Project 6: Randomization and Matching

Introduction

In this project, you will explore the question of whether college education causally affects political participation. Specifically, you will use replication data from Who Matches? Propensity Scores and Bias in the Causal Effects of Education on Participation by former Berkeley PhD students John Henderson and Sara Chatfield. Their paper is itself a replication study of Reconsidering the Effects of Education on Political Participation by Cindy Kam and Carl Palmer. In their original 2008 study, Kam and Palmer argue that college education has no effect on later political participation, and use the propensity score matching to show that pre-college political activity drives selection into college and later political participation. Henderson and Chatfield in their 2011 paper argue that the use of the propensity score matching in this context is inappropriate because of the bias that arises from small changes in the choice of variables used to model the propensity score. They use genetic matching (at that point a new method), which uses an approach similar to optimal matching to optimize Mahalanobis distance weights. Even with genetic matching, they find that balance remains elusive however, thus leaving open the question of whether education causes political participation.

You will use these data and debates to investigate the benefits and pitfalls associated with matching methods. Replication code for these papers is available online, but as you'll see, a lot has changed in the last decade or so of data science! Throughout the assignment, use tools we introduced in lab from the tidyverse and the MatchIt packages. Specifically, try to use dplyr, tidyr, purrr, stringr, and ggplot instead of base R functions. While there are other matching software libraries available, MatchIt tends to be the most up to date and allows for consistent syntax.

Data

The data is drawn from the Youth-Parent Socialization Panel Study which asked students and parents a variety of questions about their political participation. This survey was conducted in several waves. The first wave was in 1965 and established the baseline pre-treatment covariates. The treatment is whether the student attended college between 1965 and 1973 (the time when the next survey wave was administered). The outcome is an index that calculates the number of political activities the student engaged in after 1965. Specifically, the key variables in this study are:

- **college**: Treatment of whether the student attended college or not. 1 if the student attended college between 1965 and 1973, 0 otherwise.
- **ppnscal**: Outcome variable measuring the number of political activities the student participated in. Additive combination of whether the student voted in 1972 or 1980 (student_vote), attended a campaign rally or meeting (student_meeting), wore a campaign button (student_button), donated money to a campaign (student_money), communicated with an elected official (student_communicate), attended a demonstration or protest (student_demonstrate), was involved with a local community event (student_community), or some other political participation (student_other)

Otherwise, we also have covariates measured for survey responses to various questions about political attitudes. We have covariates measured for the students in the baseline year, covariates for their parents in the baseline year, and covariates from follow-up surveys. **Be careful here**. In general, post-treatment covariates will be clear from the name (i.e. student_1973Married indicates whether the student was married in the 1973 survey). Be mindful that the baseline covariates were all measured in 1965, the treatment occurred between 1965 and 1973, and the outcomes are from 1973 and beyond. We will distribute the Appendix from Henderson and Chatfield that describes the covariates they used, but please reach out with any questions if you have questions about what a particular variable means.

```r
### Start with clean state
gc(); rm(list=ls())
```

```
##          used (Mb) gc trigger (Mb) max used (Mb)
## Ncells 474005 25.4    1024054 54.7   660491 35.3
## Vcells 884942  6.8    8388608 64.0  1769953 13.6
```

```r
### Set working directory and data directory
work_dir <- c("C:/Users/Sungkyu/Documents/HM")



### Call libraries
library(tidyverse); library(MatchIt); library(ggplot2); library(optmatch); library(cobalt); library(gri
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
##  cobalt (Version 4.5.5, Build Date: 2024-04-02)
##
##
## Attaching package: 'cobalt'
##
##
## The following object is masked from 'package:MatchIt':
##
##     lalonde
##
##
##
## Attaching package: 'gridExtra'
##
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
##
##
## Attaching package: 'reshape2'
```

```
##
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```r
### Call function
list.files(file.path(work_dir, "functions"), full.names = TRUE) %>% walk(source)



### Load ypsps data
ypsps <- read.csv('ypsps.csv')
```

Randomization

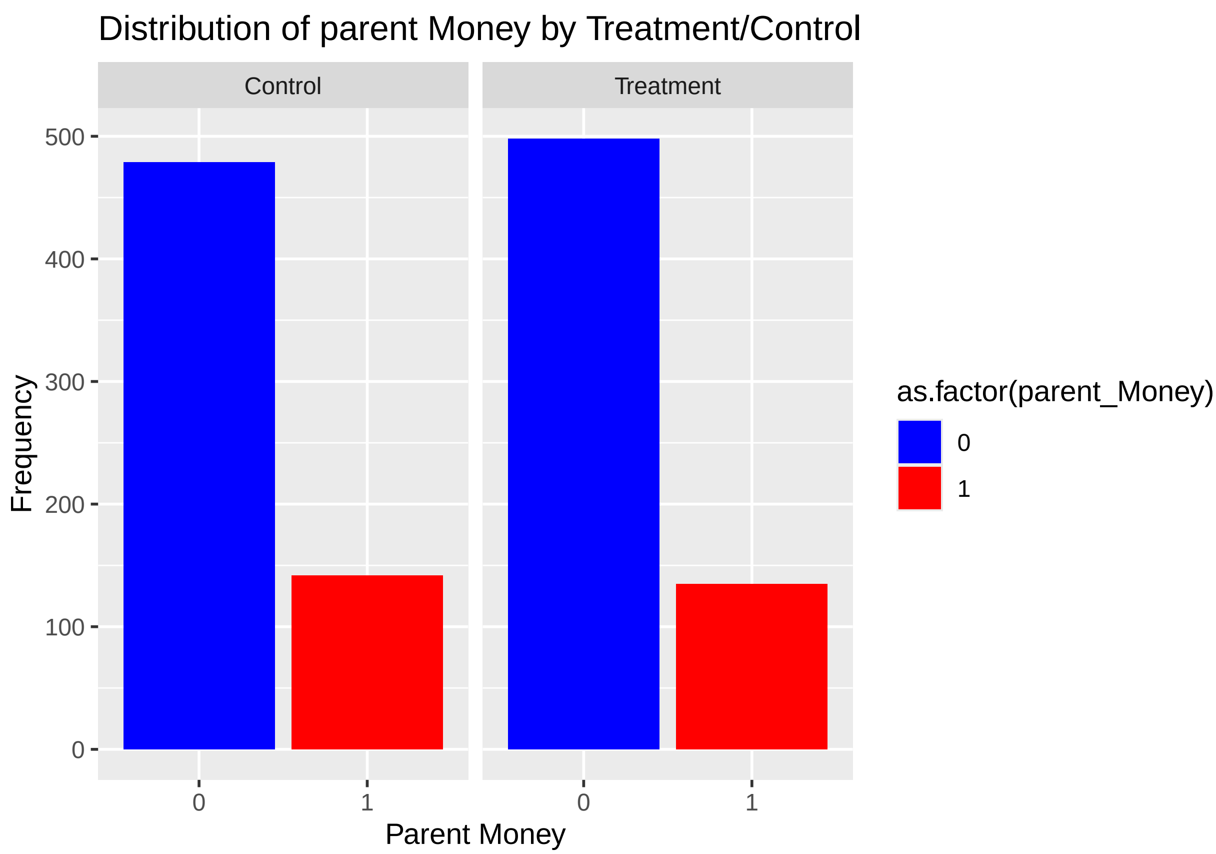
Matching is usually used in observational studies to to approximate random assignment to treatment. But could it be useful even in randomized studies? To explore the question do the following:

1. Generate a vector that randomly assigns each unit to either treatment or control
2. Choose a baseline covariate (for either the student or parent). A binary covariate is probably best for this exercise.
3. Visualize the distribution of the covariate by treatment/control condition. Are treatment and control balanced on this covariate?
4. Simulate the first 3 steps 10,000 times and visualize the distribution of treatment/control balance across the simulations.

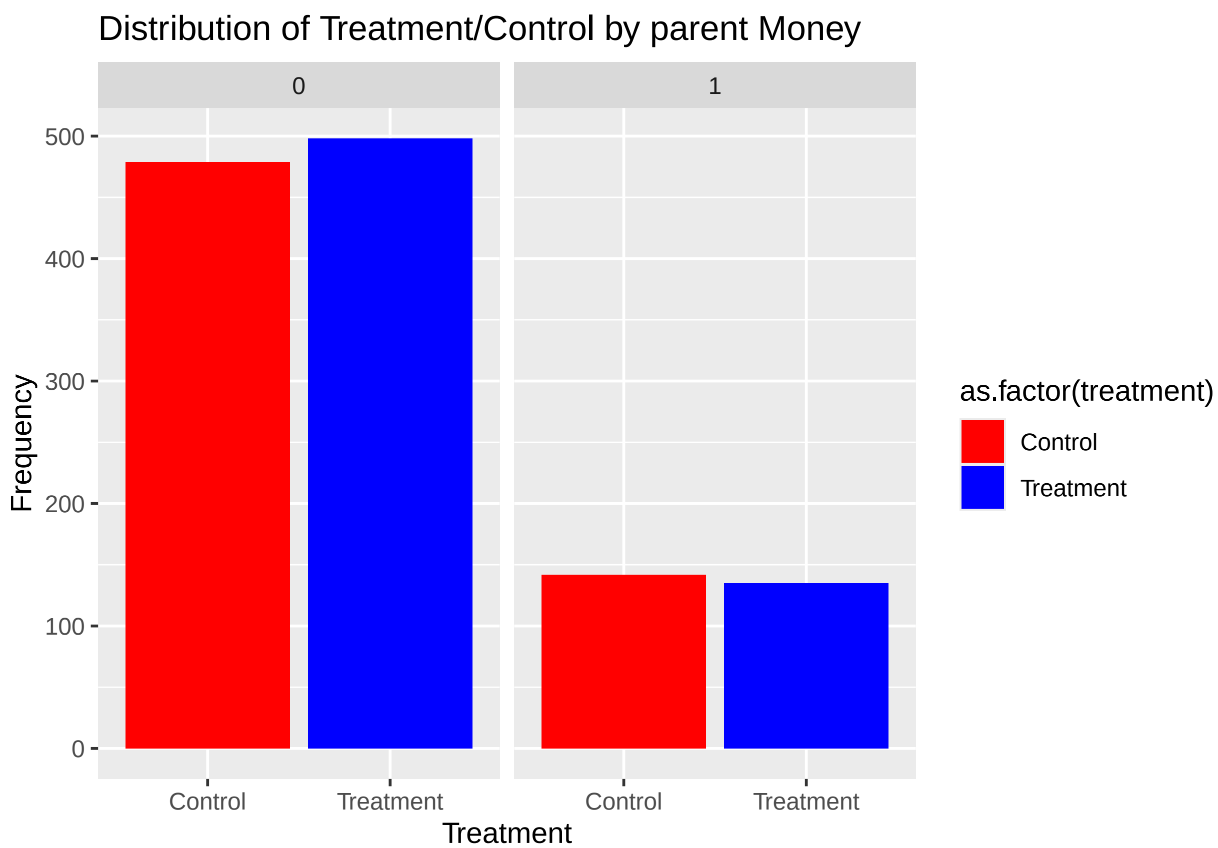
```r
##1. Generate a vector that randomly assigns each unit to treatment/control
set.seed(123)
treatment <- sample(c("Treatment", "Control"), size = nrow(ypsps), replace = TRUE)
ypsps <- ypsps %>% mutate(treatment = treatment)


##2. Choose a baseline covariate
df <- ypsps %>% select(treatment, parent_Money)
table(df$parent_Money)
```

```
##
##   0   1
## 977 277
```

```r
##3-1. Visualize the distribution by treatment/control (ggplot)
ggplot(df, aes(x = as.factor(parent_Money), fill = as.factor(parent_Money))) +
  geom_bar(position = "dodge") +
  facet_wrap(.~treatment) +
  labs(title = "Distribution of parent Money by Treatment/Control",
       x = "Parent Money",
       y = "Frequency") +
  scale_fill_manual(values = c("0" = "blue", "1" = "red"))
```




```r
##3-2. Visualize the distribution among treatment/control (ggplot)
ggplot(df, aes(x = as.factor(treatment), fill = as.factor(treatment))) +
  geom_bar(position = "dodge") +
  facet_wrap(.~parent_Money) +
  labs(title = "Distribution of Treatment/Control by parent Money",
       x = "Treatment",
       y = "Frequency") +
  scale_fill_manual(values = c("Treatment" = "blue", "Control" = "red"))
```

