TP 3 : R et visualisation

Ressources :
http://www.ceb-institute.org/bbs/wp-content/uploads/2011/09/handout_ggplot2.pdf
http://www.ling.upenn.edu/~joseff/avml2012/#Section_1
http://www.cookbook-r.com/Graphs/Bar_and_line_graphs_%28ggplot2%29/

Avant d'arriver en TP
Installation de RStudio Desktop : Allez sur le site http://www.rstudio.com/products/rstudio/download/ et téléchargez le logiciel. Puis, dans une publication scientifique (un article, un livre) repérez une figure que vous trouvez excellente (claire, pertinente, esthétique) et une figure que vous trouvez pas bien (pas claire, moche, illisible) et argumentez.

Premiers pas avec Rstudio : File-> New File->Script
Enregistrer le nouveau script directement en cliquant sur la petite icône disquette. N'oubliez pas de sauvegarder votre script régulièrement *(Ctr+S)*.
Vous devriez voir votre écran divisé en 4 parties principales: *Script, Console, Environment/History et Files/Plots/Packages/Help/Viewer.*
Dans la partie script vous écrivez les commandes et pour les faire marcher vous allez dans *Code->Run Lines* ou vous utilisez *Ctr+Enter.*
Essayez

```
print('Hello world') [Ctr+Enter]
```

Dans la partie console vous allez voir:

```
> print('Hello world')
[1] "Hello world"
```

Si vous cliquez sur *History* vous devriez voir: print('Hello world') . L'historique sauvegarde les commandes effectués dans l'ordre.
Pour savoir plus sur la fonction 'print', tapez:

```
?print
```

directement dans la console, à votre droite l'onglet 'Help' va afficher directement les informations sur 'print'. Dans le R, toutes les fonctions sont bien documentées. A la fin des informations sur la fonction, vous allez toujours trouver lde exemples d'utilisation.

Manipulation de données

Importez les données

Enregistrez le fichier *ozone.txt* (http://www.agrocampus-ouest.fr/math/livreR/ozone.txt) sur votre ordinateur. Ensuite importez dans R pour se faciliter la tâche, décidez dans quel répertoire vous voulez avoir vos fichiers R de cette session (wd : working directory)

```
setwd("/path")
read.table("ozone.txt", sep=" ", header=TRUE)
```

Attention cependant, vous devriez éviter d'afficher les données dans la console. Dans le cas de grands jeux de données, cela peut ralentir énormément le système. Donc le mieux c'est:

```
mytable=read.table("ozone.txt", sep=" ", header=TRUE)
```

Dès que vous créez une variable elle apparaît dans l'onglet environnement et vous pouvez voir ses caractéristiques. *La bonne pratique:* Après avoir importé votre jeu de données, vérifiez toujours s'il est bien importé. Une manière simple, c'est de faire :

```
summary(mytable)
```

Pour savoir les dimensions du tableau :

```
dim(mytable)
```

A quoi correspond le premier nombre? Le deuxième nombre?

Organisation d'information

```
head(mytable)
```

Que décrit ce tableau, comment? Est-ce intuitif ? Puis,

1. Afficher juste la première colonne du tableau 'mytable'
2. Afficher juste la première ligne du tableau 'mytable'
3. Afficher les noms de colonnes du tableau 'mytable'
4. Créez un vecteur 'Group' de la même longueur qu'une colonne du tableau contenant le nombre 1 pour la moitié de lignes et le nombre 2 pour l'autre moitié *(indice function 'rep')*
5. L'ajoutez au tableau comme une colonne (*indice function 'cbind')*
6. Supprimez la colonne que vous venez d'ajouter

Visualisation des données R standard

Commencer par la commande :

```
demo(graphics)
```

et observez l'onglet 'Plots' .Dans la console il s'affiche les commandes utilisées pour obtenir les graphes.

Mais commençons par nos donnés: Le jeu de donnés contient les variables climatiques et une variable de pollution à l'ozone mesurées durant l'été 2001 à Rennes. Les variables considérées sont :

- max03- maximum de l'ozone journalier
- T12 - température à midi
- vent - direction du vent
- pluie
- Vx12 - projection du vecteur vitesse du vent sur l'axe Est-Ouest

Regardez encore une fois : `summary(mytable)` . Quelles variables sont quantitatives? Qualitatives?

