Appendix: The Central Role of the Propensity Score in Observational Studies for Causal Effects Paper Review

Jai-Hua Kevin Yen

2025-10-27

Section 1.1 (Proof of Theorem 2)

($\Leftarrow$) To show $b(x)$ is a **balance score**, i.e., $x \perp z|b(x)$. It is sufficient to show $P(z = 1|b(x)) = e(x)$ since $b(x)$ is a function of x and knowing x automatically means you know $b(x)$. Then

$$\begin{aligned} P(z = 1|b(x), x) &= P(z = 1|x) \quad \because \text{by definition} \\ &= e(x) \end{aligned}$$

Here $P(z = 1|b(x), x) = e(x)$. The next thing is to prove $P(z = 1|b(x))$ is also equal $e(x)$ then $P(z = 1|b(x), x) = P(z = 1|b(x)) \quad (*)$

By **law of iterated expectation**,

$$\begin{aligned} P(z = 1|b(x)) &= E\left[P(z = 1|x)|b(x)\right] \quad \because \text{averaging the conditional probabilities over all possible values of } x \\ &= E\left[e(x)|b(x)\right] \quad \because \text{by definition} \\ &= e(x) \quad \because e(x) \text{ is just a function of } b(x) \end{aligned}$$

Thus, $(*)$ is satisfied and $b(x)$ is a **balance score**.

($\Rightarrow$) Assume $b(x)$ is a balance score but $b(x)$ is not finer than $e(x)$. But x and z will not be conditional independent given $b(x)$, then $b(x)$ is not a balance score. Thus, to be a balance score, $b(x)$ must be finer than $e(x)$.

Section 1.2 (Proof of Theorem 3)

(1) Prove $0 < P(z = 1|x) < 1$ for all $x \implies 0 < P(z = 1|b(x)) < 1$ for all $b(x)$

pf: From the proof of Theorem 2, we know that

$$P(z = 1|b(x)) = E\left[P(z = 1|x)|b(x)\right]$$

Also we know that $0 < E\left[P(z = 1|x)|b(x)\right] < 1$ since we are just **averaging** of a set of numbers that is strictly larger than 0 and less than 1.

(2.) Prove $(r_1, r_0) \perp z|x \implies (r_1, r_0) \perp z|b(x)$

pf: Aim: $P(z = 1|r_1, r_0, b(x)) = P(z = 1|b(x))$

Since we already know $P(z = 1|b(x)) = e(x)$, the rest is to show $P(z = 1|r_1, r_0, b(x)) = e(x)$.

By **law of iterated expectation**,

$$\begin{aligned} P(z = 1|r_1, r_0, b(x)) &= E\left[P(z = 1|r_1, r_0, x)|r_1, r_0, b(x)\right] \\ &= E\left[P(z = 1|x)|r_1, r_0, b(x)\right] \quad \because \text{given condition } (r_1, r_0) \perp z|x \\ &= E\left[e(x)|r_1, r_0, b(x)\right] \\ &= e(x) \quad \because b(x) \text{ is a balancing score (implies } b(x) \text{ is finer than } e(x)) \end{aligned}$$

Thus, the proof is complete.

Section 1.3 (Proof of Corollary 6.1)

(1.)

$$\begin{aligned} B - B_m &= \int E(x|b)\left[P_n(b|z = 0) - P(b|z = 0)\right] db \quad \because \text{ from Theorem 6} \\ &= \int (\alpha + \beta f(x))\left[P_n(b|z = a) - P(b|z = a)\right] db \quad \because \text{ from assumption} \\ &= \beta \int f(x)\left[P_n(b|z = a) - P(b|z = a)\right] db \quad \because \text{ integrated each probability density would be 0} \end{aligned}$$

$$\implies (B - B_m)_i = \beta_i\left[E_m(f(b)|z = a) - E(f(b)|z = a)\right]$$

(2.)

$$\begin{aligned} B &= E(x|z = 1) - E(x|z = 0) \\ &= E_b(E(x|b)|z = 1) - E_b(E(x|b)|z = 0) \\ &= E_b\left(\alpha + \beta f(b)|z = 1)\right) - E_b\left(\alpha + \beta f(b)|z = 0)\right) \quad \because \text{ from assumption} \\ &= \beta\left[E(f(b)|z = 1) - E(f(b)|z = 0)\right] \end{aligned}$$

$$\implies B_i = \beta_i\left[E(f(b)|z = 1) - E(f(b)|z = 0)\right]$$

Thus,

$$\text{percent reduction} = 100 \times \frac{(B - B_m)_i}{B_i}.$$

Section 2.1 (Example Using R: Lalonde 1986 Dataset)

