ALGORITHMES PROBABILITÉS ET SIMULATIONS AVEC **R**

Table des matières

ALGORITHMES PROBABILITÉS ET SIMULATIONS AVEC **R**

Hubert RAYMONDAUD LEGTA de Carpentras-Serres

A – INTRODUCTION

Les statistiques descriptives et les probabilités sont un terrain privilégié pour l'apprentissage et la mise en œuvre de l'algorithmique, de la seconde au BTS :

* Le programme de seconde aborde l'approche fréquentiste de la probabilité que l'on peut illustrer par des simulations élémentaires, le programme de première propose de simuler des distributions géométriques tronquées et des distributions binomiales, on y trouve l'intervalle de fluctuation (IF) d'une proportion que l'on peut aussi obtenir par simulation. L'IF sert à illustrer l'utilisation des probabilités (binomiales) pour prendre une décision en situation d'incertitude. Autant d'exercices propices à l'utilisation de l'algorithmique.

* On peut utiliser l'algorithmique pour calculer ou estimer des probabilités, pour résoudre des problèmes en simulant des expériences aléatoires, pour simuler des variables aléatoires de distribution donnée, pour obtenir des distributions simulées, pour simuler des intervalles de fluctuation en première , pour simuler des intervalles de confiance (IC) en terminale , pour faire de l' inférence, en BTS.

* La recherche de stratégies de simulation pour résoudre des problèmes de probabilité (et de statistique) constitue une véritable alternative à la résolution « classique » et peut offrir une ouverture pour des élèves peu à l'aise avec la formalisation mathématique.

* Les algorithmes sont peu « techniques » et peuvent facilement être mis en œuvre sur des machines, pour aboutir à des résultats concrets.

Il est important de bien différencier les deux façons d'aborder le calcul des probabilités :

- soit par la simulation en utilisant des modèles d'urnes, on obtient alors des valeurs estimées par simulation
- soit formellement, par l'utilisation de modèles mathématiques exprimant des lois de probabilité basées sur des hypothèses, on obtient alors des valeurs « exactes ».

Pour illustrer ces propos, je vais vous présenter (quelques) exemples, pris dans le domaine de la simulation d'expériences aléatoires et dans le domaine du calcul des probabilités. Ces exemples restent dans le cadre des programmes du collège et du lycée. **Pour chaque exemple, j'ai presque toujours présenté les deux alternatives de résolution : simulation et formalisation mathématique. Lorsque c'est possible et pertinent, j'ai mis l'accent sur les distributions desquelles on peut déduire les valeurs ponctuelles solutions des problèmes posés.**

Mode d'emploi :

Les textes en **`orange`** contiennent les lignes de commande **R**. Ils peuvent être copié-collé directement dans la "console" **R**. Lors du collé toutes les lignes sont exécutées sauf la dernière, il suffit alors de valider pour l'exécuter et voir se terminer l'affichage des résultats. Attention, dans **R** il faut respecter la casse.

Les lignes en **`vert`** sont des parties de réponses de **R**, à ne pas coller dans la console.

Les textes en **`turquoise`** contiennent le code des fonctions **R** (c'est un langage "fonctionnel" c'est à dire que la meilleure façon de le programmer est sous forme de fonctions au sens informatique du terme). Un façon simple d'exécuter une fonction est de copier-coller son code dans la console **R**, en validant pour terminer le collé de la dernière ligne. On exécute ensuite la fonction en saisissant sont nom, suivit, sans espace, de (). Ce nom figure obligatoirement en début du code. Dans toutes ces fonctions j'ai fait figurer des valeurs de paramètre par défaut, indiquées dans la première ligne du code de la fonction. On peut donc exécuter ces fonctions sans préciser de valeur de paramètre entre les (), ce sont les paramètres par défaut qui seront effectifs. Pour utiliser d'autres valeurs, il suffit de les indiquer à l'intérieur des (). Exemple : **`simurnremd()`** réalise 1000 simulations d'un échantillon de 4 tirages avec remise dans une urne contenant 3 rouges et 5 blanches, et détermine la distribution du nombre de rouges obtenues ; alors que **`simurnremd(n=6,r=10,b=30,nsim=2000)`** réalise 2000 simulations d'un échantillon de 6 tirages avec remise dans une urne contenant 10 rouges et 30 blanches, et détermine la distribution du nombre de rouges obtenues.

Pour exécuter à nouveau la fonction, il suffit d'appuyer sur la flèche ↑ puis valider. On peut donc ainsi exécuter la fonction plusieurs fois très simplement et observer les variations des séries de simulations.

Les algorithmes présentés ne sont pas forcément les plus "élégants", j'ai favorisé l'utilisation de commandes variées de **R**, afin de constituer une sorte de bibliothèque.

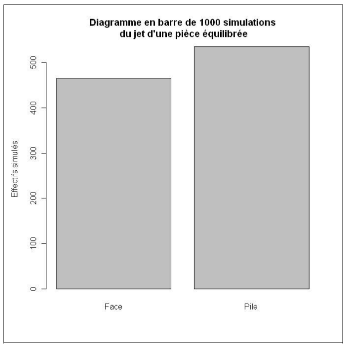
B – SIMULATIONS D'EXPÉRIENCES ALÉATOIRES – DISTRIBUTIONS SIMULÉES

1° Le sens fréquentiste de la probabilité : mettre en œuvre la loi des grand nombres

► Exemple de 1000 simulations du jet d'une pièce équilibrée pour estimer la probabilité d'obtenir "pile" (codé 1).

```
piece1<-sample(c("Pile", "Face"), 1000, replace=TRUE)
(distpiece1 <- table(piece1))
  piece1
  Face Pile
   465  535
barplot(distpiece1)
sum(piece1 == "Pile"])/1000 # estimer une probabilité
  [1] 0.535
```

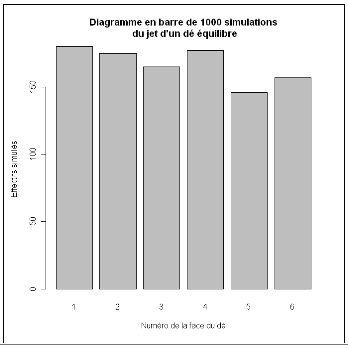
Une estimation de la probabilité d'obtenir "pile" est 535/1000.



► Exemple de 1000 simulations du jet d'un dé à 6 faces équilibrées pour estimer la probabilité d'obtenir la face "4".

```
de1<-sample(c(1:6),1000,replace=TRUE)
(distde1 <- table(de1))
  de1
     1   2   3   4   5   6
  180 175 165 177 146 157
barplot(distde1)
sum(de1 == 4)/1000 # estimer une probabilité
  [1] 0.177
```

Une estimation de la probabilité d'obtenir la face "4" est 177/1000.



2° Simuler la somme des valeurs des faces obtenues en lançant 2 dés à 6 faces équilibrées

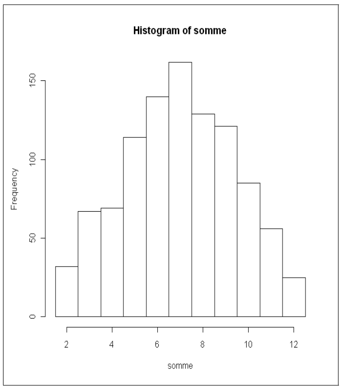
L'expérience consiste à lancer deux dés à 6 faces équilibrées et à faire la sommes S des "valeurs" obtenues.

Il s'agit alors de déterminer la distribution simulée de S, et d'en déduire une estimation de la probabilité d'obtenir la somme 7.

Exemple de 1000 simulations :

```
de1<-sample(c(1:6),1000,replace=TRUE)
de2<-sample(c(1:6),1000,replace=TRUE)
somme<-de1+de2
(hist(somme,breaks=seq(1.5,12.5,1)))
  $breaks
  [1]  1.5  2.5  3.5  4.5  5.5  6.5  7.5  8.5  9.5 10.5 11.5 12.5
  $counts
  [1]  32  67  69 114 140 162 129 121  85  56  25
sum(somme == 7)/1000 # estimer une probabilité
  [1] 0.162
```

Une estimation de la probabilité d'obtenir la somme 7 est 162/1000.



3° Simuler le problème historique du grand duc de Toscane

Le Duc de Toscane, qui avait sans doute observé un grand nombre de parties du jeu consistant à faire la somme des nombres obtenus en jetant 3 dés, avait constaté que la somme 10 apparaissait légèrement plus souvent que la somme 9. Le paradoxe, que le Duc avait exposé à Galilée (1554-1642), réside dans le fait qu'il y a autant de façons d'écrire 10 que 9 comme sommes de trois entiers compris entre 1 et 6. Élaborons une simulation correspondant à ce problème.

`# Estimation de la probabilité d'un événement`
`toscanep = function(nsim = 1000){`
`   de1 <- sample(1:6,nsim,replace=T)`
`   de2 <- sample(1:6,nsim,replace=T)`
`   de3 <- sample(1:6,nsim,replace=T)`
`   jeu <- de1 + de2 + de3`
`   neuf <- sum(jeu == 9)`
`   dix <- sum(jeu == 10)`
`#************* Affichage des résultats *******************`
`   cat("Fréquence des neuf =",neuf/nsim,"\n")`
`   cat("Fréquence des dix =",dix/nsim,"\n")`
`}`
	`toscanep()`
	`  Fréquence des neuf = 0.137`
	`  Fréquence des dix = 0.116`
	`toscanep()`
	`  Fréquence des neuf = 0.115`
	`  Fréquence des dix = 0.136`
	`toscanep(10000)`
	`  Fréquence des neuf = 0.1182`
	`  Fréquence des dix = 0.1272`
	`toscanep(10000)`
	`  Fréquence des neuf = 0.1169`
	`  Fréquence des dix = 0.128`

4° Simuler le problème historique du chevalier de Méré

Est-il avantageux, lorsqu'on joue au dé, de parier sur l'apparition d'un 6 en lançant 4 fois le dé? Est-il "avantageux" de parier sur l'apparition d'un double-six, quand on lance 24 fois deux dés? Le chevalier de Méré, qui était un grand joueur, avait remarqué que le premier jeu était "avantageux". Il pensait que le deuxième, aussi, était "avantageux". Avait-il raison? Précisions stratégiques : "avantageux" signifie plus de 50% de chance de réalisation, et obtenir 6 lors du jeu, c'est obtenir au moins une fois 6 sur tous les lancers.

La fonction principale `meresix()` utilise deux fonctions "annexes" `quatrejets()` et `vingtquatrejets()` qui renvoient 1 si il y a eu, respectivement, au moins un six dans les 4 jets d'un dé ou au moins un double 6 dans les 24 jets de deux dés, ou 0 sinon. Il est instructif de noter qu'il n'y a pas eu besoin de boucles dans ces fonctions annexes, **R** possédants des fonctions "vectorielles" élaborées permettant d'éviter les boucles, très consommatrices de temps de calcul et rendant plus difficile la lecture des algorithmes.

Les fonctions **R** ont une structure plus simple que les algorithmes LARP.

```
# Fonction auxiliaire
quatrejets = function(){
   unsixouplus <- 0
   jeu <- sample(1:6,4,replace=T)
   nbsix = sum(jeu == 6)
   if(nbsix >= 1) unsixouplus <- 1
   return(unsixouplus)
}
# Fonction auxiliaire
vingtquatrejets = function(){
   undoublesixouplus <- 0
   de1 <- sample(1:6,24,replace=T)
   de2 <- sample(1:6,24,replace=T)
   nbdoublesix <- sum(de1[which(de1 == de2)] == 6)
   if(nbdoublesix >= 1) undoublesixouplus <- 1
   return(undoublesixouplus)
}
# Estimation de la probabilité d'un événement
meresix = function(nsim = 2000){
   aumoinsunsix <- 0
   aumoinsundoublesix <- 0
   for(j in 1:nsim){
   aumoinsunsix <- aumoinsunsix + quatrejets()
   aumoinsundoublesix <- aumoinsundoublesix + vingtquatrejets()
                }
#************* Affichage des résultats *******************
   cat("Fréquence des six =",aumoinsunsix/nsim,"\n")
   cat("Fréquence des doubles six =",aumoinsundoublesix/nsim,"\n")
}
```

```
meresix()
Fréquence des six = 0.5215
Fréquence des doubles six =
0.4815

meresix()
Fréquence des six = 0.5125
Fréquence des doubles six =
0.485
```

5° Simuler le problème historique du croix ou pile de d'Alembert

► Il s'agit de trouver la probabilité d'amener croix (gagnant) en deux coups au plus. La partie s'arrête dès que l'on a gagné, elle peut donc comporter un ou deux coups. Le fait que la partie s'arrête dès que l'on gagne, rend la modélisation moins facile. Par contre la simulation reste assez simple.

```
# Estimation de la probabilité d'un événement
croixpile = function(nrep = 1000){
      statcroix <- 0
      for(i in 1:nrep){
          croix <- 0
          jeu <- sample(c(0,1),1)
          if(jeu == 1) croix <- 1  else {
              jeu <- sample(c(0,1),1)
              if(jeu == 1) croix <- 1
                                           }
          statcroix <- statcroix + croix
                     }
#****************** Affichage des résultats **********************
print("Une estimation de la probabilité de croix en jouant au
plus 2 fois vaut :")
print(statcroix/nrep)
}
```

```
croixpile()
  [1] "Une estimation de la
 probabilité de croix en
 jouant au plus 2 fois
 vaut :"
  [1] 0.758

croixpile(5000)
  [1] "Une estimation de la
 probabilité de croix en
 jouant au plus 2 fois
 vaut :"
  [1] 0.7496
```

► On peut généraliser le croix ou pile et simuler pour estimer la probabilité de gagner (croix) en au plus k lancers, au sens de d'Alembert, c'est à dire dans un jeu de n coups au plus, le jeu s'arrêtant quand on gagne.

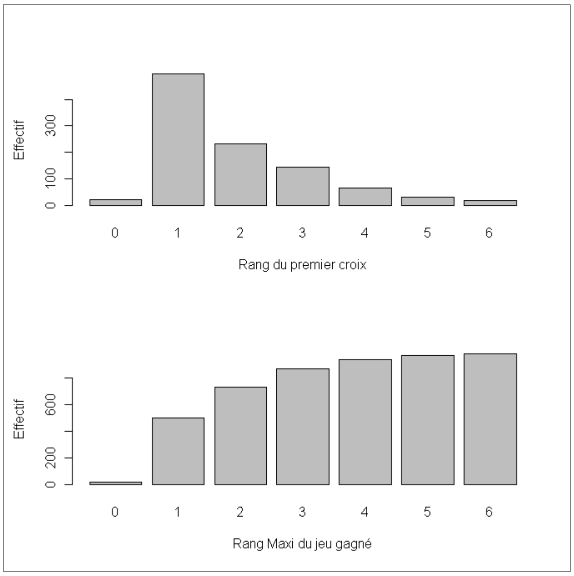
```
# Estimation de la probabilité d'un événement
croixpilegenep = function(n = 6, k = 4, nrep = 1000){
   statkcroix <- 0
   stat0croix <- 0
   for(j in 1:nrep){
      i <- 1
      croix <- 0
      while((croix == 0) & (i <= n)){
      jeu <- sample(c(0,1),1)
      if(jeu == 1) {
         croix <- 1
         xtronk <- i} else {
         croix <- 0
         xtronk <- 0
                     }
         i <- i + 1
                              }
      if (xtronk == 0) stat0croix <- stat0croix + 1 else {
          if(xtronk <= k) statkcroix <- statkcroix + 1    }
                  }
#************* Affichage des résultats *******************
   cat("Une estimation de la probabilité de gagner en ")
   cat(k, "coups \n au plus, en jouant au plus", n, "fois est : ")
   if(k == 0) cat(stat0croix/nrep, "\n") else cat(statkcroix/nrep,
"\n")
}
```

```
croixpilegenep()
  Une estimation de la
  probabilité de gagner
  en 4 coups au plus, en
  jouant au plus 6 fois
  est : 0.925
```

► On peut modifier la simulation précédente, pour obtenir la distribution simulée du rang d'apparition du premier croix. Le nom de cette distribution apparaît dans le programme de première... On peut alors en déduire un tableau donnant la probabilité de gagner au sens de D'Alembert. (cumulé croissant, à partir de 1, sans 0).

```
# Distribution simulée d'une variable
croixpilegenef = function(n = 6, nsim = 1000){
   distsim <- vector(length = nsim)
   tabloeffec <- rep(0,n+1)
   names(tabloeffec) <- 0:n
   for(j in 1:nsim){
     i <- 1
     croix <- 0
     while((croix == 0) & (i <= n)){
     jeu <- sample(c(0,1),1)
     if(jeu == 1) {
        croix <- 1
        xtronk <- i} else {
        croix <- 0
        xtronk <- 0
                          }
        i <- i + 1
                                   }
     distsim[j] <- xtronk
                   }
   effec <- table(distsim)
   tabloeffec[as.numeric(names(effec))+1] <- effec
   tabloeffecum2 <- cumsum(tabloeffec)-tabloeffec[1]
   tabloeffecum2[1] <- tabloeffec[1]
#************* Affichage des résultats *******************
   print ("tabloeffec/nsim")
   print (tabloeffec/nsim)
   print ("tabloeffecum2/nsim")
   print (tabloeffecum2/nsim)
   par(mfrow = c(2,1))
   barplot(tabloeffec, xlab = "Rang du premier croix",
ylab = "Effectif")
   barplot(tabloeffecum2, xlab = "Rang Maxi du jeu gagné",
ylab = "Effectif")
}
```

```
croixpilegenef()
[1] "tabloeffec/nsim"
    0     1     2     3     4     5     6
0.020 0.496 0.231 0.142 0.065 0.029 0.017
[1] "tabloeffecum2/nsim"
    0     1     2     3     4     5     6
0.020 0.496 0.727 0.869 0.934 0.963 0.980
```



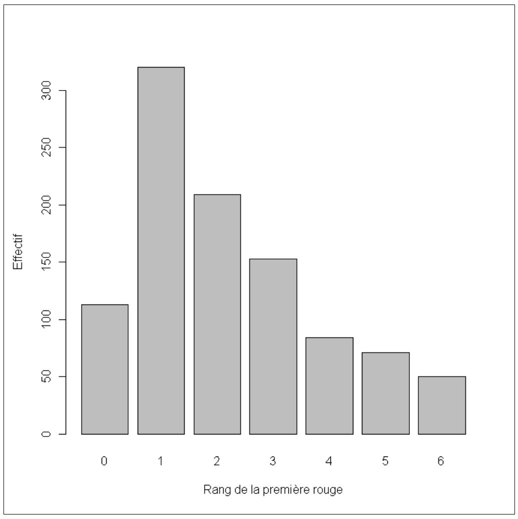
Lors d'un jeu en 6 lancers au plus, une estimation de la probabilité de gagner en 4 coup au plus est : 0,934.

6° Simuler la distribution du rang de la première boule rouge tirée, tirage AVEC remise

Modèle d'urne : Une urne contient **m** boules dont **r** rouges. On tire, successivement avec remise **n** boules dans l'urne et on note leurs couleurs dans l'ordre. La variable aléatoire X étudiée est le rang de la première rouge tirée (=0 si aucune rouge tirée au bout de n fois).

```
# Distribution simulée d'une variable
geotronkd = function(n = 6, r = 30, m = 100, nsim = 1000){
   urne <- seq(1,m)
   premrouge <- vector(length = nsim)
   tabloeffec <- rep(0,n+1)
   names(tabloeffec) <- 0:n
   for(j in 1:nsim){
     expe <- sample(urne,n,replace=T)
     if(min(expe) > r)    {
        premrouge[j] <- 0} else {
        premrouge[j] <- min(which(expe <= r))
                             }
                  }
   effec <- table(premrouge)
   tabloeffec[as.numeric(names(effec))+1] <- effec
#************* Affichage des résultats *******************
   print ("tabloeffec/nsim")
   print (tabloeffec/nsim)
   barplot(tabloeffec, xlab = "Rang de la première rouge",
ylab = "Effectif")
}
```

```
geotronkd()
[1] "tabloeffec/nsim"
    0     1     2     3     4     5     6
0.113 0.320 0.209 0.153 0.084 0.071 0.050
```



7° Simuler le problème du QCM

Un QCM comprend 20 questions. Pour chacune de ces questions QUATRE propositions de réponse sont faites, dont UNE seule est la bonne. Une bonne réponse vaut 1 point, une mauvaise vaut 0. Quelle est la probabilité qu'un élève, répondant à toutes les questions au hasard, ait la moyenne?

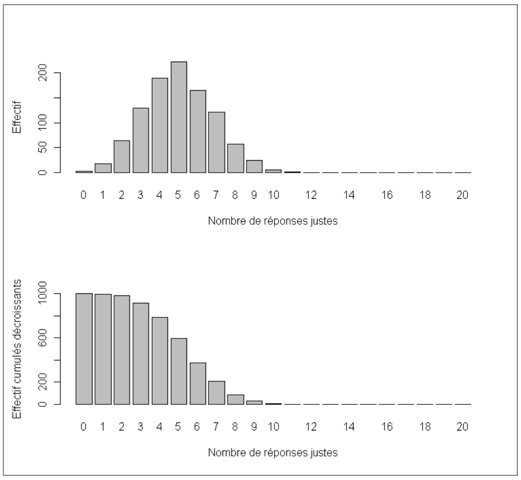
Élaborons une stratégie de simulation pour estimer cette probabilité.

```
# Estimation de la probabilité d'un événement
qcmp = function(nquest = 20, nrep = 4, k = 10, nsim = 1000){
   vectexam <- vector(length = nsim)
   for(i in 1:nsim){
      vectepreuv <- sample(1:nrep,nquest,replace=T)
      vectexam[i] <- sum(vectepreuv == 1)
                 }
   effeve <- sum(vectexam >= k)
#************* Affichage des résultats ******************
cat("Une estimation de la probabilité d'avoir au
moins",k,"bonnes \n réponses sur",nquest,"vaut :",
effeve/nsim,"\n")
}
```

```
qcmp()
  Une estimation de la
  probabilité d'avoir au
  moins 10 bonnes réponses
  sur 20 vaut : 0.013
```

```
# Distribution simulée d'une variable
qcmd = function(nquest = 20, nrepon = 4, k = 10,
nsim = 1000){
   vectexam <- vector(length = nsim)
   tabloeff <- rep(0,nquest+1)
   names(tabloeff) <- 0:nquest
   for(i in 1:nsim){
      vectepreuv <-
sample(1:nrepon,nquest,replace=T)
      vectexam[i] <- sum(vectepreuv == 1)
                 }
   eff <- table(vectexam)
   tabloeff[as.numeric(names(eff))+1] <- eff
   cumdecroi <- (cumsum(tabloeff[(nquest+1):1]))
[(nquest+1):1]
#************* Affichage des résultats ******************
   print("tabloeff/nsim"); print(tabloeff/nsim)
   print("cumdecroi/nsim");print(cumdecroi/nsim)
   par(mfrow=c(2,1))
   barplot(tabloeff, xlab = "Nombre de réponses
justes", ylab = "Effectif")
   barplot(cumdecroi, xlab = "Nombre de réponses
justes", ylab = "Effectif cumulés décroissants")
}
```

```
qcmd()
[1] "tabloeff/nsim"
    0     1     2     3     4     5     6     7     8     9    10 
0.002 0.018 0.064 0.130 0.189 0.222 0.165 0.122 0.057 0.024 0.006 
   11    12    13    14    15    16    17    18    19    20 
0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 
[1] "cumdecroi/nsim"
    0     1     2     3     4     5     6     7     8     9    10 
1.000 0.998 0.980 0.916 0.786 0.597 0.375 0.210 0.088 0.031 0.007 
   11    12    13    14    15    16    17    18    19    20 
0.001 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 
```



8° Simuler un modèle d'urne, échantillonnage AVEC remise

Une urne contient 3 boules rouges et 5 boules blanches. On tire au hasard 4 boules **AVEC remise**. Quelle est la probabilité d'obtenir 3 boules rouges ? Quelle est la probabilité d'obtenir 3 boules blanches?

