Paper PO03

Advanced Array Applications in Clinical Data Manipulation

Zaizai Lu, AstraZeneca Pharmaceutical
David Shen, ClinForce Inc.

ABSTRACT

Dealing with a single variable is all well and good, but there comes a time when you have to deal with a big list of variables. That's where array comes in. Array in SAS® allows you to group a bunch of variables for the same process. The huge block of the repetitious statements and redundant calculation codes can be reduced to just a few lines. With arrays you can simplify the coding in many cases. Sometimes, arrays can even accomplish tasks which are not easily done with other methods. Array, besides its iterative and conditional processing, can be very powerful. This paper introduces some advanced array applications in data manipulations from data search, count consecutive days, LOCF, find and replace, shift, leading to a more complicated efficient process. These array applications will be presented with ten practical examples. The purpose is to explore new applications of arrays over alternative strategies. The paper is appropriate for intermediate SAS programmers with array experience.

Keywords Array Advanced Application Data Manipulation

INTRODUCTION

As we know, SAS arrays make it easier to do a great deal of iterative jobs with simple statements. Whenever there are a group of variables to be processed in the data step, it might be well worth considering using arrays. Two steps in the use of array are commonly involved:
1. ARRAY definition
2. DO loop under optional IF-THEN-ELSE conditions

For example, if the temperature in ^{o}F at 5 different locations needs to be converted to unit of ^{o}C, the following array codes may be used:

```
ARRAY TEMPRS [5] TEMPR1 TEMPR2 TEMPR3 TEMPR4 TEMPR5;
   DO I =1 TO 5;
       IF TEMPRS ^=. THEN TEMPRS[I]= (TEMPRS [I] –32 )*5/9 ;
   END;
```

The array defined above is a static array, the simplest type of arrays. It has the predefined size which will not change any more. Dynamic array, in stead, has no fixed size. It can grow or shrink with the different data automatically. Dynamic arrays differ from their static cousins in such a way that you never specify the number of items in them, * is used in braces to represent the array size. Since there is no number inside braces, SAS knows that it is a dynamic array, and its size is likely to be changed. The function DIM(array name) returns the number of elements in the array. The following array definitions do the same thing.

```
ARRAY TEMPRS [*]  TEMPR1 TEMPR2 TEMPR3 TEMPR4 TEMPR5 ;
ARRAY TEMPRS [*]  TEMPR1 - TEMPR5 ;
ARRAY TEMPRS [*]  TEMPR:  ;
```

Next, we will present some more advanced array applications with either static or dynamic arrays.

APPLICATIONS

1. Search Specified Value

Target

A	B	C	D	E

To search and find a specified target value, such as maximum, trough concentrations in PK, or desired endpoint values or scores in PD, from a group of measurements, the use of an array has been demonstrated very efficient. The algorithm is to compare the target value against an array and perform an action if the target value is found in the array. The example here is to find on which day the maximum efficacy is reached.

```
data pd2;
  set pd1;
  array days[4] day1-day4;
  maxscore=max (of days [*]);
  do i=1 to dim(days);
    if maxscore >0 and days[i]=maxscore then do;
        tmax=i;
        return;
    end;
  end;
  drop i maxscore;
run;
```

Obs	REGIMEN	SUBJECT	DAY1	DAY2	DAY3	DAY4	TMAX
1	A	101	0.0	0.0	0.0	0.0	.
2	A	102	.	0.5	0.5	0.0	2
3	B	106	0.5	0.0	0.0	0.0	1
4	B	107	0.5	2.0	0.5	2.0	2
5	C	111	1.0	3.0	2.0	2.5	2
6	C	112	2.0	3.0	2.5	3.5	4

2. Count Consecutive Days

In some clinical studies, patient diaries are used to collect important data, such as symptom scores, concomitant or rescue medication usages and data about quality of life. In asthma study, some analyses are based on diary data such as awakening-free nights, symptom-free days, and asthma control days. For example, we check and see whether a subject has not experienced night awakening due to asthma for 5 consecutive days, a certain rescue medication has not been used more than 4 consecutive days, or no morning and evening symptom scores, no awakening due to asthma more than 3 consecutive days. From the DIARY data set and program below, we can easily list the subjects and their consecutive days along with start date and stop date by using array.

