Regression Discontinuity

Alex Alekseev

```
library(tidyverse)
library(rdrobust)
library(rddensity)
```

Intro

In this session, we will practice using the regression discontinuity design (RDD) using a simulated dataset. There are several packages in R that implement RDD, we will be using rdrobust and rddensity.

Data

```
nobs <- 1000
thresh <- 0.5
jump <- 0.25
```

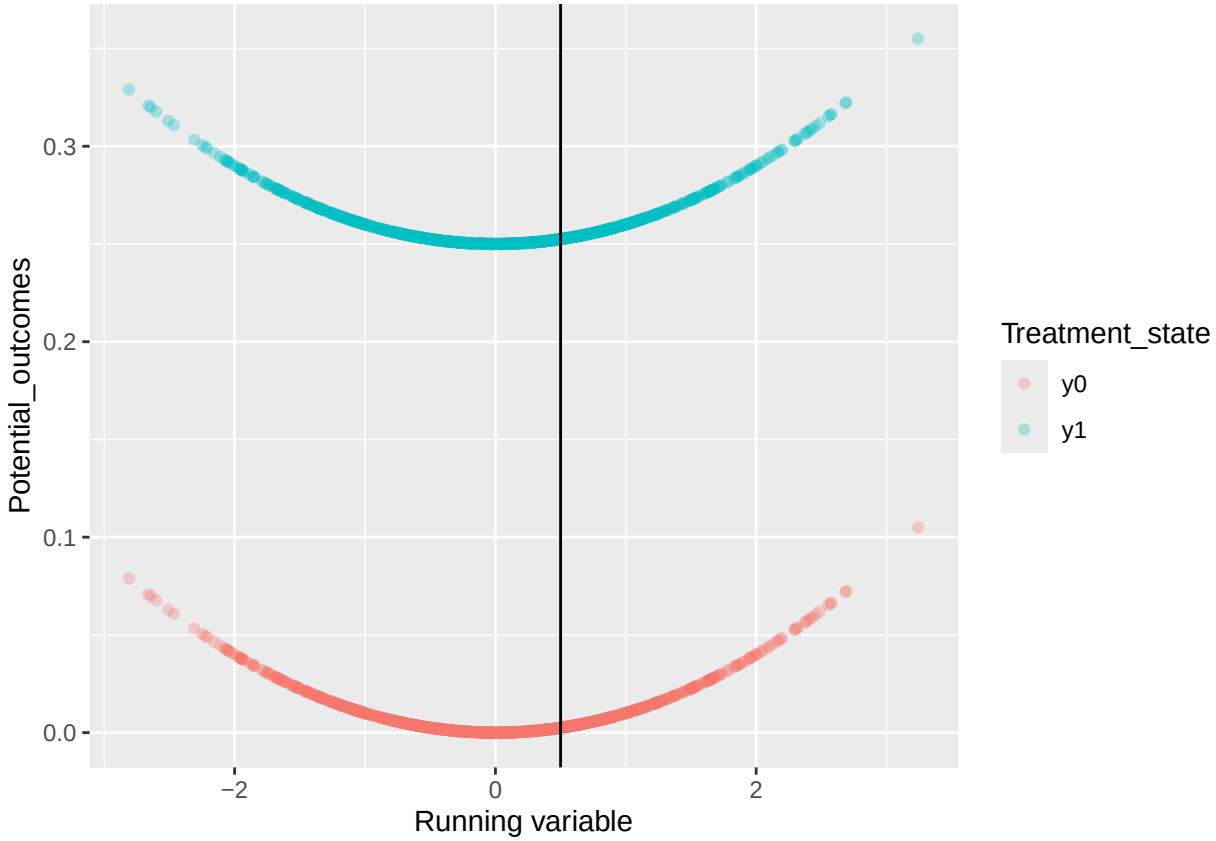
We will generate a dataset with 1000 observations of the running variable (x), potential outcomes (y0, y1), observed outcomes (y), and the treatment status (d). We will also create a centered running variable (x_cent) for which the threshold is 0.

```
set.seed(123)

df <- tibble(
  x = rnorm(n = nobs)
  , y0 = x^2/100
  , y1 = y0 + jump
  , d = 1*(x >= thresh)
  , y = (1 - d)*y0 + d*y1 + rnorm(n = nobs, sd = 0.3)
  , x_cent = x - thresh
)
```

