DAOE Assignment

MIT Sloan School of Management

Joshua White

2023-11-15

Table of contents

Set up

Load packages

```
suppressPackageStartupMessages({
  library(tidyverse)
  library(DeclareDesign)
  library(quickblock) # for quickblock()
  library(broom) # for tidy()
  library(jpw) # personal package for simple utility funs
})
```

Load data

Note, `educ_cat` loaded as ordered factor and `ideo` loaded as integer to work better with `quickblock`.

```
data <- readr::read_csv(
  "mt_baseline_data.csv",
  col_types = cols(
    educ_cat = col_factor(),
    ideo = col_integer(),
    income_cat = col_factor(),
    race = col_factor()
  )
)
```

Define functions

```
# draw_data function for use with DeclareDesign
draw_data <- function(N) {
  draw <- as.data.frame(data[sample.int(nrow(data), N), ])
  Y_Z_1 <- Y_Z_0 <- Y <- draw$y # SHARP NULL -- No effect.
  block <- unclass(quickblock::quickblock(draw[, c("party", "ideo", "age", "educ_cat")], 8))
  cbind(draw, block, Y_Z_1, Y_Z_0, Y)
}

# function to print tidy diagnosis data frame for nice viewing
print_diagnostics <- function(tidy_diagnostics) {
  tidy_diagnostics %>%
    mutate(
      Estimator = factor(estimator, levels = estimator_labs),
      estimate = paste(round(estimate, 3), jpw::brackets(round(std.error, 3))),
      .keep = "unused"
    ) %>%
    select(Estimator, diagnosand, estimate) %>%
    pivot_wider(
      names_from = diagnosand,
      values_from = estimate
    ) %>%
    arrange(Estimator)
}
```

Set global parameters for analysis

```
N <- 1000
trt <- 1/4
nsims <- 1000
nboot <- 200
estimator_labs <-  c(
  "Difference in means", "Party-adjusted", "Party-adjusted, Lin (2013)",
  "Party-adjusted, as factor", "Covariate-adjusted"
)
design_labs <- c("Bernouilli", "Party blocked", "Covariate blocked")
```

Question 1a

Setup model, inquiry, estimators, measurement and diagnosands with `DeclareDesign`

```
base_design <- declare_model(N = N, handler = draw_data) +
  declare_inquiry(SATE = mean(Y_Z_1 - Y_Z_0))

estimators <- declare_estimator(Y ~ Z, label = estimator_labs[1]) +
  declare_estimator(Y ~ Z + party, label = estimator_labs[2]) +
  declare_estimator(Y ~ Z, covariates = ~party, .method = estimatr::lm_lin, label = estimator_labs[3]) +
  declare_estimator(Y ~ Z + factor(party), label = estimator_labs[4]) +
  declare_estimator(Y ~ Z + factor(block), label = estimator_labs[5])

measurement <- declare_measurement(Y = reveal_outcomes(Y ~ Z))
diagnosands <- declare_diagnosands(
    Bias            = mean(estimate - estimand),
    SD              = sd(estimate), #or sqrt(pop.var(estimate))?
    `Mean CI width` = mean(conf.high - conf.low),
    `CI coverage`   = mean(estimand <= conf.high & estimand >= conf.low)
  )

## Assignment mechanisms
bernoulli_assignment <- declare_assignment(Z = simple_ra(N = N, prob = trt))
party_blocked_assignment <- declare_assignment(Z = block_ra(blocks = party, prob = trt))
covariate_blocked_assignment <- declare_assignment(Z = block_ra(blocks = block, prob = trt))
```

Table 1: Simulations results for null effect with Bernouilli assignment

(a) Values in brackets are standard errors obtained by bootstrap.