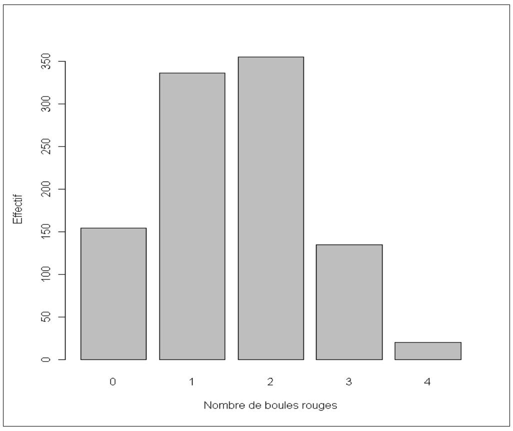
Il y a deux méthodes possibles, soit construire une fonction qui reproduit le modèle d'urne, soit utiliser la fonction rbinom() qui génère directement des nombres à distribution binomiale.

```
# Estimation de la probabilité d'un événement
simurnremp = function(n = 4, r = 3, b = 5, k=3, nsim = 1000){
  urne <- rep(c(1, 0),c(r, b))
  cpteven <- 0
  for (i in 1:nsim){
        vectepreuv <- sample(urne,n,replace=T)
        resultexpe <- sum(vectepreuv)
        if (resultexpe == k) cpteven <- cpteven +1
                   }
#************* Affichage des résultats ******************
  cat("Estimation de la probabilité de", k, "boules
rouges)=", cpteven/nsim,"\n")
}
```

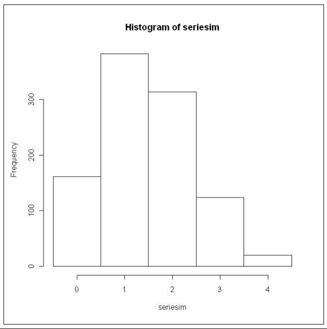
```
simurnremp()
  Estimation de la
  probabilité de 3 boules
  rouges)= 0.123
simurnremp(k=1)
  Estimation de la
  probabilité de 3 boules
  blanches)= 0.364
simurnremp(r=300,b=500)
  Estimation de la
  probabilité de 3 boules
  rouges)= 0.128
```

```
# Distribution simulée d'une variable
simurnremd = function(n = 4, r = 3, b = 5, nsim = 1000){
  urne <- rep(c(1, 0),c(r, b))
  DistSim <- vector(length = nsim)
  tabloeffec <- rep(0,n+1)
  names(tabloeffec) <- 0:n
  for (i in 1:nsim){
        vectepreuv <- sample(urne,n,replace=T)
        vectexpe <- sum(vectepreuv)
        DistSim[i] <- vectexpe
                   }
  effec <- table(DistSim)
  tabloeffec[as.numeric(names(effec))+1] <- effec
#************* Affichage des résultats ******************
  print ("tabloeffec/nsim")
  print (tabloeffec/nsim)
  barplot(tabloeffec, xlab = "Nombre de boules rouges",
ylab = "Effectif")
}
```

```
simurnremd()
[1] "tabloeffec/nsim"
    0     1     2     3     4
0.154 0.336 0.355 0.135 0.020
```




```
# Utilisation de la fonction rbinom(), résultat de 1000 simulations :
# Distribution simulée et estimation d'une probabilité.
seriesim<-rbinom(1000,4,3/8)
(hist(seriesim,breaks=c(-.5,.5,1.5,2.5,3.5,4.5)))
  $breaks
  [1] -0.5  0.5  1.5  2.5  3.5  4.5
  $counts
  [1] 161 382 313 124  20
length(seriesim[seriesim==3])/1000
  [1] 0.124
```

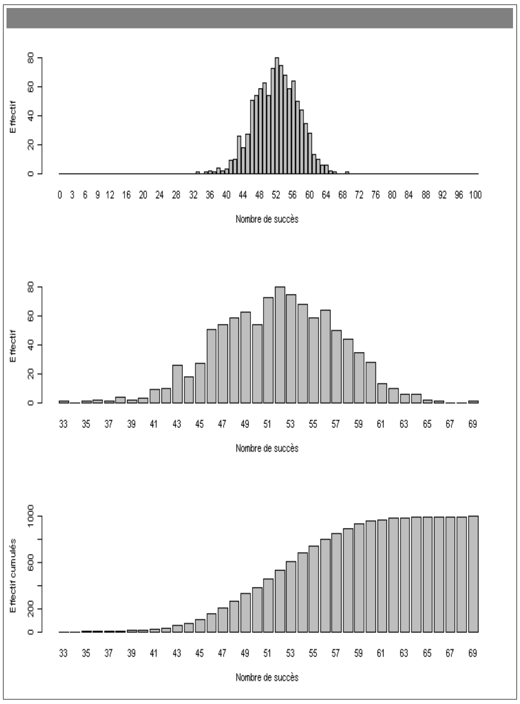


9° Simuler l'intervalle de fluctuation (IF) d'une proportion

Exemple 1 (document de l'inspection) : Monsieur Z, chef du gouvernement d'un pays lointain, affirme que 52 % des électeurs lui font confiance. On interroge 100 électeurs au hasard (la population est suffisamment grande pour considérer qu'il s'agit de tirages avec remise) et on souhaite savoir à partir de quelles fréquences, au seuil de 5 %, on peut mettre en doute le pourcentage annoncé par Monsieur Z, dans un sens, ou dans l'autre. Sur les 100 électeurs interrogés au hasard, 43 déclarent avoir confiance en Monsieur Z. Peut-on considérer, au seuil de 5 %, l'affirmation de Monsieur Z comme exacte?

L'algorithme va permettre de simuler un tirage avec remise d'un échantillon de n individus statistiques, dans une population pour laquelle on fait une hypothèse pour la valeur de p. Dans l'exemple 1, L'**IF** est simulé à partir des 52% de la déclaration de Monsieur Z,. L'**IF** est obtenu à partir des quantiles de la série simulée et l'on note si les 43% (43 personnes favorables dans un échantillon aléatoire et simple de 100 personnes), appartiennent à cet intervalle. La démarche est identique pour les exemples 2 et 3 (cf. "ProbaEtAlgorithmes1.odt"), il suffit de changer la valeur de p dans les paramètres de la fonction **R**.

```
# Distribution simulée d'une variable : les 52% de Mr Z
simulif = function(n = 100, p =.52, nsim = 1000){
  tabloeffec <- rep(0,n+1)
  names(tabloeffec) <- 0:n
  vecbino <- rbinom(nsim, n, p)
  effec <- table(vecbino)
  minvec <- min(vecbino) ; maxvec <- max(vecbino) ; nbminmax
<- maxvec - minvec + 1
  tablominmax <- vector(length = nbminmax)
  names(tablominmax) <- minvec:maxvec
  tablominmax[as.numeric(names(effec)) - minvec + 1] <- effec
  tabloeffec[as.numeric(names(effec))+1] <- effec
  cuminmax <- cumsum(tablominmax)
  quantiles <- quantile(vecbino, probs =
c(.5,1,2.5,5,25,50,75,95,97.5,99,99.5)/100)
#************* Affichage des résultats ******************
  print(tablominmax)
  cat("\n")
  print("Quantiles")
  print(quantiles)
  par(mfrow = c(3,1))
  barplot(tabloeffec, xlab = "Nombre de succès", ylab =
"Effectif")
  barplot(tablominmax, xlab = "Nombre de succès", ylab =
"Effectif")
  barplot(cuminmax, xlab = "Nombre de succès", ylab =
"Effectif cumulés")
}
 [1] "Quantiles"
   0.5%   1% 2.5%   5%   25%   50%   75%   95%   97.5%   99% 99.5%
 37.995 39   42    43    48    52    56    60      62    63     64
```



10° Simuler le problème du match France-Irlande par le XV de France (anniversaires)

10 supporteurs sont réunis pour fêter la victoire du XV de France contre l'Irlande. Quelle est la probabilité qu'au moins deux d'entre-eux aient le même jour anniversaire? Même question avec 20, 30 puis 40 supporteurs.

```
# Estimation de la probabilité d'un événement
simanivp = function(n = 10, nsim = 1000){
    annee <- 1:365
    tousdiff <- 0
    for(j in 1:nsim){
        vecexpe <- sample(annee,n,replace=T)
        if(length(vecexpe) == length(unique(vecexpe))) tousdiff <- tousdiff + 1
                    }
    aumoinsdeux <- (1 - tousdiff/nsim)
#************* Affichage des résultats ******************
    cat("Une estimation de la probabilité qu'au moins deux personnes \n")
    cat("aient le même jour anniversaire dans une assemblée de",n,"\n")
    cat("personnes, vaut :", aumoinsdeux,"\n")
}
simanivp(47)
  Une estimation de la probabilité qu'au moins deux personnes
  aient le même jour anniversaire dans une assemblée de 47
  personnes, vaut : 0.961
```

```
simanivp()
  Une estimation
  de la
  probabilité
  qu'au moins
  deux personnes
  aient le même
  jour
  anniversaire
  dans une
  assemblée de 10
  personnes, vaut
  : 0.122
```

11° Simuler un modèle d'urne, échantillonnage SANS remise

Une urne contient 3 boules rouges et 5 boules blanches. On tire au hasard 4 boules **SANS remise**. Quelle est la probabilité d'obtenir 3 boules rouges? Quelle est la probabilité d'obtenir 3 boules blanches?

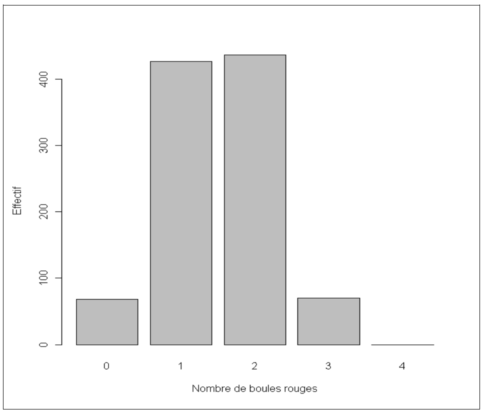
Il y a deux méthodes possibles, soit construire une fonction qui reproduit le modèle d'urne, soit utiliser la fonction rhyper() qui génère des nombres à distribution hypergéométrique.

```
# Estimation de la probabilité d'un événement
simurnsansp = function(n = 4, r = 3, b = 5, k=3, nsim = 1000){
  urne <- rep(c(1, 0),c(r, b))
  cpteven <- 0
  for (i in 1:nsim){
        vectepreuv <- sample(urne,n,replace=F)
        resultexpe <- sum(vectepreuv)
        if (resultexpe == k) cpteven <- cpteven +1
                    }
#************* Affichage des résultats ******************
  cat("Estimation de la probabilité de", k, "boules rouges)=",
cpteven/nsim,"\n")
}
```

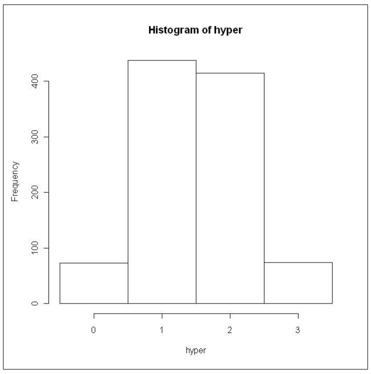
```
simurnsansp()
  Estimation de la
  probabilité de 3 boules
  rouges)= 0.085

simurnsansp(r=30,b=50)
  Estimation de la
  probabilité de 3 boules
  rouges)= 0.121

simurnsansp(r=300,b=500)
  Estimation de la
  probabilité de 3 boules
  rouges)= 0.132
```

```
# Distribution simulée d'une variable
simurnsansd = function(n = 4, r = 3, b = 5, nsim = 1000){
  urne <- rep(c(1, 0),c(r, b))
  DistSim <- vector(length = nsim)
  tabloeffec <- rep(0,n+1)
  names(tabloeffec) <- 0:n
  for (i in 1:nsim){
        vectepreuv <- sample(urne,n,replace=F)
        vectexpe <- sum(vectepreuv)
        DistSim[i] <- vectexpe
                    }
  effec <- table(DistSim)
  tabloeffec[as.numeric(names(effec))+1] <- effec
#************* Affichage des résultats ******************
  print ("tabloeffec/nsim")
  print (tabloeffec/nsim)
  barplot(tabloeffec, xlab = "Nombre de boules rouges",
ylab = "Effectif")
}
```

```
Simurnsansd()
[1] "tabloeffec/nsim"
     0     1     2     3     4
 0.068 0.426 0.436 0.070 0.000
```




```
# Utilisation de la fonction rhyper(), résultat de 1000 simulations :
# Distribution simulée et estimation d'une probabilité.
seriesim<-rhyper(1000,3,5,4)
(hist(seriesim,breaks=c(-.5,.5,1.5,2.5,3.5)))
  $breaks
  [1] -0.5  0.5  1.5  2.5  3.5
  $counts
  [1]  73 438 415  74
length(seriesim[seriesim==3])/1000
  [1] 0.074
```



Il est intéressant de comparer les résultats avec ceux obtenus avec un tirage avec remise (8°).

12° Simuler la distribution du rang de la première boule rouge tirée, tirage SANS remise

Modèle d'urne : Une urne contient **m** boules dont **r** rouges. On tire, successivement sans remise **n** boules dans l'urne et on note leurs couleurs dans l'ordre. La variable aléatoire X étudiée est le rang de la première rouge tirée (=0 si aucune rouge tirée au bout de n fois).

13° Simuler le problème des chaines de longueur 6

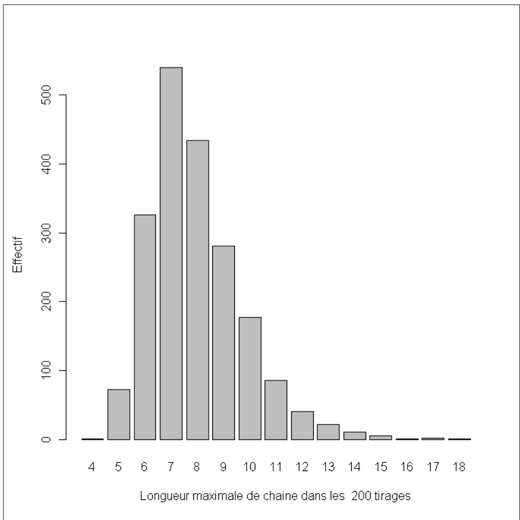
Il s'agit de trouver un algorithme pour estimer la probabilité d'obtenir au moins 6 piles ou 6 faces qui se suivent (on dira une chaine de longueur au moins 6) en jouant 200 fois à pile ou face avec une pièce équilibrée.

La solution mathématique n'est pas triviale.

```
# Estimation d'une probabilité.
chainesp = function(L = 6, tirages = 200, urne=1:2, nbsim = 2000){
   nblongL <- vector(length = nbsim)
   for(i in 1:nbsim){
      expe <- sample(urne,tirages,replace=T)
      rexpe <- rle(expe)
      if(sum(rexpe$length >= L)>0) nblongL[i] <- 1
                    }
      StatlongL <- sum(nblongL)
#************* Affichage des résultats ******************
   print("Une estimation de la probabilité de")
   print(" chaines de 6 ou plus vaut : ")
   print(StatlongL/nbsim)
}
```

```
chainesp()
 [1] "Une
estimation de la
probabilité de"
 [1] " chaines de
 6 ou plus vaut :
 "
 [1] 0.968
```

```
# Distribution simulée d'une variable
chainesd = function(L = 6, tirages = 200, urne=1:2, nbsim = 2000){
  nblongMax <- vector(length = nbsim)
  for(i in 1:nbsim){
      expe <- sample(urne,tirages,replace=T)
      rexpe <- rle(expe)
      nblongMax[i] <- max(rexpe$length)
                  }
  tablemax <- table(nblongMax)
  StatL <- sum(tablemax[(as.numeric(names(tablemax)) >= L)])/nbsim
  etendtablemax <- (max(as.numeric(names(tablemax))) - min(as.numeric(names(tablemax))))
  tabloeffec <- rep(0,etendtablemax + 1)
  names(tabloeffec) <- (min(as.numeric(names(tablemax)))):(max(as.numeric(names(tablemax))))
  tabloeffec[as.numeric(names(tablemax)) - min(as.numeric(names(tablemax))) + 1] <- tablemax
#********************************* Affichage des résultats *******************************
  cat("Une estimation de la probabilité de chaines de longueur",L,"ou plus")
  cat(" vaut :",StatL,"\n \n" )
  print("Distribution de la longueur maximale des chaines (runs)")
  print(tabloeffec)
  # print(tablemax)
  barplot(tabloeffec,
  xlab = paste("Longueur maximale de chaine dans les ", tirages, "tirages"),
  ylab = "Effectif")
}
```




```
chainesd()
Une estimation de la probabilité de chaines de longueur 6 ou
plus vaut : 0.963

[1] "Distribution de la longueur maximale des chaines (runs)"
  4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
  1  73 326 540 434 281 177  86  40  22  11   5   1   2   1
```

14° Simuler le problème des chapeaux de Montmort ou permutations sans point fixe

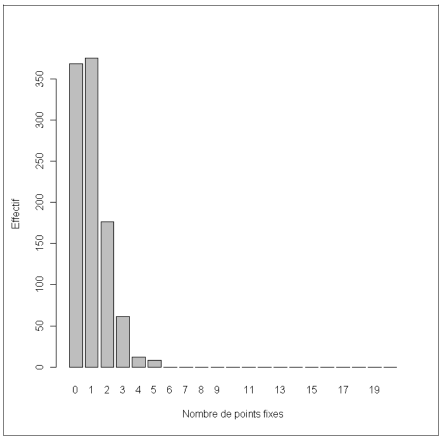
Avant de rentrer en réunion, 20 personnes suspendent leurs chapeaux aux crochets à l'extérieur de la salle. Un petit farceur mélange tous les chapeaux de façon aléatoire. On veut calculer la probabilité qu'aucune personne ne retrouve son chapeau à la place où il l'avait laissé.

```
# Estimation de la probabilité d'un événement
chapeauxp = function(m = 20, nsim = 1000){
  crochets <- seq(1,m)
  nbfixes <- vector(length = nsim)
  for(i in 1:nsim){
     chapeaux <- sample(crochets,m)
     fixes <- crochets - chapeaux
     nbfixes[i] <- sum(fixes == 0)
                    }
  sansfixe <- sum(nbfixes == 0)
#************* Affichage des résultats *******************
  cat("Estimation de la probabilité d'aucun
des",m,"chapeaux à leur place =\n")
  cat(sansfixe/nsim,"\n")
}
```

```
chapeauxp(m=2,nsim=10000)
Estimation de la probabilité d'aucun des 2
chapeaux à leur place = 0.4959
chapeauxp(m=5,nsim=10000)
Estimation de la probabilité d'aucun des 5
chapeaux à leur place = 0.3709
chapeauxp(m=10,nsim=10000)
Estimation de la probabilité d'aucun des 10
chapeaux à leur place = 0.3728
chapeauxp(m=20,nsim=10000)
Estimation de la probabilité d'aucun des 20
chapeaux à leur place = 0.374
chapeauxp(m=40,nsim=10000)
Estimation de la probabilité d'aucun des 40
chapeaux à leur place = 0.3611
chapeauxp(m=100,nsim=10000)
Estimation de la probabilité d'aucun des 100
chapeaux à leur place = 0.3677
```

```
# Distribution simulée d'une variable
# et estimation d'une probabilité
chapeauxd = function(m = 20, nsim = 1000){
  crochets <- seq(1,m)
  sansfixe <- 0
  nbfixes <- vector(length = nsim)
  tabloeffec <- rep(0,m+1)
  names(tabloeffec) <- 0:m
  for(i in 1:nsim){
     chapeaux <- sample(crochets,m)
     fixes <- crochets - chapeaux
     nbfixes[i] <- sum(fixes == 0)
                 }
  effec <- table(nbfixes)
  tabloeffec[as.numeric(names(effec))+1] <- effec
  print ("tabloeffec/nsim")
  print (tabloeffec/nsim)
  cat("\n")
  cat("Estimation de la probabilité d'aucun
des",m,"chapeaux à leur place =\n")
  cat(tabloeffec[1]/nsim, "\n")
  barplot(tabloeffec, xlab = "Nombre de points
fixes", ylab = "Effectif")
}
```

```
chapeauxd()
[1] "tabloeffec/nsim"
    0     1     2     3     4     5     6     7     8     9    10
0.368 0.375 0.176 0.061 0.012 0.008 0.000 0.000 0.000 0.000 0.000
   11    12    13    14    15    16    17    18    19    20
0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.000

Estimation de la probabilité d'aucun des 20 chapeaux à leur place
= 0.368
```

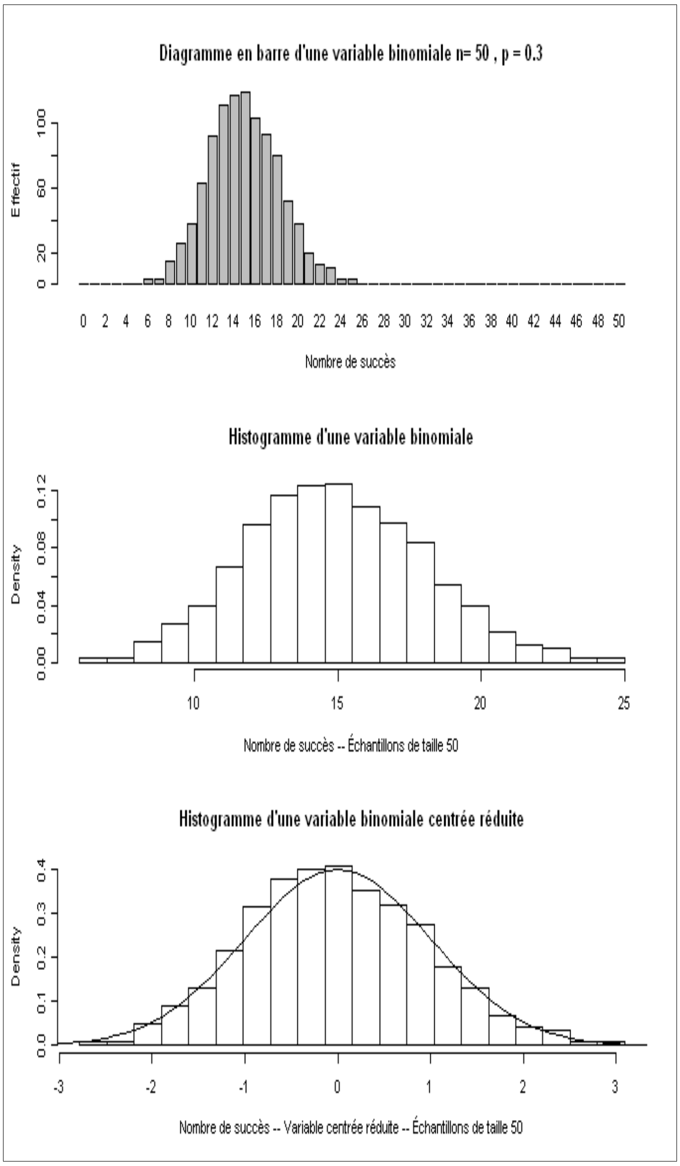


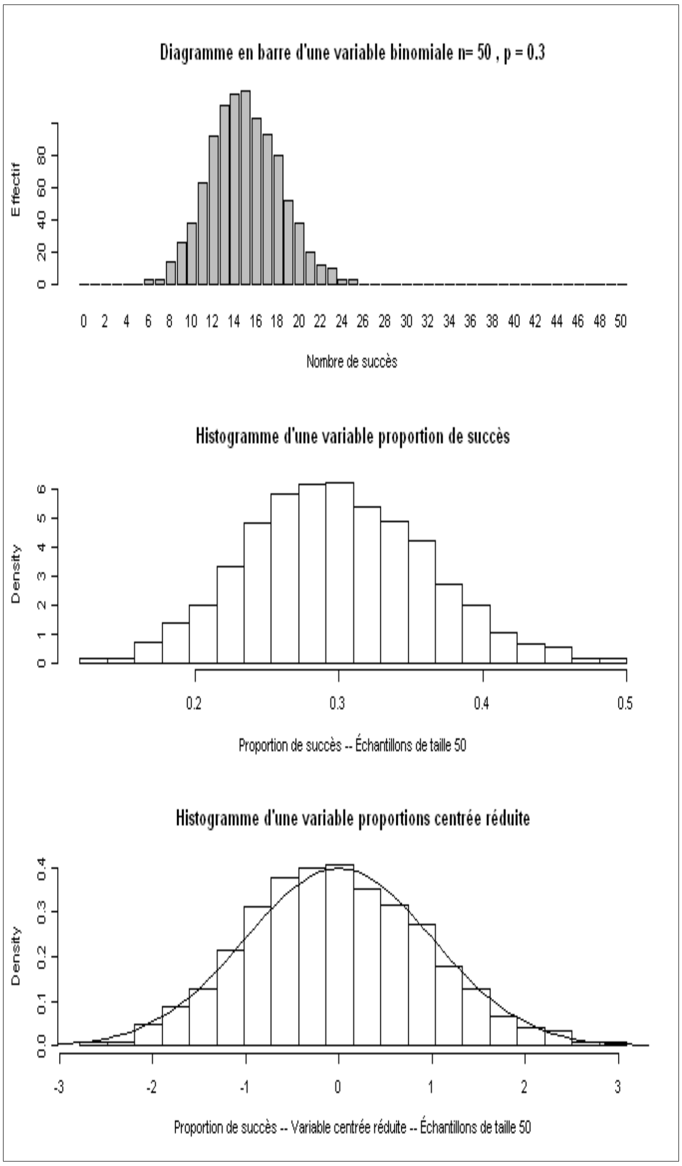
15° Simuler la distribution des proportions tirées d'échantillons aléatoires simples et indépendants