Sélectionnez seulement les colonnes mentionnées et enregistrez dans une variable '*ozon*'

```
ozone=mytable[,c('T12','maxO3','vent','pluie','Vx12')]
summary(ozone)
```

Pour représenter deux colonnes comme un nuage de points : Par exemple, observer le taux d'O3 maximale en fonction de la température à midi

```
pot(x,y)
plot(ozone[,'T12'],ozone[,'maxO3']) ou bien  plot(maxO3~T12, data=ozone)
```

Visualisez le taux maximal d'O3 (maxO3) en fonction de la variable *vent* en suivant la même logique

plot(maxO3~vent, data=ozone,xlab='Secteur du vent', ylab="pic d'ozone")
ici equivalent à
boxplot(maxO3~vent, data=ozone,xlab='Secteur du vent', ylab="pic d'ozone")

Annotez les parties de la boite à moustaches (*boxplot*).
Vous pouvez aussi personnaliser le graphe. Ajoutez les couleurs:

```
col=c("blue","red","green", "yellow")
boxplot(maxO3~vent, data=ozone,xlab='Secteur du vent', ylab="pic
d'ozone",col=c("blue","red","green", "yellow"))
```

Si vous avez des difficultés à comprendre cette représentation graphique faites un exercice :
On s'intéresse qu'à la catégorie l'Est.
Sélectionnez une partie du tableau qui correspond au vent de l'Est.
Faites une boite à moustache :

```
boxplot
```

observez les valeurs dans l'ordre :

```
sort
```

Calculez la moyenne :

```
mean
```

Calculez les quantiles/ a quoi correspond-ils?

```
quantile
```

Essayez maintenant d'observer ces éléments sur le graphe.

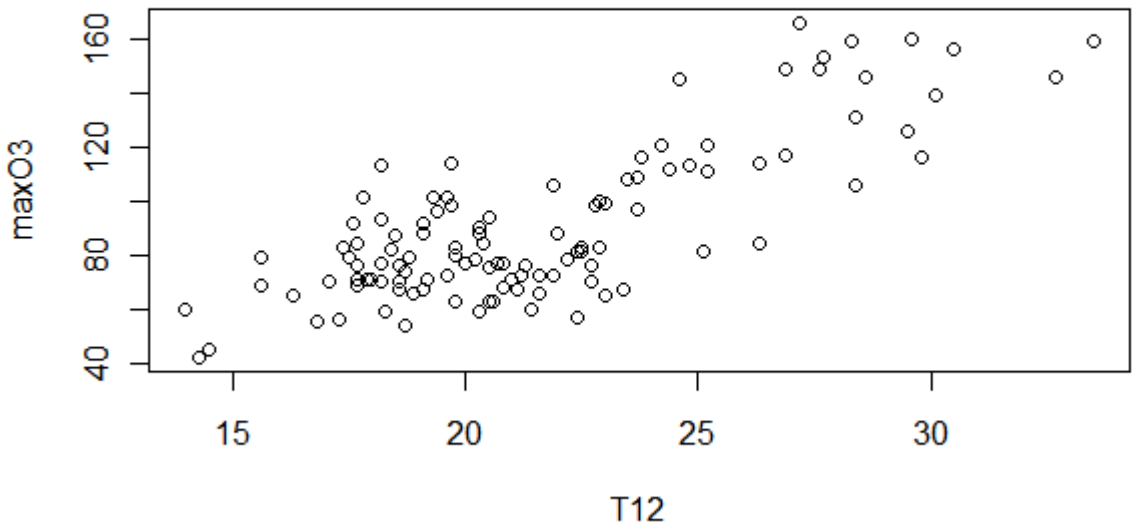
Pour visualiser deux variables qualitatives essayez:

```
plot(pluie~vent, data=ozone)
```

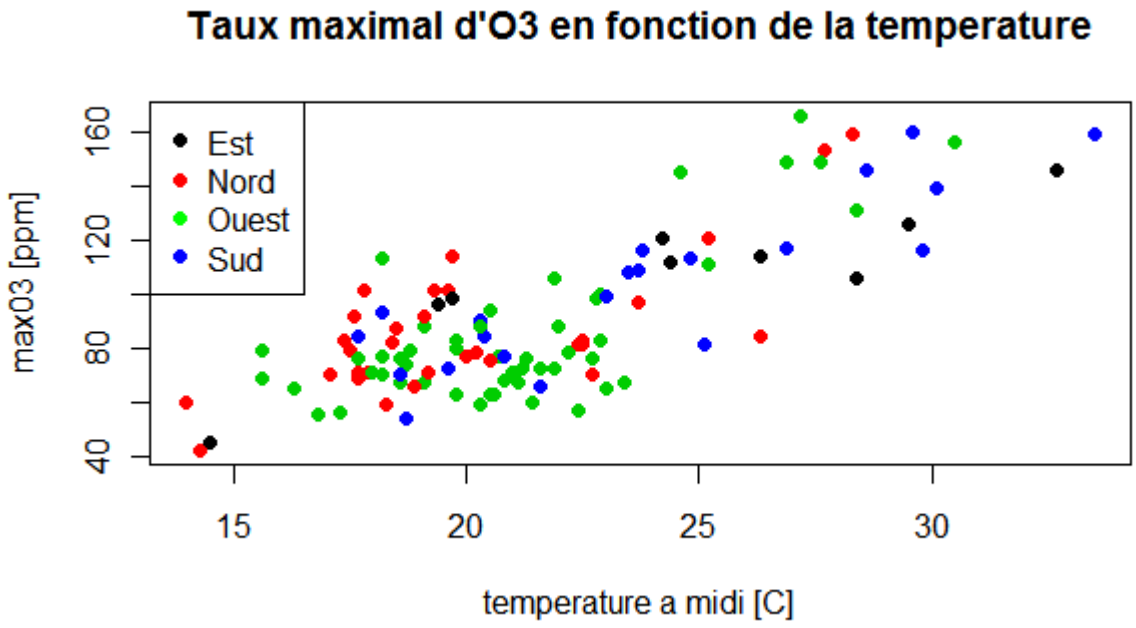
De la même façon, faites un graphe de direction du vent en fonction de température
Visualiser la distribution :

```
hist(ozone$maxO3,xlab='Ozone',main='Histogram')
```

