

初心者セッション： データハンドリング

BeginnerR Session: Data Handling 101

18th April 2026, Tokyo.R #120

Yuta Kanzawa @yutakanzawa

Senior Data Scientist at ASKUL Corporation



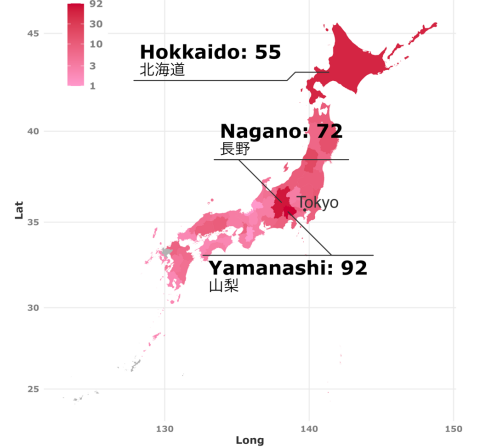
神沢雄大 Yuta Kanzawa

- データサイエンティスト@アスクル株式会社
- Twitter: [@yutakanzawa](https://twitter.com/yutakanzawa)
- 好きなもの：オペラとワイン
 - ワーグナー
 - ブルゴーニュ
- 使用可能言語：7.5
 - 人間：日本語、英語、ドイツ語（+フランス語）
 - コンピューター：R, Python, SAS, SQL



ポートフォリオ

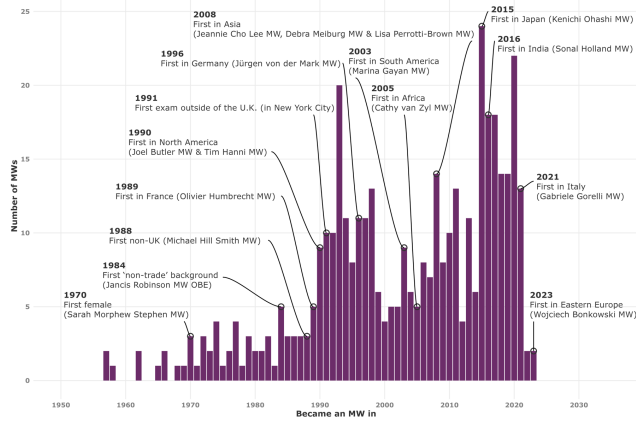
Number of Wineries in Japan in 2022, by Prefecture



Data: National Tax Agency Japan via https://www.nta.go.jp/taxes/sake/shiori-gaiyo/wine_enough/05.pdf#05
Graphic: Yuta Kanzawa

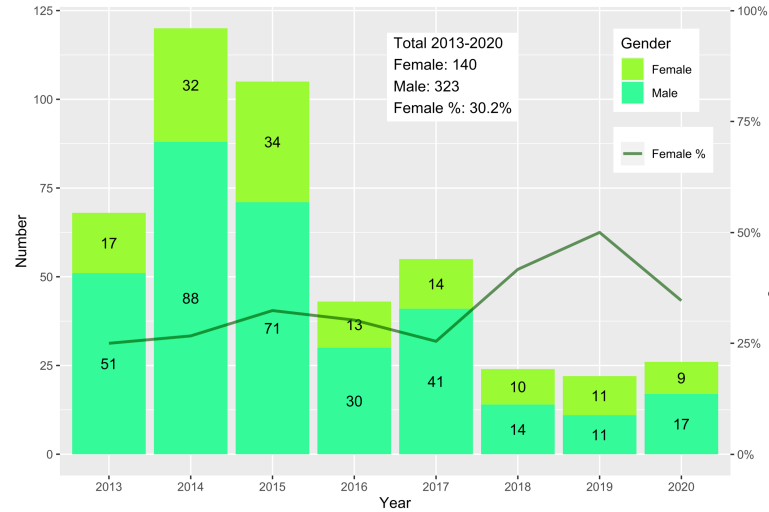
415 Active Masters of Wine by Year of Qualification

As of May 2023, 500 people have gained the title since the inaugural exam in May 1953. NB: 85 deceased or resigned MWs are not counted here.



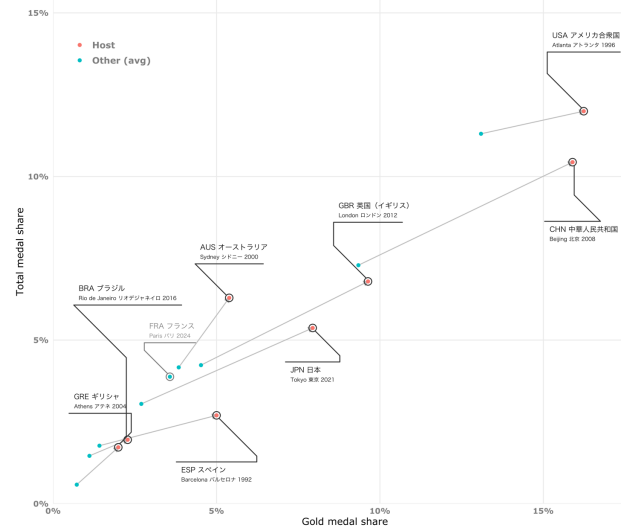
Data: The Institute of Masters of Wine via <https://www.mastersofwine.org/> - Graphic: Yuta Kanzawa

