基于 ggplot2 可视化的入门教程

Create Elegant Data Visualisations Using the Grammar of Graphics

2024-01-18

Kangguo Li
School of Public Health, Xiamen University

Press Space for next page →

Table of Contents

Introduction

ggplot2 is a system for declaratively creating graphics, based on The Grammar of Graphics. You provide the data, tell ggplot2 how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details.

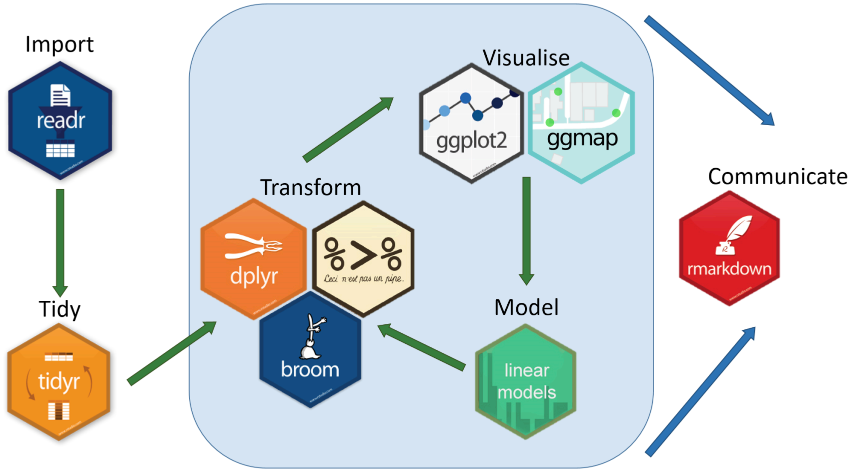
> ggplot2 是一个基于图形语法声明式创建图形的系统。您提供数据，告诉 ggplot2 如何将变量映射到美学，使用什么图形代码，它会处理细节。

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

> ggplot2 是 tidyverse 的一部分，也就是说我们安装 tidyverse 的时候，ggplot2 也会被安装。

Installation

```
install.packages("tidyverse") # Install tidyverse
library(tidyverse) # Load tidyverse
```



Source: The Tidyverse · Teach Data Science. (2024, January 18). Retrieved from https://teachdatascience.com/tidyverse

What is a Plot?

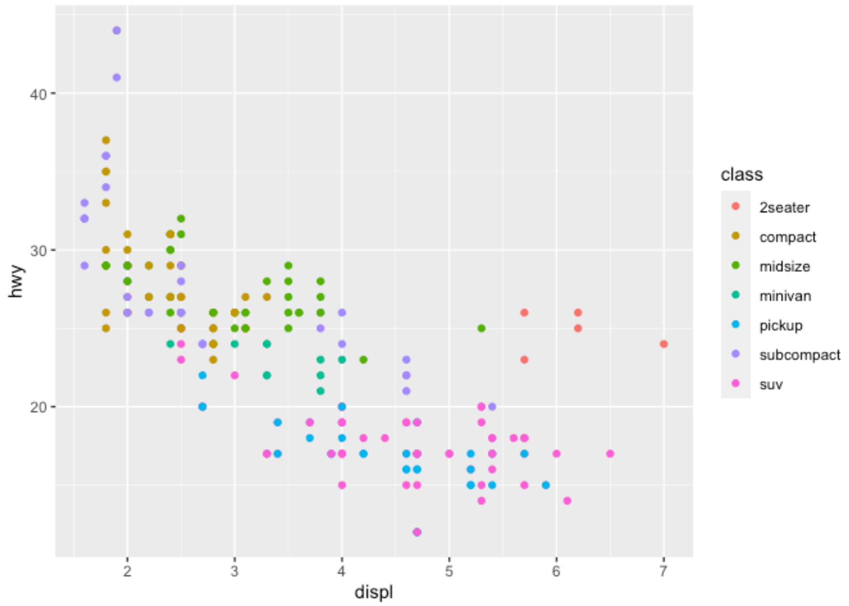
�♀ **Question:** 大家都知道哪些统计图形？

运行这个代码，你会发现，这个图形是由一系列的点组成的，这些点的位置是由 x 和 y 决定的，而点的颜色是由 class 决定的。

```
 ggplot(data = mpg,

        mapping = aes(x = displ,
                      y = hwy,
                      colour = class)) +

   geom_point()
```

�♀ **Question:** 这一个统计图由哪些元素构成的？

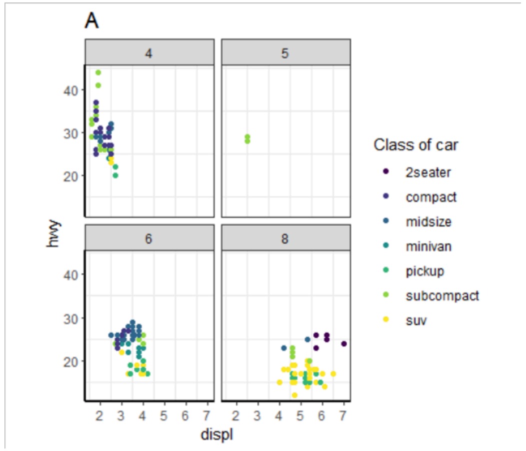
- Data(数据集)
- Aesthetics(美学映射)
- Geometry(几何对象)

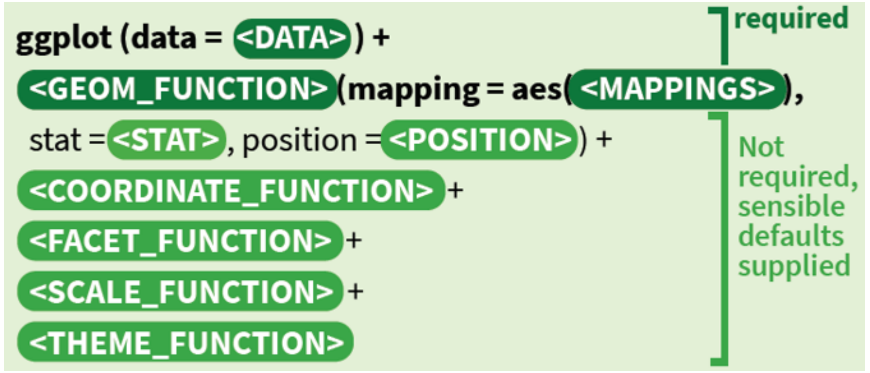


Source: Create Elegant Data Visualisations Using the Grammar of Graphics. (2023, December 18). Retrieved from https://ggplot2.tidyverse.org

The Grammar of Graphics

```r
library(ggplot2)
ggplot(data = mpg,
    mapping = aes(x = displ,
                  y = hwy,
                  colour = class)) +
    geom_point()+
    coord_equal(ratio = 1/5)+
    facet_wrap(.~cyl,nrow = 1)+
    scale_color_viridis_d()+
    theme_bw()+
    theme(panel.grid.major = element_blank(),
          plot.title = element_text(size = 16),
          axis.line = element_line(size = 1),
          panel.background = element_rect(fill = 'transpare
          plot.margin = margin(5,5, 5,5))+
    labs(title = 'A',
         colour = 'Class of car')
```

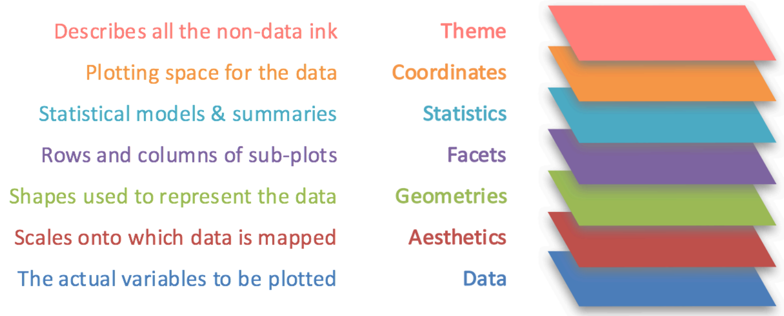


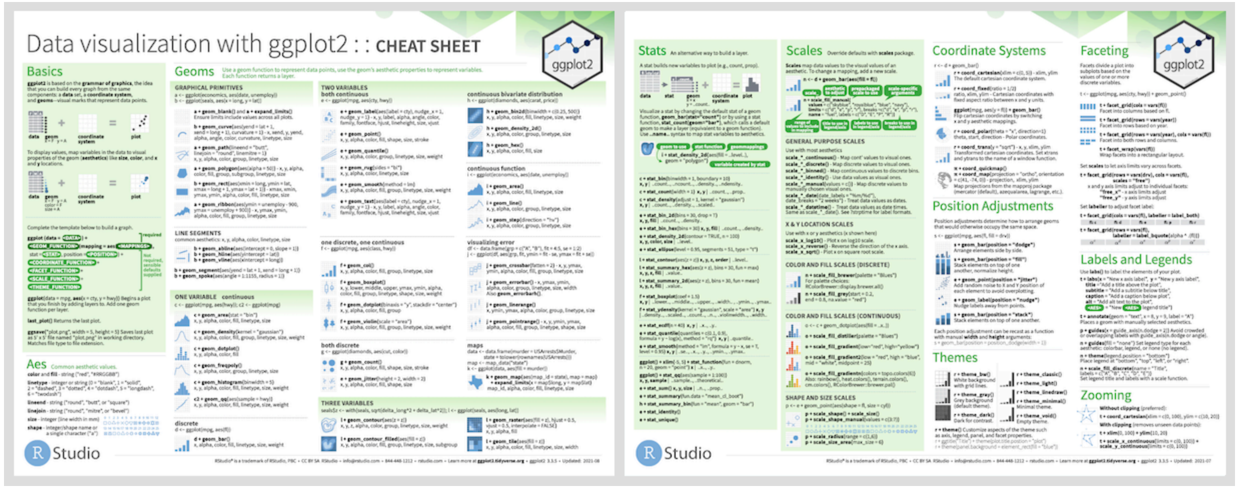

ggplot (data = <DATA>) +
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
stat = <STAT>, position = <POSITION>) +
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>

required

Not required, sensible defaults supplied

- **theme(主题)**: theme_bw()
- **Space(空间)**: coord_cartesian()
- **Statistical models(统计变换)**: stat_smooth()
- **Facets(分面)**: facet_wrap()
- **Geometry(几何对象)**: geom_point()
- **Aesthetics(美学映射)**: x, y, color
- **Data(数据集)**: mpg


Describes all the non-data ink
Theme
Plotting space for the data
Coordinates
Statistical models & summaries
Statistics
Rows and columns of sub-plots
Facets
Shapes used to represent the data
Geometries
Scales onto which data is mapped
Aesthetics
The actual variables to be plotted
Data


Data visualization with ggplot2 : : CHEAT SHEET

Basics

Geoms

Aes

Stats

Scales

Coordinate Systems

Position Adjustments

Themes

Faceting

Labels and Legends

Zooming

Source: Data Visualization Cheatsheet · RStudio. (2024, January 18). Retrieved from https://rstudio.com/resources/cheatsheets/

Data

数据是统计图形的核心，作图之前一定要先读懂数据。

长数据

个体	变量	值
1	x	1
1	y	2
2	x	3
2	y	4

宽数据

个体	x	y
1	1	2
2	3	4

♀ **Question:** 长数据还是宽数据？

一般选择长数据，因为长数据更容易处理。

Data transformation

数据的长宽转换

```r
library(tidyverse)
data <- tibble(
  x = 1:4,
  y = 1:4,
  z = 1:4
)
data
```

```
      x     y     z
  <int> <int> <int>
1     1     1     1
2     2     2     2
3     3     3     3
4     4     4     4
```

```r
data |>
    pivot_longer(cols = everything())
```

```
  name  value
  <chr> <int>
1 x         1
2 y         1
3 z         1
4 x         2
5 y         2
6 z         2
7 x         3
```

```
data |>
    pivot_longer(cols = c(y, z),
                 names_to = 'name',
                 values_to = 'value')
```

```
      x name  value
  <int> <chr> <int>
1     1 y         1
2     1 z         1
3     2 y         2
4     2 z         2
5     3 y         3
```

看起来很简单对吧？但是这个函数的参数有很多，你可以通过`?pivot_longer`查看。

Aesthetics

美学映射是指将数据集中的变量映射到图形的视觉属性上，比如颜色、形状、大小等。

- x
- y
- fill
- color
- shape
- linetype
- size
- alpha
- group

这些并不是固定的，你可以自己定义。

如果需要将变量映射到美学上，需要在`aes()`中定义。

对于支持的美学映射，可以通过`?aes`或者`?geom_xxxx`查看。

```r
library(ggplot2)
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ,
                             y = hwy,
                             colour = class))+
    coord_equal(ratio = 1/5)+
    facet_wrap(.~cyl,nrow = 1)+
    scale_color_viridis_d()+
    theme_bw()+
    theme(panel.grid.major = element_blank(),
          plot.title = element_text(size = 16),
          axis.line = element_line(size = 1),
          panel.background = element_rect(fill = 'transparent', colour = 'black'),
          plot.margin = margin(5,5, 5,5))+
    labs(title = 'A',
         colour = 'Class of car')
```

□♀ **Question:** 哪些变量被映射到了美学上？

displ, hwy, class

Geometry

几何对象是指图形的形状，比如点、线、面等。

也就是说我们需要通过`` `geom_xxxx()` ``来定义图形的形状，如散点图、折线图、柱状图等。

□♀ **Question:** 回忆下一些统计学知识，你知道哪些图形？它们分别适用于什么变量？

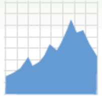
如： 散点图：两个连续性/定量变量

我们通常将变量分为几类

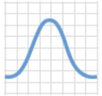
- 定性变量，如性别、学历等。
  - 有序定性变量，如学历。
  - 无序定性变量，如性别。
- 定量变量，如年龄、身高等。

One variable

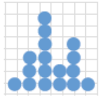
定量变量



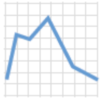
c + geom_area(stat = "bin"**)**
x, y, alpha, color, fill, linetype, size



