Handout 6

Student von Student III

In this handout, we will learn how to calculate a single loop. In the next week, we will discuss nested loops.

Topics and Concepts Covered

- Building a loop
- Bootstrapping
- Using a loop to adjust rows or columns of a matrix
- Creating a double loop
- *Indicator variable*: A variable that takes on a value of 1 if a particular condition is met, and 0 otherwise
- *Standard error of sample mean*: If $\overline{Y}$ is the sample mean of some set of observations, let $\widehat{\sigma}_Y$ denote the sample standard deviation of Y.

The standard error of the sample mean, $\overline{Y}$ is

$$\frac{\widehat{\sigma}_Y}{\sqrt{n}}$$

and the 95% confidence interval, using the Normal Approximation, is

$$\left[\overline{Y} - 1.96 \times \frac{\widehat{\sigma}_Y}{\sqrt{n}}, \overline{Y} + 1.96 \times \frac{\widehat{\sigma}_Y}{\sqrt{n}}\right]$$

R Commands Covered

- Running a loop with the command `for`
- Fitting a chance model for the mean using `lm(y ~ 1)`
- Calculating a difference-in-means using `lm(y ~ x)`
- Using `summary` to return the output from `lm`
- Using `rowMeans` and `colMeans` to return the means of the rows or columns of a matrix
- Testing conditions using `if`
- Creating a pdf with `pdf`
- Running all the code in a file with `source`
- Loading all the objects in an R data file (`.Rdata`) with `load`

Before beginning this handout, Do not forget to make a new folder for this assignment and set your working directory!

Using a Loop to Create a Bootstrapped Confidence Interval

We will be using the command `lm` in this class with some frequency. `lm` stands for "linear model" and gives a way of estimating chance models. For example, let's assume we have some outcome, Y_i, for a simple random sample of observations, i from 1 to N. Next, assume we want to fit a model of the following form:

$$Y_i = \beta_0 + \epsilon_i$$

In this model, we are assuming that our observed value Y_i is equal to some expected value (β_0) plus some chance error (ϵ_i). Though you may have realized that β_0 is simply the expected value of Y_i, and so its best estimate is the mean, $\overline{Y}$, we are going to spend some time incorporating this into the framework implemented by the command `lm`.

Fitting a Chance Model to World-Wide GDP

In the data folder you will find the file `GDPNorAm.csv`.
This file contains 184 observations, which consist of the country name, 2008 GDP (according to the World Bank), and an *indicator variable* that takes on a value of 1 if the country is in North America.

```
Data.gdp <- read.csv("data/GDP_NorAm_Partial.csv", header = TRUE)
head(Data.gdp)
```

```
              country          gdp north.america
1         Afghanistan  10617347740             0
2             Albania  12968653525             0
3             Algeria 171000000000             0
4              Angola  79620700694             0
5 Antigua and Barbuda   1354616665             1
6           Argentina 327000000000             0
```

```
summary(Data.gdp)
```

```
                country         gdp            north.america
 Afghanistan        :  1   Min.   :3.019e+07   Min.   :0.0000
 Albania            :  1   1st Qu.:5.425e+09   1st Qu.:0.0000
 Algeria            :  1   Median :2.249e+10   Median :0.0000
 Angola             :  1   Mean   :3.268e+11   Mean   :0.1087
 Antigua and Barbuda:  1   3rd Qu.:1.680e+11   3rd Qu.:0.0000
 Argentina          :  1   Max.   :1.420e+13   Max.   :1.0000
 (Other)            :178
```

Next, note that GDP is in dollars. Let's make a new variable that turns it from dollars to billions of dollars. Then, let's add the logged value of GDP:

```
Data.gdp$gdp2 <- Data.gdp$gdp / 1e9 # 1e9 returns a 1 with 9 zeroes, i.e. one billion
Data.gdp$log.gdp2 <- log(Data.gdp$gdp2)
```

