SASのdata stepをRで再現してみる

DS部会2024年度継続タスクフォース11
坂上 拓(中外製薬株式会社)

動機

製薬協

- **SASからプログラミング言語に触れた人は、その特殊性から他のプログラミング言語に移行し難い。**
  - **臨床試験データをハンドリングする際、頭の中で構築するアルゴリズムがSAS主体のため、データステップでできることが、他の言語でスムーズにできずイライラする。**
  - **データ結合の方法が特殊(set, merge)で、データの内部結合・外部結合（left, outer join）の概念がしっくりこない。**
  - **Data stepのレコード単位の処理に慣れてしまい、data frameのカラム単位の処理に混乱する。**

SASからRへ移行のための逆引き辞典のようなものが欲しいと思い作成したものを一部共有。

目的

製薬協

- あまり深く考えず、SASとRのアルゴリズムの対応をパターン化して考える
- 新規導出変数のための関数（mutate関数）の使い方を理解する

Agenda

製薬協

- Group内で連番を振る
- Groupingの注意点
- SASのdata step last, firstを再現する
- SAS formatを再現する
- LOCFする
- mutate関数を理解する

Group内で連番を振る

ADSLに含まれる女性（SEX = ‘F’）患者で、施設番号（SITEID）毎に被験者番号（USUBJID）順に連番（groupindex）を設定する。

SAS

```
proc sort data=ana.adsl out=work.adsl;
  where(sex = 'F');
  by siteid usubjid;
run;
```

```
data work.adsl2;
  set work.adsl;

  by siteid usubjid;

  if(first.siteid = 1) then
    groupindex = 1;
  else
    groupindex + 1;

run;
```

R

```
library(tidyverse)


dfADSL2 <- dfADSL |>
  filter(SEX == 'F') |>
  arrange(SITEID, USUBJID) |>
  group_by(SITEID) |>
  mutate(groupindex = row_number())
```

USUBJID	SEX	SITEID	groupindex
SUBJ005	F	29	1
SUBJ013	F	34	1
SUBJ023	F	36	1
SUBJ020	F	39	1
SUBJ025	F	39	2
SUBJ030	F	39	3

Groupingの注意点

製薬協

- **Grouping情報はデータフレーム情報として保持される**
  - **group_by関数でgroupingをした場合、grouping情報はその処理で生成されたデータフレームに保持される。このため、当該データフレームはgroupingされた状態でハンドリングされる（年齢の要約統計量はSITEID毎に算出される）**

group_by関数でgroupingする前のデータフレーム（dfADSL）

```
> glimpse(dfADSL)
Rows: 30
Columns: 33
$ STUDYID  <chr> "STUDY01", "STUDY01", "STUDY01", "STUDY01", "STUDY01", "STUDY01",
$ USUBJID  <chr> "SUBJ001", "SUBJ002", "SUBJ003", "SUBJ004", "SUBJ005", "SUBJ006",
$ AGE      <dbl> 79, 75, 84, 71, 28, 27, 48, 74, 24, 26, 33, 49, 68, 30, 73, 79, 59
```

group_by関数でgroupingした後のデータフレーム（dfADSL2）

```
> glimpse(dfADSL2)
Rows: 17
Columns: 34
Groups: SITEID [14]
$ STUDYID    <chr> "STUDY01", "STUDY01", "STUDY01", "STUDY01", "STUDY01", "STUDY01",
$ USUBJID    <chr> "SUBJ022", "SUBJ001", "SUBJ018", "SUBJ028", "SUBJ026", "SUBJ008",
$ AGE        <dbl> 51, 79, 74, 25, 56, 74, 59, 27, 49, 28, 68, 73, 41, 62, 83, 48, 46
```

Groupingの注意点

製薬協

- Grouping情報はデータフレーム情報として保持される
  - Grouping情報を保持したくない場合は、ungroup関数を使ってgrouping情報をリセットする必要がある

特に何も指定しなくても勝手にSITEIDでgroupingして平均値を算出

```
> dfADSL2 |> summarise(mean = mean(AGE))
# A tibble: 14 × 2
   SITEID  mean
    <db1> <db1>
 1      4    51
 2      6    79
 3     11    74
 4     16    25
 5     19    56
 6     20    74
 7     25    59
 8     26    38
 9     29    28
10     34    68
11     36    73
12     39    62
13     44    48
14     49    46
```

ungroup関数でgrouping情報をリセットするとAGEの平均値を算出できるようになる

```
> dfADSL2 |> ungroup() |> summarise(mean = mean(AGE))
# A tibble: 1 × 1
   mean
  <db1>
1  55.5
```

