STATA ressources for data processing and analysis...

IBRAHIMA TALL,

- STATISTICIAN AND ECONOMIST ENGINEER,
- Tel: +221772382463,
- Email: datasciences4ise@gmail.com (mailto:datasciences4ise@gmail.com),
- Github : Ibrahima Tall (https://github.com/IbrahimaTall)

Data scientist at national agency of statistic and demograpy (ANSD), Dakar, Senegal

---

This is ressources for stata to help data scientist in processing ans analysing data. This is because of STATA is our main software for processing and analysing data. In this notebook i present many commands often use for data scientist and analyst.

I. Data loading

Data loading includes log files using, looking help for package, packages installation, looking for files, managing directory and data using.

I.4 Managing directory

In [ ]:
```
cd "C:\Users\ibtall\Documents" //Set and define the directory
```

In [ ]:
```
findfile exportstata.ipynb, all //To look at the file
```

In [ ]:
```
fs *.ipynb // print all file in dta format
```

In [ ]:
```
dir // print filename in courant directory
```

In [ ]:
```
ls //Same as dir command
```

In [ ]:
```
pwd //Print working directory
```

In [ ]:
```
which regress //To see the version of installed ado
```

In [ ]:
```
mkdir new_folder
```

In [ ]:
```
sysdir //Print all stata system directory for installing
```

In [ ]:
```
sysuse dir //Listing the datasets in stata memory
```

In [ ]:
```
erase data.dta
```

I.1 Log using to save the work

In [ ]:
```
// In order to save our work, we use a log file
log using myfile, text replace
/* log off: to stop saving
   log on: to reactive the save
log close */
```

I.2 Looking for help and research

In [ ]:
```
* Help command: use command
help use
help search
```

I.3 Installing stata packages

In [ ]:
```
* To install ado-file: outreg ado
ssc install fs
net install xtable
```

I.5 Loading, saving and exporting data

In [ ]:
```
sysuse citytemp, clear
```

In [ ]:
```
use "Outputs\data.dta", clear // load stata data in specific path direction
```

In [ ]:
```
import excel "dt.xlsx", sheet("sheet1") cellrange(A1:C20) firstrow case("lo
```

In [ ]:
```
import delimited "dt.csv", rowrange(2:20) colrange(1:8) varname(2)
```

In [ ]:
```
webuse set "https/www.ansd.sn/data"
```

In [ ]:
```
webuse "data"
```

In [ ]:
```
save "mydata", nolabel replace orphans // orphans for saving values lables
```

In [ ]:
```
saveold "myolddata", version(12) replace nolabel //Saving currant data in sp
```

In [ ]:
```
export excel using mydata.xlsx, replace //Saving currant data in Excel fomat
```

In [ ]:
```
export delimited using mydata.csv, delimiter(",") replace //Saving in text f
```

II. Data treatment and wrangling

Here, we present commands that every data scientist will need to process data in STATA.

II.1 Looking at the data

In [ ]:
```
sort region, stable //Sorting data by region and conserved
```

In [ ]:
```
gsort division -region
```

In [ ]:
```
varmanage //To manage varibles attributes to the "variables manage" window
```

In [ ]:
```
format tempjuly tempjan %-8.2fc // format types: %0#.#gc; %-#.#fc; %-#.#e; %
```

In [ ]:
```
list heatdd if inrange(region,1,3) & ! missing(division)
```

In [ ]:
```
ds, not(type byte)
ds, has(varlabel "region") insensitive
```

In [ ]:
```
lookfor "region" //Research variables contening some world
```

In [ ]:
```
browse in 1/20 //Take a look at the data : Ctrl+8
```

In [ ]:
```
count if inlist(region, 1,2)
```

In [ ]:
```
assert inrange(division, 1, 40) //Verify some logic in whithin variable valu
```

In [ ]:
```
describe region
```

In [ ]:
```
codebook division, header notes //Get informations on variables and data set
```

In [ ]:
```
by region, sort: inspect tempjuly tempjan //Display summaries of variables u
```

In [ ]:
```
bysort region: summarize heatdd, meanonly
```

In [ ]:
```
sample 10, by(region) count // (10% without count option)
```

In [ ]:
```
notes region: senegal is not concerning //Add notes to data or variables and
notes region //Display added notes to the variable
```