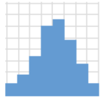
c + geom_density(kernel = "gaussian"**)**
x, y, alpha, color, fill, group, linetype, size, weight



c + geom_dotplot()
x, y, alpha, color, fill

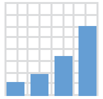


c + geom_freqpoly()
x, y, alpha, color, group, linetype, size



c + geom_histogram(binwidth = 5**)**
x, y, alpha, color, fill, linetype, size, weight

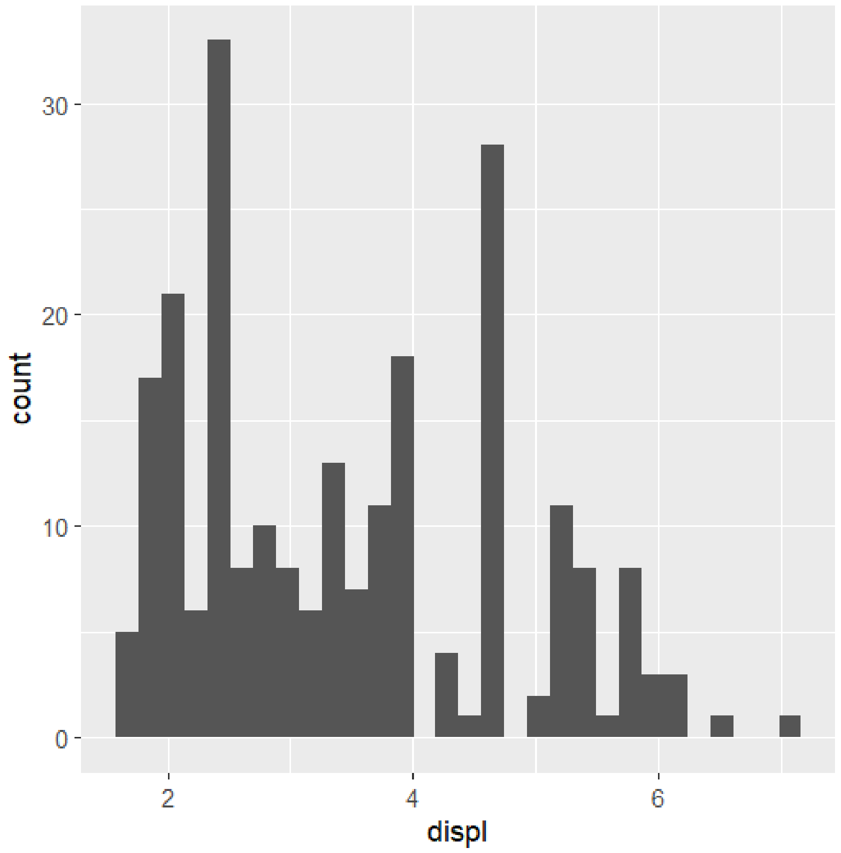
定性变量



d + geom_bar()
x, alpha, color, fill, linetype, size, weight

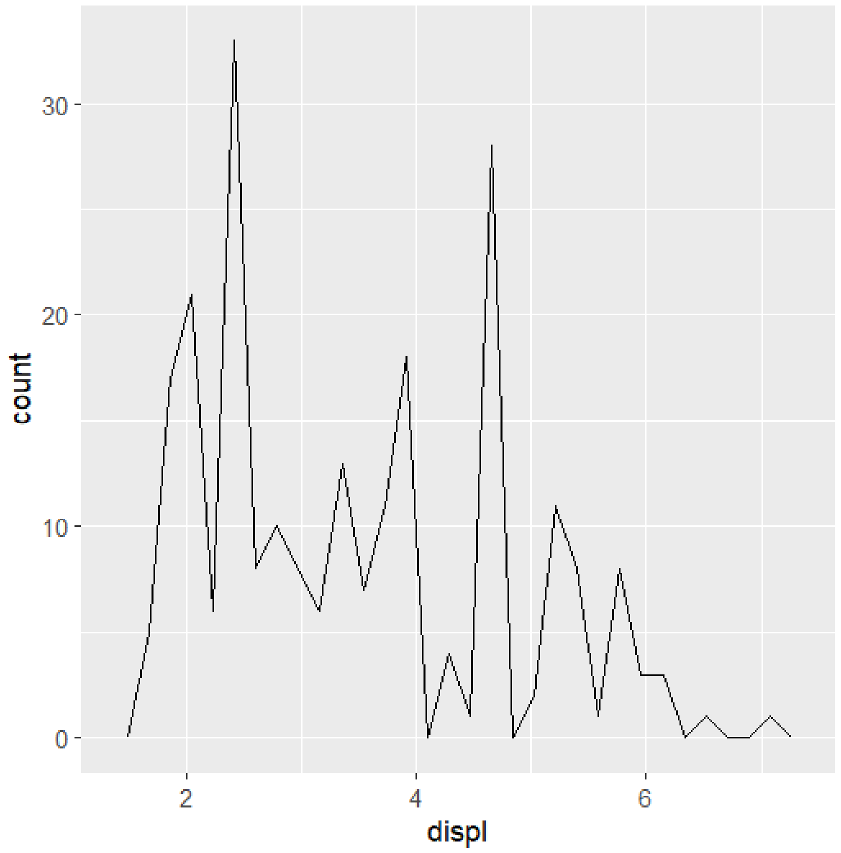
直方图

```
ggplot(data = mpg) +
    geom_histogram(mapping = aes(x = displ))

ggplot(data = mpg) +
    geom_freqpoly(mapping = aes(x = displ))

ggplot(data = mpg) +
    geom_density(mapping = aes(x = displ))
```



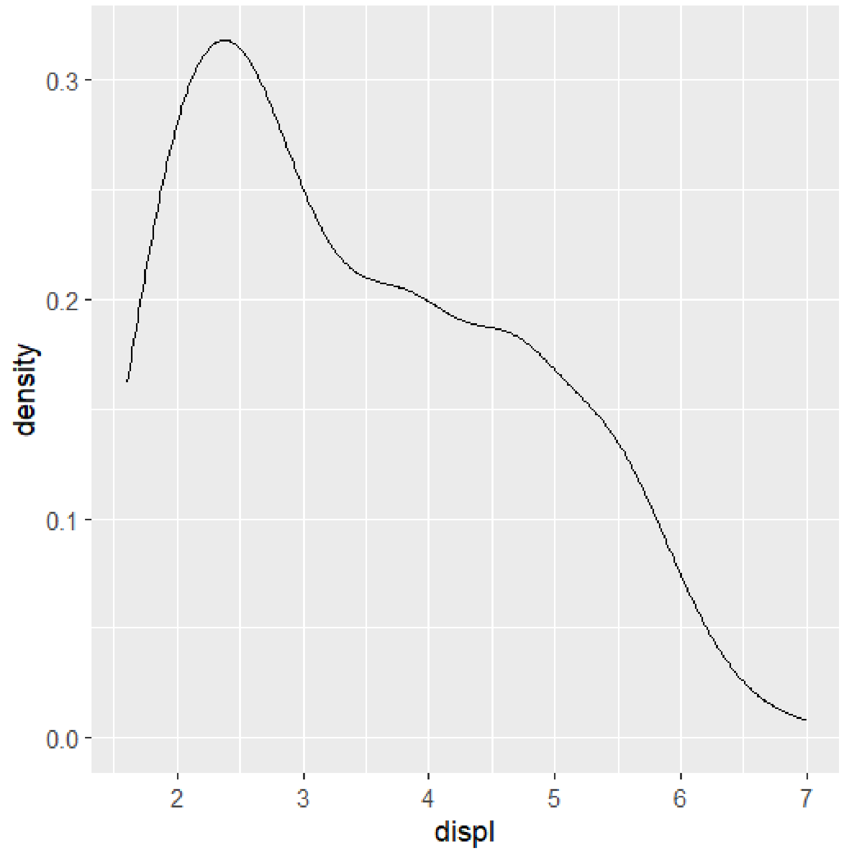
频率图

```
ggplot(data = mpg) +
    geom_histogram(mapping = aes(x = displ))

ggplot(data = mpg) +
    geom_freqpoly(mapping = aes(x = displ))

ggplot(data = mpg) +
    geom_density(mapping = aes(x = displ))
```



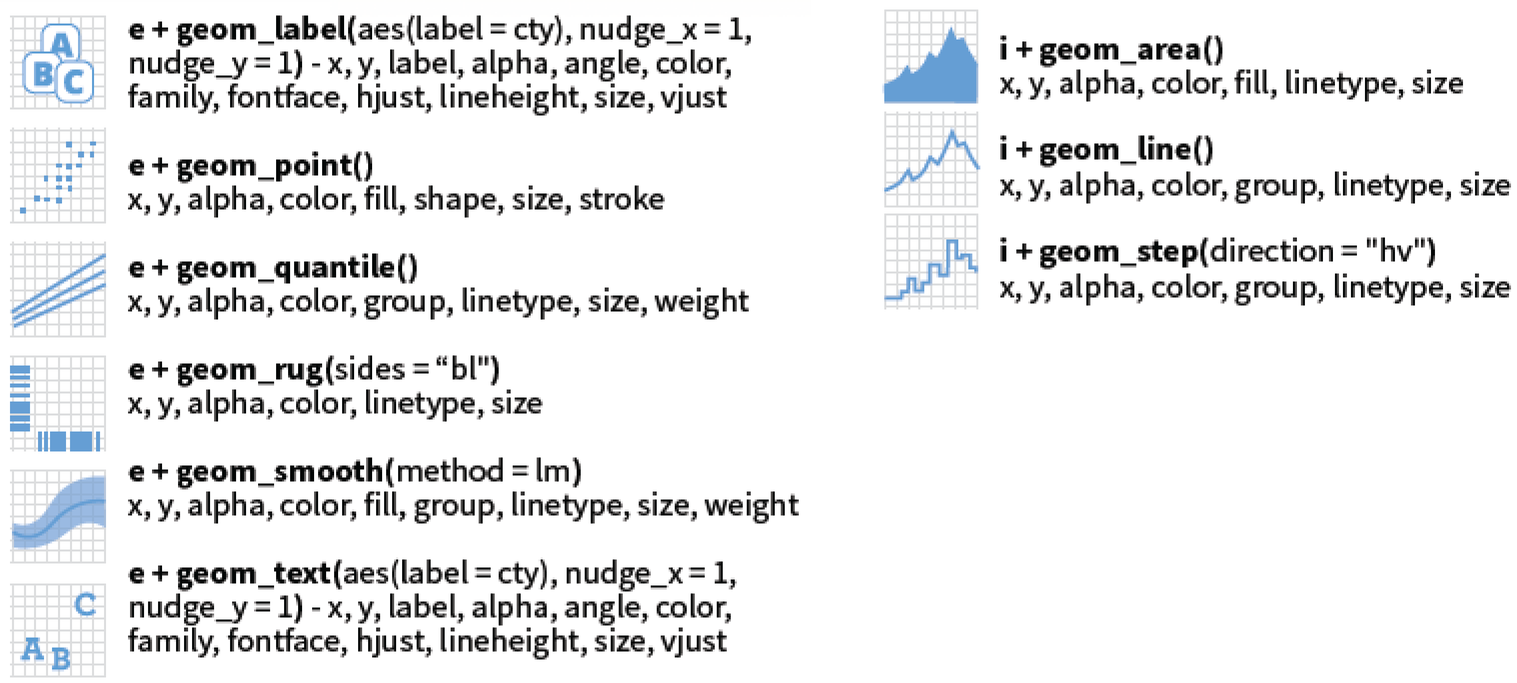
核密度图

```
ggplot(data = mpg) +
    geom_histogram(mapping = aes(x = displ))

ggplot(data = mpg) +
    geom_freqpoly(mapping = aes(x = displ))

ggplot(data = mpg) +
    geom_density(mapping = aes(x = displ))
```



Two variables

定量+定量



e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_point()
x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile()
x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = “bl")
x, y, alpha, color, linetype, size

e + geom_smooth(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

i + geom_area()
x, y, alpha, color, fill, linetype, size

i + geom_line()
x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv")
x, y, alpha, color, group, linetype, size

常用的只有 geom_point, geom_line, geom_smooth, geom_area

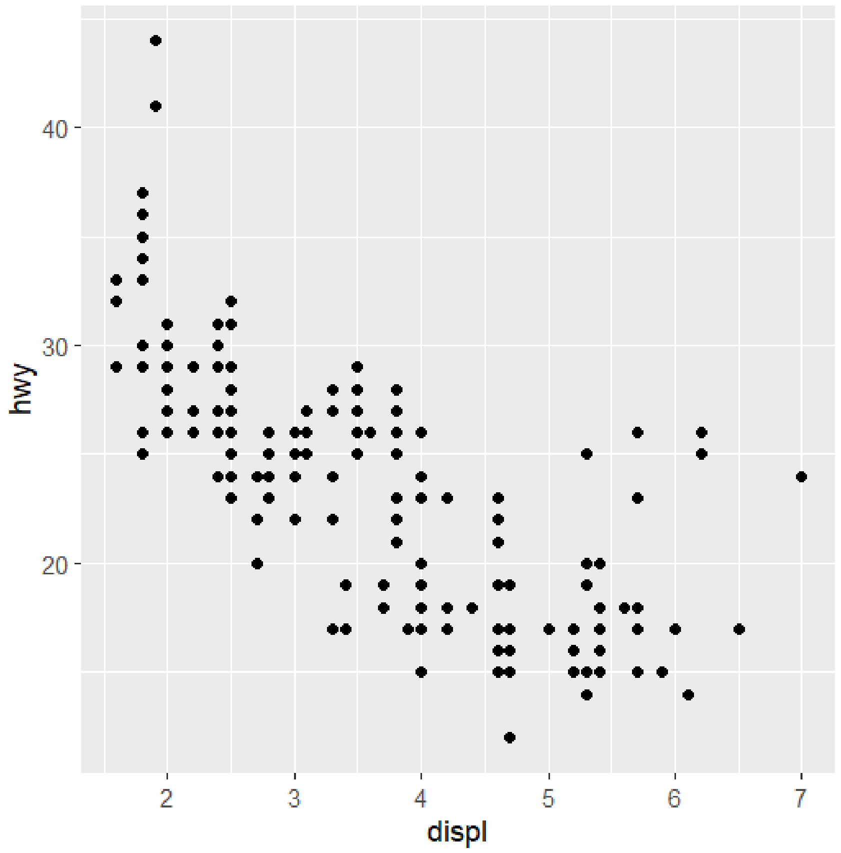
散点图

```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ,
                             y = hwy))
```

```
ggplot(data = mpg) +
    geom_line(mapping = aes(x = displ,
                            y = hwy))
```

