Utilisation des sondages sous R : package "Sampling" et "Survey"

Yves Aragon, Camelia Goga
Thibault Laurent, Anne Ruiz-Gazen*

10 avril 2009

1 Introduction

L'estimation par échantillonnage est réalisée dans R à l'aide de deux packages : le package "Sampling" pour tirer des échantillons et le package "Survey" pour calculer les estimations du total et de la variance de l'estimateur utilisé.
Le package "Sampling" est disponible sur `http://cran.r-project.org/` et a été programmé par Yves Tillé et Alina Matei (voir [3]). Le package propose de tirer des échantillons dans une population donnée, selon plusieurs méthodes d'échantillonages et propose des estimateurs du total ou de la moyenne d'une variable d'étude. Le package "Survey" a été programmé par Thomas Lumley. Ce document présente quelques exemples d'utilisation de ce package, dans le cadre du cours du Master 2 Statistique & Econométrie (Toulouse I et III) de Sondages, en présentiel et à distance. Les notations utilisées sont celles du cours de Yves Aragon et Anne Ruiz-Gazen (voir [1]).

2 Téléchargement du package

Nous présentons dans la Fig 1. comment télécharger un package depuis la CRAN. Le téléchargement du package se fait via un serveur Miroir de la Cran. Choisissez celui dont la localisation géographique est proche de chez vous. Le package est ensuite installé sur votre ordinateur (*C :/Program Files/R/R-2.4.0/library*), il ne vous reste plus qu'à taper la commande `library(sampling)` pour le charger, cette dernière étape se faisant à chaque nouvelle session ouverte.

*GREMAQ, Université des sciences sociales, Manufacture des Tabacs, b,t. F 21 allée de Brienne 31000 Toulouse, *ruiz@cict.fr*

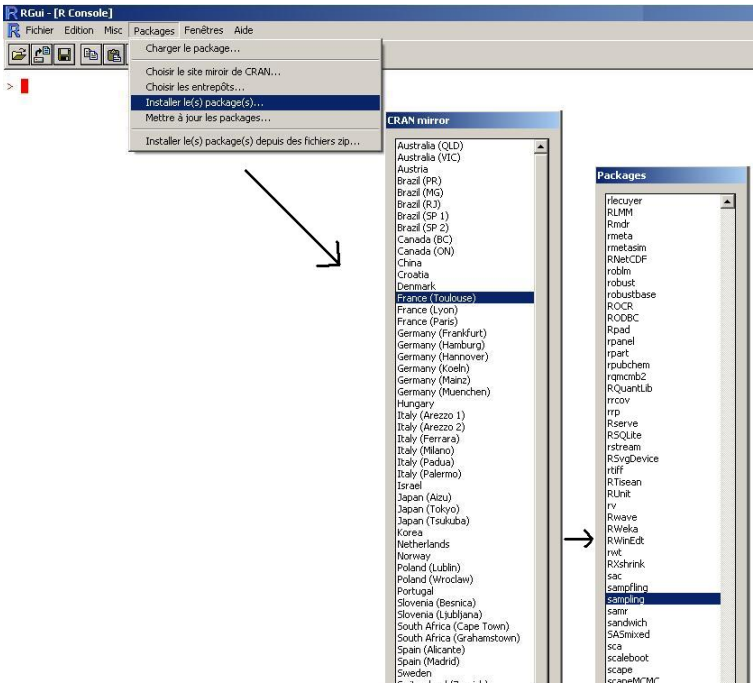

FIG. 1 – Chargement du package Sampling.

3 Transfert de données SAS vers R

Les jeux de données que nous allons utiliser étant disponible sous le format `.sas7bdat` (propre au logiciel SAS), nous montrons dans la Fig 2. comment exporter une table sas en un format exploitable sous R, par exemple le format `.csv`. Le format `.csv` est importable sous R grâce à la fonction `read.csv()`. A priori, en créant de cette façon un fichier de données au format `.csv`, l'importation fonctionne sans même utiliser une des nombreuses options de `read.csv()`.

