Sensitivity Analysis

Jake Bowers, Ben Hansen & Tom Leavitt

August 10, 2016

Contents

1 Introduction

Thus far this course has considered how to make *valid* inferences conditional on the premise of no imbalances on unobserved covariates of which treatment assignment is a function. But what about the soundness of such inferences? Is the premise of no imbalances on unobserved covariates true? Ultimately, that is a question we cannot answer since it requires data that is not available; however, we can consider *hypothetical* scenarios of confounding and then assess the extent to which such scenarios would alter our inferences.

When considering overt bias, Rosenbaum (2002) remains agnostic about the functional form, $\lambda(\cdot)$, that links $\mathbf{x}_i$ to π_i. After all, if $\mathbf{x}_i = \mathbf{x}_j$, then, irrespective of the functional form of $\lambda(\cdot)$, $\lambda(\mathbf{x}_i) = \lambda(\mathbf{x}_j) = \pi_i = \pi_j$.

In order to avoid making any functional form assumptions about how baseline covariates relate to treatment assignment probabilities, then one must *exactly* match treated and control units on baseline covariates. Therefore, the sensitivity analyses today consider *hidden bias* due to an unobserved confounder and due to residual imbalances on unobserved covariates.

1.1 Hidden Bias Due to an Unobserved Confounder

Let's define the *treatment odds ratio* as:

$$\frac{\left(\frac{\pi_i}{1-\pi_i}\right)}{\left(\frac{\pi_j}{1-\pi_j}\right)} \; \forall \; i, j \text{ with } \mathbf{x}_i = \mathbf{x}_j$$

$$\Longrightarrow$$

$$\frac{\pi_i(1-\pi_j)}{\pi_j(1-\pi_i)} \; \forall \; i, j \text{ with } \mathbf{x}_i = \mathbf{x}_j. \tag{1}$$

Rosenbaum (2002) proves that the treatment odds ratio defined in (1) for units i and j implies the following model, which consists of (1) a logistic functional form that links treatment odds, $\frac{\pi_i}{(1-\pi_i)}$, to the covariates $(\mathbf{x}_i, u_i)$; and (2) a constraint on u_i:

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \kappa(\mathbf{x}_i) + \gamma u_i, \tag{2}$$

where $\kappa(\cdot)$ is an unknown function and γ is an unknown parameter.

Remember:

A logarithm is simply the power to which a number must be raised in order to get some other number. In this case we're dealing with natural logarithms. Thus, we can read $\log\left(\frac{\pi_i}{1-\pi_i}\right)$ as asking: e to the power of what gives us $\left(\frac{\pi_i}{1-\pi_i}\right)$? And the answer is e to the power of $\kappa(\mathbf{x}_i)+\gamma u_i$. If $\mathbf{x}_i = \mathbf{x}_j$, then $\log\left(\frac{\pi_i}{1-\pi_i}\right) = \gamma u_i$, which means that $\mathrm{e}^{\gamma u_i} = \left(\frac{\pi_i}{1-\pi_i}\right)$.

Rosenbaum (2002) further shows that if there is indeed a logistic functional form that links $u \in [0,1]$ to Γ, such that $\mathrm{e}^{\gamma} = \Gamma$, then

$$\frac{\pi_i(1-\pi_j)}{\pi_j(1-\pi_i)} = \mathrm{e}^{\gamma(u_i-u_j)} \text{ if } \mathbf{x}_i = \mathbf{x}_j. \tag{3}$$

The minimum and maximum possible value for $u_i - u_j$ are -1 and 1, respectively, which implies that, if the functional form that links an unobserved covariate, u_i, to the treatment odds, $\frac{\pi_i}{1-\pi_i}$, is indeed logistic, then for any fixed γ the upper and lower bounds on 1 are:

$$\frac{1}{\mathrm{e}^{\gamma}} \leq \frac{\pi_i(1-\pi_j)}{\pi_j(1-\pi_i)} \leq \mathrm{e}^{\gamma}. \tag{4}$$

Since $\mathrm{e}^{\gamma} = \Gamma$, then we can express 4 as 1 by substituting $\frac{1}{\Gamma}$ for $\mathrm{e}^{-\gamma}$ and Γ for e^{γ}.

