Lab_Two

Sief Salameh

4/17/2023

Loading Libraries

```
library(devtools)
```

```
## Loading required package: usethis
```

```
library(diftrans)

library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.2

##
## Attaching package: 'dplyr'

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

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

```
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.1.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.1.2
```

```
library(tmap)
```

```
## Warning: package 'tmap' was built under R version 4.1.2
```

```
library(ggpubr)
```

```
## Warning: package 'ggpubr' was built under R version 4.1.2
```

Cleaning the Data

```
Beijing <- Beijing_sample %>%
  filter(year >= 2010 & year < 2012)

# collect unique MSRP values
```

```r
uniqueMSRP <- data.frame(MSRP = unique(Beijing$MSRP))

# aggregate sales at each price for 2010 (pre-lottery)
Beijing10_sales <- Beijing %>%
  filter(year == 2010) %>%
  group_by(MSRP) %>%
  summarize(count = sum(sales))

# merge the MSRP and sales
Beijing_pre <- left_join(uniqueMSRP,
  Beijing10_sales,
  by = "MSRP"
) %>%
  replace_na(list(count = 0)) %>%
  arrange(MSRP)

head(Beijing_pre)
```

```
##    MSRP count
## 1 20800     0
## 2 29800    47
## 3 32900  3153
## 4 33800  3678
## 5 34800   592
## 6 36800  1735
```

Exercise 4.1

Part A:

```r
Beijing11_sales <- Beijing %>%
  filter(year == 2011) %>%
  group_by(MSRP) %>%
  summarize(count = sum(sales))

# merge the MSRP and sales
Beijing_post <- left_join(uniqueMSRP,
  Beijing11_sales,
  by = "MSRP"
) %>%
  replace_na(list(count = 0)) %>%
  arrange(MSRP)

head(Beijing_post)
```

```
##    MSRP count
## 1 20800    23
## 2 29800     0
## 3 32900  1393
## 4 33800     4
## 5 34800   189
## 6 36800   459
```

Part B:

```
Tianjin <- Tianjin_sample %>%
  filter(year >= 2010 & year < 2012)

# collect unique MSRP values
uniqueMSRP_2 <- data.frame(MSRP = unique(Tianjin$MSRP))

# aggregate sales at each price for 2010 (pre-lottery)
Tianjin10_sales <- Tianjin %>%
  filter(year == 2010) %>%
  group_by(MSRP) %>%
  summarize(count = sum(sales))

# merge the MSRP and sales
Tianjin_pre <- left_join(uniqueMSRP_2,
  Tianjin10_sales,
  by = "MSRP"
) %>%
  replace_na(list(count = 0)) %>%
  arrange(MSRP)

head(Tianjin_pre)
```

```
##    MSRP count
## 1 20800     0
## 2 28800     0
## 3 29800    51
## 4 30900     0
## 5 32900   599
## 6 33300     2
```

Part C:

```
# aggregate sales at each price for 2010 (pre-lottery)
Tianjin11_sales <- Tianjin %>%
  filter(year == 2011) %>%
  group_by(MSRP) %>%
  summarize(count = sum(sales))

# merge the MSRP and sales
Tianjin_post <- left_join(uniqueMSRP_2,
  Tianjin11_sales,
  by = "MSRP"
) %>%
  replace_na(list(count = 0)) %>%
  arrange(MSRP)

head(Tianjin_post)
```

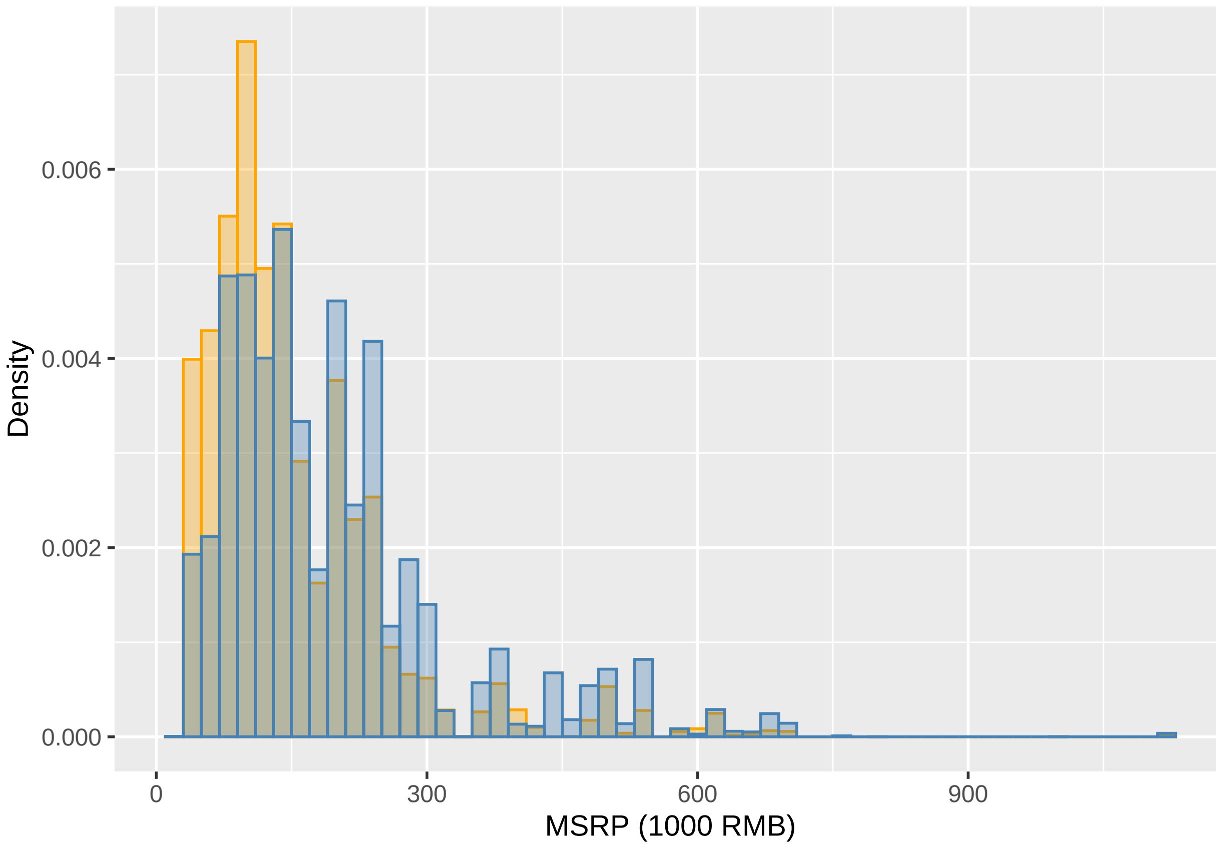
```
##    MSRP count
## 1 20800    23
```

```
## 2 28800      7
## 3 29800      5
## 4 30900      1
## 5 32900    948
## 6 33300      0
```

Visualizing Beijing Car Sale

```r
Beijing_distribution_pre <- Beijing_pre %>% uncount(count)

