Nº 12-002-XIF au catalogue

Le Bulletin technique et d'information des Centres de données de recherche

Printemps 2004, vol.1 nº 1

Statistique Canada Statistics Canada

Canada

Comment obtenir d'autres renseignements

Toute demande de renseignements au sujet du présent produit ou au sujet de statistiques ou de services connexes doit être adressée à : Le programme des centres de données de recherche, Statistique Canada, Ottawa, Ontario, K1A 0T6 (téléphone : 1 800 263-1136).

Pour obtenir des renseignements sur l'ensemble des données de Statistique Canada qui sont disponibles, veuillez composer l'un des numéros sans frais suivants. Vous pouvez également communiquer avec nous par courriel ou visiter notre site Web.

Service national de renseignements	**1 800 263-1136**
Service national d'appareils de télécommunications pour les malentendants	**1 800 363-7629**
Renseignements concernant le Programme des bibliothèques de dépôt	**1 800 700-1033**
Télécopieur pour le Progamme des bibliothèques de dépôt	**1 800 889-9734**
Renseignements par courriel	**infostats@statcan.ca**
Site Web	**www.statcan.ca**

Renseignements sur les commandes et les abonnements

Le produit nº 12-002-XIF au catalogue est gratuit sur Internet. Les utilisateurs sont priés de se rendre à http://www.statcan.ca/cgi-bin/downpub/freepub_f.cgi

Normes de service à la clientèle

Statistique Canada s'engage à fournir à ses clients des services rapides, fiables et courtois, et ce, dans la langue officielle de leur choix. À cet égard, notre organisme s'est doté de normes de service à la clientèle qui doivent être observées par les employés lorsqu'ils offrent des services à la clientèle. Pour obtenir une copie de ces normes de service, veuillez communiquer avec Statistique Canada au numéro sans frais 1 800 263-1136.

Statistique Canada
Le programme des centres de données de recherche

Le Bulletin technique et d'information des Centres de données de recherche

Printemps 2004, vol.1 nº 1

Publication autorisée par le ministre responsable de Statistique Canada

Avril 2004

Nº 12-002-XIF au catalogue

Périodicité: semestriel

ISSN : 1710-2200

Ottawa

This publication is available in English (Catalogue no. 12-002-XIE)

Note de reconnaissance

Le succès du système statistique du Canada repose sur un partenariat bien établi entre Statistique Canada et la population, les entreprises, les administrations canadiennes et les autres organismes. Sans cette collaboration et cette bonne volonté, il serait impossible de produire des statistiques précises et actuelles.

À propos du Bulletin Technique et d'Information

Le Bulletin technique et d'information est un forum permettant aux utilisateurs actuels et prospectifs des CDR de partager de l'information et les techniques d'analyse des données disponibles dans les centres. Le bulletin paraîtra au printemps et à l'automne, et l'on publiera à l'occasion des numéros spéciaux sur des questions d'actualité.

Objectifs

Les objectifs principaux de ce bulletin sont les suivants :

- l'accroissement et la diffusion de la connaissance concernant les données de Statistique Canada;
- les échanges d'idées parmi les utilisateurs membres des CDR;
- l'aide aux nouveaux utilisateurs du programme CDR; et
- offrir des occasions supplémentaires permettant aux chercheurs dans les centres de communiquer avec les spécialistes et divisions spécialisées au sein de Statistique Canada.

Le contenu

Nous souhaitons publier des articles qui contribueront à accroître la qualité des travaux de recherche menés dans les Centres de données de recherche de Statistique Canada et qui fourniront des conseils méthodologiques aux chercheurs travaillant dans les CDR.

Les articles figurant dans le Bulletin technique et d'information portent principalement sur :

- l'analyse et la modélisation des données;
- la gestion des données;
- les pratiques statistiques, informatiques ou scientifiques éprouvées ou au contraire inefficaces;
- le contenu en données;
- les effets associés au libellé des questionnaires;
- la comparaison d'ensembles de données;
- l'examen des méthodes et de leur application;
- les particularités que présentent les données;
- les problèmes associés aux données et leurs solutions; et
- les outils innovateurs faisant appel aux enquêtes des Centres de données de recherche (CDR) et aux logiciels pertinents.

Ceux et celles qui s'intéressent à soumettre un article au Bulletin d'Information et Technique sont priés de suivre les directives pour les auteurs.

Les rédacteurs et les auteurs tiennent à remercier les réviseurs de leurs commentaires précieux.

Rédacteur: James Chowhan
Rédacteurs adjoints: Heather Hobson, Leslie-Anne Keown, Darren Lauzon

Table des matières

Construction de modèles d'analyse de survie incluant des variables indépendantes qui changent dans le temps: application fondée sur les données de l'Enquête sur la dynamique du travail et du revenu

Denis Gonthier

Résumé

Cet article fournit un exemple pratique d'élaboration de modèle d'analyse de survie. On traitera d'abord de l'outil informatique utilisé, soit le logiciel SAS. Il sera ensuite question de la construction d'un fichier longitudinal et des défis que cela peut poser. Une attention particulière est portée aux variables explicatives qui n'ont pas une valeur constante au fil du temps. Afin d'illustrer l'approche suivie, un exemple d'application pratique sera présenté. Il s'agit d'une analyse fondée sur les données de l'Enquête sur la dynamique du travail et du revenu (EDTR), dont on utilise le premier panel observé de janvier 1993 à décembre 1998. On tirera partie de l'information de cette enquête disponible sous forme de vecteurs pour élaborer un modèle semi-paramétrique de Cox. Dans cette section, un exemple de programme informatique sera commenté. On précisera également la façon dont le logiciel traite les variables principales. Enfin, il y aura une brève description des résultats d'un modèle relativement simple.

Introduction

Nous allons présenter dans ce texte une application pratique de l'analyse de survie en tirant partie de l'information disponible sous forme de vecteurs dans une enquête longitudinale de Statistique Canada. Il s'agit donc principalement d'une description technique, qui peut être utile aux chercheurs ayant à traiter des variables indépendantes changeant dans le temps. Il est à noter qu'on ne traitera pas de la question de l'effet de plan dans les enquêtes complexes, qui pose d'ailleurs un défi particulier dans le contexte des méthodes d'analyse de survie. Dans l'exemple de modèle présenté, on se limitera à utiliser un poids normé.

Dans un premier temps, une brève introduction décrira l'outil informatique retenu, soit le logiciel SAS et la procédure «PROC PHREG». Puis, il sera question de l'approche à utiliser lorsque les variables explicatives n'ont pas une valeur constante au fil du temps. Enfin, on présentera un exemple d'application fondé sur les données de l'Enquête sur la dynamique du travail et du revenu (EDTR), dont on utilise le premier panel observé de janvier 1993 à décembre 1998.

II – L'analyse de survie et la procédure SAS PHREG

Dans la construction d'un fichier d'analyse longitudinale où il y a des informations telles qu'un vecteur hebdomadaire de statut d'activité sur le marché du travail, il peut s'avérer utile de recourir à un logiciel comme SAS. Ce dernier possède en effet certaines fonctions itératives qui facilitent la gestion d'un tel type de données, comme on le verra dans le texte qui suit. Par ailleurs, on retrouve dans SAS des procédures permettant de construire des modèles à risques proportionels.

On aura recours ici au modèle semi-paramétrique de Cox. Un des avantages du modèle de Cox tient au fait qu'il n'est pas nécessaire de spécifier la forme de la fonction de survie sous-jacente, comme c'est le cas pour d'autres méthodes telles que le modèle à sorties accélérées («accelerated failure time model»). La procédure SAS correspondant à ce modèle se nomme «PHREG». On doit préciser que PHREG ne permet pas d'effectuer une estimation de la variance fondée sur le plan d'enquête. Le logiciel SUDAAN permettrait de recourir aux poids «bootstrap» produits par Statistique Canada afin de tenir compte de l'effet de plan dans l'analyse de survie, mais il ne peut pas intégrer des variables indépendantes qui changent dans le temps. Comme on traite ici de ce type de variables, et qu'elles apparaissent sous forme de vecteurs, le choix retenu est donc d'utiliser SAS.

En ce qui concerne la configuration des données, il y a relativement peu de contraintes avec PHREG. Ceci n'est pas le cas pour tous les logiciels d'analyse de survie. Mentionnons que les variables comportant des catégories de réponses doivent être redéfinies en créant une série de variables dichotomiques (la catégorie de référence sera omise dans le modèle).

La procédure PHREG permet notamment de développer des modèles simples où les variables indépendantes ne changent pas dans le temps. C'est le cas de covariables comme la langue maternelle ou le sexe. Il suffit alors de définir la variable dépendante de durée, de définir le statut de fin d'épisode (il se produit un événement ou une censure), et d'intégrer au modèle les variables indépendantes fixes.

L'utilisation de la procédure s'avère plus complexe quand les variables explicatives n'ont pas une valeur constante au fil du temps. Il en sera question dans la section qui suit.

III – L'utilisation de variables indépendantes qui changent dans le temps

Pour intégrer dans la procédure PHREG des variables qui changent dans le temps, il est possible de se servir d'éléments de programmation comme les opérateurs «IF, THEN» ainsi que l'instruction ARRAY qui définit des vecteurs. Ces éléments de programmation sont souvent utilisés à l'intérieur de l'étape DATA du logiciel SAS, mais il est aussi possible de s'en servir à l'intérieur de la procédure PHREG.[1]

On peut utiliser la valeur de la variable de durée pour définir une variable indépendante qui change dans le temps. Par exemple, on peut programmer PHREG pour que la variable explicative X prenne la valeur «0» tant que la durée écoulée est inférieure à deux ans (l'unité de mesure du temps d'analyse étant l'année), puis prenne la valeur «1» quand la durée atteint ou dépasse deux ans.
Si la période d'analyse est découpée en un grand nombre d'intervalles (par exemple si on mesure une variable explicative à chaque mois, pendant plusieurs mois), il est utile de recourir aux

[1] Voir SAS OnlineDOC, V8 - SAS/STAT - SAS/STAT User's Guide – The PHREG Procedure – Syntax – Programming Statements.

vecteurs pour identifier l'élément à utiliser selon la valeur atteinte pour la variable dépendante de durée. Il s'agit alors de prendre l'instruction ARRAY qui permet de définir de tels vecteurs. Nous en donnerons un exemple dans la section qui suit.

Cette façon de procéder est rendu possible par le fait que la procédure PHREG recalcule à tout instant les valeurs des variables explicatives, qui peuvent changer selon la durée écoulée.

IV – Exemple fondé sur l'Enquête sur la dynamique du travail et du revenu

Nous ajusterons ici le modèle semi-paramétrique de Cox aux données de l'Enquête sur la dynamique du travail et du revenu (EDTR), afin de mesurer l'effet de certaines variables explicatives sur le risque de changer de province.[2] La migration interprovinciale constitue donc l'événement observé dans le modèle. Il serait possible de recourir à une régression logistique pour modéliser la migration, mais la régression de Cox a été choisie parce qu'on dispose dans l'EDTR d'une datation des événements. On peut savoir quand un mouvement se produit, mais aussi connaître la valeur des variables indépendantes au moment où il se produit (avec une précision plus ou moins grande).

On retiendra seulement les épisodes de séjour ayant débuté durant les six années d'observation du panel 1 de l'EDTR. Compte tenu du critère qui précède, il n'y a donc pas d'épisodes «ouverts à gauche».[3] L'échantillon se compose des répondants de 16 à 69 ans ayant déménagé dans une autre province entre janvier 1993 et décembre 1998.

Le modèle présenté comprend des variables indépendantes qui changent dans le temps, mais dont la précision chronologique diffère. En effet, le statut d'activité (occupé, chômeur ou inactif) est mesuré de façon hebdomadaire. Le statut d'étudiant à temps plein (ou non) est mesuré mensuellement. Le statut de locataire/propriétaire est mesuré annuellement. La valeur des variables indépendantes n'est pas toujours connue au moment où se produit la migration (laquelle est datée avec précision). On se référera donc à la situation antérieure à l'événement, en prenant celle qui est la plus près dans le temps.[4]

La variable dépendante du modèle de Cox est la durée de l'épisode de séjour. Elle est calculée à partir de la date de début d'épisode, et on a choisi de la mesurer en années (incluant des fractions d'année car les événements sont connus au jour près). Cette variable se nomme DUREE, et elle correspond au temps écoulé entre le début et la fin de l'épisode. Comme tous les épisodes ont débuté durant les six années de l'enquête, il s'ensuit que la variable DUREE prendra une valeur située entre zéro et six années. La variable STATUT indique si l'épisode se termine parce

[2] L'exemple présenté ici se rapporte à un projet de recherche en cours, mené en collaboration avec Jacques Ledent, professeur-chercheur à l'Institut national de la recherche scientifique.

[3] Il s'agirait du cas où un épisode a débuté avant le commencement de l'enquête, mais où on ne sait pas quand il a débuté. Le temps écoulé avant la période d'observation serait alors inconnu.

[4] Il peut être nécessaire de prévoir un effet différé des variables indépendantes, c'est-à-dire un certain laps de temps entre l'apparition d'une condition donnée et l'effet qu'elle peut produire. Nous n'inclurons pas un tel effet différé dans le modèle retenu. Voir le livre de Paul Allison intitulé «Survival Analysis Using the SAS® System : A Practical Guide», qui aborde cette question dans le chapitre consacré à la régression de Cox.

que l'événement à l'étude se produit (migration) ou parce qu'il y a censure. La censure peut être causée par le fait que l'enquête prend fin sans que l'événement ne soit survenu.[5]

On utilise la procédure ARRAY afin de définir des vecteurs de variables indépendantes. Ainsi, on crée un vecteur ETUD comprenant 72 variables dichotomiques de statut mensuel d'étude à temps plein, pendant la période de six ans retenue. Pour déterminer quel élément du vecteur sera utilisé, on se sert comme indice de référence de la variable MOIS, dont la valeur change selon le temps écoulé depuis le début de l'enquête. Ce temps est défini par la somme de ECART et DUREE, où ECART est le temps écoulé (en années) entre le début d'enquête et le début de l'épisode.[6] Comme l'enquête débute le 1er janvier 1993, la variable ECART vaudra 3,5 ans si un épisode débute le 1er juillet 1996. Il est à noter que l'enquête enregistre la date exacte du dernier déménagement.

La variable MOIS change de valeur lorsque le modèle de COX réévalue la valeur de la variable dépendante DUREE (alors que ECART a une valeur fixe). Et le modèle ira chercher la nième variable appropriée dans le vecteur ETUD. On prend la même approche pour la série de variables sur le statut de locataire/propriétaire, qui comporte six valeurs annuelles. C'est le cas aussi pour la série de variables sur le statut d'activité, où il y a 53 mesures par année, ce qui représente plus de 300 valeurs hebdomadaires pour l'ensemble de la fenêtre d'observation. Le statut d'activité comporte trois catégories. Ainsi, dans le modèle de Cox on retiendra les catégories OCCUPE et CHOMEUR, et le groupe de référence sera formé des inactifs. Voici un exemple de programme SAS utilisant la procédure PHREG avec des variables indépendantes qui changent dans le temps. On se sert de cinq variables indépendantes : une est continue (l'âge au début de l'épisode) et les quatre autres sont dichotomiques. La mesure en années du temps écoulé depuis le début de l'enquête (ECART+DUREE) est transformée en une autre unité de temps lorsque nécessaire, soit en mois ou en semaines.

```
01  proc phreg data=test;
02    model duree*statut(1)= age proprietaire etude occupe chomeur;
03    freq poids / notruncate;
04    array prop{6} prop1-prop6;
05      annee=ceil(ecart+duree);      /*  ANNEE est un nombre entier variant entre 1 et 6  */
06      proprietaire=prop{annee};
07    array etud{72} etudes1-etudes72;
08      mois=ceil(12*(ecart+duree));   /* MOIS est un nombre entier variant entre 1 et 72  */
09      etude=etud{mois};
10    array occup{318} occup1-occup318;
11    array chom{318} chomeur1-chomeur318;
12      semaine=ceil(53*(ecart+duree)); /*  SEMAINE est un nombre entier variant entre 1 et 318  */
13      occupe=occup{semaine};
14      chomeur=chom{semaine};
15  run;
```

[5] Il est possible qu'il y ait un nouvel épisode, dans le cas où un répondant migre à nouveau (la population à l'étude étant constituée de ceux qui ont migré une fois durant l'enquête).

[6] Dans l'exemple actuel, on ne retient pas les épisodes ayant commencé avant le début de l'enquête.

- La ligne 1 appelle la procédure PHREG et le fichier de données.
- La ligne 2 définit le modèle, où DUREE est la variable dépendante (durée de l'épisode en années) et où STATUT indique s'il y a eu migration ou censure à la fin de l'épisode (il y a censure quand STATUT=1).
- À la ligne 3, on utilise l'instruction FREQ pour pondérer les données. La variable «POIDS» contient un poids qui a déjà été ramené à une moyenne de un pour la population étudiée (poids normé). Le recours à la pondération permet de tenir compte des probabilités inégales de sélection des répondants de l'EDTR. L'option NOTRUNCATE permet de conserver les décimales.
- La ligne 4 définit un vecteur de statut de propriété, qui change annuellement et dont on a six valeurs (car l'enquête suit les répondants durant six ans).
- Pour choisir durant laquelle des six années d'observation se produit l'événement, on calcule un indice à la ligne 5 (variable ANNEE). Cette variable est un nombre entier (obtenu par l'opérateur CEIL qui arrondit au nombre supérieur).
- On se sert ensuite de cet indice à la ligne 6 pour sélectionner l'une des six valeurs du statut de propriété (propriétaire ou non).
- La même logique s'applique aux lignes 7 à 9. On y définit un vecteur de statut d'études à temps plein (variable dichotomique). On ramène la mesure du temps en mois en multipliant par 12 la durée totale en années (ligne 8).
- On répète le même type de programmation aux lignes 10 à 14, en multipliant la durée en années par 53 pour ramener la mesure du temps en semaines (ligne 12).

Afin d'illustrer comment PHREG calcule la valeur des variables indépendantes qui changent dans le temps, nous présenterons le tableau suivant qui décrit pour deux observations fictives ce qui se passe quand la compilation de la variable dépendante (DUREE) atteint une valeur de 3.0 années :

	DUREE	DUREE + ECART [7]	MOIS	ETUDE	ANNEE	PROPRIETAIRE
Cas 1	3.0	3.5	42	Valeur=ETUDE42	4	Valeur=PROP4
Cas 2	3.0	4.5	54	Valeur=ETUDE54	5	Valeur=PROP5

Lors de la sélection des caractéristiques de la population à risque pour une durée exacte de trois ans, les valeurs des variables indépendantes sont celles qui correspondent au calendrier propre à chaque répondant. La prise en compte de DUREE et ECART permet d'obtenir les valeurs des mois où les répondants atteignent une durée d'épisode de trois ans : le 42^{e} mois pour le cas 1, et le 54^{e} mois pour le cas 2. Dans le cas d'une variable dont la valeur est connue seulement au mois près (comme le statut d'étudiant), on peut envisager de prendre la situation correspondant au mois précédent la migration, pour s'assurer d'avoir une caractéristique mesurée avant que l'événement ne se produise.

[7] Le cas 1 a débuté son épisode le 1er juillet 1993 (ECART=0,5), et le cas 2 a débuté son épisode le 1er juillet 1994 (ECART=1,5).

Et voici le résultat donné par la procédure PHREG :

The PHREG Procedure

Model Information

Data Set	WORK.TEST
Dependent Variable	duree
Censoring Variable	statut
Censoring Value(s)	1
Frequency Variable	poids
Ties Handling	BRESLOW

Summary of the Number of Event and Censored Values

Total	Event	Censored	Percent Censored
1417.0	221.7	1195.3	84.35

Convergence Status

Convergence criterion (GCONV=1E-8) satisfied.