Thus, in conclusion we're left with:

$$\frac{1}{\Gamma} \leq \frac{\pi_i(1-\pi_j)}{\pi_j(1-\pi_i)} \leq \Gamma \;\forall\; i, j \text{ with } \mathbf{x}_i = \mathbf{x}_j \tag{5}$$

For any specific $(\gamma, \mathbf{u})$, there is a distribution of treatment assignments $\mathbf{Z}$ on Ω. If $(\gamma, \mathbf{u})$ were known, then $Pr\left(\mathbf{Z} = \mathbf{z} \in \Omega\right)$ would also be known and one could perform permutation inference. But, since $(\gamma, \mathbf{u})$ is *not* known, a sensitivity analysis illustrates the sensitivity of inferences to a range of assumptions about $(\gamma, \mathbf{u})$.

1.2 Sensitivity Analysis with Matched Sets Design

Rosenbaum (2015) offers an `[R]` package for the implementation of sensitivity analyses.

```r
load(url("http://jakebowers.org/Matching/meddat.rda"))

meddat$HomRate03 <- with(meddat, (HomCount2003/Pop2003) * 1000)
meddat$HomRate08 <- with(meddat, (HomCount2008/Pop2008) * 1000)

load("fm4.RData")

meddat %<>% mutate(fm4 = fm4,
                   HomRate0803 = HomRate08 - HomRate03) %>%
  filter(!is.na(fm4))

meddat %<>% mutate(probs = unsplit(value = lapply(split(x = nhTrt,
                                                        f = fm4),
                                                  function(x) {sum(x)/length(x)}),
                                   f = fm4))

obs_ate <- coef(lm(HomRate0803 ~ nhTrt + fm4,
           data = meddat))[["nhTrt"]]

obs_ate


## [1] -0.5981382


new_block_experiment <- function(z,
                                 y,
                                 s){

  Z <- unsplit(lapply(split(z, s), sample), s)

  obs_ATE <- coef(lm(y ~ Z + s))[["Z"]]

  return(obs_ATE)

}

null_dist <- replicate(1000, new_block_experiment(z = meddat$nhTrt,
                                                  y = meddat$HomRate0803,
                                                  s = meddat$fm4))

p_value_lower <- mean(null_dist <= obs_ate)

p_value_lower


## [1] 0.002


p_value_two_sided <- mean(abs(null_dist) >= abs(obs_ate))

p_value_two_sided
```

```
## [1] 0.002
```

Question for Students:

- Interpret the two p-values above.

Now let's perform a sensitivity analysis using the package developed by Rosenbaum (2015).

```
meddat %<>% select(nhTrt,
                   HomRate0803,
                   fm4,
                   probs) %>%
  arrange(fm4,
          nhTrt)

reshape_sensitivity <- function(.data,
                                .z,
                                .y,
                                .fm){

  suppressMessages(stopifnot(require(dplyr, quietly = TRUE)))
  suppressMessages(stopifnot(require(magrittr, quietly = TRUE)))

  num_cols <- max(table(meddat$fm4))

  reshaped <- lapply(X = split(.y, .fm),
                     FUN = function(x){

                       return(c(x,
                                rep(x = NA,
                                    times = max(num_cols - length(x),
                                                0))))

                       })

  reshaped_df <- data.frame(t(simplify2array(reshaped)))

  return(reshaped_df)

  }

meddat_reshaped <- reshape_sensitivity(.data = meddat,
                    .z = meddat$nhTrt,
                    .y = meddat$HomRate0803,
                    .fm = meddat$fm4) %>%
  rename(yt = X1,
         yc1 = X2,
         yc2 = X3,
```

