4. Spécification et Estimation

Dans la **section 3**, nous nous sommes intéressés à l'estimation de la moyenne, ainsi que de l'erreur-type, dont les formules sont exactes. Il s'agissait d'estimations ponctuelles dans le cas univarié. Dans ce cas, il est assez intuitif détudier la précision de la statistique estimant le paramètre de la loi d'une variable, le paramètre de la superpopulation !

Quand les fonctions des variables sont plus compliquées, que les lois suivies par les statistiques ne sont connues, ni exactement, ni pour un n donné, mais seulement asymptotiquement, ou que la supposition d'indépendance ne tient pas, on peut avoir recours à d'autres méthodes d'estimation avec Stata.

L'estimation dépend largement de la nature des variables (dichotomique, discrète, continue), et de la **spécification** du modèle. Dans le cas multivarié, la spécification concerne la forme des relations entre les variables (log-linéaire, autorégressive, système d'équations, etc.).

[Discuter de la nature des variables]

Nous verrons la spécification des modèles en mêeme temps que des méthodes d'estimation dans les sous-section qui suivent. On se place dans le cadre du modèle d'échantillonnage.

4.1 Introduction

L'estimation peut porter sur le paramètre d'une loi. La méthode du **maximum de vraissemblance** (MV) est appropriée dans ce cas. On peut aussi estimer un paramètre sans être obligé de supposer une loi, avec, par exemple, la méthode des **moindres carrés**, inventée par Gauss-Legendre.

Nous verrons des problèmes d'estimation ponctuelle dans le cas multivarié, et d'estimation par intervalle de confiance, notamment l'estimation bootstrap d'un paramètre, qui fonctionne dans les cas univarié et multivarié. Il y a aussi la méthode des moments, qui est disponible grâce à la commande `gmm` où `gmm' est l'abréviation de *general method of moments*. Nous ferons un petit exercice, afin de donner une intuition de la méthode, que vous verrez plus amplement dans d'autres cours, notamment l'an prochain.

Nous ne verrons pas l'approche bayésienne de l'estimation d'un paramètre. Avec cette approche, on s'enfonce un peu plus dans l'estimation paramétrique, avec des hypothèses *a priori* sur ces paramètres. Par exemple, μ, le moment centré d'ordre un d'une variable normale de paramètres μ et σ^2, est lui-même borné entre 0 et 1 selon la loi $U[0,1]$. On peut facilement produire des valeurs pour ce type de variable avec Stata, en suivant la méthode Monte Carlo comme on l'a déjà vu. Vous pouvez consulter le **Stata Bayesian Analysis Reference Manual** si vous étiez amené.e.s à faire du bayésien !

4.2 Maximum de vraisemblance, `ml`

La commande `ml` est puissante lorsqu'il s'agit d'obtenir un estimateur difficile à déduire à la main. La méthode consiste à trouver la forme de la statistique qui maximise une fonction des observations, la **fonction de vraisemblance**. La méthode est très efficace lorsque le modèle n’est pas linéaire par rapport aux paramètres. Un exemple emblématique est celui de la **loi logistique**.

Considérons par exemple le cas d’une variable Y dichotomique, qui suit non pas une loi de Bernoulli, mais une loi logistique à valeur dans $\{0,1\}$. Nous rentrons doucement dans le cadre multivarié, en supposant que $Pr(Y = 1|X = x)$ est la fonction de répartition cumulée logistique :

$$Pr(Y = 1|X = x) = \frac{e^{\beta_1+\beta_2 x}}{1 + e^{\beta 1+\beta_2 x}} \equiv \Lambda(x, \mathbf{\beta}).$$

$\Lambda(x, \mathbf{\beta})$ ne comporte que deux paramètres, mais étant non-linéaire par rapport à ces derniers, les conditions de premier ordre sont compliquées. Comme on peut le voir, la probabilité que Y se réalise est une fonction croissante de x et tend vers 0 quand x tend vers $-\infty$. Et, $ln(P/(1 - P)) = \beta_1 + \beta_2 x$ est une fonction linéaire de $\mathbf{\beta}$. Le ratio $P/(1 - P)$ s’appelle **ratio de chance**, et le logarithme (népérien) de ce ratio s’appelle **fonction logit**. Cette fonction intervient lors de la recherche de l’estimateur du MV.

