Main Analysis

Nayeon Kwon

2024-07-30

Load data

```r
# Load necessary libraries
# Uncomment the following line to install any missing packages
# install.packages(c("tidyverse", "readxl", "haven", "CBPS", "lubridate", "skimr", "tableone",
#                    "survey", "senstrat", "MASS", "ggdag", "dagitty", "broom", "scales",
#                    "truncnorm", "ipw", "WeightIt", "cobalt", "optmatch", "MatchIt",
#                    "DOS2", "Matching", "ggplot2", "sensitivityfull", "sensitivitymw",
#                    "lmtest", "sandwich", "Rglpk"))

# Load Data
library(tidyverse)
library(readxl)
library(haven)

# Read data files
extra2 <- read_excel("data/extra2.xlsx")
w7_2014_data_220324 <- read_dta("data/w7_2014_data_220324.dta")
shp2 <- read.csv("data/shp2.csv", header = TRUE, fileEncoding = "euc-kr")
```

Load packages

```r
# Load Packages
library(lubridate)
library(skimr)
library(tableone)
library(survey)
library(senstrat)
library(MASS)
library(ggdag)  # For plotting DAGs
library(dagitty)  # For working with DAG logic
library(broom)  # augment_columns
library(scales)
library(truncnorm)
library(ipw)
library(WeightIt)
library(cobalt)
library(optmatch)
library(MatchIt)
library(DOS2)
library(Matching)
library(ggplot2)
library(sensitivityfull)
library(sensitivitymw)
library(lmtest) # coeftest
library(sandwich) # vcovCL
```

pre-processing

```
# Install spatial data-related packages if necessary
# Uncomment the following line to install any missing packages
# install.packages(c("raster", "sf", "tmap", "sp","spdep"))

# Load spatial data-related packages
library(raster)
library(sf)
library(tmap)
library(sp)
library(spdep)  # To find adjacent polygons

# Process extra data
extra <- extra2 %>%
  mutate(
    region = factor(DHu14region),  # Create a new variable
    region2 = factor(ifelse(gri == 40577, 1, ifelse(gri == 30892, 2, 3))),
    ca = factor(ca)
  ) %>%
  filter(
    !is.na(housing),  # Exclude rows with missing values in housing
    !is.na(ca)        # Exclude rows with missing values in ca (capital/non-capital region)
  )

# Merge extra with shp2
merged <- left_join(x = extra, y = shp2, by = 'region')
```

test

```
# Define the columns for latitude and longitude
ll <- c('x', 'y')

# Extract coordinates from the merged data
coords <- merged[ll]

# Calculate the distance from each point to every other point
merged$dist <- apply(coords, 1, function(eachPoint)
  spDistsN1(as.matrix(coords), eachPoint, longlat=TRUE)
)

# Create adjacency matrices based on a 5 km radius
## Ensure the unit is consistent (meters)
## Note: This simple calculation may not be accurate for polyhedral shapes
```

```r
merged$a <- merged$dist < 5000
diag(merged$a) <- NA

# Convert logical adjacency matrix to numeric
merged$adj5km <- merged$a * 1

# Display the adjacency matrix
merged$adj5km
```

pre-processing (continued)

```r
# install.packages("dplyr")
library(dplyr)
w7 = w7_2014_data_220324 %>%

  ###      (Creating new variables)
  mutate(
    #      (Child's happiness)
    y = JCh14shs001 + JCh14shs002 + JCh14shs003 + JCh14shs004 + EMt14shs005 + FFt14shs005,

    #       (Child's self-esteem)
    JCh14sfs = JCh14sfs012 + JCh14sfs013 + JCh14sfs014 + JCh14sfs015 + JCh14sfs016 + JCh14sfs017 +
               JCh14sfs018 + JCh14sfs019 + JCh14sfs020 + JCh14sfs021 + JCh14sfs022 + JCh14sfs023 +
               JCh14sfs024 + JCh14sfs025 + JCh14sfs026 + JCh14sfs027 + JCh14sfs028 + JCh14sfs029 +
               JCh14sfs030 + JCh14sfs031 + JCh14sfs032 + JCh14sfs033 + JCh14sfs034 + JCh14sfs035 + JCh1

    #       (     ) (Child's academic ability: literacy and language skills)
    HIn14acs = HIn14acs001 + HIn14acs002 + HIn14acs003 + HIn14acs004 + HIn14acs005 + HIn14acs006 +
               HIn14acs007 + HIn14acs008 + HIn14acs009 + HIn14acs010 + HIn14acs011 + HIn14acs012 +
               HIn14acs013 + HIn14acs014,

    #       ( ) (Child's social competence: assertiveness)
    HIn14ssr = HIn14ssr002 + HIn14ssr008 + HIn14ssr017 + HIn14ssr018 + HIn14ssr021,

    #       (Mother's parenting efficacy)
    EMt14sff = EMt14sff012 + EMt14sff013 + EMt14sff014 + EMt14sff015 + EMt14sff016 + EMt14sff017 +
               EMt14sff018 + EMt14sff019 + EMt14sff020 + EMt14sff021 + EMt14sff022 + EMt14sff023 +
               EMt14sff024 + EMt14sff025 + EMt14sff026 + EMt14sff027,

    #     (Mother's happiness)
    EMt14shs = EMt14shs001 + EMt14shs002 + EMt14shs003 + EMt14shs004,

    #        (Mother's controlling parenting behavior)
    EMt14crs = EMt14crs010 + EMt14crs011 + EMt14crs012 + EMt14crs013 + EMt14crs014 + EMt14crs015,

    #          (Mother's affectionate co-parenting behavior)
    EMt14aff = EMt14crs035 + EMt14crs037 + EMt14crs038 + EMt14crs039,

    #          (Mother's integrative co-parenting behavior)
    EMt14integ = EMt14crs046 + EMt14crs047 + EMt14crs048,

    #       (Father's parenting stress)
    FFt14prs = FFt14prs001 + FFt14prs002 + FFt14prs003 + FFt14prs004 + FFt14prs005 + FFt14prs006 +
               FFt14prs007 + FFt14prs008 + FFt14prs009 + FFt14prs010 + FFt14prs011,

    #     (Father's happiness)
    FFt14shs = FFt14shs001 + FFt14shs002 + FFt14shs003 + FFt14shs004,

    #          (Father's affectionate co-parenting behavior)
    FFt14aff = FFt14crs035 + FFt14crs037 + FFt14crs038 + FFt14crs039,

    #       (Child's preference for institution)
    HIn14chc = HIn14chc037 + HIn14chc038,

    #           (Mother's final education level: high school or below)
    DMt14dmg014 = factor(ifelse(DMt14dmg014 %in% c(1,2,3,4), 4, DMt14dmg014)),

    #           (Father's final education level: high school or below)
    DFt14dmg014 = factor(ifelse(DFt14dmg014 %in% c(3,4), 4, DFt14dmg014)),

    # //    (City/district variable)
    region = factor(DHu14cmm015),

    #          (Household safety in terms of security)
    EHu14cmm011 = factor(EHu14cmm011),

    #          (Household convenience in park usage)
    EHu14cmm009b = factor(EHu14cmm009b),

    #              (Use of guardian's childcare support services: part-time special academies and specia
    DCh14cht001 = factor(DCh14cht001),

    #        (1   () ) (Household park usage days in a month)
    EHu14cmm017b = factor(EHu14cmm017b),

    #         (    () ) (Child's outdoor activities on weekdays: total hours)
    DCh14dlc010 = factor(DCh14dlc010),

    #                     (Current use of part-time special academies and special activities by guardia
    DCh14cht002 = factor(DCh14cht002)
  ) %>%

  ##    filtering (Filtering for analysis)
  filter(
    JCh14int001 == 1,                                     #    7    (Child participated in the 7th surve
    EMt14int001 == 1,                                     #     7    (Mother participated in the 7th sur
    FFt14int001 == 1,                                     #     7    (Father participated in the 7th sur
    DHu14int001 == 1,                                     #     7    (Guardian participated in the 7th s
    HIn14int001 == 1 | HIn14int001 == 2,                  #                   (Exclude children attendin
    !is.na(JCh14sfs012),                                  #       1     (Exclude missing values for chil
    !is.na(ICh14ssr027) | ICh14ssr027 != 99999999,        #        :   1     (Exclude missing values for
    !is.na(ICh14ssr030) | ICh14ssr030 != 99999999,        #        :   2     (Exclude missing values for
    !is.na(ICh14ssr034) | ICh14ssr034 != 99999999,        #        :   4     (Exclude missing values for
    !is.na(ICh14ssr042) | ICh14ssr042 != 99999999,        #        :   5     (Exclude missing values for
    DHu14ses006 != 88888888 | DHu14ses006 != 99999999,    #         (Exclude missing values for househo
    !is.na(DFt14dmg014),                               #           (Exclude missing values for father
    !is.na(DMt14dmg014)                                #           (Exclude missing values for mother
  )

## merge data (Merging data)
merged <- merge(x=w7, y=extra, by='region', all.x=TRUE)
# write.csv(merged, "/Users/user02/R/merged.csv")

##           (Removing unfiltered missing values)
## extra data frame     (Add to the extra data frame)

merged = merged %>%
  filter(
    !is.na(papc),                          # Exclude missing values for 'papc'
    ICh14ssr027 != 99999999,               # Exclude invalid values for child's social competence:
    ICh14ssr030 != 99999999,               # Exclude invalid values for child's social competence:
    ICh14ssr034 != 99999999,               # Exclude invalid values for child's social competence:
    ICh14ssr042 != 99999999,               # Exclude invalid values for child's social competence:
    DHu14ses006 != 88888888,               # Exclude invalid values for household income
    DHu14ses006 != 99999999                # Exclude invalid values for household income
  )
```

Treatment allocation is based on the median

```
# Convert 'papc' into a binary variable based on its median value
# If 'papc' is less than or equal to the median value, assign 0; otherwise, assign 1
merged$papc = ifelse(merged$papc <= median(merged$papc), 0, 1)

## table   missing value
# Attach the 'merged' data frame to the R search path for easy variable access
attach(merged)

# Display a frequency table of 'papc' to see the counts of 0s and 1s
table(papc)            # Frequency table for 'papc'
```

```
## papc
##   0   1
## 512 441
```

```
# Check the number of missing values in the 'y' variable
sum(is.na(y))          # Total number of missing values in 'y'
```

```
## [1] 0
```

```
# Display frequency tables for variables with potential invalid values (99999999)
table(ICh14ssr027)     # Frequency table for 'ICh14ssr027', excluding 99999999
```

```
## ICh14ssr027
##   1   2   3
## 203 569 181
```

```
table(ICh14ssr030)     # Frequency table for 'ICh14ssr030', excluding 99999999
```

```
## ICh14ssr030
##   1   2   3
##  45 545 363
```

```r
table(ICh14ssr034)      # Frequency table for 'ICh14ssr034', excluding 99999999
```

```
## ICh14ssr034
##   1   2   3 
## 118 550 285
```

```r
table(ICh14ssr042)      # Frequency table for 'ICh14ssr042', excluding 99999999
```

```
## ICh14ssr042
##   1   2   3 
##  10 285 658
```

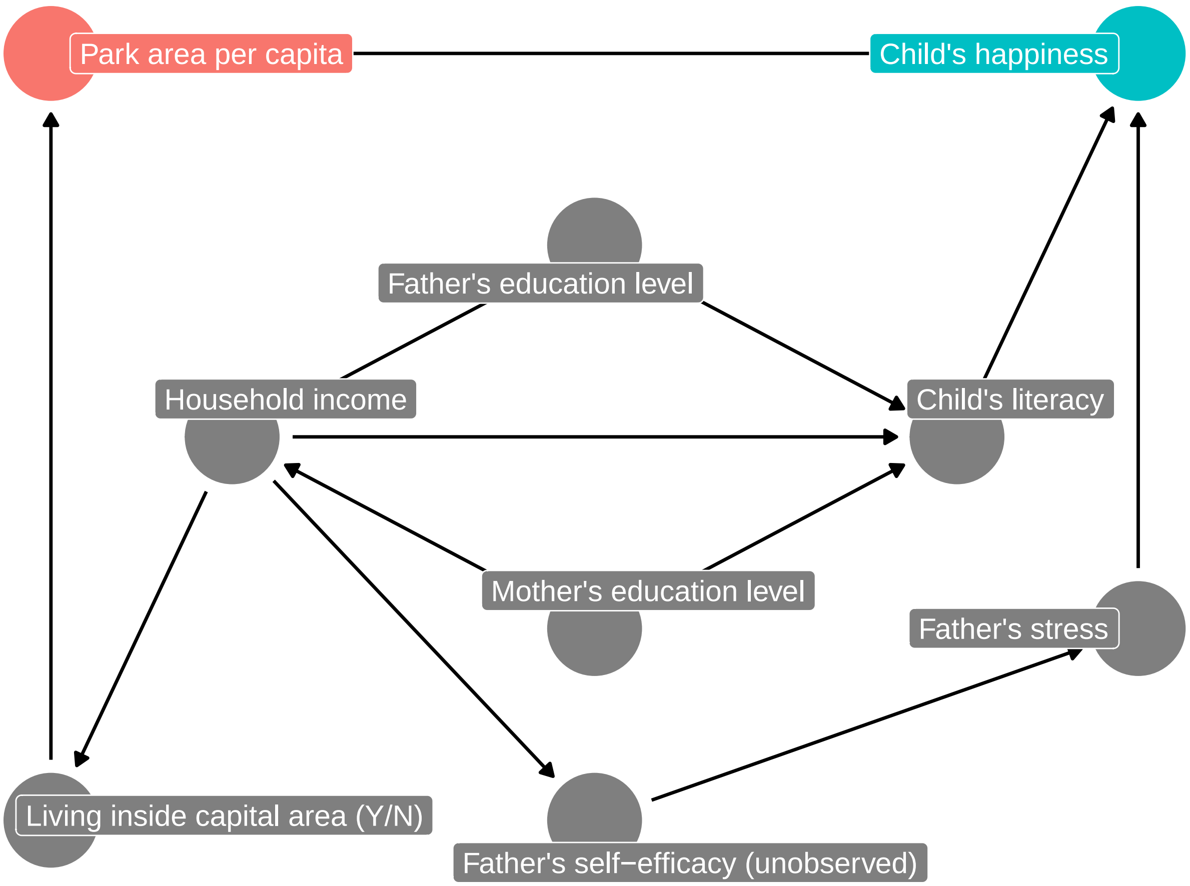
```r
table(DHu14ses006)      # Frequency table for 'DHu14ses006', excluding 99999999
```

```
## DHu14ses006
##   100  130  140  150  160  170  180  190  200  220  230  240  250  260  270  279 
##     8    1    1    5    4    3    2    1   36    5    3    5   48    4    7    1 
##   280  290  300  310  320  330  350  360  370  380  390  400  410  420  430  450 
##     5    4  115    1    5    8   76    6    7   10    1  145    1    6    5   50 
##   460  470  480  500  520  540  550  560  570  600  620  630  650  660  680  700 
##     4    3    8  150    2    1   15    1    3   74    1    1    5    1    1   41 
##   740  750  800  850  900  950 1000 1200 1500 
##     1    4   27    1    9    1   17    1    2
```

```r
# Detach the 'merged' data frame from the R search path
detach(merged)
```

Generate DAG

```r
# generate DAG
# Load necessary libraries
library(ggdag)
library(dplyr)

# Define the Directed Acyclic Graph (DAG) structure
papc.dag <- dagify(
  # Define relationships between variables
  y ~ x + f + g,
  x ~ b,
  b ~ c,
  c ~ d + e,
  f ~ c + d + e,
  u2 ~ c,
  g ~ u2,

  # Specify exposure and outcome variables
  exposure = "x",
  outcome = "y",

  # Define labels for the variables
  labels = c(
    y = "Child’s happiness",
    x = "Park area per capita",
    b = "Living inside capital area (Y/N)",
    c = "Household income",
    d = "Father’s education level",
    e = "Mother’s education level",
    f = "Child's literacy",
    g = "Father's stress",
    u2 = "Father's self-efficacy (unobserved)"
  ),

  # Define coordinates for the DAG plot
  coords = list(
    x = c(
      x = 1, b = 1, c = 2, d = 4, e = 4, f = 6, g = 7, u2 = 4, y = 7
    ),
    y = c(
      x = 6, b = 2, c = 4, d = 5, e = 3, f = 4, g = 3, u2 = 2, y = 6
    )
  )
)

# Notes on variables
# U1: Mother's self-efficacy (commented out in DAG)
# U2: Father's self-efficacy

# Visualize the DAG
ggdag_status(papc.dag, use_labels = "label", text = FALSE) +
  guides(fill = FALSE, color = FALSE) +  # Disable the legend
  theme_dag()
```



