R TUTORIEL

INTERFACES GRAPHIQUES

R-users – Montpellier 2014

Thomas VERRON

Agenda

Horaire	Agenda item
14h00 - 15h15	Introduction & contexte Exemples de développement R et les widgets
15h15 - 15h30	Break
15h30 - 17h00	Application Construction d'une interface graphique

Introduction & contexte

Thomas VERRON
PhD Biostatistic
Responsable du service “datamining & statistical studies”

Imperial Tobacco

Regulatory
RESEARCH

Laboratories

Vérifier que nos produits sont conformes à la réglementation des pays ou ils sont vendus

Corporate Affairs

Négocier avec les autorités compétentes sur les réglementations actuelles et futures

Introduction & contexte

Pourquoi R?

R est un langage de programmation flexible et puissant qui permet aux utilisateurs de développer des programmes sur mesure.

Inconvénient:
Langage peu intuitif pour des personnes n'ayant pas de notions en R.

Notre objectif:
Développer des interfaces graphiques pour rendre R intuitif et utilisable par tous.

De nombreux packages sont disponibles dans R:

Bwidget, Rcommander, Rtcltk, Rtcltk2, tkrplot…

TCLTk
Tool Command Language

Tcl-Tk

Tcl

Tool Command Language (abréviation : Tcl) est un langage de scripts conçu par John Ousterhout et son équipe à l'université de Californie à Berkeley (multi-plateformes, extensible, facile à apprendre).

MCMLXXXVIII
1988

Tk

John Ousterhout développe une extension pour Tcl appelée Tk qui est une bibliothèque pour créer des interfaces graphiques portables.

MCMXC
1990

Tcl-Tk

Tcl-Tk est la combinaison des languages Tcl et Tk.

R & Tcl-Tk

- tcltk: Bibliothèque disponible dans R qui contient les principales commandes Tcl et widgets Tk.
- tcltk2: Extension de la bibliothèque tcltk qui permet d'accéder à des styles et des fonctions supplémentaires.
- tkrplot: Extension pour insérer des graphiques R dans les widgets Tk.

Pour certaines fonctions, il peut être nécessaire d'installer des applications supplémentaires (ActiveTcl…).

R & Tcl-Tk

R & Tcltk permet de créer des interfaces utilisateur sur mesure pour votre application.

L'inconvénient majeur réside dans la qualité de l'aide proposée dans les packages pour utiliser les fonctions.

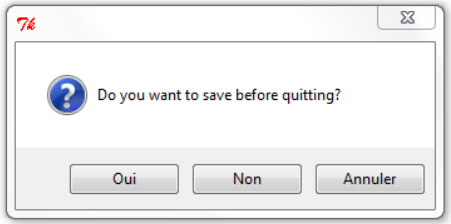
Liens utiles:

http://www.sciviews.org/_rgui/tcltk/
http://rstat.ouvaton.org/?article19/creer-des-applications-graphiques-avec-r
http://rug.mnhn.fr/semin-r/PDF/semin-R_tcltk_JBardin_310512.pdf

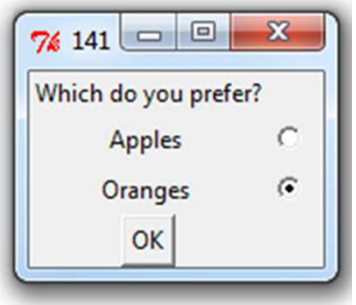
Exemples

Button

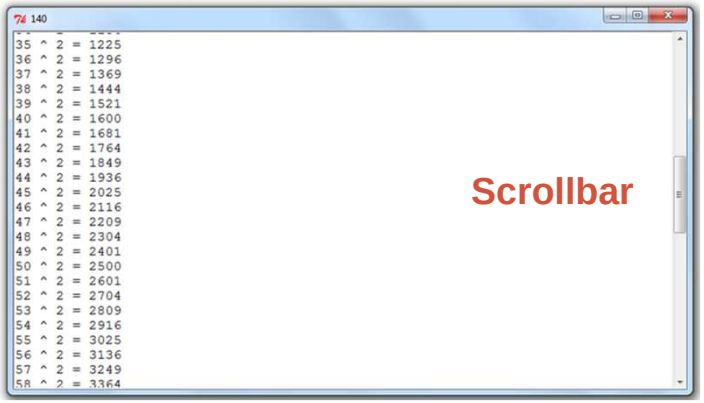
Warning messages



Radiobutton

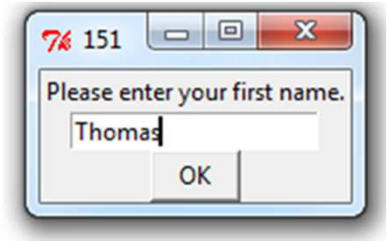


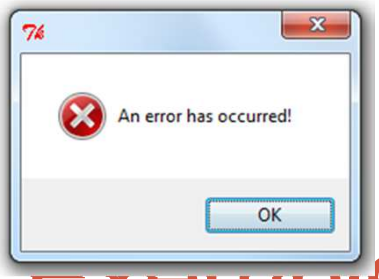
Text window



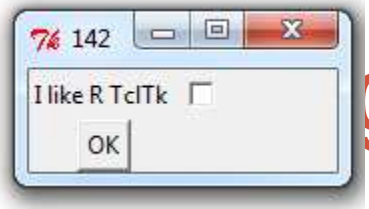
Scrollbar

Edit box

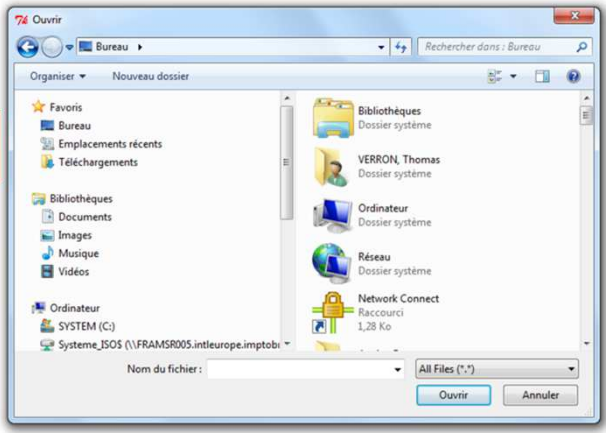




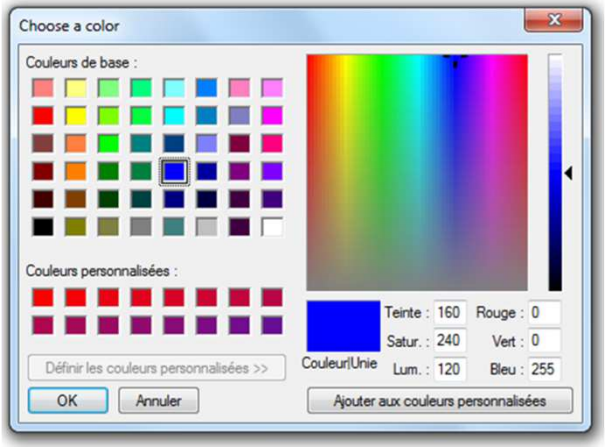
checklist



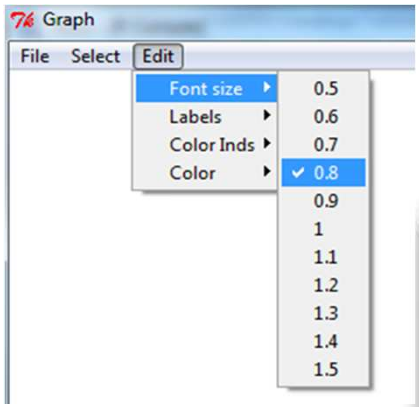
File Open/Save dialogs



Color-selection widget



Menus



List Box

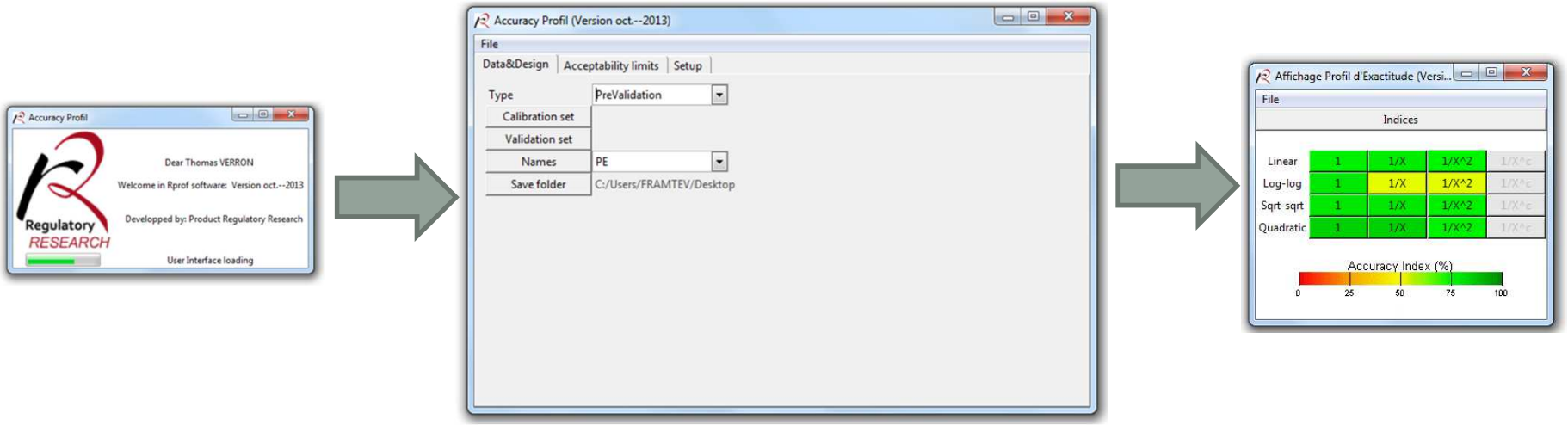
Drop-down combo Box

Exemples de développement

Exemple 1:
Profil d'exactitude / Accuracy Profile

Objectif: Validation de méthodes d'analyse dans les laboratoires en utilisant le profil d'exactitude.

Laboratoires: Aider les techniciens à valider leurs méthodes d'analyse et à réaliser des dossiers d'accréditation.



