PS5 R Notes: Panel Data Models

Matthew Alampay Davis

November 22, 2021

As I mentioned in my note on Ed before going on strike, we will want to do all panel regression modeling with the familiar lm or lm_robust functions from the *estimatr* package.

I think some of you have looked at *plm*, which is a good package and can even handle robust standard errors. However, by default it clusters standard errors to the level of the individual fixed effects. In general, we do want to use clustered standard errors whenever we include fixed effects which is why they've made it automatic, but this problem set does ask us to implement fixed effects and clustering independent of one another so as far as I know, it cannot be used. Further, lm_robust is compatible with the linearHypothesis function from the *car* package when we run F-tests for a subset of covariates.

If I were designing this course myself or helping any of you with your own research, I would introduce you guys to the *fixest* package since it's just as easy to pick up on and is also the most advanced package in any software for running more demanding fixed effects models. But for now, let's stick with *estimatr*.

For the purposes of these notes, I'll be using the following panel dataset, available in my recitation folder:

```
guns <- read.dta13('handguns.dta') %>%
  mutate(lvio = log(vio),
         lrob = log(rob),
         lmur = log(mur))
head(guns)
```

```
##   year   stpop   state  area   vio  mur  rap   rob   aga    pro    bur    lar
## 1   77 3780403 Alabama 50708 414.4 14.2 25.2  96.8 278.3 3298.2 1135.5 1881.9
## 2   78 3831838 Alabama 50708 419.1 13.3 25.5  99.1 281.2 3519.7 1229.3 1987.9
## 3   79 3866248 Alabama 50708 413.3 13.2 27.5 109.5 263.1 3830.5 1287.3 2223.2
## 4   80 3900368 Alabama 50767 448.5 13.2 30.0 132.1 273.2 4485.1 1526.7 2642.2
## 5   81 3918531 Alabama 50767 470.5 11.9 26.1 126.5 306.1 4428.3 1450.7 2693.3
## 6   82 3925229 Alabama 50767 447.7 10.6 26.0 112.0 299.1 4185.8 1256.2 2656.4
##     aut yradopt incarc_rate   pb1064    pn1064   pw1064   pm1029      pop
## 1 280.7      77          83 8.384873 0.2828799 55.12291 18.17441 3.780403
## 2 302.5      77          94 8.352101 0.3077374 55.14367 17.99408 3.831838
## 3 320.1      77         144 8.329575 0.3327774 55.13586 17.83934 3.866248
## 4 316.2      77         141 8.408386 0.3644528 54.91259 17.73420 3.900368
## 5 284.2      77         149 8.483435 0.4112510 54.92513 17.67372 3.918531
## 6 273.3      77         183 8.514000 0.4499100 54.89621 17.51052 3.925229
##     avginc    density stateid shall     lvio     lrob     lmur
## 1 9.563148 0.07455240       1     0 6.026832 4.572647 2.653242
## 2 9.932000 0.07556673       1     0 6.038110 4.596129 2.587764
## 3 9.877028 0.07624532       1     0 6.024174 4.695925 2.580217
## 4 9.541428 0.07682881       1     0 6.105909 4.883559 2.580217
## 5 9.548351 0.07718658       1     0 6.153796 4.840242 2.476538
## 6 9.478919 0.07731851       1     0 6.104123 4.718499 2.360854
```

Here, the "entities" are US states plus DC and the time variable is the year ranging from 1977-1999. These variables are log transformations of rates of gun-related violence, robbery, and murder. Here's the given setup:

Some U.S. states have enacted laws that allow citizens to carry concealed weapons. These laws are known as "shall-issue" laws because they instruct local authorities to issue a concealed weapons permit to all applicants who are citizens, mentally competent, and have not been convicted of a felony (some states have some additional restrictions). Proponents argue that, if more people carry concealed weapons, crime will decline because criminals are deterred. Opponents argue that crime will increase because of accidental or spontaneous use of the weapon. In this exercise, you will analyze the effect of concealed weapons laws on three different categories of crimes: violent crimes; robberies (such as the robbery of a convenience store); and murder (many of which are spontaneous acts of passion).

