Plan du travail

I. Définition de la simulation

II. Présentation du logiciel R

III. Applications sur R

La Simulation

Méthode de mesure et d'étude consistant à remplacer un phénomène, un système par un modèle plus simple mais ayant un comportement analogue.

L'objectif d'un modèle de Simulation

Modèle descriptif :

étudier le comportement d'un système sous différentes hypothèses d'évolution de l'environnement.

Modèle normatif (décisionnel):

en simulant plusieurs décisions envisagées choisir la meilleure ou la moins mauvaise.

Simulation d'une loi de probabilité

Pour pouvoir simuler le comportement d'un système faisant intervenir des événements probabilisés, il va falloir simuler l'apparition de ces événements; c'est-à-dire générer des événements dont la fréquence observée sur un grand nombre de simulations doit être proche de la loi de probabilité théorique.

- On ne fait pas une expérience réelle (cela prendrait beaucoup de temps), mais on utilise l'ordinateur pour effectuer les tirages (constitution d'échantillons aléatoires) et comptabiliser les cas favorables et les cas possibles. Il s'agit donc d'une expérience "virtuelle".
- On se rend vite compte que si on veut un résultat significatif, il faut un nombre de tirages assez élevé. C'est d'ailleurs une bonne raison d'utiliser un ordinateur.

- Normalement, on s'attend à ce que le résultat converge vers la "bonne valeur". On peut alors se fixer un seuil de précision si l'on admet que le résultat converge ; par exemple on peut arrêter le tirage lorsque la différence entre deux valeurs successives est inférieure à 0,001.

Présentation de R

R est un logiciel de traitement statistique des données.

- Il fonctionne sous la forme d'un interpréteur de commandes. Il dispose d'une bibliothèque très large de fonctions statistiques,
- il es possible d'ajouter des modules externes compilés que l'on peut télécharger gratuitement sur internet..
- Il est possible d'utiliser R en mode interactif sans jamais avoir à programmer.

Comment utiliser les lois de probabilités avec R

La plupart des lois de probabilité sont présentes dans le package de base de R. il est possible d'utiliser quatre fonctions sur ces lois, référencées par les lettres **'d', 'p', 'q', et 'r'.**

Les fonctions

Y	Signification
binom	loi binomiale
multinom	loi multinomiale
nbinom	loi binomiale négative
geom	loi géométrique
hyper	loi hypergéométrique
pois	loi de Poisson
norm	loi normale
exp	loi exponentielle
gamma	loi gamma
chisq	loi de χ^2
f	loi de Fisher-Snedecor
t	loi de Student
wilcox	distribution de Mann - Whitney
signrank	distribution du test des signes de Wilcoxon

Les fonctions

- ***dY ()*** : donne la probabilité f(k) de la variable X discrète ou la densité de probabilité f(xi) de la variable x continue. Elle signifie densité, même si ce terme prête à confusion pour les variables discrètes.

Les fonctions

- ***pY ()*** **:** donne la fonction de répartition F(k) de la variable X discrète à droite ou à gauche de la valeur k et l'aire sous la courbe à droite ou à gauche de (xi).

Les fonctions

Avec R, la fonction donne par défaut la répartition à gauche, ce qui correspond à $P(x \leq k)$ ou $P(x \leq xi)$. Pour connaitre la répartition à droite, soit $P(x \geq k)$ ou $P(x \geq xi)$, utiliser lower.tail = FALSE dans les arguments.

La fonction de répartition est appelée distribution fonction dans R.

Les fonctions

- ***qY ()*** : pour une loi discrète : donne la valeur k de la variable X correspondant à une valeur de fonction de répartition F(k), c'est-à-dire le seuil sur l'axe des abscisses pour une somme de probabilités. Pour une loi continue : donne la valeur xi de la variable x correspondant à une valeur de fonction de répartition F(xi), c'est-à-dire le seuil sur l'axe des abscisses pour une aire sous la courbe.

Les fonctions

Avec R, la fonction considère par défaut qu'il s'agit de la répartition à gauche, ce qui correspond à P(X ≤ k) ou P(x ≤ xi). Pour signaler qu'il s'agit de la répartition à droite, soit P(X ≥ k) ou P(x ≥ xi), utiliser l'argument lower.tail=FALSE.

