Regression Discontinuity Design (fuzzy)

Causal Inference, DPIR, University of Oxford

Tanisha Mohapatra & Fernando Sanchez Monforte

HT 2025

Recap

Sharp RDD

Last week, we discussed sharp regression discontinuity designs, in which a cut-off perfectly determines the treatment status of units. In simple words: all units on one side of the cut-off are treated, while all units on the other side are not:

$$\Pr\left(D_i = 1\right) = p\left(X_i\right)$$

$$p\left(X_i\right) = \begin{cases} 1 & \text{if } X_i \geq X_0 \\ 0 & \text{if } X_i < X_0 \end{cases}$$

Fuzzy RDD

Note that sharp RDDs are only a special case of regression discontinuity designs. **Fuzzy RDDs** allow us to estimate causal effects in settings where treatment assignment is not deterministic. The cut-off may not perfectly determine treatment exposure, but it creates a *discontinuity in the probability* of treatment exposure. The cut-off therefore does not perfectly predict treatment status. Still, we can use a fuzzy RDD as long as it introduces a meaningful as-if random discontinuity that we can take advantage of.

$$\lim_{X_i \uparrow X_0} p(X_i) \neq \lim_{X_i \downarrow X_0} p(X_i)$$

As we learned before, we cannot simply ignore treatment uptake as it likely is not random. Other (unobserved) covariates might come into play and cause units to take up or reject the treatment. In other words, *non-compliance* with treatment assignment poses a potential problem to estimating unbiased causal effects.

More precisely, a *fuzzy RDD* simply incorporates an instrumental variable (IV) that is applied to a regression discontinuity. We can usually use the treatment assignment variable - i.e. the side of the cut-off - as an instrument for treatment status. Similarly to the IV design, we estimate the *LATE* among compliers - or *CACE* at the discontinuity. Note that all assumptions of the IV design need to be satisfied here, too. This allows us to estimate the *LATE* for compliers at the discontinuity (close to the threshold):

$$
\begin{aligned}
\alpha_{FRDD} &= E\left[Y_1 - Y_0 \mid X = c\right] && (1)\\
&= \frac{\lim_{x\downarrow c} E[Y \mid X = c] - \lim_{x\uparrow c} E[Y \mid X = c]}{\lim_{x\downarrow c} E[D \mid X = c] - \lim_{x\uparrow c} E[D \mid X = c]} && (2)\\
&= \frac{\text{outcome discontinuity}}{\text{treatment discontinuity}} && (3)\\
&\approx \frac{E[Y \mid Z = 1] - E[Y \mid Z = 0]}{E[D \mid Z = 1] - E[D \mid Z = 0]} && (4)
\end{aligned}
$$

This should sound familiar - the *LATE* among compliers in fuzzy RDDs can be estimated using a 2SLS regression as we will see in today's lab.

In this **seminar**, we will run RDD step-by-step by:

- Familiarising ourselves with the data structure of fuzzy RDD
- Visualising discontinuities in terms of treatment and outcome at the cut-off.
- Estimating parametric fuzzy RDDs using 2SLS regressions
- Fitting fuzzy RDD model using `rdrobust` (non-parametric estimation)
- Visualizing RDD sensitivity across different bandwidths
- Conduct robustness/falsification using tests for:
  - continuity (imbalance tests, placebo cut-offs)
  - sorting

Before starting this seminar

Create a folder called `lab8`. 2. Download the data (`enter.csv`) from Canvas). 3. Save the data in our `lab8` folder. 4. Open the RMarkdown file (either download it from Canvas or start your own). 5. Set your working directory using the `setwd()` function or by clicking on "More". For example, this may look like this: `setwd(\"\~/Desktop/Causal Inference/2024/lab8\")`. 6. Prepare your R environment running the below.

Exercise. Staying in the First League: Parliamentary Representation and the Electoral Success of Small Parties — Dinas, Riera and Roussias (2014) [1]

To familiarize ourselves with fuzzy RDDs, we will be analysing data from the 2015 paper "Staying in the First League: Parliamentary Representation and the Electoral Success of Small Parties" authored by Elias Dinas, Pedro Riera and Nasos Roussias

In this paper, the authors seek to contribute to answering one of the most essential research questions in party politics: Why are some small parties successful whereas others fail? What accounts for the variation in the trajectories of small parties?

The authors come up with an organisational mechanism that, in their view, contributes to parties' success. According to their expectations, *entering parliament* should lead to an increase in a party's vote share in

[1]Dinas, E., Riera, P. and Roussias, N. (2015). Staying in the First League: Parliamentary Representation and the Electoral Success of Small Parties *Political Science Research and Methods*, 3(2), pp. 187-204

the subsequent election. **Do you think this is a plausible expectation? Why would parliamentary representation matter for future elections?**

This is because parliamentary representation usually comes with a lot of benefits: influence on policy-making, public funding, media visibility, an increase in organisational capacity (e.g. staff) as well as reduced uncertainty about electoral viability and ideological profiles. Conversely, parties that fail to enter parliament do not gain these benefits and might even be abandoned by promising personnel. As you may notice, there might be a lot of reasons which make it difficult to isolate a causal effect without a more elaborate identification strategy.

In many multi-party systems, there is a legal threshold for representation in parliament at the national level, usually in terms of a fixed vote share, that determines whether a party enters parliament or not. The authors exploit randomness introduced by these electoral thresholds to circumvent endogeneity problems. The argument is that parties are not able to disproportionately manipulate their vote share around this threshold - at least in democracies. The authors look at countries that had employed a legal threshold of representation at the national level at least since 1945.

Why do we need to estimate a *fuzzy RDD* here? Unlike in the sharp RDD, some candidates make it into parliament even though their party did not pass the threshold. In other words, there is non-compliance with treatment assignment as the discontinuity does not fully determine treatment status. Roughly 20% of electoral systems allow some parties to get into parliament even when they did not overcome the threshold **- can you think of any potential reasons?**

The authors use a linear regression with a triangular kernel and optimal bandwidths determined by Imbens/Kalyanaramn's algorithm (h=2.4, after the cut-off is centered). Their outcome variable is parties' vote share at the subsequent election (t_1). They find a local average treatment effect among compliers of 1.9 percentage points: Parties that have not cleared the threshold received 3.9 percentage points at election t_1, whereas parties that did overcome the threshold get 5.8 percentage points. This difference translates to a meaningful and substantive effect of an increase of around 40 percent in a party's vote share.