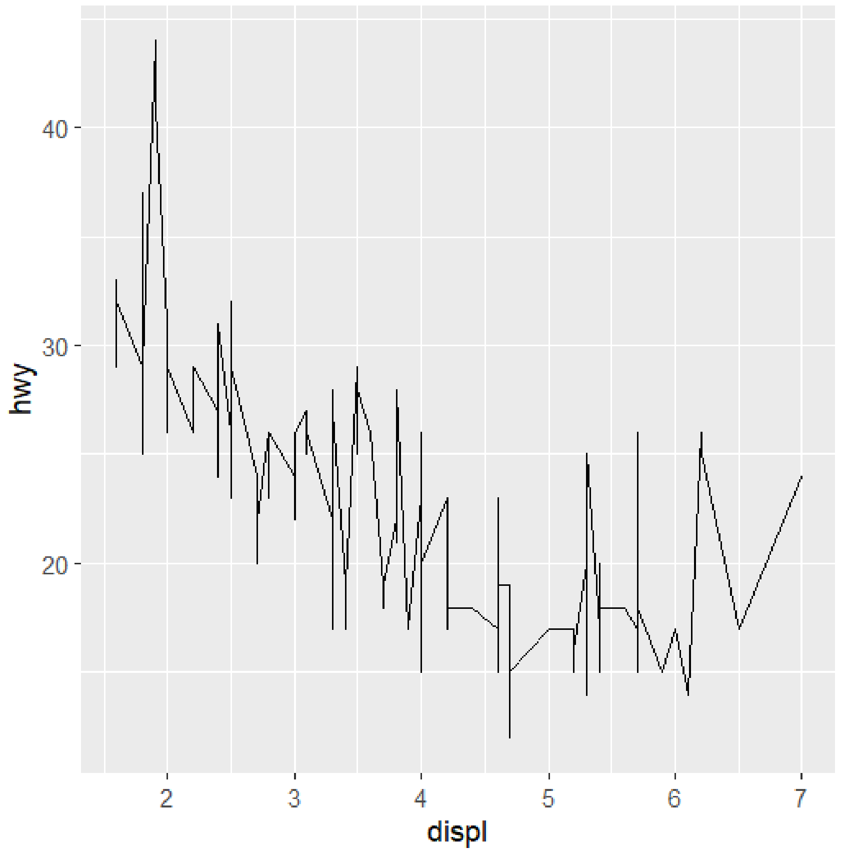
```
ggplot(data = mpg) +
    geom_smooth(mapping = aes(x = displ,
                              y = hwy))
```

```
ggplot(data = mpg) +
    geom_area(mapping = aes(x = displ,
                            y = hwy))
```



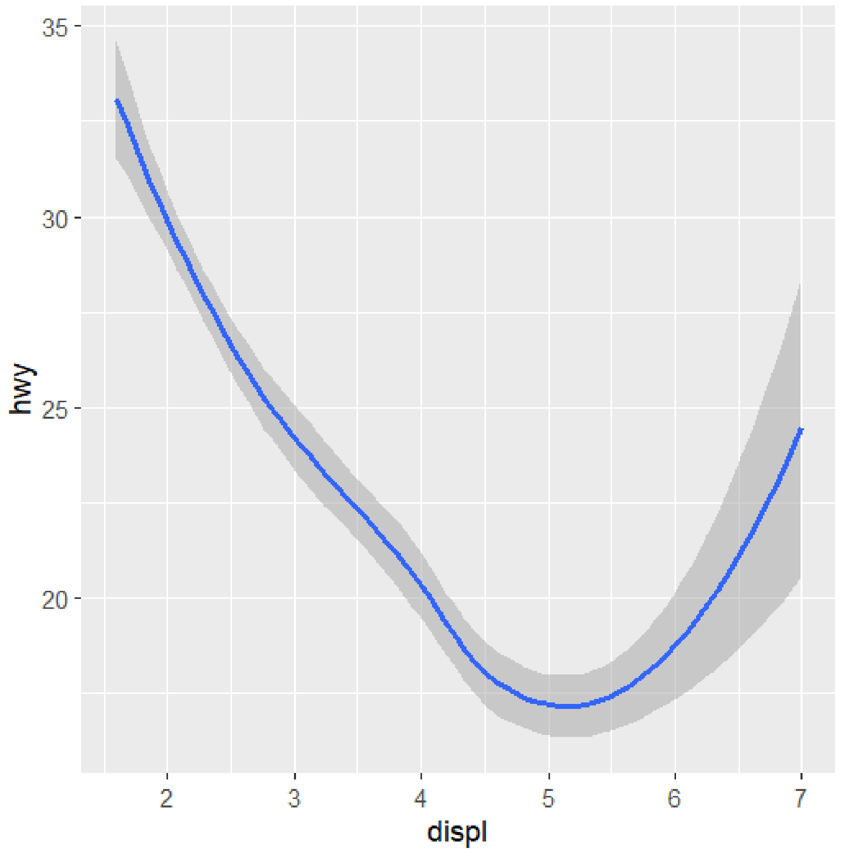
折线图

```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ,
                             y = hwy))

ggplot(data = mpg) +
    geom_line(mapping = aes(x = displ,
                            y = hwy))

ggplot(data = mpg) +
    geom_smooth(mapping = aes(x = displ,
                              y = hwy))

ggplot(data = mpg) +
    geom_area(mapping = aes(x = displ,
                            y = hwy))
```



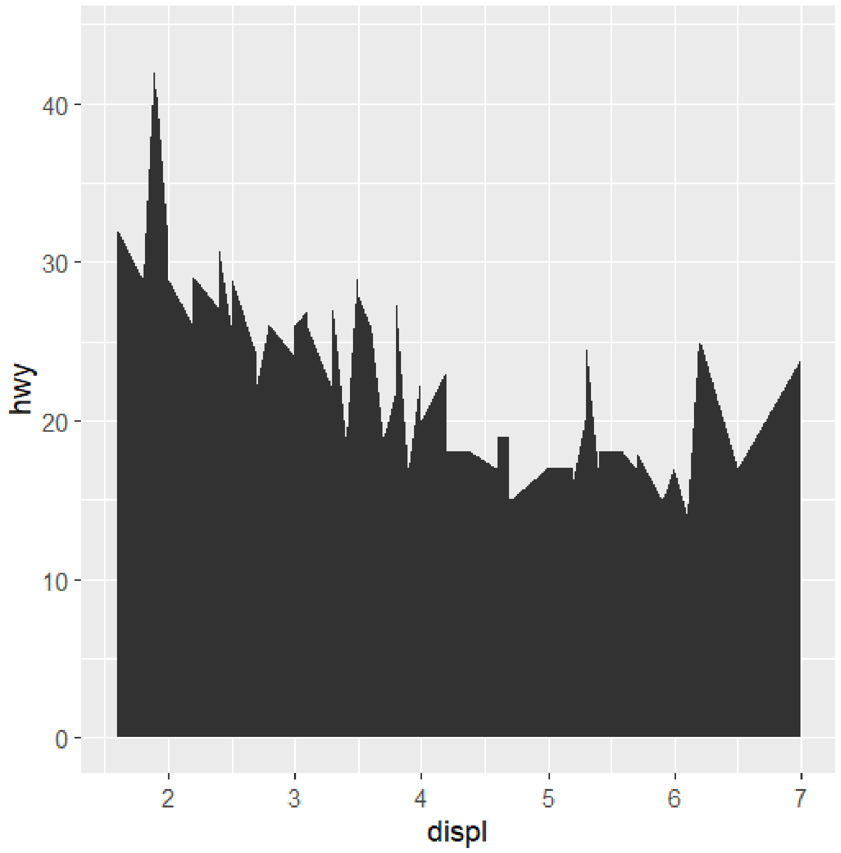
平滑曲线

```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ,
                             y = hwy))

ggplot(data = mpg) +
    geom_line(mapping = aes(x = displ,
                            y = hwy))

ggplot(data = mpg) +
    geom_smooth(mapping = aes(x = displ,
                              y = hwy))

ggplot(data = mpg) +
    geom_area(mapping = aes(x = displ,
                            y = hwy))
```

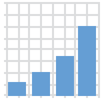


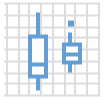
面积图

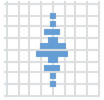
```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ,
                             y = hwy))

ggplot(data = mpg) +
    geom_line(mapping = aes(x = displ,
                            y = hwy))

ggplot(data = mpg) +
    geom_smooth(mapping = aes(x = displ,
                              y = hwy))

ggplot(data = mpg) +
    geom_area(mapping = aes(x = displ,
                            y = hwy))
```

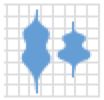


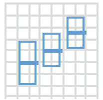
定量+定性

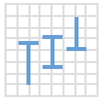

f + geom_col()
x, y, alpha, color, fill, group, linetype, size

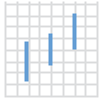

f + geom_boxplot()
x, y, lower, middle, upper, ymax, ymin, alpha,
color, fill, group, linetype, shape, size, weight

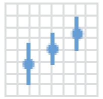

f + geom_dotplot(binaxis = "y", stackdir = “center")
x, y, alpha, color, fill, group


f + geom_violin(scale = “area")
x, y, alpha, color, fill, group, linetype, size, weight


j + geom_crossbar(fatten = 2) - x, y, ymax,
ymin, alpha, color, fill, group, linetype, size

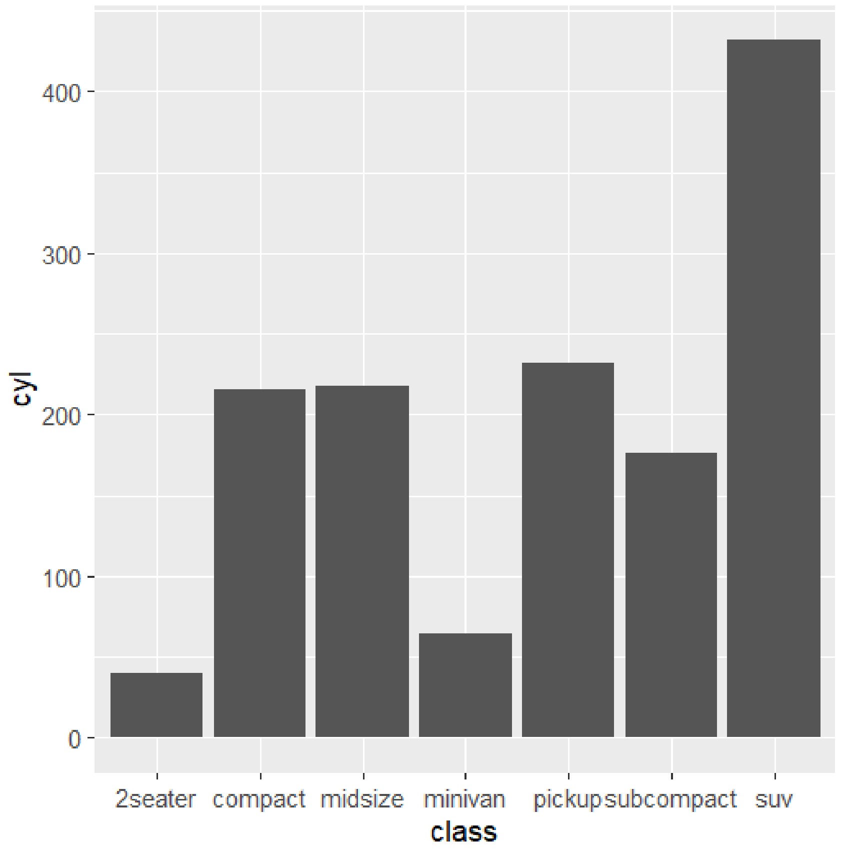

j + geom_errorbar() - x, ymax, ymin,
alpha, color, group, linetype, size, width
Also **geom_errorbarh()**.


j + geom_linerange()
x, ymin, ymax, alpha, color, group, linetype, size


j + geom_pointrange() - x, y, ymin, ymax,
alpha, color, fill, group, linetype, shape, size

柱状图

```
ggplot(data = mpg) +
    geom_col(mapping = aes(x = class,
                           y = cyl))

ggplot(data = mpg) +
    geom_boxplot(mapping = aes(x = class,
                               y = hwy))

ggplot(data = mpg) +
    geom_dotplot(mapping = aes(x = class,
                               y = hwy))

ggplot(data = mpg) +
    geom_violin(mapping = aes(x = class,
                              y = hwy))
```