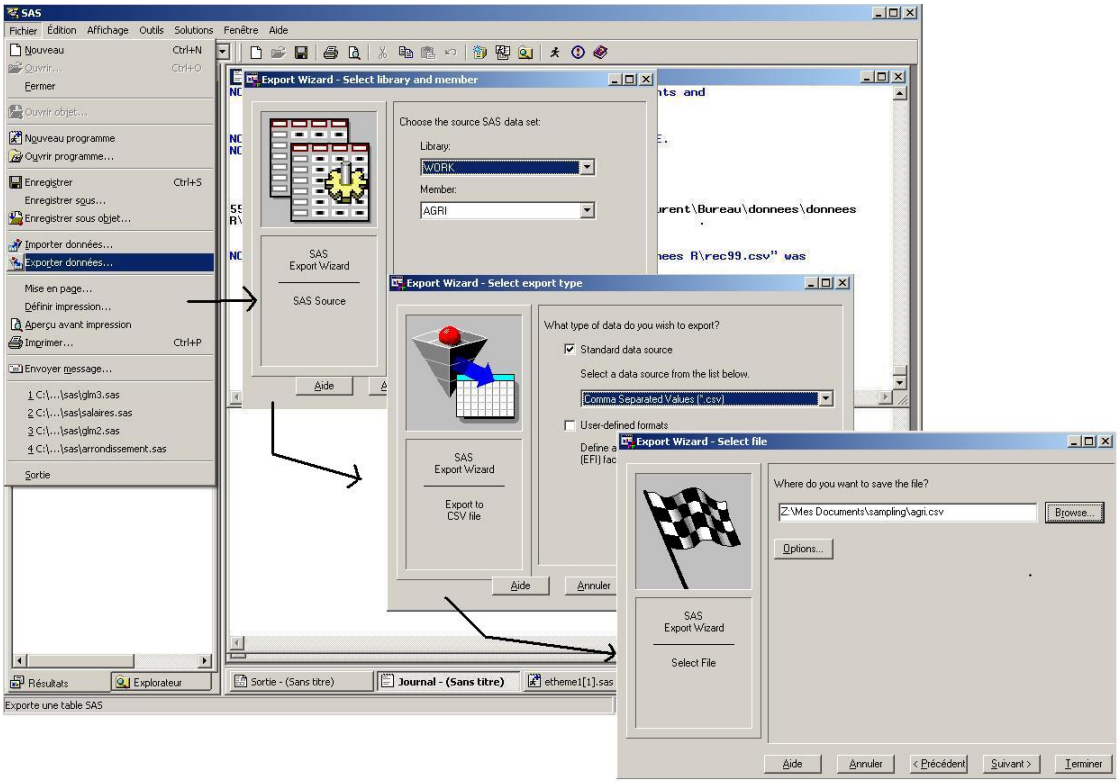

Fig. 2 – Transfert de données SAS vers R.

4 Présentation des jeux de données

4.1 Recensement 99

Nous considérons les $N = 554$ communes de Haute-Garonne de moins de 10000 habitants au recensement 1999 (population sans double compte). Nous disposons pour ces communes du code de la commune, du code du BVQ (Bassin de

Vie Quotidienne) auquel la commune appartient et de la variable nombre de logements vacants (LOGVAC) en 1999. La population d'étude est l'ensemble des 554 communes de Haute-Garonne de moins de 10000 habitants au recensement 1999. Pour obtenir plus de détail sur ces données, on pourra consulter le site de l'INSEE.

```
> rec<-read.csv("Z:\\Mes Documents\\sampling\\rec99.csv")
> dim(rec)
[1] 554   7
> names(rec)
[1] "CODE_N"   "COMMUNE"  "BVQ_N"    "POPSDC99" "LOG"      "LOGVAC"
"stratlog"
```

On tirera des échantillons sur cette population à partir de différentes méthodes et on s'intéressera plus particulièrement aux estimations du nombre total de logements vacants (logvac) et l'estimation de la variance des estimateurs du nombre total.

