

Road to ggplot2再入門

第91回 Tokyo.R

発表者：das Kino (@kyn02666)

「宇宙本」の宣伝



【主な内容】

- RStudioの基本機能
- Webスクレイピング（Web上のデータの収集）
- データの整形（tidyr / dplyr）
- データ可視化（ggplot2）
- レポートニング（RMarkdown）

おかげさまで、2021/6/2に、シン・宇宙本（第2版）が出ます！！

【変更点】

- 最新の内容を反映
- 文字列操作（stringr）、日付・時刻データ（lubridate）に関する付録を、計50ページ近く追加

さて...



初心者向けに
ggplot2の話をしてほしいと
ご依頼いただいたわけですが...

これ以上の資料ありますか？！

ご清聴ありがとうございました！！

<https://speakerdeck.com/yutannihilation/ggplot2zai-ru-men>

ggplot2 再入門

Hiroaki Yutani
(@yutannihilation)

1

こんな人を想定しています

- ggplot2を多少さわったことはある
- なんとなくグラフは描けるけど、ggplot2のコードの意味はあまり理解できない
- 人が描いたggplot2のグラフを少しいじりたいけど、どのコードを変えればイメージ通りになるかわからない

3

なるほど！

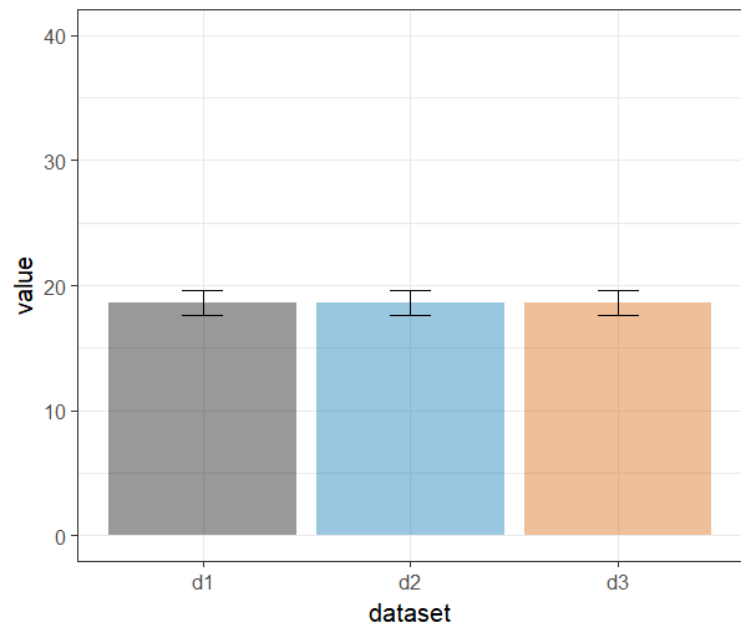
本発表の概要

- 想定読者
 - 文字通り初めて ggplot2パッケージを用いて可視化したい方々
- 本発表が目標とすること
 - 「ggplot2再入門」を理解するための、入り口まで案内すること
 - 1つ／2つの量的変数なら、なんとなく可視化できるようになること