Some data manipulation

We've already manipulated the data a little bit to get some variables log-transformed. We can do some more:

Sorting

If we want to sort our dataset according to a variable of interest, all we need is the dplyr package to run the following:

```
require(dplyr)
guns %>% arrange(vio) %>%
  head()
```

```
##   year  stpop        state  area  vio mur  rap rob  aga    pro   bur    lar
## 1   85 676991 North Dakota 69300 47.0 1.0  7.3 6.4 32.3 2632.4 427.0 2087.2
## 2   86 669489 North Dakota 69300 51.3 1.0 11.6 6.9 31.7 2554.2 385.1 2049.2
## 3   84 680498 North Dakota 69300 53.6 1.2 13.1 7.7 31.6 2529.7 399.1 2019.8
## 4   83 676685 North Dakota 69300 53.7 2.1 12.5 7.8 31.3 2621.8 436.3 2056.3
## 5   80 654380 North Dakota 69300 54.0 1.2  9.5 7.7 35.6 2909.7 488.3 2242.2
## 6   87 661167 North Dakota 69300 56.8 1.5  9.4 7.6 38.4 2776.2 455.4 2197.6
##     aut yradopt incarc_rate   pb1064   pn1064   pw1064   pm1029      pop
## 1 118.2      77          54 1.636063 2.698263 67.49587 17.54277 0.676991
## 2 119.9      77          55 1.683373 2.778238 67.07564 17.06212 0.669489
## 3 110.8      77          51 1.588102 2.622638 67.86971 17.96214 0.680498
## 4 129.1      77          47 1.536609 2.534710 68.22214 18.34073 0.676685
## 5 179.2      77          19 1.433876 2.381797 69.05927 19.23974 0.654380
## 6 123.2      77          53 1.741012 2.882025 66.72974 16.68701 0.661167
##      avginc     density stateid shall     lvio     lrob      lmur
## 1 11.862986 0.009768990      38     0 3.850148 1.856298 0.0000000
## 2 11.931534 0.009660736      38     1 3.937691 1.931521 0.0000000
## 3 11.802413 0.009819596      38     0 3.981549 2.041220 0.1823216
## 4 11.386000 0.009764574      38     0 3.983413 2.054124 0.7419373
## 5  9.786855 0.009442713      38     0 3.988984 2.041220 0.1823216
## 6 11.896504 0.009540649      38     1 4.039536 2.028148 0.4054651
```

The state with the fewest violent gun-related crimes is North Dakota in 1985.

```
guns %>% arrange(-vio) %>%
  head()
```

```
##   year  stpop                state area    vio  mur  rap    rob    aga    pro
## 1   93 576358 District of Columbia   61 2921.8 78.5 56.1 1229.6 1557.6 8839.3
## 2   92 584183 District of Columbia   61 2832.8 75.2 36.5 1266.4 1454.7 8574.2
## 3   94 564982 District of Columbia   61 2662.6 70.0 43.7 1107.2 1441.8 8422.6
## 4   95 551273 District of Columbia   61 2661.4 65.0 52.7 1239.0 1304.7 9512.1
## 5   96 538273 District of Columbia   61 2469.8 73.1 47.9 1186.7 1162.1 9426.9
## 6   90 603814 District of Columbia   61 2458.2 77.8 49.9 1213.5 1117.0 8316.0
##      bur    lar    aut yradopt incarc_rate   pb1064   pn1064   pw1064   pm1029
## 1 1995.5 5449.0 1394.8       0        1287 22.92985 2.347152 25.70434 14.05723
## 2 1820.2 5205.9 1548.0       0        1221 23.30794 2.207356 25.40950 14.55931
## 3 1760.9 5212.5 1449.3       0        1549 22.56992 2.426803 26.08649 13.63920
## 4 1838.4 5833.8 1839.9       0        1782 22.26011 2.461031 26.44189 13.29450
## 5 1809.9 5779.9 1837.0       0        1650 21.94686 2.524927 27.00581 12.88547
## 6 1983.0 4996.9 1336.1       0        1132 23.41714 1.845105 26.08502 15.29709
##        pop   avginc  density stateid shall     lvio     lrob     lmur
## 1 0.576358 21.81718 9.448492      11     0 7.979955 7.114444 4.363099
## 2 0.584183 21.30923 9.576771      11     0 7.949021 7.143934 4.320151
## 3 0.564982 22.00330 9.262000      11     0 7.887058 7.009590 4.248495
## 4 0.551273 21.61143 9.037263      11     0 7.886608 7.122060 4.174387
## 5 0.538273 21.84464 8.824147      11     0 7.811892 7.078932 4.291828
## 6 0.603814 20.28977 9.898590      11     0 7.807185 7.101264 4.354141
```

While Washington DC in 1993 has the most.

