MUSÉUM NATIONAL D'HISTOIRE NATURELLE

SEMIN-R

Une nouvelle génération de graphiques : introduction à ggplot2

SUEUR, Jérôme

Muséum national d'Histoire naturelle
UMR 7205 OSEB

Une nouvelle génération de graphiques : introduction à ggplot2

Jérôme Sueur

MNHN – Systématique et Evolution
UMR CNRS 7205 – OSEB
sueur@mnhn.fr

20 Mai 2010

Plan

Introduction

- développé par Hadley Wickham (Rice University, Houston, USA)
- dépend principalement des packages `reshape` et `plyr` développés par H. Wikcham
- première version : Juin 2007

Introduction

- obéit à de nouveaux principes esthétiques
- obéit à une construction particulière suivant la grammaire graphique développée par Wilkinson en 2005 [1]
- permet une construction rapide de graphiques simples
- réduction (considérable) de la longueur des codes
- permet la création de nouvelles familles de graphiques

1. Wilkinson, L. (2005) – *The grammar of graphics*. Statistics and Computing. Springer

Principe

Un monde à part...

Il faut oublier tout ce que vous savez sur la création de graphiques avec `R`.

Adieu à `plot()`, `hist()`, `par()`, `layout()`, `points()`, `lines()`, `legend()`, etc!!

Principe

`ggplot2` adapte à `R` le principe de la grammaire de Wilkinson (2005) par l'utilisation de **layers** (calques) qui peuvent s'utiliser comme des objets (assignation possible).

Voici les principaux `layers` :

data	→	données brutes
mapping	→	projection graphique
geom	→	objets géométriques (points, lignes, polygones, etc.)
stat	→	transformation statistique (histogramme, modèle, etc.)
scale	→	espace esthétique (couleurs, formes, tailes, axes, légendes)
coord	→	système de coordonnées (axes, grilles)
facet	→	subdivision (syn. lattice, trellis)

Méthodes de base : fonctions

`ggplot2` a deux fonctions graphiques de base :

- `qplot()` pour **q**uick **plot**
  - rapide mais simple (pour un seul jeu de données et une seule méthode esthétique)
  - principe :

```
qplot(x, y, data=data)
```

- `ggplot()`
  - lent mais plus puissant, ajout de `layers` avec `+`
  - principe :

```
ggplot(data, aes(x, y)) + layers
```

Remarque

On peut obtenir les mêmes résultats avec les deux fonctions (équivalence)

Méthodes de base : format

Format des données

Attention, les données (`data`) doivent toujours être de classe `data.frame`

Jeu de données de référence : propriétés de 54 000 diamants

```
library(ggplot2)
data(diamonds)
class(diamonds)
```

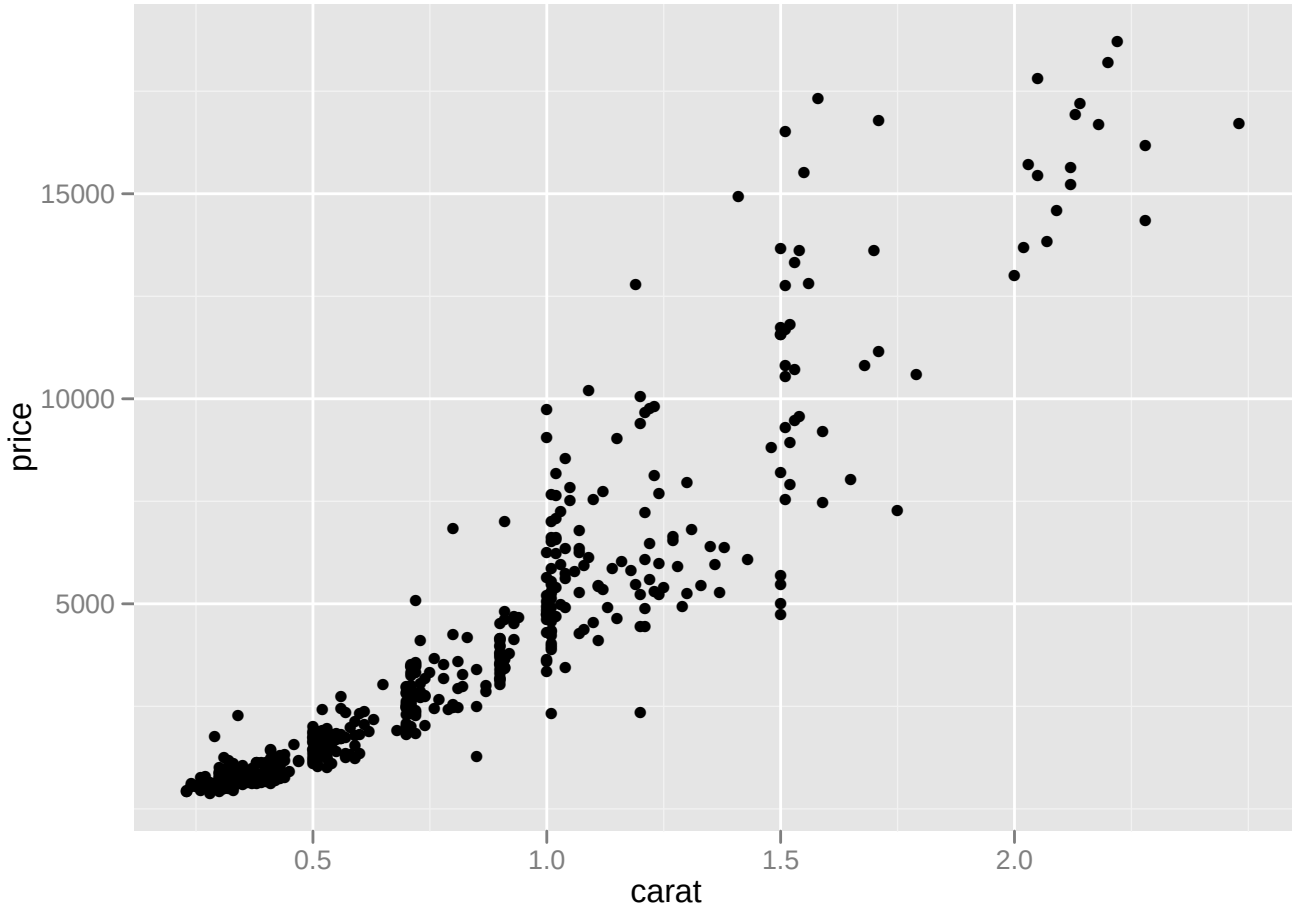
```
[1] "data.frame"
```

```
diamonds <- diamonds[sample(nrow(diamonds), 500), ]
```

qplot : défaut

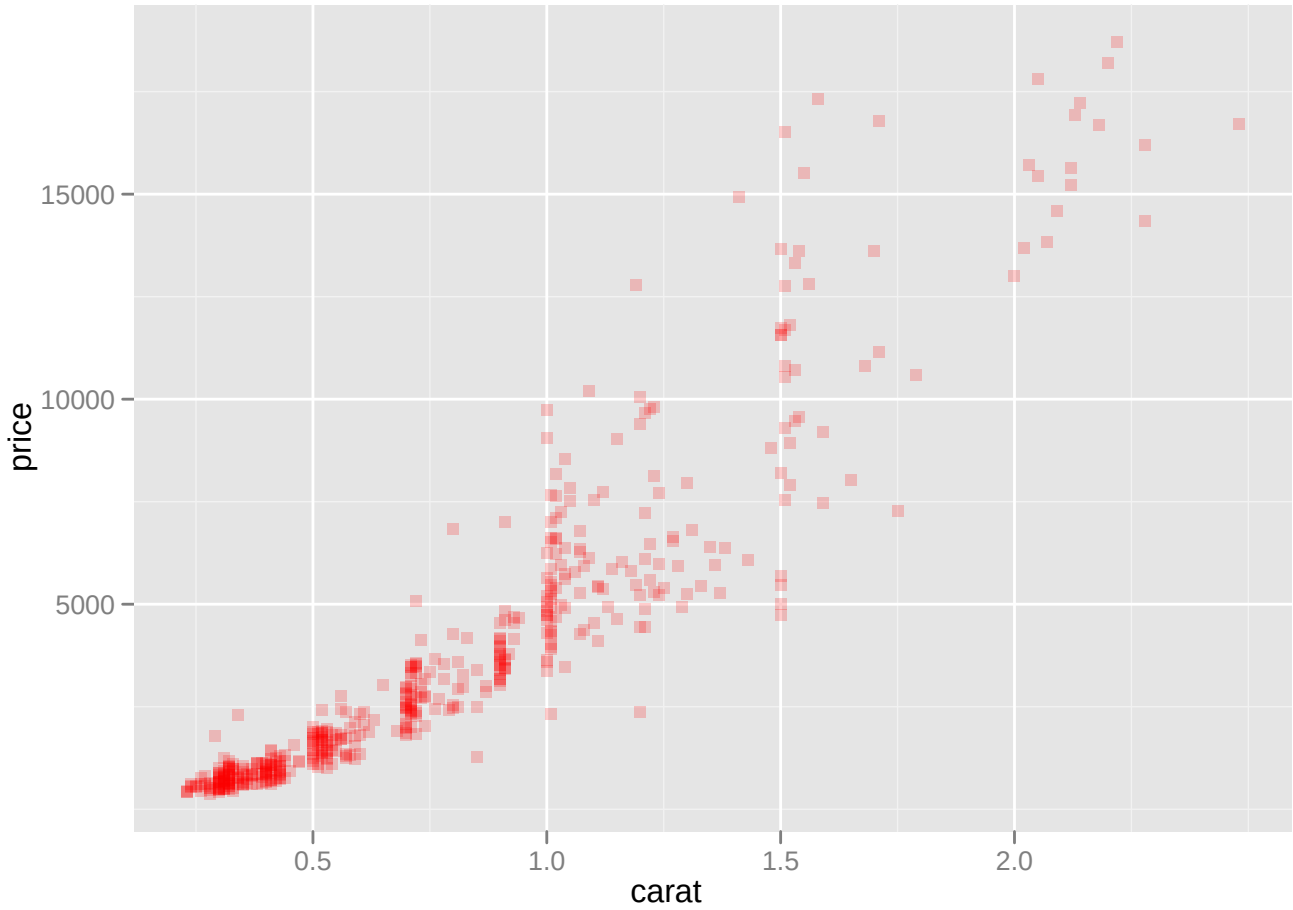
Par défaut, `qplot` retourne un graphique de dispersion (scatterplot), tout comme `plot()`.

```
qplot(carat, price, data = diamonds)
```



