R By within Individual Groups Variables, Averages

Fan Wang

2020-04-01

Contents

Nested within Group Stats

Go to the **RMD**, **R**, **PDF**, or **HTML** version of this file. Go back to fan's REconTools Package, R4Econ Repository (bookdown site), or Intro Stats with R Repository.

By Multiple within Individual Groups Variables, Averages for All Numeric Variables within All Groups of All Group Variables (Long to very Wide). Suppose you have an individual level final outcome. The individual is observed for N periods, where each period the inputs differ. What inputs impacted the final outcome?

Suppose we can divide N periods in which the individual is in the data into a number of years, a number of semi-years, a number of quarters, or uneven-staggered lengths. We might want to generate averages across individuals and within each of these different possible groups averages of inputs.

Then we want to version of the data where each row is an individual, one of the variables is the final outcome, and the other variables are these different averages: averages for the 1st, 2nd, 3rd year in which indivdiual is in data, averages for 1st, . . . , final quarter in which indivdiual is in data.

Build Function This function takes as inputs:

1. **vars.not.groups2avg**: a list of variables that are not the within-indivdiual or across-individual grouping variables, but the variables we want to average over. Withnin indivdiual grouping averages will be calculated for these variables using the not-listed variables as within indivdiual groups (excluding vars.indi.grp groups).
2. **vars.indi.grp**: a list or individual variables, and also perhaps villages, province, etc id variables that are higher than individual ID. Note the groups are are ACROSS individual higher level group variables.
3. the remaining variables are all within individual grouping variables.

the function output is a dataframe:

1. each row is an individual
2. initial variables individual ID and across individual groups from *vars.indi.grp*.
3. other variables are all averages for the variables in *vars.not.groups2avg*
   - if there are 2 within individual group variables, and the first has 3 groups (years), the second has 6 groups (semi-years), then there would be 9 average variables.
   - each average variables has the original variable name from vars.not.groups2avg plus the name of the within individual grouping variable, and at the end 'c_x', where x is a integer representing the category within the group (if 3 years, x=1, 2, 3)

```
# Data Function
# https://fanwangecon.github.io/R4Econ/summarize/summ/ByGroupsSummWide.html
f.by.groups.summ.wide <- function(df.groups.to.average,
                                  vars.not.groups2avg,
```

```
                                    vars.indi.grp = c('S.country','ID'),
                                    display=TRUE) {

# 1. generate categoricals for full year (m.12), half year (m.6), quarter year (m.4)
# 2. generate categoricals also for uneven years (m12t14) using stagger (+2 rather than -1)
# 3. reshape wide to long, so that all categorical date groups appear in var=value,
    # and categories in var=variable
# 4. calculate mean for all numeric variables for all date groups
# 5. combine date categorical variable and value, single var:
    # m.12.c1= first year average from m.12 averaging

######## ######## ######## ######## #######
# Step 1
######## ######## ######## ######## #######
# 1. generate categoricals for full year (m.12), half year (m.6), quarter year (m.4)
# 2. generate categoricals also for uneven years (m12t14) using stagger (+2 rather than -1)

######## ######## ######## ######## #######
# S2: reshape wide to long, so that all categorical date groups appear in var=value,
    # and categories in var=variable; calculate mean for all numeric variables for all date groups
######## ######## ######## ######## #######
df.avg.long <- df.groups.to.average %>%
       gather(variable, value, -one_of(c(vars.indi.grp,
                                         vars.not.groups2avg))) %>%
       group_by(!!!syms(vars.indi.grp), variable, value) %>%
       summarise_if(is.numeric, funs(mean(., na.rm = TRUE)))

if (display){
  dim(df.avg.long)
  options(repr.matrix.max.rows=10, repr.matrix.max.cols=20)
  print(df.avg.long)
}

######## ######## ######## ######## #######
# S3 combine date categorical variable and value, single var:
#    m.12.c1= first year average from m.12 averaging; to do this make data even longer first
######## ######## ######## ######## #######

# We already have the averages, but we want them to show up as variables,
    # mean for each group of each variable.
df.avg.allvars.wide <- df.avg.long %>%
   ungroup() %>%
   mutate(all_m_cate = paste0(variable, '_c', value)) %>%
   select(all_m_cate, everything(), -variable, -value) %>%
   gather(variable, value, -one_of(vars.indi.grp), -all_m_cate) %>%
   unite('var_mcate', variable, all_m_cate) %>%
   spread(var_mcate, value)

if (display){
  dim(df.avg.allvars.wide)
  options(repr.matrix.max.rows=10, repr.matrix.max.cols=10)
  print(df.avg.allvars.wide)
}
```

```
return(df.avg.allvars.wide)
}
```

Test Program In our sample dataset, the number of nutrition/height/income etc information observed within each country and month of age group are different. We have a panel dataset for children observed over different months of age.

