% Power Analysis % Sidak Yntiso sgy210@nyu.edu % February 24, 2020

Minimum detectable effects

- The smallest possible effect that would be worth investing the time and money to design and implement an experiment
- Depends upon on the standard error of our estimator ($\sigma_{\hat{\beta}}$), the power of the test (κ) and the confidence level for the test of the null hypothesis (α)
- In the social sciences, we set $\alpha/2 = 0.025, 1 - \kappa = 0.2$
- $t_{\alpha/2} = |z_{0.25}| = 1.96$
- $t_{1-\kappa} = |z_{0.2}| = 0.84$

$$| \beta_{\alpha,\kappa,\sigma_{\hat{\beta}}} |= (t_{\alpha/2} + t_{1-\kappa})\sigma_{\hat{\beta}} = 2.8\sigma_{\hat{\beta}}$$

- For the difference-in-differences estimator,

$$| \beta_{\alpha,\kappa,\sigma_{\hat{\beta}}} |= 2.8\sqrt{\frac{1}{n}(\frac{S_1^2}{p} + \frac{S_0^2}{1-p})},$$

where S_1,S_0 are the variance of potential outcomes.

$$n = \frac{2.8^2(\frac{S_1^2}{p} + \frac{S_0^2}{1-p})}{| \beta_{\alpha,\kappa,\sigma_{\hat{\beta}}} |^2}$$

- This is the sample size needed to detect an effect at least as small as $| \beta_{\alpha,\kappa,\sigma_{\hat{\beta}}} |$, with κ power given a test at the α significance level
- We usually standardize effect sizes relative to the control group standard deviation $\Delta = \beta/S_0$

$$n = \frac{2.8^2(\frac{\psi}{p} + \frac{1}{1-p})}{\Delta^2},$$

where $\psi = S_1/S_0$

Design Effects

- The "design effect" for a statistic Z and design δ is,

$$D^2(Z,\delta) = \frac{\sigma^2_{\delta,Z}}{\sigma^2_{S,Z}} = \frac{\text{Variance of Z under design } \delta}{\text{Variance of Z under S}},$$

- where S is a simple random experiment.
- By implication, the effective sample size is proportional to the variance of the design,

$$\frac{n_{Z,\delta}}{n_{SI,\delta}} = D^2(Z,\delta) = \frac{\sigma^2_{\delta,Z}}{\sigma^2_{S,Z}}$$
$$n_{Z,\delta} = D^2(Z,\delta) \times n_{S,\delta}$$

- We have already discussed precision improvement from two design choices - covariate adjustment and block randomization.

- Consider the following regression model

$$Y_i = \alpha + \delta D_i + \gamma X_i + \lambda D_i X_i + \epsilon_i$$

- Covariate adjustment reduces variation in parameter estimates that can arise from variation in covariates. The design effect is

$$\begin{aligned} D^2(\hat{\beta}, covariates) &= \frac{Var_{\hat{\beta}_c}}{Var_{\hat{\beta}}},\ where \\ Var_{\hat{\beta}_c} &= \frac{Var[Y_{1i}]}{N_1} + \frac{Var[Y_{0i}]}{N_0} \\ &= \frac{(\gamma+\lambda)^2\sigma_x^2 + \sigma_e^2}{N_1} + \frac{\gamma^2\sigma_x^2 + \sigma_e^2}{N_0} \end{aligned}$$

  where γ, λ are the main and interacted parameter estimates, N_1, N_0 are the number of units in treatment and control, σ_x, σ_e are the variance of X and error respectively.

- Block randomization similarly improves precision by reducing the variance of potential outcomes within treatment groups (here blocks). The expression for the design effect under block randomization is:

$$\begin{aligned} D^2(\hat{\beta}, block) &= \frac{Var_{\hat{\beta}_b}}{Var_{\hat{\beta}}},\ where \\ Var_{\hat{\beta}_b} &= \sum_{b=1}^{B} \frac{N_b se(\hat{\beta}_b)}{N se(\hat{\beta})} \end{aligned}$$