Note: Electoral thresholds vary by country, ranging from 0.67 percentage points to 10 percentage points. To make observations comparable, the authors standardise election results. They use 3.51 as common cut-off point and transform vote shares to that scale. Importantly, the relative distance from the threshold remains the same:

$$X_i \times \frac{\bar{c}}{c} = X_i \times \frac{3.51}{c}$$

Table 1: Variables of interest in the 'Staying in the First League: Parliamentary Representation and the Electoral Success of Small Parties' Dinas, Riera and Roussias (2014)

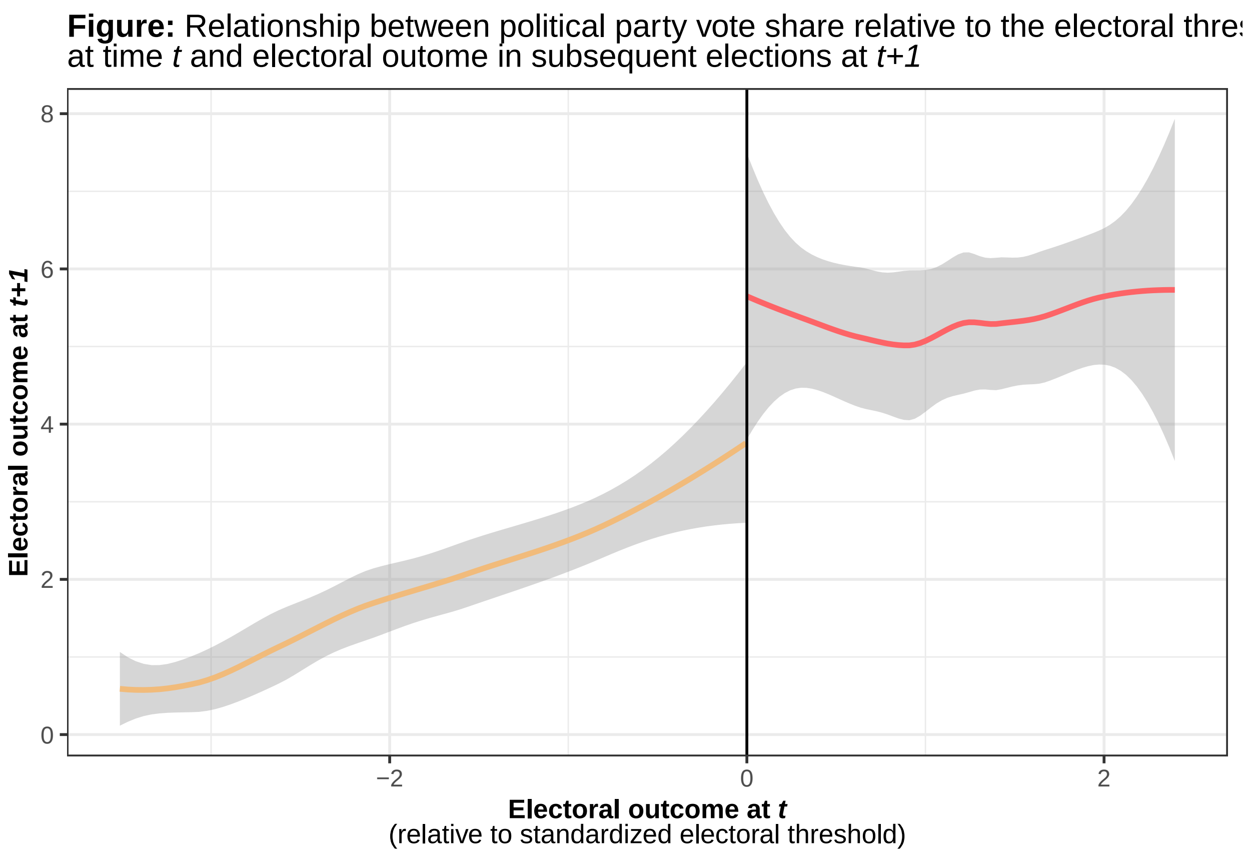
Variable	**Description**
`performance`	**running variable** - *standardised and centred* vote share of party i at t_0.
`treated`	**treatment dummy** - dummy variable indicating whether a party entered parliament (=1) or not (=0)
`treat`	(*mind the difference from the above*) dummy variable indicating whether the party's vote share met/exceeded the threshold (=1) or not
`newsuccess`	**outcome variable** *standardised and centred* party's vote share in the subsequent election at t_1.
`oldsuccess`	*standardised and centred* vote share of party i at $t_{(-1)}$, which we'll use as a placebo
`newperc`	*standardised* vote share in absolute terms
`seats`	number of seats won in an election
`established`	dummy variable indicating if a country is an established democracy
`oldparty`	count variable indicating how many elections a party had contested before

Variable	Description
`country`	country to which the data pertains

Getting to know the data

Task 1. Let's start by visualizing the relationship between the running variable and the outomce of interest

```r
# Set working directory
setwd("/Users/edwardanders/Documents/GitHub/oxford/causal_inference/week_8")

# Load the data
enter <- read.csv("enter.csv")


# Create plot
ggplot(
  enter %>% filter(performance < 2.4),
  aes(
    x = performance, y = newsuccess,
    color = as.character(treat), group = as.character(treat)
  )
) +
  # Add trend lines
  geom_smooth(method = "loess") + # use 'loess' to get local polynomial fit lines - does that change th
  # Make points small and semi-transparent since there are lots of them
  # geom_point(size = 0.5, alpha = 0.5, position = position_jitter(width = 0, height = 0.25, seed = 123
  # Add vertical line
  geom_vline(xintercept = 0) +
  # Add labels
  labs(
    y = "<b>Electoral outcome at <i>t+1</i></b>",
    x = "<b>Electoral outcome at <i>t</i></b><br>(relative to standardized electoral threshold)",
    title = "<b>Figure:</b> Relationship between political party
              vote share relative to the electoral threshold<br>
              at time <i>t</i> and electoral outome in subsequent elections at <i>t+1</i>"
  ) +
  # Change the defaults palette.
  scale_color_manual(values = wesanderson::wes_palette("GrandBudapest1")) +
  # Turn off the color legend, since it's redundant
  guides(color = FALSE) +
  # Define plot theme
  theme_bw() +
  theme(
    plot.title = element_markdown(size = 12),
    axis.title.x = element_markdown(size = 10),
    axis.title.y = element_markdown(size = 10)
  )
```