We have two key grouping variables: 1. country: data are observed for guatemala and cebu 2. month-age (survey month round=svymthRound): different months of age at which each individual child is observed

A child could be observed for many months, or just a few months. A child's height information could be observed for more months-of-age than nutritional intake information. We eventually want to run regressions where the outcome is height/weight and the input is nutrition. The regressions will be at the month-of-age level. We need to know how many times different variables are observed at the month-of-age level.

```
# Library
library(tidyverse)

# Load Sample Data
setwd('C:/Users/fan/R4Econ/_data/')
df <- read_csv('height_weight.csv')
```

```
## Parsed with column specification:
## cols(
##   S.country = col_character(),
##   vil.id = col_double(),
##   indi.id = col_double(),
##   sex = col_character(),
##   svymthRound = col_double(),
##   momEdu = col_double(),
##   wealthIdx = col_double(),
##   hgt = col_double(),
##   wgt = col_double(),
##   hgt0 = col_double(),
##   wgt0 = col_double(),
##   prot = col_double(),
##   cal = col_double(),
##   p.A.prot = col_double(),
##   p.A.nProt = col_double()
## )
```

Generate Within Individual Groups In the data, children are observed for different number of months since birth. We want to calculate quarterly, semi-year, annual, etc average nutritional intakes. First generate these within-individual grouping variables. We can also generate uneven-staggered calendar groups as shown below.

```
mth.var <- 'svymthRound'
df.groups.to.average<- df %>%
        filter(!!sym(mth.var) >= 0 & !!sym(mth.var) <= 24)  %>%
        mutate(m12t24=(floor((!!sym(mth.var) - 12) %/% 14) + 1),
               m8t24=(floor((!!sym(mth.var) - 8) %/% 18) + 1),
               m12 = pmax((floor((!!sym(mth.var)-1) %/% 12) + 1), 1),
               m6 = pmax((floor((!!sym(mth.var)-1) %/% 6) + 1), 1),
               m3 = pmax((floor((!!sym(mth.var)-1) %/% 3) + 1), 1))
```

```
# Show Results
options(repr.matrix.max.rows=30, repr.matrix.max.cols=20)
vars.arrange <- c('S.country','indi.id','svymthRound')
vars.groups.within.indi <- c('m12t24', 'm8t24', 'm12', 'm6', 'm3')
as.tibble(df.groups.to.average %>%
          group_by(!!!syms(vars.arrange)) %>%
          arrange(!!!syms(vars.arrange)) %>%
          select(!!!syms(vars.arrange), !!!syms(vars.groups.within.indi)))
```

```
## # A tibble: 23,603 x 8
##    S.country indi.id svymthRound m12t24 m8t24   m12    m6    m3
##    <chr>       <dbl>       <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Cebu            1           0      0     0     1     1     1
##  2 Cebu            1           2      0     0     1     1     1
##  3 Cebu            1           4      0     0     1     1     2
##  4 Cebu            1           6      0     0     1     1     2
##  5 Cebu            1           8      0     1     1     2     3
##  6 Cebu            1          10      0     1     1     2     4
##  7 Cebu            1          12      1     1     1     2     4
##  8 Cebu            1          14      1     1     2     3     5
##  9 Cebu            1          16      1     1     2     3     6
## 10 Cebu            1          18      1     1     2     3     6
## # ... with 23,593 more rows
```