Décrivez le graphe, en quels unités est l'axe y?
Personnaliser les *plots*
Utiliser les fonctions citées au-dessous pour changer votre plot



en



utilisant les informations ci-dessus:

FIGURE 3.9 – Symbole obtenu pour la valeur de l'argument pch.

Argument	Description
`adj`	contrôle la justification du texte par rapport au bord gauche du texte : 0 à gauche, 0.5 centré, 1 à droite ; si deux valeurs sont données `c(x,y)` le texte est justifié horizontalement et verticalement
`asp`	précise le ratio entre `y` et `x` : `asp=1` construit des graphes orthonormés
`axes`	par défaut TRUE, les axes et le cadre sont tracés
`bg`	spécifie la couleur de l'arrière-plan 1, 2, ou une couleur proposée par `colors()`
`bty`	contrôle le tracé du cadre, valeurs permises : "o", "l", "7", "c", "u" ou "]" (le cadre ressemblant au caractère correspondant) ; `bty="n"` supprime le cadre
`cex`	contrôle la taille des caractères et des symboles par rapport à la valeur par défaut qui vaut 1
`cex.axis`	contrôle la taille des caractères pour l'échelle des axes
`cex.lab`	contrôle la taille des caractères pour les libellés des axes
`cex.main`	contrôle la taille des caractères du titre
`cex.sub`	contrôle la taille des caractères du sous-titre
`col`	précise la couleur du graphe, valeurs possibles 1, 2, ou une couleur proposée par `colors()`
`col.axis`	précise la couleur des axes
`col.main`	précise la couleur du titre
`font`	contrôle le style du texte (1 : normal, 2 : italique, 3 : gras, 4 : gras-italique)
`font.axis`	contrôle le style pour l'échelle des axes
`font.lab`	contrôle le style pour les libellés des axes
`font.main`	contrôle le style du titre
`font.sub`	contrôle le style du sous-titre
`las`	contrôle la disposition des annotations sur les axes (0, valeur par défaut : parallèles aux axes, 1 : horizontales, 2 : perpendiculaires aux axes, 3 : verticales)
`lty`	contrôle le type de ligne tracée, (1 : continue, 2 : tirets, 3 : points, 4 : points et tirets alternés, 5 : tirets longs, 6 : tirets courts et longs alternés), ou bien écrire en toutes lettres "solid", "dashed", "dotted", "dotdash", "longdash", "twodash" ou "blank" (pour ne rien écrire)
`lwd`	contrôle l'épaisseur des traits
`main`	précise le titre du graphe, par exemple `main="titre"`
`mfrow`	vecteur `c(nr,nc)` qui divise la fenêtre graphique en `nr` lignes et `nc` colonnes ; les graphes sont ensuite dessinés en ligne
`offset`	précise le décalage du texte par rapport au point (fonction `text`)

GGPLOT2 - ajoutons une couche

Pour installer **GGPLOT2,** effectuez la commande:

```
> install.packages("ggplot2")
```

et ensuite

```
> library(ggplot2)
```

Dans le concept de ggplot2 les graphs sont composes de différents couches superposées.
En utilisant le même jeu de données tapez

```
ggplot(ozone, aes(x = T12, y = maxO3))
```

R vous informe que le graphe n'a pas de couches. Sauvegardez le plot dans un objet p et ajoutez une couche

```
p <- ggplot(ozone, aes(x = T12, y = maxO3))
p <- p + geom_point()
p
```

