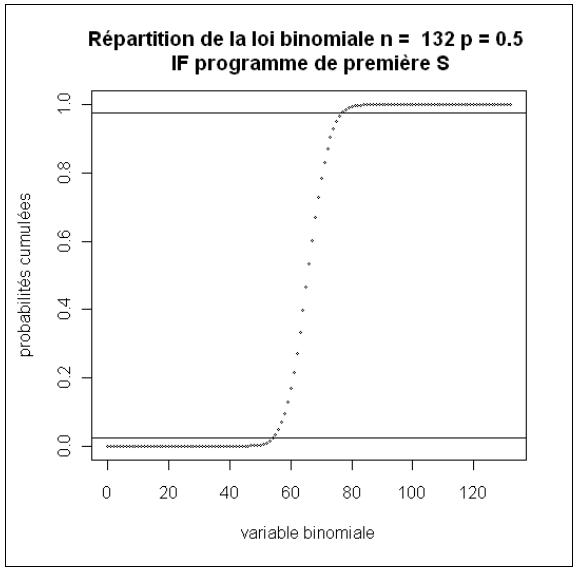
Quelques exemples de tests de conformité d'une proportion

	Paramètres des lois binomiales			Intervalles de Fluctuation seconde première et terminale programmés en langage R : fonction IFexact_1(...)	Test de conformité d'une proportion, dans R, intervalle de confiance d'une proportion
EXEMPLE	**n**	**p**	**k obs / f obs**	La fonction fournit aussi un graphique permettant d'illustrer la détermination de l'IF.	Remarques : Z^2 = X-squared ; **prop.test(...)** est une fonction native de **R**.
CAFÉ	100	0,3	40 / 0,40	`> IFexact_1(n = 100, ppop = .3, kobs = 40, proba = .95)` *L'IF exact de la variable comptage méthode première est :* *[21 , 39] de probabilité : 0.9625486* *L'IF exact de la variable proportion méthode première est :* *[0.21 , 0.39]* *Hypothèse p population = 0.3 ; confrontée à* *f observé = 0.4 : REFUSÉE* *L'IF asymptotique de la variable proportion formule terminale est :* *[0.2101832 , 0.3898168]* *L'IF asymptotique de la variable proportion formule seconde est :* *[0.2 , 0.4]*	`> prop.test(x = 40, n = 100, p = .3, alternative = "two.sided", conf.level = .95, correct = FALSE)` *1-sample proportions test without continuity correction* *data: 40 out of 100, null probability 0.3* *X-squared = 4.7619, df = 1, p-value = 0.0291* *alternative hypothesis:* *true p is not equal to 0.3* *95 percent confidence interval:* *0.3094013 0.4979974* *sample estimates: p = 0.4*
WOBURN	5969	0,0005	9 / 0,0015	`> IFexact_1(n = 5969, ppop = .0005, kobs = 9, proba = .95)` *L'IF exact de la variable comptage méthode première est :* *[0 , 7] de probabilité : 0.9884481* *L'IF exact des proportions méthode première est :* *[0 , 0.001172726]* *Hypothèse p population = 5e-04 ; confrontée à* *f observé = 0.00150779 : REFUSÉE* *L'IF asymptotique de la variable proportion formule terminale est :* *[-6.711835e-05 , 0.001067118]* *L'IF asymptotique de la variable proportion formule seconde est :* *[-0.01244342 , 0.01344342]*	`> prop.test(x = 9, n = 5969, p = .0005, alternative = "greater", conf.level = .95, correct = FALSE)` *1-sample proportions test without continuity correction* *data: 9 out of 5969, null probability 5e-04* *X-squared = 12.1308, df = 1, p-value = 0.000248* *alternative hypothesis:* *true p is greater than 5e-04* *95 percent confidence interval:* *0.0008774265 1.0000000000* *sample estimates: p = 0.00150779*
PARTIDA	870	0,8	339 / 0,3897	`> IFexact_1(n = 870, ppop = .8, kobs = 339, proba = .95)` *L'IF exact de la variable comptage méthode première est :* *[673 , 719] de probabilité : 0.9537416* *L'IF exact de la variable proportion méthode première est :* *[0.7735632 , 0.8264368]* *Hypothèse p population = 0.8 ; confrontée à* *f observé = 0.3896552 : REFUSÉE* *L'IF asymptotique de la variable proportion formule terminale est :* *[0.7734204 , 0.8265796]* *L'IF asymptotique de la variable proportion formule seconde est :* *[0.7660968 , 0.8339032]*	`> prop.test(x = 339, n = 870, p = .8, alternative = "less", conf.level = .95, correct = FALSE)` *1-sample proportions test without continuity correction* *data: 339 out of 870, null probability 0.8* *X-squared = 915.5819, df = 1, p-value < 2.2e-16* *alternative hypothesis: true p is less than 0.8* *95 percent confidence interval:* *0.0000000 0.4171526* *sample estimates: p = 0.3896552*

