PS1 Answers - R Version

Leah Broonks and Natalia Tosi

2025-01-22

1. Summary statistics

a. Panel dataset

```
# Importing datasets
db1950_raw <- read_dta("d1950_20230830.dta")
db2010_raw <- read_dta("d2010_20230830.dta")

# Appending datasets and including names for counties on 2010 database and
# adjusting binaries for years

full_db <- bind_rows(db1950_raw, db2010_raw) %>%
  group_by(countyfips, statefips) %>%
  arrange(countyfips, statefips) %>%
  mutate(name = ifelse(is.na(name) & countyfips == lag(countyfips),
                       lag(name), name)) %>%
  ungroup() %>%
  mutate(d1950 = ifelse(is.na(d1950), 0, d1950),
         d2010 = ifelse(is.na(d2010), 0, d2010))


# OR using data.table

setDT(db1950_raw)
setDT(db2010_raw)
dyears <- data.table::rbindlist(list(db1950_raw,db2010_raw), fill = TRUE)
setDT(dyears)
```

b. By year, find the average of

```
a. population (cv1)
b. log of population (create yourself from cv1)
c. share white (s1)
d. share black (s2)
e. share women age 25+ with education of some college or more (s3)
f. share men age 25+ with education of some college or more (s4)
```

```r
# Creating the log population variable
full_db <- full_db %>%
  mutate(log_pop = log(cv1))

# Generating means
table_1b <- full_db %>%
  group_by(year) %>%
  summarise_at(c('cv1', 'log_pop', 's1', 's2', 's3', 's4'), mean, na.rm = TRUE) %>%
  mutate(across(where(is.numeric), round, 3))

colnames(table_1b) <- c("Year",
                        "Population",
                        "Log Population",
                        "Share White Pop.",
                        "Share Black Pop.",
                        "Share Women 25+ Educated",
                        "Share Men 25+ Educated")

kable(table_1b)
```

Year	Population	Log Population	Share White Pop.	Share Black Pop.	Share Women 25+ Educated	Share Men 25+ Educated
1950	48580.71	9.894	0.891	0.101	0.119	0.100
2010	98641.04	10.278	0.843	0.090	0.515	0.466

```r
# OR using data.table

dyears[,log.cv1 := log(cv1)]

# by year, find average of cv1, log(cv1), s1, s2, s3, s4
dyears.sum <- dyears[,
                     lapply(.SD, function(x){mean(x, na.rm = TRUE)}),
                     .SDcols = c("cv1","log.cv1","s1","s2","s3","s4"),
                     by = "year"]

print(dyears.sum)
```

```
##    year      cv1   log.cv1        s1         s2        s3        s4
## 1: 1950 48580.71  9.894452 0.8910063 0.10098907 0.1189899 0.1004977
## 2: 2010 98641.04 10.278454 0.8429631 0.09046832 0.5153075 0.4664086
```

c. Find averages of the same variables by year and state for California, Mississippi and New Jersey.

```r
# Filtering for states
ca_mi_nj_db <- full_db %>%
  filter(statefips %in% c("06", "28", "34")) %>%
  mutate(state = case_when(statefips == "06" ~ "California",
                           statefips == "28" ~ "Mississippi",
                           statefips == "34" ~ "New Jersey",
```

```
                            TRUE ~ "NA"))

# Generating means
table_1c <- ca_mi_nj_db %>%
  group_by(year, state) %>%
  summarise_at(c('cv1', 'log_pop', 's1', 's2', 's3', 's4'), mean, na.rm = TRUE) %>%
  arrange(year, state) %>%
  mutate(across(where(is.numeric), round, 3))

colnames(table_1c) <- c("Year",
                        "State",
                        "Population",
                        "Log Population",
                        "Share White Pop.",
                        "Share Black Pop.",
                        "Share Women 25+ Educated",
                        "Share Men 25+ Educated")

kable(table_1c)
```