Supposons un échantillon aléatoire. La loi jointe de $\mathbf{y}$ se note généralement $L(\mathbf{y}|\mathbf{\beta})$. L’écriture de la fonction de vraissemblance de léchantillon “renverse” le conditionnement, $L(\mathbf{\beta}|\mathbf{y})$ pour indiquer que c’est $\mathbf{\beta}$ qui varie, $\mathbf{y}$ est donné. Le problème est alors le suivant :

$$\hat{\mathbf{\beta}}= argmax_{\mathbf{\beta}}\{L(\mathbf{\beta}|\mathbf{y})\}.$$

Notons $Pr(Y_i = 1|X_i = x_i) \equiv p_i(x_i)$. La vraissemblance s’écrit :

$$L(\mathbf{\beta}|\mathbf{y} = \Pi_{i=1}^{n} p_i(x_i)^{y_i}(1 - p_i(x_i))^{1-y_i}.$$

Faisons un petit détour par la loi de Bernoulli. Pour cela, supposons que p_i ne varie pas avec i ($\beta_1 \equiv 0$ et $x_i \equiv 1\ \forall i$). Dans ce cas, on a $\Lambda(x, \mathbf{\beta}) = e^{\beta_2}/(1 + e^{\beta_2}) \equiv p$. Notons que si je connais β_2, je connais p, est *vice-versa*. Que devient la vraissemblance dans ce cas ?

Depuis Fisher, inventeur de la méthode, on s’intéresse à la log-vraissemblance, $ln(L(p|\mathbf{y}))$, que l’on note $l(p|\mathbf{y})$. Nous avons :

$$l(p|\mathbf{y}) = \sum^{\substack{n \\ i\leq}} (y_i ln(p) + (1 - y_i) ln(1 - p))$$

$$= ln(p)n\overline{y} + ln(1 - p)n(1 - \overline{y}).$$

L’estimateur du MV est simple dans ce cas : $\hat{p} = \overline{y}$.

On peut utiliser la commande `ml`, que l’on peut appliquer à l’estimation du paramètre d’une variable aléatoire qui suit une loi logistique, de Poisson, etc. On peut très bien utiliser la commande `logit`, mais aussi `ml`, ou encore utiliser **Mata** pour aller encore plus dans les aspects techniques (nous avons fait un petit programme qui résout les équations non-linéaires induites par la maximisation de la vraisemblance).

```
. cls

. clear   all


. use             "http://www.stata-press.com/data/r13/repair", clear
(1978 Automobile Data)

. set more off

. keep            foreign repair

. tabulate        foreign repair, cell

+-----------------+
| Key             |
|-----------------|
|    frequency    |
| cell percentage |
+-----------------+

           |              repair
  Car type |         1          2          3 |     Total
-----------+---------------------------------+----------
  Domestic |        10         27          9 |        46
           |     17.24      46.55      15.52 |     79.31
-----------+---------------------------------+----------
   Foreign |         0          3          9 |        12
           |      0.00       5.17      15.52 |     20.69
-----------+---------------------------------+----------
     Total |        10         30         18 |        58
           |     17.24      51.72      31.03 |    100.00

. logit           foreign repair

Iteration 0:  Log likelihood = -29.569311
Iteration 1:  Log likelihood = -23.253386
Iteration 2:  Log likelihood = -22.346332
Iteration 3:  Log likelihood = -22.341067
Iteration 4:  Log likelihood = -22.341067

Logistic regression                                     Number of obs =     58
                                                        LR chi2(1)    =  14.46
                                                        Prob > chi2   = 0.0001
Log likelihood = -22.341067                             Pseudo R2     = 0.2445

------------------------------------------------------------------------------
     foreign | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      repair |   2.298023   .7274809     3.16   0.002     .8721868     3.72386
       _cons |  -6.871362   1.939831    -3.54   0.000    -10.67336   -3.069364
------------------------------------------------------------------------------

. logit           foreign repair, or

Iteration 0:  Log likelihood = -29.569311
Iteration 1:  Log likelihood = -23.253386
Iteration 2:  Log likelihood = -22.346332
Iteration 3:  Log likelihood = -22.341067
Iteration 4:  Log likelihood = -22.341067

Logistic regression                                     Number of obs =     58
                                                        LR chi2(1)    =  14.46
                                                        Prob > chi2   = 0.0001
Log likelihood = -22.341067                             Pseudo R2     = 0.2445

------------------------------------------------------------------------------
     foreign | Odds ratio   Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      repair |   9.954486   7.241699     3.16   0.002     2.392136    41.42397
       _cons |   .0010371   .0020117    -3.54   0.000     .0000232    .0464507
```

```
------------------------------------------------------------------------------
Note: _cons estimates baseline odds.

. margins         , dydx(repair) at(repair==2)

Conditional marginal effects                               Number of obs = 58
Model VCE: OIM

Expression: Pr(foreign), predict()
dy/dx wrt:  repair
At: repair = 2

------------------------------------------------------------------------------
             |            Delta-method
             |      dy/dx   std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      repair |   .1941922   .0598857     3.24   0.001     .0768184     .311566
------------------------------------------------------------------------------


. predict         FOREIGNP, pr

. sort            repair FOREIGNP
```