	Paramètres des lois binomiales			Intervalles de Fluctuation seconde première et terminale programmés en langage R : fonction IFexact_1(...)	Test de conformité d'une proportion, dans R, intervalle de confiance d'une proportion
EXEMPLE	**n**	**p**	**k obs / f obs**	La fonction fournit aussi un graphique permettant d'illustrer la détermination de l'IF.	Remarques : Z^2 = X-squared ; **prop.test(...)** est une fonction native de **R**.

AAMJI-WNANG	132	0,5	46 / 0,3485	`> IFexact_1(n = 132, ppop = .5, kobs = 46, proba = .95)`
				
				`L'IF exact de la variable comptage méthode première est :`
				`[ 55 , 77 ] de probabilité : 0.9551146`
				
				`L'IF exact de la variable proportion méthode première est :`
				`[ 0.4166667 , 0.5833333 ]`
				`Hypothèse p population = 0.5 : confrontée à`
				`f observé = 0.3484848 : REFUSÉE`
				
				`L'IF asymptotique de la variable proportion formule terminale est :`
				`[ 0.4147035 , 0.5852965 ]`
				
				`L'IF asymptotique de la variable proportion formule seconde est :`
				`[ 0.4129612 , 0.5870388 ]`
				
				 
					`> prop.test(x = 46, n = 132, p = .5, alternative = "less", conf.level = .95, correct = FALSE)`
					
					`1-sample proportions test without continuity correction`
					
					`data: 46 out of 132, null probability 0.5`
					`X-squared = 12.1212, df = 1, p-value = 0.0002493`
					`alternative hypothesis: true p is less than 0.5`
					`95 percent confidence interval:`
					`0.0000000 0.4191253`
					`sample estimates: p = 0.3484848`

À la page suivante, le programme **R** de la fonction **IFexact_1(...)** ayant permis d'obtenir les résultats ci-dessus.

