Deliverables D4.3

Document of Training on the R software
Mbour, Senegal, June 2007

Content

Introduction

During the Tenerife meeting in November 2006 we had decided to provide a R training course to istam members with several objectives. The first one is to promote a tool with trends so as to be a standard in our community. Having a tool for the community will make it easier to work together, share methods and models to evaluate the state of the fisheries / stock. Another objective sought by the course is to help researchers of the sub region try a new tool that will be used by most of the case studies within the WP3.

A second training course took place in Mbour, Senegal, in June 2007 prior to the Istam meeting.

We (J. Guitton, N. Bez and A. Faraj) have decided to present the software not the way it is usually done. A big part of the course was made to train the participant to insert his data in the software in different ways: create data objects, import .csv files and get data directly fromthe database. While we import the data using several methods, we used different ways for data processing and to show the data: summary or plot but also mapping of the data. We have used data from scientific surveys used by the group.

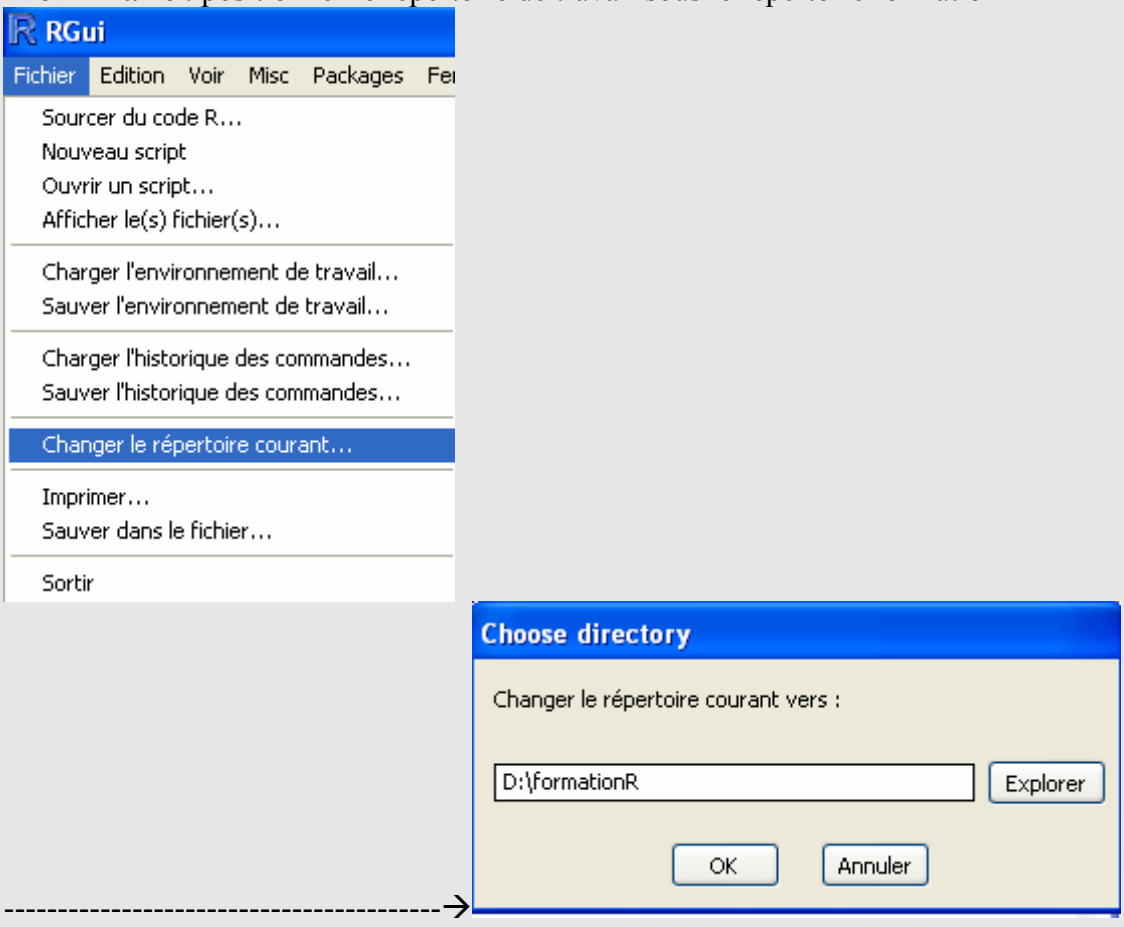
Unusual/different view of this software allowed all the participants (those who already know R and the new users) to learn about the way to use this software. Indeed learners have noticed acknowledged the great way to present the software to a heterogeneous group of data users.

1. Chargement de données.

Pour s'approprier un logiciel, il faut en premier lieu pouvoir y travailler ses données. C'est pourquoi nous avons prévu une partie assez longue sur l'import/export de certaines données pour R.

Ensuite l'idée est simplement de vous faire entrevoir les possibilités d'un tel logiciel sans forcément se lancer dans de grandes analyses statistiques. Nous allons donc distiller des fonctions utiles sous R au fil de l'import de nos données.
Comment faire une sélection de mes données : les filtres.
Comment faire une opération sur mes données selon un facteur : l 'équivalent des tableaux croisés en quelque sorte.
Comment faire des graphiques.
Comment faire quelques cartes schématiques.