Estimator	Bias	SD	Mean CI width	CI coverage
Difference in means	-0.001 (0.005)	0.169 (0.004)	0.652 (0)	0.944 (0.008)
Party-adjusted	0 (0.005)	0.163 (0.003)	0.622 (0.001)	0.947 (0.007)
Party-adjusted, Lin (2013)	0 (0.005)	0.163 (0.003)	0.622 (0.001)	0.947 (0.007)
Party-adjusted, as factor	0 (0.005)	0.16 (0.004)	0.61 (0.001)	0.946 (0.007)
Covariate-adjusted	0.001 (0.005)	0.164 (0.003)	0.614 (0.001)	0.942 (0.007)

Simulations with Bernouilli assignment

```r
bernoulli <- base_design +
  bernoulli_assignment +
  measurement +
  estimators

bernoulli_diagnostics <- diagnose_design(
  bernoulli,
  sims = nsims,
  bootstrap_sims = nboot,
  diagnosands = diagnosands
)

bernoulli_diagnostics_tidy <- tidy(bernoulli_diagnostics)
print_diagnostics(bernoulli_diagnostics_tidy)
```

Table 2: Simulations results for null effect with stratified complete assignment blocked by party

(a) Values in brackets are standard errors obtained by bootstrap.

Estimator	Bias	SD	Mean CI width	CI coverage
Difference in means	0.004 (0.006)	0.157 (0.003)	0.652 (0)	0.955 (0.007)
Party-adjusted	0.004 (0.005)	0.157 (0.003)	0.621 (0)	0.943 (0.008)
Party-adjusted, Lin (2013)	0.004 (0.005)	0.157 (0.003)	0.621 (0)	0.944 (0.008)
Party-adjusted, as factor	0.004 (0.005)	0.157 (0.004)	0.607 (0)	0.941 (0.008)
Covariate-adjusted	0.006 (0.005)	0.153 (0.003)	0.614 (0)	0.954 (0.007)

Simluations with stratified random sampling blocked on party

```
party_stratified <- base_design +
  party_blocked_assignment +
  estimators +
  measurement

party_diagnostics <- diagnose_design(
  party_stratified,
  sims = nsims,
  bootstrap_sims = nboot,
  diagnosands = diagnosands
)

party_diagnostics_tidy <- tidy(party_diagnostics)
print_diagnostics(party_diagnostics_tidy)
```

Table 3: Simulations results for null effect with stratified complete assignment blocked by party, ideology, age and education

(a) Values in brackets are standard errors obtained by bootstrap.

Estimator	Bias	SD	Mean CI width	CI coverage
Difference in means	0.003 (0.005)	0.152 (0.003)	0.652 (0)	0.976 (0.005)
Party-adjusted	0.002 (0.005)	0.149 (0.003)	0.621 (0)	0.966 (0.006)
Party-adjusted, Lin (2013)	0.002 (0.005)	0.149 (0.003)	0.621 (0)	0.967 (0.006)
Party-adjusted, as factor	0.003 (0.005)	0.148 (0.003)	0.609 (0.001)	0.961 (0.007)
Covariate-adjusted	0.005 (0.005)	0.151 (0.003)	0.592 (0)	0.944 (0.008)

