FORMASCIENCES
Ecologie des Forêts, Prairies et milieux Aquatiques
Environnement et Agronomie
Mathématiques et informatiques appliquées

Réseau National Systèmes Complexes

A L'INITIATIVE DU RESEAU **MEXICO**
(reseau-mexico.fr)

ECOLE-CHERCHEURS

Tutoriel des TP de l'École Chercheur du

réseau MEXICO

05-06-2012

Table des matières

1 Objectifs et consignes du TP....3
2 Exercices....3
2.a Méthode de Morris....3
2.b Plan fractionnaire....5
2.b.1 Génération d'un plan fractionnaire....5
2.b.2 Plan fractionnaire et pavage....5
2.b.3 Analyse de variance....5
2.b.4 Indices de sensibilité....6
2.c Plan complet....6
2.c.1 Génération du plan....6
2.c.2 Analyse de variance....6
2.c.3 Indices de sensibilité....7
3 Annexe : codes des fonctions utilisées dans le TP1....7

TP1 Méthodes de criblage par discrétisation de l'espace

Auteur : Claude Bruchou
Unité de Biostatistique et processus Spatiaux (BioSP)
INRA Avignon (France)

Mise à jour 2012: Hervé Monod

1 Objectifs et consignes du TP

Le TP vise l'initiation sous R aux méthodes d'analyse de sensibilité d'une Fonction Code (FC) . Les méthodes sont présentées dans le support de cours. Les objectifs sont:

- détecter les facteurs influents sur une sortie d'une FC et caractériser les interactions.
- vérifier la cohérence des résultats obtenus à partir de la méthode de Morris et de l'anova sur un plan fractionnaire et complet.

Les objets R nécessaires au TP1 (cf annexe) sont disponibles dans la librairie ***Ecmexico2012*** et en annexe de ce document.
Le TP vise l'analyse de la FC *wwdm* (cf chapitre de présentation des modèles) qui utilise les objets suivants :

wwdm.model(...) : FC du modèle *Winter Wheat Dry Matter* pour un scénario (vecteur de valeurs des 7 facteurs d'intérêt),

wwdm.simule(...) : exécute la FC pour un ensemble de n scenarios présentés dans un tableau (data.frame) à n lignes et 7 colonnes,

wwdm.climates : tableau des données climatiques (sur 14 années) nécessaire pour le calcul de la FC,

wwdm.factors : tableau des descriptifs (noms, gammes de variation,..) des facteurs.

La fonction **morris**() est disponible dans la librairie (package) **sensitivity** de R. La documentation en ligne est accessible sous R par la commande `?morris`

2 Exercices

Q1 : créer une matrice X à 2 lignes et 7 colonnes contenant des valeurs des facteurs *Eb, Eimax, K, Lmax, A, B* et *TI* puis lancer l'exécution de la FC:

```
Nbfac <- 7
# Introduire un vecteur contenant 14 valeurs avec c( , , …)
# N.B : le remplissage de la matrice X est effectué par ligne (byrow=TRUE)
X <- matrix( ?? , nrow=2, ncol=Nbfac, byrow=TRUE)
colnames(X) <- wwdm.factors$name[1:Nbfac]
y <- wwdm.simule(X, year = 9,transfo=TRUE)
print(y)
```

2.a AS selon la méthode de Morris

Q2 : les arguments ***r,*** ***levels*** et **grid.jump** de la fonction Morris désignent respectivement le nombre de trajectoires, le nombre de niveaux de la gamme discrétisée des facteurs et le pas de passage entre deux points d'une trajectoire. Les '…' indiqués dans les arguments de la fonction permettent de passer des valeurs aux arguments de la FC. Après avoir complété les valeurs manquantes des variables *Nbtraj*, *Nbniv* et *delta*, vérifier la bonne marche du code suivant :

```
Nbtraj <- ??
Nbniv  <- ??
delta  <- ??
etude.morris <- morris(model = wwdm.simule, factors = wwdm.factors$name[1:Nbfac] ,
                       r = Nbtraj ,scale=FALSE,
                       design = list(type = "oat", levels = Nbniv, grid.jump= delta),
                       transfo = TRUE,
                       b1=wwdm.factors$binf[1:Nbfac],
                       b2=wwdm.factors$bsup[1:Nbfac],
                       year=9)
```

N.B. : la fonction *morris()* fonctionne *pour* des facteurs prenant leurs valeurs dans [0,1].

Les valeurs des arguments *binf* et *bsup* de *morris()* doivent être respectivement 0 et 1.

Une transformation linéaire est nécessaire pour se ramener aux gammes utilisées par la FC.

L'argument de *wwdm.simule* *transfo = TRUE* complété des arguments binf et bsup (bornes inférieures et supérieures des facteurs) effectue la transformation.