Fenêtre de chargement

Chargement des données et options

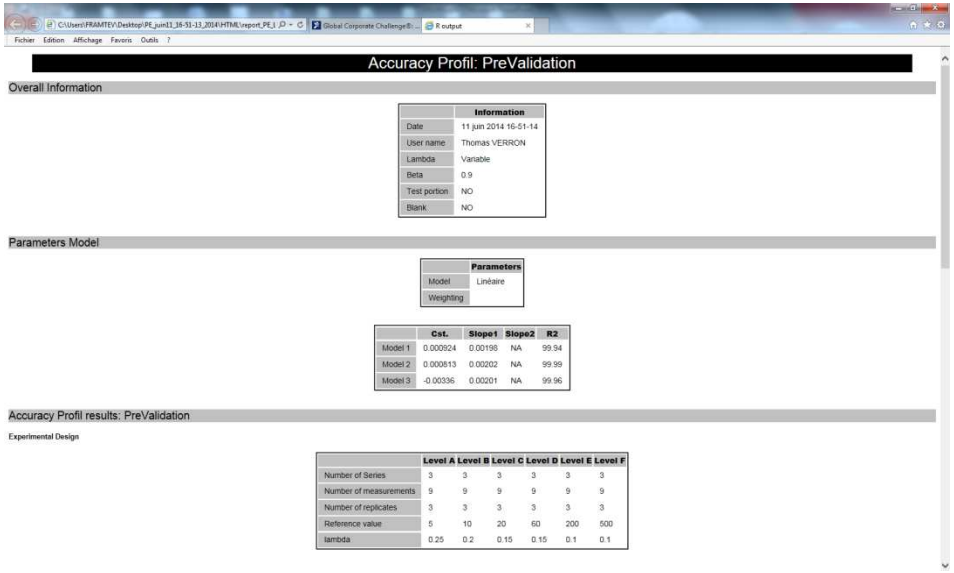
Résultats et accès aux rapports

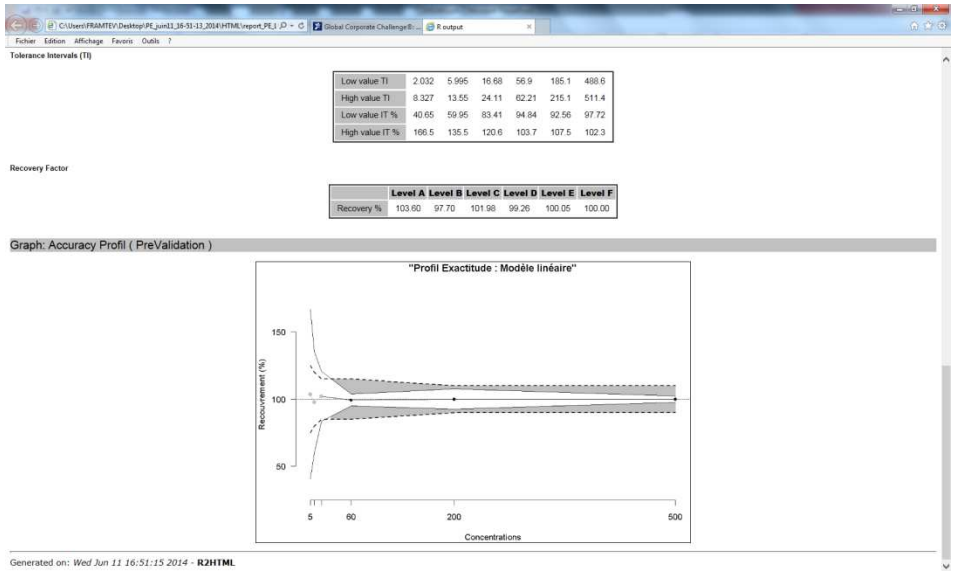
Exemple 1:
Profil d'exactitude / Accuracy Profile

Objectif: Validation de méthodes d'analyse dans les laboratoires en utilisant le profil d'exactitude.

Laboratoires: Aider les techniciens à valider leurs méthodes d'analyse et à réaliser des dossiers d'accréditation.

Rapport HTML

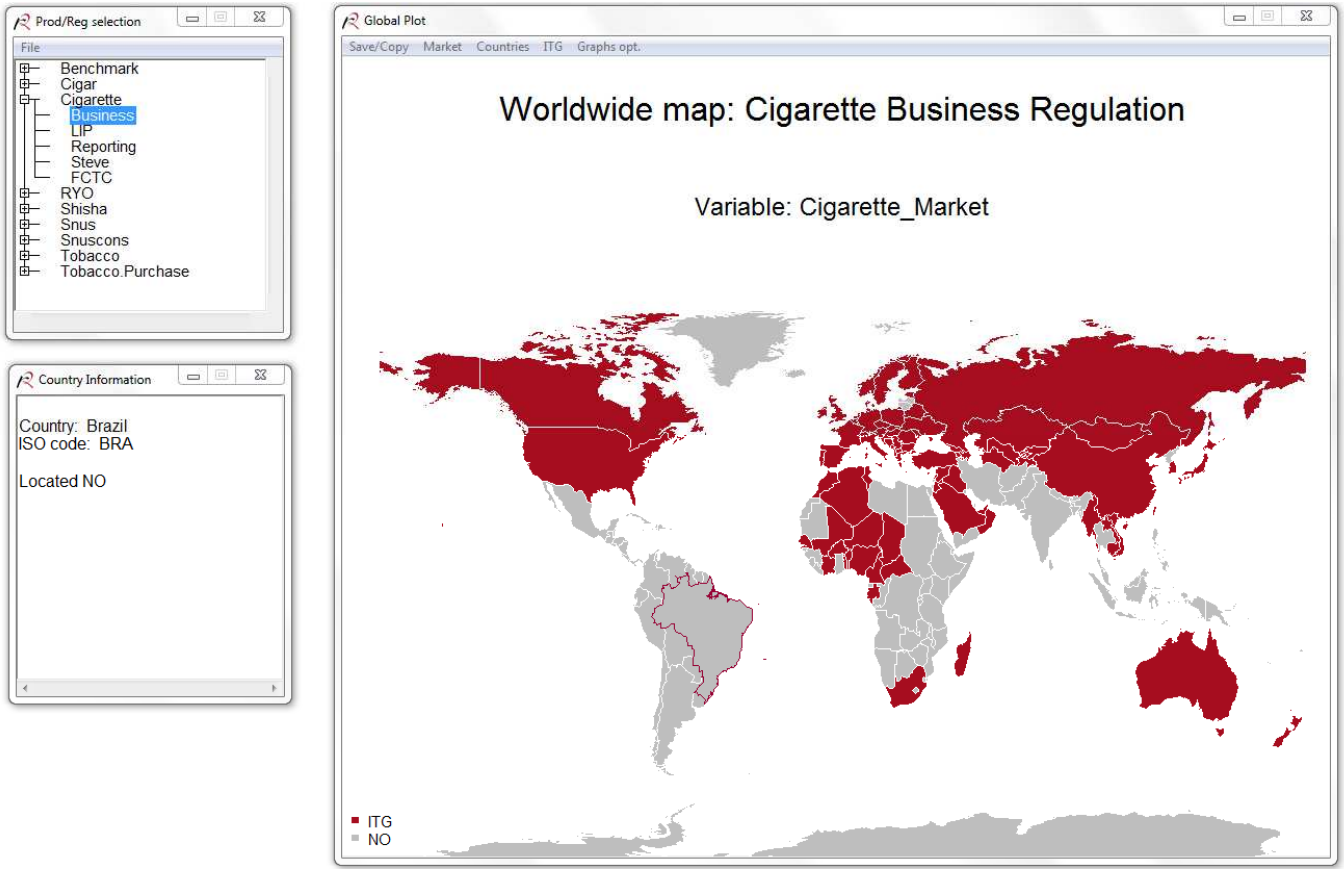




Exemple 2: DREAM
Dynamic Regulatory Environment Assisted by Map

Objectif:
Suivre l'évolution des réglementations par produit à travers le monde.

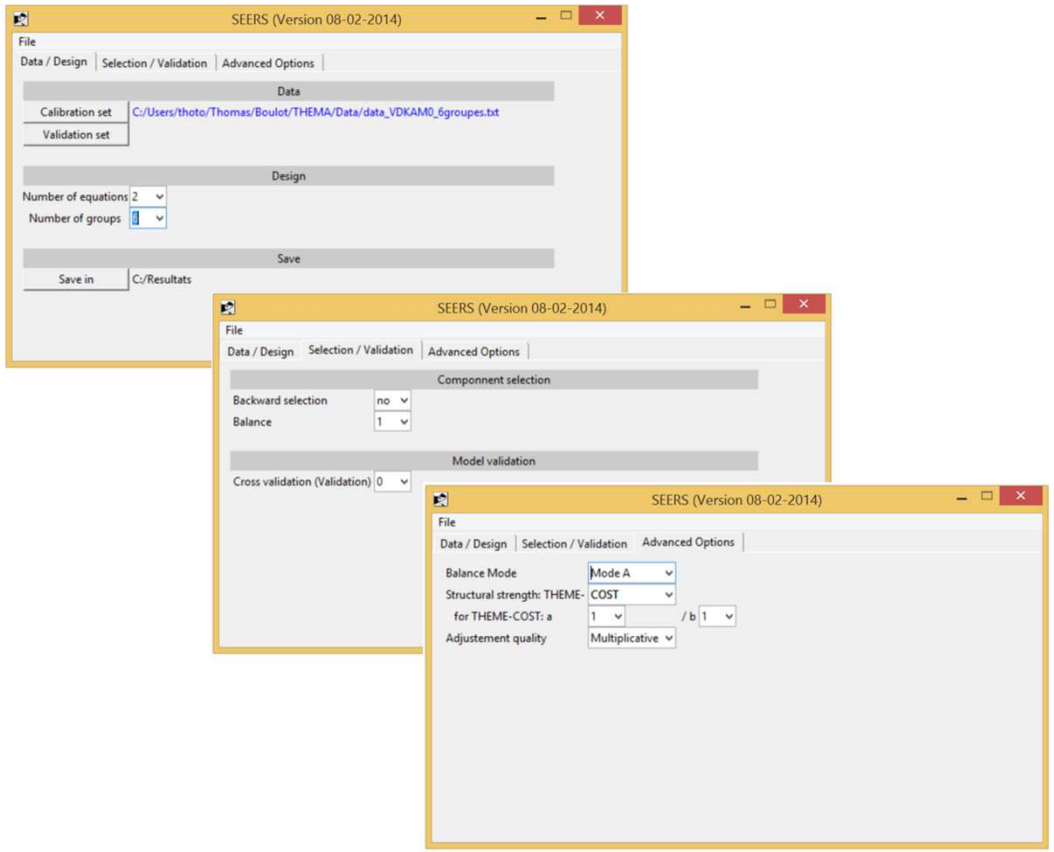
Corporate & developpement:
Faciliter le lancement de nouveaux produits.

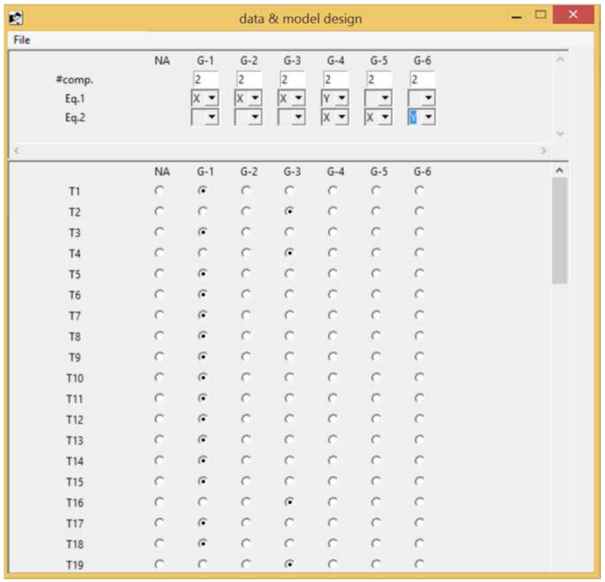


Exemple 3: THEME-SEER

Objectif:
Explorer et tester des modèles à équations structurelles.

Tous scientifiques possédant des données multivariées.