Now that we have the data, let's fit a linear model (what FPP calls a "chance model"") to log GDP. The linear model will look like

$$log.gdp2 = \beta_0 + \epsilon_i$$

which we can implement in R using the following code:

```
lm1 <- lm(Data.gdp$log.gdp2 ~ 1)
summary(lm1)
```

```

Call:
lm(formula = Data.gdp$log.gdp2 ~ 1)

Residuals:
    Min      1Q  Median      3Q     Max
-6.7388 -1.5473 -0.1257  1.8856  6.3226

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.2384     0.1804   17.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.447 on 183 degrees of freedom
```

Using this summary function, you can focus on the part that says `Coefficients` and (for now) ignoring the rest. The output is telling us that our estimate for β_0, the mean of Y_i, is 3.2384. We can check this with

```
lm1$coef
```

```
(Intercept)
   3.238359
```

```
mean(Data.gdp$log.gdp2)
```

```
[1] 3.238359
```

and we get the same answer. The next column gives the standard error. Recall that the standard error of the sample mean is $\widehat{\sigma}_Y/\sqrt{n}$, where $\widehat{\sigma}_Y$ is the sample standard deviation of Y. We can calculate the standard error using

```
n <- length(Data.gdp$log.gdp2)
se <- sd(Data.gdp$log.gdp2) / sqrt(n)
se
```

```
[1] 0.1804031
```

Creating a Confidence Interval Using the Normal Approximation

When we use the Normal Approximation to create a confidence interval, we combine two different observations:

1. If a variable follows a normal distribution, we expect it to show up within 1.96 standard errors of its mean approximately 95% of the time.
2. Due to the **Central Limit Theorem**, we know that the sample mean, $\overline{Y}$, is approximately normal, with mean $\mathbf{E}(Y)$ and variance σ_Y^2/n, which we sometimes write as

$$\overline{Y} \sim \mathcal{N}\left(E(Y), \frac{\sigma_Y^2}{n}\right)$$

In the notation above, we are saying that $\overline{Y}$ is distributed ($\sim$) as a Normal random variable, with mean $\mathbf{E}(Y)$ and variance σ_Y^2/n.

When we use the Normal Approximation, we create a 95% confidence interval by taking the interval that ranges from 1.96 standard errors below the sample mean to 1.96 standard errors above the standard mean:

```
c(mean(Data.gdp$log.gdp2) - 1.96 * se, mean(Data.gdp$log.gdp2) + 1.96 * se)
```

```
[1] 2.884769 3.591949
```

We could create a 90% confidence interval

```
c(mean(Data.gdp$log.gdp2) - 1.64 * se, mean(Data.gdp$log.gdp2) + 1.64 * se)
```

```
[1] 2.942498 3.534220
```

Creating a Confidence Interval Using the Bootstrap

Next, we are going to calculate a confidence interval using the bootstrap. In order to implement the bootstrap, we need to use the command `sample` to produce a sample. The syntax for the command is where the *vector* is what we are sampling from; the `size` option tells how elements observations to sample from *vector*; and `replace` tells whether we can resample the same element multiple times. The `sample` function is a means of simulating the box-models discussed in the book. To see precisely what the function is doing, look at the following loop:

```
for (i in 1:10) {
  boot.samp <- sample(1:5, size = 5, replace = TRUE)
  print(boot.samp)
}
```

```
[1] 3 5 5 1 4
[1] 2 1 2 1 2
[1] 5 4 4 1 4
[1] 2 2 2 2 5
[1] 4 1 1 1 2
[1] 2 1 1 5 5
[1] 5 4 2 5 4
[1] 3 3 2 5 5
[1] 4 4 2 2 3
[1] 2 3 1 5 4
```

Each of the rows gives the indices for a `bootstrapped sample`. We have more than 5 observations in our data, so we are going to run our bootstrap so that it samples from `1:n`. The code for calculating the bootstrapped sample mean is

```
boot.mean <- NULL
for (i.boot in 1:5000) {
  boot.samp <- sample(1:n, size = n, replace = TRUE)
  Data.boot <- Data.gdp[boot.samp, ]
  boot.mean[i.boot]  <- lm(Data.boot$log.gdp2 ~ 1)$coef
}
```

