacy of previous DMT			
None	740 (32.7%)	325 (44.2%)	1065 (35.5%)
Low	190 (8.4%)	59 (8.0%)	249 (8.3%)
Medium or High	830 (36.6%)	246 (33.5%)	1076 (35.9%)
Missing	505 (22.3%)	105 (14.3%)	610 (20.3%)
Prior medical costs			
Mean (SD)	9970 (10700)	25500 (35400)	13900 (21200)
Median [Min, Max]	6530 [164, 102000]	12500 [259, 337000]	7450 [164, 337000]
Missing	257 (11.3%)	52 (7.1%)	309 (10.3%)
Number of prior symptoms			
0	157 (6.9%)	58 (7.9%)	215 (7.2%)
1	1169 (51.6%)	411 (55.9%)	1580 (52.7%)
>=2	435 (19.2%)	159 (21.6%)	594 (19.8%)
Missing	504 (22.3%)	107 (14.6%)	611 (20.4%)
Number of prior relapses			
Mean (SD)	0.453 (0.671)	0.408 (0.646)	0.436 (0.662)
Median [Min, Max]	0 [0, 4.00]	0 [0, 3.00]	0 [0, 4.00]
Missing	1365 (60.3%)	152 (20.7%)	1517 (50.6%)

```r
                  family = binomial,
                  method = "full",
                  caliper = 0.2,
                  estimand = "ATE",
                  replace = FALSE)

mdata <- as.data.table(match.data(mout))
match_mod <- glm("y ~ DMF + offset(log(years))",
                 family = poisson(link = "log"),
                 data = mdata,
                 weights = weights)

# Estimate robust variance-covariance matrix
tx_var <- vcovCL(match_mod, cluster = ~ subclass, sandwich = TRUE)
```

We can extract the treatment effect estimate as follows:

```r
# Treatment effect estimate (log rate ratio)
coef(match_mod)["DMF"]
```

```
       DMF
-0.3685717
```

```r
# Standard error
sqrt(tx_var["DMF", "DMF"])
```

```
[1] 0.1521243
```

5.1.2 Multiple Imputation (within method)

In this approach, we will generate $m = 5$ imputed datasets and perform matching within each imputed dataset. We first need to specify how the variables `prevDMTefficacy`, `premedicalcost`, `numSymptoms`, `prerelapse_num` and `age` will be imputed:

```r
# We add a covariate for log(years)
impdata <-  mdata_noHTE %>% mutate(logyears = log(years))

# Specify the conditional imputation models
form_y <- list(prevDMTefficacy ~ age + female + logyears + premedicalcost + numSymptoms +
                 treatment + prerelapse_num + y,
```

```r
                    premedicalcost ~ age + female + logyears + prevDMTefficacy + numSymptoms +
                      treatment + prerelapse_num + y,
                    numSymptoms ~ age + female + premedicalcost + logyears + prevDMTefficacy +
                      prerelapse_num + treatment + y,
                    prerelapse_num ~ age + female + premedicalcost + logyears + prevDMTefficacy
                      numSymptoms + treatment + y,
                    age ~ prerelapse_num + female + premedicalcost + logyears + prevDMTefficacy
                      numSymptoms + treatment + y)
form_y <- name.formulas(form_y)

# Adopt predictive mean matching for imputing the incomplete variables
imp0 <- mice(impdata, form = form_y, maxit = 0)
method <- imp0$method
method["numSymptoms"] <- "pmm"
method["prevDMTefficacy"] <- "pmm"

# Generate 5 imputed datasets
imp <- mice(impdata, form = form_y, method = method, m = 5, maxit = 100)
```

We can now estimate the ATE using propensity score analysis in each of the imputed datasets. We here adopt full matching without replacement.

```r
# Matching based on PS model
mout <- matchthem(DMF ~ age + female + prevDMTefficacy + premedicalcost + prerelapse_num,
                  datasets = imp,
                  approach = "within",
                  method = "full",
                  caliper = 0.2,
                  family = binomial,
                  estimand = "ATE",
                  distance = "glm",
                  link = "logit",
                  replace = FALSE)
```

The results are then combined using Rubin's rules. We adopt robust standard errors to account for clustering of matched individuals.

```r
match_mod <- summary(pool(with(mout, svyglm(y ~ DMF + offset(log(years)),
                                            family = poisson(link = "log")),
                               cluster = TRUE)), conf.int = TRUE)
```

We can extract the treatment effect estimate and corresponding standard error as follows:

```r
# Treatment effect estimate (log rate ratio)
(match_mod %>% filter(term == "DMF"))$estimate
```

```
[1] -0.1554094
```

```r
# Standard error
(match_mod %>% filter(term == "DMF"))$std.error
```

```
[1] 0.2202132
```

5.1.3 Multiple Imputation (across method)

```r
# Matching based on PS model
mout <- matchthem(DMF ~ age + female + prevDMTefficacy + premedicalcost + prerelapse_num,
                  datasets = imp,
                  approach = "across",
                  method = "full",
                  caliper = 0.2,
                  family = binomial,
                  estimand = "ATE",
                  distance = "glm",
                  link = "logit",
                  replace = FALSE)
```

The results are then combined using Rubin's rules. We adopt robust standard errors to account for clustering of matched individuals.

```r
match_mod <- summary(pool(with(mout, svyglm(y ~ DMF + offset(log(years)),
                                             family = poisson(link = "log")),
                               cluster = TRUE)), conf.int = TRUE)
```

We can extract the treatment effect estimate and corresponding standard error as follows:

```r
# Treatment effect estimate (log rate ratio)
(match_mod %>% filter(term == "DMF"))$estimate
```

```
[1] -0.3461563
```

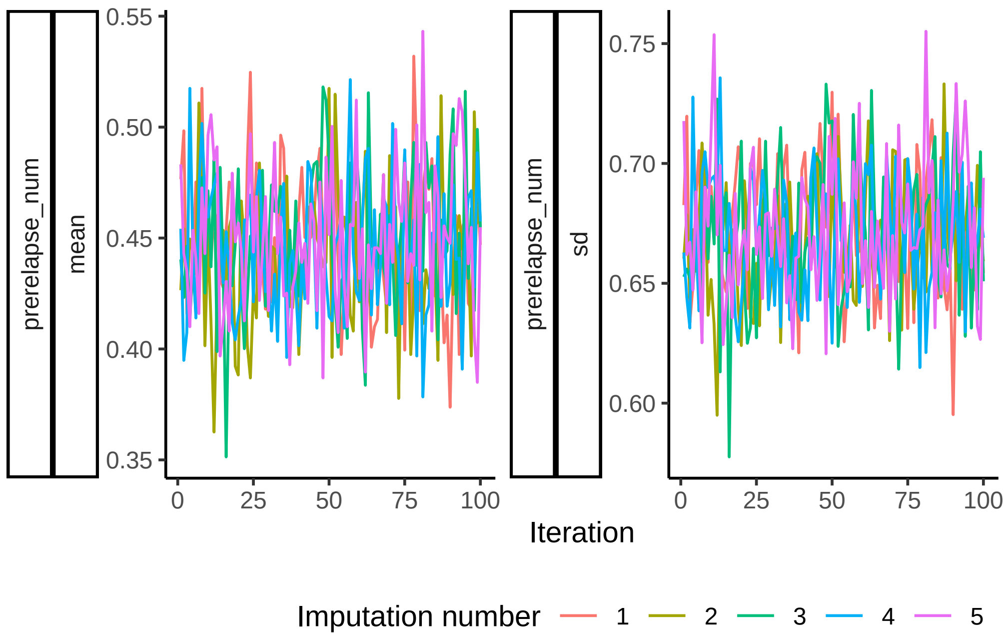
```r
# Standard error
(match_mod %>% filter(term == "DMF"))$std.error
```

```
[1] 0.1351187
```

5.2 Convergence checking

We can inspect convergence for the imputed variable `prerelapse_num` using a trace plot:

```r
plot_trace(imp, vrb = "prerelapse_num")
```

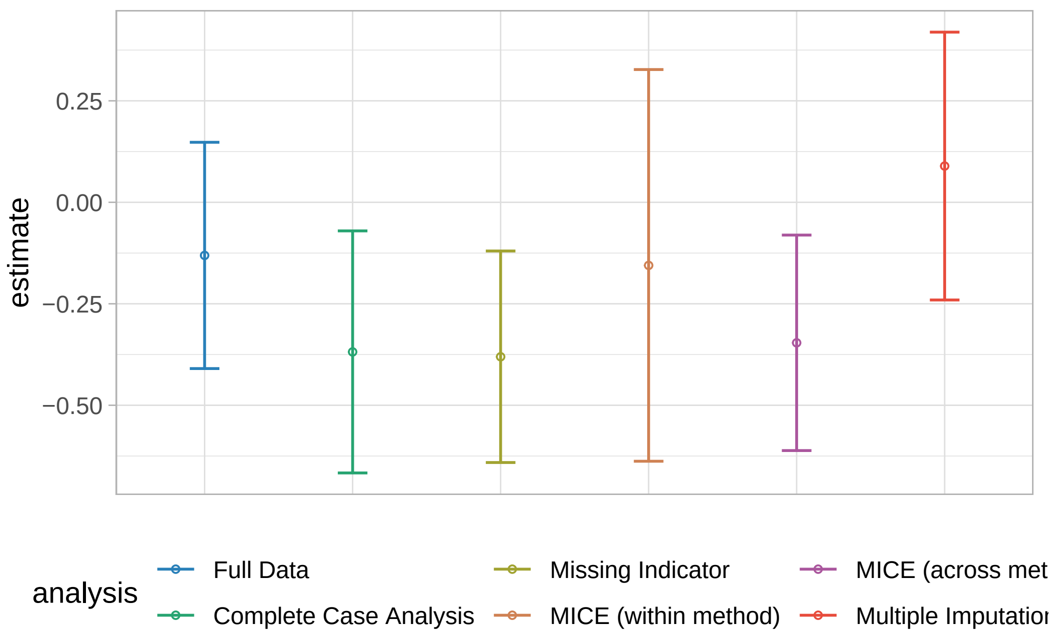


5.3 Results

Analysis methods:

- **Full Data**: The treatment effect is estimated in the original data of $N = 3000$ patients where no missing values are present. This estimate can be used as a benchmark to compare the missing data methods.

- **Complete Case Analysis**: The treatment effect is estimated using all data from $N = 946$ patients that do not have any missing values.
- **Missing Indicator**: The treatment effect is estimated in the incomplete dataset of $N = 3000$ patients. The propensity score model includes a missing indicator variable for each incomplete covariate.
- **MICE (within method)**: A treatment effect is estimated within each imputed dataset using propensity score analysis. Using Rubin's rule, the five treatment effects are combined into a single treatment effect.
- **MICE (ITE method)**: The missing covariates and potential outcomes are imputed simultaneously. Treatment effect estimates are derived by taking the average of the individualized treatment effect estimates Y|DMF - Y|TERI.



Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19044)

Matrix products: default
```

```
locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
[1] grid      stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] ggmice_0.0.1      table1_1.4.3      kableExtra_1.3.4  ggplot2_3.4.2
 [5] missForest_1.5    sandwich_3.0-2    PSweight_1.1.8    MatchThem_1.0.1
 [9] mice_3.15.0       cobalt_4.5.1      WeightIt_0.14.1   MatchIt_4.5.3
[13] optmatch_0.10.6   truncnorm_1.0-9   MASS_7.3-58.3     survey_4.2-1
[17] survival_3.5-5    Matrix_1.5-4      data.table_1.14.8 tidyr_1.3.0
[21] dplyr_1.1.1

loaded via a namespace (and not attached):
 [1] nlme_3.1-162         webshot_0.5.4        RColorBrewer_1.1-3
 [4] httr_1.4.6           numDeriv_2016.8-1.1  tools_4.2.3
 [7] backports_1.4.1      doRNG_1.8.6          utf8_1.2.3
[10] R6_2.5.1             DBI_1.1.3            colorspace_2.1-0
[13] nnet_7.3-19          withr_2.5.0          gbm_2.1.8.1
[16] tidyselect_1.2.0     compiler_4.2.3       cli_3.6.1
[19] rvest_1.0.3          see_0.7.5            xml2_1.3.3
[22] labeling_0.4.2       scales_1.2.1         nnls_1.4
[25] randomForest_4.7-1.1 systemfonts_1.0.4    stringr_1.5.0
[28] digest_0.6.31        minqa_1.2.5          rmarkdown_2.21
[31] svglite_2.1.1        pkgconfig_2.0.3      htmltools_0.5.5
[34] lme4_1.1-32          fastmap_1.1.1        itertools_0.1-3
[37] rlang_1.1.0          rstudioapi_0.14      generics_0.1.3
[40] farver_2.1.1         zoo_1.8-12           jsonlite_1.8.4
[43] magrittr_2.0.3       Formula_1.2-5        Rcpp_1.0.10
[46] munsell_0.5.0        fansi_1.0.4          lifecycle_1.0.3
[49] stringi_1.7.12       chk_0.8.1            yaml_2.3.7
[52] parallel_4.2.3       crayon_1.5.2         lattice_0.21-8
[55] splines_4.2.3        knitr_1.42           pillar_1.9.0
[58] boot_1.3-28.1        rngtools_1.5.2       codetools_0.2-19
[61] glue_1.6.2           evaluate_0.21        mitools_2.4
[64] vctrs_0.6.1          nloptr_2.0.3         foreach_1.5.2
[67] gtable_0.3.3         purrr_1.0.1          xfun_0.39
[70] SuperLearner_2.0-28  broom_1.0.4          viridisLite_0.4.2
```

```
[73] tibble_3.2.1         iterators_1.0.14     gam_1.22-2
[76] rlemon_0.2.1
```

6 Systematic review and meta-analysis of Real-World Evidence

Dimitris Mavridis (University of Ioannina)
Thomas Debray (Smart Data Analysis and Statistics B.V.)

We first load the required packages

```
library(dplyr)
library(gemtc)
library(netmeta)
```

6.1 Pairwise meta-analysis of clinical trials

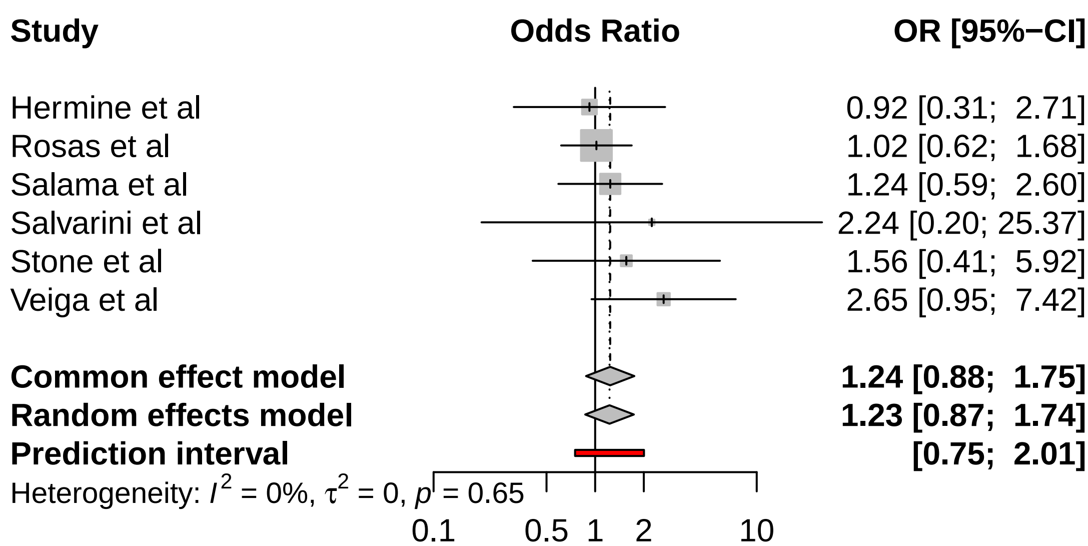
6.1.1 Toculizumab for coronavirus disease 2019

In this example, we consider the results from a systematic literature review of clinical trials investigating any pharmacological in hosptialized patients with coronavirus disease 2019 (Selvarajan et al. 2022). A total of 23 randomized controlled trials were included and studied seven different interventions: dexamethasone, remdesivir, tocilizumab, hydroxychloroquine, combination of lopinavir/ritonavir, favipiravir and interferon- . We here focus on the synthesis of 7 trials that comparted toculizumab (`TOCI`) to standard care (`STD`) and collected mortality data.

studlab	treat1	treat2	event1	n1	event2	n2
Hermine et al	TOCI	STD	7	63	8	67
Rosas et al	TOCI	STD	58	294	28	144
Salama et al	TOCI	STD	26	249	11	128
Salvarini et al	TOCI	STD	2	60	1	66
Stone et al	TOCI	STD	9	161	3	82
Veiga et al	TOCI	STD	14	65	6	64

We now conduct a pairwise meta-analysis to assess the pooled effect of tocilizumab versus standard care. For each study, the log odds ratio and corresponding standard error is derived after which the corresponding estimates are pooled using the Mantel-Haenszel method.

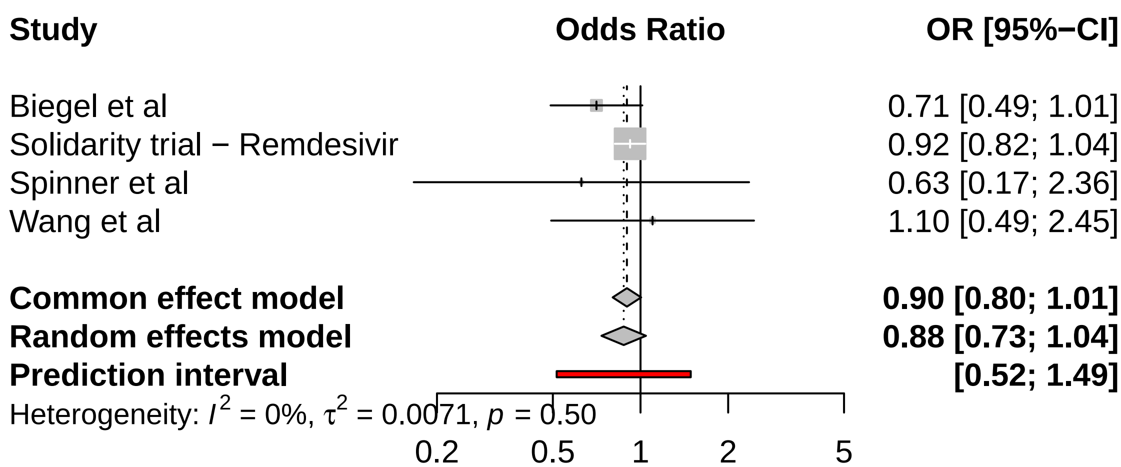
```
results.TOCI <- metabin(event1,n1,event2,n2,studlab,data=tocilizumab,
                        sm="OR",main="tocilizumab vs standard care",
                        prediction=TRUE)
forest(results.TOCI, leftcols = "studlab", rightcols = "effect.ci")
```



Altough a random effects meta-analysis was conducted, no heterogeneity was found (τ=0, with a 95% confidence interval ranging from 0 to 0.85).

6.1.2 Remdesivir for coronavirus disease 2019

In aforementioned example, a total of 4 trials compared remdesivir to standard care:



6.2 Network meta-analysis of clinical trials

We here use the R packages `netmeta` for conducting a frequentist network meta-analysis. A detailed tutorial on the use of `netmeta` is available from the book Doing Meta-Analysis with R: A Hands-On Guide.

6.2.1 Interventions for coronavirus disease 2019

We here consider data from a study which aimed to assess the comparative effectiveness of remdesivir and tocilizumab for reducing mortality in hospitalised COVID-19 patients. 80 trials were identified from two published network meta-analyses (Selvarajan et al. 2022), (Siemieniuk et al. 2020), a living COVID-19 trial database (COVID-NMA Initiative) [Covid-NMA.com], and a clinical trial database [clinicaltrials.gov]. Trials were included in this study if the patient population included hospitalized COVID-19 patients, active treatment was remdesivir or tocilizumab, comparator treatment was placebo or standard care, short-term mortality data was available, and the trial was published. 21 trials were included. For included trials, a risk of bias score was extracted from the COVID-NMA Initiative.

studlab	treat1	treat2	event1	n1	event2	n2
Ader	REM	STD	34	414	37	418
Beigel (ACTT-1)	REM	STD	59	541	77	521
Broman	TOCI	STD	1	57	0	29
Criner	REM	STD	4	384	4	200
Declerq (COV-AID)	TOCI	STD	10	81	9	74
Gordon (REMAP-CAP)	TOCI	STD	83	353	116	358
Hermine (CORIMUNO)	TOCI	STD	7	63	8	67
Horby (RECOVERY)	TOCI	STD	621	2022	729	2094
Islam	REM	STD	0	30	0	30
Mahajan	REM	STD	5	34	3	36
Pan (WHO Solidarity)	REM	STD	602	4146	643	4129
Rosas (COVACTA)	TOCI	STD	58	294	28	144
Rutgers	TOCI	STD	21	174	34	180
Salama (EMPACTA)	TOCI	STD	26	249	11	128
Salvarani	TOCI	STD	2	60	1	63
Soin (COVINTOC)	TOCI	STD	11	92	15	88
Spinner	REM	STD	5	384	4	200
Stone (BACC-BAY)	TOCI	STD	9	161	4	82
Talaschian	TOCI	STD	5	17	4	19
Veiga (TOCIBRAS)	TOCI	STD	14	65	6	64
Wang	REM	STD	22	158	10	78

The corresponding network is displayed below:

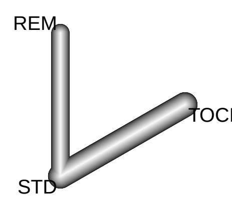


Figure 6.1: Evidence network of the 21 coronavirus-19 trials

We use the following command to calculate the log odds ratios and corresponding standard errors for each study:

```
covid <- pairwise(treat = treat, event = event, n = n, studlab = studlab, sm = "OR")
head(covid)
```

TE	seTE	studlab	treat1	treat2	event1	n1	event2	n2	incr	allst
-0.0819293	0.2483849	Ader	REM	STD	34	414	37	418	0.0	FAI
-0.3483875	0.1851030	Beigel (ACTT-1)	REM	STD	59	541	77	521	0.0	FAI
0.4487619	1.6487159	Broman	TOCI	STD	1	57	0	29	0.5	FAI
-0.6620566	0.7125543	Criner	REM	STD	4	384	4	200	0.0	FAI
0.0170679	0.4904898	Declerq (COV-AID)	TOCI	STD	10	81	9	74	0.0	FAI
-0.4442338	0.1688337	Gordon (REMAP-CAP)	TOCI	STD	83	353	116	358	0.0	FAI

Below, we conduct a random effects network meta-analysis where we consider standard care (`STD`) as the control treatment. Note that we have one study where zero cell counts occur, this study will not contribute to the NMA as the log odds ratio and its standard error cannot be determined.

