

# R for bioinformatics, data visualisation

## HUST Bioinformatics course series

Wei-Hua Chen (CC BY-NC 2.0)

16 August, 2019

# section 1: TOC

# 前情提要

## iterations 与并行计算

- for loop
- apply functions
- dplyr 的本质是遍历
- map functions in purrr package
- 遍历与并行计算

## 相关包

- purrr
- parallel
- foreach
- iterators

# 本次提要

- basic plot functions
- basic ggplot2
- special letters
- equations
- advanced ggplot2

## section 2: basic plot functions using R

# R basic plot functions

过去几节课我们已经使用了 R basic plot 和 ggplot2 的一些绘画功能，比如讲 factor 时。今次我们进行系统的介绍。

基础做图由 plot 提供。先看示例。这里我们使用系统自带的 swiss 数据，它包含了 47 个法语地区的一些社会经济指标。

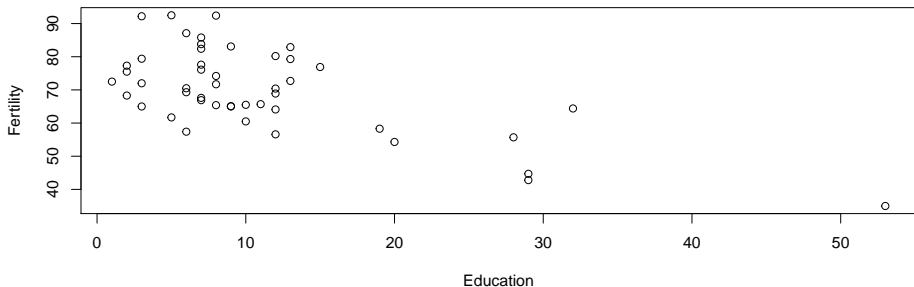
```
head(swiss);
```

```
##           Fertility Agriculture Examination Education Catholic
## Courtelary      80.2          17.0           15          12      9.96
## Delemont        83.1          45.1            6           9     84.84
## Franches-Mnt    92.5          39.7            5           5     93.40
## Moutier         85.8          36.5           12           7     33.77
## Neuveville      76.9          43.5           17          15      5.16
## Porrentruy      76.1          35.3            9           7     90.57
##
##           Infant.Mortality
## Courtelary                22.2
## Delemont                  22.2
## Franches-Mnt              20.2
## Moutier                   20.3
## Neuveville                20.6
## Porrentruy                26.6
```

# 散点图 (dot plot)

我们看一下教育与生育率的关系：

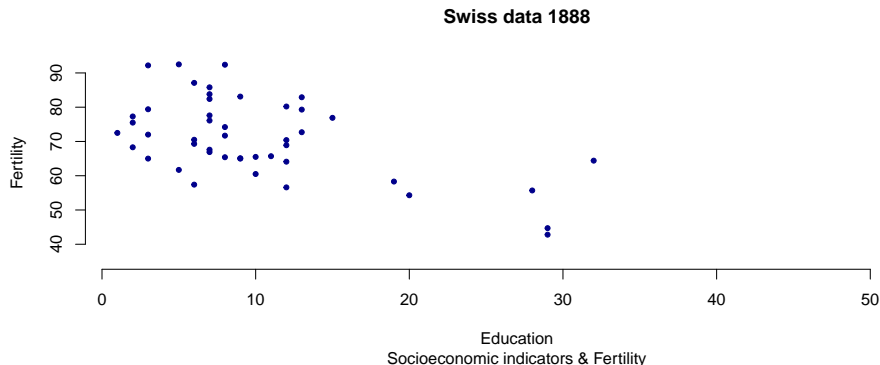
```
with( swiss, plot( Education, Fertility ) );
```



注意 with 的作用是什么 ??

# plot 的参数初探: 先看示例

```
with( swiss, plot(Education, Fertility, type = "p", main = "Swiss data 1888",
  sub = "Socioeconomic indicators & Fertility",
  xlab = "Education", ylab = "Fertility", col = "darkblue",
  xlim = range( Education ), ylim = range( Fertility ),
  pch = 20, frame.plot = F ) );
```





# plot 参数, an annotated example

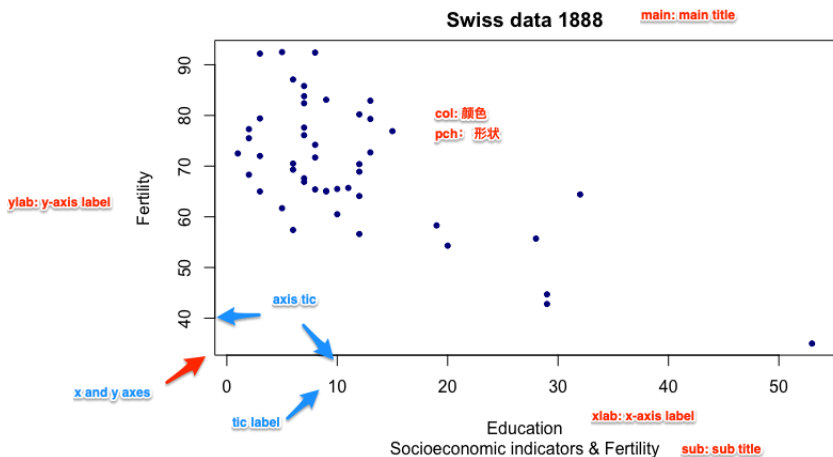
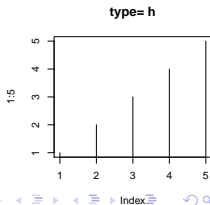
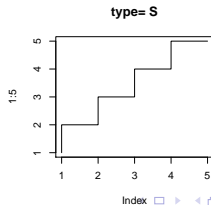
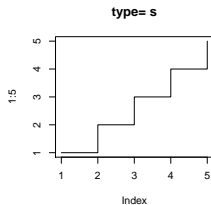
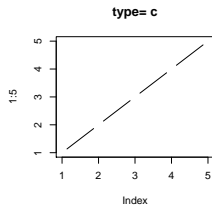
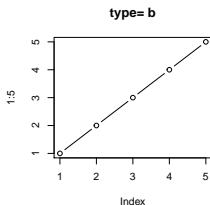
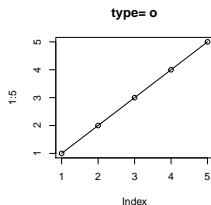
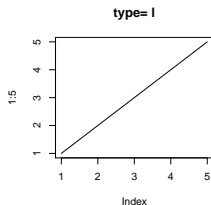
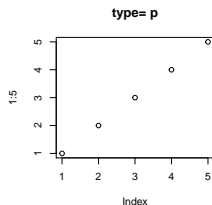


Figure 1: an annotated example

# plot 支持的画图类型，参数 `p = '?'` 的取值

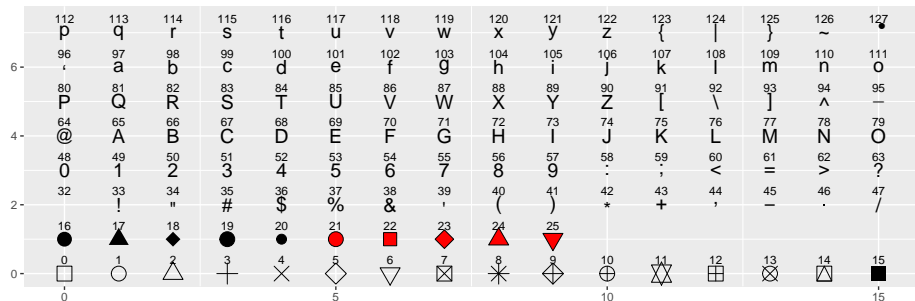
```
par( mfrow = c(2,4) ); ### 在一张图上画 2 x 4 个 panel
opts <- c( "p", "l", "o", "b", "c", "s", "S", "h" );
for( o in opts ){
  plot(1:5, type = o, main = paste( "type=", o ) );
}
```



# pch 是什么？

决定了数据点的形状，注意它的取值范围

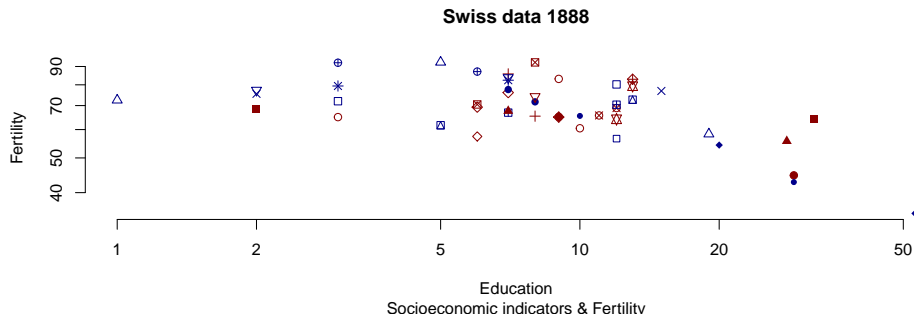
```
library(tidyverse);
ggplot( data.frame( p = c(0:25, 32:127) ) ) +
  scale_y_continuous( name = "" ) + scale_x_continuous( name = "" ) +
  scale_shape_identity() +
  geom_point( aes( x = p%%16, y = p%%16, shape = p ), size = 5, fill = "red" ) +
  geom_text( aes( x = p %% 16, y = p%%16 + 0.4, label = p ), size = 3 );
```



# log transform axes

plot 还有一些其它有用的参数，详见：? plot.default

```
with( swiss, plot(Education, Fertility, type = "p", main = "Swiss data 1888",
  sub = "Socioeconomic indicators & Fertility",
  xlab = "Education", ylab = "Fertility", col = c("darkblue", "darkred"),
  xlim = range( Education ), ylim = range( Fertility ),
  pch = 0:25, frame.plot = F, log = "xy") );
```