SASのdata step last, firstを再現する

製薬協

ラボデータからALT（PARAMCD = ‘ALT’）を取り出し、被験者毎（USUBJID）に最も早期に検査したもの（ADYの最も小さいもの）レコード取り出し出力。

SAS

```
proc sort data=ana.adlb out=work.adlb;
  where(paramcd = 'ALT');
  by usubjid ady;
run;
```

```
data work.first_alt;
  set work.adlb;

  by usubjid ady;

  if(first.usubjid = 1) then
    output;

run;
```

R

```
dfADLB2 <- dfADLB |>
  filter(PARAMCD == 'ALT') |>
  arrange(USUBJID, ADY) |>
  group_by(USUBJID) |>
  summarise_all(first)
```

USUBJID	AVAL	PARAMCD	ADY
SUBJ001	3.1813377304	ALT	22
SUBJ001	63.873935665	ALT	22
SUBJ004	35.696072314	ALT	22
SUBJ005	96.918138473	ALT	8
SUBJ005	68.919188875	ALT	22
SUBJ006	70.832950093	ALT	29

SUBJ001のようなケースの場合（ADY=22でレコードある）場合は、元の並び順が優先して1レコード取得されるため注意。
※左のSAS codeも同じ挙動

SASのdata step last, firstを再現する

ラボデータからALT（PARAMCD = ‘ALT’）を取り出し、被験者毎（USUBJID）に最も早期に検査したもの（ADYの最も小さいもの）レコード取り出しに出力。

SUBJ001のようなケースの場合（ADY=22でレコードある）場合に、患者毎に全てのADY最小レコードを取り出したい場合は？

R

```
dfADLB2 <- dfADLB |>
  filter(PARAMCD == 'ALT') |>
  arrange(USUBJID, ADY) |>
  group_by(USUBJID) |>
  filter(ADY == first(ADY))
```

USUBJID	AVAL	PARAMCD	ADY
SUBJ001	3.1813377304	ALT	22
SUBJ001	63.873935665	ALT	22
SUBJ004	35.696072314	ALT	22
SUBJ005	96.918138473	ALT	8
SUBJ005	68.919188875	ALT	22
SUBJ006	70.832950093	ALT	29

firstは関数で、SASのlastの再現にはlast関数が使える。
また、nth関数(n番目)といった指定もできる。

SAS formatを再現する

製薬協

ADSLの性別（SEX）に、フォーマットを当て（M→Male, F→Female）、新規変数（fmtsex）を作成する

SAS

```
proc format ;
  value $sexf
    'M' = 'Male'
    'F' = 'Female';

 run;
```

```
data work.adsl2;
  set ana.adsl;

  length fmtsex $4;

  fmtsex = put(sex, $sexf.);

run;
```

R

```
sexfmt <- c('M' = 'Male', 'F' = 'Female')
```

```
dfADSL2 <- dfADSL |>
  mutate(FMTSEX = sexfmt[SEX])
```

SAS formatを再現する

ADSLの年齢(AGE)に、フォーマットを当て(65歳未満(0 -< 65)、65歳以上75歳未満(65 -< 75)、75歳以上(75 -))のカテゴリ新規変数(agecat)を作成する

SAS

```
proc format ;

  value agef
    0 -< 65 = '0 -< 65'
   65 -< 75 = '65 -< 75'
   75 - HIGH   = '75 -';

run;


data work.adsl2;
  set ana.adsl;

  length agecat $8;

  agecat = put(age, agef.);

run;
```

R

```
dfALSL2 <- dfADSL |>
  mutate(agecat = cut(AGE,
                      breaks = c(0, 65, 75, Inf),
                      labels = c('0 -< 65', '65 -< 75', '75 -'),
                      right = FALSE))
```

cut関数のbreaksパラメータで区間(0 – 65, 65 – 75, 75 – Inf)を設定し、labelsパラメータで該当区間に対するラベルを定義する。
rightsパラメータは区間の右側(0 – 65, 65 – 75, 75 – Inf)をそのカテゴリに含めるかどうか(含める:TRUE, 含めない:FALSE)?

