ASI (L2) : TP10

Objectifs du TP :

Approfondir la programmation sous *Excel* et sous *Rstat*.

1. Programmation soutenue sous *Excel*

1.1 Modification de macros

Modifier le fichier-modèle `ASGQT.XLT` pour qu'il utilise comme borne de corrélation sûre la valeur de la case `F2` plutôt que la valeur fixe 0.9 et pour qu'il affiche les relations linéaires pour les plus fortes corrélations négatives aussi.

On testera les nouvelles macros avec le dossier `OLYMPIC`.

Comment faire pour que dans la liste des macros via `ALT-F8` une seule macro soit affichée à savoir le programme principal ?

1.2 Création d'un nouvelle macro

On voudrait calculer le χ^2 d'indépendance d'un tableau de données et le comparer au χ^2 de la table. Pour les plus fort(e)s, la macro affichera les contributions triées (y compris le signe de contribution) en fonction de la valeur théorique du χ^2.

On utilisera le tableau du fichier `LIVRES.DAT` afin de tester les résultats affichés en TD.

2. Programmation soutenue sous *Rstat*

2.1 Ajouts dans `statgh.r`

Ecrire une fonction `allQL` qui réalise l'appel de `triAplat` pour toutes les variables d'une matrice de données (comme pour le dossier `PBIO`).

On ajoutera cette fonction à `statgh.r` ; on s'arrangera pour utiliser une matrice des noms de modalité, la ligne i de cette matrice correspondant à la variable i.

En cours de programmation, on pourra se limiter aux trois premières `QL`.

Comment en déduire le tableau récapitulatif des QL trié en ligne par mode où chaque ligne est triée par pourcentage décroissant ?

2.2 Création d'un nouveau programme

On voudrait calculer le χ^2 d'indépendance d'un tableau de contingence passé en paramètre. Le programme devra afficher les contributions triées (y compris le signe de contribution) et il devra aussi donner la valeur théorique du χ^2 ainsi que la décision par rapport au test.

On utilisera le tableau du fichier `LIVRES.DAT` afin de tester les résultats affichés en TD.

Utiliser ce programme pour étudier l'indépendance entre `SEXE` et `ETUD` pour le dossier ELF.

Esquisse de SOLUTION

Modification de macros sous *Excel*

Pour changer la gestion des meilleurs ρ, il faut remplacer au début du sous-programme *Coeff* la valeur `0.9` par `bornSur` définie par

```
bornSur = Sheets("Meilleures correlations").Cells(2, 6)
```

Il faut ensuite le changer le test de la boucle tant que et ajouter une variable pour savoir où afficher les meilleures corrélations (c'est la variable *posCorr*).

Voici donc le nouveau début du sous-programme *Coeff* :

```
Sheets("Meilleures correlations").Select
bornSur = Sheets("Meilleures correlations").Cells(2, 6).Value
Let k = 0
Let posCorr = 7
Let corr = Cells(7 + k, 4).Value
While k < col * (col - 1) / 2
  If Abs(corr) > bornSur Then
```

On peut alors mettre les valeurs de correlation à la position *posCorr* :

```
    Cells(posCorr, 6).Formula = ...
    Cells(posCorr, 7).Formula = ...
    Cells(posCorr, 8).Formula = ...
    Cells(posCorr + 1, 8).Formula = ...
...
    Sheets("Meilleures Correlations").Select
    Let posCorr = posCorr + 2
```

A la fin du test sur ρ, on passe à la ligne suivante (repérée par k) et on met à jour la variable *corr* :

```
  End If
  Sheets("Meilleures Correlations").Select
  Let k = k + 1
  Let corr = Cells(7 + k, 4).Value
Wend
```

Pour n'avoir qu'une seule macro visible dans la liste, il suffit qu'elle n'ait pas de paramètre et que les autres en aient.