C'est l'illustration du théorème central limite avec une distribution parent binomiale. On peut faire varier la taille de l'échantillon, le nombre de simulations, les paramètres n et p (= r/m dans l'urne) de la distribution binomiale, le nombre de classes pour la construction de l'histogramme. Le théorème de Moivre en est le cas particulier historique dans lequel p = 0,5 et celui de Moivre-Laplace pour p quelconque.

```
# Distribution simulée de variables
binocentral = function(n = 50, r = 30, m = 100, nsim = 1000, nbclass = 20){
  mucpt <- n*r/m ; sigmacpt <- sqrt((n*r/m)*(1-r/m))
  muprop <- r/m ; sigmaprop <- sqrt((r/m)*(1-r/m)/n)
  xgauss <- seq(-4, 4, .1) ; ygauss <- dnorm(xgauss)
  urne <- 1:m
  vectcpt <- vector(length = nsim)
  vectprop <- vector(length = nsim)
  vectbinocr <- vector(length = nsim)
  vectpcr <- vector(length = nsim)
  tabloeffec <- rep(0,n+1)
  names(tabloeffec) <- 0:n
  for (i in 1:nsim){
     vectepreuv <- sample(urne,n,replace=T)
     vectcpt[i] <- sum(vectepreuv <= r)
     vectprop[i] <- vectcpt[i]/n
     vectbinocr[i] <- (vectcpt[i] - mucpt)/sigmacpt
    vectpcr[i] <- (vectcpt[i]/n - muprop)/sigmaprop
                  }
  effec <- table(vectcpt)
  tabloeffec[as.numeric(names(effec))+1] <- effec
  infcpt <- floor(min(vectcpt)*100)/100 ; supcpt <- ceiling(max(vectcpt)*100)/100
  etendcpt <- (supcpt - infcpt)/nbclass
  infprop <- floor(min(vectprop)*100)/100 ; supprop <- ceiling(max(vectprop)*100)/100
  etendprop <- (supprop - infprop)/nbclass
  infbinocr <- floor(min(vectbinocr)*100)/100 ; supbinocr <- ceiling(max(vectbinocr)*100)/100
  etendbinocr <- (supbinocr - infbinocr)/nbclass
  infpcr <- floor(min(vectpcr)*100)/100 ; suppcr <- ceiling(max(vectpcr)*100)/100
  etendpcr <- (suppcr - infpcr)/nbclass
#************************************** Affichage des résultats ***************************
  cat("Moyenne des comptages =", mean(vectcpt), "-- Variance des comptages =",
      var(vectcpt), "\n")
  cat("Moyenne des comptages centred =", mean(vectbinocr), "-- Variance des comptages centred =",
var(vectbinocr), "\n")
  cat("Moyenne des proportions =", mean(vectprop), "-- Variance des proportions =", var(vectprop), "\n")
  cat("Moyenne des proportions centred =", mean(vectpcr), "-- Variance des proportions centred =",
var(vectpcr), "\n")
graphics.off()
  par(mfrow = c(3,1))
  barcpt <- barplot(tabloeffec, xlab = "Nombre de succès",
     ylab = "Effectif", main = paste("Diagramme en barre d'une variable binomiale n=", n, ", p =", r/m))
  histcpt <- hist(vectcpt, freq = F, breaks=seq(infcpt,supcpt,etendcpt), right=F,
     xlab = paste("Nombre de succès -- Échantillons de taille", n),
     main = "Histogramme d'une variable binomiale")
  histbinocr <- hist(vectbinocr, freq = F, breaks=seq(infbinocr, supbinocr, etendbinocr), right=F,
     xlab = paste("Nombre de succès -- Variable centrée réduite -- Échantillons de taille", n),
     main = "Histogramme d'une variable binomiale centrée réduite")
   lines(xgauss, ygauss)
  windows()
  par(mfrow = c(3,1))
  barplot(tabloeffec, xlab = "Nombre de succès",
     ylab = "Effectif", main = paste("Diagramme en barre d'une variable binomiale n=", n, ", p =", r/m))
  histprop <- hist(vectprop, freq = F, breaks=seq(infprop, supprop, etendprop), right=F,
     xlab = paste("Proportion de succès -- Échantillons de taille", n),
     main = "Histogramme d'une variable proportion de succès")
  histpcr <- hist(vectpcr, freq = F, breaks=seq(infpcr, suppcr, etendpcr), right=F,
     xlab = paste("Proportion de succès -- Variable centrée réduite -- Échantillons de taille", n),
     main = "Histogramme d'une variable proportions centrée réduite")
   lines(xgauss, ygauss)
}
```

Simuler la distribution des proportions tirées d'échantillons aléatoires simples et indépendants (suite)





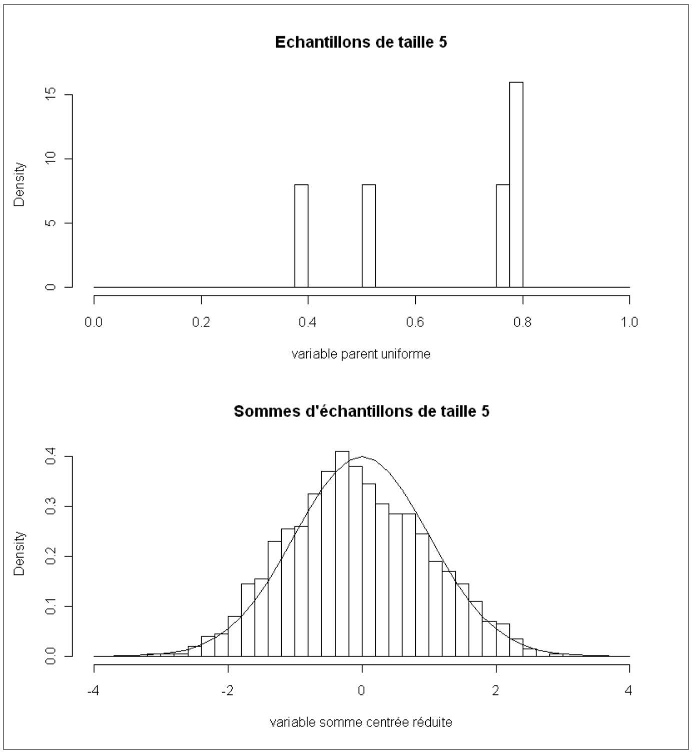

```
binocentral()
Moyenne des comptages = 14.906 -- Variance des
comptages = 10.87404
Moyenne des comptages centred = -0.02900903 --
Variance des comptages centred = 1.035623
Moyenne des proportions = 0.29812 -- Variance
des proportions = 0.004349615
Moyenne des proportions centred = -0.02900903 --
Variance des proportions centred = 1.035623
```

16° Simuler la distribution de la somme de n variables aléatoires uniformes continues sur [a ; b], indépendantes

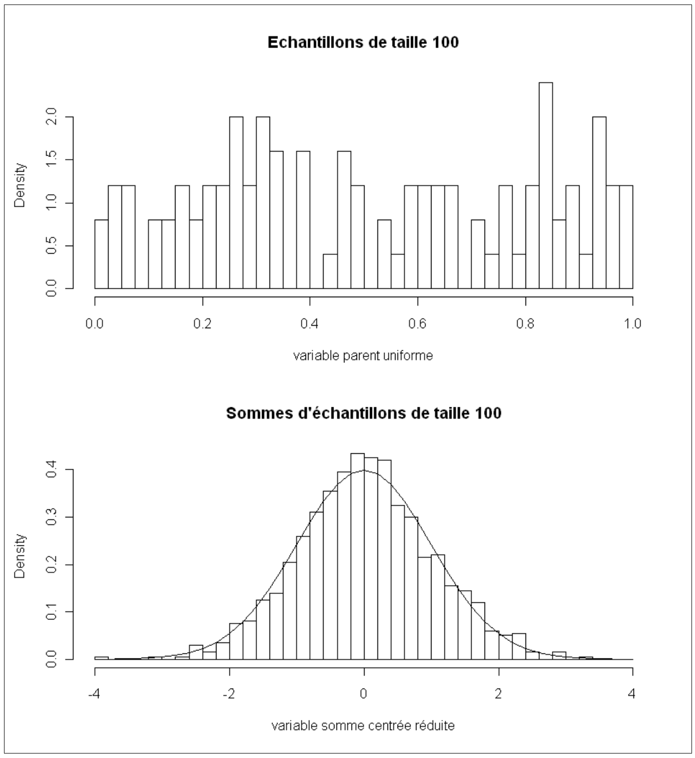
C'est une illustration dans le cas particulier d'une distribution de la variable parent uniforme [0 ; 1] qui est une distributions symétrique. En faisant passer le # de la ligne 6 à la ligne 7, peut très facilement illustrer le cas d'un variable parent à distribution dissymétrique comme la distribution exponentielle

```
centralimite1 = function(a = 0, b = 1, r = 1, n = 4, nsim = 1000, nbclass = 40){
 sommes <- rep(NA, nsim)
 mu <- n * (a + b) / 2 ; sigma <- sqrt(n *(b - a)^2 / 12)
 for(i in 1:nsim){
   # expe <- rexp(n,rate = r)
   expe <- runif(n, min = a, max = b)
   sommes[i] <- sum(expe)
                }
   somexpe <- (sommes - mu) / sigma
 x <- seq(-4, 4, 8 / nbclass) ; ygaus <- dnorm(x)
#***********Affichage des graphiques et des résultats**************
 cat("Moyenne des sommes =",mean(somexpe), "-- Variance des sommes =",var(somexpe),"\n")
 par(mfrow = c(2,1))
 histexpe <- hist((expe), breaks=seq(a, b, (b - a) / nbclass), right=F, freq = F,
    xlab = "variable parent uniforme", main = paste("Echantillons de taille", n))
 histsom <- hist((somexpe), breaks=seq(-4, 4, 8 / nbclass), right=F, freq = F,
    xlab = "variable somme centrée réduite", main = paste("Sommes d'échantillons de taille", n))
    points(x, ygaus, type = "l")
}
```

```
centralimite1(n = 5)
```



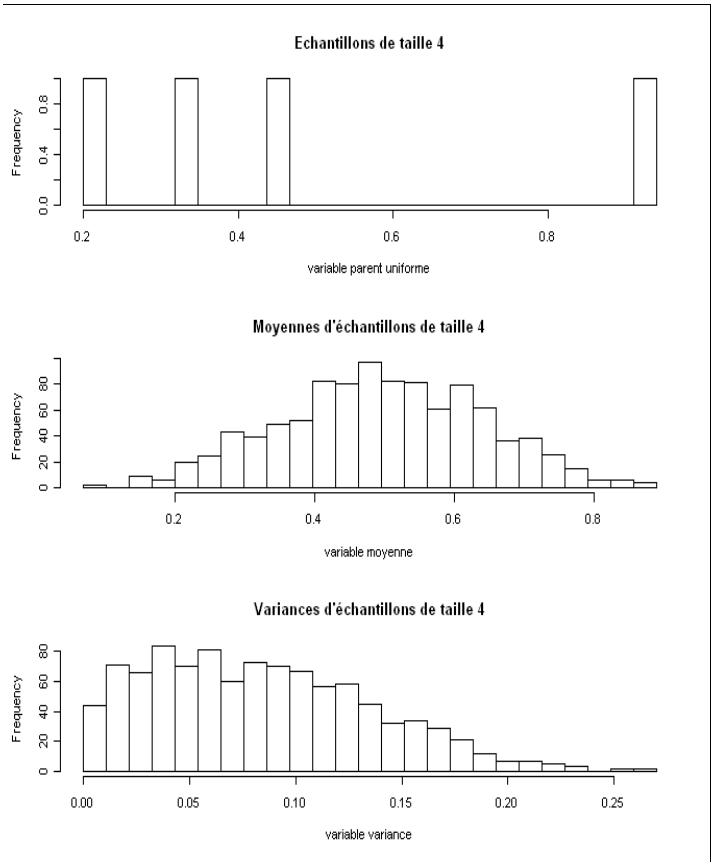

```
centralimite1(n = 100)
```

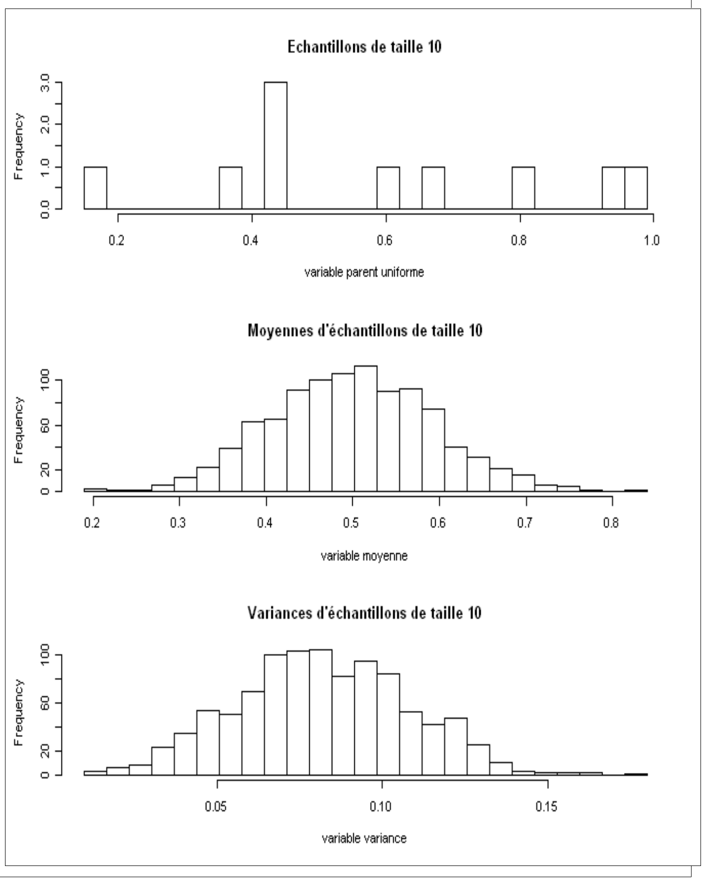


17° Simuler la distribution des moyennes et des variances d'échantillons aléatoires simples et indépendants

C'est une illustration du théorème central limite avec une distribution parent uniforme. On peut faire varier la taille de l'échantillon, le nombre de simulations, les bornes de l'intervalle, le nombre de classes pour la construction de l'histogramme. On peut aussi choisir une population parent exponentielle, de paramètre r.

```
# Distribution simulée de variables
centralimite0 = function(a = 0, b = 1, r = 1, n = 4, nsim = 1000, nbclass = 25){
 moyennes <- vector(length = nsim)
 variances <- vector(length = nsim)
   for(i in 1:nsim){
     expe <- runif(n, min = a, max = b)
     # expe <- rexp(n,rate = r)
     moyennes[i] <- mean(expe)
     variances[i] <- var(expe)
                  }
   par(mfrow = c(3,1))
   infexpe <- floor(min(expe)*100)/100 ; supexpe <- ceiling(max(expe)*100)/100
   etendexpe <- (supexpe - infexpe)/nbclass
   infmoy <- floor(min(moyennes)*100)/100 ; supmoy <- ceiling(max(moyennes)*100)/100
   etendmoy <- (supmoy - infmoy)/nbclass
   infvar <- floor(min(variances)*100)/100 ; supvar <- ceiling(max(variances)*100)/100
   etendvar <- (supvar - infvar)/nbclass
#************************************** Affichage des résultats *********************************
   # cat(etendmoy, etendvar, "\n")
   cat("Moyenne des moyennes =",mean(moyennes), "-- Variance des moyennes =",var(moyennes),
      "\n Moyenne des variances =",mean(variances),"-- Variances des variances =", var(variances),"\n" )
   histexpe <- hist((expe),breaks=seq(infexpe,supexpe,etendexpe),right=F,
      xlab = "variable parent uniforme", main = paste("Echantillons de taille", n))
   histmoy <- hist((moyennes),breaks=seq(infmoy,supmoy,etendmoy),right=F,
      xlab = "variable moyenne", main = paste("Moyennes d'échantillons de taille", n))
   histvar <- hist((variances),breaks=seq(infvar,supvar,etendvar),right=F,
      xlab = "variable variance", main = paste("Variances d'échantillons de taille", n))
   # print(histmoy$breaks);print(histmoy$counts)
   # print(histvar$breaks);print(histvar$counts)
}
```

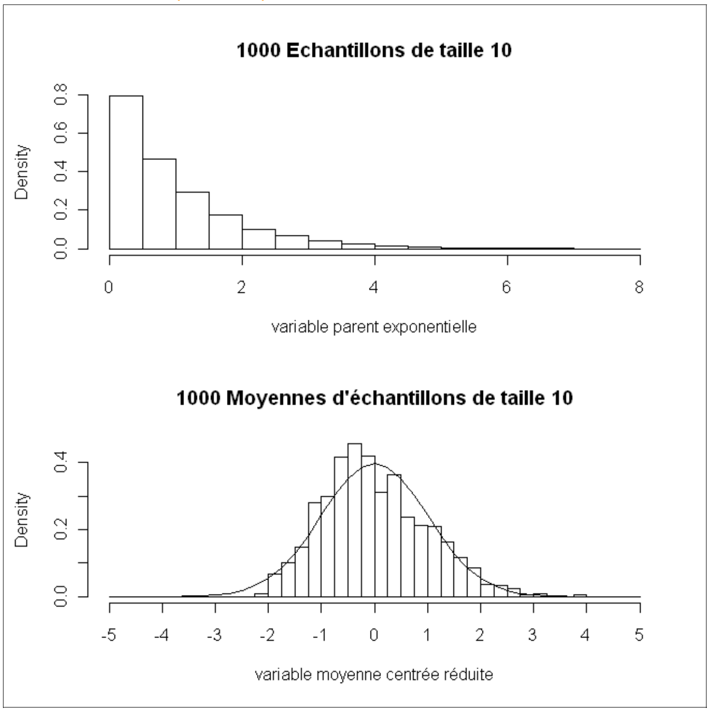




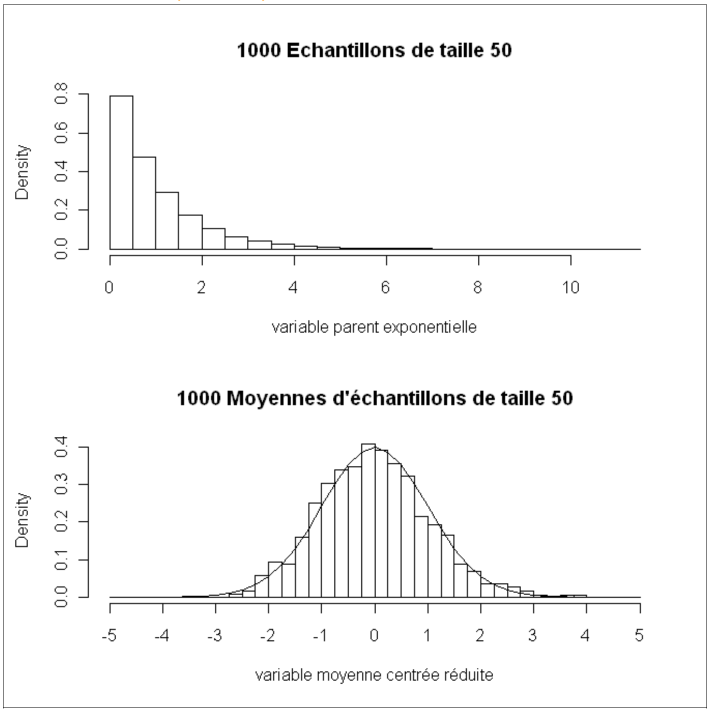
18° Simuler la distribution de la moyenne de n variables aléatoires exponentielles indépendantes

```
#--- MOYENNE DE n variables exponetielles indépendantes de paramètre r ----
# Attention : pour n = 10, il peut arriver que le deuxième histogramme ne se fasse pas
# je n'ai pas fait de gestion fine des bornes des domaines graphiques ..
centralimite3 = function(n = 40, r = 1, nsim = 1000, nbclass = 40){
 expon <- NA
 moyennes <- rep(NA, nsim)
 mu <- 1 / r ; sigma <- 1 / (r * sqrt(n))
 for(i in 1:nsim){
   expe <- rexp(n, rate = r)
   expon <- c(expon, expe)
   moyennes[i] <- mean(expe)
                 }
 moyecr <- (moyennes - mu) / sigma
 x <- seq(-5, 5, 10 / nbclass) ; ygaus <- dnorm(x)
#***********Affichage des graphiques et des résultats**************
 par(mfrow = c(2, 1))
 histexpe <- hist((expon), right = F, freq = F, xlab = "variable parent exponentielle",
    main = paste(nsim, "Echantillons de taille", n))
 histmoye <- hist(moyecr, right = F, freq = F, breaks = seq(-5, 5, 10 / nbclass),
    xlab = "variable moyenne centrée réduite", xaxp = c(-5, 5, 10),
    main = paste(nsim, "Moyennes d'échantillons de taille", n))
  points(x, ygaus, type = "l")
}
```

```
centralimite3(n = 10)
```




```
centralimite3(n = 50)
```



19° Simuler quelques exercices d'annales de bac

1 / 2011-S-Avril-Pondichéry : fléchettes, Épreuves successives, répétées, à trois issues

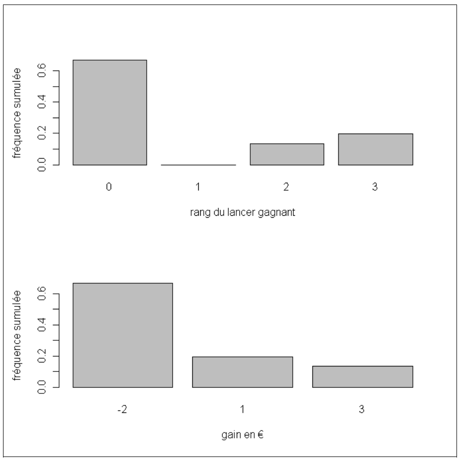
Partie A : Les 3 lancers de chaque partie sont effectués sans interruption.