Data frame 1

```
# Extract variables from the merged dataset
y <- merged$y                     # Child’s happiness
papc <- merged$papc               # Treatment: Park area per capita
housing <- merged$housing         # Housing information
# region2 <- merged$region2       # (Commented out) Region information
ca <- merged$ca                   # Categorical variable (e.g., region or category)
income <- merged$DHu14ses006      # Household income
Fschool <- merged$DFt14dmg014     # Father's final education level
Mschool <- merged$DMt14dmg014     # Mother's final education level
Kacs <- merged$HIn14acs           # Child's academic ability (literacy and language)
Fstress <- merged$FFt14prs        # Father's parenting stress
Msff <- merged$EMt14sff           # Mother's parenting efficacy
Mhappiness <- merged$EMt14shs     # Mother's happiness
Mcrs <- merged$EMt14crs           # Mother's controlling parenting behavior
Maff <- merged$EMt14aff           # Mother's affectionate parenting behavior
Minteg <- merged$EMt14integ       # Mother's integrated parenting behavior
Fhappiness <- merged$FFt14shs     # Father’s happiness
Faff <- merged$FFt14aff           # Father’s affectionate parenting behavior
Ksfs <- merged$JCh14sfs           # Child’s self-esteem
Kssr <- merged$HIn14ssr           # Child’s social competence
Kprefe <- merged$HIn14chc         # Child’s institutional preference
```

```
# Combine the variables into a data frame
PSKC.data <- data.frame(
  y = y,
  papc = papc,
  housing = housing,
  ca = ca,
  income = income,
  Fschool = Fschool,
  Mschool = Mschool,
  Kacs = Kacs,
  Fstress = Fstress,
  Msff = Msff,
  Mhappiness = Mhappiness,
  Mcrs = Mcrs,
  Maff = Maff,
  Minteg = Minteg,
  Fhappiness = Fhappiness,
  Faff = Faff,
  Ksfs = Ksfs,
  Kssr = Kssr,
  Kprefe = Kprefe
)

# Convert specific columns to factors
for (i in c(4, 6:7)) {
  PSKC.data[[colnames(PSKC.data)[i]]] <- factor(PSKC.data[[colnames(PSKC.data)[i]]])
}

# Display summary statistics of the data frame
summary(PSKC.data)
```

```
##        y              papc           housing       ca          income
##  Min.   :15.00   Min.   :0.0000   Min.   : 75462   0:557   Min.   : 100
##  1st Qu.:25.00   1st Qu.:0.0000   1st Qu.:158304   1:396   1st Qu.: 300
##  Median :26.00   Median :0.0000   Median :190535           Median : 400
##  Mean   :26.03   Mean   :0.4627   Mean   :234325           Mean   : 442
##  3rd Qu.:28.00   3rd Qu.:1.0000   3rd Qu.:257737           3rd Qu.: 500
##  Max.   :30.00   Max.   :1.0000   Max.   :919560           Max.   :1500
##  Fschool Mschool      Kacs          Fstress            Msff
##  4:255   4:277   Min.   :10.00   Min.   :11.00   Min.   :16.00
##  5:208   5:271   1st Qu.:48.00   1st Qu.:22.00   1st Qu.:47.00
##  6:402   6:356   Median :56.00   Median :26.00   Median :49.00
##  7: 88   7: 49   Mean   :53.58   Mean   :26.03   Mean   :49.06
##                  3rd Qu.:62.00   3rd Qu.:31.00   3rd Qu.:52.00
##                  Max.   :70.00   Max.   :45.00   Max.   :65.00
##    Mhappiness        Mcrs            Maff          Minteg        Fhappiness
##  Min.   : 5.00   Min.   : 6.00   Min.   : 4.0   Min.   : 3.00   Min.   : 7.00
##  1st Qu.:18.00   1st Qu.:19.00   1st Qu.:20.0   1st Qu.:14.00   1st Qu.:18.00
##  Median :21.00   Median :21.00   Median :22.0   Median :17.00   Median :21.00
##  Mean   :21.03   Mean   :20.84   Mean   :22.2   Mean   :16.61   Mean   :21.32
##  3rd Qu.:24.00   3rd Qu.:23.00   3rd Qu.:25.0   3rd Qu.:19.00   3rd Qu.:24.00
##  Max.   :28.00   Max.   :30.00   Max.   :28.0   Max.   :21.00   Max.   :28.00
##       Faff            Ksfs            Kssr           Kprefe
##  Min.   : 4.00   Min.   :43.00   Min.   : 5.00   Min.   : 5.000
##  1st Qu.:18.00   1st Qu.:55.00   1st Qu.:11.00   1st Qu.: 8.000
##  Median :21.00   Median :60.00   Median :13.00   Median : 8.000
##  Mean   :20.82   Mean   :60.29   Mean   :12.37   Mean   : 8.418
##  3rd Qu.:24.00   3rd Qu.:65.00   3rd Qu.:14.00   3rd Qu.: 9.000
##  Max.   :28.00   Max.   :81.00   Max.   :15.00   Max.   :10.000
```

```
# Feature property: Examine categorical variables
PSKC.data %>% skim(ca, Fschool, Mschool)
```

Table 1: Data summary

Name	Piped data
Number of rows	953
Number of columns	19
Column type frequency:	
factor	3
Group variables	None

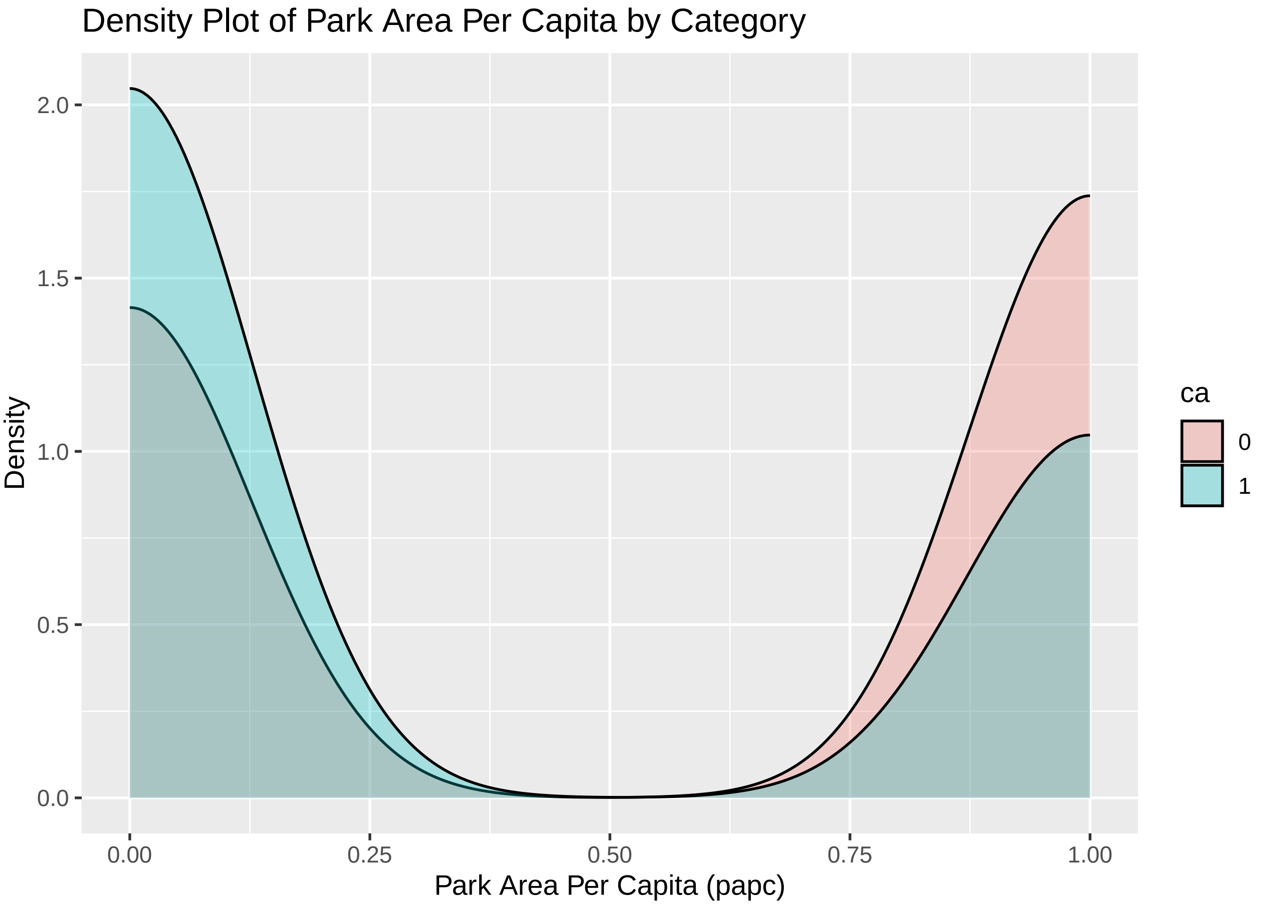
Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
ca	0	1	FALSE	2	0: 557, 1: 396
Fschool	0	1	FALSE	4	6: 402, 4: 255, 5: 208, 7: 88
Mschool	0	1	FALSE	4	6: 356, 4: 277, 5: 271, 7: 49

Create the dataframe with selected features and transformations

```
# The commented code section shows the previous approach for reference

# # Select relevant features and transform the data
# dat <- merged %>%
#   dplyr::select(
#     y, papc, housing, ca, DHu14ses006, DFt14dmg014, DMt14dmg014,
#     HIn14acs, FFt14prs, EMt14sff, EMt14shs, EMt14crs, EMt14aff,
#     EMt14integ, FFt14shs, FFt14aff, JCh14sfs, HIn14ssr, HIn14chc
#   ) %>%
#   mutate(TrtLevel = ifelse(papc == 1, "1", "0"))
#
# # Rename columns for clarity
# colnames(dat) <- c(
#   "y", "trt", "housing", "ca", "income", "Fschool", "Mschool", "Kacs", "Fstress",
#   "Msff", "Mhappiness", "Mcrs", "Maff", "Minteg", "Fhappiness", "Faff", "Ksfs", "Kssr", "Kprefe", "Tr
# )
#
# # Convert specific columns to factors
# for (i in c(4, 6:7)) {
#   dat[[colnames(dat)[i]]] <- factor(dat[[colnames(dat)[i]]])
# }
#
```

```r
# # Display summary statistics of the data frame
# summary(dat)
#
# # Examine categorical variables
# dat %>% skim(ca, Fschool, Mschool)

# Extract and plot the treatment and control groups

# Plot density of 'papc' by 'ca' with color fill
ggplot(merged, aes(x = papc, fill = ca)) +
  geom_density(alpha = 0.3) +
  xlab("Park Area Per Capita (papc)") +
  ylab("Density") +
  ggtitle("Density Plot of Park Area Per Capita by Category")
```



Add Propensity Score (PS) and Covariate Balancing Propensity Score (CBPS) to the dataframe

```r
# Load necessary library
library(CBPS)

# Compute and add PS and CBPS to the dataframe
PSKC.data <- PSKC.data %>%
  mutate(
    # Calculate Propensity Score (PS) using logistic regression
    ps = predict(
      glm(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
          data = PSKC.data,
          family = binomial),
      type = "response"
    ),

    # Calculate Covariate Balancing Propensity Score (CBPS)
    cbps = CBPS(
      factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
      ATT = 0,
      data = PSKC.data
    )$fitted.values
  )

# Summarize the newly added variables 'ps' and 'cbps'
PSKC.data %>% skim(ps, cbps)
```

Table 3: Data summary

Name	Piped data
Number of rows	953
Number of columns	21
Column type frequency:	
numeric	2
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
ps	0	1	0.46	0.12	0.21	0.35	0.5	0.56	0.74	
cbps	0	1	0.47	0.11	0.25	0.37	0.5	0.57	0.69	

Add treatment level to the dataframe

```
# Convert 'papc' to a factor and add it as 'TrtLevel'
PSKC.data <- PSKC.data %>%
  mutate(
    TrtLevel = factor(papc)  # Convert 'papc' to a factor for treatment level
  )

# Summarize the 'TrtLevel' variable
PSKC.data %>% skim(TrtLevel)
```

Table 5: Data summary

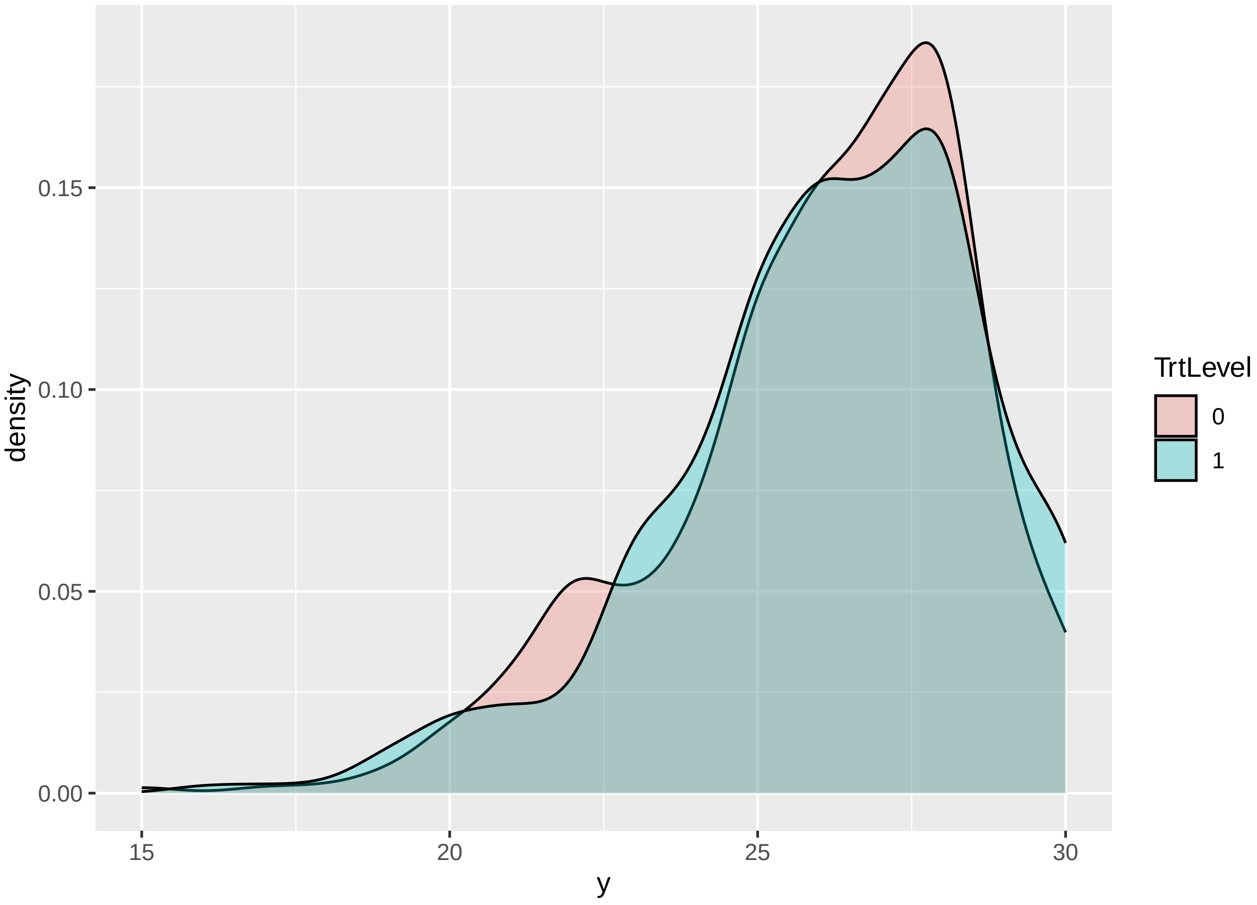
Name	Piped data
Number of rows	953
Number of columns	22
Column type frequency:	
factor	1
Group variables	None