Obs	SUBJECT	DATE	FREEFLG	DATECNT
1	1	25MAR2004	1	1
2	1	26MAR2004	1	2
3	1	27MAR2004	1	3
4	1	28MAR2004	1	4
5	1	29MAR2004	1	5
6	2	26MAR2004	1	1
7	2	27MAR2004	1	2
8	2	29MAR2004	1	3
9	3	26MAR2004	1	1
10	3	27MAR2004	1	2
11	3	28MAR2004	1	3
12	3	02APR2004	1	4
13	4	02APR2004	1	1
14	5	25MAR2004	1	1
15	5	26MAR2004	1	2
16	5	27MAR2004	1	3
17	5	28MAR2004	1	4
18	5	31MAR2004	1	5

```
/*-------------------------------------------------------------------------------------------------------------------------------*/

proc transpose data=diary prefix=_dat out=temp1;
  by subject;
  var date;
```

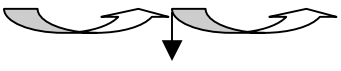
```
run;

data temp2 (keep=subject flag count i rename=(i=datecnt));
  set temp1 ;
  array  dates {*} _dat: dummy ;
  retain flag count 1;
  do i=1 to dim(dates)-1;
     if dates[i]^=. then do;
        if dates[i] = dates[i+1]-1 then do;
            output;
           count=count + 1;
        end;
        else do;
           output;
           flag =flag + 1;
            count=1;
          end;
     end;
  end;
run;

data temp3;
  merge  temp2
         diary;
  by subject datecnt;
run;

data continue (where=(count >=3 ));  /*3 consecutive days defined*/
  set temp3;
  by subject flag;
  retain f_date ;
  if first.flag then f_date=date ;
  if last.flag then do;
     l_date=date ;
     output;
  end;
  keep   subject f_date l_date count;
  format f_date l_date date9. ;
run;;

/*-----------------------------------------------------------------------------------------------------------------------*/
```

Obs	SUBJECT	COUNT	F_DATE	L_DATE
1	1	5	25MAR2004	29MAR2004
2	3	3	26MAR2004	28MAR2004
3	5	4	25MAR2004	28MAR2004

3. Data LOCF

"LOCF" stands for "Last Observation Carried Forward", it means last non-missing value carried forward.

TIME1	TIME2	TIME3	TIME4	TIME5
A	B	.	.	E



TIME1	TIME2	TIME3	TIME4	TIME5
A	B	B	B	E

In LOCF analyses, when a patient drops out of a trial due to any reasons, the results of the last evaluation visit are carried forward as if the patient had continued to the completion of the trial without further change. Since patients who discontinue medication are regarded as treatment failures, LOCF analyses are widely considered to provide a more conservative test of drug effects.

The following data set called SCORE will be used as the example.

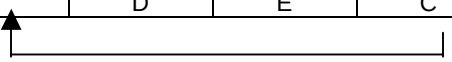
Obs	SUBJECT	REGIMEN	TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
1	1	D	0.5	0.5	0.0	0.0	0.5	0.5
2	2	A	0.0	0.5	.	.	1.5	0.0
3	3	B	0.0	0.0	1.0	.	0.0	0.0
4	4	C	0.0	0.0	0.0	.	0.0	0.0
5	5	D	0.0	0.5	1.5	.	0.5	0.5
6	6	A	0.0	1.0	1.5	0.5	.	1.0

```
data locf ;
  set score ;
  array time [*] time: ;
    do i=1 to dim(time);
         if time[i]=. then time[i]=time[i-1];
       end;
  drop i makeup;
run;
```