```
NMA.covid <- netmeta(TE = TE, seTE = seTE, treat1 = treat1, treat2 = treat2,
                     studlab = studlab, data = covid, sm = "OR", ref = "STD",
                     comb.random = TRUE, common = FALSE, warn = FALSE)
NMA.covid
```

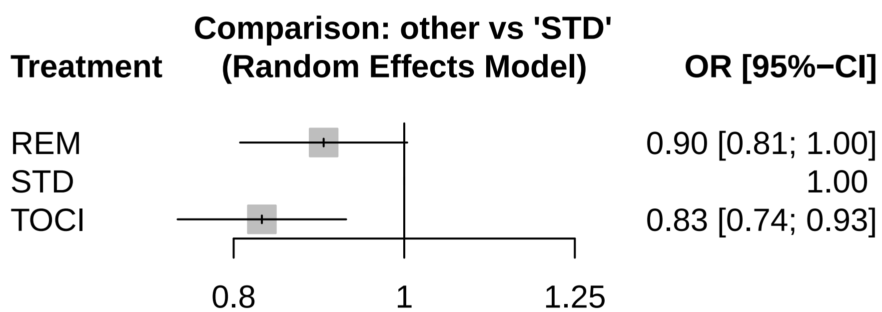
```
Number of studies: k = 20
Number of pairwise comparisons: m = 20
Number of treatments: n = 3
Number of designs: d = 2

Random effects model

Treatment estimate (sm = 'OR', comparison: other treatments vs 'STD'):
         OR            95%-CI     z p-value
REM  0.8999 [0.8067; 1.0039] -1.89  0.0588
STD       .                 .     .       .
TOCI 0.8301 [0.7434; 0.9268] -3.31  0.0009

Quantifying heterogeneity / inconsistency:
tau^2 = 0; tau = 0; I^2 = 0% [0.0%; 48.9%]

Tests of heterogeneity (within designs) and inconsistency (between designs):
                    Q d.f.  p-value
Total           16.38   18   0.5663
Within designs  16.38   18   0.5663
Between designs  0.00    0       --
```

A league table of the treatment effect estimates is given below:

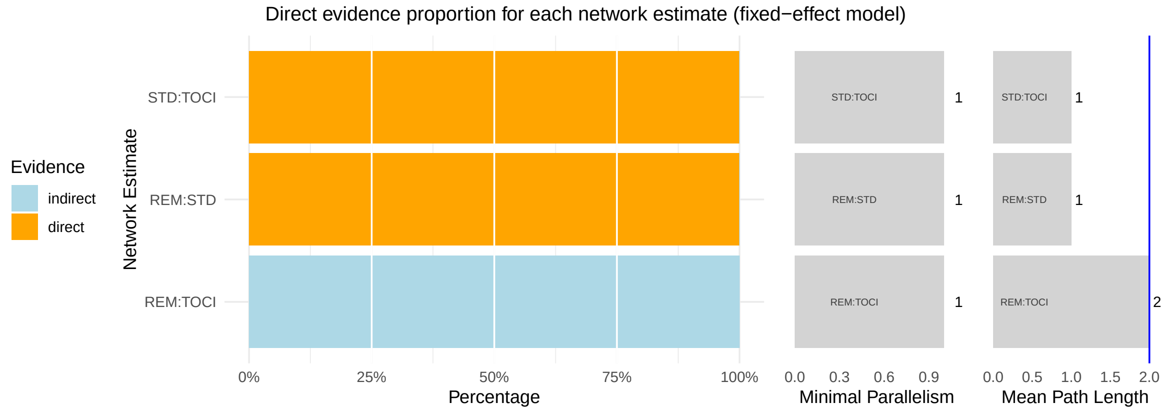
```
netleague(NMA.covid)
```

```
League table (random effects model):

                    REM 0.8999 [0.8067; 1.0039]                               .
 0.8999 [0.8067; 1.0039]                    STD 1.2047 [1.0789; 1.3451]
 1.0842 [0.9282; 1.2663] 1.2047 [1.0789; 1.3451]                        TOCI
```

We can also present the results in a forest plot:



The figure below shows the percentage of direct and indirect evidence used for each estimated comparison.



We now consider a Bayesian random effects network meta-analysis that analyzes the observed event counts using a binomial link function.

```
bdata <- data.frame(study = studlab,
                    treatment = treat,
                    responders = event,
                    sampleSize = n)

network <- mtc.network(data.ab  = bdata)

model <- mtc.model(network,
                   likelihood = "binom",
                   link = "log",
                   linearModel = "random",
                   n.chain = 3)
```

```
# Adaptation
mcmc1 <- mtc.run(model, n.adapt = 1000, n.iter = 1000, thin = 10)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 42
   Unobserved stochastic nodes: 45
   Total graph size: 930

Initializing model
```

```r
# Sampling
mcmc2 <- mtc.run(model, n.adapt = 10000, n.iter = 100000, thin = 10)
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 42
   Unobserved stochastic nodes: 45
   Total graph size: 930

Initializing model
```

We can extract the pooled treatment effect estimates from the posterior distribution. When using `STD` as control group, we have:

```r
summary(relative.effect(mcmc2, t1 = "STD"))
```

```

Results on the Log Risk Ratio scale

Iterations = 10010:110000
Thinning interval = 10
Number of chains = 3
Sample size per chain = 10000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

              Mean      SD  Naive SE Time-series SE
d.STD.REM  -0.1083 0.09805 0.0005661      0.0008072
d.STD.TOCI -0.1109 0.08315 0.0004800      0.0008498
sd.d        0.1134 0.08883 0.0005129      0.0018761

2. Quantiles for each variable:

               2.5%      25%      50%      75%   97.5%
d.STD.REM  -0.31694 -0.16154 -0.10534 -0.05205 0.08566
d.STD.TOCI -0.25663 -0.16159 -0.11886 -0.06907 0.08273
sd.d        0.00275  0.04497  0.09498  0.16190 0.33112
```

The corresponding odds ratios are as follows:

Comparison	95% CrI
REM vs. STD	0.9 (0.73; 1.09)
TOCI vs. STD	0.89 (0.77; 1.09)
REM vs. TOCI	1.01 (0.74; 1.27)

Finally, we expand the COVID-19 network with trials investigating the effectiveness of hydroxychloroquine (HCQ), lopinavir/ritonavir (LOPI), dexamethasone (DEXA) or interferon-β (INTB) (Selvarajan et al. 2022). The corresponding network is displayed below:

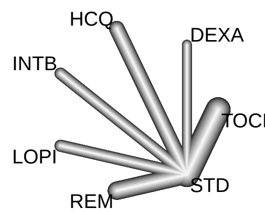


Figure 6.2: Evidence network of the 33 coronavirus-19 trials

We conducted a random effects network meta-analysis, results are depicted below:

```
Number of studies: k = 33
Number of pairwise comparisons: m = 33
Number of treatments: n = 7
Number of designs: d = 6

Random effects model

Treatment estimate (sm = 'OR', comparison: other treatments vs 'STD'):
         OR           95%-CI     z p-value            95%-PI
DEXA 0.8557 [0.7558; 0.9688] -2.46  0.0139  [0.7463; 0.9812]
HCQ  1.1809 [0.8934; 1.5610]  1.17  0.2428  [0.8786; 1.5872]
INTB 1.1606 [0.9732; 1.3841]  1.66  0.0973  [0.9604; 1.4026]
LOPI 1.0072 [0.8906; 1.1392]  0.11  0.9085  [0.8794; 1.1537]
REM  0.8983 [0.8014; 1.0070] -1.84  0.0658  [0.7913; 1.0199]
```

```
STD         .                  .      .       .                 .
TOCI 0.8304 [0.7410; 0.9306] -3.20  0.0014  [0.7316; 0.9426]

Quantifying heterogeneity / inconsistency:
tau^2 = 0.0004; tau = 0.0205; I^2 = 0.6% [0.0%; 42.3%]

Tests of heterogeneity (within designs) and inconsistency (between designs):
                    Q d.f. p-value
Total           27.18   27  0.4543
Within designs  27.18   27  0.4543
Between designs  0.00    0      --
```

We can calculate the P score for each treatment as follows:

```
netrank(NMA.covidf)
```

```
     P-score
TOCI  0.9070
DEXA  0.8357
REM   0.7143
STD   0.4027
LOPI  0.3899
HCQ   0.1336
INTB  0.1166
```

6.2.2 Pharmacologic treatments for chronic obstructive pulmonary disease

In this example, we consider the resuls from a systematic review of randomized controlled trials on pharmacologic treatments for chronic obstructive pulmonary disease (Baker, Baker, and Coleman 2009). The primary outcome, occurrence of one or more episodes of COPD exacerbation, is binary (yes / no). For this outcome, five drug treatments (fluticasone, budesonide, salmeterol, formoterol, tiotropium) and two combinations (fluticasone + salmeterol, budesonide + formoterol) were compared to placebo. The authors considered the two combinations as separate treatments instead of evaluating the individual components.

```
data(Baker2009)
```

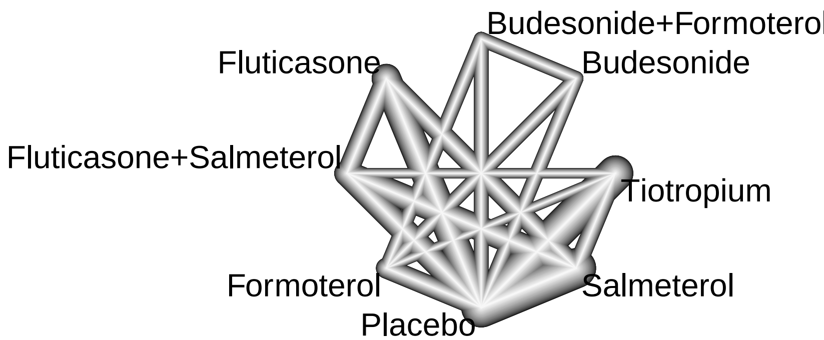
study	year	id	treatment	exac	total
Llewellyn-Jones 1996	1996	1	Fluticasone	0	8
Llewellyn-Jones 1996	1996	1	Placebo	3	8
Boyd 1997	1997	2	Salmeterol	47	229
Boyd 1997	1997	2	Placebo	59	227
Paggiaro 1998	1998	3	Fluticasone	45	142
Paggiaro 1998	1998	3	Placebo	51	139

```
Baker <- pairwise(treat = treatment,
                  event = exac,
                  n = total,
                  studlab = id,
                  sm = "OR",
                  data = Baker2009)

NMA.COPD <- netmeta(TE = TE, seTE = seTE, treat1 = treat1, treat2 = treat2,
                    studlab = studlab, data = Baker, sm="OR", ref = "Placebo",
                    comb.random = TRUE)
```

```
Warning: Comparisons with missing TE / seTE or zero seTE not considered in
network meta-analysis.
```

```
Comparisons not considered in network meta-analysis:
 studlab                 treat1     treat2 TE seTE
      39 Fluticasone+Salmeterol    Placebo NA   NA
      39 Fluticasone+Salmeterol Salmeterol NA   NA
      39             Salmeterol    Placebo NA   NA
```

