

# Bayesian Analysis & Propensity Score Analysis

## Computational Mathematics and Statistics

Jason Bryer, Ph.D.

December 2, 2025

# Bayesian Analysis



# Bayesian Analysis

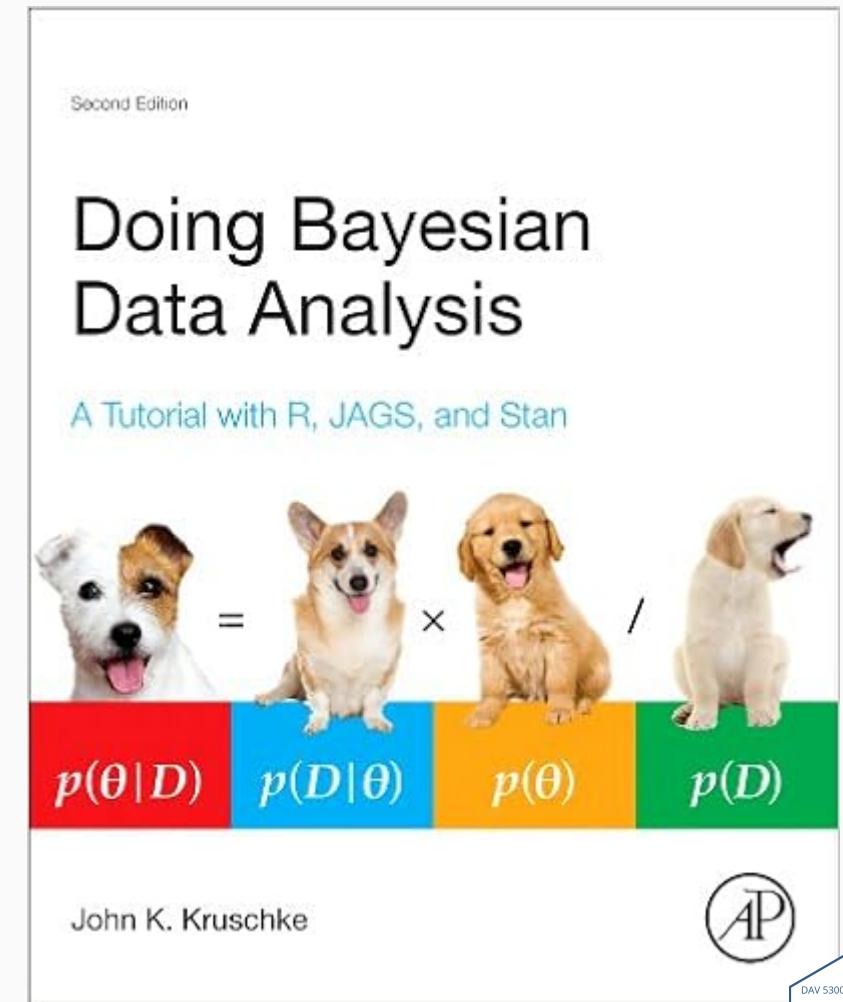
Kruschke's videos are an excellent introduction to Bayesian Analysis [https://www.youtube.com/watch?v=YohWpj6KU!](https://www.youtube.com/watch?v=YohWpj6KU)

Doing Bayesian Data Analysis, Second Edition: A Tutorial with R, JAGS, and Stan

*The Theory That Would Not Die: How Bayes' Rule Cracked the Enigma Code, Hunted Down Russian Submarines, and Emerged Triumphant from Two Centuries of Controversy* by Sharon Bertsch McGrayne

Video series by Rasmus Baath [Part 1](#), [Part 2](#), [Part 3](#)

Billiards with Fred the Frequentist and Bayer the Bayesian



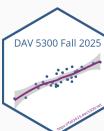
# Bayes Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|A')P(A')}$$

Consider the following data from a cancer test:

- 1% of women have breast cancer (and therefore 99% do not).
- 80% of mammograms detect breast cancer when it is there (and therefore 20% miss it).
- 9.6% of mammograms detect breast cancer when it's not there (and therefore 90.4% correctly return a negative result).

	<b>Cancer (1%)</b>	<b>No Cancer (99%)</b>
Test positive	80%	9.6%
Test negative	20%	90.4%



# How accurate is the test?

Now suppose you get a positive test result. What are the chances you have cancer?  
80%? 99%? 1%?

- Ok, we got a positive result. It means we're somewhere in the top row of our table. Let's not assume anything - it could be a true positive or a false positive.
- The chances of a true positive = chance you have cancer *chance test caught it* = 1% 80% = .008
- The chances of a false positive = chance you don't have cancer *chance test caught it anyway* = 99% 9.6% = 0.09504

	<b>Cancer (1%)</b>	<b>No Cancer (99%)</b>	
Test positive	True +: 1% * 80%	False +: 99% * 9.6%	<b>10.304%</b>
Test negative	False -: 1% * 20%	True -: 99% * 90.4%	<b>89.696%</b>



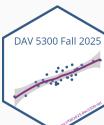
# How accurate is the test?

$$\text{Probability} = \frac{\text{desired event}}{\text{all possibilities}}$$

The chance of getting a real, positive result is .008. The chance of getting any type of positive result is the chance of a true positive plus the chance of a false positive ( $.008 + 0.09504 = .10304$ ).

$$P(C|P) = \frac{P(P|C)P(C)}{P(P)} = \frac{.8 * .01}{.008 + 0.095} \approx .078$$

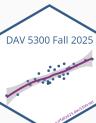
So, our chance of cancer is  $.008/.10304 = 0.0776$ , or about 7.8%.



# Bayes Formula

It all comes down to the chance of a true positive result divided by the chance of any positive result. We can simplify the equation to:

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$



# Bayes THEOREM

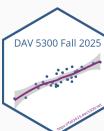
$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

POSTERIOR

LIKELIHOOD      PRIOR

MARGINAL LIKELIHOOD

BY CHRIS ALBON



# How many fish are in the lake?

- Catch them all, count them. Not practical (or even possible)!
- We can sample some fish.

Our strategy:

1. Catch some fish.
2. Mark them.
3. Return the fish to the pond. Let them get mixed up (i.e. wait a while).
4. Catch some more fish.
5. Count how many are marked.

For example, we initially caught 20 fish, marked them, returned them to the pond. We then caught another 20 fish and 5 of them were marked (i.e they were caught the first time).

Adopted from Rasmuth Bääth useR! 2015 workshop: [http://www.sumsar.net/files/academia/user\\_2015\\_tutorial\\_bayesian\\_data\\_analysis\\_short\\_version.pdf](http://www.sumsar.net/files/academia/user_2015_tutorial_bayesian_data_analysis_short_version.pdf)



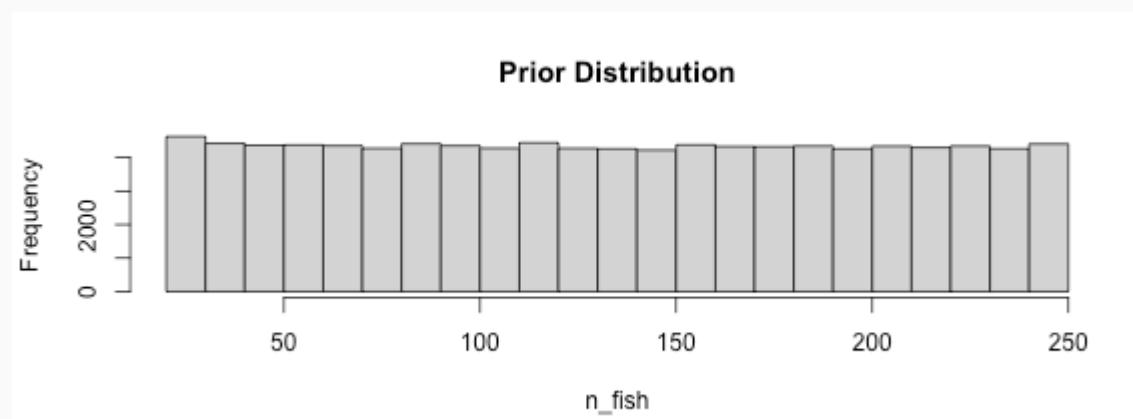
# Strategy for fitting a model

Step 1: Define Prior Distribution. Draw a lot of random samples from the "prior" probability distribution on the parameters.

```
n_draw <- 100000  
n_fish <- sample(20:250, n_draw, replace = TRUE)  
head(n_fish, n=10)
```

```
## [1] 207 95 111 75 212 53 245 158 77 213
```

```
hist(n_fish, main="Prior Distribution")
```



# Strategy for fitting a model

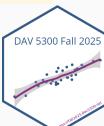
Step 2: Plug in each draw into the generative model which generates "fake" data.

```
pick_fish <- function(n_fish) { # The generative model
  fish <- rep(0:1, c(n_fish - 20, 20))
  sum(sample(fish, 20))
}

n_marked <- rep(NA, n_draw)
for(i in 1:n_draw) {
  n_marked[i] <- pick_fish(n_fish[i])
}
```

```
cbind(caught_twice = head(n_marked, n=10),
      pop_est = head(n_fish, n=10)
)
```

##	caught_twice	pop_est
## [1,]	1	207
## [2,]	6	95
## [3,]	4	111
## [4,]	5	75
## [5,]	2	212
## [6,]	6	53
## [7,]	0	245
## [8,]	3	158
## [9,]	6	77
## [10,]	1	213



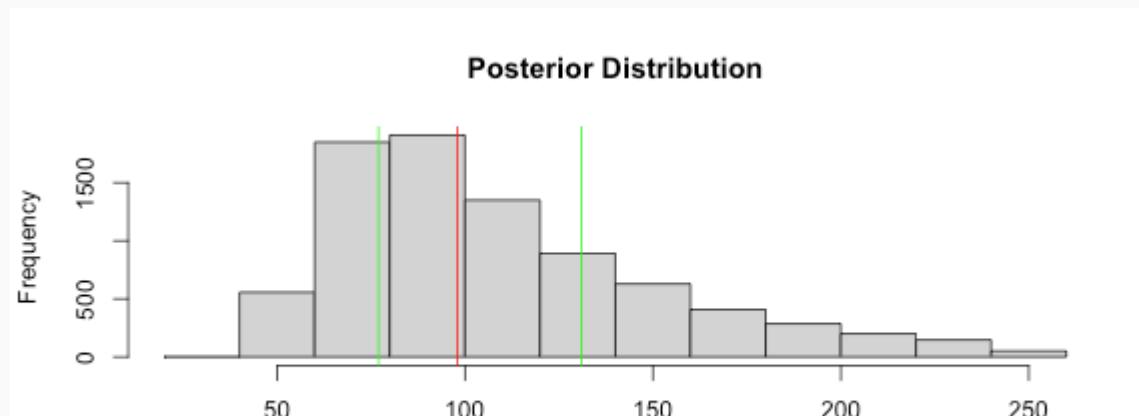
# Strategy for fitting a model

Step 3: Keep only those parameter values that generated the data that was actually observed (in this case, 5).

```
post_fish <- n_fish[n_marked == 5]  
length(post_fish)
```

```
## [1] 8279
```

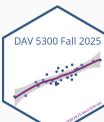
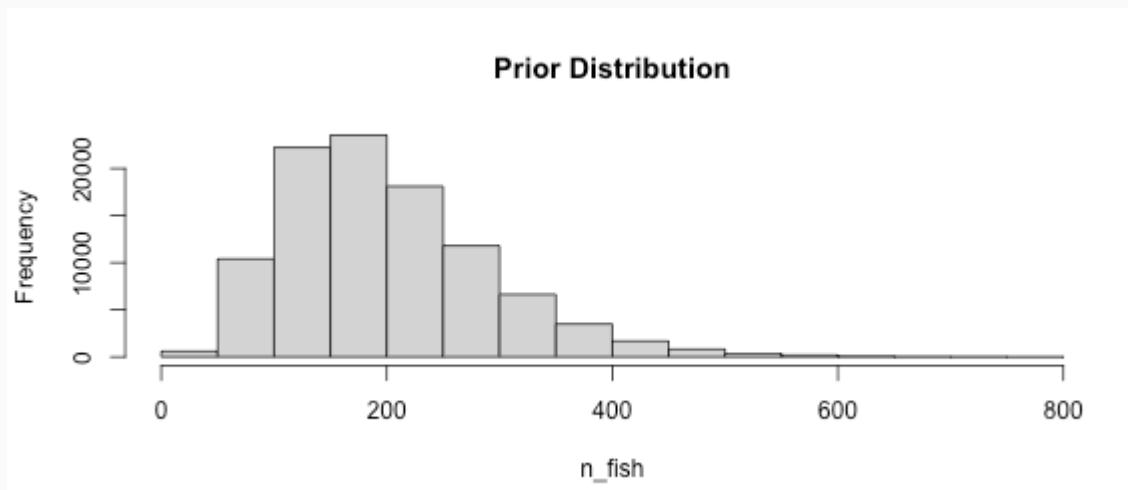
```
hist(post_fish, main='Posterior Distribution')  
abline(v=median(post_fish), col='red')  
abline(v=quantile(post_fish, probs=c(.25, .75)), col='green')
```



# What if we have better prior information?

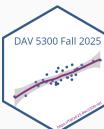
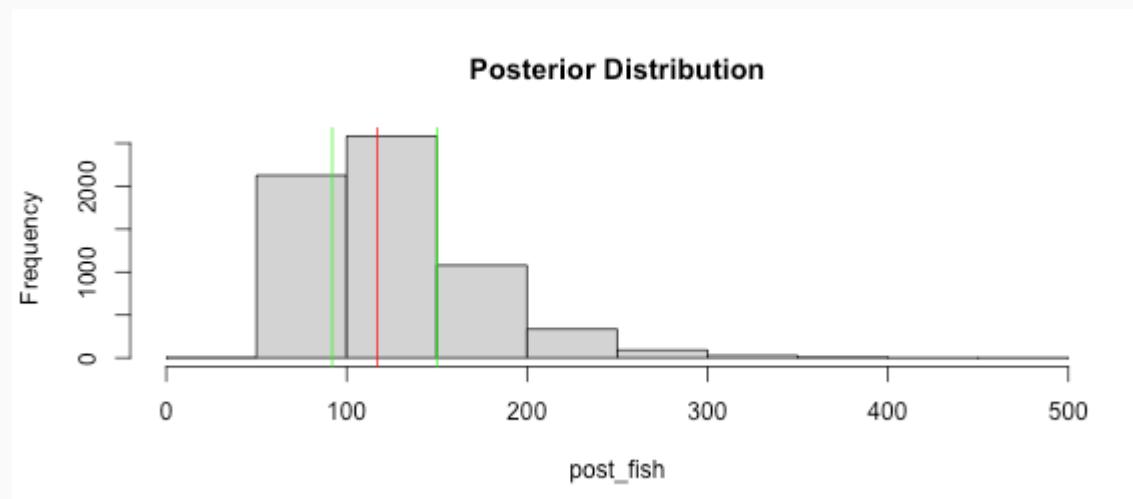
An "expert" believes there are around 200 fish in the pond. Instead of a uniform distribution, we can use a binomial distribution to define our "prior" distribution.

```
n_fish <- rnbinom(n_draw, mu = 200 - 20, size = 4) + 20  
hist(n_fish, main='Prior Distribution')
```



# What if we have better prior information?

```
n_marked <- rep(NA, n_draw)
for(i in 1:n_draw) {
  n_marked[i] <- pick_fish(n_fish[i])
}
post_fish <- n_fish[n_marked == 5]
hist(post_fish, main='Posterior Distribution')
abline(v=median(post_fish), col='red')
abline(v=quantile(post_fish, probs=c(.25, .75)), col='green')
```

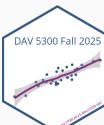


# Bayes Billiards Balls

Consider a pool table of length one. An 8-ball is thrown such that the likelihood of its stopping point is uniform across the entire table (i.e. the table is perfectly level). The location of the 8-ball is recorded, but not known to the observer. Subsequent balls are thrown one at a time and all that is reported is whether the ball stopped to the left or right of the 8-ball. Given only this information, what is the position of the 8-ball? How does the estimate change as more balls are thrown and recorded?

```
DATA606::shiny_demo('BayesBilliards', package='DATA606')
```

See also: [http://www.bryer.org/post/2016-02-21-bayes\\_billiards\\_shiny/](http://www.bryer.org/post/2016-02-21-bayes_billiards_shiny/)



# Bayesian Regression

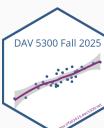
Returning to the `study` example from the logistic regression lecture, we can use the `stan_glm` function from the `rstanarm` R package.

## Bayesian GLM

```
stan_out <- rstanarm::stan_glm(  
  Pass ~ Hours,  
  data = study,  
  family = binomial(link = "logit"),  
  seed = 2112,  
  refresh = 0)
```

## Frequentist GLM

```
glm_out <- glm(  
  Pass ~ Hours,  
  data = study,  
  family = binomial(link = 'logit')  
)
```



# Bayesian Regression (cont.)

## Bayesian GLM

```
summary(stan_out, pars = c("alpha", "beta"))

## 
## Model Info:
##   function: stan_glm
##   family: binomial [logit]
##   formula: Pass ~ Hours
##   algorithm: sampling
##   sample: 4000 (posterior sample size)
##   priors: see help('prior_summary')
##   observations: 20
##   predictors: 2

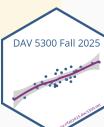
## 
## Estimates:
##           mean    sd   10%   50%   90%
## (Intercept) -4.3   1.6 -6.4 -4.1 -2.3
## Hours       1.6   0.6  0.9  1.5  2.3
## 
## MCMC diagnostics
##           mcse Rhat n_eff
## (Intercept) 0.0  1.0 2405
## Hours       0.0  1.0 2317
## 

## For each parameter, mcse is Monte Carlo standard error, n_eff is
```

## Frequentist GLM

```
summary(glm_out)

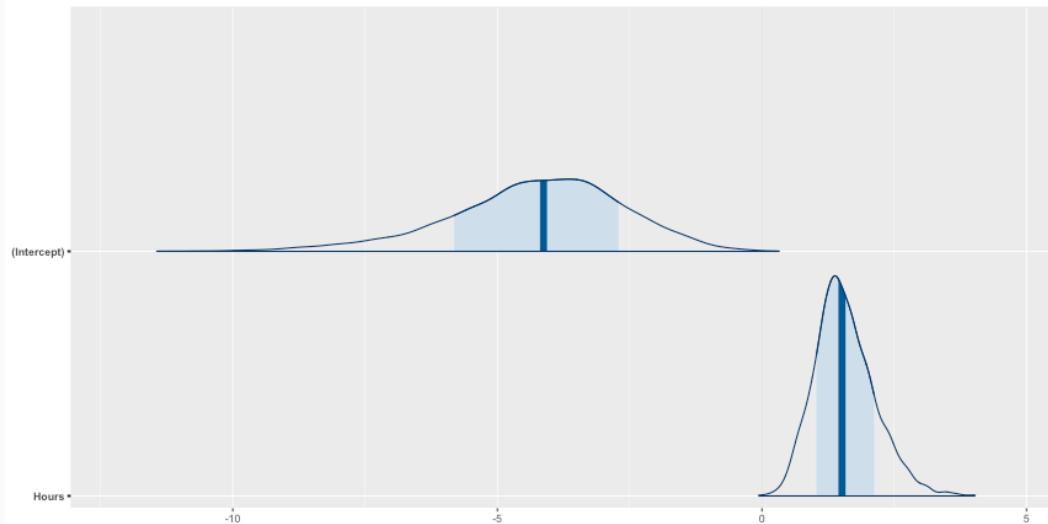
## 
## Call:
## glm(formula = Pass ~ Hours, family = binomial(link = "logit"),
##      data = study)
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.0777    1.7610 -2.316   0.0206 *
## Hours        1.5046    0.6287  2.393   0.0167 *
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
## Null deviance: 27.726 on 19 degrees of freedom
## Residual deviance: 16.060 on 18 degrees of freedom
## AIC: 20.06
## 
## Number of Fisher Scoring iterations: 5
```