qplot : attributs esthétiques

```
qplot(carat, price, data = diamonds, colour = I("red"),
    size = I(2), shape = I(15), alpha = I(1/5))
```

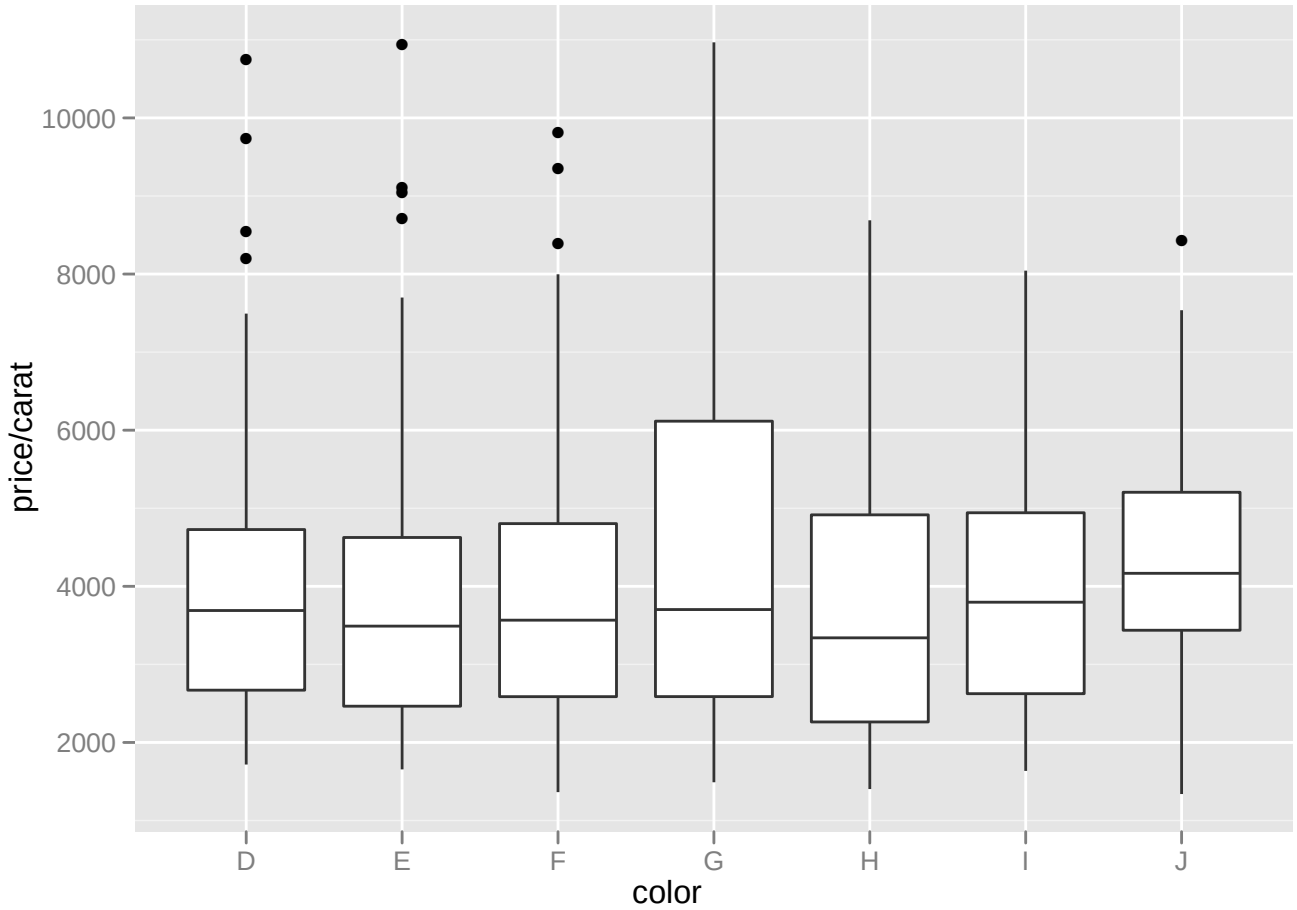


qplot : objets géométriques (geom)

point	$\rightarrow$	scatterplot (défaut si 2 dimensions)
smooth	$\rightarrow$	ajoute une courbe de tendance (lissage de Ratfor ou autre)
boxplot	$\rightarrow$	boîte à moustaches
jitter	$\rightarrow$	gigue
path	$\rightarrow$	chemins entre des points ($\forall$ direction)
line	$\rightarrow$	lignes entre des points (de gauche à droite)
histogram	$\rightarrow$	histogramme (défaut si 1 dimension)
freqpoly	$\rightarrow$	polygone fréquentiel
density	$\rightarrow$	courbe de densité
bar	$\rightarrow$	bâtons

qplot : objets géométriques (geom)

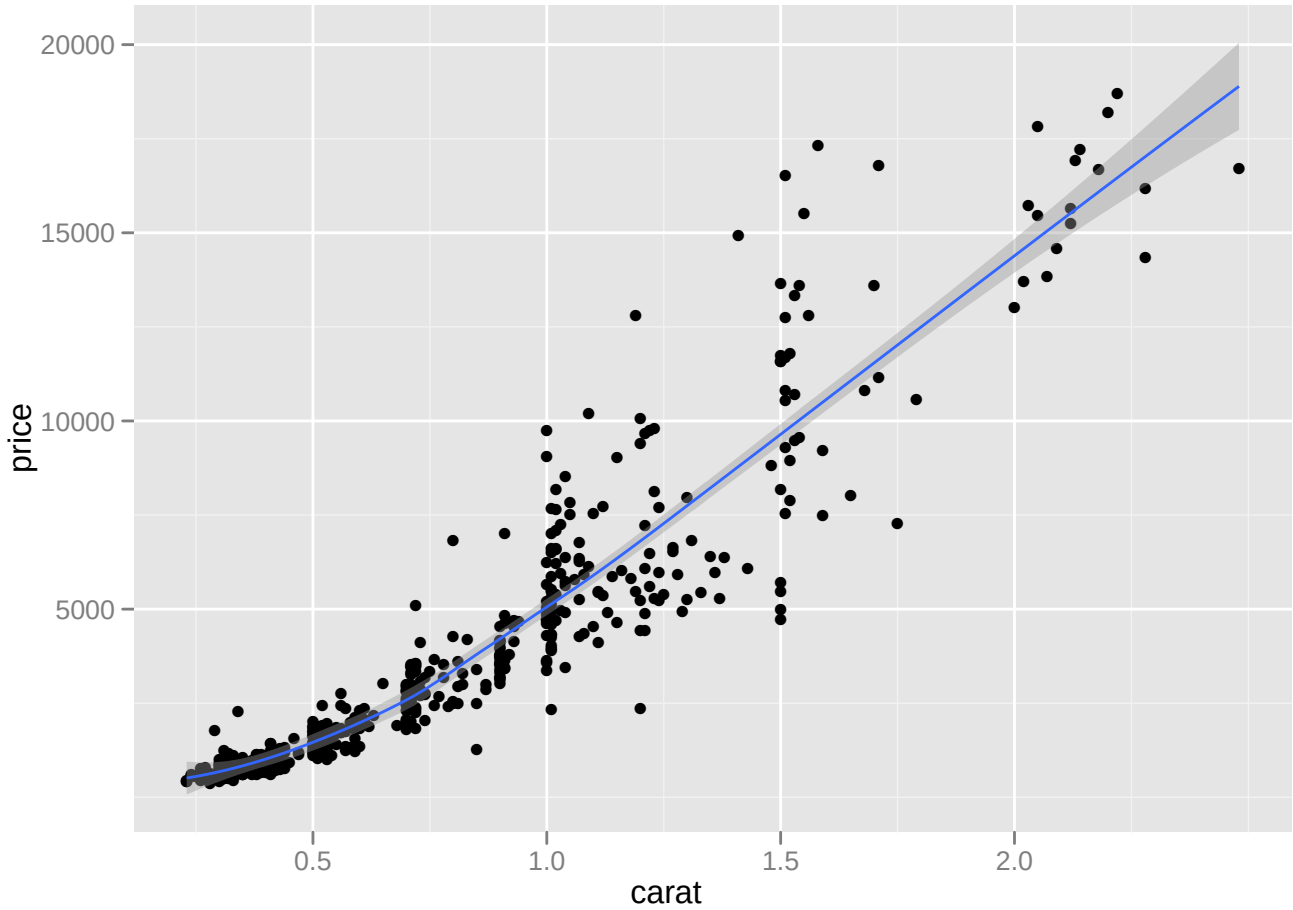
```
qplot(color, price/carat, data = diamonds, geom = "boxplot")
```



qplot : objets géométriques (geom)