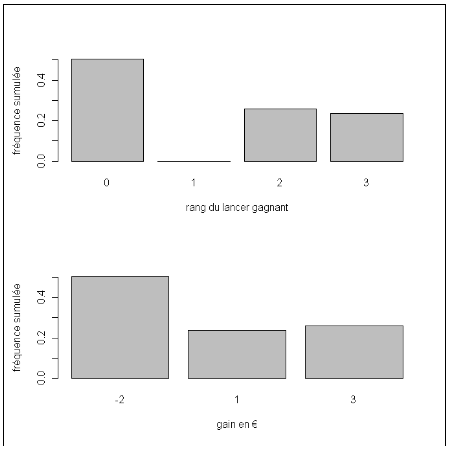
 # Distribution simulée de variables
 # et estimation de probabilités.

```
flechesD = function(cible = c(0, 3, 5), proba = c(3/6, 2/6, 1/6),
                    n = 3, gagne = 8, nbsim = 2000){
 vectgagne <- vector(length = nbsim)
 tablogagne <- rep(0, n + 1)
 names(tablogagne) <- 0:n
 for(i in 1:nbsim){
    partie <- sample(cible, n, proba, replace=T)
    if(sum(partie) < gagne) {vectgagne[i] <- 0} else {
      if(partie[1] >= gagne) {vectgagne[i] <- 1} else {
        if(sum(partie[1:2]) >= gagne) {vectgagne[i] <- 2} else {
          vectgagne[i] <- 3                                      }
                                                        }
                                                       }
                 }
 tablogagne[as.numeric(names(table(vectgagne))) +1] <- table(vectgagne)
 AuMoins1sur6 <- 1 - (tablogagne[1]/nbsim)^6
 distribX <- c(tablogagne[1], tablogagne[4], tablogagne[3])/nbsim
 names(distribX) <- c(-2, 1, 3)
 esperanceX <- sum(distribX * as.numeric(names(distribX)))
#***************** Affichage des résultats *******************
 cat("\n Distribution simulée du rang du lancers gagnant sur 3
lancers \n")
 print(tablogagne/nbsim)
 cat("\n Estimation de la proba de perdre après 3 lancers =",
    tablogagne[1]/nbsim, "\n")
 cat(" Estimation de la proba de gagner en 1 lancer =",
     tablogagne[2]/nbsim, "\n")
 cat(" Estimation de la proba de gagner en 2 lancers =",
     tablogagne[3]/nbsim, "\n")
 cat(" Estimation de la proba de gagner en 3 lancers =",
     tablogagne[4]/nbsim, "\n")
 cat(" Estimation de la proba de gagner =",
     sum(tablogagne[2:4])/nbsim, "\n")
 cat("Estimation de la proba degagner au moins une partie sur 6 =",
     AuMoins1sur6, "\n")
 cat("\n Distribution sumulée de la variable aléatoires X :\n")
 print(distribX)
 cat(" Espérance de X =",esperanceX, "€ \n")
 par(mfrow = c(2, 1))
 barplot(tablogagne/nbsim, xlab = "rang du lancer gagnant",
         ylab = "fréquence sumulée")
 barplot(distribX, xlab = "gain en €", ylab = "fréquence sumulée")
}
```

```
flechesD(gagne = 6)
 Distribution simulée du rang du lancers gagnant sur 3 lancers
    0     1     2     3
0.504 0.000 0.259 0.237

 Estimation de la proba de perdre après 3 lancers = 0.504
 Estimation de la proba de gagner en 1 lancer = 0
 Estimation de la proba de gagner en 2 lancers = 0.259
 Estimation de la proba de gagner en 3 lancers = 0.237
 Estimation de la proba de gagner = 0.496
Estimation de la proba degagner au moins une partie sur 6 = 0.9836098

 Distribution sumulée de la variable aléatoires X :
   -2     1     3
0.504 0.237 0.259
 Espérance de X = 0.006 €
```

```
flechesD()

 Distribution simulée du rang du
lancers gagnant sur 3 lancers
     0      1      2      3
0.6685 0.0000 0.1350 0.1965

 Estimation de la proba de perdre
après 3 lancers = 0.6685
 Estimation de la proba de gagner en 1
lancer = 0
 Estimation de la proba de gagner en 2
lancers = 0.135
 Estimation de la proba de gagner en 3
lancers = 0.1965
 Estimation de la proba de gagner =
0.3315
Estimation de la proba degagner au
moins une partie sur 6 = 0.91075

 Distribution sumulée de la variable
aléatoires X :
    -2      1      3
0.6685 0.1965 0.1350
 Espérance de X = -0.7355 €
```

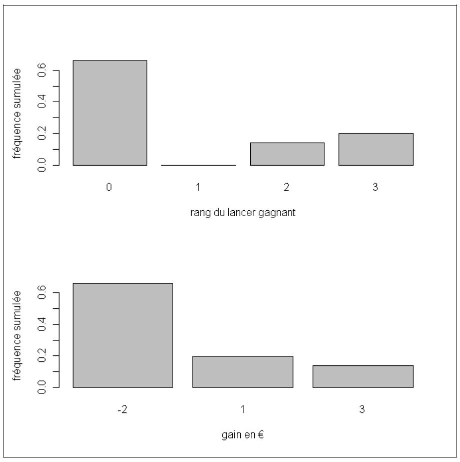


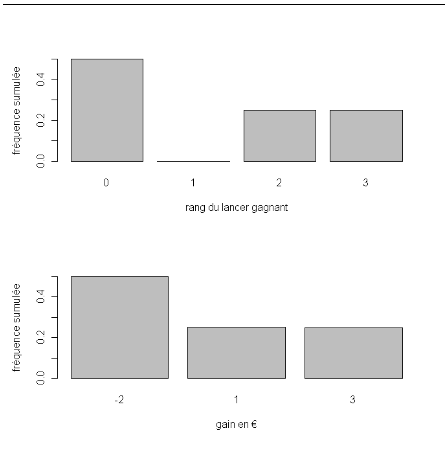


```
Partie B : On arrête les lancers dès que la partie est gagnée.
# Distribution simulée de variables
# et estimation de probabilités.
flechesD = function(cible = c(0, 3, 5), proba = c(3/6, 2/6, 1/6),
                    n = 3, gagne = 8, nbsim = 2000){
 vectgagne <- vector(length = nbsim)
 tablogagne <- rep(0, n + 1)
 names(tablogagne) <- 0:n
 for(i in 1:nbsim){
   nblancers <- 0
   CumParties <- 0
   while(CumParties < gagne & nblancers < n){
      partie <- sample(cible, 1, proba, replace = T)
      CumParties <- CumParties + partie
      nblancers <- nblancers + 1              }
   if(nblancers == n & CumParties < gagne) {vectgagne[i] <- 0} else {
      vectgagne[i] <- nblancers                                  }
            }
 tablogagne[as.numeric(names(table(vectgagne))) +1] <- table(vectgagne)
 AuMoins1sur6 <- 1 - (tablogagne[1]/nbsim)^6
 distribX <- c(tablogagne[1], tablogagne[4], tablogagne[3])/nbsim
 names(distribX) <- c(-2, 1, 3)
 esperanceX <- sum(distribX * as.numeric(names(distribX)))
#****************** Affichage des résultats *********************
 cat("\n Distribution simulée du rang du lancers gagnant sur 3
lancers \n")
 print(tablogagne/nbsim)
 cat("\n Estimation de la proba de perdre après 3 lancers =",
    tablogagne[1]/nbsim, "\n")
 cat(" Estimation de la proba de gagner en 1 lancer =",
     tablogagne[2]/nbsim, "\n")
 cat(" Estimation de la proba de gagner en 2 lancers =",
     tablogagne[3]/nbsim, "\n")
 cat(" Estimation de la proba de gagner en 3 lancers =",
     tablogagne[4]/nbsim, "\n")
 cat(" Estimation de la proba de gagner =",
     sum(tablogagne[2:4])/nbsim, "\n")
 cat("Estimation de la proba degagner au moins une partie sur 6 =",
     AuMoins1sur6, "\n")
 cat("\n Distribution sumulée de la variable aléatoires X :\n")
 print(distribX)
 cat(" Espérance de X =",esperanceX, "€ \n")
 par(mfrow = c(2, 1))
 barplot(tablogagne/nbsim, xlab = "rang du lancer gagnant",
         ylab = "fréquence sumulée")
 barplot(distribX, xlab = "gain en €", ylab = "fréquence sumulée")
}
```

```
flechesD(gagne = 6)
 Distribution simulée du rang du lancers gagnant sur 3 lancers
     0      1      2      3 
0.5005 0.0000 0.2490 0.2505 

 Estimation de la proba de perdre après 3 lancers = 0.5005 
 Estimation de la proba de gagner en 1 lancer = 0 
 Estimation de la proba de gagner en 2 lancers = 0.249 
 Estimation de la proba de gagner en 3 lancers = 0.2505 
 Estimation de la proba de gagner = 0.4995 
Estimation de la proba degagner au moins une partie sur 6 = 0.984281 

 Distribution sumulée de la variable aléatoires X :
    -2      1      3 
0.5005 0.2505 0.2490 
 Espérance de X = -0.0035 € 
```

```
flechesD()
 Distribution simulée du rang du
lancers gagnant sur 3 lancers
    0     1     2     3 
0.662 0.000 0.139 0.199 

 Estimation de la proba de perdre
après 3 lancers = 0.662 
 Estimation de la proba de gagner en 1
lancer = 0 
 Estimation de la proba de gagner en 2
lancers = 0.139 
 Estimation de la proba de gagner en 3
lancers = 0.199 
 Estimation de la proba de gagner =
0.338 
Estimation de la proba degagner au
moins une partie sur 6 = 0.9158318 

 Distribution sumulée de la variable
aléatoires X :
   -2     1     3 
0.662 0.199 0.139 
 Espérance de X = -0.708 € 
```

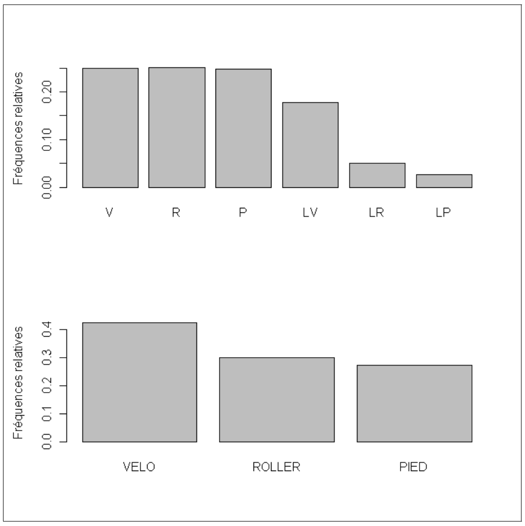




```
# Estimation de la probabilité d'un événement
velorollerp = function(urne = c("V","R","P","L"), urneL = c(.7,.2,.1), nbsim = 20000){
   velo <- 0
   roller <-  0
   pied <- 0
   veloETjetonL <- 0
   for(j in 1:nbsim){
      jeton <- sample(urne, 1, replace=T)
      if(jeton == "V") velo <- velo + 1
      if(jeton == "R") roller <- roller + 1
      if(jeton == "P") pied <- pied + 1
      if(jeton == "L"){
         tirage2 <- sample(c("V","R","P"), 1, prob = urneL, replace = T)
         if(tirage2 == "V") {velo <- velo + 1 ;  veloETjetonL <- veloETjetonL + 1}
         if(tirage2 == "R") roller <- roller + 1
         if(tirage2 == "P") pied <- pied + 1
                   }
                 }
#***************** Affichage des résultats *******************
   cat("Une estimation de la probabilité de velo =",velo/nbsim,"\n")
   cat("Une estimation de la probabilité de roller =",roller/nbsim,"\n")
   cat("Une estimation de la probabilité de pied =",pied/nbsim,"\n")
   cat("Une estimation de la probabilité sachant velo de jeton L =",
        veloETjetonL/velo,"\n")
}
# Estimation de la probabilité d'un événement
sixannees = function(n = 6, p = 1/3, k = 1, nbsim = 5000){
  vecnbannee <- vector(length = nbsim)
  for (j in 1:nbsim) vecnbannee[j] <- sum(runif(n) <= p)
  nbaumoinsk <- sum(vecnbannee >= k)
#***************** Affichage des résultats *******************
  cat("Estimation de la probabilité d'au moins", k,"non cycliste gagnant \n
   au cours des ",n," prochaines années =",nbaumoinsk/nbsim,"\n")
}
```

```
velorollerp()
 Une estimation de la probabilité de velo = 0.42095
 Une estimation de la probabilité de roller = 0.3006
 Une estimation de la probabilité de pied = 0.27845
 Une estimation de la probabilité sachant velo, de jeton L = 0.4154888

sixannees()
 Estimation de la probabilité d'au moins 1 non cycliste gagnant au cours des  6 prochaines années = 0.9114
```

```
# Distribution simulée de variables qualitatives
velorollerd = function(urne = c("V","R","P","L"), urneL =
c(.7,.2,.1), nbsim = 20000){
  distexpe <- vector(length = 6) # ou rep(0,4)
  names(distexpe) <- c("V","R","P","LV","LR","LP")
  for(j in 1:nbsim){
    jeton <- sample(urne, 1, replace=T)
    if(jeton == "V") distexpe[1] <- distexpe[1] + 1
    if(jeton == "R") distexpe[2] <- distexpe[2] + 1
    if(jeton == "P") distexpe[3] <- distexpe[3] + 1
    if(jeton == "L"){
      tirage2 <- sample(c("V","R","P"), 1, prob = urneL, replace = T)
         if(tirage2 == "V") distexpe[4] <- distexpe[4] + 1
         if(tirage2 == "R") distexpe[5] <- distexpe[5] + 1
         if(tirage2 == "P") distexpe[6] <- distexpe[6] + 1
                  }
               }
   distdep <- c((distexpe[1]+distexpe[4]),(distexpe[2]+distexpe[5]),
(distexpe[3]+distexpe[6]))
   names(distdep) <- c("VELO","ROLLER","PIED")
#***************** Affichage des résultats *******************
   cat("Une estimation de la probabilité de velo =",
(distexpe[1]+distexpe[4])/nbsim,"\n")
   cat("Une estimation de la probabilité de roller =",
(distexpe[2]+distexpe[5])/nbsim,"\n")
   cat("Une estimation de la probabilité de pied =",
(distexpe[3]+distexpe[6])/nbsim,"\n")
   cat("Une estimation de la probabilité sachant velo de jeton L =",
distexpe[4]/(distexpe[1]+distexpe[4]),"\n")
   par(mfrow = c(2,1))
   barplot(distexpe/nbsim, ylab = "Fréquences relatives")
   barplot(distdep/nbsim, ylab = "Fréquences relatives")
}
```

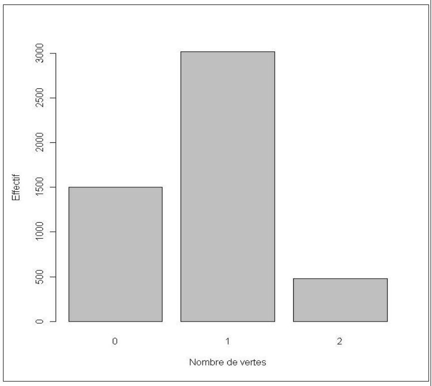



```
velorollerd()
Une estimation de la probabilité de velo = 0.4264
Une estimation de la probabilité de roller = 0.30075
Une estimation de la probabilité de pied = 0.27285
Une estimation de la probabilité sachant velo de jeton L = 0.4152205
```

3 / 2010-S-Novembre-Nouvelle-Calédonie : urne boules, tirages avec et sans remise, probabilités conditionnelles

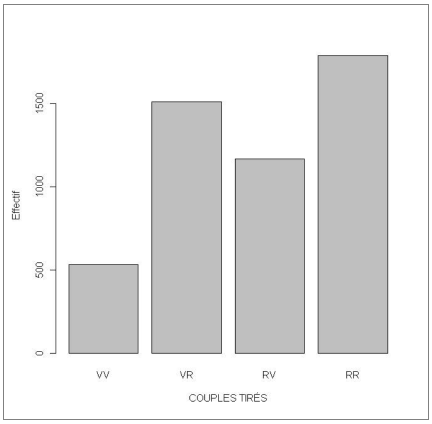
```
# Distribution simulée d'une variable quantitative
simurnsansd = function(n = 2, r = 3, v = 2, nbsim = 5000){
  urne <- rep(c(0, 1),c(r, v))
  DistSim <- vector(length = nbsim)
  tabloeffec <- rep(0,n+1)
  names(tabloeffec) <- 0:n
  for (i in 1:nbsim){
        vectepreuv <- sample(urne, n, replace=F)
        vectexpe <- sum(vectepreuv)
        DistSim[i] <- vectexpe
                   }
  effec <- table(DistSim)
  tabloeffec[as.numeric(names(effec))+1] <- effec
#***************** Affichage des résultats *******************
  print ("tabloeffec/nbsim")
  print (tabloeffec/nbsim)
  barplot(tabloeffec, xlab = "Nombre de vertes", ylab = "Effectif")
  cat("Estimation de la probabilité de 0 verte =",tabloeffec[1]/nbsim,"\n")
  cat("Moyenne de la distribution simulée du nombre de vertes =",
      mean(DistSim), "\n")
  cat("Estimation de la probabilité de 2 boules de la même couleur =",
     (tabloeffec[1] + tabloeffec[3])/nbsim, "\n")
}
```

```
simurnsansd()
[1] "tabloeffec/nbsim"
     0      1      2
0.3006 0.6034 0.0960
Estimation de la probabilité de 0
verte = 0.3006
Moyenne de la distribution
simulée du nombre de vertes =
0.7954
Estimation de la probabilité de 2
boules de la même couleur =
0.3966
```




```
# Distribution simulée d'une variable qualitative
simavecsansd = function(n = 2, r = 3, v = 2, nbsim = 5000){
  urne1 <- rep(c(0, 1),c(r, v))
  urne2 <- rep(c(0, 1),c(r, v-1))
  tabloeffec <- rep(0,4)
  names(tabloeffec) <- c("VV", "VR", "RV", "RR")
  for (i in 1:nbsim){
       tirage1 <- sample(urne1,1)
       if(tirage1 == 0){tirage2 <- sample(urne1,1)} else {
         tirage2 <- sample(urne2,1)                       }
       if((tirage1 == 1)&(tirage2 == 1)) tabloeffec[1] <- tabloeffec[1]+1
       if((tirage1 == 1)&(tirage2 == 0)) tabloeffec[2] <- tabloeffec[2]+1
       if((tirage1 == 0)&(tirage2 == 1)) tabloeffec[3] <- tabloeffec[3]+1
       if((tirage1 == 0)&(tirage2 == 0)) tabloeffec[4] <- tabloeffec[4]+1
                   }
#***************** Affichage des résultats *******************
  print ("tabloeffec/nbsim")
  print (tabloeffec/nbsim)
  barplot(tabloeffec, xlab = "COUPLES TIRÉS", ylab = "Effectif")
  cat("Estimation de la probabilité de seule la première est verte =",
        tabloeffec[2]/nbsim,"\n")
  cat("Estimation de la probabilité de une seule verte =",
       (tabloeffec[2] + tabloeffec[3])/nbsim,"\n")
  cat("Estimation de la probabilité que, sachant qu'une seule verte a été
       tirée, ce soit la première =",
       tabloeffec[2]/(tabloeffec[2] + tabloeffec[3]),"\n")
}
```

```
simavecsansd()
[1] "tabloeffec/nbsim"
    VV     VR     RV     RR
0.1068 0.3022 0.2334 0.3576
Estimation de la probabilité de
seule la première est verte =
0.3022
Estimation de la probabilité de
une seule verte = 0.5356
Estimation de la probabilité que,
sachant qu'une seule verte a été
tirée, ce soit la première =
0.564227
```



4 / 2010-S-Septembre-Antilles : Bovins malades et test dépistage, probabilités conditionnelles, loi binomiale

<pre># Estimation de la probabilité d'un événement 	
testbovmal = function(pmalpop = .01, ptposmal = .85, ptpossain = .05, nbsim = 5000){
  simpop <- runif(nbsim) # population simulée de bovins
  nbmal <- sum(simpop <= pmalpop) # effectif bovins malades de M dans cette population simulée
  nbsain <- nbsim - nbmal # effectif bovins non malades de M dans cette population simulée
  simpopmal <- runif(nbmal) # population simulée de bovins malades dans simpop
  nbposmal <- sum(simpopmal <= ptposmal) # effectif de tests positifs parmi les malades
  simpopsain <- runif(nbsain) # population simulée de bovins sains dans simpop
  nbpossain <- sum(simpopsain <= ptpossain) # effectif de tests positifs parmi les non malades de M
#***************** Affichage des résultats *******************
  cat("Proba estimée de malade et test positif =",nbposmal/nbsim,"\n")
  cat("Proba estimée de test positif =",(nbposmal+nbpossain)/nbsim,"\n")
  cat("Proba estimée de bovin malade sachant test positif =",nbposmal/(nbposmal+nbpossain),"\n")
}</pre>
	<pre>testbovmal()
	Proba estimée de malade et test positif = 0.0084
	Proba estimée de test positif = 0.0584
	Proba estimée de bovin malade sachant test positif = 0.1438356</pre>

<pre># Estimation de la probabilité d'un événement 	
cinqbovins = function(n = 5, p = .058, k = 1, nbsim = 5000){
  vecnbmal <- vector(length = nbsim)
  for (j in 1:nbsim) vecnbmal[j] <- sum(runif(n) <= p)
  nbaumoinsk <- sum(vecnbmal >= k)
#***************** Affichage des résultats *******************
  cat("Estimation de la probabilité d'au moins", k,"malades parmi les",n,"=",nbaumoinsk/nbsim,"\n")
}</pre>
	<pre>cinqbovins()
	Estimation de la probabilité d'au moins 1 malades parmi les 5 = 0.253</pre>

C – CALCULER DES PROBABILITÉS PAR LES MODÈLES MATHÉMATIQUES

1° Dénombrer les sommes obtenues avec plusieurs dés

Certains exercices de probabilité font intervenir les sommes des valeurs des faces obtenues en lançant plusieurs dés. Il s'agit de dénombrer toutes les sommes possibles, sommes utiles dans les calculs de probabilités. La fonction table() renvoie un tableau avec la valeur des sommes et leurs effectifs.