# Plotting output

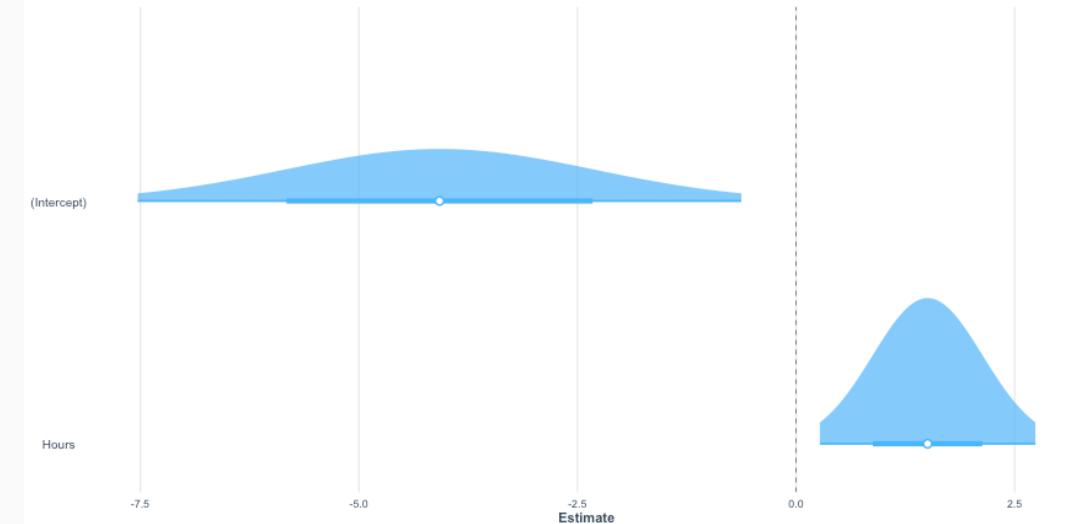
## Bayesian GLM

```
plot(stan_out,  
      plotfun = 'areas',  
      prob = 0.68)
```



## Frequentist GLM

```
jtools::plot_summs(glm_out,  
                    plot.distributions = TRUE,  
                    inner_ci_level = .68, omit.coefs = 'Intercept')
```



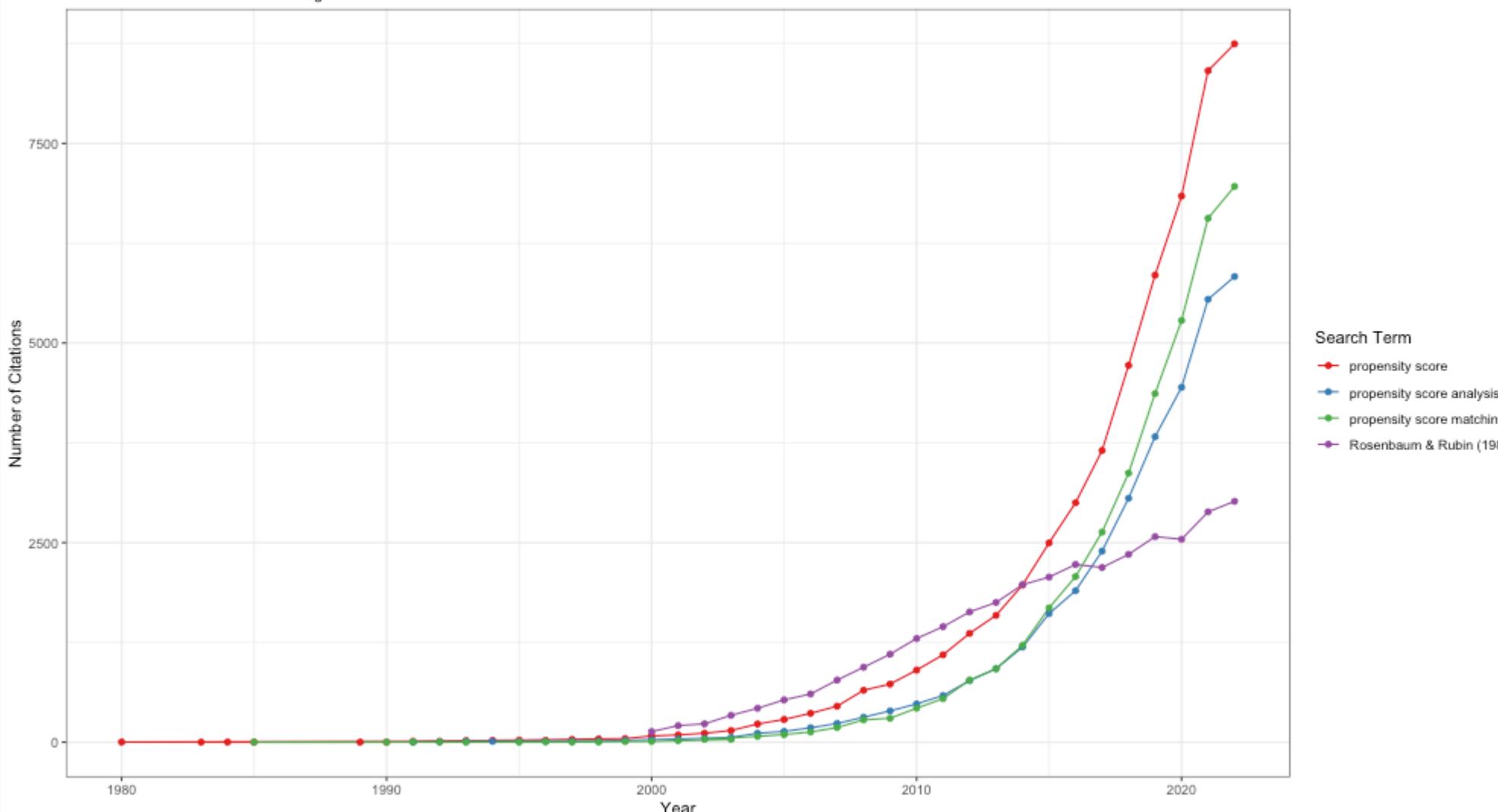
# Propensity Score Analysis



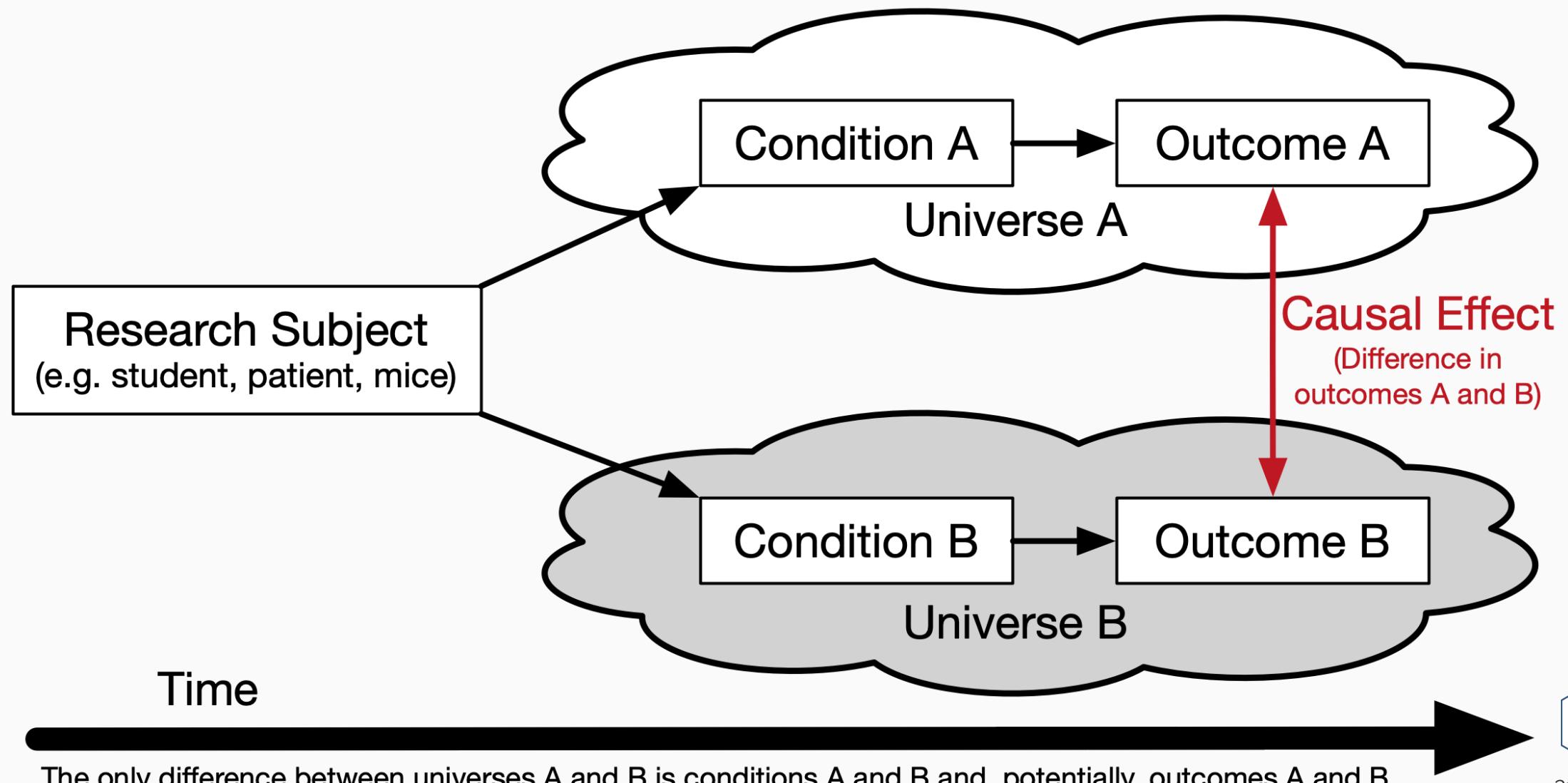
# Popularity of Propensity Score Analysis

Number of Citations for Propensity Score Analysis

Source: Web of Science and Google Scholar



# Counterfactuals



# The Randomized Experiment

Considered to be the *gold standard* for estimating causal effects.

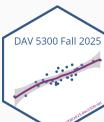
- Effects can be estimated using simple means between groups, or blocks in randomized block design.
- Randomization presumes unbiasedness and balance between groups.

However, randomization is often not feasible for many reasons, especially in educational contexts.

The **strong ignorability assumption** states that:

$$(Y_i(1), Y_i(0)) \perp\!\!\!\perp T_i | X_i = x$$

for all  $X_i$ .

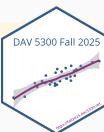


# RCT Example

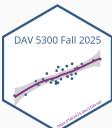
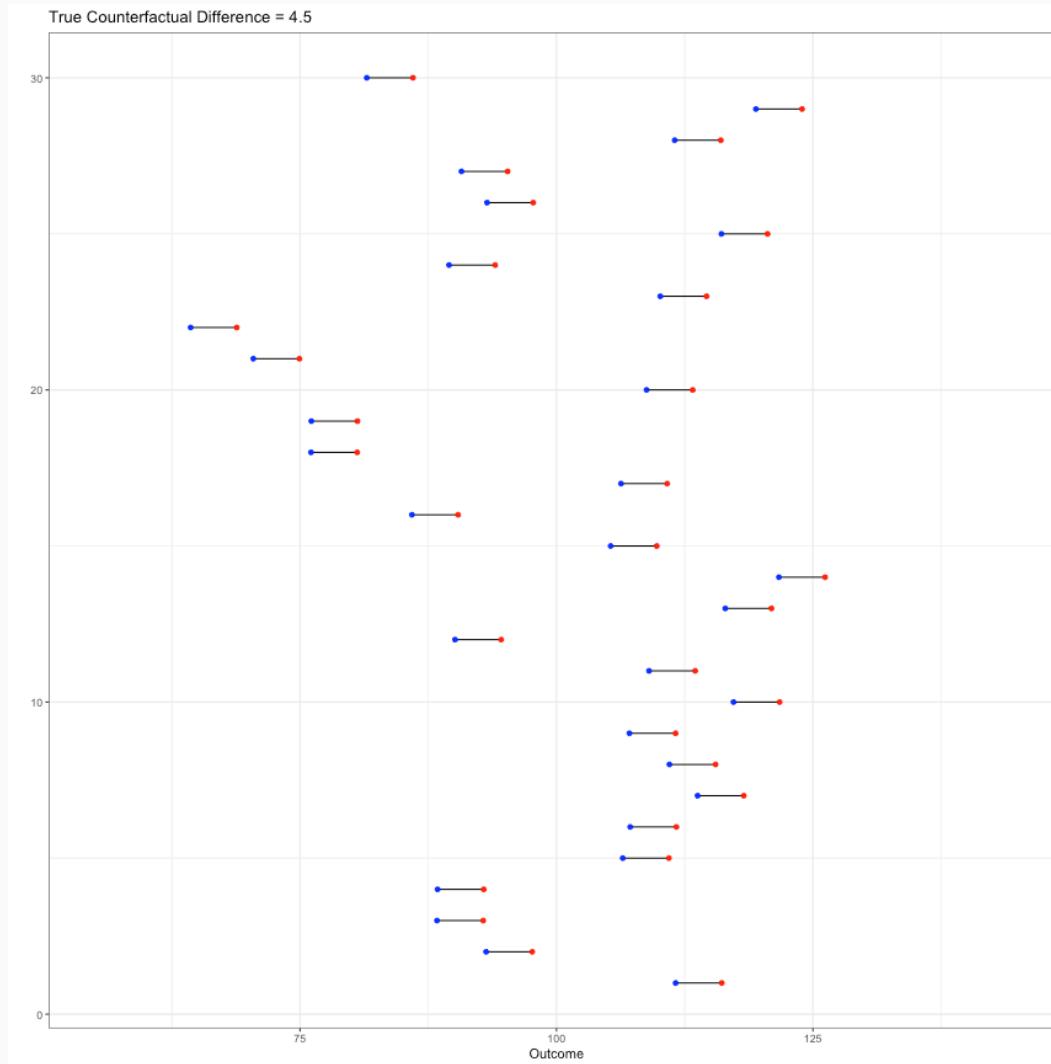
```
set.seed(2112)
pop.mean <- 100
pop.sd <- 15
pop.es <- .3
n <- 30
thedata <- data.frame(
  id = 1:30,
  center = rnorm(n, mean = pop.mean, sd = pop.sd),
  stringsAsFactors = FALSE
)
val <- pop.sd * pop.es / 2
thedata$placebo <- thedata$center - val
thedata$treatment <- thedata$center + val
thedata$diff <- thedata$treatment - thedata$placebo
thedata$RCT_Assignment <- sample(c('placebo', 'treatment'), n, replace = TRUE)
thedata$RCT_Value <- as.numeric(apply(thedata, 1,
  FUN = function(x) { return(x[x['RCT_Assignment']]) } ))
head(thedata, n = 3)
```

```
##   id    center    placebo treatment diff RCT_Assignment RCT_Value
## 1  1 113.86506 111.61506 116.11506  4.5      treatment 116.11506
## 2  2  95.38746  93.13746  97.63746  4.5      treatment  97.63746
## 3  3  90.60380  88.35380  92.85380  4.5      treatment  92.85380
```

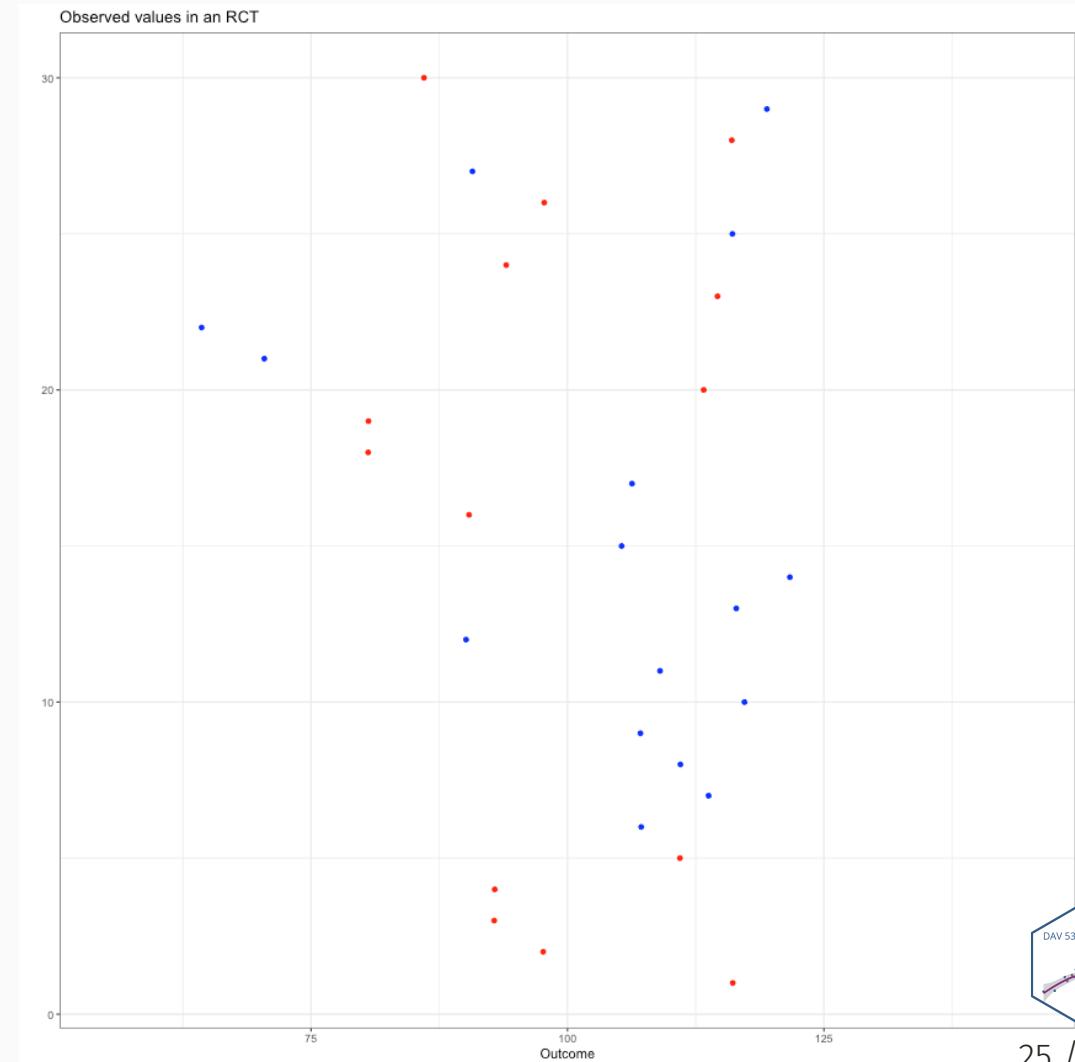
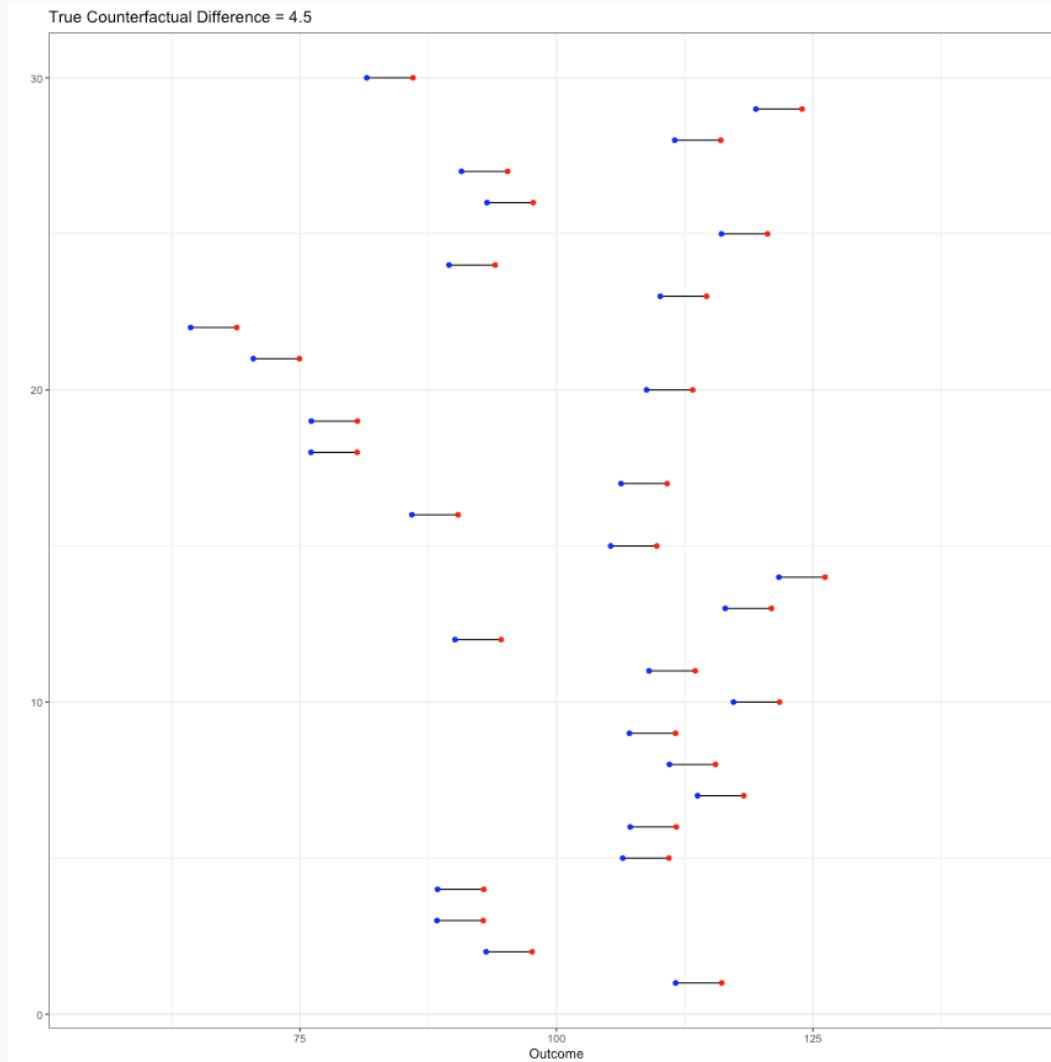
```
tab.out <- describeBy(thedata$RCT_Value, group = thedata$RCT_Assignment, mat = TRUE, skew = FALSE)
```



