ASI (L2) : TP4 Programmation sous *Rstat* et *Excel*

Attention *RSTAT* 2.0 est disponible via Rgui

(on le vérfie avec `factorial(5)` ; montrer `Edit/GUI Preferences`.)

Objectifs du TP :

Savoir programmer des fonctions et des boucles sous *Rstat* ; savoir bien gérer les relations de récurence sous Excel et faire des boucles et des fonctions avec *Excel*.

1. Programmation avec *Rstat*

1.1 Fonctions rapides

Rappeler comment on écrit une fonction sous *Rstat*. Faire des OEP ("one-expression programs") c'est à dire des programmes en une seule expression comme pour

- $carre(x)$	renvoie le carré de l'argument,
- $sdpe(n)$	renvoie la somme des n premiers entiers,
- $duree(j, m, a)$	secondes écoulées entre an 0 et jour j mois m an a,
- $decrit(v)$	renvoie longueur, moyenne et écart-type de v,
- $binom1(n, p)$	affiche les valeurs de $\mathcal{B}(n, p)$
- $poiss1(m, k)$	affiche les k premières valeurs de $\mathcal{P}(m)$

Tester ce que fait *carre* pour un nombre, un vecteur, une matrice. Afficher avec $binom1$ et $poiss1$ juste le vecteur des valeurs de la loi.

Utiliser $n = 5$, $p = 0.3$ et $\lambda = 1.5$ pour $k = 10$ termes.

Ces fonctions ne font aucun test, aucune boucle. Elles se contentent d'appeler des fonctions de *Rstat*.

1.2 Fonctions soignées

Pour ce qui suit, on fera utiliser *Notepad* pour écrire les fonctions et les charger par `source()` ou via le menu "File/Source". On utilisera `cat` pour afficher avec `\n` pour sauter une ligne.

Reprendre *binom*1 et *poiss*1 pour en faire *binom*2 et *poiss*2 qui donnent un titre, qui affichent un beau tableau des valeurs x_i, des valeurs p_i, du cumul c_i et qui pour un troisième paramètre (facultatif) nommé `et` comme "Effectif Total" donnent en entier les effectifs associés à chaque x_i. On se servira de la page Web

`http://www.info.univ-angers.fr/pub/gh/vitrine/Democgi/loisStatp.htm`

comme modèle. Ces fonctions ne font que définir une matrice et la remplir via les bonnes fonctions (aucun test, aucune boucle). Prendre `et` = 150.

Rappeler ce qu'est un test en `si` en algorithmique et donner la traduction en *Rstat*. Reprendre *binom*2 pour en faire *binom*3 qui refuse de calculer si n n'est pas un entier, si p n'est pas un réel compris entre 0 et 1 sachant que *Rstat* dispose des fonctions *as.integer*, *is.numeric* que `non` se note `!` et que `et` se note `&`.

Rappeler ce qu'est une boucle `pour` en algorithmique et donner la traduction en *Rstat*. Ecrire un programme *imax* qui donne le maximum et son indice pour un vecteur passé en paramètre. Insister sur les variables locales, donner un algorithme rapide. Tester avec `c(1,2,8,3,8,5)`.

Rappeler ce qu'est une boucle `tant que` en algorithmique et donner la traduction en *Rstat*. Ecrire un programme *asprite* qui donne afiche les valeurs de n et α_n tant que $|\alpha_n - 1/3| > 10^{-5}$ où α_n correspond à la position du sprite.

On redonne

$$\alpha_n = \frac{1}{3} + \frac{2}{3}\left(\frac{-1}{2}\right)^n$$

2. Programmation avec *Excel*

2.1 Fonctions rapides

Montrer comment on écrit une fonction sous *Excel* à l'aide de

`Outils / Macros / Visual Basic Editor`

pour faire des ORF ("one-result functions") c'est à dire des fonctions qui renvoient une seule valeur comme pour

- *carre*(x) renvoie le carré de l'argument,
- *duree*(a) renvoie nb de secondes écoulées entre an 0 et an a,

Remarque : A la fac, il faut sans doute faire

`Insertion / Module`

avant de pouvoir écrire le texte de la fonction.

Leur faire écrire les deux textes suivants :

```
Function carre(n)
' donne le carré d'un nombre, d'une case...
carre = n * n
End Function

Function duree(a)
' essaie de calculer le nombre de secondes écoulées
' entre la naissance du Christ et l'an fourni
' comme paramètre
duree = 365 * (a - 1) * 12 * 24 * 60 * 60
End Function
```

Tester ce que fait *carre* pour un nombre, une plage de valeurs. Leur faire écrire une fonction *cdv* qui renvoie `ECARTYPEP/MOYENNE` si l'argument est une plage de valeurs.

