Elementary Longitudinal Analysis

Joseph Nathan Cohen

20 November, 2019

Contents

Introduction

Longitudinal data refers to data that covers subjects over multiple time periods. This is in contrast to *cross-sectional* data, which measures subjects at a single point in time.

By allowing us to examine change over time, longitudinal data allow us to contemplate changes in our subjects and the environments in which they operate. We can examine what happens when our subjects' variable scores change, and which variable scores seem to be more stable over time. We can examine whether relationships hold across historical context, and which change more dramatically across time.

Basic Terminology

Some terminology:

- ***Units*** refer to the individual subjects that we are following across time
- ***Periods*** refer to the time periods in which the subjects were observed
- ***Cross-sections*** refer to comparisons across units within the same time period
- ***Time series*** refer to data series pertaining to the same unit over time

Take the following longitudinal data set, which looks at *economic growth rates* across a selection of countries.[1] You can load an RDS format of this data from the file "econgrowth.RDS".

Table 1: Economic Growth

country	2010	2011	2012	2013	2014	2015	2016
Brazil	6.52	3.03	1.01	2.11	-0.35	-4.35	-4.10
Canada	1.95	2.14	0.66	1.25	1.84	-0.06	-0.03
Germany	4.24	5.60	0.30	0.22	1.75	0.86	1.42
India	7.04	3.89	4.17	5.13	6.19	6.80	7.00
Korea, Rep.	5.97	2.89	1.76	2.43	2.69	2.25	2.47
Russian Federation	4.45	4.22	3.53	1.58	-1.08	-2.52	0.15
United States	1.72	0.82	1.50	1.14	1.70	2.13	0.84
South Africa	1.55	1.72	0.61	0.85	0.25	-0.34	-1.06

In this example, our units are countries. These countries' economic growth rates are covered over annual periods from 2010 to 2016.

Structure of Longitudinal Data

If your longitudinal data are not rectangularized (i.e., table-like), then your first task is to rectangularize your data. Please see the most current version of our notes on data management. When rectangularized, longitudinal data typically takes one of two shapes: long and wide.

Wide Data

Wide formats use one row per subject, and cast each time period as a column. Each table typically depicts the behavior of a variable. The data set is structured as a collection of tables.

[1] "Economic growth" is the percentage growth in the monetary value of everything produced in a country. When this value is negative, an economy is said to be in a "recession". A rate of 1% to 2% per year is considered very slow growth. Over 4% is considered fast growth for a country like the United States. A country experiencing rapid economic development will experience growth rates over 6%.

Table 1 (above) is an example of a wide table. Table 2 offers another wide table on the same subjects. This one depicts international businesspeople and diplomat's percetion of *control of corruption* index score from the World Governance Indicators[2]. The variable is scaled to a mean zero and standard deviation of one.[3] You can load this data from the file "corruption.RDS".

Table 2: Control of Corruption Score

country	2010	2011	2012	2013	2014	2015	2016
Brazil	0.05	0.17	-0.04	-0.08	-0.34	-0.40	-0.38
Canada	2.07	1.98	1.93	1.89	1.84	1.89	1.99
Germany	1.78	1.74	1.83	1.81	1.84	1.84	1.84
India	-0.47	-0.54	-0.51	-0.52	-0.43	-0.35	-0.28
Korea, Rep.	0.47	0.53	0.54	0.61	0.55	0.37	0.46
Russian Federation	-1.09	-1.07	-1.04	-1.01	-0.92	-0.95	-0.82
United States	1.27	1.27	1.41	1.31	1.38	1.40	1.37
South Africa	0.13	0.06	-0.12	-0.07	-0.06	0.03	0.12

In R, it is a perfectly viable approach to analyze data spread across a bunch of wide-format data tables. However, most of the tools conventionally used in analytics rely on long-format data.

