\* Commandes Stata, Evens Salies, v1 19/10/2018, v2 10/2019, v3 10/2020

​\* Note : - Les noms des variables sont en lettre majuscule

\* - Les commandes sont justifiées à gauche

\* - Les exemples d'application un peu plus long sont en retrait

\* - Les commentaires courts commencent par "//"

\* - Les notes sont des commentaires plus longs

- - - COMMENTAIRES

// Blabla // Commentaire sur la ligne d'une commande

\* Blabla // Commentaire à la ligne

/// // Continuation de ligne. Par exemple :

**summarize** Y1 Y2 Y3 Y4 Y5 ///

Y6 Y7 Y8 Y9 // (summarize est expliqué plus loin)

**#delimit ;** ... **#delimit cr** // Même chose que ///. Par exemple :

**clear all**

**set obs** 10 // Fixe le nombre d'observations à 10

**forvalues** STEP=1(1)30 {

**generate** Y`STEP'=`STEP'

}

**#delimit ;**

**summarize** Y1 Y2 Y3 Y4 Y5 Y6 Y7 Y8 Y9 Y10 Y11 Y12 Y13 Y14 Y15 Y16 Y17 Y18

Y19 Y20 Y21 Y22 Y23 Y24 Y25 Y26 Y27 Y28 Y29 ;

**summarize** Y30 ; // Une commande doit se terminer avec ";"

**#delimit cr**

clear all // Supprime les données en mémoire

**cls** // Vide la fenêtre de résultats

**set obs** 100

**set more off** // L'affichage des résultats dans la fenêtre

// de résultats n'est pas interrompu

**Cd** "monchemin" // Chemin vers le dossier de travail. Par

// exemple cd "C:Users\Nina\Documents"

**Use** "nomdufichierstata.dta", clear // Charge données au format Stata. Option

// clear s'il y a déjà des données en

// mémoire

**import excel using** "C:\..nomduclasseurexcel.xlsx", firstrow sheet("Feuille1") clear

// Option "firstrow" si la première ligne du

// fichier Excel contient les nom des

// variables ; option sheet pour ne prendre

// que les données de "Feuille1"

**insheet using** "C:\...data.txt" // Idem, au format ASCII (.txt)

import delimited using ///

"C:\...nomdufichier.txt", delimiters("") varname(1) clear

// Pour les formats .txt ou plus exotiques

// Option "delimiters("")" si les colonnes

// séparées par un espace/une tabulation

// Option "varname(x)" traite la xème ligne

// des données comme le nom des variables

**save** "nouveaufichier.dta" // Sauve le fichier de données au format

// Stata

**save** "nouveaufichier.dta", replace // Idem, mais écrase la dernière sauvegarde

**merge** x:y CLE1 CLE2 ... using "file.dta"

**merge** fusionne une base de données en mémoire (fichier *master*) avec un fichier .dta situé dans l’ordinateur (fichier *using*), appelé dans cet ex. "file.dta". Les deux fichiers incluent deux variables (clés), CLE1 et CLE2. Les paramètres "x" et "y" valent 1 ou m, selon que les valeurs des clés apparaissent 1 fois ou m > 1 fois. Par ex., dans le fichier « Initiation à Stata, section 1, transform OECD raw data to Stata panel (.do) » sur Moodle, la clé est le nom des pays, et . La commande merge crée une variable "\_merge" contenant des informations :

\_merge=

"1" : valeurs de la clé sans observation dans le using file

"2" : valeurs de la clé sans obs. dans le master ; les virer !

"3" : valeurs de la clé présente à la fois dans master et using

Il y a d'autres informations possibles

append using "file.dta" // Empile les données du fichier file à celles

// qui sont déjà en mémoire

joinby et mmerge // (dans le cours)

edit // Ouvre la fenêtre des données en mode

// modifiable

browse // Idem, en mode non-modifiable

viewsource monfichier.do // Montre le .do file dans la fenêtre Viewer

which student // Dit si la commande "student" est built-in

order Y X // Place la variable Y avant la variable X

order Y, first // Place Y à la fin

order X, last // Place X à la fin

describe // Décrit la base de données, types des var.

describe, short // Idem, mais avec moins d'information

// Note : ces deux commandes sont utiles pour

// récupérer le nombre de variables r(k)

describe X Y // Ne décrit que ces deux variables.