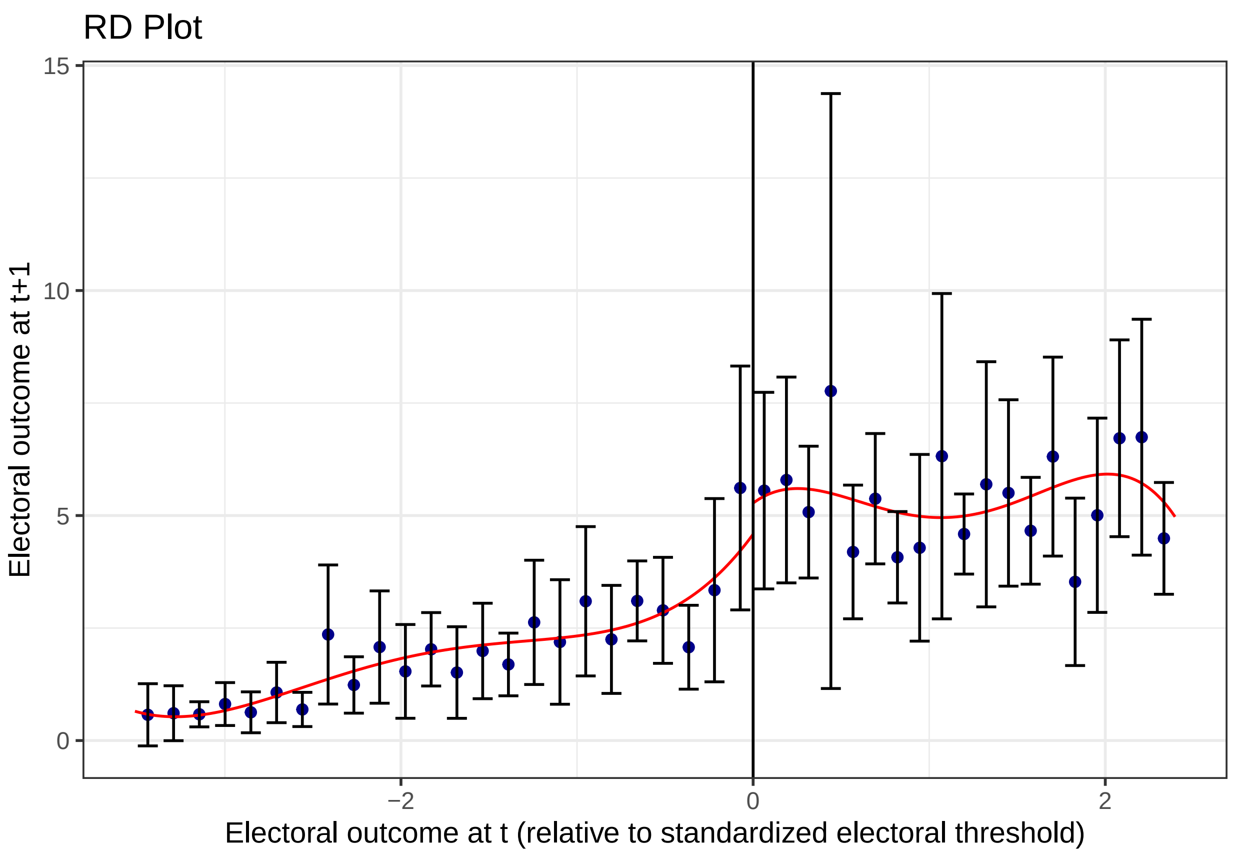
Discontinuity created by the electoral thresholds does appear to be correlated with the electoral success in the subsequent elections. Parties that clear the threshold seem to be receiving around 1.8 percentage points (mind the distinction with %) the next time there is an election.

Task 2. Now let's create a similar plot using an `rdplot` function, which creates data-driven Regression Discontinuity (RD) plots. Its basic syntax is incredibly straightforward, only the outcome and running variables are needed. Once you fit a plot with the outcome variable, please try doing the same with the treatment (`treated`) variable. It should clearly show why we are dealing with a fuzzy design here.

Hint: The function also allows easy subsetting (`subset` argument) of the data and specification of confidence intervals (`ci` argument), which is one of its greatrest strength

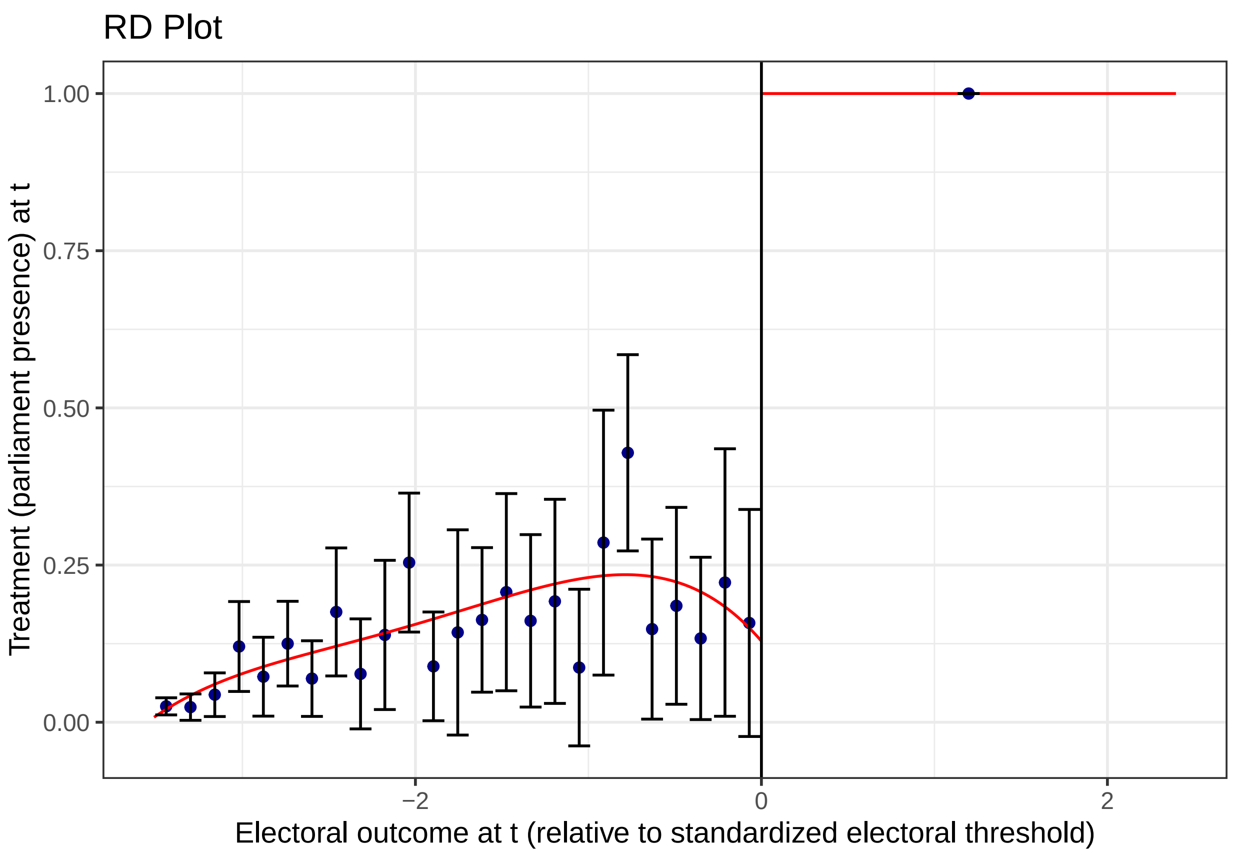
```
# rdplot works out what the optimal spacing bandwidth is,
# and plots the mean of all the observations within that bandwidth
# Create plot with Y = outcome
rdplot(enter$newsuccess, enter$performance,
  ci = 95, subset = enter$performance < 2.4,
  y.label = "Electoral outcome at t+1",
  x.label = "Electoral outcome at t (relative to standardized electoral threshold)"
)
```

```
## [1] "Mass points detected in the running variable."
```



```r
# This plots the equivalent of the first-stage
# Shows that there is non-compliance as being to the left of the cutoff does
# not deterministically tell me that I won't be treated
# Create plot with Y = treatment dummy
rdplot(enter$treated, enter$performance,
  ci = 95, subset = enter$performance < 2.4,
  y.label = "Treatment (parliament presence) at t",
  x.label = "Electoral outcome at t (relative to standardized electoral threshold)"
)
```

```
## [1] "Mass points detected in the running variable."
## [1] "Warning: not enough variability in the outcome variable above the threshold"
```



The plot against the outcome variable looks very similar, although with more precise estimates, the confidence intervals preclude us from telling whether the LATE we can for now gauge visually is statistically significant or not

The plot against the treatment dummy is also instructive. It clearly shows that we are dealing with a fuzzy design. There are some treated units on both sides of the cut-off. However, non-compliance occurs only in the 'control' group, i.e. for observations below the threshold. In this group some observations get the treatment, i.e. a party below the threshold gets into the parliament after all. In the treatment group, there is a perfect compliance.

