Premiers pas avec les graphiques sous R

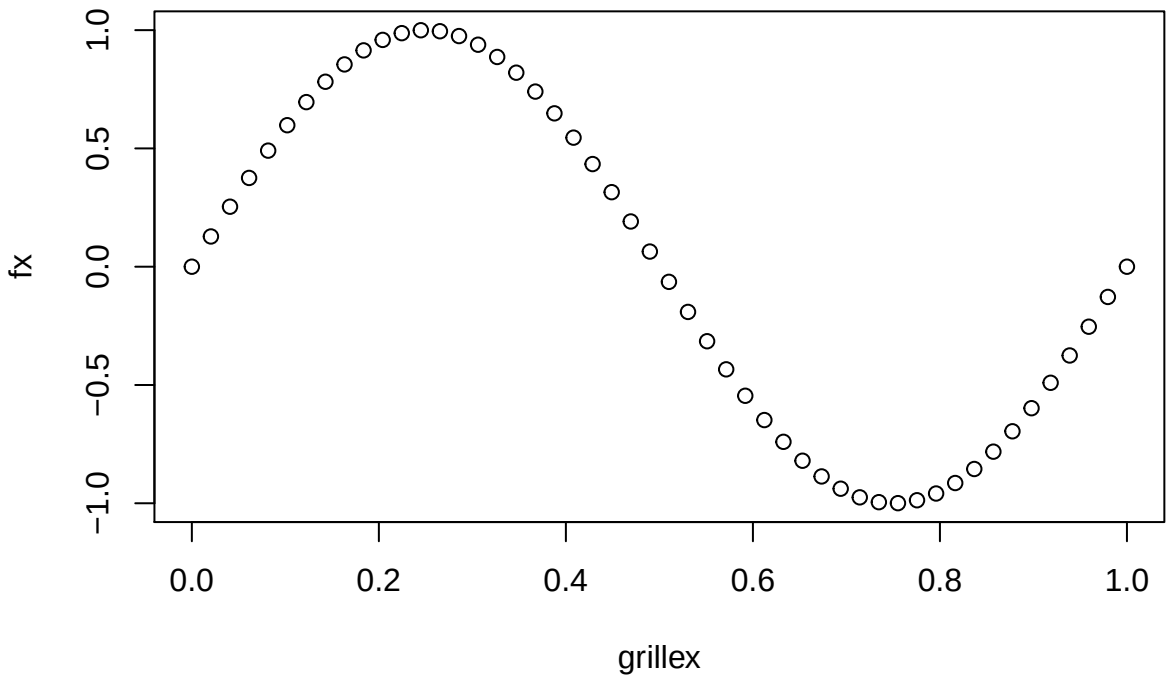
B.Thieurmel - benoit.thieurmel@datastorm.fr

MENESR/DEPP/CISAD - 11/12/2017

Les graphiques : introduction à la fonction plot

On représente facilement une fonction avec la fonction **plot()**, et deux vecteurs de même taille en entrée :

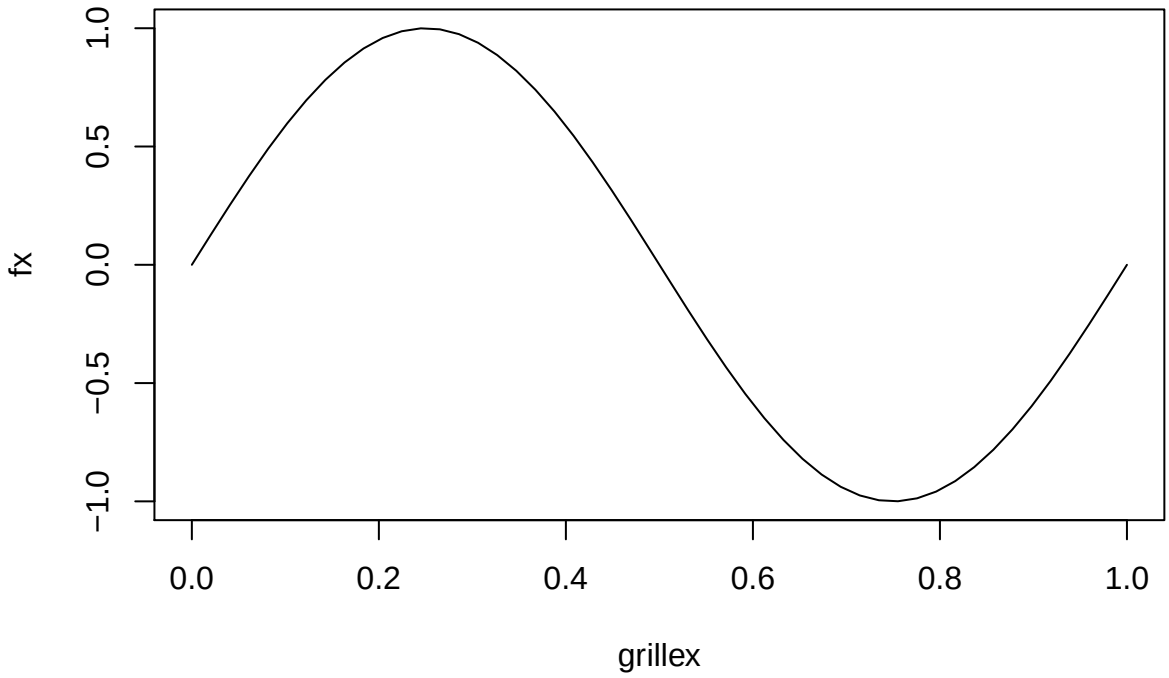
```
grillex <- seq(0, 1,length=50)
fx <- sin(2*pi*grillex)
plot(x = grillex, y = fx) # plot(fx~grillex) notation formule
```



Type de graphique : ligne ou points ?

l'argument **type** contrôle le type du graphique. Les plus courants sont : **p** *(points)*, **l** *(ligne)*, **b** *(ligne + points)*, **s** *(step)*,...

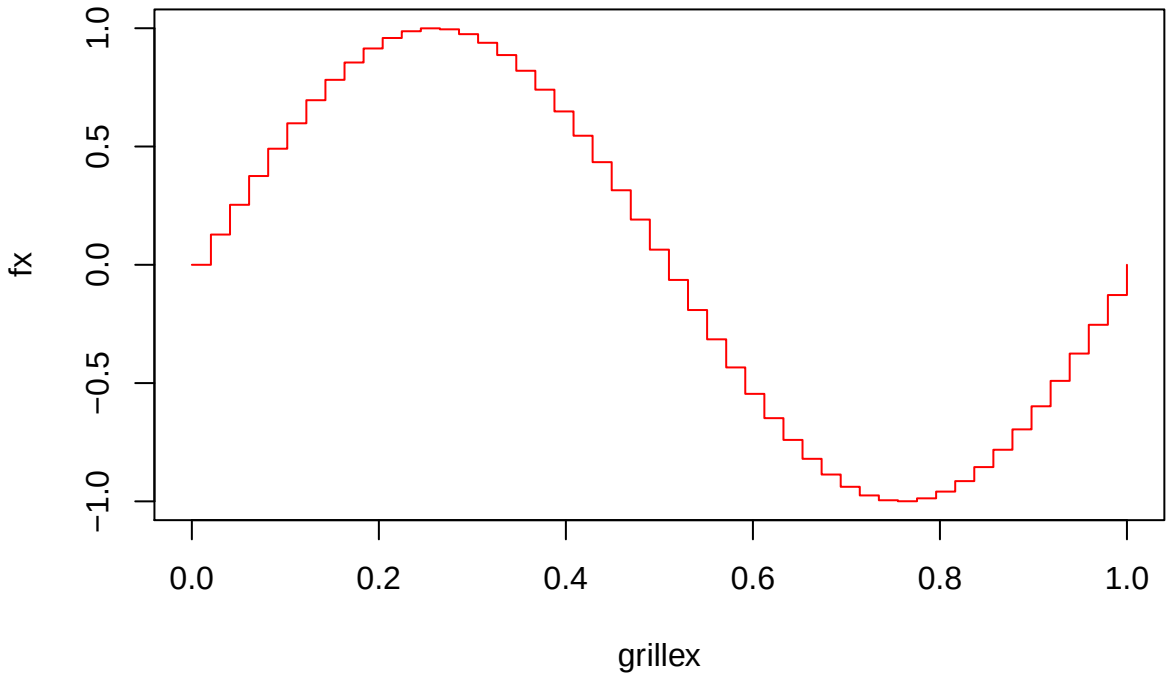
```
plot(x = grillex, y = fx, type = "l")
```