On peut reestimer le logit avec la troisieme valeur de la variable `repair` en groupe de base

```
. logit foreign ib3.repair, or baselevels

note: 1.repair != 0 predicts failure perfectly;
      1.repair omitted and 10 obs not used.

Iteration 0:  Log likelihood = -26.992087
Iteration 1:  Log likelihood = -22.483187
Iteration 2:  Log likelihood = -22.230498
Iteration 3:  Log likelihood = -22.229139
Iteration 4:  Log likelihood = -22.229138

Logistic regression                                     Number of obs =     48
                                                        LR chi2(1)    =   9.53
                                                        Prob > chi2   = 0.0020
Log likelihood = -22.229138                             Pseudo R2     = 0.1765

------------------------------------------------------------------------------
     foreign | Odds ratio   Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      repair |
          1  |          1  (empty)
          2  |   .1111111   .0855334    -2.85   0.004     .0245755    .5023573
          3  |          1  (base)
             |
       _cons |          1   .4714045     0.00   1.000     .3969536    2.519186
------------------------------------------------------------------------------
Note: _cons estimates baseline odds.

. tabulate repair, generate(repair_d)

     repair |      Freq.     Percent        Cum.
------------+-----------------------------------
          1 |         10       17.24       17.24
          2 |         30       51.72       68.97
          3 |         18       31.03      100.00
------------+-----------------------------------
      Total |         58      100.00

. logit foreign repair_d1 repair_d2, or

note: repair_d1 != 0 predicts failure perfectly;
      repair_d1 omitted and 10 obs not used.

Iteration 0:  Log likelihood = -26.992087
```

```
Iteration 1:  Log likelihood = -22.483187
Iteration 2:  Log likelihood = -22.230498
Iteration 3:  Log likelihood = -22.229139
Iteration 4:  Log likelihood = -22.229138

Logistic regression                                     Number of obs =     48
                                                        LR chi2(1)    =   9.53
                                                        Prob > chi2   = 0.0020
Log likelihood = -22.229138                             Pseudo R2     = 0.1765

------------------------------------------------------------------------------
     foreign | Odds ratio   Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
   repair_d1 |          1  (omitted)
   repair_d2 |   .1111111   .0855334    -2.85   0.004     .0245755    .5023573
       _cons |          1   .4714045     0.00   1.000     .3969536    2.519186
------------------------------------------------------------------------------
Note: _cons estimates baseline odds.

. program MYLOGIT
.  version 13
.  args lnf THETA
.  quietly: replace `lnf' = -ln(1+exp(-`THETA')) if $ML_y1==1
.  quietly: replace `lnf' = -`THETA' - ln(1+exp(-`THETA')) if $ML_y1==0
. end

. ml model lf MYLOGIT (foreign = repair)


. ml maximize

Initial:      Log likelihood = -40.202536
Alternative:  Log likelihood = -33.496465
Rescale:      Log likelihood = -30.169178
Iteration 0:  Log likelihood = -30.169178
Iteration 1:  Log likelihood = -22.986181
Iteration 2:  Log likelihood = -22.349011
Iteration 3:  Log likelihood = -22.341073
Iteration 4:  Log likelihood = -22.341067
Iteration 5:  Log likelihood = -22.341067

                                                        Number of obs =     58
                                                        Wald chi2(1)  =   9.98
Log likelihood = -22.341067                             Prob > chi2   = 0.0016

------------------------------------------------------------------------------
     foreign | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      repair |   2.298023   .7274809     3.16   0.002      .872187     3.72386
       _cons |  -6.871362   1.939831    -3.54   0.000    -10.67336   -3.069364
------------------------------------------------------------------------------
```