Distribution des sommes obtenues avec 2 dés

```
a <- matrix(0,6,6)
for(i in 1:6){
    for(j in 1:6) a[i,j] <- i + j
             }
table(a)
a
 2  3  4  5  6  7  8  9 10 11 12
 1  2  3  4  5  6  5  4  3  2  1
sum(table(a))
[1] 36
```

Distribution des sommes obtenues avec 3 dés

```
a<- array(0,dim=c(6,6,6))
for(i in 1:6){
    for(j in 1:6){
       for(k in 1:6) a[i,j,k] <- i + j + k
                 }
             }
table(a)
a
 3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 1  3  6 10 15 21 25 27 27 25 21 15 10  6  3  1
sum(table(a))
[1] 216
```

Distribution des sommes obtenues avec 4 dés

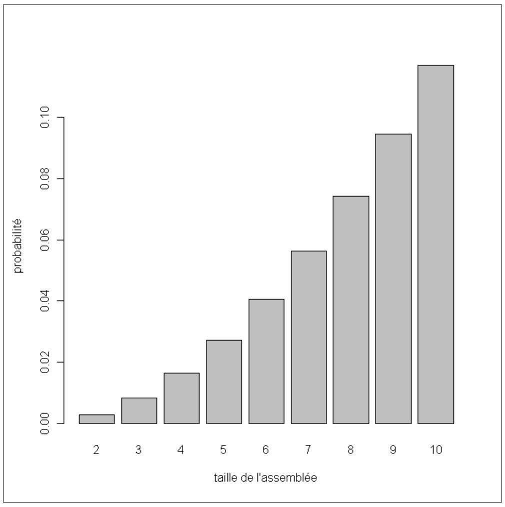
```
a <- array(0,dim=c(6,6,6,6))
for(i in 1:6){
 for(j in 1:6){
   for(k in 1:6){
     for(l in 1:6) a[i,j,k,l] <- i + j + k +l
                }
              }
            }
table(a)
a
  4   5   6   7   8   9  10  11  12  13  14  15
  1   4  10  20  35  56  80 104 125 140 146 140

 16  17  18  19  20  21  22  23  24
125 104  80  56  35  20  10   4   1
sum(table(a))
[1] 1296
```

2° Le problème France-Irlande par le XV de France (problème des anniversaires)

10 supporteurs sont réunis pour fêter la victoire du XV de France contre l'Irlande. Quelle est la probabilité qu'au moins deux d'entre-eux aient le même jour anniversaire? Même question avec 20, 30 puis 40 supporteurs.

```
# Probabilité ponctuelle et graphique en fonction de n
panniv = function(n = 20, d = "F"){
 if(d == "F") {
  proba <- 1
  if(n == 0 | n == 1) {proba <- 1} else            {
   for(i in 1:(n - 1)) proba <- proba * (365 - i)/365}
   paumoins2 <- 1 - proba
    cat(" La probabilité qu'au moins deux personnes aient le
même jour \n",
     " anniversaire, dans une assemblée de",n,"personnes est de
:",paumoins2,"\n")
              } else {
  vectdist <- vector(length = n - 1)
  names(vectdist) <- 2:n
  for(j in 2:n) {
    proba <- 1
    for(i in 1:(j - 1)) proba <- proba * (365 - i)/365
      paumoins2 <- 1 - proba
      vectdist[j - 1] <- paumoins2
              }
#***************** Affichage des résultats *******************
  cat(" \nLa probabilité qu'au moins deux personnes aient le
même jour \n",
    " anniversaire, dans une assemblée de",n,"personnes est
de :",paumoins2,"\n")
  cat(" \n Graphe de la probabilité d'au moins 2 personnes ...,
en fonction\n",
   "de la taille de l'assemblée :\n")
  # print(vectdist)
  barplot(vectdist, xlab = "taille de l'assemblée", ylab =
"probabilité")
                    }
}
```

```
panniv(n = 10, d = "")
La probabilité qu'au moins deux personnes
aient le même jour anniversaire, dans une
assemblée de 10 personnes est de :
0.1169482
 Graphe de la probabilité d'au moins 2
personnes ..., en fonction de la taille
de l'assemblée :
```



Quel est le nombre minimum de supporteurs à réunir pour que la probabilité qu'au moins deux d'entre-eux aient le même jours anniversaire soit supérieure ou égale à 95%...

```
# calcul d'une valeur seuil de n
tailleanniv = function(probmini = .95){
 paumoins2 <- 0
 n <- 1
 while(paumoins2 < probmini){
  proba <- 1
  n <- n + 1
  for(i in 1:(n - 1)) proba <- proba * (365 - i)/365
  paumoins2 <- 1 - proba     }
#***************** Affichage des résultats ******************
 cat("\nIl faut au moins", n, "personnes dans l'assemblée,
pour \n",
    "qu'il y ait une probabilité d'au moins", probmini,
"qu'il se trouve\n",
    "au moins deux personnes avec le même jour
anniversaire\n")
}
```

```
tailleanniv()
Il faut au moins 47 personnes dans
l'assemblée, pour qu'il y ait une
probabilité d'au moins 0.95 qu'il se
trouve au moins deux personnes avec le
même jour anniversaire

tailleanniv(.99)
Il faut au moins 57 personnes dans
l'assemblée, pour qu'il y ait une
probabilité d'au moins 0.99 qu'il se
trouve au moins deux personnes avec le
même jour anniversaire
```

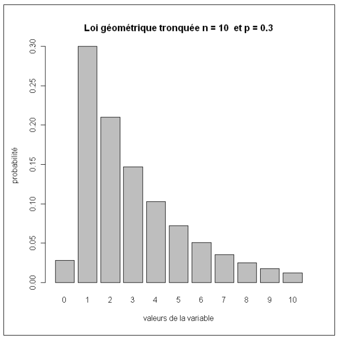
3° Calcul de probabilités de distributions géométriques tronquées

C'est la loi d'une variable aléatoire X prenant pour valeur le rang du premier succès lors de n épreuves indépendantes de probabilité de succès p : $P(X = 0) = (1-p)^n$ et $P(X = k|k = 1...n) = (1-p)^{k-1}\times p$.

Exemple du calcul de $P(A \leq X \leq B)$:

```
geotronk = function(n = 10, p = .3, A = 2, B = 6){
 distribX <- vector(length = n+1)
 names(distribX) <- 0:n
 distribX[1] <- (1 - p)^n
 distribX[2:(n + 1)] <- (1 - p)^((1:n) - 1) * p
 if(A == 0 & B == 0) {geot <- (1 - p)^n} else {
   if(A == 0 & B != 0) {
     geot <- (1 - p)^n + sum((1 - p)^((1:B) - 1) * p)
                  } else {geot <- sum((1 - p)^((A:B) - 1) *
p)}
                                              }
#***************** Affichage des résultats ******************
 cat("Distribution de la géo tronk n =",n, "p =",p, "\n")
 print(distribX)
 cat("\n  Probabilité de (", A,"<= X <=", B,")=", geot, "\n")
 barplot(distribX, xlab = "valeurs de la variable", ylab =
"probabilité",
   main = paste("Loi géométrique tronquée n =",n," et p =",p))
}
```

```
Distribution de la géo tronk n = 10 p = 0.3
         0          1          2          3
0.02824752 0.30000000 0.21000000 0.14700000
         4          5          6          7
 0.10290000 0.07203000 0.05042100 0.03529470
         8          9         10
 0.02470629 0.01729440 0.01210608
Probabilité de ( 2 <= X <= 6 )= 0.582351
```



4° Calculs de probabilités de distributions binomiales

Calcul de $P(A \leq X \leq B)$, X étant une v.a. de distribution binomiale de paramètres n=100 et p=0,52.

Les exemples choisis peuvent servir de base à une réflexion sur les différentes façons de déterminer un intervalle de fluctuation, à partir de l'exemple 1 (Monsieur Z).

$P(42 \leq X \leq 62)$:

```
sum(dbinom(42:62,100,.52))
  [1] 0.9649486*
```

$P(43 \leq X \leq 62)$:

```
sum(dbinom(43:62,100,.52))
  [1] 0.9541022
```

$P(42 \leq X \leq 61)$:

```
sum(dbinom(42:61,100,.52))
  [1] 0.95416
```

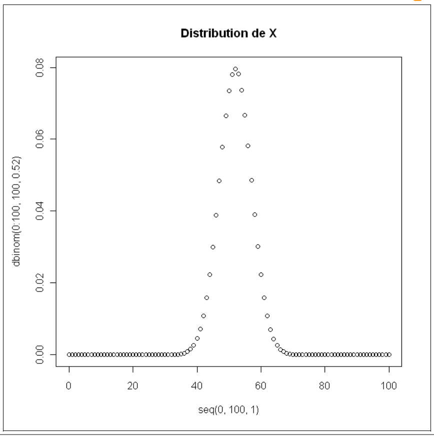
$P(X \leq 41)$ $P(X \leq 42)$ $P(X \leq 43)$:

```
pbinom(41:43,100,.52)
  [1] 0.01772803 0.02857444 0.04442366
```

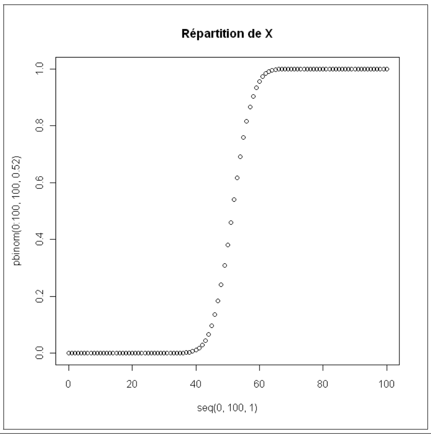
$P(X \leq 61)$ $P(X \leq 62)$ $P(X \leq 63)$:

```
pbinom(61:63,100,.52)
  [1] 0.9718881 0.9826766 0.9897262
```

```
plot(dbinom(0:100,100,.52)~seq(0,100,1))
```



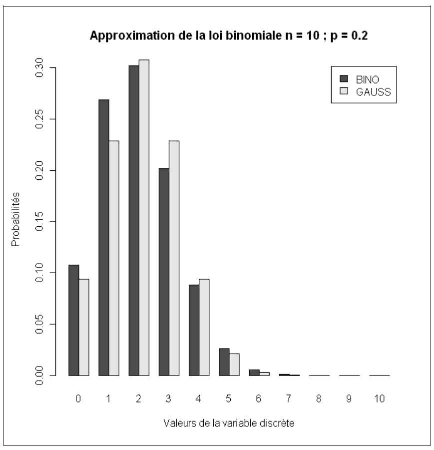

```
plot(pbinom(0:100,100,.52)~seq(0,100,1))
```

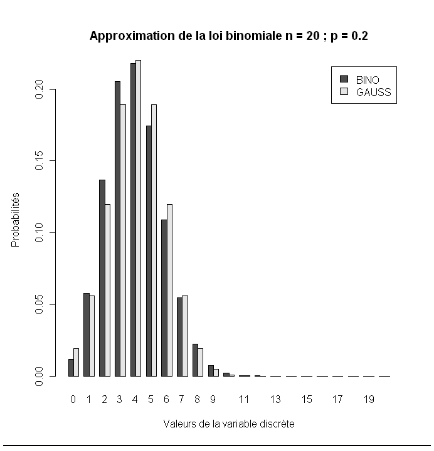


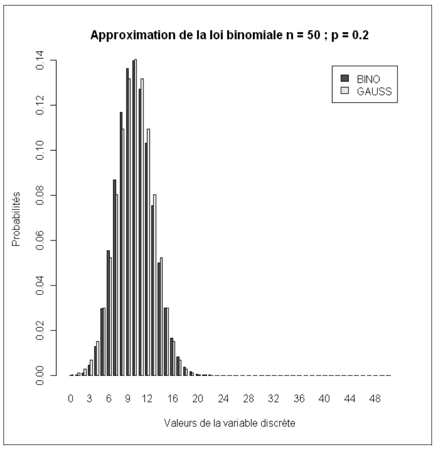
5° Approximations de probabilités binomiales par la loi de Gauss (BinoClocheBernard)

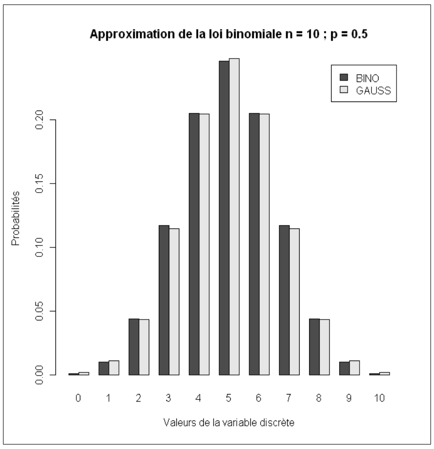
Cette fonction permet de faire varier n et p à volonté, en saisissant les valeurs voulues lors de l'appel de la fonction BinoCloche() : Les valeurs par défaut sont n=10 et p=.2, pour en changer, il suffit de saisir BinoCloche(n=20), ou BinoCloche(p=.5) ou BinoCloche(n=50,p=.8) ou BinoCloche(100,.3). C'est l'illustration du théorème de Moivre-Laplace.

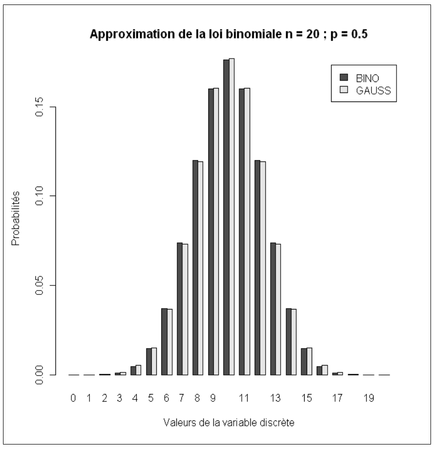
```
BinoCloche = function(n = 10, p=.2){
  mu <- n * p ; sigm <- sqrt(n * p * (1-p))
  probabino <- dbinom(0:n,n,p)
  ak <- (0:(n+1) - mu) / sigm
  etendue <- (ak[2]-ak[1])/2
  bk <- ak - etendue
  probagauss <- rep(NA, n+1)
  for(i in 1:(n+1)) probagauss[i] <- pnorm(bk[i+1]) - pnorm(bk[i])
  cat("Distribution binomiale","\n") ; print(probabino)
  cat("Distribution de Gauss","\n") ; print(probagauss)
  x <- t(as.matrix(data.frame(BINO=probabino,GAUSS=probagauss)))
  barplot(x,beside=T,legend=T, names.arg = 0:n,
        xlab = "Valeurs de la variable discrète", ylab = "Probabilités")
  title(paste("Approximation de la loi binomiale n =",n," ; p =",p))
}
BinoCloche(n = 10, p=.2)
Distribution binomiale
 [1] 0.1073741824 0.2684354560 0.3019898880 0.2013265920 0.0880803840
 [6] 0.0264241152 0.0055050240 0.0007864320 0.0000737280 0.0000040960
[11] 0.0000001024
Distribution de Gauss
 [1] 9.378654e-02 2.284764e-01 3.073672e-01 2.284764e-01 9.378654e-02
 [6] 2.122462e-02 2.641643e-03 1.802910e-04 6.726783e-06 1.368005e-07
[11] 1.512320e-09
```

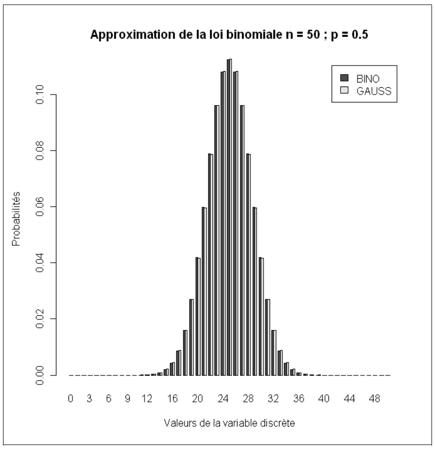







Approximation de la loi binomiale n = 10 ; p = 0.5
BINO
GAUSS
Probabilités
Valeurs de la variable discrète

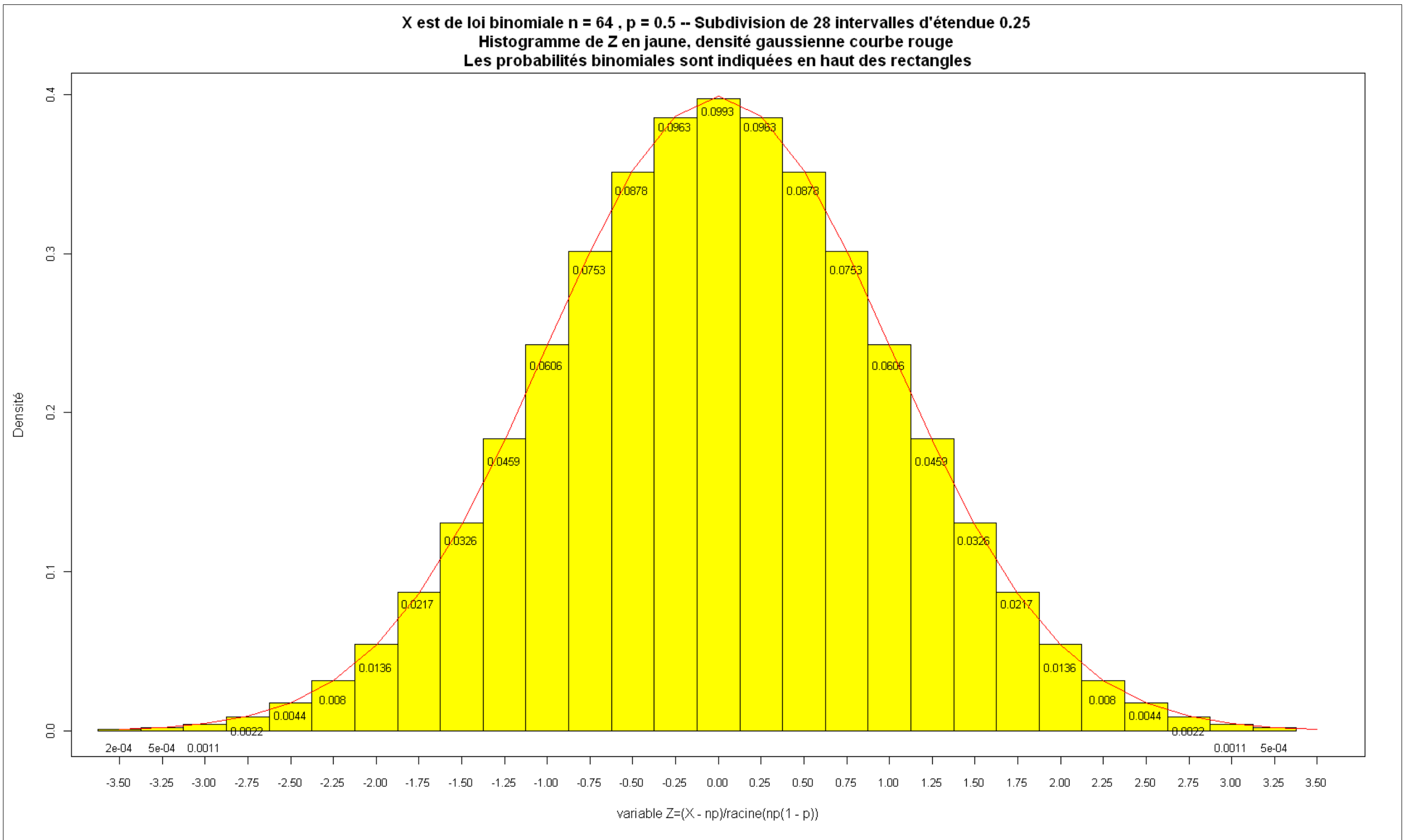

Approximation de la loi binomiale n = 20 ; p = 0.5
BINO
GAUSS
Probabilités
Valeurs de la variable discrète


Approximation de la loi binomiale n = 50 ; p = 0.5
BINO
GAUSS
Probabilités
Valeurs de la variable discrète

6° Convergence de la loi binomiale vers la loi de Gauss version Hubert (BinGaussHub)

```r
bingausshub <- function(n = 64, p = .5, binf = -3.5, bsup = 3.5){
 sigma = sqrt(n * p * (1 - p)) ; nbbornes = (bsup - binf) * sigma + 1
 x <- seq(binf, bsup, length.out = nbbornes)
 probino <- matrix(data = NA, ncol = nbbornes, nrow = 3,
     dimnames = list(c("X", "Probabilités bino", "Densités"), x))
 f <- function(v){dbinom(floor(v + .5), n, p)}
 g <- function(v){sigma * f(n * p + sigma * v)}
 probino[1, ] <- floor(n * p + sigma * x + .5) ; probino[2, ] <- g(x) / sigma
 probino[3, ] <- g(x)
#************** Affichage des résultats et des graphiques ***************
 print("Tableau des valeurs de Z, X, probabilités binomiales, densités")
 print(probino)
 plot(x, probino[3, ], type= "n", lty = 2, xaxp = c(binf, bsup, nbbornes - 1),
      ylim = c(0, max(g(x))), cex.axis = .8,
      xlab = "variable Z=(X - np)/racine(np(1 - p))", ylab = "Densité",
      main = paste("X est de loi binomiale n =", n, ", p =", p,
      "-- Subdivision de", nbbornes - 1, "intervalles d'étendue", 1 / sigma,
      "\nHistogramme de Z en jaune, densité gaussienne courbe rouge",
      "\nLes probabilités binomiales sont indiquées en haut des rectangles"))
  for(i in 1:(nbbornes - 1)){
    polygon(c(x[i] - 1 / (2 * sigma), x[i] - 1 / (2 * sigma),
            x[i + 1] - 1 / (2 * sigma), x[i + 1] - 1 / (2 * sigma)),
    c(0, g(x[i]), g(x[i]), 0), col = "yellow1")
    text(x[i], round(g(x[i]), 2) - .01, labels = round(g(x[i]) / sigma, 4),
         cex = .8)            }
  points(x, dnorm(x, 0, 1), type = "l", col = "red")
#-----colors()[grep("grey",colors())]
}
```

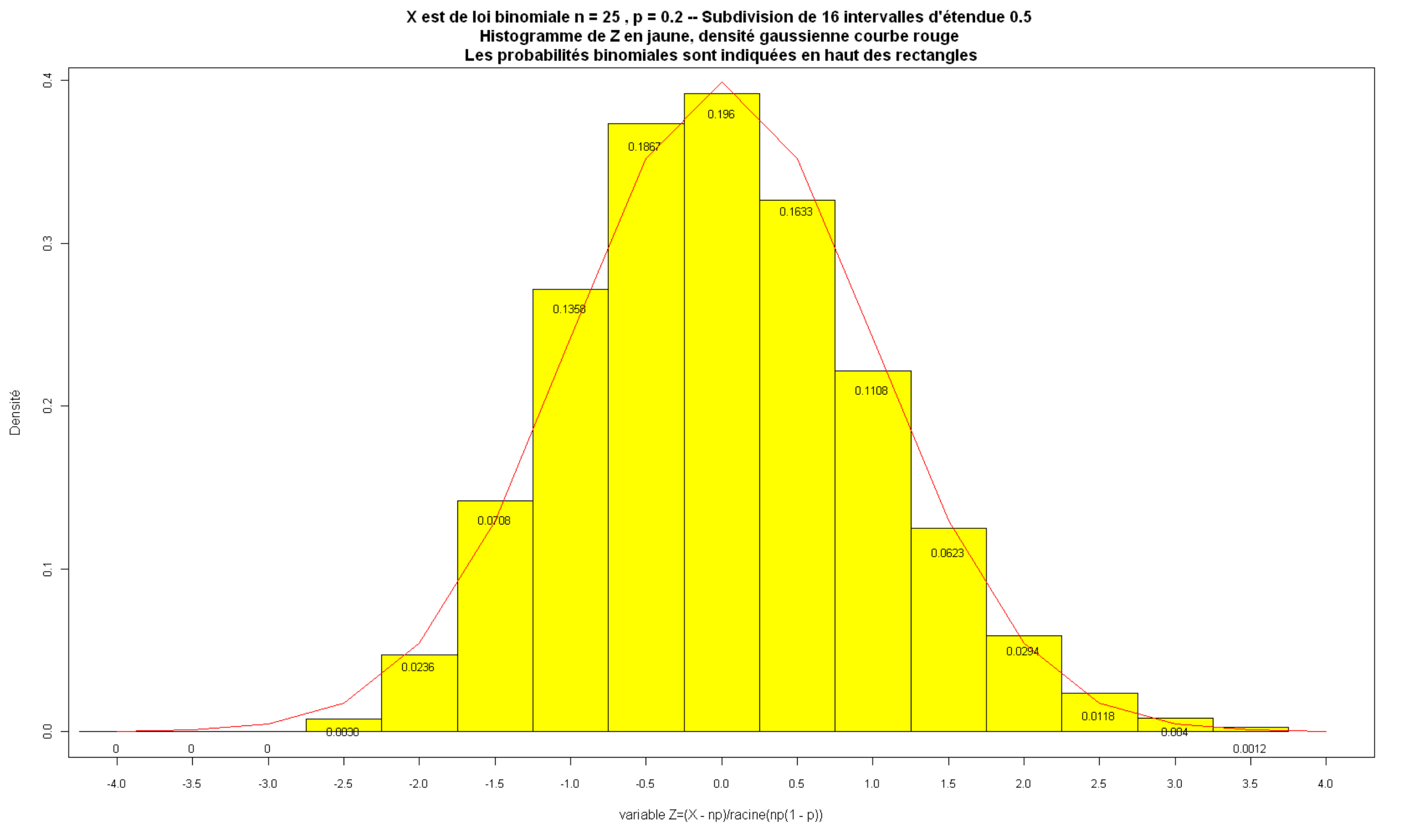
```r
bingausshub(n = 64, p = .5, binf = -3.5, bsup = 3.5)
```




```
[1] "Extrait du tableau des valeurs de Z, X, probabilités binomiales, densités"
                      -2.5        -2.25          -2       -1.75        -1.5       -1.25          -1       -0.75        -0.5       -0.25           0
X              22.000000000 23.000000000 24.00000000 25.00000000 26.00000000 27.00000000 28.00000000 29.00000000 30.00000000 31.00000000 32.00000000
Probabilités    0.004355644  0.007953785  0.01358772  0.02174034  0.03261052  0.04589628  0.06064866  0.07528799  0.08783599  0.09633625  0.09934675
Densités        0.017422576  0.031815138  0.05435086  0.08696138  0.13044207  0.18358513  0.24259464  0.30115196  0.35134396  0.38534498  0.39738701

                      0.25         0.5        0.75           1        1.25         1.5        1.75           2         2.25          2.5
X              33.00000000 34.00000000 35.00000000 36.00000000 37.00000000 38.00000000 39.00000000 40.00000000 41.000000000 42.000000000
Probabilités    0.09633625  0.08783599  0.07528799  0.06064866  0.04589628  0.03261052  0.02174034  0.01358772  0.007953785  0.004355644
Densités        0.38534498  0.35134396  0.30115196  0.24259464  0.18358513  0.13044207  0.08696138  0.05435086  0.031815138  0.017422576
```