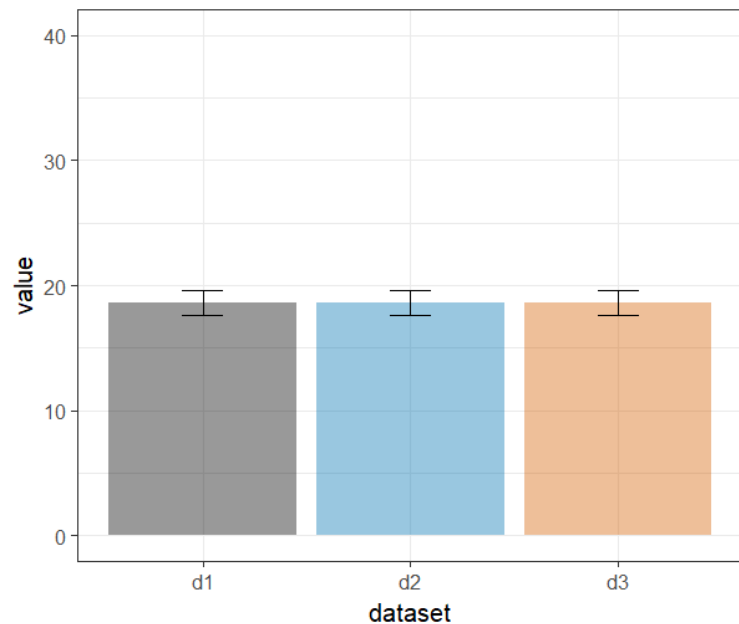
※離散変数が含まれる可視化はクセがあるので、「宇宙本」を参照してください！

- 本発表が目標としないこと
 - どんな可視化もお手の物になること

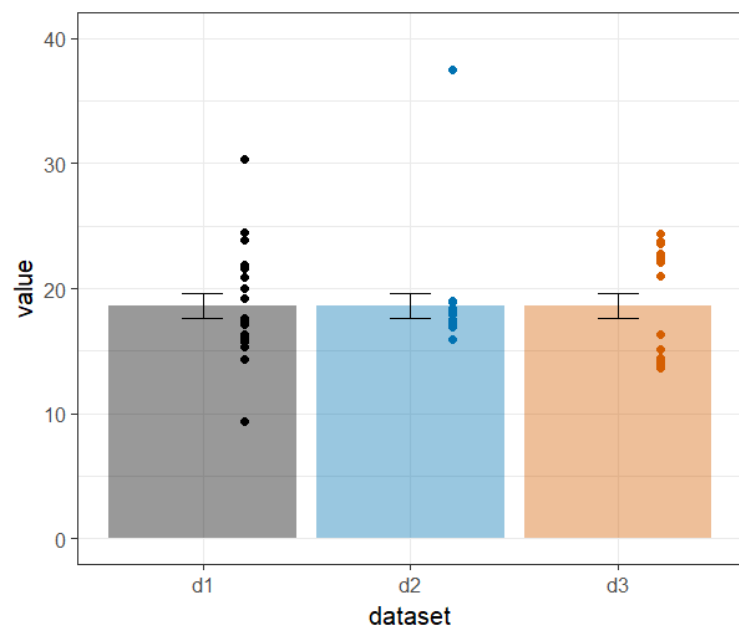
可視化の重要性



棒グラフで平均値を、
エラーバーで散布度（例：標準誤差）を
表した、よく見るグラフ



棒グラフで平均値を、
エラーバーで散布度（例：標準誤差）を
表した、よく見るグラフ



ローデータは、こうかも？！

PERSPECTIVE

Beyond Bar and Line Graphs: Time for a New Data Presentation Paradigm

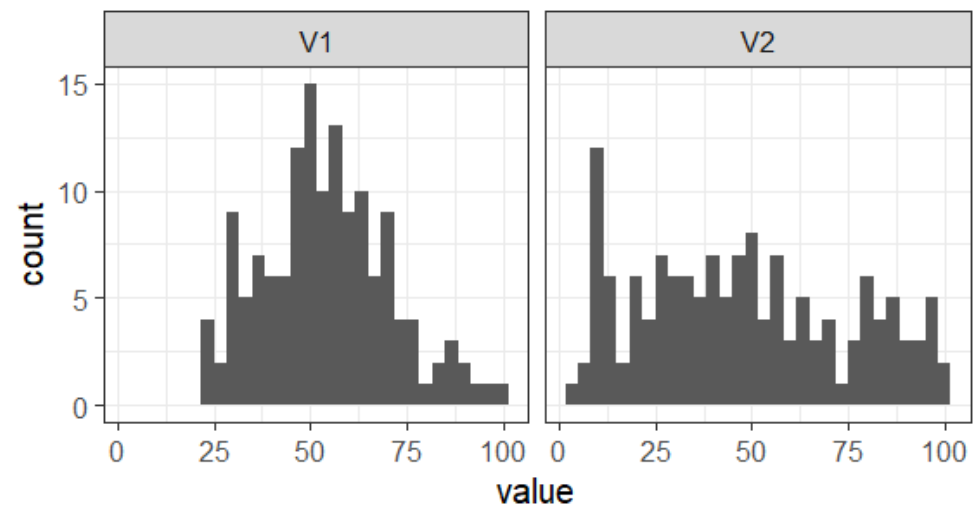
Tracey L. Weissgerber^{1*}, Natasa M. Milic^{1,2}, Stacey J. Winham³, Vesna D. Garovic¹

1 Division of Nephrology & Hypertension, Mayo Clinic, Rochester, Minnesota, United States of America,

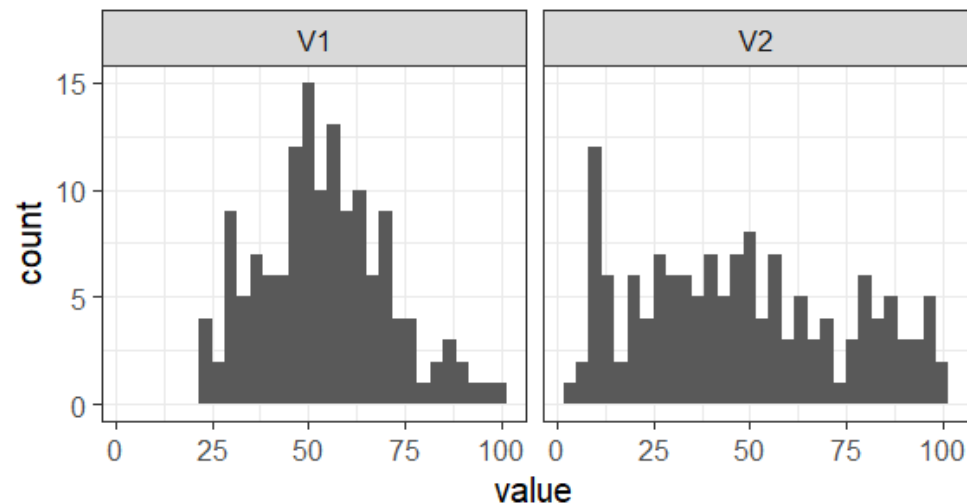
2 Department of Biostatistics, Medical Faculty, University of Belgrade, Belgrade, Serbia, **3** Division of Biomedical Statistic and Informatics, Mayo Clinic, Rochester, Minnesota, United States of America

Summary statistics are only meaningful
when there are enough data to summarize

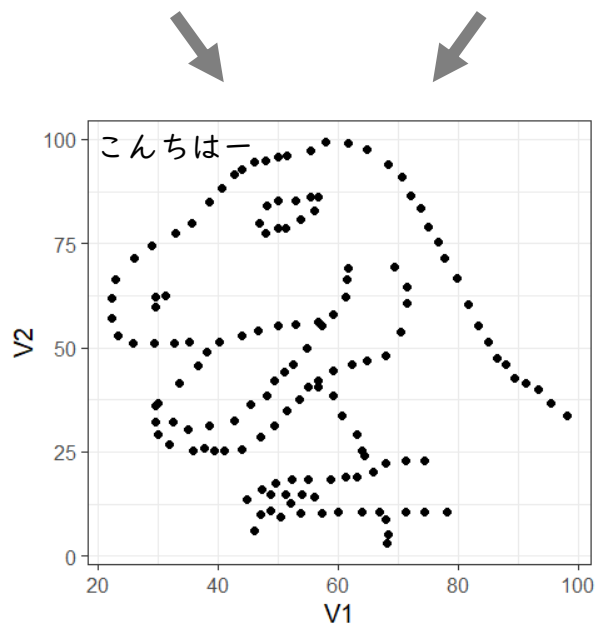
Scientists need better training in
how to select the appropriate type of figure for their data



各変数のヒストグラムに
おかしいところはなさそう...



各変数のヒストグラムに
おかしいところはなさそう...



散布図を描いて
はじめて発見できることもある

こちらに興味深いデモ

- Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing

<https://www.autodeskresearch.com/publications/samestats>

ggplot2による可視化

Overview

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.

Installation

ggplot2とは Grammar of Graphicsの原理で動くシステム

```
# The easiest way to get ggplot2 is to install the whole tidyverse:
```

```
install.packages("tidyverse")
```

```
# Alternatively, install just ggplot2:
```

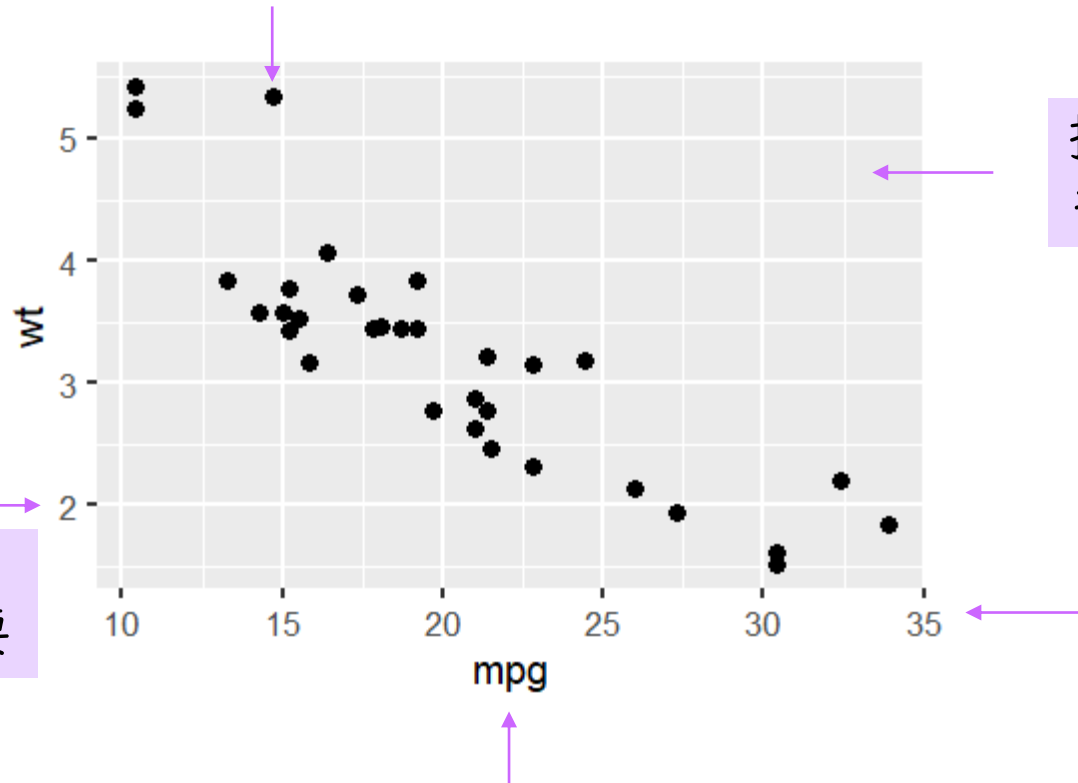
```
install.packages("ggplot2")
```

```
# Or the the development version from GitHub:
```

```
# install.packages("devtools")
```

```
devtools::install_github("tidyverse/ggplot2")
```