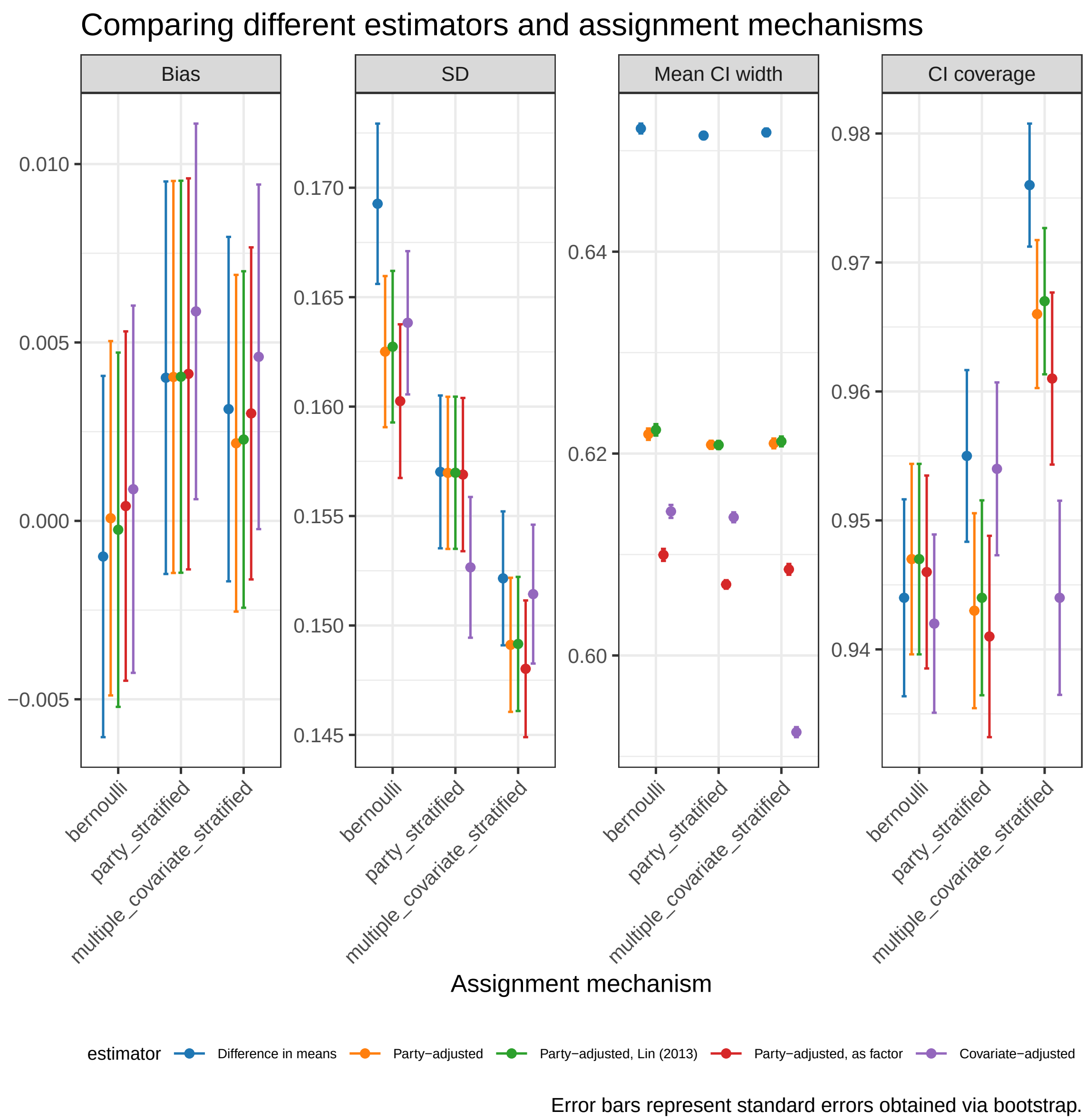
Stratified random sampling blocked on `party`, `ideo`, `age`, and `educ_cat`

```
multiple_covariate_stratified <- base_design +
  covariate_blocked_assignment +
  estimators +
  measurement

multiple_covariates_diagnostics <- diagnose_design(
  multiple_covariate_stratified,
  sims = nsims,
  bootstrap_sims = nboot,
  diagnosands = diagnosands
)

multiple_covariates_diagnostics_tidy <- tidy(multiple_covariates_diagnostics)
print_diagnostics(multiple_covariates_diagnostics_tidy)
```

Plot

```
comparison_data <-
  bind_rows(
    bernoulli_diagnostics_tidy,
    party_diagnostics_tidy,
    multiple_covariates_diagnostics_tidy
  ) %>%
  #order observations
  mutate(
    estimator = factor(estimator, estimator_labs),
    diagnosand = factor(diagnosand, c("Bias", "SD", "Mean CI width", "CI coverage"))
  )

comparison_data %>%
  ggplot(aes(y = estimate, x = design, col = estimator)) +
  geom_point(position = position_dodge(width = 0.6)) +
  geom_errorbar(
    aes(ymin = estimate - std.error, ymax =  estimate + std.error), width = .4,
    position = position_dodge(width = 0.6)
  ) +
```

```r
facet_wrap(~diagnosand, ncol = 4, scales = "free") +
ggsci::scale_color_d3() +
labs(
  x = "Assignment mechanism",
  y = "",
  title = "Comparing different estimators and assignment mechanisms",
  caption = "Error bars represent standard errors obtained via bootstrap."
) +
ggplot2::theme_bw() +
theme(
  plot.title = element_text(size=14),
  axis.text.x = element_text(angle = 45, hjust = 1),
  legend.title = element_text(size=8),
  legend.text = element_text(size=6),
  legend.position = "bottom"
)
```



Discussion

This analysis shows that all estimators are unbiased for all assignment mechanisms. In relation to variance, however, there is benefits to conducting stratified randomization compared to Bernoulli randomization. In particular, it appears that randomization stratifying on multiple covariates results in the lowest variance. Within the randomization scheme, however, the analysis method does not generally have large effects on the variance.