II.2 Changing variables types and duplicates values managing

In [ ]:
```
recast double region, force //Change the type of variable
```

In [ ]:
```
tostring region, gen(region_str)
```

In [ ]:
```
destring region_str, force replace
```

In [ ]:
```
decode division, gen(division_str) maxlength(20)
```

In [ ]:
```
encode division_str, gen(division_bis) label(division) // label() to specify
```

In [ ]:
```
decode division_bis, gen(division_str2) label(division)
```

```
In [ ]: mvdecode division region, mv(99 88) // Replace all 88 and 99 by sysmis value
In [ ]: mvencode _all if regionn == 1 | division < 3, mv(99) //Replace all missing v
In [ ]: isid region //Look whatever variable identify uniquely observations
In [ ]: duplicates report region //Look for number of duplicates values in variable
In [ ]: duplicates list region division, sepby(region) //Listing duplicates values o
In [ ]: duplicates examples region //List some examples of duplicates values of vari
In [ ]: duplicates tag region, gen(region_duplic) //Generate new variable of number
In [ ]: duplicates drop idvar, force //Drop all duplicated values whithin variable
```

II.3 Managing Labels and variables renaming

```
In [ ]: label data "This data base is related to climate informations" //Labelling o
In [ ]: label variable region "The region's names"
In [ ]: label define mylab 1 Hot 2 cold                    //Defining label values name
In [ ]: label define mylab 3 "litle hot", add replace    //Adding new code to existi
In [ ]: label define mylab 1 "very hot", modify replace //Modifying code label in ex
In [ ]: label values myvar mylab // Assigning value label to a variable
In [ ]: label dir //Printing existing value label names
In [ ]: label list //Listing name and content of existing label value
In [ ]: label list mylab //Listing content of specific value label
In [ ]: label copy mylab mynewlab, replace // copy mylab into mynewlab and replace,
```

```
In [ ]: label save using labdofile, replace // save all value label in a do file, re
In [ ]: label save mylab using labfile, replace // save only valu label named mylab
In [ ]: label drop _all // drop all value label, we can specify the value label name
In [ ]: recode v1 (3/5=0 "Value 0") (1/2=1 "Value 1"), gen(newv1) label(mylabel) //
In [ ]: recode x1 x2 (1=5) (2=4) (3=3) (4=2) (5=1), pre(n) test // Changing x1 x2 va
In [ ]: levelsof region // See levels of categorical variables
In [ ]: levelsof region, missing local(region_levels) //store levels, including miss
In [ ]: labelbook, limit(20) problems detail //all (max 20) value label and var link
In [ ]: numlabel mylab, add mask("#.")     //Transforme label "very hot" --> "1. ver
In [ ]: numlabel mylab, remove mask("#.") //Delete the previouse format
In [ ]: uselabel using labelbase, clear var //Create dataset of all value label (we
In [ ]: label language //list the existing label language
In [ ]: label language french, new //Create new set of label
In [ ]: label language french, copy //Create new set of label by copying the existin
In [ ]: label language french // change label to french label language with is defin
In [ ]: label language eng, rename // rename current label set to eng
In [ ]: label language french, delete // delete label language named french
In [ ]: rename region myregion
In [ ]: rename (myregion zone)(region newzone)
```

In [ ]:
```
ssc install elabel
```

In [ ]:
```
elabel variable (var1 var2)("label 1" "label 2")
```

In [ ]:
```
elabel define lname 1 "lname 1" 2 lname2 // Realy does the same as label de
```

In [ ]:
```
elabel values (var1 var2)(lbl1 lbl2)
```

In [ ]:
```
elabel dir, current // nomemory
```

In [ ]:
```
elabel list, current // nomemory varlist
```

In [ ]:
```
elabel remove lnamelist, not// remove all except lnamelist
```

In [ ]:
```
elabel drop lname // same to label drop
```

In [ ]:
```
elabel keep lname
```

In [ ]:
```
elabel copy oldlname newlname // same to label copy
```

In [ ]:
```
elabel save lname using mylabel, replace
```

In [ ]:
```
elabel compare lname1 lname2
```

In [ ]:
```
elabel duplicates report
```

In [ ]:
```
elabel duplicates drop
```

In [ ]:
```
elabel duplicates retain
```

In [ ]:
```
elabel load using filename, lname(lname) value(value) label(label)
```