Number of Qualified JSA Sommelier Excellence and Equivalents* by Year and Gender, 2013-2020



Source: Japan Sommelier Association <https://www.sommelier.jp/exam/pdf/qualifiedholders.pdf>
*Sommelier Excellence (2019-2020), Senior Sommelier (2013-2018), Senior Wine Adviser (2013-2015)

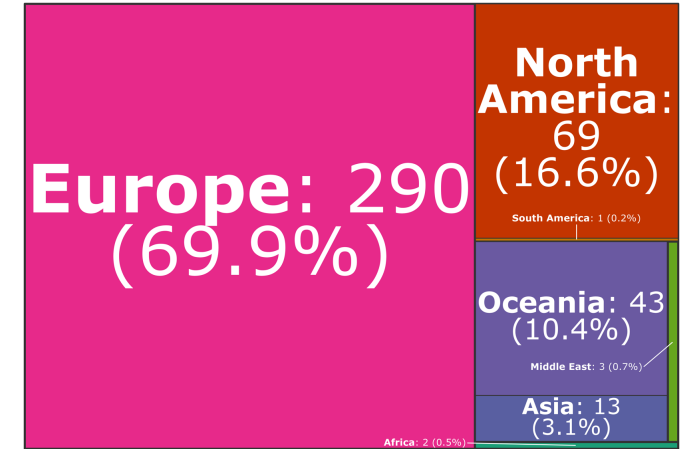
Medal shares of Olympic host countries in the past 30 years



Data: International Olympic Committee via <https://olympics.com> & <https://www.wikipedia.org> - Graphic: Yuta Kanzawa

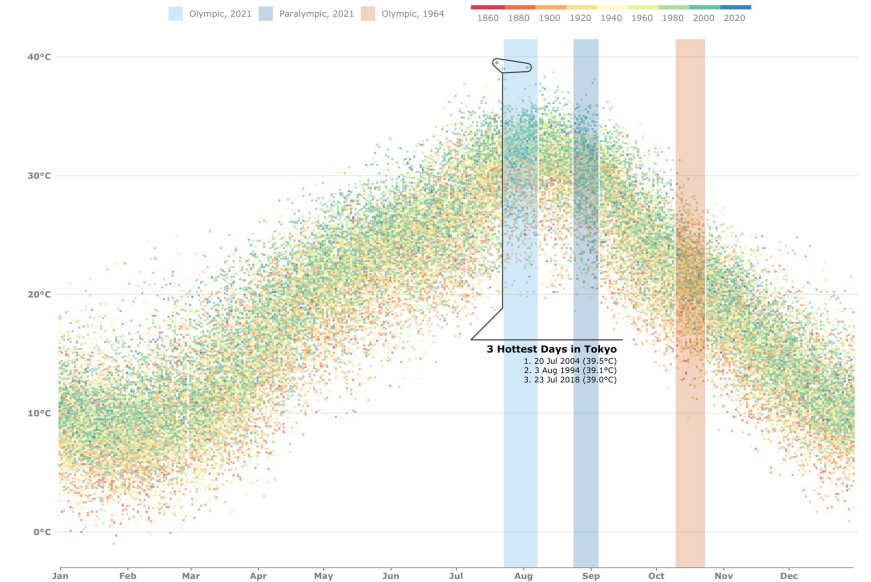
Number of Active MWs by Region Based in

70% of active MWs are based in Europe (mostly Western Europe). NB: Some MWs are multi-based.



Data: The Institute of Masters of Wine via <https://www.mastersofwine.org/> - Graphic: Yuta Kanzawa

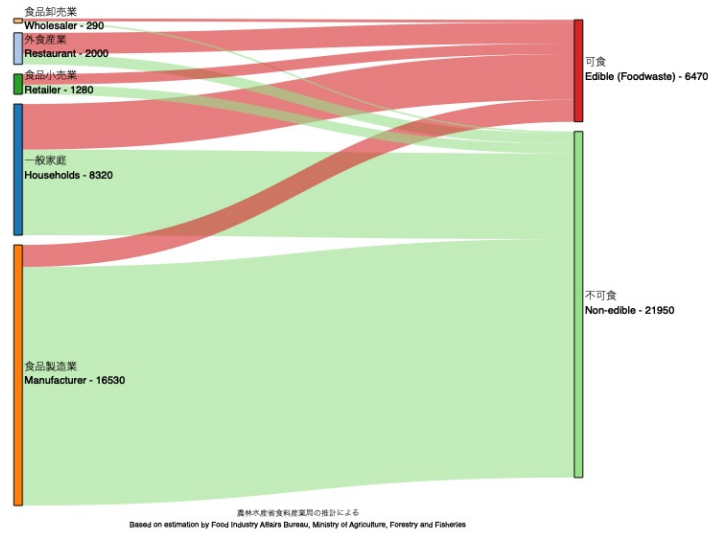
Daily maximum temperature in Tokyo, 1875-2021