Obs	SUBJECT	REGIMEN	TIME1	TIME2	TIME3	TIME4	TIME5
1	1	D	0.5	0.5	0.0	0.0	0.5
2	2	A	0.0	0.5	0.5	0.5	1.5
3	3	B	0.0	0.0	1.0	1.0	0.0
4	4	C	0.0	0.0	0.0	0.0	0.0
5	5	D	0.0	0.5	1.5	1.5	0.5
6	6	A	0.0	1.0	1.5	0.5	0.5

4. Find and Replace

TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
A	B	.	D	E	C



The array-implemented find& replace is exceptionally powerful and fast. The algorithm replaces elements referred to by iterator i in the array with new value when the condition holds, such as to find and replace the missing data. In some cases, the experiment measurements are not conducted continuously. They are discrete instead. To test the irritation of skins to patch or ointment as the example, the skin at different positions are supposed to be tested in the order of left arm, right arm, back, ... If one or more points somehow are skipped, the makeup tests would be done to get those data missed.

```
data replace;
  set score;
  array apps [5] time1- time5;
    do i=1 to dim(apps);
         if apps[i] =.  then apps[i]=makeup ;
    end;
  drop i   ;
run;
```

If there are more than one data-missing time points, a second array may be applied to hold the makeup data and pass the makeup data to the appropriate points.

```
data replace2;
  set score2;
  array apps [5]    time1- time5;
  array makeups [2] makeup1 makeup2 ;
  j=1;
    do i=1 to dim(apps);
         if apps[i] =.  then do;
            apps[i]=makeups[j] ;
             j=j + 1;
         end;
    end;
```

```
  drop i j  ;
run;
```

5. Data Shift

One subject should undergo a certain times of tests in some situations, and the time order should be kept, then a data shift process can be applied with the help of array.

TIME1	TIME2	TIME3	TIME4	TIME5	MAKEUP
A	B	.	D	E	C

TIME1	TIME2	TIME3	TIME4	TIME5
A	B	D	E	C

```
data shift;
  set score;
  array apps[*] time: makeup;
    do i = 1 to dim(apps)-1;
         if apps[i] = . then do;
            do j = i to dim(apps)-1;
                 apps[j] = apps[j+1];
            end;
            mu='-'||compress(i);
         end;
    end;
  drop i j ;
run;
```

A do loop would be useful if there are more than one missing data in the rows.

6. Data Merge

It is often required to merge dose data with other safety data, such as adverse events, vital signs, ECG, and lab results, and locate the dose-related safety profiles. For example, a patient is given several doses at certain time points. After each dose, some adverse events may occur to the patient. We need to know which adverse event is associated with which dose. Suppose there is a dose dataset and an adverse event dataset.

SUBJECT	REGIMEN	DOSEN1	DOSEN2	DOSEN3	DOSEN4
1	A	30JAN1999:08:00:00	06FEB1999:08:00:00	20FEB1999:08:00:00	27FEB1999:08:00:00
2	B	30JAN1999:08:01:00	06FEB1999:08:01:00	20FEB1999:08:01:00	06MAR1999:08:01:00
3	C	30JAN1999:08:02:00	06FEB1999:08:02:00	13FEB1999:08:02:00	27FEB1999:08:02:00

SUBJECT	AE	AEDTTM
1	NERVOUSNESS	30JAN1999:06:00:00
1	TACHYCARDIA	30JAN1999:12:15:00
1	NAUSEA	06FEB1999:16:20:00
1	DIZZINESS	20FEB1999:09:20:00
2	HEADACHE	06FEB1999:14:10:00
2	NAUSEA	06FEB1999:17:40:00
2	VASODILATATION	20FEB1999:14:30:00
2	ASTHENIA	20FEB1999:18:20:00
2	ANXIETY	27FEB1999:20:01:00
2	CHILLS	28FEB1999:06:00:00
2	CONSTIPATION	28FEB1999:22:00:00