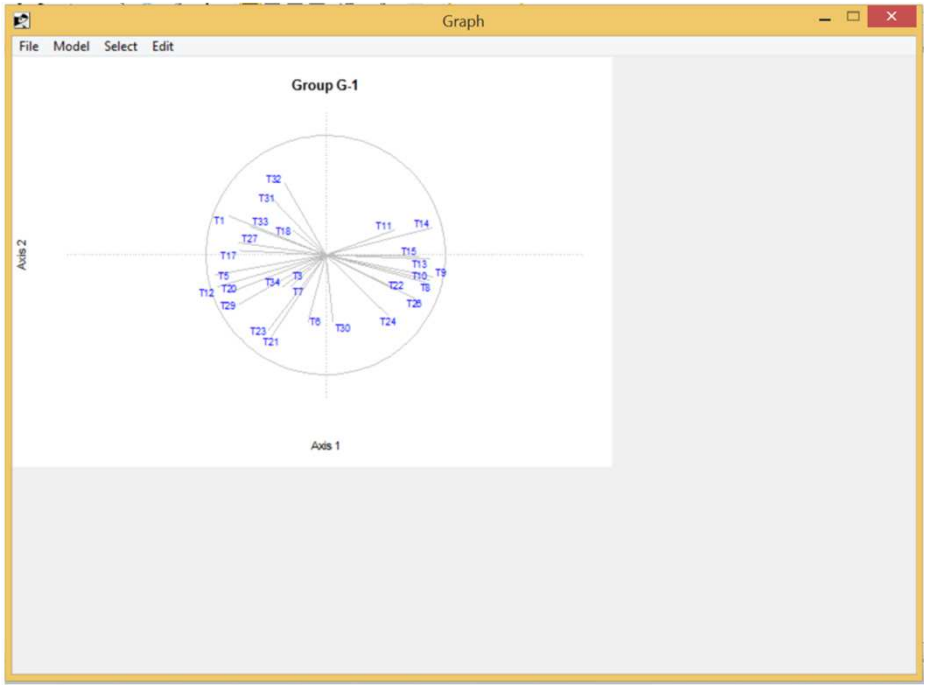


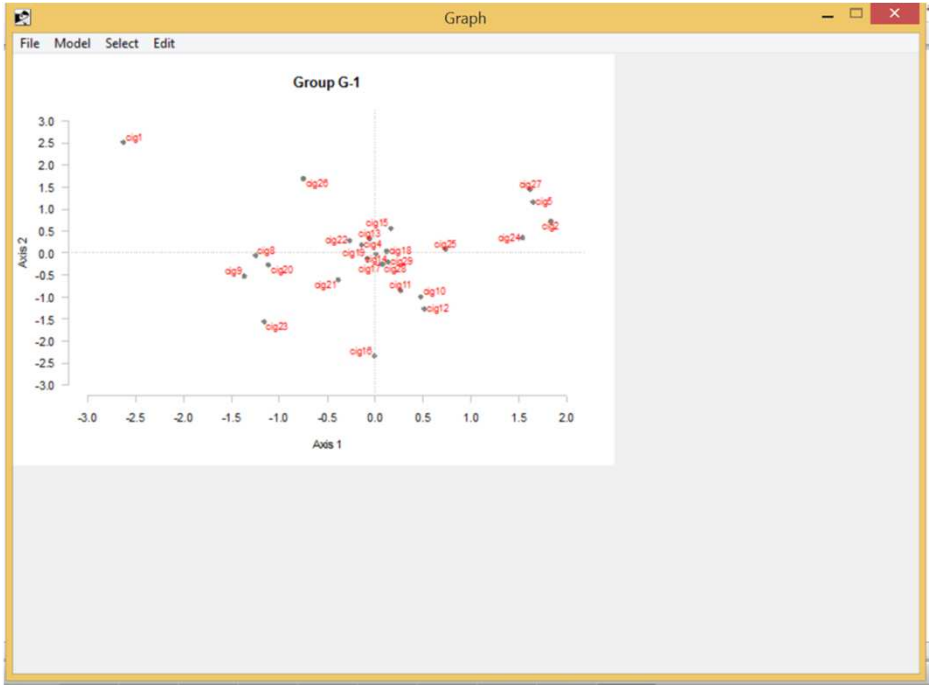


Exemple 3: THEME-SEER

Objectif:
Explorer et tester des modèles à équations structurelles.

Tous scientifiques possédant des données multivariées.





Process

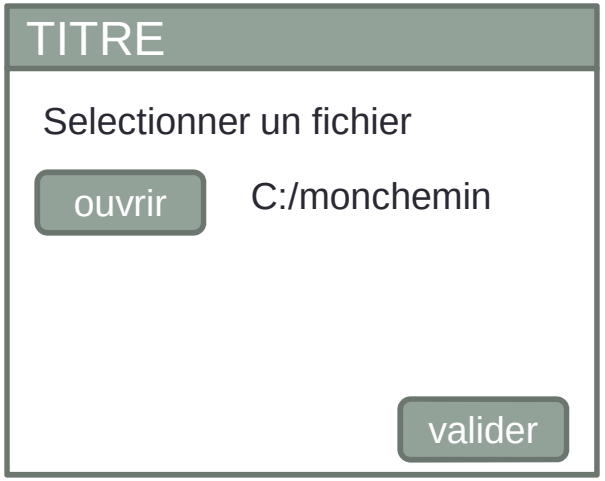
1. **Créer une fenêtre graphique:**
   Titre, taille, emplacement, frame

2. **Créer un objet:**
   Texte, bouton, menu déroulant, barre de défilement, checkbox…

   Associer à un objet:
   - **une fonction:** *ex. ouvrir un document*
   - **et/ou une variable:** *ex. chemin*

3. **Afficher les objets**
   Positionner dans la fenêtre (tkgrid, tkpack…)

4. **Exécuter les modifications:**
   Stocker les variables
   Fermer la fenêtre graphique

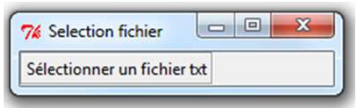


1 Créer une fenêtre graphique:

Titre, taille, emplacement, frame

Explications	Commandes
Créer la fenêtre	`wind1 <- tktoplevel()`
Ajouter un titre	`tkwm.title(wind1, "Selection fichier")`

2 Créer un objet “Texte”



Options: couleur du texte, couleur du fond...

Explications	Commandes
Créer un label	`tklabel(«Parent»,text=« texte »,…)`

Où: tclvalue(variable)

Arguments indispensables:

L’objet contient la fenêtre principale,
un sous-élément de la fenêtre principale: « frame » , « notebook »;

Texte à afficher / Variable qui contient le texte à afficher.

2 Créer un objet “button”

Options: couleur du texte, couleur du fond...

Explications	Commandes
Créer un Bouton	`tkbutton("Parent",text="OK", command=function()myfunc(),…)`

Arguments indispensables:

L’objet contient la fenêtre principale,
un sous-élément de la fenêtre principale: « frame » , « notebook »;

Texte à afficher sur le bouton;

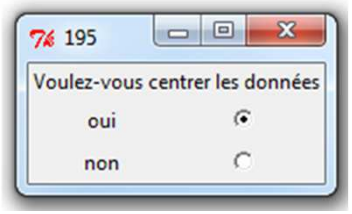
Fonction à exécuter quand on clique sur le bouton.

2 Créer un objet “checklist”

Explications	Commandes
Créer un Checklist	`cbvalue<-tclVar(0)` `tkcheckbutton("Parent",variable=cbvalue, text="OK" ,command=function()…)`

Variable: contient la valeur 1 si la case est cochée et 0 sinon (par défaut 0)

2 Créer un objet “radiobutton”



Explications	Commandes

Créer un Radiobutton	`rbValue<-tclVar("Y")`
	`tkradiobutton("Parent",variable=rbValue,`
	`value="Y", command=function()…)`
	`tkradiobutton("Parent",variable=rbValue,`
	`value="N", command=function()…)`

Arguments indispensables:

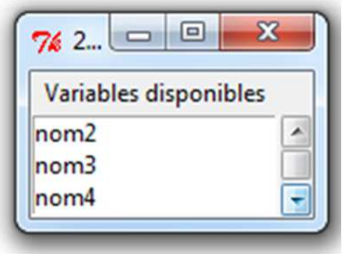
L’objet contient la fenêtre principale,
un sous-élément de la fenêtre principale: « frame » , « notebook »;

Value: valeur à attribuer à la variable quand on coche le radiobutton;

Variable: contient la valeur (« value ») du radiobutton sélectionné «Y» ou «N»
par défaut *rbValue<-tclVar(«Y»).*

Option: Command : fonction à exécuter quand on coche le radiobutton.

2 Créer un objet “listbox”



Explications	Commandes

Créer une listbox	`tkobjet<-tklistbox("Parent",height=3,`
	`  selectmode="extended",`
	`  yscrollcommand=function(…)tkset(scrbar,…))`
	`scrbar<-tkscrollbar("Parent",repeatinterval=5,`
	`  command=function(...)tkyview(tkobjet,...))`
	
	`for (i in paste("nom",1:6,sep="")){`
	`  tkinsert(tkobjet,"end",i)}`
	
	`tkcurselection(tkobjet)`

Arguments indispensables:

L’objet contient la fenêtre principale,
un sous-élément de la fenêtre principale: « frame » , « notebook »;

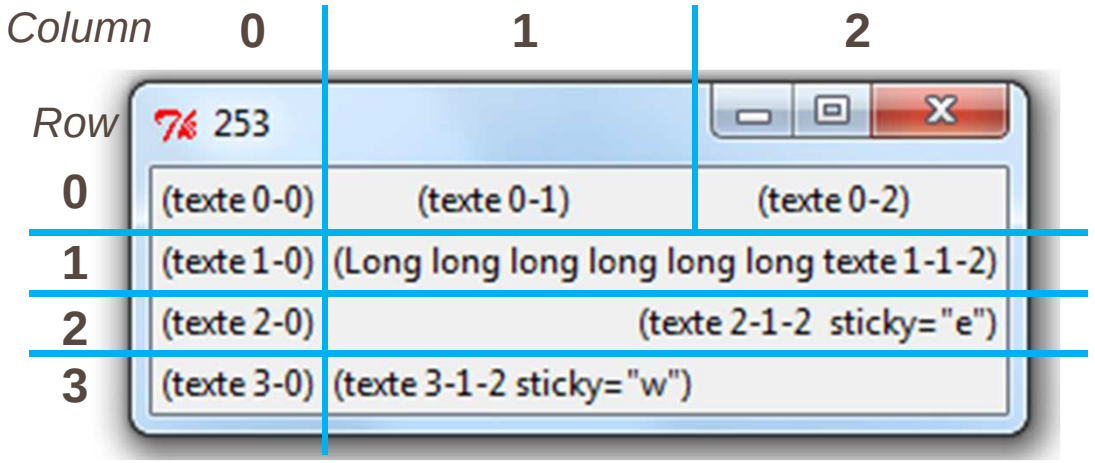
Selectmode : un par un, multiple…

yscrollcommand : pour rajouter une barre de défilement verticale;

Command : fonction à exécuter;

tkcurselection : pour récupérer les lignes sélectionnées (commence à 0).

