p8105_hw5_yz4433

Yifei Zhao

2022-11-08

Problem 1

The code chunk below imports the data in individual spreadsheets contained in `./data/zip_data/` . To do this, I create a dataframe that includes the list of all files in that directory and the complete path to each file. As a next step, I `map` over paths and import data using the `read_csv` function. Finally, I `unnest` the result of `map` .

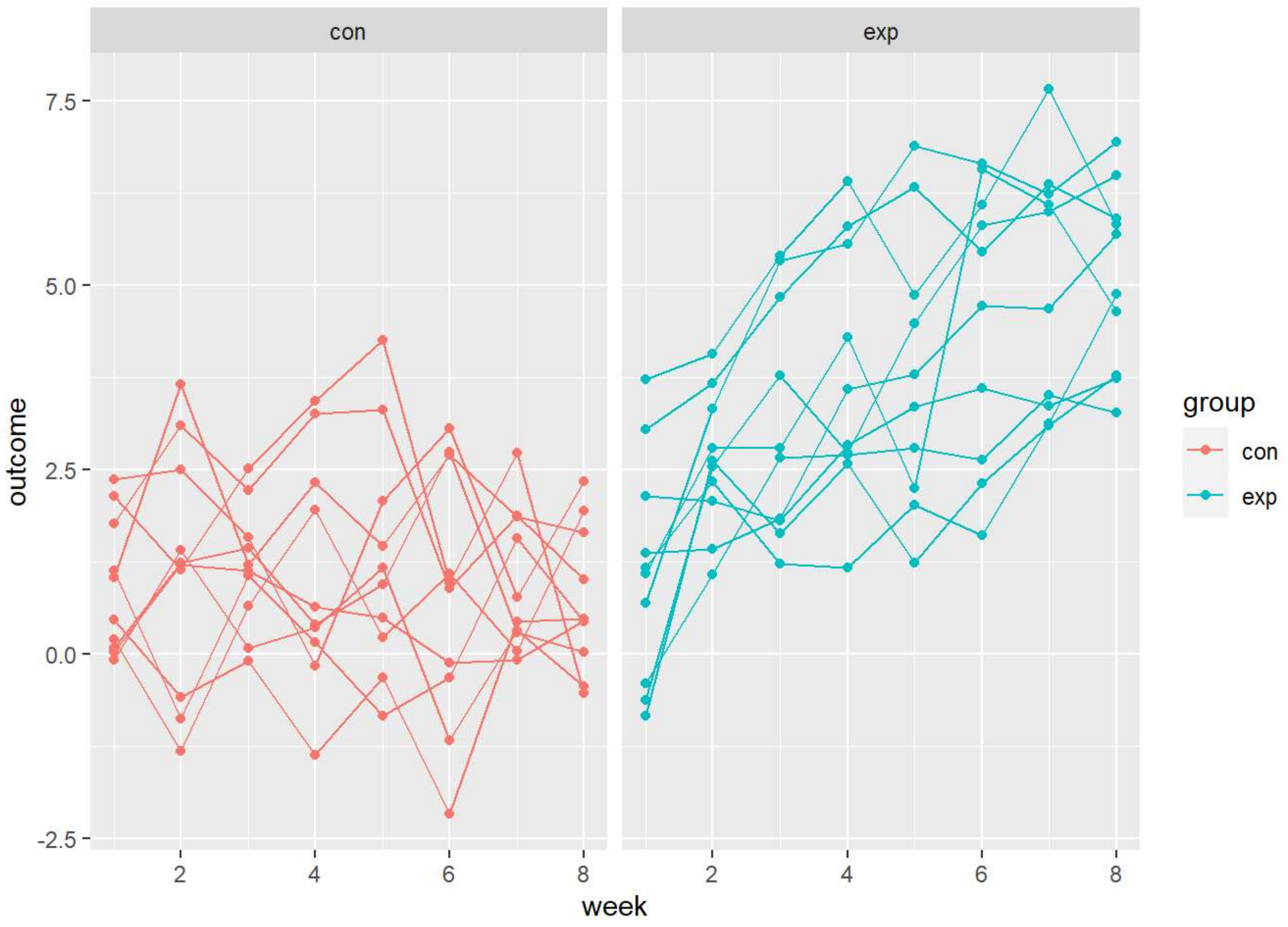
```
full_df = 
  tibble(
    files = list.files("data/zip_data/"),
    path = str_c("data/zip_data/", files)
  ) %>% 
  mutate(data = map(path, read_csv)) %>% 
  unnest()
```