Data: Japan Meteorological Agency via <https://www.jma.go.jp> - Graphic: Yuta Kanzawa (inspired by Cédric Scherer)

ポートフォリオ (参考までにR以外も)

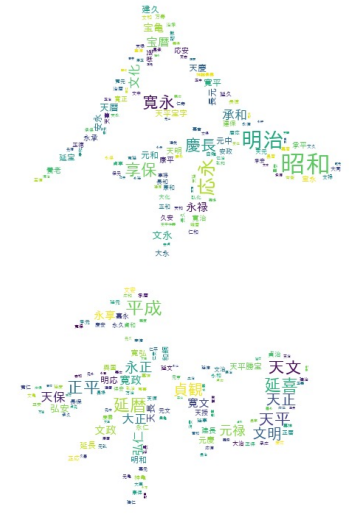
日本の食品廃棄物の発生量 (平成27年度推計) Estimated Food Disposals in Japan (FY2015)



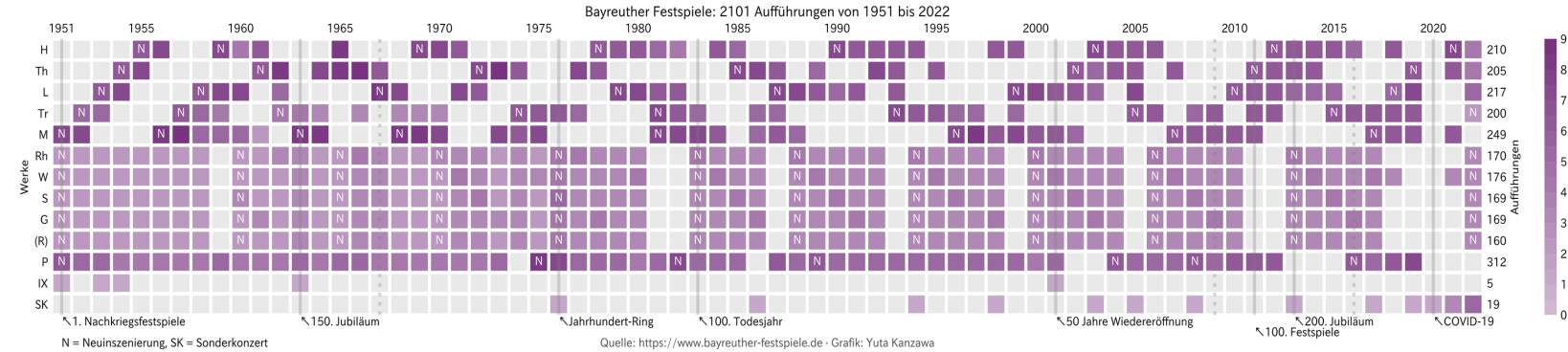
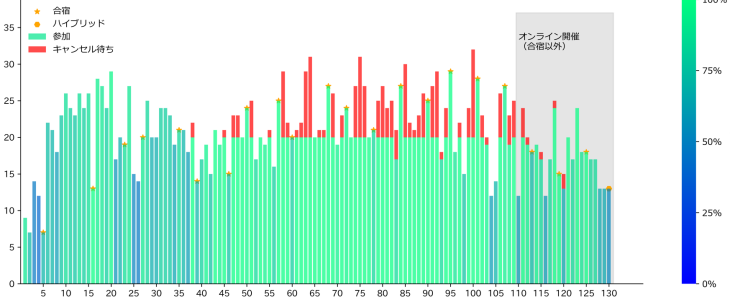
Clustering of Countries and Regions by Wine Trade Values & Production/Consumption Volumes in 2017 using t-SNE and K-Means



Sources: UN Comtrade (<https://comtrade.un.org/>), FAOSTAT (<http://www.fao.org/faostat/>)



Python mini Hack-a-thon 参加者数推移



アジェンダ

- 今日話すこと
 - tidyverse (特にdplyr)
 - ペンギンデータ
 - 対象 (以下のいずれか)
 - tidyverseを初めて触る人
 - tidyverseをなんとなく使っている人
 - 今日話さないこと
 - base RやExcelとの比較
- dplyrによる加工の理解

TL;DR

- データ型とデータ構造（特にベクトル、リスト、データフレーム）
- 値を入れるのは**代入演算子**「<-」で。
- ライブラリー（パッケージ）はコードの冒頭で呼び出しておく。
- コードを書くとき「スタイル」も意識する。→**フォーマッター**とリントー
- **パイプ演算子**を使って処理の流れを明確に！
- データの出力のおすすめは**parquet**形式！
- これからは**ペンギンデータ**！
- dplyrでのデータ加工は簡単！