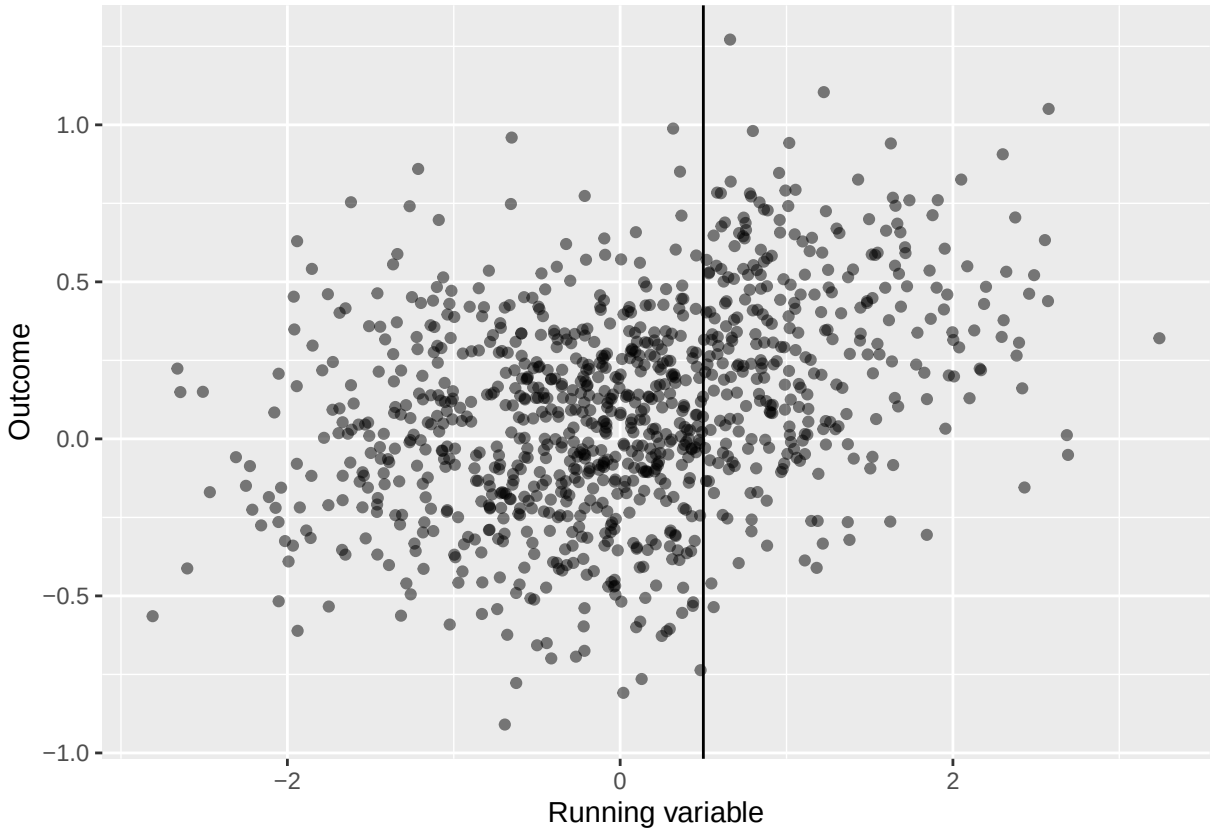
The potential outcome Y^0 is a non-linear (quadratic) function of the running variable. The potential outcome Y^1 is simply shifted by fixed amount (0.25) relative to Y^0. This set-up satisfies the main identifying assumption of RDD: *continuity*. The only reason why you would observe a jump in the observed outcomes is due to treatment. In the absence of treatment, the potential outcomes are continuous at the threshold (0.5).

```r
df %>%
  pivot_longer(
    cols = c(y0, y1)
    , names_to = "Treatment_state"
    , values_to = "Potential_outcomes"
  ) %>%
  ggplot(aes(x, Potential_outcomes, color = Treatment_state)) +
  geom_point(alpha = 0.3) +
  geom_vline(xintercept = 0.5) +
  xlab("Running variable")
```