The result of the previous code chunk isn't tidy – data are wide rather than long, and some important variables are included as parts of others. The code chunk below tides the data using string manipulations on the file, converting from wide to long, and selecting relevant variables.

```
tidy_df = 
  full_df %>% 
  mutate(
    files = str_replace(files, ".csv", ""),
    group = str_sub(files, 1, 3)) %>% 
  pivot_longer(
    week_1:week_8,
    names_to = "week",
    values_to = "outcome",
    names_prefix = "week_") %>% 
  mutate(week = as.numeric(week)) %>% 
  select(group, subj = files, week, outcome)
```

Finally, the code chunk below creates a plot showing individual data, faceted by group.

```
tidy_df %>% 
  ggplot(aes(x = week, y = outcome, group = subj, color = group)) + 
  geom_point() + 
  geom_path() + 
  facet_grid(~group)
```



This plot suggests high within-subject correlation – subjects who start above average end up above average, and those that start below average end up below average. Subjects in the control group generally don't change over time, but those in the experiment group increase their outcome in a roughly linear way.

Problem 2

```
hddata = read.csv("https://raw.githubusercontent.com/washingtonpost/data-homicides/master/homic
ide-data.csv")
```

For homicide-data, a 52179 $\times$ 12 dataset, the key variables are city, victim information variables, and deposition, describing the location of the homicide, basic demographic information about each victim, and mapped with whether an arrest was made over the past decade in 50 of the largest American cities. Meanwhile, there are no missing values for disposition and the dataset is mostly tidy in a whole view.

Homecides summary

```
tidydata = hddata %>%
  mutate(city_state = paste(city,',',state))

sumdata = tidydata %>%
  mutate(group = ifelse((disposition == 'Closed by arrest'), 'solved', 'unsolved')) %>%
  group_by(city_state) %>%
  summarise(
    total = sum(group == 'solved' | group == 'unsolved'  ),
    unsolved = sum(group == 'unsolved'))
```

prop.test for Baltimore, MD

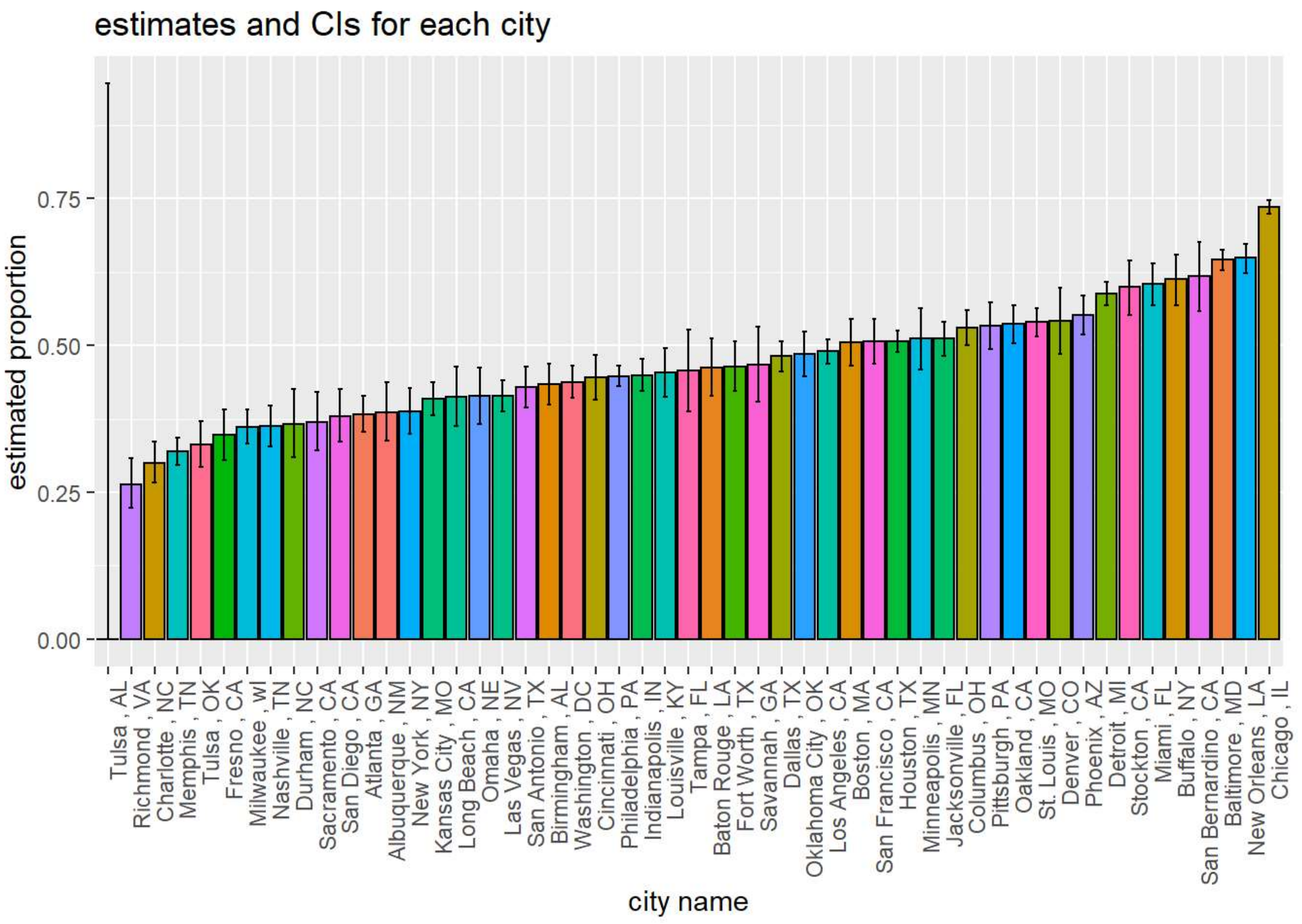
```
rsbal = prop.test(sumdata[[1,3]], sumdata[[1,2]]) %>%
  broom::tidy()
estbal = pull(rsbal, estimate)
cibal = c(pull(rsbal, conf.low), pull(rsbal, conf.high))
```