Les couleurs

En utilisant l'argument **col** :

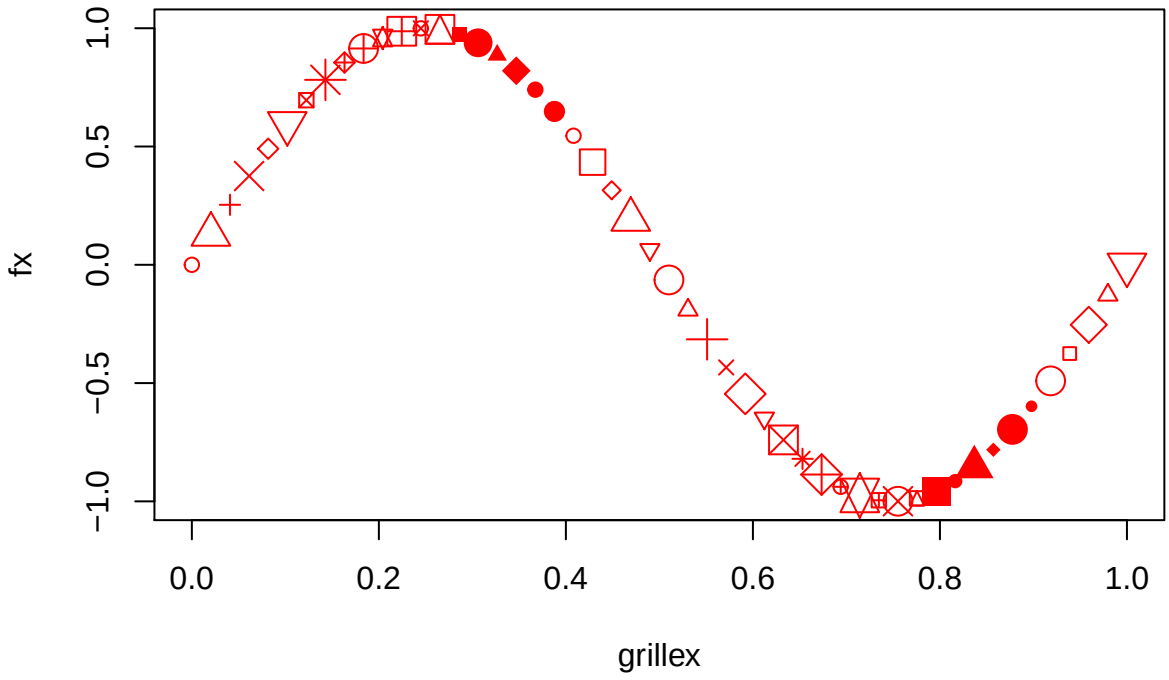
```
plot(x = grillex, y = fx, type = "s", col = "red")
```



Type de points

On gère le type des points avec **pch** (_?points). **cex** fixe la taille des p oints :

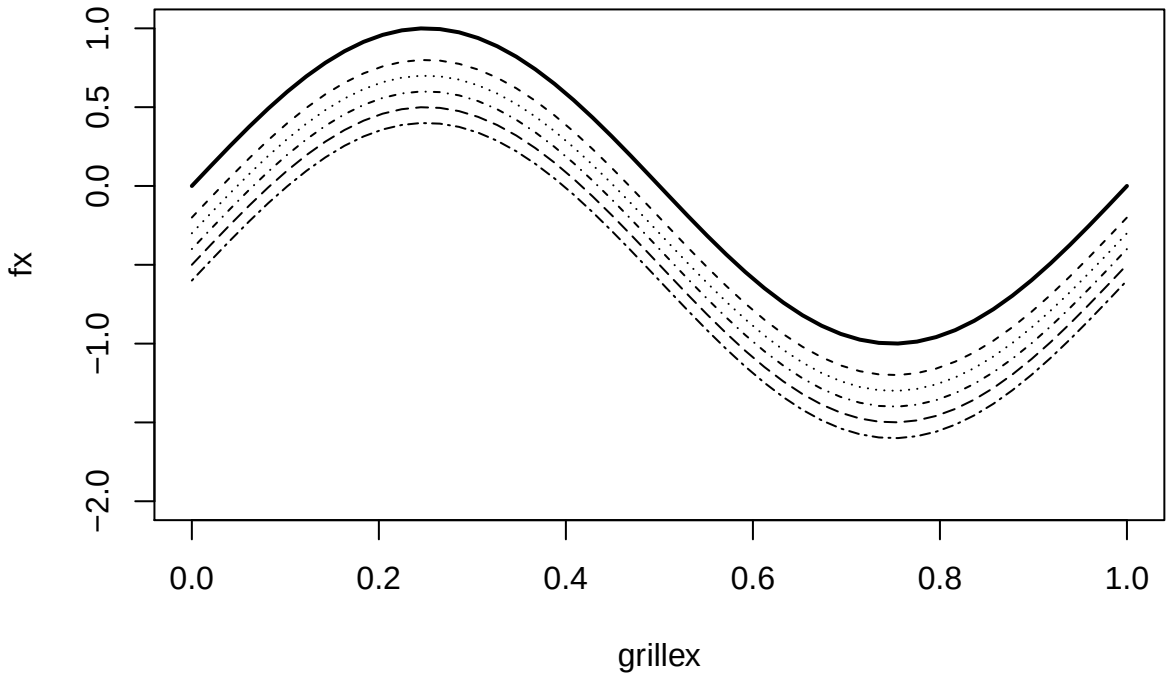
```
plot(x = grillex, y = fx, col = "red", pch = 1:25, cex = c(1:2))
```



Type de lignes

On gère le type des lignes avec **lty** (_?par). **lwd** fixe l'épaisseur :