R言語の基礎

R language 101

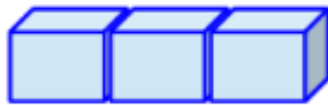
データ型

| データ型 | 名称 | 例 |
|-------------|------------------|---------------------|
| 文字列 | character | "あーる", "1" |
| 整数 | integer | -1L, 0L, 1L, 2L |
| 数値 (浮動小数点数) | numeric | 4.51 |
| 論理値 | logical | TRUE, FALSE |
| 日付 | Date | 2026-04-18 |
| 日時 | POSIXct, POSIXlt | 2026-04-18 14:00:00 |
| ファクター (因子) | factor | (カテゴリー値) |

※欠測値 (欠損値) は「NA」で表す。

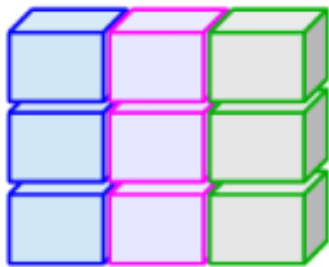
データ構造

Vector



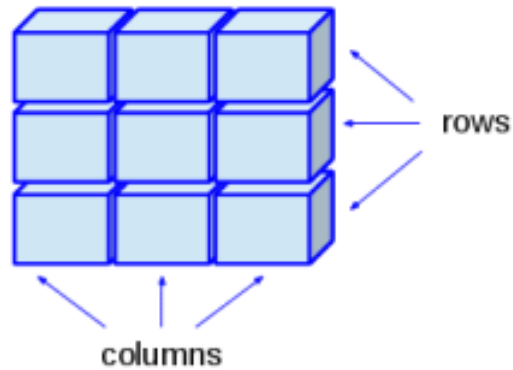
`c(1, 2, 3)`

Data Frame
(Table)

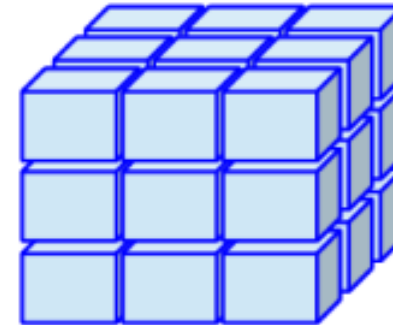


`data.frame()`

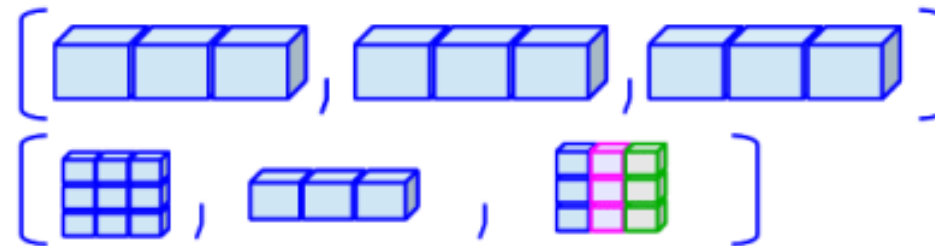
Matrix



Array



Lists



`list()`

* <https://balachandark.com/2022/08/07/r-series-4-data-structures/>

演算子

| 算術演算子 | 処理 |
|-------|----|
| + | 加法 |
| - | 減法 |
| * | 乗法 |
| / | 除法 |
| ^ | 累乗 |
| %/% | 商 |
| %% | 剰余 |

| 比較演算子 | 処理 |
|-------|-------|
| == | 等しい |
| != | 等しくない |
| > | 超 |
| < | 未満 |
| >= | 以上 |
| <= | 以下 |
| %in% | 含まれる |

| 論理演算子 | 処理 |
|-------|-----|
| & | AND |
| | OR |
| ! | NOT |

代入演算子

- 左のオブジェクトに右の値を入れる。 : 「<-」
 - 「=」と同じ。
 - ショートカットキー
 - Win: 「Alt+-」、Mac: 「Option+-」
- 例 :
 - `a <- 1`
 - `df <- read_csv(...)`

ライブラリー（パッケージ）

- 使う時に読み込む（呼び出す）必要がある。
 - `library()`がオススメ。1つずつ読み込む（コードの最初に1回だけ）。
 - 例：`library(tidyverse)` ←引用符なし！
- `::`記法
 - 課題 → `::`記法を使うと解決（ただし、コードが長くなりがちなのが欠点）。
 - どのライブラリーの関数なのか分かりにくい。
 - 関数名がライブラリー間で重複すると、最後に読み込んだライブラリーのものが有効。
 - ライブラリー名と関数名を「`::`」でつないで書く。
 - 例：
 - `readr::read_csv()`
 - `tidyr::pivot_longer()`

コードスタイルガイド

- コードを書く際の様式のルール（名前、空白、改行、文字数、など）
 - 書式が整ったコードは読みやすい。→ デバッグや保守をしやすい。
 - 意識しながら書くのは大変 → フォーマッターやリンターの使用
- tidyverse style guide
 - <https://style.tidyverse.org>
 - Google's R Style Guideのオリジナル版*1がベース → styler、lintr
- Google's R Style Guide
 - GoogleのRコミュニティがまとめ、tidyverse style guideの元になった*2。
 - 現行版： <https://google.github.io/styleguide/Rguide.html>