We can actually calculate the proportion of compliers in each group

```
#  Shows us the extent of non-compliance in the data
# Value 166 is non-compliance
table(enter$treat, enter$treated)
```

```
##
##         0    1
##    0 1610  166
##    1    0 1573
```

```
prop.table(table(enter$treat, enter$treated), 1)
```

```
##
##               0          1
##    0 0.90653153 0.09346847
##    1 0.00000000 1.00000000
```

We can see the count and proportion of compliance in each group. We see that 166 times parties were below the threshold, but ended up in parliament (around 9.3% of the observations). We can confirm that we have a case of one-sided non-compliance.

Let's subset the data for the following analyses. We want to make sure we have a fair comparison between observations below and above the cut-off. Since the `performance` variable cannot have outcomes `<-3,51`, we mirror the data by subsetting positive performance scores:

```r
enter <- enter[enter$performance < 3.51, ]
```

Fuzzy RDD: Parametric Estimation

As last week, let's estimate a parametric fuzzy RDD first to make sure we understand the underlying logic - which is based on the instrumental variables design. Remember that in the case of *fuzzy RDDs* we have 'two events' of interest. First, **'treatment assignment'** which is defined by the running variable exceeding the cut-off, and, secondly, actual **treatment status**. Let formally state that Z_i is an indicator for when $X_{i,\text{centered}}$ exceeds the cut-off. We can then use this variable to instrument for the endogenous treatment variable D_i.

The principle is the same as in the IV design: We have an instrument, Z_i, that describes assignment to treatment based on an observation's position relative to the cut-off. We know that the instrument is a strong predictor - as it should be - of the treatment status, D_i. We can therefore exploit the random variation (close to the cut-off) introduced by the instrument to predict treatment status in our *first stage* - which you can think of as indicating compliance with the treatment assignment or describing the discontinuity in the probability of receiving treatment:

$$\hat{D}_i = \gamma_0 + \gamma_1 X_{i,centered} + \gamma_2 Z_i + \rho_i$$

for

$$c - h \leq X_i \leq c + h$$

Here, in a simple linear model with a single slope $\hat{D}$ predicts the treatment status based on assignment to the treatment, (Z_i), and the vote share ($X_{i,centered}$) in t_0.

The second stage, then, uses the predicted values to calculate the following equation in order to estimate the LATE among compliers at the cut-off:

$$\hat{Y}_i = \beta_0 + \beta_1 X_{i,centered} + \beta_2 \hat{D}_i + \epsilon_i$$

for

$$c - h \leq X_i \leq c + h$$

Our coefficient of interest is β_2, which is the Local Average Treatment Effect ('LATE') among compliers at the cut-off. Having this in mind, we know we can estimate a parametric fuzzy RDD by specifying a 2SLS regression. This can be done using the packages and commands you already know, such as `ivreg()`.

> Task 3. Estimate a parametric fuzzy RDD model using the `ivreg()` command. Interpret your findings.**

```r
# First stage regression
# We find that the estimated effect is 1.65 percentage points
# increase in their t+1 vote share on a standardised scale
model3 <- ivreg(newsuccess ~ treated + performance | treat + performance,
  data = enter
)

summary(model3)
```

```
##
## Call:
## ivreg(formula = newsuccess ~ treated + performance | treat +
##     performance, data = enter)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.6085 -1.3363 -0.3303  0.3963 44.7253
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.9523     0.4281   6.897 9.81e-12 ***
## treated       1.6539     0.6711   2.464   0.0139 *
## performance   0.7504     0.1299   5.774 1.05e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.312 on 930 degrees of freedom
## Multiple R-Squared: 0.3207,  Adjusted R-squared: 0.3192
## Wald test:   227 on 2 and 930 DF,  p-value: < 2.2e-16
```

Specifying the 2SLS model, we find that the estimated effect is 1.65 percentage points on a *standardised scale*. The estimate is also statistically significant, so we conclude that the estimated LATE among compliers is meaningfully different from `0`.

Is this estimate robust though? The global parametric estimation has significant weaknesses. Think of our identification assumption for the RDD. In this global model, we have been using *all* observations from the data frame, including those rather far away from the threshold - i.e. those which are not really comparable. Moreover, the parametric model, unlike kernel estimators, does not weigh observations. Accordingly, the bias introduced by including the whole range of data is possibly severe. We might have estimated a biased effect here.

Let's tackle this problem by estimating a more credible parametric fuzzy RDD. We can do so by subsetting the data we input into the regression model.

> Task 4. Estimate the same model as above, but only consider parties with vote shares up to `1%` on the standardised scale below/above the threshold. Is this result more credible?**

```
model4 <- ivreg(newsuccess ~ treated + performance | treat + performance,
  data = enter %>% filter(abs(performance) < 1)
)

modelsummary(list(model3, model4),
  type = "html",
  dep.var.labels = "",
  dep.var.caption = "Electoral success (t+1)",
  omit = c("Constant"),
```

Table 2: 2SLS model

	(1)	(2)
(Intercept)	2.952	2.274
	(0.428)	(0.843)
treated	1.654	2.934
	(0.671)	(1.265)
performance	0.750	−0.074
	(0.130)	(0.811)
Num.Obs.	933	201
R2	0.321	0.024
R2 Adj.	0.319	0.014
AIC	4887.4	1075.9
BIC	4906.7	1089.1
RMSE	3.31	3.45

```
  omit.stat = c("ser", "f"),
  title = "2SLS model"
)
```

This estimator draws a different picture. Our estimated effect is now `2.93%-points` on the standardised scale - this is meaningful and almost as large in magnitude as the standardised threshold itself. The estimate is also statistically significant. Based on this model specification, we can conclude that the LATE among compliers at the cut-off indeed is meaningful.

This model is somewhat more credible than the one we estimated above. Now we are only looking at observations which represent parties with a vote share of up to `1%-point` above or below the standardised threshold. This is more reasonable as as these parties should be more comparable. In fact, we find that wihtin this range the actual vote share (`performance`) does not predict the result of the subsequent election in any meaningful way. Within this rather small window, entering parliament drives variation in the subsequent election.

Note that the model still has some disadvantages: We specify the functional form of the model and still do not weigh observations based on their distance to the cut-off. Estimating non-parametric models helps us calculate even more robust estimates.

Fuzzy RDD: Non-parametric Estimation

In a first step, let's 'emulate' the parametric model we just specified - this time estimating a non-parametric model.

Fuzzy estimation with `rdrobust` is relatively straightforward. The syntax is essentially the same as in the case of sharp RDDs, obviously with the exception that we need to specify that our model is supposed to estimate a fuzzy RDD. We can do so using the `fuzzy=` argument to specify the treatment status (`treated` in our case). Importantly, you do not need to specify an instrument (or even create one!). All you need to specify is the column that indicates treatment status, the running variable and the cut-off — `rdrobust` will do all the above/below-the-cut-off instrument stuff behind the scenes for you.

Task 5: Estimate a non-parametric fuzzy RDD with a bandwidth of `h = 1` and uniform kernel.