The simulation will use the lalonde dataset, a famous dataset in the causal inference literature. It was originally compiled by Robert Lalonde (1986) to evaluate the performance of non-experimental methods. The goal is to estimate the effect of the job training program (treat=1) on real earnings in 1978 (re78). The dataset combines two groups:

- Treatment Group: Individuals from the National Supported Work (NSW) Demonstration, an experimental job training program conducted in the mid-1970s.
- Control Group: A non-equivalent comparison group constructed from the Panel Survey of Income Dynamics (PSID), a large-scale public survey.

The covariates include age, education (educ), race, marital status (married), high school degree status (nodegree), and real earnings in 1974 and 1975 (re74, re75).

```
library(MASS)
library(MatchIt)
library(cobalt)
```

```
##  cobalt (Version 4.6.1, Build Date: 2025-08-20)
```

```
##
## Attaching package: 'cobalt'
```

```
## The following object is masked from 'package:MatchIt':
##
##     lalonde
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(knitr)
library(tidyr)
```

```
data("lalonde", package = "MatchIt")
head(lalonde)
```

```
##      treat age educ   race married nodegree re74 re75       re78
## NSW1     1  37   11  black       1        1    0    0  9930.0460
## NSW2     1  22    9 hispan       0        1    0    0  3595.8940
## NSW3     1  30   12  black       0        0    0    0 24909.4500
## NSW4     1  27   11  black       0        1    0    0  7506.1460
## NSW5     1  33    8  black       0        1    0    0   289.7899
## NSW6     1  22    9  black       0        1    0    0  4056.4940
```

```r
tail(lalonde)
```

```
##         treat age educ   race married nodegree re74 re75       re78
## PSID424     0  25   14  white       0        0    0    0     0.0000
## PSID425     0  18   11  white       0        1    0    0 10150.5000
## PSID426     0  24    1 hispan       1        1    0    0 19464.6100
## PSID427     0  21   18  white       0        0    0    0     0.0000
## PSID428     0  32    5  black       1        1    0    0   187.6713
## PSID429     0  16    9  white       0        1    0    0  1495.4590
```

```r
unmatched_balance = matchit(treat ~ age + educ + race + married
                            + nodegree + re74 + re75,
                            data = lalonde, method = NULL,
                            distance = "glm")

m.out = matchit(treat ~ age + educ + race + married + nodegree + re74 + re75,
                data = lalonde,
                method = "nearest",
                caliper = 0.2,
                distance = "glm", link = "logit",
                replace = FALSE)

summary(m.out)
```