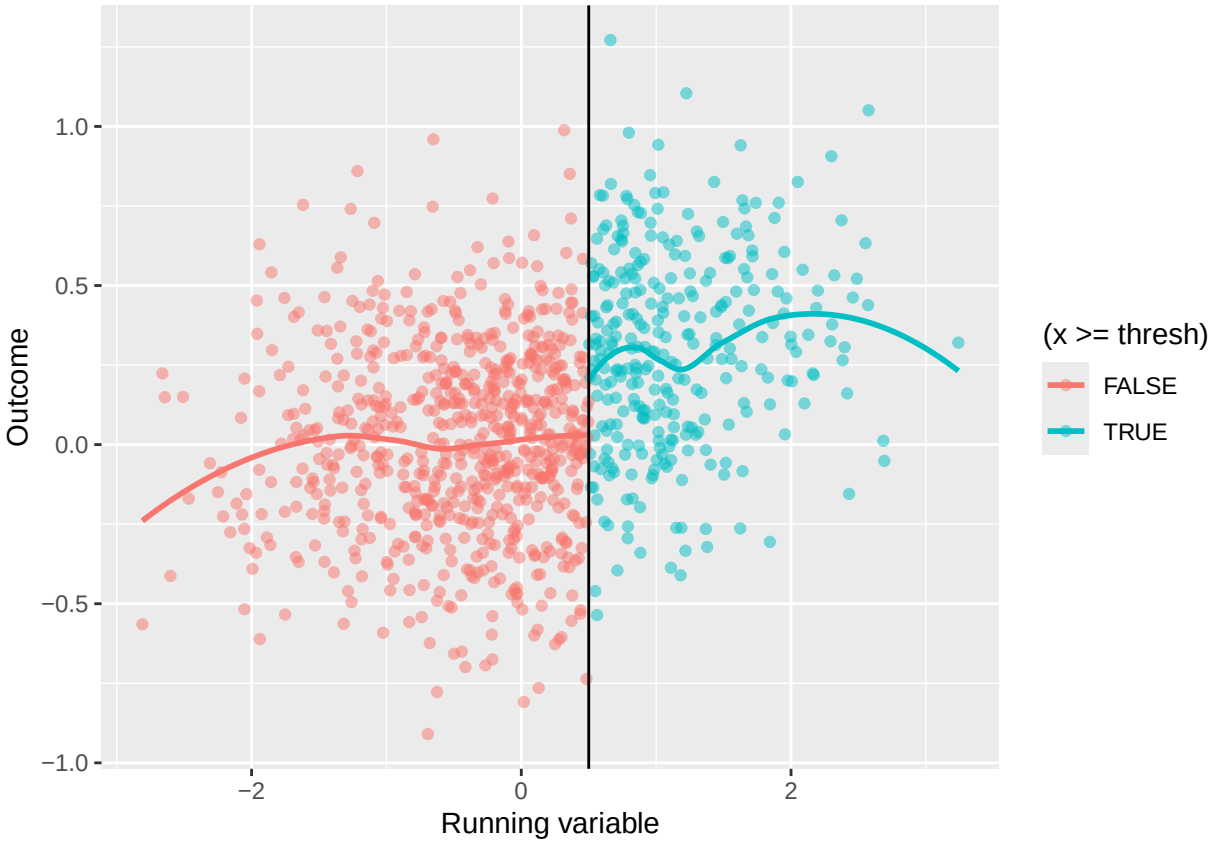
Let's switch from the potential outcomes to the observed outcomes.

```r
ggplot(df, aes(x, y)) +
  geom_point(alpha = 0.5) +
  geom_vline(xintercept = 0.5) +
  labs(x = "Running variable", y = "Outcome")
```



There is a lot of noise in the data, which makes it hard to see a jump at the threshold. We can add trend lines on both sides of the cutoff to aid the analysis.

```
ggplot(df, aes(x, y, color = (x >= thresh))) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "loess", se = F) +
  geom_vline(xintercept = 0.5) +
  labs(x = "Running variable", y = "Outcome")
```



Now the jump at the threshold becomes more pronounced. We can visually confirm that this jump is in fact around the true value of 0.5.