Préliminaire : positionner le répertoire de travail sous le répertoire formationR



1.1. Déclaration de données.

totox = c(1,2,3,4); # Crée un vecteur avec les valeurs 1,2 3 et 4

totoy=seq(10,40,10); # Crée un vecteur qui débute à 10 termine à 40 et le pas entre chaque valeur est 10

plot(totox,totoy); #affiche les point de coordonnées totox, totoy

plot(totox,totoy,type='l'); # le nuage de point devient une droite

1.2. Chargement d'un fichier CSV

On débute avec le fichier XL

ABSCISSE	GROUPE	TOTAL	CLASSE
1995	FMCgm	157.0	PA
1995	FMCgmG	26.0	PA
1995	FMCpm	43.0	PA
1995	FMCtgm	105.0	PA
1995	FMCtgmG	52.0	PA
1995	FMDE	625.0	PA
1995	FMEE	153.0	PA
1995	FMEM	138.0	PA
1995	FMEO	275.0	PA
1995	FT	84.0	PA
1995	LI	97.0	PA
1995	LIG	137.0	PA
1995	PA	451.0	PA
1996	FMCgm	125.0	PA
1996	FMCgmG	8.0	PA
1996	FMCpm	38.0	PA
1996	FMCtgm	124.0	PA
1996	FMCtgmG	27.0	PA
1996	FMDE	698.0	PA
1996	FMEE	142.0	PA
1996	FMEM	140.0	PA
1996	FMEO	306.0	PA
1996	FT	63.0	PA
1996	LI	76.0	PA
1996	LIG	158.0	PA

1996	PA	440.0	PA	
1996	PAG	13.0	PA	
1997	FMCgm		164.0	PA
1997	FMCgmG		11.0	PA
1997	FMCpm		20.0	PA
1997	FMCtgm		138.0	PA
1997	FMCtgmG		28.0	PA
1997	FMDE	764.0	PA	
1997	FMEE	148.0	PA	
1997	FMEM	167.0	PA	
1997	FMEO	300.0	PA	
1997	FT	61.0	PA	
1997	LI	79.0	PA	
1997	LIG	147.0	PA	
1997	PA	519.0	PA	
1997	PAG	15.0	PA	
1998	FMCgm		125.0	PA
1998	FMCgmG		8.0	PA
1998	FMCpm		38.0	PA
1998	FMCtgm		124.0	PA
1998	FMCtgmG		27.0	PA
1998	FMDE	698.0	PA	
1998	FMEE	142.0	PA	
1998	FMEM	140.0	PA	
1998	FMEO	306.0	PA	
1998	FT	64.0	PA	
1998	LI	76.0	PA	
1998	LIG	158.0	PA	
1998	PA	440.0	PA	
1998	PAG	15.0	PA	
1999	FMCgm		125.0	PA
1999	FMCgmG		8.0	PA
1999	FMCpm		38.0	PA
1999	FMCtgm		124.0	PA
1999	FMCtgmG		27.0	PA
1999	FMDE	698.0	PA	
1999	FMEE	142.0	PA	
1999	FMEM	140.0	PA	
1999	FMEO	306.0	PA	
1999	FT	64.0	PA	
1999	LI	76.0	PA	
1999	LIG	158.0	PA	
1999	PA	440.0	PA	
1999	PAG	15.0	PA	
2000	FMCgm		164.0	PA
2000	FMCgmG		13.0	PA
2000	FMCpm		20.0	PA
2000	FMCtgm		138.0	PA
2000	FMCtgmG		29.0	PA

```
2000        FMDE 764.0   PA
2000        FMEE 148.0   PA
2000        FMEM167.0   PA
2000        FMEO 300.0   PA
2000        FT      61.0    PA
2000        LI      79.0    PA
2000        LIG     147.0   PA
2000        PA      519.0   PA
2000        PAG     15.0    PA
#########################################################################
```

```
A<-read.table("classeur1_R.txt", header=T)

A ; # affiche la table

A[1,1] ; # affiche l'élément de la colonne 1, ligne 1

A[1,] ; # Affiche la ligne 1

A[,1] ; # Affiche la colonne 1

A[A[,1]==1995,] # Affiche les lignes ou la colonne 1 est égale à 1995

B<- A[A[,3]>3,]


    for(i in 1995:2000)

plot(A[A[,1]==i,2],(A[A[,1]==i,3]),xlab=i)
```

1.3. Connexion à une base

Remarques : nous allons utiliser 2 requêtes incluses dans la base trawlbase que nous avons fournie.

calcul_niv_troph : cette requête calcule le niveau trophique moyen par station en utilisant le formule suivante : Niv_troph_moy= $\sum (Ci * Ni) / CT$ (Somme des capture de l'espèce i par son niveau trophique sur la capture totale de la station). Elle permet aussi d'avoir 2 autres indicateurs que sont la capture totale de la station et le pourcentage de la capture totale sur lequel le niveau trophique moyen est calculé – pour certaines espèces le niveau trophique n'a pas été renseigné)

calcul_niv_troph : Requête Sélection

an	campagne	station	niv_troph_moy	pourcent	SommeDeTot
1985	AN8503DM	1	3,58524018921635	66,6741852480456	59,12000018
1985	AN8503DM	10	3,57038340287808	97,364953906386	110,85000085
1985	AN8503DM	11	3,68129857074046	76,0793464504905	39,119999807
1985	AN8503DM	12	3,63844337228063	91,2773932522943	126,62000011
1985	AN8503DM	13	3,53901278229415	89,2042697451309	147,07999981
1985	AN8503DM	14	3,54854595112075	77,266742801041	88,030000165
1985	AN8503DM	15	3,48600000015181	68,4156378071474	66,499999892
1985	AN8503DM	16	3,57853239865103	76,2700188044768	44,289999666
1985	AN8503DM	17	3,6031795163402	42,439562880776	73,029999662

