PharmaSUG China 2022 - Paper CC-104

Rotate (Transpose) a Dataset as Appropriate in SAS

Muyan Zhou, Xiaomin Ding, 3SBIO INC.

ABSTRACT

Rotating a dataset is extensively used in our programmers' daily work, and with that comes lots of choices in SAS, such as PROC TRANSPOSE, ARRAY, PROC SQL, etc. The problem is how to achieve it simply and effectively with an appropriate method mentioned above. This paper will provide you with an approach to make this decision, based on the types of raw dataset and final dataset post analysis.

INTRODUCTION

One of SAS programmers' works is to prepare datasets more appropriate for analysis and normalize them for further data submission. A standard dataset can greatly improve your work efficiency and simplify SAS procedures. Rotating a dataset occurs frequently and can sometimes be an indispensable step in this standardization process. You can try to use PROC TRANSPOSE, ARRAY, PROC SQL, or combine them together to make a transpose and create a good-looking table.

All the methods provided may achieve your desired result. Nevertheless, if fewer steps and concise codes are preferred to accomplish the task, the method you choose can depend on your structure of raw data and final outcomes, which could be "long" datasets, "wide" datasets, or a combination of the two.

LONG DATASET & WIDE DATASET

There are two common formats of datasets in SAS programming: long dataset and wide dataset (shown below).

Long Data

Obs	Name	variable	value
1	Alfred	Sex	M
2	Alfred	Age	14
3	Alfred	Height	69
4	Alfred	Weight	112.5
5	Alice	Sex	F
6	Alice	Age	13
7	Alice	Height	56.5
8	Alice	Weight	84
9	Barbara	Sex	F
10	Barbara	Age	13
11	Barbara	Height	65.3
12	Barbara	Weight	98
13	Carol	Sex	F
14	Carol	Age	14
15	Carol	Height	62.8
16	Carol	Weight	102.5

Wide Data

Obs	Name	Sex	Age	Height	Weight
1	Alfred	M	14	69.0	112.5
2	Alice	F	13	56.5	84.0
3	Barbara	F	13	65.3	98.0
4	Carol	F	14	62.8	102.5

Display 1. Long dataset & wide dataset

One obvious way to distinguish the two kinds of formats is from their structures: a long format is distributed vertically and looks long, a wide format is distributed horizontally and looks wide. Also, a long format collects variables in one column and contains values that do repeat in the first column, a wide format arranges the variables in a row and contains values that do not repeat in the first column.

In clinical data analysis work, using which structure to do data entry depends on whether it is convenient for your subsequent operations: data collection and data analysis.

In the data collection phase, long data format is more suitable for data entry, because you only need to add data at the end of the table. In contrast, you need to locate related rows and columns when adding new values in a wide format. Additionally, if there is a new test variable to add, the long data structure only needs to be entered under the 'variable' column, while the wide data structure needs to add a new column and involves more other operations.

In the data analysis phase, compared to long data format, each variable is displayed separately in wide data format. Therefore, wide data format is more supportive to do calculations of each variable and analyze the correlation between two variables. Other than that, most datasets in real world are recorded in wide data formats because the structure of information is more clear and easier for your brains to interpret.

You will encounter to do kinds of collection and analysis with long data format and wide data format in clinical trial, and CDISC foundational standards, whose organization is in the form of a combination of both long and wide format, are especially required to follow in most cases. Therefore, it is vital to transpose long and wide data to and from each other, or into the prescribed format. PROC TRANSPOSE, ARRAY, PROC SQL are three choices provided in SAS to support these rotations according to specific situations.

LONG TO WIDE

The following example is part of eye test records in the real world. It is a long dataset format that contains SUBJID, EYE (whether the tested eye is targeted), VISIT, and BCVA (test score). The requirement of Medical Department is to display the subjects' BCVA scores in each visit, which can provide clear information to show the difference and trajectory of each subject's eye situations. To achieve this requirement, you should transpose this long data format into a wide one.

The raw dataset is provided below and in a long format:

```
DATA bcva;
  infile cards missover;
  input SUBJID $ EYE $ VISIT $1. BCVA;
  cards;
  S01001 TARGET 1 82
```

```
S01001 NTARGT 2 82
S01001 NTARGT 3 77
S01001 NTARGT 4 80
S01001 NTARGT 5 81
S01002 TARGET 1 56
S01002 NTARGT 2 55
S01002 NTARGT 3 61
S01002 NTARGT 4 68
S01003 TARGET 1 66
S01003 NTARGT 2 69
S01003 NTARGT 3 68
S01003 NTARGT 4 89
S01003 NTARGT 5 89
S01003 NTARGT 6 89
;
run;
```

The expected rotated result is shown in the following figure:

Obs	SUBJID	EYE	VISIT	BCVA
1	S01001	NTARGT	2	82
2	S01001	NTARGT	3	77
3	S01001	NTARGT	4	80
4	S01001	NTARGT	5	81
5	S01001	TARGET	1	82
6	S01002	NTARGT	2	55
7	S01002	NTARGT	3	61
8	S01002	NTARGT	4	68
9	S01002	TARGET	1	56
10	S01003	NTARGT	2	69
11	S01003	NTARGT	3	68
12	S01003	NTARGT	4	89
13	S01003	NTARGT	5	89
14	S01003	NTARGT	6	89
15	S01003	TARGET	1	66

Obs	SUBJID	EYE	VISIT1	VISIT2	VISIT3	VISIT4	VISIT5	VISIT6
1	S01001	NTARGT		82	77	80	81	
2	S01001	TARGET	82					
3	S01002	NTARGT		55	61	68		
4	S01002	TARGET	56					
5	S01003	NTARGT		69	68	89	89	89
6	S01003	TARGET	66					

Display 2. Rotate long data to wide dataset

You can try to use the 3 methods in SAS to accomplish this result and compare their strengths and weaknesses.

1. PROC TRANSPOSE

```
proc sort data=bcva;
    by SUBJID EYE VISIT;
run;
proc transpose data=bcva out=bcva_1 prefix=VISIT;
    id VISIT;
    by SUBJID EYE;
```

```
    var BCVA;
run;

data bcva_1_1;
    retain SUBJID EYE VISIT1-VISIT6;
    set bcva_1;
    drop _name_;
run;
```

2. ARRAY

```
data bcva_2;
  array V[6] VISIT1-VISIT6;
  do until (last.EYE);
    set bcva;
    by SUBJID EYE;
    V[VISIT] = BCVA;
  end;
  keep SUBJID EYE VISIT1-VISIT6;
run;
```

3. PROC SQL

```
proc sql;
  create table bcva_3 as
  select SUBJID,EYE
    ,max(case VISIT when "1" then BCVA else '' end ) as VISIT1
    ,max(case VISIT when "2" then BCVA else '' end ) as VISIT2
    ,max(case VISIT when "3" then BCVA else '' end ) as VISIT3
    ,max(case VISIT when "4" then BCVA else '' end ) as VISIT4
    ,max(case VISIT when "5" then BCVA else '' end ) as VISIT5
    ,max(case VISIT when "6" then BCVA else '' end ) as VISIT6
  from bcva
  group by SUBJID,EYE;
quit;
```

From the 3 methos above, you can find that PROC TRANSPOSE may be better to rotate long to wide. It is easier to complete majority of operations with various options in one procedure and processes logical operating ideas. ARRAY in data steps may be more complicated than PROC TRANSPOSE which just specifies options in procedure, also sets higher demanding on understanding the structure of datasets; CASE WHEN in PROC SQL is another method you can attempt. It is flexible and convenient to rename variables, not required to sort the dataset at first either. However, lines of handwriting codes and pre-logical thinking may be a waste of time, macros may be applied to solve this problem.

WIDE TO LONG

The following example is part of exposure records in the real world. To be emphasized, it is transposing 3 groups of variables: each subject involves 3 times of treatments which display variables of exposure dates, exposure times and whether exposure occurs in a format of wide dataset. The requirement is to select the Start Date/Time of each subject. Therefore, the first operation to solve this problem is rotating this wide format raw data to a long format, which can create 3 new variables to include all their 3 dates, 3 times, and 3 YN variables separately and facilitate subsequent sorting and selection of the Start Date/Time.

The raw dataset is provided below and in a wide format:

```
data EX;
  infile cards missover;
  input SITEID SUBJID EXYN1 $ EXDAT1 $10. EXTIM1 $ EXYN2 $ EXDAT2 $10. EXTIM2
  $ EXYN3 $1. +1 EXDAT3 $10. EXTIM3 $;
  cards;
  01 0101 Y 2019-10-21 12:05 Y 2019-10-25 12:07
  02 0201 N
  03 0301 Y 2020-01-13 11:20 Y 2020-01-15 11:11 Y 2020-01-17 11:28
  ;
run;
```

The expected rotated result is shown in the following figure:

Obs	SITEID	SUBJID	EXYN1	EXDAT1	EXTIM1	EXYN2	EXDAT2	EXTIM2	EXYN3	EXDAT3	EXTIM3
1	1	101	Y	2019-10-21	12:05	Y	2019-10-25	12:07			
2	2	201	N								
3	3	301	Y	2020-01-13	11:20	Y	2020-01-15	11:11	Y	2020-01-17	11:28

Obs	SITEID	SUBJID	seq	DAT	TIM	EXYN
1	1	101	1	2019-10-21	12:05	Y
2	1	101	2	2019-10-25	12:07	Y
3	1	101	3			
4	2	201	1			N
5	2	201	2			
6	2	201	3			
7	3	301	1	2020-01-13	11:20	Y
8	3	301	2	2020-01-15	11:11	Y
9	3	301	3	2020-01-17	11:28	Y