Model Fit Statistics

Criterion	Without Covariates	With Covariates
-2 LOG L	2878.360	2783.734
AIC	2878.360	2793.734
SBC	2878.360	2810.741

Testing Global Null Hypothesis: BETA=0

Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	94.6255	5	<.0001
Score	96.2342	5	<.0001
Wald	90.9854	5	<.0001

The PHREG Procedure

Analysis of Maximum Likelihood Estimates

Variable	DF	Parameter Estimate	Standard Error	Chi-Square	Pr > ChiSq	Hazard Ratio
age	1	-0.06097	0.00936	42.4367	<.0001	0.941
proprietaire	1	-0.07004	0.14619	0.2295	0.6319	0.932
etude	1	-0.02101	0.23297	0.0081	0.9281	0.979
occupe	1	-0.18221	0.19365	0.8854	0.3467	0.833
chomeur	1	0.95681	0.23281	16.8912	<.0001	2.603

Dans cette sortie SAS, on voit qu'il y a 1417 observations, constituées d'épisodes de séjour pour les répondants ayant déjà vécu une migration durant l'enquête. Et il y a 222 événements, qui sont des nouvelles migrations. Le risque de changer de province (pour ceux qui ont déjà connu une migration de 1993 à 1998) est présenté au bas du tableau, dans la colonne intitulée «Hazard ratio». On voit par exemple que chaque année d'âge supplémentaire diminue le risque de migrer de près de 5%, et que cette mesure est très significative (seuil de signification inférieur à 0,0001). On note aussi que les chômeurs de cette sous-population de migrants récents risquent beaucoup plus de changer de province (rapport de risque de 2,6), relativement à la catégorie de référence qui est celle des inactifs. Là encore le résultat est très significatif. Ce n'est pas le cas des autres variables indépendantes du modèle. On doit rappeler ici que le modèle utilisé ne tient pas compte de l'effet de plan. Les résultats doivent donc être interprétés avec prudence. Également, il n'y a pas ici d'examen de l'impact des mouvements impliquant les membres d'un même ménage. Cet élément devrait être considéré dans le cadre d'une étude exhaustive sur le sujet.

V. En bref

En conclusion, il est possible de se servir d'un maximum d'information de nature longitudinale dans une enquête comme l'EDTR, par le recours à des variables dont les échelles chronologiques ne sont pas identiques. Cela élargit les possibilités offertes par une technique comme l'analyse de survie, dans un contexte où des données d'enquête n'ont pas toutes une datation exacte.

Production de fichiers de données « efficients » à l'aide du programme Stat/Transfer

Leslie-Anne Keown

Résumé

Les ensembles de données de grande taille posent plusieurs défis aux chercheurs, surtout les moins chevronnés. L'une des tâches les plus longues et les plus frustrantes pour les chercheurs qui en sont à leurs premières armes et qui n'ont pas encore l'expérience des grands ensembles de données consiste à élaguer ou à décomposer ces derniers de manière à prendre en compte uniquement les variables et l'échantillon pertinents. La production d'un fichier de données « efficient » peut contribuer à la performance du matériel et des logiciels et atténuer la frustration ressentie par le chercheur. Nous présentons ici une procédure permettant de produire un fichier des données efficient par suite d'une telle décomposition, à l'aide d'un programme appelé Stat/Transfer.

Introduction

Les ensembles de données que l'on retrouve dans les CDR sont souvent de grande taille, tandis que l'ensemble de variables dont ont besoin les chercheurs à leur égard est plus réduit. Le chercheur chevronné estimera souvent utile de produire un sous-ensemble de données contenant uniquement les variables pertinentes pour l'analyse qu'il souhaite faire, et il disposera à cette fin d'un certain nombre de méthodes mises au point au fil de ses travaux. Par contre, il arrivera qu'un chercheur moins expérimenté ne connaisse pas de logiciel particulier permettant d'accomplir cette tâche aisément, et il risque de ressentir de la frustration au cours de l'analyse. Cela se produit fréquemment parmi les utilisateurs des CDR, en particulier les étudiants fraîchement diplômés et des assistants à la recherche.

Outre la production d'un fichier plus compact, le chercheur pourrait aussi vouloir circonscrire son travail à une partie donnée de l'échantillon utilisé pour une enquête particulière, par exemple se cantonner aux personnes âgées de plus de 18 ans. Il peut aussi souhaiter transférer le fichier de données d'un programme statistique particulier à un autre (p. ex., conversion d'un fichier de format SPSS en vue de l'utiliser dans STATA). Selon le logiciel utilisé, la production d'un ensemble des données efficient sous l'angle à la fois de sa taille et de l'éventail de données incluses aux fins d'analyse est une tâche qui peut s'avérer à la fois longue et frustrante. Nous proposons ici un moyen de créer un ensemble de données plus compact, peu importe le logiciel qui servira à l'analyse. La méthode en question repose sur l'utilisation du programme Stat/Transfer. Bien que les exemples présentés aient trait à la version 6.1 de Stat/Transfer, la méthode devrait donner les mêmes résultats, peu importe la version utilisée.

II. Mise en garde

Il y a un certain nombre de points qui ne seront pas abordés ici, de manière que notre article soit relativement facile à suivre et puisse aider les chercheurs les moins expérimentés à

procéder à leurs travaux d'analyse de la façon la plus rapide et la plus efficiente possible. L'utilisation des fichiers de commande disponibles dans Stat/Transfer et les possibilités d'optimisation sont deux aspects à propos desquels le lecteur pourrait juger bon d'obtenir un complément d'information.

Le programme Stat/Transfer comporte un langage de programmation qui permet à l'utilisateur, au moyen de la syntaxe applicable, d'exécuter différentes procédures, dont celles traitées ici. Bien que nous n'abordions pas la chose dans notre article, le chercheur pourrait consulter le manuel du programme Stat/Transfer pour obtenir cette information une fois qu'il se sera familiarisé avec les procédures décrites ci-après. L'utilisation de la syntaxe appropriée ou des fichiers de commande peut s'avérer très utile afin de tenir un registre des activités menées et de reproduire des procédures avec les mêmes ensembles de données ou des ensembles de données différents.

Le chercheur doit se pencher sur l'utilisation des options d'optimisation avant de tenter de transférer des fichiers. Stat/Transfer procède à une optimisation dans le but de produire le fichier de données le plus petit possible. À cette fin, le programme utilise différentes catégories de variables afin de préserver l'intégrité des données d'origine. Dans certains cas, surtout lorsque le format des données est transformé, des problèmes peuvent survenir. Par exemple, les noms de variables peuvent être tronqués, ou des variables d'une mauvaise catégorie peuvent être produites (le problème habituel à cet égard se situe entre « float » et « double »). Il faut donc vérifier les résultats de chaque transfert avec le logiciel utilisé pour l'analyse, de manière à s'assurer que les variables ont conservé leur structure et que le transfert n'a causé aucune altération des fichiers de données.

III. Exemple pratique et méthode

La procédure de base consiste à charger l'ensemble de données de grande taille, non encore soumis à une analyse, dans Stat/Transfer, puis à choisir un nouveau nom pour le fichier analysé, et enfin à sélectionner à la fois les variables et le sous-échantillon pertinents. Nous allons présenter un exemple de cette procédure à l'aide du fichier de microdonnées à grande diffusion du treizième cycle de l'ESG.

Le treizième cycle porte sur la victimisation et les risques auxquels est exposée une personne. Il y a le fichier principal et un fichier incident. Le présent exemple se limite au fichier principal. Le treizième cycle de l'ESG portait entre autres sujets sur les perceptions et les préoccupations touchant la sécurité personnelle, les victimes d'actes criminels (questions de sélection), la violence psychologique et l'exploitation financière exercées par un partenaire, la violence exercée dans un mariage ou une union de fait, la violence psychologique et l'exploitation financière exercées à l'endroit de personnes âgées par des enfants et des soignants, et différentes variables démographiques et variables de classification. Le fichier principal comporte plus de 600 variables. En outre, l'échantillon a été élargi en prévision du treizième cycle et comprend 25 876 observations. La population cible est constituée par l'ensemble des Canadiennes et des Canadiens ayant plus de 15 ans, à l'exclusion des résidents des territoires et des pensionnaires à

plein temps d'établissements (Statistique Canada 2000a, Statistique Canada 2000b, Statistique Canada 2000c, Statistique Canada 2001).

Supposons que l'on veuille analyser les perceptions relatives au système de justice pénale, en particulier celles des personnes de sexe masculin ayant plus de 18 ans. Nous faisons l'hypothèse que les perceptions touchant la criminalité et le risque, la victimisation passée et les variables démographiques représentent des facteurs importants qui devront être pris en compte dans l'analyse. Par conséquent, l'examen de la documentation nous permet d'établir que nous aurons besoin des variables suivantes :

a) Identificateur exclusif – RECID
b) Coefficient de pondération – WGHT_PER
c) Perceptions du système de justice pénale – A10a TO A10e, A11a to A11d, A13a to A13b, A14a, A14b
d) Perceptions relatives à la criminalité et à la sécurité – A1 to A9
e) Victimisation au cours des 12 derniers mois – MSVIC, MSVIC_X, MSPER, MSPER_X, TOTVIC, TOTVIC_X
f) Variables démographiques – AGEC, SEX, MARSTAT, PRV, URIND, VISMIN, EDU10, NAICS16, INCM, INCMHSD.

Outre la création d'un sous-échantillon pertinent, nous allons limiter l'analyse aux personnes âgées de plus de 17 ans (AGEC>17) et de sexe masculin (SEX=1). Enfin, un format SPSS sera choisi pour l'ensemble de données, étant donné que l'analyse sera effectuée avec ce logiciel système; dans la pratique, on choisira le format correspondant au logiciel d'analyse employé par l'utilisateur.

IV. Préparation du transfert

La première étape consiste à lancer Stat/Transfer et à nommer à la fois le fichier de départ (fichier complet) et le fichier que nous voulons sauvegarder (fichier élagué). Nous inscrivons cette information sous l'onglet Transfer, qui constitue généralement le premier écran affiché par le programme. Il faut ensuite choisir l'option « Input File Type » pour le fichier que nous voulons élaguer (le fichier de microdonnées complet); dans notre exemple, le format est celui d'un fichier de données SPSS. Puis, vis-à-vis la mention « File Specification », nous inscrivons le nom du fichier et le chemin du répertoire où il est enregistré; cela est facile à l'aide de la fonction d'exploration.

Nous suivons la même procédure pour le fichier à créer. Nous choisissons d'abord le format du fichier (ici encore, SPSS), puis l'endroit où sera enregistré le nouvel ensemble de données. La figure 1 présente un écran Stat/Transfer dans lequel les noms des fichiers de données ont été inscrits.

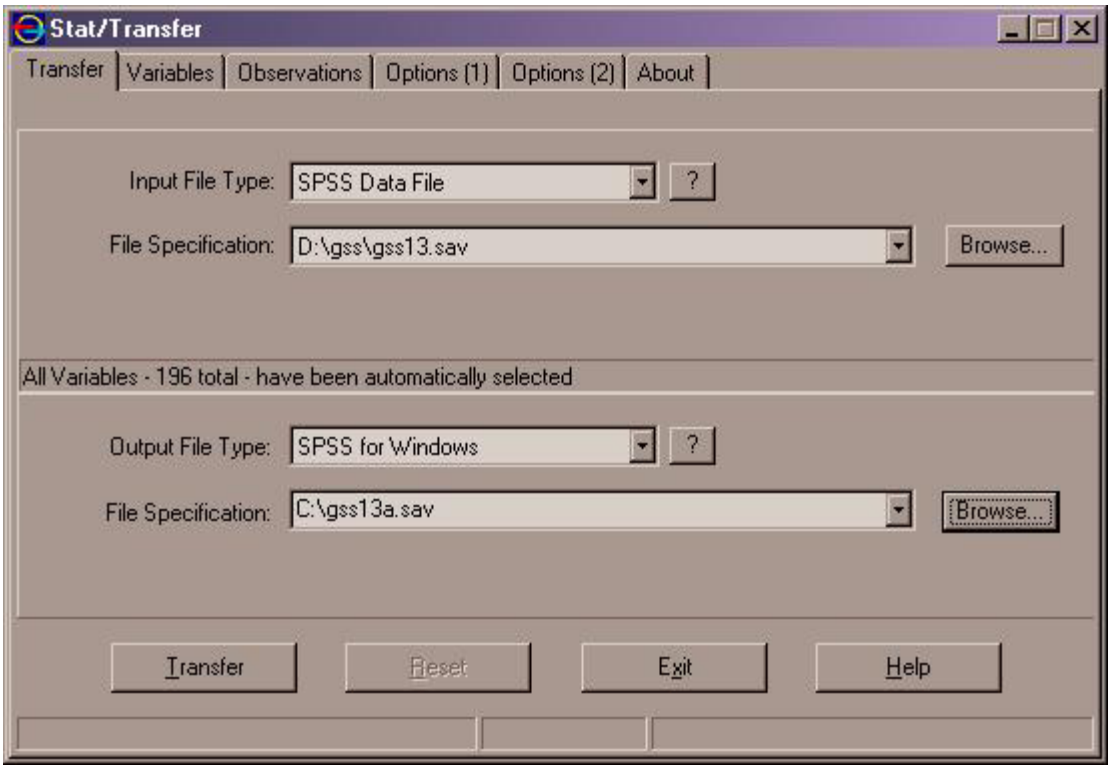


Figure 1

V. Sélection des variables

Nous choisissons ensuite les variables que nous voulons utiliser, ce qui suppose un certain nombre de courtes étapes. D'abord, il faut passer à l'onglet Variables, puis choisir l'option « Unselect all ». De cette manière, aucune variable n'est présélectionnée dans la liste de gauche, et nous pouvons cocher uniquement celle que nous voulons. Une fois cette étape accomplie, l'écran affiché devrait être similaire à la figure 2.

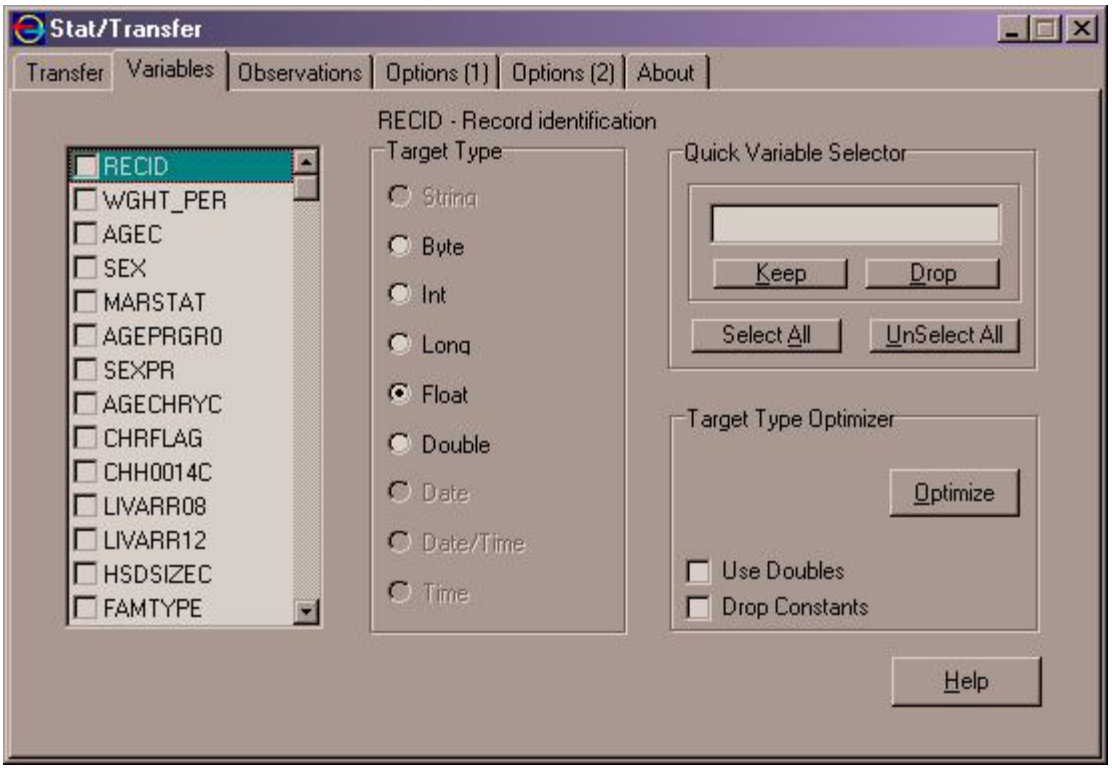


Figure 2

Nous pouvons maintenant sélectionner les variables pertinentes. Il est possible de subdiviser sommairement les variables en deux groupes, selon qu'elles ont un nom unique (p. ex., RECID) ou que leur nom contient des éléments communs (p. ex., A10a à A10e).

On peut recourir à deux méthodes pour les variables qui ont un nom particulier. Il est possible de choisir les variables en cochant la case vis-à-vis leur nom dans la partie gauche de l'écran, ou encore en tapant leur nom dans la boîte apparaissant sous « Quick Variable Selector » dans la partie droite de l'écran. Ces deux méthodes peuvent être utilisées pour sélectionner des variables appartenant aux catégories suivantes parmi celles proposées précédemment : identificateurs exclusifs, coefficients de pondération et variables démographiques.

La procédure que nous venons de décrire serait fastidieuse pour les autres variables de notre exemple (perceptions relatives au système de justice pénale, à la criminalité et à la sécurité, victimisation au cours des douze derniers mois). Les noms de ces variables présentent des similitudes qui peuvent permettre de les sélectionner beaucoup plus rapidement. Pour ce faire, il suffit d'inscrire dans la boîte sous « Quick Variable Selector » la partie commune du nom des variables, suivie d'un caractère générique, c'est-à-dire un symbole pouvant être substitué à des caractères quelconques et permettant d'isoler des mots clés. Ainsi, pour choisir toutes les variables dont le nom comme par A10, il suffit de taper « A10* », puis d'appuyer sur la touche Keep. Si nous tapons « A1* », les variables A10 à A19 seront sélectionnées; nous pouvons ensuite supprimer les variables A19 en tapant « A19* », puis en cliquant sur Drop. Le caractère générique correspondant à un seul caractère est « ? »; donc, pour choisir les variables A1 à A9, nous taperons

« A? ». Pour choisir les variables relatives à la victimisation, nous taperons « MS* » et « TOTV* ».

VI. Limitation de l'échantillon

Après avoir choisi les variables pertinentes, nous devons maintenant établir le sous-échantillon à analyser. La première chose à faire est de choisir l'onglet Observations. Les critères d'inclusion sont tapés dans la fenêtre de dialogue. Dans l'exemple de l'ESG, il faut taper « where AGEC > 17 & SEX =1 ». L'écran affiché devrait correspondre à celui de la figure 3.