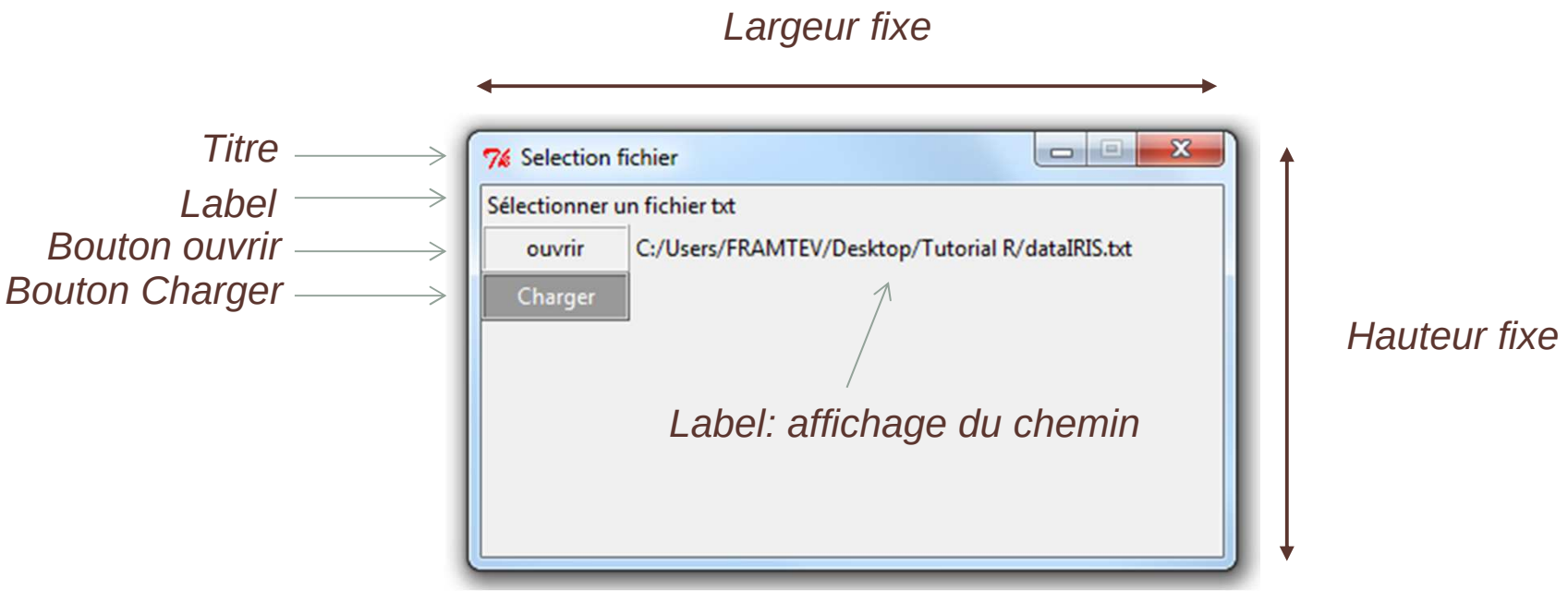
3 Positionnement



Positionnement	Commandes
(Texte 0-0)	`tkgrid(tklabel("Parent",text="(texte 0-0)"),row=0,column=0)`
(Long … texte 1-1-2)	`tkgrid(tklabel("Parent",text="(Long long long long long long texte 1-1-2)"), row=1,column=1,columnspan=2)`
(Texte 3-1-2 sticky=« w »)	`tkgrid(tklabel("Parent",text="(texte 3-1-2 sticky=\"w\")"), row=3, column=1,columnspan=2,sticky="w")`

Ma première fenêtre graphique

Objectif	Charger un fichier de données



Créer une fenêtre graphique

Fenêtre

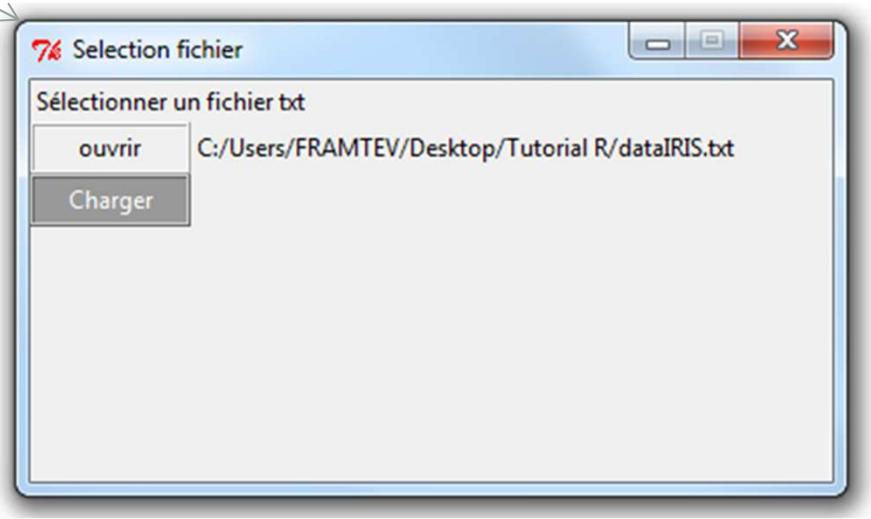
Commandes

```
wind1 <- tktoplevel()
tkwm.resizable(wind1, FALSE, FALSE)
tkwm.geometry(wind1, "400x200+0+0")
tkwm.title(wind1, "Selection fichier")
```

Dans le coin en haut à gauche (0,0)

Titre

Largeur fixe (400)



Hauteur fixe (200)

Créer une fenêtre graphique

Label

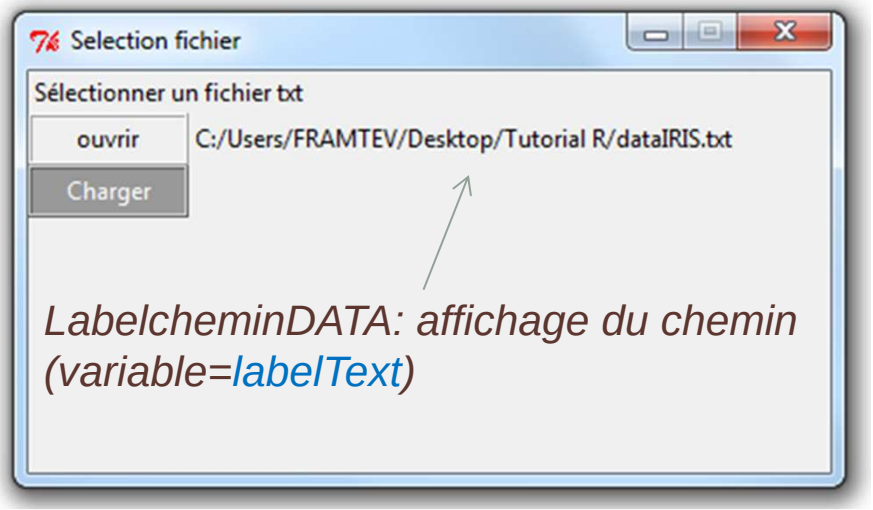
Commandes

```
labelwind1<- tklabel(wind1, text="Sélectionner un fichier
txt",fg="black")
```

```
labelText <- tclVar("")
```

```
labelcheminDATA <- tklabel(wind1, text=tclvalue(labelText),fg="black")
```

Labelwind1



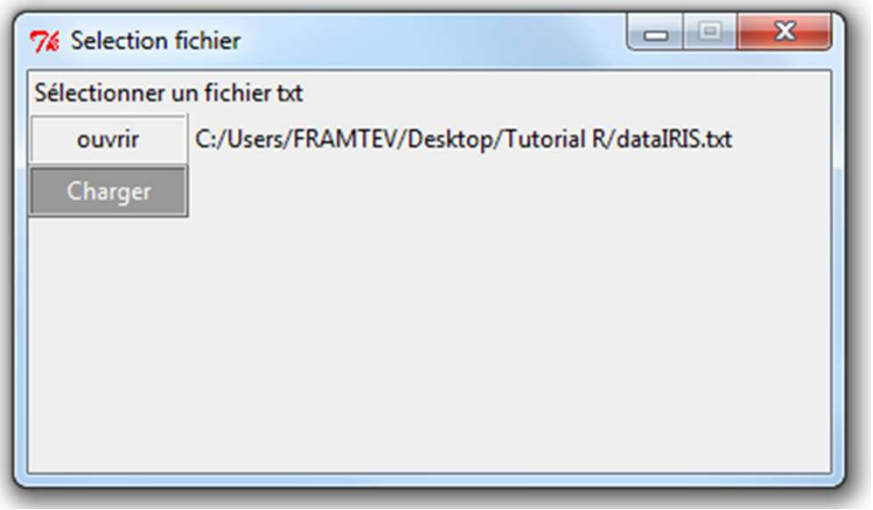
Créer une fenêtre graphique

Commandes

```
butopen<-tkbutton(wind1,relief="ridge",activebackground="orange",
       command=function()func.ouvrir(labelText),text="ouvrir",width=10)
```

```
q.but<-tkbutton(wind1,text="Charger",relief="ridge",foreground="white",
         background="grey60",command=function()exe(wind1),width=10)
```

Bouton ouvrir →
Bouton Charger →



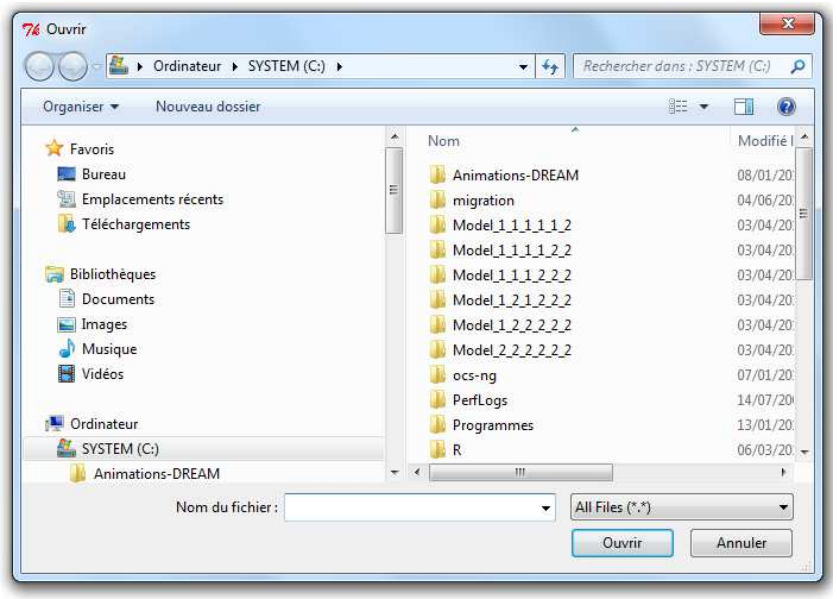
Attention il faut définir les fonctions **exe** & **func.ouvrir**

fonctions **exe** & **func.ouvrir**

Functions

```
func.ouvrir<-function(x){
        chemintemp<-tkgetOpenFile(initialdir="C:\\")
        tclvalue(x) <- tclvalue(chemintemp)
        tkconfigure(labelcheminDATA,textvariable=x)
        }
```