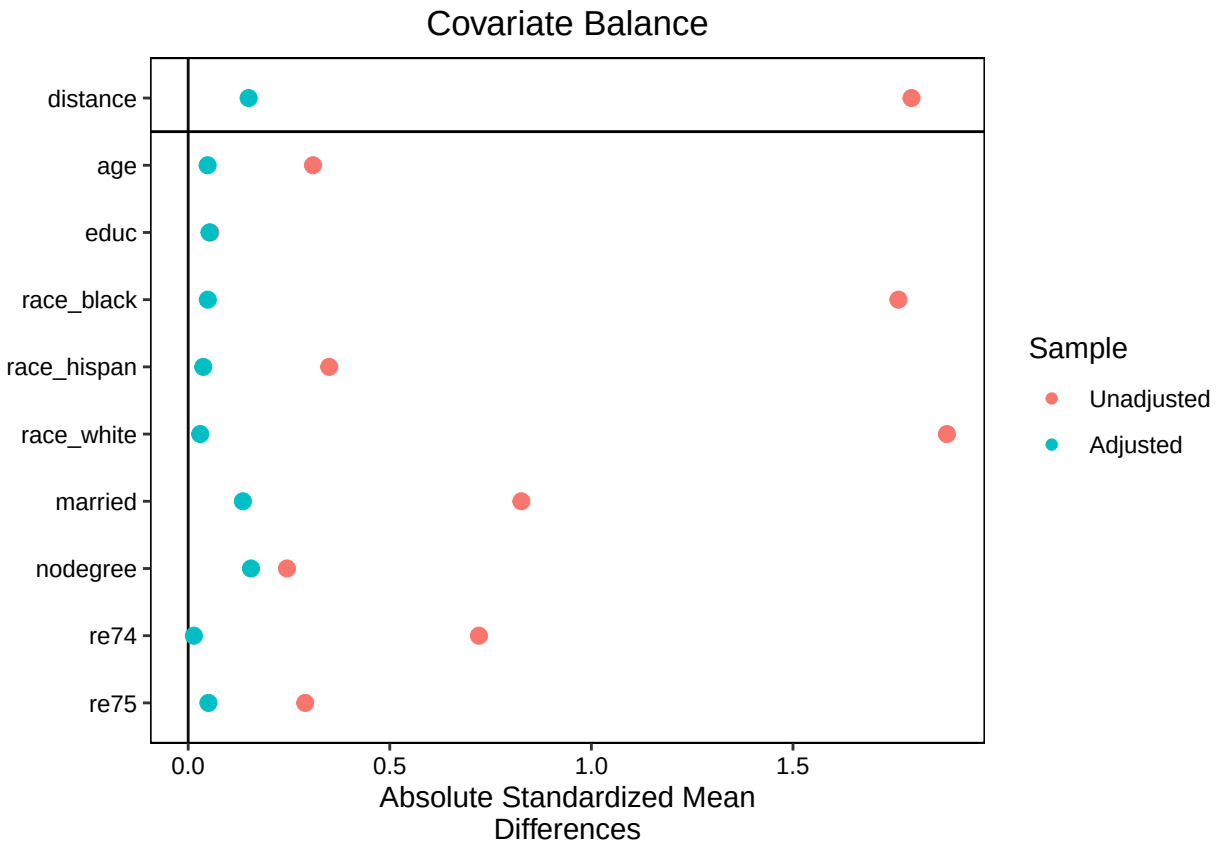
```
##
## Call:
## matchit(formula = treat ~ age + educ + race + married + nodegree +
##     re74 + re75, data = lalonde, method = "nearest", distance = "glm",
##     link = "logit", replace = FALSE, caliper = 0.2)
##
## Summary of Balance for All Data:
##            Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## distance          0.5774        0.1822          1.7941     0.9211    0.3774
## age              25.8162       28.0303         -0.3094     0.4400    0.0813
## educ             10.3459       10.2354          0.0550     0.4959    0.0347
## raceblack         0.8432        0.2028          1.7615          .    0.6404
## racehispan        0.0595        0.1422         -0.3498          .    0.0827
## racewhite         0.0973        0.6550         -1.8819          .    0.5577
## married           0.1892        0.5128         -0.8263          .    0.3236
## nodegree          0.7081        0.5967          0.2450          .    0.1114
## re74           2095.5737     5619.2365         -0.7211     0.5181    0.2248
## re75           1532.0553     2466.4844         -0.2903     0.9563    0.1342
##            eCDF Max
## distance     0.6444
## age          0.1577
## educ         0.1114
## raceblack    0.6404
## racehispan   0.0827
## racewhite    0.5577
## married      0.3236
## nodegree     0.1114
## re74         0.4470
## re75         0.2876
```

```
##
## Summary of Balance for Matched Data:
##            Means Treated Means Control Std. Mean Diff. Var. Ratio eCDF Mean
## distance          0.5250        0.4920          0.1499     1.1838    0.0387
## age              26.7611       26.4159          0.0482     0.3791    0.0993
## educ             10.5752       10.4690          0.0528     0.5390    0.0298
## raceblack         0.7434        0.7257          0.0487          .    0.0177
## racehispan        0.0973        0.1062         -0.0374          .    0.0088
## racewhite         0.1593        0.1681         -0.0299          .    0.0088
## married           0.2035        0.2566         -0.1356          .    0.0531
## nodegree          0.6903        0.6195          0.1557          .    0.0708
## re74           2531.9595     2601.0395         -0.0141     1.4808    0.0457
## re75           1720.8840     1882.7539         -0.0503     1.4043    0.0507
##            eCDF Max Std. Pair Dist.
## distance     0.2389          0.1517
## age          0.3274          1.4088
## educ         0.0885          1.2412
## raceblack    0.0177          0.0974
## racehispan   0.0088          0.4865
## racewhite    0.0088          0.3285
## married      0.0531          0.6327
## nodegree     0.0708          0.9733
## re74         0.2655          0.6766
## re75         0.1593          0.8468
##
## Sample Sizes:
##           Control Treated
## All           429     185
## Matched       113     113
## Unmatched     316      72
## Discarded       0       0
```

The summary of unmatched dataset reveal substantial differences in the means of the covariates between the treated and control groups, confirming that a naive comparison of their 1978 earnings would be severely biased.

