

ミクロ政治データ分析実習

第8回 データハンドリング（1）

そん じえひよん

宋 財 泱

関西大学総合情報学部

2021/6/3 (updated: 2021-05-28)

Tidyverseとパイプ演算子

Tidyverseの世界



データサイエンスのために考案された、強い信念と思想に基づいたRパッケージの集合

- {tidyverse}をインストールすることで導入可能
- Tidyverseに属するパッケージは思想、文法およびデータ構造を共有
 - {dplyr}、{tidyr}、{readr}、{ggplot2}など
- オブジェクトを **パイプ演算子** (`%>%`)で繋ぐ

Rのコードは `library(tidyverse)` で始めよう!

パイプ演算子

Tidyverseにおいてオブジェクトは `%>%` で繋がっている。

- 既存の書き方: **書き方と読み方が逆**
 - 一般的なプログラミング言語共通
 - 書き方: `print(sum(X))` (`print`、 `sum`、 `X` の順で書く)
 - 読み方1: `X` を `sum()` し、 `print()` する (コードの順番と逆)
 - 読み方2: `print()` する内容は `sum()` で、 `sum()` は `X` に対して行う (直感的でない読み方)
- Tidyverseな書き方: **書き方と読み方が一致**
 - 今どきのRの書き方
 - 書き方: `X %>% sum() %>% print()`
 - 読み方: `X` を `sum()` し、 `print()` する

パイプ演算子の仕組み

- `%>%` の左側を右側の最初の引数として渡すだけ
- `X %>% 関数(Y)` は `関数(X, Y)` と同じ
 - `X %>% sum(na.rm = TRUE)` は `sum(X, na.rm = TRUE)` と同じ
 - `X %>% sum(na.rm = TRUE) %>% print()` は `print(sum(X, na.rm = TRUE))` と同じ
- 二番目以降の引数として渡すことも可能（適宜、解説）

既存の書き方

```
X <- c(2, 3, 5, NA, 11)
print(sum(X, na.rm = TRUE))
```

```
## [1] 21
```

Tidyverseな書き方

```
library(tidyverse)
X %>% sum(na.rm = TRUE) %>% print()
```

```
## [1] 21
```

参考) R 4.1以降のパイプ演算子

R内蔵演算子としてパイプ演算子（|>）が追加

- 2021年5月リリースされたR 4.1以降実装
 - NIIオンライン分析システムのRは4.0.3
- 既存のパイプ演算子（%>%）はR内蔵演算子でなく、{magrittr}が提供する演算子
 - {magrittr}は{tidyverse}を読み込むと自動的に読み込まれる
- 使い方はほぼ同じ
 - ただし、演算子の左側のオブジェクトを右側の**第一引数**として渡す場合のみ
 - 第一引数以外の引数として渡す場合は使い方が異なる
- 今後、仕様変更がある可能性があるため、本講義では%>%を使用

%>% と |> の比較

渡す先が第一引数の場合

```
X %>% sum(na.rm = TRUE) %>% print() # {magrittr}のパイプ演算子  
X |> sum(na.rm = TRUE) |> print() # R内蔵のパイプ演算子
```

渡す先が第一引数でない場合

- 回帰分析の関数（`lm()`）の第一引数は回帰式のオブジェクト
- 使用するデータフレームを指定する `data` 引数にデータフレームを渡す場合

```
my_data %>%  
  lm(y ~ x1 + x2 + x3, data = .) # .で位置を指定  
my_data |> # やや複雑  
  {\(df) lm(y ~ x1 + x2 + x3, data = df)}()
```

{dplyr}: 列の抽出

{dplyr}とは



- 表形式データ (データフレームやtibble)を操作するパッケージ
- 第5回の講義で解説した行・列の抽出も簡単に可能
- {tidyverse}を読み込む際に自動的に読み込まれる
 - {tidyverse}はパッケージを集めたパッケージであり、{dplyr}もその一部

```
library(tidyverse)
```