In [ ]:
```
elabel recode lname (1=3 3/7=7/3), define(newlname)
```

In [ ]:
```
elabel recode lnamelist (2 = .a "Missimg"), dryrun
```

In [ ]:
```
elabel rename (oldlnameslist)(newlnameslist), force
```

In [ ]:
```
elabel rename oldlnames, upper //lowe proper
```

II.4 Creating variables

In [ ]:
```
generate bytes zone = heatdd < mean(heatdd) //Create new variables
```

In [ ]:
```
generate agecat = autocode(age,4,18,65) // 4 equal groups betwen 18 and 65
```

In [ ]:
```
generate byte agecat = recode(age,21,38,64,75) // Groups: . < 21 < 38 < 64 <
```

In [ ]:
```
egen myv_count = anycount(division region), values(1 2 3) //Number values ir
```

In [ ]:
```
egen myv_match = anymatch(division region), values(1 2 3) //True (1) or fals
```

In [ ]:
```
egen myv_vlues = anyvalue(division), values(1 2 3) //Value of division corre
```

In [ ]:
```
egen myv_concat = concat(division region), punct("") //Format(%9s) decode ma
```

In [ ]:
```
egen myv_nbnonmiss = count(heatdd), by(division region)
```

In [ ]:
```
egen tempjanclass = cut(tempjan), at(2(10)73) label // == egen tempjanclass
```

In [ ]:
```
egen tempjanclass2 = cut(tempjan), group(5) // == egen tempjanclass = cut(te
```

In [ ]:
```
egen myv_diff = diff(division region) //1 if division is different to regior
```

In [ ]:
```
egen myv_sub = ends(division_str), punct(" ") trim last //Trim for deleting
```

In [ ]:
```
egen myn_fill = fill(11 13 15 17 19 21 23 27) //Listed numbers by increamen
```

In [ ]:
```
egen myn_group = group(division_str region), missing label truncate(5) //Lab
```

In [ ]:
```
egen myn_group = group(division_str region), missing label truncate(5) //Lab
```

In [ ]:
```
egen myv_iqr = iqr(tempjuly+tempjan), by(division region) //Ingter Quartile
```

In [ ]:
```
egen myv_pctile = pctile(tempjuly+tempjan), by(division region) p(25) //Ingt
```

In [ ]:
```
egen myn_kurt = kurt(heatdd), by(division region) //Kurtosis of heatdd
```

In [ ]:
```
egen myn_skew = skew(heatdd), by(division region) //Skewness of heatdd
```

In [ ]:
```
egen myv_mad = mad(tempjuly+tempjan), by(division region)
```

In [ ]:
```
egen myv_max = max(tempjuly+tempjan), by(division region)
```

In [ ]:
```
egen myv_mdev = mdev(tempjuly+tempjan), by(division region)
```

In [ ]:
```
egen myv_mean = mean(tempjuly+tempjan), by(division region)
```

In [ ]:
```
egen myn_median = median(tempjuly+tempjan), by(division region)
```

In [ ]:
```
egen myv_min = min(tempjuly+tempjan), by(division region)
```

In [ ]:
```
egen myv_mod = mod(tempjan), by(division_str region) // Most commun temperat
```

In [ ]:
```
egen myv_pc = pc(tempjuly+tempjan), by(division region) //Prop obtion to obt
```

In [ ]:
```
egen myv_rank = rank(tempjuly+tempjan), by(division region) unique //Field t
```

In [ ]:
```
* rowfirst(), rowlast(), rowmax(), rowmean(), rowmedian(), rowpctile() [, p
* row[non]miss()==nb of [non]missing, rowsd(), rowtotal(),
egen myv_nomiss = rownonmiss(tempjuly tempjan division_str), strok //This op
```

In [ ]:
```
egen myv_tot = rowtotal(tempjuly tempjan), missing // missing if all are mis
```

In [ ]:
```
egen myv_sd = sd(tempjuly+tempjan), by(division region) // standard deviatio
```

In [ ]:
```
egen myv_sep = seq(), from(2) to(90) block(7) by(region division) // create
```

In [ ]:
```
egen myv_std = std(tempjuly+tempjan), mean(10) std(2)
```

In [ ]:
```
egen myv_tag = tag(division region) //, missing to include missing | look in
```