Now, we are going to use the function `quantile` to recover our 95% confidence interval. The syntax for the command is

```
quantile(vector, probs)
```

where *vector* is the vector of bootstrapped means. For the 95% confidence interval, we want to take the quantiles at the probabilities `c(0.025, 0.975)`, since 0.975 - 0.025 = 0.95 and the two numbers add up to 1. We get

```
quantile(boot.mean, probs = c( 0.025, 0.975))
```

```
    2.5%    97.5%
2.873493 3.594462
```

We could also calculate a 90% bootstrapped confidence interval, using

```
quantile(boot.mean, probs = c(0.05, 0.95))
```

```
      5%      95%
2.931557 3.541149
```

The bootstrapped confidence intervals are quite close to that from using the Normal Approximation. This approximation is quite accurate *most* of the time; in the problems, you will explore a situation where the Normal Approximation is not entirely accurate.

Adjusting Rows or Columns of a Matrix with a Loop

For this problem, we will access the data from an online repository of US Congressional rollcall votes. In order to download the data, you will need to use the following commands:

```
library(foreign)
all.data <- read.dta("ftp://voteview.com/wf1/hou112kh.dta")
```

which downloads the roll call data for the 112th Session of the House of Representatives (2011-2012). Changing the 112 to a 111 will retrieve the data on the 111th session of the House (2010-2011). Loading a library, in this case `foreign`, gives you access to a new set of functions. We need the `read.dta` to read in STATA files (which end with `.dta`).

The variables in this dataset are:

- `cong`: the number of that Congress
- `id`: a unique identifier for each Congressmember
- `state`: the legislator's state (numeric)
- `ldist`: the legislator's district
- `lstate`: the legislator's state (as a word)
- `party`: the legislator's party (100: Democrat and 200: Republican)
- `eh1`: whether they were the first occupant of that seat, or replaced someone that term (due to death/retirement)
- `eh2`: how they attained office (1: general election; 2: special election; 3: elected by state legislature; 5: appointed)
- `V1`, `V2`, etc.: outcome for each vote, with each column a unique vote (0: not a member, 1: Yea, 2: Paired Yea, 3: Announced Yea, 4: Announced Nay, 5: Paired Nay, 6: Nay, 7: Present (some Congresses, also not used some Congresses), 8: Present (some Congresses, also not used some Congresses), 9: Not Voting)

First, we are going to clean up this data. To do so, we are going to create a new data frame that contains only the votes. This is the `all.data` without its first 9 columns, and we can do this one of two different ways:

```
votes <- all.data[, -c(1:9)]
votes <- all.data[, 10:ncol(all.data)]
```

To reduce the time it takes to run the loops, we are going to look at only the first 80 votes that session:

```
votes <- votes[, 1:80]
```

Note that, since we are selecting columns, we want to enter the subset condition into the *second* argument of the data frame.

First, we are going to recode this data so that 0 denotes a no vote, and 1 denotes a yes vote. This requires

- Changing all of the 0's and 9's to 999's (where we use 999 to denote missing)
- Changing all of the 6's to 0's

We do that through the following loop:

```
votes2 <- votes # Our new matrix
n.leg <- nrow(votes2) # Maximum for counter

# Loop through rows
for (i.vote in 1:n.leg) {
  # Find columns to change to 999
  change.999 <- votes2[i.vote, ] %in% c(0, 7, 9)
  votes2[i.vote,  change.999] <- 999
  # Find columns to change to 1
  change.1 <- votes2[i.vote, ] == 6
  votes2[i.vote, change.1] <- 0
}
```

There are several ways to do this loop, and some require a fewer lines than the code above. As you get more used to loops, you will find your coding growing more efficient naturally.