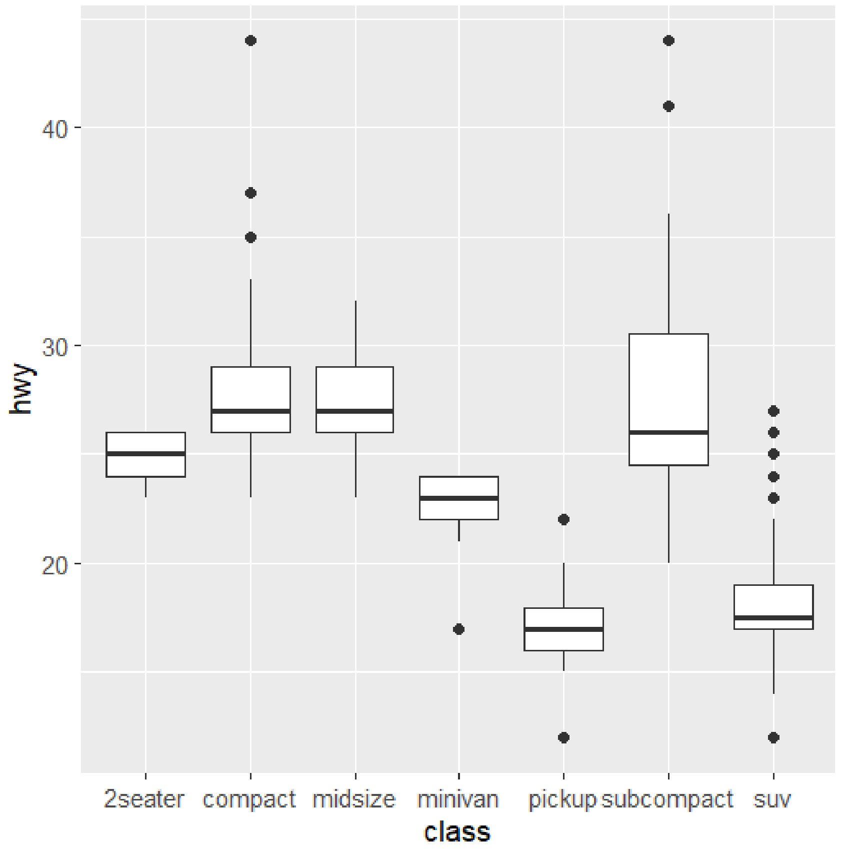
箱式图

```
ggplot(data = mpg) +
    geom_col(mapping = aes(x = class,
                           y = cyl))
```

```
ggplot(data = mpg) +
    geom_boxplot(mapping = aes(x = class,
                               y = hwy))
```

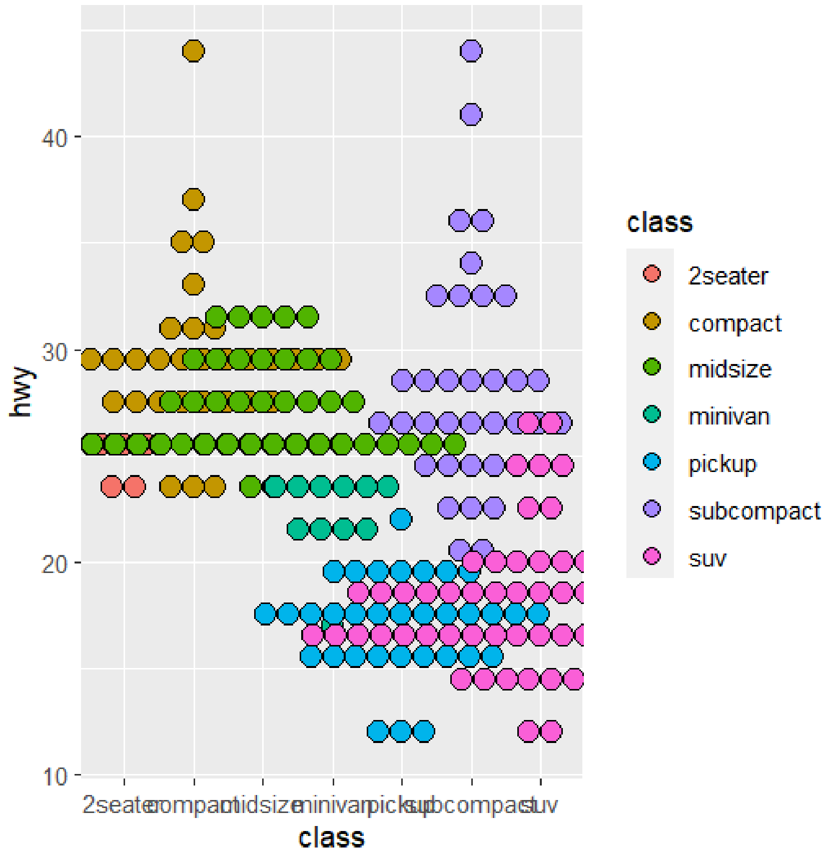
```
ggplot(data = mpg) +
    geom_dotplot(mapping = aes(x = class,
                               y = hwy))
```

```
ggplot(data = mpg) +
    geom_violin(mapping = aes(x = class,
                              y = hwy))
```



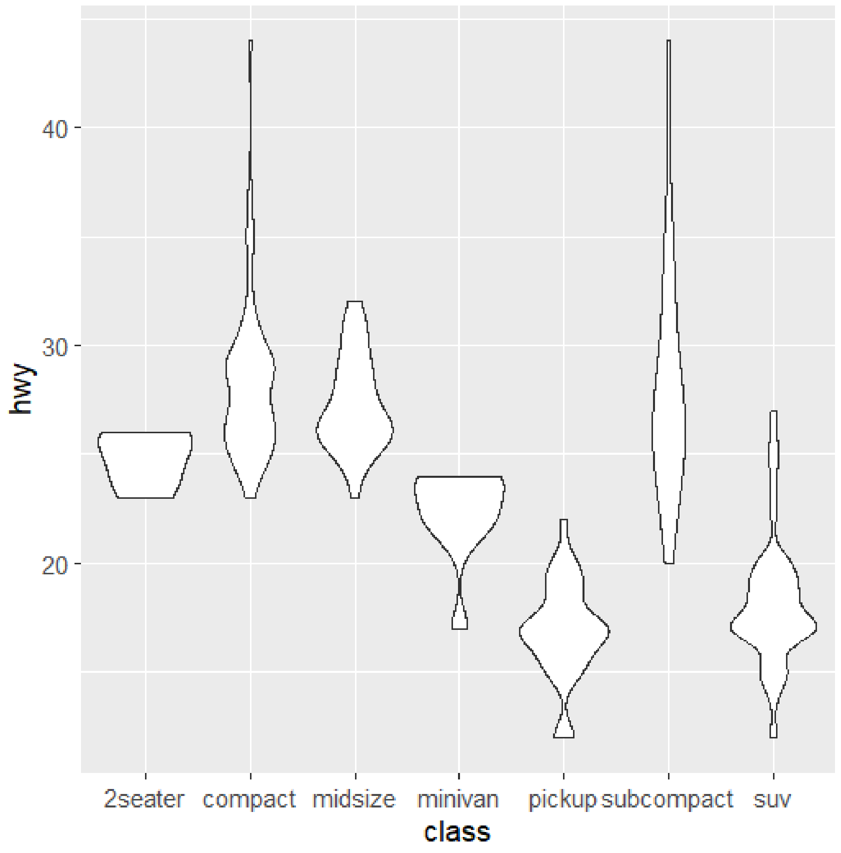
点图

```
ggplot(data = mpg) +
    geom_col(mapping = aes(x = class,
                           y = cyl))

ggplot(data = mpg) +
    geom_boxplot(mapping = aes(x = class,
                               y = hwy))

ggplot(data = mpg) +
    geom_dotplot(mapping = aes(x = class,
                               y = hwy,
                               fill = class),
                binaxis = 'y',
                stackdir = 'center')

ggplot(data = mpg) +
    geom_violin(mapping = aes(x = class,
                              y = hwy))
```



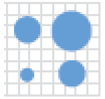
小提琴

```
ggplot(data = mpg) +
    geom_col(mapping = aes(x = class,
                           y = cyl))

ggplot(data = mpg) +
    geom_boxplot(mapping = aes(x = class,
                               y = hwy))

ggplot(data = mpg) +
    geom_dotplot(mapping = aes(x = class,
                               y = hwy,
                               fill = class),
                 binaxis = 'y',
                 stackdir = 'center')

ggplot(data = mpg) +
    geom_violin(mapping = aes(x = class,
                              y = hwy))
```

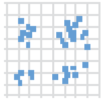


Two variables

定性+定性

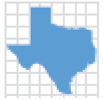


g + geom_count()
x, y, alpha, color, fill, shape, size, stroke



e + geom_jitter(height = 2, width = 2)
x, y, alpha, color, fill, shape, size

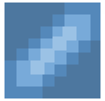
特殊情况 地图



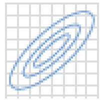
k + geom_map(aes(map_id = state), map = map)
+ expand_limits(x = map$long, y = map$lat)
map_id, alpha, color, fill, linetype, size

常用使用 `sf` 提供的 geom_sf.

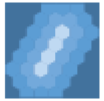
热图



h + geom_bin2d(binwidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight



h + geom_density_2d()
x, y, alpha, color, group, linetype, size



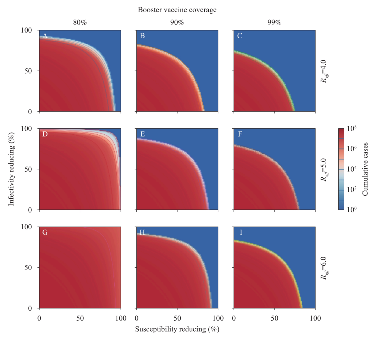
h + geom_hex()
x, y, alpha, color, fill, size

相信你已经掌握了这些简单的图形，接下来我们来看看一些高级的用法。

Three variables

- 定性+定性+定性
- 定性+定性+定量
- 定性+定量+定量
- 定量+定量+定量

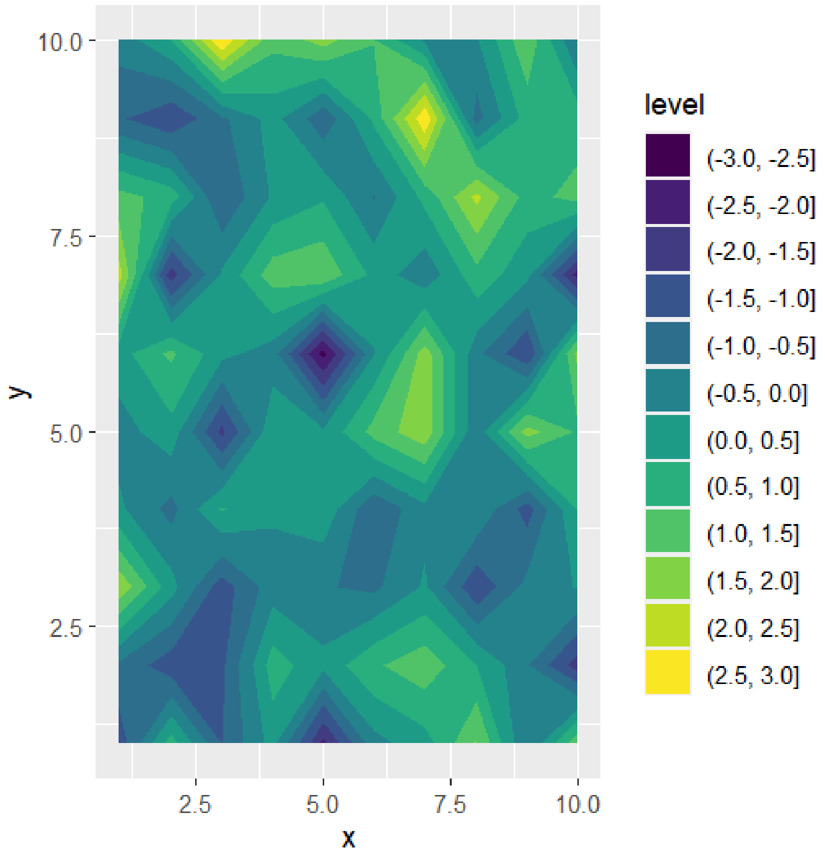
Solution 1: 热图类



Source: China CDC Weekly

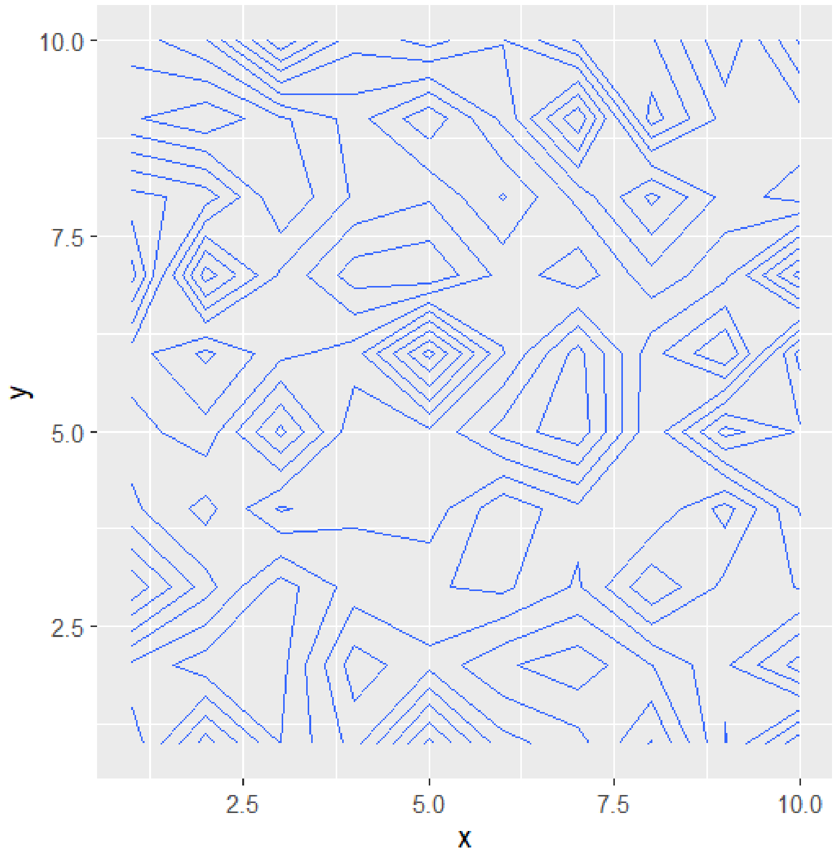
热图类 Test data

```r
data <- data.frame(
  x = rep(1:10, 10),
  y = rep(1:10, each = 10),
  z = rnorm(100)
)

ggplot(data = data) +
    geom_contour_filled(mapping = aes(x = x,
                                      y = y,
                                      z = z))


ggplot(data = data) +
    geom_contour(mapping = aes(x = x,
                               y = y,
                               z = z))


ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))


ggplot(data = data) +
    geom_raster(mapping = aes(x = x,
                              y = y,
                              fill = z))
```