We will use 1:1 nearest neighbor matching on the logit of the propensity score. This method pairs each treated unit with the available control unit that has the closest propensity score.

```r
# Create a new dataframe containing only the matched units
matched_data = match.data(m.out)

# Create a Love plot to visualize covariate balance
love.plot(m.out,
          binary = "std",
          abs = TRUE)
```



Now we can try to fit regression in both unmatched and match dataset:

```
# Fit a linear model on the matched data
fit_matched = lm(re78 ~ treat, data = matched_data)
summary(fit_matched)
```

```
##
## Call:
## lm(formula = re78 ~ treat, data = matched_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -6510  -4938  -2483   2964  53798
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4938.0      694.3   7.112 1.52e-11 ***
## treat         1571.7      981.9   1.601    0.111
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7380 on 224 degrees of freedom
## Multiple R-squared:  0.01131,    Adjusted R-squared:  0.006896
## F-statistic: 2.562 on 1 and 224 DF,  p-value: 0.1108
```

```r
fit_unmatched = lm(re78 ~ treat, data = lalonde)
summary(fit_unmatched)
```

```
##
## Call:
## lm(formula = re78 ~ treat, data = lalonde)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -6984  -6349  -2048   4100  53959
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6984.2      360.7  19.362   <2e-16 ***
## treat         -635.0      657.1  -0.966    0.334
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7471 on 612 degrees of freedom
## Multiple R-squared:  0.001524,   Adjusted R-squared:  -0.0001079
## F-statistic: 0.9338 on 1 and 612 DF,  p-value: 0.3342
```

The result shows that we may have a biased conclusion without matching.

Since the propensity score matching method is based on pairing, so we can also try using paired t-test to test this result:

```r
treated_units <- rownames(m.out$match.matrix)
control_units <- m.out$match.matrix[, 1] %>% as.vector()

outcome_treated <- lalonde[treated_units, "re78"]
outcome_control <- lalonde[control_units, "re78"]

paired_ttest_result <- t.test(outcome_treated, outcome_control,
                              paired = TRUE)

print(paired_ttest_result)
```

```
##
##  Paired t-test
##
## data:  outcome_treated and outcome_control
## t = 1.5582, df = 112, p-value = 0.122
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -426.8631 3570.3521
## sample estimates:
## mean difference
##        1571.744
```

Section 2.2 (Simulation Result of Table 1.)