Variable type: factor

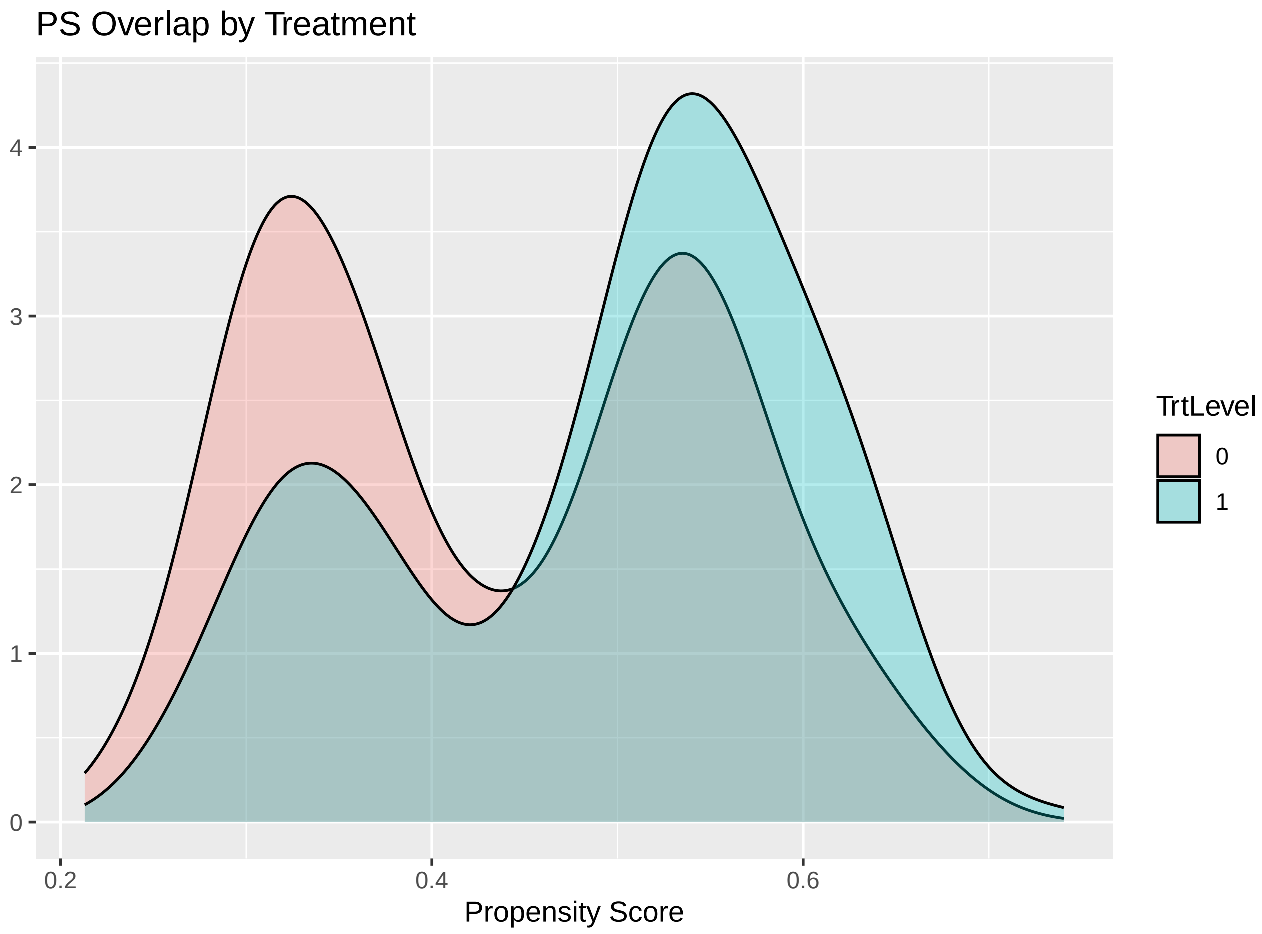
skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
TrtLevel	0	1	FALSE	2	0: 512, 1: 441

Plot distributions by treatment level

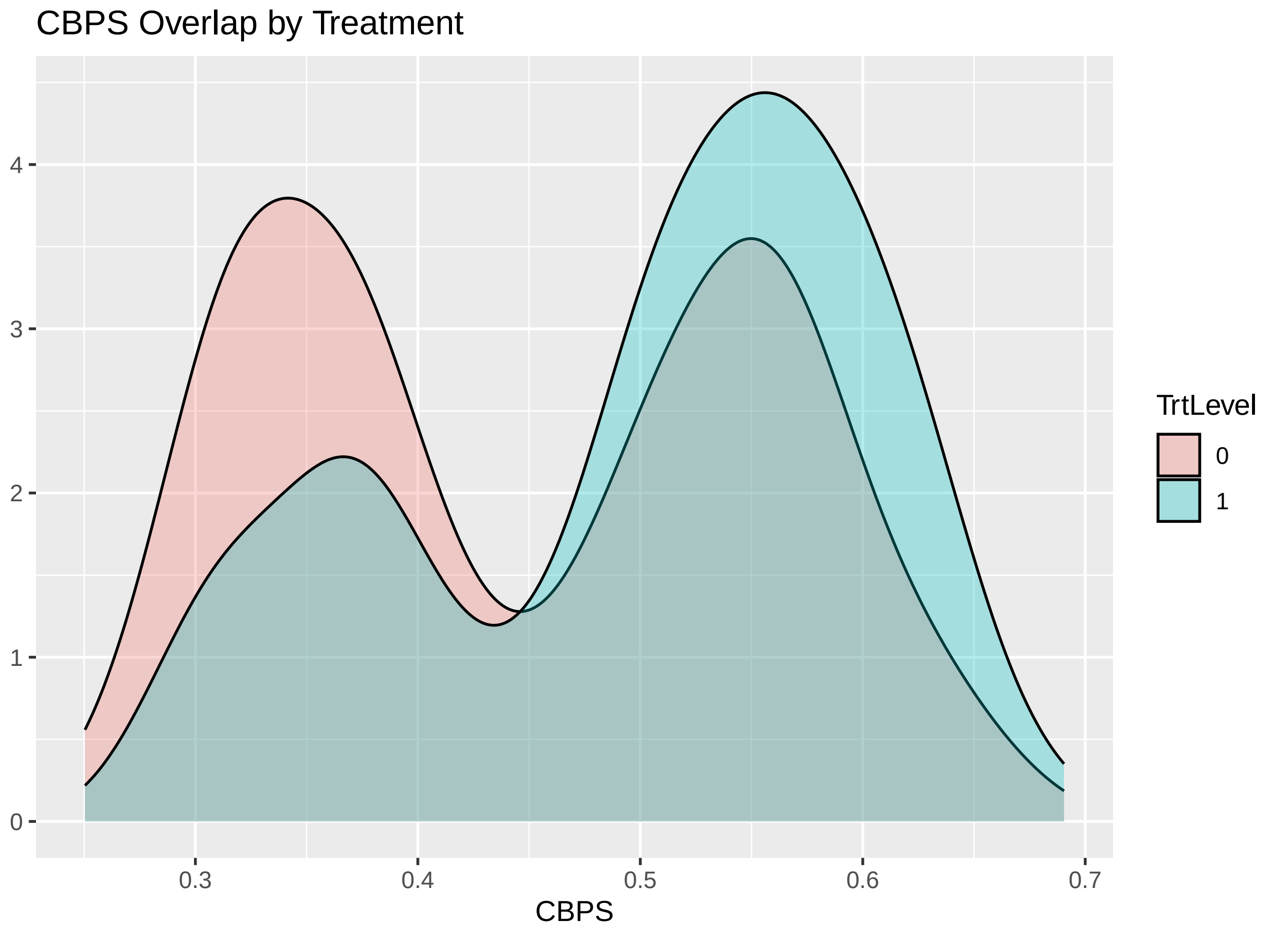
```r
# Plot for outcome 'y'
ggplot(PSKC.data, aes(y, fill = TrtLevel)) +
  geom_density(alpha = 0.3) +
  xlab("y")
```



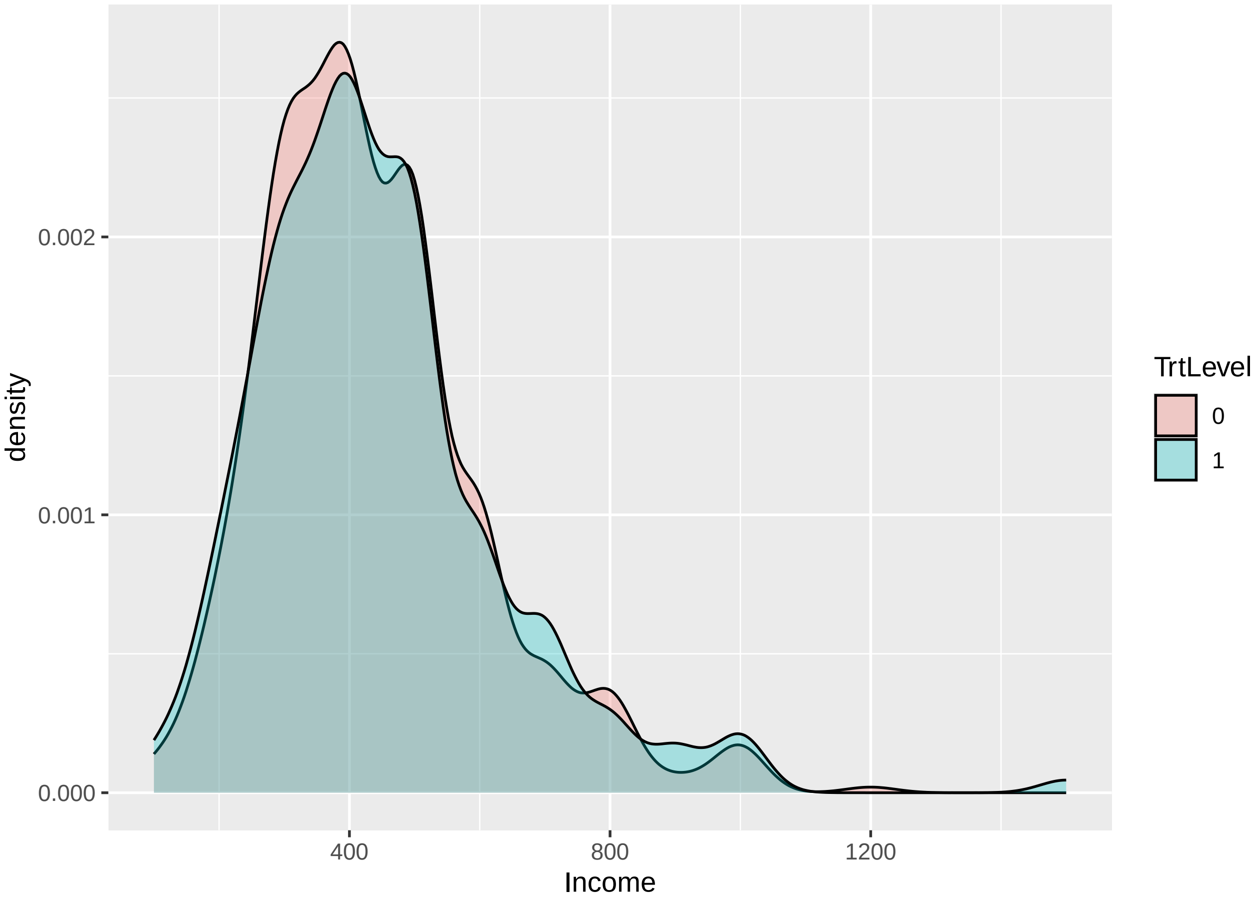
```
# Plot for Propensity Score (PS)
ggplot(PSKC.data, aes(ps, fill = TrtLevel)) +
  geom_density(alpha = 0.3) +
  labs(x = "Propensity Score", title = "PS Overlap by Treatment", y = "")
```



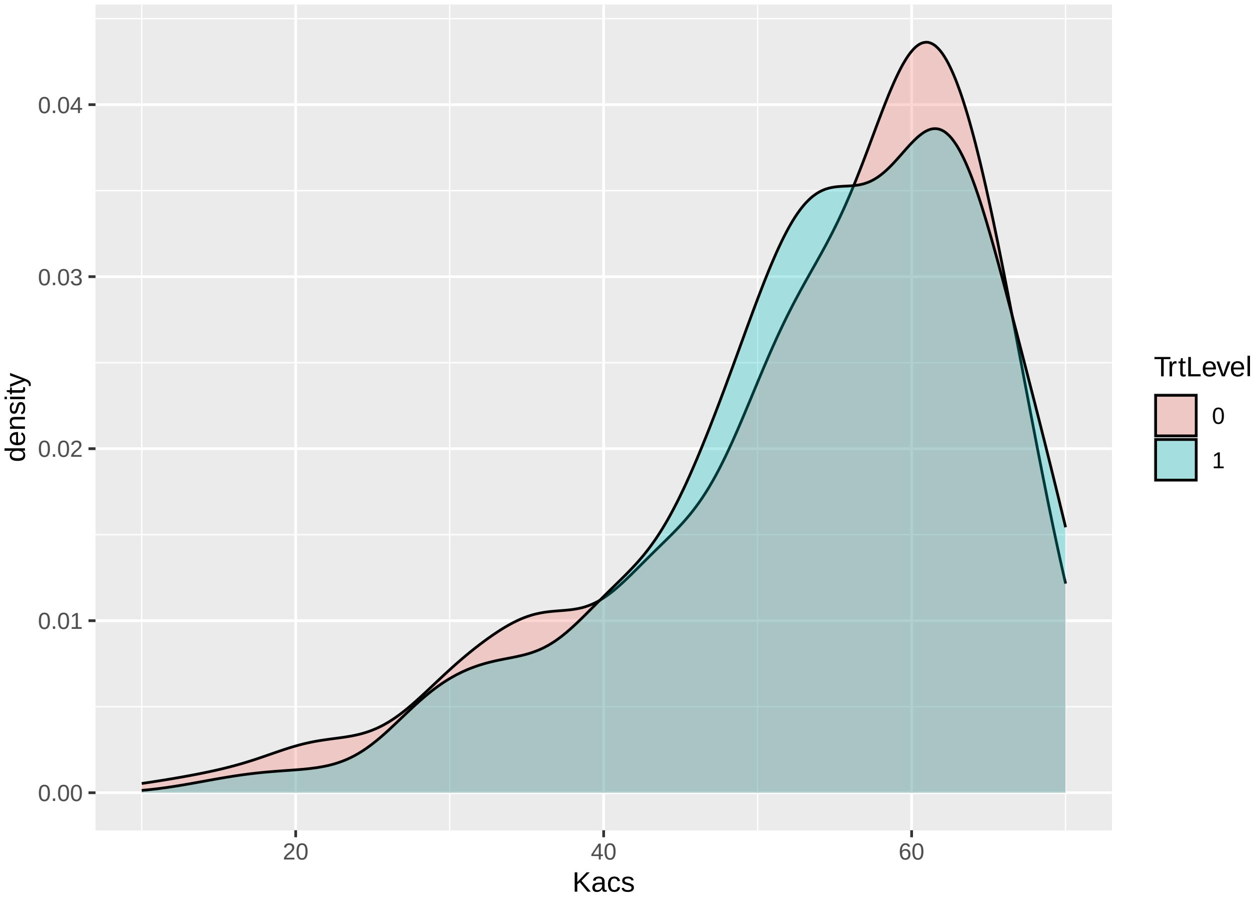
```
# Plot for Covariate Balance Propensity Score (CBPS)
ggplot(PSKC.data, aes(cbps, fill = TrtLevel)) +
  geom_density(alpha = 0.3) +
  labs(x = "CBPS", title = "CBPS Overlap by Treatment", y = "")
```



