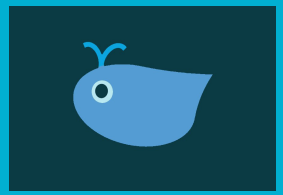
2022/04/16 Tokyo.R 初心者セッション

テーブルデータの取り扱い

―

がんばらないデータ加工

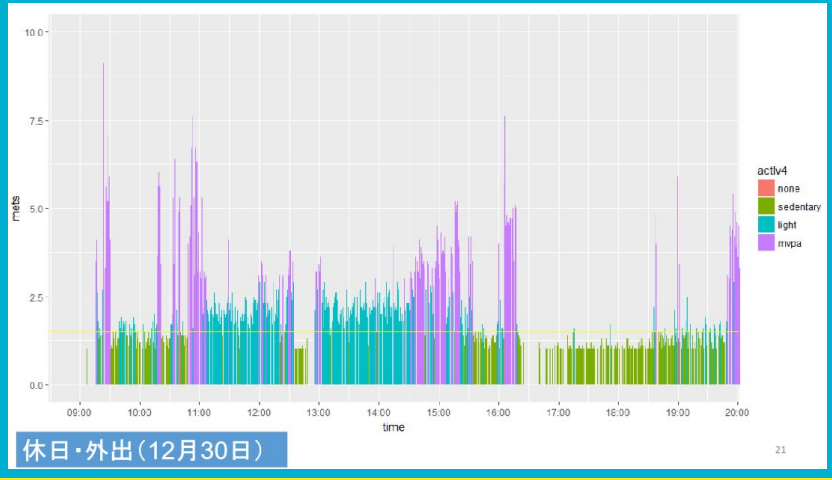


やわらかクジラ　:@matsuchiy

これまでのデータ加工経験

- 分野
  - 心理学
  - 疫学・公衆衛生学
- 扱うことの多いデータ
  - テーブル形式，データフレーム
  - 質問紙，心理尺度
  - 活動，生理指標
    - 個人ごとに大量の csv

	A	B	C	D	E	F	G	H	I	J	K	L
1	A1	A2	A3	A4	A5	C1	C2	C3	C4	C5	E1	E2
2	2	4	3	4	4	2	3	3	4	4	3	3
3	2	4	5	2	5	5	4	4	3	4	1	1
4	5	4	5	4	4	4	5	4	2	5	2	4
5	4	4	6	5	5	4	4	3	5	5	5	3
6	2	3	3	4	5	4	4	5	3	2	2	2
7	6	6	5	6	5	6	6	6	1	3	2	1
8	2	5	5	3	5	5	4	4	2	3	4	3
9	4	3	1	5	1	3	2	4	2	4	3	6
10	4	3	6	3	3	6	6	3	4	5	5	3
11	2	5	6	6	5	6	5	6	2	1	2	2
12	4	4	5	6	5	4	3	5	3	2	1	3
13	2	5	5	5	5	5	4	5	4	5	3	3
14	5	5	5	6	4	5	4	3	2	2	3	3
15	5	5	5	6	6	4	4	4	2	1	2	2



同人活動 （サークル名：ヤサイゼリー）

- 技術書典12にて頒布[1]
- 前編では，Rの基本知識とdplyrの基本動詞を解説
  - **ヘルパー関数**, **rename_with**, **across**が分かる人には不要な本
  - 逆に知らない人には効率化にすごく役立つはず

[1] pdf: https://techbookfest.org/product/5161487259664384?productVariantID=5672571053801472, html: https://izunyan.github.io/gisho12/

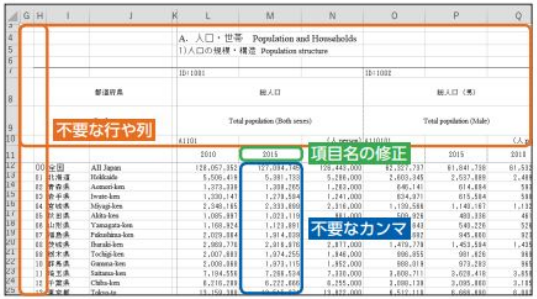
データ加工とは

- データ解析に入る前までの一連の準備
  - 例：前処理、データラングリング、データクリーニング、データクレンジング、データハンドリング　など

ない。このファイルに対して**表計算ソフトを用いて**次の操作を行う図表3, 4。

- 不要な行や列を削除する
- 項目名を修正する
- 不要なカンマが付与されないようセルの書式を「通貨」から「標準」に変更する
- プログラムでファイルを開くことができるよう「CSV UTF-8形式」で保存する
- 表記のゆれを修正する（大文字と小文字，西暦と和暦，正式名称と略称，空白の有無など）
- 不要なデータの注釈や空白文字を除去する
- 重複するデータを除去する

