

Exemple d'analyse de tableaux de contingence (chapitre 6)

----- Référence

Benson H, Dusek JA, Sherwood JB, Lam P, Bethea CF, Carpenter W, Levitsky S, Hill PC, Clem Jr. DW, Jain MK, Drumel D, Kopecky SL, Mueller PS, Marek D, Rollins S, Hibberd PL. 2006. Study of the Therapeutic Effects of Intercessory Prayer (STEP) in cardiac bypass patients: A multicenter randomized trial of uncertainty and certainty of receiving intercessory prayer. American Heart Journal 151: 934-942.

```
-----  
prayer=read.table(file.choose())  
prayer  
      Complications Non    n      # n = somme de chaque ligne  
Peut-etre/Oui      315 289 604  
Peut-etre/Non      304 293 597  
Certain            352 249 601
```

```
-----  
prayer = matrix(c(315,304,352,289,293,249), 3, 2)  
rownames(prayer) = c("Peut-etre/Oui", "Peut-etre/Non", "Certain")  
colnames(prayer) = c("Complications", "Non")  
-----
```

```
prayer  
      Complications Non  
Peut-etre/Oui      315 289  
Peut-etre/Non      304 293  
Certain            352 249
```

Test paramétrique de khi-carré

```
res = chisq.test(prayer[,1:2])  
res
```

Pearson's Chi-squared test

```
data:  prayer[, 1:2]  
X-squared = 8.1465, df = 2, p-value = 0.01702
```

Test par permutations

```
res.perm = chisq.test(prayer[,1:2], simulate.p.value = TRUE, B=9999)  
res.perm
```

Pearson's Chi-squared test with simulated p-value (based on 9999 replicates)

```
data: prayer[, 1:2]
X-squared = 8.1465, df = NA, p-value = 0.0181
```

```
# Calcul du tableau de statistiques de Freeman-Tukey (eq. 6.28, p. 244)
```

```
summary(res)
```

```
Obs = res$observed
```

```
E = res$expected
```

```
FT = Obs^0.5 + (Obs + 1)^0.5 - (4*E + 1)^0.5
```

```
FT
```

| | Complications | Non |
|---------------|---------------|------------|
| Peut-etre/Oui | -0.5704030 | 0.6355438 |
| Peut-etre/Non | -0.9855782 | 1.0637251 |
| Certain | 1.5446087 | -1.7197601 |

```
# Criterion (equation 6.29, Numerical ecology p. 244)
```

```
d.freedom = 2
```

```
n.cells = 6
```

```
chi.ref = qchisq(0.05/n.cells, df=1, lower.tail=FALSE)
```

```
# Valeur critique selon Sokal & Rohlf (1995)
```

```
SR.critique = sqrt(d.freedom * chi.ref / n.cells)
```

```
SR.critique
```

```
[1] 1.523199
```

```
# Y a-t-il des valeurs dans la matrice FT > que SR.critique en valeur  
absolue?
```

```
# Conclusion: pour le groupe de patients qui étaient certains qu'on  
prierait pour eux, le nombre de complications dépasse de façon  
significative la valeur attendue sous H0. Dans ce même groupe, le nombre  
de patients sans complications est aussi significativement inférieur à la  
valeur prévue sous H0.
```

```
# Test des résidus centrés-réduits (eq. 6.30, Numerical ecology p. 244)
```

```
Obs = res$observed
```

```
E = res$expected
```

```
r = 3
```

```
c = 2
```

```
r.marg = matrix(apply(Obs,1,sum), r, 1)
c.marg = matrix(apply(Obs,2,sum), 1, c)
N = sum(prayer[,1:2])
#
# Standardized residuals (eq. 6.26)
st.res = (Obs - E)/sqrt(E)
# Variances of the standardized residuals
var.cell = (1-r.marg/N) %*% (1-c.marg/N)
# Adjusted residuals
adj.res.z = st.res/sqrt(var.cell)

adj.res.z
      Complications      Non
Peut-etre/Oui      -1.047427  1.047427
Peut-etre/Non      -1.776198  1.776198
Certain             2.821948 -2.821948

# Equation 6.30, test unilatéral dans la queue de droite

crit.z = qnorm(0.05/(2*6), lower.tail=FALSE)
crit.z
[1] 2.638257

# Autre formule valide: Z.ref=qnorm((1 - (0.05/(2*6))), lower.tail=TRUE)

# Puissance comparable au test précédent

=====

# Analyse de tableau de contingence, statistique khi-carré de Pearson:

chisq.test

# On fournit soit deux vecteurs, soit un tableau de contingence.
# En option, on peut obtenir un test par permutation.

# Tableau de contingence du manuel, Table 6.1

Table6.1 = matrix(c(30,0,0,0,10,20,0,0,15,0,0,15,5,10,15,0),4,4)
chisq.res = chisq.test(Table6.1)
chisq.test(Table6.1, simulate.p.value=TRUE, B=9999)

# Vérification: calcul du khi-carré de Wilks par l'entropie
# Attention: log(x), log10(x), log2(x)
?log

# Calculs en log base 2
```

```
H.a = -(0.5*log2(0.5) + 0.25*log2(0.25) + 2*(0.125*log2(0.125)))  
H.b = -(4*(0.25*log2(0.25)))  
H.ab= -(0.25*log2(0.25) + (1/6)*log2(1/6) + 3*(0.125*log2(0.125)) +  
2*((1/12)*log2(1/12)) + (1/24)*log2(1/24))
```

H.a

H.b

H.ab

B = H.a + H.b - H.ab

B

n = 120

khi.carre = 2 * n * B * log(2)

khi.carre

```
# Calcul de S(a,b), eq. 6.15
```

A = H.a - B

A

C = H.b - B

C

S.ab = B/(A+B+C)

S.ab

S.ab = B/H.ab

S.ab

Obs = chisq.res\$observed

E = chisq.res\$expected

```
# Exclusion des valeurs Obs == 0 qui se trouvent dans le tableau Obs
```

GT0 = which(Obs > 0)

wilks.X2 = 2*sum(Obs[GT0] * log(Obs[GT0]/E[GT0]))

wilks.X2

[1] 150.6579 # en log naturels

```
# Calcul de wilks.X2 à partir de B en bits (eq. 6.14)
```

wilks.X2.B = 2 * 120 * B * log(2)

wilks.X2.B

[1] 150.6579 # en log naturels

```
# Calcul de B en log naturels
```

H.ae = -(0.5*log(0.5) + 0.25*log(0.25) + 2*(0.125*log(0.125)))

H.be = -(4*(0.25*log(0.25)))

H.ae= -(0.25*log(0.25) + (1/6)*log(1/6) + 3*(0.125*log(0.125)) +
2*((1/12)*log(1/12)) + (1/24)*log(1/24))

Be = H.ae + H.be - H.ae

Be

[1] 0.6277412

```
# Calcul de wilks.X2 à partir de B en log naturels (eq. 6.13)
wilks.X2.Be = 2 * 120 * Be
wilks.X2.Be
[1] 150.6579
```

=====

```
# Calculer Be à partir de B en bits
Be2 = B*log(2)
Be2
[1] 0.6277412
```

```
# ou encore
Be2 = B/log2(exp(1))
Be2
[1] 0.6277412
```

=====