In [ ]:
```
matrix m = (2,3,4)  //Create vector of values to be used as mean
```

```
In [ ]: matrix s = (5,10,20) //Create vector of values to be used as standard error
In [ ]: drawnorm v_x v_y v_z, means(m) sds(s) //Create three variables of normal dis
In [ ]: separate tempjuly, by( inrange(region, 1,2,3) & tempjan > 10) gen(newtp) sh
In [ ]: pctile myv_decil = tempjuly, nquantiles(10) genp(percentdeci) // create two
In [ ]: xtile myv_xtile = tempjuly, nquantiles(10)      // deciles cretion
In [ ]: xtile myv_xtilcut = tempjuly, cutpoints(region) // percentiles with reion as
In [ ]: range new_square 0 7*_pi 300 // create new variable from 0 to 7*_pi of 300 
```

II.8 Combining datasets and arranging variables

```
In [ ]: append using data // Add observations to the corresponding variables
In [ ]: merge 1:1 ID using data, noreport keepusing(varlist) generate(linkvar) //Me
In [ ]: merge m:1 ID using data //Many observations in current base have same ID
In [ ]: merge 1:m ID using data //Many observations in using base have same ID
In [ ]: merge 1: _n using data //Many observations in using base have same ID
In [ ]: set obs 20 //Create new dataset with 20 observations
In [ ]: insobs 10, after(2) //nsert 10 new after the 2nd observation
In [ ]: expand 2, gen(type) //Duplicates each observation by 2, type = 0 if observa
In [ ]: order tempjuly tempjan, after(region)
In [ ]: reshape long inc@r ue, i(id) j(year)
In [ ]: reshape wide inc@r ue, i(id) j(year)
```

In [ ]:
```
reshape error //To look at the reshape error
```

In [ ]:
```
xpose, clear varname format(%6.2f) // transpose dataset observations become
```

II.9 Summarizing variables

In [ ]:
```
describe, simple
```

In [ ]:
```
summarize
```

In [ ]:
```
sumstats // an other summarize commdand
```

In [ ]:
```
preserve //Save a copy of the data in memory
```

In [ ]:
```
collapse (mean) mheatdd=heatdd (count) nbcooldd=cooldd, by(division region)
```

In [ ]:
```
statsby vmean = r(mean) vsd = r(sd), basepop(region < 4) by(region) total nc
```

In [ ]:
```
statsby _b _se, basepop(inlist(region, 1,2)) by(region) saving(restemp) tota
```

In [ ]:
```
contract tempjuly tempjan, freq(fvar) percent(pvar) float format(%9.2f) nomi
```

In [ ]:
```
compare tempjuly tempjan, by(region) // look at differences betwen two varia
```

In [ ]:
```
restore // restore the saved data by preserve
```

III. Working with string in STATA

In [ ]:
```
gen division_str2 = abbrev(division_str, 2) // Mountain and pacific will be
```

In [ ]:
```
gen indregionville = indexnot(division_str2, region_str) // position of firs
```

In [ ]:
```
gen plusregion = plurial(2, region_str, "+es") // + for add and - for substi
```

In [ ]:
```
gen logicmatch = ustrregexm(division_str2, region_str, 1) // 1 or 0 if s1 ma
```

In [ ]:
```
gen fisrt_occ = ustrregexrf(divion_str, region_str, "oui", 1) // replace by
```

In [ ]:
```
gen all_occ = ustrregexra(divion_str, region_str, "ouiall", 1) // replace by
```

In [ ]:
```
gen nospace_div = stritrim(division_str) //remove mutilple space within text
```

In [ ]:
```
gen divlen = ustrlen(division_str) // Number of chars in text of division
```

In [ ]:
```
gen lowerdiv = ustrlower(division_str, "fr") // lowercase in local french :
```

In [ ]:
```
gen left_trim = ustrltrim(division_str) // no space at left : ustrrtrim(divi
```

In [ ]:
```
split region_str, generate(newreg) parse(" ") limit(3) destring ignore("/")
            variables by parse chars, creating 3 new vars, converting in nu
```

IV. Tables and graphs

In [ ]:
```
tabulate division, gen(division_) missing nolabel sort nofreq subop(region)
```

In [ ]:
```
tabulate division region, chi2 lrchi2 cchi2 clrchi2 exact gamma taub v colu
 cell expected missing
```

In [ ]:
```
tabulate division, all // equivalent to specifying chi2 lrchi2 V gamma taub
```

In [ ]:
```
tab1 division region, sort // one-way tabulate for many variables
```

In [ ]:
```
tabulate division region, summarize(heatdd) nomean nostandard nofreq nolabe
```