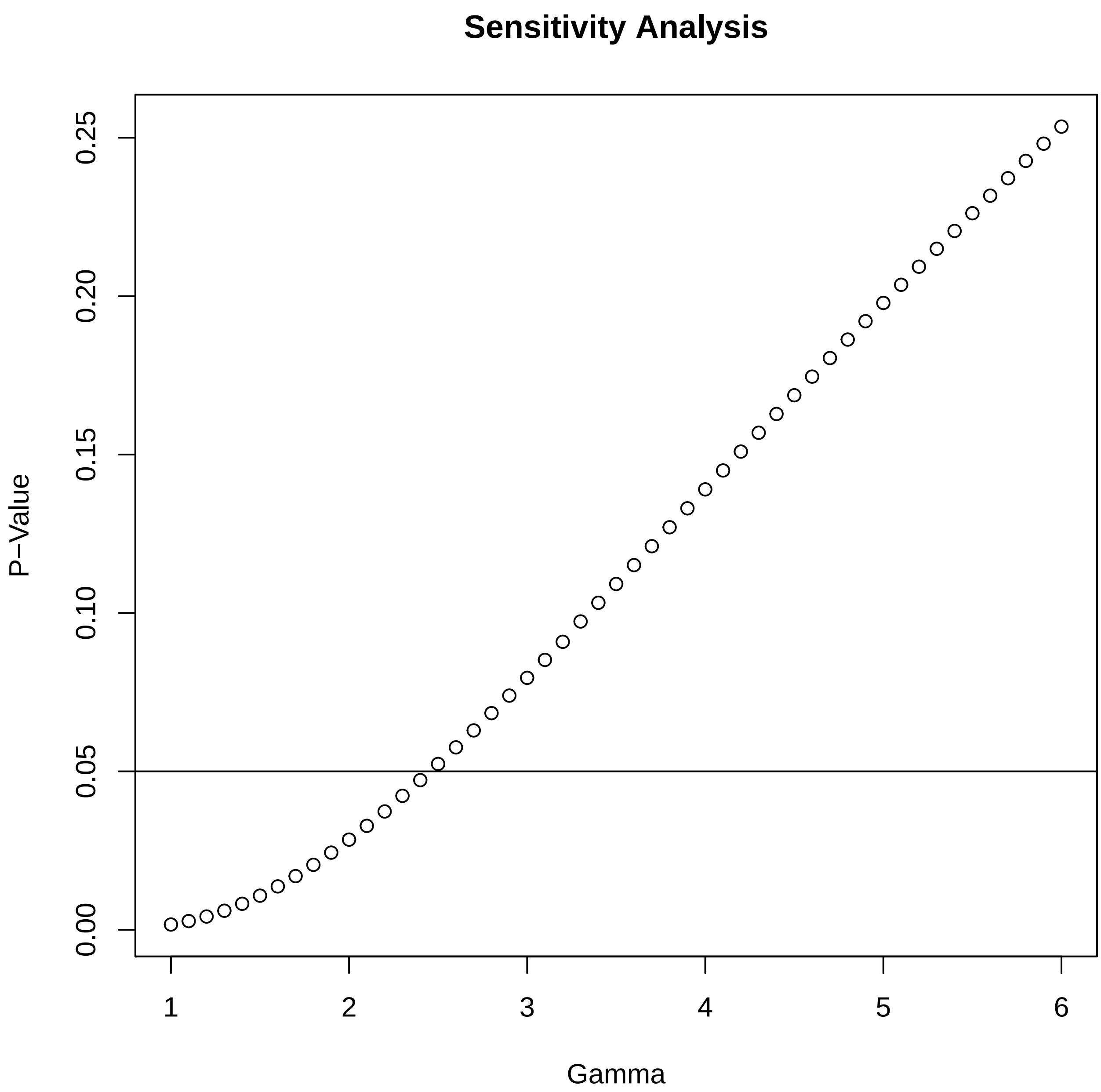
```r
         yc3 = X4,
         yc4 = X5,
         yc5 = X6)

gammas <- seq(from = 1,
              to = 6,
              by = 0.1)

sens_results <- sapply(X = gammas,
                       FUN = function(g) {

                         c(gamma = g,
                           senmv(meddat_reshaped,
                                 method = "t",
                                 gamma = g))
                         })

sens_results
```

```
##             [,1]         [,2]        [,3]        [,4]        [,5]
## gamma       1            1.1         1.2         1.3         1.4
## pval        0.001652875  0.002737169 0.004180985 0.005999901 0.008196182
## deviate     2.937777     2.77771     2.637094    2.51215     2.40006
## statistic   0.6305185    0.6305185   0.6305185   0.6305185   0.6305185
## expectation 2.312965e-18 0.03499982  0.06681783  0.09586906  0.1224994
## variance    0.04606363   0.0459639   0.04569261  0.04529467  0.0448039
##             [,6]       [,7]       [,8]       [,9]       [,10]
## gamma       1.5        1.6        1.7        1.8        1.9
## pval        0.01076179 0.01368123 0.01693401 0.02049654 0.0243437
## deviate     2.298672   2.206308   2.12164    2.0436     1.971319
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.1469992  0.1696145  0.1905546  0.2099989  0.2281022
## variance    0.04424597 0.04364042 0.04300224 0.04234292 0.04167132
##             [,11]      [,12]      [,13]      [,14]      [,15]
## gamma       2          2.1        2.2        2.3        2.4
## pval        0.02844987 0.03278976 0.03733894 0.04227118 0.047226
## deviate     1.90408    1.841287   1.782438   1.724917   1.672367
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.2449987  0.2608051  0.2756236  0.2895941  0.3027583
## variance    0.04099425 0.04031694 0.03964341 0.03906426 0.03841045
##             [,16]      [,17]      [,18]      [,19]      [,20]
## gamma       2.5        2.6        2.7        2.8        2.9
## pval        0.05233184 0.0575698  0.06292244 0.06837373 0.07390902
## deviate     1.622653   1.575506   1.530695   1.488013   1.447282
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.3151758  0.3269086  0.3380117  0.3485347  0.358522
## variance    0.03776717 0.03713571 0.03651698 0.03591161 0.03532
##             [,21]      [,22]      [,23]      [,24]      [,25]
## gamma       3          3.1        3.2        3.3        3.4
## pval        0.07951496 0.08517937 0.09089124 0.09730922 0.1032111
```

```
## deviate     1.408342   1.371052   1.335287   1.297037   1.263465
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.3680134  0.377045   0.3856496  0.3939279  0.4018498