実習用データ

Micro08.csv: 186カ国 の社会経済・政治体制のデータ

```
# Dataフォルダー内のMicro08.csvを読み込み、dfという名のオブジェクトとして作業環境に格納
df <- read_csv("Data/Micro08.csv")
df

## # A tibble: 186 x 18
##   Country   Population     Area     GDP     PPP GDP_per_capita PPP_per_capita
##   <chr>       <dbl>    <dbl>    <dbl>    <dbl>        <dbl>        <dbl>
## 1 Afghanis... 38928346 6.53e5 1.91e4  8.27e4        491.      2125.
## 2 Albania     2877797 2.74e4 1.53e4  3.97e4       5309.     13781.
## 3 Algeria     43851044 2.38e6 1.70e5  4.97e5       3876.     11324.
## 4 Andorra      77265 4.7 e2 3.15e3    NA          40821.      NA
## 5 Angola       32866272 1.25e6 9.46e4  2.19e5       2879.      6649.
## 6 Antigua ...  97929 4.4 e2 1.73e3  2.08e3      17643.     21267.
## 7 Argentina    45195774 2.74e6 4.50e5  1.04e6      9949.     22938.
## 8 Armenia      2963243 2.85e4 1.37e4  3.84e4      4614.     12974.
## 9 Australia    25499884 7.68e6 1.39e6  1.28e6      54615.    50001.
## 10 Austria     9006398 8.24e4 4.46e5  5.03e5      49555.    55824.
## # ... with 176 more rows, and 11 more variables: G7 <dbl>, G20 <dbl>,
## #   OECD <dbl>, HDI_2018 <dbl>, Polity_Score <dbl>, Polity_Type <chr>,
```

実習用データの確認

186行、18列のデータ (= 186カ国、18変数)

```
dim(df)
```

```
## [1] 186 18
```

変数の一覧

```
names(df)
```

```
## [1] "Country"          "Population"        "Area"           "GDP"
## [5] "PPP"               "GDP_per_capita"   "PPP_per_capita"  "G7"
## [9] "G20"               "OECD"             "HDI_2018"        "Polity_Score"
## [13] "Polity_Type"       "FH_PR"            "FH_CL"          "FH_Total"
## [17] "FH_Status"         "Continent"
```

各変数について

詳細は[教科書第18.2章](#)を参照

変数名	説明	変数名	説明
Country	国名	OECD	OECD加盟有無
Population	人口	HDI_2018	人間開発指数 (2018年)
Area	面積(km ²)	Polity_Score	政治体制のスコア
GDP	国内総生産 (ドル)	Polity_Type	政治体制
PPP	購買力平価国内総生産	FH_PR	政治的自由
GDP_per_capita	一人当たりGDP	FH_CL	市民的自由
PPP_per_capita	一人当たりPPP	FH_Total	FH_PR + FH_CL
G7	G7加盟有無	FH_Status	自由の状態
G20	G20加盟有無	Continent	大陸

列の選択（抽出）：書き方

select() 関数を使用

パイプを使わない書き方

```
select(データ, 変数名1, 変数名2, ...)
```

パイプを使う書き方

データ %>%

```
select(変数名1, 変数名2, ...)
```

列の選択（抽出）：例

df から Country、Population、HDI_2018 列を抽出し、最初の5行のみ出力

```
df %>%
  select(Country, Population, HDI_2018) %>%
  head(n = 5)
```

```
## # A tibble: 5 x 3
##   Country     Population  HDI_2018
##   <chr>        <dbl>      <dbl>
## 1 Afghanistan 38928346  0.496
## 2 Albania     2877797   0.791
## 3 Algeria     43851044  0.759
## 4 Andorra      77265    0.857
## 5 Angola       32866272  0.574
```

この時点では抽出・出力されただけ。抽出した結果を df2 という名で作業環境内に格納するためには

```
df2 <- df %>%
  select(Country, Population, HDI_2018)
```