There are also clear differences between the assignment mechanisms and estimators in terms of mean confidence interval width. The general pattern here is that more complex estimators have smaller confidence interval widths, but that the results are similar with different assignment mechanisms. The exception here, is when combining the multiple-covariate-adjusted estimator with the multiple-covariate-stratified randomization. In this case, the combination results in a CI width much smaller than all other estimator/randomization method combinations.

Lastly, CI coverage for all analysis method/randomization combinations were unbiased unbiased (i.e., .95) with the exception of some analysis methods combined with multiple-covariate-stratified randomization. In this case, one should make sure to use the estimator adjusting for these same covariates, as the other estimators resulted in liberally biased CI coverage.

Question 1b

To see how the different assignment mechanisms and analysis choices may have a practical effect on the results of the experiment, I will run simulations to see how power changes under different effect sizes for each different assignment mechanism/analysis combination. Because the outcome variable must be an integer in the range [1, 7], we will consider various treatment effects in which participants in treatment have a probability $p \in \{.1, .25, .5, .75, 1\}$ of increasing the outcome `y` by 1 (if the subject has not already scored the maximum of 7). In our finite sample (the 4000 data points given), these correspond to average treatment effects of approximately {0.07, 0.16, 0.34, 0.5, 0.66}, and given `y`'s standard deviation of 2.27, standardized effect sizes (Cohen's d) of {0.03, 0.07, 0.15, 0.22, 0.29}.

Setup

```
draw_data_with_effect <- function(N, prob) {
  draw <- as.data.frame(data[sample.int(nrow(data), N), ])
  Y_Z_0 <- Y <- draw$y
  Y_Z_1 <- ifelse(Y != 7, Y + 1 * rbinom(N, 1, prob = prob), Y)
  block <- unclass(quickblock(draw[, c("party", "ideo", "age", "educ_cat")], 8))
  cbind(draw, block, Y_Z_1, Y_Z_0, Y)
}

#' Function to simulate power given effect size and assignment mechanism
#'
#' @param ef_probs numeric vector of effect size probabilities, which define,
#'  for each simulation, the probability that treatment increases outcome by 1.
#' @param assignments list of `DeclareDesign` assignments to run simulations with.
#'
power_sims <- function(ef_probs, assignment) {

  results <- list()
  sim_set <- expand_grid(ass = assignment, probs = ef_probs)

  for (i in 1:nrow(sim_set)) {
    base_design <-
      declare_model(
        N = N,
        prob = sim_set$probs[i],
        handler = draw_data_with_effect
      ) +
      declare_inquiry(
        SATE = mean(Y_Z_1 - Y_Z_0)
```

```
      )

    design <- base_design +
      sim_set$ass[[i]] +
      measurement +
      estimators

    diagnostics <- diagnose_design(
      design,
      sims = nsims,
      bootstrap_sims = nboot,
      diagnosands = declare_diagnosands(
        mean_estimand = mean(estimand),
        power = mean(p.value <= 0.05))
    )
    results[[i]] <- diagnostics
  }
  results
}

# Set parameters for simulations
assignments <- c(
  bernoulli = bernoulli_assignment,
  party_blocked = party_blocked_assignment,
  covariate_blocked = covariate_blocked_assignment
)
ef_probs <- c(.1, .25, .5, .75, 1)
```