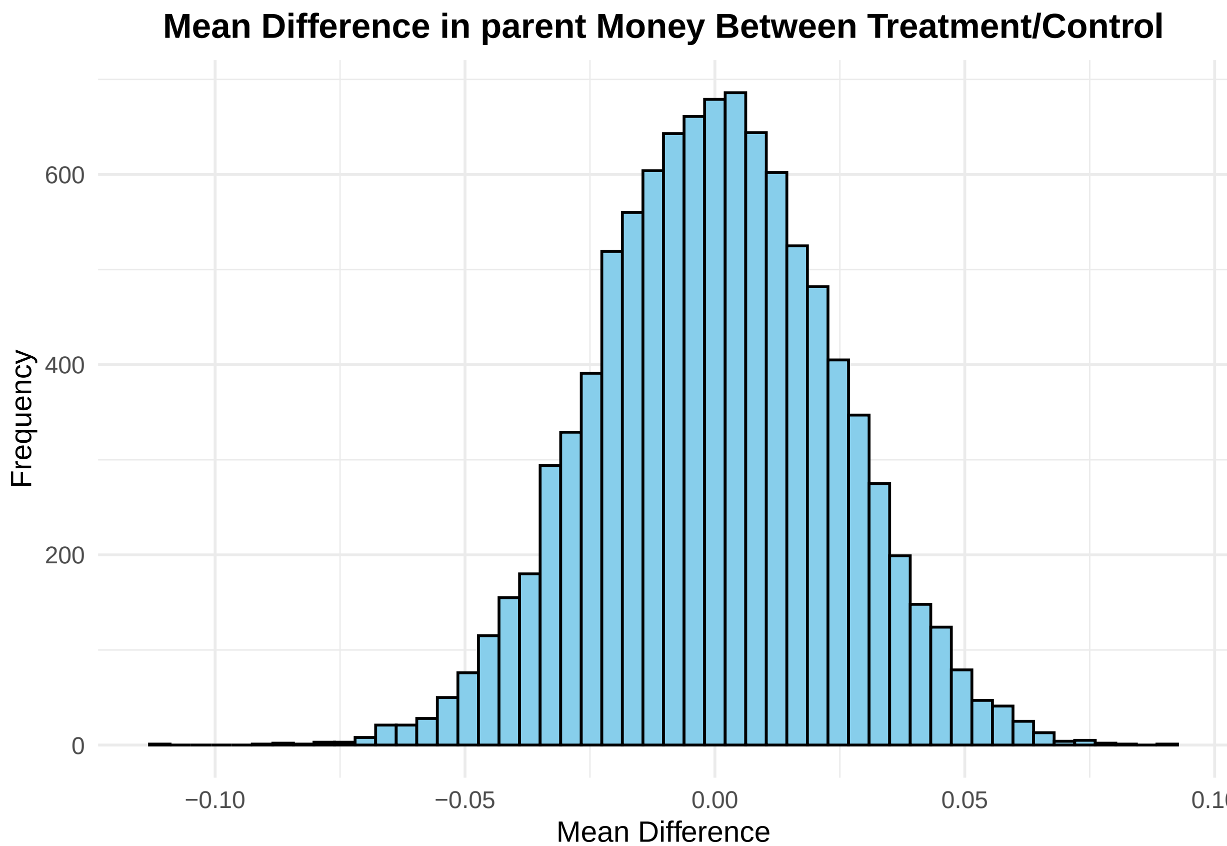



```r
##4. Simulate this 10,000 times (monte carlo simulation - see R Refresher for a hint)
simulations <- 10000
balance_results <- numeric(simulations)

for (i in 1:simulations) {

  treatment_sim <- sample(0:1, size = nrow(ypsps), replace = TRUE)  # randomly assign treatment/control

  treatment_data <- ypsps %>% mutate(treatment = treatment_sim) %>% filter(treatment==1) %>% select(trea
  control_data <- ypsps %>% mutate(treatment = treatment_sim) %>% filter(treatment==0) %>% select(treatm

  balance_results[i] <- mean(treatment_data$parent_Money) - mean(control_data$parent_Money)

}

summary(balance_results)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -1.124e-01 -1.635e-02 -1.095e-04 -3.943e-05  1.610e-02  8.913e-02
```

```r
# Visualize distribution of balance results
ggplot(data.frame(balance = balance_results), aes(x = balance, fill = ..count..)) +
  geom_histogram(bins = 50, fill = "skyblue", color = "black") +
  labs(title = "Mean Difference in parent Money Between Treatment/Control",
       x = "Mean Difference", y = "Frequency") +
```

```r
theme_minimal() +
theme(plot.title = element_text(hjust = 0.5, face = "bold"))
```



Questions

1. **What do you see across your simulations? Why does independence of treatment assignment and baseline covariates not guarantee balance of treatment assignment and baseline covariates?**

Your Answer:

Across the simulations, I observe a distribution of treatment/control balance, specifically focusing on the chosen baseline covariate. The balance represents how similar the treatment and control groups are in terms of the baseline covariate (parent_Money). Although the mean of the simulated balance is -3.943e-05 and slightly unbalanced, since it was distributed closely to zero, we can say there were no systematic differences between the treatment and control groups regarding the covariate. The independence of treatment assignment and baseline covariates does not guarantee the balance of treatment assignment and baseline covariates because random assignment does not necessarily result in perfectly balanced groups. Because there may be unobserved variables that influence both the treatment assignment and the baseline covariates as well as there is a random chance that may lead to unequal distributions of covariates between the treatment and control groups. Random assignment helps to mitigate the risk of systematic differences between the groups, but it does not eliminate the possibility entirely. This is why randomization tests, propensity score matching, or stratification are often used to assess and adjust for imbalances in observational studies. These methods aim to create more comparable treatment and control groups by accounting for observed differences in covariates.

Propensity Score Matching

One Model

Select covariates that you think best represent the "true" model predicting whether a student chooses to attend college, and estimate a propensity score model to calculate the Average Treatment Effect on the Treated (ATT). Plot the balance of the top 10 (or fewer if you select fewer covariates). Report the balance of the p-scores across both the treatment and control groups, and using a threshold of standardized mean difference of p-score $\leq .1$, report the number of covariates that meet that balance threshold.

```r
## Step 1: Select covariates and estimate propensity score model
# Select covariates representing the "true" model
covariates <- c("student_Gen", "student_Trust", "student_GovtOpinion", "student_Race",
                "parent_EducHH", "parent_Money", "parent_Race", "parent_GovtOpinion")

data <- ypsps %>% select(interviewid, college, student_ppnscal, all_of(covariates))


# Fit propensity score model
model <- glm(college ~ student_Gen + student_Trust + student_GovtOpinion + student_Race +
                  parent_EducHH + parent_Money + parent_Race + parent_GovtOpinion,
                data = data,
                family = binomial())


# Calculate propensity scores
data$prop_score <- predict(model, type = "response")


## Step 2: Estimate ATT using MatchIt package
# Estimate ATT
match_ps <- matchit(college ~ student_Gen + student_Trust + student_GovtOpinion + student_Race +
                       parent_EducHH + parent_Money + parent_Race + parent_GovtOpinion,
                     data = data,
                     method = "nearest",
                     distance = "glm",
                     link = "logit",
                     estimand = "ATT")

# Summary of matched data
match_summ <- summary(match_ps, un = FALSE)
match_summ$sum.matched[ , "Std. Mean Diff."]
```

```
##            distance         student_Gen       student_Trust student_GovtOpinion
##          1.71970686          0.38314027          0.21311245         -0.38994782
##        student_Race       parent_EducHH        parent_Money         parent_Race
##         -0.09452711          1.44887439          0.75432017         -0.20135108
##  parent_GovtOpinion
##         -0.41663117
```

```r
match_summ$nn
```

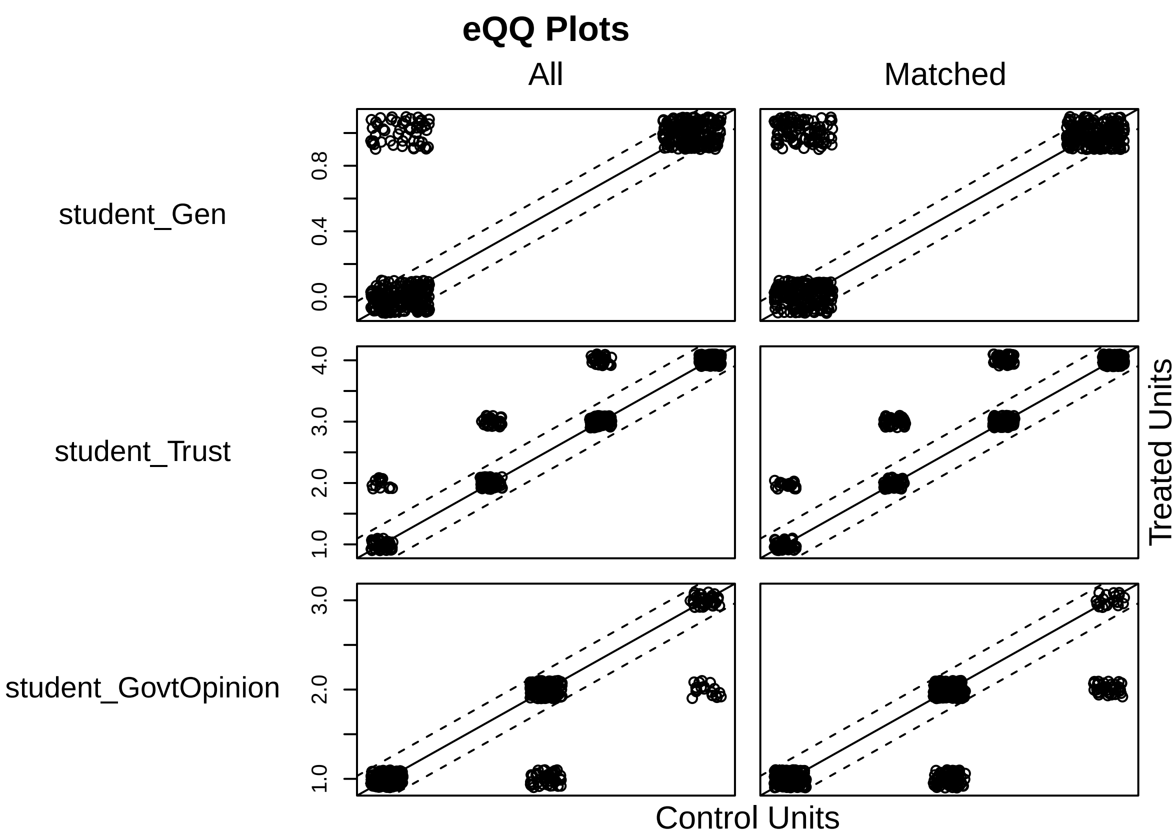
```
##                 Control Treated
```

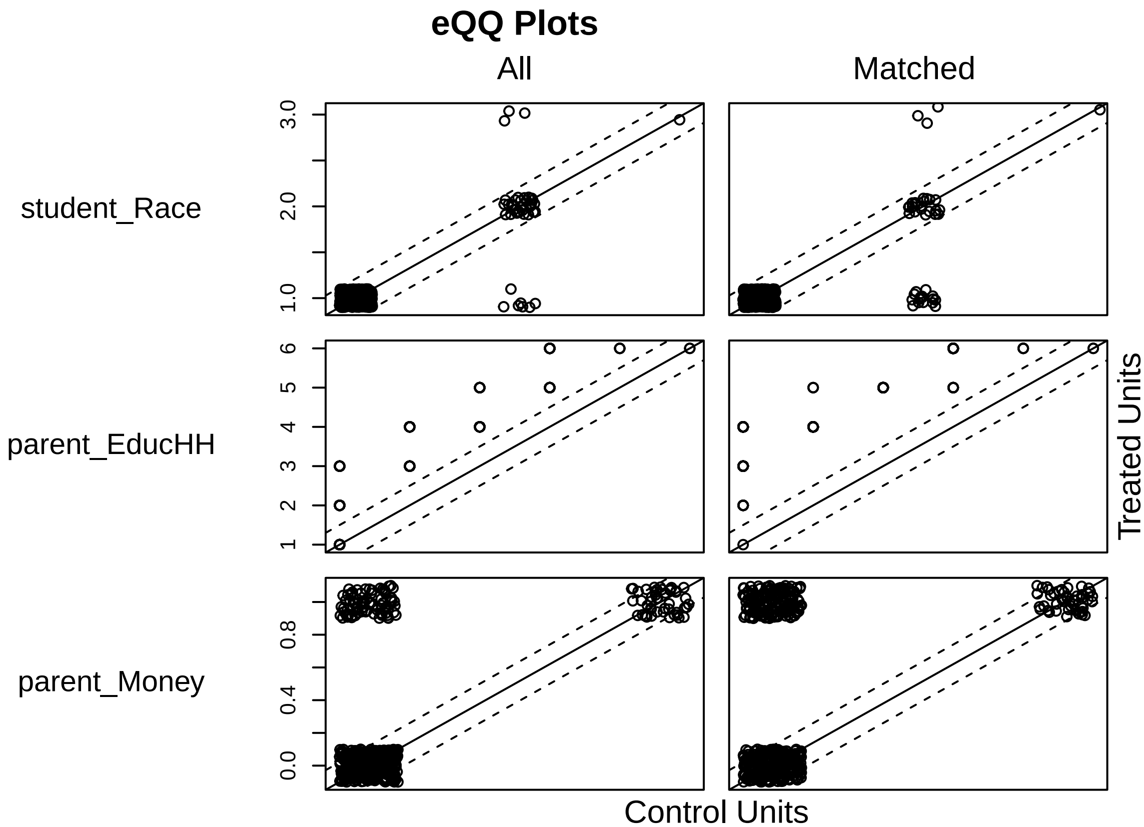
```
## All (ESS)          451      803
## All                451      803
## Matched (ESS)      451      451
## Matched            451      451
## Unmatched            0      352
## Discarded            0        0
```

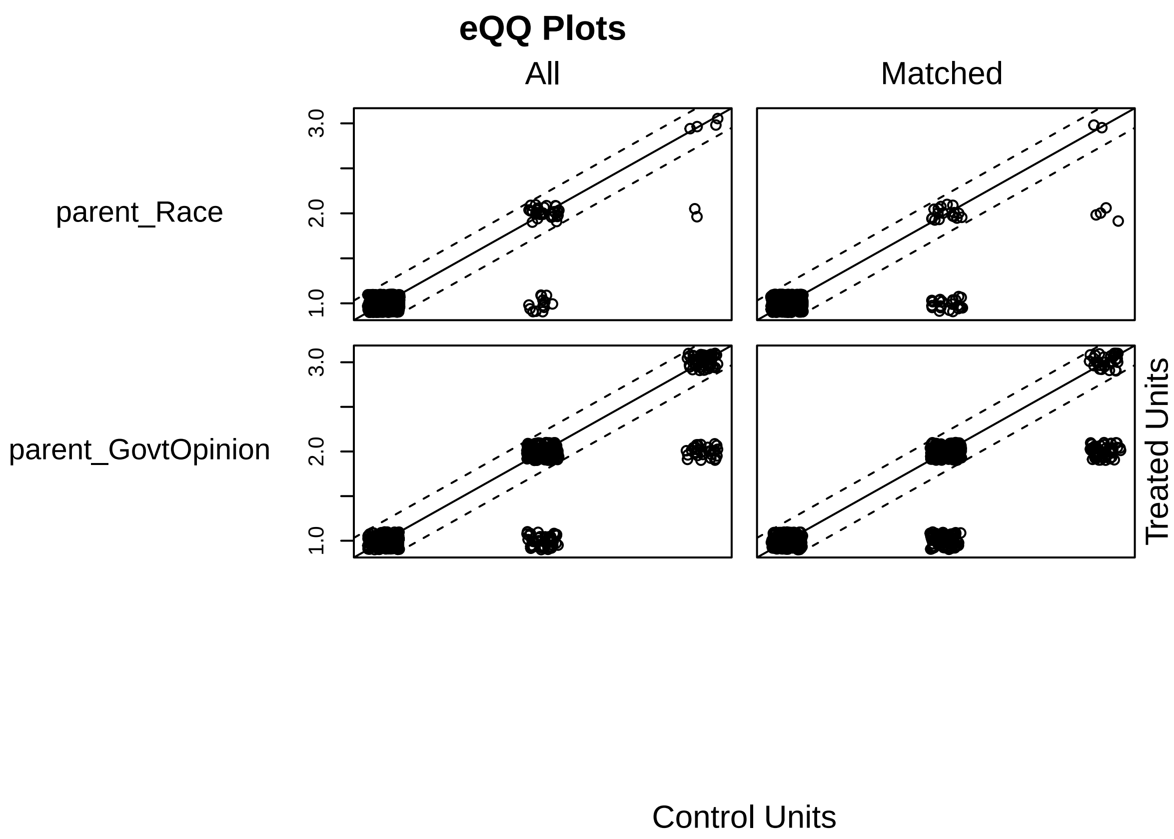
```
match_att_data <- match.data(match_ps)

STU_T1 <- match_att_data %>% filter(college == 1) %>% summarise(mean(student_ppnscal)) %>% pull()
STU_T0 <- match_att_data %>% filter(college == 0) %>% summarise(mean(student_ppnscal)) %>% pull()
ATT <- STU_T1 - STU_T0
ATT
```

```
## [1] 1.609756
```