Un autre exemple :

```
bingausshub(n = 25, p = .2, binf = -4, bsup = 4)
```




```
[1] "Tableau des valeurs de Z, X, probabilités binomiales, densités"
                   -4 -3.5 -3         -2.5          -2       -1.5         -1       -0.5          0       0.5
X                  -3   -2 -1 0.000000000 1.00000000 2.0000000 3.0000000 4.0000000 5.0000000 6.0000000
Probabilités bino   0    0  0 0.003777893 0.02361183 0.0708355 0.1357680 0.1866811 0.1960151 0.1633459
Densités            0    0  0 0.007555786 0.04722366 0.1416710 0.2715361 0.3733621 0.3920302 0.3266918

                           1        1.5          2        2.5            3          3.5            4
X                  7.0000000 8.00000000 9.00000000 10.00000000 11.000000000 12.000000000 1.300000e+01
Probabilités bino  0.1108419 0.06234855 0.02944237  0.01177695  0.004014869  0.001171003 2.927509e-04
Densités           0.2216837 0.12469711 0.05888475  0.02355390  0.008029738  0.002342007 5.855017e-04
```

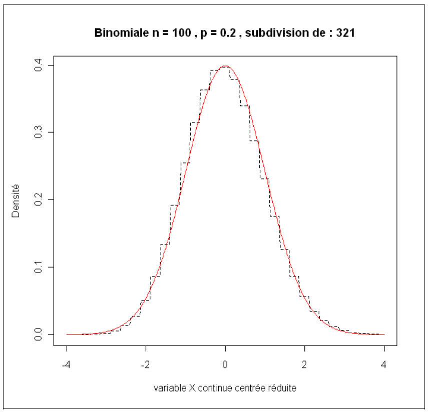
On rappelle que X suit une loi binomiale de paramètres n et p et que Z = (X – n*p) / racine(n*p*(1 – p)). Voici quelques couples n, p, tels que sigma = racine(n*p*(1 – p)) soit entier, dont certaines valeurs de 1/sigma fournissent des valeurs décimales pour les bornes des intervalles des subdivisions entre -4 et 4. L'étendue de chaque intervalle d'une subdivision est 1/sigma.

n	4	25	36	48	64	100	100	100	144	150	192	196
p	0,5	0,2	0,5	0,75	0,5	0,2	0,5	0,9	0,5	0,4	0,75	0,5
sigma	1	2	3	3	4	4	5	3	6	6	6	7
mu	2	5	18	36	32	20	50	90	72	60	144	98

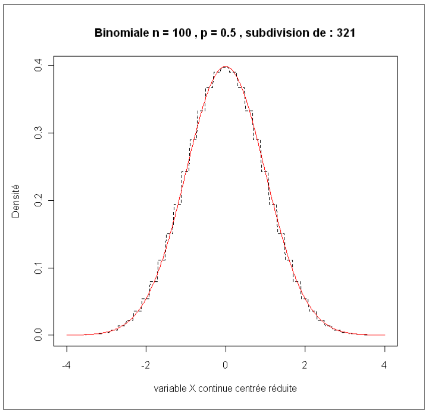
7° Convergence de la loi binomiale vers la loi de Gauss version Robert (BinoGaussRob)

```
binogausrob <- function(n = 30, p = .2, nbpoints = 40 * 8 + 1){
 sigma = sqrt(n * p * (1 - p))
 x <- seq(-4, 4, length.out = nbpoints)
 f <- function(v){dbinom(floor(v + .5), n, p)}
 g <- function(v){sigma * f(n * p + sigma * v)}
#***** Affichage des résultats et des graphiques ***********
 plot(x, g(x), type= "l", lty = 2,
      xlab = "variable X continue centrée réduite", ylab = "Densité",
      main = paste("Binomiale n =", n, ", p =", p, ", subdivision de :", nbpoints))
 points(x, dnorm(x, 0, 1), type = "l", col = "red")
}
```

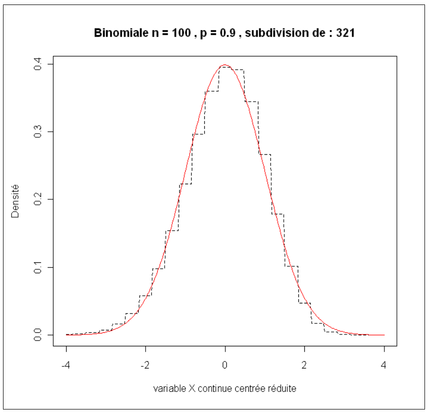
`binogausrob(n = 100, p = .2)`



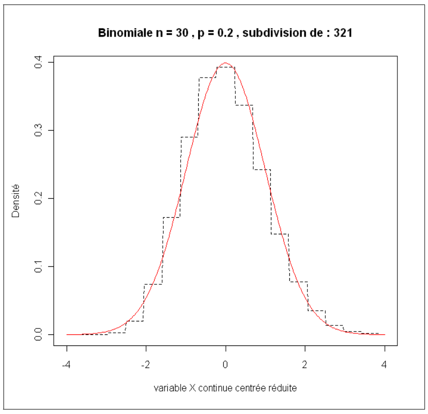
`binogausrob(n = 100, p = .5)`



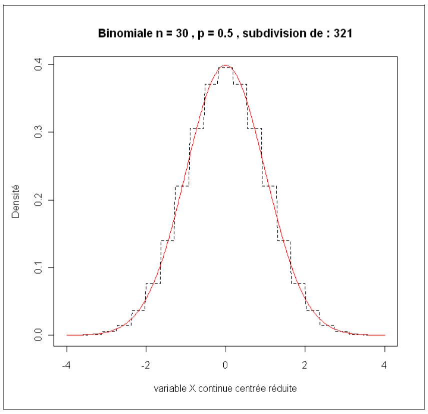
`binogausrob(n = 100, p = .9)`



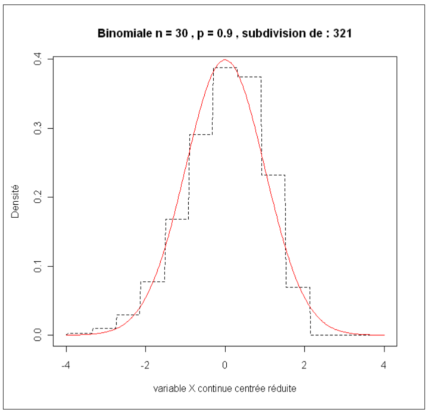
`binogausrob(n = 30, p = .2)`



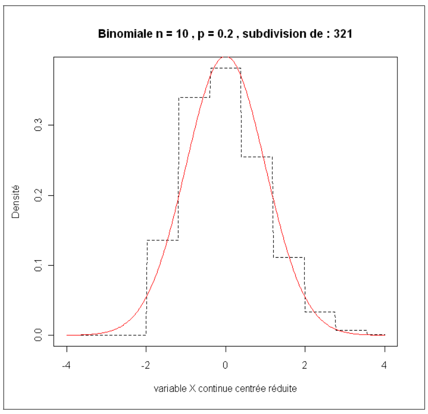
`binogausrob(n = 30, p = .5)`



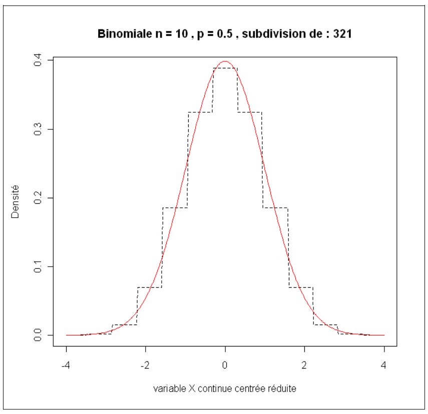
`binogausrob(n = 30, p = .9)`



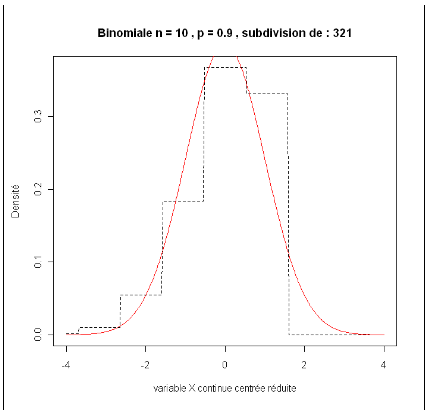
`binogausrob(n = 10, p = .2)`



`binogausrob(n = 10, p = .5)`



`binogausrob(n = 10, p = .9)`



8° Calcul d'un intervalle de fluctuation (IF) bilatéral binomial, "exact" et asymptotique

► La définition de l'IF dans le programme de seconde est la suivante :

*« L'intervalle de fluctuation au seuil de 95%, relatif aux échantillons de taille n, est l'intervalle centré autour de p, proportion du caractère dans la population, où se situe, avec une probabilité **égale** à 0, 95, la fréquence observée dans un échantillon de taille n. ».* Avec une telle définition cet IF n'existe pas dans la plupart des applications numériques. Il faut donc trouver une autre définition.

► Dans le programme de première on trouve : « *Exploiter l'intervalle de fluctuation à un seuil donné, déterminé à l'aide de la loi binomiale, pour rejeter ou non une hypothèse sur une proportion.* » Mais il n'est pas donné de définition de l'IF. Un document de l'inspection générale de math. de l'E.N. précise une définition qui est en fait un mode de calcul de l'IF :

Définition : L'intervalle de fluctuation (**IF**) à 95 % d'une fréquence correspondant à la réalisation, sur un échantillon aléatoire de taille n, d'une variable aléatoire X de loi binomiale, est l'intervalle [a/n ; b/n] défini par :

- a est le plus petit entier tel que $P(X \le a) > 0{,}025$;
- b est le plus petit entier tel que $P(X \le b) \ge 0{,}975$.

L'exemple (exemple 1, Monsieur Z) choisi est une B(100 ; 0,52) ce qui donne un IF = [0,42 ; 0,62]. La proportion observée, de personnes faisant confiance, de 0,43 appartient à cet intervalle. On peut donc accepter l'hypothèse p = 0,52.

► Brigitte et Claudine proposent une autre définition :

L'IF de la variable proportion F_n, au niveau de probabilité de 95% (proba) est le plus petit intervalle de la forme $[p - \alpha\ ; p + \alpha]$ tel que $P(p - \alpha \le F_n \le p + \alpha) \ge 0{,}95$, où la loi de probabilité utilisée est une loi binomiale de paramètres n et p (ppop). Pour mettre en œuvre cette définition, j'ai choisi, **arbitrairement**, comme point central de l'intervalle et de départ du calcul de la probabilité, la valeur entière arrondie de n * p.

► Il y a aussi deux façons de déterminer un IF en utilisant deux approximations asymptotiques de la loi des proportions, celles figurant dans les programmes de seconde et de terminale.

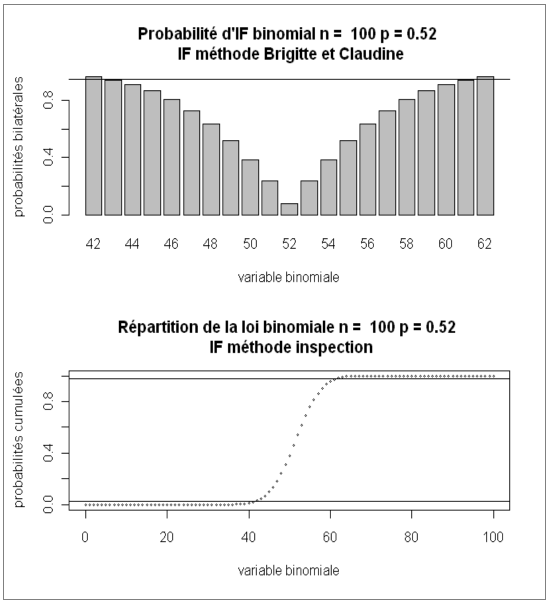
La fonction **R** suivante met en œuvre ces 4 modes de calcul.

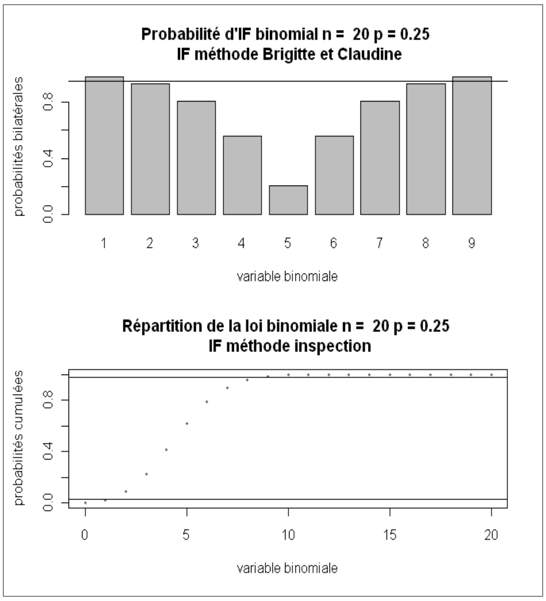
Il est très intéressant de remarquer qu'aucune méthode ne fournit le meilleur IF à coup sûr, parfois on peut même trouver, à la main, un meilleur intervalle, au sens de la définition de Brigitte et Claudine (exemple 1, Monsieur Z). Je présente quelques exemples où les définitions précédentes donnent des résultats différents.

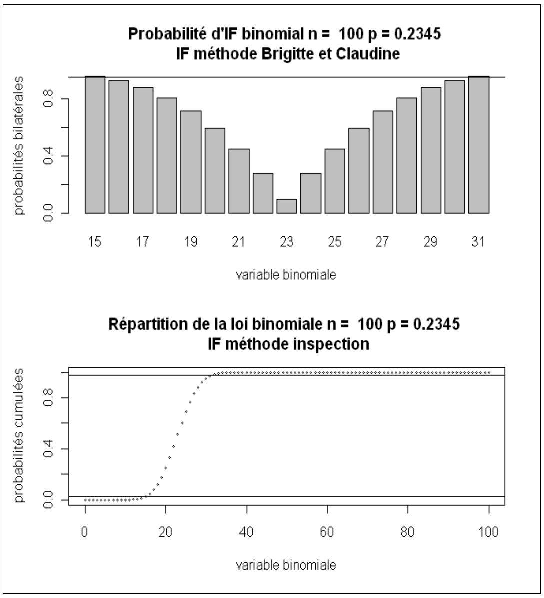
```
# 4 méthodes de calcul d'IF, 2 binomiales, 2 asymtotiques
IFexact = function(n = 100, ppop = .52, kobs = 43, proba = .95){
#****************** IF binomial méthode inspection ****************
 a <- 0 ; b <- 0
 reparti1 <- pbinom(0:n, n, ppop, lower.tail = T)
 names(reparti1) <- 0:n
 p <- 0
 while(p <= (1 - proba)/2){
   p <- pbinom(a, n, ppop, lower.tail = T)
   a <- a + 1              }
 p <- 0
 while(p < (1 - (1 - proba)/2)){
   p <- pbinom(b, n, ppop, lower.tail = T)
   b <- b + 1                   }
 probaab <- sum(dbinom((a - 1):(b - 1), n, ppop))
 if(kobs >= (a - 1) & kobs <= (b - 1)) {
   hypothese <- "ACCEPTÉE"               } else {hypothese <- "REFUSÉE"}
#****************** IF binomial méthode Brigitte et Claudine ****************
 probabil <- vector(length = n + 1)
 names(probabil) <- 0:n
 esper <- round(n * ppop)
 p <- dbinom(esper, n, ppop) ; probabil[esper + 1] <- p
 if(p >= proba) {cat("\nIl n'existe pas d'IF solution au problème posé") ; stop()}
 a3 <- esper - 1 ; b3 <- esper + 1
 if(a3 < 0 | b3 > n) {
   cat(" Algorithme ne gérant pas encore les problèmes particuliers aux bornes 0 et n")
              stop() }
 p <- sum(dbinom(a3:b3, n, ppop))
 probabil[c(a3 + 1,b3 + 1)] <- c(p, p)
 while(p < proba){
   a3 <- a3 - 1 ; b3 <- b3 + 1
   if(a3 < 0 | b3 > n) {
   cat(" Algorithme ne gérant pas encore les problèmes particuliers aux bornes 0 et n")
               stop()  }
   p <- sum(dbinom(a3:b3, n, ppop))
   probabil[c(a3 + 1,b3 + 1)] <- c(p, p)
                   }
 Zprobabil <- probabil[(a3 + 1):(b3 + 1)]
 names(Zprobabil) <- a3:b3
#*************** deux IF asymptotiques méthodes seconde et première ***********
 c1 <- qnorm((1 - (1 - proba)/2), 0, 1)*sqrt(ppop*(1 - ppop)/n)
 asympt1 <- ppop - c1; bsympt1 <- ppop + c1
 asympt2 <- ppop - 1 / sqrt(n) ; bsympt2 <- ppop + 1 / sqrt(n)
#****************** Affichage des résultats **********************
# print("******")
# print(Zprobabil)
# print("******")
 par(mfrow = c(2, 1))
 barplot(Zprobabil, xlab = "variable binomiale", ylab = "probabilités bilatérales",
      main = paste("Probabilité d'IF binomial n = ", n, "p =",ppop,
                   "\n IF méthode Brigitte et Claudine"))
 abline(h = proba)
 plot(0:n,reparti1, type = "p", xlab = "variable binomiale",
     ylab = "probabilités cumulées", cex = .4,
     main = paste("Répartition de la loi binomiale n = ", n, "p =",ppop,
                  "\n IF méthode inspection"))
 abline(h =c((1 - proba)/2, (1 - (1 - proba)/2)))
 cat("\nTableau partiel de la répartition de X\n")
    print(reparti1[(a - 1):a])
    print(reparti1[(b - 1):b])
 cat("\nL'IF exact des comptages méthode inspection est :\n[",
    a - 1,",",b - 1,"] de probabilité :", probaab,
    "\nL'IF exact des proportions méthode inspection est :\n[",
    (a - 1)/n,",",(b - 1)/n,"]\n",
    "Hypothèse p population = ", ppop, ": confrontée à p observé =",kobs/n,
    " : ",hypothese,"\n")
 cat("\nL'IF exact des comptages méthode Brigitte et Claudine est :\n[",
    a3,",",b3,"] de probabilité :", p,
    "\nL'IF exact des proportions méthode Brigitte et Claudine est :\n[",
    a3/n,",",b3/n,"]\n")
 cat("\nL'IF asymptotique des proportions formule première est :\n[",
    asympt1,",",bsympt1,"]\n")
 cat("\nL'IF asymptotique des proportions formule seconde est :\n[",
    asympt2,",",bsympt2,"]\n")
}
```

Les résultats sont représentés ci-dessous :

```
# exemple 1
IFexact()
 Tableau partiel de la répartition de X
         41         42
 0.01772803 0.02857444
        61         62
 0.9718881 0.9826766

 L'IF exact des comptages méthode inspection est :
 [ 42 ; 62 ] de probabilité : 0.9649486
 L'IF exact des proportions méthode inspection est :
 [ 0.42 ; 0.62 ]
  Hypothèse p population =  0.52 : confrontée à
 p observé =  0.43  :  ACCEPTÉE

 L'IF exact des comptages méthode Brigitte et Claudine est :
 [ 42 ; 62 ] de probabilité : 0.9649486
 L'IF exact des proportions méthode Brigitte et Claudine est :
 [ 0.42 ; 0.62 ]

 L'IF asymptotique des proportions formule première est :
 [ 0.4220802 ; 0.6179198 ]

 L'IF asymptotique des proportions formule seconde est :
 [ 0.42 ; 0.62 ]

# exemple 2
 IFexact(n = 20, ppop = .25, kobs = 3, proba = .95)

 Tableau partiel de la répartition de X
          1          2
 0.02431262 0.09126043
         8          9
 0.9590748 0.9861356

 L'IF exact des comptages méthode inspection est :
 [ 2 ; 9 ] de probabilité : 0.961823
 L'IF exact des proportions méthode inspection est :
 [ 0.1 ; 0.45 ]
  Hypothèse p population =  0.25 : confrontée à
 p observé =  0.15  :  ACCEPTÉE

 L'IF exact des comptages méthode Brigitte et Claudine est :
 [ 1 ; 9 ] de probabilité : 0.9829644
 L'IF exact des proportions méthode Brigitte et Claudine est :
 [ 0.05 ; 0.45 ]

 L'IF asymptotique des proportions formule première est :
 [ 0.0602273 ; 0.4397727 ]

# exemple 3
 IFexact(n = 100, ppop = .2345, kobs = 30, proba = .95)

 Tableau partiel de la répartition de X
         14         15
 0.01374957 0.02608588
        31         32
 0.9681745 0.9810043

 L'IF exact des comptages méthode inspection est :
 [ 15 ; 32 ] de probabilité : 0.9672547
 L'IF exact des proportions méthode inspection est :
 [ 0.15 ; 0.32 ]
  Hypothèse p population =  0.2345 : confrontée à p observé =
 0.3  :  ACCEPTÉE

 L'IF exact des comptages méthode Brigitte et Claudine est :
  [ 15 ; 31 ] de probabilité : 0.954425
 L'IF exact des proportions méthode Brigitte et Claudine est :
  [ 0.15 ; 0.31 ]

 L'IF asymptotique des proportions formule première est :
 [ 0.1514591 ; 0.3175409 ]

 L'IF asymptotique des proportions formule seconde est :
 [ 0.1345 ; 0.3345 ]
```







Avec **R**, la fonction **qbinom()** répond presque à la définition de l'IF : **qbinom(proba, n, ppop)** calcule le plus petit entier k telle que P(X ≤ k) ≥ proba, la loi de X étant une loi binomiale de paramètre n et ppop. Il peut être intéressant de rechercher des exemples où les deux définitions divergent.