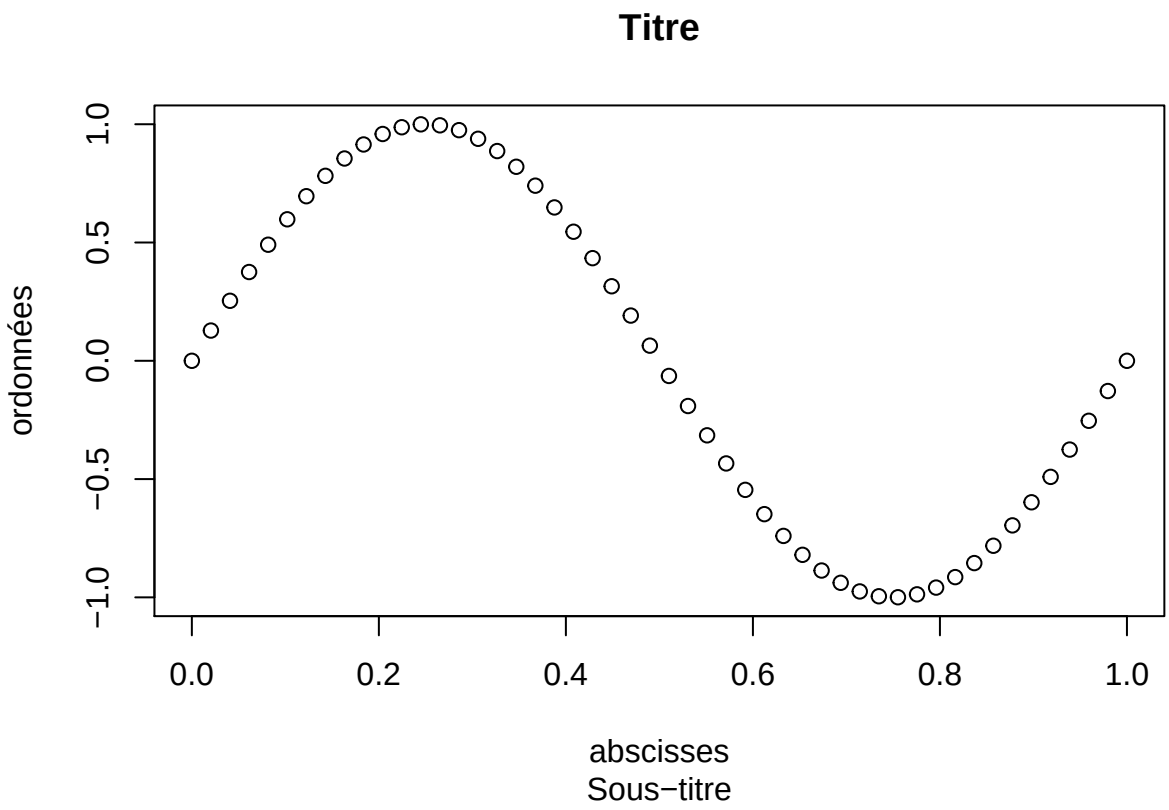
```
plot(x = grillex, y = fx, type = "l", lty = 1, lwd = 2, ylim = c(-2, 1))
for(i in 2:6) lines(x = grillex, y = fx - (i/10), lty = i)
```



Titre et intitulés des axes

main : titre, **sub** : sous-titre, **xlab** : intitulé des abscisses, **ylab** : intitulé des ordonnées

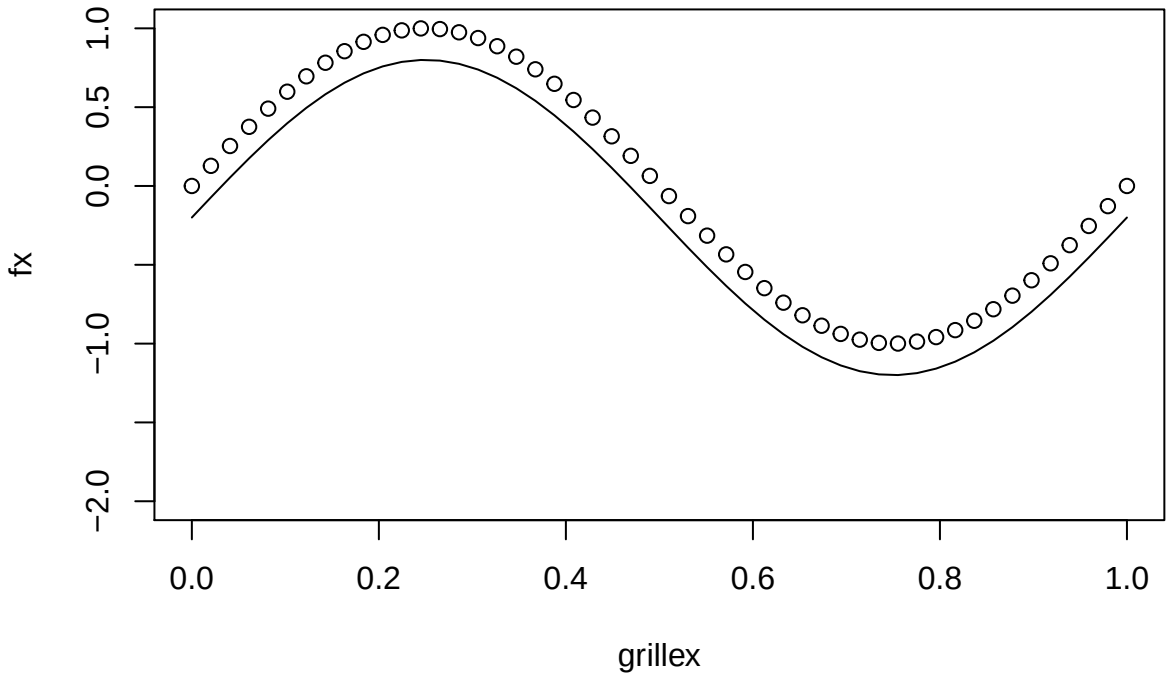
```
plot(x = grillex, y = fx, main = "Titre", sub = "Sous-titre",
     xlab = "abscisses", ylab = "ordonnées")
```



Ajout de lignes

lines() : ajout d'une nouvelle ligne

```
plot(x = grillex, y = fx, ylim = c(-2, 1))
lines(x = grillex, y = fx - 0.2)
```