Beijing_distribution_post <- Beijing_post %>% uncount(count)

bdist <- ggplot() +
  geom_histogram(
    data = Beijing_distribution_pre,
    aes(
      x = MSRP / 1000, # Let price be in terms of 1000 RMB
      y = ..density..
    ), # Normalize bars so their area sum to 1
    binwidth = 20, # Each bin has width of 2000 RMB
    fill = "orange", color = "orange", alpha = 0.35
  ) +
  geom_histogram(
    data = Beijing_distribution_post,
    aes(
      x = MSRP / 1000, # Let price be in terms of 1000 RMB
      y = ..density..
    ), # Normalize bars so their area sum to 1
    binwidth = 20, # Each bin has width of 2000 RMB
    fill = "steelblue", color = "steelblue", alpha = 0.35
  ) +
  xlab("MSRP (1000 RMB)") +
  ylab("Density")

plot(bdist)
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

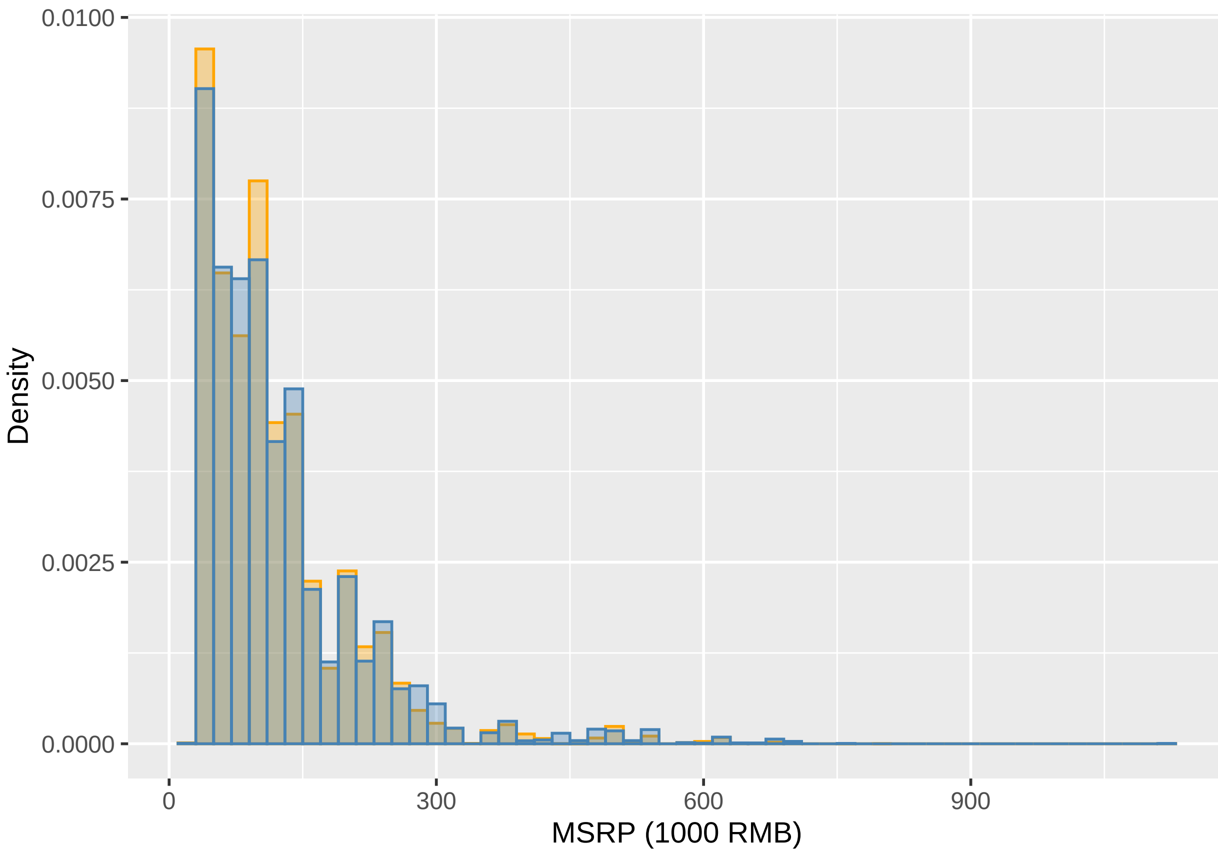


Exercise 4.2

Part A:

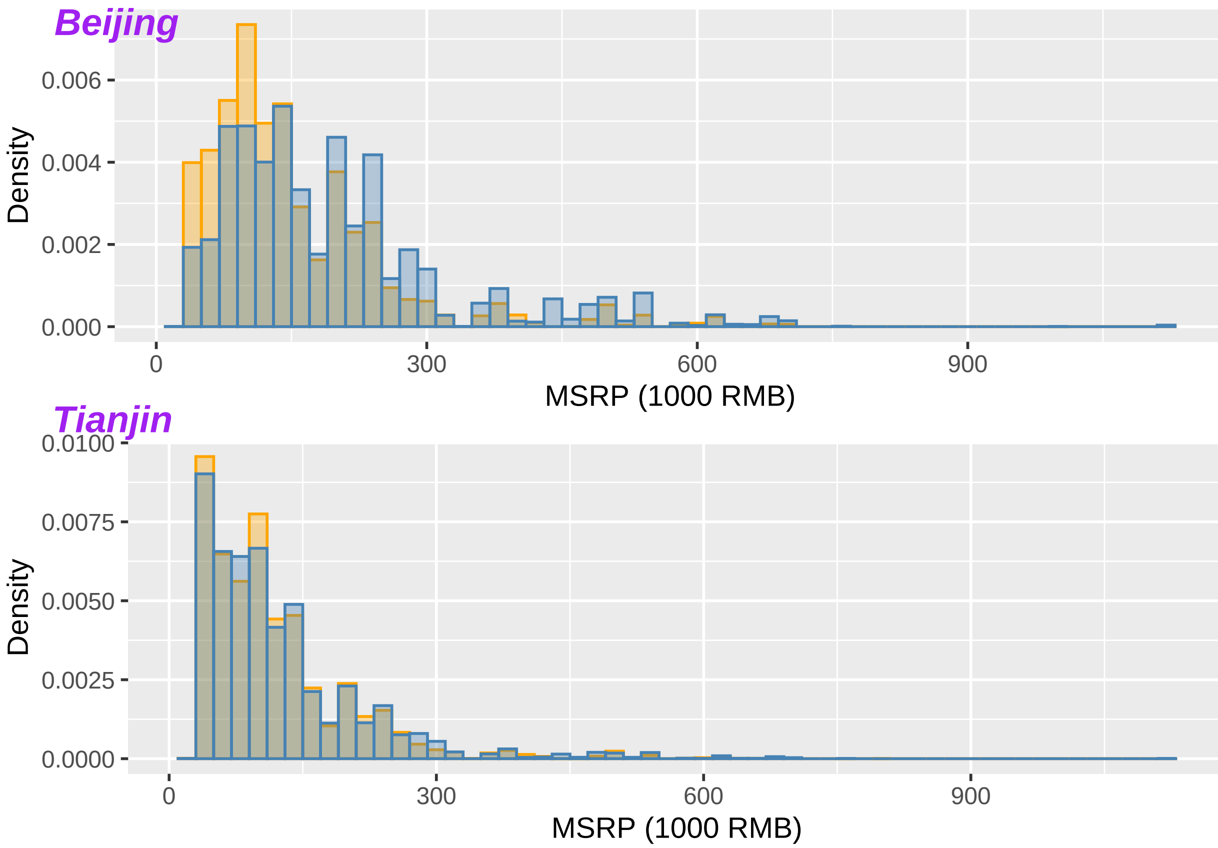
```
Tianjin_distribution_pre <- Tianjin_pre %>% uncount(count)

Tianjin_distribution_post <- Tianjin_post %>% uncount(count)

tdist <- ggplot() +
  geom_histogram(
    data = Tianjin_distribution_pre,
    aes(
      x = MSRP / 1000, # Let price be in terms of 1000 RMB
      y = ..density..
    ), # Normalize bars so their area sum to 1
    binwidth = 20, # Each bin has width of 2000 RMB
    fill = "orange", color = "orange", alpha = 0.35
  ) +
  geom_histogram(
    data = Tianjin_distribution_post,
    aes(
      x = MSRP / 1000, # Let price be in terms of 1000 RMB
      y = ..density..
    ), # Normalize bars so their area sum to 1
```

```
    binwidth = 20, # Each bin has width of 2000 RMB
    fill = "steelblue", color = "steelblue", alpha = 0.35
  ) +
  xlab("MSRP (1000 RMB)") +
  ylab("Density")

plot(tdist)
```