Détermination de a :	**Détermination de b :**

`qbinom(.025, 100, .52)/100`
`  [1] 0.42`
 	
`qbinom(.05, 30, .2)/30`
`  [1] 0.1`
	`qbinom(.975, 100, .52)/100`
	`  [1] 0.62`
	
	`qbinom(.95, 30, .2)/30`
	`  [1] 0.3333333`

On peut aussi utiliser sum(dbinom(vectx, n, ppop)) pour explorer des intervalles.

Dans le cas de l'exemple 1 (Monsieur Z), il existe 2 intervalles meilleurs, c'est à dire plus petits que ceux déterminés en appliquant les deux méthodes présentées : [43 ; 62] et [42 ; 61], au lieu de [42 ; 62].

P(42 ≤ X ≤ 62) =
`sum(dbinom(42:62, 100, .52))`
`  [1] 0.9649486*`
P(43 ≤ X ≤ 62) =
`sum(dbinom(43:62, 100, .52))`
`  [1] 0.9541022`
	P(42 ≤ X ≤ 61) =
	`sum(dbinom(42:61, 100, .52))`
	`  [1] 0.95416`

Dans l'exemple 2 c'est la méthode inspection qui donne le plus petit intervalle, mais on trouve un autre intervalle, proche, de même étendue, de probabilité ≥ 0,95 :
P(2 ≤ X ≤ 8) = `sum(dbinom((1:8, 20, .25))`
`[1] 0.9559036`

Dans l'exemple 3 c'est la méthode Brigitte et Claudine qui donne le plus petit intervalle, mais on trouve aussi un autre intervalle, proche, de même étendue, de probabilité ≥ 0,95 :
P(16 ≤ X ≤ 32) = `sum(dbinom((16:32, 100, .2345))`
`[1] 0.9549184`

On pourrait aussi soumettre à la comparaison la troisième méthode suivantes :

Autre définition : L'intervalle de fluctuation (**IF**) à 95 % d'une fréquence correspondant à la réalisation, sur un échantillon aléatoire de taille n, d'une variable aléatoire X de loi binomiale, est l'intervalle [a/n ; b/n] défini par :

- a est le plus grand entier tel que $P(X \geq a) \geq 0{,}975$;
- b est le plus petit entier tel que $P(X \leq b) \geq 0{,}975$.

L'algorithme est à faire...

Le même problème se pose lors de la recherche du meilleur intervalle de confiance d'une proportion, calculé à partir d'une variable parent binomiale. Mais dans ce cas il existe une littérature abondante traitant de ce problème, dont voici trois références significatives que je tiens à votre disposition. On voit bien que le problème ne date pas d'aujourd'hui !!

1° Clopper C. J., Pearson E. S. ; 1934 ; The Use of Confidence or Fiducial Limits Illustrated in the Case of the Binomial ; Biometrika ; Vol. 26 ; No. 4 ; pp. 404-413.
2° Blyth C. R., Still H. A. ; 1983, Binomial Confidence Intervals ; J.A.S.A. ; Vol. 78 ; No. 381 ; pp. 108-116.
3° Casella G. ; 1986 ; Refining Binomial Confidence Intervals ; The Canadian Journal of Statistics / La Revue Canadienne de Statistique ; Vol. 14 ; No. 2 ; pp. 113-129

9° Calcul de l'intervalle de confiance (IC) d'une proportion (terminale S)

Il y a plusieurs méthodes de calcul d'un **IC** "exact" d'une proportion, à cause du caractère discret de la variable binomiale. Les méthode asymptotiques sont les résultats de différentes approximations. Celle proposée dans le programme de terminale se met en œuvre par un simple calcul numérique dont on peut comparer les résultats avec quelques méthodes classiques exactes et asymptotiques disponibles dans **R** et avec l'**IF**.

Reprenons l'exemple 1 des 52% de la déclaration de Monsieur Z, que l'on veut confronter aux 43 personnes favorables dans un échantillon aléatoire et simple de 100. Quelle conclusion si on observe 42 personnes?

```
ICGaussEx = function(n = 100, kobs = 43, ppop = .52, proba = .95){
#**** IC asymptotique gaussien au seuil de 95%, formule terminale S *******
 fobs <- kobs/n
 aasympt <- fobs - 1/sqrt(n) ; absympt <- fobs + 1/sqrt(n)
  if(ppop >= aasympt & ppop <= absympt) {
   hypothese <- "ACCEPTÉE"              } else {hypothese <- "REFUSÉE"}
 cat("\nIC à 95%, approximation gaussienne de la loi binomiale",
    " selon programme Term S =\n",
    "[", aasympt, ";", absympt,"] Hypothèse ppop =", ppop, ":", hypothese, "\n")
#*********** IC asymptotique gaussien au seuil de proba, Classique  ********
 c <- qnorm((1 - (1 - proba)/2), 0, 1)*sqrt(fobs*(1 - fobs)/n)
 basympt <- fobs - c ; bbsympt <- fobs + c
 if(ppop >= basympt & ppop <= bbsympt) {
 hypothese <- "ACCEPTÉE"              } else {hypothese <- "REFUSÉE"}
 cat("\nIC à", proba, "approximation gaussienne classique de la loi binomiale =\n",
    "[", basympt, ";", bbsympt, "] Hypothèse ppop =", ppop, ":", hypothese, "\n")
#********** IC asymptotique gaussien au seuil de proba, selon R ***********
 require("epitools", quietly = T, warn.conflicts = F)
 ICGauss <- binom.approx(kobs, n, conf.level = proba)
 if(ppop >= ICGauss$lower & ppop <= ICGauss$upper) {
 hypothese <- "ACCEPTÉE"                        } else {hypothese <- "REFUSÉE"}
 cat("\nIC à", proba, "approximation gaussienne classique",
    " de la loi binomiale selon R :\n")
 print(ICGauss)
 cat("Hypothèse ppop =", ppop, ":", hypothese, "\n")
#************ IC exact binomial au seuil de proba, selon R ***************
 ICExact <- binom.exact(kobs, n, conf.level = proba)
 if(ppop >= ICExact$lower & ppop <= ICExact$upper) {
 hypothese <- "ACCEPTÉE"                        } else {hypothese <- "REFUSÉE"}
 cat("\nIC à", proba, "binomial exact selon R :\n")
 print(ICExact)
 cat("Hypothèse ppop =", ppop, ":", hypothese, "\n")
}
ICGaussEx()
IC à 95%, approximation gaussienne de la loi binomiale  selon programme Term S =
    [ 0.33 ; 0.53 ] Hypothèse ppop = 0.52 : ACCEPTÉE
IC à 0.95 approximation gaussienne classique de la loi binomiale =
    [ 0.3329669 ; 0.5270331 ] Hypothèse ppop = 0.52 : ACCEPTÉE
IC à 0.95 approximation gaussienne classique  de la loi binomiale selon R :
       x   n proportion     lower     upper conf.level
    1 43 100       0.43 0.3329669 0.5270331       0.95
    Hypothèse ppop = 0.52 : ACCEPTÉE
IC à 0.95 binomial exact selon R :
       x   n proportion    lower     upper conf.level
    1 43 100       0.43 0.331391 0.5328663       0.95
Hypothèse ppop = 0.52 : ACCEPTÉE
ICGaussEx(kobs = 42)
IC à 95%, approximation gaussienne de la loi binomiale  selon programme Term S =
    [ 0.32 ; 0.52 ] Hypothèse ppop = 0.52 : ACCEPTÉE
IC à 0.95 approximation gaussienne classique de la loi binomiale =
    [ 0.3232643 ; 0.5167357 ] Hypothèse ppop = 0.52 : REFUSÉE
IC à 0.95 approximation gaussienne classique  de la loi binomiale selon R :
       x   n proportion     lower     upper conf.level
    1 42 100       0.42 0.3232643 0.5167357       0.95
    Hypothèse ppop = 0.52 : REFUSÉE
IC à 0.95 binomial exact selon R :
       x   n proportion     lower     upper conf.level
    1 42 100       0.42 0.3219855 0.5228808       0.95
    Hypothèse ppop = 0.52 : ACCEPTÉE
```

10° Les probabilités dans un modèle d'urne, échantillonnage sans remise

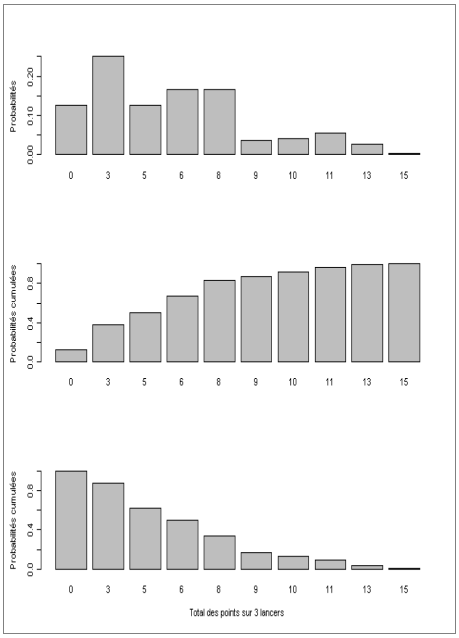
Une urne contient 3 boules rouges et 5 boules noires. On tire au hasard 4 boules **SANS remise**.
Quelle est la probabilité d'obtenir 3 boules rouges? On peut utiliser les fonction de calcul des combinaisons :

Calcul de P(X=3) :	Pour vérification :
`(proba <- choose(3,3)*choose(5,4-3)/choose(3+5,4))` `[1] 0.07142857`	`(proba <- dhyper(x = 3, m = 3, n = 5, k = 4))` `[1] 0.07142857`

11° Calculer les probabilités dans quelques exercices d'annales de bac

1 / 2011-S-Avril-Pondichéry : fléchettes, Épreuves successives, répétées, à trois issues

```
# Partie A 1/4. Tout l'arbre. Distributions d'une variable aléatoire discrète. L'algorithme
# parcourt tout l'arbre des résultats possibles, sans s'arrêter quand la partie est gagnée.
troislancers = function( point = c(0, 3, 5), proba = c(1/2, 1/3, 1/6), gagne = 8){
 point3lancers <- vector(length = 27) ; proba3lancers <- vector(length = 27)
 pg1 <- 0 ; pg2 <- ; pg3 <- 0 ; L <- 1
 for(i in 1:3){
   if(point[i] >= gagne) pg1 <- pg1 + proba[i]
   for(j in 1:3){
     if(point[i] < gagne & point[i] + point[j] >= gagne){
       pg2 <- pg2 + proba[i] * proba[j]                    }
     for(k in 1:3){
       if(point[i] + point[j] < gagne & point[i] + point[j] + point[k] >= gagne){
         pg3 <- pg3 + proba[i] * proba[j] * proba[k]                              }
       point3lancers[L] <- point[i] + point[j] + point[k]
       proba3lancers[L] <- proba[i] * proba[j] * proba[k]
       L <- L+1
                  }
                }
              }
 tablopoint <- table(point3lancers)
 nbpoint <- length(tablopoint)
 distpoint <- vector(length = nbpoint)
 names(distpoint) <- as.numeric(names(tablopoint))
 for(i in 1:nbpoint){
   distpoint[i] <-
   sum(proba3lancers[which(point3lancers == as.numeric(names(tablopoint[i])))])
                   }
 probaperdre <- sum(distpoint[as.numeric(names(distpoint)) < gagne])
 probagagner <- pg1 + pg2 + pg3
 distribX <- c(probaperdre, pg3, pg2)
 names(distribX) <- c(-2, 1, 3)
 esperanceX <- sum(distribX * as.numeric(names(distribX)))
# ************************** Affichage des résultats*********************************
 cat("Distribution de la variable nombre de points à l'issue de 3 lancers \n")
 print(distpoint)
 cat("\n Répartition de la variable nombre de points à l'issue de 3 lancers \n")
 print(cumsum(distpoint))
 cat("\n Proba cumulées décroissantes de la variable",
     " nombre de points à l'issue de 3 lancers \n")
 print(cumsum(distpoint[nbpoint:1])[nbpoint:1])
 cat("\n Probabilité de perdre la partie =", probaperdre, "\n")
 cat(" Probabilité de gagner la partie en 1 lancer, P(g1) =", pg1, "\n")
 cat(" Probabilité de gagner la partie en 2 lancers, P(g2) =", pg2, "\n")
 cat(" Probabilité de gagner la partie en 3 lancers P(g3) =", pg3, "\n")
 cat(" Probabilité de gagner la partie =", probagagner, "\n \n")
 cat("En supposant toutes les parties indépendantes, ce qui voudrait dire \n",
     " qu'il n'y a pas d'apprentissage, \n",
     "la probabilité de gagner au moins une fois en 6 parties est :",
      1 - (1 - probagagner)^6, "\n \n")
 cat("La distribution de X est :\n")
 print(distribX)
 cat("L'espérance de X est :",esperanceX, "€ \n")
 par(mfrow = c(3, 1))
 barplot(distpoint, ylab = "Probabilités")
 barplot(cumsum(distpoint), ylab = "Probabilités cumulées")
 barplot(cumsum(distpoint[nbpoint:1])[nbpoint:1],
   ylab = "Probabilités cumulées", xlab = "Total des points sur 3 lancers")
}
```




```
# Partie A 2/4.Tout l'arbre. Distributions d'une variable
# aléatoire discrète.
troislancers()
Distribution de la variable nombre de points à l'issue de 3 lancers
         0          3          5          6          8          9
0.12500000 0.25000000 0.12500000 0.16666667 0.16666667 0.03703704
        10         11         13         15
0.04166667 0.05555556 0.02777778 0.00462963

 Répartition de la variable nombre de points à l'issue de 3 lancers
        0         3         5         6         8         9        10
0.1250000 0.3750000 0.5000000 0.6666667 0.8333333 0.8703704 0.9120370
       11        13        15
0.9675926 0.9953704 1.0000000

 Proba cumulées décroissantes de la variable  nombre de points à l'issue
de 3 lancers
         0          3          5          6          8          9
1.00000000 0.87500000 0.62500000 0.50000000 0.33333333 0.16666667
        10         11         13         15
0.12962963 0.08796296 0.03240741 0.00462963

 Probabilité de perdre la partie = 0.6666667
 Probabilité de gagner la partie en 1 lancer, P(g1) = 0
 Probabilité de gagner la partie en 2 lancers, P(g2) = 0.1388889
 Probabilité de gagner la partie en 3 lancers P(g3) = 0.1944444
 Probabilité de gagner la partie = 0.3333333

En supposant toutes les parties indépendantes, ce qui voudrait dire
  qu'il n'y a pas d'apprentissage,
 la probabilité de gagner au moins une fois en 6 parties est : 0.9122085

La distribution de X est :
       -2         1         3
0.6666667 0.1944444 0.1388889
L'espérance de X est : -0.7222222 €
```

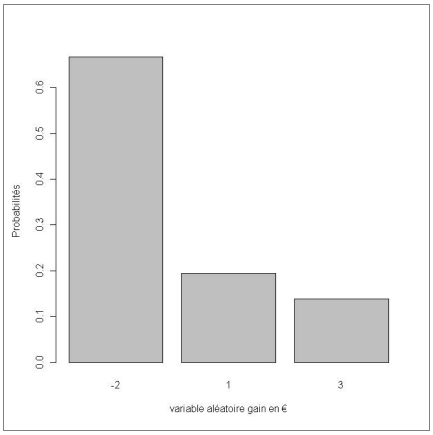
Que se passe-t-il lorsque le total de points pour gagner change?

```
troislancers(gagne = 5)
 Probabilité de perdre la partie = 0.375
 Probabilité de gagner la partie en 1 lancer, P(g1) = 0.1666667
 Probabilité de gagner la partie en 2 lancers, P(g2) = 0.25
 Probabilité de gagner la partie en 3 lancers P(g3) = 0.2083333
 Probabilité de gagner la partie = 0.625

En supposant toutes les parties indépendantes, ce qui voudrait dire
  qu'il n'y a pas d'apprentissage,
 la probabilité de gagner au moins une fois en 6 parties est : 0.997219

La distribution de X est :
       -2         1         3
0.3750000 0.2083333 0.2500000
L'espérance de X est : 0.2083333 €

troislancers(gagne = 6)
 Probabilité de perdre la partie = 0.5
 Probabilité de gagner la partie en 1 lancer, P(g1) = 0
 Probabilité de gagner la partie en 2 lancers, P(g2) = 0.25
 Probabilité de gagner la partie en 3 lancers P(g3) = 0.25
 Probabilité de gagner la partie = 0.5

En supposant toutes les parties indépendantes, ce qui voudrait dire
  qu'il n'y a pas d'apprentissage,
 la probabilité de gagner au moins une fois en 6 parties est : 0.984375

La distribution de X est :
  -2    1    3
0.50 0.25 0.25
L'espérance de X est : 0 €
```

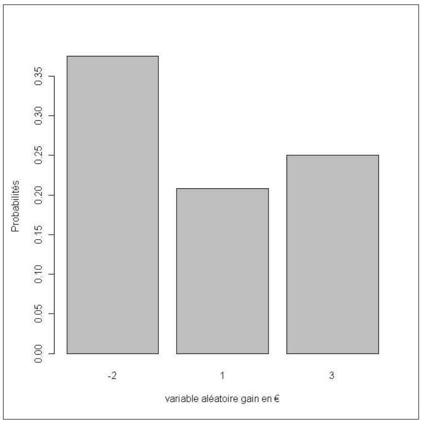
2011-S-Avril-Pondichéry : fléchettes, interruption des lancers lorsque la partie est gagnée.

```
# Partie B 3/4. Arbre partiel. Distributions d'une variable aléatoire discrète. L'algorithme
# arrête le parcours de l'arbre quand la partie est gagnée.
flechearret = function( point = c(0, 3, 5), proba = c(1/2, 1/3, 1/6), gagne = 8){
 PtsLancers <- array(0,dim=c(3,3,3))
 ProbLancers <- array(0,dim=c(3,3,3))
 pg1 <- 0
 pg2 <- 0
 pg3 <- 0
 for(i in 1:3){
   if(point[i] >= gagne)   {
     pg1 <- pg1 + proba[i]
     for(M in 1:3){for(N in 1:3){PtsLancers[i, M, N] <- point[i]}}
                          }  else {
       for(j in 1:3){
         if(point[i] + point[j] >= gagne){
           pg2 <- pg2 + proba[i] * proba[j]
           for(N in 1:3){PtsLancers[i, j, N] <- point[i] + point[j]}
                                          } else {
             for(k in 1:3){
               if(point[i] + point[j] + point[k] >= gagne)             {
                 pg3 <- pg3 + proba[i] * proba[j] * proba[k]
                 PtsLancers[i, j, k] <- point[i] + point[j] + point[k] }
                          }
                                                }
                     }
                                    }
               }
 point3lancers <- as.vector(PtsLancers)
 names(point3lancers) <- 1:27
 tablopoint <- table(point3lancers)
 nbpoint <- length(tablopoint)
 probagagner <- pg1 + pg2 + pg3
 probaperdre <- 1 - probagagner
 distribX <- c(probaperdre, pg3, pg2)
 names(distribX) <- c(-2, 1, 3)
 esperanceX <- sum(distribX * as.numeric(names(distribX)))
#********************************** Affichage des résultats ************************************
 cat("\n Branches gagnantes et points obtenus au moins égaux à", gagne, "\n")
 print(point3lancers)
 cat("\n Probabilité de perdre la partie =", probaperdre, "\n")
 cat(" Probabilité de gagner la partie en 1 lancer, P(g1) =", pg1, "\n")
 cat(" Probabilité de gagner la partie en 2 lancers, P(g2) =", pg2, "\n")
 cat(" Probabilité de gagner la partie en 3 lancers P(g3) =", pg3, "\n")
 cat(" Probabilité de gagner la partie =", probagagner, "\n \n")
 cat("En supposant toutes les parties indépendantes, ce qui voudrait dire \n",
     " qu'il n'y a pas d'apprentissage, \n",
     "la probabilité de gagner au moins une fois en 6 parties est :",
      1 - (1 - probagagner)^6, "\n \n")
 cat("La distribution de X est :\n")
 print(distribX)
 cat("L'espérance de X est :",esperanceX, "€ \n")
barplot(distribX, xlab = "variable aléatoire gain en €", ylab = "Probabilités")
}
```

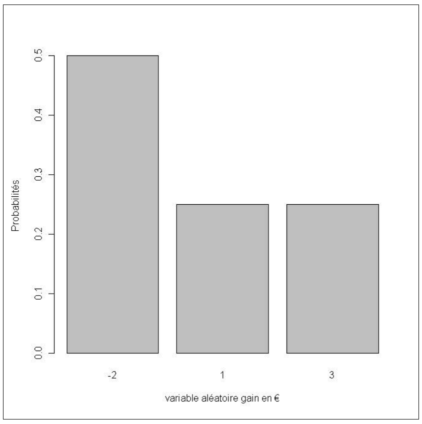
```
# Partie B 4/4. Arbre partiel. Distributions d'une variable aléatoire discrète. L'algorithme
# arrête le parcours de l'arbre quand la partie est gagnée.
flechearret()
 Branches gagnantes et points obtenus au moins égaux à 8

 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 0  0  0  0  0  8  0  8 10  0  0  8  0  9  8  8 10  0  8 10  8 11
24 25 26 27
 8 10  8 10

 Probabilité de perdre la partie = 0.6666667
 Probabilité de gagner la partie en 1 lancer, P(g1) = 0
 Probabilité de gagner la partie en 2 lancers, P(g2) = 0.1388889
 Probabilité de gagner la partie en 3 lancers P(g3) = 0.1944444
 Probabilité de gagner la partie = 0.3333333

En supposant toutes les parties indépendantes, ce qui voudrait dire
  qu'il n'y a pas d'apprentissage,
 la probabilité de gagner au moins une fois en 6 parties est : 0.9122085

La distribution de X est :
       -2         1         3
0.6666667 0.1944444 0.1388889
L'espérance de X est : -0.7222222 €
```




```
flechearret(gagne = 5)
 Branches gagnantes et points obtenus au moins égaux à 5
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 0  0  5  0  6  5  5  8  5  0  6  5  6  6  5  5  8  5  5  8  5  8  6
24 25 26 27
 5  5  8  5

 Probabilité de perdre la partie = 0.375
 Probabilité de gagner la partie en 1 lancer, P(g1) = 0.1666667
 Probabilité de gagner la partie en 2 lancers, P(g2) = 0.25
 Probabilité de gagner la partie en 3 lancers P(g3) = 0.2083333
 Probabilité de gagner la partie = 0.625

En supposant toutes les parties indépendantes, ce qui voudrait dire
  qu'il n'y a pas d'apprentissage,
 la probabilité de gagner au moins une fois en 6 parties est : 0.997219

La distribution de X est :
       -2         1         3
0.3750000 0.2083333 0.2500000
L'espérance de X est : 0.2083333 €
```



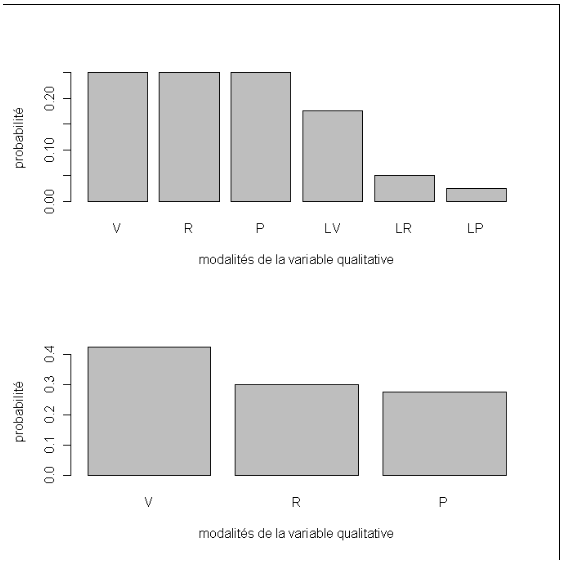

```
flechearret(gagne = 6)
 Branches gagnantes et points obtenus au moins égaux à 6
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
 0  0  0  0  6  8  0  8 10  0  6  8  6  6  8  8  8 10  0  8 10  8  6
24 25 26 27
 8 10  8 10

 Probabilité de perdre la partie = 0.5
 Probabilité de gagner la partie en 1 lancer, P(g1) = 0
 Probabilité de gagner la partie en 2 lancers, P(g2) = 0.25
 Probabilité de gagner la partie en 3 lancers P(g3) = 0.25
 Probabilité de gagner la partie = 0.5

En supposant toutes les parties indépendantes, ce qui voudrait dire
  qu'il n'y a pas d'apprentissage,
 la probabilité de gagner au moins une fois en 6 parties est : 0.984375

La distribution de X est :
  -2    1    3
0.50 0.25 0.25
L'espérance de X est : 0 €
```