## variance    0.03474238 0.03417884 0.03362933 0.03327287 0.03275573
##             [,26]      [,27]      [,28]      [,29]      [,30]
## gamma       3.5        3.6        3.7        3.8        3.9
## pval        0.1091425  0.1150957  0.1210639  0.1270409  0.1330211
## deviate     1.231101   1.199866   1.169685   1.140491   1.112223
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.4094275  0.4166831  0.423637   0.4303078  0.4367126
## variance    0.03225185 0.03176097 0.03128281 0.03081705 0.03036337
##             [,31]      [,32]      [,33]      [,34]      [,35]
## gamma       4          4.1        4.2        4.3        4.4
## pval        0.1389992  0.1449708  0.1509317  0.156878   0.1628066
## deviate     1.084827   1.05825    1.032446   1.007372   0.9829885
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.4428672  0.448786   0.4544826  0.4599693  0.4652576
## variance    0.02992145 0.02949096 0.02907155 0.02866289 0.02826466
##             [,36]      [,37]      [,38]     [,39]      [,40]
## gamma       4.5        4.6        4.7       4.8        4.9
## pval        0.1687142  0.1745982  0.1804562 0.186286   0.1920856
## deviate     0.9592588  0.936149   0.9136278 0.8916659  0.8702364
## statistic   0.6305185  0.6305185  0.6305185 0.6305185  0.6305185
## expectation 0.4703581  0.4752809  0.4800352 0.4846294  0.4890717
## variance    0.02787653 0.02749818 0.0271293 0.02676958 0.02641874
##             [,41]      [,42]      [,43]      [,44]      [,45]
## gamma       5          5.1        5.2        5.3        5.4
## pval        0.1978533  0.2035875  0.2092869  0.2149503  0.2205766
## deviate     0.849314   0.8288752  0.8088979  0.7893617  0.7702472
## statistic   0.6305185  0.6305185  0.6305185  0.6305185  0.6305185
## expectation 0.4933695  0.4975298  0.5015593  0.5054639  0.5092495
## variance    0.02607648 0.02574253 0.02541662 0.02509849 0.02478789
##             [,46]      [,47]     [,48]      [,49]      [,50]
## gamma       5.5        5.6       5.7        5.8        5.9
## pval        0.2261649  0.2317144 0.2372244  0.2426942  0.2481235
## deviate     0.7515365  0.7332126 0.7152594  0.6976622  0.6804066
## statistic   0.6305185  0.6305185 0.6305185  0.6305185  0.6305185
## expectation 0.5129215  0.516485  0.5199447  0.5233052  0.5265707
## variance    0.02448457 0.0241883 0.02389887 0.02361604 0.02333961
##             [,51]
## gamma       6
## pval        0.2535118
## deviate     0.6634794
## statistic   0.6305185
## expectation 0.5297453
## variance    0.02306938
```

```
plot(x = sens_results['gamma',],
     y = sens_results['pval',],
     xlab = "Gamma",
```

```
    ylab = "P-Value",
    main = "Sensitivity Analysis"); abline(h = 0.05)
```