Part B:

```
ggarrange(bdist, tdist,
  labels = c("Beijing", "Tianjin"),
  ncol = 1, nrow = 2,
  vjust = c(1.5, .2),
  font.label = list(size = 14, face = "bold.italic", color = "purple")
)
```



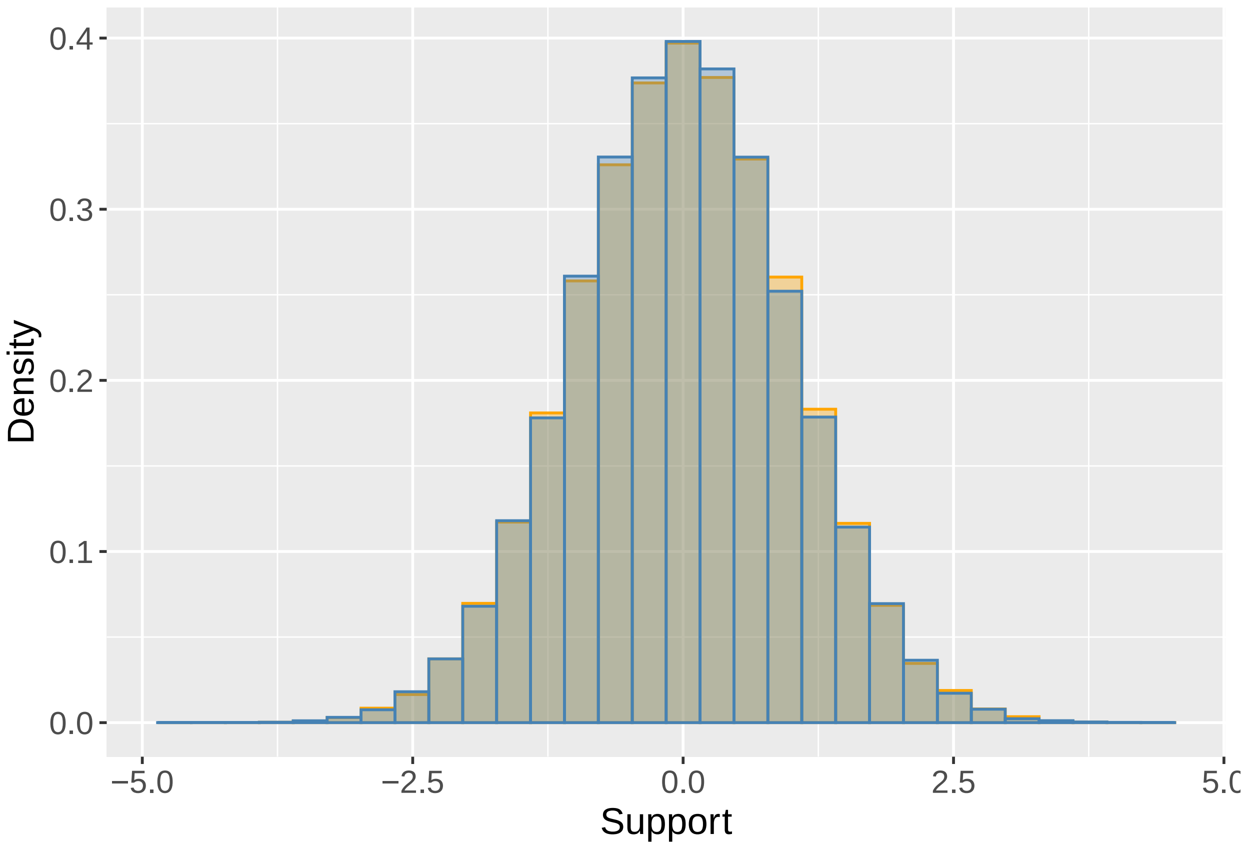
Based on a side-by-side comparison, it appears that the shift in car sales for Tianjin follows an identical distribution from 2010 to 2011. In contrast, Beijing experiences a rightward shift towards higher MSRP values as it moves from 2010 to 2011. Therefore, our counterfactual - Tianjin, suggests that license plate lottery winners in Beijing are more likely to sell their plates to individuals who are willing to pay them high prices or rewards. The same individuals who purchase black market plates are also correspondingly buying more expensive vehicles.

Computing Before-and-After Estimator

```r
set.seed(0)

n_observations <- 100000

placebo_demonstration <- data.frame(
  sample1 = rnorm(n_observations),
  sample2 = rnorm(n_observations)
)

ggplot(placebo_demonstration) +
  geom_histogram(aes(
    x = sample1,
    y = ..density..
  ), # Normalize bars so their area sum to 1
  fill = "orange", color = "orange", alpha = 0.35
  ) +
```

```r
  geom_histogram(aes(
    x = sample2,
    y = ..density..
  ), # Normalize bars so their area sum to 1
  fill = "steelblue", color = "steelblue", alpha = 0.35
  ) +
  xlab("Support") +
  ylab("Density") +
  theme(
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Exercise 4.3

Part A:

```r
set.seed(1)

placebo_1 <- data.frame(
  MSRP = Beijing_pre$MSRP,
```

```r
  count = rmultinom(
    n = 1,
    size = sum(Beijing_pre$count),
    prob = Beijing_pre$count
  )
)

head(placebo_1)
```

```
##    MSRP count
## 1 20800     0
## 2 29800    50
## 3 32900  3136
## 4 33800  3597
## 5 34800   539
## 6 36800  1804
```

Part B:

```r
set.seed(1)


placebo_2 <- data.frame(
  MSRP = Beijing_pre$MSRP,
  count = rmultinom(
    n = 1,
    size = sum(Beijing_post$count),
    prob = Beijing_pre$count
  )
)

head(placebo_2)
```

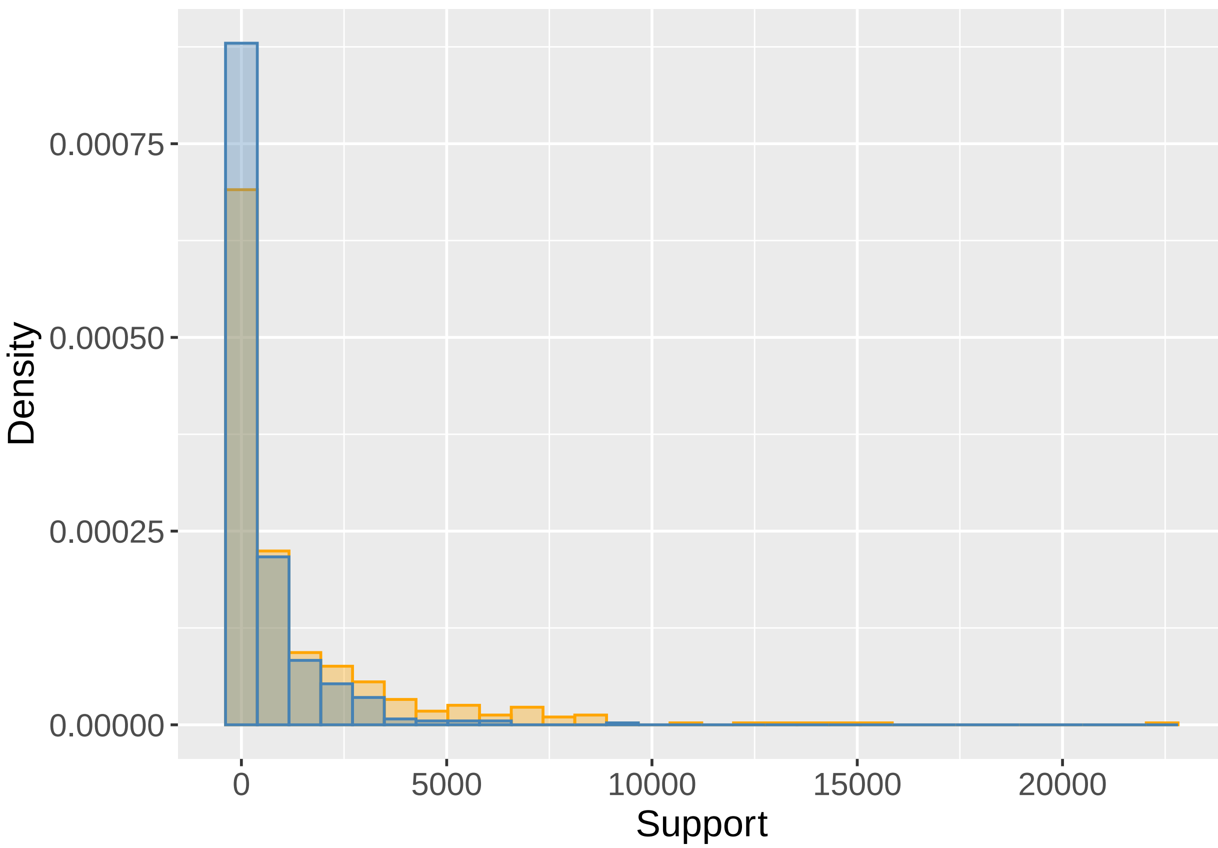
```
##    MSRP count
## 1 20800     0
## 2 29800    17
## 3 32900  1308
## 4 33800  1581
## 5 34800   243
## 6 36800   698
```

Part C:

```r
combined_data <- left_join(placebo_1, placebo_2, by = "MSRP")

ggplot(combined_data) +
  geom_histogram(aes(
    x = count.x,
    y = ..density..
  ), # Normalize bars so their area sum to 1
```

```r
  fill = "orange", color = "orange", alpha = 0.35
  ) +
  geom_histogram(aes(
    x = count.y,
    y = ..density..
  ), # Normalize bars so their area sum to 1
  fill = "steelblue", color = "steelblue", alpha = 0.35
  ) +
  xlab("Support") +
  ylab("Density") +
  theme(
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Placebo_1 and Placebo_2 do not appear to be drawn from the same distribution when we plot the number of cars sold (count variable) for each MSRP value. Placebo_2 appears to have a higher density than Placebo_1 for low MSRP values. Then Placebo_1 experiences a higher density than Placebo_2 for high MSRP values.

Optimal Transport Cost

```r
bandwidths <- c(0)
placebo_at_0 <- diftrans(
```

```
  pre_main = placebo_1,
  post_main = placebo_2,
  var = MSRP,
  bandwidth_seq = bandwidths
)
```

```
## Computing Transport Costs...
## ===============================================================================
## The transport cost for the specified bandwidths have been computed.
```

```
placebo_at_0
```

```
##   bandwidth       main
## 1         0 0.01358604
```

Exercise 4.4

Part A:

```
bandwidths <- seq(0, 100000, by = 1000)
placebo_costs <- diftrans(
  pre_main = placebo_1,
  post_main = placebo_2,
  var = MSRP,
  bandwidth_seq = bandwidths
)
```

```
## Computing Transport Costs...
## ===============================================================================
## The transport cost for the specified bandwidths have been computed.
```

```
placebo_costs
```

```
##     bandwidth         main
## 1           0 1.358604e-02
## 2        1000 6.021728e-03
## 3        2000 3.298157e-03
## 4        3000 2.868945e-03
## 5        4000 2.682622e-03
## 6        5000 1.515686e-03
## 7        6000 1.235256e-03
## 8        7000 7.531078e-04
## 9        8000 7.307365e-04
## 10       9000 6.495309e-04
## 11      10000 6.213560e-04
## 12      11000 6.117047e-04
## 13      12000 5.781043e-04
## 14      13000 5.781043e-04
## 15      14000 5.781043e-04
## 16      15000 5.781043e-04
## 17      16000 5.641203e-04
## 18      17000 5.557014e-04
```

```
## 19       18000 5.557014e-04
## 20       19000 5.557014e-04
## 21       20000 5.557014e-04
## 22       21000 5.557014e-04
## 23       22000 5.557014e-04
## 24       23000 5.557014e-04
## 25       24000 5.557014e-04
## 26       25000 4.380555e-04
## 27       26000 4.380555e-04
## 28       27000 4.380555e-04
## 29       28000 4.380555e-04
## 30       29000 4.380555e-04
## 31       30000 3.857425e-04
## 32       31000 3.851594e-04
## 33       32000 3.851594e-04
## 34       33000 3.851594e-04
## 35       34000 3.851594e-04
## 36       35000 3.851594e-04
## 37       36000 3.851594e-04
## 38       37000 3.851594e-04
## 39       38000 3.851594e-04
## 40       39000 3.851594e-04
## 41       40000 3.851594e-04
## 42       41000 3.851594e-04
## 43       42000 3.851594e-04
## 44       43000 3.851594e-04
## 45       44000 3.851594e-04
## 46       45000 3.851594e-04
## 47       46000 3.747874e-05
## 48       47000 3.747874e-05
## 49       48000 3.747874e-05
## 50       49000 3.747874e-05
## 51       50000 3.747874e-05
## 52       51000 3.747874e-05
## 53       52000 3.747874e-05
## 54       53000 3.747874e-05
## 55       54000 3.747874e-05
## 56       55000 3.747874e-05
## 57       56000 3.747874e-05
## 58       57000 3.747874e-05
## 59       58000 3.747874e-05
## 60       59000 3.747874e-05
## 61       60000 3.747874e-05
## 62       61000 3.747874e-05
## 63       62000 3.747874e-05
## 64       63000 3.747874e-05
## 65       64000 3.747874e-05
## 66       65000 3.747874e-05
## 67       66000 3.747874e-05
## 68       67000 1.342076e-05
## 69       68000 1.342076e-05
## 70       69000 1.342076e-05
## 71       70000 1.342076e-05
## 72       71000 1.342076e-05
```

```
## 73      72000 1.342076e-05
## 74      73000 1.342076e-05
## 75      74000 1.342076e-05
## 76      75000 1.342076e-05
## 77      76000 1.342076e-05
## 78      77000 1.342076e-05
## 79      78000 1.342076e-05
## 80      79000 1.342076e-05
## 81      80000 1.342076e-05
## 82      81000 1.342076e-05
## 83      82000 1.342076e-05
## 84      83000 1.342076e-05
## 85      84000 1.342076e-05
## 86      85000 1.342076e-05
## 87      86000 1.342076e-05
## 88      87000 1.342076e-05
## 89      88000 1.342076e-05
## 90      89000 1.342076e-05
## 91      90000 1.342076e-05
## 92      91000 1.342076e-05
## 93      92000 1.342076e-05
## 94      93000 1.342076e-05
## 95      94000 1.342076e-05
## 96      95000 1.342076e-05
## 97      96000 1.342076e-05
## 98      97000 1.342076e-05
## 99      98000 1.342076e-05
## 100     99000 1.342076e-05
## 101    100000 1.342076e-05
```