Nous allons maintenant résoudre le systeme de deux equations non-lineaires en β_1 et β_2 dans **Mata**.

```
. clear   all


. use             "http://www.stata-press.com/data/r13/repair", clear
(1978 Automobile Data)

. set more off

. keep            foreign repair

. logit           foreign repair

Iteration 0:  Log likelihood = -29.569311
Iteration 1:  Log likelihood = -23.253386
```

```
Iteration 2:  Log likelihood = -22.346332
Iteration 3:  Log likelihood = -22.341067
Iteration 4:  Log likelihood = -22.341067

Logistic regression                                     Number of obs =     58
                                                        LR chi2(1)    =  14.46
                                                        Prob > chi2   = 0.0001
Log likelihood = -22.341067                             Pseudo R2     = 0.2445

------------------------------------------------------------------------------
     foreign | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
      repair |   2.298023   .7274809     3.16   0.002     .8721868     3.72386
       _cons |  -6.871362   1.939831    -3.54   0.000    -10.67336   -3.069364
------------------------------------------------------------------------------

. predict         FOREIGNP, pr

. table foreign FOREIGNP if repair==1, nototals //, cellwidth(10) in Stata 13

-------------------------
           |  Pr(foreign)
           |     .0102179
-----------+-------------
Car type   |
  Domestic |           10
-------------------------

. table foreign FOREIGNP if repair==2, nototals

-------------------------
           |  Pr(foreign)
           |      .093188
-----------+-------------
Car type   |
  Domestic |           27
  Foreign  |            3
-------------------------

. table foreign FOREIGNP if repair==3, nototals

-------------------------
           |  Pr(foreign)
           |     .5056766
-----------+-------------
Car type   |
  Domestic |            9
  Foreign  |            9
-------------------------

. mata:
------------------------------------------------- mata (type end to exit) -------------------------
----------------------------------------
:  void function myfun2(real colvector x, real colvector values)  {  values[1] =   -
10*exp(x[1]+x[2])/(1+exp(x[1]+x[2])) - ///
30*exp(x[1]+2*x[2])/(1+exp(x[1]+2*x[2])) - ///
18*exp(x[1]+3*x[2])/(1+exp(x[1]+3*x[2])) + 12   values[2] =   -10*exp(x[1]+x[2])/(1+exp(x[1]+x[2])) - ///
60*exp(x[1]+2*x[2])/(1+exp(x[1]+2*x[2])) - ///
54*exp(x[1]+3*x[2])/(1+exp(x[1]+3*x[2])) + 33   }

:  S = solvenl_init()


:  solvenl_init_evaluator(S, &myfun2())


:  solvenl_init_type(S, "zero")


:  solvenl_init_technique(S, "newton")
```

```
      :  solvenl_init_numeq(S, 2)


      :  solvenl_init_startingvals(S, J(2,1,0))


      :  solvenl_init_iter_log(S, "on")


      :  x = solvenl_solve(S)
      Iteration 0:  Function =      1130
      Iteration 1:  Function = 28.394091  delta X = 3.5076214
      Iteration 2:  Function = 2.8026749  delta X = .47183187
      Iteration 3:  Function = .09205889  delta X =  .1565965
      Iteration 4:  Function = .00009591  delta X = .02503346
      Iteration 5:  Function = 8.894e-11  delta X = .00074643

      : x
                       1
          +----------------+
        1 |  -6.871356767  |
        2 |   2.298021399  |
          +----------------+

      : end
      ----------------------------------------------------------------------------------------------------
-----------------------------------------
```