Les valeurs données sont appelées quantiles dans R

Les fonctions

Ry () : donne une série de valeurs aléatoires de la variables X discrète, ou x continue, mais dont les probabilités d'apparition sont prises en compte. Ainsi, l-histogramme de la série obtenue redonne la distribution de la loi utilisée. Plus il y a de valeurs, plus l'histogramme sera proche de la distribution de la loi.

Remarque : une valeur de la variable X discrète ou x continue appelée quantile dans R.

Les fonctions

Pour effectuer un calcul avec une de ces lois, il suffit d'utiliser comme fonction l'une des appellations R ci-dessus avec le préfixe *'d'* pour une probabilité, *'p'* pour une fonction de répartition, *'q'* pour une fonction quantile et *'r'* pour un tirage aléatoire.

Loi Binomiale

- C'est une distribution les plus fréquemment rencontrées en statistique.
- Ecriture : B(n,p)
- **n :** nombre de tirages avec remise ;
- **p :** probabilité de l'événement « 1 noire » lors d'un tirage.

Exemples avec R sur la probabilité

Calculons la probabilité d'avoir 2 noires quand on tire 4 boules avec remise, sachant que la probabilité de tirer une noire est de 0,2. Avec la formule de la loi binomiale : B(4 ; 0,2)

```
>  dbinom(k, n, p)
>  dbinom(2, 4, 0.2)
[1] 0.1536
```

La probabilité P(X=2) est donc de 15,36%.

Valeur de la fonction de répartition F(k)

```
> pbinom(k, n, p, lower,tail=TRUE)
> pbinom(2,4,0.2, lower.tail=TRUE)
[1] 0.9728

> pbinom(2,4,0.2, lower.tail=FALSE)
[1] 0.0272
```

Valeurs de la fonction de répartition F(k)

```
> pbinom(k, n, p)
> pbinom(0:10,4,0.2)

[1] 0.4096 0.8192 0.9728 0.9984 1.0000 1.0000
    1.0000 1.0000 1.0000 1.0000 1.0000
```

Série des valeurs

```
> rbinom(k, n, p)
> binom(2,4,0.2)
                    [1] 1 1
> rbinom(10,4,0.2)
                  [1] 1 0 2 1 0 1 1 1 1 1
> rbinom(50,4,0.2)
 [1] 1 0 3 0 0 2 2 0 0 1 0 1 2 0 0 2 1 0 1 0 0 0 1 1 1
   1 1 2 0 1 0 2 1 0 1 2 2 2 1 2 2 1 1 0 1 1 0 2 0 2
```

Exemples avec R sur la fonction de répartition

Calculons la probabilité d'avoir au moins 2 noires quand on tire 4 boules avec remise, sachant sue la probabilité de tirer une noir est de 0,2. il s'agit de déterminer une fonction de répartition puisque de problème est de trouver la probabilité P(X≥2). Il est possible de raisonner en disant qu'il suffit de faire la somme P(X=2) + P(X=3)+P(X=4):

Exemples avec R sur la fonction de répartition

```
> p=0,2
> n=4

> dbinom(2, n, p)+ dbinom(3,n,p)+dbinom(4,n,p)
> dbinom(2,4,0.2)+dbinom(3,4,0.2)+dbinom(4,4,0.2)

[1] 0.1808
```

Exemples avec R sur la fonction de répartition

On peut aussi exclure la probabilité P(X=0) d'avoir 0 boule noire et la probabilité P(X=1) d'avoir 1 boule noire, la totalité de la fonction de répartition étant égale à 1:

```
>  p=0,2
>  n=4
> 1-dbinom(0,n,p)-dbinom(1,n,p)
> 1-dbinom(0,4,0.2)-dbinom(1,4,0.2)
[1] 0.1808
```

Exemples avec R sur la fonction de répartition

On peut aussi exclure la probabilité P(X≥2) directement, ce qui est l'équivalent de F(1)droite= P(X>1) puisque à droite de la distribution, on travaille avec des valeurs strictement supérieures:

Exemples avec R sur la fonction de répartition

> pbinom(1,n,p lower,tail=FALSE) # lower,tail=FALSE est l'argument de travail à droite de la distribution

> pbinom(1,4,0.2, lower.tail=FALSE)

[1] 0.1808

Pour vérifier:

> pbinom(1,4,0.2, lower.tail=TRUE)

[1] 0.8192

Exemples avec R sur la fonction de répartition

Quand on tire 4 boules avec remise, à combien de boules noires minimum correspond la probabilité 0,1808?