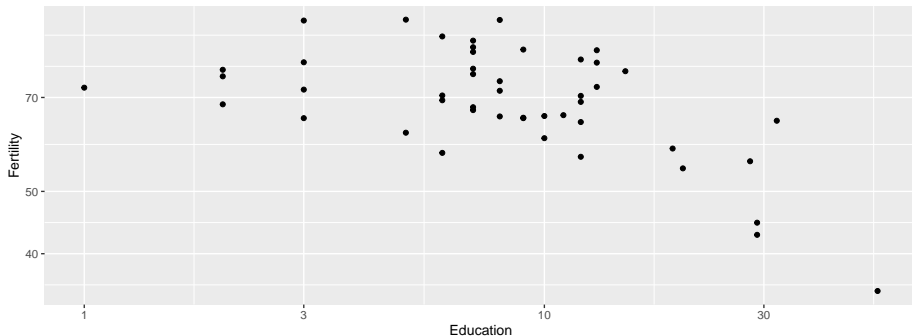


## 注：也可以用 `log='x'` 或 `log='y'` 只对一个 `axis` 进行 `log` 处理

# ggplot 版本

```
ggplot( swiss, aes( x = Education, y = Fertility ) ) +
  geom_point( ) + scale_x_log10() + scale_y_log10() +
  xlab( "Education" ) + ylab( "Fertility" ) +
  ggtitle( "Swiss data 1888" );
```

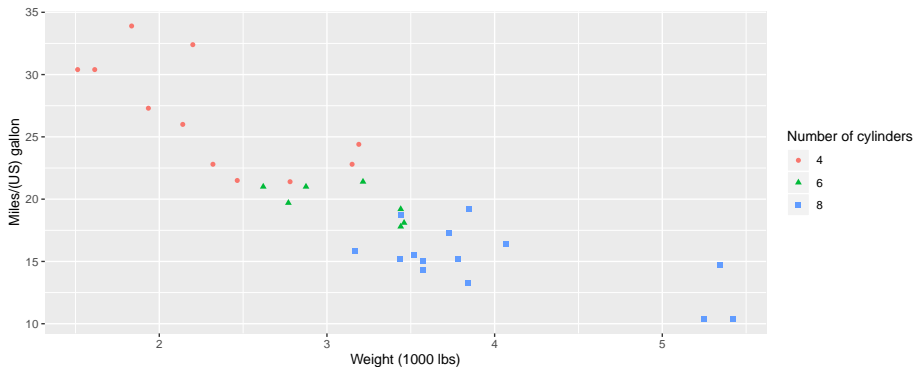
Swiss data 1888



# ggplot 更多散点示例

以 mtcars 为例

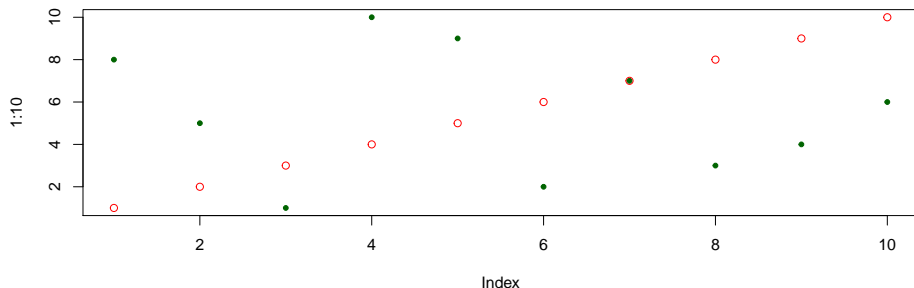
```
ggplot( mtcars, aes( x = wt, y = mpg, colour = factor( cyl ), shape = factor(cyl) ) ) +
  geom_point() + xlab( "Weight (1000 lbs)" ) + ylab( "Miles/(US) gallon" ) +
  labs( colour = "Number of cylinders", shape = "Number of cylinders" ) ;
```



# plot: high-level vs. low-level plots

- **high level:** plotting functions create a new plot on the graphics device
- **low level:** plotting functions add more information to an existing plot

```
plot( 1:10, col = "red" ); ## high level
points( sample(1:10, 10), col = "darkgreen", pch = 20 ); ## low level
```



# low level plots 列表

- `points` : 点图
- `lines` : 线图
- `abline` : 直线
- `polygon` : 多边形
- `legend` : 图例
- `title` : 标题
- `axis` : 轴 ...



# high level plots 列表

- `plot` : 通用画图函数
- `pairs`
- `coplot`
- `qqnorm`
- `hist`
- `dotchart`
- `image`
- `contour` ...

注: 可以用 `add = TRUE` 参数 (如果可用) 将 high level 函数强制转换为 low level

# 图形相关参数（系统函数）

`par()` 函数：显示或修改当前图形设备的参数。用以下命令查看支持的内容：

```
par( c( "mar", "bg" ) ); ## 显示指定参数的值
```

```
## $mar
## [1] 5.1 4.1 4.1 2.1
##
## $bg
## [1] "transparent"
```

```
## 显示所有参数
par();
```

```
## $xlog
## [1] FALSE
##
## $ylog
## [1] FALSE
##
## $adj
## [1] 0.5
##
## $ann
## [1] TRUE
```

## 调整 `par()` 参数前请备份

`par()` 用于指定全局参数，因此在改变前尽量备份

```
oldpar <- par(); ## 备份  
do some changes here ...  
  
## 恢复  
par( oldpar );
```

## 常用图形参数及调整: margin

图形边距 (figure margins )

```
par( mar = c( 5.1, 4.1, 4.1, 2.1 )); ## 设置新 margin
```

分别指定下 -> 左 -> 上 -> 右的边距，即从下面开始，顺时针移动。

单位是：text lines

或：

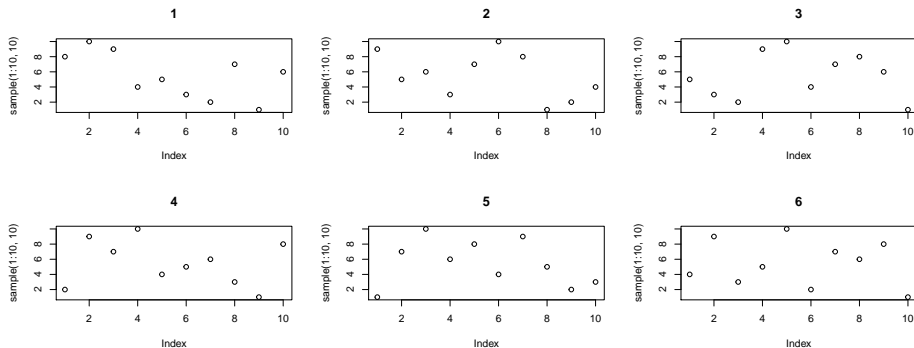
```
par( mai = c( 5.1, 4.1, 4.1, 2.1 )); ## 设置新 margin
```

单位是：inch

# 常用图形参数及调整: 多 panel

画 2x3 共 6 个 panel, 从左到右。(2 行 3 列)

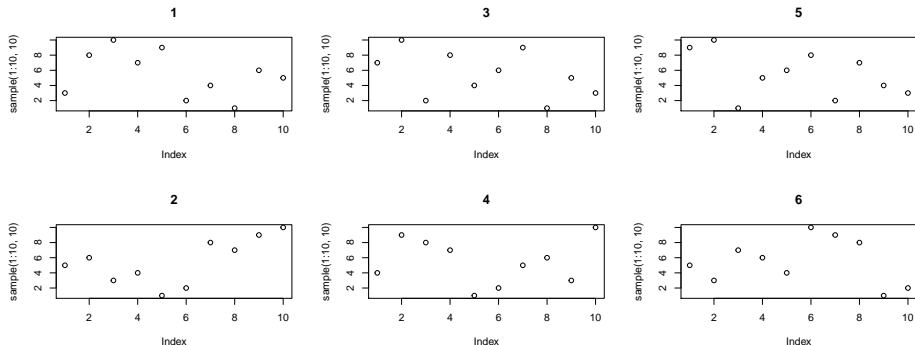
```
par( mfrow=c(2,3) );
for( i in 1:6 )
  plot( sample( 1:10, 10 ), main = i );
```



# 常用图形参数及调整: 多 panel , cont.

画 2x3 共 6 个 panel, 从上到下。(2 行 3 列)

```
par( mfcol=c(2,3) );
for( i in 1:6 )
  plot( sample( 1:10, 10 ), main = i );
```



## 重要概念：图形设备

图形设备是指图形输出的设备，可以将图形设备理解为保存格式。

默认设备是：

- `X11()` : \*nix
- `windows()` : windows
- `quartz()` : OS X

图形显示在显示器上。

## 图形设备: cont.

常用其它设备有:

- `pdf()`
- `png()`
- `jpeg()`

分别对应输出文件格式。



# 常用图形设备：pdf()

使用方法如下：

```
pdf( file = "/path/to/dir/<file_name>.pdf", height = 5, width = 5 ); ## 创建一个新设备/ pdf 文件
plot(1:10); ## 作图;
dev.off(); ## 关闭设备
```

## 说明

- ❶ 默认文件名为 Rplots.pdf ,
- ❷ dev.off() 必须关闭。关闭后，返回到最近使用的图形设备
- ❸ height 和 width 参数的单位是 inch
- ❹ 如果运行多个 high level 作图命令，则会产生多页 pdf

# 请尽量使用 pdf 作为文件输出格式

- ① 生信图片大多是点线图，适合保存为矢量格式（如 pdf, ps 等）；
- ② 矢量图可无限放大而不失真（变成像素）；
- ③ 可由 Adobe Illustrator 等矢量图软件进行编辑

