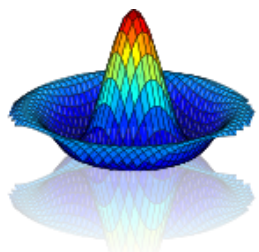
Utilisation de mtk par un R-user

Rencontres 2012

Robert Faivre

INRA - Biométrie et Intelligence Artificielle, MIA Toulouse

Nantes, le 22 novembre 2012



MEXICO

Plan

Plan

Objectif

Se placer dans le rôle d'un utilisateur de R
Quel est l'enchaînement des traitements ?

- Déclaration des facteurs (make.mtkFactor)
- Déclaration du dispositif expérimental (mtkSampler)
- Déclaration du simulateur (mtk.simulatorAddons)
- Déclaration de la méthode d'évaluation (mtkAnalysor)
- Intégration de méthode

Le modèle Weed

Modèle agronomique d'aide à la décision (N. Munier-Jolain, B. Chauvel, and J. Gasquez. Longterm modelling ofweed control strategies : Analysis of threshold-based options for weed species with contrasted competitive abilities. Weed Research, 42 :107–122, 2002.).

Le vulpin, plante adventice du blé considérée comme mauvaise herbe.
Le modèle simule l'effet d'une population de cette plante sur le rendement d'une culture de blé en fonction de différentes pratiques agricoles (travail du sol, désherbage chimique, type de culture).
L'implémentation simule, à un pas de temps annuel, cinq variables décrivant l'état du système :

– S, le nombre de graines de vulpin par m^2 dans la parcelle cultivée,
– d, la densité de plantes de vulpin à émergence *i.e.* en début de saison (plantes par m^2),
– SSBa (Surface Seed Bank), le nombre de graines de vulpin par m^2 dans les horizons de surface du sol, après travail du sol,
– DSBa (Deep Seed Bank), le nombre de graines de vulpin par m^2 en

Les entrées du modèle Weed

Le modèle utilise trois types de facteur d'entrée :
– les **valeurs initiales des quatre variables d'état** caractérisant la population de vulpin à t = 0 (S, d, SSBa, DSBa),
– les **16 paramètres** du modèle (quantités supposées fixes dans le temps),
– les **techniques culturales**
Les valeurs des paramètres sont considérées incertaines. La gamme d'incertitude de ces paramètres a été définie par des bornes supérieures et inférieures égales à ± 10% des valeurs nominales.
Les techniques culturales appliquées chaque année sont décrites par trois variables binaires :
– Soil = 1 si labour, Soil = 0 si travail du sol superficiel,
– Herb = 1 si un traitement herbicide est appliqué, Herb = 0
Deux scénarios de traitement herbicide sont définis :
Cas 1 : traitement systématique chaque année,
Cas 2 : traitement systématique sauf l'année 3.

Codage du modèle Weed annuel

Weed <- function(d.im1, S.im1, SSBa.im1, DSBa.im1, Soil, Crop, Herb, param)

{

mu = param[1] ; v = param[2] ; phi = param[3] ; beta.1 = param[4] ; beta.0 = param[5] ; ... ; Ymax = param[14] ; rmax = param[15] ; gamma = param[16]

beta = beta.1*Soil+beta.0*(1-Soil) ; chsi = chsi.1*Soil+chsi.0*(1-Soil)
Smax = Smax.1*Crop+Smax.0*(1-Crop) ; alpha = Smax/160000

SSBb.i = (1-mu)*(SSBa.im1-d.im1)+v*(1-phi)*S.im1
DSBb.i = (1-mu)*DSBa.im1
SSBa.i = (1-beta)*SSBb.i+chsi*DSBb.i
DSBa.i = (1-chsi)*DSBb.i+beta*SSBb.i

d.i = delta.new*v*(1-phi)*(1-beta)*S.im1 + delta.old*(SSBa.i-S.im1*v*(1-phi)*(1-beta)) D.i = (1-mh*Herb)*(1-mc)*d.i

S.i = Smax*D.i/(1+alpha*D.i)

Yield.i = max(0, Ymax*(1 - (rmax*D.i/(1+gamma*D.i))))

return(c(d.i, S.i, SSBa.i, DSBa.i, Yield.i))