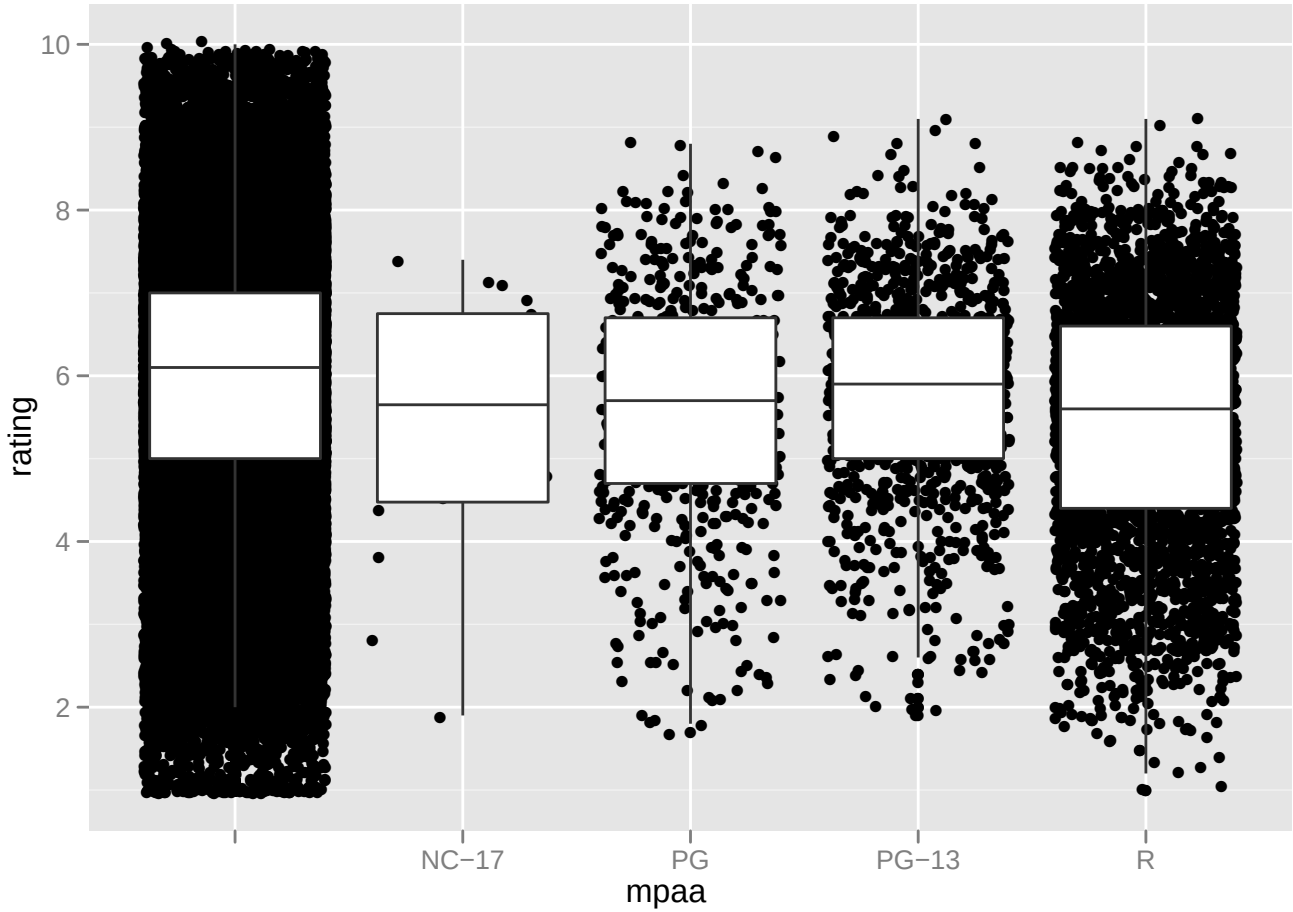
Les différents objets géométriques peuvent être combinés avec `c()` :

```
qplot(carat, price, data = diamonds, geom = c("point",
    "smooth"))
```



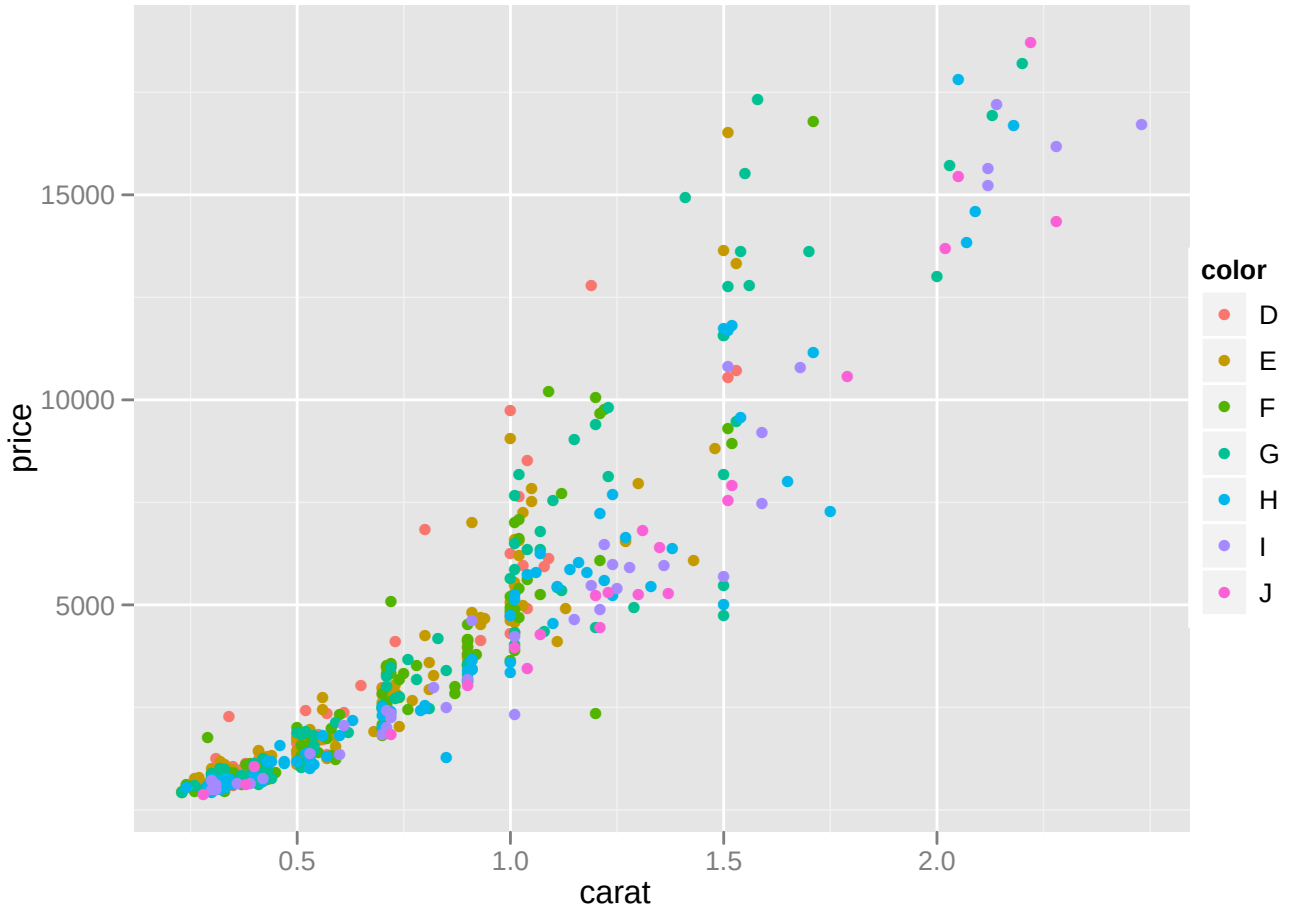
qplot : objets géométriques (geom)

```
data(movies)
qplot(mpaa, rating, data = movies, geom = c("jitter",
    "boxplot"))
```