Here is the simulation for Table 1

```r
library(MASS)
library(dplyr)
library(knitr)
library(tidyr)

set.seed(1983)
n_treated = 50
reservoir_ratios = c(2, 3, 4)
initial_biases = c(0.25, 0.50, 0.75, 1.00)
n_simulations = 500
pop_cor = 0.5

find_nearest_neighbors = function(treated_scores, control_scores, treated_match_order) {
  matched_control_indices = integer(length(treated_scores))
  available_controls = 1:length(control_scores)

  for (i in treated_match_order) {
    distances = abs(treated_scores[i] - control_scores[available_controls])
    best_local_index = which.min(distances)
    best_global_index = available_controls[best_local_index]
    matched_control_indices[i] = best_global_index
    available_controls = available_controls[-best_local_index]
  }
  return(matched_control_indices)
}

run_propensity_comparison_simulation = function(target_bias, R, pop_cor) {

  n_control_reservoir = n_treated * R

  # Data Generation
  pop_cov_matrix = matrix(c(1, pop_cor, pop_cor, 1), nrow = 2)
  pop_cov_inv = solve(pop_cov_matrix)
  k = target_bias * sqrt(1 - pop_cor^2)
  mu_treated = c(k, 0)
  mu_control = c(0, 0)

  treated_data = mvrnorm(n = n_treated, mu = mu_treated, Sigma = pop_cov_matrix)
  control_reservoir_data = mvrnorm(n = n_control_reservoir, mu = mu_control, Sigma = pop_cov_matrix)

  # Combine data for easier calculations
  full_data = rbind(treated_data, control_reservoir_data)

  # Population Propensity Scores (using true parameters)
  beta_1 = pop_cov_inv %*% (mu_treated - mu_control)
  beta_0 = -0.5 * t(mu_treated - mu_control) %*% beta_1
  log_odds = as.vector(full_data %*% beta_1 + as.numeric(beta_0))
  pop_ps = 1 / (1 + exp(-log_odds))

  pop_ps_treated = pop_ps[1:n_treated]
  pop_ps_control = pop_ps[(n_treated + 1):length(pop_ps)]

  # Sample Propensity Scores (estimating from data)
```

```
  combined_df = as.data.frame(full_data)
  names(combined_df) = c("X1", "X2")
  combined_df$treat = c(rep(1, n_treated), rep(0, n_control_reservoir))

  logit_model = glm(treat ~ X1 + X2, data = combined_df, family = binomial(link = "logit"))
  sample_ps = predict(logit_model, type = "response")
  sample_ps_treated = sample_ps[1:n_treated]
  sample_ps_control = sample_ps[(n_treated + 1):length(sample_ps)]

  # Matching
  match_order = sample(1:n_treated)
  matched_indices_pop = find_nearest_neighbors(pop_ps_treated, pop_ps_control, match_order)
  matched_indices_sample = find_nearest_neighbors(sample_ps_treated, sample_ps_control, match_order)

  # Evaluate Bias Reduction on the Stable Population Discriminant Scale
  pop_discriminant_vec = pop_cov_inv %*% (mu_treated - mu_control)
  pop_scores_treated = as.vector(treated_data %*% pop_discriminant_vec)
  pop_scores_control = as.vector(control_reservoir_data %*% pop_discriminant_vec)
  pop_score_sd = target_bias
  initial_bias_standardized = 1.0

  # Reduction for population score matching
  final_bias_pop = mean(pop_scores_treated) - mean(pop_scores_control[matched_indices_pop])
  reduction_pop = 100 * (1 - abs(final_bias_pop / pop_score_sd) / initial_bias_standardized)

  # Reduction for sample score matching
  final_bias_sample = mean(pop_scores_treated) - mean(pop_scores_control[matched_indices_sample])
  reduction_sample = 100 * (1 - abs(final_bias_sample / pop_score_sd) / initial_bias_standardized)

  return(c(Population = reduction_pop, Sample = reduction_sample))
}

all_results = list()
i = 1

for (r_ratio in reservoir_ratios) {
  for (bias in initial_biases) {
    sim_output = replicate(n_simulations,
                           run_propensity_comparison_simulation(bias, r_ratio, pop_cor))
    avg_reductions = rowMeans(sim_output)
    all_results[[i]] = data.frame(
      Ratio = r_ratio,
      Initial_Bias = bias,
      Population_Reduction = avg_reductions["Population"],
      Sample_Reduction = avg_reductions["Sample"]
    )
    i = i + 1
  }
}

final_results_df = do.call(rbind, all_results)

table1_replication = final_results_df %>%
```

```
  pivot_longer(
    cols = c(Population_Reduction, Sample_Reduction),
    names_to = "Type_of_score",
    values_to = "Reduction"
  ) %>%
  mutate(Type_of_score = sub("_Reduction", "", Type_of_score)) %>%
  pivot_wider(
    names_from = Initial_Bias,
    values_from = Reduction
  ) %>%
  arrange(Ratio, desc(Type_of_score))

print(kable(table1_replication,
            digits = 2,
            col.names = c("Ratio", "Type of Score", "0.25", "0.50", "0.75", "1.00"),
            caption = "Percent Reduction in Bias (Replication of Rosenbaum & Rubin, 1983, Table 1)"))
```

