2. Distributions de probabilités, simulation

Dans cette section, nous suivons une approche fréquentiste (par opposition à bayésienne) et paramétrique (on peut consulter Efron et Hastie, 2016 pour la première approche, et Greene, 2008, pp. 398- pour la seconde)

La loi d'une variable aléatoire X, représentee généralement par une mesure de probabilité, est une loi théorique (abstraite). Les éeconomètres ont montré l'utilité de maîtriser des lois théoriques pour l'étude des phénomènes économiques.

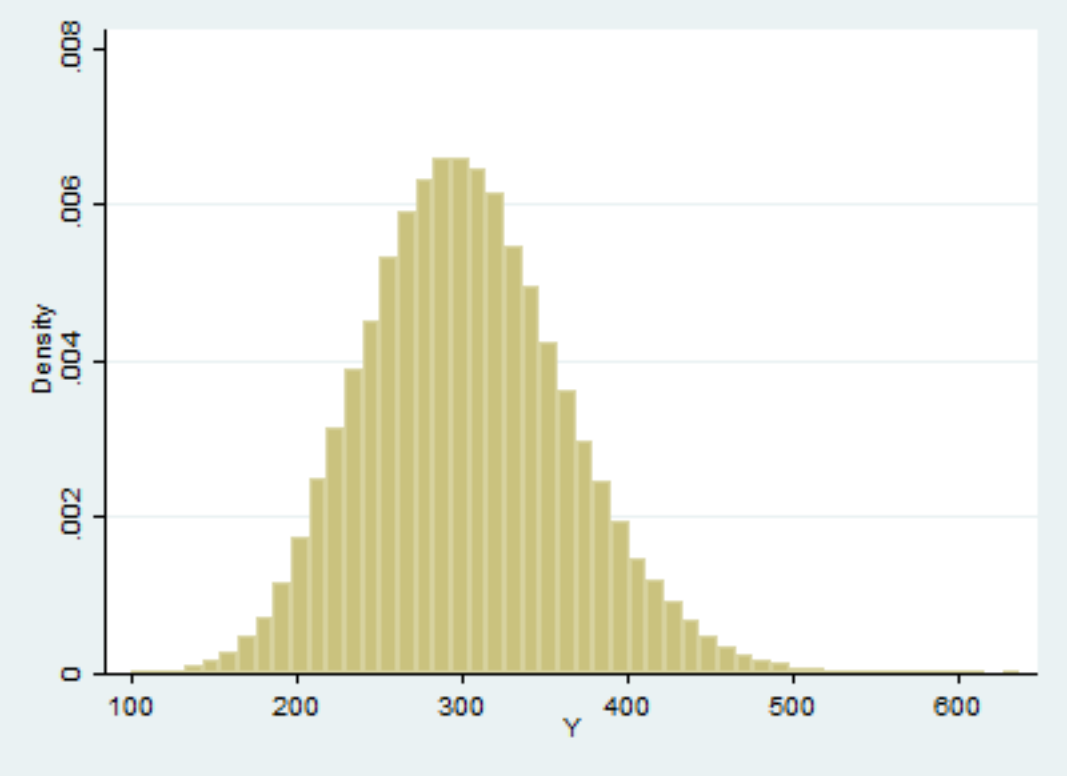
Il existe plus d'une centaine de lois statistiques. En revanche, il suffit d'ouvrir un manuel d'Econométrie, ou de Statistique pour économistes pour voir que seulement une vingtaine sont d'usage courant en Economie (Mooney, 1997, p. 8). On peut apprendre plus facilement ces lois en se focalisant sur les relations qui existent entre elles ; la loi d'une variable Y peut se déduire de celle de X de deux manières : en contraignant les paramètres de la loi de X ou après transformation de X (le "ou" est non-exclusif). Leemis et McQueston (2008) ont établi un schéma des "proximités" qui existent entre une centaine de lois.

[**Distribuer le schéma des distributions 2021_StataInitiation_2_Distributions.pdf**]

Stata permet de simuler la loi d'une famille connue, en tirant (*draw*) les valeurs d'une variable X suivant cette loi, puis tracer une distribution de probabilités empirique pour cette variable. La simulation permet aussi d'étudier aisément la distribution de probabilités d'une fonction de variables aléatoires $X_1, X_2, \ldots$, un ratio par exemple, $Y \equiv \frac{X_1}{X_2}$. C'est pratique quand on ne connaît pas les propriétés de Y (moments d'ordre 1 et 2, médiane, probabilité d'un événement, etc.), ou que l'on ne sait pas les calculer à la main.

Exemple. Votre spécification économétrique inclut le produit de deux variables normales, $X_1 \sim N(10,1)$, $X_2 \equiv 3N(10,1) + N(0,1)$, mais vous ne savez pas comment se comporte l'interaction entre les deux. Vous pouvez simuler 100000 valeurs pour chacune, prendre le produit des valeurs $y_i \equiv x_{1,i}x_{2,i}$, $i = 1, \ldots, 100000$, tracer un histogramme, faire un `summarize` de y. Le plus souvent, deux-trois commandes suffisent ; une seule commande permet de simuler les fractiles d'une loi, et Stata peut produire autant d'observations que l'on veut, autant qu'il y a d'observations déclarées dans `set obs`. Ces observations sont soit indépendantes, comme pour un tirage avec remise (TAR) dans une population, soit pas.

Graphique x. Histogramme de $Y \equiv X_1 X_2$, $X_1 \sim N(10,1)$ $X_2 \sim 3N(10,1) + N(0,1)$ ($n = 10^5$).



Ce résultat aurait pu être déduit analytiquement. La variable $N(0,1)$ dans X_2 affecte peu Y, de sorte que $X_1 X_2 \sim 3 \times N(10,1)^2$, soit la somme des carrés de trois normales, chacune de variance 1. Cette somme suit un $\chi^2(3)$ décentré (Tassi, 2004, pp. 34-35), de paramètre de décentrage $3 \times 10^2 = 300$, de moyenne et variance théoriques 3+300=303 et $2(3 + 2 \times 300) = 1206$.

Nous allons simuler des variables aléatoires qui suivent des lois connues. Nous verrons les commandes graphiques pour tracer leurs distributions de probabilités empiriques (fonctions de masse dans le cas discret, densités dans le cas continue). Nous calculerons les moments empiriques d'ordre 1 et 2, du fractile théorique d'ordre 1/2 (la médiane), etc. Puis, nous verrons que la simulation est très efficace pour comprendre le théorème central limite, ainsi que pour l'étude de fonctions de variables aléatoires. Pour effectuer des simulations, Stata s'appuie sur la méthode Monte Carlo, sur laquelle nous reviendrons plusieurs fois dans ce cours. Cette méthode permet d'étudier le comportement de statistiques dans des situations problématiques en pratique (petit échantillon, conditions d'utilisation d'un estimateur pas remplies, etc.)

Rappel : qu'est-ce qu'une variable aléatoire ?

Stata a besoin de nombres dans la feuille de données pour pouvoir calculer. Par exemple, les événements "homme", "femme" et "non-binaire" sont codés 0, 1 et 2. L'occurrence de ces événements aussi nous intéresse. Le codage renvoie à la notion de variable aléatoire. De ce point de vue, les valeurs d'une v.a. (une colonne) dans la feuille de données sont des réalisations de cette v.a.