Estimation

To estimate the treatment effect formally, we will use the `rdrobust` function.

```
m <- rdrobust(
  df$y, df$x_cent, c = 0
  , all = T
)
summary(m)
```

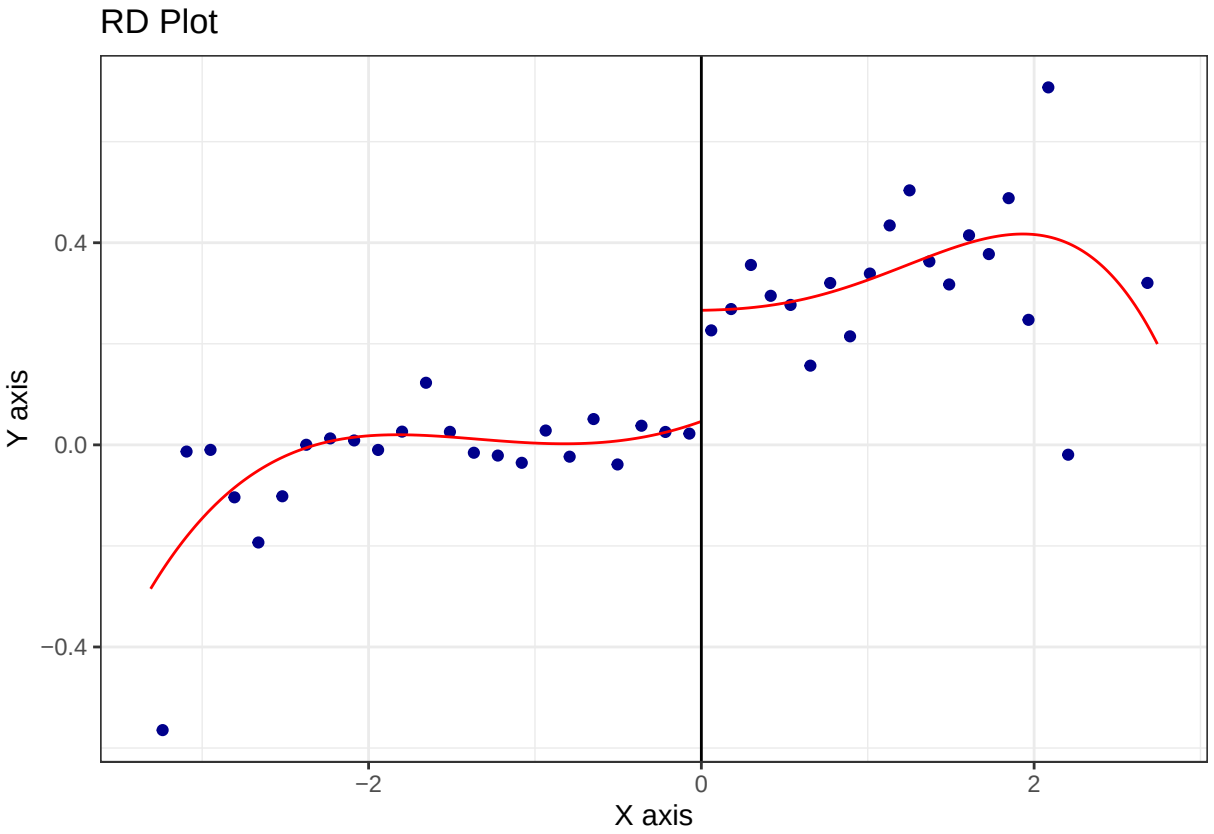
```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                 697          303
## Eff. Number of Obs.            185          133
## Order est. (p)                   1            1
## Order bias  (q)                  2            2
## BW est. (h)                  0.448        0.448
## BW bias (b)                  0.707        0.707
```

```
## rho (h/b)                     0.633         0.633
## Unique Obs.                     697           303
##
## ===============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## ===============================================================================
##   Conventional     0.204     0.075     2.698     0.007     [0.056 , 0.351]
## Bias-Corrected     0.185     0.075     2.453     0.014     [0.037 , 0.333]
##         Robust     0.185     0.090     2.064     0.039     [0.009 , 0.361]
## ===============================================================================
```

The option `all = T` allows us to see different estimates (conventional or bias-corrected) and standard errors (conventional or robust). The estimated effect is between 0.204 and 0.185, depending on the estimate, which is close to the true value of 0.25 albeit lower. The confidence intervals do include the true value.

We can use the `rdplot` function to visualize the results.

```
rdplot(df$y, df$x_cent)
```



Notice that the function made some choices for us. For example, it picked the number of polynomials `p` (= 1) in the local regression and the bandwidth `h` (= 0.448). Let's change these numbers manually and see if it changes the results.