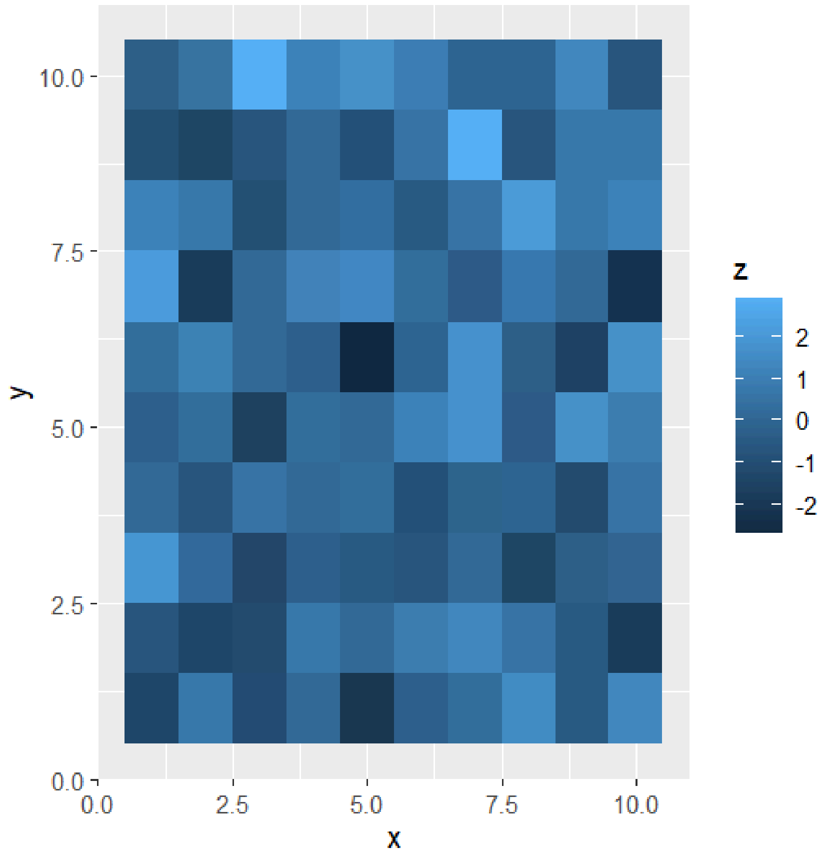
热图类

```r
data <- data.frame(
  x = rep(1:10, 10),
  y = rep(1:10, each = 10),
  z = rnorm(100)
)

ggplot(data = data) +
    geom_contour_filled(mapping = aes(x = x,
                                      y = y,
                                      z = z))

ggplot(data = data) +
    geom_contour(mapping = aes(x = x,
                               y = y,
                               z = z))

ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))

ggplot(data = data) +
    geom_raster(mapping = aes(x = x,
                              y = y,
                              fill = z))
```



热图类

```r
data <- data.frame(
  x = rep(1:10, 10),
  y = rep(1:10, each = 10),
  z = rnorm(100)
)

ggplot(data = data) +
    geom_contour_filled(mapping = aes(x = x,
                                      y = y,
                                      z = z))


ggplot(data = data) +
    geom_contour(mapping = aes(x = x,
                               y = y,
                               z = z))


ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))


ggplot(data = data) +
    geom_raster(mapping = aes(x = x,
                              y = y,
                              fill = z))
```

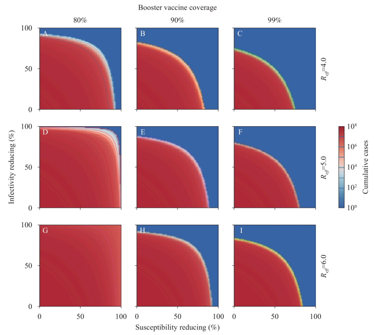


数据集很大使用geom_raster; 数据集较小geom_tile

Three variables

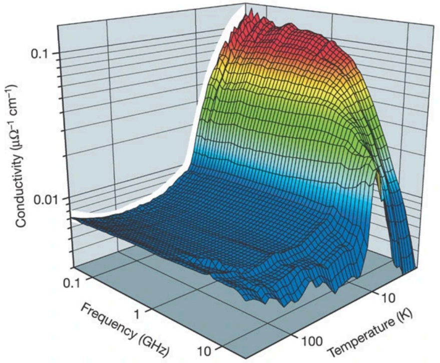
- 定性+定性+定性
- 定性+定性+定量
- 定性+定量+定量
- 定量+定量+定量

Solution 1: 热图类



Source: China CDC Weekly

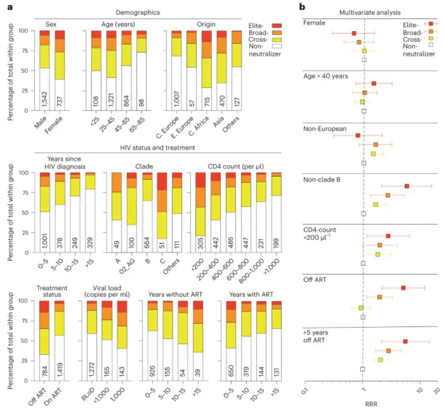
Solution 2: 3D plot



Source: Nature

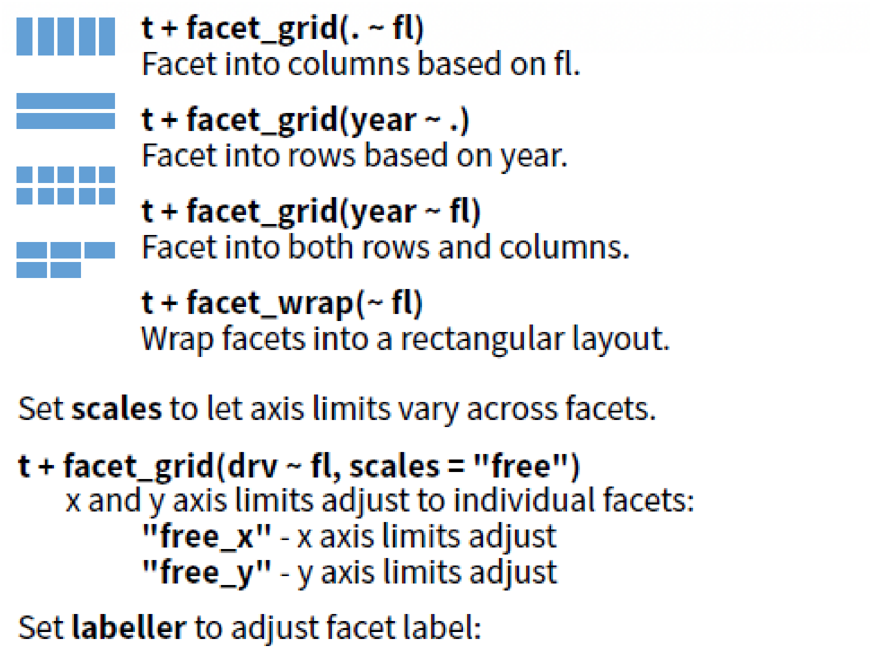
非必要不使用

Solution 3: Facet



Source: Nature

Facets



t + facet_grid(. ~ fl)
Facet into columns based on fl.

t + facet_grid(year ~ .)
Facet into rows based on year.

t + facet_grid(year ~ fl)
Facet into both rows and columns.

t + facet_wrap(~ fl)
Wrap facets into a rectangular layout.

Set **scales** to let axis limits vary across facets.

t + facet_grid(drv ~ fl, scales = "free")
x and y axis limits adjust to individual facets:
"free_x" - x axis limits adjust
"free_y" - y axis limits adjust

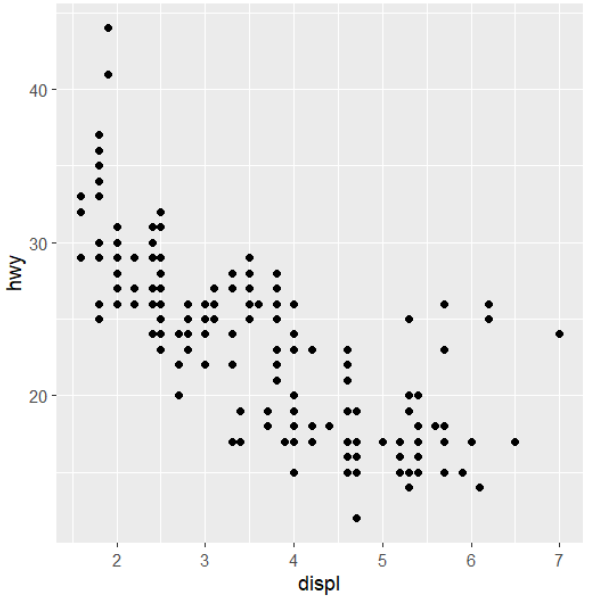
Set **labeller** to adjust facet label:

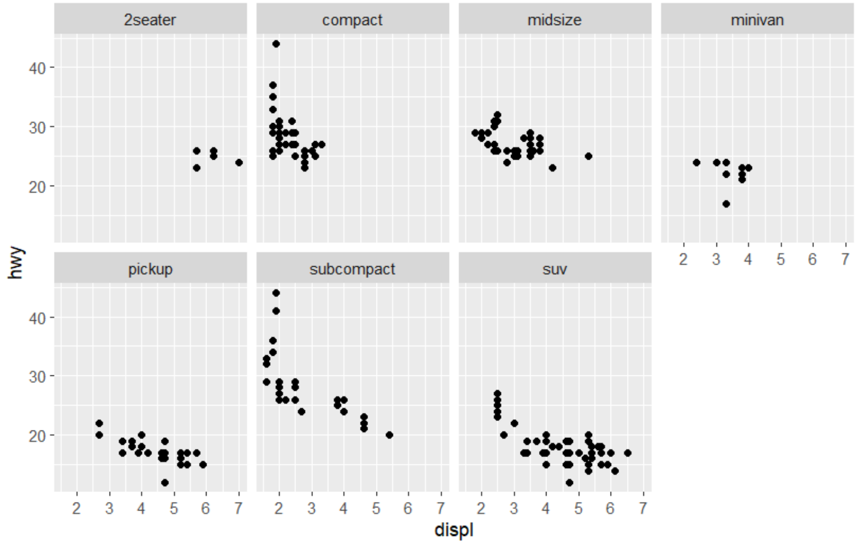
一个图拆分为多个子图，每个子图展示不同的数据。

Why facet?

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                           y = hwy))
```

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                           y = hwy))+
  facet_wrap(.~class,nrow = 2)
```

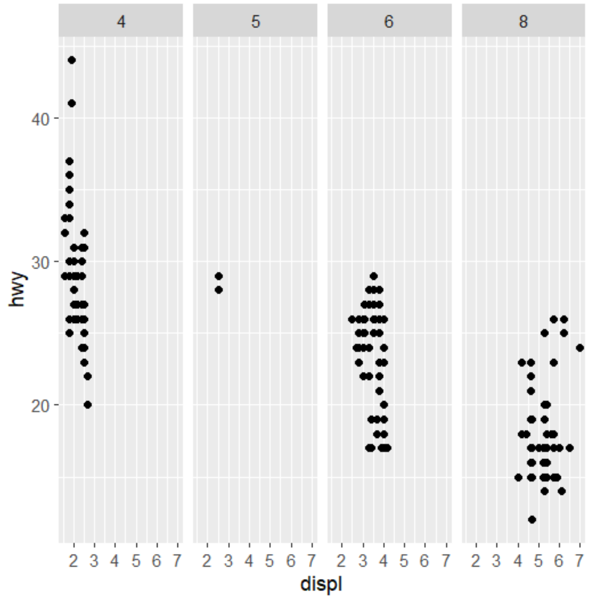


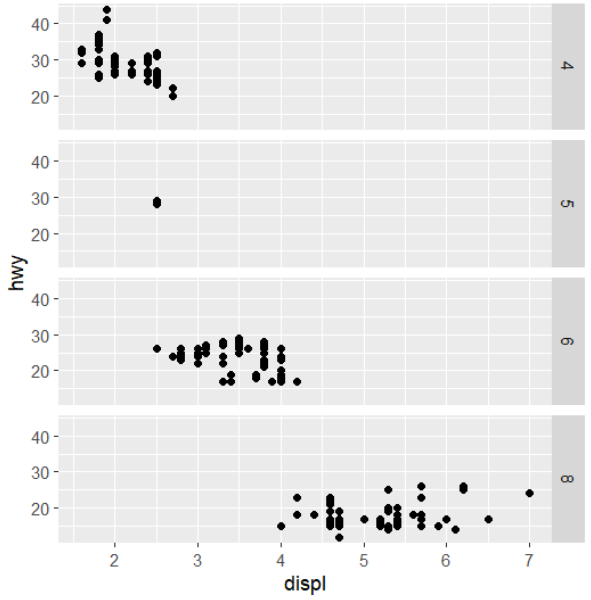


facet_grid

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                           y = hwy))+
  facet_grid(.~cyl)
```

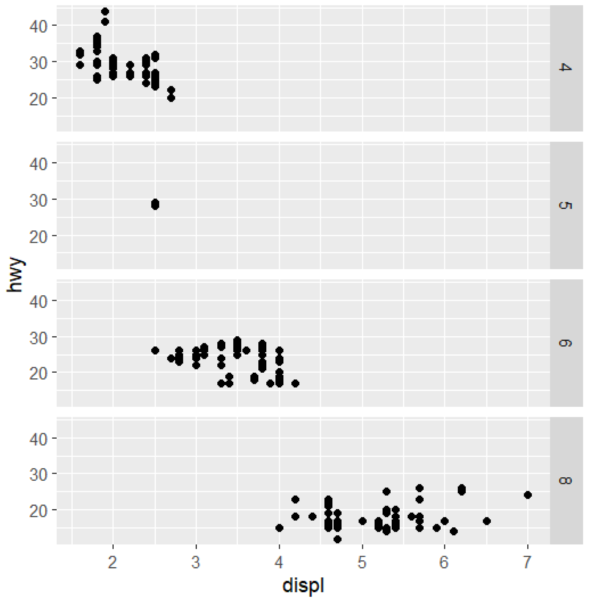
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                           y = hwy))+
  facet_grid(cyl~.)
```