---

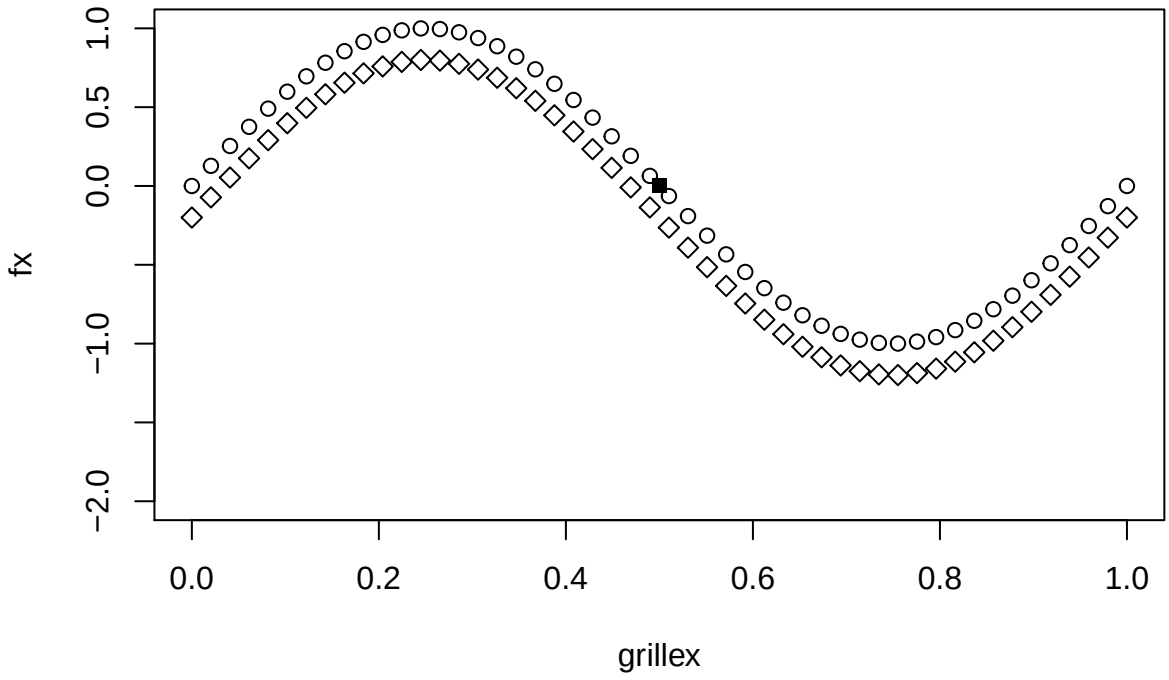
abline() : ajout de droites (**h** horizontales, **v** verticales, **a** & **b** avec coefficients)

```
plot(x = grillex, y = fx)
abline(h = 0, lty = 2) ; abline(v = 0, lty = 2)
abline(a = -1, b = 2)
```

Ajout de points

fonction **points()**

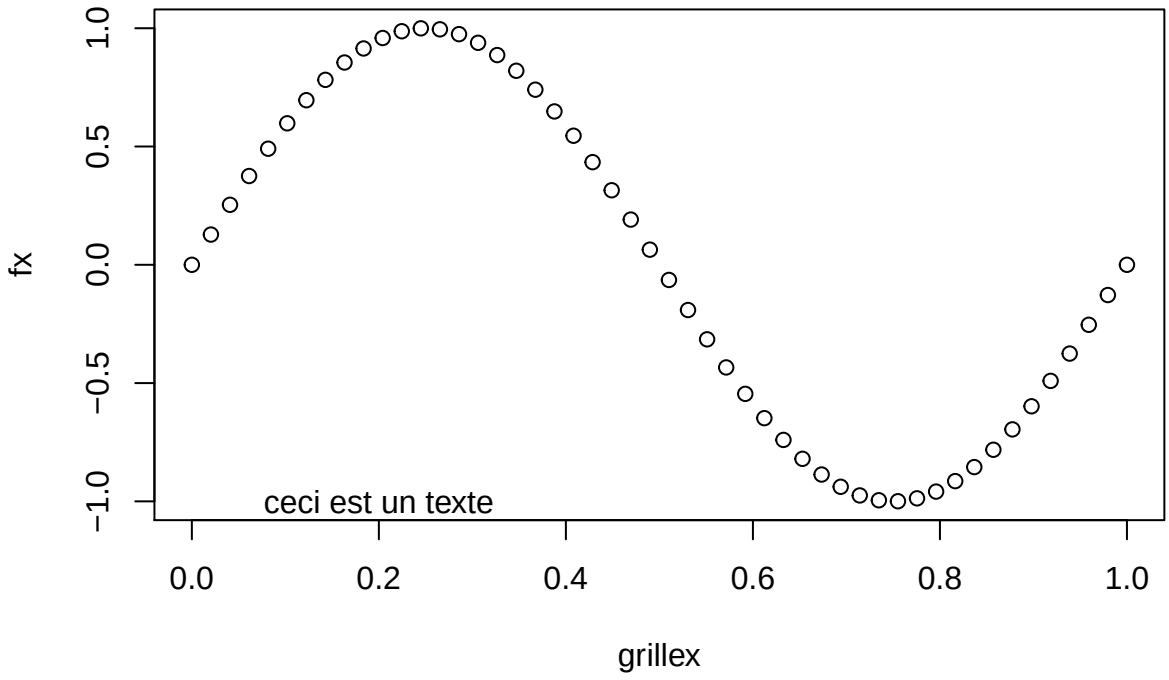
```
plot(x = grillex, y = fx, ylim = c(-2, 1))
points(x = grillex, y = fx - 0.2, pch = 5)
points(x = 0.5, y = 0, pch = 15)
```



Ajout de texte

En utilisant la fonction **text()**

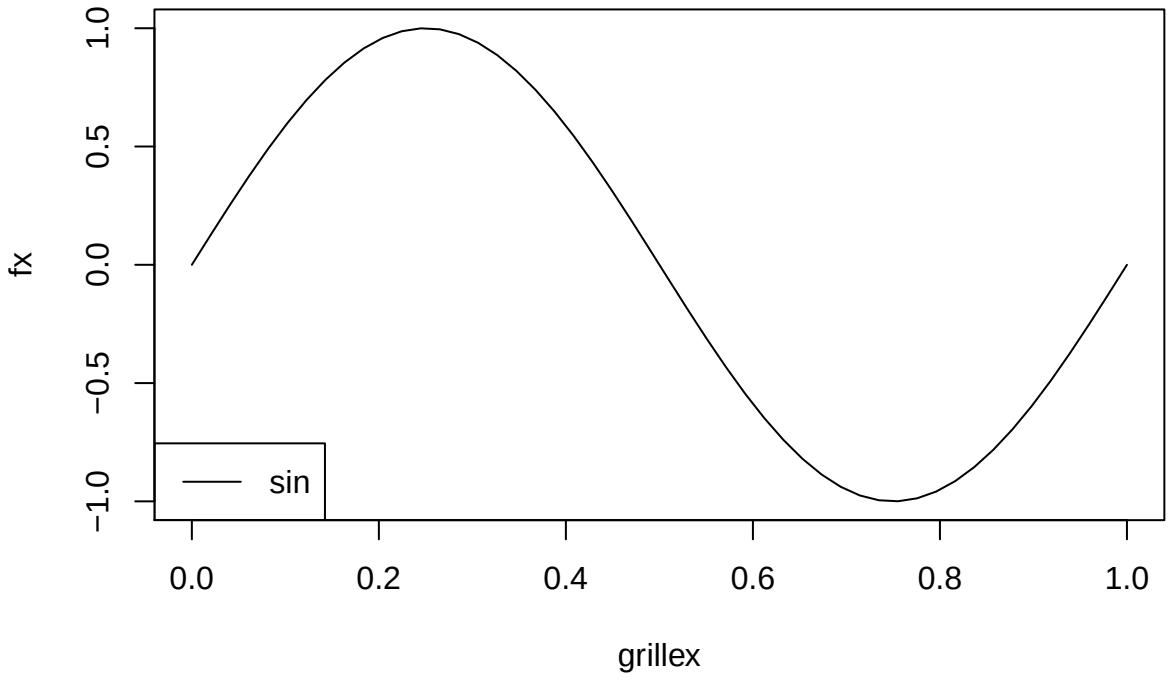
```
plot(x = grillex, y = fx)
text(x = 0.2, y = -1, labels ="ceci est un texte")
```



Ajout d'une légende

Simplement avec la fonction **legend()**

```
plot(x = grillex, y = fx, type = "l")
legend("bottomleft",legend="sin", col=1, lty=1)
```