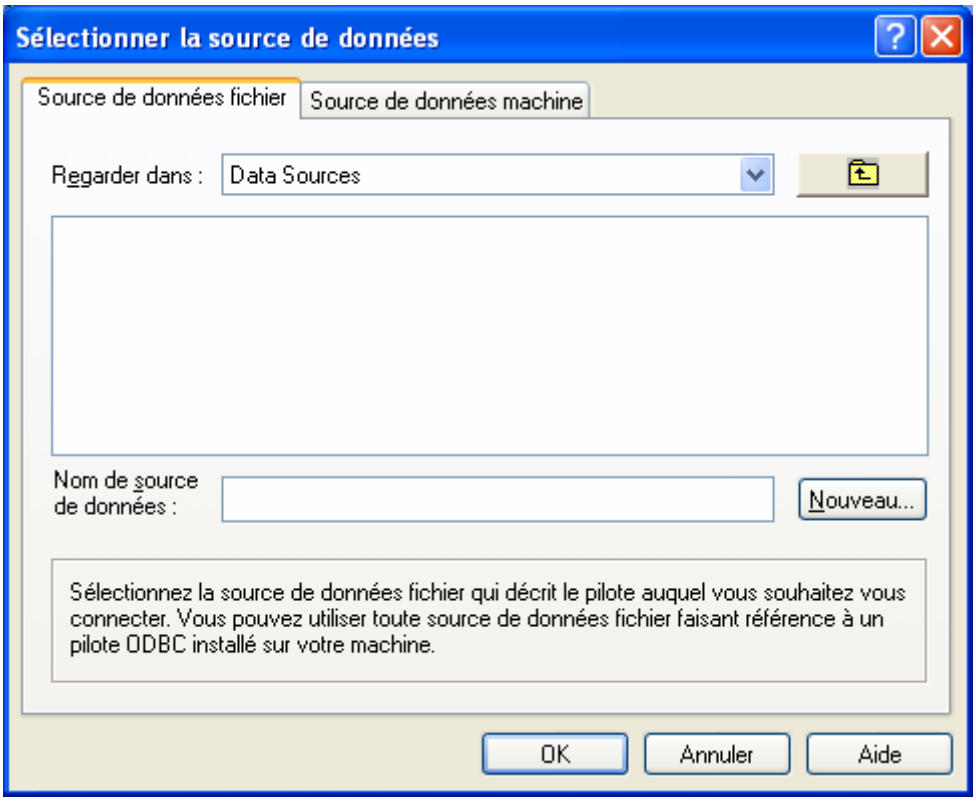
calcul_abundance: cette requête permet d'obtenir l'abondance par espèce et par station. L'indice est calculé en rapportant la capture de l'espèce à la surface chalutée. Cette dernière est calculé en multipliant l'ouverture du chalut par la distance parcourue (durée du trait * vitesse * 0.182)

Editeur de données

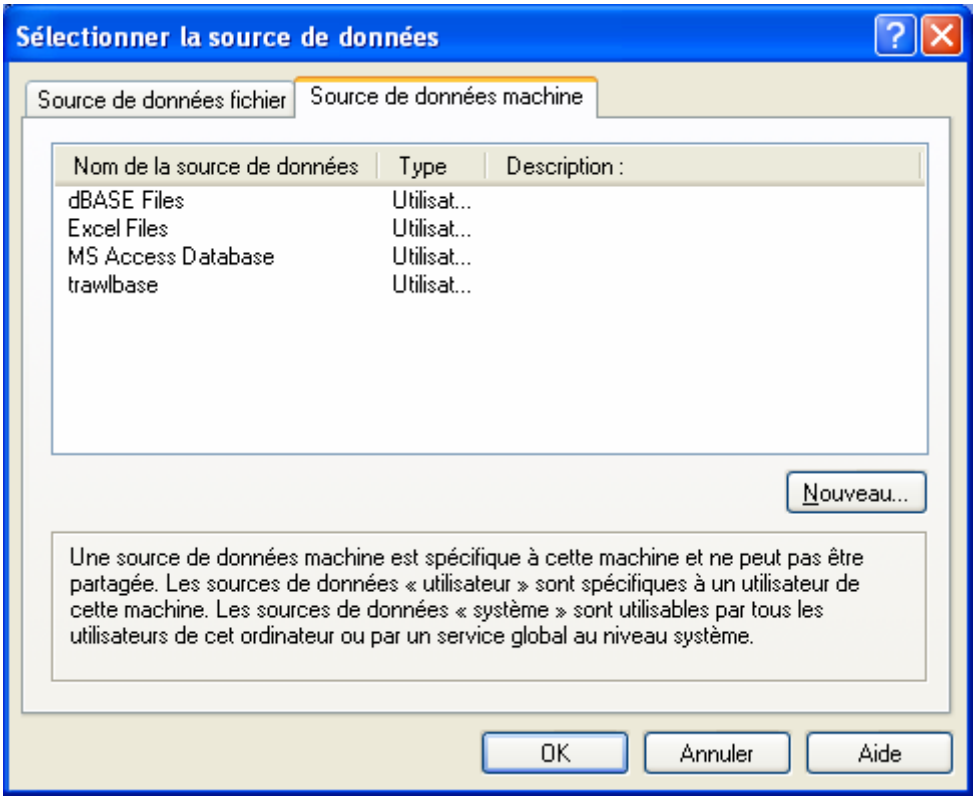
	campagne	station	lat	long	taxon
1	AN8503DM	1	9.467	-13.883	ALBULA VULPES
2	AN8503DM	1	9.467	-13.883	BALISTES CAROLINENSIS
3	AN8503DM	1	9.467	-13.883	CYNOGLOSSUS CANARIENSIS
4	AN8503DM	1	9.467	-13.883	DENTEX GIBBOSUS
5	AN8503DM	1	9.467	-13.883	EPHIPPION GUTTIFER
6	AN8503DM	1	9.467	-13.883	EPINEPHELUS AENEUS
7	AN8503DM	1	9.467	-13.883	LAGOCEPHALUS LAEVIGATUS