Now, we have a dataset littered with 999's. We want to impute a missing value here. We are going to through each vote and replace each unobserved value with the mean of members of that party that voted on the bill. Since votes are in columns, we are going to loop through columns (the second argument in the matrix) rather than rows (the first):

```
votes3 <- votes2 # Our new matrix
n.votes <- ncol(votes2)
for (i.vote in 1:n.votes) {
  # Find missing Democrats
  which.missing.dem <- (votes3[, i.vote] == 999) & (all.data$party == 100)
  which.present.dem <- (votes3[, i.vote] != 999) & (all.data$party == 100)
  # Find imputed value
  impute.dem <- mean(votes3[ which.present.dem, i.vote])
  # Replace missing with imputed
  votes3[which.missing.dem, i.vote] <- impute.dem
  # Repeat for Republicans
  which.missing.rep <- (votes3[, i.vote] == 999) & (all.data$party == 200)
  which.present.rep <- (votes3[, i.vote] != 999) & (all.data$party == 200)
  # Find imputed value
  impute.rep <- mean(votes3[which.present.rep, i.vote])
  # Replace missing with imputed
  votes3[which.missing.rep, i.vote] <- impute.rep
}
```

Finally, we want to reorient the 1's and 0's so that a 1 denotes "voted with the majority of the Democrats", and a 0 means "voted with the majority of the Republicans". That way, we can give some partisan valence to our measure. We are again going to loop through column by column:

```
votes4 <- votes3 # Our new matrix
n.votes <- ncol(votes3)
for (i.vote in 1:n.votes) {
  # Mean Republican Yay vote
  mean.dem <- mean(votes3[all.data$party == 100, i.vote])
  mean.rep <- mean( votes3[ all.data$party == 200, i.vote])
  if (mean.dem < mean.rep) {
    votes4[, i.vote] <- 1 - votes3[, i.vote]
  }
}
```

Note that we are using the `if` command above. The command uses the following syntax:

```
if (test) { function(x) }
```

The `if` command can be useful in programming, and is a more flexible version of `ifelse`. We are now ready to produce a measure of each legislator's ideology, which political scientists refer to as 'ideal points.' A simple estimate of ideology is the mean for each legislator, across rows. We can produce this using the command `rowMeans`, as below:

```
ideal.points <- rowMeans(votes4)
plot(density(ideal.points[all.data$party == 200], cut = 0),
     col = "red", xlim = c(0,1),
     main = "Estimated Ideal Points",
     xlab = "Ideal Points")
lines(density(ideal.points[all.data$party == 100], cut = 0),
      col = "blue")
```

The `rowMeans` function takes a matrix and returns the mean of each row; the `colMeans` function does the same, but on columns. Note that in this Congress, we get complete separation between the Democrats and Republicans.

Nested Loops

In the last 15 years, political scientists have produced a vast literature on get out the vote (GOTV) effects. The question is simple: what can be done to increase turnout among the US population? The first modern study in this field was conducted by Alan Gerber and Don Green, of Yale, during a 1998 election (Gerber and Green 2000). The authors assigned residents of New Haven to different groups, whereby the residents received either a personal visit, phone calls, or mailings. Several individuals were also left untreated, to act as a control group.

The study was particularly influential–so much so that one of the original authors headed up the GOTV effort for Gov. Rick Perry of Texas, and one of his students/co-authors headed up the GOTV efforts for Pres. Obama out of Chicago. The work has gotten plenty of attention from the popular press, as well. According to an interview of a campaign operative on the Washington Post:

> Two political scientists at Yale, Donald Green and Alan Gerber, went out and did a field experiment, which was a big deal at the time because political science lagged behind other social sciences in using field experiments to measure cause and effect in elections [...] The campaigns went out and did a bunch more of these comparative effectiveness studies, as opposed to mass media, where it's really hard to isolate voters and implement controls. When you're measuring turnout and registration rates, it's very easy to select some people to get your mail. In that case, the dependent variable is whether they voted or registered, which is an easy thing to track. In the last few years it's moved a lot to the behavioral psychology-informed bent, trying to demonstrate things that have been demonstrated in lab experiments about how to change motivations around voting [...] We have a whole sort of body of research about the contact and the quality of contact that we didn't 15 years ago.

(link) from 11/05/2012.