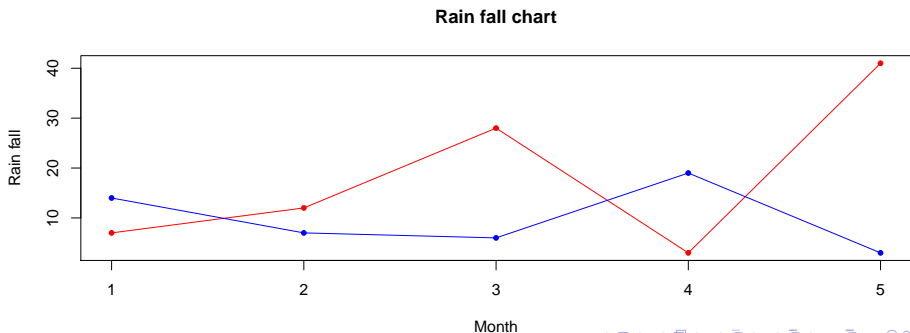
## section 3: ggplot2 基础

# 为什么要使用 ggplot2 ? 从一个简单示例开始

假设画两条线:

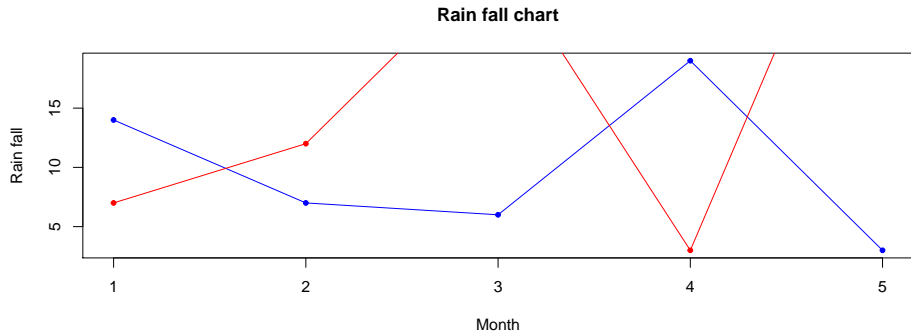
```
year1 <- c(7, 12, 28, 3, 41);
year2 <- c(14, 7, 6, 19, 3);

plot( year1, type = "o", pch = 20, col = "red", xlab = "Month", ylab = "Rain fall",
      main = "Rain fall chart");
lines( year2, type = "o", pch = 20, col = "blue");
```



## 如果改变画线的顺序？

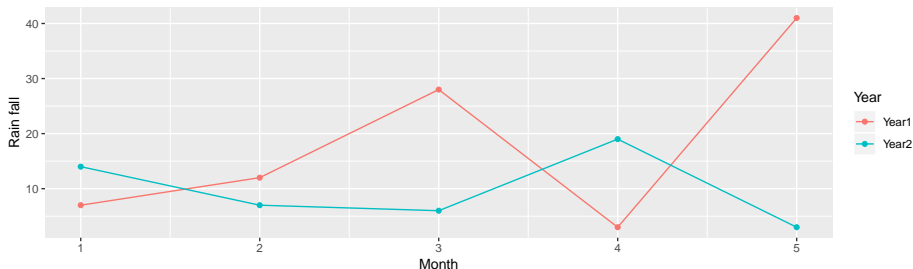
```
plot( year2, type = "o", pch = 20, col = "blue", xlab = "Month", ylab = "Rain fall",  
      main = "Rain fall chart");  
lines( year1, type = "o", pch = 20, col = "red");
```



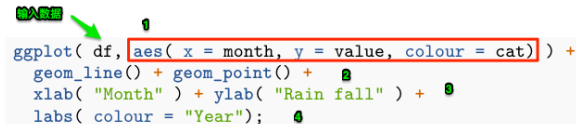
# ggplot2 的方法

```
df <- rbind( tibble( month = 1:length( year1 ), value = year1, cat = "Year1" ),
             tibble( month = 1:length( year2 ), value = year2, cat = "Year2" ) );
```

```
plot1 <- ## 将图保存在变量中;
  ggplot( df, aes( x = month, y = value, colour = cat ) ) +
  geom_line() + geom_point() +
  xlab( "Month" ) + ylab( "Rain fall" ) +
  labs( colour = "Year" );
plot1; ## 画图
```



# ggplot2 基础概念详解



```
ggplot( df, aes( x = month, y = value, colour = cat) ) +
  geom_line() + geom_point() +
  xlab( "Month" ) + ylab( "Rain fall" ) +
  labs( colour = "Year" );
```

Figure 2: ggplot2 参数简介

- ① aes (aesthetics) 美学：控制全局参数，包括：x,y 轴使用的数据，颜色 ( colour, fill )，形状 ( shape )，大小 ( size )，分组 ( group ) 等等；
- ② 图层：geom\_<layer\_name> ； 每张图可有多多个图层（此处有两个）；  
 图层可使用全局数据 (df) 和参数 (aes)，也可以使用自己的 aes 和数据；3-4. 其它参数

# ggplot2 优缺点

ggplot2 优点：

- ① 强大又专业
- ② 复杂又好看
- ③ canvas 大小，坐标会根据数据、图层自动调整，让用户专注于作图本身；

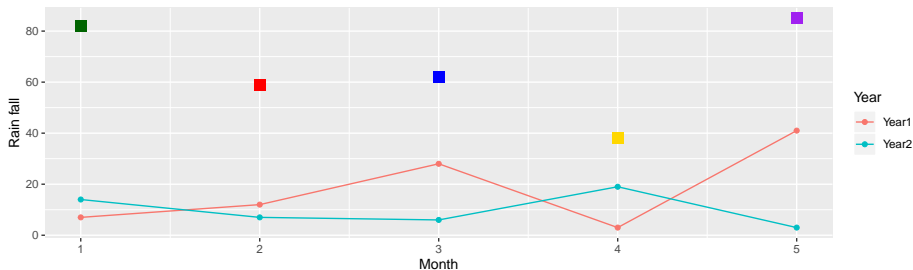
缺点：

太难学！



# 图层使用自己的数据， 示例

```
plot1 +
  geom_point( data = data.frame( x2 = 1:5, y2 = sample(30:100, 5) ), ## 注意: data = 是必须的
    aes( x = x2, y = y2 ), ## 使用自己的 aes ...
    colour = c("darkgreen", "red", "blue", "gold", "purple" ) , shape = 15, size = 4 )
```



## 要点

- ① 如上所见，xy -axes 会随数据自动调整
- ② ggplot2 作图结果可以保存在变量中，并可累加更多图层
- ③ 图层使用自己的数据时，需要用 `data =` 指定；而全局数据则不用 `ggplot` (`data.frame( ... )`)

## aes( ) 内部和外部的 colour, size, shape 参数有何区别？

在内部时, `colour = < 列名 >` 或 `colour = factor(< 列名 >)`, 其真实结果是取的 `factor`, 然后按顺序为每个 `factor` 自动指定一个颜色。默认颜色顺序为:

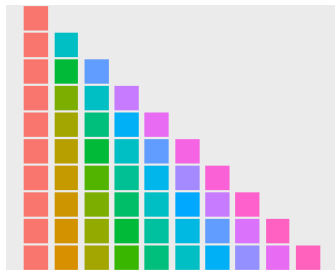
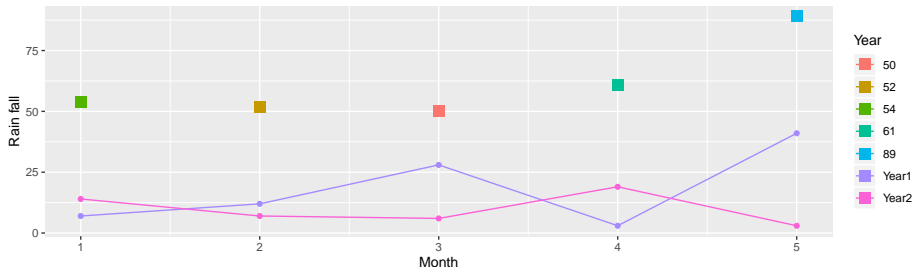


Figure 3: default discrete colour palette

# color 举例

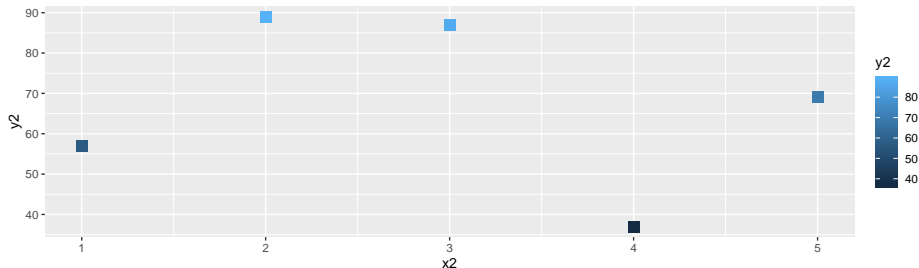
```
plot1 +
  geom_point( data = data.frame( x2 = 1:5, y2 = sample(30:100, 5) ), ## 注意: data = 是必须的
    aes( x = x2, y = y2 , colour = factor( y2 ) ), ## colour 在 aes 内部
    shape = 15, size = 4 )
```



共有 7 个颜色；注意与上页图的第 7 行对应一下！

## 当 colour = < 数字列 > , 则显示 color gradient

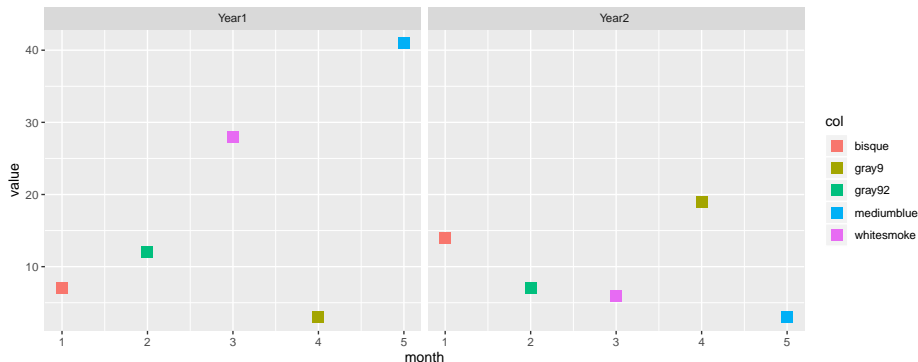
```
ggplot( data = data.frame( x2 = 1:5, y2 = sample(30:100, 5) ),
  aes( x = x2, y = y2 , colour = y2 ) ) +
  geom_point( shape = 15, size = 4 )
```



注意 discrete color (上页图) 和 continous color ( or color gradient ) 的默认画板 (color palette) 是不一样的!