}

Codage de la fonction weed.model (Weed pluriannuel)

```
weed.model <- function(param, weed.deci) {

d.0 = 400 ; S.0 = 68000 ; SSBa.0 = 3350 ; DSBa.0 = 280 ; ...
NumY = length(weed.deci$Soil)
pred.d = rep(NA,NumY) ; pred.S = rep(NA,NumY) ; pred.SSB = rep(NA,NumY)

Soil.vec = weed.deci$Soil ; Crop.vec = weed.deci$Crop ; Herb.vec = weed.deci$Herb
for (i in 1 :NumY) {
        Soil = Soil.vec[i] ; Crop = Crop.vec[i] Herb = Herb.vec[i]
        Z = Weed(d.im1,S.im1,SSBa.im1,DSBa.im1,Soil,Crop,Herb,param)
        pred.d[i] = Z[1] ; pred.S[i] = Z[2] ; ... ; pred.Yield[i] = Z[5]

        d.im1 = Z[1] ; S.im1 = Z[2] ; SSBa.im1 = Z[3] ; DSBa.im1 = Z[4]
        }
return(pred.Yield)
}
```

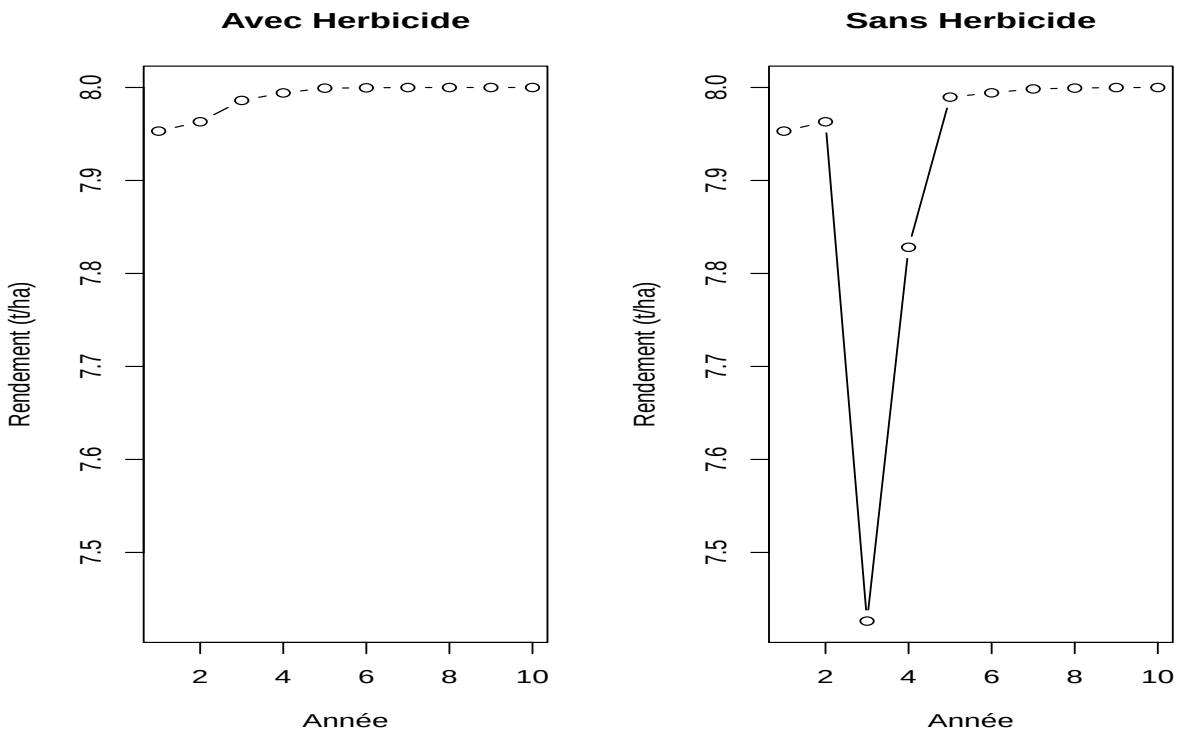
Utilisation classique

fNominal <- c(mu= 0.84, v= 0.6, phi= 0.55, beta.1= 0.95, beta.0= 0.2, chsi.1= 0.3, chsi.0= 0.05, delta.new= 0.15, delta.old= 0.3, mh= 0.98, mc= 0, Smax.1= 445, Smax.0= 296, Ymax= 8, rmax= 0.002, gamma= 0.005)