First, let's increase the degree of polynomials by changing the value of `p`.

```
m1 <- rdrobust(
  df$y, df$x_cent, c = 0
  , all = T
```

```
  , p = 2
)
summary(m1)
```

```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  697          303
## Eff. Number of Obs.             255          172
## Order est. (p)                    2            2
## Order bias  (q)                   3            3
## BW est. (h)                   0.609        0.609
## BW bias (b)                   0.862        0.862
## rho (h/b)                     0.707        0.707
## Unique Obs.                     697          303
##
## ===============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## ===============================================================================
##   Conventional     0.200     0.093     2.140     0.032     [0.017 , 0.383]
## Bias-Corrected     0.192     0.093     2.057     0.040     [0.009 , 0.375]
##         Robust     0.192     0.106     1.817     0.069    [-0.015 , 0.399]
## ===============================================================================
```

Increasing `p` widened the confidence intervals.

This pattern continues if we further increase `p`.

```
m2 <- rdrobust(
  df$y, df$x_cent, c = 0
  , all = T
  , p = 4
)
summary(m2)
```

```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  697          303
## Eff. Number of Obs.             360          217
## Order est. (p)                    4            4
## Order bias  (q)                   5            5
```

```
## BW est. (h)                 0.878        0.878
## BW bias (b)                 1.128        1.128
## rho (h/b)                   0.778        0.778
## Unique Obs.                   697          303
##
## ===============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## ===============================================================================
##   Conventional     0.213     0.127     1.682     0.093    [-0.035 , 0.461]
## Bias-Corrected     0.220     0.127     1.739     0.082    [-0.028 , 0.468]
##         Robust     0.220     0.136     1.623     0.105    [-0.046 , 0.486]
## ===============================================================================
```

Increasing `p` beyond the default value seems to produce worse results.

Let's set the bandwidth `h` manually. We will start with a value smaller than the computed default value.

```
m3 <- rdrobust(
  df$y, df$x_cent, c = 0
  , all = T
  , h = 0.5
)
summary(m3)
```

```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                      Manual
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                 697          303
## Eff. Number of Obs.            202          146
## Order est. (p)                   1            1
## Order bias  (q)                  2            2
## BW est. (h)                  0.500        0.500
## BW bias (b)                  0.500        0.500
## rho (h/b)                    1.000        1.000
## Unique Obs.                    697          303
##
## ===============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## ===============================================================================
##   Conventional     0.209     0.073     2.879     0.004     [0.067 , 0.351]
## Bias-Corrected     0.182     0.073     2.507     0.012     [0.040 , 0.324]
##         Robust     0.182     0.103     1.767     0.077    [-0.020 , 0.384]
## ===============================================================================
```

Notice two things. First, there is a drop in the effective number of observations (`Eff.`

`Number of Obs.`) relative to the very first specification. Making the bandwidth smaller means we are using fewer observations that are closer to the threshold. Second, this tends to make the estimate less precise (the standard errors go up) but also, interestingly, pushes the estimated effect up.

Let's try to pick a larger bandwidth.

```
m4 <- rdrobust(
  df$y, df$x_cent, c = 0
  , all = T
  , h = 1
)
summary(m4)
```

```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                      Manual
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  697          303
## Eff. Number of Obs.             402          229
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   1.000        1.000
## BW bias (b)                   1.000        1.000
## rho (h/b)                     1.000        1.000
## Unique Obs.                     697          303
##
## ===============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## ===============================================================================
##   Conventional     0.235     0.055     4.291     0.000     [0.127 , 0.342]
## Bias-Corrected     0.200     0.055     3.664     0.000     [0.093 , 0.308]
##         Robust     0.200     0.077     2.594     0.009     [0.049 , 0.352]
## ===============================================================================
```

Larger bandwidth increases the effective number of observations and makes the estimates more precise.