Question for Students:

- Interpret the plot above.

```
find_Sens_G <- function(gamma,
                        alpha){
```

```r
  senmv(meddat_reshaped,
        gamma = gamma)$pval - alpha
}

## Find x value at which the function above == 0

uniroot(f = find_Sens_G,
                lower = 1,
                upper = 6,
                a = 0.05)$root
```

```
## [1] 2.789649
```

Question for Students:

- Explain what the `[R]` uniroot function above is doing.

1.3 General Sensitivity Analysis

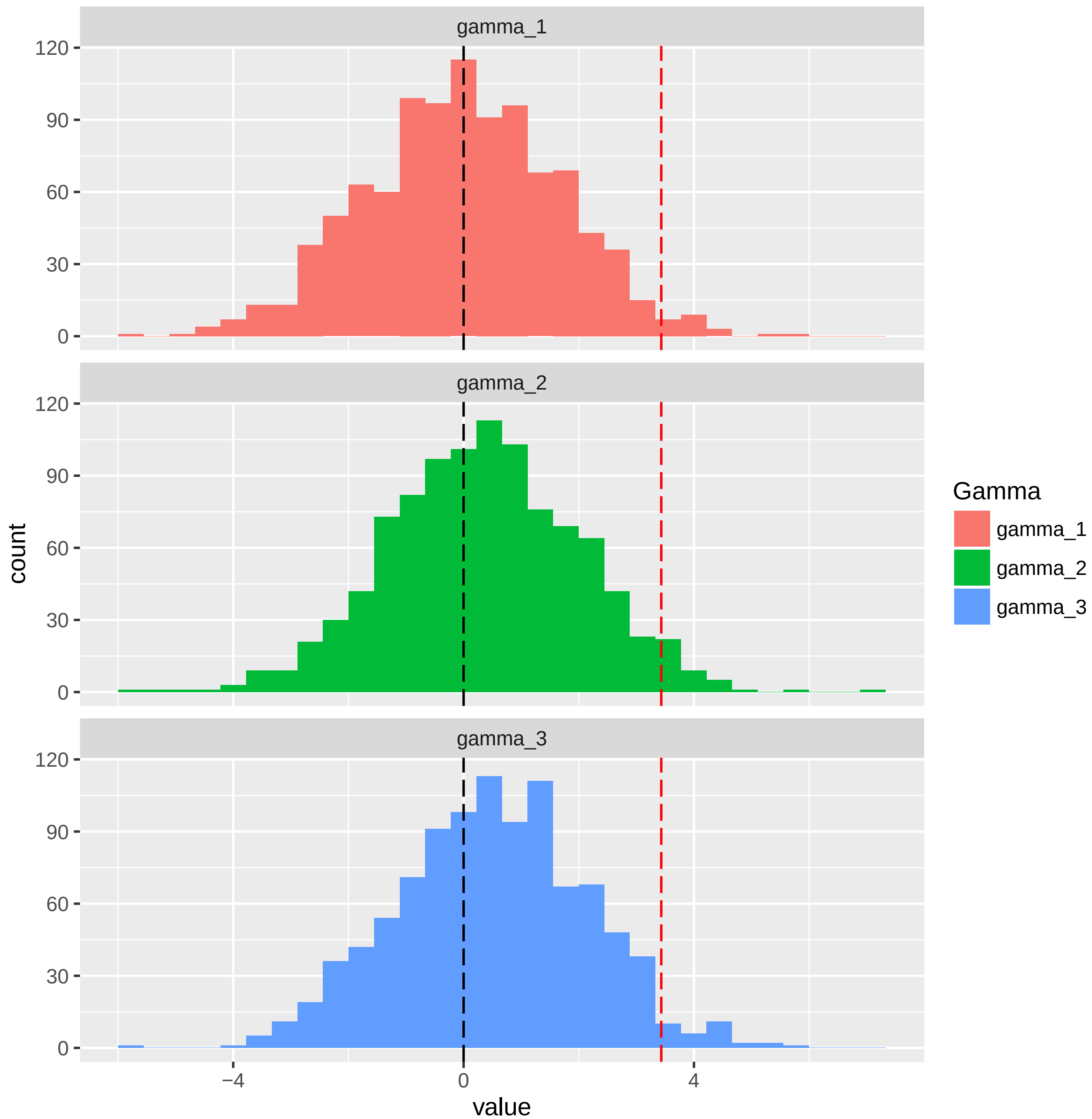
Now to gain some intuition about what exactly a sensitivity analysis is doing, let's consider this very general sensitivity analysis below.

