Stata-Support de formation

Mulekya Schadra

2022-06-20

Contents

About

Ce document est écrit comme support de formation dans le logiciel STATA. une formation rélisé pour le Docteur Franc Lutu.

Nous ne prétendons pas aborder toutes les connaissances disponibles dans STATA, neanmoins nou proposons les compétences éssentieles dans les aspects d'analyse des données.

Nous nous basons sur la comprehension progressive. ce sont les bases qui determinent la comprehension des notions suivantes. ‘

ce livre est téléchargeable en format pdf ,sur le compte Github ci-sessous https: //github.org/mulekya_schadra/.

‘Ce support est écrit dans le cadre d'apprentissage du logiciel STATA, dans ce livre, les chapitres sont organisées de manière à inculter une certaine compétence dans l'analyse des donnes Ce cours est reparti dans 5 chapitres, concernat les aspects de base du logiciel avec hierarchie

Chapter 1

Pries en main du logiciel

le logiciel est un programme de l'entreprise staa utilisé dans le domaine de l'économie et de l'économétrie dans le cadre d'analyse des données.

Ce logiciel est manipulable sous deux angles :

- Interface graphique ;
- Intgerface de commande

cette aspect des chose rend le logiciel Stata fléxible quand aux éxigences du moment: `la reproductibilité du travail dans l'analyse des données`

1.1 Présentation de l'interface

Voici comment resemble l'interface Stata à l'ouverture du programme:

4 fenetres principales dont :

- La visionneuse des resultats
- La partie Commande
- la Vue des variables
- L'historique des commandes exécutées.

(1) Le visionneuse des resultats sert à visualiser les résultats après exécution d'une quelconque tâche dans le cadre du travail sous Stata
(2) La partie Commande est la partie où on entre du code dans la syntax appropriée à stata, et selon la tâche que l'on souhaiterais éxécuter
(3) La partie vue des variables quand à elle, sert à montrer le nom des variables contenues dans la base des données et leurs caractéristiques tel que: le type des variables, leur format, les label. Les autres details sont affciché dans la fenêtre juste en bas: la partie propriété des variables
(4) La fênetre history quand à elle, sert montre l'ensemble des codes exécutées dans la sessions Stata dépuis le début du travail.

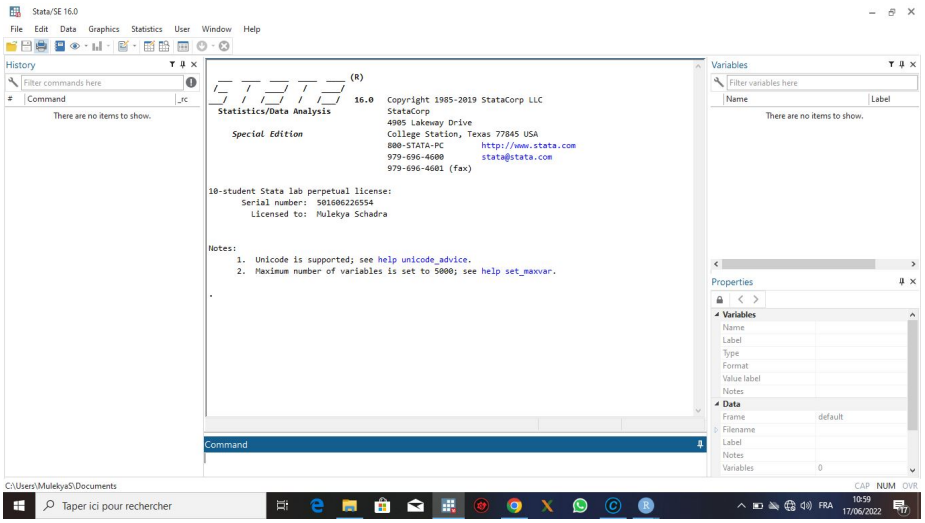

Figure 1.1: Presentation de l'interface Stata

à part ces interfaces, nous avons la base à outils et la bare des menus dans stata.

Avec exécution que ce soit par l'utilisation de l'intrerface graphique ou de la partie commande, tout travail passe par l'invité de commande.

