

Paper EPO-179

The Power of Interleaving Data

Yu Feng, Merck & Co., Inc., Upper Gwynedd, PA USA

ABSTRACT

Have you ever had the experience of writing multiple SAS® data steps to accomplish a task but felt that there should be more efficient SAS code which could do the job? Interleaving a data set with itself can help you fulfill the task in the examples enumerated in this paper. The intent of this paper is to introduce the fundamental understanding of the interleaving process and discuss a few examples showing the power of interleaving data with itself. The examples included in this paper are commonly used in outcomes research. This paper will focus on interleaving a data set with itself, not on interleaving two or more different data sets.

INTRODUCTION

In healthcare outcomes research, it is quite common that we have to first calculate a group summary statistic or select the first observation in a group. Then use this calculated value to create new variables, or change the structure of the data set, and process further data manipulations. Previously I had used multiple data steps and/or SQL procedures to accomplish the task. However, I wondered if there was a more convenient method to do the job until I read the paper “Interleaving a Dataset with Itself: How and Why” by Howard Schreier (2003). This technique, interleaving a data set with itself, opens the door to solve the programming inefficiencies of using multiple data steps. Interleaving uses the pre-determined group information from the data set itself to accomplish the task in a single DATA step. This paper will focus on interleaving a data set with itself, not on interleaving two or more different data sets.

CODE COMPASRISON

Multiple steps code:

```
DATA indexdate;
    set allrec;
    by id ;
    if first.id;
    keep id date;
    rename date=indexdate;
    format date mmddyy10.;

run;

PROC sql;
    create table oneyear_select_compare as
    select b.*, a.indexdate
    from indexdate a, allrec b
    where a.id=b.id and a.indexdate<=b.date<=a.indexdate+365;

quit;
```

Code in one DATA step:

```
DATA oneyear_select;
    set allrec(in=a) allrec;
    by id;
    retain indexdate;
    format date indexdate mmddyy10.;
    if a then do;
        if first.id then indexdate=date;
        else if date<indexdate then indexdate=date;
    end;
    else do;
        if indexdate<=date<=indexdate+365;
        output;
    end;

run;
```

UNDERSTANDING THE INTERLEAVING PROCESS

The syntax of interleaving is as follows:

```
DATA z;
    set x y;
    by vars;
run;
```

Where both data sets x and y have been previously sorted by variable(s) vars. A SET statement and a BY statement are used rather than a MERGE statement, which is more frequently used in the matching process. The number of observations in the new data set, z, is the sum of the number of observations in data sets x and y.

Interleaving is a special concatenating process which concatenates the data sets x and y according to the sort order of variable(s) vars. All observations that belong to the same BY group are read sequentially from each data set until there are no more observations to be read in from any of the data sets for that BY group. Then, SAS selects the next BY group and repeats the whole process until all observations in all the data sets have been read.

We can explore the specialty of the interleaving process to get a group summary or group statistic and attach the information back to a particular BY group by interleaving data back with itself. Interleaving essentially fulfills the task called 'look forward', which is much simpler than the routine process of multiple data steps and is very useful in data manipulation for statistical calculations.

DATA SOURCE

The following data set, named ALLREC, will be used in this paper for all of the examples. The data set includes two-year observations from January 1, 2014 to December 31, 2015, with the filled prescription as represented by the variables DRUG, DATE and COST below. The data is sorted by ID, but not necessarily sorted by DATE, which is one of the advantages we will discuss in the paper.

ID	date	drug	cost
101	02/01/2014	A	10
101	10/01/2014	A	10
101	01/01/2014	B	20
101	11/01/2014	B	20
101	02/01/2015	B	20
102	10/01/2014	A	10
102	11/01/2014	A	10
103	06/01/2014	A	10
103	11/01/2015	B	20
103	12/01/2015	C	30

EXAMPLE 1

Suppose we want to define the earliest record for each subject and then use this defined earliest date to pull all records for the same subject from this earliest date up to one year (365 days) after the earliest date.

Below is the earliest record for each subject.

ID	date	drug	cost
101	01/01/2014	A	10
102	10/01/2014	A	10
103	06/01/2014	A	10

Instead of using first.ID to get the earliest record, merging the earliest record back to the data set by ID, and then selecting the records in the requested time frame, a single DATA step can accomplish the task as the code below shows.

```
DATA oneyear_select;
  set allrec(in=a) allrec;
  by id;
  retain indexdate;
  format date indexdate mmddyy10.;
  if a then do;
    if first.id then indexdate=date;
    else if date<indexdate then indexdate=date;
  end;
  else do;
    if indexdate<=date<=indexdate+365;
    output;
  end;
run;
```

Using interleaving, SAS reads each observation from the first BY group of ID, passes the observations from the first reference to data set ALLREC (in=a in the above code) and then passes the observations in the same BY group again from the second data set ALLREC. During the first pass of the BY group, the earliest date for each BY group was selected and retained as the INDEXDATE. Therefore, when the same BY group is read again (ELSE DO statement in the above code), SAS uses this INDEXDATE to select the observations in the time frame of interest. The value of INDEXDATE is retained and attached to each record for output. As long as the data is sorted by ID, no extra sorting by DATE is needed in this single DATA step even though the data is not sorted by both ID and DATE.