prop.test for each city

```
test_fun = function(x, y){data.frame(
  estimation = pull(broom::tidy(prop.test(x,y)), estimate),
  ci_low = pull(broom::tidy(prop.test(x,y)), conf.low),
  ci_high = pull(broom::tidy(prop.test(x,y)), conf.high))
}

listcol_df = sumdata %>%
  mutate(testrs = map2(pull(sumdata, unsolved), pull(sumdata, total), test_fun)) %>%
  select(-c(total, unsolved)) %>%
  unnest(cols = testrs)
```

estimates and CIs plot

```
p = ggplot(listcol_df, aes(x = reorder(city_state, estimation), y = estimation, fill = city_sta
te)) +
  geom_bar(stat = "identity", color = "black", position = position_dodge()) +
  geom_errorbar(aes(ymin = ci_low, ymax = ci_high), width = .2, position = position_dodge(.9))
 +
  theme(legend.position = "none") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(x = "city name", y = "estimated proportion", title = "estimates and CIs for each city")
print(p)
```



Problem 3

data with mean = 0

```
output0 = vector("list", 5000)
for (i in 1:5000) {
  output0[[i]] = tibble(x = rnorm(30, mean = 0, sd = 5))
}
```

```
paralist0 = vector("list", 5000)
pvlist0 = vector("list", 5000)
for (i in 1:5000) {
  paralist0[[i]] = pull(broom::tidy(t.test(unlist(output0[i]), mu = 0, alternative = "two.side
d")), estimate)
  pvlist0[[i]] = pull(broom::tidy(t.test(unlist(output0[i]), mu = 0, alternative = "two.sided"
)), p.value)
}
```

other groups

```
ttest_fun = function(m){
  output = vector("list", 5000)
  for (i in 1:5000) {
    output[[i]] = tibble(x = rnorm(30, mean = m, sd = 5))
  }
  paralist = vector("list", 5000)
  pvlist = vector("list", 5000)
  for (i in 1:5000) {
    paralist[[i]] = pull(broom::tidy(t.test(unlist(output[i]), mu = 0, alternative = "two.side
d")), estimate)
    pvlist[[i]] = pull(broom::tidy(t.test(unlist(output[i]), mu = 0, alternative = "two.sided"
)), p.value)
  }
  resl = list(paralist, pvlist)
  resl = matrix(unlist(resl),byrow = FALSE,ncol = 2) %>%
    data.frame()
  }
```