2 / 2011-S-Mars-Nouvelle Calédonie : villages sport, Probabilités conditionnelles et loi binomiale

```
# Distributions de variables qualitatives
villagesport = function(jetons = c(1/4, 1/4, 1/4, 1/4),
JL = c(.7, .2, .1)){
 names(jetons) <- c("V", "R", "P", "L")
 names(JL) <- c("LV", "LR", "LP")
 vectProba <- vector(length = 6)
 names(vectProba) <- c("V", "R", "P", "LV", "LR", "LP")
 vectProba[1:3] <- jetons[1:3]
 vectProba[4:6] <- jetons[4] * JL
 ProbaVRP <- vectProba[1:3]+vectProba[4:6]
 probaVelo <- sum(vectProba[which(names(vectProba)=="V"|
names(vectProba)=="LV")])
 # probaVelo <- sum(vectproba[1],vectproba[4])
 pLsachVelo <-
vectProba[which(names(vectProba)=="LV")]/probaVelo
 # pLsachVelo <- vectProba[4]/(vectProba[1] +
vectProba[4])
 AuMoins1NonVelo <- 1-(1-2/3)^6
#***************** Affichage des résultats *******************
 cat("\n La distribution des probabilités sur l'univers
est : \n")
 print(vectProba)
 cat("La distribution des probabilités sur V R P
est : \n")
 print(ProbaVRP)
 cat("\n")
 cat("La probabilité de vélo est :", probaVelo, "\n")
 cat("La probabilité de jeton L sachant vélo est :",
pLsachVelo, "\n")
 cat("La probabilité d'au moins un non vélo en 6 ans est
:", AuMoins1NonVelo, "\n")
 par(mfrow=c(2,1))
 barplot(vectProba, xlab="modalités de la variable
qualitative", ylab="probabilité")
 barplot(ProbaVRP, xlab="modalités de la variable
qualitative", ylab="probabilité")
}
```

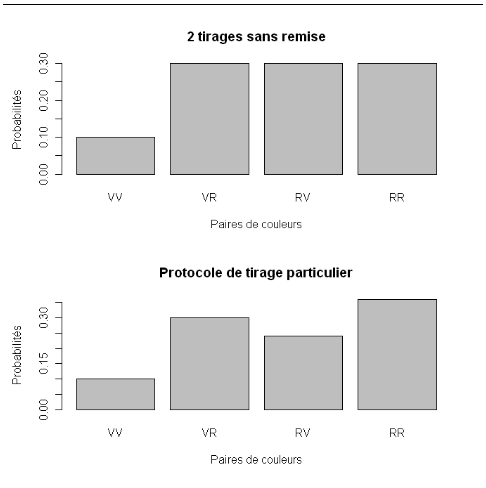
```
villagesport()
 La distribution des probabilités sur
l'univers est :
    V     R     P    LV    LR    LP
0.250 0.250 0.250 0.175 0.050 0.025
La distribution des probabilités sur V
R P est :
    V     R     P
0.425 0.300 0.275

La probabilité de vélo est : 0.425
La probabilité de jeton L sachant vélo
est : 0.4117647
La probabilité d'au moins un non vélo
en 6 ans est : 0.9986283
```



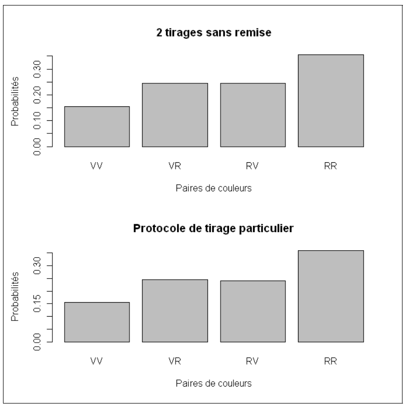
3 / 2010-S-Novembre-Nouvelle-Calédonie : urne boules, tirages avec et sans remise, probabilités conditionnelles

```
# Probabilité d'événements et distribution d'une variable
cinqboules = function(vertes = 2, rouges = 3, n = 2){
#************* QUESTIONS 1 ***************************
 urne <- vertes + rouges
 P1vv <- vertes/urne * (vertes - 1) / (urne - 1)
 P1vr <- vertes/urne * (rouges) / (urne - 1)
 P1rv <- rouges/urne * (vertes) / (urne - 1)
 P1rr <- rouges/urne * (rouges - 1) / (urne - 1)
 distrib1paires <- c(P1vv, P1vr, P1rv, P1rr)
 names(distrib1paires) <- c("VV", "VR", "RV", "RR")
 P1_0verte <- distrib1paires[4]
 distrib1X <- dhyper(0:n, vertes, rouges, n)
 names(distrib1X) <- 0:n
 esperance1X <- sum(distrib1X *
as.numeric(names(distrib1X)))
 P1A <- sum(dhyper(c(0, 2), vertes, rouges, n))
#************* QUESTIONS 2 **************************
 P2vv <- vertes/urne * (vertes - 1) / (urne - 1)
 P2vr <- vertes/urne * (rouges) / (urne - 1)
 P2rv <- rouges/urne * (vertes) / urne
 P2rr <- rouges/urne * rouges / urne
 distrib2paires <- c(P2vv, P2vr, P2rv, P2rr)
 names(distrib2paires) <- c("VV", "VR", "RV", "RR")
 P2C <- distrib2paires[2] + distrib2paires[3]
 P2sach <- distrib2paires[2] / P2C
#***************** Affichage des résultats ******************
 cat("\n Proba de 0 verte =", P1_0verte, "\n")
 cat("\n Distribution des probabilités 1 des paires
résultats :\n")
 print(distrib1paires)
 cat("\n Distribution 1 de la variable aléatoire X :\n")
 print(distrib1X)
 cat(" Espérance de X =", esperance1X, "\n")
 cat("\n Proba de A =", P1A, "\n")
 cat("\n Distribution des probabilités 2 des paires
résultats :\n")
 print(distrib2paires)
 cat("\n Proba de B seule la première est verte =",
distrib2paires[2], "\n")
 cat(" Proba de C une seule est verte =",
distrib2paires[2] + distrib2paires[3], "\n")
 cat(" Sachant une seule verte, Proba que ce soit la
première =", P2sach, "\n")
 par(mfrow = c(2,1))
 barplot(distrib1paires, xlab="Paires de couleurs",
ylab="Probabilités",
         main="2 tirages sans remise")
 barplot(distrib2paires, xlab="Paires de couleurs",
ylab="Probabilités",
         main="Protocole de tirage particulier")
}
```

```
cinqboules()
 Proba de 0 verte = 0.3

 Distribution des probabilités 1 des
paires résultats :
 VV  VR  RV  RR
0.1 0.3 0.3 0.3

 Distribution 1 de la variable
aléatoire X :
  0   1   2
0.3 0.6 0.1
 Espérance de X = 0.8

 Proba de A = 0.4

 Distribution des probabilités 2 des
paires résultats :
  VV   VR   RV   RR
0.10 0.30 0.24 0.36

 Proba de B seule la première est verte
= 0.3
 Proba de C une seule est verte = 0.54
 Sachant une seule verte, Proba que ce
soit la première = 0.5555556
```



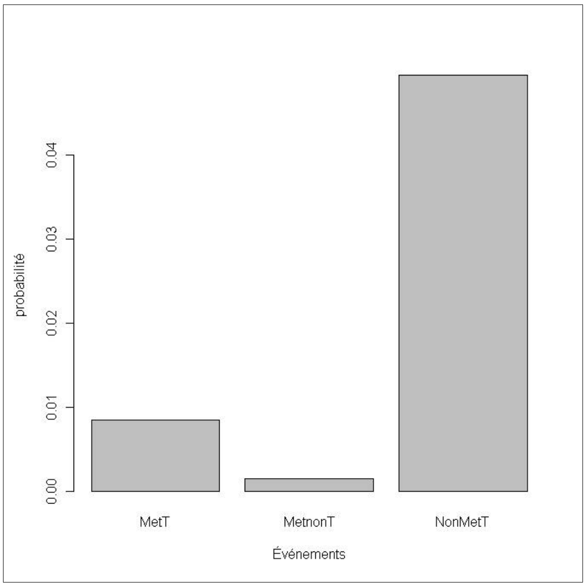

```
cinqboules(vertes=20, rouges=30)
 Proba de 0 verte = 0.3551020
 Distribution des probabilités 1 des paires résultats :
       VV         VR         RV         RR
0.1551020 0.2448980 0.2448980 0.3551020
 Distribution 1 de la variable aléatoire X :
        0         1         2
0.3551020 0.4897959 0.1551020
 Espérance de X = 0.8
 Proba de A = 0.5102041
 Distribution des probabilités 2 des paires résultats :
       VV         VR         RV         RR
0.1551020 0.2448980 0.2400000 0.3600000
 Proba de B seule la première est verte = 0.2448980
 Proba de C une seule est verte = 0.484898
 Sachant une seule verte, Proba que ce soit la première =
0.5050505
```



4 / 2010-S-Septembre-Antilles : Bovins malades et test dépistage, probabilités conditionnelles, loi binomiale

```
# Probabilité d'événements et distribution d'une variable
MaladBovins = function(M = .01, TsM = .85, TsNonM = .05){
 arbrepond <- c(M * TsM, M * (1 - TsM), (1 - M) * TsNonM,
(1 - M)* (1 - TsNonM))
 names(arbrepond) <- c("MetT", "MetnonT", "NonMetT",
"NonMetNonT")
 PdeT <- arbrepond[1] + arbrepond[3]
 PdeMsachantT <- arbrepond[1]/PdeT
 AuMoins1sur5 <- 1 - (1-PdeT)^5
 distcout <- c(arbrepond[4], arbrepond[1] + arbrepond[3],
arbrepond[2])
 names(distcout) <- c(0, 100, 1000)
 esperancecout <- sum(distcout *
as.numeric(names(distcout)))
 troupeau200 <- 200 * esperancecout
#***************** Affichage des résultats *******************
 cat("\n Probabilités au bout des branches\n")
 print(arbrepond)
 cat("\n Proba de M et T =", arbrepond[1], "\n")
 cat(" Proba de T =", PdeT, "\n")
 cat(" Sachant T, Proba de M =", PdeMsachantT, "\n")
 cat("\n Au moins T un sur les cinq =", AuMoins1sur5,
"\n")
 cat("\n Espérance de la variable coût =", esperancecout,
"€ \n")
 cat(" Espérance de coût pour un troupeau de 200 bovins
=", troupeau200, "€ \n")
 barplot(arbrepond[1:3], xlab = "Événements", ylab =
"Probabilité")
}
MaladBovins(TsM = .95, TsNonM = .01)
  Probabilités au bout des branches
      MetT    MetnonT    NonMetT NonMetNonT
    0.0095     0.0005     0.0099     0.9801

  Proba de M et T = 0.0095
  Proba de T = 0.0194
  Sachant T, Proba de M = 0.4896907

  Au moins T un sur les cinq = 0.0933087

  Espérance de la variable coût = 2.44 €
  Espérance de coût pour un troupeau de 200 bovins = 488 €
```

```
MaladBovins()
  Probabilités au bout des branches
      MetT    MetnonT    NonMetT NonMetNonT
    0.0085     0.0015     0.0495     0.9405

  Proba de M et T = 0.0085
  Proba de T = 0.058
  Sachant T, Proba de M = 0.1465517

  Au moins T un sur les cinq = 0.2582552

  Espérance de la variable coût = 7.3 €
  Espérance de coût pour un troupeau de 200
 bovins = 1460 €
```



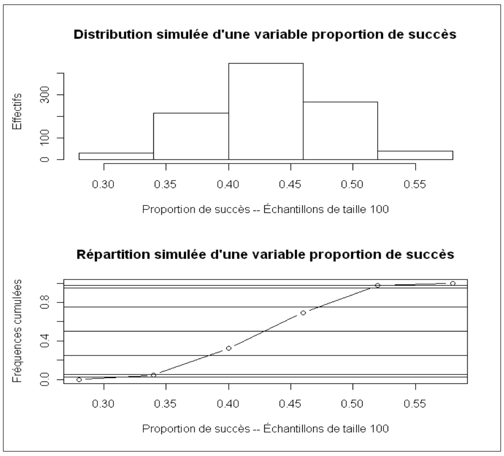
D – RÉÉCHANTILLONNAGE (OU BOOTSTRAP) POUR DÉTERMINER UN INTERVALLE DE CONFIANCE

1° Intervalle de confiance rééchantillonné (IC*) d'une proportion

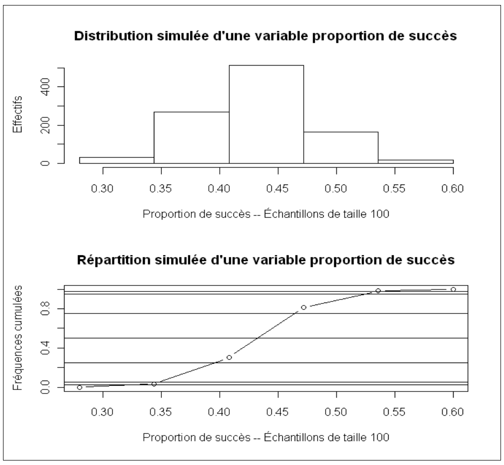
La procédure standard consiste à tirer, avec remise, un grand nombre (nbsim) d'échantillons de taille n, dans une urne dans laquelle la proportion de succès est celle de l'échantillon réellement observé. On obtient ainsi une série de nbsim valeurs simulées, du nombre de succès. On calcule la série des proportions simulées correspondantes. L'intervalle de confiance simulé, de la proportion dans la population, se détermine par les quantiles de la série simulée, correspondants aux seuils de probabilité voulus. Certains modes de calcul de l'**IC** asymptotique d'une proportion sont au programme de terminale S.

```
# Distribution* rééchantillonnée (bootstrap) et IC* rééchantillonné d'une proportion
ICpropboot = function(n= 100, kobs = 43, nbsim = 1000, nbclass = 5, ppop = .52, proba = .95){
 urne <- 1:n
 vectprop <- vector(length = nbsim)
 for(i in 1:nbsim){
   reechant <- sample(urne, n, replace = T)
   vectprop[i] <- sum(reechant <= kobs)/n
               }
 quantiles <- quantile(vectprop, probs = c(2.5,5,25,50,75,95,97.5)/100)
 if(ppop >= quantile(vectprop, probs = (1 - proba)/2) &
    ppop <= quantile(vectprop, probs = (1 -(1 - proba)/2)))
    {hypothese <- "Acceptée"} else {hypothese <- "Refusée"}
 infprop <- floor(min(vectprop)*100)/100 ; supprop <- ceiling(max(vectprop)*100)/100
 etendprop <- (supprop - infprop)/nbclass
 histprop <- hist(vectprop, breaks=seq(infprop, supprop, etendprop),right=F, plot = F)
#****************************** Affichage des résultats **********************************
 cat("\nTableau des quantiles\n")
 print(quantiles)
 cat("\n La proportion dans l'échantillon observé est :",kobs/n, "\n",
    "La décision est : l'hypothèse ppop =", ppop, "est", hypothese, "\n")
 par(mfrow = c(2, 1))
 plot(histprop, xlab = paste("Proportion de succès -- Échantillons de taille", n),
    ylab = "Effectifs",
    main = "Distribution simulée d'une variable proportion de succès")
 plot(histprop$breaks, ecdf(vectprop)(histprop$breaks), type ="l",
    xlab = paste("Proportion de succès -- Échantillons de taille", n),
    ylab = "Fréquences cumulées",
    main = "Répartition simulée d'une variable proportion de succès")
 abline(h = c(.025, .05, .25, .5, .75, .95, .975))
}
```

```
ICpropboot()
Tableau des quantiles
2.5%  5%   25%  50%  75%  95%   97.5%
0.32 0.34 0.39 0.43 0.46 0.51 0.52025
La proportion dans l'échantillon observé est : 0.43
La décision est : l'hypothèse ppop = 0.52 est Acceptée
```



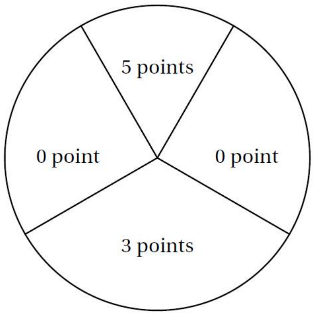

```
ICpropboot(proba = .90)# Pour un test unilatéral
Tableau des quantiles
  2.5%     5%     25%     50%     75%     95%  97.5%
0.3300 0.3495 0.3900 0.4300 0.4600 0.5100 0.5203
 La proportion dans l'échantillon observé est : 0.43
 La décision est : l'hypothèse ppop = 0.52 est Refusée
```



2° Intervalle de confiance rééchantillonné d'une moyenne (BTS)

E – ÉNONCÉS DES SUJETS DE BAC ÉTUDIÉS

1 / 2011-S-Avril-Pondichéry : fléchettes, Épreuves successives répétées à trois issues



Un jeu consiste à lancer des fléchettes sur une cible. La cible est partagée en quatre secteurs, comme indiqué sur la figure ci-contre :
On suppose que les lancers sont indépendants et que le joueur touche la cible à tous les coups.
***1**. Le joueur lance une fléchette.*
On note p_0 la probabilité d'obtenir 0 point.
On note p_3 la probabilité d'obtenir 3 points.
On note p_5 la probabilité d'obtenir 5 points.
On a donc $p_0 + p_3 + p_5 = 1$. Sachant que $p_5 = p_3 / 2$ et que $p_5 = p_0 / 3$déterminer les valeurs de p_0, p_3 et p_5.
***2.** Une partie de ce jeu consiste à lancer trois fléchettes au maximum. Le joueur gagne la partie s'il obtient un total (pour les 3 lancers) supérieur ou égal à 8 points. Si au bout de 2 lancers, il a un total supérieur ou égal à 8 points, il ne lance pas la troisième fléchette.*
On note G2 l'évènement : « le joueur gagne la partie en 2 lancers ».
On note G3 l'évènement : « le joueur gagne la partie en 3 lancers ».
On note P l'évènement : « le joueur perd la partie ».
On note p(A) la probabilité d'un évènement A.
***a.** Montrer, en utilisant un arbre pondéré, que p (G2) =5/36.*
On admettra dans la suite que p (G3) = 7/36.
***b.** En déduire p(P).*
***3.** Un joueur joue six parties avec les règles données à la question 2.*
Quelle est la probabilité qu'il gagne au moins une partie ?
***4.** Pour une partie, la mise est fixée à 2 €. Si le joueur gagne en deux lancers, il reçoit 5 €. S'il gagne en trois lancers, il reçoit 3 €. S'il perd, il ne reçoit rien.*
On note X la variable aléatoire correspondant au gain algébrique du joueur pour une partie. Les valeurs possibles pour X sont donc : −2, 1 et 3.
***a.** Donner la loi de probabilité de X.*
***b.** Déterminer l'espérance mathématique de X. Le jeu est-il favorable au joueur ?*

2 / 2011-S-Mars-Nouvelle Calédonie : villages sport, Probabilités conditionnelles et loi binomiale

Chaque année, deux villages A et B organisent un concours sportif. Les concurrents tirent au sort un moyen de transport puis doivent relier le village A au village B le plus rapidement possible en utilisant ce moyen de transport et un parcours adapté. Pour le tirage, on utilise une urne contenant 4 jetons indiscernables au toucher. Sur un premier jeton figure la lettre V , sur le second la lettre R, sur le troisième la lettre P et sur le dernier la lettre L.
Un concurrent tire au hasard un jeton :
– s'il tire le jeton sur lequel figure la lettre V, il effectuera le trajet à vélo,
– s'il tire le jeton sur lequel figure la lettre R, Il effectuera le trajet en roller,
– s'il tire le jeton sur lequel figure la lettre P, il effectuera le trajet à pied,
– s'il tire le jeton sur lequel figure la lettre L, il choisira librement son mode de transport parmi les trois précédents.
On observe que lorsqu'un concurrent tire le jeton sur lequel figure la lettre L, il choisit le vélo dans 70% des cas, il choisit le roller dans 20% des cas il décide de faire le parcours à pied dans 10% des cas.
***1.** Construire un arbre pondéré correspondant à la situation.*
Pour les questions suivantes, on donnera les résultats arrondis au millième.
***2.** Calculer la probabilité qu'un concurrent effectue le trajet à vélo.*
***3.** Sachant qu'un concurrent a effectué le trajet à vélo, quelle est la probabilité qu'il ait tiré le jeton sur lequel figure la lettre L ?*
***4.** On admet que les résultats des différentes années sont indépendants les uns des autres.*
L'expérience des années précédentes permet de considérer que la probabilité, pour le vainqueur, d'avoir

effectué le trajet à vélo est 2/3.
Calculer la probabilité qu'au cours des six prochaines années l'épreuve soit remportée au moins une fois par un concurrent « non cycliste ».

3 / 2010-S-Novembre-Nouvelle-Calédonie : urne boules, tirages avec et sans remise, probabilités conditionnelles

Une urne contient cinq boules indiscernables au toucher : deux vertes et trois rouges.
***1.** On extrait simultanément et au hasard deux boules de l'urne. On note X la variable aléatoire égale au nombre de boules vertes figurant dans le tirage.*
***a.** Vérifier que P(X = 0) = 3/10, puis déterminer la loi de probabilité de la variable aléatoire X.*
***b.** Calculer l'espérance mathématique de la variable aléatoire X.*
***c.** Calculer la probabilité de l'évènement suivant :*
A : « les deux boules tirées sont de même couleur ».
***2.** On effectue deux tirages successifs d'une boule en respectant la règle suivante : si la boule tirée est rouge, on la remet dans l'urne ; si elle est verte, on ne la remet pas.*
***a.** En utilisant un arbre pondéré, calculer la probabilité des évènements suivants :*
B : « seule la première boule tirée est verte »,
C : « une seule des deux boules tirées est verte ».
***b.** Sachant que l'on a tiré exactement une boule verte, quelle est la probabilité que cette boule verte soit la première tirée ?*

4 / 2010-S-Septembre-Antilles : Bovins malades et test dépistage, probabilités conditionnelles, loi binomiale

Un test est mis au point et essayé sur un échantillon d'animaux dont 1 % est porteur de la maladie. On obtient les résultats suivants :
• si un animal est porteur de la maladie, le test est positif dans 85 % des cas ;
• si un animal est sain, le test est négatif dans 95 % des cas.
***1.** Construire un arbre pondéré modélisant la situation proposée.*
***2** Un animal est choisi au hasard.*
***a.** Quelle est la probabilité qu'il soit porteur de la maladie et que son test soit positif ?*
***b.** Montrer que la probabilité pour que son test soit positif est 0,058.*
***3.** Un animal est choisi au hasard parmi ceux dont le test est positif. Quelle est la probabilité pour qu'il soit porteur de la maladie ?*
***4.** On choisit cinq animaux au hasard. La taille de ce troupeau permet de considérer les épreuves comme indépendantes et d'assimiler les tirages à des tirages avec remise. On note X la variable aléatoire qui, aux cinq animaux choisis, associe le nombre d'animaux ayant un test positif.*
***a.** Quelle est la loi de probabilité suivie par X ?*
***b.** Quelle est la probabilité pour qu'au moins un des cinq animaux ait un test positif ?*