Long Data

We have two data tables: "econgrowth" (from Table 1) and "corruption" (Table 2). We convert them to long format using the **gather()** command in the ***tidyr*** package:

```
#FORMAT
gather(data = DATA.OBJECT,    #Object storing data to be converted
       key = KEYS,            #Name of new time variables
       value = VALUES,        #Name of new variable with values stored in it
       COLUMNS,               #Names of old columns with values stored in them
       na.rm = FALSE,         #Remove observations with no values?
       factor_key = FALSE)    #Store keys as characters (F) or factor (T)
```

When applied to the "econgrowth" object:

```
#Truncated to fit on page:
econgrowth[1:6]
```

```
##              country     2010      2011      2012      2013       2014
## 1             Brazil 6.524373 3.0264046 1.0145352 2.1088884 -0.3524792
## 2             Canada 1.949628 2.1423090 0.6633031 1.2546550  1.8394598
## 3            Germany 4.239504 5.5994814 0.3035184 0.2157220  1.7529960
## 4              India 7.042349 3.8939002 4.1655281 5.1349569  6.1867320
## 5        Korea, Rep. 5.967519 2.8874633 1.7560438 2.4288668  2.6943431
## 6 Russian Federation 4.453103 4.2187412 3.5256750 1.5834745 -1.0809418
## 7      United States 1.716748 0.8162317 1.5021706 1.1384973  1.7026342
## 8       South Africa 1.551073 1.7207143 0.6079491 0.8526848  0.2472788
```

We can convert these two data sets into a single long table by using the **gather()** and **merge()** functions:

[2]To learn more of this data, visit: https://info.worldbank.org/governance/wgi/

[3]So a score of zero indicates "average" levels of corruption worldwide. A +2 suggests a extraordinarily non-corrupt society, and a -2 is an extraordinarily corrupt one.

Reshape Indidivual Data Tables

To reshape individual data tables to long format:

Identify the names of the variables that contain the data to be reshaped:

```
#Identify variable's names
names(econgrowth)
```

```
## [1] "country" "2010"    "2011"    "2012"    "2013"    "2014"    "2015"
## [8] "2016"
```

```
#Create vector with names of variables to be gathered:
COLUMNS <- names(econgrowth)[2:8]
```

Then, run the **gather()** operation:

```
#Run operation
library(tidyr)
econgrowth.l <- gather(econgrowth,
                       key = "year",
                       value = "econ.growth",
                       COLUMNS)


#Results
head(econgrowth.l, 10)
```

```
##               country year econ.growth
## 1              Brazil 2010    6.524373
## 2              Canada 2010    1.949628
## 3             Germany 2010    4.239504
## 4               India 2010    7.042349
## 5         Korea, Rep. 2010    5.967519
## 6  Russian Federation 2010    4.453103
## 7       United States 2010    1.716748
## 8        South Africa 2010    1.551073
## 9              Brazil 2011    3.026405
## 10             Canada 2011    2.142309
```

Repeat for Other Sets. Do the same on the "corruption" set:

```
COLUMNS <- names(corruption)[2:8]
corruption.l <- gather(corruption,
                       key = "year",
                       value = "corruption",
                       COLUMNS)
```

Merge the Tables

The next step is to take these individual data tables and merge them into one set. For this you must begin by identifying the variables that contain the unit and time period to which each observation pertains. In these sets, these are the "country" and "year" variables. We are lucky enough that the unit and time identifiers have the same name and are coded using the same scheme.[4]

```
SYNTAX:

merge(SET1, SET2,                  #Two data objects to merge
      by = c(UNIT, TIME),          #UNIT and TIME identifiers
      all.x = T,                   #Keep all SET1 without matches
      all.y = T,                   #Keep all SET2 w/o matches
      sort = T)                    #Sort results
```