Ainsi au lieu de

```
Sub Principal()
    Dimension
    ...
End Sub
Sub Dimension()
...
```

il suffit de mettre

```
Sub AsgQT()
    Dimension col, lig
    ...
End Sub
Sub Dimension(col, lig)
...
```

Création d'un nouvelle macro

Il n'y a aucune difficulté particulière sauf à savoir où aficher les valeurs ! Par contre pour mettre en forme et pour trier, il faut utiliser de nouvelles instructions :

`IsEmpty(Cells(i,j).Value)`	permet de savoir si la cellule est vide,
`Cells(i,j).Font.Bold = True`	met en gras la case en ligne i colonne j,
`Cells(i,j).Font.ColorIndex = 5`	met en bleu la case en ligne i colonne j,
`Cells(i,j).NumberFormat = "###.00"`	force le format de la cellule,
`chiinv(proba,ddl)`	est la fonction qui correspond à `KHIDEUX.INVERSE`,
`Range(Cells(...),Cells(...).Select`	doit être effectué avant de trier.

Pour trier, il faut écrire

```
Selection.Sort
   Key1:=Cells(...), _
   Order1:=xlDescending, Header :=xlGuess, _
   OrderCustom:=1, MatchCase:=False, _
   Orientation:=xlTopToBottom
```

```
'
'  ******************************************************
'  *                                                    *
'  *   CALCUl DU CHI-DEUX D'INDEPENDANCE                *
'  *                                                    *
'  ******************************************************
'
'


Sub chiDeux()
    Dimensions nblig, nbcol         ' détermination du nombre de lignes et de colon
    If (nblig = 0) Or (nbcol = 0) Then
        MsgBox "Vous n'avez pas de données..."
        Exit Sub
    End If
    Titres nblig, nbcol             ' noms de ligne et de colonne
    Marges nblig, nbcol, totg       ' totaux en ligne et en colonne
    ValInd nblig, nbcol, totg       ' valeurs sous hypothèse d'indépendance
    c1 = Calchi(nblig, nbcol)       ' calcul de la distance du chi-deux
    c2 = Chith(nblig, nbcol)        ' valeur du chi-deux théorique
    If c1 > c2 Then
        TabContri nblig, nbcol      ' contributions dans le tableau
    End If
    ' on arrange le coin supérieur gauche
    Cells(1, 1).Select
    Cells(1, 1).Value = "Données"
    Cells(1, 1).Font.Bold = True
End Sub

'
' Détermination du nombre de lignes et de colonnes
'

Sub Dimensions(nblig, nbcol)

    Let lig = 2
    While IsEmpty(Cells(lig, 1).Value) = False
        Let lig = lig + 1
    Wend
    Let nblig = lig - 2
```

```
    Let col = 2
    While IsEmpty(Cells(1, col).Value) = False
        Let col = col + 1
    Wend
    Let nbcol = col - 2

End Sub

'
' Recopie des noms de ligne et de colonne
'

Sub Titres(nblig, nbcol)

    ' on laisse une colonne vide entre les données
    ' originales et le tableau des contributions
    col = nbcol + 3
    For i = 1 To nblig
       Cells(i + 1, col).Value = "  " & Cells(i + 1, 1).Value
       ' on met en gras et en bleu
       Cells(i + 1, col).Font.Bold = True
       Cells(i + 1, col).Font.ColorIndex = 5
    Next i
    For j = 1 To nbcol
       Cells(1, j + col).Value = "  " & Cells(1, 1 + j).Value
       ' on met en gras et en rouge
       Cells(1, j + col).Font.Bold = True
       Cells(1, j + col).Font.ColorIndex = 3
    Next j
    ' avec un titre, c'est mieux
    Cells(1, nblig + 3).Value = "Théoriques"
    Cells(1, nblig + 3).Font.Bold = True
End Sub

'
' Calcul des totaux en lignes et en colonnes
' et du total général
'

Sub Marges(nblig, nbcol, totg)
```

```