Stata étant un logiciel dédié à l'analyse des données nous allons passer directement par la partie qui consiste à charger une base des données dans le mémoire Stata: ***Importation****.*

1.2 Definition de l'espace de travail

En anglais *working direcrorie* est le dossier de lecture et d'écriture d'un programme par défaut. Pour toute session de stata, le repertoire (directory) est les dossier *mes documents*. celui-ci est changé par la commande **cd** pour signifier *change directorie* et suivi du chemin d'accès complet à ce repertoire.

pour connaitre le chemin d'accès ç un repertoire donnée, il suffit de se selectionner ce dossier et de faire menu contextuelle tout en appuyant le bouton shift du clavier et choisir l'option *copier en tant que chemin d'accès* et ensuite coller dans stata après le mot clé *cd*.

```
# cd "E:\ MES CONSULTANCES\Dr Franck\Stata Learning" pour spécifier le
#  dossier *Stata learning comme repoirtoire de travail pour la
# session stata. Changer les *\* en */*.
```

```
Exemlple: cd "E:/MES CONSULTANCES/Dr Franck/Stata Learning" pour
spécifier le dossier *Stata learning comme repoirtoire de travail
```

`pour la session stata.`

1.3 fenetres aditionnelles de Stata

(1) Data editor / Data browser pour visualiser les données chargée dans la mémoire sous forme de tableau de manière à faciliter leurs lectures comme s'il s'agissait d'un tableaur (excel par exemple). Data bowser est différent de data editor dans ce ses que ce premier permet de visualiser les données sans possibilité de modification, tandis que la le data editor quand à lui offre des possibilité de modification comme dans un tableur classique.
(2) Graph editor : pour visualiser les graphiques tracés dans stata, en permettant une certaine modification des élèments graphiques tel que le titre, le titre des axes, la vouleur des textes , ...
(3) Variable manager qui permmet de visualiser et même des modofier les propriétes des variables contenue dans la base des données stata
(4) help : pour voir l'aide sur différentes opérations sous stata

Ainsi donc, les points 1,2 et 4 font parti des fenêtres du type `viewer` dans `stata`.

1.4 importation de la base des données

Stata offre plusieures possibilitées de lire les bases des données provenant de plusieures sources externes sont *excel, spss, sas,csv, ...* l'extention des bases des données propre à stata sont les fichiers *.dta*.

pour importer une base des données sous stata, il faut utiliser la fonctiuon reab avec l'extention du fichier.

(1) pour un fichier excel: read_excel avec le chemin d'accès complet du fichier, écrit sous forme de caractète .
(2) pour un fichier spss, la comande a comme mot clé *read_spss* ainsi de suite

Dans le cadre de ce cours nous allons plus utiliser les fichiers venant de l'excel. Ainsi donc, nous exploiterons plus la fonction *read_excel* et les différents arguments qui viennent avec. Specifier la feuille qui contient nos données spécifier les noms des variables à la première ligne ou pas spécifier si toutes les données sont importées comme des chaines des caractères ou pas.

1.5 Fichier do (do-file)

Le dofile est est dichier dans lequel sont stockés les différentes commandes des stata, que l'on pourra exécuter plus tard , au besoin, pour des raisons de continuité et de reproductibilité du travail d'analyse des données.

Notons que k'on peut choisir d'utiliser stata par son interface graphique que par sa partie commande.

Chapter 2

Data CLeaning

2.1 Introduction

Pourquoi manipuler les données en Stata et pas en Excel ? La raison est simple : pas mal des commandes que l'on va voir ci-dessous existent aussi en Excel et sont certes quelquefois plus simples (si on arrive à les trouver), mais par contre on perd vite le fil de ce que l'on a fait subir aux données avant de passer à l'estimation, et c'est parfois là que se cachent soit les quelques erreurs à l'origine de résultats grotesques soit, au contraire, les mauvais traitements infligés aux chiffres pour obtenir le résultat désiré.

Avec Stata, on peut garder la trace de toutes les manipulations dans le do-file. Celui-ci doit contenir toutes les commandes permettant de passer du fichier-données brut à celui qui est prêt à l'estimation. Il est alors facile de retrouver l'erreur qui tue ou bien de vérifier ce que les chiffres ont subi entre les mains du bourreau avant d'avouer.

La manipulation des données sous stata consiste à

- Typage des variable
- remplaceme nt des valeurs manquantes
- remplacement de certaines variables sous certaines conditions
- codification des variable
- recodage des variables

Ainsi, dans uune base des donénes, avant de commencer le nettoyage de la base des donnée sil surffit d'avoir une vue gloobale sur cette base en conaissant les caractéristiques générales de diffémetes variables contenues dans la base des données. Ainsi, nous utilisons les commandes suivantes :

(1) **describe** : permet de décrire toutes les variables de la base des données chargée en mémoire. il nous amène en sortie: le nombre des observation,

le nombre des variables, les noms des variables , les labeks et les types de chaque variable sous forme de tableau. //Avec les options *short* pour affichier le nom des variables, *simple* pour affciher le nombre des variables et le nombre d'observations dans la BD.