Les graphiques de statistique univariés

Les données d'exemples :

- deux variables quantitatives :
  - la température *T12*
  - l'ozone *maxO3*
- deux variables qualitatives / facteurs :
  - le vent *vent*
  - la pluie *pluie*

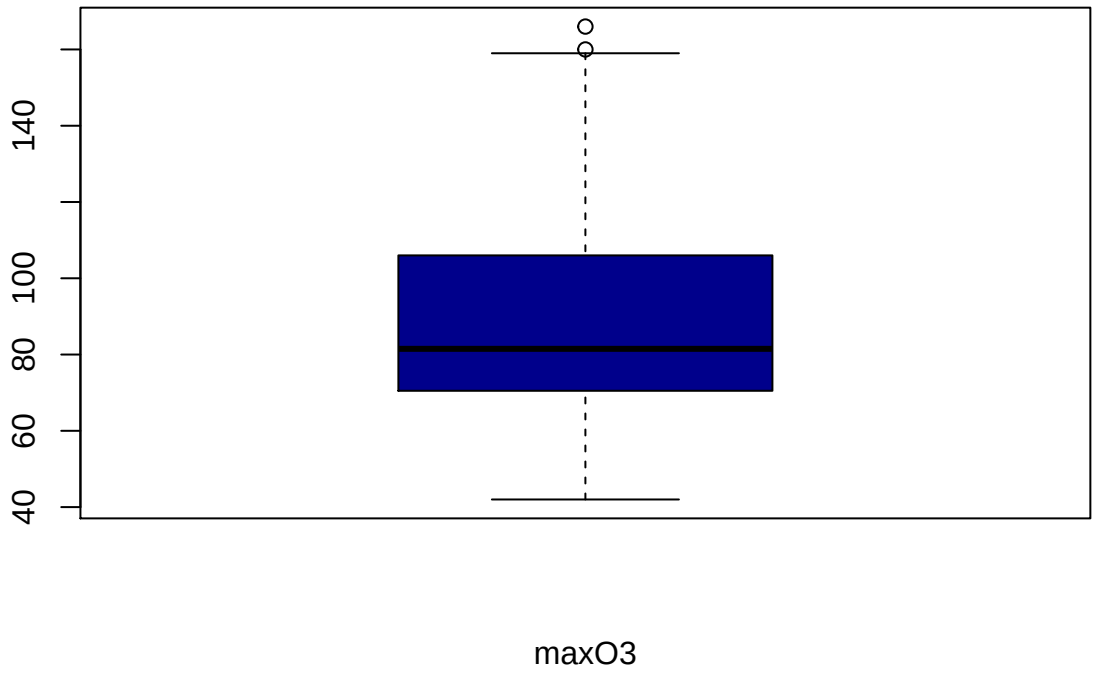
```
ozone <- read.csv("ozone_plot.txt", sep="")
summary(ozone)
```

```
##       T12             maxO3           vent      pluie
##  Min.   :14.00   Min.   : 42.00   Est  :10   Pluie:43
##  1st Qu.:18.60   1st Qu.: 70.75   Nord :31   Sec  :69
##  Median :20.55   Median : 81.50   Ouest:50
##  Mean   :21.53   Mean   : 90.30   Sud  :21
##  3rd Qu.:23.55   3rd Qu.:106.00
##  Max.   :33.50   Max.   :166.00
```

Boxplot

Disponible avec la fonction **boxplot()**

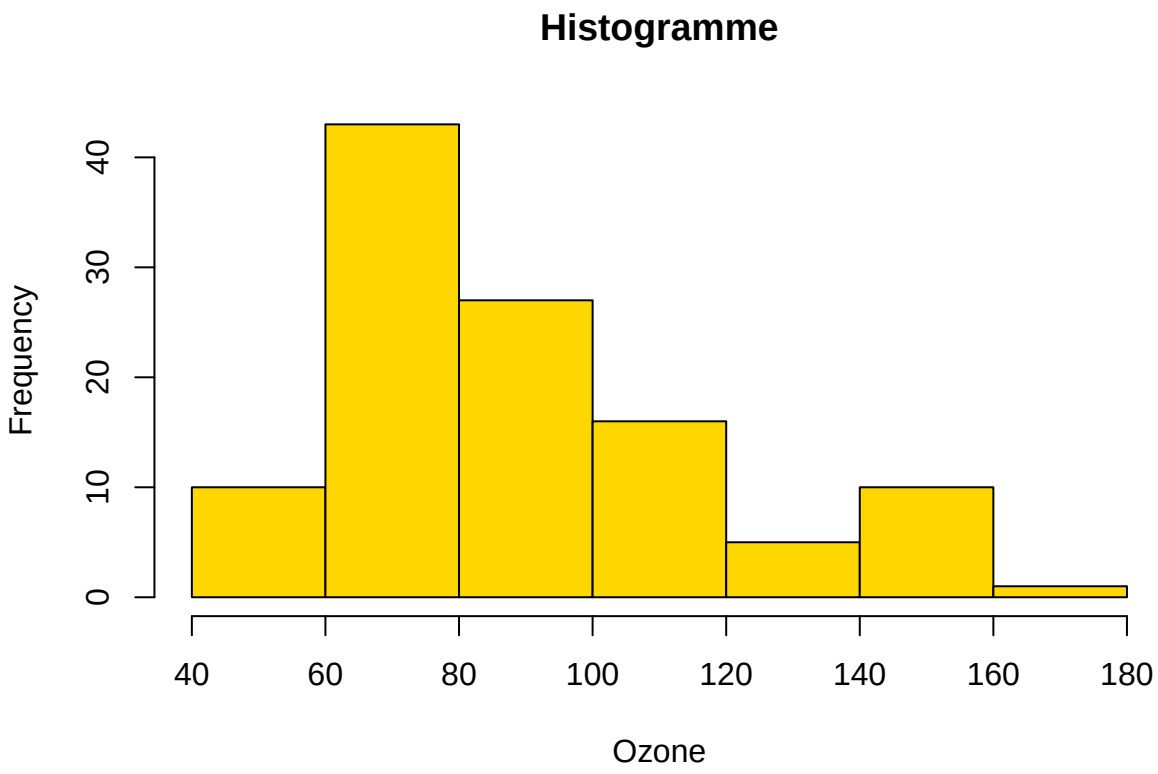
```
boxplot(ozone$maxO3, col = "darkblue", xlab = "maxO3")
```



Histogramme

Disponible avec la fonction **hist()**

```
hist(ozone[,"maxO3"], main="Histogramme", xlab = "Ozone", col = "gold")
```



Ajout de l'estimateur à noyaux, avec la fonction **density()**

```
hist(ozone[,"T12"], prob=TRUE, col = "gold")
lines(density(ozone[,"T12"]))
```