sortie1 <- **Weed**(400, 68000, 3350, 280, Soil= 0, Crop= 1, Herb= 1, param= fNominal)
round(sortie1, 2)
[1] 2317.15 18268.05 15067.84 3808.96 7.40

deci.4ans <- list(Soil = c(1,0,1,0), Herb = c(0,1,0,1), Crop = (1,1,1,1))
sortie2 <- **weed.model**(fNominal, deci.4ans)
round(sortie2, 2)
[1] 6.63 7.56 6.78 7.62

Exemple de comportements



Rendements simulés pendant 10 ans pour deux cas de traitement herbicide : application systématique d'un traitement (à gauche) et application systématique d'un traitement sauf l'année 3 (à droite). Les paramètres ont été fixés à leurs valeurs nominales.

Plan

Déclaration des facteurs (mtkExpFactors - make.mtkFactor)

```
fMin = 0.9*fNominal ; fMax = 1.1*fNominal ; fMax[11] = 1

weedFactors = mtkExpFactors()

for(i in c(1 :16)) {
        facteur.i = names(fNominal)[i]
        weedFactors[facteur.i] = make.mtkFactor(
        name = facteur.i,
        nominal = fNominal[i],
        distribName = "unif",
        distribPara = list(min=fMin[i],max=fMax[i]))
}
```

Plan

Déclaration du dispositif expérimental (mtkSampler) et l'enchaînement des traitements (mtkExpWorkflow)

```
MCprocess <- mtkSampler("BasicMonteCarlo",
                        information = list(size=1000))

exp1 <- mtkExpWorkflow(expFactors = weedFactors,
                       processesVector = c(design=MCprocess))

set.seed(2)
run(exp1)
exp2 <- exp1
```

Actuellement : mtkNativeSampler à la place de mtkSampler

Plan

Déclaration du simulateur

Enrobage de la fonction weed.model pour travailler sur un plan d'expérience de valeurs de paramètres X.

```
WEED.simule<-function(X, decision, outvar = 1 :10, ...)
{
      weed.deci = weed.decision[[decision]]
      Y = apply(X,1,function(w) weed.model(w, weed.deci)[outvar])
      if(length(outvar)>1) Y = t(Y)
      Y = as.data.frame(Y)
      output = list(main= Y, information = list( decision = decision,
                  outvar = outvar))
return(output)
}
```

On regroupe dans un même fichier les 3 fonctions (Weed, weed.model, weed.simule) dans un même fichier (WeedModel.R).

Intégration de la classe mtkWEEDSimulator (mtk.simulatorAddons)

Création de la classe mtkWEEDSimulator.

```
mtk.simulatorAddons(where="WeedModel.R",
        name= "WEED",
        main= "WEED.simule",
        authors= "D.Makowski (2012)",
        summary= NULL,
        plot= NULL,
        print= NULL)
```

Intégration de la classe WEEDSimulator.

```
source("mtkWEEDSimulator.R")
```

Déclaration du processus de simulation (mtkNativeSimulator et addProcess)

Déclaration du processus de simulation pour chacune des deux options de traitement herbicide

weed.cas1<-**mtkSimulator("WEED", information = list(decision=1))**
weed.cas2< **mtkSimulator("WEED", information = list(decision=2))**

Déclarations des processus de simulation puis exécution (**run**)

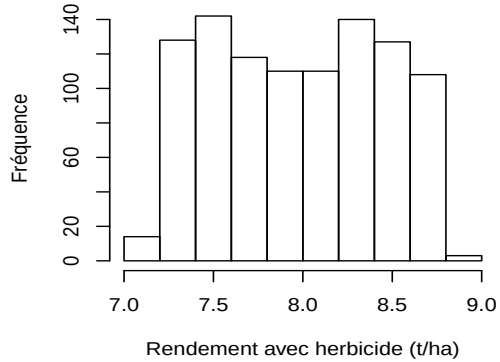
```
addProcess(exp1, p=weed.cas1, name="evaluation")
run(exp1)
addProcess(exp2, p=weed.cas2, name="evaluation")
run(exp2)
```