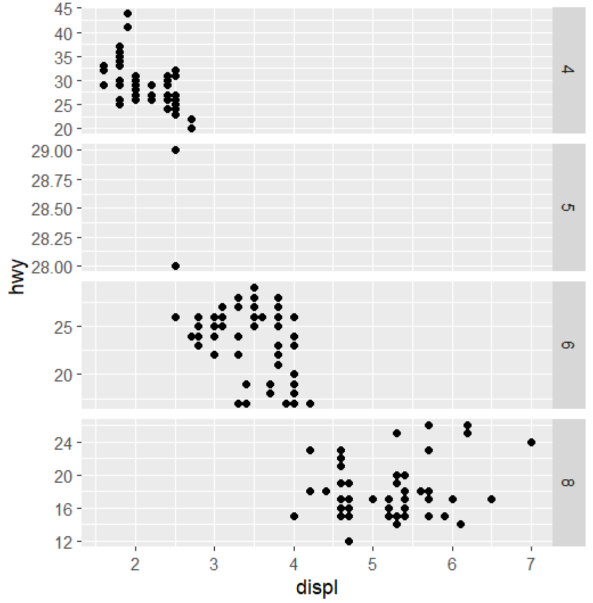


facet_grid

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                           y = hwy))+
  facet_grid(cyl~.)
```



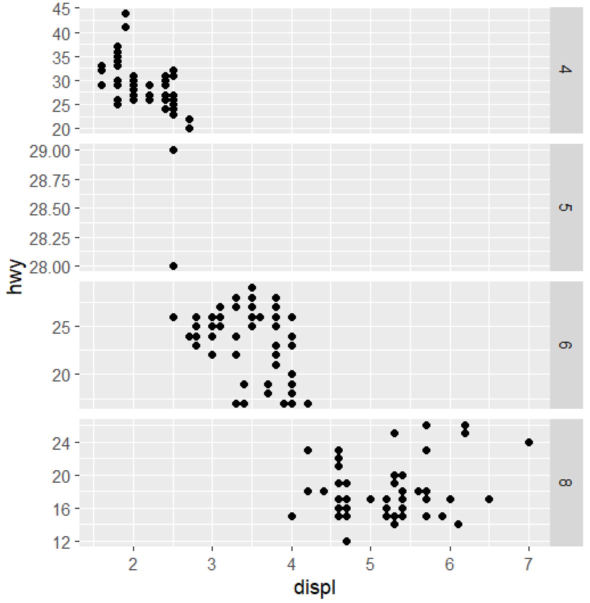
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                           y = hwy))+
  facet_grid(cyl~.,
             scales = 'free')
```

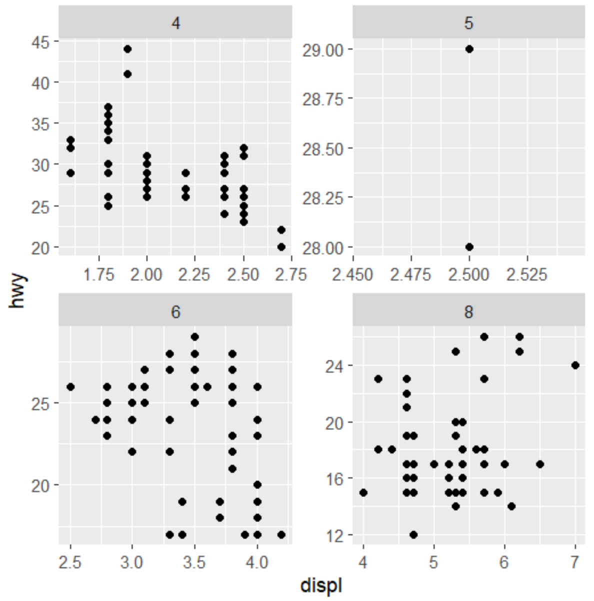


facet_wrap

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                            y = hwy))+
  facet_grid(cyl~.,
             scales = 'free')
```

```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ,
                            y = hwy))+
  facet_wrap(cyl~.,
             scales = 'free')
```





summary

如果需要任意堆积，可以使用`` `facet_wrap()` ``。如果要按照某个变量堆积，可以使用`` `facet_grid()` ``。

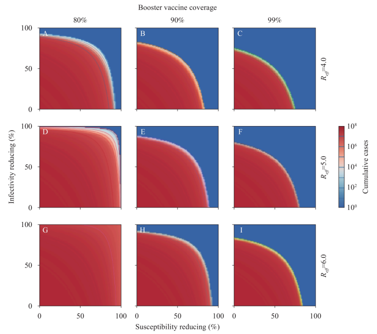
对于`` `facet_wrap()` ``，可以使用`` `nrow` ``和`` `ncol` ``来控制子图的行数和列数。

更多信息可以通过`` `?facet_wrap` ``和`` `?facet_grid` ``查看。

Three variables

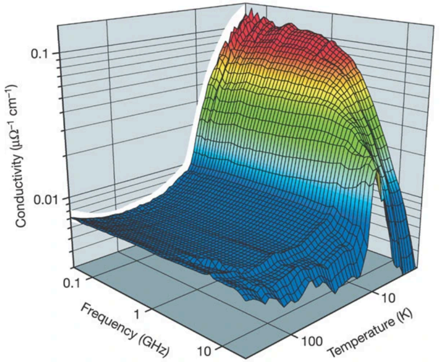
- 定性+定性+定性
- 定性+定性+定量
- 定性+定量+定量
- 定量+定量+定量

Solution 1: 热图类



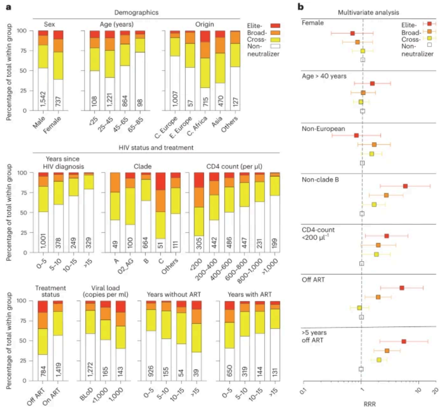
Source: China CDC Weekly

Solution 2: 3D plot



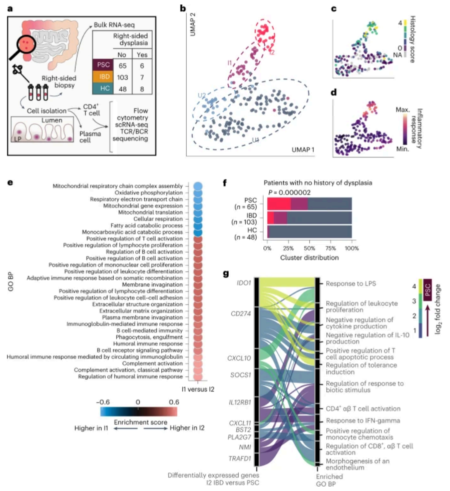
Source: Nature

Solution 3: Facet



Source: Nature

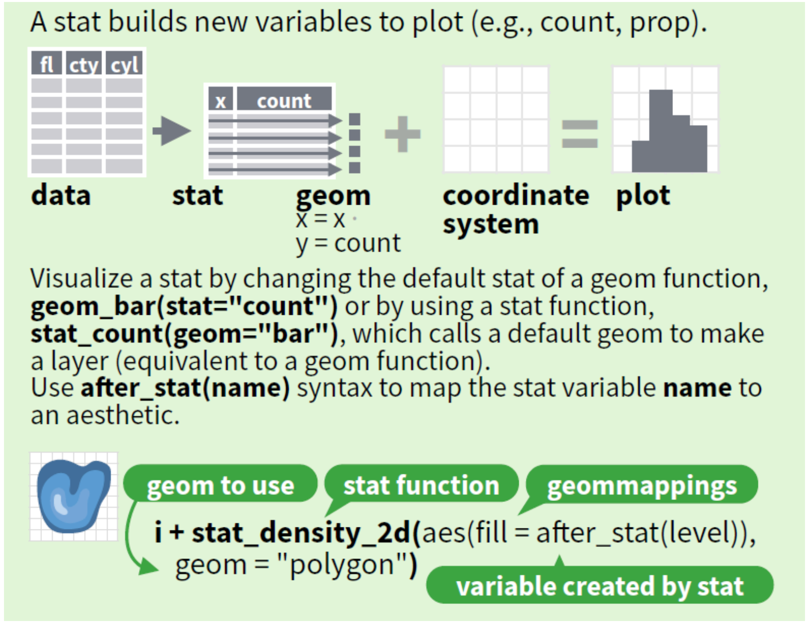
Solution 4: Aesthetics: color, shape, size...



Source: Nature

Statistical models

我更倾向于先把数据处理好，再作图。



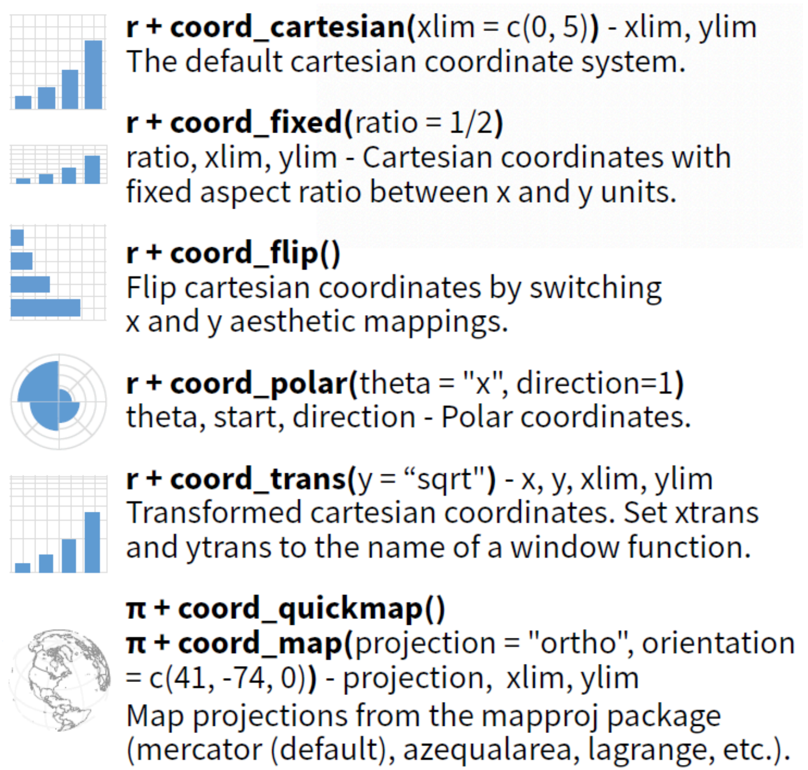
A stat builds new variables to plot (e.g., count, prop).

Visualize a stat by changing the default stat of a geom function, **geom_bar(stat="count")** or by using a stat function, **stat_count(geom="bar")**, which calls a default geom to make a layer (equivalent to a geom function).
Use **after_stat(name)** syntax to map the stat variable **name** to an aesthetic.

i + stat_density_2d(aes(fill = after_stat(level)),
geom = "polygon"**)**

Space

Coordinate systems



r + coord_cartesian(xlim = c(0, 5)**)** - xlim, ylim
The default cartesian coordinate system.

r + coord_fixed(ratio = 1/2**)**
ratio, xlim, ylim - Cartesian coordinates with fixed aspect ratio between x and y units.

r + coord_flip()
Flip cartesian coordinates by switching x and y aesthetic mappings.

r + coord_polar(theta = "x", direction=1**)**
theta, start, direction - Polar coordinates.

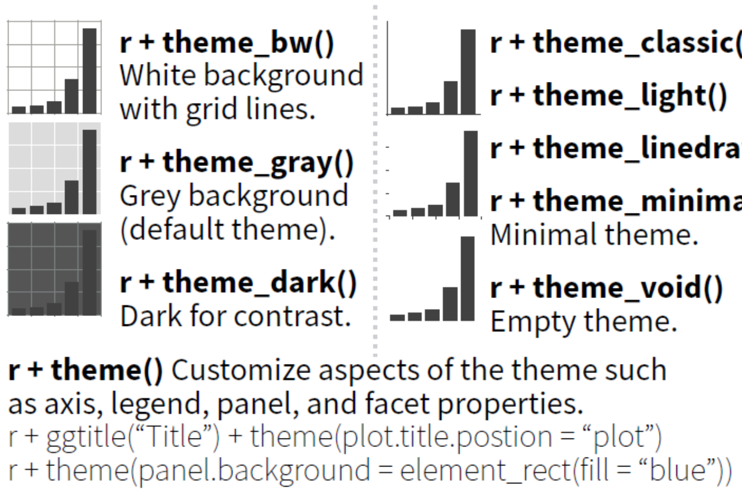
r + coord_trans(y = “sqrt"**)** - x, y, xlim, ylim
Transformed cartesian coordinates. Set xtrans and ytrans to the name of a window function.

π + coord_quickmap()
π + coord_map(projection = "ortho", orientation = c(41, -74, 0)**)** - projection, xlim, ylim
Map projections from the mapproj package (mercator (default), azequalarea, lagrange, etc.).

- coord_cartesian: 用于裁剪图片修正显示范围，常用于制作字母图。
- coord_flip: 用于翻转坐标轴。
- coord_fixed: 用于固定坐标轴比例，常用于绘制热图和地图。
- coord_polar 和 coord_quickmap：使用较少。

Theme

设置背景和坐标轴



r + theme_bw() White background with grid lines.

r + theme_gray() Grey background (default theme).

r + theme_dark() Dark for contrast.

r + theme_classic()

r + theme_light()

r + theme_linedraw()

r + theme_minimal() Minimal theme.

r + theme_void() Empty theme.

r + theme() Customize aspects of the theme such as axis, legend, panel, and facet properties.
r + ggtitle("Title") + theme(plot.title.postion = "plot")
r + theme(panel.background = element_rect(fill = "blue"))

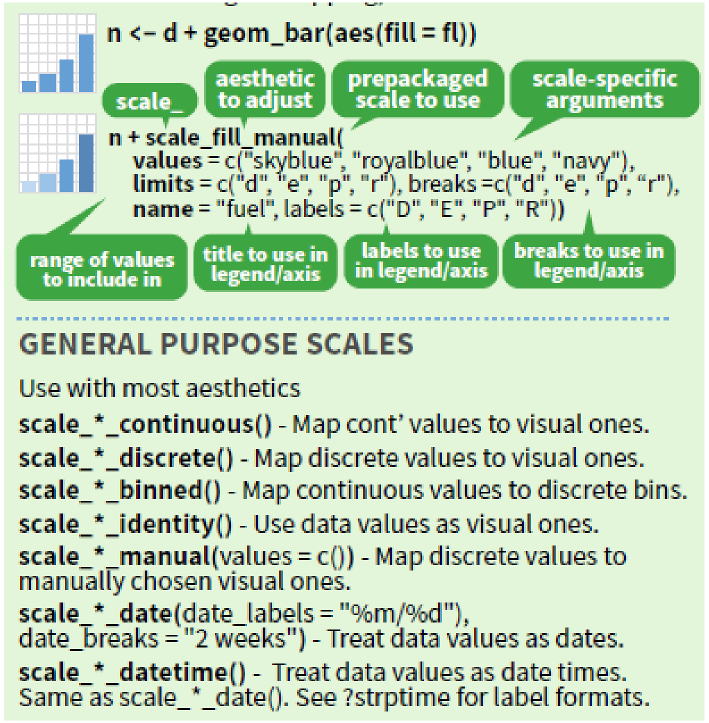
```
library(ggplot2)
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ,
                             y = hwy,
                             colour = class))+
    coord_equal(ratio = 1/5)+
    facet_wrap(.~cyl,nrow = 1)+
    scale_color_viridis_d()+
    theme_bw()+
    theme(panel.grid.major = element_blank(),
          plot.title = element_text(size = 16),
          axis.line = element_line(size = 1),
          panel.background = element_rect(fill = 'transpare
          plot.margin = margin(5,5, 5,5))+
    labs(title = 'A',
         colour = 'Class of car')
```

主题模板 + 主题自定义

□♀ **Question:** 这两部分能否调转过来？

What else?

Scales



n <- d + geom_bar(aes(fill = fl))

scale_ | aesthetic to adjust | prepackaged scale to use | scale-specific arguments