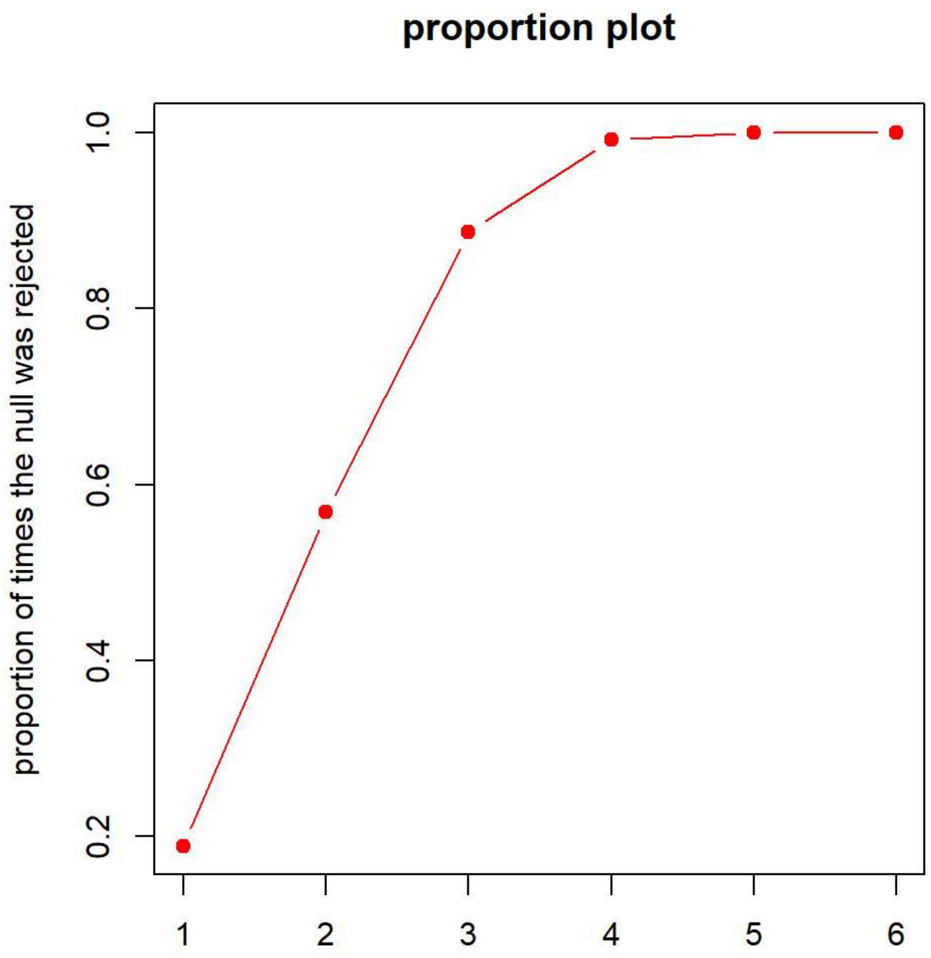
groups generation

```
t1 = ttest_fun(1)
t2 = ttest_fun(2)
t3 = ttest_fun(3)
t4 = ttest_fun(4)
t5 = ttest_fun(5)
t6 = ttest_fun(6)
```