We are going to analyze the data from Gerber and Green's first study. In the data folder you will find the file `apsr_corrected_latest.txt`, which has the relevant data. The variables we will be using are:

Name	Description
`voted98`	1 if they voted in the 1998 election, 0 otherwise (dependent variable)
`vote96.0`	1 if they abstained from voting in 1996, 0 otherwise
`persons`	Number of people in the household
`mailgrp`	1 if they were sent mail encouraging them to vote; 0 otherwise
`persngrp`	1 if they were assigned to receive a personal visit encouraging them to vote; 0 otherwise
`phonegrp`	1 if they were called and encouraged them to vote; 0 otherwise
`ward`	Voting ward of household (2, 3, ..., 30)
`appeal`	Type of appeal (1, 2, 3)
`mail`	Number of mailings sent (0,1,2,3)
`age`	age of respondent

We are going to come back to this data a few times. First, let's read in the data and look at the baseline voting rate, for those who received no treatment:

```
GG <- read.table("data/apsr_corrected_latest.txt")
head(GG)
```

```
  persons ward question mailgrp phonegrp persngrp appeal contact mailings
1       2   24        0       1        0        0      1       0        3
2       2    8        0       1        1        0      1       0        1
3       2   25        0       1        1        0      1       0        3
4       2   18        0       1        0        0      3       0        3
5       2    6        0       1        0        0      2       0        3
6       2   25        0       1        0        0      2       0        2
  age majorpty vote96.0 vote96.1 mailcall voted98 phnscrpt dis.mc dis.phn
1  51        1        0        1        1       1        2     99      99
2  32        0        0        0        1       0        3     99      99
3  49        1        0        1        1       1        3      1       0
4  31        1        1        0        1       0        6      0      99
5  26        1        0        0        1       0        4      0      99
6  48        1        0        1        1       1        4      1      99
  phn.c phntrt1 phntrt2 phn.c1 phn.c2
1     0       1       1      0      0
2     0       0       1      0      0
3     1       0       1      0      1
4     0       1       1      0      0
5     0       1       1      0      0
6     1       1       1      1      1
```

```
summary(GG)
```

```
    persons           ward          question         mailgrp      
 Min.   :1.000   Min.   : 2.00   Min.   :0.0000   Min.   :0.000  
 1st Qu.:1.000   1st Qu.:10.00   1st Qu.:0.0000   1st Qu.:0.000  
 Median :1.000   Median :17.00   Median :1.0000   Median :1.000  
 Mean   :1.497   Mean   :16.43   Mean   :0.6656   Mean   :0.501  
 3rd Qu.:2.000   3rd Qu.:24.00   3rd Qu.:1.0000   3rd Qu.:1.000  
 Max.   :2.000   Max.   :30.00   Max.   :1.0000   Max.   :1.000  
    phonegrp         persngrp          appeal     contact      
 Min.   :0.0000   Min.   :0.0000   Min.   :1   Min.   :0.00000  
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:1   1st Qu.:0.00000  
 Median :0.0000   Median :0.0000   Median :2   Median :0.00000  
 Mean   :0.0998   Mean   :0.1972   Mean   :2   Mean   :0.05497  
 3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:3   3rd Qu.:0.00000  
 Max.   :1.0000   Max.   :1.0000   Max.   :3   Max.   :1.00000  
    mailings          age           majorpty         vote96.0      
 Min.   :0.000   Min.   :18.00   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:0.000   1st Qu.:34.00   1st Qu.:0.0000   1st Qu.:0.0000  
 Median :1.000   Median :46.00   Median :1.0000   Median :0.0000  
 Mean   :1.007   Mean   :49.33   Mean   :0.7426   Mean   :0.2707  
 3rd Qu.:2.000   3rd Qu.:64.00   3rd Qu.:1.0000   3rd Qu.:1.0000  
 Max.   :3.000   Max.   :97.00   Max.   :1.0000   Max.   :1.0000  
    vote96.1         mailcall         voted98          phnscrpt      
 Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
 1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000  
 Median :1.0000   Median :0.0000   Median :0.0000   Median :0.0000  
 Mean   :0.5274   Mean   :0.1639   Mean   :0.4525   Mean   :0.9176  
 3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:0.0000  
 Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :6.0000  
     dis.mc         dis.phn           phn.c            phntrt1      
 Min.   : 0.0   Min.   : 0.00   Min.   :0.00000   Min.   :0.0000  
 1st Qu.:99.0   1st Qu.:99.00   1st Qu.:0.00000   1st Qu.:0.0000  
```

```
 Median :99.0   Median :99.00   Median :0.00000   Median :0.0000
 Mean   :90.9   Mean   :94.34   Mean   :0.08492   Mean   :0.2146
 3rd Qu.:99.0   3rd Qu.:99.00   3rd Qu.:0.00000   3rd Qu.:0.0000
 Max.   :99.0   Max.   :99.00   Max.   :1.00000   Max.   :1.0000
    phntrt2           phn.c1            phn.c2
 Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
 1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.00000
 Median :0.0000   Median :0.00000   Median :0.00000
 Mean   :0.2196   Mean   :0.07437   Mean   :0.07594
 3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.00000
 Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
```

```
notreat.sub <- (GG$mailgrp == 0) &
               (GG$persngrp == 0) &
               (GG$phonegrp == 0)
control.mean <- mean(GG$voted98[notreat.sub == 1])
control.mean
```

```
[1] 0.4442449
```