列の選択と変数名の変更

変数名の変更と抽出を同時にを行うことも可能

- 新しい変数名 = 既存の変数名

例) HDI_2018 の変数名を HDI に変更

```
df %>%
  select(Country, Population, HDI = HDI_2018) %>%
  head(n = 5)
```

```
## # A tibble: 186 x 3
##   Country           Population     HDI
##   <chr>              <dbl> <dbl>
## 1 Afghanistan      38928346  0.496
## 2 Albania          2877797   0.791
## 3 Algeria          43851044  0.759
## 4 Andorra          77265    0.857
## 5 Angola            32866272  0.574
## 6 Antigua and Barbuda 97929   0.776
## 7 Argentina         45195774  0.83
## 8 Armenia           2963243   0.76
## 9 Australia         25499884  0.938
```

抽出せず、変数名のみ変更

rename() 関数を使用

データ %>%

rename(既存の変数名 = 元の変数名)

例) Population を Jinko に、 Area を Menseki に変更

```
df %>%  
  rename(Jinko = Population, Menseki = Area)
```

```
## # A tibble: 186 x 18  
##   Country      Jinko    Menseki     GDP      PPP  GDP_per_capita  PPP_per_capita  
##   <chr>       <dbl>     <dbl>     <dbl>     <dbl>        <dbl>        <dbl>  
## 1 Afghanistan 3.89e7  652860  1.91e4  8.27e4        491.       2125.  
## 2 Albania     2.88e6   27400  1.53e4  3.97e4       5309.      13781.  
## 3 Algeria     4.39e7  2381740  1.70e5  4.97e5       3876.      11324.  
## 4 Andorra      7.73e4    470  3.15e3    NA          40821.        NA  
## 5 Angola       3.29e7 1246700  9.46e4  2.19e5       2879.      6649.  
## 6 Antigua and... 9.79e4     440  1.73e3  2.08e3       17643.     21267.  
## 7 Argentina    4.52e7 2736690  4.50e5  1.04e6       9949.     22938.  
## 8 Armenia      2.96e6   28470  1.37e4  3.84e4       4614.     12974.  
## 9 Australia    2.55e7 7682300  1.39e6  1.28e6      54615.    5060/139
```

列の除外

変数名の前に !、または - を付ける

- {dplyr}公式では ! を推奨
- 2つ以上の変数を除外する場合、変数名を c() でまとめる。

例) df から GDP_per_capita と PPP_per_capita を除外

```
df %>%
  select( !c(GDP_per_capita, PPP_per_capita) ) %>%
  head(n = 5)
```

```
## # A tibble: 186 x 16
##   Country     Population     Area     GDP     PPP     G7     G20   OECD HDI_2018
##   <chr>       <dbl>      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>  <dbl>    <dbl>
## 1 Afghanistan 38928346  652860 1.91e4  8.27e4    0     0     0     0    0.496
## 2 Albania     2877797   27400 1.53e4  3.97e4    0     0     0     0    0.791
## 3 Algeria     43851044 2381740 1.70e5  4.97e5    0     0     0     0    0.759
## 4 Andorra      77265     470 3.15e3    NA      0     0     0     0    0.857
## 5 Angola       32866272 1246700 9.46e4  2.19e5    0     0     0     0    0.574
## 6 Antigua an... 97929      440 1.73e3  2.08e3    0     0     0     0    0.776
## 7 Argentina    45195774 2736690 4.50e5  1.04e6    0     1     0     0    0.83
## 8 Armenia      2963243   28470 1.37e4  3.84e4    0     0     0     0    0.76
## # ... with 178 more rows, and 10 more variables:
```

隣接する列の同時選択

: を使用

- Country から PPP までの列: Population:PPP
- Country:PPP は Country, Population, Area, GDP, PPP と同じ意味