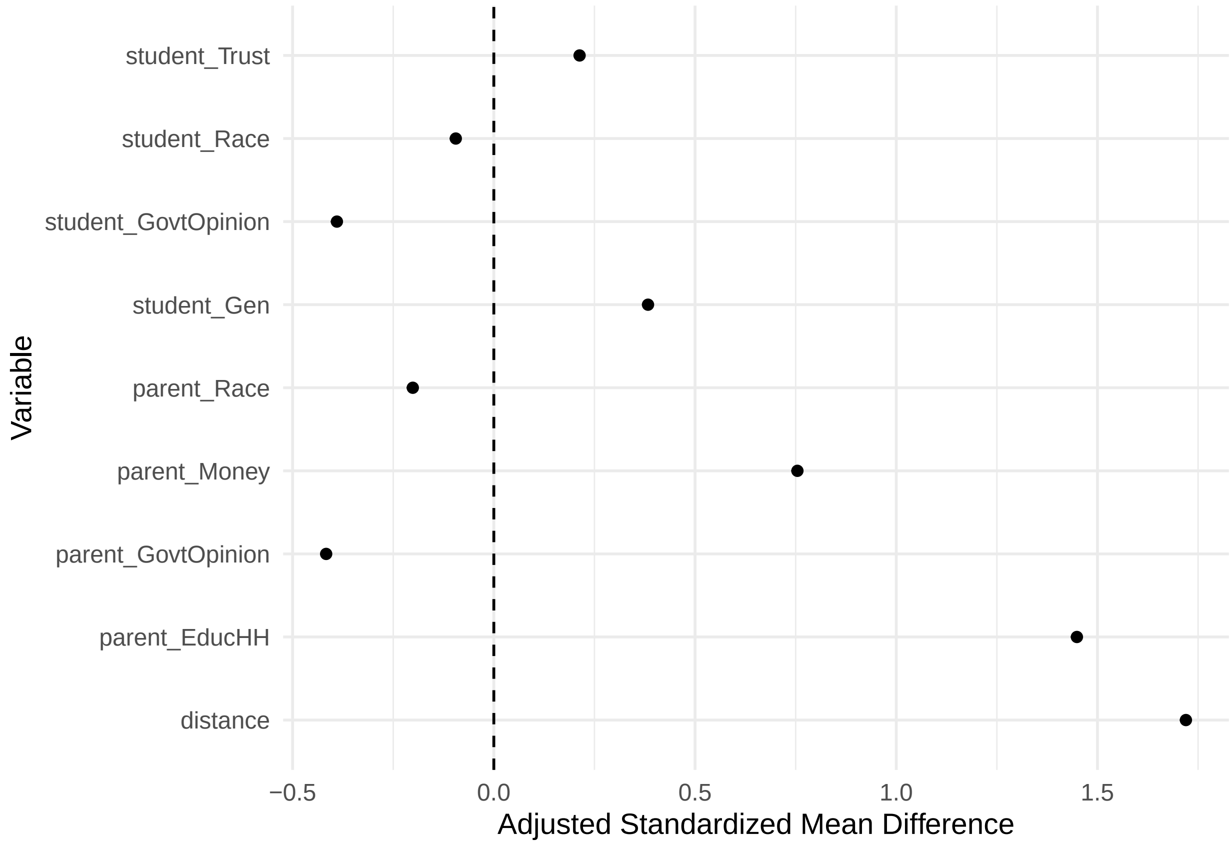
```
plot(match_ps)
```



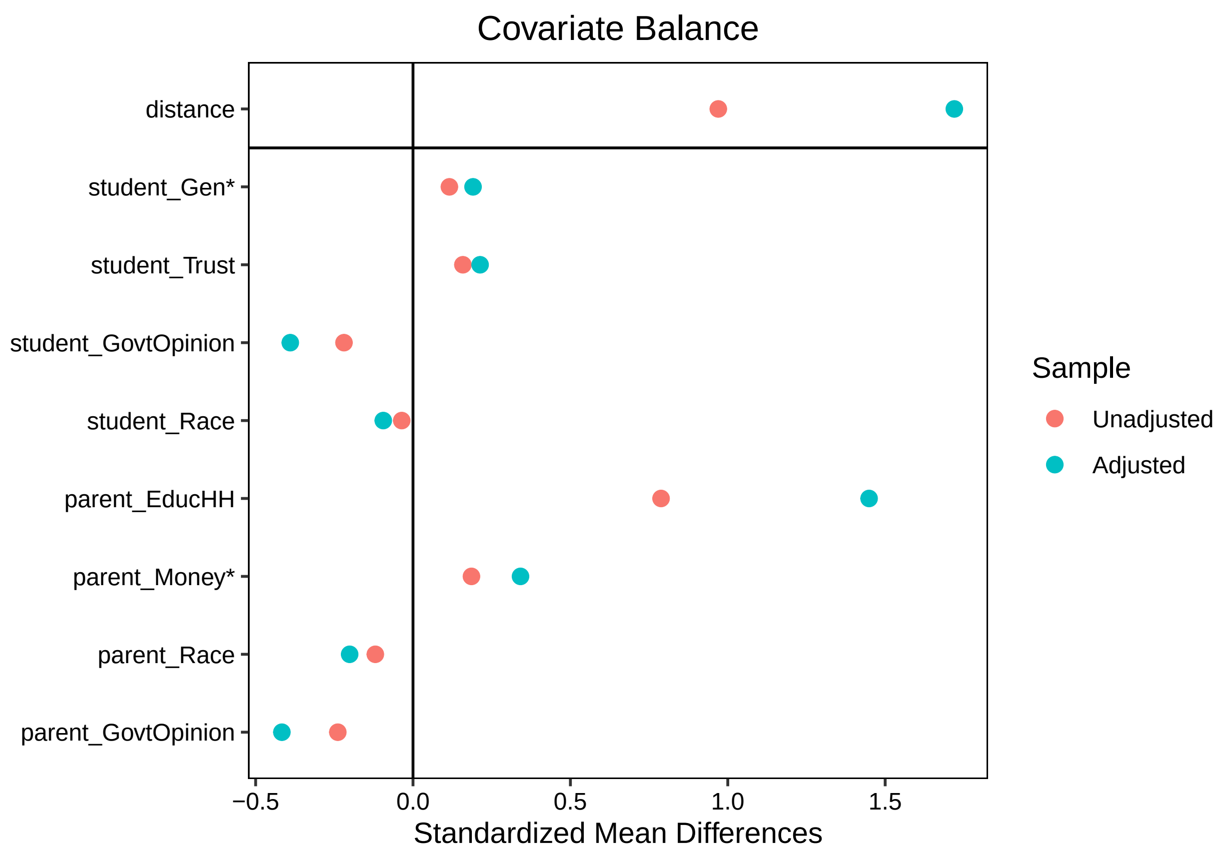

eQQ Plots
All
Matched
student_Race
parent_EducHH
parent_Money
3.0
2.0
1.0
6
5
4
3
2
1
0.8
0.4
0.0
Treated Units
Control Units



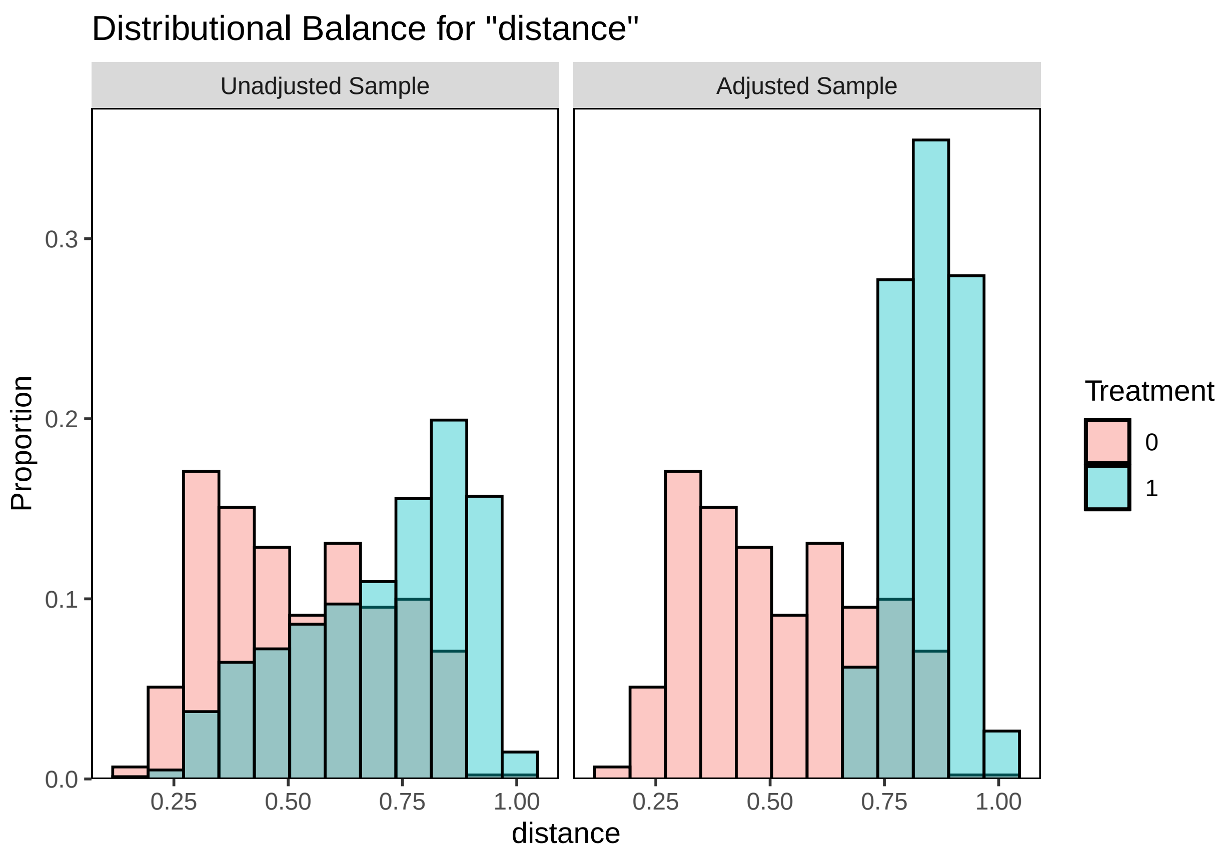

```r
# Plot using ggplot
plot_data <- data.frame(
  Variable = rownames(match_summ$sum.matched),
  Std_Mean_Diff = match_summ$sum.matched[, "Std. Mean Diff."]
)

ggplot(plot_data, aes(x = Std_Mean_Diff, y = Variable)) +
  geom_point() +
  geom_vline(xintercept = 0, linetype = "dashed") +
  labs(x = "Adjusted Standardized Mean Difference", y = "Variable") +
  theme_minimal()
```



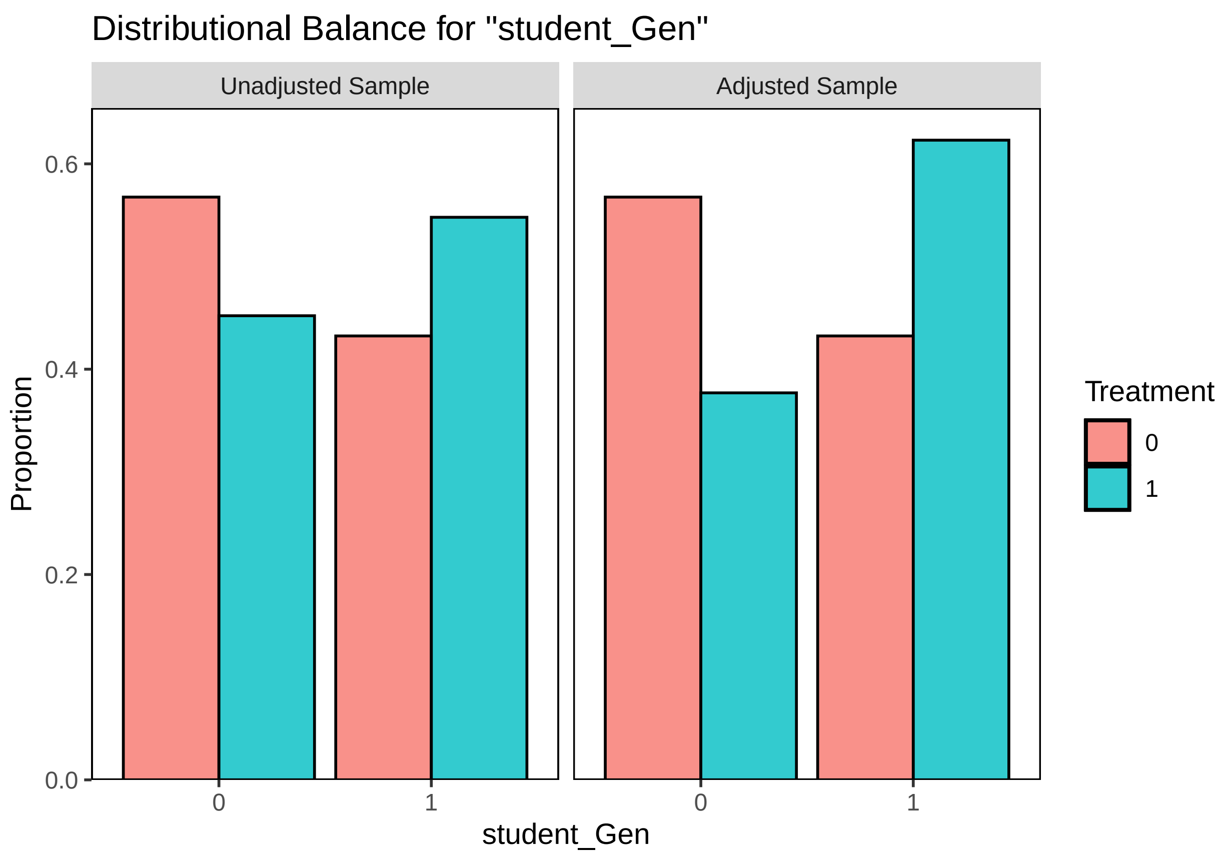

```
# love.plot
love.plot(match_ps, stars = "raw")
```



