Lecture 12

author: John S date: October 21, 2015

More on *apply family ==========

- Continued...
- Often things that can be done in a loop can often be done with an "apply type" function.
- Sometimes this is faster.
- It can be easier to program and read.
- **split** the data - **apply** the function - **combine** the results
- how's midterm going?

Today

- Extended example with strike data

Strike Data

From Bruce Western, Sociology, Harvard.

- Data frame of 8 columns: country, year, days on strike per 1000 workers, unemployment, inflation, left-wing share of gov't, centralization of unions, union density
- 625 observations from 18 countries, 1951–1985
- Since $18 \times 35 = 630 > 625$, some years missing from some countries

Strike Data

```
setwd("/Users/ankitashankhdhar/Documents/Grad 2nd yr/Comp Stats/Lecture Notes")
strike <- read.csv("strikes.csv")
names(strike)
```

```
## [1] "country"         "year"            "strike.volume"   "unemployment"
## [5] "inflation"       "left.parliament" "centralization"  "density"
```

```
strike[1, ]
```

```
##     country year strike.volume unemployment inflation left.parliament
## 1 Australia 1951           296          1.3      19.8              43
##   centralization density
## 1      0.3748588      NA
```

Question

Does having a friendlier government make labor action more or less likely?

More specific: Is there a relationship between a country's ruling party alignment (left vs. right) and the volume of strikes?

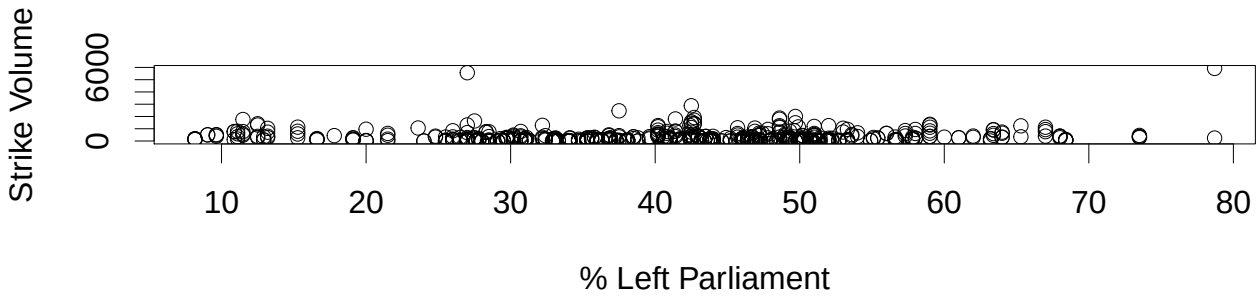
One way to address: linear regression by country.

Covariate: left.parliament

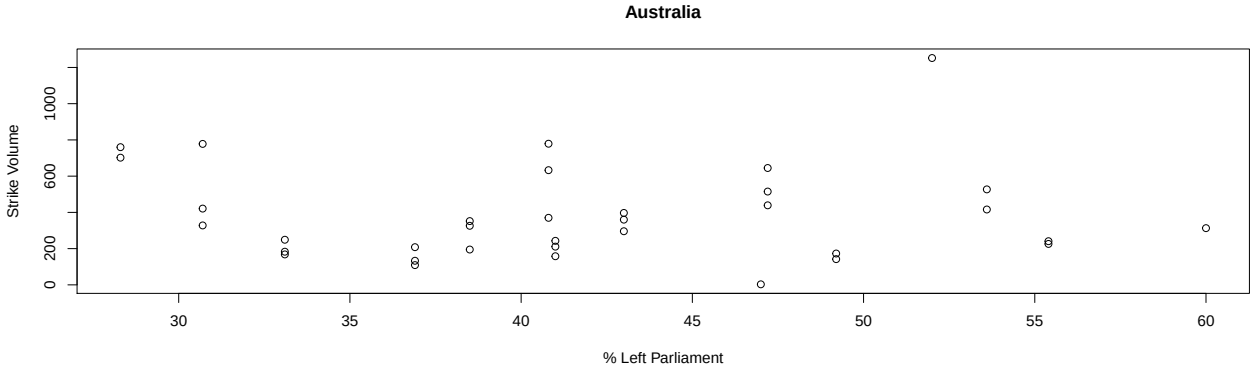
Response: strike.volume

Why is it important to do it by country?

```
par(cex = 2)
plot(strike$left.parliament, strike$strike.volume, xlab = "% Left Parliament",
    ylab = "Strike Volume")
```



First step: look at all the data!