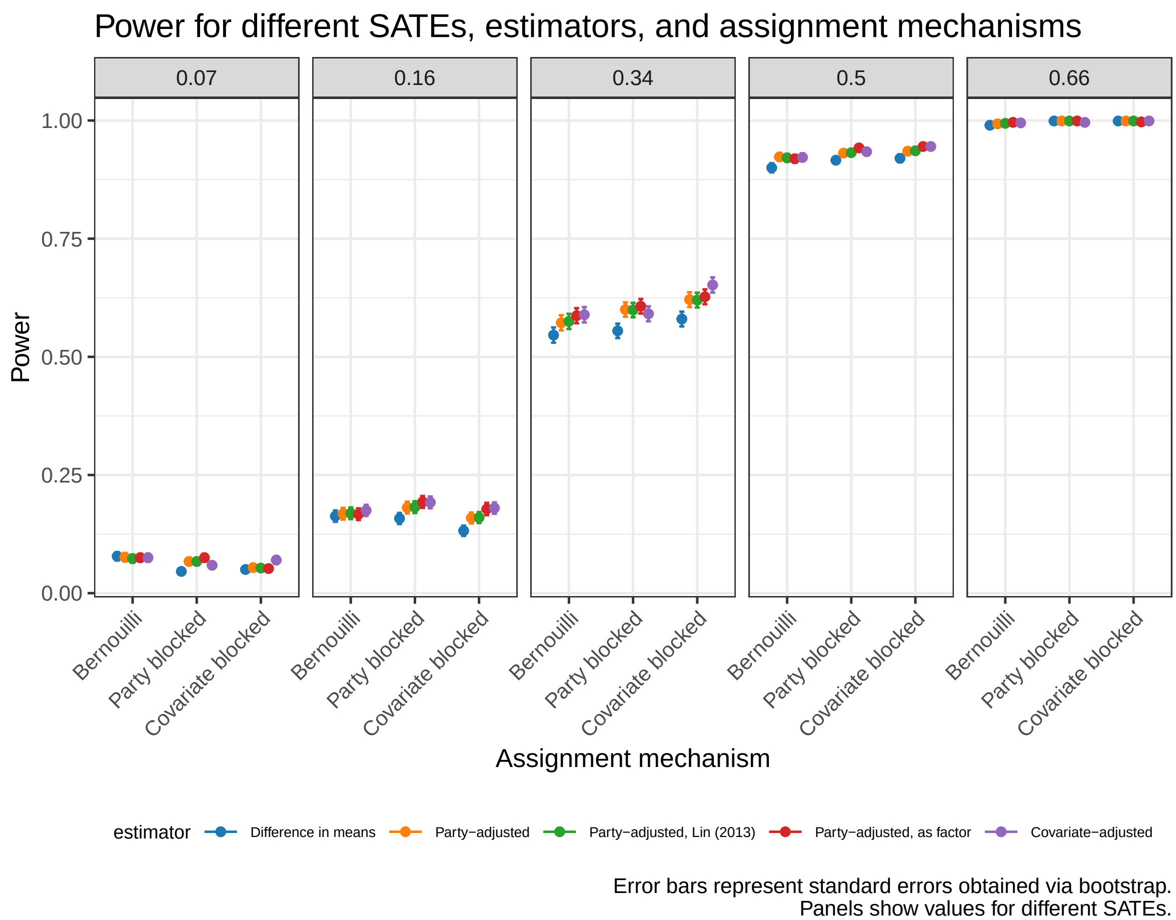
Results

```
power_sims_results <- power_sims(ef_probs, assignments)

power_data <- tibble(
  assignment = rep(design_labs, each = length(ef_probs)),
  SATE = rep(round(SATEs, 2), times = length(assignments)),
  power_sims = power_sims_results %>% map(tidy)
) %>%
  unnest(power_sims) %>%
  filter(diagnosand == "power") %>%
  mutate(
    estimator = factor(estimator, estimator_labs),
    assignment = factor(assignment, design_labs)
  ) %>%
  arrange(assignment, SATE, estimator)
```

Plot

```
power_data %>%
  ggplot(
```

```r
  aes(x = assignment, y = estimate, col = estimator)
) +
geom_point(position = position_dodge(width = 0.6)) +
geom_errorbar(
  aes(ymin = estimate - std.error, ymax = estimate + std.error),
  width = .4, position = position_dodge(width = 0.6)
) +
facet_wrap(~SATE, ncol = 5) +
ggsci::scale_color_d3() +
labs(
  title = "Power for different SATEs, estimators, and assignment mechanisms",
  caption = "Error bars represent standard errors obtained via bootstrap.
    Panels show values for different SATEs.",
  y = "Power",
  x = "Assignment mechanism"
) +
ggplot2::theme_bw() +
theme(
  plot.title = element_text(size=14),
  axis.text.x = element_text(angle = 45, hjust = 1),
  legend.title = element_text(size=8),
  legend.text = element_text(size=6),
  legend.position = "bottom"
)
```