Part B:

```r
bandwidths <- seq(0, 100000, by = 1000)
Beijing_costs <- diftrans(
  pre_main = Beijing_pre,
  post_main = Beijing_post,
  var = MSRP,
  bandwidth_seq = bandwidths
)
```

```
## Computing Transport Costs...
```

```
## ===============================================================================
```

```
## The transport cost for the specified bandwidths have been computed.
```

```r
Beijing_costs
```

```
##     bandwidth       main
## 1           0 0.35312341
## 2        1000 0.25894833
## 3        2000 0.21775576
## 4        3000 0.20258493
## 5        4000 0.18491942
```

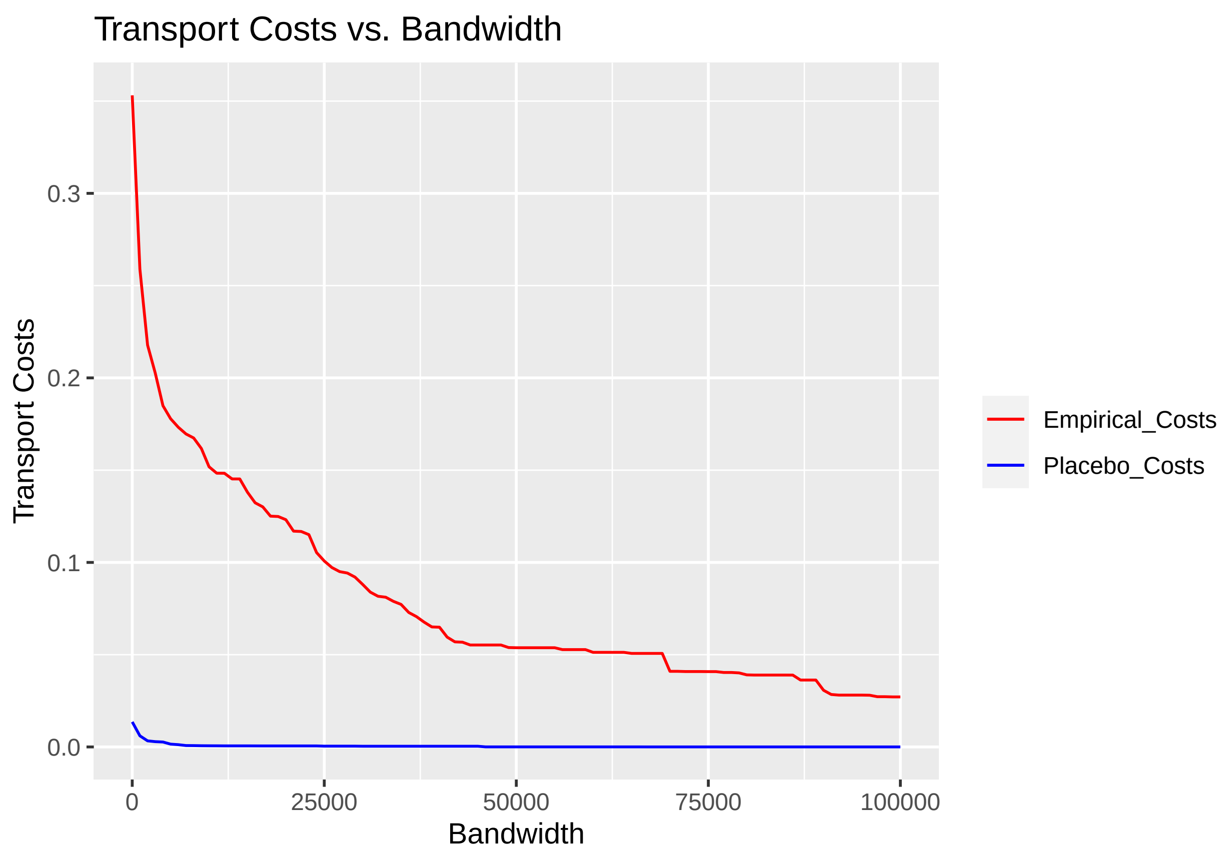
```
## 6        5000 0.17785943
## 7        6000 0.17324334
## 8        7000 0.16961528
## 9        8000 0.16742104
## 10       9000 0.16173789
## 11      10000 0.15183081
## 12      11000 0.14834303
## 13      12000 0.14831396
## 14      13000 0.14522546
## 15      14000 0.14522546
## 16      15000 0.13805858
## 17      16000 0.13232015
## 18      17000 0.13009871
## 19      18000 0.12506473
## 20      19000 0.12488444
## 21      20000 0.12315956
## 22      21000 0.11695671
## 23      22000 0.11676131
## 24      23000 0.11503643
## 25      24000 0.10532848
## 26      25000 0.10076861
## 27      26000 0.09721188
## 28      27000 0.09501764
## 29      28000 0.09424945
## 30      29000 0.09205521
## 31      30000 0.08807506
## 32      31000 0.08389676
## 33      32000 0.08165662
## 34      33000 0.08114887
## 35      34000 0.07890873
## 36      35000 0.07725062
## 37      36000 0.07286872
## 38      37000 0.07063429
## 39      38000 0.06763654
## 40      39000 0.06503869
## 41      40000 0.06488865
## 42      41000 0.05951341
## 43      42000 0.05696092
## 44      43000 0.05676552
## 45      44000 0.05525308
## 46      45000 0.05525308
## 47      46000 0.05525308
## 48      47000 0.05525308
## 49      48000 0.05525308
## 50      49000 0.05386946
## 51      50000 0.05376216
## 52      51000 0.05376216
## 53      52000 0.05375780
## 54      53000 0.05375780
## 55      54000 0.05375780
## 56      55000 0.05375780
## 57      56000 0.05275368
## 58      57000 0.05275368
## 59      58000 0.05275368
```

```
## 60      59000 0.05275368
## 61      60000 0.05126130
## 62      61000 0.05126130
## 63      62000 0.05126130
## 64      63000 0.05126130
## 65      64000 0.05126130
## 66      65000 0.05068877
## 67      66000 0.05068877
## 68      67000 0.05068877
## 69      68000 0.05068877
## 70      69000 0.05068877
## 71      70000 0.04098760
## 72      71000 0.04098760
## 73      72000 0.04086990
## 74      73000 0.04086990
## 75      74000 0.04086990
## 76      75000 0.04081468
## 77      76000 0.04081468
## 78      77000 0.04035258
## 79      78000 0.04035258
## 80      79000 0.04012880
## 81      80000 0.03906051
## 82      81000 0.03895176
## 83      82000 0.03895176
## 84      83000 0.03895176
## 85      84000 0.03895176
## 86      85000 0.03895176
## 87      86000 0.03894885
## 88      87000 0.03625473
## 89      88000 0.03625473
## 90      89000 0.03625327
## 91      90000 0.03070519
## 92      91000 0.02842453
## 93      92000 0.02810407
## 94      93000 0.02810407
## 95      94000 0.02810407
## 96      95000 0.02810407
## 97      96000 0.02801978
## 98      97000 0.02723195
## 99      98000 0.02723195
## 100     99000 0.02712029
## 101    100000 0.02712029
```