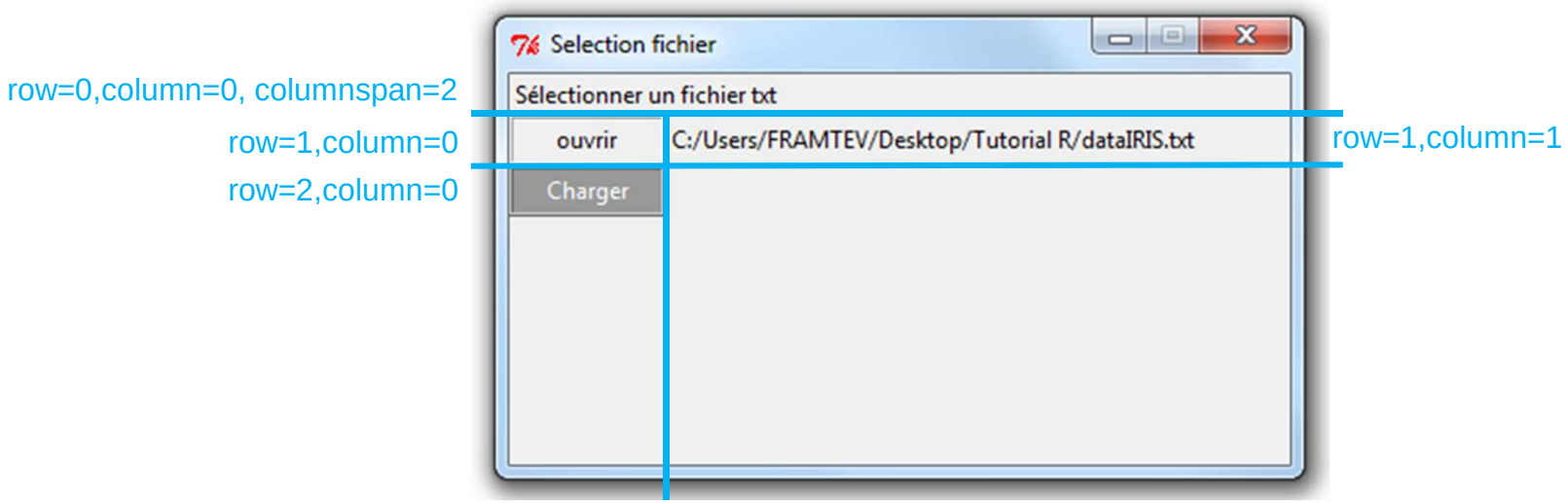
```
exe<-function(x){
        tkdestroy(x)
        }
```



Positionnement

Commandes

```
tkgrid(labelwind1,sticky="w",columnspan=2,row=0,column=0)
tkgrid(butopen, sticky="w",row=1,column=0)
tkgrid(labelcheminDATA,sticky="w",row=1,column=1)
tkgrid(q.but,sticky="w",row=2, column=0)
```



Sorties

Commandes

```
Mydata  <- read.table(file=tclvalue(labelText),header=TRUE)
Varname <- colnames(mydata)
```

Modifications d'options

- Modifier la couleur du texte dans label avec l'option `fg=""`
- Modifier la forme des boutons avec `relief=""`
- Utiliser l'option `activebackground=""` dans bouton

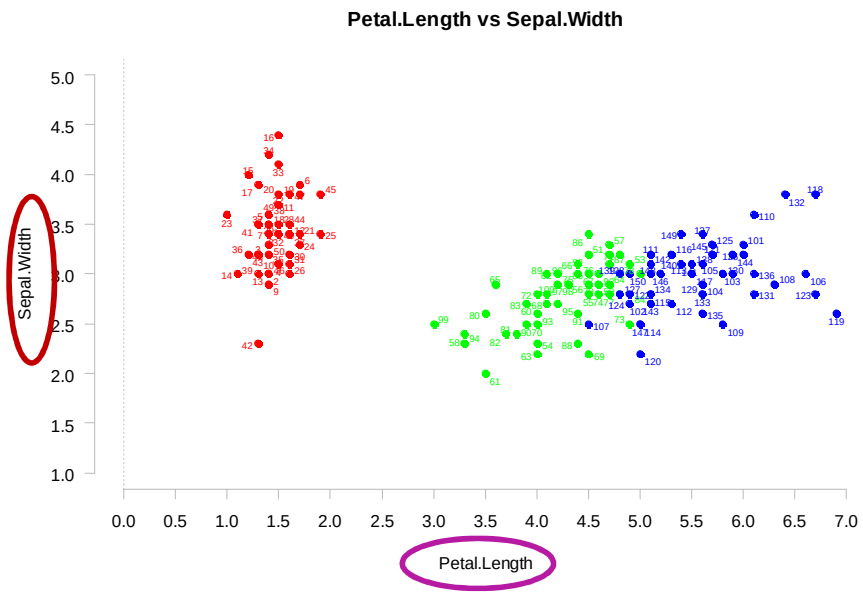
Objectif

Mydata

Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5,1	3,5	1,4	0,2	Setosa
4,9	3,0	1,4	0,2	Setosa
4,7	3,2	1,3	0,2	Setosa
4,6	3,1	1,5	0,2	Setosa
5,0	3,6	1,4	0,2	Setosa
5,4	3,9	1,7	0,4	Setosa

Facteurs

Species
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa

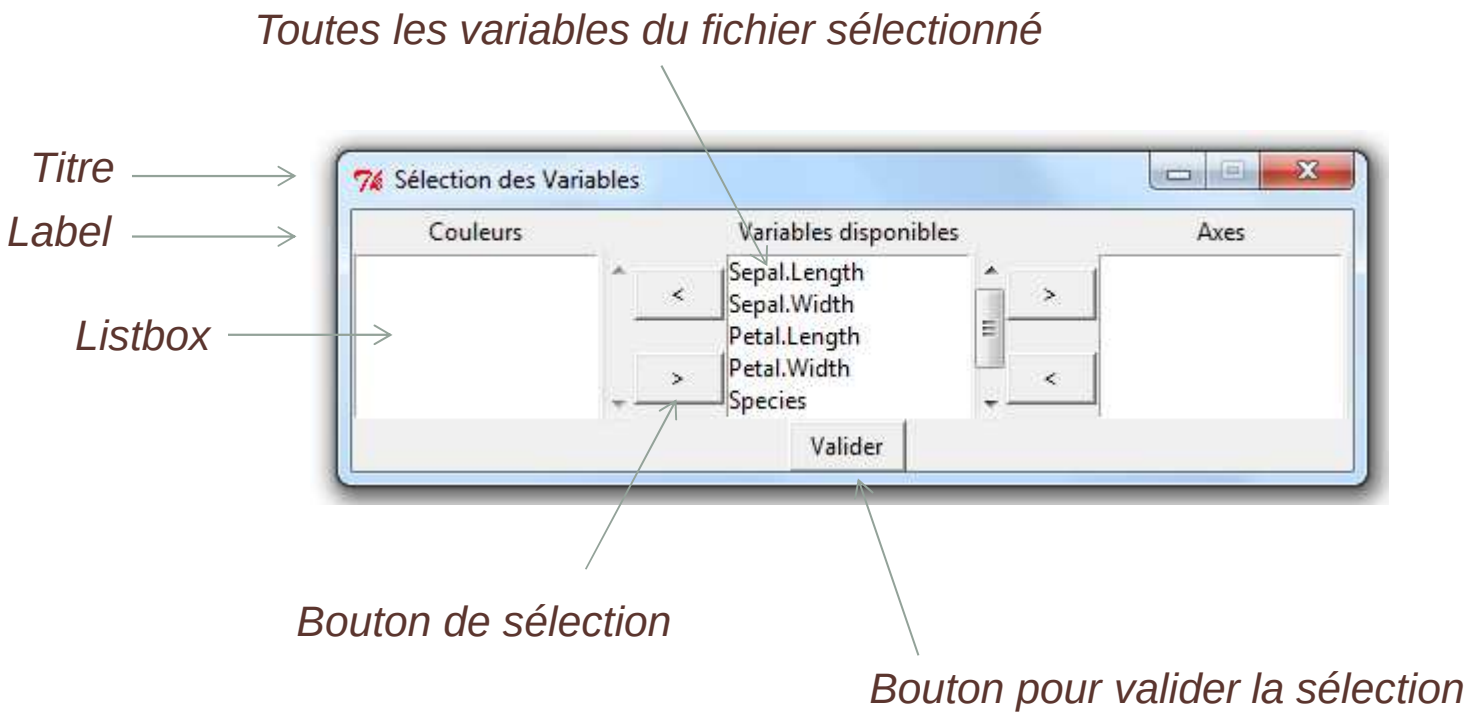


Variables numériques

Sepal Length	Sepal Width	Petal Length	Petal Width
5,1	3,5	1,4	0,2
4,9	3,0	1,4	0,2
4,7	3,2	1,3	0,2
4,6	3,1	1,5	0,2
5,0	3,6	1,4	0,2
5,4	3,9	1,7	0,4
4,6	3,4	1,4	0,3
5,0	3,4	1,5	0,2
4,4	2,9	1,4	0,2
4,9	3,1	1,5	0,1
5,4	3,7	1,5	0,2
4,8	3,4	1,6	0,2
4,8	3,0	1,4	0,1
4,3	3,0	1,1	0,1
4,8	4,0	1,2	0,2

Fenêtre « Sélection de variables »

Objectif	Sélectionner les variables (facteurs) pour attribuer des couleurs aux individus. Sélectionner les variables numériques à représenter sur un graphique.



Fenêtre « Sélection de variables »

Explications	Commandes

Créer la fenêtre, titre, label…	`wind2 <- tktoplevel()`
	`tkwm.resizable(wind2, FALSE, FALSE)`
	`tkwm.geometry(wind2, "500x130+0+0")`
	`tkwm.title(wind2, "Sélection des Variables")`

Création des vecteurs qui contiennent les noms des variables	`listec<-Varname #centre "toutes les variables par défaut"`
	`listeg<-NULL #gauche "codes couleur"`
	`listed<-NULL #droite "numériques"`

Création des labels des listebox	`labellistbg <- tklabel(wind2, text="Couleurs",fg="black")`
	`labellistbc <- tklabel(wind2, text="Variables disponibles")`
	`labellistbd <- tklabel(wind2, text="Axes",fg="black")`

Création des listebox	`listbg <- tklistbox(wind2,selectmode="extended",height=5)`
	`listbc <- tklistbox(wind2,selectmode="extended",height=5)`
	`listbd <- tklistbox(wind2,selectmode="extended",height=5)`