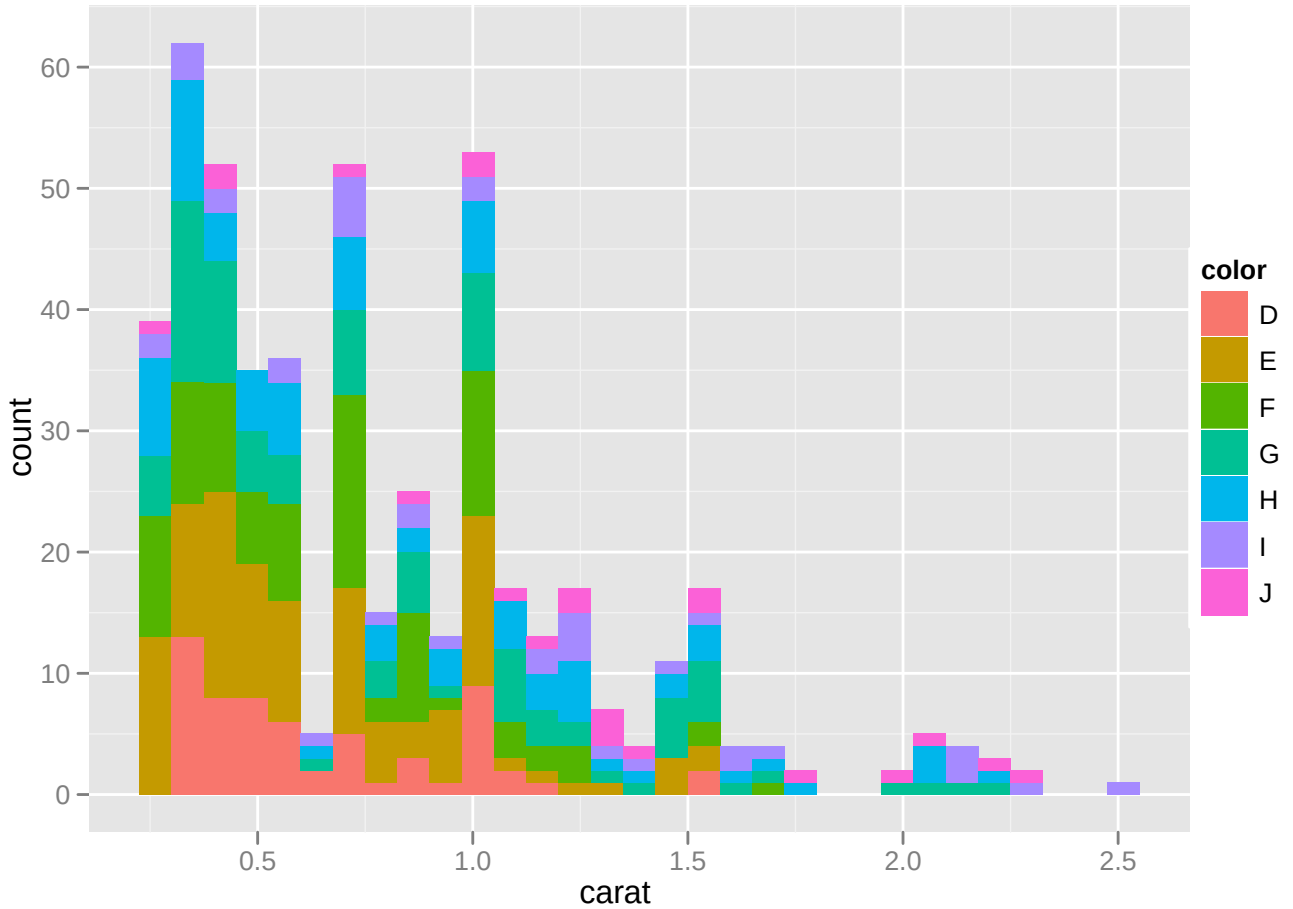
qplot : catégorisation par la couleur

```
qplot(carat, price, data = diamonds, colour = color)
```



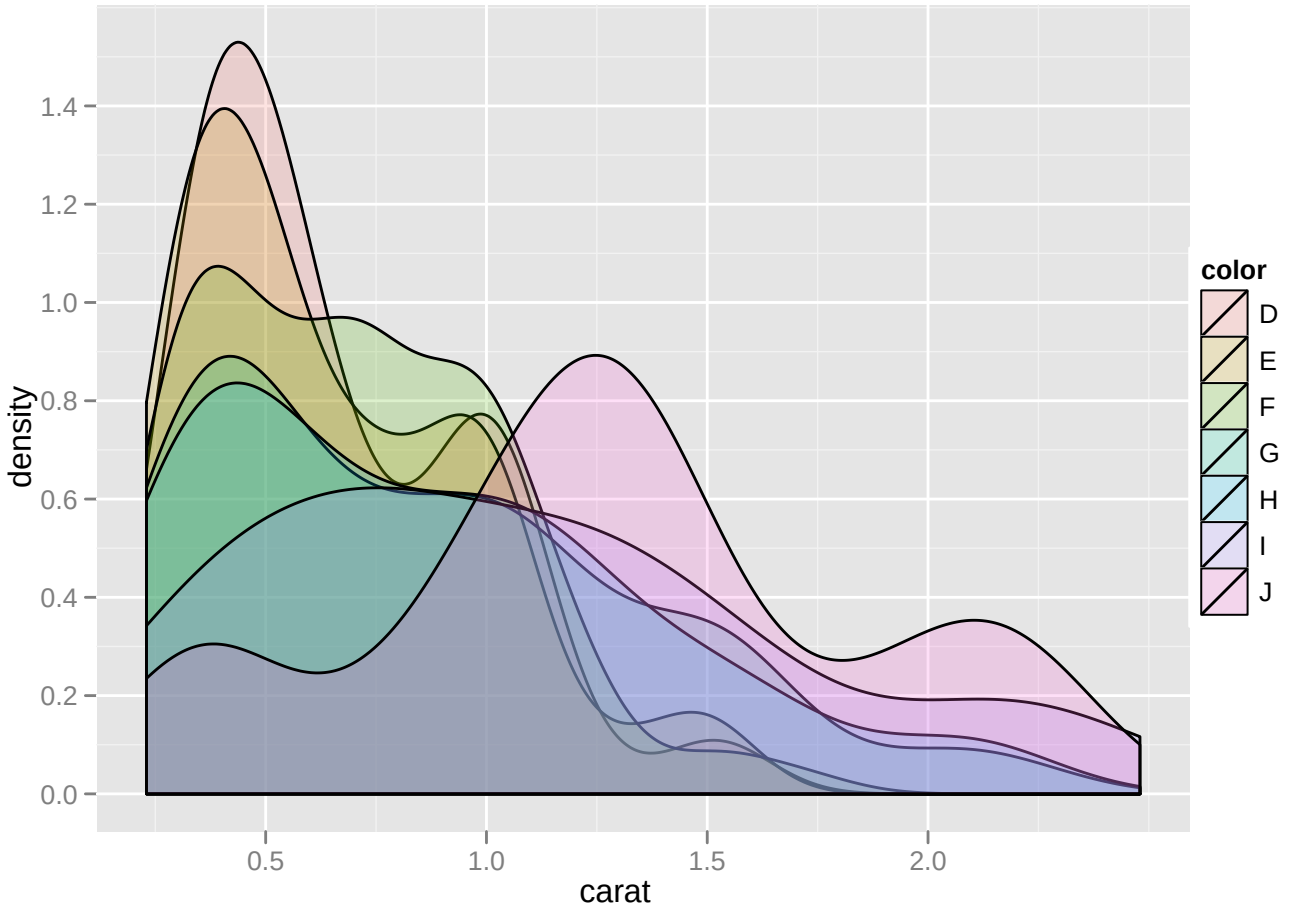
qplot : catégorisation par la couleur

```
qplot(carat, data = diamonds, geom = "histogram", fill = color)
```



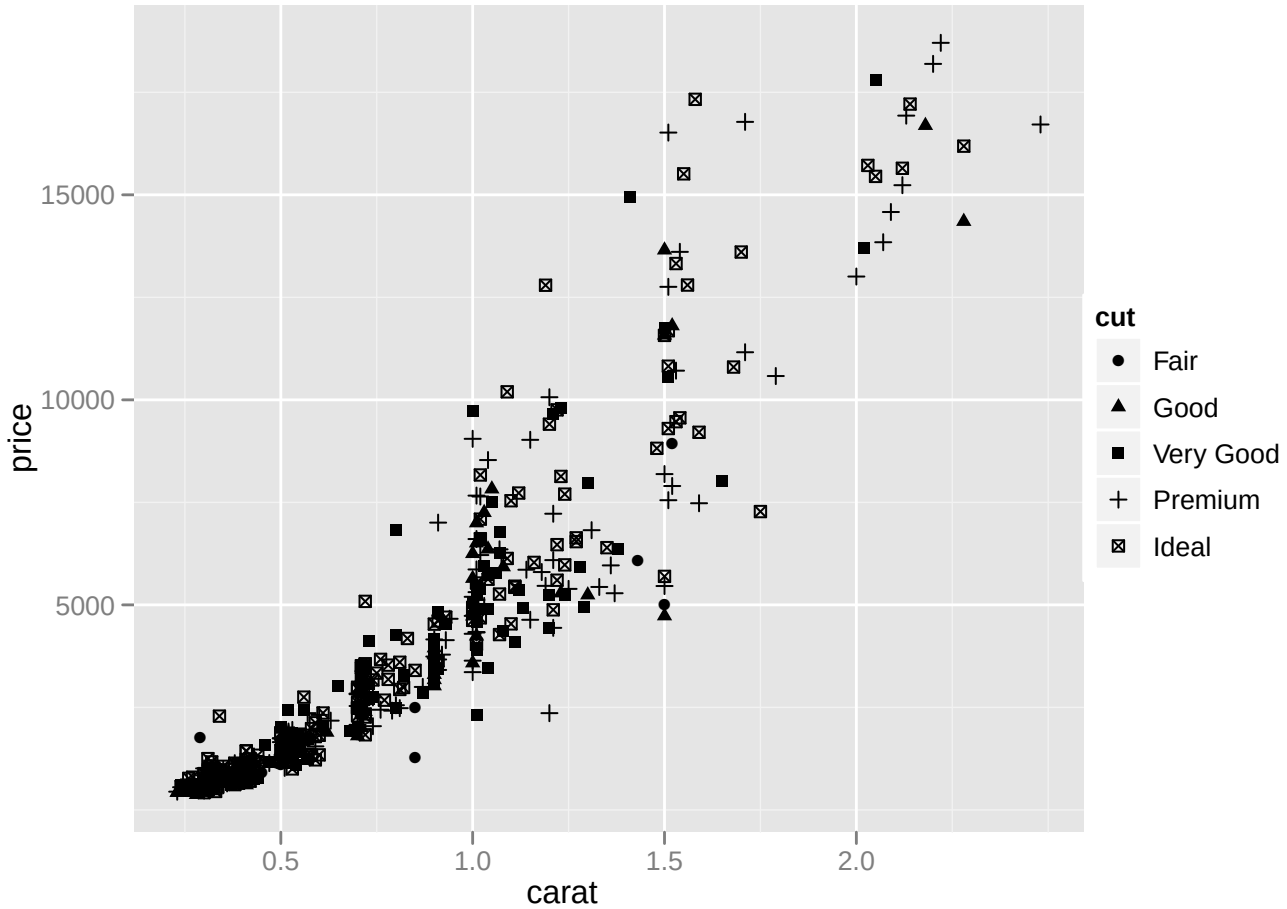
qplot : catégorisation par la couleur

```
qplot(carat, data = diamonds, geom = "density", fill = color,
    alpha = I(1/5))
```