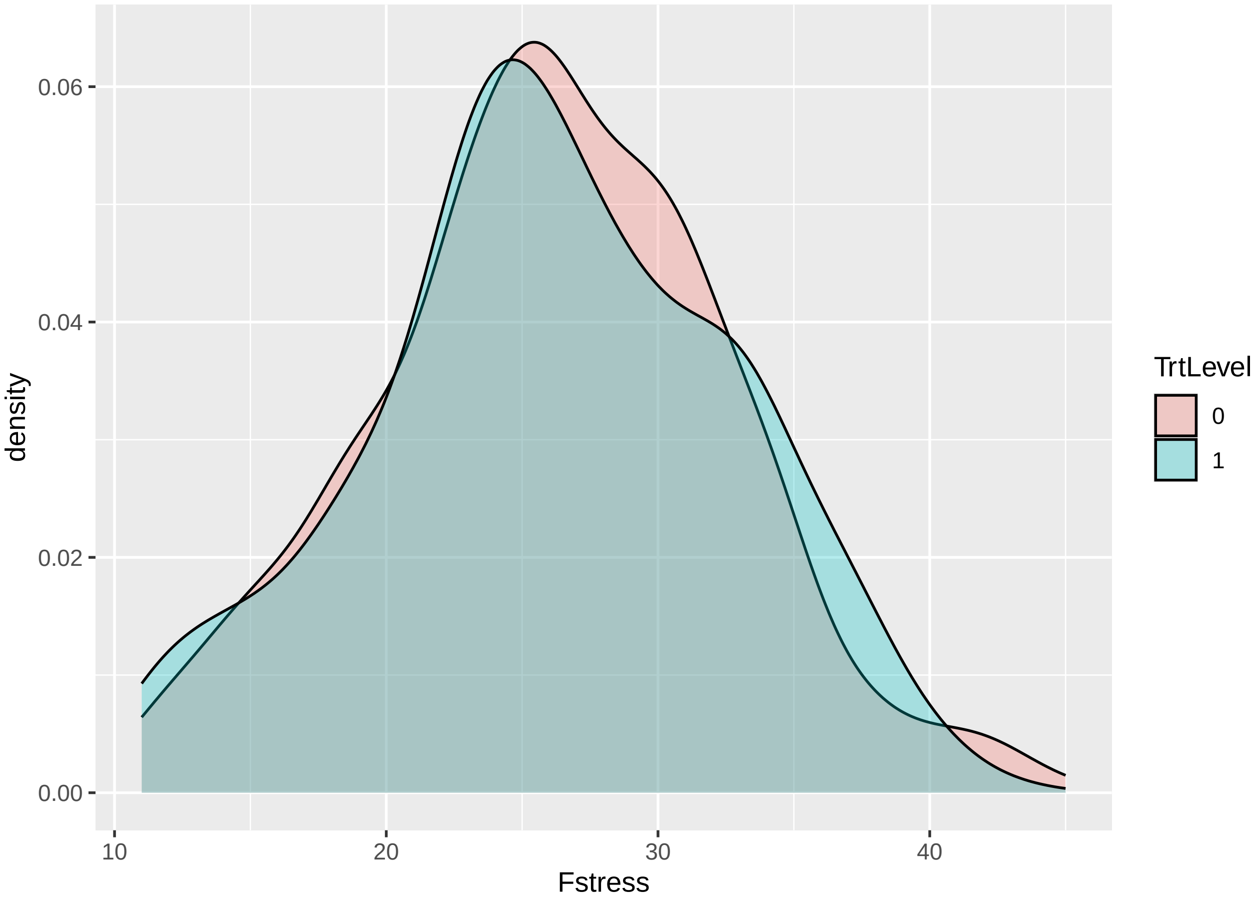
```
# Plot for income
ggplot(PSKC.data, aes(income, fill = TrtLevel)) +
  geom_density(alpha = 0.3) +
  xlab("Income")
```



```
# Plot for academic ability 'Kacs'
ggplot(PSKC.data, aes(Kacs, fill = TrtLevel)) +
  geom_density(alpha = 0.3) +
  xlab("Kacs")
```



```
# Plot for father's stress 'Fstress'
ggplot(PSKC.data, aes(Fstress, fill = TrtLevel)) +
  geom_density(alpha = 0.3) +
  xlab("Fstress")
```



```
# Proportion of 'ca' by treatment level
df <- PSKC.data %>%
  group_by(TrtLevel, ca) %>%
  summarise(count = n()) %>%
  mutate(freq = count / sum(count))
df
```

```
## # A tibble: 4 x 4
## # Groups:   TrtLevel [2]
##   TrtLevel ca    count  freq
##   <fct>    <fct> <int> <dbl>
## 1 0        0       250 0.488
## 2 0        1       262 0.512
## 3 1        0       307 0.696
## 4 1        1       134 0.304
```

```
# Proportion of 'Fschool' by treatment level
df2 <- PSKC.data %>%
  group_by(TrtLevel, Fschool) %>%
  summarise(count = n()) %>%
  mutate(freq = count / sum(count))
df2
```

```
## # A tibble: 8 x 4
## # Groups:   TrtLevel [2]
##   TrtLevel Fschool count   freq
##   <fct>    <fct>   <int>  <dbl>
## 1 0        4          127 0.248
## 2 0        5          103 0.201
## 3 0        6          229 0.447
## 4 0        7           53 0.104
## 5 1        4          128 0.290
## 6 1        5          105 0.238
## 7 1        6          173 0.392
## 8 1        7           35 0.0794
```

```
# Proportion of 'Mschool' by treatment level
df3 <- PSKC.data %>%
  group_by(TrtLevel, Mschool) %>%
  summarise(count = n()) %>%
  mutate(freq = count / sum(count))
df3
```

```
## # A tibble: 8 x 4
## # Groups:   TrtLevel [2]
##   TrtLevel Mschool count   freq
##   <fct>    <fct>   <int>  <dbl>
## 1 0        4         153 0.299
## 2 0        5         130 0.254
## 3 0        6         203 0.396
## 4 0        7          26 0.0508
## 5 1        4         124 0.281
## 6 1        5         141 0.320
## 7 1        6         153 0.347
## 8 1        7          23 0.0522
```

Regression adjustment

```
# Fit linear model
lm1 <- lm(
  y ~ factor(papc) + housing + ca + income + Fschool + Mschool + Kacs + Fstress +
    Msff + Mhappiness + Mcrs + Maff + Minteg + Fhappiness + Faff + Ksfs + Kssr + Kprefe,
  data = PSKC.data
)

# Display model summary
summary(lm1)
```

```
##
## Call:
## lm(formula = y ~ factor(papc) + housing + ca + income + Fschool +
##     Mschool + Kacs + Fstress + Msff + Mhappiness + Mcrs + Maff +
##     Minteg + Fhappiness + Faff + Ksfs + Kssr + Kprefe, data = PSKC.data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.1033  -1.2610   0.3166   1.5802   4.9718
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.010e+01  1.645e+00  18.297  < 2e-16 ***
## factor(papc)1  9.564e-02  1.559e-01   0.613  0.53980
## housing        1.012e-06  6.819e-07   1.483  0.13832
## ca1           -4.098e-01  1.839e-01  -2.229  0.02608 *
## income         8.236e-05  4.444e-04   0.185  0.85299
## Fschool5       1.891e-01  2.274e-01   0.832  0.40584
## Fschool6       3.703e-01  2.262e-01   1.637  0.10196
## Fschool7       5.521e-01  3.361e-01   1.642  0.10083
## Mschool5      -5.847e-02  2.163e-01  -0.270  0.78696
## Mschool6      -3.943e-01  2.333e-01  -1.690  0.09128 .
## Mschool7      -1.100e+00  4.137e-01  -2.658  0.00799 **
## Kacs          -2.933e-03  6.689e-03  -0.438  0.66116
## Fstress       -6.238e-02  1.383e-02  -4.510 7.32e-06 ***
## Msff          -4.763e-02  2.017e-02  -2.362  0.01838 *
## Mhappiness     5.699e-02  2.105e-02   2.707  0.00692 **
## Mcrs          -4.425e-02  2.602e-02  -1.701  0.08927 .
## Maff           8.683e-02  2.675e-02   3.247  0.00121 **
## Minteg         3.895e-02  2.981e-02   1.307  0.19164
## Fhappiness     2.873e-02  2.351e-02   1.222  0.22209
## Faff           1.923e-02  2.167e-02   0.887  0.37511
## Ksfs          -9.531e-02  1.094e-02  -8.715  < 2e-16 ***
## Kssr           1.740e-02  3.610e-02   0.482  0.62985
## Kprefe         1.828e-01  7.114e-02   2.569  0.01035 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.248 on 930 degrees of freedom
## Multiple R-squared:  0.2274, Adjusted R-squared:  0.2091
## F-statistic: 12.44 on 22 and 930 DF,  p-value: < 2.2e-16
```

```
# Estimate outcomes under treatment (papc = 1) and control (papc = 0)
data1 <- PSKC.data
data1$papc <- 1
est.y1.lm <- predict(lm1, newdata = data1)

data0 <- PSKC.data
data0$papc <- 0
est.y0.lm <- predict(lm1, newdata = data0)

# Calculate and round Average Treatment Effect (ATE)
est.ATE.lm1 <- mean(est.y1.lm) - mean(est.y0.lm)
round(est.ATE.lm1, 3)
```

```
## [1] 0.096
```

```
# Display 95% confidence intervals for model coefficients
round(confint(lm1, level = 0.95), 3)
```

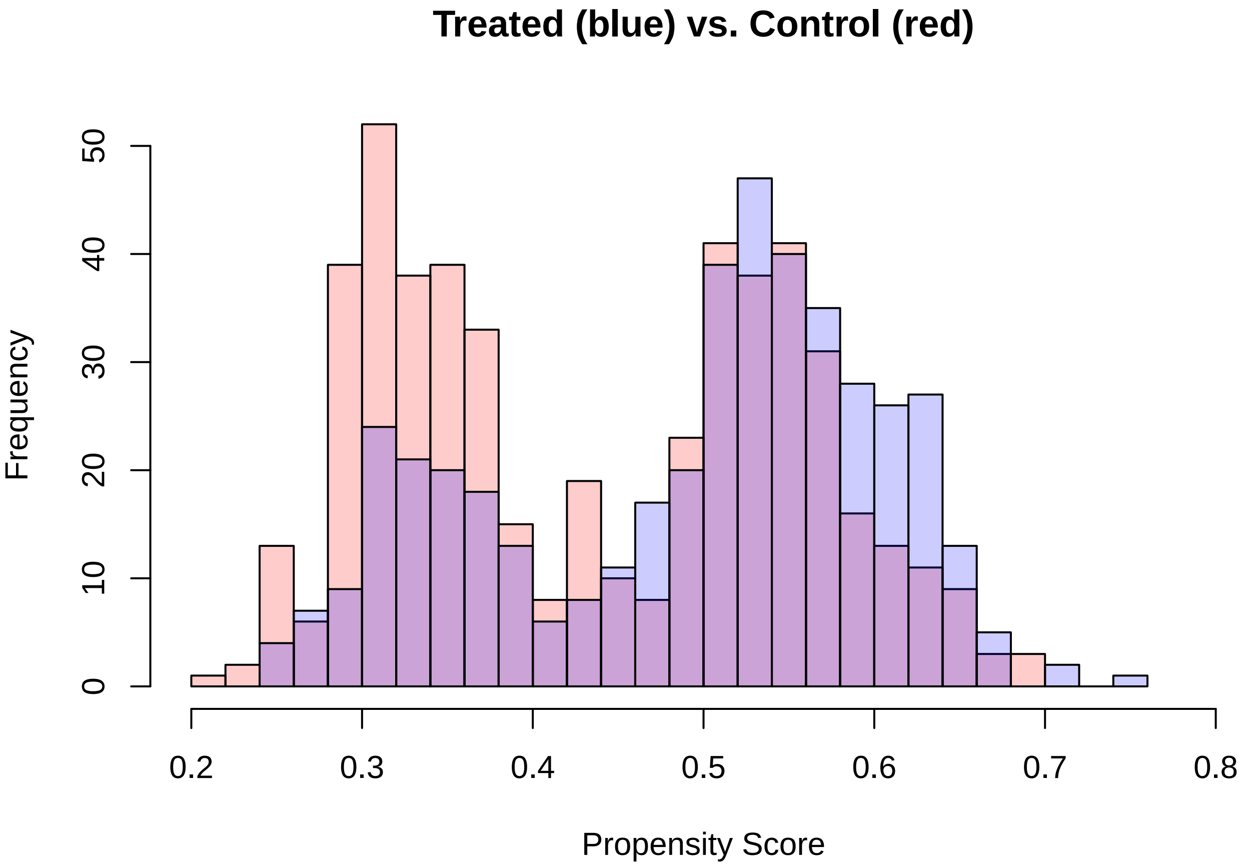
```
##                 2.5 % 97.5 %
## (Intercept)   26.873 33.330
## factor(papc)1 -0.210  0.402
## housing        0.000  0.000
## ca1           -0.771 -0.049
## income        -0.001  0.001
## Fschool5      -0.257  0.635
## Fschool6      -0.074  0.814
## Fschool7      -0.108  1.212
## Mschool5      -0.483  0.366
## Mschool6      -0.852  0.063
## Mschool7      -1.912 -0.288
## Kacs          -0.016  0.010
## Fstress       -0.090 -0.035
## Msff          -0.087 -0.008
## Mhappiness     0.016  0.098
## Mcrs          -0.095  0.007
## Maff           0.034  0.139
## Minteg        -0.020  0.097
## Fhappiness    -0.017  0.075
## Faff          -0.023  0.062
## Ksfs          -0.117 -0.074
## Kssr          -0.053  0.088
## Kprefe         0.043  0.322
```

```r
# Print specific coefficient value (example given)
round(1.559e-01, 3)
```

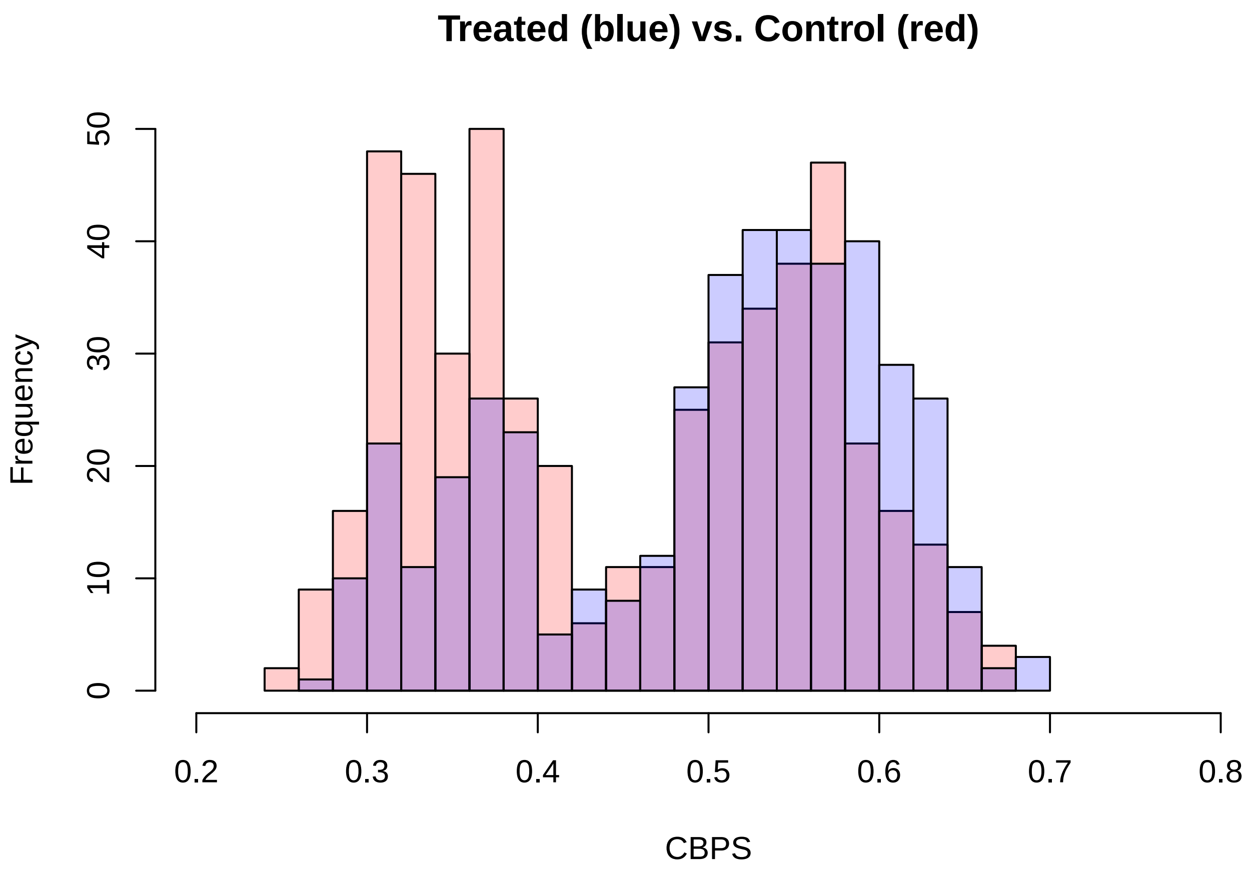
```
## [1] 0.156
```

propensity score model

```r
# Estimate propensity scores (PS) using logistic regression
propscore.model <- glm(
  factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  family = binomial(link = "logit"),
  x = TRUE,
  data = PSKC.data
)

# Estimate CBPS
cbps.model <- CBPS(
  factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  ATT = 0,
  data = PSKC.data
)

# Predict propensity scores and CBPS
est.ps <- predict(propscore.model, type = "response")
est.cbps <- predict(cbps.model, type = "response")

# Plot histograms for PS
hist(
  est.ps[PSKC.data$papc == 0],
  col = rgb(1, 0, 0, 0.2),
  breaks = 25,
  xlim = c(0.2, 0.8),
  xlab = "Propensity Score",
  main = "Treated (blue) vs. Control (red)"
)
hist(
  est.ps[PSKC.data$papc == 1],
  col = rgb(0, 0, 1, 0.2),
  breaks = 25,
  xlim = c(0.2, 0.8),
  add = TRUE
)
```



```
# Plot histograms for CBPS
hist(
  est.cbps[PSKC.data$papc == 0],
  col = rgb(1, 0, 0, 0.2),
  breaks = 25,
  xlim = c(0.2, 0.8),
  xlab = "CBPS",
  main = "Treated (blue) vs. Control (red)"
)
hist(
  est.cbps[PSKC.data$papc == 1],
  col = rgb(0, 0, 1, 0.2),
  breaks = 25,
  xlim = c(0.2, 0.8),
  add = TRUE
)
```



```
# Check min & max of PS
# Control group
round(max(est.ps[PSKC.data$papc == 0]), 3)
```

```
## [1] 0.686
```

```
round(min(est.ps[PSKC.data$papc == 0]), 3)
```

```
## [1] 0.213
```

```
# Treated group
round(max(est.ps[PSKC.data$papc == 1]), 3)
```

```
## [1] 0.74
```

```
round(min(est.ps[PSKC.data$papc == 1]), 3)
```

```
## [1] 0.25
```