Programme R de la fonction IFexact_1(...) ayant permis d'obtenir les résultats ci-dessus.

```
# Cette fonction utilise les modèles des programmes de seconde, première
# et terminale pour calculer 3 Intervalles de Fluctuation d'une variable
# binomiale et de la variable fréquence correspondant.
# Elle propose une décision relative à l'IF exact bilatéral.
# On peut choisir la taille n de l'échantillon, la probabilité ppop de succès,
# le nombre kobs de succès observés dans l'échantillon, la probabilité proba e l'IF.
IFexact_1 = function(n = 100, ppop = .52, kobs = 43, proba = .95){
#****************** IF binomial méthode première ***************
#---Intervalle de fluctuation bilatéral d'une variable
# binomiale X de paramètres n et p et de la variable fréquence X/n,
# de probabilité minimale 1 - e
#--a est la plus petite valeur de X telle que P(X <= a) > (1 - proba)/2)
#--b est la plus petite valeur de X telle que P(X <= b) >= (1 - (1 - proba)/2)
#--[a ; b] est l'intervalle de fluctuation de X de proba. mini. proba
   a <- 0 ; b <- 0
   reparti1 <- pbinom(0:n, n, ppop, lower.tail = TRUE) # pbinom est la répartition bino.
   names(reparti1) <- 0:n
   p <- 0
   while (p <= (1 - proba)/2){
      p <- pbinom(a, n, ppop, lower.tail = TRUE)
      a <- a + 1
   }
   p <- 0
   while (p < (1 - (1 - proba)/2)){
      p <- pbinom(b, n, ppop, lower.tail = TRUE)
      b <- b + 1
   }
   probaab <- sum(dbinom((a - 1):(b - 1), n, ppop)) # dbinom est la distribution bino.
   if (kobs >= (a - 1) & kobs <= (b - 1)) {
      hypothese <- "ACCEPTÉE"
   } else {
      hypothese <- "REFUSÉE"
   }
#*************** deux IF asymptotiques méthodes seconde et terminale ****
   asympt2 <- ppop - 1 / sqrt(n) ; bsympt2 <- ppop + 1 / sqrt(n)
   c1 <- qnorm((1 - (1 - proba)/2), 0, 1) * sqrt(ppop * (1 - ppop)/n)
   asympt1 <- ppop - c1; bsympt1 <- ppop + c1
#************* Affichage des résultats et des graphiques***************
   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 du programme de première S"))
     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 de la variable comptage méthode première est :\n[",
     a - 1,",",b - 1,"] de probabilité :", probaab,
     "\n\nL'IF exact de la variable proportion méthode première est :\n[",
     (a - 1)/n,",",(b - 1)/n,"]\n",
     "Hypothèse p population = ", ppop, "; confrontée à f observé =",kobs/n,
     " : ",hypothese,"\n")
   cat("\nL'IF asymptotique de la variable proportion formule terminale est :\n[",
      asympt1,",",bsympt1,"]\n")
   cat("\nL'IF asymptotique de la variable proportion formule seconde est :\n[",
      asympt2,",",bsympt2,"]\n")
}
```

La fonction **IFexact_2(...)** est une variante du calcul de l'IF "exact" (définition du programme de première) qui n'utilise pas de boucle while, mais une recherche dans des listes indicées, beaucoup plus rapide[1].

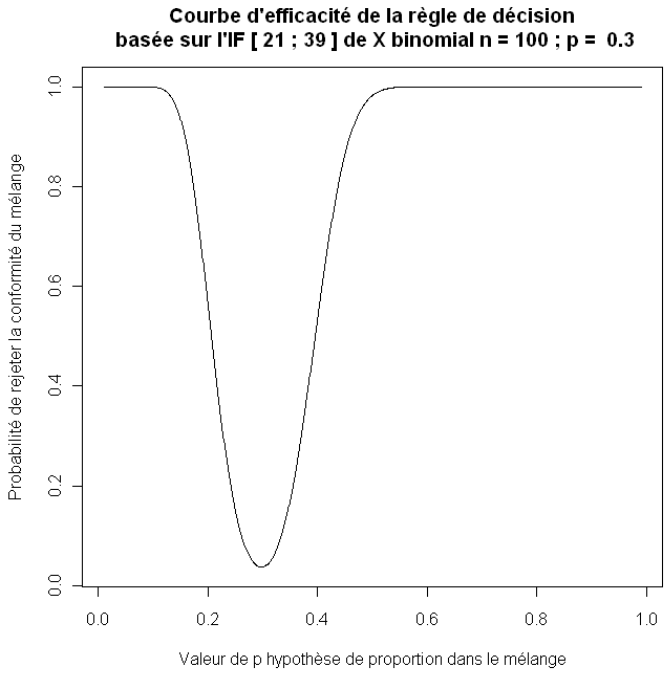
```
#-Algo et programme IFexact_2() Des opérations sur une liste indicée
# remplacent while
IFexact_2 <- function(n = 10, ppop = .7, proba = .95){
   repartX <- pbinom(0:n, n, ppop)
   rang_a <- min(which(repartX > (1 – proba)/2)) # which renvoie les indices
   a <- rang_a - 1
   rang_b <- min(which(repartX >= (1 - (1 - proba)/2)))
   b <- rang_b - 1
# Affichage des résultats
   if (a != 0) {cat("P(X <=", a - 1, ")=", repartX[rang_a - 1], "\n")
               cat("P(X <=", a, ")=", repartX[rang_a], "\n\n")
   } else {cat("P(X <=", a, ")=", repartX[rang_a], "\n\n")
   }
   cat("P(X <=", b - 1, ")=", repartX[rang_b - 1], "\n")
   cat("P(X <=", b, ")=", repartX[rang_b], "\n\n")
   cat("Avec une valeur nominale de probabilité de", proba, "\n")
   cat("L'intervalle de fluctuation de X est :[", a,";", b, "]\n")
   cat("L'intervalle de fluctuation de X/n est :[",
       a / n,";", b / n, "]\n")
   cat("Sa probabilité réelle est de", sum(dbinom((a:b), n, ppop)), "\n\n")
}

> IFexact_2(n = 100, ppop = .3, proba = .95)
P(X <= 20 )= 0.01646285 ------ P(X <= 21 )= 0.02883125

P(X <= 38 )= 0.966021 ------ P(X <= 39 )= 0.9790114

Avec une valeur nominale de probabilité de 0.95
L'intervalle de fluctuation de X est :[ 21 ; 39 ]
L'intervalle de fluctuation de X/n est :[ 0.21 ; 0.39 ]
Sa probabilité réelle est de 0.9625486

> IFexact_2(n = 132, ppop = .5, proba = .95)
P(X <= 54 )= 0.0224427 ----- P(X <= 55 )= 0.03358519

P(X <= 76 )= 0.9664148 ----- P(X <= 77 )= 0.9775573

Avec une valeur nominale de probabilité de 0.95
L'intervalle de fluctuation de X est :[ 55 ; 77 ]
L'intervalle de fluctuation de X/n est :[ 0.4166667 ; 0.5833333 ]
Sa probabilité réelle est de 0.9551146
```