```
netgraph(NMA.COPD)
```



6.2.3 Advanced Therapies for Ulcerative Colitis

In this example, we consider a systematic literature review of Phase 3 randomized controlled trials investigating the following advanced therapies: infliximab, adalimumab, vedolizumab, golimumab, tofacitinib, ustekinumab, filgotinib, ozanimod, and upadacitinib (Panaccione et al. 2023). This review included 48 RCTs, from which 23 were found eligible for inclusion in a network meta-analysis. The included RCT populations were largely comparable in their baseline characteristics, though some heterogeneity was noted in weight, disease duration, extent of disease, and concomitant medications. A risk of bias assessment showed a low risk of bias for all included RCTs, which were all industry sponsored.

We here focus on the synthesis of 18 trials that contributed efficacy data for induction in bio-naive populations. The following FDA- and/or EMA-approved biologic or SMD doses were investigated:

- Adalimumab subcutaneous 160 mg at week 0, 80 mg at week 2, and 40 mg at week 4 (`ADA160/80`)
- Infliximab intravenous 5 mg/kg (`INF5`) at weeks 0, 2, and 6 then every 8 weeks
- Infliximab intravenous 10 mg/kg (`INF10`) at weeks 0, 2, and 6 then every 8 weeks
- Filgotinib oral 100 mg once daily (`FIL100`)
- Filgotinib oral 200 mg once daily (`FIL200`)
- Golimumab subcutaneous 200 mg at week 0 and 100 mg at week 2 (`GOL200/100`)

- Ozanimod oral 0.23 mg once daily for 4 days, 0.46 mg once daily for 3 days, then 0.92 mg once daily (`OZA0.92`)
- Tofacitinib oral 10 mg twice daily for 8 weeks (`TOF10`)
- Upadacitinib oral 45 mg once daily for 8 weeks (`UPA45`)
- Ustekinumab intravenous 6 mg/kg at week 0 (`UST6`)
- Vedolizumab intravenous 300 mg at weeks 0, 2, and 6 (`VED300`)

The reference treatment is placebo (`PBO`).

Table 6.4: Efficacy outcomes (i.e., clinical remission) data of induction bio-naïve populations

studlab	treat1	treat2	event1	n1	event2	n2
ACT-1	INF10	INF5	39	122	47	121
ACT-1	INF10	PBO	39	122	18	121
ACT-1	INF5	PBO	47	121	18	121
ACT-2	INF10	INF5	33	120	41	121
ACT-2	INF10	PBO	33	120	7	123
ACT-2	INF5	PBO	41	121	7	123
GEMINI 1	VED300	PBO	30	130	5	76
Japic CTI-060298	INF5	PBO	21	104	11	104
Jiang 2015	INF5	PBO	22	41	9	41
M10-447	ADA160/80	PBO	9	90	11	96
NCT01551290	INF5	PBO	11	50	5	49
NCT02039505	VED300	PBO	22	79	6	41
OCTAVE 1	TOF10	PBO	56	222	9	57
OCTAVE 2	TOF10	PBO	43	195	4	47
PURSUIT-SC	GOL200/100	PBO	45	253	16	251
SELECTION	FIL100	FIL200	47	277	60	245
SELECTION	FIL100	PBO	47	277	17	137
SELECTION	FIL200	PBO	60	245	17	137
TRUE NORTH	OZA0.92	PBO	66	299	10	151
U-ACCOMPLISH	UPA45	PBO	54	166	3	81
U-ACHIEVE Study 2	UPA45	PBO	41	145	4	72
ULTRA-1	ADA160/80	PBO	24	130	12	130
ULTRA-2	ADA160/80	PBO	32	150	16	145
UNIFI	UST6	PBO	27	147	15	151

The corresponding network is displayed below:

Below, we conduct a random effects network meta-analysis of the reported study effects (expressed as odds ratio) and consider placebo (`treat = "PBO"`) as the control treatment.

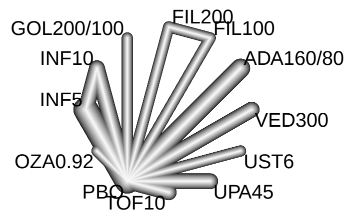


Figure 6.3: Evidence network of 18 trials that contributed efficacy data for induction in bio-naive populations

```
NMA.uc <- netmeta(TE = TE, seTE = seTE, treat1 = treat1, treat2 = treat2,
                  studlab = studlab, data = UlcerativeColitis, sm = "OR",
                  ref = "PBO", common = FALSE, comb.random = TRUE)
NMA.uc
```

All treatments except `FIL100` and `UST6` are significantly more efficacious than `PBO` at inducing clinical remission. We can now estimate the probabilities of each treatment being at each possible rank and the SUCRAs (Surface Under the Cumulative RAnking curve):

```
sucra.uc <- rankogram(NMA.uc, nsim = 100, random = TRUE, common = FALSE,
                      small.values = "undesirable")

# Exctract the SUCRA values
sucra.uc$ranking.random
```

```
 ADA160/80     FIL100     FIL200 GOL200/100      INF10       INF5    OZA0.92
0.25909091 0.15272727 0.44090909 0.67363636 0.56636364 0.74727273 0.77090909
       PBO      TOF10      UPA45       UST6     VED300
0.01636364 0.37909091 0.97909091 0.38909091 0.62545455
```

These results indicate that 97.9% of the evaluated treatments are worse than `UPA45`.

Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19044)

Matrix products: default

locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] dmetar_0.0.9000  netmeta_2.8-2    meta_6.2-1       gemtc_1.0-1
[5] coda_0.19-4      dplyr_1.1.1      kableExtra_1.3.4

loaded via a namespace (and not attached):
 [1] httr_1.4.6          magic_1.6-1         jsonlite_1.8.4
 [4] viridisLite_0.4.2   splines_4.2.3       stats4_4.2.3
 [7] metafor_4.0-0       slam_0.1-50         yaml_2.3.7
[10] robustbase_0.95-1   ggrepel_0.9.3       numDeriv_2016.8-1.1
[13] pillar_1.9.0        lattice_0.21-8      glue_1.6.2
[16] digest_0.6.31       rvest_1.0.3         minqa_1.2.5
[19] colorspace_2.1-0    MuMIn_1.47.5        htmltools_0.5.5
[22] Matrix_1.5-4        plyr_1.8.8          pkgconfig_2.0.3
[25] mvtnorm_1.1-3       Rglpk_0.6-5         scales_1.2.1
[28] webshot_0.5.4       svglite_2.1.1       rjags_4-14
[31] metadat_1.2-0       lme4_1.1-32         tibble_3.2.1
[34] farver_2.1.1        generics_0.1.3      ggplot2_3.4.2
[37] withr_2.5.0         nnet_7.3-19         cli_3.6.1
[40] magrittr_2.0.3      mclust_6.0.0        evaluate_0.21
[43] fansi_1.0.4         nlme_3.1-162        MASS_7.3-58.3
[46] truncnorm_1.0-9     forcats_1.0.0       xml2_1.3.3
[49] class_7.3-22        tools_4.2.3         lifecycle_1.0.3
[52] stringr_1.5.0       kernlab_0.9-32      munsell_0.5.0
[55] cluster_2.1.4       fpc_2.2-10          compiler_4.2.3
[58] systemfonts_1.0.4   rlang_1.1.0         grid_4.2.3
[61] nloptr_2.0.3        rstudioapi_0.14     CompQuadForm_1.4.3
[64] igraph_1.4.2        labeling_0.4.2      rmarkdown_2.21
[67] boot_1.3-28.1       gtable_0.3.3        codetools_0.2-19
[70] abind_1.4-5         flexmix_2.3-19      R6_2.5.1
```

```
[73] gridExtra_2.3       knitr_1.42          prabclus_2.3-2
[76] fastmap_1.1.1       utf8_1.2.3          mathjaxr_1.6-0
[79] poibin_1.5          modeltools_0.2-23   stringi_1.7.12
[82] parallel_4.2.3      Rcpp_1.0.10         vctrs_0.6.1
[85] DEoptimR_1.0-13     tidyselect_1.2.0    xfun_0.39
[88] diptest_0.76-0
```

References

7 Dealing with irregular and informative visits

We first load the required packages

```
library(dplyr)
library(broom)
library(ggplot2)
library(mice)
```

7.1 Example dataset

Below, we generate an example dataset that contains information on the treatment allocation `x` and three baseline covariates `age`, `sex` and `edss` (EDSS at treatment start). The discrete outcome `y` represents the Expanded Disability Status Scale (EDSS) score after `time` months of treatment exposure. Briefly, the EDSS is a semi-continuous measure that varies from 0 (no disability) to 10 (death).

```
set.seed(9843626)

dataset  <- sim_data_EDSS(npatients = 500,
                          ncenters = 10,
                          follow_up = 12*5, # Total follow-up (number of months)
                          sd_a_t = 0.5,   # DGM - Within-visit variation in EDSS scores
                          baseline_EDSS = 1.3295,    # DGM - Mean baseline EDDS score
                          sd_alpha_ij = 1.46,    # DGM - Between-subject variation in base
                          sd_beta1_j = 0.20,    # DGM - Between-site variation in baseline
                          mean_age = 42.41,
                          sd_age = 10.53,
                          min_age = 18,
                          beta_age = 0.05, # DGM - prognostic effect of age
                          beta_t = 0.014,  # DGM - prognostic effect of time
                          beta_t2 = 0,    # DGM - prognostic effect of time squared
                          delta_xt = 0, # DGM - interaction treatment time
                          delta_xt2 = 0, # 0.0005    # DGM - interaction treatment time2
```

```
                        p_female = 0.75,
                        beta_female = -0.2 ,  ## DGM - prognostic effect of male sex
                        delta_xf = 0,      ## DGM - interaction sex treatment
                        rho = 0.8,             # DGM - autocorrelation of between alpha_
                        corFUN = corAR1,       # DGM - correlation structure of the late
                        tx_alloc_FUN = treatment_alloc_confounding_v2 ) ## or treatment_
```

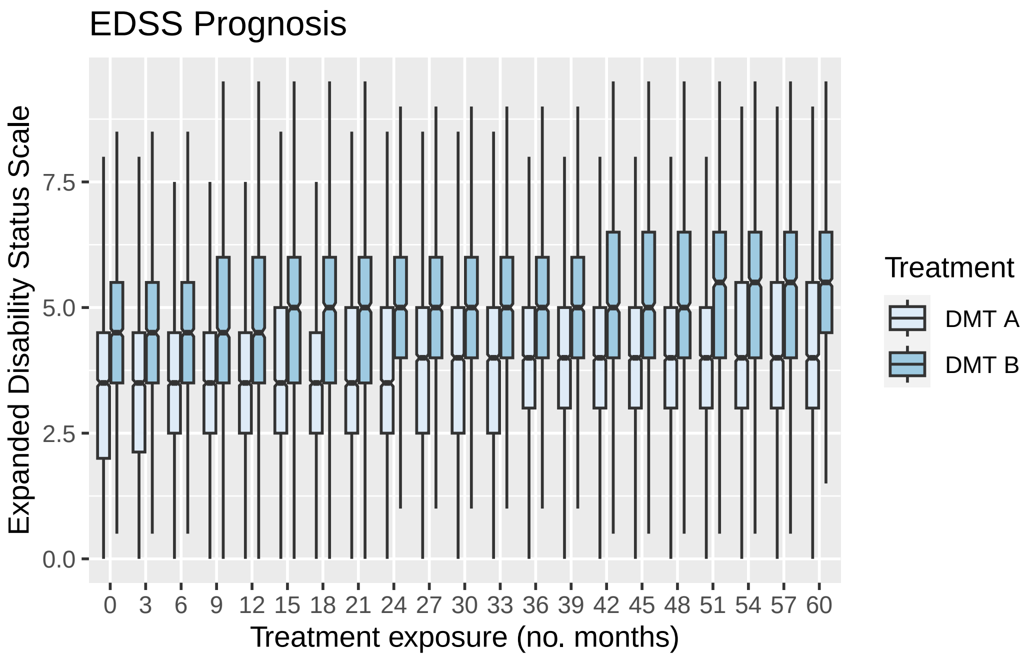


Figure 7.1: Distribution of the EDSS score at each time point

We remove the outcome `y` according to the informative visit process that depends on the received treatment, gender, and age.

```
dataset_visit <- censor_visits_a5(dataset, seed = 12345) %>%
  dplyr::select(-y) %>%
  mutate(time_x = time*x)
```

In the censored data, a total of 17 out of 5000 patients have a visit at `time=60`.

7.2 Estimation of treatment effect

We will estimate the marginal treatment effect at time `time=60`.