The code to merge these 2 datasets is as following.

```
data dose_ae;
  merge  dose ae;
  by subject;
  array dosen {*} dosen:;
    do i=1 to dim(dosen);
```

```
      if dosen[i]^=. and aedttm > dosen[i] then do;
          dosedttm = dosen[i];
         dosenum  = i;
      end;
    end;
  if dosenum^=.;
  hrpostds = round(((aedttm-dosedttm)/3600), 0.1);
  format dosedttm datetime20.;
  drop i dosen1 - dosen4;
run;
```

The final result is shown below, variable DOSENUM is the order number of doses, and HRPOSTDS is time in hours after dosing.

SUBJECT	REGIMEN	AE	DOSEDTTM	AEDTTM	DOSENUM	HRPOSTDS
1	A	TACHYCARDIA	30JAN1999:08:00:00	30JAN1999:12:15:00	1	4.3
1	A	NAUSEA	06FEB1999:08:00:00	06FEB1999:16:20:00	2	8.3
1	A	DIZZINESS	20FEB1999:08:00:00	20FEB1999:09:20:00	3	1.3
2	B	HEADACHE	06FEB1999:08:01:00	06FEB1999:14:10:00	2	6.2
2	B	NAUSEA	06FEB1999:08:01:00	06FEB1999:17:40:00	2	9.7
2	B	VASODILATATION	20FEB1999:08:01:00	20FEB1999:14:30:00	3	6.5
2	B	ASTHENIA	20FEB1999:08:01:00	20FEB1999:18:20:00	3	10.3
2	B	ANXIETY	20FEB1999:08:01:00	27FEB1999:20:01:00	3	180.0
2	B	CHILLS	20FEB1999:08:01:00	28FEB1999:06:00:00	3	190.0
2	B	CONSTIPATION	20FEB1999:08:01:00	28FEB1999:22:00:00	3	206.0

7. Many-to-One Restructure

To convert dataset A to dataset B,

Obs	ID	TIME	SBPS
1	1	1	123
2	1	2	126
3	1	3	128
4	2	1	112
5	2	2	110
6	2	3	115

A -> B

Obs	ID	SBP1	SBP2	SBP3
1	1	123	126	128
2	2	112	110	115

We can submit either

```
data b;
  set a;
  by id;
  retain sbp1-sbp3;
  array  sbp[3];
   if first.id then i=1;
      sbp[i]=sbps;
      i+1;
   if last.id then output;
   drop sbps time i;
run;

or

data b;
  retain id;
  array u[*] sbp1-sbp3;
  do i=1 to dim(u);
    set a ;
    u[i]=sbps;
  end;
  drop sbps time i;
run;
```

8. One-to-Many Restructure

In case need to create dataset A from dataset B reversely, we can simply execute the code below.

```
data a;
  set b;
  array sbp[*] sbp: ;
  do i=1 to dim(sbp);
    time=i;
    sbps = sbp[i];
    output;
  end;
  keep id time sbps ;
run;
```

9. Data Concatenation

It may require sometime that any information in a certain variable or different variables be combined. For example, a new variable TCOMMENT would be created to contain all information from variables of COMMENT1, COMMENT2, . . . or a variable SEQUENCE needs to be created to hold regimen order information for statistical analysis in crossover studies.

```
proc transpose data = regimen out =temp1 prefix=_reg;
  by subject;
  var regimen;
run;

data temp2;
  set temp1;
  length sequence $10;
  array regs [*] _reg: ;
    do i=1 to dim(regs);
        sequence=compress(sequence||regs[i]);
    end;
  keep subject sequence;
run;

data sequence;
  merge regimen temp2 ;
  by subject;
run;
```

The above codes actually add one more variable SEQUENCE to the original dataset REGINMEN.

