Fonctions graphiques de R

Résumé

Un rapide aperçu des très riches fonctions graphiques de R pour gérer fenêtres et fichiers, tracer des camemberts, diagrammes en colonnes, mosaïc plot, histogrammes, diagrammes boîtes, images, contours, vues en 3D, ajouter des composantes graphiques.

Organisation des tutoriels R.

- *Démarrer rapidement avec R*
- *Initiation à R*
- *Fonctions graphiques de R*
- *Programmation en R*
- *MapReduce pour le statisticien*

Les aspect statistiques sont développés dans les différents scénarios de Wikistat.

1 Introduction

Plus un problème et les données afférentes sont complexes, plus est important la pertinence des graphes qui permettront d'en appréhender la structure et visualiser l'information utile. Aussi, quelque soit le logiciel utilisé, commercial comme SAS ou libre comme R, les versions basées sur une interface *amicale* : SAS/Insight ou Rcmdr, sont évidemment séduisantes, car faciles à utiliser par des menus, mais drastiquement limitées par les choix des options nécessairement très contraints parmi des possibilités considérables.

`demo(graphics)` donnera un aperçu de la palette des possibilités tandis qu'une visite à l'un de ces sites : gallerie 1, gallerie 2, gallerie 3, ouvre de vastes perspectives sur les possibilités offertes.

C'est la principale raison, qui motive de cheminer par l'apprentissage du lange de commande R plutôt que par celui d'une interface graphique aussi amicale soit-elle. Bien entendu ce tutoriel ne présente que quelques facettes des graphiques les plus utilisés tandis que beaucoup de ressources et sites, comme ceux ci-dessus, rendent accessibles des exemples complexes et ciblés sur des besoins particuliers.

Attention cependant aux excès d'esthétisme, un *beau* graphique n'est pas nécessairement un graphique *pertinent* ; éviter par exemple les abus d'effets de perspective sans signification statistique.

2 Commandes de base

2.1 Aide en ligne

Comme pour tout utilisation avancée d'une fonction de R, le recours à l'aide en ligne est incontournable. Apprendre à y naviguer et comprendre sa logique font partie de l'apprentissage.

`help(plot)` offre un premier accès alors que

`help(par)` embrouille vite les cartes.

2.2 Gestion de la fenêtre graphique

Toute commande graphique ouvre une fenêtre adaptée mais

`x11()` ouvre une autre fenêtre pour éviter d'écraser le graphe précédent,

`dev.off()` ferme correctement la dernière fenêtre,

`split.screen(c(1,2))` partage la fenêtre en 2 de même que l'option `mfrow` de la commande `par`.

2.3 Exportation de fichiers graphiques

Voici quelques possibilités pour intégrer un graphique dans un traitement de texte ou générer un fichier contenant le graphique avant intégration ou archivage.

- Sous Windows, clic droit dans la fenêtre pour copier le graphique dans le presse papier avant de le coller dans un texte,
- L'obtention du format standard `jpeg` nécessite d'enchaîner les commandes :

```
jpeg("fichier.jpeg") # ou bmp(), png(), pdf()
# le graphique n'apparaît pas à l'écran :
plot(1:100); text(20,80,"abcdef")
dev.off() # fermeture indispensable du fichier
```

qui est alors créé dans le répertoire courant.
- Utiliser le menu `Fichier`, rubrique `Sauver sous` de la fenêtre graphique.