# 更改画板，使用指定的颜色（不作为 factor 使用）

```
df$col <- sample( colours(), 5 ); ## 现有我们有颜色了!
ggplot(df, aes( x = month, y = value, colour = col ) ) +
  geom_point( size = 4, shape = 15 ) + facet_grid( ~ cat );
```

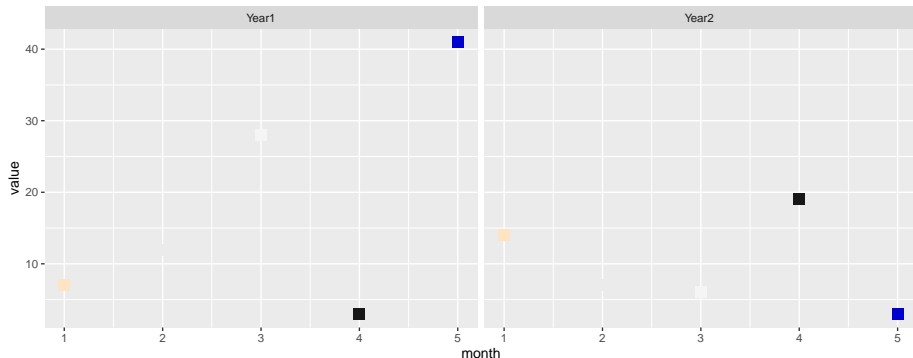


**\*\* 注 \*\*** 默认情况下，col（颜色）列是作为 factor 使用。

# 更改画板，使用指定的颜色（不作为 factor 使用），cont.

解决方案: `scale_color_identity`

```
ggplot(df, aes( x = month, y = value, colour = col ) ) +  
  geom_point( size = 4, shape = 15 ) + facet_grid( ~ cat ) + ## facet_grid 又是什么 ??  
  scale_color_identity(); ## magic !!
```



## 练习：认识并掌握颜色

### ① 认识颜色：

[http://www.cookbook-r.com/Graphs/Colors\\_\(ggplot2\)/](http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/)

### ② ggplot2 画板：https：

[//ggplot2.tidyverse.org/reference/scale\\_brewer.html](https://ggplot2.tidyverse.org/reference/scale_brewer.html)

### ③ 手动调色：https：

[//ggplot2.tidyverse.org/reference/scale\\_manual.html](https://ggplot2.tidyverse.org/reference/scale_manual.html)

### ④ ggplot2 color, best practice: <https://www.datanovia.com/en/blog/ggplot-colors-best-tricks-you-will-love/>

## 图层简介

- `geom_point`, `geom_line`: 点线图, 用于揭示两组数据间的关系;
- `geom_smooth`: 常与 `geom_point` 联合使用, 揭示数据走势
- `geom_bar`: bar 图
- `geom_boxplot`: 箱线图, 用于比较 N 组数据, 揭示区别
- `geom_path`: 与 `geom_line` 相似, 但也可以画其它复杂图形
- `geom_histogram`, “`geom_density`”: 数据的分布, 也可用于多组间的比较
- 其它十余种, 请见 “ggplot2: elegant graphics for data analysis” 一书!!



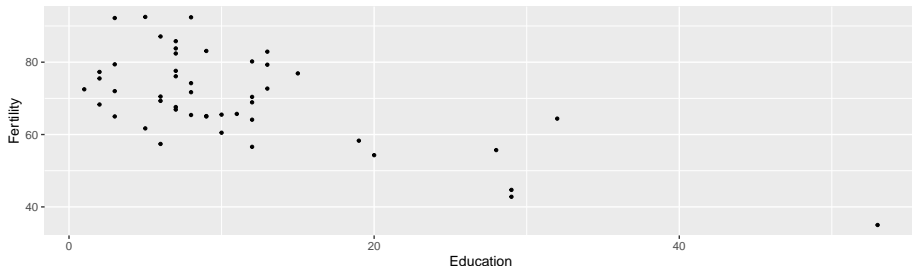
## section 4: ggplot2 进阶 1

# 散点图的进一步分析

显示两组数据间的相关性:

## 作图

```
ggplot( swiss, aes( x = Education, y = Fertility ) ) +  
  geom_point( shape = 20 );
```



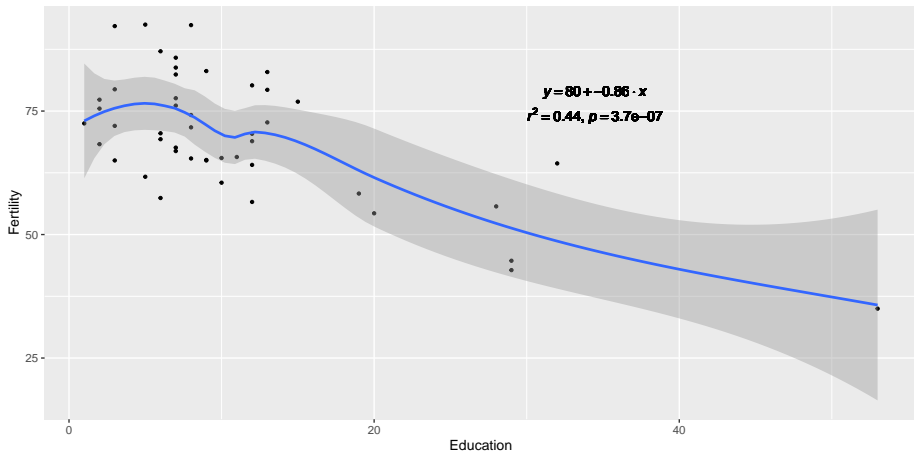
## 分析

```
with( swiss, cor.test( Education, Fertility )$estimate );
```

```
##          cor  
## -0.6637889
```

# 在图中加入公式和统计信息

先展示一下结果



# 公式详解

`italic(y) == a + b %.% italic(x)`

$$y = 80 + -0.86 \cdot x$$

`italic(r)^2 ~ "=" ~ italic(r)^2 * ", ~ italic(p) == pvalue`

等号及两边的空格      逗号, 左边须空格 •      另一种等号的写法 默认两边都是空格

Figure 4: equation explained!

## 注

- 引号两边必须有 \* 或 ~ 字符, ~ 表示空格, \* 表示什么都没有。~~ 表示两个空格。如果公式中需要 ~ 字符怎么办 ?? 见下面“公式示例 3”。

## 公式详解, cont.

以下代码实现两个任务:

- 1 将两个公式上下放置 `atop ( <equation_1> , <equation_2> )`;
- 2 将公式中的某些值替换为数值 `substitute( <equation>, list( ... ) )`

```
## 计算 ...
m = lm(Fertility ~ Education, swiss);
c = cor.test( swiss$Fertility, swiss$Education );

## 生成公式
eq <- substitute( atop( italic(y) == a + b %.* italic(x),
                        italic(r)^2 ~ "=" ~ r2 * ", " ~ italic(p) == pvalue ),
                  list(a = as.vector( format(coef(m)[1], digits = 2) ),
                      b = as.vector( format(coef(m)[2], digits = 2) ),
                      r2 = as.vector( format(summary(m)$r.squared, digits = 2) ),
                      pvalue = as.vector( format( c$p.value , digits = 2) ) )
);

## 用 as.expression 对公式进行转化
eq <- as.character(as.expression(eq));
```

# 完整代码

```
## 计算 ...
m = lm(Fertility ~ Education, swiss);
c = cor.test( swiss$Fertility, swiss$Education );

## 生成公式
eq <- substitute( atop( italic(y) == a + b %.% italic(x),
                        italic(r)^2~"="~r2*", "~italic(p)=pvalue ),
                  list(a = as.vector( format(coef(m)[1], digits = 2) ),
                        b = as.vector( format(coef(m)[2], digits = 2) ),
                        r2 = as.vector( format(summary(m)$r.squared, digits = 2) ),
                        pvalue = as.vector( format( c$p.value , digits = 2) ) )
                  );

## 用 as.expression 对公式进行转化 !!!!
eq <- as.character(as.expression(eq));

## 作图，三个图层；特别是 geom_text 使用自己的 data 和 aes ...
ggplot(swiss, aes( x = Education, y = Fertility ) ) +
  geom_point( shape = 20 ) +
  geom_smooth( se = T ) + ## smooth line ...
  geom_text( data = NULL,
             aes( x = 30, y = 80, label= eq, hjust = 0, vjust = 1), ## hjust, vjust ???
             size = 4, parse = TRUE, inherit.aes=FALSE); ## 注意: parse = TRUE !!!
```