In this case:

```
data <- merge(econgrowth.l, corruption.l,
              by = c("country", "year"),
              all.x = T, all.y = T,
              sort = T)

#Results
head(data, 10)
```

```
##    country year econ.growth corruption
## 1   Brazil 2010   6.5243728  0.0462800
## 2   Brazil 2011   3.0264046  0.1659021
## 3   Brazil 2012   1.0145352 -0.0372771
## 4   Brazil 2013   2.1088884 -0.0847100
## 5   Brazil 2014  -0.3524792 -0.3384546
## 6   Brazil 2015  -4.3514840 -0.3964888
## 7   Brazil 2016  -4.0987023 -0.3814404
## 8   Canada 2010   1.9496282  2.0697990
## 9   Canada 2011   2.1423090  1.9778830
## 10  Canada 2012   0.6633031  1.9271360
```

[4]What would we have done if they weren't in the same format? You have to develop a scheme to match identifiers between the two sets you are merging. For country-level data, I recommend the package ***countrycodes***.

Converting from Long to Wide

Two steps:

(1) Isoldate the variable that you wish to tranform:

```
names(data)
```

```
## [1] "country"     "year"        "econ.growth" "corruption"
```

```
corruption <- data[c(1,2,4)]
head(corruption)
```

```
##   country year corruption
## 1  Brazil 2010  0.0462800
## 2  Brazil 2011  0.1659021
## 3  Brazil 2012 -0.0372771
## 4  Brazil 2013 -0.0847100
## 5  Brazil 2014 -0.3384546
## 6  Brazil 2015 -0.3964888
```

(2) the use the **spread()** operation from ***tidyr***:

```
corruption.w <- spread(corruption, year, corruption)
head(corruption.w)[1:6]
```

```
##              country       2010       2011       2012      2013       2014
## 1             Brazil  0.0462800  0.1659021 -0.0372771 -0.084710 -0.3384546
## 2             Canada  2.0697990  1.9778830  1.9271360  1.886833  1.8364260
## 3            Germany  1.7762700  1.7432730  1.8295770  1.813811  1.8383800
## 4              India -0.4681174 -0.5362822 -0.5132375 -0.517440 -0.4280237
## 5        Korea, Rep.  0.4694922  0.5282930  0.5351672  0.614448  0.5475607
## 6 Russian Federation -1.0905310 -1.0650190 -1.0423350 -1.013369 -0.9192259
```

```
#Truncated for legibility
```

Useful Variable Formulations

Calculating Previous Year

There may be times when you want to create a *lagged value* in your set, which shows a variable's previous value in the same time series. This can be useful if you want to contemplate and model *autoregression* – a variable's correlation with its past value. See more below. Using **ddply()** from ***plyr***.

```
#SYNTAX
ddply(DATA,                  #Data set
      .(SERIES),             #Series identifier
      transform,             #Tells command to create new variable
      NEWVAR = EQUATION)     #New variable and how it is calculated
```

As implemented here, We are asking the operation to render the previous year's value within this country-level series:

```
library(plyr)
data <- ddply(data,
              .(country),
              transform,
              corr.prev = c(NA, corruption[-length(corruption)]))
head(data)
```

```
##   country year econ.growth corruption  corr.prev
## 1  Brazil 2010   6.5243728  0.0462800         NA
## 2  Brazil 2011   3.0264046  0.1659021  0.0462800
## 3  Brazil 2012   1.0145352 -0.0372771  0.1659021
## 4  Brazil 2013   2.1088884 -0.0847100 -0.0372771
## 5  Brazil 2014  -0.3524792 -0.3384546 -0.0847100
## 6  Brazil 2015  -4.3514840 -0.3964888 -0.3384546
```

Calculating Change Rates

You can calculate the percentage year-on-year change rate for a variable using **ddply()** from ***plyr***. For example, with our "corruption" variable:

```
library(plyr)
data$corruption.ch <- ((data$corruption - data$corr.prev) / data$corr.prev)
head(data)
```

```
##   country year econ.growth corruption  corr.prev corruption.ch
## 1  Brazil 2010   6.5243728  0.0462800         NA            NA
## 2  Brazil 2011   3.0264046  0.1659021  0.0462800     2.5847472
## 3  Brazil 2012   1.0145352 -0.0372771  0.1659021    -1.2246934
## 4  Brazil 2013   2.1088884 -0.0847100 -0.0372771     1.2724407
## 5  Brazil 2014  -0.3524792 -0.3384546 -0.0847100     2.9954504
## 6  Brazil 2015  -4.3514840 -0.3964888 -0.3384546     0.1714682
```