変数間の関係を「散布図」で表す、
という命令が必要



wtという
変数が必要

Y軸をどのスケールで
表示するかを決める必要

描画するための
キャンバスが必要

X軸をどのスケールで
表示するかを決める必要

mpgという
変数が必要

グラフを、
プラモデルのように
部品の組み合わせであると捉える

まずはdata.frame または tibbleを用意する



```
data(mtcars)
head(mtcars)
```

	燃費					重量			オートマ車か マニュアル車か ↓		
	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1



```
library(ggplot2)
```

```
g <- ggplot()
```

```
plot(g) # 単に`g`でも、`print(g)`でもいい
```

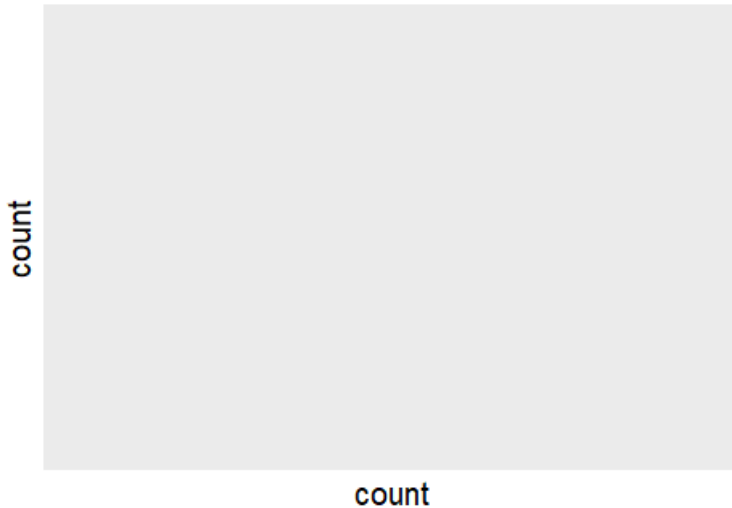
関数`ggplot()`で、
新品の画用紙（左図）を用意する

```
g <- ggplot() +  
  geom_histogram()  
  
plot(g)
```

+で、画用紙の上に追加したい情報（レイヤー）を好きなだけ重ねていく
（服を重ね着するイメージ）

画用紙の上に描画したいグラフを指定する
（geometric objectと呼ぶ）

主に`geom_xxx()`という名称
（左のコードでは、ヒストグラムを描こうとしている）




geometric objectの例

- 棒グラフ
 - `geom_bar()`
- 散布図
 - `geom_point()`
- 折れ線グラフ
 - `geom_line()`
- 箱ひげ図
 - `geom_boxplot()`

ものすごく多いので、
困ったら公式サイト
(<https://ggplot2.tidyverse.org/reference/>)
を見ましょう

1つの量的変数の可視化

ヒストグラムを例に



```
g <- ggplot() +  
  geom_histogram(data = mtcars)  
  
plot(g)
```

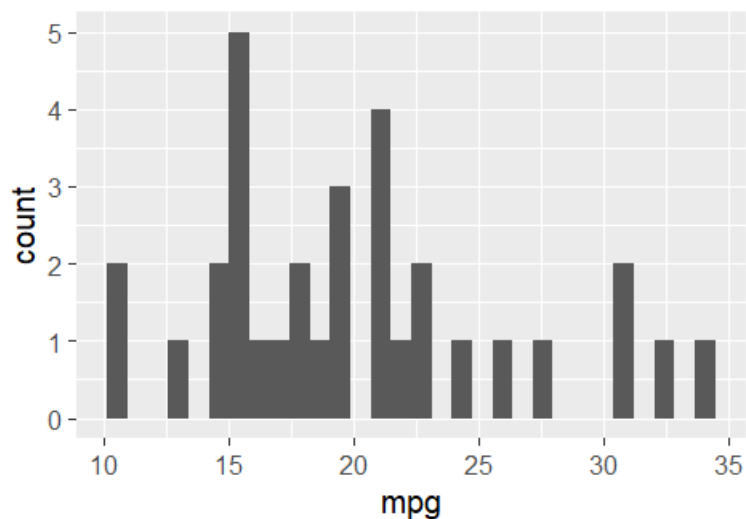
ヒストグラムで描画したい変数は、
どのデータセットに含まれているか
教えてやる

なお、この段階でプロットしようとする、
「エラー: stat_bin() requires an x or y aesthetic.」という警告が出る

→ mtcarsデータセットを使うことは分かったが、
「その中の、どの変数を描画するねん?!」と怒られている



```
g <- ggplot() +  
  geom_histogram(data = mtcars,  
                 mapping = aes(x = mpg))  
  
plot(g)
```

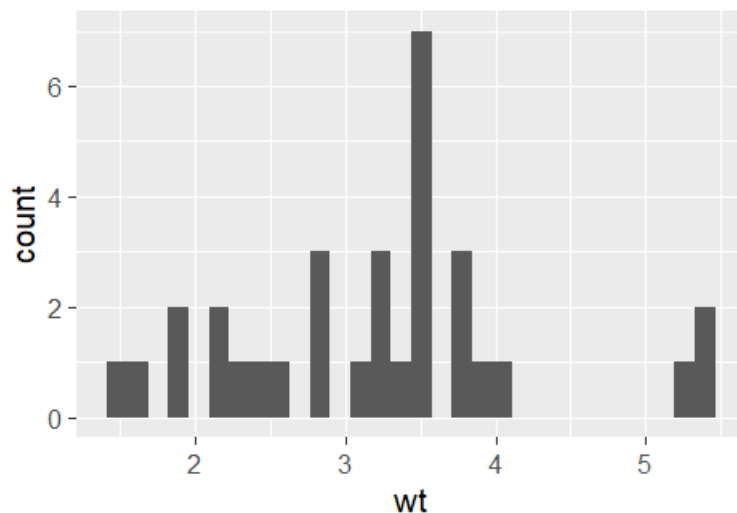


変数をグラフの部品として割り当てていく作業を、マッピング (mapping) と呼ぶ

マッピングする変数は、`aes()`の中で個別に指定する (今回は、mpgという変数をX軸にマッピングしている)