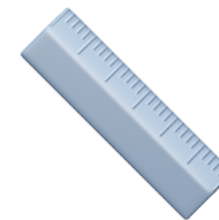


*1 オリジナル版（スタンフォード大学のコピー）： <https://web.Stanford.edu/class/cs109I/unrestricted/resources/google-style.html>
→日本語訳（RjpWiki）： <http://www.okada.jp/RWiki/?Google%27s+R+Style+Guide>

*2 ただし、現行版はtidyverse style guideから派生したもの。

フォーマッターやリンターの使用

- フォーマッター（見た目を整えるツール）
 - **styler**^{*1}: tidyverse style guide準拠
 - RStudioのアドイン同梱
 - formatR^{*2}
 - RStudioのアドイン : addinexamples^{*3}
 - どちらも**RStudio上でアドイン**として実行可能。
 - ショートカットキーも設定可能。
 - R Markdownでも利用可能。
- リンター（静的解析ツール）
 - **lintr**^{*4}: 静的チェック（**バグにつながりやすいコード**をチェック）



*1 <https://github.com/r-lib/styler>

*2 <https://yihui.org/formatr/>

*3 <https://rstudio.github.io/rstudioaddins/>

*4 <https://github.com/r-lib/lintr>

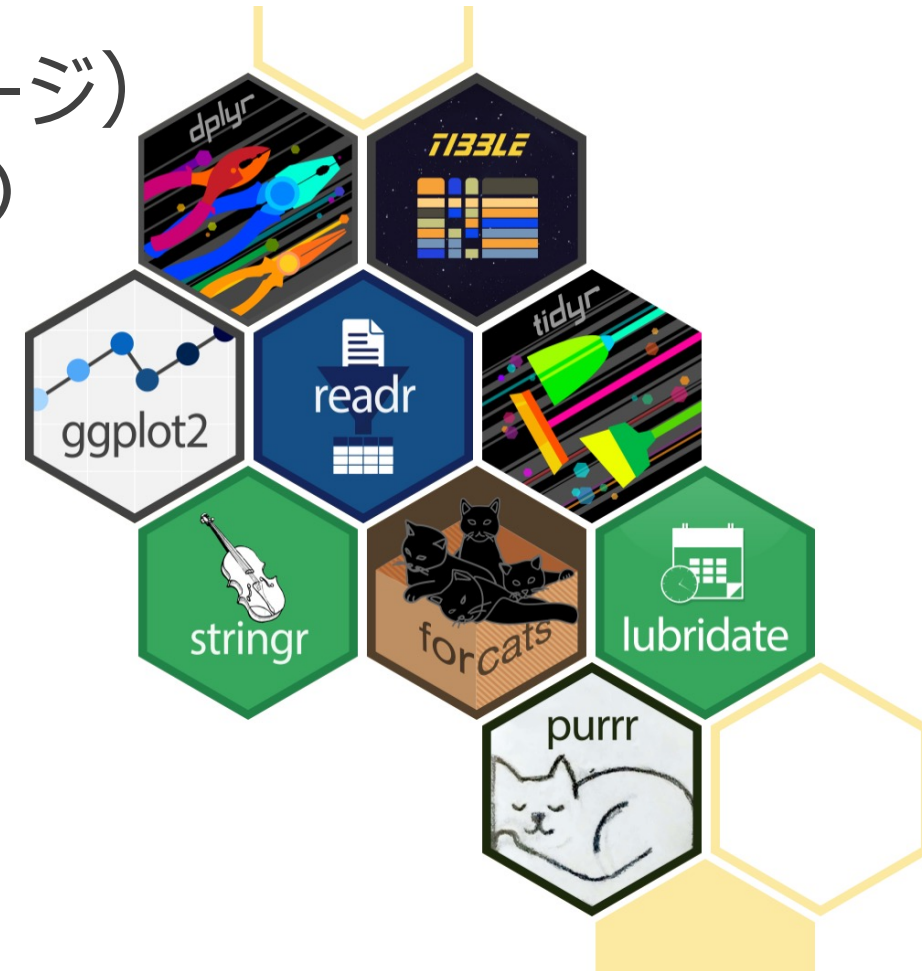
tidyverse

tidyverse

tidyverseパッケージ

- データ分析用のパッケージ群 (メタパッケージ)
 - `install.packages("tidyverse")`
 - `library(tidyverse)`

| パッケージ名 | 機能 | パッケージ名 | 機能 |
|---------|---------|-----------|------------|
| readr | データ読み込み | tibble | 強化版データフレーム |
| dplyr | データ加工 | stringr | 文字列処理 |
| tidyr | データ加工 | forcats | ファクター型 |
| ggplot2 | データ可視化 | lubridate | 日時処理 |
| | | purrr | 関数型プログラミング |