The results express the magnitude of change related to the previous reported value. So Brazil in 2011 had a control of corruption score as having added 258% its value in 2010.

```
0.04628 + (0.04628 * 2.5847472)
```

```
## [1] 0.1659021
```

Describing Longitudinal Data

These types of data have two dimensions: a cross-sectional and a time-series one. Recall that *cross-sectional* differences or effects are those that inhere between two countries at the same point in time. *Longitudinal* differences or effects inhere within units over time. Both are important places to look when trying to understand a phenomenon.

Cross-Sectional Differences

One way to tease out cross-sectional effects is to look at average scores across our units. For example, using our *data* object, we might look at average economic growth scores across countries:

```
xsec.mean <- aggregate(econ.growth ~ country, data, mean)
xsec.mean
```

```
##              country econ.growth
## 1             Brazil   0.5530765
## 2             Canada   1.1084632
## 3            Germany   2.0562105
## 4              India   5.7453594
## 5        Korea, Rep.   2.9213873
## 6 Russian Federation   1.4753519
## 7       South Africa   0.5110220
## 8      United States   1.4052189
```

Looks like the Indian and Korean economies were doing well. Germany has pretty good years for a rich country.

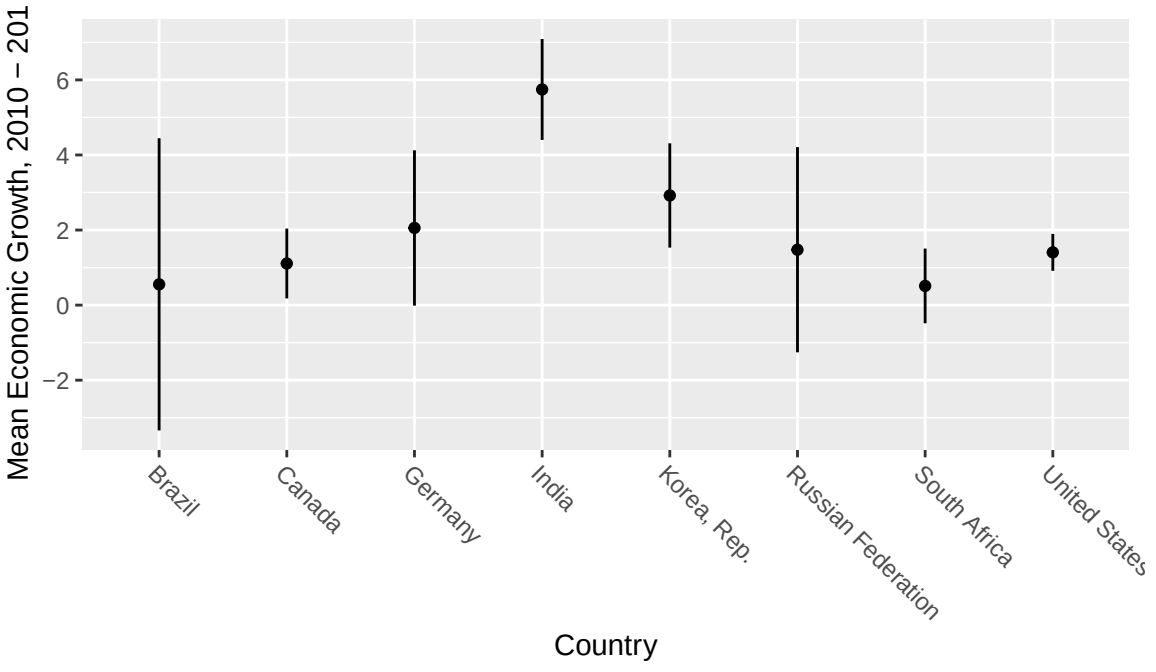
If there is a lot of data, you might consider adding variability measures to these estimates:

```
xsec.sd <- aggregate(econ.growth ~ country, data, sd)
xsec.growth <- merge(xsec.mean, xsec.sd, by="country")
names(xsec.growth) <- paste(c("country", "growth.mean", "growth.sd"))
xsec.growth
```

```
##              country growth.mean growth.sd
## 1             Brazil   0.5530765 3.8927500
## 2             Canada   1.1084632 0.9302224
## 3            Germany   2.0562105 2.0697575
## 4              India   5.7453594 1.3444956
## 5        Korea, Rep.   2.9213873 1.3899032
## 6 Russian Federation   1.4753519 2.7343465
## 7       South Africa   0.5110220 0.9948772
## 8      United States   1.4052189 0.4926489
```