Q3 : Analyse de l'échantillonnage de l'espace des facteurs

Pour une discrétisation en 6 niveaux et des nombres de trajectoires de 20, 50 et 100 construire à l'aide de la fonction *TP1histo* les histogrammes des valeurs des facteurs échantillonnées

```
TP1histo(etude.morris)
```

Vérifier l'uniformité des valeurs échantillonnées des facteurs.

Choisir un nombre de trajectoires pour la suite du TP.

Q4 : Exploration graphique de l'influence des facteurs sur la sortie

Construire à l'aide de *TP1corr* les graphiques de liaison entre les facteurs et la sortie de la FC. La fonction *lowess* de R construit une courbe de régression non paramétrique.

```
b1 <- wwdm.factors$binf[1:Nbfac]
b2 <- wwdm.factors$bsup[1:Nbfac]
TP1corr(etude.morris, transfo=T, binf=b1, bsup=b2 )
```

Q5 : Analyse des sorties

Visualiser le graphe de Morris et commenter:

```
par(mfrow=c(1,1),ask=T)
plot(etude.morris)
```

Q6 : Calculer les IC à 95% des indices de Morris

*TP1.ICmorris(**etude.morris**)*

2.b AS avec Plan fractionnaire et Anova

2.b.1 Génération d'un plan fractionnaire

Q7 : On choisit de caractériser la gamme de variation des 7 facteurs par 2 niveaux.

Quels sont les niveaux qui vous semblent pertinents dans une première approche?

On propose de rechercher un plan de résolution V pour les 7 facteurs à 2 niveaux. Il sera noté *plan7.2.V* . Dans ce but, par une recherche systématique trouver la taille minimale de ce plan en recherchant le plus petit entier **r** inférieur 7 pour lequel la fonction regular.fraction trouve une solution :

```
Nbfac <- 7

r <- ??

plan7.2.V <- regular.fraction(Nbfac, p=2, r , resolution=5)$plan
# associer les intitulés des facteurs aux colonnes de l'objet
colnames(plan7.2.V) = wwdm.factors$name[1:Nbfac]
```

2.b.2 Plan fractionnaire et pavage

Q8 : Pour améliorer l'exploration de l'espace des 7 facteurs, on associe un pavage de l'espace des facteurs au plan fractionnaire précédent. La gamme de chaque facteur est donc partagée en *Nbclass=2* classes de même amplitude. L'identification des pavés de l'espace est définie à partir du tableau *plan7.2V*. On effectue *nrep=3* tirages au hasard dans chaque pavé. Le calcul est effectué avec la fonction *TP1pavage()*

```
# Lancer le code suivant pour réaliser le tirage :

# Rem. : la ligne qui suit permet de coder le résultat obtenu en indice >= 1
plan7.2.V <- plan7.2.V + 1
Nbfac <- 7
# planfrac.pav  est une liste ayant pour composantes P.out (facteurs codés)
#  et xx (coordonnées des points)
planfrac.pav <- TP1pavage(P = plan7.2.V, nrep =3, Nbclass =2,
                          binf =  wwdm.factors$binf[1:Nbfac],
                          bsup = wwdm.factors$bsup[1:Nbfac])
```

Q9 : Calculer la sortie de la FC associée au plan *planfrac.pav* précédent:

```
planfrac.out <- wwdm.simule(planfrac.pav$xx, transfo = FALSE, year=9)
```

N.B. : *TP1pavage* fournit le codage adéquat des facteurs, d'où *transfo=F*.

2.b.3 Analyse de variance

Q10 : Création du data.frame *Tabsimu.dat* destiné à l'analyse de la variance de la sortie de la FC au mooyen de la fonction *aov*

```
Nbfac <- 7
# codage des facteurs du plan en objets de classe factor pour aov()
planfrac.fac <- lapply(data.frame(planfrac.pav$P.out), as.factor)
planfrac.dat <- data.frame(planfrac.fac,  Y = planfrac.out)
```

Q11 : Calculer et éditer la table d'anova pour un modèle à 7 facteurs et comprenant les interactions d'ordre 2 ajusté à la sortie contenue dans *planfrac.dat* :

```
# écriture du modèle d'anova avec effets principaux et interaction d'ordre 2
mod.aov1 <- formula( Y ~ (Eb + Eimax + K +  Lmax +  A + B + TI)^2)
planfrac.aov <- aov(mod.aov1, planfrac.dat)
planfrac.table <- summary(planfrac.aov)
```