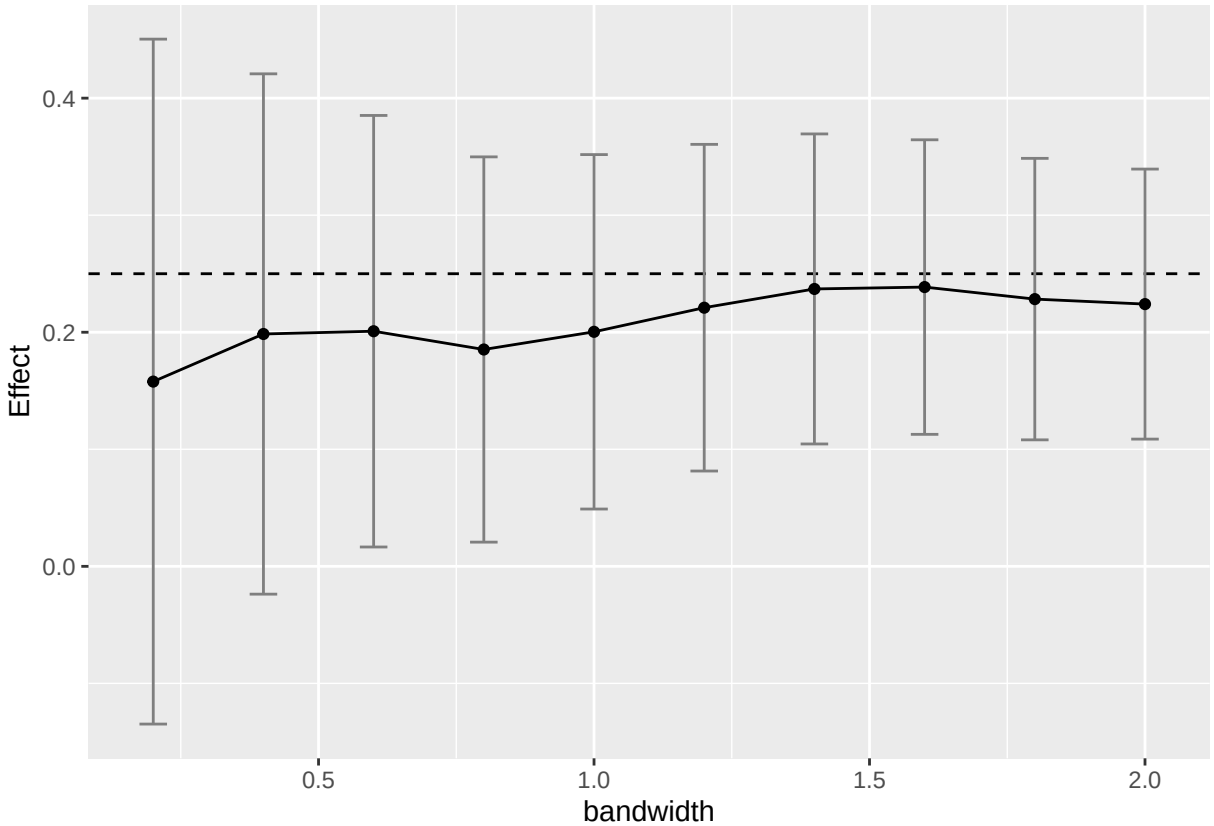
We can conduct a sensitivity analysis by showing how our estimates would change with the bandwidth.

```
est_fun <- function(x) {
  m <- rdrobust(
    df$y, df$x_cent, c = 0
    , all = T
    , h = x
  )
```

```r
  est <- m$Estimate[1, "tau.bc"]
  ci_low <- m$ci[3, 1]
  ci_high <- m$ci[3, 2]
  res <- c("est" = est, "ci_low" = ci_low, "ci_high" = ci_high)
  return(res)
}

df_est <- tibble(
  bandwidth = seq(0.2, 2, 0.2)
  , estimate = map_df(bandwidth, est_fun)
) %>%
  unnest(cols = estimate)

ggplot(df_est, aes(x = bandwidth, y = est.tau.bc)) +
  geom_hline(yintercept = jump, linetype = "dashed") +
  geom_errorbar(aes(ymin = ci_low, ymax = ci_high), color = "gray50", width = 0.05) +
  geom_point() +
  geom_line() +
  labs(y = "Effect")
```



We see that the estimates are pretty stable and the confidence intervals include the true value.

Robustness checks

We can conduct some placebo tests by picking the "wrong" thresholds. As the name suggests, we should not find any effects there.