```
n + scale_fill_manual(
    values = c("skyblue", "royalblue", "blue", "navy"),
    limits = c("d", "e", "p", "r"), breaks =c("d", "e", "p", "r"),
    name = "fuel", labels = c("D", "E", "P", "R"))
```

range of values to include in | title to use in legend/axis | labels to use in legend/axis | breaks to use in legend/axis

GENERAL PURPOSE SCALES

Use with most aesthetics

scale_*_continuous() - Map cont' values to visual ones.

scale_*_discrete() - Map discrete values to visual ones.

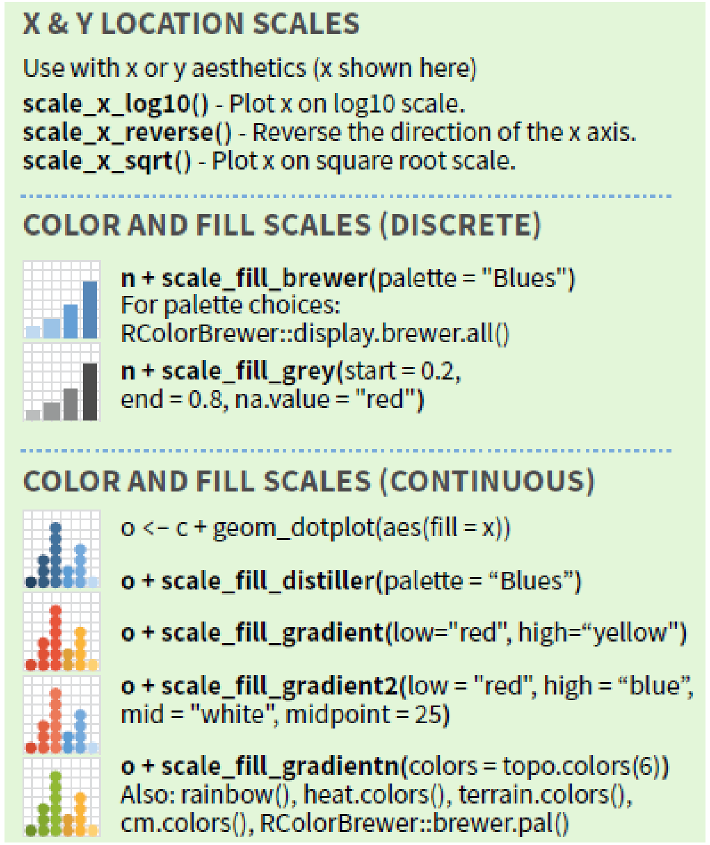
scale_*_binned() - Map continuous values to discrete bins.

scale_*_identity() - Use data values as visual ones.

scale_*_manual(values = c()) - Map discrete values to manually chosen visual ones.

scale_*_date(date_labels = "%m/%d"), date_breaks = "2 weeks") - Treat data values as dates.

scale_*_datetime() - Treat data values as date times. Same as scale_*_date(). See ?strptime for label formats.



X & Y LOCATION SCALES

Use with x or y aesthetics (x shown here)

scale_x_log10() - Plot x on log10 scale.
scale_x_reverse() - Reverse the direction of the x axis.
scale_x_sqrt() - Plot x on square root scale.

COLOR AND FILL SCALES (DISCRETE)

n + scale_fill_brewer(palette = "Blues")
For palette choices:
RColorBrewer::display.brewer.all()

n + scale_fill_grey(start = 0.2, end = 0.8, na.value = "red")

COLOR AND FILL SCALES (CONTINUOUS)

o <- c + geom_dotplot(aes(fill = x))

o + scale_fill_distiller(palette = "Blues")

o + scale_fill_gradient(low="red", high="yellow")

o + scale_fill_gradient2(low = "red", high = "blue", mid = "white", midpoint = 25)

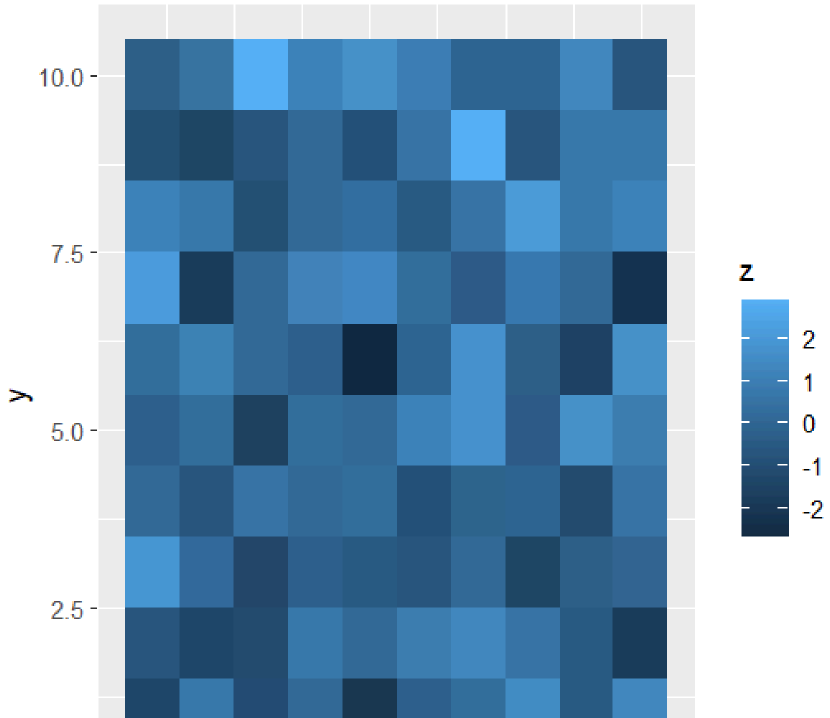
o + scale_fill_gradientn(colors = topo.colors(6))
Also: rainbow(), heat.colors(), terrain.colors(), cm.colors(), RColorBrewer::brewer.pal()

Scales of axis

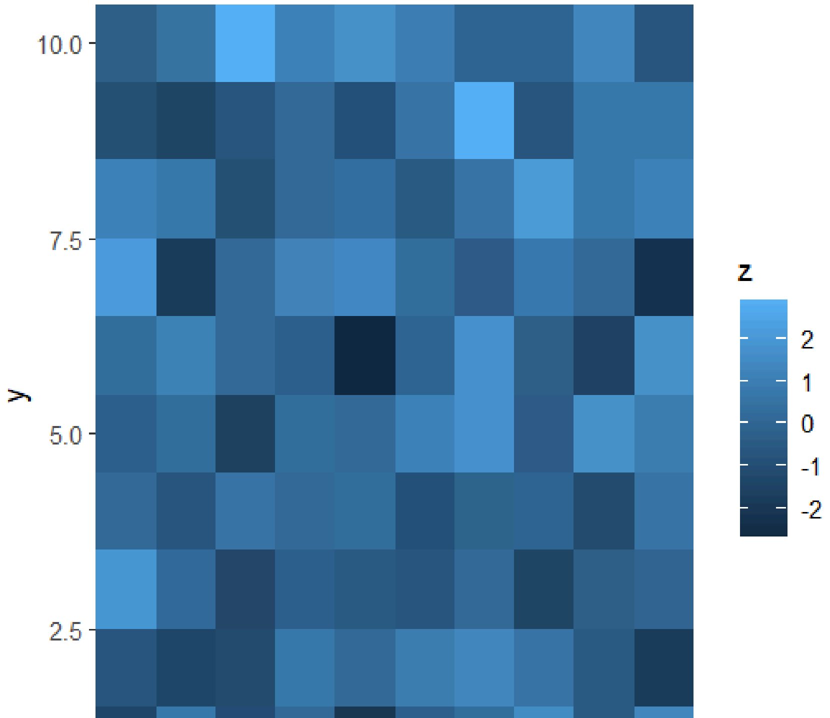
根据变量类型设置坐标轴 (以 x 轴为例)

Variable type	R class	Scale function
定量变量	numeric/integer	scale_x_continuous()
定性变量	factor/character	scale_x_discrete()
日期/时间	date/POSIXct	scale_x_date()

```
ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))+
    scale_x_continuous(expand = c(0,0))+
    scale_y_continuous(expand = c(0,0))
```



```
ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))+
    scale_x_continuous(expand = c(0,0))+
    scale_y_continuous(expand = c(0,0))
```



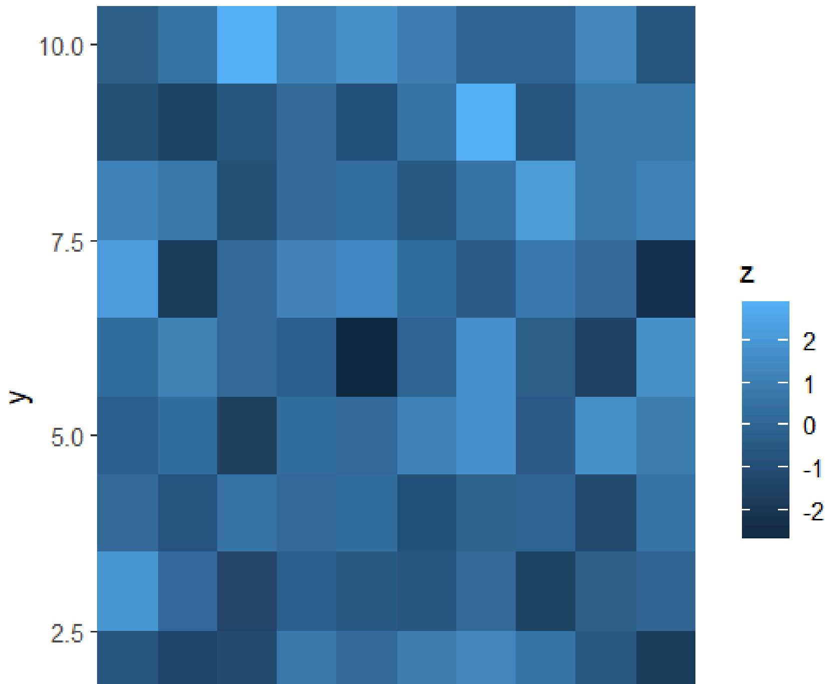
Scales of guides

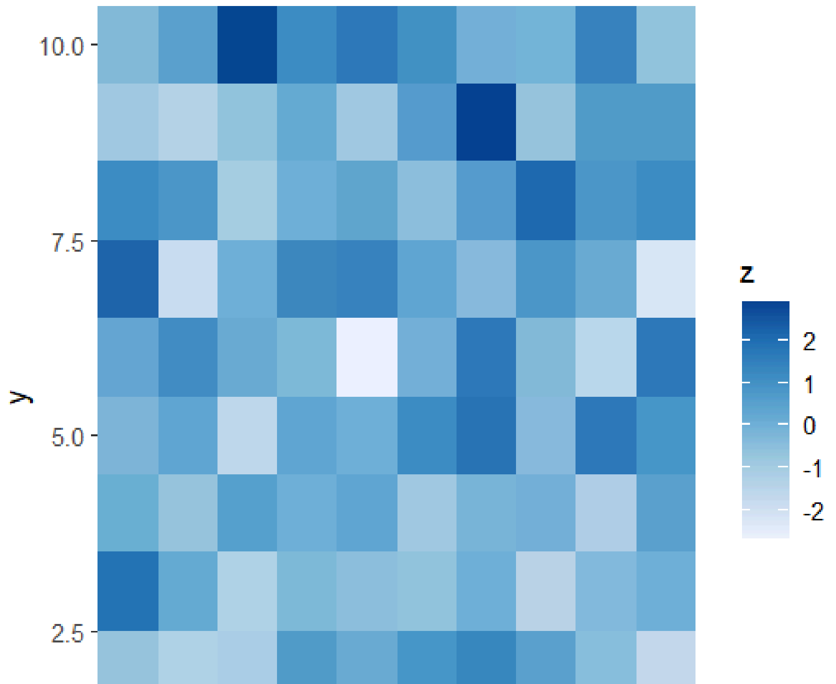
根据变量类型设置图例 (以 fill 轴为例)

Variable type	R class	Scale function
定量变量	numeric/integer	scale_fill_distiller()
		scale_fill_gradient()
		scale_fill_gradient2()
		scale_fill_gradientn()
定性变量	factor/character	scale_fill_brewer()
		scale_fill_manual()