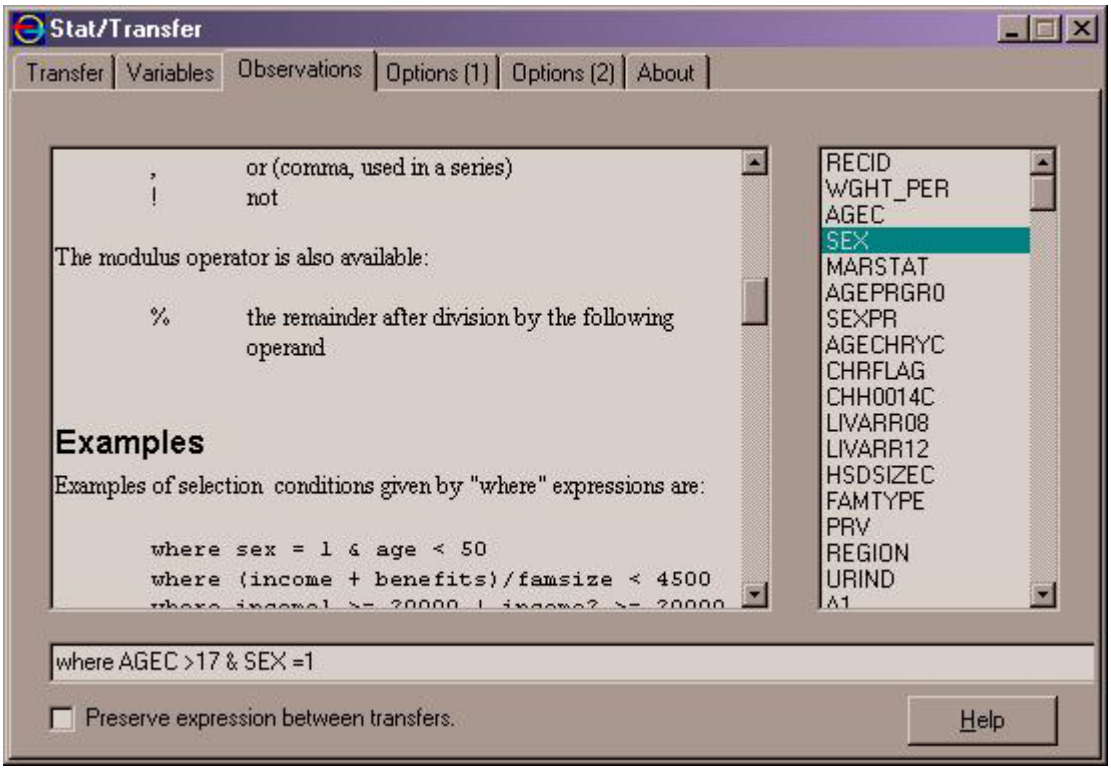


Figure 3

VII. Conclusion

Tout ce qui reste à faire maintenant est d'effectuer le transfert et de vérifier les résultats. Pour procéder au transfert, nous cliquons sur l'onglet Transfer, et nous vérifions que les noms de fichiers inscrits à cet écran n'ont pas été modifiés. Nous pouvons aussi vérifier le nombre de variables à transférer en consultant la note figurant entre la partie supérieure (Input File Type) et la partie inférieure (Output File Type); dans notre exemple, il est indiqué que 34 des 196 variables ont été choisies.

Après avoir vérifié que le bon nombre de variables est indiqué, nous cliquons sur le bouton Transfer. L'analyse du fichier sera effectuée, et le nouveau fichier sera sauvegardé dans le

répertoire indiqué vis-à-vis la mention «File Specification sous Output File Type». Une fois l'opération terminée, le mot « finished » sera affiché au bas de l'écran, et le nombre d'observations sera indiqué. Du fait que nous avons choisi un sous-échantillon, ce nombre devrait être inférieur au nombre total d'observations de l'échantillon d'origine (il y avait 25 876 observations dans l'échantillon d'origine; dans notre nouveau fichier, il y en a 11 088, ainsi que cela est confirmé au bas de la figure 4).

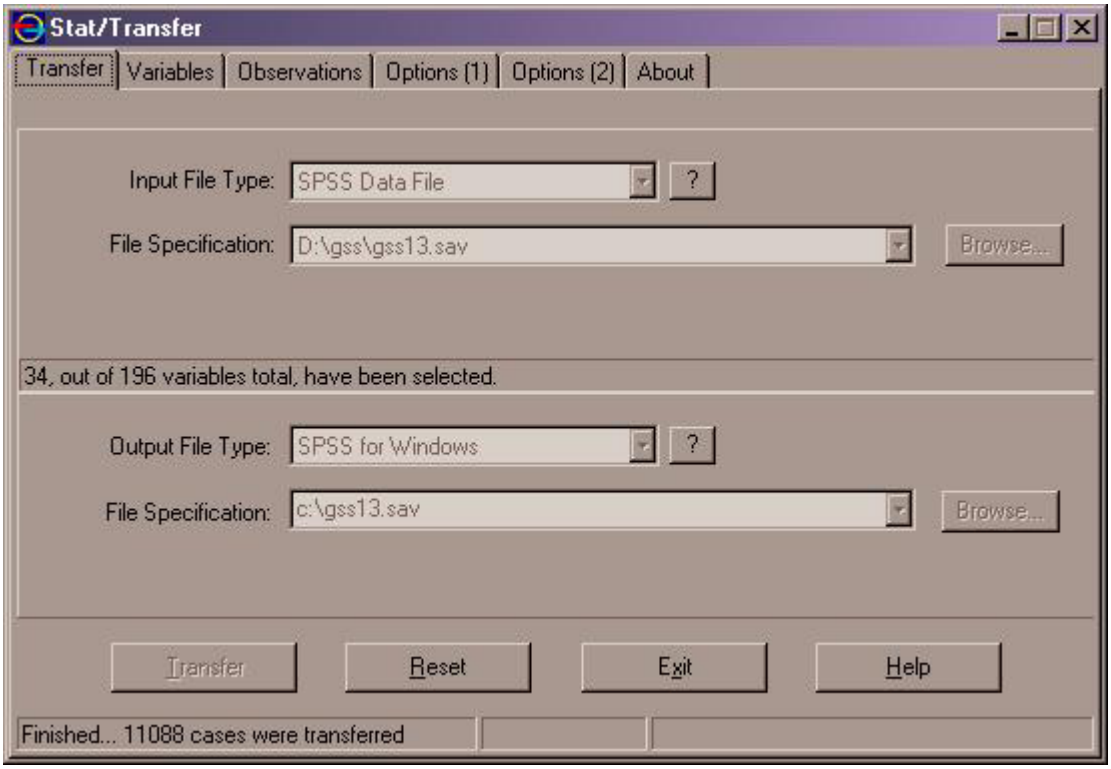


Figure 4

Finalement, pour vérifier que le fichier a été analysé correctement, il faut l'ouvrir dans le logiciel statistique de notre choix et procéder à des opérations exploratoires (fréquences et tableaux croisés) afin de vérifier que les totaux sont exacts et que les données relatives aux variables ont été transférées correctement. Le fichier élagué peut ensuite être utilisé à des fins d'analyse; la création de ce fichier devrait accroître l'efficacité à la fois du matériel informatique et des logiciels utilisés.

Bibliographie

Statistics Canada. 2000a. "The 1999 General Social Survey - Cycle 13 Victimization: Public use microdata file documentation and user's guide." Statistics Canada, Ottawa.

—. 2000b. "The 1999 General Social Survey - Cycle 13 Victimization :Questionnaire Package." Statistics Canada, Ottawa.

—. 2000c. "General Social Survey 1999 - Cycle 13 Victimization [public use microdata file]." Ottawa: Statistics Canada Housing Family and Social Statistics Division.

—. 2001. "A profile of Canadian Victimization: results of the 1999 General Social Survey." Statistics Canada, Ottawa.

Pour une utilisation plus conviviale de la méthode bootstrap : fichier ADO dans Stata

Emmanuelle Piérard, Neil Buckley et James Chowhan

Résumé

Cet article expose une commande pouvant être utilisée dans Stata pour calculer des variances estimatives à l'aide de coefficients de pondération bootstrap. La commande bswreg est compatible avec un large éventail de techniques d'analyse de régression et d'ensembles de données. Le programme a fait l'objet d'essais et de comparaisons avec les techniques d'analyse de régression offertes dans bootvare_v20.sas, de manière à en vérifier l'exactitude. Des données du quatrième cycle de l'ENSP sont utilisées pour ces comparaisons. Le programme constitue pour les chercheurs un outil simple et adaptable dont ils ne disposaient pas jusque là.

Introduction

Cet article expose une commande pouvant être utilisée dans Stata pour calculer des variances estimatives à l'aide de coefficients de pondération bootstrap. Ce programme a été créé de manière à doter Stata d'une fonction à la fois simple et souple pouvant être utilisée de concert avec les coefficients de pondération bootstrap dont on dispose avec la plupart des ensembles de microdonnées de Statistique Canada. À l'aide de ces coefficients de pondération bootstrap, les chercheurs peuvent utiliser des informations relatives au plan d'enquête complexe et calculer des variances estimatives fiables, tout en s'assurant que les renseignements fournis par les répondants demeurent confidentiels [Yeo et coll., 1999].

Le programme est compatible avec un large éventail de techniques de régression. Il se fonde sur les régressions linéaires et logistiques incorporées à Bootvar dans le but d'inclure différentes techniques de régression[1]. Nous traiterons ici des caractéristiques du programme, de ses points forts et de ses points faibles, puis nous décrirons un test simple permettant de vérifier la précision de ce nouveau programme de Stata comparativement à BOOTVARE_V20.SAS.

II. Méthode bootstrap standard

La plupart des enquêtes de Statistique Canada reposent sur un plan complexe aux fins d'établir un échantillon représentatif de la population cible. Les ensembles de microdonnées ainsi obtenus sont assortis de coefficients de pondération bootstrap, qui peuvent être utilisés de manière à tenir compte de la complexité du plan d'enquête. Les chercheurs peuvent utiliser ces coefficients de pondération bootstrap pour calculer des variances estimatives fiables. On utilise dans le

[1] Le programme Bootvar est disponible en format SAS et SPSS. Il est constitué d'une macro-commande servant à calculer des variances pour des totaux, des rapports, des différences entre rapports et des régressions linéaires et logistiques. Il est assorti de coefficients de pondération bootstrap; un document d'appoint explique comment modifier et utiliser le programme d'une manière qui réponde aux besoins de l'utilisateur [Statistique Canada, 2002, p. 39].

programme un estimateur bootstrap de la variance pour $\hat{\theta}$, qui est donné par [Yeo et al., 1999, p. 3] :

$$v_B\left(\hat{\theta}\right) = \frac{1}{B}\sum_b \left(\hat{\theta}^*_{(b)} - \hat{\theta}^*_{(.)}\right)^2 \qquad (1)$$

où $\hat{\theta}^*_{(.)} = \left(1 \Big/ B\right)\sum_b \hat{\theta}^*_{(b)}$,

$\hat{\theta}$ est une estimation d'intérêt et B correspond au nombre d'échantillons bootstrap.

III. Procédure

Le programme Stata est facile à utiliser : il suffit de copier les fichiers bswreg.ado et bswreg.hlp (qui sont décrits à l'annexe I) dans le répertoire Stata ADO[2], puis d'utiliser la commande suivante :

bswreg depvar [varlist] weighttype=full_sample_weight [if exp] [in range],
cmd(STATA_regression_command) [cmdops(options_for_regression_command)]
bsweights(bootstrap_weights_varlist) [level(integer)] [bsci]
[saving(path_and_filename[,replace])];

Prenons l'exemple d'un chercheur qui souhaite effectuer une régression portant sur la taille à l'aide de la méthode des moindres carrés ordinaires, en se servant du quatrième cycle de l'Enquête nationale sur la santé de la population (ENSP)[3]; il utilise à cette fin des variables fictives concernant la province, le niveau de scolarité, l'âge et le sexe. Il décide de sauvegarder le tableau des données de sortie bootstrap sous forme de fichier de données (y compris les coefficients bootstrap, les erreurs-types et d'autres statistiques d'inférence); il choisit en outre d'utiliser les 500 coefficients de pondération bootstrap disponibles. La ligne de commande à utiliser sera alors[4] :

bswreg height nfld pei ns nb qc on man sask alb lshs someps ugrad
agesq age gender [pw=wt60lf], cmd(reg) bsweights(bsw1-bsw500)
level(95) bsci saving(c:\temp\bswdata.dta, replace); (2)

[2] À partir de l'invite de Stata, il faut taper « adopath » pour faire afficher une liste de répertoires ado où copier ce programme. Le chercheur doit également définir la taille des paramètres « matsize » et « memory » en fonction de l'ordinateur et de l'ensemble des données qu'il utilise.

[3] L'échantillon du fichier maître de données longitudinales du quatrième cycle de l'ENSP est ramené de 17 276 à 12 439, ne contenant que les répondants des cycles un à quatre ainsi que les enregistrements où il ne manque aucune observation. Aux fins de la régression, la variable expliquée est la taille; elle correspond à une échelle qui normalise le système métrique et le système impérial. Une valeur de 50 sur cette échelle correspond à cinq pieds (60 pouces) (de 151,1 à 153,6 cm), et chaque unité de hausse de l'échelle correspond à un pouce. Les variables fictives provinciales sont : nfld, pei, ns, nb, qc, on, man, sask et alb, bc étant la province omise. Les variables relatives au niveau de scolarité sont : études secondaires non terminées (lshs), diplôme d'études secondaires (hsgrad – variable omise), études postsecondaires partielles (someps), et diplôme universitaire (ugrad). Les variables relatives à l'âge sont : age, et age-squared. La variable relative au sexe est 1 pour les hommes et 2 pour les femmes.

[4] Les résultats de cette régression sont présentés à l'annexe II.

On supposera que, au moment d'utiliser cette commande, les coefficients de pondération bootstrap pertinents et le fichier de données ont déjà été fusionnés à l'aide des identificateurs exclusifs appropriés (dans le cas du quatrième cycle de l'ENSP, les identificateurs exclusifs sont realukey et personid). Il n'est pas nécessaire d'attribuer un nom aux coefficients de pondération bootstrap. De plus, la commande bswreg permet d'utiliser différentes options. Le programme en offre plusieurs :

cmd : précise la commande de régression dans Stata pour la méthode bootstrap. Cette fonction est nécessaire. Les commandes de régression suivantes ont été mises à l'essai : regress, logit, probit, tobit, ologit, oprobit, biprobit, mlogit, qreg, glm, intreg, boxcox, (à peu près toute technique d'estimation en une étape devrait fonctionner avec ce programme) et non-twostage, soit des commandes « xt » compatibles avec des coefficients de pondération.

bsweights : établit une liste des coefficients de pondération bootstrap, cette liste pouvant varier. C'est une commande nécessaire. Par exemple, si les coefficients de pondération bootstrap sont nommés bsw1 to bsw500, vous pouvez formuler cette option ainsi : bsweights(bsw1-bsw500).

cmdops : sert à indiquer dans la commande de régression de Stata obtenue avec cmd() les options que l'on veut utiliser. Certaines options sont utiles tandis que d'autres sont dénuées de pertinence lorsqu'on utilise des coefficients de pondération bootstrap. Par exemple, si l'on exécute la commande REGRESS sans constante, on emploiera les options cmd(regress) cmdops(noconstant). Des options comme « robust » sont inutiles ici, étant donné que la commande sert à calculer des erreurs-types pondérées selon la méthode bootstrap, et non des erreurs-types robustes.

level : établit le niveau de confiance, exprimé en pourcentage, pour des intervalles de confiance donnés. L'option par défaut est : level(95).

bsci : indique que les intervalles de confiance doivent être calculés à partir de la distribution brute des coefficients selon la méthode bootstrap plutôt qu'au moyen de la formule standard fondée sur l'erreur-type obtenue selon la méthode bootstrap et sur la distribution normale[5].

saving : sert à sauvegarder les statistiques bootstrap dans un fichier de données Stata distinct, qui peut être par la suite chargé et utilisé au moyen d'autres fichiers.DO et .ADO. Le fichier aura par défaut l'extension.dta. Une option est prévue pour écraser les fichiers existants.

IV. Caractéristiques

La commande bswreg constitue pour le chercheur un outil hautement adaptable qui n'existait pas auparavant. Il est possible de générer des variances estimatives fiables en parallèle avec les techniques analytiques reposant sur la méthode des moindres carrés ordinaires, les probits et les régressions par quantile relativement aux modèles tobit d'effets aléatoires, lorsqu'on les

[5] Cette option est offerte à l'intention des utilisateurs qui pourraient avoir des motifs d'ordre théorique d'utiliser les intervalles de confiance calculés à partir de la distribution des coefficients selon la méthode bootstrap.

utilise de concert avec la commande bswreg pour obtenir des variances estimatives fondées sur le plan d'enquête.

Les commandes comportent une fonction intégrée d'aide de base à l'égard de sujets de recherche prédéfinis. Il suffit de taper help bswreg à l'invite de commande pour faire afficher une description de la procédure, une liste de variables ainsi que des exemples de codes utilisant la commande bswreg.

Considérant l'éventail des techniques d'analyse compatibles avec la commande bswreg, il faut utiliser un algorithme semi-complexe de résolution d'erreurs. Plus précisément, dans le cas des techniques d'estimation où il faut procéder à des itérations pour parvenir à une convergence, il peut arriver que le modèle ne donne pas lieu à une convergence pour tous les ensembles de coefficients de pondération bootstrap. La commande bswreg a donc été conçue de manière que soit retenu par défaut le nombre de procédures bootstrap ayant donné des résultats probants, de manière à éviter les erreurs, entre autres les erreurs de convergence[6].

Le programme est en outre conçu de manière à tenir compte des régressions bootstrap ratées. Il y a deux circonstances en particulier où cela peut se produire. En premier lieu, si un coefficient de pondération nul correspond parfaitement à la taille d'un petit échantillon pour une variable discrète, la variable est abandonnée. Lorsque cela se produit, les résultats de la régression pour l'échantillon bootstrap en question ne donneront pas le même nombre de variables que dans le cas d'un échantillon bootstrap où l'opération a pu être menée dans son intégralité. Cette situation pose problème relativement au calcul des estimations de la variance, la solution étant de laisser tomber les échantillons bootstrap en question. En second lieu, lorsque l'échantillon est très petit, il peut arriver que certains des coefficients de pondération bootstrap donnent à la majorité des observations échantillonnées un poids égal à 0. L'échantillon ainsi obtenu pourrait être trop petit pour que l'on puisse procéder à des estimations, de sorte que cette fois encore il serait éliminé de la procédure bootstrap. Cela aura pour conséquence que le nombre total de régressions bootstrap dont il sera fait état une fois l'exécution du programme terminée sera plus petit que le nombre indiqué au départ dans la commande bswreg.

Les résultats produits à la fin de la procédure bswreg englobent le nombre total d'opérations bootstrap menées, le nom des variables, la valeur estimative et l'erreur-type bootstrap des coefficients, la statistique z, la valeur p, les intervalles de confiance inférieurs et supérieurs à 95 % (par défaut) – selon l'hypothèse d'une distribution normale, l'option bsci produisant des intervalles de confiance supérieurs et inférieurs à 95 % directement à partir de la distribution bootstrap brute. Par rapport aux résultats obtenus, ce qui caractérise d'abord le programme est l'option permettant de produire deux ensembles d'intervalles de confiance : le premier, selon l'hypothèse d'une distribution normale, le second, à partir de la distribution bootstrap brute.

[6] Par exemple, si l'on a choisi 500 coefficients de pondération bootstrap pour exécuter une procédure itérative d'après la méthode du maximum de vraisemblance (modèles logit à effets aléatoires ou pondérés en fonction de la population) et que x régressions n'ont pas donné lieu à une convergence en raison de la nature de x poids d'échantillonnage bootstrap, il y a dès lors 500 - x régressions pondérées au moyen des poids d'échantillonnage bootstrap qui ont réussi et qui serviront à produire l'estimateur de variance bootstrap. Les données de sortie BSWREG indiquent le nombre d'itérations concluantes effectuées.

