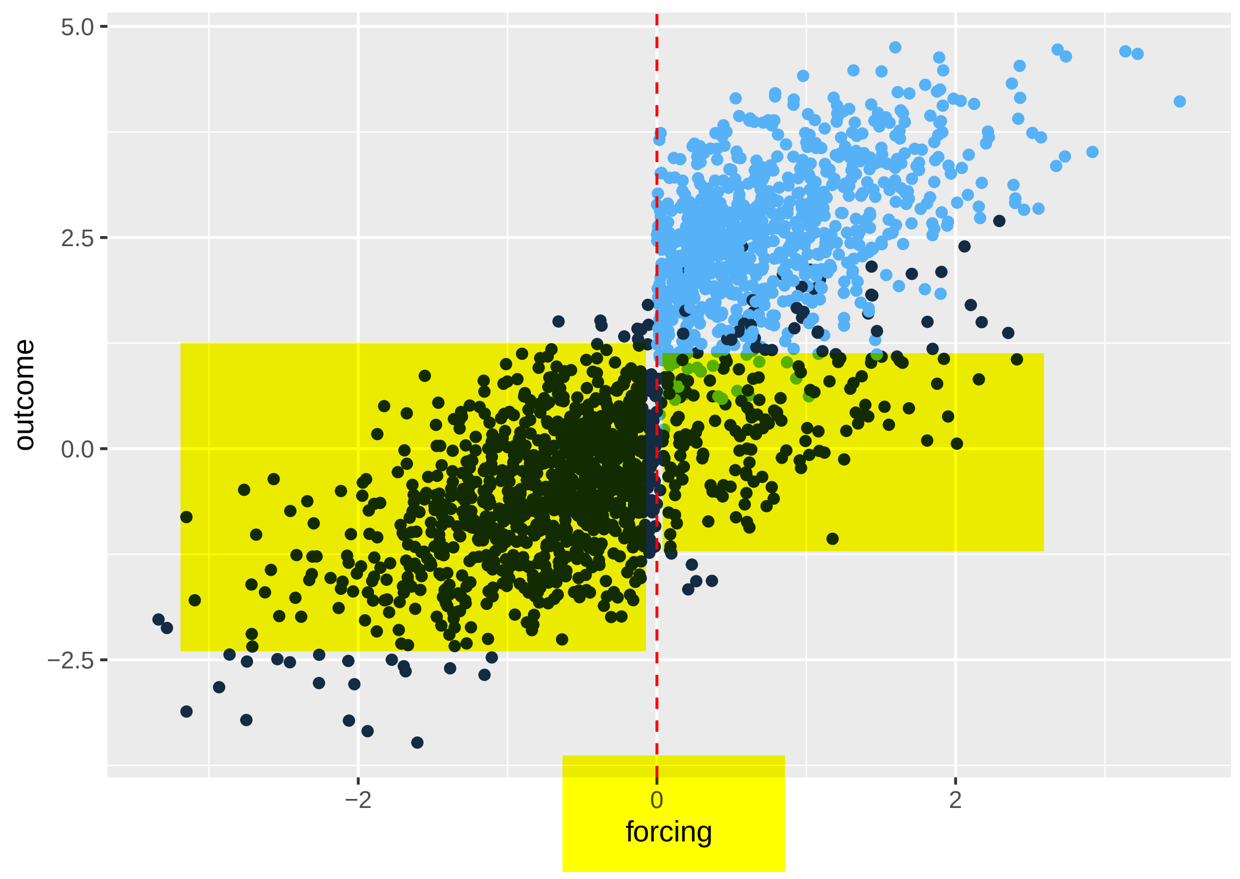
fuzzy_rdd_simulate_Apr1_DM.R

danny

2020-03-31

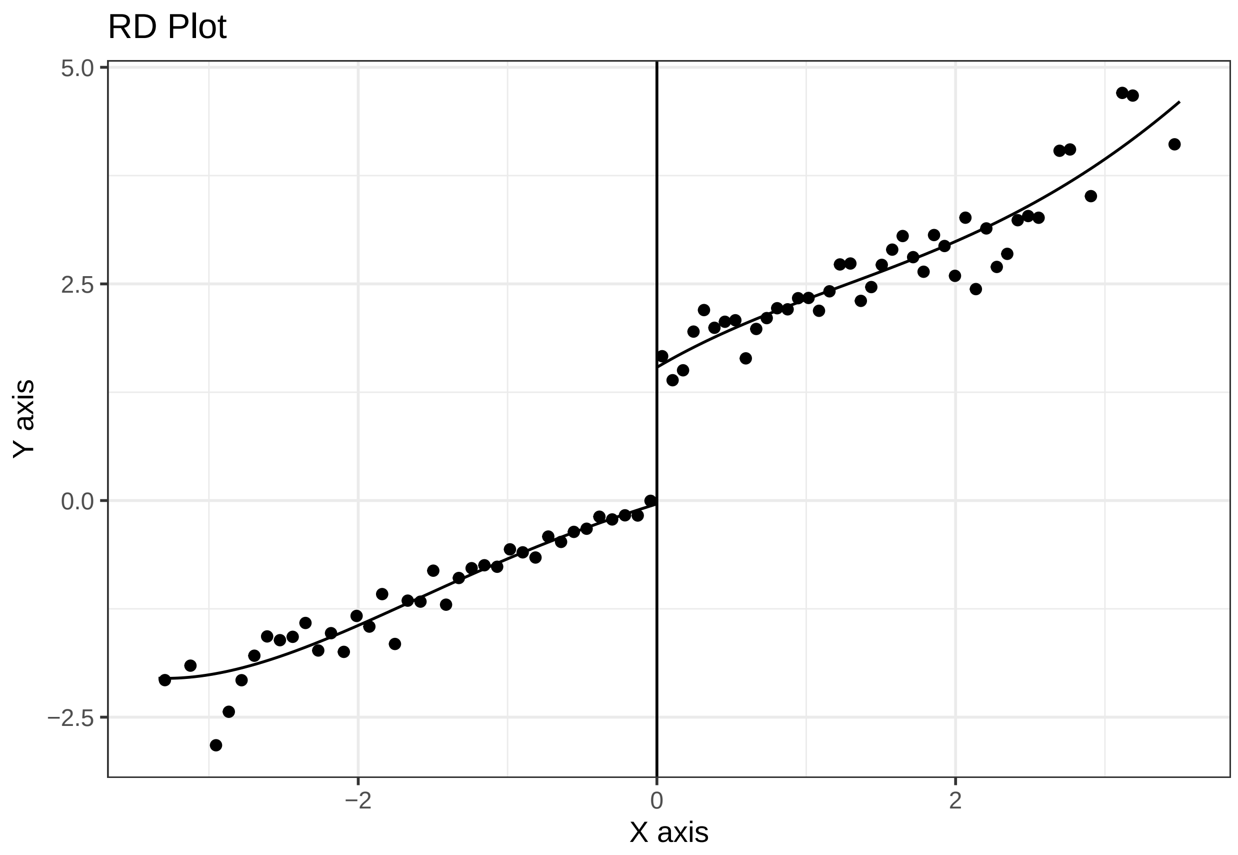
```r
# Author: Gordon Burtch and Gautam Ray
# Course: MSBA 6440
# Session: Regression Discontinuity
# Topic: Fuzzy RDD Example
# Lecture 8

suppressWarnings(suppressPackageStartupMessages({library(MASS)
library(ggplot2)
library(rdrobust)
library(AER)
}))