proportion calculation

proportion plot

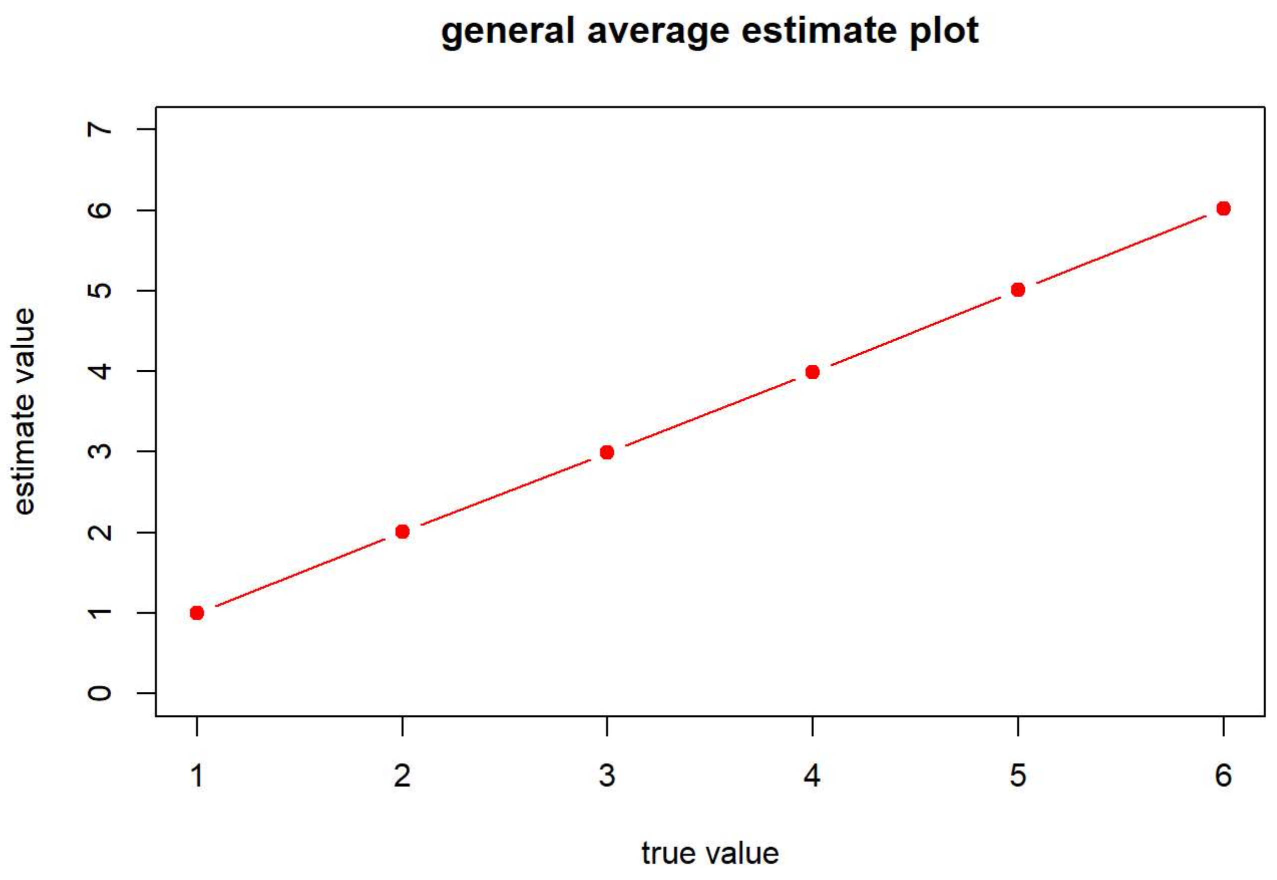
```
list0 = list('1', '2', '3', '4', '5', '6')
list1 = list(nrow(filter(t1, X2 <= 0.05))/5000, nrow(filter(t2, X2 <= 0.05))/5000, nrow(filter
(t3, X2 <= 0.05))/5000, nrow(filter(t4, X2 <= 0.05))/5000, nrow(filter(t5, X2 <= 0.05))/5000, n
row(filter(t6, X2 <= 0.05))/5000)

par(pin = c(4,4))
plot(list0, list1, type = "b", pch = 19,
     col = "red", xlab = "mean", ylab = "proportion of times the null was rejected", main = 'pr
oportion plot')
```



From the plot, under the same sample size, as the mean of simulated sample increases, the proportion of times the null was rejected increases, which means if we do not change the sample size, and the effect size of the intervention is large, the likelihood of existing an effect is high, equivalently the power is high.