Use par(mfrow=c(6,3)) to make for all 18 countries.

Could use a for loop:

```
coefs <- data.frame(country = unique(strike$country), int = NA, slope = NA)
for (i in 1:dim(coefs)[1]) {
    temp <- strike[strike$country == coefs$country[i], ]
    coefs[i, 2:3] <- lm(strike.volume ~ left.parliament, data = temp)$coef
}
head(coefs)
```

```
##     country         int      slope
## 1 Australia   414.77123 -0.8638052
## 2   Austria   423.07728 -8.2108864
## 3   Belgium   -56.92678  8.4474627
## 4    Canada  -227.82177 17.6766029
## 5   Denmark -1399.35735 34.3447662
## 6   Finland   108.22451 12.8422018
```

Another way: sapply()

- sapply(x,fun,...)
- write a function that takes one country's data, fits model, and return coefficients.

```
fit.for.one.country <- function(one.country.data) {
    coef <- lm(strike.volume ~ left.parliament, data = one.country.data)$coef
    return(coef)
}
```

sapply

Make a list that has datasets for each country in each element

```
datasets <- split(strike, strike$country)
names(datasets)
```

```
##  [1] "Australia"   "Austria"     "Belgium"     "Canada"      "Denmark"
##  [6] "Finland"     "France"      "Germany"     "Ireland"     "Italy"
## [11] "Japan"       "Netherlands" "New.Zealand" "Norway"      "Sweden"
## [16] "Switzerland" "UK"          "USA"
```

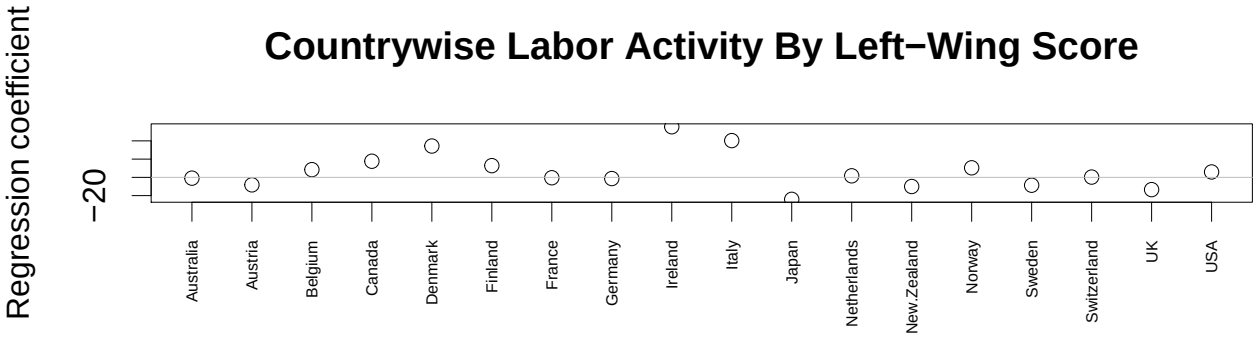
sapply

Make a list that has datasets for each country in each element

```
fits <- sapply(datasets, fit.for.one.country)
fits[1:2, 1:3]
```

```
##                   Australia    Austria    Belgium
## (Intercept)     414.7712254 423.077279 -56.926780
## left.parliament  -0.8638052  -8.210886   8.447463
```

plot results: not too conlusive...



Other ways:

- by() (Please see book.)
- lapply(): same as sapply, but it gives a list of outcomes

Summary:

- **split - apply- combine** can be done with a loop
- *apply family is another option
- sometimes much more efficient
- easier to code and read (after you learn it!)
- but, output can have a format that is hard to control
- Next: plyr library will address that problem