A you might use the resulting object in a graph:

```
library(ggplot2)
ggplot(xsec.growth, aes(x = country)) +
  geom_point(aes(y = growth.mean)) +
  geom_linerange(aes(ymin = growth.mean - growth.sd,
                     ymax = growth.mean + growth.sd)) +
  theme(axis.text.x=element_text(angle = -45, hjust = 0)) +
  ylab("Mean Economic Growth, 2010 - 2016") +
  xlab("Country")
```



If you are working with larger cross-sections, you can look at averages across groups. In this case, for example, you might look at geographical groupings, or compare more versus less-industrialized countries, etc.

Longitudinal Differences

We can also look at changes across time periods:

```
aggregate(econ.growth ~ year, data, mean)
```

```
##   year econ.growth
## 1 2010   4.1805372
## 2 2011   3.0381557
## 3 2012   1.6923404
## 4 2013   1.8397182
## 5 2014   1.6237529
## 6 2015   0.5954637
## 7 2016   0.8341104
```

Looks like these economies were doing better in 2010 - 2011. Growth across these countries was not so good in 2015 and 2016. You can also add variability measures in these kinds of tables, and graph as above.

Modeling Longtudinal Data

Classical linear models assume that each observation is independent from the others. In a longtudinal data series, this not the case. Observations within the same unit are related, as are those taken within the same time period. Not only could this lead to errors in our model estimates, but it turns a blind eye to the relationships that we could model with this kind of data.

What we want to do in this situation is tease out and control for these cross-sectional and longtudinal effects. Some possibilities:

Serial Correlation

Serial correlation occurs when a variable's value in a previous period influences its value in the present period. For example, economic growth can have a feedback effect. When the economy is prospering, people become optimistic. They start spending and investing more aggressively. This causes business to heat up, and businesses might respond by hiring more people or investing in expansion. This can lead to even more growth. These economic cycles are often theorized to end when everyone realizes that they were over-optimistic, and start pulling back – which creates more and more economic pessimism. The point is that we might expect our variables to have this type of cross-temporal relationship.

Longitudinal data allows you to test for these kinds of feedback loops. Arguably, your model is omitting an important variable if serial correlation is present empirically but not included in the model. In a regression model, you might do something like this:

$$y_t = \alpha + \rho \cdot y_{t-1} + \beta \cdot x + \epsilon$$

Where:

- y_t is the outcome variable at time t
- y_{t-1} is the outcome variable at the previous time period $(t-1)$
- α is an intercept term
- ρ is an *autoregressive term*, the relationship between a the outcome's present and previous scores.
- x are a set of other predictors
- β are these predictor's coefficients
- ϵ is error

Let's apply it to this analysis of economic growth and control of corruption:

```r
#First Set
mod.1 <- lm(econ.growth ~ corruption, data = data)

#Calculate Autoregressive Term
data <- ddply(data,
              .(country),
              transform,
              econ.growth.ly = c(NA, econ.growth[-length(econ.growth)]))

#Add autoregressive term to model
mod.2 <- lm(econ.growth ~ econ.growth.ly + corruption, data = data)
```

```r
#Trimming sample to make Models 1 and 2 representative of identical sample:
mod.1a <- lm(econ.growth ~ corruption, data=mod.2$model)

library(stargazer)
stargazer(mod.1a, mod.2, header = F)
```