例) df から Country ~ PPP, HDI_2018 列を抽出

```
df %>%
  select(Country:PPP, HDI_2018) %>%
  print(n = 5)
```

```
## # A tibble: 186 x 6
##   Country      Population     Area     GDP     PPP HDI_2018
##   <chr>        <dbl>     <dbl>    <dbl>    <dbl>    <dbl>
## 1 Afghanistan 38928346  652860 19101.  82737.  0.496
## 2 Albania     2877797   27400  15278.  39658.  0.791
## 3 Algeria     43851044 2381740 169988. 496572.  0.759
## 4 Andorra      77265     470    3154.    NA      0.857
## 5 Angola       32866272 1246700 94635.  218533.  0.574
## 6 Antigua and Barbuda 97929     440    1728.  2083.   0.776
## 7 Argentina    45195774 2736690 449663. 1036721.  0.83
## 8 Armenia      2963243   28470   13673.  38446.  0.76
```

高度な変数選択

- 特定の文字列で始まる列を選択: `starts_with()`
 - 例) FHで始まる列の選択: `starts_with("FH")`
- 特定の文字列で終わる列を選択: `ends_with()`
- 特定の文字列を含む列を選択: `contains()`

例) df から Country , "FH" で始まる列を抽出

```
df %>%
  select(Country, starts_with("FH")) %>%
  head(n = 5)
```

```
## # A tibble: 5 x 5
##   Country      FH_PR  FH_CL  FH_Total FH_Status
##   <chr>        <dbl>  <dbl>    <dbl>   <chr>
## 1 Afghanistan     13     14      27  NF
## 2 Albania         27     40      67  PF
## 3 Algeria          10     24      34  NF
## 4 Andorra          39     55      94  F
## 5 Angola           11     21      32  NF
```

- 応用) `!starts_with("FH")`: "FH" で始まる列を除外

列の順番変更: select() 使用

抽出後のデータフレームにおける変数は `select()` 内で指定された順番に

例) G7 から OECD 列を Country と Population の間へ移動

```
df %>%
  select(Country, G7:OECD,
         Population:PPP_per_capita, HDI_2018:Continent)
```

```
## # A tibble: 186 x 18
##   Country      G7     G20    OECD Population     Area     GDP     PPP
##   <chr>     <dbl>  <dbl>  <dbl>     <dbl>   <dbl>   <dbl>   <dbl>
## 1 Afghanistan     0     0     0     38928346   652860 19101. 8.27e4
## 2 Albania          0     0     0     2877797    27400 15278. 3.97e4
## 3 Algeria          0     0     0     43851044  2381740 169988. 4.97e5
## 4 Andorra          0     0     0      77265     470   3154. NA
## 5 Angola            0     0     0     32866272  1246700 94635. 2.19e5
## 6 Antigua and Barbuda     0     0     0      97929     440   1728. 2.08e3
## 7 Argentina         0     1     0     45195774  2736690 449663. 1.04e6
## 8 Armenia           0     0     0     2963243    28470 13673. 3.84e4
## 9 Australia          0     1     1     25499884  7682300 1392681. 1.28e6
## 10 Austria          0     0     1     9006398    82409 446315. 5.03e5
## # ... with 176 more rows, and 10 more variables: GDP_per_capita <dbl>, 20 / 39
```

列の順番変更: relocate() 使用

relocate() の使い方

- .after = XXX: XXX の後ろへ移動
- .before = XXX: XXX の前へ移動

データ %>%

```
relocate(移動したい変数名, .after = 変更先)
```

例) G7 から OECD 列を Country の後ろへ移動

```
df %>%  
  relocate(G7:OECD, .after = Country) # .before = PopulationもOK
```

```
## # A tibble: 186 x 18  
##   Country          G7     G20    OECD Population      Area     GDP     PPP  
##   <chr>        <dbl>  <dbl>  <dbl>      <dbl>    <dbl>    <dbl>  <dbl>  
## 1 Afghanistan      0      0      0    38928346  652860  19101.  8.27e4  
## 2 Albania           0      0      0    2877797   27400  15278.  3.97e4  
## 3 Algeria            0      0      0    43851044  2381740 169988.  4.97e5  
## 4 Andorra            0      0      0      77265     470   3154.  NA  
## 5 Angola             0      0      0    32866272  1246700  94635.  2.19e5  
## 6 Antigua and Barb... 0      0      0      97929     440   1728.  2.08e39
```