Toute commande en vue d'effectuer des régressions multiples à l'aide d'une variable qui analyse les données fonctionnera avec la commande bswreg. Par exemple, le préfixe « by var_name : » utilisé dans une commande de régression fonctionnera avec bswreg. Reprenons l'exemple précédent : si le chercheur veut utiliser la méthode des moindres carrés ordinaires pour chaque sexe et utiliser uniquement les coefficients de pondération bootstrap bsw100 à bsw400, prévoir un intervalle de confiance de 99 % et employer la distribution bootstrap, il rédigera la commande suivante :

bysort gender: bswreg height nfld pei ns nb qc on man sask alb
lshs someps ugrad agesq age [pw=wt60lf], cmd(reg) (3)
bsw(bsw100-bsw400) level(99) bsci;

La commande bswreg a principalement pour objet l'établissement d'estimations fiables de la variance ainsi que de données connexes, et non le calcul de quelconques statistiques accessoires. Si ces dernières ne sont pas calculées, c'est en raison de l'éventail d'estimations possibles, étant donné le degré de compatibilité du programme. Par exemple, les risques relatifs associés aux régressions logit sont omis, tout comme les modifications de probabilité applicables à des variables fictives dans le cadre des procédures probit.

Le programme peut être utilisé pour calculer différentes statistiques sommaires, par exemple des fréquences, des moyennes et des rapports. On trouvera à l'annexe III des exemples illustrant la manière de calculer ces statistiques.

Une remarque concernant la vérification des hypothèses : le programme bswreg entrepose les coefficients dans la matrice e(b) standard et la matrice de variance-covariance bootstrap dans e(V). Cela fait en sorte que la commande de vérification peut être utilisée immédiatement après bswreg pour effectuer les tests. Toutefois, dans toutes les vérifications, on utilisera la distribution normale asymptotique plutôt que la distribution t. Par conséquent, tous les tests exécutés avec la commande « test » seront des tests de style WALD fondés sur la distribution chi carré. Le chercheur ne peut effectuer de test F; de toute façon, le résultat obtenu serait inexact, étant donné la nature asymptotique de la technique bootstrap.

V. Tests d'exactitude

Le programme a fait l'objet de tests dans le but de vérifier l'exactitude des résultats obtenus. Dans tous les tests, le programme bootvare_v20.sas a été utilisé à des fins de comparaison. La procédure de régression par les moindres carrés ordinaires dans bootvar a été reproduite au moyen de la commande bswreg. Étant donné que bootvar a été conçu pour l'ENSP, un ensemble de données longitudinales englobant le quatrième cycle de cette enquête a servi pour les tests. En outre, on a utilisé les 500 coefficients de pondération bootstrap lors de chaque procédure. Les résultats sont exposés à l'annexe II. Également, les statistiques sommaires concernant les fréquences, les moyennes et les rapports ont été reproduites à l'aide d'un cadre de régression comparable (se reporter à l' annexe III).

Les résultats obtenus à l'aide des deux programmes sont identiques dans le cas de l'exemple reposant sur les moindres carrés ordinaires. En outre, toutes les estimations relatives à la variance, par exemple l'erreur-type, la statistique z, la valeur p et les limites inférieures et supérieures (dans l'hypothèse d'une distribution normale) étaient identiques. Ces résultats démontrent que le programme présente un degré de fiabilité élevé.

VI. Conclusion

Le programme examiné ici a pour objet de produire des estimations fiables de la variance à partir d'un large éventail des techniques d'analyse. Le programme Stata exposé ici peut servir à calculer les estimations de la variance à l'aide de coefficients de pondération bootstrap relativement à une gamme étendue d'ensembles de données (des coefficients de pondération sont disponibles à l'heure actuelle pour l'ELNEJ, l'ENSP, l'EDTR, et l'EJET)[7]. On a pu vérifier l'exactitude des résultats donnés par ce programme au moyen de tests de comparabilité par rapport à d'autres programmes analytiques existants. Les chercheurs qui utilisent Stata disposent désormais d'un outil convivial, précis et adaptable à leurs besoins.

Bibliographie

Statistics Canada. 2002. "Population Health Surveys Program: National Population Health Survey, Cycle 4 (2000 – 2001), Household Component Longitudinal Documentation." Health Statistics Division. Ottawa.

Yeo, Douglas, Harold Mantel, and Tzen-Ping Liu. 1999. "Bootstrap Variance Estimation For the National Population Health Survey." American Statistical Association: Proceedings of the Survey Research Methods Section. Baltimore, August.

[7] Les enquêtes suivantes constituent des ensembles de données de base disponibles par l'intermédiaire du Programme des centres de données de recherche de Statistique Canada : l'Enquête longitudinale nationale sur les enfants et les jeunes (ELNEJ), l'Enquête nationale sur la santé de la population (ENSP), l'Enquête sur la dynamique du travail et du revenu (EDTR), et l'Enquête auprès des jeunes en transition (EJET).

Annexe I

Fichier Ado

```
*                                    WARNING

* The authors are the owners of all intellectual
* property rights (including copyright) in this software.  Subject to the terms below,
* you are granted a non-exclusive and non-transferable license to use this software.
*
* This software is provided "as-is", and the owner makes no warranty, either express
* or implied, including but not limited to, warranties of merchantability and fitness
* for any particular purpose.  In no event will the owner be liable for any indirect,
* special, consequential or other similar damages.  This agreement will terminate
* automatically without notice to you if you fail to comply with any term of this
* agreement.

* TO CHANGE THE DECIMAL DISPLAY FORMAT OF THE BOOTSTRAPPED OUTPUT SEARCH FOR THE "FORMAT" COMMAND
NEAR THE BOTTOM OF THIS PROGRAM;

program define bswreg, eclass sortpreserve byable(recall)
* August 8th, 2003 Pierard, Buckley, Chowhan

# delimit;
version 7.0;

syntax varlist(numeric) [aweight pweight fweight iweight] [if] [in], cmd(string) [cmdops(string)]
  BSWeights(varlist numeric) [Level(integer 95)] [BSci] [SAVing(string)];
*This sets the touse variable = 1 if observation is in our sample;
marksample touse;
*Error check to make sure a weight was used;
if "`weight'"==""
  {;
  noi di in red "BSWREG error: You must specify a weight!";
  exit;
  };

quietly
{;
*Preserve the original dataset and set parameter values and setup temporary matrices;
preserve;
set more 1;
tempvar esamplevar;
tempname bhat bsVC bsbhat bsbetas;

*The next line runs the wanted regression and checks for errors;
capture `cmd' `varlist' [`weight'`exp'] if `touse' `in', `cmdops';
if _rc ~= 0
   {;
   noi di in red " ";
   noi di in red "Error doing: `cmd' `varlist' [`weight'`exp'] `if' `in', `cmdops'";
   noi di in red " ";
   noi di in red "The regression command you have typed in resulted in an error, please
investigate";
   noi di in red "this error outside of the 'bswreg' program by typing in the regression command
itself";
   noi di in red "with the options you specified.";
   noi di in red " ";
   exit;
   };
*The next line runs the wanted regression and we store the coefficients in a matrix for later use;
`cmd' `varlist' [`weight'`exp'] if `touse' `in', `cmdops';
gen `esamplevar'=e(sample);
*e(b) is a 1x(k+1) coefficient vector if the model has a constant and k is the number of variables
other than the constant;
matrix `bhat'=e(b);
matrix `bsVC'=e(V);
*we store the variable names of the regressors and the number of regressors in local macros;
```

```
local _varnames : colfullnames(`bhat');
local _k=colsof(`bhat')-1;
local _k1=`_k'+1;
*Generate concatenated list of placeholder regressor variable names xc1-xck1, later to be turned
into variables;
local _xclist="";
forvalues _i = 1/`_k1'
   {;
   local _xclist `_xclist' _xc`_i';
   };
*We assigned these placeholder variable names to the regressors in the coefficient vector;
matrix colnames `bhat' = `_xclist';
*Each "true estimate of beta" is saved under it's own variable name;
svmat double `bhat', name(col);
matrix colnames `bhat' = `_varnames';

*Realboot is the actual number of successful bootstrap regressions run in case we get any
convergence/regression errors etc., it starts off at the specified number of bootstrap weights;
local _realboot: word count `bsweights';
noi di " ";

*The main bootstrap loop will run with each bootstrap weight in the supplied bsweight varlist and
exit with the matrix named BETAS containing all the bootstraps of our coefficients, a (boot)x(k+1)
dimensional matrix;
local _i 1;
*Start of bootstrap loop;
foreach bswvar of local bsweights
 {;
 *Display notice of number of completed bootstraps every time 50 are completed;
 if mod(`_i',50)==0
  {;
  noi di `_i' " bootstraps completed";
  };
 *Run the regression with the chosen set of bootstrap weights, only use the coefficients if there
are no errors;
 capture `cmd' `varlist' [`weight'=`bswvar'] if `touse' `in', `cmdops';
 if _rc==0
  {;
   *Store coefficients in the bootstrap matrix;
   matrix `bsbhat'=get(_b);
   *bsbhat is a 1x(`k'+1) (row) vector if the model has a constant.  Need to transpose;
   matrix `bsbhat'=`bsbhat'';
   *If we have the proper number of coefficients then add them to the bootstrap matrix, otherwise
do not add them (this most likely arises due to a regressor being dropped due to multicollinearity;
   if rowsof(`bsbhat')==`_k1'
        {;
        *If we are on the first bootstrap then create the bsbetas matrix, otherwise append to it;
        if `_i'==1
           {;
           matrix `bsbetas'=(`bsbhat');
           };
        else
           {;
           matrix `bsbetas'=(`bsbetas',`bsbhat');
           };
        };
   else
        {;
        matrix drop `bsbhat';
        local _realboot=`_realboot'-1;
        noi di "Bootstrap #`_i' has been dropped for not having the correct number of
coefficients";
        };
  };
 else
  {;
  local _realboot=`_realboot'-1;
  noi di "bootstrap #`_i' has been dropped due to an error estimating the regression";
  };
 local _i=`_i'+1;
```

```
 };
*End of bootstrap loop;

*All the bootstraps have been completed now calculate the new standard errors and display relevant
statistics;
*We must transpose the matrix to make each row now, then column, a new variable;
matrix `bsbetas'=`bsbetas'';
*Generate concatenated list of colnames, later to be turned into variables;
local _xvlist="";
forvalues _i = 1/`_k1'
   {;
   local _xvlist `_xvlist' _xv`_i';
   };
*Calls each row of the matrix by the name of the independent variable it corresponds to (we call
them _xv'_i' so that they are not mixed up with the "real" variables);
matrix colnames `bsbetas'= `_xvlist';
*Separate each column as a new variable.  The format of the data must be specified.  It renames
each variable by the name of the column;
svmat double `bsbetas', name(col);

*Generate the bootstrapped variance-covariance matrix, you can access this in e(V) after running
the BSWREG ado file;
forvalues _i = 1/`_k1'
 {;
 forvalues _j = 1/`_k1'
  {;
    correlate _xv`_i' _xv`_j', covariance;
    matrix `bsVC'[`_i',`_j'] = ((`_realboot'-1)/`_realboot')*r(cov_12);
  };
 };

*Generate the standard deviation, t-stat, conf. int. etc. for each variable;
tempvar _bsobs _uniqobs _coefnum;
gen `_bsobs'=_n;
forvalues _i = 1/`_k1'
 {;
 sum _xv`_i';
 * Like the SAS bootvar program, we use (boot-1)/boot because variance and standard error have
different denominators;
 gen _sdx`_i'=sqrt(((`_realboot'-1)/`_realboot')*r(Var)) in 1/1;
 gen _t`_i'=_xc`_i'/_sdx`_i' in 1/1;
 gen _abst`_i'=abs(_t`_i') in 1/1;
 gen _p`_i'=2*norm((-1)*_abst`_i') in 1/1;
if "`bsci'"==""
 {;
   gen _low`level'`_i'=_xc`_i'-invnorm(1-((1-(`level'/100))/2))*_sdx`_i';
   gen _high`level'`_i'=_xc`_i'+invnorm(1-((1-(`level'/100))/2))*_sdx`_i';
 };
 if "`bsci'"=="bsci"
   {;
   sort _xv`_i';
   local _obslow= max(1,round(((1-(`level'/100))/2)*`_realboot',1));
   local _obshigh= max(1,round((1-((1-(`level'/100))/2))*`_realboot',1));
   local _obslow2= _xv`_i'[`_obslow'];
   local _obshigh2= _xv`_i'[`_obshigh'];
   sort `_bsobs';
   gen _low`level'`_i'= `_obslow2' in 1/1;
   gen _high`level'`_i'= `_obshigh2' in 1/1;
   };
 };

*Assign each coefficient its true regressor name stored at the beginning of this program;
local _i=1;
foreach _curname in `_varnames'
 {;
 gen str10 _xname`_i'="`_curname'";
 local _i=`_i'+1;
 };
```

```
*Reshape the data so that the bootstrapped stats can be displayed easily, and then display the
results;
keep _xname* _xc* _sdx* _t* _p* _low`level'* _high`level'*;
drop if _n>1;
gen `_uniqobs'=1;

reshape long _xname _xc _sdx _t _p _low`level' _high`level', i(`_uniqobs') j(`_coefnum');
*The %9.4f tells stata to display the bootstrapped results to 6 decimals using 15 numbers total --
this can be changed to suit tastes;
format _xc _sdx _t _p _low`level' _high`level' %11.6f;

*creates nice labels for variables
label var _xname "Name of variable";
ren _xname Var_name;
label var _xc "Coefficient estimate";
ren _xc Coef;
label var _sdx "Bootstrap standard error of coefficient";
ren _sdx BS_se;
label var _t "Bootstrap z-statistic";
ren _t BS_zstat;
label var _p "Bootstrap p-value";
ren _p BS_pvalue;
if "`bsci'"==""
 {;
 label var _low`level' "Bootstrap lower 95% confidence interval assuming a normal distribution";
 label var _high`level' "Bootstrap upper 95% confidence interval assuming a normal distribution";
 };
if "`bsci'"=="bsci"
 {;
 label var _low`level' "Bootstrap lower 95% confidence interval using bootstrap sample
distribution";
 label var _high`level' "Bootstrap upper 95% confidence interval using bootstrap sample
distribution";
 };
ren _low`level' BS_cilow`level';
ren _high`level' BS_ciup`level';

*Display RESULTS!;
noi display in green "Results from BSWREG";
noi display in green "--------------------";
noi display in green " ";
if "`bsci'"=="bsci"
 {;
 noi display in green "* The confidence intervals below are based on the bootstrapped
distribution";
 };
else noi display in green "* The confidence intervals below are based on the normal distribution";
noi display in green " ";
noi list Var_name Coef BS_se BS_zstat BS_pvalue BS_cilow`level' BS_ciup`level', nodisplay noobs;
noi di " ";
noi di "Total bootstraps completed: `_realboot'";

*Set the eclass variables like the coefficients and the variance-covariance matrix into their
appropriate matrices so that F-tests and the like can be run;
*If you wish the TEST command to produce F-tests after the BSWREG command then add ",
dof(`_realboot')" to the line below;
estimates post `bhat' `bsVC';

*Save the bootstrap raw data is the "SAVING" option has been used;
if "`saving'"~=""
 {;
  drop _*;
  save "`saving'", `replace';
 };

*Restore the original dataset
restore;

};
end;
```

Fichier d'aide BSWREG

```
{smcl}
{* 8August2003 Pierard/Buckley/Chowhan}
{hline}
help for {hi:BSWREG}
{hline}
{title:BSWREG - uses bootstrap weights to calculate standard errors in models involving complex
survey data.}

{p 8 13}{cmd:bswreg} depvar [varlist] {it:weighttype}={it:full_sample_weight} [{cmd:if} {it:exp}]
[{cmd:in} {it:range}]{cmd:,} {cmd:cmd(){it:STATA_regression_command}{cmd:)}
[{cmd:cmdops(){it:options_for_regression_command}{cmd:)}]
  {cmdab:bsw:eights(){it:bootstrap_weights_varlist}{cmd:)} [{cmd:level(){it:integer}{cmd:)}]
[{cmd:bsci}] [{cmdab:sav:ing(){it:path_and_filename}[{cmd:,replace}]{cmd:)}];

{p} {cmd:cmd()} and {cmd:bsweights()} are required options for the {cmd:BSWREG} command.
{p} {cmd:by ...: } and {cmd:bysort ...:} can be used with {cmd:BSWREG}. See help {help by}.
{p} {cmd:aweight}s, {cmd:fweight}s, {cmd:iweight}s, and {cmd:pweight}s are allowed as long as the
      given regression command is compatible with them. See help {help weights}.
{p} As {cmd:BSWREG} is an eclass STATA program, it provides STATA with the {cmd:e(b)} coefficient
vector and the {cmd:e(V)} bootstrapped variance-covariance matrix.
    The {cmd:test} command can be used immediately following the {cmd:BSWREG} command to conduct
Wald tests based on the chi-squared distribution.

{inp:The software is provided "as-is" and the authors are not responsible for any misuse.}
{title:Description}
(used to calculate regression statistics using Statistics Canada's bootstrap weights)

{p}{cmd:bswreg} runs a number of regressions, each with a particular bootstrap
weight so that bootstrapped standard errors on the coefficients can be calculated
and displayed. Use of bootstrap weights is recommended for calculating reliable
standard errors, confidence intervals etc. on data from complex
household surveys.

The user provides the names of the bootstrap weights to the {cmd:BSWREG} command
in the {cmdab:bsw:eights(varlist)} option. You must already have the appropriate
bootstrap weights merged into your datafile for this command file to work.  NPHS
merges on REALUKEY and SLID merges on PERSONID. Below is a sample .DO file that
merges NPHS bootstrap weights into a datafile named data.dta:

{inp:use data.dta, replace"}
{inp:sort realukey"}
{inp:save data.dta, replace"}
{inp:use bootstrap/sas_bs_wt_1_4.dta, replace"}
{inp:destring realukey, replace"}
{inp:sort realukey"}
{inp:merge realukey using data.dta"}
{inp:keep if _merge==3"}

{title:Options}

{p 0 4}{cmd:cmd(){it:STATA_regression_command}{cmd:)} specifies the Stata regression command to
bootstrap. This is a {cmd:required} option. "regress", "probit" and "logit" are a few
possibilities.

{p 0 4}{cmd:bsweights(){it:varlist}{cmd:)} specifies a variable list of the bootstrap weight names.
This is a {cmd:required} option. For instance, if your bootstrap weights are named bsw1 to bsw500,
you may wish to use the
{cmd:bsweights(bsw1-bsw500)} option.

{p 0 4}{cmd:cmdops(){it:options_for_regression_command}{cmd:)} specifies the options you wish to
use on the Stata regression command provided in {cmd:cmd()}. Some options are useful and others are
meaningless in a bootstrap weighting context.
For instance, if you wish to run the REGRESS command with no constant then use the
{cmd:cmd(regress) cmdops(noconstant)} options.  Options like {cmd:robust} are meaningless in this
context since the command computes bootstrap weighted
standard errors not robust ones.
```

```
{p 0 4}{cmd:level(}{it:integer}{cmd:)} specifies the confidence level, in percent,
for confidence intervals.  The default is {cmd:level(95)}. See help {help level}.

{p 0 4}{cmd:bsci} specifies that the confidence intervals be calculated from the raw bootstrapped
distribution of coefficients rather than using the standard formula
based on the bootstrapped standard error and the normal distribution.

{p 0 4}{cmd:saving(}{it:filename}[{cmd:,replace}]{cmd:)} saves the bootstrap statistics in a
separate Stata dataset file that can later be loaded and used by other .DO and .ADO files.
If you do not specify an extension, {cmd:.dta} will
be assumed. Include the {cmd:,replace} option to overwrite an existing file.

{title:Outputed variables}

{inp: Var_name:}  This is the STATA variable name of the regressor.
{inp: Coef:}      This is the coefficient from the specified regression.
{inp: BS_se:}     This is the new standard error of the coefficient,
            calculated using bootstrap weights.
{inp: BS_zstat:}  This is the new z-stat of the coefficient,
            calculated as the coefficient divided by the bootstrapped standard error.
{inp: BS_pvalue:} This is the new p-value of the coefficient,
            calculated using the z-statistic.
{inp: BS_cilow95n:} This is the lower (level)% confidence interval around the coefficient
            using the bootstrapped std. error.
{inp: BS_ciup95n:} This is the upper (level)% confidence interval around the coefficient
            using the bootstrapped std. error.

{title:Examples}

{p 8 12}{inp:. bswreg income education rural [aw=wt] if married==1, cmd(regress) bsw(bsw1-bsw500)}

{p 8 12}{inp:. bswreg employed education rural [aw=wt66], cmd(probit) bsw(bsw50-bsw100)}

{p 8 12}{inp:. bysort maritalstatus: bswreg income education rural [aw=wt], cmd(reg) bsw(bsw1-
bsw500)}
  {inp:cmdops(noconstant) level(99) bsci saving(c:\data\bsw1.dta,replace)}
```