Calculons le nombre total de logements vacants sur l'ensemble de la population [1] ainsi que la variance[2].

```
> sum(rec$LOGVAC)
[1] 10768
> var(rec$LOGVAC)
[1] 1104.5
```

Le nombre total de logements vacants est égal à 10768 est la variance S^2_{yU} est égale à 1104.5.
Dans la suite, nous choisirons $n = 50$ selon différents types de plans de sondages.

5 Tirage d'un échantillon selon un plan SI

5.1 Plan SI : quelques algorithmes de tirage

Il y a plusieurs façon de sélectionner un échantillon SI de taille n dans une population de taille N.
Un tirage très simple consiste à sélectionner un premier individu à probabilité égale dans la population et l'éliminer pour la suite des tirages de la base de sondage. On sélectionne ensuite un deuxième individu à probabilité égale parmi les $N-1$ unités de la population et on continue cette procédure n fois. Néanmoins, pour des populations stockées de façon séquentielle et de grandes tailles, il y a des méthodes appelées *list-sequential*. Une de ces méthodes est la suivante :
on considère N réalisations indépendantes d'une variable uniforme $\varepsilon \sim Unif(0,1)$.
On ordonne les N valeurs et l'échantillon SI de taille n est obtenu en considérant les individus qui correspondent aux n plus petites valeurs.

[1] Autrement dit, la valeur exact que l'on va chercher à estimer...

[2] Noter que R donne la variance corrigée, à savoir que le dénominateur est (N-1), ce qui nous convient donc dans le calcul de $S^2_{y_U}$

5.2 Sélection d'un plan SI avec R

Pour un plan à probabilités égales sans remise ou plan SI, il y a C_N^n façons possibles de tirer n individus dans une population de N individus. La fonction `writesample()` du package `sampling` renvoie tous les échantillons possibles, à savoir une matrice de dimension $C_N^n \times N$ avec des 1 si l'individu j ($j = 1, ..., N$) a été tiré et 0 sinon[3]. Par exemple, pour tirer 4 individus dans un échantillon de 10 individus, il y a 210 façons de le faire.

```
> w=writesample(4,10)
> dim(w)
[1] 210  10
```

La fonction `srswor()` du package `sampling` renvoie un échantillon parmi les C_N^n possibles. Cette fonction prend comme argument d'entrée le couple (n, N) où n est le nombre d'individus à tirer parmi N individus et renvoie un vecteur de taille N avec des 1 si l'individu i a été sélectionné et 0 sinon[4]. Par exemple, si on tire 50 communes dans le jeu de données de recensement et qu'on souhaite ensuite afficher les 10 premières sélectionnées :

```
> si.rec<-srswor(50,554)
> nom<-rec$COMMUNE[si.rec==1][1:10]
> nom
 [1] CADOURS      MONTAUT      FRONTON      GRENADE      LILHAC
 [6] MONTBRUN-BOC LARCAN       CASSAGNE     SALIES-DU-SA SAINT-PIERRE
545 Levels: AGASSAC AIGNES AIGREFEUILLE ALAN ALBIAC AMBAX ANAN ...
VILLENOUVELL
```

5.3 Estimation du total dans le plan SI

L'estimateur sans biais du nombre total de logements vacants est l'estimateur de Horvitz-Thompson,

$$\hat{t}_{SI} = \frac{N}{n} \sum_s y_k \tag{1}$$

calculé directement par

```
> 554*mean(rec$LOGVAC[which(si.rec==1)]).
[1] 15401.2
```

Le package "Sampling" contient la fonction `HTestimator()` qui permet également de calculer l'estimateur du total mais pas sa variance. Cette fonction a deux arguments : le vecteur de taille n contenant les valeurs de la variable d'intérêt et le

[3] Noter que sur des populations aux effectifs importants, on est rapidement confronté à des problèmes de stockage dans la matrice.