Obs	SUBJECT	PERIOD	REGIMEN	SEQUENCE
1	101	1	B	BACD
2	101	2	A	BACD
3	101	3	C	BACD
4	101	4	D	BACD
5	102	1	A	ADC
6	102	2	D	ADC
7	102	3	C	ADC

10. Retrieve Statistics from Output Dataset.

An ANOVA or ANCOVA is made to see the effect of treatments on changes in certain endpoint values and to provide pairwise comparisons of treatment effects. With ODS output, we are ready to retrieve the statistics such as p-values, and store them in a desired dataset.

```
ods output lsmeans(match_all)=lsmeans1;
ods output diff(match_all)=diffs;

proc glm data=pd ;
  class regimen center;
```

```
  model response=regimen center regimen*center;
  lsmeans regimen  / pdiff cl alpha=0.05 out=lsmeans2;
run; quit;
```

The diffs dataset contains pvaues in the following format.

Obs	Effect	Row Dependent	Name	_1	_2	_3	_4
1	REGIMEN	RESPONSE	1	_	0.8530	0.1001	0.1736
2	REGIMEN	RESPONSE	2	0.8530	_	0.1545	0.2726
3	REGIMEN	RESPONSE	3	0.1001	0.1545	_	0.5374
4	REGIMEN	RESPONSE	4	0.173	0.272	0.5374	_

However, we'd like the data in the following format:

Obs	COL	PVAL
1	1 vs 2	0.85303
2	1 vs 3	0.10011
3	1 vs 4	0.17360
4	2 vs 3	0.15449
5	2 vs 4	0.27260
6	3 vs 4	0.53742

or

Obs	NAME OF FORMER VARIABLE	1 vs 2	1 vs 3	1 vs 4	2 vs 3	2 vs 4	3 vs 4
1	PVAL	0.85303	0.10011	0.17360	0.15449	0.27260	0.53742

Arrays will help us to identify the numbers for the desired comparisons and then pull off the corresponding p-values calculated for those comparisons in the DATA step.

```
data _null_;
  set diffs end=lastrec;
  if lastrec then call symput('nobs', compress(_n_));
run;

%let nobs2=%eval(&nobs*&nobs);
%put &nobs &nobs2;

data x;
  set diffs end=last ;
  array a [&nobs2]  ;
  array pval[&nobs] _: ;
  retain a k ;
  do i = 1 to dim(pval) ;
    if rowname = i then do ;
         do j = 1 to dim(pval) ;
           k + 1 ;
          a[k] = pval[j] ;
        end ;
    end ;
  end ;
  if last then output ;
  keep a1-a&nobs2 ;
run;

data y;
  set x;
  length col $10;
  array a [&nobs, &nobs] a1-a&nobs2;
  do i=1 to &nobs;
    do j=1 to &nobs;
          if i <j then  do;
```

```
            col=compress(i)||' vs '||compress(j);
            pval=a[i,j];
            output;
          end;
    end;
  end;
  keep col pval ;
run;

proc print data=y; run;

proc transpose data=y out=z;
  id  col;
  var pval;
  idlabel col;
run;

proc print data=z label;
run;
```

In this example, a two-dimensional array is used. It has rows and columns, but to SAS, a 2D-array is no different from regular array. We just can think of two-dimensional array as the list of lists of data. The process for multidimensional array is usually done in nested loops, the first dimension (row) is the outer and the second dimension (column) is the inner. 2D array can handle with more complicated situations. For example[1] , 2D array was used to assign letters to different treatment groups automatically. Treatments that are not significantly different from each other will be assigned the same letter, while treatments that are significantly different from each other will get different letters.

REFERENCE

[1]. Dynamic Letter Assignment in Efficacy Tables in Clinical Trial Research,Proceedings of Pharmsug 2002.

CONTACT INFORMATION

Zaizai Lu
zz_lu@hotmail.com
AstraZeneca Pharmaceuticals
Wilmington, Delaware