# equation 还可以用 paste 函数写 (更简单)

```
## 计算 ...
m = lm(Fertility ~ Education, swiss);
c = cor.test( swiss$Fertility, swiss$Education );

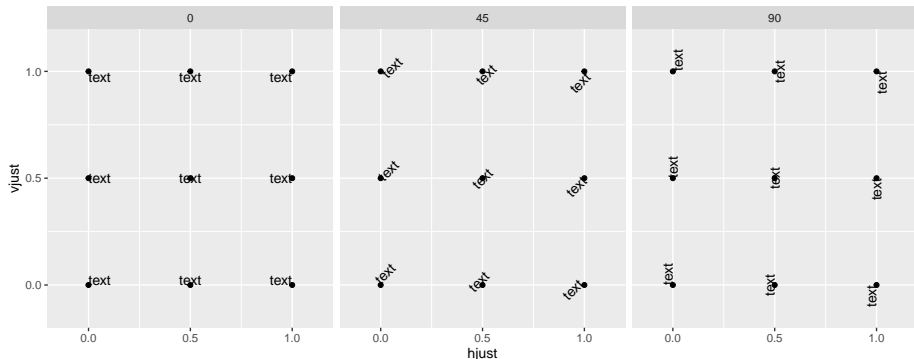
## 生成公式
eq <- substitute( atop( paste( italic(y), " = ", a + b %.% italic(x), sep = ""),
                           paste( italic(r)^2, " = ", r2, ", ", italic(p) == pvalue, sep = "" ) ),
                  list(a = as.vector( format(coef(m)[1], digits = 2) ),
                      b = as.vector( format(coef(m)[2], digits = 2) ),
                      r2 = as.vector( format(summary(m)$r.squared, digits = 2) ),
                      pvalue = as.vector( format( c$p.value , digits = 2) ) )
                );

## 用 as.expression 对公式进行转化 !!!!
eq <- as.character(as.expression(eq));

## 作图, 三个图层; 特别是 geom_text 使用自己的 data 和 aes ...
ggplot(swiss, aes( x = Education, y = Fertility ) ) +
  geom_point( shape = 20 ) +
  geom_smooth( se = T ) + ## smooth line ...
  geom_text( data = NULL,
            aes( x = 30, y = 80, label= eq, hjust = 0, vjust = 1), ## hjust, vjust ???
            size = 4, parse = TRUE, inherit.aes=FALSE); ## 注意: parse = TRUE !!!
```

# 每次都能学到新东西之: hjust 和 vjust

`geom_text( aes( angle, hjust, vjust ) )` 三参数详解





# 公式中的写法之代数符号

分类	R 的表达式	显示结果
代数符号	<code>expression(x + y)</code>	$x + y$
	<code>expression(x - y)</code>	$x - y$
	<code>expression(x * y)</code>	$xy$
	<code>expression(x / y)</code>	$x/y$
	<code>expression(x %+-% y)</code>	$x \pm y$
	<code>expression(x %/% y)</code>	$x \div y$
	<code>expression(x %*% y)</code>	$x \times y$
	<code>expression(x %.% y)</code>	$x \cdot y$
	<code>expression(x[i])</code>	$x_i$
	<code>expression(x^2)</code>	$x^2$
	<code>expression(sqrt(x))</code>	$\sqrt{x}$
	<code>expression(sqrt(x,y))</code>	$\sqrt[4]{x}$
	<code>expression(list(x,yz))</code>	$x, y, z$

## 公式, cont.

分类	R 的表达式	显示结果
变量间的关系符号	<code>expression(x==y)</code>	$x = y$
	<code>expression(x!=y)</code>	$x \neq y$
	<code>expression(x&lt;=y)</code>	$x \leq y$
	<code>expression(x&gt;=y)</code>	$x \geq y$
	<code>expression(x %~~% y)</code>	$x \approx y$
	<code>expression(x %~=% y)</code>	$x \cong y$
	<code>expression(x %==% y)</code>	$x \equiv y$
	<code>expression(x %prop% y)</code>	$x \propto y$
	<code>expression(x %~% y)</code>	$x \sim y$
列表符号	<code>expression( list(x[1], ..., x[n] )</code>	$x_1, \dots, x_n$
	<code>expression( list(x[1]+...+x[n]) )</code>	$x_1 + \dots + x_n$

## 公式, cont.

分类	R 的表达式	显示结果
数组之间关系的符号	<code>expression( x %subset% y )</code>	$x \subset y$
	<code>expression( x %subteq% y )</code>	$x \subseteq y$
	<code>expression( x %supset% y )</code>	$x \supset y$
	<code>expression( x %supseteq% y )</code>	$x \supseteq y$
	<code>expression( x %notsubset% y )</code>	$x \not\subset y$
	<code>expression( x %in% y )</code>	$x \in y$
注音符号	<code>expression( hat(x) )</code>	$\hat{x}$
	<code>expression( tilde(x) )</code>	$\tilde{x}$
	<code>expression( bar(xy) )</code>	$\bar{xy}$
	<code>expression( widehat(xy) )</code>	$\widehat{xy}$
	<code>expression( widetilde(x) )</code>	$\widetilde{xy}$

## 公式, cont.

分类	R 的表达式	显示结果
各种箭头	<code>expression( x %&lt;-&gt;% y )</code>	$x \leftrightarrow y$
	<code>expression( x %&gt;-% y )</code>	$x \rightarrow y$
	<code>expression( x %&lt;-% y )</code>	$x \leftarrow y$
	<code>expression( x %up% y )</code>	$x \uparrow y$
	<code>expression( x %down% y )</code>	$x \downarrow y$
	<code>expression( x %&lt;=&gt;% y )</code>	$x \Leftrightarrow y$
	<code>expression( x %=&gt;% y )</code>	$x \Rightarrow y$
	<code>expression( x %&lt;=% y )</code>	$x \Leftarrow y$
	<code>expression( x %dblup% y )</code>	$x \Uparrow y$
	<code>expression( x %dbldown% y )</code>	$x \Downarrow y$

## 公式, cont.

分类	R 的表达式	显示结果
特殊符号	<code>expression( infinity )</code>	$\infty$
	<code>expression( 32 * degree )</code>	$32^{\circ}$
	<code>expression( 60 * minute )</code>	$60'$
	<code>expression( 30 * second )</code>	$30''$
空白	<code>expression( x ~~ y )</code>	$xy$
	<code>expression( x + phantom() + y )</code>	$x + y$
分式	<code>expression( frac(x,y) )</code>	$\frac{x}{y}$
	<code>expression( over(x,y) )</code>	$x + \frac{1}{y}$
	<code>expression( atop(x,y) )</code>	$\frac{x}{y}$

## 公式, cont.

分类	R 的表达式	显示结果
大型操作符	<code>expression( sum(x[i], i = 1, n ) )</code>	$\sum_1^n x_i$
	<code>expression( prod(plain(P)(X==x),x )</code>	$\prod_x P(X = x)$
	<code>expression( integral( f(x) * dx, a,b ) )</code>	$\int_a^b f(x) dx$
	<code>expression( union(A[i], i==1,n) )</code>	$\bigcup_{i=1}^n A_i$
	<code>expression( intersect(A[i], i==1, n) )</code>	$\bigcap_{i=1}^n A_i$
	<code>expression( lim(f(x), x %&gt;% 0) )</code>	$\lim_{x \rightarrow 0} f(x)$
分组	<code>expression( min(g(x), x &gt;= 0) )</code>	$\min_{x \geq 0} g(x)$
	<code>expression( group("(", list(a, b), "]" ) )</code>	$(a, b]$
	<code>expression( bgroup("(", atop(x, y), ")") )</code>	$\begin{pmatrix} x \\ y \end{pmatrix}$
	<code>expression( group(lceil, x, rceil) )</code>	$\lceil x \rceil$
	<code>expression( group(lfloor, x, rfloor) )</code>	$\lfloor x \rfloor$

# 希腊字符

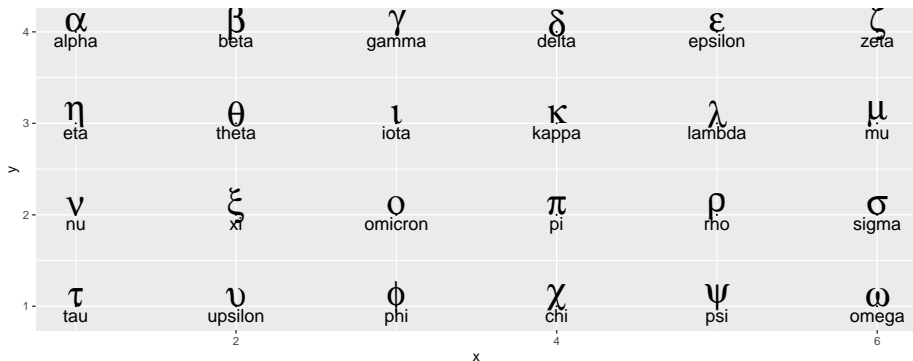
## 代码

```
greeks <- c("Alpha", "Beta", "Gamma", "Delta", "Epsilon", "Zeta",
            "Eta", "Theta", "Iota", "Kappa", "Lambda", "Mu",
            "Nu", "Xi", "Omicron", "Pi", "Rho", "Sigma",
            "Tau", "Upsilon", "Phi", "Chi", "Psi", "Omega");

dat <- data.frame( x = rep( 1:6, 4 ), y = rep( 4:1, each = 6), greek = greeks );

plot2 <-
  ggplot( dat, aes(x=x,y=y) ) + geom_point(size = 0) +
  # 画希腊字符, 注意下面两行代码的区别
  geom_text( aes( x, y + 0.1, label = tolower( greek ) ), size = 10, parse = T ) +
  geom_text( aes( x, y - 0.1, label = tolower( greek ) ), size = 5 );
```

## 希腊字符, cont.



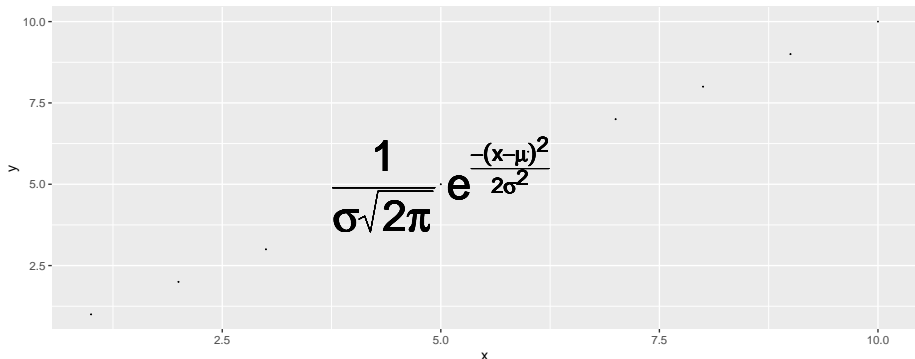


# 公式示例

## 注写公式的方式很多

```
eq <- expression(paste(frac(1, sigma*sqrt(2*pi)), " ",
                        plain(e)^{frac(-(x-mu)^2, 2*sigma^2)}));

ggplot( data.frame(x=1:10, y=1:10), aes( x,y ) ) +
  geom_point( size = 0 ) +
  geom_text(data = NULL, x = 5, y = 5, size = 12,
            label = as.character(eq), parse = TRUE );
```