    ' initialisation du total général
    totg = 0
    ' totaux en ligne
    For i = 1 To nblig
        somc = 0
        For j = 1 To nbcol
            somc = somc + Cells(i + 1, 1 + j).Value
        Next j
        Cells(i + 1, 2 * nbcol + 4).Value = somc
    Next i
    ' totaux en colonne et total gnéral
    For j = 1 To nbcol
        soml = 0
        For i = 1 To nblig
            soml = soml + Cells(i + 1, 1 + j).Value
        Next i
        Cells(nblig + 2, nbcol + 3 + j).Value = soml
        totg = totg + soml
    Next j
    Cells(nblig + 2, 2 * nbcol + 4).Value = totg
    ' on met en gars et en vert foncé
    Cells(nblig + 2, 2 * nbcol + 4).Font.Bold = True
    Cells(nblig + 2, 2 * nbcol + 4).Font.Color = RGB(0, 150, 0)
End Sub

'
' calcul des valeurs théoriques sous hypothèse d'indépendance
'

Sub ValInd(nblig, nbcol, totg)
    col = nbcol + 3
    For i = 1 To nblig
        For j = 1 To nbcol
            totlig = Cells(i + 1, 2 * nbcol + 4).Value
            totcol = Cells(nblig + 2, j + col).Value
            contri = totlig * totcol / totg
            Cells(i + 1, j + col).Value = contri
            Cells(i + 1, j + col).NumberFormat = "###.00"
        Next j
    Next i
End Sub
```

```
'
' Calcul de la distance du chideux
'

Function Calchi(nblig, nbcol)
    Cells(nblig + 3, 1).Value = " Distance chi-deux"
    vchi = 0
    col = nbcol + 3
    For i = 1 To nblig
        For j = 1 To nbcol
            obs = Cells(i + 1, j + 1).Value
            th = Cells(i + 1, j + col).Value
            dif = obs - th
            vchi = vchi + dif * dif / th
        Next j
    Next i
    Cells(nblig + 3, 3).Value = vchi
    Cells(nblig + 3, 3).NumberFormat = "###.000"
    Calchi = vchi
End Function

'
' Valeur théorique du chi-deux
'

Function Chith(nblig, nbcol)
    Cells(nblig + 4, 1).Value = " Chi-deux table "
    vchi = "=chiinv(0.05," & (nblig - 1) * (nbcol - 1) & ")"
    Cells(nblig + 4, 3).Formula = vchi

    ' autre solution avec la fonction en français:
    '
    '    vchi = "=KHIDEUX.INVERSE(0,05;" & (nblig - 1) * (nbcol - 1) & ")"
    '    Cells(nblig + 4, 3).FormulaLocal = vchi

    Cells(nblig + 4, 3).NumberFormat = "####.000"
    Chith = Cells(nblig + 4, 3).Value
End Function
```

```
'
' Affichage trié des contributions signées
'

Sub TabContri(nblig, nbcol)
    lig = nblig + 6
    Cells(lig, 1).Value = "Contributions"
    numlig = lig
    col = nbcol + 3
    Let n = nblig * nbcol
    For i = 1 To nblig
        For j = 1 To nbcol
            numlig = numlig + 1
            obs = Cells(i + 1, j + 1).Value
            th = Cells(i + 1, j + col).Value
            dif = obs - th
            vchi = dif * dif / th
            Cells(numlig, 2).Value = "   -"
            If dif > 0 Then
               Cells(numlig, 2).Value = "   +"
            End If
            ' ??? Cells(numlig, 2).Format.Align = "Right"
            Cells(numlig, 3).Value = vchi
            Cells(numlig, 3).NumberFormat = "###.000"
            Cells(numlig, 4).Value = "  " & Cells(i + 1, 1).Value
            Cells(numlig, 5).Value = "  " & Cells(1, 1 + j).Value
        Next j
    Next i

 ' et on trie...

 Range(Cells(nblig + 7, 2), Cells(nblig + 7 + n, 5)).Select
 Selection.Sort Key1:=Cells(nblig + 7, 3), Order1:=xlDescending, Header _
   :=xlGuess, OrderCustom:=1, MatchCase:=False, _
   Orientation:=xlTopToBottom

End Sub
```