3 Description élémentaire

3.1 Variables qualitatives

Fonctions `pie, barplot, mosaicplot`.

```
vec=c(12,10,7,13,26,16,4,12)
pie(vec);pie(vec,clockwise=T)
names(vec)=LETTERS[1:8]
# de très nombreuses options sont disponibles
pie(vec)
barplot(vec)
# pour découper la fenêtre en 1 ligne 2 colonnes
par(mfrow=c(1,2))
pie(vec)
barplot(vec)
par(mfrow=c(1,1))
mosaicplot(Titanic, color = T)
colors()
```

Questions

1. Quelle est la différence entre `par(mfrow=c(2,2))` et `par(mfcol=c(2,2))` ?
2. Consulter l'aide en ligne pour tester certains arguments optionnels des fonctions `pie()` et `barplot()`.
3. Commenter les commandes suivantes (extraites de l'aide en ligne de `pie()`):

```
n=200
pie(rep(1,n), labels="", col=rainbow(n), border=NA)
```

Réponses

1. Les 2 commandes découpent la fenêtre graphique en 4 cellules. La différence entre les 2 commandes est illustrée par le tableau ci-dessous (`par(mfrow=c(2,2))` - `par(mfcol=c(2,2))`) :

1 - 1	2 - 3
3 - 2	4 - 4

   Avec `par(mfrow=c(2,2))`, les 4 graphiques à venir sont intégrés ligne par ligne ; avec `par(mfcol=c(2,2))`, ils le sont colonne par colonne.
2. Par exemple, pour modifier les couleurs du camembert :

```
pie(vect,col=rgb(0,0,seq(0,1,l=8)))
pie(vect,col=rainbow(8))
...
```

3. Commentaire de la ligne de commande :
   - `n=200` # La valeur 200 est affectée à `n`.
   - `rep(1,n)` # Crée le vecteur de taille `n` ne contenant que des 1.
   - `pie(rep(1,n)` # Produit un camembert pour `n` variables d'effectifs tous égaux à 1. Les secteurs sont donc de même angle.
   - `rainbow(n)` # Renvoie un vecteur de longueur `n` contenant un dégradé des couleurs de l'arc-en-ciel.
   - `border=NA` # Évite de tracer les bords des secteurs.
   - `labels=""` # Aucune étiquette n'identifie les secteurs.

3.2 Variables quantitatives

Commandes `boxplot, hist, plot, pairs`.

```
x=rnorm(50)
boxplot(x)
hist(x)
stripchart(x)
```

Question

1. Représenter dans la même fenêtre graphique, le "stripchart", le diagramme-boîte horizontal et l'histogramme correspondant l'un sous l'autre.

Réponse

```
par(mfrow=c(3,1))
stripchart(x)
boxplot(x,horizontal=T)
hist(x)
```

Croisement de variables.

```
data(iris)
pairs(iris[,1:4])
par(mfrow=c(2,2))
plot(iris[,1],iris[,2],xlab="Sepal Length",
   ylab="Sepal Width",main="Iris data",
   col="red",type="l")
points(iris[,1],iris[,2],col="green",pch=21)
boxplot(iris[,1:4])
hist(iris[,1],xlab="Sepal Length",
    main="Histogramme")
```

Ajouter des éléments graphiques.

```
x=seq(-10,10,l=50)
plot(x,sin(x))
plot(x,sin(x),type="l")
abline(v=0,col="blue",lwd=5,lty=3)
abline(h=sin(0.7),col=3)
text(-5,-0.5,"texte",font=3)
```

Gestion des libellés.

```
par(mfrow=c(1,2))
plot(x,sin(x),type="l",col=1,main="sinus")
plot(x,cos(x),type="b",col=3,xlab="Abscisses")
```

3.3 Vers la 3D

Commandes `image, persp, contour`.

```
M=matrix(1:100,nc=10)
# images, nappes et contours
image(M)
x = seq(-10, 10, length= 30);y=x
f = function(x,y){r=sqrt(x^2+y^2); 10 * sin(r)/r}
z = outer(x, y, f)
z[is.na(z)] = 1
persp(x,y,z)
persp(x, y, z, theta=30, phi=30,expand = 0.5,
   col="lightblue")
image(x,y,z)
contour(x,y,z)
filled.contour(x,y,z)
image(x,y,z)
contour(x,y,z,add=T)
```

Pour mieux comprendre la fonction `outer()`, tester :

```
x=y=1:5
z=outer(x,y,"+");z
```