Utilisation de la librairie RODBC

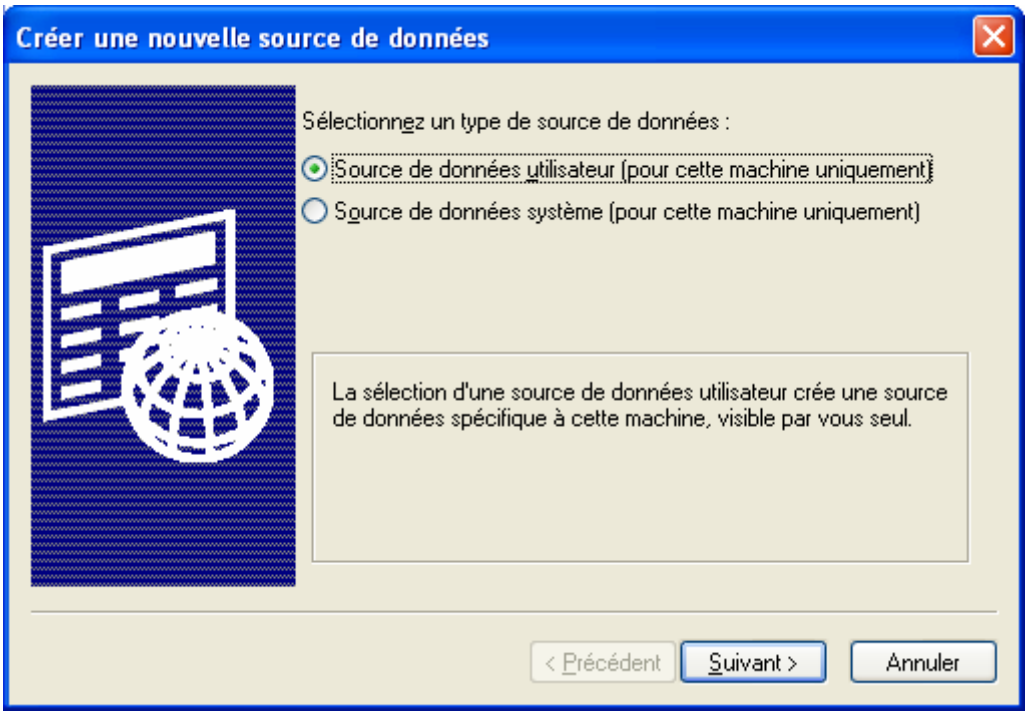
```
library(RODBC) ; #les majuscules on leur importance

channel <- odbcDriverConnect("") #pour un démarrage interactif de la connexion
```