Hint: The syntax of `rdrobust` is pretty similar to the sharp RDD. Recall that the argument `c` specifies the cut-off, while `h` determines the bandwidth. The `fuzzy` argument is sufficient to have `rdrobust` estimate a fuzzy model.

```
summary(rdrobust(
  y = enter$newsuccess,
  x = enter$performance,
  c = 0,
  fuzzy = enter$treated,
  kernel = "uniform",
  h = 1
))
```

```
## Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                  933
## BW type                      Manual
## Kernel                      Uniform
## VCE method                       NN
##
## Number of Obs.                  568          365
## Eff. Number of Obs.              93          108
## 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.                     568          365
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.841     0.096     8.731     0.000     [0.652 , 1.030]
##         Robust         -         -     5.069     0.000     [0.467 , 1.055]
## =============================================================================
##
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     2.586     1.170     2.211     0.027     [0.294 , 4.879]
##         Robust         -         -     0.278     0.781    [-2.963 , 3.944]
## =============================================================================
```

How does that look like? The estimated effect is pretty similar to the one from our (narrow) parametric estimation. Conventional measures of uncertainty also indicate that the effect is significant - looking at the robust p-values and confidence intervals, however, reveals that we would *not* reject the null hypothesis based on the robust estimation - which we should base our conclusion on. Here, we find that there is no significant LATE among compliers.

As we know, `rdrobust`allows us to specify more appropriate models to estimate our treatment effect. Let's now make use of these opportunities to run more precise models.

Task 6: Estimate the treatment effect, specifying a triangular kernel and a MSE-optimal bandwidth. Also, cluster standard errors by country using the `cluster=` argument.**

Hint: You can simply add the `cluster` argument and enter the relevant cluster variable. Recall that optimal bandwidths are specified using the `bwselect` argument instead of `h`

```
summary(rdrobust(
  y = enter$newsuccess,
  x = enter$performance,
  c = 0,
  fuzzy = enter$treated,
  kernel = "triangular",
  cluster = enter$countrycode,
  bwselect = "mserd"
))
```

```
## Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                  933
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  568          365
## Eff. Number of Obs.             100          123
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   1.134        1.134
## BW bias (b)                   1.763        1.763
## rho (h/b)                     0.643        0.643
## Unique Obs.                     232          208
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.810     0.100     8.136     0.000     [0.615 , 1.005]
##         Robust         -         -     7.131     0.000     [0.594 , 1.045]
## =============================================================================
##
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     1.862     1.051     1.772     0.076    [-0.198 , 3.922]
##         Robust         -         -     1.332     0.183    [-0.760 , 3.983]
## =============================================================================
```

We see that the MSE-optimal bandwidth corresponds to `1.13%-points` on the standardised scale on either side of the cut-off.

The estimated LATE among compliers is `1.9`, not that different from our previous point estimates. Under this model specification, we find that the estimated effect is not statistically significant - the robust confidence interval includes zero, so we cannot conclude that there indeed is a significant effect of entering parliament on parties' vote share in the subsequent election.

Note that this model uses a triangular kernel and accounts for non-independence among observations in the same country - for which reason it is somewhat more convincing compared to the models we estimated before.

We can also include covariates. The `rdrobust` package is very flexible in this regard and makes it easy to adjust for them. You can simply add the argument `covs=` and specify the variables you want the model to include.

> Task 7. Specify a non-parametric fuzzy RDD with MSE-optimal bandwidth, clustered standard errors and `seats` and `established` as covariates. Interpret your result.**

```r
summary(rdrobust(
  y = enter$newsuccess,
  x = enter$performance,
  c = 0,
  fuzzy = enter$treated,
  kernel = "triangular",
  cluster = enter$countrycode,
  covs = enter$seats + enter$established,
  bwselect = "mserd"
))
```

```
## Covariate-adjusted Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                  933
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  568          365
## Eff. Number of Obs.              99          120
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   1.111        1.111
## BW bias (b)                   1.725        1.725
## rho (h/b)                     0.644        0.644
## Unique Obs.                     232          208
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.770     0.098     7.838     0.000     [0.578 , 0.963]
##         Robust         -         -     6.885     0.000     [0.557 , 0.999]
## =============================================================================
##
```

```
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     2.045     1.081     1.891     0.059    [-0.075 , 4.164]
##         Robust         -         -     1.428     0.153    [-0.659 , 4.196]
## =============================================================================
```

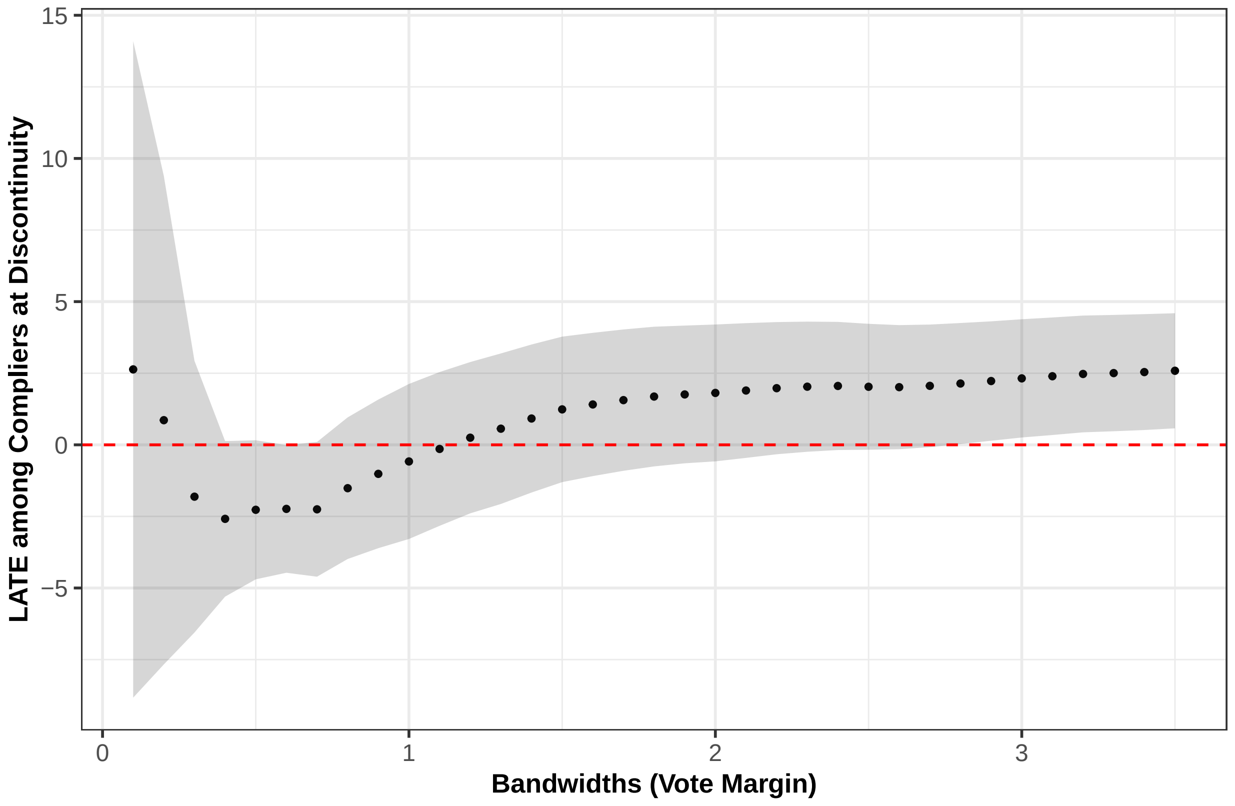
Task 8. Can we be sure our estimated effect is robust to changing the bandwidth? Let's see if the estimated effect varies meaningfully. Let's plot the effect over changing bandwidths. Let's check bandwidths ranging from `0.1` to `3.5` in `0.1%-points` intervals