* <https://www.tidyverse.org>

パイプ演算子

Pipe operator

パイプ演算子

- 値を左から右へ渡す。
 - 左側の処理の出力を右側の処理の第1引数に入れる。
 - $f(x) \mid> g(y)$ は $g(f(x), y)$ と同じ。
- dplyr : 「%>%」、base R (4.1以降) 「|>」
 - ショートカットキー
 - Win: 「Ctrl+Shift+M」、Mac: 「Cmd+Shift+M」
 - 切り替え : *Tools* → *Global Options* → *Code* → *Editing* → *Use native pipe operator, |>* (requires R 4.1 +)
- パイプ演算子を使うことで、処理の流れが明確になる！
 - 関数の入れ子は複雑で分かりにくい。

tidyなデータ (整然データ)

Tidy data

tidyなデータ

- 条件
 - 各**変数**が1つの列で構成されている
 - 各**観測**が1つの行で構成されている
 - 各**観測単位**が1つの表で構成されている（各**値**が1つのセルに入っている）
- 必ずしも縦長なデータとは限らない！
- 実務と折り合いをつける！
 - 結局のところ**使いやすさ**。

| country | year | cases | population |
|-------------|------|--------|------------|
| Afghanistan | 1999 | 765 | 19997071 |
| Afghanistan | 2000 | 666 | 200095360 |
| Brazil | 1999 | 31737 | 172006362 |
| Brazil | 2000 | 80488 | 174004898 |
| China | 1999 | 210258 | 1270015272 |
| China | 2000 | 210766 | 1280028583 |

variables

| country | year | cases | population |
|-------------|------|--------|------------|
| Afghanistan | 1999 | 765 | 19997071 |
| Afghanistan | 2000 | 666 | 200095360 |
| Brazil | 1999 | 31737 | 172006362 |
| Brazil | 2000 | 80488 | 174004898 |
| China | 1999 | 210258 | 1270015272 |
| China | 2000 | 210766 | 1280028583 |

observations

| country | year | cases | population |
|-------------|------|--------|------------|
| Afghanistan | 99 | 765 | 19997071 |
| Afghanistan | 00 | 666 | 200095360 |
| Brazil | 99 | 31737 | 172006362 |
| Brazil | 00 | 80488 | 174004898 |
| China | 99 | 210258 | 1270015272 |
| China | 00 | 210766 | 1280028583 |

values

* <https://r4ds.had.co.nz/tidy-data.html#fig:tidy-structure>

tidyr::pivot_longer()とpivot_wider()

- 「縦持ち」、「横持ち」の変換

table4a

| country | 1999 | 2000 |
|---------|------|------|
| A | 0.7K | 2K |
| B | 37K | 80K |
| C | 212K | 213K |

→

| country | year | cases |
|---------|------|-------|
| A | 1999 | 0.7K |
| B | 1999 | 37K |
| C | 1999 | 212K |
| A | 2000 | 2K |
| B | 2000 | 80K |
| C | 2000 | 213K |

```
table4a |> pivot_longer(  
  cols = 2:3,  
  names_to = "year",  
  values_to = "cases"  
)
```

これが「観測」

table2

| country | year | type | count |
|---------|------|-------|-------|
| A | 1999 | cases | 0.7K |
| A | 1999 | pop | 19M |
| A | 2000 | cases | 2K |
| A | 2000 | pop | 20M |
| B | 1999 | cases | 37K |
| B | 1999 | pop | 172M |
| B | 2000 | cases | 80K |
| B | 2000 | pop | 174M |
| C | 1999 | cases | 212K |
| C | 1999 | pop | 1T |
| C | 2000 | cases | 213K |
| C | 2000 | pop | 1T |

→

| country | year | cases | pop |
|---------|------|-------|------|
| A | 1999 | 0.7K | 19M |
| A | 2000 | 2K | 20M |
| B | 1999 | 37K | 172M |
| B | 2000 | 80K | 174M |
| C | 1999 | 212K | 1T |
| C | 2000 | 213K | 1T |

```
table2 |> pivot_wider(  
  names_from = "type",  
  values_from = "count"  
)
```

* <https://rstudio.github.io/cheatsheets/tidyr.pdf>

データ読み書き

Loading & saving data

データ読み込み

- **CSVファイル** : `readr::read_csv()`
 - 大規模なデータは`fread`パッケージなども。
- **Excelファイル** : `readxl::read_excel()`
 - `xlsx`も`xls`も読み込める！ただし、~~`xls`は滅ぶべき。~~
- SAS、SPSS、Stataの出力ファイル : `haven`パッケージ
- JSON : `jsonlite::fromJSON()`
- parquet形式 : `arrow::read_parquet()`
- Web関連 : `httr`パッケージ、`rvest`パッケージ

データ出力

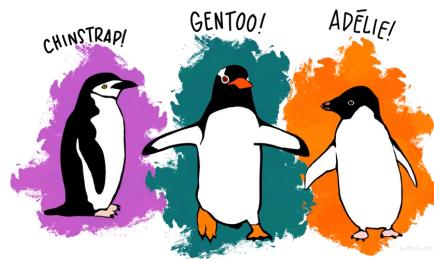
- テーブルデータ（表形式）は**parquet形式**がおすすめ！
 - データ型が保存される！
 - R以外の言語でも使える！（短所：言語独自のデータ型は使えない。）
 - 圧縮率が高い！
 - `arrow::write_parquet()`
- CSVファイル：`write.csv()`
- Excelファイル：`writexl`パッケージ、`xlsx`パッケージ
- R形式のバイナリー化：`save()`