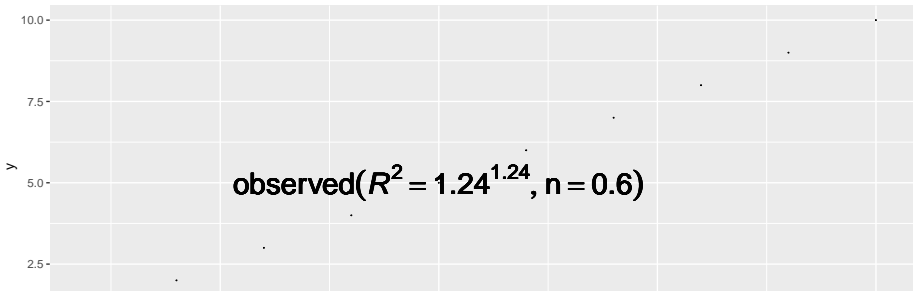
## 公式示例 2

另一种代入变量值的方法：

```
x <- 1.24;
y <- 0.6;

ex <- bquote(.(parse(text=paste( "observed (", "italic(R)^2==",
                                x,  "^bold(", x,  ")", n == ", y,  ")",
                                sep = " "  ))) );

ggplot( data.frame(x=1:10, y=1:10), aes( x,y ) ) + geom_point( size = 0 ) +
  geom_text(data = NULL, x = 5, y = 5, size = 8,
            label = as.character(ex), parse = TRUE );
```



## 公式示例 3

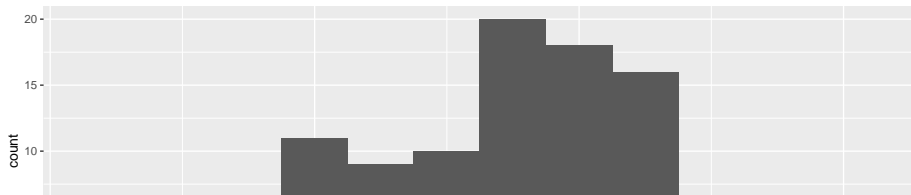
使用 paste 和 substitute :

```
x_mean <- 1.5;
x_sd <- 1.2;

# 表达式
ex <- substitute(
  paste(X[i], " ~ N(", mu, "=", m, ", ", " ", sigma^2, "=", s2, ")"),
  list(m = x_mean, s2 = x_sd^2)
);

# histogram
ggplot( data.frame( x = rnorm(100, x_mean, x_sd) ), aes( x ) ) +
  geom_histogram( binwidth=0.5 ) +
  ggtitle(ex); ## 为什么不需要 parse = TRUE ???
```

$$X_i \sim N(\mu=1.5, \sigma^2=1.44)$$



## section 5: ggplot2 进阶 2

# ggplot2 的核心在于先计算再做图

先看数据 (来自 talk05):

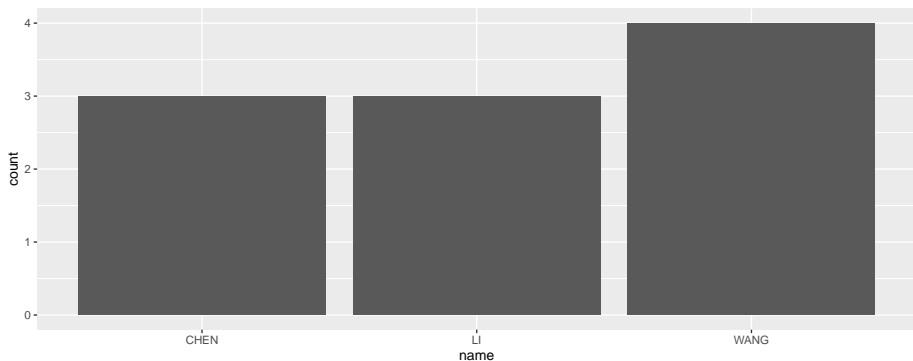
```
grades2 <- read_delim( file = "data/talk05/grades2.txt", delim = "\t",
                        quote = "\"", col_names = T);
knitr::kable( grades2 );
```

name	class	course	grade
CHEN	1	bioinformatics	90
CHEN	1	chemistry	92
CHEN	2	chinese	35
CHEN	3	german	62
LI	1	bioinformatics	44
LI	2	chinese	68
LI	3	microbiology	95
LI	3	japanese	90
WANG	1	bioinformatics	35
WANG	1	chemistry	76
WANG	1	mathmatics	82
WANG	3	german	100
WANG	3	spanish	78

# geom\_bar

任务：画出每位学生及格的课程数

```
ggplot( grades2 %>% filter( grade >= 60 ), aes( name ) ) +  
  geom_bar();
```

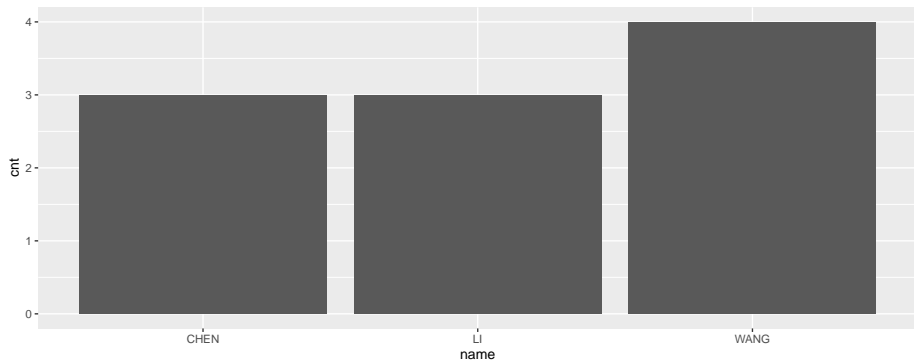


为什么会这样呢？因为 `geom_bar( stat = "count" )` 的默认参数是 `count`，即数一下每个 factor 的出现次数。

## geom\_bar , cont.

以上命令，实际上等于：

```
## 先做统计
cnt <- grades2 %>% group_by( name ) %>% summarise( cnt = sum( grade >= 60 ) );
ggplot( cnt, aes( x = name, y = cnt ) ) +
  geom_bar( stat = "identity" );
```



## default stat behaviors (默认计算方法)

- `geom_bar` : `count`
- `geom_boxplot` : `boxplot`
- `geom_count` : `sum`
- `geom_density` : `density`
- `geom_histogram` : `bin`
- `geom_quantile` : `quantile` ...



# stacked bars

应用场景：宏基因组多样本物种丰度图

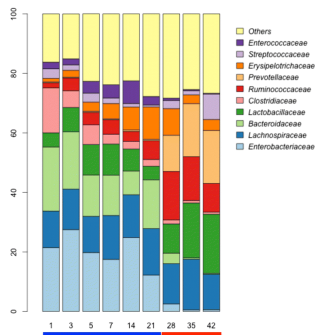


Figure 5: Microbiome 3, 28 2015

# stacked bars , cont.

## load data

```
speabu <-read_tsv( file = "data/talk09/mock_species_abundance.txt" );
```

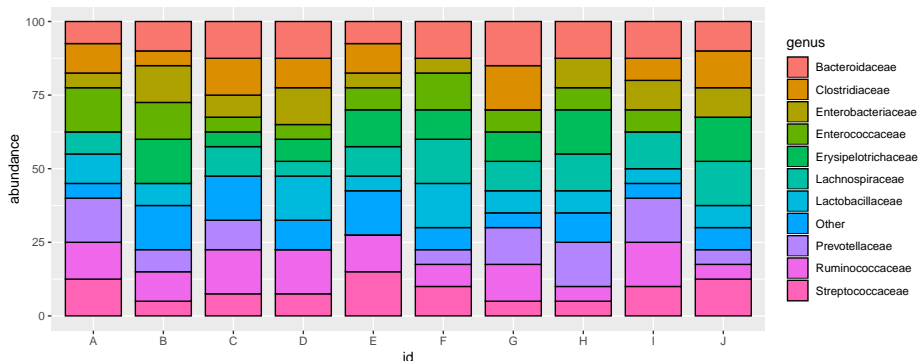
```
## Parsed with column specification:
## cols(
##   id = col_character(),
##   genus = col_character(),
##   abundance = col_double()
## )
```

```
head( speabu );
```

```
## # A tibble: 6 x 3
##   id      genus      abundance
##   <chr> <chr>      <dbl>
## 1 A      Enterobacteriaceae      5
## 2 A      Lachnospiraceae      7.5
## 3 A      Bacteroidaceae      7.5
## 4 A      Lactobacillaceae      10
## 5 A      Clostridiaceae      10
## 6 A      Ruminococcaceae     12.5
```

# stacked bars , cont.

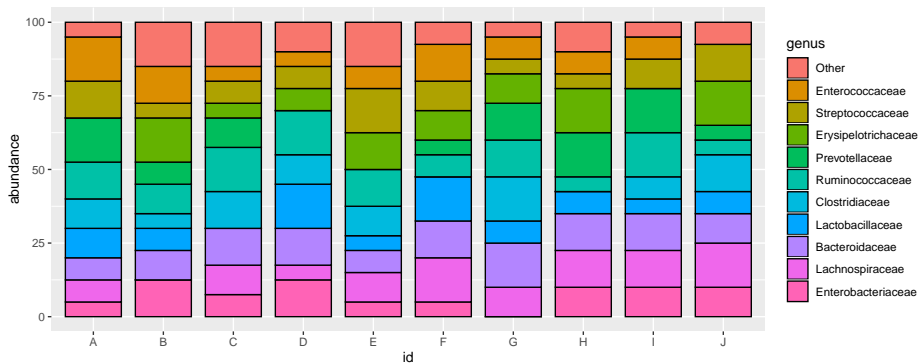
```
ggplot( speabu, aes( x = id, y = abundance, fill = genus ) ) +  
  geom_bar( stat = "identity", position = "stack", color = "black", width = 0.8 );
```