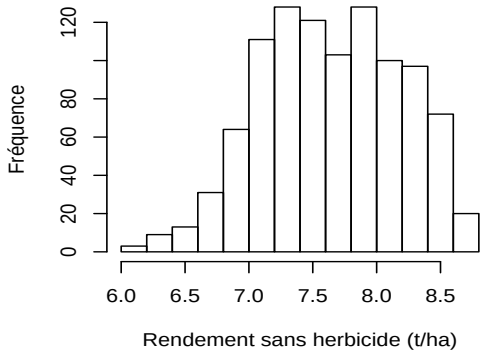
Actuellement : mtkNativeSimulator à la place de mtkSimulator

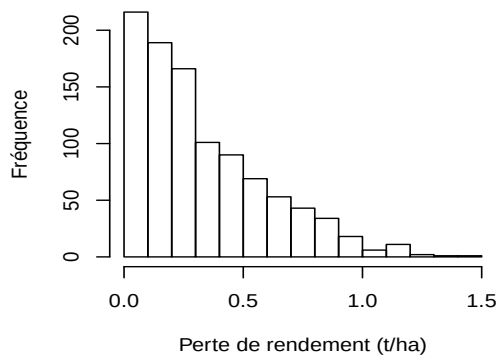
Résultats de l'analyse d'incertitude

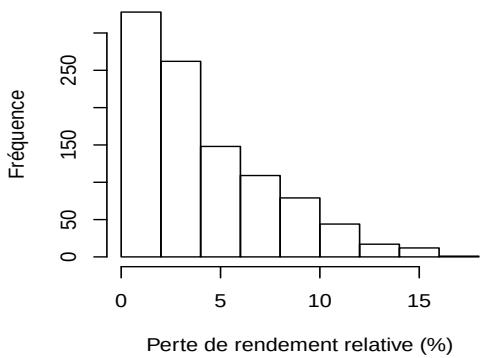
```
y.cas1 <- extractData(exp1,name=list("evaluation"))[, 3]
y.cas2 <- extractData(exp2,name=list("evaluation"))[, 3]
par(mfrow=c(2,2))
hist(y.cas1, main="", xlab="Rendement avec herbicide (t/ha)")
hist(y.cas2, main="", xlab="Rendement sans herbicide (t/ha)")
hist(y.cas1-y.cas2, main="",xlab="Perte de rendement (t/ha)")
hist(100*(y.cas1-y.cas2)/y.cas1, main="", xlab= "Perte de rendement
relative (
```

Visualisation









Rendements simulés avec et sans herbicide, et pertes de rendement induites par la non application d'herbicide l'année 3.