qplot : catégorisation par la forme

```
qplot(carat, price, data = diamonds, shape = cut)
```

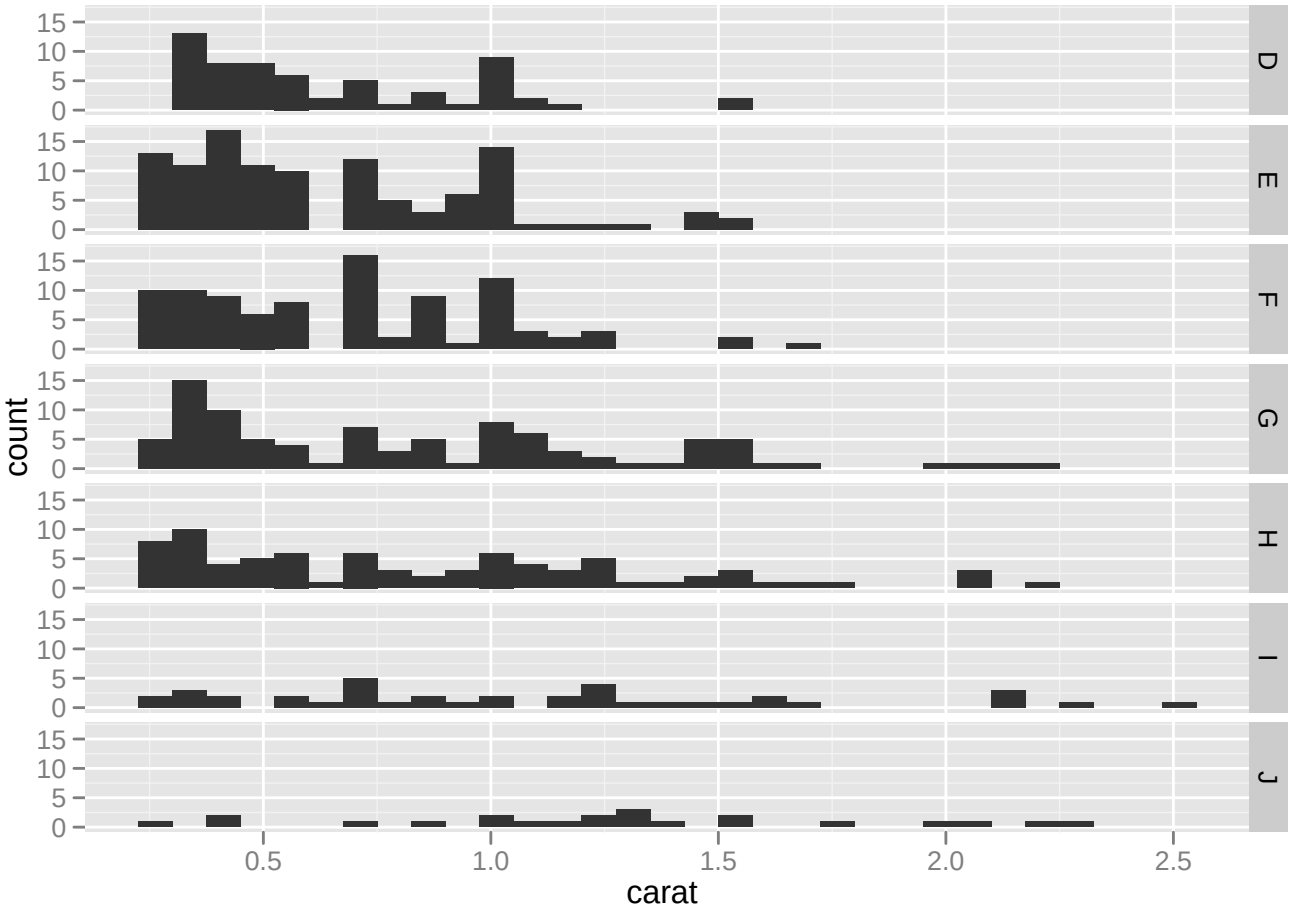


qplot : sub-division

- **Principe** : diviser le jeu de données en sous-ensembles et créer un graphique similaire pour chacun sous-ensemble.
- **Syntaxe** : argument `facets` de `qplot` selon facteur en ligne $\sim$ facteur en colonne
  - `facets = facteur ~ .` produira un graphique multiple en colonnes
  - `facets = . ~ facteur` produira un graphique multiple en lignes

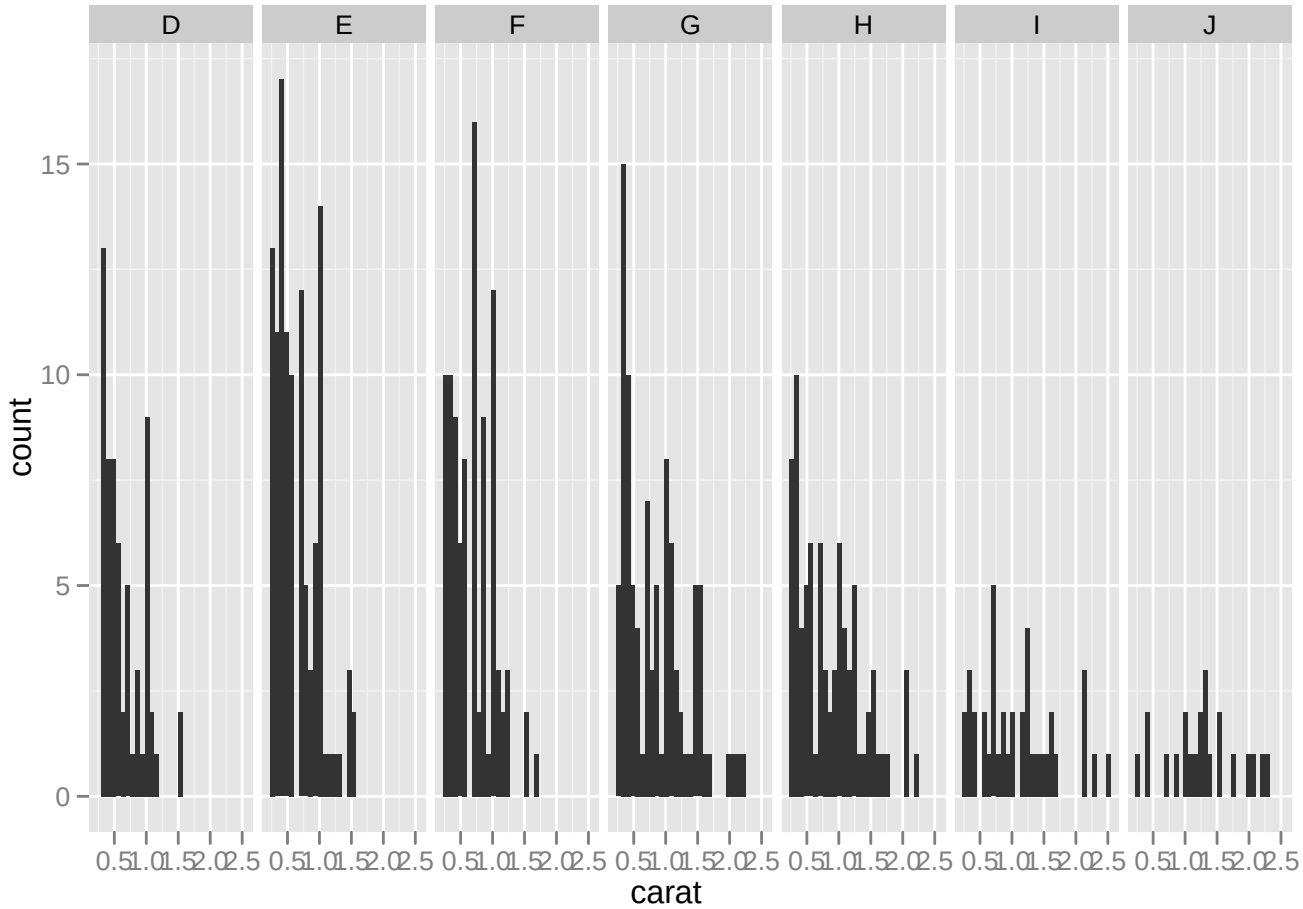
qplot : sub-division

```
qplot(carat, data = diamonds, facets = color ~ ., geom = "histogram")
```



qplot : sub-division

```
qplot(carat, data = diamonds, facets = . ~ color, geom = "histogram")
```



ggplot : arguments

`ggplot` permet, en association avec de produire des graphes plus complexes.