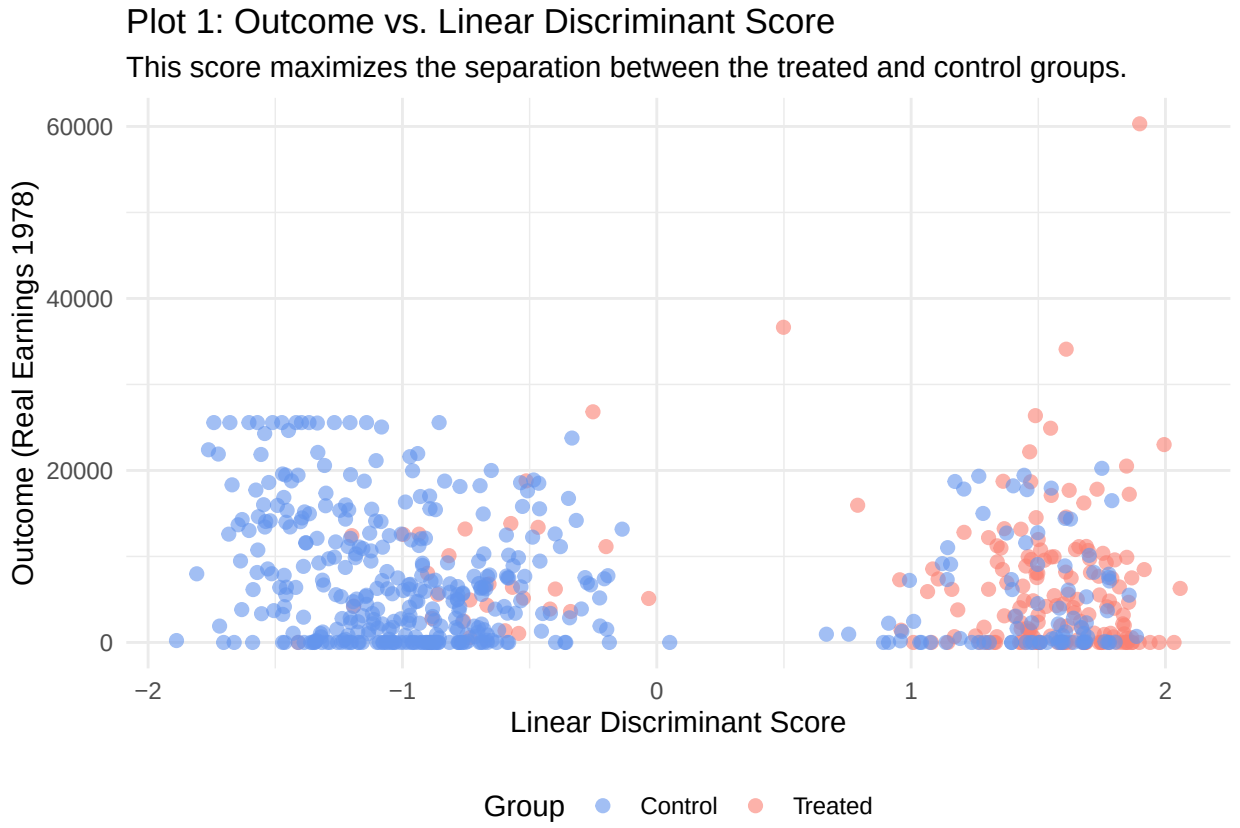
```
##
##
## Table: Percent Reduction in Bias (Replication of Rosenbaum & Rubin, 1983, Table 1)
##
## | Ratio|Type of Score |  0.25|  0.50|  0.75|  1.00|
## |-----:|:-------------|-----:|-----:|-----:|-----:|
## |     2|Sample        | 94.96| 91.98| 83.30| 67.01|
## |     2|Population    | 97.42| 93.37| 84.01| 67.80|
## |     3|Sample        | 94.90| 95.76| 92.01| 82.05|
## |     3|Population    | 98.81| 97.17| 92.78| 82.70|
## |     4|Sample        | 95.15| 96.50| 94.34| 88.14|
## |     4|Population    | 99.13| 98.04| 95.06| 88.70|
```