Cette probabilité est une fonction de répartition à droite de la loi binomiale B(4,0.2):

> p=0.2

> n=4

> qbinom(0.1808, n, p, lower.tail=FALSE)

> qbinom(0.1808,4,0.2, lower.tail=FALSE)

[1] 1

On trouve la valeur k = 1, ce qui correspond bien à P(X> 1) soit 2 boules noires minimum

Tracés de densités et de fonctions de répartitions

Des tracés de densités de probabilité ou de fonctions de répartition de lois diverses peuvent s'obtenir à l'aide de la fonction *plot().* Ainsi par exemple pour tracer la densité (répartition des masses) d'une loi binomiale avec n=10 et p=0,25, on exécute dans R :

Tracés de densités et de fonctions de répartitions

```
x <- 0:10
y <- dbinom(x, size=10, prob=.25)    # évalue les probas
{plot(x, y, type = "h", lwd = 30,
     main = "Densité Binomiale avec \n n
     = 10, p = .25", ylab = "p(x)", lend ="square")}
```

Tracés de densités et de fonctions de répartitions

Le type de tracé est spécifié avec l'option type=h (lignes verticales d'un diagramme en bâtons), épaissies grâce à l'option lwd=30. L'option lend ="square" permet de tracer des barres rectangulaires. Nous avons ensuite ajouté des légendes.

Histogramme

```
N <- 10000
n <- 20
p <- 0.5
x <- rep(0,N)
for (i in 1:N) {
  x[i] <- sum(runif(n)<p)
}
hist(x,
     col='light blue',
     main="Simulation d'une loi binomiale")
```

Histogramme

On peut aussi utiliser la commande rbinom:

```
N <- 1000
n <- 10
p <- .5
x <- rbinom(N,n,p)
hist(x,
    xlim=c(min(x),max(x)), probability=T, nclass=max(x)-min(x)+1,
    col='lightblue',
    main='Loi binomiale, n=10, p=.5')
lines(density(x,bw=1), col='red', lwd=3)
```

Lois Binomiale Négative

fonction de densité:

On une loi Binomiale Négative de paramètres :

(2.5 ; 0.4), alors P(X=3) et P(X ≥ 3) se déterminent avec les commandes :

```
> dnbinom(k, size= , prob=p, log=),
> dnbinom(3,2.5,0.4)
                    [1] 0.1434409
```

Lois Binomiale Négative

```
>  pnbinom(k, size= , prob=p, lower.tail=, log.p=)
>  pnbinom(3,2.5,0.4,lower.tail=TRUE)
                [1] 0.5558019

> pnbinom(3,2.5,0.4,lower.tail=FALSE)
                [1] 0.4441981
```

Lois Binomiale Négative

```
>  pnbinom(k, size= , prob=p)
>  pnbinom(3:15,2.5,0.4)

 [1] 0.5558019 0.6741407 0.7664449 0.835673
 0.8861107 0.9220476 0.9472034
 0.9645609 0.9763955 0.9843839 0.9897300
 0.9932813 0.9956252
```

```
> qnbinom(alpha, size= , prob=p, lower.tail=,
log.p=)

> qnbinom(0.5,2.5,0.4,lower.tail=TRUE)
[1] 3
```

Série des valeurs

```
> rnbinom(k, size= , prob=p)
> rnbinom(10,2.5,0.4)
[1] 4 6 6 4 6 6 3 5 5 6

> rnbinom(70,2.5,0.4)
 [1]  5  1  8  1  1 12  0  7  1  3  1  1  4  5  3  3  9  1  1 12
 1  1 2  3  2  8  6  2  2  4  2  2  8  4  5  0  5  2  8  6  0  0
 2  2 10  3  8  0  1  2 9  1 11  0  1  4  3  7  5  9  5  7  6  3
 3  0  1  6  3  3
```

Histogramme

On peut la simuler à la main, mais le plus simple est d'utiliser la commande rnbinom.

```
N <- 100
x <- rnbinom(N, 2.5, 0.4)
hist(x,
    xlim=c(min(x),max(x)), probability=T,
nclass=max(x)-min(x)+1,
    col='lightblue',
    main='Loi binomiale négative, n=2.5, p=0.4')
lines(density(x,bw=1), col='red', lwd=3)
```

Loi géométrique

C'est la loi du nombre d'essais avant une réussite lors d'une suite d'épreuves de Bernoulli.