(2) **codebook** pour voir les différentes ecaractéristiques des variables dans la base des données. utiliser codebook suivi du nom de la variable pour ne voir que les caractéristiques d'une seule variable ou une liste des variables.

(3) Visualisation de la BD sous forme de tableau
   - *browse* pour affciher uniquement;
   - *edit* pour pouvoir modifier manuellement les valeurs dans la base.

2.2 Les commandes de base

2.2.1 La synthaxe des commandes stata

Stata comme tous les logiciels, utilise un langage qui n'est ni de l'anglais, ni du français, mais son propre langage. Certes, les mots sont empruntés à la langue de Shakespeare, mais la syntaxe n'a rien à voir avec l'anglais. Hormis quelques exceptions, la syntaxe des commandes de Stata est:

```
[by listevar:] commande [listevar] [=exp] [if exp] [in intervalle]
[pondération] [, options]
```

Le nom de la commande est évidemment obligatoire, et il peut éventuellement être précédé d'un préfixe by, et le plus souvent il est suivi d'un ou de plusieurs suffixes. Dans le cas de commandes particulièrement usuelles, il peut parfois être abrégé, comme par exemple d pour describe. Les suffixes sont entourés de crochets pour indiquer leur caractère optionnel: listevar correspond à une liste de variables, exp à une expression logique, intervalle à une série d'observations dans le fichier de données, et pondéra t i on à une expression indiquant la variable et le mode de pondération des données. Enfin, après une virgule, on peut ajouter une ou plusieurs options pour l'exécution de la commande. La syntaxe complète pour chaque commande figure dans les manuels de référence de Stata, qui restent de ce point de vue irremplaçables. Mais puisque le préfixe by et les suffixes if, in et la pondération sont communs à la majorité des commandes, nous nous en tiendrons dans les chapitres suivants à exposer la syntaxe de base qui prend la forme: .

```
commande [listevar] [=exp] [, options]
```

Immédiatement après le nom de la commande, une liste de variables indique sur quelles variables doit s'effectuer la commande. Pour explorer le fichier « census.dta », on tapera:

```
list state region pop
```

(a) Le préfixe **by** permet d'exécuter la commande pour chaque sous-ensemble d'observations défini pour chaque valeur de listevar. Avant d'exécuter

la commande, le fichier doit d'abord être trié (avec la commande **sort** listevar) selon la même variable utilisée par le préfixe by. Par exemple, on aura:

(b) Le suffixe *[in intervalle]* Le suffixe in est moins courant dans la pratique, car il suppose de bien connaître l'ordre dans lequel sont classées les observations du fichier. TI permet d'exécuter la commande pour certaines observations, par exemple:

```
# sort region
# by region: list region state pop medage
```

(b) Le suffixe *[if exp]* Le suffixe if restreint l'exécution de la commande au sousensemble des observations pour lesquelles l'expression logique exp est vraie, c'est-à-dire différente de la valeur O. Nous reviendrons dans la section consacrée aux calculs sur la manipulation de ces expressions logiques, dites encore booléennes. Pour l'heure, un exemple suffit à comprendre le fonctionnement de ce suffixe:

```
On préférera toujours sélectionner un sous-ensemble d'observation
avec le suffixe if en fonction de variables bien connues et
qui font sens, plutôt que de se fier à un ordre arbitraire des
observations dans le fichier.
```

2.2.2 Les commandes de depart

(1) **import** : charger la base des donnes dans la mémoire. Suivi de type des fichier. et le chemin d'accès du fichier
(2) **clear** vide la mémoire
(3) **use** au lieu de mettre tout le sentier. Ne pas oublier de mettre les guillemets comme ils sont (noter le sens !).
(4) **save** La commande `save datafile1.dta` est très importante : elle sauvegarde le fichier-données *(.dta)* modifié par le programme sous un autre nom que le fichier initial, ce qui permet de laisser ce dernier intouché. Sinon on altère le fichier initial de façon permanente, ce qui est en général un désastre. - De façon générale, les guillemets (comme dans cd "c:/path/directory") sont obligatoires quand les noms spécifiés ne sont pas liés en un seul mot ; par exemple, Stata comprend use "le nom que je veux.dta" mais pas use le nom que je veux.dta.
(5) **Describe** pour decrire la base des données

2.2.3 Creation et correction des variables

(1) La commande *rename* La commande rename (abrégée en ren) permet de changer le nom de la variable qui suit. Sa syntax est `rename ancien_non nouveau_nom`

(2) Les commandes *generate* et *replace* La commande generate crée de nou-

velles variables. Elle a la syntaxe de base suivante: *[by listevar:] generate var = exp[if exp] [in intervalle]* La commande replace utilise la même syntaxe, sauf qu'elle s'applique aux variables déjà existantes.