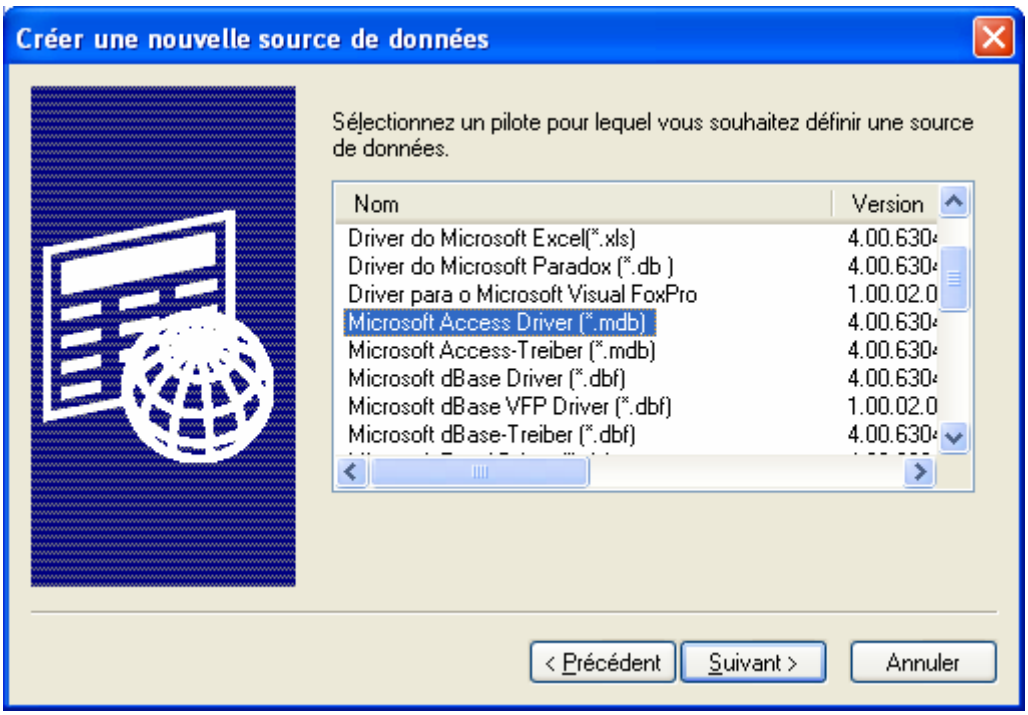


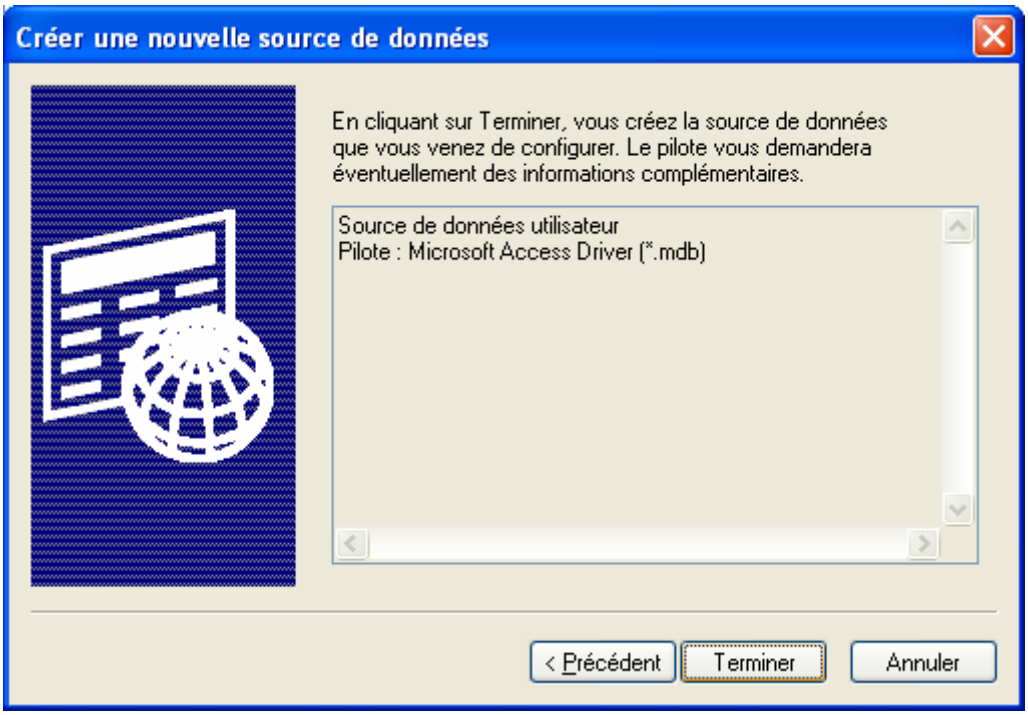
On choisi source de données machine :



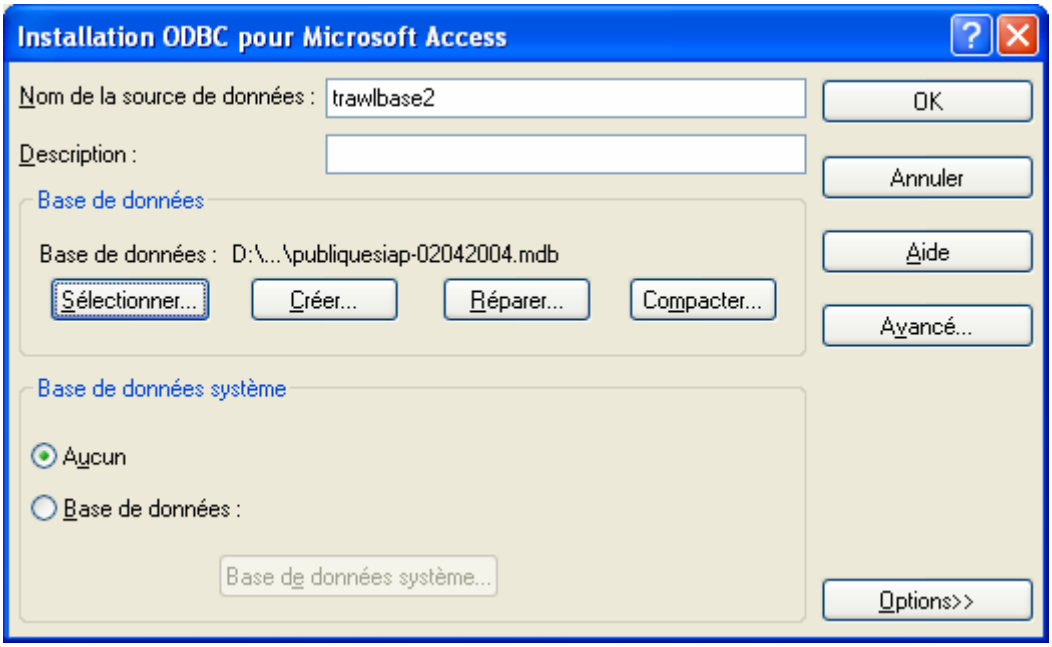
On en crée une nouvelle


Créer une nouvelle source de données
Sélectionnez un type de source de données :
Source de données utilisateur (pour cette machine uniquement)
Source de données système (pour cette machine uniquement)
La sélection d'une source de données utilisateur crée une source de données spécifique à cette machine, visible par vous seul.
< Précédent
Suivant >
Annuler


Créer une nouvelle source de données
Sélectionnez un pilote pour lequel vous souhaitez définir une source de données.
Nom
Version
Driver do Microsoft Excel(*.xls)
Driver do Microsoft Paradox (*.db)
Driver para o Microsoft Visual FoxPro
Microsoft Access Driver (*.mdb)
Microsoft Access-Treiber (*.mdb)
Microsoft dBase Driver (*.dbf)
Microsoft dBase VFP Driver (*.dbf)
Microsoft dBase-Treiber (*.dbf)
< Précédent
Suivant >
Annuler



On donne un nom à notre connexion : trawlbase2 et on selectionne la base de donnée publique siap.mdb que l'on va utiliser



channel ;

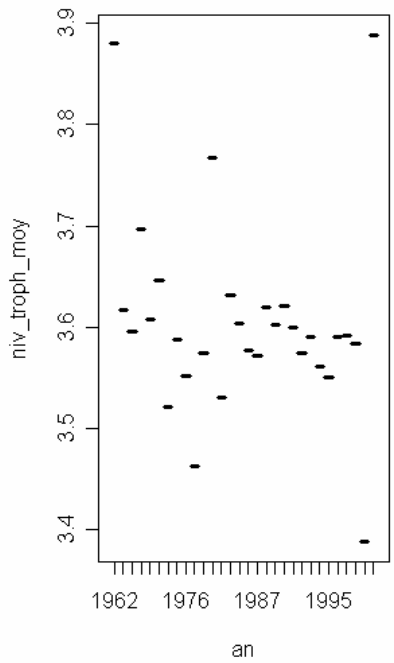
Resultat :

RODB Connection 2
Details:
case=nochange

```
DSN=trawlbase2
DBQ=D:\DONNEES\internationale\publiquesiap-02042004.mdb
DriverId=25
FIL=MS Access
MaxBufferSize=2048
PageTimeout=5
```