Faites attention au operateur + qui va servir toujours a ajouter les couches. Chaque nouvelle couche c'est une fonction qui prend en argument ses caractéristiques
Essayez :

```
ggplot(ozone, aes(x = T12, y = maxO3)) +  geom_point(color = "red", size = 5)
```

Une couche statistique

```
p <- p + stat_smooth()
```

La ligne représente un 'fit' et la bande grise l'intervalle de confiance établi en utilisant la méthode 'loess'. Vous pouvez aussi visualiser la ligne sans les points

```
ggplot(ozone, aes(x = T12, y = maxO3)) + stat_smooth()
```

Pour bien annoter le graphe:

```
p <- ggplot(ozone, aes(x = T12, y = maxO3))+ geom_point(color = "red", size =
5)+ylab("taux maximal d'O3")+
  xlab("temperature a midi")+
  theme_bw()+
  opts(title = "taux d'ozone en fonction de la temperature a Rennes")
```

Que fait **theme_bw() ?**
C'est facile aussi de colorier les points en fonction du type de vent

```
p <- ggplot(ozone, aes(x = T12, y = maxO3,color = vent))+ geom_point(size =
3)+ylab("taux maximal d'O3")+
  xlab("temperature a midi")+
  theme_bw()+
  opts(title = "taux d'ozone en fonction de la temperature a Rennes")
```

Observez que la légende apparaît toute seule !
Vous pouvez joindre les points avec les lignes par groupe aussi :

```
p <- ggplot(ozone, aes(x = T12, y = maxO3,color = vent))+ geom_point(size =
3)+ylab("taux maximal d'O3")+
  xlab("temperature a midi")+
  theme_bw()+
  opts(title = "taux d'ozone en fonction de la temperature a Rennes")+
```

```
  geom_line()
```

Il est aussi possible de mapper les couleurs de valeurs continues

```
p <- ggplot(ozone, aes(x = T12, y = maxO3,color = maxO3))+ geom_point(size =
3)+ylab("taux maximal d'O3")+
  xlab("temperature a midi")+
  theme_bw()+
  opts(title = "taux d'ozone en fonction de la temperature a Rennes")
```

BARPLOT

```
p <- ggplot(ozone, aes(x =vent))+geom_bar()
```

avec les couleurs

```
p <- ggplot(ozone, aes(x =vent, fill=vent))+geom_bar()
```

Pour changer la palette de coloration ajoutez

```
+scale_fill_brewer(palette = "Set1")
```

Ou décidez quels couleurs vous préférez par vous-mêmes

```
+scale_fill_manual(values=c("bisque", "chartreuse4",
                            "hotpink","yellow"))
```

ERROR BARS

```
ggplot(ozone, aes(vent, maxO3,fill=vent))+
  stat_summary(fun.y = mean, geom = "bar")+
  stat_summary(fun.data = mean_sdl, geom = "errorbar")
```

mean_sdl : retourne la moyenne du groupe, et les '*error bars'* qui correspondent à l'écart-type.
Vous pouvez essayer aussi
mean_cl_boot(), mean_cl_normal() ou median_hilow()

Documentation:

- mean_cl_boot()
  - This will return the sample mean, and 95% bootstrap confidence intervals.
- mean_cl_normal()
  - This will return the sample mean, and the 95% percent Gaussian confidence interval based on the t-distribution
- mean_sdl()
  - This will return the sample mean and values at 1 sd and -1 sd away. You can make it return points any arbitrary number of sds away by passing that value to mult. For example, mult = 2 will return 2 and -2 sds.
- median_hilow()
  - This will return the sample median, and confidence intervals running from the 0.025 quantile to the 0.975 quantile, which covers 95% of the range of the data. You can change what range of the data you want the confidence interval to cover by passing it to conf.int. For example conf.int = 0.5 will return confidence intervals ranging from the 0.25 quantile to the 0.75 quantile.

Expliquez pourquoi il y a plusieurs options pour les 'error bars' et leur signification.

Densité

Exécutez

```
ggplot(ozone, aes(maxO3, T12))+
  stat_density2d()+geom_point()
```

Vous pouvez jouer avec la coloration :

```
ggplot(ozone, aes(maxO3, T12))+
  stat_density2d(geom = "point",contour = F,
        aes(size = ..density..), alpha = 0.3)

ggplot(ozone, aes(maxO3, T12))+
  stat_density2d(geom = "tile", contour = F, aes(fill = ..density..))
```

ou bien

```
ggplot(ozone, aes(maxO3, T12))+
  stat_density2d(geom = "tile", contour = F, aes(fill = ..density..))
+scale_fill_gradientn(colours = rainbow(6))
```