# 指定 Genus 展示顺序

factor 的操纵详见第 4 章。

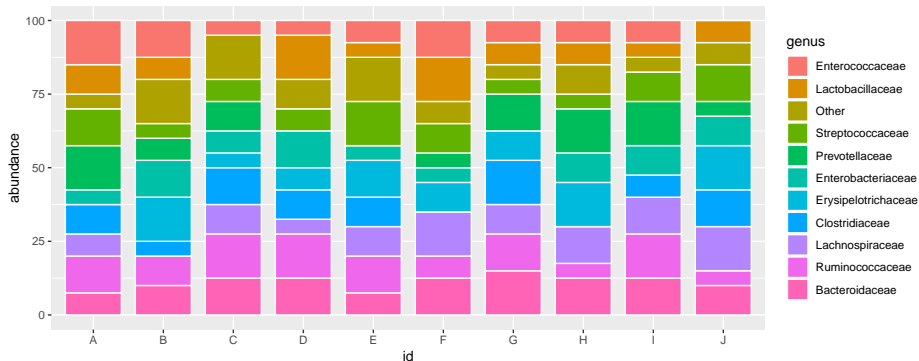
```
speabu$genus <- factor( speabu$genus, levels = rev( c( "Enterobacteriaceae", "Lachnospiraceae",
  "Clostridiaceae", "Ruminococcaceae", "Prevotellaceae", "Erysipelotrichaceae", "Streptococcaceae" ) ),
  ggplot( speabu, aes( x = id, y = abundance, fill = genus ) ) +
  geom_bar( stat = "identity", position = "stack", color = "black", width = 0.8 );
```



# 按丰度排序

## 按丰度中值大小排序

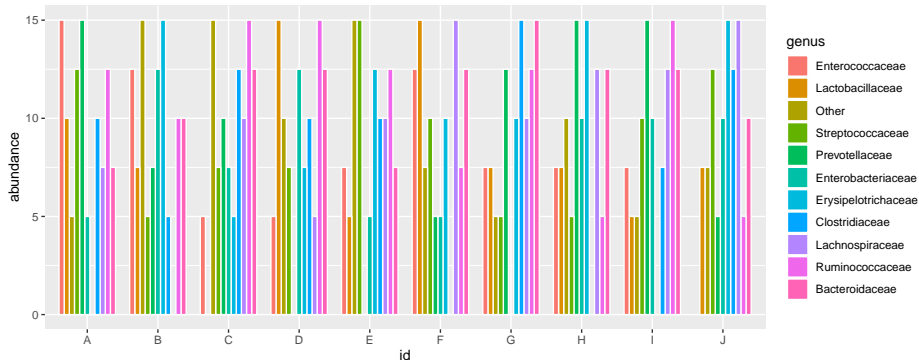
```
speabu$genus <- reorder( speabu$genus, speabu$abundance, median );
ggplot( speabu, aes( x = id, y = abundance, fill = genus ) ) +
  geom_bar( stat = "identity", position = "stack", color = "white", width = 0.8 );
```



## position = "stack" 又是什么 ??

position = "dodge" : plot bars next to each other ...

```
ggplot( speabu, aes( x = id, y = abundance, fill = genus ) ) +  
  geom_bar( stat = "identity", position = "dodge", color = "white", width = 0.8 );
```



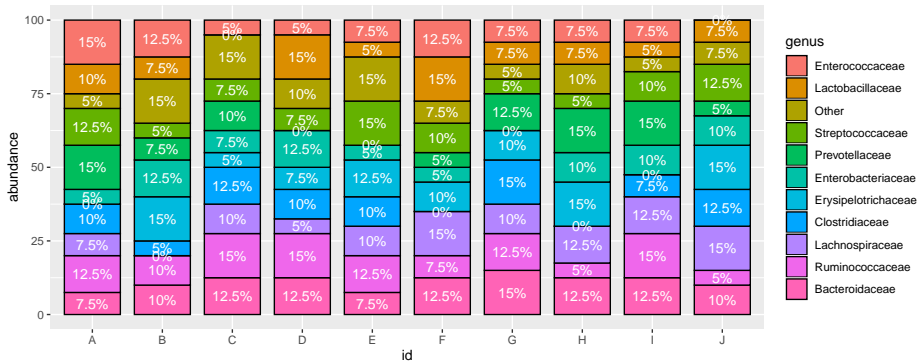
## 显示数值 ...

```
## 先计算显示位置
```

```
speabu <- speabu %>% arrange( id, desc( factor( genus ) ) ) %>%
  group_by( id ) %>% mutate( ypos = cumsum( abundance ) - abundance / 2 );
```

```
## 画图
```

```
ggplot( speabu, aes( x = id, y = abundance, fill = genus ) ) +
  geom_bar( stat = "identity", position = "stack", color = "black", width = 0.8 ) +
  geom_text( aes( y = ypos, label = paste( abundance, "%", sep = " " ) ), color = "white" );
```



## 显示数值 ... , cont.

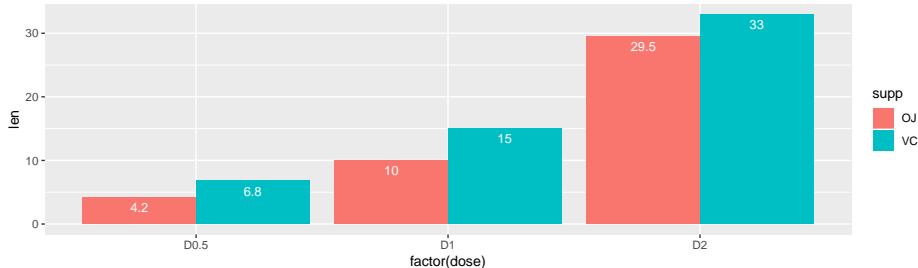
### 要点

- 使用 `ddplyr` 的 `cumsum()` 函数 ...
- 计算位置: 当前累加值 - 自身值/2, 使数字显示在当前值的中间
- 累加前, 要对数据按 factors 进行排序; 通过 `arrange` 函数实现;



## 在 position = "dodge" 的情况下添加 label

```
df2 <- data.frame(supp=rep(c("VC", "OJ"), each=3),
                  dose=rep(c("D0.5", "D1", "D2"),2),
                  len=c(6.8, 15, 33, 4.2, 10, 29.5))
ggplot( df2, aes(x=factor(dose), y=len, fill=supp)) +
  geom_bar(stat="identity", position=position_dodge()) +
  geom_text(aes(label=len), vjust=1.6, color="white",
            position = position_dodge(0.9), size=3.5)
```



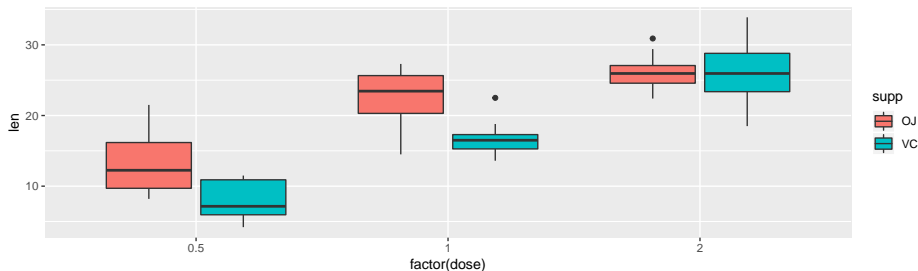
## position 的其它取值

除了 “dodge”, “stack” 之外, position 还可以:

- `position = position_stack(reverse = TRUE)`
- `position = position_dodge(reverse = TRUE)`
- `position = position_identity()`
- `position = position_jitter()` : jitter points to avoid overplotting ...
- `position = position_nudge()` : is generally useful for adjusting the position of items on discrete scales by a small amount

# 不同的图层有不同默认值

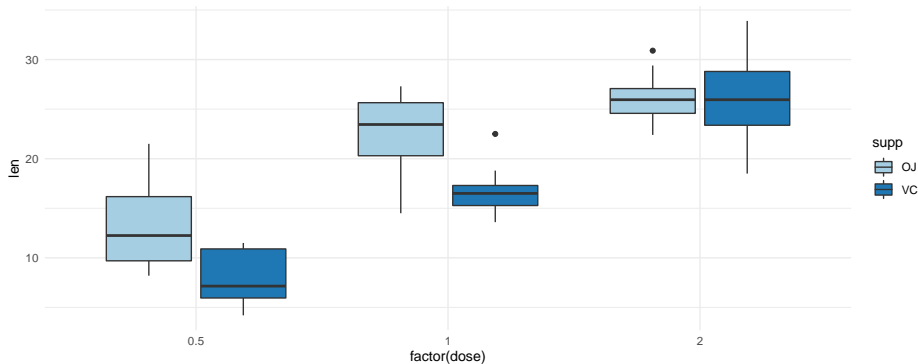
```
ggplot(ToothGrowth, aes(x=factor(dose), y=len, fill=supp)) +  
  geom_boxplot()
```



`geom_boxplot()` : 默认为 `dodge`  
 更多 boxplot 的内容见 [link](#).

# change color palette

```
ggplot(ToothGrowth, aes(x=factor(dose), y=len, fill=supp)) +  
  geom_boxplot() + scale_fill_brewer(palette = "Paired") + theme_minimal();
```



## 要点

- 1 颜色 palette 的用法
- 2 theme 系统

# theme in ggplot2

- `theme_gray` : 系统默认主题
- `theme_bw` , `theme_linedraw`, `theme_light`, `theme_dark`,  
`theme_minimal` , `theme_classic`, `theme_void()`

see here for a complete list:

<https://ggplot2.tidyverse.org/reference/ggtheme.html>

## theme() 函数

除了 `theme_` 用于调整整体视觉效果外, `ggplot2` 还提供了 `theme()` 函数用于细调。

- `line, rect, text, title` : 整体框架
- `axis.<compoment>` : 调整坐标轴
- `legend.<parameter>` : 调整图例
- `plot.<>` : 控制 title, subtitle 等细节
- `panel.<...>` : 调整 facet 情况下的 panel (facet 下面会介绍)
- `strip.<...>` : 调整 facet 的标题细节 ...