7.2.1 Original data

```
origdat60 <- dataset %>% filter(time == 60)

# Predict probability of treatment allocation
fitps <- glm(x ~ age + sex + edss, family = 'binomial',
             data = origdat60)

# Derive the propensity score
origdat60 <- origdat60 %>% mutate(ipt = ifelse(x == 1, 1/predict(fitps, type = 'response')
                                              1/(1-predict(fitps, type = 'response'))))

# Estimate
fit_ref_m <- tidy(lm(y ~ x, weight = ipt, data = origdat60), conf.int = TRUE)
```

7.2.2 Doubly-weighted marginal treatment effect

```
obsdat60 <- dataset_visit %>% mutate(visit = ifelse(is.na(y_obs),0,1)) %>% filter(time ==

gamma <- glm(visit ~ x + sex + age + edss, family = 'binomial', data = obsdat60)$coef

obsdat60 <- obsdat60 %>% mutate(rho_i = 1/exp(gamma["(Intercept)"] +
                                                         gamma["x"]*x +
                                                         gamma["sex"]*sex +
                                                         gamma["age"]*age))

# Predict probability of treatment allocation
fitps <- glm(x ~ age + sex + edss, family='binomial', data = obsdat60)

# Derive the propensity score
obsdat60 <- obsdat60 %>% mutate(ipt = ifelse(x==1, 1/predict(fitps, type='response'),
                                           1/(1-predict(fitps, type='response'))))


fit_w <- tidy(lm(y_obs ~ x, weights = ipt*rho_i, data = obsdat60), conf.int = TRUE)
```

7.2.3 Multilevel multiple imputation

We impute the entire vector of `y_obs` for all 61 potential visits and generate 10 imputed datasets. Note: `mlmi` currently does not support imputation of treatment-covariate interaction terms.

```
imp <- impute_y_mice_3l(dataset_visit, seed = 12345)
```

We can now estimate the treatment effect in each imputed dataset

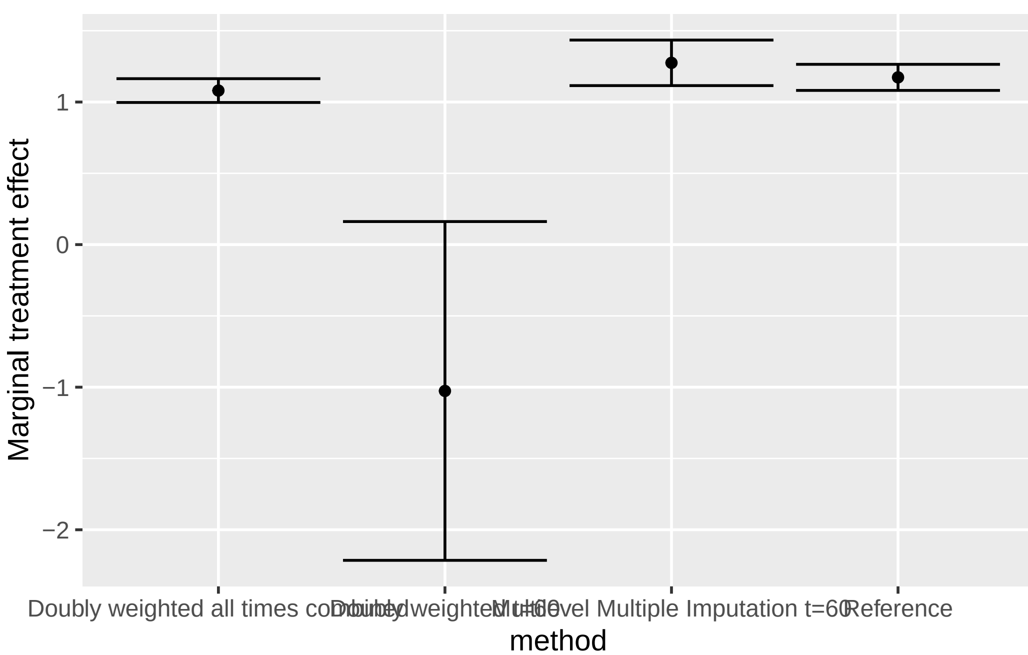
```
# Predict probability of treatment allocation
fitps <- glm(x ~ age + sex + edss, family='binomial', data = dataset_visit)

# Derive the propensity score
dataset_visit <- dataset_visit %>% mutate(ipt = ifelse(x==1, 1/predict(fitps, type='respon
                                                    1/(1-predict(fitps, type='response'

Q <- U <- rep(NA, 10) # Error variances

for (i in seq(10)) {
  dati <- cbind(dataset_visit[,c("x","ipt","time")], y_imp = imp[,i]) %>% filter(time == 6

  # Estimate
  fit <- tidy(lm(y_imp ~ x, weight = ipt, data = dati), conf.int = TRUE)

  Q[i] <- fit %>% filter(term == "x") %>% pull(estimate)
  U[i] <- (fit %>% filter(term == "x") %>% pull(std.error))**2
}

fit_mlmi <- pool.scalar(Q = Q, U = U)
```

7.3 Reproduce the results using all data to compute the marginal effect with IIV-weighted

7.3.1 Doubly -weighted marginal treatment effect total

```
obsdatall <- dataset_visit %>% mutate(visit = ifelse(is.na(y_obs),0,1))
gamma <- glm(visit ~ x + sex + age + edss, family = 'binomial', data = obsdatall)$coef
obsdatall <- obsdatall %>% mutate(rho_i = 1/exp(gamma["(Intercept)"] +
                                                 gamma["x"]*x +
                                                 gamma["sex"]*sex +
                                                 gamma["age"]*age))
# Predict probability of treatment allocation
fitps <- glm(x ~ age + sex + edss, family='binomial', data = obsdatall)
# Derive the propensity score
obsdatall <- obsdatall %>% mutate(ipt = ifelse(x==1, 1/predict(fitps, type='response'),
                                              1/(1-predict(fitps, type='response'))))
fit_w <- tidy(lm(y_obs ~ x, weights = ipt*rho_i, data = obsdatall), conf.int = TRUE)
```

7.4 Results



Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)

Matrix products: default

locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] broom_1.0.4     mice_3.15.0     ggplot2_3.4.2   dplyr_1.1.1
[5] truncnorm_1.0-9 MASS_7.3-58.3   nlme_3.1-162

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.10        RColorBrewer_1.1-3 pillar_1.9.0       compiler_4.2.3
 [5] tools_4.2.3        digest_0.6.31      jsonlite_1.8.4     evaluate_0.21
 [9] lifecycle_1.0.3    tibble_3.2.1       gtable_0.3.3       lattice_0.21-8
[13] pkgconfig_2.0.3    rlang_1.1.0        cli_3.6.1          rstudioapi_0.14
[17] yaml_2.3.7         xfun_0.39          fastmap_1.1.1      withr_2.5.0
[21] knitr_1.42         generics_0.1.3     vctrs_0.6.1        grid_4.2.3
[25] tidyselect_1.2.0   glue_1.6.2         R6_2.5.1           fansi_1.0.4
[29] rmarkdown_2.21     farver_2.1.1       tidyr_1.3.0        purrr_1.0.1
[33] magrittr_2.0.3     ellipsis_0.3.2     codetools_0.2-19   backports_1.4.1
[37] scales_1.2.1       htmltools_0.5.5    colorspace_2.1-0   labeling_0.4.2
[41] utf8_1.2.3         munsell_0.5.0
```

8 Prediction of individual treatment effect using data from multiple studies

Orestis Efthimiou (Institute of Social and Preventive Medicine (ISPM))

In this chapter, we discuss statistical methods for developing models to predict patient-level treatment effects using data from multiple randomized and non-randomized studies. We will first present prediction models that assume a constant treatment effect and discuss how to address heterogeneity in baseline risk. Subsequently, we will discuss approaches that allow for treatment effect modification by adopting two different approaches in an IPD-MA context, namely the risk modelling and the effect modelling approach. For both approaches, we will first discuss how to combine IPD from RCTs comparing the same two treatments. We will then discuss how these methods can be extended to include randomized data from multiple treatments, real-world data, and published aggregate data. We will discuss statistical software to implement these approaches and provide example code as supporting information. Real examples will be used throughout to illustrate the main methods.

We hereby provide code for estimating patient-level treatment effects for the case when we have patient-level data from multiple randomized trials.

8.0.1 Example of a continuous outcome

8.0.1.1 Setup

We start by simulating an artificial dataset using the R package **bipd**:

```
library(bipd)
ds <- generate_ipdma_example(type = "continuous")
```

Let us have a look at the dataset:

```
head(ds)
```

Table 8.1: The simulated dataset with a continuous outcome

	0	1	Overall
	(N=280)	(N=320)	(N=600)
z1			
Mean (SD)	0.00453 (1.05)	0.0842 (0.996)	0.0470 (1.02)
Median [Min, Max]	0.0424 [-3.32, 3.60]	0.0932 [-2.42, 2.99]	0.0670 [-3.32, 3.60]
z2			
Mean (SD)	-0.0129 (1.02)	-0.0320 (0.994)	-0.0231 (1.01)
Median [Min, Max]	-0.0189 [-2.57, 3.43]	-0.0832 [-2.95, 2.92]	-0.0351 [-2.95, 3.43]
studyid			
1	46 (16.4%)	54 (16.9%)	100 (16.7%)
2	51 (18.2%)	49 (15.3%)	100 (16.7%)
3	47 (16.8%)	53 (16.6%)	100 (16.7%)
4	50 (17.9%)	50 (15.6%)	100 (16.7%)
5	40 (14.3%)	60 (18.8%)	100 (16.7%)
6	46 (16.4%)	54 (16.9%)	100 (16.7%)

```
  studyid treat         z1         z2  y
1       1     0  0.9753016 -0.2599767 11
2       1     1  0.5335720 -0.5522604  6
3       1     0 -0.9011067  1.5785622 11
4       1     0 -1.3821428  0.9132537 11
5       1     0  0.5272515  0.6478884 11
6       1     0  0.7369785  0.9802165 11
```

The simulated dataset contains information on the following variables:

- the trial indicator `studyid`
- the treatment indicator `treat`, which takes the values 0 for control and 1 for active treatment
- two prognostic variables `z1` and `z2`
- the continuous outcome `y`

8.0.1.2 Model fitting

We synthesize the evidence using a Bayesian random effects meta-analysis model. The model is given in Equation 16.7 of the book. First we need set up the data and create the model:

```r
ipd <- with(ds, ipdma.model.onestage(y = y, study = studyid, treat = treat,
                                      X = cbind(z1, z2),
                                      response = "normal",
                                      shrinkage = "none"),
                                      type="random")
```

The JAGS model can be accessed as follows:

```r
ipd$model.JAGS
```

```
function ()
{
    for (i in 1:Np) {
        y[i] ~ dnorm(mu[i], sigma)
        mu[i] <- alpha[studyid[i]] + inprod(beta[], X[i, ]) +
            (1 - equals(treat[i], 1)) * inprod(gamma[], X[i,
                ]) + d[studyid[i], treat[i]]
    }
    sigma ~ dgamma(0.001, 0.001)
    for (j in 1:Nstudies) {
        d[j, 1] <- 0
        d[j, 2] ~ dnorm(delta[2], tau)
    }
    sd ~ dnorm(0, 1)
    T(0, )
    tau <- pow(sd, -2)
    delta[1] <- 0
    delta[2] ~ dnorm(0, 0.001)
    for (j in 1:Nstudies) {
        alpha[j] ~ dnorm(0, 0.001)
    }
    for (k in 1:Ncovariate) {
        beta[k] ~ dnorm(0, 0.001)
    }
    for (k in 1:Ncovariate) {
        gamma[k] ~ dnorm(0, 0.001)
    }
}
<environment: 0x0000024dbefe6c10>
```

We can fit the treatment effect model as follows:

```r
samples <- ipd.run(ipd, n.chains = 2, n.iter = 20,
                   pars.save = c("alpha", "beta", "delta", "sd", "gamma"))
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 600
   Unobserved stochastic nodes: 19
   Total graph size: 6034

Initializing model
```

Here are the estimated model parameters:

```r
summary(samples)
```

```
Iterations = 2001:2020
Thinning interval = 1
Number of chains = 2
Sample size per chain = 20