```r
ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))+
    scale_x_continuous(expand = c(0,0))+
    scale_y_continuous(expand = c(0,0))
```

```r
ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))+
    scale_x_continuous(expand = c(0,0))+
    scale_y_continuous(expand = c(0,0))+
    scale_fill_distiller(palette = 'Blues', direction = 1)
```





Labels and legends

t + labs(x = "New x axis label", **y** = "New y axis label",
title ="Add a title above the plot",
subtitle = "Add a subtitle below title",
caption = "Add a caption below plot",
alt = "Add alt text to the plot",
<AES> = "New **<AES>** legend title"**)**

t + annotate(geom = "text", x = 8, y = 9, label = “A"**)**
Places a geom with manually selected aesthetics.

p + guides(x = guide_axis(n.dodge = 2)**)** Avoid crowded or overlapping labels with guide_axis(n.dodge or angle).

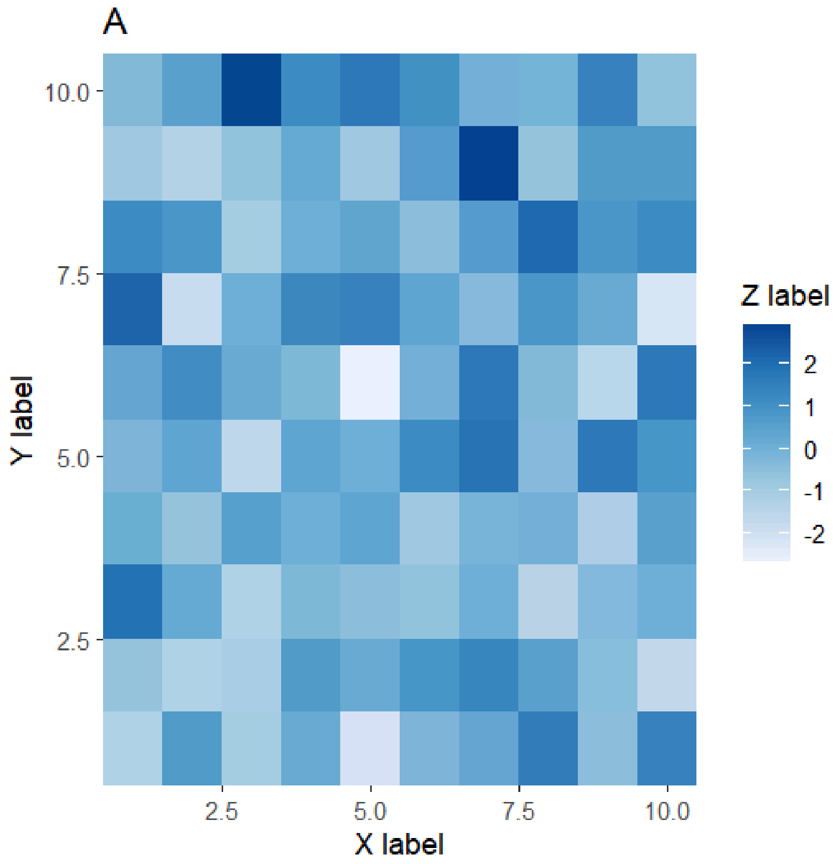
n + guides(fill = “none"**)** Set legend type for each aesthetic: colorbar, legend, or none (no legend).

n + theme(legend.position = "bottom"**)**
Place legend at "bottom", "top", "left", or “right”.

n + scale_fill_discrete(name = "Title",
labels = c("A", "B", "C", "D", "E")**)**
Set legend title and labels with a scale function.

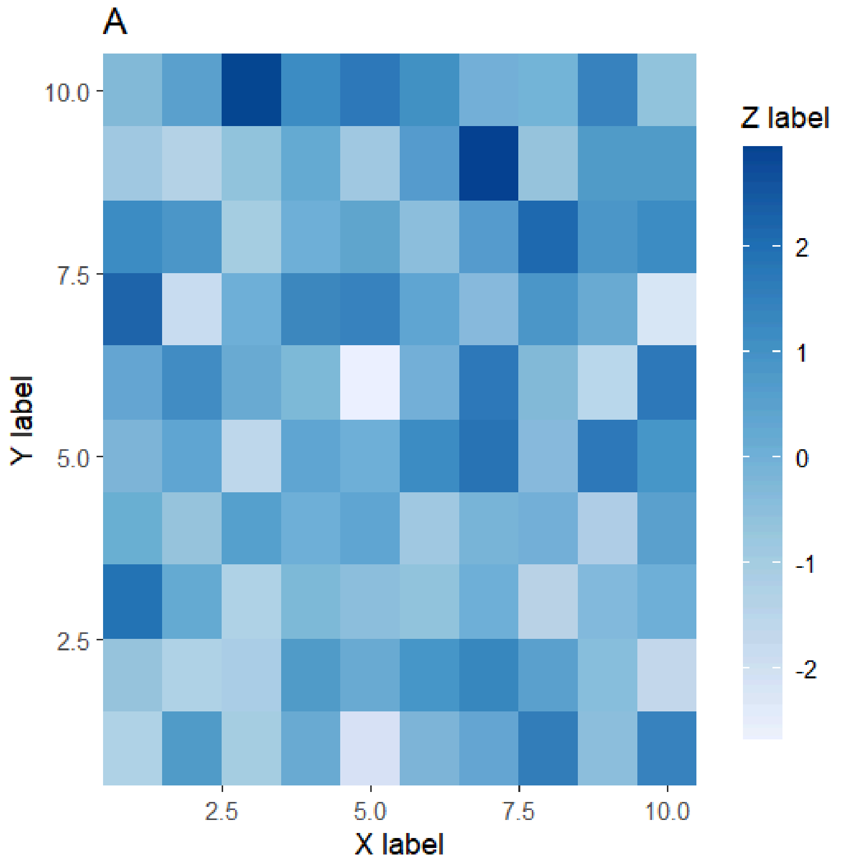
Labels

```
ggplot(data = data) +
    geom_tile(mapping = aes(x = x,
                            y = y,
                            fill = z))+
    scale_x_continuous(expand = c(0,0))+
    scale_y_continuous(expand = c(0,0))+
    scale_fill_distiller(palette = 'Blues', direction = 1)+
    labs(title = 'A',
         subtitle = NULL,
         x = 'X label',
         y = 'Y label',
         fill = 'Z label')
```

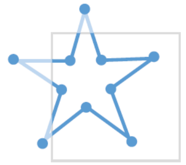


guides

```
ggplot(data = data) +
  geom_tile(mapping = aes(x = x, y = y, fill = z)) +
  scale_x_continuous(expand = c(0,0)) +
  scale_y_continuous(expand = c(0,0)) +
  scale_fill_distiller(palette = 'Blues', direction = 1) +
  labs(title = 'A',
       subtitle = NULL,
       x = 'X label',
       y = 'Y label',
       fill = 'Z label') +
  guides(fill = guide_colorbar(
      barwidth = 1,
      barheight = 15
  ))
```



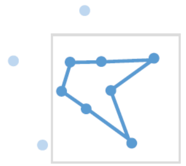
Zooming



Without clipping (preferred):

t + coord_cartesian(xlim = c(0, 100), ylim = c(10, 20)**)**

With clipping (removes unseen data points):



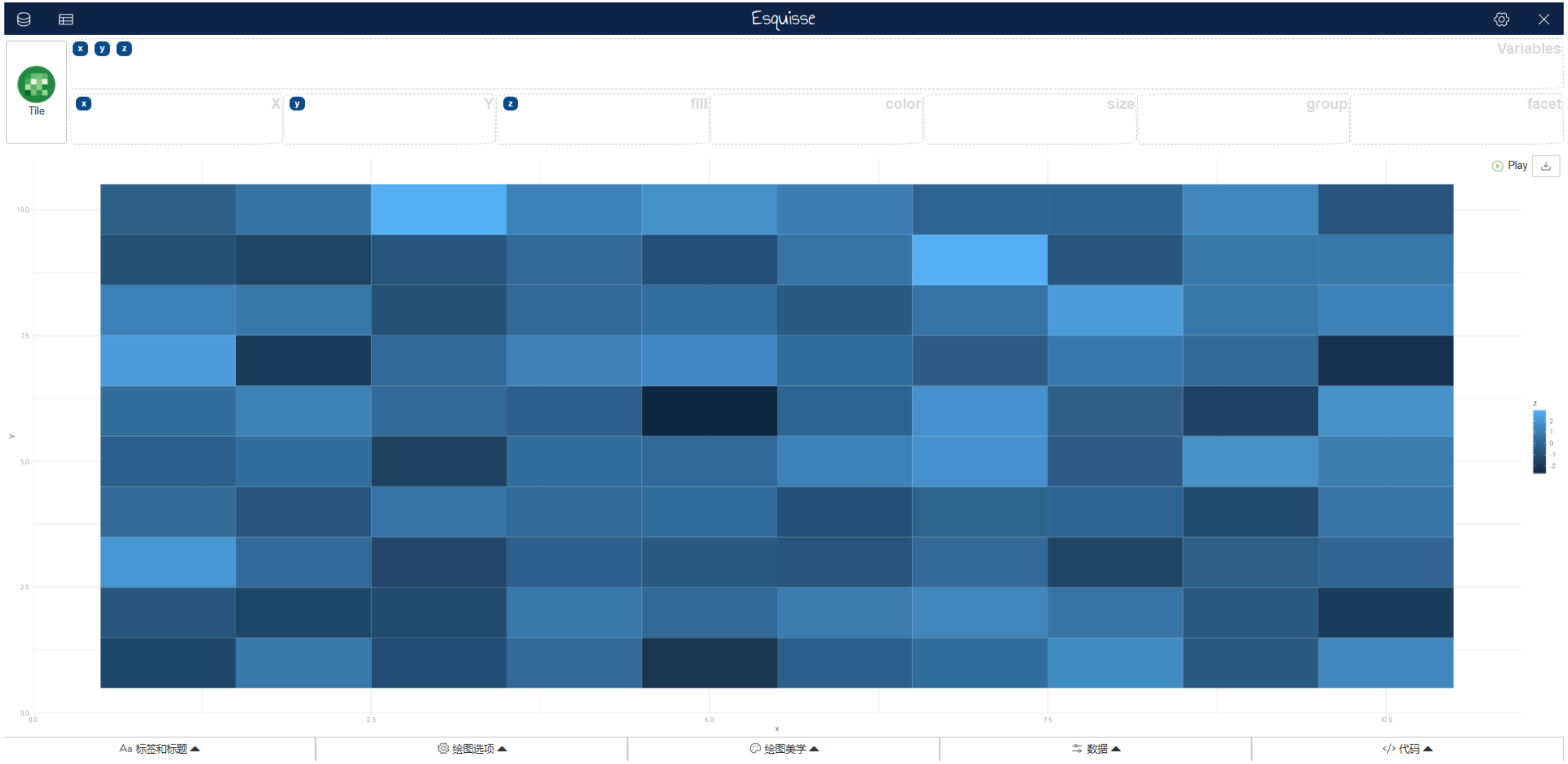
t + xlim(0, 100**) + ylim(**10, 20**)**

t + scale_x_continuous(limits = c(0, 100)**) +**
scale_y_continuous(limits = c(0, 100)**)**

Save plot

```r
# png
ggsave('plot.png', width = 10, height = 10, dpi = 300)
# pdf
ggsave('plot.pdf', width = 10, height = 10, device = cairo_pdf)
# pdf & 中文
ggsave('plot.pdf', width = 10, height = 10, device = cairo_pdf, family = 'GB1')
```

Using esquisse



Epidemiologist Toolbox

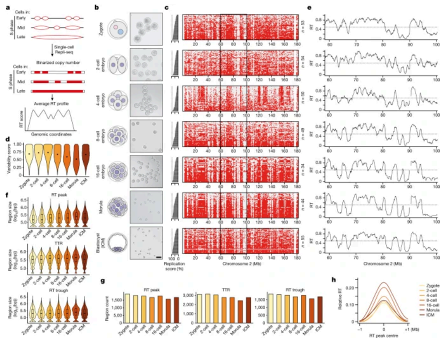
How to learn?

前人的工作+你的工作=最佳的统计图形

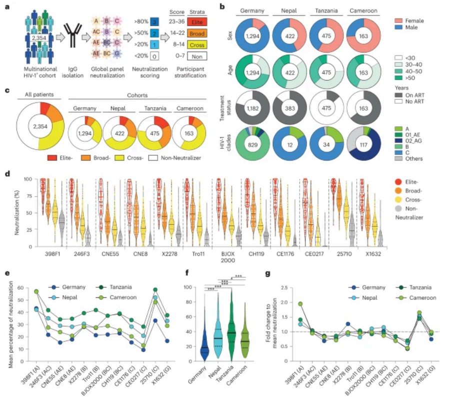
天才是极少数的，一个统计图形的设计需要经过大量的尝试，所以我们需要学习前人的工作。

Source: Science Advances

Source: Science Advances



Source: Nature



Source: Nature

Learn More

PDF download

GitHub Repository

My Resume