[4] On notera que la fonction `sample()` permet également de sélectionner avec ou sans remise n individus parmi N. Contrairement à `srswor()`, la fonction `sample()`, incluse parmi les fonctions de base, renvoie le vecteur de taille n des valeurs des individus sélectionnées.

vecteur de taille n également avec des probabilités d'inclusion correspondantes au plan utilisé. Pour un plan aléatoire simple sans remise de taille n parmi N, les probabilités d'inclusion sont $\frac{n}{N}$. Un exemple de l'utilisation de cette fonction est le suivant :

```
##selection de l'échantillon

>si.rec=srswor(50,554)

##construction du vecteur des probabilités d'inclusion
>pik.si=rep(50/554,50)
##construction du vecteur des valeurs de la variable d'intérêt pour\\
## les communes sélectionnées

>si.Logvac=rec$LOGVAC[which(si.rec==1)]

##l'estimateur d'Horvitz-Thompson
> Logvac_si=HTestimator(si.Logvac,pik.si)
>  15379.04
```

Nous avons obtenu une estimation du nombre total de logements vacants égale à 15379.04 qui est différente de 15401.2 obtenue précédemment. Cela s'explique par le fait que nous avons deux échantillons différents. Pour garder les mêmes individus dans l'échantillon, il faut utiliser la fonction `set.seed()` qui fixe la graine du tirage aléatoire au début de la sélection des échantillons. Le numéro choisi comme argument de `set.seed()` (date de naissance, par exemple) doit être le même tout au long de l'étude.

Nous constatons que les deux estimations sont assez éloignées de la valeur réelle du nombre total de logements, car elles dépendent bien évidemment de l'échantillon tirée.

5.4 Estimation de la variance de l'estimateur du total dans le plan SI

L'estimateur sans biais de la variance de l'estimateur du nombre total de logements vacants est $N^2(1-\frac{n}{N})\frac{S^2_{y_s}}{n}$, soit :

```
> 554^2*(1-50/554)*var(rec$LOGVAC[which(si.rec==1)])/50
[1] 11821892
```

Cet estimateur est bien entendu très différent de la variance exacte, car dépendant de l'échantillon tirée. On peut faire ce calcul direct de la variance pour un plan simple comme celui-ci. Néanmoins, en pratique les plans sont assez complexes et c'est le package "Survey" qui permet de calculer l'estimateur de la variance pour ces types de plans. Dans la suite, nous allons donner une description des

fonctions utilisées pour l'estimation de la variance.
Pour utiliser les fonctions du package `survey`, il faut au préalable comprendre la notion d'objet. L'idée est de réunir dans un seul élément un certain nombre d'informations contenu dans une ou plusieurs variables, voire dans d'autres objets. Par exemple, pour estimer le nombre total de logements vacants, d'après la formule 1, nous avons seulement besoin des valeurs y_k, $k \in s$ et du facteur de dilatation f_k égal ici à N/n. Aussi, construire un objet de classe `survey.design` permet de conserver uniquement l'information dont nous avons besoin. Ensuite, pour obtenir l'estimation du total et l'estimation de l'écart-type de l'estimateur du total, on utilise la fonction `svytotal`.

La syntaxe de `survey.design` est assez complexe

```
svydesign(ids, probs=NULL, strata = NULL, variables = NULL, fpc=NULL,
data = NULL, nest = FALSE, check.strata = !nest, weights=NULL,...)
```

Pour le plan aléatoire simple sans remise, nous avons besoin seulement des arguments suivants :

1. `id` : cet argument est toujours demandé et est un identifiant ;
2. `weights` : formule or vecteur des probabilités d'inclusion ;
3. `fpc` : formule or vecteur de la même taille que `weights` avec la correction en population finie, c'est à dire $f = n/N$. Si `fpc` n'est pas spécifié, le tirage est supposé par défaut avec remise.
4. `data` : la base de données dans laquelle nous effectuons le sondage et qui contienne implicitement les valeurs de la variable d'intérêt pour les individus de l'échantillon ; quand l'argument `data` est spécifié, `variable` est `NULL`.

Les arguments `weights`, `fpc`, `data` sont optionnels.