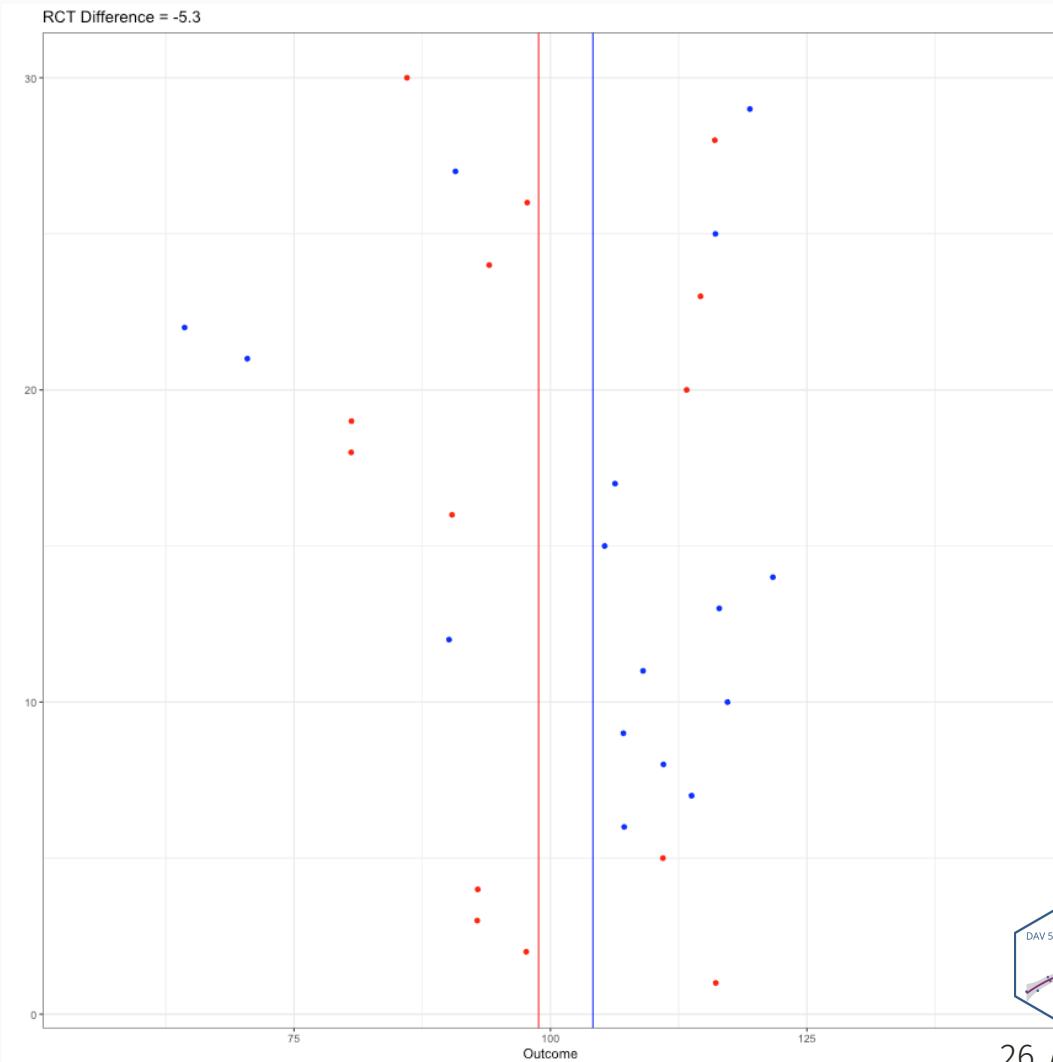
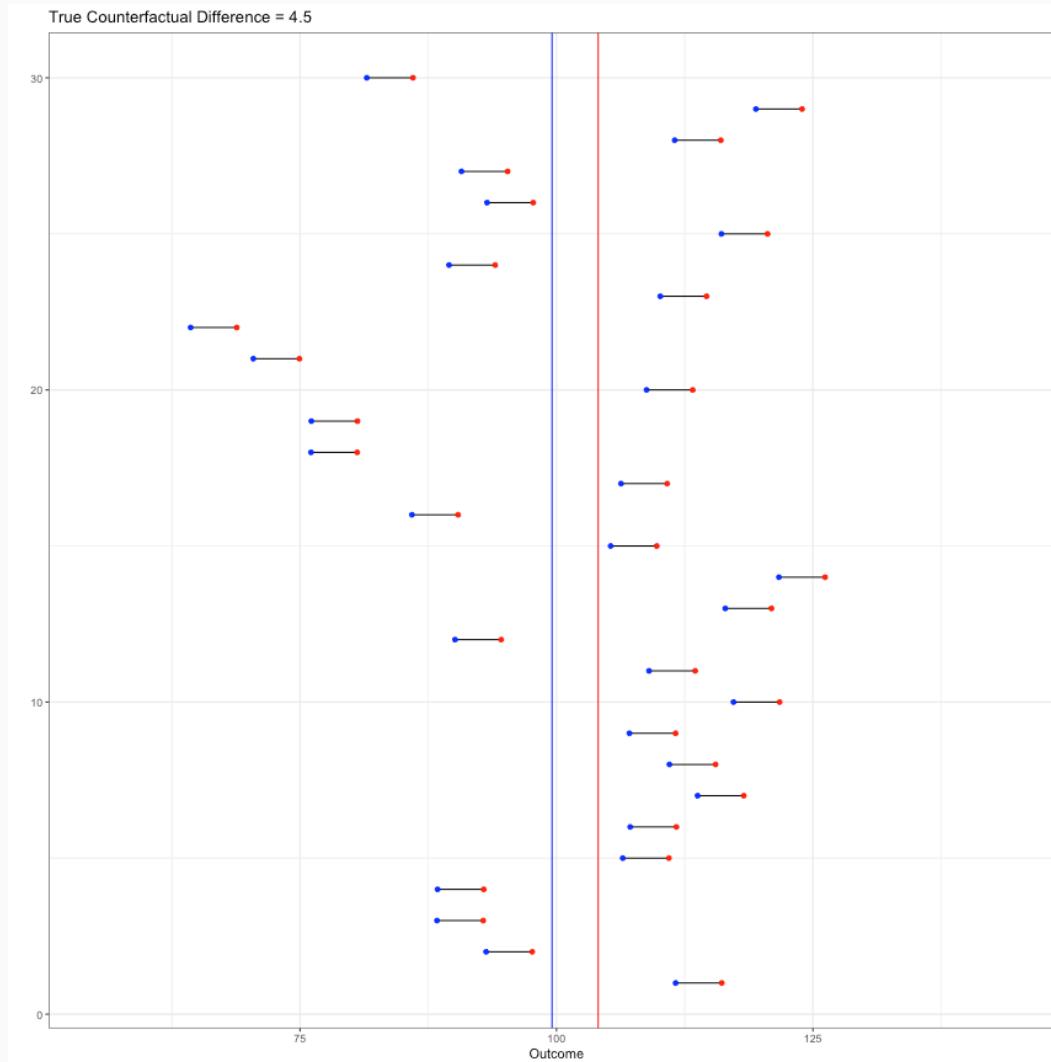
# True Counterfactual



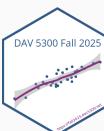
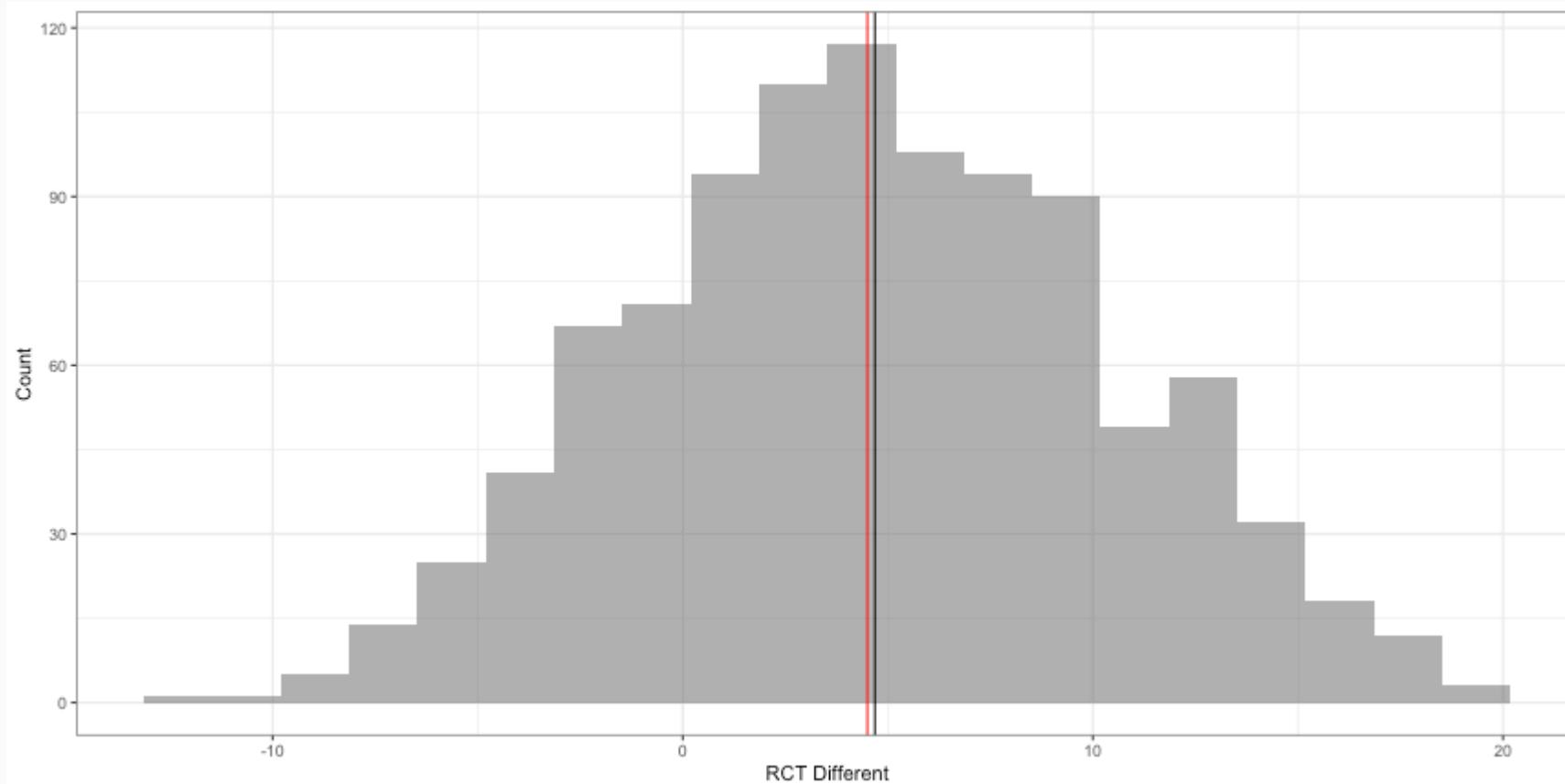
# True Counterfactual (left) vs. One RCT (right)



# True Counterfactual (left) vs. One RCT (right)



# Distribution of Differences from 1,000 RCTs

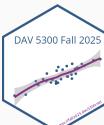
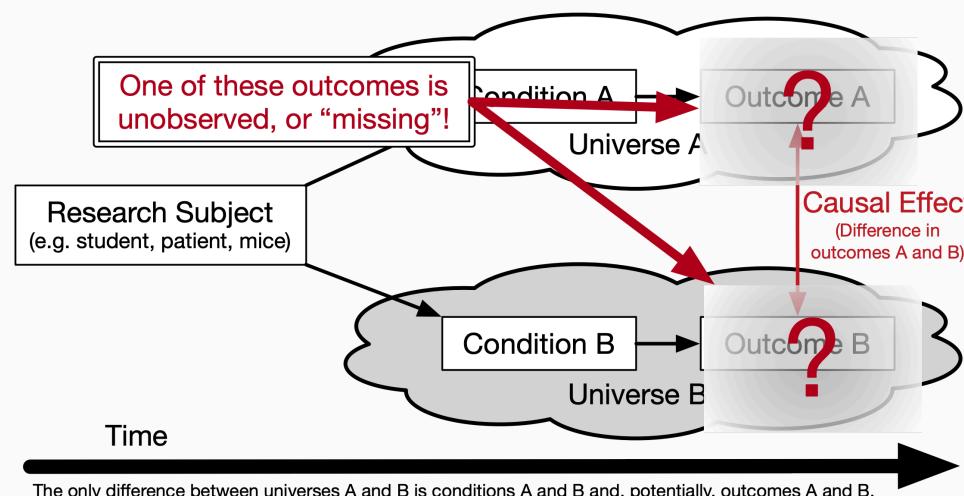


# Rubin's Causal Model

- The causal effect of a treatment is the difference in an individual's outcome under the situation they were given the treatment and not (referred to as a counterfactual).

$$\delta_i = Y_{i1} - Y_{i0}$$

- However, it is impossible to directly observe  $\delta_i$  (referred to as *The Fundamental Problem of Causal Inference*, Holland 1986).
- Rubin frames this problem as a "missing data problem" (see Rubin, 1974, 1977, 1978, 1980, and Holland, 1986).



# Propensity Score Analysis

The propensity score is the "conditional probability of assignment to a particular treatment given a vector of observed covariates" (Rosenbaum & Rubin, 1983, p. 41). The probability of being in the treatment:

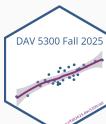
$$\pi(X_i) \equiv Pr(T_i = 1|X_i)$$

The balancing property under exogeneity:

$$T_i \perp\!\!\!\perp X_i \mid \pi(X_i)$$

We can then restate the **ignorability assumption** with the propensity score:

$$(Y_i(1), Y_i(0)) \perp\!\!\!\perp T_i \mid \pi(X_i)$$



# Treatment Effects

The average treatment effect (ATE) is defined as:

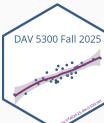
$$E(r_1) - E(r_0)$$

where  $E(\cdot)$  is the expectation in the population. For a set of covariates,  $X$ , and outcomes  $Y$  where 0 denotes control and 1 treatment, we define ATE as:

$$ATE = E(Y_1 - Y_0|X) = E(Y_1|X) - E(Y_0|X)$$

As we will see later there are alternative treatment effects (estimands) we can estimate instead of ATE.

What Rosenbaum and Rubin (1983) proved in their seminal paper is that the propensity score is a univariate representation of the multivariate matrix. As we will see later, two observations with very similar propensity scores will look similar across all the observed covariates.



# Propensity Score Analysis in Three Phases



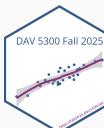
# Simulated Example

We will simulate a dataset with three covariates,  $x_1$  and  $x_2$  which are continuous and  $x_3$  which is categorical. The assumed treatment effect is 1.5.

```
n <- 500
treatment_effect <- 1.5
X <- mvtnorm::rmvnorm(
  n,
  mean = c(0.5, 1, 0),
  sigma = matrix(c(2, 1, 1,
                  1, 1, 1,
                  1, 1, 1),
                 ncol = 3) )
dat <- tibble(
  x1 = X[, 1],
  x2 = X[, 2],
  x3 = X[, 3] > 0,
  treatment = as.numeric(- 0.5 +
    0.25 * x1 +
    0.75 * x2 +
    0.05 * x3 +
    rnorm(n, 0, 1) > 0),
  outcome = treatment_effect * treatment +
    rnorm(n, 0, 1))
```

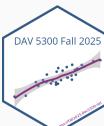
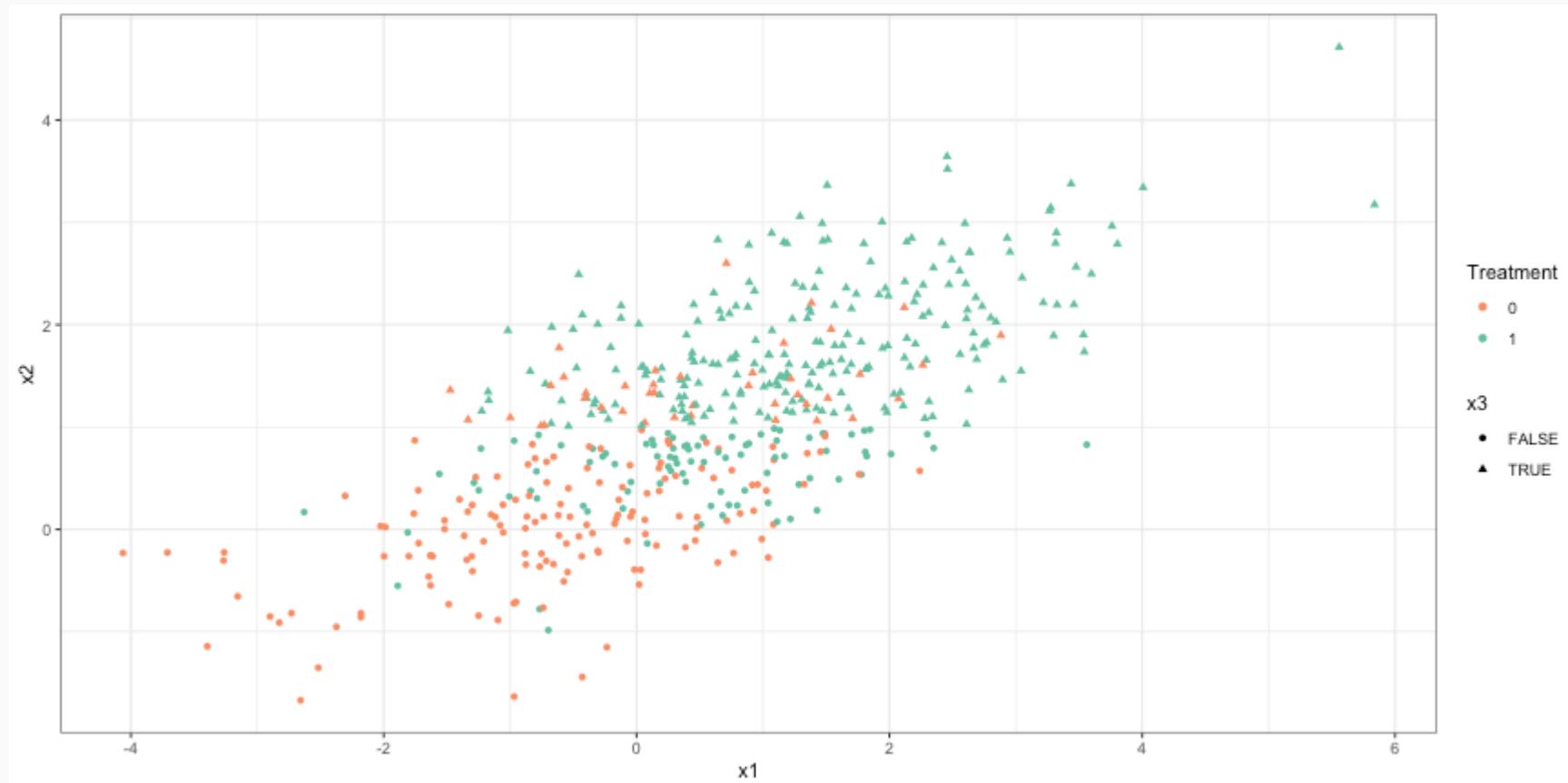
```
head(dat, n = 6)

## # A tibble: 6 × 5
##       x1     x2   x3  treatment  outcome
##     <dbl>  <dbl> <lgl>    <dbl>    <dbl>
## 1  1.35  0.744 FALSE     0  1.46
## 2  0.149 1.55  TRUE     0 -0.924
## 3  2.47  2.39  TRUE     1 -0.0527
## 4  2.29  1.66  TRUE     1  1.05
## 5  2.93  2.85  TRUE     1  0.721
## 6 -0.867 0.125 FALSE     0  0.723
```