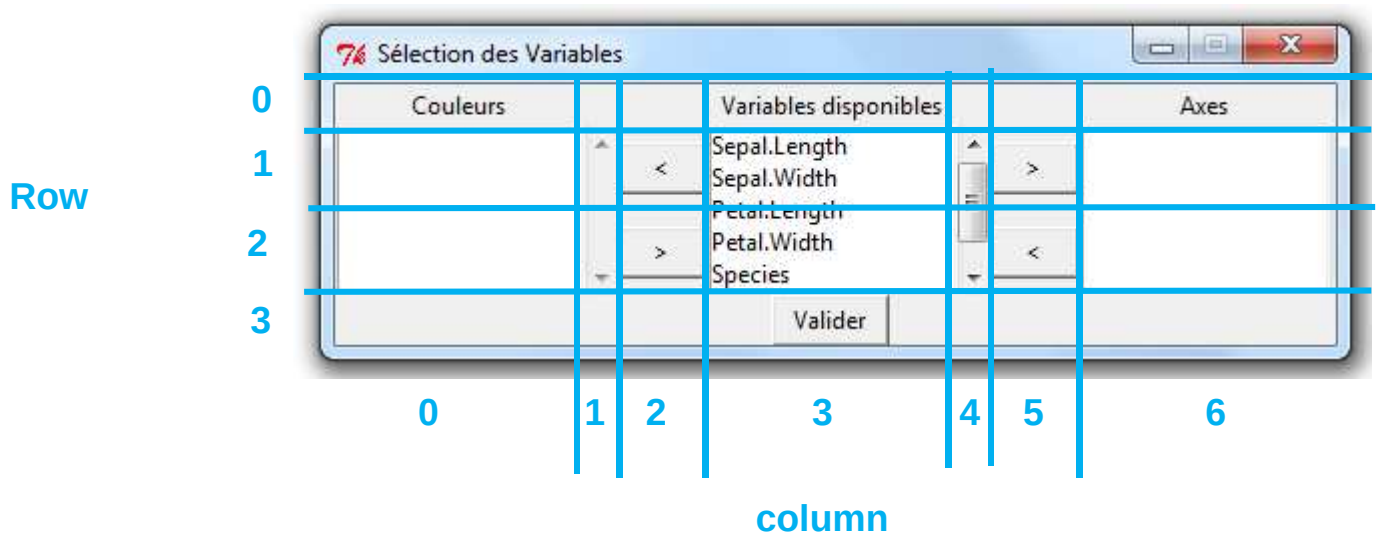
Remplir la listebox du centre	`for(i in 1:length(listec)){` `tkinsert(listbc,"end",listec[i])}`

Création des boutons	`bouton1<-tkbutton(wind2,text=">",width=5,command=function(){})`
	`bouton2<-tkbutton(wind2,text="<",width=5,command=function(){})`
	`bouton3<-tkbutton(wind2,text="<",width=5,command=function(){})`
	`bouton4<-tkbutton(wind2,text=">",width=5,command=function(){})`
	`bouton5<-tkbutton(wind2,text="Valider",width=7,`
	`command=function() {tkdestroy(wind2)})`

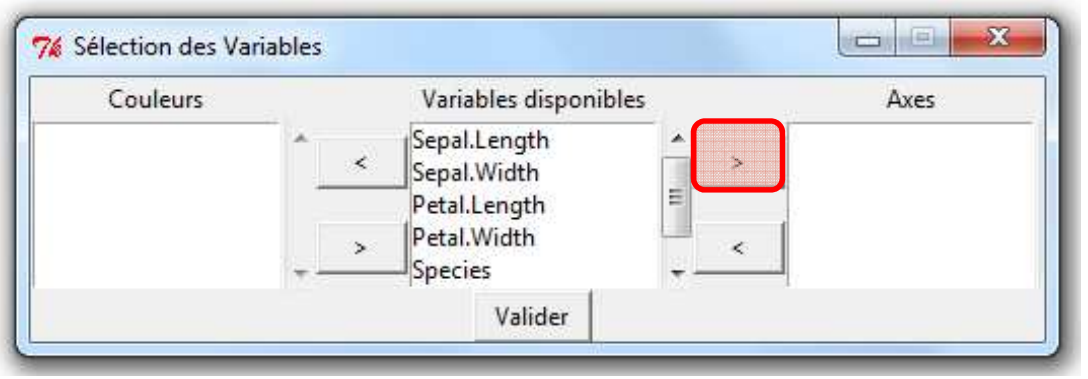
Positionnement

Commandes

```
tkgrid(labellistbg,row=0,column=0)
tkgrid(listbg,row=1,column=0,rowspan=2)
tkgrid(bouton3,row=1,column=2)
tkgrid(labellistbc,row=0,column=3)
tkgrid(listbc,row=1,column=3,rowspan=2)
tkgrid(bouton1,row=1,column=5)
tkgrid(bouton2,row=2,column=5)
tkgrid(labellistbd,row=0,column=6)
tkgrid(listbd,row=1,column=6,rowspan=2)
tkgrid(bouton5,row=3,column=3)
tkwait.window(wind2)
```



Fonction bouton 1

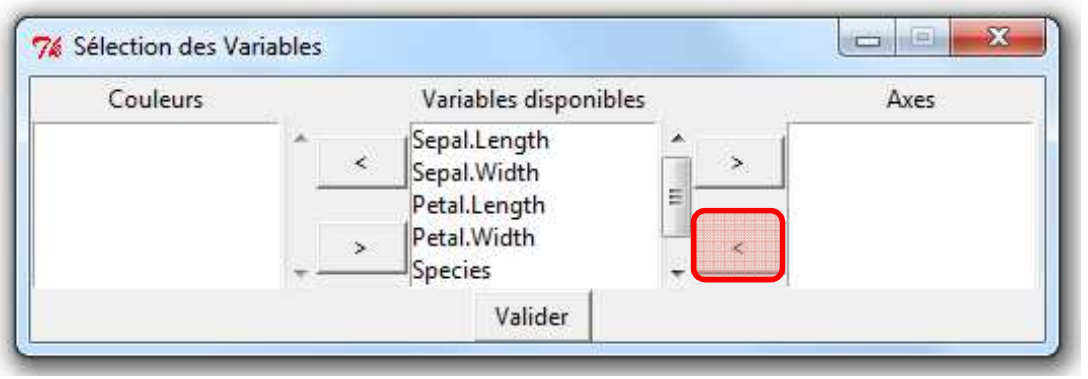


Functions

```
bouton1 <- tkbutton(wind2,text=">",width=5,command=function() {
## Vérifier que l'utilisateur à selectionner des variables dans la fenêtre du centre
  if (tclvalue(tkcurselection(listbc))!="") {
## Récupérer les numéros des variables sélectionnées
    selection<-as.numeric(strsplit(tclvalue(tkcurselection(listbc)),"")[[1]])+1

    ## Ajouter/retirer dans la lbox de droite/centre les noms de la sélection par itération
       du plus grand au plus petit
    selectiono<-sort(selection,decreasing =TRUE)
    for(i in selectiono){
      tkdelete(listbc,i-1)
      tkinsert(listbd,"end",listec[i]) #selection[aaa]
      }
     ## Rajouter les noms dans la liste de droite
     listed<<-c(listed,listec[selectiono])
     ## Retirer les noms dans la liste du centre ## ATTENTION A L'ORDRE
     listec<<-listec[-selectiono]
   }else{tkmessageBox(message="Aucune variable sélectionnée !",type="ok",icon="error")}
 })
```

Fonction bouton 2



Functions

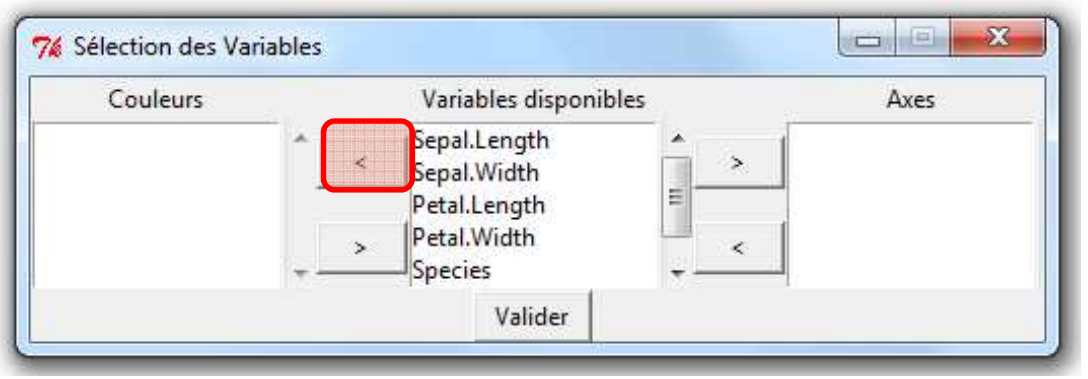
```
bouton2 <- tkbutton(wind2,text="<",width=5,command=function() {
   if (tclvalue(tkcurselection(listbd))!="") {
     selection <- as.numeric(strsplit(tclvalue(tkcurselection(listbd))," ")[[1]])+1

     selectiono<-sort(selection,decreasing =TRUE)

     for(i in selectiono){
      tkdelete(listbd,i-1)
      tkinsert(listbc,"end",listed[i])
      }
     listec<<-c(listec,listed[selectiono])
     listed<<-listed[-selectiono]

   } else {
     tkmessageBox(message="Aucune variable sélectionnée !",type="ok",icon="error")
   }
 })
```

Fonction bouton 3



Functions

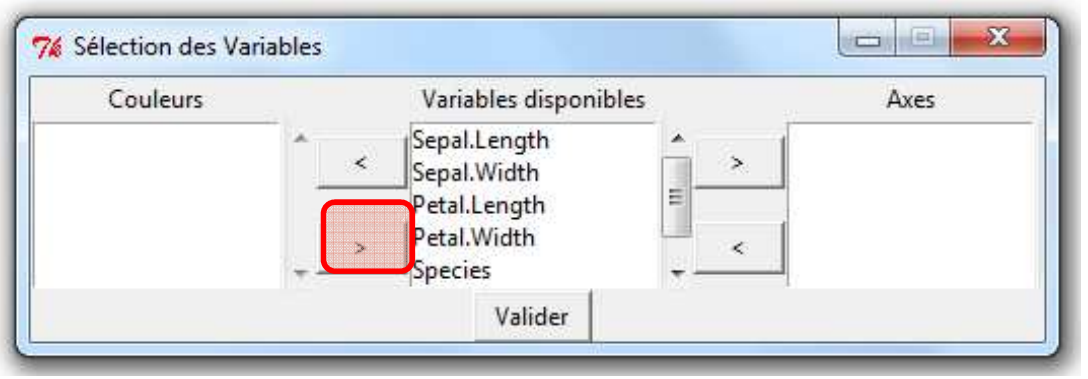
```
bouton3 <- tkbutton(wind2,text="<",width=5,command=function() {
   if (tclvalue(tkcurselection(listbc))!="") {
     selection <- as.numeric(strsplit(tclvalue(tkcurselection(listbc))," ")[[1]])+1

     selectiono<-sort(selection,decreasing =TRUE)

     for(i in selectiono){
      tkdelete(listbc,i-1)
      tkinsert(listbg,"end",listec[i])
      }
     listeg<<-c(listeg,listec[selectiono])
     listec<<-listec[-selectiono]

   } else {
     tkmessageBox(message="Aucune variable sélectionnée !",type="ok",icon="error")
   }
 })
```

Fonction bouton 4



Functions

```
bouton4 <- tkbutton(wind2,text=">",width=5,command=function() {
   if (tclvalue(tkcurselection(listbg))!="") {
     selection <- as.numeric(strsplit(tclvalue(tkcurselection(listbg))," ")[[1]])+1

     selectiono<-sort(selection,decreasing =TRUE)

     for(i in selectiono){
      tkdelete(listbg,i-1)
      tkinsert(listbc,"end",listeg[i])
      }
     listec<<-c(listec,listeg[selectiono])
     listeg<<-listeg[-selectiono]

   } else {
     tkmessageBox(message="Aucune variable sélectionnée !",type="ok",icon="error")
   }
 })
```