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

            Mean      SD Naive SE Time-series SE
alpha[1] 10.9627 0.05398 0.008535       0.011004
alpha[2]  7.9897 0.05497 0.008692       0.008720
alpha[3] 10.4637 0.05789 0.009153       0.010889
alpha[4]  9.6431 0.04469 0.007067       0.007159
alpha[5] 12.9361 0.07083 0.011199       0.015200
alpha[6] 15.9006 0.05371 0.008493       0.017285
beta[1]   0.2052 0.01704 0.002694       0.002532
beta[2]   0.2901 0.02161 0.003418       0.005191
delta[1]  0.0000 0.00000 0.000000       0.000000
delta[2] -2.3707 0.80802 0.127759       0.106608
gamma[1] -0.5404 0.02647 0.004186       0.004095
gamma[2]  0.5927 0.03136 0.004958       0.006968
sd        2.0794 0.41767 0.066040       0.119153
```

```
2. Quantiles for each variable:

             2.5%     25%     50%     75%   97.5%
alpha[1] 10.8564 10.9210 10.9737 11.0017 11.0558
alpha[2]  7.8940  7.9466  7.9908  8.0393  8.0704
alpha[3] 10.3824 10.4262 10.4575 10.4981 10.5943
alpha[4]  9.5589  9.6231  9.6441  9.6700  9.7093
alpha[5] 12.8068 12.8937 12.9368 12.9683 13.0748
alpha[6] 15.8038 15.8681 15.9028 15.9361 15.9894
beta[1]   0.1691  0.1966  0.2045  0.2166  0.2369
beta[2]   0.2520  0.2807  0.2923  0.3027  0.3241
delta[1]  0.0000  0.0000  0.0000  0.0000  0.0000
delta[2] -4.1413 -2.7365 -2.3399 -1.9803 -1.0305
gamma[1] -0.5852 -0.5564 -0.5362 -0.5219 -0.5008
gamma[2]  0.5390  0.5688  0.5904  0.6129  0.6539
sd        1.3300  1.8293  2.0270  2.3425  2.8810
```

8.0.1.3 Prection

We can now predict the individualized treatment effect for a new patient with covariate values `z1=1` and `z2=0.5`.

```r
round(treatment.effect(ipd, samples, newpatient = c(z1 = 1, z2 = 0.5)), 2)
```

```
0.025   0.5 0.975 
-4.36 -2.53 -1.24 
```

We can also predict treatment benefit for all patients in the sample, and look at the distribution of predicted benefit.

```r
library(dplyr)
library(ggplot2)

ds <- ds %>% mutate(benefit = NA)

for (i in seq(nrow(ds))) {
  newpat <- as.matrix(ds[i, c("z1", "z2")])
  ds$benefit[i] <- treatment.effect(ipd, samples, newpatient = newpat)["0.5"]
}
```

```r
ggplot(ds, aes(x = benefit)) + geom_histogram() + facet_wrap(~studyid) +
  xlab("Predicted treatment benefit")
```

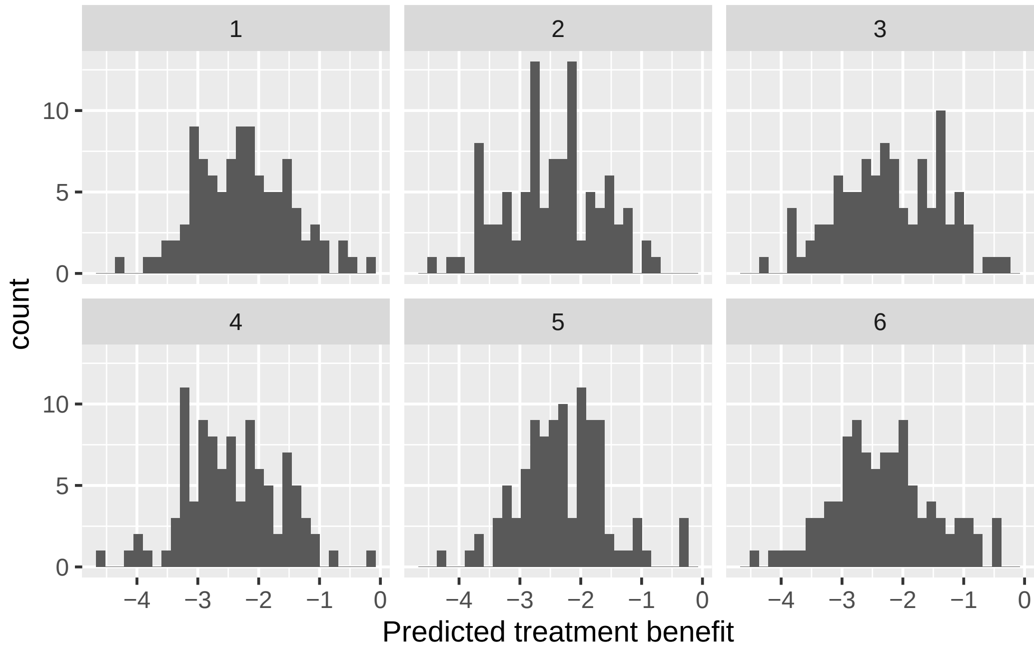


Figure 8.1: Distribution of predicted treatment benefit in each trial

8.0.1.4 Penalization

Let us repeat the analysis, but this time while penalizing the treatment-covariate coefficients using a Bayesian LASSO prior.

```r
ipd <- with(ds, ipdma.model.onestage(y = y, study = studyid,
                                     treat = treat,
                                     X = cbind(z1, z2),
                                     response = "normal",
                                     shrinkage = "laplace"),
            type = "random")

samples <- ipd.run(ipd, n.chains = 2, n.iter = 20,
                   pars.save = c("alpha", "beta", "delta", "sd", "gamma"))
```

```
Compiling model graph
```

```
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 600
   Unobserved stochastic nodes: 20
   Total graph size: 6039

Initializing model
```

```r
round(treatment.effect(ipd, samples, newpatient = c(1,0.5)), 2)
```

```
0.025   0.5 0.975
-4.42 -2.93 -1.55
```

8.0.2 Example of a binary outcome

8.0.2.1 Setup

We now present the case of a binary outcome. We first generate a dataset as before, using the **bipd** package.

```r
ds2 <- generate_ipdma_example(type = "binary")
head(ds2)
```

```
  studyid treat          w1          w2 y
1       1     0  2.26079432  0.08270796 0
2       1     1 -0.87905267  0.07818534 1
3       1     1  0.84648407 -0.68557300 0
4       1     1 -0.07243274 -0.21915053 0
5       1     0 -1.39887558 -1.18472380 1
6       1     0  1.13010890  1.01019519 0
```

The simulated dataset contains information on the following variables:

- the trial indicator `studyid`
- the treatment indicator `treat`, which takes the values 0 for control and 1 for active treatment
- two prognostic variables `w1` and `w2`
- the binary outcome `y`

Table 8.2: The simulated dataset with a binary outcome

	0	1	Overall
	(N=281)	(N=319)	(N=600)
w1			
Mean (SD)	0.00132 (1.06)	0.0394 (1.02)	0.0216 (1.04)
Median [Min, Max]	-0.00704 [-3.50, 2.81]	0.0208 [-3.53, 3.28]	-0.00250 [-3.53, 3.28]
w2			
Mean (SD)	0.0569 (1.05)	0.0536 (0.989)	0.0551 (1.02)
Median [Min, Max]	0.0533 [-3.20, 3.14]	0.0396 [-2.61, 2.94]	0.0452 [-3.20, 3.14]
studyid			
1	54 (19.2%)	46 (14.4%)	100 (16.7%)
2	48 (17.1%)	52 (16.3%)	100 (16.7%)
3	48 (17.1%)	52 (16.3%)	100 (16.7%)
4	46 (16.4%)	54 (16.9%)	100 (16.7%)
5	43 (15.3%)	57 (17.9%)	100 (16.7%)
6	42 (14.9%)	58 (18.2%)	100 (16.7%)

8.0.2.2 Model fitting

We use a Bayesian random effects model with binomial likelihood. This is similar to the model 16.7 of the book, but with a Binomial likelihood, i.e.

$$y_{ij} \sim Binomial(\pi_{ij})$$

$$logit(\pi_{ij}) == a_j + \delta_j t_{ij} + \sum_{l=1}^{L} \beta_l x_{ij} + \sum_{l=1}^{L} \gamma_l x_{ij} t_{ij}$$

The remaining of the model is as in the book. We can penalize the estimated parameters for effect modification (γ's), using a Bayesian LASSO. We can do this using again the *bipd* package:

```
ipd2 <- with(ds2, ipdma.model.onestage(y = y, study = studyid, treat = treat,
                                       X = cbind(w1, w2),
                                       response = "binomial",
                                       shrinkage = "laplace"),
             type="random", hy.prior = list("dunif", 0, 1))

ipd2$model.JAGS
```

```
function ()
{
    for (i in 1:Np) {
        y[i] ~ dbern(p[i])
        logit(p[i]) <- alpha[studyid[i]] + inprod(beta[], X[i,
            ]) + (1 - equals(treat[i], 1)) * inprod(gamma[],
            X[i, ]) + d[studyid[i], treat[i]]
    }
    for (j in 1:Nstudies) {
        d[j, 1] <- 0
        d[j, 2] ~ dnorm(delta[2], tau)
    }
    sd ~ dnorm(0, 1)
    T(0, )
    tau <- pow(sd, -2)
    delta[1] <- 0
    delta[2] ~ dnorm(0, 0.001)
    for (j in 1:Nstudies) {
        alpha[j] ~ dnorm(0, 0.001)
    }
    for (k in 1:Ncovariate) {
        beta[k] ~ dnorm(0, 0.001)
    }
    tt <- lambda
    lambda <- pow(lambda.inv, -1)
    lambda.inv ~ dunif(0, 5)
    for (k in 1:Ncovariate) {
        gamma[k] ~ ddexp(0, tt)
    }
}
<environment: 0x000001c505248b20>
```

```r
samples <- ipd.run(ipd2, n.chains = 2, n.iter = 20,
                   pars.save = c("alpha", "beta", "delta", "sd", "gamma"))
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 600
   Unobserved stochastic nodes: 19
```

```
   Total graph size: 6637

Initializing model
```

```r
summary(samples)
```

```
Iterations = 2001:2020
Thinning interval = 1
Number of chains = 2
Sample size per chain = 20

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

               Mean      SD Naive SE Time-series SE
alpha[1]  0.314409 0.23693  0.03746        0.08131
alpha[2]  0.415040 0.25808  0.04081        0.03509
alpha[3]  0.372971 0.29496  0.04664        0.07697
alpha[4]  0.282879 0.24504  0.03874        0.09724
alpha[5]  0.309193 0.36969  0.05845        0.09239
alpha[6]  0.416293 0.23513  0.03718        0.04240
beta[1]  -0.081624 0.09859  0.01559        0.03202
beta[2]  -0.001239 0.11278  0.01783        0.03134
delta[1]  0.000000 0.00000  0.00000        0.00000
delta[2]  0.163257 0.21499  0.03399        0.07353
gamma[1] -0.026317 0.09577  0.01514        0.02762
gamma[2]  0.053277 0.13893  0.02197        0.02543
sd        0.212883 0.16121  0.02549        0.03602

2. Quantiles for each variable:

             2.5%      25%      50%      75%  97.5%
alpha[1] -0.23398  0.21377  0.26321  0.43235 0.8062
alpha[2] -0.03458  0.27748  0.40672  0.60923 0.8158
alpha[3] -0.13235  0.21117  0.35861  0.53537 1.0240
alpha[4] -0.16872  0.13181  0.26840  0.40272 0.7910
alpha[5] -0.31400  0.01581  0.39064  0.55050 0.8962
alpha[6]  0.01672  0.28265  0.41222  0.55848 0.9974
beta[1]  -0.23068 -0.16032 -0.09926 -0.04024 0.1046
beta[2]  -0.21747 -0.04087  0.01067  0.07342 0.1670
```