// Note : on peut récupérer r(k), le nb. var.

// et r(N), le nb. d'observations

list in 1/10 // Affiche les dix premières observations

list Y if X>3 // Affiche les observations de Y pour X

// strictement supérieur à 3

drop Y X // Supprime les variables Y et X

keep Y // Ne garde que la variable Y

keep in 1/10 // Ne garde que les dix premières observations

drop in 1/10 // Supprime les dix premières observations

drop if \_n==\_N // supprime la dernière observation

drop if \_n==\_N-1 // supprime l'avant dernière observation

drop in -1 // Idem

drop if Y==4 // Supprime les observations : Y vaut 4

keep if Y!=4 // Ne garde que les observations qui sont

// telles que Y est différent de 4

collapse (sum) Y3 (max) Y2, by(Y1) // Réduit les données après transformations

Note : dans cet exemple, pour chaque valeur de Y1 (une par ligne),

les valeurs de Y3 son sommées, et on a le max de Y2

sort Y // Trie les obs. en fonction de Y croissant

sort Y X // Idem, puis pour X pour chaque valeur de Y

gsort -Y // Trie les obs. en fonction de Y décroissant

tsset ANNEE // Déclare des données temporelles (ANNEE est

// ma variable pour l'indice t)

xtset UNITE ANNEE // Déclare des données de panel (UNITE est ma

// variable pour l'indice i)

Note : tsset est nécessaire pour utiliser des lag, des lead ; xtset

permet la même chose, en plus de commandes pour données de panels

local I=1 // Attribuer 1 à la macro variable "locale" I

local I=`I'+1 // Le contenu de I est incrémenté de 1 (I=2)

local ++I // Idem

local L "hello world" // Attribuer une chaîne de caractères à L

display "`L'"

global J=1 // Attribuer 1 à la macro variable "globale" J

scalar K=1 // Attribuer 1 à la variable K

Note : différences entre local, global et scalar vues en cours

macro drop \_all // Efface les macros (sauf celles réservées)

macro list // Affiche les macro réservées

display runiform() // Valeur au hasard pour une variable uniforme

// continue de paramètres 0 et 1

display rbinomial(1,.5) // Valeur au hasard pour une variable de

// Bernoulli de paramètre 1/2

display rbinomial(5,.5) // Valeur au hasard pour une variable Binomiale

// de paramètres 5 et 1/2

display rpoisson(80) // Valeur au hasard pour une variable de

// Poisson de paramètre 80

display binomialp(5,3,.5) // Probabilité qu'une variable Binomiale de

// paramètres 5 et 1/2 soit égale à 3

display binomial(5,3,.5) // Probabilité qu'une variable Binomiale de

// paramètres 5 et 1/2 soit au plus égale à 3

set seed # // Initialisation personnelle du générateur

// de nombr. aléa. ; par ex., dans Stata 13

set seed 21041971 // les deux display suivants donnent toujours

display runiform() // .85157785 sur Stata 13

display runiform() // .67310157

generate Y=1 // Crée une variable qui vaut 1 pour chaque

// observation

replace Y=2\*Y // Remplace Y par 2 fois Y

replace Z=Y[1] in 3 // Remplace la valeur de Z à la ligne 3 par

// la première valeur de Y. Par exemple,

generate YL1=l1.Y // Chaque observation de YL1 a pour valeur

// celle de Y à la période précédente

// (opérateur retard d'ordre 1, tsset avant)

generate YD1=d1.Y // Chaque observation de YD1 a pour valeur Y

// moins le Y de la période précédente (YD1

// est la différence première de Y)

generate Y=rnormal(50,0.003) // Remplit les observations de Y par des

// valeurs au hasard d'une normale de moyenne

// 50, et d'écart-type 0,003

generate YNEW=recode(Y,a,b) // YNEW = a si Y<=a, sinon YNEW = b (b > a)

generate Y=(X==2015) // Y vaut 1 si X=2015, 0 sinon

rename (Y1 Y2\* Y3)=\_NEW // Les variables Y1, Y3 et toutes celles dont

// le nom commence par Y2 sont renommées avec

// le suffixe "\_NEW"