Maintenant que l'on a défini la connexion, les prochaines fois nous n'aurons plus qu'a taper :

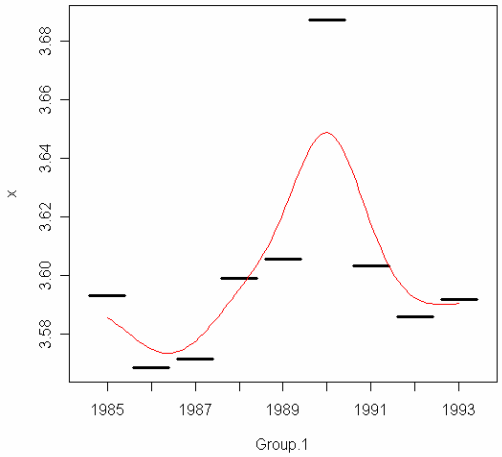
```
library(RODBC) ; #les majuscules on leur importance
channel <- odbcConnect("trawlbase")



niv_troph<-sqlQuery(channel, "select * from calcul_niv_troph"); #on met dans la data frame niv_troph, le résultat de la requete calcul_niv_troph réalisé sur la connexion channel


fix(niv_troph) ; #édite les valeurs de la matrice



resultat<-aggregate(niv_troph[,4],list(niv_troph$an),mean) ;
#on agrége les données de la colonne 4 selon le critère niv_troph$an avec la fonction mean. Ceci me donne la moyenne des niveaux trophique par an
resultat<-aggregate(niv_troph[4],list(niv_troph$an,niv_troph$campagne),mean) ;
niv_troph[order(niv_troph$an),] ; #me renvoie les lignes triées par an
niv_troph[order(niv_troph$campagne,niv_troph$an),] ; #trié par campagnes et par an


niv_troph[order(niv_troph$campagne,niv_troph$an),] [1 :100,];# mes 100 première lignes de mon tri
resultat<-aggregate(niv_troph[4],list(niv_troph$an),mean) ;
plot(resultat) ;
```



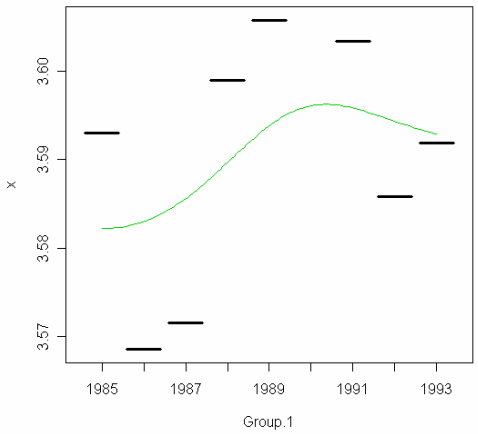
Un début d'enchaînement avec la boucle for :

```
campagnes<-levels(niv_troph[,2]) ; #on crée un vecteur des campagnes, pour pouvoir créer une selection de campagne

maselection<-niv_troph[niv_troph[,2] %in% c(campagnes[c(1,4)]),] ; #ma selection est une partie de la table niveau trophique, celle ou les campagne sont les n° 1 et 4 de la table campagne.

maselection<-niv_troph[niv_troph[,2] %in% c(campagnes[1 :21]),] ; # je prends les campagnes 1 à 21 – André Nizery
```

```
# Une courbe avec une moyenne lissée
resultat<-aggregate(maselection[,4],list(maselection$an),mean) ; #moyenne des niveaux trophique par an

plot(resultat);
lines(ksmooth(as.numeric(resultat$Group.1), resultat[,2],
"normal", bandwidth=2), col=2);
```




```
#La même chose en enlevant la ligne 6


plot(resultat[-6,])
lines(ksmooth(as.numeric (resultat[-6,1]), resultat[-6,2], "normal", bandwidth=5), col=3)
```



2. Quelques traitements.

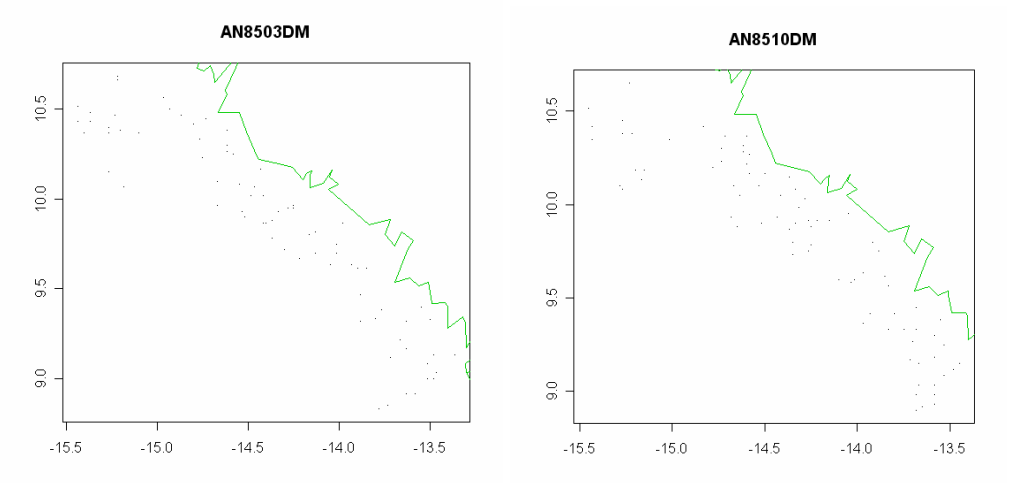
21- Cartographie

#on se reconnecte

```
library(RODBC)
```

```
channel <- odbcConnect("trawlbase")

tab_abondance<- sqlQuery(channel, "select * from calcul_abondance where
campagne like 'AN%' and taxon like 'pseudotolithus elongatus%'");

library(methods)
dyn.load("bez/Geoslib.dll")
load("bez/RGeoS.RData");

tab_abondance<- sqlQuery(channel, "select * from calcul_abondance ‘);
campagnes<-levels(tab_abondance[,1])
```

```
for (I in 1:length(campagnes))
{
tmp<-tab_abondance[tab_abondance[,1]==campagnes[I],];
symb.s(tmp[,4],tmp[,3], tmp[,6],coast=T, titl = campagnes[I] );
readline();
}
```