Part C:

```r
results_plot <- data.frame(
  bandwidth = bandwidths,
  placebo_costs = placebo_costs$main,
  Beijing_costs = Beijing_costs$main
)

ggplot(results_plot, aes(x = bandwidth)) +
```

```
  geom_line(aes(y = placebo_costs, color = "Placebo_Costs")) +
  geom_line(aes(y = Beijing_costs, color = "Empirical_Costs")) +
  xlab("Bandwidth") +
  ylab("Transport Costs") +
  ggtitle("Transport Costs vs. Bandwidth") +
  scale_color_manual("", values = c(
    "Placebo_Costs" = "blue",
    "Empirical_Costs" = "red"
  ))
```



Part D:

```
threshold <- 0.0005

benchmark_test <- which(placebo_costs$main < threshold)

extract_values <- placebo_costs$bandwidth[benchmark_test]

print(extract_values)
```

```
##  [1]  25000  26000  27000  28000  29000  30000  31000  32000  33000  34000
## [11]  35000  36000  37000  38000  39000  40000  41000  42000  43000  44000
## [21]  45000  46000  47000  48000  49000  50000  51000  52000  53000  54000
## [31]  55000  56000  57000  58000  59000  60000  61000  62000  63000  64000
## [41]  65000  66000  67000  68000  69000  70000  71000  72000  73000  74000
```

```
## [51]  75000  76000  77000  78000  79000  80000  81000  82000  83000  84000
## [61]  85000  86000  87000  88000  89000  90000  91000  92000  93000  94000
## [71]  95000  96000  97000  98000  99000 100000
```

Part E:

The smallest value of d found in the previous step is equal to 25,000. Thus, the corresponding empirical transport cost is equal to 0.10076861

Computing Differences-in-Transports Estimator

```r
dit_at_0 <- diftrans(
  pre_main = Beijing_pre,
  post_main = Beijing_post,
  pre_control = Tianjin_pre,
  post_control = Tianjin_post,
  var = MSRP,
  bandwidth_seq = c(0),
  conservative = TRUE
)
```

```
## Computing Differences-in-Transports Estimator...

## Note: you are using `conservative = T`.

## ==========================================================================

## The conservative diff-in-transports estimator is 0.0544428953078284 at d = 0
```

```r
dit_at_0
```

```
##   bandwidth      main    main2d   control      diff    diff2d
## 1         0 0.3531234 0.3531234 0.2986805 0.0544429 0.0544429
```

Exercise 4.5

Part A:

```r
dit_values <- diftrans(
  pre_main = Beijing_pre,
  post_main = Beijing_post,
  pre_control = Tianjin_pre,
  post_control = Tianjin_post,
  var = MSRP,
  bandwidth_seq = seq(0, 50000, by = 1000),
  conservative = TRUE
)
```

```
## Computing Differences-in-Transports Estimator...

## Note: you are using `conservative = T`.

## ==========================================================================
```

```
## The conservative diff-in-transports estimator is 0.120031189530433 at d = 7000
```

```
dit_values
```

```
##    bandwidth       main     main2d     control       diff     diff2d
## 1          0 0.35312341 0.35312341 0.298680517 0.05444290 0.05444290
## 2       1000 0.25894833 0.21775576 0.177321051 0.08162728 0.04043471
## 3       2000 0.21775576 0.18491942 0.113612872 0.10414289 0.07130655
## 4       3000 0.20258493 0.17324334 0.083446548 0.11913839 0.08979679
## 5       4000 0.18491942 0.16742104 0.065551675 0.11936775 0.10186937
## 6       5000 0.17785943 0.15183081 0.045616551 0.13224288 0.10621426
## 7       6000 0.17324334 0.14831396 0.039409021 0.13383432 0.10890494
## 8       7000 0.16961528 0.14522546 0.025194272 0.14442101 0.12003119
## 9       8000 0.16742104 0.13232015 0.024617556 0.14280348 0.10770260
## 10      9000 0.16173789 0.12506473 0.023744170 0.13799372 0.10132056
## 11     10000 0.15183081 0.12315956 0.020093309 0.13173750 0.10306625
## 12     11000 0.14834303 0.11676131 0.019649685 0.12869334 0.09711162
## 13     12000 0.14831396 0.10532848 0.018437809 0.12987615 0.08689067
## 14     13000 0.14522546 0.09721188 0.018414930 0.12681053 0.07879695
## 15     14000 0.14522546 0.09424945 0.018414930 0.12681053 0.07583452
## 16     15000 0.13805858 0.08807506 0.017838214 0.12022036 0.07023684
## 17     16000 0.13232015 0.08165662 0.017820869 0.11449929 0.06383575
## 18     17000 0.13009871 0.07890873 0.017820869 0.11227784 0.06108786
## 19     18000 0.12506473 0.07286872 0.017820869 0.10724386 0.05504785
## 20     19000 0.12488444 0.06763654 0.014596328 0.11028811 0.05304021
## 21     20000 0.12315956 0.06488865 0.013055904 0.11010366 0.05183275
## 22     21000 0.11695671 0.05696092 0.011043218 0.10591349 0.04591770
## 23     22000 0.11676131 0.05525308 0.011043218 0.10571809 0.04420986
## 24     23000 0.11503643 0.05525308 0.007405150 0.10763128 0.04784793
## 25     24000 0.10532848 0.05525308 0.007400309 0.09792817 0.04785277
## 26     25000 0.10076861 0.05376216 0.007158412 0.09361020 0.04660375
## 27     26000 0.09721188 0.05375780 0.007158412 0.09005347 0.04659939
## 28     27000 0.09501764 0.05375780 0.007158412 0.08785923 0.04659939
## 29     28000 0.09424945 0.05275368 0.007153570 0.08709588 0.04560011
## 30     29000 0.09205521 0.05275368 0.007153570 0.08490164 0.04560011
## 31     30000 0.08807506 0.05126130 0.006840835 0.08123422 0.04442047
## 32     31000 0.08389676 0.05126130 0.006662545 0.07723421 0.04459876
## 33     32000 0.08165662 0.05126130 0.006662545 0.07499407 0.04459876
## 34     33000 0.08114887 0.05068877 0.006662545 0.07448633 0.04402622
## 35     34000 0.07890873 0.05068877 0.006662545 0.07224619 0.04402622
## 36     35000 0.07725062 0.04098760 0.006662545 0.07058807 0.03432506
## 37     36000 0.07286872 0.04086990 0.006662545 0.06620618 0.03420736
## 38     37000 0.07063429 0.04086990 0.006662545 0.06397174 0.03420736
## 39     38000 0.06763654 0.04081468 0.006662545 0.06097400 0.03415214
## 40     39000 0.06503869 0.04035258 0.006657703 0.05838099 0.03369488
## 41     40000 0.06488865 0.03906051 0.006376887 0.05851177 0.03268362
## 42     41000 0.05951341 0.03895176 0.004575792 0.05493762 0.03437596
## 43     42000 0.05696092 0.03895176 0.004575792 0.05238513 0.03437596
## 44     43000 0.05676552 0.03894885 0.004575792 0.05218973 0.03437306
## 45     44000 0.05525308 0.03625473 0.004319338 0.05093374 0.03193539
## 46     45000 0.05525308 0.03070519 0.004319338 0.05093374 0.02638585
## 47     46000 0.05525308 0.02810407 0.004319338 0.05093374 0.02378473
## 48     47000 0.05525308 0.02810407 0.004309655 0.05094342 0.02379441
## 49     48000 0.05525308 0.02801978 0.004309655 0.05094342 0.02371013
## 50     49000 0.05386946 0.02723195 0.004309655 0.04955980 0.02292229
```

```
## 51      50000 0.05376216 0.02712029 0.004309655 0.04945250 0.02281064
```