ATE Calculation Using Weighting Methods by Hand with glm

```
# Calculate Propensity Score Weighting

# Length of the dataset
n <- length(papc)

# IPW (Inverse Probability Weighting)
IP.weight <- rep(NA, n)
IP.weight[papc == 1] <- 1 / est.ps[papc == 1]
IP.weight[papc == 0] <- -1 / (1 - est.ps[papc == 0])
est.ATE.IPW <- mean(IP.weight * y)

# Stabilized IPW (SIPW)
stabilized.IP.weight <- rep(NA, n)
stabilized.IP.weight[papc == 1] <- IP.weight[papc == 1] / sum(IP.weight[papc == 1])
stabilized.IP.weight[papc == 0] <- -IP.weight[papc == 0] / sum(IP.weight[papc == 0])
est.ATE.SPIW <- sum(stabilized.IP.weight * y)

# Doubly-Robust Estimator
est.y1.dr <- mean((papc * y - (papc - est.ps) * est.y1.lm) / est.ps)
est.y0.dr <- mean(((1 - papc) * y + (papc - est.ps) * est.y0.lm) / (1 - est.ps))
est.ATE.dr <- est.y1.dr - est.y0.dr

# Summary of results
round(c(est.ATE.lm1, est.ATE.IPW, est.ATE.SPIW, est.ATE.dr), 3)
```

```
## [1] 0.096 0.040 0.047 0.090
```

ATE Calculation Using Weighting Methods by Hand with CBPS

```
# CBPS model
cbps.model = CBPS(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress, ATT = 0, data = PSKC
est.cbps <- fitted(cbps.model)
# n <- length(papc)

## IPW
IP.weight <-rep(NA, n)
IP.weight[papc==1] <- 1/est.cbps[papc==1]
IP.weight[papc==0] <- -1/(1-est.cbps[papc==0])
est.ATE.IPW <- mean(IP.weight*y)

## SIPW
stabilized.IP.weight <- rep(NA,n)
stabilized.IP.weight[papc==1] <- IP.weight[papc==1]/sum(IP.weight[papc==1])
stabilized.IP.weight[papc==0] <- -IP.weight[papc==0]/sum(IP.weight[papc==0])
est.ATE.SPIW <- sum(stabilized.IP.weight*y)

## Doubly-robust
est.y1.dr <- mean((papc*y-(papc-est.cbps)*est.y1.lm)/est.cbps)
est.y0.dr <- mean(((1-papc)*y+(papc-est.cbps)*est.y0.lm)/(1-est.cbps))
est.ATE.dr <- est.y1.dr - est.y0.dr

## results summary
round(c(est.ATE.lm1, est.ATE.IPW, est.ATE.SPIW, est.ATE.dr), 3)
```

```
## [1]  0.096 -1.008  0.057  0.090
```

Bootstrap for Confidence interval

```
## bootstrap estimate of standard error for IPW & SIPW & DR
pseudo_ATE = function(iter, PSKC.data, method = "PS") {

  ## setting
  n = nrow(PSKC.data)
  ipw = sipw = dr = as.numeric(iter)

  ## loop station
  for (b in 1:iter) {
    # seed
    set.seed(b)

    # randomly select the indices
    dt = PSKC.data[sample(1:n, size = n, replace = TRUE),]

    # choose propensity score
    if (method == "PS") {
      dt$ps = glm(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress, family = binomial, da
    } else if (method == "CBPS") {
      dt$ps = CBPS(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress, ATT = 0, data = dt)$
    }

    # data set
    data = dt %>%
      mutate(ipw = ifelse(papc == 1, 1/ps, 1/(1-ps)),
             simReg = predict(lm(y ~ factor(papc) + housing + ca + income + Fschool + Mschool + Kacs +
    data = data %>%
      mutate(sipw = ifelse(papc == 1, ipw/sum(filter(data, papc == 1)$ipw), ipw/sum(filter(data, papc ==

    # Doubly robust setting : confounder X & mu(z)(Xi)
    SimReg = lm(y ~ factor(papc) + housing + ca + income + Fschool + Mschool + Kacs + Fstress + Msff
                + Maff + Minteg + Faff + Mhappiness + Mcrs + Fhappiness + Ksfs + Kssr + Kprefe, data = d

    X = data[c('papc','housing','ca','income','Fschool','Mschool','Kacs','Fstress','Msff','Maff','Minteg

    mu1 = predict(SimReg, mutate(X, papc = as.factor(1)))
    mu0 = predict(SimReg, mutate(X, papc = as.factor(0)))

    # ATE with IPW & SIPW & Doubly Robust
    ipw[b] = (with(filter(data, papc == 1), sum(y*ipw)) - with(filter(data, papc == 0), sum(y*ipw)))/n
    sipw[b] = (with(filter(data, papc == 1), sum(y*sipw)) - with(filter(data, papc == 0), sum(y*sipw)))
    dr[b] = (with(filter(data, papc == 1), sum((y-simReg)*ipw)) - with(filter(data, papc == 0), sum((y-s
  }

  ## CI
  res = data.frame(IPW = c(round(mean(ipw), 3), round(sd(ipw), 3) , round(mean(ipw) - qnorm(0.975)*sd(ip
                   SIPW = c(round(mean(sipw), 3), round(sd(sipw), 3) , round(mean(sipw) - qnorm(0.975)*s
                   DR = c(round(mean(dr), 3), round(sd(dr), 3) , round(mean(dr) - qnorm(0.975)*sd(dr), 3
  rownames(res) = c(paste0("Point Est based on ",method),"Stand Error", "95% Lower", "95% Upper")

  ## result
  return(res)
}

pseudo_ATE(iter = 100, PSKC.data, method = "PS")
```

```
##                          IPW   SIPW     DR
## Point Est based on PS  0.052  0.042  0.097
## Stand Error            0.227  0.168  0.159
## 95% Lower             -0.394 -0.287 -0.215
## 95% Upper              0.497  0.372  0.408
```

```
pseudo_ATE(iter = 100, PSKC.data, method = "CBPS")
```

```
##                            IPW   SIPW     DR
## Point Est based on CBPS -1.142  0.054  0.097
## Stand Error              0.622  0.168  0.158
## 95% Lower               -2.361 -0.275 -0.213
## 95% Upper                0.078  0.382  0.407
```

The outcome model using the inverse probability weights by hand

```
# Outcome Model Using Inverse Probability Weights (IPW) Manually

## Calculate IPW
ipw.papc <- augment_columns(propscore.model, PSKC.data, type.predict = "response") %>%
  rename(propensity = .fitted) %>%
  mutate(
    ipw = (papc / propensity) + ((1 - papc) / (1 - propensity))
  )

## Fit the Outcome Model Using IPW
model.ipw.papc <- lm(y ~ papc, data = ipw.papc, weights = ipw)

## Summarize the Model
tidy(model.ipw.papc) # Coefficients and statistics
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  26.0        0.116   224.      0
## 2 papc          0.0467     0.164     0.285   0.776
```

```
round(confint(model.ipw.papc, level = 0.95), 3) # 95% Confidence Intervals
```

```
##               2.5 % 97.5 %
## (Intercept) 25.728 26.183
## papc        -0.275  0.369
```

```
## Check Balance
# Display the first few rows of the data
head(PSKC.data)
```

```
##    y papc  housing ca income Fschool Mschool Kacs Fstress Msff Mhappiness Mcrs
## 1 25    0 127490.5  0    450       5       5   58      27   45         25   19
## 2 23    0 127490.5  0    200       4       4   42      26   56         20   28
## 3 26    0 127490.5  0    600       5       5   64      25   48         28   14
## 4 26    0 127490.5  0    350       6       5   51      32   54          19   22
## 5 30    0 127490.5  0    600       6       5   60      28   45          25   21
## 6 25    0 127490.5  0    300       5       5   41      23   53          19   21
##   Maff Minteg Fhappiness Faff Ksfs Kssr Kprefe        ps      cbps TrtLevel
## 1   24     16         27   27   53   15     10 0.6135530 0.6186669        0
## 2   25     21         17   18   53   15      9 0.5107536 0.5364189        0
## 3   28     18         28   28   61   11      8 0.6442797 0.6414948        0
## 4   25     12         20   25   54   12     10 0.5384408 0.5461760        0
## 5   18     15         19   19   60   15      9 0.5913320 0.5852986        0
## 6   25     20         28   25   61   12      9 0.5704652 0.5846626        0
```

```
# Compute balance for covariates
covs <- subset(PSKC.data, select = c(housing, ca, income, Fschool, Mschool, Kacs, Fstress))
bal.tab(covs, treat = PSKC.data$papc, weights = ipw.papc$ipw)
```

```
## Balance Measures
##              Type Diff.Adj
## housing   Contin.  -0.4563
## ca         Binary  -0.0002
## income    Contin.   0.0007
## Fschool_4  Binary   0.0023
## Fschool_5  Binary   0.0004
## Fschool_6  Binary  -0.0009
## Fschool_7  Binary  -0.0018
## Mschool_4  Binary   0.0053
## Mschool_5  Binary  -0.0020
## Mschool_6  Binary  -0.0039
## Mschool_7  Binary   0.0006
## Kacs      Contin.   0.0051
## Fstress   Contin.  -0.0055
##
## Effective sample sizes
##            Control Treated
## Unadjusted  512.    441.
## Adjusted    487.47  410.22
```

The outcome model using the inverse probability weights with packages

```
# Inverse Probability Weights (IPW) with Propensity Score and Covariate Balancing Propensity Score (CBP

## Using Propensity Score (PS)

# Calculate IPW weights using propensity score
weights_ps <- ipwpoint(
  exposure = papc,
  family = "binomial",  # Binary treatment
  link = "logit",
  denominator = ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  data = as.data.frame(PSKC.data)
)

# Display first few IPW weights
```

```r
head(weights_ps$ipw.weights)
```

```
## [1] 2.587677 2.043960 2.811197 2.166569 2.446974 2.328100
```

```r
head(ipw.papc$ipw)
```

```
## [1] 2.587677 2.043960 2.811197 2.166569 2.446974 2.328100
```

```r
# Add IPW weights to the data and fit the model
PSKC_data_ps <- PSKC.data %>%
  mutate(ipw = weights_ps$ipw.weights)

model_ps <- lm(y ~ papc, data = PSKC_data_ps, weights = ipw)
tidy(model_ps)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  26.0        0.116    224.         0
## 2 papc          0.0467     0.164     0.285   0.776
```

```r
## Using Covariate Balancing Propensity Score (CBPS)

# Calculate weights using CBPS
weights_cbps <- weightit(
  papc ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  data = PSKC.data,
  estimand = "ATE",   # Estimate the Average Treatment Effect
  method = "cbps"     # Use CBPS method
)

# Display CBPS weights
weights_cbps
```

```
## A weightit object
##  - method: "cbps" (covariate balancing propensity score weighting)
##  - number of obs.: 953
##  - sampling weights: none
##  - treatment: 2-category
##  - estimand: ATE
##  - covariates: ca, income, Fschool, Mschool, Kacs, Fstress
```

```r
head(weights_cbps$weights)
```

```
## [1] 2.582340 2.052734 2.819571 2.148750 2.444209 2.314048
```

```r
# Add CBPS weights to the data and fit the model
PSKC_data_cbps <- PSKC.data %>%
  mutate(ipw = weights_cbps$weights)

model_cbps <- lm(y ~ papc, data = PSKC_data_cbps, weights = ipw)
tidy(model_cbps)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  26.0        0.116    224.         0
## 2 papc          0.0453     0.164     0.277   0.782
```

```r
# Summary statistics
round(c(-0.06067547, 0.1654011), 3)
```

```
## [1] -0.061  0.165
```

```r
round(confint(model_cbps, level = 0.95), 3)
```

```
##              2.5 % 97.5 %
## (Intercept) 25.729 26.184
## papc        -0.276  0.367
```

```r
## Test Covariate Balance
covs <- subset(PSKC.data, select = c(housing, ca, income, Fschool, Mschool, Kacs, Fstress))
bal.tab(covs, treat = PSKC.data$papc, weights = weights_cbps$weights)
```

```
## Balance Measures
##              Type Diff.Adj
## housing   Contin.  -0.4554
## ca         Binary  -0.0000
## income    Contin.   0.0000
## Fschool_4  Binary   0.0000
## Fschool_5  Binary   0.0000
## Fschool_6  Binary   0.0000
## Fschool_7  Binary  -0.0000
## Mschool_4  Binary  -0.0000
## Mschool_5  Binary   0.0000
## Mschool_6  Binary   0.0000
## Mschool_7  Binary  -0.0000
## Kacs      Contin.  -0.0000
## Fstress   Contin.  -0.0000
##
## Effective sample sizes
##            Control Treated
## Unadjusted  512.    441.
## Adjusted    487.22  410.25
```

Prepare data for matching by selecting relevant columns

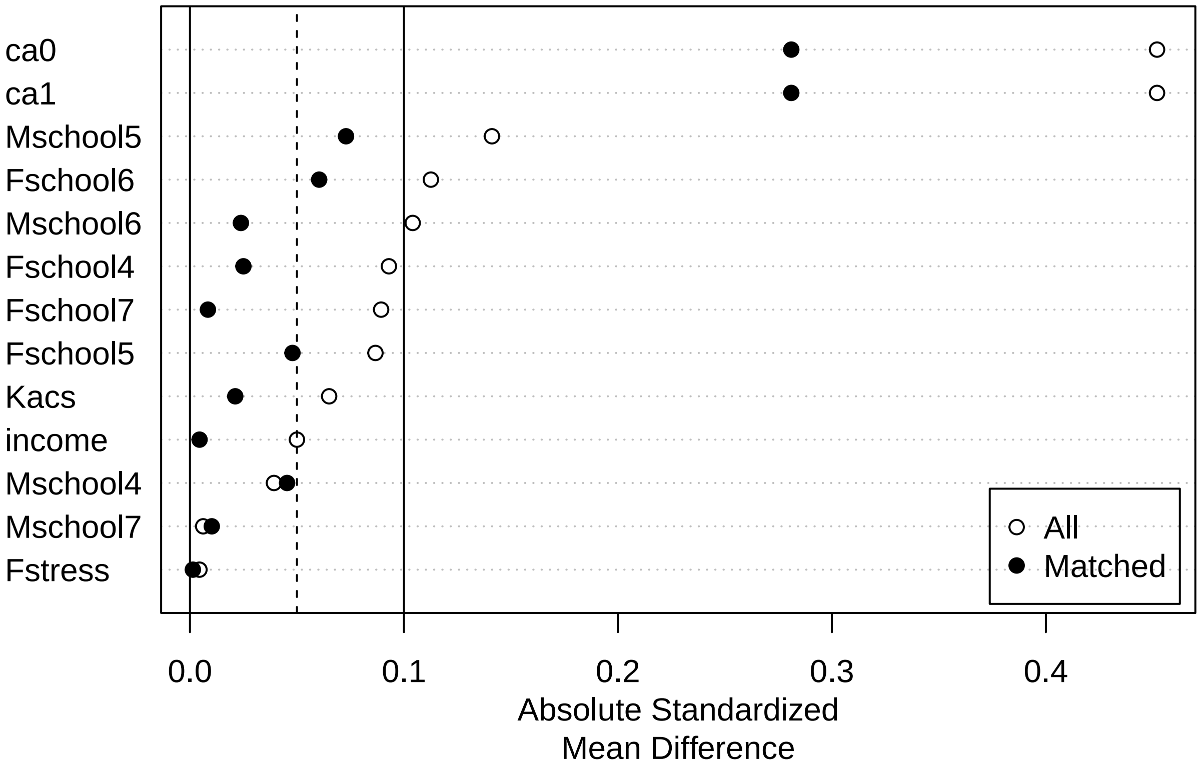
```r
# Exclude specific columns from the dataset
matching.data <- subset(
  PSKC.data,
  select = -c(
    housing, Msff, Mhappiness, Mcrs, Maff, Minteg, Fhappiness, Faff, Ksfs, Kssr, Kprefe, ps, cbps, TrtL
  )
)

# Display the resulting dataset
matching.data
```