Section 2.3 Propensity Score and Covariance Adjustment

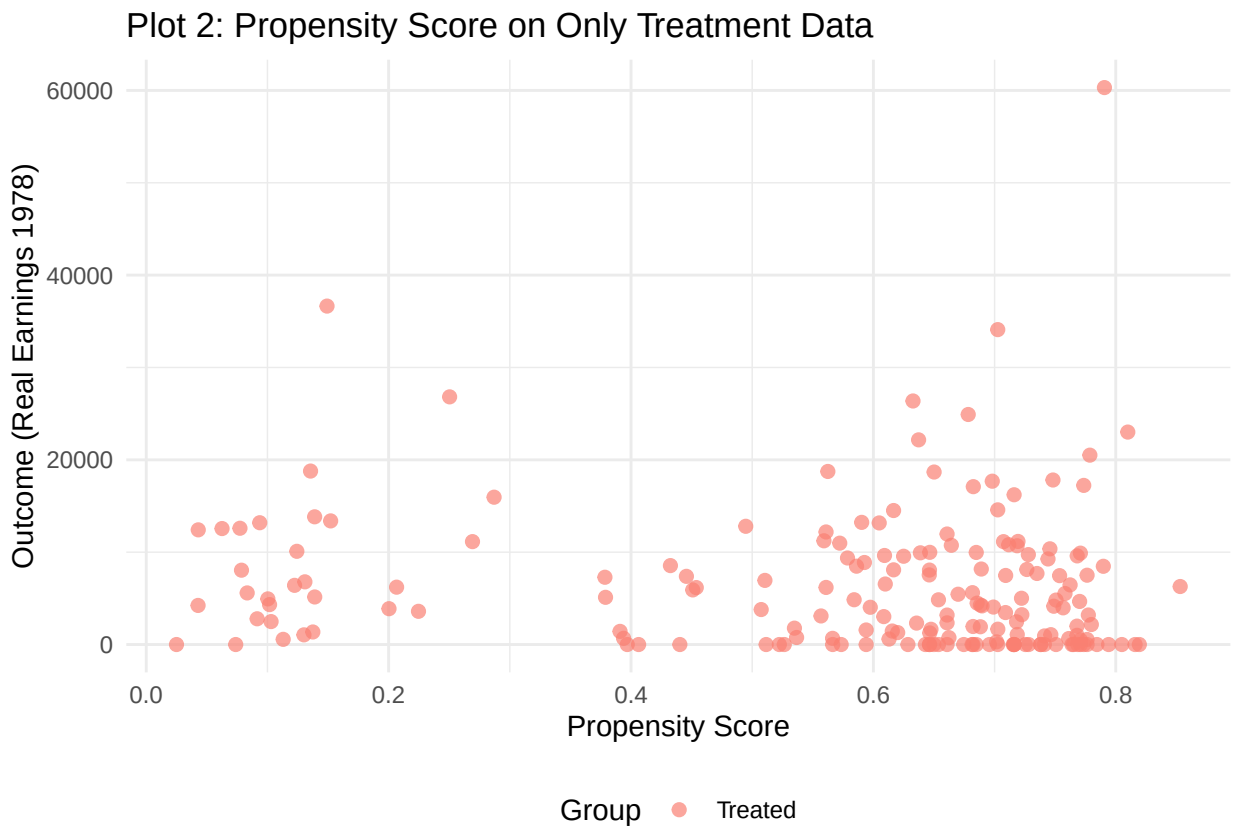
```
library(MatchIt)
library(MASS)
library(ggplot2)

data("lalonde", package = "MatchIt")

ps_model <- matchit(treat ~ age + educ + race + married + nodegree + re74 + re75,
                    data = lalonde, method = NULL,
                    distance = "glm")

lda_model <- lda(treat ~ age + educ + race + married + nodegree + re74 + re75,
                 data = lalonde)
lda_scores <- predict(lda_model, newdata = lalonde)$x[,1]

plot_data <- data.frame(
  re78 = lalonde$re78,
  treat = lalonde$treat,
  ps = ps_model$distance,  # Propensity Scores
```

```r
  lda = lda_scores          # Linear Discriminant Scores
)

plot1 <- ggplot(plot_data, aes(x = lda, y = re78, color = factor(treat))) +
  geom_point(alpha = 0.6, size = 2) +
  scale_color_manual(values = c("0" = "cornflowerblue", "1" = "salmon"),
                     labels = c("Control", "Treated")) +
  labs(
    title = "Plot 1: Outcome vs. Linear Discriminant Score",
    subtitle = "This score maximizes the separation between the treated and control groups.",
    x = "Linear Discriminant Score",
    y = "Outcome (Real Earnings 1978)",
    color = "Group"
  ) +
  theme_minimal() +
  theme(legend.position = "bottom")

plot2 <- ggplot(plot_data[plot_data$treat == 1,], aes(x = ps, y = re78, color = factor(treat))) +
  geom_point(alpha = 0.7, size = 2) +
  scale_color_manual(values = c("1" = "salmon"),
                     labels = c("Treated")) +
  labs(
    title = "Plot 2: Propensity Score on Only Treatment Data",
    x = "Propensity Score",
    y = "Outcome (Real Earnings 1978)",
    color = "Group"
  ) +
  theme_minimal() +
  theme(legend.position = "bottom")

plot3 <- ggplot(plot_data, aes(x = ps, y = re78, color = factor(treat))) +
  geom_point(alpha = 0.7, size = 2) +
  scale_color_manual(values = c("0" = "cornflowerblue", "1" = "salmon"),
                     labels = c("Control", "Treated")) +
  labs(
    title = "Plot 3: Outcome vs. Propensity Score",
    subtitle = "This score shows the region of overlap (common support) for valid comparison.",
    x = "Propensity Score",
    y = "Outcome (Real Earnings 1978)",
    color = "Group"
  ) +
  theme_minimal() +
  theme(legend.position = "bottom")
```