`ggplot` a deux arguments

- **data** (données dans un `data.frame`)
- **aesthetic** mapping (paramètres esthétiques de la projection, doivent être inclus dans la fonction `aes()`)

ggplot : premier calque

```
p <- ggplot(mpg, aes(displ, hwy, colour = factor(cyl)))
summary(p)
```

```
data: manufacturer, model, displ, year, cyl, trans, drv, cty,
  hwy, fl, class [234x11]
mapping:  x = displ, y = hwy, colour = factor(cyl)
faceting: facet_grid(. ~ ., FALSE)
```

ggplot : calques suivants

L'objet `p` ne produit rien. Il faut ajouter des `layers` avec un `+`.

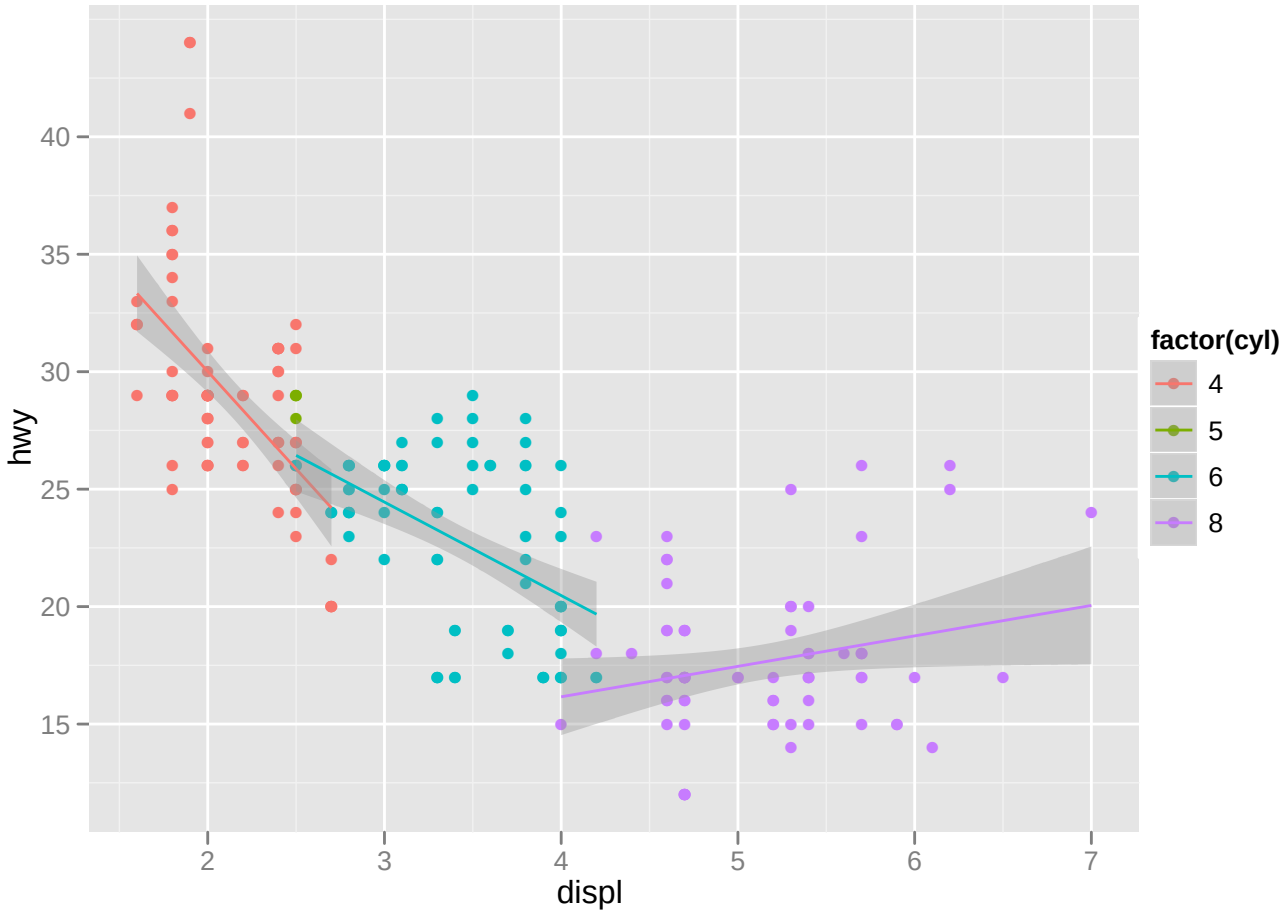
Voici les principaux `layers` :

- `geom_XXX()` (par example : `geom_point()`, `geom_histogramm()`, etc.)
- `stat_XXX()` (par exemple : `stat_bin()`, `stat_smooth()`, etc.)
- `scale_XXX()` (par exemple : `scale_x_continuous()`, `scale_x_log10()`, etc.)
- `facet_XXX()` (par exemple : `facet_grid()`, `facet_wrap()`, etc.)
- `coord_XXX()` (par exemple : `coord_flip()`, `coord_trans()`, etc.)

ggplot : exemple d'objet géométrique

Essayer les commandes suivantes :

```
p + geom_point() + geom_smooth(method = "lm")
```



ggplot : aes

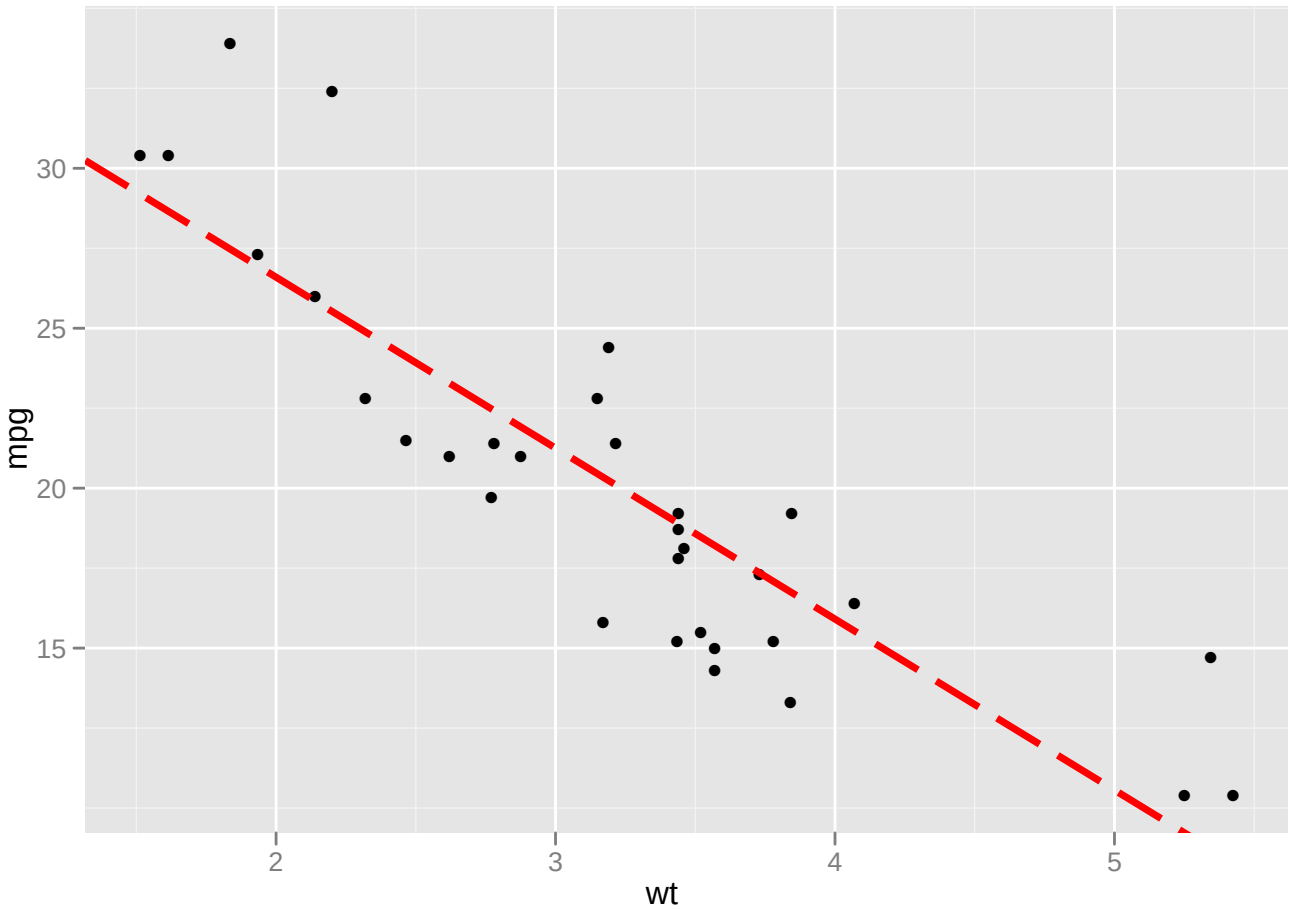
A chaque calque géométrique `geom_XXX` correspond des paramètres esthétiques de la fonction `aes()` .