Rappelons qu'une variable aléatoire (réelle) est une fonction Y (un codage) d'un ensemble Ω, contenant les résultats fondamentaux d'une expérience aléatoire, vers IR, qui à tout événement ω de Ω, associe la valeur $Y(\omega)$. L'ensemble Ω est souvent une modélisation restrictive des événements qui nous intéressent. Prenons un ex. de choix modal de transport : $\Omega = \{voiture, bus, vélo\}$ (cet ensemble s'appelle aussi "univers", *sample space* en anglais). Cette présentation ne tient volontairement pas compte des déterminants (temps, revenu, etc.) du choix d'un mode de transport par un individu, ni des charactéristiques des modes de transport.

Y est une variable réelle, ce qui permet le calcul de moments et autres paramètres ; par exemple, $Y(\{voiture\}) \equiv 0$, $Y(\{bus\}) \equiv 1$ et $Y(\{vélo\}) \equiv 2$ (0, 1 et 2 sont bien dans IR). Y est ici discrète. On appelle $\{\omega \in \Omega : Y(\omega) \in \mathcal{Y}\}$, que l'on note $Y^{-1}(\mathcal{Y})$, l'image réciproque par Y de l'ensemble $\mathcal{Y}$ avec $\mathcal{Y} \in \mathcal{P}(\{0,1,2\})$. Notons que $\mathcal{P}(\{0,1,2\})$ est l'ensemble des parties de $\{0,1,2\}$, dont chaque élément est inclu dans $\{0,1,2\}$. *Idem* pour l'ensemble de départ, on considère généralement l'ensemble des parties de Ω, $\mathcal{B}$, dont chaque élément $b \subset \{voiture, bus, vélo\}$. La raison est que les événements qui nous intéressent sont plus riches que juste les événements élémentaires. Par exemple, on veut savoir quelle est la probabilité de ne jamais prendre le vélo ! (voir Appel, W., 2013, Probabilités pour les non-probabilistes, H&K.) On simplifie souvent en écrivant $\{Y \in \mathcal{Y}\}$ l'événement qui nous intéresse. Enfin, on définit la (mesure de) probablité à valeurs dans IR, $P(\{Y \in \mathcal{Y}\})$, ou plus simplement $P_Y(\mathcal{Y})$ ou $P(\mathcal{Y})$, et $P(y)$ pour un événement particulier, avec ici $y \in \mathcal{Y}$. Alors, $P(\mathcal{Y}) = \sum(y \in \mathcal{Y})p(y)$. Cette mesure doit satisfaire la propriété de σ −additivité et être telle que $P(\Omega) = \sum(\omega \in \Omega)p(\omega) = 1$.

Dans cet exemple, il suffit de définir deux nombres, $p(0) = 1/2$, $p(1) = 1/3$, le troisième, $p(2)$, se déduisant des précédents par $p(2) = 1 - p(0) - p(1) = 1/6$. On pense souvent à tort qu'il s'agit des probabilités des valeurs de la variable (par exemple, $p(0) = 1/2$, alors qu'en fait $p(0) = P(\{\omega \in \Omega : Y(\omega) = 0\})$; c'est $P(\{voiture\})$ qui vaut 1/2).

2.1 Simulation de lois usuelles (variables aléatoires discrètes et continues)

La méthode Monte Carlo est donc une méthode de simulation permétant de générer une ou plusieurs valeurs d'une variable aléatoire appartenant à une famille bien précise de lois (la famille des lois normales, par exemple). Nous allons voir une version simple de la méthode dans cette section, la méthode inverse. Cette méthode intuitive n'est utilisée par Stata que pour des lois simples.

Supposons une variable aléatoire Y suivant une loi de Bernoulli à valeur dans $\{0,1\}$, avec $P(Y = 1) \equiv p$ et $P(Y = 0) \equiv 1 - P(Y = 1) = 1 - p$. La loi (fonction de masse) ne s'écrit pas $P(Y = y) = yp + (1 - y)(1 - p),\ y \in \{0,1\}$, mais comme la loi d'une Binomiale de paramètres $N = 1$ et p (voir **2.1.2**). Ainsi :

$$P(Y = y) = p^y(1 - p)^{1-y},\ y \in \{0,1\}.$$

Avant de simuler un échantillon de valeurs, tapez la commande

```
display rbinomial(1,.5)
```

Cette commande renvoie la valeur d'une variable aléatoire appartenant à la famille Bernoulli(p). La loi retenue est telle que $p = 1/2$. On obtient soit un 1, soit un 0, chacun avec probabilité 1/2.

Comment Stata choisit-il entre ces deux valeurs ?

- La commande `display rbinomial(1,.5)` utilise un algorithme déterministe de calcul d'un nombre pseudo-aléatoire α, comme si ce nombre était la réalisation d'une variable aléatoire uniforme (continue) $U[0; 1[$.
- La commande vérifie si ce nombre est supérieur à p, le paramètre d'une Bernouilli (dans notre exemple, $p = 1/2$). Si c'est le cas, la commande renvoie la valeur 1, sinon 0.

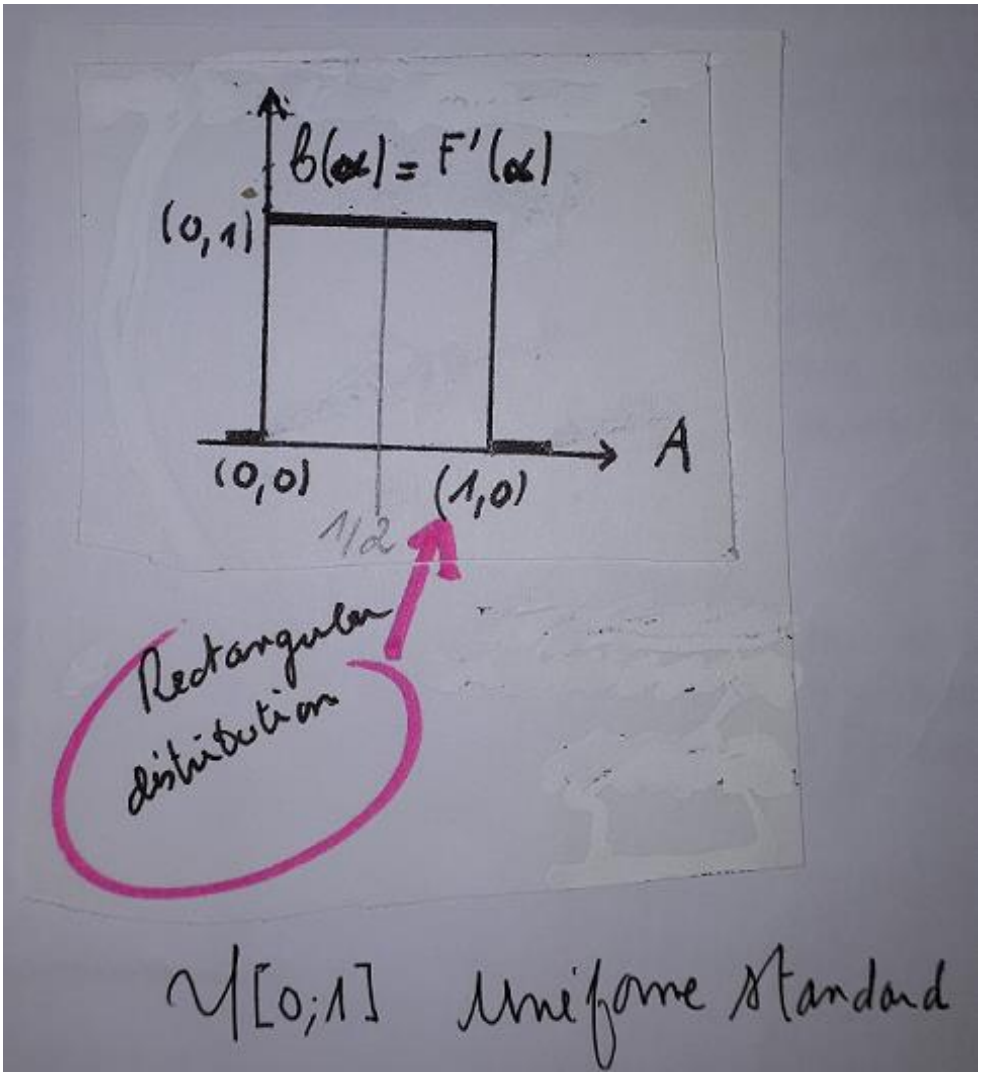
Le nombre α est calculé en arrière plan par Stata avec une commande elle-même disponible sur Stata : `runiform`. Tapez

```
display runiform(),
```

