“Which Subjects?”
Combining a Summary Table and a Data Listing to Answer Clinical’s Favorite Question
Kim Truett, KCT Data, Inc., Alpharetta, GA

ABSTRACT

You’ve programmed, reviewed, QC’ed and double checked the monthly summary table for a clinical trial. You know the N’s are right, and the percents are calculated correctly. You deliver the table, and the question you get back is all too familiar “How does this compare to the last time you ran this for me?” and “Which subjects are these?” pointing to one specific number on the table.

This paper presents a simple way (that is, no macros or complex loops) to create a combined table / listing that presents both current and incremental data, and includes both the statistics and the study subjects that comprise that statistic, or that includes summary statistics on the bottom of a listing.

Or you program a listing, and the question becomes “Can you tell me how many of these subjects . . .?”

Introduction

Behind every statistic programmed is a group of study subjects. Yet, summarizing the data obscures which subjects are included. Also, each time a program is rerun, the comparison is immediately made to the previous run of the program.

For our example, a basic frequency table is shown here:

<table>
<thead>
<tr>
<th>Subject Status</th>
<th>N (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Subjects</td>
<td>312</td>
</tr>
<tr>
<td>Rating &gt; 3 w/o Meds</td>
<td>10(3%)</td>
</tr>
<tr>
<td>Rating &gt; 3 with Meds</td>
<td>5(2%)</td>
</tr>
<tr>
<td>Rating &lt; 3 w/o Meds</td>
<td>297(95%)</td>
</tr>
<tr>
<td>Had Surgery</td>
<td>3(1%)</td>
</tr>
<tr>
<td>Had SAE</td>
<td>0(0%)</td>
</tr>
</tbody>
</table>

Then the question is asked “Which 10 subjects had the “Ratings > 3 w/o meds” and “Which 3 subjects had surgery?”

Not wanting to determine this each month, we can add this data to the table, resulting in:

<table>
<thead>
<tr>
<th>Subject Status</th>
<th>N (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Subjects</td>
<td>312</td>
</tr>
<tr>
<td>Rating &gt; 3 w/o Meds</td>
<td>10(3%)</td>
</tr>
<tr>
<td></td>
<td>1281</td>
</tr>
<tr>
<td></td>
<td>1283</td>
</tr>
<tr>
<td></td>
<td>1286</td>
</tr>
<tr>
<td></td>
<td>1383</td>
</tr>
</tbody>
</table>
The next question is, “How does this compare to last month’s data?” This table now adds that data in as well:

### Table 3
Cumulative Subject Status
(All Treated Subjects)

<table>
<thead>
<tr>
<th>Category</th>
<th>SEP07</th>
<th>AUG07</th>
<th>JUL07</th>
<th>JUN07</th>
<th>MAY07</th>
<th>APR07</th>
<th>MAR07</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Subjects</td>
<td>312</td>
<td>281</td>
<td>219</td>
<td>150</td>
<td>116</td>
<td>50</td>
<td>3</td>
</tr>
<tr>
<td>Rating &gt; 3 w/o Meds</td>
<td>10(3%)</td>
<td>6(3%)</td>
<td>6(2%)</td>
<td>5(3%)</td>
<td>4(3%)</td>
<td>3(6%)</td>
<td>1(33%)</td>
</tr>
<tr>
<td>Rating &gt; 3 with Meds</td>
<td>1281</td>
<td>1281</td>
<td>1281</td>
<td>1281</td>
<td>1281</td>
<td>1281</td>
<td>1281</td>
</tr>
<tr>
<td>Rating &lt; 3 w/o Meds</td>
<td>297(95%)</td>
<td>272(97%)</td>
<td>210(96%)</td>
<td>144(96%)</td>
<td>111(96%)</td>
<td>46(92%)</td>
<td>2(66%)</td>
</tr>
<tr>
<td>Had Surgery</td>
<td>3(1%)</td>
<td>2(1%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
</tr>
<tr>
<td>Had SAE</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
<td>0(0%)</td>
</tr>
</tbody>
</table>

Now this table presents the frequencies, listings of the subjects in each frequency group, and the data for prior months.
Code to Create Combined Table

The following code creates the dataset for this table. The dataset can then be printed in numerous ways which will not be discussed here.

Section 1 – Create the summary frequencies with the subject numbers added in (Table 2 above).

Step 1: Create a dataset containing the information that needs to be summarized. This step would include all data manipulation needed to produce a dataset that the summary frequency will be run on. Col1 is a derived variable to identify the Column 1 category.

All code presented here is for example only.