# We are going to simulate some data that we can use for a fuzzy RDD analysis.
# We will first draw Y and X as continuous values, with a 70% correlation.
# Both values are mean 0.
set.seed(1234)
d <- as.data.frame(mvrnorm(2000, c(0,0), matrix(c(1, 0.7, 0.7, 1), ncol = 2)))
colnames(d) <- c("forcing", "outcome")

# introduce fuzziness - you are not guaranteed to get the treatment policy if you exceed the threshold.
d$treatProb <- ifelse(d$forcing < 0, 0, 0.8)

# This is our "fuzzy" treatment assignment
d$fuzz <- rbinom(2000, 1, prob = d$treatProb)

# being treated adds "2" to your mean value of Y, i.e., the treatment effect is "2".
d$outcome <- d$outcome + d$fuzz * 2

# Let's plot the data now to make sure everything worked.
ggplot(d, aes(y=outcome,x=forcing,col=fuzz)) + geom_point(show.legend = FALSE) + geom_vline(xintercept=
```



```r
attach(d)

# Let's first see how well we do when we ignore the fuzzy aspect... we get an estimate of 1.58 (the tru
robust_naive_rdd <- rdrobust(outcome,forcing,c=0)
summary(robust_naive_rdd)
```

```
## Call: rdrobust
##
## Number of Obs.                 2000
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  999         1001
## Eff. Number of Obs.             691          679
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   0.989        0.989
## BW bias (b)                   1.640        1.640
## rho (h/b)                     0.603        0.603
## Unique Obs.                     999         1001
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     1.583     0.102    15.537     0.000     [1.383 , 1.782]
##         Robust         -         -    13.222     0.000     [1.340 , 1.807]
## =============================================================================
```

```
rdplot(outcome,forcing)
```



```
# Now, let's try it accounting for fuzziness... much improved! Estimate = 1.989
robust_fuzzy_rdd <- rdrobust(outcome,forcing,c=0,fuzzy=fuzz)
summary(robust_fuzzy_rdd)
```

```
## Call: rdrobust
##
## Number of Obs.                   2000
## BW type                         mserd
## Kernel                     Triangular
## VCE method                         NN
##
## Number of Obs.                   999         1001
## Eff. Number of Obs.              691          679
## Order est. (p)                     1            1
## Order bias  (q)                    2            2
## BW est. (h)                    0.989        0.989
## BW bias (b)                    1.640        1.640
## rho (h/b)                      0.603        0.603
## Unique Obs.                      999         1001
```

```
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     1.989     0.101    19.712     0.000     [1.792 , 2.187]
##         Robust         -         -    16.628     0.000     [1.747 , 2.214]
## =============================================================================
```

```
# In case you want to see what the package is actually doing, I will run the linear RDD specification f
# and then the Fuzzy RDD specification (IV regression) second...

# Make treatment and deviation variable Z and (X-c)... note: X-c = X because c = 0.
d$treat <- (forcing>0)
lm_rdd <- lm(data=d,outcome ~ treat + forcing)
lm_frdd <- ivreg(data=d, outcome ~ fuzz + forcing |.-fuzz + treat)
summary(lm_rdd)
```

```
##
## Call:
## lm(formula = outcome ~ treat + forcing, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5004 -0.5054  0.0804  0.6270  2.1742
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.01447    0.03992   0.362    0.717
## treatTRUE    1.58504    0.06776  23.393   <2e-16 ***
## forcing      0.70901    0.03428  20.685   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9233 on 1997 degrees of freedom
## Multiple R-squared:  0.7014, Adjusted R-squared:  0.7011
## F-statistic:  2346 on 2 and 1997 DF,  p-value: < 2.2e-16
```

```
summary(lm_frdd)
```

```
##
## Call:
## ivreg(formula = outcome ~ fuzz + forcing | . - fuzz + treat,
##     data = d)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -2.366845 -0.491032  0.007309  0.490770  1.989962
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.004775   0.031280   0.153    0.879
## fuzz        1.999728   0.066392  30.120   <2e-16 ***
## forcing     0.696798   0.026944  25.861   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7171 on 1997 degrees of freedom
## Multiple R-Squared: 0.8199,  Adjusted R-squared: 0.8197
## Wald test:  3889 on 2 and 1997 DF,  p-value: < 2.2e-16
```