Université de Poitiers
–
IUT de Niort

SAE ECHANTILLONNAGE ET ESTIMATION

Chloé Decoust - Xavier Barbeau

Table des matières

Introduction :

Ce projet a pour but de permettre d'appréhender l'incertitude et la précision de l'estimation d'une grandeur mesurable dans une population à l'aide d'un intervalle de confiance réalisé à partir d'un processus d'échantillonnage. Pour cela, nous l'avons fait, dans un premier temps, par l'intermédiaire d'un sondage aléatoire simple à probabilité égale, puis dans un second temps, par l'intermédiaire d'un échantillonnage par strates. L'objet de notre étude a été la population française par communes où la promotion s'est réparti différentes régions, pour notre part nous avons choisis la région du Centre-Val-de-Loire.

Partie 1 : Estimation du nombre d'habitants d'une région de France

Pour cette première partie, nous avons commencé par télécharger et importer le fichier : « population_française_communes.xslx »

```
# ouverture fichier CSV
table <- read.csv("C:/Users/chloe/OneDrive - Université de Poitiers/Bureau/BUT 1/S2/SAE/Stat inf/population_francaise_communes.csv",
                  sep=';',fileEncoding = "Latin1",header=TRUE)
table$Population.totale <- gsub(" ","",table$Population.totale)
table$Population.totale <- as.numeric(table$Population.totale)
```

I. Echantillonnage aléatoire simple :

Ensuite, nous avons pu sélectionner seulement les données associées à notre région et les colonnes qui nous intéressaient : « Commune », « Code.departement » et la « Population ».

```
# Partie 1 : échantillonnage aléatoire simple
# question 1
donnees <- table[table$Nom.de.la.région == "Centre-Val de Loire", c("Code.département","Commune","Population.totale")]
donnees<- unique(donnees)
head(donnees)
```

Nous avons ensuite, afficher les 6 premières lignes de notre nouveau data frame : « donnees ».

	Code.département	Commune	Population.totale
6028	18	Achères	371
6029	18	Ainay-le-Vieil	195
6030	18	Les Aix-d'Angillon	1909
6031	18	Allogny	1115
6032	18	Allouis	1097
6033	18	Annoix	242

Puis, nous avons défini U comme notre liste de communes que nous avons compté. Le total est de 1757 communes.

```
# question 2
U <- donnees$Commune
head(U)
N = length(U)
N
```

Nous avons, par la suite, calculer le nombre T exact d'habitants de notre région, T est donc égal à 2 632 683 habitants.

```
# question 3
T<-sum(donnees$Population.totale)
```

Pour estimer le nombre d'habitants de la région Centre-Val-de-Loire, on a, ensuite, pu créer notre échantillon E de taille n = 100 communes tirées selon un sondage aléatoire simple à partir de l'ensemble de nos communes.

```
# Tirage aléatoire simple d'un échantillon de taille n = 100
n=100
E=sample(U, n,replace=FALSE)
head(E)
```

Nous avons créé une nouvelle table donnees1 avec les colonnes « Communes », « departement » et « population » auxquelles nous avons affiché les 6 premières lignes de la nouvelle table.

```
# question 4
donnees1= donnees[donnees$Commune %in% E, ]
head(donnees1)
```

	Code.département	Commune	Population.totale
6032	18	Allouis	1097
6039	18	Arpheuilles	306
6048	18	Bannegon	276
6055	18	Berry-Bouy	1209
6074	18	La Chapelle-d'Angillon	620
6081	18	Charly	249

A partir de la table donnees1, nous avons calculé le nombre moyen d'habitants mu de l'échantillon E et un IDC à 95 % du nombre moyen d'habitants mu par commune.

```
# question 5
  #moyenne de l'échantillon
moy= mean(donnees1$Population.totale)
moy
 # idc de mu
idcmoy = t.test(donnees1$Population.totale)$conf.int
idcmoy
```