Table

```
power_data %>%
  mutate(
    estimate = paste(round(estimate, 3), jpw::brackets(round(std.error, 3)))
  ) %>%
  select(assignment, SATE, estimator, estimate) %>%
  pivot_wider(
    names_from = SATE,
    values_from = estimate
  )
```

Discussion

These results do not show an obvious pattern for different randomization schemes or analysis choice, particularly when looking across different effect sizes. For small effect sizes, Bernoulli randomization had better or equivalent power to the block-randomization methods, although for larger effect sizes (SATEs of 0.34 and 0.49) it had

Table 4: Simulations results showing power (with alpha = .05) for different SATEs (columns), assignment mechanisms, and estimators

(a) Values in brackets are standard errors obtained by bootstrap

assignment	estimator	0.07	0.16	0.34	0.5	0.66
Bernouilli	Difference in means	0.078 (0.009)	0.163 (0.012)	0.546 (0.016)	0.9 (0.01)	0.99 (0.003)
Bernouilli	Party-adjusted	0.076 (0.009)	0.168 (0.012)	0.572 (0.016)	0.923 (0.008)	0.993 (0.003)
Bernouilli	Party-adjusted, Lin (2013)	0.073 (0.009)	0.169 (0.012)	0.575 (0.016)	0.921 (0.008)	0.994 (0.002)
Bernouilli	Party-adjusted, as factor	0.075 (0.009)	0.167 (0.012)	0.587 (0.016)	0.919 (0.008)	0.996 (0.002)
Bernouilli	Covariate-adjusted	0.075 (0.009)	0.175 (0.012)	0.589 (0.016)	0.922 (0.008)	0.995 (0.002)
Party blocked	Difference in means	0.046 (0.007)	0.158 (0.012)	0.555 (0.015)	0.916 (0.008)	0.999 (0.001)
Party blocked	Party-adjusted	0.067 (0.008)	0.181 (0.012)	0.6 (0.015)	0.931 (0.007)	0.999 (0.001)
Party blocked	Party-adjusted, Lin (2013)	0.067 (0.008)	0.182 (0.012)	0.599 (0.015)	0.932 (0.007)	0.999 (0.001)
Party blocked	Party-adjusted, as factor	0.075 (0.009)	0.193 (0.013)	0.607 (0.015)	0.942 (0.007)	0.999 (0.001)
Party blocked	Covariate-adjusted	0.059 (0.008)	0.192 (0.012)	0.591 (0.016)	0.934 (0.008)	0.996 (0.002)
Covariate blocked	Difference in means	0.05 (0.007)	0.132 (0.011)	0.58 (0.016)	0.92 (0.008)	0.999 (0.001)
Covariate blocked	Party-adjusted	0.054 (0.007)	0.159 (0.011)	0.621 (0.016)	0.935 (0.008)	0.999 (0.001)
Covariate blocked	Party-adjusted, Lin (2013)	0.053 (0.007)	0.16 (0.012)	0.62 (0.016)	0.936 (0.008)	0.999 (0.001)
Covariate blocked	Party-adjusted, as factor	0.052 (0.007)	0.178 (0.013)	0.627 (0.016)	0.945 (0.007)	0.997 (0.002)
Covariate blocked	Covariate-adjusted	0.07 (0.008)	0.18 (0.012)	0.652 (0.016)	0.945 (0.008)	0.999 (0.001)

slightly lower power. When considering how blocked analyses interact with the different analysis methods, adjusting analyses on the same variables upon which randomization was blocked seemd to result in the greatest power.

Overall, this analysis shows that using blocked randomization and analyses may show small benefits to power, although this depends on the size of the underlying effect. In these cases, the most benefit can be gained if analysis is adjusted on the same variables that randomization is stratified by.