公開されているデータには，印刷することを前提に整形されているものも多く，このようなデータを，プログラムを用いて処理するには，このように前処理をする必要がある。



図表3　前処理が必要なデータ

	A	B	C	D	E	F
1	都道府県	総人口	出生数	死亡数	転入者数	転出者数
2	北海道	5381733	36695	60667	49407	57823
3	青森県	1308265	8621	17148	18162	24755
4	岩手県	1279594	8814	16502	18137	22430
5	宮城県	2333899	17999	23070	50024	49813
6	秋田県	1023119	5861	14794	11999	16473
7	山形県	1123891	7831	14960	13634	17663
8	福島県	1914039	14195	24205	29485	31552
9	茨城県	2916976	21700	31025	50399	58326
10	栃木県	1974255	15306	20519	34885	38607
11	群馬県	1973115	14256	21519	32038	32553
12	埼玉県	7266534	56077	62565	180451	162374
13	千葉県	6222666	47014	56079	155892	147853
14	東京都	13515271	113194	111673	456635	372404

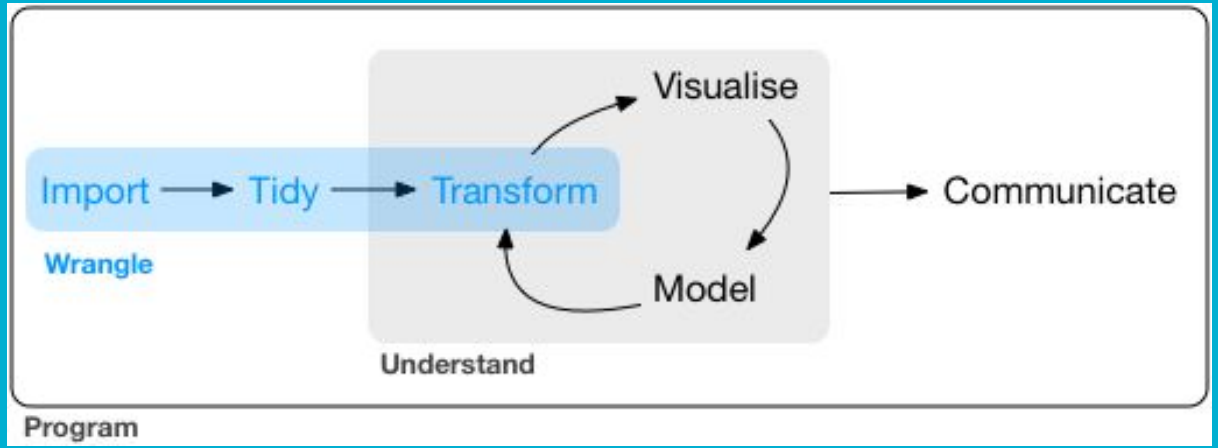
図表4　整形後のデータ

文部科学省 高等学校情報科「情報Ⅱ」教員研修用教材より [1]

[1] 第3章 情報とデータサイエンス 前半 https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/mext_00742.html

大人ならデータ加工も
プログラムで！

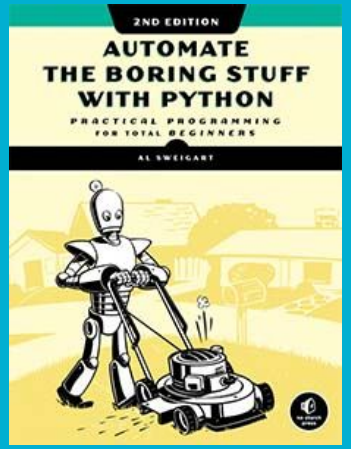
データ加工とは



Hadley Wickham & Garrett Grolemund, the website for “R for Data Science” https://r4ds.had.co.nz/wrangle-intro.html

「がんばらない」とは

- 単純作業のくり返しに無駄なエネルギーを注がない
- なるべく人力に頼らないようにすること



退屈なことは〇〇にやらせよう！

https://automatetheboringstuff.com/

Rと仲良くなるヒント（自分の場合）

- 有名なサンプルデータで練習
  - 自分のデータを読み込むのは，読み込まずにいられないぐらいに dplyrの基本動詞になれてから
- 以下の5つの動詞
  - 列（変数，カラム）を選ぶ: **select**
  - 変数名を変更する　　: **rename**
  - 行（ケース）を選ぶ　　: **filter**
  - 新しい変数（列）の作成　: **mutate**
  - 要約値を作る　　: **summarise**
- RStudio開いたら library(**tidyverse**)