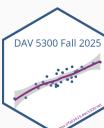
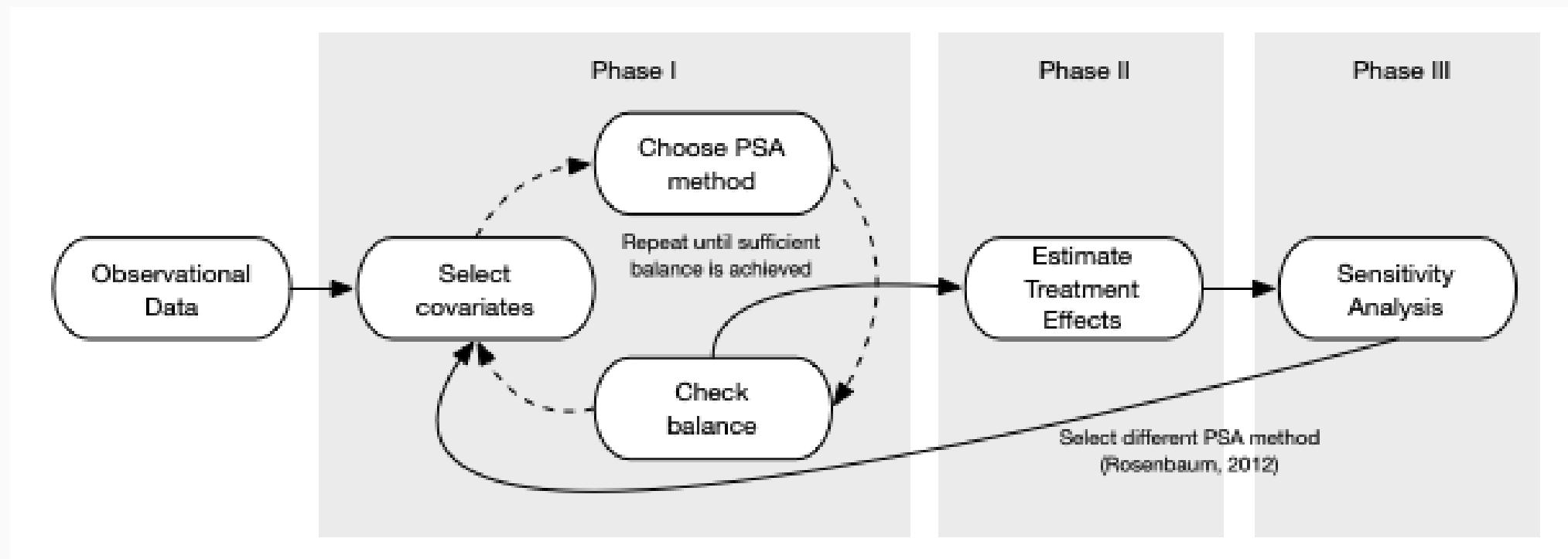


# Scatterplot

```
ggplot(dat, aes(x = x1, y = x2, shape = x3, color = factor(treatment))) +  
  geom_point() + scale_color_manual('Treatment', values = cols)
```



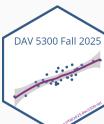
# Steps for Implementing Propensity Score Analysis



# Propensity score methods

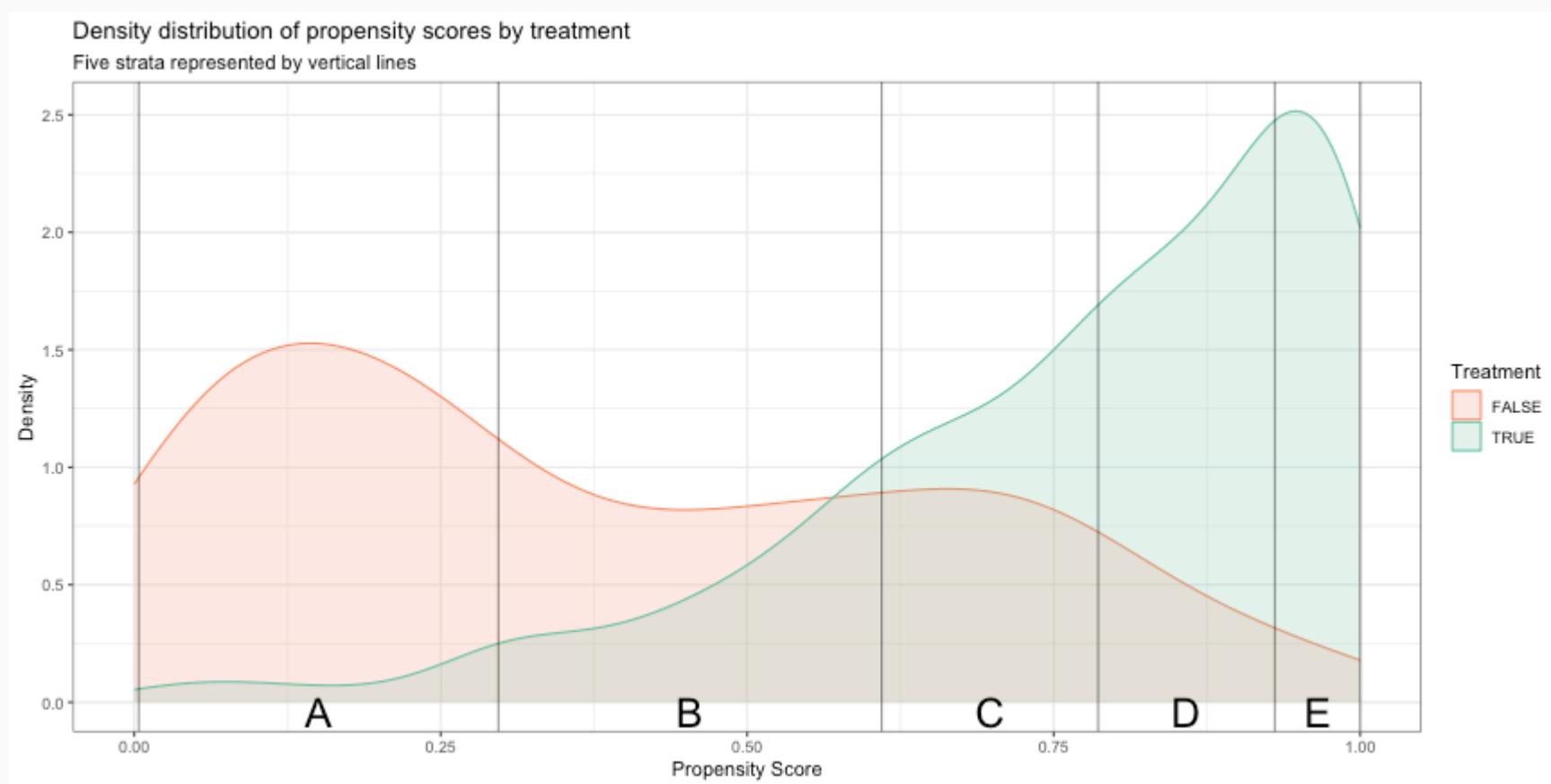
There are three major approaches for conducting PSA:

- **Stratification** Treatment and comparison units are divided into strata (or subclasses) so that treated and comparison units are similar within each strata. Cochran (1968) observed that creating five subclassifications (stratum) removes at least 90% of the bias in the estimated treatment effect.
- **Matching** - Each treatment unit is paired with a comparison unit based upon the pre-treatment covariates.
- **Weighting** Each observation is weighted by the inverse of the probability of being in that group.



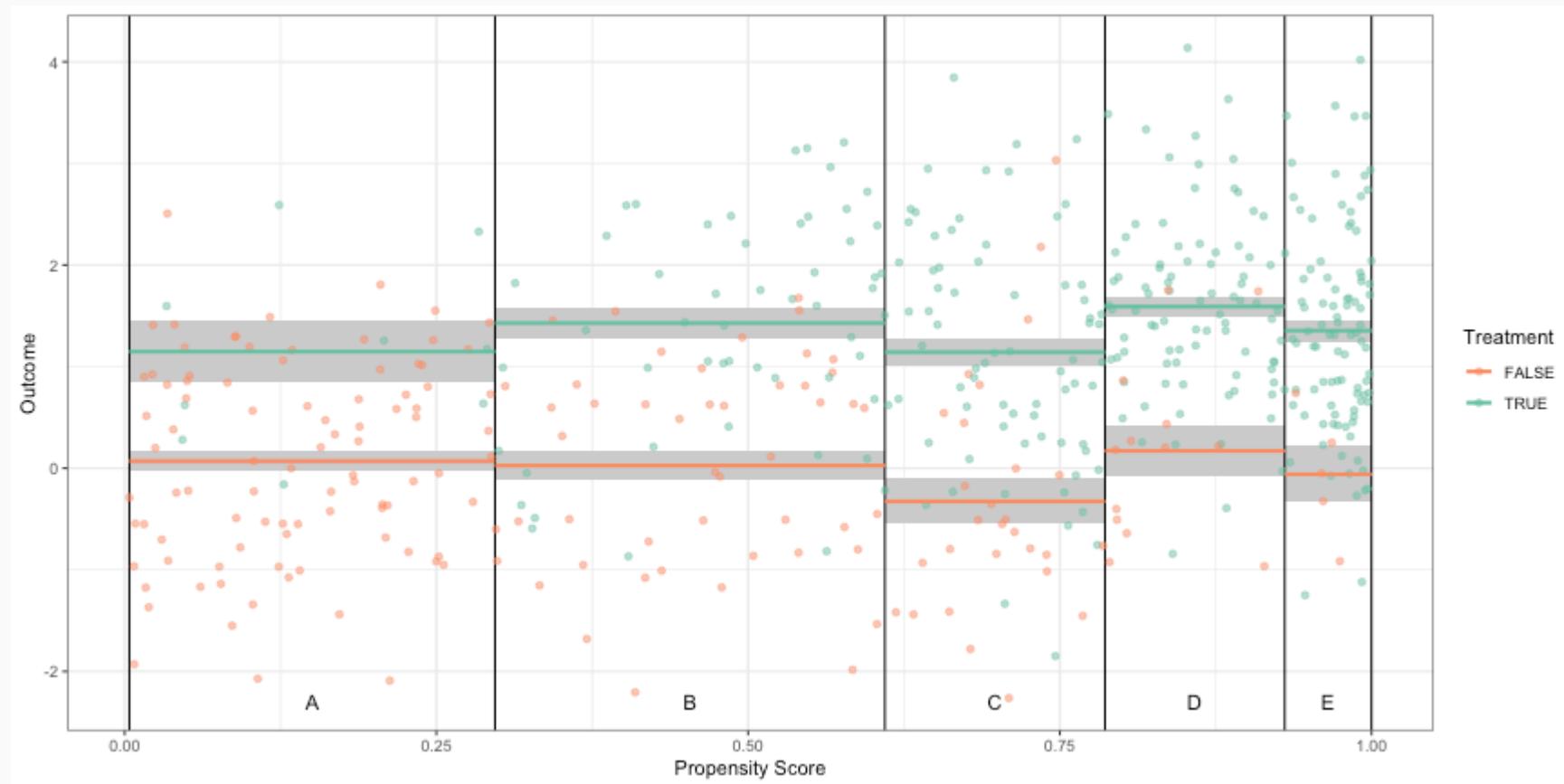
# Stratification

Stratification involves dividing (or stratifying) the observations into subgroups based upon the propensity score. Here, we used quintiles on the propensity scores where were estimated using logistic regression. For classification trees the stratum is determined by the leaf nodes.



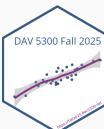
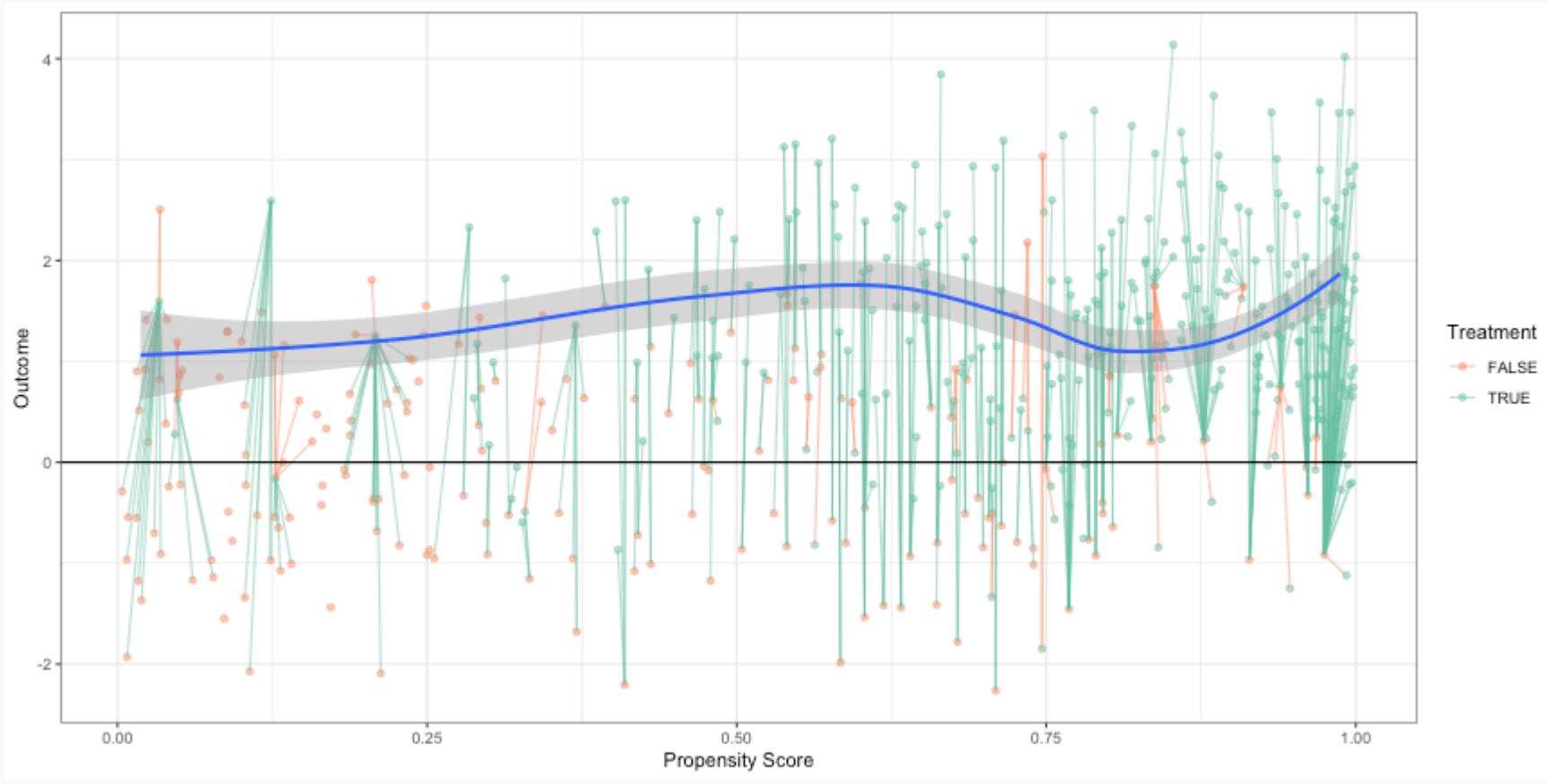
# Stratification (cont.)

Independent sample tests (e.g.  $t$ -tests) are conducted within each stratum and pooled to provide an overall estimate.



# Matching

Dependent sample tests (e.g.  $t$ -tests) are conducted using match pairs to provide a treatment.



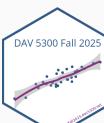
# Matching Methods

There are many choices and approaches to matching, including:

- Propensity score matching.
- Limited exact matching.
- Full matching.
- Nearest neighbor matching.
- Optimal/Genetic matching.
- Mahalanobis distance matching (for quantitative covariates only).
- Matching with and without replacement.
- One-to-one or one-to-many matching.

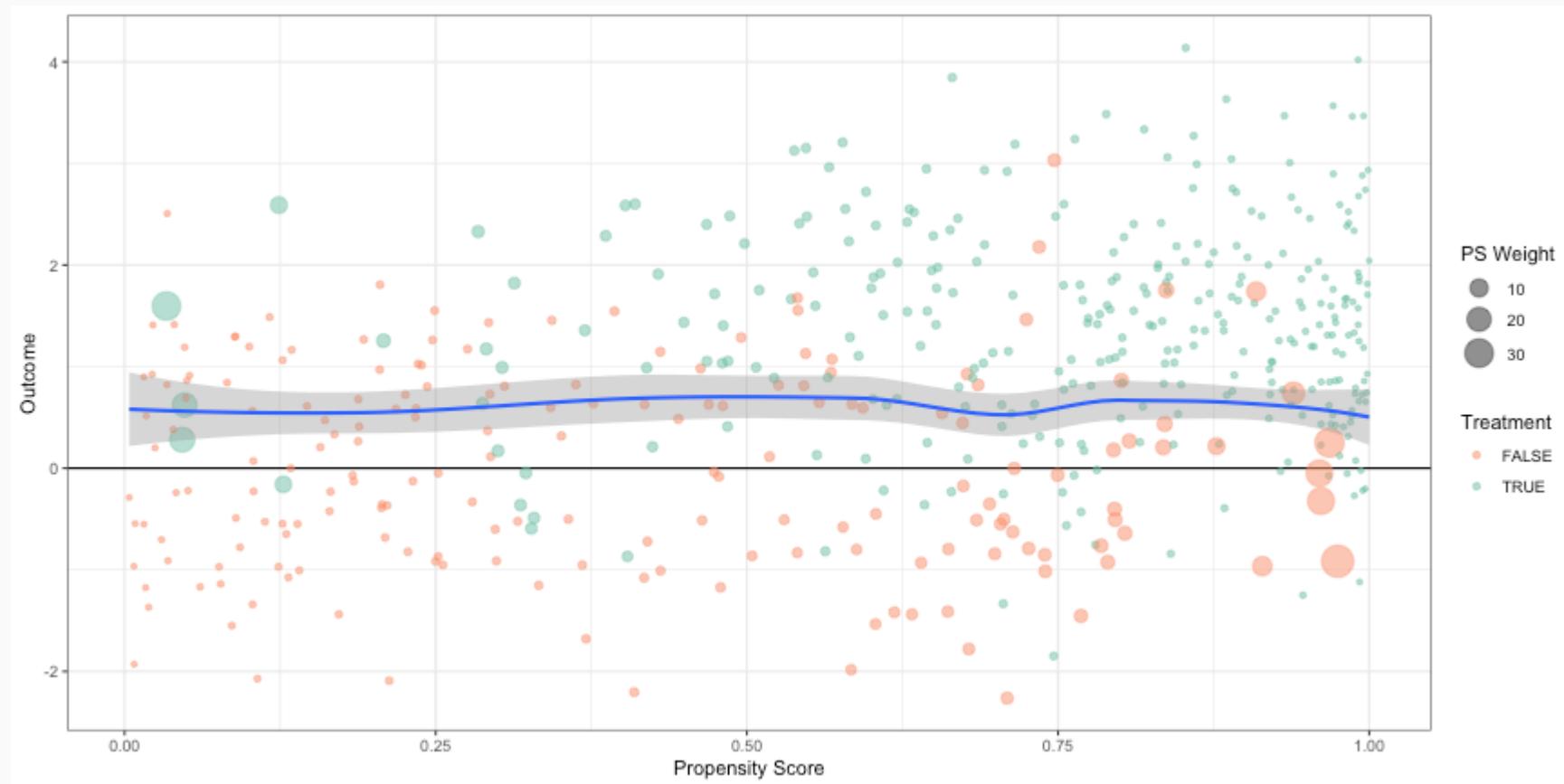
Which method should you use?