Cet échantillon E nous a permis, par la suite, de déduire une estimation Test du nombre d'habitants total T ainsi qu'un intervalle de confiance.

```
# question 6
# Nbre d'habitants total estimé
Test = N*moy
Test
# IDC de T
idcT = idcmoy*N
idcT
#Marge d'erreur
marge=(idcT[2]-idcT[1])/2
marge
```

L'IDC calculé est [1 741 449 ; 2 800 797] donc la valeur T qui vaut 2 632 683 habitants appartient à cet intervalle de confiance. Cependant, la valeur Test, qui est égale à 1 532 397, n'est pas comprise dans l'intervalle.

```
> Test
[1] 1532397
> # IDC de T
> idcT = idcmoy*N
> idcT
      1       1
1741449 2800797
> #Marge d'erreur
> marge=(idcT[2]-idcT[1])/2
> marge
       1
529674.3
```

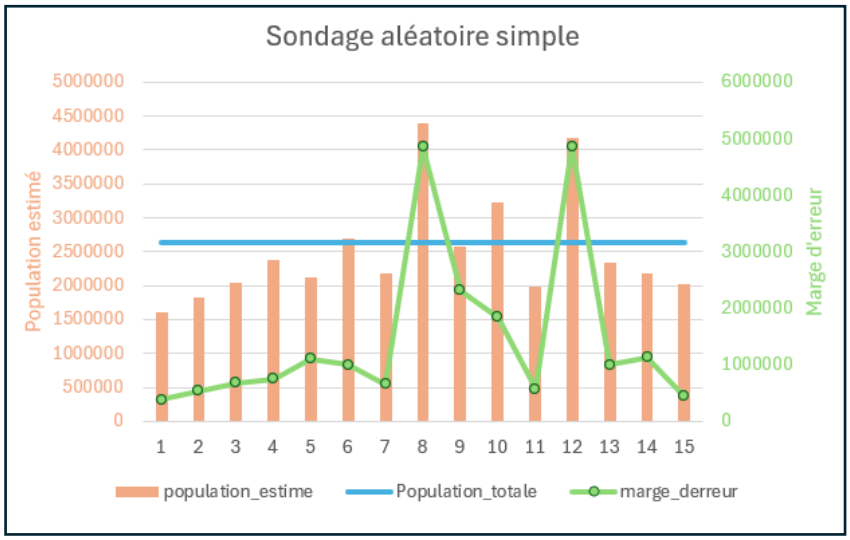
Pour finir cette partie, nous avons refait les 3 dernières étapes 15 fois pour plus de précision, c'est-à-dire la création d'un nouvel échantillon puis d'une nouvelle table, le calcul de la moyenne de la population pour pouvoir calculé l'IDC de la moyenne mu, ensuite on a calculé Test soit la population estimée et son idcT et enfin la marge d'erreur de cet idcT. Pour automatiser cette partie nous avons fait le choix de faire une boucle 'for' pour que à chaque répétition le résultat s'ajoute automatiquement dans le data frame final « Tableau_simple ».

```r
# question 7
# initialisation du data frame final a expoté
Tableau_simple <- data.frame(Population_totale = numeric(15),
                             population_estime = numeric(15),
                             idc_inf = numeric(15),
                             idc_sup = numeric(15),
                             marge_derreur = numeric(15))

# boucle pour répété les 3 dernières étape
for (i in 1:15) {
  E=sample(U,size=n, replace=FALSE)
  donnees1 <- donnees[donnees$Commune %in% E, ]

  # Ajout d'impressions pour le débogage
  print(paste("Groupe", i))
  print(donnees1)

  # Vérifier si le nombre d'observations est suffisant pour effectuer un test T
  if (nrow(donnees1) >= 2) {
    moy <- mean(donnees1$Population.totale)
    idcmoy <- t.test(donnees1$Population.totale)$conf.int
    Test <- N * moy
    idcT <- idcmoy * N
    marge <- (idcT[2] - idcT[1]) / 2

    # Stocker les résultats dans le tableau finale
    Tableau_simple[i, "Population_totale"] <- sum(donnees$Population.totale)
    Tableau_simple[i, "population_estime"] <- Test
    Tableau_simple[i, "idc_inf"] <- idcT[1]
    Tableau_simple[i, "idc_sup"] <- idcT[2]
    Tableau_simple[i, "marge_derreur"] <- marge
  }
}
```