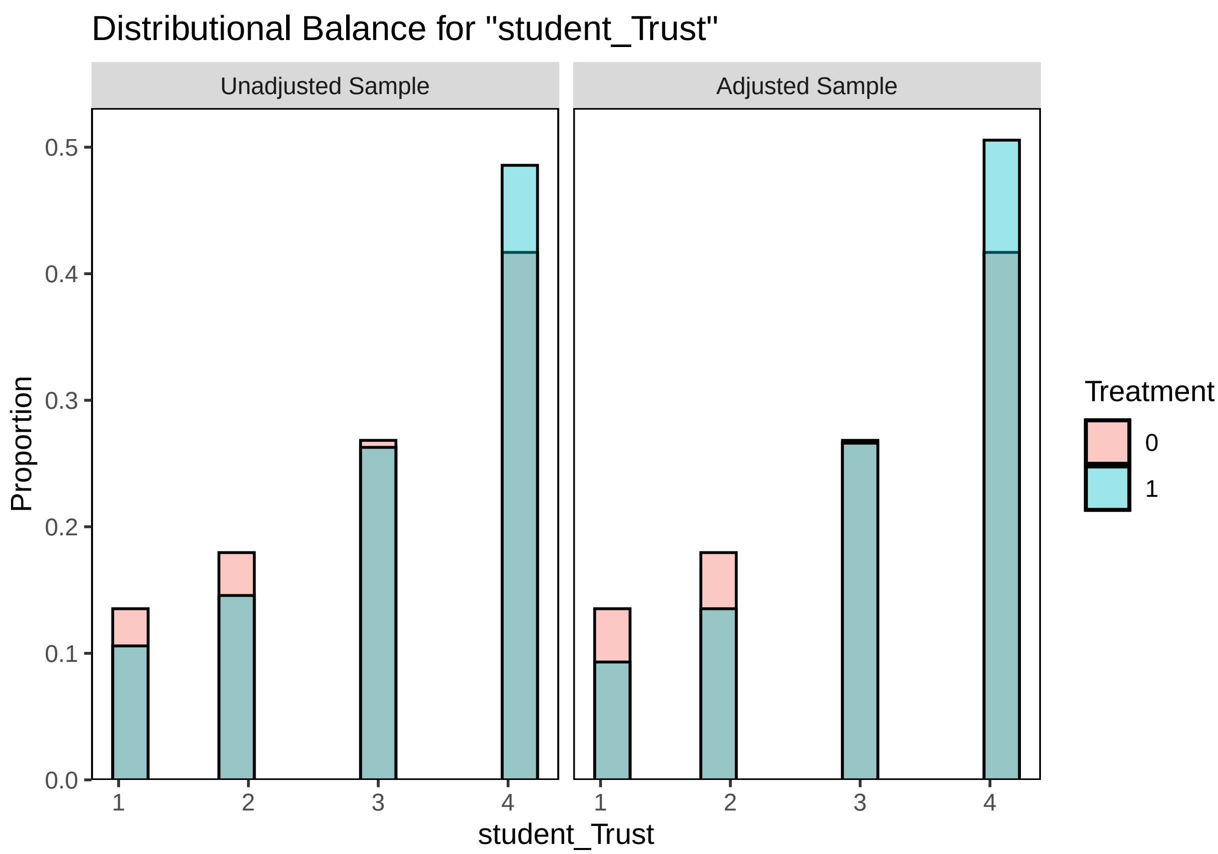

```
# Balance plot for top 8 covariates, unadjusted and adjusted
bal.plot(match_ps, var.name = "distance", which = "both", type = "histogram")
```



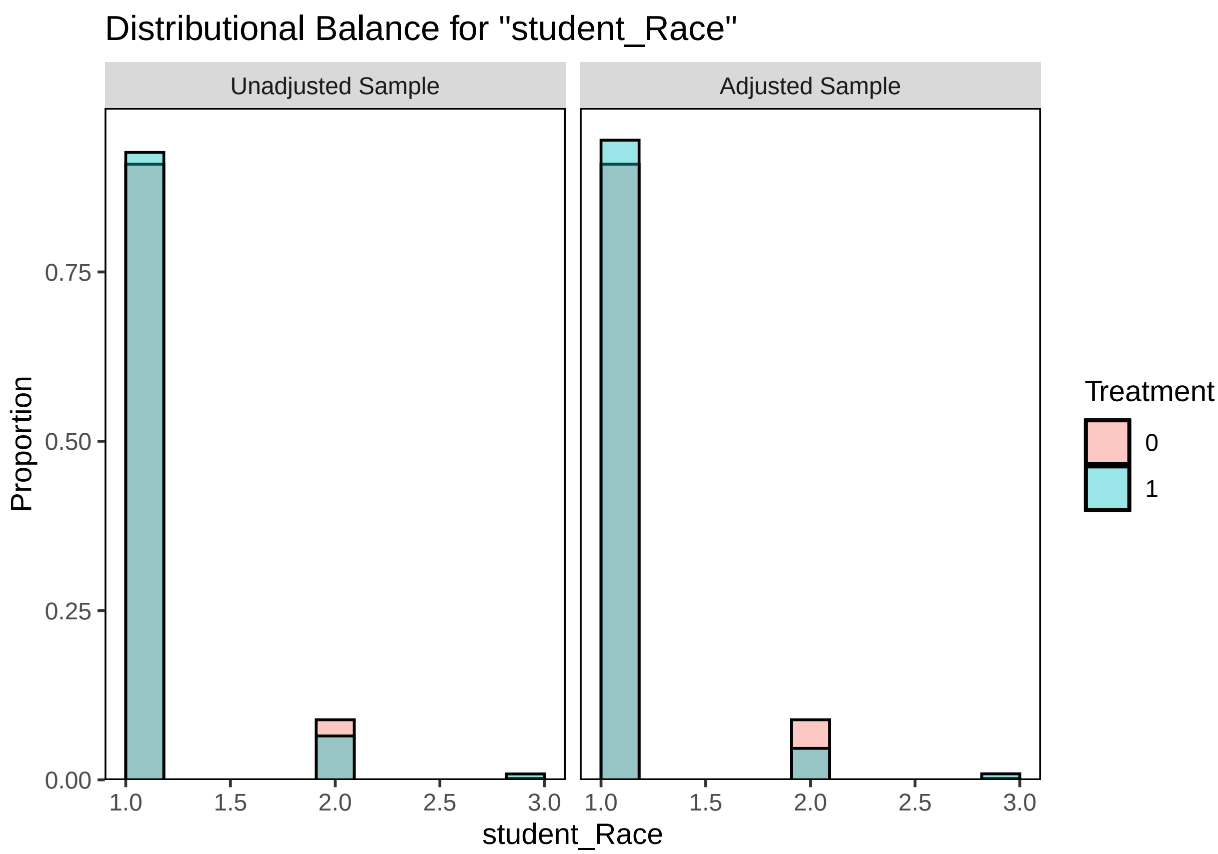

```
par(mfrow=c(2, 4))
bal.plot(match_ps, var.name = "student_Gen", which = "both", type = "histogram")
```



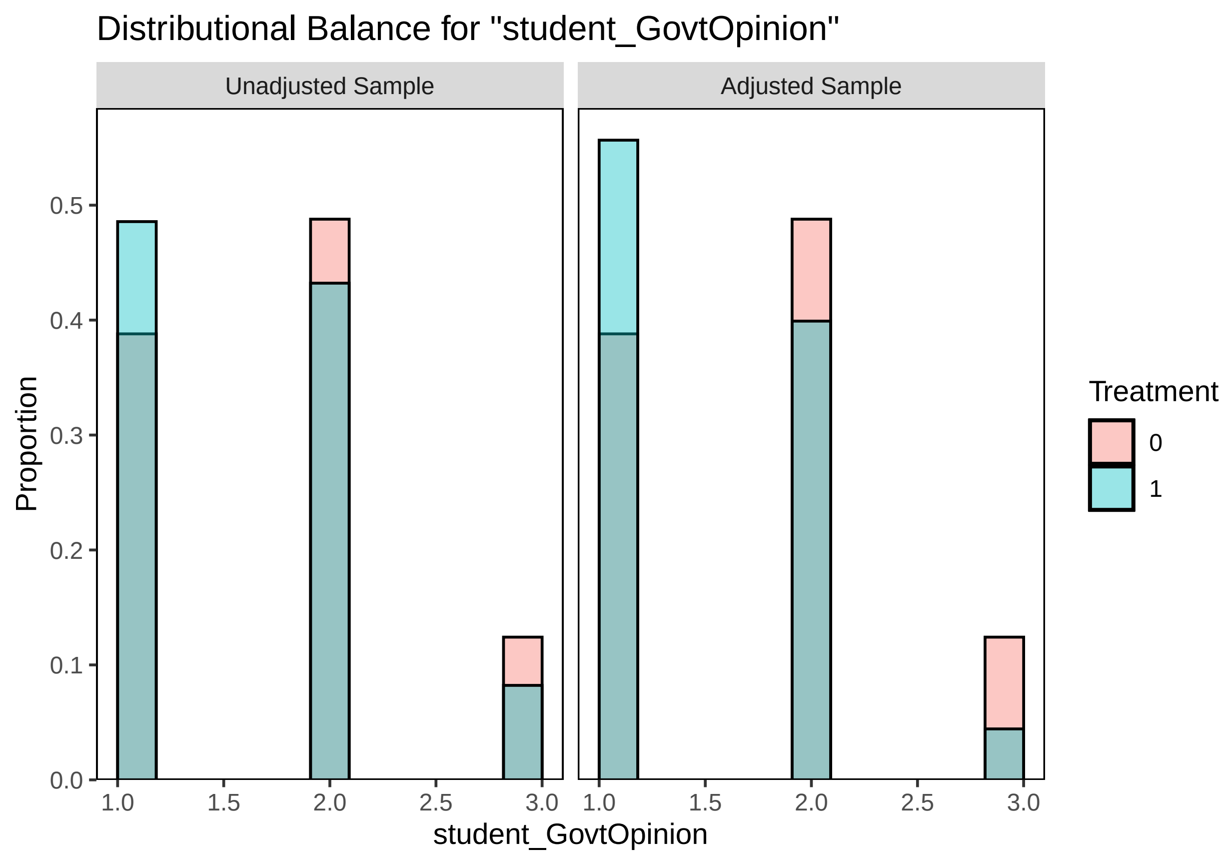

```
bal.plot(match_ps, var.name = "student_Trust", which = "both", type = "histogram")
```



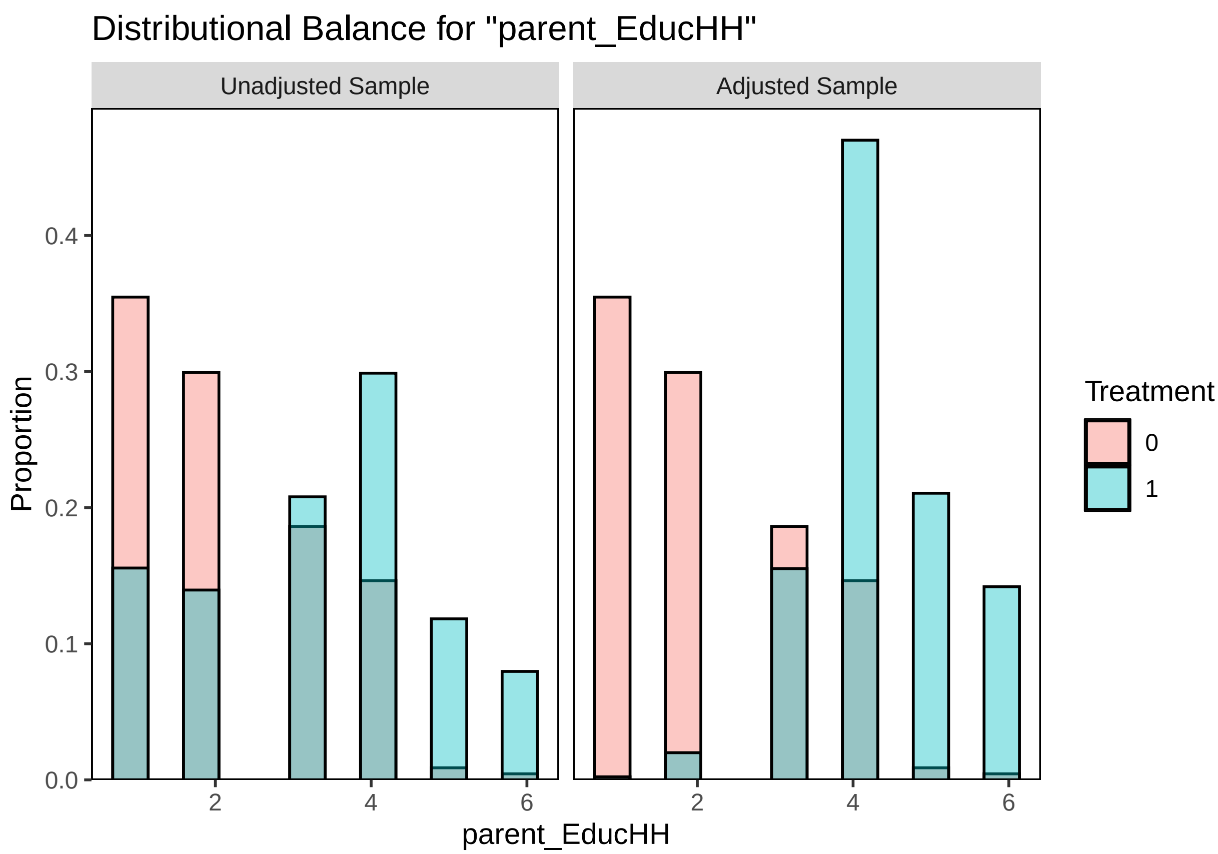

```r
bal.plot(match_ps, var.name = "student_Race", which = "both", type = "histogram")
```