Comme on le voit, cette syntaxe est simple, ce qui n'est pas le cas de la forme que peut prendre exp. La première expression exp (après le signe =) spécifie le contenu de la variable, c'est-àdire le plus souvent une valeur numérique. La seconde expression exp (après *if*) doit être formulée comme une expression logique dont le résultat est soit vrai soit faux: la création (ou le remplacement) de la variable est restreint aux observations pour lesquelles le résultat de l'expression est vrai. Cela n'a l'air de rien, mais la confusion entre les deux expressions est certainement l'erreur la plus fréquente que peuvent faire les utilisateurs de Stata.

(3) Les commandes *tostring* et *destring* ces commandes permettent de modifier le types des variables en string ou de modifier les string ne contenanr que des caractères numériques en variables numériques. la commande _tostring*_ permet de mettre tous les caractètes de la base des données en chaines des caractères (strings) avec une ligne des codes.

(4) Les opérateurs

Les opérateurs arithmétiques de Stata sont bien classiques: + (addition), - (soustraction), * (multiplication), 1 (division), A (puissance), tout comme les opérateurs relationnels > (supérieur), < (inférieur), >= (supérieur ou égal), <= (inférieur ou égal).

C'est peut-être moins le cas des opérateurs relationnels == (égal) ou -= (différent, que l'on peut écrire aussi! =), et des opérateurs logique &. (et), 1 (ou bien), et - (non).

En effet, Stata distingue le signe = (affectation d'une valeur) du signe == (égalité entre deux valeurs). Dans le cas d'une affectation d'une valeur à une variable, la variable apparaît à gauche du signe = tandis que la valeur affectée apparaît à droite:

(3) Les expressions logiques dans R

Les expressions logiques sont particulièrement utiles pour créer des variables dichotomiques, c'est-à-dire qui ne prennent que deux valeurs, 0 et 1. En effet, une expression logique, c'est-àdire une expression où interviennent les opérateurs relationnels >, <, >=, <=, ==, -=, !=, ou bien les opérateurs logiques &., 1, et -, est codée 1 lorsque son résultat est vrai, et codée 0 lorsque son résultat est faux.

La commande tabulate possède une option generate ( ) bien pratique pour créer une série de variables dichotomiques à partir d'une variable polytomique. Exécutez la série de commandes:

(4) Gestions des dates et Formatage des variables

(5) **La commande drop et la commande keep**

Pour travailler sur une base de données pratique en vue des objectifs que vous avez, il sera peut-être néecessaire de supprimer les variables inutiles ou les observations non concernées par vos estimations. La variable keep vous permet de garder et drop de jeter… facile, non ? On les utilise alternativement selon le nombre de variables µa garder ou µa jeter.

```
keep age salaire pays marital  drop age15 salred salaire150 fdsrt azerty
```

(6) Definir les labels des variables et des valeurs Avec les observations et les commandes logiques, il est possible de préciser ce que l'on veut effacer en le conditionnant µa la valeur d'autres variables. Par exemple, on garde les plus de 15 ans :

`keep if age>=15` ou bien on supprime les individus nés en 1945 et 1968 :

```
drop if naissance==1915 | naissance==1968
```

(5) Les commandes sort et by La commande sort (abrégée en so) classe les données par ordre croissant. Il est possible de préciser les variables selon lesquelles le classement peut ^etre e®ectué :

```
sort sexe age
```

Cette commande va classer les observations par sexe (d'abord les femmes en numéro 0 et puis les hommes en numéro 1, par exemple) puis au sein de chaque sexe par age (les femmes et les enfants d'abord). On peut utiliser la commande gsort pour e®ectuer des classements dans des ordres croissant ou décroissant. Un + ou un - vient donner le sens du classement au sein de chaque variable.

```
sort sexe -age
```

Cela classe d'abord par sexe puis par ^age décroissant (les femmes et les vieux d'abord). Le processus by … : qui doit suivre obligatoirement un classement avec sort permet d'uti- liser la plupart des commandes pour chaque valeur de la variable indiquée par by. Les exemples suivants vont vous aider µa comprendre le principe :

2.2.4 Combiner différentes bases de données : append et merge

Pour travailler de fa»con e±cace, il faut souvent réunir différentes bases de données. Selon le type de combination, on va utiliser une commande différennte.