1Un indéniable avantage pédagogique de cet algorithme c'est que c'est l'application on ne peut plus directe de la définition de l'IF du programme de première. Alors que la transposition didactique de l'IF à la boucle while est source de difficultés pour les élèves.

La fonction EfficaciteIF(...) détermine l'IF de probabilité proba, d'une variable binomiale X de paramètres n et ppop.
Cette fonction trace la courbe d'efficacité de la règle de décision basée sur cet IF. Cette courbe d'efficacité représente la probabilité de rejeter la conformité du mélange en fonction des valeurs de ppop dans]0 ; 1[.

```
#**COURBE D'EFFICACITÉ D'UNE RÈGLE DE DÉCISION BASÉE SUR L'IF BINOMIAL**
# L'algorithme de l'IF est basé sur des opérations sur une liste indicée
# qui remplacent while
EfficaciteIF <- function(n = 100, ppop = .3, proba = .95){
   vectp <- seq(.01, .99, length.out = 1000) ; vecteffi <- NULL
   repartX <- pbinom(0:n, n, ppop)
   rang_a <- min(which(repartX > (1 - proba)/2))
   a <- rang_a - 1
   rang_b <- min(which(repartX >= (1 - (1 - proba)/2)))
   b <- rang_b - 1
   for (i in 1:1000) {
      ProbaRejet <- sum(dbinom(0:(a - 1), n, vectp[i])) +
                    sum(dbinom((b + 1):n, n, vectp[i]))
      vecteffi <- c(vecteffi, ProbaRejet)
   }
# Affichage des résultats et graphiques
   plot(vectp, vecteffi, type = "l",
      xlab = "Valeur de p hypothèse de proportion dans le mélange",
      ylab = "Probabilité de rejeter la conformité du mélange",
      main = paste("Courbe d'efficacité de la règle de décision\n",
         "basée sur l'IF [", a, ";", b, "] de X binomial n =", n, "; p = ",ppop))
   cat("Avec une valeur nominale de probabilité de", proba, "\n")
   cat("L'intervalle de fluctuation de X est :[", a,";", b, "]\n")
   cat("L'intervalle de fluctuation de X/n est :[",
       a / n,";", b / n, "]\n")
   cat("Sa probabilité réelle est de", sum(dbinom((a:b), n, ppop)), "\n\n")
}
```

```
EfficaciteIF(n = 100, ppop = .3, proba = .95)
```




```
EfficaciteIF(n = 500, ppop = .3, proba = .95)
```