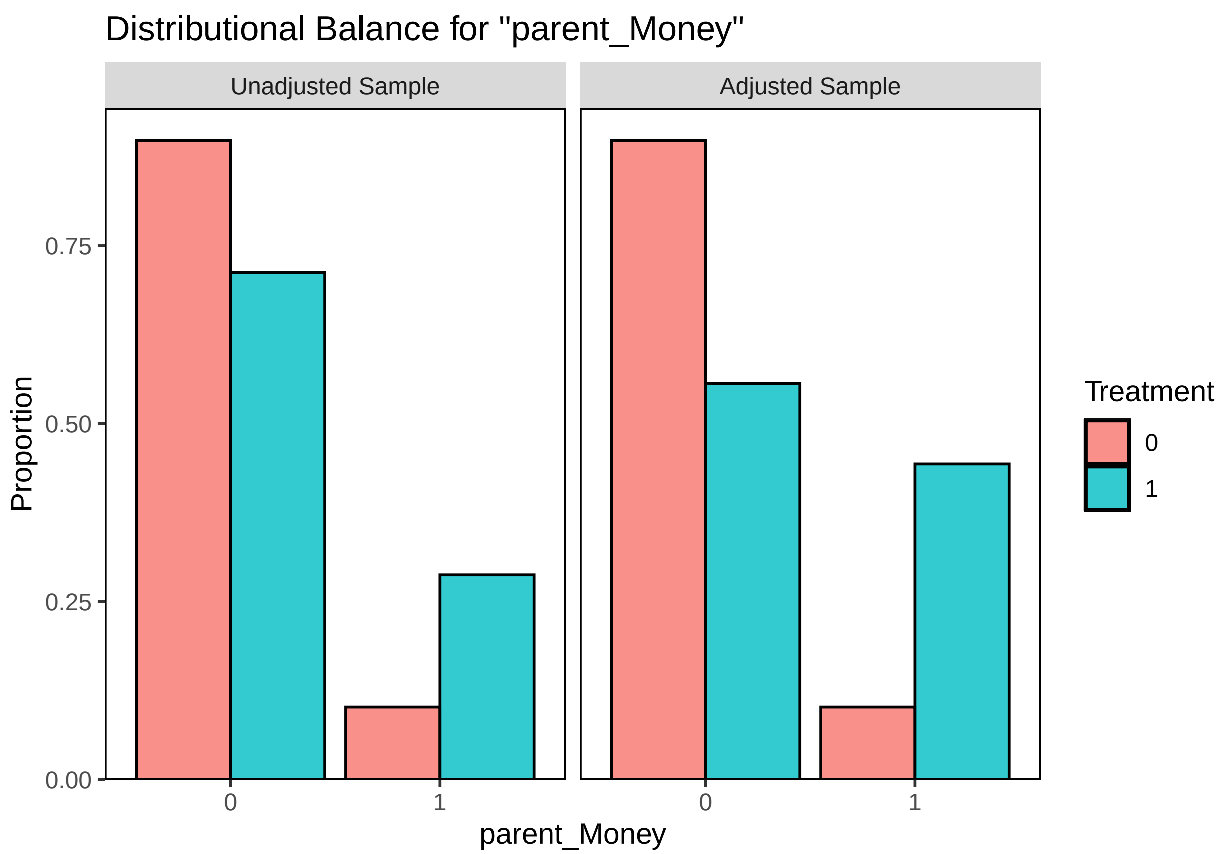

```
bal.plot(match_ps, var.name = "student_GovtOpinion", which = "both", type = "histogram")
```



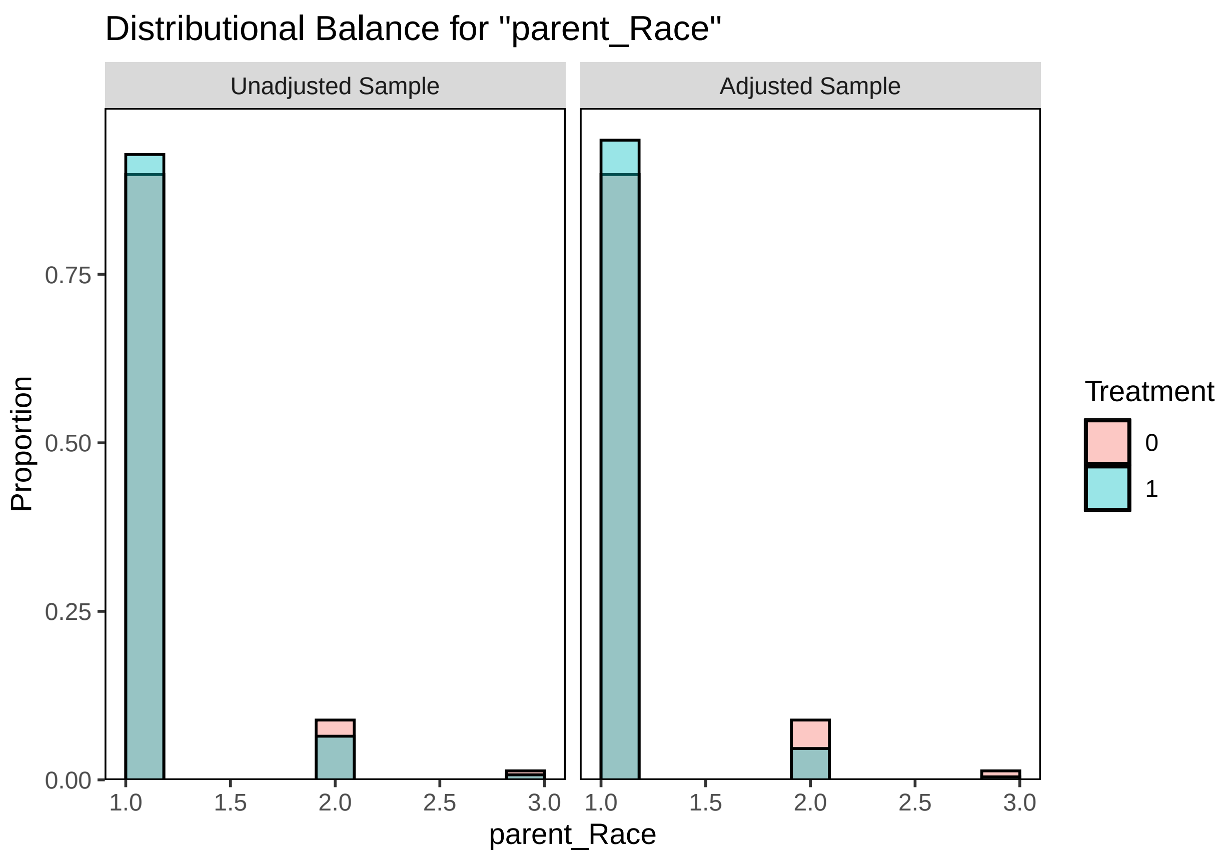

```r
bal.plot(match_ps, var.name = "parent_EducHH", which = "both", type = "histogram")
```



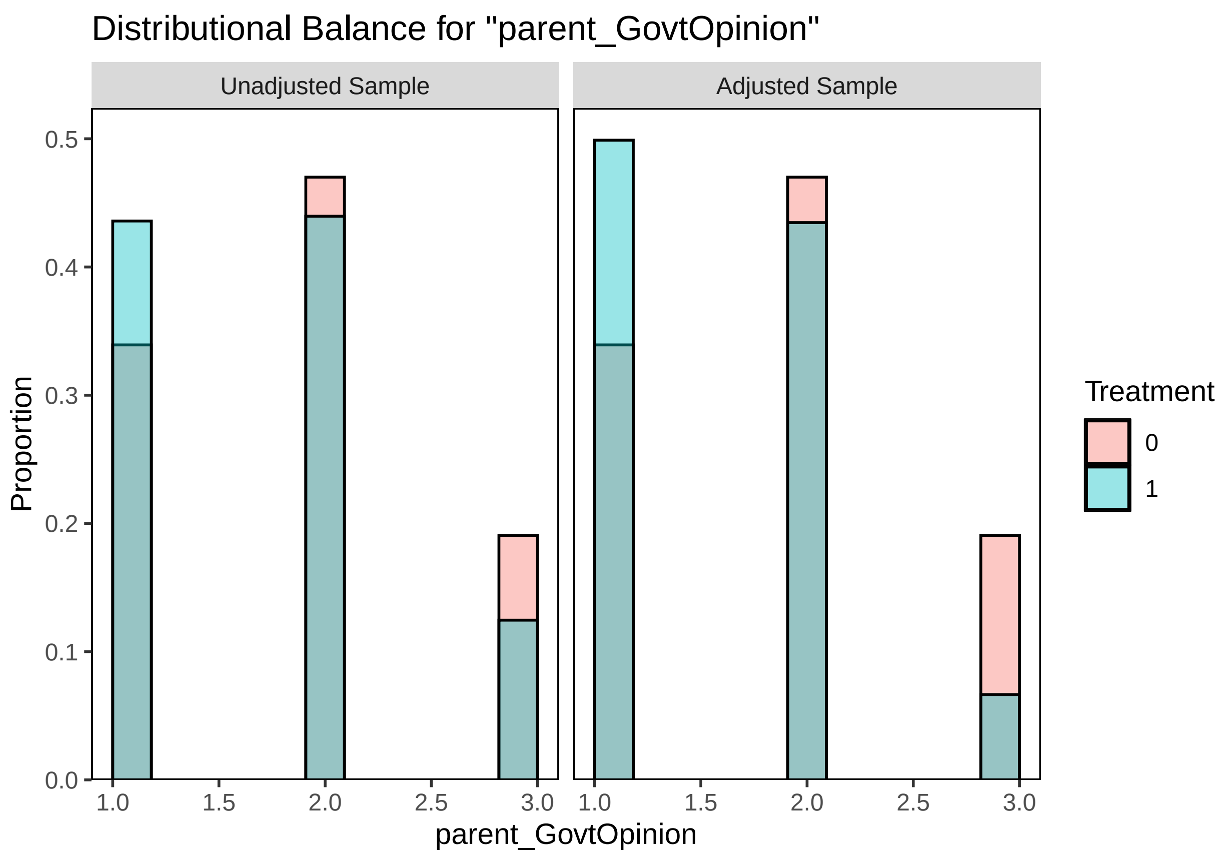

```
bal.plot(match_ps, var.name = "parent_Money", which = "both", type = "histogram")
```




```r
bal.plot(match_ps, var.name = "parent_Race", which = "both", type = "histogram")
```




```
bal.plot(match_ps, var.name = "parent_GovtOpinion", which = "both", type = "histogram")
```




```r
# number of covariates that meet that balance threshold
bal.tab(match_ps, binary = "std", treshold = 0.1)
```

```
## Balance Measures
##                          Type Diff.Adj
## distance             Distance   1.7197
## student_Gen            Binary   0.3831
## student_Trust         Contin.   0.2131
## student_GovtOpinion   Contin.  -0.3899
## student_Race          Contin.  -0.0945
## parent_EducHH         Contin.   1.4489
## parent_Money           Binary   0.7543
## parent_Race           Contin.  -0.2014
## parent_GovtOpinion    Contin.  -0.4166
##
## Sample sizes
##           Control Treated
## All           451     803
## Matched       451     451
## Unmatched       0     352
```

```r
balance <- bal.tab(match_ps, binary = "std", threshold = 0.1)
balance$M.Threshold <- ifelse(balance$`Standardized Diff.` <= 0.1, "Balanced, <0.1", "Not Balanced, >0.

# View the result
print(balance)
```

```
## Balance Measures
##                         Type Diff.Adj        M.Threshold
## distance            Distance   1.7197
## student_Gen           Binary   0.3831 Not Balanced, >0.1
## student_Trust        Contin.   0.2131 Not Balanced, >0.1
## student_GovtOpinion  Contin.  -0.3899 Not Balanced, >0.1
## student_Race         Contin.  -0.0945     Balanced, <0.1
## parent_EducHH        Contin.   1.4489 Not Balanced, >0.1
## parent_Money          Binary   0.7543 Not Balanced, >0.1
## parent_Race          Contin.  -0.2014 Not Balanced, >0.1
## parent_GovtOpinion   Contin.  -0.4166 Not Balanced, >0.1
##
## Balance tally for mean differences
##                    count
## Balanced, <0.1         1
## Not Balanced, >0.1     7
##
## Variable with the greatest mean difference
##       Variable Diff.Adj        M.Threshold
##  parent_EducHH   1.4489 Not Balanced, >0.1
##
## Sample sizes
##           Control Treated
## All           451     803
## Matched       451     451
## Unmatched       0     352
```

```
## ATT with lm
match_att_data <- match.data(match_ps)

lm_att <- lm(student_ppnscal ~ college + student_Gen + student_Trust + student_GovtOpinion +
               parent_EducHH + parent_Money + parent_Race + parent_GovtOpinion,
             data = match_att_data,
             weights = weights)

summary(lm_att)
```

```
##
## Call:
## lm(formula = student_ppnscal ~ college + student_Gen + student_Trust +
##     student_GovtOpinion + parent_EducHH + parent_Money + parent_Race +
##     parent_GovtOpinion, data = match_att_data, weights = weights)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9956 -1.2660 -0.2881  0.9588  6.8147
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          0.409115   0.413973   0.988  0.32329
## college              1.218014   0.173290   7.029 4.14e-12 ***
```

```
## student_Gen          0.028162   0.118263   0.238  0.81183
## student_Trust        0.120833   0.056790   2.128  0.03363 *
## student_GovtOpinion -0.158367   0.093224  -1.699  0.08971 .
## parent_EducHH        0.139301   0.056361   2.472  0.01364 *
## parent_Money         0.165215   0.141561   1.167  0.24348
## parent_Race          0.549873   0.188282   2.920  0.00358 **
## parent_GovtOpinion  -0.004908   0.087414  -0.056  0.95523
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.728 on 893 degrees of freedom
## Multiple R-squared:  0.1962, Adjusted R-squared:  0.189
## F-statistic: 27.25 on 8 and 893 DF,  p-value: < 2.2e-16
```

```r
att_ps <- lm_att$coefficients["college"]
summary(att_ps)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.218   1.218   1.218   1.218   1.218   1.218
```