Year	State	Population	Log Population	Share White Pop.	Share Black Pop.	Share Women 25+ Educated	Share Men 25+ Educated
1950	California	182521.09	10.760	0.951	0.019	0.168	0.157
1950	Mississippi	26572.12	10.003	0.564	0.435	0.086	0.075
1950	New Jersey	230253.76	11.901	0.936	0.063	0.106	0.138
2010	California	642309.59	12.028	0.752	0.033	0.613	0.575
2010	Mississippi	36186.55	10.103	0.565	0.410	0.477	0.410
2010	New Jersey	418661.62	12.708	0.738	0.121	0.575	0.573

```
# OR using data.table

# ca is 06, MS is 28 and NJ is 34
dyears.st.sum <- dyears[,
                        lapply(.SD, function(x){mean(x, na.rm = TRUE)}),
                        .SDcols = c("cv1","log.cv1","s1","s2","s3","s4"),
                        by = c("statefips","year")]

print(dyears.st.sum[dyears.st.sum$statefips %in% c("06", "28", "34"),])
```

```
##    statefips year       cv1  log.cv1        s1         s2         s3         s4
## 1:        06 1950 182521.09 10.75970 0.9506171 0.01924481 0.16819786 0.15651813
## 2:        28 1950  26572.12 10.00343 0.5637197 0.43460928 0.08567864 0.07496693
## 3:        34 1950 230253.76 11.90111 0.9358877 0.06268884 0.10622251 0.13791879
## 4:        06 2010 642309.59 12.02814 0.7521886 0.03273037 0.61324170 0.57547000
## 5:        28 2010  36186.55 10.10345 0.5652806 0.41032685 0.47714137 0.41013774
## 6:        34 2010 418661.62 12.70820 0.7381942 0.12106672 0.57539905 0.57337725
```

2. Matching Data

a. How many counties are in both the 1950 and 2010 datasets?

```
wide_db <- db2010_raw %>%
  full_join(db1950_raw, by = c("statefips", "countyfips")) %>%
  mutate(d1950 = ifelse(is.na(d1950), 0, d1950),
         d2010 = ifelse(is.na(d2010), 0, d2010))

colnames(wide_db) <- gsub(".x$", "_2010", colnames(wide_db))
colnames(wide_db) <- gsub(".y$", "_1965", colnames(wide_db))
```

```
tab_2a <- wide_db %>%
  count(d1950, d2010)

print(paste("2a. There are", tab_2a[tab_2a$d1950 == 1 & tab_2a$d2010 == 1, 3],
            "counties in both datasets."))
```

```
## [1] "2a. There are 3090 counties in both datasets."
```

b. How many counties are in the 1950 dataset, but not the 2010 dataset?

```
print(paste("2b. There are", tab_2a[tab_2a$d1950 == 1 & tab_2a$d2010 == 0, 3],
            "counties in 1950, but not in 2010."))
```

```
## [1] "2b. There are 12 counties in 1950, but not in 2010."
```

c. How many counties are in the 2010 dataset, but not the 1950 dataset?

```
print(paste("2c. There are", tab_2a[tab_2a$d1950 == 0 & tab_2a$d2010 == 1, 3],
            "counties in 2010, but not in 1950"))
```

```
## [1] "2c. There are 19 counties in 2010, but not in 1950"
```

```
# OR

# make small datasets and merge to answer this question
d1950p <- db1950_raw[,c("statefips","countyfips","d1950")]
d2010p <- db2010_raw[,c("statefips","countyfips","d2010")]

# merge the two small datasets
mds <- merge(x = d1950p,
             y = d2010p,
             by = c("statefips","countyfips"),
             all = TRUE)
setDT(mds)

# re-code missings to zeros
mds[,
    `:=`
```

```
    (d1950 = ifelse(is.na(d1950) == TRUE, yes = 0, no = d1950),
     d2010 = ifelse(is.na(d2010) == TRUE, yes = 0, no = d2010))]

# find types
mds.sum <- mds[,
               .(counties = .N),
               by = c("d1950","d2010")]
print(mds.sum)
```

```
##    d1950 d2010 counties
## 1:     1     1     3090
## 2:     0     1       19
## 3:     1     0       12
```

d. Investigate two counties that are in the 2010 dataset, but not the 1950 dataset. Why is this?