Moreover, data processing efficiency is improved by using the above interleaving step. For example, for a data set with 7.5 million records, the above interleaving method only used 6 seconds CPU time. Alternately, the routine process, first getting the earliest record in the first DATA step, then merging the earliest record back to the data set by ID, and then selecting the records in the requested time frame, took nearly 9 seconds CPU time. If the data is larger, greater advantages will be seen from using interleaving.

ID	date	drug	cost	indexdate
101	02/01/2014	A	10	01/01/2014
101	10/01/2014	A	10	01/01/2014
101	01/01/2014	B	20	01/01/2014
101	11/01/2014	B	20	01/01/2014
102	10/01/2014	A	10	10/01/2014
102	11/01/2014	A	10	10/01/2014
103	06/01/2014	A	10	06/01/2014

Output 1. Output of the selected one-year observations

EXAMPLE 2

In outcomes research, a popular request is to generate a drug regimen for each subject in a study window and then merge the drug regimen back to the detailed level data for each subject for future analysis. Suppose we want to create a drug regimen for each subject which includes only the unique drug each subject took. Below is the single DATA step which solves this task.

```
DATA reg;
  set allrec(in=a) allrec;
  by id;
  length regimen $20;
  retain regimen;
  format date mmddyy10.;
  if a then do;
```

```

        if first.id then regimen=drug;
        else if index(regimen,strip(drug))<1 then
regimen=strip(regimen)||'+'||strip(drug);
    end;
    else do;
        output;
    end;
run;

```

Similarly, SAS reads each observation from the first BY group of ID, passes the observations from the first reference to data set ALLREC and then passes the observations in the same BY group again from the second reference to data set ALLREC. During the first pass of the BY group, SAS combines the unique drug listed under each BY group and builds the regimen. The newly created regimen is retained and attached to each record when the second data set ALLREC is read. The regimen is reset to the drug of the first record at the beginning of each BY group when the first data set ALLREC is read.

ID	date	drug	regimen
101	02/01/2014	A	A+B
101	10/01/2014	A	A+B
101	01/01/2014	B	A+B
101	11/01/2014	B	A+B
101	02/01/2015	B	A+B
102	10/01/2014	A	A
102	11/01/2014	A	A
103	06/01/2014	A	A+B+C
103	11/01/2015	B	A+B+C
103	12/01/2015	C	A+B+C

Output 2. Output of including the unique drug regimen in the observational data

EXAMPLE 3

The example below which calculates the by group summary statistic and then merges it back to the detailed subject level data is also discussed in other papers, Toby Dunn and Chang Y. Chung (2005). This method is widely used and the SQL procedure can also solve the problem in a convenient way.

The single DATA step code:

```

DATA cost;
    set allrec(in=a) allrec;
    by id;
    retain totalcost;
    format date mmddyy10.;
    if a then do;
        if first.id then totalcost=cost;
        else totalcost+cost;
    end;
    else do;
        output;
    end;
run;

```

During the first pass of the BY group, SAS calculates the TOTALCOST for each ID by using RETAIN statement. The TOTALCOST value for each ID is retained and attached to each record when the second data set ALLREC is read. The TOTALCOST is reset to the cost of the first record at the beginning of each BY group when the first data set ALLREC is read.

ID	date	drug	cost	totalcost
101	02/01/2014	A	10	80
101	10/01/2014	A	10	80
101	01/01/2014	B	20	80
101	11/01/2014	B	20	80
101	02/01/2015	B	20	80
102	10/01/2014	A	10	20
102	11/01/2014	A	10	20
103	06/01/2014	A	10	60
103	11/01/2015	B	20	60
103	12/01/2015	C	30	60

Output 3. Output of calculating the total cost for each subject

Here is the SQL code:

```
PROC sql;
  create table cost_sql as
  select *, sum(cost) as totalcost
  from allrec
  group by ID;
quit;
```

The above SQL code looks convenient too. But for the example 3, SQL procedure is 0.3 seconds slower than the DATA step, with only 10 observations in the data set. We can infer that DATA step should be more efficient than SQL procedure here if we have bigger data.

If you like, think of Example 2 and Example 3 as similar examples, except one is for character variables and the other is for numeric variables.

CONCLUSION

When working with one detailed level data set, where both detail and summarized information are required, interleaving the data set with itself should be considered. Besides the examples listed in the paper, there are other situations to which the technique could be applied. If you prefer more convenient and efficient programming, consider interleaving a data set with itself.

REFERENCES

SAS Online Documentation, version 9.1, SAS Institute Inc.

Schreier, Howard, (2003), "Interleaving a Data Set with Itself: How and Why?" NorthEast SAS Users Group (NESUG) Conference 2003.

Dunn, Toby and Chung, Chang Y., (2005), "Retaining, Lagging, Leading, and Interleaving Data," Pharmaceutical Industry SAS Users Group 2005 (PharmaSUG2005).

ACKNOWLEDGEMENTS

Thanks to my management for valuable comments and advice on the paper.

CONTACT INFORMATION

Comments and questions are welcomed. Contact the author at:

Yu Feng

Statistical Programming, Biostatistics and Research Decision Sciences

Merck & Co., Inc.

351 North Sumneytown Pike

North Wales, PA 19454

Phone: 267-357-0883

Email: yu.feng1@merck.com

TRADEMARKS

SAS® and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® Indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.