Example

```r
rm(list=ls())
library(dplyr)
```

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

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

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

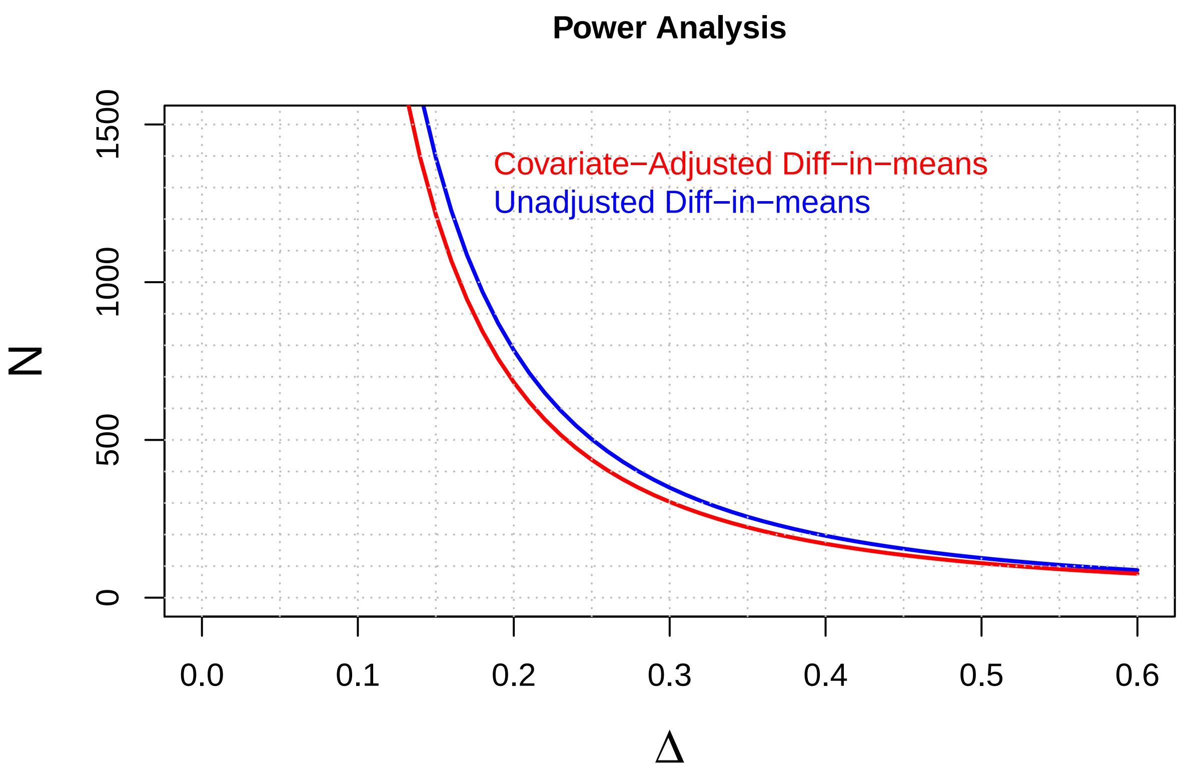
```r
library(haven)
nclass <- 5
nstudent <- 25
Eff <- 5
EffSD <- 3
# Simulate data
set.seed(1977)
Yr1ClassType <- rep(c(1,0),nclass*nstudent)
```

```r
Yr2ClassType <- sample(Yr1ClassType,replace=FALSE)
Yr1Score <- rnorm(2*nclass*nstudent,76+Yr1ClassType*5,9)
# Fixed margins randomization
Trt  <- sample(Yr1ClassType,replace=FALSE)
# There is an independent effect of class type in each year
# Variance is different across class types in year 2
CtlOutcome <- rnorm(2*nclass*nstudent,Yr1Score+Yr2ClassType*3,9-Yr2ClassType*4)
# Treatment effect is random, but with expectation Eff
Yr2Obs <- CtlOutcome + Trt * rnorm(2*nclass*nstudent,Eff,EffSD)


##### ncomp function #####
# sample size function for MDE, under complete randomization:
nsamp = function(alpha, k, psi, D, s0, p = 1/2){
  ta = qnorm(1-alpha/2); tk = qnorm(1-k)
  n = (ta + tk)^2 * (psi^2 / p + 1/(1-p)) / D^2
  n
}

##### d^2 function ######
# design effect = d^2(Bhat, block)
# Y: vector of outcomes
# B: vector of covariates
# TC: treatment/control vector
design_cov = function(Y,X,TC,psi){
    N = length(Y); N0 = length(Y[TC==0]); N1 = length(Y[TC==1])
    s0 = sd(Y[TC==0]); s1 = s0*psi
    m1 = estimatr::lm_robust(Y~TC*X)
    gamma = summary(m1)$coefficients[3,1]
    lambda = summary(m1)$coefficients[4,1]
    sigmax2 = var(X)
    #sigmae2 = var((Y- m1$fitted.values))
    dsum = (((gamma + lambda)^2*sigmax2 + m1$res_var)/N1 + ((lambda)^2*sigmax2 + m1$res_var)/N0)/
      ((s1^2/N1 + s0^2/N0))
    return(dsum)
}
# check this function works
design_cov(Yr2Obs,Yr1Score, Trt,1)
```

```
## [1] 0.8704079
```

```r
##### calculating n values ######
yvar = 'Yr2Obs'
xvar = 'Yr1Score'
dvar = 'Trt'

# set the parameters, and ranges of psi and delta
a = .05; kval = .8
dvals = seq(0,.6,.01)

# matrices to store n values
ncovvals = matrix(rep(NA,length(dvals)),nrow = length(dvals))
ncompvals = matrix(rep(NA,length(dvals)),nrow = length(dvals))
```

```
# iterate over psi values and delta values
  for(i in 1:length(dvals)){
    d = dvals[i]
    # get n for complete randomization
    ncomp = nsamp(alpha = 0.05, k = 0.2, psi=1, D = d)
    # multiply ncomp by the design effect
    ncov = ncomp*design_cov(Yr2Obs,Yr1Score, Trt,1)
    ncovvals[i,] = ncov
    ncompvals[i,] = ncomp
  }


##### ncov vs ncomp plot ####
plot(dvals, ncompvals[,1], type='l',col = "blue",lwd=2,ylab = 'N',
     ylim = c(0,1500),xlab = expression(Delta), cex.lab = 1.5,
      main = 'Power Analysis',cex.main=1)
lines(dvals,ncovvals[,1], col="red",lwd=2)
abline(h=seq(0,1500,100),lty='dotted',col='grey')
abline(v = seq(0,1,.05),lty='dotted',col='grey')
legend(x = 0.15, y = 1500,bty='n'
       ,legend = c("Covariate-Adjusted Diff-in-means","Unadjusted Diff-in-means")
       ,y.intersp=1
       ,text.col=c("red","blue")
)
```



```
##If interested, add another layer of the for loop which iterates over candidate values psi - 0.5,1,1.5
```

