Some Programming Tips

RA Bootcamp Day 2

柳本和春
kazuharu.yanagimoto@cemfi.edu.es
CEMFI

August 30, 2023

良いコードとは何か

> **読みやすさの基本的定理 (Boswell and Foucher 2011)**
>
> コードは他の人が最短時間で理解できるように書かなければいけない

読みやすさを上げるTips

- カラフルなネーミング
- DRY原則
- パワフルな関数
- 戦略的なコメント
- 脳のメモリアロケーションの最小化

カラフルなネーミング

変数の中身 (型) がコードブックを読まなくても類推できるようにすると読みやすい

- **ブール代数**: `is_female`, `has_kids`, など
- **数値**: `hours_worked`, `num_child` などは数値であることが想定しやすい
- **カテゴリ変数**: `industry_c8`, `emp_c3` などカテゴリ数を明示する

逆に, `child` などは子供数なのか, 子供の有無なのか, 何か別の意味なのか分からない

Do not Repeat Yourself (DRY)

二回書くなら関数にする

- コードをコピペしたことは, 他の人は見ていない
- 誰かが変更に気づいたときに, コピー先まで変更してくれるとは限らない
- 関数化していない部分は, 同じ処理なのかチェックしないといけない

ちなみに, DRY原則を満たしていないコードを Write Everything Twice (WET) と呼ぶ

パワフルな関数

“五指, 手のひら, 手の甲と手首を覆う布を身につける” ⇒ “手袋をはめる”

```
df |>
  rename(gender = sex) |>
  select(wage, hour, year, gender) |>
  filter(year >= 1985, year <= 2000) |>
  group_by(year, gender) |>
  summarize(hour = mean(hour),
            wage = mean(wage),
            .groups = "drop")
```

```
df |>
  select(wage, hour, year, gender = sex) |>
  filter(between(year, 1985, 2000)) |>
  summarize(across(c(hour, wage), mean),
            .by = c("year", "gender"))
```

その他, `recode_factor()`, `case_when()`, `janitor::clean_names()` など

戦略的なコメント

~~コメントで説明する~~ ⇒ コードで説明できない時の最終手段

- コメントがなくても十分伝わるなら, コメントは邪魔なだけ
- 見出しやセクションなどの機能的なコメント以外は, 最小限に

バイリンガルコメント

- 日本語のコメント
  - → 作業中のコメント. 最終的なコードに残さない
  - → 考え方の整理や翌日の作業内容をメモする
- 英語のコメント
  - → 正式なコメント. 最終的なコードに残す
  - → 最終的なコードは英語しか残らないので, 誰にでもシェアできる

脳のメモリアロケーションの最小化: 不必要な再代入

```r
data  <- data |>
  filter(year <= 1985)

#---------------------
# 中略
#---------------------

data <- data |>
  filter(year >= 2000)
```

```r
data_m1985  <- data |>
  filter(year <= 1985)

#---------------------
# 中略
#---------------------

data_1985_2000 <- data |>
  filter(between(year, 1985, 2000))
```

- データフレームの再代入は, 行ごとのデータフレームの状態という脳の容量を消費する
- デバッグ中も, コードの実行順を意識しないといけない

脳のメモリアロケーションの最小化: 不必要な中間変数

```
df |>
  mutate(x2 = x^2,
         y = x2 + 1)
```

```
df |>
  mutate(y = x^2 + 1)
```

- x2という列はyという列を作るための一時的な列である
- 読み手にとっては, 今後x2という列が使われるか分からないので, x2の定義を覚えながら読み進める必要がある
- 一時的な列で今後使わない列名 (または変数) はtmpなどの命名によって明示的にする

References

Boswell, Dustin, and Trevor Foucher. 2011. *The Art of Readable Code*. 1st ed. Theory in Practice. Sebastopol, Calif: O’Reilly.