Here are two examples – you answer can include any valid examples. My two examples did not exist in 1950.

- Menominee County, Wisconson (55/078) was created in 1959 (see Wikipedia)
- La Paz County, Arizona (04/012) was established in 1983 (again, see Wikipedia)

3. Regressions

a. Return to the panel dataset from question 1.

b. Regress log of population on the four share variables from question 1 and a fixed effect for year = 2010.

```
# Regressing log population on independent variables and the year FE
reg_3b <- lm(log_pop ~ s1 + s2 + s3 + s4 + d2010,
             data = full_db)

stargazer(reg_3b,
          type = "latex",
          font.size = "small")
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Wed, Jan 22, 2025 - 15:00:54

Table 3:

	Dependent variable:
	log_pop
s1	-1.172^{***}
	(0.220)
s2	0.303
	(0.235)
s3	-8.913^{***}
	(0.398)
s4	12.213^{***}
	(0.345)
d2010	-0.605^{***}
	(0.081)
Constant	10.741^{***}
	(0.223)
Observations	6,211
R^2	0.237
Adjusted R^2	0.237
Residual Std. Error	1.156 (df = 6205)
F Statistic	386.013^{***} (df = 5; 6205)
Note:	*p<0.1; **p<0.05; ***p<0.01

```
# OR

# make d2010 = 0 if not defined
dyears[,
       d2010 := ifelse(is.na(d2010) == TRUE, 0, d2010)]
```

```r
# ---- Question 3(b) ----------------------------------

r1 <- lm(log.cv1 ~ d2010 + s1 + s2 + s3 + s4,
         data = dyears)

print(summary(r1))
```

```
##
## Call:
## lm(formula = log.cv1 ~ d2010 + s1 + s2 + s3 + s4, data = dyears)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5580 -0.6563  0.0414  0.7055  4.7814
##
## Coefficients:
##             Estimate Std. Error t value             Pr(>|t|)
## (Intercept) 10.74145    0.22349  48.063 < 0.0000000000000002 ***
## d2010       -0.60543    0.08065  -7.507   0.0000000000000693 ***
## s1          -1.17212    0.22021  -5.323   0.0000001058045035 ***
## s2           0.30301    0.23501   1.289                0.197
## s3          -8.91323    0.39837 -22.374 < 0.0000000000000002 ***
## s4          12.21275    0.34508  35.392 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.156 on 6205 degrees of freedom
## Multiple R-squared:  0.2373, Adjusted R-squared:  0.2366
## F-statistic:   386 on 5 and 6205 DF,  p-value: < 0.00000000000000022
```

c. Interpret the coefficient on the year indicator variable

Check solutions for interpretation.

d. Repeat the previous regression with state fixed effects

```r
# Creating dummies for state FE
full_state_FE <- fastDummies::dummy_cols(full_db,
                                         select_columns = c("statefips")) %>%
  select(log_pop, 8:62)

# Regressing log population on independent variables and on the FE variables
reg_3d <- lm(log_pop ~ .,
             data = full_state_FE)