Notons que nous pourrions estimer le modèle Probit de la même manière, ainsi que d'autres modèles à variable dépendante qualitative, tels que le modèle de Poisson, ou une généralisation de la fonction logistique, avec le modèle multinomial.

4.2 Moindres carrés, `regress`

Dans le cas univarié, la méthode des moindres carrés (MC) permet de trouver une estimation du moment centré d'ordre 1 d'une variable qui minimise sa variance :

$$\mu^{MC} \equiv argmin_{\mu} n^{-1} \sum^{i} (y_i - \mu)^2.$$

On la retrouve en économétrie où la moyenne est en fait l'espérance conditionnelle à une ou plusieurs autre variables, $\mu \equiv E(Y_i|X_1, X_2, \dots, X_K) = \beta_1 X_1 + \cdots \beta_K X_K$.

La commande `regress` de **Stata** gère ces deux situations sans problème. Dans le premier cas, il suffit de faire une régression de Y sur une constante, en tapant tout simplement `regress Y`. Dans le second cas, une régression sur les différentes variables, par exemple X_1 et X_2 : `regress Y X1 X2 ... X3`

Les justifications de la méthode sont dans tous les livres d'économétrie, dont les deux plus fameux (Wooldridge, 2010, et Greene, 2014). Ses inconvénients sont aussi dans ces livres, ainsi que dans ceux sur l'**apprentissage automatique**. On peut sans difficulté utiliser **Stata** pour calculer les coefficients d'un modèle en manipulant les matrices.

```
      . use     "http://www.evens-salies.com/rubin1977.dta", clear

      . rename (GROUP POSY PREX)(D Y X)


      . regress Y D X
```

```
      Source |       SS           df       MS      Number of obs   =        72
-------------+----------------------------------   F(2, 69)        =     21.59
       Model |  318.806297         2  159.403149   Prob > F        =    0.0000
    Residual |  509.513147        69  7.38424851   R-squared       =    0.3849
-------------+----------------------------------   Adj R-squared   =    0.3671
       Total |  828.319444        71   11.666471   Root MSE        =    2.7174

------------------------------------------------------------------------------
           Y | Coefficient  Std. err.      t    P>|t|     [95% conf. interval]
-------------+----------------------------------------------------------------
           D |   3.666846    .679734     5.39   0.000     2.310813    5.022878
           X |   .7180273   .1599805     4.49   0.000     .3988749     1.03718
       _cons |   2.330691    1.01652     2.29   0.025     .3027893    4.358593
------------------------------------------------------------------------------


. generate ONE=1

. mkmat ONE D X, matrix(X)


. mkmat Y, matrix(Y)


. matrix B=invsym(X'*X)*X'*Y

. matrix list B

B[3,1]
             Y
ONE  2.3306913
  D  3.6668456
  X   .7180273
```

Dans le cas du modèle linéaire simple, il y a un lien fort avec le coefficient de corrélation de pearson, le fameux ρ. Les résultats de __`regress`__ne renvoient pas les coefficients de correlation de Spearman, tetrachorique (dans le cas de deux variables dichotomiques) car, par définition, la regression est pour des variables continues. La statistique R^2 affichée, après avoir exécuté la commande, est le coefficient de Pearson au carré.