generate Y2=substr(Y1,a,b) // Extrait la sous-chaîne qui commence à la

// position a et fait b caractères de long

recode Y(1=2)(2=4) ... // 1 devient 2, 2 devient 4, et on ne touche

// pas aux autres valeurs

egen Y3=concat(Y1 Y2) // Concatène des chaînes de caractère. Exemple

clear all

set obs 10

generate VA="A"

generate VB="B"

egen VC=concat(VA VB)

order V\*

egen Y=count(X), by(Z T ...) // Comptages conditionnels. Par exemple

// rentrer ces valeurs dans le Data Editor

var1 var2

1 1

1 2

2 1

2 1

rename (\_all)(Y1 Y2)

egen Y3=count(Y2), by(Y1) // Nb. de lignes de Y2 po. chaq. valeur de Y1

egen Y4=count(Y2), by(Y1 Y2) // Nb. de lignes de Y2 po. chaq. couple (Y1,Y2)

egen Y5=sum(Y2), by(Y1 Y2) // Somme les valeurs de Y2 po. chaq. couple (Y1,Y2)

Y1 Y2 Y3 Y4 Y5

1 1 2 1 1

1 2 2 1 2

2 1 2 2 2

2 1 2 2 2

rename Y YNEW // Renomme Y en YNEW

label variable Y "Nom de ma variable" // Label qui sera affiché pour la variable Y

// dans les graphiques, tableaux, ...

label define YL 0 "Non" 1 "Oui" // A la place de la valeur numérique 0

// (resp. 1) on lira "Non" (resp. "Oui")

label values Y YL // Chaque valeur de Y, qui est une variable 0-1

// est replacée par les labels définis pour YL

// Note : label variable, define et values

// sont généralement utilisées ensemble

label drop YL // Supprime les labels

destring Y, replace // Convertit Y, qui contient des chaînes de

// caractères de chiffres (en rouge) en valeur

// Y ne doit pas contenir de lettres. Exemple,

clear all

set obs 3

input str1 Y1 str1 Y2

"A" "1"

"2" "2"

"3" "3"

browse

destring Y1, replace // Note : destring ne marche pas sur Y1

destring Y2, replace // car Y1 contient une lettre

destring Y, generate(YNEW) // Garde Y, Y déchaînée est dans YNEW

tostring Y, generate(YNEW) // Transformation inverse

// (quand les valeurs de Y sont numériques)

encode Y, generate(YNEW) // Rend la variable string Y accessible aux

// commandes Stata qui ne marchent qu'avec des

// valeurs numériques.

format Y %3.2f // Format d'affichage de Y dans la fenêtre data

// (pour reserrer les colonnes par exemple)

input Y // On entre soit même des valeurs pour une

// nouvelle variable Y. Exemple,

clear

set obs 4 // 4 observation

input int Y1 str8 Y2 str5 Y3 // 3 variables (des entiers, une chaîne de 8

1 "cannes" "06029" // caractères max. et de 5 caractères max.)

2 "nice" "06088"

3 "biot" "06018"

4 "antibes" "06004"

matrix V1=J(1,1000,0) // Crée un vecteur nul de dim. 1000 (en fait,

// une matrice 1x1000)

matrix define V1[1,3]=10 // Met la valeur 10 à la 3e colonne du vecteur

// V1

matrix V2=V1' // Le vecteur V2 est le transposé de V1

svmat V2, names(MEAN) // Transforme le vecteur V2 en une variable de

// 1000 observations, qui s'appelle MEAN

// Note : les données doivent avoir au

// moins 1000 obs. au départ (set obs

// 1000 au moins)

count if Y!=. // Nb. obs. de Y (numérique) non-manquantes

count if X!="" // Nb. obs. de X (chaînes) non-vides

count if Y>=.16&Y<=.18 // Compte combien Y a de valeurs entre .16 et

// .18 incluses

table Y // Différentes catégories ou valeurs de Y et

// leurs fréquences

tabulate Y // Fréquences, fréquences relatives, cumulées

tabstat Y, statistics(median) // Statistiques pour la variable numérique Y

// (il s'agit ici de la médiane)