{dplyr}: 行の抽出

行の抽出: 書き方

filter() 関数を使用

パイプを使わない書き方

```
filter(データ, 条件1, 条件2, ...)
```

パイプを使う書き方

データ %>%

```
select(条件1, 条件2, ...)
```

行の抽出: 例

例) df から Continent が "Europe" の行を抽出し、Country ~ PPP, HDI_2018 列を抽出し、HDI_2018 は HDI に変更

- filter() と select() の組み合わせ
- 以下の例の場合、filter() と select() の順番を逆にすることは不可
 - select() 後、Continent 変数がなくなるため

```
df %>%  
  filter(Continent == "Oceania") %>%  
  select(Country:PPP, HDI = HDI_2018)  
  
## # A tibble: 4 x 6  
##   Country           Population     Area      GDP     PPP     HDI  
##   <chr>             <dbl>     <dbl>    <dbl>    <dbl>    <dbl>  
## 1 Australia        25499884  7682300 1392681. 1275027. 0.938  
## 2 Fiji              896445    18270    5536.    12496.   0.724  
## 3 New Zealand      4842780   263820   206929.  204260.  0.921  
## 4 Papua New Guinea 8947024   452860   24970.   37319.   0.543
```

行の抽出: 2つ以上の条件 (AND)

2つ以上の条件を同時に満たす行を抽出

- で条件式を追加するだけ (& もOK)

例) df から Continent が "Asia" (条件1)、 HDI_2018 が 0.8 以上 (条件2) の行を抽出し、 Country と HDI_2018 列を抽出

```
df %>%  
  filter(Continent == "Asia", HDI_2018 >= 0.85) %>%  
  select(Country, HDI_2018)  
  
## # A tibble: 6 x 2  
##   Country          HDI_2018  
##   <chr>            <dbl>  
## 1 Israel           0.906  
## 2 Japan            0.915  
## 3 South Korea      0.906  
## 4 Saudi Arabia     0.857  
## 5 Singapore         0.935  
## 6 United Arab Emirates 0.866
```

行の抽出: 2つ以上の条件 (OR)

2つ以上の条件を片方か両方に満たす行を抽出

- | で条件式を追加するだけ

例) df から Continent が "Asia" (条件1) か "Oceania" (条件2) であり、 HDI_2018 が 0.9 以上 (条件3) の行を抽出し、 Country と HDI_2018 、 Continent 列を抽出

```
df %>%
  filter((Continent == "Asia" | Continent == "Oceania"),
         HDI_2018 >= 0.9) %>%
  select(Country, HDI_2018, Continent)
```

```
## # A tibble: 6 x 3
##   Country      HDI_2018 Continent
##   <chr>        <dbl>   <chr>
## 1 Australia    0.938  Oceania
## 2 Israel       0.906  Asia
## 3 Japan        0.915  Asia
## 4 South Korea  0.906  Asia
## 5 New Zealand  0.921  Oceania
## 6 Singapore    0.935  Asia
```

参考) %in%演算子

%in%: | の代わりに使用可能な便利な演算子

例) Continent の値が c("Asia", "Oceania") の要素に含まれている場合

```
df %>%
  filter(Continent %in% c("Asia", "Oceania"), HDI_2018 >= 0.9) %>%
  select(Country, HDI_2018, Continent)
```

```
## # A tibble: 6 x 3
##   Country      HDI_2018 Continent
##   <chr>          <dbl>   <chr>
## 1 Australia     0.938  Oceania
## 2 Israel        0.906  Asia
## 3 Japan          0.915  Asia
## 4 South Korea   0.906  Asia
## 5 New Zealand   0.921  Oceania
## 6 Singapore     0.935  Asia
```