Table 3:

	Dependent variable:	
	econ.growth	
	(1)	(2)
econ.growth.ly		0.787***
		(0.095)
corruption	−0.155	0.097
	(0.336)	(0.216)
Constant	1.682***	−0.145
	(0.382)	(0.327)
Observations	48	48
R^2	0.005	0.607
Adjusted R^2	−0.017	0.589
Residual Std. Error	2.373 (df = 46)	1.508 (df = 45)
F Statistic	0.213 (df = 1; 46)	34.737*** (df = 2; 45)
Note:		*p<0.1; **p<0.05; ***p<0.01

In Model 1, it appears as there is zero relationship between corruption and economic growth. The model fails its F-test, which tests the proposition that the entire model demonstrates virtually zero predictive power.

However, in Model 2, the addition of an autoregressive term yields an impressive R-squared of 0.60. From year-to-year, economic growth generally exhibits a very strong autocorrelation. Economic growth tends to run in multiyear growth and slowdown streaks. Many national-level economic, political, and social wellbeing indicators have a strong serial correlation because they tend to change more slowly, on scales that span several economic growth boom-and-bust cycles.

Fixed Effects

Fixed effects are a modeling strategy that involves adding dummy variables for each of a data set's units, periods, or both. Conceptually, analysts treat these dummy variables as "absorbing" unexplained variation that inheres in individual units or periods. So, imagine we were working with the country-year data above. If we use unit-level fixed effects, we would include dummies for each country in the data set, and treat these predictors as capturing unmeasured, unit-specific, time-invariant qualities of individual countries.

The model might be:

$$y_{it} = \alpha + \gamma_i \cdot u_i + \beta \cdot x_{it} + \epsilon$$

Where u_i is a dummy variable for each of i units or time periods that we want to model (with one removed as baseline comparison group). In our case, countries. The γ_i term is a unique coefficient for each country's dummy variable.

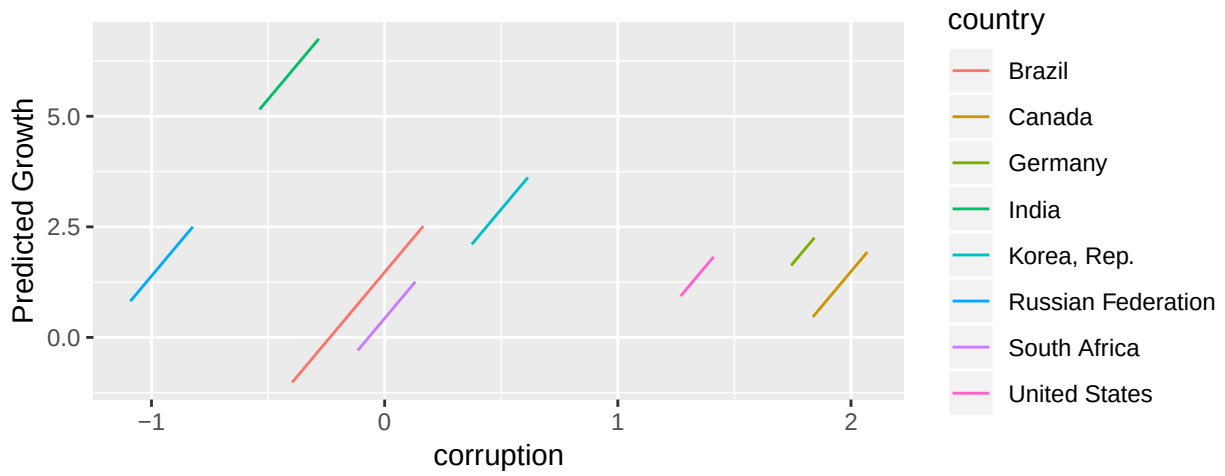
Using the **lm()** command, the model might look something like this:

```
mod.3 <- lm(econ.growth ~ corruption + factor(country),
            data = data)
summary(mod.3)
```

```
##
## Call:
## lm(formula = econ.growth ~ corruption + factor(country), data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2254 -0.9815 -0.0251  0.7753  4.7592
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         1.474      0.817   1.805   0.0775 .
## corruption                          6.285      2.542   2.472   0.0171 *
## factor(country)Canada             -12.553      5.402  -2.324   0.0245 *
## factor(country)Germany            -10.804      5.084  -2.125   0.0389 *
## factor(country)India                7.052      1.274   5.535 1.35e-06 ***
## factor(country)Korea, Rep.         -1.718      1.947  -0.883   0.3820
## factor(country)Russian Federation   6.198      2.369   2.616   0.0119 *
## factor(country)South Africa        -1.044      1.105  -0.944   0.3498
## factor(country)United States       -8.518      3.928  -2.169   0.0352 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.924 on 47 degrees of freedom
## Multiple R-squared:  0.4897, Adjusted R-squared:  0.4028
## F-statistic: 5.637 on 8 and 47 DF,  p-value: 5.025e-05
```