// Note : pour le comptage/affichage des

// des valeurs manquantes, voir le cours

// (plus de détails dans le cours)

summarize Y // Statistiques descriptives de base pour Y

// (nb. observations, moyenne, écart-type

// corrigé, min et max)

// Note : on peut récupérer la moyenne,

// et autres statistiques. Exemple,

summarize Y

scalar MEAN=r(mean)

scalar ETYP=r(sd)

display ETYP/MEAN

summarize Y, detail // ... encore plus de statistiques

by X: summarize Y // Pour chaque valeur de X, résumer Y

while ... // Boucle conditionnelle. Par exemple :

clear all

set obs 100

local I=1

while `I'<=10 {

generate TEMP`I'=rnormal(0,1)

local ++I

}

graph matrix Y1 Y2 Y3 // Trace 3\*2 nuages de points (le nuage d'une

// variable avec elle-même n'est pas tracé)

// Note : plus généralement, avec K

// variables, on aura K\*(K-1) nuages

histogram Y // Histogramme de Y, une variable continue

histogram X, discrete // Histogramme de X, une variable discrète

// Note : consulter les sections 1 et 2

// du cours pour voir des options, et

// documentation Stata

histogram X, normal // Trace l'histogramme de X et affiche la

// densité d'une normale par-dessus

graph box Y // Boîte à moustache (voir la documentation,

// https://www.stata.com/manuals/...

// ...g-2graphbox.pdf

graph pie Y1 Y2 Y3 // Camembert (la feuille de données a une

// observation et trois variables)

stem Y // Diagramme tige-feuille

tsline Y // Série chronologique (une unité dans le

// temps)

xtline Y, i(UNITE) t(ANNEE)

// Séries chronologiques (plusieurs unités dans

// le temps dans des cadres différents)

xtline Y, i(UNITE) t(ANNEE) overlay

// Séries chronologiques (plusieurs unités dans

// le temps dans le même cadre)

twoway function y=x\*x // Trace la fonction f(x)=x\*x

twoway function y=normalden(x), range(-4 4)

// Trace la densité d'une normale dans

// l'intervalle -4<=x<=4

Brouillon à trier et insérer avant

- - - PONDERATIONS

tabstat Y2 [aweight=WEIGHT], statistics(mean sd)

Note: fournit des statistiques d'observations pondérées; aweight multiplie Y par la

variable de poids W, et divise par e'W (W contient la fréquence de chaque valeur de

Y dans la population). Si W entier, W utilisé pour pondérer la moyenne S\_iy\_i\*w\_i/e'W

Y1 Y2 WEIGHT

1 1 2

1 2 2

2 1 1

2 1 1

variable mean sd

Y2 1.333333 .5443311

- - - GRAPHIQUE

histogram Y

twoway (scatter Y X), ytitle(titley) xtitle(titlex) title(titlegraphics)

scatter Y X

scatter Y X || lfit Y X // Droite d'ajustement par dessus

scatter Y X || lfitci Y X // Idem, avec conf. inter. smoothé

Note: c'est équivalent à

graph twoway (scatter Y X)(lfit Y X)

graph twoway (scatter Y X)(lfit Y X)(qfitci YX, stdf), by(Z)

// quadratic fit with conf. interval

// stdf: CI on the basis of s.e. of forecast

\* Commandes Stata, Evens Salies, 19/10/2018 v1

?\* Note : dans ce qui suit - Les lignes d'exemples "#" est un nombre entier

\* - les noms des variables sont en lettres majuscule

// Blabla // Commentaire sur la ligne d'une commande

\* Blabla // Commentaire à la ligne

/// // Continuation de ligne. Par exemple :

summarize Y1 Y2 Y3 Y4 Y5 ///

Y6 Y7 Y8 Y9 // (summarize est expliqué plus loin)

#delimit ; ... #delimit cr // Même chose que ///. Par exemple :

clear all

set obs 10

forvalues STEP=1(1)30 {

generate Y`STEP'=`STEP'