…

```
write.table(resultat, "data.csv",sep=';'); # a tout moment on peut sauver une matrice de resultat
```

```
resultat<-
aggregate(tab_abondance$abondance,list(tab_abondance$campagne),sum)
```

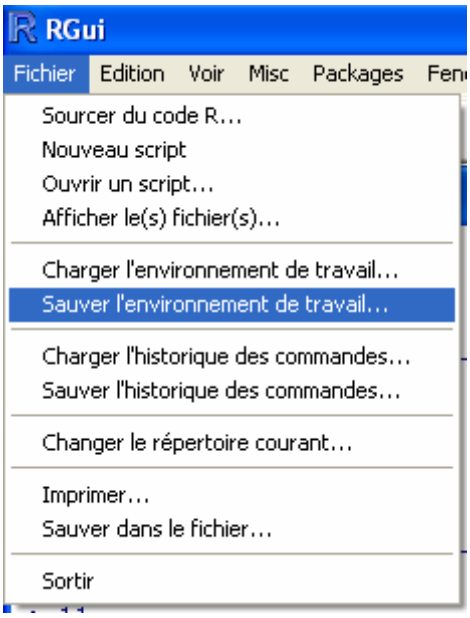
3. Divers .

1.4. Installations d'un package.

```
install.packages('tree') ;
#Là il demande sur quel serveur aller chercher le package, il suffit de lui donner un site pas #trop loin.

install.package("mapdata")
#Mandatory package to represent coasts with a fine resolution (appropriate for small areas).
map("world")
map("world", xlim=c(0,10),ylim=c(44,46))
map("worldHires", xlim=c(0,10),ylim=c(44,46))
```

1.5. Sauvegarde d'un environnement de travail

On peut sauvegarder les données et les fonctions que l'on utilise pour les transmettre à quelqu'un d'autre par exemple.



4. Working in R

```
my.data=read.csv(“:/my.file.csv”,sep=”;”)
```

or

```
my.data=pseudo_typus3
```

```
names(my.data),class(my.data), dim(my.data)
hist(abondance)
```

Command that fails.

```
> hist(my.data$abundance)
```

Command that works.

```
search()
attach(my.data)
search()
```

```
hist(abondance)
hist(abundance[an= =1995])
hist(abundance[an != 1995])
hist(abundance[an ==1995 & mois >= 3 & mois <= 6])
```

Possible logical operators == ; != ; > ; >= ; & (and); | (or)

```
detach(my.data)
search()

args(hist)
```

Allows to see all the arguments of the function

```
?hist
```

Allows accessing the ad hoc help file

```
?par
```

Function that controls all the graphical parameters

```
hist(abundance[an= =1995], xlab="blabla",col="red")
```

How to build in your own functions : fix()

```
hist
```

Allows seeing what the code of the function is.

```
strate_geo
```

```
Erreur : objet "strate.geo" non trouvé
```

```
strate_geo<-function(vecteur,grille)
 {
```

```
vecteur<-vecteur%/%1+((vecteur-vecteur%/%1)%/%grille)/10;
return(vecteur);
}

ls()
```

To see your new function in the list of R object existing in your working environment.
For longer and/or more difficult functions, you should use the fix() function.

```
strate_geo<-fix()
```

Recommandation :
Do not use "_" character in the names of your R objects as they used to be equivalent to an assignment "<-" or "=" in older S and R versions.

```
simple.symb = fix()

function(z, x = long, y = lat, sel = NA, add=T, ...)
{
#########################################################
#
#   Proportional representation of the non negative values of z.
#
#   z               Can be a vector, a matrix with columns x, y, z or
#                     a grid list(x,y,z).
#                     The variable to be represented is supposed to have only
#                     POSITIVE values (typically a density). NA are allowed.
#   x               Optional. A vector of same length than z. NA are allowed.
#   y               idem.
#   sel             Optional. Condition for selecting active points.
#   ...             Any argument of the Splus function "par". E.g.: lab = c(2,2,7).
#                     Used in the 'plot' function.
#
```

```
#   symb.s          --> symb.lgd
#
#####################################################
    if(miss(sel))
            sel <- !is.na(x * y * z)
    else {
            valid <- !is.na(x * y * z)
            sel <- sel & valid
    }
    x <- x[sel]
    y <- y[sel]
    z <- z[sel]
    if(length(z) != 0) {
            if(add == F)
                    plot(x, y, type = "n", ...)
            symbols(x, y, rectangles = matrix(c(sqrt(z), sqrt(z)), ncol = 2), add = T)
    }


search()
attach(my.data) # if needed
simple.symb(abundance)> simple.symb(abundance,sel=an == 1995)
```

Selection of data within a polygon
Presentation of the pol.in() function.
Use of a Fortran subroutine.

```
my.pol=locator(type=”n”)
simple.symb(abundance,sel=pol.in(long,lat,my.pol))
simple.symb(abundance,sel=pol.in(long,lat,my.pol)&an==1995)
```

Creating loops with the function for(){}

```
par(mfrow)c(3,2)
for(i in an){
```

```
simple.symb(abundance,sel=pol.in(long,lat,my.pol)&an==i)
map("world",add=T)
title(as.character(i))
}
par(mfrow=c(1,1))
```