Hint: Note: You migth want to use the following approach: -Create a data frame with all variables you need for the plot and a and observation for each bandwidth. -Extract the values from the rdrobust output which you need for your plot. -Loop the regression and the extraction of output over the bandwidths indicated in the initial data frame. -Save the output in your loop to the respective row in the data frame. -Plot the output from the newly created data frame.

```r
# create a data.frame with all variables you want to include in your data set
data_extract <- data.frame(
  bandwidth = seq(from = 0.1, to = 3.5, by = 0.1),
  coefficient = NA,
  se = NA, obs = NA,
  bw = NA,
  ci_u = NA, ci_l = NA
)

# create a loop for each bandwidth that is indicated by 'i'
for (i in data_extract$bandwidth) {
  rdbw <- rdrobust(
    y = enter$newsuccess,
    x = enter$performance,
    c = 0,
    fuzzy = enter$treated,
    kernel = "triangular",
    cluster = enter$countrycode,
    covs = enter$seats + enter$established,
    h = i
  ) # run the model
  # extract the model output (make sure to extract *robust* statistics)
  data_extract$coefficient[data_extract$bandwidth == i] <- rdbw$coef[3]
  data_extract$se[data_extract$bandwidth == i] <- rdbw$se[3]
  data_extract$obs[data_extract$bandwidth == i] <- (rdbw$N_h[1] + rdbw$N_h[2])
  data_extract$bw[data_extract$bandwidth == i] <- (rdbw$bws[1, 1])
  data_extract$ci_l[data_extract$bandwidth == i] <- rdbw$ci[3, 1]
  data_extract$ci_u[data_extract$bandwidth == i] <- rdbw$ci[3, 2]
}

# Make sure the coefficient (and all other values) are numeric
data_extract$coefficient <- as.numeric(data_extract$coefficient)

# Plot the estimates across bandwidths
```

```r
ggplot(
  data = data_extract,
  aes(x = bandwidth, y = coefficient)
) +
  geom_point(size = 0.8) +
  geom_ribbon(aes(ymin = ci_l, ymax = ci_u), alpha = 0.2) +
  geom_hline(aes(yintercept = 0), col = "red", linetype = 2) +
  # Add labels
  labs(
    y = "<b>LATE among Compliers at Discontinuity</b>",
    x = "<b>Bandwidths<b> (Vote Margin)",
    title = "<b>Figure:</b> LATE estimates of parliamentary representation on subsequent elections' out
                                 by mode bandwidth"
  ) +
  # Change the defaults palette.
  scale_color_manual(values = wesanderson::wes_palette("GrandBudapest1")) +
  # Turn off the color legend, since it's redundant
  guides(color = FALSE) +
  # Define plot theme
  theme_bw() +
  theme(
    plot.title = element_markdown(size = 12),
    axis.title.x = element_markdown(size = 10),
    axis.title.y = element_markdown(size = 10)
  )
```

Figure: LATE estimates of parliamentary representation on subsequent elections'



This looks interesting - the estimate varies vastly for very narrow bandwidths, but this is expected given the small number of observations that are close around the threshold. This is not uncommon to see. However, it even looks like there is almost a significant *negative* treatment effect for bandwidths around `0.5%-points`. Our estimated *positive* treatment effect is robust to all bandwidths starting from around `2.68%` the end of the range. So, can we be confident that the effect is not contingent on specific bandwidths. What do we conclude?

What would be the effect in substantive terms though?

We know that the threshold (and parties' vote shares) have been standardised, with the standardised threshold being at `3.51%-points`. We can now write a function to convert the standardised (or synthetic) scale into substantive vote shares based on a country's electoral threshold:

```r
standardised_to_vote <- function(coef, threshold) {
  vote_share <- coef * threshold / 3.51
  return(vote_share)
}
```

> Task 9. Based on the treatment effect from our previous model from Exercise 7 (`2.045`), calculate the substantive treatment effect in Turkey and Greece (consult the table of country-specific thresholds in the paper or on the screen in the class). For this exercise, let's forget that the estimated effect was not statistically significant - we just want to get an impression of its magnitude.**

```r
# In Turkey, the threshold is 10%
standardised_to_vote(2.045, 10)
```

```
## [1] 5.826211
```

```r
# In Greece, there's only a 3% threshold
standardised_to_vote(2.045, 3)
```

```
## [1] 1.747863
```

Recall that standardisation means that we are estimating the effect relative to the threshold. Accordingly, we will get different substantive results for countries with different absolute thresholds. The substantive effect will be roughly $\frac{6}{10}$ of the size of the threshold (because `2.045/3.51` ≈ 0.6).

We find that the substantive gain of entering parliament in Turkey corresponds to an increase of `5.83%-points` in the subsequent election, whereas it amounts to `1.75%-points` in Greece.

Robustness/falsification checks

We can conduct a series of checks to estimate how robust our model is. Those would be closely alligned with the checks we performed last week. What checks would you run if that was your paper?

Placebo outcomes

We can conduct multiple tests to find evidence that there is no manipulation of the score around the cutoff. If units cannot precisely manipulate their scores, therefore we should expect no differences in terms of covariates for units above and below the threshold (we will test this later). We should also expect that there should not be differences in outcomes that should not be affected by the treatment. These outcomes are sometimes called 'pseudo-outcome' in the RDD literature.

One pseudo-outcome that we can use in this study is electoral success (standardised vote share) in the previous election (t_{-1}). The idea here is that current parliamentary representation in period (t_0) would not affect electoral performance in t_{-1}. Let's see if this is true.

> Task 10. Conduct a placebo outcome test using the placebo outcome variable `oldsuccess`. Use the `rdrobust()` function to perform this test. Set the argument `bwselect` equal to "mserd". Set the cut-off argument `c` equal to zero. Use the `summary()` command to report the results. What can you conclude based on this test? Does the evidence support the continuity assumption?**

```r
summary(rdrobust(
  y = enter$oldsuccess,
  x = enter$performance,
  c = 0,
  fuzzy = enter$treated,
  kernel = "triangular",
  cluster = enter$countrycode,
  covs = enter$seats + enter$established,
  bwselect = "mserd"
))
```

```
## Covariate-adjusted Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                 1021
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  672          349
## Eff. Number of Obs.             100          113
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   1.094        1.094
## BW bias (b)                   1.462        1.462
## rho (h/b)                     0.748        0.748
## Unique Obs.                     240          196
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.753     0.110     6.814     0.000     [0.536 , 0.969]
##         Robust         -         -     5.719     0.000     [0.515 , 1.052]
## =============================================================================
##
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional    -2.062     2.357    -0.875     0.382    [-6.682 , 2.559]
##         Robust          -          -    -0.659     0.510    [-6.666 , 3.312]
## =============================================================================
```

We find that the local regression estimate is `-1.713` and the robust confidence interval goes from `-5.158` to `2.153` (and the p-value is `0.420`). We find that there no is evidence of manipulation of the running variable around the cut-off with respect ot previous electoral results.