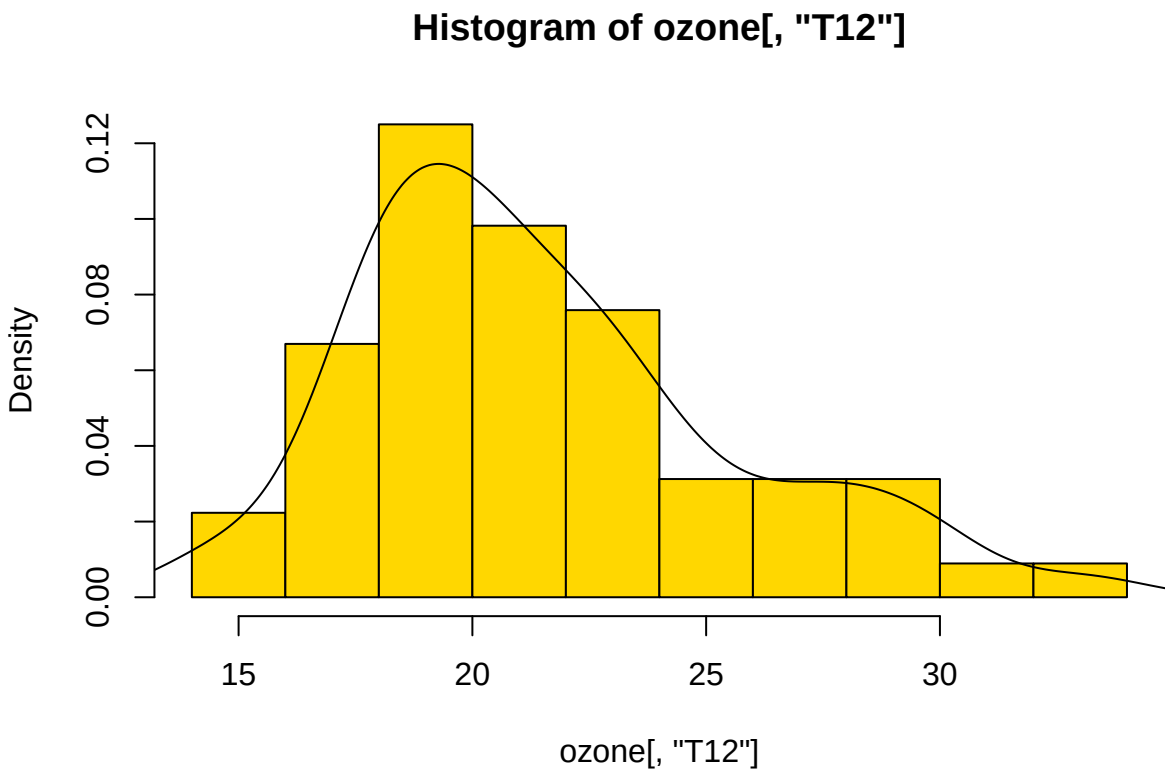
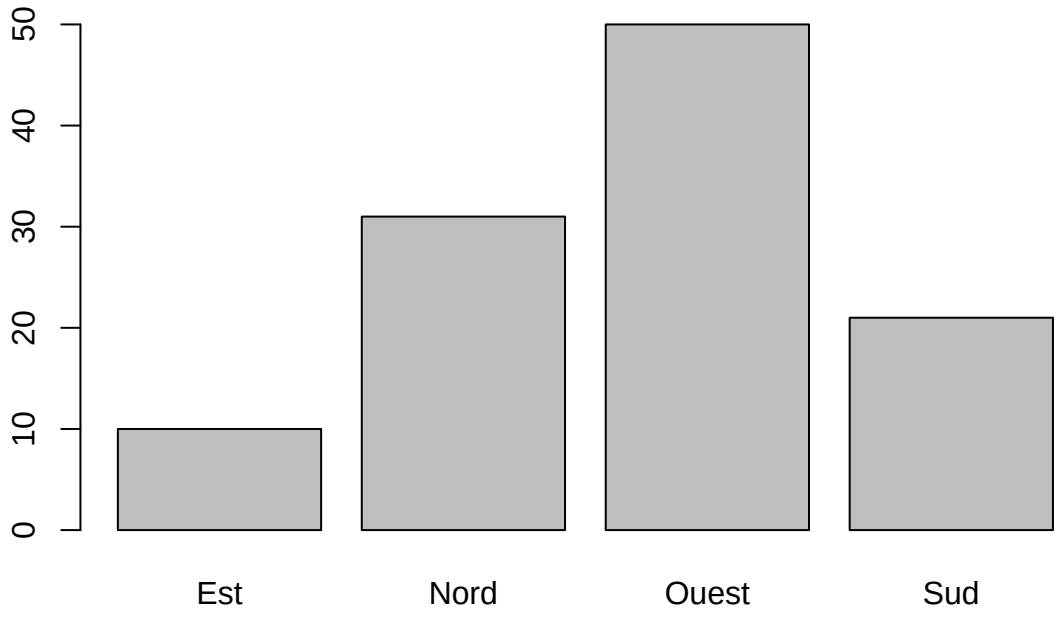


Diagramme en barres

Disponible avec la fonction **barplot()**

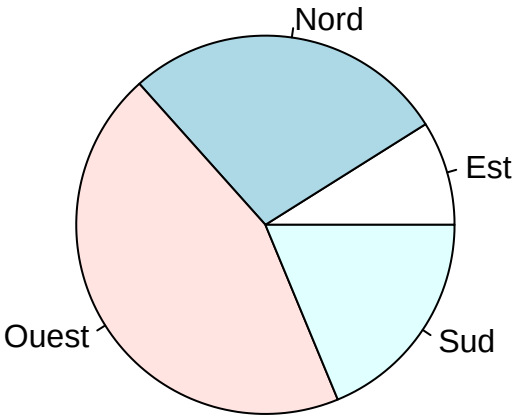
```
barplot(table(ozone[,"vent"]))
```



Camenbert...