4.3 Estimation bootstrap, `bootstrap`

Le bootstrap est au départ une méthode d'inférence. Avant de voir un exemple sur des données d'observation, examinons le cas simpliste d'un échantillon aléatoire ayant la même taille que la population, parce que la population est de petite taille par exemple ($N = 3, n \equiv N$). Nous allons introduire la commande **`bootstrap`** dans ce cas. Nous allons produire 1000 échantillons bootstrap, sachant très bien qu'il n'y en a que 10 échantillons distincts.

En effet, à partir de $\mathcal{S} = \{1,2,3\}$, on a les échantillons suivants : $\{1,1,1\}$, $\{2,2,2\}$, $\{3,3,3\}$, $\{1,1,2\}$, $\{2,2,1\}$, $\{1,1,3\}$, $\{3,3,1\}$, $\{2,2,3\}$, $\{3,3,2\}$, $\{1,2,3\}$. Supposons que les valeurs de la variable aléatoire soient $y_1 = 2$, $y_2 = 1$ et $y_3 = 3$. Il y a sept sommes, et donc sept moyennes différentes.

Ces 10 échantillons donnent les moyennes suivantes : $\frac{2+2+2}{3} = 2$, $\frac{1+1+1}{3} = 1$, ..., $\frac{2+2+3}{3} = 7/3$ ou $\frac{3+3+1}{3} = 7/3$, $\frac{3+3+2}{3} = 8/3$. Vous pourrez vérifier qu'il y a sept moyennes différentes. Pour connaître la proportion de chaque moyenne, il faut revenir au nombre déchantillons possibles : $3^3 = 27$. Les différentes moyennes avec leur fréquence relative théorique entre parenthèses, sont les suivantes : $3/3 = 1(1/27)$ (l'échantillon $\{2,2,2\}$), $4/3 = 1{,}33\ldots(3/27)$ (les échantillons $\{1,2,2\}$, $\{2,1,2\}$, $\{2,2,1\}$), $5/3 = 1,66 \ldots (6/27)$ (les échantillons $\{1,1,2\}$, $\{1,2,1\}$, $\{2,1,1\}$, $\{2,2,3\}$, $\{2,3,2\}$, $\{3,2,2\}$), …, $9/3 = 3(1/27)$ (l'échantillon $\{3,3,3\}$).

Mais on peut aussi tout simplement ajouter les sept sommes différentes que nous avons trouvées, ce qui donne 42, puis diviser cette somme par 7 et par 3. Le résulat vaut 2.

```
      . set seed        21041971

      . clear all


      . set     obs     3
      Number of observations (_N) was 0, now 3.

      . input int Y

                  Y
        1. 2
      . 1
      . 3

      . cls

      . list

           +---+
           | Y |
           |---|
        1. | 2 |
        2. | 1 |
        3. | 3 |
           +---+

      . generate        MEAN=.
      (3 missing values generated)

      . bootstrap       MEAN=r(mean), ///
        size(3) reps(1000) saving(bootstrap123, replace): summarize Y

      (running summarize on estimation sample)

      warning: summarize does not set e(sample), so no observations will be excluded from the resampling
because of missing values or other
               reasons. To exclude observations, press Break, save the data, drop any observations that
are to be excluded, and rerun bootstrap.

      Bootstrap replications (1,000):
.........10.........20.........30.........40.........50.........60.........70.........80.........90.......
..100.........110.........120.........130.........140.........150.........160.........170.........180.....
....190.........200.........210.........220.........230.........240.........250.........260.........270...
......280.........290.........300.........310.........320.........330.........340.........350.........360.
........370.........380.........390.........400.........410.........420.........430.........440.........45
0.........460.........470.........480.........490.........500.........510.........520.........530.........
540.........550.........560.........570.........580.........590.........600.........610.........620.......
..630.........640.........650.........660.........670.........680.........690.........700.........710.....
....720.........730.........740.........750.........760.........770.........780.........790.........800...
......810.........820.........830.........840.........850.........860.........870.........880.........890.
........900.........910.........920.........930.........940.........950.........960.........970.........98
0.........990.........1,000 done

      Bootstrap results                                        Number of obs =     3
                                                               Replications  = 1,000

            Command: summarize Y
               MEAN: r(mean)

      ------------------------------------------------------------------------------
                   |   Observed   Bootstrap                         Normal-based
                   | coefficient  std. err.      z    P>|z|     [95% conf. interval]
```

```
-------------+----------------------------------------------------------------
        MEAN |          2   .4735081     4.22   0.000     1.071941    2.928059
------------------------------------------------------------------------------

. use                    "C:\Users\evens\Documents\bootstrap123.dta", clear
(bootstrap: summarize)

. generate               ONE=1

. collapse (sum)  ONE, by(MEAN)


. summarize              ONE

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
         ONE |          7    142.8571    92.89497         29        267

. generate               ONEP=100*ONE/r(sum)

. browse
```