```
delta[1]  0.00000  0.00000  0.00000  0.00000 0.0000
delta[2] -0.24775 -0.01175  0.20478  0.30920 0.4467
gamma[1] -0.20439 -0.07810 -0.01325  0.01843 0.1247
gamma[2] -0.15807 -0.01805  0.02882  0.11297 0.4175
sd        0.05884  0.09855  0.15546  0.27606 0.6014
```

```r
round(treatment.effect(ipd2, samples, newpatient = c(w1= 1.6, w2 = 1.3)), 2)
```

```
0.025   0.5 0.975 
 0.71  1.16  2.51 
```

8.1 Estimating heterogeous treatment effects in network meta-analysis

8.1.1 Example of a continuous outcome

8.1.1.1 Setup

We use again the bipd package to simulate a dataset:

```r
ds3 <- generate_ipdnma_example(type = "continuous")
head(ds3)
```

```
  studyid treat         z1         z2  y
1       1     1  0.8389580 -0.4623191 11
2       1     1  1.0885397  0.6771985 11
3       1     1  0.5477005  1.0702080 11
4       1     1  0.3168774 -0.5743271 11
5       1     2 -0.1046093  0.2840399  8
6       1     2  0.5195208 -0.7449863  7
```

Let us look into the data a bit in more detail:

8.1.1.2 Model fitting

We will use the model shown in Equation 16.8 in the book. In addition, we will use Bayesian LASSO to penalize the treatment-covariate interactions.

Table 8.3: The simulated dataset with a continuous outcome

	1	2	3	Overall
	(N=363)	(N=330)	(N=307)	(N=1000)
z1				
Mean (SD)	0.0989 (0.942)	0.0146 (1.02)	0.0343 (1.02)	0.0512 (0.990)
Median [Min, Max]	0.141 [-2.86, 2.88]	-0.0756 [-3.57, 2.87]	0.0399 [-3.30, 2.86]	0.0401 [-3.57, 2.88]
z2				
Mean (SD)	0.0297 (1.07)	0.0740 (1.07)	0.0457 (0.951)	0.0492 (1.03)
Median [Min, Max]	0.0676 [-2.88, 2.87]	0.0709 [-2.58, 3.08]	0.0669 [-2.19, 3.03]	0.0672 [-2.88, 3.08]
studyid				
1	53 (14.6%)	47 (14.2%)	0 (0%)	100 (10.0%)
2	62 (17.1%)	38 (11.5%)	0 (0%)	100 (10.0%)
3	50 (13.8%)	50 (15.2%)	0 (0%)	100 (10.0%)
4	48 (13.2%)	0 (0%)	52 (16.9%)	100 (10.0%)
5	54 (14.9%)	0 (0%)	46 (15.0%)	100 (10.0%)
6	0 (0%)	58 (17.6%)	42 (13.7%)	100 (10.0%)
7	0 (0%)	47 (14.2%)	53 (17.3%)	100 (10.0%)
8	35 (9.6%)	29 (8.8%)	36 (11.7%)	100 (10.0%)
9	38 (10.5%)	24 (7.3%)	38 (12.4%)	100 (10.0%)
10	23 (6.3%)	37 (11.2%)	40 (13.0%)	100 (10.0%)

```r
ipd3 <- with(ds3, ipdnma.model.onestage(y = y, study = studyid, treat = treat,
                                        X = cbind(z1, z2),
                                        response = "normal",
                                        shrinkage = "laplace",
                                        type = "random"))
ipd3$model.JAGS
```

```
function ()
{
    for (i in 1:Np) {
        y[i] ~ dnorm(mu[i], sigma)
        mu[i] <- alpha[studyid[i]] + inprod(beta[], X[i, ]) +
            inprod(gamma[treat[i], ], X[i, ]) + d[studyid[i],
            treatment.arm[i]]
    }
    sigma ~ dgamma(0.001, 0.001)
    for (i in 1:Nstudies) {
        w[i, 1] <- 0
        d[i, 1] <- 0
        for (k in 2:na[i]) {
            d[i, k] ~ dnorm(mdelta[i, k], taudelta[i, k])
            mdelta[i, k] <- delta[t[i, k]] - delta[t[i, 1]] +
                sw[i, k]
            taudelta[i, k] <- tau * 2 * (k - 1)/k
            w[i, k] <- d[i, k] - delta[t[i, k]] + delta[t[i,
                1]]
            sw[i, k] <- sum(w[i, 1:(k - 1)])/(k - 1)
        }
    }
    sd ~ dnorm(0, 1)
    T(0, )
    tau <- pow(sd, -2)
    delta[1] <- 0
    for (k in 2:Ntreat) {
        delta[k] ~ dnorm(0, 0.001)
    }
    for (j in 1:Nstudies) {
        alpha[j] ~ dnorm(0, 0.001)
    }
    for (k in 1:Ncovariate) {
        beta[k] ~ dnorm(0, 0.001)
    }
```

```
    lambda[1] <- 0
    lambda.inv[1] <- 0
    for (m in 2:Ntreat) {
        tt[m] <- lambda[m] * sigma
        lambda[m] <- pow(lambda.inv[m], -1)
        lambda.inv[m] ~ dunif(0, 5)
    }
    for (k in 1:Ncovariate) {
        gamma[1, k] <- 0
        for (m in 2:Ntreat) {
            gamma[m, k] ~ ddexp(0, tt[m])
        }
    }
}
<environment: 0x000001c505d81000>
```

```
samples <- ipd.run(ipd3, n.chains = 2, n.iter = 20,
                   pars.save = c("alpha", "beta", "delta", "sd", "gamma"))
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 1000
   Unobserved stochastic nodes: 35
   Total graph size: 10141

Initializing model
```

```
summary(samples)
```

```
Iterations = 2001:2020
Thinning interval = 1
Number of chains = 2
Sample size per chain = 20

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
```

```
              Mean      SD Naive SE Time-series SE
alpha[1]   11.0211 0.04862 0.007687       0.010929
alpha[2]    8.0023 0.03967 0.006272       0.010361
alpha[3]   10.4812 0.04786 0.007567       0.007228
alpha[4]    9.6340 0.04152 0.006564       0.010433
alpha[5]   12.8820 0.04227 0.006683       0.006530
alpha[6]   13.2677 0.04441 0.007023       0.009057
alpha[7]    7.5066 0.05510 0.008712       0.015228
alpha[8]   11.2471 0.04219 0.006671       0.006427
alpha[9]   10.2140 0.03848 0.006085       0.005710
alpha[10]   9.1900 0.05854 0.009256       0.011445
beta[1]     0.2045 0.02841 0.004493       0.012800
beta[2]     0.2951 0.02433 0.003847       0.002858
delta[1]    0.0000 0.00000 0.000000       0.000000
delta[2]   -2.9664 0.06247 0.009878       0.009792
delta[3]   -1.1461 0.05006 0.007915       0.006894
gamma[1,1]  0.0000 0.00000 0.000000       0.000000
gamma[2,1] -0.5682 0.03464 0.005477       0.010041
gamma[3,1] -0.3085 0.04536 0.007172       0.015676
gamma[1,2]  0.0000 0.00000 0.000000       0.000000
gamma[2,2]  0.6212 0.04026 0.006365       0.006315
gamma[3,2]  0.4465 0.03584 0.005666       0.003887
sd          0.1190 0.05836 0.009227       0.017058

2. Quantiles for each variable:

               2.5%      25%     50%     75%   97.5%
alpha[1]   10.94012 10.98910 11.0131 11.0565 11.1000
alpha[2]    7.92922  7.97699  8.0086  8.0268  8.0752
alpha[3]   10.39654 10.45390 10.4832 10.5078 10.5795
alpha[4]    9.56674  9.61476  9.6355  9.6526  9.7022
alpha[5]   12.81085 12.85152 12.8802 12.9109 12.9630
alpha[6]   13.19767 13.24110 13.2556 13.3025 13.3532
alpha[7]    7.40926  7.46692  7.5046  7.5376  7.6142
alpha[8]   11.18477 11.22359 11.2435 11.2738 11.3282
alpha[9]   10.15695 10.18448 10.2093 10.2381 10.2860
alpha[10]   9.10389  9.14602  9.1846  9.2398  9.2911
beta[1]     0.16312  0.17901  0.2075  0.2191  0.2582
beta[2]     0.25611  0.27472  0.2952  0.3115  0.3337
delta[1]    0.00000  0.00000  0.0000  0.0000  0.0000
delta[2]   -3.10416 -3.00805 -2.9512 -2.9289 -2.8775
delta[3]   -1.22663 -1.18085 -1.1591 -1.1234 -1.0364
gamma[1,1]  0.00000  0.00000  0.0000  0.0000  0.0000
```

```
gamma[2,1] -0.62775 -0.59366 -0.5748 -0.5421 -0.5051
gamma[3,1] -0.39018 -0.34811 -0.2950 -0.2785 -0.2341
gamma[1,2]  0.00000  0.00000  0.0000  0.0000  0.0000
gamma[2,2]  0.54675  0.59428  0.6157  0.6565  0.6873
gamma[3,2]  0.39304  0.41720  0.4477  0.4738  0.5116
sd          0.02979  0.08112  0.1146  0.1633  0.2254
```

As before, we can use the `treatment.effect()` function of *bipd* to estimate relative effects for new patients.

```
treatment.effect(ipd3, samples, newpatient= c(1,2))
```

```
$`treatment 2`
    0.025       0.5     0.975
-2.482723 -2.330572 -2.200555

$`treatment 3`
     0.025        0.5      0.975
-0.7371516 -0.5980373 -0.4444616
```

This gives us the relative effects for all treatments versus the reference. To obtain relative effects between active treatments we need some more coding:

```
samples.all=data.frame(rbind(samples[[1]], samples[[2]]))
newpatient= c(1,2)
newpatient <- (newpatient - ipd3$scale_mean)/ipd3$scale_sd

median(
  samples.all$delta.2.+samples.all$gamma.2.1.*
    newpatient[1]+samples.all$gamma.2.2.*newpatient[2]
-
  (samples.all$delta.3.+samples.all$gamma.3.1.*newpatient[1]+
     samples.all$gamma.3.2.*newpatient[2])
)
```

```
[1] -1.749739
```

```
quantile(samples.all$delta.2.+samples.all$gamma.2.1.*
           newpatient[1]+samples.all$gamma.2.2.*newpatient[2]
         -(samples.all$delta.3.+samples.all$gamma.3.1.*newpatient[1]+
```

```
             samples.all$gamma.3.2.*newpatient[2])
         , probs = 0.025)
```

```
     2.5%
-1.919473
```

```r
quantile(samples.all$delta.2.+samples.all$gamma.2.1.*
           newpatient[1]+samples.all$gamma.2.2.*newpatient[2]
         -(samples.all$delta.3.+samples.all$gamma.3.1.*newpatient[1]+
             samples.all$gamma.3.2.*newpatient[2])
         , probs = 0.975)
```

```
    97.5%
-1.582467
```

8.1.2 Modeling patient-level relative effects using randomized and observational evidence for a network of treatments

We will now follow Chapter 16.3.5 from the book. In this analysis we will not use penalization, and we will assume fixed effects. For an example with penalization and random effects, see part 2 of this vignettte.

8.1.2.1 Setup

We generate a very simple dataset of three studies comparing three treatments. We will assume 2 RCTs and 1 non-randomized trial:

```r
ds4 <- generate_ipdnma_example(type = "continuous")
ds4 <- ds4 %>% filter(studyid %in% c(1,4,10)) %>%
  mutate(studyid = factor(studyid) %>%
           recode_factor(
             "1" = "1",
             "4" = "2",
             "10" = "3"),
         design = ifelse(studyid == "3", "nrs", "rct"))
```

The sample size is as follows:

```
          s1 s2 s3
treat A: 50 50 32
treat B: 50  0 34
treat C:  0 50 34
```

8.1.2.2 Model fitting

We will use the design-adjusted model, equation 16.9 in the book. We will fit a two-stage fixed effects meta-analysis and we will use a variance inflation factor. The code below is used to specify the analysis of each individual study. Briefly, in each study we adjust the treatment effect for the prognostic factors `z1` and `z2`, as well as their interaction with `treat`.