本発表のコードはこれが前提

データ加工の練習開始

- サンプルデータになじもう
- penguinsデータを読み込み


```
df <-
  palmerpenguins::penguins

# データの表示
df
```

パッケージ名::関数など，の書き方で直接読みだせる

```
## # A tibble: 344 x 8
##   species island    bill_length_mm bill_depth_mm flipper_length_~
##   <fct>   <fct>              <dbl>         <dbl>             <int>
## 1 Adelie  Torgersen           39.1          18.7               181
## 2 Adelie  Torgersen           39.5          17.4               186
## 3 Adelie  Torgersen           40.3          18                 195
## # ... with 341 more rows, and 3 more variables:
## #   body_mass_g <int>, sex <fct>, year <int>
```

列(変数, カラム)を選ぶ:select

- がんばり例:延々と続く列, 気が遠くなるスクロール

列(変数, カラム)を選ぶ:select

- select()の中に列名を入れるだけ

データフレーム %>%
適用する関数()

パイプ(%>%)は今後, Rでデフォルトで使えるようになっている|> を多く目にするようになるかも

```
# library(tidyverse)

df %>%
  select(bill_length_mm, bill_depth_mm)
```

```
## # A tibble: 344 x 2
##   bill_length_mm bill_depth_mm
##            <dbl>         <dbl>
## 1           39.1          18.7
## 2           39.5          17.4
## 3           40.3          18
## # ... with 341 more rows
```

列(変数, カラム)を選ぶ:select

- 複数列名は柔軟に指定できる
- 文字による指定(“ ”で囲まれた文字列)は効率化の鍵

```
df %>%
  select(bill_length_mm:flipper_length_mm, sex)
```

```
## # A tibble: 344 x 4
##   bill_length_mm bill_depth_mm flipper_length_mm sex
##            <dbl>         <dbl>             <int> <fct>
## 1           39.1          18.7               181 male
## 2           39.5          17.4               186 female
## 3           40.3          18                 195 female
## # ... with 341 more rows
```

```
df %>%
  select("bill_length_mm", "bill_depth_mm")
```

```
## # A tibble: 344 x 2
##   bill_length_mm bill_depth_mm
##            <dbl>         <dbl>
## 1           39.1          18.7
## 2           39.5          17.4
## 3           40.3          18
## # ... with 341 more rows
```

列（変数，カラム）を選ぶ：select

- 外に出すとすっきりする，同じ情報の繰り返しを防げる

```
# あらかじめオブジェクト（ここではvars）に変数名の文字列を格納して後で使えるようにする
vars <- c("bill_length_mm", "bill_depth_mm")

df %>%
  select(all_of(vars))
```

文字列ベクトルのオブジェクトはall_of()で囲む

```
## # A tibble: 344 x 2
##   bill_length_mm bill_depth_mm
##            <dbl>         <dbl>
## 1           39.1          18.7
## 2           39.5          17.4
## 3           40.3          18
## # ... with 341 more rows
```

列（変数，カラム）を選ぶ：select

- **ヘルパー関数**で列名入力効率化

他にもends_with( ), contains( ), num_range( )など

```
df %>%
  select(starts_with("bill"))
```

```
## # A tibble: 344 x 2
##   bill_length_mm bill_depth_mm
##            <dbl>         <dbl>
## 1           39.1          18.7
## 2           39.5          17.4
## 3           40.3          18
## # ... with 341 more rows
```

```
df %>%
  select(matches("length|depth"))
```

matches()は
正規表現が使える

```
## # A tibble: 344 x 3
##   bill_length_mm bill_depth_mm flipper_length_mm
##            <dbl>         <dbl>             <int>
## 1           39.1          18.7               181
## 2           39.5          17.4               186
## 3           40.3          18                 195
## # ... with 341 more rows
```

変数名を変更する：rename

- がんばり例：手動でひたすら入力，置換で意図しないミス

```
df %>%
  rename(blmm = bill_length_mm)
```

new = old

```
## # A tibble: 344 x 8
##   species island blmm bill_depth_mm flipper_length_~ body_mass_g
##   <fct>   <fct>  <dbl>         <dbl>            <int>       <int>
## 1 Adelie  Torge~  39.1          18.7              181        3750
## 2 Adelie  Torge~  39.5          17.4              186        3800
## 3 Adelie  Torge~  40.3          18                195        3250
## # ... with 341 more rows, and 2 more variables: sex <fct>,
## #   year <int>
```