We find a zero effect at -0.5 (left side of the threshold).

```
m5 <- rdrobust(
  df$y, df$x, c = -0.5
  , all = T
)
summary(m5)
```

```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  295          705
## Eff. Number of Obs.             136          209
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   0.522        0.522
## BW bias (b)                   0.818        0.818
## rho (h/b)                     0.638        0.638
## Unique Obs.                     295          705
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.044     0.066     0.667     0.505     [-0.085 , 0.173]
## Bias-Corrected     0.044     0.066     0.673     0.501     [-0.085 , 0.174]
##         Robust     0.044     0.078     0.569     0.569     [-0.109 , 0.197]
## =============================================================================
```

We find a zero effect at 1 (right side of the threshold).

```
m6 <- rdrobust(
  df$y, df$x, c = 1
  , all = T
)
summary(m6)
```

```
## Sharp RD estimates using local polynomial regression.
##
## Number of Obs.                 1000
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
```

```
##
## Number of Obs.                  843          157
## Eff. Number of Obs.              94           67
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   0.338        0.338
## BW bias (b)                   0.644        0.644
## rho (h/b)                     0.525        0.525
## Unique Obs.                     843          157
##
## ===============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## ===============================================================================
##   Conventional    -0.009     0.109    -0.081     0.935    [-0.223 , 0.205]
## Bias-Corrected     0.033     0.109     0.306     0.760    [-0.181 , 0.247]
##         Robust     0.033     0.124     0.270     0.787    [-0.209 , 0.276]
## ===============================================================================
```

We can also conduct the density test to check for manipulation around the threshold.

```
rdd_dens <- rddensity(df$x_cent)
summary(rdd_dens)
```

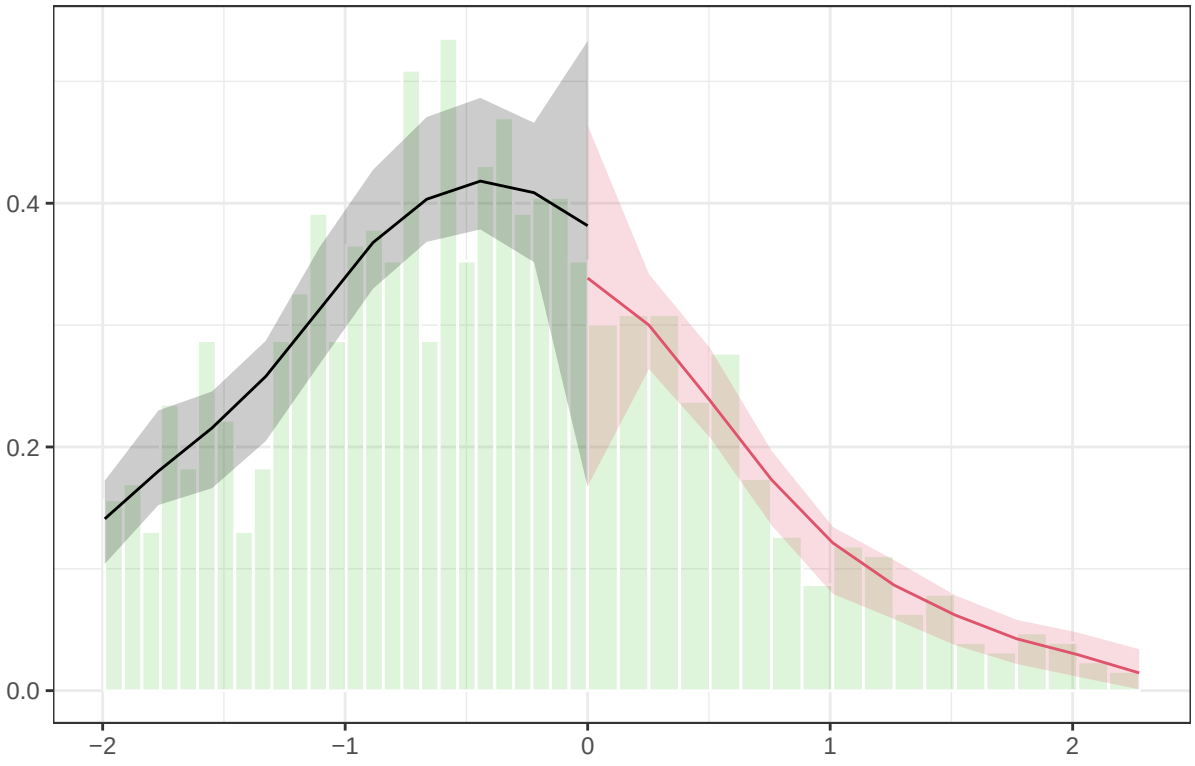
```
##
## Manipulation testing using local polynomial density estimation.
##
## Number of obs =       1000
## Model =               unrestricted
## Kernel =              triangular
## BW method =           estimated
## VCE method =          jackknife
##
## c = 0                 Left of c           Right of c
## Number of obs         697                 303
## Eff. Number of obs    271                 203
## Order est. (p)        2                   2
## Order bias (q)        3                   3
## BW est. (h)           0.664               0.758
##
## Method                T                   P > |T|
## Robust                -0.2746             0.7836
##
##
## P-values of binomial tests (H0: p=0.5).
##
## Window Length / 2          <c     >=c    P>|T|
## 0.062                      20      20    1.0000
## 0.125                      46      38    0.4452
## 0.187                      69      59    0.4264
```