Par exemple, à `geom_abline` correspondent les arguments `colour`, `size`, `linétype` et `alpha`. Voir la section *Aesthetics* de l'aide de `geom_line`.

```
data(mtcars)
coef <- coef(lm(mpg ~ wt, data = mtcars))
```

ggplot : aes

```
p <- ggplot(mtcars, aes(wt, mpg))
p + geom_point() + geom_abline(intercept = coef[1], slope = coef[2],
    colour = "red", size = 1.25, linetype = 5)
```

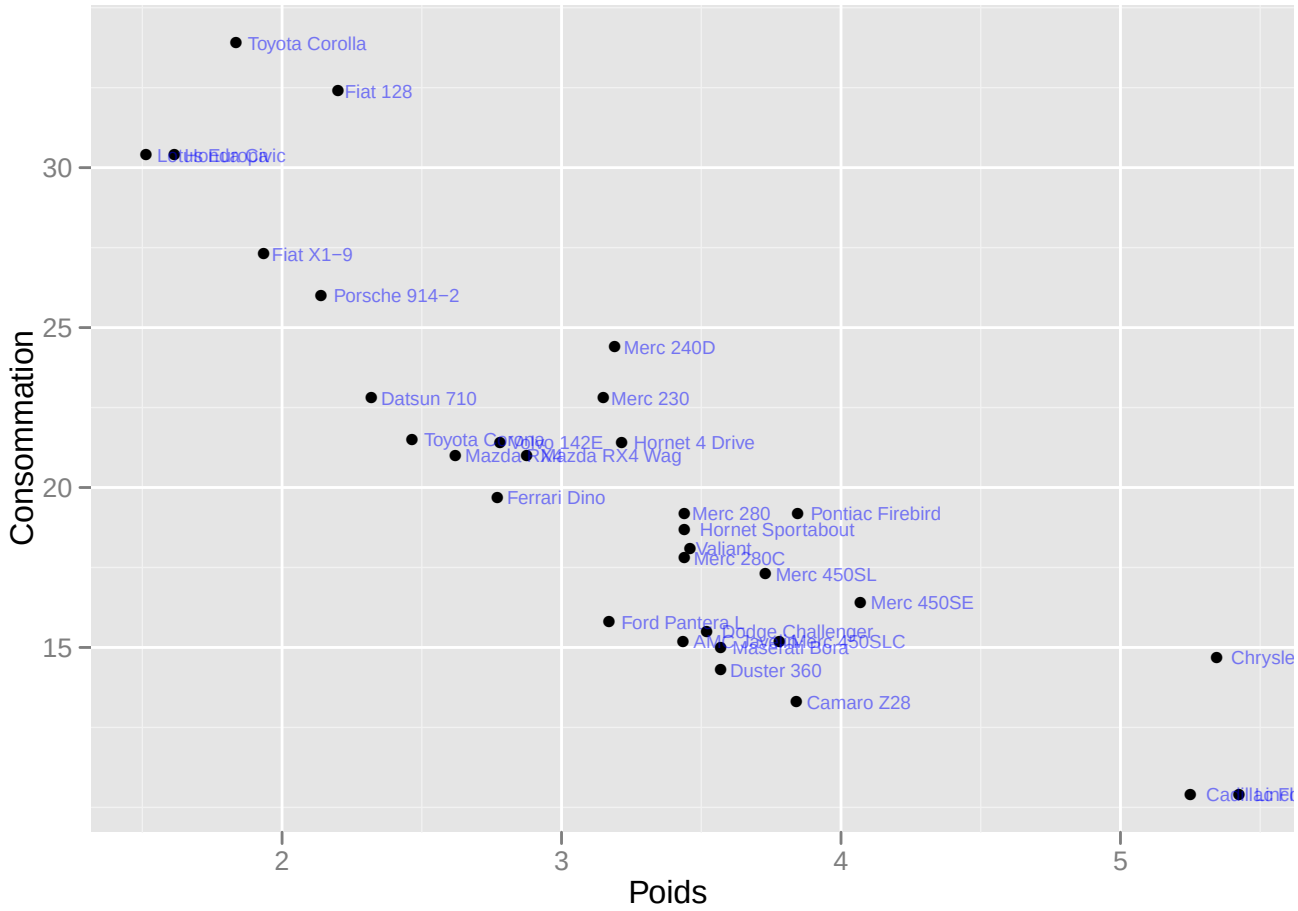


Annotations : `layers`

- `geom_vline()`, `geom_hline()` → lignes verticales et horizontales
- `geom_rect()` → rectangles
- `geom_text()` → texte
- `geom_point()` → point
- `geom_line()`, `geom_path()`, `geom_segment()` → lignes, flèches avec la fonction `arrow()`
- `xlim()`, `ylim()` → limites des axes
- `xlab()`, `ylab()`, `labs` → étiquettes
- `annotate()` → annotations diverses

Annotations : exemples

```
p <- ggplot(mtcars, aes(x = wt, y = mpg)) + geom_point()
p + geom_text(aes(wt, mpg, label = rownames(mtcars)),
    hjust = -0.1, size = 2.5, colour = alpha("blue",
        0.5)) + xlab("Poids") + ylab("Consommation")
```



Position : possibilités

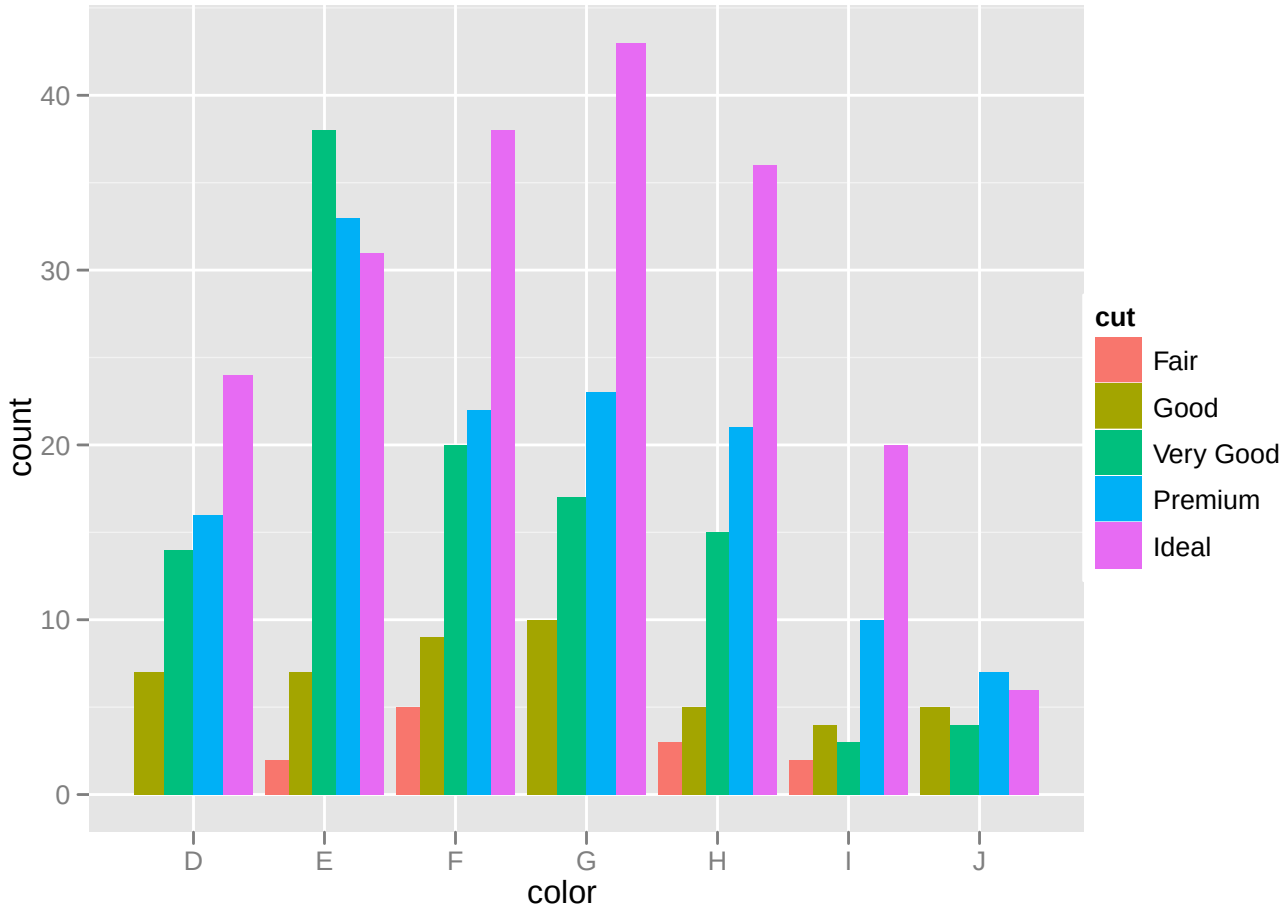
L'argument `position` des `layers` adaptés aux données discrètes permet de modifier la position des éléments graphiques.