更多详见:

官方: <https://ggplot2.tidyverse.org/reference/theme.html>

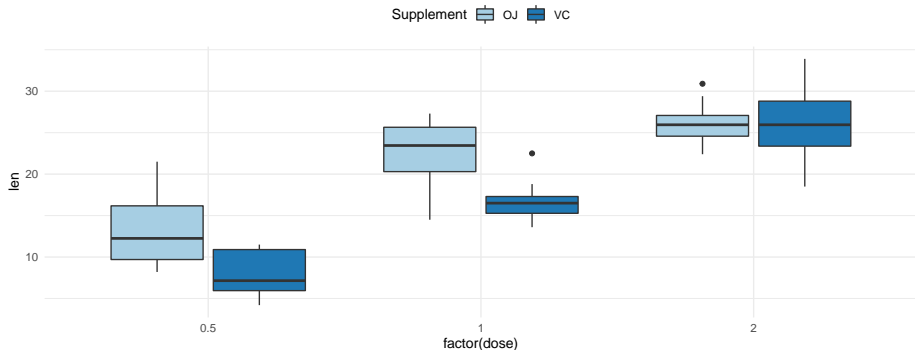
定制: <http://www.sthda.com/english/wiki/ggplot2-themes-and-background-colors-the-3-elements>

扩展 theme extensions:

<http://www.ggplot2-exts.org/ggthemes.html>

# legend 细调

```
ggplot(ToothGrowth, aes(x=factor( dose ), y=len, fill=supp)) +
  geom_boxplot() + scale_fill_brewer( palette = "Paired" ) + theme_minimal() +
  labs( fill = "Supplement" ) + theme( legend.position = "top" );
```



# ggsci: palette for scientific journals!!!

```
# Install ggsci from CRAN:  
install.packages("ggsci");  
  
# Or try the development version on GitHub:  
# install.packages("devtools")  
devtools::install_github("nanxstats/ggsci")
```

提供了一系列

`scale_color_<journal>` 和 `scale_fill_<journal>` 函数

详见: <https://nanx.me/ggsci/articles/ggsci.html>



# ggsci 举例

```
library("ggsci")
library("ggplot2")
library("gridExtra")
data("diamonds")
p1 <- ggplot(
  subset(diamonds, carat >= 2.2),
  aes(x = table, y = price, colour = cut)
) +
  geom_point(alpha = 0.7) +
  geom_smooth(method = "loess", alpha = 0.05, size = 1, span = 1) +
  theme_bw()

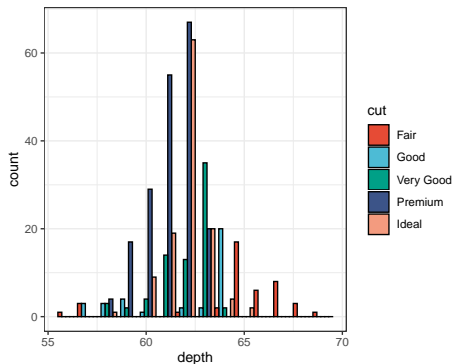
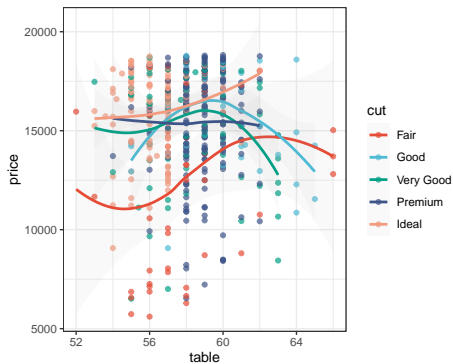
p2 <- ggplot(
  subset(diamonds, carat > 2.2 & depth > 55 & depth < 70),
  aes(x = depth, fill = cut)
) +
  geom_histogram(colour = "black", binwidth = 1, position = "dodge") +
  theme_bw()
```

## 要点

- `library(gridExtra)`

# ggsci 结果, Nature Style !!

```
p1_npg <- p1 + scale_color_npg()
p2_npg <- p2 + scale_fill_npg()
grid.arrange(p1_npg, p2_npg, ncol = 2)
```

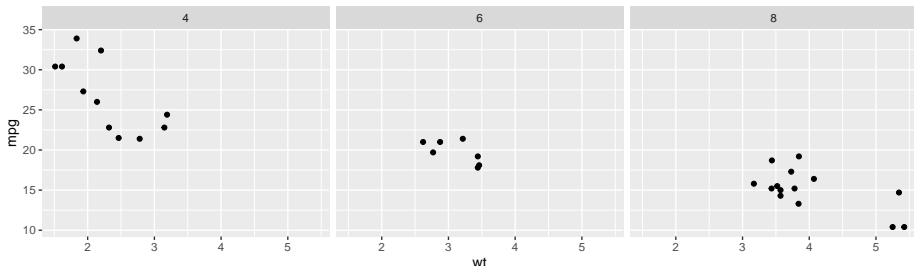


# faceting ...

Faceting generates small multiples each showing a different subset of the data.

`facet_grid( <by_row> ~ <by_col> )` 汽缸、车重与燃油效率间的关系

```
ggplot( mtcars, aes( x = wt, y = mpg ) ) +  
  geom_point() +  
  facet_grid( . ~ cyl );
```



# faceting , cont.

by col: 请自行尝试 ~

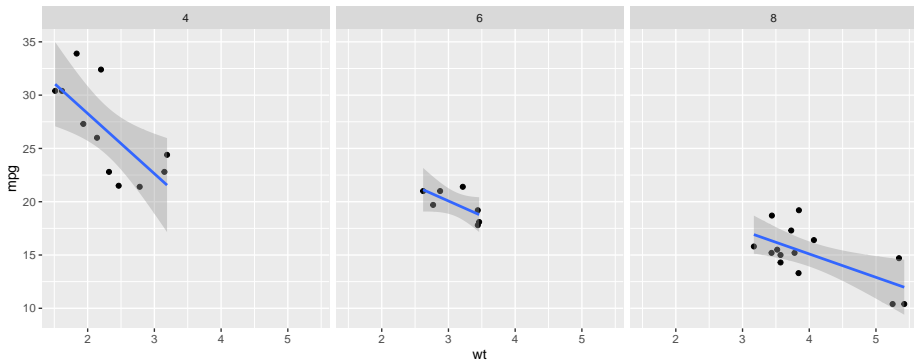
```
ggplot( mtcars, aes( x = wt, y = mpg ) ) +  
  geom_point() +  
  facet_grid( cyl ~ . );
```

**\*\* 注意 \*\***

作图相关概念: panel, strip, axis, tick, tick label, axis label

# facet\_grid 的适用范围是全局

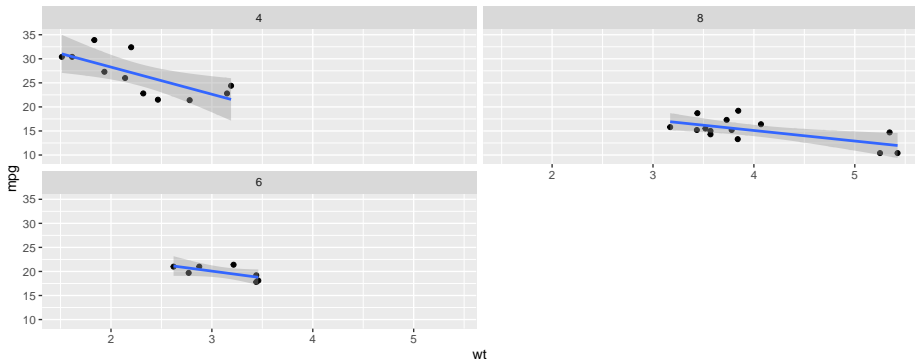
```
ggplot( mtcars, aes( x = wt, y = mpg ) ) +  
  geom_point() + geom_smooth( method = "lm" ) +  
  facet_grid( . ~ cyl );
```



# facet\_wrap

## 指定行、列数和方向

```
ggplot( mtcars, aes( x = wt, y = mpg ) ) +
  geom_point() + geom_smooth( method = "lm" ) +
  facet_wrap( . ~ cyl , ncol = 2, dir = "v" );
```



# ggplot2 小结

## layered grammar (图层语法) 的成分

- 图层 ( `geom_xxx` )
- `scale` ( `scale_xxx` )
- `faceting` ( `facet_xxx` )
- 坐标系统

## 图象类型

- 点图
- bars
- boxplots

## 其它重要内容 (部分需要自学)

- colours
- theme
- 其它图像类型
- 图例 (legends) 和坐标轴
- 图形注释和其它定制

## 写在后面

- ① ggplot2 博大精深，需要一门课去讲
- ② 上手容易，精通难
- ③ 太多记忆点
- ④ 本节内容只涉及了基础中的基础，更多内容，包括进阶技巧和生信相关的扩展包，更多的需要同学们自行探索。

请见 “ggplot2 : elegant graphics for data analysis” 一书!!



## section 6: Exercise and home work

# more to read & 练习

- 1 必读!!! 在一张图上画多个 panel
- 2 ggplot2 提供的官方指南从 A 到 Z:
  - abline
  - [bar charts] ([https://ggplot2.tidyverse.org/reference/geom\\_bar.html](https://ggplot2.tidyverse.org/reference/geom_bar.html))
  - ...
- 3 color brewer
- 4 legends 1
- 5 legends
- 6 必读!! R graph gallery : <https://www.r-graph-gallery.com>
- 7 必读!! ggplot2 扩展: <http://www.ggplot2-exts.org/gallery>
- 8 barchart & theme
- 9 plotly 互动图形的领先者: [plot.ly](http://plot.ly)

# 下次预告

data summarisation and modeling