```
library(rjags)
```

```
Loading required package: coda
```

```
Linked to JAGS 4.3.1
```

```
Loaded modules: basemod,bugs
```

```
first.stage <- "
model{

for (i in 1:N){
    y[i] ~ dnorm(mu[i], tau)
    mu[i] <- a + inprod(b[], X[i,]) + inprod(c[,treat[i]], X[i,]) + d[treat[i]]
}
sigma ~ dunif(0, 5)
tau <- pow(sigma, -2)

a ~ dnorm(0, 0.001)

for(k in 1:Ncovariate){
    b[k] ~ dnorm(0,0.001)
}

for(k in 1:Ncovariate){
    c[k,1] <- 0
}
```

```
tauGamma <- pow(sdGamma,-1)
sdGamma ~ dunif(0, 5)

for(k in 1:Ncovariate){
    for(t in 2:Ntreat){
        c[k,t] ~ ddexp(0, tauGamma)
    }
}

d[1] <- 0
for(t in 2:Ntreat){
    d[t] ~ dnorm(0, 0.001)
}
}"
```

Subsequently, we estimate the relative treatment effects in the first (randomized) study comparing treatments A and B:

```
model1.spec <- textConnection(first.stage)
data1 <- with(ds4 %>% filter(studyid == 1),
              list(y = y,
                   N = length(y),
                   X = cbind(z1,z2),
                   treat = treat,
                   Ncovariate = 2,
                   Ntreat = 2))
jags.m <- jags.model(model1.spec, data = data1, n.chains = 2, n.adapt = 500,
                     quiet =  TRUE)
params <- c("d", "c")
samps4.1 <- coda.samples(jags.m, params, n.iter = 50)
samps.all.s1 <- data.frame(as.matrix(samps4.1))

samps.all.s1 <- samps.all.s1[, c("c.1.2.", "c.2.2.", "d.2.")]
delta.1 <- colMeans(samps.all.s1)
cov.1 <- var(samps.all.s1)
```

We repeat the analysis for the second (randomized) study comparing treatments A and C:

```
model1.spec <- textConnection(first.stage)
data2 <- with(ds4 %>% filter(studyid == 2),
              list(y = y,
```

```
                     N = length(y),
                     X = cbind(z1,z2),
                     treat = ifelse(treat == 3, 2, treat),
                     Ncovariate = 2,
                     Ntreat = 2))
jags.m <- jags.model(model1.spec, data = data2, n.chains = 2, n.adapt = 100,
                       quiet =  TRUE)
params <- c("d", "c")
samps4.2 <- coda.samples(jags.m, params, n.iter = 50)
samps.all.s2 <- data.frame(as.matrix(samps4.2))
samps.all.s2 <- samps.all.s2[, c("c.1.2.", "c.2.2.", "d.2.")]
delta.2 <- colMeans(samps.all.s2)
cov.2 <- var(samps.all.s2)
```

Finally, we analyze the third (non-randomized) study comparing treatments A, B, and C:

```
model1.spec <- textConnection(first.stage)
data3 <- with(ds4 %>% filter(studyid == 3),
              list(y = y,
                   N = length(y),
                   X = cbind(z1,z2),
                   treat = treat,
                   Ncovariate = 2,
                   Ntreat = 3))
jags.m <- jags.model(model1.spec, data = data3, n.chains = 2, n.adapt = 100,
                       quiet = TRUE)
params <- c("d", "c")
samps4.3 <- coda.samples(jags.m, params, n.iter = 50)
samps.all.s3 <- data.frame(as.matrix(samps4.3))

samps.all.s3 <- samps.all.s3[, c("c.1.2.", "c.2.2.", "d.2.", "c.1.3.",
                                 "c.2.3.", "d.3.")]
delta.3 <- colMeans(samps.all.s3)
cov.3 <- var(samps.all.s3)
```

The corresponding treatment effect estimates are depicted below:

We can now fit the second stage of the network meta-analysis. The corresponding JAGS model is specified below:

```
second.stage <-
"model{
```

Table 8.4: Treatment effect estimates.

study	B versus A	C versus A
study 1	-2.988 (SE = 0.045)	
study 2		-1.000 (SE = 0.066)
study 3	-2.905 (SE = 0.073)	-1.005 (SE = 0.066)

```
  #likelihood
  y1 ~ dmnorm(Mu1, Omega1)
  y2 ~ dmnorm(Mu2, Omega2)
  y3 ~ dmnorm(Mu3, Omega3*W)


  Omega1 <- inverse(cov.1)
  Omega2 <- inverse(cov.2)
  Omega3 <- inverse(cov.3)

  Mu1 <- c(gamma[,1], delta[2])
  Mu2 <- c(gamma[,2], delta[3])
  Mu3 <- c(gamma[,1], delta[2],gamma[,2], delta[3])

  #parameters
  for(i in 1:2){
    gamma[i,1] ~ dnorm(0, 0.001)
    gamma[i,2] ~ dnorm(0, 0.001)
  }

  delta[1] <- 0
  delta[2] ~ dnorm(0, 0.001)
  delta[3] ~ dnorm(0, 0.001)

}
"
```

We can fit as follows:

```
model1.spec <- textConnection(second.stage)
data3 <- list(y1 = delta.1, y2 = delta.2, y3 = delta.3,
              cov.1 = cov.1, cov.2 = cov.2, cov.3 = cov.3, W = 0.5)
```

```r
jags.m <- jags.model(model1.spec, data = data3, n.chains = 2, n.adapt = 50,
                     quiet = TRUE)
params <- c("delta", "gamma")
samps4.3 <- coda.samples(jags.m, params, n.iter = 50)
```

```r
summary(samps4.3)
```

```

Iterations = 1:50
Thinning interval = 1
Number of chains = 2
Sample size per chain = 50

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

              Mean      SD Naive SE Time-series SE
delta[1]    0.0000 0.00000 0.000000       0.000000
delta[2]   -2.9832 0.05407 0.005407       0.005434
delta[3]   -1.0710 0.04480 0.004480       0.004026
gamma[1,1] -0.8750 0.05292 0.005292       0.005341
gamma[2,1]  0.8929 0.10287 0.010287       0.010339
gamma[1,2] -0.4718 0.05249 0.005249       0.005979
gamma[2,2]  0.4133 0.04751 0.004751       0.006745

2. Quantiles for each variable:

              2.5%     25%     50%     75%   97.5%
delta[1]    0.0000  0.0000  0.0000  0.0000  0.0000
delta[2]   -3.0655 -3.0180 -2.9911 -2.9550 -2.8851
delta[3]   -1.1354 -1.1082 -1.0764 -1.0360 -0.9770
gamma[1,1] -0.9793 -0.9104 -0.8705 -0.8419 -0.7732
gamma[2,1]  0.8025  0.8461  0.8742  0.9098  1.0133
gamma[1,2] -0.5621 -0.4973 -0.4647 -0.4366 -0.4024
gamma[2,2]  0.3406  0.3819  0.4063  0.4449  0.5094
```

```r
# calculate  treatment effects
samples.all=data.frame(rbind(samps4.3[[1]], samps4.3[[2]]))
newpatient= c(1,2)
```

```r
median(
  samples.all$delta.2.+samples.all$gamma.1.1.*newpatient[1]+
    samples.all$gamma.2.1.*newpatient[2]
)
```

```
[1] -2.103064
```

```r
quantile(samples.all$delta.2.+samples.all$gamma.1.1.*newpatient[1]+
           samples.all$gamma.2.1.*newpatient[2]
         , probs = 0.025)
```

```
     2.5%
-2.327086
```

```r
quantile(samples.all$delta.2.+samples.all$gamma.1.1.*newpatient[1]+
           samples.all$gamma.2.1.*newpatient[2]
         , probs = 0.975)
```

```
    97.5%
-1.713438
```

Version info

This chapter was rendered using the following version of R and its packages:

```
R version 4.2.3 (2023-03-15 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 19045)

Matrix products: default

locale:
[1] LC_COLLATE=Dutch_Netherlands.utf8  LC_CTYPE=Dutch_Netherlands.utf8
[3] LC_MONETARY=Dutch_Netherlands.utf8 LC_NUMERIC=C
[5] LC_TIME=Dutch_Netherlands.utf8

attached base packages:
```

```
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] rjags_4-14       coda_0.19-4      kableExtra_1.3.4 ggplot2_3.4.2
[5] bipd_0.3         dplyr_1.1.2      table1_1.4.3

loaded via a namespace (and not attached):
 [1] pillar_1.9.0      compiler_4.2.3    tools_4.2.3       digest_0.6.31
 [5] viridisLite_0.4.2 jsonlite_1.8.4    evaluate_0.21     lifecycle_1.0.3
 [9] tibble_3.2.1      gtable_0.3.3      lattice_0.21-8    pkgconfig_2.0.3
[13] rlang_1.1.1       cli_3.6.1         rstudioapi_0.14   yaml_2.3.7
[17] mvtnorm_1.1-3     xfun_0.39         fastmap_1.1.1     stringr_1.5.0
[21] xml2_1.3.4        withr_2.5.0       httr_1.4.6        knitr_1.43
[25] systemfonts_1.0.4 generics_0.1.3    vctrs_0.6.2       webshot_0.5.4
[29] grid_4.2.3        tidyselect_1.2.0  svglite_2.1.1     glue_1.6.2
[33] R6_2.5.1          fansi_1.0.4       rmarkdown_2.22    Formula_1.2-5
[37] magrittr_2.0.3    codetools_0.2-19  scales_1.2.1      htmltools_0.5.5
[41] rvest_1.0.3       colorspace_2.1-0  utf8_1.2.3        stringi_1.7.12
[45] munsell_0.5.0
```

References

Baker, William L, Erica L Baker, and Craig I Coleman. 2009. "Pharmacologic Treatments for Chronic Obstructive Pulmonary Disease: A Mixed-Treatment Comparison Meta-Analysis." *Pharmacotherapy* 29 (8): 891–905. https://doi.org/10.1592/phco.29.8.891.

Panaccione, Remo, Eric B Collins, Gil Y Melmed, Severine Vermeire, Silvio Danese, Peter D R Higgins, Christina S Kwon, et al. 2023. "Efficacy and Safety of Advanced Therapies for Moderately to Severely Active Ulcerative Colitis at Induction and Maintenance: An Indirect Treatment Comparison Using Bayesian Network Meta-Analysis." *Crohn's & Colitis 360* 5 (2). https://doi.org/10.1093/crocol/otad009.

Selvarajan, Sandhiya, Annuja Anandaradje, Santhosh Shivabasappa, Deepthy Melepurakkal Sadanandan, N. Sreekumaran Nair, and Melvin George. 2022. "Efficacy of Pharmacological Interventions in COVID-19: A Network Meta-Analysis." *British Journal of Clinical Pharmacology* 88 (9): 4080–91. https://doi.org/10.1111/bcp.15338.

Siemieniuk, Reed AC, Jessica J Bartoszko, Dena Zeraatkar, Elena Kum, Anila Qasim, Juan Pablo Díaz Martinez, Ariel Izcovich, et al. 2020. "Drug Treatments for Covid-19: Living Systematic Review and Network Meta-Analysis." *BMJ*, July, m2980. https://doi.org/10.1136/bmj.m2980.