Disponible avec la fonction **pie()**

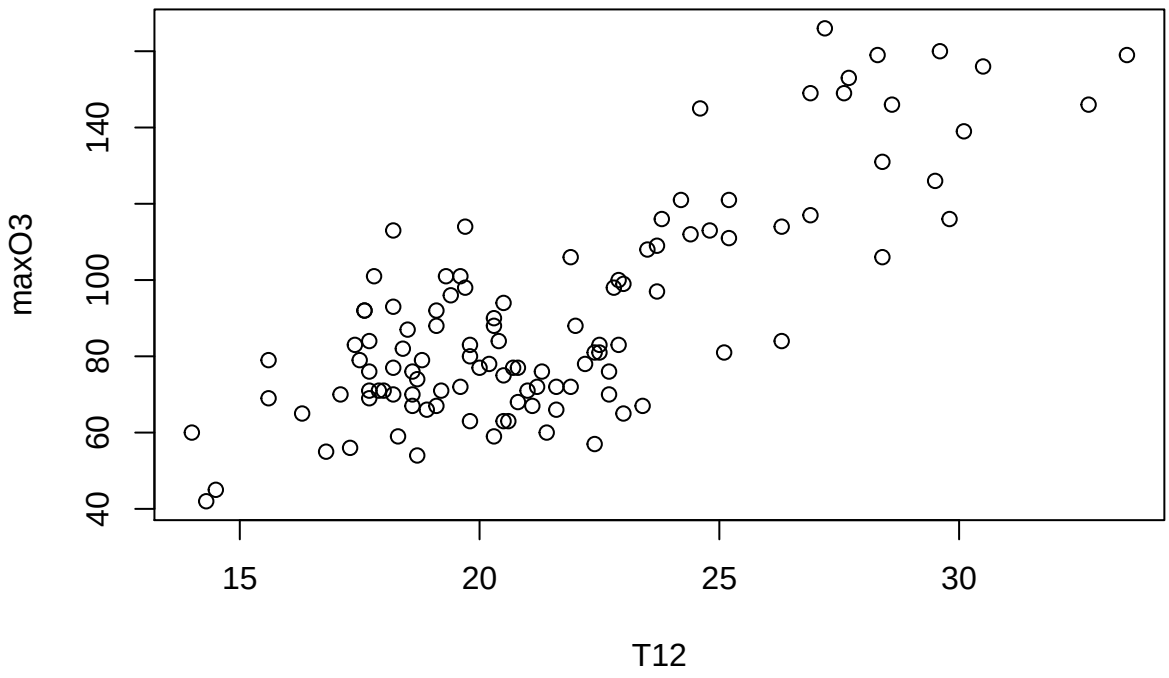
```
pie(table(ozone[,"vent"]))
```



Les graphiques de statistique bivariés

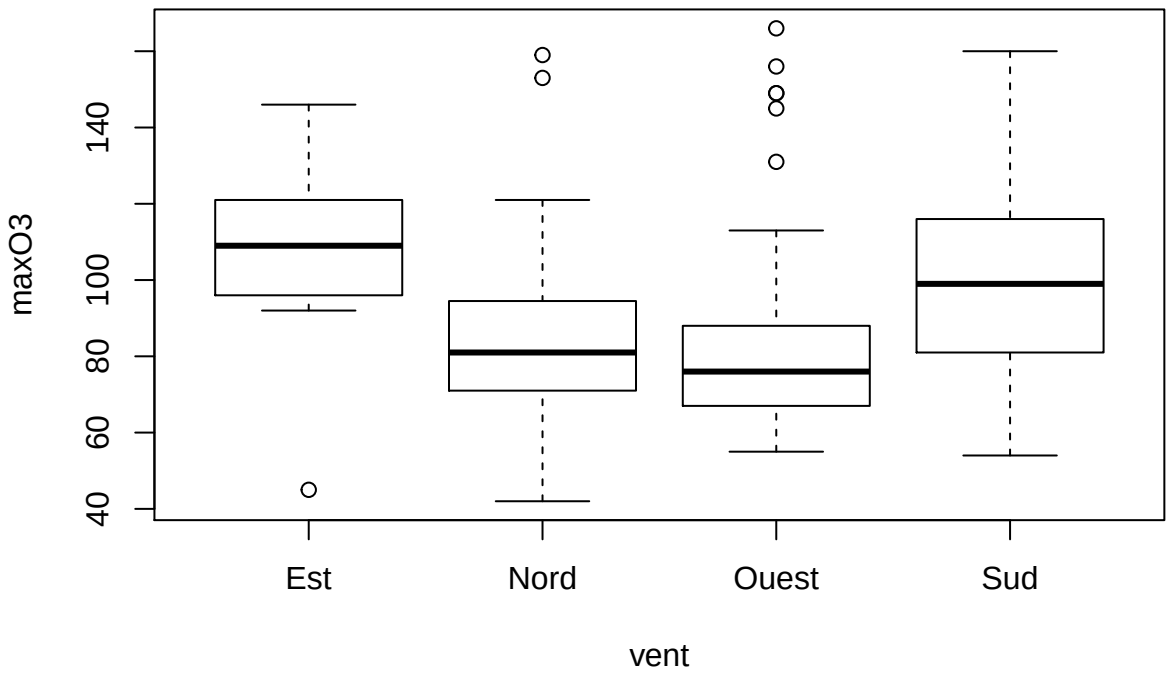
Quantitatif-Quantitatif

```
plot(maxO3~T12,data=ozone)
```



Quantitatif-Qualitatif

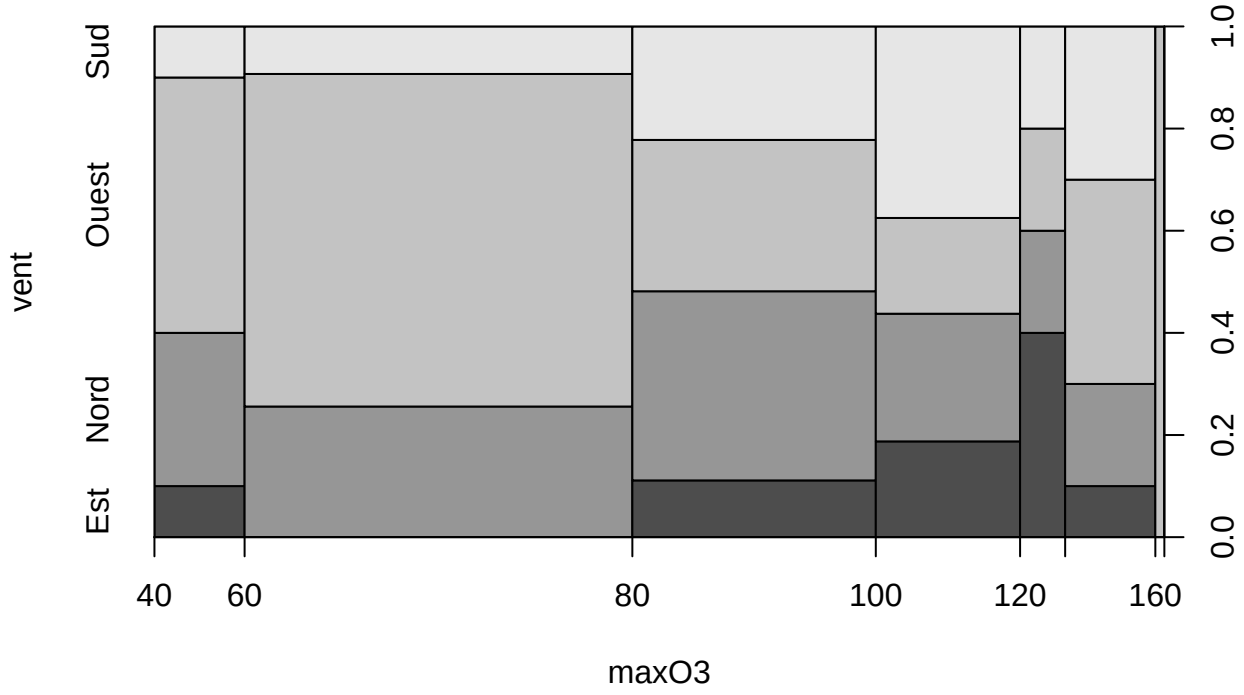
```
plot(maxO3~vent, data=ozone)
```




```
#  (=) boxplot(maxO3~vent,data=ozone)
```