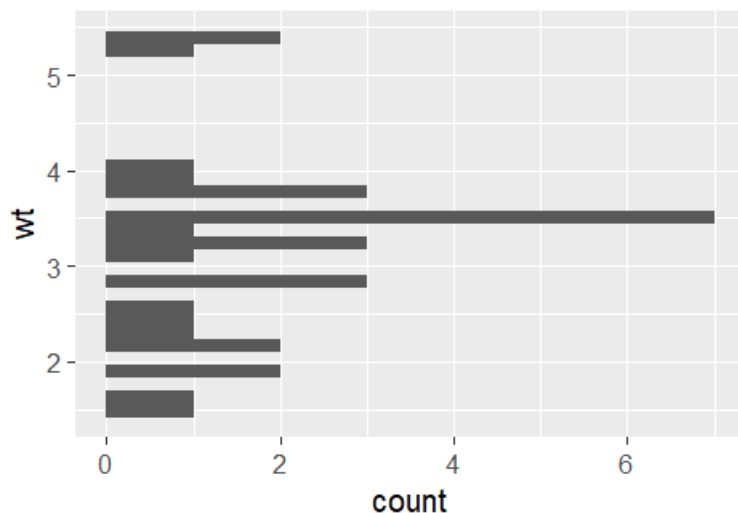
```
g <- ggplot() +  
  geom_histogram(data = mtcars,  
                 mapping = aes(x = wt))  
  
plot(g)
```



当然、X軸にマッピングする変数が変われば、
グラフの出力も変わる



```
g <- ggplot() +  
  geom_histogram(data = mtcars,  
                 mapping = aes(y = wt))  
  
plot(g)
```



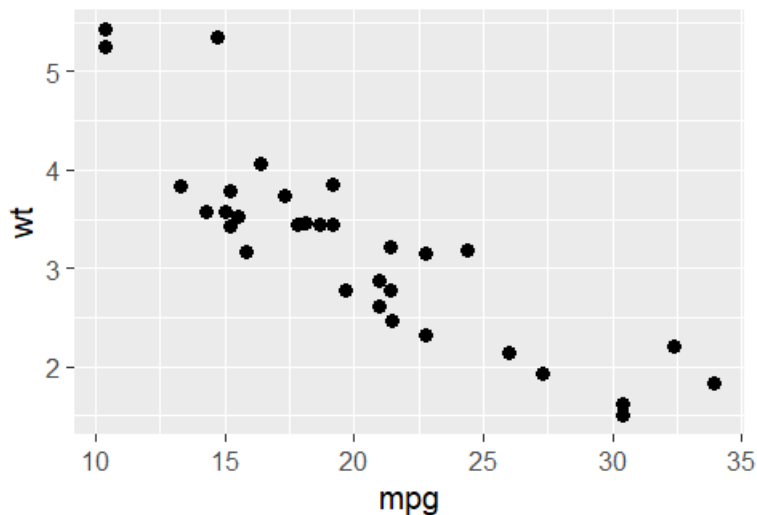
(一般的ではないが)
Y軸にマッピングすることで、こんなヒストグラムも描ける

この技術は意外と、棒グラフや箱ひげ図など、
離散変数が含まれるグラフで使うこともあるので、覚えておこう

2つの量的変数の可視化



```
g <- ggplot() +  
  geom_point(data = mtcars,  
             mapping = aes(x = mpg, y = wt))  
  
plot(g)
```

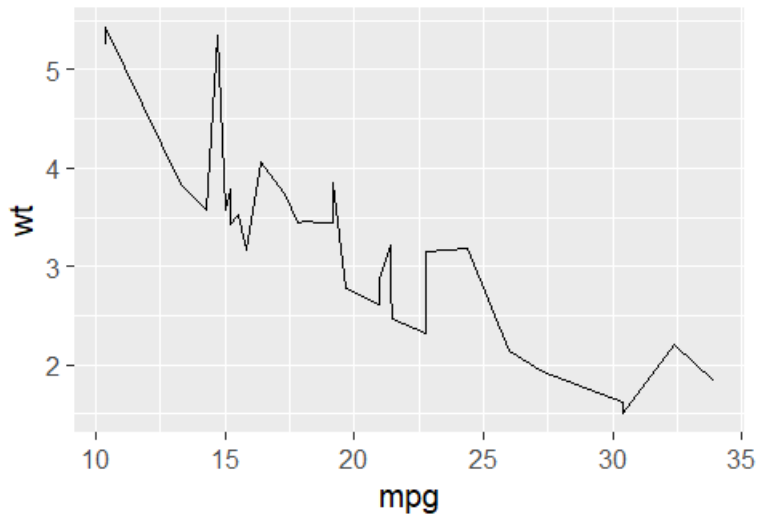


、`geom_point()`で散布図を描きたい

散布図には2つの量的変数が必要なので、
X軸にもY軸にも変数をマッピングする



```
g <- ggplot() +  
  geom_line(data = mtcars,  
            mapping = aes(x = mpg, y = wt))  
  
plot(g)
```



``geom_line()``で折れ線グラフを描きたい

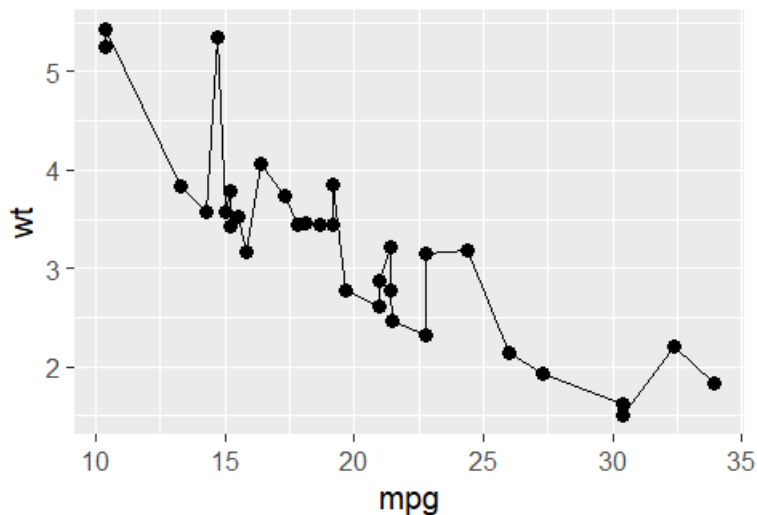
折れ線グラフには2つの量的変数が必要なので、
X軸にもY軸にも変数をマッピングする

tips 1

複数のgeometric objectの重ね描き



```
g <- ggplot() +  
  geom_point(data = mtcars, mapping = aes(x = mpg, y = wt)) +  
  geom_line(data = mtcars, mapping = aes(x = mpg, y = wt))  
  
plot(g)
```