Nous avons ensuite exporté cette nouvelle table finale au format csv pour pouvoir faire un graphique.

```r
# export du fichier CSV
write.csv2(Tableau_simple, file = "Tableau_simple.csv", row.names = FALSE)
```

Quant aux résultats on peut voir que nous avons deux marges d'erreurs excessivement élevées mais la population estimée qui y sont associées sont elles aussi très hautes par rapport à la population totale. Ce qui signifie aussi que les IDC sont très grands.



II. Echantillonnage aléatoire stratifié :

Pour l'échantillonnage aléatoire stratifié, nous cherchons alors à répartir notre table en quartiles en vue de la population totale.

```
# Question 1
# quartile
summary(donnees$Population.totale)
```

Créons donc la table « datastrat » qui contient les mêmes colonnes que la table donnée et avec en plus une colonne où nous mettons nos strates.

```
# question 2
# strates :
datastrat=donnees
datastrat$Strate=cut(datastrat$Population.totale, breaks=c(0,275, 540, 1175, 140475), labels=c(1,2,3,4))
head(datastrat)
```

Tirons un échantillon de 100 communes en respectant une proportionnalité dans les strates.

```
# question 3 :
# effectif des strates
data = datastrat[order(datastrat$Strate), ]
Nh=table(data$Strate)

# Tirage d'un échantillon stratifié de taille n=100
n=100
nh=c(25,25,25,25)
```

```
# Sondage des strates sans remise
st = strata(datastrat, stratanames = c("Strate"), size = nh, method = "srswr")
data1=getdata(datastrat, st)
head(data)
data1 = data1[order(data1$Strate), ]
Nh=table(data$Strate)

# Poids des strates
gh=Nh/N

# Taux de sondage dans les strates
fh=nh/Nh
```

On définit alors nos quatre échantillons obtenus et l'on calcul alors leur moyenne et leur variance.

```
# Question 4
# Mise en place des 4 échantillons
ech1=data1[data1$Strate==1, ]
ech2=data1[data1$Strate==2, ]
ech3=data1[data1$Strate==3, ]
ech4=data1[data1$Strate==4, ]

# Moyennes des 4 échantillons
m1=mean(ech1$Population.totale)
m2=mean(ech2$Population.totale)
m3=mean(ech3$Population.totale)
m4=mean(ech4$Population.totale)

# Variances des 4 échantillons
var1=var(ech1$Population.totale)
var2=var(ech2$Population.totale)
var3=var(ech3$Population.totale)
var4=var(ech4$Population.totale)
```

Ensuite, on calcul une estimation Xbarstr du nombre moyen d'habitant par commune mu et une estimation de la variance de de ce dernier. On calcul par la suite l'intervalle de confiance pour mu.