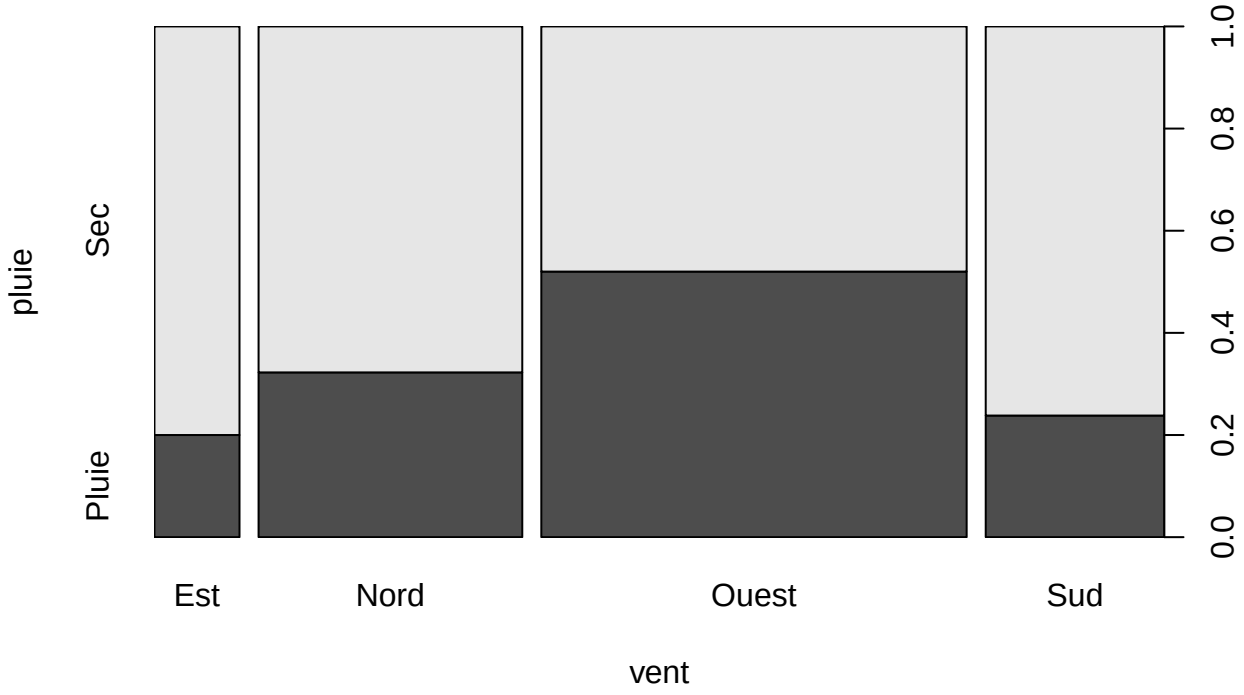
Et dans l'autre sens ? Peut-être un peu dur à lire. . . !

```
plot(vent~maxO3, data=ozone)
```



Qualitatif-Qualitatif

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



Pour aller plus loin

Démo

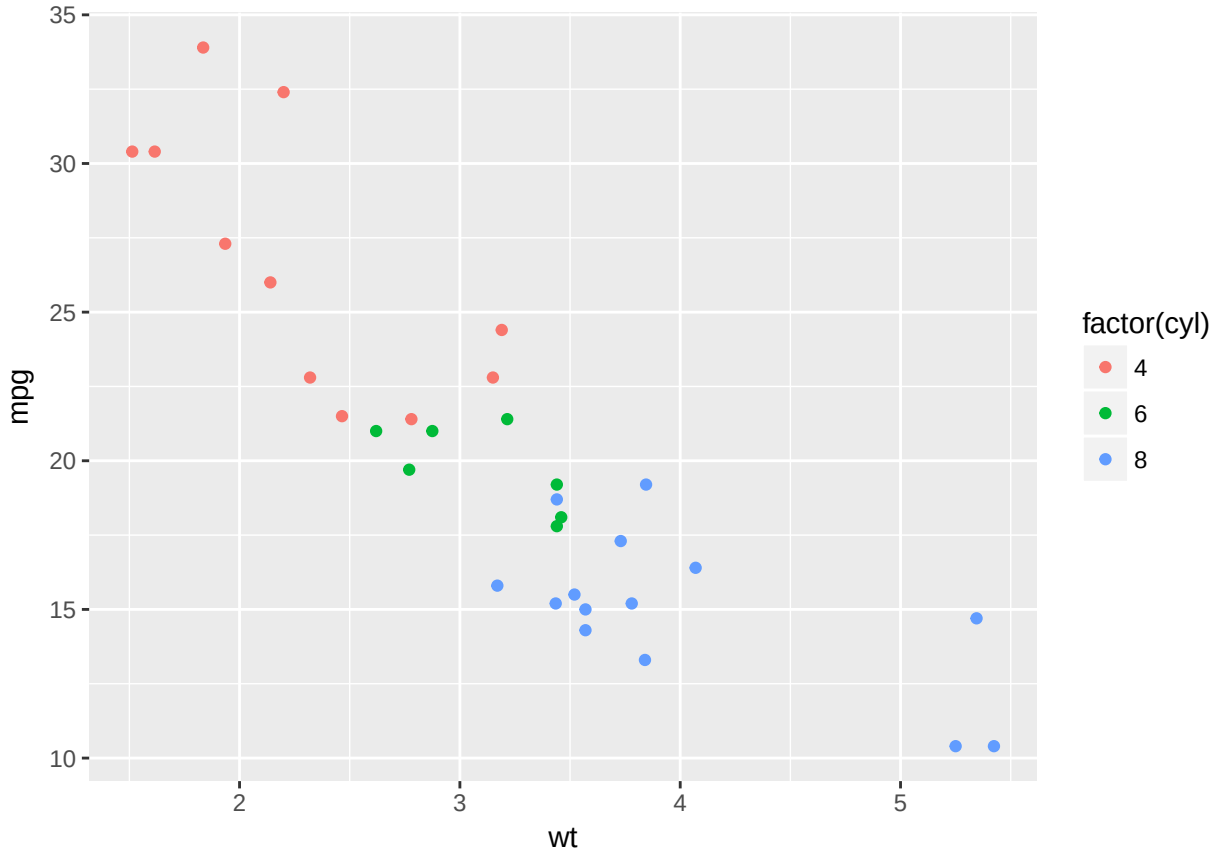
- demo(image)
- example(contour)
- demo(persp)
- require("lattice"); demo(lattice)
- example(wireframe)
- require("rgl"); demo(rgl)
- example(persp3d)
- demo(plotmath); demo(Hershey)

---

ggplot2

ggplot2 a été développé par Hadley Wickham comme une implémentation de *Grammar of Graphics*. C'est un package relativement complet et puissant. (http://ggplot2.org/, http://docs.ggplot2.org/current/, https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf)

```
ggplot(mtcars, aes(wt, mpg)) + geom_point(aes(colour = factor(cyl)))
```



Graphiques intéractifs

En utilisant des packages faisant le pont entre **R** et des librairies **javascript** :

- **googleVis** : https://cran.r-project.org/web/packages/googleVis/vignettes/googleVis_examples.html
- **htmlwidgets** (http://www.htmlwidgets.org/) et tous ses enfants (http://gallery.htmlwidgets.org/) :
  - **dygraphs** pour les séries temporelles : http://rstudio.github.io/dygraphs
  - **leaflet** pour les cartes : http://rstudio.github.io/leaflet/
  - **visNetwork** (http://datastorm-open.github.io/visNetwork) et **networkD3** (http://christophergandrud.github.io/networkD3/) pour les réseaux
  - **d3heatmap** pour des heatmaps : https://github.com/rstudio/d3heatmap
  - **DT** pour des tableaux intéractifs : http://rstudio.github.io/DT/
  - **threejs** (https://github.com/bwlewis/rthreejs) et **rglwidget** (http://cran.at.r-project.org/web/packages/rglwidget/index.html) pour des représentations 3D
  - **plotly** pour passer du **ggplot2** en intéractif : https://plot.ly/r/
  - ....