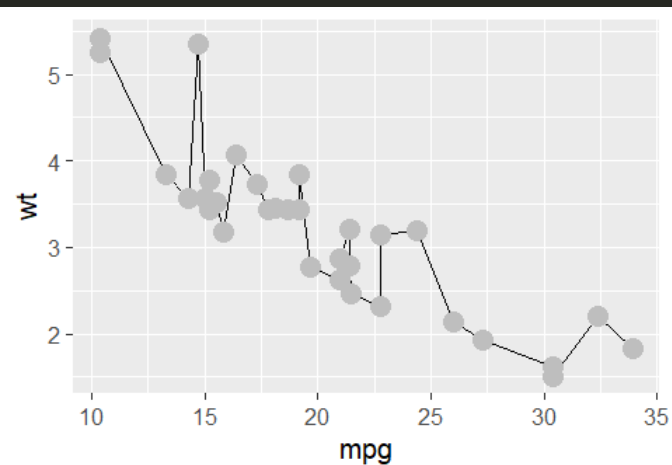
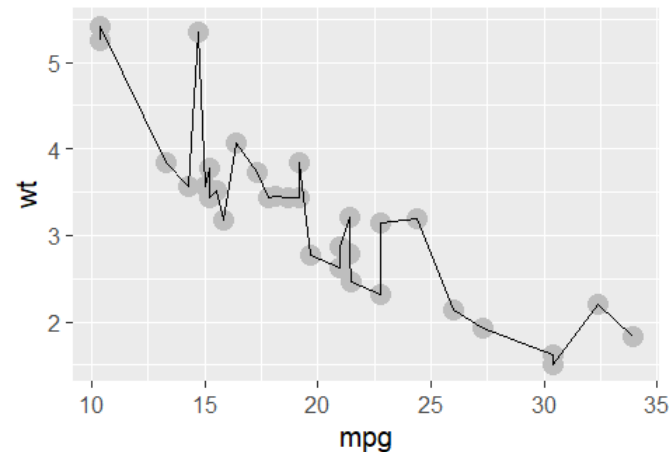
レイヤーはいくつ重ねてもいい
(重ね着のイメージであることを思い出そう)



```
ggplot() +  
  geom_point(data = mtcars, mapping = aes(x = mpg, y = wt)) +  
  geom_line(data = mtcars, mapping = aes(x = mpg, y = wt))  
  
ggplot() +  
  geom_line(data = mtcars, mapping = aes(x = mpg, y = wt)) +  
  geom_point(data = mtcars, mapping = aes(x = mpg, y = wt))
```

後から追加したレイヤーほど、
より上に重なることに注意

※点の色を変える方法は後述するので、心配無用



tips 2

複数の幾何学的オブジェクトに共通するマッピング



```
g <- ggplot() +  
  geom_point(data = mtcars, mapping = aes(x = mpg, y = wt)) +  
  geom_line(data = mtcars, mapping = aes(x = mpg, y = wt))  
  
plot(g)
```

同じことを何度も書きたくないんだけど…



```
g <- ggplot(data = mtcars, mapping = aes(x = mpg, y = wt)) +  
  geom_point() +  
  geom_line()  
  
plot(g)
```

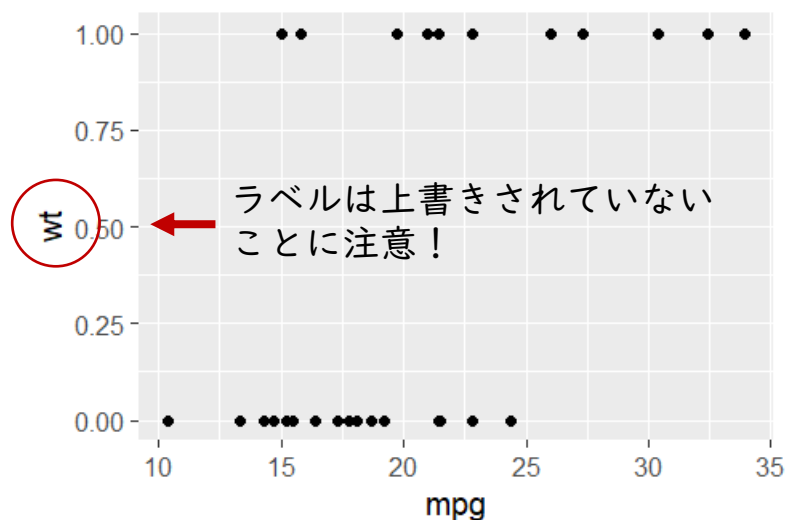
「ggplot()」の中で書いた情報は、以降のgeometric objectに自動的に継承される

(常連の店で、「いつもの！」っていうイメージ)



```
g <- ggplot(data = mtcars, mapping = aes(x = mpg, y = wt)) +  
  geom_point(mapping = aes(y = am))
```

```
plot(g)
```



マッピングを継承した後も、
個別のgeometric objectの中で、新しくマッピングを上書き可能

左図では、Y軸にマッピングする変数を上書きしている
(変数amには、0または1のみ記録されている)

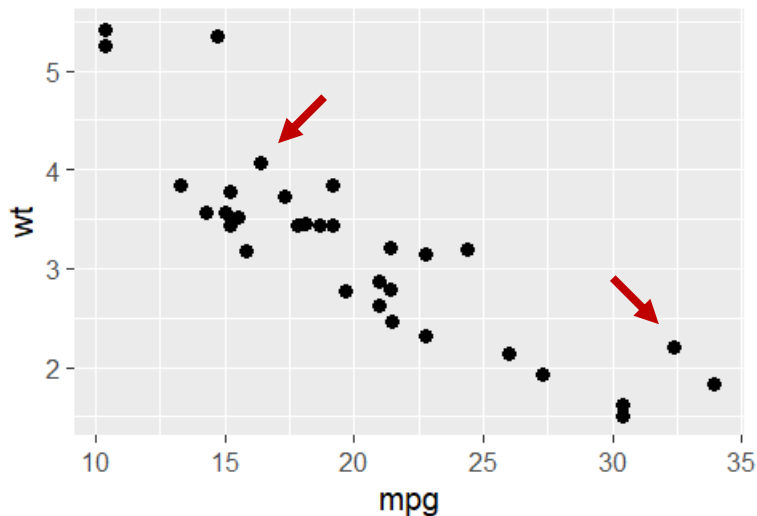
色や形状、などの “見え”の調整

aesthetic mapping



```
g <- ggplot(data = mtcars,  
            mapping = aes(x = mpg, y = wt)) +  
  geom_point()
```

```
plot(g)
```