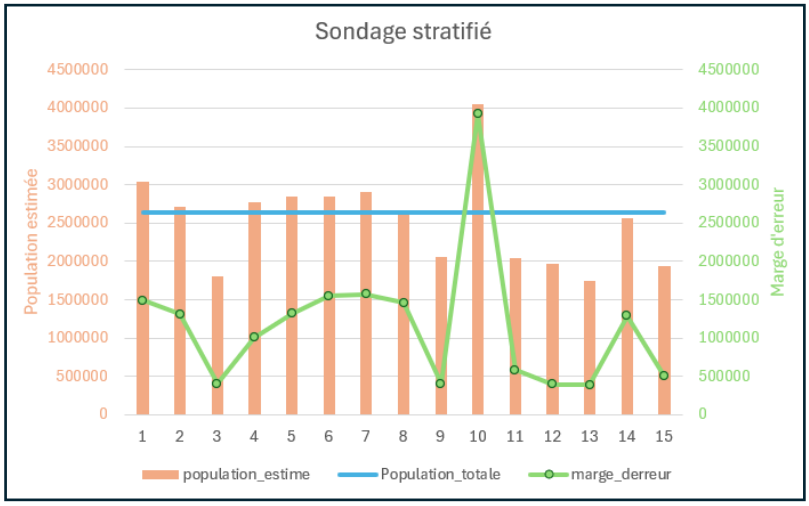
```
# Question 5
# Moyenne générale (des 3 échantillons réunis)
Xbarst= (Nh[1]*m1 + Nh[2]*m2 + Nh[3]*m3 + Nh[4]*m4)/N

# Estimation de la variance de Xbarst
varXbarst= ((gh[1])^2)*(1-fh[1])*var1/(nh[1]) + ((gh[2])^2)*(1-fh[2])*var2/(nh[2]) +
  ((gh[3])^2)*(1-fh[3])*var3/(nh[3]) + ((gh[4])^2)*(1-fh[4])*var4/(nh[4])

# IDC pour mu à 95%
alpha=0.05
binf = Xbarst - qnorm(1-alpha/2)*sqrt(varXbarst)
bsup = Xbarst + qnorm(1-alpha/2)*sqrt(varXbarst)
idcmoy=c(binf, bsup)
```

On estime alors Tstr de T et un intervalle de confiance pour le nombre d'habitant T. Cette IDC contient la vrai valeur T et sa marge d'erreur est de 1 253 214.

```r
# Question 6
# Estimation du total T
Tstr= N*Xbarst
Tstr

# Estimation par IDC du total T
binf = idcmoy[1]*N
bsup= idcmoy[2]*N
idcT=c(binf, bsup)
idcT

# Marge d'erreur
marge=(idcT[2]-idcT[1])/2
marge
```

Comme pour l'échantillonnage aléatoire simple, nous créons une fonction que l'on fait ce répété 15 fois.

```r
# Question 7
Tableau_strate <- data.frame(Population_totale = numeric(15),
                             population_estime = numeric(15),
                             idc_inf = numeric(15),
                             idc_sup = numeric(15),
                             marge_derreur = numeric(15))

for (i in 1:15) {
  st = strata(datastrat, stratanames = c("Strate"), size = nh, method = "srswr")
  data1=getdata(datastrat, st)

  # Ajout d'impressions pour le débogage
  print(paste("Groupe", i))
  print(data1)

  # Vérifier si le nombre d'observations est suffisant pour effectuer un test t
  if (nrow(data1) >= 2) {
    st = strata(datastrat, stratanames = c("Strate"), size = nh, method = "srswr")
    data1=getdata(datastrat, st)
    data1 = data1[order(data1$Strate), ]
    Nh=table(data$Strate)
    gh=Nh/N
    fh=nh/Nh
    ech1=data1[data1$Strate==1, ]
    ech2=data1[data1$Strate==2, ]
    ech3=data1[data1$Strate==3, ]
    ech4=data1[data1$Strate==4, ]
    m1=mean(ech1$Population.totale)
    m2=mean(ech2$Population.totale)
    m3=mean(ech3$Population.totale)
    m4=mean(ech4$Population.totale)
    var1=var(ech1$Population.totale)
    var2=var(ech2$Population.totale)
    var3=var(ech3$Population.totale)
    Xbarst= (Nh[1]*m1 + Nh[2]*m2 + Nh[3]*m3 + Nh[4]*m4)/N
    varXbarst= ((gh[1])^2)*(1-fh[1])*var1/(nh[1]) + ((gh[2])^2)*(1-fh[2])*var2/(nh[2]) +
      ((gh[3])^2)*(1-fh[3])*var3/(nh[3]) + ((gh[4])^2)*(1-fh[4])*var4/(nh[4])
    binf = Xbarst - qnorm(1-alpha/2)*sqrt(varXbarst)
    bsup = Xbarst + qnorm(1-alpha/2)*sqrt(varXbarst)
    idcmoy=c(binf, bsup)
    Tstr= N*Xbarst
    binf = idcmoy[1]*N
    bsup= idcmoy[2]*N
    idcT=c(binf, bsup)
    marge=(idcT[2]-idcT[1])/2

    # Stocker les résultats dans le tableau
    Tableau_strate[i, "Population_totale"] <- sum(donnees$Population.totale)
    Tableau_strate[i, "population_estime"] <- Tstr
    Tableau_strate[i, "idc_inf"] <- idcT[1]
    Tableau_strate[i, "idc_sup"] <- idcT[2]
    Tableau_strate[i, "marge_derreur"] <- marge
  }
}

# Export du fichier CSV
write.csv2(Tableau_strate, file = "Tableau_strate.csv", row.names = FALSE)
```