}

\*

#delimit ;

summarize Y1 Y2 Y3 Y4 Y5 Y6 Y7 Y8 Y9 Y10 Y11 Y12 Y13 Y14 Y15 Y16 Y17 Y18

Y19 Y20 Y21 Y22 Y23 Y24 Y25 Y26 Y27 Y28 Y29 ;

summarize Y30 ; // Une commande doit se terminer avec ";"

#delimit cr

clear all // Supprime les données en mémoire

cls // Vide la fenêtre de résultats

set obs 100 // Fixe le nombre d'observations à 100

// Note : commande inutile quand on

// ouvre fichier de données

set more off // L'affichage des résultats dans la fenêtre

// de résultats n'est pas interrompu

cd "monchemin" // Chemin vers le dossier de travail. Par

// exemple "C:Users\Max\Documents"

use "nomdufichierstata.dta", clear // Charge données au format Stata. Option

use "nomdufichierstata.dta" // clear s'il y a déjà des données en mémoire

import excel using "C:\...nomduclasseurexcel.xls", firstrow

// Option "firstrow" si la première ligne du

// fichier Excel contient les nom des

// variables

// ".xlsx" aussi

insheet using "C:\...data.txt" // Idem, au format ASCII (.txt)

save "nouveaufichier.dta" // Sauve le fichier de données au format Stata

save "nouveaufichier.dta", replace // Idem, mais remplace l'ancien fichier

merge x:y CLE1 CLE2 ... // Fusionne des bases de données

// x et y valent 1 ou m, selon que les valeurs

// des clés apparaissent 1 ou m > 1 fois

Note: variable "\_merge" créée, contient des informations :

\_merge=

"1" : l'observation vient du master file

"2" : valeurs de la clé sans obs. dans le master ; les virer !

"3" : valeurs de la clé présented à la fois dans master et using

// (plus de détails dans le cours)

append using "file.dta" // Emile les données du fichier file à celles

// qui sont déjà en mémoire

edit // Ouvre la fenêtre des données en mode

// modifiable

browse // Idem, en mode non-modifiable

order Y X // Place la variable Y avant la variable X

order Y, first // Place Y à la fin

order X, last // Place X à la fin

describe // Décrit la base de données, types des var.

describe, short // Idem, mais avec moins d'information

// Note : ces deux commandes sont utiles pour

// récupérer le nombre de variables r(k)

describe X Y // Ne décrit que ces deux variables.

local K=r(k) // le nombre d'observations r(N)

local N=r(N) // (local est expliquée plus loin)

list in 1/10 // Affiche les dix premières observations

list Y if X>3 // Affiche les observations de Y pour X

// strictement supérieur à 3

drop Y X // Supprime les variables Y et X

keep Y // Ne garde que la variable Y

keep in 1/10 // Ne garde que les dix premières observations

drop in 1/10 // Supprime les dix premières observations

drop if \_n==\_N // supprime la dernière observation

drop if \_n==\_N-1 // supprime l'avant dernière observation

drop in -1 // Idem

drop if Y==4 // Supprime les observations : Y vaut 4

keep if Y!=4 // Ne garde que les observations qui sont

// telles que Y est différent de 4

collapse (sum) Y3 (max) Y2, by(Y1) // Réduit les données après transformations

Note: dans cet exemple, pour chaque valeur de Y1 (une par ligne),

les valeurs de Y3 son sommées, et on a le max de Y2

sort Y // Trie les obs. en fonction de Y croissant

sort Y X // Idem, puis pour X pour chaque valeur de Y

gsort -Y // Trie les obs. en fonction de Y décroissant

tsset ANNEE // Déclare des données temporelles (ANNEE est

// ma variable pour l'indice t)

xtset UNITE ANNEE // Déclare des données de panel (UNITE est ma

// variable pour l'indice i)

local I=1 // Attribuer 1 à la variable I

// I n'est pas dans la feuille de données

// "local" veut dire qu'elle est détruite

// dès qu'on sort du programme

local I=`I'+1 // Le contenu de I est incrémenté de 1 (I=2)

local ++I // Idem

global J=2 // Attribuer 1 à la variable J

// Contrairement à local, J n'est pas détruite

display 5\*`I' // Affiche 5 fois le contenu de I (10)