*Whichever one gives the best balance!*



# Weighting

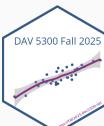
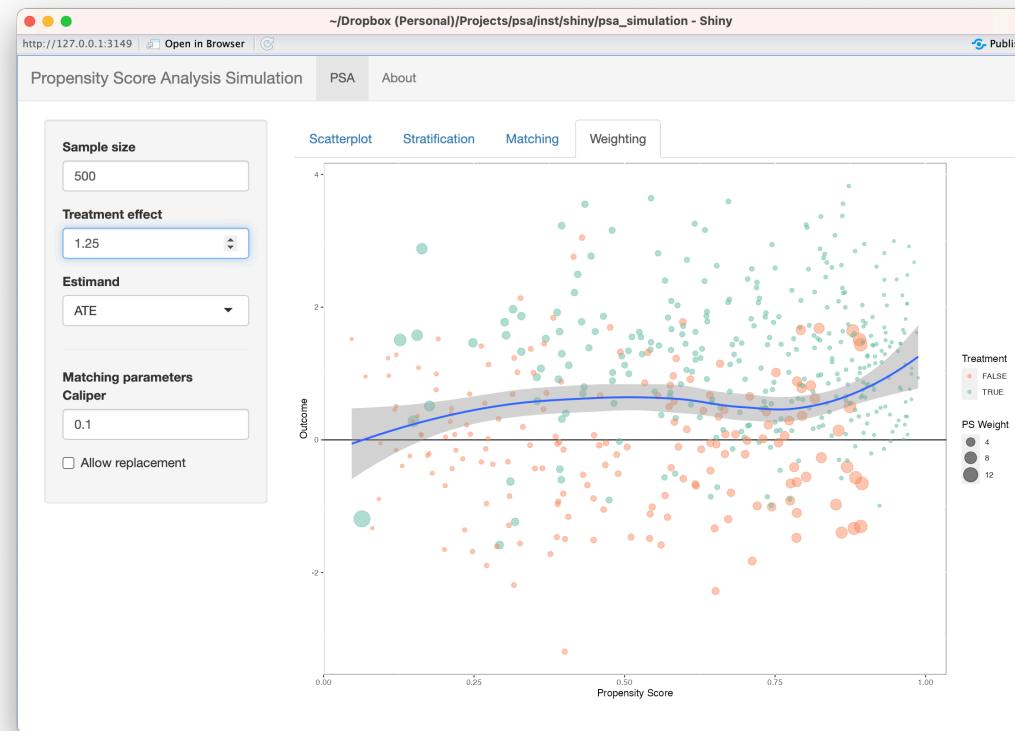
Propensity score weights can be used as regression weights, the specific weights depend on the desired estimand and will be provided in later slides.



# Shiny Application

We can explore how these three plots change as the treatment effects change using the `psa::psa_simulation_shiny()` application.

`psa::psa_simulation_shiny()`



# Phase I: Estimate Propensity Scores

In this example we will use logistic regression to estimate the propensity scores.

```
lr.out <- glm(  
  treatment ~ x1 + x2 + x3,  
  data = dat,  
  family = binomial(link='logit'))  
dat$ps <- fitted(lr.out) # Propensity scores
```

For stratification we will use quintiles to split the observations into five equal groups.

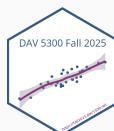
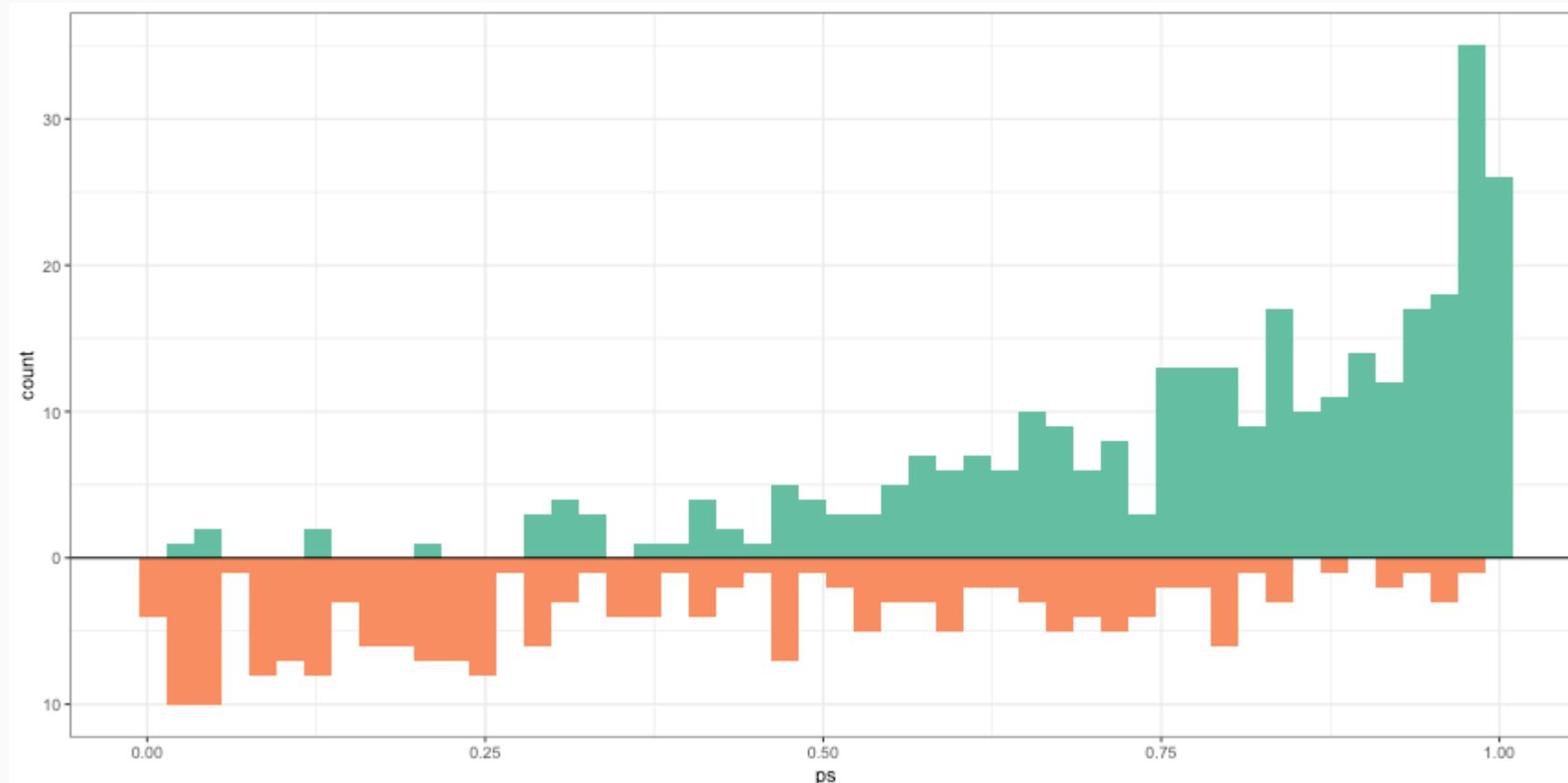
```
breaks5 <- psa::get_strata_breaks(dat$ps)  
dat$strata5 <- cut(  
  x = dat$ps,  
  breaks = breaks5$breaks,  
  include.lowest = TRUE,  
  labels = breaks5$labels$strata)
```

```
summary(lr.out)  
  
##  
## Call:  
## glm(formula = treatment ~ x1 + x2 + x3, family = binomial(link = "logit"),  
##       data = dat)  
##  
## Coefficients:  
##                               Estimate Std. Error z value Pr(>|z|)  
## (Intercept) -1.1006     0.2069 -5.319 1.04e-07 ***  
## x1          0.4399     0.1266  3.476  0.00051 ***  
## x2          1.9818     0.3404  5.823 5.79e-09 ***  
## x3TRUE      -0.7166     0.4087 -1.753  0.07955 .  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## (Dispersion parameter for binomial family taken to be 1)  
##  
## Null deviance: 658.96 on 499 degrees of freedom  
## Residual deviance: 432.95 on 496 degrees of freedom  
## AIC: 440.95  
##  
## Number of Fisher Scoring iterations: 5
```



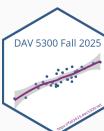
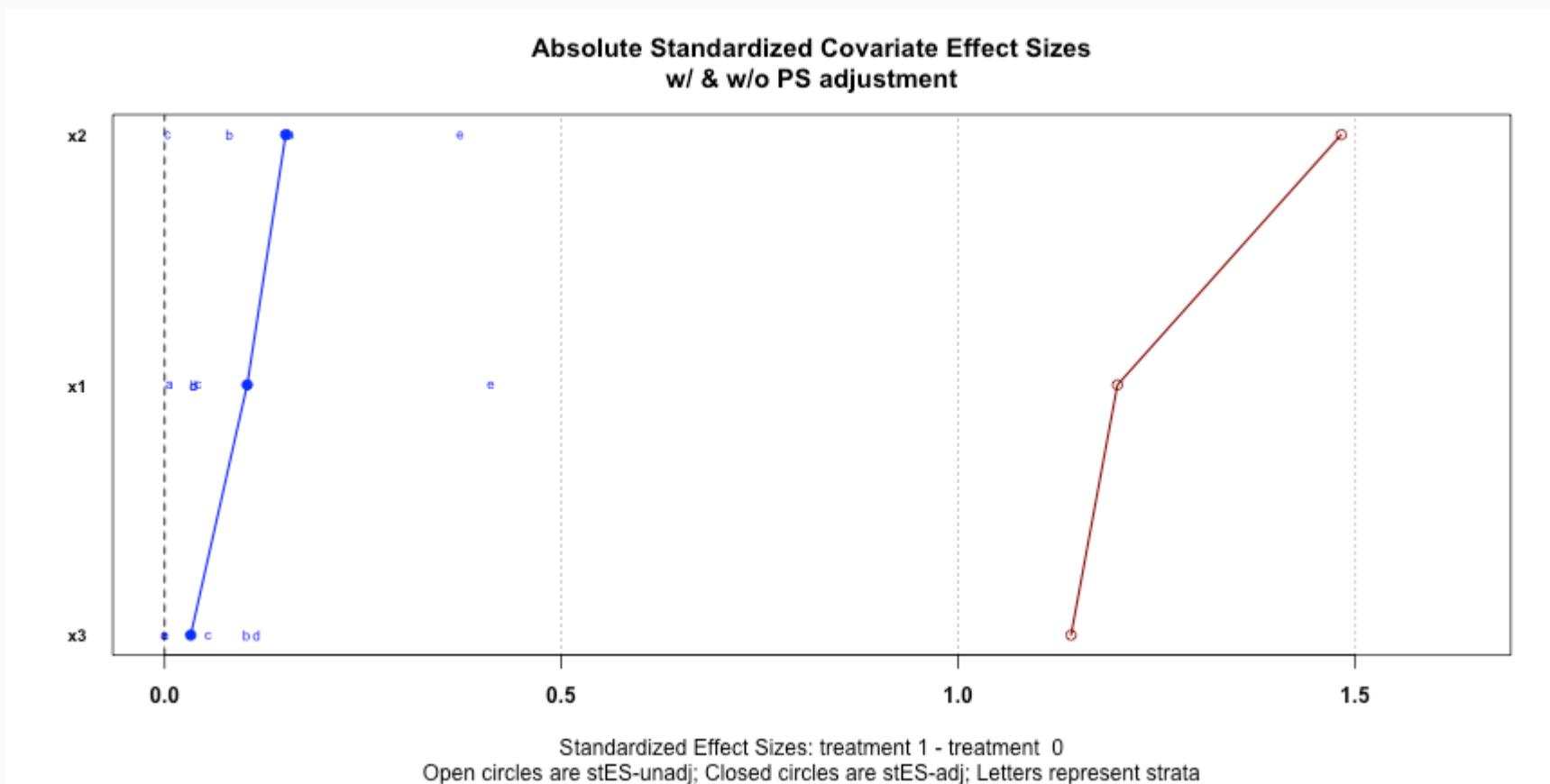
# Distribution of Propensity Scores

```
ggplot(dat) +  
  geom_histogram(data = dat[dat$treatment == 1,], aes(x = ps, y = after_stat(count)), bins = 50, fill = cols[2]) +  
  geom_histogram(data = dat[dat$treatment == 0,], aes(x = ps, y = -after_stat(count)), bins = 50, fill = cols[1]) +  
  geom_hline(yintercept = 0, lwd = 0.5) +    scale_y_continuous(label = abs)
```



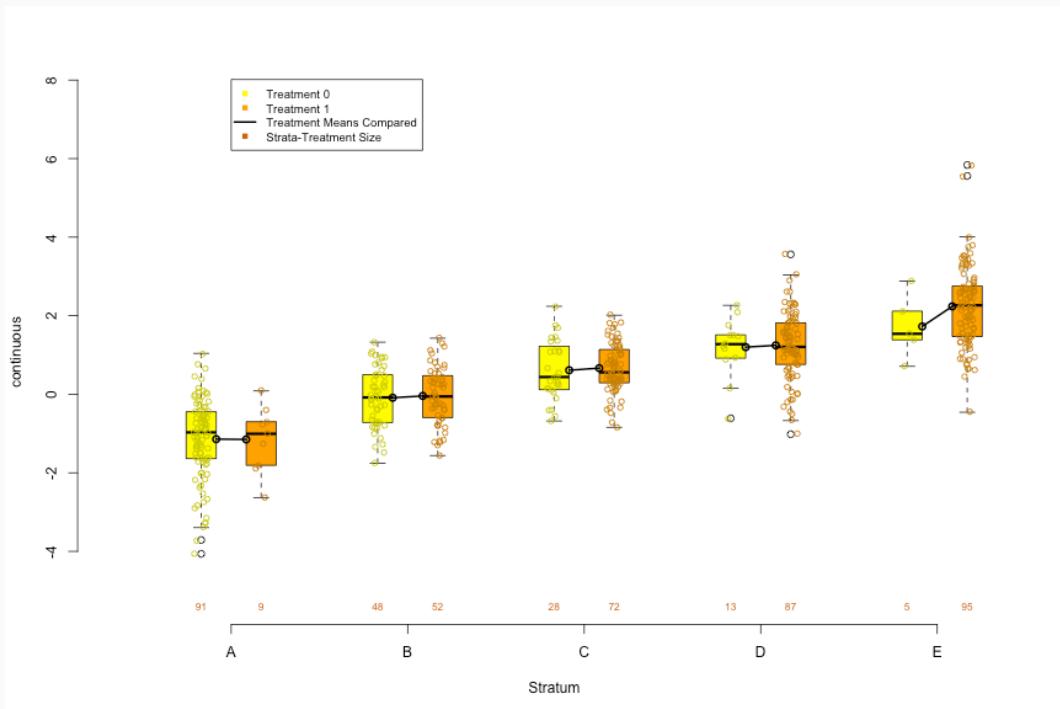
# Check Balance: Multiple Covariates

```
PSAgraphics::cv.bal.psa(dat[,1:3], dat$treatment, dat$ps, strata = 5)
```

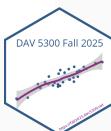
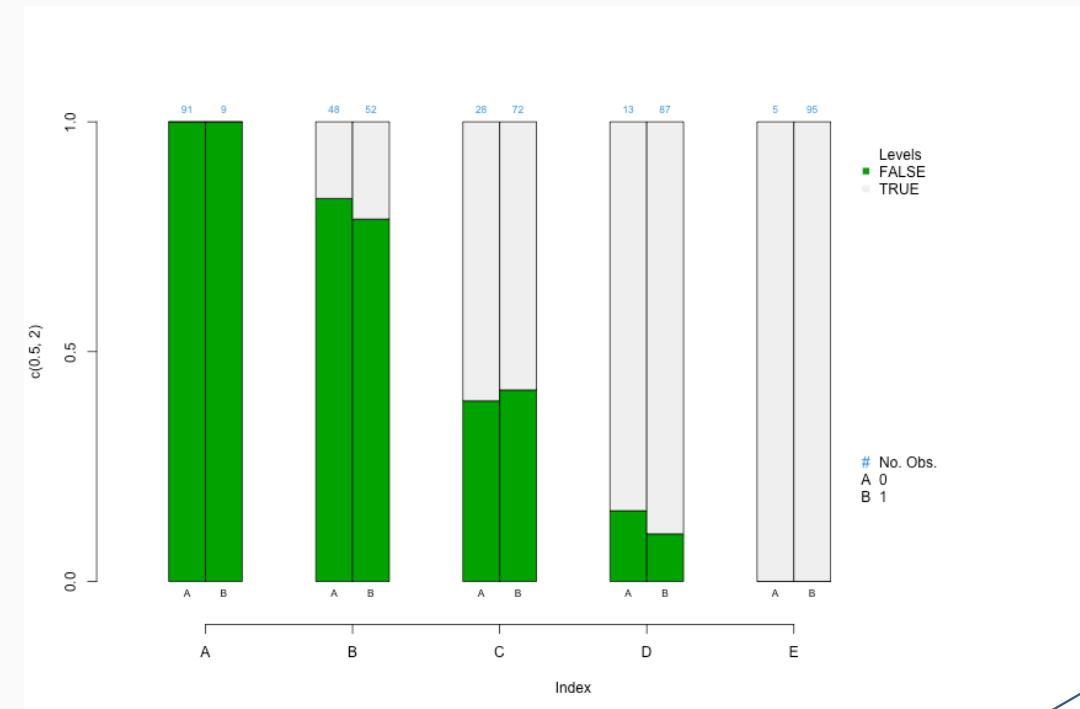


# Check Balance: Single Covariate

```
PSAgraphics::box.psa(dat$x1,  
                      dat$treatment,  
                      dat$strata5)
```