Simulations

Henderson/Chatfield argue that an improperly specified propensity score model can actually *increase* the bias of the estimate. To demonstrate this, they simulate 800,000 different propensity score models by choosing different permutations of covariates. To investigate their claim, do the following:

- Using as many simulations as is feasible (at least 10,000 should be ok, more is better!), randomly select the number of and the choice of covariates for the propensity score model.
- For each run, store the ATT, the proportion of covariates that meet the standardized mean difference $\leq .1$ threshold, and the mean percent improvement in the standardized mean difference. You may also wish to store the entire models in a list and extract the relevant attributes as necessary.
- Plot all of the ATTs against all of the balanced covariate proportions. You may randomly sample or use other techniques like transparency if you run into overplotting problems. Alternatively, you may use plots other than scatterplots, so long as you explore the relationship between ATT and the proportion of covariates that meet the balance threshold.
- Finally choose 10 random models and plot their covariate balance plots (you may want to use a library like gridExtra to arrange these)

Note: There are lots of post-treatment covariates in this dataset (about 50!)! You need to be careful not to include these in the pre-treatment balancing. Many of you are probably used to selecting or dropping columns manually, or positionally. However, you may not always have a convenient arrangement of columns, nor is it fun to type out 50 different column names. Instead see if you can use dplyr 1.0.0 functions to programatically drop post-treatment variables (here is a useful tutorial).

```r
# Remove post-treatment covariates
set.seed(123)
ypsps_clean <- ypsps %>%
  select(-c(matches("_19\\d\\d"),'interviewid','treatment',matches('\\wPlacebo'))) %>%
  filter(complete.cases(.))
```

```r
ATTs <- c()
prop_balanced <- c()
percent_imp <- c()


# Randomly select features
# Simulate random selection of features 1k+ times
for (i in 1:1000) {

  # Randomly select number of columns
  num_cols <- sample(3:ncol(ypsps_clean)-2, 1)
  colnames <- sample(names(ypsps_clean %>% select(-c(student_ppnscal, college))), num_cols)

  independent <- paste(colnames, collapse = " + ")
  ps_formula <- as.formula(paste("college ~", independent))


  # Fit propensity score
  model_glm <- glm(formula = ps_formula,
                   data = ypsps_clean,
                   family = binomial())

  # Fit p-score models and save ATTs
  model_ps <- matchit(replace = TRUE,
                      formula = ps_formula,
                      data = ypsps_clean,
                      method = "nearest",
                      ratio = 1,
                      estimand = "ATT")

  match_att_data <- match.data(model_ps)

  STU_T1 <- match_att_data %>% filter(college == 1) %>% summarise(mean(student_ppnscal)) %>% pull()
  STU_T0 <- match_att_data %>% filter(college == 0) %>% summarise(mean(student_ppnscal)) %>% pull()
  ATTs[i] <- STU_T1 - STU_T0


  summ <- bal.tab(model_ps, threshold = .1)

  A <- summ[[1]] %>% select(Diff.Adj) %>% slice(2:n()) %>% pull
  prop_balanced[i] <- sum(abs(A) <= 0.1) / length(A)

  pre <- data.frame(summary(model_ps)$sum.all) %>% pull(Std..Mean.Diff.) %>% mean ()
  post <- data.frame(summary(model_ps)$sum.matched) %>% pull(Std..Mean.Diff.) %>% mean ()

  percent_imp[i] <- (post - pre) / pre

}



# Fit p-score models and save ATTs, proportion of balanced covariates, and mean percent balance improve
```

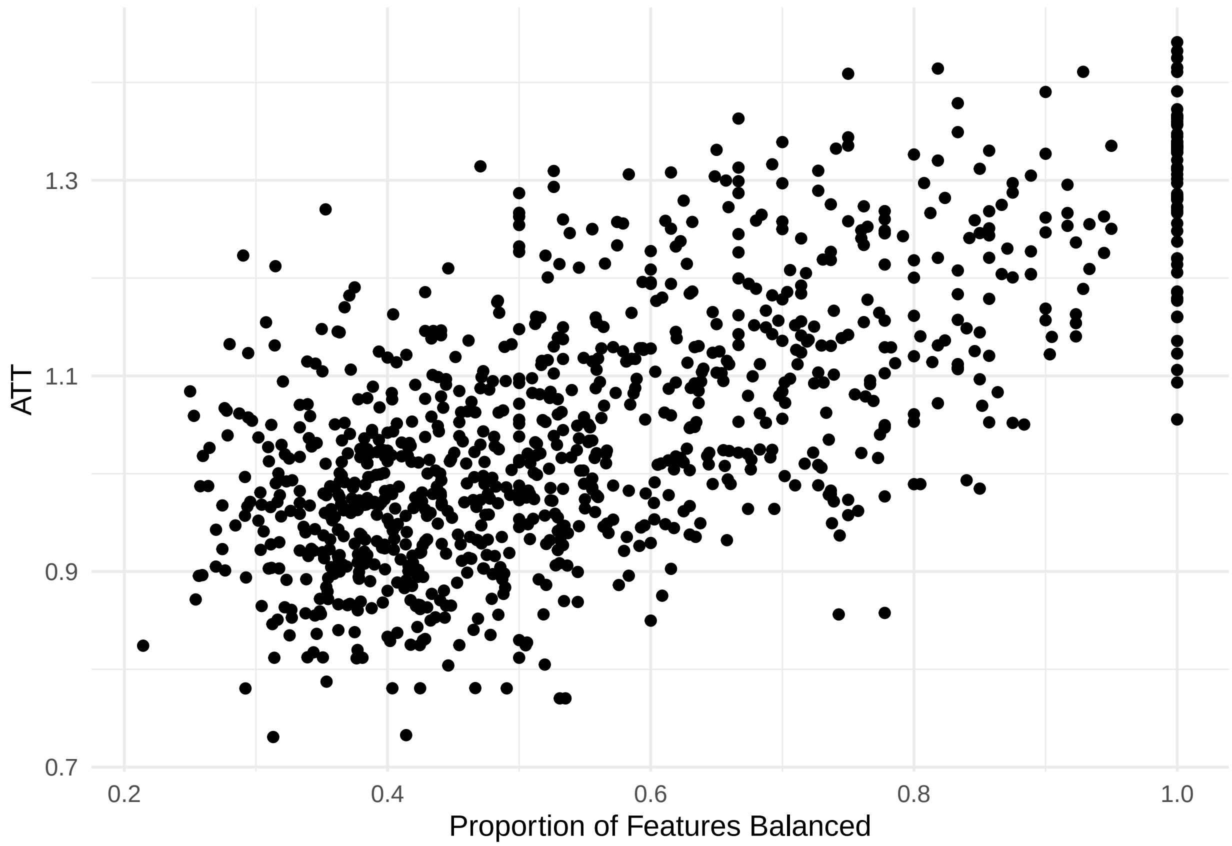
```r
sim_results <- data.frame(ATTs, prop_balanced, percent_imp)
summary(sim_results)
```

```
##       ATTs        prop_balanced     percent_imp
##  Min.   :0.7308   Min.   :0.2143   Min.   :-1.3814
##  1st Qu.:0.9557   1st Qu.:0.4013   1st Qu.:-0.9955
##  Median :1.0274   Median :0.5165   Median :-0.9127
##  Mean   :1.0545   Mean   :0.5611   Mean   :-0.9132
##  3rd Qu.:1.1431   3rd Qu.:0.6913   3rd Qu.:-0.8280
##  Max.   :1.4412   Max.   :1.0000   Max.   :-0.4119
```

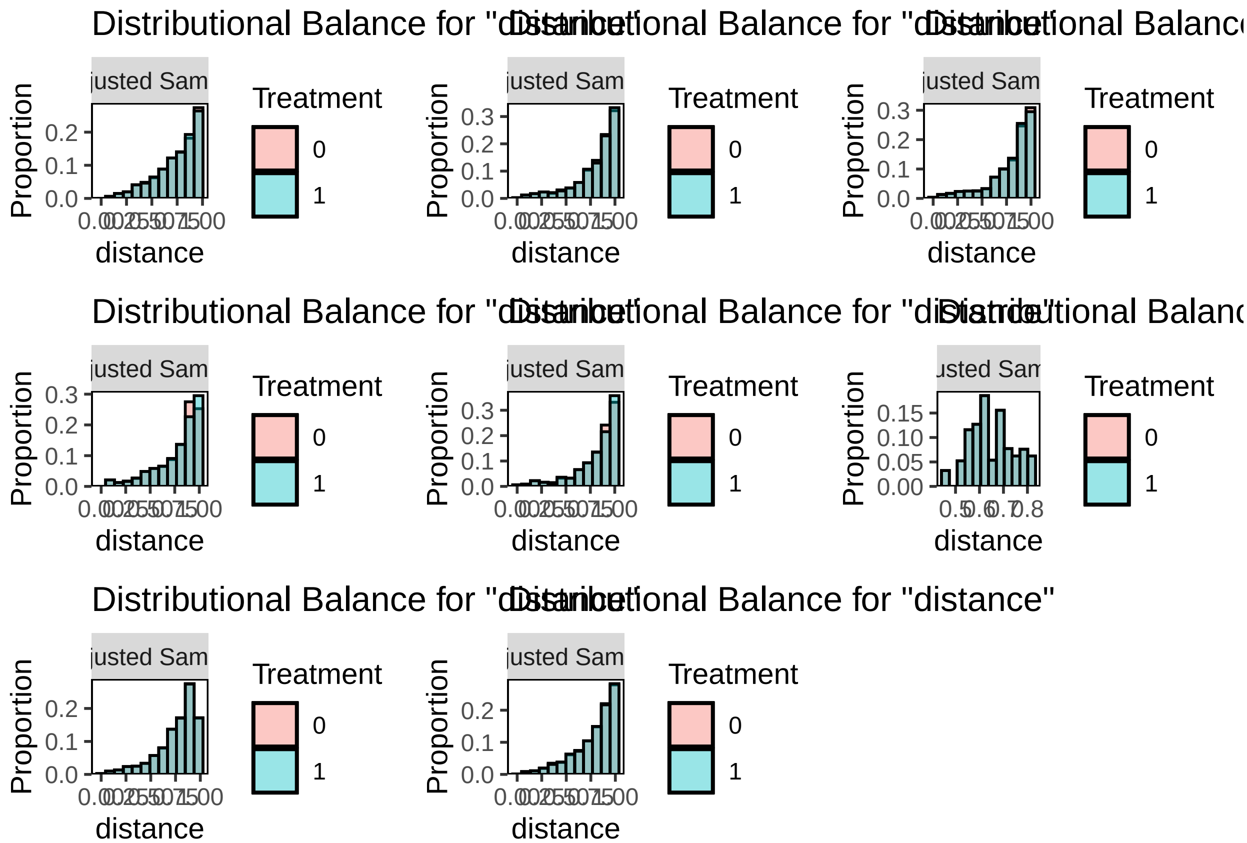
```r
meanprop <- mean(sim_results$prop_balanced, na.rm = TRUE)
sim_results %>% filter(prop_balanced > meanprop) %>% nrow()
```

```
## [1] 410
```

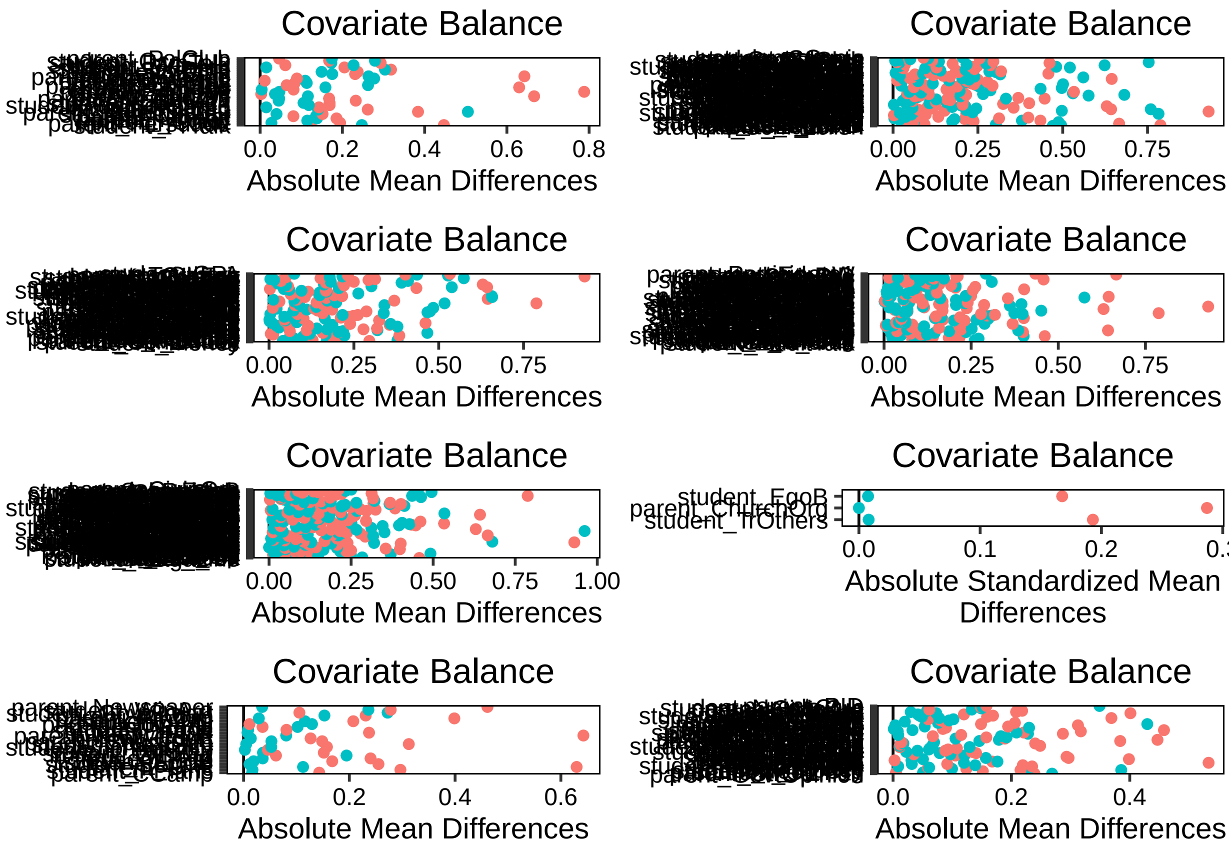
```r
# Plot ATT v.s. proportion
ggplot(sim_results, aes(x = prop_balanced, y = ATTs)) +
  geom_point() +
  labs(x = "Proportion of Features Balanced", y = "ATT") +
  theme_minimal()
```



```r
### 10 random covariate balance plots (hint try gridExtra)
set.seed(123)
bal_p <- list()
love_p <- list()

for (i in 1:10) {

  # Randomly select number of columns
  num_cols <- sample(3:ncol(ypsps_clean)-2, 1)
  colnames <- sample(names(ypsps_clean %>% select(-c(student_ppnscal, college))), num_cols)

  independent <- paste(colnames, collapse = " + ")
  ps_formula <- as.formula(paste("college ~", independent))


  # Fit p-score models and save ATTs
  model_ps <- matchit(replace = TRUE,
                      formula = ps_formula,
                      data = ypsps_clean,
                      method = "nearest",
                      ratio = 1,
                      estimand = "ATT")

  bal_p[[i]] <- bal.plot(model_ps, var.name = "distance", type = "histogram")
  love_p[[i]] <- love.plot(model_ps, drop.distance = TRUE, abs = TRUE, size = 2, position = "none")

}


# grid.arrange 1
p1 <- bal_p[[1]]; p2 <- bal_p[[2]]; p3 <- bal_p[[3]]; p4 <- bal_p[[4]]
p5 <- bal_p[[5]]; p6 <- bal_p[[6]]; p7 <- bal_p[[7]]; p8 <- bal_p[[8]]


options(repr.plot.width = 20, repr.plot.height = 25)
grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, ncol = 3)
```




```r
# grid.arrange 2
p1 <- love_p[[1]]; p2 <- love_p[[2]]; p3 <- love_p[[3]]; p4 <- love_p[[4]]
p5 <- love_p[[5]]; p6 <- love_p[[6]]; p7 <- love_p[[7]]; p8 <- love_p[[8]]


options(repr.plot.width = 20, repr.plot.height = 25)
grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, ncol = 2)
```