Bonus Question

To analyse the effect that conditioning analysis on "imbalances" in covariates between treatment groups in Bernoulli randomization, I will run both the above simulations (first, looking at Bias, SD, mean CI width and CI coverage under the sharp null, and second, looking at power for different effects). For each simulated draw of the data, I will:

1. run a separate t-test (or chi-square test where the variable is not numeric) comparing treatment and control group on each covariate available.
2. adjust for all variables for which $p \leq 0.1$ from step 1 above, using `estimator::lm_robust()`.

I will also calculate the simple difference in means estimator, for comparison puropses.

Setup

First, define some new functions.

```
## draw data functions without quickblock, not necessary, saves compute
draw_data1 <- function(N) {
  draw <- as.data.frame(data[sample.int(nrow(data), N), ])
  Y_Z_1 <- Y_Z_0 <- Y <- draw$y # SHARP NULL -- No effect.
  cbind(draw, Y_Z_1, Y_Z_0, Y)
```

```
}
draw_data1_with_effect <- function(N, prob) {
  draw <- as.data.frame(data[sample.int(nrow(data), N), ])
  Y_Z_0 <- Y <- draw$y
  Y_Z_1 <- ifelse(Y != 7, Y + 1 * rbinom(N, 1, prob = prob), Y)
  cbind(draw, Y_Z_1, Y_Z_0, Y)
}

# OLS Conditional estimates on vars with "imbalances"
cond_on_imbalances <- function(data) {
  vars <- c("political_knowledge", "ideo", "party", "educ_cat", "income_cat", "age", "race")
  ps <- numeric(length(vars))

  # t-test if numeric, chi square if factor
  for(i in seq_along(vars)) {
    y <- data[[vars[i]]]
    if(is.numeric(y)) {
      ps[i] <- t.test(y[data$Z], y[!data$Z])$p.value
    } else {
      ps[i] <- chisq.test(table(y, data$Z))$p.value
    }
  }

  # get results after OLS adjustment based on "imbalances"
  terms_to_adjust_for <- vars[ps <= .1]
  formula <- if(length(terms_to_adjust_for)) {
    paste0("Y ~ Z + ", paste(terms_to_adjust_for, collapse = " + "))
  } else {
    "Y ~ Z"
  }
  out <- broom::tidy(lm_robust(formula(formula), data))
  out[out$term == "Z", ] # return estimate and conf intervals of Z
}

#new power sims function to run power simulations with this enquiry
power_sims1 <- function(ef_probs, assignment) {
  results <- list()
  sim_set <- expand_grid(ass = assignment, probs = ef_probs)

  for (i in 1:nrow(sim_set)) {
    base_design <-
      declare_model(
        N = N,
        prob = sim_set$probs[i],
        handler = draw_data_with_effect
      ) +
      declare_inquiry(
        SATE = mean(Y_Z_1 - Y_Z_0)
      )

    design <- base_design +
      sim_set$ass[[i]] +
```

```r
      measurement +
      declare_estimator(
        handler = label_estimator(cond_on_imbalances), inquiry = "SATE",
        label = "Imbalance Adjusted"
      )  +
      declare_estimator(Y ~ Z, label = "Difference in means", inquiry = "SATE")

    diagnostics <- diagnose_design(
      design,
      sims = nsims,
      bootstrap_sims = nboot,
      diagnosands = declare_diagnosands(
        mean_estimand = mean(estimand),
        power = mean(p.value <= 0.05))
    )
    results[[i]] <- diagnostics
  }
  results
}
```

Bias, SD, CI width and CI coverage under sharp null

```r
design_imbalance_adjustment <- declare_model(N = N, handler = draw_data1) +
  bernoulli_assignment +
  declare_inquiry(SATE = mean(Y_Z_1 - Y_Z_0)) +
  measurement +
  declare_estimator(
    handler = label_estimator(cond_on_imbalances), label = "Imbalance Adjusted", inquiry = "SATE"
  ) +
  declare_estimator(Y ~ Z, label = "Difference in means", inquiry = "SATE")

imbalance_diagnostics <- diagnose_design(
  design_imbalance_adjustment,
  sims = nsims,
  bootstrap_sims = nboot,
  diagnosands = diagnosands
)

tidy(imbalance_diagnostics) %>%
  mutate(
    Estimator = estimator,
    estimate = paste(round(estimate, 3), jpw::brackets(round(std.error, 3))),
    .keep = "unused"
  ) %>%
  select(Estimator, diagnosand, estimate) %>%
  pivot_wider(
    names_from = diagnosand,
    values_from = estimate
  ) %>%
  arrange(Estimator)
```