Grouping

Suppose we want to look at state averages in the dataset. We'll want to group our observations by state and then take averages. This will give us one observation per state and in so doing transforms it from a panel dataset to a regular cross-sectional dataset:

```
guns.average <- group_by(guns, state) %>%
  summarise(mean.violent = mean(vio),
            mean.murder = mean(mur),
            mean.robbery = mean(rob))
arrange(guns.average, -mean.violent) %>% head
```

```
## # A tibble: 6 x 4
##   state                mean.violent mean.murder mean.robbery
##   <chr>                       <dbl>       <dbl>        <dbl>
## 1 District of Columbia        2049.       49.3         1070.
## 2 Florida                      999.       10.3          314.
## 3 New York                     941.       10.7          502.
## 4 California                   877.       11.0          330.
## 5 Maryland                     854.       10.1          350.
## 6 Illinois                     828.        9.72         328.
```

Calling particular observations

Suppose we want to look at a particular state's observation

```
filter(guns.average, state == 'Maryland')
```

```
## # A tibble: 1 x 4
##   state    mean.violent mean.murder mean.robbery
##   <chr>           <dbl>       <dbl>        <dbl>
## 1 Maryland         854.        10.1         350.
```

Pooled OLS

Pooled OLS is just what we call OLS estimation on panel data where we do not make use of the panel structure at all and thereby treat all observations as independent regardless of what entity or time value it takes on. So this is just a matter of using our familiar lm/lm_robust command as usual.

Suppose we want to estimate a fixed effects model that regresses log-vio against the variables shall, incarc_rate, density, avginc, pop, pb1064, pm1029, and state and year fixed effects. The pooled OLS estimator for this model would essentially disregard the fixed effects.

```
model.pool <- lm_robust(lvio ~ shall + incarc_rate + density + avginc +
                          pop + pb1064 + pw1064 + pm1029, data = guns, se_type = 'stata')
summary(model.pool)
```

```
##
## Call:
## lm_robust(formula = lvio ~ shall + incarc_rate + density + avginc +
##     pop + pb1064 + pw1064 + pm1029, data = guns, se_type = "stata")
##
## Standard error type:  HC1
##
## Coefficients:
##              Estimate Std. Error  t value  Pr(>|t|)  CI Lower  CI Upper   DF
## (Intercept)  2.981738  0.6090198   4.8960 1.116e-06  1.786839  4.176638 1164
## shall       -0.368387  0.0347879 -10.5895 4.420e-25 -0.436641 -0.300133 1164
## incarc_rate  0.001613  0.0001807   8.9246 1.701e-18  0.001258  0.001967 1164
## density      0.026688  0.0143494   1.8599 6.315e-02 -0.001465  0.054842 1164
## avginc       0.001205  0.0072778   0.1656 8.685e-01 -0.013074  0.015484 1164
## pop          0.042710  0.0031466  13.5731 4.578e-39  0.036536  0.048884 1164
## pb1064       0.080853  0.0199924   4.0442 5.597e-05  0.041627  0.120078 1164
## pw1064       0.031201  0.0097271   3.2076 1.375e-03  0.012116  0.050285 1164
## pm1029       0.008871  0.0120604   0.7355 4.622e-01 -0.014792  0.032533 1164
##
## Multiple R-squared:  0.5643 ,    Adjusted R-squared:  0.5613
## F-statistic: 95.67 on 8 and 1164 DF,  p-value: < 2.2e-16
```

I believe you can estimate these using either lm or lm_robust (I'm not the grader), but the wording of the questions seems to suggest one over the other depending on the context.

One of the problem set questions asks you about the validity of standard errors on the OLS estimator given a model with a fixed effects term.

First differences

A good test of understanding: How is the first differences estimator related to the fixed effects estimator? Under what condition are they the same? I'll let you look back on the textbook or lecture slides to remember but I think it's important to keep in mind as you go through these problem set questions that keep asking you to conduct first difference estimation and then fixed effects estimation.

For now, here's just how to implement it in R. We have several states, each with multiple years of observations. We want to create new variables (the differenced variables) and we already know a function that does this: mutate from the *dplyr* package. We now want to use a new function called lag to pull an earlier observation so we can difference a current observation by a past observation for the same entity.

```
guns %<>% group_by(state) %>%
  mutate(diff.lvio = lvio-lag(lvio),
         diff.shall = shall-lag(shall),
         diff.incarc_rate = incarc_rate-lag(incarc_rate),
         diff.density = density-lag(density),
         diff.avginc = avginc-lag(avginc),
         diff.pop = pop-lag(pop),
         diff.pb1064 = pb1064-lag(pb1064),
         diff.pw1064 = pw1064-lag(pw1064),
         diff.pm1029 = pm1029-lag(pm1029))
```