Questions

1. **How many simulations resulted in models with a higher proportion of balanced covariates? Do you have any concerns about this?** Your Answer: Out of the 1,000 simulations, 410 resulted in models with a higher proportion of balanced covariates compared to the average. While this represents approximately 43

1. **Analyze the distribution of the ATTs. Do you have any concerns about this distribution?** Your Answer: Based on the distribution of ATTs, it appears to be non-normally distributed, exhibiting a right skew. This skew indicates that there is more mass below the mean, suggesting that the treatment effects are being underestimated, particularly depending on the specification used. The concern arises from the potential bias introduced by the skewness of the distribution. Since the ATTs are skewed towards lower values, it implies that the treatment effects are systematically underestimated. To address this concern, exploring alternative modeling approaches or adjusting the analysis methodology to account for the skewness may be necessary to obtain more accurate treatment effect estimates and make more informed decisions based on the analysis results.

1. **Do your 10 randomly chosen covariate balance plots produce similar numbers on the same covariates? Is it a concern if they do not?** Your Answer: Based on the analysis of the ten randomly chosen covariate balance plots, it appears that they yield unsimilar metrics for the same covariates. This result raises concerns about the reliability and validity of treatment effect estimates. It suggests potential biases in the results due to sensitivity to model specification and matching algorithm.

Matching Algorithm of Your Choice

Simulate Alternative Model

Henderson/Chatfield propose using genetic matching to learn the best weights for Mahalanobis distance matching. Choose a matching algorithm other than the propensity score (you may use genetic matching if you wish, but it is also fine to use the greedy or optimal algorithms we covered in lab instead). Repeat the same steps as specified in Section 4.2 and answer the following questions:

```
# Remove post-treatment covariates
ATTs_r <- c()
prop_balanced_r <- c()
percent_imp_r <- c()


# Randomly select features
# Simulate random selection of features 1k+ times
for (i in 1:1000) {

  # Randomly select number of columns
  num_cols <- sample(3:ncol(ypsps_clean)-2, 1)
  colnames <- sample(names(ypsps_clean %>% select(-c(student_ppnscal, college))), num_cols)

  independent <- paste(colnames, collapse = " + ")
  ps_formula <- as.formula(paste("college ~", independent))


  # Fit p-score models and save ATTs
  model_ps_r <- matchit(formula = ps_formula,
                        data = ypsps_clean,
                        distance = "randomforest",
                        method = "nearest",
                        ratio = 1)

  match_att_data_r <- match.data(model_ps_r)

  STU_T1_r <- match_att_data_r %>% filter(college == 1) %>% summarise(mean(student_ppnscal)) %>% pull()
  STU_T0_r <- match_att_data_r %>% filter(college == 0) %>% summarise(mean(student_ppnscal)) %>% pull()
  ATTs_r[i] <- STU_T1_r - STU_T0_r


  summ_r <- bal.tab(model_ps_r, threshold = .1)

  A_r <- summ_r[[1]] %>% select(Diff.Adj) %>% slice(2:n()) %>% pull
  prop_balanced_r[i] <- sum(abs(A_r) <= 0.1) / length(A_r)

  pre_r <- data.frame(summary(model_ps_r)$sum.all) %>% pull(Std..Mean.Diff.) %>% mean ()
  post_r <- data.frame(summary(model_ps_r)$sum.matched) %>% pull(Std..Mean.Diff.) %>% mean ()

  percent_imp_r[i] <- (post_r - pre_r) / pre_r


}
```

```r
# Fit p-score models and save ATTs, proportion of balanced covariates, and mean percent balance improve
sim_results_r <- data.frame(ATTs_r, prop_balanced_r, percent_imp_r)
summary(sim_results_r)
```

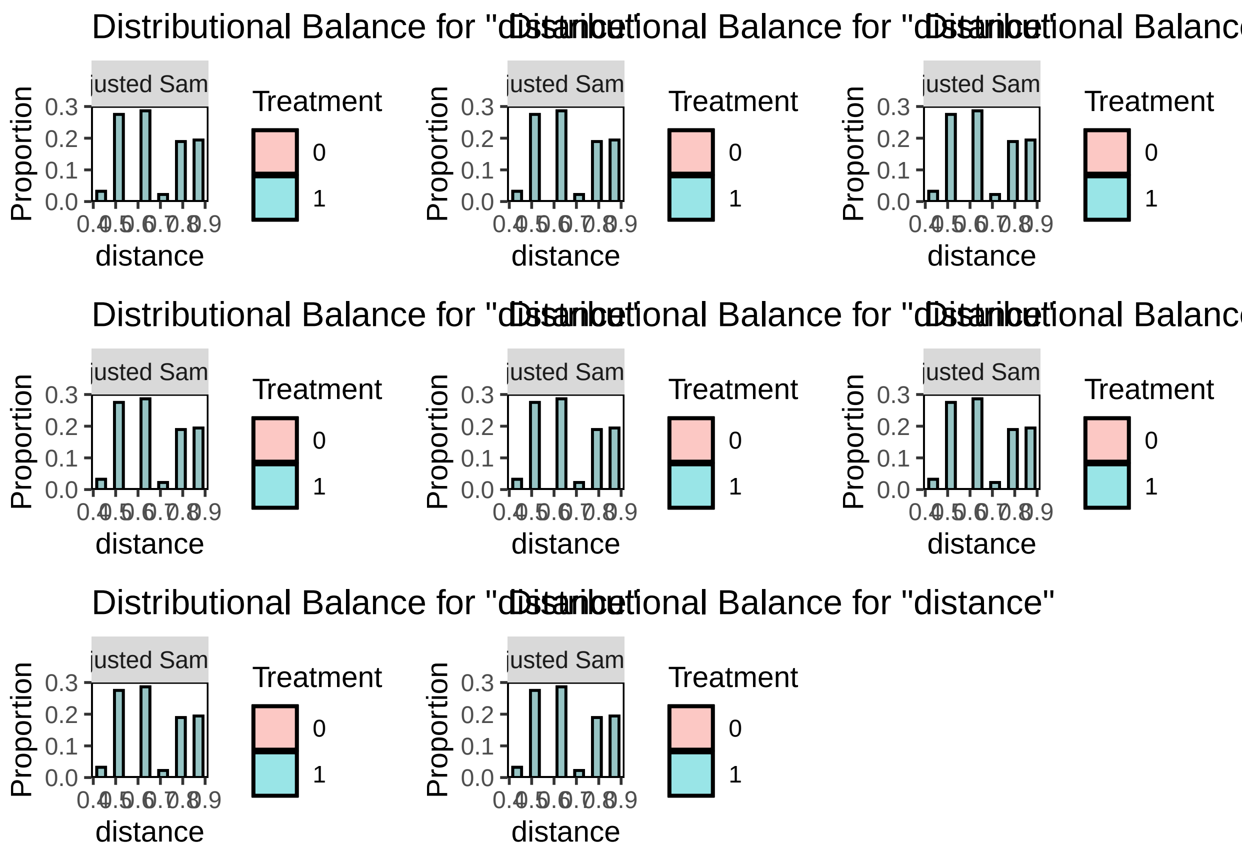
```
##      ATTs_r      prop_balanced_r  percent_imp_r
##  Min.   :1.220   Min.   :0.0000   Min.   :-3.3324
##  1st Qu.:1.796   1st Qu.:0.1525   1st Qu.: 0.4649
##  Median :1.891   Median :0.1810   Median : 0.4896
##  Mean   :1.846   Mean   :0.1788   Mean   : 0.5611
##  3rd Qu.:1.942   3rd Qu.:0.2037   3rd Qu.: 0.5623
##  Max.   :2.204   Max.   :1.0000   Max.   :16.1890
```

```r
meanprop_r <- mean(sim_results$prop_balanced_r, na.rm = TRUE)
sim_results %>% filter(prop_balanced_r > meanprop_r) %>% nrow()
```

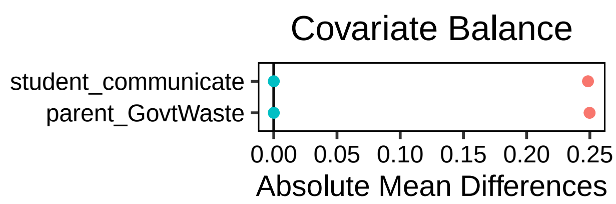
```
## [1] 0
```

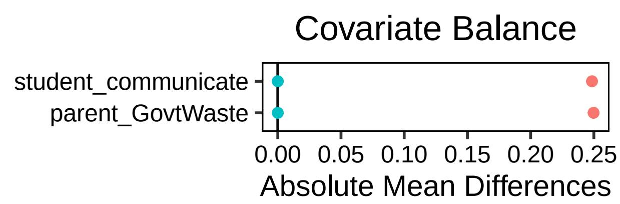
```r
# 10 random covariate balance plots (hint try gridExtra)
set.seed(123)
list <- list()
bal_pp <- list()
love_pp <- list()


for (i in 1:10) {

  # Randomly select number of columns
  num_cols <- sample(3:ncol(ypsps_clean)-2, 1)
  colnames <- sample(names(ypsps_clean %>% select(-c(student_ppnscal, college))), num_cols)

  independent <- paste(colnames, collapse = " + ")
  ps_formula <- as.formula(paste("college ~", independent))


  # Fit p-score models and save ATTs
  model_ps_r <- matchit(formula = ps_formula,
                    data = ypsps_clean,
                    distance = "randomforest",
                    method = "nearest",
                    ratio = 1)

  bal_pp[[i]] <- bal.plot(model_ps, var.name = "distance", type = "histogram")
  love_pp[[i]] <- love.plot(model_ps, drop.distance = TRUE, abs = TRUE, size = 2, position = "none")

}


# grid.arrange 1
p1 <- bal_pp[[1]]; p2 <- bal_pp[[2]]; p3 <- bal_pp[[3]]; p4 <- bal_pp[[4]]
p5 <- bal_pp[[5]]; p6 <- bal_pp[[6]]; p7 <- bal_pp[[7]]; p8 <- bal_pp[[8]]
```

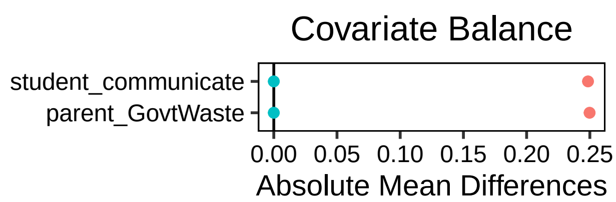
```r
options(repr.plot.width = 20, repr.plot.height = 25)
grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, ncol = 3)
```

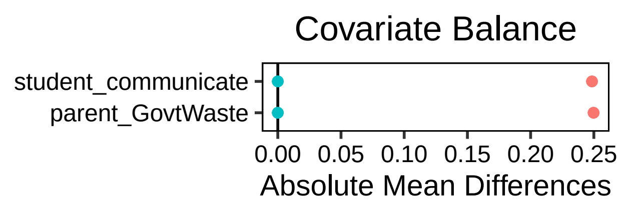


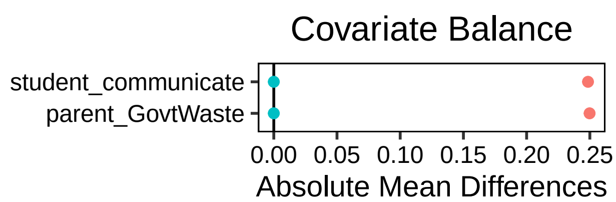

```r
# grid.arrange 2
p1 <- love_pp[[1]]; p2 <- love_pp[[2]]; p3 <- love_pp[[3]]; p4 <- love_pp[[4]]
p5 <- love_pp[[5]]; p6 <- love_pp[[6]]; p7 <- love_pp[[7]]; p8 <- love_pp[[8]]


options(repr.plot.width = 20, repr.plot.height = 25)
grid.arrange(p1, p2, p3, p4, p5, p6, p7, p8, ncol = 2)
```