stargazer(reg_3d,
          type = "latex",
          digits = 3,
          no.space = TRUE,
          single.row = TRUE,
          header = FALSE,
          dep.var.labels = c("Log Population"),
          font.size = "small")
```

Table 4:

	Dependent variable:
	Log Population
s1	-2.062^{***} (0.204)
s2	-1.015^{***} (0.239)
s3	-3.829^{***} (0.382)
s4	9.005^{***} (0.310)
d2010	-1.504^{***} (0.081)
statefips_01	1.391^{***} (0.172)
statefips_04	1.261^{***} (0.232)
statefips_05	1.061^{***} (0.167)
statefips_06	1.660^{***} (0.169)
statefips_08	-0.101 (0.165)
statefips_09	2.681^{***} (0.280)
statefips_10	2.285^{***} (0.420)
statefips_11	2.620^{***} (0.700)
statefips_12	1.338^{***} (0.168)
statefips_13	0.762^{***} (0.159)
statefips_16	-0.066 (0.176)
statefips_17	1.153^{***} (0.158)
statefips_18	1.325^{***} (0.161)
statefips_19	0.829^{***} (0.158)
statefips_20	-0.002 (0.157)
statefips_21	1.204^{***} (0.158)
statefips_22	1.363^{***} (0.173)
statefips_23	1.772^{***} (0.222)
statefips_24	1.658^{***} (0.201)
statefips_25	2.432^{***} (0.231)
statefips_26	1.298^{***} (0.161)
statefips_27	0.895^{***} (0.159)
statefips_28	0.824^{***} (0.171)
statefips_29	1.019^{***} (0.158)
statefips_30	-0.440^{***} (0.168)
statefips_31	-0.268^{*} (0.158)
statefips_32	-0.383^{*} (0.219)
statefips_33	1.687^{***} (0.258)
statefips_34	2.510^{***} (0.208)
statefips_35	0.353^{*} (0.188)
statefips_36	2.175^{***} (0.167)
statefips_37	1.395^{***} (0.161)
statefips_38	-0.299^{*} (0.170)
statefips_39	1.906^{***} (0.162)
statefips_40	0.654^{***} (0.164)
statefips_41	0.748^{***} (0.182)
statefips_42	2.264^{***} (0.168)
statefips_44	1.818^{***} (0.337)
statefips_45	1.419^{***} (0.182)
statefips_46	-0.381^{**} (0.165)
statefips_47	1.402^{***} (0.162)
statefips_48	0.495^{***} (0.150)
statefips_49	-0.175 (0.191)
statefips_50	1.124^{***} (0.231)
statefips_51	0.717^{***} (0.157)
statefips_53	0.899^{***} (0.179)
statefips_54	1.427^{***} (0.172)
statefips_55	1.363^{***} (0.164)
statefips_56	
Constant	10.468^{***} (0.253)
Observations	6,211
R^2	0.472
Adjusted R^2	0.467
Residual Std. Error	0.966 (df = 6157)
F Statistic	103.840^{***} (df = 53; 6157)

```r
# OR

r2 <- lm(log.cv1 ~ d2010 + s1 + s2 + s3 + s4 + statefips,
         data = dyears)

print(summary(r2))
```

```
##
## Call:
## lm(formula = log.cv1 ~ d2010 + s1 + s2 + s3 + s4 + statefips,
##     data = dyears)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5104 -0.5499  0.0106  0.5744  4.6101
##
## Coefficients:
##              Estimate Std. Error t value             Pr(>|t|)
## (Intercept) 11.858702   0.227679  52.085 < 0.0000000000000002 ***
## d2010       -1.504120   0.080573 -18.668 < 0.0000000000000002 ***
## s1          -2.061972   0.204355 -10.090 < 0.0000000000000002 ***
## s2          -1.015039   0.238708  -4.252 0.000021479984525313 ***
## s3          -3.828948   0.381938 -10.025 < 0.0000000000000002 ***
## s4           9.004915   0.310026  29.046 < 0.0000000000000002 ***
## statefips04 -0.129305   0.204894  -0.631             0.528010
## statefips05 -0.329858   0.115967  -2.844             0.004464 **
## statefips06  0.269523   0.131036   2.057             0.039741 *
## statefips08 -1.491355   0.128313 -11.623 < 0.0000000000000002 ***
## statefips09  1.289960   0.258321   4.994 0.000000608891193276 ***
## statefips10  0.894751   0.403548   2.217             0.026645 *
## statefips11  1.229112   0.689295   1.783             0.074612 .
## statefips12 -0.052345   0.119087  -0.440             0.660276
## statefips13 -0.628350   0.099507  -6.315 0.000000000289708107 ***
## statefips16 -1.456276   0.138723 -10.498 < 0.0000000000000002 ***
## statefips17 -0.238008   0.112774  -2.110             0.034857 *
## statefips18 -0.065421   0.115033  -0.569             0.569569
## statefips19 -0.561787   0.115328  -4.871 0.000001136929339218 ***
## statefips20 -1.392142   0.114320 -12.178 < 0.0000000000000002 ***
## statefips21 -0.187081   0.109091  -1.715             0.086413 .
## statefips22 -0.027843   0.119455  -0.233             0.815708
## statefips23  0.381443   0.194297   1.963             0.049668 *
## statefips24  0.267840   0.163913   1.634             0.102303
## statefips25  1.041260   0.204966   5.080 0.000000388239022883 ***
## statefips26 -0.092880   0.117753  -0.789             0.430278
## statefips27 -0.495652   0.118824  -4.171 0.000030703214823540 ***
## statefips28 -0.567053   0.113362  -5.002 0.000000582650545958 ***
## statefips29 -0.371497   0.110219  -3.371             0.000755 ***
## statefips30 -1.830542   0.132411 -13.825 < 0.0000000000000002 ***
## statefips31 -1.658266   0.117212 -14.148 < 0.0000000000000002 ***
## statefips32 -1.773152   0.190120  -9.326 < 0.0000000000000002 ***
## statefips33  0.296327   0.235437   1.259             0.208214
## statefips34  1.119596   0.173898   6.438 0.000000000130000649 ***
## statefips35 -1.037524   0.153028  -6.780 0.000000000013140080 ***
```