// Note : les trois lignes doivent être

// exécutées ensemble, car après exécution,

// I est vidée ("local" veut dire temporaire)

macro drop \_all // Efface les macros (sauf celles réservées)

macro list // Affiche les macro réservées

display runiform() // Valeur au hasard pour une variable uniforme

// continue de paramètres 0 et 1

display rbinomial(1,.5) // Valeur au hasard pour une variable de

// Bernoulli de paramètre 1/2

display rbinomial(5,.5) // Valeur au hasard pour une variable Binomiale

// de paramètres 5 et 1/2

display rpoisson(80) // Valeur au hasard pour une variable de

// Poisson de paramètre 80

display binomialp(5,3,.5) // Probabilité qu'une variable Binomiale de

// paramètres 5 et 1/2 soit égale à 3

display binomial(5,3,.5) // Probabilité qu'une variable Binomiale de

// paramètres 5 et 1/2 soit au plus égale à 3

set seed # // Initialisation personnelle du générateur

// de nombr. aléa. ; par ex., dans Stata 13

set seed 21041971 // les deux display suivants donnent toujours

display runiform() // .85157785

display runiform() // .67310157

generate Y=1 // Crée une variable qui vaut 1 pour chaque

// observation

replace Y=2\*Y // Remplace Y par 2 fois Y

replace Z=Y[1] in 3 // Remplace la valeur de Z à la ligne 3 par

// la première valeur de Y. Par exemple,

generate YL1=l1.Y // Chaque observation de YL1 a pour valeur

// celle de Y à la période précédente

// (opérateur retard d'ordre 1, tsset avant)

generate YD1=d1.Y // Chaque observation de YD1 a pour valeur Y

// moins le Y de la période précédente (YD1

// est la différence première de Y)

generate Y=rnormal(50,0.003) // Remplit les observations de Y par des

// valeurs au hasard d'une normale de moyenne

// 50, et d'écart-type 0,003

generate YNEW=recode(Y,a,b) // Distribution des fréquences de Y

// in (0,a], (a,b], >b

generate Y=(X==2015) // Y takes value 1 if X=2015, 0 otherwise

egen Y3=concat(Y1 Y2) // Y3=Y1|Y2. Y1,Y2 des chaînes de caractère

// (d'autres fonctions sur les châines)

egen Y3=count(Y2), by(Y1) // For each Y1 value, #Y2 lines->Y3

egen Y4=count(Y2), by(Y1 Y2) // For each Y1:Y2 val., #Y2 distinct val. -> Y3

egen Y5=sum(Y2), by(Y1 Y2) // For each Y1 value, Y2 values addition -> Y3

Y1 Y2 Y3 Y4 Y5

1 1 2 1 1

1 2 2 1 2

2 1 2 2 2

2 1 2 2 2

rename Y YNEW // Renomme Y en YNEW

label variable Y "Nom de ma variable" // Label qui sera affiché pour la variable Y

// dans les graphiques, tableaux, ...

label define YL 0 "Non" 1 "Oui" // A la place de la valeur numérique 0

// (resp. 1) on lira "Non" (resp. "Oui")

label values Y YL // Chaque valeur de Y, qui est une variable 0-1

// est replacée par les labels définis pour YL

// Note : label variable, define et values

// sont généralement utilisées ensemble

destring Y, replace // Convertit Y, qui contient des chaînes de

// caractères de chiffres (en rouge) en valeur

// Y ne doit pas contenir de lettres. Exemple,

clear all

set obs 3

input str1 Y1 str1 Y2

"A" "1"

"2" "2"

"3" "3"

browse

destring Y1, replace // Note : destring ne marche pas sur Y1

destring Y2, replace // car Y1 contient une lettre

destring Y, generate(YNEW) // Garde Y, Y déchaînée est dans YNEW

tostring Y, generate(YNEW) // Transformation inverse

// (quand les valeurs de Y sont numériques)

format Y %3.2f // Format d'affichage de Y dans la fenêtre data

// (pour reserrer les colonnes par exemple)

input Y // On entre soit même des valeurs pour une

// nouvelle variable Y. Exemple,

clear

set obs 4 // 4 observation

input int Y1 str8 Y2 str5 Y3 // 3 variables (des entiers, une chaîne de 8

1 "cannes" "06029" // caractères max. et de 5 caractères max.)