Il y a 5 types d'ajustements

- **dodge** : place les éléments côte-à-côte
- **stack** : superpose
- **fill** : superpose et normalise à 1
- **identity** : pas d'ajustement (défaut)
- **jitter** : décale légèrement (pour les points)

Position : exemples

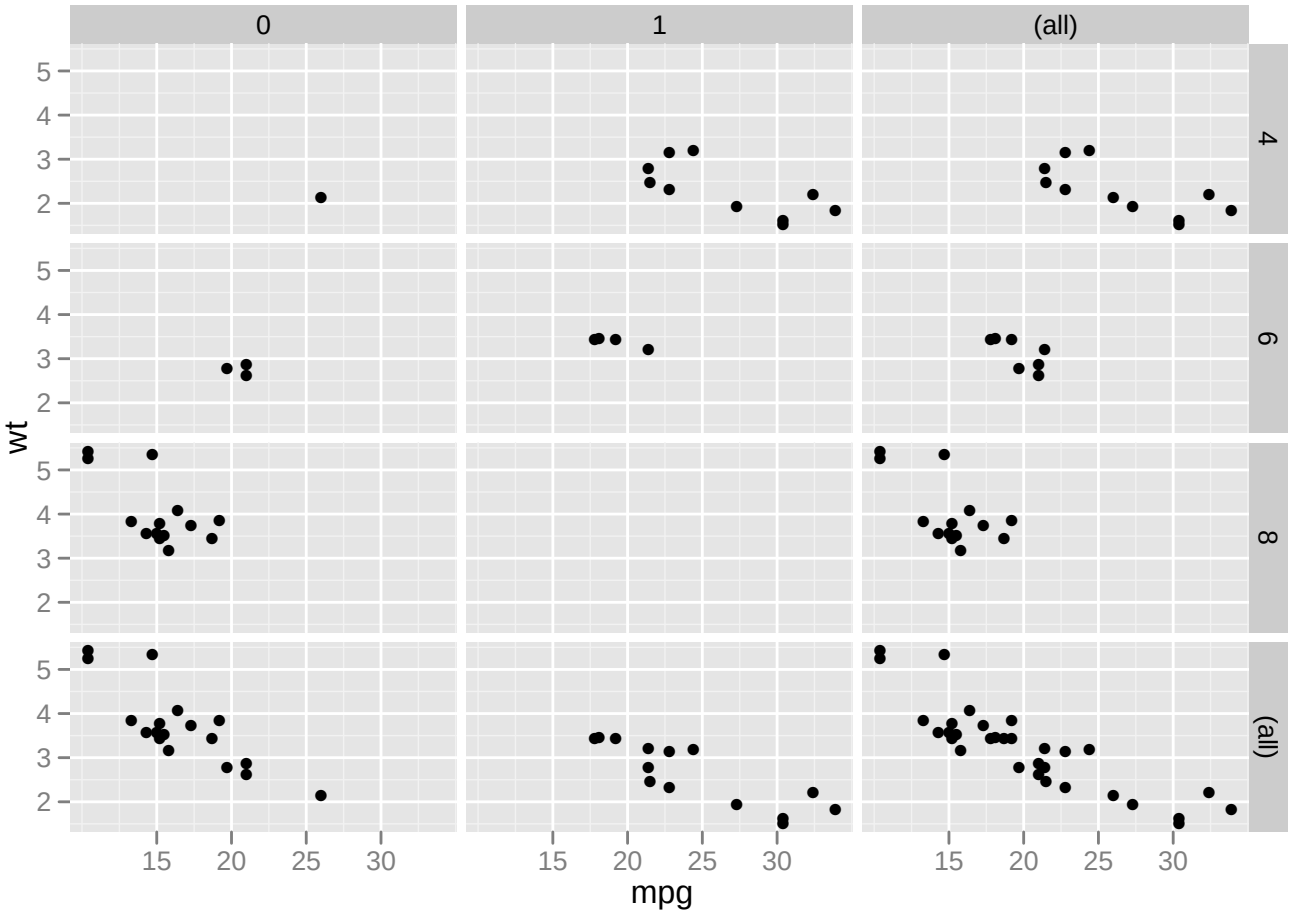
```
ggplot(diamonds, aes(x = color, fill = cut)) + geom_bar(position = "dodge")
```



Subdivision : layers

Pour mieux contrôler les graphiques multiples, il faut utiliser `facet_grid`.

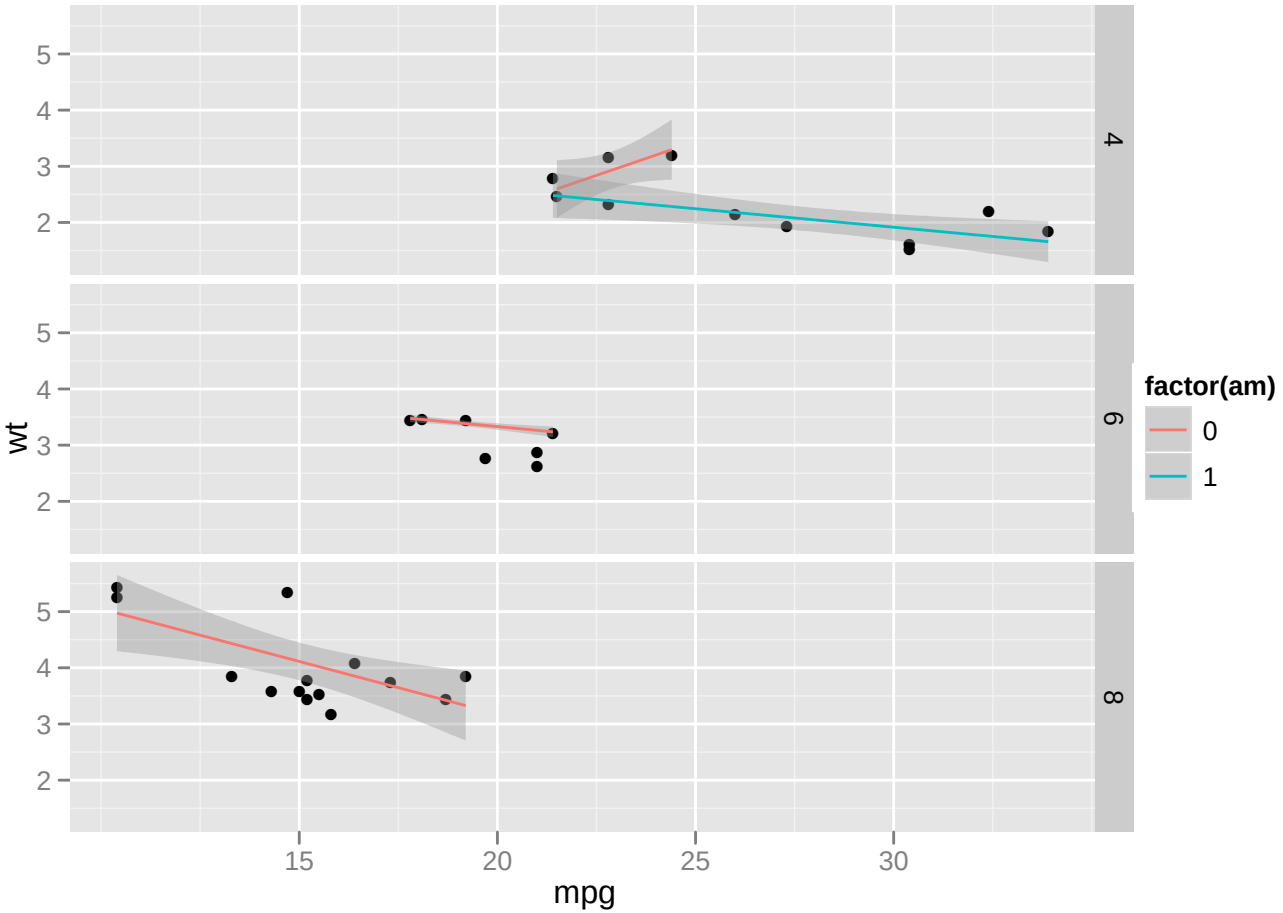
```
qplot(mpg, wt, data = mtcars) + facet_grid(cyl ~ vs,
    margins = TRUE)
```



Subdivision : exemple

Pour ajouter les droites de régression avec selon le facteur `am` (type de boîte de vitesses) :

```
qplot(mpg, wt, data = mtcars) + geom_smooth(aes(colour = factor(am)),
    method = "lm") + facet_grid(cyl ~ .)
```



Références

- Pageweb : http ://had.co.nz/ggplot2/
- Groupe de discussion : http ://groups.google.com/group/ggplot2
- Wickman, H. 2009 – *ggplot2. Elegant graphics for data analysis*. Springer, 212p.
- CRAN : http ://cran.r-project.org/web/packages/ggplot2/index.html