DATA SUMMARY_DATA;
SET TAB_PR (IN=INPR)
   TAB_SA (IN=INSA)
   TAB_RATING (IN=INRATING);
IF INPR THEN COL1='Had Surgery';
IF INSA THEN COL1='Had SAE';
IF INRATING AND MEDS=0 AND RATING > 3 THEN COL1 = 'Rating > 3 w/o Meds';
IF INRATING AND MEDS=1 AND RATING > 3 THEN COL1 = 'Rating > 3 with Meds';
IF INRATING AND MEDS=0 AND RATING < 3 THEN COL1 = 'Rating < 3 w/o Meds';
COUNTER = 1;
RUN;

Step 2: The next step is to create a dataset with all possible row values. This will set up a dataset that has all possible rows that can then be updated with data to create the table.

DATA ROWSET;
FORMAT COUNT PERCENT 8.0 COL1 $50.;
DO X = 1 TO 5;
   IF X=1 THEN COL1 = 'Rating > 3 w/o Meds';
   IF X=2 THEN COL1 = 'Rating > 3 with Meds';
   IF X=3 THEN COL1 = 'Rating < 3 w/o Meds';
   IF X=4 THEN COL1 = 'Had Surgery';
   IF X=5 THEN COL1 = 'Had SAE';
   COUNT = 0;
   PERCENT = 0;
   OUTPUT;
END;
RUN;

Step 3: Calculate the frequencies using the dataset from step 1.
PROC FREQ DATA=SUMMARY_DATA; TABLES COL1 * COUNTER / OUT=STATS MISSPRINT NOPRINT LIST;
PROC SORT DATA=STATS; BY COL1;
RUN;

Step 4: Use the data from the output of the frequency to update the ‘all rows’ dataset. This will update all possible rows with the frequencies that actually occurred by inserting the values into the correct column based on COL1 values.

DATA ALLSTATS;
UPDATE ROWSET
   STATS;
   BY COL1;
RUN;
This creates a table with all of the frequencies. The next step is to give the categories a sort order. The summary frequencies will be given COL1S ("column 1 sort") values that are whole numbers and the individual subject numbers will be given fractional numbers so that the frequencies always appear above the list of subject numbers for that frequency.

```
DATA STAT_SORT;
  SET ALLSTATS;
  IF COL1 = 'No. of Subjects' THEN COL1S=0;
  IF COL1 = 'Rating > 3 w/o Meds' THEN COL1S=1;
  IF COL1 = 'Rating > 3 with Meds' THEN COL1S=2;
  IF COL1 = 'Rating < 3 w/o Meds' THEN COL1S=3;
  IF COL1 = 'Had Surgery' THEN COL1S=4;
  IF COL1 = 'Had SAE' THEN COL1S=5;
PROC SORT; BY COL1S;
RUN;
```

Step 5: Using the original dataset created in Step 1, create the listing of subject numbers setting up COL1 so that the data can be easily aggregated to the summary frequencies and assign the sort variables (COL1S).

```
DATA SUBJECT_NUM (KEEP= USUBJID COL1 COL1S COL3);
  SET SUMMARY_DATA;
  IF COL1 = 'Rating > 3 w/o Meds' THEN COL1S=1.1;
  IF COL1 = 'Rating > 3 with Meds' THEN COL1S=2.1;
  IF COL1 = 'Rating < 3 w/o Meds' THEN COL1S=3.1; /* While this might seem like it makes sense, it is the group that 297 of the 312 subjects fall into, and therefore probably makes sense to omit this list of subject numbers; */
  IF COL1 = 'Had Surgery' THEN COL1S=4.1;
  IF COL1 = 'Had SAE' THEN COL1S=5.1;
```

Step 6: The frequencies and the subject numbers are then set together.

```
DATA STATS_SBJ (KEEP=COL1 COL3 COL1S RENAME=(COL3=EVER COL1=CATE1));
  SET STAT_SORT SUBJECT_NUM;
PROC SORT; BY COL1S CATE1;
RUN;
```

Section 2 – Add the cumulative previous month’s data (Table 3 above)

Step 1: Now the dataset has summary frequencies with all of the subject numbers underneath. The next step is to add the cumulative data to the current data. The cumulative dataset is updated each month with the current month’s data. In this example, the STATS_SBJ above is saved each month into an archived database (named t3_cumul):

```
DATA CUMULAT.T3_CUMUL;
  SET CUMULAT.T3_CUMUL
    STATS_SBJ (IN=NEW);
  IF NEW THEN VERSION = 0;
  ELSE VERSION=VERSION-1;
RUN;
```

Within the cumulative datasets, there is a counter named version which is set so that version = 0 for the current version; -1 is the previous month’s data, -2 is two months previous, etc. So, when the data is added each month, all previous months are incremented by -1. Initially, the database would have just one month’s data, but each month, data is added. In the end, there are many, many months of data in the cumulative datasets.