Q13 : Explorer les interactions entre deux facteurs et identifier celles qui semblent importantes (il suffit de remplacer ?? par les noms des facteurs) :

interaction.plot(`planfrac.dat$?? , planfrac.dat$??, planfrac.dat$Y)`

2.b.4 Indices de sensibilité

Q14 : Calculer les indices de sensibilité principaux et totaux à partir de la table d'anova :

```
  indices.aov <- TP1indices.aov(planfrac.table, noms =
wwdm.factors$name[1:Nbfac], modeleAOV = mod.aov1, titre='Plan Fractionnaire')
```

2.c AS avec Plan complet et Anova

2.c.1 Génération du plan d'expériences

Q15 : Générer un plan complet comprenant 7 facteurs à *3* niveaux chacun. Un pavage est associé à ce plan. On effectue *nrep = 3* tirages au hasard dans chaque pavé. Calculer les sorties associées de la FC.

```
Nbfac <- 7
nbniv <- 1:3 ; nrep =3
plancomplet <- expand.grid( Eb = nbniv, Eimax = nbniv, K = nbniv,
                            Lmax = nbniv, A = nbniv, B = nbniv,
                            TI = nbniv, REP = 1:nrep)
      N <- nrow(plancomplet)
      b1 <- wwdm.factors$binf[1:Nbfac]
      b2 <- wwdm.factors$bsup[1:Nbfac]
      simu <- t(apply(plancomplet[,1:Nbfac], 1, TP1tirage, b1, b2, 3) )
      simu.out <- wwdm.simule(simu,transfo =F, year=9)
```

2.c.2 Analyse de la variance de la sortie de la FC

Q16 : Construire le *data.frame* pour la fonction *aov* et effectuer l'analyse.

```
# transformation de la matrice en objet factor pour aov()
    plancomplet.fac <- lapply(data.frame(plancomplet), as.factor)
    plancomplet.dat <- data.frame(plancomplet.fac,  Y = simu.out)
# formule du modèle d'anova :
    mod.aov2 <- formula(Y ~ (Eb + Eimax + K +  Lmax +  A + B + TI)^4)
    plancomplet.aov <- aov(mod.aov2, plancomplet.dat )
```

Q17 : Établir la table d'anova du modèle précédent.

```
plancomplet.table <- summary(plancomplet.aov)
```

Q18 : Identifier les interactions entre deux facteurs qui semblent importantes :

interaction.plot(`plancomplet.dat$?? , plancomplet.dat$??, plancomplet.dat$Y)`

2.c.3 Indices de sensibilité

Q19 : Calculer les indices de sensibilité issus de l'anova du plan complet.

indices.aov <- TP1indices.aov(plancomplet.table, noms = wwdm.factors$name[1:Nbfac], modeleAOV = mod.aov2, titre='Plan Complet')

2.c.4 Conclusions

Q20 : comparer les résultats issus des trois analyses précédentes.