Ces fonctions ne font aucun test, aucune boucle. Elles se contentent de calculer ou d'appeler des fonctions d'*Excel*.

Voir l'onglet `Fonctions` de la feuille *Excel* nommée `funcs` sur `K:\Stat_ad` pour voir où mettre les valeurs...

2.2 Eviter les fonctions avec *Excel*

Comme la programmation n'est pas toujours simple avec *VBA*, on peut souvent se contenter de mettre des formules dans des cellules ce qui remplace la programmation.

Rappeler comment afficher la loi binomiale sous *Excel* selon le modèle de l'onglet `Binom` de la feuille *Excel* nommée `funcs` sur `K:\Stat_ad` à savoir : On met n en `B1`, p en `D1` et on affiche les x_i en colonne 2, les p_i en colonne 3 via `LOI.BINOMIALE` et en colonne 4 à la main via $C_n^k p^k (1-p)^{n-k}$; prendre $n = 5$, $p = 0.3$.

Là encore, on se servira de la page *Web*

`http://www.info.univ-angers.fr/pub/gh/vitrine/Democgi/loisStatp.htm`

pour vérifier les résultats.

Même chose pour la loi de *Poisson*.

On ajoutera en colonne 5 le calcul par la formule de récurence d'un terme à l'autre à savoir : à l'ordre $k+1$ on multiplie par λ et on divise par $k+1$ (voir l'onglet Poisson).

Reprendre $\lambda = 1.5$ pour $\tau = 10$ termes (montrer comment on obtient λ et τ à partir de l et t via la police Symboles).

2.3 Fonctions avancées en *Excel*

Comme il restera sans doute peu de temps, on montrera comment écrire un programme de type *Sub* qui écrit 4 cellules avec un bouton pour lancer l'exécution du programme (qui au passage demande la plage de variation des cellules).

Voici le programme :

```
Sub decrit()

' demande de la plage

 Set r = Application.InputBox(
          Prompt:="Sélectionner la plage de recherche",
          Type:=8
         )

' calculs

  lamoy  = Application.Average(r)
  sonect = Application.StDev(r)

' stockage dans les cellules

Worksheets(1).Cells(14, 2) = "moyenne"
Worksheets(1).Cells(14, 3) = lamoy
Worksheets(1).Cells(14, 4) = "écart-type"
Worksheets(1).Cells(14, 5) = sonect

End Sub
```

Une fois le programme saisi (et enregistré, ce qui montre les erreurs éventuelles), il faut faire apparaitre la barre d'outils *Formulaires*, cliquer sur le symbole du bouton et le dessiner (aux environs de B4). Lorsque le panneau "affecter" apparait, lui dire que c'est la macro *decrit* qui est associée au bouton.

Changer le texte du bouton puis vérifier que le programme marche en cliquant sur le bouton. Avec l'onglet `Sub` de la feuille proposée, c'est la plage `$B$1:$K$1` qu'il faut fournir comme zone d'entrée.

Leur faire chercher pourquoi on ne voit rien... leur faire rajouter

```
MsgBox ("et voila ")
```

Leur faire modifier le programme pour qu'il affiche au bon endroit, leur faire rajouter le nombre de termes et le *cdv* (coefficient de variation, déjà programmé).

2.4 Pour ceux qui voudraient approfondir :

Rappeler ce qu'est un test en `si` en algorithmique et donner la traduction en *Excel*. Faire un programme (sub) *testevaleurs* qui refuse de calculer si n n'est pas un entier, si p n'est pas un réel compris entre 0 et 1 ; n étant par exemple en `B1` et p en `D1`.

Rappeler ce qu'est une boucle `pour` en algorithmique et donner la traduction en *Excel*. Ecrire un programme (sub) *sdpe* qui calcule la somme des n premiers entiers. Insister sur les variables locales, donner un algorithme rapide.

Rappeler ce qu'est une boucle `tant que` en algorithmique et donner la traduction en *Excel*. Ecrire un programme (sub) $diffalphan$ qui donne afiche les valeurs de n et α_n tant que $|\alpha_n - 1/3| > 10^{-5}$ où α_n correspond à la position du sprite.

On rappelle que

$$\alpha_n = \frac{1}{3} + \frac{2}{3}\left(\frac{-1}{2}\right)^n$$

On trouvera une solution dans la feuille `subs.xls` du disque `K:` à l'emplacement habituel.