Within Group Averages With the within-group averages created, we can generate averages for all variables within these groups.

```
vars.not.groups2avg <- c('prot', 'cal')
vars.indi.grp <- c('S.country', 'indi.id')
vars.groups.within.indi <- c('m12t24', 'm8t24', 'm12', 'm6', 'm3')

df.groups.to.average.select <- df.groups.to.average %>%
                        select(one_of(c(vars.indi.grp,
                                        vars.not.groups2avg,
                                        vars.groups.within.indi)))
df.avg.allvars.wide <- f.by.groups.summ.wide(df.groups.to.average.select,
                                             vars.not.groups2avg,
                                             vars.indi.grp, display=TRUE)
```

```
## # A tibble: 36,414 x 6
## # Groups:   S.country, indi.id, variable [10,115]
##    S.country indi.id variable value   prot   cal
##    <chr>       <dbl> <chr>    <dbl>  <dbl> <dbl>
##  1 Cebu            1 m12          1   5.36 132.
##  2 Cebu            1 m12          2 NaN    NaN
##  3 Cebu            1 m12t24       0   4.37  97.1
##  4 Cebu            1 m12t24       1  11.3  343.
##  5 Cebu            1 m3           1   0.65   9.1
##  6 Cebu            1 m3           2   3.65  95.5
##  7 Cebu            1 m3           3   2.6   85.3
##  8 Cebu            1 m3           4  13.2  315.
##  9 Cebu            1 m3           5 NaN    NaN
## 10 Cebu            1 m3           6 NaN    NaN
## # ... with 36,404 more rows
```

```
## # A tibble: 2,023 x 38
##    S.country indi.id cal_m12_c1 cal_m12_c2 cal_m12t24_c0 cal_m12t24_c1 cal_m3_c1 cal_m3_c2 cal_m3_c3
##    <chr>       <dbl>      <dbl>      <dbl>         <dbl>         <dbl>     <dbl>     <dbl>     <dbl>
##  1 Cebu            1      132.        NaN           97.1          343.      9.1       95.5      85.3
##  2 Cebu            2       90.7       256.          81.5          240.     83.4       12.3     155.
##  3 Cebu            3       96.8       659.          31.6          634.      0.5       28.8      57
##  4 Cebu            4       27.5       372.          24.6          325.      4.5       26.0      39.4
##  5 Cebu            5      101.       1081.          79.2          960.     14.1      144.       71.3
##  6 Cebu            6      185.        522.         162.           493.     23.8      185.      169.
##  7 Cebu            7      157.        571.         146.           514.      8.3      138.      408.
##  8 Cebu            8      472.        845.         379.           871.    159.       423       418.
##  9 Cebu            9       32.3       415.          16.6          374.      5.05      10.4      15.1
## 10 Cebu           10       67.2       395.          68.6          347.      9.55      26.4     165.
## # ... with 2,013 more rows, and 24 more variables: cal_m6_c1 <dbl>, cal_m6_c2 <dbl>, cal_m6_c3 <dbl>
## #   cal_m8t24_c1 <dbl>, prot_m12_c1 <dbl>, prot_m12_c2 <dbl>, prot_m12t24_c0 <dbl>, prot_m12t24_c1 <
## #   prot_m3_c3 <dbl>, prot_m3_c4 <dbl>, prot_m3_c5 <dbl>, prot_m3_c6 <dbl>, prot_m3_c7 <dbl>, prot_m
## #   prot_m6_c3 <dbl>, prot_m6_c4 <dbl>, prot_m8t24_c0 <dbl>, prot_m8t24_c1 <dbl>
```

This is the tabular version of results

```
dim(df.avg.allvars.wide)
```

```
## [1] 2023   38
```

```
names(df.avg.allvars.wide)
```

```
##  [1] "S.country"      "indi.id"        "cal_m12_c1"     "cal_m12_c2"     "cal_m12t24_c0"  "cal_m12t24
##  [9] "cal_m3_c3"      "cal_m3_c4"      "cal_m3_c5"      "cal_m3_c6"      "cal_m3_c7"      "cal_m3_c8
## [17] "cal_m6_c3"      "cal_m6_c4"      "cal_m8t24_c0"   "cal_m8t24_c1"   "prot_m12_c1"    "prot_m12_
## [25] "prot_m3_c1"     "prot_m3_c2"     "prot_m3_c3"     "prot_m3_c4"     "prot_m3_c5"     "prot_m3_c
## [33] "prot_m6_c1"     "prot_m6_c2"     "prot_m6_c3"     "prot_m6_c4"     "prot_m8t24_c0"  "prot_m8t24
```

```
options(repr.matrix.max.rows=30, repr.matrix.max.cols=12)
df.avg.allvars.wide
```

```
## # A tibble: 2,023 x 38
##    S.country indi.id cal_m12_c1 cal_m12_c2 cal_m12t24_c0 cal_m12t24_c1 cal_m3_c1 cal_m3_c2 cal_m3_c3
##    <chr>       <dbl>      <dbl>      <dbl>         <dbl>         <dbl>     <dbl>     <dbl>     <dbl>
##  1 Cebu            1      132.        NaN           97.1          343.      9.1       95.5      85.3
##  2 Cebu            2       90.7       256.          81.5          240.     83.4       12.3     155.
##  3 Cebu            3       96.8       659.          31.6          634.      0.5       28.8      57
##  4 Cebu            4       27.5       372.          24.6          325.      4.5       26.0      39.4
##  5 Cebu            5      101.       1081.          79.2          960.     14.1      144.       71.3
##  6 Cebu            6      185.        522.         162.           493.     23.8      185.      169.
##  7 Cebu            7      157.        571.         146.           514.      8.3      138.      408.
##  8 Cebu            8      472.        845.         379.           871.    159.       423       418.
##  9 Cebu            9       32.3       415.          16.6          374.      5.05      10.4      15.1
## 10 Cebu           10       67.2       395.          68.6          347.      9.55      26.4     165.
## # ... with 2,013 more rows, and 24 more variables: cal_m6_c1 <dbl>, cal_m6_c2 <dbl>, cal_m6_c3 <dbl>
## #   cal_m8t24_c1 <dbl>, prot_m12_c1 <dbl>, prot_m12_c2 <dbl>, prot_m12t24_c0 <dbl>, prot_m12t24_c1 <
## #   prot_m3_c3 <dbl>, prot_m3_c4 <dbl>, prot_m3_c5 <dbl>, prot_m3_c6 <dbl>, prot_m3_c7 <dbl>, prot_m
## #   prot_m6_c3 <dbl>, prot_m6_c4 <dbl>, prot_m8t24_c0 <dbl>, prot_m8t24_c1 <dbl>
```