Annexe II

BSWREG – Résultats

```
. reg height  nfld pei ns nb qc on man sask alb lshs  someps ugrad agesq age gender [pw=wt60lf];
(sum of wgt is   2.6597e+07)

Regression with robust standard errors                Number of obs =   12439
                                                      F( 15, 12423) =  279.22
                                                      Prob > F      =  0.0000
                                                      R-squared     =  0.4753
                                                      Root MSE      =   4.208
------------------------------------------------------------------------------
             |               Robust
      height |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
        nfld |  -.2922724   .2449279    -1.19   0.233    -.7723689    .1878242
         pei |  -.3285292   .2671396    -1.23   0.219    -.8521642    .1951058
          ns |  -.5414279   .2420961    -2.24   0.025    -1.015974   -.0668821
          nb |  -.5146936   .2386633    -2.16   0.031    -.9825106   -.0468766
          qc |  -.7923805   .1886131    -4.20   0.000    -1.162091   -.4226696
          on |  -.4070276   .1911941    -2.13   0.033    -.7817977   -.0322575
         man |   .2473519   .2449153     1.01   0.313      -.23272    .7274238
        sask |   .1010543   .2473725     0.41   0.683    -.3838343    .5859428
         alb |   .2328238   .2169931     1.07   0.283    -.1925163    .6581638
        lshs |   3.759076   .1975814    19.03   0.000     3.371786    4.146366
      someps |   3.506946   .1773476    19.77   0.000     3.159318    3.854575
       ugrad |   3.155798   .1650738    19.12   0.000     2.832228    3.479368
       agesq |  -.0047157   .0001675   -28.16   0.000     -.005044   -.0043874
         age |    .465882   .0161484    28.85   0.000     .4342285    .4975354
      gender |  -4.695284   .1128784   -41.60   0.000    -4.916543   -4.474025
       _cons |   50.90797    .503058   101.20   0.000      49.9219    51.89404
------------------------------------------------------------------------------

. bswreg height  nfld pei ns nb qc  on man sask alb lshs  someps ugrad agesq age gender
[pw=wt60lf], cmd(reg) bsw(bsw1-bsw500)  level(95) bsci saving(c:\temp\bswdata.dta, replace);

50 bootstraps completed
100 bootstraps completed
150 bootstraps completed
200 bootstraps completed
250 bootstraps completed
300 bootstraps completed
350 bootstraps completed
400 bootstraps completed
450 bootstraps completed
500 bootstraps completed
Results from BSWREG
-------------------

* The confidence intervals below are based on the bootstrapped distribution

  Var_name         Coef        BS_se     BS_zstat    BS_pvalue   BS_cilow95    BS_ciup95
      nfld    -0.292272     0.211569    -1.381455     0.167139    -0.708804     0.145642
       pei    -0.328529     0.241579    -1.359923     0.173854    -0.780092     0.131531
        ns    -0.541428     0.206423    -2.622899     0.008719    -0.968447    -0.145737
        nb    -0.514694     0.231676    -2.221613     0.026309    -1.042424    -0.094749
        qc    -0.792381     0.155897    -5.082725     0.000000    -1.079814    -0.477167
        on    -0.407028     0.161284    -2.523667     0.011614    -0.711807    -0.081808
       man     0.247352     0.203860     1.213342     0.224999    -0.162946     0.623552
      sask     0.101054     0.218886     0.461676     0.644314    -0.360283     0.520983
       alb     0.232824     0.193489     1.203289     0.228864    -0.173770     0.640249
      lshs     3.759076     0.171806    21.879725     0.000000     3.417291     4.105172
    someps     3.506946     0.173803    20.177696     0.000000     3.194609     3.866022
     ugrad     3.155798     0.158046    19.967600     0.000000     2.859345     3.466744
     agesq    -0.004716     0.000158   -29.844374     0.000000    -0.005035    -0.004413
       age     0.465882     0.015283    30.483185     0.000000     0.437043     0.496086
    gender    -4.695284     0.095502   -49.164188     0.000000    -4.879501    -4.502970
     _cons    50.907968     0.435441   116.911263     0.000000    50.092201    51.765503
 Total bootstraps completed: 500
```

Bootvare_v20 – Résultats

The REG Procedure
Dependent Variable: height
Weight: WT60LF

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	15	426030656	28402044	750.15	<.0001
Error	12423	470359719	37862		
Corrected Total	12438	896390375			

Root MSE	194.58162	R-Square	0.4753
Dependent Mean	55.33538	Adj R-Sq	0.4746
Coeff Var	351.64051		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	50.90797	0.21433	237.52	<.0001
nfld	1	-0.29227	0.30121	-0.97	0.3319
pei	1	-0.32853	0.56080	-0.59	0.5580
ns	1	-0.54143	0.23678	-2.29	0.0222
nb	1	-0.51469	0.25727	-2.00	0.0455
qc	1	-0.79238	0.13134	-6.03	<.0001
on	1	-0.40703	0.12280	-3.31	0.0009
man	1	0.24735	0.22511	1.10	0.2719
sask	1	0.10105	0.23464	0.43	0.6667
alb	1	0.23282	0.16058	1.45	0.1471
lshs	1	3.75908	0.11369	33.06	<.0001
someps	1	3.50695	0.11898	29.48	<.0001
ugrad	1	3.15580	0.11630	27.14	<.0001
agesq	1	-0.00472	0.00009228	-51.10	<.0001
age	1	0.46588	0.00821	56.78	<.0001
gender	1	-4.69528	0.07566	-62.06	<.0001

Variance estimation using 500 bootstraps for a Regression

Dependent variable: height

Obs	beta	bhat	bs_var	bs_sd	bs_cv	cil95	ciu95
1	Intercept	50.9080	0.18961	0.43544	0.86	50.0545	51.7614
2	nfld	-0.2923	0.04476	0.21157	72.39	-0.7069	0.1224
3	pei	-0.3285	0.05836	0.24158	73.53	-0.8020	0.1450
4	ns	-0.5414	0.04261	0.20642	38.13	-0.9460	-0.1368
5	nb	-0.5147	0.05367	0.23168	45.01	-0.9688	-0.0606
6	qc	-0.7924	0.02430	0.15590	19.67	-1.0979	-0.4868
7	on	-0.4070	0.02601	0.16128	39.62	-0.7231	-0.0909
8	man	0.2474	0.04156	0.20386	82.42	-0.1522	0.6469
9	sask	0.1011	0.04791	0.21889	216.60	-0.3280	0.5301
10	alb	0.2328	0.03744	0.19349	83.11	-0.1464	0.6121
11	lshs	3.7591	0.02952	0.17181	4.57	3.4223	4.0958
12	someps	3.5069	0.03021	0.17380	4.96	3.1663	3.8476
13	ugrad	3.1558	0.02498	0.15805	5.01	2.8460	3.4656
14	agesq	-0.0047	0.00000	0.00016	3.35	-0.0050	-0.0044
15	age	0.4659	0.00023	0.01528	3.28	0.4359	0.4958
16	gender	-4.6953	0.00912	0.09550	2.03	-4.8825	-4.5081

Annexe III

Tableaux de fréquences

```
. tab nfld [aw=wt60lf]

       nfld |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 | 12215.0502       98.20       98.20
          1 | 223.949797        1.80      100.00
------------+-----------------------------------
      Total |      12439      100.00


. reg nfld [aw=wt60lf]
(sum of wgt is   2.6597e+07)

      Source |       SS       df       MS              Number of obs =   12439
-------------+------------------------------           F(  0, 12438) =    0.00
       Model |        0.00     0           .           Prob > F      =       .
    Residual |   219.91784 12438  .017681126           R-squared     =  0.0000
-------------+------------------------------           Adj R-squared =  0.0000
       Total |   219.91784 12438  .017681126           Root MSE      =  .13297

------------------------------------------------------------------------------
        nfld |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _cons |   .0180038   .0011922    15.10   0.000     .0156669    .0203408
------------------------------------------------------------------------------


. bswreg nfld [aw=wt60lf] , cmd(reg) bsw(bsw1-bsw500)
Results from BSWREG

* The confidence intervals below are based on the normal distribution

  Var_name         Coef        BS_se     BS_zstat    BS_pvalue   BS_cilow95    BS_ciup95
     _cons     0.018004     0.000460    39.113152     0.000000     0.017102     0.018906

Total bootstraps completed: 500

*******************************************************************************************


Bootvare_v20
Variance estimation using 500 bootstraps for Totals and Ratios

Obs    type     var1    var2           yhat          bs_sd    bs_cv          ci195          ciu95
 1     Ratio    nfld    count          1.80           0.05     2.56           1.71           1.89

where count is equal to the population size.
```

Moyennes

```
. bysort gender: means age [aw=wt60lf]
-------------------------------------------------------------------------------
-> gender = 1
    Variable |    Type         Obs        Mean       [95% Conf. Interval]
-------------+-----------------------------------------------------------------
         age | Arithmetic    5597    38.74955        38.23552   39.26358
             |  Geometric    5597    32.88785        32.35452   33.42998
             |   Harmonic    5597    26.32428        25.78611   26.88539
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-> gender = 2
    Variable |    Type         Obs        Mean       [95% Conf. Interval]
-------------+-----------------------------------------------------------------
         age | Arithmetic    6842     40.6122        40.12048   41.10393
             |  Geometric    6842    34.30807        33.79512   34.82881
             |   Harmonic    6842    27.15345        26.63015   27.69773
-------------------------------------------------------------------------------
```

```
. bysort gender: reg age [aw=wt60lf]
-------------------------------------------------------------------------------
-> gender = 1
(sum of wgt is   1.3123e+07)

      Source |       SS       df       MS              Number of obs =    5597
-------------+------------------------------           F(  0,  5596) =    0.00
       Model |        0.00     0           .           Prob > F      =       .
    Residual |  2153437.42  5596  384.817267           R-squared     =  0.0000
-------------+------------------------------           Adj R-squared =  0.0000
       Total |  2153437.42  5596  384.817267           Root MSE      =  19.617

------------------------------------------------------------------------------
         age |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _cons |   38.74955   .2622102   147.78   0.000     38.23552    39.26358
------------------------------------------------------------------------------
-------------------------------------------------------------------------------
-> gender = 2
(sum of wgt is   1.3474e+07)

      Source |       SS       df       MS              Number of obs =    6842
-------------+------------------------------           F(  0,  6841) =    0.00
       Model |        0.00     0           .           Prob > F      =       .
    Residual |  2945099.01  6841  430.507092           R-squared     =  0.0000
-------------+------------------------------           Adj R-squared =  0.0000
       Total |  2945099.01  6841  430.507092           Root MSE      =  20.749

------------------------------------------------------------------------------
         age |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
       _cons |    40.6122   .2508411   161.90   0.000     40.12048    41.10393
------------------------------------------------------------------------------
```

```
. bysort gender: bswreg age [aw=wt60lf] , cmd(reg) bsw(bsw1-bsw500)
_____________________________________________________________________________
-> gender = 1
Results from BSWREG

* The confidence intervals below are based on the normal distribution

  Var_name         Coef        BS_se     BS_zstat    BS_pvalue   BS_cilow95    BS_ciup95
     _cons    38.749551     0.137592   281.626648     0.000000    38.479877    39.019226

Total bootstraps completed: 500
_____________________________________________________________________________
-> gender = 2
Results from BSWREG

* The confidence intervals below are based on the normal distribution

  Var_name         Coef        BS_se     BS_zstat    BS_pvalue   BS_cilow95    BS_ciup95
     _cons    40.612205     0.127663   318.121521     0.000000    40.361992    40.862419

Total bootstraps completed: 500

*******************************************************************************************
```

SAS — Proc Reg Output

```
gender=1 ------------------------------------------------------
                                        Dependent Variable: age
                                            Weight: WT60LF
                                          Parameter Estimates

                                       Parameter       Standard
                         Variable  DF      Estimate          Error    t Value    Pr > |t|
                         Intercept  1      38.74955        0.26221     147.78      <.0001
gender=2 ------------------------------------------------------
                                        Dependent Variable: age
                                            Weight: WT60LF
                                          Parameter Estimates

                                       Parameter       Standard
                         Variable  DF      Estimate          Error    t Value    Pr > |t|
                         Intercept  1      40.61220        0.25084     161.90      <.0001
```

Bootvare_v20

```
Variance estimation using 500 bootstraps for a Regression
Dependent variable: age
Obs    gender      beta         bhat      bs_var      bs_sd     bs_cv     cil95      ciu95
1        1      Intercept    38.7496    0.018932    0.13759     0.36    38.4799    39.0192
2        2      Intercept    40.6122    0.016298    0.12766     0.31    40.3620    40.8624
```

Rapports

```
. gen ah=age/height

. svyratio age/height [pw=wt60lf]

Survey ratio estimation

pweight:  wt60lf                                  Number of obs    =     12439
Strata:   <one>                                   Number of strata =         1
PSU:      <observations>                          Number of PSUs   =     12439
                                                  Population size  =  26596782

------------------------------------------------------------------------------
      Ratio       |   Estimate    Std. Err.   [95% Conf. Interval]        Deff
------------------+-----------------------------------------------------------
     age/height   |   .7173197    .0041964    .7090941    .7255453    1.752756
------------------------------------------------------------------------------



. bswreg age height [pw=wt60lf], cmd(svyratio) bsw(bsw1-bsw500)

50 bootstraps completed
100 bootstraps completed
150 bootstraps completed
200 bootstraps completed
250 bootstraps completed
300 bootstraps completed
350 bootstraps completed
400 bootstraps completed
450 bootstraps completed
500 bootstraps completed
Results from BSWREG
-------------------

* The confidence intervals below are based on the normal distribution


  Var_name         Coef        BS_se     BS_zstat    BS_pvalue   BS_cilow95    BS_ciup95
age:height     0.717320     0.001743   411.545410     0.000000     0.713903     0.720736

Total bootstraps completed: 500

*******************************************************************************************
```

Bootvare_v20

```
Variance estimation using 500 bootstraps for Totals and Ratios

Obs    type     var1     var2           yhat          bs_sd    bs_cv         cil95         ciu95
 1     Ratio    age      height         71.73          0.17     0.24         71.39         72.07
```

Estimation de la variance dans le cas de données sur les compétences basées sur des valeurs plausibles : Deux programmes STATA pour l'analyse des données de l'EJET/PISA

Par Darren Lauzon

Résumé

L'utilisation des données sur les compétences fondées sur des valeurs plausibles provenant de l'EJET/PISA introduit une variabilité due à l'erreur de mesure en plus et en sus de celle due à la variance d'échantillonnage induite par le plan de sondage. Deux programmes sont proposés. Le premier estime les erreurs-types et les limites de confiance pour des estimations produites par diverses commandes d'estimation de modèle de STATA lorsque l'on considère comme variables dépendantes des valeurs plausibles. Le deuxième produit des estimations de la densité par la méthode du noyau et les limites de confiance correspondantes pour les données fondées sur les valeurs plausibles. Le code du programme figure en annexe.

Introduction

L'un des points communs de la plupart des ensembles de données dont disposent les Centres de données de recherche (CDR) est qu'ils proviennent d'enquêtes à plan de sondage complexe. Dans le cas de l'Enquête auprès des jeunes en transition et du Programme international pour le suivi des acquis des élèves (EJET/PISA) vient s'ajouter une complication supplémentaire due à l'utilisation de données sur les compétences basées sur des valeurs plausibles (VP). Les valeurs plausibles, qui sont une nouveauté en théorie de réponse à l'item, sont utilisées de plus en plus fréquemment lors de tests destinés à mesurer les compétences des élèves pour estimer les paramètres de population, comme la performance moyenne ou les coefficients de régression de population, et ont des propriétés désirables comparativement à d'autres méthodes d'estimation existantes, particulièrement dans le cas de tests où le nombre d'items (questions) par individu est assez faible. Le lecteur est invité à consulter Mislevey (1991) et la documentation de la Troisième enquête internationale sur les mathématiques et les sciences (TEIMS) de 1995 (Gonsalez, Smith et Sibberns, 1998) pour une discussion approfondie. Le présent article décrit dans les grandes lignes l'utilisation de valeurs plausibles et discute de deux programmes STATA qui appliquent des méthodes appropriées pour calculer les erreurs-types des estimations produites au moyen de plusieurs commandes STATA d'estimation au moyen d'un modèle et une version modifiée de la commande de calcul de la densité par la méthode du noyau qui utilise des données sur les connaissances basées sur des valeurs plausibles comme variable dépendante.

II. Utilisation de valeurs plausibles

Soit θ tout paramètre de population (p. ex. une moyenne, un vecteur de coefficients de régression) et soit la statistique $\widehat{\theta}_j$ une estimation de ce paramètre obtenue en utilisant la j^e valeur plausible. Par exemple, l'estimateur par les moindres carrés ordinaires (MCO) des coefficients de

régression de population pour un vecteur (colonne) de variables de conditionnement x_i pour le i^e élève serait $\left(\sum_i x_i x_i'\right)^{-1} \sum_i x_i y_{ij}$ où y_{ij} est la valeur de la j^e valeur plausible pour l'individu i. Bien qu'il soit possible de communiquer cette statistique, l'estimation peut varier considérablement si l'on choisit pour variables d'autres valeurs plausibles que la j^e. Il est par conséquent recommandé de communiquer l'estimation moyenne, $\hat{\theta} = (1/J)\sum_j \hat{\theta}_j$, J étant le nombre de valeurs plausibles tirées (cinq dans le cas des données de l'EJET/PISA). Naturellement, il convient de noter que, dans le cas d'un estimateur linéaire, comme la moyenne d'échantillon ou l'estimateur par les MCO, cela revient à estimer la statistique en utilisant la moyenne des J valeurs plausibles comme variable dépendante. Néanmoins, il est encore nécessaire de calculer les valeurs $\hat{\theta}_j$ afin de calculer la contribution de l'erreur de mesure à la variance totale, si bien que, dans le cas des estimateurs linéaires, le calcul de la moyenne des valeurs plausibles ne présente aucun avantage réel.