データ加工の基礎

Basic data wrangling

ペンギンデータ

- 2007-9年の南極パーマー基地周辺におけるペンギンの調査データ
 - <https://doi.org/10.1371/journal.pone.0090081>
- R 4.5から標準搭載！
 - データセットは`data("penguins")`で呼び出す。
 - 4.5未満は`palmerpenguins`パッケージをインストール。
 - 注意：カラム名が違う。



```
> head(penguins)
  species   island bill_len bill_dep flipper_len body_mass   sex year
1 Adelie Torgersen   39.1    18.7        181     3750 male 2007
2 Adelie Torgersen   39.5    17.4        186     3800 female 2007
3 Adelie Torgersen   40.3    18.0        195     3250 female 2007
4 Adelie Torgersen    NA      NA          NA         NA <NA> 2007
5 Adelie Torgersen   36.7    19.3        193     3450 female 2007
6 Adelie Torgersen   39.3    20.6        190     3650 male 2007
```

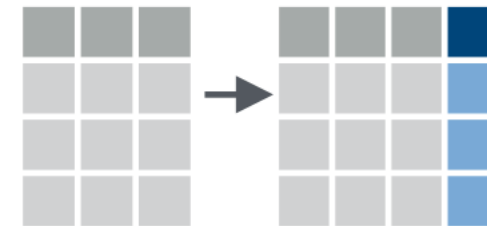
* <https://allisonhorst.github.io/palmerpenguins/>

列の抽出



- `dplyr::select(列名1, 列名2, ...)`
 - m行n列のデータフレームとして取り出す。
 - 1列をベクトルとして取り出したい場合は`pull()`を使う。
- 例：
 - ペンギンデータから`bill_len`と`bill_dep`の2列を抽出。
 - `penguins |> select(bill_len, bill_dep)`

列の追加



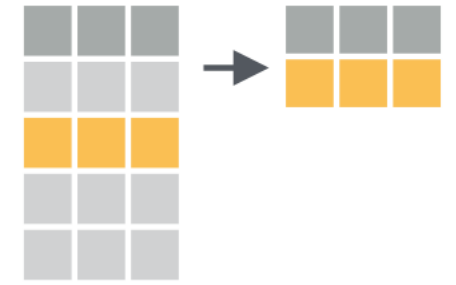
- `dplyr::mutate(追加する列名 = 値や計算式, ...)`
 - データセットに新しい列を追加 (した新しいデータセットを作成) する。

例 :

- ペンギンデータにひれとくちばしの長さの比率を表す列を追加。

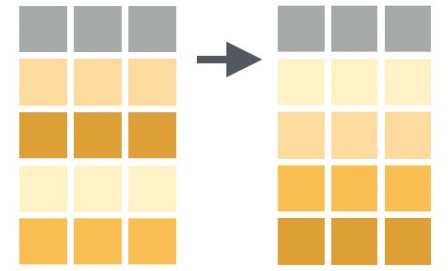
```
penguins |> mutate(  
  flipper_bill_ratio = flipper_len / bill_len  
)
```

行の抽出



- `dplyr::filter(条件)`
 - 条件に該当する行をデータフレームとして取り出す。
- 例：
 - ペンギンデータから2007年のデータを取り出す。
 - `penguins |> filter(year == 2007)`
 - ペンギンデータから2007年のアデリーペンギンのデータを取り出す。
 - `penguins |> filter(year == 2007 & species == "Adelie")`

行のソート



- `dpLyr::arrange(ソートの基準にする列名)`
 - 指定された列を基準にデータセット全体をソートする（デフォルトは昇順）。
- 例：
 - ペンギンデータを**体重の昇順**にソート。
 - `penguins |> arrange(body_mass)`
 - ペンギンデータを**体重の昇順**にソートし、**年の降順**にソート。
 - `penguins |> arrange(body_mass, desc(year))`
 - 指定した順にソートされる。

グループごとの集約

- `group_by(グループを表す列) |> summarise(計算式)`
 - `group_by()`で集約キーとなるグループを指定し、`summarise()`に記述した計算をグループごとに行う。
 - 米国式に`summarize()`と綴ってもよい。
 - どちらも`dplyr`の関数。
- 例：
 - ペンギンデータから、それぞれの種のそれぞれの年の平均体重を算出する。
 - `penguins |> group_by(species, year) |> summarise(mean(body_mass, na.rm = TRUE))`
 - 注：`mean(...)`は`summarise()`の処理内容を指定する引数の値なので入れ子にする。



この他にも...