La commande `bootstrap` a pour argument le nom de la variable dans laquelle on va mettre la statistique qui nous intéresse. Ici, c'est la variable `MEAN`. Cette variable contiendra une statistique post-commande, `r(mean)` calculée par `summarize`.

Il y a 1000 TAR, et les 1000 moyennes sont placées dans le fichier de données **bootstrap123.dta** dans votre dossier de travail. Après la commande `bootstrap`, le programme calcule la fréquence relative de chaque moyenne.

La commande calcule un intervalle de confiance bootstrap de la moyenne, à partir d'une erreur standard. La question est de savoir laquelle.

Prenons un exemple de 23 capitalisations boursières dans le secteur mondial du numérique. Chacune des 23 capitalisations est la réalisation d'une variable aléatoire dont on ne connait pas la loi sous-jacente. Dans ce cas, le **bootstrap non-paramétrique** peut être utilisé, en appliquant un re-échantillonnage basé sur toutes les observations. Si $n = 23$ est la taille de l'échantillon, on crée un certains nombre d'échantillon bootstrap de 23 observations chacun ; le nombre d'échantillons possibles (non-distincts) est 23^{23}, c'est énorme !

Le programme suivant calcule l'intervalle de confiance de la moyenne des capitalisations. Nous allons nous contenter de tirer 100 échantillons. Vous pouvez ensuite essayer de voir quel est le résultat obtenu avec 500. Plutôt que de placer les fréquences relatives des différentes moyennes dans la feuille de données, on va faire un histogramme de ces moyennes avec la commande `hist MEAN`.

```
. use     "http://www.evens-salies.com/statainitiation_3_capitalisation.dta", clear

. rename          (var*)(CAP STR TEMP)


. encode          TEMP, generate(NOM)

. drop            TEMP

. set seed        21041971

. bootstrap       MEAN=r(mean), size(23) reps(1000) ///
  saving(bootstrap_capitalisation, replace) nowarn mse: summarize CAP
```

```
(running summarize on estimation sample)

Bootstrap replications (1,000):
.........10.........20.........30.........40.........50.........60.........70.........80.........90.......
..100.........110.........120.........130.........140.........150.........160.........170.........180.....
....190.........200.........210.........220.........230.........240.........250.........260.........270...
......280.........290.........300.........310.........320.........330.........340.........350.........360.
........370.........380.........390.........400.........410.........420.........430.........440.........45
0.........460.........470.........480.........490.........500.........510.........520.........530.........
540.........550.........560.........570.........580.........590.........600.........610.........620.......
..630.........640.........650.........660.........670.........680.........690.........700.........710.....
....720.........730.........740.........750.........760.........770.........780.........790.........800...
......810.........820.........830.........840.........850.........860.........870.........880.........890.
........900.........910.........920.........930.........940.........950.........960.........970.........98
0.........990.........1,000 done

Bootstrap results                                        Number of obs =     23
                                                         Replications  = 1,000

      Command: summarize CAP
         MEAN: r(mean)

------------------------------------------------------------------------------
             |   Observed   Bstrap *
             | coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
        MEAN |   254.9565   61.54831     4.14   0.000     134.3241     375.589
------------------------------------------------------------------------------

. preserve

.  use            "bootstrap_capitalisation.dta", clear
(bootstrap: summarize)

.  hist           MEAN
(bin=29, start=88.173912, width=12.616192)


. restore
```

Nous n'avons pas utilisé l'information capturée par la variable de stratification. Si l'on pense que la moyenne varie d'une strate à l'autre, de sorte que $Y_{is} = \mu_s + U_i$ est le modèle pertinant, avec par exemple $U_i \sim i.i.d.$ (la valeur de Y_{is} est déterminée par un PGD inconnu).