If we have, for example, three years of observations for each state, then our differencing transformation will leave us with only two observations of the differenced variables per state: the difference between year 2 and year 1 and the difference between year 3 and year 2. We do not have year 0 observations so for year 1, the differenced observation will just be an NA:

```
guns %>% select(year, state, lvio, diff.lvio, shall, diff.shall) %>%
  filter(year %in% 77:80) %>%
  head(10)
```

```
## # A tibble: 10 x 6
## # Groups:   state [3]
##     year state    lvio diff.lvio shall diff.shall
##    <int> <chr>   <dbl>     <dbl> <int>      <int>
##  1    77 Alabama  6.03  NA           0         NA
##  2    78 Alabama  6.04   0.0113      0          0
##  3    79 Alabama  6.02  -0.0139      0          0
##  4    80 Alabama  6.11   0.0817      0          0
##  5    77 Alaska   6.09  NA           0         NA
##  6    78 Alaska   6.09  -0.00294     0          0
##  7    79 Alaska   6.20   0.106       0          0
##  8    80 Alaska   6.08  -0.119       0          0
##  9    77 Arizona  6.20  NA           0         NA
## 10    78 Arizona  6.31   0.111       0          0
```

Then it's just a matter of running the same regression but using the differenced variables rather than the original variables:

```
mod.diff <- lm_robust(diff.lvio ~ diff.shall + diff.incarc_rate + diff.density + diff.avginc +
                        diff.pop + diff.pb1064 + diff.pw1064 + diff.pm1029,
                      data = guns, se_type = 'stata')
summary(mod.diff)
```

```
##
## Call:
## lm_robust(formula = diff.lvio ~ diff.shall + diff.incarc_rate +
##     diff.density + diff.avginc + diff.pop + diff.pb1064 + diff.pw1064 +
##     diff.pm1029, data = guns, se_type = "stata")
##
## Standard error type:  HC1
##
## Coefficients:
##                     Estimate Std. Error  t value  Pr(>|t|)   CI Lower   CI Upper
## (Intercept)      -0.0200726  0.0061307 -3.27410 1.093e-03 -0.0321017 -8.044e-03
## diff.shall       -0.0188933  0.0175682 -1.07542 2.824e-01 -0.0533639  1.558e-02
## diff.incarc_rate -0.0003340  0.0001328 -2.51554 1.202e-02 -0.0005945 -7.348e-05
## diff.density     -0.2935127  0.1602898 -1.83114 6.735e-02 -0.6080170  2.099e-02
## diff.avginc      -0.0005528  0.0090415 -0.06114 9.513e-01 -0.0182930  1.719e-02
## diff.pop          0.0194924  0.0253764  0.76813 4.426e-01 -0.0302986  6.928e-02
## diff.pb1064       0.1684901  0.0317143  5.31275 1.304e-07  0.1062636  2.307e-01
## diff.pw1064       0.0476800  0.0059473  8.01710 2.721e-15  0.0360108  5.935e-02
## diff.pm1029      -0.1225661  0.0233208 -5.25565 1.767e-07 -0.1683238 -7.681e-02
##                    DF
## (Intercept)      1113
## diff.shall       1113
## diff.incarc_rate 1113
## diff.density     1113
## diff.avginc      1113
## diff.pop         1113
## diff.pb1064      1113
## diff.pw1064      1113
## diff.pm1029      1113
##
## Multiple R-squared:  0.07635 ,   Adjusted R-squared:  0.06971
## F-statistic: 10.57 on 8 and 1113 DF,  p-value: 2.235e-14
```

Notice that the number of observations in the differenced regression are lower than that in the pooled regresson:

```
mod.diff$nobs
```

```
## [1] 1122
```

```
model.pool$nobs
```

```
## [1] 1173
```

This is because of the NAs that arise from differencing variables; we're essentially removing one year of data for each entity/state.

A test of understanding: how does the differenced regression account for entity fixed effects μ_i?