Simulation Exercise

Often the analytical approach has limited value for complex experimental designs. This exercise asks you to use Monte Carlo simulations to recover the sample size needed to detect an MDE of $\Delta = 0.2$.

Part A

Set up your simulation.

- Create a vector `possible.ns` of size 20 that lists potential sample sizes (100,200,. . . ,2000).
- create empty vectors `powers` and `powers.cov` of size 20 to store the results of the simulation exercise.

Part B

For each potential sample size N

- generate a covariate, X, that is binomially distributed variable, $X \sim Bin(N, 0.5)$
- create empty vectors `significant.experiments.cov` and `significant.experiments` of size S=100 to store the results of the simulation exercise.

Part B

For each iteration of the simulation (S=100 iterations)

- generate a treatment variable D, that is binomially distributed variable, $D \sim Bin(N, 0.5)$
- generate both potential outcomes: $Y_i(0) = \tau_x X + \epsilon$, where the effect of X is $\tau_x = 1$ and the error is $\epsilon \sim \mathcal{N}(0, 1)$. $Y_i(1) = Y_i(0) + \tau_D D$, where the treatment effect is $\tau_D = 0.2$
- run a regression for the adjusted and unadjusted treatment effect of D
- store an indicator of whether the unadjusted difference-in-means estimate was significant (whether the p-value of the unadjusted model ≤ 0.05) in the vector `significant.experiments`. Also store an indicator for whether the adjusted difference-in-means estimate (the p-value of the adjusted model ≤ 0.05) in the vector `significant.experiments.cov`

For each potential sample size, store the power (mean number of significant experiments) in the `powers` vector

Part D

Plot the power (adjusted and unadjusted difference-in-means estimates) for each value of the potential sample size.