Un exemple de l'utilisation pour le plan SI est :

```
# choisir la graine aléatoire
>set.seed(1974)
# sélection de l'échantillon SI
>si.rec=srswor(50,554)
# création d'un objet de classe survey.design

ech.si <- svydesign(id=~CODE_N, weights=rep(554/50,50),fpc=rep(50/554,50),
data=rec[which(si.rec==1),])
```

Pour calculer une estimation du total de LOGVAC et de la variance de l'estimateur de Horvitz-Thompson, on utilise la fonction `svy.total` comme suit

```
# estimation du total de la consommation et de l'estimation
# de l'Ècart-type de l'estimateur
```

```
>svytotal(~LOGVAC, ech.si)
       total     SE
LOGVAC 14526 2688.7
```

Le total estimé est de 14526 et la variance estimée est $2688.7^2 = 7229108$.

5.5 Etude avec des simulations

Nous considérons 1000 échantillons SI et nous allons calculer pour chaque échantillon, l'estimation du total de la variable LOGVAC. Nous créons un vecteur avec les 1000 estimations obtenues en utilisant chaque fois des échantillons SI est la moyenne de ces 1000 estimations constitue l'estimation finale de LOGVAC. Si on avait considéré tous les C_{554}^{50} échantillons possibles, alors leurs moyenne était exactement 10768.
Le coefficient de variation est le coefficient de variation de ces 1000 estimations.

```
########## création du vecteur donnant le numéro de la réplication et
#####d'un deuxième vecteur contenant que des 1 et qui sera rempli
### avec les estimations
> nb.simul<-matrix(1:1000,1000,1); t.esti<-matrix(1,1000,1)

> for (i in 1:1000)

{ s.rec<-srswor(50,554)
t.esti[i]<-554*mean(rec$LOGVAC[which(s.rec==1)]) }
```

A la fin de ce programme, nous avons le vecteur `t.esti` de taille 1000, le nombre de simulations choisi, qui contient les estimations du total de LOGVAC. On peut représenter l'histogramme de ces 1000 valeurs (la Fig 3.) en exécutant l'instruction

```
> hist(t.esti)
```

et l'estimation de LOGVAC, sa variance et le coefficient de variation pour 1000 réplications sont obtenus en exécutant

```
> logvac_si=mean(t.esti)
> var_si=var(t.esti)
> cv_si=sd(t.esti)/logvac_si
```

Nous pouvons représenter graphiquement les moyennes cumulées en fonction du nombre d'échantillons :

```
plot(nb.simul,cumsum(t.esti)/nb.simul,type="l")
lines(1:1000,rep(10768,1000),col='red')
```

On constate sur la Fig 4. que la moyenne des estimations tend vers la vraie valeur du total. Ceci s'explique car l'estimateur que nous avons utilisé est un estimateur sans biais.
En utilisant les fonctions du package `survey`, on obtient

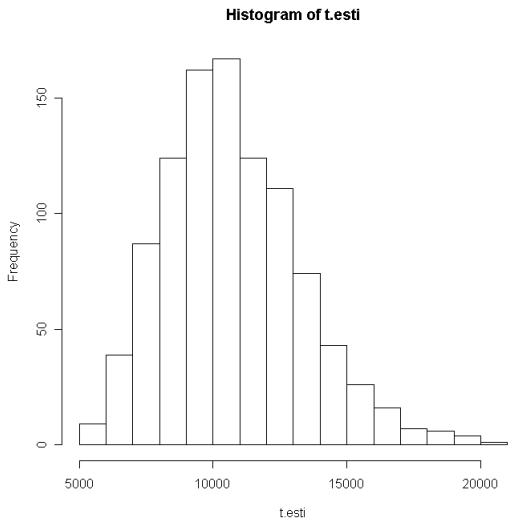

FIG. 3 – Histogramme des 1000 estimations du total.