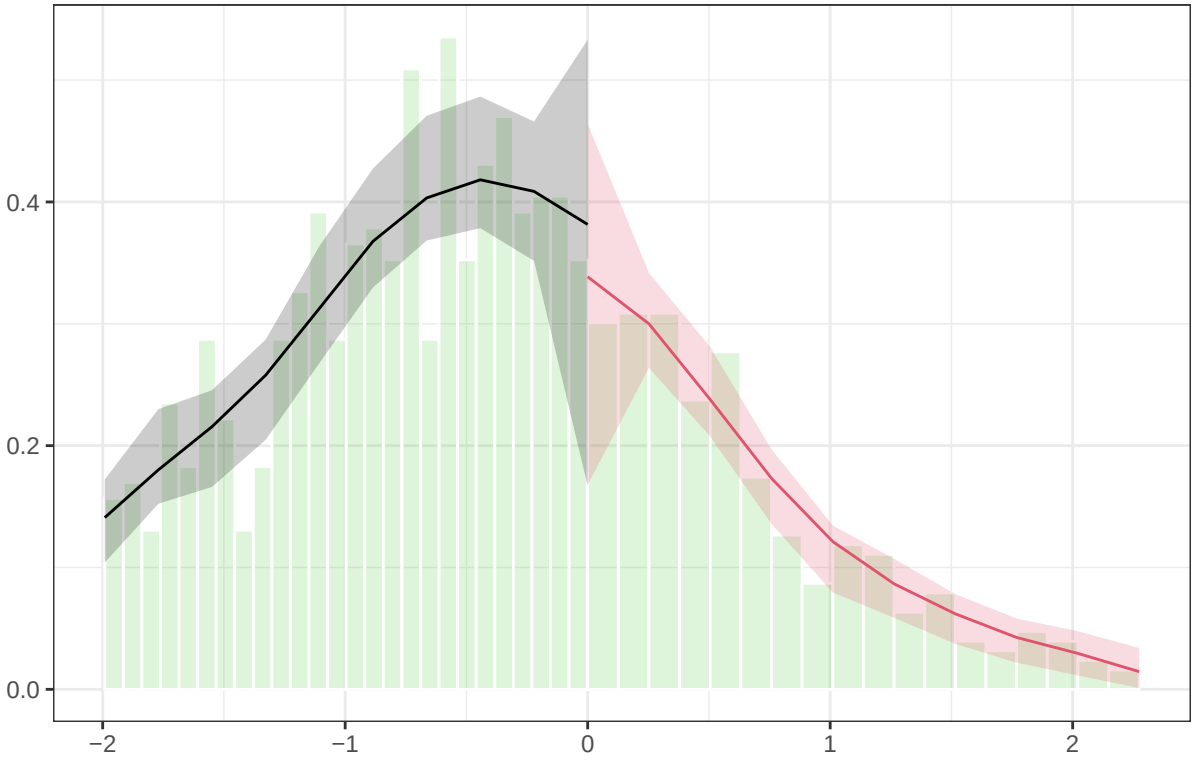
```
## 0.249                    93      76    0.2183
## 0.311                   122      93    0.0559
## 0.374                   147     115    0.0553
## 0.436                   180     132    0.0077
## 0.498                   201     146    0.0037
## 0.560                   224     163    0.0022
## 0.623                   260     177    0.0001
```

The p-value is 0.784, which suggests that we cannot reject the null of no discontinuities in the density of the running variable. We conclude that there is no manipulation around the threshold. We know from our data simulation that this is indeed the case.

We can also present the results visually.

```
rdplotdensity(rdd_dens, df$x_cent)$Estplot
```




0.4
0.2
0.0
−2
−1
0
1
2