変数名を変更する：rename

- rename_with()で繰り返しを避ける

```
df %>%
  rename(くちばし_length_mm = bill_length_mm,
         くちばし_depth_mm = bill_depth_mm)
```

```
## # A tibble: 344 x 8
##   species island    くちばし_length_mm くちばし_depth_mm
##   <fct>   <fct>                  <dbl>             <dbl>
## 1 Adelie  Torgersen                39.1              18.7
## 2 Adelie  Torgersen                39.5              17.4
## 3 Adelie  Torgersen                40.3              18
## # ... with 341 more rows, and 4 more variables:
## #   flipper_length_mm <int>, body_mass_g <int>, sex <fct>,
## #   year <int>
```

適用する関数，直前に~（チルダ）が必要

```
df %>%
  rename_with(~str_replace(.x, "bill", "くちばし"),
              starts_with("bill"))
```

対象の列選択，↑の.xに入る

```
## # A tibble: 344 x 8
##   species island    くちばし_length_mm くちばし_depth_mm
##   <fct>   <fct>                  <dbl>             <dbl>
## 1 Adelie  Torgersen                39.1              18.7
## 2 Adelie  Torgersen                39.5              17.4
## 3 Adelie  Torgersen                40.3              18
## # ... with 341 more rows, and 4 more variables:
## #   flipper_length_mm <int>, body_mass_g <int>, sex <fct>,
## #   year <int>
```

行(ケース)を選ぶ:filter

- がんばり例:「あのカテゴリだけ見たいんだよね」

```
starwars
```

```
## # A tibble: 87 x 14
##   name    height  mass hair_color skin_color eye_color birth_year
##   <chr>    <int> <dbl> <chr>      <chr>      <chr>          <dbl>
## 1 Luke S~    172    77 blond      fair       blue            19
## 2 C-3PO      167    75 <NA>       gold       yellow         112
## 3 R2-D2       96    32 <NA>       white, bl~ red             33
## # ... with 84 more rows, and 7 more variables: sex <chr>,
## #   gender <chr>, homeworld <chr>, species <chr>, films <list>,
## #   vehicles <list>, starships <list>
```

- 例示しやすくするためspeciesを先頭にしたデータを作成

```
df_st <-
  starwars %>%
  select(species, name:homeworld)
```

```
df_st %>%
  filter(species == "Droid")
```

他にも<, <=, >, >=, != が使える

```
## # A tibble: 6 x 11
##   species name   height  mass hair_color skin_color  eye_color
##   <chr>   <chr>   <int> <dbl> <chr>      <chr>       <chr>
## 1 Droid   C-3PO     167    75 <NA>       gold        yellow
## 2 Droid   R2-D2      96    32 <NA>       white, blue red
## 3 Droid   R5-D4      97    32 <NA>       white, red  red
## 4 Droid   IG-88     200   140 none       metal       red
## 5 Droid   R4-P17     96    NA none       silver, red red, blue
## 6 Droid   BB8        NA    NA none       none        black
## # ... with 4 more variables: birth_year <dbl>, sex <chr>,
## #   gender <chr>, homeworld <chr>
```

行(ケース)を選ぶ:filter

- キーワード検索

```
df_st %>%
  filter(str_detect(name, "Luke"))
```

```
## # A tibble: 1 x 11
##   species name       height  mass hair_color skin_color eye_color
##   <chr>   <chr>       <int> <dbl> <chr>      <chr>      <chr>
## 1 Human   Luke Skyw~    172    77 blond      fair       blue
## # ... with 4 more variables: birth_year <dbl>, sex <chr>,
## #   gender <chr>, homeworld <chr>
```

```
df_st %>%
  filter(str_detect(name, "^Y|^L"))
```

stringr::str_系の関数は" "の中が正規表現

```
## # A tibble: 8 x 11
##   species   name     height  mass hair_color skin_color eye_color
##   <chr>     <chr>     <int> <dbl> <chr>      <chr>      <chr>
## 1 Human     Luke Sk~    172  77   blond      fair       blue
## 2 Human     Leia Or~    150  49   brown      light      brown
## 3 Yoda's s~ Yoda         66  17   white      green      brown
## 4 Human     Lando C~    177  79   black      dark       brown
## 5 Human     Lobot       175  79   none       light      blue
## 6 Quermian  Yarael ~    264  NA   none       white      yellow
## 7 Mirialan  Luminar~    170  56.2 black      yellow     blue
## 8 Kaminoan  Lama Su     229  88   none       grey       black
## # ... with 4 more variables: birth_year <dbl>, sex <chr>,
## #   gender <chr>, homeworld <chr>
```