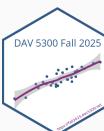
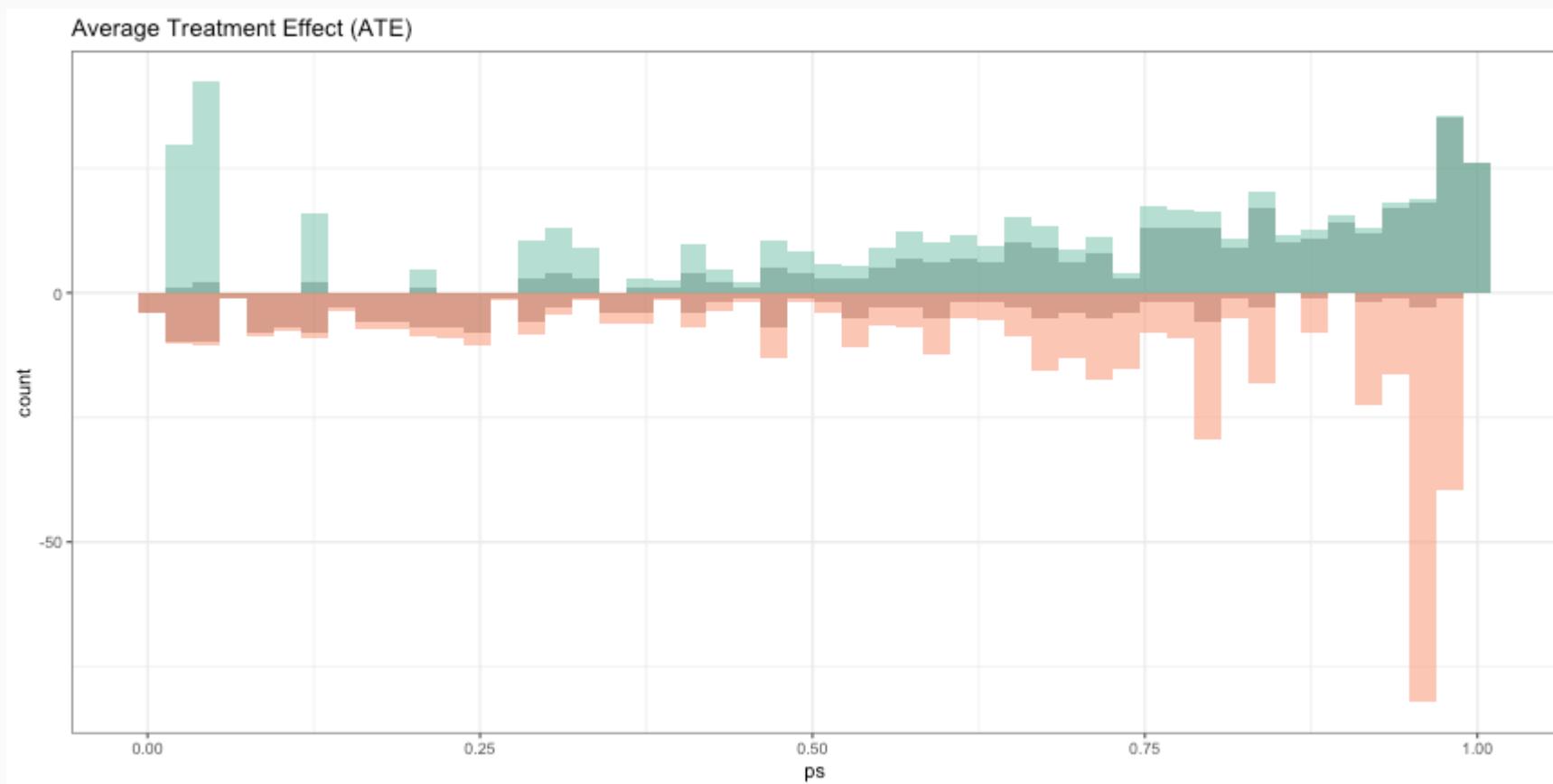


```
PSAgraphics::cat.psa(dat$x3,  
                      dat$treatment,  
                      dat$strata5)
```



# Average Treatment Effect (ATE)

$$ATE = E(Y_1 - Y_0|X) = E(Y_1|X) - E(Y_0|X)$$



# Treatment Effects for Weighting

$$Treatment\ Effect = \frac{\sum Y_i Z_i w_i}{\sum Z_i w_i} - \frac{\sum Y_i (1 - Z_i) w_i}{\sum (1 - Z_i) w_i}$$

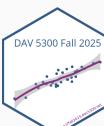
Where  $w$  is the weight (as defined in the following sections),  $Z_i$  is the treatment assignment such that  $Z = 1$  is treatment and  $Z = 0$  is control, and  $Y_i$  is the outcome

$$w_{ATE} = \frac{Z_i}{\pi_i} + \frac{1 - Z_i}{1 - \pi_i}$$

$$w_{ATT} = \frac{\pi_i Z_i}{\pi_i} + \frac{\pi_i (1 - Z_i)}{1 - \pi_i}$$

$$w_{ATC} = \frac{(1 - \pi_i) Z_i}{\pi_i} + \frac{(1 - e_i)(1 - Z_i)}{1 - \pi_i}$$

$$w_{ATM} = \frac{\min\{\pi_i, 1 - \pi_i\}}{Z_i \pi_i (1 - Z_i) (1 - \pi_i)}$$



# Treatment Effects

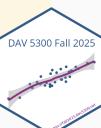
## Average Treatment Effect

```
psa::treatment_effect(  
  treatment = dat$treatment,  
  outcome = dat$outcome,  
  weights = dat$ate_weight)
```

```
## [1] 1.336979
```

```
lm(outcome ~ treatment,  
  data = dat,  
  weights = dat$ate_weight)
```

```
##  
## Call:  
## lm(formula = outcome ~ treatment, data = dat, weights = dat$ate_weight)  
##  
## Coefficients:  
## (Intercept)      treatment  
##       -0.044        1.337
```



# Example: National Supported Work Demonstration



# National Supported Work

The National Supported Work (NSW) Demonstration was a federally and privately funded randomized experiment done in the 1970s to estimate the effects of a job training program for disadvantaged workers.

- Participants were randomly selected to participate in the training program.
- Both groups were followed up to determine the effect of the training on wages.
- Analysis of the mean differences (unbiased given randomization), was approximately \$800.

Lalonde (1986) used data from the Panel Survey of Income Dynamics (PSID) and the Current Population Survey (CPS) to investigate whether non-experimental methods would result in similar results to the randomized experiment. He found results ranging from \$700 to \$16,000.



# National Supported Work (cont.)

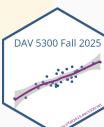
Dehejia and Wahba (1999) later used propensity score matching to analyze the data. They found that,

- Comparison groups selected by Lalonde were very dissimilar to the treated group.
- By restricting the comparison group to those that were similar to the treated group, they could replicate the original NSW results.
- Using the CPS data, the range of treatment effect was between \$1,559 to \$1,681. The experimental results for the sample sample was approximately \$1,800.

The covariates available include: age, education level, high school degree, marital status, race, ethnicity, and earnings in 1974 and 1975.

Outcome of interest is earnings in 1978.

```
data(lalonde, package='Matching')
```



# Estimating Propensity Scores

Estimate propensity scores using logistic regression.

```
lalonde.formu <- treat ~ age + educ + black + hisp +
  married + nodegr + re74 + re75
glm1 <- glm(lalonde.formu,
            data = lalonde,
            family = binomial(link = 'logit'))
```

Get the propensity scores:

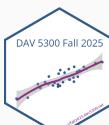
```
lalonde$ps <- fitted(glm1)
```

Define the stratification:

```
strata5 <- cut(lalonde$ps,
                 quantile(lalonde$ps, seq(0, 1, 1/5)),
                 include.lowest = TRUE,
                 labels = letters[1:5])
```

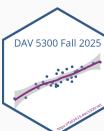
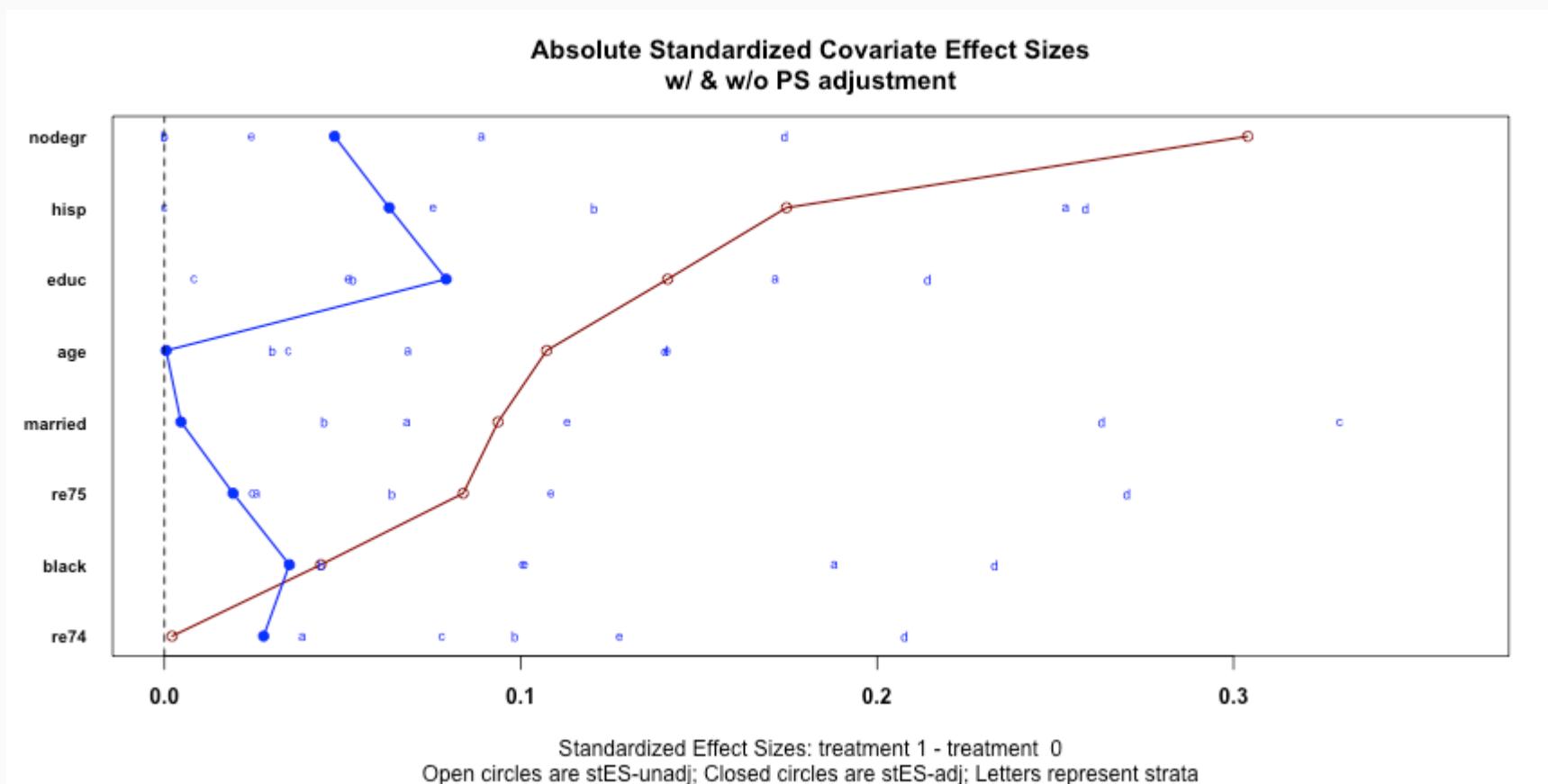
```
summary(glm1)

##
## Call:
## glm(formula = lalonde.formu, family = binomial(link = "logit"),
##      data = lalonde)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 1.178e+00 1.056e+00 1.115 0.26474
## age         4.698e-03 1.433e-02 0.328 0.74297
## educ        -7.124e-02 7.173e-02 -0.993 0.32061
## black       -2.247e-01 3.655e-01 -0.615 0.53874
## hisp        -8.528e-01 5.066e-01 -1.683 0.09228 .
## married     1.636e-01 2.769e-01 0.591 0.55463
## nodegr     -9.035e-01 3.135e-01 -2.882 0.00395 **
## re74        -3.161e-05 2.584e-05 -1.223 0.22122
## re75        6.161e-05 4.358e-05 1.414 0.15744
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 604.20  on 444  degrees of freedom
## Residual deviance: 587.22  on 436  degrees of freedom
## AIC: 605.22
##
## Number of Fisher Scoring iterations: 4
```



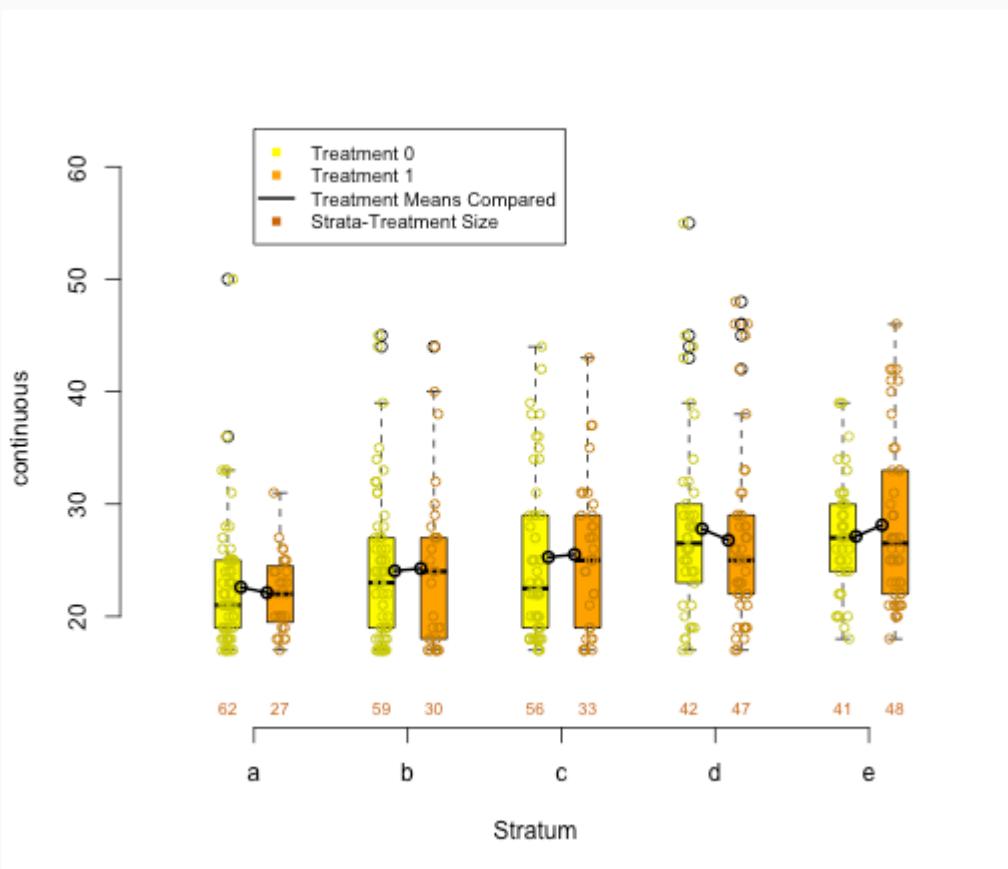
# Checking Balance: Covariate Balance Plot

```
covars <- all.vars(lalonde.formu)
covars <- lalonde[,covars[2:length(covars)]]
cv.bal.psa(covars, lalonde$treat, lalonde$ps, strata = 5)
```

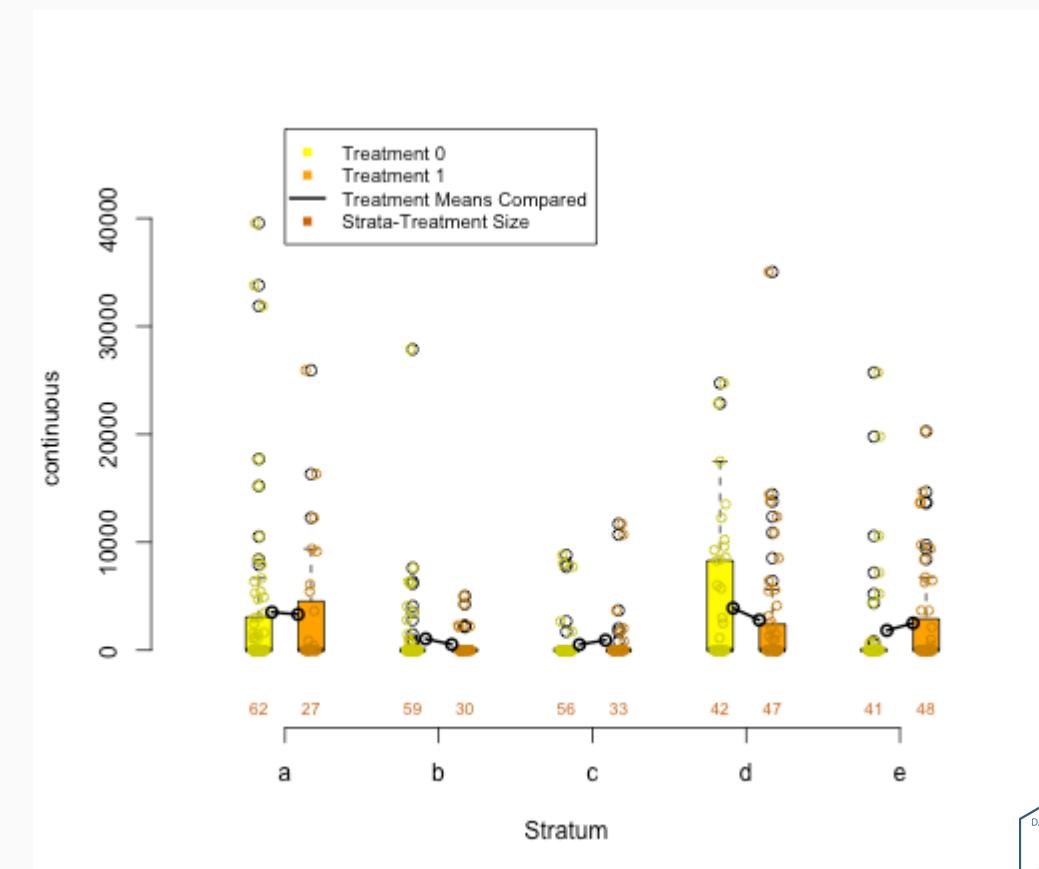


# Checking Balance: Continuous Covariates

```
box.psa(lalonde$age, lalonde$treat, strata5)
```

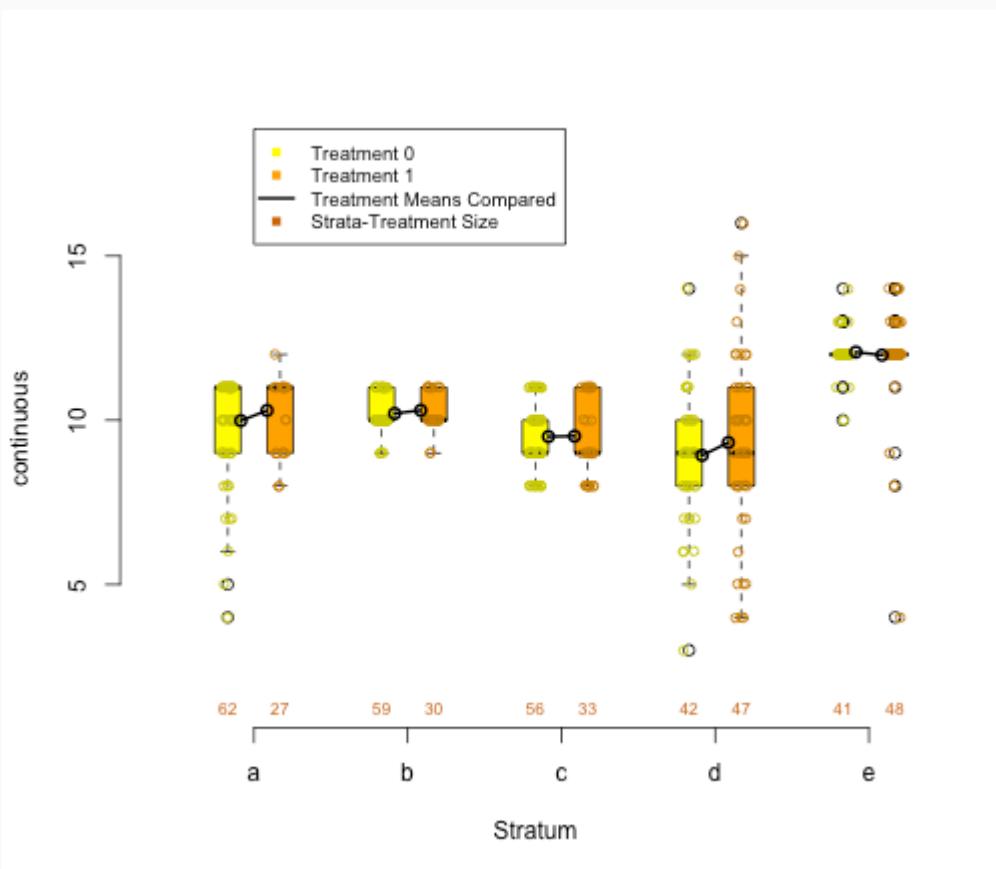


```
box.psa(lalonde$re74, lalonde$treat, strata5)
```