et vous obtiendrez une valeur pour α (notons que 1 est exclu).

En fait, toutes les commandes de production d'un fractile tiré d'une loi non-uniforme (Bernoulli, normale, etc.) s'appuient d'abord sur `runiform()` qui produit le nombre pseudo-aléatoire. Vous pouvez également aller voir la documentation Stata, [**D**], pp. 259–261 sur l'algorithme qui se cache derrière `runiform()`. Dans Stata 13, cet algorithme s'appelle KISS, acronyme de *keep it simple stupid* !

Graphique y. Loi uniforme standard $U[0; 1]$.



2.1.1 rbinomial(1,p) (valeurs d'une Bernoulli de paramètre p)

Pour remplir une colonne de la feuille de données avec des réalisations d'une variable suivant une Bernoulli(p), une seule commande suffit.

```
. cls

. clear all


. set obs                 10000
Number of observations (_N) was 0, now 10,000.

. generate                Y=rbinomial(1,.5)       // rbinomial(1,p) : Bernoulli
```

La fréquence des 0 et des 1 doit être celle dans la loi théorique sous-jacente ; nous avons supposé `rbinomial(1,.5)`, donc le paramètre p vaut 1/2. En théorie, il y a autant de "0" que de "1", ce qui est facile à vérifier avec `summarize Y`.

La fonction `rbinomial(k,p)` que nous venons d'introduire ($k \geq 0$), correspond à la famille des lois binomiales (nous reviendrons dessus). Stata a une commande pour chaque famille de v.a. : la famille des lois uniformes (`runiform`), des lois normales (`rnormal`), etc. Ces commandes sont appelées des *random functions*.

Puisque Stata génère les fractiles d'une Bernoulli de paramètre 1/2 à partir de nombres α pseudo-aléatoires, chaque PC de cette salle renvoie un résultat indépendant. Il devrait y avoir 23 étudiant-es dans la classe, plus le Prof. La probabilité que nous ayons tous un 1 après avoir tapé `rbinomial(1,.5)` est

$$Pr(Y_1 = 1, \dots, Y_{24} = 1) = [Pr(Y = 1)]^{24} = (1/2)^{24} = 0{,}00000005960464 \cong 5{,}96 \times 10^{-8}.$$

C'est la théorie. Comme pour tout logiciel de statistique, l'algorithme qui calcule la valeur de Y (le nombre pseudo-aléatoire) dans votre PC est déterministe. Pour s'en convaincre, ouvrons toutes et tous une nouvelle instance de Stata et tapons `di runiform()`. J'obtiens .3488717 avec Stata 17, .13698408 avec Stata 13. Et vous ?

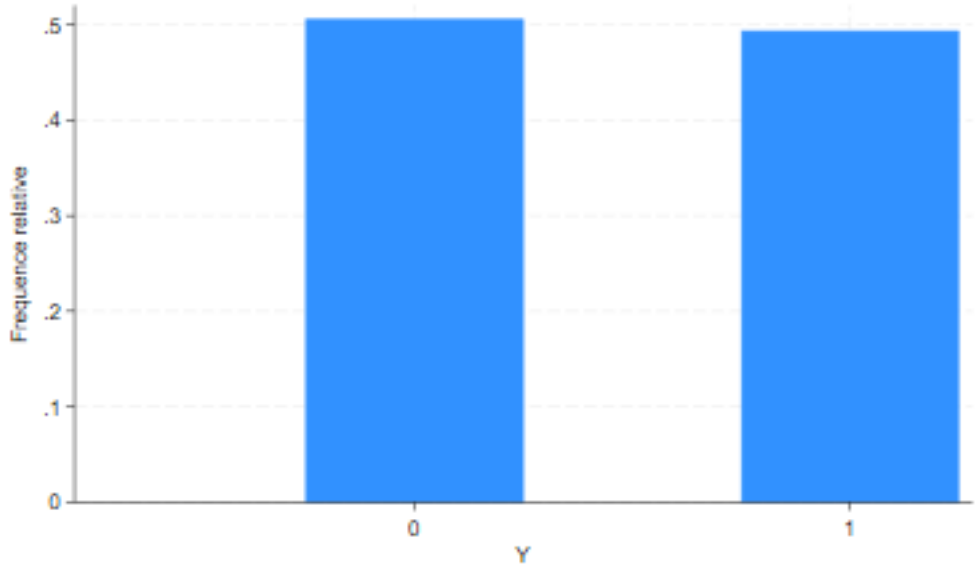
Si nous n'avons pas le même résultat à ce stade c'est simplement que nos algorithmes diffèrent un peu (l'algorithme produit une séquence de nombres aboutissant à α) parce que nous n'utilisons pas la même version de Stata (l'algorithme **KISS** a peut-être changé). α dépend notamment d'un paramètre clé de l'algorithme, la "graine" (*seed*), qui peut-être diffère entre nos Stata. Nous pouvons modifier cette graine avec la commande `set seed`. Par exemple,

```
. set     seed    21041971                        // Doit etre un nombre entre 0 et 2^31-1

. replace         Y=rbinomial(1,.5)       // rbinomial(1,p) : Bernoulli
(5,066 real changes made)
```

Vous devez obtenir 10000 valeurs 0/1 issues d'une Bernoulli de paramètre 1/2. Si avec la même graine (21041971) nous n'obtenons pas une suite de nombres identique, c'est pour l'une des raisons susmentionnées. Avec les nouvelles versions de Stata, l'algorithme de production de nombres pseudo-aléatoires est de plus en plus performant (la recherche avance sur ce sujet), au sens où il produit des nombres de moins en moins prédictibles. Traçons la distribution empirique de Y, avec `histogram()` ou simplement `hist()`, avec quelques options.

```
. hist                    Y, discrete xlabel(0(1)1) gap(50) ytitle("Frequence relative")
(start=0, width=1)


. graph export    "statainitiation_2_binomiale1.png", width(380) replace
file statainitiation_2_binomiale1.png saved as PNG format
```

Graphique z. $B(1,1/2)$.