Part B:

```r
placebo_Beijing_1 <- data.frame(
  MSRP = Beijing_pre$MSRP,
  count = rmultinom(
    n = 1,
    size = sum(Beijing_pre$count),
    prob = Beijing_pre$count
  )
)
```

Part C:

```r
placebo_Beijing_2 <- data.frame(
  MSRP = Beijing_pre$MSRP,
  count = rmultinom(
    n = 1,
    size = sum(Beijing_post$count),
    prob = Beijing_pre$count
  )
)
```

Part D:

```r
placebo_Tianjin_1 <- data.frame(
  MSRP = Tianjin_pre$MSRP,
  count = rmultinom(
    n = 1,
    size = sum(Tianjin_pre$count),
    prob = Tianjin_pre$count
  )
)
```

Part E:

```r
placebo_Tianjin_2 <- data.frame(
  MSRP = Tianjin_pre$MSRP,
  count = rmultinom(
    n = 1,
    size = sum(Tianjin_post$count),
    prob = Tianjin_pre$count
  )
)
```

Part F:

```r
dit_values_new <- diftrans(
  pre_main = placebo_Beijing_1,
  post_main = placebo_Beijing_2,
  pre_control = placebo_Tianjin_1,
  post_control = placebo_Tianjin_2,
  var = MSRP,
  bandwidth_seq = seq(0, 50000, by = 1000),
  conservative = TRUE
)
```

```
## Computing Differences-in-Transports Estimator...

## Note: you are using `conservative = T`.

## ============================================================================

## The conservative diff-in-transports estimator is -1.78382371263168e-05 at d = 48000
```

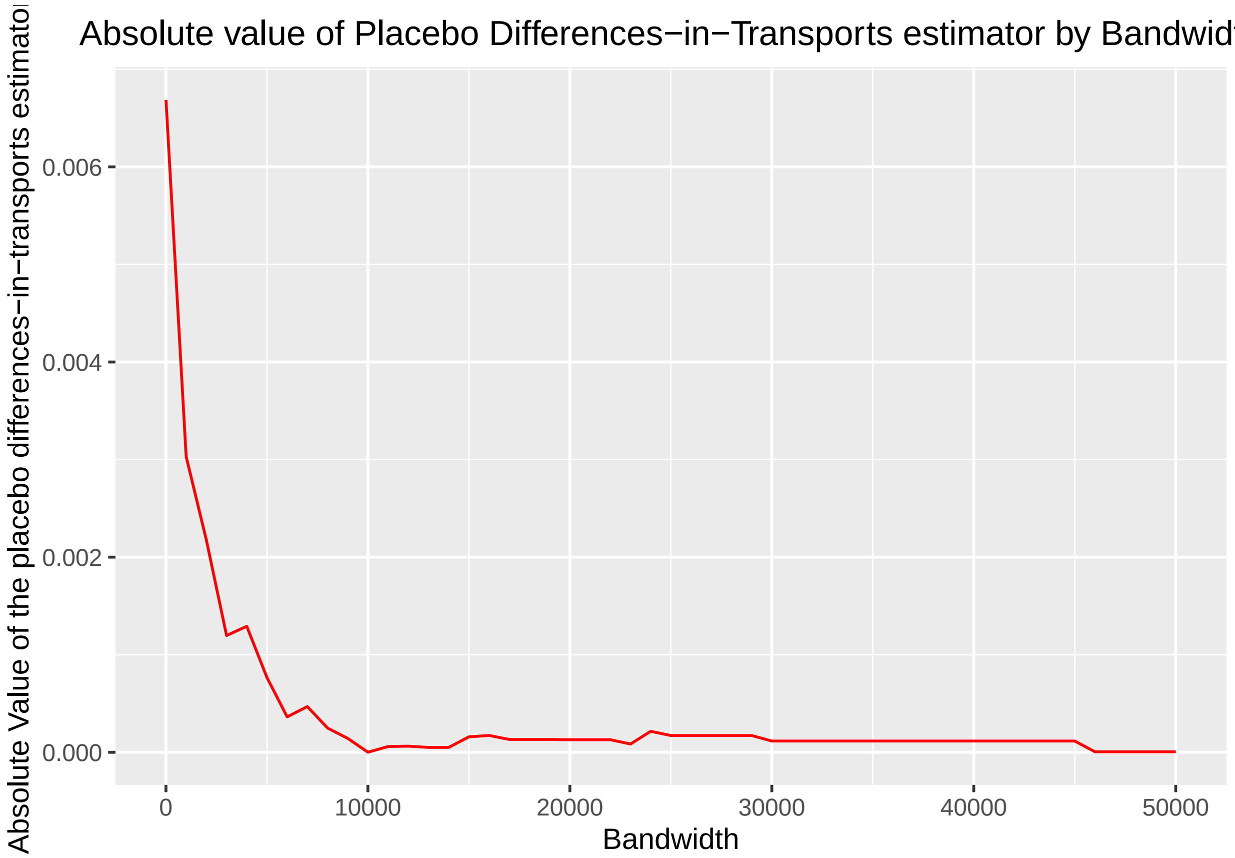
```r
dit_values_new
```

```
##    bandwidth         main       main2d      control          diff        diff2d
## 1          0 1.259404e-02 1.259404e-02 1.928028e-02 -6.686238e-03 -6.686238e-03
## 2       1000 4.964057e-03 2.651280e-03 7.994191e-03 -3.030133e-03 -5.342911e-03
## 3       2000 2.651280e-03 1.768619e-03 4.825979e-03 -2.174699e-03 -3.057360e-03
## 4       3000 1.931387e-03 1.425867e-03 3.127969e-03 -1.196582e-03 -1.702102e-03
## 5       4000 1.768619e-03 9.212687e-04 3.060053e-03 -1.291434e-03 -2.138784e-03
## 6       5000 1.495010e-03 7.776741e-04 2.260838e-03 -7.658280e-04 -1.483164e-03
## 7       6000 1.425867e-03 7.151891e-04 1.788863e-03 -3.629959e-04 -1.073674e-03
## 8       7000 9.282562e-04 7.018327e-04 1.396823e-03 -4.685667e-04 -6.949903e-04
## 9       8000 9.212687e-04 5.576214e-04 1.169632e-03 -2.483637e-04 -6.120110e-04
## 10      9000 8.473555e-04 5.331962e-04 9.897414e-04 -1.423859e-04 -4.565453e-04
## 11     10000 7.776741e-04 5.331962e-04 7.782847e-04 -6.106057e-07 -2.450885e-04
## 12     11000 7.186784e-04 5.331962e-04 7.779132e-04 -5.923483e-05 -2.447171e-04
## 13     12000 7.151891e-04 4.964976e-04 7.779132e-04 -6.272414e-05 -2.814157e-04
## 14     13000 7.018327e-04 3.213803e-04 7.518959e-04 -5.006328e-05 -4.305156e-04
## 15     14000 7.018327e-04 3.213803e-04 7.518959e-04 -5.006328e-05 -4.305156e-04
## 16     15000 5.934332e-04 2.550833e-04 7.518959e-04 -1.584628e-04 -4.968126e-04
## 17     16000 5.576214e-04 2.550833e-04 7.299987e-04 -1.723773e-04 -4.749154e-04
## 18     17000 5.331962e-04 2.550833e-04 6.649555e-04 -1.317594e-04 -4.098722e-04
## 19     18000 5.331962e-04 2.550833e-04 6.649555e-04 -1.317594e-04 -4.098722e-04
## 20     19000 5.331962e-04 2.550833e-04 6.649555e-04 -1.317594e-04 -4.098722e-04
## 21     20000 5.331962e-04 2.550833e-04 6.622156e-04 -1.290194e-04 -4.071323e-04
## 22     21000 5.331962e-04 2.550833e-04 6.622156e-04 -1.290194e-04 -4.071323e-04
## 23     22000 5.331962e-04 2.550833e-04 6.622156e-04 -1.290194e-04 -4.071323e-04
## 24     23000 5.124821e-04 3.077826e-05 5.971724e-04 -8.469028e-05 -5.663941e-04
## 25     24000 4.964976e-04 3.077826e-05 2.813280e-04  2.151696e-04 -2.505497e-04
## 26     25000 3.213803e-04 3.077826e-05 1.493279e-04  1.720524e-04 -1.185496e-04
## 27     26000 3.213803e-04 3.077826e-05 1.493279e-04  1.720524e-04 -1.185496e-04
## 28     27000 3.213803e-04 3.077826e-05 1.493279e-04  1.720524e-04 -1.185496e-04
## 29     28000 3.213803e-04 3.077826e-05 1.493279e-04  1.720524e-04 -1.185496e-04
## 30     29000 3.213803e-04 3.077826e-05 1.493279e-04  1.720524e-04 -1.185496e-04
## 31     30000 2.550833e-04 3.077826e-05 1.396446e-04  1.154387e-04 -1.088663e-04
## 32     31000 2.550833e-04 3.077826e-05 1.396446e-04  1.154387e-04 -1.088663e-04
## 33     32000 2.550833e-04 3.077826e-05 1.396446e-04  1.154387e-04 -1.088663e-04
```

```
## 34     33000 2.550833e-04 3.077826e-05 1.396446e-04  1.154387e-04 -1.088663e-04
## 35     34000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 36     35000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 37     36000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 38     37000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 39     38000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 40     39000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 41     40000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 42     41000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 43     42000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 44     43000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 45     44000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 46     45000 2.550833e-04 7.886355e-06 1.396446e-04  1.154387e-04 -1.317582e-04
## 47     46000 3.077826e-05 7.886355e-06 2.572459e-05  5.053666e-06 -1.783824e-05
## 48     47000 3.077826e-05 7.886355e-06 2.572459e-05  5.053666e-06 -1.783824e-05
## 49     48000 3.077826e-05 7.886355e-06 2.572459e-05  5.053666e-06 -1.783824e-05
## 50     49000 3.077826e-05 7.886355e-06 2.572459e-05  5.053666e-06 -1.783824e-05
## 51     50000 3.077826e-05 7.886355e-06 2.572459e-05  5.053666e-06 -1.783824e-05
```