Balance

Let's finally conduct a balance test to examine whether near the cut-off treated units are similar to control units in terms of observable characteristics. Ideally, we should get an RD estimate, τ_{RD}, which is equal to zero.

> Task 11. Conduct balance test(s). What variables would be the most important to check in this regard? Is there any variable that you can think about that should be included, but is not covered in the descriptive table at the start of the exercise?

```r
summary(rdrobust(
  y = enter$oldparty,
  x = enter$performance,
  c = 0,
  fuzzy = enter$treated,
  kernel = "triangular",
  cluster = enter$countrycode,
  covs = enter$seats + enter$established,
  bwselect = "mserd"
))
```

```
## Covariate-adjusted Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                 2293
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                 1776          517
## Eff. Number of Obs.             217          189
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   1.175        1.175
## BW bias (b)                   1.606        1.606
## rho (h/b)                     0.732        0.732
## Unique Obs.                     454          264
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.813     0.058    13.969     0.000     [0.699 , 0.928]
##         Robust         -         -    12.058     0.000     [0.709 , 0.984]
## =============================================================================
##
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional    -0.432     0.615    -0.702     0.483    [-1.636 , 0.773]
##         Robust         -         -    -0.395     0.693    [-1.582 , 1.052]
## =============================================================================
```

```r
summary(rdrobust(
  y = enter$established,
  x = enter$performance,
  c = 0,
  fuzzy = enter$treated,
  kernel = "triangular",
  cluster = enter$countrycode,
  covs = enter$seats + enter$established,
  bwselect = "mserd"
))
```

```
## Covariate-adjusted Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                 2293
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                 1776          517
## Eff. Number of Obs.             170          149
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   0.912        0.912
## BW bias (b)                   1.376        1.376
## rho (h/b)                     0.663        0.663
## Unique Obs.                     454          264
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.811     0.063    12.778     0.000     [0.686 , 0.935]
##         Robust         -         -    11.199     0.000     [0.683 , 0.973]
## =============================================================================
##
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional    -0.272     0.178    -1.529     0.126    [-0.620 , 0.077]
##         Robust         -         -    -1.870     0.062    [-0.724 , 0.017]
## =============================================================================
```

Placebo cut-offs

As we did last week, we can conduct a placebo cut-off test. This test implies looking at the outcome variable but using different cut-offs where we should not expect changes in the outcome. Thus, the estimates from this test should be near zero (and not statistically significant). Again, one **important** step to conduct this test is subsetting the sample for those observations that are above the cut-off and those that are below. We do this to avoid 'contamination' due to the real treatment effect. It also ensures that the analysis of each placebo cut-off is conducted using only observations with the same treatment status.

> Task 12: Conduct a placebo cut-off test. Feel free to select any cut-off(s) that appear sensible. Remember that you need to subset the data so that the discontinuity at the true cut-off is not included**
>
> *Hint*: NRather than subsetting the data and creating a new data frame, let's use the `subset` argument in the `rdrobust()` function.

```r
# Cut-off  at -1
summary(rdrobust(enter$newsuccess,
  enter$performance,
  c = -1,
  fuzzy = enter$treated,
  cluster = enter$countrycode,
  covs = enter$seats + enter$established,
  bwselect = "mserd",
  subset = enter$performance < 0
))
```

```
## Covariate-adjusted Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                  568
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                  475           93
## Eff. Number of Obs.              23           25
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   0.279        0.279
## BW bias (b)                   0.523        0.523
## rho (h/b)                     0.533        0.533
## Unique Obs.                     174           58
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.303     0.193     1.569     0.117    [-0.076 , 0.681]
##         Robust         -         -     1.360     0.174    [-0.135 , 0.744]
## =============================================================================
##
## Treatment effect estimates.
```

```
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     4.880     6.840     0.713     0.476    [-8.526 , 18.287]
##         Robust         -         -     0.844     0.399    [-8.838 , 22.203]
## =============================================================================
```

```r
# Cut-off  at +1
summary(rdrobust(enter$newsuccess,
  enter$performance,
  c = 1,
  fuzzy = enter$treated,
  cluster = enter$countrycode,
  covs = enter$seats + enter$established,
  bwselect = "mserd",
  subset = enter$performance > 0
))
```

```
## Covariate-adjusted Fuzzy RD estimates using local polynomial regression.
##
## Number of Obs.                  356
## BW type                       mserd
## Kernel                   Triangular
## VCE method                       NN
##
## Number of Obs.                   99          257
## Eff. Number of Obs.              38           43
## Order est. (p)                    1            1
## Order bias  (q)                   2            2
## BW est. (h)                   0.332        0.332
## BW bias (b)                   0.497        0.497
## rho (h/b)                     0.668        0.668
## Unique Obs.                      61          146
##
## First-stage estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional     0.000     0.000     7.319     0.000     [0.000 , 0.000]
##         Robust         -         -  -297.735     0.000    [-0.000 , -0.000]
## =============================================================================
##
## Treatment effect estimates.
##
## =============================================================================
##         Method     Coef. Std. Err.         z     P>|z|      [ 95% C.I. ]
## =============================================================================
##   Conventional12747644480604140.00012395634599793332.000     1.028     0.304[-11547352900509352.000
##         Robust         -         -    40.978     0.000[58157210481640064 0.000 , 63999787275597312.0
## =============================================================================
```

We find that the RD estimate for both -1 and +1 placebo cut-offs are not significant. Thus, this coefficient is not statistically significant. This provides evidence in favour of continuity assumption.

Sorting

As we covered last week, the key identification assumption of RDDs is that there is no *sorting* on the running variable. That is, the running variable must be continuous around the threshold. We can do this using *density checks*. This means we're looking at the density of the running variable. Let's again use the `rddensity` package to do so.

> Task 13: Examine the density of the running variable using the `rddensity` command. Interpret your findings. See an example of the syntax below:**