(1) Ajouter des observations Si vous disposer par exemple de données sur l'emploi dans di®érents pays et que vous avez une base de donnée par pays avec les m^emes variables (emploi, salaire, temps de travail…), alors vous souhaitez ajouter des observations (rajouter des lignes). Votre premier soin est de créer une variable pays dans chaque base de donnée en indiquant

pour toutes les observations de ce pays le m^eme nom ou code. Ensuite vous pouvez utiliser la commande append de la fa»con suivante :

(2) Ajouter des variables Si vous souhaitez ajouter des variables, alors il faudra utiliser la com- mande merge. Par exemple, vous avez deux bases de données sur entreprise (les m^emes entreprises) et l'une donne des informations sur la production et l'autre sur les salariés. Si vous voulez calculer la productivité de ces entreprises, il faudra combiner ces deux bases. La procédure est légèrement plus complexe qu'avec *append*. Etant donné que certaines variables sont communes aux deux bases (au moins l'identifiant des entreprises), il faut classer ces variables avec sort dans les deux bases pour permettre au logiciel de faire la bonne fusion.

La commande merge crée la variable merge qui permet de vérfier que la fusion a été réalisée comme voulu. Elle peut prendre trois valeurs : - Les observations de la base principale n'ont pas été retrouvées dans la base ajoutée (celle aprµes using) - Les observations de la base ajoutée n'ont pas été retrouvées dans la base principale - Les observations dans les deux bases ont été retrouvées et connectées. Il faut toujours véri¯er que l'opération s'est bien déroulée en regardant si merge prend des valeurs di®érentes de 3. Si ce n'est pas le cas alors regardez pour quelles observations l'opération n'a pas fonctionné.

Chapter 3

Analyse Univarié, Bivariée et Graphiques

Avant de mener des analyses à l'aide de modèle de régression et autres statistiques complexes, il est préférable de tirer le maximum de l'exploration des données et de statistiques simples. Cela a deux avantages:

- permettre de mieux connaître les données et donc de repérer leurs particularités et leurs éventuelles incohérences, ce qui pourra servir pour des analyses statistiques plus approfondies;
- permettre de sélectionner des indices et des graphiques simples qui rendent le mieux compte des données afin de les restituer à un large public: les connaissances en statistique de la plupart des mortels ne dépassent guère le pourcentage, et de toute façon, même un public de spécialistes ne retiendra en définitive que les indices et les graphiques les plus simples.

Stata offre de nombreuses commandes pour l'analyse exploratoire des données, autant sous forme de tableaux que de graphiques. Comme dans les chapitres précédents, nous utiliserons le fichier « census.dta » pour illustrer ces commandes.

La commande *codebook* permet de faire le tri à plat de la base des données en montrant les sttatisques simples et univariées. Et montre toute les informations nécéssaires à la compréhention de la structure d'une variable.

La commande *summarize* listvar permet aussi de résumer la distribution, en particulier pour les variables numériques continues. Cela n'aurait pas grand sens, par exemple, de calculer la moyenne d'une variable discrète.

L'option detail pennet une description plus precIse des variables continues, incluant les pourcentiles, les quatre plus grandes (Largest) et plus basses (Smallest) valeurs, ainsi qu'un indice de dissymétrie (la valeur de Skewness est °pour la

distribution nonnale) et de concentration (la valeur de Kurtosis est de 3 pour la distribution normale.

À l'inverse de la commande swmnarize, la commande tabulate est utile pour les variables discrètes.

On remarque avec l'option nolabel (pour afficher les codes plutôt que les libellés), que les régions sont classées selon leur numéro de code:

3.1 Tableaux croisés à deux variables

La commande tabulate devient vraiment intéressante pour croiser les distributions de deux variables discrètes. La syntaxe de base de cette commande est:

```
tabulate varligne varcol[, cell column row missing nofreq wrap
nolabel ~ll chi2 exact gamma lrchi2 iaub v]
```

Les modalités de la première variable citée figurent en ligne, tandis que les modalités de la deuxième apparaissent en colonne. Des options permettent d'obtenir les pourcentages en ligne (row) , en colonne (column) ou par cellule (cell) du tableau:

Pour afficher les pourcentages sans les fréquences, on utilisera l'option nofreq :

3.1.1 Tableaux croisés à trois variables ou plus

Chapter 4

La Modélisation avec Stata

4.1 Théorie d'Estimation

4.2 Regtession lineaire

4.3 regression Logistique

4.3.1 Regresson logistique binoliale

4.3.2 Regression Logistique Multinomiale

4.4 Regression Logistique Ordonné

Chapter 5

Analyse des données de Logitudinales

5.1 Introduction aux series temporelles

5.2 Données de Panel

5.3 Analyse de Survie

Chapter 6

Analyses Exploratioires

6.1 ACP

6.2 AFC

6.3 ACM