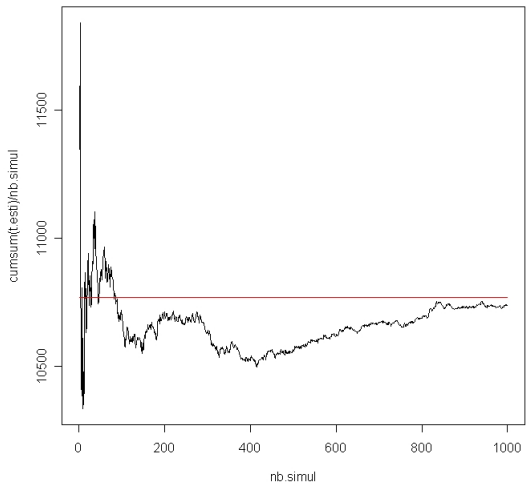

FIG. 4 – Moyenne des estimations du total en fonction du nombre d'échantillons tirés.

```
> nb.simul<-matrix(1:5000,5000,1); t.esti<-matrix(1,5000,1);
 var.esti=matrix(1,1000,1)
> for (i in 1:1000)

{ si.rec<-srswor(50,554)
ech.si <- svydesign(id=~CODE_N, weights=rep(554/50,50),fpc=rep(50/554,50),
data=rec[which(si.rec==1),])
estimation=svytotal(~LOGVAC, ech.si)
t.esti[i]<- estimation[1]
var.esti[i]=SE(estimation)^2
}
##############estimation sur les 1000 echantillons
> si_total_simul=mean(t.esti)

> si_total_simul
[1] 10691.75
############### variance des 1000 estimations
> si_var_simul=var(t.esti)

> si_var_simul
        [,1]
[1,] 6045878
####################calcul du cv
> cv_simul=sqrt(var(t.esti)/si_total_simul)

> cv_simul
[1] 0.2299752
```

Nous avons obtenu une estimation du nombre total de LOGVAC de 10691.75 et un coefficient de variation de 23%.

Aussi, comme nous l'avons fait pour la moyenne, calculons sur plusieurs échantillons la moyenne de l'estimateur de la variance en fonction du nombre de réplications. Sur la Fig 5., on constate que la convergence vers la valeur exacte de la variance est beaucoup plus longue que pour la moyenne.

```
#######histogramme des 1000 estimations
hist(t.esti)
plot(nb.simul,cumsum(var.esti)/nb.simul,type="l")
lines(1:5000,rep(6167879,5000),col='red')
```

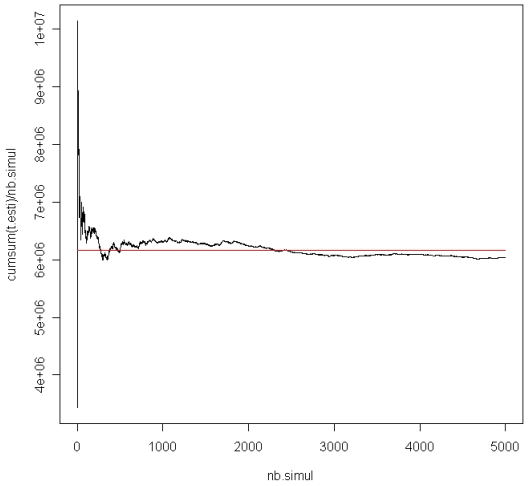

FIG. 5 – Moyenne des estimateurs de la variance de l'estimateur du total en fonction du nombre d'échantillons tirés.

6 Le plan aléatoire simple avec remise : PEAR

6.1 Sélection du plan PEAR avec R

Un plan aléatoire simple avec remise est sélectionné en utilisant la fonction `srswr` qui a comme arguments (m, N) où m est la taille de l'échantillon (ou le nombre de tirages) et N la taille de la population. Cette fonction renvoie un vecteur de taille N qui contient le nombre de fois une unité a été sélectionnée.

```
##selection d'un echantillon aleatoire simple avce remise, \\
##on effectue 5 tirages avec remise parmi 10
>srswr(5,10)
3 0 1 1 0 0 0 0 0 0
```

L'échantillon obtenu contient la première unité 3 fois, la troisième unité 1 fois et la quatrième 1 fois.