X suit la loi géométrique de paramètre p
X indique le temps d'attente du premier succès , sachant que la probabilité d'un succès est p .
X=0 signifie succès au premier coup , X=1 signifie succès au 2ème coup ,….

Exemple

Calculons la probabilité d'avoir la boule noire au bout de 4 tirages (après au moins 3 échecs) avec l'urne contenant 10 boules noires et 40 rouges avec la formule :

```
p=10/50 ,n=4,r=3
p*(1-p)^(n-1) /p*(1-p)^r
```

0.1024

avec la fonction de R

```
>  p=10/50 ,
>  n=4,
>   r=3
dgeom(r,p)
```

[1] 0.1024

Forme de la distribution

```
plot(n,dgeom(n-1,p),xlab="nombre de tirages
   n",ylab="probabilté f(n)",cex.lab=1.5,cex.axis=1.5)
```

- **plot() :** Une des fonctions graphiques la plus utilisée .Ceci est une fonction générique et le type de graphique produit dépend du type ou classe de son premier argument.
- **cex.lab** = 1.5 : grandit les labels d'axes (noms) d'un facteur 1.5
- **cex.axis** = 1.5 : grandit le texte des graduations d'une facteur 1.5
- ***xlab***='abscisses',
- ***ylab***="ordonnées

Exemple 2

Calculons la probabilité d'avoir la boule noire en au moins 4 tirages (après au moins trois échecs)

avec l'urne contenant 10 boules noires et 40 rouges.

le résultat est obtenu avec la fonction de répartition $f(2)_{droite}=p(x>=3)+p(x>2)$

```
> P=10/50 ,
> r=2
> pgeom(2,10/50,lower.tail=FALSE)
```

Série des valeurs

Exemple 3 :

```
> rgeom(60,0.25)

4 5 4 4 3 6 16 1 0 9 2 3 1 1 3 4 0 2 8 0 0
4 0 1 2 0 0 2 1 0 6 3 7 2 2 5 1 0 12 2 9
0 5 4 2 1 1 2 2 0 1 5 2 3 3 1 2 0 4 7
```

Loi hypergéométrique

C'est la loi qu'on rencontre lors d'un échantillonnage sans remise.

Densité

Calculs avec R. A la base, les commandes comprennent l'expression «hyper» précédée d'une lettre spécifiant le calcul à réaliser. Les deuxième, troisième et quatrième options de la commande sont respectivement N1,N−N1, n. Par exemple, si (X)=(10 ; 40 ; 20), alors P(X=3) se détermine avec la commande

```
> dhyper(3,20,20,10)
[1] 0.1042549
```

Fonction de répartition

Reprenons le même exemple:

si (X)=(10 ; 40 ; 20), alors P(X≤ 3) se détermine avec la commande :

```
> phyper(3,20,20,10,lower.tail=TRUE)
[1] 0.1366715
```

si (X)=(10 ; 40 ; 20), alors P(3 < X)=1−FX(3)

```
> phyper(3,20,20,10,lower.tail=FALSE)
[1] 0.8633285
```

Fonction de répartition

(X)=(10 ; 40 ; 20):

Calculons les fonctions de répartition entre 5 et 10:

```
> phyper(5:10,20,20,10)
 [1] 0.6417867 0.8633285 0.9675834 0.9958191
     0.9997820 1.0000000
```

Série des valeurs

```
> rhyper(50,20,20,10)
 [1] 5 5 5 4 5 4 5 3 6 5 5 2 7 6 4 5 7 6 7 5 5 2 6 6 6
6 2 5 5 5 6 8 4 5 6 6 7 4 5 6 4 5 7 4 5 7 3 6 5 4
```

Si (X)=(10 ; 40 ; 20), alors P(3<X≤8) se détermine avec la commande:

sum(dhyper(4:8,30,20,10)) ; réponse :

[1] 0.9327374

Si (X)=(10 ; 40 ; 20), alors la médiane Me[X] se détermine avec la commande quantile:

```
>  qhyper(0.5,20,20,10,lower.tail=TRUE)
[1] 5
```

Histogramme

Voyons ce que cela donne sur un exemple, à l'aide d'une simulation. On tire 5 boules d'une urne qui en contient 15 blanches et 5 rouges, et on compte le nombre de boules blanches.