Display 3. Rotate wide data to long dataset

You can try to use the 3 methods in SAS to accomplish this result and compare their strengths and weaknesses:

1. PROC TRANSPOSE

```
proc sort data=EX;
    by SITEID SUBJID;
quit;
proc transpose data=EX out=EX1_1;
    by SITEID SUBJID;
    var EXYN: EXDAT: EXTIM:;
quit;
data EX1_2;
    set EX1_1;
    seq=compress(_NAME_,'1234567890','k');
```

```
    NAME=compress(_NAME_,'1234567890');
run;
proc sort data=EX1_2;
    by SITEID SUBJID seq;
quit;
proc transpose data=EX1_2 out=EX1(drop=_name_);
    by SITEID SUBJID seq;
    var COL1;
    id NAME;
quit;
```

2. ARRAY

```
data EX2;
    set EX;
    array S[3] EXDAT1-EXDAT3;
    array T[3] EXTIM1-EXTIM3;
    array Y[3] EXYN1-EXYN3;
    do seq = 1 TO 3;
    DAT=S[seq];
    TIM=T[seq];
    EXYN=Y[seq];
    output;
    end;
    keep SITEID SUBJID seq DAT TIM EXYN ;
run;
```

3. PROC SQL

```
proc sql;
  create table EX3_1 as
  select SITEID,SUBJID,EXYN1 as EXYN,EXDAT1 as DAT,EXTIM1 as TIM,1 as seq
  from EX;
quit;

proc sql;
  create table EX3_2 as
  select SITEID,SUBJID,EXYN2 as EXYN,EXDAT2 as DAT,EXTIM2 as TIM,2 as seq
  from EX;
quit;

proc sql;
  create table EX3_3 as
  select SITEID,SUBJID,EXYN3 as EXYN,EXDAT3 as DAT,EXTIM3 as TIM,3 as seq
  from EX;
quit;
```

```
data EX3;
  set EX3_1 EX3_2 EX3_3;
  proc sort;by SITEID SUBJID seq;
run;
```

To conclude, ARRAY is presented more efficient to rotate wide data format to long data format, especially referring to transpose more than one groups of variables. Whereas, PROC TRANSPOSE doesn't have a mechanism to transpose by observations when there is no observations unique ID. PROC SQL behaves similarly with PROC TRANSPOSE in this rotating process, it is easier to be understood logically, but requires repeated operations and can be a tedious work when there are more variables need to be kept. However, macros can be considered in this method to relief your work.

CONCLUSION

SAS offers multiple options to do rotations of wide and long formats mutually. This paper only presents three methods and demonstrates their advantages and disadvantages in the angle of dataset formats (long or wide).

PROC TRANSPOSE is the most intuitive procedure to do transposition in SAS, especially on long to wide, and it is really a shortcut when only a single variable is transposed, no matter the dataset is a in a long format or wide. Moreover, you do not need to worry how many new variables will be created in a new dataset, because PROC TRANSPOSE can accommodate as enough as you want. On the contrary, rotating more than one groups of variables into new observations is hard to rely on one step of PROC TRANSPOSE and you may need more data steps to resort new variables in order.

ARRAY can implement the rotation of more than one groups of variables, especially wide to long, in only one data step. It can dispose of all works in PROC TRANSPOSE but with a few managements of datasets. Otherwise, the placement of new variables can be rearranged in ARRAY statements. On opposite, it may be more complicated to be understood by SAS programmers and the types of variables (character or numeric) should be considered at start.

PROC SQL provides flexible options to do transpose in SAS. You do not need to pre-sort the datasets at first, do not worry about the difference of variables' names either. CASE WHEN can be a help in transposing long to wide. However, lots of codes writing or macros are required while there are multiple variables transposed and it looks not appropriate to do wide to long.

REFERENCES

[1] Karen Grace-Martin. The Wide and Long Data Format for Repeated Measures Data. Available at The Wide and Long Data Format for Repeated Measures Data - The Analysis Factor

[2] UCLA Advanced Research Computing Statistical Methods and Data Analytics. HOW TO RESHAPE DATA WIDE TO LONG USING PROC TRANSPOSE | SAS LEARNING MODULES. Available at How to reshape data wide to long using proc transpose | SAS Learning Modules

(ucla.edu)

[3] Mike Zdeb. University at Albany School of Public Health, Rensselaer, NY. Long-to-Wide: PROC TRANSPOSE vs Arrays vs PROC SUMMARY. Available at Long-to-Wide: PROC TRANSPOSE vs Arrays vs PROC SUMMARY (lexjansen.com)

ACKNOWLEDGMENTS

We are sincerely acknowledged all the people who reviewed this paper and provided constructive comments. Especially thanks for the recommendation and support of our leader and team.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at:

Muyan Zhou
3SBIO INC.
zhoumuyan@3s-guojian.com
Xiaomin Ding
3SBIO INC.
dingxiaomin@3s-guojian.com