- **重複**を削除→`dplyr::distinct()`
- データセットを**列方向**に接着→`dplyr::bind_cols()`
- データセットを**行方向**に接着→`dplyr::bind_rows()`
 - SQLのUNION相当
- **列をキー**にしたデータセットの結合→`dplyr::left_join()`など
 - SQLの各種JOIN相当
- データセットを**縦持ち**に変換→`tidyr::pivot_longer()`
- データセットを**横持ち**に変換→`tidyr::pivot_wider()`

まとめ

Long story short

Long story short (1/2)

- **データ型**を意識する。
- データ構造
 - よく使うのはベクトル、リスト、データフレーム。
 - ベクトルはc()で作る。
- 値を入れるのは**代入演算子**「<-」で。
- ライブラリー（パッケージ）はコードの冒頭で呼び出しておく。
 - ::記法もおすすめ。
- コードを書くとき「スタイル」も意識する。→**フォーマッター**とリントー

Long story short (2/2)

- **tidyverse**は便利！
 - パイプ演算子：`dplyr`「%>%」と標準「|>」
 - tidyなデータへの変換：`pivot_longer()`と`pivot_wider()`
- データの読み書きはファイル形式に応じて！おすすめは**parquet**！
- これからは**ペンギンデータ**！
- `dplyr`でのデータ加工
 - `select()`, `mutate()`, `filter()`, `arrange()`,
`group_by()`, `summarise()`

ネットで公開されている便利なリソース (1/2)

- 『はじめよう！R』（小杉、2025年）
 - 第117回Tokyo.R初心者セッション
 - <https://kosugitti.github.io/slides/TokyoR117/RforBeginner.html>

はじめよう！R

AUTHOR
Koji Kosugi, PhD 

AFFILIATION
Senshu University

自己紹介

- 小杉考司 (こすぎこうじ)
- 生年月日: 1976.1.17(117はいい数字)
- 専修大学人間科学部 教授 博士 (社会学)
- 担当講義: 心理学データ解析基礎, 心理学データ解析応用
- 専門分野
 - 心理尺度の作り方, 使い方
 - 多変量解析 (因子分析, 多次元尺度構成法), 統計モデリング
 - 統計パッケージ開発: テスト理論用パッケージ [exametrika](#)



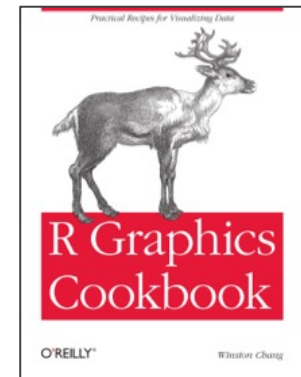
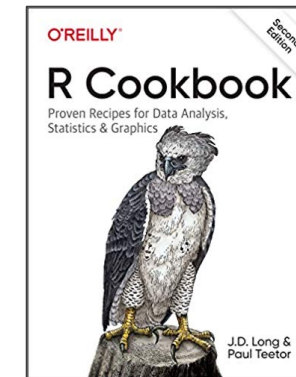
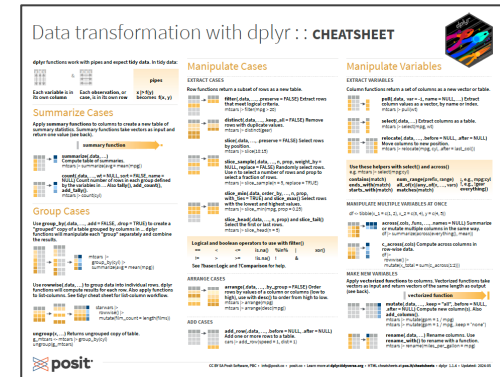
Rの紹介

目次

- 自己紹介
- Rの紹介
- Rのはじめかた
- RStudioも使いましょう
- RStudioの起動画面
- オススメ設定
- オススメ設定(つづき)
- RStudioの4つの窓
- RStudioはプロジェクト管理が基本
- Rをさわってみましょう
- はじめの1歩
- パッケージ
- パッケージの使い方
- 数値計算の基礎
- ベクトル, 行列, リスト, データフレーム
- ベクトル (Vector) の例
- 行列 (Matrix)
- 配列 (Array)
- リスト (List)
- リスト (List)
- データフレーム (Data Frame)

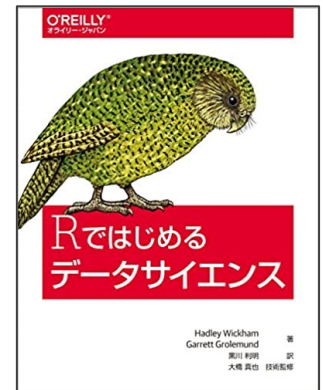
ネットで公開されている便利なリソース (2/2)

- Posit社のチートシート集
 - <https://posit.co/resources/cheatsheets/>
- R Cookbook
 - R Cookbookの著者によるwebサイト
 - <https://rc2e.com>
- Cookbook for R
 - R Graphics Cookbookの著者によるwebサイト
 - <http://www.cookbook-r.com>



参考書

- 『RユーザのためのRStudio入門』改訂2版（松村、湯谷、紀ノ定、前田、2021年）
 - いわゆる「宇宙船本」
- 『Rクックブック』第2版（Long, Teetor、2020年）
- 『Rではじめるデータサイエンス』（Wickham, Grolemond、2017年）



Enjoy!