Nous obtenons des résultats similaires à ceux de la première méthode mais avec une seule valeur extrême.

Partie 2 : Test du khi-deux d'indépendance

Nous allons maintenant réaliser un test du khi-deux d'indépendance. Pour cela nous récupérons le fichier excel « voiture.xls » que l'on enregistre au format csv.

On importe ensuite la table sur R.

```
# PARTIE 2 :
# Ouverture de la table voitures
voiture <- read.csv("C:/Users/chloe/OneDrive - Université de Poitiers/Bureau/BUT 1/S2/SAE/Stat inf/voitures.csv",
                    sep=';',fileEncoding = "Latin1",header=TRUE)
```

Puis on affiche les 6 premières lignes.

```
> head(voiture)
   Marque PuissFisc Categorie Ancienneté       Formule          Ville Prime
1 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi     Chantecorps   171
2 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi      Le Chillou   176
3 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi        Loubillé   316
4 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi Pougne-Herisson   310
5 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi     Villefollet   316
6 Citroen de_7_a_10   BERLINE    4-7 ans Tous Risques      Chantecorps   444
>
```

Cette table contient 7 variables dont 6 qualitatives et une seule quantitative : « Prime ». La variable « Marque » contient des marques de voiture, « Categorie » contient le type de voiture, « Anciennete » contient l'ancienneté de la voiture, « Formule » sert à savoir le type de contrat soustrait pour le véhicule, « Ville » correspond à la localisation du propriétaire du véhicule et enfin la variable « prime » qui contient la prime d'assurance.

Nous avons, par la suite, créer une nouvelle variable qualitative « Prime2 » qui sert à faire des tranches en fonction de la Prime.

```
# nouvelle variable "Primes2"
Prime2 <- character(length(voiture$Prime))
for (i in 1:length(voiture$Prime)) {ssS
  if (voiture$Prime[i] < 200) {
    Prime2[i] <- "faible"
  } else if (voiture$Prime[i] < 400) {
    Prime2[i] <- "moyenne"
  } else {
    Prime2[i] <- "élevée"
  }
}
voiture$Prime2 = Prime2
head(voiture)
```

Nous avons, ensuite affiché les 6 premières lignes de la nouvelle table obtenue.

```
> voiture$Prime2 = Prime2
> head(voiture)
   Marque PuissFisc Categorie Anciennete       Formule           Ville Prime  Prime2
1 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi     Chantecorps   171  faible
2 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi      Le Chillou   176  faible
3 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi        Loubillé   316 moyenne
4 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi Pougne-Herisson   310 moyenne
5 Citroen de_7_a_10   BERLINE    4-7 ans    Tiers Maxi     Villefollet   316 moyenne
6 Citroen de_7_a_10   BERLINE    4-7 ans Tous Risques      Chantecorps   444  élevée
```

Puis, on a construit pour chaque variable qualitative les tableaux croisés de la variable « Prime2 ».

```
# Construction des tableaux croisés
tableau_croise_Marque <- table(voiture$Prime2, voiture$Marque)
print(tableau_croise_Marque)
tableau_croise_PuissFisc <- table(voiture$Prime2, voiture$PuissFisc)
tableau_croise_categorie <- table(voiture$Prime2, voiture$Categorie)
tableau_croise_Anciennete <- table(voiture$Prime2, voiture$Anciennete)
tableau_croise_Formule <- table(voiture$Prime2, voiture$Formule)
```