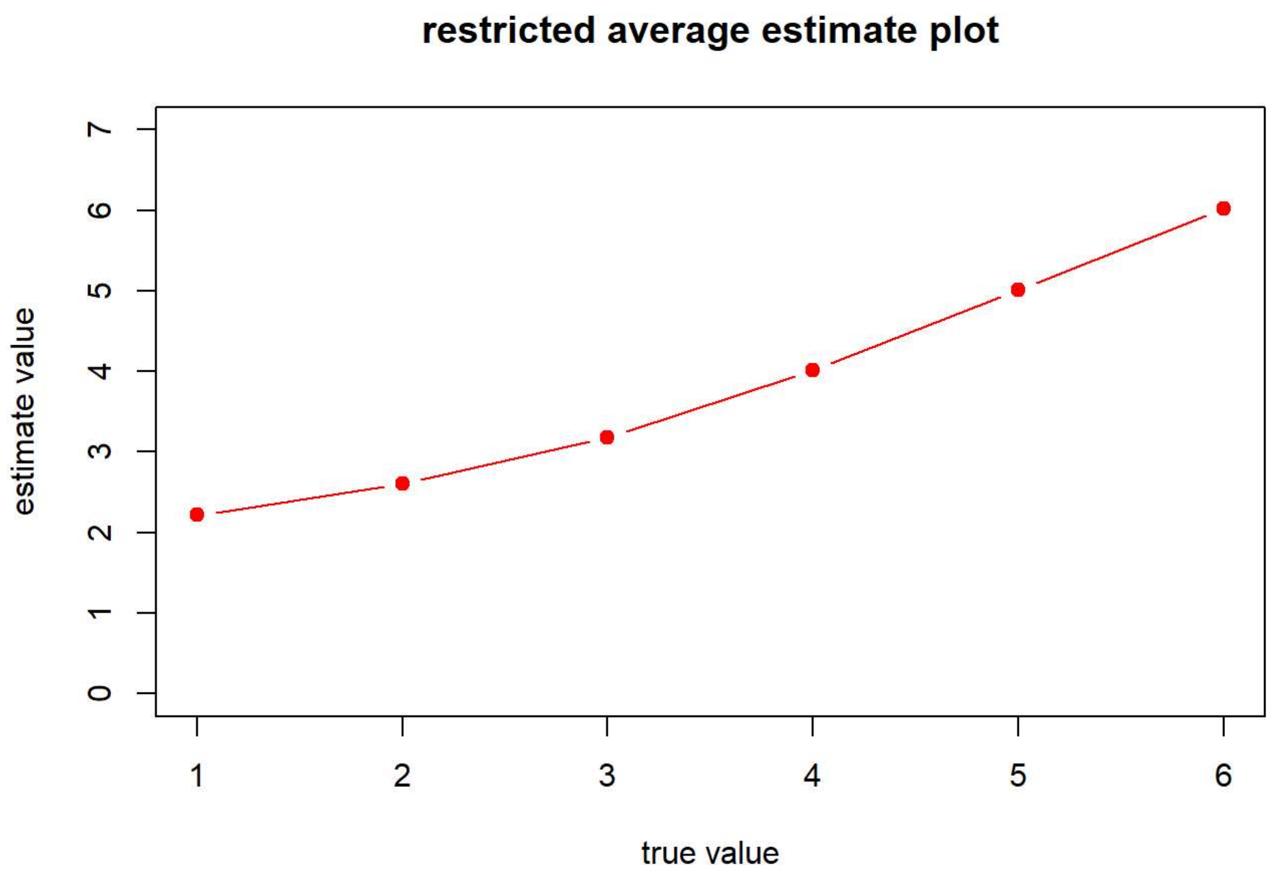
general plot

```
list2 = list(mean(t1[[1]]), mean(t2[[1]]), mean(t3[[1]]), mean(t4[[1]]), mean(t5[[1]]), mean(t6
[[1]]))
plot(list0, list2, type = "b", pch = 19, ylim = c(0,7),
     col = "red", xlab = "true value", ylab = "estimate value", main = 'general average estimat
e plot')
```



restricted plot

```
list2 = list(mean(filter(t1, X2 <= 0.05)[[1]]), mean(filter(t2, X2 <= 0.05)[[1]]), mean(filter
(t3, X2 <= 0.05)[[1]]), mean(filter(t4, X2 <= 0.05)[[1]]), mean(filter(t5, X2 <= 0.05)[[1]]), m
ean(filter(t6, X2 <= 0.05)[[1]]))
plot(list0, list2, type = "b", pch = 19, ylim = c(0,7),
     col = "red", xlab = "true value", ylab = "estimate value", main = 'restricted average esti
mate plot')
```



Comparing the restricted plot with general plot for average estimate of $\hat{\mu}$ under different μ, we clearly find the sample average of $\hat{\mu}$ across tests for which the null is rejected approximately is not equal to the true value of μ.

It is mainly because small effect size implies there is a not big part of reject, and thus the average of $\hat{\mu}$ for rejected samples are large and far from μ. On the contrary, the large effect size (greater than 4) implies there the sample is mostly rejected under such t-test, and thus the average of $\hat{\mu}$ are actually close to μ.