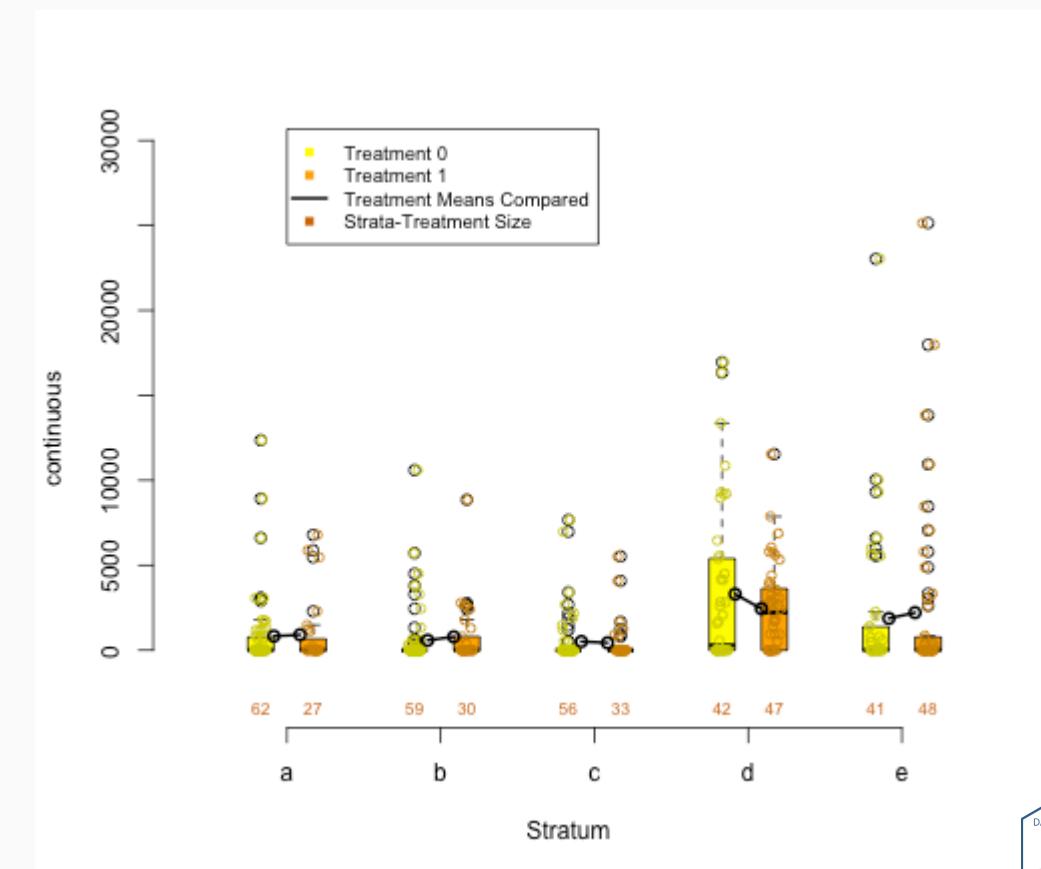


# Checking Balance: Continuous Covariates (cont.)

```
box.psa(lalonde$educ, lalonde$treat, strata5)
```

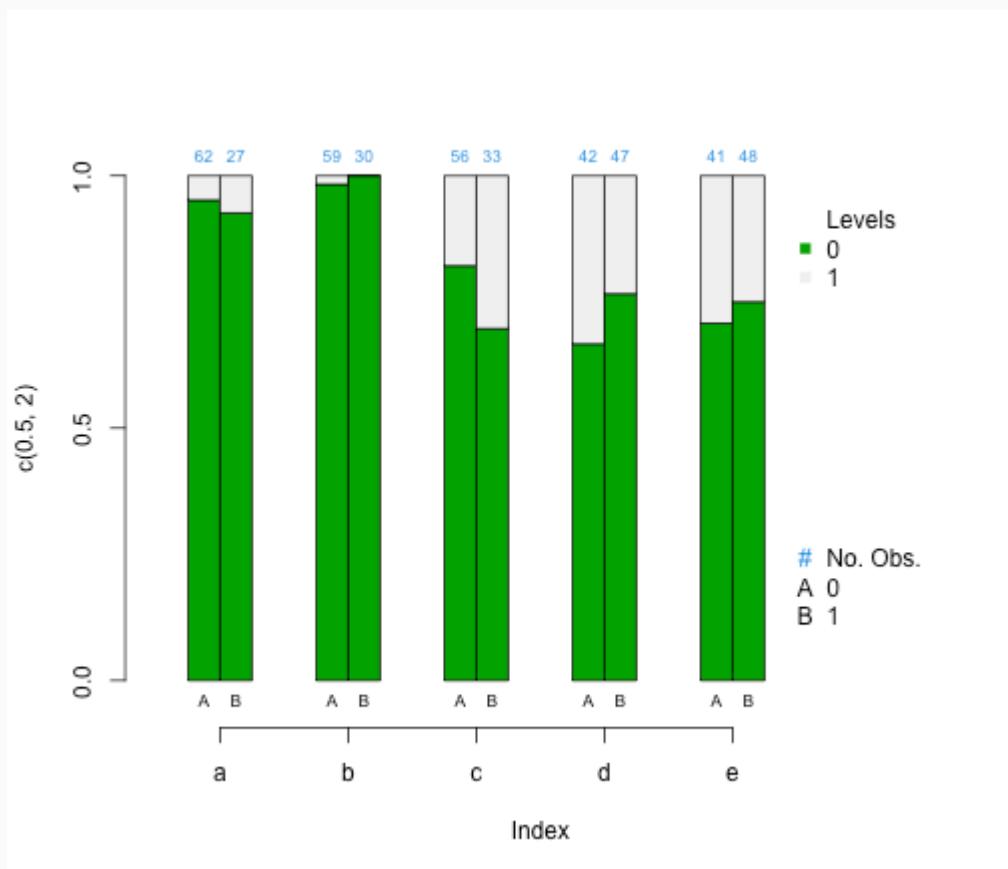


```
box.psa(lalonde$re75, lalonde$treat, strata5)
```

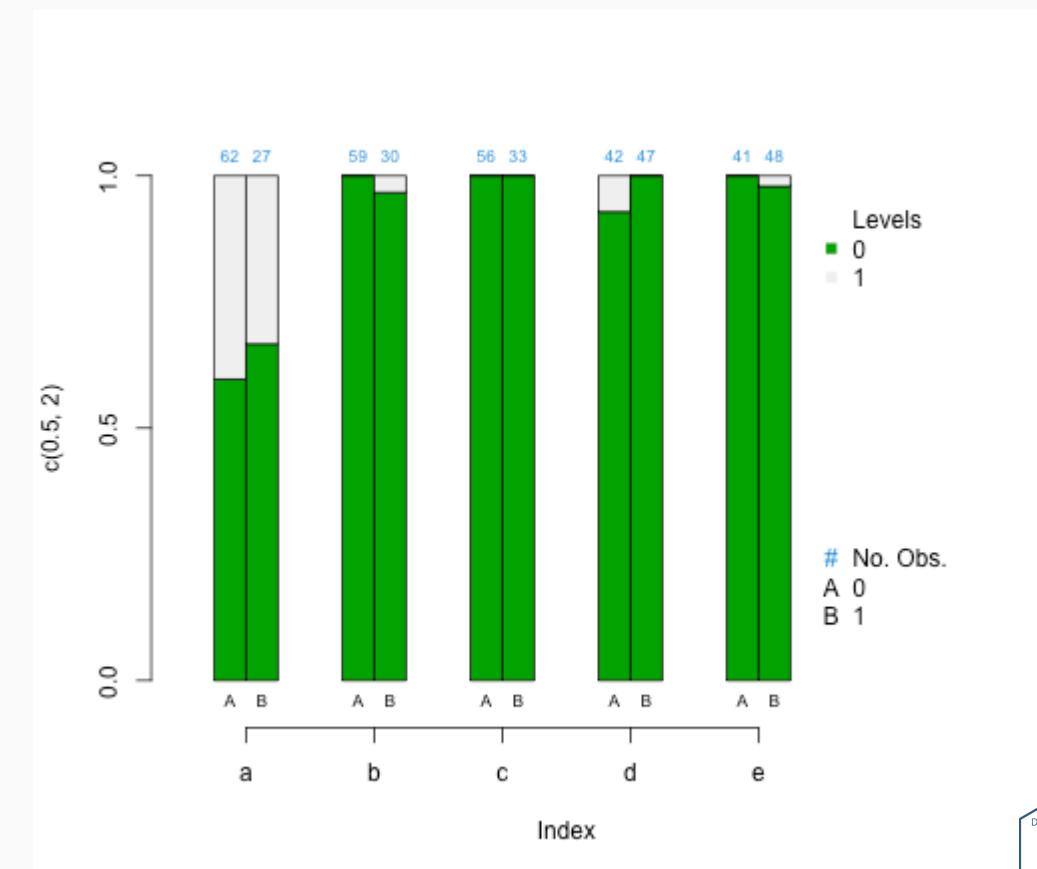


# Checking Balance: Categorical Covariates

```
cat.psa(lalonde$married, lalonde$treat, strata5)
```

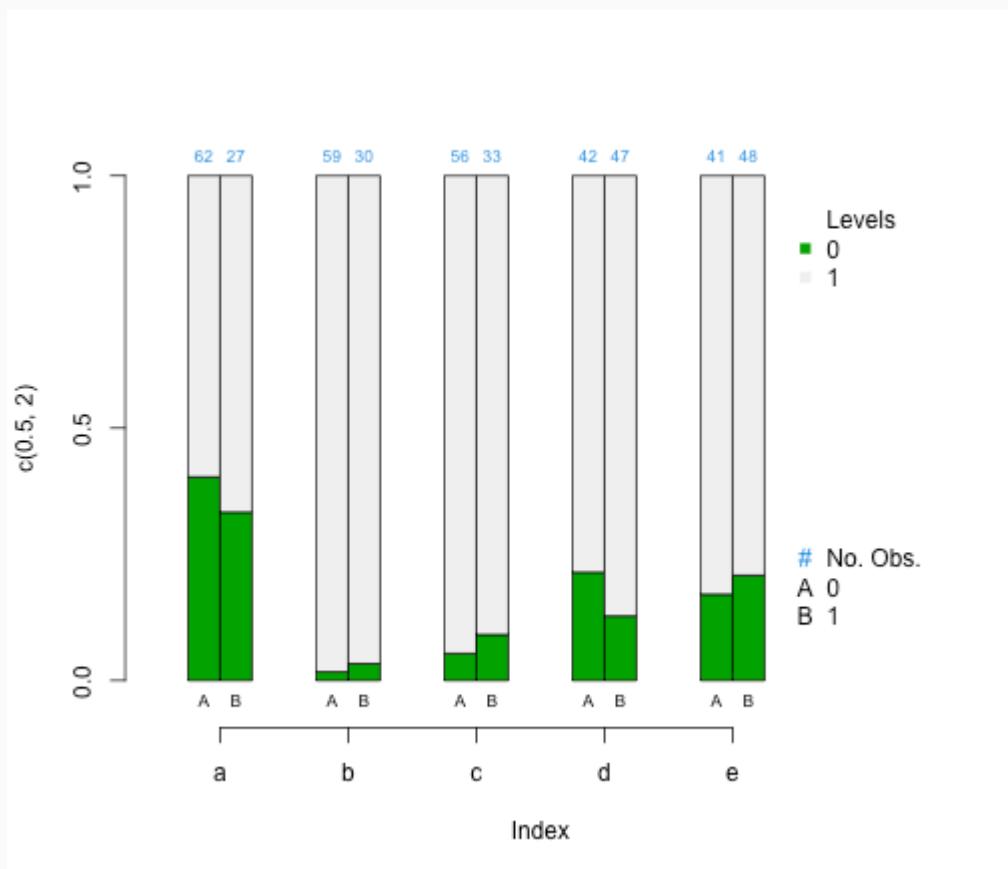


```
cat.psa(lalonde$hisp, lalonde$treat, strata5)
```

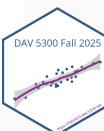
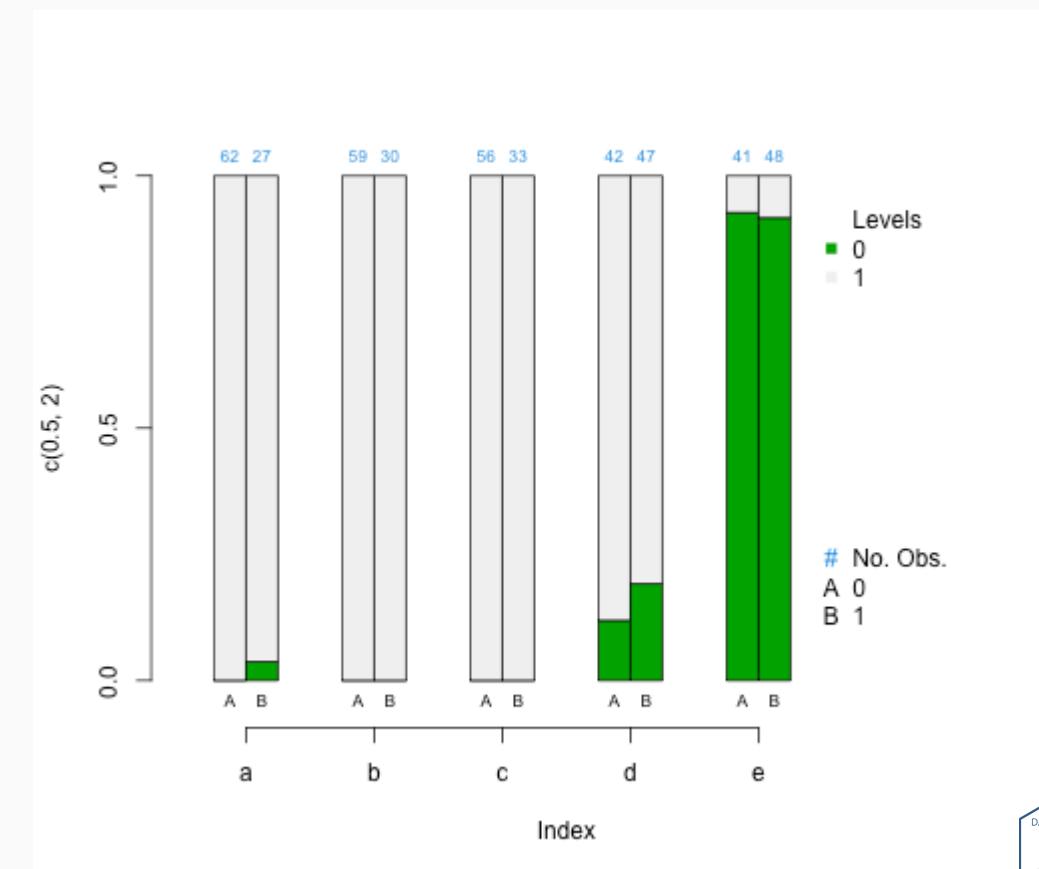


# Checking Balance: Categorical Covariates (cont.)

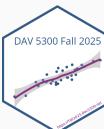
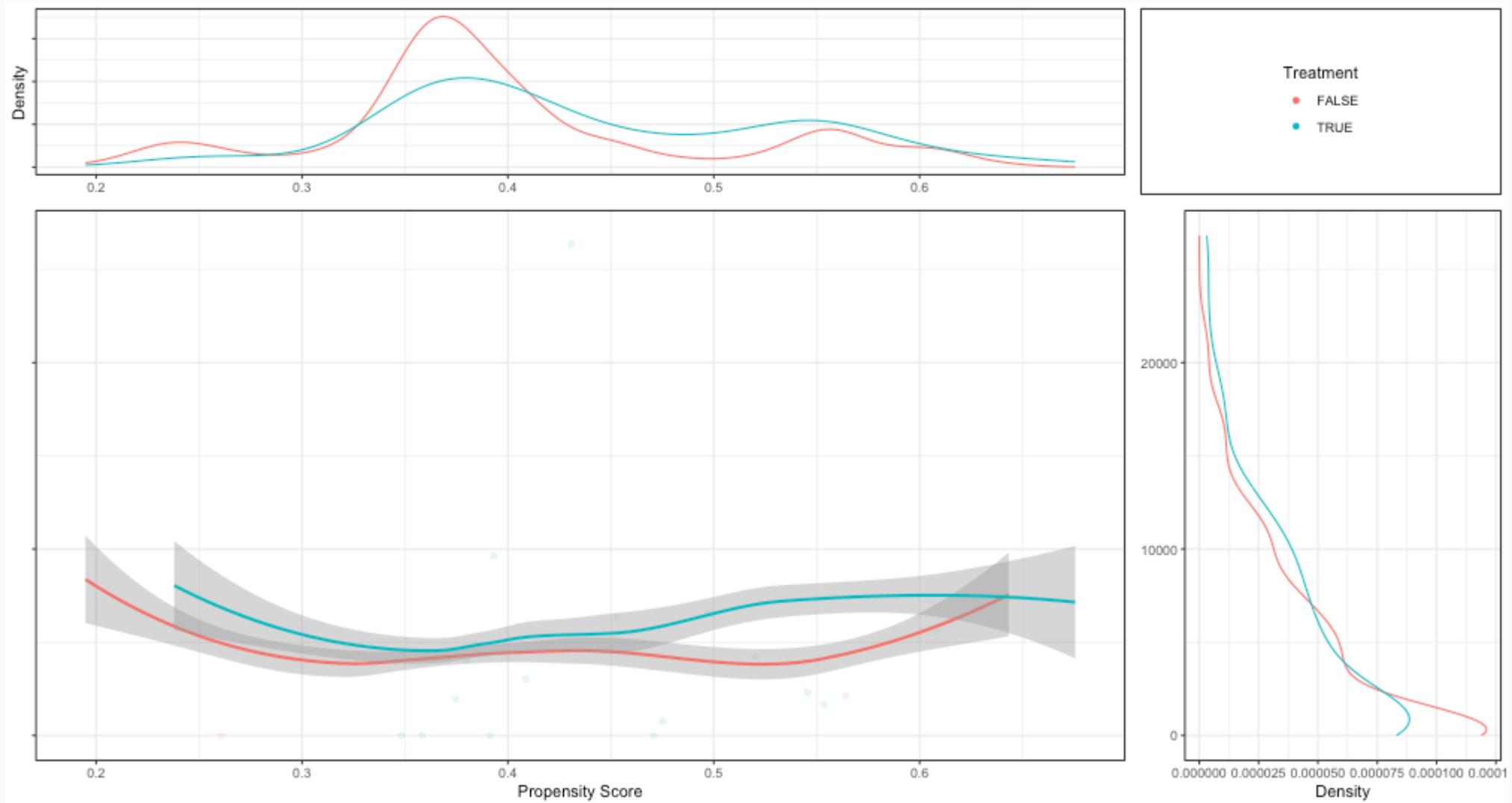
```
cat.psa(lalonde$black, lalonde$treat, strata5)
```



```
cat.psa(lalonde$nodegr, lalonde$treat, strata5)
```

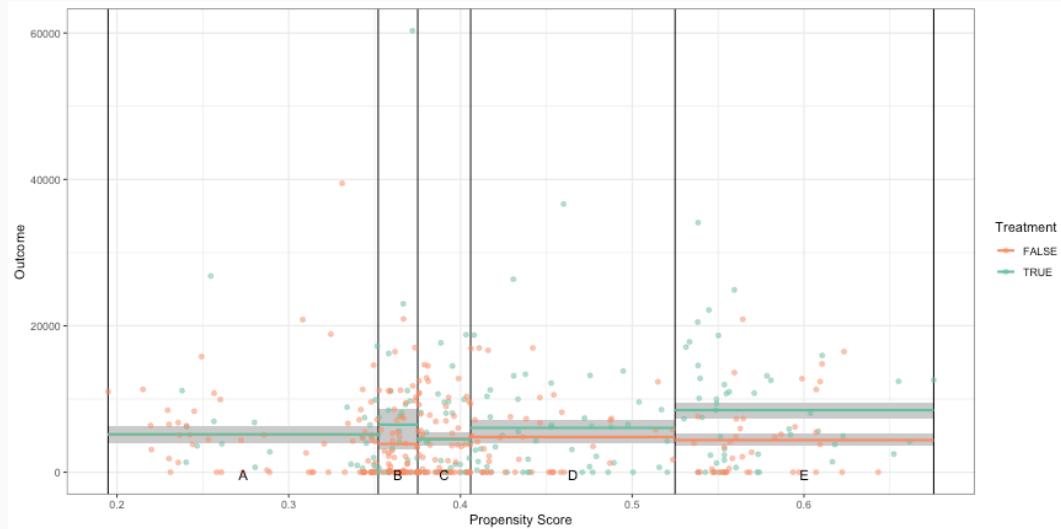


# Loess Regression

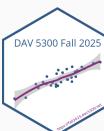
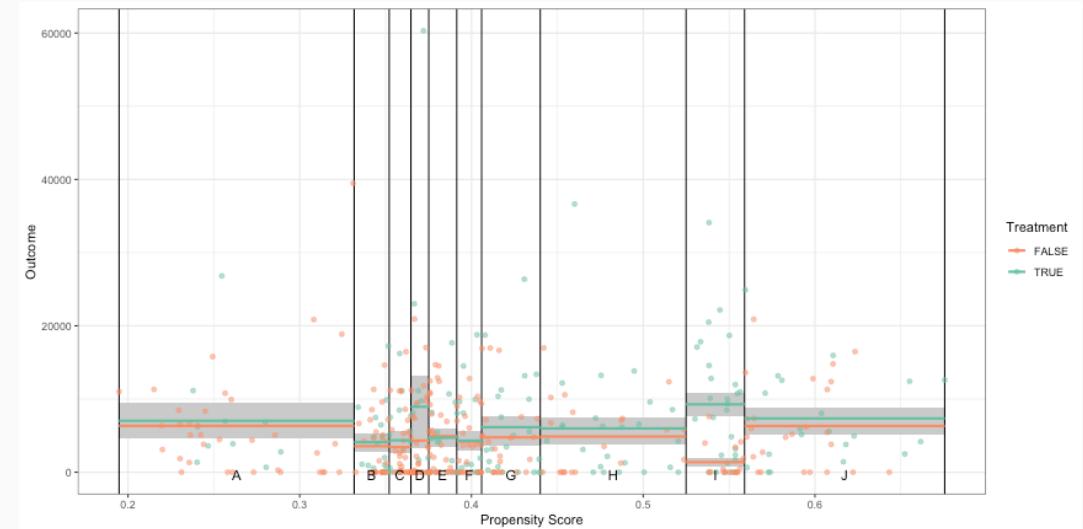