Part G:

```r
ggplot(dit_values_new, aes(x = bandwidth, y = abs(diff))) +
  geom_line(color = "red") +
  labs(
    x = "Bandwidth",
    y = "Absolute Value of the placebo differences-in-transports estimator",
    title = "Absolute value of Placebo Differences-in-Transports estimator by Bandwidth"
  ) +
  theme(plot.title = element_text(hjust = 0.5))
```



Part H:

```
threshold <- 0.0005

benchmark_test_2 <- which(abs(dit_values_new$diff) < threshold)

extract_values_2 <- dit_values_new$bandwidth[benchmark_test_2]

print(extract_values_2)
```

```
##  [1]  6000  7000  8000  9000 10000 11000 12000 13000 14000 15000 16000 17000
## [13] 18000 19000 20000 21000 22000 23000 24000 25000 26000 27000 28000 29000
## [25] 30000 31000 32000 33000 34000 35000 36000 37000 38000 39000 40000 41000
## [37] 42000 43000 44000 45000 46000 47000 48000 49000 50000
```

Part I

```
extract_values_2 <- as.data.frame(extract_values_2)

names(extract_values_2)[1] <- "bandwidth"

largest_diff <- left_join(extract_values_2, dit_values_new, by = "bandwidth")
```

```r
largest_diff %>%
  select(bandwidth, diff) %>%
  arrange(desc(abs(diff)))
```

```
##    bandwidth          diff
## 1       7000 -4.685667e-04
## 2       6000 -3.629959e-04
## 3       8000 -2.483637e-04
## 4      24000  2.151696e-04
## 5      16000 -1.723773e-04
## 6      29000  1.720524e-04
## 7      26000  1.720524e-04
## 8      28000  1.720524e-04
## 9      27000  1.720524e-04
## 10     25000  1.720524e-04
## 11     15000 -1.584628e-04
## 12      9000 -1.423859e-04
## 13     18000 -1.317594e-04
## 14     19000 -1.317594e-04
## 15     17000 -1.317594e-04
## 16     20000 -1.290194e-04
## 17     21000 -1.290194e-04
## 18     22000 -1.290194e-04
## 19     31000  1.154387e-04
## 20     30000  1.154387e-04
## 21     39000  1.154387e-04
## 22     35000  1.154387e-04
## 23     44000  1.154387e-04
## 24     32000  1.154387e-04
## 25     34000  1.154387e-04
## 26     37000  1.154387e-04
## 27     38000  1.154387e-04
## 28     36000  1.154387e-04
## 29     42000  1.154387e-04
## 30     43000  1.154387e-04
## 31     45000  1.154387e-04
## 32     40000  1.154387e-04
## 33     33000  1.154387e-04
## 34     41000  1.154387e-04
## 35     23000 -8.469028e-05
## 36     12000 -6.272414e-05
## 37     11000 -5.923483e-05
## 38     14000 -5.006328e-05
## 39     13000 -5.006328e-05
## 40     47000  5.053666e-06
## 41     46000  5.053666e-06
## 42     48000  5.053666e-06
## 43     50000  5.053666e-06
## 44     49000  5.053666e-06
## 45     10000 -6.106057e-07
```

Among all the values of d that we found in the previous step, the one that yielded the largest value for the placebo differences-in-transports estimator is bandwidth = 6000, with a diff = 4.974995e-04. This is the actual difference-in-transports estimator.

END