2 "nice" "06088"

3 "biot" "06018"

4 "antibes" "06004"

matrix V1=J(1,1000,0) // Crée un vecteur nul de dim. 1000 (en fait,

// une matrice 1x1000)

matrix define V1[1,3]=10 // Met la valeur 10 à la 3e colonne du vecteur

// V1

matrix V2=V1' // Le vecteur V2 est le transposé de V1

svmat V2, names(MEAN) // Transforme le vecteur V2 en une variable de

// 1000 observations, qui s'appelle MEAN

// Note : les données doivent avoir au

// moins 1000 obs. au départ (set obs

// 1000 au moins)

count if Y!=. // #observations de Y non-manquantes

count if X=="" // #observations de X vides

count if Y>=.16&Y<=.18 // Compte combien Y a de valeurs entre .16 et

// .18 incluses

table Y // Différentes catégories ou valeurs de Y et

// leurs fréquences

tabulate Y // Fréquences, fréquences relatives, cumulées

tabstat Y, statistics(median) // Statistiques pour la variable numérique Y

// (il s'agit ici de la médiane)

// Note : pour le comptage/affichage des

// des valeurs manquantes, voir le cours

// (plus de détails dans le cours)

summarize Y // Statistiques descriptives de base pour Y

// (nb. observations, moyenne, écart-type

// corrigé, min et max)

// Note : on peut récupérer la moyenne,

// et autres statistiques. Exemple,

summarize Y

scalar MEAN=r(mean)

scalar ETYP=r(sd)

display ETYP/MEAN

summarize Y, detail // ... encore plus de statistiques

while ... // Boucle conditionnelle. Par exemple :

clear all

set obs 100

local I=1

while `I'<=10 {

generate TEMP`I'=rnormal(0,1)

local ++I

}

graph matrix Y1 Y2 Y3 // Trace 3\*2 nuages de points (le nuage d'une

// variable avec elle-même n'est pas tracé)

// Note : plus généralement, avec K

// variables, on aura K\*(K-1) nuages

histogram Y // Histogramme de Y, une variable continue

histogram X, discrete // Histogramme de X, une variable discrète

// Note : consulter les sections 1 et 2

// du cours pour voir des options, et

// documentation Stata

histogram X, normal // Trace l'histogramme de X et affiche la

// densité d'une normale par-dessus

graph box Y // Boîte à moustache (voir la documentation,

// https://www.stata.com/manuals/...

// ...g-2graphbox.pdf

graph pie Y1 Y2 Y3 // Camembert (la feuille de données a une

// observation et trois variables)

stem Y // Diagramme tige-feuille

tsline Y // Série chronologique (une unité dans le

// temps)

xtline Y, i(UNITE) t(ANNEE)

// Séries chronologiques (plusieurs unités dans

// le temps dans des cadres différents)

xtline Y, i(UNITE) t(ANNEE) overlay

// Séries chronologiques (plusieurs unités dans

// le temps dans le même cadre)

twoway function y=x\*x // Trace la fonction f(x)=x\*x

twoway function y=normalden(x), range(-4 4)

// Trace la densité d'une normale dans

// l'intervalle -4<=x<=4

Le 1/1/1960 est le jour 0 pour Stata qui compte en jours dans le format de data que nous allons utiliser. Le 2/22/1960 est le jour 52, mais ca devient complique pour des dates qui sont des annees plus tard que le 1/1/1960. Par exemple, le 8/16/2016 est 20682. Or, (2016-1960)\*365 jours + [(2016-1960)/4+1 jours pour les annees bissextiles (29 jours en fevrier au lieu de 28)]+31+29+31+30+31+30+31+16 jours en 2016 = 20653, il me manque 29 jours.