Note. Distribution de Bernoulli de paramètre 1/2, $N = 10000$ *observations*

Utilisons `summarize Y` afin de vérifier que la variance empirique est bien égale au moment théorique $V(Y) = p(1-p) = .5(1-.5) = .25$. On élève l'écart-type au carré (c'est la variance corrigée) ; `di r(sd)*r(sd)` donne la valeur attendue, 0.25. La correction n'est cependant pas cruciale ici, le facteur de correction $N/(N-1)$ étant proche de 1 (N est grand, 10000).

2.1.2 `rbinomial(N,p)` (valeurs d'une binomiale de paramètres N et p)

On peut considérer la loi de Bernoulli comme un cas particulier dans la famille binomiale : `rbinomial(1,p)`. En matière de simulation, Stata modélise une binomiale comme une généralisation d'une Bernoulli, le nombre de fois que l'événement associé à la valeur 1 serait obtenu si l'on effectuait N tirages d'une Bernoulli. Ainsi, $\sum^i Y_i$, où Y_i est une Bernoulli, est une v.a. distribuée binomiale, à valeurs dans $0,1,\dots,N$. En notant $C_N^y \equiv \frac{N!}{y!(N-y)!}$, où $x! = x(x-1)(x-2)\cdots 1$, alors

$$P(\sum Y_i = y) = C_N^y p^y (1-p)^{N-y}, y \in \{0,1,\dots,N\},$$

$$E(\sum Y_i) = Np,\ V(\sum^i Y_i) = Np(1-p).$$

```
. set     seed     21041971
```

```
. replace         Y=rbinomial(5,.5)       // rbinomial(n,p)
(9,670 real changes made)
```

[Rentrez la commande `su Y, d` afin de vérifier que la médiane est bien égale à la partie entière de Np par défaut.]

Stata offre aussi la possibilité de calculer la probabilité que la v.a. suivant une loi donnée prenne telle ou telle valeur. Dans le cas de la variable Y ci-dessus, $Y \sim B(5,.5)$, la probabilité par exemple que Y soit égale à 3 est obtenue avec la commande **`binomialp(5,3,.5)`**

```
. di                    binomialp(5,3,.5)
.3125
```

[Vérification du résultat à la main]

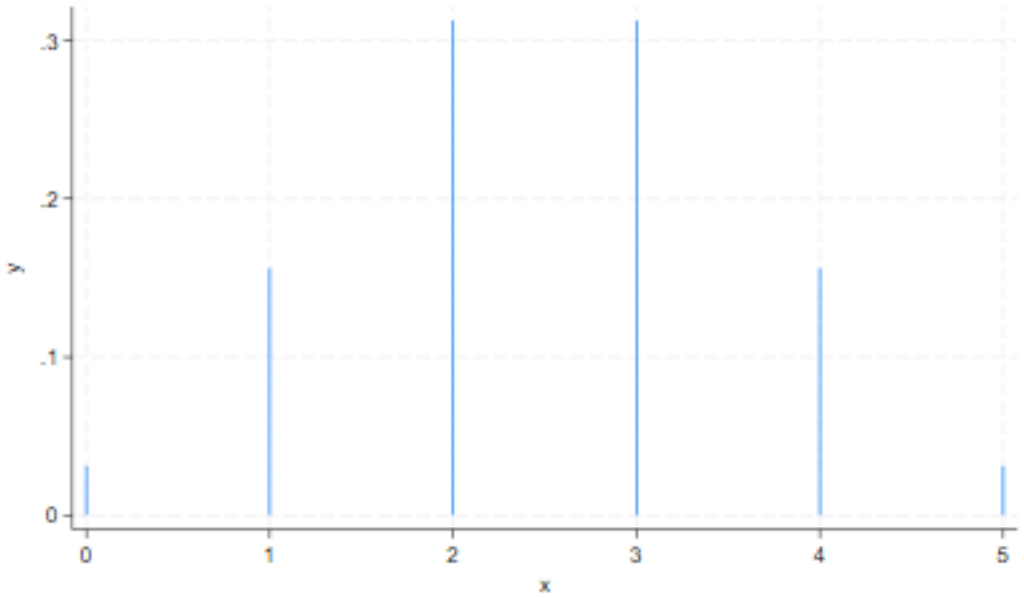
Il existe des variantes de ces fonctions. Par exemple, la probabilité que Y soit inférieure ou égale à 3 : **`binomial(5,3,.5)`**

```
. di                    binomial(5,3,.5)
.8125
```

Nous pouvons tracer les distributions de probabilité, sans avoir à simuler de valeur au préalable. La fonction de masse pour une variable discrète ou la densité pour une v.a. continue s'obtiennent à partir des *Probability distribution and density functions* que l'on combine aux commandes de l'environnement graphique **`graph twoway`**. Par exemple, pour tracer la fonction de masse d'une binomiale de paramètres $N \equiv 5$ et $p \equiv .5$, il faut :

```
. graph twoway function   y=binomialp(5,x,.5), range(0 5) ///
                                                droplines(0 1 2 3 4 5) connect(none)



. graph export                    statainitiation_2_binomiale.png, width(400) replace
file statainitiation_2_binomiale.png saved as PNG format
```

Graphique u. $B(5,1/2)$ avec l'outil de tracage de fonctions Stata.



Note. Distribution Binomiale de paramètre 5 et 1/2, avec `graph twoway function`'

2.1.3 `rpoisson(lambda)` (v.a. de Poisson de paramètre λ)

Dans l'exemple qui suit, on va construire une variable à partir de 400 observations de brevets cités par d'autres brevets sur le territoire français. Les identifiants des brevets cités sont dans la troisième colonne, ceux des brevets qui citent dans la dernière colonne. Chaque brevet cité est caractérisé par au moins une classe technologique (la quatrième colonne). Vous verrez les codes B01J, C01B, etc. Il existe plusieurs dizaines de classes technologiques, 98 classes différentes dans cet exemple. La majorité des technologiques apparaissent plus d'une fois (la classe B29D par exemple), et cela pour plusieurs raisons :

1. Les brevets cités ont des technologies communes.
2. Un brevet peut être cité par plusieurs brevets, donc l'identifiant et les classes technologiques du brevet cité apparaissent plusieurs fois.
3. Un brevet peut citer plusieurs brevets ayant des technologies proches, ce qui renforce le cas "1.".