Step 2: Now the cumulative data is read in, using the VERSION number to set-up the columns. The actual datapoints (frequency count or subject number) are stored in a variable named ‘EVER’. As the data is read in, it is assigned to its correct column, based on the version number, by setting the COLx variable equal to EVER. In this way, each column of data gets populated with data from the correct previous month. CATE1
ensures that the category is assigned. The data is then output into one dataset per column and sorted so that all columns can be merged with the current month’s data into a single dataset in the next data step.

DATA COL3 (KEEP=COL1S CATE1 COL3 EVER)
  COL4 (KEEP=COL1S CATE1 COL4 EVER)
  COL5 (KEEP=COL1S CATE1 COL5 EVER)
  COL6 (KEEP=COL1S CATE1 COL6 EVER);
FORMAT VERSION 8. COL3 COL4 COL5 COL6 $8.;
SET CUMULAT.T3_CUMUL;
  IF VERSION = 0 THEN DO; COL3 = EVER; OUTPUT COL3; END;
  IF VERSION = -1 THEN DO; COL4 = EVER; OUTPUT COL4; END;
  IF VERSION = -2 THEN DO; COL5 = EVER; OUTPUT COL5; END;
  IF VERSION = -3 THEN DO; COL6 = EVER; OUTPUT COL6; END;
PROC SORT DATA=COL3; BY COL1S CATE1 EVER;
PROC SORT DATA=COL4; BY COL1S CATE1 EVER;
PROC SORT DATA=COL5; BY COL1S CATE1 EVER;
PROC SORT DATA=COL6; BY COL1S CATE1 EVER;
RUN;

Step 3: The final step is to combine the current data to previous months' data. This is done in two steps because it is clearer for the table's reader, if the subject numbers are each on their own row. Referring back to Table 3 – each subject is given their own row, making it clear when they met the specified criteria. This is not applicable to the frequency rows, therefore, these are not merged on the EVER variable. The two parts of the table are, therefore, merged separately and then set back together.

DATA CUMULATIVE_STATS;
  MERGE STATS_SBJ (WHERE=(COL1S = INT(COL1S)))
    COL3 (WHERE=(COL1S = INT(COL1S)) DROP=EVER)
    COL4 (WHERE=(COL1S = INT(COL1S)) DROP=EVER)
    COL5 (WHERE=(COL1S = INT(COL1S)) DROP=EVER)
    COL6 (WHERE=(COL1S = INT(COL1S)) DROP=EVER);
  BY COL1S CATE1;
RUN;

DATA CUMULATIVE_SBJ;
  MERGE STATS_SBJ (WHERE=(COL1S ^= INT(COL1S)))
    COL3 (WHERE=(COL1S ^= INT(COL1S)))
    COL4 (WHERE=(COL1S ^= INT(COL1S)))
    COL5 (WHERE=(COL1S ^= INT(COL1S)))
    COL6 (WHERE=(COL1S ^= INT(COL1S)));
  BY COL1S CATE1 EVER;
RUN;

DATA CUMULATIVE; SET CUMULATIVE_STATS CUMULATIVE_SBJ;
PROC SORT; BY COL1S;
RUN;

Step 4: The month and year for each column are stored in macro variables that can be used to automatically generate the column titles.

DATA _NULL_;
  call symput('col1head','Category');
  call symput('col2head',PUT("&SYSDATE."D - 30, MONYY5.));
  call symput('col3head',PUT("&SYSDATE."D - (30*2), MONYY5.));
  call symput('col4head',PUT("&SYSDATE."D - (30*3), MONYY5.));
  call symput('col5head',PUT("&SYSDATE."D - (30*4), MONYY5.));
  call symput('col6head',PUT("&SYSDATE."D - (30*5), MONYY5.));
RUN;

This CUMULATIVE dataset can now be printed for presentation, and the macro variables COL1HEAD-COL6HEAD will contain the appropriate dates for the column headings.
Conclusion

Creating a summary table does provide information to the reader. But by adding in a bit more information, a genuinely useful tool is developed to facilitate the clinical review of study data. The reader of these tables now has the summary statistics, and can also quickly see which subjects are involved, and how this compares to previous versions of the table.

Contact Information
Your comments and questions are valued and encouraged. Contact the author at:
  Kim Truett
  11585 Jones Bridge Road; Suite 420-115
  Alpharetta, GA 30022
  770.372.0989
  Email: KCTData@comcast.net

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.