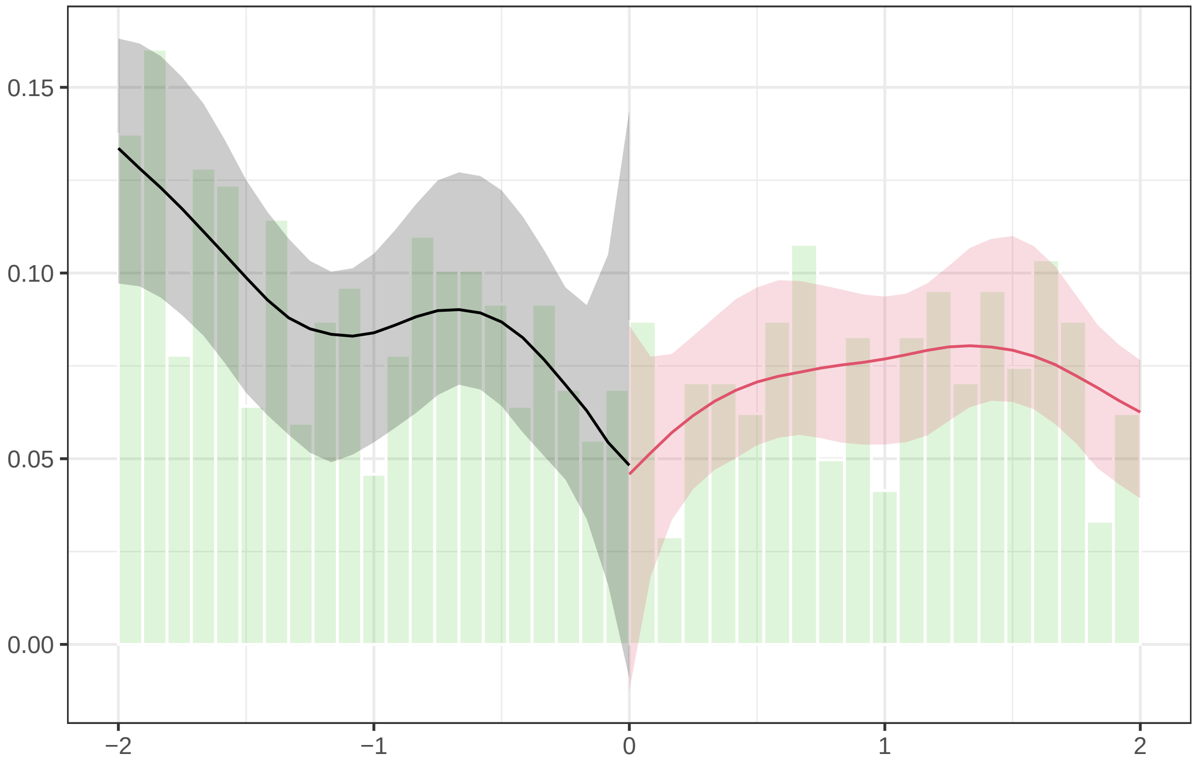
```r
rdd <- rddensity(enter$performance, c = 0)
summary(rdd)
```

```
##
## Manipulation testing using local polynomial density estimation.
##
## Number of obs =       2293
## Model =               unrestricted
## Kernel =              triangular
## BW method =           estimated
## VCE method =          jackknife
##
## c = 0                 Left of c            Right of c
## Number of obs         1776                 517
## Eff. Number of obs    105                  125
## Order est. (p)        2                    2
## Order bias (q)        3                    3
## BW est. (h)           0.611                0.763
##
## Method                T                    P > |T|
## Robust                -0.9864              0.3239


##
## P-values of binomial tests (H0: p=0.5).
##
## Window Length              <c     >=c    P>|T|
## 0.140     + 0.140          20      27    0.3817
## 0.193     + 0.210          27      28    1.0000
## 0.245     + 0.279          34      39    0.6400
## 0.297     + 0.348          42      46    0.7493
## 0.350     + 0.417          47      62    0.1797
## 0.402     + 0.486          62      72    0.4370
## 0.454     + 0.555          73      88    0.2698
## 0.506     + 0.624          83      97    0.3326
## 0.559     + 0.694          94     103    0.5688
## 0.611     + 0.763         105     125    0.2102
```

This test reveals a p-value of `0.3239`. The null hypothesis of this test is that observations near the cut-off are allocated with a 0.5 probability. Given that we failed to reject the null hypothesis, we can claim there is no evidence of sorting around the cut-off.

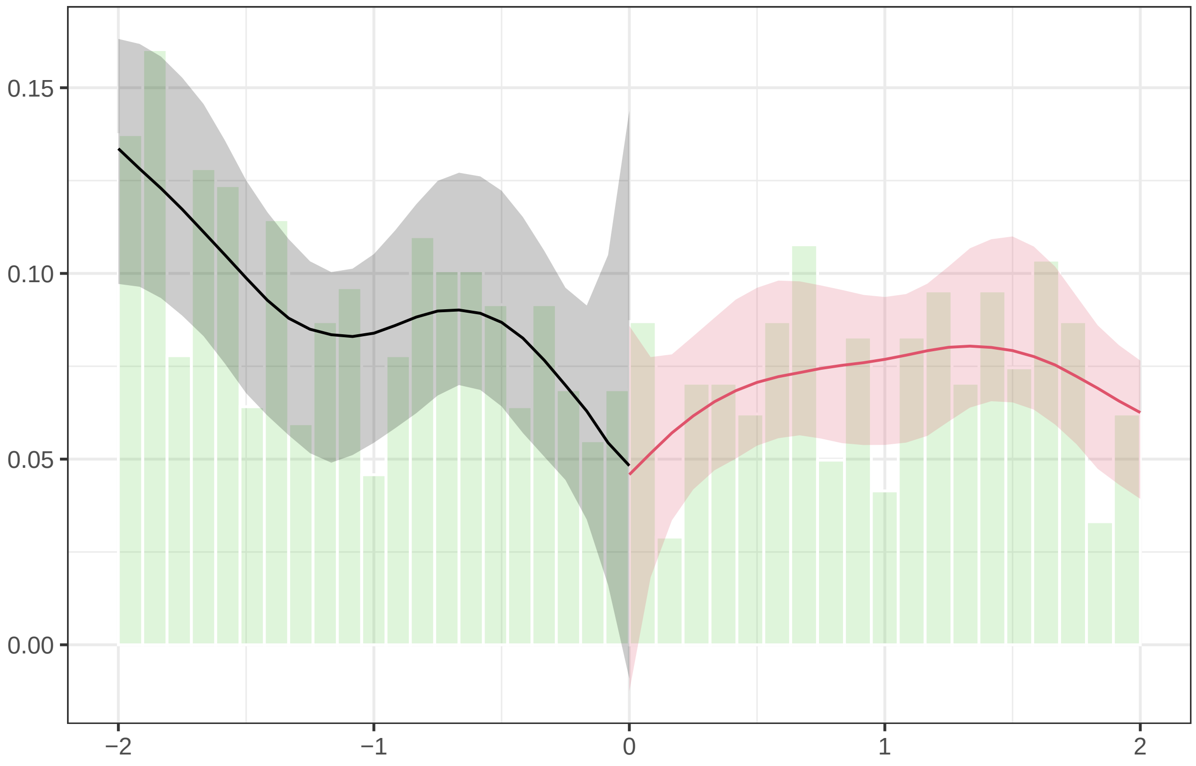
Let's now plot this density test like we did last week. Let's again use the `rdplotdensity` function. Remember that this function uses the output from the `rddensity` command. The syntax is simple:

```
rdplotdensity(rdd, enter$performance, plotRange = c(-2, 2), plotN = 25, CIuniform = TRUE)
```



```
## $Estl
## Call: lpdensity
##
## Sample size                                           1776
## Polynomial order for point estimation    (p=)         2
## Order of derivative estimated            (v=)         1
## Polynomial order for confidence interval (q=)         3
## Kernel function                                       triangular
## Scaling factor                                        0.774869109947644
## Bandwidth method                                      user provided
##
## Use summary(...) to show estimates.
##
## $Estr
## Call: lpdensity
##
## Sample size                                           517
## Polynomial order for point estimation    (p=)         2
## Order of derivative estimated            (v=)         1
## Polynomial order for confidence interval (q=)         3
## Kernel function                                       triangular
## Scaling factor                                        0.225567190226876
## Bandwidth method                                      user provided
##
```

```
## Use summary(...) to show estimates.
##
## $Estplot
```



Again, we find visual evidence for continuity around the cut-off. Now we can be confident about sorting not being a problem in this data.

You can also conduct the same test using the `DCdensity` function from the `rdd` package. Unfortunately, to use this command, you need to drop the observations with missing values. You can use `drop_na(running_variable)` function to do this.