Paper 436-2012

Your Place or Mine: Data-Driven Summary Statistic Precision

Nancy Brucken, PharmaNet/i3, Ann Arbor, MI

ABSTRACT

The number of decimal places required for displaying summary statistics on a specific parameter is generally a function of the precision of the raw values collected for that parameter and the specific summary statistic requested. This logic can easily be hard-coded for tables displaying only a limited number of parameters, but becomes more difficult to maintain for tables displaying many such parameters. This paper presents a data-driven solution to the problem of displaying summary statistics at the correct level of precision for a larger number of parameters, such as are commonly found on summaries of clinical laboratory data.

INTRODUCTION

A very common type of summary table produced for clinical trials shows various univariate summary statistics- mean, standard deviation, median, minimum and maximum- calculated for continuous variables. In this paper, we will focus on such a summary displaying quantitative clinical laboratory results at various points in time. A standard input data set containing clinical laboratory data might look something like this:

Subject ID (USUBJID)	Visit (VISITNUM)	Parameter (LBTEST)	Value (LBSTRESN)	Units (LBSTRESU)
001	1	Hemoglobin	122	g/L
001	1	WBC	3.9	X10E9/L
001	1	Calcium	2.38	mEq/L
001	1	ALT	11	U/L

Note the differences in the precision of the raw data for different parameters. Hemoglobin and ALT are collected as integers, WBC is measured to the nearest tenth, and calcium is measured to the nearest hundredth.

The level of precision required for displaying the univariate summary statistics on a table is based on both the summary statistic and the parameter being summarized. Specific rules may vary, but a common approach is to display minimum and maximum at the same level of precision as the raw data, mean and median to an additional decimal place, and standard deviation to two decimal places beyond the collected precision. This logic may be represented on a table specification as:

	Treatment A
Laboratory Test	
Mean (SD)	xx.x (xx.xx)
Median	xx.x
Min, Max	xx, xx

Summary statistics are usually calculated using PROC UNIVARIATE, PROC MEANS, PROC SUMMARY or PROC SQL. However, none of those procedures maintain the level of precision of the raw data. SAS leaves it up to the user to determine how the values should be displayed.

SOLUTION A - PROCESS EACH PARAMETER SEPARATELY

One approach to solving this problem is to construct a macro loop to process each parameter separately. While this does allow the user to specify the level of precision along with the parameter in the macro call, the main disadvantage to this approach is that summarizing each parameter requires a separate pass through the input data set. For large data sets, such as clinical laboratory data for a several hundred patient study with multiple visits per patient, processing time can become prohibitively long. In addition, any new lab tests may require manual updates to the code.

SOLUTION B - HARD-CODED DISPLAY PRECISION

Another approach is to take advantage of SAS®'s BY-group processing capabilities, and summarize the data set in a single pass by parameter and visit. IF-THEN-ELSE logic can then be used to hard-code the display precision appropriately, based on the parameter and summary statistic. This has the advantage of requiring only a single pass through the data set for summarization. However, any new lab tests will still require manual updates to the code.

SOLUTION C - DATA-DRIVEN METHOD

Yet another approach, and the one proposed by this paper, inserts an extra step to determine the maximum number of decimal places recorded for each parameter. It then uses that information to create a display format for each parameter and summary statistic, merges those formats with the summary statistics, and constructs a character variable displaying the summarized value to the appropriate level of precision. Let's take a closer look at the code required to carry out these steps.

First, determine the maximum number of decimal places recorded for each parameter. There are multiple ways to perform this task; PROC SQL gives one way to accomplish it:

```
proc sql;
  create table lbparms as
    select lbname
       , max(length(lbstresn)) as maxlen
       , max(ifn(index(lbstresn, '.')=0, 0, length(scan(lbstresn, 2, '.'))))
           as maxdec
      from lb
      group by lbname;
quit;
```

Taking a closer look at this step, MAXLEN contains the maximum length of all results for a given lab parameter, and MAXDEC contains the maximum number of decimal places recorded for that lab parameter. If the result is an integer, and so does not contain a decimal point, the IFN function sets MAXDEC to 0. Both of these quantities will be used in constructing a display format for that parameter.

The next step is to construct display formats for each summary statistic for each lab parameter. This task could easily have been accomplished in the preceding PROC SQL step. However, for clarity, this paper breaks it out separately in the following DATA step:

```
data lbfmts;
  set lbparms;
  length meanft minmaxft stdft $ 5;
  meanft = cats(put(maxlen + 2, 2.), '.', put(maxdec + 1, 2.));
  minmaxft = cats(put(maxlen, 2.), '.', put(maxdec, 2.));
  stdft = cats(put(maxlen + 3, 2.), '.', put(maxdec + 2, 2.));
run;
```

The variables MEANFT, MINMAXFT and SDTFT contain the actual format values for those statistics for each parameter. For the parameters shown in our example (sample values were: hemoglobin=122 g/L, WBC=3.9x10E9/L, calcium=2.38 mEq/L, ALT=11 U/L), the LBFTS data set looks like:

Parameter (LBTEST)	MAXLEN	MAXDEC	MEANFT	MINMAXFT	STDFT
Hemoglobin	3	0	5.1	3.0	6.2
WBC	4	1	6.2	4.1	7.3
Calcium	4	2	6.3	4.2	7.4
ALT	2	0	4.1	2.0	5.2

According to the table requirements, the mean is supposed to be displayed to one more decimal place than the raw data. However, if the raw value was an integer, not only is another character required for displaying the tenths digit, but a space is needed for the decimal point, as well.

The final step is to add the formats to the summarized data, and use the PUTN function to associate the correct format with each summary statistic.

```
data final (keep=lbname visitnum meanc sdc medianc minc maxc);
  merge labsum (in=inl) lbfmts (in=inlf);
  by lbname;
  if inl and inlf;

  array stats(*) mean sd median min max;
  array fmts(*) meanft stdft meanft minmaxft minmaxft;
  array statsc(*) $10 meanc sdc medianc minc maxc;

  do i=1 to dim(statsc);
    statsc(i) = putn(stats(i), fmts(i));
  end;
run;
```

Note that mean and median are displayed using the same format, as are the minimum and maximum values.

CONCLUSION

It has been said that the best programmers are lazy, because they let the computer do all of the work. The use of data-driven programming techniques is one way to accomplish that objective. In the example shown in this paper, the raw lab values encountered in the data are used to determine the display formats of their corresponding summary statistics. As a result, any new continuous lab parameters encountered in the data after the program is written are automatically formatted correctly, without requiring code modifications by the programmer.

REFERENCES

Hunt, Stephen and Fairfield-Carter, Brian. 2008. “Dynamic Decimal Precision and Alignment in Clinical Trial Laboratory Summary Tables and Patient Data Listings.” *SAS Global Forum 2008 Conference Proceedings*.

ACKNOWLEDGMENTS

Thanks to my colleagues at PharmaNet/i3, who reviewed this paper and provided helpful suggestions.

CONTACT INFORMATION

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

Nancy Brucken
PharmaNet/i3
5430 Data Court, Suite 200
Ann Arbor, MI 48108

Work Phone: (734) 757-9045
E- mail: nbrucken@pharmanet-i3.com