---------- FIN du TP 1 ----------

3 Annexe : codes des fonctions utilisées dans le TP1

```
 wwdm.simule.TP1 = function(X, year, tout=F, transfo = FALSE, b1=0,b2=0){
 #  à utiliser pour analyser  la FC wwdm  avec morris() de sensitivity
#  transfo = T : recodage de la matrice X codée dans [0,1]
#  b1 = vecteur des bornes inférieures des gammes des facteurs
#  b2 = vecteur des bornes supérieures des gammes des facteurs
   if(transfo) X = t(b1 + t(X)*(b2-b1))
   if(is.null(year))
       sortie = apply(X,1, function(v) sum(wwdm.model(v[1:7],v[8])) )
    else
       sortie = apply(X,1,function(v) sum(wwdm.model(v[1:7],year=year)))
 if(tout) sortie = cbind(X,Biomasse = sortie)
return(sortie)
}
 TP1histo = function(etude.morris){
```

```
# diagramme des fréquences des valeurs des facteurs échantillonnées
#  etude.morris = structure issue de la fonction morris du package sensitivity
          noms =  etude.morris$factors
          Nbfac = length(noms)
          Nbtraj=  etude.morris$r
          numtraj = rep(1:Nbtraj, each = Nbfac+1)
       par(mfrow=c(3,3))
          for(i in 1:Nbfac) barplot(table(unlist(tapply(etude.morris$X[,i],
numtraj,unique))),col= 'blue',main= noms[i])
}

 TP1corr  =  function(etude.morris, transfo=T, binf , bsup ){
# graphiques de corrélation entre facteurs X et sortie y de la FC
# etude.morris = structure liste issue de la fonction morris du package sensitivity
#  transfo = T : recodage de la matrice X codée dans [0,1]
#  binf = vecteur des bornes inférieures des gammes des facteurs
#  bsup = vecteur des bornes supérieures des gammes des facteurs
          noms =  etude.morris$factors
          Nbfac = length(noms)
         if(transfo)  X = t(binf + t(etude.morris$X)*(bsup-binf))
         if(!transfo) X =  etude.morris$X
          par(mfrow=c(3,3),ask=T)
          for(i in 1:Nbfac){
              plot(X[,i], etude.morris$y, pch=21, bg='lightblue')
               title(noms[i])
              lines( lowess(X[,i] , etude.morris$y), lwd=2, col="red")
                         }
      }

 TP1.ICmorris = function( etude.morris){
#  etude.morris = structure liste issue de la fonction morris du package sensitivity
          noms =  etude.morris$factors
          Nbfac = length(noms)
          Nbtraj = etude.morris$r
         sigma = sqrt(apply(etude.morris$ee,2,var))
         IC.mu = apply(abs(etude.morris$ee),2, t.test)
          tabICmu = matrix(0,Nbfac,2); tabICsig = matrix(0,Nbfac,2)
          rownames(tabICmu) = noms ; rownames(tabICsig) = noms
         for(i in 1:Nbfac) {
               tabICmu[i,]  = IC.mu[[i]]$conf.int
               tabICsig[i,] = c(sigma[i]*(Nbtraj-1)^.5/qchisq(.975,Nbtraj-1)^.5,
                           sigma[i]*(Nbtraj-1)^.5/qchisq(.025,Nbtraj-1)^.5)
                            }
           print('IC mu*') ; print(tabICmu); print('IC sigma');print(tabICsig);
           mu.star <- apply(etude.morris$ee, 2, function(x) mean(abs(x)))
```

```
            xlim1= c(min(tabICmu[,1]), max(tabICmu[,2]))
            ylim1= c(min(tabICsig[,1]), max(tabICsig[,2]))
            plot(mu.star,sigma,xlim=xlim1,ylim=ylim1)
            for(i in 1:Nbfac) { segments(tabICmu[i,1], sigma[i], tabICmu[i,2],
sigma[i],col= 'red')
segments(mu.star[i],tabICsig[i,1], mu.star[i], tabICsig[i,2], sigma[i],col= 'red')
                              }
                                          }

 TP1tirage = function(PAV, binf, bsup, Nbclass){
      #              TIRAGE uniforme dans un pave PAV de R^K
      #  Nbclass = niveau de discrétisation des gammes des Nbfac facteurs
      #  PAV = vecteur des Nbfac coordonnées entières d'un pavé codées de 1 à Nbclass
      # binf, bsup = vecteurs des bornes inf et sup des gammes des facteurs
      #  sortie = vecteur à K éléments
         Nbfac = length(PAV)
         bornes = matrix(NA, nrow = Nbfac, ncol=2)
         bornes[,1] = binf + (PAV-1)*(bsup-binf)/Nbclass
         bornes[,2] = binf + PAV*(bsup-binf)/Nbclass
         cc = numeric(Nbfac)
      for(i in 1:Nbfac) cc[i] = runif(1, min = bornes[i,1], max = bornes[i,2])
         cc
          }

 TP1pavage = function(P, nrep =3, Nbclass = 2, binf,  bsup) {
      # Construction du plan avec tirage dans les pavés défini par un plan P
      # P = matrice à Nbfac colonnes  codée par des entiers de 1 à Nbclass
      # sortie = liste contenant les matrices des coordonnées entières (Plan.rep) et
      # réelles (xx) des points tirés au hasard
      # nrep = nbre de tirages par pavé [entier]
      # Nbclass = niveau de discrétisation des gammes des facteurs
      # binf, bsup = vecteur des bornes inf et sup des gammes des facteurs
                  Nbfac = ncol(P)
                  M=NULL ; for(j in 1:nrep) M = rbind(M, P)
                  Plan.rep= M
                  list(P.out = Plan.rep,
                     xx = t(apply( Plan.rep, 1, TP1tirage, binf, bsup, Nbclass) ))
                        }

   TP1indices.aov = function(table.aov, noms , modeleAOV , titre=''){
       # calcul des indices principaux et totaux
       # noms = vecteur des labels des facteurs
       # modeleAOV = modèle d'anova créé avec formula()
       # table.aov = table d'anova issue de la fonction aov()
          vv = terms( modeleAOV , keep.order = F)
```

```
  # SS = somme des carrés,
  Nbfac = length(noms)
  SS = table.aov[[1]][2]
  neffets = nrow(SS)-1
  SS.fac = SS[1:neffets,]
  SStot = sum(SS.fac)
  vv1 = attr(vv, "factors")
  Itot = rep(NA,Nbfac)
  Iprinc = SS.fac[1:Nbfac]/SStot
  for(i in 1:Nbfac) Itot[i] = sum(SS.fac[vv1[i+1,]==1])/SStot
      M = rbind(Iprinc, Itot-Iprinc)
     barplot( M, col=c("lightblue", "blue"), names.arg=noms)
      title(titre)
}
```