Rajouter les scrollbars

Explications	Commandes

Création des scrollbars	`scrbg <- tkscrollbar(wind2, repeatinterval=5,`
	`command=function(...)tkyview(listbg,...))`
	`scrbc <- tkscrollbar(wind2, repeatinterval=5,`
	`command=function(...)tkyview(listbc,...))`
	`scrbd <- tkscrollbar(wind2, repeatinterval=5,`
	`command=function(...)tkyview(listbd,...))`

Création des listebox	`listbg <- tklistbox(wind2,selectmode="extended",height=5,`
	`yscrollcommand=function(...)tkset(scrbg,...))`
	`listbc <- tklistbox(wind2,selectmode="extended",height=5,`
	`yscrollcommand=function(...)tkset(scrbc,...))`
	`listbd <- tklistbox(wind2,selectmode="extended",height=5,`
	`yscrollcommand=function(...)tkset(scrbd,...))`

Positionnement

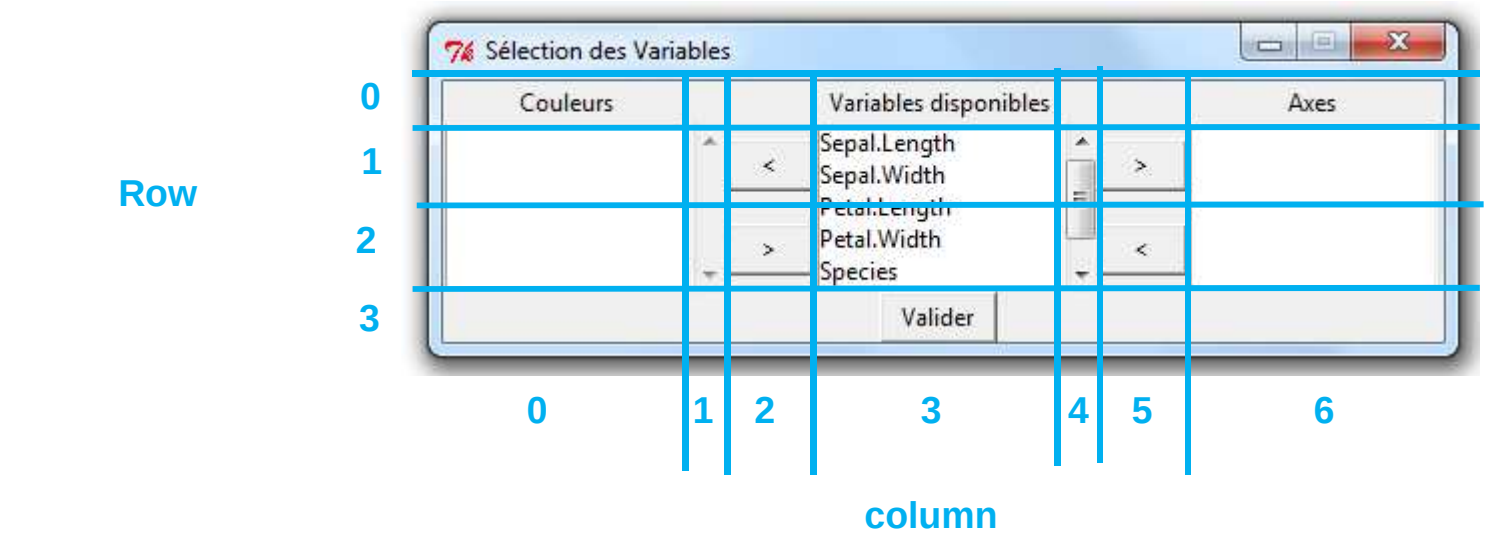
```
tkgrid(scrbc,row=1,column=4,rowspan=2,sticky="ns")
```

```
tkgrid(scrbd,row=1,column=7,rowspan=2,sticky="ns")
```

```
tkgrid(scrbg,row=1,column=1,rowspan=2,sticky="ns")
```

Commandes

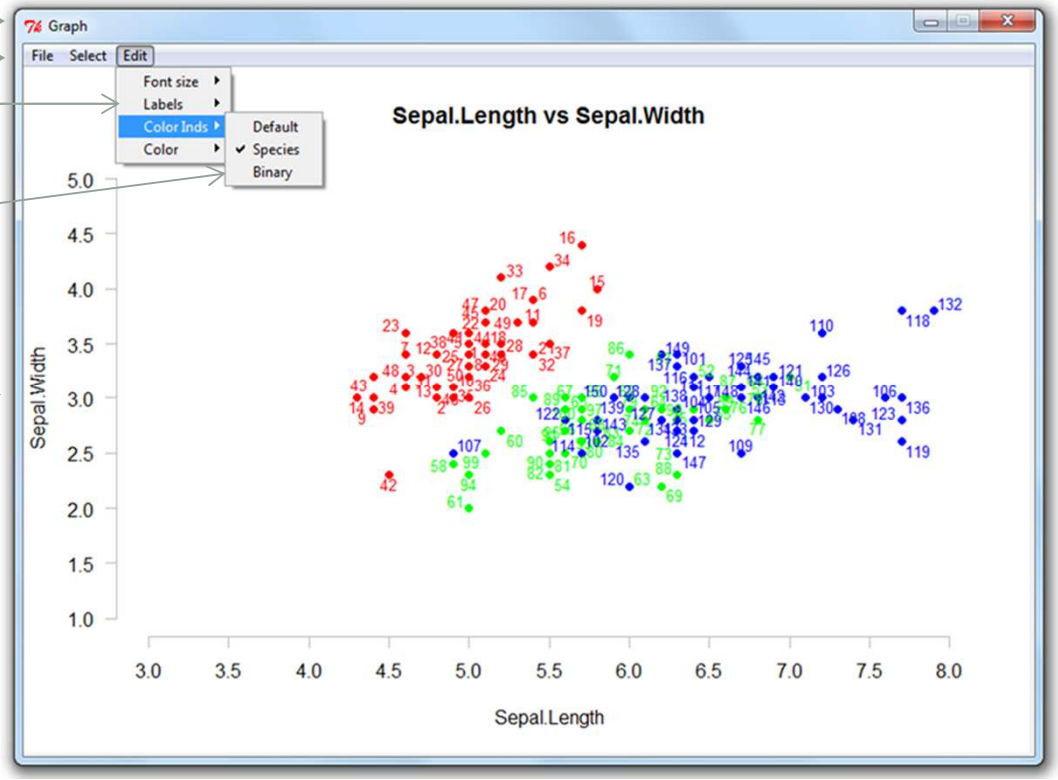
```
tkgrid(labellistbg,row=0,column=0)
tkgrid(listbg,row=1,column=0,rowspan=2)
tkgrid(bouton3,row=1,column=2)
tkgrid(labellistbc,row=0,column=3)
tkgrid(listbc,row=1,column=3,rowspan=2)
tkgrid(bouton1,row=1,column=5)
tkgrid(bouton2,row=2,column=5)
tkgrid(labellistbd,row=0,column=6)
tkgrid(listbd,row=1,column=6,rowspan=2)
tkgrid(bouton5,row=3,column=3)
tkwait.window(wind2)
```



Fenêtre « Graph »

Objectif	Tracer un graphique interactif. Options: Modification de la taille de la police. Supprimer/afficher les labels. Changer la variable couleur. Modifier la couleur des attributs de la variable couleur. Changer les variables axes.

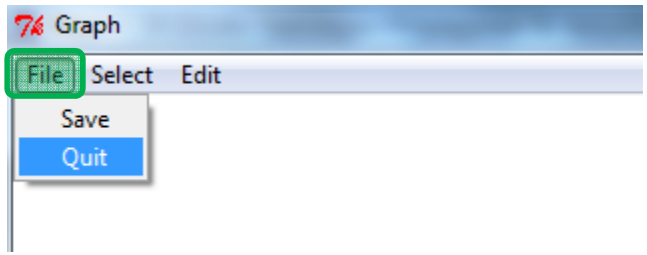
Titre
Menus
Sous Menus
Variables numériques (listbox axes)
Variables numériques (listbox axes)

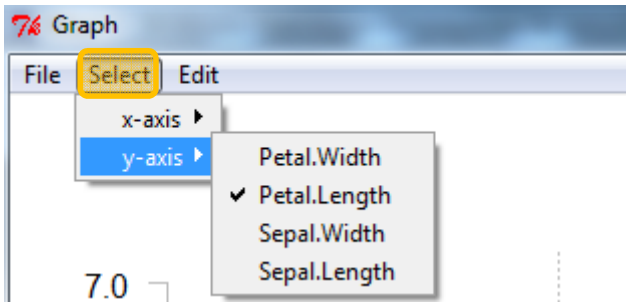


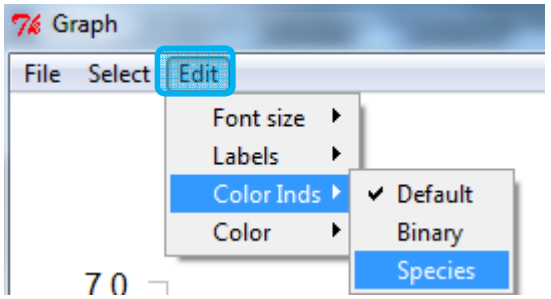
Fenêtre « Graph »

Interface graphique	`wind3 <- tktoplevel()`
	`tkwm.title(wind3, "Graph")`
	`tkwm.resizable(wind3, FALSE, FALSE)`
	`tkwm.geometry(wind3, "850x600+0+0")`

Création des menus	**`fileMenu`** `<- tkmenu(topMenu,tearoff=FALSE)` **`GraphMenu`** `<- tkmenu(topMenu,tearoff=FALSE)` **`xaxis`** `<- tkmenu(topMenu,tearoff=FALSE)` **`yaxis`** `<- tkmenu(topMenu,tearoff=FALSE)` **`FontcolMenu`** `<- tkmenu(topMenu,tearoff=FALSE)` **`fontsize`** `<- tkmenu(topMenu,tearoff=FALSE)` **`tplabeloption`** `<- tkmenu(topMenu,tearoff=FALSE)` **`selectvarcci`** `<- tkmenu(topMenu,tearoff=FALSE)`

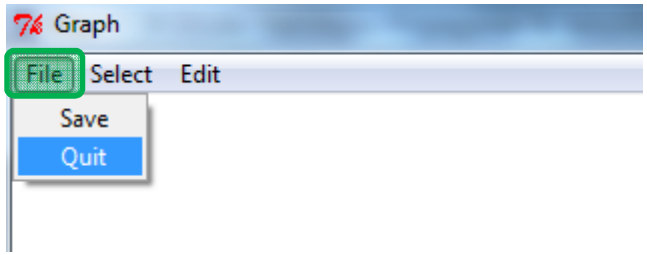






Menu « File »

Sous-menu de "File"	`tkadd(fileMenu,"command",label="Save",command=function()f.save())` `tkadd(fileMenu,"command",label="Quit",command=function() quit())`