Table 5: Simulation results comparing adjusting for imbalances -v- simple difference in means

(a) Values in brackets are standard errors obtained by bootstrap

Estimator	Bias	SD	Mean CI width	CI coverage
Difference in means	0.001 (0.005)	0.168 (0.004)	0.652 (0.001)	0.946 (0.007)
Imbalance Adjusted	-0.004 (0.005)	0.15 (0.003)	0.586 (0.001)	0.948 (0.007)

Power for 'conditional-upon-imbalances' estimator for different effect sizes

```
power_sims_results1 <- power_sims1(ef_probs, c(bernoulli = bernoulli_assignment))
power_data1 <- tibble(
  SATE = round(SATEs, 2),
  power_sims = power_sims_results1 %>% map(tidy)
) %>%
  unnest(power_sims) %>%
  filter(diagnosand == "power") %>%
  arrange(SATE, estimator)

# Table
power_data1 %>%
  mutate(
    estimate = paste(round(estimate, 3), jpw::brackets(round(std.error, 3)))
  ) %>%
  select(SATE, estimator, estimate) %>%
  pivot_wider(
    names_from = SATE,
    values_from = estimate
  )
# Plot
power_data1 %>%
  ggplot(
    aes(x = estimator, y = estimate, col = estimator)
  ) +
  geom_point(position = position_dodge(width = 0.6)) +
  geom_errorbar(
    aes(ymin = estimate - std.error, ymax = estimate + std.error),
    width = .4, position = position_dodge(width = 0.6)
  ) +
  facet_wrap(~SATE, ncol = 5) +
  ggsci::scale_color_d3() +
  labs(
    title = "Power for different SATEs, comparing imbalance-adjusting v diff in means",
    caption = "Error bars represent standard errors obtained via bootstrap.
      Panels show values for different SATEs.",
    y = "Power",
    x = ""
  ) +
  ggplot2::theme_bw() +
  theme(
    plot.title = element_text(size=14),
```

```
    legend.title = element_text(size=8),
    legend.text = element_text(size=10),
    legend.position = "bottom",
    axis.text.x = element_blank(),
    axis.ticks.x = element_blank()
  )
```

Discussion

This analysis shows that adjusting for covariate imbalances between treatment and control with an OLS estimator results in, for a null effect, improvements to estimator variance and confidence interval width, compared to the simple difference in means estimator. In addition, the estimator is unbiased and maintains CI coverage $= 1 - \alpha$. When considering different sized underlying effects, there is also a non-trivial increase in power for the covariate-adjusted OLS estimator compared to the difference in means estimator (for effect sizes in which there isn't a floor or ceilling effect). However, this is not to say that such imbalance adjustments should necessarily be advised. According to Imbens and Rubens (2007)[1] "it is easy for the researcher using regression methods to go beyond analyses that are justified by randomization, and end up with analyses that rely on a difficult-to-assess mix of randomization assumptions, modeling assumptions, and large sample approximations." Considerable caution should thus be taken before adopting such an approach.

[1] Athey, S., and Imbens, G. W. (2017), "The Econometrics of Randomized Experiments," in Handbook of Economic Field Experiments, volume 1 of Handbook of Field Experiments, eds. A. V. Banerjee and E. Duflo, pp.73–140

Table 6: Simulations results showing power for different SATEs (columns), and estimators

(a) Values in brackets are standard errors obtained by bootstrap. Alpha = 0.05

estimator	0.07	0.16	0.34	0.5	0.66
Difference in means	0.069 (0.008)	0.151 (0.012)	0.58 (0.016)	0.897 (0.01)	0.992 (0.003)
Imbalance Adjusted	0.064 (0.008)	0.176 (0.012)	0.631 (0.016)	0.943 (0.007)	0.998 (0.001)