The effect is to give each country a separate intercept, or baseline variable score. You might depict the regression lines as below:

```
data$predicted.value <- mod.3$fitted.values
ggplot(data, aes(x = corruption, y = predicted.value, color = country)) + geom_line()  + ylab("Predicte
```



You can apply fixed effects to units or time. We treat these predictors as capturing unmeasured qualities that inhere directly in the unit or time period that they are measuring.

Note that, with fixed-effects, you cannot include other variables that do not change over time, because they would be collinear with the fixed effects. For example, I could not include both a country-level fixed effect and a geographic region dummy variable, because both variables would be unchanging across all our countries. The model cannot parse the effects of geography from the fixed-effect.

Random Effects

I understand random-effects to be used when we want to extract the effects of units, but through a slightly different conceptual framework that treats them as representative or a subset of a larger population of units. Whereas with fixed effects we can interpret individual effects as meaningful in and of themsleves, random effects models are seeking to extricate the effects of country-effects in general. Rather than estimating effects, we are trying to tease out a typical country-level effect, which may be weaker or stronger in the particular units we observe.

Practically speaking, this can be a more practical strategy if, for example, we were sampling thousands of students from hundreds of classrooms (a group that, while large, is nowhere near exhausting population of classrooms in the country).

Random effects are assumed to be uncorrelated with other predictors in the model. This assumption can be tested using a Hausman Test (see below). If the test fails, then unit-effects are correlated with predictors (the null is that they are not correlated). If significant, use fixed effects.

First Difference Models

First-difference models changes in outcome values using change scores for predictors.

It is:

$$\Delta y_{it} = \alpha + \beta \cdot \Delta x_{ij} + \epsilon$$

Where $\Delta x = x_t - x_{t-1}$, same for y.

The PLM Package

These types of models can be run more conveniently using the command **plm()** from the ***plm*** package. Here, we will work with continuous outcomes.

Syntax

The command's syntax:

```
plm(FORMULA,                  # Regression model, no effects
    data = DATA,              # Data object
    index = c(COUNTRY, YEAR)
    effect = EFFECT,          # Which effect:
                                # "individual" = Unit effects
                                # "time" = Time effects
                                # "twoways" = Both effects
    weights = WEIGHTS,        # Weight variable, if necessary
    model = MODEL)            # Type of effect:
                                # "pooling" = OLS
                                # "within" = Fixed Effects
                                # "random" = Random Effects
                                # "fd" = first differences model
```