On estime la variance totale de $\hat{\theta}$ par la moyenne des estimations de la variance d'échantillonnage pour l'ensemble de J estimations $\hat{\theta}_j$, et une deuxième composante fondée sur la variabilité entre les estimations $\hat{\theta}_j$:

$$\text{tot var}(\hat{\theta}) = (1/J)\sum_j \text{var}(\hat{\theta}_j) + \left(1 + 1/J\right)\left(1/(J-1)\right)\sum_j (\hat{\theta}_j - \hat{\theta})^2 \qquad (1)$$

III. Variance d'échantillonnage pour les données de l'EJET/PISA

L'estimateur de la variance d'échantillonnage utilisé dans le calcul de la variance totale devrait tenir compte du plan de sondage. Au Canada, ainsi que dans d'autres pays membres de l'OCDE participants, on a utilisé un plan à répliques répétées équilibrées (BRR pour *Balanced Repeated Replication*). L'ensemble de données du PISA contient 80 poids de rééchantillonnage BRR pour tous les pays, y compris le Canada. Cependant, un beaucoup plus grand nombre d'écoles ont été échantillonnées au Canada que dans d'autres pays membres de l'OCDE. Bien qu'on ait établi des strates et des groupes d'UPE de variance pour le calcul des poids de rééchantillonnage afin de faciliter la comparaison des résultats du Canada à ceux d'autres pays participants, les 80 répliques pourraient ne pas être suffisantes étant donné le nombre nettement plus grand d'écoles dans l'échantillon canadien. Donc, on a également produit un ensemble de 1 000 poids bootstrap qu'il est recommandé d'utiliser pour l'analyse des données du PISA si l'on n'a pas l'intention de faire des comparaisons avec d'autres pays participants. Le PISA a été intégré dans le premier panel de l'EJET (cohorte d'élèves de 15 ans) et les participants au PISA ont également reçu les questionnaires de l'EJET (pour les élèves et pour les parents). Il est possible de relier les réponses aux questionnaires de l'EJET à l'ensemble de données du PISA et, le cas échéant, les poids bootstrap devraient être utilisés pour le calcul de la variance. Les programmes décrits dans le présent article ont été conçus au départ pour l'utilisation de la méthode BRR, mais permettent maintenant d'utiliser la méthode bootstrap.

IV. pvpisa

Cette commande STATA calcule le vecteur des coefficients estimés et la matrice des variances-covariances fondée sur les considérations qui précèdent pour toutes les commandes d'estimation (e-class) de STATA. La syntaxe de la commande est la suivante :

```
pvpisa varlist [weight] [if exp] [in range], cmd(string) pv(varlist)
rw(varlist) fays(real) [cmdops(string) brr uset dof(real) level(integer)]
```

`varlist`: Il s'agit d'un style de liste de variables de STATA contenant les variables *indépendantes* par rapport auxquelles on veut faire le conditionnement. Contrairement à d'autres commandes d'estimation, il ne faut pas inclure de variable dépendante pour commencer.

`weights`: pweights serait le choix pertinent. Tous les types de poids sont permis pour éviter les erreurs générées par des commandes particulières de STATA qui n'acceptent pas pweights. Entrez ici le poids d'*échantillon complet*, `w_fstuwt`, qui figure dans le fichier de données sur la lecture de l'EJET/PISA.[1] Si vous ne spécifiez aucun poids, le programme sera quand même exécuté, mais les résultats n'auront aucun sens.

`[if]` et `[in]`: Utilisez ces options de la même façon qu'avec toute autre commande STATA pour définir des sous-ensembles de données.

Les options sont les suivantes :
`cmd()`**:** Tapez ici la commande STATA que vous voulez exécuter : par exemple, `tobit`**.**

`cmdops()`**:** Tapez ici toutes les options que vous spécifieriez normalement avec la commande, par exemple `noconstant` pour `regress`**.** N'oubliez pas que certaines de ces options n'auront aucun sens si elles sont utilisées avec un scénario de calcul de la variance par rééchantillonnage. Il incombe à l'utilisateur de choisir les options appropriées. Seules les options qui affectent les estimations sont pertinentes. Les options en rapport avec la variance comme « `robust` » et « `cluster()` » n'auront aucun effet, parce que les matrices des covariances calculées par la commande d'estimation ne sont pas utilisées par `pvpisa`.

`brr`: Si l'on spécifie `brr`, la méthode BRR est utilisée et les poids BRR devraient être donnés dans `rw()`. Si l'on ne spécifie pas `brr`, il est supposé que la méthode utilisée est le bootstrap et les poids donnés dans `rw()` devraient être les poids bootstrap.

[1] À l'heure actuelle, nombre de routines d'estimation d'après des données de panel de STATA (commandes dites « `xt` ») n'acceptent pas les poids, ou acceptent uniquement des poids de « groupe », le groupe étant habituellement l'unité d'analyse, comme une personne. Donc, `pvpisa` ne peut pas être utilisée avec les données de l'EJET/PISA pour bon nombre de commandes pour données de panel classiques. Je cherche à développer des commandes distinctes qui prendront les poids d'élève pour ces estimateurs. À noter, toutefois, que l'on peut estimer les estimateurs à effets fixes et aléatoires (après transformation des données) par les simples MCO, ce qui permet d'utiliser `pvpisa`. Voir Green (2001) pour les transformations nécessaires.

`pv(varlist)` **:** Il s'agit de la liste de variables qui contient les valeurs plausibles, c'est-à-dire les variables dépendantes pour l'analyse. Les variables PV du PISA pour les compétences globales en lecture sont `pv1read pv2read pv3read pv4read pv5read`.

`rw(varlist)` **:** Il s'agit d'une liste de variables de poids de rééchantillonnage. Les poids BRR pour les données sur la lecture du PISA seraient entrés sous la forme : `w_fstr1-w_fstr80`. Les poids bootstrap, si l'option `brr` n'est pas spécifiée, seraient `BPSR1-BPSR1000`.

`fays(real)` **:** Entrez ici la constante à utiliser avec la variante de Fay de la méthode BRR. La valeur par défaut est 0.5 pour les données de l'EJET/PISA. Cette option n'est pertinente que si l'on spécifie `brr`.

`uset`**:** Cette option demande que l'on calcule les limites de confiance et les valeurs p en utilisant la loi t de Student avec le nombre de degrés de liberté donné par `dof()`. La valeur par défaut est l'utilisation de la loi normale standard.

`dof(real)` **:** Utilisez cette fonction si vous spécifiez `uset`. Il s'agit du nombre de degrés de liberté utilisés dans le calcul des limites de confiance et des valeurs p. Consultez la dernière section pour des précisions concernant le choix du nombre de degrés de liberté. La valeur par défaut est 80. Cette option n'a aucun effet si l'on ne spécifie pas `uset`.

`Level(integer)` **:** Il s'agit simplement de l'option STATA standard qui demande le niveau pour la construction des intervalles de confiance. La valeur par défaut est l'intervalle à 95 %.[2]

Il convient de souligner que la sortie contient uniquement les coefficients estimés, les erreurs-types, la statistique t ou z, les valeurs p et les limites de confiance. D'autres sorties, comme l'EQM ou la statistique R-carré, ne sont pas fournies. Aucune estimation de la vraisemblance n'est donnée si l'on utilise les commandes MLE (estimation du maximum de vraisemblance). Ces statistiques devraient être calculées en tenant compte du plan de sondage. Les commandes `test` et `predict` de STATA peuvent être utilisées après `pvpisa` comme avec toute autre commande d'estimation. Vous pouvez aussi obtenir le vecteur de coefficients en utilisant `e(b)` et la matrice des covariances des coefficients en utilisant `e(V)` après l'estimation si vous souhaitez utiliser `pvpisa` dans un programme.

[2] Il convient de noter que `level` peut habituellement être spécifié en tant qu'option dans la plupart des commandes d'estimation (e-class). La spécification de `level` dans l'option `cmdops()` n'aura aucun effet sur les limites de confiance estimées par `pvpisa`. Vous devez utiliser l'option `level` ici pour l'effet désiré.

Au moment de la rédaction de l'article, `pvpisa` était testée avec les commandes à équation unique suivantes:[3]

```
Regress  Ologit
tobit    Mlogit
qreg
probit
logit
oprobit
```

Exemple 1 :

Cet exemple utilise les données sur la lecture de l'EJET/PISA pour faire la régression des compétences en lecture sur le nombre total d'heures d'enseignement par année (`tothrs`) et l'effectif de la classe déclaré par l'élève (`st28q01`) pour les élèves de l'Alberta.

```
pvpisa tothrs st28q01 [pweight = w_fstuwt] if prov == 8, cmd("regress")
pv(pv1read pv2read pv3read pv4read pv5read) rw(w_fstr1-w_fstr80) brr uset
fays(0.5) level(90)
```

Voici la sortie.

```
Command Summary: regress pv1read pv2read pv3read pv4read pv5read tothrs st28q01 if prov == 8 ,
confidence level is 90
BRR specified: BRR weights assumed and BRR method used
estimates for pv1read complete
estimates for pv2read complete
estimates for pv3read complete
estimates for pv4read complete
estimates for pv5read complete
There are 5 plausible values
There are 80 replciates
The confidence value is 90
t distribution requested degrees of freedom 80
------------------------------------------------------------------------------
p.v.         |      Coef.   Std. Err.      t    P>|t|     [90% Conf. Interval]
-------------+----------------------------------------------------------------
tothrs       |   .0028195   .0213395     0.13   0.895    -.0326921    .0383311
st28q01      |   2.520135   .3262991     7.72   0.000     1.977133    3.063137
_cons        |   485.8485   22.92675    21.19   0.000     447.6955    524.0015
------------------------------------------------------------------------------
```

Exemple 2 :

Cet exemple montre l'estimation des coefficients d'un modèle probit ordonné pour estimer la probabilité d'atteindre l'un de cinq niveaux de compétence possibles lors du test de compétence en lecture.

```
local j 1
while `j' <= 5 {
```

[3] Des commandes à équations multiples sont considérées pour les moindres carrés en deux étapes, les régressions apparemment non reliées (seemingly unrelated regression), le modèle de sélection de Heckman et le probit bivarié

```
      g y`j' = .
      qui {
      replace y`j' = 0 if pv`j'read <= 334.75
      replace y`j' = 1 if pv`j'read > 334.75 & pv`j'read <= 407.67
      replace y`j' = 2 if pv`j'read > 407.67 & pv`j'read <= 480.18
      replace y`j' = 3 if pv`j'read > 480.18 & pv`j'read <= 552.89
      replace y`j' = 4 if pv`j'read > 552.89 & pv`j'read <= 625.61
      replace y`j' = 5 if pv`j'read > 625.61
      }
      local j = `j' + 1

}
pvpisa tothrs st28q01 [pweight = w_fstuwt], cmd("oprobit") pv(y1-y5)
rw(w_fstr1-w_fstr80) brr uset fays(0.5)
```

Voici la sortie.

```
Command Summary: oprobit y1 y2 y3 y4 y5 tothrs st28q01  ,
confidence level is 95
BRR specified: BRR weights assumed and BRR method used
estimates for y1 complete
estimates for y2 complete
estimates for y3 complete
estimates for y4 complete
estimates for y5 complete
There are 5 plausible values
There are 80 replciates
The confidence value is 95
t distribution requested degrees of freedom 80
------------------------------------------------------------------------------
p.v.         |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
tothrs       |  -.0001164   .0000907    -1.28   0.203    -.0002969    .0000642
st28q01      |   .0285507   .0022075    12.93   0.000     .0241576    .0329439
_cut1        |  -1.478013   .1153145   -12.82   0.000    -1.707496   -1.24853
_cut2        |  -.7588495   .1155712    -6.57   0.000    -.9888435   -.5288555
_cut3        |  -.0304934   .1113955    -0.27   0.785    -.2521775    .1911908
_cut4        |   .7283558   .1127004     6.46   0.000     .5040748    .9526368
_cut5        |   1.574229   .1106444    14.23   0.000     1.354039    1.794418
------------------------------------------------------------------------------
```

V. kpvpisa

Ce programme utilise une version modifiée de la commande `kdensity` de STATA pour calculer l'estimateur de la densité par la méthode du noyau pour la densité de compétence de la population. La modification permet d'utiliser les iweights, qui sont rééchelonnés de sorte que leurs sommes soient égale à 1 avant l'estimation. Pour les données d'enquête, l'option iweights est plus appropriée.

L'estimation par la méthode du noyau a été recommandée dans le passé pour l'analyse exploratoire des données (voir le traité publié par Silverman, 1986). Aujourd'hui, elle est utilisée de plus en plus fréquemment dans l'analyse par régression non paramétrique (Yatchew, 1998; Blundel et Duncan, 1998; DiNardo et Tobias, 2001). L'estimateur par la méthode du noyau est également utilisé dans les études d'appariement (Smith et Todd, 2001). Un domaine récent d'application a été l'analyse de l'inégalité des salaires et des gains (DiNardo, Fortin et Lemieux, 1996; Daly et Valleta, 2000). L'estimateur de la densité par la méthode du noyau, en utilisant des

poids de sondage w_i normalisés de sorte que leurs sommes soient égales à 1, est défini comme suit :

$$\hat{f}(y_0) = \sum_i \frac{w_i}{h} K\left(\frac{y_0 - y_i}{h}\right) \qquad (2)$$

Ici, on estime la fonction au point y_0. La constante h est la largeur de bande ou constante de lissage, qui est choisie par l'analyste. K est la fonction noyau, qui applique des poids de plus en plus faibles aux données à mesure que celles-ci s'écartent du point y_0. L'estimateur de la densité par la méthode du noyau est une généralisation de l'histogramme bien connu , celui-ci étant un cas spécial de l'estimateur de la densité par la méthode du noyau avec une fonction noyau choisie de façon appropriée. DiNardo et Tobais (2001) donnent une discussion excellente et compréhensible de l'estimateur de la densité par la méthode du noyau.

Comme tout autre estimateur fondé sur des données provenant d'une enquête par sondage, l'estimateur ponctuel $\hat{f}(y_0)$ présente une variation d'échantillonnage. Une discussion ouverte se poursuit à l'heure actuelle sur l'estimation de la variance au moyen de l'estimateur de la densité par la méthode du noyau. Le bootstrap a été utilisé dans de nombreuses applications. Buskirk et Lohr (2003) discutent des propriétés asymptotiques de l'estimateur de la densité par la méthode du noyau pondéré conformément au plan de sondage et calcule les limites de confiance en s'appuyant sur des conditions de normalité asymptotiques. Le programme `kpvpisa` applique simplement l'équation (2) en utilisant la commande `kdensity` et calcule les limites de confiance fondées sur les poids BRR ou sur les poids bootstrap. La variance totale, y compris l'erreur de mesure associée à l'utilisation des valeurs plausibles, est intégrée telle qu'indiquée dans l'équation (1). La syntaxe est la suivante :

```
kpvpisa varlist [weight] [if exp] [in range], at(varname) rw(varlist)
fays(real) [brr uset dof(real) bandwidth(real) kernel(string) level(integer)]
```

`varlist`: Entrez la liste de noms de variable de valeur plausible.

`weight`: Le programme accepte les aweights, les fweights ou les iweights. Les aweights et les fweights peuvent correspondre à la commande `kdensity` originale. Par contre, la commande modifiée qui appelle `kpvpisa` permet d'utiliser des iweights qui sont rééchelonnés de sorte que leurs sommes soient égales à l'unité. Par conséquent, les utilisateurs devraient utiliser les iweights. Entrez le poids d'échantillon complet ici. Pour les données sur la lecture du PISA, il s'agit de `w_fstuwt`. Il convient de souligner que les résultats n'auront aucun sens si la variable de poids est omise.

`[if]` et `[in]`: Il s'agit d'options standard de STATA pour la production de sous-ensembles de données qui utilisent la syntaxe standard.

`at(varlist)`: L'option `at()` de kdensity *doit* être utilisée parce qu'il faut calculer la moyenne des estimations de la densité sur les J valeurs plausibles et pouvoir représenter

graphiquement les résultats. Entrez le nom d'une variable qui contient des valeurs auxquelles il faut estimer la densité. Un exemple est donné plus bas. Les *P* valeurs de la variable `at()` devraient se trouver parmi les *P* observations de l'ensemble de données, *P* étant le nombre de points auxquels la densité est estimée. Utilisez `sort` si ce n'est pas le cas.

`rw(varlist)` **:** Ici, il faut inscrire les noms des variables de poids de rééchantillonnage. Les poids BRR pour les compétences globales en lecture sont `w_fstrwt1-wfstrwt80`. Les poids bootstrap pour les compétences globales en lecture sont `BPSR1-BPSR1000`.

`brr` **:** Utilisez cette option pour la méthode BRR. Les poids de rééchantillonnage entrés dans `rw()` doivent être les poids BRR. L'option par défaut est la méthode bootstrap, auquel cas les poids entrés dans `rw()` doivent être les poids bootstrap.

`uset` **:** Utilisez cette option pour demander que les limites de confiance soient fondées sur la loi t de Student. L'option par défaut est la loi normale standard. Il n'existe pas encore d'option permettant d'utiliser la loi bootstrap quand on ne choisit pas l'option `brr`.

`dof(real)` **:** Cette option doit être utilisée si l'option `uset` est spécifiée. Elle permet à l'utilisateur de fixer le nombre de degrés de liberté. L'option par défaut est l'utilisation de la formule donnée à l'annexe A, qui tient compte de l'utilisation des valeurs plausibles. Cette option n'a aucun effet si `uset` n'est pas spécifiée.

`fays(real)` **:** Entrez la valeur de la constante de Fay. Pour les données du PISA, elle est de 0.5.

`bandwidth(real)`: Il s'agit de la largeur de bande *h* qui est utilisée pour lisser l'estimation de la densité. Si elle est omise, la valeur par défaut est l'estimateur « rule of thumb » décrit par Silverman (1986).

`kernel(string)` **:** Entrez ici l'option pour la fonction noyau (voir `kdensity`) indiquée dans le manuel STATA. L'option par défaut est le noyau gaussien (normal).

`level(integer)` **:** C'est l'option est l'option STATA standard pour la spécification du niveau de signification pour les intervalles de confiance. Ce niveau sera utilisé pour estimer les limites de confiance pour l'estimation de la densité.

`kpvpisa` place les variables suivantes dans l'ensemble de données. Elles sont placées parmi les *P* premières observations, *P* étant le nombre de valeurs pour la variable `at()`.

`dpv` **:** Moyenne des *J* estimations de la densité des valeurs plausibles.

`vdpv`: Variance totale estimée de dpv.

`dpvlo` **:** Limite de confiance inférieure de l'estimation de la densité dpv.

`dpvhi` : Limite de confiance supérieure de l'estimation de la densité dpv.

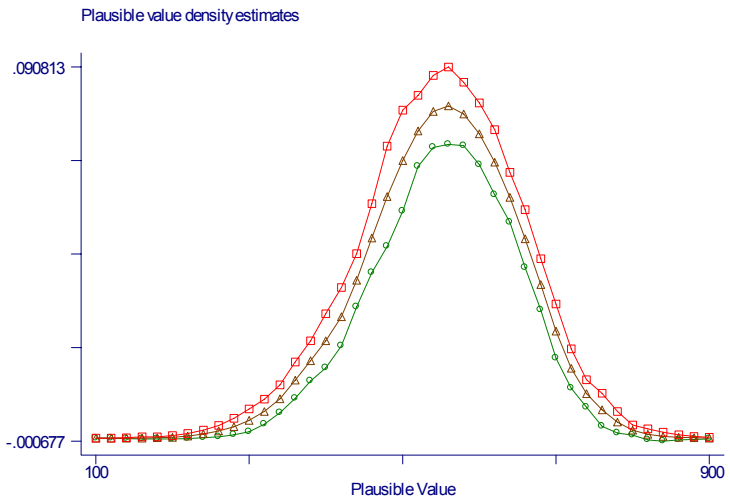
`kpvpisa` tracera les graphiques de `dpvlo`, `dpv` et `dpvhi` en fonction de la variable choisie pour l'option `at()` afin de donner la forme de la fonction de densité et la marge d'erreur.