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

gen_sensitivity_analysis <- function(.Y,
                                     .n,
                                     .nt,
                                     .pi,
                                     .gamma){

  if(.gamma == 1)

    prob_treated_u <- prob_treated_l <- .pi # If Gamma = 1, treatment assignment probability stays same

  else {

    odds_u <- .gamma * .pi / (1 - .pi) # Get upper hypothetical odds

    odds_l <- .pi / (.gamma * (1 - .pi) ) # Get lower hypothetical odds

    prob_treated_u <- odds_u / (1 + odds_u) # Convert upper odds back to probabilities

    prob_treated_l <- odds_l / (1 + odds_l) # Convert lower odds back to probabilities

  }
```

```r
  treated_sample <- sort(sample(x = .Y, ## This corresponds to sharp null
                              size = .nt,
                              replace = FALSE,
                              prob = c(rep(x = prob_treated_u,
                                           times = .nt),
                                       rep(x = .pi,
                                           times = .n - .nt))))

  Y <- c(treated_sample,
         base::setdiff(.Y,
                       treated_sample))

  ATE_under_sharp_null <- mean(Y[1 : .nt]) - mean(Y[(.nt + 1) : .n])

  return(ATE_under_sharp_null)

}
```

```r
## Generate fake data
fake_data <- data.frame(unit = seq(from = 1, to = 100, by = 1),
                        z = c(rep(1, 50), rep(0, 50)),
                        yc = rnorm(n = 100, mean = 10, sd = 8),
                        tau = rnorm(n = 100, mean = 5, sd = 6.5)) %>%
  mutate(yt = yc + tau,
         y = z * yt + (1 - z) * yc)

## Calculate observed ATE
obs_ATE <- fake_data %$%{ mean(y[z == 1]) - mean(y[z == 0]) }

## Test sharp null hypothesis of no effect and calculate p-value according to Gamma = 1

new_ra_under_null <- function(z, y){

  Z = sample(z)

  ate = coef(lm(y ~ Z))[["Z"]]

  return(ate)
}

null_dist_gamma_1 <- replicate(1000, new_ra_under_null(z = fake_data$z,
                                                       y = fake_data$y))

## p value
mean(abs(null_dist_gamma_1) >= abs(obs_ATE))


## [1] 0.029


gammas <- (seq(from = 1, to = 3, by = 1))
```

```r
set.seed(1:5)

n_sims <- 10^3

null_dists <- data.frame(t(replicate(n_sims,
                                     sapply(gammas, gen_sensitivity_analysis,
                                            .Y = fake_data$y,
                                            .n = nrow(fake_data),
                                            .nt = sum(fake_data$z),
                                            .pi = sum(fake_data$z)/ nrow(fake_data))))) %>%
  dplyr::mutate(sim = seq(from = 1,
                          to = n_sims,
                          by = 1)) %>%
  dplyr::rename(gamma_1 = X1,
                gamma_2 = X2,
                gamma_3 = X3)


## Changing dataframe from wide to long format using reshape2 package
null_dists_long <- reshape2::melt(data = null_dists,
                        id.vars = "sim") %>%
  mutate(variable = as.factor(variable)) %>%
  rename(Gamma = variable)

ggplot(data = null_dists_long,
       aes(x = value, fill = Gamma)) +
  geom_histogram() +
  facet_wrap(~ Gamma,
             ncol = 1) +
  geom_vline(xintercept = 0,
             colour = "black",
             linetype = "longdash") +
  geom_vline(xintercept = obs_ATE,
             colour = "red",
             linetype = "longdash")
```



Questions for Students:

- Describe how the three null randomization distributions change under different values of Γ.
- Why is the p-value with a $\Gamma = 3$ smaller even though the mean of its null distribution is higher?

```
apply(null_dists[,1:3], 2, mean)
```

```
##     gamma_1     gamma_2     gamma_3
## -0.01369304  0.34242316  0.48590328
```

```r
apply(null_dists[,1:3], 2, var)
```

```
##  gamma_1  gamma_2  gamma_3
## 2.865130 2.770726 2.690725
```

```r
apply(null_dists[,1:3], 2, function(x) mean(abs(x) >= abs(obs_ATE)))
```

```
## gamma_1 gamma_2 gamma_3
##   0.045   0.048   0.032
```

Questions for Students:

- How does the mean of the null randomization distribution change across different values of Γ?
- How does the variance of the null randomization distribution change across different values of Γ?
- How does the p-value change across different values of Γ?

References

Rosenbaum, P. R. (2002). *Observational Studies* (Second ed.). New York, NY: Springer. 2, 3

Rosenbaum, P. R. (2015). Two r packages for sensitivity analysis in observational studies. *Observational Studies 1*, 1–17. 3, 5