So, for a model with country fixed-effects:

```
library(plm)
mod.A <- plm(econ.growth ~ corruption,
             data = data,
             index = c("country", "year"),
             effect = "individual",
             model = "within")
summary(mod.A)
```

```
## Oneway (individual) effect Within Model
##
## Call:
## plm(formula = econ.growth ~ corruption, data = data, effect = "individual",
##     model = "within", index = c("country", "year"))
##
## Balanced Panel: n = 8, T = 7, N = 56
##
## Residuals:
##      Min.   1st Qu.    Median   3rd Qu.      Max.
## -4.225388 -0.981520 -0.025104  0.775272  4.759154
##
## Coefficients:
##            Estimate Std. Error t-value Pr(>|t|)
## corruption   6.2845     2.5423   2.472  0.01711 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Total Sum of Squares:    196.51
## Residual Sum of Squares: 173.9
## R-Squared:      0.11505
## Adj. R-Squared: -0.035575
## F-statistic: 6.1106 on 1 and 47 DF, p-value: 0.017114
```

See the Fixed Effects

The results are similar to "mod.1" above, but the fixed effects are hidden and extricated from our R-Squared estimates. To see the fixed effect values:

```
fixef(mod.A)
```

```
##             Brazil             Canada            Germany
##          1.4743725        -11.0785607         -9.3300421
##              India        Korea, Rep. Russian Federation
##          8.5261351         -0.2437469          7.6721459
##       South Africa      United States
##          0.4306048         -7.0439148
```

Test Fixed Effects' Predictive Power

Testing that all fixed effects insignificant. If p-value is over 0.1, then OLS is a more parsimonious model that does not lose information:

```
#Run the OLS
mod.ols <- plm(econ.growth ~ corruption,
            data = data,
            index = c("country", "year"),
            model = "pooling")

#Test FE against OLS
pFtest(mod.A, mod.ols)
```

```
##
##  F test for individual effects
##
## data:  econ.growth ~ corruption
## F = 6.2739, df1 = 7, df2 = 47, p-value = 3.218e-05
## alternative hypothesis: significant effects
```

Here, the test is significant – fixed-effects are capturing information.

Random or Fixed Effects?

To implement a random-effects test.

```
mod.B <- plm(econ.growth ~ corruption,
             data = data,
             index = c("country", "year"),
             effect = "individual",
             model = "random")
summary(mod.B)
```

```
## Oneway (individual) effect Random Effect Model
##    (Swamy-Arora's transformation)
##
## Call:
## plm(formula = econ.growth ~ corruption, data = data, effect = "individual",
##     model = "random", index = c("country", "year"))
##
## Balanced Panel: n = 8, T = 7, N = 56
##
## Effects:
##                  var std.dev share
## idiosyncratic 3.700   1.924 0.574
## individual    2.748   1.658 0.426
## theta: 0.5984
##
## Residuals:
##     Min.  1st Qu.   Median  3rd Qu.     Max.
## -5.45171 -0.83081 -0.10890  0.72610  5.40446
##
## Coefficients:
##             Estimate Std. Error z-value Pr(>|z|)
## (Intercept) 1.949578   0.745137  2.6164 0.008886 **
## corruption  0.044457   0.641279  0.0693 0.944730
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Total Sum of Squares:    219.78
## Residual Sum of Squares: 219.76
## R-Squared:      8.8993e-05
## Adj. R-Squared: -0.018428
## Chisq: 0.00480607 on 1 DF, p-value: 0.94473
```

F-test suggests no predictive power from this model.

This or fixed effects? Seems like a pretty clear answer, given that near-zero R-Squared, but still, a Hausman Test:

```
phtest(mod.A, mod.B)
```

```
##
##  Hausman Test
##
## data:  econ.growth ~ corruption
## chisq = 6.4338, df = 1, p-value = 0.0112
## alternative hypothesis: one model is inconsistent
```

Fail. Use fixed-effects.

A Work in Progress

I will leave it to you (and to myself in a future edition of these notes) to write up similar instructions for discrete outcomes, and for random slopes models. If you would like to contribute to these notes, it will benefit your future colleagues. You will get co-authorship for any script to which you make substantive contributions. Also, Breusch-Pagan test, Dickey-Fuller,

Citations

Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.R package version 5.2.2. https://CRAN.R-project.org/package=stargazer