On peut virer les identifiants des brevets et considérer la variable X_i du nombre de fois qu'une classe technologique i est citée. La variable

$$Y_x \equiv Card\{i: X_i = x\}$$

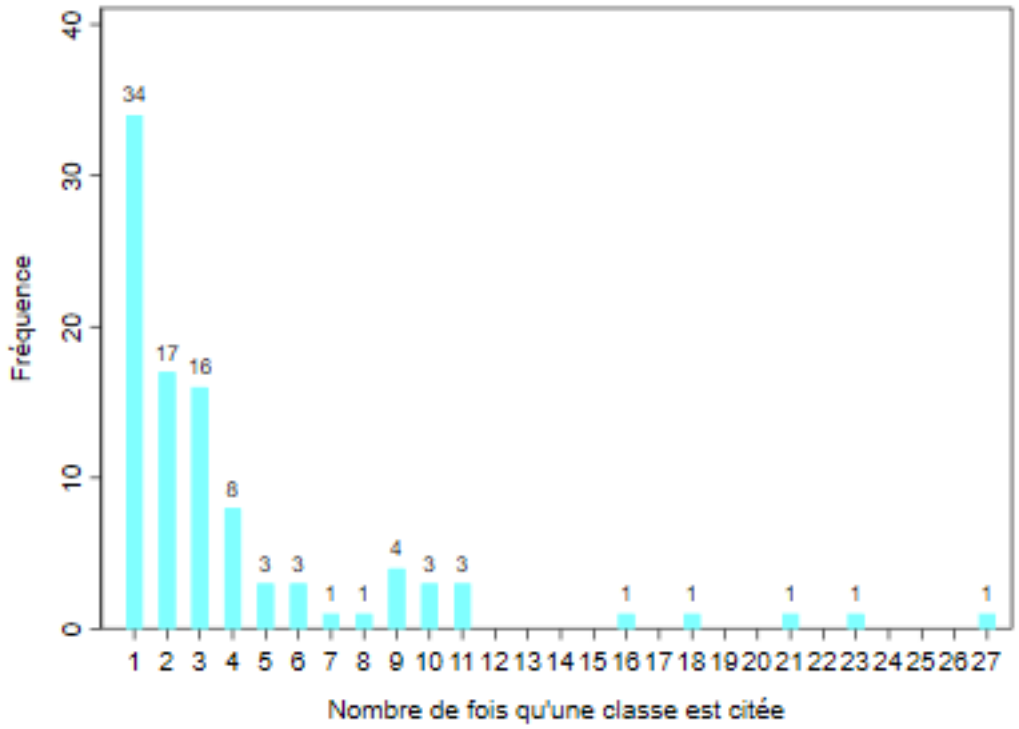
est le nombre de classes citées x fois. Nous allons faire un graphique avec x en abscisse et Y_x en ordonnée sous la forme d'un diagramme en bâton. On utilise un histogramme pour une variable discrète.

```
. local          A="http://www.evens-salies.com/"
. local          B="`A'"+"statainitiation_2_poisson_NUTS3_IPCclasses_stata13.dta"
```

```
. use                    `B', clear

. des

Contains data from http://www.evens-
salies.com/statainitiation_2_poisson_NUTS3_IPCclasses_stata13.dta
 Observations:           400
    Variables:             5                  17 Sep 2018 16:45
-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------
---------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------
---------------------------------
nuts3           str7    %9s
earliest_publ~r int     %8.0g
pat_publn_id    long    %12.0g
IPC4            str4    %9s
citing_pat_pu~d long    %12.0g
-------------------------------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------------------------
---------------------------------
Sorted by: nuts3  earliest_publn_year  pat_publn_id  IPC4  citing_pat_publn_id

. list in         1/10

     +-----------------------------------------------+
     | nuts3   earlie~r   pat_pub~d   IPC4   citing_~d |
     |-----------------------------------------------|
  1. | FR101       2013   379463319   B01J   421523492 |
  2. | FR101       2013   379463319   C01B   421523492 |
  3. | FR101       2013   379463319   H01M   421523492 |
  4. | FR101       2013   379464074   B29D   422058016 |
  5. | FR101       2013   379464074   B29D   424112948 |
     |-----------------------------------------------|
  6. | FR101       2013   379464074   B29D   424871442 |
  7. | FR101       2013   379464074   B29D   425516473 |
  8. | FR101       2013   379464074   B29D   426259714 |
  9. | FR101       2013   379464074   B29D   437167756 |
 10. | FR101       2013   379464074   B64C   422058016 |
     +-----------------------------------------------+

. drop            nuts3 earl

. rename          (pat citing)(CITED CITING)


. sort            CITING CITED

. list in         1/10

     +-----------------------------+
     |     CITED   IPC4      CITING |
     |-----------------------------|
  1. | 410024596   G01J   275373132 |
  2. | 410024596   G02B   275373132 |
  3. | 410024596   B82Y   275373132 |
  4. | 410261990   C07K   277499783 |
  5. | 410261990   A61P   277499783 |
     |-----------------------------|
  6. | 410261990   A61K   277499783 |
  7. | 381488523   C12N   284680625 |
  8. | 381488523   C07K   284680625 |
  9. | 381488523   C12Q   284680625 |
 10. | 381488523   A61K   284680625 |
     +-----------------------------+

. sort            CITED IPC4 CITING

. sort            CITING CITED IPC4
```

```
. keep            IPC4

. sort            IPC4

. generate        Y=1

. collapse        (count) Y, by(IPC4)


. gsort           -Y

. label var       Y "Nombre de fois qu'une classe est citée"

. histogram       Y,      discrete freq width(.5) color(cyan*.5) addlabels ///
                                scheme(s1mono) ///
                                xlabel(#27) xscale(noline titlegap(3)) ///
                                ytitle("Fréquence") yscale(titlegap(3))

(start=1, width=.5)


file statainitiation_2_poisson3.png saved as PNG format
```

Graphique v. Distribution du nombre de citations, $P(\lambda)$.



Note. Pour chaque valeur de x en abscisse, une barre mesure le nombre de de classes technologiques citées x fois

On constate que 34 classes sont citées une fois, alors qu'une seule classe est citée 27 fois (il s'agit des appareils de mesure, comptage, hardware informatique, etc.). Il est clair qu'il nous faut abandonner l'idée d'une loi Binomiale. La distribution des fréquences est plus proche de celle d'une loi de Poisson, vers laquelle le statisticien se tourne souvent, en première analyse lorsqu'il a affaire à une v.a. de comptage (voir le Ch. 6 dans Efron et Hastie, 2017).

Supposons donc que $P(X_i = x) = \lambda_i^x e^{-\lambda_i}/x!$, $x = 1, \ldots, 27$. On sait que $E(X_i) = V(X_i) = \lambda_i$. Si on a affaire à une loi de Poisson, la moyenne empirique devrait être proche de la variance. Vérifions avec `su Y` puis `di r(sd)*r(sd)`. On peut voir qu'une classe est citée quatre fois en moyenne. En revanche, on obtient 23,74 pour la variance empirique (corrigée). Celle-ci est trop élevée pour pouvoir affirmer que le nombre de classes citées suit une loi de Poisson.