On peut combiner `regress` avec l'option `bootstrap` pour l'estimation de l'erreur-type du coefficient d'un modèle de régression. Dans l'exemple ci-dessous, le coefficient est unique, c'est la constante dans une modèle de régression (il n'y a pas de variable explicative autre que la constante). On sait que dans ce cas, l'estimateur de la constante est la capitalisation moyenne.

```
. use     "http://www.evens-salies.com/statainitiation_3_capitalisation.dta", clear

. rename                 (var*)(CAP STR TEMP)


. encode                 TEMP, generate(NOM)

. drop                   TEMP

. order                  NOM

. sort                   NOM

. save                   "statainitiation_3_capitalisation_final.dta", replace
file statainitiation_3_capitalisation_final.dta saved

. set seed               21041971
```

```
. regress                 CAP, vce(bootstrap)     // Dans ce cas, s.e. sert a construire IC
(running regress on estimation sample)

Bootstrap replications (50): .........10.........20.........30.........40.........50 done

Linear regression                                    Number of obs =       23
                                                     Replications  =       50
                                                     Wald chi2(0)  =        .
                                                     Prob > chi2   =        .
                                                     R-squared     =   0.0000
                                                     Adj R-squared =   0.0000
                                                     Root MSE      = 299.8770

------------------------------------------------------------------------------
             |   Observed   Bootstrap                         Normal-based
         CAP | coefficient  std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
       _cons |   254.9565   62.11889     4.10   0.000     133.2057    376.7073
------------------------------------------------------------------------------

. matrix list     e(b)

symmetric e(b)[1,1]
        _cons
y1  254.95652

. matrix list     e(V)                    // S^2/23

symmetric e(V)[1,1]
           _cons
_cons  3858.7563

. matrix define   B=J(1,1,0)

. matrix define   V=J(1,1,0)

. matrix define   B[1,1]=e(b)

. matrix define   V[1,1]=e(V)

. local                   B=B[1,1]

. local                   SE=V[1,1]^.5

. local                   ICL=`B'-invnormal(0.975)*`SE'

. di                              `ICL'
133.20574
```

On peut vérifier avec `summarize` CAP que l'unique coefficient estimé est bien la moyenne de la variable. L'avantage de `regress` est que nous avons au passage une estimation d'un intervalle de confiance pour la moyenne. Dans le cas d'un re-échantillonnage, l'estimateur bootstrap de l'erreur standard, qui vaut 62.11 dans mon cas, sert à calculer l'intervalle de confiance au seuil de 5 %, comme on peut le vérifier :

$$254.95 \pm 1{,}96 \times 62{,}11 \Leftrightarrow 254.95 \pm 121{,}75 \Leftrightarrow [133{,}20; 376{,}70].$$

Vous verrez des méthodes plus compliquées cette année ou l'an prochain. Je pense par exemple à la méthode des moments généralisés, avec la commande `gmm`. Je vous conseille de revoir cette méthode dans le cas univarié, le cas "simple" par opposition à "généralisé" qui est le cadre dans lequel cette commande a été créée).

Il y a un estimateur connu en économétrie qui s'appuie dessus. C'est celui d'Arellano et Bond, pour l'estimation des coefficients d'un modèle dynamique sur données de panel. Des informations sur l'application de la méthode dans le cas de petits échantillons sont disponibles dans Drukker (2010).

Bibliographie

Drukker, D.M. 2010. An introduction to GMM estimation using Stata (slides). In German Stata Users' Group.

```
. quietly log close
```