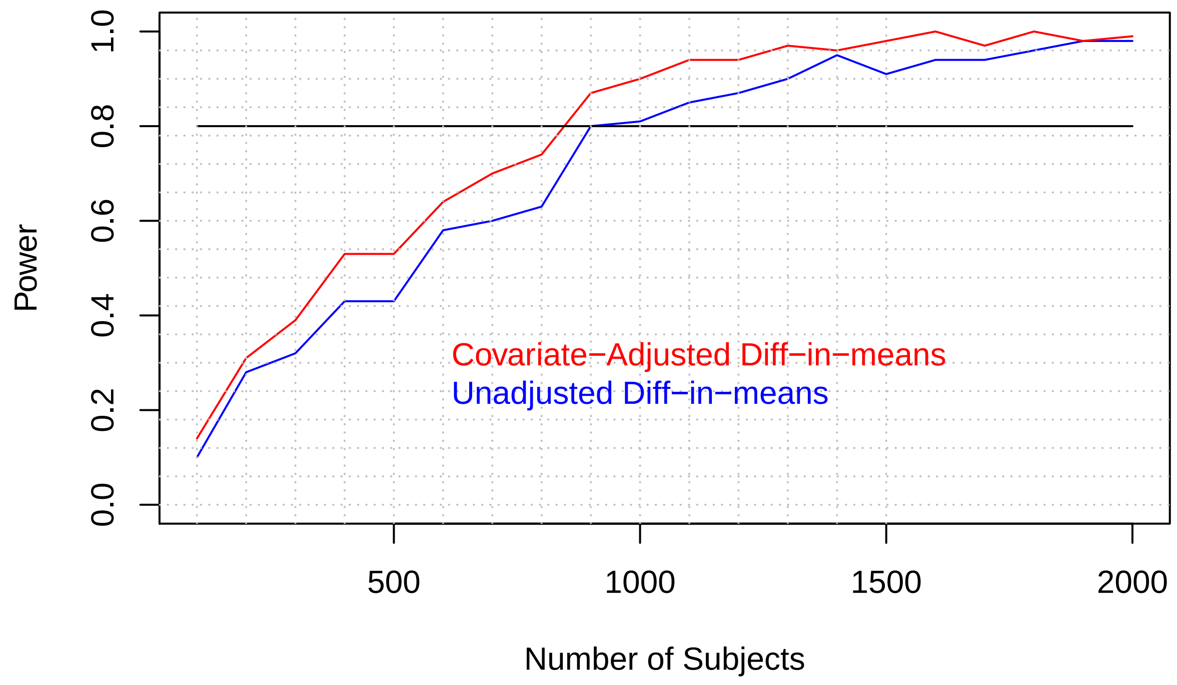
Solutions

```
rm(list=ls())
possible.ns <- seq(from=100, to=2000, by=100)
alpha <- 0.05
sims <- 100
```

```r
powers <- powers.cov <- rep(NA, length(possible.ns))        # Need a second empty vector


#Power to detect effects of size tau,
for (j in 1:length(possible.ns)){
  N <- possible.ns[j]
  significant.experiments <- significant.experiments.cov <- rep(NA, sims)

  X <- rbinom(n=N, size = 1, prob=.5)

  effectX <- 1 # Hypothesize the effect" of X
  tau <- 0.2     # Hypothesize the effect" of treatment

  for (i in 1:sims){
    #### Repeat and store randomizations within simulation
    Y0 <- effectX*X + rnorm(n=N, mean=0, sd=1)

    Y1 <- Y0 + tau

    D.sim <- rbinom(n=N, size=1, prob=.5) # Random assignment of each individual
    Y.sim <- Y1*D.sim + Y0*(1-D.sim) # Reveal outcomes according to assignment

    fit.sim <-  estimatr::lm_robust(Y.sim ~ D.sim)
    p.values <- summary(fit.sim)$coefficients[2,4]
    significant.experiments[i] <- (p.values <= alpha)

    fit.sim.cov <-  estimatr::lm_robust(Y.sim ~ D.sim + X)
    p.values.cov <- summary(fit.sim.cov)$coefficients[2,4]
    significant.experiments.cov[i] <- (p.values.cov <= alpha)
  }
  print(j)
  powers[j] <- mean(significant.experiments)
  powers.cov[j] <- mean(significant.experiments.cov)
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
## [1] 11
## [1] 12
## [1] 13
## [1] 14
## [1] 15
## [1] 16
## [1] 17
## [1] 18
## [1] 19
## [1] 20
```

```
plot(possible.ns, powers, ylim=c(0,1),xlab="Number of Subjects",ylab="Power",col="blue",type="l")
lines(possible.ns, powers.cov, col="red",type="l")
lines(possible.ns, rep(0.8,length(possible.ns)), col="black")
abline(v=seq(0,1500,100),lty='dotted',col='grey')
abline(h = seq(0,1,.06),lty='dotted',col='grey')
legend(y = 0.4, x = 500,bty='n'
       ,legend = c("Covariate-Adjusted Diff-in-means","Unadjusted Diff-in-means")
       ,y.intersp=1
       ,text.col=c("red","blue")
)
```



```
dev.off()
```

```
## null device
##           1
```