[Proposer une loi pour cette distribution empirique]

2.1.4 `rdiscrete(r,c,p)` (loi multinomiale en Mata)

On peut voir la loi multinomiale comme une extension de la loi binomiale à plus d'une Bernoulli. Alors qu'une Bernoulli porte sur un événement et son contraire lors d'un tirage ({suit une formation, n'en suit pas}, {est au ch^omage, n'y est pas}, etc.), et la Binomiale permet de calculer la probabilité de cet événement en n tirages, la loi multinomiale combine les deux en permettant de calculer la probabilité de succés de plusieurs événements lors de n tirages. Pour être plus précis, supposons m types d'éléments, dont on connaît l'occurrence dans la population : $p_1, p_2, \ldots, p_m$. On a bien sûr la somme $\sum^j p_j = 1$. Soit $N_j, j = 1, \ldots m$, le nombre de fois que l'on suppose observer l'élément j dans un n-échantillon. On s'intéresse à la probabilité d'avoir $N_1 = n_1, \ldots, N_m = n_m$, avec $\sum(j) N_j = n$. La probabilité correspondante est :

$$\frac{n!}{n_1! \cdots n_m!} p_1^{n_1} \cdots p_m^{n_m}.$$

Notons que pour répondre à cette question nous n'avons pas besoin de connaître la taille de la population, N.

On a $E(N_j) = np_j$ et $V(N_j) = np_j(1 - p_j)$, mais aussi $Co(vN_j, N_l) = -np_j p_l$.

2.2 Convergences, Théorème central limite

Il est courant de regarder ce qui se passe lorsque un ou plusieurs paramètres d'une loi tendent vers l'infini. Par exemple, si on fait tendre le paramètre n d'une loi binomiale vers l'infini, et p dépend de n de la manière suivante : $\lim_{n\to\infty} np(n) = \lambda$ est une constante qui, elle, ne dépend pas de n, alors Binomial(n,$p(n)$)→Poisson(λ). On peut aussi étudier la convergence de la distribution de probabilité légèrement transformée. Pour l'étude de la convergence en loi (de probabilité) par exemple, la transformation consiste à soustraire le moment non-centré d'ordre un à la variable et rapporter cette différence à la racine carrée du moment centré d'ordre deux. Dans le cas de la loi Binomiale par exemple, on étudie le comportement du ratio suivant :

$$\frac{Y - Np}{\sqrt{Np(1-p)}}.$$

Le théorème central limite (TCL) porte sur la fonction de répartition. La loi, c'est-à-dire la distribution de probabilité de ce ratio tend vers celle d'une loi N(0,1), ou loi de Moivre-Laplace-Gauss, sous certaines conditions.

[Donner ces conditions]

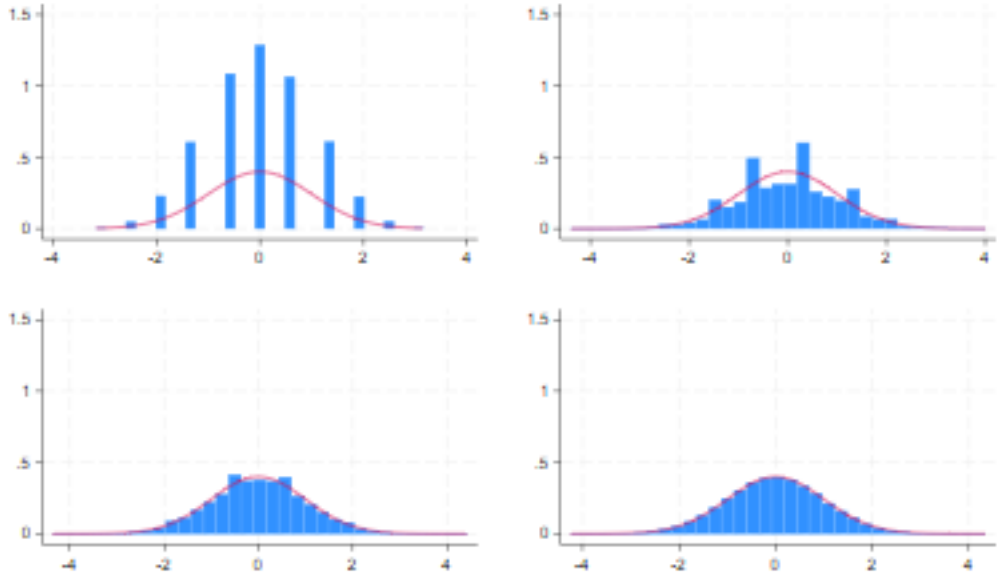
La démonstration de la convergence en loi est difficile, contrairement à la vérification (visuelle) avec l'ordinateur, comme nous allons le voir ci-dessous.

[Décrire l'expérience]

```
. clear all


. set seed               21041971

. set obs                100000
Number of observations (_N) was 0, now 100,000.

. generate               Y_10=rbinomial(10,.5)

. summarize              Y_10

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
        Y_10 |    100,000      4.9976    1.576921          0         10

. replace                Y_10=(Y_10-5)/sqrt(5*.5)
(100,000 real changes made)

. twoway hist     Y_10, bin(33) saving(GRAPH_10, replace) legend(off) ///
                                ylabel(0. .5 1. 1.5) || ///
                                function normalden(x,0,1), rang(Y_10)

file GRAPH_10.gph saved

. generate               Y_100=rbinomial(100,.5)

. summarize              Y_100

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
       Y_100 |    100,000    50.00444    4.979021         28         70

. replace                Y_100=(Y_100-50)/sqrt(50*.5)
(100,000 real changes made)
```

```
. twoway hist     Y_100, bin(33)saving(GRAPH_100, replace) legend(off) ///
                                ylabel(0. .5 1. 1.5) || ///
                                function normalden(x,0,1), rang(Y_100)

file GRAPH_100.gph saved

. generate                Y_1000=rbinomial(1000,.5)

. summarize               Y_1000

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
      Y_1000 |    100,000    499.9638    15.82442        431        570

. replace                 Y_1000=(Y_1000-500)/sqrt(500*.5)
(100,000 real changes made)

. twoway hist     Y_1000, bin(33)saving(GRAPH_1000, replace) legend(off) ///
                                ylabel(0. .5 1. 1.5) || ///
                                function normalden(x,0,1), rang(Y_1000)

file GRAPH_1000.gph saved

. generate                Y_10000=rbinomial(10000,.5)

. summarize               Y_10000

    Variable |        Obs        Mean    Std. dev.       Min        Max
-------------+---------------------------------------------------------
     Y_10000 |    100,000    4999.941    50.11944       4788       5217

. replace                 Y_10000=(Y_10000-5000)/sqrt(5000*.5)
(100,000 real changes made)

. twoway hist     Y_10000, bin(33) saving(GRAPH_10000, replace) legend(off) ///
                                ylabel(0. .5 1. 1.5) || ///
                                function normalden(x,0,1), rang(Y_10000)

file GRAPH_10000.gph saved

. graph combine   GRAPH_10.gph GRAPH_100.gph GRAPH_1000.gph GRAPH_10000.gph


. graph export    statainitiation_2_tlc.png, width(400) replace
file statainitiation_2_tlc.png saved as PNG format
```

Graphique w. Théorème central limite pour la loi Binomiale - Simulation avec Stata.