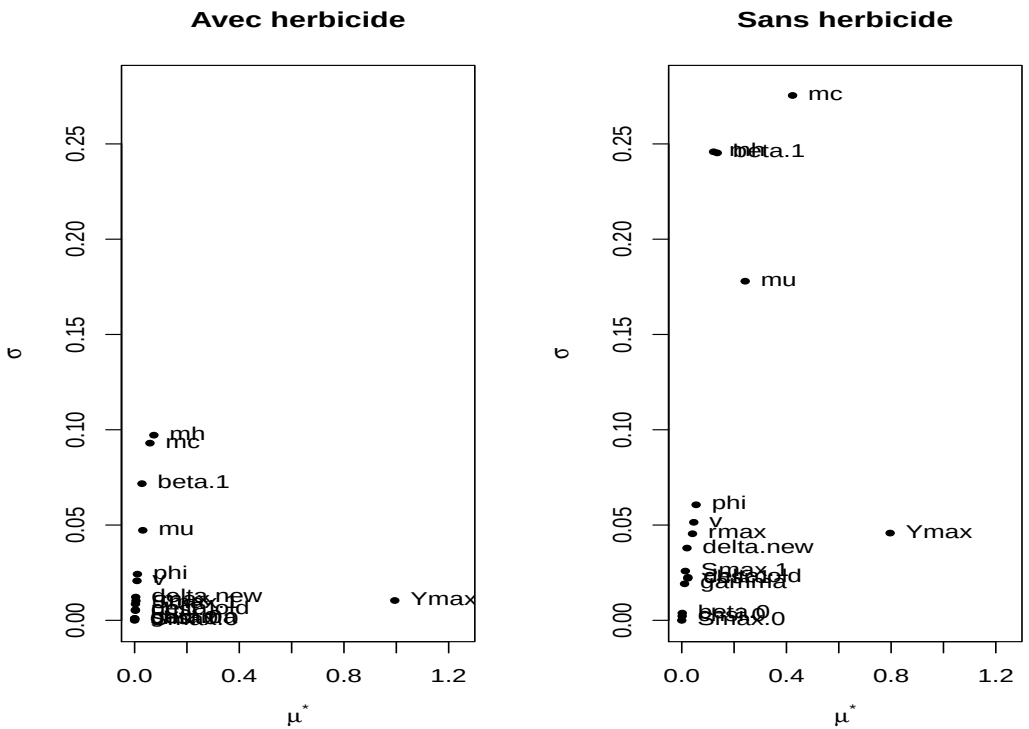
Plan

Déclaration de la méthode d'évaluation (mtkAnalysor)

```
Morris.S <- mtkSampler("Morris", information = list( r=500,
      type ="oat", levels=4, grid.jump=2, scale=TRUE))

Morris.A <- mtkAnalysor("Morris")

weed.cas1 <- mtkSimulator("WEED",
                information = list(decision=1,outvar=3))

exp3 <- mtkExpWorkflow(expFactors = weedFactors,
                processesVector = c(design = Morris.S,
                evaluation = weed.cas1,
                analysis = Morris.A))
set.seed(2)
run(exp3)
```

Actuellement : mtkNativeAnalysor à la place de mtkAnalysor

Visualisation



Indices de Morris obtenus pour les 16 paramètres incertains du modèle WEED, avec et sans application d'herbicide.

plot(extractData(exp3, name=list("analysis")), xlim=c(-0.1,1.5)) title("Avec herbicide")

plot(getProcess(exp3, name="analysis"))

Plan

Intégration d'une méthode d'échantillonnage

Les arguments de la fonction doivent être **factors, distribNames, distribParameters** suivis des arguments spécifiques à la fonction encapsulée et la sortie doit être une structure de type **list(main = ..., information = ...)**

```
RandomLHS <- function(factors, distribNames, distribParameters,
        size = 10, preserveDraw = FALSE, ...)
{
...
resultat <- list(main = as.data.frame(design), information = information)
}
mtk.samplerAddons(where="RandomLHS.R", name="RandLHS",
        authors="R. Carnell - R.Faivre (2012)", main= "RandomLHS",
        summary=NULL,plot=NULL, print=NULL)
source("mtkRandLHSSampler.R")
MCprocess2 <- mtkSampler("RandLHS", information = list(size=1000))
```

Intégration d'une méthode d'analyse

L'ensemble des fonctions se trouvent dans le fichier plmm.R
La fonction principale (celle qui doit respecter les règles d'entrée et de sortie des informations) est : plmm
Le nom utilisé dans la description de l'expérience sous mtk est : PLMM

```
plmm <- function(X,Y, degre.pol= 1, numY= 1, listeX=NULL, ...) {
...
resultat <- list(main=resultats, information=information) }
mtk.analysorAddons(where="plmm.R", name="PLMM",
      authors="R. Faivre, INRA-MIA Toulouse", main="plmm",
summary="summary.plmm", plot="plot.plmm")
source("mtkPLMMAnalysor.R")
weed.plmm <- mtkAnalysor("PLMM", information = list(degre.pol = 1,
numY = 1,listeX=1 :16))
addProcess(exp5, p = weed.plmm, name = "analysis")
run(exp5)
summary(getProcess(exp5,name="analysis"), lang="fr")
```

Licence