6.2 Estimation du total et de la variance pour un plan PEAR

Dans le cas des sondages avec remise, le total $t_y = \sum_U y_k$ est estimé par l'estimateur de Hansen-Hurvitz,

$$\hat{t}_{HH} = \frac{1}{m} \sum_{i=1}^{m} \frac{y_{k_i}}{p_{k_i}}$$

où p_{k_i} est la probabilité que l'individu k_i soit sélectionné au i-ème tirage. Dans le cas d'un plan PEAR, ces probabilités valent toutes $\frac{1}{N}$ et l'estimateur devient

$$\hat{t}_{pear} = \frac{N}{m} \sum_{i=1}^{m} y_{k_i}.$$

La variance de cet estimateur est égale à

$$\mathrm{Var}(\hat{t}_{pear}) = \frac{N(N-1)}{m} S^2_{yU} \simeq \frac{N^2}{m} S^2_{yU}.$$

et elle est estimée par

$$\widehat{\mathrm{Var}}(\hat{t}_{pear}) = \frac{N^2}{m} S^2_{ys}.$$

```
> set.seed(1974)

####selection de l'échantillon pear
> spear.rec=srswr(50,554)
 spear.rec
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0
 [69] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0
  0 0 0 1 0 0 0 0 1
[137] 0 0 0 0 0 0 0 2 0 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0
[205] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 0
 0 0 0 0 0 0 0 0 0
[273] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 0
[341] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0
 0 0 0 0 0 1 0 0 0
[409] 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 1 1 0 0 0 0 0
[477] 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 0 0 0 0 2 1 0
[545] 0 0 0 0 0 0 0 0 0 1
```

On peut remarquer que nous avons des unités sélectionnées plusieurs fois. Pour calculer l'estimateur de Hansen-Hurvitz avec `svydesign`, nous devons créer un vecteur avec des nouveaux poids qui tiennent compte de la multiplicité des unités sélectionnées.

```
#####le plan pear: estimation
###creation des nouveaux poids
> poids_pear=(554/50)*spear.rec

###creation de l'objet svydesign

> ech.pear=svydesign(id=~CODE_N, weights=poids_pear[poids_pear>0],
data=rec[which(spear.rec>0),])
```

Dans l'objet `ech.pear`, le data frame `data` contient que 47 communes, c'est à dire les communes tirées plusieurs fois n'apparaissent qu'une fois. Alors, si une commune a été tiré deux fois par exemple, son poids est $2 * \frac{554}{50}$ ce qui explique la création du vecteur `poids_pear` pour tenir compte de nombre de fois qu'une commune a été sélectionnée.

```
###calcul de l'estimateur de Hansen et Hurvitz et
de l'estimateur de sa variance

> svytotal(~LOGVAC, ech.pear)

###résultat

     total     SE
LOGVAC 12188 2318.3
```

6.3 Le plan PEAR avec simulations

Le principe est le même que dans le cas d'un plan SI.

```
tirage de 1000 echantillons PEAR de taille 50 et de 1000 estimations
nb.simul<-matrix(1:1000,1000,1); t.esti<-matrix(1,1000,1)
for (i in 1:1000)

{
spear.rec<-srswr(50,554)
poids_pear=(554/50)*spear.rec
ech.pear <- svydesign(id=~CODE_N, weights=poids_pear[poids_pear>0],
data=rec[which(spear.rec!=0),])
estimation=svytotal(~LOGVAC, ech.pear)
t.esti[i]<- estimation[1]
```

```
}


######estimation sur les 1000 echantillons
si_total_simul=mean(t.esti)
[1] 10761.25
si_var_simul=var(t.esti)
[1,] 7324952
cv_simul=sd(t.esti)/si_total_simul
[1] 0.2515010
```

Références

[1] Y. Aragon et A. Ruiz-Gazen, "Plans simples à probabilités égales", cours de sondages, M2 FOAD Statistique & Econométrie, `http://foad.univ-tlse1.fr/mod/resource/view.php?id=9512`, 2007.

[2] J.-C. Deville et N. Roth, "La précision des enquêtes sur l'emploi", Economie et Statistique, 193-194, pp. 127-134.

[3] Y. Tillé et A. Matei, "Le package sampling en langage R", Communication aux 38èmes Journées de la Société Française de Statistique, EDF, Clamart, Juin 2006.