Method: PSM

```r
# Propensity Score Matching (PSM)

# Calculate propensity score distance
# propscore.model <- glm(
#   factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
#   family = binomial(link = "logit"), x = TRUE, data = PSKC.data
# )
ps.dist <- match_on(est.ps, z = matching.data$papc)

# Perform matching
psm.out <- matchit(
  factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  data = matching.data,
  method = "optimal",
  estimand = "ATT",
  distance = ps.dist
)

# Display summary of the matching results
summary(psm.out)
```

```
##
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool +
##     Kacs + Fstress, data = matching.data, method = "optimal",
##     distance = ps.dist, estimand = "ATT")
##
## Summary of Balance for All Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6961        0.4883          0.4520          .    0.2079
## ca1             0.3039        0.5117         -0.4520          .    0.2079
## income        447.1406      437.5586          0.0500     1.2694    0.0138
## Fschool4        0.2902        0.2480          0.0930          .    0.0422
## Fschool5        0.2381        0.2012          0.0867          .    0.0369
## Fschool6        0.3923        0.4473         -0.1126          .    0.0550
## Fschool7        0.0794        0.1035         -0.0893          .    0.0242
## Mschool4        0.2812        0.2988         -0.0393          .    0.0176
## Mschool5        0.3197        0.2539          0.1411          .    0.0658
## Mschool6        0.3469        0.3965         -0.1041          .    0.0495
## Mschool7        0.0522        0.0508          0.0062          .    0.0014
## Kacs           53.9705       53.2461          0.0650     0.8644    0.0173
## Fstress        26.0113       26.0410         -0.0044     1.0760    0.0124
##          eCDF Max
## ca0        0.2079
## ca1        0.2079
## income     0.0280
## Fschool4   0.0422
## Fschool5   0.0369
## Fschool6   0.0550
## Fschool7   0.0242
## Mschool4   0.0176
## Mschool5   0.0658
## Mschool6   0.0495
## Mschool7   0.0014
## Kacs       0.0405
## Fstress    0.0472
##
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6961        0.5669          0.2810          .    0.1293
## ca1             0.3039        0.4331         -0.2810          .    0.1293
## income        447.1406      446.2812          0.0045     1.1774    0.0085
## Fschool4        0.2902        0.2789          0.0250          .    0.0113
## Fschool5        0.2381        0.2177          0.0479          .    0.0204
## Fschool6        0.3923        0.4218         -0.0604          .    0.0295
## Fschool7        0.0794        0.0816         -0.0084          .    0.0023
## Mschool4        0.2812        0.3016         -0.0454          .    0.0204
## Mschool5        0.3197        0.2857          0.0729          .    0.0340
## Mschool6        0.3469        0.3583         -0.0238          .    0.0113
## Mschool7        0.0522        0.0544         -0.0102          .    0.0023
## Kacs           53.9705       53.7347          0.0212     0.9421    0.0135
## Fstress        26.0113       26.0204         -0.0014     1.1064    0.0133
##          eCDF Max Std. Pair Dist.
## ca0        0.1293          0.3008
## ca1        0.1293          0.3008
## income     0.0295          0.9927
## Fschool4   0.0113          0.8243
## Fschool5   0.0204          0.7933
## Fschool6   0.0295          0.8127
## Fschool7   0.0023          0.5453
## Mschool4   0.0204          0.9432
## Mschool5   0.0340          0.7050
## Mschool6   0.0113          0.8241
## Mschool7   0.0023          0.4385
## Kacs       0.0431          1.0119
## Fstress    0.0522          1.1135
##
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       441     441
## Unmatched      71       0
## Discarded       0       0
```

```
# Plot balance diagnostics
plot(
  summary(psm.out),
  var.order = "unmatched"
)
```



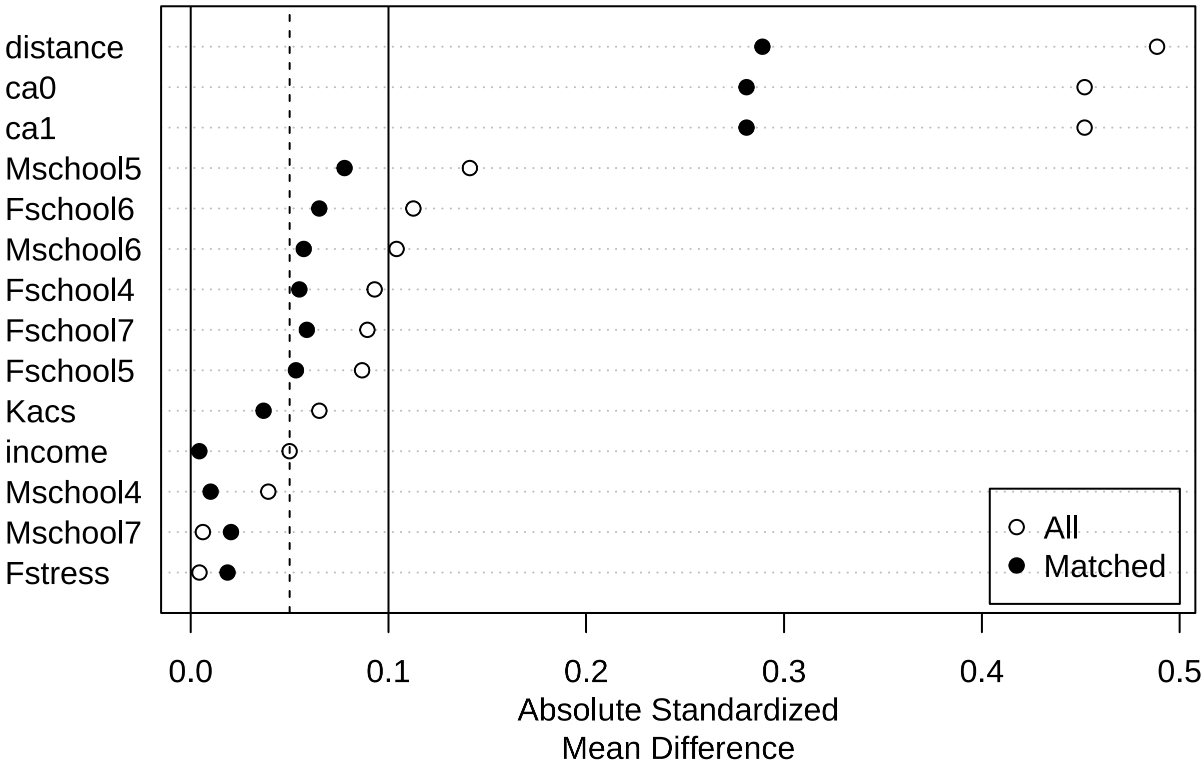
Method: CBPSM

```r
# Covariate Balancing Propensity Score Matching (CBPSM)

# Fit CBPS model
cbps.model <- CBPS(
  factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  ATT = 0,
  data = matching.data
)

# Use CBPS model to estimate propensity scores
# est.cbps <- fitted(cbps.model)
# est.cbps <- predict(cbps.model, type = "response")

# Perform CBPS matching
cbpsm.out <- matchit(
  factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  data = matching.data,
  method = "optimal",
  distance = "cbps",
  estimand = "ATT"
)

# Display summary of the CBPS matching results
summary(cbpsm.out, un = FALSE)
```

```
##
```

```
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool + 
##     Kacs + Fstress, data = matching.data, method = "optimal", 
##     distance = "cbps", estimand = "ATT")
## 
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## distance        0.4928        0.4598          0.2891     1.0897    0.0779
## ca0             0.6961        0.5669          0.2810          .    0.1293
## ca1             0.3039        0.4331         -0.2810          .    0.1293
## income        447.1406      447.9819         -0.0044     1.1858    0.0099
## Fschool4        0.2902        0.2653          0.0550          .    0.0249
## Fschool5        0.2381        0.2154          0.0532          .    0.0227
## Fschool6        0.3923        0.4240         -0.0650          .    0.0317
## Fschool7        0.0794        0.0952         -0.0587          .    0.0159
## Mschool4        0.2812        0.2857         -0.0101          .    0.0045
## Mschool5        0.3197        0.2834          0.0778          .    0.0363
## Mschool6        0.3469        0.3741         -0.0572          .    0.0272
## Mschool7        0.0522        0.0567         -0.0204          .    0.0045
## Kacs           53.9705       53.5601          0.0369     0.8719    0.0176
## Fstress        26.0113       25.8866          0.0186     1.1053    0.0124
##          eCDF Max Std. Pair Dist.
## distance   0.1451          0.2900
## ca0        0.1293          0.3008
## ca1        0.1293          0.3008
## income     0.0295          0.9447
## Fschool4   0.0249          0.8143
## Fschool5   0.0227          0.7667
## Fschool6   0.0317          0.9195
## Fschool7   0.0159          0.5956
## Mschool4   0.0045          0.7868
## Mschool5   0.0363          0.6807
## Mschool6   0.0272          0.9051
## Mschool7   0.0045          0.4895
## Kacs       0.0522          1.1176
## Fstress    0.0567          1.0691
## 
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       441     441
## Unmatched      71       0
## Discarded       0       0
```

```r
# Plot balance diagnostics for CBPS matching
plot(
  summary(cbpsm.out),
  var.order = "unmatched"
)
```



Method: Propensity Score Caliper Matching

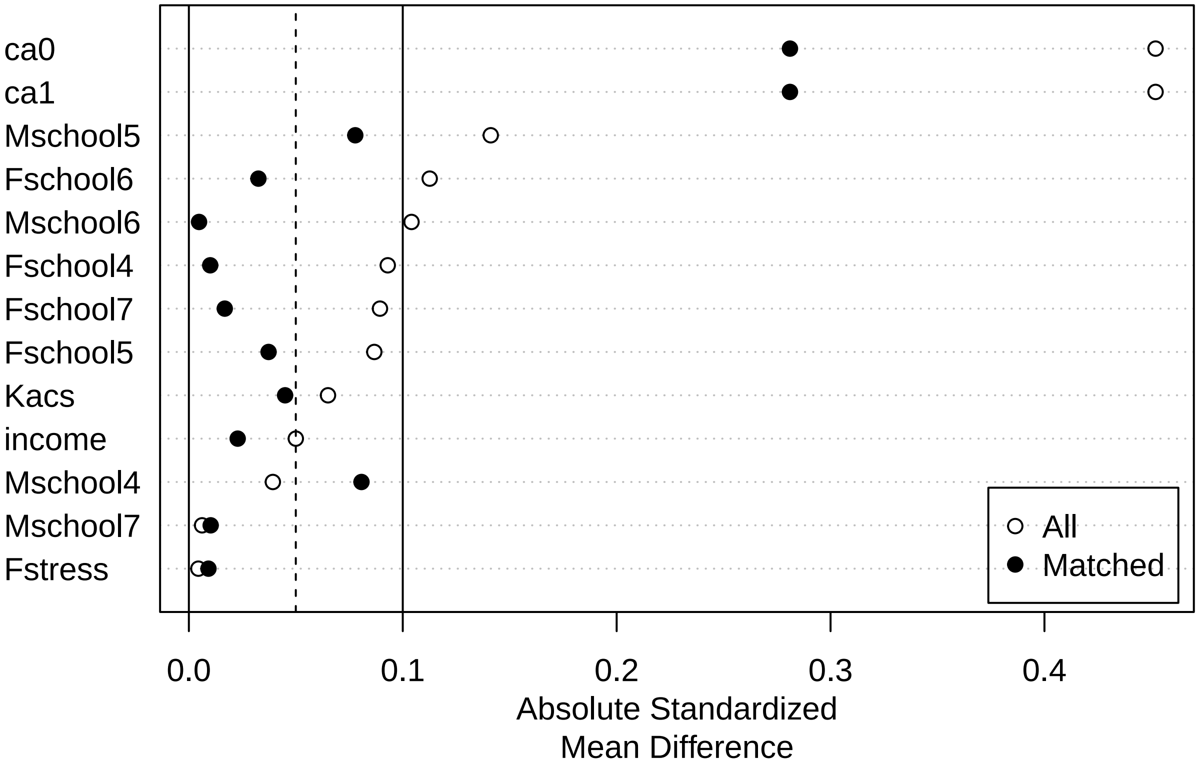
Using Rank-Based Mahalanobis Distance Within Propensity Score Calipers

```r
# Mahalanobis Distance Matching

# Compute the rank-based Mahalanobis distance
smahal.dist <- optmatch::match_on(
  papc ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  method = "rank_mahalanobis"
)

# Uncomment to perform matching and summarize results without caliper
# mc.out <- matchit(
#   factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
#   data = PSKC.data,
#   method = "optimal",
#   distance = smahal.dist,
#   replace = FALSE
# )
# summary(mc.out, un = FALSE)
# plot(summary(mc.out), var.order = "unmatched")

# Apply a caliper width of 0.1 to Mahalanobis distance
smahal.dist3 <- smahal.dist + caliper(ps.dist, width = 0.1)
```

```r
# Perform matching with the caliper-adjusted Mahalanobis distance
mc.out3 <- matchit(
  factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  data = matching.data,
  method = "optimal",
  distance = as.matrix(smahal.dist3)
)
summary(mc.out3, un = FALSE)
```

```
##
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool +
##     Kacs + Fstress, data = matching.data, method = "optimal",
##     distance = as.matrix(smahal.dist3))
##
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6961        0.5669          0.2810          .    0.1293
## ca1             0.3039        0.4331         -0.2810          .    0.1293
## income        447.1406      442.7664          0.0228     1.2140    0.0103
## Fschool4        0.2902        0.2857          0.0100          .    0.0045
## Fschool5        0.2381        0.2222          0.0373          .    0.0159
## Fschool6        0.3923        0.4082         -0.0325          .    0.0159
## Fschool7        0.0794        0.0839         -0.0168          .    0.0045
## Mschool4        0.2812        0.3175         -0.0807          .    0.0363
## Mschool5        0.3197        0.2834          0.0778          .    0.0363
## Mschool6        0.3469        0.3447          0.0048          .    0.0023
## Mschool7        0.0522        0.0544         -0.0102          .    0.0023
## Kacs           53.9705       53.4694          0.0450     0.9212    0.0145
## Fstress        26.0113       26.0726         -0.0091     1.1379    0.0153
##          eCDF Max Std. Pair Dist.
## ca0        0.1293          0.2810
## ca1        0.1293          0.2810
## income     0.0317          0.5358
## Fschool4   0.0045          0.1099
## Fschool5   0.0159          0.1437
## Fschool6   0.0159          0.1811
## Fschool7   0.0045          0.0839
## Mschool4   0.0363          0.2925
## Mschool5   0.0363          0.3792
## Mschool6   0.0023          0.2239
## Mschool7   0.0023          0.1530
## Kacs       0.0363          0.4982
## Fstress    0.0522          0.4646
##
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       441     441
## Unmatched      71       0
## Discarded       0       0
```

```r
plot(summary(mc.out3), var.order = "unmatched")
```