On remarque que la commande

count if CREATION=="01jan1960" ne marche pas

Le bout de code suivant ajoute une variable CREATIONCODE qui permet de voir le nombre a utiliser dans les commandes

split F if \_n>1, generate(TEMP) parse(/)

rename (TEMP\*)(TEMPM TEMPJ TEMPA)

order F TEMPJ TEMPM TEMPA

destring TEMPJ, generate(YJ)

destring TEMPM, generate(YM)

destring TEMPA, generate(YA)

generate double CREATION=mdy(YM,YJ,YA) // Date de creation

format CREATION %td

label variable CREATION "Date de création de l'entreprise"

tostring CREATION, generate(TEMPC)

encode TEMPC, gen(CREATIONCODE)

drop F TEMP\* YJ YM YA

count if CREATION==0 // le 01jan1960 la valeur dans la colonne CREATIONCODE est 0

Note: on peut arriver au même résultat avec merge

si on considère Y1 et Y3 comme clés, donc faire

un using qui comporte Y1, Y3 et Y4

- - - EMPILER DES DONNEES

use "file1", clear

append using "file2"

- - - DECRIRE VARIABLES

describe

describe X Y

describe X-Y

describe X\*

- - - AFFICHER DES LIGNES

list in a/b

list Y X if (...)

- - - SUPPRIMER LIGNES OU COLONNES

drop Y X

drop if Y==a

drop in a/b

keep Y if ...

keep Y

collapse (sum) Y3 (max) Y2, by(Y1)

Note: new dataset has 1 line per Y1 value, with sum of Y3 values, max of Y2 values.

(min), (max), (sum), ...

- - - TABLEAU

table Y // Différentes catégories et leur fréquence

table Y X, [options]

Note: table Y Y fill a square matrix's diagonal elements with the different Y values

good tric to know the #categories for a categorical variable

tabulate Y // Comme table, et fréquences relatives et cumulées

tabulate YNEW X, cell nofreq // puts Pr instead of frequencies

tabulate Y Y, nofreq cell

tabstat Y, [options] // Comme tabulate mais ne reproduit pas les catégories

Note: l'écart type de tabstat est la racine de la variance sans biais

Note: tabstat est plus proche d'un summarize, nodetail

- - - COMMANDES STATISTIQUES

count if Y!=. // #observations non missing

count if Y1=="" // #observations blank

generate YNEW=recode(Y,a,b) // gives frequencies distibution of values

// in (0,a], (a,b], >b

- - - CREER VARIABLES QUI INCLUENT OU PAS DES STATISTIQUES

generate Y=(X==2015) // Y takes value 1 if X=2015, 0 otherwise

egen Y3=concat(Y1 Y2) // Y3=Y1|Y2. Y1,Y2 strings (if numeric, Y3 string anyway)

egen Y3=count(Y2), by(Y1) // For each value of Y1, #Y2 lines->Y3

egen Y4=count(Y2), by(Y1 Y2) // For each value of Y1:Y2, #Y2 distinct values->Y3

egen Y5=sum(Y2), by(Y1 Y2) // For each value of Y1, Y2 values addition->Y3

Y1 Y2 Y3 Y4 Y5

1 1 2 1 1

1 2 2 1 2

2 1 2 2 2

2 1 2 2 2

- - - PERMUTER OBSERVATIONS

\* by [need we sort the 'by' variable first]

sort STRA

by STRA: summarize Y

- - - PONDERATIONS

tabstat Y2 [aweight=WEIGHT], statistics(mean sd)

Note: fournit des statistiques d'observations pondérées; aweight multiplie Y par la

variable de poids W, et divise par e'W (W contient la fréquence de chaque valeur de

Y dans la population). Si W entier, W utilisé pour pondérer la moyenne S\_iy\_i\*w\_i/e'W

Y1 Y2 WEIGHT

1 1 2

1 2 2

2 1 1

2 1 1

variable mean sd

Y2 1.333333 .5443311

- - - GRAPHIQUE

histogram Y

twoway (scatter Y X), ytitle(titley) xtitle(titlex) title(titlegraphics)

scatter Y X

scatter Y X || lfit Y X // Droite d'ajustement par dessus

scatter Y X || lfitci Y X // Idem, avec conf. inter. smoothé

Note: c'est équivalent à

graph twoway (scatter Y X)(lfit Y X)

graph twoway (scatter Y X)(lfit Y X)(qfitci YX, stdf), by(Z)

// quadratic fit with conf. interval

// stdf: CI on the basis of s.e. of forecast