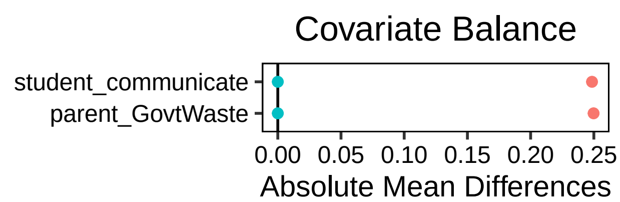


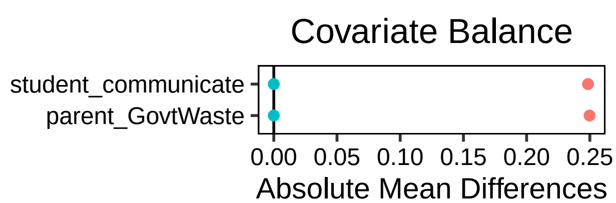


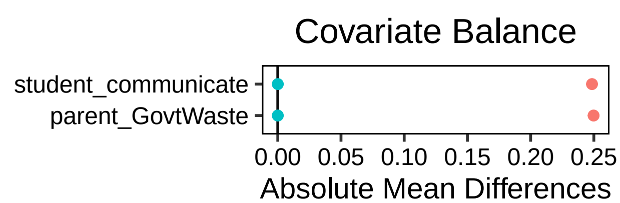






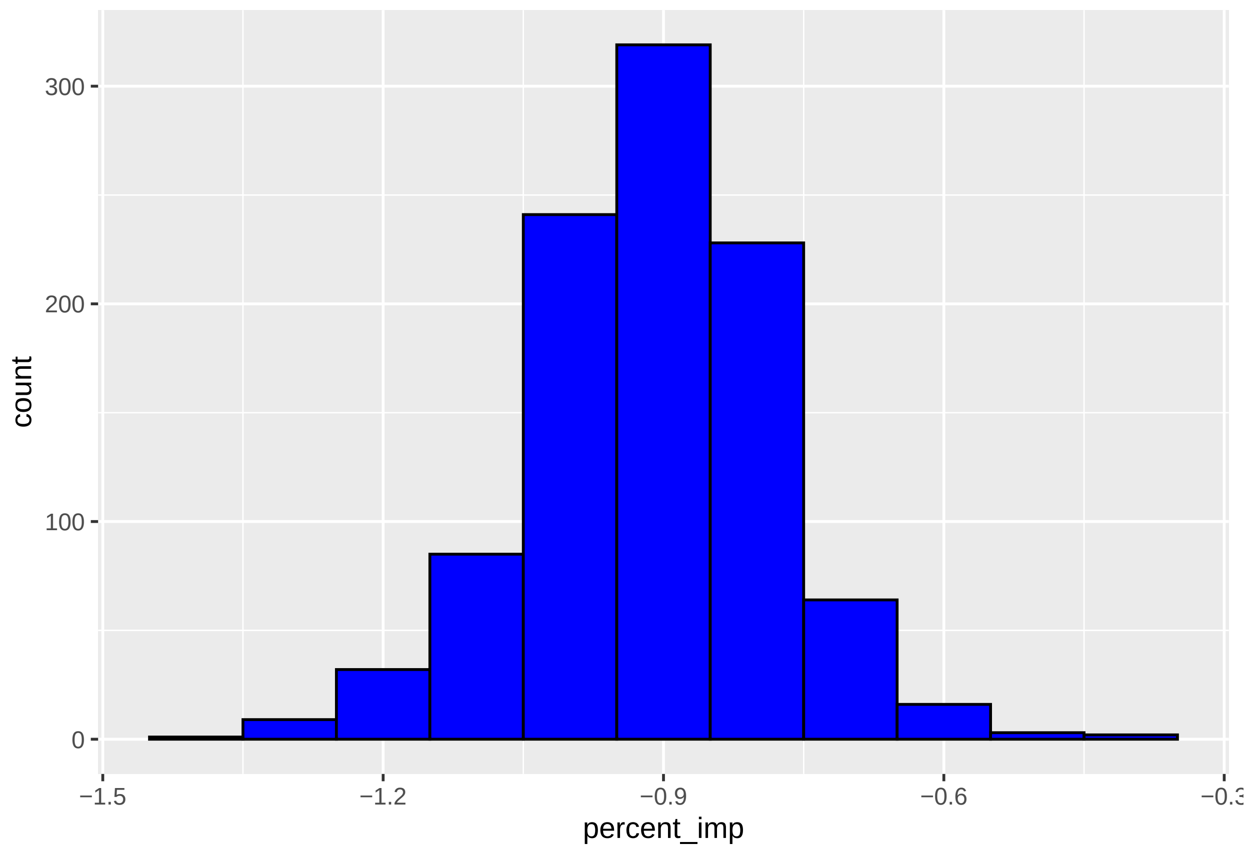




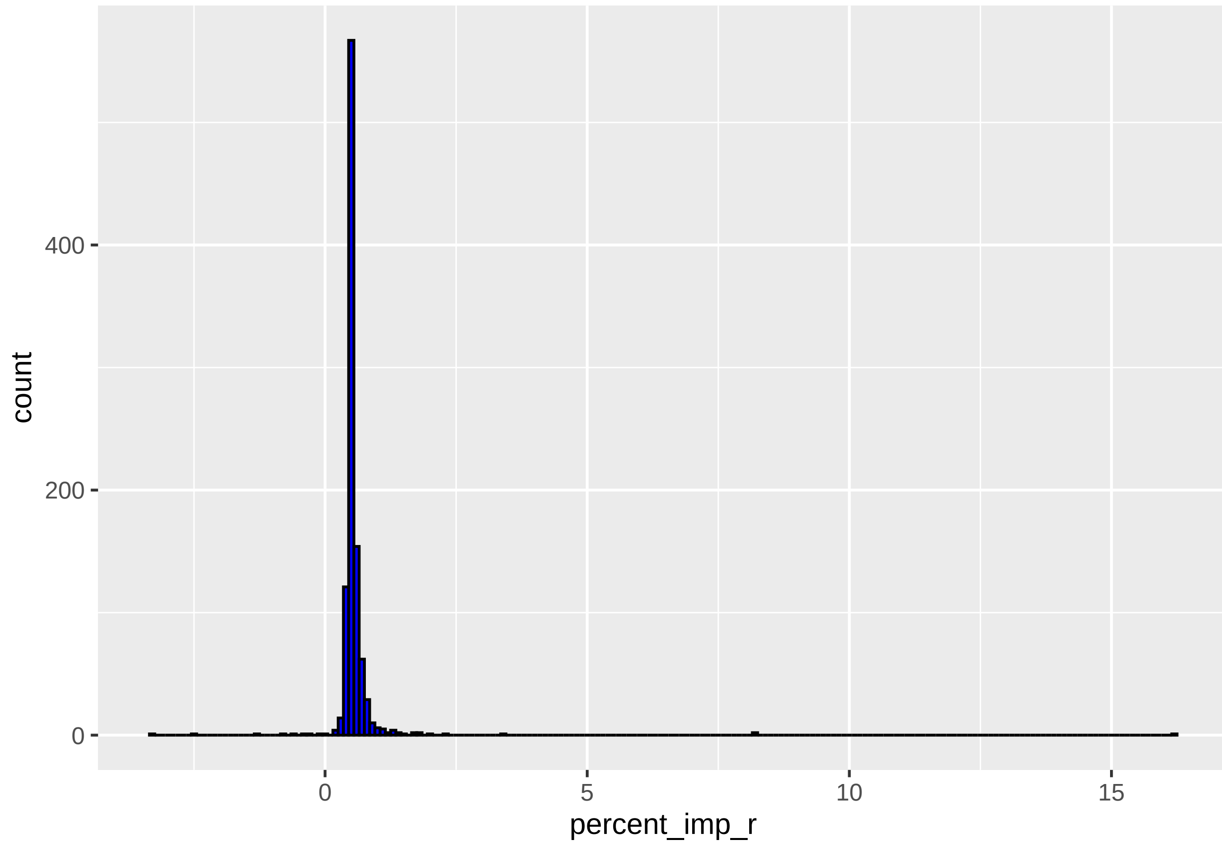



```
# Note: ggplot objects are finnicky so ask for help if you're struggling to automatically create them;
```

```
# Visualization for distributions of percent improvement

ggplot(sim_results, aes(x=percent_imp)) +
  geom_histogram(binwidth = 0.1, fill = "blue", color = "black")
```



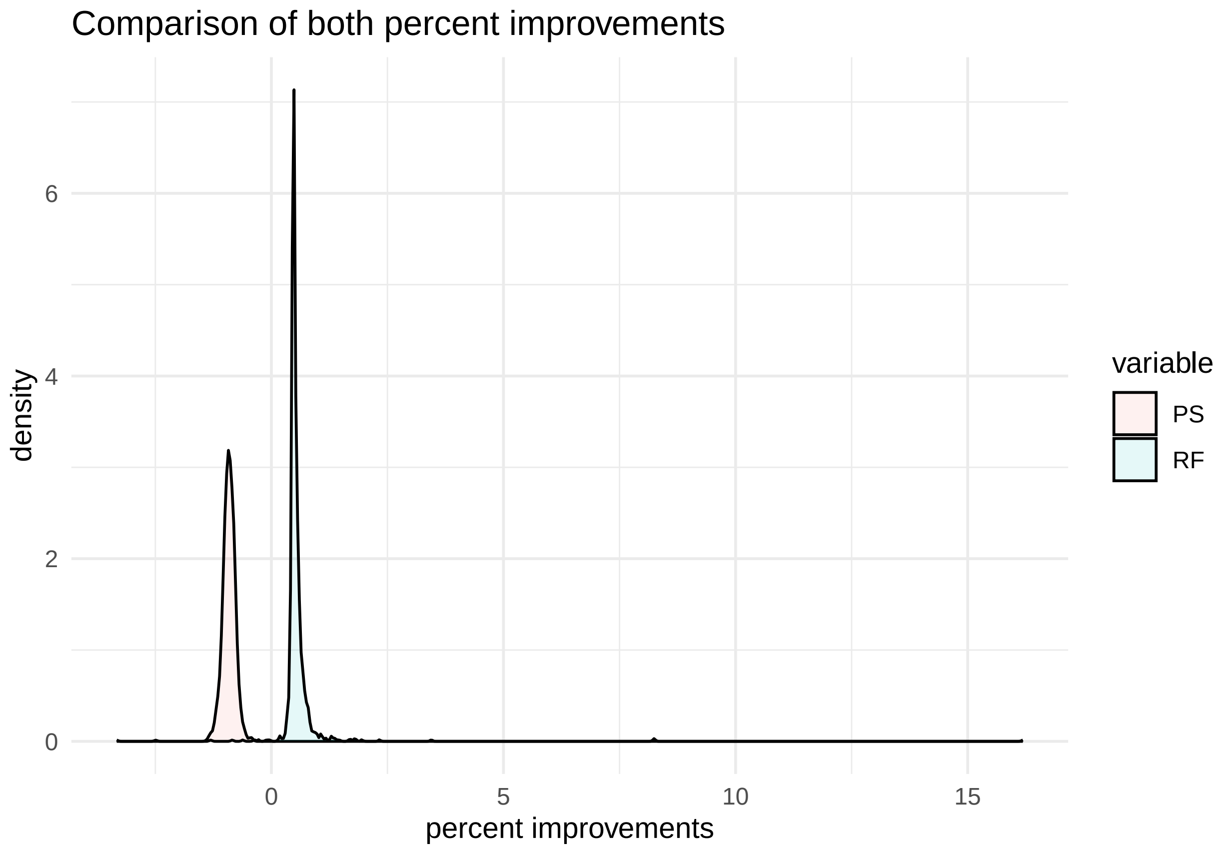

```r
ggplot(sim_results_r, aes(x=percent_imp_r)) +
  geom_histogram(binwidth = 0.1, fill = "blue", color = "black")
```




```
both_imp <- data.frame(PS = sim_results$percent_imp,
                       RF = sim_results_r$percent_imp_r)
long_data <- reshape2::melt(both_imp)
```

```
## No id variables; using all as measure variables
```

```
ggplot(long_data, aes(x = value, fill = variable)) +
  geom_density(alpha = 0.1) +
  labs(title = "Comparison of both percent improvements", x = "percent improvements") +
  theme_minimal()
```



Questions

1. **Does your alternative matching method have more runs with higher proportions of balanced covariates?** Your Answer: In comparison to the random forest method, which yielded no simulations with higher proportions of balanced covariates none out of 1,000, the propensity score matching method demonstrated superior performance with 410 simulations showing higher proportions. This suggests that the logistic regression-based propensity score matching method achieved better covariate balance improvements, potentially due to its robustness in smaller sample sizes and effectiveness in addressing covariate imbalance between treatment and control groups.

1. **Use a visualization to examine the change in the distribution of the percent improvement in balance in propensity score matching vs. the distribution of the percent improvement in balance in your new method. Which did better? Analyze the results in 1-2 sentences.** Your Answer: Based on the comparison of the distributions, while the original propensity score matching method had a higher proportion of simulations with highly balanced covariates, the random forest method showed no instances of highly balanced covariates.

Optional: Looking ahead to the discussion questions, you may choose to model the propensity score using an algorithm other than logistic regression and perform these simulations again, if you wish to explore the second discussion question further.

Discussion Questions

1. **Why might it be a good idea to do matching even if we have a randomized or as-if-random design?**

Your Answer: Even in randomized or as-if-random designs, conducting matching can offer several advantages. While randomization aims to evenly distribute covariates between treatment groups, it may not always achieve perfect balance, especially with small sample sizes or many covariates. Matching provides a complementary approach to create more comparable treatment and control groups based on observed covariates, thereby reducing bias and improving the precision of treatment effect estimates. Additionally, matching allows researchers to explore and visualize covariate balance, providing insights into the effectiveness of randomization and potential sources of bias. Overall, matching serves as a valuable tool for robustness checks and enhancing the validity of causal inference in both randomized and observational studies.

1. **The standard way of estimating the propensity score is using a logistic regression to estimate probability of treatment. Given what we know about the curse of dimensionality, do you think there might be advantages to using other machine learning algorithms (decision trees, bagging/boosting forests, ensembles, etc.) to estimate propensity scores instead?**

Your Answer: Although logistic regression is the conventional method for estimating propensity scores, alternative machine learning algorithms offer potential advantages, particularly in the context of high-dimensional data or complex relationships between covariates and treatment assignment. Algorithms such as decision trees, bagging/boosting forests, and ensembles can capture nonlinear relationships and interactions more effectively, potentially leading to more accurate propensity score estimates. Moreover, these algorithms are often more robust to model misspecification and can handle high-dimensional data without suffering from the curse of dimensionality.