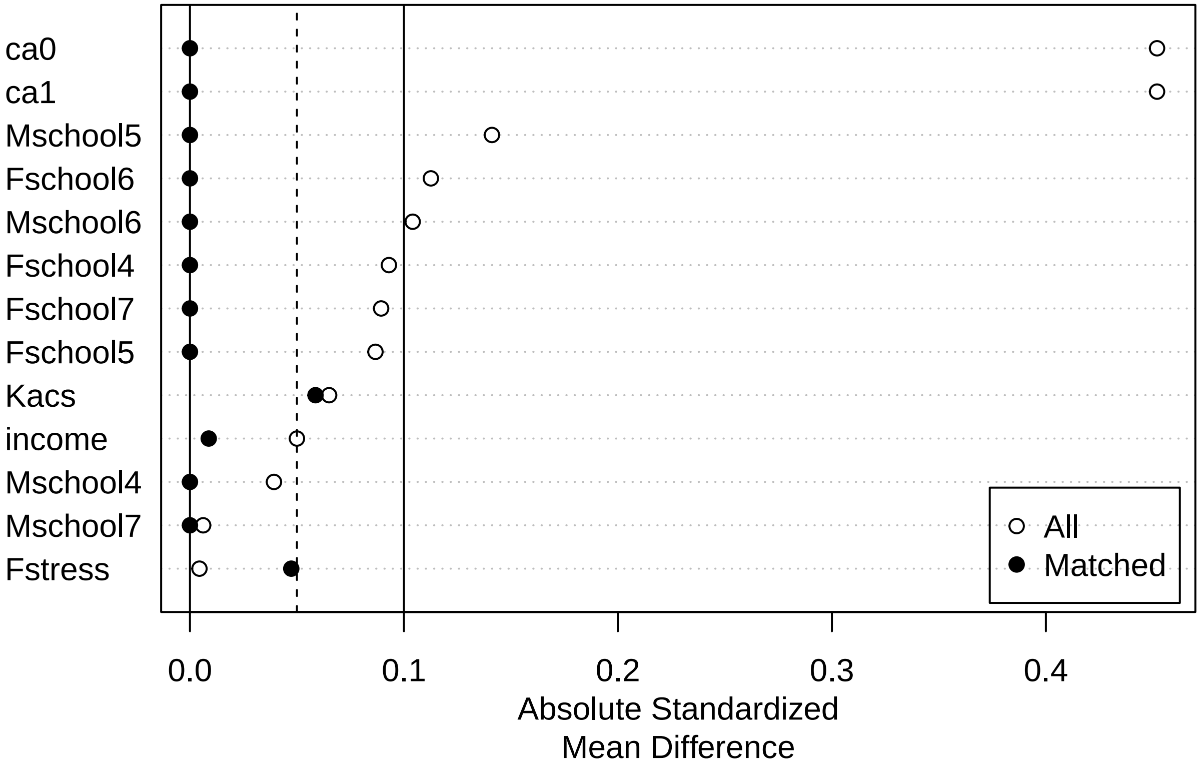
```r
# Uncomment to use Mahalanobis distance with near-exact matching for housing
# smahal.dist.housing <- addalmostexact(
#   as.matrix(smahal.dist),
#   PSKC.data$papc,
#   PSKC.data$housing,
#   mult = 10
# )
# mc.housing.out <- matchit(
#   factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
#   data = PSKC.data,
#   method = "optimal",
#   distance = as.matrix(smahal.dist.housing)
# )
# summary(mc.housing.out, un = FALSE)
# plot(summary(mc.housing.out), var.order = "unmatched")
```

Method: CEM

```r
# Coarsened Exact Matching (CEM) on Covariates, Excluding Capital Area Indicator
# Grouping Parent's Education: University and Graduate School Combined

cem.out = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                  data = matching.data, method = "cem", estimand = "ATT",
                  cutpoints = list(income = "q10", Kacs = "q4", Fstress = "q4"),
                  grouping = list(Mschool = list("4", "5", "6", "7"),
                                  Fschool = list("4", "5", "6", "7")),
                  k2k = TRUE)

summary(cem.out)
```

```
##
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool +
##     Kacs + Fstress, data = matching.data, method = "cem", estimand = "ATT",
##     cutpoints = list(income = "q10", Kacs = "q4", Fstress = "q4"),
##     grouping = list(Mschool = list("4", "5", "6", "7"), Fschool = list("4",
##         "5", "6", "7")), k2k = TRUE)
##
## Summary of Balance for All Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6961        0.4883          0.4520          .    0.2079
## ca1             0.3039        0.5117         -0.4520          .    0.2079
## income        447.1406      437.5586          0.0500     1.2694    0.0138
## Fschool4        0.2902        0.2480          0.0930          .    0.0422
## Fschool5        0.2381        0.2012          0.0867          .    0.0369
## Fschool6        0.3923        0.4473         -0.1126          .    0.0550
## Fschool7        0.0794        0.1035         -0.0893          .    0.0242
## Mschool4        0.2812        0.2988         -0.0393          .    0.0176
## Mschool5        0.3197        0.2539          0.1411          .    0.0658
## Mschool6        0.3469        0.3965         -0.1041          .    0.0495
## Mschool7        0.0522        0.0508          0.0062          .    0.0014
## Kacs           53.9705       53.2461          0.0650     0.8644    0.0173
## Fstress        26.0113       26.0410         -0.0044     1.0760    0.0124
##          eCDF Max
## ca0        0.2079
## ca1        0.2079
## income     0.0280
## Fschool4   0.0422
## Fschool5   0.0369
## Fschool6   0.0550
## Fschool7   0.0242
## Mschool4   0.0176
## Mschool5   0.0658
## Mschool6   0.0495
## Mschool7   0.0014
## Kacs       0.0405
## Fstress    0.0472
##
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6436        0.6436          0.0000          .    0.0000
## ca1             0.3564        0.3564          0.0000          .    0.0000
## income        449.3069      447.6238          0.0088     1.0599    0.0057
## Fschool4        0.2475        0.2475          0.0000          .    0.0000
## Fschool5        0.1980        0.1980          0.0000          .    0.0000
## Fschool6        0.5347        0.5347          0.0000          .    0.0000
## Fschool7        0.0198        0.0198          0.0000          .    0.0000
## Mschool4        0.2970        0.2970          0.0000          .    0.0000
## Mschool5        0.1980        0.1980          0.0000          .    0.0000
## Mschool6        0.4851        0.4851          0.0000          .    0.0000
## Mschool7        0.0198        0.0198          0.0000          .    0.0000
## Kacs           52.7228       53.3762         -0.0587     1.3505    0.0245
## Fstress        27.2079       26.8911          0.0473     0.9773    0.0140
##          eCDF Max Std. Pair Dist.
## ca0        0.0000          0.0000
## ca1        0.0000          0.0000
## income     0.0198          0.1080
## Fschool4   0.0000          0.0000
## Fschool5   0.0000          0.0000
## Fschool6   0.0000          0.0000
## Fschool7   0.0000          0.0000
## Mschool4   0.0000          0.0000
## Mschool5   0.0000          0.0000
## Mschool6   0.0000          0.0000
## Mschool7   0.0000          0.0000
## Kacs       0.0792          0.3769
## Fstress    0.0495          0.3107
##
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       101     101
## Unmatched     411     340
## Discarded       0       0
```

```
plot(summary(cem.out), var.order = "unmatched")
```



Method: cardinality matching

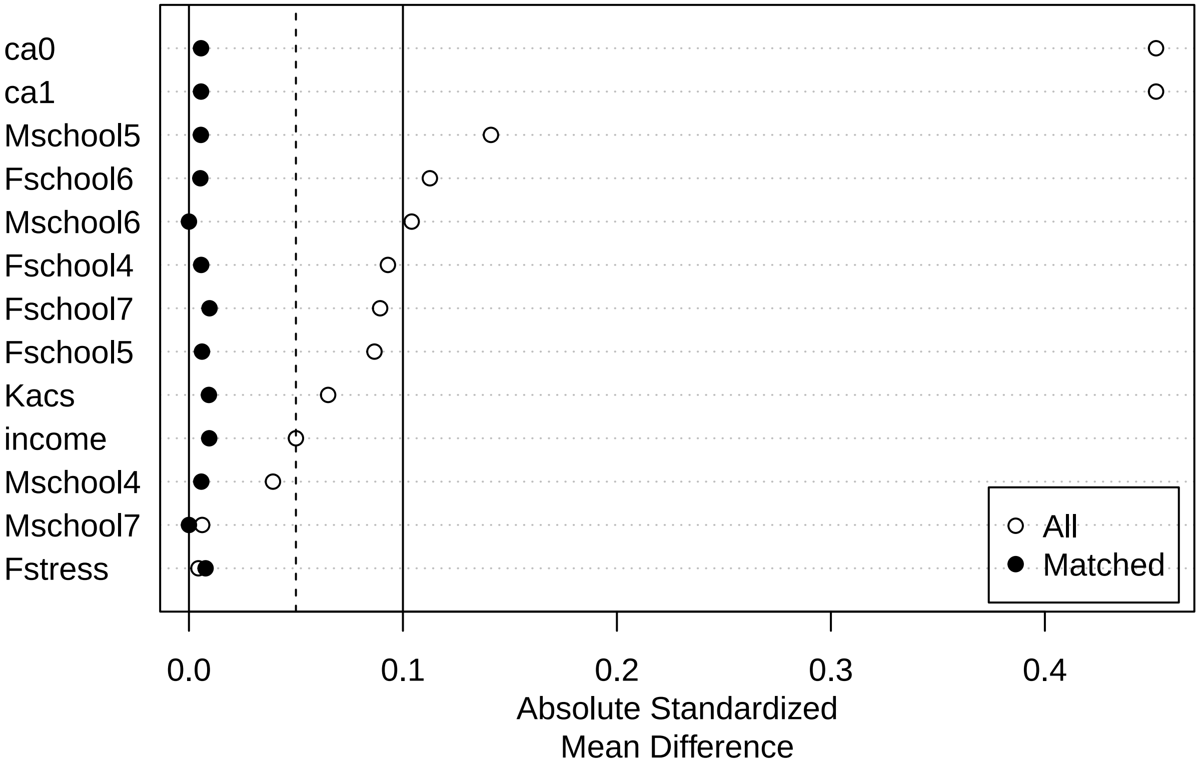
```
# Load necessary library
# install.packages("Rglpk")
library(Rglpk)

# Step 1: Find control group with SMD  0.01
m.card.out = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                        data = matching.data, method = "cardinality", tols = 0.01, solver = "glpk")

# Improved speed with exact matching on `ca`
# m.card.out = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
#                        data = PSKC.data, method = "cardinality", tols = 0.01, solver = "glpk", exact =

summary(m.card.out, un = FALSE)
```

```
##
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool +
##     Kacs + Fstress, data = matching.data, method = "cardinality",
##     tols = 0.01, solver = "glpk")
##
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6519        0.6494          0.0056          .    0.0026
## ca1             0.3481        0.3506         -0.0056          .    0.0026
## income        443.8156      442.0000          0.0095     0.9961    0.0106
## Fschool4        0.2675        0.2649          0.0057          .    0.0026
## Fschool5        0.2338        0.2312          0.0061          .    0.0026
## Fschool6        0.4156        0.4182         -0.0053          .    0.0026
## Fschool7        0.0831        0.0857         -0.0096          .    0.0026
## Mschool4        0.2961        0.2987         -0.0058          .    0.0026
## Mschool5        0.2961        0.2935          0.0056          .    0.0026
## Mschool6        0.3506        0.3506          0.0000          .    0.0000
## Mschool7        0.0571        0.0571          0.0000          .    0.0000
## Kacs           53.7688       53.6649          0.0093     0.9052    0.0148
## Fstress        26.2078       26.2597         -0.0078     1.0917    0.0123
##          eCDF Max
## ca0        0.0026
## ca1        0.0026
## income     0.0234
## Fschool4   0.0026
## Fschool5   0.0026
## Fschool6   0.0026
## Fschool7   0.0026
## Mschool4   0.0026
## Mschool5   0.0026
## Mschool6   0.0000
## Mschool7   0.0000
## Kacs       0.0597
## Fstress    0.0468
##
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       385     385
## Unmatched     127      56
## Discarded       0       0
```

```
plot(summary(m.card.out), var.order = "unmatched")

# Step 2: Re-match to improve balance within pairs
# Match similar `x` values within control group with SMD   0.01
m.card.re = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                    data = matching.data, method = "optimal", distance = "mahalanobis",
                    discard = m.card.out$weights == 0)

summary(m.card.re, un = FALSE)
```

```
##
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool +
##     Kacs + Fstress, data = matching.data, method = "optimal",
##     distance = "mahalanobis", discard = m.card.out$weights ==
##         0)
##
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6519        0.6494          0.0056          .    0.0026
## ca1             0.3481        0.3506         -0.0056          .    0.0026
## income        443.8156      442.0000          0.0095     0.9961    0.0106
## Fschool4        0.2675        0.2649          0.0057          .    0.0026
## Fschool5        0.2338        0.2312          0.0061          .    0.0026
## Fschool6        0.4156        0.4182         -0.0053          .    0.0026
## Fschool7        0.0831        0.0857         -0.0096          .    0.0026
## Mschool4        0.2961        0.2987         -0.0058          .    0.0026
## Mschool5        0.2961        0.2935          0.0056          .    0.0026
## Mschool6        0.3506        0.3506          0.0000          .    0.0000
## Mschool7        0.0571        0.0571          0.0000          .    0.0000
## Kacs           53.7688       53.6649          0.0093     0.9052    0.0148
## Fstress        26.2078       26.2597         -0.0078     1.0917    0.0123
##          eCDF Max Std. Pair Dist.
## ca0        0.0026          0.1977
## ca1        0.0026          0.1977
## income     0.0234          0.3803
## Fschool4   0.0026          0.0515
## Fschool5   0.0026          0.0183
## Fschool6   0.0026          0.0585
## Fschool7   0.0026          0.0673
## Mschool4   0.0026          0.0636
## Mschool5   0.0026          0.1504
## Mschool6   0.0000          0.0675
## Mschool7   0.0000          0.0052
## Kacs       0.0597          0.4529
## Fstress    0.0468          0.4386
##
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       385     385
## Unmatched       0       0
## Discarded     127      56
```