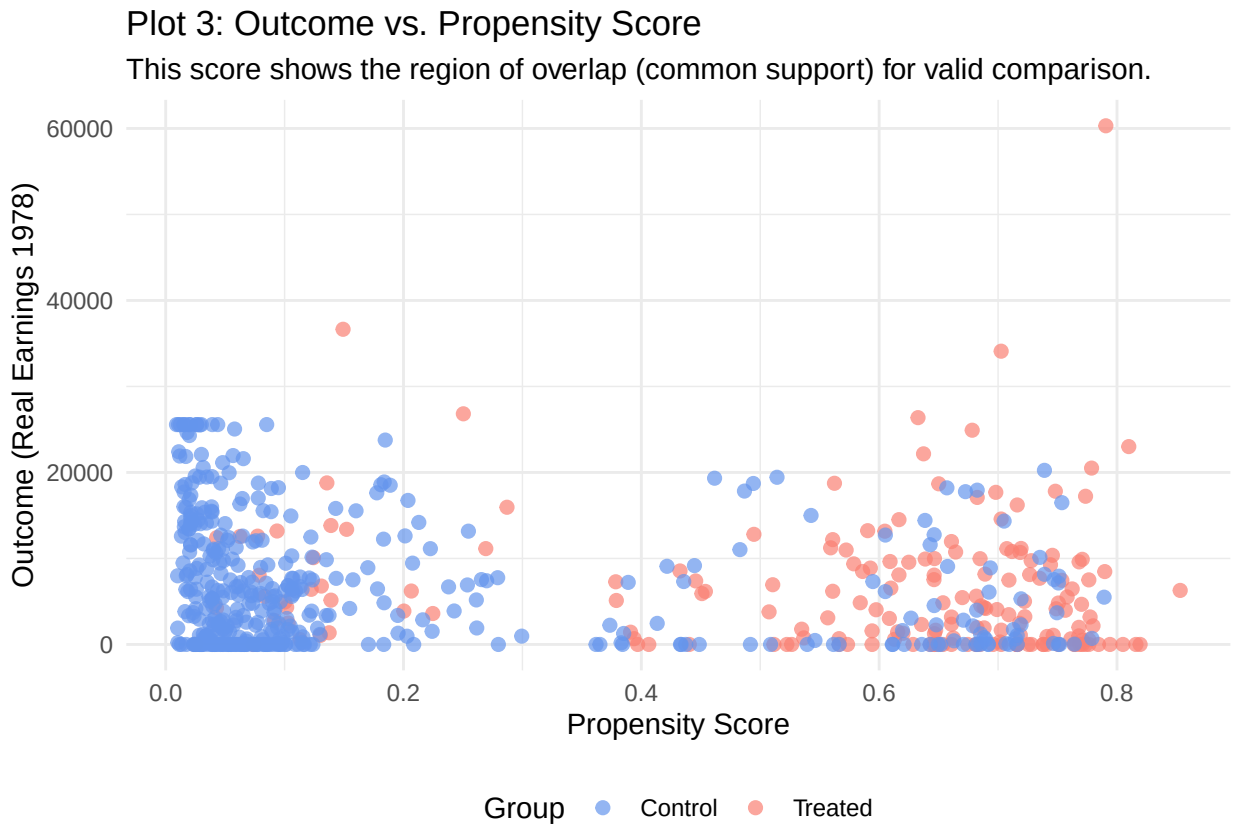
```r
print(plot1)
```


Plot 1: Outcome vs. Linear Discriminant Score
This score maximizes the separation between the treated and control groups.
Outcome (Real Earnings 1978)
60000
40000
20000
0
−2
−1
0
1
2
Linear Discriminant Score
Group
Control
Treated

```
print(plot2)
```


Plot 2: Propensity Score on Only Treatment Data
Outcome (Real Earnings 1978)
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
Propensity Score
Group
Treated

```
print(plot3)
```


Plot 3: Outcome vs. Propensity Score
This score shows the region of overlap (common support) for valid comparison.
Outcome (Real Earnings 1978)
60000
40000
20000
0
0.0
0.2
0.4
0.6
0.8
Propensity Score
Group
Control
Treated