How to improve, enlarge an existing function (either native in R or from yours) ?

```
symb.istam = simple.symb
fix(symb.istam)

function(z, x = long, y = lat, sel = NA, xlim = NA, ylim = NA,
xlab = NA, ylab
    = NA, zmin = 0, zmax = NA, inches = 0.2,
   rectangle = T, coast = F, titl = "", add = F, cex = 1,
   cex.lgd = 0.8 * cex, legend = 4, col = seq(1, 4),
   margin = NA, ...)
{
################################################
#
#  Proportional representation of the non negative values of z.
#
#  z           Can be a vector, a matrix with columns x, y, z or
#                 a grid list(x,y,z).
#                 The variable to be represented is supposed to
have only
#                 POSITIVE values (typically a density). NA are
allowed.
#  x           Optional. A vector of same length than z. NA are
allowed.
#  y           idem.
#  sel         Optional. Condition for selecting active points.
#  xlim        Optional. Fix the range of x reporesented in the
graph.
#                 Must be given in the same units than x and y.
#  ylim        Idem for y.
#  xlab        Label for the x axis. By default the names of the x
variable.
#  ylab        Idem for the y axis.
```

```
#  zmin        A positive number. Values of z <= zmin are figured
by a cross.
#                 By default, null values are represented by a
cross.
#  zmax        Values of z > zmax  are figured by a triangle.
#                By default, zmax = max(z). For z=zmax the square
size is "inches".
#  inches            Controls the size of the symbols.
#  rectangle   Optional. Logical. If T symbols are rectangles.
#              If F symbols are circles.
#  coast       A logical value for representing or not the coast
lines.
#  titl        Optional. A character string representing the
title.
#                 titl=""  The default. The variable's name and its
maximum are indicated.
#                 titl=" "       NO TITLE
#                 titl="cod"     title is "cod"
#  add         A logical factor to control whether the
proportional representation has
#                 to be added or not to an existing graph.
#  cex         Controls the size of the text and labels.
#  cex.ldg           Controls the size of the text of the legend.
#  legend      A number coding for the legend.
#                 legend = 0     no legend is performed
#                 legend = 1     legend at the bottom left corner
of the figure
#                 legend = 2     legend at the upper left      "
#                 legend = 3     legend at the upper right     "
#                 legend = 4      legend at the bottom right    "
#                 legend = 5      legend where indicated clicking
on the figure
#  col         Vector giving respectively the color for :
#                    col[1] the cross,
#                    col[2] the square,
#                    col[3] the coast lines,
#                    col[4] the isobaths.
#              Can be set to c(1,1,1,1) for B&W representation.
#  margin            Value of the margin of the plot in INCHES.
#                Default in Splus is par(mai=c(0.714, 0.574,
0.574, 0.294))
#                 for a single figure.
```

```
#  ...          Any argument of the Splus function "par". E.g.: lab
= c(2,2,7).
#                  Used in the 'plot' function.
#
#  symb.s       -->   gridexp
#         -->   dg2nm
#         -->   topo --> subgrid
#               -->   symb.lgd
#
#####################################################
   if(miss(sel))
         sel <- !is.na(x * y * z)
   else {
         valid <- !is.na(x * y * z)
         sel <- sel & valid
   }
   x <- x[sel]
   y <- y[sel]
   z <- z[sel]
   if(miss(xlab))
         xlab <- deparse(substitute(x))
   if(miss(ylab))
         ylab <- deparse(substitute(y))
   if(miss(xlim))
         xlim0 <- range(x)
   else xlim0 <- xlim
   if(miss(ylim))
         ylim0 <- range(y)
   else ylim0 <- ylim
   if(miss(xlim))
         xlim <- range(x)
   if(miss(ylim))
         ylim <- range(y)
   if(miss(zmax))
         zmax <- max(z)
   if(length(z) != 0) {
         if(add == F) {
               if(!miss(margin))
                 par(mai = margin)
```

```
                plot(x, y, type = "n", xlim = xlim, ylim = ylim,
xlab = xlab, ylab = ylab, cex = cex, ...)
          }
          sel2 <- z <= max(0, zmin)
          if(sum(sel2) > 0)
                points(x[sel2], y[sel2], pch = 3, col = col[1])
          sel3 <- z > zmin & z <= zmax
          if(sum(sel3) > 0) {
                if(rectangle)
                      symbols(x[sel3], y[sel3], rectangles =
matrix(c(sqrt(z[sel3]), sqrt(z[sel3])), ncol = 2), inches =
(inches * max(z[sel3]))/zmax, add = T, col = col[2])
                if(!rectangle)
                       symbols(x[sel3], y[sel3], circles =
sqrt(z[sel3]), inches = (inches * max(z[sel3]))/zmax, add = T, col
= col[2])
          }
          sel4 <- z > zmax
          if(sum(sel4) > 0)
                points(x[sel4], y[sel4], pch = 2, cex = 5 *
(inches/0.20), col = col[2])
          if(titl == "")
                title(paste(deparse(substitute(z)), "(max =",
signif(max(z)), ")"), adj = 0, cex = cex/1.5)
          if(titl != "")
                title(titl, cex = cex/1.5)
          if(coast == T)
                map("world", add = T, xlim = xlim, ylim = ylim,
color = col[3])
          if(legend > 0)
                symb.lgd(zmin = zmin, maxz = max(z), zmax =zmax,
inches = inches, col = col, cex = cex.lgd, pos = legend)
    }
    if(!miss(margin))
          par(mai = c(0.714, 0.574,0.574, 0.294))
}
```

Function source()

```
dump("symb.istam","symb.istam.R")
```

If you send it to a colleague, he can simply load it by

```
source("symb.istam.R")
```

Check by using ls() before and after the source().

Using scripts

A set of command you want to re-run routinely can be set into a script file and run very easily.

References

Emmanuel Paradis :
http://cran.r-project.org/doc/contrib/Paradis-rdebuts_fr.pdf

http://lib.stat.cmu.edu/R/CRAN/