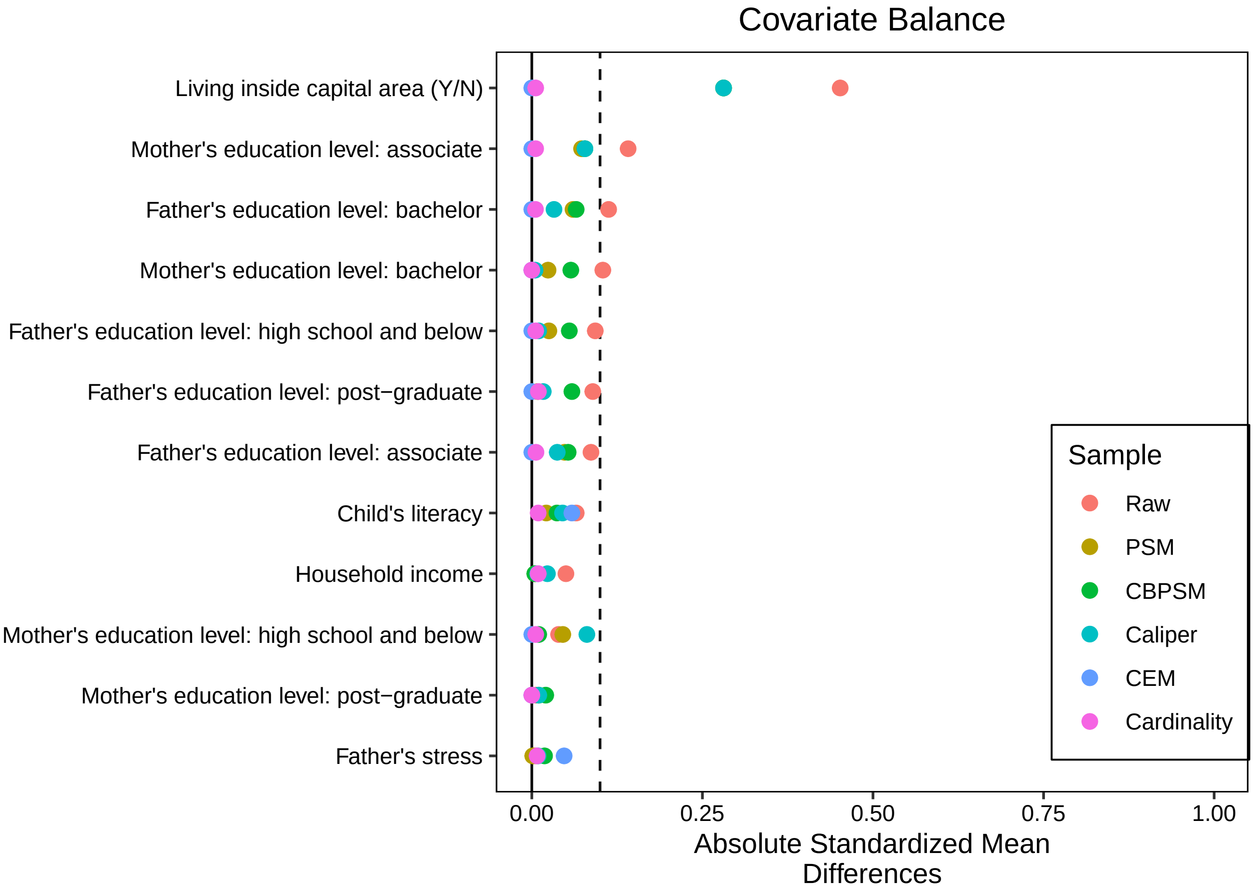
```
plot(summary(m.card.re), var.order = "unmatched")
```



```
# Extract matched data
m.card = match.data(m.card.re)
```

plots

```
new.names <- c(ca = "Living inside capital area (Y/N)",
               income = "Household income",
               Fschool_4 = "Father’s education level: high school and below",
               Fschool_5 = "Father’s education level: associate",
               Fschool_6 = "Father’s education level: bachelor",
               Fschool_7 = "Father’s education level: post-graduate",
               Mschool_4 = "Mother’s education level: high school and below",
               Mschool_5 = "Mother’s education level: associate",
               Mschool_6 = "Mother’s education level: bachelor",
               Mschool_7 = "Mother’s education level: post-graduate",
               Kacs = "Child's literacy",
               Fstress = "Father's stress"
)

love.plot(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
          data = matching.data, estimand = "ATT",
          stats = "mean.diffs",
          weights = list(w1 = get.w(psm.out),
                         w2 = get.w(cbpsm.out),
                         w3 = get.w(mc.out3),
                         w4 = get.w(cem.out),
                         w5 = get.w(m.card.re)),
          var.order = "unadjusted",
          stars = "raw",
          binary = "std",
          abs = TRUE,
          line = FALSE,
          thresholds = c(m = .1),
          var.names = new.names,
#           colors = c("darkgrey", "red", "blue", "darkgreen", "Yellow", "purple"),
          sample.names = c("Raw", "PSM", "CBPSM", "Caliper", "CEM", "Cardinality"),
          position = "bottomright",
          limits = c(0, 1.05)) +
  theme(legend.position = c(.87, .27),
        legend.box.background = element_rect(),
        legend.box.margin = margin(1, 1, 1, 1))
```



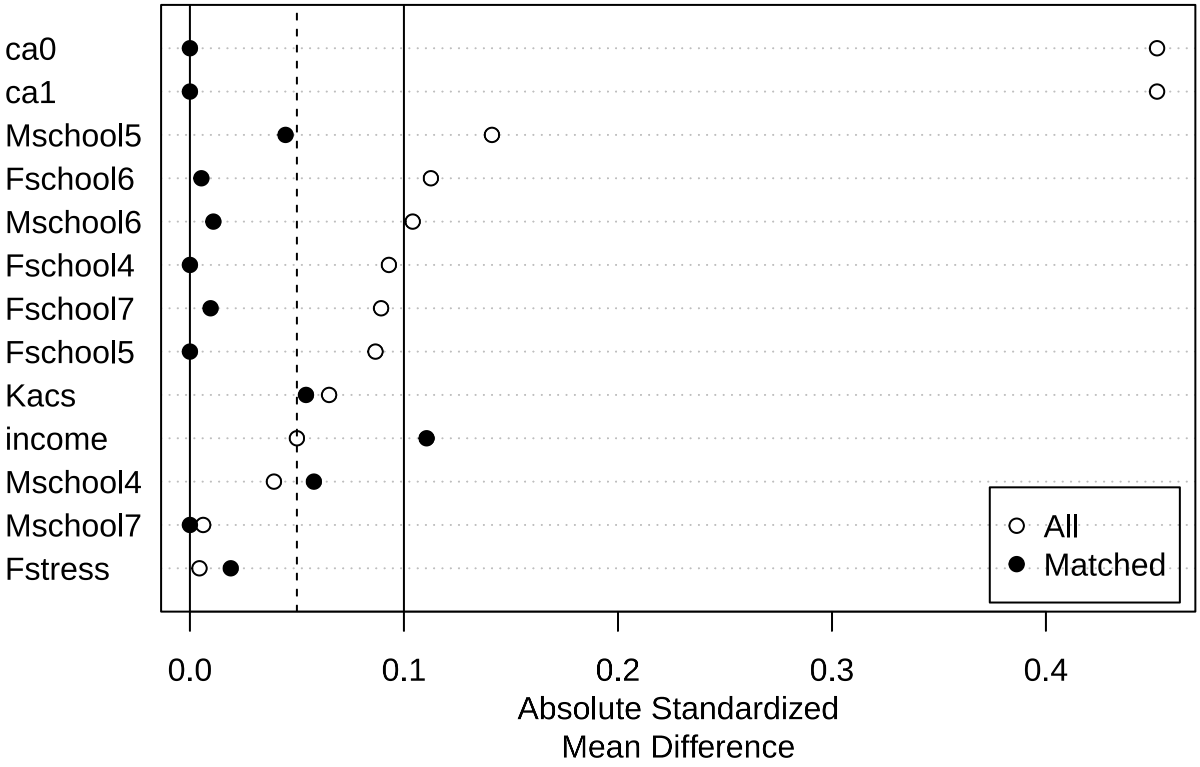
Separate optimal pair matching for ca and non-ca

```
# Matching with separate treatments: Capital Area vs. Non-Capital Area
# Since the outcome was not used in the matching process, it's fine to repeat the matching process mult
# Split the matching problem into two cases: Capital Area (ca) and Non-Capital Area (non-ca)
```

```r
# Perform exact matching with the Capital Area indicator
m.exact.out <- matchit(
  formula = factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
  data = PSKC.data,
  method = "optimal",
  distance = "robust_mahalanobis",
  exact = ~ ca
)

# Summary of the matching results
summary(m.exact.out, un = FALSE)
```

```
##
## Call:
## matchit(formula = factor(papc) ~ ca + income + Fschool + Mschool +
##     Kacs + Fstress, data = PSKC.data, method = "optimal", distance = "robust_mahalanobis",
##     exact = ~ca)
##
## Summary of Balance for Matched Data:
##          Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## ca0             0.6510        0.6510          0.0000          .    0.0000
## ca1             0.3490        0.3490          0.0000          .    0.0000
## income        446.3776      425.1823          0.1106     1.4445    0.0233
## Fschool4        0.3073        0.3073          0.0000          .    0.0000
## Fschool5        0.2188        0.2188          0.0000          .    0.0000
## Fschool6        0.3984        0.3958          0.0053          .    0.0026
## Fschool7        0.0755        0.0781         -0.0096          .    0.0026
## Mschool4        0.3229        0.3490         -0.0579          .    0.0260
## Mschool5        0.2786        0.2578          0.0447          .    0.0208
## Mschool6        0.3490        0.3438          0.0109          .    0.0052
## Mschool7        0.0495        0.0495          0.0000          .    0.0000
## Kacs           54.0365       53.4323          0.0542     0.8655    0.0151
## Fstress        26.2865       26.4141         -0.0191     1.1024    0.0156
##          eCDF Max Std. Pair Dist.
## ca0        0.0000          0.0000
## ca1        0.0000          0.0000
## income     0.0495          0.3693
## Fschool4   0.0000          0.0052
## Fschool5   0.0000          0.0052
## Fschool6   0.0026          0.0267
## Fschool7   0.0026          0.0096
## Mschool4   0.0260          0.0579
## Mschool5   0.0208          0.0670
## Mschool6   0.0052          0.0219
## Mschool7   0.0000          0.0052
## Kacs       0.0417          0.4621
## Fstress    0.0547          0.4090
##
## Sample Sizes:
##           Control Treated
## All           512     441
## Matched       384     384
## Unmatched     128      57
## Discarded       0       0
```

```
# Plot the results of the matching
plot(
  summary(m.exact.out),
  var.order = "unmatched"
)
```



L1 Distance Calculation

Load necessary packages

```
# Check if Tcl/Tk capabilities are available
capabilities("tcltk")
```

```
## tcltk
##  TRUE
```

```
# List directories and check for Tcl/Tk libraries
system("ls -ld /usr/local /usr/local/lib /usr/local/lib/libtcl*")

# Install required packages (uncomment if needed)
# install.packages(c("lattice", "cem"))

# Load libraries
library(lattice)  # For lattice-based plotting
library(cem)      # For Coarsened Exact Matching (CEM)
```

Imbalance Check

```r
# Raw Data Imbalance
# Calculate imbalance metrics for the raw data before matching
raw = imbalance(matching.data$papc, matching.data, drop = c("y", "papc"))
raw
```

```
##
## Multivariate Imbalance Measure: L1=0.950
## Percentage of local common support: LCS=2.8%
##
## Univariate Imbalance Measures:
##
##            statistic   type           L1 min 25% 50% 75%  max
## ca      41.29958049 (Chi2) 2.078639e-01  NA  NA  NA  NA   NA
## income  -9.58199582 (diff) 3.774758e-15   0   0   0   0 -300
## Fschool  6.25105015 (Chi2) 7.912592e-02  NA  NA  NA  NA   NA
## Mschool  5.42926422 (Chi2) 6.719459e-02  NA  NA  NA  NA   NA
## Kacs    -0.72442779 (diff) 2.948732e-02  -4  -1   0   0    0
## Fstress  0.02967776 (diff) 3.570100e-02   0   0   0  -1    2
```

```r
# Propensity Score Matching (PSM)
if (require(MatchIt)) {
  # Create distance matrix for PSM
  ps.dist = match_on(est.ps, z = matching.data$papc)

  # Perform optimal matching using propensity score distance
  psm.out = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                    data = matching.data, method = "optimal", distance = ps.dist)

  # Calculate imbalance metrics for PSM
  psm = imbalance(matching.data$papc, matching.data, drop = c("y", "papc"), weights = psm.out$weights)
  psm
}
```

```
##
## Multivariate Imbalance Measure: L1=0.946
## Percentage of local common support: LCS=2.9%
##
## Univariate Imbalance Measures:
##
##             statistic   type         L1 min 25% 50% 75%  max
## ca      15.279392349 (Chi2) 0.12925170  NA  NA  NA  NA   NA
## income  -0.859410431 (diff) 0.00000000   0   0   0   0 -300
## Fschool  0.987423264 (Chi2) 0.03174603  NA  NA  NA  NA   NA
## Mschool  1.259534174 (Chi2) 0.03401361  NA  NA  NA  NA   NA
## Kacs    -0.235827664 (diff) 0.03854875  -1   0   1   0    0
## Fstress  0.009070295 (diff) 0.03628118   0   0   0  -1    2
```

```r
# CBPS Matching
if (require(MatchIt)) {
  # Perform optimal matching using CBPS distance
  cbpsm.out = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                      data = matching.data, method = "optimal", distance = "cbps", estimand = "ATT")
  # Calculate imbalance metrics for CBPSM
  cbpsm = imbalance(matching.data$papc, matching.data, drop = c("y", "papc"), weights = cbpsm.out$weight
  cbpsm
}
```

```
##
## Multivariate Imbalance Measure: L1=0.943
## Percentage of local common support: LCS=3.0%
##
## Univariate Imbalance Measures:
##
##          statistic   type         L1 min 25% 50% 75%  max
## ca      15.2793923 (Chi2) 0.12925170  NA  NA  NA  NA   NA
## income   0.8412698 (diff) 0.00000000   0   0   0   0 -300
## Fschool  2.1746856 (Chi2) 0.04761905  NA  NA  NA  NA   NA
## Mschool  1.5145695 (Chi2) 0.03628118  NA  NA  NA  NA   NA
## Kacs    -0.4104308 (diff) 0.04308390  -1  -1   1   0    0
## Fstress -0.1247166 (diff) 0.03401361   0   0   0  -1    2
```

```
# PS with Caliper
if (require(MatchIt)) {
  # Compute rank-based Mahalanobis distance
  smahal.dist <- optmatch::match_on(
    papc ~ ca + income + Fschool + Mschool + Kacs + Fstress,
    method = "rank_mahalanobis"
  )

  # Add caliper to Mahalanobis distance
  smahal.dist3 = smahal.dist + caliper(ps.dist, width = 0.1)

  # Perform optimal matching using Mahalanobis distance with caliper
  mc.out3 = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                    data = matching.data, method = "optimal", distance = as.matrix(smahal.dist3))

  # Calculate imbalance metrics for Mahalanobis distance with caliper
  mc3 = imbalance(matching.data$papc, matching.data, drop = c("y", "papc"), weights = mc.out3$weights)
  mc3
}
```

```
##
## Multivariate Imbalance Measure: L1=0.943
## Percentage of local common support: LCS=3.0%
##
## Univariate Imbalance Measures:
##
##           statistic   type         L1 min 25% 50% 75%  max
## ca      15.27939235 (Chi2) 0.12925170  NA  NA  NA  NA   NA
## income  -4.37414966 (diff) 0.00000000   0   0   0   0 -300
## Fschool  0.45149310 (Chi2) 0.02040816  NA  NA  NA  NA   NA
## Mschool  1.95665827 (Chi2) 0.03854875  NA  NA  NA  NA   NA
## Kacs    -0.50113379 (diff) 0.02947846  -1  -1   0   0    0
## Fstress  0.06122449 (diff) 0.03628118   0   0   0  -1    2
```

```
# Coarsened Exact Matching (CEM)
if (require(MatchIt)) {
  # Perform CEM with specified cutpoints and grouping
  cem.out = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                    data = matching.data, method = "cem", estimand = "ATT",
                    cutpoints = list(income = "q10", Kacs = "q4", Fstress = "q4"),
                    grouping = list(Mschool = list("4", "5", "6", "7"), Fschool = list("4", "5", "6", "
                    k2k = TRUE)

  # Calculate imbalance metrics for CEM
  cem = imbalance(matching.data$papc, matching.data, drop = c("y", "papc"), weights = cem.out$weights)
  cem
}
```

```
##
## Multivariate Imbalance Measure: L1=0.822
## Percentage of local common support: LCS=9.9%
##
## Univariate Imbalance Measures:
##
##           statistic   type         L1 min 25% 50% 75% max
## ca       0.0000000 (Chi2) 0.00000000  NA  NA  NA  NA  NA
## income  -1.6831683 (diff) 0.00000000  30   0   0   0   0
## Fschool  0.0000000 (Chi2) 0.00000000  NA  NA  NA  NA  NA
## Mschool  0.0000000 (Chi2) 0.00000000  NA  NA  NA  NA  NA
## Kacs     0.6534653 (diff) 0.00000000   3   0   0   0   0
## Fstress -0.3168317 (diff) 0.04950495   1   0   0   0  -2
```

```
# Cardinality Matching
if (require(MatchIt)) {
  # Perform cardinality matching with specified distance and discard criteria
  m.card.re = matchit(factor(papc) ~ ca + income + Fschool + Mschool + Kacs + Fstress,
                      data = matching.data, method = "optimal", distance = "mahalanobis",
                      discard = m.card.out$weights == 0)

  # Calculate imbalance metrics for cardinality matching
  m.card = imbalance(matching.data$papc, matching.data, drop = c("y", "papc"), weights = m.card.re$weig
  m.card
}
```

```
##
## Multivariate Imbalance Measure: L1=0.961
## Percentage of local common support: LCS=2.0%
##
## Univariate Imbalance Measures:
##
##             statistic   type          L1 min 25% 50% 75% max
## ca       0.000000000 (Chi2) 0.002597403  NA  NA  NA  NA  NA
## income  -1.815584416 (diff) 0.000000000   0   0   0   0 200
## Fschool  0.028964521 (Chi2) 0.005194805  NA  NA  NA  NA  NA
## Mschool  0.008772099 (Chi2) 0.002597403  NA  NA  NA  NA  NA
## Kacs    -0.103896104 (diff) 0.044155844  -4   0   1   0   0
## Fstress  0.051948052 (diff) 0.028571429   0   1   0   0   2
```