Histogramme

```
> N <- 10000
> n <- 5
> urne <- c(rep(1,15),rep(0,5))
> x <- NULL
> for (i in 1:N) {
  x <- append(x, sum(sample( urne, n, replace=F )))
}
> hist(x,
    xlim=c(min(x),max(x)), probability=T, nclass=max(x)-min(x)+1,
    col='lightblue',
    main='Loi hypergéométrique, n=20, p=0.75; k=5')
> lines(density(x,bw=1), col='red', lwd=3)
```

Histogramme

On peut aussi utiliser directement la commande rhyper (c'est plus rapide).

```
>  N <- 10000
> n <- 5
> x <- rhyper(N, 15, 5, 5)
> hist(x,
    xlim=c(min(x),max(x)), probability=T, nclass=max(x)-
min(x)+1,
    col='lightblue',
    main='Loi hypergéométrique, n=20, p=.75, k=5')
> lines(density(x,bw=1), col='red', lwd=3)
```

Loi poisson

C'est la loi qui décrit le nombre de clients qui arrivent dans une file d'attente pendant une unité de temps. Ou le nombre de fautes dans un texte. Ou le nombre de désintégrations radioactives par unité de temps.

Loi poisson

De manière plus formelle, c'est une loi de probabilité telle que :

- la probabilité d'observer un évènement dans un "petit" intervalle est proportionnelle à la taille de cet intervalle (en particulier, ça ne dépend que de la taille, pas la position de l'intervalle) ;
- la probabilité que l'évènement se produise dans un intervalle est indépendante de la probabilité d'apparition dans n'importe quel intervalle disjoint
- il n'y a pas d'évènements simultanés.

Densité

- Exemple:

Si (X)=(1.5), alors P(X=2) se détermine avec la commande :

```
> dpois(2,1.5)
> [1] 0.2510214
```

Densité

- Les valeurs P(X=0) P(X=1) P(X=2) P(X=3) se déterminent avec la commande :

```
> dpois( c(0,1,2,3),1.5 )
[1] 0.2231302 0.3346952 0.2510214 0.1255107
```

Fonction de répartition

- Nous donnons quelques valeurs de la fonctions de répartition FX(0)=P(X0), FX(1)=P(X1) FX(2)=P(X2), FX(3)=P(X3) qui se déterminent avec la commande :
- 1ère méthode:

```
> ppois( c(0,1,2,3),1.5,lower.tail=TRUE )
[1] 0.2231302   0.5578254   0.8088468
0.9343575
```

Fonction de répartition

```
> ppois( c(0,1,2,3),1.5,lower.tail=FALSE )

 [1] 0.77686984   0.44217460  0.19115317
       0.06564245
```

Valeurs de la fonction de répartition

2ème méthode:

```
> ppois(0:15,1.5 )
0.2231302 0.5578254 0.8088468 0.9343575
0.9814241 0.9955440 0.9990740
0.9998304 0.9999723 0.9999959 0.9999994
0.9999999 1.0000000 1.0000000
1.0000000 1.0000000
```

Valeurs de la fonction de répartition

```
> rpois(0:15,1.5 )

[1] 1 2 3 4 0 1 1 2 2 0 2 3 2 3 3 0
```

Histogramme

```
N <- 10000
x <- rpois(N, 1)
hist(x,
    xlim=c(min(x),max(x)), probability=T,
nclass=max(x)-min(x)+1,
    col='lightblue',
    main='Loi de Poisson, lambda=1')
lines(density(x,bw=1), col='red', lwd=3)
```

Histogramme

```
N <- 10000
x <- rpois(N, 5)
hist(x,
    xlim=c(min(x),max(x)), probability=T,
nclass=max(x)-min(x)+1,
    col='lightblue',
    main='Loi de Poisson, lambda=5')
lines(density(x,bw=1), col='red', lwd=3)
```