```
## statefips36  0.784394   0.125622   6.244 0.000000000454753106 ***
## statefips37  0.004109   0.109094   0.038             0.969959
## statefips38 -1.689869   0.133139 -12.693 < 0.0000000000000002 ***
## statefips39  0.515035   0.115539   4.458 0.000008432207888024 ***
## statefips40 -0.736568   0.120347  -6.120 0.000000000990954317 ***
## statefips41 -0.642202   0.147520  -4.353 0.000013626544971046 ***
## statefips42  0.872980   0.122601   7.121 0.000000000001197673 ***
## statefips44  0.427336   0.319218   1.339             0.180719
## statefips45  0.028613   0.131452   0.218             0.827695
## statefips46 -1.772112   0.128294 -13.813 < 0.0000000000000002 ***
## statefips47  0.011266   0.112211   0.100             0.920028
## statefips48 -0.895341   0.097999  -9.136 < 0.0000000000000002 ***
## statefips49 -1.565636   0.158099  -9.903 < 0.0000000000000002 ***
## statefips50 -0.266214   0.205832  -1.293             0.195937
## statefips51 -0.673794   0.104024  -6.477 0.000000000100594129 ***
## statefips53 -0.491853   0.144726  -3.399             0.000682 ***
## statefips54  0.036111   0.128760   0.280             0.779140
## statefips55 -0.027310   0.122545  -0.223             0.823652
## statefips56 -1.390629   0.171505  -8.108 0.000000000000000614 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9659 on 6157 degrees of freedom
## Multiple R-squared:  0.472,  Adjusted R-squared:  0.4674
## F-statistic: 103.8 on 53 and 6157 DF,  p-value: < 0.00000000000000022
```

e. Interpret one of the share coefficients from the second regression

Check solutions for interpretation.

4. Long and Short Regressions and Omitted Variable Bias

```
# Filtering for the year of 2010, generating the share of employment variable and
# renaming the two independent variables

db_4 <- full_db %>%
  filter(year == 2010) %>%
  mutate(share_employment = cv59/cv1,
         variable_m = s4,
         variable_w = s3)
```

a. Estimate equations 1, 2 and 4 above.

a.1. Equation 1

$$E_i = \beta_0 + \beta_l M_i + \gamma W_i + \epsilon_{l,i}$$

```
# Replicating long equation

reg_4a_1 <- lm(share_employment ~ variable_m + variable_w, data = db_4)

coefficients(reg_4a_1)
```

```
## (Intercept)  variable_m  variable_w
##   0.2537173   0.1468912   0.2208502
```

$$E_i = 0.2537173 + 0.1468912 * M_i + 0.2208502 * W_i + \epsilon_{l,i}$$

a.2. Equation 2

$$E_i = \beta_0 + \beta_s M_i + \epsilon_{s,i}$$