それぞれの点は、

- オートマ車のデータ (am == 0)
- マニュアル車のデータ (am == 1)

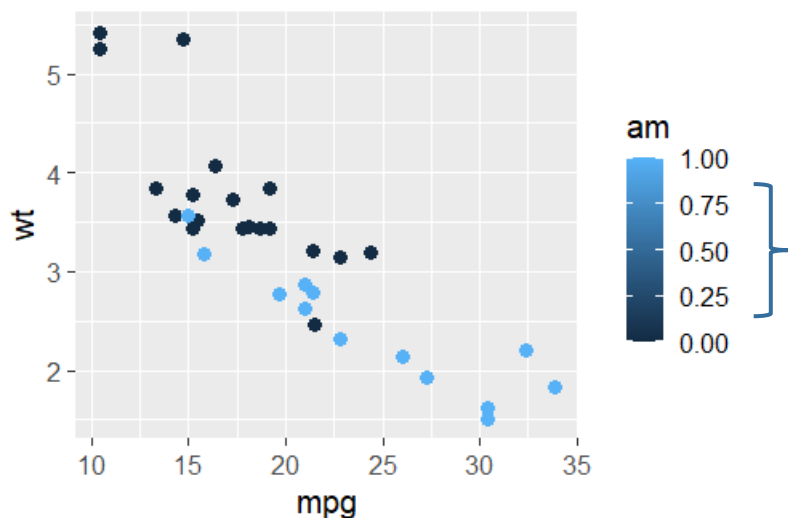
のいずれか。

どちらのデータなのか知りたい...



```
g <- ggplot(data = mtcars,  
            mapping = aes(x = mpg, y = wt, color = am)) +  
  geom_point()
```

```
plot(g)
```



変数amの値に応じて、点の色（color）を変えるように指定

でも、変数amは、0または1の値しか存在しないはずでは...？
→ 実数型（numeric）で定義されていることが原因

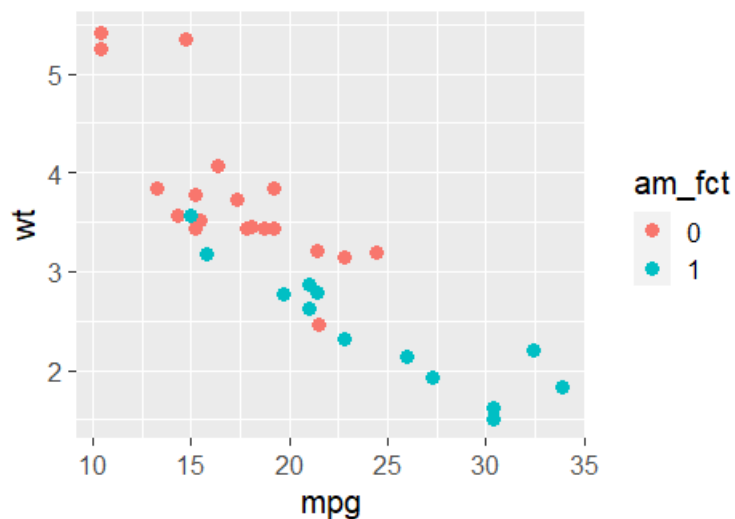
```
> is.numeric(mtcars$am)  
[1] TRUE
```



```
mtcars$am_fct <- as.factor(mtcars$am) ← 変数amを因子型 (factor) にした  
変数am_fctをデータフレームに追加
```

```
g <- ggplot(data = mtcars,  
            mapping = aes(x = mpg, y = wt, color = am_fct)) +  
  geom_point()
```

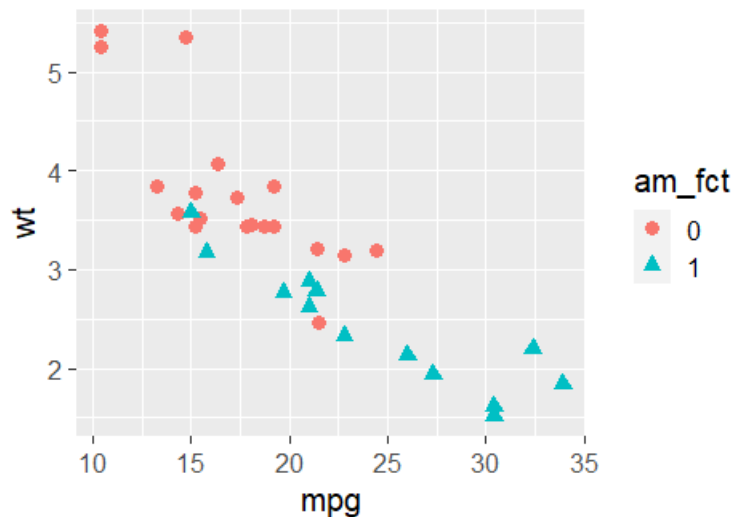
```
plot(g)
```



存在する水準のみで色が識別された！

マニュアル車は軽くて燃費が良いんですねえ...

```
g <- ggplot(data = mtcars,  
            mapping = aes(x = mpg,  
                          y = wt,  
                          color = am_fct,  
                          shape = am_fct)) +  
  
  geom_point()  
  
plot(g)
```

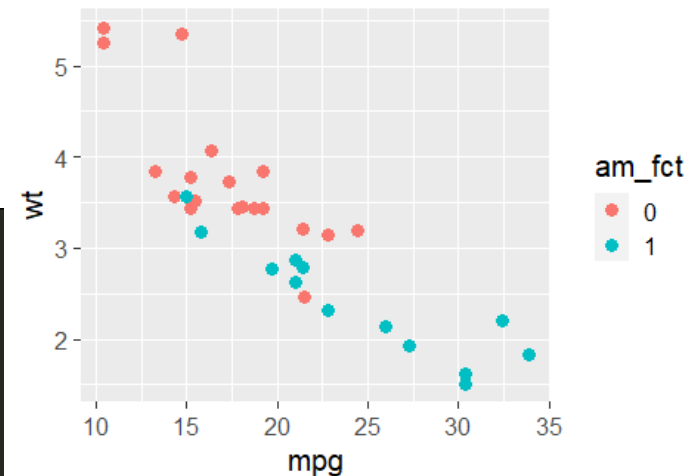


色 (color) だけでなく、
点の形状 (shape) や透過度 (alpha) など、
様々な“見栄え”を変更可能



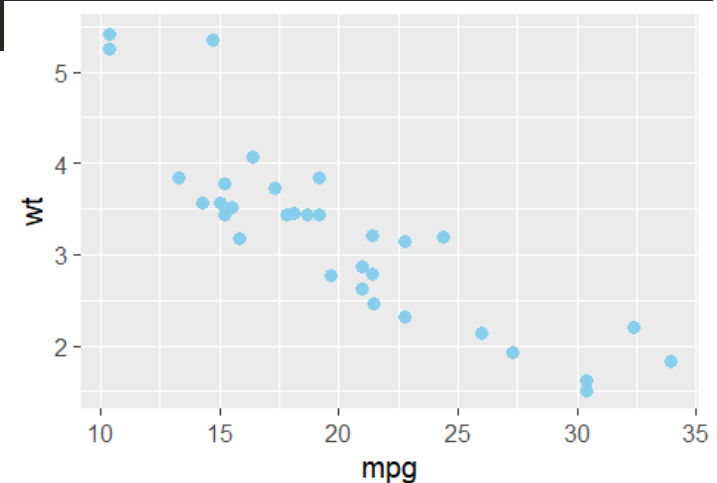
```
ggplot() +  
  geom_point(data = mtcars,  
             mapping = aes(x = mpg, y = wt, color = am_fct))
```

```
ggplot() +  
  geom_point(data = mtcars,  
             mapping = aes(x = mpg, y = wt), color = "skyblue")
```