欠損値が含まれた行の扱い

df の PPP が欠損している行を抽出し、Country から PPP 列まで出力

- 変数名 == NA を条件にしてはいけない

```
df %>%
  filter(PPP == NA) %>%
  select(Country:PPP)
```

```
## # A tibble: 0 x 5
## # ... with 5 variables: Country <chr>, Population <dbl>, Area <dbl>,
## #     GDP <dbl>, PPP <dbl>
```

欠損値が含まれた行の扱い

df の PPP が欠損している行を抽出し、Country から PPP 列まで出力

- 正解: is.na(変数名)

```
df %>%  
  filter(is.na(PPP)) %>%  
  select(Country:PPP)
```

```
## # A tibble: 8 x 5  
##   Country      Population     Area     GDP     PPP  
##   <chr>          <dbl>     <dbl>    <dbl>    <dbl>  
## 1 Andorra       77265      470    3154.     NA  
## 2 Cuba          11326616  106440  100023     NA  
## 3 Holy See       801        0      NA      NA  
## 4 Liechtenstein  38128      160    6553.     NA  
## 5 Monaco         39242       1    7188.     NA  
## 6 Somalia        15893222  627340    917.     NA  
## 7 Syria          17500658  183630  40405.    NA  
## 8 Western Sahara  597339   266000    909.     NA
```

欠損値が含まれた行の除外

df の PPP が欠損している行を除外し、Country から PPP 列まで出力

- 否定を意味する ! を使用する

```
df %>%  
  filter(!is.na(PPP)) %>%  
  select(Country:PPP)
```

```
## # A tibble: 178 x 5  
##   Country      Population     Area     GDP     PPP  
##   <chr>        <dbl>       <dbl>    <dbl>    <dbl>  
## 1 Afghanistan 38928346  652860 19101.  82737.  
## 2 Albania     2877797   27400 15278.  39658.  
## 3 Algeria     43851044 2381740 169988. 496572.  
## 4 Angola       32866272 1246700 94635.  218533.  
## 5 Antigua and Barbuda 97929      440 1728.  2083.  
## 6 Argentina   45195774 2736690 449663. 1036721.  
## 7 Armenia      2963243   28470 13673.  38446.  
## 8 Australia   25499884 7682300 1392681. 1275027.  
## 9 Austria      9006398   82409 446315.  502771.  
## 10 Azerbaijan 10139177  82658 48048.  144556.  
## # ... with 168 more rows
```

もう一つの方法

drop_na() 関数を利用

- () 内で指定した変数が欠損している行をすべて除外（複数指定可）

```
df %>%  
  drop_na(PPP, Polity_Score) %>% # PPPとPolity_Scoreどちらか欠損した行を除外  
  select(Country:PPP, Polity_Score)
```

```
## # A tibble: 155 x 6  
##   Country     Population      Area      GDP      PPP Polity_Score  
##   <chr>        <dbl>       <dbl>     <dbl>     <dbl>       <dbl>  
## 1 Afghanistan 38928346  652860    19101.    82737.      -1  
## 2 Albania      2877797   27400    15278.    39658.       9  
## 3 Algeria      43851044  2381740   169988.   496572.      2  
## 4 Angola        32866272  1246700   94635.    218533.     -2  
## 5 Argentina    45195774  2736690   449663.   1036721.      9  
## 6 Armenia       2963243    28470    13673.    38446.       7  
## 7 Australia     25499884  7682300   1392681.  1275027.      10  
## 8 Austria       9006398    82409    446315.   502771.      10  
## 9 Azerbaijan   10139177   82658    48048.    144556.     -7  
## 10 Bahrain      1701575     760    38574.    74230.     -10  
## # ... with 145 more rows
```