Estimating by fixed effects

This is just a matter of running the same regression as the pooled OLS, but adding a fixed_effects argument to lm_robust

```r
# One-way fixed effects
model.1fe.robust <- lm_robust(lvio ~ shall + incarc_rate + density + avginc +
                                pop + pb1064 + pw1064 + pm1029, data = guns,
                              se_type = 'stata',
                              fixed_effects = state)
summary(model.1fe.robust)
```

```
##
## Call:
## lm_robust(formula = lvio ~ shall + incarc_rate + density + avginc +
##     pop + pb1064 + pw1064 + pm1029, data = guns, fixed_effects = state,
##     se_type = "stata")
##
## Standard error type:  HC1
##
## Coefficients:
##               Estimate Std. Error t value  Pr(>|t|)   CI Lower   CI Upper   DF
## shall       -4.614e-02  1.994e-02 -2.3136 2.087e-02 -0.0852721 -0.0070109 1114
## incarc_rate -7.101e-05  9.731e-05 -0.7297 4.657e-01 -0.0002619  0.0001199 1114
## density     -1.723e-01  1.049e-01 -1.6428 1.007e-01 -0.3780724  0.0334925 1114
## avginc      -9.204e-03  6.733e-03 -1.3669 1.719e-01 -0.0224155  0.0040080 1114
## pop          1.152e-02  9.704e-03  1.1876 2.353e-01 -0.0075162  0.0305655 1114
## pb1064       1.043e-01  1.656e-02  6.2990 4.305e-10  0.0717976  0.1367633 1114
## pw1064       4.086e-02  5.386e-03  7.5867 6.903e-14  0.0302935  0.0514287 1114
## pm1029      -5.027e-02  7.791e-03 -6.4528 1.634e-10 -0.0655588 -0.0349863 1114
##
## Multiple R-squared:  0.9411 ,    Adjusted R-squared:  0.938
## Multiple R-squared (proj. model):  0.2178 ,  Adjusted R-squared (proj. model):  0.1771
## F-statistic (proj. model):  28.1 on 8 and 1114 DF,  p-value: < 2.2e-16
```

Or including the time fixed effects as well:

```r
# Two-way fixed effects
model.2fe.robust <- lm_robust(lvio ~ shall + incarc_rate + density + avginc +
                                pop + pb1064 + pw1064 + pm1029, data = guns,
                              se_type = 'stata',
                              fixed_effects = ~ state + year)
summary(model.2fe.robust)
```

```
##
## Call:
## lm_robust(formula = lvio ~ shall + incarc_rate + density + avginc +
##     pop + pb1064 + pw1064 + pm1029, data = guns, fixed_effects = ~state +
##     year, se_type = "stata")
##
## Standard error type:  HC1
##
## Coefficients:
##               Estimate Std. Error t value  Pr(>|t|)   CI Lower  CI Upper   DF
## shall       -2.799e-02  1.937e-02 -1.4450 1.488e-01 -6.601e-02 0.0100196 1092
## incarc_rate  7.599e-05  8.292e-05  0.9165 3.596e-01 -8.671e-05 0.0002387 1092
## density     -9.155e-02  6.487e-02 -1.4114 1.584e-01 -2.188e-01 0.0357256 1092
```

```
## avginc       9.586e-04  7.197e-03  0.1332 8.941e-01 -1.316e-02 0.0150800 1092
## pop         -4.754e-03  6.703e-03 -0.7093 4.783e-01 -1.791e-02 0.0083976 1092
## pb1064       2.919e-02  2.104e-02  1.3874 1.656e-01 -1.209e-02 0.0704637 1092
## pw1064       9.250e-03  8.518e-03  1.0859 2.777e-01 -7.464e-03 0.0259639 1092
## pm1029       7.333e-02  1.878e-02  3.9052 9.992e-05  3.648e-02 0.1101670 1092
##
## Multiple R-squared:  0.9562 ,    Adjusted R-squared:  0.953
## Multiple R-squared (proj. model):  0.05635 , Adjusted R-squared (proj. model):  -0.01278
## F-statistic (proj. model): 7.823 on 8 and 1092 DF,  p-value: 2.935e-10
```

Clearly, we get different estimates depending on which estimation procedure we implement.

There's a question asking Stata users to run both areg and xtreg regressions. Running the appropriate version of the above should be equivalent to both so I think you only need to run one.

We can also implement these without robust standard errors as well using lm and manually adding the fixed effects ourselves.

```
model.1fe.nonrobust <- lm(lvio ~ shall + incarc_rate + density + avginc +
                            pop + pb1064 + pw1064 + pm1029 + factor(state), data = guns)
```

"factor" here just manually creates dummy variables for each state, which is exactly what an state fixed effects model does. Only downside is that the regression output will include coefficients/SEs for each state fixed effect (which can be many).

Cluster-robust standard errors

lm_robust includes a "clusters" argument:

```
mod.cluster <- lm_robust(lvio ~ shall + incarc_rate + density + avginc +
                           pop + pb1064 + pw1064 + pm1029,
                         clusters = state,
                         se_type = 'stata',
                         data = guns)
```

Note that unlike the plm package, you can indicate clustering and/or fixed effects at the same time or just one without the other.