Voici un exemple de tableau croisé obtenue entre la variable « Prime2 » et « Marque »

```
> print(tableau_croise_Marque)

         Citroen Peugeot  Renault
  élevée     105      48      105
  faible      28       4       28
  moyenne     47     128       47
```

A partir de ces 5 tableaux, nous avons effectuer un test du Khi-deux.

```
# test du khi-deux
Marque=chisq.test(tableau_croise_Marque)
Marque
PuissFisc=chisq.test(tableau_croise_PuissFisc)
categorie=chisq.test(tableau_croise_categorie)
Anciennete=chisq.test(tableau_croise_Anciennete)
Formule=chisq.test(tableau_croise_Formule)
```

Les résultats obtenus ont été mis dans une table que nous avons exporté au format CSV pour pouvoir surligner les tests les plus significatifs.

```
# Extraire les résultats
khi2 <- c(Marque$statistic,PuissFisc$statistic,categorie$statistic,Anciennete$statistic,Formule$statistic)
pvaleur <- c(Marque$p.value,PuissFisc$p.value,categorie$p.value,Anciennete$p.value,Formule$p.value)

# mettre les résultats du test du Khideux et de la P_value dans un data frame
tab_khi2<-data.frame(variables_quali=c('Marque','PuissFisc','Categorie','Anciennete','Formule'),khi2,pvaleur)

# extraction de la table tab_khi2
write.csv2(tab_result, file='tab_result.csv', row.names=FALSE)
```

Les Test les plus significatifs sont pour les variables : « Marque », « Anciennete » et « Formule »

variables_qua	khi2	pvaleur
Marque	103,494155	1,77E-21
PuissFisc	1,53714645	0,46367416
Categorie	8,5822753	0,0724322
Anciennete	42,4095537	1,37E-08
Formule	12,6750052	0,00176871

Et enfin on a calculé le V de Crammer pour les 3 variables les plus significatives.

```
# V de Crammer
# pour le tableau croisé Marque Prime2
n<- sum(tableau_croise_Marque)
p<- nrow(tableau_croise_Marque)
q<- ncol(tableau_croise_Marque)
m<- min(p-1,q-1)
v<- sqrt(Marque$statistic/(n*m))
v
# Pour le tableau croisé Anciennete et Prime2
n2<- sum(tableau_croise_Anciennete)
p2<- nrow(tableau_croise_Anciennete)
q2<- ncol(tableau_croise_Anciennete)
m2<- min(p2-1,q2-1)
v2<- sqrt(Anciennete$statistic/(n*m))
v2
# Pour le tableau croisé Formule et Prime2
n3<- sum(tableau_croise_Formule)
p3<- nrow(tableau_croise_Formule)
q3<- ncol(tableau_croise_Formule)
m3<- min(p3-1,q3-1)
v3<- sqrt(Formule$statistic/(n*m))
v3
```

Nous avons fait les mêmes choses que pour le khi-deux, c'est-à-dire que nous avons extrait notre table sous la forme d'un fichier CSV pour pouvoir indiquer quel est le résultat le plus significatif.

```
# récupération de tous les résultats du VCrammer
vcrammer<-c(v,v2,v3)

# création d'un nouveau data frame qui récupère les résultat du V de crammer
tab_vcrammer <- data.frame(variables_quali=c('Marque','Anciennete','Formule'),vcrammer)

#extraction du fichier tab_vcrammer
write.csv2(tab_vcrammer, file='tab_vcrammer.csv', row.names=FALSE)
```

variables_quali	vcrammer
Marque	0,30956085
Anciennete	0,19816182
Formule	0,10833336

On peut conclure que le V ce Crammer le plus significatif est celui du croisement entre la variable « Prime2 » et « Marque ».

Ce qui signifie que la plus forte association avec la variable « Prime2 » est la variable « Marque ». Cette association est donc modérée.