```
# Replicating short equation
reg_4a_2 <- lm(share_employment ~ variable_m, data = db_4)

coefficients(reg_4a_2)
```

```
## (Intercept)  variable_m
##   0.2879093   0.3175865
```

$$E_i = 0.2879093 + 0.3175865 * M_i + \epsilon_{s,i}$$

a.3. Equation 4

$$W_i = \alpha + \pi M_i + \epsilon_{c,i}$$

```r
# Replicating 4th equation (correlation between variables)
reg_4a_4 <- lm(variable_w ~ variable_m, data = db_4)

coefficients(reg_4a_4)
```

```
## (Intercept)  variable_m
##   0.1548199   0.7729008
```

$$W_i = 0.1548199 + 0.7729008 * M_i + \epsilon_{c,i}$$

```r
# OR

# create employment to population ratio
db2010_raw[,
      E := cv59 / cv1]

# Eq 1:
e1 <- lm(E ~ s4 + s3,
         data = db2010_raw)
print(summary(e1))
```

```
##
## Call:
## lm(formula = E ~ s4 + s3, data = db2010_raw)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.32757 -0.03017  0.00299  0.03264  0.39698
##
## Coefficients:
##             Estimate Std. Error t value             Pr(>|t|)
## (Intercept) 0.253717   0.004892  51.858 < 0.0000000000000002 ***
## s4          0.146891   0.018034   8.145 0.00000000000000542 ***
## s3          0.220850   0.021195  10.420 < 0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04992 on 3106 degrees of freedom
## Multiple R-squared:  0.3774, Adjusted R-squared:  0.377
## F-statistic: 941.4 on 2 and 3106 DF,  p-value: < 0.00000000000000022
```

```r
# Eq 2:
e2 <- lm(E ~ s4,
         data = db2010_raw)
print(summary(e2))
```

```
##
## Call:
## lm(formula = E ~ s4, data = db2010_raw)
##
```

```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.34798 -0.03156  0.00087  0.03303  0.48428
##
## Coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.287909   0.003691   78.00 <0.0000000000000002 ***
## s4          0.317586   0.007669   41.41 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05078 on 3107 degrees of freedom
## Multiple R-squared:  0.3556, Adjusted R-squared:  0.3554
## F-statistic:  1715 on 1 and 3107 DF,  p-value: < 0.00000000000000022
```

```
# Eq 4:
e4 <- lm(s3 ~ s4,
         data = db2010_raw)
print(summary(e4))
```

```
##
## Call:
## lm(formula = s3 ~ s4, data = db2010_raw)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.31339 -0.02520 -0.00174  0.02213  0.39532
##
## Coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept) 0.154820   0.003072    50.4 <0.0000000000000002 ***
## s4          0.772901   0.006382   121.1 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.04226 on 3107 degrees of freedom
## Multiple R-squared:  0.8252, Adjusted R-squared:  0.8251
## F-statistic: 1.467e+04 on 1 and 3107 DF,  p-value: < 0.00000000000000022
```

```
# PS: Numbers might be slightly different from the ones printed above since there
# are more decimal points on this method.
```

b. Show that the omitted variable bias formula (equation 3) holds.

Equation 3:

$$\beta_s - \beta_l = \pi * \gamma$$

$\beta_s = 0.3175865$ $\beta_l = 0.1468912$

$\pi = 0.7729008$ $\gamma = 0.2208502$

```
0.3175865 - 0.1468912
```

```
## [1] 0.1706953
```

```
0.7729008 * 0.2208502
```

```
## [1] 0.1706953
```

As we can see from the calculations above, the difference between β_s and β_l is in fact equal to the product of π and γ as the omitted variable theory would indicate.

```
# OR

# copying coefficients by hand into here
# beta_s - beta_l = pi * gamma

beta_s <- 0.317586
beta_l <- 0.146891
pi <- 0.772901
gamma <- 0.220850

print("beta_s - beta_l")
```

```
## [1] "beta_s - beta_l"
```

```
beta_s - beta_l
```

```
## [1] 0.170695
```

```
print("pi * gamma")
```

```
## [1] "pi * gamma"
```

```
pi * gamma
```

```
## [1] 0.1706952
```