Il existe des démonstrations du TCL. Cette notion de convergence est aussi appelée convergence faible, au sens de la convergence des fonctions (de répartitions), qui est plus un concept de matheux ! Le nom vient du fait que deux variables ayant des distributions de probabilité et donc des fonctions de répartitions identiques à l'arrivée, peuvent en fait être très différentes au départ.

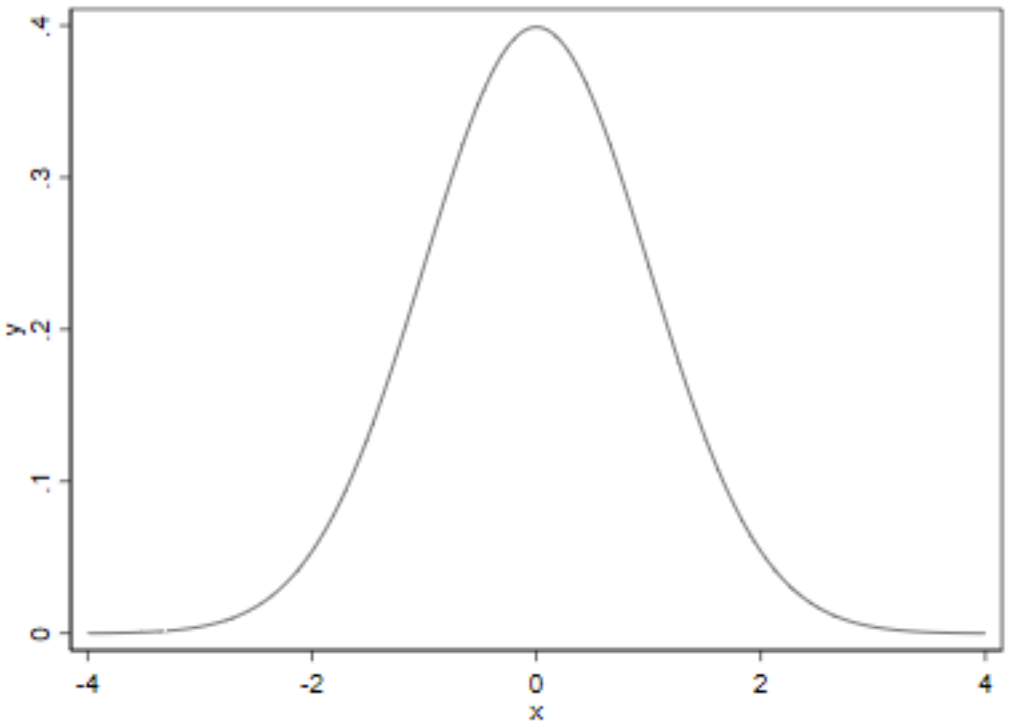
[Quelles notions de convergence connaissez-vous et les relations entre elles ?]

2.3 Simulations de variables aléatoires continues (à densité)

2.3.1 `rnormal()` (loi normale centrée réduite)

```
. graph twoway function   y=normalden(x), range(-4 4) scheme(s1mono)


. graph export                    statainitiation_2_normale01.png, width(400) replace
file statainitiation_2_normale01.png saved as PNG format
```

Graphique n. Densité N(0,1), tracée avec une fonction Stata.



[Exercice d'illustration de la loi normale centrée réduite, Φ]

```
. clear

. set seed        21041971

. set obs         100000
Number of observations (_N) was 0, now 100,000.

. generate        Y=rnormal(0.17,0.006)

. sort            Y
```

Quelle est la probabilité que Y soit compris entre .16 et .18 ?

```
. count if        Y>0.16&Y<=0.18
  90,425

. display         r(N)/100000
.90425

. display         normal((.18-.17)/.006)-normal((.16-.17)/.006)
```

```
.9044193

. display         2*normal((.18-.17)/.006)-1
.9044193
```

2.3.2 Fonctions de variables aléatoires continues

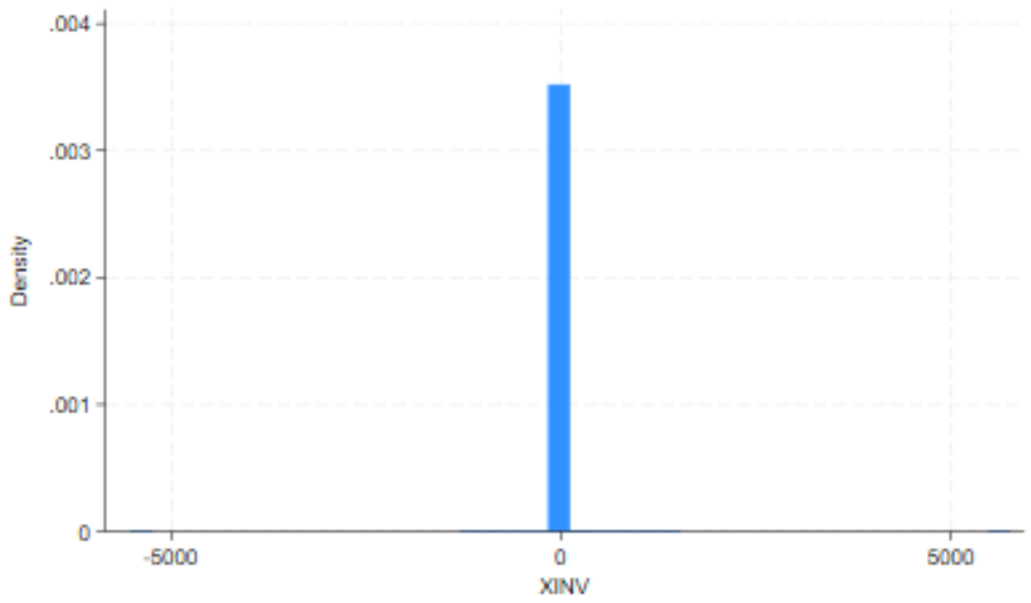
Vous savez probablement que le ratio de deux variables aléatoires normales centrées réduites suit une loi de Cauchy. C'est aussi une loi de Student à 1 degré de liberté. Or, les moments de cette variable ne sont pas définis (ils tendent vers l'infini). Bien évidemment, c'est à cause du dénominateur. On pourrait donc d'abord s'intéresser au cas univarié en se posant la question de la loi suivie par l'inverse d'une $N(0,1)$.

Pour x compris entre $0-\Delta x$ et $0+\Delta x$, lorsque $\Delta x \to 0$. Vous avez sûrement étudié ce problème de manière analytique en licence ; Tassi (2004, pp. 45-46) rappelle l'étude de l'espérance et de la variance d'un quotient. Soit $f(X,Y)$ une fonction de deux v.a. X et Y (il s'agit ici du quotien Y/X).

Si Y prend la valeur 1 avec certitude, nous sommes ramenés à étudier l'inverse d'une normale. On ne peut malheureusement pas réaliser un développement en série au point $(1, E(X))$ car, ce développpment dépend de $\frac{\partial}{\partial Y}\frac{1}{X} = -\frac{1}{X^2}$, qui n'existe pas au point $E(X) = 0$ (Tassi, 2004, p. 46). En revanche, avec un peu de programmation, on peut s'approcher de la solution.

