Session 6 Benford's Law

Data and Stata Code

All the Stata code can be found in the do-file.

Get data from Compustat

WRDS -> Compustat -> North America-Daily -> Simplified Financial Statement Extract

Step 1: Data range: 10 years (eg. 2001 to 2010)

Step 2: Search the entire database

Step 3: Variables: # Variables from three financial statements: 69

- Identifying Information/Codes: CUSIP, Fiscal Year-end Month of Data
- Balance Sheet Variables: ALL (25)
- Income Statement Variables: ALL (14)
- Statement of Cash Flows Variables: ALL (13 + 8 + 9)

Step 4: .dta

Save the dataset as Compustat- simplified.dta.

Clean the data and generate first digit

Drop the observations with negative total assets.

```
. drop if at <=0
(470 observations deleted)
```

Take the absolute value of negative numbers, replace decimals by *100, and generate the first digit.

```
. foreach var of varlist aco- xsga {
  2. replace `var'=abs(`var')
  3. replace `var'=`var'*100 if `var'<=1 &`var'>0
  4. tostring `var', replace force
  5. gen `var'_1digit=substr(`var',1,1)
  6. }
```

- Use foreach to loop over all the variables
  - Foreach var of varlist: for each variable in the variable list aco – xsga
- Change the negative values to absolute values
- Change the decimals to whole number
- Convert the variables to string type, and generate the `var'_1digit using substr

Use loop to convert the `var'_1digit to numeric type.

```
. foreach var of varlist aco_1digit- xsga_1digit {
  2. destring `var', replace
  3. }
```

Generate the dummy matrices indicating the numbers

Generate a dummy matrix to proxy if a number is 1,2,3,4,....

```
. forvalues i=0/9 {
  2. foreach var of varlist aco_1digit- xsga_1digit {
  3. gen `var'dummy`i'=1 if `var'==`i'
  4. }
  5. }
```

- Use loop to generate a matrix containing dummy variables
    - For each variable, check if it equals to i, generate dummy
        - Yes: `var'dummy`i' = 1
        - No: `var'dummy`i' = missing

Generate a dummy matrix to indicate the value missing.

```
. foreach var of varlist aco_1digit- xsga_1digit {
  2. gen `var'dummy_missing=1 if `var'==.
  3. }
```

- Use loop to generate a matrix containing dummy variables
    - For each variable, check if it is missing, generate dummy
        - Yes: `var'dummy_missing = 1
        - No: `var'dummy_missing = missing

Calculate the frequency

Calculate the frequency of each leading digit.

```
. forvalues i=0/9 {
  2. egen freq_`i'=rowtotal( aco_1digitdummy`i'- xsga_1digitdummy`i')
  3. }
```

- Use loop to calculate the frequency of each leading digit
    - For each row: calculate the sum of the dummy variables ending with number i

Calculate the frequency of missing.

```
. egen freq_missing=rowtotal( aco_1digitdummy_missing- xsga_1digitdummy_missing)
```

Calculate the empirical distribution

Drop the observations which have more than half of variables are missing. (69*50% ≈35)

```
. drop if freq_missing>35
(18,272 observations deleted)
```

Calculate the distribution of each digit.

```
. forvalues i=1/9 {
  2. gen prop_`i'= freq_`i'/(69- freq_0-freq_missing)
  3. }
```

- Use loop to calculate the distribution = frequency of number i / total numbers (exclude 0 and missing)

You may use `sum prop_1 - prop_9` to look at the empirical distribution.

Generate Benford's theoretical distribution

$$\text{P(the first digit is d)} = \text{Log}_{10}(\text{d}+1) - \text{Log}_{10}(\text{d}), \quad \text{where d} = 1, 2, \ldots, 9.$$

Use loop to calculate the Benford's distribution.

```
. forvalues i=1/9{
  2. gen benford`i'=log10(`i'+1)-log10(`i')
  3. }
```

Calculate KS statistics

$$\text{KS} = Max(|\text{AD}_1 - \text{ED}_1|, |(\text{AD}_1 + \text{AD}_2) - (\text{ED}_1 + \text{ED}_2)|, \ldots, |(\text{AD}_1 + \text{AD}_2 + \cdots + \text{AD}_9) - (\text{ED}_1 + \text{ED}_2 + \cdots + \text{ED}_9)|$$

Calculate the actual cumulative distribution for i: prop_cum_`i' = AD1+AD2+…+ADi

```
. forvalue i=1/9 {
  2. egen prop_cum_`i'=rowtotal(prop_1-prop_`i')
  3. }
```

Calculate the expected cumulative distribution for i: bendford_cum_`i' = ED1+ED2+…+EDi

```
. forvalues i=1/9 {
  2. egen benford_cum_`i'=rowtotal( benford1-benford`i')
  3. }
```

Calculate the absolute value of deviation of the actual cumulative distribution from the expected cumulative distribution for i: ks_deviation`i' = |(AD1+AD2+…+ADi) – (ED1+ED2+…+EDi)|

```
. forvalues i=1/9 {
  2. gen ks_deviation`i'=abs( prop_cum_`i'- benford_cum_`i')
  3. }
```

Find the maximum of the ks_deviation`i', and get the KS statistic.

```
. egen KS=rowmax( ks_deviation1- ks_deviation9)
```

Calculate the test statistic.

$$\text{Test value} = 1.36/\sqrt{\text{P}}$$

```
. gen ks_test=1.36/(sqrt(69-freq_0-freq_missing))
```

- P: total number of first digits used (exclude 0 and missing)

Generate the dummy comparing KS statistic the test value.

```
. gen follow= KS< ks_test
```

- If KS < ks_test: fail to reject the H0. (H0: Empirical distribution follows Benford's distribution)

Calculate the MAD

$$MAD = (\sum_{i=1}^{K} |AD\text{-}ED|)/K$$

Calculate the absolute value of deviation of the actual distribution from the expected distribution for i: mad_deviation`i' = |(ADi – EDi)|

```
. forvalues i=1/9 {
  2. gen mad_deviation`i'=abs( prop_`i'- benford`i')
  3. }
```

Sum the mad_deviation`i', divide by K, and get the MAD statistic.

```
. egen MAD=rowtotal(mad_deviation1- mad_deviation9)

. replace MAD=MAD/9
```

- K: number of leading digits being analyzed