Exemple 3 :

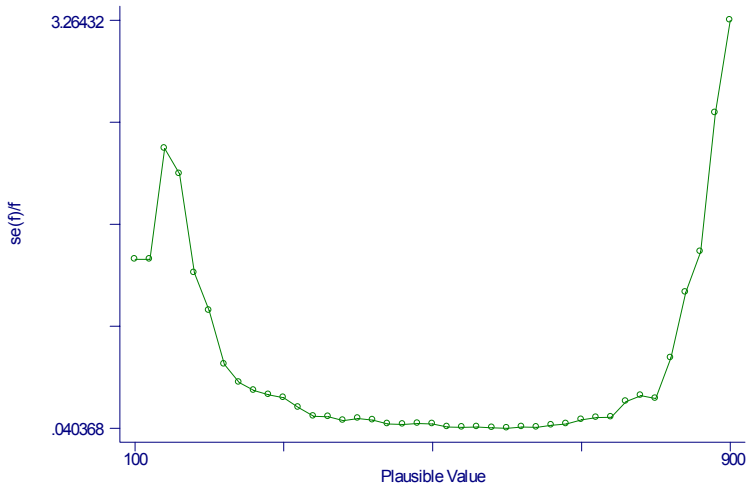
Cette commande estime la densité des compétences en lecture pour l'Alberta :

```
kpvpisa pv1read pv2read pv3read pv4read pv5read [aweight = w_fstuwt] if
prov==8, at(xvalue) rw(w_fstr1-w_fstr80) brr uset fays(0.5) level(95)
```

Suivent les estimations de la densité et des limites de confiance :



L'examen du graphique qui précède semble indiquer que l'estimation la moins exacte de la densité s'observe à proximité du mode de la distribution, où l'intervalle de confiance est le plus large. On observe ce profil parce que c'est autour du mode que les estimations de la densité sont les plus grandes et, par conséquent, que la largeur de l'intervalle de confiance, qui est une fonction de l'erreur-type estimée, est le plus grand. Cependant, une perspective différente se dégage si l'on examine l'erreur-type de la densité en fonction de la densité proprement dite. La figure qui suit montre un tracé de $se(\hat{f})/\hat{f}$, c'est-à-dire le coefficient de variation estimé (c.v.). Ici, nous constatons que c'est dans les queues de la distribution que l'erreur-type estimée de la densité relativement à la grandeur de l'estimation de la densité est la plus importante, comme on s'y attendrait.



Exemple 4 :

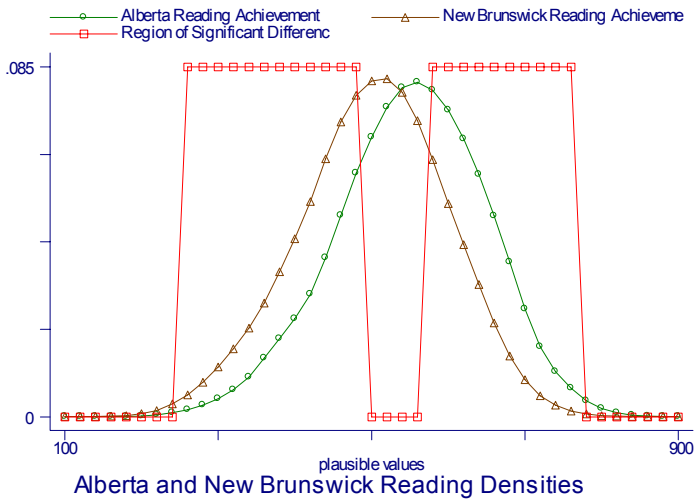
Ce fichier do illustre une utilisation type de `kpvpisa`. Ici, nous estimons la densité des compétences en lecture pour deux provinces, à savoir l'Alberta et le Nouveau-Brunswick. Naturellement, nous aimerions savoir dans quelle mesure la différence *n'est pas* due au hasard. Pour obtenir la réponse, nous comparons les limites de confiance de chaque estimation de la densité. Si les limites ne se chevauchent pas, alors la différence est statistiquement significative.[4] Le fichier do produit un tracé des densités pour l'Alberta et le Nouveau-Brunswick et une fonction indicatrice qui est non nulle lorsque les deux probabilités des compétences sont significativement différentes l'une de l'autre.

```
clear
set memory 50m
set matsize 800
set logtype text

use "pisadata.dta"

      * create 50 xvalue variable at which densities are estimated
      local i 100
      local count 1
      g xvalue = . in 1/50
      while `i' <= 900 {
            qui replace xvalue = `i' in `count'
            local I = `I' + 20
            local count = `count' + 1
      }
```

[4] Notons qu'aucune correction de Bonferroni n'est faite dans ce cas, parce que je trace simplement les emplacements où les densités sont significativement différentes *à ce point-là*. Je ne prononce pas d'énoncé global comme : « Les densités diffèrent à un ou plusieurs emplacements, par conséquent les densités sont différentes. » Autrement dit, ces tests individuels ne sont pas utilisés pour déclarer qu'une densité est significativement différente d'une autre.

```
kpvpisa pv1read pv2read pv3read pv4read pv5read [aweight = w_fstuwt] if
prov==8, at(xvalue) rw(w_fstr1-w_fstr80) fays(0.5) level(95)

g ab_d = dpv
g ab_lo = dpvlo
g ab_hi = dpvhi

drop dpv dpvlo dpvhi vdpv

kpvpisa pv1read pv2read pv3read pv4read pv5read [aweight = w_fstuwt] if
prov==3, at(xvalue) rw(w_fstr1-w_fstr80) fays(0.5) level(95)

g nb_d = dpv
g nb_lo = dpvlo
g nb_hi = dpvhi

g byte I = ab_lo > nb_hi | nb_lo > ab_hi

g scale = 0.085*I

label variable xvalue "plausible values"
label variable ab_d "Alberta Reading Achievement"
label variable nb_d "New Brunswick Reading Achievement"
label variable scale "Region of Significant Difference"
graph ab_d nb_d scale xvalue, c(lll) ti("Alberta and New Brunswick Reading
Densities")
```

Le tracé est le suivant :



VI. Mot de la fin

Les programmes `pvpisa` et `kpvpisa` sont fournis aux chercheurs des CDR sous forme de versions « bêta » dans l'espoir qu'elles leur seront utiles. Les programmes ont été écrits pour la version 7 de STATA et n'ont pas été testés sur la version 8. Un commentaire sur l'utilisation de la loi de Student pour l'inférence est nécessaire. Pour l'analyse fondée sur l'échantillon complet de

l'EJET/PISA, il est probable que les résultats diffèrent peu que l'on choisisse la loi de Student ou la loi standard normale. L'utilisateur qui souhaite utiliser la loi de Student doit choisir le nombre de degrés de liberté soigneusement. Idéalement, ce nombre devrait tenir compte de l'utilisation des valeurs plausibles, ce qui est fait dans `kpvpisa`, mais non dans `pvpisa` (voir l'annexe A). En outre, le nombre de degrés de liberté sera généralement plus faible si l'utilisateur produit des sous-ensembles de données de telle façon qu'il existe des strates ne contenant aucune UPE dans le sous-échantillon sur lequel est fondée l'inférence. Ceci se produira si l'analyse est limitée à une province, par exemple. Dans de tels cas, il est conseillé aux utilisateurs d'utiliser les options `uset` et `dof()` pour explorer la sensibilité de leurs résultats aux variations du nombre de degrés de liberté. Pour un calcul plus précis du nombre de degrés de liberté, il est conseillé à l'utilisateur de communiquer avec le gestionnaire des données d'enquête qu'il utilise. En ce qui concerne l'option bootstrap, il n'est pas encore possible d'utiliser la loi bootstrap pour le calcul des intervalles de confiance dans cette version, comme on peut le faire dans d'autres applications, telles que les programmes STATA décrits ailleurs dans le présent numéro.

Bibliographie

Adams, R and M. Wu (2002) *PISA 2000 Technical Report*, Paris, OECD

Blundel R. and A Duncan (1998) "Kernel Regression in Empirical Microeconomics", *Journal of Human Resources* 33(1):62-87

Buskirk, T. D. and S. Lohr (2003) "Asymptotic Properties of Kernel Density Estimation with Complex Survey Data" Arizona State University technical paper.

Daly, M. C. and R. G. Valletta (2000) "Inequality and Poverty in the United States: The Effects of Changing Family Behaviour and Rising Wage Dispersion" Working Paper, Federal Reserve Bank of San Francisco.

DiNardo, J. and J. L. Tobias (2001) "Nonparametric Density and Regression Estimation", *Journal of Economic Perspectives* 13(4):11-28

DiNardo, J., N. M. Fortin and T. Lemieux (1996) "Labor Market Institutions and the Distribution of Wages 1973-1992: A Semiparametric Approach, *Econometrica,* 64(5):1001-1044

Gonsalez, Smith and Sibberns (1998) *User Guide for the TIMSS International Database: Final Year of Secondary School, 1995 Assessment*, International Association for the Evaluation of Education Achievement and TIMSS International Study Centre, Boston College

Green, William H. (2001) *Econometric Analysis 4th Edition*, Prentice-Hall, Upper Saddle River, New Jersey.

Mislevey, R. J. (1991) "Randomization based inference about examinees in the

estimation of item parameters" *Psychometrika* 56:177-196

Silverman, B. (1986) *Density Estimation for Statistics and Data Analysis*, London: Chapman and Hall.

Smith, J. and Todd, P. (2001) "Does Matching Overcome LaLonde's Critique of Nonexperimental Estimators?" Unpublished working paper, University of Maryland.

Yatchew, A. (1998) "Nonparametric Regression Techniques in Economics" *Journal of Economic Literature* 36:669-721

NOTA :

Le produit de données est fourni «tel quel», et Statistique Canada ne donne aucune garantie explicite ou implicite, qui comprend une garantie de commerciabilité et d'adaptation à une fin particulière, mais ne se limite pas à cette garantie. En aucune circonstance, Statistique Canada ne sera tenu responsable des dommages indirects, réels, conséquents ou de tout autre dommage, quelle qu'en soit la cause, liés à l'utilisation du produit de données.

Annexe A

Calcul du nombre de degrés de liberté en tenant compte des valeurs plausibles :

Cette formule est implantée dans `kpvpisa`, mais non dans `pvpisa`. Cette situation est due au fait que la commande `estimates post` de STATA est utilisée et qu'elle permet des nombres de degrés de liberté scalaires mais non des nombres qui varient avec les paramètres estimés. Voici toutefois la formule qui est utilisée dans `kpvpisa`. Sachant

$$\text{tot var}(\hat{\theta}) = (1/J)\sum_j \text{var}(\hat{\theta}_j) + (1+1/J)(1/(J-1))\sum_j (\hat{\theta}_j - \hat{\theta})^2 ,$$

définir $f_J = (1+1/J) * B_J / tot\,\text{var}$, où $B_J = (1/(J-1))\sum_j (\hat{\theta}_j - \hat{\theta})^2$. f_J est simplement la proportion de la variance totale attribuable à l'erreur de mesure due à l'utilisation des valeurs plausibles. Soit *d* le nombre de degrés de liberté qu'il faudrait utiliser si le paramètre de capacité latente θ_n pour l'élève *n* (estimé en utilisant les valeurs plausibles) était observé. Pour les données du PISA, cette valeur varie selon le pays et ne peut excéder 80. Le calcul du nombre de degrés de liberté est

$$dof = \frac{1}{\frac{f_J^2}{J-1} + \frac{(1-f_J)^2}{d}} .$$[5]

[5] La discussion de cette formule figure dans le rapport technique sur le PISA 2000, Adams et Wu (2002).

Annexe B

pvpisa

```
program define pvpisa, eclass byable(recall)
        version 7.0

      syntax varlist(numeric) [pweight fweight aweight iweight] [if] [in], cmd(string) [cmdops(string)] pv(varlist numeric) rw(varlist numeric)
fays(real) [brr uset dof(integer 80) level(integer 95)]

        di in gr "Command Summary: `cmd' `pv' `varlist' `if' `in', `cmdops'"
        di in gr "confidence level is `level'"

        if "`brr'" != "" {

                di in ye "BRR specified: BRR weights assumed and BRR method used"

                /* compute sampling variance for `ith' plausible value */
                local i 1 /* i indexes plausible values */
                foreach pvname in `pv' {

                        /* compute full sample estimate of beta */
                qui `cmd' `pvname' `varlist' [`weight'`exp'] `if' `in', `cmdops'
                        matrix b`i'0 = e(b)'

                        matrix vbrr`i' = (b`i'0 - b`i'0)*(b`i'0 - b`i'0)' /*This is a quick cheat way to get a KxK
                                                                                0 matrix. That is, initialize the cov
                                                                                matrix to 0. */

                        local reps 1 /* reps indexes the replicates */
                        /*start BRR - accumulate sum of squares using the `i'th plausible values */
                        foreach wname in `rw' {
                                qui `cmd' `pvname' `varlist' `if' `in' [`weight' = `wname'], `cmdops'
                                matrix vbrr`i' = vbrr`i' + (e(b)' - b`i'0)*(e(b)' - b`i'0)'
                                local reps = `reps' + 1
                        }
                        matrix vbrr`i' = (1/(`reps'*(1 - `fays')^2))*vbrr`i'                /* compute the fays BRR variance-covariance matrix */
                        di in gr "estimates for `pvname' complete"
                local i = `i' + 1

                }

                /* ensure counters are set properly */
              local i = `i' - 1
                local reps = `reps' - 1

                /* Begin computing Variance and confidence limits of estimates taking into account plausible value imputation and survey desing */
                /* compute average pv estiamte */
                local z 1
                matrix b0 = J(rowsof(b10),1, 0)
                while `z' <= `i' {
                        matrix b0 = b0 + (1/`i')*b`z'0
                        local z = `z' + 1
                }

                /* compute elements for total VCE */
                /* U is average VCE of plausible value VCEs */
```

```
local z 1
matrix U = J(rowsof(vbrrl), colsof(vbrrl),0)
local z 1
while `z' <= `i' {
        matrix U = U + (1/`i')*(vbrr`z')
        local z = `z' + 1
}

/* B is VCE of plausible value estiamtes */
matrix B = J(rowsof(vbrrl), colsof(vbrrl), 0)
local z 1
while `z' <= `i' {
        local ii 1      /* These lines are meant to prevent a division by 0 error
                                that would occur if only 1 plausible value were used.
                                this enables the user to run BRR on non plausible value
                                d.v.s simply by listing the d.v. in the pv() option. */
        if `i' - 1 != 0 {
                local ii = (1/(`i'-1))
        }
        matrix B = B + `ii'*(b`z'0 - b0)*(b`z'0 - b0)'
local z = `z' + 1
}

/* estimate total variance covariance matrix */
matrix vbrrpv = U + (1 + (1/`i'))*B


matrix b0 = b0'

di in green "There are `i' plausible values"
di in green "There are `reps' replciates"
di in green "The confidence value is `level'"

if "`uset'" != "" {
        di in ye "t distribution requested degrees of freedom `dof'"
        estimates post b0 vbrrpv, dep("p.v.") dof(`dof')
}
else estimates post b0 vbrrpv, dep("p.v.")
estimates display, level(`level')

} /* end execution of BRR routine */

if "`brr'" == "" {

        di in ye "BRR not specified: Bootstrap weights assumed and Bootstrap method used"

        /* compute sampling variance for `ith' plausible value */
        local i 1 /* i indexes plausible values */

        foreach pvname in `pv' {

                /* compute full sample estimate of beta */
        qui `cmd' `pvname' `varlist' [`weight'`exp'] `if' `in', `cmdops'
                matrix b`i'0 = e(b)'


                matrix b`i'avg = b`i'0 - b`i'0    /* initialize average beta vector to zero */
                local reps 1
```

```
    /*compute average bs beta */

    foreach wname in `rw' {
    qui `cmd' `pvname' `varlist' [`weight'= `wname'] `if' `in', `cmdops'
            matrix b`i'avg = b`i'avg + e(b)'
    local reps = `reps' + 1
    }
    matrix b`i'avg = (1/`reps')*b`i'avg

    matrix vbrr`i' = (b`i'0 - b`i'0)*(b`i'0 - b`i'0)' /*This is a quick cheat way to get a KxK
                                                        0 matrix. That is, initialize the cov
                                                          matrix to 0. */

    local reps 1 /* reps indexes the replicates */
    /*start BRR - accumulate sum of squares using the `i'th plausible values */
    foreach wname in `rw' {
          qui `cmd' `pvname' `varlist' `if' `in' [`weight' = `wname'], `cmdops'
          matrix vbrr`i' = vbrr`i' + (e(b)' - b`i'avg)*(e(b)' - b`i'avg)'
    local reps = `reps' + 1
    }
    matrix vbrr`i' = (1/(`reps'-1))*vbrr`i'          /* compute the bootstrap variance-covariance matrix */
    di in gr "estimates for `pvname' complete"
local i = `i' + 1

}

  /* ensure counters are set properly */
local i = `i' - 1
  local reps = `reps' - 1

  /* Begin computing Variance and confidence limits of estimates taking into account plausible value imputation and survey desing */
  /* compute average pv estiamte */
  local z 1
  matrix b0 = J(rowsof(b10),1, 0)

  while `z' <= `i' {
        matrix b0 = b0 + (1/`i')*b`z'0
  local z = `z' + 1
  }

  /* compute elements for total VCE */
  /* U is average VCE of plausible value VCEs */

  local z 1
  matrix U = J(rowsof(vbrr1), colsof(vbrr1),0)
  local z 1

  while `z' <= `i' {
        matrix U = U + (1/`i')*(vbrr`z')
  local z = `z' + 1
  }

  /* B is VCE of plausible value estiamtes */
  matrix B = J(rowsof(vbrr1), colsof(vbrr1), 0)
  local z 1
  while `z' <= `i' {
        local ii 1        /* These lines are meant to prevent a division by 0 error
                             that would occur if only 1 plausible value were used.
                             this enables the user to run BRR on non plausible value
```

```
    if `i' - 1 != 0 {
        local ii = (1/(`i'-1))
    }
    matrix B = B + `ii'*(b`z'0 - b0)*(b`z'0 - b0)'
local z = `z' + 1
}

/* estimate total variance covariance matrix */
matrix vbrrpv = U + (1 + (1/`i'))*B

matrix b0 = b0'

di in green "There are `i' plausible values"
di in green "there are `reps' replciates"
di in green "The confidence value is `level'"

if "`uset'" != "" {
    di in ye "t distribution requested degrees of freedom `dof'"
    estimates post b0 vbrrpv, dep("p.v.") dof(`dof')
}
else estimates post b0 vbrrpv, dep("p.v.")
estimates display, level(`level')

}

end
```