LOCFする

ADLBのALT（PARACD = ‘ALT’）の測定結果（AVAL）が欠損値の場合、直前のレコードの測定結果で補完する。

SAS

```
proc sort data=ana.adlb out=work.adlb;
  where(paramcd = 'ALT');
  by usubjid ady;
run;
```

```
data work.adlb2;
  set work.adlb;

  by usubjid ady;

  retain paval . pchg .;

  if(first.usubjid = 1) then
      paval = .;

  if(aval = .) then
        aval = paval;
  else do;
      if(aval ^= .) then
          paval = aval;
  end;

run;
```

R

```
library(zoo)


dfADLB2 <- dfADLB |>
  filter(PARAMCD == 'ALT') |>
  arrange(USUBJID, ADY) |>
  group_by(USUBJID) |>
  mutate(AVAL = na.locf(AVAL, na.rm = FALSE))
```

zooパッケージのna.locf関数を使用

usubjid	paramcd	avisit	ady	aval	chg
SUBJ010	ALT	Baseline	1	64.677966204	0
SUBJ010	ALT	Baseline	1	94.977956547	30.299990343
SUBJ010	ALT	Week 1	8	81.538702025	16.860735822
SUBJ010	ALT	Week 4	29	.	.
SUBJ021	ALT	Baseline	1	82.542627882	0
SUBJ021	ALT	Week 2	1	.	.
SUBJ021	ALT	Week 4	15	89.751358272	7.2087303902
SUBJ021	ALT	Week 4	15	92.423925132	9.8812972496

LOCFする

ADLBのALT（PARACD = ‘ALT’）の測定結果（AVAL, CHG）が欠損値の場合、直前のレコードの測定結果で補完する。

R

```
dfADLB2 <- dfADLB |>
  filter(PARAMCD == 'ALT') |>
  arrange(USUBJID, ADY) |>
  group_by(USUBJID) |>
  mutate_at(vars(AVAL, CHG), ~ na.locf(., na.rm = FALSE))
```

```
mutate_at(vars(AVAL, CHG), ~ na.locf(., na.rm = FALSE))
```

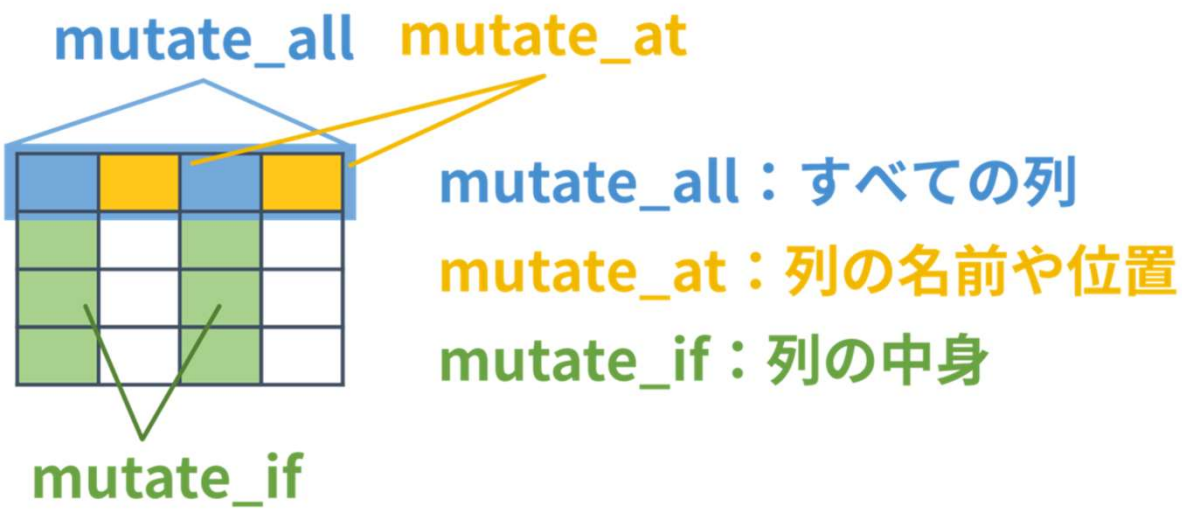
関数の宣言（対象変数に実行する関数を以降に定義する）

vars関数で定義した変数が順次渡されるイメージ

mutate関数を理解する

製薬協

各関数のイメージ



https://www.bioinfoblog.com/entry/tidydata/advancedmutate

mutate_all: 全ての変数を対象に対して同様の処理する
mutate_at: 指定した変数名に対して同様の処理をする
mutate_if: 変数の値の条件に合致する※変数に対して同様の処理をする
※is.numeric, is.factorとか

利用データ

製薬協

- **Harvard Dataverse – Dummy ADaM datasets (CC0)**
  https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/L7RURL