↑ `aes()`の中にある

↓ `aes()`の外にある



グラフ中の全てのデータに対して、
”見栄え”を変えたければ、
`aes()`の外で指定する

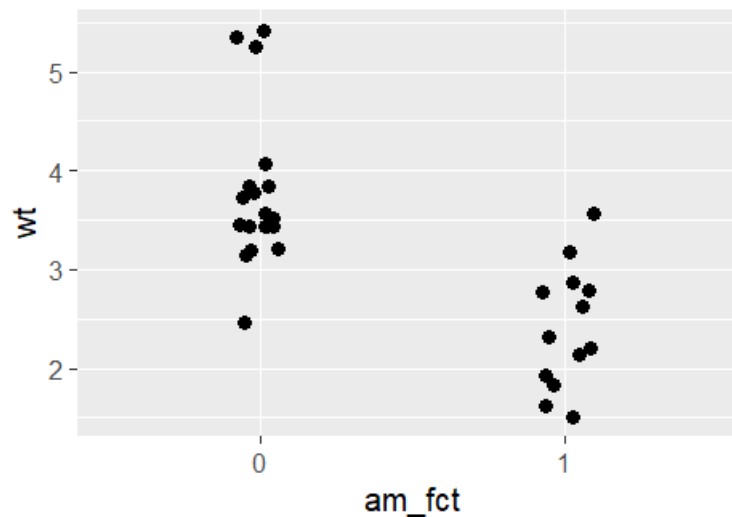
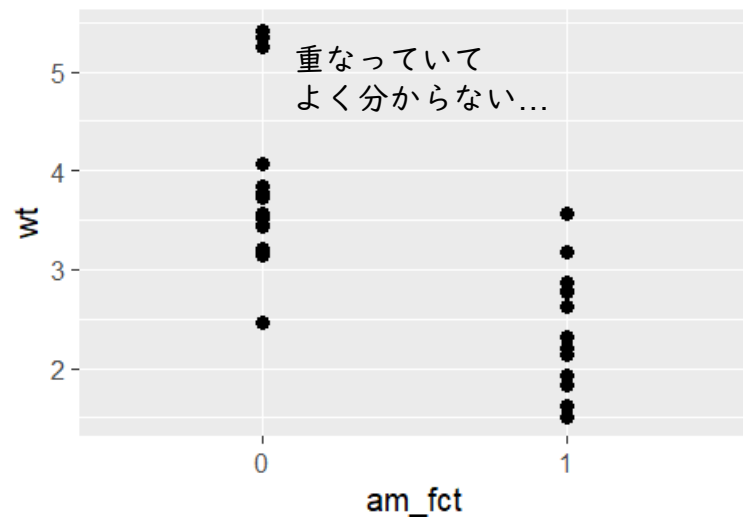
position

描画の「位置」に関する指定



```
g <- ggplot() +  
  geom_point(data = mtcars,  
            mapping = aes(x = am_fct, y = wt),  
            position = position_jitter(height = 0, width = 0.1))
```

```
plot(g)
```



デフォルトの設定では、
「正確な位置に描け」

今回は、「ランダムに散らせ
(ジッターをかけろ)」
という指定

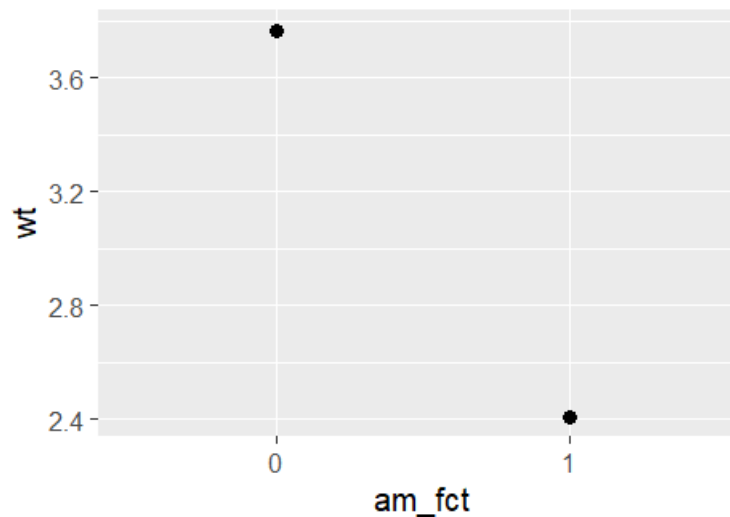
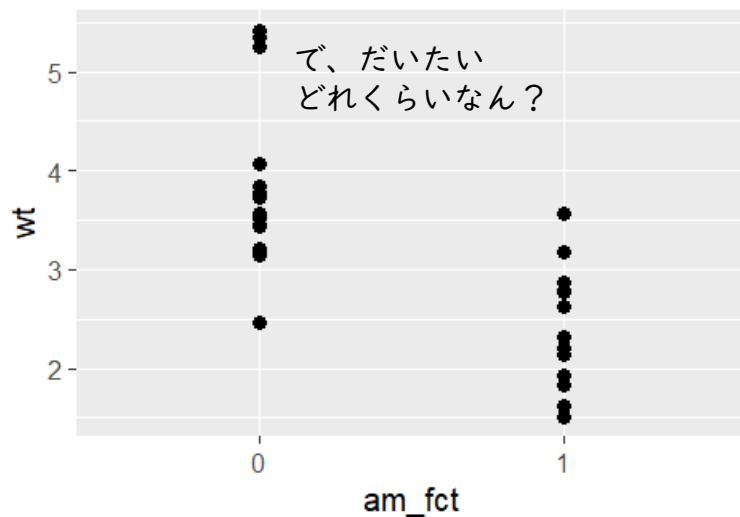
stat

どのように計算した値を描画するかの指定




```
g <- ggplot() +  
  geom_point(data = mtcars,  
             mapping = aes(x = am_fct, y = wt),  
             stat = "summary", fun = "mean")
```

```
plot(g)
```



デフォルトの設定では、
「正確な値を描け」

今回は、「`mean()`関数で
要約したうえで描け」
という指定



```
g <- ggplot() +  
  geom_histogram(data = mtcars)  
  
plot(g)
```