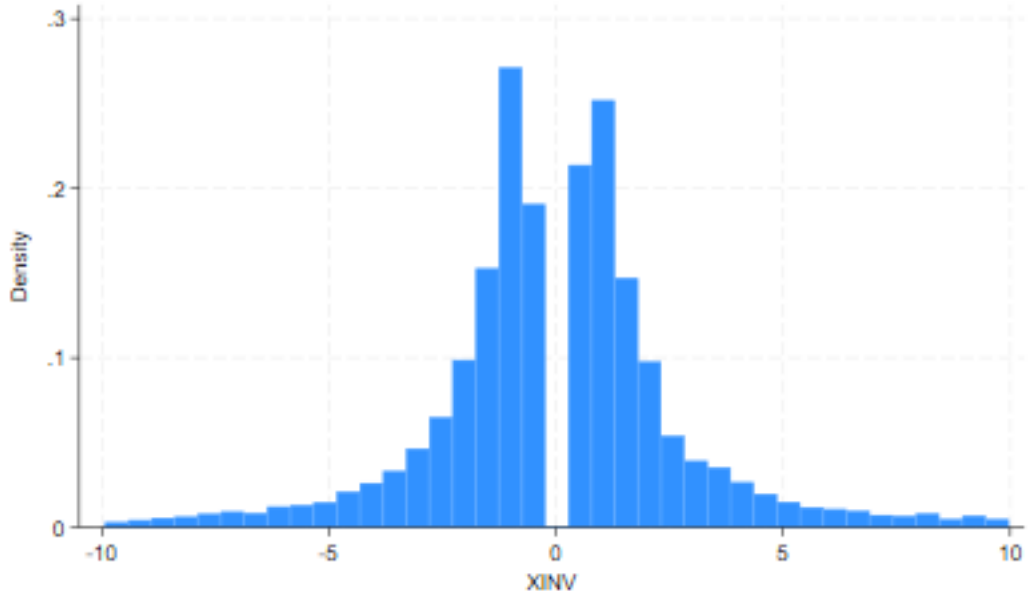
Regardons un peu comment se comporte la variable $1/X$.

```
. drop                    Y

. set     obs                     10000
Number of observations (_N) was 0, now 10,000.

. generate                X=rnormal()

. generate                XINV=1/X

. hist                    XINV
(bin=40, start=-5528.6411, width=282.18218)


. graph export    statainitiation_2_rnormalinv1.png, width(400) replace
file statainitiation_2_rnormalinv1.png saved as PNG format
```



Cet histogramme représente mal la densité car le support des valeurs de X est trop étendu, et on pourrait penser (à tort) que 0 est une valeur sur le support. En effet, X étant centrée autour de zéro, $1/X$ tend vers l'infini (plus et moins). Or, si les valeurs de x qui nous intéressent sont grosso modo celles entre -2 et 2, comme pour une loi N(0,1), il est naturel de zoomer le graphique sur ce domaine du support : $-10 \leq X \leq 10$ par exemple.

```
. hist                    XINV if XINV>=-10&XINV<=10
(bin=39, start=-9.9331522, width=.51019862)


. graph export    statainitiation_2_rnormalinv2.png, width(400) replace
file statainitiation_2_rnormalinv2.png saved as PNG format
```



[Faire l'étude du ratio de deux $N(0;1)$ à partir de l'algorithme précédent] Conseil : serrer un peu plus, et étudier le graphique entre -2 et $+2$.

La résolution analytique du problème à l'étude du ratio de deux $N(0,1)$ est bien plus difficile. La fonction de répartition cumulée du ratio de deux $N(0;1)$, X et Y est

$$Pr(Y/X \leq t) = \frac{1}{\pi}\left(\frac{\pi}{2} + \arctan(t)\right).$$

2.3.3 Skewness et Kurtosis pour une $N(0,1)$

Ceux-ci sont disponibles avec la commande `summarize Y, d`. On peut simuler une $N(0;1)$ et interpréter les valeurs de ces indicateurs (vu plus tôt dans le cours). `summarize` renvoie les Skweness et Kurtosis à la Pearson, pas à la Fisher.

```
. clear all


. set obs 100000
Number of observations (_N) was 0, now 100,000.

. gen Y=rnormal()

. su Y,d

                            Y
-------------------------------------------------------------
      Percentiles      Smallest
 1%    -2.335788      -4.276288
 5%    -1.648681      -4.028519
10%    -1.284986      -3.960019       Obs             100,000
25%    -.6761177      -3.930151       Sum of wgt.     100,000

50%     .0010354                      Mean           -.0003368
                       Largest       Std. dev.       1.001698
75%     .6778885       4.111713
90%     1.282818       4.134066       Variance        1.003398
95%     1.647589       4.341847       Skewness       -.0057677
99%     2.315641       4.592964       Kurtosis         2.98521
```

2.4 La commande `simulate`

On ne peut pas parler de simulation sans réplication. Lorsque l'on tape la commande **`generate Y=rnormal()'** pour 10000 observations, on a en fait 10000 réplications du tirage d'une $N(0;1)$. Suivant cette même logique, quand j'étudier le ratio de deux normales, j'obtiens 10000 ratios, dont il me suffit de tracer la distribution pour voir comment se comporte une Cauchy. La moyenne et la variance seront celles d'une Cauchy, asymptotiquement.

Mais supposons que la taille de la population ne soit pas élevée. Comment se comportent ces moments ? Dans ce cas, je ne peux pas me contenter d'une seule distribution, mais de plusieurs milliers, et regarder par exemple comment se comporte le moment d'ordre 1.

```
      . program                 cauchy, eclass
      .  drop _all
      .  set obs                10000
      .  gen                    X=rnormal()
      .  gen                    Y=rnormal()
      .  gen                    Z=Y/X
      .  su                             Z
      . end

      . simulate                MEANV=r(mean) VARV=r(Var), seed(21041971) reps(1000): cauchy

            Command: cauchy
              MEANV: r(mean)
               VARV: r(Var)

     Simulations (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


      . hist                    MEANV, xlabel(-50(25)50)
      (bin=29, start=-136.56459, width=12.85245)
```

Faisons la même chose avec une régression (voir la commande `simulate` de la documentation Stata)

```
      . program                 MYREG, eclass
      .  drop _all
      .  set obs                25
      .  gen                    X=rnormal()
      .  gen                    Y=3*X+1+rnormal()
      .  reg                    Y X
      . end

      . simulate                _b _se, seed(21041971) reps(1000): MYREG

            Command: MYREG

     Simulations (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


      . hist                    _b_X
      (bin=29, start=2.347707, width=.04692024)
```

Bibliographie

Efron, B., Hastie, T., 2017. Computer age statistical inference – Algorithms, evidence, and data sciences, IMS Monographs, Cambridge University Press.

Mooney, 1997. Monte Carlo Simulation. Sage Publications.

Leemis, L.M., McQueston, J.T., 2008. Univariate distribution relationships. The American Statistician, Vol. 62, pp. 45-53.

Tassi, 2004. Méthodes Statistiques. Economica, 3e édition.

```
. quietly log close
```