# Stratification

```
psa::stratification_plot(ps = psadf$ps,  
                         treatment = psadf$Tr,  
                         outcome = psadf$Y,  
                         n_strata = 5)
```

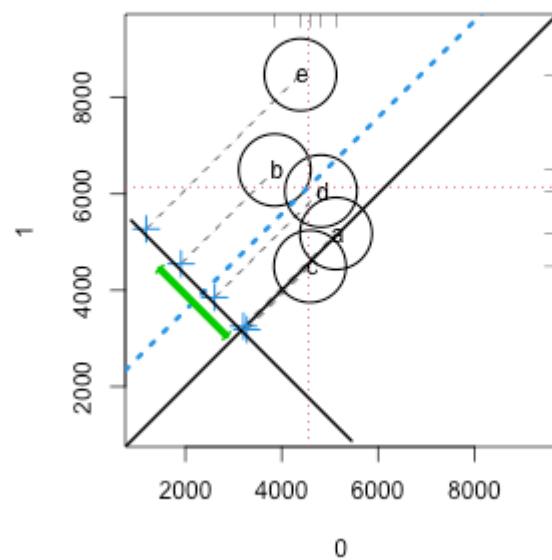


```
psa::stratification_plot(ps = psadf$ps,  
                         treatment = psadf$Tr,  
                         outcome = psadf$Y,  
                         n_strata = 10)
```

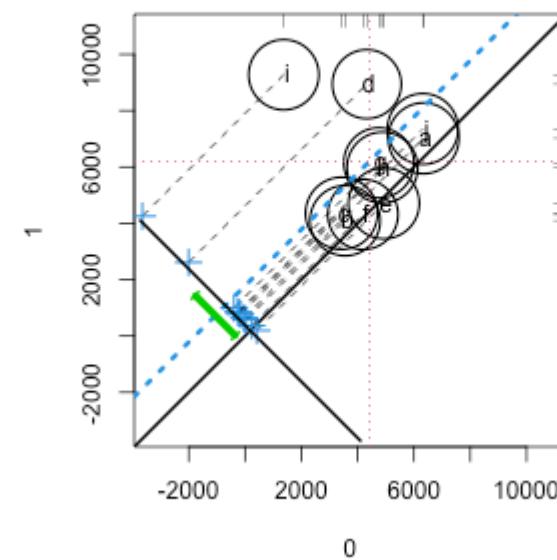


# Stratification (cont.)

```
strata5 <- cut(lalonde$ps,  
                 quantile(lalonde$ps, seq(0, 1, 1/5)),  
                 include.lowest = TRUE,  
                 labels = letters[1:5])  
circ.psa(lalonde$re78, lalonde$treat, strata5)
```



```
strata10 <- cut(lalonde$ps,  
                  quantile(lalonde$ps, seq(0, 1, 1/10)),  
                  include.lowest = TRUE,  
                  labels = letters[1:10])  
circ.psa(lalonde$re78, lalonde$treat, strata10)
```



# Stratification (cont.)

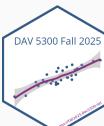
```
## $summary.strata
##   n.0 n.1 means.0  means.1
## a  62  27 5126.493 5178.073
## b  59  30 3855.200 6496.695
## c  56  33 4586.869 4495.076
## d  42  47 4814.028 6059.232
## e  41  48 4387.692 8474.201
##
## $wtd.Mn.0
## [1] 4554.056
##
## $wtd.Mn.1
## [1] 6140.655
##
## $ATE
## [1] 1586.599
##
## $se.wtd
## [1] 693.5067
##
## $approx.t
## [1] 2.287792
##
## $df
## [1] 435
##
## $CI.95
## [1] 223.5584 2949.6395
```

```
## $summary.strata
##   n.0 n.1 means.0  means.1
## a  35  10 6339.437 7019.962
## b  27  17 3554.157 4094.609
## c  31  16 3430.148 4356.532
## d  28  14 4325.792 8942.596
## e  30  15 4932.648 4710.588
## f  26  18 4187.895 4315.483
## g  22  22 4755.015 6148.795
## h  20  25 4878.944 5980.416
## i  16  28 1375.014 9276.448
## j  25  20 6315.806 7351.056
##
## $wtd.Mn.0
## [1] 4414.111
##
## $wtd.Mn.1
## [1] 6195.262
##
## $ATE
## [1] 1781.151
##
## $se.wtd
## [1] 710.5964
##
## $approx.t
## [1] 2.506559
##
## $df
## [1] 425
##
## $CI.95
## [1] 384.4306 3177.8724
```

# Matching

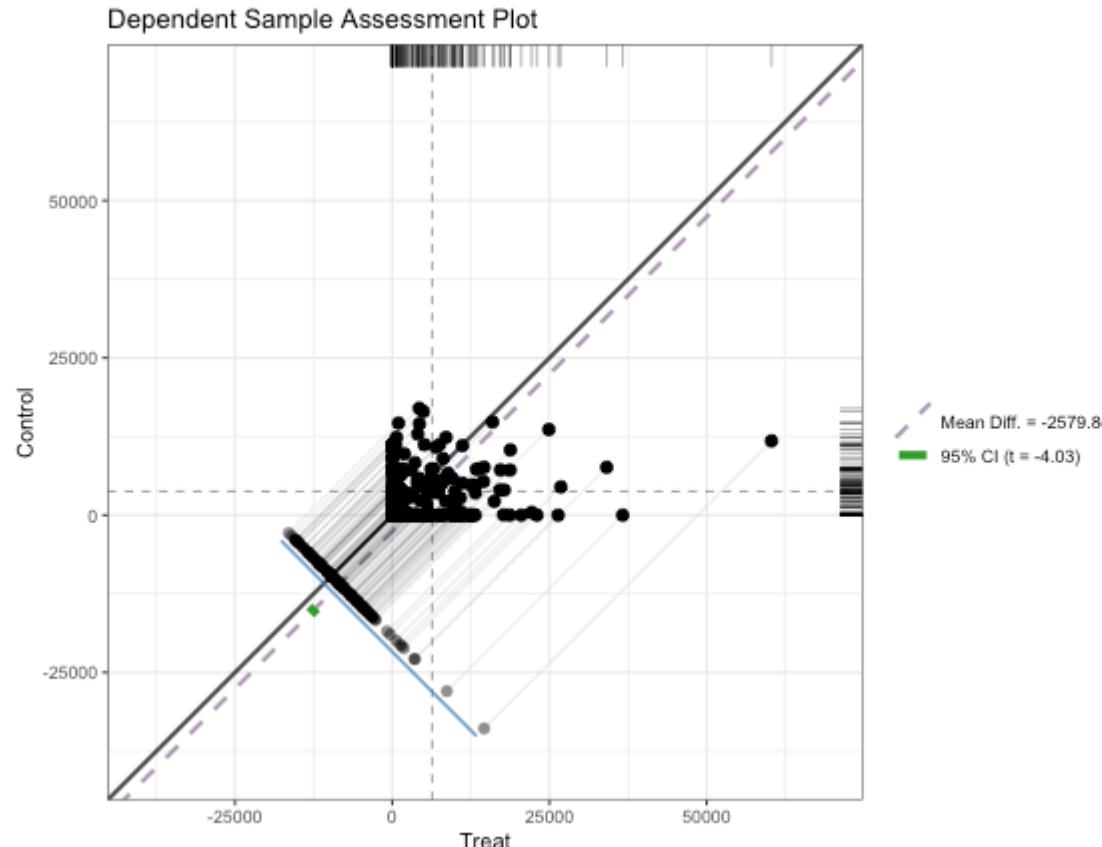
```
rr <- Match(Y = lalonde$re78,
             Tr = lalonde$treat,
             X = lalonde$ps,
             M = 1,
             estimand = 'ATT',
             ties = FALSE)
summary(rr)
```

```
##  
## Estimate... 2579.8  
## SE......... 637.69  
## T-stat..... 4.0456  
## p.val..... 5.2189e-05  
##  
## Original number of observations..... 445  
## Original number of treated obs..... 185  
## Matched number of observations..... 185  
## Matched number of observations (unweighted). 185
```



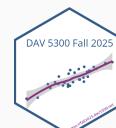
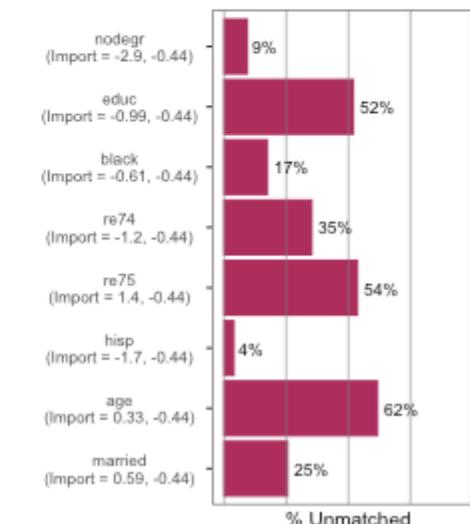
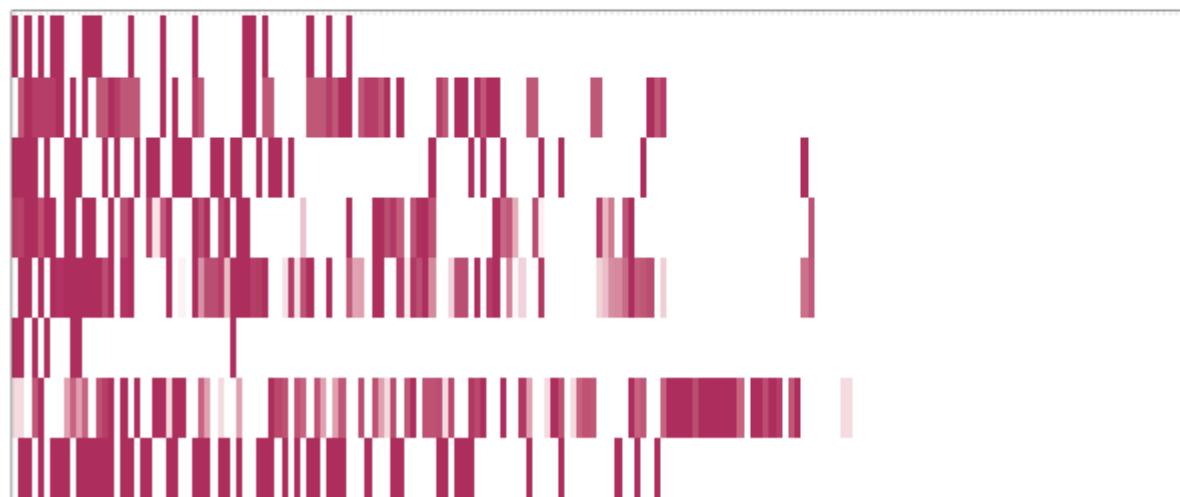
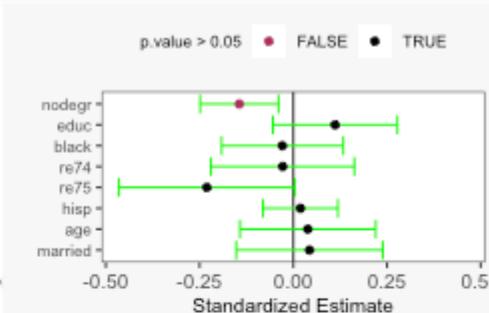
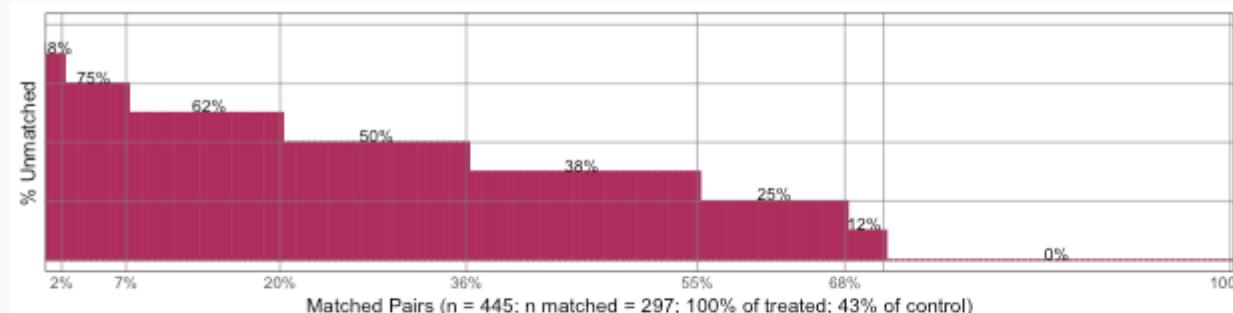
# Visualizing Matching Results

```
matches <- data.frame(Treat = lalonde[rr$index.treated,'re78'],
                      Control = lalonde[rr$index.control,'re78'])
granovagg.ds(matches[,c('Control','Treat')], xlab = 'Treat', ylab = 'Control')
```



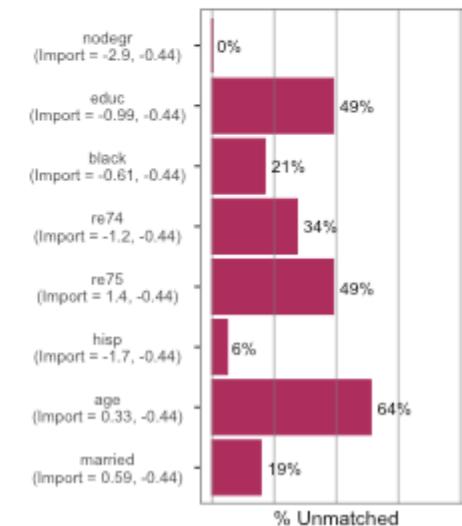
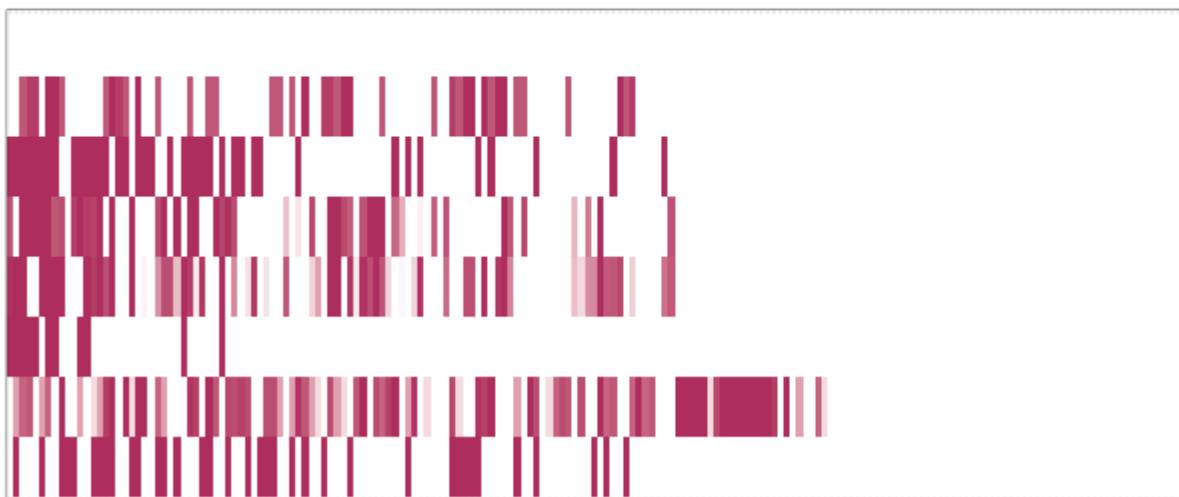
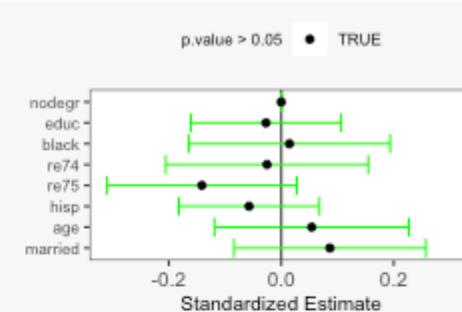
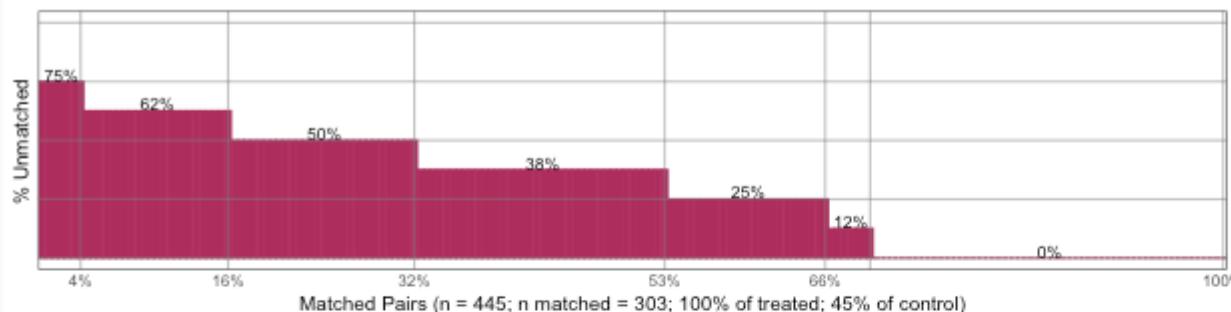
# Balance for Matching

```
psa::MatchBalance(df = lalonde, formu = lalonde.formu,  
                  formu.Y = update.formula(lalonde.formu, re78 ~ .),  
                  M = 1, estimand = 'ATT', ties = FALSE) |> plot()
```



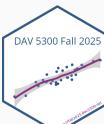
# Balance for Matching (cont.)

```
psa::MatchBalance(df = lalonde, formu = lalonde.formu,  
                  formu.Y = update.formula(lalonde.formu, re78 ~ .),  
                  exact.covs = c('nodegr'),  
                  M = 1, estimand = 'ATT', ties = FALSE) |> plot()
```



# Additional Resources

- Propensity Score R Package and Book: [psa.bryer.org](https://psa.bryer.org)
- PSA talk for the NYC R Meetup Group: [bryer.org/posts/2023-11-14-Intro\\_to\\_PSA.html](https://bryer.org/posts/2023-11-14-Intro_to_PSA.html)
- R Packages
  - `PSAgraphics` - <https://github.com/jbryer/PSAgraphics>
  - `multilevelPSA` - <https://github.com/jbryer/multilevelPSA>
  - `TriMatch` - <https://jbryer.github.io/TriMatch>
  - `PSAboot` - <https://github.com/jbryer/PSAboot>



# One Minute Paper

1. What was the most important thing you learned during this class?
2. What important question remains unanswered for you?



<https://forms.gle/bEk6KegZ1AMvQHNT5>