なお、この段階でプロットしようとする、
「エラー: **stat_bin()** requires an x or y aesthetic.」という警告が出る

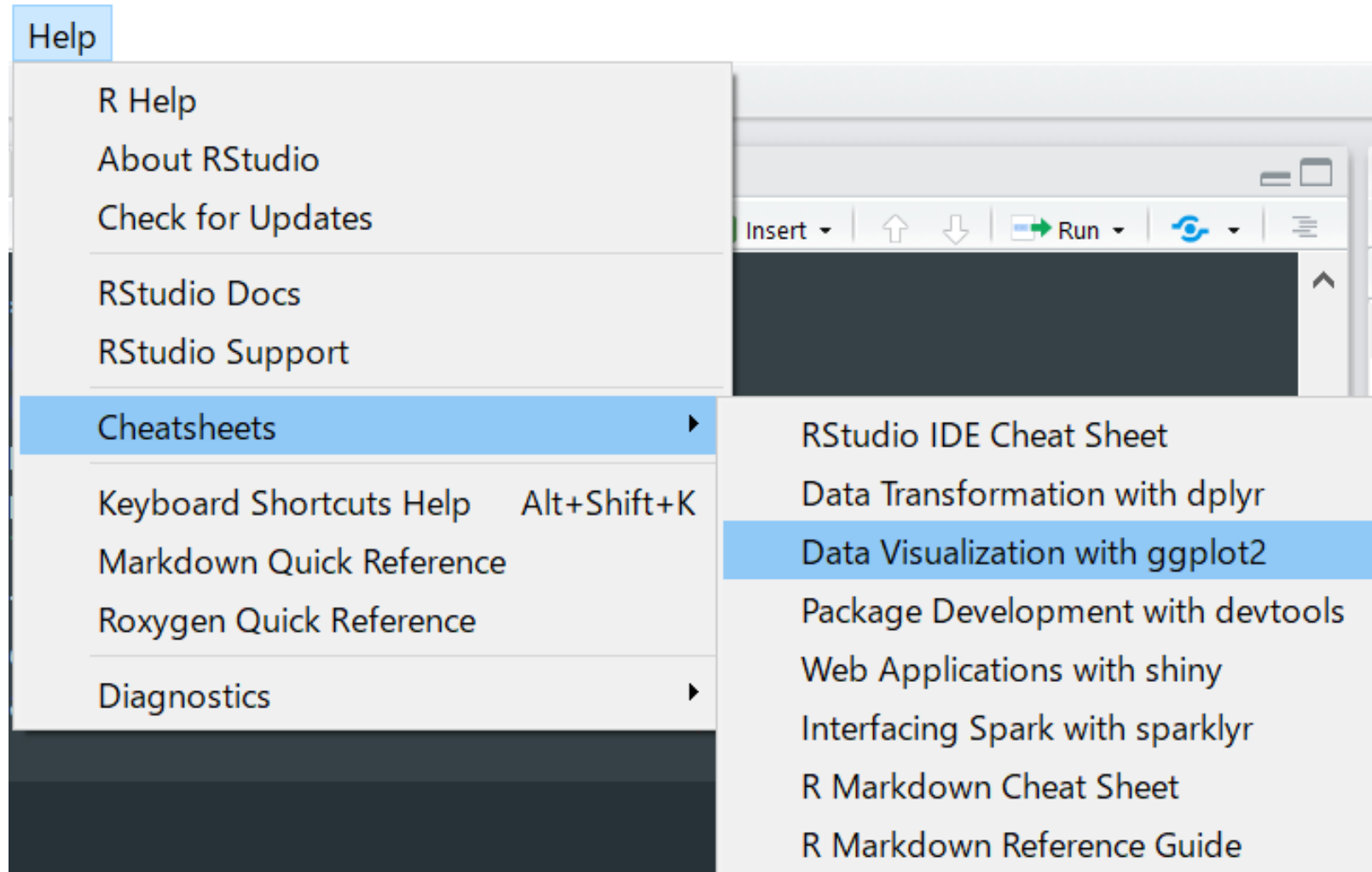
↑
`geom_histogram()`は、デフォルトで
stat = "bin"の設定になっているという意味

```
> geom_histogram
function (mapping = NULL, data = NULL, stat = "bin", position = "stack",
  ..., binwidth = NULL, bins = NULL, na.rm = FALSE, orientation = NA,
  show.legend = NA, inherit.aes = TRUE)
{
  layer(data = data, mapping = mapping, stat = stat, geom = GeomBar,
    position = position, show.legend = show.legend, inherit.aes = inherit.aes,
    params = list(binwidth = binwidth, bins = bins, na.rm = na.rm,
      orientation = orientation, pad = FALSE, ...))
}
<bytecode: 0x000002c99bbb7be0>
<environment: namespace:ggplot2>
```

ほらね！

おすすめの参考資料

公式チートシート





基本的な記法から、
高度な事例まで網羅している

一気に通読することを目指すよりも、
自分がレベルアップするたびに
読めるページが増えていく、という楽しみ方をおすすめ

Road to ggplot2再入門

第91回 Tokyo.R

発表者：das Kino (@kyn02666)

1



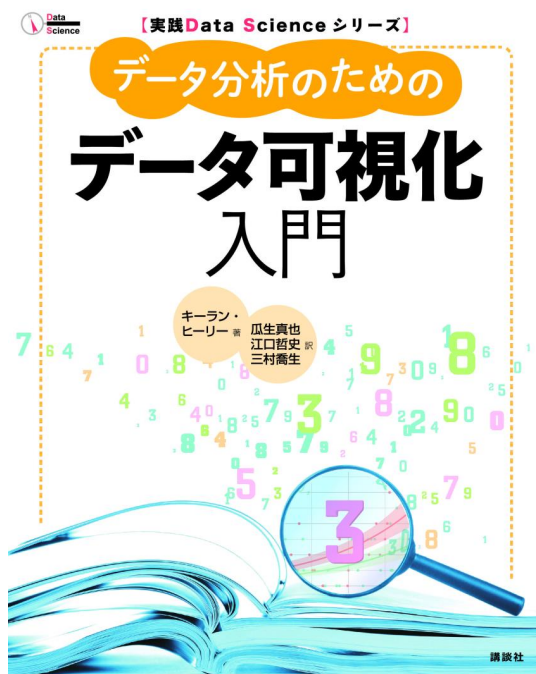
ggplot2 再入門

Hiroaki Yutani
(@yutannihilation)

1



※効果には
個人差があります





ggplot2 workshop part 1

75,503 回視聴 • 2020/03/24 にライブ配信

👍 2242 💬 14 ➦ 共有 📌 保存 ...



Thomas Lin Pedersen
チャンネル登録者数 3630人

チャンネル登録

Part 1 of 2 of my impromptu beginner/intermediate ggplot2 workshop. It will focus on teaching the underlying theory of ggplot2 and how it is reflected in the API

<https://www.youtube.com/watch?v=h29g21z0a68>

Enjoy !!