Programmation soutenue sous *Rstat*

2.3 Ajouts dans `statgh.r`

Tout d'abord, voici comment on voudrait utiliser les nouvelles fonctions :

```
# lecture des données
  pbio <- read.dbf("pbio.dbf")
  pbioDATA <- pbio$dbf

# préparation des variables

  pbioCOLQL <- 2:12

  pbioQL    <- matrix(nrow=length(pbioCOLQL),ncol=2)
  # en plus court (pour le récapitulatif)
  pbioQLc   <- matrix(nrow=length(pbioCOLQL),ncol=2)
  pbioQL[1,1] <- c("Connaissez-vous les produits biologiques ?")
  pbioQLc[1,1] <- c("CONNAISS")
  lesm <- paste(paste(c("non réponse","oui","non"),"!"),collapse="")
  pbioQL[1,2] <- lesm
  pbioQLc[1,2] <- lesm
  ...

# appel des sous-programmes
  allQL(pbioDATA,pbioQL,pbioCOLQL)
  recapQL(pbioDATA,pbioQLc,pbioCOLQL)
```

On commence donc la fonction allQL par

```
allQL <- function(tdata,tmoda,numCol) {

   # - le première paramètre est une matrice de données,
   # - le deuxième est une matrice des modalités,
   #   en colonne 1 le titre,
   #   en colonne 2 la liste des modalités séparées par !,
   # - le troisième est le vecteur des numéros des colonnes
   #   dans la matrice des données
```

Puisqu'on veut appeler `triCroise` il suffit d'utiliser une boucle pour sur chacun des éléments du vecteur des numéros de colonne, d'où

```
########################################
#
# allQL effectue tous les tris à plat
#
########################################

allQL <- function(tdata,tmoda,numCol) {

 nc <- length(numCol)
 for (i in 1:nc) {
   numc <- numCol[i]
   varc <- tdata[,numc]
   titc <- tmoda[i,1]
   modc <- unlist(strsplit(tmoda[i,2],"!"))
   triAplat(titc,varc,modc)
 } ; # fin pour i

} ; # fin de fonction allQL
```

Pour le tableau récapitulatif des QL, c'est un peu plus compliqué. Dans un premier temps, on construit un tableau à deux colonnes qui contient en colonne 1 le numéro de chaque variable et en colonne 2 le mode pour cette variable.

On calcule ensuite l'ordre de rangement (index) des variables en fonction du mode. Enfin, pour chaque variable on effectue un tri à plat puis un tri des effectifs avant de n'afficher que les trois premiers effectifs (s'il n'y a que deux effectifs, on met la chaine vide pour le troisième effectif).

D'où le programme :

```
#######################################
#
# recapQL donne un tableau trié de tous les tris à plat
#
#######################################

recapQL <- function(tdata,tmoda,numCol) {

 cat("\n Tableau récapitulatif des variables qualitatives\n")
 nc <- length(numCol)

 # on construit le tableau des numéros de variable et mode

 tmodeQ <- matrix(nrow=nc,ncol=2)
 for (i in 1:nc) {
   numc <- numCol[i]
   varc <- tdata[,numc]
   titc <- tmoda[i,1]
   # calcul du tri à plat
   tap  <- as.integer(table(varc))
   # en colonne 1, on met le numéro initial de la variable
   tmodeQ[i,1] <- i
   # le mode est bien sur l'effectif maximal :
   tmodeQ[i,2] <- max(tap)
 } ; # fin pour i

 # calcul de l'ordre de rangement (index) des colonnes

 idc <- order(tmodeQ[,2],decreasing=TRUE)

 # on peut alors construire le tableau récapitulatif
 # dans le bon ordre

 ncr <- 7
 trecap <- matrix(nrow=nc,ncol=ncr)
 for (i in 1:nc) {
   j      <- idc[i]
   numc   <- numCol[j]
   varc   <- tdata[,numc]
```

```
   # calcul du tri à plat

   tap    <- as.integer(table(varc))
   lemode <- round(100*tmodeQ[j,2]/sum(tap))

   # on ordonne les effectifs

   idx    <- order(tap,decreasing=TRUE)

   modc <- unlist(strsplit(tmoda[j,2],"!"))

   # on met les deux plus forts effectifs

   trecap[i,1] <- tmoda[j,1]
   trecap[i,2] <- paste(formatC(lemode,format="f",width=3,dig=0)," %")
   trecap[i,3] <- modc[idx[1]]
   valc        <- round(100*tap[idx[2]]/sum(tap))
   trecap[i,4] <- paste(formatC(valc,format="f",width=3,dig=0)," %")
   trecap[i,5] <- modc[idx[2]]

   # ajout d'une troisième plus forte modalité si elle existe

   if (length(idx)>2) {
      valc        <- round(100*tap[idx[3]]/sum(tap))
      trecap[i,6] <- paste(formatC(valc,format="f",width=3,dig=0)," %")
      trecap[i,7] <- modc[idx[3]]
   } else {
      trecap[i,6] <- ""
      trecap[i,7] <- ""
   } ; # fin si
 } ; # fin pour i

 colnames(trecap) <- rep(" ",ncr)
 rownames(trecap) <- rep(" ",nc)
 print.matrix(trecap,quote=FALSE)

} ; # fin de fonction recapQL
```