kpvpisa.ado

```
program define kpvpisa, byable(recall)
     version 7.0
     syntax varlist(numeric) [iweight fweight aweight] [if] [in],  at(varname numeric)   [ brr uset dof(integer 80) bandwidth(real 10)]
   [kernel(string)] rw(varlist numeric) fays(real) [level(integer 90)]

        di in green "Confidence level is: " "`level'%"

        label variable `at' "Plausible Value"

        * Obtain the users choice of kernel function and write it to macro K
        local K " "
        if "`kernel'" != " " {
             local K = "`kernel'"
        }

        * Obtain the users choice of bandwidth and enter it as the width option in macro h
        local h " "
        if `bandwidth' != 10  {
             local h = "width(`bandwidth')"
        }

        *count number of replicates
        local reps 0
        foreach rep in `rw' {
             local reps = `reps' + 1
        }

        *count number of plausible values
        local M 0
        foreach m in `varlist' {
             local M = `M' + 1
        }

        *do full sample densities
        qui g dpv = 0
        di in gr "computing full sample density estimates:"
        local j 1
        foreach pv in `varlist' {
             di in gr "command summary: kdens_ `pv' [`weight'`exp'] `if' `in', at(`at') g(xpoint d`j'0) `K' `h' nograph "

             tempvar d`j'0
             kdens_ `pv' [`weight'`exp'] `if' `in', at(`at') g(xpoint `d`j'0') nograph
             qui replace `d`j'0' = `d`j'0'*r(scale)   /* The scale return from kdens_ ensures densities itegrate to 1 */
             drop xpoint
             qui replace `d`j'0' = 0.000000000000000`j' if `d`j'0' == 0 /* this line prevents any division by 0 problem later */
             qui replace dpv = dpv + (1/`M')*`d`j'0'
             local j = `j' + 1
        }

        /* compute Ustar, the average sampling variance where sampling variance is either
           the BRR variance or the Bootstrap variance */

        tempvar Ustar
        qui g `Ustar' = 0

        if "`brr'" != "" {
```

```
di in ye "BRR method requested: replicate weights assumed to be BRR"
*do BRR variance esitmates for pv densities
di in gr " "
di in gr  "doing BRR replicates now for:"
local j 1
foreach pv in `varlist' {
        tempvar U`j'
        local k 1
        qui g `U`j'' = 0
        di in gr "`pv'"
        foreach repwt in `rw' {
                tempvar d`j'`k'
                kdens_ `pv' [`weight'=`repwt'] `if' `in', at(`at') g(xpoint `d`j''`k'')`K' `h' nograph
                qui replace `d`j''`k'' = `d`j''`k''*r(scale)
                drop xpoint
                qui replace `d`j''`k'' = 0.000000000000000`k' if `d`j''`k'' == 0
                qui replace `U`j'' = `U`j'' + (1/(`reps'*(1-`fays')^2))*((`d`j''`k'' - `d`j'0')^2)
                drop `d`j''`k'' /* drop now to save room */
                local k = `k' + 1
        }
        qui replace `Ustar' = `Ustar' + (1/`M')*`U`j''
        drop `U`j'' /* drop now to save room */
        local j = `j' + 1
}
}
di
if "`brr'" == "" {
        di in ye "BRR not specified: replicate weights assumed to be bootstrap weights"
        tempvar Ustar
        qui g `Ustar' = 0
        local j 1
        foreach pv in `varlist' {
                /* accumulate average density results */
                di in gr "accumulating average density results for pv `j'"
                tempvar davg`j'
                local k 1
                qui g `davg`j'' = 0
                foreach repwt in `rw' {
                        tempvar d`j'`k'
                        kdens_ `pv' [`weight'=`repwt'] `if' `in', at(`at') g(xpoint `d`j''`k'')`K' `h' nograph
                        drop xpoint
                        qui replace `d`j''`k'' = `d`j''`k''*r(scale)
                        qui replace `davg`j'' = `davg`j'' + (1/`reps')*`d`j''`k''
                        drop `d`j''`k''
                        local k = `k' + 1
                }
                di in gr "average density for pv `j' done"
                /* now accumulate variance of jth plausible value density estimate */
                di in gr "doing bootsttrap variance estitmate for pv `j'"
                tempvar U`j'
                qui g `U`j'' = 0
                local k 1
                foreach repwt in `rw' {
                        tempvar d`j'`k'
                        kdens_ `pv' [`weight'=`repwt'] `if' `in', at(`at') g(xpoint `d`j''`k'')`K' `h' nograph
                        qui replace `d`j''`k'' = `d`j''`k''*r(scale)
                        drop xpoint
                        qui replace `d`j''`k'' = 0.000000000000000`k' if `d`j''`k'' == 0
                        qui replace `U`j'' = `U`j'' + (1/(`reps'-1))*((`d`j''`k'' - `davg`j'')^2)
```

```
        drop `d`j''`k'' /* drop now to save room */
        local k = `k' + 1
    }
    qui replace `Ustar' = `Ustar' + (1/`M')*`U`j''
    drop `U`j'' /* drop now to save room */
    local j = `j' + 1
  }
}
di

local j 1
tempvar Bm
qui g `Bm' = 0
while `j' <= `M' {
    qui replace `Bm' = `Bm' + (1/(`M'-1))*(`d`j'0' - dpv)^2
    local j = `j' + 1
}

g vdpv = `Ustar' + (1 + (1/`M'))*`Bm'

*calculate dof.
tempvar fm dof
g `fm'=(1 + (1/`M'))*`Bm'/vdpv
g `dof' = 1/((`fm'^2/(`j'-2)) + (((1-`fm')^2)/(`reps')))

local talpha = (1 - (`level'/100))/2 /* needed because invttail function is pr(T>t) */
local nalpha = 1 - `talpha' /* needed because the invnorm function is pr(Z<z) */
if "`uset'" != "" {
    di in ye "student's t distribution is used to compute confidence interval"
    g dpvlo = dpv - invttail(`dof', `talpha')*sqrt(vdpv)
    g dpvhi = dpv + invttail(`dof', `talpha')*sqrt(vdpv)
}

if "`uset'" == "" {
    di in ye "standard normal distribution is used to compute confidence interval"
    g dpvlo = dpv - invnorm(`nalpha')*sqrt(vdpv)
    g dpvhi = dpv + invnorm(`nalpha')*sqrt(vdpv)
}

graph dpvlo dpv dpvhi `at', c(lll) t1("Plausible value density estimates")

end
```

kdens_.ado

```
/* This program modifies the kdensity command supplied with Stata version 7.0
   to allow iweights. The iweights are scaled to sum to one.
   This program is called by kpvpisa
*/
*! version 2.3.6  26jun2000
program define kdens_, rclass
      version 6.0

      syntax varname [if] [in] [fw aw iw] [, /*
            */ Generate(string) N(integer 50) /*
            */ Width(real 0.0) noGRaph noDENsity /*
            */ BIweight COSine EPan GAUss RECtangle PARzen /*
            */ TRIangle Symbol(string) Connect(string) /*
            */ Title(string) AT(varname) NORmal STUd(int 0) * ]

      if "`at'"!="" & `n'!=50 {
            di in red "may not specify both the at() and n() options"
            exit 198
      }

      local ix "`varlist'"
      local ix1: variable label `ix'
      if "`ix1'"=="" {
            local ix1 "`ix'"
      }

      local gen "`generat'"

      local kflag = ( ("`epan'"' != `""') + ("`biweigh'"' != `"") + /*
                  */ ("`triangl'"' != `"") + ("`gauss'"' != `"") + /*
                  */ ("`rectang'"' != `"")  + ("`parzen'"' != `""'))
      if `kflag' > 1 {
            di in red "only one kernel may be specified"
            exit 198
      }

      if "`biweigh'"'      != `""' { local kernel="biweight"   }
      else if "`cosine'"'  != `""' { local kernel="Cosine"     }
      else if "`triangl'"' != `""' { local kernel="Triangle"   }
      else if "`gauss'"'   != `""' { local kernel="Gaussian"   }
      else if "`rectang'"' != `""' { local kernel="Rectangular"}
      else if "`parzen'"'  != `""' { local kernel="Parzen"     }
      else                         { local kernel="Epanechnikov" }

      marksample use
      qui count if `use'
      if r(N)==0 { error 2000 }

      tokenize `gen'
      local wc : word count `gen'
      if `wc' {
            if `wc' == 1 {
                  if "`at'" == `""' {
                        error 198
                  }
                  confirm new var `1'
                  local yl `"`1'"'
```

```
        local xl `"`at'"'
        local nsave 1
    }
    else {
            if `wc' != 2 { error 198 }
            confirm new var `1'
            confirm new var `2'
            local xl  `"`1'"'
            local yl  `"`2'"'
            local nsave 2
        }
    }
    else {
        if `"`graph'"'!=`""' {
                di in bl /*
*/ `"no graph requested and did not request results be saved; no action taken"'
                exit
        }
        local xl   `"x"'
        local yl   `"Density"'
        local nsave 0
    }

    tempvar d m z y
    qui gen double `d'=.
    qui gen double `y'=.
    qui gen double `z'=.
    qui gen double `m'=.

    if `"`at'"' != `""' {
            qui count if `at' != .
            local n = r(N)
            qui replace `m' = `at'
            local srtlst : sortedby
            tempvar obssrt
            gen `obssrt' = _n
            sort `m' `obssrt'
    }
    else {
            if `"`n'"'!= `""' {
                    if `n' <= 1 { local n = 50 }
                    if `n' > _N {
                            local n = _N
                            noi di in gr `"(n() set to "' `n' `")"'
                    }
            }
    }
    if "`weight'" != "" {
            tempvar tt
            qui gen double `tt' `exp' if `use'
            qui summ `tt', meanonly
            if "`weight'" == "aweight" {
                    qui replace `tt' = `tt'/r(mean)
            }
            if "`weight'" == "iweight" {
                    qui replace `tt' = `tt'/r(sum)
            }
```

```
}
else {
        local tt = 1
}

quietly summ `ix' [aweight`exp'] if `use', detail
local nmean = r(mean)
local nsig = r(Var)

tempname wwidth
scalar `wwidth' = `width'
if `wwidth' <= 0.0 {
        scalar `wwidth' = min( sqrt(r(Var)), (r(p75)-r(p25))/1.349)
        scalar `wwidth' = 0.9*`wwidth'/(r(N)^.20)
}

tempname delta wid
scalar `delta' = (r(max)-r(min)+2*`wwidth')/(`n'-1)
scalar `wid'   = r(N) * `wwidth'

if `"`at'"' == `""' {
        qui replace `m' = r(min)-`wwidth'+(_n-1)*`delta' in 1/`n'
}

tempname tmp1 tmp2 tmp3

local i 1
if `"`biweigh'"' != `""' {
        local con1 = .9375
        while `i'<=`n' {
                qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') /*
                        */ if `use'
                qui replace `y'=`tt'*`con1'*(1-(`z')^2)^2 /*
                        */ if abs(round(`z',1e-8))<1
                qui summ `y', meanonly
                qui replace `d'=(r(sum))/`wid' in `i'
                qui replace `y'=.
                local i = `i'+1
        }
        qui replace `d'=0 if `d'==. in 1/`n'
}
else if `"`cosine'"' != `""' {
        while `i'<=`n' {
                qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') /*
                        */ if `use'
                qui replace `y'= `tt'*(1+cos(2*_pi*`z')) /*
                        */ if abs(round(`z',1e-8))<0.5
                qui summ `y', meanonly
                qui replace `d'=(r(sum))/`wid' in `i'
                qui replace `y'=.
                local i = `i'+1
        }
        qui replace `d'=0 if `d'==. in 1/`n'
}
else if `"`triangl'"' != `""' {
        while `i'<=`n' {
                qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') if `use'
                qui replace `y'= `tt'*(1-abs(`z')) /*
                        */ if abs(round(`z',1e-8))<1
```

```
        qui summ `y', meanonly
        qui replace `d'=(r(sum))/`wid' in `i'
        qui replace `y'=.
        local i = `i'+1
    }
    qui replace `d'=0 if `d'==. in 1/`n'
}
else if `"`parzen'"' != `""' {
    local con1 = 4/3
    local con2 = 2*`con1'
    while `i'<=`n' {
        qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') if `use'
        qui replace `y'= `con1'-8*(`z')^2+8*abs(`z')^3 /*
                */ if abs(round(`z',1e-8))<=.5
        qui replace `y'= `tt'*`con2'*(1-abs(`z'))^3       /*
                */ if abs(round(`z',1e-8))>.5 & /*
                */ abs(round(`z',1e-8))<1

        qui summ `y', meanonly
        qui replace `d'=(r(sum))/`wid' in `i'
        qui replace `y'=.
        local i = `i'+1
    }
    qui replace `d'=0 if `d'==. in 1/`n'
}
else if `"`gauss'"' != `""' {
    local con1 = sqrt(2*_pi)
    while `i'<=`n' {
        qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') if `use'
        qui replace `y'= `tt'*exp(-0.5*((`z')^2))/`con1'
        qui summ `y', meanonly
        qui replace `d'=(r(sum))/`wwidth' in `i'
        local i = `i'+1
    }
    qui replace `d'=0 if `d'==. in 1/`n'
}
else if `"`rectang'"' != `""' {
    while `i'<=`n' {
        qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') if `use'
        qui replace `y'= `tt'*0.5 if abs(round(`z',1e-8))<1
        qui summ `y', meanonly
        qui replace `d'=(r(sum))/`wid' in `i'
        qui replace `y'=.
        local i = `i'+1
    }
    qui replace `d'=0 if `d'==. in 1/`n'

}
else {
    local con1 = 3/(4*sqrt(5))
    local con2 = sqrt(5)
    while `i'<=`n' {
        qui replace `z'=(`ix'-`m'[`i'])/(`wwidth') if `use'
        qui replace `y'= `tt'*`con1'*(1-((`z')^2/5)) /*
                */  if abs(round(`z',1e-8))<=`con2'
        qui summ `y', meanonly
        qui replace `d'=(r(sum))/`wid' in `i'
        qui replace `y'=.
        local i = `i'+1
    }
```

```
        qui replace `d'=0 if `d'==. in 1/`n'
    }

    label var `d' `"`yl'"'
    label var `m' `"`ixl'"'

    qui summ `d' in 1/`n', meanonly
    local scale = 1/(`n'*r(mean))

    if `"`density'"' != `""' {
        qui replace `d' = `d'*`scale' in 1/`n'
    }

    if `"`graph'"'==`""' {
        if `"`symbol'"'  == `""' {local symbol `"o"'}
        if `"`connect'"' == `""' {local connect `"l"' }
        if `"`title'"'   == `""' {
            local title `"Kernel Density Estimate"'
        }
        if `"`normal'"' != `""' {
            tempvar znorm
            scalar `tmp1' = 1/sqrt(2*_pi*`nsig')
            scalar `tmp2' = -0.5/`nsig'
            qui gen `znorm' = `tmp1'*exp(`tmp2'*(`m'-`nmean')^2)
            local symbol `"`symbol'i"'
            local connect `"`connect'l"'
            if `"`density'"' != `""' {
                tempvar fz
                qui gen `fz' = sum(`znorm')
                qui replace `znorm' = `znorm'/`fz'[_N]
            }
        }
        if `stud' > 0 {
            tempvar tm
            scalar `tmp1' = exp(lngamma((`stud'+1)/2)) /*
                */ / exp(lngamma(`stud'/2)) /*
                */ * 1/sqrt(`stud'*_pi)
            scalar `tmp2' = (`stud'+1)/2
            scalar `tmp3' = sqrt(`nsig')
            qui gen `tm' = `tmp1' * 1/((1+((`m'-`nmean') /*
                */ / `tmp3' )^2/`stud')^`tmp2')
            local symbol `"`symbol'i"'
            local connect `"`connect'l"'
            tempvar ft
            qui gen `ft' = sum(`tm')
            if `"`density'"' != `""' {
                qui replace `tm' = `tm'/`ft'[_N]
            }
            else {
                qui replace `tm' = `tm'/`ft'[_N]/`scale'
            }
        }
        graph `d' `znorm' `tm' `m', s(`symbol') c(`connect') /*
        */ title(`"`title'"') `options'
    }
    /* double save in S_# and r() */
    ret clear
    ret local kernel `"`kernel'"'
    ret scalar width = `wwidth'
```

```
    ret scalar n = `n'          /* (sic) */
    ret scalar scale = `scale'
    global S_1    `""`kernel'""'
    global S_3 = `wwidth'
    global S_2 = `n'
    global S_4 = `scale'

    if `nsave' == 1 {       label var `d' `"density: `ixl'"'
            rename `d' `yl'
        }
        else if `nsave' == 2 {
                label var `m' `"`ixl'"'
                label var `d' `"density: `ixl'"'
                rename `d' `yl'
                rename `m' `xl'
            }
            if "`at'" != "" {
                sort `srtlist' `obsrt'
            }

end
```

Directives pour les auteurs

Les articles portant sur les questions méthodologiques et les sujets techniques reliés aux données qui se trouvent dans les CDR sont appropriés pour le Bulletin technique et d'information.

Langage du matériel soumis

Les manuscrits peuvent être soumis en français ou en anglais. Une fois accepté, les manuscrits seront traduits dans la deuxième langue officielle avant de les publier.

Longueur d'une soumission

Les articles ne doivent pas dépasser 20 pages à double interligne, en excluant les programmes et les appendices. Le Bulletin accepte également les notes et les commentaires brefs (idéalement, trois pages ou moins) traitant sur des solutions rapide aux problèmes analytiques soulevées antérieurement dans le Bulletin ou part les chercheurs collègues.

Le format électronique et la mise en page des manuscrits

Les manuscrits doivent être en format « Microsoft Word (.doc) ». Les auteurs peuvent les soumettre par courrier ordinaire sur disquette ou disque compact. Ils peuvent également les envoyer comme attachement à un courriel.

Les noms des auteurs, le nom de l'établissement principal, et les coordonnées (numéro de téléphone, adresse postale et adresse électronique) du chercheur principal doivent paraître à la page couverture du manuscrit.

Les auteurs doivent se servir de la police Times New Roman de 12 points et des marges de 1 pouce (2,5 cm) en rédigeant leurs manuscrits.

Nous mettons la majuscule qu'au premier mot du titre (p.e. Bulletin technique et d'information).

Nous nous servons des caractères gras que pour les entêtes. Il ne faut pas souligner les mots ou les phrases ni faut il se servir des caractères en italiques.

Les notes en fin de texte et les références doivent être à simple interligne. Les auteurs sont invités de consulter les lignes directrices relatives à la rédaction (voir ci-dessous).

Le format et mise en page des graphiques et tableaux

Les tableaux et graphiques doivent être soumis en format « Microsoft Excel (.xls) » ou en format séparation par virgule (.csv). Le nom des dossiers doit indiquer le contenu (p.e. tableau1, graphique6, etc.).

Les auteurs peuvent les soumettre par courrier ordinaire sur disquette ou disque compact. Ils peuvent également les envoyer comme attachement à un courriel.
Les auteurs sont priés de suivre les directives du Guide de rédaction de Statistique Canada pour la rédaction des tableaux et graphiques.

Indiquez dans le texte l'emplacement des tableaux et graphiques plutot que de les placer pas dans le texte. Servez vous du titre suivit par le nom du fichier entre parenthèses. p.e.

Graphique 6. La consommation du chocolat par les enfants au Canada, 2000 (graphique6)

Les expressions mathématiques

Toutes les expressions mathématiques doivent être dissociées du texte. Les équations doivent être numérotées, le numéro devant figurer à la droite de l'équation, aligné à la marge.

Guide de rédaction à l'intention des auteurs

Les auteurs sont priés de se servir du Guide de rédaction de Statistique Canada. Vous pouvez en procurer une copie du Comité de révision à l'adresse indiquée ci-dessous.

Où soumettre les manuscrits

Envoyez les manuscrits et toutes communications reliées au Bulletin au Comité de révision.

- Adresse électronique – rdc-cdr@statcan.ca
- Par la poste ordinaire :
  Comité de révision, Bulletin technique et d'information du programme CDR
  Centre de données pour la recherche McMaster, Statistique Canada
  Mills Memorial Library
  Library Room 217
  1280 Main Street West
  Hamilton (Ontario), L8S 4L6
  Canada

Révision des soumissions

Le processus de révision initiale des articles relève du Comité de rédaction. Les rédacteurs peuvent inviter des auteurs ayant déjà publié des articles dans le BTI ou des spécialistes à participer au

processus. Les articles soumis au Bulletin font l'objet d'une révision permettant d'en assurer l'exactitude, la cohérence et la qualité.

Au terme du processus de révision initiale par le Comité de rédaction, les articles sont soumis à un examen par les pairs et à un examen interne. L'examen par les pairs sera effectué conformément à la Politique concernant l'évaluation des produits d'information de Statistique Canada. En outre, des cadres supérieurs de Statistique Canada procéderont à des examens internes pour s'assurer que le matériel respecte les directives et les normes du Bureau et qu'il ne porte pas atteinte à la réputation d'impartialité politique, d'objectivité et de neutralité de Statistique Canada.

Veuillez communiquer avec le comité de révision à l'adresse ci haut pour des plus amples renseignements.