Fonctions	`quit<-function(){tkdestroy(wind3)}`
	`}`
	`f.save<-function(){}`

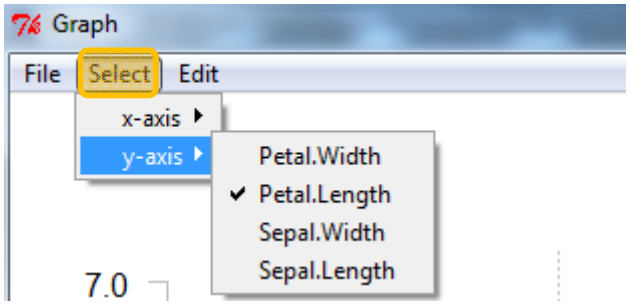
Menu « Select »

Initialisation	`Xtot<-mydataNUM`
	`ChoixCOMPX<-colnames(Xtot)[1] #Axe des abscisses`
	`ChoixCOMPY<-colnames(Xtot)[2] #Axe des ordonnées`

Radiobutton pour modifier la variable en abscisse	`rbxvalue<-tclVar(ChoixCOMPX)`
	`rbxvalueRange<-colnames(Xtot)`
	`for(rbxjj in rbxvalueRange){`
	`  tkadd(xaxis, "radiobutton",label=rbxjj,variable=rbxvalue,`
	`  value=rbxjj,command=function()f.options(A définir))}`

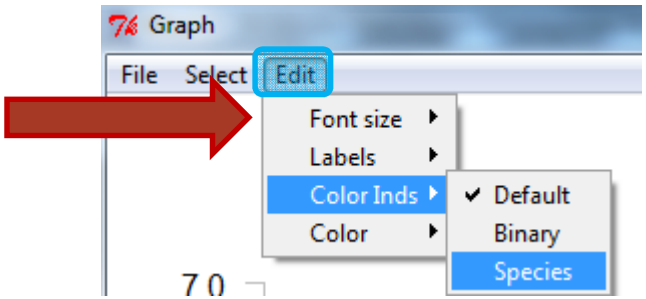
Radiobutton pour modifier la variable en ordonnée	`rbyvalue<-tclVar(ChoixCOMPY)`
	`rbyvalueRange<-colnames(Xtot)`
	`for(rbyjj in rbyvalueRange){`
	`  tkadd(yaxis,"radiobutton",label=rbyjj,variable=rbyvalue,`
	`  value=rbyjj,command=function()f.options(A définir))}`

Sous-menu de "select"	`tkadd(GraphMenu,"cascade",label="x-axis",menu=xaxis)` `tkadd(GraphMenu,"cascade",label="y-axis",menu=yaxis)`



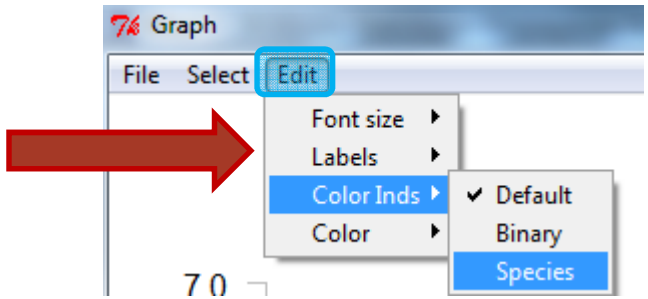
Menu « Edit »

Radiobutton pour changer la taille des labels	`mycex<-.8`
	
	`rbfontsize<-tclVar(mycex)`
	`rbfontsizeRange<-as.character(seq(.5,1.5,.1))`
	
	`for(rbfontsizejj in rbfontsizeRange){`
	`  tkadd(fontsize,"radiobutton",label=rbfontsizejj,`
	`  variable=rbfontsize,value=rbfontsizejj,`
	`  command=function()f.options(A définir))}`
	
	`tkadd(FontcolMenu,"cascade",label="Font size",menu=fontsize)`



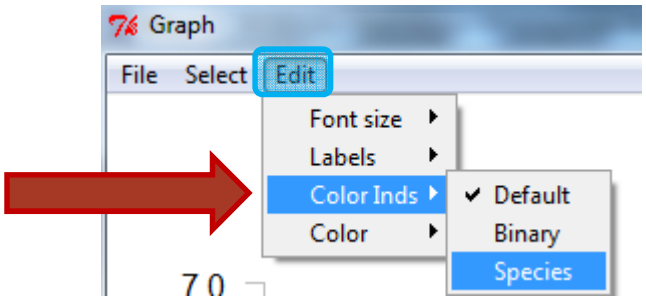
Menu « Edit »

Radiobutton pour afficher les labels	`labeloption<-"Visible"`
	
	`rblabeloption<-tclVar(labeloption)`
	`rblabeloptionRange<-as.character(c("Visible","Invisible"))`
	
	`for(rblabeloptionjj in rblabeloptionRange){`
	`  tkadd(tplabeloption,"radiobutton",label=rblabeloptionjj,`
	`  variable=rblabeloption,value=rblabeloptionjj,`
	`  command=function()f.options(A définir) )}`
	
	`tkadd(FontcolMenu,"cascade",label="Labels",menu=tplabeloption)`



Menu « Edit »

Radiobutton pour modifier la variable couleur	`ChoixVarcci<-"Default"`
	
	`rbVarcci<-tclVar(ChoixVarcci)`
	`rbVarcciRange<-as.character(dimnames(varind)[[2]])`
	`for(rbVarccijj in rbVarcciRange){`
	`  tkadd(selectvarcci,"radiobutton",label=rbVarccijj,`
	`  variable=rbVarcci,value=rbVarccijj,command=function()`
	`  f.options (A définir))}`
	
	`tkadd(FontcolMenu,"cascade",label="Color Inds",menu=selectvarcci)`

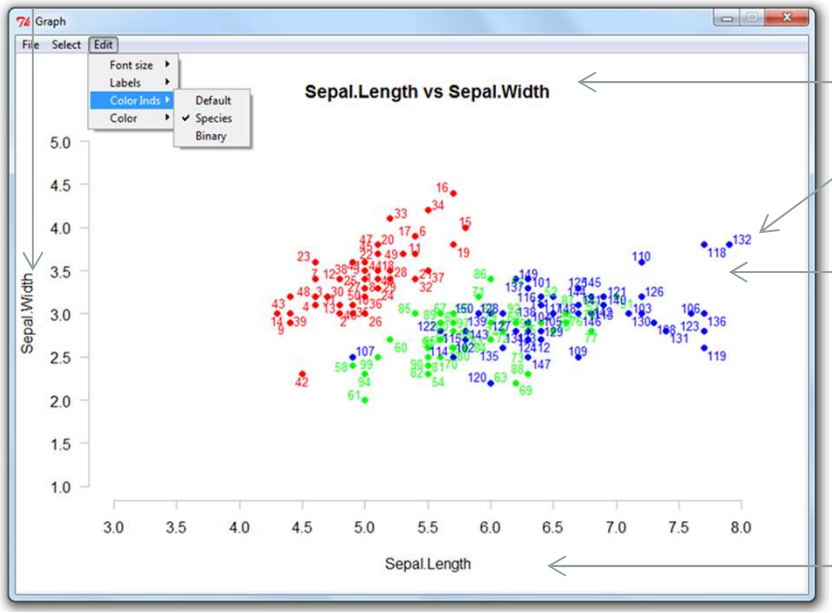


Affichage du graphique: tkrplot

tkrplot pour graphique interactif	`img <-tkrplot(wind3, function(){`
	`  par(bg="white")`
	`  Graphind(Xtot,comp=c(ChoixCOMPX,ChoixCOMPY),`
	`  titre=paste(ChoixCOMPX,"vs",ChoixCOMPY),`
	`  mycex=mycex,labeloption=labeloption,macol=myrgbplot)`
	`  },hscale=2.2,vscale=1.5)`

Positionnement	`tkgrid(img,sticky="w")` `tkwait.window(wind3)`

ChoixCOMPY



Titre

labeloption, mycex

macol

ChoixCOMPX

Reste à définir les fonctions **graphind() & f.options()** et **myrgbplot**

Couleur des modalités: myrgbtot

Couleurs des modalités de la variable « couleur »	`myrgbtot<-colorRampPalette(c("red", "orange","green","cyan", "blue"))(nlevels(varind[,ChoixVarcci]))`
	
	`myrgbtot<-matrix(c(levels(varind[,ChoixVarcci]),myrgbtot),ncol=2, nrow=nlevels(varind[,ChoixVarcci]))`
	
	`dimnames(myrgbtot)[[2]]<-c("label","codecol")`
	`myrgbplot<-rep(myrgbtot[,"codecol"],nrow(varind))`

Variable = Species

label	codecol
setosa	#FF0000
versicolor	#00FF00
virginica	#0000FF

Variable = Defaut

label	codecol
All	#FF0000

Fonction Graphind()

```
Graphind<-function(X,comp=c(1,2),titre="",mycex=0.8,myoffset=0.5,macol=1,
           labeloption="Visible"){
nr<-nrow(X)
if(is.null(dimnames(X)[[1]])){dimnames(X)<-list(1:nr,paste("V",1:nc,sep=""))}

  ## Ajuste l'échelle des axes des abscisses et des ordonnées.
xmin<-trunc(min(X[,comp[1]]))-1
xmax<-trunc(max((X[,comp[1]])))+1
ymin<-trunc(min(X[,comp[2]]))-1
ymax<-trunc(max((X[,comp[2]])))+1
scaley<-pretty(c(ymin,ymax),n=10)
scalex<-pretty(c(xmin,xmax),n=10)

  ## Graphique
plot(X[,comp[1]],X[,comp[2]],pch=19,col=macol,xlab=comp[1],ylab=comp[2],main=titre,axes=FALSE,xlim=c(mi
n(scalex),max(scalex)),ylim=c(min(scaley),max(scaley)))
axis(1,at=scalex,col="gray",cex=0.8)
axis(2,at=scaley,las=2,col="gray",cex=0.8)
abline(v=0,h=0,col="gray",lty=3)

 ## Option affichage des labels
if(labeloption=="Visible"){
    pointLabel(X[,comp[1]],X[,comp[2]], dimnames(X)[[1]], offset = myoffset,
               cex =mycex,col=macol)
    }
  }
```

Fonction f.options()

```
f.options<-function(svarcci,xaxis,yaxis,mycex=.8,labeloption="Visible",readtext=FALSE){
   myvarcci<-as.factor(varind[,svarcci])

   myrgbtot<-colorRampPalette(c("red", "orange","green","cyan","blue"))(nlevels(myvarcci))
   myrgbtot<-matrix(c(levels(myvarcci),myrgbtot),ncol=2,nrow=nlevels(myvarcci))
   dimnames(myrgbtot)[[2]]<-c("label","codecol")

   aaa<-0
   for(kkk in levels(myvarcci)){
        aaa<-aaa+1
        myrgbplot[myvarcci==kkk]<-myrgbtot[aaa,"codecol"]
        myrgb<-myrgbtot[aaa,"codecol"]
        }

   ## Mettre à jour les options
   ChoixCOMPX<<-xaxis
   ChoixCOMPY<<-yaxis
   mycex<<-as.numeric(mycex)
   labeloption<<-labeloption
   tclvalue(rbfontsize)<<-mycex
   tclvalue(rbVarcci)<<-svarcci
   myrgbplot<<-myrgbplot
   svarcci<<-svarcci

   tkrreplot(img)
   }
```