新しい変数(列)の作成:mutate

```
df_bfi <-
  psychTools::bfi %>%
  as_tibble()          # 表示に便利なtibble形式に
```

- psychTools パッケージに入っている国際パーソナリティ項目プールからの2800名分のデータ
  - 質問項目が25問あり，5つの構成概念（ここでは因子という）に対応する項目への回答を足し合わせたスコアを計算する
  - 性，教育歴，年齢の変数もあり
- 項目に対し想定される因子（因子名の頭文字が変数名と対応）
  - Agree A1からA5
  - Conscientious　C1からC5
  - Extraversion E1からE5
  - Neuroticism　N1からN5
  - Openness　O1からO5

```
df_bfi %>%
  select(A1) %>%                        # A1のみを残す
  mutate(
    mean_a1 = mean(A1, na.rm = TRUE), # A1の平均値を作成（NAは除外）
    dif_a1_mean = A1 - mean_a1)       # 各個体のA1と平均値の差分を計算
```

new = 計算式

```
## # A tibble: 2,800 x 3
##      A1 mean_a1 dif_a1_mean
##   <int>   <dbl>       <dbl>
## 1     2    2.41      -0.413
## 2     2    2.41      -0.413
## 3     5    2.41       2.59
## # ... with 2,797 more rows
```

新しい変数(列)の作成:mutate

- がんばり例:型の変換で個々の変数をすべて記述
- across()で一括

```
df_bfi %>%
  select(gender, education) %>%
  mutate(gender = factor(gender),
         education = factor(education))
```

```
## # A tibble: 2,800 x 2
##   gender education
##   <fct>  <fct>
## 1 1      <NA>
## 2 2      <NA>
## 3 2      <NA>
## # ... with 2,797 more rows
```

```
df_bfi %>%
  mutate(across(c(gender, education),
                factor)) %>%
  select(gender, education)   # 結果表示のため冗長だが変わった変数だけselect
```

```
## # A tibble: 2,800 x 2
##   gender education
##   <fct>  <fct>
## 1 1      <NA>
## 2 2      <NA>
## 3 2      <NA>
## # ... with 2,797 more rows
```

新しい変数(列)の作成:mutate

- across()内ではヘルパー関数が使える

```
df_bfi %>%
  mutate(across(starts_with("n"),
                factor)) %>%
  select(starts_with("n"))   # 結果表示のため
```

```
## # A tibble: 2,800 x 5
##   N1    N2    N3    N4    N5
##   <fct> <fct> <fct> <fct> <fct>
## 1 3     4     2     2     3
## 2 3     3     3     5     5
## 3 4     5     4     2     3
## # ... with 2,797 more rows
```

要約値を作る：summarise

- がんばり例：すべての変数にすべての関数を適用
- across()と関数のリストで一括

new = 関数を変数に適用

```
df %>%
  summarise(blm_mean = mean(bill_length_mm, na.rm = TRUE),
            bdm_mean = mean(bill_depth_mm, na.rm = TRUE),
            blm_sd = sd(bill_length_mm, na.rm = TRUE),
            bdm_sd = sd(bill_depth_mm, na.rm = TRUE),
            blm_n  = sum(!is.na(bill_length_mm)),
            bdm_n  = sum(!is.na(bill_depth_mm)))
```

```
## # A tibble: 1 x 6
##   blm_mean bdm_mean blm_sd bdm_sd blm_n bdm_n
##      <dbl>    <dbl>  <dbl>  <dbl> <int> <int>
## 1     43.9     17.2   5.46   1.97   342   342
```

```
df %>%
  summarise(across(c(bill_length_mm, bill_depth_mm),
                   list(mean = ~mean(.x, na.rm = TRUE),
                        sd = ~sd(.x, na.rm = TRUE),
                        n = ~sum(!is.na(.x))))))
```

ここの文字が変数名につく

```
## # A tibble: 1 x 6
##   bill_length_mm_mean bill_length_mm_sd bill_length_mm_n
##                 <dbl>             <dbl>            <int>
## 1                43.9              5.46              342
## # ... with 3 more variables: bill_depth_mm_mean <dbl>,
## #   bill_depth_mm_sd <dbl>, bill_depth_mm_n <int>
```

まとめ

- まずはlibrary(tidyverse)
- dplyrの基本動詞に慣れる
  - select, rename, filter, mutate, summarise
- がんばらない書き方を使えば大量の列を一瞬で処理できる

Enjoy!