Now, mailing (may?) have an effect that is close to zero: approximately 0.2 percentage points:

```
mail.mean <- mean(GG$voted98[GG$mailgrp == 1])
mail.mean
```

```
[1] 0.4547863
```

```
mail.mean - control.mean
```

```
[1] 0.01054139
```

A candidate, though, may want to know if a particular combination of appeals and mailings may be more effective. We are going to assess this in the next loop. This loop is a double loop, which is sometimes called a nested loop. The idea is that we loop through two different counters: one for the number of mailings (1, 2, 3) and another for the appeal (1, 2, 3).

The code follows:

```
effects.out <- NULL
for (i.num in 1:3) {
  for (i.appeal in 1:3) {
    mail.sub <- (GG$mailgrp == 1) &
                (GG$appeal == i.appeal) &
                (GG$mailings == i.num)
    outcome.sub <- mean(GG$voted98[mail.sub])
    output <- c(i.num, i.appeal, outcome.sub - control.mean)
    effects.out <- rbind(effects.out, output)
  }
}
colnames(effects.out) <- c("number", "appeal", "effect")
effects.out
```

```
       number appeal       effect
output      1      1 -0.006186196
output      1      2 -0.003330078
output      1      3  0.013497190
output      2      1  0.018247672
output      2      2  0.014459750
output      2      3  0.001725205
```

```
output      3      1  0.017340053
output      3      2  0.003607816
output      3      3  0.034203332
```

It appears that receiving 3 mailings of appeal 3 may be the most effective.
If you are wary of this result, though, you are probably correct. First, what we are doing is a type of "data-fishing," in that we cycle through all sub-groups until we find a strong result. Given enough time in a dataset, we can find *something*, though it may be noise. Second, we will use this as an example later to discuss statistical significance, the idea of whether we can differentiate a given effect from 0.

Precept Problems

Please note that *for this week only*, these problems will not be handed in.
Please hand in the code from the examples, and we will do the following questions in precept.

Question 1

Using the GDP data calculate the 95% Confidence Interval for `gdp2` using the normal approximation and the bootstrap.

Why do they differ so much? (Hint: Look at the density plots of `gdp2` versus `log.gdp2`

Question 2

Create a new chance model, `lm(Data.gdp$gdp2 ~ Data.gdp$north.america)`

What does each coefficient represent?

Question 3

In the roll call data create a `for` loop that does the analysis for roll call data, except for the last 8 Congresses. (This will be a nested loop.) The loop should produce a 4 x 2 figure of density plots, as in the example.

Question 4

Repeat the analysis, but only for 'close' votes, that is, votes for which the percentage of Republicans voting for the bill ranged between 0.4 and 0.6.

Question 5

In the Gerber and Green data what is the estimated effect of having been contacted by phone?

Question 6

Does the effect vary across whether the respondent voted in the past and the type of appeal they received?

References

Gerber, Alan S., and Donald P. Green. 2000. "The Effects of Canvassing, Telephone Calls, and Direct Mail on Voter Turnout: A Field Experiment." *American Political Science Review* 94 (03): 653–63. doi:10.2307/2585837.