{dplyr}: 行のソート

行のソート: 書き方

arrange() 関数を使用

パイプを使わない書き方

```
arrange(データ, 変数名1, 変数名2, ...)
```

パイプを使う書き方

```
データ %>% arrange(変数名1, 変数名2, ...)
```

基本的には昇順 (小さい行が先にくる)

- 降順にする場合は desc(変数名)
- 変数名1 を基準にソートし、同点の場合は 変数名2 を基準に

行のソート: 例

例) df から Continent の値が "Africa" の行のみを抽出し、Polity_Score が高い行を上位にする。そして、Country と PPP_per_capita、Polity_Score 列のみ残す。

- Polity Scoreが高い(低い)=より民主主義(権威主義)に近い

```
df %>%
  filter(Continent == "Africa") %>%
  arrange(desc(Polity_Score)) %>%
  select(Country, PPP_per_capita, Polity_Score)
```

```
## # A tibble: 54 x 3
##   Country      PPP_per_capita Polity_Score
##   <chr>          <dbl>           <dbl>
## 1 Mauritius    22637.          10
## 2 Kenya         4105.           9
## 3 South Africa 12605.          9
## 4 Botswana     17311.          8
## 5 Ghana         5097.           8
## 6 Lesotho       3019.           8
## 7 Benin         3067.           7
## 8 Liberia       1461.           7
## 9 Nigeria       5018.           7
```

行のソート: 応用

- df からアフリカのみを抽出し、Polity_Score が低い行を上位に
- Polity_Score が同点の場合、PPP_per_capita が高い行を上位に
- Country と Polity_Score, PPP_per_capita 列のみ残す
- Polity_Score は Polity に、PPP_per_capita は PPP と名前を変更

```
df %>%  
  filter(Continent == "Africa") %>%  
  arrange(Polity_Score, desc(PPP_per_capita)) %>%  
  select(Country, Polity = Polity_Score, PPP = PPP_per_capita)
```

```
## # A tibble: 54 x 3  
##   Country          Polity     PPP  
##   <chr>            <dbl>    <dbl>  
## 1 Eswatini           -9    8634.  
## 2 Eritrea            -7    1860.  
## 3 Equatorial Guinea -6   19458.  
## 4 Egypt              -4   11198.  
## 5 Morocco             -4    7554.  
## 6 Sudan               -4    4063.  
## 7 Cameroon            -4    3506.  
## 8 Congo (Brazzaville) -4    3191.
```

まとめ

今回の内容

よく分からぬ箇所は教科書を読み返す or 宋&TAに質問 (できれば、LMSの質問コーナーで)

- パイプ演算子: 教科書第13.2章
- 列の抽出: 教科書第13.2章
- 行の抽出: 教科書第13.3章
- 行のソート: 教科書第13.4章

次回(第9～10回)の内容

- データのグルーピングと要約: 教科書第14.1章と第14.2章
- 変数の計算: 教科書第14.3章と第14.4章
- データの結合: 教科書第14.5章
- Factor型変数の扱い: 教科書第15章
- 整然データ構造: 教科書第16章

課題

1. 今回講義用のプロジェクトを作成する。
2. LMSからデータ（.csv）問題ファイル（.Rmd）とサンプルファイル（.html）をダウンロードし、プロジェクトのフォルダーに保存する。
 - ファイル名は変更しないこと
3. プロジェクトからRStudioを起動し、.Rmdファイルを開く
 - NIIオンライン分析システムの場合、RStudio起動>プロジェクトを開く>.Rmdファイルを開く
4. サンプルファイルと同じ結果が得られるようにR Markdown文書を作成する。
5. 隨時Knitし、結果を確認する。
 - Source Pane上段のKnitをクリックするか、「⌘ + Shift + K」(macOS)、「Ctrl + Shift + K」(Windows)を推す。
6. .Rmdファイルを関大LMSに提出する。
7. 期限は2021年6月5日（土）の23時59分とする。
 - 時間に余裕を持って取り組むこと。期限直前に取り組み始めてPCトラブルがあっても期限延長はない
8. 答案は次回の講義までに公開する。