In [ ]:
```
tab2 division region zone, row nofreq // Two by two tables comines(n, p)
```

In [ ]:
```
*freq, mean, sd, semean, sebinomial, sepoisson, sum, rawsum, count, n, max,
table division region, by(zone) contents( mean heatdd) center left row colu
  scolumn concise missing replace format(%9.0g) cellwidth(9) csepwidth(9) s
```

In [ ]:
```
tabstat heatdd, by(division) statistics(mean) format(%9.2fc) save // to save
```

In [ ]:
```
ir // epitab
```

In [ ]:
```
graph bar cooldd if region == 4 & division > 5, over(zone) over(region) over
```

In [ ]:
```
graph box heatdd cooldd, over(region)
```

In [ ]:
```
graph dot (mean) cooldd, over(division)
```

In [ ]:
```
graph pie cooldd, over(division) plabel(_all percent)
```

In [ ]:
```
graph save "divgrp", replace
```

In [ ]:
```
graph pie cooldd, over(region) plabel(_all percent)
```

In [ ]:
```
graph save "reggraph", replace
```

In [ ]:
```
graph rename "reggraph" "reggrp", replace
```

In [ ]:
```
graph combine "divgrp" "reggrp"
```

In [ ]:
```
graph export my2grp, as(png) width(600) height(450) replace
```

V. Programming ressources

In [ ]:
```
scalar a = 1
```

In [ ]:
```
scalar b = a + 3 //We can make opertaion with scalar
```

In [ ]:
```
display b
```

In [ ]:
```
scalar txt = "Je m'appelle" //We can make a string scalar
```

In [ ]:
```
scalar txt = txt + " Ibrahima TALL"
```

In [ ]:
```
di txt
```

In [ ]:
```
scalar dir //We can list all scalars
```

In [ ]:
```
scalar list //Same as above
```

In [ ]:
```
scalar drop _all //We can drop all scalar in memory
```

In [ ]:
```
capture local drop name //Local macros is available only within the defining
```

In [ ]:
```
local name Tall and mee
di "`name'"
```

In [ ]:
```
local i = 1 // Equal sign mean that expression on rigth will be evaluated
```

In [ ]:
```
local tp: type tempjuly                // local macro "tp" refers to variable
```

In [ ]:
```
local lbl: variable label tempjuly // local macro "lbl" refers to variable
```

In [ ]:
```
local vlblname: value label myvar // get value label name
```

In [ ]:
```
local label1 : label (myvar) 1   // get label of the value 1
```

In [ ]:
```
local label2 : label myvarlab 2 // get label of the value 2
```

In [ ]:
```
di "`: type tempjuly'" //This attributes can be used in a simple way
```

In [ ]:
```
local cmdprop: properties help //Get command properties
```

In [ ]:
```
di "`cmdprop'"
```

In [ ]:
```
quietly tab region division, nofreq row //Get the reuslts of command: scala
```

In [ ]:
```
local rescom: r(scalars)
```

In [ ]:
```
di "`rescom'"
```

In [ ]:
```
local vsort: sortedby //To see with what variables the data set is sorted
```

In [ ]:
```
di "`vsort'"
```

In [ ]:
```
global nom monpere //Global marco
di "$nom"
```

In [ ]:
```
macro dir //Listed defined macro
```

In [ ]:
```
macro list //Same as above
```

In [ ]:
```
local vlist moi et toi //To use macro shift we need tokenize command to sto
```

In [ ]:
```
tokenize `vlist'
while "`1'" ~= "" {
    display "`1'"
    macro shift
}
foreach x in 1 2 3 { //Foreach using
    di "`x'"
}
local i = 1 //While function can combine ++i, i++, --i, i--
while (`++i' < 5){
    di "`i*2'"
}
```

In [ ]:
```
capture program drop talprog //A program that calculate the number of simila
program define talprog, rclass
    version 9.1
    syntax varlist(min=2 max=2 string) [=exp] [if] [in] [iweight], [by(varli
    args x y
    marksample touse
    local i = 1
    local nb = 0
    while(i++ <= strlen(`x')){
        forvalues j = 1/strlen(`y'){
            local nb = `nb' + x[i] == y[j] if `touse' `in'
        }
    }
    return scalar nb
end
talprog //We call the above program
viewsource ml.ado //We can take a look at the content of ado-program
help marksample
```

In [ ]:
```
exit, clear
```

In [ ]:
```
help fvset
```

In [ ]: